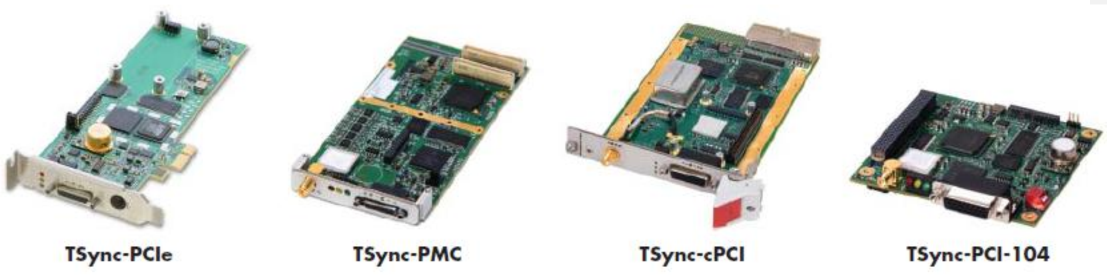
dBus-level timing board (TSync and TPRO/TSAT families) support assistance document

Note: This document contains both **"Public Information"** and **"Internal Use Only Information"** (do not release this document in its entirety to anyone outside of Orolia)

Table of Contents

Login for our Tech Support PC:

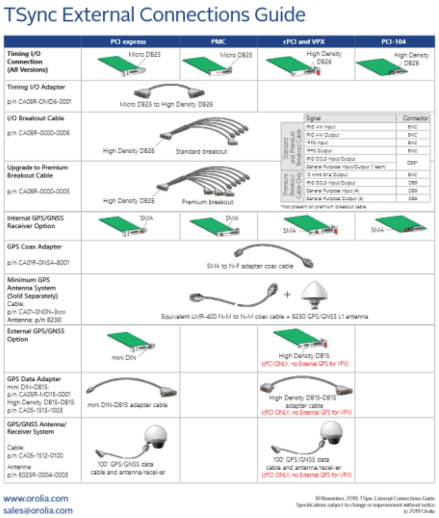
User name: root   Password: abc123



TSync family of bus-level timing boards

Available TSync bus Connectors

- Refer to **TSync External Connections Guide** (on our website): https://www.orolia.com/sites/default/files/document-files/TSync_External_Connections_Guide_Letter_v2_11-19-19.pdf



TSync External Connections Guide

	PCI express	PMC	cPCI and VPX	PCI-104
Timing I/O Connection (All Versions)	Micro DB25	Micro DB25	High Density DB26	High Density DB26
Timing I/O Adapter p/n CA08R-DMD6-0001	Micro DB25 to High Density DB26			
I/O Breakout Cable p/n CA08R-0000-0006	High Density DB26 Standard breakout			
Upgrade to Premium Breakout Cable p/n CA08R-0000-0005	High Density DB26 Premium breakout			
Internal GPS/GNSS Receiver Option	SMA	SMA	SMA	SMA
GPS Coax Adapter p/n CA01R-0NSA-8001	SMA to N-F adapter coax cable			
Minimum GPS Antenna System (Sold Separately) Cable: p/n CA01-0N0N-3xxx Antenna: p/n 8230	Equivalent LMR-400 N-M to N-M coax cable + 8230 GPS/GNSS L1 antenna			
External GPS/GNSS Option	mini DIN		High Density DB15 (cPCI ONLY, no External GPS for VPX)	
GPS Data Adapter mini DIN-DB15: p/n CA05R-MD15-0001 High Density DB15-DB15: p/n CA05-1515-1003	mini DIN-DB15 adapter cable		High Density DB15-DB15 adapter cable (cPCI ONLY, no External GPS for VPX)	
GPS/GNSS Antenna/ Receiver System Cable: p/n CA06-1512-0100 Antenna: p/n E025R-0004-0003	100' GPS/GNSS data cable and antenna/receiver		100' GPS/GNSS data cable and antenna/receiver (cPCI ONLY, no External GPS for VPX)	

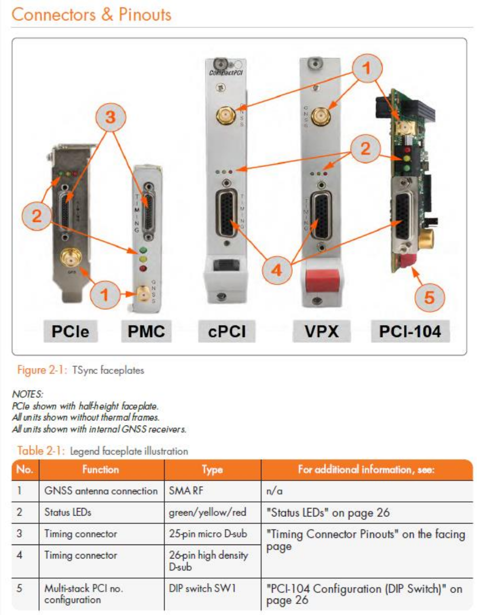
	Signal	Connector
Standard and Premium Breakout Cable	IRIG AM Input	BNC
	IRIG AM Output	BNC
	1PPS Input	BNC
	1PPS Output	BNC
	IRIG DCLS Input/Output	DB9*
	General Purpose Input/Output (1 each)	
Premium Breakout Cable Only	10 MHz Sine Output	BNC
	IRIG DCLS Input/Output	DB9
	General Purpose Input (4)	DB9
	General Purpose Output (4)	DB9

*Not present on premium breakout cable

www.orolia.com
sales@orolia.com

18 November, 2019. TSync External Connections Guide
Specifications subject to change or improvement without notice
© 2019 Orolia

(From the Tsync series user manual)


3
2
1
1
2
4
5
PCIe
PMC
cPCI
VPX
PCI-104

Connectors & Pinouts

Figure 2-1: TSync faceplates

NOTES:
PCIe shown with half-height faceplate.
All units shown without thermal frames.
All units shown with internal GNSS receivers.

Table 2-1: Legend faceplate illustration

No.	Function	Type	For additional information, see:
1	GNSS antenna connection	SMA RF	n/a
2	Status LEDs	green/yellow/red	"Status LEDs" on page 26
3	Timing connector	25-pin micro D-sub	"Timing Connector Pinouts" on the facing page
4	Timing connector	26-pin high density D-sub	
5	Multi-stack PCI no. configuration	DIP switch SW1	"PCI-104 Configuration (DIP Switch)" on page 26

TsyncE-PCIe (TSync-PCIe-002 and TSync-PCIe-012) boards with external GPS/GNSS receiver

Per presentation from Emmanuel (~Dec 2020)

Obsolete the PCIe-0x2 variants which have the Acutime external GPS antenna, except for Schneider Electric (special P/N)

Three (3) digit Model Number configuring scheme for TSyncs (TSync-xxxx -XYZ) / TSyncI/TSyncE

From TSync data sheet:

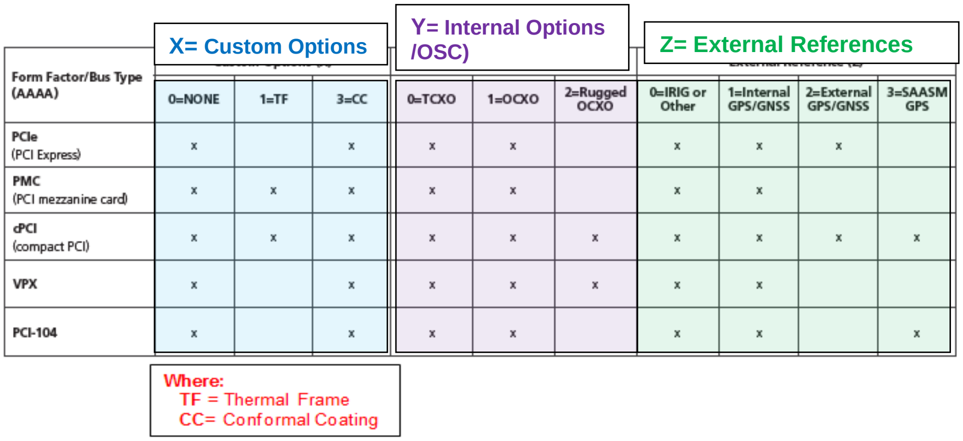
TSync-PCIe

TSync-PMC

TSync-cPCI

TSync-VPX

TSync-PCI-104



Form Factor/Bus Type (AAAA)	X= Custom Options			Y= Internal Options /OSC)			Z= External References			
	0=NONE	1=TF	3=CC	0=TCXO	1=OCXO	2=Rugged OCXO	0=IRIG or Other	1=Internal GPS/GNSS	2=External GPS/GNSS	3=SAASM GPS
PCIe (PCI Express)	x		x	x	x		x	x	x	
PMC (PCI mezzanine card)	x	x	x	x	x		x	x		
cPCI (compact PCI)	x	x	x	x	x	x	x	x	x	x
VPX	x		x	x	x	x	x	x		
PCI-104	x		x	x	x		x	x		x

Where:
TF = Thermal Frame
CC= Conformal Coating

Model TSync-PCIe - X Y Z

X is Custom Options

0= None
1= Thermal Frame (TF)
2= not applicable
3= Conformal Coat (CC)

Y is Oscillator

0= TCXO
1= OCXO (5PPB)
2= Ruggedized OCXO (1PPB)

Z is GNSS receiver/exterenal references

0= No GPS/GNSS functionality (IRIG input only)
1= Internal/onboard commercial receiver
2= External commercial receiver (receiver is located inside the antenna housing)
3= SAASM receiver

Note that not all selections above are available for all board types (for example, "Thermal frame" is not available with TSync-PCIe).

TSyncI boards vs TSyncE boards

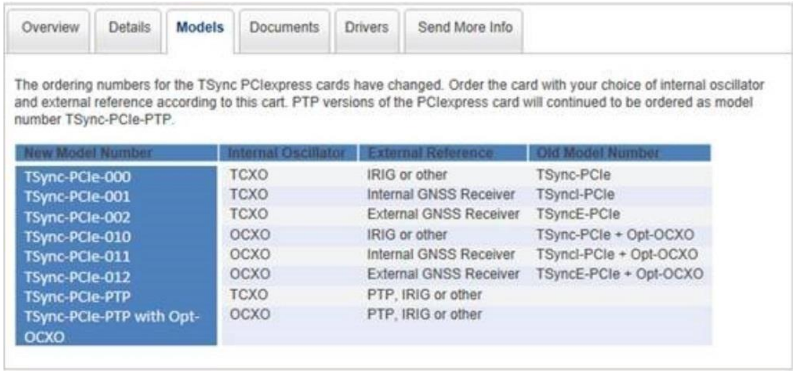
- Z= 1 (TSyncI board)

- Z=-2 (TSyncE board)

Specific examples

Model	Configuration		
TSync-PCIe-000	**0**= no Thermal Frame or Conf Coat	**0**= TCXO	**0**=No GPS/GNSS functionality (IRIG input only)
TSync-PCIe-002	**0**= no Thermal Frame or Conf Coat	**0**= TCXO	**2**=External commercial receiver (receiver is located inside the antenna housing)
TSync-PCIe-011	**0**= no Thermal Frame or Conf Coat	**1**= OCXO.	**1**=Internal/onboard commercial receiver
TSync-PCIe-012	**0**= no Thermal Frame or Conf Coat	**1**= OCXO	**2**=External commercial receiver (receiver is located inside the antenna housing)
TSAT-VPX-100	**1**= Thermal Frame	**0**= TCXO	**0**= No GPS/GNSS functionality (IRIG input only)

below is excerpt from our website



The ordering numbers for the TSync PCIexpress cards have changed. Order the card with your choice of internal oscillator and external reference according to this cart. PTP versions of the PCIexpress card will continued to be ordered as model number TSync-PCIe-PTP.

New Model Number	Internal Oscillator	External Reference	Old Model Number
TSync-PCIe-000	TCXO	IRIG or other	TSync-PCIe
TSync-PCIe-001	TCXO	Internal GNSS Receiver	TSyncI-PCIe
TSync-PCIe-002	TCXO	External GNSS Receiver	TSyncE-PCIe
TSync-PCIe-010	OCXO	IRIG or other	TSync-PCIe + Opt-OCXO
TSync-PCIe-011	OCXO	Internal GNSS Receiver	TSyncI-PCIe + Opt-OCXO
TSync-PCIe-012	OCXO	External GNSS Receiver	TSyncE-PCIe + Opt-OCXO
TSync-PCIe-PTP	TCXO	PTP, IRIG or other	
TSync-PCIe-PTP with Opt-OCXO	OCXO	PTP, IRIG or other	

TCXO

- TSync-PCIe-0**00** = TCXO and no GPS receiver (1191-1001-0600)
- TSync-PCIe-0**01** = TCXO and Internal GPS receiver (1191-2001-0600)
- TSync-PCIe-0**02** = TCXO and external GPS receiver (1191-3001-0600) (No longer available for purchase)

OCXO

- TSync-PCIe-0**10** = OCXO and no GPS receiver (1191-1002-0600)
- TSync-PCIe-0**11** = OCXO and Internal GPS receiver (1191-2002-0600)
- TSync-PCIe-0**12** = OCXO and external GPS receiver (1191-3002-0600) (No longer available for purchase)

PTP (no longer available for purchase)

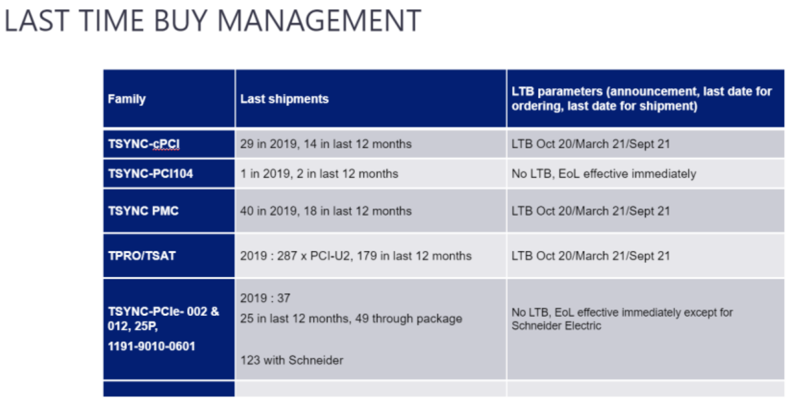
- TSync-PCIe-0**00** = TCXO and PTP (1191-6001-0600)
- TSync-PCIe-0**10** = OCXO and no GPS receiver (1191-6002-0600)

TSync series discontinuance: Last Time Buy/TSync-PCIe replacements

A) Updated info, as of Dec 2020 (from a presentation from Emmanuel)

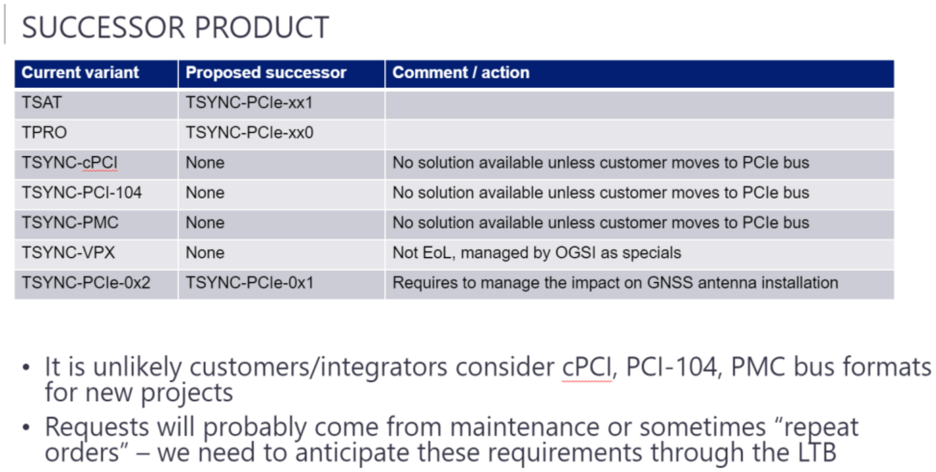
we will obsolete the TSAT/TPRO as well as some non PCIe variants, offering a last time buy opportunity to customers, starting in Oct. 2020, with deliveries ending Sept 2021. TSYNC-PCIe and VPX will be the only offered variants afterwards.

(as of Dec 2020)



LAST TIME BUY MANAGEMENT

Family	Last shipments	LTB parameters (announcement, last date for ordering, last date for shipment)
TSYNC-cPCI	29 in 2019, 14 in last 12 months	LTB Oct 20/March 21/Sept 21
TSYNC-PCI104	1 in 2019, 2 in last 12 months	No LTB, EoL effective immediately
TSYNC PMC	40 in 2019, 18 in last 12 months	LTB Oct 20/March 21/Sept 21
TPRO/TSAT	2019 : 287 x PCI-U2, 179 in last 12 months	LTB Oct 20/March 21/Sept 21
TSYNC-PCIe- 002 & 012, 25P, 1191-9010-0601	2019 : 37 25 in last 12 months, 49 through package 123 with Schneider	No LTB, EoL effective immediately except for Schneider Electric



SUCCESSOR PRODUCT

Current variant	Proposed successor	Comment / action
TSAT	TSYNC-PCIe-xx1	
TPRO	TSYNC-PCIe-xx0	
TSYNC-cPCI	None	No solution available unless customer moves to PCIe bus
TSYNC-PCI-104	None	No solution available unless customer moves to PCIe bus
TSYNC-PMC	None	No solution available unless customer moves to PCIe bus
TSYNC-VPX	None	Not EoL, managed by OGSI as specials
TSYNC-PCIe-0x2	TSYNC-PCIe-0x1	Requires to manage the impact on GNSS antenna installation

- It is unlikely customers/integrators consider cPCI, PCI-104, PMC bus formats for new projects
- Requests will probably come from maintenance or sometimes "repeat orders" – we need to anticipate these requirements through the LTB

Lifecycle/end of life statement

- Refer to statement: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\lifecycle- end of life letter

~~Timekeeper software with a TSync series timing board~~

- ~~Refer to the Timekeeper software section towards the beginning of the VelaSync/Geo tech assistance document for info on using this software with a TSnc board.I:\Customer Service\1- Cust Assist documents\VelaSyncAndGeoCustAssist.pdf~~

Non-Spectracom "TSync" software program

- Product from another vendor using the "TSync" name.

Email from Denis Reilly (7/25/12)
An Indian company is selling software called TSync....
http://www.maniworld.com/tsync10.html

Email from Dave Lorah to a "customer" (not A Spectracom Customer)
Good Day Sir,
I believe you are contacting the wrong company for support on this product. Please see http://www.maniworld.com/tsync10.html for more information.

TSync-PTP (TSync-PCIe-PTP boards) are now discontinued

- TSync-PTP boards were discontinued in Sept 2014 (Other non-PTP TSync board are still available)

MTBF for TSync boards

- Refer to "**MTBF/MTTR**" in the custserviceassist doc: I:\Customer Service\1- Cust Assist documents\CustomerServiceAssistance.pdf

Agency Approvals/Compliance (TUV/UL/CE Declaration of Conformity/EMC/EMI/ESD)

Agency approval

TUV and UL: The TSync-PCIe boards are not TUV or UL approved.

UL approval/certification (UL 60950)

Q. Could you please tell me if this item complies to any safety specifications such as UL60950
A. Email from Dave S: There is a CE safety certification for the TSync, but not UL. (also in \\rocfnp02\idrive\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\TSYNC-PCIe\Agency-CE approval

CE Approval/CE Declaration of Confirmity (DoC)

- TSync timing boards are CE approved/ CE Commpliant (Refer to the TSync-PCIe data sheet for confirmation)
  - The TSync boards are CE compliant, so a Declaration of Conformity is available upon request.
  - The "**Declaratiion of Conformity**" document (**DEC-CONF-TSync"**) is stored in Arena at: https://app.bom.com/items/detail-spec?item_id=1217209149&version_id=10453630238

EMI/EMC requirements

Questions below from Tony Diflorio's customer- responses in red from Tom Richardson:
What I am trying to do is evaluate the unit to make sure we can integrate it into our system and I am mostly interested in the mechanical aspect of the unit. More specifically, there are EMI and EMC requirements we need to verify (MIL-STD-461) as well as some materials specifications.

Do you have any information regarding conducted and radiated emissions? See the DOC (SecureSync Declaration of Conformity). We do not and cannot meet Mil-Std 461…

Does the unit have any EMI power line filters incorporated? Yes.

Are ALL electrical, electronic, and electromagnetic parts used in the unit new? Ans: yes.

Are any mechanical or structural parts made of recycled plastics or metals? Not to my knowledge

Terms associated with TSync boards

µC (Micro Computer) this refers to the computer in which the TSync-PCIe board is installed.

HOST this refers to the computer in which the TSync-PCIe board is installed.

KTS refers to the "Kramden Timing System"

- This is the Spectracom internal code name for timing chip Spectracom uses in SecureSync as well as the TSYNC board. It is an FPGA that actually does all the time and frequency synchronization.

Zero Latency: This refers to time stamping of events

- With reference to the time reads our information mentions zero latency. What does that mean?
- **Email Keith sent to Tony Diflorio (8 Oct 2014)** I clarified this question this morning with Denis.   It's referring to the ability for the TSync board to time stamp up to four simultaneous incoming events. Then after the events have already occurred, the time stamp for that particular event can be read out of the board. Each time an event is received, a time stamp for that event is performed and then its stored in a FIFO buffer to be read out later.

ISO 8 cleanroom requirements

Email from Sylvain to Tim T and Tom Richardson (8 Mar 16): Could you find the information if the Tsync PCIe is compliant with "ISO 8 cleanroom requirements". I think this is linked to the PCB class mainly.

Tom Richardson replied:
Hi Guys,
I found this.
http://www.terrauniversal.com/cleanrooms/iso-classification-cleanroom-standards.php
I don't see how it applies.

GPS 1024 week rollover (such as in 2019)

Refer to the custserviceassist document: I:\Customer Service\1- Cust Assist documents\CustomerServiceAssistance.pdf

Conformal Coating (also “CC” in data sheet)

- Listed as an Option in the Data Sheet.
- For info on Conformal Coating (below), refer to: https://en.wikipedia.org/wiki/Conformal_coating

Conformal coating material is a thin polymeric film which ‘conforms’ to the contours of a printed circuit board to protect the board's components. Typically applied at 25-250 μm.https://en.wikipedia.org/wiki/Conformal_coating - cite_note-What_is_Conformal_Coating-1 It is applied to electronic circuitry to act as protection against moisture, dust, chemicals, and temperature extremes that, if uncoated (non-protected), could result in damage or failure of the electronics to function. When electronics must withstand harsh environments and added protection is necessary, most circuit board assembly houses coat assemblies with a layer of transparent conformal coating rather than potting.

Tin, lead and gold on TSync boards

- Refer to salesforce case 163239 (specifically inquriring about 1191-1001-0600 and the **CA08R-DMSA-E002** cable)

email from Tom Richardson (7 May 18) Tin is in the solder and the metallization (end caps) of the components. Gold is used in the board pad material. There should be no lead in the assemblies but I see she did not ask about that.

Follow-up qestions from same customer (Responses from Tom R, 10 May 18, in red)

Is the solder made up of tin-lead? I would like to confirm whether there is pure tin used on the board. Solder is lead-free. Solder is SAC305, which has tin, silver and copper. No pure tin.

Are the metallization (end caps) of the electronic components attached to the PCB Boards made with pure tin? The metallization on the components is not pure tin.

Also what is thickness of the gold that is used for the PCB board pad? Thank you for your help. The gold is on the PCIe connector, 30 microinches minimum and the component pads which are minimum 1.97 microinches

Conduction Cooling Thermal Frame (also “TF” in TSync datasheet)

- Listed as an Option in the Data Sheet.
- Also known as “Conduction Cooled Card Frame” and “heat frames”
- Provides Conduction Cooling of the PCB assembly
- For applications where air cooling isn’t available (the metal frame moves the heat to the edge of the board)
- For info on Thermal Frame (below), refer to sites such as: https://www.google.com/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&ved=0CB0QFjAAahUKEwjllInB3JzIAhUDNz4KHX-IAgE&url=http%3A%2F%2Fwww.kontron.com%2Fdownloads%2Fwhite_papers%2Fwhitepaper-conduction-cooling-techniques-for-rugged-computers_eng.pdf%3Fproduct%3D88959&v6u=https%3A%2F%2Fs-v6exp1-v4.metric.gstatic.com%2Fgen_204%3Fip%3D66.193.84.98%26ts%3D1443546161494390%26auth%3Dxapq4

r6bq6fk3wgrpsb43acwru5f2lk4%26rndm%3D0.5992074745694829&v6s=2&v6t=1635&usg=AFQjCNFOwSENuuc9CQpsv2Pbh2g_FlOddw&cad=rja

Onboard battery/Battery backed time (for sync to itself after every boot-up)

- Unlike SecureSync, TSync-PCIe boards do not have any batteries onboard. They do not have a Real Time Clock (RTC) like SecureSync has. So they do not need any batteries.
- Self/Self reference allows the board to power-up into sync.

Number of TSync boards that can be installed in a single PC/more than one TSync board installed

- The PCIe driver is set to support up to (8) eight TSync boards in a single system. For all example programs provided, the first parameter passed in is the board instance number, which will identify the card to perform the action on.
- Query the Serial Number of the boards to positively identify which board is which

Q How we can identify the actual physical card were are talking to via the API – do the cards have any kind of hardware ID that we can use so we can be absolutely certain which one we are talking to? Maybe the 0 or 1 is based on the PCIe slot?"

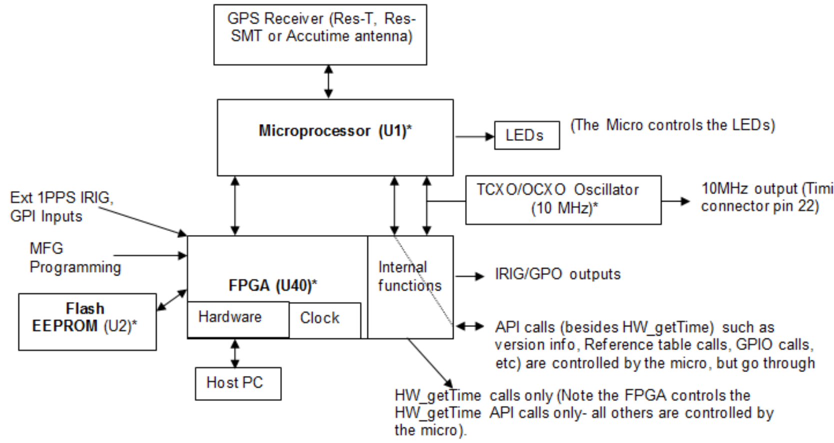
A **Email from Rond Dries (18 Dec 2019)** The cards will be assigned an instance, starting from 0, based on which one comes up first when the system boots.

They can query the card for its serial number to determine which is which, in case that the instances flip for some reason in the system.

For info on querying the Serial Number, refer to: Tsync-PCIe's Model and Serial Number

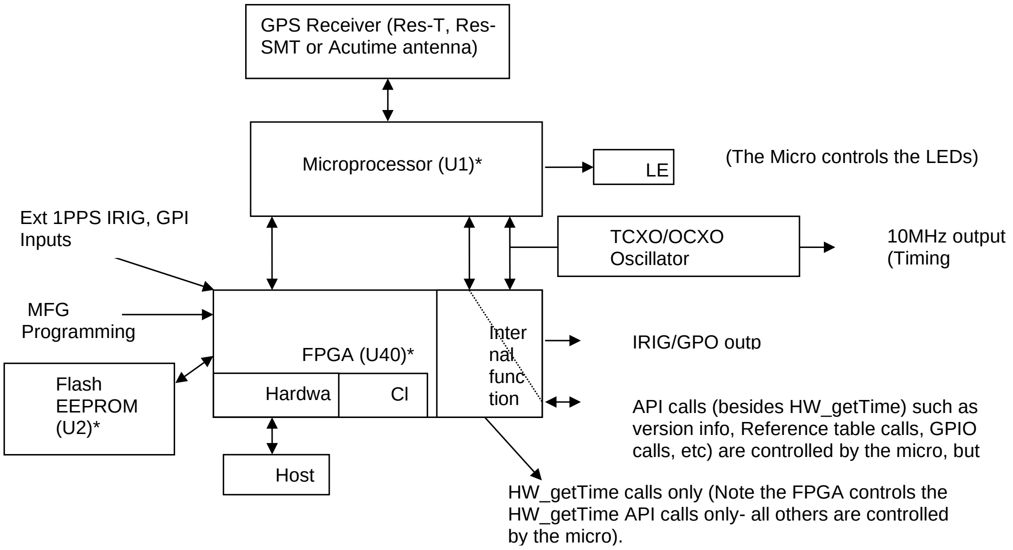
Hardware block diagram (for internal use only)

- Shortcut to TSync-PCIe board schematics (not breakout cables): **(1191** -> **“1191-1000-0200 board files”** -> **”1191-1001-0200)** in Arena: https://app.bom.com/items/detail-spec


GPS Receiver (Res-T, Res-SMT or Accutime antenna)
Microprocessor (U1)*
LEDs
(The Micro controls the LEDs)
TCXO/OCXO Oscillator (10 MHz)*
10MHz output (Timing connector pin 22)
Ext 1PPS IRIG, GPI Inputs
MFG Programming
FPGA (U40)*
Internal functions
Hardware
Clock
Flash EEPROM (U2)*
IRIG/GPO outputs
API calls (besides HW_getTime) such as version info, Reference table calls, GPIO calls, etc) are controlled by the micro, but go through
Host PC
HW_getTime calls only (Note the FPGA controls the HW_getTime API calls only- all others are controlled by the micro).

* denotes item is part of the Kramden Timing System (KTS)

(Just for editing)


GPS Receiver (Res-T, Res-SMT or Acutime antenna)
Microprocessor (U1)*
LE
(The Micro controls the LEDs)
TCXO/OCXO Oscillator
10MHz output (Timing
Ext 1PPS IRIG, GPI Inputs
MFG Programming
FPGA (U40)*
Inter nal func tion
IRIG/GPO outp
Flash EEPROM (U2)*
Hardwa
Cl
API calls (besides HW_getTime) such as version info, Reference table calls, GPIO calls, etc) are controlled by the micro, but
Host
HW_getTime calls only (Note the FPGA controls the HW_getTime API calls only- all others are controlled by the micro).

* denotes item is part of the Kramden Timing System (KTS)

Functional descriptions:

ColdFire Microprocessor (U1):

- Refer to: http://cache.freescale.com/files/32bit/doc/ref_manual/MCF5282UM.pdf
- The Micro controls all API calls (except HW_GetTime) such as version info, Reference table calls, GPIO calls, etc), but these calls go through the FPGA (not directly to/from the micro).
- The Micro controls the GPS receiver operation. The "GR_" API calls for the GPS receiver go through the FPGA to the micro.
- **Mfg:** Freescale
- **Model and P/N:** ColdFire MCF5216

FPGA (U40):

- Refer to: http://www.mouser.com/ds/2/225/LatticeECP2MFamilyHandbook-311154.pdf
- The FPGA handles the communications with the Microprocessor and PCIe bus.
- HW_GetTime calls are direct register reads inside the FPGA (micro not involved with these particuar calls, like it is with all other calls)
- **Mfg**: Lattice
- **Lattice P/N:** LFE2M35E-6FN256I

Flash EEPROM (U2):

- The EEPROM programs and stores info for the FPGA.
- **original model is going EOL (end of life) (reported April, 2019- see more info directly below)**

ECO-001630 (April, 2019- Release new TSync F001 & F002 assemblies due to original model going end of life)

- ECO-1630 (in Arena): https://app.bom.com/changes/detail-summary?change_id=2393305513&

Description: This ECO will release the new F001 & F002 that fixes many outstanding issues and adds new features.

* Create a new F000 assembly, 1191-0000-F002 for OCXO option configurations. (ECR-000142)

* Remove the bracket, 1191-1000-0700, from the F000 assemblies.

* Updated assembly drawings to address DEV-000069.

Notes about Flash EEPROM:

- The EEPROM is reprogrammed during the firmware update process.
- EEPROM stores configuration information.
- A programming issue with the EEPROM will prevent the micro from working.
- A problem with only the micro allows HW_GetTime API calls to still work, but no other API calls will work (try several different API calls to verify micro issue).
- A problem with the FPGA will allow GPS operation to be normal (can still sync and the LEDs will operate normally) but the user won't be able to communicate with the GPS receiver, for instance, because API calls can't get through.
- If the FPGA is not operating, even if the micro is still running, there will be no communication with the micro, so the micro will appear to not be running. However, with the micro still running, the LEDs will still work.

I was just speaking to one of our timing board hardware engineers about the symptoms the TSync-PCIe board is exhibiting (specifically, only the HW_GetTime APIs working but no other APIs are working).

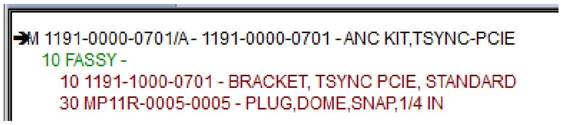
Based on these symptoms, it appears there may have been a problem with the firmware upgrade to the EPROM that is associated with the microprocessor. If the EEPROM is “corrupted” in any way, the microprocessor can’t run.   The FPGA is likely still running as it handles the hardware time API (as you reported).   However, all other APIs are handled by the microprocessor, which is apparently not running, because all other API calls are failing.

The EEPROM can become “corrupted” if the wrong upgrade bundle is applied first, then another bundle is applied over the top of the wrong update bundle. You chose the correct upgrade bundle to use (TCXO with no GPS), but either another bundle was used first, or more likely, something happened to the EEPROM during the firmware upgrade process.

With these symptoms, it looks like this TSync board will need to be returned for a reprogramming of the EEPROM.   This is especially evident since the issue cleared with the spare board installed in its place! Based on the symptoms, I don’t think installing this board in another CPU will make any difference. It will likely operate the same way in the other CPU, confirming this explanation.

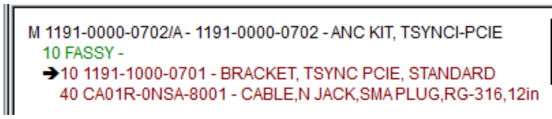
TSync Ancillary kits

TSync with no GPS receiver (1191-0000-0701)



➔M 1191-0000-0701/A - 1191-0000-0701 - ANC KIT,TSYNC-PCIE
10 FASSY -
10 1191-1000-0701 - BRACKET, TSYNC PCIE, STANDARD
30 MP11R-0005-0005 - PLUG,DOME,SNAP,1/4 IN

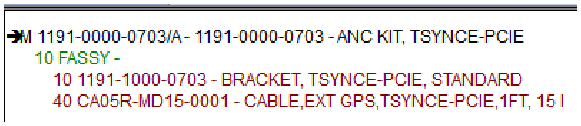
TSyncI-PCIe (1191-0000-0702)



M 1191-0000-0702/A - 1191-0000-0702 - ANC KIT, TSYNCI-PCIE
10 FASSY -
➔10 1191-1000-0701 - BRACKET, TSYNC PCIE, STANDARD
40 CA01R-0NSA-8001 - CABLE,N JACK,SMA PLUG,RG-316,12in

12 inch GPS coax cable (SMA to N adapter)

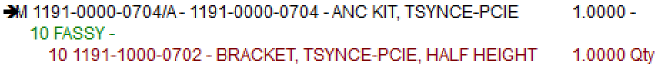
TSyncE-PCIe (1191-0000-0703)



➔M 1191-0000-0703/A - 1191-0000-0703 - ANC KIT, TSYNCE-PCIE
10 FASSY -
10 1191-1000-0703 - BRACKET, TSYNCE-PCIE, STANDARD
40 CA05R-MD15-0001 - CABLE,EXT GPS,TSYNCE-PCIE,1FT, 15 I

TSyncE-PCIe, SP364 1191-0000-0704

SP364: For Invensys (from the Specials database- "Add cable, update software and FPGA to emulate an PCI-U with FXA option. Refer to ECN 2341")



➔M 1191-0000-0704/A - 1191-0000-0704 - ANC KIT, TSYNCE-PCIE 1.0000 -
10 FASSY -
10 1191-1000-0702 - BRACKET, TSYNCE-PCIE, HALF HEIGHT 1.0000 Qty

P/N 07013049 (SMA Male to TNC Female adapter)

Google purchased (16) SMA Male to TNC Female adapter with our P/N 07013049. Reference Order Number T-39734. Not sure why these were purchased for use with the TSync-PCIe boards.

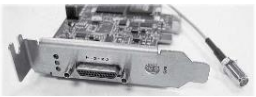
Mounting Brackets for TSync boards (half height/full height)

A) Bracket for Tsyncl boards (internal/onboard GPS receiver)

Summary:

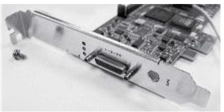
- **TSyncI** boards ship from the factory with the **half height bracket** (P/N 1191-1000-0700) attached
- One **Standard (Full height) bracket** (P/N 1191-1000-0701) is included in the Anc Kit.
- Refer to the TSync user manual for info on swapping out the bracket.

1. **Half height bracket (P/N 1191-1000-0700) attached to TSync board:**



- Bracket in Arena: https://app.bom.com/items/detail-spec?item_id=1202839888&version_id=10221274938

2. **AvailableStandard (Full height) bracket (P/N 1191-1000-0701) supplied in Anc Kit:**



- This bracket in Arena: https://app.bom.com/items/detail-spec?item_id=1202839889&version_id=10221274948
- One full height bracket is included in the Ancillary kit (either **1191-0000-0701** or **1191-0000-0702**)

Anc kit P/N 1191-0000-0701 for no-GPS receiver boards (IRIG input only)

Note: This Anc kit includes a hole plug to place in the bracket hole, where the GPS connector is normally located

In Salesforce: https://orolia.my.salesforce.com/01t0h000005e7g9?srPos=3&srKp=01t

In Arena: https://app.bom.com/items/detail-spec?item_id=1202839883&version_id=10299429028&orb_msg_single_search_p=1

Anc kit P/N 1191-0000-0702 for GPS board installed (GPS and IRIG input):

In Salesforce: https://orolia.my.salesforce.com/01t1A000004sodK?srPos=1&srKp=01t

In Arena: https://app.bom.com/items/detail-spec?item_id=1202839884&version_id=10299429038&orb_msg_single_search_p=1

B) Bracket for TsyncE boards (external GPS receiver-receiver is in the Acutime antenna)

Summary:

- **TSyncE** boards ship from the factory with the half height bracket (P/N **1191-1000-0702**) attached
- One Standard (Full height) bracket (P/N **1191-1000-0703**) is included in the Anc Kit.

3. **Half height bracket (P/N 1191-1000-0702) attached to TSync board:**

4.

   - This bracket in Arena: https://app.bom.com/items/detail-spec?item_id=1202833274&version_id=10298086718
   - Refer to the TSync user manual for info on swapping out the bracket.

5. **Available Standard (Full height) bracket (P/N 1191-1000-0703) supplied in Anc Kit**

   - This bracket in Arena: **https://app.bom.com/items/detail-spec?item_id=1202839890&version_id=10221274958**
   - A half height bracket is included in the Ancillary kit (Anc kit P/N is **1191-0000-0703**) Note the P/N for the bracket itself is **1191-1000-0702**
     - Anc kit (1191-0000-0703) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202839885&version_id=10299429048&orb_msg_single_search_p=1
   - Refer to the TSync user manual for info on swapping out the bracket.

C) Bracket for TSync-PTP boards

- All TSync-PTP boards can us only one specific bracket (P/N **1206-1000-0700**) It is a full-height bracket

Low-profile bracket for low profile applications using TSync-PTP boards.

- The PTP boards do not have a low profile bracket available, due to the RJ-45 connector and its associated small circuit board
- There are no alternate brackets available for the TSync-PTP boards due to the Ethernet jack.
- The TSync-PTP board was not intended for low profile applications. However, the low profile bracket for the non-PTP boards CAN be used with the TSync-boards.   HOWEVER, the customer is on their own with custom mounting the RJ-45 connector.

Q Regarding the Tsync-PCIe-PTP card, it’s now going in a 2U server so we need a low profile PCI bracket for it. The plan is to move the RJ45 module off the PTP PCI card bracket to a secondary bracket.

Keith’s response To begin, the TSync-PTP timing board was not intended for low profile applications. However, it can be, but the small circuit board for the RJ-45 connector needs to be custom-mounted and you will be on your own for this.. We do not have any other means to mount the RJ-45 connector and its associated small PCB board, besides the full height bracket that comes attached to the TSync board.

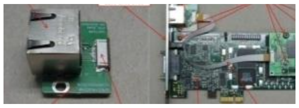
Q. Will the Tsync-PCIe-001 low profile PCI bracket or other Spectracom LP bracket fit on the PTP card if the RJ45 is removed from the PTP?

Keith’s response: Yes it will! The low profile bracket (we only have one low profile bracket) for the non-PTP variant of the

Tsync boards will work with the TSync-PTP, with the exception of the RJ-45 connector and its associated circuit board. The rest of the TSync panel (such as the Timing connector) besides the PTP connector in the end is exactly the same on all of the variants of the TSync board. So the low profile bracket can be attached to the PTP board as long as the RJ45 connector is relocated elsewhere.

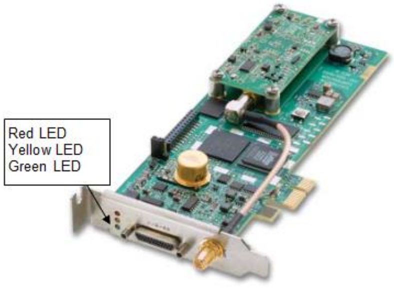
Q. If so are these low profile PCI brackets available from Spectracom for purchase?
Keith's response: The low profile brackets are not currently on our price list for purchase. How many of these low profile brackets would you need? If you only need a couple, I can just send them to you, no cost. If you need more than this, Mike can work with our Product Manager to make them available to you for purchase.

Q. Is the PTP RJ45 module and cable assembly easily removed from the PTP PCI card bracket?
Keith's response: The RJ45 connector is mounted onto its own small circuit board, which is separate from the main circuit board.   There are just two small screws that attach this circuit board to the TSync bracket. Below are two photos for your reference (excerpted from a Process Detail, so just disregard the red lines).

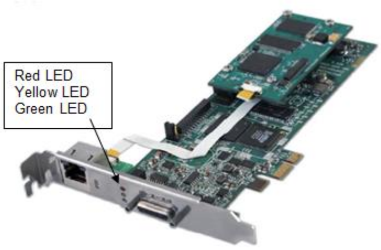


Q. If not, is the RJ45 module and cable assembly available from Spectracom for purchase?
Keith's response: We don't offer the RJ45 and cable assembly as a separate item. Another RJ45 connector shouldn't be needed and as long as the ribion cable is long enough for your application/custom mounting, another cable shouldn't be needed either. Both of these components can be "re-used".

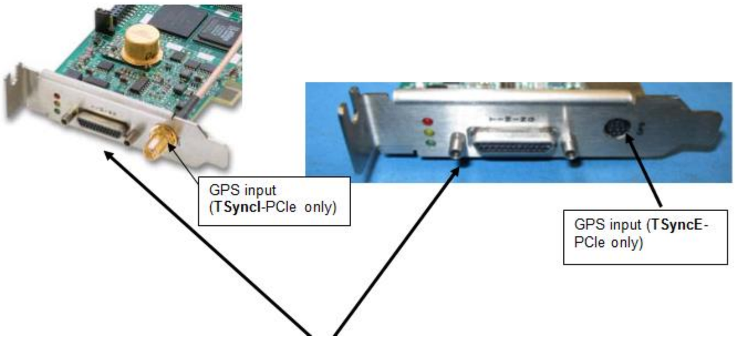
TSync-PCIe timing boards



TSyncl-PCIe



TSync-PTP



3.1.1 *Timing Connector Pinout*

Table 3.1—Pinout

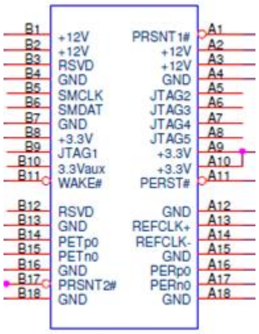
Pin	Signal	Pin	Signal
1	GPIO Output 2	14	GPIO Output 3
2	Ground	15	Ground
3	GPIO Output 0	16	GPIO Output 1
4	GPIO Input 2	17	GPIO Input 3
5	Ground	18	Ground
6	GPIO Input 0	19	GPIO Input 1
7	External 1PPS Input	20	1PPS Output
8	Ground	21	Ground
9	IRIG AM Output	22	10MHz Output
10	IRIG AM Input +	23	Ground
11	IRIG AM Input -	24	IRIG DCLS Input -
12	IRIG DCLS Output -	25	IRIG DCLS Input +
13	IRIG DCLS Output +		

TSync-PCIe Shortcuts/Links

- **Shortcut to user manual (1191-5000-0050) in Arena:** https://app.bom.com/items/detail-spec?item_id=1202833276&version_id=10221277928&orb_msg_Single_Search_p=1&redirect_Seqno=5860862847
- **Shortcut to newer combined TSync family driver guide (1219-5001-0050) in Arena:** https://app.bom.com/files/detail-summary?file_master_id=1234037277&file_id=1745186948
- **Shortcut to Application Programmers Manual** (1191-5002-0050): I:\New Released\Manuals\1191-xxxx-xxxx
- **Link to board schematic (1191** -> **"1191-0000-F001** -> **and it's under "1191-1000-0200") in Arena**: https://app.bom.com/items/detail-spec?item_id=1202839887&version_id=10221218578
- **Link to Tsync-PTP Process detail ("1191-6002-PD) in Arena**: https://app.bom.com/items/detail-spec?item_id=1202838851&version_id=10212782618
- **Shortcut to data sheet:** I:\Company Wide\My Box Files\Marketing\Spectracom Datasheets
- **Shortcut to drivers CD (1219-5003-6001) in Arena:** https://app.bom.com/files/detail-summary?file_master_id=1234039656&file_id=1745191706
- **Shortcut to firmware and driver version upgrades in custservice:** I:\Customer Service\PSB, PSP software updates\TSYNC boards
- **Shortcut to PCIe specs:** I:\Engineering\Specs and Standards\PCISIG

HOST PCIe BUS interface

PCIe bus connector pin-out information



Card Edge 18x2

PCI Generation/PCIe bus speeds supports

- Tsync-PCIe boards are a Generation 1x (Gen1x) device
- **Data rate** is as set by Gen 1 standards (250 MB/s)
- **Transfer rate** is as set by Gen 1 standard (2.5 GT/s. Note: GT/S stands for gigatransfers per second)

As of at least Oct 2014 – the TSync-PCIe is a 1x (1.1) PCIe board. It is not 2x, 4x, 8x, 16x or 32x. It is also a 1.x compliant board. It is not 2.x or 3.x compliant.

Email from Dave Sohn- “The PCIe 2.x spec motherboards are required to be fully backwards compatible with PCIe 1.x cards. If their motherboard is not following the specification I’m not sure what recourse we have. We are not PCIe 2.x compliant. Are they trying to put the card into a graphics card slot? I’ve read about incompatibilities when using non-graphics cards in the x16 graphics card slots. Otherwise, according to the spec, a x1 card is allowed to operate in any width slot.”

Below excerpt about 1.0, 2.0, 3.0 is from http://en.wikipedia.org/wiki/PCI_Express#PCI_Express_1.0

PCI Express 1.0

In 2004, Intel introduced PCIe 1.0, with a data rate of 250 MB/s and a transfer rate of 2.5 GT/s.

[edit] PCI Express 2.0

PCI-SIG announced the availability of the PCI Express Base 2.0 specification on 15 January 2007.[9] The PCIe 2.0 standard doubles the per-lane throughput from the PCIe 1.0 standard's 250 MB/s to 500 MB/s. This means a 32-lane PCI connector (x32) can support throughput up to 16 GB/s aggregate. The PCIe 2.0 standard uses a base clock speed of 5.0 GHz, while the first version operates at 2.5 GHz.

PCIe 2.0 motherboard slots are fully backward compatible with PCIe v1.x cards. PCIe 2.0 cards are also generally backward compatible with PCIe 1.x motherboards, using the available bandwidth of PCI Express 1.1. Overall, graphic cards or motherboards designed for v 2.0 will be able to work with the other being v 1.1 or v 1.0.

The PCI-SIG also said that PCIe 2.0 features improvements to the point-to-point data transfer protocol and its software architecture.[10]

In June 2007 Intel released the specification of the Intel P35 chipset which supports only PCIe 1.1, not PCIe 2.0.[11] Some people may be confused by the P35 block diagram which states the Intel P35 has a PCIe x16 graphics link (8 GB/s) and 6 PCIe x1 links (500 MB/s each).[12] For simple verification one can view the P965 block diagram which shows the same number of lanes and bandwidth but was released before PCIe 2.0 was finalized.[original research?] Intel's first PCIe 2.0 capable chipset was the X38 and boards began to ship from various vendors (Abit, Asus, Gigabyte) as of October 21, 2007.[13] AMD started supporting PCIe 2.0 with its RD700 chipset series and nVidia started with the MCP72.[14] The specification of the Intel P45 chipset includes PCIe 2.0.

[edit] PCI Express 2.1

PCI Express 2.1 supports a large proportion of the management, support, and troubleshooting systems planned to be fully implemented in PCI Express 3.0. But, the speed is the same as PCI Express 2.0.

[edit] PCI Express 3.0

In August 2007, PCI-SIG announced that PCI Express 3.0 will carry a bit rate of 8 gigatransfers per second. The spec will be backwards-compatible with existing PCIe implementations and a final spec is due in the second quarter of 2010.[15]

New features for PCIe 3.0 specification include a number of optimizations for enhanced signaling and data integrity, including transmitter and receiver equalization, PLL improvements, clock data recovery, and channel enhancements for currently supported topologies.[16]

Following a six-month technical analysis of the feasibility of scaling the PCIe interconnect bandwidth, PCI-SIG's analysis found out that 8 gigatransfers per second can be manufactured in mainstream silicon process technology, and can be deployed with existing low-cost materials and infrastructure, while maintaining full compatibility (with negligible impact) to the PCIe protocol stack.

PCIe 2.0 delivers 5 GT/s but employed an 8b/10b encoding scheme which took 20 percent overhead on the overall raw bit rate. By removing the requirement for the 8b/10b encoding scheme, and replacing it with a 128b/130b encoding scheme with only ~1.5 percent overhead,[17] PCIe 3.0's 8 GT/s bit rate effectively delivers double PCIe 2.0 bandwidth.

According to an official press release by PCI-SIG on 8 August 2007:

"The final PCIe 3.0 specifications, including form factor specification updates, may be available by late 2009, and could be seen in products starting in 2010 and beyond."[18]

As of January 2010, the release of the final specifications has been delayed until Q2 2010.[19] PCI-SIG expects the PCIe 3.0 specifications to undergo rigorous technical vetting and validation before being released to the industry. This process, which was followed in the development of prior generations of the PCIe Base and various form factor specifications, includes the corroboration of the final electrical parameters with data derived from test silicon and other simulations conducted by multiple members of the PCI-SIG.

PICMG 1.3 (also known as SHB Express)

Email from Denis Reilly regarding a customer having an issue with time stamping when PC is connected to a network (14 Nov 2014) Just so you guys understand, they are using PICMG 1.3 (also known as SHB Express), which is a scheme where there is a main processor board that plugs into a special slot on a backplane board. Other cards plug into normal slots on this backplane board as well.

Issues with timestamping when PC is connected to a network (works fine with Ethernet port not connected) OR if other PCIe devices affect the operation of the Tsync board

- Network Interface Cards (NICs) are usually PCIe-based, just like the TSync board
- Due to topography of the PCIe bus/slots, the NIC is likely affecting the TSync board.
- Get the specs of the computer (Manufacturer/Model) to look at the system to see if the NIC and TSync board are on the same PCIe bus.

Email from Denis Reilly regarding a customer having an issue with time stamping when PC is connected to a network (14 Nov 2014) Just so you guys understand, they are using PICMG 1.3 (also known as SHB Express), which is a scheme where there is a main processor board that plugs into a special slot on a backplane board. Other cards plug into normal slots on this backplane board as well.

It may make a difference which backplane / sister board they are connected to, since we need to know which slots they are using and what they might be connected to on the main processor card.

PCI to PCIe adapter (for installing a TSync board into a system with no available PCIe slots)

Email Keith sent to Edouard (21 May 2020)

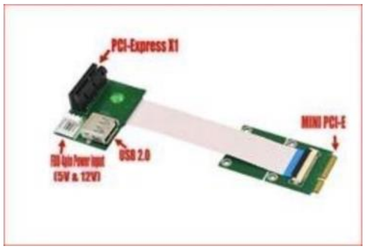
Though we don’t offer an adapter for this, fortunately I recently had a customer find one and he shared with us his success 😊!

Below is the email thread, with the most recent email directly below:

Hello Gentlemen,

Based on the customers’ requirements, this version on the Tsync card was required.

I figured as much that the Tsync could not handle the 5V. We changed the +5 input and tapped in to 3.3V off the backplane. I’m happy to say that the Tsync card is working with the mini-PCIe to PCIe adapter. Mounting the card and this adapter is a challenge and we are having a special mounting bracket manufactured by a local machine shop today.



Thanks,
Peter

From: Keith Wing [mailto:Keith.Wing@orolia.com]
Sent: Monday, March 16, 2020 6:20 AM
To: Peter Pantic
Cc: David Lorah
Subject: Tsync-PCI adapter

Hi Peter,

Good morning.   I hope you had a nice weekend!

Thanks for your question last week about the adapter providing 5vdc to the TSync board’s 3.3vdc input.

I have confirmed with our Senior Timing board Engineer that “We can't accept 5VDC on the PCIe bus. It will likely damage the components.”

In case you continue having trouble finding an adapter, please note that we still currently offer a PCI slot Timing board (the TPRO/TSAT-PCI timing boards). These “legacy” timing boards don’t have all the capabilities of t he newer design TSync board, but may be an alternative that you can use in place of a PCIe-based timing board. Our Sales team can provide you with additional info on this earlier series timing board, If you are interested.

National Instruments (NI) PXi-e chassis/ PXI Express

- Refer to Salesforce Case 237548 (June 2020)
- Refer to sites such as: https://www.ni.com/en-us/shop/pxi.html
- **Issue**: TSync series bus connectors aren’t compatible with PXI/PXIe (PXI Express) connectors on the chassis

- Refer to: https://www.pickeringtest.com/en-us/kb/hardware-topics/switching-platform-selection/comparing-pxi-and-pxi-express

What Is PXI?

In a PXI system, a chassis provides power, cooling, and a communication bus for modular instruments or I/O modules. You can control these modules from either an embedded controller or an external PC, using one or several of NI's specialized engineering software tools to customize your system.

PCI/PCIe versus PXI/PXI-e (PXI Express)

Per: https://forums.ni.com/t5/Multifunction-DAQ/PCI-vs-PCI-Express-Vs-PXIe/td-p/3938455?profile.language=en

Older desktop PC's tended to have several PCI slots. PCI is an older technology with more limited bandwidth than the newer PCIe. Increasingly, one can't assume a new PC will have *any* PCI slots. They aren't extinct, but if you need to support a PCI board, you need to be careful what desktop you choose.

PXIe is more of an "**industrial**" bus rather than a "**consumer PC**" bus. PXIe boards are not physically compatible with normal desktop PC's. You would need a special PXI chassis for them. One would also often buy a special PXI controller, basically a special form factor motherboard that runs a PXI system.

It has pros and cons, one of the cons being a pretty drastic jump in cost.

Email from Keith to Trevor Dougherty (25 June 2020) FYI- I just spoke to Dave Sohn about this (we have a once-weekly meeting with apps to discuss Salesforce Cases).

He said the software/OS side would be typical for the TSync boards. But, he said the issue is the PXI-e chassis doesn't have any of the bus connectors (PCIe, VPX, PMC etc) that the TSync series boards support. So unfortunately, they don't have a way to connect the board into the chassis.

He said its highly unlikely we would be willing to spin a TSync board that has a compatible bus interface for this type chassis.

> **Email Keith sent to customer**; I checked with our Senior Applications Engineer about using the TSync series boards in a PXI-e chassis. He confirmed my suspicions...
>
> From the software-side (syncing Linux/NTP with a TSync board) this would be standard/normal operation for the TSync board. However, he said the problem is that the PXI-e chassis doesn't have available any of the bus-connector types (such as PCIe, PMC, cPCI, etc) that the TSync series timing boards support. This prevents the ability for a TSync series timing board from being able to physically connect to/interface with the chassis.

Host systems that have difficulty with TSync-PCIe board install

- TSync Firmware update version 2.2.3 (not applicable for Tsync-PTP boards) should resolve all of these issues.

HP computers
(as of at least 3/30/09) Apparently, there is an issue with HP being able to support PCIe cards in HP computers. They are currently investigating:

I installed the card into an HP Proliant DL360G4 server using a PCI-E riser card. When I powered on the server, it did not clear the POST. It just sat there with a black screen. I then removed the card and installed it into an HP Proliant DL380G5 server, which has native PCI-E slot on the main board and I got the same result. I made sure I give it plenty of time, but after 30 minutes in each server there was no activity when the card is installed. I double-checked my servers' BIOS and they are at the latest available levels from HP. When the TSYNC-PCIE card is not in them both servers clear POST and boot up OK.

****HP DL380G8 (Generation 8)**

- Refer to Morgan Stanley Japan (related Salesforce Cases 10117,11048,11396,11807)
- 5 of their batch of 16 TSync-PCIe boards not being detected by the BIOS (one of the 5 works fine in an earlier generation of an HP DL380).
- Tim Tetreault has been able to duplicate here with some boards. Problem clears if he moves the jumper to leave the TSync-PCIe board in bootloader mode. He suspects changes to our software may be causing this to happen.

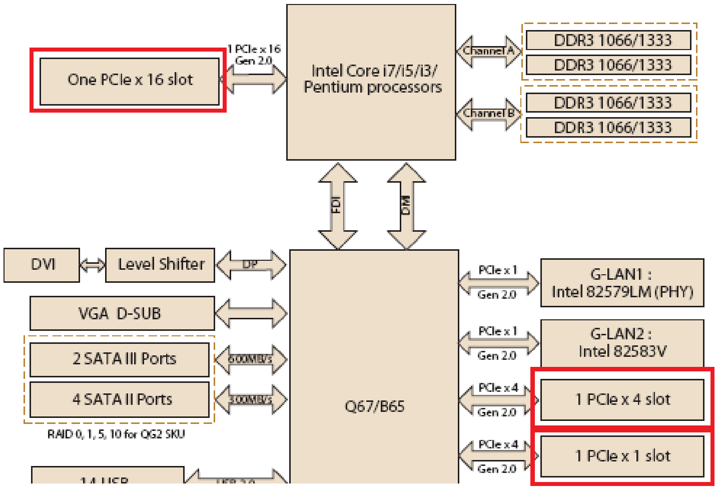
****HP DL380G7 (Generation 7)**

- Refer to “ATOS WorldGrid” (related Salesforce Cases 8297 and 11399)

Intel Core processor/Sandy bridge CPU’s PCI roor port hub

- Refer to Gilbert Young in Salesforce.

After several tests, I found out the detection issue was probably not related to ASPM. I tested the TSync-PCI PCIe card in the motherboard shown below. The MB has three PCIe slots: PCIe X16, PCIe x4 and PCIe x1. The PCIe x 16 slot is from the CPU. The PCIe x 4 and x1 are from the PCH. The TSync-PCI card got detected OK in either x4 or x1 slots. However, TSync-PCI card was not detected in the x16 slot. Even after I disabled ASPM in the BIOS, the card was still not detected in the PCIe x16 slot. I feel like there is a compatibility issue between the TSync-PCI and the CPU’s PCIe signals. The CPU that I used was i3-2120. There is a PCI express root port in the CPU. Have you heard of such compatibility issue between TSync-PCI and Sandy bridge CPU’s PCIe root port hub?



HARDWARE (Drawings, Specs and FAQs about the board)

EEPROM

- **U060R-2564-000K** (in Area at: https://app.bom.com/items/detail-notifications?item_id=1202847691&version_id=11010015318)
- Firmware update version 3.4.7 to version 3.4.9 included necessary changes to accommodate a change to the EEPROM (due to end-of-life of the earlier rev of this component)
  - In Arena at: https://app.bom.com/items/detail-spec?item_id=1260760865&version_id=11322181888
  - Refer to "***Hardware changes to the TSync PCB boards***" a little further below for more infoon EEPROM change

TSync-PCIe and Acutime Antennas for Invensys/Schneider Electric

- Refer to ECO-1789 (in Arena), Aug 2018

Associated Model 1169 GPS Fiber Optic Isolator for Accutime antennas

- Refer to "**Model 1169 GPS Fiber Optic Isolator (TX and RX) for GPS**" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- Optic Isolator kits helps protect the Timing board from being damaged by lightning strike to the Acutime antenna.

Acutime 360 GNSS Antennas (pre-configured at out factory to output NMEA- instead of the TSIP)

- Refer also to:..\CustomerServiceAssistance.pdf
- **Our P/N for Acutime 360 antenna configured to output NMEA: E025R-9000-0001 (in Arena at)** https://app.bom.com/items/detail-spec?item_id=1255453450&version_id=11133851398&orb_msg_single_search_p=1

  being released with just the Acutime 360 on it. Final BOM with have PD and software to program for NMEA output. This will be added prior to customer shipment.

Support for Acutime 360 antenna

- Support to use Acutime 360 antenna was added in TSync firmware update version 3.4.9 (~7 Dec 2018)
- Refer to **ECO-FAI-1886** in Arena: https://app.bom.com/changes/detail-summary?change_id=2395198855&orb_msg_single_search_p=1
- Refer also to: ..\..\PSB, PSP software updates\TSYNC boards\TSync firmware updates (PCIe, cPCI, PCI104)\TSync-PCIe\TSync-PCIe firmware updates

TSync-PCIe-002 boards capable of "NMEA input" from Acutime Antennas

- **Our P/N: 1191-9000-0600** (in Arena at): https://app.bom.com/items/detail-spec?item_id=1255453528&version_id=11176660778&orb_msg_single_search_p=1

Note: TSync-PCIe-002 (TSyncE-PCIe) boards were last time / discontinued ~Jan 2021.

Various packages associated with TSync-PCIe boards (such as for Invensys)

A) K0204AZ PACKAGE (TSync-PCIe-002, NMEA capable, and Anc kit)

- ➢ Refer to (in Arena): https://app.bom.com/items/detail-bom?item_id=1255453518&version_id=11275979668

Includes the following:
One 1191-9000-0600 (TSync-PCIe-002, NMEA capable)

One 1191-0000-0703 (anc kit), consisting of:
1191-1000-0703 (Bracket)
1191-1006-0803 (Label)
CA05R-MD15-0001 (1 ft extension cable)

B) K0204BB PACKAGE (“Spare/Repair PCI Interface Card”)

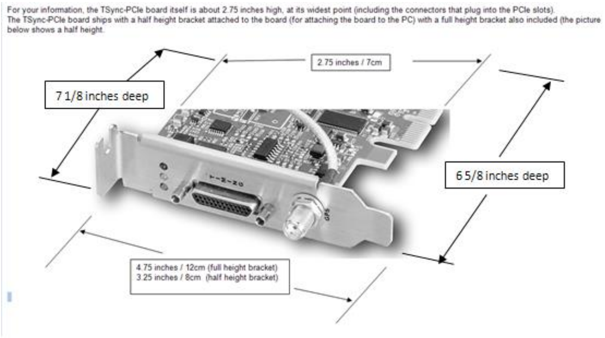
- ➢ Refer to ECO-2150 (in Arena): https://app.bom.com/changes/detail-summary?change_id=2396784379&

Includes the following:

*(1) **E025R-9000-0001**: Acutime 360 antenna (configured as NMEA Output)

*(1) **CAO5-1512-0100**: 100 ft Antenna Cable for the Trimble Acutime 360 antenna

Size of the TSync-PCIe board



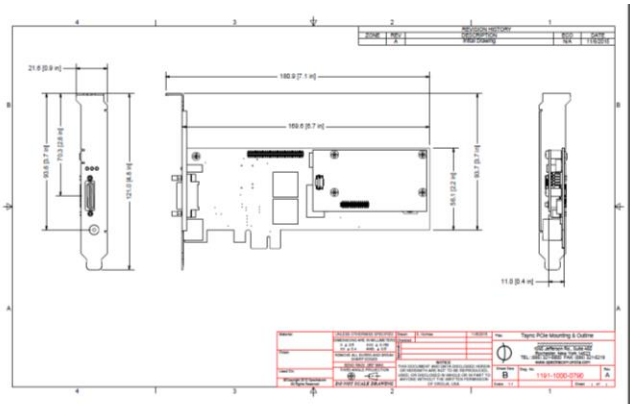
For your information, the TSync-PCIe board itself is about 2.75 inches high, at its widest point (including the connectors that plug into the PCIe slots).
The TSync-PCIe board ships with a half height bracket attached to the board (for attaching the board to the PC) with a full height bracket also included (the picture below shows a half height.

2.75 inches / 7cm
7 1/8 inches deep
6 5/8 inches deep
4.75 inches / 12cm (full height bracket)
3.25 inches / 8cm (half height bracket)
GPS

Note: Half-height bracket not available with TSync-PCIe-PTP boards due to need for Ethernet jack.

Dimensional drawing of the TSync-PCIe board

- ➢ Refer to (1191-1000-0790):..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\TSYNC-PCIe\Drawings



Weight of the Tsyncl-PCIe

The weight of the TSyncI-PCIe (with the GPS receiver attached to the board) is less than one pound. Per the scale, it looks like it's about 0.3 pounds.

PCIe board generation (Gen 1, Gen 2, Gen 3 , x1, x2, x16, etc) and Lanes

Q. we are hearing that many motherboards are not including a PCIe x1 slot. Is PCIe x1 falling out of favor? Is there some new interface that you are moving to? I have heard that I might be able to plug the PCIe x1 into a PCIe x4 or x16 slot?

(Reply from Keith after talking to Denis Reilly 15 Apr 2014) FYI- we intentionally chose this form factor while designing this timing board, because we wanted to be compatible with as many PCIe slots as possible. The definite advantage of being an x1 product is the PCIe specs favor downward compatibility with PCIe slots.

Higher lane PCIe slots are downward compatible, so an x1 board, such as Tsync-PCIe, can be plugged into an x2, x4, x16, slot, But if we were an x16 PCIe board, this would significantly limit the number of PCIe slots we would be compatible with. Because of downward compatibility of PCIe slots, there is no need for us to change the form factor of the Tsync-PCIe boards, even if x1 slots become non-existent ☺!!

Q, What PCIe versions does this card support?
Gen 1
Gen 2
Gen 3

Reply from Denis Reilly (7 Jun 13) We are a Gen 1 PCI Express board, and putting that board in a Gen 2 or Gen 3 slot is supposed to work. Furthermore, we are only a 1-lane (x1) card, and x1 cards are supposed to work in slots with more lanes.

I asked because there are some fiddly electrical revisions in Gen 1. I believe we support all of them, but if his question was more like "Do we support PCIe 1.1 or 1.0b", I would have had to look around a bit more.

TSync-PCIe Specs:

"Short term tracking": Is the oscillator stability when the board is synchronized to GPS. The values relate to our stability while disciplining the oscillator to GPS.

"Loss of satellites" is the oscillator stability when the board is in holdover. So for TCXO our oscillator stability is 1E-6 (1 ppm), and for OCXO our oscillator stability is 5E-8 (50 ppb).

"200 MHz Disciplined On-Board Clock": This is the speed of the KTS timing system. This is the speed that all of the functions (such as disciplining) operate at. This is how we get the 5ns accuracy resolution in the specs.

FAQs about the TSync specs

- The user manual says that the GPS output will be within +/- 50 ns of the Internal 1 PPS source. If it's true that the 1 PPS output will also be +/- 50 ns within of the Internal 1 PPS source, then that would mean that the 1 PPS output will be within +/- 100 ns of the GP output. Are the variations of GP output and 1 PPS output with respect to the Internal 1 IPP source constant and do they track with each other? If so, would it be possible to use the 5 ns offset capability on each of the outputs to reduce the time difference between them to something less than +/- 100 ns?

- The spec Rich is referring to is +/-50ns for the 1PPS and GPO to the reference source (GPS, IRIG, etc.), not the internal 1PPS of the TSync. The GPO and 1PPS should be within +/- 10ns of each other.

TEMPERATURE SPEC:

Q. (from Google) "For the PCI Express Time Code Processor card, what is the max temperatures for the processors/components on the board? I see in your spec you have a 75C max operating temperature but I believe this is an ambient spec. Can you clarify the actual chip max temp and where it is located?"

B. The OCXO oscillator is the component on the TSync-PCIe board with the lowest recommended temperature spec. The maximum case temperature spec for this component is +85C. The timing board is therefore derated to a spec of +75C to allow for local temperature rise on the board to remain below this specified threshold.

Power dissipation rating (Watts):

TSyncl-PCIe-OCXO

- 3.3V:0.8A, ~2.7W
- 12V:0.25A,~3W
- Total: ~6W out of 10W allowed (per the PCIe spec).

To begin, the 3.3v and 12v for the TSync-PCIe board are provided by the host computer through the PCIe connector. As for the power consumption of the TSync-PCI board:

The TSync-PCIe 3.3V power is mainly consumed by the FPGA and microprocessor, 1.2V DC-DC converter, and OCXO. The 12V power is consumed mainly by the IRIG input, output circuitry and the +12V to -12V DC-DC converter. The 5V for the GPS antenna power is also consumed from the +12V by a 5V DC-DC converter.

All the parts dump their heat into the PCB and ambient air, but mainly the PCB which causes about a 10C rise of the PCB/parts surface temperature. That is why the board is rated to 75C. The IC's and oscillators are all rated to 85C, so with a 10C rise on the board, they will be at 85C when the ambient is 75C.

GPS Receiver standoffs:
Note: GPS receiver stand-offs are installed on ALL boards at the board-house. If a customer doesn't want them to be installed, they can be removed by drilling them out.

Q. We're interested in possibly using the holes for the standoffs to add additional constraint to the card since we're putting this card in a mobile application. Can you give me more information on what the standoffs are and how they attach to the board? If it's easier to talk to an engineer, let me know and we can arrange a call.

A. The GPS receiver standoffs are ¼ inch round, 5/16 inch tall, 440 threaded and are swaged onto the PC board. For your reference, the standoffs we use are from RAF electrical with their P/N of 3048-B-440-AL-0. (Refer to http://www.alliedelec.com/Search/SearchResults.aspx?N=0&Ntk=Secondary&Ntt=2191608 for more information on this component).

Hardware changes to the TSync PCB boards

A) ECO-ITEM-000901 (TSync-PCIe PCB Update, June 2018)

- Refer to (in Arena) https://app.bom.com/changes/detail-summary?change_id=2386442842&

This ECO will release an updated 1191 PCB that fixes many outstanding issues and adds new features. (See Implementation notes for more details):

1) Create a new F000 assembly, 1191-0000-F002 for OCXO option configurations. (ECR-000142)

2) Remove the bracket, 1191-1000-0700, from the F000 assemblies.

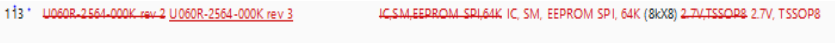
3) Updated assembly drawings to address DEV-000069.

PVP-000030 (Test procedure) was used to verify both 1191-0000-F001 & 1191-0000-F002 assemblies. PVR results are attached to ECO-001588 (in Arena at: https://app.bom.com/changes/detail-summary?change_id=2392856811)

The new software release V3.4.7 has the following updates/changes:

1. Added support for new FLASH
2. U-blox M8T upgrade support
3. Oscillator disciplining improvements
4. GPS/GNSS receivers can now save their default mode independently in the EEPROM.
    a. Support for up to 3 GNSS receivers

5. New EEPROMs will need to default the default mode to the desired value.
    - **U060R-2564-000K** (in Area at: https://app.bom.com/items/detail-notifications?item_id=1202847691&version_id=11010015318
    - **Changed from Rev 2 to Rev 3**



1↑3 * ~~U060R-2564-000K rev 2~~ U060R-2564-000K rev 3 ~~IC,SM,EEPROM SPI,64K~~ IC, SM, EEPROM SPI, 64K (8kX8) ~~2.7V,TSSOP8~~ 2.7V, TSSOP8

6. Reference Monitoring is added and can be accessed via the TSync Driver APIs
7. GPS receiver will now restart survey on power-up or board reset.
8. GPS/GNSS setting now defaults to GPS/GAL not GPS/GLO For u-blox M8T receivers only.

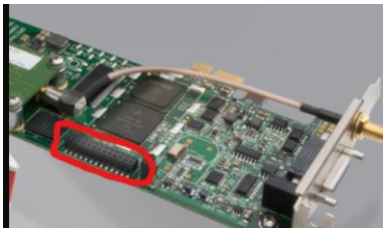
firmware change to support new Flash IC's (version 3.4.6-> 3.4.7)

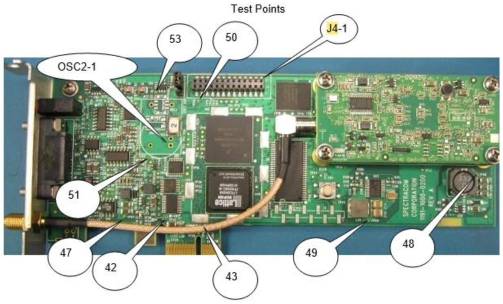
- Initially incorporated in version 3.4.6 (ECO01565 in Arena at https://app.bom.com/changes/detail-summary?change_id=2392755994&orb_msg_single_search_p=1)

  "This ECO will release the software that will support the new Flash IC.
  It will also support the new Temperature sensor that is being added to the update TSync-PCIe boards."

- V3.4.6 wasn't cut into production, so changes were actually incorporated in 3.4.7??

J4: Wiggler/JTAG (USB Coldfire) Flash programming connector

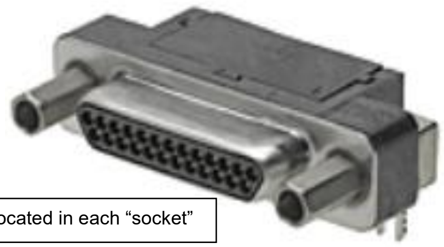




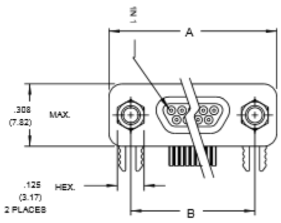
HP_20140108_001.jpg
Type : Fichier JPG
Taille : 2,57 Mo
Dimension : 3552 x 2000 pixels

Timing Connector for all TSync variants

- Micro-D Sub 25 (D25) connector on the TSync board.
- The **Micro-D Sub connector (J8)** is a Molex connector (Molex P/N is 83614-9012).

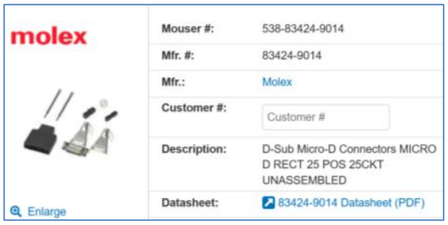


Small pin located in each “socket”



- **The mating connector** from Molex is their P/N 83424-9014 (https://www.mouser.com/ProductDetail/Molex/83424-9014?qs=x6EjVpvqMVMz3h01sH%2FmhQ%3D%3D&gclid=EAIaIQobChMI_K-_hcXL7AIVg4bACh3pPArYEAAYASAAEgIpBPD_BwE)

Commented [KW1]:



Mouser #:	538-83424-9014
Mfr. #:	83424-9014
Mfr.:	Molex
Customer #:	Customer #
Description:	D-Sub Micro-D Connectors MICRO D RECT 25 POS 25CKT UNASSEMBLED
Datasheet:	83424-9014 Datasheet (PDF)

Q. The gender of the connector on the model is male, but on your website the photos of the card show a female connector. Which gender is correct?

A. The Timing connector configuration can be easily deceived from photos as the opposite sex of what they really are. The connector on the edge of the board is actually a Male connector as there are recessed metal pins inside each hole. Its mating connector (on either the standard or the available premium break-out cable assemblies) is actually a Female connector with metal sockets for the pins on the board to insert into. The connector is displayed on page 3-1 of the attached user manual (the “1191-5000-0050 Manual Rev C” document). However, the photograph in the manual and in the data sheets can be deceiving because you can’t see the pins or sockets inside of the connector.

The Micro D-Sub connector (J8) is a Molex connector (their P/N is 83614-9012).
The mating connector from Molex is their P/N 83424-9014.
(http://www.molex.com/molex/products/datasheet.jsp?part=active/0834249014_IO_CONNECTORS.xml)

3.1.1 Timing Connector Pinout

Table 3.1—Pinout

Pin	Signal	Pin	Signal
1	GPIO Output 2	14	GPIO Output 3
2	Ground	15	Ground
3	GPIO Output 0	16	GPIO Output 1
4	GPIO Input 2	17	GPIO Input 3
5	Ground	18	Ground
6	GPIO Input 0	19	GPIO Input 1
7	External 1PPS Input	20	1PPS Output
8	Ground	21	Ground
9	IRIG AM Output	22	10MHz Output
10	IRIG AM Input +	23	Ground
11	IRIG AM Input -	24	IRIG DCLS Input -
12	IRIG DCLS Output -	25	IRIG DCLS Input +
13	IRIG DCLS Output +		

Breakout cables/Interface cable for the “Timing” connector

- **Link to Breakout cable data sheet:** I:\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards
  - refer to the second page of the “**TSync _Config and ordering**” document for a table).

Info related to both the Standard and Premium cables

- Refer to the online TSync user guide: http://manuals.spectracom.com/TS/Content/TS/Topics/OptionsAccies.htm#top

BNC connectors on the ends of the breakout cable dongles.

- Is there any way you would be able to provide me with the manufactures part number for the BNC connector to ensure that our custom cable will mate well with it?

Keith’s response: All of the BNC connectors on the breakout cable are Tyco Electronics P/N 5413779-3 (BNC JACK,COMMERCIAL, HEX CRIMP, RG174,500v). Refer to sites such as: http://www.te.com/catalog/pn/en/5413779-3.

Temperature specs for breakout cables

Q, Do you also have the operational temperature of the premium breakout cable?

A. **Keith’s response:** The breakout cables are rated for at least the same specs as the TSync-PCIe board itself (we haven’t tested the cables beyond the specs of the Tsync-PCIe board). Per page 2 of the attached TSync-PCIe datasheet, the operating temp specs for the TSync-PCIe board is -40° C to 80° C (-40°F to +176°F).

CA08R-DMD6-0001: Timing Interface Adapter cable (for the newer DB25 connector on standard and premium breakout cables)

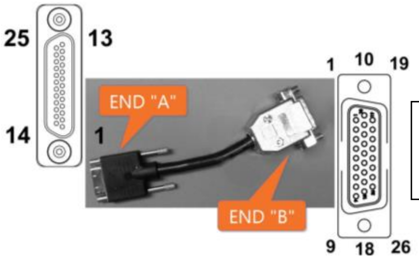
- Refer to the online TSync user guide: http://manuals.spectracom.com/TS/Content/TS/Topics/OptionsAccies.htm#top

PCIe and PMC cards are shipped with a 15 cm (6") adapter cable that is used to connect the micro 25-pin timing interface connector on the card to the breakout cable:

CA08R-DMD6-0001 (Micro-D25 to HD26 adapter cable)

Micro D 25-pin connector

To “**Timing**” connector

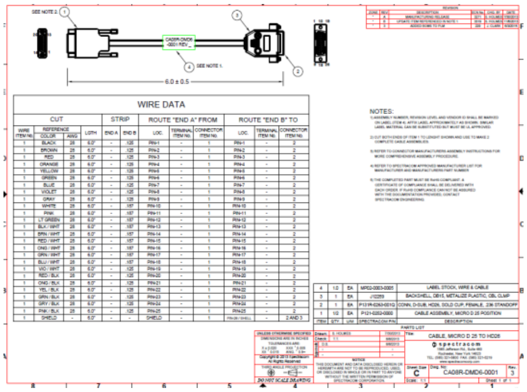


26-pin HD connector

To Breakout cable

Note: This adapter cable ships standard with both Standard and Premium breakout cables (not included in the anc kit- its shipped separately)

- In Arena at: https://app.bom.com/items/detail-spec?item_id=1202841931&version_id=10266255388&orb_msg_single_search_p=1
- **Link to .pdf cable drawing (CA08R-DMD6-0001**) in Arena at: https://files.bom.com/download/qSI6aGhZ3t2pJAX8yV2CegqFE9XwgCkM/uhtbojhevduouwmvipeyveggrsoonnwm/CA08R-DMD6-0001r3.pdf



REVISION					
ZONE	REV	DESCRIPTION	ECN No.	CHG. BY	DATE
*	A	MANUFACTURING RELEASE	3271	S. HOLMES	[illegible]
*	B	UPDATE ITEM REFERENCED IN NOTE 1	[illegible]	S. HOLMES	[illegible]
*	3	ADDED BOMS TO PLM	[illegible]	J. CLARK	[illegible]

WIRE DATA

CUT				STRIP		ROUTE "END A" FROM			ROUTE "END B" TO		
WIRE ITEM No.	REFERENCE COLOR	REFERENCE AWG	LGTH	END A	END B	LOC.	TERMINAL ITEM No.	CONNECTOR ITEM No.	LOC.	TERMINAL ITEM No.	CONNECTOR ITEM No.
1	BLACK	28	6.0"	-	.125	PIN-1	-	1	PIN-1	-	2
1	BROWN	28	6.0"	-	.125	PIN-2	-	1	PIN-2	-	2
1	RED	28	6.0"	-	.125	PIN-3	-	1	PIN-3	-	2
1	ORANGE	28	6.0"	-	.125	PIN-4	-	1	PIN-4	-	2
1	YELLOW	28	6.0"	-	.125	PIN-5	-	1	PIN-5	-	2
1	GREEN	28	6.0"	-	.125	PIN-6	-	1	PIN-6	-	2
1	BLUE	28	6.0"	-	.125	PIN-7	-	1	PIN-7	-	2
1	VIOLET	28	6.0"	-	.125	PIN-8	-	1	PIN-8	-	2
1	GRAY	28	6.0"	-	.125	PIN-9	-	1	PIN-9	-	2
1	WHITE	28	6.0"	-	.157	PIN-10	-	1	PIN-10	-	2
1	PINK	28	6.0"	-	.157	PIN-11	-	1	PIN-11	-	2
1	LT GREEN	28	6.0"	-	.157	PIN-12	-	1	PIN-12	-	2
1	BLK / WHT	28	6.0"	-	.157	PIN-13	-	1	PIN-13	-	2
1	BRN / WHT	28	6.0"	-	.157	PIN-14	-	1	PIN-14	-	2
1	RED / WHT	28	6.0"	-	.157	PIN-15	-	1	PIN-15	-	2
1	ONG / WHT	28	6.0"	-	.157	PIN-16	-	1	PIN-16	-	2
1	GRN / WHT	28	6.0"	-	.157	PIN-17	-	1	PIN-17	-	2
1	BLU / WHT	28	6.0"	-	.157	PIN-18	-	1	PIN-18	-	2
1	VIO / WHT	28	6.0"	-	.125	PIN-19	-	1	PIN-19	-	2
1	RED / BLK	28	6.0"	-	.125	PIN-20	-	1	PIN-20	-	2
1	ONG / BLK	28	6.0"	-	.125	PIN-21	-	1	PIN-21	-	2
1	YEL / BLK	28	6.0"	-	.125	PIN-22	-	1	PIN-22	-	2
1	GRN / BLK	28	6.0"	-	.125	PIN-23	-	1	PIN-23	-	2
1	GRY / BLK	28	6.0"	-	.125	PIN-24	-	1	PIN-24	-	2
1	PNK / BLK	28	6.0"	-	.125	PIN-25	-	1	PIN-25	-	2
1	SHIELD	-	6.0"	-	-	SHIELD	-	1	PIN-26 / SHELL	-	2 AND 3

NOTES:

1) ASSEMBLY NUMBER, REVISION LEVEL AND VENDOR ID SHALL BE MARKED ON LABEL (ITEM 4). AFFIX LABEL APPROXIMATELY AS SHOWN. SIMILAR LABEL MATERIAL CAN BE SUBSTITUTED BUT MUST BE UL APPROVED.

2) CUT BOTH ENDS OF ITEM 1 TO LENGHT SHOWN AND USE TO MAKE 2 COMPLETE CABLE ASSEMBLIES.

3) REFER TO CONNECTOR MANUFACTURERS ASSEMBLY INSTRUCTIONS FOR MORE COMPREHENSIVE ASSEMBLY PROCEDURE.

4) REFER TO SPECTRACOM APPROVED MANUFACTURER LIST FOR MANUFACTURER AND MANUFACTURERS PART NUMBER.

5) THE COMPLETED PART MUST BE RoHS COMPLIANT. A CERTIFICATE OF COMPLIANCE SHALL BE DELIVERED WITH EACH ORDER. IF RoHS COMPLIANCE CAN NOT BE ASSURED WITH THE DOCUMENTATION PROVIDED, CONTACT SPECTRACOM ENGINEERING.

ITEM	QTY.	U/M	SPECTRACOM P/N	DESCRIPTION
4	1.0	EA	MP02-0003-0005	LABEL STOCK, WIRE & CABLE
3	1	EA	J12259	BACKSHELL, DB15, METALIZE PLASTIC, DBL CLMP
2	1	EA	P131R-0263-001G	CONN, D-SUB, HD26, SOLD CUP, FEMALE, .236 STANDOFF
1	1/2	EA	P121-0252-0000	CABLE ASSEMBLY, MICRO D 25 POSITION

PARTS LIST

UNLESS OTHERWISE SPECIFIED
DIMENSIONS ARE IN INCHES
TOLERANCES ARE:
X ± 0.020 XXX ± 0.005
XX ± 0.010 ANG. ± 0.5°

Copyright © 2013 Spectracom
All Rights Reserved

THIRD ANGLE PROJECTION

DO NOT SCALE DRAWING

Drawn:	S. HOLMES	[illegible]
Check:	[illegible]	[illegible]
Approvals	D.S.	[illegible]
	-	-
	-	-

NOTICE
THIS DOCUMENT AND DATA DISCLOSED HEREIN OR HEREWITH ARE NOT TO BE REPRODUCED, USED, OR DISCLOSED IN WHOLE OR IN PART TO ANYONE WITHOUT THE WRITTEN PERMISSION OF SPECTRACOM CORPORATION.

Title: CABLE, MICRO D 25 TO HD26

spectracom
1565 Jefferson Rd., Suite 460
Rochester, New York 14623
TEL: (585) 321-5800 FAX: (585) 321-5219
www.spectracomcorp.com

Sheet Size: C | Dwg. No: CA08R-DMD6-0001 | Rev: 3
Scale: 1:1 | Sheet 1 of 1

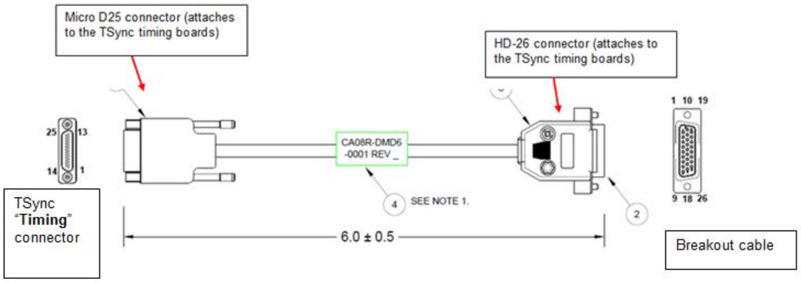
- Installed between TSync board and the newer design breakout cables (which started shipping around Nov 2013)
- Micro-D25 end of adapter attaches to TSync's Timing connector.
- HD (High Density) 26 end attaches to the new design breakout cable.
- Refer to "Connector Change" further below for the reason for this adapter cable being required when using newer breakout cabes. since about the end of 2013.

AWG of the internal wires inside the cable

- the cable conductors are 28 AWG

Q Regarding the cable, the customer needs the AWG of the cables (signal I/O) used. Sorry, I understood that he needs the section of the big black cable, **but no it is the AWG of cables inside it**.

A Reply from Dave Lorah (10 Apr 2019) The cable conductors are 28 AWG. Attached is a drawing for the cable assembly.


Micro D25 connector (attaches to the TSync timing boards)
HD-26 connector (attaches to the TSync timing boards)
1 10 19
25
13
14
1
CA08R-DMD6 -0001 REV _
4
SEE NOTE 1.
2
9 18 26
TSync "Timing" connector
6.0 ± 0.5
Breakout cable

Note: Pin 26 of HD-26 doesn't connect to the Micro D25 connector. It is connected to the shell of the HD-26 connector.

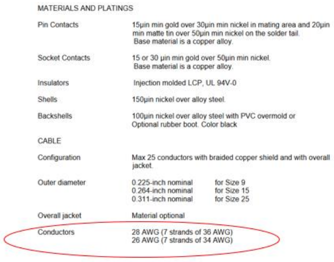
Voltage rating for the inter-connect cabling used between the two Molex connectors

- Cable spec is 30 Vrms

Q I am contacting you as we need to ship a product developed by your company with the following Part Number:
- **CA08R-DMD6-0001.** In order to fill the requirements, we need to obtain the voltage rating for this cable. Could you please let us know which would be that value?

- Refer to SF case 121624
- The Wiring harness is our P/N **P121-0252-0000** (MFG and Part Number: Molex P/Ns 083424905, 0834249058 or 0834249059

- Refer to http://www.molex.com/pdm_docs/ps/PS-83421-001-001.pdf for the Molex product specs (below)



MATERIALS AND PLATINGS		
Pin Contacts	15µin min gold over 30µin min nickel in mating area and 20µin min matte tin over 50µin min nickel on the solder tail. Base material is a copper alloy.	
Socket Contacts	15 or 30 µin min gold over 50µin min nickel. Base material is a copper alloy.	
Insulators	Injection molded LCP, UL 94V-0	
Shells	150µin nickel over alloy steel.	
Backshells	100µin nickel over alloy steel with PVC overmold or Optional rubber boot. Color black	
CABLE		
Configuration	Max 25 conductors with braided copper shield and with overall jacket.	

Outer diameter	0.225-inch nominal
	0.264-inch nominal
	0.311-inch nominal
		for Size 9
		for Size 15
		for Size 25

Overall jacket	Material optional	
Conductors	28 AWG (7 strands of 36 AWG) 26 AWG (7 strands of 34 AWG)	

- Refer to https://accuglassproducts.com/product.php?productid=16176&cat=0&page=1

For max currenthttps://www.powerstream.com/Wire_Size.htm (for AWG ratings)

voltage rating**.** Cable spec is 30 Vrms

AWG 36 = 0.035 amps

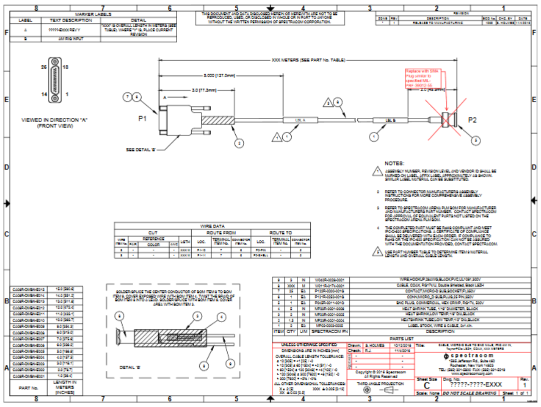
AWG 34 = 0.056 amps

A   As confirmed by Tom Richardardson (13 Nov 17) ***voltage rating*****:** Cable spec is **30 Vrms**

C) Special/Custom Breakout cables for Timing connector

TSync cables with SMA connector for Raytheon (1 to 8 meter lengths)

- ➢ Refer to ECO-1662 (in Arena): https://app.bom.com/changes/detail-ecrs?change_id=2393442862
- ➢ SMA connector must be MIL-PRF-39012-55 compliant
- ➢ Refer to cable spec drawing (in Arena): https://app.bom.com/ecrs/detail-attach?ecr_id=1450527169



Available Part Numbers

- CA08R-DMSA-E001 (1 meter cable)
- CA08R-DMSA-E002 (2 meter cable)
- CA08R-DMSA-E003 (3 meter cable)
- CA08R-DMSA-E004 (4 meter cable)
- CA08R-DMSA-E005 (5 meter cable)
- CA08R-DMSA-E006 (6 meter cable)
- CA08R-DMSA-E007 (7 meter cable)
- CA08R-DMSA-E008 (8 meter cable)

1. **CA08R-DMBN-0003**
   - ➢ CABLE, TSYNC, IRIG DCLS INPUT, 1 meter
   - ➢ Refer to ECO-ITEM-00089 (in Arena): https://app.bom.com/items/detail-spec?item_id=1251299771&version_id=11065917198

2. **CA08R-DMBN-Exxx (where xxx is length in meters)**
   - ➢ CABLE, MICRO-D SUB TO **BNC** MALE, IRIG AM IN, Tsync-PCIe LSZH, COAX (in meters)
   - ➢ **Examples**: CA08R-DMBN-E001, CA08R- DMBN-E003, etc

- Search Salesforce for “**CA08R-DMBN-**”

3. **CA08R-DMSA-Exxx (where xxx is length in meters)**
    - CABLE, MICRO-D SUB TO **SMA** MALE, IRIG AM IN, Tsync-PCIe LSZH, COAX, (METERS)
    - **Examples**: CA08R-DMSA-E001, CA08R-DMSA-E002, CA08R-DMSA-E003, etc
    - Search Arena for “**CA08R-DMSA-**”

D) Standard Breakout cable ("Partial" breakout Cable)

- provided with each TSync board (at no additional cost) unless Premium breakout cable is purchased.
- Refer to the online TSync user guide: http://manuals.spectracom.com/TS/Content/TS/Topics/OptionsAccies.htm#top

Note: As of March 2014, there are two versions of the Standard breakout cable. Number 1 below is the current version as of ~Apr 2014, while Number 2 was replaced by the first one below.

- P/N for the newer variant (after Mar 2014) of this Standard cable is: CA08R-0000-0006
- P/N for the earlier variant of this cable (before Mar 2014) was CA08R-0000-0003

DB25 Connector Change

- Due to the pins on the main connector of the breakout cable (which attaches to the TSync board's "**Timing**" connector) periodically getting pushed-in, this connector on the breakout cable was changed:

A) Newer connector: High Density HD26 connector

- **Our P/N** P131R-0263-011Q
- **MFG. P/N** 180-YYY-20YLYY1)
- **In Arena**: https://app.bom.com/items/detail-spec?item_id=1205867520&version_id=10266027468&orb_msg_Single_Search_p=1&redirect_Seqno=6149025570

Changed from the following previous connector

B) Original connector: Molex Micro-D 25-pin connector

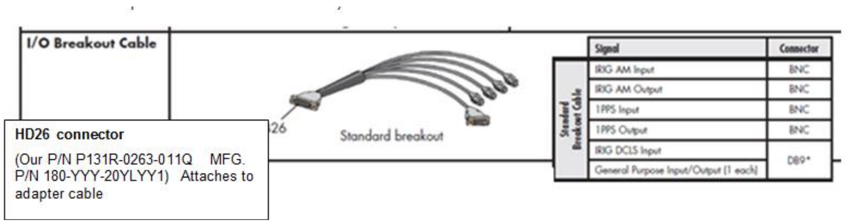
- (Our P/N P121R-0252-001G Molex P/N 83424-9014)

Breakout cable change due to connector change described above

1. **CA08R-0000-0006 (newer design Standard breakout cable, as of ~ Apr 2014)**
   - Shortcut to **cable drawing** (CA08R-0000-0006) in Arena: https://app.bom.com/items/detail-spec?item_id=1202841927&version_id=10221334728&orb_msg_Single_Search_p=1
   - Shortcut to **basic Breakout cable schematics/drawing** (CA08R-0000-0006) I:\Engineering\Archive\New Released\Cable Drawings

   Note: the pin-outs (and therefore the schematic) didn't change with the newer breakout cables. They are still the same since the product was first released.

   Note: To use these newer breakout cables with any TSync board variant, an adaptor cable will also be needed. (Our P/N for the adapter cable is CA08R-DMD6-0001)



HD26 connector

(Our P/N P131R-0263-011Q MFG. P/N 180-YYY-20YLYY1) Attaches to adapter cable

Signal	Connector
IRIG AM Input	BNC
IRIG AM Output	BNC
1PPS Input	BNC
1PPS Output	BNC
IRIG DCLS Input	DB9*
General Purpose Input/Output (1 each)	

Specific to IRIG DCLS output

Standard breakout cable

- IRIG DCLS output is not **available on Standard Breakout cable (it is available on the Premium Breakout cable)**
- **Note:** IRIG DCLS is on the "**Timing" connector pins 12 and 13** (pin 13 and ground for single-ended). Table for Timing connector is below

Timing connector pin-out

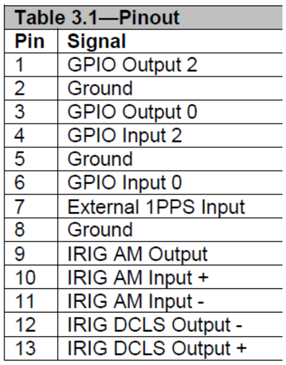
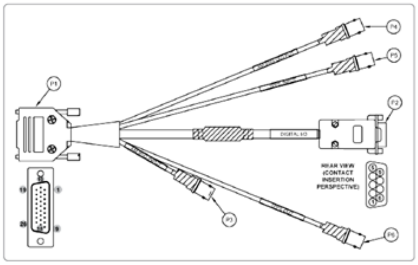


Table 3.1—Pinout

Pin	Signal
1	GPIO Output 2
2	Ground
3	GPIO Output 0
4	GPIO Input 2
5	Ground
6	GPIO Input 0
7	External 1PPS Input
8	Ground
9	IRIG AM Output
10	IRIG AM Input +
11	IRIG AM Input -
12	IRIG DCLS Output -
13	IRIG DCLS Output +



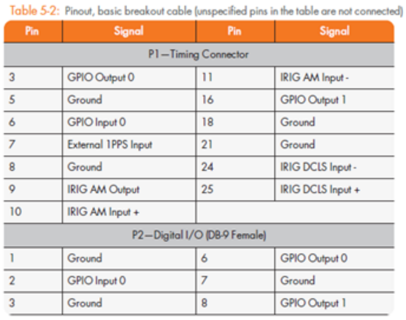
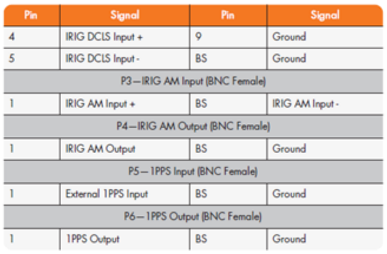


Table 5-2: Pinout, basic breakout cable (unspecified pins in the table are not connected)

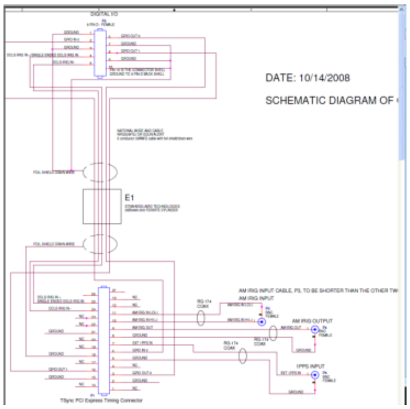


Pin	Signal	Pin	Signal
	P1—Timing Connector		
3	GPIO Output 0	11	IRIG AM Input -
5	Ground	16	GPIO Output 1
6	GPIO Input 0	18	Ground
7	External 1PPS Input	21	Ground
8	Ground	24	IRIG DCLS Input -
9	IRIG AM Output	25	IRIG DCLS Input +
10	IRIG AM Input +		
	P2—Digital I/O (DB-9 Female)		
1	Ground	6	GPIO Output 0
2	GPIO Input 0	7	Ground
3	Ground	8	GPIO Output 1
4	IRIG DCLS Input +	9	Ground
5	IRIG DCLS Input -	BS	Ground
	P3—IRIG AM Input (BNC Female)		
1	IRIG AM Input +	BS	IRIG AM Input -
	P4—IRIG AM Output (BNC Female)		
1	IRIG AM Output	BS	Ground
	P5—1PPS Input (BNC Female)		
1	External 1PPS Input	BS	Ground
	P6—1PPS Output (BNC Female)		
1	1PPS Output	BS	Ground

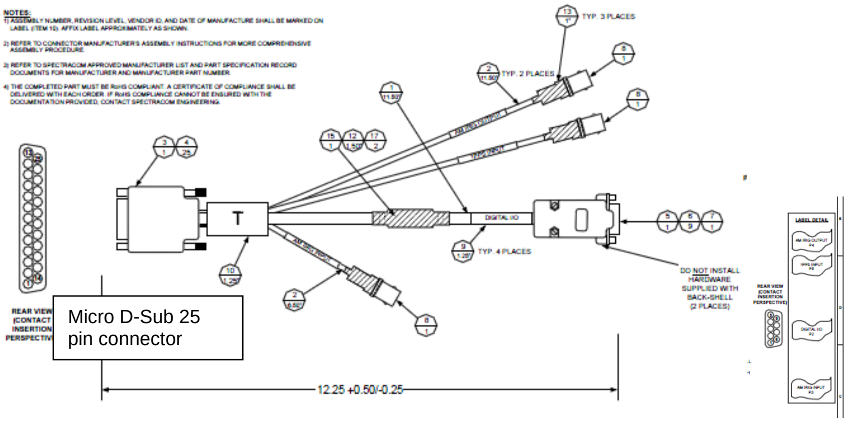
2. CA08R-0000-0003 earlier Standard Breakout cable (Discontinued Mar 2014)

- ~~CA08R-0000-0003~~ replaced by CA08R-0000-0006 on ECN 3433, Mar 2014

Shortcut to basic Breakout cable schematics (1191-1001-0401) I:\Engineering\Schematic\1191-1001-0401
Note: If one or more signals to or from the timing board don't appear to be present, verify the pins of the large 25 pin connector (which plugs into the Timing connector) doesn't have any pushed-in pins. Dave Sohn says he has seen the pins pushed (so they aren't making contact with the pins of the Timing connector) in a couple of instances.



Shortcut to picture of discontinued standard breakout cable (**CA08R-0000-0003**, as shown below): in Arena https://app.bom.com/items/detail-spec?item_id=1202841926&version_id=10294592608&orb_msg_Single_Search_p=1&redirect_Seqno=7068646412



NOTES:

1) ASSEMBLY NUMBER, REVISION LEVEL, VENDOR ID, AND DATE OF MANUFACTURE SHALL BE MARKED ON LABEL (ITEM 10). AFFIX LABEL APPROXIMATELY AS SHOWN.

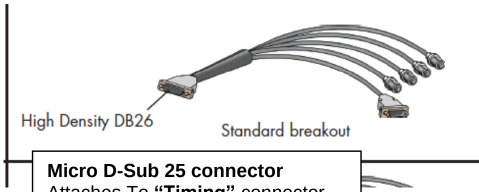
2) REFER TO CONNECTOR MANUFACTURER'S ASSEMBLY INSTRUCTIONS FOR MORE COMPREHENSIVE ASSEMBLY PROCEDURE.

3) REFER TO SPECTRACOM APPROVED MANUFACTURER LIST AND PART SPECIFICATION RECORD DOCUMENTS FOR MANUFACTURER AND MANUFACTURER PART NUMBER.

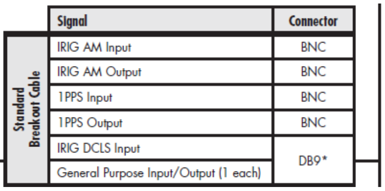
4) THE COMPLETED PART MUST BE RoHS COMPLIANT. A CERTIFICATE OF COMPLIANCE SHALL BE DELIVERED WITH EACH ORDER. IF RoHS COMPLIANCE CANNOT BE ENSURED WITH THE DOCUMENTATION PROVIDED, CONTACT SPECTRACOM ENGINEERING.

DO NOT INSTALL HARDWARE SUPPLIED WITH BACK-SHELL (2 PLACES)

Micro D-Sub 25 pin connector



Micro D-Sub 25 connector
Attaches To **“Timing”** connector on TSync board

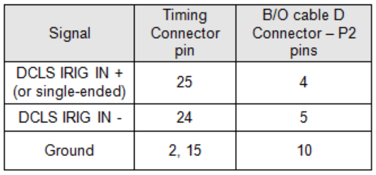


Standard Breakout Cable	Signal	Connector
	IRIG AM Input	BNC
	IRIG AM Output	BNC
	1PPS Input	BNC
	1PPS Output	BNC
	IRIG DCLS Input	DB9*
	General Purpose Input/Output (1 each)	

Input Signals:

1 ea. IRIG AM (BNC –P3)

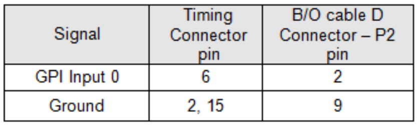
1 ea. IRIG DCLS (D Connector – P2)



Signal	Timing Connector pin	B/O cable D Connector – P2 pins
DCLS IRIG IN + (or single-ended)	25	4
DCLS IRIG IN -	24	5
Ground	2, 15	10

1 ea. 1PPS (BNC – P5)

1 ea. GPI (General Purpose Input) (D Connector – P2)

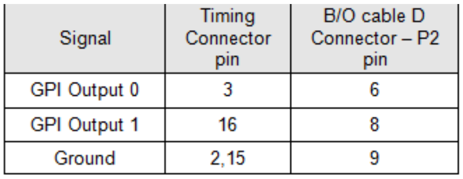


Signal	Timing Connector pin	B/O cable D Connector – P2 pin
GPI Input 0	6	2
Ground	2, 15	9

Output Signals:
1 ea 1PPS output (added ~Oct 2013)

1 ea. IRIG AM (BNC connector - P4)

- ea. GPO (General Purpose Output) (D Connector – P2)



Signal	Timing Connector pin	B/O cable D Connector – P2 pin
GPI Output 0	3	6
GPI Output 1	16	8
Ground	2,15	9

E) Premium Breakout cable

- ➢ Refer to the online TSync online user guide: http://manuals.spectracom.com/TS/Content/TS/Topics/OptionsAccies.htm#top
- ➢ P/N for the newer variant (after Mar 2014) of this Standard cable is: **CA08R-0000-0005**
- ➢ P//N for the earlier variant of this cable (before Mar 2014) was **CA08R-0000-0001**

Note: As of march 2014, there are two versions of the Premium breakout cable. Number 1 below is the current version as of ~Apr 2014, while Number 2 was replaced by the first one below.

Connector change

Due to the pins on the main connector of the breakout cable (which attaches to the TSync board's "**Timing**" connector) periodically getting pushed-in, this connector on the breakout cable was changed from:

- ➢ **Original connector: Molex Micro-D 25-pin connector** (Our P/N P121R-0252-001G Molex P/N 83424-9014)

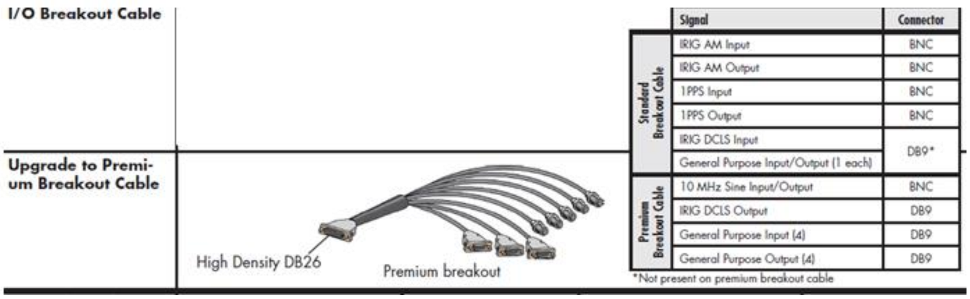
To

- ➢ **Newer: High Density HD26 connector** (Our P/N P131R-0263-011Q MFG. P/N 180-YYY-20YLYY1)

  In Arena: https://app.bom.com/items/detail-spec?item_id=1205867520&version_id=10266027468&orb_msg_Single_Search_p=1&redirect_Seqno=6149025570

Associated cable change because connector changed

1. **CA08R-0000-0005** (More recent Premium breakout cable as of ~Apr 2014)



	Signal	Connector
Standard Breakout Cable	IRIG AM Input	BNC
	IRIG AM Output	BNC
	1PPS Input	BNC
	1PPS Output	BNC
	IRIG DCLS Input	DB9*
	General Purpose Input/Output (1 each)	
Premium Breakout Cable	10 MHz Sine Input/Output	BNC
	IRIG DCLS Output	DB9
	General Purpose Input (4)	DB9
	General Purpose Output (4)	DB9

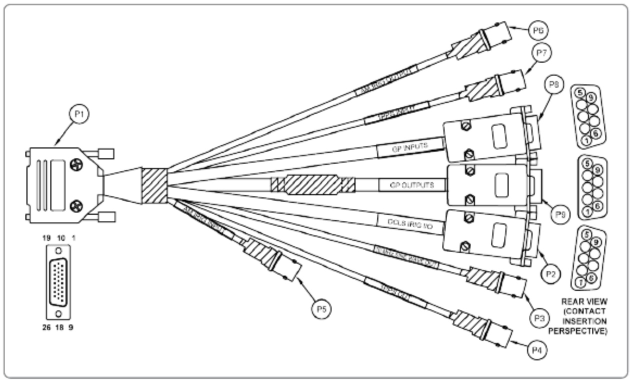
*Not present on premium breakout cable

- ➢ Replaces discontinued breakout cable CA08R-0000-0001 (ECN 3433, Mar 2014)
- ➢ Shortcut to premium Breakout cable schematics (1191-1001-0400) I:\Engineering\Schematic\1191-1001-0400
- ➢ Note: the pin-outs (and therefore the schematic) didn't change with the newer breakout cables. They are still the same since the product was first released.
- ➢ Shortcut to cable drawing (CA08R-0000-0005) in Arena: https://app.bom.com/items/detail-spec
- ➢ **Shortcut to Process Detail (CA08R-0000-0005) in Arena**: https://app.bom.com/items/detail-spec?item_id=1203165794&version_id=10257958418&orb_msg_Single_Search_p=1&redirect_Seqno=6150335046

Note: To use these newer breakout cables with any TSync board, an adaptor cable will also be needed. (CA08R-DMD6-0001)

Premium Breakout Cable

The premium breakout cable breaks out all features from the timing connector to separate BNC and DB-9 connectors for use. See table below for details.



IRIG DCLS output

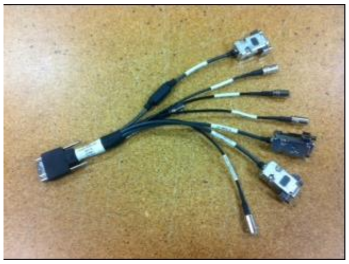
Premium breakout cable only

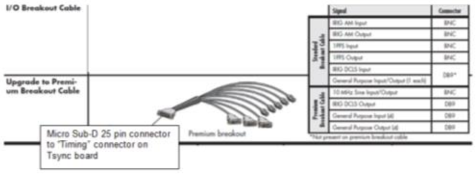
- IRIG DCLS output is not **available on Standard Breakout cable (it is available on the Premium Breakout cable)**

Table 5-3: Pinout, premium breakout cable (unspecified pins are not connected in the cable)

Pin	Signal	Pin	Signal
P1—Timing Connector			
1	GPIO Output 2	14	GPIO Output 3
2	Ground	15	Ground
3	GPIO Output 0	16	GPIO Output 1
4	GPIO Input 2	17	GPIO Input 3
5	Ground	18	Ground
6	GPIO Input 0	19	GPIO Input 1
7	External 1PPS Input	20	1PPS Output
8	Ground	21	Ground
9	IRIG AM Output	22	10MHz Output
10	IRIG AM Input +	23	Ground
11	IRIG AM Input -	24	IRIG DCLS Input -
12	IRIG DCLS Output -	25	IRIG DCLS Input +
13	IRIG DCLS Output +	26	Shield
IRIG DCLS I/O (DB-9 Female)			
2	Ground	6	IRIG DCLS Output +
3	Ground	7	IRIG DCLS Output -
4	IRIG DCLS Input +	BS	Ground
5	IRIG DCLS Input -		
P3— 10MHz Output (BNC Female)			
1	10MHz Output	BS	Ground
P4— 1PPS Output (BNC Female)			
1	1PPS Output	BS	Ground
P5—IRIG AM Input (BNC Female)			
1	IRIG AM Input +	BS	IRIG AM Input -
P6—IRIG AM Output (BNC Female)			
1	IRIG AM Output	BS	Ground
P7— 1PPS Input (BNC Female)			
1	External 1PPS Input	BS	Ground
P8—GP Input (DB-9 Female)			
1	GPIO Input 0	7	Ground
2	GPIO Input 1	8	Ground
3	GPIO Input 2	9	Ground
4	GPIO Input 3	BS	Ground
6	Ground		
P9—GP Output (DB-9 Female)			
1	GPIO Output 0	7	Ground
2	GPIO Output 1	8	Ground
3	GPIO Output 2	9	Ground
4	GPIO Output 3		
6	Ground	BS	Ground

2. **CA08R-0000-0001** (original breakout cable Discontinued, Mar 2014)





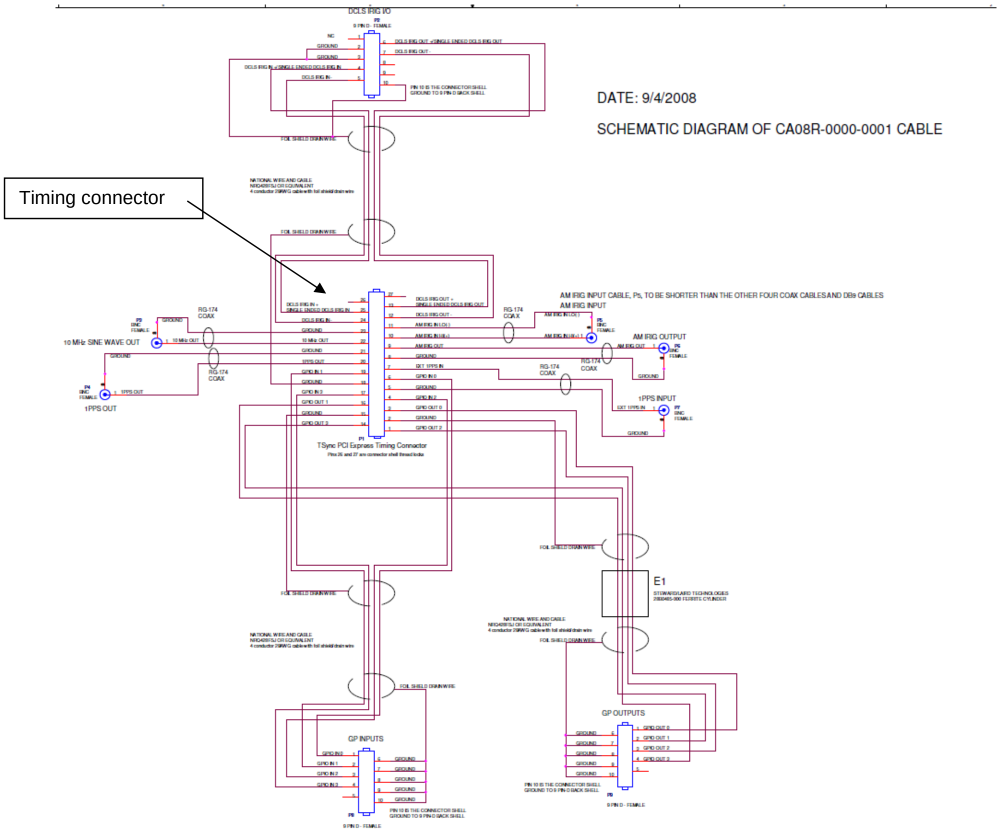
I/O Breakout Cable
Upgrade to Premium Breakout Cable
Micro Sub-D 25 pin connector to "Timing" connector on Tsync board
Premium breakout

	Signal	Connector
Standard Breakout Cable	IRIG AM Input	BNC
	IRIG AM Output	BNC
	1PPS Input	BNC
	1PPS Output	BNC
	IRIG DCLS Input	DB9*
	General Purpose Input/Output (1 each)	
Premium Breakout Cable	10 MHz Sine Input/Output	BNC
	IRIG DCLS Output	DB9
	General Purpose Input (4)	DB9
	General Purpose Output (4)	DB9

*Not present on premium breakout cable

- ➢ CA08R-0000-0001 was replaced by CA08R-0000-0005 on ECN 3433, Mar 2014
- ➢ **Link to process detail** in Arena: https://app.bom.com/items/detail-spec
- ➢ **Shortcut to premium Breakout cable schematics (1191-1001-0400)** I:\Engineering\Schematic\1191-1001-0400

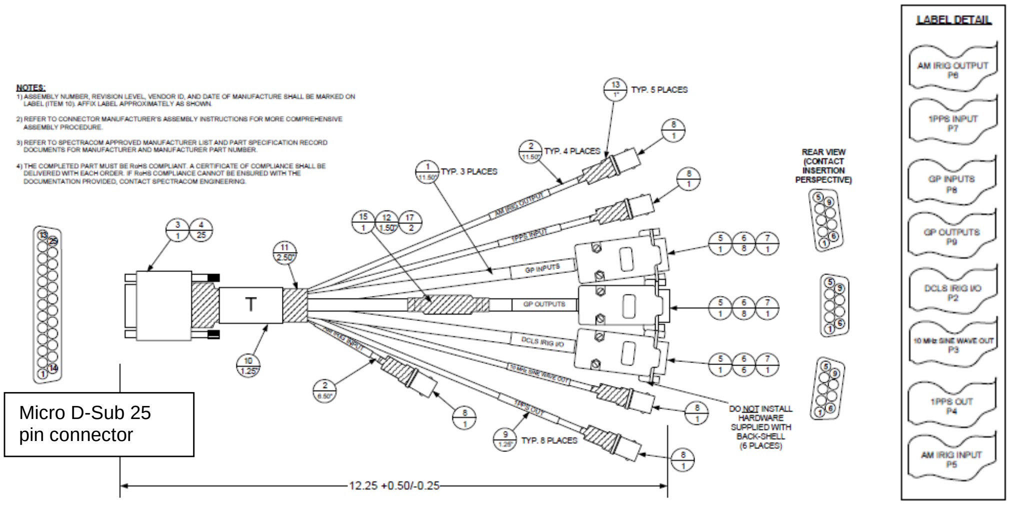
Note: If one or more signals to or from the timing board don't appear to be present, verify the pins of the large 25 pin connector (which plugs into the Timing connector) doesn't have any pushed-in pins. Dave Sohn says he has seen the pins pushed (so they aren't making contact with the pins of the Timing connector) in a couple of instances.



Timing connector

DATE: 9/4/2008

SCHEMATIC DIAGRAM OF CA08R-0000-0001 CABLE



NOTES:

1) ASSEMBLY NUMBER, REVISION LEVEL, VENDOR ID, AND DATE OF MANUFACTURE SHALL BE MARKED ON LABEL (ITEM 10). AFFIX LABEL APPROXIMATELY AS SHOWN.

2) REFER TO CONNECTOR MANUFACTURER'S ASSEMBLY INSTRUCTIONS FOR MORE COMPREHENSIVE ASSEMBLY PROCEDURE.

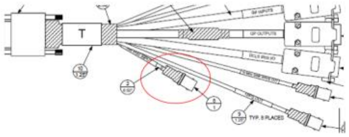
3) REFER TO SPECTRACOM APPROVED MANUFACTURER LIST AND PART SPECIFICATION RECORD DOCUMENTS FOR MANUFACTURER AND MANUFACTURER PART NUMBER.

4) THE COMPLETED PART MUST BE RoHS COMPLIANT. A CERTIFICATE OF COMPLIANCE SHALL BE DELIVERED WITH EACH ORDER. IF RoHS COMPLIANCE CANNOT BE ENSURED WITH THE DOCUMENTATION PROVIDED, CONTACT SPECTRACOM ENGINEERING.

Premium Breakout Cable supports the following:

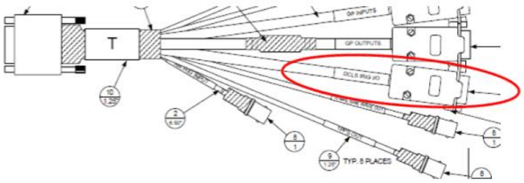
A) Input Signals:

1 ea. IRIG AM (BNC - P5) (shortest length of BNC cable)



1 ea. IRIG DCLS (D Connector on end of "DCLS IRIG I/O" cable – P2)

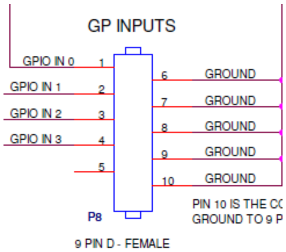
Signal	Timing Connector pin	B/O cable D Connector – P2 pin
DCLS IRIG In + (or single ended)	25	4
DCLS IRIG In -	24	5



`

1 ea. 1PPS input (BNC - P7)

4 ea. GPI (General Purpose Inputs) (DB9F Connector - P8)

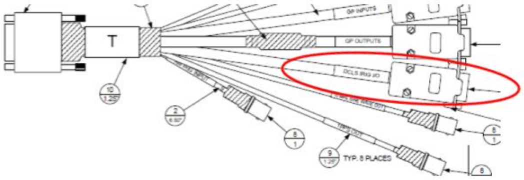


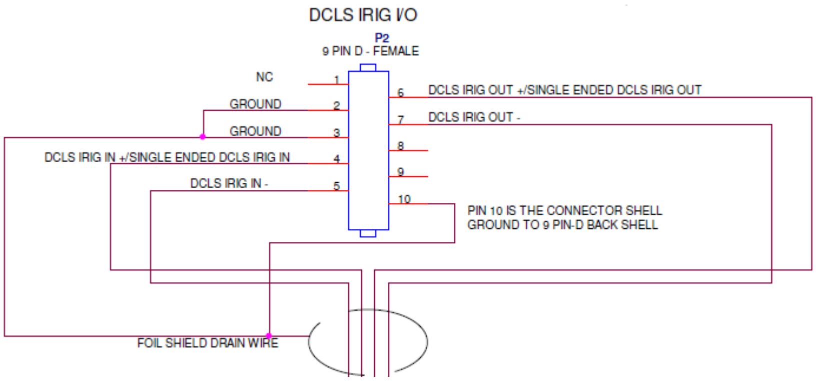
Signal	Timing Connector pin	B/O cable D Connector – P8 pin
GPI in 0	6	1
GPI in 1	19	2
GPI in 2	4	3
GPI in 3	17	4
Ground	2,15	6,7,8,9

B) Output Signals:

1 ea. IRIG AM (BNC – P6)

1 ea. IRIG DCLS (D Connector on end of “DCLS IRIG I/O” cable – P2)



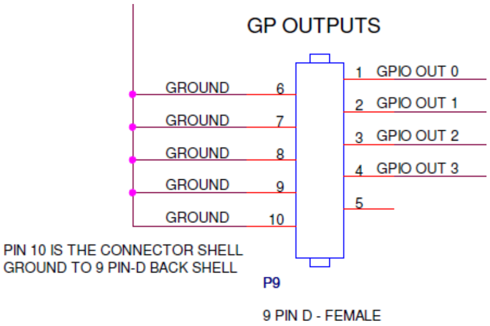

DCLS IRIG I/O
P2
9 PIN D - FEMALE
NC
1
GROUND
2
GROUND
3
DCLS IRIG IN +/SINGLE ENDED DCLS IRIG IN
4
DCLS IRIG IN -
5
6
DCLS IRIG OUT +/SINGLE ENDED DCLS IRIG OUT
7
DCLS IRIG OUT -
8
9
10
PIN 10 IS THE CONNECTOR SHELL
GROUND TO 9 PIN-D BACK SHELL
FOIL SHIELD DRAIN WIRE

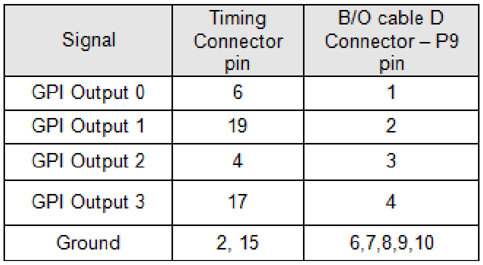
Signal	Timing Connector pin	B/O cable D Connector – P2 pin
DCLS IRIG Out + (or single ended)	13	6
DCLS IRIG Out -	12	7
Ground	2,15	10

1 each 1PPS Output (BNC – P4)

1 ea. 10 MHZ Output (BNC – P3)

4 ea. GPO (General Purpose Outputs) (DB9F Connector – P9)





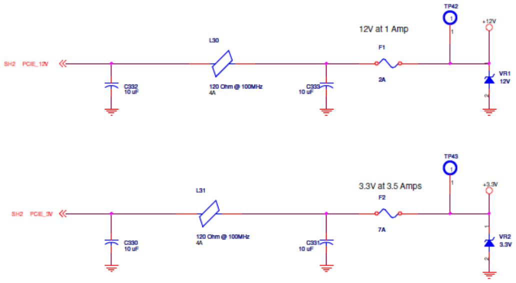
Signal	Timing Connector pin	B/O cable D Connector – P9 pin
GPI Output 0	6	1
GPI Output 1	19	2
GPI Output 2	4	3
GPI Output 3	17	4
Ground	2, 15	6,7,8,9,10

Power (+3.3vdc +12vdc from PCI bus) +5vdc and -12vdc generation

- **Link to board schematic (1191-1001-0200):** **(1191** -> **"1191-1000-0200 board files"** -> **"1191-1001-0200)**: [I:\New Released\PCB Documentation]
- **Shortcut to PCIe specs:** [I:\Engineering\Specs and Standards\PCISIG]

Fuses

- **Fuse F1**: for 12vdc input from PCIe bus slot
- **Fuse F2**: for 3.3vdc input from PCIe bus slot



Power consumption

Summary: TSync-PCIe boards are designed to and compliant with the PCI specifications. Refer to **Section 4.2: Power Consumption** (starting on page 35) of the "PCI_Express_CEM_1.1.pdf" document at [I:\Engineering\Specs and Standards\PCISIG]

- +12vdc and +3.3vdc are obtained from the PCIe bus
- -12vdc is generated from the +12vdc input.
- +5vdc is generated from +12vdc input.

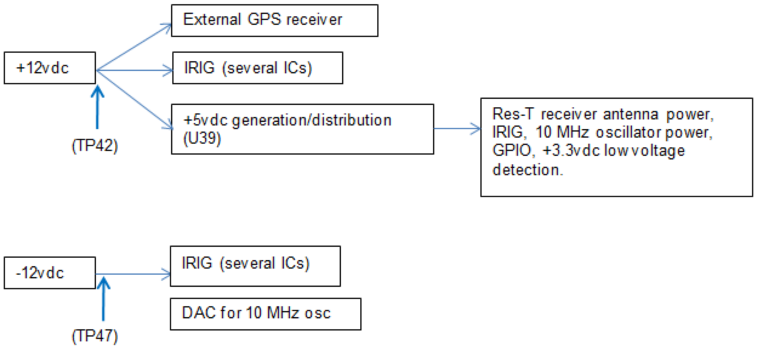
C) +5vdc generation

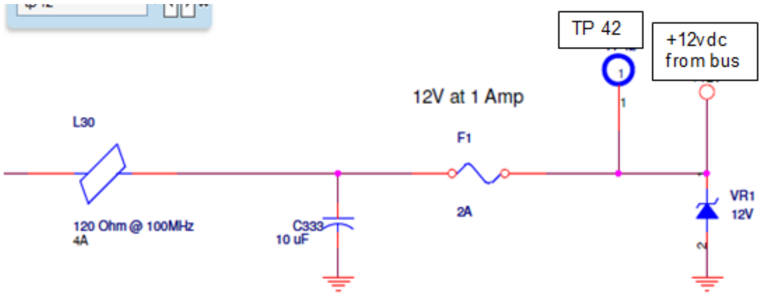
- +5vdc is generated from +12vdc input (via IC U39)
- +5vdc is used with the Res-T antenna power, IRIG, 10 MHz VCXO voltage, GPIO and +3.3vdc low voltage detection.

D) +12vdc / -12vdc distribution

E) Fuse F1: for 12vdc input from PCIe bus slot

- +12vdc generates +5vdc
- +12vdc also generates -12vdc
- +12vdc /-12vdc used with IRIG.


External GPS receiver
+12vdc
IRIG (several ICs)
+5vdc generation/distribution (U39)
(TP42)
Res-T receiver antenna power, IRIG, 10 MHz oscillator power, GPIO, +3.3vdc low voltage detection.
-12vdc
IRIG (several ICs)
DAC for 10 MHz osc
(TP47)



F) +3.3vdc power

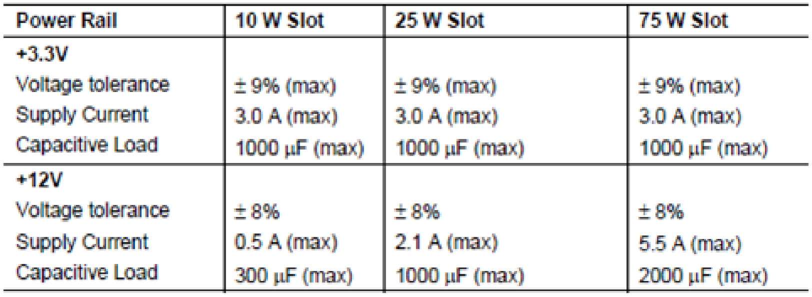
Summary: TSync-PCIe boards are designed to and compliant with the PCI specifications. Refer to **Section 4.1: Power Supply Requirements** (starting on page 35) of the "PCI_Express_CEM_1.1.pdf document at I:\Engineering\Specs and Standards\PCISIG

G) Fuse F2: for 3.3 vdc input from PCIe bus slot

- TSync-PCIe is a 3.3v power/logic card. It is not compatible with 5v systems.
- Because of this, a YELLOW key should be installed in the bus connector- to indicate it is a 3.3v board

Below for your reference are the PCIe power level specifications:

Table 4-1: Power Supply Rail Requirements

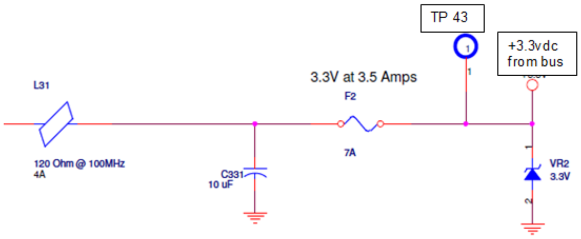


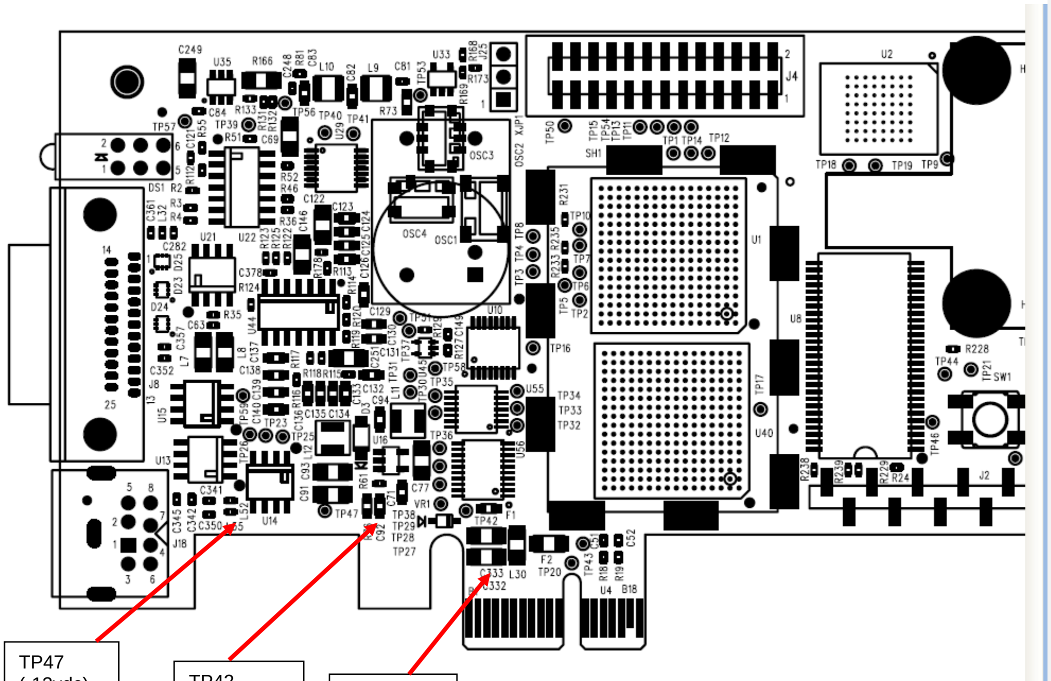
Power Rail	10 W Slot	25 W Slot	75 W Slot
+3.3V			
Voltage tolerance	± 9% (max)	± 9% (max)	± 9% (max)
Supply Current	3.0 A (max)	3.0 A (max)	3.0 A (max)
Capacitive Load	1000 µF (max)	1000 µF (max)	1000 µF (max)
+12V			
Voltage tolerance	± 8%	± 8%	± 8%
Supply Current	0.5 A (max)	2.1 A (max)	5.5 A (max)
Capacitive Load	300 µF (max)	1000 µF (max)	2000 µF (max)

Note: Based on the +3.3vdc specs, the maximum voltage drop for this rail is about 3.003vdc.

3.3vdc Low voltage detection circuit

- 3.3vdc is essential for operation of the board.
- If 3.3vdc is good, the three status LEDs on the edge of the board will all momentarily turn-on at power–up, then all will go out (thereafter, they remain dark until it syncs).
- If 3.3vdc is low, low voltage detection circuit will hold the board in reset- resulting in all of the LEDs remaining lit.
- If no 3.3 vdc present, the LEDS won't even momentarily turn on.





TP47 (-12vdc)

TP42 (+12vdc)

TP43 (+3.3vdc)

Ground: Use the metal edge of the board, which is grounded

Email from Tim Tetreault to Mike Vinskus (6/26/12)
Your testing confirms what we were thinking is the problem. The 3.3v rail is dropping causing the low voltage IC on the board to hold it in reset mode. The threshold for the low voltage IC is 3.07v +/- 0.5% typ acc. So if you do the math, our spec for the 3.3v rail is 5% or about 3.14v, just above the threshold voltage.

When all the LED's on the board are light, that means it has been reset. When the 3.3v rail recovers, the board should come out of reset mode and start running again.

Note: The tolerance for the 3.3v rail is the same between both TSync-PCIe and TSync-CPCI boards, 5%. The current draw is different.

3.3vdc input operating voltage from the PC:

Q. In evaluating our system that includes the TSync, we note that the PCI 3.3V supply to the card is only providing 3.15v. Your hardware spec says 3.3 +/-5%, which would mean it can work down to 3.13v. Any idea if running the voltage this close to the edge could cause issues? (We're trying to eliminate any possible sources of problems)

A. (From Dave Sohn) Problems with the 3.3V voltage would become very obvious as the board would be held into reset. It would lose all connectivity. If it is right on the edge, we've seen boards cycling through reset condition.

A (From Dave Lorah): The 3.3V needs to be within the tolerance. We have seen boards have problems if the voltage drops below the lower limits. The boards would either cycle through a reset condition or go into constant reset, therefore halting all board functions. I would recommend not running the board that close to the lower limit to avoid potential problems if the voltage fluctuates.

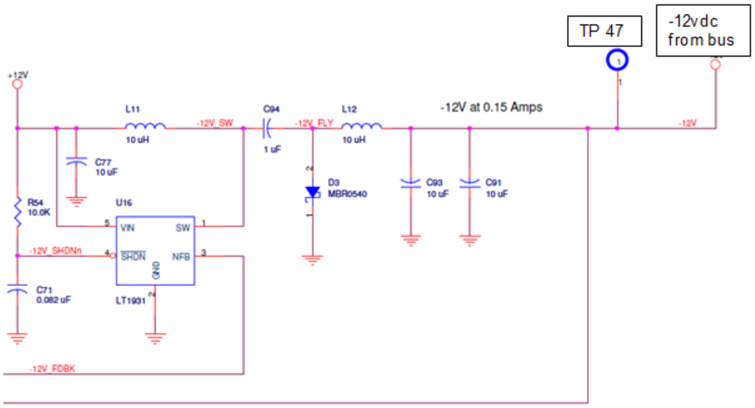
The Board may not be receiving the minimum of 3.1 vdc from the motherboard (stuck in a reset condition)

The slot that the timing board is plugged into may be a video-only slot (as either set by the motherboard manufacturer or as configured in BIOS). Example from David Hill "…. It turns out that the PCI express slot I was attempting to use was a video-only slot, so I am rounding up a new PC with an all-purpose PCIe slot".

Electrically, the slot should be no different than any other PCIe slot.   The only difference should be the BIOS looking for a video card to be present. This would, of course, prevent the board from being able to communicate with the bus. However, as long as the +3.3vdc is present, the board should at least still be able to boot-up (the LED on the board should illuminate and remain lit- not be turning off).

H) -12vdc power

IRIG input/output circuitry



Resetting the TSync-PCIe board (Reset API call) and PTP module (if installed)

- Command to reboot: **SS_Reset 0 0**
- Shouldn't ever have to reset the timing board during normal operation.
- This command is recommended for debug/diagnostics only. Causes even the PCIe Core to reset, and system BIOS may not like the Tsync board dropping off the bus. This may result in system issues.
- If it's desired to clear out the timing board, it's recommended to reset on the "way down" (as part of a shutdown script) than on the "way up" (as part of a start-up script).

Note: Rebooting the PC doesn't restart the TSync-PCIe board. Must power cycle the PC, power cycle the card or issue the Reset API call in order to restart the board.

Note: Resetting the board while other applications are accessing/trying to access the timing board can cause the machine to hang. It's recommended to stop all applications that use the board prior to resetting board. (Or just power cycle the PC).

Resets other than "0" are available, but since they only reset a portion of the board instead of the entire board, a power cycle of the TSync board would likely need to be done after the other reset calls are issued.   For example, a "**TSYNC_DRV_CONNECTION_ERR**" message may occur because the firmware is now unable to communicate with the FPGA because only part of the board was reset. A power cycle after the reset was performed should fix this issue.

Email to Jerry Walsh with Aeroflex, after he performed a reset other than "0".
As you mentioned in your email, there are a couple of different types of resets that can be applied to the TSync board. However, Reset 0 is the only one that resets the entire board and prevents the likely need to power cycle the board after the reset is performed. The other resets only reset portions of the board, resulting in the likely need to power cycle the board to get "everything back in sync with each other". This is why the other resets are not recommended to be performed. It's not a case where the other resets will cause harm to the board, but actually the Reset 0 is just to avoid the likely "inconvenience" of having to power cycle the board after performing one of the other types of resets.

API call that should always be used to reset the board without the likely need to power cycle the board is: "SS_Reset 0 0". When the card resets, you should see all three LED's on the card flash briefly, then go out. If you are connected to an active reference, you should see the green "sync" light go on shortly after.

- **Important Note about the reset command to keep the system from potentially hanging**: Per Tim Tetreault - Make sure NTP and any application software that accesses the board is stopped before issueing a reset command. If anything such as NTP is accessing the board when the reset command is performed, the system may crash/hang!!

I have a quick question (and some info) for you that may help, before we "delve" deeper into the symptoms you are observing with the Spectracom TSync-PCIe timing boards. The question is specifically about your phrase "**reset the card** it causes the system to crash and reboot". It sounds like you are resetting just the timing board and not the whole system, correct? Are you using the "**SS_Reset**" API call/example program to reset just the TSync-PCIe board after the system is already powered up? If so, besides the "0" that follows the "reset" command to specify which installed board to reset, what other number are you using in conjunction with the reset command?

Please be aware that when using the SS_Reset command, the corresponding number should be a "0", in addition to the other "0" to specify the board to reset. The full command should be **SS_Reset 0 0** if there is only one board installed in the system. Using the reset command with anything other than a 0 causes only a partial board reset of the board and could cause the TSync-PCIe board to become inoperative, until the system is power cycled. If the board becomes inoperative because of a partial reset, it could hang the PCIe bus and therefore also the system. Note that when the SS_Reset 0 0 command is issued, you should observe the three status LEDs on the edge of the TSync-PCIe board flash briefly, then all go out (until it resyncs to its input reference). If you aren't using SS_Reset 0 0 to reset the board, the LEDs will not likely operate in this fashion (they may not light at all or may remain lit).

If you aren't issuing a SS_Reset command to reset the board after system power-up, let me know exactly what is being done and then we can go from there! If you have been using a command other than SS_Reset 0 0, please let me know this resolves the issue you reported – Thanks (I appreciate you letting me know that you are all set)!!!

Reset of the PTP Module (if installed)

Also, the PTP module on the TSync-PCIe-PTP is its own embedded system that needs to be reset independently from the TSync card. Instructions on how to do that are in Section 5.7.1.1. of the Driver Guide. (Section 5.7 of the Driver Guide has lots of good information for understanding how to drive the TSync-PCIe-PTP card.)

To reset the PTP operation:
PTR_ResetModule <device> <inst> <reset type>
Reset Type 0 = Cold Reset
Reset Type 1 = Hot Reset
Reset Type 2 = Reset to Factory Defaults

Intermittent or constant system crashes/reboots occurring after a reset command is issued.

This abnormal operation could be caused in a Linux system that has ASPM still enabled (especially with the later version 5 and earlier version 6 linux distributions) ASPM should be disabled to prevent it from adversely affecting the timing board. Refer to ASPM: **(ASPM) Power Saver/Power Monitor feature** for info on disabling ASPM.

Specials

SP364: For Invensys (from the Specials database- “Add cable, update software and FPGA to emulate an PCI-U with FXA option. Refer to ECN 2341”)

Hardware status:

Run the **IN_GetStatus 0** command. Then run the **HA_GetCaps 0** command. Example responses below:

```
 [root@ntp01 /tmp]# ./IN_GetStatus 0

  Module                          | Result
  -----------------------------------------
  IRQ Driver                       | PASS
  Control Status Driver            | PASS
  SPI Driver                       | PASS
  Watchdog Driver                  | PASS
  Timer Driver                     | PASS
  Reset Driver                     | PASS
  Internal Flash Driver            | PASS
  External Flash Driver            | PASS
  LED Driver                       | PASS
  DAC Driver                       | PASS
  Digital Pot. Driver              | PASS
  EEPROM Driver                    | PASS
  UART Driver                      | PASS
  Local Bus Driver                 | PASS
  Time Control Driver              | PASS
  1PPS Control Driver              | PASS
  Disciplining Driver              | PASS
  TCB Output Driver                | PASS
  ASCII Input Driver               | PASS
  ASCII Output Driver              | PASS
  IRIG Input Driver                | PASS
  IRIG Output Driver               | PASS
  GPI Driver                       | PASS
  GP Output Driver                 | PASS
  Fixed-Freq Output Driver         | PASS
  1PPS Output Driver               | PASS
  Time-Stamping Driver             | PASS
  Component Interface              | PASS
  Persistent Data Service          | PASS
  Supervisor Service               | PASS
  Flash Manager Service            | PASS
  Clock Service                    | PASS
  Log Service                      | PASS
  Reference Monitor Service        | PASS
  Oscillator Monitor Service       | PASS
  Upgrade Service                  | PASS
  Initializer                      | PASS
  Shared Memory Service            | PASS
  GPS Reference Component          | PASS
  IRIG Reference Component         | PASS
  ASCII Reference Component        | PASS
  HaveQuick Reference Component    | PASS
  1PPS Reference Component         | PASS
  Host Reference Component         | PASS
  Self Reference                  | PASS
  IRIG Output Component           | PASS
  ASCII Output Component          | PASS
  1PPS Output Component           | PASS
  Fixed-Freq Output Component     | PASS
  LED Control Component           | PASS
  Oscillator Component            | PASS
  GP Input Component              | PASS
  GP Output Component             | PASS
  Host Agent                      | PASS
  Internal Agent                  | PASS


[root@ntp01 /tmp]# ./HA_GetCaps 0

  CAI  | IID  | Access
  --------------------
  0x25 | 0x00 | Both
       | 0x01 | Both
       | 0x02 | Get
       | 0x03 | Get
       | 0x04 | Get
       | 0x05 | Both
       | 0x06 | Get
       | 0x07 | Get
       | 0x08 | Set
       | 0x09 | Get
  0x28 | 0x00 | Get
       | 0x01 | Get
       | 0x02 | Get
       | 0x03 | Get
  0x23 | 0x01 | Both
       | 0x02 | Both
       | 0x03 | Set
       | 0x05 | Both
       | 0x07 | Both
       | 0x08 | Both
       | 0x0A | Both
       | 0x0B | Both
       | 0x0C | Both
  0x40 | 0x00 | Get
       | 0x01 | Both
       | 0x02 | Get
  0x24 | 0x00 | Set
       | 0x01 | Set
       | 0x02 | Set
       | 0x03 | Get
       | 0x04 | Get
       | 0x05 | Get
       | 0x06 | Set
       | 0x07 | Set
       | 0x08 | Both
       | 0x09 | Both
       | 0x0A | Get
  0x21 | 0x00 | Get
       | 0x01 | Set
       | 0x02 | Set
       | 0x03 | Both
       | 0x04 | Get
```

0x26	0x00	Set
	0x01	Set
	0x02	Set
	0x03	Set
	0x04	Get
0x20	0x00	Get
0x29	0x00	Both
	0x02	Get
	0x03	Both
	0x04	Both
	0x05	Both
	0x06	Get
	0x07	Get
	0x08	Get
	0x0B	Get
	0x0C	Get
	0x0D	Both
	0x0E	Get
	0x0F	Set
0x2A	0x00	Both
	0x02	Get
	0x03	Both
	0x04	Both
	0x05	Get
	0x06	Both
	0x07	Both
	0x08	Both
	0x09	Get
	0x0A	Get
	0x0B	Get
	0x0C	Both
	0x0D	Both
0x2E	0x00	Both
	0x01	Both
	0x02	Get
	0x03	Get
0x2D	0x00	Both
0x37	0x00	Both
	0x01	Both
0x38	0x00	Get
	0x01	Both
	0x02	Both
	0x03	Get
0x39	0x00	Both
	0x01	Both
	0x02	Get
	0x03	Both
	0x04	Both
	0x05	Both
	0x06	Both
	0x07	Get

Interface issues with certain systems (such as HP and Dell computers)

- Power Management (ASPM)- Related to firmware versions 2.2.1 and 2.2.2 (~Aug/Sept 2014)
- Refer to Citadel (Salesforce case 15595) and Raytheon (Salesforce cases 15473/15804)
- Earlier firmware version 2.2.0 and below not affected
- Use the F**S_GetVersion 0** Example program/API to ascertain this value

Email from Tim Tetreault (16 Sept 2014)
Here is a list of computers that I am aware of that the TSync-PCIe has been having problems with:

- HP ProLiant DL380p(Gen8)
- Dell PET320
- Dell R720
- Dell R820
- SuperMicro X9SRW-F (Same computer used for the VelaSync)

So far, these are the customers that we know for sure are using these computers and are seeing the problems:

- Raytheon
- Citadel

It doesn’t matter what OS they are using. The problems have been seen with Windows & Linux.

The symptom is on a reset of a computer, the computer could hang or the PCIe could just disappear and not be seen by the OS.

L0 and L1 ASPM modes (starting in PCIe version 2.1)

Currently, two low power modes are specified by the PCIe 2.0 specification; L0s and L1 mode.

- **L0:** A low resume latency, energy saving “standby” state. This mode concerns setting low power mode for one direction of the serial link only, usually downstream of the PHY controller.
- **L1**: The second mode, L1, is bidirectional and results in greater power reductions, though with the penalty of greater exit latency.

Very good link for Windows Vista power management settings: http://msdn.microsoft.com/en-US/library/windows/hardware/dn550976#acpi___power_management_archive (Click on “Active State Power Management in Windows Vista”

(Starting in PCIe version 2.1)

From the link above: Windows Vista supports the new Link power states that are defined in *PCI Express Base Specification* by enabling power management activities within the hardware and firmware. Enabling ASPM in Windows Vista is based on the following:

- **Hardware capabilities**. Hardware must be capable of ASPM as reported in its Link Capabilities register.
- **Exit time latencies**. The maximum exit time latency for the PCI Express hierarchy must be less than the maximum acceptable exit latency reported by the Endpoint in the hierarchy.
- **System power policy**. ASPM follows the overall system power policy in effect on the system for the PCI Express hierarchy.
- **System-level controls.** *PCI Express Base Specification* revision compliance of devices and platform firmware mechanisms may enable or disable ASPM.

- **Device-level ASPM controls.** Devices may choose to opt in or opt out of ASPM, based on known capabilities.

Good link for Dell power management BIOS settings:
http://en.community.dell.com/techcenter/b/techcenter/archive/2012/12/10/bios-performance-and-power-guidelines-for-dell-poweredge-12th-generation-servers. (Especially take a look at pages 21, 22 and 33 of the PDF). In general, I recommend not selecting any value that is associated with reduced power operation.

Timing board conversions

**Swapping out/Replacing an existing TPRO/TSAT board with a TSync-PCIe board

- Refer to Salesforce Cases such as 268908

MODIFIED Email from Tim Tetreault to Sadie (18 Nov 2013) (Note it was pertaining to TPRO/TSAT-PMC boards)

I want to make sure that everyone understands that the new TSync-PCIe board is not a drop-in replacement for the old KSI boards. It will support all of the functions that the older KSI PCI board had,but customers will need to use our new TSync driver and update their software.

Drivers

The TSync-PCIe drivers contain all of the legacy TPRO/TSAT API calls. The TPRO.lib and TPRO.h files are compatible with the newly installed TSync board, as long as there is no need/desire to take advantage of the newer API calls (such as holdover, for instance), customer application software doesn't need to be changed. But, the application software needs to be re-compiled with the TSync-PCIe driver before it will work with the TSync-PCIe board.

Note: Customer must install the TSync-PCIe driver when TSync-PCIe board is installed. The previously installed TPRO/TSAT driver won't be able to communicate with the installed TSync-PCIe board (the device ID is different). So, the TSync-PCIe driver also needs to be installed (the TPRO/TSAT driver can either be uninstalled or installed simultaneously with the TSync driver, as desired.

The customer will still need to install the associated TSync driver (available at no cost from our website). The TSync drivers contain all the legacy TPRO/TSAT API calls. The TPRO.lib and TPRO.h files are compatible with the newly installed TSync board, as long as there is no need/desire to take advantage of the newer API calls (such as holdover, for instance), customer application software doesn't need to be changed. But, the application software needs to be re-compiled with the TSync driver before it will work with the TSync-cPCI board.

Note: Customer must install the TSync driver when TSync-PCIe board is installed. The previously installed TPRO/TSAT driver won't be able to communicate with the installed TSync-cPCI board (the device ID is different). So, the TSync driver also needs to be installed (the TPRO/TSAT driver can either be uninstalled or installed simultaneously with the TSync driver, as desired).

Email Keith sent to customer 8 July 2021 (excerpted from Case 268908)

In summary: The TSync-PCIe board's driver supports all the "legacy" TPRO_ API calls to support the TSync functions that match the earlier TPRO functions. Unless you wish to take advantage of newer functions/capabilities that the TSync boards also support (in addition to the TPRO functions carried forward), there is no need to change your application software. However, you still have to recompile your application software with the TSync driver.

If you haven't already, you will still need to install the latest version of the associated TSync linux driver (available at no cost from our website at https://www.orolia.com/support/timing/tsync/pcie and scroll down to "**Software**") into the machine.

With the TSync driver successfully installed (no compile errors reported), you should be able to successfully perform the individual Example Programs provided in the TSync driver (such as SS_GetSync 0 0 for instance). This ensures the TSync board can communicate successfully with its TSync linux driver.
Attached is a copy of the TSync Driver Guide. Refer to page 2-2 for info on building the Example programs (if you haven't already) for

their use to communicate with the TSync board,

The TSync drivers contain all the legacy TPRO/TSAT API calls. The TPRO.lib and TPRO.h files are compatible with the newly installed TSync board, as long as there is no need/desire to take advantage of the newer API calls (such as holdover mode, for instance), customer application software doesn't need to be changed. But, the application software needs to be re-compiled with the TSync driver before it will work with the TSync-PCIe timing board.

Once you have recompiled your application software with the newer installed TSync driver, your application software (the associated Tpro API calls) can also be used to also communicate with the TSync board (in addition to the Example Programs included in the TSync driver).

Please let me know that you are currently able to successfully communicate with the TSync board using the Example programs provided in the TSync driver. And, once you have recompiled your application software with the newly installed TSync driver, your application software is also able to communicate with the TSync board.

GPS connection when converting from TSAT to TsyncE timing board

(**Note**- only the TSyncE can use the existing TSAT (Acutime) antenna. TsyncI requires a different antenna be installed)

The antenna connector that was originally plugging into the TSAT board now connects to an adapter cable (included in the TSyncE ancillary kit). The other end of the adapter connects to the antenna connector on the TSyncE board. for more information on the adapter cable and Acutime antenna, refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

Heartbeat output signal from a TPRO/TSAT becomes one of the four GPO pins of the TSync board

Note:much of the Info below is from the TPRO/TSAT section of this doc (much further below)

**Heartbeat output

Output pin (TPRO/TSAT board): The Heartbeat output is on **Pin 6 of the DB15** timing connector.

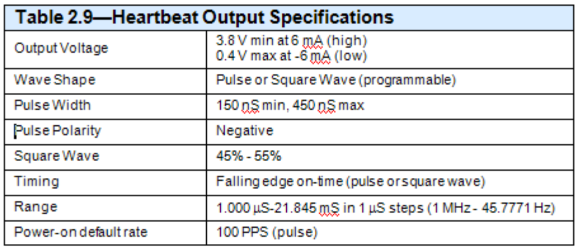


Table 2.9—Heartbeat Output Specifications	
Output Voltage	3.8 V min at 6 mA (high) 0.4 V max at -6 mA (low)
Wave Shape	Pulse or Square Wave (programmable)
Pulse Width	150 nS min, 450 nS max
Pulse Polarity	Negative
Square Wave	45% - 55%
Timing	Falling edge on-time (pulse or square wave)
Range	1.000 µS-21.845 mS in 1 µS steps (1 MHz - 45.7771 Hz)
Power-on default rate	100 PPS (pulse)

equavilent GPO 0 signal (TSync board): The GPO 0 output is on Pin ~~6~~ of the DB25 Timing connector or its on the of the breakout cable

Default heartbeat output: The power-on default for a TPRO is 100 PPS, pulse mode.The power-on default for a Tsync board is 1 PPS, pulse mode similar to HB1PPS

Allowable Heartbeat frequency ranges:

- **Without HB1PPS option enabled ("Standard" default configuration):** 45.7771Hz to 1.000MHZ

- **With HB1PPS option enabled:** 0.152593Hz to 500Hz (in exact multiples of 1ms)

Programming the heartbeat output frequency

Note: The POKE API call is used to set the register (PEEK API call is used to read the register).

- Refer to Section 7.18 of the PCI user manual for heartbeat frequencies above 1 Hz
- Refer to Section 7.19 for heartbeat frequencies of 1 Hz or less

Pulse or square wave output

The heartbeat output can be programmed for a square wave or a pulse, and can be programmed either to start immediately, or at the beginning of the next cycle. Send one of the following commands to command port:

```
0x00e5                  ; Pulse mode, starts at beginning of next cycle
0x00e6                  ; Pulse mode, starts immediately
0x00e7                  ; Square wave, starts at beginning of next cycle
0x00e8                  ; Square wave, starts immediately
```

Need to wait 8 seconds after setting heartbeat settings

Email from Tim T

Whenever you change the heartbeat to the PCI board, you should wait 8 seconds then retest for SYNC before trying to set the heartbeat again. When the board is using an IRIG as its time source, extra time is required to SYNC after the heartbeat is set due to a software filter and decoding the IRIG signal.

Desire to stop the "Heartbeat output" / GPO pin

- Refer to Salesforce case 118344
- TPRO/TSAT board's Heartbeat out can't be disabled
- Tsync board's GPO output can be disabled (but not via the legacy TPRO calls in the Tsync driver) - require the use of the more recent TSync calls in the TSync driver

**HB1PPS Option

- Desire for 1PPS (or slower) signal on the Heartbeat output (otherwise heartbeat can't go to this low of a frequency)
- Note: Refer to "HB1PPS" in the "Options" section above

~~**Email from Tim T:** To get a 1Hz heartbeat, they need to change the DIP switches to configure the board for the HB1PPS option.~~

~~**Email to customer:** In order to provide a 1Hz heartbeat output, you just need to enable the "-HB1PPS" option. This option allows for the 1Hz frequency to be available on the Heartbeat output.~~

~~The option is enabled using DIP switches/jumpers on the board. This option is described in Section 7.19, page 7-15 of the TPRO/TSAT manual (I have attached a copy of this manual, for your reference). Section 4, page 4.1 of the manual shows the location and switch positions used to enable each of the available options.~~

~~The first table on page 4.1 is for TPRO board options and the second table is for TSAT board options. Both tables contain a row for the HB1PPS option. Follow the "-HB1PPS" row in the applicable table to determine the switch and jumper settings for 1PPS output.~~

Timing outputs connected to the DB15 on the TPRO/TSAT need to move to the DB25 Timing connector on the TSync board (or to the ends of a breakout cable)

- The Pinouts for TPRO/TSAT's Heartbeat out, and the Tsync's GPO 0 output arenptso an adapter cable will likely need to be used when switching from TPRO/TSAT to a Tsync board

**Desire to convert a TSyncI to a TsyncE / TsyncE to TSyncI

Email from Dave Lorah to David Higgins (5/25/12)
We do not normally retrofit bus level boards but if you need to we can retrofit one for the following cost.

To add a GPS to your board will require you return the board to Spectracom for retrofit and reprogramming. Turnaround time is approximately 2 weeks.

To update to a TSyncI-PCIe (GPS with internal receiver) it would cost $1100. You would still need to additionally purchase a model 8225 antenna (about $250) and the appropriate antenna cable length and surge suppressor.

To update to a TSyncE-PCIe (GPS with external receiver built into the antenna) it would cost $1776.00. This would include the antenna and 100 foot cable. Please note this model cannot be used in a mobile application.

Warranty is 90 days or the remaining life of the warranty of the board, whatever is longer.

**Troubleshooting TSync boards/known issues

A) Two or more TSync-PCIe boards installed, but only one being detected

Troubleshoot: Try swapping the known good board with one that is not being recognized (put it in the slot that is recognizing the board being installed). If the other board is recognized, the board is OK. It's a system issue preventing the other board(s) from being detected.

B) kernel: You have some hardware problem, likely on the Error messages on system for the PCI bus

PCI bus.
Message from syslogd@kepler at Jan 29 14:08:15 ...
kernel:Dazed and confused, but trying to continue

- Refer to Salesforce case 17009

Per Tim Tetreault (29 Jan 15, edited by Keith) The error the customer is reporting is similar to error I have seen on the HP server before we fix the FPGA issue. If I had to guess, they are using a PCIe board that doesn't have the latest firmware. 2.2 3**. (As long as the Tsync board is NOT a Tsync-PTP board)** Have them update the board and I think it will fix their issue. (If it's a TSync-PTP board, it should be downgraded to version 2.11 instead of updating it)

C) Known potential issue associated with the TSync PCB: TSync board is "in sync", even though there are no external inputs/Self mode is not in use (Reference NC-002529)

- Refer to **ECO-FAI-1857** for info on a PCB board spin for this condtion (in Arena): https://app.bom.com/changes/detail-summary?change_id=2394867944&

Summary/Description (excerpted from ECO-FAI-001857)

TSync-PCIe board spin to implement corrective actions for NC-002529 (stop shipment due to Tsync PCB's showing sync with no connections made) and AM IRIG input circuit error discovered on VersaSync AM IRIG option card.

NC-002529 corrective action - Add 10k pull downs resistors to J8 floating inputs at U55 pins 1, 3, 5, 9, 11.

AM IRIG input circuit error - Add 11k resistor in series with U29 pin 1, 2, 14.

FIRMWARE Versions

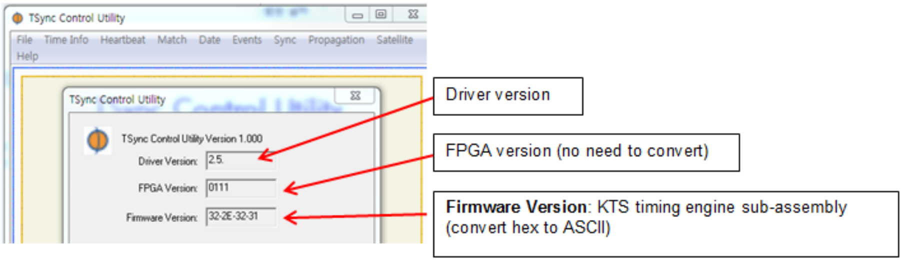
**Firmware version/updates - Firmware/FPGA/PTP module

- Link to TSync Firmware version change document (and instructions to perform the firmware upgrade):

Refer to: PSB, PSP software updates\TSYNC boards

Obtain Firmware/Software version information

- Using the Control Utility (Windows driver only)
- **Help** -> **About** menu



TSync Control Utility Version 1.000

Driver Version: 2.5.

FPGA Version: 0111

Firmware Version: 32-2E-32-31

Driver version

FPGA version (no need to convert)

Firmware Version: KTS timing engine sub-assembly (convert hex to ASCII)

D) FPGA version

- Same as the **FS_GetVersion** Example program/API call
- No need to convert the value.

Note about "0111" Tim Tetreault says this field probably won't work correctly on newer versions because the Control Utility hasn't been updated to reflect the newer FPGA versions *(such as 2.2.2 or 2.2.3 for instance).

- Use the **FS_GetVersion 0** Example program/API to ascertain this value.

E) KTS Firmware version

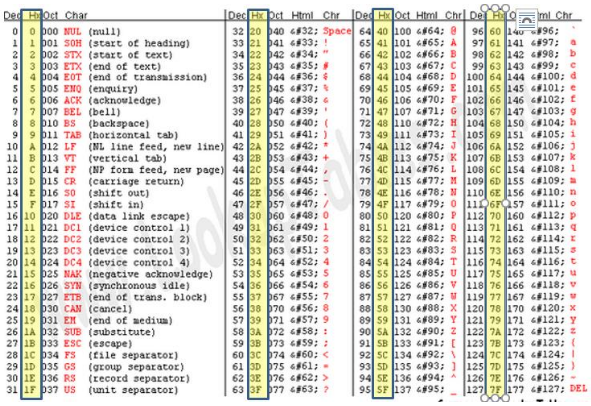
- Summary list of firmware versions: I:\Customer Service\PSB, PSP software updates\TSYNC boards\Tsync Firmware updates
- Same as the **LS_GetVersion** Example program/API call
- Reports the version of the KTS software sub-assembly
- Convert the Hex to ASCII

To convert the Firmware version, refer to a:

- using a "Hex to ASCII converter", such as http://www.dolcevie.com/js/converter.html (Use colons between each set of two numbers)

OR

- use the below “Hex to an ASCII” table, from http://www.asciitable.com/ (hex is the yellow highlight colums The ASCII corresponding ASCII value is in red)



Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT (end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1 (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2 (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3 (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4 (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC (escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

- ➢ Where “2E” is a decimal point

In this example:
“**32**” Hex is “2” in ASCII
“**2E**” Hex is a “.” in ASCII
“**32**” Hex is “2” in ASCII
“**31**” Hex is “1” in ASCII

Other hex to ASCII conversions
“**33**” Hex is “3” in ASCII

Specific firmware conversions:
33-2E-30-34 (33:2E:30:34) = version 3.04
33-2E-33-32 (33:2E:33:32) = version 3.32

- ➢ **Using the API calls/example programs**

  LS_GetVersion / FS_GetVersion (API calls for version info)

  - ➢ **Note:** For a full list of KTS and TSync firmware versions, refer to: [I:\Customer Service\PSB, PSP software updates\TSYNC boards\Tsync Firmware updates\TSync-PCIe\TSync-PCIe firmware updates](I:\Customer Service\PSB, PSP software updates\TSYNC boards\Tsync Firmware updates\TSync-PCIe\TSync-PCIe firmware updates)

(Email response from Denis Reilly 11/23/10) -Version information- It's complicated, because we are managing versions of the same timing software across various product lines. TSync and SecureSync share the same firmware, but are upgraded at different times.

To reconcile this, we created a Product-level versioning scheme, attached to each upgrade image, which is separate from the versions of the individual parts. For example, the release of TSync 2.1.0 actually incorporates KTS Firmware version 2.2.1, as well as TSync FPGA Version 2.1.0.

FS_GetVersion 0 0 (or 0 1) will get you the Product Version of the image.
(one set of arguments gets you the firmware version image version, and the other gets you the FPGA image version)

Email from Tim Tetreault (17 Mar 2014)
The only way to check the revision of the images on the board is to use the API command “./FS_GetVersion”.

“./FS_GetVersion 0 0” will return the version for the runtime image of the firmware
“./FS_GetVersion 0 1” will return the version for the runtime image of the FPGA
“./FS_GetVersion 0 6” will return the version of the EEPROM image

LS_GetVersion 0 will get you the KTS Firmware Version only.

- **Note:** For a full list of KTS and Tsync firmware versions, refer to:

  I:\Customer Service\PSB, PSP software updates\TSYNC boards\Tsync Firmware updates\TSync-PCIe\Summary of TSync firmware changes.docx

  I:\Customer Service\PSB, PSP software updates\TSYNC boards\Tsync Firmware updates\TSync-PCIe\TSync-PCIe firmware updates

Example below:

Firmware Version	**KTS version implemented**
2.2.2	3.0.4

HW_GetFpgaInfo 0 will get you the FPGA firmware Version Information only.

PTR_GetModuleInfo will get you the PTP Module firmware (or use the QuickPTP GUI program)

- For a list of PTP module versions, refer to: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\TSYNC-PCIe\TSYNC-PTP

TSync PTP Module firmware updates

Link to PTP module Firmware version change document (and instructions to perform the firmware upgrade):

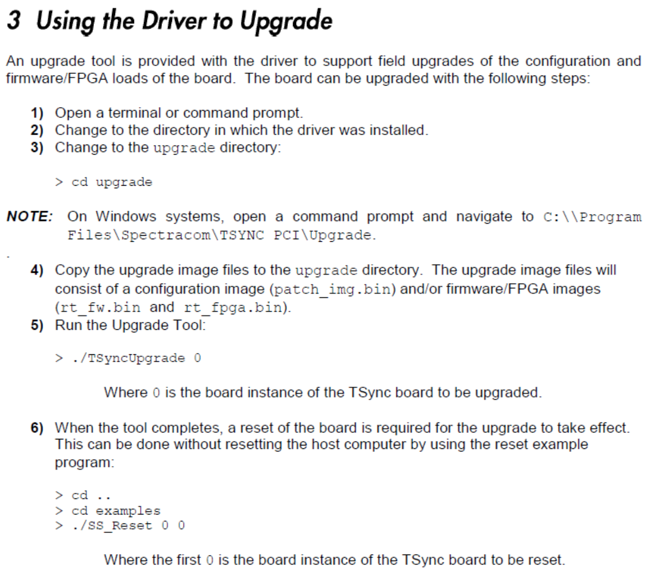
- Refer to: I:\Customer Service\PSB, PSP software updates\TSYNC boards\TSync driver updates (PCIe, cPCI, PCI104)

TSync-PCIe firmware updates/software issues

A) Updating the TSync board's firmware

- ➢ Refer to the combined TSync family driver guide for the procedure to perform a firmware upgrade of the Tsync board (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1 (Section 3, page 3-1?).

Tsync firmware Upgrade process is excerpted below



3 Using the Driver to Upgrade

An upgrade tool is provided with the driver to support field upgrades of the configuration and firmware/FPGA loads of the board. The board can be upgraded with the following steps:

1) Open a terminal or command prompt.
2) Change to the directory in which the driver was installed.
3) Change to the `upgrade` directory:

```
> cd upgrade
```

NOTE: On Windows systems, open a command prompt and navigate to `C:\\Program Files\Spectracom\TSYNC PCI\Upgrade`.

.

4) Copy the upgrade image files to the `upgrade` directory. The upgrade image files will consist of a configuration image (`patch_img.bin`) and/or firmware/FPGA images (`rt_fw.bin` and `rt_fpga.bin`).
5) Run the Upgrade Tool:

```
> ./TSyncUpgrade 0
```

Where 0 is the board instance of the TSync board to be upgraded.

6) When the tool completes, a reset of the board is required for the upgrade to take effect. This can be done without resetting the host computer by using the reset example program:

```
> cd ..
> cd examples
> ./SS_Reset 0 0
```

Where the first 0 is the board instance of the TSync board to be reset.

B) Updating the Tsync board's onboard Res-SMT-GG receiver

- **such as the version 1.09 update for the leap second issue, for example**

  - ➢ Refer to the Tsync receiver update bundle file on our website: https://spectracom.com/support/tsync/tsync-pcie-support (Scroll down to "**Support Bulletin**" Then click "download Update file here")

  - This bundle consists of update instructions and one update file for the Res-SMT-GG receiver ("rt_Gnss0E.fw.bin").

Prerequisties to be able to update the GNSS receiver in the files (as indicated in the update instructions)

3. **The TSync board's firmware version must be version 3.3.2 or higher (as reported by the FS_GetVersion 0 0 example program)**

   And

4. **The Driver version:**

   A) The Linux driver needs to be at version 3.2.0 or higher.

   B) The Windows driver needs to be at version 3.21 or higher.

Known issues with RES-SMT-GG receivers

****(Sept 2016) RES-SMT-GG receiver firmware version 1.09 (1.9) doesn't resurvey/resync after being relocated**

- ➢ Refer to the Res-SMT-GG receiver section of the custserviceassistance doc for details
- ➢ Have to delete GPS position after GPS Time data will be valid (green) but GPS PPS will remain not valid (red) unit position is deleted in the **Interfaces** -> **GNSS 0** page of the browser.relocating receiver before it will go into sync (even though its tracking satellites just fine)

(June 2015) firmware versions 1.0.7 and 1.0.6 (fixed in version 1.0.8)

Note: As of 30 June, version 1.08 has not been made available for field update and hasn't been shipped in any units

A) Loss of GPS reception for just one or a couple of seconds

Email from Paul Myers (30 June 15) Results in testing thus far show the RES-SMT-GG v1.08 improves PPS Stability when using GPS+GLO. Also our observation of Qualification Logs on SecureSync's shows good tracking performance with multiple SecureSync's showing no drops to 0 satellites when running v1.08, while a control unit running v1.07 does show a few drops to 0 satellites for 1-2 sec.

Tim T has been doing board testing with 1.08 and we should check to see if he is seeing any 0 satellites log entries.

V1.08 will be in the upcoming SecureSync 5.3.0 release.

B) Poor 1PPS performance when Glonass is enabled

- For SecureSyncs, this only applies with units that have Glonass license installed and have Glonass enabled
- **Short term fix (as of 30 Jun 2015) until a field update is available** is to disable Glonass, if possible. Or the GNSS receiver can be returned to us for update to 1.0.8 if Glonass reception is necessary.

(Sept 2014) RES-SMT-GG receiver firmware version 1.0.6

Note: All of the receiver issues listed below were resolved with the RES-SMT-GG Receiver firmware upgrade version 1.0.7, implemented with the version 5.1.7 and above software update.

C) Potential for RES-SMT-GG receiver (v1.06 firmware) to cause time jumps of several seconds in TSync boards

Potential Issue: Res-SMT-GG receivers can potentially start to output the wrong number of seconds of UTC offset for a period of time, resulting in System Time jumping by the error in this value, when the TSync board is synced to GPS. The GNSS receiver automatically corrects the time with the next ephemeris download (within 12.5 minutes).

- ➢ Can affect TSync boards with Res-SMT GG receiver (cut-in sometime after18 March 2014 and when firmware version 1.0.6 is installed in the receiver, while synced to GPS.
- ➢ Can occur in software Tsync firmware versions 2.2.1 through at least version 2.2.3 (when a RES-SMT-GG receiver with version 1.06 firmware is attached to the Tsync board).

Note: As of Dec, 2014, we are still shipping the receiver with v1.06 software installed

- ➢ Res-T receiver can start to output the wrong number of seconds of UTC offset for a period of time, resulting in System Time jumping by the error in this value, when the board is synced to GPS. The GNSS receiver automatically corrects the time with the next ephemeris download (within 12.5 minutes).

- Fix will require a Tsync software update or Trimble releasing a software update for the GNSS receiver (but this won't be able to be applied in the field with software version 2.2.1 and below in the board).
- Crabel Capital observed a 12 second time jump occur on a SecureSync for about 6 minutes. Then it automatically corrected (refer to Salesforce case 15783).

D) 1PPS jump of a few hundred milliseconds

- Refer to Mantis case 2935 http://cvsmantis.int.orolia.com/mantis/view.php?id=2935
- **Summary**: 1PPS Jump in RES-SMT-GG might be caused by a position change in receiver calculations.
- **Description**: The unexpected 1PPS jump to a few hundred msec off of the GPS 1PPS 'might' be caused by an intermittent error in the position calculation causing the GNSS receiver to incorrectly adjust 1PPS.

E) Erratic leap second warning asserted

- Reportedly fixed with the Res-SMT-GG version 1.0.7 firmware update

Note: This erroneous leap second being scheduled is not likely to be applied to the system. So it's highly unlikely to affect operation of the SecureSync. With versions 5.1.7 and below, and as long as the System time scale is still set to UTC, the leap second is only applied if the amount of leap second to be applied is "+1" or "-1"/.

The erroneous scheduled leap second from version 1.06 receiver firmware is instead setting this to a much higher value than 1, such as +6 or +72 for examples. Even though the leaps second is scheduled, this bad data won't result in a leap second correction occurring (unless its erroneously set to a 1). And the receiver is scheduling out the leap second, long after the one scheduled for June 2015. So it will be cancelled out by the real leap second.

However, if the system is in GPS or TAI timescale with version 5.1.7 or below installed, the leap second will take if
scheduled before June 2015. The Jan 2015 release is adding additional protection that the system will only accept a 1 second change no matter what the system timescale is set to.
The leap second scheduled for June 2015 will override/delete any erroneous scheduled leap seconds. Or a user can manually delete with the browser, if they wish.

Per Dave Sohn (8 Jan 15, referring to v5.1.7) Better protections for this are already in place and will be included in the next release (referring to 5.2.0 release in Jan 2015). However, despite the reporting of the leap second, a leap second of greater than one second will not occur if the unit is operating in a UTC timescale, which is the default for SecureSync. Also, a leap second can be removed from the system, if present, by using the delete mechanism from the Edit Leap Second window on the Management -> Time Management page of the browser.

- GPS goes not valid every other second (if no other references, goes into Holdover every other second)
- Likely (but not confirmed) fixed with the Res-SMT-GG version 1.0.7 firmware update, being made available in the SecureSync version 5.1.7 update (Nov, 2014). Emmanuel observed this same issue with an Epsilon clock with a Res-SMT-GG receiver installed.
- Cold reset of SecureSync with RES-SMT-GG results in return to last saved GNSS Receiver Mode
- **Refer to** Mantis case 2932 http://10.30.1.21/mantis/view.php?id=2932
- **Summary**: A Cold Reset deletes all RAM data and acts like a power cycle of the RES-SMT-GG (or RES-T).

Desire to downgrade to an earlier firmware version

- The version desired to downgrade to can't be an earlier version than the version it was at when it was originally purchased/shipped
- For more info, refer to: I:\Customer Service\PSB, PSP software updates\TSYNC boards\Tsync Firmware updates\TSync-PCIe\Downgrading firmware.docx

Note: In order to downgrade TSync boards in the field, the desired version to downgrade does NOT need to be at the same or a newer version than the version that the board was originally shipped at. The boards can be downgraded to any firmware version (other than phase 1 from a very long time ago) ,.

Denis Reilly confirmed that the runtime image versions are not checked or compared with each other, and when the updates are applied, both are updated at the same time

Desire to downgrade version 2.x firmware to version 1.x firmware in the field

Due to significant changes between the two major revs, this downgrade has to be performed at the factory.

Email KW sent to Arnaud with Acquisys (5/17/12)
I have been discussing this desired firmware downgrading of the TSync-PCIe boards with our Engineering team and have some information for you.

To begin, due to the significant differences between the earlier TSync-PCIe firmware releases (versions 1.x) and those firmware versions that were released after a major firmware upgrade was incorporated (firmware versions 2.x), the TSync-PCIe firmware in newer TSync-PCIe boards can no longer be successfully downloaded in the field (even though the reported version may indicate it was downgraded to an earlier version, the steps to downgrade it are corrupting the contents of the EEPROM). This firmware downgrade must now be performed at either our facility here in the US, or at Spectracom France.

Before the version 2.x major firmware revision was incorporated, earlier versions of firmware in the version 1 series could be downloaded in the field. However due to significant changes that were implemented in the newer 2.x versions, the newer TSync boards can no longer be downgraded in the field. These three boards that have had the attempted downgrade performed will need to be returned to Eric and his team for a factory reprogramming. And in the future, if the desired earlier version of firmware is inadvertently not specified at time of order, those TSync boards will also need to be returned to Spectracom France for reprogramming. Ordering them with the desired, specific version of firmware installed will alleviate this requirement.

****Phase 2 firmware upgrade**

- ➢ ECN 2299 cut-in date to incorporate the phase 2 upgrade- 4/1/09.

Added the following changes

- Oscillator Disciplining
- IRIG Output (AM/DCLS)
- Supports formats A, B, G, E, and NASA 36
- Signature Control
- External GPS Support
- General Purpose Output
- Direct Value – Can put out Logic Low Or Logic High
- Square Wave – Put out digital square wave with variable polarity and pulse width aligned to the 1 PPS (offsets available)
- Time Match – You can set the outputs to go high or low at specified times in the future (up to 100 days out)
- 1PPS Output Control
- Signature Control
- Polarity – Rising or falling edge PPS
- Pulse Width - Defaults to 200 ms, but adjustable 50 ns to 900 ms
- Offset: -500 µs to +500 µs
- 10MHz Output Control
- Signature Control

Firmware updates/driver updates

With the exception of when updating the TSync-PCIe firmware from version 1.x to 2.x, Denis Reilly recommends updating the TSync-PCIe driver to the latest version (if it isn't already) before performing firmware updates. When updating from a 1.x version of firmware to a 2.x version, he recommends updating the firmware version first and then updating the driver.

Note: Steps/directions to perform a TSync-PCIe firmware update are in listed in Section 3-1 of the TSync-PCIe driver guide.

PTP module updates

FTP
The PTP module firmware updates use FTP to transfer the update file from a PC connected to the PTP port on the PTP module. The SecureSync PTP Option Card has an FTP enable/disable field that needs to be set to enabled to perform the update. PTP on the TSync module is always enabled (PTP can't be disabled on the TSync board).

TSync-PCIe FPGA program header (J25) / LED's (Programming and Status) / LEDs upon reboot

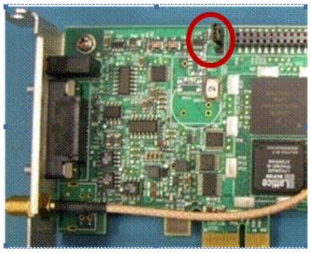
Note: this section specific to **TSync-PCIe** boards. For other variants (such as TSync-PMC, TSync-cPCI, etc) refer to its specific section of this document

F) FPGA programming header (J25) and FPGA load status LED (DS2)

1. Programming header (J25)

Verify the red "FPGA" loaded LED is lit (verify Jumper J25 is in the correct position)

Check jumper setting on J25 and make sure that it is configured on the pins to the outside edge of the board, as seen in the red circle in the photo below:



If this jumper is in the wrong position, on a reboot the timing board will never look for the new images and will always load the bootloader (Green and Red LED s will be blinking and the Yellow LED will be not lit). Please place this two position jumper as shown in the picture above (with the jumper across the two outer pins (closest to the edge of the TSync-PCIe board.

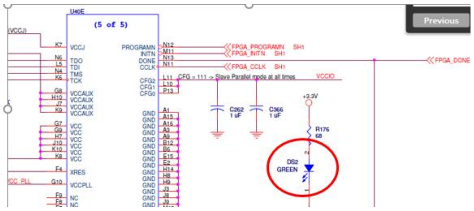
There is also a **green** "load" LED on the back-side of the TSync-PCIe boards (in the same area of the board as the jumper described above, but on the other side of the timing board. This LED should light shortly after the TSync-PCIe board powers-up and then should remain lit, from that point forward. Please let me know if this LED turns on and then remains lit green while the PC is on.

Right after the TSync-PCIe board powers-up, the three LEDs on the edge of the board (red, yellow and green) turn on for just a moment and then turn off. After that, the LEDs will remain not lit, until a valid input reference (IRIG, GPS, or HST, Self) becomes available.

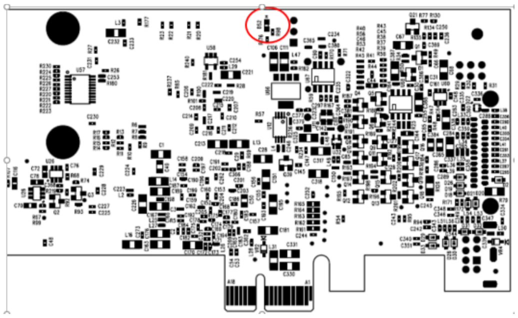
2. Green FPGA Progam LED (DS2)

- ➢ Green LED located on **bottom** of PCB board (same underneath area as J25 program header)

(*Below is from 1191-1001-0200*)



(*Below is from 1191-1000-0200*)



DS2 LED Blink/flash patterns indicating a potential issue

- *The code indicates the fault condition. It blinks the number of times indicated with a 2-second pause between each set.*

1 Blink (1 Flash) = FPGA programming error

2 Blinks (2 Flashes) = Failure to decompress

3 Blinks (3 Flashes) = CRC failure writing to flash

4 Blinks (4 Flashes) = Self-test failure

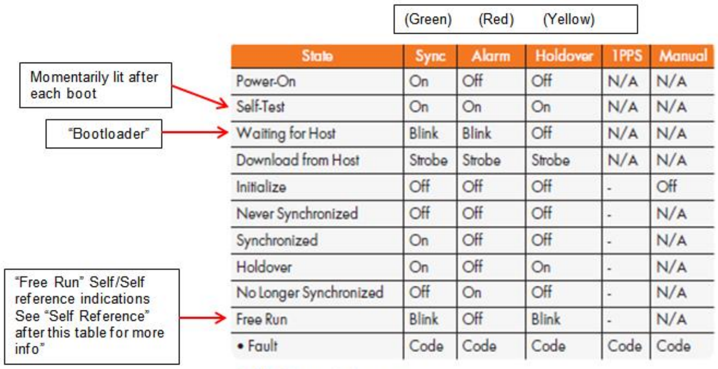
5 Blinks (5 Flashes) = Timing system failure (**Note**: refer to SR 5558 in SAP for an actual example)

G) DS1: Three Status LEDS on edge of TSync-PCIe board (red, yellow and green)

DS1 Status LED states during operation

- **Green**: Sync LED
- **Yellow**: Holdover LED
- **Red**: Alarm LED

NOTE: During the power-on, self-test, wait-for-host, and download-from-host states, modes are directly allocated to the LEDs (sync is green, holdover is yellow, and alarm is red). During normal operation, the user may set any LED to any of the operational modes.



(Green) (Red) (Yellow)

State	Sync	Alarm	Holdover	1PPS	Manual
Power-On	On	Off	Off	N/A	N/A
Self-Test	On	On	On	N/A	N/A
Waiting for Host	Blink	Blink	Off	N/A	N/A
Download from Host	Strobe	Strobe	Strobe	N/A	N/A
Initialize	Off	Off	Off	-	Off
Never Synchronized	Off	Off	Off	-	N/A
Synchronized	On	Off	Off	-	N/A
Holdover	On	Off	On	-	N/A
No Longer Synchronized	Off	On	Off	-	N/A
Free Run	Blink	Off	Blink	-	N/A
• Fault	Code	Code	Code	Code	Code

Momentarily lit after each boot

"Bootloader"

"Free Run" Self/Self reference indications See "Self Reference" after this table for more info"

· Table 5-1: LED Flash Patterns

Associated questions to ask/info to obtain if the LEDs are indicating one of these patterns

1) Does this happen right away on power-up of the sytem the board its installed in? Or does it happen after some time of normal operation?
2) Serial Number of the board and the installed firmware version (FS_GetVersion 0 0)
3) Does the board still operate while in this abnormal state
4) Does it respond to the get log call of: LS_GetMessage 0

In Sync mode

Green LED- Lit solid
Yellow LED- Not lit
Red LED- Not lit

In Holdover mode

Green LED- Lit solid
Yellow LED- Lit solid
Red LED- Not lit

LED indicators when using Self reference for sync (Self/Self Reference)

Green LED and Yellow LED- blink simultaneously
Red LED- Not lit

The Self/Self reference has a unique LED pattern where the green (sync) and yellow (Holdover) LEDs will both blink simultaneously) when using the Self reference for sync. In at least the Rev E and below versions of the TSync-PCIe manual, this pattern is indicted in the "LED Flash patterns" table as "Free Run".

While in this flash pattern, Sync will still indicate true and Holdover will indicate false. This is to indicate the TSync-PCIe is being treated as if it's in continuous sync, as well as indicating the oscillator is in free-run mode (oscillator is not being disciplined).

Troubleshooting based on LED status

Check the Status LEDs on the edge and the green “FPGA load” LEDs on the back side of the TSync-PCIe board:

- Verify all three Status LEDs (red, yellow and green) on the edge of the TSync board turn on and then go out (and remain not lit until an input reference becomes valid).

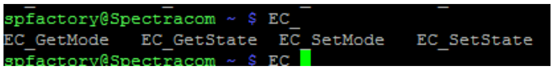
Right after the PC with the TSync-PCIe board powers-up, the three LEDs on the edge of the TSync-PCIe (the red, yellow and green LEDs), should momentarily illuminate and then turn off (until an input reference is connected). Do all of these LEDs momentarily turn on when the PC is first booted-up?

If the LEDs are not lit at power-up /all three constantly lit/ exhibit the correct pattern:

- First, try to re-seat the TSync-PCIe board in the system to make sure that all the contacts are solid.
- **If the green and red LEDs are blinking on your PCIe card,** it is likely that the board is in the bootloader waiting for image downloads from the host. This could signify that either the board needs to be reprogrammed or that there was a failure in the flash forcing it to fail to load the run-time or backup default images, landing back into the bootloader.
- **If all three LEDs remain constantly lit** (instead of all being lit only momentarily) Microprocessor likely “hosed”
- **If all three LEDs flash five times** (customer reported this intermittenty happening)
    - Refer to SR 5556 in SAP

Controlling the three (Red,yellow and green) Status LEDs

The “EC_” commands are used to change the LED operation from their defaults.



```
spfactory@Spectracom ~ $ EC_
EC_GetMode    EC_GetState  EC_SetMode    EC_SetState
spfactory@Spectracom ~ $ EC_
```

From the Tsync driver guide

Where display ‘**state’**
0 = LED **off solid**
1 = LED **on solid**
2 = LED **blinks on/off** (at a 2 Hz rate)
3 = LED **blinks a code** (2 Hz rate with a 2 second pause)

Where display ‘**mode’**
0 = Display **Sync** status (factory default for Sync LED)
1 = Display **Holdover** status (factory default for Holdover LED)
2 = Display **Alarm** status (factory default for Alarm LED)
3 = **Blink on 1PPS**
4= Switch to manual control of LED

Where ‘**led’**
0 = Alarm (Red) LED
1 = Holdover (Yellow) LED
2 = Sync (Green) LED

TSync defaults
Green LED: Sync status
Yellow LED: Holdover status
Red LED: Alarm status

Available “EC” calls for getting/setting the LEDs

- ***TSYNC_EC_GetMode*** Get the LED usage mode state. (syntax: **EC_GetMode <device index> <led>)**
- ***TSYNC_EC_SetMode*** Set the LED usage mode state. (syntax: **EC_SetMode <device index> <led> <mode>)**
- ***TSYNC_EC_GetSate*** Get the LED display state. EC_GetState (syntax **EC_GetState <device index> <led>)**
- ***TSYNC_EC_SetState*** Set the LED display state. Settable only in manual LED mode (syntax: **EC_SetState <device index> <led> <state>)**

Example usage:

To take manual control of an LED:

EC_SetMode 0 <led> <4> (where 4 is for manual control of this LED)

Then change the state of each LED, as desired:

EC_SetState 0 <led> <state> (example: EC_SetMode 0 0 3 will cause the Sync LED to blink once a second)

MEMORY/Certificate of Volatility (COV)/ Letter of Volatility (LOV)

**Letters of Volatility/Sanitization/Memory storage after board is powered-down and removed

- Customers can refer to: https://support.spectracom.com/bulletins/tsync-certificate-volatility
- On our website at: https://support.spectracom.com/sites/kb_files/bulletin_docs/tsync%20letter%20of%20volatility.pdf
- "**COV-TSync**" in Arena at: https://app.bom.com/items/detail-spec?item_id=1277556100&version_id=11523404078
- Also refer to the documents in: \\ROCFNP01\IDrive\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\TSYNC-PCIe\MRG real-time Linux\Memory (All Tsync boards)

**On-board memory

The Spectracom TSync-PCIe bus-level timing board is a module which can be installed in a PC to provide precision timing and interrupt functionality for a PC.

The TSync board contains a few persistent memory devices on the board. There is an 8 KB EEPROM, an 8MB Flash, and an internal 512KB Flash in the microcontroller.

A) 8MB FLASH Memory

The 8MB Flash contains the run-time, default firmware and FPGA images; these FPGA images are binary files used to store the software and FPGA images used by the board during operation. These images are upgradeable via Spectracom-provided upgrade images using specific upgrade API calls.

Notes about the 8MB Flash memory

- The flash memory is a surface mount (not socketed) IC.
- This IC can be removed/replaced, but special steps are required upon a new IC being soldered onto the board in its place (this IC does not need to be pre-programmed before its installed, but other steps are needed after its been replaced- the board would need to be returned to us or special instructions would have to be supplied and performed in the field).
- If a customer desires to remove this IC and then send the TSync-PCIe board to us for "recovery":
- If the TSync board needs to be returned to us for a warranty type repair and the customer desired to remove this IC, we should treat this as a warranty modification.
- If the customer desired to replace the IC "just because" its being removed from a sensitive area, this should be treated as a non-warranty modification (time, materials and shipping fees should apply).

Partial email Keith sent to Jarred Gallagher with Raytheon (12/17/12)
Regarding your question as to if the flash memory can be removed from the TSync-PCIe board: The flash memory is a surface mount IC soldered to the TSync-PCIe board (it is not a socketed component). This IC can be removed from the TSync-PCIe board, if desired, and then it can be replaced with a new IC (this IC does not need to be pre-programmed before it's installed on the board). Recovery of the TSync-PCIe board with this new IC installed would then require either special instructions to be performed in the field, or the TSync board would need to be returned to the factory for "recovery". As a reminder, no user data is stored in the Flash memory.

B) Internal 512KB flash in the microcontroller

The internal 512KB flash in the microcontroller contains the boot loader firmware and a compressed FPGA image. This is not accessible during normal operation.

C) 8KB EEPROM

The 8KB EEPROM contains configuration data, which write the following during normal operation:

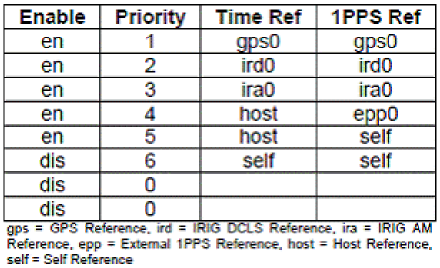
- Status flags for the images stored in the 8MB Flash, these are updated during the upgrade process.

- An uptime counter, which tracks the powered-on time of the board.
- The user reference table, which can be stored by a command from the user. This stored table can be reset to the factory default or overwritten by the user.

No other run-time information (time, position, user configuration) is persisted across power-cycles.

Resetting the 8k EEPROM: The user-configurable reference table is saved to EEPROM via the ***TSYNC_RS_SaveUserDef*** API call/example program. These values are reset to factory defaults via the ***TSYNC_RS_SetFactDef*** API call/example program.

The following was an email from either Denis Reilly or Dave Sohn
The reference table is the reference priority entries used to determine which references, and in which order references are used to synchronize the card. The user can choose to modify the default table and store it to persistent data in order to restore the modified settings on power-up. Below is an example reference table from the TSync-PCIe manual:



Enable	Priority	Time Ref	1PPS Ref
en	1	gps0	gps0
en	2	ird0	ird0
en	3	ira0	ira0
en	4	host	epp0
en	5	host	self
dis	6	self	self
dis	0		
dis	0		

gps = GPS Reference, ird = IRIG DCLS Reference, ira = IRIG AM Reference, epp = External 1PPS Reference, host = Host Reference, self = Self Reference

Table 5-2 — Example Default Table

Also, I'm not sure what they want us to sanitize on return. We generated a listing of the persisted memory and the data that is contained within. I don't think we take any special steps to sanitize anything on return. There shouldn't be any information that the user can store that would require sanitization. If they need something beyond that I'm sure we're willing to listen to any requests.

Partial email Keith sent to Jarred Gallagher with Raytheon (12/17/12)
Regarding your question "Where can I get step by step instructions to reset the EEPROM to factory defaults?": The EEPROM stores only the following Runtime items- 1) the "Image OK" status 2) the Uptime counter 3) the User Reference Priority table (this is the list of input references that the TSync board can sync with and their priority of selection for its synchronization). Everything else stored in the EEPROM is limited to factory configuration information (such as oscillator calibration information).

The only item in the EEPROM that can be restored to factory default settings, if desired, is the User (Input Reference Priority) table. This input reference selection table (which has very specific, limited available entries- Such as "GPS 0", "IRIG 0" and "EPP 0" for examples) can be reset (changed), if desired. To reset this User table, the User table can be modified/saved with different values than the currently stored values. For example, if the Factory default table has been modified and saved (resulting in a User table being created in the EEPROM) that lists "GPS 0" as a Time reference, and tied with an External 1PPS input ("EPP 0") , as the first entry in this table, this particular "ID (row of the table) can be reconfigured as " GPS 0" /" GPS 0", for instance, and then the table saved. The User table will then no longer have the currently configured values.

The User table configuration uses the "**RS_"** API calls. Attached is a copy of the TSync-PCIe driver guide. Refer to Section 4.2.24 (starting on page 4-218) for a list of all of the available RS commands used to configure the User (and Working) Reference Priority tables.

**System Memory allocation

Email from Tim Tetreault (7/10/12) "Our TSync-PCIe board currently allocates 512 bytes in memory".

**Limited NVRAM/most configs such as present year don't persist power cycle

- Refer to Salesforce Cases such as 282902 (and earlier cases such as 17773

Per your inquiry
"We are synching on IRIG B which does not include the date. We can manually set the year and it appears to work fine. Upon recycling power, the first time the card sets the PC system time, the year goes to 2000 and the day is one day behind. We cannot find how to fix this in the manuals. We need a solution ASAP. Thanks for your help."

I have some information for you…

In summary of all the info below, for security reasons, the TSync-PCIe board has very little onboard NVRAM to remember configurations upon loss of its input power. The only configurations in the TSync-PCIe board which persist thorough loss of it is input power is its "Saved" Reference Priority table (this allows for minimal onboard non-volatile memory). All other values, such as the year/date/time, configurations for its IRIG input, output configurations, etc, need to be reset each time the board powers-up.

FYI-: there are three types of Reference Priority tables for the TSync board – the default, working and saved tables. The saved table is the only configuration saved between reboots. If anyone had modified the working table (since the last time the board was powered-up,) by either disabling GPS input or deleting the GPS entry from the table, and then saved the table, this new saved table is then automnatically used as the priority table after each power-up. Your code does not need to call a table at start-up. Otherwise the default table is always used at start-up (the working table is lost upon power cycle, unless it's saved before power is lost).

TSync board users often create their own custom start-up scripts to run at each power-up of the board. Custom scripts consist of the particular example programs (provided by the Timing board driver) that are used to configure the TSync-PCIe board as desired,with its necessary configurations. The custom start-up script can optionally initially set the current year/date/time using specific example programs, as well as configure the IRIG input as appropriate for the IRIG signal its receiving. Once a valid IRIG signal is being received, it will then overwrite any year/date/time that was initially set using the start-up script.

As long as the Windows/Linux system the TSync-PCIe board is installed in is only warm re-booted (not power cycled) the TSync board will remain powered-up and operating through the system reboot. As the board doesn't lose input power in this scenario, the TSync board's operation is not affected and no settings will be lost. However, if the system it's installed in is cold booted or power cycled, the TSync board will inherently be powered-down. This will cause most of its configurations to be reset to its default settings.

As mentioned above, the year and date information does not persist through each power-down. So it must to be set upon each reboot. The default year value at each power-up is **2000**, with the date and time counting up from **January 1st at 00:0:00** until the year, date and time have been set:

1) via the external reference (GNSS or IRIG input)
2) set via API calls/example programs provided by the driver.
3) via the Windows Control Utility in the driver (when used in a Windows machine).

When a supported format IRIG signal is applied to the IRIG input, as long as the TSync-PCIe board is configured correctly for the specific parameters of the IRIG signal, it will automatically sync to that input. It is important that the IRIG signal be generated by a reference that is able to provide a very stable IRIG output. The board needs a stable input in order to be able to lock to it. This usually requires the IRIG be generated/supplied by a IRIG source that is externally synced to GPS or has a very stable 10MHz oscillator inside, such as a OCXO or Rubidium oscillator. If there is excessive jitter on the IRIG input, the card may not be able to discipline to it.

Unlike GPS input, which always provides the TSync board with the current year value (such as "2022" for instance), the current year information is not a required field in the IRIG specifications. For this reason, some IRIG outputs contain/provide the present year, while others do not. Whether or not the year is included is up to the manufacturer of the IRIG generator. When the year is provided, it's contained in the Control Field section of the signal. The TSync-PCIe board can be configured to either read the control field if the year value is provided in the RIG signal, or it can be configured to ignore this section of the signal. Also, the year value can be in a different location inside the Control Field, so the layout of the control field can also be configured.

If the year value is not able to read by the TSync board (either because its nor provided in the IRIG signal or the TSync board is not configured to read it), and if GPS is not present, the year value must be set in the board each time it powers up, before the board can sync to the IRIG input (the current year value is not retained through a reboot/power cycle of the board so if it can't read it from the IRIG and GPS is not present, it has to be set each time it powers up).

If the year value cannot be obtained from the IRIG input signal and GPS reception is not available, the current year can be set a couple of different ways. When installed in a Windows PC, the year can be set via the Windows Control Utility (included as part of the Windows driver) or via API calls. When installed in Linux or Solaris machines, the year value can be set via an API call in the driver. The default year value for the board is 2000.

When setting the year via the Windows Control Utility, the year is set in the Date -> Set Year menu (as shown below): Note the year value (along with the time and date) can be retrieved via the Date -> Retrieve Gregorian Date menu.

If the year value is not able to be read from the IRIG input and GPS reception is not available, the year can also be manually set using the CS_setYear API call/example program.

- The example program to **set** the year is CS_SetYear 0 xxxx <enter>. (where xxxx is the present year)
- The example program to **read** the year is CS_GetYear 0 <enter>.

Temperature Sensor (firmware versions 3.4.7 and above)

- **Firmware version 3.4.7** added support for temp sensor on TSync-PCIe boards
- **PCB rev 6 and higher** have a temp sensor on the PCB board.

- Per the v3.4.7 update Release Notes: “Added support for temperature sensor on new revision TSync-PCIe boards.”
  - Temperature sensor in on **PCB boards. revision 6 or higher.**

  This board has a temperature sensor built-in. So the drivers now have a call for this function.

  TSYNC_HW_GetTemperature() Reads the current board temperature in counts.

Input references / Time Sync / Holdover

**Input Reference Priority tables

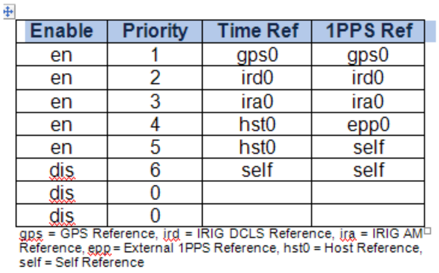
Three Reference Priority tables are maintained by the system:

- **0** = **Default factory table** (this table never changes)
- **1** = **User Default (Saved)** table that persists across reboots
- **2** = **Current working table** (becomes the new User Default table when saved, or lost upon reboot/power- down)

FYI-: there are three types of Reference Priority tables – the default, working and saved tables. The saved table is the only configuration saved between reboots. If anyone had modified the working table by either disabling GPS input or deleting the GPS entry from the table, and then saved it, this new saved table is then used as the priority table after each reboot. Your code does not need to call a table at start-up. Otherwise the default table is always used at start-up (the working table is lost upon power cycle, unless it's saved before power is lost).

A) Default Factory table (Table 0)

- This table never changes
- The Factory Table will load on startup if there is no User table defined.
- A default (factory) table, which provides the default reference pairings in timing accuracy priority



Enable	Priority	Time Ref	1PPS Ref
en	1	gps0	gps0
en	2	ird0	ird0
en	3	ira0	ira0
en	4	hst0	epp0
en	5	hst0	self
dis	6	self	self
dis	0		
dis	0		

gps = GPS Reference, ird = IRIG DCLS Reference, ira = IRIG AM Reference, epp = External 1PPS Reference, hst0 = Host Reference, self = Self Reference

Table 5-5-2 — Example Default Table

B) User Default (Saved) table (Table 1)

- persists across reboots
- A user table, which can be stored persistently and, if present, will be loaded into the working table at startup.

To reset the User Default table and reset to the factory default table

Email from Denis Reily
blow away your default table and reset to the factory default. ./RS_SetFactDef 0
After you issue that, Table 2 should match Table 0, and Table 1 should be undefined. (The Factory Table will load on startup if there is no User table defined.)

C) Current Working table (Table 2)

- Becomes the new User Default table when saved, or lost upon reboot/power- down

- A working (Active) table, which is the table used for selecting reference inputs

Email from Denis Reilly
You can play with the Working table as much as you want, and then save it to the User table when you get it right to persist across reboots.

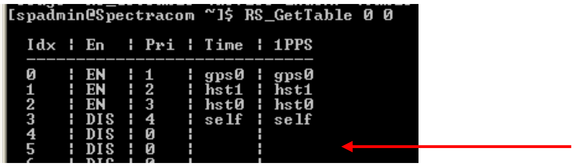
(Modified) Email from Tim Tetreault
The Working/active table is what the board always uses. On power-up, if you have previously entered (Saved) a user table, the TSync board will load that into the working table and use it.   If there is no user table on power-up, it will load the factory table into the working table.

The User (Saved) table is retained during reboots and power cycles. You just need to save the working reference table to the saved user table (**RS_SaveUserDef).**

Number of entries that can be added to the Reference Priority table:

There can be up to 14 combinations of references in the table (there are 14 indexes in the table

Trying to “Enable” indexes in tables that do not have references configured



```
[spadmin@Spectracom ~]$ RS_GetTable 0 0

  Idx | En   | Pri | Time | 1PPS
  -----------------------------
  0   | EN   | 1   | gps0 | gps0
  1   | EN   | 2   | hst1 | hst1
  2   | EN   | 3   | hst0 | hst0
  3   | DIS  | 4   | self | self
  4   | DIS  | 0   |      |
  5   | DIS  | 0   |      |
```

- In summary,Table indexes that are empty can’t be enabled.

Q.,If we try to ¿consult? again the states with ./RS_GetEnable, we get:

```
 RS Table Entry 0 Enabled
 RS Table Entry 1 Enabled
 RS Table Entry 2 Enabled
 RS Table Entry 3 Enabled
 RS Table Entry 4 Enabled
 RS Table Entry 5 Disabled
 RS Table Entry 6 Disabled
```

You can see that number 5 and 6 didn’t change as we tried.

Q. Can you help us for avoid the error we get when configure card with **RS_SetEnable 0 5 1 command.**
Error: invalid param (HA_RC_INVALID_PARAM).

B. How many entries are in their reference table (RS_GetTable 0 2)? If they only have 5 entries, they will only be able to configure from index 0 to 4.
If the index is empty, it has to remain disabled and cannot be changed to enabled without adding references to it.

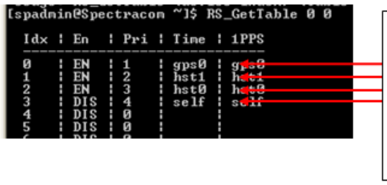
List of “RS” calls/Example programs

“**TSync_RS**” (Reference Service) are the API calls associated with Input Reference. Refer to the TSync-PCIe driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121

Note: Where **<enable>**: 0 for disabled, or 1 for Enabled.

API call/example program	Function
RS_GetTable	Get specified reference priority table.
RS_GetBestRef	Get current best working priority table entry.
RS_GetEntry	Get working priority table entry by index.
RS_addEntry	Add an entry to the working priority table.
RS_SetFactDef	Reset the User priority table to the Default factory table.
RS_SetUserDef	Reset the Working table to the User default settings, if saved user priority table exists.
RS_SaveUserDef	Save the Working priority table to the user priority table
RS_deleteEntry	Delete a working priority table entry by index.
RS_GetPriority	Get specified working priority table entry’s priority.
RS_SetPriority	Set specified working priority table entry’s priority.
RS_GetEnable	Get specified working priority table entry’s enable state.
RS_SetEnable	Set specified working priority table entry’s enable state.
RS_GetStateTable	Get the reference validity state table.

RS_GetTable: Get specified reference priority table.

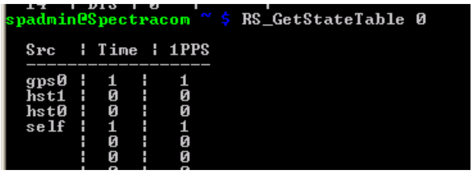


```
[spadmin@Spectracom ~]$ RS_GetTable 0 0

  Idx | En   | Pri | Time | 1PPS
  ----------------------------------
  0   | EN   | 1   | gps0 | gps0
  1   | EN   | 2   | hst1 | hst1
  2   | EN   | 3   | hst0 | hst0
  3   | DIS  | 4   | self | self
  4   | DIS  | 0   |      |
  5   | DIS  | 0   |      |
```

Each Horizontal row of the table consists of a paired Time and PPS reference.

In order for a lower priority reference to become selected, both the Time AND PPS references in that row need to be valid (and the EN” field of that row needs to “EN” for Enabled). If one is valid and the other is not, it can’t be selected.

RS_GetStateTable*:* Get the reference validity state table



```
spadmin@Spectracom ~ $ RS_GetStateTable 0

  Src   | Time | 1PPS
  -------------------
  gps0 |  1   |   1
  hst1 |  0   |   0
  hst0 |  0   |   0
  self |  1   |   1
       |  0   |   0
       |  0   |   0
```

Use the **RS_SetEnable 0 5 1** API call to enable the Self/Self input reference

RS_GetBestRef: Get the current best working priority table entry.

User Table configuration Calls:

- **RS_SaveUserDef:** resets the working reference table to the User (saved) table
- **RS_SetUserDef 0:** resets the working reference table to the User (saved) table.

Desire to reset Reference Priority table back to default settings (for sanitization):

- Use the RS_SetFactDef API call/ example program

The number of entries that can be added to the Reference Priority table depends on how many “Indexes” (rows) are in the Reference Priority table. This is a user-configurable value. Issue the example program command: **RS_GetTable 0 2** to determine how many entries are in the table.

The first row starts with “Index 0”, so if there are only 5 entries in the table, you can only configure index 0 to index 4 (which is a total of 5 entries). If you need to add additional entries to this table, use the **RS_SetTable 0 2** command to add additional indexes, as desired.

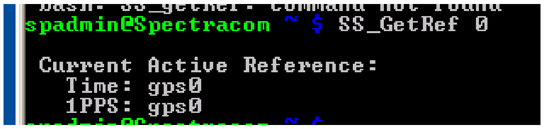
Reference Monitor/ Reference Status table (RS_ calls)

- API calls associated with Refernce Status table
- Refer to Tim Tetreault’s “cheat sheet” at: ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\Tsync driver calls cheat sheet
- These are examples - additional calls may have since been added

REF MONITOR commands
RS_AddEntry
RS_DeleteEntry
RS_GetBestRef
RS_GetEnable
RS_GetEntry
RS_GetPriority
RS_GetStateTable
RS_GetTable
RS_SaveUserDef
RS_SetEnable
RS_SetFactDef
RS_SetPriority
RS_SetUserDef
RS_AddEntry

Obtain the currently selected Time and PPS references (SS_GetRef 0)

SS_GetRef 0 Gets the currently selected time and 1PPS reference.



```
bash: SS_getRef: command not found
spadmin@Spectracom ~ $ SS_GetRef 0

 Current Active Reference:
   Time: gps0
   1PPS: gps0
spadmin@Spectracom ~ $
```

Switching from input reference to another

TSync-PCIe can be "synced" to more than one reference at a time, with only the highest priority input selected at any given moment. Theoretically, if the highest priority input goes away or becomes not valid, SecureSync will seamlessly transition to the next highest priority without going into Holdover alarm (an entry is created indicating SecureSync switched references). However, Engineering has seen it glitch when switching from one input to another, where the Holdover alarm is generated for one second, and then it's back in sync by the next second. You can let the customer know that the switchover from one reference to another will take no more than one second, as long as more than one valid input is present.

Host reference and Self reference

Self/EPP0 Reference versus hst0/EPP0 Refernce

**Host Reference: TSync-PCIe board without external inputs

The “Self” and “hst0” modes are very similar. Both require a command to set the time/date. The biggest difference is hst0 mode provides more control on declaring the time valid:

- **With Self mode**, once time is valid, it will always be valid while it remains powered-up (unless superseded by a higher priority reference).
- **Host mode** allows the time to be commanded to be valid.

A) Host Reference

- With Hst0/EPP0, board will not go into sync, until 1PPS is valid AND the **HR_SetTime <device index> <index> <year> <doy> <hour> <minute> <second> <nanoseconds>** call has been performed.
- Once the HR_settime call above has been performed (with a valid 1PPS input) the board will go into Sync
- TSync remains in full sync (not Holdover), as long as 1PPS remains valid (causing board to go into Holdover).
- If Time sync is lost (either holdover expires before 1PPS is restored, or board is rebooted, the board **will not** go back in goes right back into Sync again, until the HR_SetTime call mentioned above is performed again/

Allows the time to be set and then used as valid time. Note that Host time input is not considered valid until a user commands is to be valid via an API call). Host mode is for time input only. A separate 1PPS input has to be used (such as hst0/epp0). See Section A below.

Note: Typically, a script needs to be written to read the Host time, and then an API call or example program is used to set this in the board. In firmware versions above 1.x.x, the same call/example now allows the time to be set and the time declared valid, with one call, instead of two separate calls.

Refer to: “**A) Host mode”** further below

C) Self reference:

- Self/EPP0 will sync the TSync board as long as the 1PPS input in valid
- No commands need to be entered to go into sync
- TSync remains in full sync (not Holdover), as long as 1PPS remains valid (causing board to go into Holdover).
- If Time sync is lost (either holdover mode expires before 1PPS is restored, or the board is rebooted, the goes the TSync board go right back into Sync (with no commands/ user interaction needed) as soon as 1PPS input is valid again

Having a user manually set the time and 1PPS. Self Time/1PPS is automatically considered valid (No user interaction is required to declare the time is valid). See Section B below.

Note: The “Self” and “hst0” modes are very similar. Both require a command to set the time/date. The biggest difference is hst0 mode provides more control on declaring the time valid. With Self mode, once time is valid, it will always be valid while it remains powered-up (unless superseded by a higher priority reference). Host mode allows the time to be commanded to be invalid.

Refer to: “**B) Self/Self mode**” further below

API calls associated with Host Reference calls

- Refer to Tim Tetreault’s “cheat sheet” at: ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\Tsync driver calls cheat sheet
- These are examples - additional calls may have since been added

Host Reference Commands
HR_GetLocal
HR_GetNumInst
HR_GetRefId
HR_GetTimeScale
HR_GetValidity
HR_SetLocal
HR_SetTime
HR_SetTimeScale
HR_SetValidity

Commented [KW2]:

A) Host Reference mode (use host PC to be the external time reference for the board)

- Refer to Section 5.4 (**Using the Host PC as an External Time Reference for the TSync Board)** of the TSync-PCIe driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121

Note: While the time of the Tsync board is manually set (no other higher priority references available), there is no external 1PPS reference available to use for monitoring the oscillator frequency. Therefore, the Frequency Error alarm will remain asserted until an external 1PPS input reference becomes available.

From the user manual

The TSync-PCIe can be set to use the host as the source of date and time information, while another input reference is required to serve as the source of frequency input. This allows the host to provide time to the board while providing a means to determine and indicate whether that time is valid for synchronization. Using the host as a reference means it could conceivably be used to receive date and time information from a source not available to the board, providing that information to the board for synchronization to it (while using a separate frequency input).

1/11/11 KW- Note that "Host" in TSync-PCIe has been changed to "hst0"

Due to "recent" change to KTS in SecureSync to have more than one host in SecureSync (one "host" is for user-set time and the other is for NTP input). In TSync-PCIe, as of today anyways, there is only one host, so the only reference that will be used for "host" is now "Hst0" (instead of "Host". "Hst1" is only used in SecureSync).

Email from Dave Sohn about this: The host reference is no longer a singular reference. We can have more than one, so the reference names for host are now similar to other. It is "hstx" where x is the host instance number starting from 0. So, the TSync includes a "hst0" reference now instead of "host". We will have to update any documentation that might still refer to "host" as a reference name.

Response from Adam Wright: Tim (Tetreault) just helped me get this updated – the changes were made in both the user manual Rev E., and the driver guide Rev. G (ECN 2070).

Firmware change pertaining to host reference:

- In firmware versions prior to version 2.0.0, using host was a two-step process. The first step was to set the time to the host time. The second was to then declare this time valid.
- In firmware versions 2.0.0 and above, one call both sets the time and as long as Host is the selected input, the time is declared valid at that particular moment only.

The TSync-PCIe boards do not retain the Year, date or time information. So at each start-up, it must be able to obtain it from a reference, or be manually configured with this information. In earlier versions of the TSync-PCIe firmware, you needed to set the time and then declare the host reference time valid with a separate API call, each time the board starts up.

In newer versions of the firmware, the setting of the time to the host time (**HR_SetTime**) automatically sets the date/time and declares it a valid input for that one particular second, assuming the host reference is the selected input reference (as long as no other higher priority input references in the Input Reference Priority –such as GPS, IRIG, etc are available). If there are any other higher priority inputs when the time is set, the host time is not considered valid.

To manually set the time of the TSync-PCIe board to the host system using host mode:

- With the Host mod (hst0), the time of the TSync board is manually set using the:

  HR_SetTime API call (The **CS_SetTime** call is used with the Self reference = not with host).

Syntax: **HR_SetTime (device index – typically 0) (index – 0) (year) (DOY) (Hour) (minute) (second) (nanoseconds)**

Note: In firmware version **above 1.x.x,** this command also declares the time valid.

Reading the PC's time to use in the HR_SetTime command

- In between the HR_SetTime commands, the TSync board is in free-run mode.
- In SecureSync, we run the HR_SetTime command once-per-minute when syncing to NTP (stratum 2 operation). This is a good interval to use for typical NTP accuracies, even with just a TCXO oscillator installed. However, they can perform it more often than once a minute, if they wish

Email from Dave Sohn (9 Mar 15) They can do the same thing with the TSync that we do inside the SecureSync for Stratum 2 operation. If the host reference is set as the time and 1PPS reference, it will set the time as well as adjust the 1PPS point as well. In order to keep synchronization, they will need to periodically set the host reference time using HR_SetTime.

- Most accurate method to auto-read and periodically set the time
- This method is meant for Linux- not Windows
- Refer to the draft email Keith has for the actual script attachment.

Email Keith sent based on info from Tim Tetreault (9 March 15) Regarding the desire to sync the Spectracom TSync-PCIe timing board to your NTP-synced system, I have some additional information for you that should help...

Below you should find a simple script that can be used to periodically set the TSync's board to your system's time:

```
#!/bin/sh

# set HW_SetTime with UTC system time
./HR_SetTime 0 0 `date +%Y.%j.%H.%M.%S.%N -u| tr "\." " "`

exit
```

Note that we also use this script in the code of one of our other products which is designed around the TSync-PCIe architecture, for a similar function as you are using. We run this code once-per-minute to keep the system time aligned with the host's time. We have found this interval to be sufficient enough to keep the time well within the capabilities of NTP. However, you can run this script more often than once-per-minute to set the time in the TSync board more often, if you wish.

CS_SetTime call/example program (Less accurate method than HR_SetTime, to read and set the time – per an earlier email)

Host time allows the time of the TSync board to be set to a user-specified time and with the same call or example program, the time is also declared valid. The TSync-PCIe board and associated driver don't have the ability to read the computer's time/date automatically (The computer's time can be read automatically, but this has to be done outside of the TSync-PCIe's driver). Then, once the time/date has been read, a call can be used to simultaneously set the time and declare the time valid. The reading of the computer's time and the setting of the time/date in the TSync-PCIe board can be performed with a customer-written Windows script.

A good starting point to create a Visual Basic script that can read the Windows PCs date/time is http://technet.microsoft.com/en-us/library/ee198917/

(**Note**: Applicable to **1.x.x** firmware versions only) With Host mode, you also have to command validity of the Host time before it can use the Host time as a reference. **TSYNC_HR_SetValidity 0 0 1 1** <enter> (Set the reference validity of the Host)

Delay after setting the time. before the "old" time is replaced with the "new" time

A) from Salesforce case 183640

Email from Dave L (23 Jan 19) I have confirmed what you are seeing is the normal operation of the Tsync.
The delay is due to processing that ensures all timing is properly aligned when the time is set. This takes approximately 2 seconds to

apply. There is nothing we can do to change this operation.

The TSync compensates for this alignment by switching to 2 seconds instead of 0.

HR_SetTime should be able to handle nanoseconds, however if you want the time to snap you can do a CS_SubSecAdj which will snap the sub second time to 0 .

B) from earlier emails/cases

Q. I noticed that after I set the time with: **TSYNC_CS_SetTimeSec(index, seconds, ns**) I continue to read old time values for approximately 1250ms. Is this normal? This is causing me problems. Other than sleeping for 2 seconds I don't see a good workaround. What do you recommend?

A Below is the response from Engineering related to your recent question:

> "The delay the customer is seeing when running a CS_SetTime makes sense. The time is only set on the "On time point" which is once a second. There is also some overhead required on the board so I can see it taking 1.5 seconds for the new time to be seen."

B) Self/Self Reference (Desire to sync the TSync board to itself)

The TSync-PCIe board can accept Self/Self mode for synchronization with no external references. However, after each power-up, the user has to set the time, to declare the time is valid.

LED indicators when using Self reference for sync
The Self/Self reference has a unique LED pattern where the green (sync) and yellow (Holdover) LEDs will both blink simultaneously) when using the Self reference for sync. In at least the Rev E and below versions of the TSync-PCIe manual, this pattern is indicted in the "LED Flash patterns" table as "Free Run".

While in this flash pattern, Sync will still indicate true and Holdover will indicate false. This is to indicate the TSync-PCIe is being treated as if it's in continuous sync, as well as indicating the oscillator is in free-run mode (oscillator is not being disciplined).

Info from the TSync-PCIe user manual:

The TSync-PCIe provides a built-in reference that allows the board to operate without a separate input reference. The date and time or frequency information from this self reference is always considered valid. This allows a user to operate the board as if it were synchronizing to an input reference, without a valid external reference input. The self reference priority table entry defaults to "disabled. (By factory default, the input reference table contains a Self/Self entry. However, this entry is disabled by default)

The steps to manually sync the board are below

Note: These steps have to be performed each time the TSync-PCIe board boots-up

Note: With the Self mode (self), the CS_SetTime call is used with the Self reference. ("HR_SetTime" is only used with the host mode).

Regarding the need to manually sync the TSync-PCIe board to itself, before it can sync a PTP slave, I have the following information for you:

Syncing the TSync-PCIe-PTP that is configured as the PTP master to itself consists of two steps. The first step consists of enabling a default disabled row of the Input Reference Priority table. This configuration change only needs to be performed once, as changes t this table persists through each reboot. The second step is after each time the board powers-up, the time/date of the board needs to be set/declared valid.

FYI- there are three types of Input Reference Priority tables. One of these is the Default table, (defined as type 0). This table contains the factory default settings. The second is the User table (defined as type 1). This table is created when the default table is changed and those changes are saved. The third table is the Current working table (defined as type 2). This table is created when changes have been made to either of the first two tables, but the changes have not yet been saved. This table does not persist through a reboot of the TSync-PCIe board (The changes have to be saved in order for it to persist). The changes are saved as the User table.

By factory default, the Default Input Reference Priority table contains an entry that lists "self" as both the "Time" and "PPS" input references. It is row 5 of the default priority table, But, by factory default, this row (referred to as an "index") is set to disabled. It needs to be changed to enabled, before the board can sync to itself.

If any changes have already been made/saved to the default Input Reference Priority table, the default priority table becomes the User table. If no changes have yet been made to the default table, the default table is still used to define the input references. So, if there have been no changes made to the table yet, you want to edit the Default table. Otherwise, if changes have already been made to the default table, you want to edit the User table.

RS_GetTable 0 0 <enter> API call/example program will show the current settings of the Default Reference table, while **RS_GetTable 0 1** will show the current settings of the User (saved) Reference Table. To enable the Self/Self entry, edit the Default table if no changes have been made/saved to this table. Or, edit the User table, if any changes have already been made to this table.

The steps to manually sync the board are below

Note: These steps have to be performed each time the TSync-PCI104 board boots-up

A) To enable the Self/Self Input reference (This reference is disabled by factory default)

1. Type: **RS_SetEnable 0 5 1** to enable the Self/Self input reference in the working table
2. Type: R**S_SaveUserDef 0** to save the working table as the User table

B) To manually set the time of the TSync-PCI104 board using Self mode (this step needs to be performed each time the board is powered-up/rebooted

Use the **CS_SetTime** API call to set the time/date as desired

Note: Entered as: **CS_SetTime 0 0 <year> <DOY> <Hour> <Minutes> <Seconds>**

Example: CS_Settime 0 0 2012 123 12 12 12 <enter> (year is 2012, 123rd day of the year, hours 12, minutes 12 seconds 12)

To enable the Self/Self row of the User table, use the API call/ Example program entry by typing: **RS_SetEnable 0 5 1** <enter>. Then, to save the working table as the User table, use the example program **RS_SaveUserDef 0** <enter>. Now, when the board is rebooted, the entry will still be enabled.

Each time the TSync-PCIe board powers back up, the time/date needs to be set, to validate the time. Use the CS_SetTime API call/example program to set the date and time. The format of this call is as follows:

CS_SetTime 0 0 <YEAR> <Day> <Hour> <Minute> <Second> (where "Day" represents the "day of the year")

Example: Type**: CS_Settime 0 0 2012 123 12 13 14** <enter> (sets the TSync-PCIe to year of 2012, the 123rd day of the year, and the time as 12:13:14).

After entering this line, using the desired values, the TSync-PCIe board will go into holdover mode (Sync LED flashing green). Holdover mode is treated like the board is in sync.

**Desire to sync TSync board via NTP (indirectly)

The TSync-PCIe boards can't sync via NTP directly. However, they can sync indirectly to NTP, if desired, when installed in a Linux box running NTP software (Refer to "Hailer IT/Jeffries in SalesForce). This was suggested by Dave Sohn, for this customer of Jeremy Onyan's.

To sync the TSync board via NTP, use the NTP software to sync the linux kernel to the desired NTP server(s). Then, sync the TSync board to the system using the host/host input reference. The TSync board can then sync to the linux kernel time. NTP maintains the kernel time and host mode maintains the TSync's time.

Also refer to the previous section on using the Host mode to sync to the kernel time: Desire to sync TSync-PCIe board without external inputs (host/EPP and Self/Self).

**Desire to sync TSync board via an Endrun Praecis CDMA receiver (or other external references via ASCII data input)

Email from Dave Sohn (18 Dec 2012) There is currently no ASCII input on TSync boards, so there are no plans to support any ASCII formats on the TSync. This could potentially be done with the SecureSync, or a SecureSync synchronizing to the EndRun and then providing IRIG timing to the TSync.

If we wanted to consider this as a special, then we could look at it that way.

**Time Sync and Holdover status/ green "sync" LED indication

Known potential issue associated with the TSync PCB: TSync board is "in sync", even though there are no external inputs/Self mode is not in use (Reference NC-002529)

- Refer to **ECO-FAI-1857** for info on a PCB board spin for this condtion (in Arena): https://app.bom.com/changes/detail-summary?change_id=2394867944&

Summary/Description (excerpted from ECO-FAI-001857)

TSync-PCIe board spin to implement corrective actions for NC-002529 (stop shipment due to Tsync PCB's showing sync with no connections made) and AM IRIG input circuit error discovered on VersaSync AM IRIG option card.

NC-002529 corrective action - Add 10k pull downs resistors to J8 floating inputs at U55 pins 1, 3, 5, 9, 11.

AM IRIG input circuit error - Add 11k resistor in series with U29 pin 1, 2, 14.

As soon as the TSync-PCIe board goes into sync with its input references, the sync status in all **HW_Gettime** calls as well as the current sync status reported with the **SS_GetSync** call update immediately. However, the green "sync" LED (and the other two LEDs on the edge of the board) only update once per second. So, the calls could read synced just before the "sync" LED turns green.

There are a few different way to obtain sync/Holdover status of the TSync board

A) Determining current sync status via the LEDs on the edge of the board

- refer to: Status LED's/Status LEDs upon reboot for more info

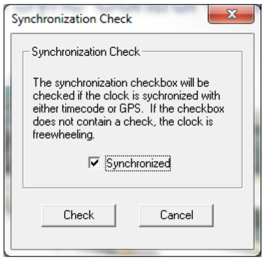
Observe the three LEDs on the edge of the Tsync board

The green led will illuminate when it's actively receiving and synced to IRIG (or GPS) (the other two LEDS will not be lit). Both the Green and Yellow LEDs will illuminate if the board loses IRIG input (GPS) after it's in sync, and until the Holdover period expires (the two LEDs being lit indicates it's in Holdover mode).

B) Using the Windows Control Utility to verify sync status

To open the Windows control utility: Start / All Programs / Spectracom Corp / TSync PCI / TSync Control utility

The sync status is reported by selecting Sync -> Status. The checkbox will be selected if the board is in sync (as shown below):



Synchronization Check

Synchronization Check

The synchronization checkbox will be checked if the clock is sychronized with either timecode or GPS. If the checkbox does not contain a check, the clock is freewheeling.

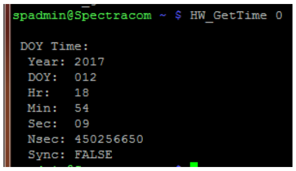
☑ Synchronized

C) Sync status associated API calls/example programs

Keep in mind that when using input references such as PTP or IRIG for synchronization, these inputs often have some jitter on the PPS its providing as a reference. The ontime point (1PPS output) of the TSync board will not jump to keep synced with this jitter. Instead, it will be very slowly slewed via disciplining to align with the reference. This will help dampen out any jitter of the PPs input reference (the TSync board will only hardware coarse adjust to the input PPS the first time after power-up that a PPS reference is selected and it syncs to the reference. Thereafter, it's always slewed to the input reference.

Using the HW_GetTime 0 API call/example program to determine current sync status

- The end of the **HW_GetTime 0** call reports the board's current Sync status (note that CS_GetTime calls do not provide sync status)



```
spadmin@Spectracom ~ $ HW_GetTime 0

 DOY Time:
  Year: 2017
  DOY:  012
  Hr:   18
  Min:  54
  Sec:  09
  Nsec: 450256650
  Sync: FALSE
```

Using the SS_GetSync and SS_GetHoldver API calls/example programs to determine current sync status

- The current sync status is also available via a couple of API calls/example programs.
- The current **Sync** status can be always be queried using the **SS_GetSync** call

The **SS_GetSync** API call indicates True when TSync-PCIe is either synced or in the Holdover mode. It reports false when not currently in sync or in Holdover mode.

- The current **Holdover** status can be always be queried using the **SS_GetHoldover** call (Responds with either "TRUE" or "False"

To differentiate between fully Synced mode and Holdover mode, the **SS_GetHoldover** API responds with True if in Holdover mode only. It will return a False if the board is in not in Sync or not in sync.

The command to read sync status is **SS_GetSync 0** API call (the "0" assumes there is only one TSync-PCIe board installed). The response will indicate "true" when in sync or while in Holdover mode (all references have been lost). To differentiate between in full sync and Holdover mode, also use the **SS_GetHoldover 0** API command. If Sync is true but Holdover is false, it's in full sync. But if Sync is true and Holdover is true, the board is currently in Holdover mode (no references currently available).

While in Holdover mode, the green LED on the edge of the board will remain lit and the yellow LED will also be lit. If both commands report False, the board is out of sync (not in sync and not in Holdover. Either it powered up and hasn't synced yet, or it initially went into Holdover mode and Holdover mode has since expired, because no references were restored before it timed-out).

D) Verify the TFOM and Phase Error values are low

When the TSync board is closely aligned with its 1PPS input reference, the TFOM and Phase Errors will be lower numbers (these numbers are oscillator-type dependent), Typically TFOM will be 4 or less and the Phase error will be like 200ns or less when in sync with the input PPS reference. The command to read the current TFOM value is **SS_GetTFOM 0 and the command to read the current phase error is XO_GetPhaseError 0.**

E) SS_GetTfom

Q. How does the TSync-PCIe board calculate the TFOM Value?

A. **reply from Keith (26 Jun 14)** The TSync board uses the calculated phase error value as the input for the TFOM Calculation. This data is collected in a variable size window and the result compared to a table of set values to select the TFOM value.

The variable min and max measurement window gets complex. It is set depending on the accuracy of the reference source, the stability of the data and the type of oscillator used.
The TFOM is an estimate of the timing error of the board and meant to be used as an indicator of the general performance of the timing;

The Time Figure of Merit (TFOM) enumeration is defined to provide a dimensionless quality measure of the overall time/1PPS synchronization based on an Estimated Time Error (ETE). This measure is used for overall sync status of our clock to the internal 1PPS. Our synchronization definition is derived from our use of time and 1PPS to achieve both synchronization of time and synchronization of frequency.

F) Current sync status determined by time stamps received from the TSync board

- The timestamp captured when the sync status last changed states (either into or out of sync) can be queried with the **SS_GetTimestamp** call**.**

Compare the 1PPS output of the breakout cable to the PPS input reference.
Another way to confirm the TSync board is in sync is to use an oscilloscope to compare the 1PPS output pin of the Timing connector to the 1PPS output/input reference. The closer the two 1PPS outputs are aligned with the other, the more closely aligned the TSync board is to its selected PPS reference.

Oscillator lock status API call/example program

TSYNC_SS_GetFreeRun: Gets the current freerun state.

**“Time” Input synchronization (PTP Slave, GPS input, IRIG input, Self, Hst 0)

- **Default power-up year/date/time**: **Jan 1, 2000 00:00:00 UTC**

FYI- the TSync-PCIe board does not have a Real Time Clock (RTC) installed. Each time the TSync-PCIe board powers-up, it starts up as Jan 1st 2000 00:00:00 UTC and then its count-up each second from there (this is the same date/time its outputting via IRIG). The start-up date/time can be automatically corrected via GPS or IRIG input (GPS also corrects the year, while IRIG input can correct the year in some cases). The date/time can also be manually corrected, if no external inputs are being applied to sync it.

Input Reference designations for TSync

Reference Designation	Input	Refer to hyperlink Section further below
gps0	GPS/Glonass	GPS input
ira0	IRIG **AM** reference	IRIG input
ird0	IRIG **DCLS** reference	IRIG input
epp0	External 1PPS reference	EPP0 input
hst0 (was host)	Host System Reference	Host mode
ptp0	PTP	PTP input
Self	Internal reference	Self/Self)

When the TSync board first boots-up and receives its external input (such as GPS, IRIG or PTP), it will jump to the correct time and the sub-second counter will be zeroized to the correct microsecond. If the IRIG or other input is then removed, the oscillator disciplining controls the time. When the input is restored, the seconds will re-align and the oscillator disciplining will steer the sub-second counter. This is different from the older PCI cards which always jump to the correct time when the input is restored.

“The reason for the offset is that the PCIe does not automatically latch onto the 1PPS on time point in the way the TPRO-PCI does. Instead, the PCIe uses oscillator disciplining to gradually sync the IRIG signal to the on time point. The slew rate (rate of adjustment) in the oscillator disciplining algorithm is about **4 us/sec**. This means that it would take some time for the disciplining to eliminate an offset as large as the customer has seen (100 – 250 ms). However, the disciplining will eventually correct the offset. The customer just has to allow enough time for this to happen, as a 4 us/sec adjustment will not be seen on the scope using a millisecond scale. “

Note: Course adjustment only occurs after a power-up. After course we adjust, it switches to fine adjust. This process takes about three minutes. However, if the board goes into holdover mode and then the reference(s) return, course adjust does not occur. If the oscillator drifted over the time it was in holdover, it is fine-tuned (slewed) to the input 1PPS over a very long time-frame. If it had drifted quite a bit, it may take several minutes to several hours to re-aligning.

Oscillator Slew Rate

After initial oscillator correction after power-up, the oscillator is then slewed to its input. The slew rate is very slow (~100ns per second).

Based on a slew rate of 4 us/sec, this equates to:

- **Per hour:** 14,400 us/hour (14 ms/hour)
- **Per day**: 345,600 us/day (345 ms/day)

Timing board synchronization via IRIG playback from a tape

Q When we play the tape from the Digital Recorder and feed the recorded IRIG-B into our Tsync board[1], the Tsync board says we are in Holdover mode[2] . But several other devices[3] seem happy and lock with the IRIG-B signal from the tape[4].

When we connect the Tsync board directly to our GPS Receiver, it is fine and locks (Sync = True, Holdover = False) – this is our normal operating mode.

A **Per Dave Lorah 2 Feb 16** I know what is happening here. The Timing boards have an IRIG Input accuracy spec that is too tight for tape playback. The IRIG Signal from tape playback is not stable enough for the oscillator to lock onto so the board will never declare sync.

This I why the board will sync to the more accurate GPS Receiver IRIG signal and not to the jittery tape playback signal.

This is a factor of the precision oscillator on the TSync board. Unfortunately there is no way to resolve the problem due to the instability of the tape.

NMEA input (NMEA 0183) for synchronization

- Refer to Salesforce case 262258 (info below is excerpted from this case)
- Discontinued **TSync-PCIe-002** (External GPS receiver) and **TSync-cPCI** boards had NMEA 0183 input capabilities
- TSync-PCIe boards with no receiver and TSync-PCIE boards with internal (onboard) GNSS receiver do not have NMEA input capabilities.
- Alrternate approach is instead of using TSync board, use a SecureSync with 1204-02 (RS-232) or 1204-04 (RS-485) ASCII in/out Option Card to input NMEA 0183 data strings.

Q I think I've asked this question before, but it's an ongoing pain point for us: Is there any possible way to set the TSync clock using NMEA messages from our existing GPS receiver? If it were possible, it would remove an annoying wrinkle in our pipeline.

A My response
Note for the info below that, a couple of variants of TSync boards (Such as the TSync-PCIe board with external PS receiver, and TSync-cPCI boards) have either already been discontinued or nearing end of life. Matt Mang can be provide you with a better exact status on availability of these TSync variants.

Up until just recent, we offered two different variants of TSync-PCIe timing boards with GPS receiver capabilities. One variant, the PCIe-002 timing boards, which interfaced with a GPS receiver housed inside its GPS antenna) was discontinued/last time buy around January 2021, I believe. This unique configuration used an NMEA 0183 input from its antenna for synchronization.

The other TSync variant that could accept NMEA input was the TSync-cPCI timing boards. I believe these may still be available, but are reaching end-of-life, if they are still available.

The other "readily available (not discontinued, oreven nearing end-of-life) product that can accept NMEA input directly for its synchronization is our SecureSyncs,with a Model 1204-02 or 1204-04 Option Card installed. The SecureSync is essentially a "TSync in a box". It gives the same type functionality as a Tsync board, plus much more.

Matt can give you more info on this product, but it's like having a TSync board in a 1U 19 inch enclosure (instead of plugging into a PCIe slot. It has Option Slots on its rear panel to provide configurable inputs/outputs (such as NMEA input, for instance). Having a GNSS receiver installed,or not installed, is optional. Our Model 1204-02 and 1204-04

Option Cards are ASCII input/output (either RS-232 or RS-485) that plug into the Option Slots to allow a SecureSync to sync to NMEA 0183 inputs. It can then provide timing outputs similar to a TSync board.

In summary, our readily available TSync-PCIe boards with either no GNSS receiver capability, or having the GNSS receiver attached to the TSync board, do not have NMEA input capabilities. The now discontinued TSync-PCIe-002 boards with a connector to attach to an Acutime GPS antenna having a GPS receiver inside the antenna housing received NMEA 0183 messages from the antenna for its sync. So these boards could sync via NMEA 0183 messages.

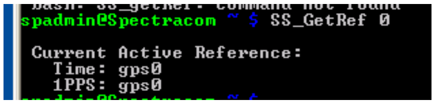
The alternate approach is to use a SecureSync having an ASCII input Option Card installed, so that it can interface with/receive NMEA 0183 messages for its sync (it can also use GPS input for its primary sync, or as an automatic backup that syncs this device if the NMEA input happened to be lost).

**Selected reference/ Reference change notification

Selected Reference

- Obtain the currently selected Time and PPS references

SS_GetRef 0 Get the currently selected time and 1PPS reference.



```
bash: SS_getRef: command not found
spadmin@Spectracom ~ $ SS_GetRef 0

 Current Active Reference:
   Time: gps0
   1PPS: gps0
spadmin@Spectracom ~ $
```

Reference Changes

Q. Does the TSync board raise any type of event or interrupt when the timing source changes (next in priority-order)? If not, is there any way to find out if the timing source has changed other than polling the table or a loop calling TSYNC_RS_GetBestRef? Thanks.

- Besides polling for selected reference, there is no other way to alert to reference changes.

A) PTP Slave mode (PTP input)

- Refer to the TSync-PCI driver guide for an extensive discussion on how to set up the PTP boards.
- TSync-PCIe boards are factory configured as PTP Slaves (can be re-configured in the field as PTP Masters).
- The PTP input calls/example programs are the "**PTR**" calls

Note: Refer to: **PTP Precision Timing Protocol (for all products)** in the Customerserviceassistance doc for more info on PTP.

Command to verify PTP input is valid: PTR_GetValidity

PTP module not responding to a ping

- As of at least 6/6/12, the SecureSync PTP Option Card will not respond to ping from a Linux box. It only responds to ping from a Windows PC. This likely applies to the TSync-PCIe's PTP module as well.

**PTP client software exhibiting excessive NTP offset and Jitter

NTP software running on a system (which is syncing to a TSync-PTP Slave) is exhibiting higher than expected jitter and offset values:

Email from Dave Sohn to Peter Sadowski (10/8/12)
Those are strange results of NTP versus board sync. Are there any substantial differences between this server and the rest? Is its processing load different? Is the PTP network routing different? Can you send the results of the below commands? Thanks.

SS_GetRef 0
SS_GetTfom 0
XO_GetState 0

XO_GetDac 0
XO_GetFreqError 0
LS_GetAlarm 0 2
LS_GetAlarm 0 5

****Some PTP calls working while others aren't**

- Some, but not all, PTP calls will fail when using a TSync driver with a version beyond 2.5.1.
- Driver versions beyond 2.5.1 are not compatible with newer PTP calls that were added in KTS to support SecureSync-PTP, so they will fail when run.

```
  Example call failing
usr/local/bin/PTR_GetPtpItf 0 0
Error: unknown id (HA_RC_UNKNOWN_ID).
```

B) GPS/Glonass input synchronization

Antenna Cabling for TSync-PCIe-002 / TSync-PCIe-012 (TSyncE-PCIe board with external GPS receiver)

TSync-PCIe-002 /TSync-PCIe-012 (TSyncE with external GPS receiver) boards ship with:

Note: TSync-PCIe-002/ TSync-PCIe-012 (TSyncE-PCIe) boards were last time buy/ discontinued ~Jan 2021.

~~**CA08R-0000-0003**~~ (Breakout cable) **Note**: Discontinued

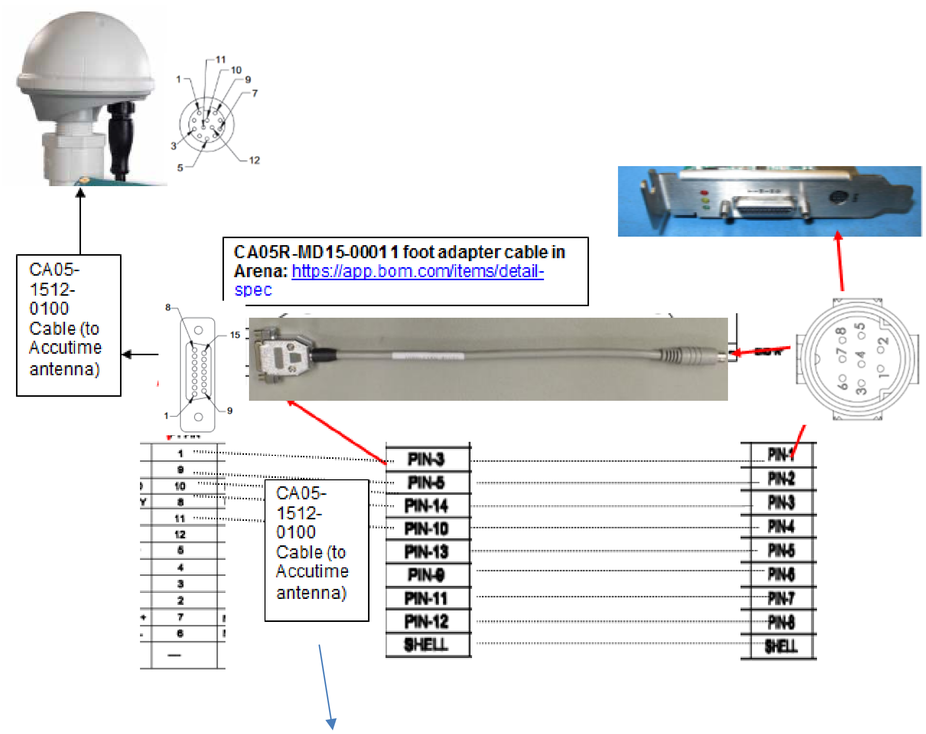
- **E025-0004-0002** (Acutime GPS antenna)
  - refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- **1159-0000-0700** (GPS mast)

  (in Arena) https://app.bom.com/items/detail-spec?item_id=1202841005&version_id=10221291538&orb_msg_Single_Search_p=1&redirect_Seqno=6678014486
- **CA05-1512-0100** (100 foot GPS cable) or **CA05-1512-0200** (200 foot GPS cable)
- **1191-0000-0710** (BNC T connector and 50 ohm terminator for 1PPS input)

  (**in Arena**): https://app.bom.com/supplier-items/detail-spec?item_id=1218663225

  (**In Salesforce**): https://orolia.my.salesforce.com/01t1A000004SUKm?srPos=2&srKp=01t

Accutime GPS antenna (E025-0004-0002)



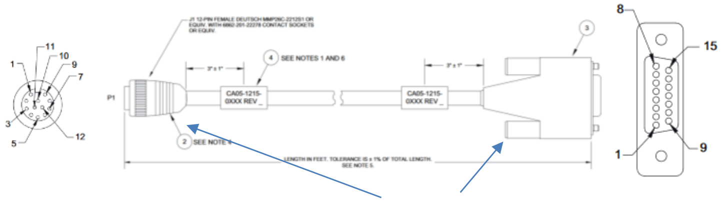
CA05-1512-0100 Cable (to Accutime antenna)

CA05R-MD15-0001 1 foot adapter cable in **Arena:** https://app.bom.com/items/detail-spec

CA05-1512-0100 Cable (to Accutime antenna)

PIN-3	PIN-1
PIN-5	PIN-2
PIN-14	PIN-3
PIN-10	PIN-4
PIN-13	PIN-5
PIN-9	PIN-6
PIN-11	PIN-7
PIN-12	PIN-8
SHELL	SHELL

Antenna cable (CA05-1512-0100 and CA05-1512-0200) pin-out:


11
10
9
7
1
3
5
12
J1 12-PIN FEMALE DEUTSCH MMP26C-2212S1 OR EQUIV. WITH 6862-201-22278 CONTACT SOCKETS OR EQUIV.
4 SEE NOTES 1 AND 6
3" ± 1"
3" ± 1"
3
P1
CA05-1215-0XXX REV _
CA05-1215-0XXX REV _
2 SEE NOTE 4
LENGTH IN FEET. TOLERANCE IS ± 1% OF TOTAL LENGTH. SEE NOTE 5.
8
15
1
9

	SIGNAL	ITEM 2	ITEM 3	COLOR
PAIRED	+12V	1	3	RED
	GND	9	5	BLACK
PAIRED	UNUSED	10	NO CONNECT	BLUE
	BATTERY	8	NO CONNECT	GREEN
PAIRED	IPPS +	11	9	ORG/WHT
	IPPS -	12	14	BLK/WHT
PAIRED	DOWN +	5	10	YELLOW
	DOWN -	4	11	BROWN
PAIRED	UP +	3	12	ORANGE
	UP -	2	13	VIOLET
PAIRED	PORT A +	7	NO CONNECT	GRAY
	PORT A -	6	NO CONNECT	WHITE
FOIL AND DRAIN WIRE	SHIELD	—	SHELL	—

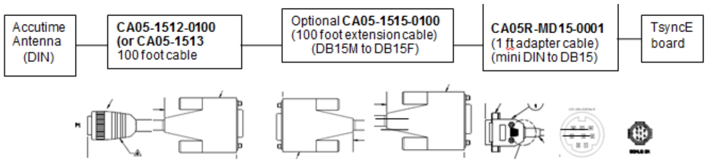
Desire to extend the Acutime antenna cable length (to more than 100 feet)

Two available solutions:

A) Available (optional) 100 foot GPS extension cable: Our P/N CA05-1515-0100

- Can purchase from us/add one (or more) 100 ft extension cables (our P/N: CA05-1515-0100) between the supplied 100 ft cable and the TSync board. As Trimble offers custom length cables up to 1800 feet, this is the minimum length which can be run (suspect even futher is possible). So a customer could add SEVERAL 100 ft extension cables for a long distance cable run.
  - The disadvantage of this method (besides cost) is multiple connection points which could fail/concerns of water, etc

Functional cable drawing for adding one (or more) extension cables


Accutime Antenna (DIN)
CA05-1512-0100 (or CA05-1513 100 foot cable
Optional CA05-1515-0100 (100 foot extension cable) (DB15M to DB15F)
CA05R-MD15-0001 (1 ft adapter cable) (mini DIN to DB15)
TsyncE board

B) Purchase an Acutime antenna cable directly from a Trimble dealer/rep

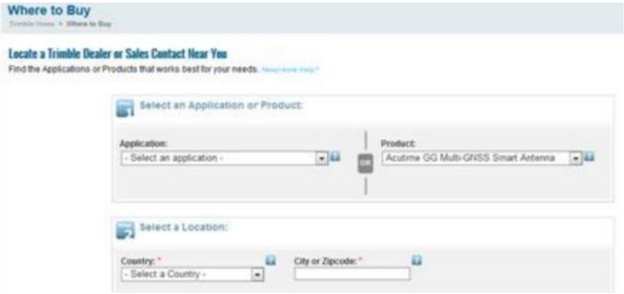
- **Trimble** (where we buy our 100 ft cables from) offers several standard and custom length cables for the Acutime antenna.
- **Per Dave Sohn, we do not want to offer multiple standard or custom length Acutime cables. So customer should purchase the cable from a Trimble delear/reseller**

- Customer can purchase one extension cable to **attach** to the existing 100 ft cable. Or even better:
- Customer can purchase one **replacement** cable which spans the **entire distance** betweeh the antenna/TSync board.

Note: Refer to great, lengthy draft email from Keith (1 Jun 17) for purchasing cable from Trimbe I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\TSYNC-PCIe\GPS\TsyncE (ext rcvr)

Excerpts from the email mentioned above

They can also refer to the following link to find a Trimble dealer/Sales Rep that they can work with directly to purchase the antenna cable from: http://dealerlocator.trimble.com/ (for the "**Product**" drop-down on this page, select "**Acutime GG Multi GNSS Smart Antenna**"):



Where to Buy

Trimble Home > Where to Buy

Locate a Trimble Dealer or Sales Contact Near You

Find the Applications or Products that works best for your needs. Need more help?

Select an Application or Product:

Application: - Select an application -

Product: Acutime GG Multi-GNSS Smart Antenna

Select a Location:

Country: * - Select a Country -

City or Zipcode: *

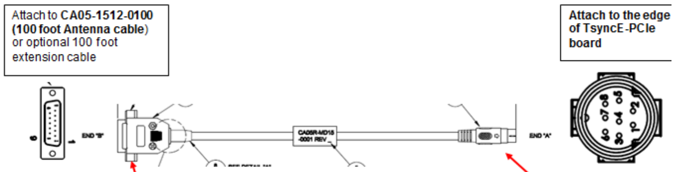
If they do want to purchase the cable directly from Trimble and want to purchase a non-standard cable from Spectracom, Dave Sohn will need to become involved at that point.

CA05R-MD15-0001 (1 foot DB15 to DIN adapter cable)

- This 1 ft adapter cable connects to the DIN connector on the edge of the TSync board and other end connects to the 100 ft cable (**CA05-1512-100).**
- Refer to the next section down for additional info on the 100 ft CA05-1512-100 cable CA05-1512-0100 (100 foot Antenna cable)

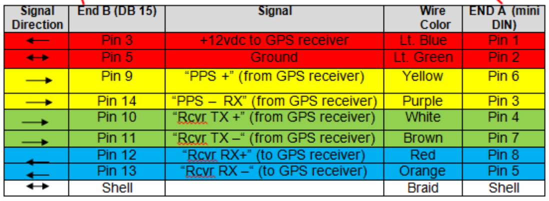
This **CA05R-MD15-0001** adapter cable plugs into the 8 pin MINI-DIN connector on the edge of the TsyncE board. Then, the **CA05-1512-100** cable is connected to the D Sub 15 pin connector on the other end of the supplied adapter cable.

Supplied 1 foot Adapter cable P/N **CA05R-MD15-0001** (refer to partial drawing below or full drawing (CA05R-MD15-0001) in Arena: https://app.bom.com/items/detail-spec is included in the TsyncE ancillary kit (1191-0000-0703)



Attach to **CA05-1512-0100** **(100 foot Antenna cable)** or optional 100 foot extension cable

Attach to the edge of TsyncE-PCIe board



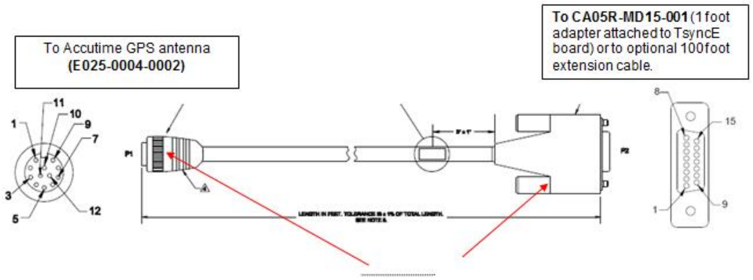
Signal Direction	End B (DB 15)	Signal	Wire Color	END A (mini DIN)
←	Pin 3	+12vdc to GPS receiver	Lt. Blue	Pin 1
↔	Pin 5	Ground	Lt. Green	Pin 2
→	Pin 9	"PPS +" (from GPS receiver)	Yellow	Pin 6
→	Pin 14	"PPS – RX" (from GPS receiver)	Purple	Pin 3
→	Pin 10	"Rcvr TX +" (from GPS receiver)	White	Pin 4
→	Pin 11	"Rcvr TX –" (from GPS receiver)	Brown	Pin 7
←	Pin 12	"Rcvr RX+" (to GPS receiver)	Red	Pin 8
←	Pin 13	"Rcvr RX –" (to GPS receiver)	Orange	Pin 5
↔	Shell		Braid	Shell

CA05-1512-0100 (100 foot Antenna cable and CA05-1512-0200 (200 foot Antenna cable)

- Goes between the
- GPS (Acutime) antenna cable used with TSyncE-PCIe boards (CA05-1512-0100) and (CA05-1512-0200).
- Can be combined with additionall 100 ft lengths of this same cable to extend the distance between the Tsync board and the Acutime antenna.

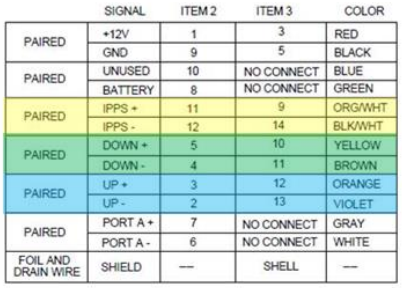
Pin-out (CA05-1512-0XXX)

- in Arena: https://app.bom.com/items/detail-spec



To Accutime GPS antenna **(E025-0004-0002)**

To CA05R-MD15-001 (1 foot adapter attached to TsyncE board) or to optional 100foot extension cable.



	SIGNAL	ITEM 2	ITEM 3	COLOR
PAIRED	+12V	1	3	RED
	GND	9	5	BLACK
PAIRED	UNUSED	10	NO CONNECT	BLUE
	BATTERY	8	NO CONNECT	GREEN
PAIRED	IPPS +	11	9	ORG/WHT
	IPPS -	12	14	BLK/WHT
PAIRED	DOWN +	5	10	YELLOW
	DOWN -	4	11	BROWN
PAIRED	UP +	3	12	ORANGE
	UP -	2	13	VIOLET
PAIRED	PORT A +	7	NO CONNECT	GRAY
	PORT A -	6	NO CONNECT	WHITE
FOIL AND DRAIN WIRE	SHIELD	—	SHELL	—

Note: For pin-out info on this cable, refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

Mini-DIN Schematic (1191-1001-0200) from: I:\Engineering\Schematic\1191-1001-0200

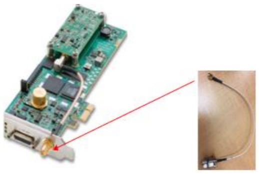
GPS / Glonass / A-GPS

Model 8230 GNSS antenna (8226 surge suppressor/8227 preamp)

- Model 8230 Antenna is used with TSync boards having an onboard GPS/GNSS receiver

- **Refer also to:**
  - the “**Models 8230 and 8230-00**” section of the custassist doc: [I:\Customer Service\1- Cust Assist documents\CustomerServiceAssistance.pdf](I:\Customer Service\1- Cust Assist documents\CustomerServiceAssistance.pdf)
  - Section 4.1 (installation section) of the TSync-PCIe user manual (in the online TSync user manual at: http://manuals.spectracom.com/TS/Content/TS/Topics/AntSysInst.htm)

CA01R-0NSA-800 (Type N to SMA Adapter cable)



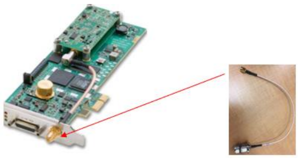
- The antenna cable attaches to the SMA connector on the TSync board using our P/N: **CA01R-0NSA-8001** (CABLE,N JACK,SMA PLUG,RG-316,12in)
- Link to mechanical drawing of this adapter cable (in Arena): https://app.bom.com/bom-auth/detail-download/S2BvVZ/1744604338/CA01R-0NSA-8001ra.zip

Email Keith sent to a TSync customer 8 Jan 18) Regarding GNSS (GPS) antennas for the TSync-PCIe timing boards, also attached you should find a datasheet and manual for the Spectracom Model 8230 GNSS antenna. This antenna should be installed outdoors, with a good view of the sky (360 degree view of the horizon is preferable but not required).

We recommend a Spectracom Model 8226 GPS surge suppressor be installed inline to help protect the TSync board from being damaged by surge. Note we also offer the Model 8227 GPS preamp for RF-400 cable distances greater than 400 feet.

The GNSS antenna attaches to the edge of the TSync board using a 50 ohm coax cable (we offer/recommend using Belden RF-400 or equivalent cable for distances up to about 400 feet without the need to install any inline amplifiers. Cables are pre-terminated on both ends with Type N adapters).   We also offer a plenum-rated variant of this cable, for use in overhead ceilings (it has better fire retardancy to prevent toxic fumes).   Sadie Nedo can provide you with additional info on the Model 8230 antenna, associated antenna cabling, the Model 8226 surge suppressor and the Model 8227 preamp, as desired.

One end of the building antenna cable attaches directly to the type N connector on the bottom of the Model 8230 GNSS antenna. The other end of the cable is attached to the edge of the TSync board, via a 12 inch Type N to SMA adapter cable (supplied with the TSync board)



AGPS (A-GPS, Assisted GPS) / tsync_agps.h file

- The Tsync driver contains a file called **tsync_agps.h.** This file is to support AGPS in the SecureSync/9483. As of at least Feb 2015, it's not supported/compatible with the Tsync family timing boards.
- The TSync boards do not support AGPS and there are no intentions at this time (Feb, 2015) to add AGPS to the timing boards
- While looking at this, I went through the linux folder on the CD and began unzipping things and found a file called tsync_agps.h, and just made a copy in the windows source tree to see if I could progress. It did seem to fix the immediate compiler errors.

Reply from Morgan (16 Feb 14) From talking to our engineer's the TSync boards do not support Assisted GPS. And at this time, I'm not aware of any intentions of it being added. Any reference that you may have seen to Assisted GPS in the Tsync driver is strictly due to this same driver also being shared with SecureSync, which does support GPS. So thereis no tsync_agps.h file present in the driver.

He also said that you shouldn't be having any issues compiling the driver that would be related to Assisted PS. But if you are getting errors when you are compiling or using the TSync driver, please send me a screenshots of the error messages this will help us diagnose the problem.

Also the TPro are a legacy boards and the TSync are the new one. The information is combined on the CD because we have customer's that have upgraded from the TPro and are familiar with the API calls. You should be using the TSync API. Let me know if this helps.

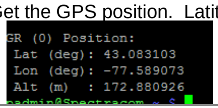
**GPS Reference Component (GR_) Calls

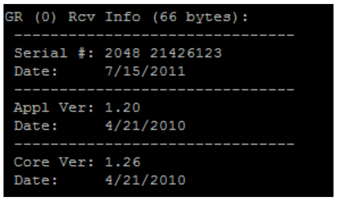
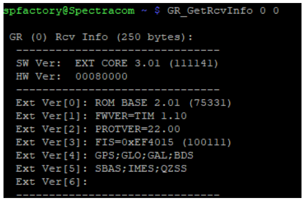
- GR calls execute the GPS receiver's protocol and determine 1PPS and time validity.
- Refer to the TSync family driver guide for a list of all GPS related API calls (1219 -5001-0050) in Arena
- **Refer to Tim Tetreault's "KITS cheat sheet" at: ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\Tsync driver calls cheat sheet**

Note the same "GR_" calls can be used with any Model receiver installed (RES-T, RES-SMT-GG as well as ublox). The Tsync drivers are upward and downward compatible with all of these receivers (the changes are for the ublox receiver are in the board's firmware- so there was no need to change driver calls for the ublox receiver/.

- These are examples - additional calls may have since been added

GPS-associated API calls

API Call	Function
TSYNC_GR_GetConstSel 0 0	The command to switch between GPS and GLONASS on a TSync board (see notes further below. Only applicable to boards with GNSS receivers
TSYNC_GR_SetConstSel 0 0	The command to switch between GPS and GLONASS on a TSync board (see notes further below. Only applicable to boards with GNSS receivers
TSync_GR_GetUtc 0 0	Shoud get UTC offset from the receiver. **Note**: this call may not be compatible with ublox receivers With a uBlox installed in a SecureSync, this command responds wth "RC_BAD_STATE". May only work with Trimble receivers.
TSYNC_GR_GetOffset 0 0	Get the GPS reference's 1PPS input offset. Offset is in nanoseconds. **GR (0) Offset: 0**
TSYNC_GR_SetOffset 0 0	Set the GPS reference's 1PPS input offset. Offset is in nanoseconds from -500 msec to +500 msec.
TSYNC_GR_GetValidity 0 0	Get the GPS validity structure. GR (0) Reference Validity: Time: 1 1PPS: 1
TSYNC_GR_GetPosition 0 0	Get the GPS position. Latitude and longitude are in radians. Altitude is in meters.  GR (0) Position: Lat (deg): 43.083103 Lon (deg): -77.589073 Alt (m) : 172.880926 padmin@Spectracom ~ $
TSYNC_GR_SetPosition 0 0	Set the GPS position. Latitude and longitude are in radians. Altitude is in meters.
TSYNC_GR_GetMode 0 0	Get the GPS receiver mode.
TSYNC_GR_SetMode 0 0	Set the GPS receiver mode.
TSYNC_GR_GetDynamics 0 0	This API call is deprecated (use TSYNC_GR_GetMode instead)
TSYNC_GR_SetDynamics 0 0	This API call is deprecated (use TSYNC_GR_SetMode instead)
TSYNC_GR_GetFixData 0 0	Get the GPS position fix data.
TSYNC_GR_GetSatData 0 0	Get the GPS satellite data. GR (0) Sat Data: ##: ch id st t f fl -------------------------- 00: 00 G06 44 0 1 00 01: 00 G24 44 0 1 00 02: 00 G28 48 0 1 00 03: 00 G17 42 0 1 00 ~ ~ ~ ~ ~ ~ ~
TSYNC_GR_GetSurveyProg 0 0	Get the GPS survey progress GR (0) Survey Progress: 0%
TSYNC_GR_GetMfrMdl 0 0	Get the GPS receiver manufacturer and model. GR (0) Manufacturer/Model: Mfr: Trimble Mdl: Resolution T
TSYNC_GR_GetRcvInfo 0 0	Get the GPS receiver info.

	With a Trimble Receiver  GR (0) Rcv Info (66 bytes): ------------------------------- Serial #: 2048 21426123 Date: 7/15/2011 ------------------------------- Appl Ver: 1.20 Date: 4/21/2010 ------------------------------- Core Ver: 1.26 Date: 4/21/2010 **With a ublox receiver**  spfactory@Spectracom ~ $ GR_GetRcvInfo 0 0 GR (0) Rcv Info (250 bytes): ------------------------------- SW Ver: EXT CORE 3.01 (111141) HW Ver: 00080000 ------------------------------- Ext Ver[0]: ROM BASE 2.01 (75331) Ext Ver[1]: FWVER=TIM 1.10 Ext Ver[2]: PROTVER=22.00 Ext Ver[3]: FIS=0xEF4015 (100111) Ext Ver[4]: GPS;GLO;GAL;BDS Ext Ver[5]: SBAS;IMES;QZSS Ext Ver[6]: -------------------------------
TSYNC_GR_GetCustom 0 0	Get the last unhandled GPS message.
TSYNC_GR_SetCustom 0 0	Send a custom message to the GPS.
TSYNC_GR_GetNumInst 0 0	Get number of GPS references instances present in the system.
TSYNC_GR_DelPos 0 0	Clear any position information that is stored in persistent inside the GPS receiver.
TSYNC_GR_GetRefId 0 0	Get reference identifier for a GPS reference instance. "GR (0) Reference Identifier: gps0"
TSYNC_GR_Reset 0 0	Reset the GPS receiver.
TSYNC_GR_GetAntenna 0 0	Get the GPS receiver antenna status GR (0) Antenna Status: (2) OPEN

Refresh rate for the GPS calls

Q Can you please tell me how accurate is the position information read from the TSYNC PCIe ?
Does the GPS position is updated constantly (interpolated?).   On a moving vehicle, I need to accurately tag image capture with time and position information. How fast can we call the TSYNC_GR_GetPosition() ?

A Reply from Keith after talking with Tim T (Jan 15) the GR calls can be performed repeatedly, several times per second.   But theTSync board only queries the receiver for information once per second.   So if the calls are performed several times per second, the calls will respond with "stale data" every second.

Note: Where <setmode>

0= Single satellite mode
1= Standard mode
2= Continuous mode
3= Averaging mode
4= Timing mode
5= Standby mode
6= Self mode

Where <dyn>

0= Land
1= Sea
2= Air
3=Stationary

Where <index> should always be a value of "0" with only one board installed.

**TSYNC_DRV_CONNECTION_ERR with “GR” or “IR” calls

- This error message may occur because:
  - The firmware is unable to communicate with the FPGA / Micro
  - There is an issue with the EEPROM (such as corrupted during a bad upgrade). The EEPROM stores configuration info for the board, such as if a GPOS receiver is installed.
  - When this error is seen with GR calls, it can indicate a problem communicating with the GPS receiver or may indicate the EEPROM may be corrupt. The antenna being connected or not connected has no effect on the ability for GR calls to respond.
- Try performing other API calls that go through the FPGA (any calls except the HW_Gettime call). A good example to try us the CS_GetTime 0 0 call. If this call works, likely an EEPROM issue (get the version info for the board). If the other calls also fail, could be an issue with the FPGA.
- **Examples below**

./GR_GetPosition 0 0

```
root@conker:~/tsync/examples#  ./GR_GetPosition 0 0
Error: connection error (TSYNC_DRV_CONNECTION_ERR).
```

./GR_GetAntenna 0 0

```
root@conker:~/tsync/examples# ./GR_GetAntenna 0 0
Error: connection error (TSYNC_DRV_CONNECTION_ERR).
```

./GR_GetAntenna 0 1

```
root@conker:~/tsync/examples#./GR_GetAntenna 0 1
Error: connection error (TSYNC_DRV_CONNECTION_ERR).
```

./GR_GetAntenna 0 2

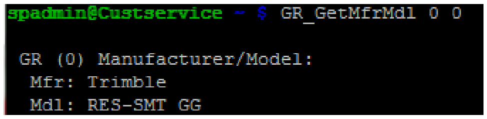
```
root@conker:~/tsync/examples#./GR_GetAntenna 0 2
Error: connection error (TSYNC_DRV_CONNECTION_ERR).
```

**Determining type of receiver installed and receiver firmware version

- There are two API calls/example programs to use in conjunction with each other to determine type of receiver installed and receiver firmware version.

A) GR_GetMfrMdl 0 0:

- Reports the receiver manufacturer and Model (Example below is with a RES-SMT-GG receiver)

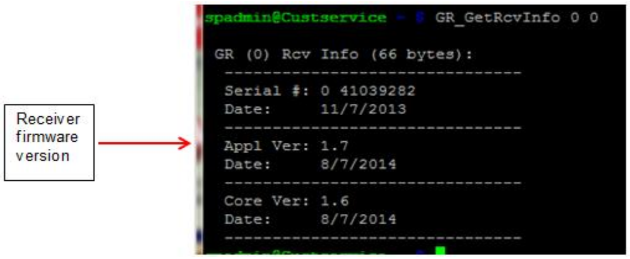


```
spadmin@Custservice ~ $ GR_GetMfrMdl 0 0

 GR (0) Manufacturer/Model:
  Mfr: Trimble
  Mdl: RES-SMT GG
```

B) GR_GetRcvInfo 0 0:

- Reports receiver firmware version (Example below is with a RES-SMT-GG receiver)

Note: Trimble refers to this release as 1.0.7, but the receiver reports it is 1.7 by dropping the ".0" in the response)



```
spadmin@Custservice ~ $ GR_GetRcvInfo 0 0

 GR (0) Rcv Info (66 bytes):
  --------------------------------
  Serial #: 0 41039282
  Date:     11/7/2013
  --------------------------------
  Appl Ver: 1.7
  Date:     8/7/2014
  --------------------------------
  Core Ver: 1.6
  Date:     8/7/2014
  --------------------------------
```

Note: As of Dec, 2014, we were still shipping the RES-SMT-GG receivers with v1.06 software installed. The batch of boards shipped to Google on 19 Dec 2014 have 1.0.7 firmware installed.

****Google Part Numbers (receiver configurations for Google's TSync boards)

Per Sadie Nedo (4 March 2019) …They have a couple different flavors of receivers in the field.

Google Part Number	Model	GPS Receiver
07012761	TSync	ResT
07096377	TSync	GG
07133775	TSync-PCIe-9001	uBlox
1046328	TSync-PCIe-9002	uBlox

**U-Blox M8T receiver (M8T)

- Refer also to "u-Blox" section in the Custserviceassistance doc: I:\Customer Service\1- Cust Assist documents\CustomerServiceAssistance.pdf

  - **ECO cut-in to support this new receiver**: Refer to ECO-00699 https://app.bom.com/changes/detail-summary?change_id=2385698104
  - **Tsync Firmware version required to use the ublox receiver**: Firmare version 3.3.2 (https://app.bom.com/items/detail-spec?item_id=1220441728&version_id=10511374578)
  - Cut-in ~24 Mar 2016 (**TSync S/N 4601 and above** have M8T receiver installed. per Jill)
  - Refer to the u-**blox Model M8T** section of the custserviceassistance doc for more info on this newer receiver.
  - All Tsync boards with GNSS receiver ship pre-licensed for Glonass (Glonass enabled)
  - Requires driver version 3.2.0 (March 2016) or higher be installed. Refer to ECO-0803
  - Requires EPROM upgrade

UBox receiver Cut-in info

TSync-PCIe conversion from Trimble Res-SMT-GG to uBlox M8T GNSS receiver

- Refer to ECO-0695 (in Arena at: https://app.bom.com/changes/detail-summary?change_id=2385697496&)
- **ECO released**: ~11 Jul, 16
- **Approximate S/N cut-in (based on date above):** S/N 4711 and above (S/N 4711 shipped on 7 Jul 2016).

TSync-PMC conversion from Trimble Res-SMT-GG to uBlox M8T GNSS receiver

- Refer to ECO-00978 (in Arena at: https://app.bom.com/changes/detail-summary?change_id=2386955663&)
- **ECO released**: ~21 Aug, 2017 (updates TSync-PMC firmware from version 1.0.0 to 1.1.0
- **Approximate S/N cut-in (based on date above):** beyond 5965 (S/N 5965 shipped on 23 July, 2017, before the ECO was released).

TS-OPT-BSH (TSync Family Interference Detection Suite Option)

- TSync Family Interference Detection Suite Option (Talen-X software for ublox receiver)
- Refer to **ECO-002685** (Feb 2021) for first usage of TS-Opt-BSH in a TSync Special.
- For only u-blox M8T GPS/GNSS receivers

Description: Adds TSync Family Interference Detection Suite support to TSync with u-blox M8T GPS/GNSS receiver (Note: Not available with SAASM GPS configuration).

***Customer desire/need to replace Trimble Res-SMT receiver with UBlox M8T

- Refer to Salesforce Case 256115
- Due to differences in TSync bootloader firmware, TSync-PCIe needs to be returned to factory for this change to be implemented.

Q Keith sent to Dave West

Don M recommended I run this question by you (he believes my thoughts are correct) … A TSync customer in New Zealand (I believe) is asking what needs to be done to switch from Trimble Res-SMT-GG to uBlox M8T GNSSS receiver. Does this conversion have to be performed at the factory, or can it be performed in field? I believe EEPROM needs to be reprogrammed at factory?? Thanks much, Keith

A [2:21 PM] David West (21 Jan 2021)

The bootloader is the guilty party and requires being returned to us.

EEPROM might be able to upgrade in field, but don't really know. Either way, it doesn't matter because bootloader is different

[2:24 PM] David West

I just checked and confirmed the FPGA is the same P/N and rev on them

---

Galileo constellation support with ublox M8T receiver

- Galileo availability started with ublox M8T receviers having firmware version 3.0.1 installed (~Dec, 2016).
- Galileo was not initially available in the first version installed (v2.3.0) when we started using ublox receivers.
- GPS and Galileo (no longer GPS/Glonass) are the default constellatons starting in TSync-PCIe firmware version 3.4.7.

Need to reset GPS position if the TSync is moved to a new stationary location

A) TSync firmware versions 3.4.7 and above (ECO 1625 for 3.4.7 released 29 March 2018)

- Receiver resurveys after every power-up
  - per the v3.4.7 release notes: "GNSS receiver will now restart survey on power-up or board reset"

B) TSync-PCIe firmware versions prior to v3.4.6 and below

- Receiver Position needs to be deleted to start a new survey
  - Perform a **GR_DelPos 0 0** to force a resurvey.

~~**(Unlike SecureSyncs/9400s) the work-around of clearing postion on reboot is not available for TSync series timing boards is not available**~~

- Similar info below is also in the "ublox" section of the custsserviceassist doc.
- The same work-around for SecureSyncs/9400s to delete position/resurvey after each reboot was implemented in TSync firmware upgrade version 3.4.7 (~29 Mar 2018)
- TSync boards with firmware versions prior to v3.4.7, which are synced to GPS after shipment from our factory and then relocated elsewhere will need the GPS position manually cleared before it can sync at the new location.

~~**Email from Keith to Tony DiFlorio (20 Feb 17)** Tim T just clarified that the work-around that we are using for SecureSyncs and NetClock 9400s with a ublox receiver installed (clearing the position hold after each boot-up for a new survey to be performed) is not available as a work-around for the TSync boards with a ublox receiver (or for a Res-SMT-GG receiver with version 1.0.9 firmware installed).~~

~~The work-around that was added to the SecureSyncs and 9400s needed to be implemented in the "network processor software" (not in the KTS Timing system software). The TSync boards share the KTS timing system software with the SecureSyncs/9400s. But unlike the SecureSyncs/9400s, the TSync boards don't contain the Network processor software, preventing this work-around from being able to be added to the TSync boards.~~

~~The GNSS receiver is cleared of its position before its shipped from our factory. But if the TSync board is synced to GPS at a location after the board had been received and then relocated thereafter, the TSync board will continue to need its position manually cleared before it can sync at the new location. Unlike SecureSyncs, there are no expected work-arounds/changes expected for the TSync series boards to be able to clear their position automatically.~~

Q. It never achieved stable synchronization -- the XO disciplining state was oscillating wildly between state 2 and 5.

A per Dave S (10 Jun 17) One of the characteristics of the new u-blox receiver is that it does not automatically "resurvey" its position in standard (stationary) mode if the unit is relocated. Unfortunately, the position on this unit does not appear to have been cleared before shipment to allow for resurvey and operation once powered at Google. Deleting the position on the receiver should allow this to resynchronize properly. This can be achieved using the "GR_DelPos 0 0" command. If operating in stationary mode, this will be required whenever the card is relocated. An alternative would be to operate in mobile mode, which doesn't require the survey.

Factory default enabled satellite constellations

A) Firmware versions 3.4.7 and above: GPS and Galileo

- ***Per the 3.4.7 Release Notes*** "GPS/GNSS setting now defaults to GPS/GAL not GPS/GLO for U-blox M8T receivers.

B) Firmware versions 3.4.6 and below:

- **Firmware versions 3.4.6 and below:** GPS and Glonass

Desire to upgrade to u-blox receiver (from Res-T or Res-SMT-GG receivers)

A) firmware versions 3.4.7 and above:

- uBlox M8T receiver firmwware can now be updated in the field.
- Per the v3.4.7 Release Note: "Added support for ublox M8T upgrade"

B) firmware versions 3.4.6 and below:

- TSync board/ublox receiver needs to be returned to factory for firmware and EEPROM change??
- Requires linux driver be updated to at least version 3.2.0 (Mar 2016). Refer to ECO-0803

Resurvey: Need to reset the u-Blox receiver positon (if relocated while in Stationary mode)

A) TSync firmware versions 3.4.7 and above (ECO 1625 for 3.4.7 released 29 March 2018)

- ➢ Receiver resurveys after every power-up
- per the v3.4.7 release notes: "GNSS receiver will now restart survey on power-up or board reset"

B) TSync-PCIe firmware versions prior to v3.4.6 and below

- ➢ Receiver Position needs to be deleted to start a new survey
- Perform a **GR_DelPos 0 0** to force a resurvey.

~~➢ **Note:** The work-around of resurvey after every power-up (like SecureSyncs/9400s) is not available for TSync series timing boards~~

- ~~➢ The TSync boards don't contain the network processor software, so there is no way to implement the work-around to the TSync boards.~~
- ~~➢ TSync boards that are synced to GPS after shipment from our factory and then relocated elsewhere will need the GPS position manually cleared before it can sync at the new location.~~

~~**Email from Keith to Tony DiFlorio (20 Feb 17)** Tim T just clarified that the work-around that we are using for SecureSyncs and NetClock 9400s with a ublox receiver installed (clearing the position hold after each boot-up for a new survey to be performed) is not available as a work-around for the TSync boards with a ublox receiver (or for a Res-SMT-GG receiver with version 1.0.9 firmware installed).~~

~~The work-around that was added to the SecureSyncs and 9400s needed to be implemented in the "network processor software" (not in the KTS Timing system software). The TSync boards share the KTS timing system software with the SecureSyncs/9400s. But unlike the SecureSyncs/9400s, the TSync boards don't contain the Network processor software, preventing this work-around from being able to be added to the TSync boards.~~

~~The GNSS receiver is cleared of its position before its shipped from our factory. But if the TSync board is synced to GPS at a location after the board had been received and then relocated thereafter, the TSync board will continue to need its position manually cleared before it can sync at the new location. Unlike SecureSyncs, there are no expected work-arounds/changes expected for the TSync series boards to be able to clear their position automatically.~~

Email from Dave to a customer (15 Nov 2018) When moving to a different location or in the field, establishing time sync as quickly as possible will ensure the TSync board will properly stabilize. I am assuming these will all be stationary after powering up in the field.

The receivers should be location reset first.

Try this procedure:

1. Set-up GPS antenna with no obstructions to the sky.
2. Power up equipment and login into host PC.
3. Set the mode of the TSync board to stationary land mode. GR_SetMode.exe 0 0 1 0 (this should be set be default)
4. Send command GR_SetConstSel.exe 0 0 3 to board
5. Delete current position. **GR_DelPos 0 0**
6. Wait a few minutes then verify that TSync board is using both GPS and GLONASS constellations.
7. Wait 30 minutes until the resurvey has completed and the TSync-PCIe board has stabilized.
8. Check the phase error or TFOM level.

This process should result in the fastest stabilization.

Please let me know if you have any questions.

**Trimble Res-SMT-GG receiver

- Refer also to "Res-SMT-GG" section in the Custserviceassistance doc: I:\Customer Service\1- Cust Assist documents\CustomerServiceAssistance.pdf
- (Internal GNSS receiver) cut-in ~28 March 2014 (ECN 3436) with Tsync Firmware version 2.21 (RES-SMT-GG receiver firmware was at version 1.06).   Note, it looks like the first Work Order using the newer RES-SMT-GG receivers was 22896 (Rev J). I BELIEVE the Cut-in Serial Number is 02572.
- Earlier Firmware versions 2.20 and below must be updated to be able to communicate with a RES-SMT-GG receiver (using a RES-SMT-GG receiver with an earlier version firmware below 2.2.1 will cause Antenna Sense = unknown and won't track any satellites for example)
- TSync boards with firmware versions 2.21 or higher are backward compatible with the earlier Trimble RES-T receivers (versions 2.21 and above can sense which Model receiver is installed).
- All Tsync boards with GNSS receiver ship pre-licensed for Glonass (Glonass enabled)

Ability to upgrade firmware of a RES-SMT-GG receiver on a TSync board in the field

- **Linux**: Requires newer Linux driver **version 3.2.0** (or higher) be installed in the system
- **Windows system**: Requires newer Windows driver **version 3.2.1** (or higher) be installed in the syste,m

Res-SMT-GG Firmware versions (summary info, in ascending version order)

Trimble RES-SMT-GG Version	Notes	Associated ECN/ECO
1.06	Has intermittent issues such as UTC offset changing on its own. Has one second error starting ~Jul 2016 (associated with Leap second pending)	N/A
1.07	Has intermittent issues such as UTC offset changing on its own. Has one second error starting ~Jul 2016 (associated with Leap second pending)	
1.08	Fixes intermittent issues associated with factors such as UTC offset changing on its own. Has one second error starting ~Jul 2016 (associated with Leap second pending)	
1.09	"Permanent fix" for the one second error starting ~Jul 2016 (associated with Leap second pending)	**ECO 979** (~12 Sept 2016)

Resurvey: Need to reset the RES-SMT-GG receiver positon if relocated while in Stationary mode

A) TSync firmware versions 3.4.7 and above (ECO 1625 for 3.4.7 released 29 March 2018)

- Receiver resurveys after every power-up
- per the v3.4.7 release notes: "GNSS receiver will now restart survey on power-up or board reset"

B) TSync-PCIe firmware versions prior to v3.4.6 and below

- Receiver Position needs to be deleted to start a new survey
  - Perform a **GR_DelPos 0 0** to force a resurvey.

***Distinguishing Glonass satellites versus GPS satellites

- GPS satellites have an Satellite ID number range of 0 to 59 and are reported as "**GP**"
- Glonass satellites have an Satellite ID number range of 60 and above and are reported as "**GL**"
- Glonass satellites have an Satellite ID number range of ?? and above and are reported as "??"

Requirements to use Glonass constellation satellites

- TSyncI (internal receiver) requires Model 8230 antenna and RES-SMT-GG receiver be attached to the board (RES-SMT-GG receivers started shipping with all Tsync boards near the end of March 2014).

  Note: it looks like the first Work Order using the newer RES-SMT-GG receivers was 22896 (Rev J). I BELIEVE the Cut-in Serial Number is 02572.

- TSyncE (External receiver) need to use newer Acutime GG antenna (refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf)
- Both TSyncI and TSyncE boards require firmware versions 2.2.1 (or v2.2.2 for PTP boards) or higher to support Glonass capability

Requirements to use Galilieo consteallation satellites

- TSyncI (internal receiver) requires Model 8230 antenna and ublox receiver that has at least firmware version 3.0.1 installed (I don't believe Galileo is available with the RES-SMT-GG receiver)
  - Galileo availability started with ublox recevirs having firmware version 3.0.1 installed (~Dec, 2016).
  - Galileo was not initially available in the first version installed (v2.3.0) when we started using ublox receivers.
- ~~TSyncE (External receiver) need to use newer Acutime GG antenna (Refer to Acutime GPS Antennas)~~
- Both TSyncI and TSyncE boards require firmware versions 2.2.1 (or v2.2.2 for PTP boards) or higher to support Glonass/Galileo capability??

**Enabling /Disabling Constellations (GPS, Glonass and/or Galileo)

- Can enable just GPS, just Glonass, or both constellations.
- There are only two commands associated with Glonass (**GR_GetConstSet** and **GR_SetConstSet**)

  Email from Tim Tetreault (29 Apr 2014)
  The command to switch between GPS and GLONASS on a TSync board is "GR_SetConstSel".

  - "./GR_SetConstSel 0 0 1" will set to only "GPS".
  - "./GR_SetConstSel 0 0 2" will set to only "GLONASS".
  - "./GR_SetConstSel 0 0 3" will set to "GPS & GLONASS". (This is the DEFAULT setting for our unit)

  They will be able to see a change in the # SAT seen by using "GR_GetFixData 0 0" & "GR_GetSatData 0 0".

The testing that I did showed that the receiver did not remember the settings on a power cycle. It always came back to GPS/GLONASS.

GPS/Glonass/Galileo sync requirements/time to achieve sync in Standard-Stationary mode

- Tsync boards need to initially track four satellites to complete the GPS survey.
- Once the survey has completed, after each subsequent boot-up the receiver needs to obtain the GPS to UTC offset value before it can achieve sync. This can take up to 13.5 minutes after starting to track satellites, depending on where the GPS signal is in relation to when it started receiving the GPS signal.

Alleviating the need to perform a GPS survey after initial install

- Can use the **GR_GetPosition** API call/example program to manually enter current position (latitude, longitude and height above sea level) to alleviate need to perform a GPS survey after initial install.

  GR_GetPosition <device ID> <index> <lat> <log> <alt>

  (With one Tsync board installed in the system): GR_GetPosition 0 0 <lat> <log> <alt>
- If call is performed during a GPS survey, it will exit the GPS survey and go into Position Hold mode.
- After each subsequent boot-up, the receiver still needs to detect the UTC offset, or have this value entered manually/with a script file (see info below on how to set this value, if desired)

Speeding up time it takes to sync after each boot-up

- Can use the **CS_SetTimeScaleOff 0 2 16** API call/example program to manually enter position to alleviate need to perform

I wanted to let you know there is a command available to be able to speed up the time it takes to sync the TSync-PCIe board after each boot-up. The delay to sync after each boot-up is the time it takes for the receiver to read the GPS to UTC offset (currently set to 16 seconds). This piece of data is transmitted by the GPS satellites every 13.5 minutes. Depending on when the receiver starts listening to the GPS signal in relation to when this information was last transmitted, it can take up to 13.5 minutes to read this value (worse-case scenario. it if just missed it- so it has to wait for it to be transmitted again). The TSync board doesn’t declare sync until it knows this value after each boot-up.

This value can be programmed into the board after each boot-up, as desired (this value is not stored inside the receiver or inside the timing board). Note this particular value only changes if a leap second is asserted to UTC time (which typically only happens every couple of years).

The command used to set the offset value to the current value of 16 seconds, to allow sync to occur without waiting for it to be read from GPS is **./CS_SetTimeScaleOff 0 2 16**. To check if it is set correctly, run **./CS_GetTimeScaleOff 0 2**. It should report back “**16**”. Sync can then occur!

Where:

UTC=0 (power-up default)

TAI=1

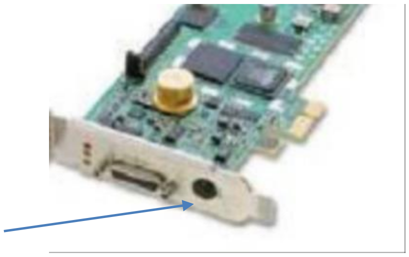
GPS=2

Local=3

**Desire to Blacklist (Mask/Unmask) a Satellite from being received by a Trimble RES-SMT-GG receiver

- Refer to Salesforce case 179110
- Refer to the sob-section “Desire to Blacklist (Mask/Unmask) as satellite:” (in the “**Trimble RES-SMT-GG**” section) of the cust assist tech note: ..\CustomerServiceAssistance.pdf

**Acutime antennas (“TSyncE” / “TSync-PCIe-002 / TSync-PCIe-012”)



- **Refer to: “Acutime antennas**” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- **External GPS or GNSS receiver (**GPS/GNSS receiver is inside the Acutime antenna housing**-** not on the TSync-PCIe board)

Note: TSync-PCIe-002/ TSync-PCIe-012 (TSyncE-PCIe) boards are last time buy/being discontinued ~Jan 2021.

Acutime GPS/GNSS antennas (GPS or GNSS receiver is inside Antenna housing)

A) Acutime 360 antennas

- Starting with TSync firmware version 3.4.9 (Dec 2018), TSync boards are now compatible with Trimble AcUtime 360 antennas (**Trimble P/N**: 106406-00)

Acutime 360 GNSS Antennas (pre-configured at factory to output NMEA- instead of the TSIP)

- Refer also to:..\CustomerServiceAssistance.pdf
- Our P/N **E025R-9000-0001 (in Arena at)** https://app.bom.com/items/detail-spec?item_id=1255453450&version_id=11133851398&orb_msg_single_search_p=1
- being released with just the Acutime 360 on it. Final BOM with have PD and software to program for NMEA output. This will be added prior to customer shipment.

Support for Acutime 360 antenna:

- Support to use Acutime 360 antenna was added in TSync firmware update version 3.4.9 (~7 Dec 2018)
  - Refer to **ECO-FAI-1886** in Arena: https://app.bom.com/changes/detail-summary?change_id=2395198855&orb_msg_single_search_p=1

  Refer also to: ..\..\PSB, PSP software updates\TSYNC boards\TSync firmware updates (PCIe, cPCI, PCI104)\TSync-PCIe\TSync-PCIe firmware updates

Associated Model 1169 GPS Fiber Optic Isolator for Accutime antennas

- Refer to “**Model 1169 GPS Fiber Optic Isolator (TX and RX) for GPS**” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

Lightning Protection/Lightning suppression for TSync-PCI boards

- Refer to: Salesforce Case 181946
- Refer to:..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\Lightning Protection

Email from Dave L (8 Jan 2019) I am sorry this took a while but I was able to get some information for a suitable Lightning protector for our Timing Board the TSync-PCIe-002. This will be compatible with the external receiver model.

This can be purchased from a company called Zap-Tech.

http://www.zap-tech.com/#pg-home-3

The model number you would order is #40-422-001.

I have attached a photo and spec sheet. We do not have this in our catalog but you may order direct from Zap-Tech.

B) Acutime GPS/GG antennas (listed oldest to newest Models)

- TSyncE-PCIe boards iniitally used **Acutime GPS (Acutime Gold)** antenna until **~Mar 2014.**
  - they then started using the **Acutime Res-SMT-GG (Acutime GG**) GNSS antennas for GPS + Glonass
  - as the Acutime-GG has now been discontinued (sometime before Aug, 2018) we are in the process of transitioning to the **Acutime 360 antennas**

TSIP Protocol versus NMEA protocol (TsyncE boards and Acutime Antenna)

- TsyncE-PCIe boards are normally programmed in their EEPROM to use **TSIP** protocol packets to communicate with the GPS/GNSS receiver located inside both the earlier **Trimble Acutime Gold** (GPS) and its successor, the **Trimble Acutime-GG** (mulit-GNSS) antenna housing.
- The **Acctime-GG** antennas were shipped to us pre-programmed for **TSIP** protocol
  - We need to reprogram the Acutime-GG antennas here at our factory to “**NEMA**” protocol (instead of TSIP) in order for this antenna to work with the legacy **TSAT timing boards /** or **with the fiber optic converters**

email from Sadie to Invensys (7 Aug 2018) If you’re intending to use the Fiber up/down converter for this customer you’ll require the board that we pre-program as NMEA specifically for Schneider to be compatible with the fiber kit. The programming for the board is part of the K0204A kit development which hasn’t been started….this will be initiated with a first time buy on a PO.

TsyncE board (external receiver) doesn’t recognize Acutime-GG antenna is connected

1) A TSync board needs to have **firmware versions 2.2.1** or above installed to be compatible with the Acutime-GG antenna. versions prior to versions 2.2.1 are only coomptable with Acutime gold antennas.
   - The version isn’t a requirement when the TSync board is using an Acutime Gold GPS antenna (the predecessor to the Acutime-GG antennas).
2) The TsyncE-PCIe boards are programmed in their EEPROM to use the **TSIP** protocol to interface with its

external receiver

- The Acutime-GG antenna needs to be “unprogrammed” (as we receive them from Trimble) to interface with TSync boards. They need to be reprogrammed to use **NMEA** protocol when being used with a legacyTSAT timing board.

Q. I connected the new receiver/antenna but it is not detected by the TSync. GR_GetAntenna 0 returns UNKNOWN
I reconnected the old receiver/antenna and that returns OK.
My Firmware is 2.2.1 already so I did not update it.

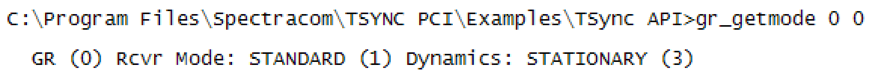
A. (From Tim Tetreault 3 Apr 2014) The GR_GetAntenna command will tell if the antenna is OK. I tested this with the new “GG” antenna and it does work.
Have the customer try:
./GR_GetMfrMdl 0 0
./DCS_GetOptions 0
./FS_GetVersion 0 6

- The manufacture’s info should report back “**Acutime GG**”
- The options command should report back “**MULTIGNSS**” if the card is setup correctly
- The version command should report back “**2.10**” for the correct EEPROM version.

**Calls for setting the TSync-PCIe GPS receiver Mode/Dynamics code:

TSync-PCIe command to switch between Stationary, Mobile or single sat modes:

GR_SetMode 0 0 “A” “B” (Where “A” defines the receiver mode and “B” defines the dynamics code, as listed below)



```
C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API>gr_getmode 0 0

  GR (0) Rcvr Mode: STANDARD (1) Dynamics: STATIONARY (3)
```

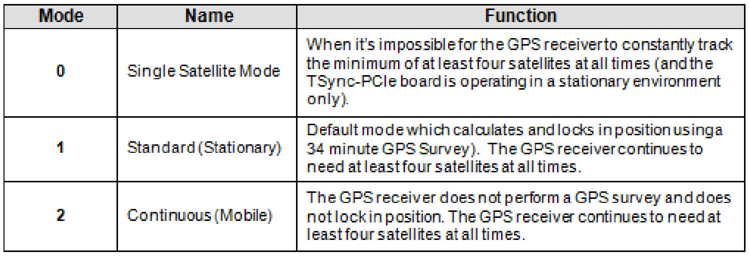
Example: GR_SetMode 0 0 2 2 (sets the Mobile Mode for Airborne use**)**

“A” values
0= Single Sat mode
1= Standard (Stationary) mode
2= Continuous (Mobile) mode
3= Averaging
4= Time-only
5= Standby
6= Self

“B” values
0= Land
1= Sea
2= Air
3= Stationary
4= Unknown

Note: Modes 3, 4, 5 and 6 are supported by the GPS receiver but are not used with the TSync-PCIe boards. They are for use in SecureSync, which uses the same GPS receiver and similar software.

There are **seven** receiver mode available for the GPS receiver, though **four of which are not used with the TSyncl-PCIe** boards. The modes that are used with the TSync boards are as follows:



Mode	Name	Function
0	Single Satellite Mode	When it's impossible for the GPS receiver to constantly track the minimum of at least four satellites at all times (and the TSync-PCIe board is operating in a stationary environment only).
1	Standard (Stationary)	Default mode which calculates and locks in position using a 34 minute GPS Survey). The GPS receiver continues to need at least four satellites at all times.
2	Continuous (Mobile)	The GPS receiver does not perform a GPS survey and does not lock in position. The GPS receiver continues to need at least four satellites at all times.

- For more info on the Standard, Mobile and Single Sat modes for a Res-T receiver, refer to: Mobile/Stationary/Single Satellite modes with a Resolution-T receiver (NetClock, TSync and SecureSync)

- This GPS Mode configuration persists in the GPS receiver through power cycles and reboots. It does not need to be issued each time the timing board starts-up.

(5/17/11) Known issue exists with TSyncE-PCIe boards when used with Acutime antenna (at least firmware versions 2.1.0 and 2.0.0- maybe newer if not resolved in the next update – Version 2.00 added the mobile mode, so earlier versions not applicable).

Like SecureSync, the default setting for the TSync-PCIe's Dynamics code is "Land". Also like SecureSync, when the GPS mode is set to Standard (Stationary mode) and after the GPS survey has been completed, the dynamics code is supposed to automatically switch to "Stationary". An issue with the TSync-PCIe board's firmware is causing the dynamics code to continue being reported as "Land", even after the survey has completed. This appears to be a reporting issue only, as it acts like it it's in stationary mode.

A) GPS Continuous/ Mobile Mode

- Requires at least four satellites be tracked at all times.
- Must requires UT1 correction bit before going into sync (transmitted by sats every 12.5 minutes)
- GPS survey is not performed while in Mobile mode.

3. Mobile mode with a TSync series board and internal (on-board) GPS reciever

To enter **mobile mode**, you need to set the GPS reference to continuous mode (mode = 2). We also recommend then changing the dynamics mode to land (dynamics = 0). With example programs, it would be the following:

GR_SetMode 0 0 2

GR_SetDyn 0 0 0

GR_SetMode 0 0 "A" "B" (Where "A" defines the receiver mode and "B" defines the dynamics code, as listed below)

Example: **GR_SetMode 0 0 2 0** (sets the Single sat Mode for land use)

"A" values
0= Single Sat mode
1= Standard (Stationary) mode
2= Continuous (Mobile) mode
(see note below) 3= Averaging
(see note below) 4= Time-only
(see note below) 5= Standby

"B" values
0= Land
1= Sea
2= Air
3= Stationary
4= Unknown

(see note below) 6= Self

Note: Modes (“A” above) 3, 4, 5 and 6 are supported by the GPS receivers, but are not used with the TSync-PCIe boards. They are for use in SecureSync, which uses the same GPS receivers and similar software.

4. Mobile mode with a TSync board and external GPS reciever

GPS Mobile mode with TSyncE-PCIe timing boards (GPS receiver located in the Acutime antenna- not on the board) 1/03/11 KW- According to Denis Reilly and Paul Myers, mobile mode has not been tested with the TSyncE-PCIe boards and is not currently supported (Currently, mobile mode is only supported with TSyncI-PCIe boards). It is not known at this time whether the Acutime antennas will operate with the same command structure as the Resolution-T receivers that are used in the TSyncI-PCIe boards. And even if the commands are compatible, it’s not known if the Acutime will operate normally while moving.

Email from Dave Sohn Normal stationary mode requires 4 satellites to start a position survey, which is ~30 minutes. There is a single satellite mode, but it still requires a position survey, which is ~30 minutes. Based on what you’re looking to do, you will want to run the GPS receiver in mobile mode, which doesn’t require a position survey. However, mobile mode will be slightly less accurate than stationary mode.

Degradation of timing/positional accuracies in mobile mode (Continuous mode)

- The amount of degradation while in motion varies depending on whether the Timing board has the earlier Model Res-T GPS receiver installed or a newer Res-SMT-GG GNSS receiver installed.
- Refer to either the “**Trimble Res-T**” or “**Trimble Res-SMT-GG**” receiver section (as applicable) in the Customerserviceassistance document for details.

B) GPS single satellite mode

- Must still complete the GPS survey (requires tracking at least four satellite for 34 minutes) before the minimum requirement drops to only one satellite to prevent going into holdover mode
- Once the GPS survey has been completed, min satellites then drops to just one.

GR_SetMode 0 0 “A” “B” (Where “A” defines the receiver mode and “B” defines the dynamics code, as listed below)

Example: **GR_SetMode 0 0 0 0** (sets the Single sat Mode for land use)

“A” values
0= Single Sat mode
1= Standard (Stationary) mode
2= Continuous (Mobile) mode
(see note below) 3= Averaging
(see note below) 4= Time-only
(see note below) 5= Standby
(see note below) 6= Self

“B” values
0= Land
1= Sea
2= Air
3= Stationary
4= Unknown

Note: Modes (“A” above) 3, 4, 5 and 6 are supported by the GPS receivers, but are not used with the TSync-PCIe boards. They are for use in SecureSync, which uses the same GPS receivers and similar software.

In single satellite mode, it is not enough just to manually enter the position information (lat and long). Each time the board is started-up, it has to be commanded to single sat mode and must be able to track at least four satellites in order to obtain the UT1 correction factor before it will go into initial sync (about 13 minutes or so). However, after that, unlike the standard mode, it only needs one satellite to maintain sync (instead of the normal four satellites).

Email from Dave Sohn:
My initial testing seems to show that in order to get initial sync, we need to acquire 4 or more satellites and download the almanac info. Once that has happened, we can drop down and sync with a single satellite. No survey is necessary to be completed for single satellite mode.

Satellite mode will have to be set each time the board is restarted. Setting an accurate position should help the timing and should be stored across power cycles.

Email from Dave Sohn
Normal stationary mode requires 4 satellites to start a position survey, which is ~30 minutes. There is a single satellite mode, but it still requires a position survey, which is ~30 minutes. Based on what you're looking to do, you will want to run the GPS receiver in mobile mode, which doesn't require a position survey. However, mobile mode will be slightly less accurate than stationary mode.

To enter single sat mode, you need to set the GPS reference to continuous mode (mode = 2). We also recommend then changing the dynamics mode to land (dynamics = 0). With example programs, it would be the following:
GR_SetMode 0 0 2
GR_SetDyn 0 0 0

Email Keith sent to Steve Rodgers (8/31/12)
I was wondering if the TSync-PCIe timing board had declared sync yet (SS_GetSync 0 should report "TRUE")? It may have, by now.

We normally only recommend the single satellite mode be used when absolutely necessary, so we don't deal with it very often. One thing that I had forgotten about this seldom used mode is the initial requirement that the TSync-PCIe board must still be able to track at least four satellites at the beginning. Then, once sync has been declared (sync is true) this minimum requirement is then relaxed to just one satellite tracked from that point forward (unless the timing board is relocated somewhere else). Normally, it requires at least four satellites be tracked all the time to remain in sync.

The issue with not initially requiring four satellites is that without a good 3D fix being calculated (latitude/longitude and antenna height), timing from a GPS receiver without this positional information can be very incorrect. The GPS receiver needs a good position to know where it's at, relative to the satellites in order to provide good timing information. It takes a minimum of four satellites to achieve this 3D fix.

In the single satellite mode, the GPS survey does still need to be initially performed before sync status goes true. The total (cumulative) time for a survey to be completed is about 34 minutes, as long as the receiver can continue to track at least four satellites. When tracking at least four satellites continuously, the survey will continue to progress and will then complete about 34 minutes after tracking at least four satellites. Each time the survey is interrupted by dropping below four satellites, the survey halts until the receiver is tracking at least four satellites again. So if it's often dropping to less than four satellites, it will take much longer for the GPS survey to complete.

The GPS survey percentage complete can be read using the **GR_GetSurveyProg 0 0** API call or example program. The closer the GPS survey is to 100%, the closer it is to sync status going true.

Error log entries for GPS

A) GR entered failure state

Q A previously not seen error log entry tripped our internal safe-guard: "TSync error log entry: 2000 001 00:00:04 005 GR entered failure state."

A per Dave Sohn (10 Jun 17) This is a known entry that can occur sometimes at power up while the firmware tries to establish communication with the receiver.

Antenna cable delay/1PPS offset.

For LMR-400 or equivalent antenna cable, the antenna cable delay is 1.2ns/100 feet of antenna cable. Multiply total cable distance x 1.2 and the result is the cable delay.

Enter the calculated delay as a negative value (this moves the PPS input back, in order to account for the forward advancing that occurs as it goes through the antenna cable).

Use the **GR_SetOffset** to enter the desired cable delay / PPS offset.

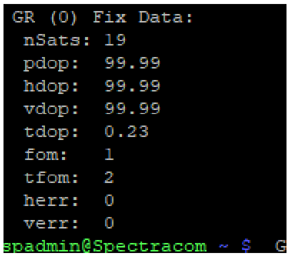
GR_GetFixData 0 0 response (GPS PDOP/HDOP/VDOP/TDOP/FOM/TFOM) FOR TSync-PCIe

```
Examples\TSync API\bin> .\GR_GetFixData.exe 0 0
```

A) Trimble receiver (fom/tfom always 0)

```
GR (0) Fix Data:
  nSats: 15
  pdop:  1.11
  hdop:  0.68
  vdop:  0.88
  tdop:  0.56
  fom:   0
  tfom:  0
  herr:  0
  verr:  0
```

B) uBlox receiver (fom/tfom not 0)



```
GR (0) Fix Data:
 nSats: 19
 pdop:  99.99
 hdop:  99.99
 vdop:  99.99
 tdop:  0.23
 fom:   1
 tfom:  2
 herr:  0
 verr:  0
spadmin@Spectracom ~ $  G
```

nats/PDOP/HDOP/VDOP/TDOP/FOM/TFOM

The parameters provided by **GR_GetFixData** are provided by the GPS/GNSS receiver itself. Depending on the type of fixes being done, and the mode of the receiver, we may only receive data for some of those variables, and zeroes for the others. We don't receive FOM and TFOM from the Trimble receivers (but do with uBlox receiver), so those will always be zero. Below are generic definitions:

HDOP (Horizontal Dilution Of Precision) is a measure of how well the positions of the satellites, used to generate the Latitude and Longitude solutions, are arranged. PDOP less than 4 gives the best accuracy, between 4 and 8 gives acceptable accuracy and greater than 8 gives unacceptable poor accuracy. Higher HDOP values can be caused if the satellites are at high elevations.

VDOP (Vertical Dilution Of Precision) is a measure of how well the positions of the satellites, used to generate the vertical component of a solution, are arranged. Higher VDOP values mean less certainty in the solutions and can be caused if the satellites are at low elevations.

TDOP (Time Dilution Of Precision) is a measure of how the satellite geometry is affecting the ability of the GPS receiver to determine time.

PDOP (Positional Dilution Of Precision) is a measure of overall uncertainty in a GPS position solution with TDOP not included in the estimated uncertainty. The best PDOP (lowest value) would occur with one satellite directly overhead and three others evenly spaced about the horizon.

FOM (0 with Trimble receivers)

TFOM (0 with Trimble receivers)

The most common issue we see when customers use an external 1PPS input ("EPP0") as the 1PPS input reference is due to ringing of the signal because of improper termination. The input impendence of the 1PPS signal is high impedance. However, the 1PPS source may desire a 50 ohm termination into the TSync-PCIe-PCIe board (instead of high impedance). If the source desires a 50 ohm load termination, the input 1PPS may likely "ring" inside of the TSync-PCIe board. The ringing of the 1PPS signal prevents a "crisp" point at which the TSync-PCIe board can trigger on. With ringing of the signal present, the detected trigger point may coincidentally be at the right point, but it could also trigger too early or too late. So, with each 1PPS input when the signal is ringing, it may trigger at a different point on the signal each time. This prevents the disciplining circuit from being able to receive a stable 1PPS that is can accurately discipline to, and so the TFOM values will also be much higher than expected.

We have seen a few other similar cases where an external 1PPS input was causing a jittery 1PPS input because of improper termination of the signal at the input of the board. These were all resolved with a 50 ohm input being applied.

To see this affect, if you have access to an oscilloscope, input the 1PPS signal from the source with a high input impedance setting on the scope. Look at the rising edge. You will likely see a ringing of the signal with no defined rising edge. Then, switch the scope to 50 ohm termination and you will likely notice the signal a become a crisp 1PPS with a very defined rising edge. This is what the board is also likely receiving and seeing, as well. To prevent the ringing of the 1PPS input from occurring and affecting the TSync-PCIe-PCIe board, terminate the 1PPS input from the source with a 50 ohm load resistor. This will often "clean up the signal" so that it can be a better reference into the board.

Note that various cable lengths with the wrong termination at the end can have differing effects on the ringing of the 1PPS input to the boards. Each TSync board includes a 50 ohm load and BNC connector (in a bag labeled with the Spectracom P/N of 1191-0000-0710. If the 1PPS line is currently unterminated, I highly recommend adding this terminator to the end of the line.

Let me know if adding the 50 ohm load to the end of the line allows the 1PPS phase error values to decrease.

****GPS reception issues with TSync boards**

- For more info, refer to: GPS\GPS reception Ap Notes\TSync-PCIe timing cards

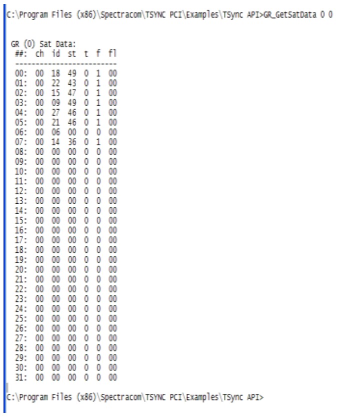
How to verify if the GPS receiver is tracking any satellites (GR_GetSatData 0 0)

- **GR_GetSatData 0 0** <enter>

This is an API Call/example program to report the number of satellites being tracked (and signal strengths of those sats)

This call returns a table with 32 rows (00 through 31 labeled as ##) (note- only the first 12 rows are used at this time). The "ST" column is the strengths for each satellite. The total number of rows where the ST is not 00 is the number of satellites currently being tracked

Note: With the antenna installed outdoors with a clear view of the sky sky - as much of a 360 degree view of the horizon as possible- no less than four rows in this list should normally have an "ST" value in the high 30s or 40s- these typical values indicate the signal strength is usable by the receiver.



```
C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>GR_GetSatData 0 0

 GR (0) Sat Data:
  ##:  ch  id  st  t  f  fl
  -------------------------
  00:  00  18  49  0  1  00
  01:  00  22  43  0  1  00
  02:  00  15  47  0  1  00
  03:  00  09  49  0  1  00
  04:  00  27  46  0  1  00
  05:  00  21  46  0  1  00
  06:  00  06  00  0  0  00
  07:  00  14  36  0  1  00
  08:  00  00  00  0  0  00
  09:  00  00  00  0  0  00
  10:  00  00  00  0  0  00
  11:  00  00  00  0  0  00
  12:  00  00  00  0  0  00
  13:  00  00  00  0  0  00
  14:  00  00  00  0  0  00
  15:  00  00  00  0  0  00
  16:  00  00  00  0  0  00
  17:  00  00  00  0  0  00
  18:  00  00  00  0  0  00
  19:  00  00  00  0  0  00
  20:  00  00  00  0  0  00
  21:  00  00  00  0  0  00
  22:  00  00  00  0  0  00
  23:  00  00  00  0  0  00
  24:  00  00  00  0  0  00
  25:  00  00  00  0  0  00
  26:  00  00  00  0  0  00
  27:  00  00  00  0  0  00
  28:  00  00  00  0  0  00
  29:  00  00  00  0  0  00
  30:  00  00  00  0  0  00
  31:  00  00  00  0  0  00

C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>
```

Breakdown of the Satellite Data table:

ch: (aka "**chnum**") receiver channel Number

Id: (aka "**SVN**" - Space Vehicle Number) Space Vehicle Number / ID Number of each satellite being tracked

Distinguishing between GPS satellites and Glonass satellites, when the Glonass Option is available (ID field)

- The Satellite ID number will indicate if that particular satellite is a GPS satellite or a Glonass satellite.
- GPS satellites have a Satellite ID Number of **0 to 59**
- Glonass satellites have a Satellite ID Number of **60 and above**

st: signal strength of the satellite being tracked (reported in dB/Hz, as provided by the receiver).

t: TRAIM (aka "**bTraim**") Is this satellite accepted by the receiver's TRAIM algorithm? (always reported as "00" with RES-T, RES-SMT and RES-SMT-GG receivers).

f: Fix status (aka "**bInfix**") Is the receiver using this satellite in its positional fix? ("0" if not in fix, "1" if satellite is being used in fix)

fl: Flags reported by the receiver (always reported as "00" with RES-T, RES-SMT and RES-SMT-GG receivers. Used only with SAASM receivers)

Driver Issue: GR_GetSatData 0 0 responding with invalid five (5) digit Satellite ID numbers (instead of two digits)

Q Several satellite PRNs reported by the GPS receiver were not within the expected range for GPS or GLONASS (such as 393238, 393237 and 393226.

A Reply from Dave Sohn (10 Jun 17) The GR_GetSatData information was updated with the change to the new receiver. Those numbers correlate to a problem of the older TSync driver reading that data. If you utilize the new TSync driver, those should report correctly.

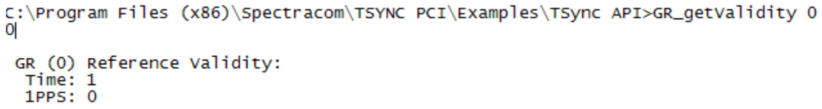
By looking at the Status LEDs on the edge of the board

(Refer to: Status LED's/Status LEDs upon reboot for more info)

Using the SS_GetSync and SS_GetHoldver API calls/example programs to determine current sync status

The current sync status is also available via a couple of API calls/example programs. The **SS_GetSync** API call indicates True when TSync-PCIe is synced or in holdover mode, or false when not in sync. To differentiate between fully Synced mode and Holdover mode, the **SS_GetHoldover** API responds with True if in Holdver mode only. It will return a False if the Board is in full Sync or if not in sync.

Verify GPS input validity using the GR_Getvalidity API call/example program (shown below)



```
C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>GR_getValidity 0
0

 GR (0) Reference Validity:
  Time: 1
  1PPS: 0
```

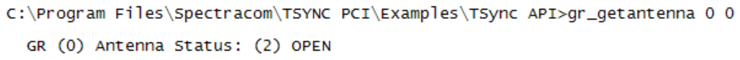
As long as at least four satellites are being tracked, the **GR_GetValidity** API call/Example program response will indicate a "1" for both Time and 1PPS. If the receiver is not tracking at least four satellites, GPS is not valid, and both responses will be a "0" instead of a "1".

Steps to troubleshoot GPS reception issues:

1) Send the GPS reception troubleshooting guide (discusses the below items)
2) Verify one instance of GPS is present (if not, GPS input not available on the board)
   - **Gr_GetNumInst 0**. Response should indicate GR Number of instances 1.
   - Look at how many sats are being tracked

3) **GR_GetSatData 0 0** <enter> (example is shown just prior to these steps).
   - The GR_GetSatData API call/example program reports the number of satellites currently being tracked. If this value is less than 4, GPS reception is not qualified.

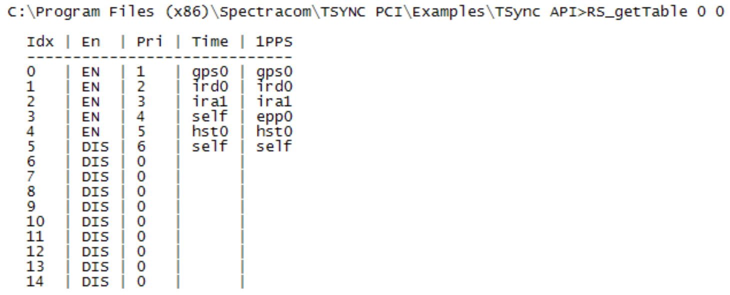
4) Verify Antenna Problem alarm not asserted: **GR_GetAntenna**



```
C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API>gr_getantenna 0 0

  GR (0) Antenna Status: (2) OPEN
```

The **GR_GetAntenna** API call/example program can be used to determine if a loss of GPS reception is due to an open or short being detected in the Antenna cable. The antenna status will respond with “Open” or “short” if a cable issue has been detected.

5) Verify “gps0” is listed as an input in the Input Reference Priority table using the RS_GetTable 0 2 API call/example program (shown below)



```
C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>RS_getTable 0 0

  Idx | En  | Pri | Time | 1PPS
  ----------------------------
  0   | EN  | 1   | gps0 | gps0
  1   | EN  | 2   | ird0 | ird0
  2   | EN  | 3   | ira1 | ira1
  3   | EN  | 4   | self | epp0
  4   | EN  | 5   | hst0 | hst0
  5   | DIS | 6   | self | self
  6   | DIS | 0   |      |
  7   | DIS | 0   |      |
  8   | DIS | 0   |      |
  9   | DIS | 0   |      |
  10  | DIS | 0   |      |
  11  | DIS | 0   |      |
  12  | DIS | 0   |      |
  13  | DIS | 0   |      |
  14  | DIS | 0   |      |
```

Make sure a row of the table lists “gps0” in both the Time and 1PPS columns, and that row indicates “EN” (Enabled) in the “EN” column

6) If its tracking at least 4 satellites, look at the GPS survey progression status

GR_GetSurveyProg 0 0 <enter>. 50% is about 17 minutes remaining before Sync occurs.

Specific GPS reception issue scenarios:

A) GPS receiver is tracking at least 4 satellites, but green Sync LED not lit

If this happens, verify the input reference priority has a row (index) where “GPS 0” is the Time and PPS reference and make sure this row is enabled. If it’s disabled or the entry has been removed from the table, the board can’t sync to GPS, even if it’s tracking at least four satellites.

To verify GPS is still an enabled reference in the saved (User) table (or the default table, if the saved table doesn't exist because a working table hasn't ever been saved), first perform a **RS_GetTable 0 0** (shows the default factory table configuration). Then perform an **RS_GetTable 0 1** (shows the default User table configuration).

Make sure both tables contain a row that has "GPS 0" in both the Time and 1PPS columns. If it does, make sure this row is set to Enabled. If this row is either disabled or does not exist, even if **GR_GetValidity** is true, the TSync-PCIe board can't sync to GPS input.

Verify GPS survey- it may still be in progress- **GR_GetSurveyProg 0 0** <enter>. 50% is about 17 minutes remaining before Sync occurs.

GPS receiver is tracking at least 4 satellites, but green and yellow LEDs are both lit (indicating it's in Holdover mode).

B) TSync is in Holdover mode.

Verify that if the board is moving, The TSync-PCIe been reconfigured for Mobile mode. If not, the movement of the platform will cause the receiver to go into Holdover mode.

C) GPS receiver is not syncing to Spectracom GSG GPS simulator

Default GSG output power level is too low. GSG defaults to -125dBm output power, while the Res-T receivers need -115dBm output power for satellite acquisition.

**DAGR GPS receiver ("Defense Advanced GPS Receiver" interface)

- Refer to "**Model 1204-02"** in the Option Card information document: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\SecureSync\Option Cards

**IRIG AM/ IRIG DCLS input synchronization

Refer to IRIG for all products – Cabling/Surge Protection/Output for a list of all available IRIG inputs and outputs.

- TSync-PCIe boards accept (AM or DCLS) = IRIG A, B, E (100Hz or 1kHz), G, NASA 36, IEEE 1344
- Via a dedicated IRIG AM input and dedicated IRIG DCLS input

**IRIG cabling (such as RG-58 coax cable)

- Refer to "**IRIG Cabling/max recommended cable distances**" in:..\CustomerServiceAssistance.pdf

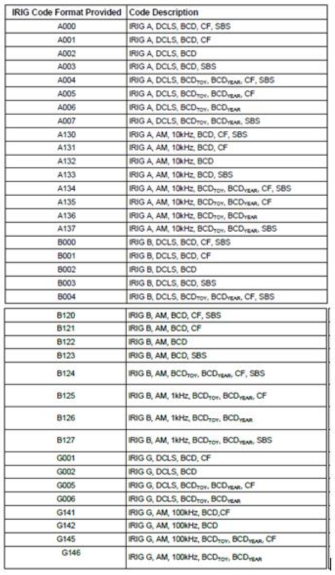
In summary

1) There is no specific recommended maximum distance for an IRIG cable to be run, as many factors can affect the lengths of cable that can be run (such as types of cable used, noise sources located near the coax cable, etc).
2) Typically, hundreds of feet of cable can be run from an IRIG source to one (or two) IRIG input devices with no problem at all. **Based on cable loss alone, 1000 feet of cable should be fine.**
3) We recommend using RG-58 coax cable
4) Though it's used on the breakout cables, we **DO NOT** recommend using **RG-174** cable (due to high DC voltage drop through cabling)

Available TSync IRIG Input / Output Formats:

Note: Does not support Manchester modulated codes Note:The Tsync manual and table does not specify B127, but according to Dave Sohn, it should also be a supported input format.

Email from Ron Dries (Sep, 2017) The TSync boards all support the same IRIG formats as the SecureSync. You are correct that the TSync manual needs to be updated. Here is a more current list from the SecureSync manual**.**



IRIG Code Format Provided	Code Description
A000	IRIG A, DCLS, BCD, CF, SBS
A001	IRIG A, DCLS, BCD, CF
A002	IRIG A, DCLS, BCD
A003	IRIG A, DCLS, BCD, SBS
A004	IRIG A, DCLS, BCD_{TOY}, BCD_{YEAR}, CF, SBS
A005	IRIG A, DCLS, BCD_{TOY}, BCD_{YEAR}, CF
A006	IRIG A, DCLS, BCD_{TOY}, BCD_{YEAR}
A007	IRIG A, DCLS, BCD_{TOY}, BCD_{YEAR}, SBS
A130	IRIG A, AM, 10kHz, BCD, CF, SBS
A131	IRIG A, AM, 10kHz, BCD, CF
A132	IRIG A, AM, 10kHz, BCD
A133	IRIG A, AM, 10kHz, BCD, SBS
A134	IRIG A, AM, 10kHz, BCD_{TOY}, BCD_{YEAR}, CF, SBS
A135	IRIG A, AM, 10kHz, BCD_{TOY}, BCD_{YEAR}, CF
A136	IRIG A, AM, 10kHz, BCD_{TOY}, BCD_{YEAR}
A137	IRIG A, AM, 10kHz, BCD_{TOY}, BCD_{YEAR}, SBS
B000	IRIG B, DCLS, BCD, CF, SBS
B001	IRIG B, DCLS, BCD, CF
B002	IRIG B, DCLS, BCD
B003	IRIG B, DCLS, BCD, SBS
B004	IRIG B, DCLS, BCD_{TOY}, BCD_{YEAR}, CF, SBS
B120	IRIG B, AM, BCD, CF, SBS
B121	IRIG B, AM, BCD, CF
B122	IRIG B, AM, BCD
B123	IRIG B, AM, BCD, SBS
B124	IRIG B, AM, BCD_{TOY}, BCD_{YEAR}, CF, SBS
B125	IRIG B, AM, 1kHz, BCD_{TOY}, BCD_{YEAR}, CF
B126	IRIG B, AM, 1kHz, BCD_{TOY}, BCD_{YEAR}
B127	IRIG B, AM, 1kHz, BCD_{TOY}, BCD_{YEAR}, SBS
G001	IRIG G, DCLS, BCD, CF
G002	IRIG G, DCLS, BCD
G005	IRIG G, DCLS, BCD_{TOY}, BCD_{YEAR}, CF
G006	IRIG G, DCLS, BCD_{TOY}, BCD_{YEAR}
G141	IRIG G, AM, 100kHz, BCD,CF
G142	IRIG G, AM, 100kHz, BCD
G145	IRIG G, AM, 100kHz, BCD_{TOY}, BCD_{YEAR}, CF
G146	IRIG G, AM, 100kHz, BCD_{TOY}, BCD_{YEAR}

API calls associated with IRIG input

- Refer to Tim Tetreault's "cheat sheet" at: ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\Tsync driver calls cheat sheet
- These are examples - additional calls may have since been added

IRIG input Commands
IR_GetCodedExp
IR_GetCtrlField
IR_GetFormat
IR_GetFreq
IR_GetMessage
IR_GetMod
IR_GetMode
IR_GetNumInst
IR_GetOffset
IR_GetValidity
IR_GetCfData
IR_GetLocal
IR_GetTimeScale
IR_SetCodedExp
IR_SetCtrlField
IR_GetTimeScale
IR_SetCodedExp
IR_SetCtrlField
IR_SetFreq
IR_SetMessage
IR_SetMod
IR_SetMode
IR_SetOffset
IR_SetLocal
IR_SetTimeScale
IR_GetRefId

IRIG input minimum signal level requirements (from the TSync data sheet)

A) IRIG AM (Amplitide Mudulation) ("ira0" reference)

- **Amplitude required**: 500 mV p-p minimux,10 V p-p maximum
- **Modulation Ratio**: 2:1 min, 6:1 max• Input Impedance: >10 K Ohms
- **Common Mode Voltage**: ±150 VDC max
- **Input Stability**: Better than 100 ppm

B) IRIG DCLS (Differential or Single Ended) ("ird0" reference)

- **Differential Amplitude**: 200 mV p-p min, 5 V p-p max - 7V to +12 V DC max common mode voltage (RS-485/RS-422 compatible)
- **Single Ended Amplitude**: +1.3V VILmin, +2 V VIH max (TTL compatible)

Note: when using a Fiber converter (such as the Model 2817-R-SM) between the IRIG source and the TSync board, need to use IRIG DCLS (fiber converters are not compatible with IRIG AM signals).

Input impedances/input termination requirements

A) IRIG AM Input impedance= 10 kohms (minimum)

- Does not need to be terminated into 50 ohms

Signal levels

- **Minimum IRIG AM input** signal level= 500 mVp-p
- **Maximum IRIG AM input** signal level= 10Vp-p

Email from Dave L (27 Jan 17) The IRIG AM + and IRIG AM – input does not require a 50 ohm load. The input impedance is 10 K ohms. Sharp edged signals like IRIG DCLS and 1PPS are affected by ringing so they generally need to have a terminator. The AM Signals do not.

Desire to use Fiber converters between IRIG AM source and IRIG AM input

- Fiber converters can’t be used with IRIG AM. Can only be used with IRIG DCLS input (additional info futher below)

B) IRIG DCLS

- Should be terminated into 50 ohms to prevent ringing

Email from Dave L (27 Jan 17) The IRIG AM + and IRIG AM – input does not require a 50 ohm load. The input impedance is 10 K ohms. Sharp edged signals like IRIG DCLS and 1PPS are affected by ringing so they generally need to have a terminator.

Use of Fiber converter into the IRIG DCLS input of a TSync board (DCLS input requires input termination)

Note: Email below was specifically referring to Model 2817-R-SM Fiber to Copper converter

Our Engineering team has reviewed your intended custom TSync-PCIE board cable, and has some feedback/recommend changes for you:

To begin and just to clarify: As the IRIG AM signal cannot go through the fiber converters, an IRIG DCLS signal will need to be provided (in place of the IRIG AM signal). So in case you weren’t already aware of this, and were already intending to provide IRIG DCLS from the IRIG Source, this IRIG DCLS signal needs to be connected to the IRIG DCLS pins of the TSync board’s Timing connector. With a single-ended IRIG DCLS signal being provided (instead of a differential signal) this signal should be connected to the “IRIG DLCS +” input pin and to ground (leaving the “IRIG DCLS –“ input pin of the Timing connector floating-it won’t have a connection to your cable).

As for the General Purpose signal being provided to the GPI input of the Timing connector, this should not only consist of the signal being provided to the GPI input pin. There should also be a ground connection on both sides of the custom cable for GPI input, as well.   In addition, if the source of the event (GPI) input to the Timing board is expecting a 50 ohm termination (instead of a high impedance termination), an external 50 ohm terminator will need to be obtained/placed at the TSync board’s end of this line). Note that https://www.pasternack.com/default.aspx? is a company that offers 50 ohm terminators, if you need one.

Here is a summary table for the cable pin-out:

Signal	**FROM**	**TO**
IRIG DCLS+	P1-25	P3-Core
GND	P1-23	P3-Shield
GPI 0	P1-6	P2-1
GND	P1-5	P2-2

Associated earlier email from Dave Sohn (16 Dec 16) They can't run IRIG AM through the fiber converter. I am assuming then that they are running IRIG DCLS. That means they need to connect into the IRIG DCLS inputs. As they don't really have a differential signal, they need to connect into the positive side input only, and ground. Leave the negative side floating. They will also need to add 50 ohm termination on the IRIG signal from the fiber converter.

They should be connecting the General Purpose signal to the GPI input, but also connecting a ground. If the signal is meant to terminate into 50 ohms, an external terminator will need to be provided as well.

P/N 1191-0000-0710 (BNC T connector and 50 ohm terminator for 1PPS input)

- (**in Arena**): https://app.bom.com/supplier-items/detail-spec?item_id=1218663225
- (**In Salesforce**): https://orolia.my.salesforce.com/01t1A000004SUKm?srPos=2&srKp=01t
  - ➢ One is included with each TSync board
  - ➢ Note this BNC terminator won't work for IRIG DCLS input, because IRIG DCLS input is via a **DB9** connector on breakout cable (not a BNC connector).

Default IRIG input configuration:

- ➢ Auto-detects (after changing from "manual" to "auto-detect" mode) either **IRIG B122** or **IRIG B123**

Modulation is automatically detected because there are two distinct input connectors- one for AM and a separate one for DCLS. Auto-detects IRIG Mode (B122, B123, etc). Does not auto-detect the Control field information (such as the 1344 extensions, year info). This needs to be defined with API call/example program.

According to Paul Myers, auto detect is not very robust. It's better to manually configure the input configurations using the API calls or example programs, especially if it's not auto-detecting the input.

Accuracy to IRIG input reference:

Refer also to "**IRIG accuracy (for products such as TPRO/TSAT and TSync)**" in the custservice doc: I:\Customer Service\1- Cust Assist documents\CustomerServiceAssistance.pdf

- ➢ **IRIG accuracy to UTC**: Consists of the synchronization to the IRIG source PLUS the stability/accuracy of the IRIG source.
- ➢ Refer to the "**IRIG input accuracy**" section of the Customerserviceassistance document for additional info on IRIG sync accuracies (such as AM versus DCLS for example)
  - In summary of the Customerserviceassistance document:

To calculate the accuracy of IRIG input to UTC:

1) The IRIG generator needs to be synced to an external reference (such as GPS).
2) You need to add the vendor's accuracy spec of its IRIG output to the typical IRIG input accuracies of:
   - **IRIG DCLS input:** +/- 100 nanoseconds
   - **IRIG AM input :** +/- 20 to 200 microseconds

Email from Keith (10 Feb 17) To begin, a TSync board using an IRIG AM input for its 1PPS input reference is inherently going to see less than optimal operation (as a factor of the slow rise-time of the AM signal). For optimal timing, either an IRIG DCLS input signal or a very stable external 1PPS input signal from a disciplined reference should instead always be used.

The typical accuracy of a TSync board syncing via IRIG AM input is anywhere between +/- 20 and +/- 200 microseconds, due to the slow rise-time of the AM signal. In comparison, this typical accuracy drops to +/- 100 nanoseconds when using an IRIG DCLS signal.

Due to the slow rise-tine, there will likely be a fairly large initial 1PPS phase error (offset) between the IRIG AM source and the TSync board after each power-up (and this value will likely vary widely after each individual power-up of the board). Thereafter, the TSync board's oscillator disciplining will be continuously "chasing" the jitter of the input reference (the oscillator of the TSync board helps dampen out this jitter, because its oscillator doesn't jump into alignment after the initial alignment. Instead, the oscillator is VERY slowly slewed to try to remain aligned with its 1PPS reference.

If the 1PPS reference needs to switch from an external 1PPS input to an IRIG AM input anytime after initial sync, the oscillator will need to very slowly slew back into alignment with the newly selected 1PPS reference (it doesn't jump into alignment with its newly selected reference). Due to the jitter of the IRIG AM input, the phase errors will remain fairly high until its fairly closely aligned to the 1PPS reference. Then it will be slowly "chasing" the IRIG AM input's jitter to try to stay aligned to it..

A TSync board that is synced with an external 1PPS which is being generated/outputted from a disciplined/synced reference, and that is properly terminated at the end of the signal, can be much more optimally synced to its reference. And its 1PPS phase error can remain much lower, because the oscillator is no longer needing to "chase" its 1PPS reference.

The most common issue we see when customers use an external 1PPS input ("EPP0") as the 1PPS input reference is due to ringing of the signal because of improper termination. The input impendence of the 1PPS signal is high impedance. However, the 1PPS source may desire a 50 ohm termination into the TSync-PCIe-PCIe board (instead of high impedance). If the source desires a 50 ohm load termination, the input 1PPS may likely "ring" inside of the TSync-PCIe board. The ringing of the 1PPS signal prevents a "crisp" point at which the TSync-PCIe board can trigger on. With ringing of the signal present, the detected trigger point may coincidentally be at the right point, but it could also trigger too early or too late. So, with each 1PPS input when the signal is ringing, it may trigger at a different point on the signal each time. This prevents the disciplining circuit from being able to receive a stable 1PPS that is can accurately discipline to, and so the TFOM values will also be much higher than expected.

We have seen a few other similar cases where an external 1PPS input was causing a jittery 1PPS input because of improper termination of the signal at the input of the board. These were all resolved with a 50 ohm input being applied. To see this affect, if you have access to an oscilloscope, input the 1PPS signal from the source with a high input impedance setting on the scope. Look at the rising edge. You will likely see a ringing of the signal with no defined rising edge. Then, switch the scope to 50 ohm termination and you will likely notice the signal a become a crisp 1PPS with a very defined rising edge. This is what the board is also likely receiving and seeing, as well. To prevent the ringing of the 1PPS input from occurring and affecting the TSync-PCIe-PCIe board, terminate the 1PPS input from the source with a 50 ohm load resistor. This will often "clean up the signal" so that it can be a better reference into the board.

Note that various cable lengths with the wrong termination at the end can have differing effects on the ringing of the 1PPS input to the boards. Each TSync board includes a 50 ohm load and BNC connector (in a bag labeled with the Spectracom P/N of 1191-0000-0710. If the 1PPS line is currently unterminated, I highly recommend adding this terminator to the end of the line.

Let me know if adding the 50 ohm load to the end of the line allows the 1PPS phase error values to decrease.

Regarding the TSync-PCIe timing boards, we intentionally don't specify the IRIG input synchronization accuracy. The IRIG input synchronization capabilities are going to vary, based on a few factors, including the stability of the IRIG input signal (coming from the external IRIG source) and whether IRIG Amplitude Modulation (AM) or Phase Modulation (TTL) is being provided to the timing board.

The more stable the IRIG input signal (i.e. an IRIG signal being provided by an IRIG generator that is externally synchronized to an external reference such as GPS will be more stable than a free-running IRIG source), the better the TSync-PCIe board can synchronize to its IRIG input. If there is jitter on the IRIG reference being applied to the timing board, the TSync-PCIe's oscillator is going to have difficulty keeping aligned with the input. But, if the IRIG signal is very stable, the oscillator can easily remain disciplined to its input.

Along the same lines, an IRIG DCLS input signal has a very fast rise-time (so it has a more "crisp" on-time point), leading to a more accurate synchronization, while an IRIG AM signal has a much slower rise time, which can affect the "trigger point" to derive on on-time point from an Amplitude Modulated sine wave. Because of the slow rise-time of IRIG AM, the exact trigger point of the sine-wave may vary. This is why an IRIG TTL signal results in better synchronization than an IRIG AM signal.

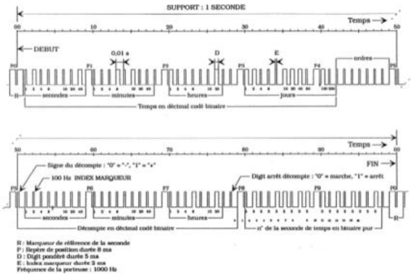
“Pseudo IRIG-B” timecode- IRIG plus Countdown (Count-down) / IRIG with Countdown and Countup (count-up)

- **Definition of Pseudo**: fake, false, mock, bogus (in this case, a non-standard, modified signal).
- We have seen two different scenarios from Spectracom France for support of count-down with the IRIG input
  - The first one was just **IRIG Countdown embedded** into the IRIG signal (not countdown and launch). Though not confirmed with our testing, engineering suspects this scenario should work OK, if the board is configured for IRIG B002 or B122 input to mask the countdown data). Refer to “A” below.
  - The second one was regarding **Countdown** and **Launch** as **Pseudo IRIG** timecodes. Per Dave Sohn, we don’t support this functionality (as of at least Nov 2015 and doubt we ever will). Refer to “B” below.

A) First scenario (countdown only- embedded into the IRIG time code)

Q (two emails From Sylvain) Our integrator partner ask us if the Tsync card IRIG Input is able to decode the following IRIG signal:

It is a pseudo IRIG B signal distributing the normal time reference (called TU) placed on P0,P1 …. Of the Frame + a countdown signal (called TD “temps Décompte”) placed in P5,P6,P7,P8,P9 of the frame. See below the detail of the frame:


SUPPORT : 1 SECONDE
Temps →
00
10
20
30
40
50
DEBUT
0,01 s
D
E
ordres
P0
P1
P2
P3
P4
P5
R
secondes
minutes
heures
jours
Temps en décimal codé binaire
Temps →
50
60
70
80
90
00
FIN →
Signe du décompte : "0" = "-", "1" = "+"
100 Hz INDEX MARQUEUR
Digit arrêt décompte : "0" = marche, "1" = arrêt
P6
P7
P8
P9
P0
secondes
minutes
heures
Décompte en décimal codé binaire
n° de la seconde de temps en binaire pur
R : Marqueur de référence de la seconde
P : Repère de position durée 8 ms
D : Digit pondéré durée 5 ms
E : Index marqueur durée 2 ms
Fréquence de la porteuse : 1000 Hz

They want to know if the Tsync will get the TU time without any disturb caused by the countdown message? The goal is to decode the IRIG B signal as it was a normal standard frame and don’t care about the additional frame words.

Can you confirm that it can work without any additional development?
Let me clarify my request please:

The signal input will be TU + TD (TU is the standard IRIG format (goes up) and TD is the countdown).
Normally, it shouldn’t be an issue caused by :”with the input time counting down, instead of up”. Because With this such Frame, the IRIG signal standard is still going up (because the TU is standard).
This is the TD which decreases and it is provided on other part of the FRAME.

So my hypothesis is: if the Tsync do a mask to get the time only on P0-P5 (as it should normally be) : it will work because the Tsync won’t use P6-P9.
So I think the Tsync should work. GOOD NEWS so!
But if the mask is not done. The data for TD may disturb the signal input integrity …. And so : Bad news.

Per Dave Sohn I haven’t seen this before, but it might work using IRIG B002 or B122 formats which only look at the ToY in P0 – P4.

Per Paul Myers to Tim Tetreault Our IRIG Code can be configured to use only portions of the sub-frames as David Sohn says. If the Downcounting part is compatible with IRIG sub-framing and our FPGA decoding is not interfered with the data will be clocked in. You then can configure as David Sohn says to try to only read P0-P4. As long as the other fields are not read they will be ignored. I don't think any IRIG data checking will come into play as long as they are not read.

Q Can they demo a TSYNC card?

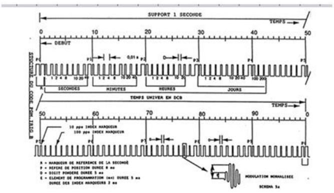
A Reply from Tim Tetreault to Sylvain (5 Aug 2014) After talking to some of the engineers, they think it might work if the IRIG is setup to use IRIG B002 or B122 formats which only look at the ToY in P0 – P4. But without having something to test with, we cannot be sure.

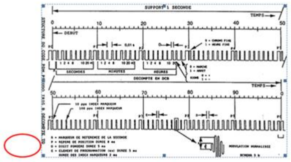
I would recommend that the customer demo a unit and try it out.

B) Second scenario- Pseudo IRIG-B (Countdown and launch as pseudo IRIG data streams)

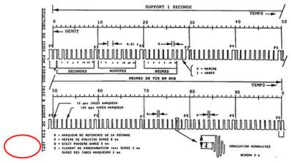
- *From: http://timefreq.com/wp-content/uploads/2013/08/Download-M355-11-Digit-LED-Display-Datasheet.pdf*

Pseudo IRIG B does not contain the time-of-day but a countdown / countup time. Both countdown and countup times are incrementing in the same direction but countdown time is displayed as a negative quantity which gets smaller as it approaches zero.





Pseudo IRIG Count-down



Pseudo IRIG Launch

IRIG input pin-outs

There are separate IRIG inputs on the breakout cables for either IRIG AM input or DCLS input. Refer to: Breakout cables for the “Timing” connector.

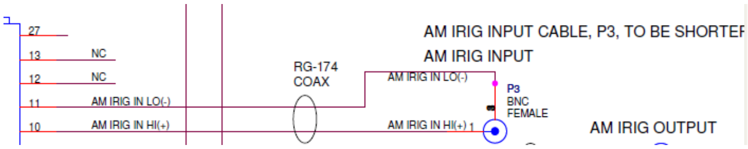
Note: If the Premium/Standard breakout cable assembly was received after the end of 2013, a short adapter cable needs to be installed between the Timing connector on the board and the DB25 connector on the Premium/Standard breakout cable. The breakout cable can’t connect directly to the TSync board with changes that were incorporated to both type breakout cables back.

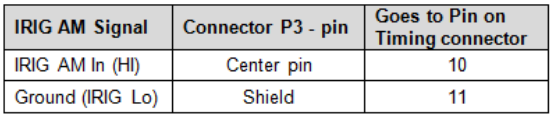
Refer to the breakout cable section for more details: Adapter cable (for the newer DB25 connector on standard and premium breakout cables)

A) Dedicated IRIG AM input (“ira0” reference)

- Can be single-ended or differential input
- IRIG input is via one of the three methods:
  - a BNC connector on either the Standard Breakout cable (refer to “1” below) Or
  - Premium breakout cable (refer to “2” below)”. Or,
  - Can bypass breakout cable altogether and connect directly to the Timing connector.

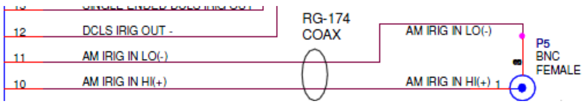
1. Using a Standard Breakout cable (BNC Connector P3)

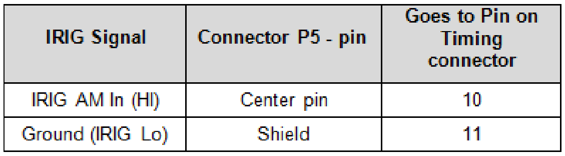




IRIG AM Signal	Connector P3 - pin	Goes to Pin on Timing connector
IRIG AM In (HI)	Center pin	10
Ground (IRIG Lo)	Shield	11

2. Using a Premium Breakout cable (BNC Connector P5)





IRIG Signal	Connector P5 - pin	Goes to Pin on Timing connector
IRIG AM In (HI)	Center pin	10
Ground (IRIG Lo)	Shield	11

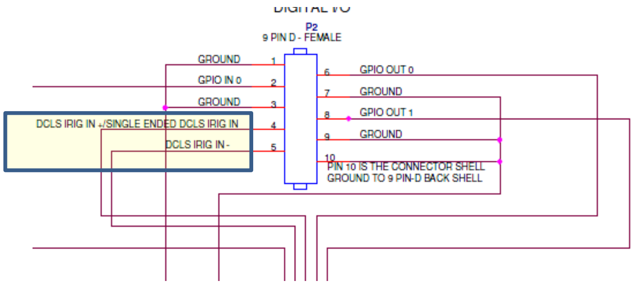
B) Dedicated IRIG DCLS input ("ird0" reference)

- IRIG DCLS input is via the "D connector" (P2)
- Can be single-ended or differential input

Connection to Standard or Premium Breakout cables

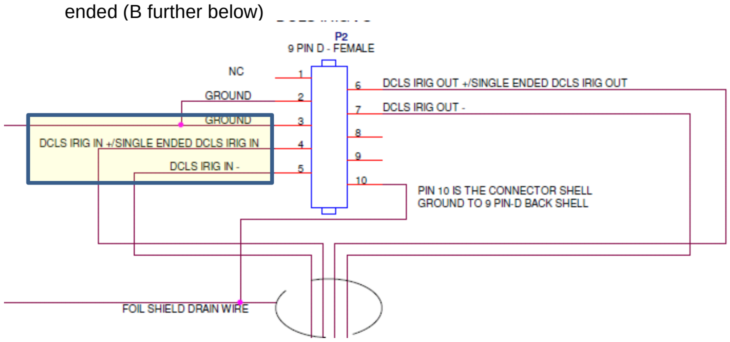
1. Using a Standard Breakout cable

Note: The actual pin-out depends on whether the IRIG input is balanced (A further below) or single-ended (B further below)



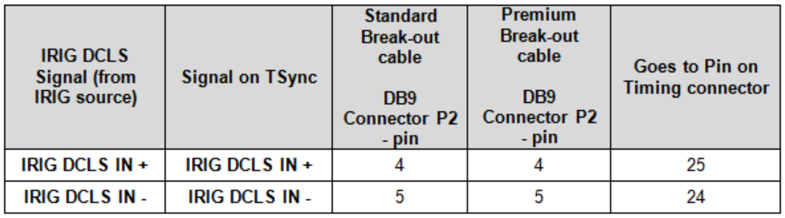
2. Using a Premium Breakout cable

Note: The actual pin-out depends on whether the IRIG input is balanced (A further below) or single-ended (B further below)


P2
9 PIN D - FEMALE
NC
GROUND
GROUND
DCLS IRIG IN +/SINGLE ENDED DCLS IRIG IN
DCLS IRIG IN -
1
2
3
4
5
6
7
8
9
10
DCLS IRIG OUT +/SINGLE ENDED DCLS IRIG OUT
DCLS IRIG OUT -
PIN 10 IS THE CONNECTOR SHELL
GROUND TO 9 PIN-D BACK SHELL
FOIL SHIELD DRAIN WIRE

C) Balanced DCLS IRIG input (two signals and ground)

Please make sure you are connected to the DCLS Input (pins 4 and 5 of the DB9 connector on the breakout cable) if using IRIG DCLS. **Also if the DCLS is single ended instead of a balanced two wire input just tie the + to the DCLS Input + and the – to Ground.** The input should auto-sense the format so if it is connected OK the board should sync after about 10 seconds.



IRIG DCLS Signal (from IRIG source)	Signal on TSync	Standard Break-out cable DB9 Connector P2 - pin	Premium Break-out cable DB9 Connector P2 - pin	Goes to Pin on Timing connector
IRIG DCLS IN +	IRIG DCLS IN +	4	4	25
IRIG DCLS IN -	IRIG DCLS IN -	5	5	24

Note: If the Premium/Standard breakout cable assembly was received after the end of 2013, a short adapter cable needs to be installed between the Timing connector on the board and the DB25 connector on the Premium/Standard breakout cable. The breakout cable can't connect directly to the TSync board with changes that were incorporated to both type breakout cables back.

Refer to the breakout cable section for more details: Adapter cable (for the newer DB25 connector on standard and premium breakout cables)

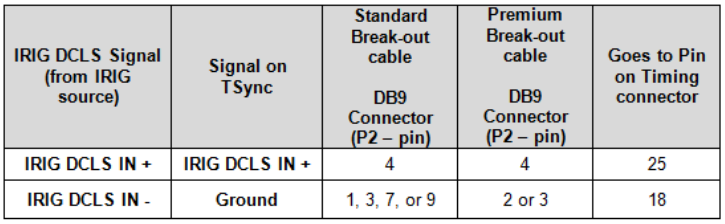
D) Single-ended DCLS input

For IRIG DCLS single ended input:

A. The "**IRIG DCLS In +"** signal from the IRIG generator should be connected to the "**IRIG DCLS Input +"** pin.

B. The **IRIG "DCLS IN –"**signal from the IRIG generator should be connected to one of the **ground** pins on the breakout cable.

C. The "**IRIG DCLS Input –"** pin should be left unconnected on the TSync-PCIe side



IRIG DCLS Signal (from IRIG source)	Signal on TSync	Standard Break-out cable DB9 Connector (P2 – pin)	Premium Break-out cable DB9 Connector (P2 – pin)	Goes to Pin on Timing connector
IRIG DCLS IN +	IRIG DCLS IN +	4	4	25
IRIG DCLS IN -	Ground	1, 3, 7, or 9	2 or 3	18

Email from Dave L (16 Sept 17) Single-ended DCLS input does not connect the GND to the (-) Input.

For IRIG DCLS single ended input:

A. The "IRIG DCLS In +" signal from the IRIG generator should be connected to the "IRIG DCLS Input +" pin.

B. The IRIG "DCLS IN –"signal from the IRIG generator should be connected to one of the ground pins onthe breakout cable.

C. The “IRIG DCLS Input –“ pin should be left unconnected on the TSync-PCIe side

This is something we overlooked at the beginning. Please try this and let me know how it works.

Note: If the breakout cable assembly was received after the end of 2013, a short adapter cable needs to be installed between the Timing connector on the board and the DB25 connector on the Premium/Standard breakout cable. The breakout cable can’t connect directly to the TSync board with changes that were incorporated to both type breakout cables back.

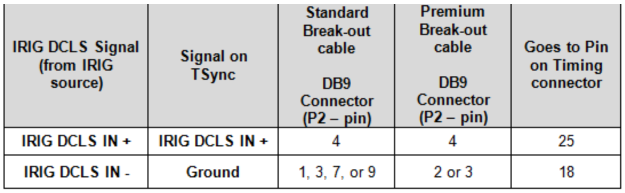
If it will not sync, the problem could be with the format of the data in the signal. Can you verify what the IRIG Format is coming from the Symmetricom IRIG source? It must match the IRIG Input requirements listed in the Tsync manual. Sometimes the IRIG signal is a non-standard format so please check this.

IRIG DCLS being provided by a SecureSync (via an IRIG output Option Card)

- ➢ IRIG output will be a BNC or terminal block connector the IRIG Output Option card
- **Models 1204-05/1204-15** provides **Single-ended** IRIG via BNC
- **Model 1204-22** provides **differential** IRIG via terminal block
- **Models 1204-1E/1204-27** are Fiber connectors

E) IRIG Option Card Models 1204-05/1204-15 (IRIG Single-ended output)

- ➢ **BNC** connector of Option Card goes to **DB9** connector on Standard/Premium Breakout cable
- IRIG DCLS output signal (center conductor of cable) goes to +DCLS on DB9 (**pin 4)** of breakout
- Shield of cable goes to “**ground**” on DB9 connector of breakout cable



IRIG DCLS Signal (from IRIG source)	Signal on TSync	Standard Break-out cable DB9 Connector (P2 – pin)	Premium Break-out cable DB9 Connector (P2 – pin)	Goes to Pin on Timing connector
IRIG DCLS IN +	IRIG DCLS IN +	4	4	25
IRIG DCLS IN -	Ground	1, 3, 7, or 9	2 or 3	18

F) IRIG Option Card Model 1204-22 (IRIG differential output)

- **IRIG DCLS +** goes to **IRIG DCLS +**
- **IRIG DCLS -** goes to **IRIG DCLS -**

IRIG input configuration

Note: In the TSync-PCIe drivers, the IRIG input is configured using the ‘IR” commands

IR command syntax:

- Use the <Index> value of **0** for **DCLS** in/out ports.
- Use the <Index> value of **1** for **AM** in/out ports.

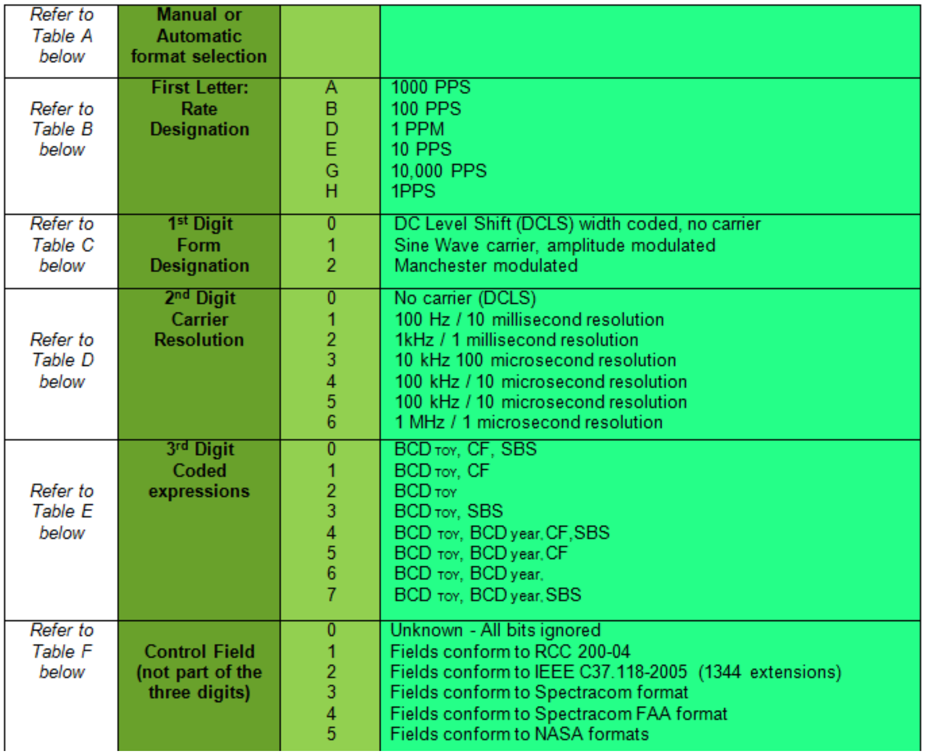
Email from Dave Sohn:
There are two IRIG inputs on the TSync. One is AM and the other is DCLS. (There are a total of two instances, 0 and 1). They can see this by the return of "**IR_GetMod 0 0**" and "**IR_GetMod 0 1**".

O=DCLS
1=AM

Available Tsync IRIG Input / Output Formats:

RIG Code Format Provided	Code Description
A000	IRIG A, DCLS, BCD, CF, SBS
A001	IRIG A, DCLS, BCD, CF
A002	IRIG A, DCLS, BCD
A003	IRIG A, DCLS, BCD, SBS
A004	IRIG A, DCLS, BCD$_{TOY}$, BCD$_{YEAR}$, CF, SBS
A005	IRIG A, DCLS, BCD$_{TOY}$, BCD$_{YEAR}$, CF
A006	IRIG A, DCLS, BCD$_{TOY}$, BCD$_{YEAR}$
A007	IRIG A, DCLS, BCD$_{TOY}$, BCD$_{YEAR}$, SBS
A130	IRIG A, AM, 10kHz, BCD, CF, SBS
A131	IRIG A, AM, 10kHz, BCD, CF
A132	IRIG A, AM, 10kHz, BCD
A133	IRIG A, AM, 10kHz, BCD, SBS
A134	IRIG A, AM, 10kHz, BCD$_{TOY}$, BCD$_{YEAR}$, CF, SBS
A135	IRIG A, AM, 10kHz, BCD$_{TOY}$, BCD$_{YEAR}$, CF
A136	IRIG A, AM, 10kHz, BCD$_{TOY}$, BCD$_{YEAR}$
A137	IRIG A, AM, 10kHz, BCD$_{TOY}$, BCD$_{YEAR}$, SBS
B000	IRIG B, DCLS, BCD, CF, SBS
B001	IRIG B, DCLS, BCD, CF
B002	IRIG B, DCLS, BCD
B003	IRIG B, DCLS, BCD, SBS
B004	IRIG B, DCLS, BCD$_{TOY}$, BCD$_{YEAR}$, CF, SBS
B006?	IRIG B, DCLS, BCD$_{TOY}$, BCD$_{YEAR}$ (not yet confirmed this format is supported)
B120	IRIG B, AM, BCD, CF, SBS
B121	IRIG B, AM, BCD, CF
B122	IRIG B, AM, BCD
B123	IRIG B, AM, BCD, SBS
B124	IRIG B, AM, BCD$_{TOY}$, BCD$_{YEAR}$, CF, SBS
B125	IRIG B, AM, 1kHz, BCD$_{TOY}$, BCD$_{YEAR}$, CF
B126	IRIG B, AM, 1kHz, BCD$_{TOY}$, BCD$_{YEAR}$
B127	IRIG B, AM, 1kHz, BCD$_{TOY}$, BCD$_{YEAR}$, SBS
G001	IRIG G, DCLS, BCD, CF
G002	IRIG G, DCLS, BCD
G005	IRIG G, DCLS, BCD$_{TOY}$, BCD$_{YEAR}$, CF
G006	IRIG G, DCLS, BCD$_{TOY}$, BCD$_{YEAR}$
G141	IRIG G, AM, 100kHz, BCD,CF
G142	IRIG G, AM, 100kHz, BCD
G145	IRIG G, AM, 100kHz, BCD$_{TOY}$, BCD$_{YEAR}$, CF

IRIG Input configuration commands



Refer to Table A below	**Manual or Automatic format selection**		
Refer to Table B below	**First Letter: Rate Designation**	A B D E G H	1000 PPS 100 PPS 1 PPM 10 PPS 10,000 PPS 1PPS
Refer to Table C below	**1st Digit Form Designation**	0 1 2	DC Level Shift (DCLS) width coded, no carrier Sine Wave carrier, amplitude modulated Manchester modulated
Refer to Table D below	**2nd Digit Carrier Resolution**	0 1 2 3 4 5 6	No carrier (DCLS) 100 Hz / 10 millisecond resolution 1kHz / 1 millisecond resolution 10 kHz 100 microsecond resolution 100 kHz / 10 microsecond resolution 100 kHz / 10 microsecond resolution 1 MHz / 1 microsecond resolution
Refer to Table E below	**3rd Digit Coded expressions**	0 1 2 3 4 5 6 7	BCD$_{TOY}$, CF, SBS BCD$_{TOY}$, CF BCD$_{TOY}$ BCD$_{TOY}$, SBS BCD$_{TOY}$, BCD$_{year}$, CF,SBS BCD$_{TOY}$, BCD$_{year}$, CF BCD$_{TOY}$, BCD$_{year}$, BCD$_{TOY}$, BCD$_{year}$, SBS
Refer to Table F below	**Control Field (not part of the three digits)**	0 1 2 3 4 5	Unknown - All bits ignored Fields conform to RCC 200-04 Fields conform to IEEE C37.118-2005 (1344 extensions) Fields conform to Spectracom format Fields conform to Spectracom FAA format Fields conform to NASA formats

IR_GetMode/IR_SetMode

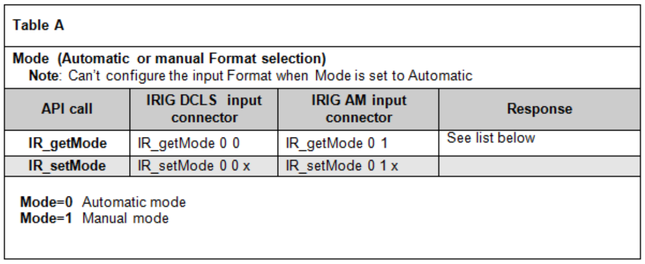


Table A			
Mode (Automatic or manual Format selection) **Note**: Can't configure the input Format when Mode is set to Automatic			
API call	**IRIG DCLS input connector**	**IRIG AM input connector**	**Response**
IR_getMode	IR_getMode 0 0	IR_getMode 0 1	See list below
IR_setMode	IR_setMode 0 0 x	IR_setMode 0 1 x	
Mode=0 Automatic mode **Mode=1** Manual mode			

IR_GetFormat/IR_SetFormat

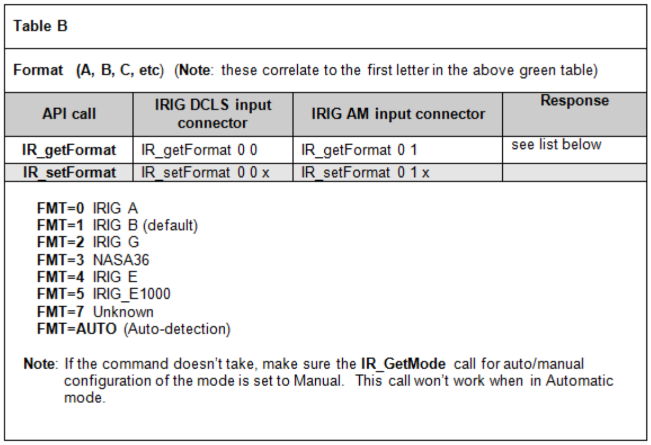


Table B

Format (A, B, C, etc) (**Note**: these correlate to the first letter in the above green table)

API call	IRIG DCLS input connector	IRIG AM input connector	Response
IR_getFormat	IR_getFormat 0 0	IR_getFormat 0 1	see list below
IR_setFormat	IR_setFormat 0 0 x	IR_setFormat 0 1 x	

FMT=0 IRIG A
FMT=1 IRIG B (default)
FMT=2 IRIG G
FMT=3 NASA36
FMT=4 IRIG E
FMT=5 IRIG_E1000
FMT=7 Unknown
FMT=AUTO (Auto-detection)

Note: If the command doesn't take, make sure the **IR_GetMode** call for auto/manual configuration of the mode is set to Manual. This call won't work when in Automatic mode.

IR_GetMod/IR_SetMod

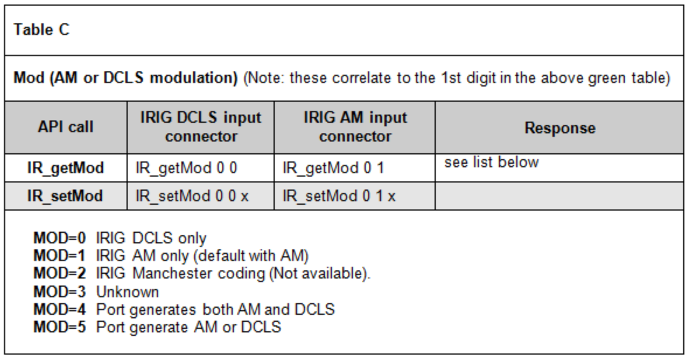


Table C

Mod (AM or DCLS modulation) (Note: these correlate to the 1st digit in the above green table)

API call	IRIG DCLS input connector	IRIG AM input connector	Response
IR_getMod	IR_getMod 0 0	IR_getMod 0 1	see list below
IR_setMod	IR_setMod 0 0 x	IR_setMod 0 1 x	

MOD=0 IRIG DCLS only
MOD=1 IRIG AM only (default with AM)
MOD=2 IRIG Manchester coding (Not available).
MOD=3 Unknown
MOD=4 Port generates both AM and DCLS
MOD=5 Port generate AM or DCLS

IR_GetFreq/IR_SetFreq

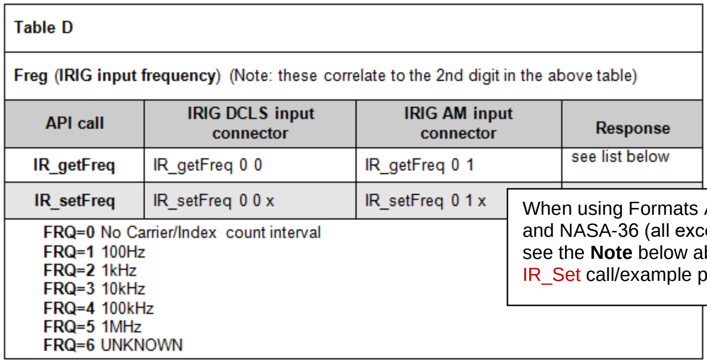


Table D

Freg (**IRIG input frequency**) (Note: these correlate to the 2nd digit in the above table)

API call	**IRIG DCLS input connector**	**IRIG AM input connector**	**Response**
IR_getFreq	IR_getFreq 0 0	IR_getFreq 0 1	see list below
IR_setFreq	IR_setFreq 0 0 x	IR_setFreq 0 1 x	

FRQ=0 No Carrier/Index count interval
FRQ=1 100Hz
FRQ=2 1kHz
FRQ=3 10kHz
FRQ=4 100kHz
FRQ=5 1MHz
FRQ=6 UNKNOWN

When using Formats A, B, G and NASA-36 (all except "E"), see the **Note** below about this IR_Set call/example program.

Note for IR_SetFreq: The Frequency field is tied to the configured Format value (A, B, G, etc). If a Format which only supports one particular Frequency (such as Format 'G' only supports '100 kHz') is selected, the **IR_SetFreq** command will respond with: Error: opt err ("RC_STUB_FUNC) **even when using the correct values**. The Frequency field will be set to '100 kHz' automatically when "G' has been set . FYI- The correct frequency value being automatically selected can be verified by performing an IR_GetFreq 0 0 (for DCLS input) or IR_GetFreq 0 1 (for AM input)

The one Format which does allow the Frequency field to be configured is **IRIG E AM** (which can be set to either 100 Hz, or 1000 Hz). The frequency field defaults to **'(1) 100 Hz'** for IRIG E AM, so the IR_SetFreq command only needs to be performed if its desired to use IRIG E at **1000 Hz** instead of 100Hz).

IR_GetCtrlField/ IR_SetCtrlField

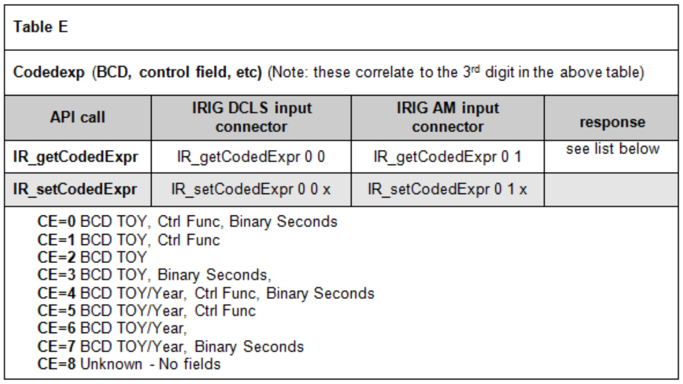


Table E

Codedexp (**BCD, control field, etc**) (Note: these correlate to the 3rd digit in the above table)

API call	**IRIG DCLS input connector**	**IRIG AM input connector**	**response**
IR_getCodedExpr	IR_getCodedExpr 0 0	IR_getCodedExpr 0 1	see list below
IR_setCodedExpr	IR_setCodedExpr 0 0 x	IR_setCodedExpr 0 1 x	

CE=0 BCD TOY, Ctrl Func, Binary Seconds
CE=1 BCD TOY, Ctrl Func
CE=2 BCD TOY
CE=3 BCD TOY, Binary Seconds,
CE=4 BCD TOY/Year, Ctrl Func, Binary Seconds
CE=5 BCD TOY/Year, Ctrl Func
CE=6 BCD TOY/Year,
CE=7 BCD TOY/Year, Binary Seconds
CE=8 Unknown - No fields

IR_GetCtrlField/ IR_SetCtrlField

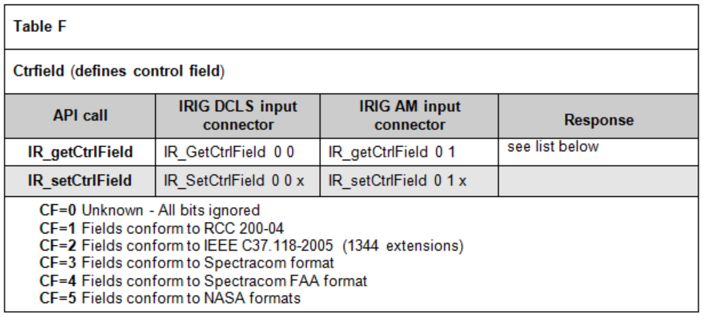


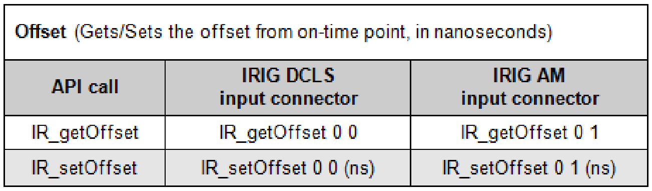
Table F

Ctrfield (defines control field)

API call	IRIG DCLS input connector	IRIG AM input connector	Response
IR_getCtrlField	IR_GetCtrlField 0 0	IR_getCtrlField 0 1	see list below
IR_setCtrlField	IR_SetCtrlField 0 0 x	IR_setCtrlField 0 1 x	

CF=0 Unknown - All bits ignored
CF=1 Fields conform to RCC 200-04
CF=2 Fields conform to IEEE C37.118-2005 (1344 extensions)
CF=3 Fields conform to Spectracom format
CF=4 Fields conform to Spectracom FAA format
CF=5 Fields conform to NASA formats

IR_GetOffset/ IR_SetOffset

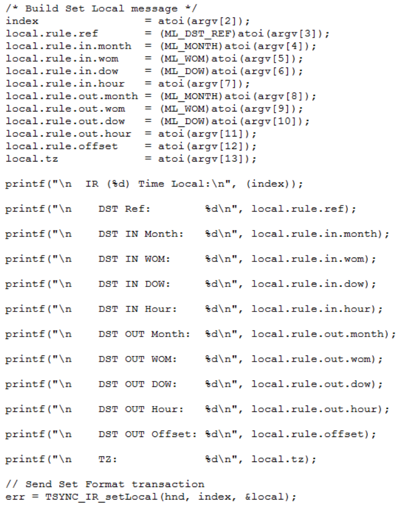


Offset (Gets/Sets the offset from on-time point, in nanoseconds)

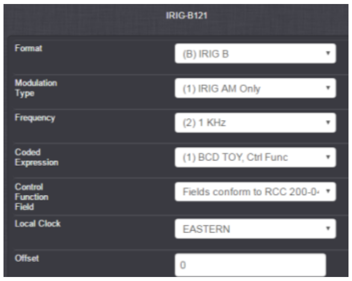
API call	IRIG DCLS input connector	IRIG AM input connector
IR_getOffset	IR_getOffset 0 0	IR_getOffset 0 1
IR_setOffset	IR_setOffset 0 0 (ns)	IR_setOffset 0 1 (ns)

IRIG generator providing local time input (“Local clock” via IR_SetLocal)

- When the input from the IRIG generator is providing the board with local time, the TSync board needs to be programmed with how much time offset is being provided, so that it can internally convert the local time to UTC timescale. Note the timezone and DST offsets are configured in seconds.



```
/* Build Set Local message */
index                 = atoi(argv[2]);
local.rule.ref        = (ML_DST_REF)atoi(argv[3]);
local.rule.in.month   = (ML_MONTH)atoi(argv[4]);
local.rule.in.wom     = (ML_WOM)atoi(argv[5]);
local.rule.in.dow     = (ML_DOW)atoi(argv[6]);
local.rule.in.hour    = atoi(argv[7]);
local.rule.out.month  = (ML_MONTH)atoi(argv[8]);
local.rule.out.wom    = (ML_WOM)atoi(argv[9]);
local.rule.out.dow    = (ML_DOW)atoi(argv[10]);
local.rule.out.hour   = atoi(argv[11]);
local.rule.offset     = atoi(argv[12]);
local.tz              = atoi(argv[13]);

printf("\n  IR (%d) Time Local:\n", (index));

printf("\n    DST Ref:        %d\n", local.rule.ref);

printf("\n    DST IN Month:   %d\n", local.rule.in.month);

printf("\n    DST IN WOM:     %d\n", local.rule.in.wom);

printf("\n    DST IN DOW:     %d\n", local.rule.in.dow);

printf("\n    DST IN Hour:    %d\n", local.rule.in.hour);

printf("\n    DST OUT Month:  %d\n", local.rule.out.month);

printf("\n    DST OUT WOM:    %d\n", local.rule.out.wom);

printf("\n    DST OUT DOW:    %d\n", local.rule.out.dow);

printf("\n    DST OUT Hour:   %d\n", local.rule.out.hour);

printf("\n    DST OUT Offset: %d\n", local.rule.offset);

printf("\n    TZ:             %d\n", local.tz);

// Send Set Format transaction
err = TSYNC_IR_setLocal(hnd, index, &local);
```



IRIG-B121

Format	(B) IRIG B
Modulation Type	(1) IRIG AM Only
Frequency	(2) 1 KHz
Coded Expression	(1) BCD TOY, Ctrl Func
Control Function Field	Fields conform to RCC 200-0
Local Clock	EASTERN
Offset	0

From associated ktsif file for the above browser config

4,5,1,42,0,0,0,IR_SetLocal 0 0 0 3 2 0 2 11 1 0 2 3600 -18000
~~4,5,1,42,0,0,0,IR_SetOffset 0 0 0~~
~~4,5,1,42,0,0,0,IR_SetTimeScale 0 0 3~~
~~4,5,1,42,0,0,0,IR_SetFormat 0 0 1~~
~~4,5,1,42,0,0,0,IR_SetCodedExp 0 0 1~~
~~4,5,1,42,0,0,0,IR_SetCtrlField 0 0 1~~

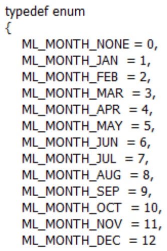
IR_SetTimeScale 0 3 (where “3” is for **Local** timescale)

IR_SetLocal <device index> <index> <ref> <in.mon> <in.wom> <in.dow> <in.hr> <out.mon> <out.wom> <out.dow> <out.hr> <offset> <tz>

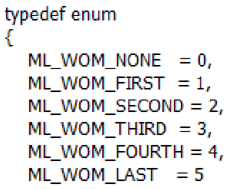
From example screenshot above:

IR_SetLocal 0 0	0	3	2	0	2	11	1	0	2	3600	18000
	<ref>	<in.mon>	<in.wom>	<in.dow>	<in.hr>	<out.mon>	<out.wom>	<out.dow>	<out.hr>	<offset>	<tz>
	(UTC)	(Mar)	(2nd)	(Sun)	(2AM)	(Nov)	(First)	(Sunday)	(2AM)	(1 hour)	(5 hours)

Where



```
typedef enum
{
   ML_MONTH_NONE = 0,
   ML_MONTH_JAN  = 1,
   ML_MONTH_FEB  = 2,
   ML_MONTH_MAR  = 3,
   ML_MONTH_APR  = 4,
   ML_MONTH_MAY  = 5,
   ML_MONTH_JUN  = 6,
   ML_MONTH_JUL  = 7,
   ML_MONTH_AUG  = 8,
   ML_MONTH_SEP  = 9,
   ML_MONTH_OCT  = 10,
   ML_MONTH_NOV  = 11,
   ML_MONTH_DEC  = 12
```



```
typedef enum
{
   ML_WOM_NONE   = 0,
   ML_WOM_FIRST  = 1,
   ML_WOM_SECOND = 2,
   ML_WOM_THIRD  = 3,
   ML_WOM_FOURTH = 4,
   ML_WOM_LAST   = 5
```

<in.mon> and **<out.mon> (Month) use 3 for March and 11 for November**

<in.wom> and **<out.wom> (week) use 2 for Second and 0 for First**

<in.dow> and **<out.dow> (Day of week) use 3600 (for 1 hour)**

<offset> (amount of change) use 3600 (for 1 hour)

<tz> = Standard time offset in seconds (multiply offset x 3600**)**

- **Eastern (5 hrs)** = use 18000
- **Central (6 hrs)** = use 21600
- **Mountain (7 hrs)** = use 25200
- **Pacific (8 hrs)** = use 28800

Referring to salesforce case 24962, email Keith sent (6 Apr 17) Thanks for your reply and for confirming the board is syncing via IRIG. The IRIG generator outputting local time would certainly explain the operation you are observing.

When providing IRIG input with the time being provided in local time, the two API call/example programs that need to be performed at each boot-up of the board to allow the board to convert the local time input to UTC is as follows:

IR_SetTimeScale 0 0 3 (sets the timescale of the input to local)

With the IRIG generator providing "Eastern Time Zone, with auto DST correction being applied, the call is:
IR_SetLocal 0 0 0 3 2 0 2 11 1 0 2 3600 -18000

Note the full syntax of this call above, which configures the time zone offset and DST rules, is as follows:
IR_SetLocal <device index> <index> <ref> <in.mon> <in.wom> <in.dow> <in.hr> <out.mon> <out.wom> <out.dow> <out.hr> <offset> <tz>

There is no need to reboot the board for these settings to become effective.

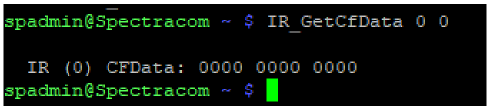
Please let me know that you start seeing UTC time when performing the HW_GetTime calls after running these two calls.

Reading raw IRIG input signal data (IR_GetMessage/IR_GetCfData)

IR_GetCFData

- ➢ Available API call/example program used to demodulate/display the raw data in the IRIG input CF section message provided to the TSync board.
- ➢ Data is reported when the IRIG signal is present.
- ➢ IRIG input can be disabled in the Reference Priority table, with this data still being reported (IRIG input doesn't have to be syncing the board in order to be reported)

Interpreting the data reported in the IR_GetCfData response

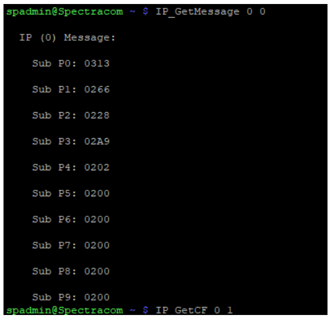


```
spadmin@Spectracom ~ $ IR_GetCfData 0 0

  IR (0) CFData: 0000 0000 0000
spadmin@Spectracom ~ $ 
```

- ➢ The response reports ??? as shown
- ➢ Need to convert hex to ?? (see Apps team/Ron Dries)

IR_GetMessage

Note the example below is from an IRIG output (**IP**_GetMessage 0 0) instead of IRIG input, but format is the same



```
spadmin@Spectracom ~ $ IP_GetMessage 0 0

  IP (0) Message:

    Sub P0: 0313

    Sub P1: 0266

    Sub P2: 0228

    Sub P3: 02A9

    Sub P4: 0202

    Sub P5: 0200

    Sub P6: 0200

    Sub P7: 0200

    Sub P8: 0200

    Sub P9: 0200
spadmin@Spectracom ~ $ IP GetCF 0 1
```

- ➢ Available API call/example program used to demodulate/display the raw data in the IRIG input message provided to the TSync board
- ➢ Data is reported when the IRIG signal is present.
- ➢ IRIG input can be disabled in the Reference Priority table, with this data still being reported (IRIG input doesn't have to be syncing the board in order to be reported)

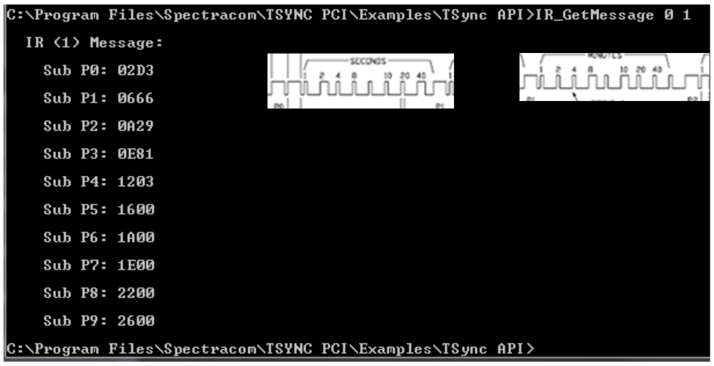
Table G			
Message (AM or DCLS input Message)			
API call	**IRIG DCLS output connector**	**IRIG AM output connector**	**response**
IR_GetMessage	**IR_GetMessage 0 0**	**IR_GetMessage 0 1**	see list below
IR_SetMessage	IR_SetMessage 0 0 (Sub P0 –Sub P9)	**IR_SetMessage 0 1 (Sub P0 –Sub P9)**	
(Sub P0 –Sub P9)			

Interpreting the data reported in the IR_GetMessage response

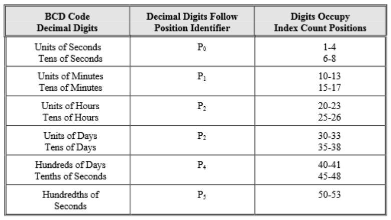
- The response reports seconds, minutes, hours and day of year (in this order, starting with P0) as shown below the example screenshot.
- Need to convert hex to ?? (see Paul Myers or Ron Dries)

Keith,

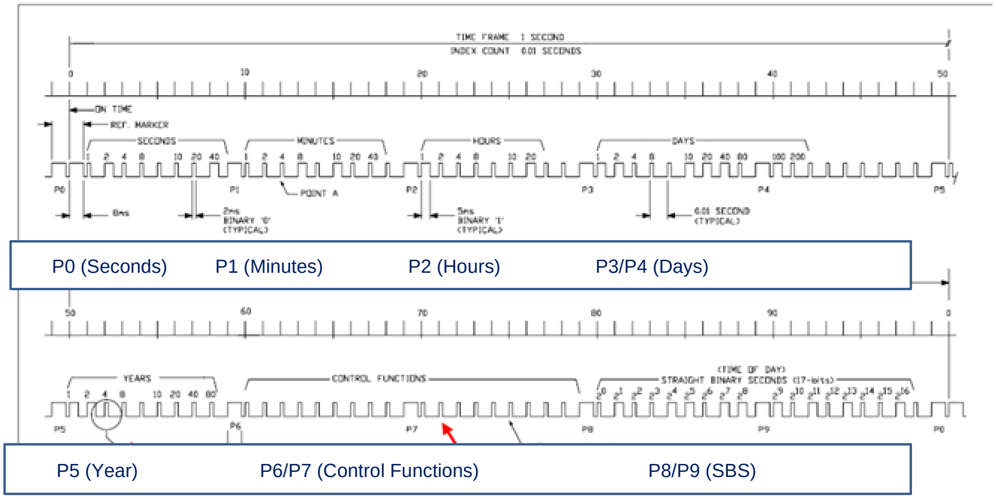
This is the IR_GetMessage response at 2:36:40 PM



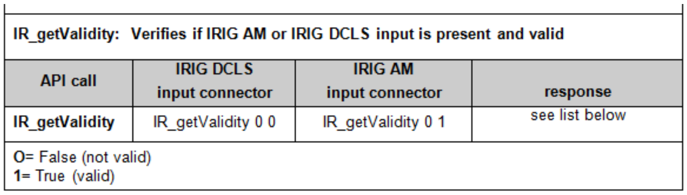
```
C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API>IR_GetMessage 0 1
  IR (1) Message:
    Sub P0: 02D3
    Sub P1: 0666
    Sub P2: 0A29
    Sub P3: 0E81
    Sub P4: 1203
    Sub P5: 1600
    Sub P6: 1A00
    Sub P7: 1E00
    Sub P8: 2200
    Sub P9: 2600
C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API>
```



BCD Code Decimal Digits	Decimal Digits Follow Position Identifier	Digits Occupy Index Count Positions
Units of Seconds Tens of Seconds	P_0	1-4 6-8
Units of Minutes Tens of Minutes	P_1	10-13 15-17
Units of Hours Tens of Hours	P_2	20-23 25-26
Units of Days Tens of Days	P_2	30-33 35-38
Hundreds of Days Tenths of Seconds	P_4	40-41 45-48
Hundredths of Seconds	P_5	50-53


TIME FRAME 1 SECOND
INDEX COUNT 0.01 SECONDS
0
10
20
30
40
50
ON TIME
REF. MARKER
SECONDS
1 2 4 8 10 20 40
MINUTES
1 2 4 8 10 20 40
HOURS
1 2 4 8 10 20
DAYS
1 2 4 8 10 20 40 80 100 200
P0
P1
POINT A
P2
P3
P4
P5
8ms
2ms
BINARY '0'
(TYPICAL)
5ms
BINARY '1'
(TYPICAL)
0.01 SECOND
(TYPICAL)
P0 (Seconds)
P1 (Minutes)
P2 (Hours)
P3/P4 (Days)
50
60
70
80
90
0
YEARS
1 2 4 8 10 20 40 80
CONTROL FUNCTIONS
(TIME OF DAY)
STRAIGHT BINARY SECONDS (17-bits)
2^0 2^1 2^2 2^3 2^4 2^5 2^6 2^7 2^8 2^9 2^10 2^11 2^12 2^13 2^14 2^15 2^16
P5
P6
P7
P8
P9
P0
P5 (Year)
P6/P7 (Control Functions)
P8/P9 (SBS)

****IRIG AM and IRIG DCLS input validity (IR_GetValidity) / Troubleshooting IRIG input



IR_getValidity: Verifies if IRIG AM or IRIG DCLS input is present and valid			
API call	**IRIG DCLS input connector**	**IRIG AM input connector**	**response**
IR_getValidity	IR_getValidity 0 0	IR_getValidity 0 1	see list below
O= False (not valid) **1**= True (valid)			

If the TSync is not declaring sync after IRIG is lost and then regained, I recommend you look at the "validity" of the input reference, once it's been reconnected. The API call to confirm the TSync board is declaring the IRIG input valid is IR_GetValidity 0 0. This command will respond with a Time and 1PPS validity. With the IRIG signal connected, both values should be "1". If either, or both, of these values is a "0", there is an issue with the IRIG signal that is preventing the TSync board from declaring the IRIG input as valid (either with the signal or the configurations).

If the IRIG AM or IRIG DCLS Input is "not valid"

If the IRIG input cable is connected to the TSync board, but IRIG input is "not valid", configure the IRIG Source as IRIG B AM 1kHz (any coded expression is fine) and configure the TSync's IRIG input to the most basic input configuration (IRIG B122 which is just: IRIG B AM 1kHz coded expression of BCD). If it still doesn't indicate valid, the issue is likely a cable or IRIG output issue- not a configuration issue.

A) IRIG AM input

1) Make sure the IRIG source is connected to the IRIG AM input on the TSync (not the IRIG DCLS input). It's the shorter of the octopus cables.
2) Make sure the IRIG source is configured to output IRIG B AM (1000 Hz) to the TSync-PCIe board.
3) Configure the TSync's IRIG input as follows:
   - **IR_SetMode 0 1 1** (For **manual** configuration mode)
   - **IR_SetFormat 0 1 1** (For **IRIG B** output) (This is the default setting)

   Note: IRIG B is always 1kHz input -This is the default setting, and since it can't be changed, no need to configure/verify this value with IR_SetFreq 0 1 1
4) **IR_SetCodedExpr 0 1 2** (for **BCD** output only)

   Note: (IRIG AM input) IR_ getMod 0 1 1 should always indicate "(1) AM" (Can't be configured as DCLS)
5) Verify the green Sync LED on the edge of the TSync-PCIe board illuminates.

   Note: if the Sync LED is not green, perform an **IR_getValidity 0 1**. If Time and PPS are both valid, make sure the reference of "IR(1)" is listed and enabled in the reference priority table.

B) IRIG DCLS input

1) Make sure the IRIG source is connected to the IRIG DCLS input on the TSync (not the IRIG AM input)

   Single-ended DCLS input pinout

   Email from Dave L (16 Sept 17) Single-ended DCLS input does not connect the GND to the (-) Input.

   For IRIG DCLS single ended input:

   A. The "IRIG DCLS In +" signal from the IRIG generator should be connected to the "IRIG DCLS Input +" pin.

   B. The IRIG "DCLS IN –"signal from the IRIG generator should be connected to one of the ground pins on the breakout cable.

C. The “IRIG DCLS Input –“ pin should be left unconnected on the TSync-PCIe side

This is something we overlooked at the beginning. Please try this and let me know how it works.

2) Make sure the IRIG source is configured to output IRIG B DCLS (1000 hz)

3) Configure the TSync’s IRIG input as follows:

   IR_SetMode 0 0 1 (For manual input configuration mode)

   IR_SetFormat 0 0 1 (For IRIG B input) (B is the default setting. But it can accept A, B, G or E)

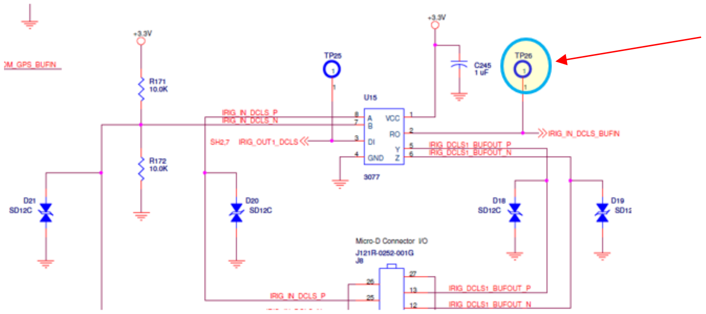
   Note: IRIG B is always 1kHz input -This is the default setting, and since it can’t be changed, no need to configure this value with **IR_SetFreq0 1 1**

   IR_SetCodedExpr 0 0 2 (for BCD only)

   Note: (IRIG DCLS input) **IR_ getMod 0 0 1** should always indicate “(**0) DCLS”** (Can’t be configured as AM)

4) Verify the green Sync LED on the edge of the TSync-PCIe board illuminates.

5) IRIG DCLS signal should be present at **TP26**.



6) If the IRIG source is expecting 50 ohm termination, need to add 50 ohm termination resistor to the IRIG DCLS input (the TSync’s IRIG DCLS input is high impedance.

Error: opt err (RC_STUB_FUNC)

- Occurs if the IRIG Format is currently set to a Format (such as B) and the IRIG Frequency is being changed (via IR_SetFreq) to an invalid selection for that particular Format (such as trying to set it to 100 Hz when with its currently set to Format B, which is always 1000 Hz only)
- Refer to Salesforce case 19282

```
[root@lab-r46-05 tsync]# ./examples/IR_SetFreq 0 0 1
     Error: opt err (RC_STUB_FUNC).
             IR (0) Freq: (1) 100 Hz

          [root@lab-r46-05 tsync]# ./examples/IR_SetFreq 0 0 2

              IR (0) Freq: (2) 1 kHz
             Error: opt err (RC_STUB_FUNC).
```

**New Year rollover

- The Tsync family of timing boards know when it’s the last day of the year to automatically increment the year (whether or not it’s synced with an external input).
- Whether it’s a leap year (366 days) or not (365 days) is based on internal rules (not an internal calendar)

Email Keith sent (3 Aug 2015) Regarding year rollover, the TSync-PMC board uses its own rules to know when the New Year rollover is occurring (including whether or not the current year is a leap year with 366 days instead of 365) so that It can automatically increment the year value on the first day of each New Year. The user doesn’t need to do anything with the board to have it automatically increment the year on the correct date. This is true whether or not the Tsync board is receiving an external IRIG input signal.

*Setting the year value upon each TSync-PCIe power-up

- Per Denis Reilly, the year can be set with a four digit value between 1970 and 2099.
- Default year after each power-up is year **2000.**

The **Default year value is 2000**, if the TSync-PCIe board doesn't have GPS input, the TSync-PCIe board can't read the year from the IRIG input, or if a user does not manually set the year.

The year value DOES NOT need to be set in order for it to sync to the IRIG input (the year starts at 2000 and simply counts-up from there). Refer to the details further below for info on either automatically or manually configuring the year value:

There are a few different ways to set the year value in the TSync board (the year needs to be set after every power-up).

1) GPS input will automatically set the year when using GPS as an input reference.
2) When using IRIG input, the IRIG source may provide the year in the data stream. If it does, the TSync board
3) can be programmed to read the year.

   When using an IRIG signal that does not contain year information, the year can be set with either an API call or an example program. API calls can be scripted so that each time the board powers-up, the script can be run to perform all desired configurations.

   FYI: the API call/example program to set the year is **CS_SetYear 0 xxxx** <enter> (where "0" is for only one- or the first- TSync-PCIe board installed in the system, and "**xxxx**" is the desired year value, such as "2019" for example).

4) If using the board in a Windows PC, the Windows driver for the TSync-PCIe board contains a GUI interface called the "**Control Utility**". The year can be set using the **Date** -> **Set Year** menus in this GUI interface.

Apparently, the year can also be set with a command that can be run in a custom batch file

Q. Is there a command I could run to automate this procedure on startup via batch script or vbs script?

A. **Reply from Dave Lorah (16 Jul 14)** We do not have a batch file but if you run this command: **setyear –y %date:10,4%** on startup this will set the year to what the computer's year is set to.

Setting the year value

Where this gets interesting is specific to the location of the year value in the Control Field. The IRIG specs have been changed a couple of times over the years. It has always been an option potion of the IRIG signal, allowing manufacturers of IRIG output devices to decide if they desire to include data in this section of the signal. When the specs were first released, there was no defined location for the year value. So, Spectracom just added the year value to the section. Many vendors designed their systems to read the year in the location we randomly selected. Years later, the IRIG specs changed. The year was still optional, but the specs started defining where the year should be placed, if it was included. Unfortunately, this was 1 place away from where we were already including the year.

We couldn't just change our location of the year, without causing every other vendor that was reading out year to have to change their software. So we left it where it was. However, not all systems were designed around Spectracom IRIG output, so we had to start supporting both the old and new locations. The "Spectracom Format" above is for our Spectracom products that have the year one place away from the RCC 200-04 spec. The "Spectracom FAA format" was a special deviation for a particular customer. "NASA" has their own IRIG formatting and the "IEEE" format adds other values (known as extension), which includes data such as sync status/validity of the IRIG signal. We can use these extensions, if they are provided by the IRIG source.

Local (defines local time offset on input)
(IR_GetLocal / IR_SetLocal)

Timescale (UTC, TAI, local, etc)
(IR_GetTimeScale / IR_SetTimeScale)

Getref (reference identified for reference).

**IRIG input Year info

The power-up default year **value is "2000"**, and it will remain at "2000" if the TSync board doesn't have GPS/GNSS input, the TSync board can't read the year from the IRIG input, or if a user does not manually set the year.

The year value DOES NOT need to be set in order for it to sync to the IRIG input (the year starts at 2000 and simply counts-up from there). Refer to the details further below for info on either automatically or manually configuring the year value:

There are a few ways to set the year value in the TSync board (the year needs to be set after every power-up).

1) GPS/GNSS input will automatically set the year when using GPS as an input.
    - ➢ When using IRIG, the IRIG source may provide the year in the data stream. If it does, the TSync board can be programmed to read the year.
2) When using an IRIG signal that does not contain year information, the year can be set with either an API call or an example program. API calls can be scripted so that each time the board powers-up, the script can be run to perform all desired configurations.
    - ➢ FYI- the API call/example program to set the year is **CS_SetYear 0 xxxx** <enter> (where "0" is for only one TSync board installed and "xxxx" is the desired year value).

3) If using the board in a Windows PC. The Windows driver for the TSync-PCI104 board contains a GUI interface called the "Control Utility". The year can be set using the **Date** -> **Set Year** menus in this GUI interface.
    - ➢ Present year is available via either the **Control Field (CF**) and/or the **BCDyear** portions of the IRIG signal, If they are present in the signal being provided by the IRIG source (a third digit of: 4, 5, 6 or 7, such as **B124, B125, B126** for examples)

Sample list of codes (with ones that can provide year listed in blue)

IRIG Code	Description of IRIG string	Notes about year
B000	IRIG B, TTL, BCD,CF and SBS	Can technically provide year info in the Control Field as BCDYear, **BUT the board wont read it**, because the coded expression doesn't include "BCDYear"
B120	IRIG B, AM, 1 kHz , BCD, **CF** and SBS	Can technically provide year info in the Control Field as BCDYear, BUT the board wont read it, because the coded expression doesn't include "BCDYear"
B121	IRIG B, AM, BCD, CF	Can technically provide year info in the Control Field as BCDYear, BUT the board wont read it, because the coded expression doesn't include "BCDYear"
B122	IRIG B, AM, 1 kHz, and BCD $_{TOY}$	No year info provided. (Must set year in the device(s) receiving the IRIG signal)
B123	IRIG B, AM, 1 kHz , BCD $_{TOY}$, SBS	No year info provided (Must set year in the device(s) receiving the IRIG signal)
B124	IRIG B, AM, BCD$_{TOY}$, **BCD$_{YEAR}$, CF,** SBS	Can provide year info in BCDyear and/or CF
B125	IRIG B, AM, 1kHz, BCD$_{TOY}$, **BCD$_{YEAR}$, CF**	Can provide year info in BCDyear and/or CF
B126	IRIG B, AM, 1kHz, BCD$_{TOY}$, **BCD$_{YEAR}$**	Can provide year info in BCDyear
B127	IRIG B, AM, 1kHz, BCD$_{TOY}$, **BCD$_{YEAR}$,** SBS	Can provide year info in BCDyear
E000	IRIG E TTL, BCD$_{TOY}$,**CF** and SBS	Can provide year info in CF (as BCDYear)
E110	IRIG E AM, 100 Hz, BCD $_{TOY}$,**CF** and SBS	Can provide year info in CF(as BCDYear)
E120	IRIG E AM, 1 kHz, BCD $_{TOY}$,**CF** and SBS	Can provide year info in CF (as BCDYear)

IRIG B120 contains data in all portions of the IRIG B data stream, including data in the optional Control Field section, BCD data and SBS (straight Binary Seconds) data.   IRIG B120 does contain more data than IRIG B122 and so IRIG B122 is a subset of IRIG B120. IRIG B120 indicates that the system requires an IRIG B signal with BCD data, but does not require any data be present in the Control Field section (it ignores the Control Field and does not require the SBS data to be present.

Most IRIG input systems commonly ignore the Control Field section altogether, so this system is not unique in this respect. If there is any data present in this section of the data steam (such as the Model 9383 populates), they usually just ignore this data (the primary data that is usually contained in the section of the data stream is the current year information). Since most systems (such as those that are B122) ignore the Control Field section, the year value in that system cannot be automatically set by the Model 9383 and so it will need to be manually set. Otherwise, a B122 system ignoring the Control Field is not a problem. As far as B122 not reading the SBS data, this should not be a problem either.   With B122, the other system should just ignore the SBS data also contained in the B123 data stream.

In summary: Any system that is IRIG B122 input should just ignore the additional data that is contained in an IRIG B120 data stream and so it should interface with the Model 9383 with no problem, other than the need for the year to be manually set in that system because the year is only contained in the Control Field section, which the system ignores.

Determining which method to use is dependent on the presence of GPS or the capabilities of the IRIG source:

1) Is GPS input available to the TSync board?
2) Is the IRIG time code generator a Spectracom SecureSync, Model 9483 or another TSync-PCIe board?
3) Is the IRIG time code generator a Spectracom Model 8183, 9183, 9283, 9383 or a TPRO/TSAT timing board?
4) Or, is the IRIG time code generator an IRIG generator from another vendor?
5) If the IRIG generator is from another vendor, can it provide?:

   A) Example BCD Coded expressions containing "BCD year" (either of the following):

   - **B124** (BCD TOY / **BCD$_{\textbf{YEAR}}$** / CF / SBS)
   - **B125** (BCD TOY / **BCD$_{\textbf{YEAR)}}$** / CF)
   - **B126**, (BCD$_{\text{TOY}}$, **BCD$_{\textbf{YEAR)}}$**

   B) Control Functions (and containing either of the following):

   - RCC 200-04
   - IEEE C37.118-2005 (IEEE-1344 extensions)
   - Spectracom IEEE C37.118-2005 (Spectracom-modified IEEE-1344 extensions)

In summary:

- **If GPS is available as an input to the TSync board:**
  - Refer to I**tem 1** directly below (GPS automatically sets the year)

- **If GPS is not available as an input to the TSync board:**
  - If the IRIG source is a Spectracom SecureSync, Model 9483 or another TSync-PCIe board, refer to **item 4** directly below (Using IEEE 1344 and BCD year in the Coded Expression)
  - If the IRIG source is an earlier Spectracom Model 8183, 9183, 9283 or 9383 refer to **item 6** directly below (Using "Spectracom Format" year in the Coded Expression).

Summary of methods to provide year information to the TSync-PCIe board:

(**Note**: Details for each method are further below in)

1) **Use GPS as the input reference (GPS sets the year automatically).**
2) **If the Windows driver is installed, the Windows Control Utility can be used to manually set the year after each boot-up.**
3) **Manually set the year after each TSync-PCIe boot-up using the CS_SetYear 0 xxxx call/example program (Refer to "A" below).**

   Note: Apparently, the year can also be set in Windows with a command that can be run in a custom batch file

   Q. Is there a command I could run to automate this procedure on startup via batch script or vbs script?
   A. **Reply from Dave Lorah (16 Jul 14)** We do not have a batch file but if you run this command: **setyear –y date:10,4%** on startup this will set the year to what the computers year is set to.

4) **Use the “IEEE 1344” Control Field, combined with a Coded Expression that contains both CF and year information (such as B124 or B125). (Refer to “B” below). Use this configuration with a SecureSync, 9483 or TSync-series board as the IRIG source.**

Note: This is the recommended configuration to use (If GPS input is not available) when IEEE 1344 extensions and CF are available from the IRIG generator (such as when using a SecureSync or TSync as the IRIG source). This configuration provides all of the data that is provided in the IEEE 1344 extension (such as Leap Second pending notification, DST / Time Zone Offset info and time Quality info) plus the BCD year.

SecureSyncs, Model 9483s and other TSync-PCI104 boards have this capability.

5) **Use a “Coded Expression” that contains BCDyear information (B124, B125, B126, etc). (Refer to “C” below).**

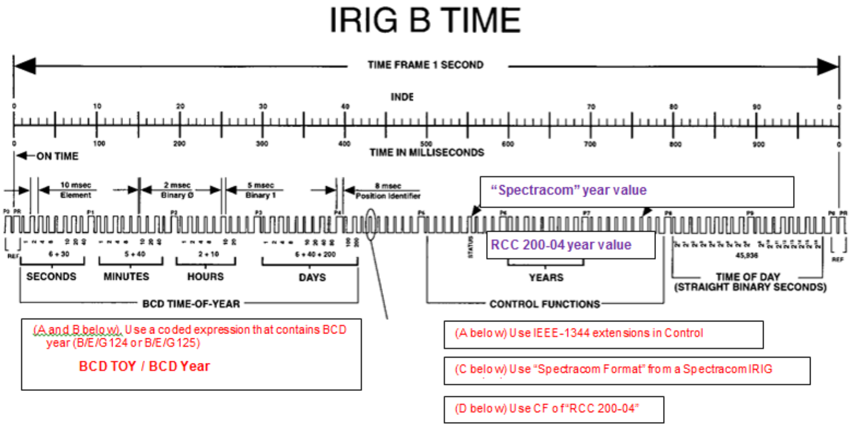
Note: This is the second best configuration to use, (If GPS input is not available) when CF is available from the IRIG generator, but the IRIG source can’t provide the IEEE 1344 extensions.

6) **Use the IRIG output from an earlier Spectracom Model 8183, 9183, 9283 or 9383 NetClock and select “Spectracom Format” IRIG input in the Control Field to automatically set it. (Refer to “D” below).**

Note: this Option is only available when using a Spectracom GPS NetClock /SecureSync as the Time Generator. As the SecureSync/9483 can also provide IEEE 1344 extensions, this configuration should only be used when the IRIG reference is an earlier Model 8183, 9183, 9283 or 9383. When using a SecureSync/9483 or another TSync-PCIe board, use the IEEE 1344/ BCD year info method above, instead)

7) **Use “RCC 200-04” in the Control Function field to automatically set it. (Refer to “E” below).**

Via an IRIG input from an IRIG generator



Note: The Coded expression defines both the BCD format and whether CF functions are present

If using another TSync board as the IRIG generator to sync a TSync board via IRIG input

- After each power-up , no year information is provided (default format is B000)

Note: Refer to this section if using a **TSync board** as the IRIG Master providing IRIG to another IRIG Slave. Or skip this section if the IRIG source is not another Tsync board

The TSync boards do not output the year in its IRIG output after each boot-up (default format is B002 (until it's been reprogrammed to output the year. When using another TSync board as the IRIG source, refer to the info below:

To output year information from a TSync board, via the Control Functions

A) Provide the year via the Control Functions: Change the Coded Expression of the output to one that includes Control Functions **AND BCDYear** (can use 4, 5, 6, or 7). Refer to Coded expression table below for details

CodedExp (**AM or DCLS output Coded Expression**)			
API call	**IRIG DCLS output connector**	**IRIG AM output connector**	**response**
IP_GetCodedExp	**IP_GetCodedExp 0 0**	**IP_GetCodedExp 0 1**	see list below
IP_SetCodedExp	**IP_SetCodedExp 0 0 z**	**IP_SetCodedExp 0 1**	see list below

Available selections for "x" above (all but the first one provide year)

0 = BCD TOY, Control Function, Straight Binary Seconds (power-up default, no year provided and doesn't read BCDYear, so can't use this one)

1= BCD TOY, Control Functions (no BCDyear provided and doesn't read BCDYear, so can't use this one)

2= BCD TOY (no BCDyear provided and doesn't read BCDYear, so can't use this one)

3= BCD TOY, Straight Binary Seconds(no year provided and doesn't read BCDYear, so can't use this one)

4= **BCD TOY/BCDYear**, **Control Functions**, Binary Seconds (SBS)

5= BCD TOY/ B**CDYe**ar, Binary Seconds (SBS)

6= BCD TOY/**BCDYear**

7= BCD TOY/**BCDYear**, Binary Seconds (SBS)

A) Provide the year in the Control Field: Change the Control Field of the output to a selection that includes year information (refer to the Ctrlfield table below for details)

- Need to configure the Control Field for the desired year location (and the coded Expression needs to contain "BCDyear, as well).

CtrlField (**defines AM or DCLS Control Field**)			
API call	**IRIG DCLS input connector**	**IRIG AM input connector**	**response**
IP_GetCtrlField	**IP_GetCtrlField 0 0**	**IP_GetCtrlField 0 1**	see list below
IP_SetCtrlField	**IP_SetCtrlField 0 0 x**	**IP_SetCtrlField 0 1 x**	see list below

Available selections for "x" above (all of these provide year)

CF=1 *Fields conform to* ***RCC 200-04***

CF=2 *Fields conform to* ***IEEE C37.118-2005 (1344 extensions)***

CF=3 *Fields conform to* ***Spectracom format***

Important Note*: Although these three Control Fucntion settings contain BCDyear, the BCDyear in the Control Function field will ONLY be read by the system, if the* ***Coded Expression*** *field includes* ***"BCDYear"***

Example*: although* ***B000*** *contains "CF field", its coded expression doesn't include "BCDyear". So the system will ignore the "BCDYear value" in the CF field. The year will not be set in the TSync board.*

Details on how to program a TSync board as an IRIG Slave that can read the year

A) Use the "IEEE 1344" Control Functions, combined with a Coded Expression that contains both Control Field (CF) as well as BCD year information (IRIG A/B/G/E124 or A/BG/E125). Use this configuration with a SecureSync, 9483 or TSync-PCIe board as the IRIG source.

Note: IRIG source must be able to provide either IRIG A/B/G/E**124** or A/B/G/E**125** and support IEEE-1344 extensions in its Control Functions field

B. **BCD TOY / BCD Year / CF / SBS** OR

C. **BCD TOY / BCD Year / CF**

Note: This is the recommended configuration to use (If GPS input is not available) when IEEE 1344 extensions and CF are available from the IRIG generator (such as when using a SecureSync or TSync as the IRIG source). This configuration provides all of the data that is provided in the IEEE 1344 extension (such as Leap Second pending notification, DST / Time Zone Offset info and time Quality info) plus the BCD year.

Note: SecureSyncs, Model 9483s and other TSync-PCIe boards have the capability to output 1344 extensions.

Configure the IRIG time code generator (IRIG source):

1. Configure the IRIG source/Reference (SecureSync, another TSync board or a Model 9483 for examples) to provide IEEE 1344 Control Field information. For a Spectracom IRIG generator, in the "Control Field" drop-down, select "IEEE 1344 (C37.118-2005)".
2. Configure the IRIG output to include the year information in the "Coded Expression" field. Example "Coded Expressions" include:

   A. **BCD TOY / BCD Year / CF / SBS OR**

   B. **BCD TOY / BCD Year / CF**

B) With a Spectracom SecureSync as the IRIG source:

In the "Coded Expression" field drop-down list, there are four selections that all provide BCD Year information (they are selections "(4)", "(5)", "(6)" and "(7)"). Of these four available selections, two of these also include the "Control Field" section (they are selections "(4)" and "(5)").

If you configure the Coded Expression field in the RIG output to either (4) or "(5)" (with the "Control Field" selection still set to "IEEE 1344), the SecureSync will provide the TSync board with all of the IEEE1344 information and the current year.

C) With another Tsync board as the IRIG source

To output year information from a Tsync board, need to use Either 1 or 2 below (based on the device receiving the IRIG signal):

1) **Provide the year via BCDyear**: Change the Coded Expression of the output to one that includes BCDyear information (can use 4, 5, 6, or 7). Refer to the Coded expression table below for details (values that can be used are listed in blue).

CodedExp (**AM or DCLS output Coded Expression**)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetCodedExp	**IP_GetCodedExp 0 0**	**IP_GetCodedExp 0 1**	see list below
IP_SetCodedExp	**IP_SetCodedExp 0 0 z**	**IP_SetCodedExp 0 1**	see list below

0 = BCD TOY, Control Function, Straight Binary Seconds (power-up default, no year provided and doesn't read BCDYear, so can't use this one)
1= BCD TOY, Control Functions (no BCDyear provided and doesn't read BCDYear, so can't use this one)
2= BCD TOY (no BCDyear provided and doesn't read BCDYear, so can't use this one)
3= BCD TOY, Binary Seconons (no year provided and doesn't read BCDYear, so can't use this one)
4= **BCD TOY/BCDYear**, **Control Functions**, Binary Seconds (SBS)
5= BCD TOY/ B**CDYe**ar, Binary Seconds (SBS)
6= BCD TOY/**BCDYear**
7= BCD TOY/**BCDYear**, Binary Seconds (SBS)

2) **Provide the year in the Control Field**: Change the Control Field of the output to a selection that includes year information (refer to the Ctrlfield table below for details, available selections in blue)

   - ➢ Need to configure the Control Field for the desired year location.

Note: Should be able to select any of three listed in blue. However, Tim T and I were not able to have "2" (IEEE extensions) set the year in the slave. Right after we just changed both the IRIG output and IRIG input to "3" (Spectracom), the year was automatically set in the Slave board/

CtrlField (**defines AM or DCLS Control Field**)

API call	IRIG DCLS input connector	IRIG AM input connector	response
IP_GetCtrlField	**IP_GetCtrlField 0 0**	**IP_GetCtrlField 0 1**	see list below
IP_SetCtrlField	**IP_SetCtrlField 0 0 x**	**IP_SetCtrlField 0 1 x**	see list below

Available selections for "x" above (all of these provide year)
CF=1 Fields conform to **RCC 200-04**
CF=2 Fields conform to I**EEE C37.118-2005 (1344 extensions)**
CF=3 Fields conform to **Spectracom format**

Connect the Tsync board to its IRIG source/reference via either a dedicated IRIG AM or IRIG DCLC connection

1. **Using IRIG AM input (Note: refer to the next section down for DCLS input)**
   - ➢ Configure the TSync-PCI's "AM" IRIG input for IEEE 1344 and BCD Year data:

The TSync board's IRIG AM input connector needs to be configured as **IRIG B124** or **IRIG B125** (IRIG B AM and Codedexpr set to either selections "4" or "5")

(IRIG B AM, CtrlField set to "IEEE *C37.118-2005*" and Codedexpr set to either selections of "4" or "5")

1. Make sure the IRIG source is connected to the IRIG AM input on the TSync-PCI104 board (not the IRIG DCLS

input)

2. Enter: **IR_SetMode 0 1 1** (For manual configuration mode)
3. Enter: **IR_SetFormat 0 1 1** (For IRIG B input) (This is the default setting)
4. Enter: **IR_SetMod 0 1** ***1*** (For IRIG AM input) (This is the default setting)

Note: IRIG B is always 1kHz input -This is the default setting, and since it can't be changed, no need to configure/verify this value with **IR_SetFreq 0 1 1**

5. Enter: **IR_SetCtrlField 0 1 2** (for "IEEE C37.118-2005" IEEE 1344 Control Function).

CtrlField (**defines AM or DCLS Control Field**)

API call	IRIG AM input connector	response
IR_GetCtrlField	**IR_GetCtrlField 0 1**	see list below
IR_SetCtrlField	**IR_SetCtrlField 0 1 x**	see list below

Available selections for "x" above (all of these provide year)

CF =1 Fields conform to **RCC 200-04**
CF =2 Fields conform to I**EEE C37.118-2005 (1344 extensions)**
CF =3 Fields conform to **Spectracom format**

6. Enter: **IR_SetCodedExp 0 1 4** OR **IR_ SetCodedExp 0 1 5** (either value will work).

CodedExp (**AM or DCLS output Coded Expression**)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IR_GetCodedExp	**IR_GetCodedExp 0 0**	**IR_GetCodedExp 0 1**	see list below
IR_SetCodedExp	**IR_SetCodedExp 0 0 z**	**IR_SetCodedExp 0 1**	see list below

Available selections for "x" above in blue

0 = BCD TOY, Control Functions, Binary Seconds (SBS)
1 = BCD TOY, Control Functions (power-up default)
2 = BCD TOY
3 = BCD TOY, Binary Seconds (SBS)
4 = BCD TOY/Year, Control Functions, Binary Seconds (SBS)
5 = BCD TOY/Year, Binary Seconds (SBS)
6 = BCD TOY/Year
7 = BCD TOY/Year, Binary Seconds (SBS)
8 = Unknown- no fields

2. Using IRIG DCLS input (Note: refer to the previous section for AM input)

Single-ended DCLS input pinout

Email from Dave L (16 Sept 17) Single-ended DCLS input does not connect the GND to the (-) Input.

For IRIG DCLS single ended input:

A. The “IRIG DCLS In +” signal from the IRIG generator should be connected to the “IRIG DCLS Input +” pin.

B. The IRIG “DCLS IN –“signal from the IRIG generator should be connected to one of the ground pins onthe breakout cable.

C. The “IRIG DCLS Input –“ pin should be left unconnected on the TSync-PCIe side

This is something we overlooked at the beginning. Please try this and let me know how it works.

Configure the TSync-PCI’s IRIG “DCLS Input for IEEE 1344 and BCD Year data:

The TSync-PCI104 ’s IRIG AM input connector needs to be configured as **IRIG B124** or **IRIG B125** (IRIG B AM, CtrlField set to “IEEE *C37.118-2005*” and Codedexpr set to either selections “4” or “5”)

1. Make sure the IRIG source is connected to the IRIG AM input on the TSync-PCI104 board (not the IRIG DCLS input)
2. Enter: **IR_SetMode 0 0 1** (For manual configuration mode)
3. Enter: **IR_SetFormat 0 0 1** (For IRIG B input) (B is the default setting. But it can accept A, B, G or E)
4. Enter: **IR_SetMod 0 0 1** (For IRIG AM input) (This is the default setting)

Note: IRIG B is always 1kHz input -This is the default setting, and since it can’t be changed, no need to configure/verify this value with **IR_SetFreq 0 0 1**

5. Enter: **IR_SetCtrlField 0 0** 2 (for “IEEE C37.118-2005” IEEE 1344 Control Function).

CtrlField (**defines AM or DCLS Control Field**)

API call	**IRIG DCLS input connector**	**response**
IR_GetCtrlField	**IR_GetCtrlField 0 0**	see list below
IR_SetCtrlField	**IR_SetCtrlField 0 0 x**	see list below

Available selections for “x” above in blue (all of these provide year)
CF=1 Fields conform to **RCC 200-04**
CF=2 Fields conform to I**EEE C37.118-2005 (1344 extensions)**
CF=3 Fields conform to **Spectracom format**

6. Enter: **IR_SetCodedExp 0 0 4**   OR   **IR_SetCodedExp 0 0 5** (either value will work).

- **Where CodedExpression 4 =** BCD TOY/Year, Ctrl Func (CF), Straight Binary Seconds (SBS)
- **Where CodedExpression 5 =** BCD TOY/Year, Ctrl Func (CF)

CodedExp (**AM or DCLS output Coded Expression**)		
API call	**IRIG DCLS** **output connector**	**response**
IR_GetCodedExp	**IR_GetCodedExp 0 0**	see list below
IR_SetCodedExp	**IR_SetCodedExp 0 0 z**	see list below
Available selections for “x” above (all but the first one provide year) **0** = BCD TOY, Control Functions, Binary Seconds (SBS) **1** = BCD TOY, Control Functions (power-up default) **2** = BCD TOY **3** = BCD TOY, Binary Seconds (SBS) **4** = BCD TOY/Year, Control Functions, Binary Seconds (SBS) **5** = BCD TOY/Year, Binary Seconds (SBS) **6** = BCD TOY/Year **7** = BCD TOY/Year, Binary Seconds (SBS) **8** = Unknown- no fields		

D) Use a “Coded Expression” configuration that contains both Control Field (CF) as well as BCD Year information (IRIG generator does not include IEEE 1344 extensions)

Note: This is the second best configuration to use, (If GPS input is not available) when CF is available from the IRIG generator/source, but the IRIG source can’t provide the IEEE 1344 extensions.

Configure the IRIG output to include the year information in the “Coded Expression” field. Example “Coded Expressions” include either of the following:

Note: IRIG source must be able to provide either IRIG A/B/G/E**124** or A/B/G/E**125**

A. **BCD TOY / BCD Year / CF / SBS** OR

B. **BCD TOY / BCD Year / CF**

E) Use the IRIG input from an earlier Spectracom Model 8183, 9183, 9283 or 9383 Spectracom NetClock’s (or a TSync timing board’s) IRIG output to set the year (Refer to “D” below).

Note: this Option is only available when using a Spectracom GPS NetClock /SecureSync or timing board as the Time Generator.

As the SecureSync/9483/TSync series timing board can also provide IEEE 1344 extensions, this configuration should only be used when the IRIG reference is an earlier Model 8183, 9183, 9283 or 9383. When using a SecureSync/9483 or another TSync-PCIe board, use the IEEE 1344/ BCD year info method above, instead)

In the TSync series timing board

When using GPS input, the TSync series board receives the year from the GPS receiver at each start-up (the year is not stored). When using IRIG input instead, it may be able to read the year value at each start-up if the IRIG generator provides the year in the data stream. In the case of a Spectracom NetClock IRIG output (8183, 9283, 9383, etc), the board needs to be configured each power-up to be able to read the year.

Edited email Dave Lorah sent to a customer:
Since you are using a Spectracom NetClock as the IRIG source, the year information is contained in the control field portion of the IRIG message. So the TSync board can read the year info from the IRIG input. However, the Spectracom IRIG output does not put the year in the conventional place in the IRIG signal. When the products were developed there was no convention for the year info, so it was placed in the control field portion of the message. The Tsync PCIe board has the capability to be set to read the year info from the Spectracom IRIG but it must be told the input is the Spectracom version of IRIG. Also the board must be told to read the year info from the IRIG input.

This can be done by running a batch file from the startup menu on your PC, so every time the PC it started the board gets set to read the Spectracom IRIG year input.

For IRIG AM input- The batch file should run these two commands

IR_SetCodedExp.exe 0 1 0 (sets the IRIG inputs Coded expression)

The TSync-PCI104 board supports IRIG inputs of the following coded expressions combinations
for BCDTOY, CF, SBS, and BCDYEAR fields:
0 – BCDTOY, CF, SBS (NetClock's IRIG output)
1 – BCDTOY, CF
2 – BCDTOY
3 – BCDTOY, SBS
4 - BCDTOY, BCDYEAR, CF, SBS
5 - BCDTOY, BCDYEAR, CF

IR_SetCtrlField.exe 0 1 3 (sets the IRIG Control field configuration of the IRIG data stream – where the 3 is used to define the Control Field as it is in the NetClock).

If using IRIG DCLS:
.IR_SetCodedExp 0 0 0
.IR_SetCtrlField 0 0 3
These programs are contained in the TSync API in Examples.

F) Use "RCC 200-04" in the Control Function field to automatically set it.

Details for the methods to provide year information to the TSync-PCIe board:

CS_SetYear (Manually sets just the year)

- The **CS_SetYear 0 xxxx** API call/example program (where xxxx is the current year value) sets the year to a specified value.
- This command needs to be run after each power-up, if the year can't be read from the GPS or IRIG input.

1. Using IRIG AM input (refer to the next section down for IRIG DCLS input)

If the IRIG source (Such as a NetClock or SecureSync) provides the "RCC 200-04" configuration (in the Control Function field), the TSync timing board can read the year value from the IRIG generator.

The TSync board's IRIG AM input connector needs to be configured as **IRIG B124** or **IRIG B125** (IRIG B AM Codedexpr set to either 4 or 5, CtrlField set to "RCC 200-04").

- Make sure the IRIG source is connected to the IRIG AM input on the TSync (not the IRIG DCLS input)

- ➢ IR_ SetMode 0 1 1 (For manual configuration mode)
- ➢ IR_ SetFormat 0 1 1 (For IRIG B output) (This is the default setting)
- ➢ IR_ SetMod 0 1 1 (For IRIG AM input) (This is the default setting)

Note: IRIG B is always 1kHz input -This is the default setting, and since it can't be changed, no need to configure/verify this value with **IR_SetFreq0 1 1**

- ➢ IR_ SetCodedExpr 0 1 4 OR IR_ SetCodedExpr 0 1 5 (either value will work)

CE=4 BCD TOY/Year, Ctrl Func, Binary Seconds
CE=5 BCD TOY/Year, Ctrl Fun

- ➢ IR_SetCtrlField 0 1 1 (for RCC 200-04 Control Function).

2. Using IRIG DCLS input (refer to the section above for IRIG AM input)

If the IRIG source (Such as a NetClock or SecureSync) provides the "RCC 200-04" configuration (in the Control Function field), the TSync series board can read the year value from the IRIG generator.

The TSync-board's IRIG AM input connector needs to be configured as **IRIG B124** or **IRIG B125** (IRIG B AM Codedexpr set to either 4 or 5, CtrlField set to "RCC 200-04").

- ➢ Make sure the IRIG source is connected to the IRIG AM input on the TSync (not the IRIG DCLS input)
- ➢ IR_ SetMode 0 0 1 (For manual configuration mode)
- ➢ IR_ SetFormat 0 0 1 (For IRIG B output) (This is the default setting)
- ➢ IR_ SetMod 0 0 1 (For IRIG AM input) (This is the default setting)

Note: IRIG B is always 1kHz input -This is the default setting, and since it can't be changed, no need to configure/verify this value with **IR_SetFreq 0 0 1**

- ➢ IR_ SetCodedExpr 0 0 4 OR IR_ SetCodedExpr 0 0 5 (either value will work)

CE=4 BCD TOY/Year, Ctrl Func, Binary Seconds
CE=5 BCD TOY/Year, Ctrl Fun

- ➢ IR_SetCtrlField 0 0 1 (for RCC 200-04 Control Function).

G) Use "IEEE 1344" Control Field, combined with a "Coded Expression" configuration that contains both Control Field (CF) as well as BCD Year information (IRIG generator also provides 1344 extensions).

Note: This is the recommended configuration to use, when available from the IRIG generator.

Configure the IRIG Generator's IRIG output to include the year information in the "Coded Expression" field. Example "Coded Expressions" include either of the following:

A. **BCD TOY / BCD Year / CF / SBS** OR
B. **BCD TOY / BCD Year / CF**

Configure the IRIG time code generator:

1. Configure the IRIG Reference (SecureSync, another TSync board or a Model 9483 for examples) to provide IEEE 1344 Control Field information. For a Spectracom IRIG generator, in the "**Control Field**" drop-down, select "**IEEE 1344 (C37.118-2005)**".
2. Configure the IRIG output to include the year information in the "Coded Expression" field.

Example “Coded Expressions” include:

A. **BCD TOY** / **BCD Year** / **CF** / **SBS** OR

B**.** **BCD TOY** / **BCD Year** / **CF**

With a Spectracom SecureSync as the IRIG reference:

In the “Coded Expression” field drop-down list, there are four selections that all provide BCD Year information (they are selections “(4)”, “(5)”, “(6)” and “(7)”). Of these four available selections, two of these also include the “Control Field” section (they are selections “(4)” and “(5)”).

If you configure the Coded Expression field in the RIG output to either (4) or “(5)” (with the “Control Field” selection still set to “IEEE 1344), the SecureSync will provide the TSync board with all of the IEEE1344 information and the current year.

3. IRIG AM input (refer to the next section down for DCLS input)

Configure the TSync-PCI’s “AM” IRIG input for IEEE 1344 and BCD Year data:
The TSync-PCIe’s IRIG AM input connector needs to be configured as either **IRIG A/B/G/E124** or **IRIG A/B/G/E125** (IRIG AM, CtrlField set to “IEEE *C37.118-2005*” and Codedexpr set to either selections “4” or “5”)

1) Make sure the IRIG source is connected to the IRIG AM input on the TSync-PCIe board (not the IRIG DCLS input)

2) Enter: **IR_SetMode 0 1 1** (For manual configuration mode)

3) Enter: **IR_SetFormat 0 1 1** (For IRIG B input) (This is the default setting)

4) Enter: **IR_SetMod 0 1 1** (For IRIG AM input) (This is the default setting)

Note: IRIG B is always 1kHz input -This is the default setting, and since it can’t be changed, no need to configure/verify this value with **IR_SetFreq 0 1 1**

5) Enter: **IR_SetCtrlField 0 1 2** (for **“**IEEE C37.118-2005” IEEE 1344 Control Function).

6) Enter: **IR_SetCodedExp 0 1 4** OR **IR_ SetCodedExp 0 1 5** (either value will work).

- **Where CodedExpression 4 =** BCD TOY/Year, Ctrl Func (CF), Straight Binary Seconds (SBS)
- **Where CodedExpression 5 =** BCD TOY/Year, Ctrl Func (CF)

4. IRIG DCLS input (refer to the previous section for AM input)

Configure the TSync-PCI’s IRIG “DCLS” input for IEEE 1344 and BCD Year data:
The TSync-PCIe’s IRIG AM input connector needs to be configured as **IRIG B124** or **IRIG B125** (IRIG B AM, CtrlField set to “IEEE *C37.118-2005*” and Codedexpr set to either selections “4” or “5”)

1) Make sure the IRIG source is connected to the IRIG AM input on the TSync-PCIe board (not the IRIG DCLS input)

2) Enter: **IR_SetMode 0 0 1** (For manual configuration mode)

3) Enter: **IR_SetFormat 0 0 1** (For IRIG B input) (B is the default setting. But it can accept A, B, G or E)

4) Enter: **IR_SetMod 0 0 1** (For IRIG AM input) (This is the default setting)

Note: IRIG B is always 1kHz input -This is the default setting, and since it can’t be changed, no need to configure/verify this value with **IR_SetFreq 0 0 1**

5) Enter: **IR_SetCtrlField 0 0 2** (for **“**IEEE C37.118-2005” IEEE 1344 Control Function).

6) Enter: **IR_SetCodedExp 0 0 4** OR **IR_ SetCodedExp 0 0 5** (either value will work).

Where CodedExpression 4 = BCD TOY/Year, Ctrl Func (CF), Straight Binary Seconds (SBS)

Where CodedExpression 5 = BCD TOY/Year, Ctrl Func (CF)

H) Use a “Coded Expression” configuration that contains both Control Field (CF) as well as BCD Year information (IRIG generator does not include IEEE 1344 extensions)

Note: This is the second best recommended configuration to use, when available from the IRIG generator.

Configure the IRIG output to include the year information in the “Coded Expression” field. Example “Coded Expressions” include either of the following:

A. BCD TOY / BCD Year / CF / SBS

B. BCD TOY / BCD Year / CF

1. IRIG AM input (refer to the next section down for DCLS input)

Configure the TSync-PCI’s “AM” IRIG input for IEEE 1344 and BCD Year data:

The TSync-PCIe’s IRIG AM input connector needs to be configured as IRIG B124 or IRIG B125 (IRIG B AM and Codedexpr set to either selections “4” or “5”)

1) Make sure the IRIG source is connected to the IRIG AM input on the TSync-PCIe board (not the IRIG DCLS input)

2) Enter: IR_SetMode 0 1 1 (For manual configuration mode)

3) Enter: IR_SetFormat 0 1 1 (For IRIG B input) (This is the default setting)

4) Enter: IR_SetMod 0 1 1 (For IRIG AM input) (This is the default setting)

Note: IRIG B is always 1kHz input -This is the default setting, and since it can’t be changed, no need to configure/verify this value with IR_SetFreq 0 1 1

5) Enter: IR_SetCodedExp 0 1 4 OR IR_ SetCodedExp 0 1 5 (either value will work).

Where CodedExpression 4 = BCD TOY/Year, Ctrl Func (CF), Straight Binary Seconds (SBS)

Where CodedExpression 5 = BCD TOY/Year, Ctrl Func (CF)

2. IRIG DCLS input (refer to the previous section for AM input)

Configure the TSync-PCI’s IRIG “DCLS“ input for IEEE 1344 and BCD Year data:

The TSync-PCIe’s IRIG AM input connector needs to be configured as IRIG B124 or IRIG B125 (IRIG B AM, CtrlField set to “IEEE *C37.118-2005*” and Codedexpr set to either selections “4” or “5”)

1) Make sure the IRIG source is connected to the IRIG AM input on the TSync-PCIe board (not the IRIG DCLS input)

2) Enter: **IR_SetMode 0 0 1** (For manual configuration mode)

3) Enter: **IR_SetFormat 0 0 1** (For IRIG B input) (B is the default setting. But it can accept A, B, G or E)

4) Enter: **IR_SetMod 0 0 1** (For IRIG AM input) (This is the default setting)

Note: IRIG B is always 1kHz input -This is the default setting, and since it can't be changed, no need to configure/verify this value with IR_SetFreq 0 0 1

5) Enter: **IR_SetCtrlField 0 0 2** (for "IEEE C37.118-2005" IEEE 1344 Control Function).

6) Enter: **IR_SetCodedExp 0 0 4** OR **IR_ SetCodedExp 0 0 5** (either value will work).

Where CodedExpression 4 = BCD TOY/Year, Ctrl Func (CF), Straight Binary Seconds (SBS)

Where CodedExpression 5 = BCD TOY/Year, Ctrl Func (CF)

I) Use the IRIG input from an earlier Spectracom Model 8183, 9183, 9283 or 9383 Spectracom NetClock's (or a TPRO/TSAT timing board's) IRIG output to set the year in the TSync-PCIe board

When using GPS input, the TSync-PCIe board receives the year from the GPS receiver at each start-up (the year is not stored). When using IRIG input instead, it may be able to read the year value at each start-up if the IRIG generator provides the year in the data stream. In the case of a Spectracom NetClock IRIG output (8183, 9283, 9383, etc), the board needs to be configured each power-up to be able to read the year.

Edited email Dave Lorah sent to a customer:
Since you are using a Spectracom NetClock as the IRIG source, the year information is contained in the control field portion of the IRIG message. So the TSync board can read the year info from the IRIG input. However, the Spectracom IRIG output does not put the year in the conventional place in the IRIG signal. When the products were developed there was no convention for the year info, so it was placed in the control field portion of the message. The Tsync PCIe board has the capability to be set to read the year info from the Spectracom IRIG but it must be told the input is the Spectracom version of IRIG. Also the board must be told to read the year info from the IRIG input.

This can be done by running a batch file from the startup menu on your PC, so every time the PC it started the board gets set to read the Spectracom IRIG year input.

For IRIG AM input- The batch file should run these two commands

IR_SetCodedExp.exe 0 1 0 (sets the IRIG inputs Coded expression)

The TSync-PCIe board supports IRIG inputs of the following coded expressions combinations for BCDTOY, CF, SBS, and BCDYEAR fields: ↓

0 – BCDTOY, CF, SBS (NetClock's IRIG output)
1 – BCDTOY, CF
2 – BCDTOY
3 – BCDTOY, SBS
4 - BCDTOY, BCDYEAR, CF, SBS
5 - BCDTOY, BCDYEAR, CF

IR_SetCtrlField.exe 0 1 3 (sets the IRIG Control field configuration of the IRIG data stream – where the **3** is used to define the Control Field as it is in the NetClock).
If using IRIG DCLS:
.IR_SetCodedExp 0 0 0
.IR_SetCtrlField 0 0 3
These programs are contained in the TSync API in Examples.

J) Use "RCC 200-04" in the IRIG Control Function field

IRIG AM input (refer to the next section down for DCLS input)
If the IRIG source (Such as a NetClock or SecureSync) provides the "RCC 200-04" configuration (in the Control Function field), the TSync-PCIe can read the year value from the IRIG generator.

The TSync-PCIe's IRIG AM input connector needs to be configured as **IRIG B124** or **IRIG B125** (IRIG B AM

Codedexpr set to either 4 or 5, CtrlField set to “RCC 200-04”).

1) Make sure the IRIG source is connected to the IRIG AM input on the TSync (not the IRIG DCLS input)
2) **IR_ SetMode 0 1 1** (For manual configuration mode)
3) **IR_ SetFormat 0 1 1** (For IRIG B output) (This is the default setting)
4) **IR_ SetMod 0 1 1** (For IRIG AM input) (This is the default setting)

   Note: IRIG B is always 1kHz input -This is the default setting, and since it can’t be changed, no need to configure/verify this value with **IR_SetFreq0 1 1**

5) **IR_ SetCodedExpr 0 1 4** OR **IR_ SetCodedExpr 0 1 5** (either value will work)

   CE=4 BCD TOY/Year, Ctrl Func, Binary Seconds
   CE=5 BCD TOY/Year, Ctrl Fun

6) **IR_SetCtrlField 0 1 1** (for RCC 200-04 Control Function).

IRIG DCLS input (refer to the section above for IRIG AM input)

If the IRIG source (Such as a NetClock or SecureSync) provides the “RCC 200-04” configuration (in the Control Function field), the TSync-PCIe can read the year value from the IRIG generator.

The TSync-PCIe’s IRIG AM input connector needs to be configured as **IRIG B124** or **IRIG B125** (IRIG B AM Codedexpr set to either 4 or 5, CtrlField set to “RCC 200-04”).

1) Make sure the IRIG source is connected to the IRIG AM input on the TSync (not the IRIG DCLS input)
2) **IR_ SetMode 0 0 1** (For manual configuration mode)
3) **IR_ SetFormat 0 0 1** (For IRIG B output) (This is the default setting)
4) **IR_ SetMod 0 0 1** (For IRIG AM input) (This is the default setting)

   Note: IRIG B is always 1kHz input -This is the default setting, and since it can’t be changed, no need to configure/verify this value with **IR_SetFreq 0 0 1**

5) **IR_ SetCodedExpr 0 0 4** OR **IR_ SetCodedExpr 0 0 5** (either value will work)

   CE=4 BCD TOY/Year, Ctrl Func, Binary Seconds
   CE=5 BCD TOY/Year, Ctrl Fun

6) **IR_SetCtrlField 0 0 1** (for RCC 200-04 Control Function).

Desire to receive the 1344 extensions (IEEE C37.118-2005) from the IRIG input's Control Function field

- For details on IEEE-1344, refer to the "**IEEE-1344 extensions (IEEE C37.118-2005**)" section in the Custserviceassist document

(below info is from: http://www.cyber-sciences.com/documents/TN-102_IRIG-B.pdf)
Year information was not specified in the IRIG standard prior to its 2004 revision. Before 2004, the IEEE adopted a standard (IEEE-1344) which included year data as part of the IRIG-B signal. This variation came to be known as "IEEE-1344 extensions."

IEEE-1344 extensions use extra bits of the Control Functions (CF) portion of the IRIG-B time code. Within this portion of the time code, bits are designated for additional features, including:

Calendar Year (now called **BCDYEAR**)

Leap seconds, and leap seconds pending

Daylight Saving Time (DST), and DST pending

Local time offset

Time quality

Parity

Position identifiers

If it's desired to pass the IEEE 1344 data (located in the Control Function field), from the IRIG source (such as a SecureSync with an IRIG output Option Card installed) to the TSync-PCIe board, the TSync-PCIe board needs to be programmed to read this data.

FYI -IEE 1344 includes leap second, DST/time zone, and time quality information. For more info on IEEE 1344, refer to: IEEE-1344 (IEEE C37.118-2005)

Below is the specific info to sync a TSync board to a SecureSync configured to output from a SecureSync either "B124" or "B125" with IEEE-1344 extensions included from a SecureSync:

Note: Refer to the "IRIG" Section of the SecureSync Option Cars assist document if the IRIG signal is being generated by a SecureSync: ..\SecureSync Option Card information.pdf

The TSync-PCIe's IRIG AM input connector needs to be configured as either **IRIG B124** or **IRIG B125** (IRIG B AM 1kHz, CtrlField set to 2 and Codedexpr set to either 4 or 5).

1) Make sure the IRIG source is connected to the IRIG AM input on the TSync (not the IRIG DCLS input)
2) **IR_ SetMode 0 1 1** (For manual configuration mode)
3) **IR_ SetFormat 0 1 1** (For IRIG B output) (This is the default setting)
4) **IR_ SetMod 0 1 1** (For IRIG AM input) (This is the default setting)

   Note: IRIG B is always 1kHz input -This is the default setting, and since it can't be changed, no need to configure/verify this value with **IR_SetFreq0 1 1**
5) **IR_ SetCodedExpr 0 1 4** OR **IR_ SetCodedExpr 0 1 5** (either value will work, as indicted below):

   CE=4: BCD TOY/Year, Ctrl Func, Binary Seconds
   CE=5: BCD TOY/Year, Ctrl Func
6) **IR_SetCtrlField 0 1 2** (for **"**IEEE C37.118-2005" 1344 extensions).

7) Verify the green Sync LED on the TSync board illuminates

Note: The year value will still need to be set using either the API call/example program, or with the Windows Control Utility. The IEEE 1344 bits do not contain year information.

**Troubleshooting IRIG input synchronization

First, as “no problems found” is not covered under Spectracom warranty, and as there is necessary re-configuration of the TSync-PCIe board after each power-up and other external factors can also prevent it from being able to sync to IRIG input, I have a couple of basic questions for you:

1) Do you happen to have more than one TSync-PCIe board at this site? If you do, have you already tried swapping it out with another one to ensure the reported symptom follows this board? This will also confirm the issue is not due to a problem with the IRIG generator or the associated IRIG cable.

2) If you have swapped it out with another board, do you have application software that automatically reconfigured both the trial board and this board after power-up. This will confirm the IRIG input configuration parameters are set correctly to match the settings of the IRIG generator’s output signal.

3) Was this TSync-PCIe board syncing to IRIG input at one time, but has since stopped syncing? Or is this a new installation of the TSync-PCIe board and it hasn’t synced-up yet? If it’s a new install, we just need to ensure the IRIG input configurations match the settings of the IRIG generator.

   There is no need to respond back to me with the answers to the above questions. They are only meant to act as a “trigger” for troubleshooting steps that you may not have tried yet. If these recommended troubleshooting steps have already been performed, please send it back to us for a closer evaluation.

**TSYNC_DRV_CONNECTION_ERR with “IR” calls

Error: connection error (TSYNC_DRV_CONNECTION_ERR).

- This error message may occur because:
  1) The firmware is unable to communicate with the FPGA / Micro
  2) There is an issue with the EEPROM (such as corrupted during a bad upgrade). The EEPROM stores configuration info for the board, such as if a GPS receiver is installed.
- Try performing other API calls that go through the FPGA (any calls except the HW_Gettime call). A good example to try us the **CS_GetTime 0 0** call. If this call works, likely an EEPROM issue (get the version info for the board). If the other calls also fail, could be an issue with the FPGA/Micro .

Error: opt err (RC_STUB_FUNC)

- Occurs if the IRIG Format is currently set to a Format (such as B) and the IRIG Frequency is being changed (via **IR_SetFreq)** to an invalid selection for that particular Format (such as trying to set it to 100 Hz when with its currently set to Format B, which is always 1000 Hz only)
- Refer to Salesforce case 19282

```
[root@lab-r46-05 tsync]# ./examples/IR_SetFreq 0 0 1
 Error: opt err (RC_STUB_FUNC).
  IR (0) Freq: (1) 100 Hz

[root@lab-r46-05 tsync]# ./examples/IR_SetFreq 0 0 2

   IR (0) Freq: (2) 1 kHz
  Error: opt err (RC_STUB_FUNC).
```

**UT1 correction factor (leap seconds)

Leap Seconds (UT1 correction /TAI offset)

UT1 correction is the number of leap seconds that have occurred since GPS went online. This value is updated by the GPS receiver after the information has been obtained from GPS ephemeris data download (it can take up to about 13 minutes for it to be obtained from GPS).

To read the current UTC offset, use the **CS_GetTimeScaleOff 0 0** call (where the second 0 designates the UTC timescale)

Where:

UTC=0 (power-up default)
TAI=1
GPS=2
Local=3

The **CS_GetTimeScaleOff** call/example can also be used to read the offset for TAI and local, as well.
For TAI offset, use the **CS_GetTimeScaleOff 1** call (where the 1 designates the TAI timescale)

***Leap Smear (Smearing the leap second instread of a one second instananious correction)

- Refer to "**Leap Smear"** section of the custserviceassist document for more infor

External 1PPS" Input Reference (Self/EPP0 and Hst0/EPP0)

Accuracies when using 1PPS input from a source other than the TSync's GPS receiver.

- **Per Dave Sohn (Sept 2019)** "we cannot spec 1PPS accuracies of a TSync board not syncing to its own GNSS/GPS receiver, as the 1PPS source's accuracy and jitter are not known."

Email Keith sent to Acquisys (9 Sept 2019) Thanks for your great inquiry about syncing a TSync-PCIe board using an external 1PPS (Referred to as the "epp 0" input reference). My apologies for the delay in being able to get back to you, as I needed to discuss your email with the TSync board's Product Manager, here in the US.
Unlike when using a GPS receiver to sync a TSync-PCIe board, unfortunately we are not able to provide any typical accuracy specs for a 1PPSbeing received from another source. With a GNSS/GPS receiver's 1PPS output to the TSync-PCIe board, we know the accuracy and jitter characteristics of that 1PPS input. But we don't know what the 1PPS jitter characteristics are, when the 1PPS input signal is being applied by a customer's 1PPS generator. The 1PPS jitter of the signal is a large factor in how closely aligned the TSync-PCIe board's outputs will be, to the incoming 1PPS signal

Please keep in mind that the TSync-PCIe board does not try to "keep up and match" any the jitter of the incoming 1PPS signal. Instead, the TSync board's own internal 1PPS is "de-coupled" from the input PPS. The TSync-PCIe board only coarse adjusts to the input 1PPS signal the first time it's applied after each boot-up of the TSync board – after that, the system 1PPS is very slowly slewed into alignment with the input PPS. If there is a lot of jitter on the input PPS, the TSync-PCIe's internal 1PPS is not going to have similar jitter, because of this very slow disciplining. This helps significantly dampen out excessive jitter on the input PPS from also being observed on the system PPS. With jitter on the input PPS, and the very slow disciplining, the result will be larger offsets between the input 1PPS and the internal system PPS. But with very little jitter on the input PPS, the internal PPS will be better able to be more closely aligned to the input PPS.

The Product Manager did mention that if your customer's 1PPS source's 1PPS signal has similar (or better) accuracy and as low (or even less) jitter than observed with the TSync's own GNSS receiver, the accuracy specs provided in the TSync's datasheet (attached) will be similar to those that your customer can expect to see when using their own supplied 1PPS (the typical accuracy/jitter with the GPS receiver's 1PPS is +/- 25ns). But if the received 1PPS signal has less accuracy/more jitter than the 1PPS outputted from the GPS receiver, the accuracies your customer can expect to see will be inherently less than those provided in the datasheet, depending on the level of jitter on the 1PPS signal.

Connection of external 1PPS input to the TSync board

- Connect the external 1PPS input **to Pin 7 of the Timing Connector** (**1PPS input pin**) or to the standard/premium breakout cable.

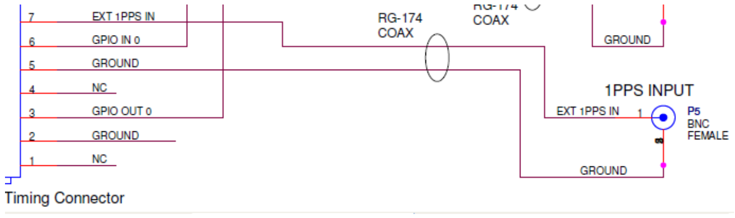
Note: ECN 2638 (released 5/2/11) added a BNC T connector and 50 ohm terminator to TSync-PCIe ancillary kit to terminate the 1PPS input ("Exception to this is TSync-PCIe with SP364)

- **P/N 1191-0000-0710**

Important note: When applying an external 1PPS input to TSync, it is important to observe proper termination of the 1PPS signal at the input to the TSync-PCIe board. A 50 ohm termination is likely needed to be added to the 1PPS input. This requirement is based on the requirements of the 1PPS source equipment. If the 1PPS source expects 50 ohm output termination, a 50 ohm termination needs to be added. If the source expects high impedance output termination, do not add any termination and connect the 1PPS directly to the timing board. Failure to properly terminate the signal will likely result in a ringing of the 1PPS, which will cause the TFOM values to be significantly higher than expected (like TFOMs of 7 or 8, versus expected 2 or 3).

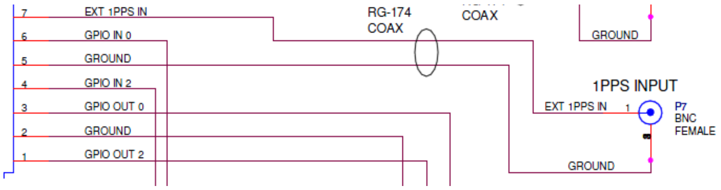
Note: the actual pin-out depends on which breakout cable is used)

1) **Using Standard Breakout cable (BNC Connector P5)**



1PPS Signal	Connector P5 - pin	Goes to Pin on Timing connector
Ext 1PPS In	Center pin	7
Ground	Shield	5

2) **Using Premium Breakout cable (BNC Connector P7)**



1PPS Signal	Connector P7 - pin	Goes to Pin on Timing connector
Ext 1PPS In	Center pin	7
Ground	Shield	5

1191-0000-0710: 50 ohm termination

Per Dave Sohn: The 50 ohm termination should be present on the 1PPS for proper operation with a 50ohm 1PPS source.”

P/N 1191-0000-0710 (BNC T connector and 50 ohm terminator for 1PPS input)

- Both parts are included in a bag with each TSync board

  (**in Arena**): https://app.bom.com/supplier-items/detail-spec?item_id=1218663225

  (**In Salesforce**): https://orolia.my.salesforce.com/01t1A000004SUKm?srPos=2&srKp=01t

Our P/N: 8140-1000-1001
AMPHENOL P/N: 46650-51RFX

Our P/N: CP00000
AMPHENOL P/N: 31-208-RFX

Self/EPP0 input Reference versus hst0/EPP0 input Reference

A) Self/EPP0

- Self/EPP0 will sync the TSync board as long as the external 1PPS input is valid
  - No calls need to be entered/no user interface required to go into sync
- TSync board remains in full sync (not Holdover mode), as long as external 1PPS input remains valid
- TSync board will go into Holdover mode if external PPS is lost or declared not valid/
- If Time sync is lost (either holdover mode expires before 1PPS is restored, or the board is rebooted, the goes the TSync board go right back into Sync (with no commands/ user interaction needed) as soon as 1PPS input is valid again

Desire to change the current System Date/Time

CS_Settime 0 0 2012 123 12 12 12 <enter> (year is 2012, 123rd day of the year, hours 12, minutes 12 seconds <enter>

Notes

1) The self/self or self/epp0 reference needs to be entered into the reference priority table (and the table needs to be saved via a call, if either of these were just entered into the table) before the HR_SetTime (or "CS_SetTime"??) call above will make the "Time" valid.

   Note the two "RS" calls persist thru all reboots

   RS_AddEntry 0 1 1 self epp0 <enter>
   RS_SaveUserDef <enter>

B) Hst0/EPP0

- With Hst0/EPP0 input reference, TSync board will not go into sync, until external 1PPS is valid AND the **HR_SetTime <device index> <index> <year> <doy> <hour> <minute> <second> <nanoseconds>** call has been performed.
- Once the HR_settime call above has been performed (with a valid 1PPS input) the TSync will then go into Sync
- TSync remains in full Sync (not Holdover mode), as long as the external 1PPS remains valid
- TSync board will go into Holdover mode if external PPS is lost or declared not valid
- If Time Sync is lost (either holdover expires before 1PPS is restored, or board is rebooted, the board **will not** go back in goes right back into Sync again, until the HR_SetTime call mentioned above is performed again/

Notes

1) The self/self or self/epp0 reference needs to be entered into the reference priority table (and the table needs to be saved via a call, if either of these were just entered into the table) before the HR_SetTime (or "CS_SetTime"??) call above will make the "Time" valid.

   Note the two "RS" calls persist thru all reboots

   RS_AddEntry 0 1 1 self epp0 <enter>
   RS_SaveUserDef <enter>

Add 1PPS input "EPP0" to reference priority table

Note for the 50 om terminator referenced below

- P/N **1191-0000-0710** (BNC T connector and 50 ohm terminator for 1PPS input)
- Included with each TSync

Email from Keith to a customer
As mentioned above, the TSync board can be synced to an external 1PPS source, if desired (just keep in mind, it also needs a Time reference, which can be an external time source (such as GPS, for instance) or it can "sync to itself", if desired. When it's desired to sync the board using an external 1PPS source, the 1PPS needs to connect to the 1PPS input pin (pin 7) of the Timing connector. This can be via the supplied Standard Breakout cable or via the available Premium Breakout cable. The 1PPS input should be terminated at the breakout cable with a 50 ohm terminator to prevent reflections being generated inside the cable.

When using the Standard breakout cable, the 1PPS input should be connected to the BNC connector labeled ("P5"). Or connector (P7) when using the purchased Premium Breakout cable.

External PPS input reference is called "EPP0". "EPP0 needs" to be paired with a Time reference (such as GPS, IRIG or Self) in the Reference Priority table so that the TSync board can achieve sync status. Attached is a copy of the TSync driver Guide. Section 4.2.24 (starting on page 4-218) of this guide provides information on the API calls used to configure the Input Reference priority table (the User table) with Time and PPS references. The Factory table is the default table, the Working table is useable while the board is powered up but doesn't persist a reboot and which becomes the User table once it's saved. The User table is loaded at each reboot, once it has been created.

Modified Email from Dave Sohn
The default priority table doesn't include an entry for the external 1PPS without a time reference. The easiest solution is to add an entry into the priority table and then save that table, so it will always start up with it. To add an entry, they can use the **RS_AddEntry API** with the following parameters **RS_AddEntry 0 1 1 self epp0**, which has the following meanings:

0 = card instance 0
1 = enable entry
1 = priority level 1 (top priority)
Self = the card can use its current time as a reference for time (this doesn't preclude updating the time from the host)
del = external 1PPS is the frequency reference.

They can then save the priority table by using the **RS_SaveUserDef <enter>** API call.

To determine validity status of all input references, you can use the "RS_GetStateTable.exe 0" example program included in the TSync driver. Look in the table for the entry for epp0 and see if it is considered valid by the TSync board. If it's not, you might need to look at the waveform of your 1PPS input using an oscilloscope to see what the signal actually looks like. Is the 1PPS source expecting a 50 ohm termination load? In that case it may need to be terminated as the board's input impedance is probably too high and may cause the board to see excessive ringing. Placing a 50 ohm load at the input may "clean up the signal".

From a "hardware perspective"- The 1PPS input is normally supplied to the TSync board via a breakout cable (octopus type cable assembly). The 1PPS reference should be connected to the breakout cable assembly on the"leg" that is labeled "1PPS input" (P5) (or P7 of the Premium breakout cable). To ensure there is 1PPS input is present to the TSync board, if an oscilloscope is available, make sure there is a 1PPS input signal from the 1PPS source to the breakout cable assembly (as measured at the breakout cable end of the cable coming from the 1PPS reference). Then, disconnect the breakout cable from the TSync board and make sure there is a 1PPS going through the breakout cable leg to the DB25 connector (the connector that plugs into the TSync board's "timing" connector). The external 1PPS input should be present on pin 7 of this connector.

As a temporary test, the breakout cable can even be bypassed altogether by connecting the 1PPS directly to pin 7 of the timing connector on the TSync board (ground is on pins 5, 8, 18, 23, etc). Refer to page 3-1 of the attached user manual for a picture of the connector. Note that pin 1 is the upper left pin on the DB25 connector and pin 25 is the lower-right pin.

Note: After first oscillator correction, the oscillator is then slewed to its input. The slew rate is very slow (~100ns per second).

(EPP0) External 1PPS input validity (PR_GetValidity 0 0)

PR_GetValidity 0 0: Verifies if (EPP 0) external 1PPS input is present and valid – "1"   OR   If input is not valid –"0")

O=False (not valid)
1-True (valid)

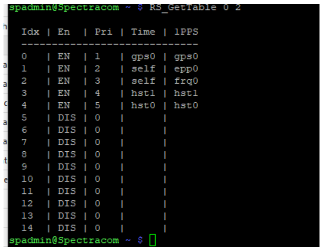
If the TSync is not declaring sync after 1PPS input is applied, I recommend you look at the "validity" of the 1PPS input reference, once it's been reconnected. The API call to confirm the TSync board is declaring the 1PPS input valid is **PR_GetValidity 0 0**. This command will respond with a Time and 1PPS validity. With the IRIG signal connected, both values should be "1". If either or both of these values is a "0", there is an issue with the IRIG signal that is preventing the TSync board from declaring the IRIG input as valid.

If the 1PPS input is connected and the 1PPS signal is present, but it's still being declared not valid, there may be a 1PPS signal level or termination issue. Improper termination of the 1PPS input can cause intermittent or continuous ringing of the 1PPS. This can result in the 1PPS signal being declared not valid. If the ***1PPS generator is expecting 50 ohm termination, as the TSync-PCIe board has a high impedance input, a 50 ohm termination needs to be added to the TSync's 1PPS input to prevent potential ringing of the signal.

Troubleshooting EPP0 (external 1PPS input)

A) TSync-PCIe board not going into sync, even though external 1PPS input is connected

- Check the validity of the external 1PPS input (**PR_GetValidity 0 0**)
- See if the MaxTfom value (**SS_GetMaxTfom 0**) is exceeding TFOM (**SS_GetTfom 0**)
- Verify Self/EPP0 or hst0/EPP0 refernce is in Reference Priority table and is enabled (**RS_GetTable 2**)



```
spadmin@Spectracom ~ $ RS_GetTable 0 2

 Idx | En  | Pri | Time | 1PPS
 -----------------------------
 0   | EN  | 1   | gps0 | gps0
 1   | EN  | 2   | self | epp0
 2   | EN  | 3   | self | frq0
 3   | EN  | 4   | hst1 | hst1
 4   | EN  | 5   | hst0 | hst0
 5   | DIS | 0   |      |
 6   | DIS | 0   |      |
 7   | DIS | 0   |      |
 8   | DIS | 0   |      |
 9   | DIS | 0   |      |
 10  | DIS | 0   |      |
 11  | DIS | 0   |      |
 12  | DIS | 0   |      |
 13  | DIS | 0   |      |
 14  | DIS | 0   |      |
spadmin@Spectracom ~ $ 
```

B) High/jittery 1PPS Phase Errors

- Add 50 ohm terminator to the 1PPS input (only at the TSync board end of the PPS cable)

Oscillators: Osc cal and disciplining to PPS ref / TFOM / Holdover

API calls associated with the oscillator

- Refer to Tim Tetreault's "cheat sheet" at: ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\Tsync driver calls cheat sheet
- These are examples - additional calls may have since been added

Oscillator Commands
XO_GetAlarm
XO_GetCalVal
XO_GetDac
XO_GetFreqError
XO_GetMfrMdl
XO_GetMode
XO_GetOscType
XO_GetPhaseError
XO_GetSerialNo
XO_GetState
XO_SetDac
XO_SetMode

Oscillator Monitor Commands
XS_GetMeterData
XS_GetWindowSize
XS_Register
XS_SetMeterCmd
XS_SetWindowSize
XS_GetMeterData
XS_GetWindowSize

** Oscillator calibration (recalibration) / HR (Hertz range)

- Calibrated once at the factory and then stored in EEPROM (not erased with a Clean).
- API call/example program to read the current cal value: **XO_GetCalVal 0** (requires linux driver version 3.1.0 or higher be installed).

Typical/expected cal values for TCXO or OCXO

Email from Tim T (22 Apr 17) So I did a quick test of some TSync-PCIe board that I have.

- Board with **TCXO** had cal value of **0.0032** Hz per DAC step
- Boards with **OCXO** had cal value of **0.00032** Hz per DAC step

(per email below, the range for OCXO should be: **0.00015 Hz < X > 0.001 Hz**)

Email from Dave S (10 May 17) The calibration value reported from the **TSync OCXO** boards should **bgreater than 0.00015 Hz per DAC step** and **less than 0.001** Hz per DAC step. The value should never be negative for the TSync OCXO boards.

TCXO/OCXO oscillator recalibration in the field (BAD/questionable oscillator calibration/XO_SetMode 0 3 call/example program)

- Refer to the Word document in the folder: EQUIPMENT\SPECTRACOM EQUIPMENT\SecureSync\10 MHz Oscillators\Recalibrate a TCXO or OCXO osc
- **Larrge batch of TSync boards which potentially have bad cal values due to production test issue (discovered May, 2017**) (Note The ECO that rev'd the TSync's firmware and broke the test script was released on 03/24/2016) : S/Ns S/N **4579-5829**
- It was confirmed that this test issue DID NOT affect SecureSyncs. Securesyncs should all be cal'd correctly,

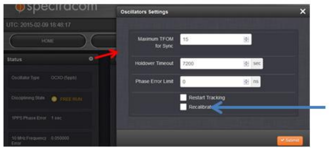
A) Earlier firmware/driver versions (prior to the "XO_SetMode 3" API call/example program being available)

- For TCXO/OCXO oscillators only
- If a bad occurs requires an EEPROM patch update file (patch_img.bin) be uploaded by a customer, Then we need to remote ssh in to issue a command that applies the patch.
- A patch is required to reset HR back to "0", so that another oscillator calibration can be automatically performed the next time a reference is applied.
- *LS_Message 0* command will display the HR value when its performed, but when its read out with this command or the board is power cycled/rebooted, there is no way to retrieve it again (there is no API call for this value).

C) "XO_SetMode 0 3" API call /example program to recalibrate TCXO/OCXO

- **"XO_SetMode 0 3"** is the same call added to SecureSync update 5.2.0 for the "**Recalibrate**" checkbox to the newer browser.
- Refer to Mantis case 1731.

Just FYI: This checkbox (**Management** -> **Disciplining** page of the browser) allows the oscillator to be recalibrated by a customer without the need to run a patch or login as spfactory mode.



spectracom
UTC: 2015-02-09 18:48:17
HOME
Status
Oscillator Type OCXO (5ppb)
Disciplining State FREE RUN
1PPS Phase Error 1 sec
10 MHz Frequency Error 0.050000
Submit

Oscillators Settings

Maximum TFOM for Sync: 15

Holdover Timeout: 7200 sec

Phase Error Limit: 0 ns

□ Restart Tracking

□ Recalibrate

If a bad oscillator calibration happens to occurs, it won't run the cal again: Can perform the **"XO_SetMode 3"** call/example program to reset the calValue.

Important Note: Before performing recal, verify in full sync (not in Holdover) and the unit is synced to a very stable refernce, such as GPS or a very stable IRIG input **(SS_getSync 0 true and SS_GetHoldover 0 false)**

Email from Tim T 4/24/17) Keith, I have used that command in the past to we know it works. The thing you need to make clear to Google is the board needs to be connected to GPS or a good 1pps reference before running the command. After they have run the command, they need to wait about 5 to 10 min. Then they can read the cal value to see if it has changed.

This is only to test out to see if the board was never cal in the first place. This is a factory cal and we don't want Google to start running this command on all of the boards all the time.

My corresponding response to Google

Based on the symptoms of the increasing large phase errors and especially due to the cal values observed, we are suspecting a potential Factory oscillator calibration-related condition may be occurring with these particular boards.

As a test of one of the TSync-PCIe boards exhibiting the increasing phase error, can you try recalibrating its oscillator using the procedure below:

1) Connect either GPS or an external 1PPS input. After confirming the TSync board is synced to the input reference, wait a few minutes longer (having GPS sync or a very stable 1PPS input is important for the oscillator cal).

2) Confirm the current calibration value for this board **(XO_GetCalVal 0)**

3) Issue the recalibrate oscillator command of **XO_SetMode 0 3** (0 for the board and 3 for the recalibrate mode).

4) Wait about 10 minutes or so for the calibration to be performed.

5) Obtain the new calibration value (**XO_GetCalVal 0**). Please let us know if it has changed from the previous and what this value becomes.

6) While leaving the external reference connected, see if the phase error continues to increase (as you have been observing). Or, does it now stay a fairly consistent, expected lower value?

Oscillator API calls

Tsync_XO_GetCalVal: New API command implemented to get the calibration value of the oscillator.

Note: this command requires TSync Linux driver version 3.1.0 (or higher) be installed.

Note: to determine type of osc installed: **XO_GetOscType 0**

A) OCXO

1) **XO_GetCalVal**", such as **XO_GetCalVal 0**

This value shouldn't ever be the following: +0.000000 (This indicates a bad calibration occurred)

Example response from a SecureSync with an OCXO (21 Apr 17):

XO_GetCalVal 0: "**0.000135 Hz per DAC step**"

Typical/expected cal value for OCXOs is around: +/- .00013xxxx to +/- .00015xxxx

B) TCXO

1) XO_GetCalVal", such as **XO_GetCalVal 0** and **XO_GetCalVal 1**.

This value shouldn't ever be the following: +0.000000 (This indicates a bad calibration occurred)

Example response from same customer above with two TSync boards having TCXO installed (11 Apr 17):

- **XO_GetCalVal 0**: 0.000366822
- **XO_GetCalVal 1**: 0.000358157

Note*: Tim T confirmed these both look fine for tcxo step sizes*

Typical/expected cal value for OCXOs is around: +/- .00013xxxx to +/- .00015xxxx

XO_GetSerialNo

(KW 9/18/12, per Tim Tetreault 9/18/12) This is a new command that was added to the TSync driver specifically for the TSync-cPCI boards. It's not currently supported with the TSync-PCIe timing boards.

Determining Oscillator lock status via API call/example program

TSYNC_SS_GetFreeRun: Gets the current freerun state.

Oscillator warm-up time (time to warm up to operating temperature)

C) OCXO oscillator

Per Dave Sohn (19 Nov 14) "The OCXO warm-up period for the TSync-PCIe is 60 seconds maximum".

D) TCXO oscillator

- Not applicable. TCXO oscillator is not an ovenized oscillator

---

Allan Deviation (oscillator's frequency stability)

Refer to "Allan deviation in the custserviceassistance doc.

Oscillator disciplining

Purpose: Aligns the TSync's System 1PPS (on-time point) to the input 1PPS input reference.

Slew rate

After initial oscillator correction after power-up, the oscillator is then slewed to its input. The slew rate is very slow (~100ns per second).

Based on a slew rate of 4 us/sec, this equates to:

Per hour: 14,400 us/hour (14 ms/hour)
Per day: 345,600 us/day (345 ms/day)

Note: CLI command to determine oscillator type installed: XO_GetOscType 0 <enter>

Disciplining modes (XO_GetMode call/example program)

XO_GetMode 0

0= Disciplining

1= Test (for Engineering use)

2= Reset Tracking (should be implemented in June 2012 release) Used to "coarse adjust" 1PPS to the reference, in case the PPS reference has been lost for a while and its desired not to wait for the PPS to slew back to the restored reference.

Disciplining Lock States (XO_GetState call/example program)

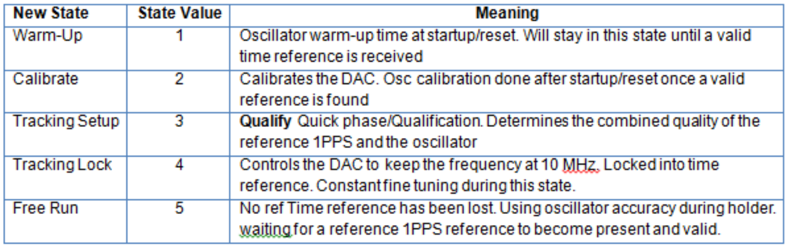
For more info, refer to S:\Engineering\Projects\Kramden\KTS_Project Development\200 Engineering Documents\System Architecture **(Important Note**: For in-house use only. Do not release to customers).

- The “**Lock**” state is the value that indicates the oscillator is locked to the GPS 1PPS input reference
- **Example program: XO_GetState 0 (API call: XO_GetDiscState 0**)

This is the best call/example program to determine if the 10MHz output of a TCXO/OCXO oscillator is phase locked to GPS.

Responses from **xo_Getdiscstate** API call/ **xo_Getstate** example program
NONE **=** 0,

Disciplining States (Oscillator disciplining Algorithm modes):



New State	State Value	Meaning
Warm-Up	1	Oscillator warm-up time at startup/reset. Will stay in this state until a valid time reference is received
Calibrate	2	Calibrates the DAC. Osc calibration done after startup/reset once a valid reference is found
Tracking Setup	3	**Qualify** Quick phase/Qualification. Determines the combined quality of the reference 1PPS and the oscillator
Tracking Lock	4	Controls the DAC to keep the frequency at 10 MHz. Locked into time reference. Constant fine tuning during this state.
Free Run	5	No ref Time reference has been lost. Using oscillator accuracy during holder. waiting for a reference 1PPS reference to become present and valid.

NOREF **=** 6 (**Note**: State 6 was removed sometime prior to firmware version 2.2.1)

The center for the OCXO DAC is 0x8000 so your values show that it is nowhere near the limits. The DAC values will continue to change as the unit fine tunes itself to the external reference. I would like you to run a different command that tells us different states the unit is in.The command is “XO_GetState 0”

firmware version changes associated with disciplining

- **firmware version 3.4.6**: “Oscillator disciplining improvements”

Phase error measurements

XO_GetPhaseError

Phase error can be read using **XO_GetPhaseError** API call or example program to see if on-time point is getting close to the input reference (should continue to decrease as it gets closer).

Q. The XO_GetPhaseError returns a value in Degrees. How will this convert to a time value?

A reply from Dave Lorah (26 Jun 14) I was a little puzzled about this question as I remember the Phase Error was reported in nS not degrees. As it had been some time since I ran the command we ran the command today and the board reported the value in nS. You can also run the XO_GetFreqError 0 to get the frequency error of the oscillator in Hz.

I discussed this with our driver engineer and he said to check the firmware of the board and also the driver rev to make sure they are up to date.

Use the FS_GetVersion 0 0 to check the board firmware is at current version 2.11. Use the modinfo tsyncpci to check the driver version is current 3.12.

If these are correct the XO_GetPhaseError 0 will report the value in seconds.

Monitoring the performance of the oscillator

Regarding your question on monitoring the performance of the TSync's 10 MHz oscillator, there are a couple of ways to perform this. Two are internal to the timing board, using an API call and one is external to the timing board, using external test equipment. Internal to the TSync-PCIe board, there are two API calls that can be used to monitor the oscillator. The **XO_GetFreqErr** and **XO_GetPhaseError** API calls will provide a measured frequency and phase error of the 10 MHz oscillator.

The **SS_GetTfom** API call/example program allows the oscillator disciplining of the TSync-PCIe's 1PPS on-time point (which the board's outputs are based on) in comparison to the board's 1PPS input reference (such as IRIG or GPS). This comparison of how well the oscillator is disciplined to the 1PPS input reference is in known as the TFOM value (Time Frequency of Merit). The reported TFOM value will indicate how closely aligned the oscillator disciplining is able to keep the board's on-time point aligned to its input reference. If the input reference is lost (known as the holdover mode), the TFOM value is incremented accordingly, based on the inherent drift of the oscillator.

Below is additional information about TFOM:
The Time Figure of Merit (TFOM) enumeration is defined to provide a dimensionless quality measure of the overall time/1PPS synchronization based on an Estimated Time Error (ETE). This measure is used for overall sync status of our clock to the internal 1PPS. Our synchronization definition is derived from our use of time and 1PPS to achieve both synchronization of time and synchronization of frequency.

The TFOM value is a very accurate comparison (Estimate- not a measurement) of the input reference 1PPS versus the 1PPS output from the oscillator disciplining circuit (which uses the input 1PPS as its reference). As the TSync-PCIe board is able to discipline itself to a very stable and accurate 1PPS, the TFOM values will decrease. However, if the input reference 1PPS is jittery or does not have a "crisp" rising edge that it can definitively define as the trigger point, each time the TFOM is calculated, the value is going to be higher than expected.

The oscillator can also be monitored external to the TSync-PCIe using a frequency counter. Spectracom offers frequency counters, such as the CNT90 and CNT-91 that can very accurately monitor the output frequency of the oscillator. Pin 22 of the Timing connector is a 10 MHZ output that can be connected to a frequency counter to indicate the frequency of the 10 MHz oscillator (The standard breakout cable provided with the timing board does not provide a connection to this pin, but the available Premium breakout cable does provide a connection to this pin to allow for monitoring of the oscillator).

When the TSync board is closely aligned with its 1PPS input reference, the TFOM and Phase Errors will be lower numbers (these numbers are oscillator-type dependent), Typically TFOM will be 4 or less and the Phase error will be like 200ns or less when in sync with the input PPS reference. The command to read the current TFOM value is **SS_GetTFOM 0 and the command to read the current phase error is XO_GetPhaseError 0.**

SS_GetTfom

Q. How does the TSync-PCIe board calculate the TFOM Value?

A. **reply from Keith (26 Jun 14)** The TSync board uses the calculated phase error value as the input for the TFOM Calculation. This data is collected in a variable size window and the result compared to a table of set values to select the TFOM value.
The variable min and max measurement window gets complex. It is set depending on the accuracy of the reference source, the stability of the data and the type of oscillator used.
The TFOM is an estimate of the timing error of the board and meant to be used as an indicator of the general performance of the timing;

The Time Figure of Merit (TFOM) enumeration is defined to provide a dimensionless quality measure of the overall time/1PPS synchronization based on an Estimated Time Error (ETE). This measure is used for overall sync status of our clock to the internal 1PPS. Our synchronization definition is derived from our use of time and 1PPS to achieve both synchronization of time and synchronization of frequency.

Note about "tfom" value (can result in an "incorrect" tfom value being reported)

- The GR_GetFixData call/example program also contains a "**fom**" and "**tfom"** value in its response (these two fields are always 0 with a Trimble receiver installed, or apparently a "2" or "5" when a ublox receiver is installed. The "tfom" value in this other call is associated only with the receiver itself, and not associated in any way with the TSync board.
- At least one customer was found to be using the tfom value from GR_GetFixData instead of the response of SS_GetTfom, resulting in him seeing TFOM of "0" with a Trimble receiver, and a "2" or "5" with a ublox receiver.

Smart Reference Monitoring/Phase Threshold (firmware v3.4.7 and above only)

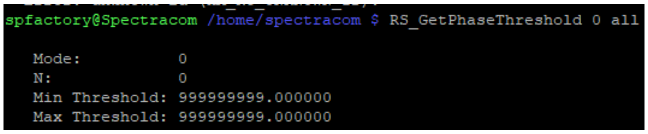
- New capability added in firmware update version 3.4.7 (ECO 1625 released 29 March 2018)
  - From 3.4.7 Release Notes: "Reference Monitoring is added and can be accessed via the TSYNC Driver APIs"
- Refer to "**Smart Reference Monitor**" section of the SecureSync tech note for more details: I:\Customer Service\1- Cust Assist documents\SecureSync CustAssist.pdf
- Provides ability to disqualify individual input references based on their "quality" instead of simply present/not present.

Associated Calls:

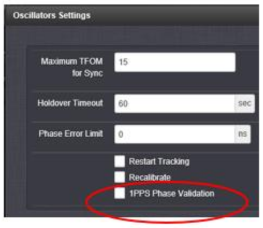
Note: these calls were performed on a SecureSync, with included screenshots from the browser- to show the effects of the various calls:

- **Verify if Smart Reference Monitor is enabled**: **RS_GetPhaseThreshold 0 all**<enter>

A) Smart reference NOT ENABLED (Mode = 0 and N = 0)



```
spfactory@Spectracom /home/spectracom $ RS_GetPhaseThreshold 0 all

   Mode:           0
   N:              0
   Min Threshold:  999999999.000000
   Max Threshold:  999999999.000000
```



Oscillators Settings

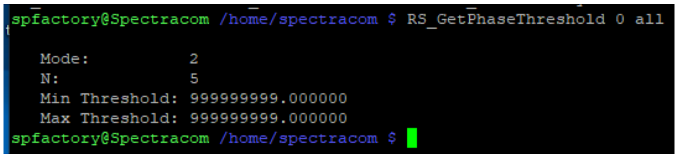
Maximum TFOM for Sync: 15

Holdover Timeout: 60 sec

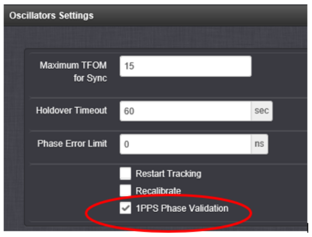
Phase Error Limit: 0 ns

□ Restart Tracking
□ Recalibrate
□ 1PPS Phase Validation

B) Smart reference ENABLED (Mode = 2 and N = 5)



```
spfactory@Spectracom /home/spectracom $ RS_GetPhaseThreshold 0 all

   Mode:           2
   N:              5
   Min Threshold:  999999999.000000
   Max Threshold:  999999999.000000
spfactory@Spectracom /home/spectracom $
```



Oscillators Settings

Maximum TFOM for Sync: 15

Holdover Timeout: 60 sec

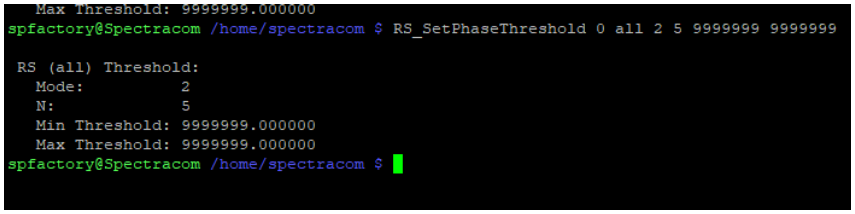
Phase Error Limit: 0 ns

□ Restart Tracking

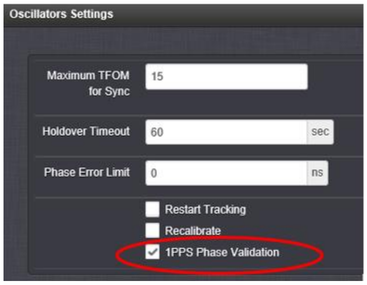
□ Recalibrate

☑ 1PPS Phase Validation

- **ENABLE Smart Reference Monitor:** **RS_SetPhaseThreshold 0 all 2 5 999999 999999**<enter>



```
   Max Threshold: 9999999.000000
spfactory@Spectracom /home/spectracom $ RS_SetPhaseThreshold 0 all 2 5 9999999 9999999

 RS (all) Threshold:
   Mode:          2
   N:             5
   Min Threshold: 9999999.000000
   Max Threshold: 9999999.000000
spfactory@Spectracom /home/spectracom $
```



Oscillators Settings

Maximum TFOM for Sync: 15

Holdover Timeout: 60 sec

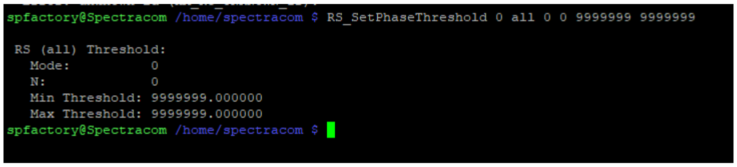
Phase Error Limit: 0 ns

□ Restart Tracking

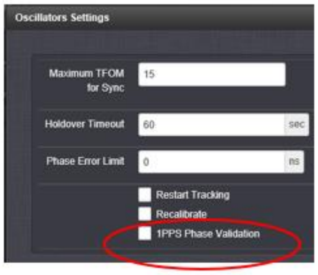
□ Recalibrate

☑ 1PPS Phase Validation

- **Disable Smart Reference Monitor:** **RS_SetPhaseThreshold 0 all 0 0 999999 999999**<enter>



```
spfactory@Spectracom /home/spectracom $ RS_SetPhaseThreshold 0 all 0 0 9999999 9999999

 RS (all) Threshold:
   Mode:          0
   N:             0
   Min Threshold: 9999999.000000
   Max Threshold: 9999999.000000
spfactory@Spectracom /home/spectracom $
```



Oscillators Settings

Maximum TFOM for Sync: 15

Holdover Timeout: 60 sec

Phase Error Limit: 0 ns

□ Restart Tracking

□ Recalibrate

□ 1PPS Phase Validation

Isssue: Oscillator locks to GPS/IRIG input, but then starts drifting off (phase error slowly increasing)

- This symptom can indicate a potential bad oscillator or bad oscillator calibration
- Refer to salesforce cases such as case **24534** (for LMCO).
  - They have two boards in the same system, syncing via IRIG AM, exhibing oscillator drift while the IRIG input continues to be present.

Per Tim T (10 Apr 17) request the following items:

Note : Also need to know if the TSync board has a TCXO or OCXO onboard.

The Oscillator type can be determined by either:

- The Serial Number of the board (**LS_GetSerialNo 0** call) or
- Via the oscillator type call (**XO_GetOscType 0**)

1) From both TSync boards- its **oscillator's DAC** value (the response to the following example program: "**XO_GetDac**", such as **XO_GetDac 0** and **XO_GetDac 1**).
   - The center for the OCXO DAC is 0x**8000** (the DAC should be on either side of and close to this value)
   - The center for the TCXO DAC is 0x**32768** (the DAC should be on either side of and close to this value)

   Example response from customer with two TSync_boards having a TCXO installed (11 Apr 17):
   - **XO_GetDac 0:** 46450
   - **XO_GetDac 1**: 28455

   Note: *Tim T confirmed these both look fine because they are both on either side of 0x**32768***

2) From both TSync boards- its **oscillator's cal value** (the response to the following example program: "XO_GetCalVal", such as **XO_GetCalVal 0** and **XO_GetCalVal 1**).

   This value shouldn't ever be the following: +0.000000 (This indicates a bad calibration occurred)

   Example response from same customer above with two TSync boards having **TCXO** installed (11 Apr 17):
   - **XO_GetCalVal 0**: 0.000366822

- **XO_GetCalVal 1**: 0.000358157

Note*: Tim T confirmed these both look fine for tcxo step sizes*

Typical/expected cal value for OCXOs is around: +/- .00013xxxx to +/- .00015xxxx

Email from Tim T (22 Apr 17) So I did a quick test of some TSync-PCIe board that I have.

- Board with **TCXO** had cal value of **0.0032** Hz per DAC step
- Boards with **OCXO** had cal value of **0.00032** Hz per DAC step

3) If you have a digital scope which supports this capability, a screenshot from the scope showing the IRIG AM input signal levels to both boards. If you are using an analog/digital scope which can't generate a screenshot, please let us know the amplitude of the larger of the two amplitudes (this is the Mark amplitude) into high impedance

DAC VALUES FOR OSC DISCIPLINING (XO_GetDac 0 call)

- API call/example program to read the DAC value: **XO_GetDac 0** <enter>
  - **Note**: This is the second best configuration to use, (If GPS input is not available) when CF is available from the IRIG generator/source, but the IRIG source can't provide the IEEE 1344 extensions.

E) 1PPS disciplining, when an TCXO oscillator is installed

DAC values (the command to get the D/A Value is **XO_GetDac 0**)

- **Min D/A:** 0000
- **Max D/A:** 0xFFFF
- **The center for TCXO DAC** is 0x**32768** (DAC values should be on either side of this value)

Valid D/A range is: 0x2AAA to 0xD555

TCXO D/A vdc steering range: 0-3vdc

Oscillator alarm is asserted when D/A +/- 10% (0.2vdc) of the valid D/A range

F) disciplining, when an OCXO oscillator is installed

DAC values: (the command to get the D/A Value is **XO_GetDac 0**)

- **Min D/A:** 0x0000
- **Max D/A:** 0xFFFF
- **The center for the OCXO DAC is** 0x**8000** (DAC values should be on either side of this value)

Valid D/A range is:

OCXO D/A Vdc steering range: 0-5vdc

HR (Hertz range) is calculated once at the factory and then stored in EEPROM (not erased with a Clean).

“Oscillator Alarm” is asserted when D/A +/- 10% (0.2vdc) of the valid D/A range ???

Typical SecureSync Oscillator log entries with an OCXO installed and being disciplined

```
Aug 28 19:20:32 Spectracom spectracom: [system] 2012 241 19:20:32 000 New DAC Setting: 0x7F3A
Aug 28 19:18:52 Spectracom spectracom: [system] 2012 241 19:18:52 000 New DAC Setting: 0x7F3E
Aug 28 19:18:52 Spectracom spectracom: [system] 2012 241 19:18:52 000 Frequency error recalculated:
00.000050 (5.00x10^-12)
```

Example DAC from our OCXO SecureSync synced with GPS: DAC value: 0x8E52

D/A value not valid

DAC values such as **oxEEEE** or **0xD555** indicate a bad OCXO oscillator that needs to be replaced.

Note: If the oscillator is failing, the TFOM value may start off, or be drifting, higher than expected (such as TFOM 6 or 7, for example).

If the D/A value appears to have been thrown off, causing problems with oscillator disciplining, (which can affect outputs such as NTP, 10MHz, 1PPS, etc) , there is a **patch.img** file available that can reset the oscillator calibration value back to 0. This causes the board to re-perform the oscillator calibration procedure, with a new cal value calculated and the D/A value being placed at a reasonable value.

Time it takes for OCXO oscillator to lock

- TFOM exceeding MaxTFOM is 10 times faster slew rate than if MaxTFOM is not exceeded.
- It only stays at the increased slew rate while MaxTFOM is exceeded. As the System PPS is pulled in, TFOM will continue to decrease. It may eventually fall below MaxTFOM, at which point is slows down to the slower slew rate until its back into alignment.

Also keep in mind that oscillator disciplining is dependent upon a very stable input PPS. If there is excessive jitter in the reference PPS, oscillator disciplining will be trying to keep up with the jitter. This will inherently help dampen out the input jitter so the System PPS is not as jittery as the reference. But the TFOM may not be able to ever decrease, because it can never actually be able to become aligned with the Reference PPS.

Email after talking to Mark Goodlein

After a cold-start, The OCXO oscillators take about 4 minutes to warm-up. A quick-phase oscillator adjustment then occurs as soon as the 1PPS reference has been detected and is considered valid. This quick phase-align adjustment takes about a minute to complete. Then, it takes around 2 minutes for the oscillator to be disciplined “right on”.

Total oscillator alignment process time, after a cold start (assuming the input reference is connected at power-up) is about 6 to 7 minutes from power-up.

For the TSync with an OCXO oscillator, the 1PPS is always slewed during re-alignment.
The rate at which it is slewed depends upon how the “Maximum TFOM” setting has been configured. This setting allows the user to define how accurate the 1PPS must be in order for the box to report that it is “In Sync”.

If the current TFOM value is less than or equal to the Max TFOM setting, then the frequency of the oscillator is offset by a maximum of 0.4 Hz (40 ns/s) until the 1PPS is realigned.
If the current TFOM value is more than the Max TFOM setting, then the frequency of the oscillator is offset by a maximum of 4 Hz (400 ns/s) until the 1PPS is realigned.
As you can see by these slew rates, it may take a long time to re-align the 1PPS if it is very far off. However, there are always 10 million cycles between each 1PPS (except during power-up).

Keep in mind that it only stays at the increased slew rate while MaxTFOM is exceeded. As the System PPS is pulled in, TFOM will continue to decrease. It may eventually fall below MaxTFOM, at which point is slows down to the slower slew rate until its back into alignment.

80 microseconds of phase error is 80,000 nanoseconds. So if TFOM is not exceeding MaxTFOM, at 40ns/second of slew rate, it will take 2000 seconds (about 30 minutes) for the System PPS to be aligned. However, if TFOM exceeds MAXTFOM, it will speed up to 400ns/second (until TFOM no longer exceeds MaxTFOM, which causes it to slow back down to 40ns/second). Assuming TFOM exceeds MaxTFOM the entire time until it was back into alignment, at 400ns/second, it will take 200 seconds for it to be realigned.

Also keep in mind that oscillator disciplining is dependent upon a very stable input PPS. If there is excessive jitter in the reference PPS, oscillator disciplining will be trying to keep up with the jitter. This will inherently help dampen out the input jitter so the System PPS is not as jittery as the reference. But the TFOM may not be able to ever decrease, because it can never actually be able to become aligned with the Reference PPS. So the TFOM will remain a much higher number than usual.

Desire to provide an offset to the board's "clock" time by a set amount (Delay/advance the System 1PPS/on-time point)

The **TSYNC_CS_SubsecAdj** API is used to make one-time adjustments to the 1PPS on-time point. That adjustment is really only useful when trying to manually align the 1PPS without a true input reference. If they are using an input reference with an on time point like GPS, IRIG, or the external 1PPS they should use the input offset APIs for each of those references: TSYNC_GR_SetOffset, TSYNC_IR_SetOffset, or TSYNC_PR_SetOffset. Those allow for a continuous offset for the system when using the corresponding input reference. The input offset is in nanoseconds with a range of ±500 milliseconds. Because the offset is on the input itself, the effect will propagate to all aspects of the system: system time, timing outputs, match time, timestamps, etc.

There are three available TSync-PCIe calls to offset both the input time references (GPS and/or IRG) as well as the input reference 1PPS as well. Refer to the TSync-PCIe driver guide for more information on these three commands (attached for your reference).

- **TSYNC_GR_SetOffset** is used to offset the input GPS time (Page 4-102)

  (Offset is in nanoseconds from -500 msec to +500 msec)

- **TSYNC_IR_SetOffset** is used to offset the IRIG input's time. (Page 4-109)

  (Offset is in nanoseconds from -500 msec to +500 msec)

- **TSYNC_PR_SetOffset** is used to offset the external 1PPS (EPP 0) 1PPS input. (Page 4-37)

  (Offset is in nanoseconds from -500 msec to +500 msec)

Note: The driver guide "Description" for PR_SetOffset indicates it gets the offset value. It is a typo that should read that it sets the offset.

Using MAX TFOM to go into Holdover Mode/ cause oscillator coarse adjust to occur again

Besides resetting just the timing board itself in order for the oscillator coarse adjustment to automatically occur again (if the TFOM is excessively high), you can also take advantage of the "Max TFOM" value, if you want. If the estimated TFOM value ever exceeds the configured "Max TFOM" value, the coarse adjustment will occur again, until the TFOM value no longer exceeds Max TFOM.

The default Max TFOM is a value of "15". With Max TFOM still set to the highest possible value of "15", TFOM is never exceeded. However, you can lower the Max TFOM, so that if estimated timing errors exceed the desired values, the oscillator adjustment will occur much faster, instead of it being slewed very slowly. As the 1PPS is brought it, the TFOM value will decrease., Then, when its much closer, the TFOM will be within its set limits, and it will then switch from coarse adjust to software fine-tune slewing mode.

The Max TFOM value is set using **SS_SetMaxTfom**.

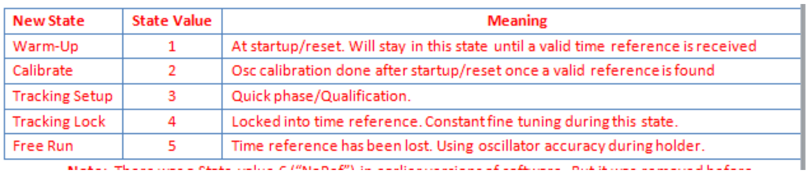
Example: The Max TFOM is changed to "5". If the estimated TFOM is 6 or greater, the TSync will go into Holdover mode and a coarse adjustment occurs, until TFOM becomes 5. Then, it switches back to slewing mode, until TFOM becomes the lower possible value.

Exceeding MaxTFOM is 10 times faster slew rate than if MaxTFOM is not exceeded

- Restart tracking was added in TSync FPGA Firmware v2.2.1 (KTS version 3.0.4). Note: V2.2.1 was released 18 Mar, 2013 (ECN 3436).
- The command to get the current D/A Value is: **XO_GetDac 0**
- **The command to coarse adjust the oscillator (Restart Tracking) without needing to power cycle the board is**: **XO_SetMode 0 2**

Email from Tim Tetreault to a customer The DAC values you are reporting look fine. The center for the DAC is 0x8000 so your values show that it is nowhere near the limits. The DAC values will continue to change as the unit fine tunes itself to the external reference.

I would like you to run a different command that tells us different states the unit is in.
The command is “XO_GetState 0”
The table below are what the different states mean:



New State	State Value	Meaning
Warm-Up	1	At startup/reset. Will stay in this state until a valid time reference is received
Calibrate	2	Osc calibration done after startup/reset once a valid reference is found
Tracking Setup	3	Quick phase/Qualification.
Tracking Lock	4	Locked into time reference. Constant fine tuning during this state.
Free Run	5	Time reference has been lost. Using oscillator accuracy during holder.

Note: There was a State value 6 (“NoRef”) in earlier versions of software. But it was removed before firmware version 2.2.1.

If you have a valid time reference connector to the board, the state should go from “1” to “4”, “4” indicating it has locked in to the reference. If you were to remove the time reference, the state would go to “5” indicating that the unit was now in holdover mode.

When the unit is in state “4” and you were to run the command “XO_SetMode 0 2”, you should see the state change to “3” indicating that the unit is in the quick phase state and that the tracking has restarted.

Can you verify this on your setup? This will tell us for sure if the “reset tracking” is working in your setup.

Restart tracking not working (no DAC change or shift in 1PPS after issuing command)

1) Verify the FPGA firmware version is v2.2.1 or higher (v2.2.1 added this capability).
2) Make sure an external input reference is present and valid.

Using a custom driver (not a Spectracom supplied driver) to perform Restart tracking

Attached for your reference (just in case you don’t already have it) is a copy of the latest version of the TSync-PCIe driver guide. Refer to page 4-118 of this document, for the ***XO_SetMode*** command.

The API call that is used to perform a Restart Tracking is an **XO_SetMode 0 2**. This call will send an **XO_Mode 2** command to the timing board, which is a “Reset” of the oscillator disciplining.

Phase Error limit

- Function added in SecureSync v5.2.0 to automate "Restart Tracking"
- Refer to "**Phase Error limit field**" in the SecureSync cust assist doc for more info: ..\SecureSync-VersaSync CustAssist.pdf

Note: (Aug 2017) This function should be in more recent TSync firmware versions (after it was added to KTS and after that version of KTS was implemented in TSync). But not sure if this function has ever yet been added to the TSync drivers to be able to configure the limit value.

Oscillator 10 MHz Accuracy (Oscillator accuracy when locked to a reference)

- Oscillator accuracy is based on being locked to an external reference that we can discipline with (such as GPS, IRIG, etc)
- Our Oscillator accuracy specs (manual and data sheet) are based on GPS being the selected reference

What if the selected reference is another reference other than GPS (such as IRIG input)?

- Oscillator accuracy will depend on the quality of the external input being provided

A) IRIG input

- IRIG DCLS input is better than IRIG AM input
- A SecureSync/Tsync board feeding IRIG DCLS to a Tsync board can achieve synchronization close to that of a 1PPS connection because of how stable the IRIG output of the SecureSync (IRIG master) is.

Email from Dave Sohn (19 Nov 14) The statement from the manual is a generic statement that we've been looking at revising. The real answer is that the stability and accuracy will depend on the quality of the IRIG provided to the SecureSync. For instance, a SecureSync feeding IRIG DCLS to another SecureSync can achieve synchronization close to that of a 1PPS connection because of how stable the IRIG output of the SecureSync is.

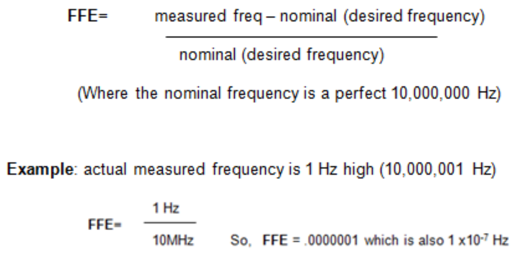
Email from Dave Sohn (19 Nov 14) Stability during holdover isn't based at all on the synchronizing reference prior to entry into holdover. It is entirely based on the oscillator and the effects due to aging, temperature, etc. The accuracy will be affected by the input reference. Similar to my email on SecureSync IRIG input, the accuracy of the system will depend on the quality of the IRIG reference.

Fractional Frequency Error (FFE)

- **Note:** Fractional Frequency errors are reported in “Hertz” (Hz)
- **Go**od conversion website: http://www.convertworld.com/en/frequency/Millihertz.html

Notes for using a scientific calculator

1) Use the **“+/- “** and the **“EXP”** buttons on calculator to enter values in scientific notation
2) The “**F-E**” button converts the value to scientific notation.



$$\textbf{FFE=}\ \frac{\text{measured freq} - \text{nominal (desired frequency)}}{\text{nominal (desired frequency)}}$$

(Where the nominal frequency is a perfect 10,000,000 Hz)

Example: actual measured frequency is 1 Hz high (10,000,001 Hz)

$$\textbf{FFE=}\ \frac{1\ \text{Hz}}{10\text{MHz}}$$

So, **FFE** = .0000001 which is also 1 x10^{-7} Hz

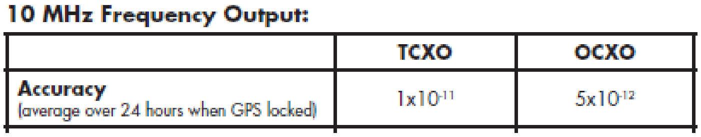
- Example: actual measured output frequency is exactly 1 Hz high (10,000,001 Hz)

$$\textbf{FFE=}\ \frac{10{,}000{,}001\ \text{Hz}}{10\text{MHz}}$$

So, **FFE** = .0000001 (which is also 1 x10^{-7} Hz)

Oscillator Accuracies (From the TSync-PCIe Data Sheet):

Important Note: these average accuracy specs below are average over 24 hours when locked to GPS.



10 MHz Frequency Output:

	TCXO	OCXO
Accuracy (average over 24 hours when GPS locked)	$1\mathrm{x}10^{-11}$	$5\mathrm{x}10^{-12}$

TCXO osc: $1\mathrm{x}10^{-11}$ /day:

$1\mathrm{x}10^{-11}$ Hz/day = 0.00000000001 Hz/ day = 0.01 nano Hz/day

OCXO osc $5\mathrm{x}10^{-12}$ Hz/day=

$5\mathrm{x}10^{-12}$ Hz/day = 0.000000000005 Hz /day = 0.005 nano Hz /day

Holdover mode / Oscillator 10 MHz free-run (Oscillator Stability)

- Several factors affect oscillator stability
  - Aging rate
  - Temperature variations
  - Oscillator's input Power fluctuations
  - Humidity
  - And others

Oscillator drift is essentially the oscillator aging rate, plus temperature drift, plus the effects of other variables.

- factors that don't affect stability
  - The Selected reference before oscillator goes into Free run mode (such as **IRIG** input instead of **GPS**)

Email from Dave Sohn (19 Nov 14) The statement from the manual is a generic statement that we've been looking at revising. The real answer is that the stability and accuracy will depend on the quality of the IRIG provided to the SecureSync. For instance, a SecureSync feeding IRIG DCLS to another SecureSync can achieve synchronization close to that of a 1PPS connection because of how stable the IRIG output of the SecureSync is.

Email from Dave Sohn (19 Nov 14) Stability during holdover isn't based at all on the synchronizing reference prior to entry into holdover. It is entirely based on the oscillator and the effects due to aging, temperature, etc. The accuracy will be affected by the input reference. Similar to my email on SecureSync IRIG input, the accuracy of the system will depend on the quality of the IRIG reference.

Important note: Oscillator Drift is non-linear

Email from Dave Sohn (17 Oct 2013) Drift in holdover is not linear as implied here. The phase drift will be exponential as the frequency drifts. So, no oscillator will provide <1ms per day indefinitely. For example, the TSync OCXO will provide less than 1ms per day for around 6-7 days, but more than that afterwards. Also, all of our models are based on good starting values starting out from GPS synchronization, which won't be available here.

Email from Tony Diflorio to a potential customer looking for 1ms/day accuracies A couple of other ideas to help you: You could use a portable (battery operated) GPS synchronization device to provide re-synchronization to the TSync-PCIe-Opt-OCXO card, such as our model GPS-12/12R (see attached). This re-synchronization would have to be done every few days though to maintain the TSync-PCIe-OCXO to <1ms per day of drift. Or, you might be able to use our popular "SecureSync", rack mounted GPS Synchronization system (see attached) with a Rubidium internal oscillator. This depends on whether you must have specific features that only a Bus-Level card can provide. Using our SecureSync with a Rubidium internal oscillator will provide the holdover accuracy you are looking for, over a much longer period of time (almost indefinitely).

Selected reference (GPS, IRIG, etc) before Holdover/oscillator free run begins

- What type of input reference (GPS, IRIG, external PPS, etc) that was selected before the oscillator goes into Freerun mode has no effects on free-run operation.

Per Dave Sohn (19 Nov 2014) Stability during holdover isn't based at all on the synchronizing reference prior to entry into holdover. It is entirely based on the oscillator and the effects due to aging, temperature, etc. The accuracy will be affected by the input reference. Similar to my email on SecureSync IRIG input, the accuracy of the system will depend on the quality of the IRIG reference.

Oscillator aging rates

- **Refer to:** I:\Engineering\Oscillators
- Aging rates are based on 30 days of continuous operation
- Rates are much higher with less than 30 days of operation (it's a huge curve)

Examples

- **TCXO**: Aging rate is
- **OCXO**: Aging rate is

Our Specs

- **Medium term Stability** is over a longer period of time (such as either after 24 hours or after 1 month, as specified)
- **Short term Stability** is over shorter, moving "windows" of time (such as in just 1 second, over 10 seconds, or over 100 seconds).

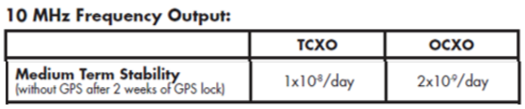
  Note: For TSync-PCIe boards, the data sheet and user manual only report Medium term Stability- not short term. The SecureSync data sheet reports both Medium and Short term stability

- Our specs (discussed below) are TYPICAL or average values – not Absolute values!

B) Oscillator Medium stability in free run: From TSync Data Sheet (with some modification for presentation)

- Holdover" values over one day or one month (as specified)

Important Note: this medium term stability specs below are average values upon loss of input reference that has been present for at least two weeks (system running the whole time). Shortening this period can severely degrade these values.



10 MHz Frequency Output:

	TCXO	OCXO
Medium Term Stability (without GPS after 2 weeks of GPS lock)	$1x10^{8}$/day	$2x10^{9}$/day

.

TCXO: $1x10^{-8}$ /day:

$1x10^{-8}$ Hz/day = 0.00000001 Hz/ day = 10 nano Hz/day

OCXO: $2x10^{-9}$ Hz/day=
$2x10^{-9}$ Hz/day = 0.000000002 Hz /day = 2 nano Hz /day

- **Oscillator Short Term stability in free run:** not provided in data sheet or manual. Only medium term specs are provided.

TCXO/OCXO comparison chart

- **Default Holdover period** is 7200 seconds (2 hours)

The following is a link to a chart which shows accuracy when locked and free run mode:
I:\Engineering\Products\ProductPerformanceSpecs5.xls

API calls to get current Sync and Holdover states:

TSYNC_SS_GetSync*:* Get the current sync state.
TSYNC_SS_GetHoldover: Get the current holdover state.
TSYNC_SS_GetHoldoverTO: Get the current holdover timeout.
TSYNC_SS_SetHoldoverTO: Set the current holdover timeout.
SYNC_SS_GetFreeRun: Get the current freerun state.

Oscillator free-run associated API calls

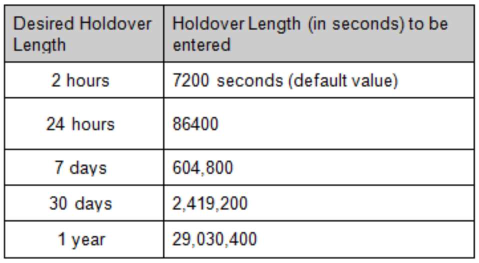
TSYNC_SS_GetFreeRun: Get the current freerun state.

The holdover mode begins when all input references are considered invalid and ends when either at least one input is considered valid or until the holdover period expires.

The length of holdover is set using the **TSYNC_SS_SetHoldoverTO** API call (the current value is read using TSYNC_SS_GetHoldoverTO. The value is configured in seconds with a valid range of 1 second up to 5 years. The default holdover period is 2 hours.

- The Holdover period can be changed from two hours as desired, after each boot-up, using the following API call/example program, **SS_SetHoldoverTo 0 x** **(**where **x** represents the number of seconds of allotted time it can remain in Holdover mode upon loss of all input references

When a valid input is connected, the board is in sync, and not in the holdover mode. Once the input is lost, for a certain duration, the timing board will go into the holdover mode of operation. During this interval (default Holdover is 2 hours), the time is being derived from the board’s oscillator and so it also remains in the sync mode in addition to the holdover mode being active. If the external reference is restored before the holdover period expires, the board remains in sync, but holdover is turned off. If two hours elapses with the reference not returning, both sync and holdover are then removed.



Desired Holdover Length	Holdover Length (in seconds) to be entered
2 hours	7200 seconds (default value)
24 hours	86400
7 days	604,800
30 days	2,419,200
1 year	29,030,400

FAQs

Q Why is the default holdover value 2 hours

A As for 7200 seconds of default holdover- this equates to 2 hours of holdover, which just happened to be the same value as the "worst-case" NetClock (With TCXO osc). Because the default Holdover value is not linked to the type of on-board oscillator, they went with the default of the NetClock/TCXO being 2 hours of holdover. This value is customer configurable for up to 5 years.

Oscillator stability with reference removed (in freeeun)

1PPS output holdover specs with OCXO oscillator

Email from Dave Sohn (10/3/11) regarding low phase noise drift (no GPS) after at least two weeks of lock:
Estimate for holdover at constant temperature after 2 weeks of GPS lock

Days	Drift
1	10 us
2	40 us
3	80 us
7	430 us

Email from Tim Tetreault to Will Hickey:
Will,
The Holdover spec's for the TCXO is the same between the TSync and SecureSync:

Standard TCXO:
450usec/24hr (constant temp after 2 weeks of GPS lock)

The Holdover spec's for the OCXO is not the same because they use different OCXO's. The spec for the TSync is:

Optional OCXO:
90usec/24hr (constant temp after 2 weeks of GPS lock)

So if the customer wanted to calculate the drift per seconds:
The TCXO would be:
45E-5/(60sec x 60min x 24 hours) = 5.2 ppb

The error for the OCXO would be:
9E-5/(60sec x 60min x 24 hours) = 1.0 ppb

- **Disciplined to Timecode**: 2 x 10–7
- **Undisciplined Freewheel Drift**: 1 x 10–6

Oscillator stability when the board is in holdover mode:

- TCXO our oscillator stability is 1E-6 (1 ppm),
- OCXO our oscillator stability is 5E-8 (50 ppb).

There are two main error factors for oscillator stability – Initial startup and aging over time. The initial startup and lifetime aging specs are as follows:

Oscillator	Initial startup	Lifetime aging
OCXO oscillator	0.2 PPM	2.0 PPM
TCXO oscillator	1.0 PPM	5.0 PPM

With an external input reference (such as GPS and IRIG), we can account for these errors.

To calculate the estimated amount of Freewheel drift with a TCXO:
Convert the time to seconds and multiply by 1.5E-6.

Examples (where 1 hr = 3600sec):
Amount of drift per hour = 3600 * 1.5E-6 = 0.0054 seconds per hour (or **5.4 milliseconds**) of estimated drift

Amount of drift per second= 0.0054sec / 3600 (seconds) = .0000001 seconds per second (or **0.1 microseconds)** of estimated drift

Amount of drift per day= 0.0054sec x 24 (hours) = .0123 seconds per day (or **12 milliseconds**) of estimated drift

To calculate the estimated amount of Freewheel drift with an OCXO:
Convert the time to seconds and multiply by 5E-8.

Examples (where 1 hr = 3600sec):

Amount of drift per hour = 3600 * 5E-8= 0.00018 seconds per hour (or **0.18 milliseconds**) of estimated drift

Amount of drift per second= 0.00018sec / 3600 (seconds) = 0.00000005 seconds per second (**or 0.05 microseconds)** of estimated drift

Amount of drift per day= 0.00018 sec x 24 (hours) = 0.00432 seconds per day (or **4.32 milliseconds**) of estimated drift

"1PPS not in Specification" / "1PPS Restored to Specification" log entries

These are events that occur and alarm entries that are asserted. "1PPS not in Specification" occurs if the current TFOM values exceed the user-configurable max TFOM value. When the TFOM falls below Max TFOM, the Restored to Specification event/alarm entry is asserted.

Regarding the "**1PPS Not in Specification**" alarm, this alarm only occurs if the current TFOM exceeds the user-configurable max TFOM value. The SecureSync going into Holdover mode will not initially and automatically cause the TFOM to increase and exceed the max TFOM value. However, if the unit continues to operate for an extended period of time without any valid input references, the oscillator will begin to drift. As long as the max TFOM is not set to 15, eventually, the max TFOM will be exceeded and the alarm is asserted. The type of oscillator (OCXO or Rubidium) and the setting of max TFOM will determine the typical amount of time in holdover required to exceed the max TFOM.

**TFOM / Max TFOM / Time Accuracies of TSync-PCIe boards

TCXO/OCXO comparison chart

The following is a link to a chart which shows accuracy when locked and free run mode:
I:\Engineering\Products\ProductPerformanceSpecs5.xls

TFOM (Time Figure of Merit)

- Refer to Oscillator stability/estimated timing accuracies with reference removed for more info on TFOM
- **SS_GetTfom:** Get the current system TFOM.
- **SS_SetMaxTfom:** Set the maximum TFOM threshold to stay in sync.
- **SS_GetMaxTfom:** Get the maximum TFOM threshold to stay in sync.
- **XO_GetPhaseError** Phase error can be read using API call or example program to see if on-time point is getting close to the input reference (should continue to decrease as it gets closer).

SS_GetTfom:

- Gets the current system TFOM.

When the TSync board is closely aligned with its 1PPS input reference, the TFOM and Phase Errors will be lower numbers (these numbers are oscillator-type dependant), Typically TFOM will be 4 or less and the Phase error will be like 200ns or less when in sync with the input PPS reference. The command to read the current TFOM value is **SS_GetTFOM 0 and the command to read the current phase error is XO_GetPhaseError 0.**

Q. How does the TSync-PCIe board calculate the TFOM Value?
The TSync board uses the calculated phase error value as the input for the TFOM Calculation. This data is collected in a variable size window and the result compared to a table of set values to select the TFOM value.
The variable min and max measurement window gets complex. It is set depending on the accuracy of the reference source, the stability of the data and the type of oscillator used.
The TFOM is an estimate of the timing error of the board and meant to be used as an indicator of the general performance of the timing;

The Time Figure of Merit (TFOM) enumeration is defined to provide a dimensionless quality measure of the overall time/1PPS synchronization based on an Estimated Time Error (ETE). This measure is used for overall sync status of our clock to the internal 1PPS. Our synchronization definition is derived from our use of time and 1PPS to achieve both synchronization of time and synchronization of frequency.

The TFOM value is a very accurate comparison (Estimate- not a measurement) of the input reference 1PPS versus the 1PPS output from the oscillator disciplining circuit (which uses the input 1PPS as its reference). As the SecureSync is able to discipline itself to a very stable and accurate 1PPS, the TFOM values will decrease. However, if the input reference 1PPS is jittery or does not have a “crisp” rising edge that it can definitively define as the trigger point, each time the TFOM is calculated, the value is going to be higher than expected.

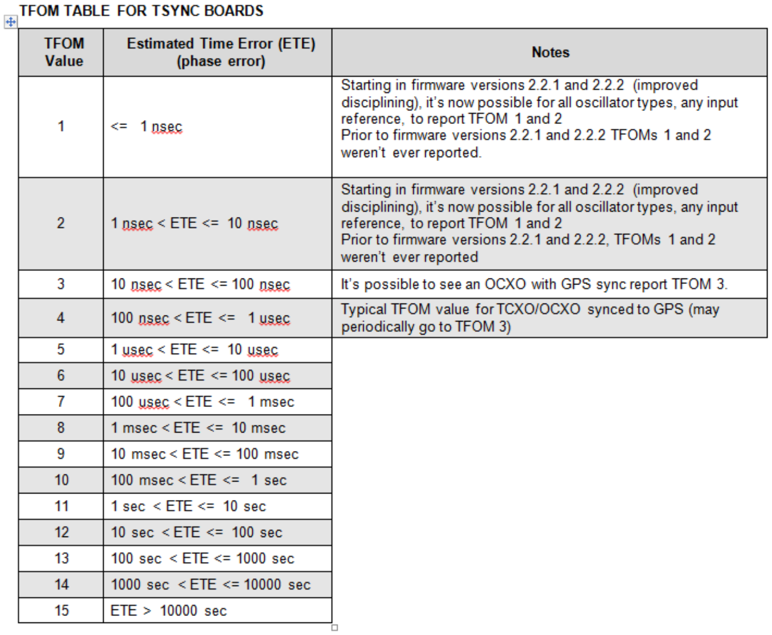
The most common issue we see when customers use an external 1PPS input (“EPP0”) as the 1PPS input reference is due to ringing of the signal because of improper termination. The input impendence of the 1PPS signal is high impedance. However, the 1PPS source may desire a 50 ohm termination into the TSync-PCIe-PCIe board (instead of high impedance). If the source desires a 50 ohm load termination, the input 1PPS may likely “ring” inside of the TSync-PCIe board. The ringing of the 1PPS signal prevents a “crisp” point at which the TSync-PCIe board can trigger on. With ringing of the signal present, the detected trigger point may coincidentally be at the right point, but it could also trigger too early or too late. So, with each 1PPS input when the signal is ringing, it may trigger at a different point on the signal each time. This prevents the disciplining circuit from being able to receive a stable 1PPS that is can accurately discipline to, and so the TFOM values will also be much higher than expected.

We have seen a few other similar cases where an external 1PPS input was causing a jittery 1PPS input because of improper termination of the signal at the input of the board. These were all resolved with a 50 ohm input being applied. To see this affect, if you have access to an oscilloscope, input the 1PPS signal from the source with a high input impedance setting on the scope. Look at the rising edge. You will likely see a ringing of the signal with no defined rising edge. Then, switch the scope to 50 ohm termination and you will likely notice the signal a become a crisp 1PPS with a very defined rising edge. This is what the board is also likely receiving and seeing, as well. To prevent the ringing of the 1PPS input from occurring and affecting the TSync-PCIe-PCIe board, terminate the 1PPS input from the source with a 50 ohm load resistor. This will often “clean up the signal” so that it can be a better reference into the board.

Let me know if adding the 50 ohm resistor from the input to ground allows the TFOM values to significantly decrease (TFOM values of 3 or 4 would be expected). For your information, the TFOM value is recalculated/updated about every 20 seconds or so.

Notes about TFOM

- TFOM value is updated:
  - About once-per-second with a Rubidium oscillator
  - About every 10 seconds with a TCXO or either type of OCXO oscillator installed. This is the amount of time it takes for the TFOM value to be calculated.
- The TFOM is a calculated phase error between the reference 1PPS vs the disciplined 1PPS. With an RB, the oscillator reports this phase error every second. With an OCXO it takes about 10 minutes to calculate this phase error.
- The calculated TFOM also adds on any correction applied in order to phase align the disciplined 1PPS to the 10 MHz from the oscillator. As corrections are made for this continuous alignment the amount of error correction are applied to the TFOM.
- TFOM is a worst-case error measurement. It may be better than the measured error, but it won't be any worse the measured error.
- TFOM with an OCXO oscillator installed will be no less than 3 (never 1 or 2) because of values intentionally added to the measurements if they are "too good" for what is expected to be measured with an OCXO. This is to make sure the TFOM will be no less than 3 with the OCXO oscillator (Since we sample for very short periods of time, the values read may be VERY good during that time frame, but too good to be expected, so we throw in a correction factor so that the calculated TFOM isn't BETTER than what the realistic values are.
- TFOM with an Rb oscillator, synced to GPS should have a TFOM value of 3. It may on occasion go to a value of 2 for short periods of time, but for the most part, it should be a value of 3.



TFOM TABLE FOR TSYNC BOARDS

TFOM Value	Estimated Time Error (ETE) (phase error)	Notes
1	<= 1 nsec	Starting in firmware versions 2.2.1 and 2.2.2 (improved disciplining), it's now possible for all oscillator types, any input reference, to report TFOM 1 and 2 Prior to firmware versions 2.2.1 and 2.2.2 TFOMs 1 and 2 weren't ever reported.
2	1 nsec < ETE <= 10 nsec	Starting in firmware versions 2.2.1 and 2.2.2 (improved disciplining), it's now possible for all oscillator types, any input reference, to report TFOM 1 and 2 Prior to firmware versions 2.2.1 and 2.2.2, TFOMs 1 and 2 weren't ever reported
3	10 nsec < ETE <= 100 nsec	It's possible to see an OCXO with GPS sync report TFOM 3.
4	100 nsec < ETE <= 1 usec	Typical TFOM value for TCXO/OCXO synced to GPS (may periodically go to TFOM 3)
5	1 usec < ETE <= 10 usec	
6	10 usec < ETE <= 100 usec	
7	100 usec < ETE <= 1 msec	
8	1 msec < ETE <= 10 msec	
9	10 msec < ETE <= 100 msec	
10	100 msec < ETE <= 1 sec	
11	1 sec < ETE <= 10 sec	
12	10 sec < ETE <= 100 sec	
13	100 sec < ETE <= 1000 sec	
14	1000 sec < ETE <= 10000 sec	
15	ETE > 10000 sec	

TFOM values decreasing over a long period of time (start out high, but decrease over time)

If an input reference is lost over a period of time, or an input reference switch occurs, it can take a long time for TFOM to eventually decrease to expected values (coarse adjust happens only once, after the TSync-PCIe board is started).
One of our engineers does agree that a TFOM value of 6 is higher than expected. However, he reminded me a condition that could cause the TFOM to be higher than expected for a fairly lengthy period of time (but will allow it to eventually be brought in to a lower number).

To better understand the situation, it's important to know that the oscillator is only "jumped" to the 1PPS input reference the first time after power-up that an input reference is applied to the timing board. After an initial oscillator correction, the oscillator is disciplined for fine-tuning. If either of the two conditions below happens to occur after the initial alignment of the oscillator, the oscillator is not "jumped" again (coarse-aligned) to the input reference:

1) If the timing board loses the input reference (or it is considered not valid) for a length of time, without a power cycle of the board occurring, and then the same reference is later returned.

2) If the timing board switches from one input to another (such as initially being synced to IRIG input, but then switching to GPS input) and the two 1PPS inputs are not closed aligned to each other.

In conditions where the Timing board continues to operate without an input reference present or not valid, the oscillator will begin to drift (this is called Holdover, or free-run). The type of installed oscillator determines how much drift will occur, over a period of time, when no input references are available. When an input (the same or another) becomes available again without the board being reset, the oscillator is then very slowly slewed to the input reference. Over a period of time (determined by the initial starting amount of error and the slew rate), the phase error will decrease and the quality of TFOM will increase.

Based on all of this information, did the timing boards continue to operate for any length of time with no GPS input, but then the GPS reception was restored, without a power cycle/reset of the board (not just the machine) occurring in between? If this occurs, a starting TFOM of 6, but decreasing over a period of time to a 5 or less, is not unexpected.

The nice thing is that you don't have to know the answer to this with any level of certainty. There is an API call that will allow you to observer if the oscillator is slewing to GPS, now that it has been restored. The API call to monitor the phase error being brought in over time is **XO_GetPhaseErr**. As you repeat this call over time, the phase error values should be getting smaller.

In a test environment, where there really isn't any problems with just cycling the timing boards, cycling the boards will then cause the oscillator to perform the coarse adjustment first, then disciple in the oscillator for fine-tuning. This will reduce the amount of time needed for the TFOM to decrease, as the TFOM values should be better within a very short amount of time after applying GPS input.

Let me know if you find the phase errors decreasing over time, or if power cycling the board allow the oscillator to snap in, resulting in lower TFOM values being reported.

Q. I would like to know what the drift is for the card when it is in holdover mode (no satellite reception) for a period of 24 hours when it is using the standard TCXO, and also when it is using the optional OCXO.
A. TCXO: 1ppm oscillator so absolute worst case would be 86.4ms over 24 hours
OCXO: 50ppb oscillator so absolute worst case would be 4.32ms over 24 hours

---

- Also, as far as I understand, this card provides bus level timing for computer. To what degree of accuracy will the system clock be held as compared to the time on the TSync card?

A. Depends on the OS. If using Linux with patched NTP and card… system clock should be within 10us.
Under Windows, we don't have the benefit of NTP controlling the OS time and tweaking it to align to the card. We only have our time set utility, which will on a periodic rate, read the card and set the OS time to it. It doesn't adjust any of the timing of the OS, so it will drift off from the card as soon as the time is written until the next time the utility reads the card and sets it. The accuracy of the system should be within 10us directly after the write to the OS time, but then drift away until the next write to the OS time. That would be dependent on the quality of the oscillator on the host machine, which usually aren't very good.

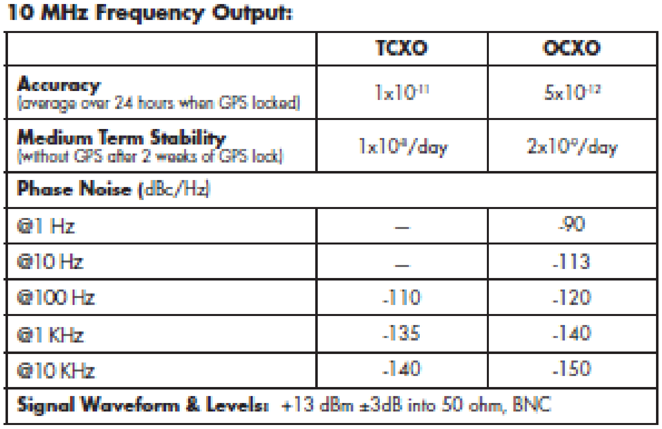
Max TFOM

- When current TFOM value exceeds Max TFOM, TSync will go into Holdover mode, until TFOM falls below Max TFOM.
- When current TFOM exceeds Max TFOM, OCXO/TCXO oscillator disciplining occurs faster.
- If the current TFOM value is less than or equal to the Max TFOM setting, then the frequency of the oscillator is offset by a maximum of 0.4 Hz (40 ns/s) until the 1PPS is realigned.
- If the current TFOM value is more than the Max TFOM setting, then the frequency of the oscillator is offset by a maximum of 4 Hz (400 ns/s) until the 1PPS is realigned.
- Max TFOM can be configured.
- Default Max TFOM is 15 (TFOM never exceeds Max TFOM).
- If Max TFOM is set to less than 15, User/User mode will not allow SecureSync to go into Sync.

The "**Maximum TFOM**" (Time Figure of Merit) field defines the largest TFOM value (TFOM is TSync's estimation of how accurately it is synchronized with its time and 1PPS reference inputs, based on several factors - known as the estimated time error or "ETE") that is allowed before disciplining is no longer performed on the oscillator. The larger the TFOM value, the less accurate TSync believes it is aligned with its 1PPS input that is used to perform disciplining. If this estimated error is too large, it could adversely affect the performance of oscillator disciplining. The available Max TFOM range is 1 through 15.

Outputs

*10MHz output



10 MHz Frequency Output:

	TCXO	OCXO
Accuracy (average over 24 hours when GPS locked)	$1x10^{-11}$	$5x10^{-12}$
Medium Term Stability (without GPS after 2 weeks of GPS lock)	$1x10^{8}$/day	$2x10^{9}$/day
Phase Noise (dBc/Hz)		
@1 Hz	—	-90
@10 Hz	—	-113
@100 Hz	-110	-120
@1 KHz	-135	-140
@10 KHz	-140	-150
Signal Waveform & Levels: +13 dBm ±3dB into 50 ohm, BNC		

Sine wave output only (no 10 MHz DCLS output available)

Output level

- Output level is not adjustable
- +13 dBm +/-3dB into 50 ohm, BNC (**10dBm**=2.00vpp **16dBm**=3.99vpp)
- 5Vp-p nominal into high impedance, +/-3dB

Output impedance: 50 ohms

Output Amplitude 12 dBm (2.5Vp-p), nominal into 50 ohm, +/-3dB,
5Vp-p nominal into high impedance, +/-3dB,

By default, The 10MHz output is enabled. With no external input, the signal will be derived from the free-running internal oscillator. When the board is synchronized to itself, the oscillator is not disciplined. In order to be disciplined, the board needs an external 1PPS reference, such as GPS or external 1PPS input. This output configuration is limited to disabling the output when the board is not synced (this is known as Signature Control") or disabled altogether (no output, whether or not the board is synced).

The calls to reconfigure when the 10 MHz is not present are listed in the driver guide as “FP” calls /commands. If you search the TSYNC driver guide for this value, it will show you all of the available configuration calls that can be issued to reconfigure the 10MHz output as desired.

Note: Further below is additional information on Signature Control.

Phase alignment of 10 MHz outputs from multiple TSync-PCIe boards or SecureSyncs

Because KTS disciplining performs phase alignment of the 10 MHz output, unlike earlier NetClocks which only perform Frequency adjustments, if more than one TSync or SecureSync are synced to the same IRIG or GPS signal, the 10 MHZ outputs will be in phase with each other (not just corrected for frequency).

Harmonics and Spurs of the 10 MHz output

Refer to the TSync data sheet for specs on harmonics. ..\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards

10 MHz Harmonics and spurs

For plots of the 10 MHz harmonics and spurs with an OCXO installed, refer to the email from Tom Richardson (8/2/12) in the following folder: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\10MHz output

10 MHz output Signature Control

Email KW sent to a customer:
The TSync-PCIe timing board has a feature that can be enabled, which squelches the 10 MHz output, when either the TSync-PCIe board's Sync state is not "true" OR when the TSync-PCIe has no valid input references (depending on how this mode is configured). This mode is called "Signature Control".

Signature Control can be enabled each time the TSync-PCIe board is powered-up (it is disabled by default) using either an API call or an example program. Attached for your reference is a copy of the TSync-PCIe driver guide (in case you don't already have a copy of it). Please refer to Section 4.2.9 (starting on page 4-153) for a list of API calls/example programs that are associated with the 10MHz output.

Specifically, the API call/example program to enable Signature Control for the 10MHz output is **FP_SetSigCtrl** (as discussed in Section 4.2.9.2 of the attached driver guide), where "SIG_CTL":

0: No Signature Control – 10 MHz always present

1**: Signature Control based on SYNC** (10 status MHz output is only outputted when the TSync board is either in the Sync or Holdover mode).

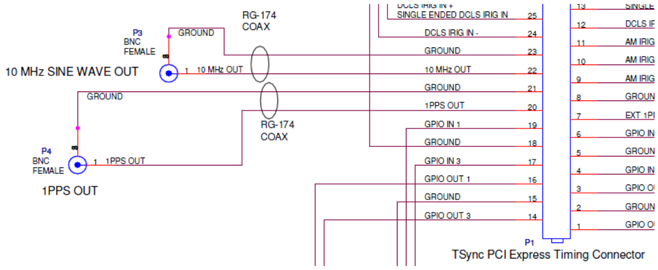
2: **Signature Control based on Reference status** (10 MHz output is only outputted when the TSync board has a valid input reference present).

3: Ultimate Signature Control - always OFF (10 MHz never outputted)

FYI- The call to read the current status of Signature Control is **FP_GetSigCtrl**.

10MHz output pin-outs

- ➢ Not available on Basic/Standard Breakout cable
- ➢ Using Premium Breakout cable (BNC Connector P3)



10MHZ Signal	Connector P3 - pin	Goes to Pin on Timing connector
10MHz	Center pin	22
Ground	Shield	23

*1PPS output

- **Note**: The 1PPS/ “Heartbeat” output is always 1PPS, but the pulse width of the output can be edited. For true heartbeat output (ability to change the frequency) refer to GPO outputs.

1PPS output impedance (TSync-PCIe only)

- (as of at least Mar, 2017) Currently **high impedence**, but may change to 50 ohm in the future (if we ever decide to respin the TSync-PCe board.
- **Note**: The 1PPS input is also high input impedance.
- **Note**: for the other TSync series board (besides the TSync-PCIe) their 1PPS output is **50 ohm impedance**.

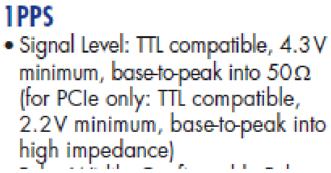
Q (Refer to SR4827 in SAP) From what I see in the user manual, it states that's it's not made to drive 5V TTL into a 50 ohm load? Is this correct? What sort of drivers are you using? I'm trying to get an idea of the output impedance and drive strength capability. All I see from the user manual is that it can drive 2.2V TTL (min) into a high impedance load only for this specific card whereas the other CCAs can typically drive 4.3V (min) into 50 ohms.

A **Reply from Keith after talking to Tim T (13 May 16)** To begin, your understanding of the info in the TSync family data sheet regarding the 1PPS output drive capability is correct. The TSync-PCIe board (the first timing board in the TSync family) is currently designed to drive a high impedance input with a TTL level signal. This was the requirement for the TSync-PCIe 1PPS output when this TSync board was made available. Due to a few customer requests since then, we have incorporated the 50 ohm drive capability into the designs of the other variant timing boards that have been more recently added to the TSync family (such as PMC, cPCI, etc). So these boards have a larger output and can drive a 50 ohm load.

We will likely change the TSync-PCIe boards to a 50 ohm output in the future, if we ever decide to re-spin the PCB board for the TSync-PCIe to incorporate any other changes to the design of this particular product.

1PPS Output level

- **TSync-PCIe only** the 1PPS output level/drive capability is smaller on the Tsync-PCIe (2.2v min,TTL)
- **Other TSync series boards (such as cPCI, PMC, etc):** output level is larger than the TSync-PCIe boards,at 4.3v min into 50 ohms) as indicated on the TSync series data sheet”:



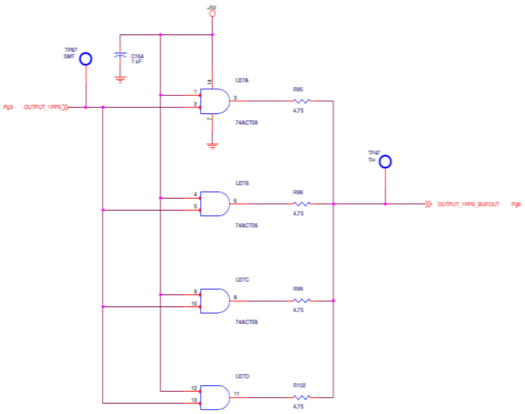
1PPS

- Signal Level: TTL compatible, 4.3V minimum, base-to-peak into 50Ω (for PCIe only: TTL compatible, 2.2V minimum, base-to-peak into high impedance)

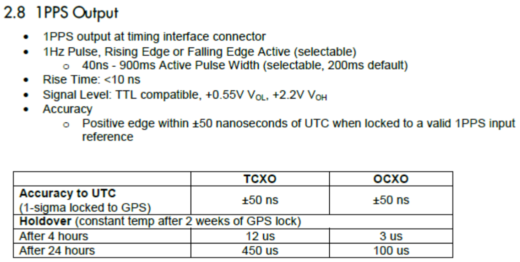
Tsync-PCIe 1PPS:

Other TSync family boards 1PPS: U27 is a 74ACT08 for 50 ohm load (as shown below)

1PPS OUTPUTS, 5V with 50 Ohm LOAD



5. 1PPS output Specs from manual:

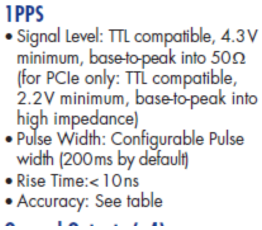


2.8 1PPS Output

- 1PPS output at timing interface connector
- 1Hz Pulse, Rising Edge or Falling Edge Active (selectable)
  - 40ns - 900ms Active Pulse Width (selectable, 200ms default)
- Rise Time: <10 ns
- Signal Level: TTL compatible, +0.55V V_{OL}, +2.2V V_{OH}
- Accuracy
  - Positive edge within ±50 nanoseconds of UTC when locked to a valid 1PPS input reference

	TCXO	OCXO
Accuracy to UTC (1-sigma locked to GPS)	±50 ns	±50 ns
Holdover (constant temp after 2 weeks of GPS lock)		
After 4 hours	12 us	3 us
After 24 hours	450 us	100 us

6. 1PPS output Specs from TSync data sheet:



1PPS

- Signal Level: TTL compatible, 4.3V minimum, base-to-peak into 50Ω (for PCIe only: TTL compatible, 2.2V minimum, base-to-peak into high impedance)
- Pulse Width: Configurable Pulse width (200ms by default)
- Rise Time:< 10ns
- Accuracy: See table

Two methods to output 1PPS (connectors/pin-out info)

1) Dedicated 1PPS output is on **pin 20** of the Timing connector   This correlates to either
    - BNC connector **P6** of the **Standard** breakout cable
    - BNC connector **P4** of the optional **Premium** breakout cable.
2) Configure one or more of the GPO pins for 1PPS (10PPS, etc)

- ➢ Note: the (4) GPO output pins can be programmed to also output 1PPS, in addition to 10PPS, 100PPS, etc .

Q. from Elotek "…asked if he can program the card to output **10 PPS** instead of **1 PPS**?"
A. Per Tim Tetreault (28 Apr 15) the dedicated 1PPS output can't be configured for this. But the four GPO pins on the Timing connector can be configured for signals such as 10PPS, 100PPS, etc. Refer to the GPOs for more info.

API calls for 1PPS/Heartbeat output: The 1PPS output calls for TSync-PCIe are the "**TSync _ PP"** calls. (Refer to the TSync-PCIe driver guide 1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121

- ➢ Refer to Tim Tetreault's "cheat sheet" at: ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\Tsync driver calls cheat sheet
- ➢ These are examples - additional calls may have since been added

1PPS output Commands
PP_GetEdge
PP_GetFreq
PP_GetNumInst
PP_GetOffset
PP_GetPulseWidth
PP_GetSigCtrl
PP_SetEdge
PP_SetOffset
PP_SetPulseWidth
PP_SetSigCtrl

Available configuration API calls for 1PPS output on pin 20

- Get/Set Signature Control
- Get Frequency (1Hz)
- Get/Set offset (account for cable delays to external device)
- Get/Set the active edge (defines which edge is aligned to the on-time point)
- Get/Set the pulse width of the 1PPS signal
- Get the number of instances (usually just one on the TSync board)

Per the TSync-PCIe manual:
The TSync-PCIe board generates a digital 1PPS output from the internal 1PPS of the system. Several parameters of the 1PPS can be controlled. The active edge can be set to either rising or falling edge, the pulse width can be adjusted, and an offset can be applied to adjust its relationship to the internal system, 1PPS from -500 msec to +500 msec in 5 nsec increments. The 1PPS output provides signature control similar to the IRIG outputs.

The TSync-PCIe timing board has legacy TPRO API calls available to configure the “Heartbeat” output (this is the 1PPS output on pin 20 of the timing connector). Unlike the TPRO/TSAT boards, TSync itself does not have a dedicated output called “Heartbeat”. This output was replaced by just a “1PPS output” instead. So the legacy heartbeat calls just configure this 1PPS output pin. There are no “TSync Heartbeat” API calls available in the TSync-PCIe drivers (just the legacy “TPRO heartbeat” calls. The 1PPS output calls for TSync-PCIe are the **TSync-PCIe_PP** calls. (Refer to the TSync-PCIe driver guide).

1PPS output holdover specs with OCXO oscillator

Email from Dave Sohn (10/3/11) regarding low phase noise drift (no GPS) after at least two weeks of lock:
Estimate for holdover at constant temperature after 2 weeks of GPS lock

Days	Drift
1	10 us
2	40 us
3	80 us
7	430 us

Signature Control for 1PPS output

- Signature control is available for 1PPS output (from the Tsync user manual “The 1PPS output provides signature control similar to the IRIG outputs”

API calls/example programs for 1PPS signature control

- **PP_GetSigCtrl (**PP_SetSigCtl <device index> <index>**)**
- **PP_SetSigCtrl (**PP_SetSigCtl <device index> <index> <mode>)

Where <index> should be 0, for the single 1PPS output

Where
SIG_CTL_NONE = 0, // No Signature Control - always ON
SIG_CTL_SYNC = 1, // Signature Control based on SYNC status
SIG_CTL_REF = 2, // Signature Control based on Reference status
SIG_CTL_OFF = 3, / Ultimate Signature Control - always OFF

The TSync board’s 1PPS output (via the DB25 “Timing connector” on the edge of the board) can be squelched (temporarily turned-off when the board is not in sync, or while its in Holdover Mode) using **Signature Control.** Signature Control, available for many TSync and SecureSync outputs is used for this function.

The Signature Control function for the 1PPS output is via the API call/example program o: PP_SetSigCtrl 0 x command (where x is one of the four following values):
0 = No Signature Control (1PPS output always on/present)

1 = Signature Control based on SYNC status (1PPS output is present when the board is in Sync, but squelched/not present while not in sync)

2 = Signature Control based on Reference status (1PPS output is present when the board has an input. Squelched when the board is in Holdover mode because it doesn’t have any available/valid inputs)

3 = Ultimate Signature Control - always OFF (1PPS is never present)

1PPS output offset

API calls/example programs for 1PPS output offset

- **PP_GetOffset (**PP_GetOffset <device index> <index>)
- **PP_SetOffset (**PP_SetOffset <device index> <index> <offset>)

Where <index> should be 0, for the single 1PPS output

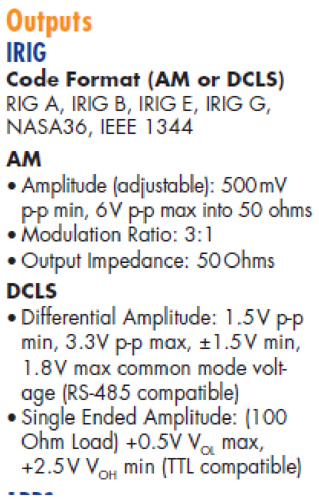
*IRIG output

IRIG output configuration

Power-up default settings for the IRIG AM and DCLS outputs

- **Year:** no year information provided by default
- **Date/Time** : 00:00:00 UTC and then it counts-up each second from there
- **IRIG Format:** IRIG B (IRIG B002 or B122)
- **Signature Control**: No Signature Control
- **Control Field data**: No data
- **Time Scale**: UTC
- **IPPS Offset**: No offset
- **RIG AM amplitude**: 128
- **Coded expression:** (1) BCD, TOY, CF (no year information)

IRIG output specs (from data sheet)



Outputs

IRIG

Code Format (AM or DCLS)
RIG A, IRIG B, IRIG E, IRIG G, NASA36, IEEE 1344

AM

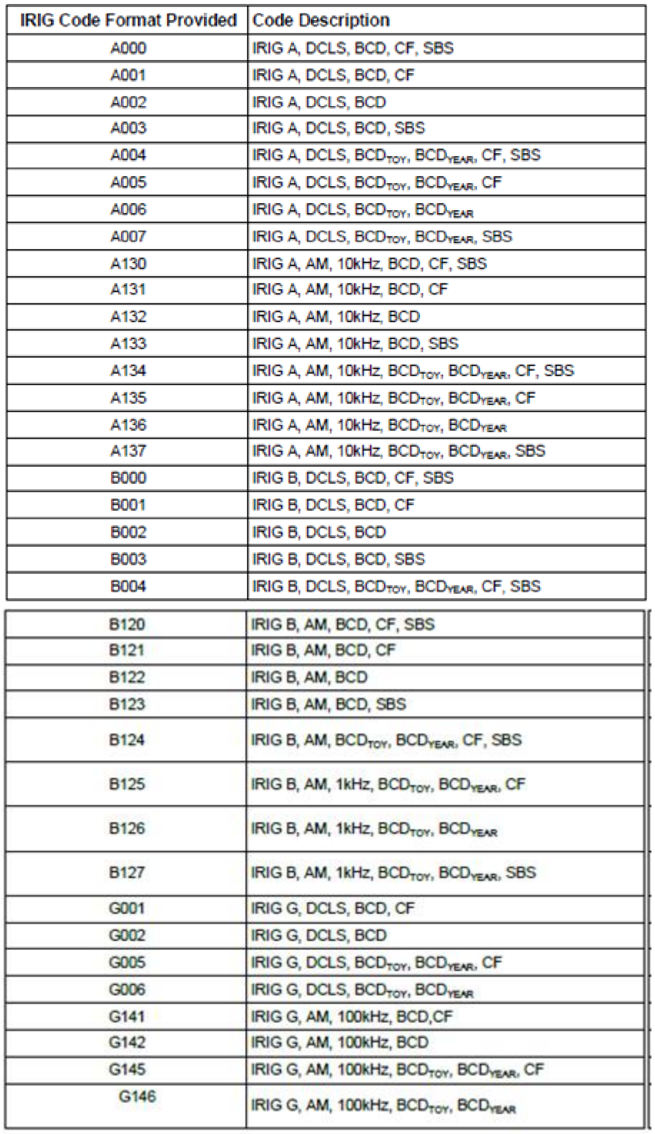
- Amplitude (adjustable): 500mV p-p min, 6V p-p max into 50 ohms
- Modulation Ratio: 3:1
- Output Impedance: 50Ohms

DCLS

- Differential Amplitude: 1.5V p-p min, 3.3V p-p max, ±1.5V min, 1.8V max common mode voltage (RS-485 compatible)
- Single Ended Amplitude: (100 Ohm Load) +0.5V V_{OL} max, +2.5V V_{OH} min (TTL compatible)

Available IRIG output formats supported

Email from Ron Dries (Sep, 2017) The TSync boards all support the same IRIG formats as the SecureSync. You are correct that the TSync manual needs to be updated. Here is a more current list from the SecureSync manual.



IRIG Code Format Provided	Code Description
A000	IRIG A, DCLS, BCD, CF, SBS
A001	IRIG A, DCLS, BCD, CF
A002	IRIG A, DCLS, BCD
A003	IRIG A, DCLS, BCD, SBS
A004	IRIG A, DCLS, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF, SBS
A005	IRIG A, DCLS, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF
A006	IRIG A, DCLS, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$
A007	IRIG A, DCLS, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, SBS
A130	IRIG A, AM, 10kHz, BCD, CF, SBS
A131	IRIG A, AM, 10kHz, BCD, CF
A132	IRIG A, AM, 10kHz, BCD
A133	IRIG A, AM, 10kHz, BCD, SBS
A134	IRIG A, AM, 10kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF, SBS
A135	IRIG A, AM, 10kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF
A136	IRIG A, AM, 10kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$
A137	IRIG A, AM, 10kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, SBS
B000	IRIG B, DCLS, BCD, CF, SBS
B001	IRIG B, DCLS, BCD, CF
B002	IRIG B, DCLS, BCD
B003	IRIG B, DCLS, BCD, SBS
B004	IRIG B, DCLS, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF, SBS
B120	IRIG B, AM, BCD, CF, SBS
B121	IRIG B, AM, BCD, CF
B122	IRIG B, AM, BCD
B123	IRIG B, AM, BCD, SBS
B124	IRIG B, AM, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF, SBS
B125	IRIG B, AM, 1kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF
B126	IRIG B, AM, 1kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$
B127	IRIG B, AM, 1kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, SBS
G001	IRIG G, DCLS, BCD, CF
G002	IRIG G, DCLS, BCD
G005	IRIG G, DCLS, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF
G006	IRIG G, DCLS, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$
G141	IRIG G, AM, 100kHz, BCD,CF
G142	IRIG G, AM, 100kHz, BCD
G145	IRIG G, AM, 100kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$, CF
G146	IRIG G, AM, 100kHz, $\mathrm{BCD}_{\mathrm{TOY}}$, $\mathrm{BCD}_{\mathrm{YEAR}}$

IRIG output impedances:

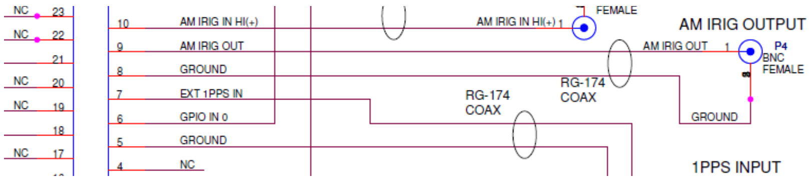
- The IRIG AM output is 50 ohms.
- The IRIG DCLS output is120 ohms.

IRIG output rise-time

- Can be adversely affected by coax cable attached to the board/breakout cable.

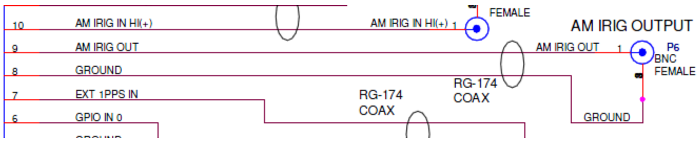
IRIG AM Output pin-outs

A) Using Basic/Standard Breakout cable (BNC Connector P4)



IRIG AM output Signal	Connector P4 - pin	Goes to Pin on Timing connector
IRIG AM out	Center pin	9
Ground	Shield	8

B) Using Premium Breakout cable (BNC Connector P6)



IRIG AM output Signal	Connector P6 - pin	Goes to Pin on Timing connector
IRIG AM out	Center pin	9
Ground	Shield	8

IRIG DCLS output pin-outs

A) Standard breakout cable

- IRIG DCLS output is **not available on Basic/Standard Breakout cable**
- **Note:** IRIG DCLS is on the "**Timing" connector pins 12 and 13** (pin 13 and ground for single-ended). Table for Timing connector is below

Timing connector pin-out (on metal edge of TSync board)

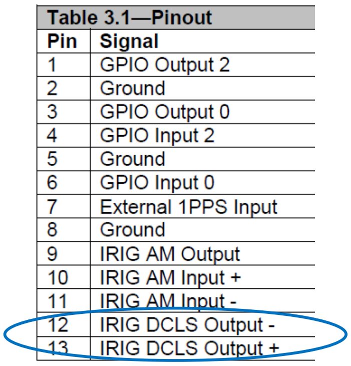
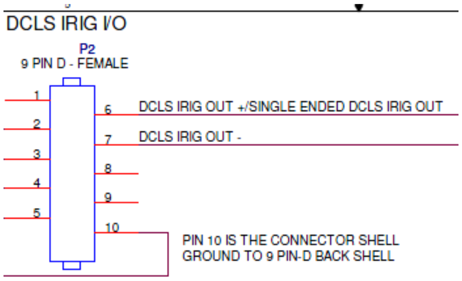


Table 3.1—Pinout	
Pin	**Signal**
1	GPIO Output 2
2	Ground
3	GPIO Output 0
4	GPIO Input 2
5	Ground
6	GPIO Input 0
7	External 1PPS Input
8	Ground
9	IRIG AM Output
10	IRIG AM Input +
11	IRIG AM Input -
12	IRIG DCLS Output -
13	IRIG DCLS Output +

B) Using Premium Breakout cable (DB9 Connector P2)


DCLS IRIG I/O
P2
9 PIN D - FEMALE
1
2
3
4
5
6
7
8
9
10
DCLS IRIG OUT +/SINGLE ENDED DCLS IRIG OUT
DCLS IRIG OUT -
PIN 10 IS THE CONNECTOR SHELL
GROUND TO 9 PIN-D BACK SHELL

IRIG DCLS output Signal	Connector P4 - pin	From Pin on Timing connector
IRIG DCLS out + (or single-ended)	6	13
IRIG DCLS out -	7	12

Also refer to: IRIG for TSync-PCIe

Both IRIG outputs are on the 25 pin “Timing” connection

- IRIG AM output is on pin 9
- IRIG DCLS is on pins 12 (-Data) and 13 (+ Data)

By default, the IRIG output is enabled. The IRIG output has the ability to be configured for different formats (A, B, E and G), local time as desired, the location of the year data, etc. These commands are listed in the driver guide as the “_IP” calls/commands. If you search the TSYNC driver guide for this value, it will show you all of the available configuration calls that can be issued to reconfigure the IRIG output as desired.

API calls for IRIG output: IRIG output calls for TSync-PCIe are the “**IP**” calls.

IR command syntax:
Use the <Index> value of:
0 for DCLS output ports.
1 for AM output ports.

TSync-PCIe boards can output= A, B, E, G, NASA36, IEEE 1344 pulse width codes (TSync and SecureSync do support IEEE extensions. Refer to: IEEE-1344:)

Note: Does not generate Manchester modulated codes.

IRIG output can provide the following coded expressions combinations for BCDTOY, CF, SBS, and BCDYEAR fields:
• 0 – BCDTOY, CF, SBS
• 1 – BCDTOY, CF
• 2 – BCDTOY
• 3 – BCDTOY, SBS
• 4 - BCDTOY, BCDYEAR, CF, SBS
• 5 - BCDTOY, BCDYEAR, CF

Output impedance: 50 ohms

Available IRIG output configuration commands (these are the “IP” calls/example programs)

Note: Technically, there is only one IRIG output with two electrical output connections for the IRIG output (one AM output on one pin and one DCLS output on two pins). Both of these outputs share their configuration, so you can’t configure one as IRIG B and the other as IRIG E, for example. And they can’t have different control fields as another example.

Note: GetMode and SetMode commands (Manual or automatic) only apply to IRIG INPUT. Not applicable to IRIG output.

Desire to turn on/off (enable/disable) the IRIG outputs using Signature Control

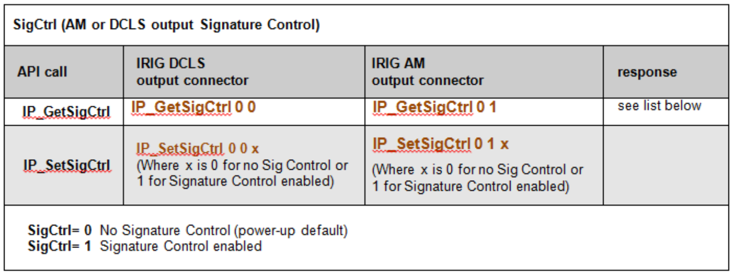
Q Is it possible to turn the IRIG AM and DCLS signals on and off through the API.

A. Reply from Keith (16 Aug 2013)

There is no direct way to turn off and one (enable/disable) the IRIG outputs of the TSync-PCIe board. However, there is an indirect method that can be used to enable/disable the IRIG outputs via the API interface.

The IRIG outputs have an available mode called “Signature Control”. Signature Control for the IRIG outputs is optionally used to squelch the modulation of the IRIG output signal, if thBy e TSync-PCIe board is not in sync/Holdover or if there are no input references currently present and valid for the TSync board to sync with. Signature Control is disabled by default each time the TSync powers- up, but it can be enabled after each power-up using the API interface, as desired.

With Signature Control for the IRIG outputs enabled via API calls any time after the TSync-PCIe board has powered-up, if a user either disconnects the external IRIG input and GPS input (as applicable), the IRIG output will then only output the carrier frequency, with no data being provided in the signal. Below are the commands to enable or disable IRIG output Signature Control:



SigCtrl (AM or DCLS output Signature Control)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetSigCtrl	**IP_GetSigCtrl 0 0**	**IP_GetSigCtrl 0 1**	see list below
IP_SetSigCtrl	IP_SetSigCtrl **0 0 x** (Where x is 0 for no Sig Control or 1 for Signature Control enabled)	**IP_SetSigCtrl 0 1 x** (Where x is 0 for no Sig Control or 1 for Signature Control enabled)	

SigCtrl= 0 No Signature Control (power-up default)
SigCtrl= 1 Signature Control enabled

Besides needing to physically remove the external input references, the input references can also be “disabled” via API calls. Just like I mentioned in my earlier email, if all references that are connected to the TSync are disabled by changing the Enable state in the Reference Status table, the references are treated as if they were disconnected. Then, they can be re-enabled when desired to allow IRIG output again. Once the input references are re-enabled in the Priority table, the TSync will sync back up again, and Signature Control will allow the IRIG modulation to be restored again. As long as the TSync-PCIe board remains powered-up, if all of the input references are disconnected or disabled, the IRIG output modulation will stop again (Signature Control isn’t a “one-time” mechanism).

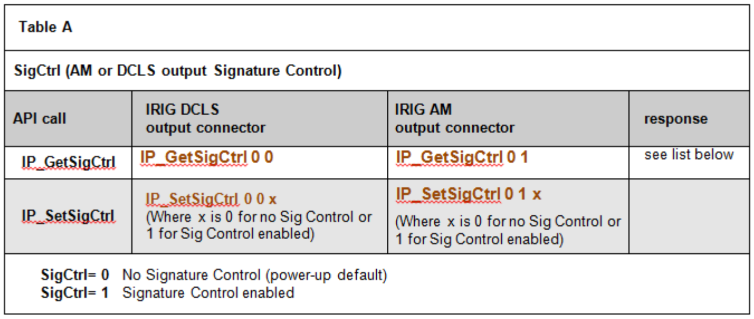


Table A

SigCtrl (AM or DCLS output Signature Control)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetSigCtrl	IP_GetSigCtrl 0 0	IP_GetSigCtrl 0 1	see list below
IP_SetSigCtrl	IP_SetSigCtrl 0 0 x (Where x is 0 for no Sig Control or 1 for Sig Control enabled)	IP_SetSigCtrl 0 1 x (Where x is 0 for no Sig Control or 1 for Sig Control enabled)	

SigCtrl= 0 No Signature Control (power-up default)
SigCtrl= 1 Signature Control enabled

Table B

Offset (IRIG AM or DCLS output Offset for cable delay)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetOffset	IP_ GetOffset 0 0	IP_ GetOffset 0 1	see list below
IP_SetOffset	IP_SetOffset 0 0 x (Where x is the desired offset)	IP_SetOffset 0 1 x (Where x is the desired offset)	

Power-up default: No cable offset entered

Table C

GetLocal (AM or DCLS output Time Zone and DST Offset)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetLocal	IP_GetLocal 0 0	IP_GetLocal 0 1	see list below
IP_SetLocal	IP_SetLocal 0 0	IP_SetLocal 0 1	

Power-up default: UTC time

Table D

Format (**AM or DCLS output Signature Control**)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_Get Format	**IP_GetFormat 0 0**	**IP_GetFormat 0 1**	see list below
IP_Set Format	**IP_SetFormat 0 0 x** (where x is the applicable value below)	**IP_SetFormat 0 1 x** (where x is the applicable value below**)**	

Format (0) IRIG A
Format (1) IRIG B (**power-up default**)
Format (2) IRIG G
Format (3) NASA-36
Format (4) IRIG E

Table E

Amplitude (**AM output amplitude**) **Note**: Not Applicable to IRIG DCLS

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetAmplitude	N/A	**IP_GetAmplitude 0 1**	see list below
IP_SetAmplitude	N/A	**IP_SetAmplitude 0 1**	

Amplitude=128 (**power-up default**)

Table F

Mod (**AM or DCLS output Modulation frequency**)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetMod	**IP_GetMod 0 0**	**IP_GetMod 0 1**	see list below
IP_SetMod	**IP_SetMod 0 0 x**	**IP_SetMod 0 1**	

0= No carrier
1= 100 Hz
2= 1 kHz (power-up default)
3= 10 kHz
4= 100 kHz
5= 1 MHz
6= Unknown frequency

Table G

Freq (**AM or DCLS output Frequency**)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetFreq	**IP_GetFreq 0 0**	**IP_GetFreq 0 1**	see list below
IP_SetFreq	**IP_SetFreq 0 0 x**	**IP_SetFreq 0 1**	

Freq= 0 1000 Hz (power-up default)
Freq= 1 100 Hz

Note: IRIG B is always 1000 Hz

Year value: Default year value/Desire to output year information

Default year value

- After each power-up , no year information is provided (default format is B002)
- If the IRIG output has been configured to provide year info, it outputs the same year as the Tsync board is sent to (the default year outputted at each power-up is 2000, until the TSync board has been synced to the current year)
- To output the current year, sync the TSync board via GPS or IRIG from an IRIG source that provides year information, or manually set the year in the TSync board. Refer to IRIG input section for more information on using IRIG input with year info.

***desire to output IEEE-1344 extensions in CF field (includes year info)

- For details, refer to the "**IEEE-1344 extensions (IEEE C37.118-2005**)" section in the Custserviceassist document
- Refer also to the 1344 Extensions in the IRIG input section of this document: IEEE-1344 Extensions

Desire to output year information

- After each power-up , no year information is provided (default format is B002)
- **To output year information, need to either (based on the device receiving the IRIG signal):**

A) Provide the year via BCD: change the Coded Expression to one that includes BCDyear information (refer to Coded expression table below for details)

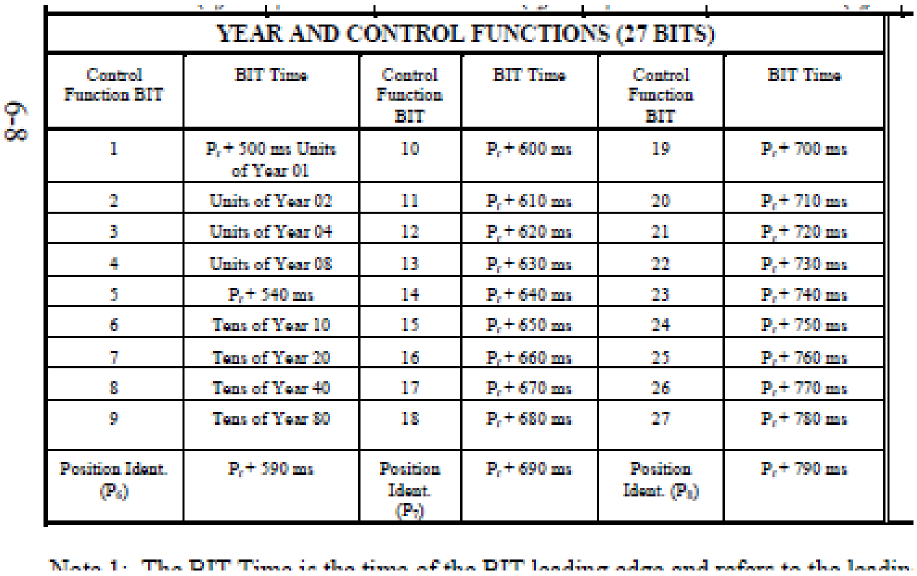
B) Provide the year in the Control Field: Change the Coded Expression to one that includes year information (refer to Coded expression table below for details)

- Need to configure the Control Field for the desired year location.

The **codedexp** and **ctrlfield** tables below provide info on configuring these two values to provide year info

Table H			
CodedExp (**AM or DCLS output Coded Expression**)			
API call	**IRIG DCLS** **output connector**	**IRIG AM** **output connector**	**response**
IP_GetCodedExp	**IP_GetCodedExp 0 0**	**IP_GetCodedExp 0 1**	see list below
IP_SetCodedExp	**IP_SetCodedExp 0 0 x**	**IP_SetCodedExp 0 1 x**	
0= BCD TOY, Control Functions, Binary Seconds (SBS) **1=** BCD TOY, Control Functions (power-up default) **2**= BCD TOY **3**= BCD TOY, Binary Seconds (SBS) **4**= BCD TOY/Year, Control Functions, Binary Seconds (SBS) **5=** BCD TOY/Year, Binary Seconds (SBS) **6**= BCD TOY/Year **7=** BCD TOY/Year, Binary Seconds (SBS) **8=** Unknown- no fields			

BCD Year



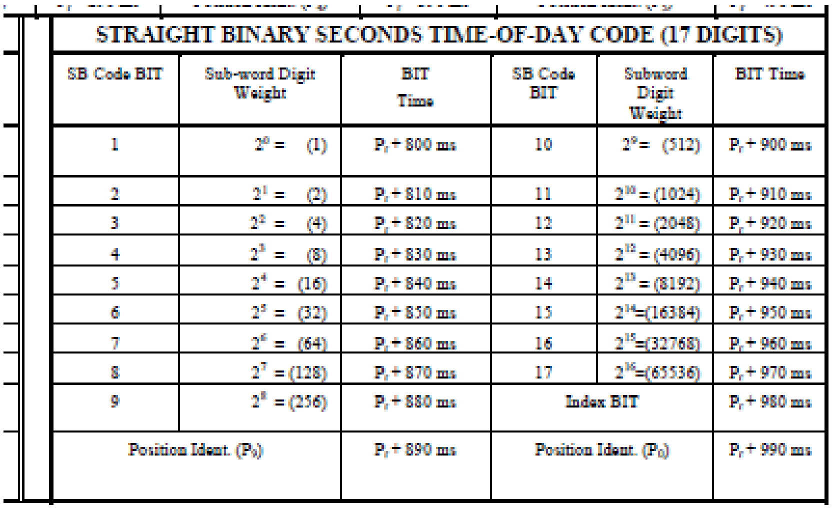
6-8

YEAR AND CONTROL FUNCTIONS (27 BITS)

Control Function BIT	BIT Time	Control Function BIT	BIT Time	Control Function BIT	BIT Time
1	P_r + 500 ms Units of Year 01	10	P_r + 600 ms	19	P_r + 700 ms
2	Units of Year 02	11	P_r + 610 ms	20	P_r + 710 ms
3	Units of Year 04	12	P_r + 620 ms	21	P_r + 720 ms
4	Units of Year 08	13	P_r + 630 ms	22	P_r + 730 ms
5	P_r + 540 ms	14	P_r + 640 ms	23	P_r + 740 ms
6	Tens of Year 10	15	P_r + 650 ms	24	P_r + 750 ms
7	Tens of Year 20	16	P_r + 660 ms	25	P_r + 760 ms
8	Tens of Year 40	17	P_r + 670 ms	26	P_r + 770 ms
9	Tens of Year 80	18	P_r + 680 ms	27	P_r + 780 ms
Position Ident. (P_6)	P_r + 590 ms	Position Ident. (P_7)	P_r + 690 ms	Position Ident. (P_8)	P_r + 790 ms

Note 1: The BIT Time is the time of the BIT leading edge and refers to the leadin

SBS



STRAIGHT BINARY SECONDS TIME-OF-DAY CODE (17 DIGITS)

SB Code BIT	Sub-word Digit Weight	BIT Time	SB Code BIT	Subword Digit Weight	BIT Time
1	2^0 = (1)	P_r + 800 ms	10	2^9 = (512)	P_r + 900 ms
2	2^1 = (2)	P_r + 810 ms	11	2^{10} = (1024)	P_r + 910 ms
3	2^2 = (4)	P_r + 820 ms	12	2^{11} = (2048)	P_r + 920 ms
4	2^3 = (8)	P_r + 830 ms	13	2^{12} = (4096)	P_r + 930 ms
5	2^4 = (16)	P_r + 840 ms	14	2^{13} = (8192)	P_r + 940 ms
6	2^5 = (32)	P_r + 850 ms	15	2^{14}=(16384)	P_r + 950 ms
7	2^6 = (64)	P_r + 860 ms	16	2^{15}=(32768)	P_r + 960 ms
8	2^7 = (128)	P_r + 870 ms	17	2^{16}=(65536)	P_r + 970 ms
9	2^8 = (256)	P_r + 880 ms	Index BIT		P_r + 980 ms
Position Ident. (P_9)		P_r + 890 ms	Position Ident. (P_0)		P_r + 990 ms

Table I

CtrlField (**defines AM or DCLS Control Field**)

API call	IRIG DCLS input connector	IRIG AM input connector	response
IP_GetCtrlField	**IP_GetCtrlField 0 0**	**IP_GetCtrlField 0 1**	see list below
IP_SetCtrlField	**IP_SetCtrlField 0 0 x**	**IP_SetCtrlField 0 1 x**	

CF=0 Unknown - All bits ignored
CF=1 Fields conform to RCC 200-04
CF=2 Fields conform to IEEE C37.118-2005 (1344 extensions)
CF=3 Fields conform to Spectracom format
CF=4 Fields conform to Spectracom FAA format
CF=5 Fields conform to NASA formats

Table J

TimeScale (AM or DCLS output TimeScale)

API call	IRIG DCLS output connector	IRIG AM output connector	response
IP_GetTimeScale	**IP_GetTimeScale 0 0**	**IP_GetTimeScale 0 1**	see list below
IP_SetTimeScale	**IP_SetTimeScale 0 0 x** where x is the desired TimeScale listed below)	**IP_SetTimeScale 0 1 x** (where x is the desired TimeScale listed below)	

TimeScale = 0 UTC
TimeScale = 1 TAI
TimeScale = 2 GPS
TimeScale = 3 Local

Table K

Message (**AM or DCLS output Message**)

API call	IRIG DCLS Output connector	IRIG AM output connector	response
IP_GetMessage	**IP_GetMessage 0 0**	**IP_GetMessage 0 1**	see list below
IP_SetMessage	**IP_SetMessage 0 0** (Sub P0 –Sub P9)	**IP_SetMessage 0 1** (Sub P0 –Sub P9)	

(**Sub P0 –Sub P9**)

No IRIG output modulation present (1 kHz carrier frequency only)

Besides Signature Control being enabled to stop modulation, IRIG modulation should always be present. However, if Signature Control is not enabled and there is no modulation (just a1 kHz carrier) present, this indicates the TSync-PCIe board is likely not running. See if any example programs will work.

Note- this picture and email below was tailored to TSync-PCIe (not other boards such as TSync-cPCI/PMC etc) though the concept still applies.

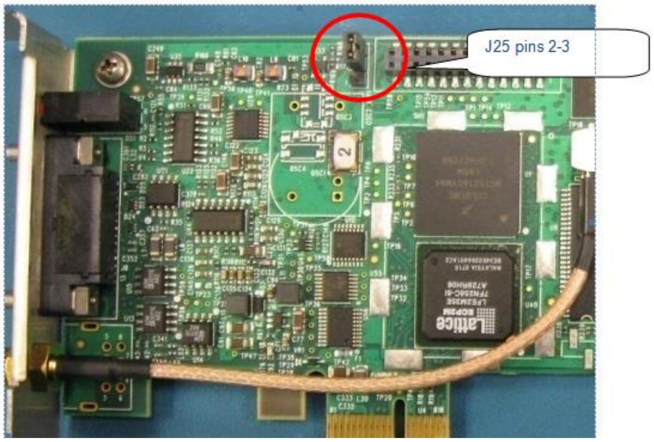
Email KW sent to Arnaud on 11/2/11

There are a couple of factors that could cause the IRIG output signal to only provide the carrier frequency, without any modulation. The most common reason would be if IRIG output Signature Control has been enabled.   Signature Control can be enabled (it's disabled by factory default) to prevent modulation from being present, if the TSync-PCIe board is either not "in sync"(or in Holdover) or if the input references are not valid. If you haven't enabled Signature Control sometime after the two boards have powered-up, this is not likely the reason for what you are observing.

The other less likely reason is if the TSync-PCIe board is not fully functional (not able to completely boot-up).   In this scenario, the IRIG carrier frequency of 1kHz can be present with no modulation. Question for you (and your customer): since the TSync-PCIe boards were installed in a machine, has there been successful communications with each board. Successful communications can be confirmed by running any of the example programs included in the TSync-PCIe drivers. If the board is properly communicating, there should be valid responses back from the example programs.

For two TSync boards to be acting the same way, one potential configuration issue comes to mind. If a certain jumper j25
in the correct position upon power-up, the TSync-PCIe boards would be "stuck" loading the bootloader and therefore the TSync-PCIe would not be able to operate. This jumper is only used for initial programming of the TSync-PCIe boards.

You can check jumper setting on J25 and make sure that it is configured on the pins to the outside edge of the board (pins 2 and 3), as seen in the red circle in the photo below:



If the jumper is not in the correct position, please change the jumper and try again. With the jumper moved to the correct position, operation of the board should then be restored. With Signature Control remaining disabled, a modulated IRIG output should now be present.

If the jumper on both boards is already in the correct position, please let me know if the TSync-PCIe driver has been installed yet. And if it has, will any of the example programs operate with the board.

**Output synchronization/accuracies (1PPS output, IRIG out, GPO out, Time out, etc)

The outputs, such as the 1PPS output are aligned to the on-time point, which is defined by the sub-second counter. Refer to the notes above regarding input synchronization and its effects on the output synchronization.

When a TSync board comes up and starts to synchronize to a reference, it will perform HW disciplining of the on-board 1PPS to attempt to bring the internal 1PPS of the TSync in line with the input reference. Once it is close, HW disciplining is turned off and SW disciplining takes over. During the period when HW disciplining is operating, the on-time point of the unit is being adjusted and can affect all users of that on-time point, including the TSync system clock and outputs. HW disciplining should be running for only about 1 minute after power-up once a reference is present for synchronization.

Note: The default edge of the input and output signals are high/rising (unless someone reconfigures for low/failing, they will be rising edge).

IRIG output accuracies (Option Cards Models 1204-05 and 1204-1E).

From the SecureSync data sheet, the IRIG outputs Accuracy is “+/- 20-200microseconds (format dependent)”

6/28/11 KW- Email from Dave Sohn:
The IRIG DCLS output is within 100ns of the 1PPS of the SecureSync. Its overall accuracy is dependent on the SecureSync reference. IRIG AM is very dependent on the output type, because of the different filters used for the various output carrier frequencies.

**Time output (Time reads)

****Time reads (used by NTP for linux sync)

HW_GetTime 0 call

****Issues with Time output

C) Time jump anomalies (one second, or several seconds,

- ➢ Res-SMT-GG receiver installed – update receiver firmware to at least version 1.8, if at an earlier rev
  - As of at least Dec 2015, updating receiver firmware requires board be returned to the factory for upgrade.
  - Refer to “Res-SMT-GG” in the custserviceassistance doc for more info on known issues with earlier versions of Res-SMT-GG firmware.

Logs and Alarms (Holdover/Time Sync/Frequency Error/Freerun/PPS Specification/Reference Change, etc)

****Enabling alarms

- Alarms need to enabled before being available (they are disabled by default).

****LS_GetAlarm call / example program

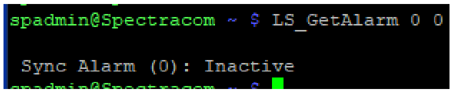
- Monitoring the Alarms using the LS_GetAlarm will allow you to check the status of certain alarm conditions on the Tysnc card. This will tell you if something has changed from normal condition.

LS_GetAlarm 0 X (where X is one of the numbers below, defining a specific alarm type) Displays the current alarm states for the specified alarm types below

List of all "Alarm" conditions (each is described further below)

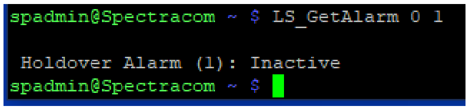
Note: Where <alarm type>

0= Time Sync alarm (Not in Sync and not in Holdover)



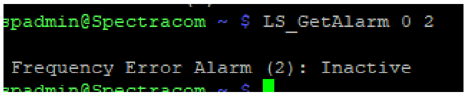
```
spadmin@Spectracom ~ $ LS_GetAlarm 0 0

 Sync Alarm (0): Inactive
spadmin@Spectracom ~ $
```

1= Holdover alarm



```
spadmin@Spectracom ~ $ LS_GetAlarm 0 1

 Holdover Alarm (1): Inactive
spadmin@Spectracom ~ $
```

2= Frequency error alarm



```
spadmin@Spectracom ~ $ LS_GetAlarm 0 2

 Frequency Error Alarm (2): Inactive
spadmin@Spectracom ~ $
```

3= Self reference only (Self/Self reference is selected)

4= Software alarm

5= 1PPS not in sync

6= Reference change

7= Hardware Alarm

- **The Time Sync alarm** indicates the TSync board has not yet synced to an input reference (such as GPS) since it was first powered-up .Or, it also indicates the TSync-PCIe was synced to an input reference, but the input reference was either lost or declared not valid (the GPS receiver was tracking four satellites, but dropped to only 3 satellites is an example of a present but not valid reference) and the Holdover period has since expired without any input references being restored since the Holdover period started (Holdover mode starts the moment there are no present/valid inputs to sync with).

To manually assert/test this alarm: After the board is in sync, temporarily disconnect the GPS and/or IRIG input (so there are no inputs still present). two hours later (by default, this alarm will be asserted. To expedite the alarm, shorten the Holdover period to less than 2 hours'/

- **The Holdover alarm** indicates the TSync board was synced since it last booted-up, but no inputs reference are currently present or valid. The oscillator is currently in a free-run mode with no oscillator disciplining occurring, until a reference is restored. While in Holdover alarm, Time Sync is still "True". But if no reference is restored before the allotted Holdover period expires, the Time Sync alarm is asserted and Time Sync status becomes "false".

  To manually assert/test this alarm: After the board is in sync, temporarily disconnect the GPS and/or IRIG input (so there are no inputs still present). Holdvoer alarm will occur within one second thereafter.

- **The Frequency Error** alarm indicates The 10 MHz onboard oscillator's DAC value is within a certain percentage of its steering range rail (it's close to the one of its two maximum limits that allows the oscillator to still be disciplined to 10 MHz).

  To manually assert/test this alarm: power cycle the system (not just warm reboot it) to power cycle the TSync board. The Frequency Error alert should be asserted after each boot-up.

- **The Freerun alarm (Self reference only is selected)** indicates the TSync-PCIe board is currently synced using the available Self/Self reference (TSync-PCIe board synced to itself with no external inputs). While in this state, the oscillator's frequency counts cannot be calculated and oscillator is not being disciplined (so it's in free-run mode).

  To manually assert/test this alarm: After the board is in sync, temporarily disconnect the GPS and/or IRIG input (so there are no inputs still present). Freerun alarm will occur thereafter.

- **The PPS specification Alarm** indicates the current TFOM value (estimated sync accuracy) is exceeding the configured MaxTFOM value (by factory default, maxTFOM is 15, the highest possible TFOM value. So, maxTFOM can't be exceeded. However, the maxTFOM value can be set to a lower value, allowing TFOM to possibly exceed maxTFOM (such as setting maxTFOM to 4, but the estimated accuracy is TFOM value of 5). If TFOM exceeds the maxTFOM, this alarm is asserted. It also causes the TSync-PCIe board to go into the Holdover mode, until TFOM no longer exceeds the MaxTFOM value."

  To manually assert/test this alarm: Temporarily Change the **MaxTFOM** value to "1" using the **SS_SetMaxTfom 0 1 call.** Then set it to the desired value (default value is 15) or it will also reset upon the next power-up of the TSync-PCIe board..

- **The Software alarm** (not likely to be asserted) indicates a software error has been detected.

  To manually assert/test this alarm: the command SW_ERROR cannot be induced manually. A SW_ERROR is asserted when the timing system logs an unexpected error has occurred. You can clear this alarm but wouldn't have the ability to assert it.

- **The Reference Change alarm** indicates the TSync-PCIe needed to switch from the highest priority reference that is present and valid as its selected reference, to using the next lower priority reference that is present and valid. This occurs if either or both the Time and/or PPS reference becomes no longer present or is declared not valid. The Reference Priority table defines what references to select from and allows the priority of the references to be assigned.

  For example, GPS/GPS is assigned Priority 1 and IRIG/IRIG is priority 2. If the GPS receiver drops to tracking 3 satellites, GPS Time and PPS are declared not valid, and if IRIG Time and PPS input is present and valid, the TSync-PCIe will select IRIG as its selected reference and since it still has a valid reference for sync, the TSync-PCIe board will not go into Holdover mode, and the oscillator continues to be disciplined to the IRIG reference.

  To manually assert/test this alarm: After the board is in sync, temporarily disconnect the GPS and/or IRIG input (so). Switching from GPS to IRIG, IRIG to GPS, or from GPS/IRIG to NONE will cause a Refernce Change alarm

- **Hardware Alarm**: indicates a bad Rubidium oscillator. (For internal use only)

  To manually assert/test this alarm: ?? (not sure if any condition will cause this to be asserted??)

Notes about other "non-alarm conditions":

- **Antenna Problem** is not classified as an "alarm" condition. It's a status condition which can be read with an API call/example program, but there is no automatic "action" tied to it, like other other "alarm" conditions.

LS_GetAlarm call/example program

Displays the current alarm states for the specified alarm types listed above.

Examples

LS_GetAlarm 0 0 = Time Sync alarm (Not in Sync and not in Holdover)
LS_GetAlarm 0 1 = Holdover alarm
LS_GetAlarm 0 2 = Frequency error alarm
LS_GetAlarm 0 3 = Self refererence only (Self/Self reference is selected) alarm
LS_GetAlarm 0 4 = software alarm
LS_GetAlarm 0 5 = 1PPS not in sync alarm
LS_GetAlarm 0 6 = Reference change alarm
LS_GetAlarm 0 7 = Hardware alarm

LS_SetAlarm 0 command call/example program

- Only used to **clear** the **Software alarm** and the **Reference alarm** which are the only two alarms that are latched on. All other alarms are "current state alarms" so no command is necessary to clear all other alarms.

Note: If the user tries to set an alarm with this command, the driver responds with "HA_RC_INVALID_PARAM

Note: Only the **Reference** and **Software** alarms are latched. All other alarms are not latched and clear when the alarm condition clears.

****Remote alarm indications/alert notification

The TSync-PCIe boards do not have alarm relays or SNMP capabilities. Alarm status (such as Time Sync alarms or Holdover alarms) can be obtained via the status LEDs on the edge of the board or via API status calls.

If a customer wants some type of remote alarm indication, the primary way to do this is to have their application software poll the status API calls (such as SS_GetSync for instance). Then if the response goes false, for example, indicating loss of sync, their application software can then perform a desired task (We don't make any recommendations for this).

Another approach is to enable Signature Control on the 10 MHz, IRIG or 1PPS outputs. If Sync is lost, the output can be squelched. Refer to the driver guide for info on enabling Signature Control.

A third approach is to monitor a GPO pin (such as GPO 0) for a signal change. The application software can command the pin to Direct Value Mode (DVM) and then set the pin to high or low. When the application software reads Time Sync status has changed, it can then set the pin to the opposite state. Refer to the GPO outputs further below for more info.

**PTP Master mode (PTP output)

- Refer to the TSync-PCI driver guide for an extensive discussion on how to set up the PTP boards.
- TSync-PCIe boards are factory configured as PTP Slaves (can be re-configured in the field as PTP Masters).

Note: Refer to: **PTP Precision Timing Protocol (for all products)** for more info on PTP.

Logs/log Messages

LS_GetMessage 0 command for KTS "Timing log" entries

- Need to "poll' this command to get "Timing Log entries (the same type entries asserted in SecureSync's Timing log
- If this command returns "**Error: opt err (RC_QUERE_ ERR)."** the log is currently empty. It needs an input reference present for log entries to be asserted.

Example Timing log entries

Associated Log entries for restart tracking

```
Aug  6 13:57:33 Spectracom Spectracom: [system] 2013 218 13:57:33 000 Oscillator control: Reset.
```

SecureSync losing/regaining IRIG input

```
Jun 22 15:01:40 Spectracom Spectracom: [system] 2015 173 15:01:39 021 IR irg0 Comm Lost.
Jun 22 15:01:44 Spectracom Spectracom: [system] 2015 173 15:01:44 000 IR irg0 Comm OK.
```

Opens/shorts in the antenna cable

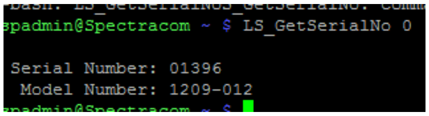
"GR antenna fault" / "GR antenna ok"

- These entries indicate opens/shorts being detected in the antenna cable and then clearing.

Tsync-PCIe’s Model and Serial Number

A) Via API call/Example program

LS_GetSerialNo 0



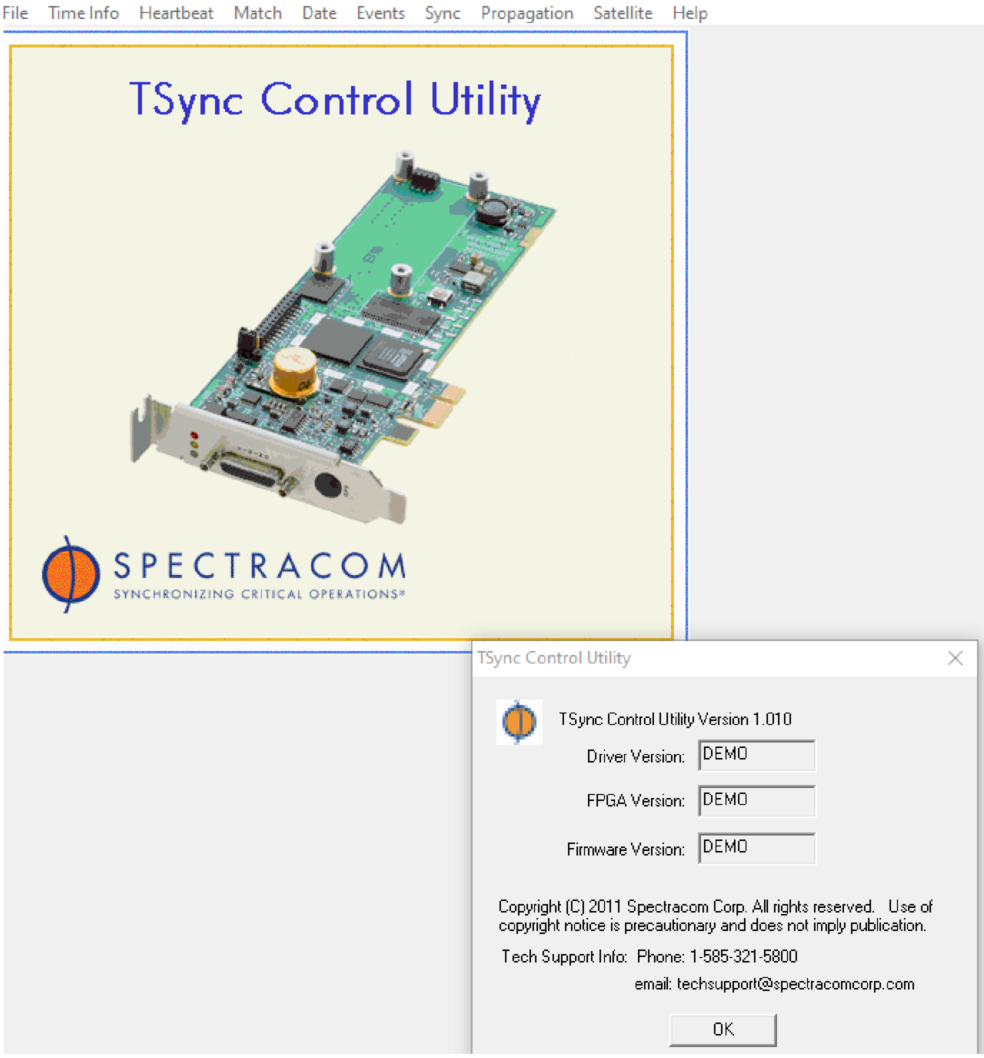
```
-bash: LS_GetSerialNoS_GetSerialNo: comm
spadmin@Spectracom ~ $ LS_GetSerialNo 0

 Serial Number: 01396
  Model Number: 1209-012
spadmin@Spectracom ~ $ 
```

B) Via Windows Control Utility

- Serial Number not available via Control Utility (Help -> About reports version info, but not Serial Number_

Serial Number for TSync-PMC boards only (not TSync-PCIe)

- Refer to: Serial number (TSync-PMC boards only)



TSync Control Utility Version 1.010

Driver Version: DEMO

FPGA Version: DEMO

Firmware Version: DEMO

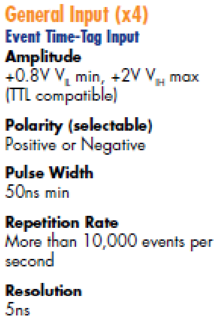
Copyright (C) 2011 Spectracom Corp. All rights reserved. Use of copyright notice is precautionary and does not imply publication.

Tech Support Info: Phone: 1-585-321-5800

email: techsupport@spectracomcorp.com

GPIO (General Purpose Input Output) pins

**GPI pins (General Purpose Inputs- “GPI”)



General Input (x4)

Event Time-Tag Input

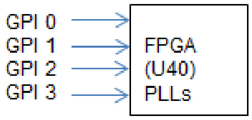
Amplitude
+0.8V V_{IL} min, +2V V_{IH} max
(TTL compatible)

Polarity (selectable)
Positive or Negative

Pulse Width
50ns min

Repetition Rate
More than 10,000 events per second

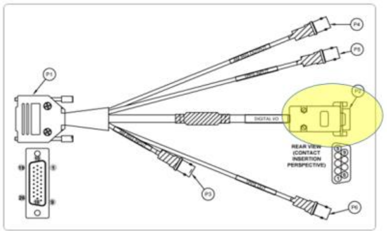
Resolution
5ns



- ➢ Four GPI pins are available (GPI 0 through GPI 3) on the **Timing** connector (in order - pins 6, 19, 4, 17)

Note: The standard breakout cable only provides connection to one GPI pin (the first GPI pin - GPI 0). The premium breakout cable is required for use if two or more GPI pins are needed.

The input signal needs to be physically applied to pin 2 (**GPIO Input 0**”) of the DB9 connector (Labeled as “P2” below and highlighted).



- ➢ **Function**: GPI pins can be used for:

  Time-tag/Event input (interrupts can also be generated when an event occurs).

- ➢ **Accuracy**: The General I/O output signals timing are accurate relative to the Input reference’s 1PPS signal to within +/- 50 nsec.

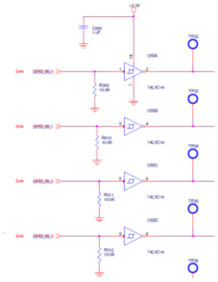
Input impedance of GPI pins:

- Question from Case Number 282325

  I am trying to use one of the GPI edge detection inputs and seeing some unexpected behavior when sensing the input. When I measure the impedance between pins 19 and 2 or 19 and 18 I get ~ 10kOhms. I thought the GPI would be high impedance. Does the GPI use a 10 kOhm pull-up instead?

Per Mark McGregor (sometime prior to the question above from Case 282325. But I included his comment in my response to that question). The GPI pins (General Purpose Input) are high impedance input (but if the source has 50 ohm output impedance, we can handle it – just add a 50 ohm load on an external BNC T connector to prevent reflections.

From the GPI section of the schematic



Associated API calls for the GPI inputs:

GPI inputs are configured with the "**Tsync_GI_"** commands. Refer to the TSync-PCIe driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121 for more info on associated API calls.

- **GI_GetEdge** and **GI_SetEdge** are for defining which edge is active
- **GI_GetValue** gets the current input value
- **GI_GetTSEnable** and **GI_SetTSEnable** gets/sets the timestamp enable state when time stamping input changes.
- **GI_GetNumInst** gets the number of GPI pins present in the system

Pin Enable: unlike GPO pins, GPI pins do not need to be "enabled" via an API call/example program.

Q. Can we individually compensate the propagation delays to each GI input channel?
A. There are no delay commands for the GPI input pins

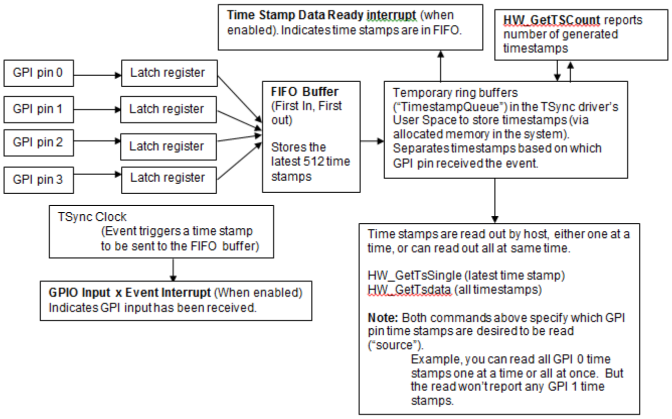
**Desire to have an event on GPI (input) pin generate timestamps (known as timetag)

- **Refer to**: TSync-PCIe time tag (timestamping/Time Stamping/Timetag/Time Tag)/External Event Input (right below)

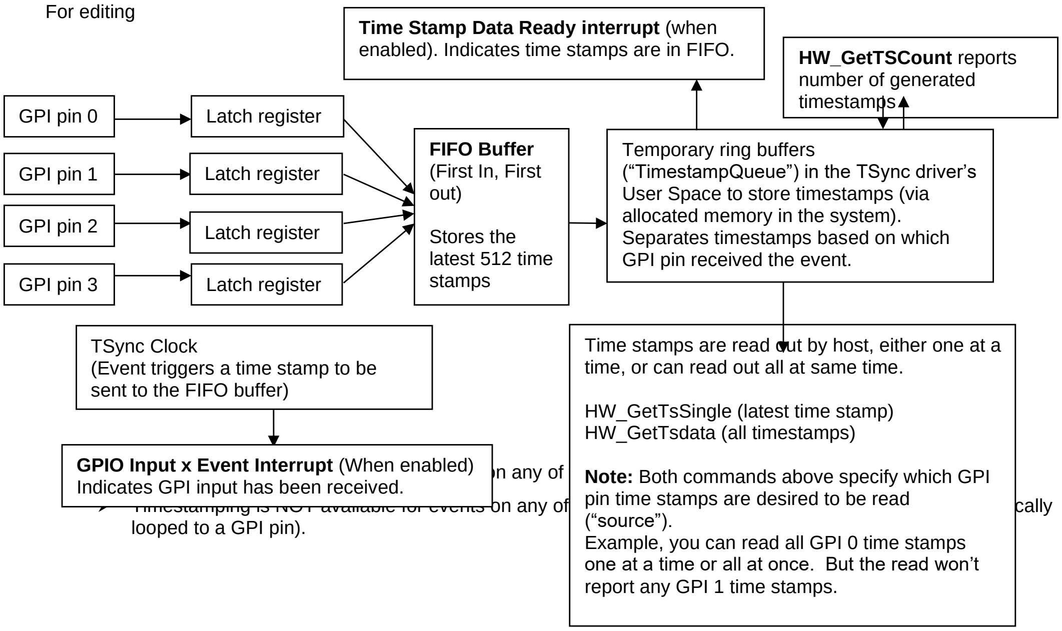
Time tag (timestamping/Time Stamping/Timetag) / External Event Input

- Refer to Section 5 (Example Routines) section in the driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121for more info on the External Event Input.

Note: in Rev J of the Driver guide- it's in Section 5.2 on pages 5-239 and 5-240.



For editing



Time Stamp Data Ready interrupt (when enabled). Indicates time stamps are in FIFO.

HW_GetTSCount reports number of generated timestamps

GPI pin 0

GPI pin 1

GPI pin 2

GPI pin 3

Latch register

Latch register

Latch register

Latch register

FIFO Buffer (First In, First out)

Stores the latest 512 time stamps

Temporary ring buffers ("TimestampQueue") in the TSync driver's User Space to store timestamps (via allocated memory in the system). Separates timestamps based on which GPI pin received the event.

TSync Clock (Event triggers a time stamp to be sent to the FIFO buffer)

GPIO Input x Event Interrupt (When enabled) Indicates GPI input has been received.

Time stamps are read out by host, either one at a time, or can read out all at same time.

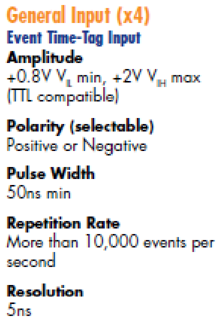
HW_GetTsSingle (latest time stamp)
HW_GetTsdata (all timestamps)

Note: Both commands above specify which GPI pin time stamps are desired to be read ("source").
Example, you can read all GPI 0 time stamps one at a time or all at once. But the read won't report any GPI 1 time stamps.

n any of

- Timestamping is NOT available for events on any of cally looped to a GPI pin).

Specs for the GPI pins (General Purpose Inputs- “GPI”) from the TSync data sheet



General Input (x4)

Event Time-Tag Input

Amplitude
+0.8V V_{IL} min, +2V V_{IH} max
(TTL compatible)

Polarity (selectable)
Positive or Negative

Pulse Width
50ns min

Repetition Rate
More than 10,000 events per second

Resolution
5ns

Precision/accuracy of generated timestamps

- Clock accuracy is +/- 5ns to the input, (but also accuracy to UTC is also dependant upom the accuracy of the input reference (GPS, IRIG, etc) to UTC

Per Denis Reilly (15 Nov 17) All Kramden products have a single hardware time counter that is used for all timing signals, including GPI events. This hardware time counter runs **at 200 MHz, so it’s precision is 5 ns.**

Accuracy depends on the quality of the reference, of course, but there is nothing different about the Kramden GPI vs. any other Kramden element that makes accuracy better or worse. So you can quote whatever accuracy numbers are appropriate for the reference.

Keith’s response to customer (15 Nov 17) Thanks for your voice message and inquiry about the typical accuracy of the timestamps generated when the TSync-PCI timing board receives an input event.

The accuracy of the generated timestamps (in relation to the TSync’s internal clock) is **+/- 5ns** of the configured active edge (either the rising or falling edge) of the incoming event signal.

Therefore, accuracy of this generated timestamp to “true UTC time”, is this +/-5 ns precision to the internal clock, plus the accuracy of the internal clock to its selected input reference (such as GPS). This value is the reported 1PPS phase error value obtained with the XO_GetPhaseError 0 call. Plus the accuracy of the selected reference (such as GPS) to true UTC.

In summary: with a TSync board synced via GPS input, the TSync board’s internal clock will typically be within about 100ns of GPS time, which is typically within about +/- 50ns of UTC. This makes the typical accuracy of the generated timestamps to be around 155ns or so, of true UTC time (as long as the TSync-PCIe board is in sync with GPS when the timestamp was generated). Whether or not the board is in sync, the accuracy of the generated timestamp will typically be +/- 5ns of the TSync board’s internal clock time!

Note that how long it takes to read out the timestamp from the FIFO buffer (once the timestamp has already been generated) has no effect on the accuracy of the timestamp, because it was generated before being sent into the buffer.

API calls/example programs associated with Timestamping

Where:

- **hnd**= the handle assigned to each installed board (starting with 0)
- **pin** = the GPI pin number
- **en**= use **1** to enable OR use **0** to not to enable
- **edge**= use **0** to trigger on rising edge OR use **1** to trigger on falling edge
- **src** = the timestamp source information (src is 1 higher than the GPI pin number. For GPI 0, src is 1)

- **pObj =**pointer to the timestamp data result

Enabling overall timestamping

TSYNC_HW_SetTsEnable(hnd, en)
Example: **HW_SetTsEnable 0 1**

Where the 0 = hnd (which Tsync board)
Where the 1= en (1 to enable timestamping)

Selecting which GPI pin and enabling this pin for timestamping

TSYNC_GI_SetTsEnable(hnd, pin, en)

Example: **GI_SetTsEnable 0 0 1** (Enables GPIO Input 0 for timestamping)

- **Where the first 0** = hnd (which Tsync board)
- **Where the second 0** = pin (which GPI pin TO enable)
- **Where the 1** = en (1 to enable the pin)

Selecting which edge of the signal to trigger on

- ➢ TSYNC_GI_SetEdge(hnd, pin, edge);
- ➢ **Example**: **GI_SetEdge 0 0 1** (Enables GPIO Input 0 rising active edge)
- Where the first 0 = hnd (which Tsync board)
- Where the second 0 = pin (which GPI pin)
- Where the 1 = en (0 to enable the rising active edge) **Note**: use 1 for the falling edge

Reading time stamps (one at a time, or all at once)

A) TSYNC_HW_GetTsSingle(hnd, src, pObj)

- Example: **HW_GetTsSingle** (read out one at a time- oldest one in buffer)

TSYNC_HW_GetTsData(hnd, src, pObj)

- **Example**: **HW_GetTsData** (read out all timestamps in the buffer at the same time)

Counting input events / clearing the event counter

- ➢ TSYNC_HW_GetTsCount(hnd, src,);   (Count all events received on each of the GPI pins)
- ➢ Example: **HW_GetTsCount 0 1**

NOTE: Index 0 is for Host generated time stamps, so the src value is one higher than the GPI pin number.

- ➢ TSYNC_HW_SetTsClear(hnd, src); (Clears TsCount counter)

Example: **HW_SetTsClear 0 1**

NOTE: Index 0 is for Host generated time stamps, so the src value is one higher than the GPI pin number.

Time Stamp Data Interrupts associated with Timestamping that can be used with Spectracom drivers

- **Ready**: interrupt to indicate each time a timestamp is in the FIFO buffer (Buffer has become non-empty).
- **GPIO Input event**: interrupt to indicate each time an event has been received on a defined GPI pin.

Example summary of customer code used to enable overall timestamping, enable GPI 0 to perform timestamping, and trigger on rising edge of event input

```
sprintf( fullDevName, "%s%s%d", DEVICE_ROOT, devName, devIdx );
err = TSYNC_open(&hnd, fullDevName);
en = 1;
err = TSYNC_HW_SetTsEnable(hnd, en);
pin = ID_PIN_0;
err = TSYNC_GI_SetTsEnable(hnd, pin, en);
edge = EDGE_RISING;
err = TSYNC_GI_SetEdge(hnd, pin, edge);
src = TMSTMP_SRC_GPI_0;
err = TSYNC_HW_SetTsClear(hnd, src);
for (EVER) {
err = TSYNC_GI_GetValue(hnd, pin, &j);
err = TSYNC_HW_GetTsCount(hnd, src, &k);
printf("Sleeping %d %d %d\n", n,j,k);
sleep(1); }
```

Notes about timestamping:

- GPI pins are used for generating time stamps for external events (configure the GPI pin for timestamping)
- The TSync-PCIe board can support up to 4 external time tags at the same time (one on each of the four GPI pins).
- Supports incoming time stamps of up to **every 50ns** (supports up to **10,000 timestamps/second)**
- There is a separate register for each of the GPI inputs to latch the time in before being sent to the FIFO. This allows simultaneous receiving of events on more than one GPI pin.
- Once each time stamp has been read out, it's automatically deleted from the buffer in the driver (no additional steps are required to remove it from the buffer, other than reading it from the buffer).
- FIFO buffer (FIFO on the board, not the temporary buffer in the driver) can store up to **512 total time stamps.** Note this value is not configurable.
- What "512" is referring to is all time stamps are stored in a First In First Out buffer, no matter which GPI pin the event is received on. The buffer can store up to the last 512 time stamps before the oldest time stamp is pushed out of the buffer and discarded.

The time stamps can be individually read out of this buffer (starting with the oldest), or they can all be read out of the buffer at the same time.

- **HW_GetTsCount 0 1** (Gets the number of timestamps collected from GPIO input 0)

  NOTE: Index 0 is for Host generated time stamps, so the index value is one higher than the GPI pin number.

TSYNC_HW_GetTsCount(hnd, TMSTMP_SRC_GPI_x, uint *nCount); //check counts on each input to determine which one generated the interrupt.

- Time Stamp inputs can be configured for either rising or falling edge (not both edges).

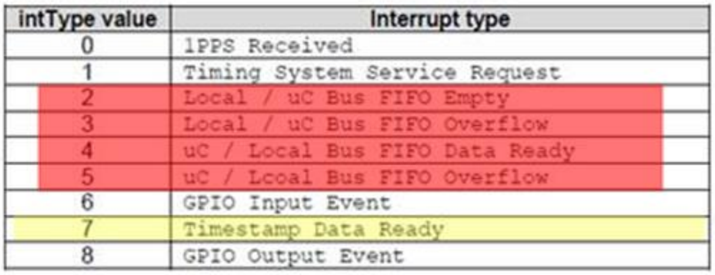
  Email from Dave Sohn: we don’t support triggering on both edges for timestamping

- Reading the time stamps, either all at once, (using **HW_GetTsdata**) or one at a time (using **HW_GetTsSingle**) clears the time stamps from the FIFO buffer (They aren’t stored in system memory).

- Use the “**Time Stamp Data Ready**” interrupt to indicate each time a timestamp is in the FIFO buffer (if an interrupt is desired when the timestamp is ready). As shown in the table below, this is the Type 7 interrupt.

  Email from Dave Sohn: This interrupt occurs when a timestamp has been taken and is available for the user to download.

Note: There are a four other interrupts that refer to the **Local / uC Bus** and **UC /Local bus** (IntType 2 through 5 showns below in red). These are not used for the timestamp FIFO buffer. They are related to the interface between the TSync-PCIe driver and the KTS timing system. Always use the “Time Stamp Data Ready” interrupt for this desired functionality. These are used when creating a custom driver. They are not used with the Spectracom driver.



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

**FIFO Buffer info (First In, First out)

- FIFO buffer can store up to a total of 512 total time stamps. Note this value is not user configurable.

Time stamping has a 512 deep FIFO buffer (First In / First Out) that covers all inputs. In the host driver, we maintain 512 deep FIFOs for timestamp inputs individually. So, as long as overall they can keep up with 2k/s reading of timestamps, they can allow up to 512 timestamps to be collected before reading.   After 512 entries, the oldest one will be discarded. What "512" is referring to is all time stamps are stored in a First In First Out buffer, no matter which GPI pin the event is received on. The buffer can store up to the last 512 time stamps before the oldest time stamp is pushed out of the buffer and discarded.

The time stamps can be individually read out of this buffer (starting with the oldest), or they can all be read out of the buffer at the same time.

Reading timestamps out of the FIFO buffer

- The GPI pin that the events were received on is referenced in the API call/ example program used to read out the timestamps (referred to as the "**index number**")

NOTE: **Index 0** is reserved for **Host-generated time stamps**, so the index value to reference when reading out timestamps is always **one higher** than the GPI pin name (GPI 0, GPI 1, etc.

Examples:

- Reading index "0" will not return any timestamps (its reserved for host timestamps in SecureSyncs- not used in TSync boards
- ***GPI pin 0***: The "index number" to use for reading out timestamps from GPI pin 0 will be "1" (not "0")
- ***GPI pin 1***: The "index number" to use for reading out timestamps from GPI pin 1 will be "2" (not "1")
- ***GPI pin 2***: The "index number" to use for reading out timestamps from GPI pin 2 will be "3" (not "2")
- ***GPI pin 3***: The "index number" to use for reading out timestamps from GPI pin 3 will be "4" (not "3")

- The actual timestamps do not reference which GPI pin the event was received on. Correlation of timestamps from multiple GPI pins needs to happen external to the TSync-PCIe board, based on the GPI referenced when they were all read out of the FIFO.

There are two commands that can be used to read out timestamps from the FIFO buffer:

- **HW_GetTsSingle 0 1 (Retrieve one collected time stamp from GPIO Input 0)**

  NOTE: Index 0 is for Host generated time stamps, so the index value is one higher than the GPI pin number

- **HW_GetTsData 0 1 (Retrieve all collected time stamps from GPIO Input 0)**

  NOTE: **Index 0** is reserved for **Host-generated time stamps**, so the index value to reference when reading out timestamps is always one higher than the GPI pin number (

Note The GetTsSingle command will only grab the "oldest" time stamp out of the FIFO buffer, one at a time.   The getTSdata command will allow the retrieval of all time stamps in the buffer instead of just retrieving one. Refer to page 4-68 of the driver guide for more information on this command.

(Example timestamps below)

```
250, 2009 02:25:29.979942255, Sync
250, 2009 02:25:30.188161995, Sync
```

```
250, 2009 02:25:30.396381735, Sync
250, 2009 02:25:30.604601480, Sync
250, 2009 02:25:30.812821220, Sync
250, 2009 02:25:31.021040965, Sync
250, 2009 02:25:31.229260705, Sync
250, 2009 02:25:31.437480445, Sync
```

Desire to detect a FIFO buffer overflow condition has occurred.

- Refer to Salesforce case 11409. https://na8.salesforce.com/500C000000TEp4m?srPos=0&srKp=500
- Recommended to use the HW_GetTsCount calls to obtain a count of all incoming events on each GPI pin. Then verify this count against all of the time stamps read out of the FIFO buffer for each GPI pin. If they are the same, no time stamps were lost/overwritten.

Counting the number of Events received on each GPI pin ("HW_GetTsCount")

- These two commands were intentionally added to the driver for a customer that was concerned about FIFO buffer overflow due to not being able to read out the timestamps fast enough.
- **TSYNC_HW_GetTsCount()**: Get the number of collected timestamps for the specified * source
- **HW_GetTsCount 0 1** (Gets number of timestamps collected from GPIO Input 0) NOTE: Index 0 is for Host generated time stamps, so the index value is one higher than the GPI pin number.
- HW_TsClear (Clears the TsCount counter)

Desire for a larger FIFO buffer than the default 512 buffer

Q. Can the FIFO size be increased?
A. Reply from Keith (14 May 13) Regarding the desire of the buffer to be increased, if the time stamps are essentially being consistently generated, as long as you are reading them out regularly, the buffer size shouldn't be a problem. The only concern is if you had a very large burst of time stamps all at the same time. They may not all be able to read out quickly enough before they were overwritten.

Increasing the buffer size, if large bursts of timestamps that could exceed the buffer size are of a concern, changing the buffer size could potentially be something that we consider as a special modification of the timing board. Please let Mike Messina, the Sales Account Manager, know if this is change that you would like to pursue. He can work with our Product Management and Engineering teams to let you know what this would entail. I have copied him in on this email for you. In case you don't have his contact info, Mike can be reached at (585) 321-5850 or via email at mikem@spectracom.orolia.com.

Obtaining timestamps

TSYNC_SS_GetTimestamp: Get the specified state change timestamp.

Triggering on Event input

Q Email from Edwin Pease (1/11/12) Triggering on both the rising and falling edges would be a nice feature.
A Reply from Dave Sohn We don't support triggering on both edges for timestamping.

Time stamping has a 512 deep FIFO buffer (First In / First Out) that covers all inputs. In the host driver, we maintain 512 deep FIFOs for timestamp inputs individually. So, as long as overall they can keep up with 2k/s reading of timestamps, they can allow up to 512 timestamps to be collected before reading. After 512 entries, the oldest one will be discarded.

The **waitFor** function will unblock on an event occurring, not when there is data in the buffer. Because of this, using the TsData command instead of the TsSingle command will be a better option to use (it will grab all timestamps that may be contained in the buffer by the time they are able to be retrieved instead of just the oldest timestamp).

Note: The events for the GPI0 input are read-out using get time 1 (not get time 0). GPIO0 is actually readout as get time 1 instead. The reason for this offset is "0" is used to store/retrieve the host time stamps (get time 0 gets host time stamps). So to retrieve GPI1 events, it is a get time 2. Gettime 5 is for support of legacy boards. It captures and combines time stamps of the host (0) and the GPI0 input into one buffer readout.

Q. It says below that after 512 event timestamps the oldest is overwritten. Any way to stop when it's full? My implementation is I have this card set up and waiting for an event. When the event occurs there will be more than 512 events very quickly. I just need the timestamp from the first event. I could write a loop that says HW_GetTsSingle 0 1, nothing there, HW_GetTsSingle 0 1, nothing there, HW_GetTsSingle 0 1 – ok got something this must be the first event. I would prefer a more deterministic approach if possible.

(From Dave Sohn 8/20/10): It is possible to be interrupted when timestamps are taken. He could then disable timestamping, read one timestamp, clear the timestamps, re-enable timestamping and then wait for the next event interrupt. Below are the APIs to he would use:

Setting active edge for input events for a given general purpose input

```
TSYNC_GI_SetEdge(
   TSYNC_BoardHandle hnd,
   ID_PIN index,
   EDGE edge)
```

Where:
0 = Falling edge
1 = Rising edge
2 = Both edges (**Note**: As also mentioned in Triggering on event input, we don't support triggering on both edges. An "opt" error message is displayed when selecting this value)**.**

Enabling/disabling timestamps on input events for a given general purpose input

```
TSYNC_GI_SetTsEnable(
   TSYNC_BoardHandle hnd,
   ID_PIN index,
   int bEnable)
```

Enabling/disabling interrupt mask for the timestamp event

```
TSYNC_HW_SetIntMask(
   TSYNC_BoardHandle handle,
   INT_TYPE intType,
   unsigned int index,
   int bEnable)
```

Waiting for the timestamp event input

```
TSYNC_waitFor(
   TSYNC_BoardHandle handle,
   INT_TYPE intType,
   uint32_t index)
```

Enabling/disabling hardware timestamping

```
TSYNC_HW_SetTsEnable(
   TSYNC_BoardHandle handle,
   int bEnable)
```

Retrieve a single timestamp

```
TSYNC_HW_GetTsSingle(
   TSYNC_BoardHandle handle,
   TMSTMP_SRC source,
   TSYNC_HWTimeObj *pObj)
```

Clear remaining timestamps in FIFO

```
TSYNC_HW_SetTsClear(
   TSYNC_BoardHandle handle,
   TMSTMP_SRC source)
```

Q. How much time it usually take the board to store a new time stamp to the FIFO buffer?

A. From Dave Sohn As far as timing between the event occurrence and timestamp availability, I don't know offhand, but is less than 100ns.

Issues with Time Stamping

- Issues with timestamping only when PC is connected to a network (works fine with Ethernet port not connected) OR if other PCIe devices affect the operation of the Tsync board
- Network Interface Cards (NICs) are usually PCIe-based, just like the TSync board
- Due to topography of the PCIe bus/slots, the NIC is likely affecting the TSync board.
- Get the specs of the computer (Manufacturer/Model) to look at the system to see if the NIC and TSync board are on the same PCIe bus.

Email from Denis Reilly regarding a customer having an issue with time stamping when PC is connected to a network (14 Nov 2014) Just so you guys understand, they are using PICMG 1.3 (also known as SHB Express), which is a scheme where there is a main processor board that plugs into a special slot on a backplane board. Other cards plug into normal slots on this backplane board as well.

It may make a difference which backplane / sister board they are connected to, since we need to know which slots they are using and what they might be connected to on the main processor card.

**Desire to have an event on GPI (input) pin, generate an output on a GPO pin

- (15 Nov 2012) TSync-PCIe customer (Salesforce case 6656) wanted to have an event input on GPI pin cause an event output to occur on the GPO (output) pins, to trigger multiple camera shutters at the same time.

Per Dave Sohn- There is no way for the GPI pins to generate events on the GPO pins. Refer to GPO information above about what functions the GPO pins can perform.

**GPO pins (General Purpose Outputs- “GPO”)

- Four GPO pins are available (GPO 0 through GPO 3) on the Timing connector (in order- pins 3, 16, 1, 14).

Function:

Each GPO pin can be configured to (as described in more detail further below):

- **Operate in Direct Value mode (signal level changed at user’s discretion).**

  In **Direct mode**, an event is triggered when the output Value in the GPIO output control / status register is changed and creates the active edge selected by the GPIO direct mode output interrupt active edge bit in that same register. This can be used to generate a “software” interrupt by setting the GPIO output appropriately.

- **Output a Square wave signal (1Hz to 10 MHz) (not sine wave).**

  In **square wave** mode, an interrupt is generated whenever the GPIO output generates the active edge as selected by the GPIO output square wave active edge bit in the GPIO output control / status register. This can be used to generate a periodic interrupt at the rate of the square wave.

- **Operate as Match Time Event pin (become active at one time and inactive at another time).**

  In **match time** mode, an interrupt is generated whenever the GPIO output high match time or GPIO output low match time registers are enabled and subsequently matched against the current system time.

- **Generate interrupts at a particular interval.**

  Q. Is there a way to command an output pulse (on GPIO output) to start at a time of my choosing and have a frequency that I specify? I’d like to create a 10 msec pulse but I need to control when that pulse starts

  A (Dave Sohn, 8 Jan 2014) You can generate a single 10ms pulse to start at their time of choosing by setting up and enabling both a match high and match low with 10 msec in between.

Using a GPO pin as the dedicated Heartbeat output from a TPRO/TSAT board

- For info on using a GPO pin as the dedicated Heartbeat output from a TPRO/TSAT board (when replacing a legacy TPRO/TSAT board with a TSync board), refer also to (in this doc): **Replacing an existing TPRO/TSAT board with a TSync-PCIe board

Output impedance of GPO pins:

- The GPO pins (General Purpose Output) are high impedance.

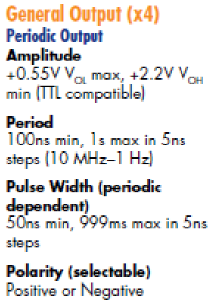
GPO Output specs

- **Level/drive: (Tsync-PCIE only)** 2.2V min, into high impedance (not 50 ohms, as it is with the other TSync boards in the family)

  Note the four GPO output levels/drive capability are smaller on the TSync-PCIe board (2.2v min,TTL) than it is on the other Tsync family of board (such as TSync-PMC, CPCI, etc) (at 4.3v min into 50 ohms) as indicated on the data sheet

- **Period**: The output period of each GPO pin can be configured for any range between 1 Hz and 10MHz.
- Each of the four GPO can be individually programmed to output digital signals such as 1PPS, 10PPS, 100PPS etc. via the ‘**period’** parameter (any frequency of 1Hz to 10MHz).
- **Pulse width** for each GPO can be individually programmed.

- **Active edge**: can be either positive or negative.



General Output (x4)

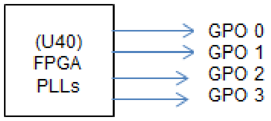
Periodic Output

Amplitude
+0.55V V_{OL} max, +2.2V V_{OH} min (TTL compatible)

Period
100ns min, 1s max in 5ns steps (10 MHz–1 Hz)

Pulse Width (periodic dependent)
50ns min, 999ms max in 5ns steps

Polarity (selectable)
Positive or Negative



Accuracies of GPO pins when using 1PPS input from a source other than the TSync's GPS receiver.

- Per Dave Sohn (Sept 2019),we cannot spec 1PPS accuracies of a TSync board not syncing to its own GNSS/GPS receiver, as the 1PPS source's accuracy and jitter are not known.

Email Keith sent to Acquisys (9 Sept 2019) Thanks for your great inquiry about syncing a TSync-PCIe board using an external 1PPS (Referred to as the "epp 0" input reference). My apologies for the delay in being able to get back to you, as I needed to discuss your email with the TSync board's Product Manager, here in the US.
Unlike when using a GPS receiver to sync a TSync-PCIe board, unfortunately we are not able to provide any typical accuracy specs for a 1PPSbeing received from another source. With a GNSS/GPS receiver's 1PPS output to the TSync-PCIe board, we know the accuracy and jitter characteristics of that 1PPS input. But we don't know what the 1PPS jitter characteristics are, when the 1PPS input signal is being applied by a customer's 1PPS generator. The 1PPS jitter of the signal is a large factor in how closely aligned the TSync-PCIe board's outputs will be, to the incoming 1PPS signal

Please keep in mind that the TSync-PCIe board does not try to "keep up and match" any the jitter of the incoming 1PPS signal. Instead, the TSync board's own internal 1PPS is "de-coupled" from the input PPS. The TSync-PCIe board only coarse adjusts to the input 1PPS signal the first time it's applied after each boot-up of the TSync board – after that, the system 1PPS is very slowly slewed into alignment with the input PPS. If there is a lot of jitter on the input PPS, the TSync-PCIe's internal 1PPS is not going to have similar jitter, because of this very slow disciplining. This helps significantly dampen out excessive jitter on the input PPS from also being observed on the system PPS. With jitter on the input PPS, and the very slow disciplining, the result will be larger offsets between the input 1PPS and the internal system PPS. But with very little jitter on the input PPS, the internal PPS will be better able to be more closely aligned to the input PPS.

The Product Manager did mention that if your customer's 1PPS source's 1PPS signal has similar (or better) accuracy and as low (or even less) jitter than observed with the TSync's own GNSS receiver, the accuracy specs provided in the TSync's datasheet (attached) will be similar to those that your customer can expect to see when using their own supplied 1PPS (the typical accuracy/jitter with the GPS receiver's 1PPS is +/- 25ns). But if the received 1PPS signal has less accuracy/more jitter than the 1PPS outputted from the GPS receiver, the accuracies your customer can expect to see will be inherently less than those provided in the datasheet, depending on the level of jitter on the 1PPS signal.

Need for Premium Breakout cable when using GPOs

- The standard breakout cable only provides connection to the first two GPO pins (GPO 0 and GPO 1). The Premium breakout cable is required if three or all four GPO's are needed.

Connecting GPOs to GPIs ("Outputs" to "Inputs")

- The GPO (and GPI) signals aren't available on the PCB board itself without going out/in through the Timing connector (There is no other connector on the board besides the timing connector to pull off the GPO signal).

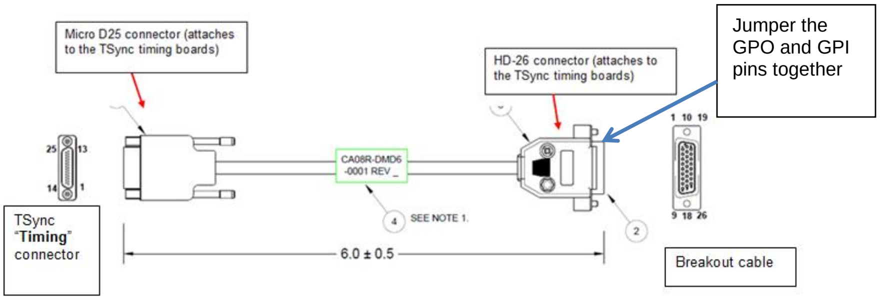
GPO outputs can be looped back into the GPIs:

- Via the Standard or Premium breakout cable (only two of the four GPO and GPI connectors are available on the standard breakout cable- Premium is required when using all for GPOs). Connect the two DB connectors on the breakout cable together with a gender changer.
- Via the short adapter cable (CA08R-DMD6-0001) that attaches the breakout cable to the newer Tsync boards. Make a custom jumper on a connector at the end of this short adapter cable. Refer to the info below about this cable

CA08R-DMD6-0001 (Micro-D25 to HD26 adapter cable)

Note: This adapter cable ships standard with both Standard and Premium breakout cables (not included in the anc kit- shipped separately)

- ➢ Link to schematic (**CA08R-DMD6-0001**) in Arena: https://app.bom.com/items/detail-spec
- ➢ Installed between TSync board and the newer design breakout cables (which started shipping around Nov 2013)
- ➢ Micro-D25 end of adapter attaches to TSync's Timing connector.
- ➢ HD (High Density) 26 end attaches to the new design breakout cable


Micro D25 connector (attaches to the TSync timing boards)
HD-26 connector (attaches to the TSync timing boards)
Jumper the GPO and GPI pins together
25
13
14
1
CA08R-DMD6 -0001 REV _
4
SEE NOTE 1.
2
1 10 19
9 18 26
TSync "Timing" connector
6.0 ± 0.5
Breakout cable

GPO Alignment to System PPS and to each other:

- ➢ The GPO outputs and the 1PPS output (on-time point) should be within +/- 10ns of each other.

Jitter of the GPO outputs:

Q. We are considering to set one of the TSync's GPIO outputs to 10MHz square wave and use this as an **FPGA PLL** input reference clock on our hardware (Altera Arria II GX). Is this possible and a recommended design practice? What maximum jitter is specified for the 10MHz square wave?

A (Per Denis Reilly) using the GPO pins as an input to FPGA PLLs would be a cascading of FPGA PLLs, because the GPO pins are derived from FPGA PLLs.

The 10 MHz sine wave output from the TSync board meets specs we provide in the data sheet. This output is derived directly from the output of the 10 MHz oscillator. However, the GPO outputs are derived from FPGA PLLs (FPGA logic). We do not provide any jitter specs for the GPO pins. The performance will be less than that of the 10 MHz output, but we don't spec by how much.

The 10 MHz sine wave output would be a much better signal to use than the GPO output, as its derived directly from the oscillator. The TSync's GPO outputs are derived from FPGA PLLs, so using the GPO to generate a 10 MHz square for FPGA PLL synchronization would simply be a "cascading" of FPGA PLLs (from one PLL to another PLL). Because the jitter/stability of the GPO outputs is inherently less than that of the 10MHz outputs, we don't provide any jitter specs on these particular outputs. They can certainly generate a 10MHz square wave from the GPO for this function. But we can't say what level of jitter they are going to see on their input.

If they are concerned by jitter being present on the 10 MHz signal, they should consider instead using the disciplined 10 MHz sine wave output as derived directly from the oscillator. However, to use this sine wave output, they will likely need to convert the sine wave to square wave external to the TSync board.

GPO API calls:

- GPO pins are configured with the "**Tsync_GO_"** commands.
- Refer to the TSync-PCIe driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121 for more info on associated API calls.

- GPO pins are enabled with **TSYNC_GO_SetEnable** API call.
- Output mode is configured with the **TSYNC_GO_SetMode** API call.
- **TSYNC_GO_GetValue:** Gets the GPOs current output value.
- Can enable Signature Control of GPO pins **(TSYNC_GO_SetSigCtrl)**
- **TSYNC_GO_SetSWOtpPW:** Set the GPO's square wave OTP pulse width. Pulse width is in nanoseconds from 20 nsec to 900,000,000 nsec.

  Note: ths call requires driver version 3.1.0 (or higher) be installed.

Function:

Each GPO pin can be configured to (as described in more detail further below):

- **Operate in Direct Value mode (signal level changed at user's discretion).**

  In **Direct mode**, an event is triggered when the output Value in the GPIO output control / status register is changed and creates the active edge selected by the GPIO direct mode output interrupt active edge bit in that same register. This can be used to generate a "software" interrupt by setting the GPIO output appropriately.

- **Output a Square wave signal (1Hz to 10 MHz) (not sine wave).**

  In **square wave** mode, an interrupt is generated whenever the GPIO output generates the active edge as selected by the GPIO output square wave active edge bit in the GPIO output control / status register. This can be used to generate a periodic interrupt at the rate of the square wave.

- **Operate as Match Time Event pin (become active at one time and inactive at another time).**

  In **match time** mode, an interrupt is generated whenever the GPIO output high match time or GPIO output low match time registers are enabled and subsequently matched against the current system time.

- **Generate interrupts at a particular interval.**

  R. Is there a way to command an output pulse (on GPIO output) to start at a time of my choosing and have a frequency that I specify? I’d like to create a 10 msec pulse but I need to control when that pulse starts

A (Dave Sohn, 8 Jan 2014) You can generate a single 10ms pulse to start at their time of choosing by setting up and enabling both a match high and match low with 10 msec in between.

Q. I am on Linux and when I search the full set of tsync directories that were created when I followed the installation intructions, I only find GO_VALUE_RECIPE in the tsync_Go.c file where it is an extern so it would need to actually be declared somewhere else. Perhaps GO_VALUE_RECIPE is in the driver though that seems to be rather unconventional driver code. If it is in the driver then it is not clear how to link to get access to that variable and your manuals have no instructions on how to link to the driver.

A. GO_VALUE_RECIPE is an extern in the tsync_Go.c file and defined in the tsync_Go_Services_recipes.c file by the line:

```
RECIPE(GO_VALUE)
```

The GO_VALUE fields are defined in the tsync_Go_Services.h file:

```
#define GO_VALUE_FIELDS                  \
            TSYNC_X(      uint32_t,   value)
```

Default state of the GPOs

Q. What is the default state of the General Purpose Outputs on power-up, before any calls are made?

A. All general purpose outputs default to low outputs at system startup.

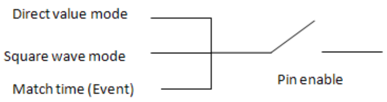
Ability to Start/Stop the GPO outputs

Q.Is there any way to start/stop a square wave output on a GPO pin at a given absolute time? The only thing I can think of is to use a match-time to generate an interrupt and then enable the GPO pin's output in response to the interrupt (assuming that the GPO is already set up for the square wave). Although, this method wastes a GPO pin just to generate the interrupt.

A. They are correct that there is no way currently to stop/start the GPO pin at a given absolute time. Their idea of using the match-time was the same idea we had. The only other think I could think of would be for them to use some external logic and use the match time pin to turn on and off the square wave.

GPO pin enable/disable

- GPO pin can be used to determine when the output signal will be present and when it’s not present.
- The output is present as soon as the pin is enabled (whatever state the configured GPO signal happens to be in, at that particular moment.



API call/example program to enable GPO pin 0: GO_SetEnable 0 0 1 (GPIO “0” enabled, where the second 0 indicates the name of the signal pin to enable and the 1 is to enable the pin)

FAQs about GPO pin enable (Email to Erik Johansson of NOAO 7/22/11)

Q. The pin is enabled and set in direct mode with an output value of 0 (low) and then disabled. Now I set the value to 1 but do not enable the pin. The output is still 0. At some later time I enable the pin. The output now changes from 0 to 1 as soon as the pin is enabled, correct?

A. Keith's reply: This is absolutely correct. In direct mode, with the pin disabled and with it changed from a 0 to a 1, the output won't change. But the output will change from a 0 to a 1 as soon as the pin is enabled.

The pin is set in square wave mode and is enabled for a while so that the output pulse train is active. I now disable the pin. The output will be high or low depending on the state of the square wave at the time the pin was disabled. I leave the pin disabled for some time. Now I enable the pin. The output immediately changes to whatever state the square wave would have been in had the pin remained enabled the entire time.

Keith's reply- Correct, in square wave mode and with the pin disabled, the 1PPS pulse train continues to be the same signal, but it's not present on the pin. When the signal is re-enabled, the output will go to the same place in the signal that it would be in had the pin not been disabled.

There is no "1PPS alignment change" of the signal when the pin is re-enabled. The 1PPS signal continues to occur, just like the pin had never been disabled to begin with. If the pin is re-enabled while the pulse is current active, part of the pulse width will be cut-off, so the pulse may have a smaller pulse-width than all of the others. If the signal is already a "high" when the pin is enabled, the output will start off as a "high" at that moment. This first active edge is not aligned to 1PPS.

In order to prevent this from occurring, we would recommend ensuring that the pin is not enabled at the on-time point itself. It should be enabled right after or any time before the next on-time point occurs. The 1PPS interrupt could be used to let you know that the on-time point has just occurred, so it's now OK to now enable the pin without the signal being currently active and preventing an un-aligned edge from occurring.

3. The pin is enabled and set to direct mode and driven to an output value of 0 and then disabled. The output remains at 0. The mode is now changed to a square wave with a rising edge and some pulse configuration. When I enable the pin, the first pulse will go active on the next coincidence of the 1 PPS (this assumes no offset) and then continue according to the pulse configuration.

Keith's reply- the pin enable /disable places no control on the signal as to the active edge being aligned to the on-time poin. If the mode is changed from direct mode to square wave mode with the pin disabled, the signal will start to be generated (the edge is aligned to the on-time point). When the pin is then enabled, the signal will be outputted as it is at that particular moment. The pin enable does not cause any delay in the signal so that the first active edge is aligned to the on-time point. If the pin is enabled any time after one pulse width has occurred and before the next active edge occurs, only then will the first edge be aligned to the on-time point.

As described above, if it's desired to prevent any false edges from occurring when the pin is first enabled, the pin should not be enabled during the active pulse. The enabling of the pin needs to be delayed until the pulse has returned to its non-active state and before the next active edge occurs. Otherwise, enabling the pin during the active state will cause the "active edge" to occur at the moment that the pin is enabled- not necessarily at the on-time point.

Modes of GPO pin operation

There are three available modes of GPO pins (each listed/described below)

- Direct Value mode (refer to "A" below)
- Square Wave mode (refer to "B" below)
- Match time mode (refer to "C" below)

A) Direct Value mode (DVM) (for GPO output pins)

- Direct Value Mode allows the user to configure the GPO pin to be a high or a low.

In **Direct mode**, an event is triggered when the output Value in the GPIO output control / status register is changed and creates the active edge selected by the GPIO direct mode output interrupt active edge bit in that same register. This can be used to generate a "software" interrupt by setting the GPIO output appropriately.

API calls/example Programs associated with direct value mode (DVM):

1. **Enable the specific GPO pin with GO_SetEnable 0 0 1 (GPIO "0" enabled)**

   The signal name of the specific output pin needs to be enabled. the following command enables GPO 0 output pin: **GO_SetEnable 0 0 1**

2. **Output mode is configured with the TSYNC_GO_SetMode API call.**

   Syntax: **GO_SetMode <device index> <pin> <mode> (example GO_SetMode 0 0 0)**
   Where <mode> **0** is Direct Value mode

   In order to operate in Direct Value mode, the particular signal pin name (such as GPO 0) needs to be set to DVM mode. The API call/example program to set GPO 0 to operate in DVM mode is GO_SetMode 0

3. **TSYNC_GO_SetDvmValue: Sets the GPOs direct value state**

   Syntax: **GO_SetDvmValue <device index> <pin> <val>**

   Where <value> **0** is to go low OR **1** is to go high

   The command to change the level of the output pin is **GO_SetDvmValue**

   - **Example command to set GPO 0 LOW: GO_SetDvmValue 0 0 0**
   - **Example command to set GPO 0 HIGH: GO_SetDvmValue 0 0 1**

4. **TSYNC_GO_GetDvmValue: Gets the GPOs currently set direct value state**

   Syntax: GO_GetValue <device index> <pin>

   Example to read curerent level of GPO 0: GO_GetValue 0 0

   Note: This command returns a 0 when low or a 1 when high.

5. **TSYNC_GO_GetValue: Gets the GPOs current value (independent of the mode)**

FAQs about DVM

Q. What is the Direct Value Mode for the GPO pins? It is not really described in the driver manual, except for the API description. Is this a way to use the GPO pins manually (i.e., setting directly by making driver calls)? The user manual does not describe direct value mode either, but discusses the possibility of configuring the GPOs as generic output pins (section 5.5.2). Is this the same as direct value mode?

A. Direct value mode allows a user to directly set the value presented on a GPO pin. You can manually set an output pin as either high or low when in this mode. The "mode" of the GPO must be set to "direct value" mode and the GPO must be enabled for it to work.

Q. Is there a difference between TSYNC_GO_GetValue and TSYNC_GO_GetDvmValue? I am guessing there is and that TSYNC_GO_GetValue returns the current value of the specified GO pin (for any configured output mode), whereas TSYNC_GO_GetDvmValue returns the current "set" value for DVM mode. Is this correct

A. Get Value gets the current value of the GPO independent of the mode. Get DVM Value gets the currently set direct mode value.

Q: Will the GPIO pins output 5vdc. I need 5vdc for the application I am using the TSync board in. The manual says they are TTL compatible.

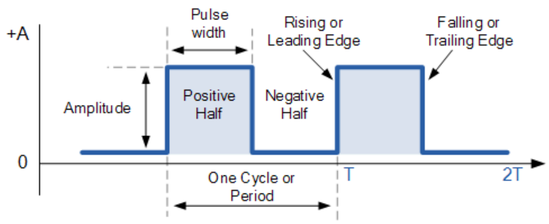
A: To answer your question about the TSync timing board outputs, the input and output levels are TTL compatible, as stated in the TSync manual.   The allowable input voltage range is 0 to 5.5vdc with a logic high being determined by about 2 vdc up to 5.5vdc. The output voltage of a logic high is guaranteed to be at least 2vdc (into high impedance), but can be as high as about 3vdc. The TTL output levels are never at 5vdc.

If 5vdc is needed when the output is “active”, you may be able to somehow convert the levels from 2 to 5vdc external to the TSync board itself.

B) Square Wave mode

- GPO pins can be configured to output a continuous square wave.

A Square Wave Waveform



In **square wave** mode, an interrupt is generated whenever the GPO output generates the active edge as selected by the GPIO output square wave active edge bit in the GPIO output control / status register. This can be used to generate a periodic interrupt at the rate of the square wave.

Conversion of "frequency" to "period"

Note: A good website to convert frequency to period**:** http://www.calctool.org/CALC/other/converters/freq

API calls associated with square wave mode output (DVM):

1. **Enable the specific GPO pin with GO_SetEnable 0 0 1 (GPIO "0" enabled)**

   The signal name of the specific output pin needs to be enabled. The following command enables GPO 0 output pin: **GO_SetEnable 0 0 1**

2. **Output mode is configured with the TSYNC_GO_SetMode API call.**

   Syntax: GO_SetMode <device index> <pin> <mode>

   Where <mode> **2** is for Square Wave mode

3. **Set square wave configuration Format TSYNC_GO_SetSqWave:**

   Syntax: GO_SetSqWave <device index> <pin > <offset> <period> <period scale> <pulse width><active edge>

   (Note: Offset, period, and duty cycle are in nanoseconds)

Examples:

./GO_SetSqWave 0 0 0 5000000 1 1000000 1 (GPIO "0", 0 offset, 500msec period, "microsecond" scale, 1msec pulse width, positive pulse)

Where

<pin> This is the desired GPO output pin number. It can be a number between 0 and 3.

<offset> * Offset is from -500 msec to +500 msec. (entered in nanoseconds)

<period> Period is in either * nanoseconds or * microseconds depending on <period scale> setting:

from 100 nsec to 20 sec in **nanosecond** scale ("period scale" field = "0")

from 100 usec to 20,000 sec in **microsecond** scale. ("period scale" field = "1")

<period scale>

Note: "period scale" value determines how the "period" value above is entered.
Enter "**0**" if configuring the "period" in **nanoseconds**.
Enter "**1**" if configuring the "period" in **microseconds**

<pulse width> is a value from 10 nsec to 999,999,990 nsec (entered in nanoseconds)

<active edge> is 0= Falling edge, 1= Rising edge 2= both (uncommon)

4. **TSYNC_GO_GetSquareWave: Gets the current squarewave configuration**

Examples:

1. **Desire to generate a 500ms (2Hz) squarewave on GPO pin 0.**

   To generate an interrupt at a desired interval, the GPIO Output Event Interrupt (type 8) should be used. One of the GPIO pins (such as GPIO pin 0) needs to be configured for 500ms square wave output (per your desired ½ second interrupt interval).

2. **To configure the GPO pin 0 for a 500msec square wave, 1 msec pulse width, positive-going pulse, use the following API calls (or example programs):**

   Note: The square wave mode has a limit on the pulse width of 900ms (900 milliseconds). If it's required to have a wider pulse width wider than 900ms, look at using Match time output mode instead of Square wave mode. This will require them to write their own software in order to constantly change the start and stop times.

   ./GO_SetMode 0 0 2 (GPIO "0" set to square wave mode)

   ./GO_SetSqWave 0 0 0 5000000 1 1000000 1 (GPIO "0", 0 offset, 500msec period, usec scale, 1msec pulse width, positive pulse)

   GO_SetEnable 0 0 1 (GPIO "0" enabled)

3. **Desire to generate a 10 MHz (100ns) squarewave on GPO pin 0.**

   ./GO_SetMode 0 0 2 GPIO "0" set to square wave mode)

   ./GO_SetSqWave 0 0 0 100 0 50 1 (For a 10MHz squarewave outputted from GPO pin 0, no offset from the System PPS, 50ns pulse width, with the rising (first) edge of the square wave aligned with the System PPS:

   GO_SetEnable 0 0 1 (GPIO "0" enabled)

**Desire to convert the GPO outputs to 1PPS outputs

- Refer to the TSync-PCIe driver guide in Arena: https://app.bom.com/items/detail-spec

Note: Be aware that there are some considerations to take into account when performing this operation. The square wave output as currently designed is only aligned to the 1PPS once when configured. That means that during initial alignment of the board's internal 1PPS to an input using HW smoothing, before the SW takes over with disciplining, the square wave outputs lose their alignment. Once SW takes over, which can be determined by looking at the disciplining state, if a square wave is setup as a 1PPS, it will always be aligned from that point on.

Email from Tim Tetreault to Insup to have 1PPS on GPO 0 (1/27/12)

Here is what you need to do:

1. Setup Pin0 to a "square wave" **:./GO_SetMode 0 0 2**
2. Enable Pin0 output: **./GO_SetEnable 0 0 1**

The default settings for the square wave output are 1PPS with a positive pulse width of 200msec.
If you want to change the frequency or pulse width, use the command:

"GO_SetSqWave A B C D E F G"

Where:
A = card index
B = I/O pin
C = offset from 1pps in nsec
D = period
E = scale for period 0 = nsec, 1 = usec
F = pulse width in nsec
 = 0 is neg pulse, 1 is pos pulse

So for example, if you want GPIO "0" to be set up for a 1pps, 100msec pulse width, positive pulse:

./GO_SetSqWave 0 0 0 1000000 1 100000000 1

Desire to convert the GPO outputs to 1PPM outputs

- Refer to Salesforce Case **271961**
- Per Ron Dries "The TSync board can output a 1PPM. They would need to set the period to **60000000** microseconds."

Keith modified the Email above from Tim Tetreault, to to have 1PPM on GPO 0 (1/27/12)

Here is what you need to do:

1. Setup Pin0 to a "square wave" **./GO_SetMode 0 0 2**
2. Enable Pin0 output: **./GO_SetEnable 0 0 1**

The default settings for the square wave output are 1PPS with a positive pulse width of 200msec.
If you want to change the frequency or pulse width, use the command:

"GO_SetSqWave A B C D E F G"

Where:
A = card index
B = I/O pin
C = offset from 1pps in nsec
D = period

E = scale for period 0 = nsec, 1 = usec
F = pulse width in nsec
= 0 is neg pulse, 1 is pos pulse

So for example, if you want GPIO “0” to be set up for a 1ppm. 100msec pulse width, positive pulse:

The command is ./GO_SetSqWave 0 0 0 **60000000** 1 100000000 1

FAQs about GPO pins

Q, I am confused about the synchronization of a square wave pulse on a GPO pin to the 1 PPS and how that relates to when the GPO pin is first enabled. Let's say I program a square wave pulse and set the GPO mode for square wave, but leave the pin disabled. Later on I come back and enable the GPO pin at some arbitrary time "T". When does the first active edge of the square wave occur? Is it coincident with the next 1PPS transition (assuming no offset) or does it occur at the next time that will allow for an eventual synchronization with 1 PPS?

I guess another way to ask the question is: if I program a square wave pulse, when does the active edge that is coincident with 1PPS occur relative to when I enable the output on the pin?

A, When the GPO pin is disabled and then re-enabled sometime thereafter, it should return to the same state that it was in when it was last stopped. From a signal perspective, the next active edge will be coincident (like it was never disabled). However, to the system that its connected to, depending on the previous state (which will be based on what state the signal is in at the time the pin is disabled), it could look like an active edge occurred as soon as the pin was re-enabled.

As this starting state could be in either state, depending on when the pin is disabled, a safer state, instead of enabling/disabling the pin, would be to switch the GPO between square wave mode to Direct Value Mode (DVM) when it’s desired to stop the output. Instead of just stopping the signal in a potentially unknown state, the DVM mode allows you to be able to command the pin to be either a constant low or a constant high. This can switch the square wave to a constant low for example, so that when the GPO is switched back to square wave mode, you can be certain on what state it’s starting at. Then, it should transition only when it’s coincident.

C) Match Time mode

- Refer to the TSync-PCIe driver guide for more info (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

Note: Section 5.3 gives an example configuration

Note: As of at least March, 2016 there is very little in the TSync user guide about Match, except for the two paragraphs below:

The General I/O is configurable as a Match Time Event pin, which will activate at a preset time and become inactive at another preset time. The Match Time Event provides two user settable times to make the General I/O pin active and inactive. The Match Time Event configured General I/O pin has a programmable edge, allowing the selection of “Low to High” or “High to Low”.

The General I/O output signals timing are accurate relative to the Input reference’s 1PPS signal to within +/- 50 nsec. The General I/O output has a programmable offset, which ranges from -500 msec to +500 msec in 5 nsec steps.

The following Example programs / CLI commands are utilized for match time:

GO_SetEnable 0 0 1 Enables the specific GPO pin (General enable of a general purpose output)
The signal name of the specific output pin needs to be enabled. The following command enables GPO 0 output pin: **GO_SetEnable 0 0**

4. **GO_SetMode**: **(Set the general purpose output to match time mode (use <mode> = 1 for Match time)**

   Syntax: GO_SetMode <device index> <pin> <mode> Where <mode> "**1**" is for Match Time mode

5. **GO_SetMatchEnable**: **(Enable/disable the time match of a general purpose output)**

   - Each output must be enabled for matching high level times and low level times.

   Syntax: GO_SetMatchEnable **<device index> <pin> <lvl> <enable>**

6. **HW_SetMatchTimeHi**: **(Set the time at which the specified general purpose output should go high. Time can be set up to 100 days ahead and as close as 50 msec.)**

   Syntax: HW_SetMatchTimeHi <device index> <gpo> <doy> <hr> <min> <sec> <ns>

7. **HW_SetMatchTimeLo:** **(Set the time at which the specified general purpose output should go low. Time can be set up to 100 days ahead and as close as 50 msec).**

   Syntax: HW_SetMatchTimeLo <device index> <gpo> <doy> <hr> <min> <sec> <ns**>**

Desire to use Match Time and have the GPO pin start off as a high after each time the board is started (instead of the default low that the pin goes to when the mode is selected)

The default setting for the GPO pins at board start-up is a low, until the first setMatchTimeLo occurs. However, it may instead be desired to have the GPO pin remain a high until the first match time occurs (the opposite). The Direct Value Mode (DVM) cannot be temporarily used to direct the pin high because when the mode is switched back from DVM to Match Time, the pin will inherently go back to a low.

To set the GPO pin high before the first real match occurs, keep the GPO pin disabled while performing a setMatchTimeHi (without performing a setMatchTimeLo afterwards). This will cause the pin to transition from a low to a high. Then, enable the GPO pin. At that point, the GPO pin will remain high until the first setMatchTimeLo time occurs.

I just spoke to one of our timing board engineers regarding your first question on Match Time. The reason that using the DVM mode to set the pin to high is not keeping the pin high after switching to Match time mode is because the mode switching is acting like a three-way switch (DVM, Match Time or pulse out). Each of the three modes starts out as a low, so while it it's in DVM mode, it can be set to high, but as soon as the "switch" is changed to Match Time, the pin then inherently goes to a low again. This is the reason that temporarily using the DVM mode won't provide you with the desired operation.

FYI- In addition to the "three-way mode switch", there is also a second "pin enable/disable switch" as well. The pin enable/disable switch applies to all three modes. If the GPO pin is not enabled, the GPO pin itself won't change states, even if a match time occurs. Once the pin is enabled, the effects of the selected mode will then be able to be sensed by the equipment connected to the pin.

To use the Match time mode and have the GPO pin start off with a "masked" starting transition of default low to a high ("masking" the transition from the default low to a high prevents this particular transition from being detected as a "false" match time occurring). We recommend performing a two-step process, in a specific order, right after the TSync-PCIe board is started.

The GPO pins are disabled at each boot-up. While the GPO pin is still disabled, switch the pin to Match Time mode. Then, use the setMatchTimeHi to have the pin go high, even before the pin has been enabled (by not following it with a setMatchTimeLo it will then remain high until the setMatchTimeLo is eventually performed). Then, once this Match has occurred, just enable the GPO pin.

From that point, the pin will be a high until the match time sets it low for the first time. And since the pin was disabled while it was being set to a starting high, the equipment won't detect the "setup" as a Match.

FAQs about Match Time

Q**.** I noticed that after I use setMatchTimeHi or setMatchTimeLo, and then read the info back using getMatchTimeHi or getMatchTimeLo, the value for the **year** returned by the API call is always "0001". Is this
a bug?

A (From Dave Sohn) Match time does not use the year information. Time matches can only be set within one year of the current time. The year information read back is not relevant and should be ignored.

Q. Is it possible to submit an array of match times to be used to generate a series of one-shot pulses on a GPO?

A. (Response from Tim Tetreault) Our TSYNC board does not generate a series of one-shot pulses like they are asking for. The only thing we have that could do something like that is our new CTC (Chronos) board that we are making for Lockheed Martin.

Q **how soon/how long before the MatchTime event is to occur, can the event be configured**

A (per the TSync driver gguide) Time can be set up to **100 days ahead** and **as close as 50 msec**

Customer request for Match time to be able to be configured even sooner than 50 msec before the desired Match Time

- **Refer to Salesforce Case** 178666

  "This customer wants to send a Match Time output faster than 50 mS in the future, like 10mS."

Desire to generate two interrupts at two different frequencies (with the interrupts synchronous to the PPS reference).

From Dave Sohn- The customer would want to use the GPIO Square Wave Output. They can set one GPIO Output to generate interrupts at a 50Hz rate and one at a 128Hz rate. The only caveat is that the square wave output will only remain aligned once we're disciplining to a reference. If you start up the board and the GPIO Outputs, and then connect your reference, it won't remain synchronized with the 1PPS. This has to do with how the first time alignment of the board to the reference and the alignment of the GPIO Output Square Wave occur. I can provide more detail into that if necessary.

In summary:

- The customer can set up two GPO Output Square Waves running at 50Hz and 128Hz.
- These Outputs can be setup to generate interrupts.
- If the outputs are setup after the board reports it is in sync, they will forever be aligned with the 1PPS even if sync is lost.

DRIVERS (Windows Linux Solaris)

API	Area of the board (Hardware, supervisory software, etc)	Description
CS_	Clock Service	Provides an abstract interface to the timing subsystem.
EC_	LED Component	Drive the LED indicators.
FP_	Frequency Output	Configures the output frequency.
FS_	Flash Manager Service	Provides access to the images stored in all flash memory devices.
GI_	General Purpose Input Component	Configure and monitor the general purpose input pins.
GO_	General Purpose Output Component	Configure and monitor the general purpose output pins.
GR_	GPS Reference Component	Execute the GPS receiver's protocol and determine 1PPS and time validity.
HA_	Host Agent	Obtain the capabilities of the TSync.
HR_	Host Reference	Uses information from the host PC to determine 1PPS and time validity.
HW_	Hardware	Provide access to the direct HW accessible control/status of the timing subsystem.
IN_	Initializer Service	Perform initial configuration and setup of each software module.
IP_	IRIG Output Component	Generate and output IRIG streams.
IR_	IRIG Reference Component	Control and process decoded IRIG input streams to determine 1PPS and time validity.
LS_	Log Service	Provides a queue for errors and maintains system alarms.
PP_	PPS Output Component	Control a 1Hz output.
PR_	PPS Reference Component	Monitor the 1PPS input reference.
PTR_	PTP Reference Component	Control and process decoded PTP network packets (either as an input reference or a time output).
RS_	Reference Monitor Service	Determine the best available input reference and maintain the reference priority table.
SS_	Supervisor Service	Maintain the time source, 1PPS source, Sync and Holdover states of the system.
US_	Upgrade Service	Upgrade the firmware and FPGA images in the external flash memories.
XO_	Oscillator Component	Analyze frequency measurements and make corrective adjustments to the timing system oscillator.
XS_	Oscillator Monitor Service	Measure and provide the accuracy and stability of the timing system oscillator.

TSync-PCIe Driver version/ driver updates

API call/example program to report the linux driver version

(4/24/17 KW) There isn’t a “TSync” call available to report the driver version. But the linux command of modinfo tsyncpci will report the linux driver version.

Open-source software/EULA software license for Spectracom TSync and TPRO/TSAT drivers

A) Licenses for TSync drivers

Refer to the "Software_Board_Driver pdf document for our EULA:

- In: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\EULA license for all drivers
- Or on our website at: https://www.orolia.com/document/eula-terms-and-conditions-for-timing-board-drivers/
- ~~(scroll down to "***EULA Terms and Conditions for TSync and TPRO/TSAT Timing Board Drivers***"):~~ ~~https://spectracom.com/products-services/product-index/documents/tprotsat~~

Q The driver is available in our website, but it is not clear for them if the driver is based on open-source code.
A from Tim Tetreault to Sylvain (21 Jan 15) We developed the TSync drivers ourselves but the Linux driver is considered to be "open-source". We do have an End User License agreement that we have on our web site that covers in more detail our software and drivers. See attachment (Note this reference to "attachment" is referring to the pdf document referenced above).

B) License request for the TSync's embedded firmware

- Refer to Salesforce case 25755 (June 2017)

Q We're trying to clean up our software trees and we need a license file for the tsync code. Preferably, we'd have a license file for the driver code (presumably GPL)and a license file for the user-level (library) code - BSD, LGPL?
A Reply from Keith (26 Jun 17): Thanks for your email pertaining to the TSync board's firmware licensing. Note I have assigned our Case Number **25755** to track your request.

I was just taking with our senior timing board engineer about your request. We were questioning why you would require a license for the embedded firmware installed in the TSync boards. We can certainly understand needing a license for the TSync board's linux driver, which is based on open source software, and as we provide the source code for this driver.

But the TSync board's firmware is proprietary and embedded. It is not open source software, and we do not provide the ability for customers to modify this firmware.

Can you please let us know why you are requesting a license file for the TSync firmware, especially if the details above aren't enough to satisfy your requirements. I will then pass this additional info over to our TSync Product Manager and Engineering team for their consideration. Thanks very much in advance!

Tsync-PTP driver limitations (Windows, Linux or Solaris)

- Due to more recent changes incorporated in the KTS Timing System for SecureSync-PTP functionality, the driver's PTP-associated API calls are no longer compatible with the TSync-PTP board's PTP module.
- Besides updating to any newer versions of the TSync-PCIe drivers (beyond version 2.5.1) breaking the "PTR" driver calls, there are no other issues updating the driver to a new version beyond version 2.5.1. If the customer isn't using the PTR API calls/example programs, they can update the driver without affecting anything else.

Highest driver versions still compatible with TSync-PTP's PTP module (newer driver versions break the PTP API calls)

- Shortcut to these particular drivers on our website for PTP boards: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=1543
  - **Linux:** Version 2.5.1

- **Windows:** Version 2.5.1
- **Solaris:** Version 2.5.1

Can a TSync-PTP customer use/update the Tsync driver to a more recent version (which no longer supports the PTP module)?

Question to ask: Is the customer using any of the PTP associated API calls/example programs (“PTR_” calls such as “PTR_ResetModule” for example)?

- **If the customer IS using any PTR calls/example programs:** They CAN upgrade the driver to any newer version if they wish. But updating the driver this will **break the PTP API calls/example programs**. So none of the PTR API calls will be compatible with the timing board. Besides the PTP calls/example programs no longer working with the PTP module, they can update to newer versions of the driver without any other issues.
- **If the customer ISN’T using any PTR calls/example programs:** They CAN upgrade the driver to any newer version if they wish, without any adverse effects on the operation of the board/driver. Updating the driver beyond the earlier versions of the driver JUST breaks the PTP associated API calls/example programs (the “PTR calls”).

Board handle and device index, index (Instance) Number

Example API call/example program syntax: **IR_SetMode <device index> <index> <mode>.**

The **device index** is an index into the number of TSync-PCIe boards present in the system. The first board installed is assigned the device index value of “0”. Any additional installed boards are assigned an incremental device index.

The **board handle** is different from the **device index.**   The **board handle** is a void pointer to an internal board information structure that is returned when the driver is opened by a user of a specific board.

The **index** is an “instance” number used in some calls to indicate either a potential or currently present similar input or output configuration. The TSync-PCIe board was designed as a timing engine that other products could be built upon. So many inputs and outputs can have more than one of it. The first instance of that particular “feature” is an index of 0. Then, each duplicate input/output is incremented by one, to identify another specific similar “feature”.

For example, there is only one IRIG DCLS input on a TSync-PCIe board, so its index is 0. Since it’s entirely possible for other products to have more than 1 IRIG DCLS input, the index number allows configuration for the multiple inputs of this reference. For the TSync-PCIe boards specifically, most of the index values are going to be “0” since it’s the only one of that particular circuitry. By specifying a unique index number, even though there may only currently be one instance, this allows the ability to be easily able to add-on additional “features” with little change to the driver calls (provides flexibility). The **mode** is used in some calls as a “variable” in order to select a particular configuration related to that particular API call.

Per your question, in each API/example program the first number after the name of the call is the device index. It indicates the command should be sent to the first TSync-PCIe board installed. The first board is assigned a device index value of “0”. If there happened to be two or more TSync-PCIe boards installed in the same machine, they would each be assigned an ascending device index. So if there is only one board installed, the first number will always be a “0”.

A second number follows the device index in certain API calls/example programs. This is the instance number. The TSync-PCIe board is designed to be a full timing engine that can be used in other products (such as the Spectracom SecureSync). Certain functions of the TSync-PCIe board or other products can potential have more than one of the same function. An example of this is GPS input. Though there is only one available GPS input currently available on the TSync, it’s always possible that someday, the TSync-PCIe or another product based on the TSync board, could have two GPS inputs. The instance number specifies which of the specific instances of the same function the API call or example program is addressing. The first instance of a function is always assigned an instance of “0”. In the case of GPS on the TSync-PCIe board, the instance will be a 0.

Processor time used when performing GPS calls

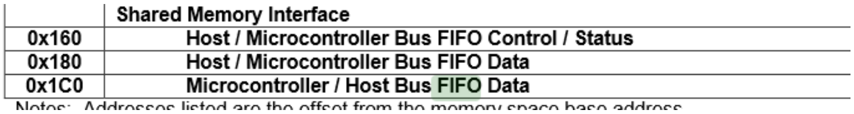
Q. I've noticed that the TSync-PCIe driver functionGR_GetPos ition (n Instance=0) uses a significant chunk of CPU time even when invoked only once per second; on a 2.33 GHz Intel Xeon, a single call occupies between 0.04 and 0.15 seconds of CPU time. Any idea why this is, and whether there's an equivalent but quicker API function for reading the position?

A. The CPU time used during this call is not unexpected operation as there are several steps involved. When the host tries to read the GPS position from the Tsync board, it sends a query to the software running on the board. That software in turn makes a query to the GPS component of the board for that information to pass back to the host.

However, the information is not always immediately available for access. Since both the software and the host are looking at the same data set, access to it needs to be controlled. When the GPS component needs access to the data set to run the communications protocol to the GPS receiver, it puts a lock on the data set. That means if the data set is locked, then the host has to wait to access that data until the component is done with it. The times reported for accesses are not out of line with the time that the component holds the data set in order to receive and process messages using the protocol.

VxWorks, Matlab Simulink or other custom drivers not available from us

- Spectracom does not supply a VxWorks, Matlab Simulink driver.
- The Linux driver for TSync board contains Source code that can be used as a reference to see how functions are performed.
- Refer to section 2 of the Application programmer’s manual for register Memory mapping.
- Refer to Section 8 of the Application programmer’s manual for info on performing different functions.
- Unlike legacy TPRO/TSAT boards, the TSync register reads need to be read through the FIFO. The registers for the FIFO buffer are defined under ‘**Shared Memory Interface**” in the App Programmers guide. With the exception of the “HW” commands (which are direct register reads) all other calls go through two FIFO buffers.



	Shared Memory Interface
0x160	Host / Microcontroller Bus FIFO Control / Status
0x180	Host / Microcontroller Bus FIFO Data
0x1C0	Microcontroller / Host Bus FIFO Data

Notes: Addresses listed are the offset from the memory space base address

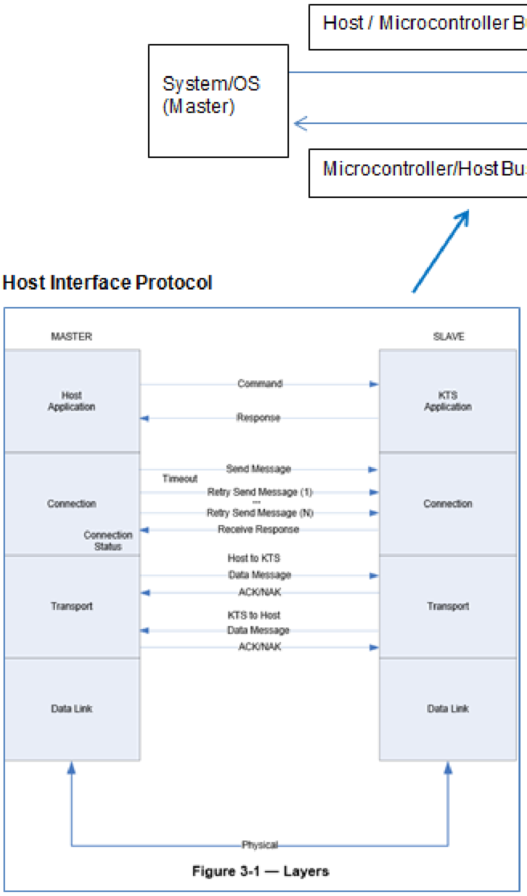
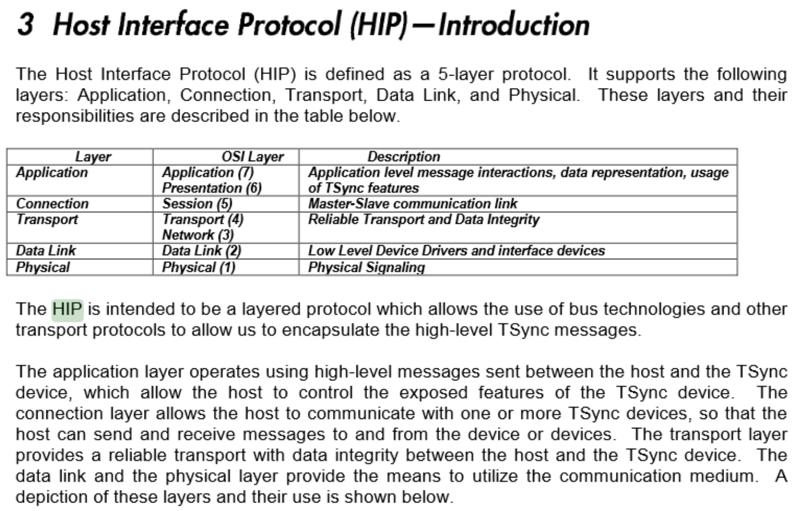


Figure 3-1 — Layers

There are three main calls in the TSync driver- Gets, Sets and Waits.

Host Interface Protocol

- Comms through the FIFO buffers use the HIP protocol (Host Interface Protocol)
- Info below is from Section 3 of the Application programmers manual
- Sections 3, 4 and 5 breaks down all of the Layers and describes this interface in great detail.



3 Host Interface Protocol (HIP)—Introduction

The Host Interface Protocol (HIP) is defined as a 5-layer protocol. It supports the following layers: Application, Connection, Transport, Data Link, and Physical. These layers and their responsibilities are described in the table below.

Layer	*OSI Layer*	*Description*
Application	*Application (7)* *Presentation (6)*	*Application level message interactions, data representation, usage of TSync features*
Connection	*Session (5)*	*Master-Slave communication link*
Transport	*Transport (4)* *Network (3)*	*Reliable Transport and Data Integrity*
Data Link	*Data Link (2)*	*Low Level Device Drivers and interface devices*
Physical	*Physical (1)*	*Physical Signaling*

The HIP is intended to be a layered protocol which allows the use of bus technologies and other transport protocols to allow us to encapsulate the high-level TSync messages.

The application layer operates using high-level messages sent between the host and the TSync device, which allow the host to control the exposed features of the TSync device. The connection layer allows the host to communicate with one or more TSync devices, so that the host can send and receive messages to and from the device or devices. The transport layer provides a reliable transport with data integrity between the host and the TSync device. The data link and the physical layer provide the means to utilize the communication medium. A depiction of these layers and their use is shown below.

HW commands (such as HW_GetTime)

- HW commands don’t go through the FIFO buffers (but still use the TSync driver)
- These are direct memory register reads
- Section 2 of the Application programmer’s manual provides register locations.

Reading/writing to registers directly “Direct Memory Access” using a Spectracom driver (Peek and Poke API calls)

- Refer to the Application Programmers Manual for more information on direct memory access (1191-5002-0050): I:\New Released\Manuals\1191-xxxx-xxxx. (**Note**: the first page of Section 2 provides the Memory Map of all of the registers).
- Values from the TSync-PCIe boards can be directly read from or written directly to the board‘s registers, if desired.
  - **“Peek” API call:** Allows registers to be read directly
  - **“Poke” API call:** Allows registers to be written to directly (Caution: Make sure to write to the correct register. Otherwise bad things can happen)

The PC BIOS assigns a unique Base address for each PCIe bus card installed. Then, each Memory Register is assigned a unit Base address extension to identify each particular register.

The TSync Application Programmers manual provides a Memory Map for the memory locations (Base address extensions) which store specific contents. The customer can then create a custom driver to read the contents directly. Refer to the Application Programmers manual for more details I:\New Released\Manuals\1191-xxxx-xxxx (1191-5002-0050).

The source code supplied with the Linux driver can assist them in identifying the Base address that is assigned by Bios and shows how we access the registers.

Pages 2-1 and 2-2 of the Application Programmers manual (latest version is attached, just in case) provides the Memory Map (Base address extension) for all of the Memory Registers on the TSync-PCIe board.

I recommend you also refer to the source code that is provided with the Spectracom TSync-PCIe Linux driver. This source code will help you determine the unique Base address that is assigned by the PC BIOS to the installed TSync-PCIe board. The source code also provides good information that can be used to help create your custom driver for accessing these memory registers.

Tsync registers reads need to be read through the FIFO buffer

Email Keith sent based on input from Tim Tetreault Please be aware that the TSync family of timing boards are setup differently than the earlier, legacy TPRO/TSAT timing boards. Time can be read from a register but all API commands that don't start with a "**HW_"** pass through a FIFO buffer to send and receive commands with the TSync boards.

All commands will use the "TSYNC_SET", "TSYNC_GET" & "TSYNC_WAIT", including even the earlier, legacy "TPRO" API commands.

The registers for the FIFO buffer are defined under '**Shared Memory Interface**" in the App Programmers guide.

Tsync .h files

Tsync_agps.h file

Q. (from Hongbo 12 Nov 14) In the new driver, the file of Tsync_agps.h will be needed, but our customer cannot find it. Could you please help advise where it can be found.

A **from Keith after talking to Tim Tetreault 12 Nov 14** I just spoke to Tim Tetreault regarding the tsync_agps.h file your customer is inquiring about.

I have a question for you. Is your customer indicating they need this file because they are getting an error when trying to compile the driver? Or did they see a reference to assisted GPS and desire to use Assisted GPS with the TSync board.

Please let your customer know that the TSync-PCIe board does not support Assisted GPS. And at this time, I'm not aware of any intentions of it being added. Any reference they may have seen to Assisted GPS in the Tsync driver is strictly due to this same driver also being shared with SecureSync, which does support GPS. So there is no tsync_agps.h file present in the driver.

They shouldn't be having any issues compiling the driver that would be related to Assisted PS. But if they are reporting they need this file because they are getting errors when they try compiling or using the TSync driver, please have your customer send us screenshots of the error messages they are seeing so we can diagnose why they are seeing errors.

Please let me know why they are saying they need this file.

tsync.dll and tpro.dll files

- When the Tsync driver is installed, it generates two .dll files, tpro.dll and tsync.dll
- The tsync.dll file support all TDync calls.
- The tpro.dll file allows customers upgrading from TSAT-PCI to be able to use the legacy commands from the TPRO/TSAT-PCIU driver, without needing to rewrite their application software.
- They still have to install the Tsync driver and recompile their software to the new driver.

Q. We are upgrading some kit for a customer who use to use a TPro PMC board. The current software uses the supplied TPRO.dll for the API.

Now they have purchased a TSync-PCIe and wish to upgrade the software, and as a consequence we downloaded the drivers and API supplied on your web site for the TSync-PCIe.

However we have noticed that the TSync-PCIe driver have both a TSYNC.dll and the TPRO.dll included. Question is, can we use the TPRO.dll to control and communicate with the TSync-PCIe card. Note we are not interested in any new features that the TSync-PCIe card may provide at this time, therefore to minimise the porting effort we would happily use the TPRO.dll if this will work?

A (Keith 19 Mar 2014) The tpro.dll file generated when the Tsync-PCIe driver is installed is actually intended for the legacy TPRO/TSAT-PCI (not PMC) timing boards. However, almost all of the calls from the TPRO/TSAT–PCI and PMC boards (and therefore this tpro.dll file) are the same.

We are in the process of releasing a new TSync-cPCI/PMC board and its driver will also contain a tpro.dll file which is for all of the legacy TPRO/TSAT-PMC/cPCI calls. We are also in the process of releasing a new version of the TSync-PCIe driver to support both Tsync-PCIe and TSync-PCI104 .

Two options for you- We should be releasing the new TSync-PCI104 boards and this new combined driver very shortly. If you like, I can let you know when it’s available. Or, you can also use the cPCI-PMC driver currently available on our website. To obtain this driver, please visit us at: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=22

***static (tsync.lib files) and dynamic (tsync.dll files) library files / .lib files (libtsync.so versus libtsync.a)

- Refer to Salesforce cases such as 117924, 114979,and 11052
- As of TSync-PCIe Linux driver version 2.3.0 (Dec 2010 time-frame), the linux driver supports both statically linked and dynamically/shared linked libraries.
  - From The linux driver v2.3.0 Release Notes: "**Added shared library support (libtsync.so).**"
- The TSync driver provides both a **static** library (**libtsync.a**) and a **shared** library (**libtsync.so**).
  - The example programs are built linked with the **static library** (**libtsync.a**)
  - To use the example programs with the **shared library**, modify the example "makefile" by replacing the libtsync.a with libtsync.so and rebuild.
    - The **libtsync.so** shared library file is generated during the driver build. It's located in the **Root/Desktop/TSync/Linux/tsync/Lib** directory

Prior to driver version 2.3.0 (before libtsync.so file was available)

Q. I am now trying to link to your driver on my 64-bit linux system since previously I actually had the driver code compiled into my library and that was leading to the undefined symbols. Unfortunately when I link I get the error message near the bottom of the following:

A. Currently our libraries are not built as shared object libraries. They are intended to be statically linked. The lowest level Makefile for our driver library with its CFLAGS is located here: \tsync\linux\tsync\lib\obj\Makefile.

(Note that this was resolved in the version 2.3.0 driver update, Dec 2010)

General info on library files

A) Static library/statically-linked library

(from wikipedia)

static library or **statically-linked library** is a set of routines, external functions and variables which are resolved in a caller at compile-time and copied into a target application by a compiler, linker, or binder, producing an object file and a stand-alone executable.[1] This executable and the process of compiling it are both known as a static build of the program. Historically, libraries could only be *static*. Static libraries are either merged with other static libraries and object files during building/linking to form a single executable or loaded at run-time into the address space of their corresponding executable at a static memory offset determined at compile-time/link-time.

B) Dynamic library

When an app is linked with a library using a static linker, the code that the app uses is copied to the generated executable file. A *static linker* collects compiled source code, known as object code, and library code into one executable file that is loaded into memory in its entirety at runtime. The kind of library that becomes part of an app's executable file is known as a static library. *Static libraries* are collections or archives of object files.

Advantages and disadvantage of static libraries (https://en.wikipedia.org/wiki/Static_library)

There are several advantages to statically linking libraries with an executable instead of dynamically linking them. The most significant is that the application can be certain that all its libraries are present and that they are the correct version. This avoids dependency problems, known colloquially as DLL Hell or more generally dependency hell. Static linking can also allow the application to be contained in a single executable file, simplifying distribution and installation. With static linking, it is enough to include those parts of the library that are directly and indirectly referenced by the target executable (or target library). With dynamic libraries, the entire library is loaded, as it is not known in advance which functions will be invoked by applications. Whether this advantage is significant in practice depends on the structure of the library.

In static linking, the size of the executable becomes greater than in dynamic linking, as the library code is stored *within the executable* rather than in separate files. But if library files are counted as part of the application then the total size will be similar, or even smaller if the compiler eliminates the unused symbols. On Microsoft Windows it is common to include the library files an application needs with the application.[2] On Unix-like systems this is less common as package management systems can be used to ensure the correct library files are available. This allows the library files to be shared between many applications leading to space savings. It also allows the library to be updated to fix bugs and security flaws without updating the applications that use the library. In practice, many executables (especially those targeting Microsoft Windows) use both static and dynamic libraries.

.lib files (Linker files)

What is a .lib file?
From: http://file.org/extension/lib

Files that contain the .lib file extension usually hold a static data library of information. The information in this file is associated with a specific computer application. The LIB files usually store the functions that are referenced by the application, which is then associated with the library.

LIB files may also contain objects like images, media and text clippings. LIB files are an alternative to DLL files, which contain dynamic link libraries. The LIB files contain static libraries that can be used as import libraries for DLL files, with the imported library files being marked with the .lib file suffix.

From Wikipedia: http://stackoverflow.com/questions/3250467/what-is-inside-lib-file-of-static-library-statically-linked-dynamic-library-an

Static library
In a static library, the .lib file contains all the actual object code and data for the functions provided by the library. In the shared version (what you referred to as statically linked dynamic library), there is just enough code to establish the dynamic linkage at runtime. The linker then identifies the bits it needs and puts them in the final executable.
If the code of the library is accessed during the build of the invoking program, then the library is called a static library. An alternative is to build the executable of the invoking program and distribute that, independently from the library implementation. The library behavior is connected after the executable has been invoked to be executed, either as part of the process of starting the execution, or in the middle of execution. In this case the library is called a dynamic library. A dynamic library can be loaded and linked as part of preparing a program for execution, by the linker. Alternatively, in the middle of execution, an application may explicitly request that a module be loaded.

Dynamic library
For a dynamic library, *the .lib file contains a list of the exported functions and data elements from the library, and information about which DLL they came from. When the linker builds the final executable then if any of the functions or data elements from the library are used then the linker adds a reference to the DLL (causing it to be automatically loaded by Windows), and adds entries to the executable's import table so that a call to the function is redirected into that DLL.*
You don't need a .lib file to use a dynamic library, but without one you cannot treat functions from the DLL as normal functions in your code. Instead you must manually call LoadLibrary to load the DLL (and FreeLibrary when you're done), and GetProcAddress to obtain the address of the function or data item in the DLL. You must then cast the returned address to an appropriate pointer-to-function in order to use it.

From Salesforce case 11052 (:

Q. I received the TSync card this week. One thing I notice is that there is no static library version of TSync.lib; all of our apps are statically linked, and I'm hoping you guys support a build of your API in this mode as well. All I see in the distribution is a dynamically linked version.

A. As of at least July 2013, the TSync-PCIe Linux driver does support both static and dyamic lib files. However, as of at least version 2.4.1 Windows driver does not support both static and dynamic .lib files.

Compiling Customer's application software

(Note: For Visual Studio info, refer to the Windows driver section further below)

Borland

- Our TSync.lib file was written for Visual C+ +. Customer desires to use Borland C + + instead:

Q. The customer has begun to write his software application. However, he is using Borland C++ and Tsync.lib seems to be wrote for Visual C++. We have found a solution however we are not sure his approach. We used Tsync.dll and applied "implib" utility in order to have Tsync.lib compatible with Borland. It seems to work but we do not have the board to verify it. Is it the good approach? If it is not, can you give us the procedure? Or do you have a Tsync.lib compatible for Borland C++?

A. Email from Tim T - The approach they are using is correct. Here is a link with more information on how to use Visual C++ DLLs with Boland C++ using the IMPLIB utility. http://bcbjournal.org/articles/vol4/0012/Using_Visual_C_DLLs_with_CBuilder.htm?PHPSESSID=046f3a5a4298523cb724d9dcf777ec0f

C# (called "C Sharp", "Managed Code C #" or "Managed Code C Sharp")

Our TSync.lib file was written for Visual C+ +. Customer desires to use C# (C #) instead:

(email from Jack Loui 3/17/11) I am trying to convert TSync C++ code to C# by using [DllImport("Tsync.dll"), CallingConvention = CallingConvertion.Cdecl)]. So far I have trouble to get this to work. Do you have any sample code wh A newer version of Microsoft Visual C++ 2010 ch is in C# to work with the TSync card?

Keith's reply: Unfortunately, we don't have any sample C# code that we can provide you with. We know that we have had one or two other customers that were able to use our dll files for the conversion, but we don't have any information on how they were able to make the conversion, nor have we done this ourselves.

Installing TSync-PCIe drivers on an ARM platform (ARM processor)

Q. Is it difficult to port the driver to ARM platform?

A. Keith's response- At this time, we are not aware of any compatibility issues with the TSync-PCIe Linux driver and an ARM platform. However, please keep in mind that a TSync-PCIe board does not need to be installed in order to test the driver for compiling. The TSync-PCIe drivers can be downloaded from the Spectracom website at no cost. After downloading the driver, your customer can then try installing and compiling the driver to see if there are any compatibility issues, even before they receive and install a TSync-PCIe board.

Email from Dave Sohn (11/7/11)

I haven't heard of anyone using the TSync with an ARM processor. I don't know of any limitations with that. There may be some endian issues, but I don't know for sure. I'm assuming this is a Linux platform. They can download the drivers and try it. If there are endian issues, there is a flag in the driver for setting endian that could be tried.

**TROUBLESHOOTING TSync boards

****TSync-PCIe board doesn't appear to be operational after install

Questions to Ask/Info to help troubleshoot (steps to troubleshoot further below)

- What is the Serial Number of the board (via the silver P/N sticker affixed to the board or with the **LS_GetSerialNo** API call/example program).
- Is this the only TSync-PCIe board that you have? If not, have you tried another timing board in its place, yet?
- Have you tried reseating the board in the slot?
- Have you tried this TSync-PCIe board in another computer, yet? If not, I recommend you try it in a different computer, to determine if there is an issue with the PC's PCIe slot.
- What are the LEDs on the edge of the board doing (Do they flash once at power-up? Are they flashing a pattern? Are they constantly on)?
- What about the FPGA LED… is that lit? Is the bootloader jumper in the correct position?
- TSync Board firmware version and driver versions.
- Is the board installed in Windows PC or Linux system?

Linux systems only

A. What is the specific linux distribution (such as Redhat, CentOS, Scientific linux, etc) and kernel version?

Notes:

- ASPM may need to be disabled
- Updating the linux distribution to v6.4 or higher may help address issues with ASPM issues.
- Class code may need to be updated (especially if the board shipped before ~ April 2013).
- System BIOS may need to be updated

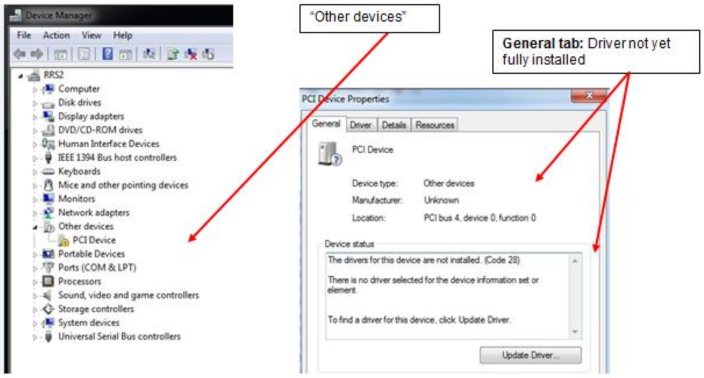
B. Has the earlier version 2.3.0/2.4.1 drivers ever been installed on this machine. If so, the Rules files need to be updated.

Windows PCs only

With the board installed in a Windows PC (whether or not the driver has been installed yet) "TSync-PCI" is it seen in "Device Manager" (**Start**-> **All Programs** -> **System** -> **Hardware**-> **Device Manager**). Device Manager should list "**Timing Boards**" (with "**TSync-PCI Timing board**" below it).

Note: Before the Windows driver is fully installed (with the board installed), it may show up as simply "PCI device" (under "other devices" as shown below:

- With the board installed in a Windows PC (whether or not the driver has been installed yet) "TSync-PCI" should be seen in "Device Manager" (**Start**-> **All Programs** -> **System** -> **Hardware**-> **Device Manager**). Device Manager should list "Timing Boards" (with "**TSync-PCI Timing board**" below it).



"Other devices"

General tab: Driver not yet fully installed

PCI Device

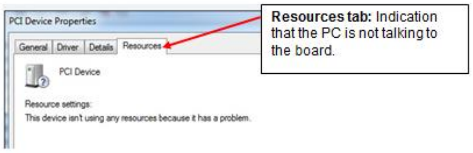
Device type: Other devices
Manufacturer: Unknown
Location: PCI bus 4, device 0, function 0

Device status

The drivers for this device are not installed. (Code 28)

There is no driver selected for the device information set or element.

To find a driver for this device, click Update Driver.



Resources tab: Indication that the PC is not talking to the board.

PCI Device

Resource settings:
This device isn't using any resources because it has a problem.

- If displayed, click on **Properties** -> **Details**-> **Hardware IDs** and see what is listed (work with Tim Tetreault as this list may indicate the class code assigned to the board is causing issues with the particular PC).
- Try the board in another PC (preferably on a different Model motherboard) to see if the other PC can recognize it's installed.
- Refer to **Class Code update fix** and **(ASPM) Power Saver/Power Monitor feature**
- Verify the PC BIOS settings for the PCie bus. Make sure the bus is not configured for "Video Only", ASPM (autopower save mode, USB) enabled, etc.

Email KW sent to a customer.
Have you tried going into "Device Manager" and running "Scan for hardware changes" from the "Action" menu?

Another idea is to verify that the system can see the board. In "Device Manager", select "Devices by connection" in the "View menu". Look under the "Microsoft ACPI-Compliant System->PCI bus" group for any unknown device. If you look at properties for that unknown device you can check to see if one of them is the board. The TSync board is Vendor ID "1AD7" and Device ID "8000". Do you have any other cards installed? Are they visible in Device Manager? Do you have any other PCIe boards to verify that the system is properly seeing the PCIe card slots?

Specific motherboard issues

ASUS motherboard

Salesforce Case 6738- The ASUS motherboard came with a utility called 'ASRock X-Fast USB' that was removed and the computer was re-booted. Immediately, the installation program operated normally and the Tsync driver was installed successfully.

Checks to make:

- Check the Status LEDs on the edge and the green “FPGA load” LEDs on the back side of the TSync-PCIe board:
- Verify all three Status LEDs (red, yellow and green) on the edge of the TSync board turn on and then go out (and remain not lit until an input reference becomes valid).

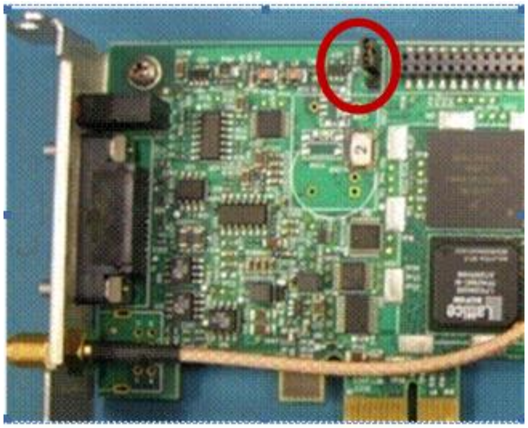
Right after the PC with the TSync-PCIe board powers-up, the three LEDs on the edge of the TSync-PCIe (the red, yellow and green LEDs), should momentarily illuminate and then turn off (until an input reference is connected). Do all of these LEDs momentarily turn on when the PC is first booted-up?

If the LEDs are not lit at power-up /all three constantly lit/exhibit the correct pattern:

- First, try to re-seat the TSync-PCIe board in the system to make sure that all the contacts are solid.
- **If the green and red LEDs are blinking on your PCIe card,** it is likely that the board is in the bootloader waiting for image downloads from the host. This could signify that either the board needs to be reprogrammed or that there was a failure in the flash forcing it to fail to load the run-time or backup default images, landing back into the bootloader.

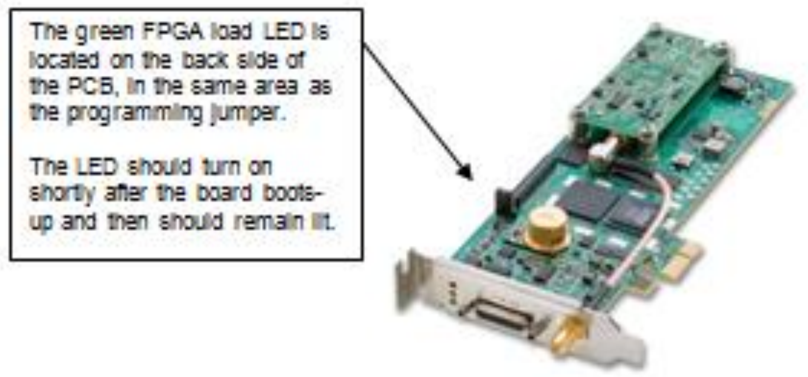
Verify the red “FPGA” loaded LED is lit (verify Jumper J25 is in the correct position)

Check jumper setting on J25 and make sure that it is configured on the pins to the outside edge of the board, as seen in the red circle in the photo below:



If this jumper is in the wrong position, on a reboot the timing board will never look for the new images and will always load the bootloader (Green and Red LED s will be blinking and the Yellow LED will be not lit). Please place this two position jumper as shown in the picture above (with the jumper across the two outer pins (closest to the edge of the TSync-PCIe board.

There is also a green ”load” LED on the back-side of the TSync-PCIe boards (in the same area of the board as the jumper described above, but on the other side of the timing board. This LED should light shortly after the TSync-PCIe board powers-up and then should remain lit, from that point forward. Please let me know if this LED turns on and then remains lit green while the PC is on.



The green FPGA load LED is located on the back side of the PCB, in the same area as the programming jumper.

The LED should turn on shortly after the board boots-up and then should remain lit.

If the green LED still doesn't turn on and remain lit with the jumper in the correct position

- Verify +3.3 vdc is present from the PCIe bus. If this DC voltage is low, the board may be stuck in constant reset state. Refer to: Power (+3.3vdc +12vdc and -12vdc from PCI bus)/logic.
- Try the board in another slot, or another system altogether and see if the green LED remains lit.

TSYNC_open() and TSYNC_close() API calls

- For multi-threading, many "get" and "set" calls require the board be opened before perforrming the call (and then closed afer performing the call). These calls are those that require "software interface", such as "CS_GetTime" calls for example.
- Other calls which are to the hardware only (such as "HW_Gettime" for example) do not require the board to be opened before performing the call.

Q Not sure what functionality of the TSYNC_open() / TSYNC_close() APIs provide? Do they act like semaphore/mutex type locks

A **(Reply from Jodi 20 Dec 17)** TSYNC_open() and TSYNC_close() are used for synchronization. The open function is used to grab a handler to issue the GET and SET commands. Then the close is used to close the connection. This is similar to reading/writing to a file in Linux.

TSYNC_COMM_ERR

- This message indicates various conditions:
- Typically indicates the installed driver isn't able to communicate with the installed board.
- With waitFor call, indicates the interrupt hasn't occurred within a set time-frame (refer to WaitFor call for more info)
- Per conversation with Dave Sohn (29 May 2019) this message can also occur
  1. If the TSync board is being overwhelmed/bogged-down by too many event inputs for timestamping (temporarily disconnect the event input and see if this message stops occurring).
  2. If the TSync-PCIe board is not being opened/closed for each thread (when using multi-threaded software) it needs to be opened for each of calls (not all calls, such as HW_Gettime for example, do not require the board to be opened). only those calls that require "software interface", such as "CS_GetTime" calls for example, need the board to be opened before the call is performed.

TSync compatibility with Matlab programming language

- Refer to Salesforce cases such as 181156, 115012 and 21696
- MATLAB is a proprietary programming language.

Per Wikipedia:
MATLAB (**mat**rix **lab**oratory) is a multi-paradigm numerical computing environment and fourth-generation programming language. A proprietary programming language developed by MathWorks, MATLAB allows matrix manipulations, plotting of functions and data, implementation of algorithms, creation of user interfaces, and interfacing with programs written in other languages, including C, C++, Java, Fortran and Python.

Q We have a TSYNC PCIe and GPS antenna that we would like to use to give time-tagging data to a MATLAB program, which is also collecting data from other timing equipment disciplined by the same GPS source. We have been able to successfully install drivers, etc., but I do not have a great deal of programming experience communicating to PCIe hardware. If you have any relevant examples or script files that you could send us, that would be wonderful.

I would like to be able to have a MATLAB script query the TSYNC PCIe for the absolute or relative time and then receive the answer. My intuition would be to use MATLAB to call the DLL, but I am not sure if that is the most efficient way and I am not sure if I should call an application file or some other type of file.

A Response from Ryan Johnson (24 Jul 17) I'm not sure if we have anything already for this, but likely the easiest way to do this would be to create a small C program to interface with the TSync driver and do the time queries, and then interface that C program to Matlab via a *mex* function: https://www.mathworks.com/help/matlab/call-mex-files-1.html?requestedDomain=www.mathworks.com

A (email from Keith, 8 Aug 17) Thanks for your earlier inquiry regarding the Spectracom TSync-PCIe bus-level timing boards, with regards to the MATLAB program.

I sincerely apologize for the very long delay in getting back to you. I initially needed some input from Engineering for your question. And once I heard back from them, I kept getting pulled-off on other calls ☹!! There are just never enough hours in the day, days in the week, weeks in the year… ☹!!

For your reference, I had assigned our case Number 115012 to your inquiry.

This was the first time recall ever receiving a question about using the TSync boards in conjunction with this program (I knew we didn't have anything specifically available for interfacing with this program. But I didn't know if there was another viable solution). So I ran your question by one of our engineers. His response also indicated we don't have anything available specific to this program. He suspected this program is similar to LabView (requiring a wrapper, which we don't have available for the TSync boards, either).

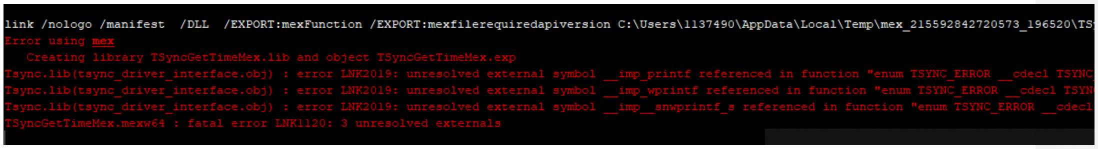
His suggestion to pass along to you was as follows: "but likely the easiest way to do this would be to create a small C program to interface with the TSync driver and do the time queries, and then interface that C program to Matlab via a mex function: https://www.mathworks.com/help/matlab/call-mex-files-1.html?requestedDomain=www.mathworks.com"

Again, my apolgies for the very long delay in getting back to you, but I hope you find this suggustion helpful!

Issue reported: "Error using mex"

- refer to Salesforce Case 181156

Q …I had a couple questions in interfacing the timing card with MATLAB via a Mex function. Currently I have a mex function that essentially is the HW_GetTime.c file, however I get the following error:



```
link /nologo /manifest  /DLL  /EXPORT:mexFunction /EXPORT:mexfilerequiredapiversion C:\Users\1137490\AppData\Local\Temp\mex_215592842720573_196520\TS
Error using mex
   Creating library TSyncGetTimeMex.lib and object TSyncGetTimeMex.exp
Tsync.lib(tsync_driver_interface.obj) : error LNK2019: unresolved external symbol __imp_printf referenced in function "enum TSYNC_ERROR __cdecl TSYNC
Tsync.lib(tsync_driver_interface.obj) : error LNK2019: unresolved external symbol __imp_wprintf referenced in function "enum TSYNC_ERROR __cdecl TSYN
Tsync.lib(tsync_driver_interface.obj) : error LNK2019: unresolved external symbol __imp__snwprintf_s referenced in function "enum TSYNC_ERROR __cdecl
TSyncGetTimeMex.mexw64 : fatal error LNK1120: 3 unresolved externals
```

I understand your team is not as familiar with MATLAB integration, but do you have any insight on why this error persists? I have successfully linked the libraries in MATLAB, but there just seems to be an error compiling the TSync_open function in the Tsync.lib file I cannot solve.

TSync Windows Driver/ Visual Studio

- **Shortcut to drivers CD (1219-5003-6001) in Arena:** https://app.bom.com/files/detail-summary?file_master_id=1234039656&file_id=1745191706
- **Shortcut to Windows driver version upgrades spreadsheet in custservice**: I:\Customer Service\PSB, PSP software updates\TSYNC boards\TSync driver updates (PCIe, cPCI, PCI104)\Tsync-PCIe drivers\WINDOWS river
- **To find all Windows driver versions in SAP**: search for "**1219-SD02-***"

Paths on PC to TSync driver and example programs (C:\)

Path on the Windows PC to the drivers/example programs:

Driver: C:\Program Files\Spectracom\TSYNC PCI\drivers\

Example programs

- **For TSync example programs:** C: \Program Files\Spectracom\TSYNC PCI\Examples\TSYNC API\
- **For Legacy TPRO example programs:** C: \Program Files\Spectracom\TSYNC PCI\Examples\TPro API

TSync.lib file

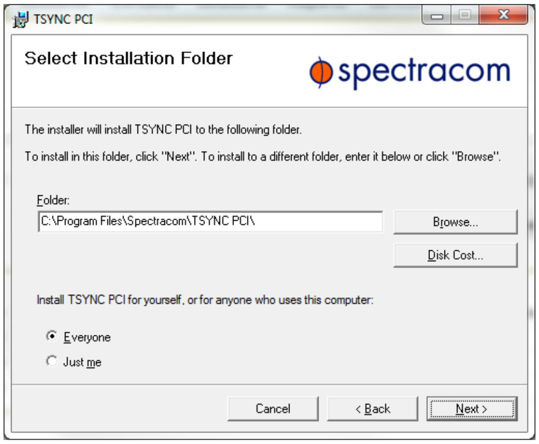
- **(July 2013) refer to Salesforce Case 11052** As of at least Version 2.4.1 of the Windows driver, this driver only support dynamically linked .lib file. It does not currently support statically linked .lib file (unlike the linux driver, which supports both).
- Per Tim Tetreault, we will need to compile the Windows driver differently when we build it, in order to support this capability.

Windows Driver versions/obtaining the latest Windows driver from our site

- Link to latest version Windows Driver (32 bit and 64 bit variants) and Release Notes, on our website: https://www.orolia.com/support/timing/tsync/pcie
- Link to excel spreadsheet with info on the various version of the Windows Driver: I:\Customer Service\PSB, PSP software updates\TSYNC boards\TSync driver updates (PCIe, cPCI, PCI104)\Tsync-PCIe drivers\WINDOWS driver

Windows driver installation

Q. I don't remember being given the option of choosing the install directory. Is that something that I can do?
A. Reply from Keith (14 Jul 2014) When installing the Windows driver, it does allow you to select where it will be installed (as shown below):



The installer will install TSYNC PCI to the following folder.

To install in this folder, click "Next". To install to a different folder, enter it below or click "Browse".

Folder:
C:\Program Files\Spectracom\TSYNC PCI\

Install TSYNC PCI for yourself, or for anyone who uses this computer:

● Everyone
○ Just me

TROUBLESHOOTING WINDOWS DRIVER INSTALL

- ➢ General-type questions to ask.

Here are some questions for you:

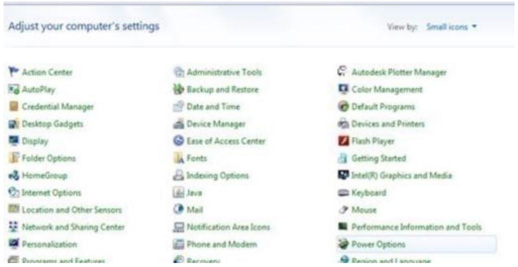
1. What is the Serial Number of the TSync-PCIe board (it's indicated on the silver Spectracom Part Number sticker affixed to the TSync-PCIe board)?
2. What is the version of the TSync-PCIe Windows driver you have?
3. Has this TSync-PCIe board been installed and operating normally in this same machine for a while, but has since stopped working with these errors just now being seen?
4. Or, Is this a TSync-PCIe board that was installed previously in another machine and then recently moved over to this IBM Windows server?
5. Or, is this the first time being installed in a machine since it was purchased?
6. At what stage of the installation process (assuming it's a new install in this machine), did the server errors start occurring? Was it when the board was first installed, while installing the driver, while working with it in conjunction with custom application software, etc?
7. The TSync-PCIe board can be installed without the driver being installed, and should be recognized in Control Panel -> Device Manager. Is this board being detected as being installed in Device Manager.
8. If you haven't already, are you able to temporarily install this TSync-PCIe board in another Linux or Windows machine to see if it can be detected and able to communicate via the bus?
9. Do you have any other PCIe products installed in the same system?
10. Send us the install log (C:\Users\kwing\AppData\Local\Temp\VSD224B.tmp -> install.log)

Potential issues with Windows power management (“ASPM for Windows)

- Could potentially cause Windows “Blue Screen” to be periodically occur
- Refer to Salesforce case 14925.

Email Keith sent to Kim (3 Aug 2014) Before returning the timing boards to you, I just found out there are some settings in Windows Control panel that could potentially cause issues with a Windows system. These settings are associated with Windows power management/sleep configuration.

From a Windows PC that was periodically exhibiting the blue screen, please send us screenshots of settings in **Control Panel** -> **Power Options**



After clicking on Power Options, send us a screenshot showing which “power plan” is currently selected.
On the left side of the page, click “Choose when the computer sleeps” and send a screenshot showing the settings for “put the computer to sleep”

Next click on “Change advanced power settings” (below “put the computer to sleep”).   In the list of items, expand “Hard disk” and “PCI Express” -> “Link State Power Management”.

Once we receive the screenshots from you, we will review them and make any necessary recommendations so that you retest once you receive the boards back. I will also have the repair Dept. start the process of sending the boards back to you. They already have the latest firmware version installed, so we did not update the firmware in them.

**Windows Driver compatibility/install issues/conflicts with other PCIe boards

A) Foreign language (non English, such as Korean or French) settings in Windows can prevent Windows driver installation

- ➢ Has been observed with computers in France and South Korea
- ➢ Refer to JIRA ticket ROC-28 (info directly below and additonal input from Chris Olin is in this ticket)
- Per Chris Olin (6 Aug 2020) either change the Windows foreign language settings to "EN-US" or manually install the portions of the driver.

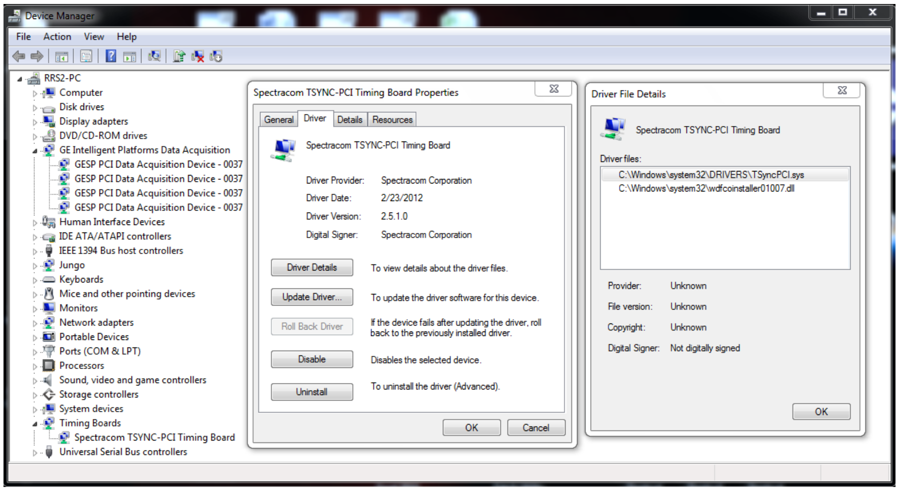
B) Apparent possible conflicts with other installed PCIe boards in the same system.

- ➢ Driver issues could cause conflicts. If boards are resetting, could also be an issue with the +3.3vdc from the bus dropping too low.

To troubleshoot possible conflicts between TSync and other installed boards, go into **Hardware** tab / **Device Manager**. For all PCIe installed products (TSync-PCIe and others).

1) Right click on each product and select "**Properties**".
2) Select the **Driver** tab.
3) Select "**Driver Details**".

This should list which driver(s) each installed board is linked to. We want to make sure the TSync-PCIe board is not linked to any other drivers and the other products are not linked to the TSync-PCIe driver. Please send a screenshot of this list to us.



RRS2-PC
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- GE Intelligent Platforms Data Acquisition
  - GESP PCI Data Acquisition Device - 0037
  - GESP PCI Data Acquisition Device - 0037
  - GESP PCI Data Acquisition Device - 0037
  - GESP PCI Data Acquisition Device - 0037
- Human Interface Devices
- IDE ATA/ATAPI controllers
- IEEE 1394 Bus host controllers
- Jungo
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Portable Devices
- Ports (COM & LPT)
- Processors
- Sound, video and game controllers
- Storage controllers
- System devices
- Timing Boards
  - Spectracom TSYNC-PCI Timing Board
- Universal Serial Bus controllers

Spectracom TSYNC-PCI Timing Board

Driver Provider: Spectracom Corporation
Driver Date: 2/23/2012
Driver Version: 2.5.1.0
Digital Signer: Spectracom Corporation

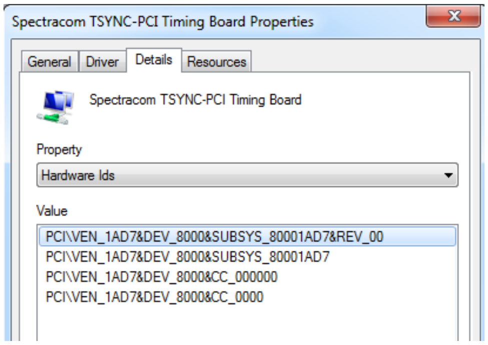
Driver Details — To view details about the driver files.
Update Driver... — To update the driver software for this device.
Roll Back Driver — If the device fails after updating the driver, roll back to the previously installed driver.
Disable — Disables the selected device.
Uninstall — To uninstall the driver (Advanced).

Spectracom TSYNC-PCI Timing Board

Driver files:
C:\Windows\system32\DRIVERS\TSyncPCI.sys
C:\Windows\system32\wdfcoinstaller01007.dll

Provider: Unknown
File version: Unknown
Copyright: Unknown
Digital Signer: Not digitally signed

4) Now click on the **Details** tab.
5) In the drop-down in the Details tab, choose "**Hardware Ids**").
6) This tab shows the sub-system ID information. Send us a screenshot of this list for all of the products.



Spectracom TSYNC-PCI Timing Board

Property: Hardware Ids

Value:

```
PCI\VEN_1AD7&DEV_8000&SUBSYS_80001AD7&REV_00
PCI\VEN_1AD7&DEV_8000&SUBSYS_80001AD7
PCI\VEN_1AD7&DEV_8000&CC_000000
PCI\VEN_1AD7&DEV_8000&CC_0000
```

If the Drivers for the TSync-PCIe and other devices don't appear to be conflicting (as determined by the info above) and the TSync-PCIe board is unexpectedly rebooting, using a scope (not a mulitmeter) check the +3.vdc going into the TSync-PCIe board via its Test point to see if this voltage is dropping below the min, even momentarily.

Note: This may be too quick or too small of a drop to catch with a multimeter. Really need to look at this test point with a scope so that it can be triggered if it drops below about 3.16vdc (resulting in a reset).

C) Blue screen of death/"WHEA uncorrectable error" message while trying to install driver on Win 2012 Server

- Refer to SF case 117434
- Customer was installing Windows v.3.2.1 driver on a

Email from Dave L to Tim T (31 Jul 17) Our customer at Peterson AFB is having trouble installing a TSync-PCIe in his Windows 2012 R2 PC.

He gets Blue Screen of Death when loading TSYNC Win x64 driver on Win 2012 R2 Server.v 3.2.1 Driver. on a Dell Powerage R730 PC. Gets a" WHEA uncorrectable error" The dmp file downloaded is attached. I can't see any useful information in the dmp file from my untrained eye. I have not encountered this type of error before. Any advice on how to resolve the issue will be greatly appreciated.

Reply from Dave L to Tim.. (*Tim's original questions below in black, and customer's responses below in red)*
Here are the customers answers. I think they have old TSync board firmware so I am going to instruct them to update the board.

Was this a new driver install or an upgrade? New driver install

Is this a new PCIe board with the latest firmware? The board is one that we had in stock. The board does not have the latest firmware. If you could help with installing the latest firmware, it would be helpful.

Does the computer run if they remove our driver but leave the TSync board installed? When booting the computer this morning I get the following:

UEFI0067: A PCIe link training failure is observed in PCIe Slot 1 and the link is disabled. Do one of the following: 1) Turn off the input power to the system and turn on again. 2) Update the PCIe device firmware.

Does the computer run if the driver is installed but the board is removed? Yes

Do they have any other boards/hardware installed in the computer? There are no other boards/hardware installed in the computer.

A Reply from Ryan (31 Jul 17) I found this online which seems to indicate that the PowerEdge 730 supports Gen 3 PCIe cards only (TSync is Gen 1)
https://serverfault.com/questions/310041/dell-poweredge-pcie-training-error-what-to-do#446061

In the 730 manual it did state that “*The PowerEdge R730 system supports PCI express (PCIe) generation 3 expansion cards*”
http://topics-cdn.dell.com/pdf/poweredge-r730_owner's%20manual_en-us.pdf

Not sure if there’s a setting in the bios to allow working with legacy PCIe cards… I didn’t see anything.

D) Windows Digital Signature requirements/support for Windows Server 2016/Windows 10

Windows cannot verify the digital signature for the drivers required for this device. A recent hardware or software change might have installed a file that is signed incorrectly or damaged, or that might be malicious software from an unknown source. (Code 52)

- Digital Signatures need to be verified in Windows 64 bit OS.
- Digital Signature verification was an issue with earlier versions of the Windows drivers.
- Per Ron Dries (17 Feb 2021) The “MSS” in the file name (such as “TSync_Windows_3.2.3_**MSS**_x64” indicates that it is Microsoft Signed, and is required for newer versions of Windows 10 and Server 2016 and 2019.
- Per Ron Dries (17 Feb 2021) Are they using the most recent Windows driver from our website? TSync Windows 64-bit Driver | Spectracom Fileshare <http://files.spectracom.com/public-downloads/tsync-windows-64-bit-driver>

- Tim T and Keith are not sure exactly when this was resolved. But at least Windows driver version 3.1.2 (if not some other previous versions) shouldn’t have any issues with Digital Signatures
- If a customer reports a Digital Signature issue, find out what version of the Windows driver is being installed when this issue occurs. Likely trying to install an earlier version.

1. Issue with Digital signature requirements (Windows driver version 3.2.3 - Windows Server 2019 and Windows Server 2016)

***Windows cannot verify the digital signature for the drivers required for this device**. A recent hardware or software change might have installed a file that is signed incorrectly or damaged, or that might be malicious software from an unknown source. (Code 52)*

- Refer to Salesforce Cases such as 258339 (excerpt further below in blue)
  - Error message observed with v3.2.3 Windows driver
  - Error message observed on Windows Server 2019 and Windows Server 2016

When installing TSYNC Timing Card on Windows Server 2019 version 1809 (OS Build 17763.1757) Virtual machine I get an error

Windows found drivers for your device but encountered an error while attempting to install

Spectracom TSYNC Timing Board

***Windows cannot verify the digital signature for the drivers required for this device**. A recent hardware or software change might have installed a file that is signed incorrectly or damaged, or that might be malicious software from an unknown source. (Code 52)*

I also get the error on Windows Server 2016 version 1607 (OS Build 14393.4104)

Reply from Ron Dries (17 Feb 2021) Are they using the most recent Windows driver from our website? TSync Windows 64-bit Driver | Spectracom Fileshare<http://files.spectracom.com/public-downloads/tsync-windows-64-bit-driver>

The MSS in the file name indicates that it is Microsoft Signed, and is required for newer versions of Windows 10 and Server 2016 and 2019.

I am guessing that the driver that Windows is providing them is not the Microsoft signed driver.

2. **Issue with Digital signature requirements (Windows driver version 3.2.1 - Server 2016 and newer iterations of Windows 10- versions 1607 and above)**
   - Due to Microsoft requiring digital signed drivers starting in Windows 10 (apparentlty versions 1607 and above) Windows TSync driver versions 3.2.1 and below (the latest driver as of at least Nov, 2018) the TSync driver should install just fine, but cannot be loaded (so it can't work with the board)

   From: *https://blogs.msdn.microsoft.com/windows_hardware_certification/2016/07/26/driver-signing-changes-in-windows-10-version-1607/*

   "Starting with new installations of **Windows 10, version 1607**, the previously defined driver signing rules will be enforced by the Operating System, **and Windows 10, version 1607 will not load any new kernel mode drivers which are not signed by the Dev Portal. OS** signing enforcement is only for new OS installations; systems upgraded from an earlier OS to Windows 10, version 1607 will not be affected by this change."

Email from Dave L (15 Jan 18) There is a problem with Windows 2016 and possibly even the latest version of Windows 10. Microsoft has changed the digital signature process in the new software and TSync is not currently compatible.

We are aware of this and have started the process to have the TSync approved by Microsoft but it is yet to be completed.
Customer requests will drive the priority of this project. So far we do not have much demand.
Do you know if this is a new sales opportunity and how large it could be?

I have copied Dave Sohn and Ryan Johnson as they have control of the resources.

3. **Windows Digital Signature requirements/support for Window 10 (prior to version 1607). Windows 8.1 and Windows 7**
   - Support for more recent versions of Windows was added in Windows driver **version 3.2.1 and above (Oct, 2016)**
   - Refer to TSync Windows Driver Release Notes
     - ***On our website*** https://spectracom.com/sites/default/files/document-files/TSync%20Release%20Notes%20for%203.2.1%20Windows%20Driver.pdf
     - In: I:\Customer Service\PSB, PSP software updates\TSYNC boards\TSync driver updates (PCIe, cPCI, PCI104)\Tsync-PCIe\Previous revs

Q. From Pierre and his customer (15 Jan 2018) I think that the driver and the TSync card should work with the **Windows server 2016** OS, but they must use the latest driver version and rebuilt it to work with a Windows 64-bit OS. However, as I am not sure of that, could you confirm me that should work or not, please?

A Reply from Ron Dries (15 Jan 18) It's in the TSync Windows driver release notes. (refer to excerpt below) https://spectracom.com/sites/default/files/document-files/TSync%20Release%20Notes%20for%203.2.1%20Windows%20Driver.pdf

Windows driver version 3.2.1

- New Windows OS versions and editions are now supported: Windows 7 (x86/x64), 8.1 (x86/x64), 10 (x86/x64), Windows Server 2008 (x64), 2012 (x64), 2016 (x64)

Issue with Digital signature requirements (Server 2016 and Windows 10, versions 1607 and above)

- see the previous section above

E) TSync-PCIe Driver support for earlier versions of Windows (Win 95, 98, CE, etc)

Info based on talk with Tim Tetreault:

The driver development tools we use start with Windows 2000 and go up from there. They do not list support for any earlier versions of Windows, including CE. Based on this information alone, the drivers are not likely compatible with CE. But, the only way to know for sure would be to install both the freely available driver in addition to the timing board (a demo board, if you have one available for them to try first, before buying??).
The driver contains individual folders for the particular version of Windows. When they install the driver, it may ask them which OS it is, since it may not be able to detect the version. They could select either XP or 2000 as alternate selections.

If they want to first try just installing the driver, they could try this, as the driver is available for free download (however, ultimately they would need to install both the driver and a timing board together to know it's going to work fine in CE).

Base Address Register (BAR) fix: Windows driver installs fine, but Control Utility has problem interfacing with the TSync-PCIe board

- Refer to Salesforce Case 19882

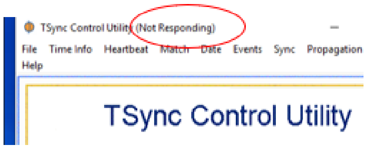
  Report from customer: "When I took away the driver signature verification, it installs but the utility control does not do anything and if I try to do anything it just freezes up and I have to restart the computer to get it to come back up".

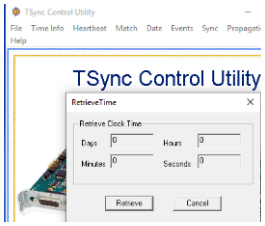
- Initially reported by Invenys/Schneider Electric (and also observed here in our lab with their system sent to us- their TSync board at firmware versiond 3.4.7/3.4.8)
- At a glance, looks like possibly a digital signature issue, but no associated error messages about digital signatures are displayed. Driver installs with no problems. Issues are observed wit the Control Utility post driver install
- Issue associated with just some Windows systems (not Windows itself) with TSync firmware versions 3.4.7 and below installed (or with Special version 3.4.8)
- TSync **firmware update version 3.4.9 (~Dec 2018)** added a fix for this issue. Refer to: ..\..\PSB, PSP software updates\TSYNC boards\TSync firmware updates (PCIe, cPCI, PCI104)\TSync-PCIe\TSync-PCIe firmware updates\TSync_PCIe firmware.xlsx

  From the v3.4.9 F/W Release Notes: "Fixed an issue regarding PCIe operation on certain systems based on PCIe Base Address assignment"

Email from Dave Sohn to Dave Lorah (29 May 2019) I mentioned to Keith, but we should check and see if these units already have the Base Address Register fix that we released to resolve some Windows 10 access issues for Schneider Electric. The screenshots from the driver seem to indicate it is installed fine. Denis or Ron may have an idea of how we can check the Base Address Register in the unit to see if it is potentially this issue

Example symptoms observed with this issue


TSync Control Utility (Not Responding)



Days 0
Hours 0
Minutes 0
Seconds 0

How to identify the Base Address (BAR) issue is occurring

Q from Dave L to Denis Reilly "How can we check to see if the base address is the issue?"

A Reply from Denis (20 May 2019) When the BAR issue happens, the driver loads but all accesses to the card are invalid. I don't recall whether the OS just stalls altogether, or whether all accesses return '0'.

The easiest way to check is to look at the properties in the **Device Manager**. Click on "**Spectracom Timing board"** and then **Resources** tab, then look at the memory address range used by the card.
The card uses a 512 byte BAR, so you will see something like "00000000FEED0000 – 00000000FEED01FF".

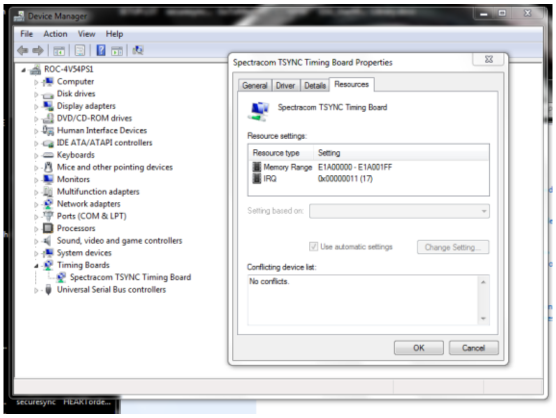
The issue is with the base address section I have highlighted in red – specifically, Bit 9 of the address. If the Base address is represented like it is above, then the customer will never see the issue, even with the old firmware, because Bit 9 is set to 0.

However, if the base address is "00000000FEED0200 – 00000000FEED0**3**FF", then that Bit 9 is high and you will see the problem unless you upgrade to the latest firmware.

Let me know if you need any more info,

Reply from Dave L to Denis I will ask the customer to verify this address for us. Attached (below) is a pic from my personal

TSync system as an example.



ROC-4V54PS1
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- Human Interface Devices
- IDE ATA/ATAPI controllers
- Keyboards
- Mice and other pointing devices
- Monitors
- Multifunction adapters
- Network adapters
- Ports (COM & LPT)
- Processors
- Sound, video and game controllers
- System devices
- Timing Boards
  - Spectracom TSYNC Timing Board
- Universal Serial Bus controllers

Spectracom TSYNC Timing Board Properties

Spectracom TSYNC Timing Board

Resource settings:

Resource type	Setting
Memory Range	E1A00000 - E1A001FF
IRQ	0x00000011 (17)

Setting based on:

☑ Use automatic settings

Conflicting device list:

No conflicts.

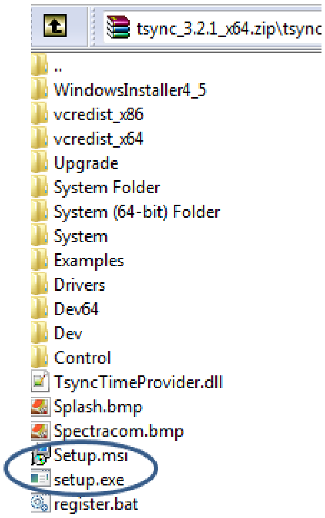
Windows driver v3.2.1 and above install issues due to the currently installed version off Visual C++ Redistributable installed on PC being newer than the version provided with the Windows driver (and don't want to remove current version to install the driver)

A) TSync driver version 3.2.1 and above

- To install the driver and Visual C++ Redistributables, they would normally run "**setup.exe**".
- If they run the "**setup.msi**", it should just install driver minus the Visual C++ Redistributables

➢ Refer to Salesforce case **25514**.

Solution: instead of running **setup.exe** file on the Driver CD, run the **setup.msi** file (this fiile just installs the driver by itself. Right-click on **setup.msi** and then click "**install**".



```
tsync_3.2.1_x64.zip\tsync
..
WindowsInstaller4_5
vcredist_x86
vcredist_x64
Upgrade
System Folder
System (64-bit) Folder
System
Examples
Drivers
Dev64
Dev
Control
TsyncTimeProvider.dll
Splash.bmp
Spectracom.bmp
Setup.msi
setup.exe
register.bat
```

Email from Tim T (2 June 17)
Keith,
There are 2 setup files that can be run to install our TSync Windows driver.

"setup.exe" & "setup.msi"

To install the driver and Visual C++ Redistributables, they would normally run "setup.exe".
If they run the "setup.msi", it should just install driver minus the Visual C++ Redistributables.

Have the customer try that and let me know.

Keith's resonse to customer

Thanks very much for your email and for reporting what was happening while installing the TSync Windows driver. Note that Spectracom Case Number **25514** was assigned to your report.

I ran this by our timing board engineers and found out there is a work-around to the Redistributables being installed. Here is the info about this (hope this good news help start you off to an even better weekend ☺)!:

Please note that there are 2 available setup files that can be run, to install our TSync Windows driver. These two files are “**setup.exe**” & “**setup.msi**”.

To install the driver and Visual C++ Redistributables, you would normally run “**setup.exe**”.

If you run the “**setup.msi**” (right-click and select install), it should just **install the driver, minus the Visual C++ Redistributables**.

B) TSync driver version 2.4.1, 2.5.0 and 2.5.1 can’t complete the driver install because a newer version of Visual Studio 2010 is already installed on the PC “A newer version of Microsoft Visual C++ 2010 Redistributable has been detected on the machine."

- Fixed in driver update version 3.1.2 for non-PTP boards

Note (24 Apr 2014): This issue was fixed in version 3.1.2 of the Windows driver.

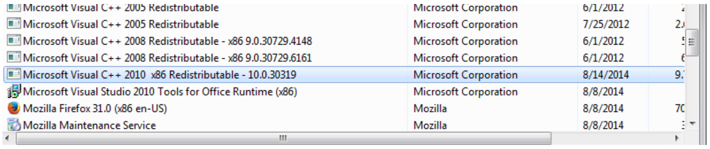
The Windows driver is fully compatible with Windows 7 and 64 bit. The specific issue you were observing is that earlier versions of the Windows driver requires the Visual C++ runtimes be installed on the machine, but it didn’t like it when it saw already installed a newer version of this distributable than it installs when it finds it’s not currently present on the machine. There wasn’t a way to skip this pre-requisite step of installing it, when a newer version is already installed.

This was fixed in the Version 3.1.2 release of the Windows driver (as indicated at the top of page 5 of the attached TSync driver Release Notes).   Note this latest version of the Windows driver can be downloaded from our website at:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=103

- This issue is not fixed in driver update 2.5.0 and 2.5.1 for PTP boards.
- Refer to Fogbugz case 1842 http://fogbugz/default.asp?pre=preMultiSearch&pg=pgList&pgBack=pgSearch&search=2&searchFor=1842&x=0&y=0
- Error message displayed when trying to install the driver: "A newer version of Microsoft Visual C++ 2010 Redistributable has been detected on the machine."
- Does not allow the driver install to proceed past this point.

Workaround for TSync driver versions 2.5.1 and below

- Temporarily uninstall all Microsoft C ++ 2010 Redistributables of 10.0.30319 and above (Control Panel -> Programs)



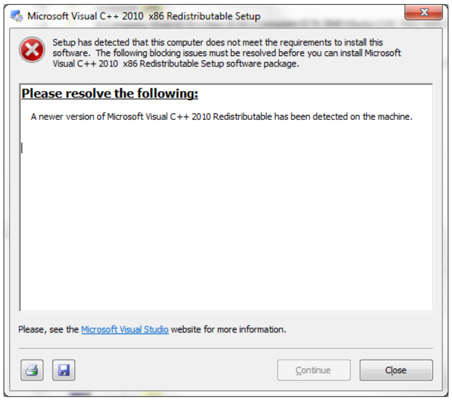
Microsoft Visual C++ 2005 Redistributable	Microsoft Corporation	6/1/2012	2
Microsoft Visual C++ 2005 Redistributable	Microsoft Corporation	7/25/2012	2.
Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.4148	Microsoft Corporation	6/1/2012	5
Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.6161	Microsoft Corporation	6/1/2012	6
Microsoft Visual C++ 2010 x86 Redistributable - 10.0.30319	Microsoft Corporation	8/14/2014	9.
Microsoft Visual Studio 2010 Tools for Office Runtime (x86)	Microsoft Corporation	8/8/2014	
Mozilla Firefox 31.0 (x86 en-US)	Mozilla	8/8/2014	70
Mozilla Maintenance Service	Mozilla	8/8/2014	3

1) Install Tsync driver. Driver will prompt if you wish to install Microsoft C ++ 2010 Redistributable **10.0.30319**
2) Press install to install this Redistributable.
3) The driver should finish installing.
4) Reinstall any newer Microsoft Redistributables as desired

Good draft email

I finally had a chance to discuss with our senior timing board engineer. I have some information and questions for you…

It appears the issue you are observing with the TSync-PCIe driver is associated with newer versions of the “Microsoft Visual C++ Redistributable” (related to Visual Studio 2012) already being installed in this PC. When the TSync-PCIe version 2.5.1 driver is being installed, it wants to install a specific version of the Microsoft 2010 Redistributable. But if there is already newer versions of the Redistributable installed, the TSync driver can’t install its version, and it can’t proceed past this required step. The screenshot below will be displayed:



Microsoft Visual C++ 2010 x86 Redistributable Setup

Setup has detected that this computer does not meet the requirements to install this software. The following blocking issues must be resolved before you can install Microsoft Visual C++ 2010 x86 Redistributable Setup software package.

Please resolve the following:

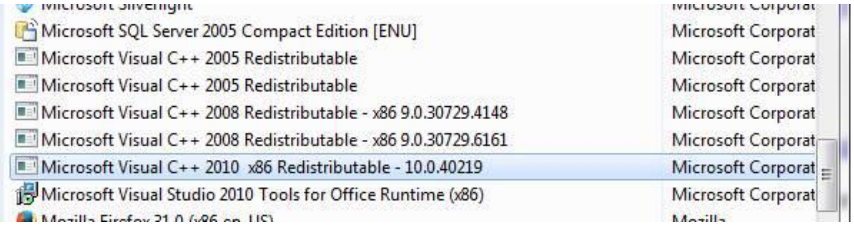
A newer version of Microsoft Visual C++ 2010 Redistributable has been detected on the machine.

Please, see the Microsoft Visual Studio website for more information.

The version 3.1.2 release of the TSync driver has incorporated changes to address this potential issue, if a newer redistributable is already installed. The catch is that the version 3.1.2 driver isn’t compatible with the PTP functionality of the TSync-PTP board. This is why you need to use the version 2.5.1 driver instead of the version 3.1.2 driver. The version 2.5.1 driver needs to also be modified to include changes incorporated in the version 3.1.2 driver, so that it can get past installing the earlier Redistributable, as necessary, I’m not sure when this Tsync driver will be able to be updated with the changes needed for this.

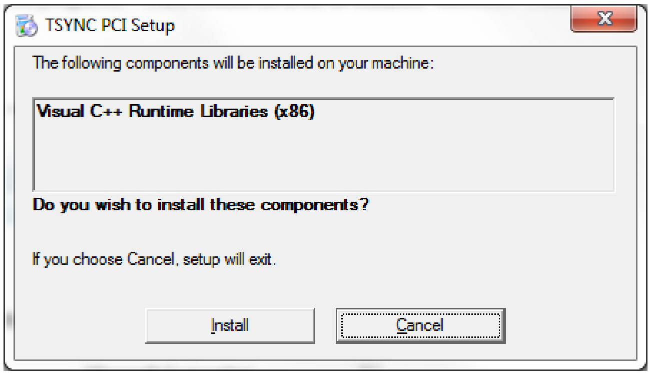
But there is a viable work-around for this condition. Would it be possible for you to temporarily uninstall the currently installed Microsoft 2010 redistributables on this PC, just long enough to install the driver? With any newer Redistributables now uninstalled from the system, the TSync driver will be able to install the earlier 2010 Redistributable included with the TSync driver and then it can complete its install process. Then, any time after the TSync-PCIe driver has been successfully installed, you can reinstall any newer version 2010 redistributables, as needed for programs such as VS-2012.

Newer version Redistributable can be uninstalled without the need to uninstall or alter Visual Studio. To uninstall newer Microsoft Redistributables, go to Control Panel -> Programs and then select View installed programs,   There should be a list of installed “Microsoft Visual C++ Redistributables” (as shown below):



Microsoft Silverlight	Microsoft Corporat
Microsoft SQL Server 2005 Compact Edition [ENU]	Microsoft Corporat
Microsoft Visual C++ 2005 Redistributable	Microsoft Corporat
Microsoft Visual C++ 2005 Redistributable	Microsoft Corporat
Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.4148	Microsoft Corporat
Microsoft Visual C++ 2008 Redistributable - x86 9.0.30729.6161	Microsoft Corporat
Microsoft Visual C++ 2010 x86 Redistributable - 10.0.40219	Microsoft Corporat
Microsoft Visual Studio 2010 Tools for Office Runtime (x86)	Microsoft Corporat
Mozilla Firefox 31.0 (x86 en-US)	Mozilla

Uninstall the 2010 Redistributables. Then run the TSync driver setup.exe installer program again. During the install, it will prompt you if you wish to install the "Visual C++ Runtime Libraries (x86) as shown below.   Click install to continue and complete the driver install and then reinstall newer Redistributables as required.



TSYNC PCI Setup

The following components will be installed on your machine:

Visual C++ Runtime Libraries (x86)

Do you wish to install these components?

If you choose Cancel, setup will exit.

Please let me know if you were able to temporarily uninstall the newer redistributables and successfully install the Tsync driver.

**Example programs included with the TSync Windows driver

The path to the drivers is: **C:/Program Files/Spectracom/TSync PCI/examples/Tsync API/bin**

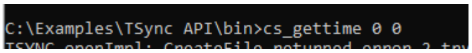
- If the Windows Example programs can't run, try using the **Windows Control Utility to** communicate with the TSync-PCIe board.

Running/performing the Example programs

- The Example programs cant be run directly from the folder where all the .exe programs are stored
- The Example programs need to be edited before being performed, and can only be run from the Windows command prompt window (type "**cmd**" in search to open command prompt
- In Windows (unlike Linux) the example programs are not case-sensitive

1) Open Windows Command Prompt window
2) Change to the directory where the Example programs are stored: Type: cd /Program Files/Spectracom/TSync PCI/examples/Tsync API/bin

   Type the desired call, just like in Linux (no need to add .exe to the command)



```
C:\Examples\TSync API\bin>cs_gettime 0 0
TSYNC_openImpl: CreateFile returned error 2 try
```

Custom Start-up scripts for Windows (auto-configuration at boot-up)

Q Keith fwd to Apps for SF Case 230571 (21 Apri2020) An example start-up script for auto configuration (I don't believe I have any examples to provide). Do you happen to have any examples?

"Configuration using the example programs was time-intensive. The user documentation mentions a startup script without providing any sample or detail: can you provide a sample windows 10 start up script and instructions for installation, etc?"

A (reply from Ron Dries 21 Apr 2020) There is no start-up script examples. There are two options that she can use:

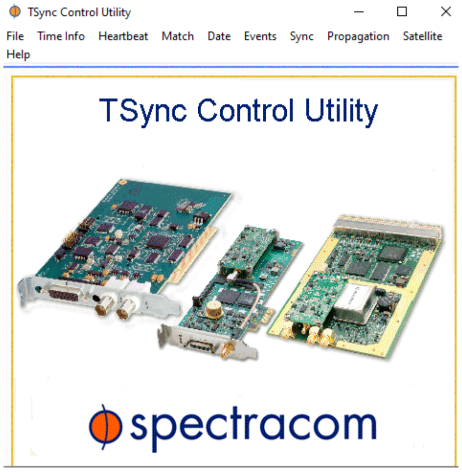
1. The Windows driver provides the source code for each of the example programs and they can write their own software and builds it against the libraries and include files.

2. They can write a script, python for example, that calls the example programs we provide in the order that they would have issued them manually.

For option 2 it may be possible for us to provide an example if necessary. Option 1 is more in depth then they probably need.

**Windows Control Utility GUI interface

- **To open the Control Utility: Start / All Programs / Spectracom Corp (**now **Orolia)/ TSync PCI / TSync Control Utility**
- Provides a basic interface with the TSync-PCIe board via a GUI (such as reading the current Sync status, for example)
- The TSync-PCIe board does not need to be installed to use this GUI. Install the Windows driver and this GUI can be run in a "demo" mode to demonstrate the capabilities of the utility



Syncing Windows using a TSync Timing board (Clock Daemon/TSync Time Provider)

TSync boards can sync Windows (PCs or VMWare) using either:

- Via Windows Time (**w32Time)** in Windows 2016, and the much more recent "**TSync Time Provider**" (included in versions 3.2.1 and above of the **64 bit** (not the 32 bit) version of the TSync-PCIe Windows driver) Refer to "**A**" below
- The much earlier **Clock Daemon program/Clock Daemon service**. Refer to "**B**" further below.

A) TSync Time Provider" (sync a newer version of Windows, such as Windows 2016, to a TSync-PCIe board using the Windows Time Service (W32Time)

1. Windows Tsync driver (versions 3.2.1 or above) and Windows 2016

- Refer to Salesforce cases such as case **171013**
- "**TSync Time Provider**" is an Interface between TSync Windows driver and W32Time, as was added to Windows and to the Tsync driver version 3.2.1 (64 bit variant of the driver only)

From the TSync Windows driver v3.2.1 release notes:

This is described in the TSync driver release notes (version 3.2.1" section, on page 3, and as excerpted below): https://spectracom.com/sites/default/files/document-files/TSync%20Release%20Notes%20for%203.2.1%20Windows%20Driver.pdf

Windows time integration: The 64-bit version of the driver now includes a TSync time provider module, allowing the Win-32 time service to communicate directly with the Spectracom TSync

Here is some information on time providers from Microsoft ("Accurate Time for Windows Server 2016"): https://docs.microsoft.com/en-us/windows-server/networking/windows-time-service/accurate-time

2. **Windows TSync driver (earlier versions prior to version 3.2.1) or Windows OS prior to Windows 2015**

   - "TSync Time Provider" capability isn't available in TSync Windows drivers before version 3.2.1, or in Windows OS prior to Windows 2016.
     - Refer to "**clock daemon**" (further below in this document) as the alternative approach (refer to "**B**" further below)

from Salesforce case 171013

Q We have the card installed in an HPE server running VMWare ESXi, and then passed up to a virtual machine running Windows Server 2016. However, the actual clock on that VM appears to be several milliseconds (3-9ms) off from the IRIG time source that the Tsync card is connected to.

I'm hoping to get some guidance on how the TSync card and software work with Windows Server 2016, and if there are any documents/guides on configurations of the Tsync product to allow the Windows server to act as a time server (I have it configured per Microsoft guidance, but since the clock appears to be off from other sources, we're being generally marked as a false ticker).

A reply from Ron Dries (7 Aug 18); You are correct that the virtual environment can have a negative impact on the timing accuracy.

However in Windows server 2016 the customer can take advantage of the TSync Time Provider included with the 64 bit version of the driver.

This is described in the release notes: https://spectracom.com/sites/default/files/document-files/TSync%20Release%20Notes%20for%203.2.1%20Windows%20Driver.pdf

Here is some information on time providers from Microsoft: https://docs.microsoft.com/en-us/windows-server/networking/windows-time-service/accurate-time

Keith's response to customer (7 Aug 18) …They have confirmed my understanding that time synchronization in a virtual environment can inherently have a negative impact on timing accuracies. The errors you are seeing aren't surprising in this type of application. With other NTP servers other providing time info. especially if they are only a couple/few hops away (reducing the likelihood if having asymmetrical path delays), it's not unexpected they will be able to provide better time than the TSync board can provide, causing it to be marked as a false ticker.

As for configuring Windows 2016 to be a true NTP server (not just able to sync its Windows clients), you can take advantage of the TSync Time provider provided with the 64-bit version of the TSync Windows driver. Additional information on this functionality is available at the two links referenced below:

This is described in the TSync Windows Driver release notes (the "version 3.2.1" section, on page 3, and as excerpted below): https://spectracom.com/sites/default/files/document-files/TSync%20Release%20Notes%20for%203.2.1%20Windows%20Driver.pdf

Windows time integration: The 64-bit version of the driver now includes a TSync time provider module, allowing the Win-32 time service to communicate directly with the Spectracom TSync card.

Here is also some information on time providers from Microsoft: https://docs.microsoft.com/en-us/windows-server/networking/windows-time-service/accurate-time

Syncing TSync board to the Windows OS (via Host Reference)

- the opposite of the normal desire to sync Windows OS using a TSync board
- Refer to Salesforce Case 174508
- Not a commonly desired configuration

A) Periodically use Host reference "HR_setTime" command (Best method for accurate sub-second sync)

See details of Host Reference mode: **Host Reference: TSync-PCIe board without external inputs (host/EPP and Self/Self) (Partial except further below)

- TSync board cannot directly read the Windows Date/Time from the OS.
- Instead of directly obtaining the Windows date/time, this info is first obtained via some external means
- In between the HR_SetTime commands, the TSync board is in free-run mode.
- Customer can create a custom script to perforn the follwong steps:

1. customer is responsible to periodically "read" the OS time, using a method they choose (such as the "time command). Refer to sites such as: https://docs.microsoft.com/en-us/windows-hardware/test/wpt/time-and-timestamp-formats and https://www.windows-commandline.com/get-date-time-batch-file/

   Note: its not recommended to read/set the time of the TSync board every second, so we recommend reading/setting the time not more often then like every two seconds.

2. Custoner is responsible to convert the date/time read from Windows OS into UTC timescale (unlesss of couse, Windows is providing the date/time in UTC timescale. already)
3. Customer is responsible for accounting for the inherent delays between reading the time, parsing it, and issueing the command to set it (parsing and issuing command in the next step below)
4. Customer is responsible for then parsing the raw date/time into the **HR_SetTime** API call/example program and then issuing the applicable command.

   Syntax: **HR_SetTime (device index – typically 0) (year) (DOY) (Hour) (minute) (second) (nanoseconds)**

Reading the PC's/System's time to use in the HR_SetTime command

- In SecureSync, we run the HR_SetTime command once-per-minute when syncing to NTP (stratum 2 operation). This is a good interval to use for typical NTP accuracies, even with just a TCXO oscillator installed. However, they can perform it more often than once a minute, if they wish

Email from Dave Sohn (9 Mar 15) They can do the same thing with the TSync that we do inside the SecureSync for Stratum 2 operation. If the host reference is set as the time and 1PPS reference, it will set the time as well as adjust the 1PPS point as well. In order to keep synchronization, they will need to periodically set the host reference time using HR_SetTime.

- Most accurate method to auto-read and periodically set the time
- This method is intended for Linux- not Windows
- Refer to the draft email Keith has for the actual script attachment.

Email Keith sent based on info from Tim Tetreault (9 March 15) Regarding the desire to sync the Spectracom TSync-PCIe timing board to your NTP-synced system, I have some additional information for you that should help...

Below you should find a simple script that can be used to periodically set the TSync's board to your system's time:

```
#!/bin/sh

# set HW_SetTime with UTC system time
./HR_SetTime 0 0 `date +%Y.%j.%H.%M.%S.%N -u| tr "\." " "`

exit
```

Note that we also use this script in the code of one of our other products which is designed around the TSync-PCIe architecture, for a similar function as you are using. We run this code once-per-minute to keep the system time aligned with the host's time. We have found this interval to be sufficient enough to keep the time well within the capabilities of NTP. However, you can run this script more often than once-per-minute to set the time in the TSync board more often, if you wish.

B) Use the "CS_SetTime" command (not recommended for accurate sub-second sync)

- Refer to: (in this doc): **Host Reference: TSync-PCIe board without external inputs (host/EPP and Self/Self) (excepted below):

CS_SetTime call/example program (Less accurate method than HR_SetTime, to read and set the time – per an earlier email)

Host time allows the time of the TSync board to be set to a user-specified time and with the same call or example program, the time is also declared valid. The TSync-PCIe board and associated driver don't have the ability to read the computer's time/date automatically (The computer's time can be read automatically, but this has to be done outside of the TSync-PCIe's driver). Then, once the time/date has been read, a call can be used to simultaneously set the time and declare the time valid. The reading of the computer's time and the setting of the time/date in the TSync-PCIe board can be performed with a customer-written Windows script.

A good starting point to create a Visual Basic script that can read the Windows PCs date/time is http://technet.microsoft.com/en-us/library/ee198917/

Q. I noticed that after I set the time with: TSYNC_CS_SetTimeSec(index, seconds, ns) I continue to read old time values for approximately 1250ms. Is this normal? This is causing me problems. Other than sleeping for 2 seconds I don't see a good workaround. What do you recommend?

A Below is the response from Engineering related to your recent question:

> "The delay the customer is seeing when running a CS_SetTime makes sense. The time is only set on the "On time point" which is once a second. There is also some overhead required on the board so I can see it taking 1.5 seconds for the new time to be seen."

(**Note**: Applicable to **1.x.x** firmware versions only) With Host mode, you also have to command validity of the Host time before it can use the Host time as a reference. **TSYNC_HR_SetValidity 0 0 1 1** <enter> (Set the reference validity of the Host)

C) Using NTP ported by Meinberg to work in Windows (not a suggusted solution!)

- per Dave Sohn (26 Nov 2018) the Meinberg port of NTP for Windows does not contain our "TSync Refernce

Clock driver". So this isn't a viable solution to sync a board to Windows via NTP

Using TSync in Virtual Machines/VMs/Virtual environments (such as VMware/ESX and ESXI, Hyper-V)

- Refer also to “**Syncing Virtual Machines/VMs (such as VMware/ESX and ESXI, Hyper-V)” in: ..\CustomerServiceAssistance.pdf**
- Refer also to various cases in Salesforce

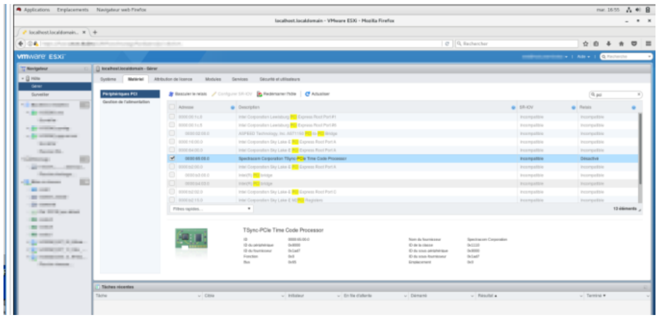
Email from Ron Dries (8 Aug 2018) I found this article from VMware about time synchronization: https://kb.vmware.com/s/article/1318

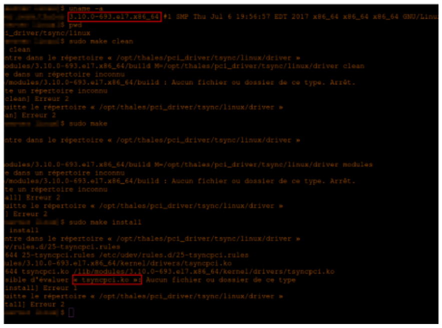
I would still recommend they try to utilize the TSync Time Provider in server 2016 as Microsoft has made many improvements with time synchronization in Windows 10 and Windows Server 2016. Also I would recommend taking a look at the VMware recommendations for time synchronization.

Driver Build/Compile errors while installing Linux driver in VMWARE ESXi (tsyncpci.ko error)

- Refer to Case 218892

(error message converted from French): “impossible to swallow <tsyncpci.ko > : no files or folders for this type. Stop





```
[redacted]$ uname -a
[redacted] 3.10.0-693.el7.x86_64 #1 SMP Thu Jul 6 19:56:57 EDT 2017 x86_64 x86_64 x86_64 GNU/Linu
[redacted]$ pwd
i_driver/tsync/linux
[redacted]$ sudo make clean
 clean
ntre dans le répertoire « /opt/thales/pci_driver/tsync/linux/driver »
odules/3.10.0-693.el7.x86_64/build M=/opt/thales/pci_driver/tsync/linux/driver clean
e dans un répertoire inconnu
/modules/3.10.0-693.el7.x86_64/build : Aucun fichier ou dossier de ce type. Arrêt.
te un répertoire inconnu
clean] Erreur 2
uitte le répertoire « /opt/thales/pci_driver/tsync/linux/driver »
an] Erreur 2
[redacted]$ sudo make

ntre dans le répertoire « /opt/thales/pci_driver/tsync/linux/driver »


odules/3.10.0-693.el7.x86_64/build M=/opt/thales/pci_driver/tsync/linux/driver modules
e dans un répertoire inconnu
/modules/3.10.0-693.el7.x86_64/build : Aucun fichier ou dossier de ce type. Arrêt.
te un répertoire inconnu
all] Erreur 2
uitte le répertoire « /opt/thales/pci_driver/tsync/linux/driver »
] Erreur 2
[redacted]$ sudo make install
 install
ntre dans le répertoire « /opt/thales/pci_driver/tsync/linux/driver »
v/rules.d/25-tsyncpci.rules
644 25-tsyncpci.rules /etc/udev/rules.d/25-tsyncpci.rules
ules/3.10.0-693.el7.x86_64/kernel/drivers/tsyncpci.ko
644 tsyncpci.ko /lib/modules/3.10.0-693.el7.x86_64/kernel/drivers/tsyncpci.ko
sible d'évaluer « tsyncpci.ko »: Aucun fichier ou dossier de ce type
install] Erreur 1
uitte le répertoire « /opt/thales/pci_driver/tsync/linux/driver »
tall] Erreur 2
[redacted]$
```

**Windows Hyper-V (ESXI hypervisor) and other Virtual machines (VMs

- Refer to “**Windows Hyper-V (ESXI hypervisor) and W32Time (Windows Time Service)” in:** ..\CustomerServiceAssistance.pdf

Per brief conversation with Tim Tetreault (~end of Aug, 2018) He is not aware of anything needing to be done to our TSync driver to allow it to be used in a virtual machine/virtual envirnment. Anything that would need to be done is all handled when the virtual machine is craeated/configured.

Email from Ron Dries to Keith (31 Aug 2018) As we discussed today the tsync driver can be installed on the virtual operating system. We do not have any support for directly installing on the hypervisor as far as I know.

I would recommend having the customer install the driver on their virtual OS and then work with VMWare to get recommendations to configure ESXI in the optimal way for the bets timing solution.

I found this article from VMware about time synchronization: https://kb.vmware.com/s/article/1318

I would still recommend they try to utilize the TSync Time Provider in server 2016 as Microsoft has made many improvements with time synchronization in Windows 10 and Windows Server 2016. Also I would recommend taking a look at the VMware recommendations for time synchronization.

Sent to Ron Dries Regarding SF case 171013, (for Northrop Grumman) I just want to confirm my intended response for this TSync-PCIe inquiry. I’m fairly certain I have the correct info to respond with, but just want to make sure before I send it. Below in red is the inquiry initially sent to Steve Visosky and forwarded to us:

We have the TSync card installed in an HPE server running VMWare ESXi, and then passed up to a virtual machine running Windows Server 2016.

We have installed the Tsync software, and the utility says it is synchronized. Also verified that the clock daemon service is running.

However, the actual clock on that VM appears to be several milliseconds (3-9ms) off from the IRIG time source that the Tsync card is connected to.

I'm hoping to get some guidance on how the TSync card and software work with Windows Server 2016, and if there are any documents/guides on configurations of the Tsync product to allow the Windows server to act as a time server (I have it configured per Microsoft guidance, but since the clock appears to be off from other sources, we're being generally marked as a false ticker).

My understanding is that Time synchronization in a virtual environment is inherently poor with lower accuracies/more jitter. And so, the time offsets being observed while syncing to the TSync board (3-4 milliseconds) doesn’t sound the least bit unexpected.

With other servers also configured (in addition to the TSync board) it’s highly likely those other severs have lower offsets (especially if they are only a couple/few hops away), inherently resulting in the TSync board being classified a false ticker.

My recommendation, if they are looking for timing accuracies, is to use a true Windows (or Linux) machine and not a virtual machine to be an NTP server.

As for making Windows be treated as a true NTP server (beyond local NTP being able to sync just Windows clients), the only way I know of doing this is to install Meinberg’s port of NTP for Windows. I don’t like to advertise this program’s availability, because it has their name all over it ☹!!

Are there any other ways (for Windows) besides Meinberg’s NTP port??

A Reply from Ron Dries (7 Aug 2018) You are correct that the virtual environment can have a negative impact on the timing accuracy.

However in Windows server 2016 the customer can take advantage of the **TSync Time Provider** included with the 64 bit version of the driver.

This is described in the release notes: https://spectracom.com/sites/default/files/document-files/TSync%20Release%20Notes%20for%203.2.1%20Windows%20Driver.pdf

Here is some information on time providers from Microsoft: https://docs.microsoft.com/en-us/windows-server/networking/windows-time-service/accurate-time

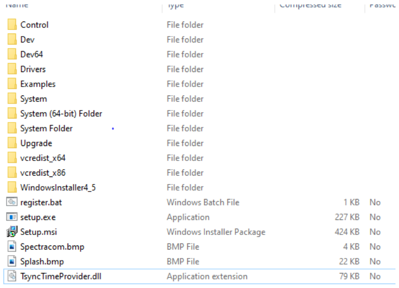
“TSync Time Provider” (“TsyncTimeProvider.dll” file) for W32Time (Windows Time Service)

- Availalable in Windows Server 2016, Windows 2010
- Refer to https://docs.microsoft.com/en-us/windows/win32/sysinfo/time-provider (excerpt below)

“The Microsoft Windows operating system provides support for a variety of hardware devices and network time protocols using the time provider architecture. Input time providers retrieve accurate time stamps from hardware or the network, and output time providers provide time stamps to other clients on the network.

Time providers are managed by the time provider manager. It is responsible for loading, starting, and stopping time providers as directed by the service control manager. This interface makes writing a time provider easier than writing a full service.”

- in Windows server 2016 (and Windows 10) the customer can take advantage of the TSync Time Provider (“**TsyncTimeProvider.dll**” file) included with the 64 bit version of the driver.



Name	Type	Compressed size	Passw
Control	File folder		
Dev	File folder		
Dev64	File folder		
Drivers	File folder		
Examples	File folder		
System	File folder		
System (64-bit) Folder	File folder		
System Folder	File folder		
Upgrade	File folder		
vcredist_x64	File folder		
vcredist_x86	File folder		
WindowsInstaller4_5	File folder		
register.bat	Windows Batch File	1 KB	No
setup.exe	Application	227 KB	No
Setup.msi	Windows Installer Package	424 KB	No
Spectracom.bmp	BMP File	4 KB	No
Splash.bmp	BMP File	22 KB	No
TsyncTimeProvider.dll	Application extension	79 KB	No

- This is described in the TSync-PCIe version 3.2.1 Windows driver release notes (and excerpted below from page 3) https://spectracom.com/sites/default/files/document-files/TSync%20Release%20Notes%20for%203.2.1%20Windows%20Driver.pdf

  “Windows time integration: The 64-bit version of the driver now includes a TSync time provider module, allowing the Win-32 time service to communicate directly with the Spectracom TSync card.”

- Here is some additonal information from Microsoft, about “**time providers**” for Windows Time: https://docs.microsoft.com/en-us/windows-server/networking/windows-time-service/accurate-time

Q **Sent to Ron Dries Regarding SF case 171013, (for Northrop Grumman**) I just want to confirm my intended response for this TSync-PCIe inquiry. I’m fairly certain I have the correct info to respond with, but just want to make sure before I send it. Below in red is the inquiry initially sent to Steve Visosky and forwarded to us:

We have the TSync card installed in an HPE server running VMWare ESXi, and then passed up to a virtual machine running Windows Server 2016.

We have installed the Tsync software, and the utility says it is synchronized. Also verified that the clock daemon service is running.

However, the actual clock on that VM appears to be several milliseconds (3-9ms) off from the IRIG time source that the Tsync card is connected to.

I'm hoping to get some guidance on how the TSync card and software work with Windows Server 2016, and if there are any documents/guides on configurations of the Tsync product to allow the Windows server to act as a time server (I have it configured per Microsoft guidance, but since the clock appears to be off from other sources, we're being generally marked as a false ticker).

My understanding is that Time synchronization in a virtual environment is inherently poor with lower accuracies/more jitter. And so, the time offsets being observed while syncing to the TSync board (3-4 milliseconds) doesn't sound the least bit unexpected.

With other servers also configured (in addition to the TSync board) it's highly likely those other severs have lower offsets (especially if they are only a couple/few hops away), inherently resulting in the TSync board being classified a false ticker.

My recommendation, if they are looking for timing accuracies, is to use a true Windows (or Linux) machine and not a virtual machine to be an NTP server.

As for making Windows be treated as a true NTP server (beyond local NTP being able to sync just Windows clients), the only way I know of doing this is to install Meinberg's port of NTP for Windows. I don't like to advertise this program's availability, because it has their name all over it ☹!!

Are there any other ways (for Windows) besides Meinberg's NTP port??

A Reply from Ron Dries (7 Aug 2018) You are correct that the virtual environment can have a negative impact on the timing accuracy.

However in Windows server 2016 the customer can take advantage of the TSync Time Provider included with the 64 bit version of the driver.

This is described in the release notes (excerpted below): https://spectracom.com/sites/default/files/document-files/TSync%20Release%20Notes%20for%203.2.1%20Windows%20Driver.pdf

> Windows time integration: The 64-bit version of the driver now includes a TSync time provider module, allowing the Win-32 time service to communicate directly with the Spectracom TSync card

Here is some information on time providers from Microsoft: https://docs.microsoft.com/en-us/windows-server/networking/windows-time-service/accurate-time

Steps to use Windows Time Provider with a TSync timing board

Note: The 64bit variant of the Windows driver (version 3.2.1 or above) needs to be installed in the PC.

(25 Feb 2020) per Ron Dries, he is not aware of any "TSync specific info" on setting up W32time to use the TSync board. He just Googles "Time Provider" and uses the Windows instructions (link below).

- As part of the Windows driver install, we put the associated **TsyncTimeProvider.**dll" file from the driver into the Windows PC (in the correct location)
- Apparently, Windows Time (w32Time) while running, will automatically start syncing to the TSync-PCIe board??
- Refer to https://docs.microsoft.com/en-us/windows/win32/sysinfo/time-provider (excerpts below)

Time providers are managed by the ***time provider manager*.** It is responsible for loading, starting, and stopping time providers as directed by the service control manager. This interface makes writing a time provider easier than writing a full service.

A) Creating a Time Provider

B) Registering a Time Provider

The time provider manager enumerates the keys under the **TimeProviders** key and starts each enabled provider. Providers are started at system startup and whenever there are parameter changes.

HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\W32Time\TimeProviders\*ProviderName*

The following table describes the values that must exist in each provider's key.

Table 1

Value	Description
DllName	The name of the DLL that contains the provider. This value has the type **REG_SZ**.
Enabled	Indicates whether the provider should be started. If this value is 1, the provider is started. Otherwise, the provider is not started. This value has the type **REG_DWORD**.
InputProvider	Indicates whether the provider is an input provider or an output provider. If this value is 1, the provider is an input provider. Otherwise, the provider is an output provider. This value has the type **REG_DWORD**.

Inherent issues with NTP Stability in VMs

- Not at all unexpected for VMs running ntpd to have NTP log entries(var/log) indicating errors of greater than 500PPM ("**frequency error -512 PPM exceeds tolerance 500 PPM**")

Recommended solutions:

1. **Recommendation to use -q and -g switch when starting NTP**

   - Refer to https://access.redhat.com/solutions/35640

   -g
   Normally, ntpd exits if the offset exceeds the sanity limit, which is 1000 s by default. If the sanity limit is set to zero, no sanity checking is performed, and any offset is acceptable. This option overrides the limit and allows the time to be set to any value without restriction; however, this can happen only once. After that, ntpd will exit if the limit is exceeded.

   -q
   Exit the ntpd just after the first time the clock is set. This behavior mimics that of the ntpdate program, which is to be retired. The -g and -x options can be used with this option.

2. **Recommendation to add "Tinker panic 0" to the TOP of the ntp.conf file of VM clients**

   - **Refer to:** https://kb.vmware.com/selfservice/microsites/search.do?language=en_US&cmd=displayKC&externalId=1006427
   - **Tinker panic 0** instructs NTP to not give up if it sees a large jump in time. This is important for coping with large time drifts and also resuming virtual machines from their suspended state.
     - **Note**: The `tinker panic 0` directive must be at the top of the `ntp.conf` file.

**B) Clock Daemon program / Clock Daemon service (for all timing boards) and 32 bit Windows

Important Note: Clock deamon (program and service) is **only compatible with 32 bit Windows**. Its not compatible with **64 bit Windows**. (**for 64 bit Windows,** use the TsyncTimeProvider. dll fille with w32Time (Windows Time program) as described above

For more info on Clock daemon, refer to:

- **TSync-PCIe boards:** TSync family driver guide (1219-5001-0050) for more information on the Clock daemon:

  in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

- **TPRO/TSAT-PCI boards:** Section 2.2 of the PCI Application Programmers manual (*"2.2 Clock Daemon Utilities")* discusses the Clock Daemon program (refer to1186-5002-0050: I:\New Released\Manuals\1186-xxxx-xxxx)0

  - ➢ Consists of the "ClockDaemon.exe" and the "ClockDaemonService.exe". Only one of these two can be used at a time.

Difference between the Clock daemon (program) and Clock daemon service:

- ➢ **Clock daemon** (program) needs to be started after each PC boot.
- ➢ **Clock daemon service** is a Windows service, so it can start automatically after each PC boot.

Accuracy of the Clock daemon program/service

- ➢ The PCIe/PMC/cPCI timing board will try to set the time to **within 1 microsecond** (the board's resolution). However, due to latencies and processing in Windows, typical expected accuracies (to its board's time) **is around 1 to 2 ms**.

Windows compatibility

- ➢ Clock Daemon and Clock Daemon service are compatible with 32 bit Windows
- ➢ Not compatible with 64 bit Windows (such as Windows 7, 64 bit)

Windows 7- report of blue screen

- ➢ Refer to Salesforce case 17384
- ➢ Email from this customer- "The clock damon is one of the few apps I've seen that is able to blue screen Win 7. This app fights everything going on in a PC and if the PC is connected to a network that is pushing time... BSCOD will be a regular occurrence. I would suggest you guys scrap this app and remove it from your software package."

(prior to driver version 3.1.0 only) Earlier email Keith sent to another customer- Please also note the Windows driver is only compatible with Windows 7, 32 bit. It is not compatible with 64 bit Windows. At this time, there are no plans to make this driver compatible with 64 bit Windows

Status update to above note (7 aug 18) - No longer the case. We have since released a 64 bit Windows driver (starting with Windows driver version 3.1.0)

Issue: Need to rename a folder from “Orolia’ to “Spectracom” after installing Windows driver (if older version driver was previously installed)

- Refer to Salesforce Case 221233
- Results in a **Spectracom** folder (for older version driver) and **Orolia** folder (for newer driver) being located in the **C:/Program Files folder**

Email from Ron Dries to Keith (23 March 2020) Nidal reported a similar issue and we found that when installing the 3.2.3 driver onto a PC that already had a previous version of the driver the clock daemon service was not being removed and reinstalled correctly to point to the new Orolia folder.

This has not been addressed yet but can be worked around by manually removing the clock daemon service before installing the 3.2.3 driver. Here are the instructions I sent to Nidal in the past:

When installing the 3.2.3 TSync Driver onto a machine with an existing 3.2.1 TSync driver we noticed this issue.

The Clock Daemon service does not get removed or overwritten during the install and the 3.2.1 version of the driver used C:/Program Files/Spectracom. In 3.2.3 we changed the installation directory to C:/Program Files/Orolia.

To correctly address this issue we would recommend uninstalling the software from Add or Remove programs, uninstalling the driver, and then running the following command to delete the old Clock Daemon Service “sc delete CLOCKDAEMON”

Then when installing the 3.2.3 driver everything will be pointing to the correct location.

Instead of deleting older driver:

1) rename the older folder in C: program Files from “**Spectracom** to “**OLD Spectracom**”

2) rename the newer folder from “**Orolia’** to “**Spectracom**”

3) restart the Clock daemon service (**Start** > **Computer Management**. Select **Services** and scroll down to **Clock Service**

Sync a Windows PC to a TSync-PCIe board using Windows Clock Daemon (instead of using “TSync Time Provider” and Windows 2016, as discussed further above)

A Windows PC won’t automatically sync to an installed TSync-PCIe-PTP board. Instead, the TSync-PCIe’s Windows driver contains a free utility program that is automatically installed as part of the driver. This program allows a Windows PC to be able to synchronize to the timing board. This utility is called “clock daemon”.
With the TSync-PCIe synced to an input reference (such as PTP, as indicated by the green LED on the edge of the board being lit), the clock daemon allows the PC to sync to the time of the TSync-PCIe board at scheduled intervals.

Attached you should find a copy of the TSync-PCIe PCI driver guide. This provided utility is discussed in Section 2.3.5 of this document (starting on page 2-7). There are two variations of this utility (clock daemon and clock daemon service) discussed in this document. You can use either one, but not both simultaneously (the second one runs as a Windows Service instead of as a program).

Desire to sync a Windows PC each time the PC boots-up

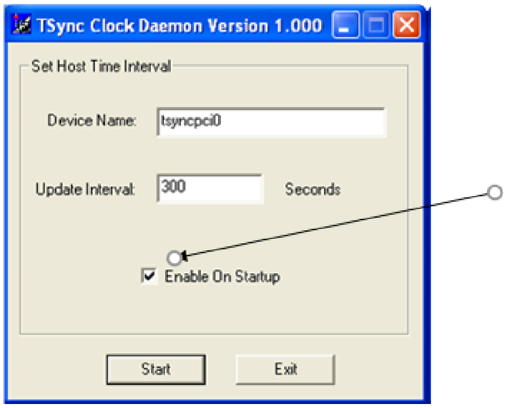
There are two supplied versions of this program. These are the “Clock Daemon” and the “Clock Daemon Service”. Only one of these two can be used at a time. Only the “Clock Daemon Service” is treated as an actual Windows service that starts up each time the PC starts. The “Clock daemon” program is not a Windows service and won’t automatically start upon a Windows reboot (user interaction is required when using this program). We recommend you switch to using the “Clock Daemon Service” if you wish for the program to start automatically after each Windows start-up.

Update interval / Update on Windows startup (for the clock daemon- not for the clock daemon service)

- Default update interval is **every 300 seconds**

Per the *Clock Daemon* section of the driver guide, it can optionally sync at Windows start-up, instead of waiting for the first interval after the program has started:

For the Clock Daemon program:



Device Name: tsyncpci0

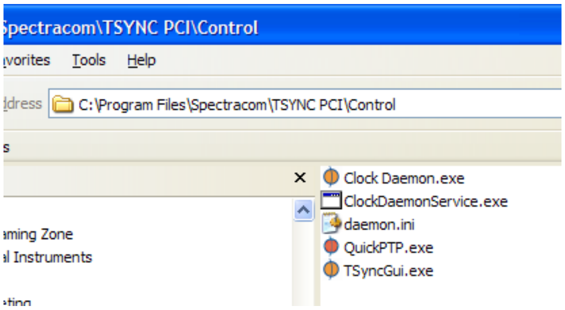
Update Interval: 300 Seconds

☑ Enable On Startup

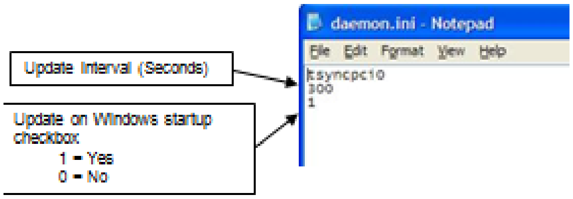
Enable On Startup: Checking this box will automatically start system clock synchronization by the clock daemon whenever the program itself starts. Alternatively, the synchronization can be started or stopped manually.

For the Clock Daemon Service:

- The controls for device name and update interval are setup by the “daemon.ini” located in the “**Spectracom\TSYNC PCI\Control**” folder with the clock daemon executables. This initialization file is shared by both clock daemon executables.



The “**1**” is “Enable on Startup” selected. When not selected, the PC will wait the Update Interval after the program starts before its first sync.

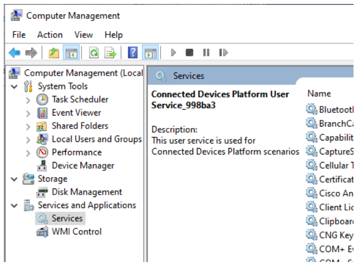

daemon.ini - Notepad
File Edit Format View Help
Update Interval (Seconds)
Update on Windows startup checkbox
1 = Yes
0 = No

```
tsyncpc10
300
1
```

How to tell if clock daemon service has been started/is running (also a way to start/stop the clock service)

- **Windows Start** > **Computer Management**. Select **Services** and then scroll down the list of Services to **Clock Service**

Per Ron Dries (25 Feb 2020) The clock daemon will start a service that will be running/ so he will need to check the Windows Services menu, or see if there is a icon in the bottom right task bar that says “**clock daemon**”. It runs in the background.



Connected Devices Platform User Service_998ba3

Description:
This user service is used for Connected Devices Platform scenarios

Local time/DST correction (hours not correct when the PC syncs to the board)

The year has to be correct in the board. Otherwise, it’s too large of a time correction for the PC.

Is the IRIG generator/source providing the TSync board with UTC time (similar to GMT)? or is the IRIG source outputting local, DST-corrected time to the board?

This may get a little “bumpy” for a few moments…

FYI- By default, the board is assuming its IRIG input is in UTC timescale, and will output to the PC this same timescale that its receiving (as UTC). Normally, the board provides the computer with UTC timescale with no offset for Time zone and no DST correction. This adjustment is normally handled by the computer (the setting for offset and DST in the Date/Time properties window, which opens when you double-click the Windows clock on the bottom of the screen,

Unless you tell the board that its receiving local time from the source by creating and applying a local clock to its IRIG input (thereby telling the board how it can convert the local tine to UTC by removing the offsets), there will be a double correction to the PCs “hours” value if the computer is currently configured to display local time. (the source is offsetting the time for time zone and DST, while the computer is also applying the same offset to the offset time,, if you see the hours are off by several hours, tell the Windows computer in the Date/time properites screen to not adjust for DST, and not to apply any local time offset (set it to be GMT time), Now only the IRIG source will be providing the PC with the offset for local/DST corrected time.

The other two alternatives would be to have the IRIG source output UTC time (instead of local time) to the board, or to create and apply a local clock to the IRIG input configuration of the TSync board, so that it can change its local input to UTC. Then the board will provide the computer with UTC time, and the computer alone with adjust the hours for local time, and for DST,

The best test to verify the clock utility is working is to offset the PCs hours and minutes to values that you know are wrong. Then with clock daemon running, both the hours and minutes should change within the time period that the daemon resyncs the computer (every 5 minutes by default), if even just the minutes change, the daemon is working. But if the hours on the PC goes to the wrong time, this is a factor of the time offsets explained above.

Logging/information about PC drift rates

Q. Does the Clock Daemon create any logs for its time syncs?

The Clock Daemon program/service by default syncs the Windows PC to the timing board every 300 seconds. However, this value can be modified by a user as desired, via its GUI interface. The only disadvantage of setting the time very often is that Windows sends a signal to all running programs each time the time has been synced. Syncing it too often (such as once a minute for example) may cause issues with programs due to this signal often being sent. Otherwise, its fine to decrease the interval to sync the PC more often (to minimize the time that the computer is drifting).

This basic program/service just resets the time of the computer at the specified interval, based on PC drift since the last sync occurred. It does not log each time it syncs the PC or display any statistics on the PCs drift that is occurring,

However, another method of syncing Windows PCs which can provide time sync information is to use one or more of our Spectracom NTP servers connected to the network (instead of a bus timing board in each PC) in conjunction with our PresenTense suite of NTP Client software. PresenTense software is also a GUI-based software solution, but it also provides logging and alert notification capabilities if computers are not able to sync to the NTP server, and therefore begin to drift. A Spectracom NTP server can provide time stamps over Ethernet connections to all machines on the network. And if more than one NTP server is on the network, this client software can be configured to get time from up to three different NTP servers for redundancy.

Attached you should find a document that discusses how to sync Windows PCs. The last section of this document (Section 11) discusses this NTP client software, including what each program in the suite can provide.

Troubleshooting the clock daemon not working correctly

A) Leap year rollover issues

- Refer to Salesforce case 20592
- The Clock daemon has no bearing on leap year rollover. It just reads the data/time provided by the TSync or TPRO/TSAT timing board.
- Leap year (day 1 through 365, or day 1 through 366) is a direct factor of the timing board.
- The most likely cause ofa leap year issue while using the Clock daemon is due to not setting the current year value in the timing board after each time the board boots-up. If the current year isn't set, whether or not it's a leap year is based on the default power-up year instead of being based on the current year.

Email Keith sent to the customer in Salesforce case 20592 (28 Jan 16) To begin, then Clock daemon program has no bearing on leap year. This program just reads the date/time from the installed timing board and sets the Windows Time to this received times stamp. Day of year, (1 through 365 or 1 through 366 depending on whether or not it is a leap year) is set in the timing board- not the clock daemon. This is based on the current year value that is set in the timing board after each power board. If the year is not set after each boot-up, whether or not it's a leap year as far as the timing board is concerned is based on its default boot-up year (and incremented each New Year from there).

We highly suspect the issue you are reporting is due to the present year not being set in each Timing boards each time they boot-up (the system each is installed in is power cycled/cold restarted, not just warm rebooted as the timing board will remain powered-up in this particular condition). The TPRO and TSync boards are a little different in this respect, but both require the default year to be corrected to the present year each time the board boots-up. Otherwise, they operate with a default year value after each boot-up. The Tsync boards default to the year value of "2000" after each boot-up. If I recall correctly, TPRO-PCI boards default to the year value of "0000" after each boot-up.

Neither the TPRO-PCI series nor the TSync series timing boards persist the current year after each boot-up. They boot-up using a default year. So unlike when using GPS satellites to sync Timing boards (TSAT-PCI or TSync boards with a GPS receiver only. TPRO-PCI boards can't sync to GPS) which sets the year automatically, when using IRIG input to sync the boards:

1) The **TPRO-PCI series** boards require the present year to be set using an API call/example program (or manually set via the Windows Control Utility) each time the boards are rebooted.

2) Unlike TPRO boards, the **TSync series** timing boards can be programmed to read the optionally provided year info in the IRIG data steam, if its reported (IRIG standards do not require year info to be provided in the IRIG data stream - whether or not it's provided is up to the manufacturer of the IRIG source and is also based on the formatting of the IRIG signal).

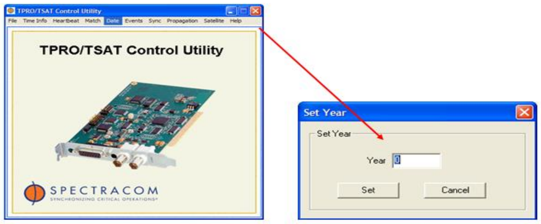
If the current year value is present in the data stream, it can be used to set the current year in the TSync timing boards automatically after each boot-up. But if the IRIG source doesn't provide the year info, or if the TSync board isn't programmed after each bot-up to read the year from the IRIG source, the TSync series timing boards also require the current year to be set after each boot-up (like the TPRO boards always require) or the TSync will also operate with the "current" year being the default year value after each power-up. Similar to the TPRO-PCI timing boards, the current year can be set using an API call or manually via the Windows Control Utility.

With both Models (always with the TPRO boards and when using IRIG input instead of GPS with the Tsync boards) if the year is not set properly after each boot-up, the timing board can't properly determine whether or not the current year is a leap year. This determination is based on its year value, whether it's been set to the current year after each boot-up, or if the year remains the default year value in the board after each boot-up.

To automatically set the year in both Models of timing boards not being synced with GPS, customers often create a custom start-up script and add the appropriate API call for setting the year to this script. Here are the two associated API calls for setting the current year in the Timing boards after each time the boards are rebooted.

The API call used to set the year in the TPRO series timing board is **TPRO_SetYear xxxx.** This same API call can also be used with the TSync boards when using its TPRO API calls (the TSync boards support the same API call set as the legacy TPRO timing boards to ease the replacement of a TPRO board with a TSync board). The API call dedicated to the TSync boards when not updating from a TPRO timing board is **CS_SetYear 0 xxxx.**

With both TPRO and TSync boards, in addition to setting the year with an API call after each boot-up, the year can also be manually set (or the current year value in the boards can be obtained) using the Windows Control utility that is installed as part of the Windows driver (as shown below, its set using the **Date** -> **Set Year** menu. The current year set in the board is obtained using the **Date** -> **Retrieve Gregorian Date** menu).



Year 0

.

B) Windows 64 bit

Note: The clock daemon in earlier versions of the PCI Windows drivers may have issues with 64 bit Windows (refer to Ken Moran from NOAA in Salesforce- email below was from him, also). We recommend updating the PCI driver to the latest version **(with Windows PCI driver v2.3.0, the TSync-PCIe driver must also be installed before the PCI driver. Uninstall PCI driver, install PCIe driver, and then re-install the PCI driver)**

C) Admin rights/ user rights

I'm fairly certain the user needs to be logged in with administrative rights in order for Windows to allow the Windows clock to be updated (this is a requirement of Windows- not the driver).

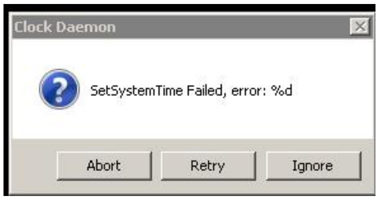
D) Configuration window not displayed

Q. We loaded version 2.3 from your website, then removed the old version. When version 2.3 is installed there is now no-way to get the old daemon window to become active (this old window allowed us to change interval synchronization time.) We tried using the daemon .exe file and never see the old style window. We expected the old window would still be available. Presently to change the sync interval we go into the control folder and edit the daemon.ini file (which contains the start-up values for the time sync = 300s) and change it to 10s. We then need to boot the PC. We verified that the sync update is now 10s.

So our question is what happened to the old sync window that allowed us to change the rate. Is it possible we are doing something wrong?

A **Answer (10/26/12):** When using the Clock daemon program in Windows PCI driver v2.3.0, the TSync-PCI driver must also be installed BEFORE installing the PCI driver. Otherwise, the clock daemon (and the Control Utility won't operate correctly, as described in the email above).

E) "Unable to set time!!" ("SetSystemTime Failed, error:! %d" displayed)


Clock Daemon

SetSystemTime Failed, error: %d

(now modified to include the fourth potential issue) Email from Keith (2 May 13) four key pointers/requirements for you, when using either of the Windows clock daemons:

2) Make sure the TSync board is in sync to IRIG or GPS input, or to itself. The green LED on the edge of the card needs to be lit in order for the timing board to be used as a reference (Sync status can also be verified in the Control Utility using the Sync -> Status menu. If this box is not selected or if the green LED on the edge of the board is not lit, let me know if its connected to a GPS antenna that is installed outdoors with a good view of the sky, or to an IRIG generator, for its input synchronization. Then we can go from there!

3) When using IRIG input for its sync (instead of GPS), the year needs to be set in the TSync-PCIe board each time it powers-up. With GPS input, the current year is automatically set after each boot-up. With IRIG input, if the IRIG generator provides the current year, the TSync-PCIe board needs to be programmed after each boot-up to know where to look for the year in the IRIG time code message.

   Until its programmed to look for the year, or if the IRIG generator doesn't provide it, a user can manually enter the year using the Windows Control Utility (Date -> Set Year menu) or with a API call/example program using the driver. To verify the year is set correctly after each boot-up, use the Date -> Retrieve Gregorian date menu in the Control Utility).

4) The Windows PC must be able to successfully talk to the board. The Control Utility can be used to verify this requirement is being met. If you successfully opened the **Date** -> **Retrieve Gregorian** date menu mentioned in the previous step, this requirement has been verified.

5) Make sure the **c:/Programs Files** folder only contains a "**Spectracom**" folder, and not also an "**Orolia**" folder, as well. Having both folders present indicates an older version of the Windows driver is co-installed and conflicting with the newer driver,

DETAILS
When installing the 3.2.3 TSync Driver onto a machine with an existing 3.2.1 TSync driver we noticed this issue.

The Clock Daemon service does not get removed or overwritten during the install and the 3.2.1 version of the driver used C:/Program Files/Spectracom. In 3.2.3 we changed the installation directory to **C:/Program Files/Orolia.**

To correctly address this issue we would recommend uninstalling the software from Add or Remove programs, uninstalling the driver, and then running the following command to delete the old Clock Daemon Service "**sc delete CLOCKDAEMON**" (per Keith, make sure Clock Daemon stopped in Services).

Then when installing the 3.2.3 driver everything will be pointing to the correct location.

Instead of deleting older driver:

1) Rename the older folder in **C:/Program Files** from “**Spectracom** to “**OLD Spectracom**”
2) Rename the newer folder from “**Orolia’** to “**Spectracom**”
3) Restart the Clock Daemon Service (**Start** > **Computer Management**. Select **Services** and scroll down the list of services to **Clock Service**

Please also note the Windows driver is only compatible with Windows 7, 32 bit. It is not compatible with 64 bit Windows. At this time, there are no plans to make this driver compatible with 64 bit Windows

N/A for PCI boards - applies to PMC, cPCI and PC104) With Windows 7, 2008, 8, etc, make sure it’s not 64 bit Windows (driver is only compatible with Windows 32 bit)

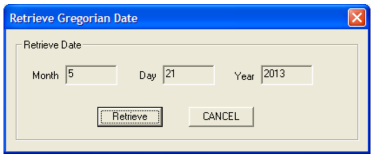
F) Set system time failed error 87" (Year needs to be set to the current year in the timing board)

Set system time failed error 87"- Seems to be caused by using the Clock daemon utility in the Windows driver when using IRIG input to a board without having the year set to the correct value. Verify the year value is set correctly in the Windows Control Utility. (See Emmett McCabe record in Customer Service database- around 2/5/08). Saw this on TPRO-PMC card with Windows driver.

Email from Keith (30 Apr 13) Also when using IRIG input (instead of GPS input) to sync the TSync board, the timing board needs to either be configured to read the current year value from the IRIG generator (if it provides year information) or it needs to be set in the TSync-PCIe board each time it boots-up. This can be accomplished with an API call/example program or via the Date-> Set Year menu in Control Utility (note this menu only sets the year-it does not read the year. The **Date**-> **Retrieve Gregorian** Date menu will read the current year set in the TSync-PCIe board. The default year at boot-up is 2000, if the year is not set by IRIG input. GPS input does automatically set the year when the TSync-PCIe boards is synced by GPS instead of IRIG input.

G) Windows must be able to communicate with the timing board

To verify the computer can communicate/access the board (and to also verify the year is set to 2013), open the Windows Control Utility (Start /All Programs/Spectracom Corp) and perform a **Date** -> **Retrieve Gregorian Date.** The Control Utility should respond with the date and correct year (example below)

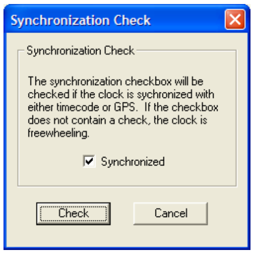


Month 5 Day 21 Year 2013

H) Timing board needs to be synced

Timing board is not synced: The timing Board must be synced in order for it to be used to sync the clock utility. With IRIG input, the year must be manually set (not applicable with GPS input) before the timing board will sync to the IRIG data. Refer to IRIG input synchronization section above.

If the date is correct, the system can access the timing board. Now check sync state. In the Control Utility, click on **Sync**--> **Status.** In the Synchronization Check window that opens, make sure the "**Synchronized**" box is selected (as shown below)



Synchronization Check

Synchronization Check

The synchronization checkbox will be checked if the clock is sychronized with either timecode or GPS. If the checkbox does not contain a check, the clock is freewheeling.

☑ Synchronized

TSync “QuickPTP Viewer / “TSync Viewer” GUI

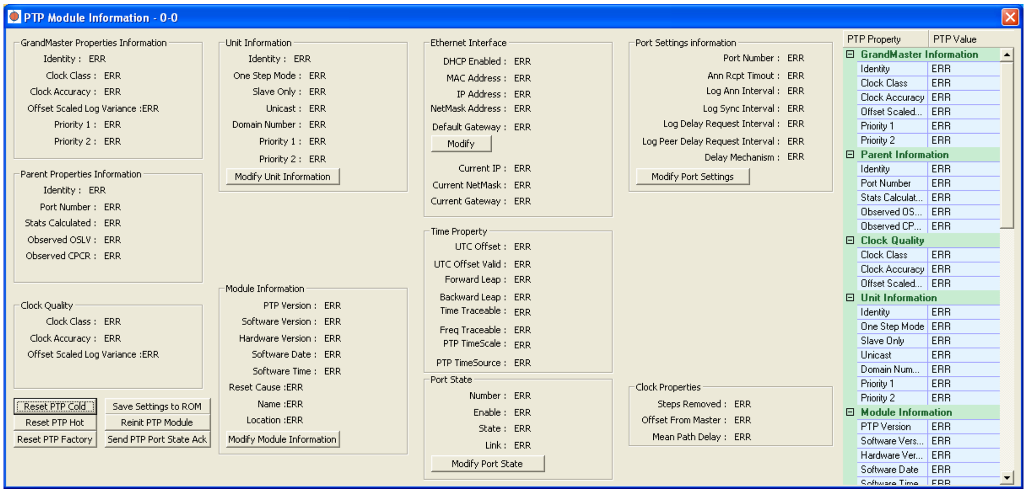
When the Windows driver is installed, it also installs the “Clock Utility” (basic GUI interface), QuickPTP (PTP module interface) and the Clock Daemon program/service for syncing the PC to the TSync board

(Free PTP application for TSync-PCIe-PTP boards and SecureSync-PTP)

- A graphical interface (GUI) installed as part of the Windows driver for the TSync-PCIe-PTP boards.
- Version 2.3.0 of the TSync-PCIe Windows driver adds a free GUI interface (called “QuickPTP”). This GUI is automatically installed in the Windows PC when the TSync Windows driver is installed.
- Time of the PTP module is displayed in HH:MM:SS.
- To access it: Start/All Programs/Spectracom Corp/TSync PTP/QuickPTP (a demo mode is not available; a TSync-PTP board needs to be installed in order to open it).
- Two screens are displayed. One is for the TSync-PCIe board (shown below on top) and the other is for the PTP module (shown below on the bottom).

QuickPTP interface

- **To open the QuickPTP interface:** Start / All Programs / Spectracom Corp / TSync PTP / QuickPTP
- Provides a basic interface with the TSync-PTP board (configure the IP address of the PTP module, for instance)
- The TSync-PCIe board DOES need to be installed to use this GUI. Install the Windows driver and this interface can be used, as long as a TSync board is installed (Unlike the Control Utility, QuickPTP does not have a “demo” mode available).



GrandMaster Properties Information
Identity : ERR
Clock Class : ERR
Clock Accuracy : ERR
Offset Scaled Log Variance :ERR
Priority 1 : ERR
Priority 2 : ERR

Parent Properties Information
Identity : ERR
Port Number : ERR
Stats Calculated : ERR
Observed OSLV : ERR
Observed CPCR : ERR

Clock Quality
Clock Class : ERR
Clock Accuracy : ERR
Offset Scaled Log Variance :ERR

Unit Information
Identity : ERR
One Step Mode : ERR
Slave Only : ERR
Unicast : ERR
Domain Number : ERR
Priority 1 : ERR
Priority 2 : ERR

Module Information
PTP Version : ERR
Software Version : ERR
Hardware Version : ERR
Software Date : ERR
Software Time : ERR
Reset Cause :ERR
Name :ERR
Location :ERR

Ethernet Interface
DHCP Enabled : ERR
MAC Address : ERR
IP Address : ERR
NetMask Address : ERR
Default Gateway : ERR
Current IP : ERR
Current NetMask : ERR
Current Gateway : ERR

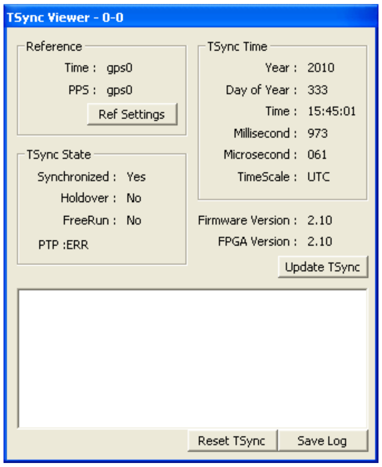
Time Property
UTC Offset : ERR
UTC Offset Valid : ERR
Forward Leap : ERR
Backward Leap : ERR
Time Traceable : ERR
Freq Traceable : ERR
PTP TimeScale : ERR
PTP TimeSource : ERR

Port State
Number : ERR
Enable : ERR
State : ERR
Link : ERR

Port Settings information
Port Number : ERR
Ann Rcpt Timout : ERR
Log Ann Interval : ERR
Log Sync Interval : ERR
Log Delay Request Interval : ERR
Log Peer Delay Request Interval : ERR
Delay Mechanism : ERR

Clock Properties
Steps Removed : ERR
Offset From Master : ERR
Mean Path Delay : ERR

PTP Property	PTP Value
⊟ GrandMaster Information	
Identity	ERR
Clock Class	ERR
Clock Accuracy	ERR
Offset Scaled...	ERR
Priority 1	ERR
Priority 2	ERR
⊟ Parent Information	
Identity	ERR
Port Number	ERR
Stats Calculat...	ERR
Observed OS...	ERR
Observed CP...	ERR
⊟ Clock Quality	
Clock Class	ERR
Clock Accuracy	ERR
Offset Scaled...	ERR
⊟ Unit Information	
Identity	ERR
One Step Mode	ERR
Slave Only	ERR
Unicast	ERR
Domain Num...	ERR
Priority 1	ERR
Priority 2	ERR
⊟ Module Information	
PTP Version	ERR
Software Vers...	ERR
Hardware Ver...	ERR
Software Date	ERR
Software Time	ERR



Reference
Time : gps0
PPS : gps0

TSync Time
Year : 2010
Day of Year : 333
Time : 15:45:01
Millisecond : 973
Microsecond : 061
TimeScale : UTC

TSync State
Synchronized : Yes
Holdover : No
FreeRun : No
PTP :ERR

Firmware Version : 2.10
FPGA Version : 2.10

"Clock Properties" section
These values will report "**ERR**" (Error) when this utility is used with the TSync-PCIe boards. (they were planned to be used withg SecureSync-PTP, but this has since changed. Quick-PTP no longer compatible with SecureSync.

The next release of TSync-PCI PTP firmware update should correct this.

Email regarding "Clock Properties"
For your information, this available PTP tool was initially intended to be used with two types of Spectracom equipment (both the TSync-PCIe bus-level timing boards and SecureSync products when an available PTP Option Card is installed)

The "Clock Properties" section of the tool that is showing the "ERR" is applicable to the SecureSync products and so all three values will return "ER" when used with the TSync-PCIe timing boards. This is not an indication of a problem with the operation of your Spectracom equipment.

Notes about QuickPTP

- This utility is owned by France, not us.
- The time displayed in the GUI is the time of the TSync-PCIe board (not the time from the PTP module).
- No literature for this program currently exists.

QuickPTP interface

To open the QuickPTP interface: Start / All Programs / Spectracom Corp / TSync PTP / QuickPTP
Provides a basic interface with the TSync-PTP board (configure the IP address of the PTP module, for instance)
The TSync-PCIe board DOES need to be installed to use this GUI. Install the Windows driver and this interface can be used, as long as a TSync board is installed (Unlike the Control Utility, QuickPTP does not have a "demo" mode available).

Reported PTP module firmware version conversion information:

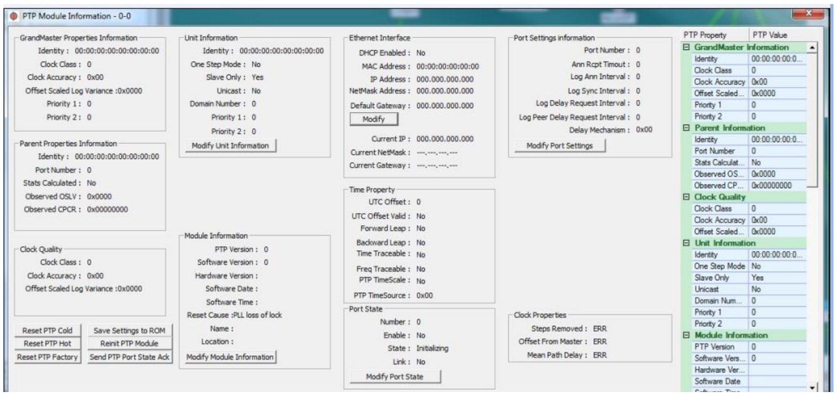
The PTP firmware version is reported in the "**Module Information**" section of the QuickPTP program ("**Software Version**" field). This field reports a value that corresponds to the installed firmware version:

Reported value	Our version number
A11 (11)	1.0.0
A19 (19)	1.1.0
A20 (20)	1.1.1

Known issues with QuickPTP

- The “Clock Quality” value is not currently supported, but is planned to be fixed in a new TSync-PCIe release to be made available sometime after the SecureSync PTP module has been released. Currently, this value is displayed as “ERR” to indicate an error occurred while trying to read this value.
- As of at least 2/15/11 (and prior), the “**Current NetMask”** and “**Current Gateway**” fields are two enhanced feature fields that have been already been added to the QuickPTP GUI in anticipation of newer reporting capabilities with the integration of the next anticipated TSync-PTP firmware upgrade. These fields being blank, at this time, does not indicate that there is a problem with the PTP port. However, you should be able to ping the port and receive a response from it.
- As of at least 2/15/11 (and prior), “Clock Properties” fields (“**Steps Removed**”, “**Offset from Master**” and “**Mean Path Delay**”) show “**ERR**” (Error). This is also supposed to be fixed in the next version of the TSync-PCIe firmware.

PTP module shows Information displays all “0”’s after cold reset (example screenshot below



PTP Module Information - 0-0

GrandMaster Properties Information
Identity : 00:00:00:00:00:00:00:00
Clock Class : 0
Clock Accuracy : 0x00
Offset Scaled Log Variance :0x0000
Priority 1 : 0
Priority 2 : 0

Parent Properties Information
Identity : 00:00:00:00:00:00:00:00
Port Number : 0
Stats Calculated : No
Observed OSLV : 0x0000
Observed CPCR : 0x00000000

Clock Quality
Clock Class : 0
Clock Accuracy : 0x00
Offset Scaled Log Variance :0x0000

Unit Information
Identity : 00:00:00:00:00:00:00:00
One Step Mode : No
Slave Only : Yes
Unicast : No
Domain Number : 0
Priority 1 : 0
Priority 2 : 0

Module Information
PTP Version : 0
Software Version : 0
Hardware Version :
Software Date :
Software Time :
Reset Cause :PLL loss of lock
Name :
Location :

Ethernet Interface
DHCP Enabled : No
MAC Address : 00:00:00:00:00:00
IP Address : 000.000.000.000
NetMask Address : 000.000.000.000
Default Gateway : 000.000.000.000
Current IP : 000.000.000.000
Current NetMask : ---.---.---.---
Current Gateway : ---.---.---.---

Time Property
UTC Offset : 0
UTC Offset Valid : No
Forward Leap : No
Backward Leap : No
Time Traceable : No
Freq Traceable : No
PTP TimeScale : No
PTP TimeSource : 0x00

Port State
Number : 0
Enable : No
State : Initializing
Link : No

Port Settings information
Port Number : 0
Ann Rcpt Timout : 0
Log Ann Interval : 0
Log Sync Interval : 0
Log Delay Request Interval : 0
Log Peer Delay Request Interval : 0
Delay Mechanism : 0x00

Clock Properties
Steps Removed : ERR
Offset From Master : ERR
Mean Path Delay : ERR

PTP Property	PTP Value
⊟ GrandMaster Information	
Identity	00:00:00:00:0...
Clock Class	0
Clock Accuracy	0x00
Offset Scaled...	0x0000
Priority 1	0
Priority 2	0
⊟ Parent Information	
Identity	00:00:00:00:0...
Port Number	0
Stats Calculat...	No
Observed OS...	0x0000
Observed CP...	0x00000000
⊟ Clock Quality	
Clock Class	0
Clock Accuracy	0x00
Offset Scaled...	0x0000
⊟ Unit Information	
Identity	00:00:00:00:0...
One Step Mode	No
Slave Only	Yes
Unicast	No
Domain Num...	0
Priority 1	0
Priority 2	0
⊟ Module Information	
PTP Version	0
Software Vers...	0
Hardware Ver...	
Software Date	

Other issues

- Machine fails if QuickPTP runs more than 90 minutes
- Refer to Salesforce case 11262 https://na8.salesforce.com/500C000000TDSs9?srPos=0&srKp=500)

I am able to consistently get the machine to fail while running the QuickPTP application for more than 90 minutes. I have now setup a script to continuously call the HW_GetTime example. IF this can run for multiple hours without issue, then I will have what I will need for my current needs. Do you expect that the QuickPTP app should be able to be run for extended periods? Do you believe we may have a faulty card? What is the RMA process if we need to swap this for a new card?

Repy from Dave Lorah (15 Aug 2013) I was discussing this with one of our engineers this morning. We are not 100% sure the QuickPTP software is not at fault. This was basically a development test tool that we decided to include with the drivers. It is not supported for bug fixes like the drivers and card firmware, so there indeed may be some issues with newer operating systems like win 2008 R2 we are not aware of. I suggest running the card without the QuickPTP running and use just the HW_GetTime calls. If there are no further failures I would say the card is good.

If there are more failures when Quick PTP is not running we should suspect the hardware

Issues running on Windows 2012

Email from Richard Hochron (15 Aug 2013) Can you confirm there is support for this card for Windows 2012 Server.

I have noticed that while running the QuickPTP app that the server becomes unstable and blue screens. I have been running the app under 2008R2 for a while and it does seem more stable.

Reply from,Dave Lorah- As far as I know, this has never been tested on a Win 2012 server. I do know it works on Win 2008 and Windows 7.

Denis Reilly commented QuickPTP is that application that Hervé wrote that we included on the driver disc for customers. It hadn't been updated since we first released it. I wouldn't be surprised if it is now out of date with the latest Windows stuff. We should decide what to do with it.

FAQS about QuickPTP

Q. (From Chuck Beck of XRtrading 2/11/11) I think I had it working but was unsure, so I click the Reset PTP Cold button and after that I was unable to set anything on the card. The GUI only shows zeros on all the fields. See attached JPG.
A. (Email from Denis- don't send this as-is to customers)
When the GUI shows all '0''s, it means that the module is not communicating with the TSync.
When the PTP module on the TSync-PTP gets cold reset, the module itself can take a minute or so initialize. (Particularly if DHCP is enabled). During initialization, the module does not communicate, and the TSync deletes all the prior PTP state information, so you will get stuck at all '0''s until communication comes back. In rare circumstances, if there is a lot of non-PTP network traffic when the card is reset, the module could be spending so much of its time filtering that that it doesn't have time to communicate properly. I saw this in development, and I really hoped they fixed it before shipping! :)

I would recommend a cold restart of the system the TSync card is in with the network cable unplugged. Communication should come back by the time the system boots. If that doesn't help, I'll have to think about it some more.

Questions from Sales about PTP:

- Regarding boundary clocks – is there general information here or do we need to understand the customer's application?
- I'm sure a customer can support NTP and PTP both from the SecureSync, but if there are any specific comments related to this, please let me know.
- Any switch that accepts PTP is compliant with Spectracom products, correct? However. I do know that the configuration of the network itself does impact accuracy, so not sure we can easily answer the "what is the accuracy" question. Please add comments here.

Not sure about this PTP with Linux Red Hat or if we support at chip level. Comments?
Email response from Denis Reilly (10/18/11)
I'll start with just some general PTP information. I apologize if it's too basic, but you say you are "coming up to speed". Let me know if I forgot to answer anything. ;)

1st basic point: on SecureSync, NTP comes out of the main network port (and the additional Gigabit network ports). PTP only comes out of the network port on the PTP card. There is no problem running NTP and PTP over the same network, but the customer will have to take care of configuring the NTP and PTP interfaces properly for the network.

Even though PTP is a master/slave protocol, PTP communication is bi-directional. PTP makes a fundamental assumption that the delays on a network are symmetrical – that the delay in one direction is the same as in the other direction. PTP also works best when the delay does not change much with time. Modern switching equipment has all manner of queues and delay sources that can foul this up – if packets are queued in one direction and not the other, this will affect timing performance.

To help compensate for this, there are two special types of PTP equipment defined in the standard. One type or the other is implemented in "PTP Enabled" switches:

Regarding Linux, our SecureSync product and our TSync product implement the entire PTP stack in "hardware". (Really a microprocessor, but close enough to hardware). If a customer wants to run a SecureSync PTP server and run PTP clients in Linux, they have several software-based alternatives. The best is to buy
er from us. TimeKeeper will synchronize the Linux system clock with PTP. It's expensive, but is much more accurate than any other software solution: we are selling it into that financial market where they want their Linux kernel synchronized to under 1 us. Timekeeper will work with a normal NIC card, but will work better with a PTP-enabled NIC card with "Hardware PTP timestamping", and will work best with a TSync PTP card (although that is very expensive – ask Will about that!)

****Spectracom batch file to capture responses from a specified API call**

- KW created an API call batch file in Dec 2012.
- Link to batch file: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\API Call batch files.

(12-03-12 KW email to Evan Dietz) Attached you should find the zipped batch file. The file extension has been change to .aok in order to send it to you. Just rename the file to .zip and unzip it to a directory on the hard drive (such as C:\temp for example) of the Windows PC which has the TSync-PCIe board installed. Then, run this batch file. The batch file will walk you through the process. I

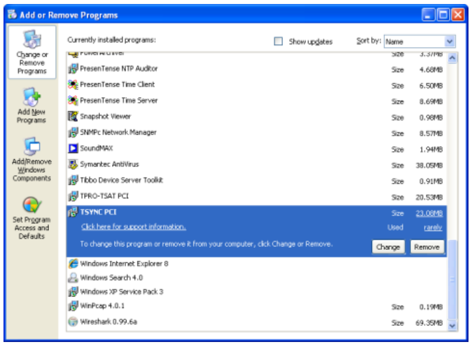
This batch file will capture the results of the desired API call (such as HW_GetPhaseError for your desired test) to a text file in c:\temp. Each time the API call is sent, a hardware time stamp from the TSync board is sent to another file, also in c:\temp. If desired, you can correlate the two text files to know when each API call was issued and the TSync board responded.

The batch file is looped, so it will continue to capture the results of the API call each time a key is pressed. When you want to complete the capture, simply “X” out of the batch. Then, if desired, you can either convert the text file to .xls or copy/paste the responses in the tsync.txt file to an excel spreadsheet. Then, you can also easily create a line or bar graph for the responses!

Uninstalling the current version of the Windows TSync-PCIe driver:

To begin, the older Windows driver should be removed before installing the latest version of the driver. The original version can be installed using the standard Windows uninstall program (depending on the version of Windows, such as Control Panel/ Add or Remove programs, for example). When clicking on this Windows program, “TSync PCI” driver will be listed as one of the installed programs that can be removed from the PC. Click on the “Remove” button.

Below is an example of using the windows Add or Remove Programs to uninstall the original driver.


Add or Remove Programs
Change or Remove Programs
Add New Programs
Add/Remove Windows Components
Set Program Access and Defaults
Change
Remove

Currently installed programs: □ Show updates Sort by: Name

PowerArchiver	Size	3.37MB
PresenTense NTP Auditor	Size	4.68MB
PresenTense Time Client	Size	6.50MB
PresenTense Time Server	Size	8.69MB
Snapshot Viewer	Size	0.98MB
SNMPc Network Manager	Size	8.57MB
SoundMAX	Size	1.94MB
Symantec AntiVirus	Size	38.05MB
Tibbo Device Server Toolkit	Size	0.91MB
TPRO-TSAT PCI	Size	20.53MB
TSYNC PCI	Size	23.08MB
Click here for support information.	Used	rarely
To change this program or remove it from your computer, click Change or Remove.		
Windows Internet Explorer 8		
Windows Search 4.0		
Windows XP Service Pack 3		
WinPcap 4.0.1	Size	0.19MB
Wireshark 0.99.6a	Size	69.35MB

Installing a new Windows driver

Per the install instructions, we recommend the customer reboots the PC after installing driver. The driver reports it has been installed correctly after install, but may not be able to talk to the board until the PC has been rebooted.

Email from Tim T: In the install instructions, we recommend that you reboot the computer after installing the driver so that the driver gets loaded properly. This is all part of Windows Plug & Pray. This is not an install failure.

The driver can be manually installed through the control panel. I think it was Dave L. I was trying to explain this to earlier this week. When I get some time I will look into the driver install to see if Windows can be forced to scan for new hardware.

Windows driver example programs:

There is an example program that can be used to obtain the current DAC value. Open Windows Explorer and navigate to the following directory: **C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API**. You will see all of the available example programs that can be run.

To run an example program:

Open a command prompt window (Search -> type “cmd” and navigate to the “**C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API**” directory. Then, run the desired example program (such as XO_ GetDac.exe to get the current DAC value, for example). The exe extension is not typed as part of the command.

Per Ron Dries (25 Feb 2020) The example programs need to be run from the command line, so he will need to run “cmd” then navigate to where the .exe example programs are and issue the commands with the parameters they are expecting.

Start->run cmd

Cd C:\..\Where\exmaple\programs\are located

Type: GR_GetSatData.exe 0 0 <enter>

Cygwin

(12/17/10 kw) Cygwin is a Linux shell that runs in a Windows environment (like DOS running in Windows). Arnaud from France has a customer that desires to compile and perform API calls right in Cygwin. Based on info from Denis and Tim T, I emailed him the following information:

> **Official response from Tim T about this desired operation:**
> I downloaded and installed Cygwin last night on my computer to test our driver. Our driver will not compile as is.
> This is not a true Linux environment. This is a Linux shell.
> Our driver compiles differently depending on what kernel is installed. When you prompt Cygwin to report what kernel is running, it reports its own revision, not the kernel.
> If this customer has experience with Cygwin, they might be able to setup the environment with the correct source files. They will also need to modify our driver to compile.
>
> Since Cygwin is already running on Windows, I would recommend that they use our Windows driver.

My response to this customer:
I have been speaking to our Engineers regarding your customer’s desire to compile and run API calls from Cygwin.   We have just tried our Linux driver in this environment and found that the driver is not directly compatible. It will not compile, as it is (this is because Cygwin is only a Linux shell and not a true Linux environment). In order for our driver to be able to compile, it needs to know the version of the linux kernel that is installed. When Cygwin is prompted for its linux kernel version, it reports its own version instead of the kernel version, preventing the driver from being able to compile.

If your customer has experience with Cygwin and desires to operate the TSync-PCIe board in Cygwin, they may able to setup Cygwin with the correct source files. They are certainly welcome to use the source code that we provide to try and create their own driver from the source code.

If possible, since Cygwin is already running on Windows, we recommend they use our Windows driver, instead. Otherwise, they will have to create to their own driver from our provided source code.   We have no experience with this desired operation and so we won’t be able to provide them any additional assistance with this desired task.

Windows with UEFI Secure Boot (Trusted Boot)/digital Signature requirements

- Refer to salesforce case 25460

Secure *Boot* is a technology where the system firmware checks that the system boot loader is signed with a cryptographic key authorized by a database contained in the firmware. With adequate signature verification in the next-stage boot loader(s), kernel, and, potentially, user space, it is possible to prevent the execution of unsigned code. *Secure Boot* is a form of *Verified Booting*. Boot path validation is also part of other technologies such as *Trusted Boot*. Boot path validation is indepedent of secure storage of cryptographic keys and remote attestation.

UEFI=Unified Extensible Firmware Interface (refer to: https://en.wikipedia.org/wiki/Unified_Extensible_Firmware_Interface)

Q (TSync-PCIe) Is this compatible with Windows 10 with secure boot?
Answer: (Dave L, 26 May 17) In order to be compatible with Windows 10 Secure Boot we will need to update the TSync Windows Driver with the additional driver signing that is required. Spectracom is working on this with Microsoft and will have the feature in a new driver release. I do not have a firm date as to when this will be completed though.

.net Frame work for TSync-PCIe

Q I am trying to synchronize my application with a TTL pulse sent from a Securesync using a TsyncPCI card. I am using Visual Studio as IDE and would to know if there is dotnet framework available to access the Tsync card?

Answer: There is no dot net application for the TSync-PCIe boards.

Visual Studio (VS)

- Used for developing application software on Windows PCs.
- There are several versions of Visual Studio, including Visual Studio 2003, Visual Studio 2005, Visual Studio 2008 Visual Studio 2010, Visual Studio 2012, Visual Studio 2013 Visual Studio 2014 and Visual Studio 2015.

Visual Studio Solution Files (*.Sln files)

Per: https://www.bing.com/search?pc=COS2&ptag=D081717-N0610AF725A490EB72436481F&conlogo=CT3332018&q=visual+studio+solution+file&form=CONBDF

In Visual Studio parlance, a **solution is a set of code files and other resources that are used to build an application.** The files in a solution are arranged hierarchically, which might or might not reflect the organization in the file system. The Solution Explorer is used to manage and browse the files in a solution.

A) Visual Studio 2017 (VS2017)

- Refer to Case 193331 (June 2019 with Windows driver v3.2.1 beig the latest release)
- Windows driver v3.2.1 states support for VS2015- not VS2017

Q We are having a problem now when I integrate the TSYNC calls (and library) into our current c++ code.

I am getting an “unresolved external symbol __except_handler4_common”

We think the problem may be because we are using Visual Studio 2017 and the TSYNC library that we have was built using Visual Studio 2015.

Is there a newer version of the TSYNC library built using a later version of Visual Studio that I can try ??

Also I am getting a linker warning regarding EDITANDCONTINUE vs OPT:ICF having to do with Tsync.lib(tsync_error_codes.obj.

Is there any way we can get a copy of the source code and try to build the Tsync.lib ourselves ??

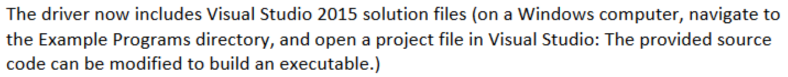
B) Visual Studio 2015 (VS2015)

Visual Studio 2015 Solutions file

Q The version of the driver that I have does not have a solution file for VS 2015. Can you provide the link on your web site where I can download this updated driver? Otherwise can you send the VS 15 solution file that is setup to utilize the correct lib and dll files?

A Email from Dave L (20 Jan 17) I have verified the current Windows driver version 3.2.1 has the capability to support Visual Studio 2015.

From the driver release notes:



The driver now includes Visual Studio 2015 solution files (on a Windows computer, navigate to the Example Programs directory, and open a project file in Visual Studio: The provided source code can be modified to build an executable.)

Q Can you please tell me how to load the DLL and LIB files into Visual Studio. Is there a way to use the library files that is supplied with the card? I am using Visual Studio 15. My application requires the card to provide the time for each rising edge of a 100Hz TTL pulse train.

A **Email from Dave L (9 May 17)** I have verified the current driver version 3.2.1 has the capability to support Visual Studio 2015.
The driver now includes Visual Studio 2015 solution files (on a Windows computer, navigate to the Example Programs directory, and open a project file in Visual Studio: The provided sourcecode can be modified to build an executable).
You can always find the latest software and firmware for your Spectracom products on our website Spectracom.com. https://spectracom.com/support/tsync/tsync-pcie-support Here is a link to get the version 3.2.1 64 Bit Windows Driver: https://spectracom.citrixdata.com/share?#/view/s7ffbe3e67e84dc39

C) Visual Studio 2012 (VS2012)

- Refer to Mantis case 2936 regarding email below (about current Windows driver version 3.12)

From: Mihai Ghita with Norpix [mg@norpix.com]
October 14, 2014 10:45 AM

We noticed that the TPro.lib and TSync.lib files are linked specifically with a certain version of the VC files (manifest dependency Microsoft.VC80.CRT' version='8.0.50727.6195').
Because of that, our interface built on the Spectracom SDK cannot load, because it cannot find those particular dependency files. We tried installing that particular version of the VC redistributables and also other versions, but it didn't help. This happens on a Windows 7, 64 bit.

Would it be possible to have this library built without a specific VC dependency? Our application is linked with Visual C++ 2012 and we install the VC++ 2012 redistributables packages. If the Spectracom SDK libraries are not linked with a specific VC dependency, it should work with the VC++ 2012 redistributables packages that our application installs. There would be no need for an extra VC++ redistributables package.

Solution: Per Tim Tetreault, this desired change will require a new version of the Windows driver be released (post version 3.1.2).

D) Visual Studio 2010 (VS2010)

Email from Tim Tetreault (30 Oct 2013) I know we have other customers that are using VS2010 but I don't know for sure if they are using it in the what your customer intends on using it. The only way for the customer to know for sure if their applications will work with our board is to give them a demo board to try.

Resource wise we are thin right now so it's hard for me to find the time to do a test setup. And if we did, it's still no guarantee that the way we test it is the way the customer intends on testing it. Is the customer open to demo the board so he can try it out?

Q. Is there a Visual Studio 2010 C++ static library .lib available for your TSync PCIe card? When I go to your website it appears that there is only a Visual Studio 2005 version available under the latest SDK 2.41.
A. Email from Tim Tetreault (9/30/11) We only have a Visual Studio 2005 C++ static library.
~~We currently have no plans in the near future to update this.~~

Update (11 Apr 16 KW) Per Tim T, we are working on a Windows driver update to be released in a couple months to be built on newer version of Visual Studio.

Has the customer tried the driver to see if it might work or if it can be imported into Studio 2010? The driver is free from our web site.

E) Visual Studio 2008 (VS2008)

Microsoft Visual C++ Runtime Library Runtime Error! R6034

Email from Tom Patterson I'm using Microsoft Visual Studio 2008 on an HP Desktop computer running Windows XP (32 bit). I have your TSync Time Code Processor Card with GPS (I think its PCI-e).

The date of tsync.lib and tsync.dll that I'm using is 3/31/2014.

When I run the Tsync Control Utility the Help / About reports:
Driver Version: 3.1.2
FPGA Version: 0111
Firmware Version: 33-2E-30-34

Everything compiles using your .h files in the C:\Program Files\Spectracom\Dev directory. However when I run I get the following error:

> **Microsoft Visual C++ Runtime Library**
> **Runtime Error!**
> R6034
> An application has made an attempt to load the C runtime library incorrectly.

Reply from Morgan (19 Aug 2015)
From the information you sent everything appears to be good, so we looked at the error (R6034) itself. This error seems to indicated issues with the registry or the compiler, I have attached a few links for your review.
http://www.r6034runtimeerrorfix.com/
https://msdn.microsoft.com/en-us/library/ms235560%28v=vs.90%29.aspx

F) Visual Studio 2005 (VS2005)

Email to a customer based on info from Tim Tetreault (~Oct 2011)
Below is the information from Engineering regarding Visual Studio 2010 with the PCI boards:
We only have a Visual Studio 2005 C++ static library.

Have you tried the driver we have on our website (see link below) to see if it can be imported into Studio 2010? If not, can you give it a shot and let us know if it works for you? The driver is free from our web site. Once downloaded, you can attempt to install it, even without the timing board installed. If there is a compatibility issue with installing it, you will know this even without having to have a timing board on-hand.

The Timing board drivers can be downloaded from our website at:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=20

G) Visual Studio 2003 (VS2003)

Q. Do you know if the TSync-PCIe is compatible with visual studio 2003?
NOTE: Refer to the "**Update**" to the answer below. The answer is no. They need to upgrade to a newer version of Visual Studio

A. just spoke to one of our timing board engineers to confirm my reply. He said that the TSync-PCIe Windows driver is likely compatible with Visual Studio 2003. However, we haven't tested against this earlier version of Visual Studio (We use Visual Studio 2005, instead).

He brought up a good point that all of the TSync-PCIe drivers are readily available for free download from the Spectracom website. And, as the timing board does not need to actually be installed in the PC order to install the driver, they can pre-test this before they purchase and receive the TSync-PCIe timing boards.
To download any of the freely available TPRO/TSAT or TSync-PCIe timing board drivers, just have them visit us at: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=20. To obtain the

TSync-PCIe drivers, click on “PCIe”. The “Documents” folder contains the TSync-PCIe manual and driver guides, while the “software” folder contains individual folder for each of the available drivers.

Update (10/26/11 KW):
We have now had two customers (refer to Ken Liske with Rockwell Collins) getting warning and error messages when using Visual Studio 2003. The warning message is ignored but the error messages prevent the code from successfully compiling. The fix is to update to Visual 2005(or higher)

As of at least 2/3/11, the TSync-PCIe version 2.3.0 Windows driver appears to not be fully compatible with application programs built with Visual Studio 2003. There will likely be tpro.lib and get link errors present. Example below:

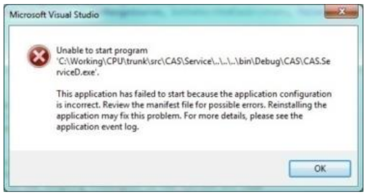
```
Compiling...
stdafx.cpp
Compiling...
IrigMgr.cpp
BtiKsiTpro_DLL.cpp
BtiKsiTPro.cpp
Generating Code...
Compiling resources...
Linking...
tpro.lib(tprodll.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
LINK : warning LNK4075: ignoring '/EDITANDCONTINUE' due to '/OPT:ICF' specification
tpro.lib(tsync_hw.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tpro_capabilities.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b''
encountered; ignored
tpro.lib(tsync_Gr.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tpro_date_time.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b''
encountered; ignored
tpro.lib(tsync_cs.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_ls.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_Ss.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_Go.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_pp.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_ip.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_ir.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_pr.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_rs.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_Generic.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b''
encountered; ignored
tpro.lib(tsync_base_transaction.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b''
encountered; ignored
tpro.lib(tsync_Gr_Services_recipes.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b''
encountered; ignored
tpro.lib(tsync_misc_Services_recipes.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b''
encountered; ignored
tpro.lib(tsync_cs_Services_recipes.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b''
```

```
encountered; ignored
tpro.lib(tsync_ls_Services_recipes.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
```

Ken Liske (with Rockwell Collins) also received the following using Visual Studio 2003:

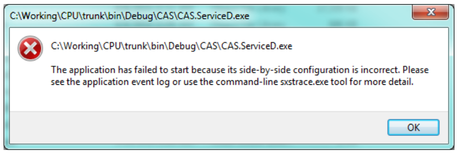
- Warning LNK 4229 “invalid directive‘/manifestdependency’ encountered; ignored" This is an ignored warning message, so doesn’t cause any issues
- **lnk2001 unresolved external symbol __imp___Snwprintf_S”** and “**fatal error LNK1120: 1 unresolved externals”** : these are both error messages that are preventing their application code from compiling. We recommended he update to Visual Studio 2005 or higher.

**CAS.Service and Side-by-Side errors (Side by Side Configuration)



Unable to start program 'C:\Working\CPU\trunk\src\CAS\Service\..\..\..\bin\Debug\CAS\CAS.ServiceD.exe'.

This application has failed to start because the application configuration is incorrect. Review the manifest file for possible errors. Reinstalling the application may fix this problem. For more details, please see the application event log.



C:\Working\CPU\trunk\bin\Debug\CAS\CAS.ServiceD.exe

The application has failed to start because its side-by-side configuration is incorrect. Please see the application event log or use the command-line sxstrace.exe tool for more detail.

- Refer to sites such as: http://www.codeproject.com/Articles/43681/Side-by-Side-Configuration-Incorrect
- Refer to Salesforce case 15464 (Carl Shirey).

Labview driver/Labview wrapper

- As of at least Jul 2017, the TSync-PCIe boards do not have an available Labview driver /Labview wrapper. There are no plans at this time to develop one
- With the shared library (xxx.so) file provided with our driver, a customer can create their own wrapper routines using this provided file. For additional info on calling .so files in Labview, refer to http://zone.ni.com/reference/en-XX/help/371361J-01/lvhowto/import_Shared_library/

Note: The website above discusses how to import functions from a shared Library file (.so file , as provided in the TSync-PCIe Windows driver) into Labview.

- The **libtsync.so** shared library file is generated during the driver build. Its located in the **TSync/Linux/Lib** directory.

For more information on wrappers, refer to websites such as:
http://digital.ni.com/public.nsf/allkb/06ECDC689DDA0F3D862574440074CD95

Problem:
What is a wrapper DLL and when do I need one?

Solution:
A wrapper is a piece of software that provides a compatibility layer to another piece of software. One is often necessary when developing LabVIEW applications because third-party DLLs were usually designed to be accessed from C (or similar low-level languages) and not LabVIEW. Such a DLL may, for example, return pointers or complex data structures which LabVIEW cannot easily handle.

Writing a wrapper DLL can be compared to writing a completely separate program in C that accesses the original DLL in the way the original author intended. In turn, this wrapper program has been specifically designed to be accessed from LabVIEW. In this sense, the new C program "wraps" around the original C program (DLL) and provides a layer of compatibility. The benefit of a wrapper is that the source code for the original DLL is not necessary, as it does not need to be modified in any way.

Q. OK, this is a concern. Are you saying that there is no shared library (xxx.so) that I can use to access the functionality of the time card? That is all I would need to access the card from LabVIEW (I can create my own wrapper routines if I have a shared library).

- My response to this email was No need to fret! The libtsync.so shared library file is generated during the driver build. It’s located in the TSync/Linux/Lib directory.

Reply Keith sent to a customer (5 Nov 2013)
To answer your question, there are no plans at this time to develop and release a Labview driver/wrapper for the TSync-PCIe timing boards.

However, please note that we provide a **libtsync.so** shared library file with the TSync drivers which a customer can use to create their own wrapper routines. This file is located on the **TSync/Library/Lib** directory. Please note that we do not provide any technical support on how to do this, but your customer is free to try this if they wish. They can download the TSync drivers from our website at no cost if they would like to look into this. The TSync-PCIe drivers can be downloaded at: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=37.

NI LabWindows/CVI 9.0,

Q.Our customer has purchased our Tsync-PCIe timing board, but when they write its testing program based on NI LabWindows/CVI 9.0, (Windows XP), they found the program can not be building. I was wondering could you please help advise for this issue.

A. (reply from Keith 27 Jan 2014)
For your information, Labview programs such as LabWindows require a wrapper to be used in conjunction with the Spectracom TSync-PCIe Windows driver. Spectracom does not provide a Labview wrapper with the TSync-PCIe board drivers, so customers that wish to use the TSync-PCI board with programs such as LabWindows will need to create their own custom wrapper, consisting in part of the API calls from the TSync-PCIe board which they want to use with the LabWindows program.

However, please note that we provide a **libtsync.so** shared library file with the TSync-PCIe drivers, which can be used to create custom wrapper routines. For more information on wrappers, refer to websites such as: http://digital.ni.com/public.nsf/allkb/06ECDC689DDA0F3D862574440074CD95.

Problem:
What is a wrapper DLL and when do I need one?

Solution:
A wrapper is a piece of software that provides a compatibility layer to another piece of software. One is often necessary when developing LabVIEW applications because third-party DLLs were usually designed to be accessed from C (or similar low-level languages) and not LabVIEW. Such a DLL may, for example, return pointers or complex data structures which LabVIEW cannot easily handle.

Writing a wrapper DLL can be compared to writing a completely separate program in C that accesses the original DLL in the way the original author intended. In turn, this wrapper program has been specifically designed to be accessed from LabVIEW. In this sense, the new C program "wraps" around the original C program (DLL) and provides a layer of compatibility. The benefit of a wrapper is that the source code for the original DLL is not necessary, as it does not need to be modified in any way.

Linux driver specific

- **Shortcut to Tsync driver support spreadsheet:** I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\Tsync family Driver support spreadsheets
- **Shortcut to firmware and driver version upgrades in custservice:** I:\Customer Service\PSB, PSP software updates\TSYNC boards
- **Shortcut to TSync-PCIe driver release notes:** I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\TSync driver updates
- **Shortcut to drivers CD (1219-5003-6001) in Arena:** https://app.bom.com/files/detail-summary?file_master_id=1234039656&file_id=1745191706
- **Shortcut to Linux driver (1219-SD01-xxxx) in Arena:** https://app.bom.com/items/detail-spec?item_id=1207257433&version_id=10292084078

Note: CentOS and Scientific Linux are Linux distributions that are based on the Red Hat linux distribution. Issues that affect Red Hat Linux will also likely affect these other distributions as well.

****Notes about Linux driver version updates**

A) Driver version 3.1.2

- Not compatible with TSync-PTP boards (will break comms between KTS and the PTP module)
- Officially tested on kernel **versions 2.6.9 to 2.6.39 and 3.0.0 through 3.13.0**
- Unofficially, can be installed on kernel versions **to version 3.16** without any issues.
- Kernel versions 3.17 and above require a patch also be installed with the driver to address one issue with newer kernels

B) Previous driver version 2.5.1

- This is the last driver update that can be applied to TSync-PTP boards without breaking PTP functionality.

C) Previous driver version 2.5.0

(Nov 2011 Version 2.5.0) The PCIe Linux driver does not support any 2.4.x kernels. It now supports 32 bit or 64 bit 2.6.x kernels from 2.6.9 through 2.6.39 and 3.0.0 (see note below)

Unofficial kernel support with version 2.5.0 linux driver

SalesForce Case 7453 Keith, the driver was able to build and load without issue on our 3.5.7 kernel.
Lisa Perdue was also able to compile this driver on kernel version 3.2.0.

Note (5 Sept 2013): Tim Tetreault has tested v2.5.0 driver on kernel versions to 3.8.0 with no problems. A minor patch is required for compiling on version 3.9, as described below (this patch will be included in the next driver update)

I found we have tested Kernel 3.8 and it works fine. Version 3.9 has one slight problem which will be addressed in a future Driver update. This should be released in early 2014. In the mean time there is a fix:

I believe the only change required is to the file "tsync_drv_2_6_13_1.c" located in "/tsync/linux/driver/" folder in our driver. The change is

```
/*
```

```
** tsyncpci_driver PCI driver information structure
*/
static struct pci_driver tsyncpci_driver =
{
    name:     TSYNC_PCI_DRV_NAME,
    id_table: tsyncpci_pci_tbl,
    probe:    tsyncpci_init_one,
#if LINUX_VERSION_CODE >= KERNEL_VERSION(3,8,0)
    remove:   tsyncpci_remove_one,
#else
    remove:   __devexit_p(tsyncpci_remove_one),
#endif
    shutdown: tsyncpci_remove_one,
};
```

Once you make this change, you should be able to compile the driver without any errors.

D) Previous Version 2.4.1 driver

(4/17/09 Version 2.4.1) The PCIe Linux driver does not support any 2.4.x kernels. It supports 32 bit or 64 bit 2.6.x kernels from 2.6.13 to 2.6.25.

5/26/09 –Compatible with x86 processor only. Does not support Sparq processor.

Driver needs to be built before use. Requires GCC and MAKE utilities and the GNU C library.

Email from Tim Tetreault to a customer: The driver its self was written using C/C++ but that doesn't mean you can't use other software languages to interface to it. I know we have other customers that have written software in Java to interface to our boards. I do not have example Java code that I can give you though.

Q. Can the card be used with a PowerPC Board (P2020 by Freescale) or just on a x86 Platform.
A. **(From Tim Tetreault 6/21/12)** The card has been used on PowerPC machines running Solaris but I am not sure about Linux. Are they running a version of Linux on the PowerPC machine?

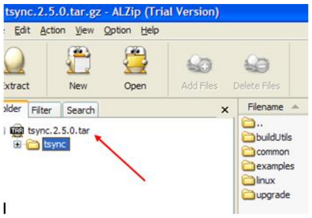
Linux Debian Stretch

Q The customer would like to install the card under Linux.Can you tell me if your drivers are available for Linux Debian Stretch?
A reply from Keith based on info from Pritam (20 Mar 18) Thanks very much for your great question. I had a pretty good idea on the answer to it, but I checked with our Engineering team just to be certain.

They confirmed my belief that there should be no problem installing our standard/freely available TSync series Linux driver onto Linux Debian Stretch (The 'Debian Stretch' is just a development version of Debian Linux, which the Linux driver is compatible with).

**Determining the current linux driver version

- The version of the driver is indicated in the name of the tsync .tar file (such as tsync.2.5.0.tar indicates driver version 2.5.0 for example).



Installing the Linux driver (no previous version of the driver already installed

Steps to install the linux driver

1) Open a terminal window.
2) Make sure you are logged in as a root user.
3) Copy the driver file to a convenient directory location.
4) Change to the directory in which the driver files were copied.
5) Extract the driver using the following command: **gunzip –c tsync.<rev>.tar.gz | tar –xvf –**
6) Build the driver by issuing the commands below:

```
cd tsync
cd linux
make clean
make
make install
```

7) Load the driver by issuing the command: **modprobe tsyncpci**
8) To verify that the driver has been installed, type at the prompt: **lsmod**
9) Verify that the driver "**tsyncpci**" is present.

Notes:

- **"rmmod tsyncpci" =** unload loadable module
- **"modprobe tsyncpci"** = rebuilds and installs the driver
- **"lsmod" =** prints the contents of the /proc/modules file. It shows which loadable kernel modules are currently loaded.

Example successful customer driver install

"Here is the final steps that I took on the machine that is working:"

```
[root@vr2b linux]# make install
make -C driver install
make[1]: Entering directory `/home/vruser/gps_tsync/tsync/tsync/linux/driver'
rm -f /etc/udev/rules.d/25-tsyncpci.rules
install -D -m 644 25-tsyncpci.rules /etc/udev/rules.d/25-tsyncpci.rules
rm -f /lib/modules/2.6.32-358.el6.x86_64/kernel/drivers/tsyncpci.ko
install -D -m 644 tsyncpci.ko /lib/modules/2.6.32-358.el6.x86_64/kernel/drivers/tsyncpci.ko
depmod -a || true
make[1]: Leaving directory `/home/vruser/gps_tsync/tsync/tsync/linux/driver'
```

```
[root@vr2b linux]# modprobe tsyncpci
in /var/log/messages:
Nov 22 07:34:49 vr2b kernel: Spectracom TSync-PCIe Timing Board - version 2.50
Nov 22 07:34:49 vr2b kernel: Copyright (c) 2009 Spectracom Corporation
Nov 22 07:34:49 vr2b kernel: tsyncpci 0000:05:00.0: PCI INT A -> GSI 34 (level, low) -> IRQ 34
Nov 22 07:34:49 vr2b kernel: TSYNC version of board: tsyncpci0
Nov 22 07:34:49 vr2b kernel: IRQ 34/tsyncpci0: IRQF_DISABLED is not guaranteed on shared IRQs
```

```
[root@vr2b vruser]# modprobe -l | grep -i tsync
kernel/drivers/tsyncpci.ko
```

```
[root@vr2b linux]# ls -al /dev/|grep -i tsync
crw-rw-rw-   1 root root    248,   0 Nov 22 07:34 tsyncpci0
```

Everything looks good at this point, and the card is in fact working.

**Updating the Linux driver (Uninstalling a previous version Linux driver and installing a newer version)

Note: When changing the linux kernel version, the Linux driver “install” steps needs to be repeated.

Email from Tim Tetreault:
If the kernel has changed, you will need to repeat the driver installation. Make sure you uninstall and “make clean” before you rebuild the driver.

A. Uninstall the earlier driver first

1) Unload the driver by issuing the following command:
2) rmmod tsyncpci
3) Change to the directory in which the driver files were copied.
4) Unload the driver by issuing the following commands:

```
cd linux
make uninstall
```

B. Install the newer version driver

1) Open a terminal window.
2) Make sure you are logged in as a root user.
3) Copy the driver file to a convenient directory location.
4) Change to the directory in which the driver files were copied.
5) Extract the driver using the following command: **gunzip –c tsync.<rev>.tar.gz | tar –xvf –**
6) Build the driver by issuing the commands below:

```
cd tsync
cd linux
make clean
make
make install
```

7) Load the driver by issuing the command: **modprobe tsyncpci**
8) To verify that the driver has been installed, type at the prompt: **lsmod**
9) Verify that the driver “**tsyncpci**” is present.

Notes:

- "**rmmod tsyncpci**" = unload loadable modules
- "**modprobe tsyncpci"** = rebuilds and installs the driver
- "**lsmod"** = prints the contents of the /proc/modules file. It shows which loadable kernel modules are currently loaded.

Attached is a copy of the TSync-PCIe driver version 2.41 release notes. The first paragraph provides the link to download the latest version of the driver, at no cost. Also attached is the latest version of the TSync-PCIe driver guide. Refer to Section 2.1.2 (page 2-2) for information on uninstalling the current driver and then refer to Section 2.1 (starting on page 2-1) for information on installing this new driver.

Specific to RT Preempt patch

Here are some RT-Preempt patch links:
https://rt.wiki.kernel.org/index.php/Main_Page
https://rt.wiki.kernel.org/index.php/RT_PREEMPT_HOWTO

- ➢ Specifically, we added the RT-Preempt patch to the RHEL 6.3 (see below) kernel, and want to run with that. We use kernel 3.2.13 and apply the RT Preempt patch, giving us kernel 3.2.13-rt23. Do I just simply need to just boot into 3.2.13-rt23, and then do the "make" and "make install" again? How can I build two tsyncpci.ko drivers, one for 2.6.32 and one for 3.2.13-rt23?

Email from Tim Tetreault (9/14/12) If the kernel has changed, you will need to repeat the driver installation. Make sure you uninstall and "make clean" before you rebuild the driver.

Be aware that we have seen other customers that have patched Red Hat with a RT patch that weren't the same kernel that ended up having "make" issues because kernel paths were broken. You will have to sort those out if they happen.

For each kernel, you will need to go through the building of the driver to get the correct "tsyncpci.ko". So you will need to do this for the 2.6.32 and again for the 3.2.13-rt23.

****Best way to verify Linux driver has been properly/successfully installed**.
Email from Tim Tetreault (9/18/12)
The first thing you can do to verify that the driver is installed and working correctly is to type:
Lspci –v –d:8000

This will identify our card on the bus. If the driver is installed correctly, it will also show that kernel driver in use and kernel module is "tsyncpci".

If all you are trying to do is verify that the driver is talking to the board correctly, you can try the following commands:

```
./HW_GetTime 0
./CS_GetTime 0
./LS_GetVersion 0
```

There are many other API calls but without knowing what you application is, I don't know what to suggest.

Note: The lspci command is discussed in more detail in the linux driver troubleshooting section just a little ways further below. Refer to**: lSPCI commmands**

Uninstalling, Loading and Unloading the Linux driver

- **"rm tsyncpci"** = uninstalls the current TSync-PCI driver
- "**make uninstall**" & "**make clean**" = to remove any old driver files.
- **modprobe tsyncpci"** = rebuilds and installs the driver

32 Bit application software on our 64 bit TSync driver (compat_ioctl function)

- Refer to Salesforce case 11668 and Fogbugz case 1856 (request from Raytheon)
- Applicable to at linux drivers at least version 2.5.0 and below (As of linux driver version 3.3.1, I don't believe this functionality has yet been added. But I don't know for certain).

Email from David Qi with Raytheon We would like our 32-bit application to work with the TSync-PCIe Linux driver when it is built to run on a 64-bit Linux kernel.

We know that a 32-bit application cannot work with the TSync-PCIe Linux driver when it is built to run on a 64-bit Linux kernel, because the TSync-PCIe code does not use provide driver support -- in the form of a compat_ioctl function -- that would allow a 32-bit application to work with the driver when it is built to run on a 64-bit kernel.

So, the question we have is this ...

Would Spectracom be able to provide us (and its other customers) with a version of the TSync-PCIe Linux driver that allows a 32-bit application to work with its driver, when its driver is built for a 64-bit Linux kernel?

***static and dynamic library files / .lib files (libtsync.so versus libtsync.a)

- As of TSync-PCIe Linux driver version 2.3.0 (Dec 2010 time-frame), the linux driver supports both statically linked and dynamically/shared linked libraries.
  - From The linux driver v2.3.0 Release Notes: “**Added shared library support (libtsync.so).**”
- The TSync driver provides both a static library (**libtsync.a**) and a shared library (**libtsync.so**).
- The example programs are built linked with the static library.
- The libtsync.so shared library file is generated during the driver build. It’s located in the **Root/Desktop/TSync/Linux/tsync/Lib** directory

Prior to driver version 2.3.0 (before libtsync.so file was available)

Q. I am now trying to link to your driver on my 64-bit linux system since previously I actually had the driver code compiled into my library and that was leading to the undefined symbols. Unfortunately when I link I get the error message near the bottom of the following:

A. Currently our libraries are not built as shared object libraries. They are intended to be statically linked. The lowest level Makefile for our driver library with its CFLAGS is located here: \tsync\linux\tsync\lib\obj\Makefile.
(Note that this was resolved in the version 2.3.0 driver update, Dec 2010)

General info on library files

A) Static library/statically-linked library

(from wikipedia)

static library or **statically-linked library** is a set of routines, external functions and variables which are resolved in a caller at compile-time and copied into a target application by a compiler, linker, or binder, producing an object file and a stand-alone executable.[1] This executable and the process of compiling it are both known as a static build of the program. Historically, libraries could only be *static*. Static libraries are either merged with other static libraries and object files during building/linking to form a single executable or loaded at run-time into the address space of their corresponding executable at a static memory offset determined at compile-time/link-time.

B) Dynamic library

When an app is linked with a library using a static linker, the code that the app uses is copied to the generated executable file. A *static linker* collects compiled source code, known as object code, and library code into one executable file that is loaded into memory in its entirety at runtime. The kind of library that becomes part of an app’s executable file is known as a static library. *Static libraries* are collections or archives of object files.

Advantages and disadvantage of static libraries (https://en.wikipedia.org/wiki/Static_library)

There are several advantages to statically linking libraries with an executable instead of dynamically linking them. The most significant is that the application can be certain that all its libraries are present and that they are the correct version. This avoids dependency problems, known colloquially as DLL Hell or more generally dependency hell. Static linking can also allow the application to be contained in a single executable file, simplifying distribution and installation. With static linking, it is enough to include those parts of the library that are directly and indirectly referenced by the target executable (or target library). With dynamic libraries, the entire library is loaded, as it is not known in advance which functions will be invoked by applications. Whether this advantage is significant in practice depends on the structure of the library.

In static linking, the size of the executable becomes greater than in dynamic linking, as the library code is stored *within the executable* rather than in separate files. But if library files are counted as part of the application then the total size will be similar, or even smaller if the compiler eliminates the unused symbols. On Microsoft Windows it is common to include the library files an application needs with the application.[2] On Unix-like systems this is less common as package management systems can be used to ensure the correct library files are available. This allows the library files to be shared between many applications leading to space savings. It also allows the library to be updated to fix bugs and security flaws without updating the applications that use the library. In practice, many executables (especially those targeting Microsoft Windows) use both static and dynamic libraries.

Example programs for the Linux driver

Linux Driver example programs:

With the TSync-PCIe board and driver installed, the provided Example programs need to first be built/generated, before they can be used. Refer to “Generating the Example Programs” in the driver guide (1219 -5001-0050) in Arena at: https://app.bom.com/items/detail-spec https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_single_search_p=1&redirect_seqno=6611475121 for the instructions.

Generating the Example programs to be able to use them
A “**makefile**” file is supplied in the **/tsync/examples** directory of the Linux driver. A “make” command is performed to build the examples. The “makefile” can be either used as-is or the customer can modify it as necessary (potentially breaking the example programs if it’s incorrectly edited).

The path to the makefile (“mkDriver”) is C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API\Src.

Generating the Example Programs (From the TSync family driver guide 1219-5001-0050 in Arena at https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002)

The TSync driver provides both a static library (libtsync.a) and a shared library (libtsync.so). The example programs are built linked with the static library. To use the example programs with the shared library, modify the example “makefile” by replacing the libtsync.a with libtsync.so and rebuild.

1) Open a terminal window.

2) Change to the directory in which the driver and its sources were extracted.

3) Build the example programs by issuing the commands below:

```
> cd tsync
> cd examples
> make clean
> make
```

Example programs are all located in: Desktop/tsync/examples

To use the example programs: Either open a terminal window (In “Applications”) or a shortcut to the executable needs to be created and then modified with parameter to specify the board number (normally always “**0**”) and the instance number (as applicable for each command). Just clicking on the ICON for a particular example won’t run that example program.

Notice: Must type **“./ “** before each command in order for the call to process. **Example**: **./LS_GetMessage 0**

- Terminal window will provide a command line prompt.
- Use the **ls** command to list the example programs.
- Use an asterisks (“*”) for a wildcard
- Commands are case-sensitive.

Email Keith sent to customer (5/31/12)
We supply the source code for all of the example programs that can be used to communicate with the TSync-PCIe board. The example programs need to first be generated using the “make” command, before they can be used.

Attached you should find the latest version of the TSync-PCIe driver guide. Please refer to Section 2.1.3 (page 2-2) of this guide to generate the Example programs (if you haven’t already).

A) Example programs for 64 bit Linux

The example programs were initially compiled for 32 bit. In order to use them with 64 bit, all of the examples need to be rebuilt for 64 bit.

The make file needs to be modified to use 64 bit examples. The path to the makefile “mkDriver” is C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API\Src. Refer to Section 2.1.3 (page 2-2 of the attached TSync-PCIe driver guide).

Once the examples are built for 64 bit, they should work fine for you.

Email Keith sent to customer (5/31/12)
We supply the source code for all of the example programs that can be used to communicate with the TSync-PCIe board. The example programs need to first be generated using the “make” command, before they can be used.

Attached you should find the latest version of the TSync-PCIe driver guide. Please refer to Section 2.1.3 (page 2-2) of this guide to generate the Example programs (if you haven’t already).

Example programs provided with Linux driver don’t work

If the following two statements are present in a “LSPCI -v” response, the TSync-PCIe board/driver are installed, but the Example programs may not have been built yet. Or the “makefile” file supplied in the Linux driver may have been modified from the default settings. Send any changes to this file to engineering for review. If lspci doesn’t show these statements, the board and driver aren’t properly installed yet, so the Example programs can’t run. Refer to the Linux driver section of this document for specific troubleshooting guidance.

Kernel driver in use: tsyncpci”
“Kernel modules: tsyncpci”

If the TSync-PCIe board and linux driver are installed, but the Example programs still don’t work, refer to: Example programs included with the drivers (in this document).

With the TSync-PCIe board and driver installed, the provided Example programs need to first be built/generated, before they can be used. Refer to “Generating the Example Programs” in the driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002 for the instructions.

Note: The example programs with the linux driver need to be generated (built) separate from the driver itself.

Refer to the instructions further above on generating the Example programs.

Multi-core / Multi-thread applications

Q. Can the TSYNC card be opened and operated via the Linux device driver by more than one program or thread of a program at a time? In other words, can multiple programs use the card at the same time and read time, set alarms, etc.?
A. Yes our Linux driver is designed to support Multicore & Multi Thread applications. (All three drivers support multi-threaded applications running on multiprocessor machines)

Blocking system calls for multi-threaded applications
None of the Hardware calls (“HW_”, such as the “HW_GetTime” call) are blocked. These calls go directly to registers and don’t go through the FIFO buffer, either. This is why they are considered “**0 latency calls**”

But all of the other API calls go through the FIFO buffer, which handles the blocking of calls as necessary.

**TROUBLESHOOTING linux driver/driver compile errrors:

****** Difficulties installing/building the driver**

- Refer to Salesforce case 10992
- Driver install requires GCC and MAKE utilities and the GNU C library.

Email/Questions Keith sent to a customer to get additional info to diagnose

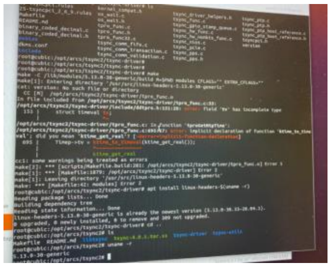
In order to better assist you with the TSync-PCIe driver install, I have some questions for you:

- What is the Serial Number of the TSync-PCIe board (it's indicated on the silver Spectracom Part Number sticker affixed to the TSync-PCIe board)?
- What is the version of the TSync driver you are trying to install? Note the latest driver version is version 2.5.0 and the version of the driver is indicated in the name of the .tar file (such as tsync.2.5.0.tar indicates version 2.5.0 for example).
- What is the linux kernel version installed in this particular machine? Note the latest driver (version 2.5.0) is officially compatible with Linux kernel versions 2.6.9 to 2.6.39, and 3.0.0. However, it has been successfully installed on kernel versions 3.2.0 and 3.5.7
- Are you getting any reported specific or general compile error message being displayed?
- Is the TSync board already installed? If it is, temporarily uninstall it. Then, try to build the driver again with it removed. Does it successfully build this time?
- Do you have GCC and Make utilities as well as the GNU C library installed

If you are getting any compile errors, please send me a copy/paste or screenshots of the error messages for our review. Based on the responses above, we may need to get our Engineering team involved with the diagnosis of why the driver isn't building for you.

Driver compile issues with specific linux distributions

1. **Compile error message "Error: Field 'tv' has incomplete type)**
   - Refer to Salesforce case 281583 (Feb 2022)



```
[illegible]rules              [illegible]
25-tsyncpci_2_0_5.rules  kernel_compat.h
Makefile                 os_wait.c            tsync_driver_helpers.h      tsync_ptp.c
README.md                os_wait.h            tsync_func.c                tsync_ptp.h
binary_coded_decimal.c   tpro_func.c          tsync_gpio_stamp_queue.c    tsync_ptp_host_reference.c
binary_coded_decimal.h   tpro_func.h          tsync_hw_func.c             tsync_ptp_host_reference.h
[illegible]              tpro_func32.c        tsync_hw_nonkts_func.c      tsyncpci.h
dkms.conf                tsync_comm_fifo.c    tsync_ioctl.c               version
include                  tsync_comm_transaction.c  tsync_pcie.c
                         tsync_comm_validation.c   tsync_pps.c
                                                   tsync_pps.h
root@cubic:/opt/arcs/tsync2/tsync-driver#
root@cubic:/opt/arcs/tsync2/tsync-driver#
root@cubic:/opt/arcs/tsync2/tsync-driver# make
make -C /lib/modules/5.13.0-30-generic/build M=$PWD modules CFLAGS="" EXTRA_CFLAGS=""
make[1]: Entering directory '/usr/src/linux-headers-5.13.0-30-generic'
cat: version: No such file or directory
  CC [M]  /opt/arcs/tsync2/tsync-driver/tpro_func.o
In file included from /opt/arcs/tsync2/tsync-driver/tpro_func.c:33:
/opt/arcs/tsync2/tsync-driver/include/ddtpro.h:151:20: error: field 'tv' has incomplete type
  151 |      struct timeval  tv;
      |                      ^~
/opt/arcs/tsync2/tsync-driver/tpro_func.c: In function 'tproGetRtcTime':
/opt/arcs/tsync2/tsync-driver/tpro_func.c:695:17: error: implicit declaration of function 'ktime_to_time
val'; did you mean 'ktime_get_real'? [-Werror=implicit-function-declaration]
  695 |      Timep->tv = ktime_to_timeval(ktime_get_real());
      |                  ^~~~~~~~~~~~~~~~
      |                  ktime_get_real
cc1: some warnings being treated as errors
make[2]: *** [scripts/Makefile.build:281: /opt/arcs/tsync2/tsync-driver/tpro_func.o] Error 1
make[1]: *** [Makefile:1879: /opt/arcs/tsync2/tsync-driver] Error 2
make[1]: Leaving directory '/usr/src/linux-headers-5.13.0-30-generic'
make: *** [Makefile:42: modules] Error 2
root@cubic:/opt/arcs/tsync2/tsync-driver# apt install linux-headers-$(uname -r)
Reading package lists... Done
Building dependency tree
Reading state information... Done
linux-headers-5.13.0-30-generic is already the newest version (5.13.0-30.33~20.04.1).
0 upgraded, 0 newly installed, 0 to remove and 309 not upgraded.
root@cubic:/opt/arcs/tsync2/tsync-driver# cd ..
root@cubic:/opt/arcs/tsync2# ls
Makefile  README.md  libtsync  tsync-4.0.1.tar.xz  tsync-driver  tsync-utils
root@cubic:/opt/arcs/tsync2# uname -r
5.13.0-30-generic
root@cubic:/opt/arcs/tsync2# 
```

Fix: (per customer) "had to use Python 2 not Python 3"

2. **Compile error message "fatal error: linux/timekeeping.h: No such file or directory"**
   - Refer to Salesfore case 206590
   - observed after customer updated kernel version 3.10 to 3.16

the error:

```
# make all
make -C driver
make[1]: Entering directory `/root/tsync/linux/driver'
make -C /lib/modules/3.16.72-2.el7.x86_64/build M=/root/tsync/linux/driver EXTRA_CFLAGS="-Wno-implicit-function-declaration" modules
make[2]: Entering directory `/usr/src/kernels/3.16.72-2.el7.x86_64'
CC [M] /root/tsync/linux/driver/tsync_drv_2_6_13_1.o
/root/tsync/linux/driver/tsync_drv_2_6_13_1.c:70:31: fatal error: linux/timekeeping.h: No such file or directory
#include <linux/timekeeping.h>
^
compilation terminated.
make[3]: *** [/root/tsync/linux/driver/tsync_drv_2_6_13_1.o] Error 1
make[2]: *** [_module_/root/tsync/linux/driver] Error 2
make[2]: Leaving directory `/usr/src/kernels/3.16.72-2.el7.x86_64'
make[1]: *** [all] Error 2
make[1]: Leaving directory `/root/tsync/linux/driver'
make: *** [all] Error 2
```

Email Keith sent (27 Aug 2019) Regarding our case number 206590 for your report of having a problem re-compiling the TSync-PCIe Linux driver, I wanted to provide you with a quick status update on this case.

One of our Apps Engineers is in the process of trying to fully update the Linux kernel to 3.16 on his test machine (he is currently testing with 3.10 installed).

He reported to me that looking at the Linux source code, it appears "timekeeping.h" was removed in kernel 3.16. The function the TSync-PCIe driver generally uses from "timekeeping.h" appears to be in a "time.h" file in the kernel version you are now running.

He said that If you would like to try changing the include to this other file, it might work (he just can't test/verify this change for you, until he is able to get the machine updated to kernel 3.16). Note that this appears to have been a "temporary change" for 3.16- from what he can tell, "timekeeping.h" was added back in, after version 3.16.

3. Compile error message "CONFIG_RETPOLINE=y but not supported by the compiler

- ➢ refer to Salesforce Case 191443

Customer report: "I am trying to compile the Linux driver for the Tsync-PCIe-000. I am running Red Hat 6.10 Workstation 64-Bit and I am getting an error message "CONFIG_RETPOLINE=y but not supported by the compiler".

Q what version Linux kernel do you have installed?
A Red Hat Enterprise Linux 6.10 Workstation – Kernel: 2.6.32.754.el6.x86_64

Solution

Per Ron Dries (6 June 2019) I setup a Red Hat 6.10 system and was able to see the "CONFIG_RETPOLINE=y" compile errors.

The issue is that the compiler, gcc, was not installed.

After installing gcc I was able to build the driver, and communicate with the card.

4. CentOS 7/Red Hat 7: "implicit declaration of function 'do_gettimeofday' [-Werror=implicit-function-declaration]"

- ➢ refer to Salesforce cases 172325 and 175630

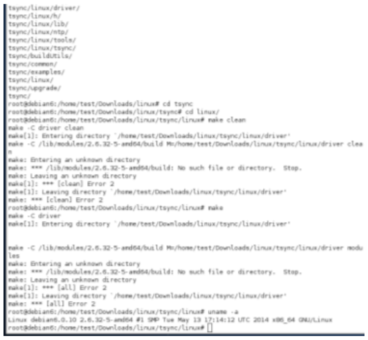
```
make -C /lib/modules/3.10.0-693.21.1.el7.x86_64/build M=/home/jfl/Software/tsync/linux/driver
EXTRA_CFLAGS="" modules
make[1]: Entering directory `/usr/src/kernels/3.10.0-693.21.1.el7.x86_64'
```

```
CC [M] /home/jfl/Software/tsync/linux/driver/tsync_drv_2_6_13_1.o
CC [M] /home/jfl/Software/tsync/linux/driver/tpro_func.o
/home/jfl/Software/tsync/linux/driver/tpro_func.c: In function ‘tproGetNtpTime’:
/home/jfl/Software/tsync/linux/driver/tpro_func.c:692:5: error: implicit declaration of function ‘do_gettimeofday’ [-Werror=implicit-function-declaration]
do_gettimeofday(&(Timep->tv));
^
cc1: some warnings being treated as errors
make[2]: *** [/home/jfl/Software/tsync/linux/driver/tpro_func.o] Error 1
make[1]: *** [_module_/home/jfl/Software/tsync/linux/driver] Error 2
make[1]: Leaving directory `/usr/src/kernels/3.10.0-693.21.1.el7.x86_64'
make: *** [all] Error 2
```

Fix (per this customer) 24 Aug 2018: The compiler error was due to a missing definition to the do_gettimeofday function. We found the definition buried in the kernel header files in the file timekeeping.h and just added to the ddtpro.h, and it compiled. It was a oneline fix.

5. Debian (and possibly others, besides Redhat)

- Customer had compile errors with Debian 6, (but Tim T found it’s the exact same case in newer versions, as well).
- Refer to Salesforce case 19761.



```
tsync/linux/driver/
tsync/linux/h/
tsync/linux/lib/
tsync/linux/ntp/
tsync/linux/tools/
tsync/linux/tsync/
tsync/buildUtils/
tsync/common/
tsync/examples/
tsync/linux/
tsync/upgrade/
tsync/
root@debian6:/home/test/Downloads/linux# cd tsync
root@debian6:/home/test/Downloads/linux/tsync# cd linux/
root@debian6:/home/test/Downloads/linux/tsync/linux# make clean
make -C driver clean
make[1]: Entering directory `/home/test/Downloads/linux/tsync/linux/driver'
make -C /lib/modules/2.6.32-5-amd64/build M=/home/test/Downloads/linux/tsync/linux/driver clea
n
make: Entering an unknown directory
make: *** /lib/modules/2.6.32-5-amd64/build: No such file or directory.  Stop.
make: Leaving an unknown directory
make[1]: *** [clean] Error 2
make[1]: Leaving directory `/home/test/Downloads/linux/tsync/linux/driver'
make: *** [clean] Error 2
root@debian6:/home/test/Downloads/linux/tsync/linux# make
make -C driver
make[1]: Entering directory `/home/test/Downloads/linux/tsync/linux/driver'


make -C /lib/modules/2.6.32-5-amd64/build M=/home/test/Downloads/linux/tsync/linux/driver modu
les
make: Entering an unknown directory
make: *** /lib/modules/2.6.32-5-amd64/build: No such file or directory.  Stop.
make: Leaving an unknown directory
make[1]: *** [all] Error 2
make[1]: Leaving directory `/home/test/Downloads/linux/tsync/linux/driver'
make: *** [all] Error 2
root@debian6:/home/test/Downloads/linux/tsync/linux# uname -a
Linux debian6.0.10 2.6.32-5-amd64 #1 SMP Tue May 13 17:14:12 UTC 2014 x86_64 GNU/Linux
root@debian6:/home/test/Downloads/linux/tsync/linux# 
```

Fix: **Email from Tim T (17 Nov 15)** Keith,
I was able to download and install Debian 6 on my open frame computer. To get the driver to compile correctly, they need to install the “Linux-Source” and “Linux-Header” files. I used the “Package Manager” to install the missing packages. Once I did that, the driver compiles and works.

*** TSync-PCIe board is unexpectedly rebooting

If the TSync-PCIe board is unexpectedly rebooting for any reason:

1) Verify the +3.3vdc supply voltage from the PCIe bus. It may be dropping below the minimum of about 3.17vdc, even for just a moment, causing the low voltage detection circuit to reset the board. Refer to Salesforce case 10297 for an example. For troubleshooting, refer to: Power (+3.3vdc +12vdc and -12vdc from PCI bus)/logic in this document.

2) Verify there are no generic class code/driver conflict issues, where something else installed is causing the board to reboot. Refer to Salesforce case 10297 for an example. For troubleshooting with a Windows PC, refer to: Windows Driver compatibility in this document.

****Two or more TSync-PCIe boards installed, but only one being detected

Troubleshoot: Try swapping the known good board with one that is not being recognized (put it in the slot that is recognizing the board being installed). If the other board is recognized, the board is OK. It's a system issue preventing the other board(s) from being detected.

- lspci commands
- Checks to see if the timing board is visible on the PCI bus
- Checks to see if PCIe driver is installed.
- Board needs to be installed to see it in the results.
- The driver does not need to be installed to have it be reported. In
- **Our Vendor ID**: note from Denis Reilly (12 Feb 2013) "Our vendor ID is **1AD7**".

Perform an lspci –vvv

- ➢ Open a terminal window and type the commands.

Issues that appear to be associated with the hardware (such as interrupts): Have the customer perform and send to us an lspci –vvv. This lists modules installed in the machine.

Even without the TSync-PCIe driver installed, this command should show the TSync board being installed.

In Linux, you can run the following command as root to look for the PCIe card specifically on the PCI bus:

"lspci -v" = Lists the devices founds on the PCI bus (-v gives minimal info, -vv give more info and -vvv gives all available info)

lspci -d 1ad7:8000

"**lspci –d:8000 –vv**" If the lspci response above has no indication of "tsyncpci", the TSync-PCIe board is not being found. This command should then be issued to see if the board can be found using the sub system ID number assigned to the TSync board.

If still no indication of "tsyncpci" in the response, may be a bad board or a major linux system issue.

Note: The following entries should also be present, if the driver is installed:
Kernel driver in use: tsyncpci
Kernel modules: tsyncpci

Results of lspci:

A) Updated Class Code (ECN 3159- released 4 Apr 2013)

- Approximate Manufacturing cut-in Serial Number to have the new class code: 01998

The new class code will now report:
Base Class = 11h (**Communications synchronizer**) Sub-Class = 10h Interface = 00h

First line of response should be:
nn.nn.n 1a:00.0 Communication synchronizer: Spectracom Corporation Device 8000

(where **nn.nn.n** is the bus ID number and is system-dependent, and 1ad7:8000 is our Vendor and Device ID for the TSync card)

Note: With the TSync board installed and the driver loaded, the following two lines should be present in an lspci response (likely at the very end of the response).
Kernel driver in use: tsyncpci"
"Kernel modules: tsyncpci"

6. lspci shows only the limited info below ("unknown header type")

07:00.0 Communication synchronizer: Spectracom Corporation TSync-PCIe Time Code Processor (rev ff) (prog-if ff)
!!! Unknown header type 7f

- Refer to Salesforce case 11923 (https://na8.salesforce.com/500C000000U0ZmA)
- lspci shows board is still installed and was working fine. Now, it can no longer communicate.
- Likely due to a recent (manual or automatic) update to the OS being applied, but the driver has been recompiled to the new kernel version yet.
- Need to recompile the driver with a make clean and normal make install.

Thanks for bringing your customer's observations to our attention.

Please ask your customer if they have performed any recent updates to their system, or if the linux distribution is setup to provide automatic updates to the OS.

When updates are performed, it can affect the header files. We recommend they just recompile their driver with a make clean and make install (they are likely very familiar with how to recompile the driver, but if they need any information on this, just let me know).

Please also let me know the lspci –vv command shows all of the normal info after the driver has been recompiled. I want to make sure that they are all set with communicating with it again (I appreciate it)!

B) Original/Generic Class Code (prior to ~March 2013)

First line of response should be either:

nn.nn.n Non-VGA unclassified device: Spectracom Corporation Device 8000

- nn.nn.n Non-VGA unclassified device: Unknown device 1ad7:8000

(where **nn.nn.n** is the bus ID number and is system-dependent, and **1ad7:8000** is our Vendor and Device ID for the TSync card)

Note: With the TSync board installed and the driver loaded, the following two lines should be present in an lspci response (likely at the very end of the response).
Kernel driver in use: tsyncpci"

“Kernel modules: tsyncpci”

The command will return nothing if the system bus can't see the card.

lsmod commands (Shows driver version and if the driver has been installed correctly)

“lsmod” = Lists the drivers that are currently installed:

Example:

```
Spectracom TSync-PCIe Timing Board - version 2.50
Copyright (c) 2009 Spectracom Corporation
tsyncpci 0000:0a:00.0: PCI INT A -> GSI 16 (level, low) -> IRQ 16
TSYNC version of board: tsyncpci0
```

- The timing board does not need to be installed for this command

```
[root@vr2a linux]# modprobe tsyncpci
```

Note: The Linux driver should be installed in the **/Dev** directory

Example

```
[root@vr2b linux]# ls -al /dev/|grep -i tsync
crw-rw-rw-   1 root root    248,   0 Nov 22 07:34 tsyncpci0
```

“Please make sure the TSync driver is located in the **/Dev** folder. “TSyncpci 0” should be displayed in this folder.

If /Dev directory does not show: “`tsyncpci0`”

- See if ASPM has been disabled.
- Has the driver been updated from earlier version (leaves behind a Rules file)

**** (ASPM) Power Saver/Power Monitor feature causing TSync board to hang

- **Also refer to document Tim Tetreault created:** EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\ASPM power monitor
- 4/19/12 KW – Newer kernel versions are enabling a power saver mode on the PCIe bus that may either cause the TSync-PCIe boards to hang-up right away or stop running after a short period of time.

Note: Refer to Rich Schmidt with USNO in Salesforce for more info on this kernel “feature”.

- Primarily affects Redhat and Redhat direritives. Typically not a problem with Ubuntu.

Symptoms of ASPM potentially being an issue include:

- Intermittent system crashes when the reset command is issued (refer to Salesforce case 9560 https://na8.salesforce.com/500C000000Qvoku
- System never being able to communicate with the board.
- Make sure ASPM is disabled in BIOS
- Follow the ASPM document to make sure ASPM is disabled in the kernel (grub.conf file).

ASPM configuration change may not persist first time its performed
Note that we have heard of instances in the past where the ASPM configuration change didn’t persist the first time through the procedure. This procedure may need to be repeated if it doesn’t take the first time through. To confirm the change did persist, after a system reboot, the response for the “cat /sys/module/pcie_aspm/parameters/policy” command will be “**performance**” as below (bolded for emphasis):

```
[root@gieib19853 ~]# cat /sys/module/pcie_aspm/parameters/policy
default [performance] powersave
```

Note: in RedHat, the line to add to the kernel is in /Boot/Grub and is called **grub.conf**

ASPM issues with CentOS

(KW 3/19/12) Updating from CentOS 5.6 to CentOS 5.7 is one example which can result in abnormal operation with the TSync-PCIe board (such as interrupt conflicts), due to the assigned Class Code (programmed into the TSync-PCIe board).

CentOS 6.2/ RHEL 6.2/scientific linux 6.2

1) CentOS version 5.8 works fine, but version 6.2 causes the TSync-PCIe board to stop responding

Notice: We highly recommend updating to version 6.4, if possible. In March 2013, **CentOS and Scientific Linux released version 6.4**. Version 6.4 has kernel fixes that address some ASPM related issues (We have even noticed here that ASPM not being completely disabled can cause intermittent boot-up issues). The changes in the ASPM document still need to be performed successfully once, but would it be possible to update the system to version 6.4 to see if it completely resolves the system you are observing? If you can update to the newer version, we have noticed that the ASPM changes made in 6.2 persist through the update to 6.4, so they don’t need to be performed again after upgrading to 6.4.

Fix: Refer to “ASPM” in “power saving feature” section below. CentOS/RHEL Version 6.2 enables this feature (for decreased laptop power consumption). We want it disabled, so the TSync-PCIe board doesn’t eventually go into a “sleep” mode. Otherwise, there will be lags in functionality, while the board is waking up.

I have some EXCELLENT news for you!! One of our timing board engineers was able to determine what was causing the issues you were observing with the installed timing boards. The issue is not related at all to either the timing boards or the TSync driver. It has to do with a power saver function being used on the PCIe bus incorporated in the newer kernel versions. With a couple of changes he made, he has been running the system with 2 TSync boards installed and NTP running since yesterday with no problems!!!!

Before sending the system back to you, he would like you to try the changes he has made, just to ensure this has fixed the issues you reported. When you get a moment, can you perform the following steps on a similar system and let us know how it goes? This process changes how the machine boots up.

1) Go to the following directory: #cd boot/grub/
2) Edit the following file: "grub.conf"
3) Add the following to the end of the command line: "pcie_aspm=off" (SEE EXAMPLE BELOW)

```
title CentOS (2.6.32-220.7.1.el6.x86_64)
          root (hd0,0)
          kernel /vmlinuz-2.6.32-220.7.1.el6.x86_64 ro root=/dev/mapper/VolGroup-lv_root rd_NO_LUKS
LANG=en_US.UTF-8 rd_NO_MD rd_LVM_LV=VolGroup/lv_Swap SYSFONT=latarcyrheb-sun16 rhgb
crashkernel=128M quiet rd_LVM_LV=VolGroup/lv_root KEYBOARDTYPE=pc KEYTABLE=us rd_NO_DM
pcie_aspm=off
          initrd /initramfs-2.6.32-220.7.1.el6.x86_64.img
```

Once you do this, reboot the system and load the driver.
Another email from Tim Tetreault (4/20/12) about this topic.
There are 2 other things they can do.

1) I updated the BIOS in the machine they set us. Its new BIOS is:
ProLiant BIOS: 026
BIOS Version: 0260414
BIOS Date: 04/14/11

2) Have them check the following settings in his BIOS:

Under "Advanced Chipset Control"
PCIe Gen2 - Gen2
Enable ASPM - DISABLED

Email from Rich about CentOS 6.2
I discovered that in CentOS 6.2 /sys/modules/pcie_aspm/parameters/policy:
default performance [powersave]

I will test setting this to [disable].

In CentOS 5.8 there is no /sys/modules/pcie_aspm

Q. (from Tim Tetreault to Rich Schmidt with USNO)
What did you do to force the OS to "performance" mode on reboots? I found an app note that recommended that we add the following kernel command line parameter:
pcie_aspm.policy=performance
Is that what you did or did you do something different?
A (from Rich) We simply followed your lead and added pcie_aspm=off to the grub.conf. Now each time we upgrade the kernel we must add this! That is what we forgot to do last time.

Class Code assigned to TSync board

- Some systems, especially newer HP boxes can have problems with our original/generic Class Code
- Updating from CentOS 5.6 to CentOS 5.7 is one example which can result in abnormal operation with the TSync-PCIe board (such as interrupt conflicts), due to the assigned Class Code (programmed into the TSync-PCIe board). A patch is available to update the class code to a newer value, which helps the OS work with the TSync-PCIe.
- For more info on this issue, refer to Salesforce cases for MIT LL (case 4986) and NOAO (case 5048).

Status Update (4 Apr 2013): ECN 3159 was released today to incorporate the Class Code update to all TSync-PCIe boards. Firmware version 2.11.

Obtaining the current Class Code assigned to the TSync-PCIe board

Perform an lspci (refer to “lspci” section above).
This command should respond with either the original class code or the newer class code. See the **Results** below.

Note: The command will return nothing if the system bus can't see the card.

Results of lspci:

A) Updated Class Code (ECN 3159- released 4 Apr 2013)

- Firmware version 2.11.
- Approximate Manufacturing cut-in Serial Number to have the new class code: 01998

The new class code will now report:

Base Class = 11h (**Communications synchronizer**) Sub-Class = 10h Interface = 00h

First line of response should be:
nn.nn.n 1a:00.0 **Communication synchronizer**: Spectracom Corporation Device 8000

(where **nn.nn.n** is the bus ID number and is system-dependent, and 1ad7:8000 is our Vendor and Device ID for the TSync card)

Note: With the TSync board installed and the driver loaded, the following two lines should be present in an lspci response (likely at the very end of the response):
Kernel driver in use: tsyncpci”
“Kernel modules: tsyncpci”

B) Original/Generic Class Code (prior to ~March 2013)

First line of response should be either:
nn.nn.n Non-VGA unclassified device: Spectracom Corporation Device 8000

nn.nn.n Non-VGA unclassified device: Unknown device 1ad7:8000

(where **nn.nn.n** is the bus ID number and is system-dependent, and **1ad7:8000** is our Vendor and Device ID for the TSync card)

```
04:00.0 Non-VGA unclassified device: Spectracom Corporation TSync-PCIe Time Code Processor
        Subsystem: Spectracom Corporation TSync-PCIe Time Code Processor
        Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B-
DisINTx-
```

```
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR-
INTx-
        Latency: 0, Cache Line Size: 256 bytes
        Interrupt: pin A routed to IRQ 16
        Region 0: Memory at fbcfe000 (32-bit, non-prefetchable) [size=512]
        Capabilities: <access denied>
        Kernel driver in use: tsyncpci
```

Note: With the TSync board installed and the driver loaded, the following two lines should be present in an lspci response (likely at the very end of the response).

Kernel driver in use: tsyncpci”
“Kernel modules: tsyncpci”

The command will return nothing if the system bus can't see the card.

Upgrading the firmware to apply the more specific Class Code

Email from Tim Tetreault to a customer (3/16/12)
Here is the patch and instructions on how to **update** your boards to fix the Class Code issue. If you install the card in a system that was having a problem, you should do the following:
Uninstall the current driver. “**rm tsyncpci**”
Run the “**make uninstall**” & “**make clean**” to remove any old driver files.
Reboot the computer to that the system can identify the card with the new class code.
Rebuild the driver and install using the “**modprobe tsyncpci**” command.

In Linux, you can run the following command as root to look for the PCIe card specifically on the PCI

BIOS Settings / Issues with “AMI UEFI core” in BIOS

System’s BIOS settings for the PCIe bus may be causing an issue. May need to update or change the BIOS settings.

Email from Alan Leung with DSPcon Got the issue resolved. Looks like it was a bios setting that needed to be changed from 16x/8x to 8x/8x in the PCI configuration.

AMI UEFI core:

Email from Tim Tetreault (2 Apr 2013) I spent a little time looking on line about the computer they using and found some people reporting issues related to the bios “AMI UEFI core” that this system uses. I have come across other customers in the past that had issue that were caused by BIOS issues and not card or hardware problems.
Ask the customer if they have checked to see if their system has a newer BIOS available.

Also, I want the customer to with just the board installed without the driver installed, run “lspci –vv” and send us the output.

Issues associated with specific linux distributions

1. **RedHat 7 does not reload properly after reboot. After reboot, BIOS and OS does not see the PCIe card**

   ➢ Refer to Salesforce Case 253041 (Nov 2020)

   TSync PCIe card running on OS: RedHat 7 does not reload properly after reboot. After reboot, BIOS and OS does not see the PCIe card. Only after performing shutdown does the OS recognize the PCIe card and subsequently functions as expected

2. **Earlier Linux drivers Versions 2.4.1 and 2.3.0 have an issue with the Rules file (Redhat/CentOS or Scientific Linux distributions)**

   The Rules files are located in /**etc/udev/rules.d**:

   Note: Refer to: “Versions 2.4.1 and 2.3.0 have an issue with the Rules file” (Redhat/CentOS or Scientific Linux distributions) below for more information.

   ➢ The correct Rules file for driver version 2.5.0 (or above) is “tsyncpci.rules”

   ➢ The older Rules file from the versions 2.4.1 and below drivers (for older kernels) was “tsyncpci 2_6_9”. (Note that this Rules file should be deleted, if present with the version 2.50 or higher driver installed)

The Rules files should be located in etc/udev/rules.d:

➢ The correct Rules file for driver version 2.5.0 (or above) is “tsyncpci.rules”

➢ The older Rules file from the versions 2.4.1 and below drivers (for older kernels) was tsyncpci 2_6_9 (this second rules file should be deleted, if present with the version 2.50 or higher driver installed)

Symptoms: Error message **“!Could not open </dev/tsyncpci0> : rc <1>”** is displayed when trying to communicate with the TSync-PCIe board (such as trying to perform a HW_GetTime call or any other call). “The board is plugged in and we see that the lights on the board itself are on (LED lights in front are off). When we do a **“lsmod | grep tsyncpci**” it shows that the driver is installed”.

Linux Red Hat distribution/ earlier driver being updated to 2.5.0 or higher

Symptom: I believe that I have solved the issue I was seeing. I was getting a permissions problem when attempting to access /dev/tsyncpci0. Every time I rebooted the machine the permission would prohibit me from utilizing the card. So, I edited the ‘/etc/udev/rules.d/25-tsyncpci.rules’ file to read ‘KERNEL==”tsyncpci*”, OPTIONS+=”last_rule”, MODE=”0666”’. And now upon boot the permissions are set so that the card is usable.

Email highly likely was from Tim T (7/06/11) The TSync-PCIe Linux driver versions 2.4.1 (and possibly earlier or later) has a known issue when using Linux Red Hat distribution (Red Hat versions 5 and 6 only – not applicable to Red Hat version 4).

A simple modification of the driver is required for the timing board to be recognized by the OS (Before making this simple mod, you can’t communicate with the timing board, as shown with “comm error” being returned, can’t run example programs, etc).

Associated Email I sent to Gigi Mathew and Steven Lockhart:
Regarding the Spectracom TSync-PCIe timing board driver issue you are seeing, I have received some additional information from our timing board engineers that will likely resolve this issue! There is a simple change to the driver that needs to be made when using the driver with the Red Hat distribution.

Please note: Before you make this mod, make sure that you uninstall the current driver before recompiling and installing the driver with the mod to the make file!

In the makefile under “tsync/linux/driver”, change the following line:

```
#-------------------------------------------------
# Test if kernel version is 2.6.13 or later
#-------------------------------------------------
ifeq ($(K_VERSION), post)
    RULEFILE:=25-tsyncpci.rules
else
    RULEFILE:=25-tsyncpci_2_6_9.rules
endif
```

Once you have done this, run “make clean”, “make”, “make install” and “modprobe tsyncpci”. Verify with “lsmod” that “tsyncpci” is present.

You should probably remake the examples.

Once you have made this minor change to the driver, please let me know that this mod resolved the issue and that you are now able to successfully run the example programs.

I was discussing this situation with one of our timing board engineers. We believe you may have an earlier version of the TSync-PCIe Linux driver currently installed. There is an issue with CentOS (and other Linux distributions) associated with a Rules file. This issue can be fixed by completely removing the earlier version of the driver and then installing the latest version (rmmod of TSync-PCI and then make uninstall). The error message you reported (“could not open…”) indicates the driver can’t talk to the TSync-PCIe board. With this Rules issue, everything looks fine, but this symptom occurs.

Attached are the TSync-PCIe driver Release Notes. The latest version of the Linux driver is version 2.5.0 (released in November 2011). If you are currently running previous versions 2.4.1 or 2.3.0 removing the driver and then installing the latest will likely resolve this issue. The latest version of the Linux driver can be downloaded at no cost from the Spectracom website. The link to the latest version of the Linux driver is: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=101.

Please let me know that uninstalling the current driver and installing the latest driver resolves this issue for you!!!

Another email sent to a different customer
A question for you- Was an earlier version of the TSync-PCIe driver ever installed on this machine? Or, is the version 2.5.0 driver the only TSync driver that has been installed?

If an earlier version of the TSync-PCIe driver was ever installed, it leaves behind a Rules file that will prevent the driver from being able to work with the TSync-PCIe board. The Rules files are stored in the **/etc/Dev** folder. The correct Rules file is “**tsyncpci.rules**”.

However, if a file called “**tsyncpci 2_6_**9” is also located in this same directory, this file should be **removed**.

Issues with input power (+3.3vdc and +12vdc)

- Issues with Input power from the PCI bus (especially with +3.3vdc) can hold the board in constant reset.
- Verify Test points for input voltages. Refer to: Power (+3.3vdc +12vdc and -12vdc from PCI bus)/logic

Missing header file reported when earlier linux version 2.3.0 driver is installed

- This is due to a change to a newer kernel version relocating where this header file is placed (so it’s not able to be found).
- The fix is to update the Linux driver from 2.3.0 to a newer version driver (v2.4.1 or beyond).

Email from Jacques Kronbauer
I am having a problem while building linux driver caused by a missing header file that was supposed to be in file tsync.1.30.tar.gz.

Specifically the problem is when I try to build it by using make, I run into compilation error caused by a missing header file that was supposed to be in file tsync.1.30.tar.gz.

It seems that there should be a directory ‘asm’ under linux and it should have a file called semaphore.h.

I checked Spectracom’s web site for a newest file but could not find it.

The problem is highlighted with ******* below.

```
make -C /lib/modules/2.6.32.12-0.7-default/build M=/home/DS/kro55737/tsync/tsync/linux/driver modules
make[2]: Entering directory `/usr/src/linux-2.6.32.12-0.7-obj/x86_64/default'
make -C ../../../linux-2.6.32.12-0.7 O=/usr/src/linux-2.6.32.12-0.7-obj/x86_64/default/. modules
  CC [M]  /home/DS/kro55737/tsync/tsync/linux/driver/tsync_drv_2_6_13_1.o
In file included from /home/DS/kro55737/tsync/tsync/linux/driver/tsync_drv_2_6_13_1.c:66:
****** /home/DS/kro55737/tsync/tsync/linux/driver/tsyncpci.h:29:27: error: asm/semaphore.h: No such file or directory
*****
```

My response, with assist from Tim Tetreault (9/18/12 KW)
This header file issue is resolved by upgrading the TSync linux driver to the latest version, version 2.4.1. In summary, this issue is due to a change in the location of the header file in newer kernel versions.

Attached is a copy of the TSync-PCIe driver version 2.41 release notes. The first paragraph provides the link to download the latest version of the driver, at no cost. Also attached is the latest version of the TSync-PCIe driver guide. Refer to Section 2.1.2 (page 2-2) for information on uninstalling the current driver and then refer to Section 2.1 (starting on page 2-1) for information on installing this new driver.

**Desire to sync a Linux machine with NTP to an installed TSync-PCIe board:

- Refer to the UNIX Tech Note (especially sections 2.1 and 4): I:\Customer Service\Software\UNIX
  - Note there are limitations to being able to use the timing board as an NTP reference, as described in this other document (and below).
- Refer to sites such as: http://www.eecis.udel.edu/~mills/ntp/html/ntpq.html and http://www.novell.com/coolsolutions/trench/418.html

Desire to sync a linux machine (NTP or Chrony) using only external 1PPS input only (no desire to use an external time source)

- Refer to Salesforce Case 281487
- Per Ron Dries, the linux box requires a TSync board be installed, to receive the external PPS.
- Configure the TSync board to sync to Self/EPP0.

(email from Ron Dries 24 Feb 2022) To set the Linux time with TSync they will need to use NTP on the system to get time from TSync. At 4.01 this can be done with Chrony.

They could use a reference for TSync as Self/PPS0 and feed a 1PPS into the TSync board.

Self will use the KTS time it has and always be valid.

Then Chrony would query the time from TSync and set that down to the Linux kernel.

I think this is what they are looking for but we will need to confirm.

NTP Type 45 driver for TSync / Need to apply NTP patch that is supplied with our Linux driver

- ➢ NTP versions 4.2.8 (it was technically actually added in development release v4.2.7p266 per http://archive.ntp.org/ntp4/ChangeLog-stable) added our reference clock driver into NTP (Type 45 driver: Spectracom TSync PCI"), alleviating the need to apply the NTP patch:

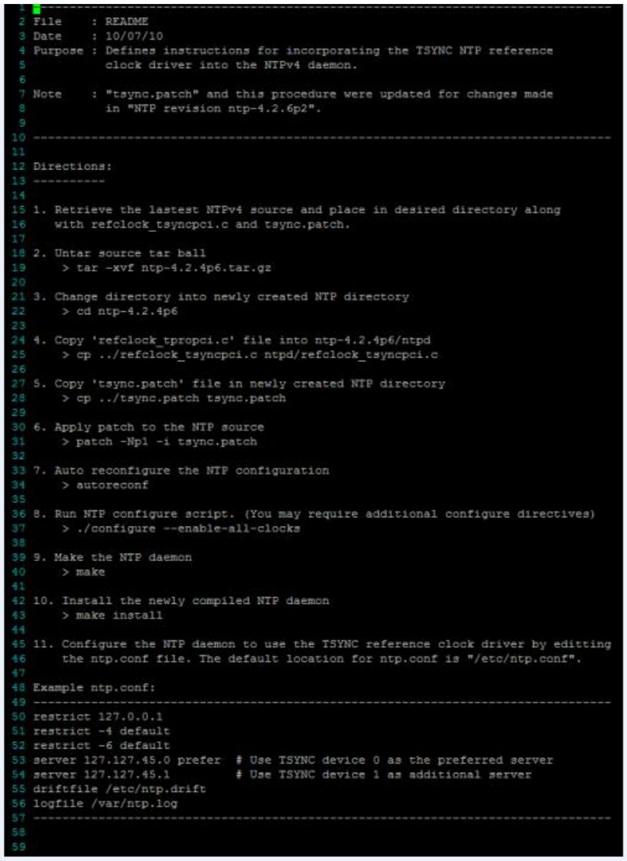
- **Refer to** https://www.eecis.udel.edu/~mills/ntp/html/drivers/driver45.html, http://lists.ntp.org/pipermail/bk-ntp-dev-send/2012-March/002535.html

"(4.2.7p266) 2012/03/21 Released by Harlan Stenn <stenn@ntp.org>
* Add refclock_tsyncpci.c (driver 45) supporting Spectracom TSYNC timing Boards".

- ➢ NTP versions 4.2.6 and below require a patch we supply in the linux driver be applied to NTP. This patch applies our reference clock driver.

- ➢ Refer to the **README** file (as shown below) contained in the Linux driver (**TSync/linux/NTP** folder) for more details

- **link to the README file: (**1191-5003-6001) (Navigate to the "ISS Source" folder and then to the "linux" folder (in the .tar file, go to **tsync/linux/ntp**



```
------------------------------------------------------------------------------
File    : README
Date    : 10/07/10
Purpose : Defines instructions for incorporating the TSYNC NTP reference
          clock driver into the NTPv4 daemon.

Note    : "tsync.patch" and this procedure were updated for changes made
          in "NTP revision ntp-4.2.6p2".

------------------------------------------------------------------------------

Directions:
----------

1. Retrieve the lastest NTPv4 source and place in desired directory along
   with refclock_tsyncpci.c and tsync.patch.

2. Untar source tar ball
    > tar -xvf ntp-4.2.4p6.tar.gz

3. Change directory into newly created NTP directory
    > cd ntp-4.2.4p6

4. Copy 'refclock_tpropci.c' file into ntp-4.2.4p6/ntpd
    > cp ../refclock_tsyncpci.c ntpd/refclock_tsyncpci.c

5. Copy 'tsync.patch' file in newly created NTP directory
    > cp ../tsync.patch tsync.patch

6. Apply patch to the NTP source
    > patch -Np1 -i tsync.patch

7. Auto reconfigure the NTP configuration
    > autoreconf

8. Run NTP configure script. (You may require additional configure directives)
    > ./configure --enable-all-clocks

9. Make the NTP daemon
    > make

10. Install the newly compiled NTP daemon
    > make install

11. Configure the NTP daemon to use the TSYNC reference clock driver by editting
    the ntp.conf file. The default location for ntp.conf is "/etc/ntp.conf".

Example ntp.conf:
------------------------------------------------------------------------------
restrict 127.0.0.1
restrict -4 default
restrict -6 default
server 127.127.45.0 prefer  # Use TSYNC device 0 as the preferred server
server 127.127.45.1         # Use TSYNC device 1 as additional server
driftfile /etc/ntp.drift
logfile /var/ntp.log
------------------------------------------------------------------------------
```

The TSync-PCIe board can be used to sync the Linux machine it's installed in. The Linux driver includes a patch to apply to NTP running on that machine and the instructions on how to do this. After they compile the new NTP, they have to run NTP from the new patched NTP and not the original compile of NTP.

**Also, refer to Section 2.1 of the Unix Ap Note: I:\Customer Service\Software\UNIX. Note there are limitations to being able to use the timing board as an NTP reference, as described in this other document (and below).

Answer: In addition to installing the Linux driver, the customer needs to download, install and compile NTPv4 software onto the PC that the TSync-PCIe board is installed in. The NTPv4 software is a freeware program that is available. To obtain the software and instructions on installing/building the software, have them visit http://www.ntp.org/
After they install NTP, the Linux driver contains a README file that explains how to patch the NTP software in order for the NTP software to use the TSync board as an available NTP reference clock driver to sync NTP. Once the patch is applied, the "patched" NTP software can be run to sync the PC.

(3/2/11 KW) Our current reference table (part of the NTP patch included with the linux driver) does not allow NTP to sync to the board if the board is synced with **Self/EPP0** (the board can declare sync with this, but NTP won't sync to the board). The minimum requirement for NTP to sync to the TSync-PCIe board is **HST0/EPP0** (so that SOME user interaction is necessary in order for NTP to just sync to whatever the board powers up with for the time).

Issues with compiling NTP in SUSE linux

- Tim Tetreault found issues with compiling NTP is SUSE Enterprise linux, even before installing our patch. Not an issue with our code.
- Refer to Salesforce Case 8916 for more info on this customer: https://na8.salesforce.com/500C000000Pvj7X

Email from Tim Tetreault to Leisa Butler
I setup a computer with the version of SLES the customer told us they were running. Then I installed our TSync board with driver. That works just fine. Then I loaded 3 different versions of NTP. I tried our patch and that works ok but when I try to compile them, I start getting errors. I removed our patch and tried just compiling the NTP source. I got the same errors.

I started looking in to what the errors were and it seem to be issues related to missing or old header files that are required for NTP to compile correctly.

I don't think the problem is our driver or patch but a resource issue with the OS and the NTP source. I think if we figure out what the resource issue is, it will resolve the NTP issue and our board will work just fine.

I have it as a task to see if I can find a fix but I don't have a time that I can give you on when I will have a fix.

SLES come with full support for customers that pay for it so you might let the customer know what we have found so they can try to use their Linux support. SUSE Linux might have a fix.

Troubleshooting timing board not syncing the system via NTP

If the linux kernel can't sync to the TSync-PCIe board:

1) Make sure the green Sync LED is lit (TSync-PCIe board needs to be synced).

   Note: You can also perform an **SS_GetSync 0** API call/example program to remotely confirm status (it should report "True", if it's in sync)

2) Verify the TSync-PCIe board and driver can successfully talk to the system by running example programs or API calls. Perform a **HW_GetTime 0** API call/example program to both verify normal operation of the TSync board and to verify the time/date information is correct.

   - (HW_Gettime responds with: Year, DOY, Hr, Min Sec, nanosecond, Sync status**)**

3) Make sure that all of the steps in the NTP README file are being performed.

4) Make sure the version of NTP installed is the one that is specified in the README file. Installing a different version of NTP is not supported and may require customer modify the patch to get it to work

5) What is the selected reference (note that there are limitations to which references can be selected in order for the board to be able to sync NTP)

- (3/2/11 KW) Our current reference table (part of the NTP patch included with the linux driver) does not allow NTP to sync to the board if the board is synced with **Self/EPP0** (the board can declare sync with this, but NTP won't sync to the board). The minimum requirement for NTP to sync to the TSync-PCIe board is **HST1/EPP0** (so that SOME user interaction is necessary in order for NTP to just sync to whatever the board powers up with for the time).

Specific error messages/conditions observed

A) errno (such as "errno = 22" for example)

- ➢ Refer to the following website for a list of errno numbers and what they mean: http://www-numi.fnal.gov/offline_Software/srt_public_context/WebDocs/Errors/unix_System_errors.html
- **For example**:"(errno = 22)" means "Invalid argument"

B) Error message of "Ntpd Select nfound =-1 error: Interrupted system call"

- ➢ Refer to Salesforce case 20912 for Cris Collins

C) "Couldn't initialize device"

- ➢ If this error is reported, follow the troubleshooting steps above to diagnose.

D) NTP Reach value not incrementing

Once they have it running, there is a known issue with the NTP Reach value not incrementing up from "1", according to Tim Tetreault on 11/4/11. This is a known issue that appears to be NTP.

Update to this issue (14 Apr 2014)

- ➢ Issue was fixed in the Linux version 2.3.0 driver update (referred to as the "NTP reference bug")
- ➢ Can update earlier versions of the driver to the latest release to address this.
- ➢ To update the driver, you will need to uninstall the current driver and reinstall the new driver. Per the README file in the "ntp" folder of the driver (also attached for your convenience), you will need to perform step 4 again. Then, recompile NTP.
- ➢ **Note**: A future "major" release of NTP (v4.2.8p3, I believe) will include our reference clock driver with the fix implemented to also address this issue.

Email Keith sent to Steve Korson (11 Apr 2014) The Reach value should normally increment to "377". But due to a minor issue, the Reach value will remain a "1" in earlier version of the driver. This is a very minor reporting condition which was fixed in a newer version of the Tsync Linux driver. Note that it doesn't affect the operation of NTP in any way. However, if you would like to update the driver to fix this issue, the latest version of the driver can be downloaded from our website at the following link: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=101

To update the driver, you will need to uninstall the current driver and reinstall the new driver. Per the README file in the "ntp" folder of the driver (also attached for your convenience), you will need to perform step 4 again. Then, recompile NTP.

Earlier update Note (5/30/12): Tim Tetreault informed me yesterday the NTP User's group found what was causing this minor issue to occur. The fix for this issue will be included in the next release of the TSync-PCIe Linux driver, as it requires a change to the clock driver software contained in the driver (not a change to NTP)

KW- Email I sent to Richard Schmidt about this (11/4/11)
Thanks for your reply and reporting your findings to us. I really appreciate it!

We are aware of the NTP Reach counter not incrementing correctly and have looked into this reporting error already. Other than the Reach value not incrementing, NTP is fully functional. As far as we were able to determine with our investigation, this is a factor of NTP itself and not a factor of the Spectracom driver.

I am going to flag your record in our database and will let you know if we are able to resolve this minor indication error in the future, with either a newer version of NTP or other work-around.

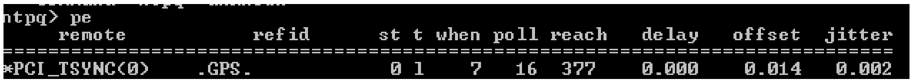
Measuring how well the TSync-PCIe board is maintaining the kernel time (using the NTPQ peers and ntpdate commands)

- Can use the **ntpq -p** (peers command) from NTP running on the linux box to report the offset between the TSync board and the kernel time.
- Can also use the **ntpdate -d** (debug) command to compare the kernel time to other machines.

Email Keith sent to a dealer for his customer (12 Dec 2013) The NTP software running on his machine has two available "functions" called NTPQ and NTPDC that are used to query NTP for information. Specifically, NTPQ has a "peers" command (ntpq -p) which reports the offset between the Linux kernel time versus other time references (such as the TSync-PCIe board for example).

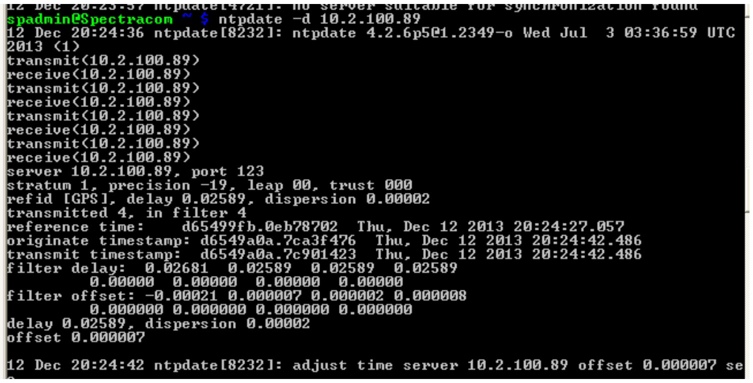
He can use the peers command to see how closely aligned the kernel time is to the TSync board. And, if he has any other NTP servers on the network with this machine or can point to any Internet Time servers, he can specify the peers command to compare the time against them as well. He can use this command without NTP syncing to the timing board (as compared to other references) and then run the command again while the kernel is being synced by the TSync board.

Below is an example response from a peers command, showing the time offset between the TSync-PCIe board and the kernel time (in this example, its 14 microseconds of offset).



```
ntpq> pe
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*PCI_TSYNC(0)    .GPS.            0 l    7   16  377    0.000    0.014   0.002
```

For more information on NTPQ peers command, NTPDC and the **ntpdate –d** command (another command he can use to compare the kernel time to other references, note the –d runs the command in debug mode so that it's not actually setting the time), please refer to sites such as: http://www.eecis.udel.edu/~mills/ntp/html/ntpq.html and http://www.novell.com/coolsolutions/trench/418.html.
Below is an example response from the **ntpdate –d** command comparing the offset between the NTP server (10.2.100.89) versus the kernel time of this machine. In this example, the offset is 7 microseconds (from the last line of the response).



```
12 Dec 20:23:57 ntpdate[4721]: no server suitable for synchronization found
spadmin@Spectracom ~ $ ntpdate -d 10.2.100.89
12 Dec 20:24:36 ntpdate[8232]: ntpdate 4.2.6p5@1.2349-o Wed Jul  3 03:36:59 UTC
2013 (1)
transmit(10.2.100.89)
receive(10.2.100.89)
transmit(10.2.100.89)
receive(10.2.100.89)
transmit(10.2.100.89)
receive(10.2.100.89)
transmit(10.2.100.89)
receive(10.2.100.89)
server 10.2.100.89, port 123
stratum 1, precision -19, leap 00, trust 000
refid [GPS], delay 0.02589, dispersion 0.00002
transmitted 4, in filter 4
reference time:    d65499fb.0eb78702  Thu, Dec 12 2013 20:24:27.057
originate timestamp: d6549a0a.7ca3f476  Thu, Dec 12 2013 20:24:42.486
transmit timestamp:  d6549a0a.7c901423  Thu, Dec 12 2013 20:24:42.486
filter delay:  0.02681  0.02589  0.02589  0.02589
         0.00000  0.00000  0.00000  0.00000
filter offset: -0.00021 0.000007 0.000002 0.000008
         0.000000 0.000000 0.000000 0.000000
delay 0.02589, dispersion 0.00002
offset 0.000007

12 Dec 20:24:42 ntpdate[8232]: adjust time server 10.2.100.89 offset 0.000007 se
```

NTP going to Year 2136 after initial sync

- Refer to SF case 120055
- Appears to be an issue associated with a non-GPS synced TSync board (synced to IRIG with no year being provided, or its year not being hand-set before it dyncs to irig input) still being its boot-up default year of 2000 when the card is in sync (in sync withoutt the year not yet being correct).

The scenario with SF Case 120055 is there are two TSync boards. The TSync master is being hand-set and the TSync slave (installed in the linux box being synced by a more recent version of NTP having our ref clock driver already installed) is synced via IRIG input (no year value present). The Slave goes into sync as soon as the IRIG input from the Master is present, even if the year has not yet been set. So NTP can use the date/year as soon as it asks for it.

Potential solution:

1) Have the IRIG Master output the year in the IRIG datastream (and configure the IRIG Slave to read the year)
2) Have the IRIG Master send the IEEE-1344 extensions in the IRIG output (and have the slave configured to look for it) so that the sync status of the Master can be passed to and prevent the Slave from syncing to the Master, until the Master's date/time/year is correct.

Chronyd (Chrony suite): desire to sync a system running Chrony via a TSync board/Linux driver

- Capability added in new linux driver (~Dec 2020)
- Refer also to “Chrony” in: ..\CustomerServiceAssistance.pdf
- Refer to: http://linux.die.net/man/8/chronyd
- Specific to Fedora: http://docs.fedoraproject.org/en-US/Fedora/18/html/System_Administrators_Guide/chap-Configuring_NTP_Using_the_chrony_Suite.html#sect-differences_between_ntpd_and_chronyd

- **Chrony is replacing NTP in newer linux distributions (**per http://docs.fedoraproject.org/en-US/Fedora/18/html/System_Administrators_Guide/chap-Configuring_NTP_Using_the_chrony_Suite.html)
  “*chrony* is a pair of programs for maintaining the accuracy of computer clocks. **chronyd** is a background daemon program that can be started at boot time.
  - **Example**: CentOS 8 switched from NTP to Chrony

chronyd is a daemon which runs in background on the system. It obtains measurements (e.g. via the network) of the system's offset relative to other systems, and adjusts the system time accordingly. For isolated systems, the user can periodically enter the correct time by hand (using *chronyc*). In either case, **chronyd** determines the rate at which the computer gains or loses time, and compensates for this.”

- Refer to Salesforce Case 239299 for customer requesting Chrony Ref Clock driver for TSync-PCIe boards

Email Keith sent to Apps team (22 July 2020) Regarding existing Salesforce Case 239299, for a US ARMY TSync-PCIe customer desiring to sync Chrony using a TSync board, is there already a Chrony Reference Clock driver available to provide them with?

If there isn’t one already included in the Linux v4.0.0 driver Release Candidate, is it available from the VersaSync software, for instance? As newer Linux versions are swapping out NTP with Chrony, it would be good for us to include both clock drivers in future linux driver releases. Should I create a JIRA ticket requesting this?

I sent this customer info on designing their own Ref Clock driver, but they pushed back asking for one (see below)...

> **(previous note- not sure from when) Per Tim Tetreault**: in order for Chrony to work with the Timing boards, the customer will need to create a reference clock driver like the one we created for NTP. Refer them to the “**refclock_tsyncpci.c**” file in the Linux driver (**tsync**->**linux** -> **ntp**) as an example they can use to create this clock driver in Chrony.
>
> > ***Note: Link to “refclock_tsyncpci.c” file referenced in email above***: *I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\chrony-chronyd (replacing NTP)*

Q Do you know what the server string to put in the conf file for chrony? For NTPD, it was 127.45.0.0 or something similar. I know there was a 45 in there off the top of my head. Is it the same for chrony?

A Email from Katie Schrack Katie.Schrack@orolia.com (2 Nov 2020)

Chrony

- Edit the file */etc/chrony/chrony.conf* and add the following line: **refclock PHC /dev/ptp-tsync poll 0 trust**
- Restart the daemon with: **systemctl restart chrony**
- Check the source availability with: **chronyc sources**

**Real-time (RT) linux (MRG Linux)

- Refer to case 00002331 for Andrew Pearson with Raytheon)

- (See also newer, similar instance of this- Billy Guan (also from Raytheon). Refer to Salesforce case 00003436)
- Will likely need to work with Redhat directly to resolve issues related to the Real Time patch preventing our driver from being installed/able to run (Likely to see a fatal error when running modprobe – more on this further below)

I spoke to our primary timing board engineer. Everything on the Spectracom side looks fine. He recommends that you contact Redhat directly regarding how the RT patch affects the kernel versioning.

We have had a few other customers have the exact same problem until they worked with Redhat directly to resolve some issues associated with this Real Time patching. And if you are paying for Enterprise edition of Redhat, this support is provided by Redhat as part of this purchase. Just let them know you are working with the Spectracom TSync-PCIe driver and have the RT patch installed. They can help you with it from there.

You shouldn't need anything else from us. After working with Redhat to resolve the kernel versioning, you should be all set! But if you happen to need anything else from us as you work with Redhat, or anytime thereafter, please let me know!

A runtime patch ("rt") can be installed onto a standard non-real time linux, to make Linux real-time. The problem this causes with our linux driver is installing this "rt" patch results in two kernel versions being created, instead of just one version, (the earlier un-patched version still exists and the newer patched version is also generated). The build command builds our driver to the earlier un-patched version of the kernel, but when they try to load the driver, it's against the newer version, which isn't the same as it was built for, so an "invalid module format error message is displayed.

The resolution to this issue is for the customer to contact Red Hat directly to assist them with installing a non-real time driver (our driver) onto a patched real time linux OS.

"Invalid module format" error message displayed while trying to install the Linux driver onto a machine that has had this runtime patch installed

Q. (1/20/11) Can you just confirm that whether or not your driver is compatible with Linux real-time OS (MRG)? (Customer – Andrew Pearson from Raytheon - was running 3.6.33.7 MRG)
A. **Reply from DL**- I had to check on this and found we had tested a Red Hat real time setup about a year ago and it did work. It was on an earlier version/patch of Red Hat though.

Email to Billy Guan on 9/26/11 (based on research/info received from Dave Sohn)
Note: Refer to the PDF document that was sent with this email - EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\MRG real-time Linux

I wanted to pass along some information to you, regarding how to make your MRG Linux compatible with the Spectracom TSync-PCIe driver. For your reference, the information below and attached was obtained from an errata report from Red Hat, against their MRG 2.0 patch.

* A missing directory in the kernel-rt-devel packages, which contains header files, prevented third-party modules from being built using only the kernel-rt-devel package. Adding these missing files now allows third-party modules to be successfully built. (BZ#718940).

The fix for this Red Hat issue updates the kernel version to 2.6.33.9-rt31.74.el6rt.x86_64. Please also make sure you have installed the kernel-rt-devel package, which is necessary for module builds. Once you have added the missing Red Hat files, try compiling/running the driver again. Please let us know that adding the files resolved the issue you initially reported!

"modprobe tsyncpci" = rebuilds and installs the driver

Modprobe failures

- "Invalid module format"

Refer to **Real-time linux (MRG Linux)

- Real Time (RT) patch is installed.
- Tim Tetreault recommended they contact Redhat directly to resolve issues with the Real Time patch.

```
root@pascal linux]# pwd
/hadas/install/tsync/linux
[root@pascal linux]# make clean > ./clean.log
[root@pascal linux]# make  > ./make.log
ar: creating libtpro.a
[root@pascal linux]# make install > ./install.log
[root@pascal linux]# modprobe tsyncpci > ./modprobe.log
FATAL: Error inserting tsyncpci (/lib/modules/3.6.11.2-rt33.39.el6rt.x86_64/kernel/drivers/tsyncpci.ko): Invalid module format
```

I've checked that the tsyncpci.ko is regenerated:

```
[root@pascal linux]# date
Mon Oct 14 20:46:48 CEST 2013
[root@pascal linux]# ll /lib/modules/3.6.11.2-rt33.39.el6rt.x86_64/kernel/drivers/tsyncpci.ko
-rw-r--r-- 1 root root 800990 Oct 14 20:45 /lib/modules/3.6.11.2-rt33.39.el6rt.x86_64/kernel/drivers/tsyncpci.ko
```

Solaris driver

Shortcut to TSync-PCIe driver release notes: I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\TSync driver updates

Solaris 11 support with Solaris driver version 3.10

Q Can you please tell us if the TSync Solaris driver supports Solaris-11 systems?
A Email from Ron Dries (18 Feb 2019) I have tested the TSync driver on Solaris 11 and it installed and communicated with the TSync board with no issues.The version of Solaris I tested was: "SunOS solaris 5.11 11.4.0.15.0 i86pc i386 i86pc" I used the 64bit TSync solaris driver.

prtconf -D example

```
System Configuration:  Sun Microsystems  i86pc
Memory size: 131045 Megabytes
System Peripherals (Software Nodes):
i86pc (driver name: rootnex)
    scsi_vhci, instance #0 (driver name: scsi_vhci)
    pci, instance #0 (driver name: npe)
        pci8086,3c00
        pci8086,3c02, instance #0 (driver name: pcieb)
            pci8086,1d74, instance #5 (driver name: pcieb)
                pci8086,1d3f, instance #6 (driver name: pcieb)
                    pci8086,1d68
                    pci8086,1d70
                    pci8086,1d71
        pci8086,3c03, instance #1 (driver name: pcieb)
            pci15d9,1521, instance #0 (driver name: igb)
            pci15d9,1521, instance #1 (driver name: igb)
        pci8086,3c04 (driver name: pcieb)
        pci8086,3c08 (driver name: pcieb)
        pci8086,3c0a (driver name: pcieb)
        pci8086,3c28
        pci8086,3c2a
        pci8086,3c2c
        pci15d9,628
        pci15d9,628
        pci15d9,628, instance #0 (driver name: ehci)
            hub, instance #1 (driver name: hubd)
                device, instance #0 (driver name: usb_mid)
                    mouse, instance #0 (driver name: hid)
                    keyboard, instance #1 (driver name: hid)
        pci15d9,628, instance #1 (driver name: ehci)
            hub, instance #0 (driver name: hubd)
                device, instance #2 (driver name: usb_mid)
                    keyboard, instance #4 (driver name: hid)
                    mouse, instance #13 (driver name: hid)
        pci8086,244e, instance #0 (driver name: pci_pci)
            display, instance #0 (driver name: vgatext)
        isa, instance #0 (driver name: isa)
            motherboard
            asy, instance #0 (driver name: asy)
            asy, instance #1 (driver name: asy)
            motherboard
            motherboard
            motherboard
            pit_beep, instance #0 (driver name: pit_beep)
        pci15d9,628, instance #0 (driver name: ahci)
            disk, instance #0 (driver name: sd)
            disk, instance #1 (driver name: sd)
            disk, instance #2 (driver name: sd)
            disk, instance #3 (driver name: sd)
        pci15d9,628
    ioapics
        ioapic, instance #0
        ioapic, instance #1
    pci, instance #1 (driver name: npe)
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,3ca0
        pci8086,3c46
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
    pci, instance #2 (driver name: npe)
        pci8086,3c01 (driver name: pcieb)
        pci8086,3c02, instance #8 (driver name: pcieb)
            pci8086,3, instance #0 (driver name: ixgbe)
            pci8086,3, instance #1 (driver name: ixgbe)
        pci8086,3c04, instance #9 (driver name: pcieb)
            pci10b5,8617, instance #11 (driver name: pcieb)
                pci10b5,8617, instance #12 (driver name: pcieb)
                    pci10ee,7, instance #0 (driver name: gandalf)
                pci10b5,8617, instance #13 (driver name: pcieb)
                    pci1823,1770, instance #0 (driver name: focalpoint)
                pci10b5,8617, instance #14 (driver name: pcieb)
                    pci1b65,abba, instance #0 (driver name: xel_hx)
        pci8086,3c08, instance #3 (driver name: pcieb)
            pci1ad7,8000, instance #1 (driver name: tsync)
        pci8086,3c28
        pci8086,3c2a
        pci8086,3c2c
    fw, instance #0 (driver name: acpinex)
        sb, instance #1 (driver name: acpinex)
            socket, instance #2 (driver name: acpinex)
                cpu, instance #0 (driver name: cpudrv)
                cpu, instance #1 (driver name: cpudrv)
                cpu, instance #2 (driver name: cpudrv)
                cpu, instance #3 (driver name: cpudrv)
                cpu, instance #4 (driver name: cpudrv)
                cpu, instance #5 (driver name: cpudrv)
                cpu, instance #6 (driver name: cpudrv)
                cpu, instance #7 (driver name: cpudrv)
                cpu, instance #8 (driver name: cpudrv)
                cpu, instance #9 (driver name: cpudrv)
                cpu, instance #10 (driver name: cpudrv)
                cpu, instance #11 (driver name: cpudrv)
            socket, instance #3 (driver name: acpinex)
                cpu, instance #12 (driver name: cpudrv)
                cpu, instance #13 (driver name: cpudrv)
                cpu, instance #14 (driver name: cpudrv)
                cpu, instance #15 (driver name: cpudrv)
                cpu, instance #16 (driver name: cpudrv)
                cpu, instance #17 (driver name: cpudrv)
                cpu, instance #18 (driver name: cpudrv)
                cpu, instance #19 (driver name: cpudrv)
                cpu, instance #20 (driver name: cpudrv)
                cpu, instance #21 (driver name: cpudrv)
                cpu, instance #22 (driver name: cpudrv)
                cpu, instance #23 (driver name: cpudrv)
        used-resources
        iscsi, instance #0 (driver name: iscsi)
        fcoe, instance #0 (driver name: fcoe)
        options, instance #0 (driver name: options)
        pseudo, instance #0 (driver name: pseudo)
        agpgart, instance #0 (driver name: agpgart)
        xsvc, instance #0 (driver name: xsvc)
```

/dev/ (partial example)

tsync0 ("tsync" should be listed once for each installed TSync-PCIe board)

Solaris driver version 2.4.1 DOES support PTP calls

(7 Mar 2013) Denis and Matt thought they found an issue with driver version 2.41 not supporting PTP calls. Update from Denis: No, we fixed that at some point, we just didn't realize it at the time.
The Driver datasheet (which we used as the basis for this in the first place) is incorrect.
Our Solaris driver is a few versions old, but includes the PTP calls.

I will find out if Tim K ever found out about this and can update the datasheet

Q. Scott would like to know if you have a driver for the TPRO-PCI-U-2 that supports Trustin Solaris 8
A. (email from Tim Tetreault 7/28/11)
Keith, I think we have had other customers using Solaris 8 but I am not 100% sure. We don't have a system setup here with Solaris 8 so I can't test it to make sure but I think they should be ok.
You can point them to the link on our web site where they can down load it to try.
Q. Can the card be used with a PowerPC Board (P2020 by Freescale) or just on a x86 Platform.
A. (From Tim Tetreault 6/21/12) The card has been used on PowerPC machines running Solaris but I am not sure about Linux. Are they running a version of Linux on the PowerPC machine?

Solaris driver support for more than one installed TSync board in a system

Email from Tim Tetreault to Matt Loomis (11 Mar 2013) I did a setup this weekend with Solaris and 2 TSync cards, one of which was a PTP card, and both worked fine. So we know the driver will support multiple cards and works fine in Solaris.

**PowerPC

Q. Can the card be used with a PowerPC Board (P2020 by Freescale) or just on a x86 Platform.
A. (From Tim Tetreault 6/21/12) The card has been used on PowerPC machines running Solaris but I am not sure about Linux. Are they running a version of Linux on the PowerPC machine?

In summary

A “**makefile**” file (“mkDriver”) file is supplied in the ***/tsync/examples*** directory of the Linux driver. A “make” command is performed to build the examples. The “makefile” can be either used as-is or the customer can modify it as necessary (potentially breaking the example programs if it’s incorrectly edited).

The path to the makefile (“mkDriver”) is C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API\Src.

The example programs were initially compiled for 32 bit. In order to use them with 64 bit, all of the examples need to be rebuilt for 64 bit.

The make file needs to be modified to use 64 bit examples. The path to the makefile “mkDriver” is C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API\Src. Refer to Section 2.1.3 (page 2-2 of the attached TSync-PCIe driver guide).

Once the examples are built for 64 bit, they should work fine for you.

Desire to use VMWARE (virtual OS/virtual machine)

VMWare allows a “Virtual OS” (such as Linux –known as the Guest OS) to be able to run on top of another OS (such as Windows- known as the Host OS). The problem with this configuration is there is limited capabilities for the Guest OS to be able to access the Host OS’s hardware devices (i.e. the Linux driver on the guest OS can’t talk to the TSync-PCIe board). In summary – this can’t work with two custom drivers that the customer would have to create. See the email below that was sent by Keith (12/21/10):

Regarding the configuration of using VMWare, I needed to work with one of our timing board Engineers to get some additional information about this.   This is the first time we've seen this desired configuration being attempted. The driver install appears to be fine, but the guest OS is not able to directly communicate with the Host's PCI bus. This is a limitation of using VMWare and not a limitation of the TSync-PCIe board itself.

We found the following link that provides additional information about the limitations of using VMware to communicate with a hardware device on the Host OS http://communities.vmware.com/thread/89286. In summary of these findings, it doesn't look like there is a way to allow access to the PCI bus through the virtual OS. The Virtual OS is intended to abstract the hardware for the most part, with a few exceptions for generic operation (such as networking, hard drives, sound, video, etc).

To make this configuration plausible, you would probably need to create some sort of additional custom driver that would abstract the hardware properly to the virtual OS, and then use another custom driver within the virtual OS to talk to that. We're not even sure if this is possible or not. However, we can say that the Spectracom drivers wouldn't be able to provide this capability (just installing the Windows driver wouldn't allow the Linux driver to be able to access the PCI bus because the drivers are not designed to inter-operate with each other). We can provide the source code for the drivers, but this is something that you would have to create, if it’s possible.

Besides creating your own inter-operating custom drivers, our recommendations is to either install the latest version of the Windows driver (the latest version of this driver adds support for Windows 7) and perform the example programs provided in the Windows driver or to install the timing board into a true Linux machine and use the linux driver to perform example programs or API calls, bypassing the use of the Virtual OS altogether.

For your information, if you would like to use the board in a Windows environment, the Windows driver can be downloaded from our website. Please visit:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=37.
Besides example programs and API calls via the driver, the Windows driver also provides a control utility that allows manual control of many of the TSync-PCIe board's functions (Start/All Programs/Spectracom Corp/TSync PCI).

Error: opt err (RC_BUSY)

Q. I tried running the command again but then I received an error of Error: opt err (RC_BUSY)
A. Reply from Paul Myers (1 June 16) Busy error means a command you sent was in progress and took some time to complete. SO you are told – “I can’t do it now”

Interrupts / Interrupt Counter

- **Refer to the TSync-family driver guide for details (1219-5001-0050)** in Arena at https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

Refer to the following document from another company which discusses Linux interrupt performance: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\Interrupts

Interrupt descriptions:

From the TSYNC-PCIe manual
5.9 Interrupts
The host bus has one interrupt line available from the TSync-PCIe. All interrupt sources destined for the host bus are multiplexed on the single interrupt line. All interrupts are masked on startup, but can be unmasked using the host bus interrupt mask register. Whether an interrupt is masked or not, the current state of the interrupt is available by reading the Host Bus Interrupt Status register. All interrupt sources are latched based on an edge transition. All interrupts are cleared in the host bus interrupt status.

Note: The four interrupts below that are in red are only used when a customer is creating their own driver, instead of using a Spectracom-provided driver. The ones in black can be used with the Spectracom driver.

- **1PPS Received**

  This interrupt is driven on the incident edge of the PPS.

- **Timing System Service Request**

  This interrupt is used by the micro to request attention from the local bus.

- **Local / µC Bus FIFO Empty**
  This interrupt is driven when the FIFO from the local bus to the microcontroller bus becomes empty. It is based on the rising edge of the FIFO's empty flag.

- **Local / µC Bus FIFO Overflow**
  This interrupt is driven when the FIFO from the local bus to the microcontroller bus is overflowed. It is based on the rising edge of the FIFO's overflow flag.

- **µC / local bus FIFO Data Available**
  This interrupt is driven when the FIFO from the microcontroller bus to the local bus is no longer empty. It is based on the falling edge of the FIFO's empty flag.

- **µC / local bus FIFO Overflow**
  This interrupt is driven when the FIFO from the microcontroller bus to the local bus is overflowed. It is based on the rising edge of the FIFO's overflow flag.

- **GPIO Input x Event (GPI)**

  This interrupt is driven when the active edge of the GPIO input signal is received (via the four available GPI pins)

  Note: For more info on using the GPI pins to generate interrupts, refer to: Information about the GPI pins (General Purpose Inputs- "GPIO")

- **Time Stamp Data Available**
  This interrupt is driven when the time stamp FIFO goes non-empty. Time stamp data is available in the time stamp FIFO when this interrupt occurs.

- **GPIO Output x Event   (GPO)**

  The interrupt is driven when an event occurs in the GPIO output. An event depends on the mode of operation of the GPIO output. In **Direct mode**, an event is triggered when the output Value in the GPIO output control / status register is changed and creates the active edge selected by the GPIO direct mode output interrupt active edge bit in that same register. This can be used to generate a "software" interrupt by setting the GPIO output appropriately. In **match time** mode, an interrupt is generated whenever the GPIO output high match time or GPIO output low match time registers are enabled and subsequently matched against the current system time. In **square wave** mode, an interrupt is generated whenever the GPIO output generates the active edge as selected by the GPIO output square wave active edge bit in the GPIO output control / status register. This can be used to generate a periodic interrupt at the rate of the square wave.

Note: For more info on using the GPO pins to generate interrupts (using either Match time or Square wave), refer to: Information about the GPO pins (General Purpose Outputs- "GPIO")

Linux kernel thread priorities for interrupt generation

Question from Loredan Neagu (1/19/12)
I would like to ask you what is the thread priority at TSYNC driver level for the interrupt generation?
Reply from Dave Sohn: We do not change any default thread priorities of the kernel.

*****Inherent latentcies associated with interrupts/ desire to very accurately timestamp when an interrupt occurs**

- Refer to SR 6146 in SAP as an example
- Refer also to Timestamping/Timetag in this doc: Time tag (timestamping/Time Stamping/Timetag) / External Event Input

When an interrupt is generated, there can be variable latencies associated with both the PCIe bus and Operating System as to the response to an interrupt that was generated.

For example, if performing a HW_GetTime eeach time an interrupt is generated to time stamp when the interrupt occurred: The interrupt is "accurate" and the HW_GetTim is an accurate time read when it occurs. But there will be a variable delay between wheh the system receives/processes the interrupt and when the HW_GetTime call is actually performed. so the timestamp of the interrupt will not likely be very accurate

A MUCH more accurate method to timestamp interrupts is to look a GPO pin to a GPI pin and use TimeStamping of the GPI to timestamp when the event was received on the GPI pim.

Email Keith Sent (associated with SR 6146) To begin, even using the HW_GetTime call to read the time when an interrupt occurs is not providing an accurate time stamp for when the interrupt actual occurred. The interrupt and the time read are both very accurate, but there are inherent variable latencies in both the PCIe bus and the Operating System between when the interrupt occurs and the "trigger" of the HW_GetTime call being performed. So an alternate method needs to be used to obtain the time stamps for when the interrupt actually occurs

We recommend you instead feed back ("loop") the output pulse(from the GPO pin) back into the General Purpose inputs (GPI pin) and use this input event to generate a timestamp when the input is received (this is referred to as "timestamping" or "time tag"). The timestamp for the input event will be latched and saved onto the FIFO (First In / First Out) buffer. Your software can read the timestamp that was generated from the FIFO buffer without losing any of the accuracies.   the FIFO buffer can actually hold several timestamps before the oldest timestamp is deleted (eventually, if the timestamps aren't read out, the oldest ones are deleted to make room for the most recent timestamps)

TimeTag/Timestamping is discussed in Section 5.2 of the TSync Driver Guide (attached for your convenience). In summary: In addition to looping the signal from the GPO pin back into to the first available GPI pin (referred to as "GPI 0"):

The Timestamping function needs to be enabled each time the Tsync board is powered up, using the call: HW_SetTsEnable 0 1   .
The GPI pin receiving the input signal (in this case, "GPI 0") needs to be enabled using the call GI_SetTsEnable 0 0 1
The individual Timestamp(s) which was generated can then be read out of the FIFO buffer as either one at a time, or all

at once (if more than one input event has been received since last reading out any timestamps).
The most recent timestamp (the oldest event) in the buffer can be read out by itself using the HW_GetTsSingle (one timestamp is returned)
All timestamps that are currently in the FIFO can be read out all at the same time using the call: HW_GetTsData (all timestamps are returned).
The number of interrupts that have occurred since the interrupt counter was last cleared can be read using the call HW_GetTsCount 0 1
The interrupt counter can be cleared using the call HW_SetTsClear 0 1

Below is an example of customer code that was written to timestamp an input event on the “GPI pin 0 (each time a rising edge was detected on the GPI 0 input pin, a timestamp is sent to the FIFO). This code also clears the interrupt counter and then reads the counter to see if an interrupt had been received yet.

```
sprintf( fullDevName, "%s%s%d", DEVICE_ROOT, devName, devIdx );
err = TSYNC_open(&hnd, fullDevName);
en = 1;
err = TSYNC_HW_SetTsEnable(hnd, en);
pin = ID_PIN_0;
err = TSYNC_GI_SetTsEnable(hnd, pin, en);
edge = EDGE_RISING;
err = TSYNC_GI_SetEdge(hnd, pin, edge);
src = TMSTMP_SRC_GPI_0;
err = TSYNC_HW_SetTsClear(hnd, src);
for (EVER) {
err = TSYNC_GI_GetValue(hnd, pin, &j);
err = TSYNC_HW_GetTsCount(hnd, src, &k);
printf("Sleeping %d %d %d\n", n,j,k);
sleep(1);
```

Also attached is a copy of the TSync’s Application Guide. Section 2.1 lists all of the System Clock time registers that are all “latched” when an timetag/timestamp is performed so that the time of the event can be captured/sent to the FIFO buffer, providing sub-millisecond time resolution for when the event occurred.,

Enabling interrupts:

The best reference for Interrupt operation is section 5.1 of the TSync-PCIe driver guide (1219-5001-0050): in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

To begin, keep in mind there is only one interrupt, with the same interrupt being able to be generated based on a few different types of events (this is the “intType value). The waitFor call is defined with which of these event types that will release it (this is the same “intType” value). If the interrupt was generated for any other reason that the reason its programmed to look for (intType), the waitFor ignores that particular interrupt.

As for interrupt generation, the part that typically causes the most trouble for users is the bEnable” value in the setIntMask is actually the opposite value of what is often assumed. The enable in this call doesn’t directly **enable** the interrupt. Instead, it enables the “mask” which **disables** the interrupt while this mask remains enabled. Interrupts can start to be generated by **disabling** the mask for only the type(s) of interrupt you wish to generate.

Here is a good summary which helps to clarify this:

- Interrupts are **enabled** by **disabling the Interrupt mask** of the desired interrupt type (intType).
- Interrupts are **disabled** by **enabling the Interrupt mask** of the desired interrupt type (intType).

All interrupts are **masked** by default at each power-up, so no interrupt is generated until the mask has been disabled for the reason you wish the interrupt be generated for (and not until that event occurs thereafter).

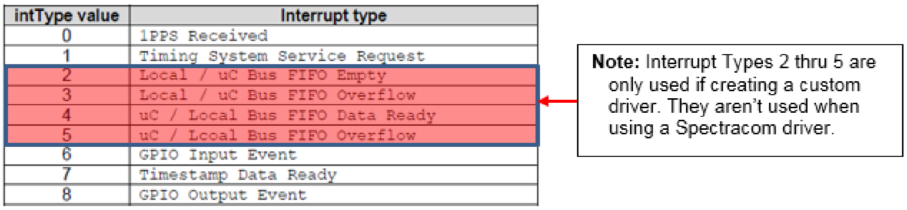
HW_SetIntMask (TSYNC_HW_SetIntMask 0 <intType> <index> <bEnable>) (see breakdown further below based on text colors)

- ➢ This API call used to enable/disable interrupt of a specific type (such as GPI in , GPO out, or 1PPS occurring)

Example: ***HW_SetIntMask 0 8 0 0*** (Disables the GPO output interrupt mask, which enables the interrupt for the GPO 0 output pin)

intType:

Use the desired value in the following table, based on desired reason for interrupts



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

Note: Interrupt Types 2 thru 5 are only used if creating a custom driver. They aren’t used when using a Spectracom driver.

Note: To enable interrupts based on an GPI input pin. use the I**ntType** value “**6**” (“GPIO Input event”) in the **SetIntMask** call (as shown in the above table)

Index:
The interrupt index info, used for certain interrupts (such as a 0 for GPO **0**, or GPI **0** for examples)

bEnable

- The **en** variable on this line should be set to **0** (FALSE) in order to **disable** the interrupt mask, thereby **enabling** the interrupt.

- The **en** variable on this line should be set to **1** (TRUE) in order to **enable** the interrupt mask, thereby **disabling** the interrupt. (note this is the factory default power-up state for all interrupts, for all interrupts to be disabled by default)

Please let me know if this info allows timestamps to start being present in the FIFO buffer (as I expected it will 😊)!!!

HW_GetIntTS

- ➢ From the TSync Windows driver **version 3.2.1** Release notes
  - "**The HW_GetIntTS** command now returns a sub-second accurate time for Windows OS's later then Windows 7. "

Interrupt Counter

- ➢ Use **HW_clrIntCnt** to reset the Interrupt counter
- ➢ Use **HW_GetIntCnt** to read the Interrupt counter

Email from Tim Tetreault (Where "8" is for interrupts based on the GPO pin)
Before enabling the interrupt, reset the interrupt counter by using the command:
./HW_ClrIntCount 0 8 0

When an interrupt has been seen, they can us the following command to read the interrupt count:
./HW_GetIntcount 0 8 0

HW_waitFor and HW_waitForTo (blocking calls)

If desired, there is an available blocking call that can be used to halt the application code, until this interrupt occurs. This blocking call is the **waitFor** command. When the waitFor command is used in application software, and its desired to wait for the 1PPS received interrupt, the full waitFor syntax is **waitFor 0 0 8** (for the first TSync board installed in the system and to wait for the "GPIO" interrupt to occur) before allowing the application software to proceed.

To begin and just so you are aware (as of ~May 2016) there are now two different variants of the "**waitFor**" call. These are the original **waitFor** call and the newer "**waitForTo**" call:

A) **HW_waitFor** is the original call/example program used to stop application software from running, until a specified type of interrupt has occurred (such as to stop the application software, until an event timestamp has occurred, for instance).

Note: This particular call has a hard-set timeout that releases the "hold" after about 100 seconds, if the interrupt hasn't been detected prior to the end of this timeout period.

Note about "TSync_waitFor" versus "HW_WaitfForInt":

Modifed email from Dave Sohn 1/18/12

- **TSYNC_waitFor** is the driver API call.
- **HW_WaitForInt** is the example program, which shows how to use TSYNC_waitFor

B) **HW_waitForTo** is a variant/modification of the original waitfor call call/example program used to stop application software from running, until a specified type of interrupt has occurred (such as to stop the application software, until an event timestamp has occurred, for instance).

This newer call (available ~May 2016), unlike the original waitfor call, can wait indefinitely for an interrupt to occur (the original "waitFor" call has an automatic timeout that will occur if the expected interrupt doesn't occur within a certain time-frame). Or, this newer call can be custom-configured to expire after a specified period of time (configured in milliseconds).

Driver versions with the newer HW_WaitForTo call variant available:

- **Windows driver:** the HW_WaitForTo call is available in Windows driver **version 3.2.1** (ECO 1039, Oct, 2016) or higher
- **Linux driver:** the HW_WaitForTo call is available in Linux driver **version 3.3.1** (ECO 1223, Apr 2017) or higher

A) The original "HW_waitfor" call

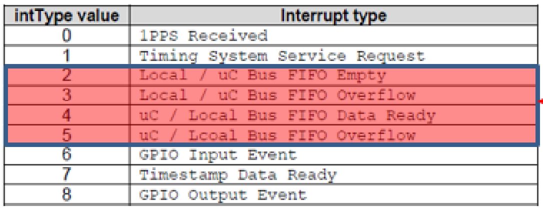
(TSYNC_waitFor 0 <intType> <index> **see breakdown futher below)**

➢ Blocking call to wait for specified interrupt (such as GPI, GPO or 1PPS)

Example: TSYNC_waitFor 0 8 0 (Call in your application software to wait for an interrupt on GPI pin "0").

<intType> (reason for the interrupt being generated)

Use the desired value in the following table, based on desired reason for interrupts



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

Note: Interrupt Types 2 thru 5 are only used if creating a custom driver. They aren't used when using a Spectracom driver.

Index:
The interrupt index info, used for certain interrupts (such as a 0 for GPO **0**, or GPI **0** for examples)

bEnable

- The **en** variable on this line should be set to **0** (FALSE) in order to **disable** the interrupt mask, thereby **enabling** the interrupt.
- The **en** variable on this line should be set to **1** (TRUE) in order to **enable** the interrupt mask, thereby

disabling the interrupt. (note this is the factory default power-up state for all interrupts, for all interrupts to be disabled by default)

Email from Tim Tetreault (Where "8" is for interrupts based on the GPO pin)
Once the interrupt is running, they can wait for it using the command:

```
./HW_WaitforInt 0 8 0
```

TSYNC_COMM_ERR associated with the waitFor blocking call

- This particular call has a hard-set timeout that releases the "hold" after about 100 seconds, if the interrupt hasn't been detected prior to the end of this timeout period.

The following is returned if the timeout occurs before interrupt is generated:

```
Waiting for GPIO Input 1........
ioctl returned 1 (c:\tsync\windows\tsync\lib\tsync_driver_interface.c:204)  Error: communication error
(TSYNC_COMM_ERR).
```

Waitfor FAQs

Q. ("**Waitfor" Timeout**) I just thought of another TSync question: if I call the TSYNC_waitFor routine in the driver, it blocks until an interrupt of the specified type occurs. Is there any way to cancel the call? There is no timeout, so this routine has the possibility of causing the calling thread to hang if the interrupt never occurs.

A Excellent question! I hadn't thought about this one before, but it certainly makes sense to think about this condition possibly happening. Though it's not mentioned in our documentation, there is a built-in time-out in this API call to prevent this condition from occurring. The time-out value is based on a set number of Operating System clock ticks, so it's not necessarily an exact, defined amount of time from one PC to the next, but a time-out does exist. The timeout is pre-defined as 100,000 OS ticks.

The waitFor call is one of the example programs included with the TSync-PCIe driver. If you wanted, you could run the example program to see that it does time-out after a period of time and it would also give you an idea how long of a duration 100,000 OS ticks would be. This pre-defined value is not adjustable with an API call. However, if it needed to be modified (shortened or lengthened) for your particular application, it can be modified. Just let us know and we can give you additional information on how to modify it.

A1. Response back from customer: Doing the subtraction, that's ~99.9879 seconds. A second test was 99.9878 seconds. Sounds like 100 to me.

A.2 email from a different customer not getting the interrupt because no signal is being applied (he is seeing a 10 second timeout
When calling TSYNC_WaitFor to wait for an interrupt from one of the GP input lines and the interrupt does not occur within 10 seconds the **API routine returns with an error 14 and the API itself prints to the screen "ioctl returned 1 (c:\tsync\windows\tsync\lib\tsync_driver_interface.c:204)**". Questions: Is this normal operation? Is the 10 second interrupt wait time programmable through the API? How can I shutoff the error message being sent to console from within the API DLL.

(3 Mar 16 KW- update for the email above): customer didn't want any changes via any custom or factory TSync driver changes . So Dave Sohn suggested their application software have a faster timeout than the waitfor and to have it kill the block before the waitfor times-out, thus preventing the waitfor timeout error needing to be asserted.

B) TSYNC_WaitForTo (TSYNC_waitForTo 0 <intType> <index>) (see breakdown below)

- Blocking call to wait for specified interrupt (such as GPI, GPO or 1PPS)
- Can wait indefinitely with no time-out period if an interrupt isn't generated.
- Created for Jerry Grasso with Harris (May 2016)
- Just a driver change (no firmware change to the board is required)

Email from Keith (22 Nov 16) Thanks very much for your reply!! Here's the reason and solution for you ☺…

FYI- for the longest time, there was only one waitFor call available. But it had an automatic time-out period that was hard-set/not configurable (the timeout was based on a number of OS clicks, which ends up being about 100 seconds).

Based on a customer request to have it be able to wait indefinitely (without "breaking" the original waitfor call for those already using it) we have more recently (May 2016 time-frame) added to the TSync-PCIe Windows driver (starting in driver version 3.2.1) a new variant of the original waitfor call that can be held indefinitely. This is the HW_waitForTo call/example program.

Here is the syntax of this newer **waitforTo** call:

A) If you are using the TSync Windows driver:

The version 3.2.1 Windows 3.21 driver can be downloaded at no cost from our website. Please visit us at: https://spectracom.com/support/tsync/tsync-pcie-support (Scroll down to "**TSync Windows driver 32-bit driver**" or "…**64-bit driver**")

Here is the syntax of this newer **waitforTo…** call in this latest Windows driver

TSYNC_waitForTo 0 <intType> <index> (use the same intType and index value you were using earlier with the original waitfor call)

B) If you are using the TSync Windows driver:
The waitForTo call is not yet been added to the TSync series Linux driver. However, I suspect it will be included in the next release of this driver. I don't have an expected date for the next release of this driver. But we periodically release new versions, especially due to compatibility with newer kernel versions as they become available**.**

If you are using the linux driver, let me know and I will flag your record in our Service database to let you know when the next version of the Linux driver which includes this newer call is released/made available.

Linux/Windows Driver support for the waitForTo call

A) Windows driver

- This call was added to **Windows driver version 3.2.1** (not available in earlier versions of the Windows driver)

From the Windows driver v3.2.1 release notes

"Added **waitForTo** API to support a time out parameter when waiting for an event (-1 for infinite wait, or numeric value for time out in **milliseconds**)"

B) Linux driver

- **waitForTo** call was added in Linux **driver verson 3.3.1**

From the Linux driver v3.3.1 release notes

Added waitForTo API to support a timeout parameter when waiting for an event.
(-1 for infinite wait, or numeric value for timeout in ***milliseconds***)

- ~~As of linux **driver version 3.2.0**, this newer API call has not yet been added to the driver~~

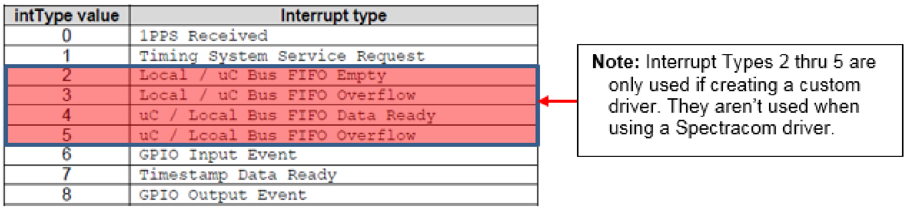
~~**Email from Dave Sohn (23 Nov 16)** It is not yet available in the released Linux driver. The Windows driver is the~~

~~only one that supports it currently. I believe it is implemented in the Linux driver codebase. I don't think it would take muh to release a 3.2.1 Linux driver, but Ron or Tim would know better.~~

Example: TSYNC_waitForTo 0 8 0 (Call in your application software to wait for an interrupt on GPIO "0").

<intType>

Use the desired value in the following table, based on desired reason for interrupts



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

Note: Interrupt Types 2 thru 5 are only used if creating a custom driver. They aren't used when using a Spectracom driver.

Index:
The interrupt index info, used for certain interrupts (such as a 0 for GPO **0**, or GPI **0** for examples)

bEnable

- The **en** variable on this line should be set to **0** (FALSE) in order to **disable** the interrupt mask, thereby **enabling** the interrupt.
- The **en** variable on this line should be set to **1** (TRUE) in order to **enable** the interrupt mask, thereby **disabling** the interrupt. (note this is the factory default power-up state for all interrupts, for all interrupts to be disabled by default)

Email from Tim Tetreault (Where "8" is for interrupts based on the GPO pin)
Once the interrupt is running, they can wait for it using the command:
./HW_WaitforInt 0 8 0

Q What are the units for the *timeout* parameter and what is the forever value**?**
The documentation does not cover the new function. Is this value the same as a Windows wait, in milliseconds and a signed value of -1 means INFINITE wait (0xFFFFFFFF)**?**
A Reply from Dave Sohn (29 Apr 16) You are correct. A -1 indicates an infinite wait. Otherwise, the values are in milliseconds.

Example scenarios for using interrupts

Note: Refer to Example below for:

- **A)**= Interrupt to occur once-per-second
- **B)=** Interrupt to occur at a specified frequency (such as every 500ms, 1kHz, 1kHz, etc).
- **C)=** Interrupt to occur when timestamps are generated (via event rececived on a GPI pin)

A) Desire to generate a once-per-second interrupt (coincident with the on-time point)

To generate a 1PPS interrupt, perform either of the following:

- Enable the "**1PPS Received**" interrupt (the easier of the two methods) (refer to "**1**" below)

  or

- Configure a **GPO pin to output 1Hz square wave** (refer to "**2**" below)

Attached are copies of both the TSync manual and driver guide, which discuss interrupt generation. There is a 1PPS interrupt available. It just needs to be unmasked. These two documents discuss this capability. Refer to Section 5.1 of the driver guide and Section 5.9 of the manual for information on interrupt generation.

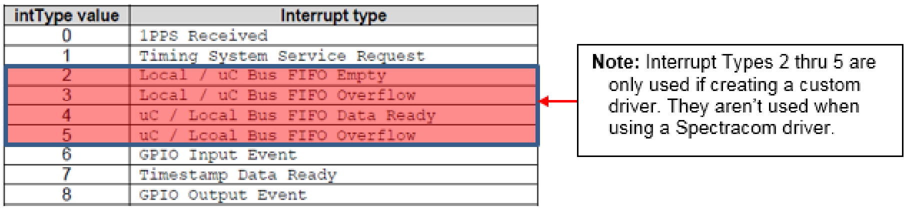
FYI:

- Interrupts are **enabled** by **disabling the Interrupt mask** of the desired interrupt type (intType).
- Interrupts are **disabled** by **enabling the Interrupt mask** of the desired interrupt type (intType)
- The API call/example program used for interrupt generation is: **HW_SetIntMask.**
- The actual call to generate an interrupt each second is as follows: **HW_GetIntMask 0 0 0** <enter> (as broken down below).

HW_SetIntMask <board handle> <**intType**> <**index**> <**benable**>

intType

The "intType" value in this command is selected from the following table. (Value "0" is selected for a once-per-second interrupt to occur).



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

Note: Interrupt Types 2 thru 5 are only used if creating a custom driver. They aren't used when using a Spectracom driver.

Index:

The interrupt index info, used for certain interrupts (such as a 0 for GPO **0**, or GPI **0** for examples)

bEnable

- The **en** variable on this line should be set to **0** (FALSE) in order to **disable** the interrupt mask, thereby **enabling** the interrupt.

- The **en** variable on this line should be set to **1** (TRUE) in order to **enable** the interrupt mask, thereby **disabling** the interrupt. (note this is the factory default power-up state for all interrupts, for all interrupts to be disabled by default)

waitFor blocking call

- refer to: HW_waitFor and HW_waitForTo (blocking calls)

1) Enable the "1PPS Received interrupt

Email Keith sent to a customer (to enable the 1PPS received interrupt)
An interrupt can be generated at the top of each second. This particular "interrupt generator" is known as the "**1PPS Received**" interrupt (designated by "intType value "0"). Attached for your reference is a copy of the TSync-PCIe driver guide. Section 5.1 (starting on page 5-238) discusses how to enable interrupts to be generated for different situations, such as each time the system PPS occurs.

In summary of the Driver guide, generating this interrupt consists of **disabling** the interrupt mask for "1PPS Received" (all interrupt masks are **enabled** by default). To disable the 1PPS Received mask (thereby enabling the "1PPS Received" interrupt), use the following call: **HW_SetIntMask 0 0 0 0** (where the first 0 is for the first/only TSync board installed in the system, the second 0 is for the "1PPS Received" interrupt, the third 0 is the index value and the last 0 disables the interrupt mask).

With the 1PPS interrupt now being generated at the very top of each second, one available option is to use the available "**waitFor**" API blocking call, to halt the application software until the "1PPS Received" interrupt has occurred. The API call to wait for this interrupt before the application software proceeds in **./HW_WaitforInt 0 0 0** (where the first 0 is for the first/only TSync board installed in the system, the second 0 is to wait for the "1PPS Received" interrupt and the third 0 is the index value).

2) Configure a GPO pin to output 1Hz square wave

Email from Tim Tetreault To use the GPIO to generate an interrupt, do the following:

1. Setup Pin0 to a "square wave": **./GO_SetMode 0 0 2**
2. Enable Pin0 output: **./GO_SetEnable 0 0**

B) Desire to generate an interrupt at a specified frequency rate (other than once-per-second)

Attached are copies of both the TSync manual and driver guide, which discuss interrupt generation. These two documents discuss this capability. Refer to Section 5.1 of the driver guide and Section 5.9 of the manual for information on interrupt generation.

There is a GPIO interrupt available. A GPO output pin can be configured for "squarewave" mode in order to generate a squarewave output. An interrupt is then generated at the active edge of each sqaurewave. The rate of the interrupt is controlled by the frequency of this square wave output.

FYI:

- Interrupts are enabled by disabling the Interrupt mask of the desired interrupt type (intType).

- Interrupts are disabled by enabling the Interrupt mask of the desired interrupt type (intType)

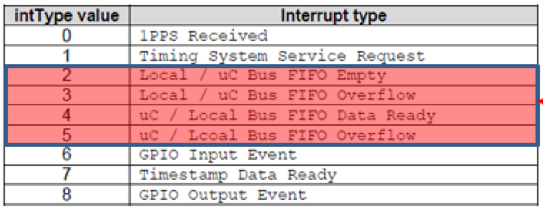
The API call/example program used for interrupt generation is: **HW_SetIntMask**

The actual call to generate an interrupt at a specified interval using the GPIO is: **HW_SetIntMask 0 8 0 0** <enter> (as broken down below):

HW_SetIntMask <board handle> <intType> <index> <benable>

intType

- The "intType" value in this command is selected from the following table. (Value "8" is selected for the GPIO interrupt to occur).



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

Note: Interrupt Types 2 thru 5 are only used if creating a custom driver. They aren't used when using a Spectracom driver.

Index:
The interrupt index info, used for certain interrupts (such as a 0 for GPO **0**, or GPI **0** for examples)

bEnable

- The **en** variable on this line should be set to **0** (FALSE) in order to **disable** the interrupt mask, thereby **enabling** the interrupt.
- The **en** variable on this line should be set to **1** (TRUE) in order to **enable** the interrupt mask, thereby **disabling** the interrupt. (note this is the factory default power-up state for all interrupts, for all interrupts to be disabled by default)

Example usage

To generate an interrupt at a desired interval, the GPIO Output Event Interrupt (type 8) should be used. The rate of the interrupt is controlled by the rate of the square wave.

3. To use the GPIO to generate an interrupt, do the following:

1) Setup Pin0 to a "square wave":
2) ./GO_SetMode 0 0 2
3) Enable Pin0 output:
4) ./GO_SetEnable 0 0 1

To configure the interrupt rate, us the following command:

"GO_SetSqWave A B C D E F G"

Where:
A = card index
B = I/O pin
C = offset from 1pps in nsec
D = period
E = scale for period 0 = nsec, 1 = usec

F = pulse width in nsec
G = 0 is neg pulse, 1 is pos pulse

So for example, if you want GPIO “0” to generate interrupts at a rate of **1kHz** :
./GO_SetSqWave 0 0 0 1000 1 10000 1

For **10kHz:** /**GO_SetSqWave 0 0 0 100 1 10000 1**

Desire to generate a periodic interrupt every 10 msec

Email from Dave Sohn (2 Oct 13) Using a general purpose output, you can set up a 10ms period signal and receive interrupts every time that signal triggers.

C) Desire to generate a periodic interrupt every 500 msec (using a squarewave on GPO pin)

One of the GPIO pins (such as GPIO pin 0) needs to be configured for 500ms square wave output (per your desired ½ second interrupt interval).

To configure the GPIO pin 0 for a 500msec square wave, 1 msec pulse width, positive-going pulse, use the following API calls (or example programs):

./GO_SetMode 0 0 2 (GPIO “0” set to square wave mode)
./GO_SetSqWave 0 0 0 5000000 1 1000000 1 (GPIO “0”, 0 offset, 500msec period, usec scale, 1msec pulse width, positive pulse)
GO_SetEnable 0 0 1 (GPIO “0” enabled)
2. Desire to generate a periodic interrupt every 1kHz (using a squarewave on GPO pin)

So for example, if you want GPIO “0” to generate interrupts at a rate of 1kHz:
./GO_SetSqWave 0 0 0 1000 1 10000 1

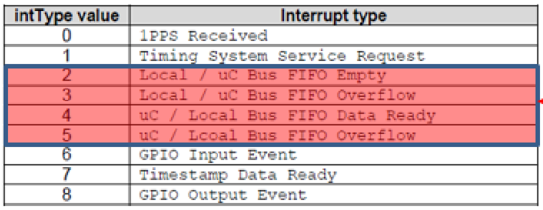
D) Desire to generate a periodic interrupt every 10kHz (using a squarewave on GPO pin)

So for example, if you want GPIO “0” to generate interrupts at a rate of 10kHz:
./GO_SetSqWave 0 0 0 100 1 10000 1

To generate interrupts, the interrupts need to "unmasked", using the HW_SetIntMask API call (set to not true, to unmask interrupts- default configuration is the interrupts being “masked”). With interrupts now being generated (unmasked), you application software can now use the TSYNC_waitFor blocking call for your software to wait for a specific type of interrupt to occur.

intType

The “intType” value in this command is selected from the following table. (Value “8” is selected for the GPIO interrupt to occur).



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

Note: Interrupt Types 2 thru 5 are only used if creating a custom driver. They aren’t used when using a Spectracom driver.

Index:

The interrupt index info, used for certain interrupts (such as a 0 for GPO **0**, or GPI **0** for examples)

bEnable

- The **en** variable on this line should be set to **0** (FALSE) in order to **disable** the interrupt mask, thereby **enabling** the interrupt.
- The **en** variable on this line should be set to **1** (TRUE) in order to **enable** the interrupt mask, thereby **disabling** the interrupt. (note this is the factory default power-up state for all interrupts, for all interrupts to be disabled by default)

waitFor blocking call

TSYNC_waitFor 0 8 0 (Call in your application software to wait for an interrupt on GPIO “0”).

Refer to: HW_waitFor and HW_waitForTo (blocking calls)

C) Desire for Interrupts to be generated when timestamps are taken (Timestamp data ready)

Email from Dave Sohn:
It is possible to be interrupted when timestamps are taken. He could then disable timestamping, read one timestamp, clear the timestamps, reenable timestamping and then wait for the next event interrupt. Below are the APIs to he would use:

Setting active edge for input events for a given general purpose input

```
TSYNC_GI_SetEdge(
   TSYNC_BoardHandle hnd,
   ID_PIN index,
   EDGE edge)
```

Where:
0 = Falling edge
1 = Rising edge
2 = Both edges (Note: As also mentioned in Triggering on event input, we don’t support triggering on both edges. An “opt” error message is displayed when selecting this value).

Enabling/disabling timestamps on input events for a given general purpose input

```
TSYNC_GI_SetTsEnable(
   TSYNC_BoardHandle hnd,
   ID_PIN index,
   int bEnable)
```

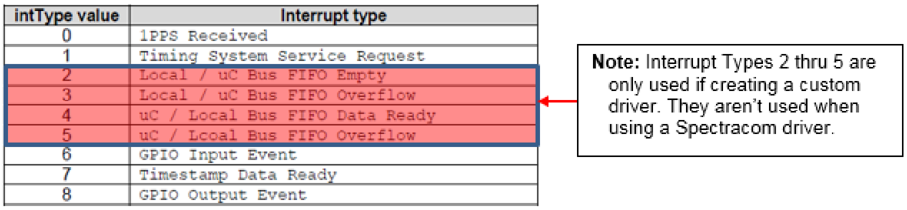
Enabling/disabling interrupt mask for the timestamp event

```
TSYNC_HW_SetIntMask(
   TSYNC_BoardHandle handle,
   INT_TYPE intType,
   unsigned int index,
   int bEnable)
```

Note: Enabling interrupts can be confusing. In order to **enable** the interrupt, you need to **disable** the mask. The following is incorrect code for enbabling interrupts

err = TSYNC_HW_SetIntMask(hnd, src, index, **1**);

intType
The “intType” value in this command is selected from the following table. (Value “8” is selected for the GPIO interrupt to occur).



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

Note: Interrupt Types 2 thru 5 are only used if creating a custom driver. They aren't used when using a Spectracom driver.

You should be calling the following to disable the mask and enable the interrupt:

```
err = TSYNC_HW_SetIntMask(hnd, src, index, 0);
```

Waiting for the timestamp event input

```
TSYNC_waitFor(
   TSYNC_BoardHandle handle,
   INT_TYPE intType,
   uint32_t index)
```

Enabling/disabling hardware timestamping

```
TSYNC_HW_SetTsEnable(
   TSYNC_BoardHandle handle,
   int bEnable)
```

Retrieve a single timestamp

```
TSYNC_HW_GetTsSingle(
   TSYNC_BoardHandle handle,
   TMSTMP_SRC source,
   TSYNC_HWTimeObj *pObj)
```

Clear remaining timestamps in FIFO

```
TSYNC_HW_TsClear(
   TSYNC_BoardHandle handle,
   TMSTMP_SRC source)
```

O goes non-empty. Time stamp data is available in the time stamp FIFO when this interrupt occurs.

NOTE: The rest of the response refers to sections in the Driver Manual.
Interrupt usage is setup by unmasking the interrupt using the TSYNC_HW_SetIntMask API (4.2.213). The customer would then wait on an interrupt utilizing the TSYNC_waitFor API (4.2.5).

Assuming they would use the General Purpose Outputs (GPO) in square wave mode to generate periodic interrupts, the customer would need to set the output mode using the TSYNC_GO_SetMode API (4.2.191). Then they would need to set the square wave configuration using the TSYNC_GO_SetSquareWave API (4.2.197). Finally, they would enable the GPO using the TSYNC_GO_SetEnable API (4.2.188).

After all this, the customer would receive an interrupt based on the period they set the GPO to.

TSYNC_hwSetIntMask

The index of TSYNC_hwSetIntMask is used for interrupt types that have more than one source, like GPIO. The index is then used to indicate which instance of that interrupt type to mask/unmask. If you wanted to unmask the GPI input 1, for example, then your interrupt type would be GPI and your index would be 1 to indicate general purpose input 1.

waitFor blocking call

If desired, there is an available blocking call that can be used to halt the application code, until this interrupt occurs. This blocking call is the **waitFor** command. When the waitFor command is used in application software, and its desired to wait for the 1PPS received interrupt, the full waitFor syntax is **waitFor 0 0 8** (for the first TSync board installed in the system and to wait for the “GPIO output event” interrupt to occur) before allowing the application software to proceed.

Interrupt FAQs

Q. Is there a way to determine how many interrupts we have missed from one call to ‘wait _*for*_interrupt’ to the next?

A. Not through the driver itself. You may be able to determine it through the OS. In linux, the OS keeps a running total of the interrupts generated by any interrupt line in /proc/interrupts. Just be aware that it is collecting all interrupts on that line, so if it is shared with other devices, those will be shown as well. Also, all interrupts from the TSync will be aggregated in the tally.

Q. Interrupts don’t appear to be generated at exactly the same moment, every time. “Also we have noticed inconsistencies in the time that our application is “woken up” after TSYNC_waitFor. This is a deal breaker for us. We run the interrupts at 128Hz. The interrupt period is 7812500 nanoseconds. With the use of some debug code we have timed the “wake ups” and we see a certain pattern. The cards generally interrupt at precisely the right period, but around once a second, the interrupt takes longer. I typically see it take somewhere in the 9000000 nanoseconds with the following interrupt being in the 5000000 nanoseconds. The average of these two interrupts is always precisely the right period but that isn’t good enough. Falling behind on the interrupts means certain functionality is postponed which defeats our realtime simulation.”

A. Regarding the report that the TSync interrupts don’t appear to be happening “when expected”, I have some information for you that I hope may be able to help you with this condition:

The TSync-PCIe board’s interrupts themselves are occurring exactly when they are supposed to occur, based on the configuration of the GPO output. The hardware interrupts are occurring “real time” but only at the kernel level. Unfortunately, factors beyond the control of the TSync-PCIe board and its associated driver may prevent the application software threads from being able to wake up at the same moment that the interrupt occurred. This is partially because Linux is not a Real Time operating system, and can also be affected by the loading of the PC. The OS may have many processes and other interrupts occurring at the same moment of a TSync interrupt occurring that will prevent the software from being able to handle the interrupt in a controlled time-frame each time the interrupt occurs. The application software thus appears to be erratic, but at the hardware level, the “average” time for the interrupts to occur is the expected times for the interrupts to be occurring. At the application software level though, the thread may wake-up sooner than it did previously, or it may take longer to wake-up than previously, depending on what else is occurring on the machine at that particular moment. At this level, the timing cannot be precisely controlled.

As for possible solutions to help improve the “real time” operational needs of your application threads, extensions may be available for the Operating System to make it more resemble a “Real Time Operating System”. Changing the priorities of other application software programs that may be running at the same time may also help. Networking is a large generator for system interrupts, which can significantly affect how long it takes for the interrupts to be handled by your application software, so if you can somehow minimize the network activity, this may help as well.

In summary, the board is generating the interrupts at exactly the right time, every time. However, your application software will not likely be able to react to the interrupts and wake-up the threads at the same exact interval that the interrupts are occurring, due to factors beyond the control of the card. The wake-ups may occur earlier than normal or may take longer than normal. But the average time is as expected, based on the configuration of the GPO.

***Issues with/Troubleshooting Interrupt Generation

1. **Interrupts stop being generated after about 5 minutes (or some other similar value)**

   A. More than one interrupt routine being performed/CPU affinity

   Interrupts initially being generated, but then just stop (interrupt counter stops incrementing, waitFor call stops working) even though the GPO to GPI input signal still being generated.

   - Refer also to Salesforce Cae 275650
   - Refer to Salesforce Case 253946
   - Refer also to earlier Salesforce 232056 and associated **JIRA Ticket BPS-41** (excerpt below in blue):
     - This customer found their dual interrupts stopping was due to their dual core processor (each core was handling one of the interrupts. Our driver apparently expects everything to be performed on the same core.

   Michael responded they figured out what was causing their "missed interrupts" (a bit over my head)….

   Keith,
   We found the problem.
   As we know the Tsync driver is designed such that to process multiple sources of interrupts with the provisions provided (WaitFor blocking call) one has to spin up multiple threads to handle each potential Tsync interrupt event source. Ok, no big deal.

   However, while the driver is supposedly thread safe, it has the assumption the all related task/threads / processes are executing on the same CPU core, thus CPU Affinity is the issue.

   In our sample program set for realtime priorities with one task at higher service priority, Windows allocates the 1PPS event tread to Core 1, while the 200Hz allocated to Core2. This explains the random corrupted stacks & heaps as the two threads on separate cores complete at different times, but apparently share common Tsync driver structure (assuming) both being accessed asynchronously in the cores that cause data corruption.

   Assigning CPU Affinity of each thread to one core results in perfect operation with only Windows scheduler / dispatch jitter of 20 to 230usec observed.

   Please have your software guys verify. I suggest a documentation update "if more than one interrupt source is desired, **each event hander must be in a separate process / thread and all tsync interrupt processes/threads must execute on the same CPU core."** A better approach would be to redesign it..one day…

2. **Interrupts weren't being generated on one platform but interrupts did occur when the board was moved to a different platform:**

   Email from David Higgins (4/21/11) After a bit of Google research and a bit of poking around with the Linux lspci command, I found the underlying reason for why one particular machine here – one using the Intel S5520HC motherboard, and the one I was using for development work, naturally – wasn't generating interrupts when it should. What I found was the PCI 2.3 specs (see: http://tinyurl.com/442xxfh) added an "Interrupt Disable" bit to the PCI configuration space Command register. lspci showed that this bit was set for the TSync-PCIe card when installed in the Intel server motherboard system, and cleared on two other systems.

   Since the tsyncpci driver is GPL'ed, I can use a one-line addition calling pci_intx() to clear this bit. The pci_intx() call has been around since at least 2.6.15, but I can't say exactly when it came into being. Here's the code change I made to tsync_drv_2_6_13_1.c, starting around line 620 or so:

```
/*
**  set PCI configuration register to enable PCI I/O and Memory access
*/
rv = pci_enable_device(pdev);
if (rv) {
    if (tsyncpci_devp[idx]) {
        kfree(tsyncpci_devp[idx]);
        tsyncpci_devp[idx] = NULL;
    }
    TPRO_LOG(TPRO_LOG_ERROR,
             (KERN_ALERT "[J%lu]%s: pci_enable_device()\n",
             jiffies, __FUNCTION__));
    return -EIO;
}
/*
**  CCUR: make sure traditional PCI interrupts are enabled
*/
pci_intx(pdev,1);
```

The last few lines are my addition. Perhaps I should have added “non-MSI” after “traditional”. Speaking of which, does the TSync-PCIe hardware support MSI and/or MSI-X interrupts?

Anyway, I can’t answer why the Interrupt Disable bit is set on the Intel based machine and not on the others – perhaps the TSync-PCIe card generated an interrupt when the BIOS touched it and the BIOS decided to shut it up. And I don’t know why it’s being set for the TSync-PCIe card but not for the PCIe IRIG card made by those-other-guys-who’s-name-starts-with-S.

But I know a workaround, and now so do you.

TSync-PCIe Uptime

TSYNC_SS_GetUptime: Get the board's total uptime in minutes.

bSync enum

- bSync is a "Sync" status (True or False) indicator included in various call responses
- Same state as the response to the **SS_GetSync 0** call

Your question

We recently purchased a TSync PCIe card which we are using to timestamp radar data by triggering the TSync with a TTL pulse. This seems to be working well and I am able to read back the timestamps using the TSYNC_HW_getTsData function included in the driver. My question is, how do I interpret the bSync value that is in the TSYNC_HWTimeObj structure? I imagine this tells me information about the timestamp quality, but looking through the documentation and header files I can't see what enum it corresponds to.

Answer (reply from Keith, 15 May 2020)

The **bSync** enum is strictly a **True** or **False** indication of whether the TSync board is currently in "Sync" status (same as the **SS_GetSync 0** call) with an external input reference, or synced to itself. The TSync board is also in Sync (Sync state True) if the board is currently in Holdover mode.

Holdover mode is a user-configurable length of allotted time (1 second to 5 years- 2 hours by power-up default) that the TSync board can remain in Synx state, after it no longer has any valid input references available for lock. Holdover mode starts the second that all valid references are lost. While in Holdover mode, the 10 MHZ oscillator is in free-run (no oscillator disciplining to account for inherent drift). The System Time and System 1PPS are currently being maintained by the free-running oscillator.

Holdover mode ends when at least one valid input reference has been restored, or the allotted Holdover period expires before a valid input has been restored (this causes the board's Sync status to go False) or if the board is rebooted (the board cannot boot back-up into Holdover mode. It has to first be able to sync to a valid input reference again, before it can go back into Holdover mode again).

Because Sync/bSync state is true while a valid reference is present, and for a period of time after losing all valid inputs, there is a way to distinguish between the two "Sync" states. Similar to a truth table:

- if **bSync** (or the **SS_GetSync 0** call) is **True**, and the SS_GetHoldover 0 call is **False**, the TSync board is in "full Sync" status, locked to an external reference. Its NOT in Holdover mode.

- if **bSync** (or the **SS_GetSync 0** call) is **True**, and the SS_GetHoldover 0 call is also **True**, the TSync board is currently in Sync and in Holdover mode (it was synced to a reference since being booted-up, but currently has no valid input reference to lock to. Its oscillator is in free-run, so the board's Time and 1PPS are slowly drifting, without any compensation).

- if **bSync** (or the **SS_GetSync 0** call) is **False**, and the SS_GetHoldover 0 call is also **False** the TSync board is NOT currently in Sync (either it hasn't yet synced to a valid reference since boot-up OR, it was Synced at one time since boot-up- but the allotted Holdover period expired with no references being restored. The oscillator is remaining in free-run, so the board's Time and 1PPS are continuing to slowly drift, without any compensation).

**Clock System (“CS” calls for time reads, Timescales, TZO/DST offsets , etc)

- CS calls provide an abstract interface to the timing subsystem.

Where <**time scale**>:

UTC=0
TAI= 1
GPS= 2
LOCAL = 3

Related CS calls/functions:
Get/Set local time offsets
Get/Set Time

Get/Set TimeScale Get or Set board's current time scale.
Get/Set Leap second
DST rules/DST state
Get/Set Year value

Get/Set Local time conversion (TZO and DST correction)
The TSync-PCIe’s system time can be converted to local time, as desired. Local time is configured using two separate values, Time Zone Offset and DST.

The example program to configure he DST settings is **CS_SetDstRule.**

Time Zone Offset (TZO)
Time Zone Offset is configured using the **SYNC_CS_SetTimeZoneOff** API call. The offset is entered as the number of total seconds of TZO offset from UTC, for that particular region’s Standard time. Example: Pacific is 8 hour offset during Standard time. 3600 seconds in an hour x 8 = 28,800 seconds for the Time Zone Offset

Desire to change timescales

- The factory default timescale for the TSync-PCIe board is UTC. It can be reconfigured to be GPS, TAI or Local timescale.
- For a visual indications of the time differences between each of the the time scales, go to:http://www.leapsecond.com/java/gpsclock.htm

Important Note: Unlike SecureSyncs (Which persist the GPS and TAI timescale offsets -such as 15 for GPS and 34 for TAI after they are first obtained/unit rebooted), TSync boards do not persist these values through power-ups. So, after each reboot, the TSync-PCIe board needs to either obtain the offset values from an external reference (such as GPS providing the UT1 value), or the offset needs to be entered using the **CS_SetTimeScaleOff 0 X** API call/example program (where x is the timescale being offset)

X values
UTC=0
TAI=1
GPS=2
Local=3

Until the timescale offsets are either automatically obtained from an external time source or have been manually entered, the timescale offsets will be a value of “0”. So when the HW_Gettime or CS_Gettime call is performed, it will continue to report UTC, no matter what the timescale is specified.

Email from Keith 6/8/12 (just before the June leap second)
I just spoke to one of our engineers, who reminded me that the TSync-PCIe boards do not store what the offset values for the other timescales besides UTC (GPS, TAI and local). So, after each power-up, the TSync-PCIe board needs to first be synced to GPS in order to automatically obtain the current UTC to GPS time scale offset value (the UT1 correction factor, which is currently “15”). Or

the current offset value needs to be set after each reboot. Based on this, your TSync-PCIe board likely hasn't synced to GPS since it was last powered-up. If you happen to be using IRIG input, instead of GPS, the correction factor has to always be "manually entered". Until the offset values are obtained either automatically or manually, they are all 0s. So the HW_GetTimes are reporting UTC – "0", which is still UTC. The "0" in this case needs to be changed to a "15", then the Time Stamps will reported in GPS time.

Everything else your doing is correct. No additional steps are required after each boot-up, as far as the time read itself is concerned. You just either need to connect it to GPS and allow it to obtain the offset from GPS automatically after each boot-up (Note this can take up to 12.5 minutes, after the GPS receiver has started tracking GPS satellites for this information to be transmitted in the GPS message). Or, set the UTC timescale offset to the current value of 15 using the **CS_SetTimeScaleOff 0 2 15** call/example program (just remember this value will increment after each leap second, such as the one occurring on June 30th.

D) Desire to change the TSYNC board's timescale from UTC (Default) to GPS

- See "Important Note" above.

Use the **TSync_CS_SetTimeScale** API call/example program

Where:

UTC=0
TAI=1
GPS=2
Local=3

Example: **CS_SetTimeScale 0 2** (where 0 is the first TSync-PCIe board installed and where 2 is the value for GPS timescale).

E) Desire to change the TSYNC board's timescale to TAI instead of UTC

- See "Important Note" above.

Use the **TSync_CS_SetTimeScale** API call/example program

Where

UTC=0
TAI=1
GPS=2
Local=3

Example: **CS_SetTimeScale 0 1** <enter> (where 0 is the first TSync-PCIe board installed and where 1 is the value for TAI timescale).

F) Desire to change the TSYNC board's timescale to local time instead of UTC

Refer to the TSync-PCIe driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

The default timescale for the TSync board is UTC time. The TSync-PCIe board can be set to use a local timescale for the core clock instead of being configured for UTC. The local timescale needs to be setup and then the system needs to be set to utilize that timescale.

First, you need to set the local Time Zone Offset from UTC using the **TSYNC_CS_SetTimeZoneOff** call. The offset should be provided in seconds offset from UTC.

If you want to provide DST information, use the **TSYNC_CS_SetDstRule** library call.

For example, the DST rule for the Eastern Time Zone is:

Reference: 0 (Local time reference)
In: Week: 2
Day: 0 (Sunday)
Month: 3 (March)
Hour: 2
Out: Week: 1
Day: 0 (Sunday)
Month: 11 (November)
Hour: 2
Offset: 3600 (1 hour)

The final step is to tell the clock to utilize the local timescale with the **TSYNC_CS_SetTimescale** library call. Once you change the board's time scale to local, you can enter the match times in local time as well as they will then correlate to the same time base.

**Time/date reads / (HW_GetDate, HW_GetTime and CS_GetTime) / Time epoch

HW_GetDate call

HW_GetDate is one of the legacy TPRO calls in the TSync driver. It is in the TSync-PCIe in the TPRO calls to support the legacy TPRO/TSAT timing boards.

HW_GetTime and CS_GetTime calls

There are two main available calls for time reads: **HW_GetTime** and **CS_GetTime**

- During the HW and CS calls, the current time is latched in the FPGA. There are several 16 bit registers to be read for each time read (Refer to Section 2 of the Application Programmers guide)

  1) Host Super-Second System Time Low (Seconds and minutes data)

     Note: Reading this register latches all of the Host Sub-Second System Time, Host Super-Second System Time, and Host Sub-Second BCD Time registers.

  2) Host Super-Second System Time Mid Low (Hours and some of the day info)
  3) Host Super-Second System Time Mid High (Some of the day info and some of the year value)
  4) Host Super-Second System Time High (Some of the year value).
  5) Host Sub-Second System Time Low
  6) Host Sub-Second System Time High (Syste)
  7) Host Super-Second Binary Time Low (Binary time data)
  8) Host Super-Second Binary Time High (Binary time data)
  9) Host Sub-Second System BCD Time Low (BCD data)
  10) Host Sub-Second System BCD Time High (BCD data)

G) HW_GetTime calls (HW_GetTime, HW_GetSecTime TPRO_GetTime)

- Refer to TSync linux driver for miore info on these calls
- **HW_GetTime (and HW_GetSecTime)** reads time directly from the FPGA with no processing performed on the timing board.
- **HW_GetTime** and **HW_GetSecTime** calls report sync status (from the high order bits register) as well as years, days, hours, minutes seconds, and sub-seconds (all from the low order bits register) with 1 microsecond resolution. The seconds are floating point numeric.
- Typical response times for **HW_GetTime** time reads is about 3 to 4 microseconds.
- The **HW_GetTime** and HW_GetSecTime API/Example programs can read the time of the board with 1 microsecond resolution.

3. TPRO_GetTime

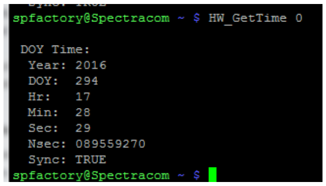
- This is one of the legacy TPRO calls and not the most accurate call to use.
- **HW_GetTime** or **HW_GetTimeSec** is more accurate (see email below):

Q On average, the time reported by the Spectracom is 0.465 second late compared to the time we got from the other GPS base device. Time in the Spectracom is retrieve via API call TPRO_GetTime()

A. (Email from Dave Sohn): The most accurate time reads will be using the HW_GetTime calls. I wouldn't use the legacy TPRO calls.

Note: If the timestamps are being returned in local time instead of UTC (even with the system timescale set to UTC) the board is lilely synced to an IRIG generator providing local time input. The IRIG input needs to be programmed for this. Search this document for “IR_SetLocal” for info on this configuration.

4. **HW_GetTime 0 call**

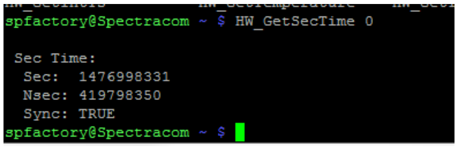


```
spfactory@Spectracom ~ $ HW_GetTime 0

 DOY Time:
  Year: 2016
  DOY:  294
  Hr:   17
  Min:  28
  Sec:  29
  Nsec: 089559270
  Sync: TRUE
spfactory@Spectracom ~ $
```

- **responds with**: Year, DOY, Hr, Min Sec, nanosecond, Sync status (“bSync”)
- Timescale is the same as the System Timescale (UTC by defaut)

5. **Note:** If the timestamps are being returned in local time instead of UTC, (even with the system timescale set to UTC) the board is lilely synced to an IRIG generator providing local time input. The IRIG input needs to be programmed for this. Search this document for “**IR_SetLocal**” for info on this configuration.

6. **HW_GetSecTime 0 call**



```
spfactory@Spectracom ~ $ HW_GetSecTime 0

 Sec Time:
  Sec:  1476998331
  Nsec: 419798350
  Sync: TRUE
spfactory@Spectracom ~ $
```

- **Responds with**: Seconds and nanoseconds as well as Sync status (“bSync”)
- Timescale is the same as the System Timescale (UTC by defaut)
- For an online converter (seconds to date/time) refer to: Unix Epoch time to human readable time converter (in thisdocument)
  - No matter what converter is used, make sure its configured to convert to “UTC” timescale instead of “local” timescale. If **its not set to output “UTC”, it will look like the timestamps are outputting local time.** The section at the link above discusses this, as well

Note: If the timestamps are being returned in local time instead of UTC, (even with the system timescale set to UTC) the board is lilely synced to an IRIG generator providing local time input. The IRIG input needs to be programmed for this. Search this document for “IR_SetLocal” for info on this configuration.

The **TSYNC_HW_GetSecTime()** call is normally the best command for reading the time.

“bSync” field: As it is likely important to know whether or not the board is currently synchronized to an external reference when the time was read, the
TSYNC_HWTimeSecondsObj struct's **bSync** field will indicate:

- “**TRUE**” when synced (the board is either currently synced to its external reference or in holdover mode) OR
- “**FALSE**, when “not in sync” (the board hasn’t synced to its external reference since it was last powered up or it went into holdover mode and the holdover period has since expired).

Time epoch

Q. This is regarding time in sec, ns form. I wanted to know what the starting time, or epoch, is for time in this form. So, I set the time on my board to zero as follows: **CS_SetTime 0 2 0**

Then I read back the time. Turns out the epoch is, which is the same as that used by Unix/Linux.

So, onto my question: This epoch information is not documented anywhere in the Spectracom docs that I can find. Will the epoch always be Jan 1, 1970 for time in this format for Spectracom TSync products?

A. This is the same epoch you referred to in your email for Linux (Jan 1, 1970). The TSync-PCIe-PCIe boards will always use this epoch for its time base.

From Wikipedia (http://en.wikipedia.org/wiki/Unix_time)

Unix time, or POSIX time, is a system for describing instants in time, defined as the number of seconds elapsed since midnight Coordinated Universal Time (UTC) of Thursday, January 1, 1970 (Unix times are defined, but negative, before that date), not counting leap seconds, which are declared by the International Earth Rotation and Reference Systems Service and are not predictable. It is used widely in Unix-like and many other operating systems and file formats. It is neither a linear representation of time nor a true representation of UTC (though it is frequently mistaken for both), as it cannot unambiguously represent UTC leap seconds (e.g. December 31, 1998 23:59:60), although otherwise the times it represents are UTC. Unix time may be checked on some Unix systems by typing date +%s on the command line.

Register Memory Mapping/values/timescales for the HW_GetTime and HW_GetSecTime calls

- If the System timescale is still set to the default value of UTC, HW_GetTime **calls always** return the time elapsed since the UNIX epoch (1 Jan 1970, 00:00:00) – if The System Timescale is changed, the time is adjusted for the system timescale's offset from UTC:

A) If the System timescale is still set to default of UTC, HW_GetTime calls will respond with the time since "1970".

B) If the System timescale is set to either GPS or TAI, HW_GetTime calls will respond with the time since "1970" adjusted by the system timescale's offset from UTC (such as "17" seconds for GPS timescale or "36" seconds for TAI timescale, for instance).

Q I would like to know if the seconds part of the TSYNC_TimeSecondsObj is supposed to be seconds elapsed since the 1970 Epoch or not. I checked the TSync driver guide which shows the structure but does not indicate if the seconds represent seconds since a particular date.

A Response from Keith (19 Oct 16) (since slightly modified) If the System timescale is still set to the power-up default of UTC timescale, the TSync_TimeSecondsObj is the seconds elapsed since the UNIX epoch of 1 Jan, 1970 00:00:00. However, if the System Timescale has been changed from the default timescale of UTC to one of the other available two timescales (TAI or GPS), the number of seconds in the response are adjusted for the offset between UTC and the selected timescale.

For example, there are currently **18** seconds of offset between UTC and GPS time scales. And there are currently **37** seconds of offset between UTC and TAI time scales. The time offset between these two other available timescales are adjusted, after the current time has been read out of the time registers to compensate for the time differences.

"Host Super Seconds System Time" Registers/Memory map

- Refer to "**Host Super-Second System Time**" in the Tsync Application Programmers manual for additional info

Q (From Salesforce case 24795) In the TSync TIME CODE PROCESSOR with OPTIONSL GPS Application Programmers guide there is a memory map. For registers 0xC and 0xE the super second binary time is from what start date?

A response from Keith (24 Mar 17) These registers for the super second binary time are from the start date of 1 January, 1970, 00:00:00 (the UNIX epoch).

If the TSync board's System timescale is still set to the power-up default of UTC timescale (the CS_GetTimeScale 0 API call/example program responding with a "0" for UTC timescale), these time registers are the seconds elapsed since the UNIX epoch of 1 Jan, 1970 00:00:00. However, if the System Timescale has been changed from the default timescale of UTC to one of the other available two timescales (TAI or GPS), (note this is not common) the number of seconds are adjusted for the offset between UTC and the selected timescale.

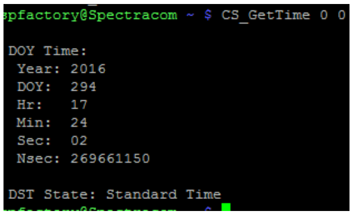
For example, there are currently 18 seconds of offset between UTC and GPS time scales. And there are currently 37 seconds of offset between UTC and TAI time scales. The time offset between these two other available timescales are adjusted, the time registers are adjusted to compensate for the time differences.

A) CS_GetTime calls

- **Sync Status: CS_GetTime** doesn't report sync status (**HW_GetTime** calls do report the sync status)
- With the **CS_Gettime** calls, the FPGA works with the micro to format the time responses. So, **CS_GetTime** is much slower at responding. We recommend using **HW_GetTime** calls for faster responses.

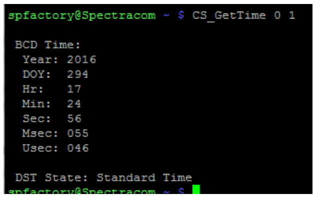
CS_Getime 'instance' values

- **CS_GetTime** Instance value of **0** returns: **Year, DOY, Hr, Min Sec, nanoseconds and DST**



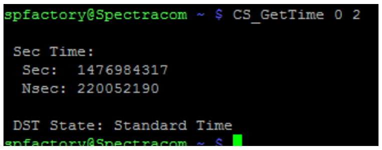
```
spfactory@Spectracom ~ $ CS_GetTime 0 0

 DOY Time:
  Year: 2016
  DOY:  294
  Hr:   17
  Min:  24
  Sec:  02
  Nsec: 269661150

 DST State: Standard Time
spfactory@Spectracom ~ $
```

- **CS_GetTime** Instance value of **1** returns: **Year, DOY, Hr, Min Sec, milliseconds, microseconds and DST.**



```
spfactory@Spectracom ~ $ CS_GetTime 0 1

 BCD Time:
  Year: 2016
  DOY:  294
  Hr:   17
  Min:  24
  Sec:  56
  Msec: 055
  Usec: 046

 DST State: Standard Time
spfactory@Spectracom ~ $
```

- **CS_GetTime** Instance value of **2** returns: **Seconds, nanoseconds and DST**.



```
spfactory@Spectracom ~ $ CS_GetTime 0 2

 Sec Time:
  Sec:  1476984317
  Nsec: 220052190

 DST State: Standard Time
spfactory@Spectracom ~ $
```

Time values/timescales for the CS_GetTime time read calls

- CS_GetTime **calls always** return the time elapsed since the UNIX epoch (1 Jan 1970, 00:00:00). This time is adjusted for the system timescale's offset from UTC, if the system timescale has been changed from the default setting of UTC.
  - **If the System timescale is set to power-up default value of UTC, CS_GetTime calls will respond with the time since "1970".**
  - **If the System timescale is changed to either GPS or TAI (instead of UTC) (the CS_SetTime 0 call will respond with a "0") CS_GetTime calls will respond with the time since "1970", adjusted by selected system timescale's offset from UTC (such as "18" seconds for GPS timescale or "37" seconds for TAI timescale).**

Q I would like to know if the seconds part of the TSYNC_TimeSecondsObj is supposed to be seconds elapsed since the 1970 Epoch or not. I checked the TSync driver guide which shows the structure but does not indicate if the seconds represent seconds since a particular date.

A Response from Keith (19 Oct 16) The TSync_TimeSecondsObj is the seconds elapsed since the UNIX epoch of 1 Jan, 1970.

However, if the System Timescale has been changed from the default timescale of UTC to one of the other available two timescales (TAI or GPS), the number of seconds in the response are adjusted for the offset between UTC and the selected timescale.

For example, there are currently 17 seconds of offset between UTC and TAI. And there are currently 36 seconds of offset between UTC and 36 seconds. The time offset between these two other available timescales are adjusted after the current time has been read out of the time registers to compensate for the time differences.

Reading/setting the current timescale

CS_GetTimeScale 0

CS_SetTimeScale 0 (syntax CS_SetTimeScale <device index> <time scale)

Where:

UTC=0 (power-up default)
TAI=1
GPS=2
Local=3

HW_GetTime versus CS_GetTime

- **HW_GetTime** reports sync status. **CS_GetTime** does not report sync status
- The **CS_GetTime** command comes from the FIFO buffer whereas the **HW_ GetTime** command is directly from a register controlled by the microprocessor. This results in fast response to the **HW_GetTime** command (only when the external reference is connected) but delays in the response from a CS_GetTime command (It was reported that the delays are not present when the card is free-wheeling).

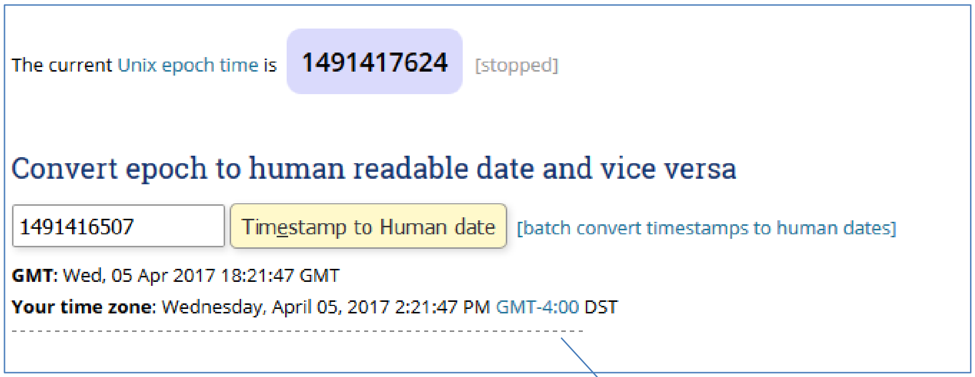
Email from a customer (discussing getDate, but it also applies to CS_GetTime calls)
This delay in getDate (or CS_GetTime) seems to occur regardless of call frequency – I made my original tests at 100Hz (10ms between calls), but the delay is apparent even down to 10Hz (100ms between calls). Once a second, a call to getDate will block for 100ms, but only while the external synchronization signal is connected to the card.

The HW_GetTime command shows no delay, only the getDate (or CS_GetTime) command. It's my thought that synchronization is interfering with FIFO operation in some way.

Normal recovery time for the FIFO buffer operations is about 200 microseconds. GetTimes can read times like every microsecond. This can lead to large time corrections. This issue is even worse when IRIG time code is present. With IRIG signal present, the micro has to process the timecode, slowing down the FIFO buffer operations, resulting in even larger time corrections occurring.

Unix Epoch time (reported in the gettime calls) to human readable time converter

http://www.epochconverter.com/



The current Unix epoch time is 1491417624 [stopped]

Convert epoch to human readable date and vice versa

1491416507

GMT: Wed, 05 Apr 2017 18:21:47 GMT

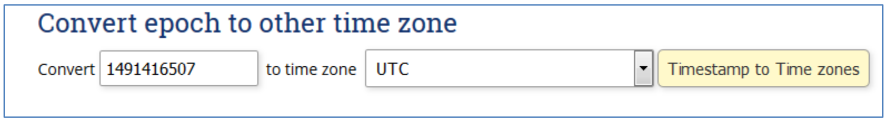
Your time zone: Wednesday, April 05, 2017 2:21:47 PM GMT-4:00 DST

Timescale setting of this onconverter

- ➢ Converter defauts to applying offset to report LOCAL time (not UTC)
- ➢ Can change it to report in UTC epoch

At the end of the "**your time zone"** row, click on the **"GMT…"** to open another browser page.

Then change the **"time zone"** drop-down to "**UTC**" (at the very bottom of the list) as shown below:



Convert epoch to other time zone

Convert 1491416507 to time zone UTC

Latencies of HW_GetTime and CS_GetTime calls:

- ➢ Refer to the next section below for more info (Latentices of API calls)

**Latencies of API calls/Cable delays (including HW_Get Time and CS_GetTime calls)

Latencies in the TSYNC board are inherent with the operation.

Accounting for Cable delays

Latencies in the TSYNC board are inherent with the operation of the TSync board and can also be caused by latency of the external input:

If latency (offset) is coming from the input reference, there are API calls specific to those references to apply an input offset:

- **TSYNC_IR_SetOffset** (IRIG Input) Used to enter an offset for the IRIG input (entered in the number of nanoseconds, as a positive or negative value)
- **TSYNC_GR_SetOffset** (GPS Input) Used to set the GPS antenna cable delay.
- **TSYNC_PR_SetOffset** (EPP 0 input) External PPS Input offset for cable and other delays.

HW_GetTime/ CS_GetTime call latencies

- ➢ Hardware time reads (reading time from the registers) have VERY low latencies (in the low microseconds range), but the process of asking for and receiving these hardware time stamps at the application layer adds quite a bit of latency to the call itself (possibly measured in the milliseconds range).
- ➢ Windows PCs will have even higher latencies than a Linux box will have.
- ➢ There are several "layers" for latency to be added to a hardware time read, as shown below (the timing board itself, PCIe bus, driver, OS kernel and Application software).

The lowest latency is with the reading of the registers on the TSync-PCIe board.
The highest latency is from the OS kernel and Application software layers.

- ➢ Hardware time reads are a "two-way" process (asking for and then receiving the time stamps). So latencies in each layer are added in both directions for each time read.
- ➢ One potential work-aound/suggestion for much lower latentcies is to use the "**1PPS Received**" interrupt to generate an interrupt at the top of each second. Since the interrupt is not generated through the driver and other layers that case latentcies, it's a "one-way direct trigger" that happens at a specific moment each second without a need to request it with an API call. Refer to the email below in red for more on this suggestion.

Email from Denis Reilly (19 Mar 2013) We experimented a bit when we first released the TSync. I don't remember the exact data, but I do remember some things:

round-trip HW_GetTime call latency was on the order of several microseconds in Linux.

In Windows, it could be twice as long.

Inherent latencies with the board (delays between each time read) versus timing accuracies

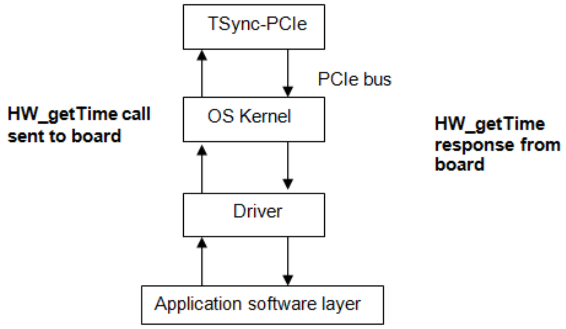
Solaris time read latency from TSYNC_HW_GetTimeSec should be in the range of 8-10 usecs. We don't have any information on latency of the Solaris systems calls gettimeofday() or gethrtime().

Q (from a customer) We are finding that it is taking approximately 3.58 microseconds to read the time of day from the TSync card. We are of the understanding that we should be able to read time from the card with 1 microsecond time of day resolution, which is what we need for our application (all of our computers, software, and data are expecting to be sync'd to 1 usec). Do you have any test programs, code snippets, or anything else we could use to try and replicate a 1 usec time of day read accuracy?

A. There is confusion between latency and accuracy. The latency between time reads can easily be the 3.58 usecs they are seeing. However, that doesn't mean the accuracy of the read is 3.58 usecs. The accuracy will depend on the accuracy of their input and the variation of the latency of their time reads. This is pretty system dependent. They could model the variation in the latency of their time reads, and that should be representative of the accuracy they are getting from the card.

There are a couple of factors to keep in mind when performing time reads of the TSync-PCIe timing board. One of these is inherent latencies between time reads that occur and another is the accuracy of each time read (as compared to the input reference for the board). Latency is the amount of time required between each time read occurring (as determined by the board, the computer, software, etc). The amount of elapsed time between each read explains why you are seeing 3.58 microseconds to read the time from the Tsync board. If you try to read the time twice in the same microsecond for example, the second read will lag the first read by the amount of time required betweens reads to occur (the latency). Responses to time reads cannot occur faster than the latency time permits. To prevent seeing delays in the time reads, allow a delay of about 4 microseconds between each time read. This will negate the ~4microsecond delays you are seeing.

The accuracy of each individual time read will be dependent upon the accuracy of the input reference to the TSync-PCIe board combined with the latencies you are measuring on each time read. The latencies may always be a set length of time, or they may vary from one read to the next. The more consistent the latency measurements are, the better the representation of the accuracy of the time reads (consistent latencies indicates more accurate time reads).


TSync-PCIe
PCIe bus
HW_getTime call sent to board
OS Kernel
HW_getTime response from board
Driver
Application software layer

Email KW sent to customer, with input from Denis Reilly (11/27/12)

To begin, reading the hardware time of the TSync-PCIe board (using the HW_GetTime call) provides the least latency for accurate time reads of the timing board. However, there are several areas where latencies/variable latencies can be introduced into the actual time read.

HW_GetTime calls are "two-way" calls (from the request, to the reply). In between, the timing board hardware, its associated driver, the PCIe bus, the OS kernel, and the application software all come into play. Each one of these areas can introduce latencies, especially at the OS kernel and Application software levels (Windows is even worse than Linux, at these two areas) and with the TSync-PCIe hardware having the least latency.

As Windows is not a real time system, its "priorities" can play a big part in latencies affecting the time reads. If a time read is sent, but the OS is busy when the time stamp is returned, the time will have induced latencies, with this being one of the biggest factors. Latencies for reading hardware registers are typically measured in the low microseconds. But when brought to the application layer, the latencies can then be measured in milliseconds (especially in a Windows environment).

Specific to the "10ms" clock, the problem with this testing is the fact that this clock is not extremely accurate, so the time read commands aren't necessarily going to occur at the same time, each time. The time reads at the time of the read itself is typically accurate to 1 microsecond, but the processing delays/latencies for when the call is sent, to when the time is returned, is going to appear as jitter/offset to the 10ms clock.

One recommendation we have for you, that may help with both testing and with your application software, is to utilize the 1PPS interrupt capabilities of the timing board. The TSync-PCIe boards can generate an interrupt at the top of each second. As these PPS interrupts don't rely on all of the API interfaces mentioned above (Application software, OS, driver, etc), a low latency "marker" can then be used as a "hardware time stamp". Generated interrupts, unlike time-reads, are a "one-way direct communication" between the timing board and the application software, thereby reducing all of the two-way latencies inherent with API calls and hardware time reads. And unlike "software only" approaches to time synchronization, these very low latency hardware interrupts can be generated by the TSync-PCIe timing boards, when they are installed.

An interrupt can be generated at the top of each second. This particular "interrupt generator" is known as the "**1PPS Received**" interrupt (designated by "intType value "0"). Attached for your reference is a copy of the TSync-PCIe driver guide. Section 5.1 (starting on page 5-238) discusses how to enable interrupts to be generated for different situations, such as each time the system PPS occurs.

In summary of the Driver guide, generating this interrupt consists of **disabling** the interrupt mask for "1PPS received" (all interrupt masks are **enabled** by default). To disable the 1PPS Received mask (thereby enabling the "1PPS Received" interrupt), use the following call: **HW_SetIntMask 0 0 0 0** (where the first 0 is for the first/only TSync board installed in the system, the second 0 is for the "1PPS Received" interrupt, the third 0 is the index value and the last 0 disables the interrupt mask).

With the 1PPS interrupt now being generated at the very top of each second, one available option is to use the available "**waitFor**" API blocking call, to halt the application software until the "1PPS Received" interrupt has occurred. The API call to wait for this interrupt before the application software proceeds in **./HW_WaitforInt 0 0 0** (where the first 0 is for the first/only TSync board installed in the system, the second 0 is to wait for the "1PPS Received" interrupt and the third 0 is the index value).

You may also be able to use the generated 1PPS interrupt in other ways, as a VERY low latency marker for your testing (unlike using the very high latency 10ms clock at the application software layer to send and receive the hardware time reads – which are affected by high latency factors in between).

Desire for faster time reads (HW_GetTime takes too long)

Instead of delays to read all of the TSync-PCIe's "Host" registers, the machine can have NTP sync to the TSync-PCIe board- this will then sync the Linux kernel. Then, the kernel time can be read directly (this is much faster than reading the TSync board, with time reads taking only about 1 microsecond, instead of about 4 microseconds)

The results you are observing are not unexpected. For low latency time reads, we recommend using the HW_GetTime command, instead of using the CS_GetTime command. Here's why:

The CS_GetTime command comes from the FIFO buffer (as the time/date data is formatted before being reported) whereas the HW_ GetTime command is directly from a register controlled by the microprocessor. This results in VERY fast response to the HW_GetTime command (only when the external reference is connected) but delays in the response from a CS_GetTime command. HW_GetTimes have about a 4 microsecond delay.

Normal recovery time for the FIFO buffer operations is about 200 microseconds. HW_GetTime can read times every microsecond. This can lead to large time corrections. This issue is even worse when IRIG time code is present. With IRIG signal present, the micro has to process the timecode, slowing down the FIFO buffer operations, resulting in even larger time corrections occurring.

If its desired to have even lower latency than the HW_GetTime call (about 1 microsecond versus about 4 microseconds) and the board is installed in a Linux machine, instead of reading the time directly from the TSync board, the TSync board can sync the Linux machine. Then, the kernel time can be read directly, instead of reading the time of the TSync boards. Attached you should find a document that discusses how to sync the Linux kernel to the time of the TSync board

Syncing a linux system/kernel

- Sync the Linux kernel time to the TSync-PCIe board using TimeKeeper software

  Note: Not applicable for Windows or Solaris

- For more info on Timekeeper software, refer to the Timekeeper software section towards the beginning of the

the VelaSync/Geo Tech note I:\Customer Service\1- Cust Assist documents\VelaSyncAndGeoCustAssist.pdf

Syncing a Linux system with a TSync-PCIe board and TimeKeeper software is the most optimum and accurate method to sync a Linux system. According to Denis, this method will provide sub 1 microsecond synchronization of the linux kernel

- Sync the Linux kernel time to the TSync-PCIe board (when not using TimeKeeper software)
- NTP in the system needs to be patched to install a reference clock driver that allows NTP to sync to the installed TSync-PCIe-PCIe board(s).
- Refer to the UNIX Application Note for general information on syncing Linux: Software\UNIX\UNIX Application Note.pdf
- Refer to the README file in the TSync Linux driver for specific information (tsync\linux\ntp folder) on patching NTP to use the TSync-PCIe board as a reference clock driver.

How to read the Linux kernel time, once it's being synced to the TSync board via the Reference Clock driver

(Email KW sent to Jeffries on 3/28/12)
As for reading the system time, we use the **gettimeofday** call in our software. We only need the precision to 1 microsecond that this call provides, so this call works well for us. There is also an available **clockgettime** call and **clockgetres** (reports the resolution of clockgettime) call that may be able to provide you with even greater precision than 1 microsecond.

We don't directly support these calls, but with a Google search, you should be able to readily find additional information and c code on these calls (as well as any others that may also exist). For examples, refer to sites such as:
http://souptonuts.sourceforge.net/code/gettimeofday.c.html

http://stackoverflow.com/questions/5362577/c-gettimeofday-for-computing-time

Email KW sent to Jeffries about this (3/26/12) about this concept
To begin, the time, date and sync status are maintained in more than one "Host" register on the TSync-PCIe board. These time registers in the TSync-PCIe board need to be latched each time a time read needs to be performed. The first time read latches the registers (to stop the other registers from continuing to increment). Then additional, consecutive register reads are performed to obtain the rest of the time/date/sync status. The HW_GetTime call allows the time registers to be latched and all of the necessary registers read to obtain this information from the timing board. Because the time and sync status are contained in more than one register, it's not the same as just reading the CPU time. The time it takes to latch and read all of the registers is about 4 usecs, on average.

If faster time responses is more vital to your application than the accuracy of the time stamps, the only way to speed up the response time would be to use NTP software to sync the Linux kernel. Then, you could read the kernel time directly. This would alleviate the delays of the TSync-PCIe's time registers from needing to be read during each gettime request. You would then be reading "CPU" time, instead. Typically, NTP can be synchronized directly to the TSync-PCIe board installed in that machine, to within just a few microseconds of the TSync-PCIe's time. The accuracy of the time stamps will be slightly degraded but the elapsed time to perform them will be less.

If you aren't familiar with NTP software syncing to the installed TSync-PCIe board, the Linux driver for the TSync-PCIe board contains a README file that discusses how to patch the NTP software on the machine to have it sync to the TSync-PCIe timing board. This reference clock driver then allows NTP to sync the kernel, which would then allow you to read the kernel time, much faster than performing time reads directly from the TSync-PCIe board.

Please let me know if you need any additional information on using the TSync-PCIe board to sync the linux kernel, as an alternate method to obtain time reads. I understand from Engineering that this method should allow time reads to be performed in about 1 microsecond or so.

Error: opt err (RC_CHG_PENDING)

- Error message potentially asserted when performing Clock commands

- See info from Dave Sohn below example log entries

```
Time Scale: LOCAL
C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>CS_SetTimeZoneOff 0 3600
Time Zone Offset: +3600
  Error: opt err (RC_CHG_PENDING).
C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>CS_SetDstRule 0 0 5 0 3 2 4 0 10 3 3600
DST Rule:
   Ref: Local
   IN:  w5 d0 m3 h2
   OUT: w4 d0 m10 h3
   Off: 3600
  Error: opt err (RC_CHG_PENDING).
```

Per Dave Sohn (10 Oct 2014) There are several calls that affect the system time on the TSync cards. In order to coordinate the correct application of these settings, we don't allow new settings to occur before the old ones are processed. That is the meaning of this message. It indicates that a time setting change is already pending, and the current setting will have to be attempted after it is completed.

Known TSync-PCIe issues

H) Time backsteps (as reported by Google and as seen with HW/CS Gettime reads)

7. Linux driver version 3.3.1 fixed

From the Linux driver v3.3.1 release notes

□ Fixed rollover/backstep bug. This would occur during the rollover to the next minute.

- 47.999992
- 47.999995
- 48.999999
- 48.000003 (Rollback)
- 48.000006

I) There are currently two known issues with the TSync firmware V2.0.0 release (limited release for Mobile Mode support):

- The 10MHz output was inadvertently disabled.
- The default reference table incorrectly listed IRIG AM as a higher priority reference than IRIG DCLS.

These issues do not affect anything else in TSync, and if a customer with 2.0.0 is not using these features they don't have to upgrade if they don't want to.

Both of these defects will be corrected with the TSync 2.1.0 update. However, these defects are in the EEPROM portion of the upgrade bundle. Therefore, ALL customers are recommended to apply the latest EEPROM patch files when upgrading to TSync 2.1.0, INCLUDING customers who are already at TSync 2.0.0.

J) Issue with the get of GPO square wave's pulse width configuration in firmware versions of at least 2.1.0 and prior.

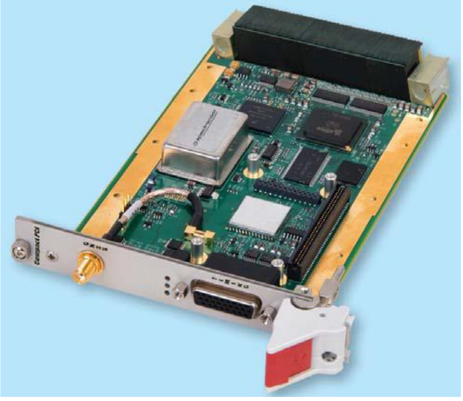
From Erik Johansson of NOAO: I noticed the following interesting output from the getSqWave example code shipped with the driver. The input values are just fake ones that I made up: offset 100ns, period 400 ns, 0 width 100 ns, active edge 0.
All I want to do is set the SqWave config for a pin and then read it back. Turns out the pulse width comes back from the TSync board with the upper bit set. If you mask off the upper bit, the returned value is correct. Is this the proper behavior?

Dave Sohn responded with: There is a bug right now in the released TSync firmware that the high bit of the pulse width may be set. The next TSync firmware release will fix that. For now, the highest order bit of the returned pulse width should be ignored. (Note: as far as I am aware, this issue was resolved in the version 2.3.0 driver update release, Dec 2010).

K) Ping not responding in Solaris (reported 14 March 2013)

Refer to Mantis case 1967. Customer verified ping from Windows but not from Solaris.

TSync-VPX timing boards (1226-xxxx-xxxx)



- Links to **combined** TSync series data sheet (which also includes TSync-VPX boards)
  - On our website: https://www.orolia.com/products-services/precision-timing/tsync-timecode-processors
  - in Sharepoint (in "**Bus-level Timing**" folder): https://oroliagroup.sharepoint.com/sites/oroliasalesmarketing/Shared%20Documents/Forms/AllItems.aspx?RootFolder=%2Fsites%2Foroliasalesmarketing%2FShared%20Documents%2FProducts%2FLiterature%2FDatasheets&FolderCTID=0x01200059B5B4DD4E9CD947A0FA76DD4495D6C0

- Link to TSync family user guide (1191-5000-0050), which also includes TSync-VPX boards (in Arena): https://app.bom.com/items/detail-spec?item_id=1202833276&version_id=10221277928

TSync-VPX boards

- 3U rack
- Very similar to Tsync-cPCI
- Uses CPCI drivers
- SAASM receiver capable.intado672C

First shipment to a customer (treated as a Special)

- **Customer**: Telephonics
- Released on ECO-00285 (~First week March, 2015)

Part Numbers

- **Tsync-VPX shipped to Telephonics**:
  - 1226-9002-0600 (TSync-VPX-423) SAASM receiver
  - 1226-T121-0600 (TSync-VPX-121) commercial GNSS receiver
- **PCI Carrier for VPX (VPX to PCIe adapter):** refer to additional info directly below

PCIe Carrier board for VPX (VPX to PCIe adapter board)

- **Our P/N:** MP41R-0001-0001
- **In Arena:** https://app.bom.com/items/detail-spec?item_id=1208880424&version_id=10315943908&orb_msg_single_search_p=1
- **Can be used to test TSync-**VPX boards on a standard system not having VPX bus connectors

Mechanical/environmental info for TSync-VPX

VPX Specifications

• 3U VPX form-factor Compliant to VITA-46
• 3.9” x 6.3” (100 mm x 160 mm)
• Connectors to VITA 46.0 for P0, P1, and P2
• Bus Interface: PCIe x1, Rev 1.1

Conduction Cooling (cPCI and VPX only)

• Per ANSI/VITA 30.1-2002 (cPCI)
• Per VITA 46/IEEE 1101.2 (VPX)
• Thermal frame available by request
• Component elevations available for custom thermal frame design

Weight

- 6.3 oz/179 g (without thermo frame)
- 1.6 oz/329 g (with thermo frame)

FAQ’s

Q What is the height of the Crystal (Rugged OCX0) and the height of the board + crystal?

Email from Tim Tetreault (7 Oct 16) The height of the Rugged oscillator is 12.7mm.

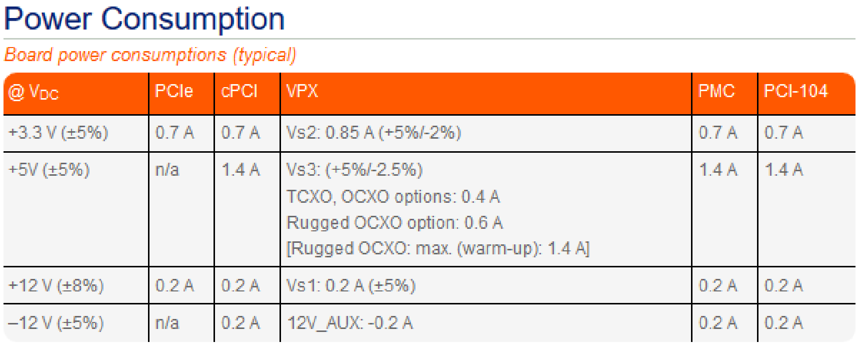
Q Any shock/vibration Data on this board (with thermal frame cooling)?
Email from Tim Tetreault (7 Oct 16) We haven't tested any of our TSync boards under shock/vib since the results would vary and would be dependent on what the customer would install the card into. The only part on our board that would be sensitive to typical shock/vib would be the oscillator. We do have spec's for our rugged OCXO.

Maximum Vibration Sensitivity (Δf/fo/g)__ 1ppb/g
Shock, Operating_____________________ 20g, 11ms, ½ sine, 3 hits in each direction in each axis. The drift error during the shock event shall not exceed 20ppb for the duration of each single shock event.
Vibration, Operating__________________ 2g RMS ; 5 – 500 Hz with maximum peak displacement of 0.75 inches.
Shock, Non-Operating_________________ 30g, 11ms, ½ sine, 3 hits in each direction in each axis.
Vibration, Non-Operating______________ 5g ; 5 – 500 Hz / IEC 68-2-06

Q. Your Environmental Spec lists Altitude: 10,000 feet. Any 40,000 ft?
Email from Tim Tetreault (7 Oct 16) Altitude, Operating____________________ 0 to 40,000 feet

Input Power/Power consumption for TSync-VPX boards

- Refer to **online TSync user guide** at: http://manuals.spectracom.com/TS/Content/TS/Topics/GenSpecs.htm?Highlight=p2

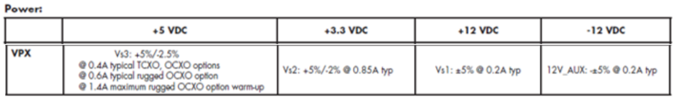


Power Consumption

Board power consumptions (typical)

@ V_{DC}	PCIe	cPCI	VPX	PMC	PCI-104
+3.3 V (±5%)	0.7 A	0.7 A	Vs2: 0.85 A (+5%/-2%)	0.7 A	0.7 A
+5V (±5%)	n/a	1.4 A	Vs3: (+5%/-2.5%) TCXO, OCXO options: 0.4 A Rugged OCXO option: 0.6 A [Rugged OCXO: max. (warm-up): 1.4 A]	1.4 A	1.4 A
+12 V (±8%)	0.2 A	0.2 A	Vs1: 0.2 A (±5%)	0.2 A	0.2 A
–12 V (±5%)	n/a	0.2 A	12V_AUX: -0.2 A	0.2 A	0.2 A

***From combined TSync data sheet**:*



Power:

	+5 VDC	+3.3 VDC	+12 VDC	-12 VDC
VPX	Vs3: +5%/-2.5% @ 0.4A typical TCXO, OCXO options @ 0.6A typical rugged OCXO option @ 1.4A maximum rugged OCXO option warm-up	Vs2: +5%/-2% @ 0.85A typ	Vs1: ±5% @ 0.2A typ	12V_AUX: -±5% @ 0.2A typ

Damage to TSync-VPX-423 by under-voltage??

Q Can the 1226-9002-0600 be damaged by under-voltage?
A Reply from Alex Payne (OGSI) 31 July 2020 I am verifying the impact of supply power out of range and will advise as soon as I have an answer – likely, early next week.

Battery/Batteries on TSync-VPX-423 boards

Q Does the 1226-9002-0600 contain batteries?
A Reply from Alex Payne (OGSI) 31 July 2020 I received your inquiry about the 1226-9002-0600 (TSync-VPX-423) specials product. I have confirmed the assembly does not contain a battery.

TSync-VPX EEPROM/FPGA/firmware updates

1. **ECO-2206: "Fix FPGA and EEPROM issue with TSync-VPX boards (~July 2019)"**
   - Refer to **ECO-2206** (in Arena): https://app.bom.com/changes/detail-summary?change_id=2397421300&
   - A new EEPROM image for the Telephonic's products will need to be created to **change the default GPS/GNSS Mode value to "Mobile AIR"**.

Summary/Description (excerpted from ECO-2206):

This ECO will address Telephonics Ticket BPS-32 **requesting VPX board GPS/GNSS UART requires 1024 byte Receive and Transmit FIFOs.**

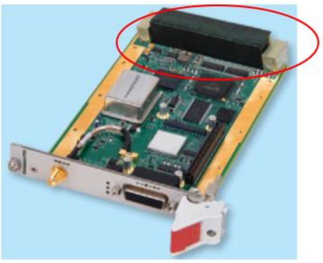
To make this change, the FPGA will need to be updated.

A new EEPROM image for the Telephonic's products will need to be created to change the default GPS/GNSS Mode value to "Mobile AIR".

See Implementation notes for more comments.

TSync pinout info for inputs/outputs

TSync-VPX Connector "P2" backplane Signal Definitions (pinouts)

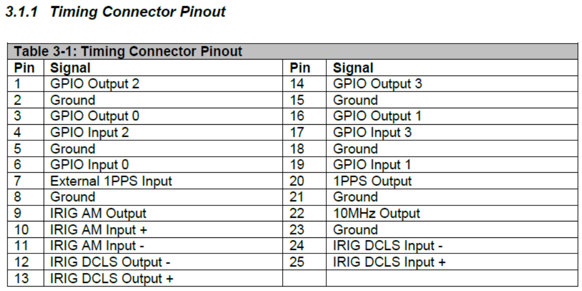


- Refer to the complete pinout document from Tim Tetrault (24 June 2019): ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\TSync-VPX\TSync-VPX P2 backplane

All available TSync-VPX Inputs

- Refer to online TSync user guide at: http://manuals.spectracom.com/TS/Content/TS/Topics/IOSpecs.htm

*Timing connector pinout



3.1.1 Timing Connector Pinout

Table 3-1: Timing Connector Pinout

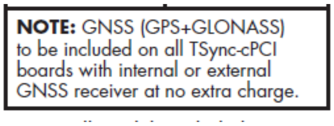
Pin	Signal	Pin	Signal
1	GPIO Output 2	14	GPIO Output 3
2	Ground	15	Ground
3	GPIO Output 0	16	GPIO Output 1
4	GPIO Input 2	17	GPIO Input 3
5	Ground	18	Ground
6	GPIO Input 0	19	GPIO Input 1
7	External 1PPS Input	20	1PPS Output
8	Ground	21	Ground
9	IRIG AM Output	22	10MHz Output
10	IRIG AM Input +	23	Ground
11	IRIG AM Input -	24	IRIG DCLS Input -
12	IRIG DCLS Output -	25	IRIG DCLS Input +
13	IRIG DCLS Output +		

GPS/GNSS input /SAASM receiver related for TSync-VPX boards

TSync-VPX boads are SAASM receiver-capable

- Can be shipped with a Rockwell Collins GB-GRAM receiver
  - For SAASM-related info, refer to (“Model 1204-1A”): U:\Engineering\SAASM-FOUO\CustomerService\SecureSync\1204-1A (GB-GRAM)
- SMA to Type N adapter cable included with convection-cooled models

GNSS + Glonass (for commercial GPS receivers)



NOTE: GNSS (GPS+GLONASS) to be included on all TSync-cPCI boards with internal or external GNSS receiver at no extra charge.

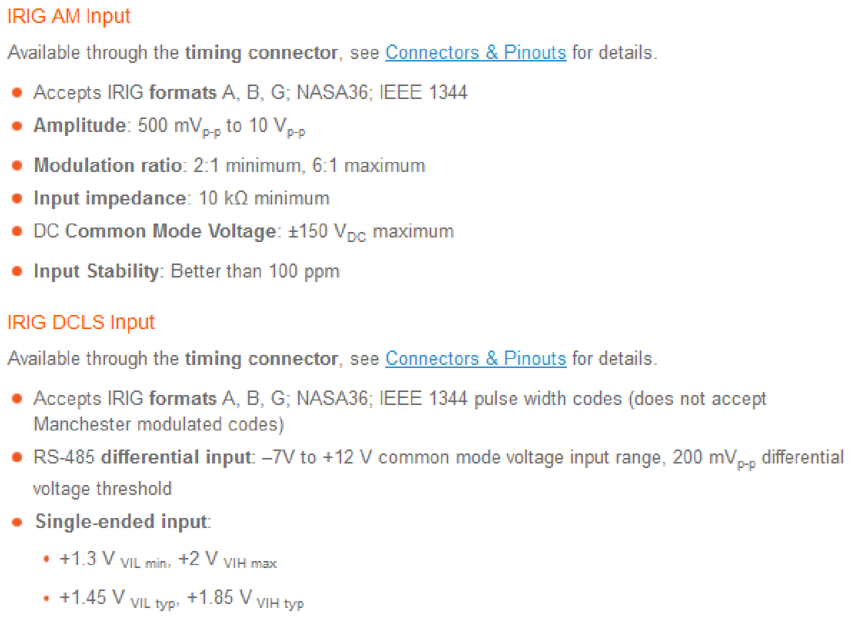
****SS-OPT-GNS:** Adds Glonass L1 1602MHz (now a standard feature)
Per Tim Tetreault (13 Nov 2013) It’s been decided that Glonass is now going to be a standard feature for all TSync-cPCI boards (does not need to be purchased). As of now, there are no available API calls dedicated to Glonass (such as enabling or disabling this function). Instead of the GPS API calls reporting up to 12 channels, they add the Glonass satellites to the count of available total satellites being tracked.

Available LVDS (low voltage differential signaling) signals

- Apparently, no inputs are compatible with LVDS (low voltage differential signaling)
- **10 MHz outputs** are the only signals compatible with LVDS (low voltage differential signaling)

**IRIG AM and IRIG DCLS inputs

- Via Timing Connector
- Dedicated IRIG AM and IRIG DCLS outputs are simulaniously available
- Refer to online TSync user guide at (excerpt below): http://manuals.spectracom.com/TS/Content/TS/Topics/IOSpecs.htm



IRIG AM Input

Available through the **timing connector**, see Connectors & Pinouts for details.

- Accepts IRIG **formats** A, B, G; NASA36; IEEE 1344
- **Amplitude**: 500 mV$_{p\text{-}p}$ to 10 V$_{p\text{-}p}$
- **Modulation ratio**: 2:1 minimum, 6:1 maximum
- **Input impedance**: 10 kΩ minimum
- DC **Common Mode Voltage**: ±150 V$_{DC}$ maximum
- **Input Stability**: Better than 100 ppm

IRIG DCLS Input

Available through the **timing connector**, see Connectors & Pinouts for details.

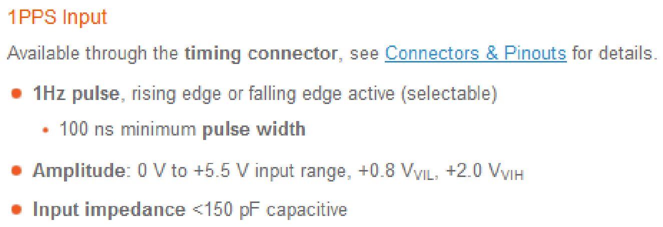
- Accepts IRIG **formats** A, B, G; NASA36; IEEE 1344 pulse width codes (does not accept Manchester modulated codes)
- RS-485 **differential input**: –7V to +12 V common mode voltage input range, 200 mV$_{p\text{-}p}$ differential voltage threshold
- **Single-ended input**:
  - +1.3 V $_{VIL\ min}$, +2 V $_{VIH\ max}$
  - +1.45 V $_{VIL\ typ}$, +1.85 V $_{VIH\ typ}$

**1PPS inputs

- Refer to online TSync user guide at (excerpt below): http://manuals.spectracom.com/TS/Content/TS/Topics/IOSpecs.htm

1PPS input levels

- TTL input only



1PPS Input

Available through the **timing connector**, see Connectors & Pinouts for details.

- **1Hz pulse**, rising edge or falling edge active (selectable)
  - 100 ns minimum **pulse width**
- **Amplitude**: 0 V to +5.5 V input range, +0.8 V_{VIL}, +2.0 V_{VIH}
- **Input impedance** <150 pF capacitive

10 Vpeak input 1PPS signal?

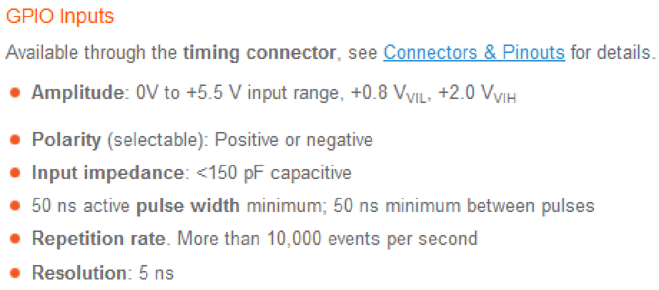
Q Can the TSync-VPX timing boards can accept a **10 Vpeak input 1PPS signal**

A To answer your question (as indicated in the attached Data Sheet and excerpted below for your convenience) of whether the TSync-VPX timing boards can accept a **10 Vpeak input 1PPS signal**, the answer is they cannot accept this level.

As you eluded to in your message, this timing board's input is TTL compatible, so it can't operate with signal levels above 5.5Vpeak.

**GPIO Inputs

- ➢ Refer to online TSync user guide at (excerpt below): http://manuals.spectracom.com/TS/Content/TS/Topics/IOSpecs.htm

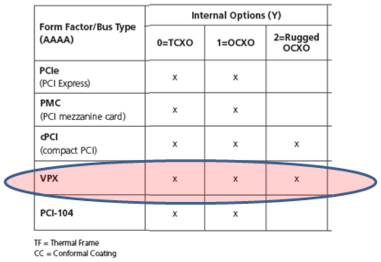


GPIO Inputs

Available through the **timing connector**, see Connectors & Pinouts for details.

- **Amplitude**: 0V to +5.5 V input range, +0.8 V_{VIL}, +2.0 V_{VIH}
- **Polarity** (selectable): Positive or negative
- **Input impedance**: <150 pF capacitive
- 50 ns active **pulse width** minimum; 50 ns minimum between pulses
- **Repetition rate**. More than 10,000 events per second
- **Resolution**: 5 ns

Available 10 MHz Oscillators



Form Factor/Bus Type (AAAA)	Internal Options (Y)		
	0=TCXO	1=OCXO	2=Rugged OCXO
PCIe (PCI Express)	x	x	
PMC (PCI mezzanine card)	x	x	
cPCI (compact PCI)	x	x	x
VPX	x	x	x
PCI-104	x	x	

TF = Thermal Frame
CC = Conformal Coating

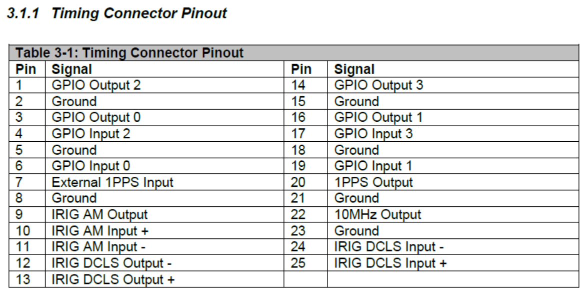
Available oscillators (TSync-cPCI boards)

- **(0) Standard TCXO:** 1ppm
- **(1) Standard OCXO**: 0.5ppm
- **(2) Rugged OCXO**: 0.5ppb., enhanced shock and vibe specs

➢ Refer to the TSync-PCIe section of this document for more info (same as the TSync-PCIe)

All available TSync-VPX Outputs

*Timing connector pinout



3.1.1 Timing Connector Pinout

Table 3-1: Timing Connector Pinout

Pin	Signal	Pin	Signal
1	GPIO Output 2	14	GPIO Output 3
2	Ground	15	Ground
3	GPIO Output 0	16	GPIO Output 1
4	GPIO Input 2	17	GPIO Input 3
5	Ground	18	Ground
6	GPIO Input 0	19	GPIO Input 1
7	External 1PPS Input	20	1PPS Output
8	Ground	21	Ground
9	IRIG AM Output	22	10MHz Output
10	IRIG AM Input +	23	Ground
11	IRIG AM Input -	24	IRIG DCLS Input -
12	IRIG DCLS Output -	25	IRIG DCLS Input +
13	IRIG DCLS Output +		

TSync-VPX P2 backplane signal Definitions

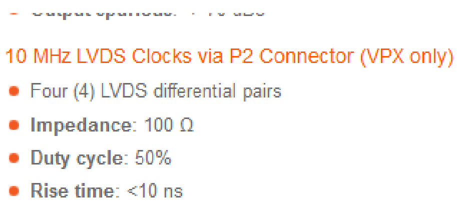
- ➢ Refer to the document from Tim Tetrault (24 June 2019): ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\TSync-VPX\TSync-VPX P2 backplane

Available LVDS (low voltage differential signaling) signals

- Apparently, 10 MHz outputs are the only signals compatible with LVDS level signal (details in 10 MHz output sction directy below)

**10 MHz outputs

- ➢ Via Timing Connector
- ➢ 10 MHz LVDS (low voltage differential signaling) Via Bus Connector (P2)
  - Refer to online TSync user guide (excerpt below): http://manuals.spectracom.com/TS/Content/TS/Topics/IOSpecs.htm



10 MHz LVDS Clocks via P2 Connector (VPX only)

- Four (4) LVDS differential pairs
- **Impedance**: 100 Ω
- **Duty cycle**: 50%
- **Rise time**: <10 ns

TSync-VPX P2 backplane signal Definitions

- Refer to the document from Tim Tetrault (24 June 2019): ..\..\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync family\TSync-VPX\TSync-VPX P2 backplane

Schematic (“1226-1001-0200”) showing the pinouts for the four pairs of outputs (two green circles on right-side of the diagram below)

- Refer to “1226-1001-0200” in the “1226-0000-F001” folder, which is in Arena at: https://app.bom.com/items/detail-spec?item_id=1207453833&version_id=10326353918

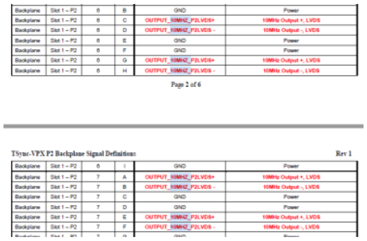
Four sets of 10MHz LVDS outputs

Output 1: P2 pins 6C/6D

Output 2: P2 pins 6G/6H

Output 3: P2 pins 7A/7B

Output 4: P2 pins 7E/7F

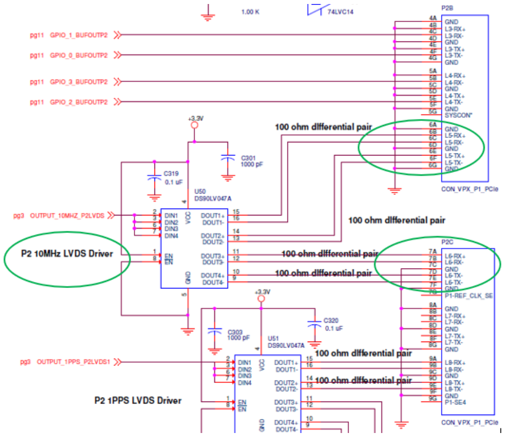


Backplane	Slot 1 – P2	6	B	GND	Power
Backplane	Slot 1 – P2	6	C	OUTPUT_10MHZ_P2LVDS+	10MHz Output +, LVDS
Backplane	Slot 1 – P2	6	D	OUTPUT_10MHZ_P2LVDS -	10MHz Output -, LVDS
Backplane	Slot 1 – P2	6	E	GND	Power
Backplane	Slot 1 – P2	6	F	GND	Power
Backplane	Slot 1 – P2	6	G	OUTPUT_10MHZ_P2LVDS+	10MHz Output +, LVDS
Backplane	Slot 1 – P2	6	H	OUTPUT_10MHZ_P2LVDS -	10MHz Output -, LVDS

Page 2 of 6

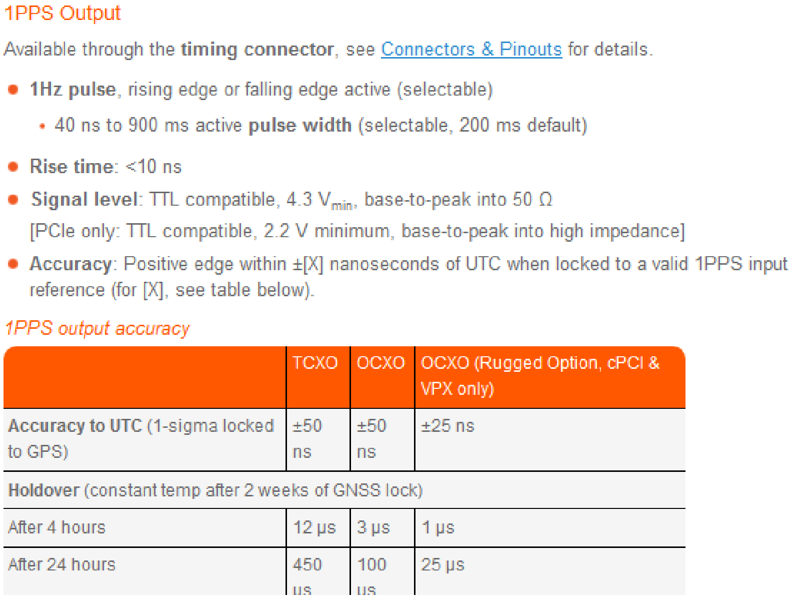
TSync-VPX P2 Backplane Signal Definitions Rev 1

Backplane	Slot 1 – P2	6	I	GND	Power
Backplane	Slot 1 – P2	7	A	OUTPUT_10MHZ_P2LVDS+	10MHz Output +, LVDS
Backplane	Slot 1 – P2	7	B	OUTPUT_10MHZ_P2LVDS -	10MHz Output -, LVDS
Backplane	Slot 1 – P2	7	C	GND	Power
Backplane	Slot 1 – P2	7	D	GND	Power
Backplane	Slot 1 – P2	7	E	OUTPUT_10MHZ_P2LVDS+	10MHz Output +, LVDS
Backplane	Slot 1 – P2	7	F	OUTPUT_10MHZ_P2LVDS -	10MHz Output -, LVDS
Backplane	Slot 1 – P2	7	G	GND	Power



**1PPS outputs/1PPS accuracies

- unlike 10MHz, 1PPS not available as LVDS signal levels
- Via Timing Connector
  - Refer to online TSync user guide at (except below): http://manuals.spectracom.com/TS/Content/TS/Topics/IOSpecs.htm



1PPS Output

Available through the **timing connector**, see Connectors & Pinouts for details.

- **1Hz pulse**, rising edge or falling edge active (selectable)
  - 40 ns to 900 ms active **pulse width** (selectable, 200 ms default)
- **Rise time**: <10 ns
- **Signal level**: TTL compatible, 4.3 V_{min}, base-to-peak into 50 Ω

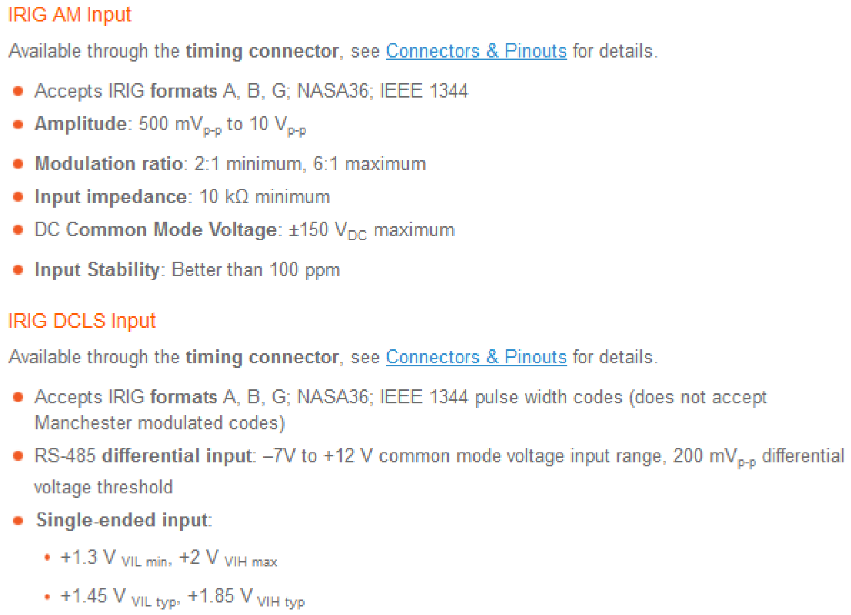
  [PCIe only: TTL compatible, 2.2 V minimum, base-to-peak into high impedance]
- **Accuracy**: Positive edge within ±[X] nanoseconds of UTC when locked to a valid 1PPS input reference (for [X], see table below).

1PPS output accuracy

	TCXO	OCXO	OCXO (Rugged Option, cPCI & VPX only)
Accuracy to UTC (1-sigma locked to GPS)	±50 ns	±50 ns	±25 ns
Holdover (constant temp after 2 weeks of GNSS lock)			
After 4 hours	12 µs	3 µs	1 µs
After 24 hours	450 µs	100 µs	25 µs

**IRIG AM and IRIG DCLS outputs

- Via Timing Connector
- IRIG AM and DCLS outputs available
  - refer to online TSync user guide at (excerpt below): http://manuals.spectracom.com/TS/Content/TS/Topics/IOSpecs.htm



IRIG AM Input

Available through the **timing connector**, see Connectors & Pinouts for details.

- Accepts IRIG **formats** A, B, G; NASA36; IEEE 1344
- **Amplitude**: 500 mV_{p-p} to 10 V_{p-p}
- **Modulation ratio**: 2:1 minimum, 6:1 maximum
- **Input impedance**: 10 kΩ minimum
- DC **Common Mode Voltage**: ±150 V_{DC} maximum
- **Input Stability**: Better than 100 ppm

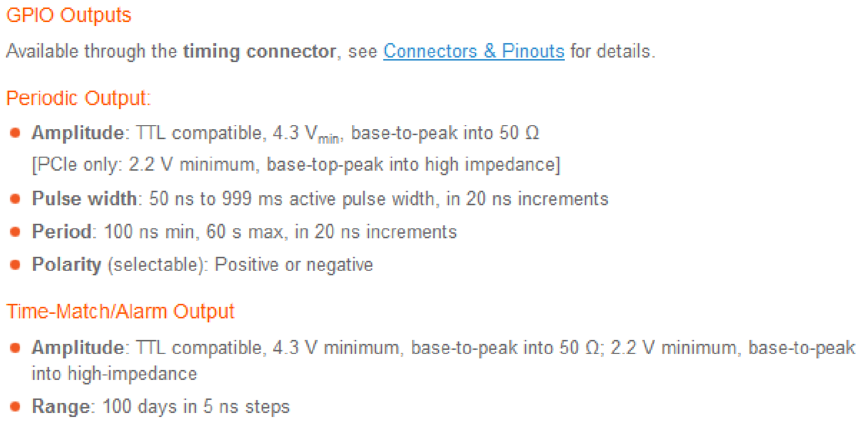
IRIG DCLS Input

Available through the **timing connector**, see Connectors & Pinouts for details.

- Accepts IRIG **formats** A, B, G; NASA36; IEEE 1344 pulse width codes (does not accept Manchester modulated codes)
- RS-485 **differential input**: –7V to +12 V common mode voltage input range, 200 mV_{p-p} differential voltage threshold
- **Single-ended input**:
  - +1.3 V $_{VIL\ min}$, +2 V $_{VIH\ max}$
  - +1.45 V $_{VIL\ typ}$, +1.85 V $_{VIH\ typ}$

****GPIO outputs/Periodic Outputs/Match-time output**

- Refer to online TSync user guide at (excerpt below): http://manuals.spectracom.com/TS/Content/TS/Topics/IOSpecs.htm



GPIO Outputs

Available through the **timing connector**, see Connectors & Pinouts for details.

Periodic Output:

- **Amplitude**: TTL compatible, 4.3 V_{min}, base-to-peak into 50 Ω
  [PCIe only: 2.2 V minimum, base-top-peak into high impedance]
- **Pulse width**: 50 ns to 999 ms active pulse width, in 20 ns increments
- **Period**: 100 ns min, 60 s max, in 20 ns increments
- **Polarity** (selectable): Positive or negative

Time-Match/Alarm Output

- **Amplitude**: TTL compatible, 4.3 V minimum, base-to-peak into 50 Ω; 2.2 V minimum, base-to-peak into high-impedance
- **Range**: 100 days in 5 ns steps

Tsync drivers for TSync-VPX boards

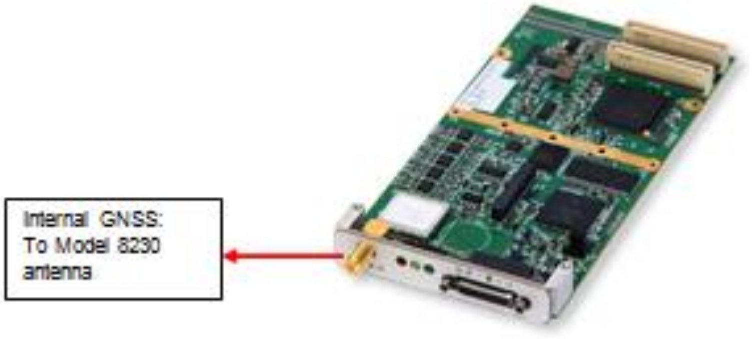
Linux/Solaris

- Added Linux/Solaris support for TSync-VPX boards in ??

**Window driver

- Support in Windows driver for TSync0VPX boards was added in Windows driver version 3.2.1
- From the v3.2.1 Release notes: “Support for the TSync VPX board was added. “

TSync-PMC board (1221-xxxx) (discontinued/no longer available)

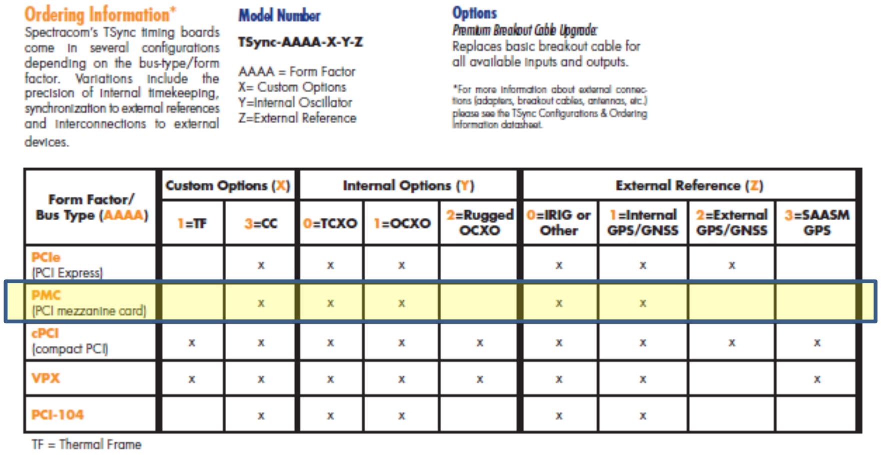


- **Shortcut to data sheet:** I:\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards
- **Shortcut to manual (1221-5000-0050) in Arena:** https://app.bom.com/items/detail-spec?item_id=1203165206&version_id=10221242448&orb_msg_Single_Search_p=1&redirect_Seqno=7664281921
- **Shortcut to drivers on our website:** http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=20
- **Shortcut to TSync-PMC drivers** (same as the TSync-PCIe drivers) in Arena **(1219-SD01-xxxx)**
    - **CD assembly** we ship containing drivers and guides (**1219-5003-6001**) in Arena https://app.bom.com/items/detail-spec?item_id=1203165725&version_id=10222045518
- **Shortcut to driver guide (1219-5001-0050**) in Arena: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002
- **Shortcut to application programmers guide (1219-5002-0050):** in Arena: https://app.bom.com/items/detail-spec?item_id=1203375450&version_id=10222044898&orb_msg_Single_Search_p=1&redirect_Seqno=8205368314
- **Shortcut to firmware and driver version upgrades:** \\Exchange\empshare$\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards
- **Comparison of legacy TPRO/TSAT-cPCI to newer TSync-cPCI:** http://partner.spectracomcorp.com/Portals/7/Other/Compact%20PCI%20Timing%20Board%20Update%20-%20Model%20TSync-cPCI.pdf
- **Shortcut to firmware and driver version upgrades:** \\Exchange\empshare$\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards

Product first released: March 2012 (on ECN 2846)

TPRO/TSAT-PMC boards no longer available/Available TSync-PMC configurations

Slightly modified email from Dave Sohn (25 Sept 2019) Only the TPRO/TSAT-PCI boards have not been discontinued. TPRO/TSAT PMC are unavailable. I suggest looking at the TSync-PMC boards instead.



Ordering Information*
Spectracom's TSync timing boards come in several configurations depending on the bus-type/form factor. Variations include the precision of internal timekeeping, synchronization to external references and interconnections to external devices.

Model Number
TSync-AAAA-X-Y-Z

AAAA = Form Factor
X= Custom Options
Y=Internal Oscillator
Z=External Reference

Options
Premium Breakout Cable Upgrade:
Replaces basic breakout cable for all available inputs and outputs.

*For more information about external connections (adapters, breakout cables, antennas, etc.) please see the TSync Configurations & Ordering Information datasheet.

Form Factor/ Bus Type (AAAA)	Custom Options (X)		Internal Options (Y)			External Reference (Z)			
	1=TF	3=CC	0=TCXO	1=OCXO	2=Rugged OCXO	0=IRIG or Other	1=Internal GPS/GNSS	2=External GPS/GNSS	3=SAASM GPS
PCIe (PCI Express)		x	x	x		x	x	x	
PMC (PCI mezzanine card)		x	x	x		x	x		
cPCI (compact PCI)	x	x	x	x	x	x	x	x	x
VPX	x	x	x	x	x	x	x		x
PCI-104		x	x	x		x	x		

TF = Thermal Frame

Swapping out a legacy TPRO/TSAT-PMC board for a TSync-PMC

Email from Tim Tetreault to Sadie (18 Nov 2013)
The equivalent to the TPRO-cPCI board would be the TSync-cPCI-000.
The equivalent to the TSAT-cPCI board would be the TSync-cPCI-002.

I want to make sure that everyone understands that the new TSync-cPCI board is not a drop-in replacement for the old KSI boards. It will support all of the functions that the older KSI cPCI board had but customers will need to use our new TSync driver and update their software.

The TSync-PMC will work in the same slot as the TPRO-PMC board was installed.

The customer will still need to install the associated TSync driver (available at no cost from our website). The TSync drivers contain all the legacy TPRO/TSAT API calls. The TPRO.lib and TPRO.h files are compatible with the newly installed TSync board, as long as there is no need/desire to take advantage of the newer API calls (such as holdover, for instance), customer application software doesn't need to be changed. But, the application software needs to be re-compiled with the TSync driver before it will work with the TSync-cPCI board.

Note: Customer must install the TSync driver when TSync-PCIe board is installed. The previously installed TPRO/TSAT driver won't be able to communicate with the installed TSync-cPCI board (the device ID is different). So, the TSync driver also needs to be installed (the TPRO/TSAT driver can either be uninstalled or installed simultaneously with the TSync driver, as desired).

Tsync-PMC memory

- Refer to: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync-PMC and cPCI

Q. Email from Brandon Reed
I know this was a while ago, but you sent a document regarding the memory devices on the TSYNC series cards (attached). Our security personal are questioning what can be stored in these memory devices. Specifically, they are asking about the 8MB of Flash and 8KB of EEPROM that is User Modifiable.

Please correct me if I am wrong in the following assumptions:

- The 8MB Flash cannot be directly written to through the API, and can only be accessed by using an upgrade program provided by Spectracom. We (the users) cannot directly write data such as freehand text into this device.
- The 8KB EEPROM contains the input reference table that is basically a register of configuration discrete. Once again, the user cannot write freehand text into this device, but can toggle bits within the register to affect the configuration of which input references the board will synchronize with.

Is my understanding correct?

I also have a few other questions regarding this card. The “persistent memory” document you sent me references two API calls. One is to save data to the input reference table (TSYNC_RS_SaveUserDef), the other is to restore that table to factory defaults (TSYNC_RS_SetFactDef). I was looking through the TSync Application Programmer’s Guide I obtained from the Spectracom website, but I do not find those API calls listed. Has the TSYNC_RS_SaveUserDef call been replaced by RS_CA_SAVE_USER_DEFAULT and TSYNC_RS_SetFactDef replaced by RS_CA_FACT_DEFAULT?

There is another call in the Application Programmer’s guide that is of concern. The PTR_CA_PTP_USER_DESCRIPTION call seems to allow the user to set a 16 character Device Name and a 16 character Device Location. I assume this would allow the user to save custom text. Is this stored in the Flash or EEPROM, or is it stored in volatile memory? If it is persistent, can we use the same command to overwrite the Device Name and Device Location with a known unclassified pattern, then read it back to verify success?

Reply from Dave Lorah (26 Feb, 2015) The TSYnc-PCIe card Flash Memory is used solely for the operating firmware. This memory cannot be written to unless a firmware update is performed. There is no other way to write data to this flash other than load firmware update using.

The TSync-PCIe EEPROM contains data for the card identity such as serial number and model number as well as a User Reference Table that tells the board what input references to use and it what order. Nothing else can be written to the EEPROM. The table data can be cleared if you like using the TSYNC_RS_SetFactDef API call.

You are correct stating no freehand data can be written to these locations.

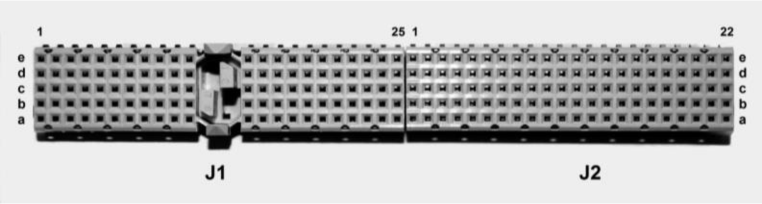
The RS_CA_SAVE_USER_DEFAULT and RS_CA_FACT_DEFAULT commands are equivalent to the TSYNC_RS_SetFactDef API call examples found in the driver tsync/examples folder. The examples are setup to run those commands.

The PTR_ commands apply only to the PTP equipped TSYNC-PTP cards. This command PTR_CA_PTP_USER_DESCRIPTION will write information to the PTP daughter board on these PTP equipped cards. It is true any text can be written to this memory. The way to clear the data would be to overwrite unclassified data to this location.
Again this only applies to TSYNC-PTP models.
I do not believe you have the PTP equipped boards.

Timing Connector/ Breakout cables

- Refer to the TSync-PCIe section in this document (same as the TSync-PCIe):

PMC Bus connectors (J1 and J2)



Note: Refer to the TSync-PMC manual for the pin-outs (likely on pages 3-1 through 3-3)

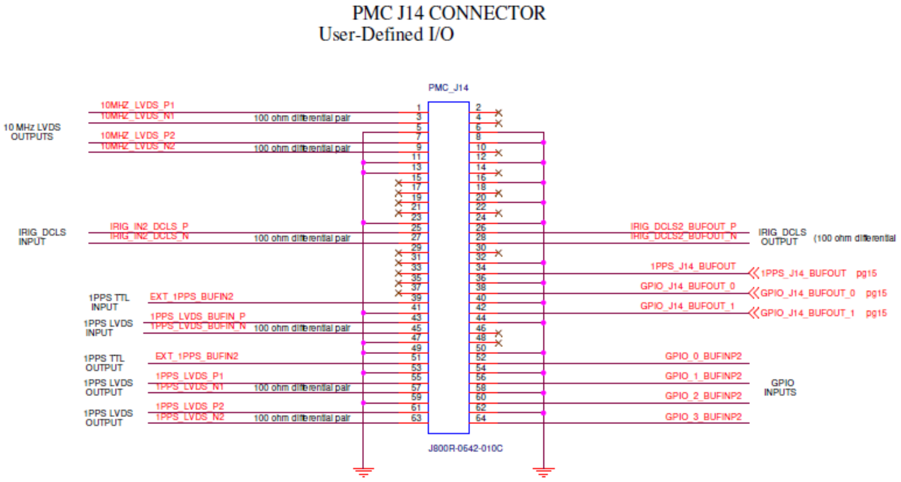
- **Link to manual**: **(1221-5000-0050) in Arena:** https://app.bom.com/items/detail-spec?item_id=1203165206&version_id=10221242448&orb_msg_Single_Search_p=1&redirect_Seqno=7664281921

User-defined PMC Bus connector (J14)

New Input/Outputs added to J14 (user defined PMC connector)

- Refer to ECO-0695 (in Arena at: https://app.bom.com/changes/detail-summary?change_id=2385697496&)
- **ECO released**: ~11 Jul, 16
- **Approximate S/N cut-in (based on date above):** > 4710

From Schematic: 1221-1001-0200 rev 4:



Temperature Sensor

- ➢ This board has a temperature sensor built-in. so the drivers now have a call for this function.

TSYNC_HW_GetTemperature() Reads the current board temperature in counts.

PCB Board Reset

When all the LED's on the board are lit, that means it has been reset. When the 3.3v rail recovers, the board should come out of reset mode and start running again.

Power/logic

TSync-cPCI is a 3.3v power/logic card. It is not compatible with 5v systems.

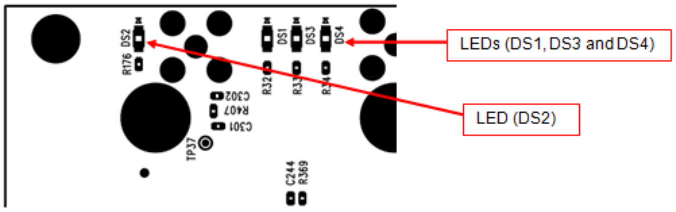
Because of this, a YELLOW key should be installed in the bus connector- to indicate it is a 3.3v board

Email from Tim Tetreault to Mike Vinskus (6/26/12)
Your testing confirms what we were thinking is the problem. The 3.3v rail is dropping causing the low voltage IC on the board to hold it in reset mode. The threshold for the low voltage IC is 3.07v +/- 0.5% typ acc. So if you do the math, our spec for the 3.3v rail is 5% or about 3.14v, just above the threshold voltage.

When all the LED's on the board are light, that means it has been reset. When the 3.3v rail recovers, the board should come out of reset mode and start running again.

Note: The tolerance for the 3.3v rail is the same between both TSync-PCIe and TSync-CPCI boards (5%). The current draw is different.

Status LEDs: DS1, DS3, DS4 (edge of board) and DS2 (FPGA Programming)



A) FPGA programming header and LED (DS2)

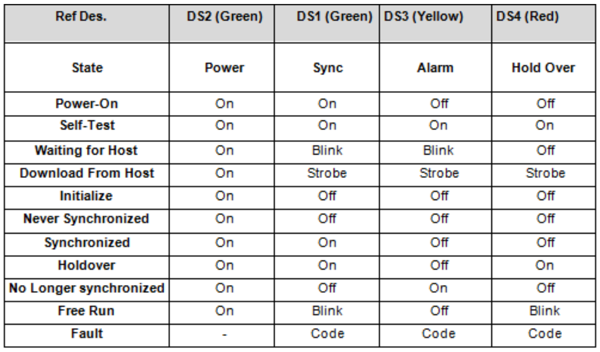
Header/Jumper for programming

LED Fault codes

- The code indicates the fault condition. It blinks the number of times indicated with a 2-second pause between each set.

  1 Blink = FPGA programming error
  2 Blinks = Failure to decompress
  3 Blinks = CRC failure writing to flash
  4 Blinks = Self-test failure
  5 Blinks = Timing system failure

L) DS1/DS2, DS3 and DS4: Three color LEDs (Red, Yellow, Green on edge of board)



Ref Des.	DS2 (Green)	DS1 (Green)	DS3 (Yellow)	DS4 (Red)
State	Power	Sync	Alarm	Hold Over
Power-On	On	On	Off	Off
Self-Test	On	On	On	On
Waiting for Host	On	Blink	Blink	Off
Download From Host	On	Strobe	Strobe	Strobe
Initialize	On	Off	Off	Off
Never Synchronized	On	Off	Off	Off
Synchronized	On	On	Off	Off
Holdover	On	On	Off	On
No Longer synchronized	On	Off	On	Off
Free Run	On	Blink	Off	Blink
Fault	-	Code	Code	Code

refer to Case 210465 Blink Blink Blink

Ability to command the LEDs (manual mode)

The LEDs operate in certain modes by default, but each LED can be configured independently to display any mode, including a manual mode. In manual mode, the user can set LEDs to *on*, *off*, or *blink*.

Inputs

**EPP0 (External IPPS input -via the Timing connector)

- Amplitude: 0V to +5.5V, +0.8V VIL, +2.0V VIH
- 1Hz Pulse, Rising Edge or Falling Edge Active (selectable)
- 100 ns minimum pulse width
- input Impedance: <150 pFcapacitive

**IRIG inputs (via the Timing connector)

- AM and DCLS input

**External 10 MHz input

- Not available on Tsync-cPCI or Tsync-PMC boards (only available on CTC-cPCI boards)

Synchronization to input references/priority lists

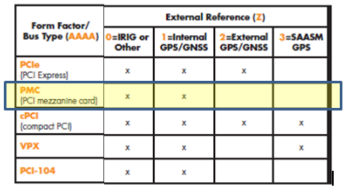
- 1PPS reference and Time Reference required for sync.
  - **Time references**: internal GPS receiver, external GPS receiver (via NMEA 0183), Host reference (requires external 1PPS reference)
  - **1PPS references**: internal GPS receiver, external GPS receiver (via NMEA 0183), external 1PPS input, external 10 MHz input, Self reference.
- Can use the factory default priority list or user may define a proprietary priority list
- Each input reference can be enabled or disabled, as desired.

Input Reference Monitor

(this is Just like TSync-PCIe)

- A default table, which provides the default reference pairings in timing accuracy priority
- A working table, which is the table used for selecting reference inputs
- A user table, which can be stored persistently and, if present, will be loaded into the working table at startup

**GPS/Glonass receiver

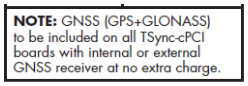


Form Factor/ Bus Type (AAAA)	External Reference (Z)			
	0=IRIG or Other	1=Internal GPS/GNSS	2=External GPS/GNSS	3=SAASM GPS
PCIe (PCI Express)	x	x	x	
PMC (PCI mezzanine card)	x	x		
cPCI (compact PCI)	x	x	x	x
VPX	x	x		x
PCI-104	x	x		

Available GNSS Receiver configurations

- (0) IRIG input only
- (1) Internal GPS (connected to Model 8230 GG antenna)

GNSS + Glonass



NOTE: GNSS (GPS+GLONASS) to be included on all TSync-cPCI boards with internal or external GNSS receiver at no extra charge.

**SS-OPT-GNS:

- Adds Glonass L1 1602MHz (now a standard feature)

Per Tim Tetreault (13 Nov 2013) It's been decided that Glonass is now going to be a standard feature for all TSync-cPCI boards (does not need to be purchased). As of now, there are no available API calls dedicated to Glonass (such as enabling or disabling this function). Instead of the GPS API calls reporting up to 12 channels, they add the Glonass satellites to the count of available total satellites being tracked.

***uBlox M8T receiver

UBox receiver Cut-in info

TSync-PCIe conversion from Trimble Res-SMT-GG to uBlox M8T GNSS receiver

- Refer to ECO-0695 (in Arena at: https://app.bom.com/changes/detail-summary?change_id=2385697496&)
- **ECO released**: ~11 Jul, 16
- **Approximate S/N cut-in (based on date above):** S/N 4711 and above (S/N 4711 shipped on 7 Jul 2016).

TSync-PMC conversion from Trimble Res-SMT-GG to uBlox M8T GNSS receiver

- Refer to ECO-00978 (in Arena at: https://app.bom.com/changes/detail-summary?change_id=2386955663&)
- **ECO released**: ~21 Aug, 2017 (updates TSync-PMC firmware from version 1.0.0 to 1.1.0
- **Approximate S/N cut-in (based on date above):** beyond 5965 (S/N 5965 shipped on 23 July, 2017, before the ECO was released).

***Customer desire/need to replace Trimble Res-SMT receiver with UBlox M8T

- Refer to Salesforce Case 256115
- Due to differences in TSync bootloader firmware, TSync-PCIe needs to be returned to factory for this change to be implemented.

Q Keith sent to Dave West

Don M recommended I run this question by you (he believes my thoughts are correct) … A TSync customer in New Zealand (I believe) is asking what needs to be done to switch from Trimble Res-SMT-GG to uBlox M8T GNSSS receiver. Does this conversion have to be performed at the factory, or can it be performed in field? I believe EEPROM needs to be reprogrammed at factory?? Thanks much, Keith

A [2:21 PM] David West (21 Jan 2021)

The bootloader is the guilty party and requires being returned to us.

EEPROM might be able to upgrade in field, but don't really know. Either way, it doesn't matter because bootloader is different

[2:24 PM] David West

I just checked and confirmed the FPGA is the same P/N and rev on them

Galilieo constellation with u-Blox

- Ublox M8T support for Galilieo constellation was added in ublox receiver firmware update v3.0.1 (~Dec 2016)

Need to reset GPS position if the TSync is moved to a new stationary location

B) TSync firmware versions 3.4.7 and above (ECO 1625 for 3.4.7 released 29 March 2018)

- Receiver resurveys after every power-up
  - per the v3.4.7 release notes: "GNSS receiver will now restart survey on power-up or board reset"

M) TSync-PCIe firmware versions prior to v3.4.6 and below

- Receiver Position needs to be deleted to start a new survey

- Perform a **GR_DelPos 0 0** to force a resurvey.

~~**(Unlike SecureSyncs/9400s) the work-around of clearing postion on reboot is not available for TSync series timing boards is not available**~~

- Similar info below is also in the “ublox” section of the custsserviceassist doc.
- The same work-around for SecureSyncs/9400s to delete position/resurvey after each reboot was implemented in TSync firmware upgrade version 3.4.7 (~29 Mar 2018)
- TSync boards with firmware versions prior to v3.4.7, which are synced to GPS after shipment from our factory and then relocated elsewhere will need the GPS position manually cleared before it can sync at the new location.

~~**Email from Keith to Tony DiFlorio (20 Feb 17)** Tim T just clarified that the work-around that we are using for SecureSyncs and NetClock 9400s with a ublox receiver installed (clearing the position hold after each boot-up for a new survey to be performed) is not available as a work-around for the TSync boards with a ublox receiver (or for a Res-SMT-GG receiver with version 1.0.9 firmware installed).~~

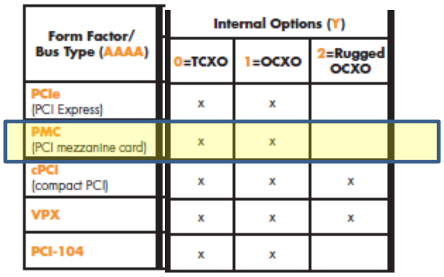
~~The work-around that was added to the SecureSyncs and 9400s needed to be implemented in the “network processor software” (not in the KTS Timing system software). The TSync boards share the KTS timing system software with the SecureSyncs/9400s. But unlike the SecureSyncs/9400s, the TSync boards don’t contain the Network processor software, preventing this work-around from being able to be added to the TSync boards.~~

~~The GNSS receiver is cleared of its position before its shipped from our factory. But if the TSync board is synced to GPS at a location after the board had been received and then relocated thereafter, the TSync board will continue to need its position manually cleared before it can sync at the new location. Unlike SecureSyncs, there are no expected work-arounds/changes expected for the TSync series boards to be able to clear their position automatically.~~

Q. It never achieved stable synchronization -- the XO disciplining state was oscillating wildly between state 2 and 5.

A per Dave S (10 Jun 17) One of the characteristics of the new u-blox receiver is that it does not automatically "resurvey" its position in standard (stationary) mode if the unit is relocated. Unfortunately, the position on this unit does not appear to have been cleared before shipment to allow for resurvey and operation once powered at Google. Deleting the position on the receiver should allow this to resynchronize properly. This can be achieved using the “GR_DelPos 0 0” command. If operating in stationary mode, this will be required whenever the card is relocated. An alternative would be to operate in mobile mode, which doesn’t require the survey.

Oscillators available for TSync-PMC



Form Factor/ Bus Type (AAAA)	Internal Options (Y)		
	0=TCXO	1=OCXO	2=Rugged OCXO
PCIe (PCI Express)	x	x	
PMC (PCI mezzanine card)	x	x	
cPCI (compact PCI)	x	x	x
VPX	x	x	x
PCI-104	x	x	

Available oscillators (TSync-PMC boards)

- **(0) Standard TCXO**: 1ppm
- **(1) Standard OCXO**: 0.5ppm

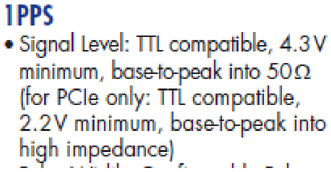
Outputs

**IRIG outputs (refer to "IRIG outputs" in TSync-PCIe section)

(Same as TSync-PCIe)

**(1) 1PPS output

1PPS Output level
Note the 1PPS output level is smaller on the Tsync-PCIe (2.2v min) than it is on the other Tsync family of board (TSync-PMC,CPCI, etc) (at 4.3v min) as indicated on the data sheet



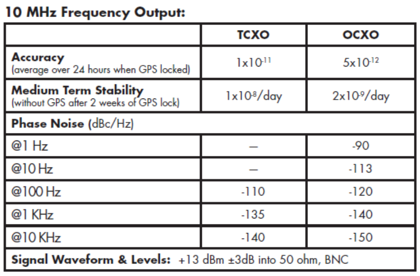
1PPS

- Signal Level: TTL compatible, 4.3V minimum, base-to-peak into 50Ω (for PCIe only: TTL compatible, 2.2V minimum, base-to-peak into high impedance)

** (1) 10 MHz sine wave output (refer to "10 MHz output" in TSync-PCIe section)

- Via Timing Connector

(Same as TSync-PCIe) Refer to: *10MHz output



10 MHz Frequency Output:

	TCXO	OCXO
Accuracy (average over 24 hours when GPS locked)	$1x10^{-11}$	$5x10^{-12}$
Medium Term Stability (without GPS after 2 weeks of GPS lock)	$1x10^{-8}$/day	$2x10^{-9}$/day
Phase Noise (dBc/Hz)		
@1 Hz	–	-90
@10 Hz	–	-113
@100 Hz	-110	-120
@1 KHz	-135	-140
@10 KHz	-140	-150
Signal Waveform & Levels: +13 dBm ±3dB into 50 ohm, BNC		

Serial number (TSync-PMC boards only)

- Obtaining Serial Number via API call/example program **XOGetSerialNo**

(KW 9/18/12, per Tim Tetreault 9/18/12) This is a new command added to the TSync driver specifically for the TSync-cPCI boards. It's not supported currently supported with the TSync-PCIe timing boards.

Firmware/Versions

- **Refer to:** PSB, PSP software updates\TSYNC boards\Tsync Firmware updates\Tsync-PMC

Drivers for TSync-PMC boards

TSync-PMC drivers

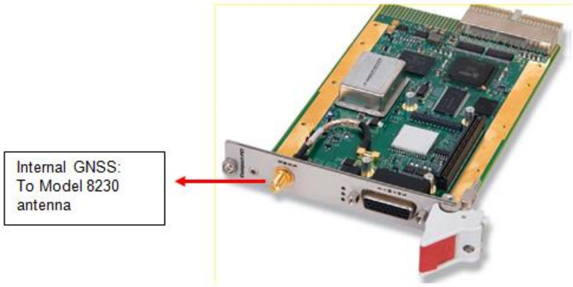
- Refer to: PSB, PSP software updates\TSYNC boards\TSync driver updates
- Link to drivers on our website: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=37
- This board uses the same drivers as the TSync-PCIe board uses.
- This board has a temperature sensor built-in so the drivers now have a call for this function

Interrupts (refer to "interrupts" in TSync-PCIe section)

- (Same as TSync-PCIe)
- Refer to (Tsync-PCIe interrupts) :Interrupt generation/Interrupt Counter

TSync-cPCI (1219-xxxx)

TSync-cPCI



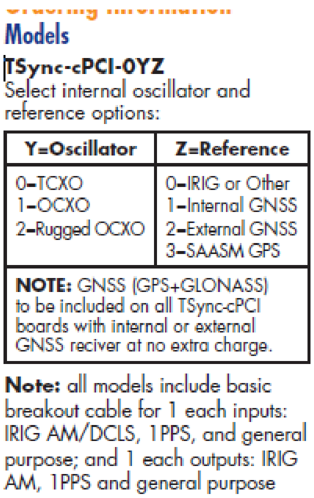
Shortcuts/Links

- **Shortcut to data sheet:** I:\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards
- **Shortcut to manual (1219-5000-0050) in Arena at:** https://app.bom.com/items/detail-spec?item_id=1203165592&version_id=10228803608
- Shortcut to **TSync-CPCI drivers** (same as the TSync-PCIe drivers) in Arena **(1219-SD01-xxxx)**
  - CD assembly we ship containing drivers and guides (**1219-5003-6001**) in Arena https://app.bom.com/items/detail-spec?item_id=1203165725&version_id=10222045518
- Shortcut to **driver guide** (**1219-5001-0050**) in Arena: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002
- **Shortcut to application programmers guide (1219-5002-0050):**
- **Shortcut to firmware and driver version upgrades:** \\Exchange\empshare$\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards
- **Shortcut to schematic (1219-1001-0200)** in the 1219-0000-F000 assembly in Arena at **https://app.bom.com/items/detail-spec?item_id=1202847114&version_id=10479873168&**

Comparison of legacy TPRO/TSAT-cPCI to newer TSync-cPCI: http://partner.spectracomcorp.com/Portals/7/Other/Compact%20PCI%20Timing%20Board%20Update%20-%20Model%20TSync-cPCI.pdf

Product first released: March 2012 (on ECN 2846)

Available configurations



Ordering Information

Models

TSync-cPCI-0YZ

Select internal oscillator and reference options:

Y=Oscillator	Z=Reference
0–TCXO 1–OCXO 2–Rugged OCXO	0–IRIG or Other 1–Internal GNSS 2–External GNSS 3–SAASM GPS
NOTE: GNSS (GPS+GLONASS) to be included on all TSync-cPCI boards with internal or external GNSS reciver at no extra charge.	

Note: all models include basic breakout cable for 1 each inputs: IRIG AM/DCLS, 1PPS, and general purpose; and 1 each outputs: IRIG AM, 1PPS and general purpose

Weight and other spes

- 6.1 oz/173 g (without thermo frame),
- 11.4 oz/323 g (with thermo frame)

Swapping out a legacy TPRO/TSAT-cPCI board for a TSync-cPCI

Email from Tim Tetreault to Sadie (18 Nov 2013)

- The equivalent to the TPRO-cPCI board would be the TSync-cPCI-000.
- The equivalent to the TSAT-cPCI board would be the TSync-cPCI-002.

I want to make sure that everyone understands that the new TSync-cPCI board is not a drop-in replacement for the old KSI boards. It will support all of the functions that the older KSI cPCI board had but customers will need to use our new TSync driver and update their software.

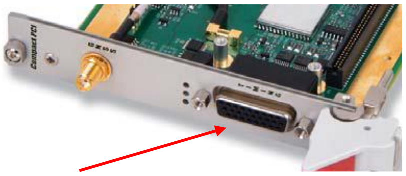
Conduction Cooled TSync-cPCI

- ➢ As of at least Feb, 2015 – not currently available

Tsync-cPCI memory

- Refer to: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync-PMC and cPCI

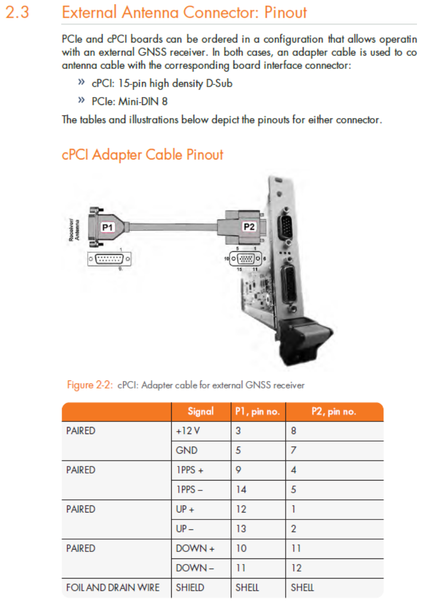
Timing Connector, adapter cable and Breakout cables



Timing connector (Refer to “Timing connector” in TSync-PCIe section)

- Refer to Timing connector in TSync-PCIe section of this document

TSync-cPCI Adapter cable that goes between Timing connector and breakout cable



2.3 External Antenna Connector: Pinout

PCIe and cPCI boards can be ordered in a configuration that allows operatin with an external GNSS receiver. In both cases, an adapter cable is used to co antenna cable with the corresponding board interface connector:

- cPCI: 15-pin high density D-Sub
- PCIe: Mini-DIN 8

The tables and illustrations below depict the pinouts for either connector.

cPCI Adapter Cable Pinout

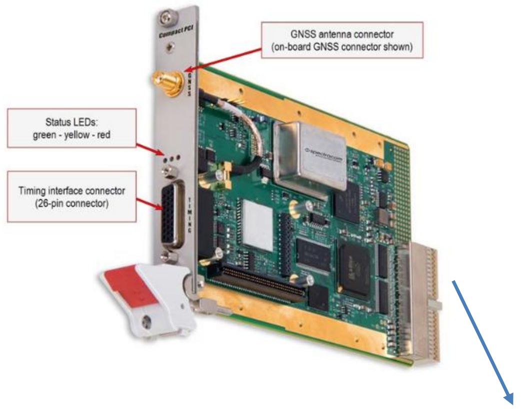
Figure 2-2: cPCI: Adapter cable for external GNSS receiver

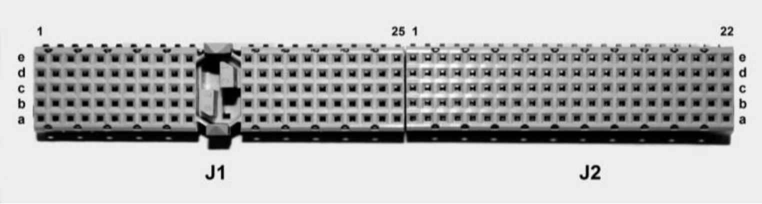
	Signal	P1, pin no.	P2, pin no.
PAIRED	+12 V	3	8
	GND	5	7
PAIRED	1PPS +	9	4
	1PPS –	14	5
PAIRED	UP +	12	1
	UP –	13	2
PAIRED	DOWN +	10	11
	DOWN –	11	12
FOIL AND DRAIN WIRE	SHIELD	SHELL	SHELL

Breakout cables for Tsync-cPCI (Refer to “Breakout cables” in TSync-PCIe section)

- Refer to breakout cables in TSync-PCIe section of this document

TSync-cPCI Bus connectors (J1 and J2)

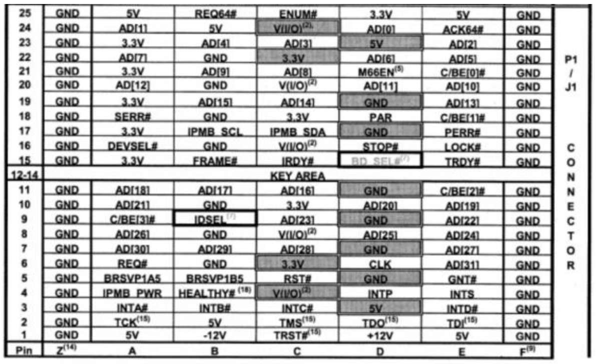




Note: Refer to the TSync-cPCI manual for the pin-outs (likely on pages 3-1 through 3-3)

- **Shortcut to TSync-cPCI manual (1219-5000-0050) in Arena at:** https://app.bom.com/items/detail-spec?item_id=1203165592&version_id=10228803608

The interface to the ePCI card is defined by the PICMG 2.0 R3.0 CompactPCI specifications. I have included them below:



25	GND	5V	REQ64#	ENUM#	3.3V	5V	GND
24	GND	AD[1]	5V	V(I/O)[2]	AD[0]	ACK64#	GND
23	GND	3.3V	AD[4]	AD[3]	5V	AD[2]	GND
22	GND	AD[7]	GND	3.3V	AD[6]	AD[5]	GND
21	GND	3.3V	AD[9]	AD[8]	M66EN[5]	C/BE[0]#	GND
20	GND	AD[12]	GND	V(I/O)[2]	AD[11]	AD[10]	GND
19	GND	3.3V	AD[15]	AD[14]	GND	AD[13]	GND
18	GND	SERR#	GND	3.3V	PAR	C/BE[1]#	GND
17	GND	3.3V	IPMB_SCL	IPMB_SDA	GND	PERR#	GND
16	GND	DEVSEL#	GND	V(I/O)[2]	STOP#	LOCK#	GND
15	GND	3.3V	FRAME#	IRDY#	BD_SEL#[7]	TRDY#	GND
12-14				KEY AREA			
11	GND	AD[18]	AD[17]	AD[16]	GND	C/BE[2]#	GND
10	GND	AD[21]	GND	3.3V	AD[20]	AD[19]	GND
9	GND	C/BE[3]#	IDSEL[7]	AD[23]	GND	AD[22]	GND
8	GND	AD[26]	GND	V(I/O)[2]	AD[25]	AD[24]	GND
7	GND	AD[30]	AD[29]	AD[28]	GND	AD[27]	GND
6	GND	REQ#	GND	3.3V	CLK	AD[31]	GND
5	GND	BRSVP1A5	BRSVP1B5	RST#	GND	GNT#	GND
4	GND	IPMB_PWR	HEALTHY#[18]	V(I/O)[2]	INTP	INTS	GND
3	GND	INTA#	INTB#	INTC#	5V	INTD#	GND
2	GND	TCK[15]	5V	TMS[15]	TDO[15]	TDI[15]	GND
1	GND	5V	-12V	TRST#[15]	+12V	5V	GND
Pin	Z[14]	A	B	C	D	E	F[9]

P1 / J1 CONNECTOR

cPCI Specifications, PCB height (3U) /Chassis size (3U, 6U)

- Bus Interface: Universal Signaling Voltage 3.3 V/5 V
- Bus Speed: 32bit address @ 33/66 MHz
- 3U Compact PCI (cPCI) Compliant to PICMG 2.0 r3.0 100 mm x 160 mm (3U card size)
  - ➢ IInstalls in a **3U chassis**

6U chassis:

Q Does someone know if we have a solution to install a Tsync-cPCI in a cPCI 6U rack

A per Dave Sohn (11/19/2019) « I'm not aware of a way to install our 3U board into a 6U chassis. »

Note: Keith responded with a recommendation to check with Kontron to see if they happen to have any available extender cards/boards for this type of install

…my recommendation is to have your customer check with company named "**Kontron**" (https://www.kontron.com/products/boards-and-standard-form-factors/cpci). We have used their chassis' before/and I have seen several customers also using them also. Kontron appears to offer many different components for 6U chassis/

Command Lspci is not showing / listing Spectracom Tsync-cPCI card

- ➢ Refer to Salesforce case 24172
- ➢ Board should be detected in the system, even if the drivers haven't been installed.

Email from Dave Lorah (22 Feb 17) We were discussing this problem more today and as I said before the board should be recognized in a standard cPCI backplane. Even with no drivers installed the Lspci should recognize the board.
So we discussed the problem and we think there must be some kind of problem with the cPCI backplane or the SBC module.

There are a few things we would like the customer to check:

1. Check to make sure the backplane supplies 3.3V, 5V, +12V and -12V are all good.
2. Remove all other cPCI boards in the system in case there is some kind of conflict with another board.
3. Use a different slot for the TSync-cPCI board. Some slots may be dedicated for certain functions.
4. If another different type of cPCI system is available for use, try installing the TSync-cPCI board in that system to test if the board is good. We do not know if the TSync-cPCI board is working or not.

I hope this helps determine where the problem lies.

Temperature Sensor

- ➢ This board has a temperature sensor built-in. So the drivers now have a call for this function.

TSYNC_HW_GetTemperature() Reads the current board temperature in counts.

PCB Board Reset

When all the LED's on the board are lit, that means it has been reset. When the 3.3v rail recovers, the board should come out of reset mode and start running again.

Power/logic

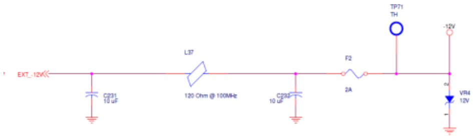
- TSync-cPCI is a 3.3v power/logic card. It is not compatible with 5v systems.
- Because of this, a **YELLOW** key should be installed in the bus connector- to indicate it is a 3.3v board

Email from Tim Tetreault to Mike Vinskus (6/26/12)
Your testing confirms what we were thinking is the problem. The 3.3v rail is dropping causing the low voltage IC on the board to hold it in reset mode. The threshold for the low voltage IC is 3.07v +/- 0.5% typ acc. So if you do the math, our spec for the 3.3v rail is 5% or about 3.14v, just above the threshold voltage.

When all the LED's on the board are light, that means it has been reset. When the 3.3v rail recovers, the board should come out of reset mode and start running again.

Note: The tolerance for the 3.3v rail is the same between both TSync-PCIe and TSync-CPCI boards (5%). The current draw is different.

-12vdc



- Required for IRIG input and IRIG output
- Gain control for TCXO/OCXO osc
- Measure at TP71
- Fuse F2

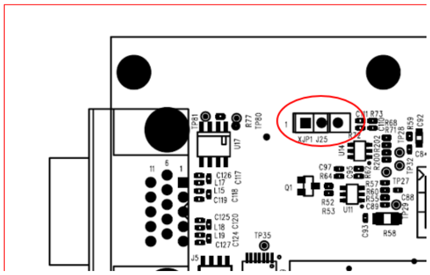
TSync-cPCI LEDs: DS1 (FPGA Programming) and DS2 (three color status on edge of board)

Note: This section is specific to **TSync-cPCI** boards. For other variants (such as TSync-PMC, TSync-cPCI, etc) refer to its specific section of this document

A) FPGA programming header (J25) and green Program load LED (DS1)

1. **Header/Jumper for programming (J25)**

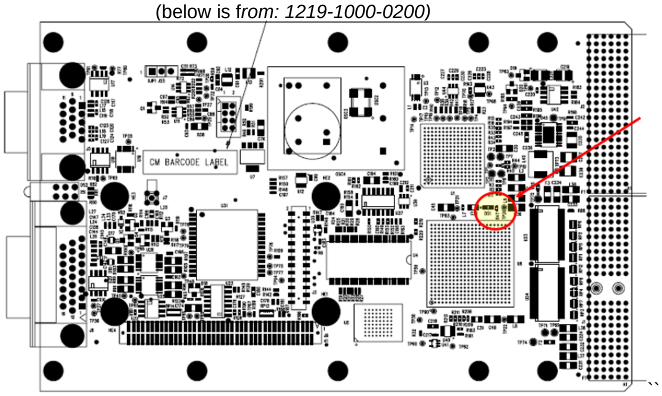
   ➢ Located on **top** of board



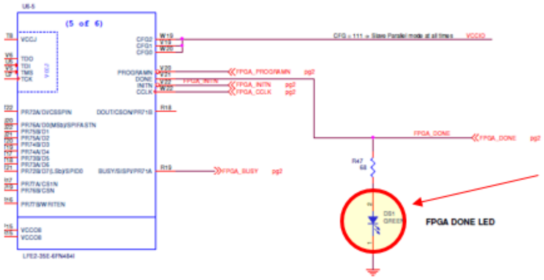
2. **Green FPGA Program load LED (DS1)**

   ➢ Unlike TSync-PCIE boards, the green FPGA Programming LED (DS1) is on the **TOP** of the board

(below is fr*om: 1219-1000-0200)*



(below is from: 1219-1001-0200)



NOTE: DS1 flashing (after program) means either jumper J25 is in wrong position, or problem with the PCB.

N) Three color status LED on edge of the board (Sync, Holdover, etc)

- Refer to "**LEDs**" in the **TSync-PCIe** section of this document (these LEDs are the same as the TSync-PCIe boards): TSync-PCIe FPGA program header (J25) / LED's (Programming and Status) / LEDs upon reboot

Inputs for TSync-cPCI

**IRIG inputs (via Timing connector)

- ➢ Refer to the TSync-PCIe section of this document (same as the TSync-PCIe) : **IRIG input synchronization

External 10 MHz input

- ➢ Not available on Tsync-cPCI (only available on CTC-cPCI boards)

** External 1PPS input (EPP 0) (Via Timing connector)

- • Amplitude: 0V to +5.5V, +0.8V VIL, +2.0V VIH
- • 1Hz Pulse, Rising Edge or Falling Edge Active (selectable)
- • 100 ns minimum pulse width
- • input impedance: <150 pFcapacitive

Synchronization to input references/priority lists

- ➢ Does not accept IRIG input!
- ➢ 1PPS reference and Time Reference required for sync.

Time references: internal GPS receiver, external GPS receiver (via NMEA 0183), Host reference (requires external 1PPS reference)

1PPS references: internal GPS receiver, external GPS receiver (via NMEA 0183), external 1PPS input, external 10 MHz input, Self reference.

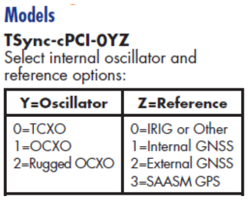
- ➢ Can use the factory default priority list or user may define a proprietary priority list
- ➢ Each input reference can be enabled or disabled, as desired.

Input Reference Monitor

(Just like TSync-PCIe)

- ➢ A default table, which provides the default reference pairings in timing accuracy priority
- ➢ A working table, which is the table used for selecting reference inputs
- ➢ A user table, which can be stored persistently and, if present, will be loaded into the working table at startup

**GPS/Glonass receiver



Models

TSync-cPCI-0YZ
Select internal oscillator and
reference options:

Y=Oscillator	Z=Reference
0=TCXO 1=OCXO 2=Rugged OCXO	0=IRIG or Other 1=Internal GNSS 2=External GNSS 3=SAASM GPS

Available Receiver configurations

- Internal GPS (connected to Model 8230 GG antenna)
- External GPS (connected to Acutime GG antenna)
  - Refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- SAASM (GB-GRAM only)

Cut-over from Trimble Res-SMT-GG to uBlox receiver receivers

Summary: "This ECO will convert the cPCI board over to the new U-BLOX GPS receiver.

Changed the buffer amp for the oscillator DAC to a single rail version.

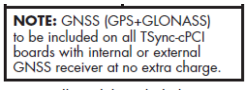
Changed the +11V_ANA regulator to new lower noise version.

- Refer to ECO 0697 (in Arena) https://app.bom.com/changes/detail-summary?change_id=2385697537&
- Jan 2017

TSync-cPCI boads are SAASM receiver-capable

- Can be shipped with a Rockwell Collins GB-GRAM receiver
- For SAASM receiver-related info, refer to ("Model 1204-1A"): U:\Engineering\SAASM-FOUO\CustomerService\SecureSync\1204-1A (GB-GRAM)
- SMA to Type N adapter cable included with convection cooled models

GNSS + Glonass (for commercial GPS receivers)



NOTE: GNSS (GPS+GLONASS) to be included on all TSync-cPCI boards with internal or external GNSS receiver at no extra charge.

****SS-OPT-GNS:** Adds Glonass L1 1602MHz (now a standard feature)

Per Tim Tetreault (13 Nov 2013) It's been decided that Glonass is now going to be a standard feature for all TSync-cPCI boards (does not need to be purchased). As of now, there are no available API calls dedicated to Glonass (such as enabling or disabling this function). Instead of the GPS API calls reporting up to 12 channels, they add the Glonass satellites to the count of available total satellites being tracked.

AGPS (A-GPS, Assisted GPS) / tsync_agps.h file

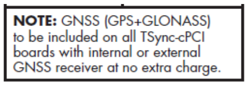
- The Tsync driver contains a file called **tsync_agps.h.** This file is to support AGPS in the SecureSync/9483. As of at least Feb 2015. It's not supported/compatible with the Tsync family timing boards.
- The TSync boards do not support AGPS and there are no intentions at this time (Frb 2015) to add AGPS to the timing boards
- While looking at this, I went through the linux folder on the CD and began unzipping things and found a file called tsync_agps.h, and just made a copy in the windows source tree to see if I could progress. It did seem to fix the immediate compiler errors.

Reply from Morgan (16 Feb 14) From talking to our engineer's the TSync boards do not support Assisted GPS. And at this time, I'm not aware of any intentions of it being added. Any reference that you may have seen to Assisted GPS in the Tsync driver is strictly due to this same driver also being shared with SecureSync, which does support GPS. So there is no tsync_agps.h file present in the driver.

He also said that you shouldn't be having any issues compiling the driver that would be related to Assisted PS. But if you are getting errors when you are compiling or using the TSync driver, please send me a screenshots of the error messages this will help us diagnose the problem.

Also the TPro are a legacy boards and the TSync are the new one. The information is combined on the CD because we have customer's that have upgraded from the TPro and are familiar with the API calls. You should be using the TSync API. Let me know if this helps.

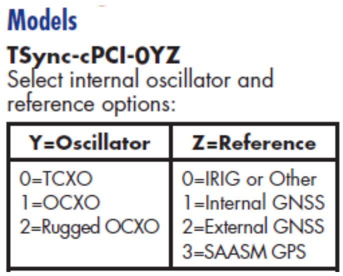
GNSS + Glonass (for commercial GPS receivers)



NOTE: GNSS (GPS+GLONASS) to be included on all TSync-cPCI boards with internal or external GNSS receiver at no extra charge.

****SS-OPT-GNS:** Adds Glonass L1 1602MHz (now a standard feature)
Per Tim Tetreault (13 Nov 2013) It's been decided that Glonass is now going to be a standard feature for all TSync-cPCI boards (does not need to be purchased). As of now, there are no available API calls dedicated to Glonass (such as enabling or disabling this function). Instead of the GPS API calls reporting up to 12 channels, they add the Glonass satellites to the count of available total satellites being tracked.

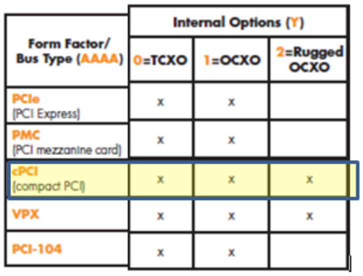
Oscillators



Models

TSync-cPCI-0YZ
Select internal oscillator and reference options:

Y=Oscillator	Z=Reference
0=TCXO 1=OCXO 2=Rugged OCXO	0=IRIG or Other 1=Internal GNSS 2=External GNSS 3=SAASM GPS



Form Factor/ Bus Type (AAAA)	Internal Options (Y)		
	0=TCXO	1=OCXO	2=Rugged OCXO
PCIe (PCI Express)	x	x	
PMC (PCI mezzanine card)	x	x	
cPCI (compact PCI)	x	x	x
VPX	x	x	x
PCI-104	x	x	

Available oscillators (TSync-cPCI boards)

- (0) Standard TCXO: 1ppm
- (1) Standard OCXO: 0.5ppm
- **(2) Rugged OCXO**: 0.5ppb., enhanced shock and vibe specs

➢ Refer to the TSync-PCIe section of this document for more info (same as the TSync-PCIe)

Outputs for TSync-cPCI

**IRIG outputs

➢ Via Timing Connector

➢ IRIG AM and DCLS outputs available

➢ Refer to the TSync-PCIe section of this doc (same as the TSync-PCIe): *IRIG output

**(1) 1PPS output

- Refer to the TSync-PCIe section of this doc (same as the TSync-PCIe): *1PPS output

**(1) 10 MHz sine wave output

- Refer to the TSync-PCIe section of this doc (same as the TSync-PCIe): *10MHz output

Serial number

Obtaining Serial Number via API call/example program

XOGetSerialNo:

(KW 9/18/12, per Tim Tetreault 9/18/12) This is a new command added to the TSync driver specifically for the TSync-cPCI boards. It's not supported currently supported with the TSync-PCIe timing boards.

Firmware/Versions

Refer to: I:\Customer Service\PSB, PSP software updates\TSYNC boards\Tsync Firmware updates\TSync-PMC and TSync-cPCI

- Version 3.01 (ECN 3210 Aug, 2013)

Adds support for newer Trimble Res-SMT-GG (GNSS) receiver.
Previous rev was version 2.20

- Version 2.20

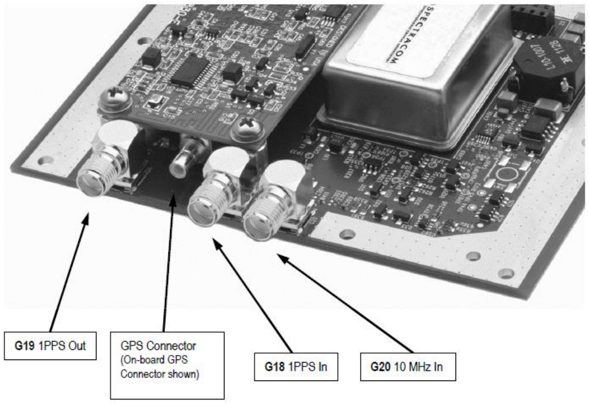
Drivers for TSync-cPCI boards

- Refer to: PSB, PSP software updates\TSYNC boards\TSync driver updates
- Link to drivers on our website: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=37
- This board uses the same drivers as the TSync-PCIe board uses.
- This board has a temperature sensor built-in so the drivers now have a call for this function

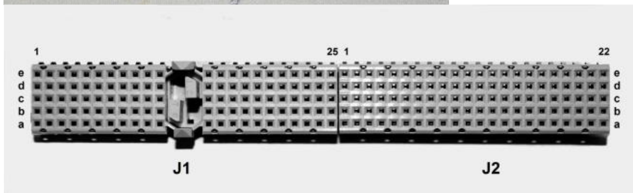
Interrupts

- (Same as TSync-PCIe)
- Refer to (Tsync-PCIe interrupts in this document): Interrupt generation/Interrupt Counter

CTC-cPCI (CTC/Blitz) and Tsync-cPCI-121(“TSync-cPCI-CC” commercial version of CTC-cPCI)



Optional thermal frame



- ➢ Shortcut to data sheet: I:\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards
- ➢ Shortcut to manual (1219-5000-0050) in Arena: https://app.bom.com/files/detail-summary?file_master_id=1233732999&file_id=1744796872
- ➢ Shortcut to driver guide (1219-5001-0050) in Arena: https://app.bom.com/files/list-main
- ➢ Shortcut to application programmers guide (1219-5002-0050) in Arena: https://app.bom.com/files/detail-summary?file_master_id=1233732999&file_id=1744796872&orb_msg_Single_Search_p=1&redirect_Seqno=6333576648
- ➢ Shortcut to TSync-CPCI drivers (same as the TSync-PCIe drivers):
- ➢ Shortcut to firmware and driver version upgrades: PSB, PSP software updates\TSYNC boards

Comparison of legacy TPRO/TSAT-cPCI to newer TSync-cPCI: http://partner.spectracomcorp.com/Portals/7/Other/Compact%20PCI%20Timing%20Board%20Update%20-%20Model%20TSync-cPCI.pdf

Product first released: March 2012 (on ECN 2846)

Part Numbers/ITAR

1205-0000-0600: CTC-cPCI (Blitz/Chronos/CTC board for Lockheed Martin)

ITAR-controlled device. They can come back to us with no restrictions. Treat them as ITAR equipment. No mods/steps are required before sending them back to us for repair.

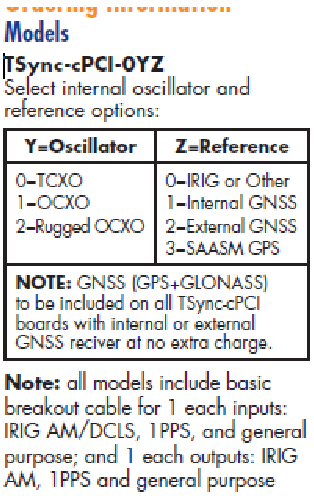
1205-0023-0600: CTC-cPCI with conformal coating (Blitz/Chronos/CTC board for Lockheed Martin)

ITAR-controlled device. They can come back to us with no restrictions. Treat them as ITAR equipment. No mods/steps are required before sending them back to us for repair.

TSync-cPCI-121 (Tsync-cPCI with Thermal Frame and Rugged OCXO)
https://na8.salesforce.com/01tC00000043jal?srPos=0&srKp=01t

The only difference between this and the Lockheed Martin CTC board is the FPGA firmware. And the Timing connector installed on the Tsync-cPCI board. Otherwise, they are identical devices). Not ITAR device.

Available configurations



Ordering Information

Models

TSync-cPCI-0YZ
Select internal oscillator and reference options:

Y=Oscillator	Z=Reference
0–TCXO 1–OCXO 2–Rugged OCXO	0–IRIG or Other 1–Internal GNSS 2–External GNSS 3–SAASM GPS

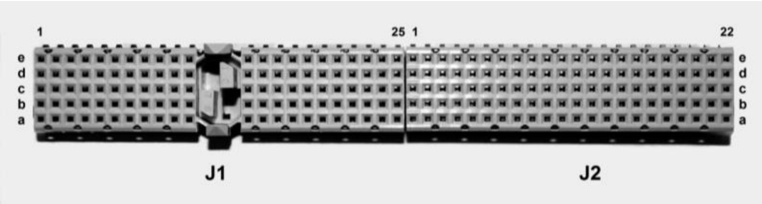
NOTE: GNSS (GPS+GLONASS) to be included on all TSync-cPCI boards with internal or external GNSS reciver at no extra charge.

Note: all models include basic breakout cable for 1 each inputs: IRIG AM/DCLS, 1PPS, and general purpose; and 1 each outputs: IRIG AM, 1PPS and general purpose

Timing Connector/ Breakout cables

- None available
- Only the TSync-cPCI boards have the Timing connector installed (The CTC-cPCI boards don't have a Timing connector).

Bus connectors (J1 and J2)



Note: Refer to the TSync-cPCI manual for the pin-outs (likely on pages 3-1 through 3-3)
Link to manual: I:\New Released\Manuals\1205-xxxx-xxxx

Temperature Sensor

- This board has a temperature sensor built-in. so the drivers now have a call for this function.

TSYNC_HW_GetTemperature() Reads the current board temperature in counts.

PCB Board Reset

When all the LED's on the board are lit, that means it has been reset. When the 3.3v rail recovers, the board should come out of reset mode and start running again.

Power/logic

TSync-cPCI is a 3.3v power/logic card. It is not compatible with 5v systems.

Because of this, a YELLOW key should be installed in the bus connector- to indicate it is a 3.3v board

Email from Tim Tetreault to Mike Vinskus (6/26/12)
Your testing confirms what we were thinking is the problem. The 3.3v rail is dropping causing the low voltage IC on the board to hold it in reset mode. The threshold for the low voltage IC is 3.07v +/- 0.5% typ acc. So if you do the math, our spec for the 3.3v rail is 5% or about 3.14v, just above the threshold voltage.

When all the LED's on the board are light, that means it has been reset. When the 3.3v rail recovers, the board should come out of reset mode and start running again.

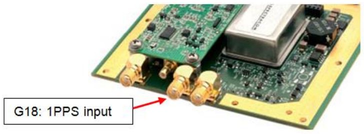
Note: The tolerance for the 3.3v rail is the same between both TSync-PCIe and TSync-CPCI boards (5%). The current draw is different.

LEDs

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

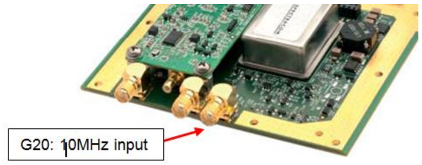
Inputs

**1PPS input



- 1PPS input is on SMA connector (G18)
- High impedance input

**10 MHz input



- **Connector**: SMA jack (G20)
- **Input Impedance**: 50 ohms
- **Level**: 0 dBm (0.64 Vp-p)

**IRIG inputs

CTC boards: None available

TSync-cPCI: AM and DCLS input

> **Note (8/21/12)**: There are a couple of references to IRIG in the early rev of the manuals- These are just inadvertent carry-overs from the TSync-PCIe manual. Tim Tetreault confirmed there are currently no IRIG inputs or outputs.

Synchronization to input references/priority lists

- Does not accept IRIG input!
- 1PPS reference and Time Reference required for sync.

Time references: internal GPS receiver, external GPS receiver (via NMEA 0183), Host reference (requires external 1PPS reference)

1PPS references: internal GPS receiver, external GPS receiver (via NMEA 0183), external 1PPS input, external 10 MHz input, Self reference.

- Can use the factory default priority list or user may define a proprietary priority list
- Each input reference can be enabled or disabled, as desired.

Input Reference Monitor

(Just like TSync-PCIe)

- A default table, which provides the default reference pairings in timing accuracy priority
- A working table, which is the table used for selecting reference inputs
- A user table, which can be stored persistently and, if present, will be loaded into the working table at startup

**GPS/Glonass receiver

Available Receiver configurations

- External GPS (connected to Acutime GG antenna) refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- SAASM (GB-GRAM only)

External GPS receiver input

NMEA input: TSync-cPCI board accepts NMEA 0183 (GGA, ZDA and RMC) input from an external GPS receiver

RS-232 port on J2 connector

- Pin E17 = Rx (Receive)
- Pin E18 = Tx (Transmit)

(**Note**: As of at least 8/22/12,Tx pin E18 is for future use only and is not currently supported). Only the RX (receive) pin for input for receiving data from an external GPS receiver is supported).

(8/22/12 Email to Don Burchette based on info from Tim Tetreault)
NMEA1, located at E17/E18 on the J2 connector, is an input from an external GPS receiver that can be used as an input reference for the timing board, if desired. Currently, only the "RX1" (E17) is supported. The "Tx1" (E18) was added for future use where two-way communication with an external GPS receiver is required. This pin is not currently supported.

External 10 MHz input

- Accepts 10MHz input on SMA connector (G20) located on edge of card (near the GPS receiver).
- 50 ohm input impedance.
- Input level 0dBm (0.64Vp-p).
- Signature Control is available

Oscillators

Rugged OCXO (Ruggedized) oscillator

- **Shock and vibe Specs**: MIL-STD-810F

Outputs

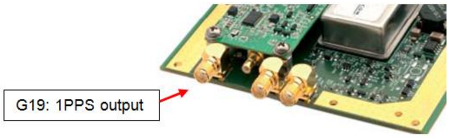
**IRIG outputs

- No IRIG outputs available

Note (8/21/12): There are a couple of references to IRIG in the early rev of the manual. These are just inadvertent carry-overs from the TSync-PCIe manual. Tim Tetreault confirmed there are currently no IRIG inputs or outputs.

**1PPS outputs (standard and LVDS)

- Standard (3.3v) 1PPS output

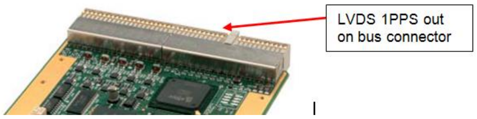


- Standard1PPS output is on SMA connector (G19) located on edge of card (near the GPS receiver).
- High impedance
- Output is 3.3vp
- Minimum pulse width is 1 microsecond

****Known potential issue that can affect the 1PPS outputs (such as drifting)

- Refer to Lockheed Martin cases such as 13797 (https://na8.salesforce.com/500C000000Xc05C)
- Earlier rev boards could potentially have the EEPROM powered off of the 1PPS input reference, even if the system is powered-down. This can potentially corrupt the Reference Priority table (and the oscillator calibration data) stored in the EEPROM.
- A resistor was added later on to prevent the EEPROM from being powered off of the 1PPS input, preventing this issue from occurring.
- Resetting the Reference Priority table using the **RS_SetFactDef** API call may fix this issue, as long as the oscillator cal data is still intact. Note that the Reference Priority is automatically configured with their desired settings each time it powers up. There is no need to reconfigure the table back to their settings, after it's been reset.
- Resetting the Cal data will require a special patch firmware update be installed. Best bet is to RMA it to fix it here, if at all possible.

****LVDS (Low Voltage Differential Signal) 1PPS outputs



1PPS Differential Outputs via J2 (cPCI bus connector)

- Six (6) differential pairs: 3.3 Vp-p,1.5 V common mode
- Three (3) LVDS differential pairs
- Impedance: 100 ohm
- Pulse width: 1 ms
- Rise time: <10 ns

**10 MHz outputs (standard and LVDS)

- 10 MHz Sine-wave outputs

Total of ten (10) 10 MHz outputs are available on connector J2:

- Five (5) are 50 ohm transformer coupled
- Five (5) are 120 ohm impedance

**10 MHz LVDS (Low Voltage Differential Signal)

Connector: J2 (cPCI bus connector)

Three (3) LVDS differential pairs:

Pins: E11/E12,   E13/E14, E15/E16
100 OHM impedance

Excessive phase noise

ECN 3035 (Oct 2012): This ECN will fix the Phase Noise issues that we have been seeing on about 20% of our boards. This was occurring on the 10MHz outputs. Shifting the operating frequency of the 5V to -8V switching regulator fixes the phase noise failures. The change requires changing R399 to 68k ohm

---

**NMEA output Serial port

(2) NMEA 0183 (GGA, ZDA and RMC) outputs are available via RS-232

RS-232 port on J2 connector

- Pin E21 = Tx

Note: No Rx (Receive) pin is present or needed. This is an output-only so bi-directional comms are not required. See note below.

Q. As the chassis and backplane provider, we have provided our customer, TriaSys, with NMEA Tx1 and Rx1 and now they are requesting NMEA2. There is only Tx2 available, no Rx2 on CPCI J2. Does NMEA2 parrot the Tx1 output? Why no Rx2?

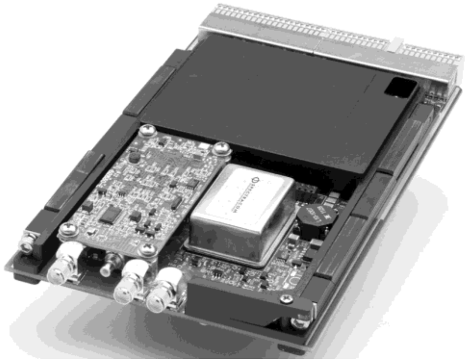
A (8/22/12 Email to Don Burchette based on info from Tim Tetreault)
To answer your question, the connections "NMEA1" and "NMEA2" have 2 separate functions, as described below:

NMEA2, located at E21 on the J2 connector, is an output only. It is a way for another system to receive the disciplined NMEA signal from the cPCI J2 connector. It is the disciplined NMEA message generated by the cPCI board, not a "parrot" of the GPS signal input. So if the time references were to be disconnected from the cPCI board, the board would still continue to transmit the NMEA message referenced to the board's 1pps signal.

As NMEA2 is an output only (it is not a be-directioal port), there is no associated "Rx2" (receive line) either required or present for NMEA2. The output data stream is present every second on "Tx2" with no input required for this to occur ("Rx2" would have only been required to be present if an external input was needed for the NMEA output data to be present every second).

Options

**Available Thermal Frame



**SS-OPT-GNS

- Adds Glonass L1 1602MHz (now a standard feature)

Per Tim Tetreault (13 Nov 2013) It's been decided that Glonass is now going to be a standard feature for all TSync-cPCI boards (does not need to be purchased). As of now, there are no available API calls dedicated to Glonass (such as enabling or disabling this function). Instead of the GPS API calls reporting up to 12 channels, hey add the Glonass satellites to the count of available total satellites being tracked.

.

Serial number

Obtaining Serial Number via API call/example program

XOGetSerialNo: (KW 9/18/12, per Tim Tetreault 9/18/12) This is a new command added to the TSync driver specifically for the TSync-cPCI boards. It's not supported currently supported with the TSync-PCIe timing boards.

Firmware/Versions

- Version 3.01 (ECN 3210 Aug, 2013)

Adds support for newer Trimble ResSMT (GNSS) receiver.

Previous rev was version 2.20

- Version 2.20

Drivers for TSync-cPCI boards

- This board uses the same drivers as the TSync-PCIe board uses.
- This board has a temperature sensor built-in so the drivers now have a call for this function
- Link to drivers on our website:

Interrupts

- (Same as TSync-PCIe)
- Refer to (Tsync-PCIe interrupts) :Interrupt generation/Interrupt Counter

---

HOST PMC BUS interface

PMC bus connector pin-out information

PMC bus speeds supports

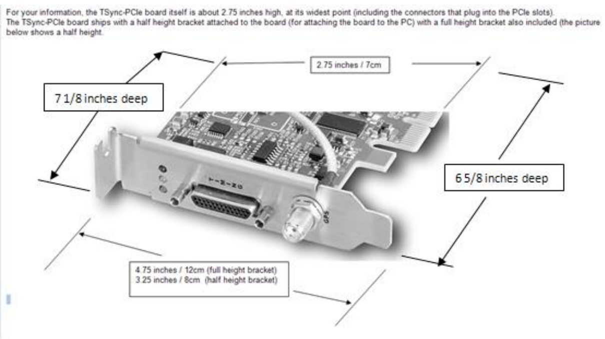
HARDWARE (Specs and FAQs about the board)

Agency approval

TUV and UL: The TSync-PCIe boards are not TUV or UL approved.
CE: The TSync boards are CE approved. Link to Declaration of Conformity:
https://oroliagroupemeamicrosoftonlinecom-1.sharepoint.emea.microsoftonline.com/Spectracom/Engineering/products/TSync/Shared%20Documents/Declaration%20of%20Conformity.pdf

Size of the TSync-PCI104 board

(this is PCIe- needs to be changed for TSync-PCI104)



For your information, the TSync-PCIe board itself is about 2.75 inches high, at its widest point (including the connectors that plug into the PCIe slots).
The TSync-PCIe board ships with a half height bracket attached to the board (for attaching the board to the PC) with a full height bracket also included (the picture below shows a half height.

2.75 inches / 7cm
7 1/8 inches deep
6 5/8 inches deep
4.75 inches / 12cm (full height bracket)
3.25 inches / 8cm (half height bracket)
GPS

Weight of the TSync-PCI104

(this PCIe- needs to be changed for TSync-PCI104)

~~The weight of the TSync-PCI104 (with the GPS receiver attached to the board) is less than one pound. Per the scale, it looks like it's about 0.3 pounds.~~

TEMPERATURE SPECs:

TSync-PCI104 MTBF

Refer to MTBF/MTTR (for all products) in this document.

Ancillary kits

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Mounting Brackets

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Timing Connector

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Breakout cables for the "Timing" connector

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Link to board schematic: I:\Engineering\Schematic\1191-1001-0200

FIRMWARE

**Firmware version information and updates -Firmware/FPGA/PTP module (if installed)

CTC-cPCI Firmware updates

- Link to CTC-cPCI Firmware version change document (and instructions to perform the firmware upgrade):
  I:\Customer Service\PSB, PSP software updates\TSYNC boards\Tsync Firmware updates

CTC-cPCI Driver updates

Link to PTP module Firmware version change document (and instructions to perform the firmware upgrade):
Refer to: I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\TSync driver updates

Obtain Firmware/Software version information

Status LED's/Status LEDs upon reboot

Refer to TSync-PCIE section

MEMORY

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

**DAGR GPS receiver ("Defense Advanced GPS Receiver" interface)

- Refer to "Model 1204-02" in the Option Card information document: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\SecureSync\Option Cards

**IRIG input synchronization

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

**"EPP0 (external 1PPS Input Reference)

Connect the external 1PPS input to Pin 7 of the timing connector (1PPS input pin) or to the standard/premium breakout

**UT1 correction factor (leap seconds)

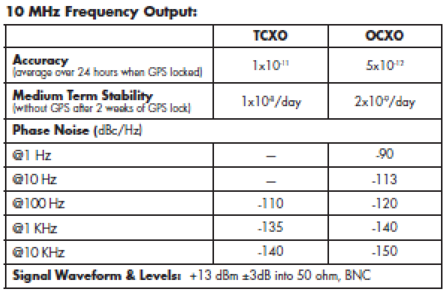
- Refer to the TSync-PCIe section (same as the TSync-PCIe)
-

Oscillators / Osc disciplining to PPS ref / TFOM / Holdover

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Outputs

**10MHz output



10 MHz Frequency Output:

	TCXO	OCXO
Accuracy (average over 24 hours when GPS locked)	1x10^{-11}	5x10^{-12}
Medium Term Stability (without GPS after 2 weeks of GPS lock)	1x10^{-8}/day	2x10^{-9}/day
Phase Noise (dBc/Hz)		
@1 Hz	–	-90
@10 Hz	–	-113
@100 Hz	-110	-120
@1 KHz	-135	-140
@10 KHz	-140	-150
Signal Waveform & Levels: +13 dBm ±3dB into 50 ohm, BNC		

AM output only (no 10 MHz DCLS output available)

Output level

- Output level is not adjustable
- +13 dBm +/-3dB into 50 ohm, BNC (10dBm=2.00vpp. 16dBm=3.99vpp)
- 5Vp-p nominal into high impedance, +/-3dB

Output impedance: 50 ohms

Output Amplitude 12 dBm (2.5Vp-p), nominal into 50 ohm, +/-3dB,
5Vp-p nominal into high impedance, +/-3dB,

By default, The 10MHz output is enabled. With no external input, the signal will be derived from the free-running internal oscillator. When the board is synchronized to itself, the oscillator is not disciplined. In order to be disciplined, the board needs an external 1PPS reference, such as GPS or external 1PPS input. This output configuration is limited to disabling the output when the board is not synced (this is known as Signature Control") or disabled altogether (no output, whether or not the board is synced).

The calls to reconfigure when the 10 MHz is not present are listed in the driver guide as "FP" calls /commands. If you search the TSYNC driver guide for this value, it will show you all of the available configuration calls that can be issued to reconfigure the 10MHz output as desired.

Note: Further below is additional information on Signature Control.

Phase alignment of 10 MHz outputs from multiple TSync-PCIe boards or SecureSyncs

Because KTS disciplining performs phase alignment of the 10 MHz output, unlike earlier NetClocks which only perform Frequency adjustments, if more than one TSync or SecureSync are synced to the same IRIG or GPS signal, the 10 MHZ outputs will be in phase with each other (not just corrected for frequency).

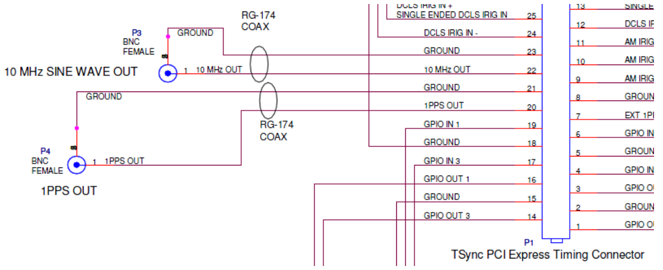
Harmonics and Spurs of the 10 MHz output

Refer to the TSync data sheet for specs on harmonics. ..\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards

For plots of the 10 MHz harmonics and spurs with an OCXO installed, refer to the email from Tom Richardson (8/2/12) in the following folder: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\10MHz output

10MHz output pin-outs

- ➢ Not available on Basic/Standard Breakout cable
- ➢ Using Premium Breakout cable (BNC Connector P3)



10MHZ Signal	Connector P3 - pin	Goes to Pin on Timing connector
10MHz	Center pin	22
Ground	Shield	23

**1PPS output

Note: The 1PPS/ “Heartbeat” output is always 1PPS, but the pulse width of the output can be edited. For true heartbeat output (ability to change the frequency) refer to GPO outputs.

1PPS output impedance: The 1PPS output is high impedance (The 1PPS input is also high input impedance)

1PPS output Specs from manual:
• 1PPS output at timing interface connector
• 1Hz Pulse, Rising Edge or Falling Edge Active (selectable) o 40ns - 900ms Active Pulse Width (selectable, 200ms default)
• +0.55V VOL, +2.2V VOH

Two methods to output 1PPS

- ➢ Dedicated 1PPS output is on pin 20 of the Timing connector.
- ➢ Configure one or more of the GPO pin for

API calls for 1PPS/Heartbeat output: The 1PPS output calls for TSync-PCIe are the “**TSync _ PP”** calls. (Refer to the TSync-PCIe driver guide 1219-5001-0050) in Arena at https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

Available configuration API calls for 1PPS output on pin 20

- ➢ Get/Set Signature Control

- Get Frequency (1Hz)
- Get/Set offset (account for cable delays to external device)
- Get/Set the active edge (defines which edge is aligned to the on-time point)
- Get/Set the pulse width of the 1PPS signal
- Get the number of instances (usually just one on the TSync board)

Per the TSync-PCIe manual:

The TSync-PCIe board generates a digital 1PPS output from the internal 1PPS of the system. Several parameters of the 1PPS can be controlled. The active edge can be set to either rising or falling edge, the pulse width can be adjusted, and an offset can be applied to adjust its relationship to the internal system, 1PPS from -500 msec to +500 msec in 5 nsec increments. The 1PPS output provides signature control similar to the IRIG outputs.

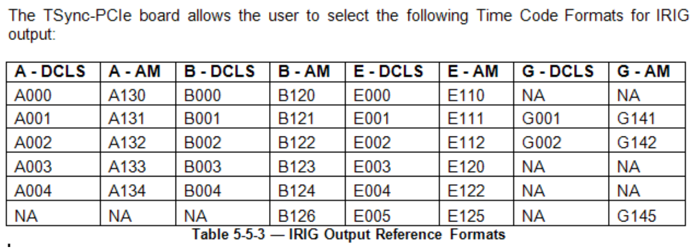
The TSync-PCIe timing board has legacy TPRO API calls available to configure the "Heartbeat" output (this is the 1PPS output on pin 20 of the timing connector). Unlike the TPRO/TSAT boards, TSync itself does not have a dedicated output called "Heartbeat". This output was replaced by just a "1PPS output" instead. So the legacy heartbeat calls just configure this 1PPS output pin. There are no "TSync Heartbeat" API calls available in the TSync-PCIe drivers (just the legacy "TPRO heartbeat" calls. The 1PPS output calls for TSync-PCIe are the TSync-PCIe_ **PP** calls. (Refer to the TSync-PCIe driver guide 1219-5001-0050 in Arena at https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002).

1PPS output holdover specs with OCXO oscillator

Email from Dave Sohn (10/3/11) regarding low phase noise drift (no GPS) after at least two weeks of lock:
Estimate for holdover at constant temperature after 2 weeks of GPS lock

Days	Drift
1	10 us
2	40 us
3	80 us
7	430 us

Also refer to: IRIG for TSync-PCIe



The TSync-PCIe board allows the user to select the following Time Code Formats for IRIG output:

A - DCLS	A - AM	B - DCLS	B - AM	E - DCLS	E - AM	G - DCLS	G - AM
A000	A130	B000	B120	E000	E110	NA	NA
A001	A131	B001	B121	E001	E111	G001	G141
A002	A132	B002	B122	E002	E112	G002	G142
A003	A133	B003	B123	E003	E120	NA	NA
A004	A134	B004	B124	E004	E122	NA	NA
NA	NA	NA	B126	E005	E125	NA	G145

Table 5-5-3 — IRIG Output Reference Formats

Both IRIG outputs are on the 25 pin "Timing" connection
IRIG AM output is on pin 9
IRIG DCLS is on pins 12 (-Data) and 13 (+ Data)

By default, the IRIG output is enabled. The IRIG output has the ability to be configured for different formats (A, B, E and G), local time as desired, the location of the year data, etc. These commands are listed in the driver guide as the "_IP" calls/commands. If you search the TSYNC driver guide for this value, it will show you all of the available configuration calls that can be issued to reconfigure the IRIG output as desired. (1219-5001-0050 in Arena at https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002)

API calls for IRIG output: IRIG output calls for TSync-PCIe are the "**IP**" calls.

IR command syntax:
Use the <Index> value of:
0 for DCLS output ports.
1 for AM output ports.

TSync-PCIe boards can output= A, B, E, G, NASA36, IEEE 1344 pulse width codes (TSync and SecureSync do support IEEE extensions. Refer to: IEEE-1344:)

Note: Does not generate Manchester modulated codes.

IRIG output can provide the following coded expressions combinations for BCDTOY, CF, SBS, and BCDYEAR fields:
• 0 – BCDTOY, CF, SBS
• 1 – BCDTOY, CF
• 2 – BCDTOY
• 3 – BCDTOY, SBS
• 4 - BCDTOY, BCDYEAR, CF, SBS
• 5 - BCDTOY, BCDYEAR, CF

Output impedance: 50 ohms

Note: In the TSync-PCIe drivers, the IRIG output is configured using the 'IP" commands

Available IRIG output configuration commands (these are the "IP" calls/example programs)

**Alarms and Log

Note: Refer to: **PTP Precision Timing Protocol (for all products)** for more info on PTP.

Log/Messages

GPIO (General Purpose Input Output) pins

Refer to Tsync-PCIe section of this document

(timestamping/Time Stamping/Timetag/Time Tag)/External Event Input

Refer to Timestamping in Tsync-PCIe Section.

**GPO pins (General Purpose Outputs- “GPO”)

VxWorks or other custom drivers not supplied by Spectracom

- Spectracom does not supply a VxWorks driver.
- The Linux driver for Tsync board contains Source code that can be used as a reference to see how functions are performed.
- Refer to section 2 of the Application programmers manual for register Memory mapping.
- Refer to Section 8 of the Application programmers manual for info on performing different functions.
- Unlike legacy TPRO/TSAT boards, the TSync register reads need to be read through the FIFO. The registers for the FIFO buffer are defined under '**Shared Memory Interface**" in the App Programmers guide.

Reading/writing to registers directly "Direct Memory Access" using a Spectracom driver (Peek and Poke API calls)

Refer to the Application Programmers Manual for more information on direct memory access (1191-5002-0050): I:\New Released\Manuals\1191-xxxx-xxxx. (**Note**: the first page of Section 2 provides the Memory Map of all of the registers).

Values from the TSync-PCIe boards can be directly read from or written directly to the board's registers, if desired.

"Peek" API call: Allows registers to be read directly
"Poke" API call: Allows registers to be written to directly (Caution: Make sure to write to the correct register. Otherwise bad things can happen)

The PC BIOS assigns a unique Base address for each PCIe bus card installed. Then, each Memory Register is assigned a unit Base address extension to identify each particular register.

The TSync Application Programmers manual provides a Memory Map for the memory locations (Base address extensions) which store specific contents. The customer can then create a custom driver to read the contents directly. Refer to the Application Programmers manual for more details I:\New Released\Manuals\1191-xxxx-xxxx (1191-5002-0050).

The source code supplied with the Linux driver can assist them in identifying the Base address that is assigned by Bios and shows how we access the registers.

Pages 2-1 and 2-2 of the Application Programmers manual (latest version is attached, just in case) provides the Memory Map (Base address extension) for all of the Memory Registers on the TSync-PCIe board.

I recommend you also refer to the source code that is provided with the Spectracom TSync-PCIe Linux driver. This source code will help you determine the unique Base address that is assigned by the PC BIOS to the installed TSync-PCIe board. The source code also provides good information that can be used to help create your custom driver for accessing these memory registers.

Tsync registers reads need to be read through the FIFO buffer

Email Keith sent based on input from Tim Tetreault Please be aware that the TSync family of timing boards are setup differently than the earlier, legacy TPRO/TSAT timing boards. Time can be read from a register but all API commands that don't start with a "**HW_"** pass through a FIFO buffer to send and receive commands with the TSync boards.

All commands will use the "TSYNC_SET", "TSYNC_GET" & "TSYNC_WAIT", including even the earlier, legacy "TPRO" API commands.

The registers for the FIFO buffer are defined under '**Shared Memory Interface**" in the App Programmers guide.

tsync.dll and tpro.dll files

- When the Tsync driver is installed, it generates two .dll files, tpro.dll and tsync.dll
- The tsync.dll file support all Tsync calls.
- The tpro.dll file allows customers upgrading from TSAT-PCI to be able to use the legacy commands from the Tpro/tsat-PCIU driver, without needing to rewrite their application software.
- They still have to install the Tsync driver and recompile their software to the new driver.

Q. We are upgrading some kit for a customer who use to use a TPro PMC board. The current software uses the supplied TPRO.dll for the API.

Now they have purchased a TSync-PCIe and wish to upgrade the software, and as a consequence we downloaded the drivers and API supplied on your web site for the TSync-PCIe.

However we have noticed that the TSync-PCIe driver have both a TSYNC.dll and the TPRO.dll included. Question is, can we use the TPRO.dll to control and communicate with the TSync-PCIe card. Note we are not interested in any new features that the TSync-PCIe card may provide at this time, therefore to minimise the porting effort we would happily use the TPRO.dll if this will work?

- (Keith 19 Mar 2014) Thanks for your email and question. I have some information for you.

The tpro.dll file generated when the Tsync-PCIe driver is installed is actually intended for the legacy TPRO/TSAT-PCI (not PMC) timing boards. However, almost all of the calls from the TPRO/TSAT–PCI and PMC boards (and therefore this tpro.dll file) are the same.

We are in the process of releasing a new TSync-cPCI/PMC board and its driver will also contain a tpro.dll file which is for all of the legacy TPRO/TSAT-PMC/cPCI calls. We are also in the process of releasing a new version of the TSync-PCIe driver to support both Tsync-PCIe and TSync-PCI104 .

Two options for you- We should be releasing the new TSync-PCI104 boards and this new combined driver very shortly. If you like, I can let you know when it's available. Or, you can also use the cPCI-PMC driver currently available on our website. To obtain this driver, please visit us at:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=22

Static and dynamic TSync.lib file (library file)

.lib files (Linker files)

What is a .lib file?
From: http://file.org/extension/lib

Files that contain the .lib file extension usually hold a static data library of information. The information in this file is associated with a specific computer application. The LIB files usually store the functions that are referenced by the application, which is then associated with the library.

LIB files may also contain objects like images, media and text clippings. LIB files are an alternative to DLL files, which contain dynamic link libraries. The LIB files contain static libraries that can be used as import libraries for DLL files, with the imported library files being marked with the .lib file suffix.

From Wikipedia: http://stackoverflow.com/questions/3250467/what-is-inside-lib-file-of-static-library-statically-linked-dynamic-library-an

Static library
In a static library, the .lib file contains all the actual object code and data for the functions provided by the library. In the shared version (what you referred to as statically linked dynamic library), there is just enough code to establish the dynamic linkage at runtime. The linker then identifies the bits it needs and puts them in the final executable.
If the code of the library is accessed during the build of the invoking program, then the library is called a static library. An alternative is to build the executable of the invoking program and distribute that, independently from the library implementation. The library behavior is connected after the executable has been invoked to be executed, either as part of the process of starting the execution, or in the middle of execution. In this case the library is called a dynamic library. A dynamic library can be loaded and linked as part of preparing a program for execution, by the linker. Alternatively, in the middle of execution, an application may explicitly request that a module be loaded.

Dynamic library
For a dynamic library, the .lib file contains a list of the exported functions and data elements from the library, and information about which DLL they came from. When the linker builds the final executable then if any of the functions or data elements from the library are used then the linker adds a reference to the DLL (causing it to be automatically loaded by Windows), and adds entries to the executable's import table so that a call to the function is redirected into that DLL.
You don't need a .lib file to use a dynamic library, but without one you cannot treat functions from the DLL as normal functions in your code. Instead you must manually call LoadLibrary to load the DLL (and FreeLibrary when you're done), and GetProcAddress to obtain the address of the function or data item in the DLL. You must then cast the returned address to an appropriate pointer-to-function in order to use it.

Refer to Salesforce case 11052
Q. I received the TSync card this week. One thing I notice is that there is no static library version of TSync.lib; all of our apps are statically linked, and I'm hoping you guys support a build of your API in this mode as well. All I see in the distribution is a dynamically linked version.

- As of at least July 2013, the TSync-PCIe Linux driver does support both static and dyamic lib files. However, as of at least version 2.4.1 Windows driver does not support both static and dynamic .lib files.

Compiling Customer's application software

(Note: For Visual Studio info, refer to the Windows driver section further below)

Borland

- Our TSync.lib file was written for Visual C+ +. Customer desires to use Borland C + + instead:
- Q. The customer has begun to write his software application. However, he is using Borland C++ and Tsync.lib seems to be wrote for Visual C++. We have found a solution however we are not sure his approach. We used Tsync.dll and applied "implib" utility in order to have Tsync.lib compatible with Borland. It seems to work but we do not have the board to verify it. Is it the good approach? If it is not, can you give us the procedure? Or do you have a Tsync.lib compatible for Borland C++?
- A. Email from Tim T - The approach they are using is correct. Here is a link with more information on how to use Visual C++ DLLs with Boland C++ using the IMPLIB utility.

http://bcbjournal.org/articles/vol4/0012/Using_Visual_C_DLLs_with_CBuilder.htm?PHPSESSID=046f3a5a4298523cb724d9dcf777ec0f

C# (called "C Sharp", "Managed Code C #" or "Managed Code C Sharp")

2) Our TSync.lib file was written for Visual C+ +. Customer desires to use C# (C #) instead:

Q (email from Jack Loui 3/17/11) I am trying to convert TSync C++ code to C# by using [DllImport("Tsync.dll"), CallingConvention = CallingConvertion.Cdecl)]. So far I have trouble to get this to work. Do you have any sample code wh A newer version of Microsoft Visual C++ 2010 ch is in C# to work with the TSync card?

A Keith's reply: Unfortunately, we don't have any sample C# code that we can provide you with. We know that we have had one or two other customers that were able to use our dll files for the conversion, but we don't have any information on how they were able to make the conversion, nor have we done this ourselves.

**TROUBLESHOOTING TSync-cPCI boards

****TSync-cPCI board doesn't appear to be operational after install

Questions to Ask/Info to help troubleshoot (steps to troubleshoot further below)

- What is the Serial Number of the board (via the silver P/N sticker affixed to the board or with the LS_GetSerialNo API call/example program).
- Is this the only TSync-cPCI board that you have? If not, have you tried another timing board in its place, yet?
- Have you tried reseating the board in the slot?
- Have you tried this TSync-cPCI board in another computer, yet? If not, I recommend you try it in a different computer, to determine if there is an issue with the PC's PCIe slot.
- Is the TSync-cPCI board installed in a standard cPCI backplane?
- DO you have a manufacturer make and model of the cPCI backplane you are using? We wanted to check once more and see if we are missing something.

  DO you have a manufacturer make and model of the cPCI backplane you are using? We wanted to check once more and see if we are missing something.

- What are the LEDs on the edge of the TSync-cPCI board doing (Do they flash once at power-up? Are they flashing a pattern? Are they constantly on)?

  There are LEDs on the front panel of the board. What are these LEDs indicating when you power up the board in the chassis?

- What about the FPGA LED… is that lit? Is the bootloader jumper in the correct position?
- TSync Board firmware version and driver versions.
- Is the board installed in a Linux system?

Linux systems only

A. What is the specific linux distribution (such as Redhat, CentOS, Scientific linux, etc) and kernel version?

Notes:

- ASPM may need to be disabled
- Updating the linux distribution to v6.4 or higher may help address issus with ASPM issues.
- Class code may need to be updated (especially if the board shipped before ~ April 2013).
- System BIOS may need to be updated

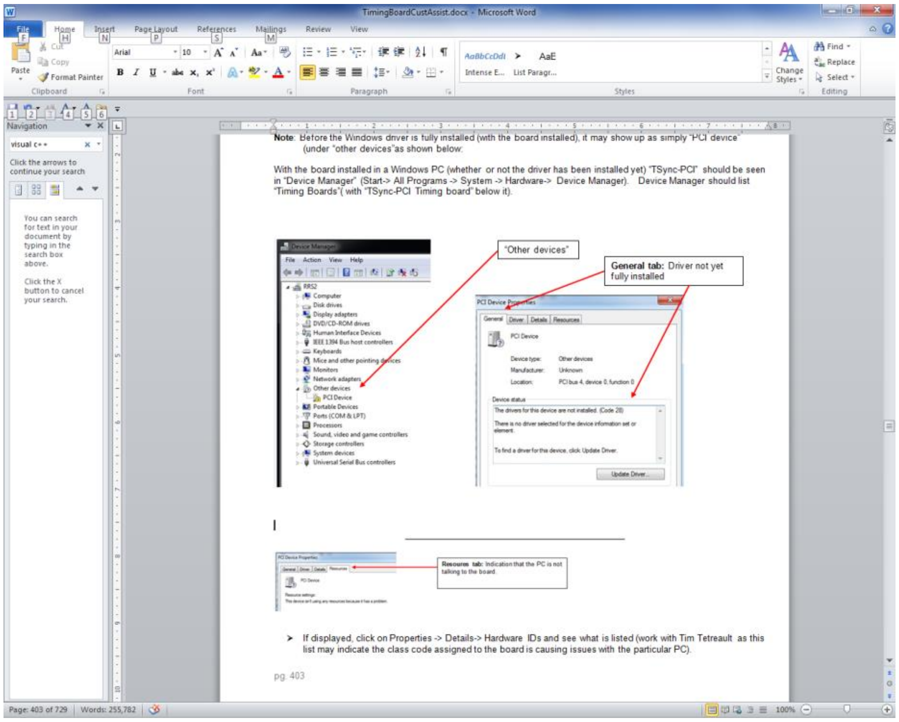
B. Has the earlier version 2.3.0/2.4.1 drivers ever been installed on this machine. If so, the Rules files need to be updated.

Windows PCs only

With the board installed in a Windows PC (whether or not the driver has been installed yet) “TSync-PCI” is it seen in “Device Manager” (Start-> All Programs -> System -> Hardware-> Device Manager).   Device Manager should list “Timing Boards” (with “TSync-PCI Timing board” below it).

Note: Before the Windows driver is fully installed (with the board installed), it may show up as simply “PCI device” (under “other devices”as shown below:

With the board installed in a Windows PC (whether or not the driver has been installed yet) “TSync-PCI” should be seen in “Device Manager” (Start-> All Programs -> System -> Hardware-> Device Manager).   Device Manager should list “Timing Boards”(with “TSync-PCI Timing board” below it).



Note: Before the Windows driver is fully installed (with the board installed), it may show up as simply “PCI device” (under “other devices”as shown below:

With the board installed in a Windows PC (whether or not the driver has been installed yet) “TSync-PCI” should be seen in “Device Manager” (Start-> All Programs -> System -> Hardware-> Device Manager).   Device Manager should list “Timing Boards”(with “TSync-PCI Timing board” below it).

“Other devices”

General tab: Driver not yet fully installed

Device type: Other devices

Manufacturer: Unknown

Location: PCI bus 4, device 0, function 0

Device status

The drivers for this device are not installed. (Code 28)

There is no driver selected for the device information set or element.

To find a driver for this device, click Update Driver.

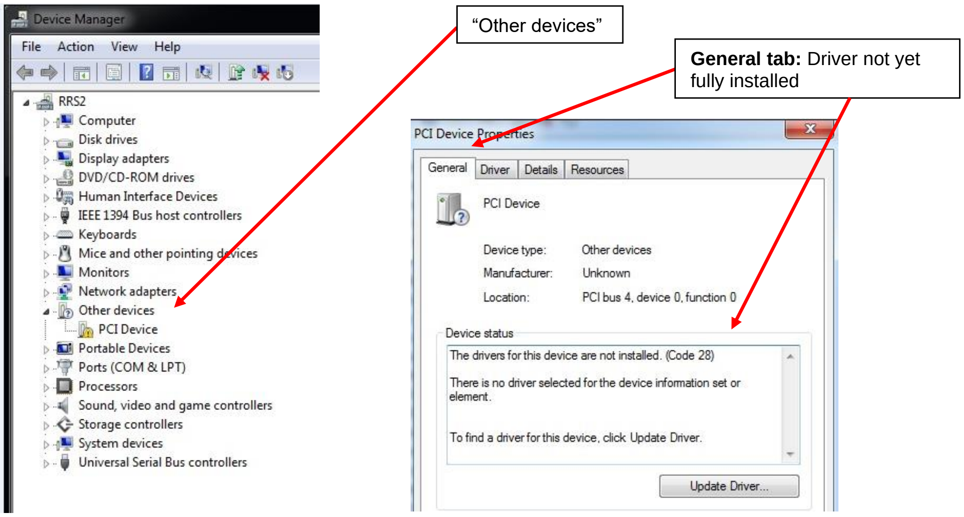
Resource tab: Indication that the PC is not talking to the board.

Resource settings:

This device isn't using any resources because it has a problem.

- If displayed, click on Properties -> Details-> Hardware IDs and see what is listed (work with Tim Tetreault as this list may indicate the class code assigned to the board is causing issues with the particular PC).

pg. 403



"Other devices"

General tab: Driver not yet fully installed

PCI Device

Device type: Other devices

Manufacturer: Unknown

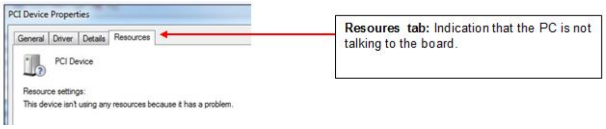
Location: PCI bus 4, device 0, function 0

Device status

The drivers for this device are not installed. (Code 28)

There is no driver selected for the device information set or element.

To find a driver for this device, click Update Driver.



PCI Device

Resource settings:

This device isn't using any resources because it has a problem.

Resoures tab: Indication that the PC is not talking to the board.

- If displayed, click on Properties -> Details-> Hardware IDs and see what is listed (work with Tim Tetreault as this list may indicate the class code assigned to the board is causing issues with the particular PC).
- Try the board in another PC (preferably on a different Model motherboard) to see if theother PC can recongnize it's installed.
- Refer to Class Code update fix and (ASPM) Power Saver/Power Monitor feature
- Verify the PC BIOS settings for the PCie bus. Make sure the bus is not configured for "Video Only", ASPM (autopower save mode, USB) enabled, etc.

Email Keith sent to a customer. Have you tried going into "Device Manager" and running "Scan for hardware changes" from the "Action" menu?

Another idea is to verify that the system can see the board. In "Device Manager", select "Devices by connection" in the "View menu". Look under the "Microsoft ACPI-Compliant System->PCI bus" group for any unknown device. If you look at properties for that unknown device you can check to see if one of them is the board. The TSync board is Vendor ID "1AD7" and Device ID "8000".

Do you have any other cards installed? Are they visible in Device Manager? Do you have any other PCIe boards to verify that the system is properly seeing the PCIe card slots?

Specific motherboard issues

ASUS motherboard

- Salesforce Case 6738- The ASUS motherboard came with a utility called 'ASRock X-Fast USB' that was removed and the computer was re-booted. Immediately, the installation program operated normally and the Tsync driver was installed successfully.

Checks to make:

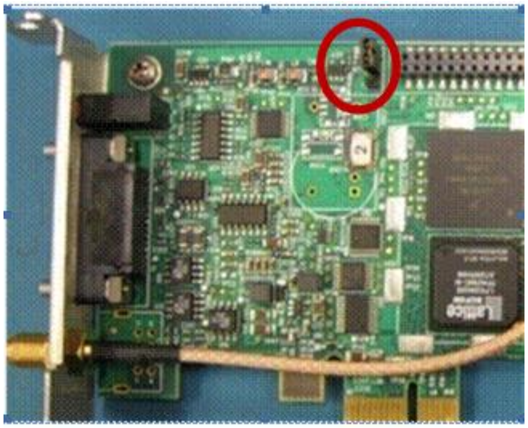
- Check the Status LEDs on the edge and the green “FPGA load” LEDs on the back side of the TSync-PCIe board:
- Verify all three Status LEDs (red, yellow and green) on the edge of the TSync board turn on and then go out (and remain not lit until an input reference becomes valid).

Right after the PC with the TSync-PCIe board powers-up, the three LEDs on the edge of the TSync-PCIe (the red, yellow and green LEDs), should momentarily illuminate and then turn off (until an input reference is connected). Do all of these LEDs momentarily turn on when the PC is first booted-up?

If the LEDs are not lit at power-up /all three constantly lit/exhibit the correct pattern:

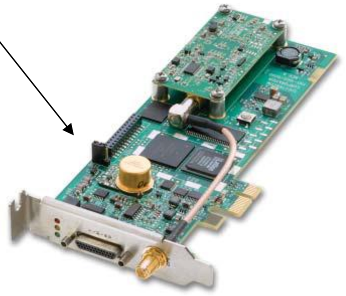
- First, try to re-seat the TSync-PCIe board in the system to make sure that all the contacts are solid.
- If the green and red LEDs are blinking on your PCIe card, it is likely that the board is in the bootloader waiting for image downloads from the host. This could signify that either the board needs to be reprogrammed or that there was a failure in the flash forcing it to fail to load the run-time or backup default images, landing back into the bootloader.

Verify the green “FPGA” loaded LED is lit (verify Jumper J25 is in the correct position)
Check jumper setting on J25 and make sure that it is configured on the pins to the outside edge of the board, as seen in the red circle in the photo below:



If this jumper is in the wrong position, on a reboot the timing board will never look for the new images and will always load the bootloader (Green and Red LED s will be blinking and the Yellow LED will be not lit). Please place this two position jumper as shown in the picture above (with the jumper across the two outer pins (closest to the edge of the TSync-PCIe board.

There is also a green "load" LED on the back-side of the TSync-PCIe boards (in the same area of the board as the jumper described above, but on the other side of the timing board. This LED should light shortly after the TSync-PCIe board powers-up and then should remain lit, from that point forward. Please let me know if this LED turns on and then remains lit green while the PC is on.



The green FPGA load LED is located on the back side of the PCB, in the same area as the programming jumper.

The LED should turn on shortly after the board boots-up and then should remain lit.

If the green LED still doesn't turn on and remain lit with the jumper in the correct position

- Verify +3.3 vdc is present from the PCIe bus. If this Dc voltage is low, the board may be stuck in constant reset state. Refer to: Power (+3.3vdc +12vdc and -12vdc from PCI bus)/logic.
- Try the board in another slot, or another system altogether and see if the gren LED remains lit.

Labview driver/Labview wrapper

- As of at least Jan 2014, the TSync-PCIe boards do not have an available Labview driver / Labview wrapper. There are no plans at this time to develop one.
- With the shared library (xxx.so) file provided with our driver, a customer can create their own wrapper routines using this provided file. For additional info on calling .so files in Labview, refer to http://zone.ni.com/reference/en-XX/help/371361J-01/lvhowto/import_Shared_library/

Note: The website above discusses how to import functions from a shared Library file (.so file , as provided in the TSync-PCIe Windows driver) into Labview.

- The libtsync.so shared library file is generated during the driver build. Its located in the TSync/Linux/Lib directory.

For more information on wrappers, refer to websites such as:
http://digital.ni.com/public.nsf/allkb/06ECDC689DDA0F3D862574440074CD95

Problem:
What is a wrapper DLL and when do I need one?

Solution:
A wrapper is a piece of software that provides a compatibility layer to another piece of software. One is often necessary when developing LabVIEW applications because third-party DLLs were usually designed to be accessed from C (or similar low-level languages) and not LabVIEW. Such a DLL may, for example, return pointers or complex data structures which LabVIEW cannot easily handle.

Writing a wrapper DLL can be compared to writing a completely separate program in C that accesses the original DLL in the way the original author intended. In turn, this wrapper program has been specifically designed to be accessed from LabVIEW. In this sense, the new C program "wraps" around the original C program (DLL) and provides a layer of compatibility. The benefit of a wrapper is that the source code for the original DLL is not necessary, as it does not need to be modified in any way.

Q. OK, this is a concern. Are you saying that there is no shared library (xxx.so) that I can use to access the functionality of the time card? That is all I would need to access the card from LabVIEW (I can create my own wrapper routines if I have a shared library).

- My response to this email was No need to fret! The libtsync.so shared library file is generated during the driver build. Its located in the TSync/Linux/Lib directory.

Reply Keith sent to a customer (5 Nov 2013)
To answer your question, there are no plans at this time to develop and release a Labview driver/wrapper for the TSync-PCIe timing boards.

However, please note that we provide a **libtsync.so** shared library file with the TSync drivers which a customer can use to create their own wrapper routines. This file is located on the **TSync/Library/Lib** directory. Please note that we do not provide any technical support on how to do this, but your customer is free to try this if they wish. They can download the TSync drivers from our website at no cost if they would like to look into this. The TSync-PCIe drivers can be downloaded at: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=37.

NI LabWindows/CVI 9.0,
Q.Our customer has purchased our Tsync-PCIe timing board, but when they write its testing program based on NI LabWindows/CVI 9.0, (Windwos XP), they found the program can not be building. I was wondering could you please help advise for this issue.

A. (reply from Keith 27 Jan 2014)
For your information, Labview programs such as LabWindows require a wrapper to be used in conjunction with the Spectracom TSync-PCIe Windows driver. Spectracom does not provide a Labview wrapper with the TSync-PCIe board drivers, so customers that wish to use the TSync-PCI board with programs such as LabWindows will need to create their own custom wrapper, consisting in part of the API calls from the TSync-PCIe board which they want to use with the LabWindows program.

However, please note that we provide a **libtsync.so** shared library file with the TSync-PCIe drivers, which can be used to create custom wrapper routines. For more information on wrappers, refer to websites such as: http://digital.ni.com/public.nsf/allkb/06ECDC689DDA0F3D862574440074CD95.

Problem:
What is a wrapper DLL and when do I need one?

Solution:
A wrapper is a piece of software that provides a compatibility layer to another piece of software. One is often necessary when developing LabVIEW applications because third-party DLLs were usually designed to be accessed from C (or similar low-level languages) and not LabVIEW. Such a DLL may, for example, return pointers or complex data structures which LabVIEW cannot easily handle.

Writing a wrapper DLL can be compared to writing a completely separate program in C that accesses the original DLL in the way the original author intended. In turn, this wrapper program has been specifically designed to be accessed from LabVIEW. In this sense, the new C program "wraps" around the original C program (DLL) and provides a layer of compatibility. The benefit of a wrapper is that the source code for the original DLL is not necessary, as it does not need to be modified in any way.

Windows Driver/ Visual Studio

Shortcut to TSync-PCIe driver release notes: I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\TSync driver updates

Paths to the TSync Windows driver and example programs
Path to the drivers/example programs:

Driver: C: \Program Files\Spectracom\TSYNC PCI\drivers\

Example programs
For TSync example programs: C: \Program Files\Spectracom\TSYNC PCI\Examples\TSYNC API\
For Legacy TPRO example programs: C: \Program Files\Spectracom\TSYNC PCI\Examples\TPro API

TSync.lib file

- (July 2013) refer to Salesforce Case 11052 As of at least Verion 2.4.1 of the Windows driver, this driver only support dynamically linked .lib file. It does not currently support statically linked .lib file (unlike the linux driver, which supports both).
- Per Tim Teatrault, we will need to compile the Windows driver differently when we build it, in order to support this capability.

TROUBLESHOOTING WINDOWS INSTALL
General-type questions to ask.
Here are some questions for you:

- What is the Serial Number of the TSync-PCIe board (its indicated on the silver Spectracom Part Number sticker affixed to the TSync-PCIe board)?
- What is the version of the TSync-PCIe Windows driver you have?
- Has this TSync-PCIe board been installed and operating normally in this same machine for a while, but has since stopped working with these errors just now being seen?
- Or, Is this a TSync-PCIe board that was installed previously in another machine and then recently moved over to this IBM Windows server?
- Or, is this the first time being installed in a machine since it was purchased?
- At what stage of the installation process (assuming it's a new install in this machine), did the server errors start occurring? Was it when the board was first installed, while installing the driver, while working with it in conjunction with custom application software, etc?
- The TSync-PCIe board can be installed without the driver being installed, and should be recognized in Device Manager. Is this board being detected as being installed in Device Manager.
- If you haven't already, are you able to temporarily install this TSync-PCIe board in another Linux or Windows machine to see if it can be detected and able to communicate via the bus?
- Do you have any other PCIe products installed in the same system?

**Windows Driver compatibility/install issues/conflicts with other PCIe boards

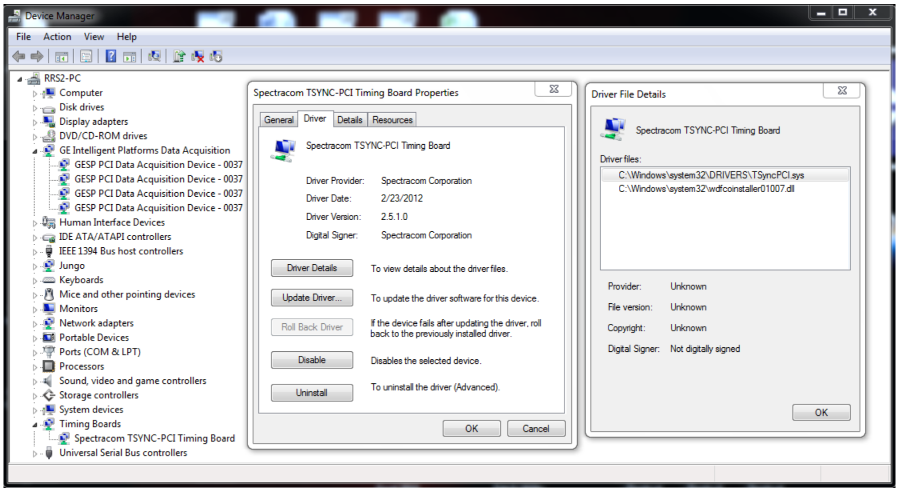
Apparent possible conflicts with other installed PCIe boards in the same system.

Driver issues could cause conflicts. If boards are resetting, could also be an issue with the +3.3vdc from the bus dropping too low.

To troubleshoot possible conflicts between TSync and other installed boards, go into Hardware tab / Device Manager. For all PCIe installed products (TSync-PCIe and others).

- Right click on each product and select “**Properties**”.
- Select the Driver tab.
- Select “**Driver Details**”.

This should list which driver(s) each installed board is linked to. We want to make sure the TSync-PCIe board is not linked to any other drivers and the other products are not linked to the TSync-PCIe driver. Please send a screenshot of this list to us.


Device Manager
File
Action
View
Help
RRS2-PC
Computer
Disk drives
Display adapters
DVD/CD-ROM drives
GE Intelligent Platforms Data Acquisition
GESP PCI Data Acquisition Device - 0037
GESP PCI Data Acquisition Device - 0037
GESP PCI Data Acquisition Device - 0037
GESP PCI Data Acquisition Device - 0037
Human Interface Devices
IDE ATA/ATAPI controllers
IEEE 1394 Bus host controllers
Jungo
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Portable Devices
Ports (COM & LPT)
Processors
Sound, video and game controllers
Storage controllers
System devices
Timing Boards
Spectracom TSYNC-PCI Timing Board
Universal Serial Bus controllers
General
Driver
Details
Resources
Driver Details
Update Driver...
Roll Back Driver
Disable
Uninstall
OK
Cancel
OK

Spectracom TSYNC-PCI Timing Board Properties

Spectracom TSYNC-PCI Timing Board

Driver Provider: Spectracom Corporation
Driver Date: 2/23/2012
Driver Version: 2.5.1.0
Digital Signer: Spectracom Corporation

To view details about the driver files.
To update the driver software for this device.
If the device fails after updating the driver, roll back to the previously installed driver.
Disables the selected device.
To uninstall the driver (Advanced).

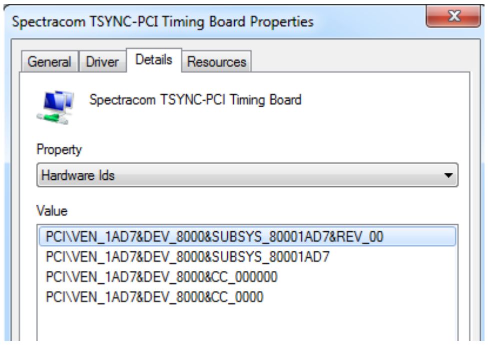
Driver File Details

Spectracom TSYNC-PCI Timing Board

Driver files:
C:\Windows\system32\DRIVERS\TSyncPCI.sys
C:\Windows\system32\wdfcoinstaller01007.dll

Provider: Unknown
File version: Unknown
Copyright: Unknown
Digital Signer: Not digitally signed

- Now click on the **Details** tab.
- In the drop-down in the **Details** tab, choose “**Hardware Ids**”).
- This tab shows the sub-system ID information. Send us a screenshot of this list for all of the products.



Spectracom TSYNC-PCI Timing Board Properties

Spectracom TSYNC-PCI Timing Board

Property: Hardware Ids

Value:

```
PCI\VEN_1AD7&DEV_8000&SUBSYS_80001AD7&REV_00
PCI\VEN_1AD7&DEV_8000&SUBSYS_80001AD7
PCI\VEN_1AD7&DEV_8000&CC_000000
PCI\VEN_1AD7&DEV_8000&CC_0000
```

If the Drivers for the TSync-PCIe and other devices don't appear to be conflicting (as determined by the info above) and the TSync-PCIe board is unexpectedly rebooting, using a scope (not a mulitmeter) check the +3.vdc going into the TSync-PCIe board via its Test point to see if this voltage is dropping below the min, even momentarily.

Note: This may be too quick or too small of a drop to catch with a multimeter. Really need to look at this test point with a scope so that it can be triggered if it drops below about 3.16vdc (resulting in a reset).

TSync-PCIe Driver support for earlier versions of Windows (Win 95, 98, CE, etc)

Note: Info below based on talk with Tim Tetreault:

The driver development tools we use start with Windows 2000 and go up from there. They do not list support for any earlier versions of Windows, including CE. Based on this information alone, the drivers are not likely compatible with CE. But, the only way to know for sure would be to install both the freely available driver in addition to the timing board (a demo board, if you have one available for them to try first, before buying??).

The driver contains individual folders for the particular version of Windows. When they install the driver, it may ask them which OS it is, since it may not be able to detect the version. They could select either XP or 2000 as alternate selections.

If they want to first try just installing the driver, they could try this, as the driver is available for free download (however, ultimately they would need to install both the driver and a timing board together to know it's going to work fine in CE).

Redistributables: TSync Driver version 2.41 can't complete the driver install because a newer version of Visual Studio 2010 is already installed on the PC "A newer version of Microsoft Visual C++ 2010 Redistributable has been detected on the machine."

Note: this issue is fixed with newer versions of the TSync driver

- Refer to Fogbugz case 1842 http://fogbugz/default.asp?pre=preMultiSearch&pg=pgList&pgBack=pgSearch&search=2&searchFor=1842&x=0&y=0
- Error message displayed when trying to install the driver: "A newer version of Microsoft Visual C++ 2010 Redistributable has been detected on the machine."

- Does not allow the driver install to proceed past this point.

Dec 2012- Refer to Salesforce case 6784/Fogbugz case 1842 for more info)

As reported by NovaSol, if Visual Studio 2010 is installed and has had updates applied to it, the Windows driver install process will halt and there is no wasy to force it to proceed (apparently it doesn't like the fact that a newer version is already installed).

We will need to release a newer version of the TSync-PCIe Windows driver that allows the upgrade proces to proceed, if a newer version is already installed on the PC.

Temporary work-around

Email from Dave Lorah to Tim T (67 Jan 2014) We had a customer last fall with a dependent vc80.crt error and he was able to resolve the issue by totally uninstalling the PCI Driver and then reinserted the CD. When the CD Autostarts it wants to load C++ for you. They selected NO - and did not install anything. They manually selected the XP 86 and loaded just our Driver software. Then after that loaded C++2005 from the web. It then worked OK.

Q. Is there a better workaround for this?

A Tim responded back with Dave,I don't have a better work around. We will probably be looking into a Windows driver update that will include this issue but I don't have a time schedule.

**Example programs included with the TSync Windows driver

The path to the drivers is: C:\Program Files\Spectracom\TSync PCI\examples\Tsync API

If the Windows Example programs can't run, try using the Windows Control Utility to communicate with the TSync-PCIe board.

Windows 32 bit OS

The example programs were initially compiled for **32** bit OS. They should work fine on any **32** bit Windows PC.

Windows 64 bit OS

Example programs run fine on 32 bit but won't work on 64 bit.

- To use the examples on a 64 bit system, they need to be rebuilt for 64 bit OS.

**Windows Control Utility GUI interface

- **To open the Control Utility:** Start / All Programs / Spectracom Corp / TSync PCI / TSync Control utility
- Provides a basic interface with the TSync-PCIe board via a GUI (such as reading the current Sync status, for example)
- The TSync-PCIe board does not need to be installed to use this GUI. Install the Windows driver and this GUI can be run in a "demo" mode to demonstrate the capabilities of the utility

**Clock Daemon program / Clock Daemon service (for all timing boards)

Refer to:

TSync-PCIe boards: TSync-PCIe driver guide (1219-5001-0050) for more information on the Clock daemon: in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

TPRO/TSAT-PCI boards: Section 2.2 of the PCI Application Programmers manual (*"2.2 Clock Daemon Utilities")* discusses the Clock Daemon program (refer to1186-5002-0050: I:\New Released\Manuals\1186-xxxx-xxxx)

- Consists of the "Clock Daemon" and the "Clock Daemon Service". Only one of these two can be used at a time.

TSync-PCI104 / (Tsync-PCI-104)

- Link to combined TSync family datasheet: I:\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards
- Link to user manual (1220-5000-0050) in Arena: https://files.bom.com/download/pfHCsYUxObQUWVjp0fkInpDJtpyHJ1Yn/vpsgvivbklzicybuqstkrnnzowdhpnre/1220-5000-0050%20TSync-PCI-104%20Rev%20A.pdf

Input References

GPS/GNSS

- All TSync-PC104 boards are GPS and Glonass capable, with Glonass enabled by factory default

Acutime GPS antennas

- Refer to "**Acutime antennas**" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

Associated Model 1169 GPS Fiber Optic Isolator for Accutime antennas

- Refer to "**Model 1169 GPS Fiber Optic Isolator (TX and RX) for GPS**" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

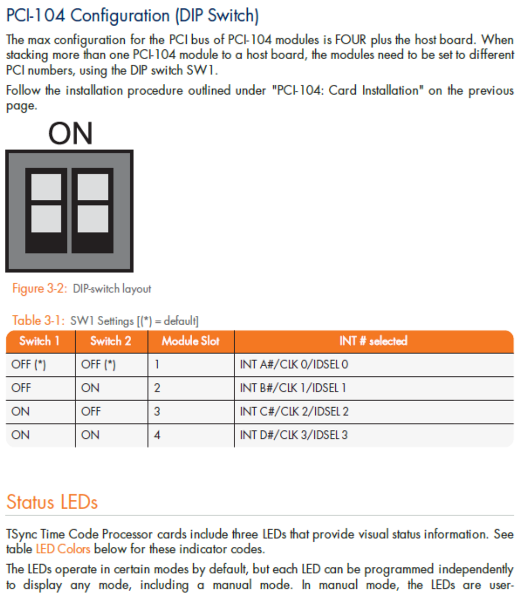
GPS Receiver standoffs:

Note: GPS receiver stand-offs are installed on ALL boards at the board-house. If a customer doesn't want them to be installed, they can be removed by drilling them out.

Q. We're interested in possibly using the holes for the standoffs to add additional constraint to the card since we're putting this card in a mobile application. Can you give me more information on what the standoffs are and how they attach to the board? If it's easier to talk to an engineer, let me know and we can arrange a call.

A. The GPS receiver standoffs are ¼ inch round, 5/16 inch tall, 440 threaded and are swaged onto the PC board. For your reference, the standoffs we use are from RAF electrical with their P/N of 3048-B-440-AL-0. (Refer to http://www.alliedelec.com/Search/SearchResults.aspx?N=0&Ntk=Secondary&Ntt=2191608 for more information on this component).

DIP Switch for TSync-PCI-104



PCI-104 Configuration (DIP Switch)

The max configuration for the PCI bus of PCI-104 modules is FOUR plus the host board. When stacking more than one PCI-104 module to a host board, the modules need to be set to different PCI numbers, using the DIP switch SW1.

Follow the installation procedure outlined under "PCI-104: Card Installation" on the previous page.

Figure 3-2: DIP-switch layout

Table 3-1: SW1 Settings [(*) = default]

Switch 1	Switch 2	Module Slot	INT # selected
OFF (*)	OFF (*)	1	INT A#/CLK 0/IDSEL 0
OFF	ON	2	INT B#/CLK 1/IDSEL 1
ON	OFF	3	INT C#/CLK 2/IDSEL 2
ON	ON	4	INT D#/CLK 3/IDSEL 3

Status LEDs

TSync Time Code Processor cards include three LEDs that provide visual status information. See table LED Colors below for these indicator codes.

The LEDs operate in certain modes by default, but each LED can be programmed independently to display any mode, including a manual mode. In manual mode, the LEDs are user-

Firmware/Drivers

Shortcut to firmware and driver version upgrades: I:\Customer Service\PSB, PSP software updates\TSYNC boards

Upgrading from existing TPRO/TSAT-PMC

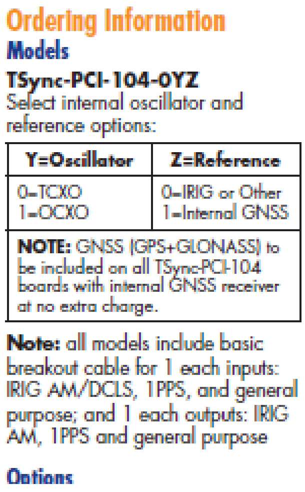
The Model TPRO/TSAT-PMC board has a dedicated frequency output pin. This is being replaced by the GPO 0 pin on the newer TSync-PCI104 board.

Upgrading from existing TPRO/TSAT-PC104

Email from Dave Sohn (9 Dec 2013): One big thing to know is that this is not a drop-in replacement. The new card will be PCI-104, not PC-104. There will be work that any customer switching to this from the old TPRO/TSAT-PC104 will need to do. The feature set is not yet finalized, but the starting point is the same features as our TSync-PMC and TSync-cPCI. That means it would have IRIG support and 1PPS input.

Note: The original TPRO/TSAT-PC104 board (ISA bus) is becoming a “PCI104” (PCI bus) bus card when its replaced as a TSync board (instead of remaining as a “PC104” board). PC104 and PCI104 are not cross-compatible with each other.

Configuration/Part Numbering



Ordering Information

Models

TSync-PCI-104-0YZ
Select internal oscillator and reference options:

Y=Oscillator	Z=Reference
0=TCXO 1=OCXO	0=IRIG or Other 1=Internal GNSS

NOTE: GNSS (GPS+GLONASS) to be included on all TSync-PCI-104 boards with internal GNSS receiver at no extra charge.

Note: all models include basic breakout cable for 1 each inputs: IRIG AM/DCLS, 1PPS, and general purpose; and 1 each outputs: IRIG AM, 1PPS and general purpose

Options

Drivers

Q. Will a driver written for QNX, for the old TPRO-PC104, work with the new board you are designing, minus any new board capabilities**.**

A Reply from Tim Tetreault (14 Nov 2013) The new board we are planning on releasing will be a PCI-104 board. So your current driver written to work with a PC-bus board will not work with our new board that is a PCI bus board. Also the board registers and interface are very different from the PC104 board. We will be supplying a Linux, Windows and Solaris driver with the new board. The new drivers include the old API calls that the old PC104 used.

Note: INFO BELOW COPY/PASTED FROM TSYNC-PCie SECTION (in process of updating for PCI-104 specific)

Login for our Tech Support PC:
User name: root   Password: abc123

Terms associated with TSync-PCI-104 boards

µC (Micro Computer) this refers to the computer in which the TSync-PCI104 board is installed.

HOST this refers to the computer in which the TSync-PCI104 board is installed.

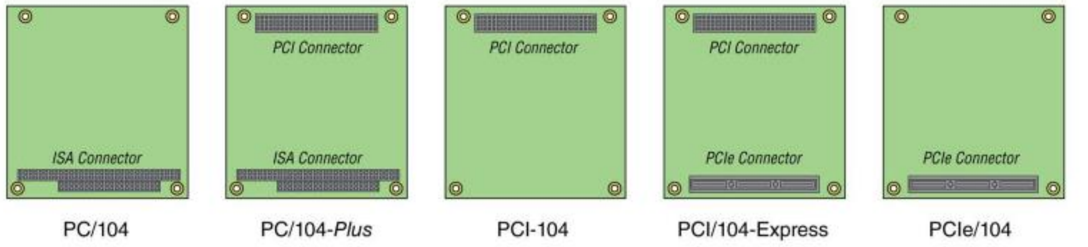
KTS refers to the "Kramden Timing System"
This is the Spectracom internal code name for timing chip Spectracom uses in SecureSync as well as the TSYNC board. It is an FPGA that actually does all the time and frequency synchronization.

HARDWARE (Specs and FAQs about the board)

Weight: 3.4 oz/96 g

HOST PCI-104 BUS interface

PCI-104 is another stacking form factor like PC/104. See images below.



PCI-104 Specifications

- Compliant to PCI-104 spec, rev 1.1
- Compliant to PCI spec, rev 2.2
- DIP switch selectable PCI-104 stack level
- Bus Interface: Universal Signaling Voltage 3.3 V/5 V
- **Bus Speed**: 32bit address @ 33/66 MHz

Agency approval

- **TUV and UL**: The TSync-PCI104 boards are not TUV or UL approved.
- **CE**: The TSync boards are CE approved. Link to Declaration of Conformity:

  https://oroliagroupemeamicrosoftonlinecom-1.sharepoint.emea.microsoftonline.com/Spectracom/Engineering/products/TSync/Shared%20Documents/Declaration%20of%20Conformity.pdf

Size of the TSync-PCI-104 board

Number of boards that can be installed in a single PC

The TSync driver is set to support up to 8 TSync boards in a single system. For all example programs provided, the first parameter passed in is the board instance number, which will identify the card to perform the action on.

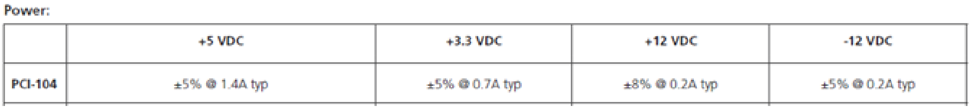
TSync-PCI104 Specs:

"Short term tracking": Is the oscillator stability when the board is synchronized to GPS. The values relate to our stability while disciplining the oscillator to GPS.

“Loss of satellites” is the oscillator stability when the board is in holdover. So for TCXO our oscillator stability is 1E-6 (1 ppm), and for OCXO our oscillator stability is 5E-8 (50 ppb).

“200 MHz Disciplined On-Board Clock”: This is the speed of the KTS timing system. This is the speed that all of the functions (such as disciplining) operate at. This is how we get the 5 ns accuracy resolution in the specs.

Power specs



Power:

	+5 VDC	+3.3 VDC	+12 VDC	-12 VDC
PCI-104	±5% @ 1.4A typ	±5% @ 0.7A typ	±8% @ 0.2A typ	±5% @ 0.2A typ

TEMPERATURE SPEC:

Q. (from Google) “For the PCI Express Time Code Processor card, what is the max temperatures for the processors/components on the board? I see in your spec you have a 75C max operating temperature but I believe this is an ambient spec. Can you clarify the actual chip max temp and where it is located?”

- The OCXO oscillator is the component on the TSync-PCI104 board with the lowest recommended temperature spec. The maximum case temperature spec for this component is +85C. The timing board is therefore derated to a spec of +75C to allow for local temperature rise on the board to remain below this specified threshold.

Power dissipation rating (Watts):

TSyncl-PCIe-OCXO

- 3.3V:0.8A, ~2.7W
- 12V:0.25A,~3W
- Total: ~6W out of 10W allowed (per the PCIe spec).

To begin, the 3.3v and 12v for the TSync-PCI104 board are provided by the host computer through the PCIe connector. As for the power consumption of the TSync-PCI board:

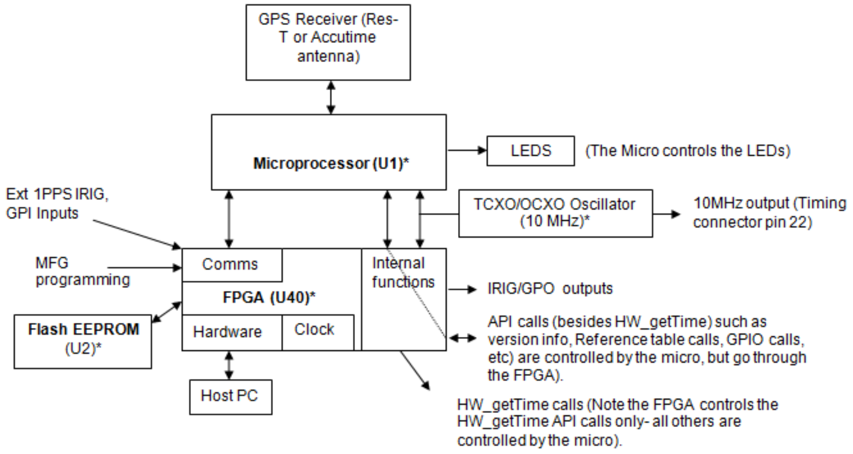
The TSync-PCI104 3.3V power is mainly consumed by the FPGA and microprocessor, 1.2V DC-DC converter, and OCXO. The 12V power is consumed mainly by the IRIG input, output circuitry and the +12V to -12V DC-DC converter. The 5V for the GPS antenna power is also consumed from the +12V by a 5V DC-DC converter.

All the parts dump their heat into the PCB and ambient air, but mainly the PCB which causes about a 10C rise of the PCB/parts surface temperature. That is why the board is rated to 75C. The IC’s and oscillators are all rated to 85C, so with a 10C rise on the board, they will be at 85C when the ambient is 75C.

TSync-PCI-104 MTBF

Refer to MTBF/MTTR (for all products) in this document.

Hardware block diagram (for internal use only)
Shortcut to TSync-PCI104 board schematics (not breakout cables): I:\Engineering\Schematic\1200-1001-0200


GPS Receiver (Res-T or Accutime antenna)
Microprocessor (U1)*
LEDS
(The Micro controls the LEDs)
Ext 1PPS IRIG, GPI Inputs
TCXO/OCXO Oscillator (10 MHz)*
10MHz output (Timing connector pin 22)
MFG programming
Comms
Internal functions
FPGA (U40)*
IRIG/GPO outputs
Flash EEPROM (U2)*
Hardware
Clock
API calls (besides HW_getTime) such as version info, Reference table calls, GPIO calls, etc) are controlled by the micro, but go through the FPGA).
Host PC
HW_getTime calls (Note the FPGA controls the HW_getTime API calls only- all others are controlled by the micro).

* denotes item is part of the Kramden Timing System (KTS)

Functional descriptions:

Microprocessor (U1):

- The Micro controls all API calls (besides HW_GetTime) such as version info, Reference table calls, GPIO calls, etc), but these go through the FPGA (not directly to/from the micro).
- The Micro controls the GPS receiver operation. The “GR_” API calls for the GPS receiver go through the FPGA to the micro.

FPGA (U40):

- The FPGA handles the communications with the Microprocessor.

Flash EEPROM (U2):

- The EEPROM programs and stores info for the FPGA.

Notes:

- The EEPROM is reprogrammed during the firmware update process.
- A programming issue with the EEPROM will prevent the micro from working.
- A problem limited to only the micro allows HW_GetTime API calls to still work, but no other API calls will work (try several different API calls to verify micro issue).
- A problem with the FPGA will allow GPS operation to be normal (can still sync and the LEDs will operate normally) but the user won’t be able to communicate with the GPS receiver, for instance, because API calls can’t get through.
- If the FPGA is not operating, even if the micro is still running, there will be no communication with the micro,

so the micro will appear to not be running. However, with the micro still running, the LEDs will still work.

I was just speaking to one of our timing board hardware engineers about the symptoms the TSync-PCI104 board is exhibiting (specifically, only the HW_GetTime APIs working but no other APIs are working).

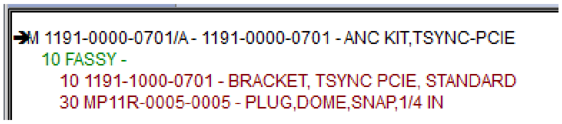
Based on these symptoms, it appears there may have been a problem with the firmware upgrade to the EPROM that is associated with the microprocessor. If the EEPROM is "corrupted" in any way, the microprocessor can't run.   The FPGA is likely still running as it handles the hardware time API (as you reported).   However, all other APIs are handled by the microprocessor, which is apparently not running, because all other API calls are failing.

The EEPROM can become "corrupted" if the wrong upgrade bundle is applied first, then another bundle is applied over the top of the wrong update bundle. You chose the correct upgrade bundle to use (TCXO with no GPS), but either another bundle was used first, or more likely, something happened to the EEPROM during the firmware upgrade process.

With these symptoms, it looks like this TSync board will need to be returned for a reprogramming of the EEPROM.   This is especially evident since the issue cleared with the spare board installed in its place! Based on the symptoms, I don't think installing this board in another CPU will make any difference. It will likely operate the same way in the other CPU, confirming this explanation.

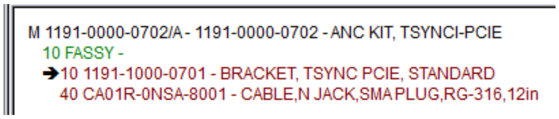
Ancillary kits
(this is PCIe- needs to be changed for TSync-PCI104)

TSync-PCI104   with no GPS receiver ~~1191-0000-0701~~



```
➔M 1191-0000-0701/A - 1191-0000-0701 - ANC KIT,TSYNC-PCIE
   10 FASSY -
      10 1191-1000-0701 - BRACKET, TSYNC PCIE, STANDARD
      30 MP11R-0005-0005 - PLUG,DOME,SNAP,1/4 IN
```

TSync-PCI104 1191-0000-0702



```
M 1191-0000-0702/A - 1191-0000-0702 - ANC KIT, TSYNCI-PCIE
   10 FASSY -
   ➔10 1191-1000-0701 - BRACKET, TSYNC PCIE, STANDARD
      40 CA01R-0NSA-8001 - CABLE,N JACK,SMA PLUG,RG-316,12in
```

12 inch GPS coax cable
(SMA to N adapter)

P/N 07013049 (SMA Male to TNC Female adapter)

Google purchased (16) SMA Male to TNC Female adapter with our P/N 07013049.   Reference Order NumberT-39734.   Not sure why these were purchased for use with the TSync-PCI104 boards.

Ancillary kits

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Mounting Brackets

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Timing Connector

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Breakout cables for the “Timing” connector

- Refer to the TSync-PCIe section (same as the TSync-PCIe)

Link to board schematic: I:\Engineering\Schematic\1191-1001-0200

Breakout cables for the “Timing” connector

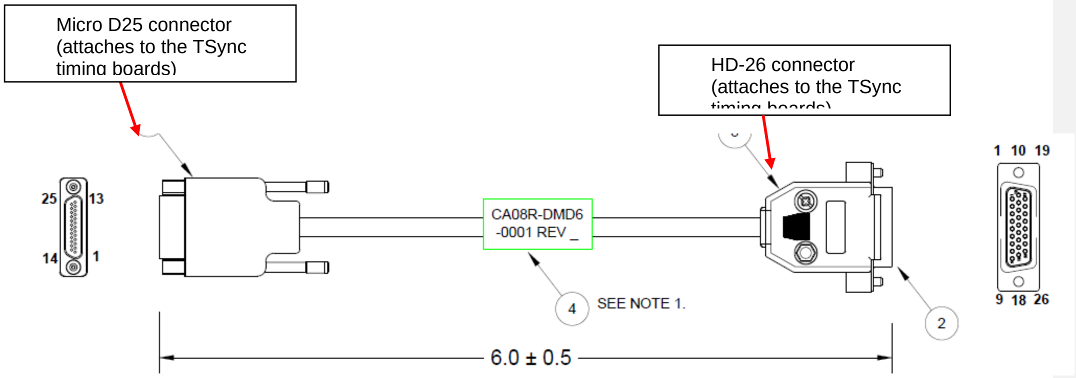
(this is PCIe- needs to be changed for TSync-PCI104)

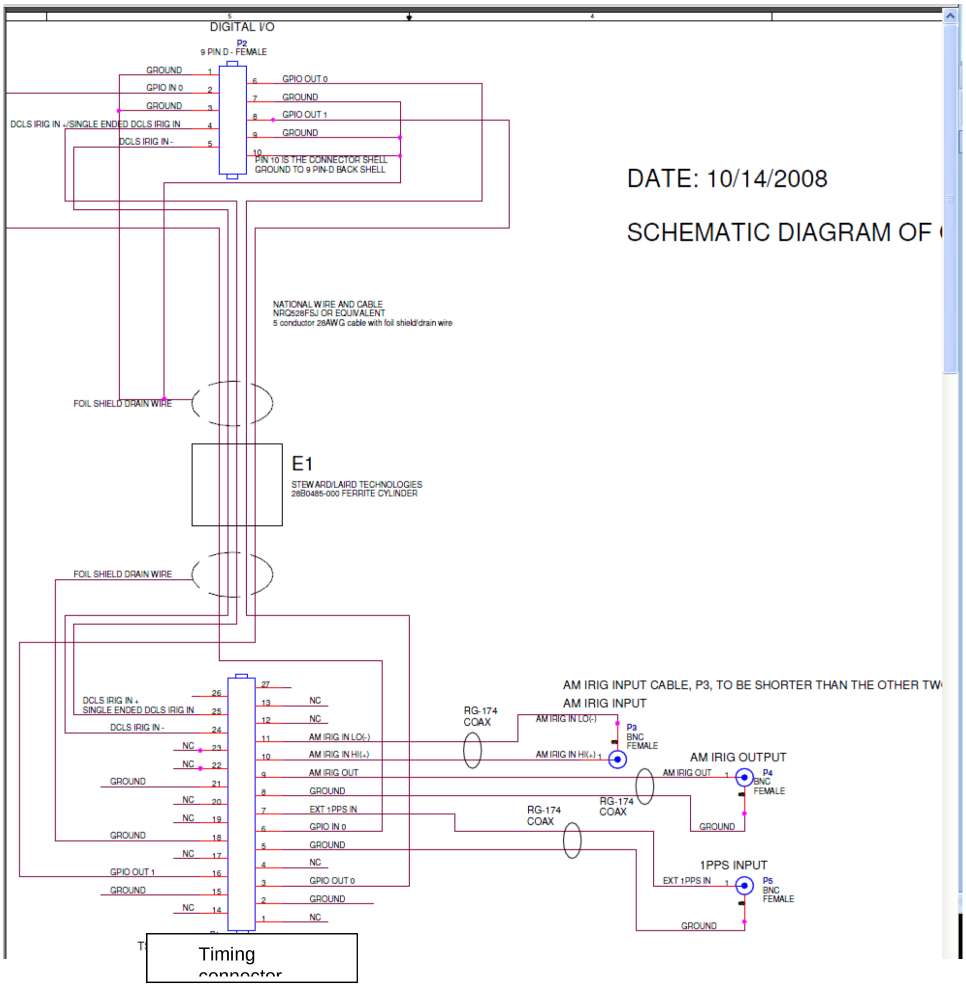
- Link to Breakout cable data sheet: I:\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards\TSync-PCI104 _Config_revB.pdf (refer to the second page of this document for a table).

CA08R-DMD6-0001 (Micro D25 to HD26 adapter cable)

- Link to schematic (CA08R-DMD6-0001) I:\New Released\Cable Drawings
- (installed between TSync boards and newer breakout cables which started shipping around Nov 2013)
- Micro D25 end attaches to timing board.

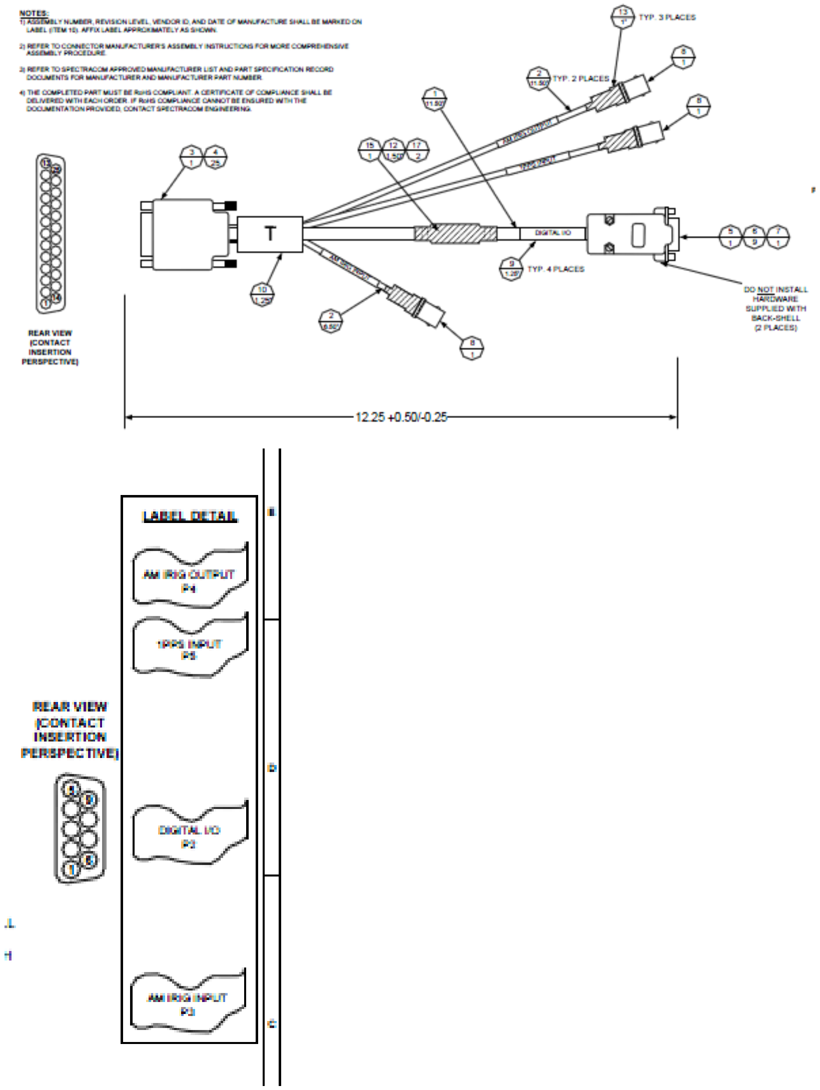
- HD26 end attaches to breakout cable





Shortcut to picture of standard breakout cable (CA08R-0000-0003, as shown below):
I:\New Released\Cable Drawings\

Note: CA08R-0000-0003 is discontinued. Replaced by CA08R-0000-0006 on ECN 3433, Mar 2014


NOTES:
1) ASSEMBLY NUMBER, REVISION LEVEL, VENDOR ID, AND DATE OF MANUFACTURE SHALL BE MARKED ON LABEL (ITEM 10). AFFIX LABEL APPROXIMATELY AS SHOWN.
2) REFER TO CONNECTOR MANUFACTURER'S ASSEMBLY INSTRUCTIONS FOR MORE COMPREHENSIVE ASSEMBLY PROCEDURE.
3) REFER TO SPECTRACOM APPROVED MANUFACTURER LIST AND PART SPECIFICATION RECORD DOCUMENTS FOR MANUFACTURER AND MANUFACTURER PART NUMBER.
4) THE COMPLETED PART MUST BE RoHS COMPLIANT. A CERTIFICATE OF COMPLIANCE SHALL BE DELIVERED WITH EACH ORDER. IF RoHS COMPLIANCE CANNOT BE ENSURED WITH THE DOCUMENTATION PROVIDED, CONTACT SPECTRACOM ENGINEERING.
TYP. 3 PLACES
TYP. 2 PLACES
TYP. 4 PLACES
T
DIGITAL I/O
DO NOT INSTALL HARDWARE SUPPLIED WITH BACK-SHELL (2 PLACES)
REAR VIEW (CONTACT INSERTION PERSPECTIVE)
12.25 +0.50/-0.25
LABEL DETAIL
AM IRIG OUTPUT P4
1PPS INPUT P5
DIGITAL I/O P2
AM IRIG INPUT P3
REAR VIEW (CONTACT INSERTION PERSPECTIVE)

Standard Breakout Cable supports the following:
Input Signals:
1 ea. IRIG AM (BNC –P3)

1 ea. IRIG DCLS (D Connector – P2)

Signal	Timing Connector pin	B/O cable D Connector – P2 pins
DCLS IRIG IN + (or single-ended)	25	4
DCLS IRIG IN -	24	5
Ground	2, 15	10

1 ea. 1PPS (BNC – P5)

1 ea. GPI (General Purpose Input) (D Connector – P2)

Signal	Timing Connector pin	B/O cable D Connector – P2 pin
GPI Input 0	6	2
Ground	2, 15	9

Output Signals:
1 ea 1PPS output (added ~Oct 2013)

1 ea. IRIG AM (BNC connector - P4)

- ea. GPO (General Purpose Output) (D Connector – P2)

Signal	Timing Connector pin	B/O cable D Connector – P2 pin
GPI Output 0	3	6
GPI Output 1	16	8
Ground	2,15	9

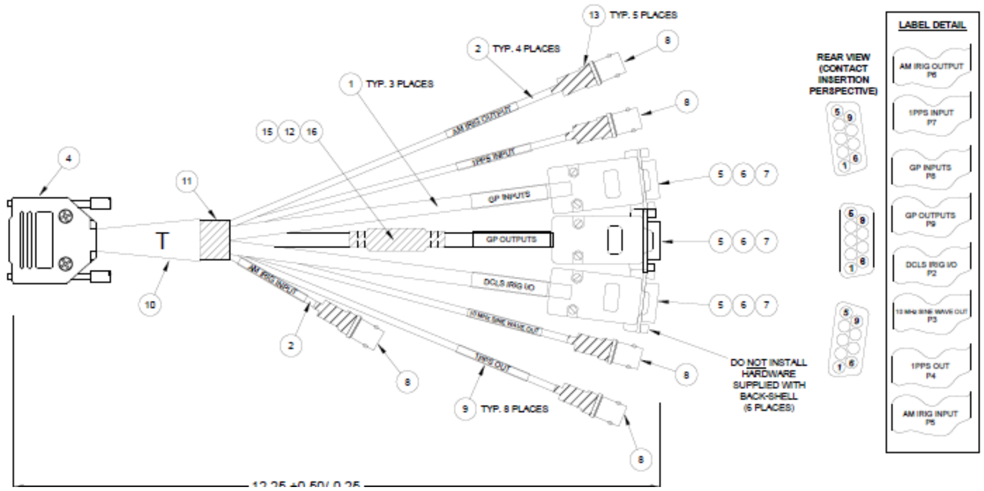
- Premium Breakout cable

(this is PCIe- needs to be changed for TSync-PCI104)

Note: As of march 2014, there are two versions of the Premium breakout cable. Number 1 below is the current version as of ~Apr 2014, while Number 2 was replaced by the first one below.

- CA08R-0000-0005 (More recent as of ~Apr 2014)
- Shortcut to schematics (CA08R-0000-0005) in Arena at: https://app.bom.com/items/detail-spec?item_id=1203165794&version_id=10257958418&orb_msg_Single_Search_p=1&redirect_Seqno=7069085981
- Replaces discontinued breakout cable CA08R-0000-0001 (ECN 3433, Mar 2014)

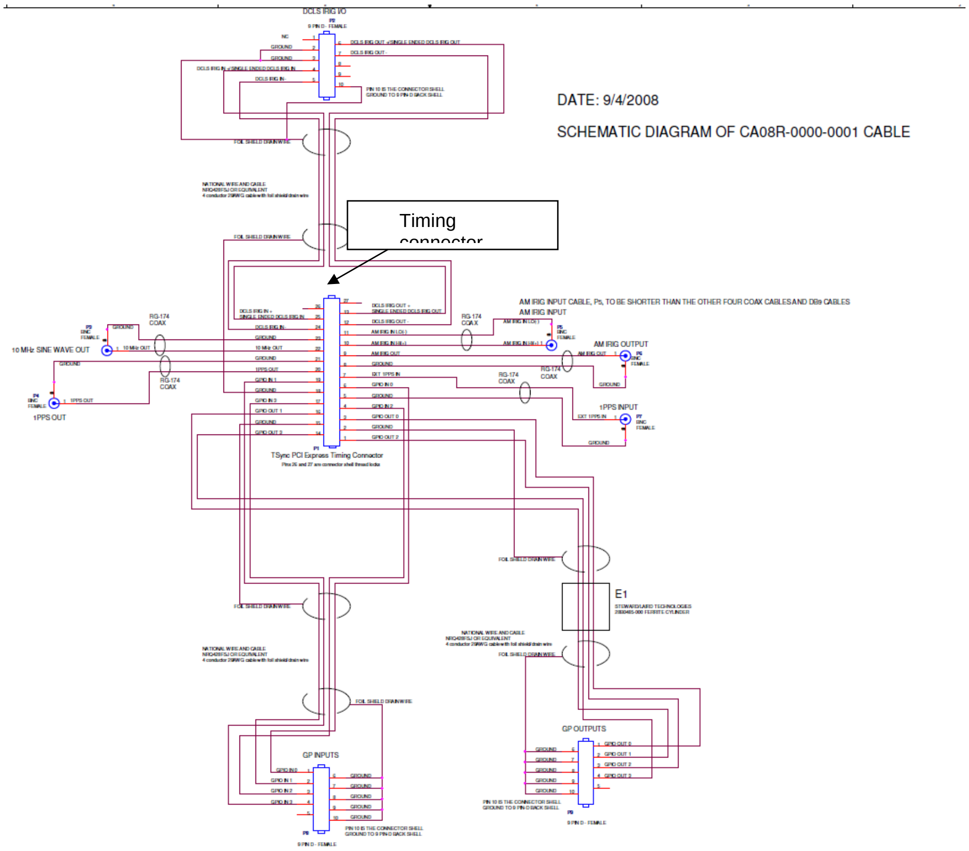
- **Note**: To use these newer breakout cables, an adaptor will also be needed. (CA08R-DMD6-0001)



- Our P/N: CA08R-0000-0001 (Discontinued, Mar 2014)
- CA08R-0000-0001 replaced by CA08R-0000-0005 on ECN 3433, Mar 2014

Note: If one or more signals to or from the timing board don't appear to be present, verify the pins of the large 25 pin connector (which plugs into the Timing connector) doesn't have any pushed-in pins. Dave Sohn says he has seen the pins pushed (so they aren't making contact with the pins of the Timing connector) in a couple of instances.

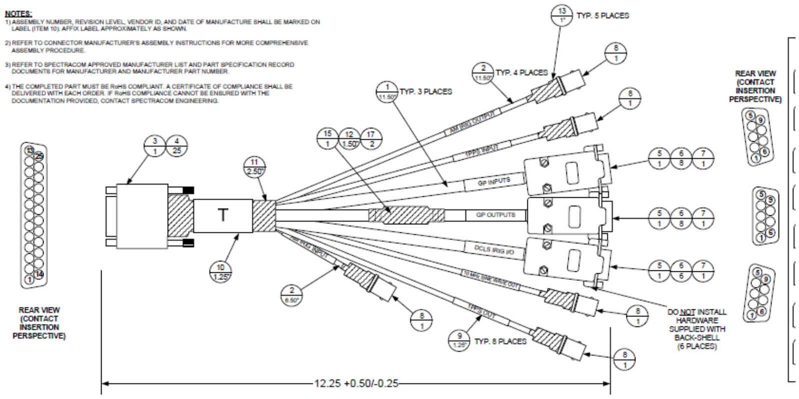
Shortcut to premium Breakout cable schematics (1191-1001-0400) I:\Engineering\Schematic\1191-1001-0400



DATE: 9/4/2008

SCHEMATIC DIAGRAM OF CA08R-0000-0001 CABLE

Timing connector



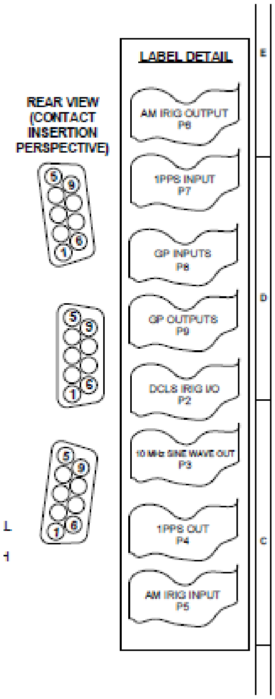
NOTES:

1) ASSEMBLY NUMBER, REVISION LEVEL, VENDOR ID, AND DATE OF MANUFACTURE SHALL BE MARKED ON LABEL (ITEM 10). AFFIX LABEL APPROXIMATELY AS SHOWN.

2) REFER TO CONNECTOR MANUFACTURER'S ASSEMBLY INSTRUCTIONS FOR MORE COMPREHENSIVE ASSEMBLY PROCEDURE.

3) REFER TO SPECTRACOM APPROVED MANUFACTURER LIST AND PART SPECIFICATION RECORD DOCUMENTS FOR MANUFACTURER AND MANUFACTURER PART NUMBER.

4) THE COMPLETED PART MUST BE RoHS COMPLIANT. A CERTIFICATE OF COMPLIANCE SHALL BE DELIVERED WITH EACH ORDER. IF RoHS COMPLIANCE CANNOT BE ENSURED WITH THE DOCUMENTATION PROVIDED, CONTACT SPECTRACOM ENGINEERING.



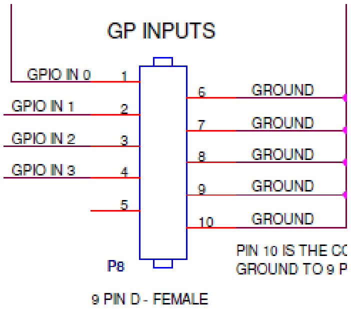
Premium Breakout Cable supports the following:
Input Signals:
1 ea. IRIG AM (BNC - P5)

- ea. IRIG DCLS (D Connector – P2)

Signal	Timing Connector pin	B/O cable D Connector – P2 pin
DCLS IRIG In + (or single ended)	25	4
DCLS IRIG In -	24	5

1 ea. 1PPS input (BNC - P7)

4 ea. GPI (General Purpose Inputs) (DB9F Connector - P8)

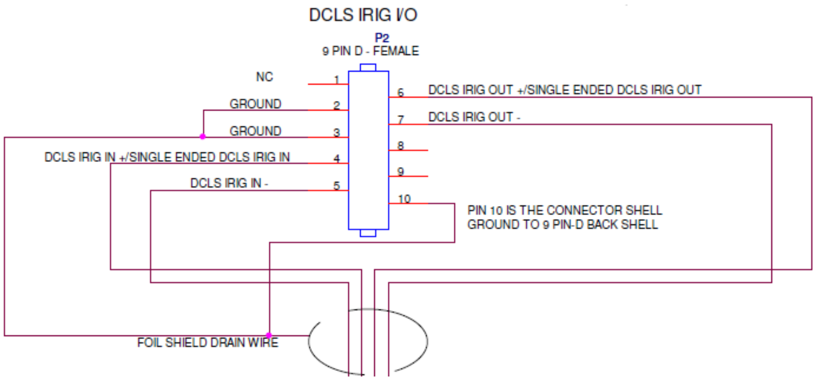


Signal	Timing Connector pin	B/O cable D Connector – P8 pin
GPI in 0	6	1
GPI in 1	19	2
GPI in 2	4	3
GPI in 3	17	4
Ground	2,15	6,7,8,9

Output Signals:

1 ea. IRIG AM (BNC – P6)

- ea. IRIG DCLS (D Connector – P2)

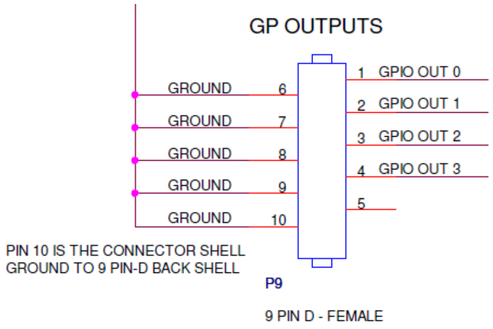

DCLS IRIG I/O
P2
9 PIN D - FEMALE
NC
GROUND
GROUND
DCLS IRIG IN +/SINGLE ENDED DCLS IRIG IN
DCLS IRIG IN -
1
2
3
4
5
6
7
8
9
10
DCLS IRIG OUT +/SINGLE ENDED DCLS IRIG OUT
DCLS IRIG OUT -
PIN 10 IS THE CONNECTOR SHELL
GROUND TO 9 PIN-D BACK SHELL
FOIL SHIELD DRAIN WIRE

Signal	Timing Connector pin	B/O cable D Connector – P2 pin
DCLS IRIG Out + (or single ended)	13	6
DCLS IRIG Out -	12	7
Ground	2,15	10

1 each 1PPS Output (BNC – P4)

1 ea. 10 MHZ Output (BNC – P3)

4 ea. GPO (General Purpose Outputs) (DB9F Connector – P9)



Signal	Timing Connector pin	B/O cable D Connector – P9 pin
GPI Output 0	6	1
GPI Output 1	19	2
GPI Output 2	4	3
GPI Output 3	17	4
Ground	2, 15	6,7,8,9,10

Power (+3.3vdc +12vdc and -12vdc from PCI bus)/logic
(this is PCIe- needs to be changed for TSync-PCI104)

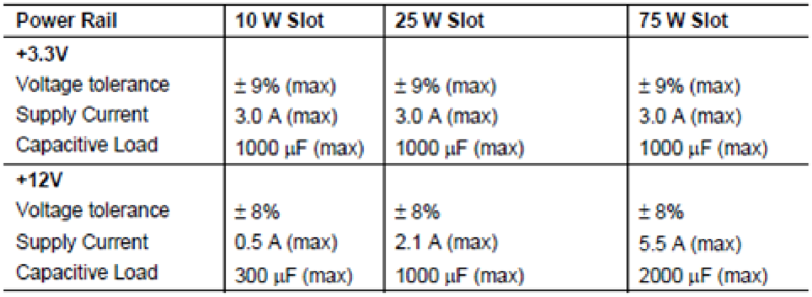
- Link to board schematic: I:\Engineering\Schematic\1191-1001-0200

+3.3vdc and +12vdc

- .TSync-PCI104 is a 3.3v power/logic card. It is not compatible with 5v systems.
- Because of this, a YELLOW key should be installed in the bus connector- to indicate it is a 3.3v board

Below for your reference are the PCIe power level specifications:

Table 4-1: Power Supply Rail Requirements



Power Rail	10 W Slot	25 W Slot	75 W Slot
+3.3V			
Voltage tolerance	± 9% (max)	± 9% (max)	± 9% (max)
Supply Current	3.0 A (max)	3.0 A (max)	3.0 A (max)
Capacitive Load	1000 µF (max)	1000 µF (max)	1000 µF (max)
+12V			
Voltage tolerance	± 8%	± 8%	± 8%
Supply Current	0.5 A (max)	2.1 A (max)	5.5 A (max)
Capacitive Load	300 µF (max)	1000 µF (max)	2000 µF (max)

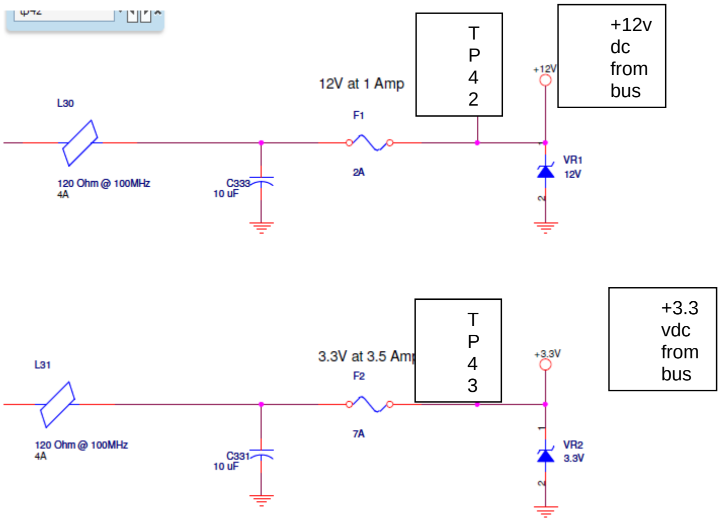
Based on the +3.3vdc specs, the maximum voltage drop for this rail is about 3.003vdc.

3.3vdc Low voltage detection circuit

- 3.3vdc is essential for operation of the board.
- If 3.3vdc is good, the three status LEDs on the edge of the board will all momemtarily turn-on at power–up, then all will go out (they remain ark until I syncs).

- If 3.3vdc is low, low voltage detection circuit will hold the board in reset- resulting in all of the LEDs remaining lit.

- If no 3.3 vdc present, the LEDS won't even mometarily turn on

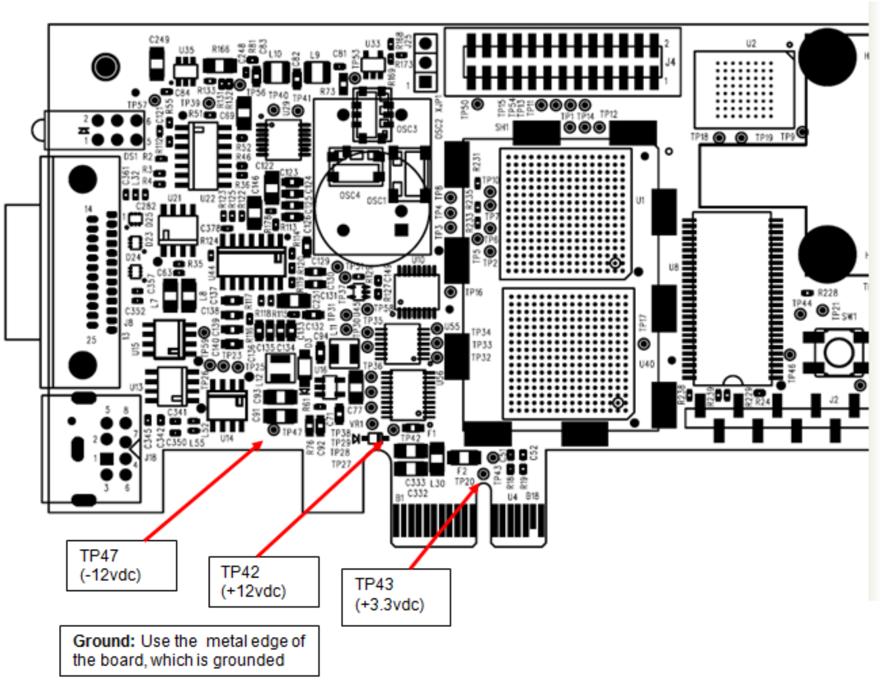


TP42

+12v dc from bus

TP43

+3.3 vdc from bus



TP47
(-12vdc)

TP42
(+12vdc)

TP43
(+3.3vdc)

Ground: Use the metal edge of the board, which is grounded

Email from Tim Tetreault to Mike Vinskus (6/26/12)
Your testing confirms what we were thinking is the problem. The 3.3v rail is dropping causing the low voltage IC on the board to hold it in reset mode. The threshold for the low voltage IC is 3.07v +/- 0.5% typ acc. So if you do the math, our spec for the 3.3v rail is 5% or about 3.14v, just above the threshold voltage.

When all the LED's on the board are light, that means it has been reset. When the 3.3v rail recovers, the board should come out of reset mode and start running again.

Note: The tolerance for the 3.3v rail is the same between both TSync-PCI104 and TSync-CPCI boards, 5%. The current draw is different.

3.3vdc input operating voltage from the PC:
Q. In evaluating our system that includes the TSync, we note that the PCI 3.3V supply to the card is only providing 3.15v. Your hardware spec says 3.3 +/-5%, which would mean it can work down to 3.13v. Any idea if running the voltage this close to the edge could cause issues? (We're trying to eliminate any possible sources of problems)

A. (From Dave Sohn) Problems with the 3.3V voltage would become very obvious as the board would be held into reset. It would lose all connectivity. If it is right on the edge, we've seen boards cycling through reset condition.

(From Dave Lorah): The 3.3V needs to be within the tolerance. We have seen boards have problems if the voltage drops below the lower limits. The boards would either cycle through a reset condition or go into constant reset, therefore halting all board functions. I would recommend not running the board that close to the lower limit to avoid potential problems if the voltage fluctuates.

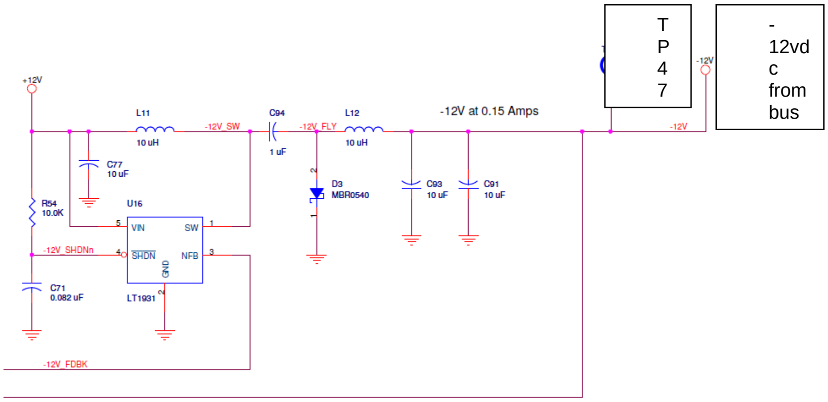
The Board may not be receiving the minimum of 3.1 vdc from the motherboard (stuck in a reset condition)

The slot that the timing board is plugged into may be a video-only slot (as either set by the motherboard manufacturer or as configured in BIOS). Example from David Hill "…. It turns out that the PCI express slot I was attempting to use was a video-only slot, so I am rounding up a new PC with an all-purpose PCIe slot".

Electrically, the slot should be no different than any other PCIe slot.   The only difference should be the BIOS looking for a video card to be present. This would, of course, prevent the board from being able to communicate with the bus. However, as long as the +3.3vdc is present, the board should at least still be able to boot-up (the LED on the board should illuminate and remain lit- not be turning off).

-12vdc

- IRIG input/output circuitry



Resetting the TSync-PCI104 board (Reset API call) and PTP module (if installed)
(this is PCIe- needs to be changed for TSync-PCI104)

- Command to reboot: **SS_Reset 0 0**

Note: Rebooting the PC doesn't restart the TSync-PCI104 board. Must power cycle the PC, power cycle the card or issue the Reset API call in order to restart the board.

API call that should always be used to reset the board without the likely need to power cycle the board is: **"SS_Reset 0 0**".   When the card resets, you should see all three LED's on the card flash briefly, then go out. If you are connected to an active reference, you should see the green "sync" light go on shortly after.

- Important Note about the reset command to keep the system from potentially hanging: Per Tim Tetreault - Make sure NTP and any application software that accesses the board is stopped before issueing a reset command. If anything such as NTP is accessing the board when the reset command is performed, the system may crash/hang!!

I have a quick question (and some info) for you that may help, before we "delve" deeper into the symptoms you are observing with the Spectracom TSync-PCI104 timing boards. The question is specifically about your phrase "**reset the**

card it causes the system to crash and reboot". It sounds like you are resetting just the timing board and not the whole system, correct? Are you using the "**SS_reset**" API call/example program to reset just the TSync-PCI104 board after the system is already powered up? If so, besides the "0" that follows the "reset" command to specify which installed board to reset, what other number are you using in conjunction with the reset command?

Please be aware that when using the SS_reset command, the corresponding number should be a "0", in addition to the other "0" to specify the board to reset. The full command should be **SS_reset 0 0** if there is only one board installed in the system. Using the reset command with anything other than a 0 causes only a partial board reset of the board and could cause the TSync-PCI104 board to become inoperative, until the system is power cycled. If the board becomes inoperative because of a partial reset, it could hang the PCIe bus and therefore also the system. Note that when the SS_reset 0 0 command is issued, you should observe the three status LEDs on the edge of the TSync-PCI104 board flash briefly, then all go out (until it resyncs to its input reference). If you aren't using SS_reset 0 0 to reset the board, the LEDs will not likely operate in this fashion (they may not light at all or may remain lit).

If you aren't issuing a SS_reset command to reset the board after system power-up, let me know exactly what is being done and then we can go from there! If you have been using a command other than SS_reset 0 0, please let me know this resolves the issue you reported – Thanks (I appreciate you letting me know that you are all set)!!!

Reset of the PTP Module (if installed)

Also, the PTP module on the TSync-PCI104 -PTP is its own embedded system that needs to be reset independently from the TSync card. Instructions on how to do that are in Section 5.7.1.1. of the Driver Guide. (Section 5.7 of the Driver Guide has lots of good information for understanding how to drive the TSync-PCI104 -PTP card.)

To reset the PTP operation:
PTR_ResetModule <device> <inst> <reset type>

```
      Reset Type 0 = Cold Reset
      Reset Type 1 = Hot Reset
      Reset Type 2 = Reset to Factory Defaults
```

Intermittent or constant system crashes/reboots occurring after a reset command is issued.

This abnormal operation could be caused in a Linux system that has ASPM still enabled (especially wih the later version 5 and earlier version 6 linux distributions) ASPM should be disabled to prevent it from adversely affecting the timing board. Refer to ASPM: **(ASPM) Power Saver/Power Monitor feature** for info on disabling ASPM.

Hardware status:
Run the **IN_GetStatus 0** command. Then run the **HA_GetCaps 0** command. Example responses below.

```
 [root@ntp01 /tmp]# ./IN_GetStatus 0

  Module                          | Result
  -----------------------------------------
  IRQ Driver                       | PASS
  Control Status Driver           | PASS
  SPI Driver                      | PASS
  Watchdog Driver                 | PASS
  Timer Driver                    | PASS
  Reset Driver                    | PASS
  Internal Flash Driver           | PASS
  External Flash Driver           | PASS
  LED Driver                      | PASS
  DAC Driver                      | PASS
  Digital Pot. Driver             | PASS
  EEPROM Driver                  | PASS
  UART Driver                    | PASS
  Local Bus Driver               | PASS
  Time Control Driver            | PASS
  1PPS Control Driver            | PASS
  Disciplining Driver            | PASS
  TCB Output Driver              | PASS
  ASCII Input Driver             | PASS
  ASCII Output Driver            | PASS
  IRIG Input Driver              | PASS
  IRIG Output Driver             | PASS
  GPI Driver                     | PASS
  GP Output Driver               | PASS
  Fixed-Freq Output Driver       | PASS
  1PPS Output Driver             | PASS
  Time-Stamping Driver           | PASS
  Component Interface            | PASS
  Persistent Data Service        | PASS
  Supervisor Service             | PASS
  Flash Manager Service          | PASS
  Clock Service                  | PASS
  Log Service                    | PASS
  Reference Monitor Service      | PASS
  Oscillator Monitor Service     | PASS
  Upgrade Service                | PASS
  Initializer                    | PASS
  Shared Memory Service          | PASS
  GPS Reference Component        | PASS
  IRIG Reference Component       | PASS
  ASCII Reference Component      | PASS
  HaveQuick Reference Component  | PASS
  1PPS Reference Component       | PASS
  Host Reference Component       | PASS
  Self Reference                 | PASS
  IRIG Output Component          | PASS
  ASCII Output Component         | PASS
  1PPS Output Component          | PASS
  Fixed-Freq Output Component    | PASS
  LED Control Component          | PASS
  Oscillator Component           | PASS
  GP Input Component             | PASS
  GP Output Component            | PASS
  Host Agent                     | PASS
  Internal Agent                 | PASS


 [root@ntp01 /tmp]# ./HA_GetCaps 0

  CAI  | IID  | Access
  --------------------
  0x25 | 0x00 | Both
       | 0x01 | Both
       | 0x02 | Get
       | 0x03 | Get
       | 0x04 | Get
       | 0x05 | Both
       | 0x06 | Get
       | 0x07 | Get
       | 0x08 | Set
```

	0x09	Get
0x28	0x00	Get
	0x01	Get
	0x02	Get
	0x03	Get
0x23	0x01	Both
	0x02	Both
	0x03	Set
	0x05	Both
	0x07	Both
	0x08	Both
	0x0A	Both
	0x0B	Both
	0x0C	Both
0x40	0x00	Get
	0x01	Both
	0x02	Get
0x24	0x00	Set
	0x01	Set
	0x02	Set
	0x03	Get
	0x04	Get
	0x05	Get
	0x06	Set
	0x07	Set
	0x08	Both
	0x09	Both
	0x0A	Get
0x21	0x00	Get
	0x01	Set
	0x02	Set
	0x03	Both
	0x04	Get
0x26	0x00	Set
	0x01	Set
	0x02	Set
	0x03	Set
	0x04	Get
0x20	0x00	Get
0x29	0x00	Both
	0x02	Get
	0x03	Both
	0x04	Both
	0x05	Both
	0x06	Get
	0x07	Get
	0x08	Get
	0x0B	Get
	0x0C	Get
	0x0D	Both
	0x0E	Get
	0x0F	Set
0x2A	0x00	Both
	0x02	Get
	0x03	Both
	0x04	Both
	0x05	Get
	0x06	Both
	0x07	Both

	0x08	Both
	0x09	Get
	0x0A	Get
	0x0B	Get
	0x0C	Both
	0x0D	Both
0x2E	0x00	Both
	0x01	Both
	0x02	Get
	0x03	Get
0x2D	0x00	Both
0x37	0x00	Both
	0x01	Both
0x38	0x00	Get
	0x01	Both
	0x02	Both
	0x03	Get
0x39	0x00	Both
	0x01	Both
	0x02	Get
	0x03	Both
	0x04	Both
	0x05	Both
	0x06	Both
	0x07	Get

Timing board conversions

**Replacing an existing TPRO/TSAT board with a TSync-PCI104 board
The TSync-PCI104 drivers contain all of the legacy TPRO/TSAT API calls. The TPRO.lib and TPRO.h files are compatible with the newly installed TSync board, as long as there is no need/desire to take advantage of the newer API calls (such as holdover, for instance), customer application software doesn't need to be changed. But, the application software needs to be re-compiled with the TSync-PCI104 driver before it will work with the TSync-PCI104 board.

Note: Customer must install the TSync-PCI104 driver when TSync-PCI104 board is installed. The previously installed TPRO/TSAT driver won't be able to communicate with the installed TSync-PCI104 board (the device ID is different). So, the TSync-PCI104 driver also needs to be installed (the TPRO/TSAT driver can either be uninstalled or installed simultaneously with the TSync driver, as desired.

GPS connection when converting from TSAT to TsyncE timing board

(**Note**- only the TSyncE can use the existing TSAT (Acutime) antenna. TsyncI requires a different antenna be installed)

The antenna connector that was originally plugging into the TSAT board now connects to an adapter cable (included in the TSyncE ancillary kit). The other end of the adapter connects to the antenna connector on the TSyncE board. Refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
.

****Desire to convert a TSyncI to a TsyncE / TsyncE to TSyncI**

Email from Dave Lorah to David Higgins (5/25/12)
We do not normally retrofit bus level boards but if you need to we can retrofit one for the following cost.

To add a GPS to your board will require you return the board to Spectracom for retrofit and reprogramming. Turnaround time is approximately 2 weeks.

To update to a TSyncI-PCIe (GPS with internal receiver) it would cost $1100. You would still need to additionally purchase a model

8225 antenna (about $250) and the appropriate antenna cable length and surge suppressor.

To update to a TSyncE-PCIe (GPS with external receiver built into the antenna) it would cost $1776.00. This would include the antenna and 100 foot cable. Please note this model cannot be used in a mobile application.

Warranty is 90 days or the remaining life of the warranty of the board, whatever is longer.

FIRMWARE

(this is PCIe- needs to be changed for TSync-PCI104)

****Firmware version information and updates -Firmware/FPGA/PTP module (if installed)**

Link to TSync Firmware version change document (and instructions to perform the firmware upgrade):

Refer to: \\\\Exchange\empshare$\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards

---

TSync-PCI104 Driver updates

Link to PTP module Firmware version change document (and instructions to perform the firmware upgrade):
Refer to: I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\TSync driver updates

---

TSync-PCI104 firmware updates

Refer to: PSB, PSP software updates\TSYNC boards\TSync firmware updates

---

Obtain Firmware/Software version information

LS_GetVersion / FS_GetVersion (API calls for version info)

(Email response from Denis Reilly 11/23/10) -Version information- It's complicated, because we are managing versions of the same timing software across various product lines. TSync and SecureSync share the same firmware, but are upgraded at different times.

To reconcile this, we created a Product-level versioning scheme, attached to each upgrade image, which is separate from the versions of the individual parts. For example, the release of TSync 2.1.0 actually incorporates KTS Firmware version 2.2.1, as well as TSync FPGA Version 2.1.0.

FS_GetVersion 0 0 (or 0 1) will get you the Product Version of the image.
(one set of arguments gets you the firmware version image version, and the other gets you the FPGA image version)

Email from Tim Tetreault (17 Mar 2014)
The only way to check the revision of the images on the board is to use the API command "./FS_GetVersion".

"./FS_GetVersion 0 0" will return the version for the runtime image of the firmware
"./FS_GetVersion 0 1" will return the version for the runtime image of the FPGA
"./FS_GetVersion 0 6" will return the version of the EEPROM image

LS_GetVersion 0 will get you the KTS Firmware Version only.

HW_GetFpgaInfo 0 will get you the FPGA firmware Version Information only.

PTR_GetModuleInfo will get you the PTP Module firmware (or use the QuickPTP GUI program)

Firmware updates/driver updates

With the exception of when updating the TSync-PCI104 firmware from version 1.x to 2.x, Denis Reilly recommends updating the TSync-PCI104 driver to the latest version (if it isn't already) before performing firmware updates. When updating from a 1.x version of firmware to a 2.x version, he recommends updating the firmware version first and then updating the driver.

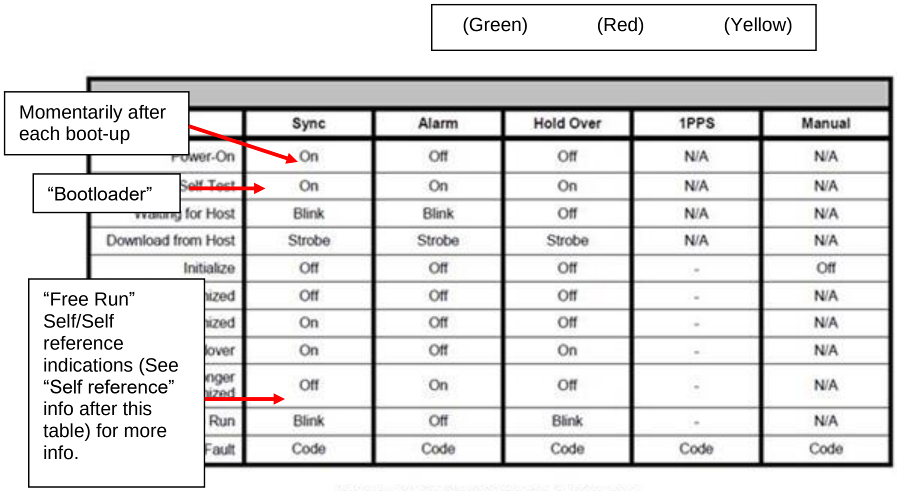
Note: Steps/directions to perform a TSync-PCI104 firmware update are in listed in Section 3-1 of the TSync-PCI104 driver guide.

Status LED’s/Status LEDs upon reboot

Right after the TSync-PCI104 board powers-up, the three LEDs on the edge of the board (red, yellow and green) turn on for just a moment and then turn off. After that, the LEDS will remain not lit, until a valid input reference (IRIG, GPS, or HST, Self) becomes available.

Status LED states during operation

- Green: Sync LED
- Yellow: Holdover LED
- Red: Alarm LED



(Green) (Red) (Yellow)

Momentarily after each boot-up

“Bootloader”

“Free Run” Self/Self reference indications (See “Self reference” info after this table) for more info.

	Sync	Alarm	Hold Over	1PPS	Manual
Power-On	On	Off	Off	N/A	N/A
Self-Test	On	On	On	N/A	N/A
Waiting for Host	Blink	Blink	Off	N/A	N/A
Download from Host	Strobe	Strobe	Strobe	N/A	N/A
Initialize	Off	Off	Off	-	Off
ized	Off	Off	Off	-	N/A
ized	On	Off	Off	-	N/A
lover	On	Off	On	-	N/A
nger ized	Off	On	Off	-	N/A
Run	Blink	Off	Blink	-	N/A
Fault	Code	Code	Code	Code	Code

Table 4-1: LED Flash Patterns

- *The code indicates the fault condition. It blinks the number of times indicated with a 2-second pause between each set.*

1 Blink = FPGA programming error
2 Blinks = Failure to decompress
3 Blinks = CRC failure writing to flash
4 Blinks = Self-test failure
5 Blinks = Timing system failure

In Sync mode

Green LED- Lit solid
Yellow LED- Not lit
Red LED- Not lit

In Holdover mode

Green LED- Lit solid
Yellow LED- Lit solid

Red LED- Not lit

LED indicators whe using Self reference for sync (Self/Self Reference)

Green LED and Yellow LED- blink simulaniously
Red LED- Not lit

The Self/Self reference has a unique LED pattern where the green (sync) and yellow (Holdover) LEDs will both blink simultaneously) when using the Self reference for sync. In at least the Rev E and below versions of the TSync-PCI104 manual, this pattern is indicted in the "LED Flash patterns" table as "Free Run".

While in this flash pattern, Sync will still indicate true and Holdover will indicate false. This is to indicate the TSync-PCI104 is being treated as if it's in continuous sync, as well as indicating the oscillator is in free-run mode (oscillator is not being disciplined).

Troubleshooting based on LED status

Check the Status LEDs on the edge and the green "FPGA load" LEDs on the back side of the TSync-PCI104 board:

- Verify all three Status LEDs (red, yellow and green) on the edge of the TSync board turn on and then go out (and remain not lit until an input reference becomes valid).

Right after the PC with the TSync-PCI104 board powers-up, the three LEDs on the edge of the TSync-PCI104 (the red, yellow and green LEDs), should momentarily illuminate and then turn off (until an input reference is connected). Do all of these LEDs momentarily turn on when the PC is first booted-up?

If the LEDs are not lit at power-up /all three constantly lit/exhibit the correct pattern:

- First, try to re-seat the TSync-PCI104 board in the system to make sure that all the contacts are solid.
- If the green and red LEDs are blinking on your PCIe card, it is likely that the board is in the bootloader waiting for image downloads from the host. This could signify that either the board needs to be reprogrammed or that there was a failure in the flash forcing it to fail to load the run-time or backup default images, landing back into the bootloader.
- If all three LEDs remain constantly lit (instead of all being lit only momentarily) Microprocessor likely"hosed"

MEMORY

**Sanitization/Memory storage after board is powered-down and removed
Also refer to the documents in: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\Memory

**On-board memory

The Spectracom TSync-PCI104 bus-level timing board is a module which can be installed in a PC to provide precision timing and interrupt functionality for a PC.

The TSYNC board contains a few persistent memory devices on the board. There is an 8 KB EEPROM, an 8MB Flash, and an internal 512KB Flash in the microcontroller.

- 8MB FLASH Memory

The 8MB Flash contains the run-time, default firmware and FPGA images; these FPGA images are binary files used to store the software and FPGA images used by the board during operation. These images are upgradeable via Spectracom-provided upgrade images using specific upgrade API calls.

Notes about the 8MB Flash memory

- The flash memory is a surface mount (not socketed) IC.
- This IC can be removed/replaced, but special steps are required upon a new IC being soldered onto the board in its place (this IC does not need to be pre-programmed before its installed, but other steps are needed after its been replaced- the board would need to be returned to us or special instructions would have to be supplied and performed in the field).
- If a customer desires to remove this IC and then send the TSync-PCI104 board to us for “recovery”:
- If the TSync board needs to be returned to us for a warranty type repair and the customer desired to remove this IC, we should treat this as a warranty modification.
- If the customer desired to replace the IC “just because” its being removed from a sensitive area, this should be treated as a non-warranty modification (time, materials and shipping fees should apply).

Partial email Keith sent to Jarred Gallagher with Raytheon (12/17/12)
Regarding your question as to if the flash memory can be removed from the TSync-PCI104 board: The flash memory is a surface mount IC soldered to the TSync-PCI104 board (it is not a socketed component).   This IC can be removed from the TSync-PCI104 board, if desired, and then it can be replaced with a new IC (this IC does not need to be pre-programmed before it’s installed on the board). Recovery of the TSync-PCI104 board with this new IC installed would then require either special instructions to be performed in the field, or the TSync board would need to be returned to the factory for “recovery”.   As a reminder, no user data is stored in the Flash memory.

- Internal 512KB flash in the microcontroller

The internal 512KB flash in the microcontroller contains the boot loader firmware and a compressed FPGA image. This is not accessible during normal operation.

- 8KB EEPROM

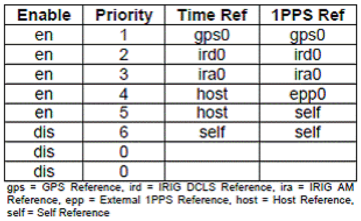
The 8KB EEPROM contains configuration data, which write the following during normal operation:

- Status flags for the images stored in the 8MB Flash, these are updated during the upgrade process.
- An uptime counter, which tracks the powered-on time of the board.
- The user reference table, which can be stored by a command from the user. This stored table can be reset to the factory default or overwritten by the user.

No other run-time information (time, position, user configuration) is persisted across power-cycles.

Resetting the 8k EEPROM: The user-configurable reference table is saved to EEPROM via the ***TSYNC_RS_SaveUserDef*** API call/example program. These values are reset to factory defaults via the ***TSYNC_RS_SetFactDef*** API call/example program.

The following was an email from either Denis Reilly or Dave Sohn
The reference table is the reference priority entries used to determine which references, and in which order references are used to synchronize the card. The user can choose to modify the default table and store it to persistent data in order to restore the modified settings on power-up. Below is an example reference table from the TSync-PCI104 manual:



Enable	Priority	Time Ref	1PPS Ref
en	1	gps0	gps0
en	2	ird0	ird0
en	3	ira0	ira0
en	4	host	epp0
en	5	host	self
dis	6	self	self
dis	0		
dis	0		

gps = GPS Reference, ird = IRIG DCLS Reference, ira = IRIG AM Reference, epp = External 1PPS Reference, host = Host Reference, self = Self Reference

Table 5-2 — Example Default Table

Also, I’m not sure what they want us to sanitize on return. We generated a listing of the persisted memory and the data that is contained within. I don’t think we take any special steps to sanitize anything on return. There shouldn’t be any information that the user can store that would require sanitization. If they need something beyond that I’m sure we’re willing to listen to any requests.

Partial email Keith sent to Jarred Gallagher with Raytheon (12/17/12)
Regarding your question “Where can I get step by step instructions to reset the EEPROM to factory defaults?”: The EEPROM stores only the following Runtime items- 1) the “Image OK” status 2) the Uptime counter 3) the User Reference Priority table (this is the list of input references that the TSync board can sync with and their priority of selection for its synchronization). Everything else stored in the EEPROM is limited to factory configuration information (such as oscillator calibration information).

The only item in the EEPROM that can be restored to factory default settings, if desired, is the User (Input Reference Priority) table. This input reference selection table (which has very specific, limited available entries- Such as “GPS 0”, “IRIG 0” and “EPP 0” for examples) can be reset (changed), if desired. To reset this User table, the User table can be modified/saved with different values than the currently stored values. For example, if the Factory default table has been modified and saved (resulting in a User table being created in the EEPROM) that lists “GPS 0” as a Time reference, and tied with an External 1PPS input (“EPP 0”) , as the first entry in this table, this particular “ID (row of the table) can be reconfigured as “ GPS 0” /” GPS 0”, for instance, and then the table saved. The User table will then no longer have the currently configured values.

The User table configuration uses the “**RS_”** API calls. Attached is a copy of the TSync-PCI104 driver guide. Refer to Section 4.2.24 (starting on page 4-218) for a list of all of the available RS commands used to configure the User (and Working) Reference Priority tables.

Input references / Time Sync / Holdover

**Input Reference Priority tables (Reference Monitor Service-“RS” API calls).

Three Reference Priority tables are maintained by the system:

0= Default factory table (this table never changes)
1= User Default (Saved) table that persists across reboots
2= Current working table (becomes the new User Default table when saved, or lost upon reboot/power- down)

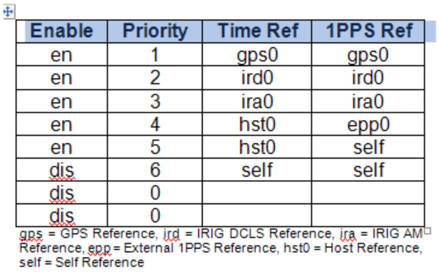
FYI- there are three types of Reference Priority tables – the default, working and saved tables. The saved table is the only configuration saved between reboots. If anyone had modified the working table by either disabling GPS input or deleting the GPS entry from the table, and then saved it, this new saved table is then used as the priority table after each reboot. Your code does not need to call a table at start-up. Otherwise the default table is always used at start-up (the working table is lost upon power cycle, unless it’s saved before power is lost).

A) Default Factory table (Table 0)

(this table never changes)

The Factory Table will load on startup if there is no User table defined.

A default (factory) table, which provides the default reference pairings in timing accuracy priority



Enable	Priority	Time Ref	1PPS Ref
en	1	gps0	gps0
en	2	ird0	ird0
en	3	ira0	ira0
en	4	hst0	epp0
en	5	hst0	self
dis	6	self	self
dis	0		
dis	0		

gps = GPS Reference, ird = IRIG DCLS Reference, ira = IRIG AM Reference, epp = External 1PPS Reference, hst0 = Host Reference, self = Self Reference

Table 5-5-2 — Example Default Table

O) User Default (Saved) table (Table 1)

(persists across reboots)

A user table, which can be stored persistently and, if present, will be loaded into the working table at startup.

To reset the User Default table and reset to the factory default table

Email from Denis Reily
blow away your default table and reset to the factory default. ./RS_SetFactDef 0
After you issue that, Table 2 should match Table 0, and Table 1 should be undefined. (The Factory Table will load on startup if there is no User table defined.)

P) Current Working table (Table 2)

(becomes the new User Default table when saved, or lost upon reboot/power- down)

A working (Active) table, which is the table used for selecting reference inputs

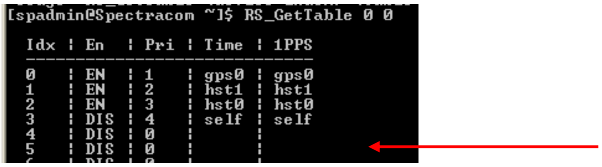
Email from Denis Reilly
You can play with the Working table as much as you want, and then save it to the User table when you get it right to persist across reboots.

(Modified) Email from Tim Tetreault
The Working/active table is what the board always uses. On power-up, if you have previously entered (Saved) a user table, the TSync board will load that into the working table and use it. If there is no user table on power-up, it will load the factory table into the working table.

The User (Saved) table is retained during reboots and power cycles. You just need to save the working reference table to the saved user table (**RS_SaveUserDef).**

Number of entries that can be added to the Reference Priority table:
There can be up to 14 combinations of references in the table (there are 14 indexes in the table

Trying to “Enable” indexes in tables that do not have references configured



```
[spadmin@Spectracom ~]$ RS_GetTable 0 0

  Idx ¦ En  ¦ Pri ¦ Time ¦ 1PPS
  ------------------------------
  0   ¦ EN  ¦ 1   ¦ gps0 ¦ gps0
  1   ¦ EN  ¦ 2   ¦ hst1 ¦ hst1
  2   ¦ EN  ¦ 3   ¦ hst0 ¦ hst0
  3   ¦ DIS ¦ 4   ¦ self ¦ self
  4   ¦ DIS ¦ 0   ¦      ¦
  5   ¦ DIS ¦ 0   ¦      ¦
  6   ¦ DIS ¦ 0   ¦      ¦
```

- ➢ In summary,Table indexes that are empty can’t be enabled.

Q.,If we try to ¿consult? again the states with ./RS_GetEnable, we get:

RS Table Entry 0 Enabled
RS Table Entry 1 Enabled
RS Table Entry 2 Enabled
RS Table Entry 3 Enabled
RS Table Entry 4 Enabled
RS Table Entry 5 Disabled
RS Table Entry 6 Disabled

You can see that number 5 and 6 didn’t change as we tried.

Can you help us for avoid the error we get when configure card with **RS_SetEnable 0 5 1 command.**
Error: invalid param (HA_RC_INVALID_PARAM).

- ➢ How many entries are in their reference table (RS_GetTable 0 2)? If they only have 5 entries, they will only be able to configure from index 0 to 4.

If the index is empty, it has to remain disabled and cannot be changed to enabled without adding references to it.

List of “RS” calls/Example programs

- ➢ “**TSync_RS**” (Reference Service) are the API calls associated with Input Reference. Refer to the TSync-family driver guide (1219-5001-0050) in Arena at:**Error! Hyperlink reference not valid.** https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

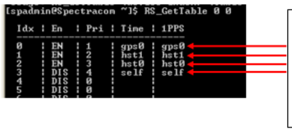
Note: Where **<enable>**: 0 for disabled or 1 for Enabled.

API call/example program	Function
RS_GetTable	Get specified reference priority table.
RS_GetBestRef	Get current best working priority table entry.
RS_GetEntry	Get working priority table entry by index.
RS_addEntry	Add an entry to the working priority table.
RS_SetFactDef	Reset the User priority table to the Default factory table.
RS_SetUserDef	Reset the Working table to the User default settings, if saved user priority table exists.
RS_SaveUserDef	Save the Working priority table to the user priority table
RS_deleteEntry	Delete a working priority table entry by index.
RS_GetPriority	Get specified working priority table entry's priority.
RS_SetPriority	Set specified working priority table entry's priority.
RS_GetEnable	Get specified working priority table entry's enable state.
RS_SetEnable	Set specified working priority table entry's enable state.
RS_GetStateTable	Get the reference validity state table.

RS_GetTable: Get specified Reference Priority table.

RS_GetTable 0 0 (default table)
RS_GetTable 0 1 (working table)
RS_GetTable 0 2 (user table)

_getTable: Get specified reference priority table.

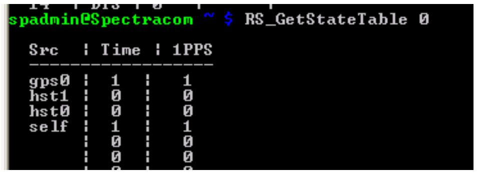


```
[spadmin@Spectracom ~]$ RS_GetTable 0 0

  Idx | En  | Pri | Time | 1PPS
  ------------------------------
  0   | EN  | 1   | gps0 | gps0
  1   | EN  | 2   | hst1 | hst1
  2   | EN  | 3   | hst0 | hst0
  3   | DIS | 4   | self | self
  4   | DIS | 0   |      |
  5   | DIS | 0   |      |
```

Each Horizontal row of the table consists of a paired Time and PPS reference.

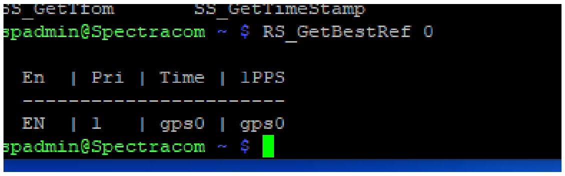
In order for a lower priority reference to become selected, both the Time AND PPS references in that row need to be valid (and the EN" field of that row needs to "EN" for Enabled). If one is valid and the other is not, it can't be selected.

RS_GetStateTable*:* Get the reference validity state table



```
spadmin@Spectracom ~ $ RS_GetStateTable 0

  Src  | Time | 1PPS
  ------------------
  gps0 |  1   |  1
  hst1 |  0   |  0
  hst0 |  0   |  0
  self |  1   |  1
       |  0   |  0
       |  0   |  0
```

RS_GetBestRef: Get the current best working priority table entry.



```
SS_GetTfom        SS_GetTimeStamp
spadmin@Spectracom ~ $ RS_GetBestRef 0

  En   | Pri | Time | 1PPS
  -----------------------
  EN   | 1   | gps0 | gps0
spadmin@Spectracom ~ $
```

User Table configuration Calls:

RS_SaveUserDef: resets the working reference table to the User (saved) table

RS_SetUserDef 0: resets the working reference table to the User (saved) table.

Desire to reset Reference Priority table back to default settings (for sanitization): Use the RS_SetFactDef API call/ example program

The number of entries that can be added to the Reference Priority table depends on how many "Indexes" (rows) are in the Reference Priority table. This is a user-configurable value. Issue the example program command: **RS_GetTable 0 2** to determine how many entries are in the table.

The first row starts with "Index 0", so if there are only 5 entries in the table, you can only configure index 0 to index 4 (which is a total of 5 entries). If you need to add additional entries to this table, use the **RS_SetTable 0 2** command to add additional indexes, as desired.

RS_AddEntry

Example: **RS_AddEntry 0 1 1 self epp0 <enter>**

Modified Email from Dave Sohn

The default priority table doesn't include an entry for the external 1PPS without a time reference. The easiest solution is to add an entry into the priority table and then save that table, so it will always start up with it. To add an entry, they can use the **RS_AddEntry API** with the following parameters **RS_AddEntry 0 1 1 self epp0**, which has the following meanings:

0 = card instance 0

1 = enable entry

1 = priority level 1 (top priority)

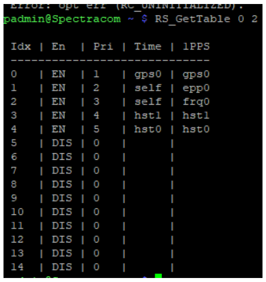
Self = the card can use its current time as a reference for time (this doesn't preclude updating the time from the host)

del = external 1PPS is the frequency reference.

They can then save the priority table by using the **RS_SaveUserDef <enter>** API call.

RS_DeleteEntry

1. Obtain the Index (left column) in the RS_GetTable for the row desired to be deleted

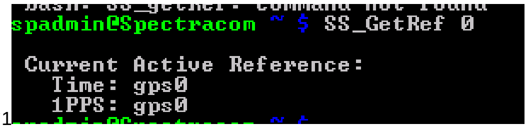


```
Error: opt err (RC_UNINITIALIZED).
padmin@Spectracom ~ $ RS_GetTable 0 2

 Idx | En  | Pri | Time | 1PPS
 -----------------------------
 0   | EN  | 1   | gps0 | gps0
 1   | EN  | 2   | self | epp0
 2   | EN  | 3   | self | frq0
 3   | EN  | 4   | hst1 | hst1
 4   | EN  | 5   | hst0 | hst0
 5   | DIS | 0   |      |
 6   | DIS | 0   |      |
 7   | DIS | 0   |      |
 8   | DIS | 0   |      |
 9   | DIS | 0   |      |
 10  | DIS | 0   |      |
 11  | DIS | 0   |      |
 12  | DIS | 0   |      |
 13  | DIS | 0   |      |
 14  | DIS | 0   |      |
```

2. Delete the desired Index (row): **RS_DeleteEntry 0 x** (where x is the Index of the row to delete)

Obtain the currently selected Time and PPS references

SS_GetRef 0 0
Get currently selected time and 1PPS reference.
SS_GetRef 0

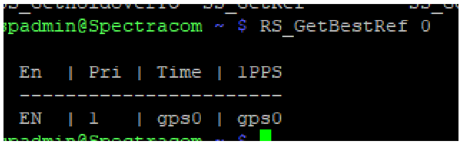


```
bash: SS_getRef: command not found
spadmin@Spectracom ~ $ SS_GetRef 0

 Current Active Reference:
    Time: gps0
    1PPS: gps0
```

`1

OR

RS_GetBestRef 0 0



```
spadmin@Spectracom ~ $ RS_GetBestRef 0

 En   | Pri | Time | 1PPS
 -----------------------
 EN   | 1   | gps0 | gps0
```

Switching from input reference to another

TSync-PCI104 can be "synced" to more than one reference at a time, with only the highest priority input selected at any given moment. Theoretically, if the highest priority input goes away or becomes not valid, SecureSync will seamlessly transition to the next highest priority without going into Holdover alarm (an entry is created indicating SecureSync switched references). However, Engineering has seen it glitch when switching from one input to another, where the Holdover alarm is generated for one second, and then it's back in sync by the next second. You can let the customer know that the switchover from one reference to another will take no more than one second, as long as more than one valid input is present.

**Desire to sync TSync-PCI104 board without external inputs (host/EPP and Self/Self)

A) Host mode: Allows the time to be set and then used as valid time. Note that Host time input is not considered valid until a user commands is to be valid via an API call). Host mode is for time input only. A separate 1PPS input has to be used (such as hst0/epp0). See Section A below.

Note: Typically, a script needs to be written to read the Host time, and then a API call or example program is used to set this in the board. In firmware versions above 1.x.x, the same call/example now allows the time to be set and the time declared valid, with one call, instead of two separate calls.

B) Self mode: Having a user manually set the time and 1PPS. Self time/1PPS is automatically considered valid (No user interaction is required to declare the time is valid). See Section B below.

Note: Self and hst0 modes are very similar. Both require a command to set the time/date. The biggest difference is hst0 mode provides more control on declaring the time valid. With Self mode, once time is valid, it will always be valid while it remains powered- up (unless superseded by a higher priority reference). Host mode allows the time to be commanded to be invalid.

A) Host mode-Desire to use the host PC to be the external time reference for the board
Refer to the TSync-PCI104 driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002

Note: While the time is manually set (no other higher priority references available), there is no external reference to use for monitoring the oscillator frequency. Therefore, the Frequency Error alarm will remain asserted until an external 1PPS input reference becomes available.

Host time allows the time of the TSync board to be set to a user-specified time and with the same call or example program, the time is also declared valid. The TSync-PCI104 board and associated driver don’t have the ability to read the computer’s time/date automatically (The computer’s time can be read automatically, but this has to be done outside of the TSync-PCI104 ’s driver). Then, once the time/date has been read, a call can be used to simultaneously set the time and declare the time valid. The reading of the computer’s time and the setting of the time/date in the TSync-PCI104 board can be performed with a customer-written Windows script.

A good starting point to create a Visual Basic script that can read the Windows PCs date/time is http://technet.microsoft.com/en-us/library/ee198917/

From the user manual
The TSync-PCI104 can be set to use the host as the source of date and time information, while another input reference is required to serve as the source of frequency input. This allows the host to provide time to the board while providing a means to determine and indicate whether that time is valid for synchronization. Using the host as a reference means it could conceivably be used to receive date and time information from a source not available to the board, providing that information to the board for synchronization to it (while using a separate frequency input).

1/11/11 KW- Note that “Host” in TSync-PCI104 has been changed to “hst0”
Due to “recent” change to KTS in SecureSync to have more than one host in SecureSync (one “host” is for user–set time and the other is for NTP input). In TSync-PCI104 , as of today anyways, there is only one host, so the only reference that will be used for “host” is now “Hst0” (instead of “Host”. ”Hst1” is only used in SecureSync).

Email from Dave Sohn about this:
The host reference is no longer a singular reference. We can have more than one, so the reference names for host are now similar to other. It is “hstx” where x is the host instance number starting from 0. So, the TSync includes a “hst0” reference now instead of “host”. We will have to update any documentation that might still refer to “host” as a reference name.

Response from Adam Tim (Tetreault) just helped me get this updated – the changes were made in both the user manual Rev E., and the driver guide Rev. G (ECN 2070).

In firmware versions prior to version 2.0.0, using host was a two-step process. The first step was to set the time to the host time. The second was to then declare this time valid. In firmware versions 2.0.0 and above, one call both sets the time and as long as Host is the selected input, the time is declared valid at that particular moment only.

The TSync-PCI104 boards do not retain the Year, date or time information. So at each start-up, it must be able to obtain it from a reference, or be manually configured with this information. In earlier versions of the TSync-PCI104 firmware, you needed to set the time and then declare the host reference time valid with a separate API call, each time the board starts up. In newer versions of the firmware, the setting of the time to the host time (**HR_SetTime**) automatically sets the date/time and declares it a valid input for that one particular second, assuming the host reference is the selected input reference (as long as no other higher priority input references in the Input Reference Priority –such as GPS, IRIG, etc are available). If there are any other higher priority inputs when the time is set, the host time is not considered valid.

To manually set the time of the TSync-PCI104 board using host mode:

- With the Host mod (hst0), the time of the TSync board is manually set using the

HR_SetTime API call (The **CS_SetTime** call is used with the Self reference = not with host).

HR_SetTime (device index – typically 0) (index – 0) (year) (DOY) (Hour) (minute) (second) (nanoseconds)

Note: In firmware version **above 1.x.x,** this command also declares the time valid.

Q. I noticed that after I set the time with: TSYNC_CS_SetTimeSec(index, seconds, ns) I continue to read old time values for approximately 1250ms.Is this normal? This is causing me problems. Other than sleeping for 2 seconds I don't see a good workaround.
What do you recommend?

- Below is the response from Engineering related to your recent question.

"The delay the customer is seeing when running a CS_SetTime makes sense. The time is only set on the "On time point" which is once a second. There is also some overhead required on the board so I can see it taking 1.5 seconds for the new time to be seen."

- (Note: Applicable to 1.x.x firmware versions only).With Host mode, you also have to command validity of the Host time before it can use the Host time as a reference. TSYNC_HR_SetValidity 0 0 1 1 <enter> *(*Set the reference validity of the Host)

B) Self/Self Reference (Desire to sync the TSync board to itself)
The TSync-PCI104 board can accept Self/Self mode for synchronization with no external references. However, after each power-up, the user has to set the time, to declare the time is valid.

LED indicators when using Self reference for sync
The Self/Self reference has a unique LED pattern where the green (sync) and yellow (Holdover) LEDs will both blink simultaneously) when using the Self reference for sync. In at least the Rev E and below versions of the TSync-PCI104 manual, this pattern is indicted in the "LED Flash patterns" table as "Free Run".

While in this flash pattern, Sync will still indicate true and Holdover will indicate false. This is to indicate the TSync-PCI104 is being treated as if it's in continuous sync, as well as indicating the oscillator is in free-run mode (oscillator is not being disciplined).

Info from the TSync-PCI104 user manual

The TSync-PCI104 provides a built-in reference that allows the board to operate without a separate input reference. The date and time or frequency information from this self reference is always considered valid. This allows a user to operate the board as if it were synchronizing to an input reference, without a valid external reference input. The self reference priority table entry defaults to “disabled. (By factory default, the input reference table contains a Self/Self entry. However, this entry is disabled by default)

The steps to manually sync the board are below

Note: These steps have to be performed each time the TSync-PCI104 board boots-up

Q) To manually set the time of the TSync-PCI104 board using Self mode (this step needs to be performed each time the board is powered-up/rebooted

Use the **CS_SetTime** API call to set the time/date as desired
Note: Entered as: CS_SetTime 0 0 <year> <DOY> <Hour> <Minutes> <Seconds>

Example: cs_Settime 0 0 2012 123 12 12 12 <enter> (year is 2012, 123rd day of the year, hours 12, minutes 12 seconds 12)

Note: With the Self mode (self), the **CS_SetTime** call is used with the Self reference. (“HR_SetTime” is only used with the host mode).
Regarding the need to manually sync the TSync-PCI104 board to itself, before it can sync a PTP slave, I have the following information for you:

Syncing the TSync-PCI104 -PTP that is configured as the PTP master to itself consists of two steps. The first step consists of enabling a default disabled row of the Input Reference Priority table. This configuration change only needs to be performed once, as changes t this table persists through each reboot. The second step is after each time the board powers-up, the time/date of the board needs to be set/declared valid.

FYI- there are three types of Input Reference Priority tables. One of these is the Default table, (defined as type 0). This table contains the factory default settings. The second is the User table (defined as type 1). This table is created when the default table is changed and those changes are saved. The third table is the Current working table (defined as type 2). This table is created when changes have been made to either of the first two tables, but the changes have not yet been saved. This table does not persist through a reboot of the TSync-PCI104 board (The changes have to be saved in order for it to persist). The changes are saved as the User table.

By factory default, the Default Input Reference Priority table contains an entry that lists “self” as both the “Time” and “PPS” input references. It is row 5 of the default priority table, But, by factory default, this row (referred to as an “index”) is set to disabled. It needs to be changed to enabled, before the board can sync to itself.

If any changes have already been made/saved to the default Input Reference Priority table, the default priority table becomes the User table. If no changes have yet been made to the default table, the default table is still used to define the input references. So, if there have been no changes made to the table yet, you want to edit the Default table. Otherwise, if changes have already been made to the default table, you want to edit the User table.

RS_GetTable 0 0 <enter> API call/example program will show the current settings of the Default Reference table, while **RS_GetTable 0 1** will show the current settings of the User (saved) Reference Table. To enable the Self/Self entry, edit the Default table if no changes have been made/saved to this table. Or, edit the User table, if any changes have already been made to this table.

To enable the Self/Self row of the User table, use the API call/ Example program entry by typing: **RS_SetEnable 0 5 1** <enter>. Then, to save the working table as the User table, use the example program **RS_SaveUserDef 0** <enter>. Now, when the board is rebooted, the entry will still be enabled.

Each time the TSync-PCI104 board powers back up, the time/date needs to be set, to validate the time. Use the CS_SetTime API call/example program to set the date and time. The format of this call is as follows:

CS_SetTime 0 0 <YEAR> <Day> <Hour> <Minute> <Second> (where “Day” represents the “day of the year”)

Example: Type**: CS_Settime 0 0 2012 123 12 13 14** <enter> (sets the TSync-PCI104 to year of 2012, the 123rd day of the year, and the time as 12:13:14).

After entering this line, using the desired values, the TSync-PCI104 board will go into holdover mode (Sync LED flashing green). Holdover mode is treated like the board is in sync.

****Desire to sync TSync board via NTP**

The TSync-PCI104 boards can’t sync via NTP directly. However, they can sync indirectly to NTP, if desired, when when installed in a Linux box running NTP software (Refer to “Hailer IT/Jeffries in SalesForce).   This was suggested by Dave Sohn, for this customer of Jeremy Onyan’s.

To sync the TSync board via NTP, use the NTP software to sync the linux kernel to the desired NTP server(s). Then, sync the TSync board to the system using the host/host input reference. The TSync board can then sync to the linux kernel time.   NTP maintains the kernel time and host mode maintains the TSync’s time.

Also refer to the previous section on using the Host mode to sync to the kernel time: Desire to sync TSync-PCI104 board without external inputs (host/EPP and Self/Self).

****Desire to sync TSync board via an Endrun Praecis CDMA receiver** (or other external references via ASCII data input)
Email from Dave Sohn (18 Dec 2012) There is currently no ASCII input on TSync boards, so there are no plans to support any ASCII formats on the TSync. This could potentially be done with the SecureSync, or a SecureSync synchronizing to the EndRun and then providing IRIG timing to the TSync.

If we wanted to consider this as a special, then we could look at it that way.

**Time Sync and Holdover status/ green “sync” LED indication

As soon as the TSync-PCI104 board goes into sync with its input references, the sync status in all **HW_Gettime** calls as well as the current sync status reported with the **SS_GetSync** call update immediately. However, the green “sync” LED (and the other two LEDs on the edge of the board) only update once per second. So, the calls could read synced just before the “sync” LED turns green.

There are a few different way to obtain sync/Holdover status of the TSync board

A) Determining current sync status via the LEDs on the edge of the board

(refer to: Status LED’s/Status LEDs upon reboot for more info)

B) Sync status associated API calls/example programs

Keep in mind that when using input references such as PTP or IRIG for synchronization, these inputs often have some jitter on the PPS its providing as a reference. The ontime point (1PPS output) of the TSync board will not jump to keep synced with this jitter. Instead, it will be very slowly slewed via disciplining to align with the reference. This will help dampen out any jitter of the PPs input reference (the TSync board will only hardware coarse adjust to the input PPS the first time after power-up that a PPS reference is selected and it syncs to the reference. Thereafter, its always slewed to the input reference.

Using the SS_GetSync and SS_GetHoldver API calls/example programs to determine current sync status The current sync status is also available via a couple of API calls/example programs.

- The current Sync status can be always be queried using the SS_GetSync call
- The **SS_GetSync** API call indicates True when TSync-PCI104 is either synced or in the Holdover mode. It reports false when not currently in sync or in Holdover mode.
- The current Holdover status can be always be queried using the SS_GetHoldover call (Responds with either "TRUE" or "False"

To differentiate between fully Synced mode and Holdover mode, the **SS_GetHoldover** API responds with True if in Holdover mode only. It will return a False if the board is in not in Sync or not in sync.

The command to read sync status is **SS_GetSync 0** API call (the "0" assumes there is only one TSync-PCI104 board installed). The response will indicate "true" when in sync or while in Holdover mode (all references have been lost). To differentiate between in full sync and Holdover mode, also use the **SS_GetHoldover 0** API command. If Sync is true but Holdover is false, its in full sync. But if Sync is true and Holdover is true, the board is currently in Holdover mode (no references currently available).

While in Holdover mode, the green LED on the edge of the board will remain lit and the yellow LED will also be lit. If both commands report False, the board is out of sync (not in sync and not in Holdover. Either it powered up and hasn't synced yet, or it initially went into Holdover mode and Holdover mode has since expired, because no references were restored before it timed-out).

Verify the TFOM and Phase Error values are low

When the TSync board is closely aligned with its 1PPS input reference, the TFOM and Phase Errors will be lower numbers (these numbers are oscillator-type dependant), Typically TFOM will be 4 or less and the Phase error will be like 200ns or less when in sync with the input PPS reference. The command to read the current TFOM value is **SS_GetTFOM 0** and the command to read the current phase error is **XO_GetPhaseError 0**.

Current sync status determined by time stamps received from the TSync board

- The timestamp captured when the sync status last changed states (either into or out of sync) can be   queried with the SS_GetTimestamp call.

Compare the 1PPS output of the breakout cable to the PPS input reference.
Another way to confirm the TSync board is in sync is to use an oscilloscope to compare the 1PPS output pin of the Timing connector to the 1PPS output/input reference. The closer the two 1PPS outputs are aligned with the other, the more closely aligned the TSync board is to its selected PPS reference.

Oscillator lock status API call/example program

TSYNC_SS_GetFreeRun: Gets the current freerun state.

**“Time” Input synchronization (PTP Slave, GPS input, IRIG input, Self, Hst0)

- Default power-up year/date/time: Jan 1, 2000 00:00:00 UTC

FYI- the TSync-PCI104 board does not have a Real Time Clock (RTC) installed. Each time the TSync-PCI104 board powers-up, it starts up as Jan 1st 2000 00:00:00 UTC and then its count-up each second from there (this is the same date/time its outputting via IRIG) . The start-up date/time can be automatically corrected via GPS or IRIG input (GPS also corrects the year, while IRIG input can correct the year in some cases). The date/time can also be manually corected, if no external inputs are being applied to sync it.

Input Reference designations for TSync

Reference Designation	Input	Refer to hyperlink Section further below
gps0	GPS	GPS input
ira0	IRIG AM reference	IRIG input
ird0	IRIG DCLS reference	IRIG input
epp0	External 1PPS reference	EPP0 input
hst0 (was host)	Host System Reference	Host mode
PTP0	PTP	PTP input
self	Internal reference	Self/Self)

When the TSync board first boots-up and receives its external input (such as GPS, IRIG or PTP), it will jump to the correct time and the sub-second counter will be zeroized to the correct microsecond. If the IRIG or other input is then removed, the oscillator disciplining controls the time. When the input is restored, the seconds will re-align and the oscillator disciplining will steer the sub-second counter. This is different from the older PCI cards which always jump to the correct time when the input is restored.

“The reason for the offset is that the PCIe does not automatically latch onto the 1PPS on time point in the way the TPRO-PCI does. Instead, the PCIe uses oscillator disciplining to gradually sync the IRIG signal to the on time point. The slew rate (rate of adjustment) in the oscillator disciplining algorithm is about **4 us/sec**. This means that it would take some time for the disciplining to eliminate an offset as large as the customer has seen (100 – 250 ms). However, the disciplining will eventually correct the offset. The customer just has to allow enough time for this to happen, as a 4 us/sec adjustment will not be seen on the scope using a millisecond scale. “

Note: Course adjustment only occurs after a power-up. After course we adjust, it switches to fine adjust. This process takes about three minutes. However, if the board goes into holdover mode and then the reference(s) return, course adjust does not occur. If the oscillator drifted over the time it was in holdover, it is fine tuned (slewed) to the input 1PPS over a very long time-frame. If it had drifted quite a bit, it may take several minutes to several hours to re-aligning.

Oscillator Slew Rate

After initial oscillator correction after power-up, the oscillator is then slewed to its input. The slew rate is very slow (~100ns per second).

Based on a slew rate of 4 us/sec, this equates to:

- **Per hour:** 14,400 us/hour (14 ms/hour)
- **Per day**: 345,600 us/day (345 ms/day)

B) GPS/Glonass input synchronization

Antenna Cabling

(this is PCIe- needs to be changed for TSync-PCI104)

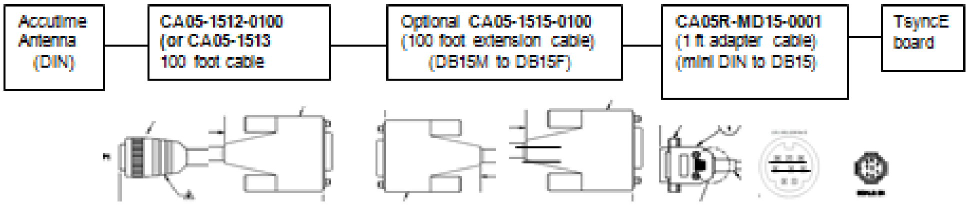
TsyncE (external GPS receiver) boards ship with:

- **E025R-0004-0003** (Acutime-GG Smart Multi-GNSS antenna) in arena at: https://app.bom.com/items/detail-spec?item_id=1202842906&version_id=10221223478
- ~~**E025-0004-0002** (Acutime-Gold Smart GPS antenna)~~ (discontinued~~)~~
  - Refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- **1159-0000-0700** (GPS mast) in Arena: https://app.bom.com/items/detail-spec?item_id=1202841005&version_id=10221291538&orb_msg_Single_Search_p=1&redirect_Seqno=6678014486
- **CA08R-0000-0006 (breakout cable)**

  Note CA08R-0000-0003 was discontinued. Replaced by CA08R-0000-0006 on ECN 3433, Mar 2014
- **CA05-1512-0100** (100 foot GPS cable) or **CA05-1512-0200** (200 foot GPS cable)
- **1191-0000-0710** (BNC T connector and 50 ohm terminator for 1PPS input)

Available (optional)100 foot GPS extension cable: CA05-1515-0100

Functional cable drawing



Accutime Antenna (DIN)

CA05-1512-0100 (or CA05-1513 100 foot cable

Optional CA05-1515-0100 (100 foot extension cable) (DB15M to DB15F)

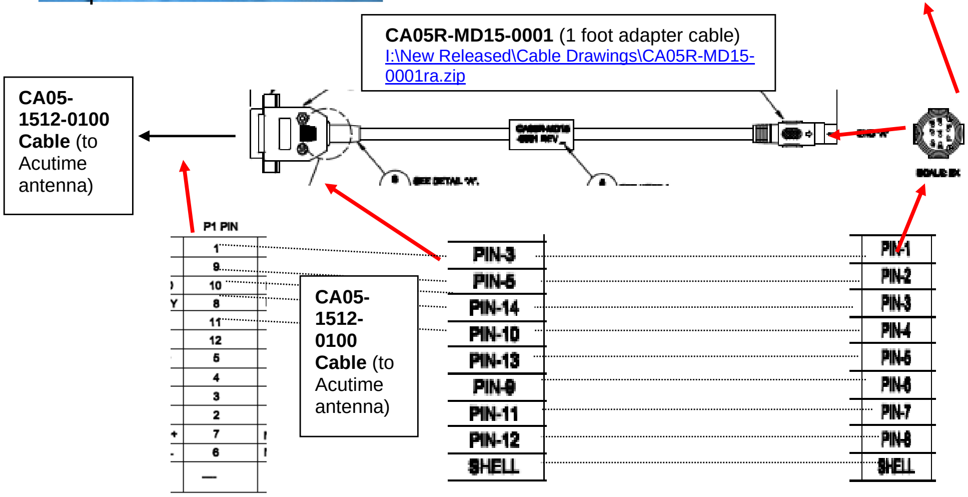
CA05R-MD15-0001 (1 ft adapter cable) (mini DIN to DB15)

TsyncE board

`

More detailed cable drawing

Acutime GPS antenna (E025-0004-0002)



CA05R-MD15-0001 (1 foot adapter cable)
[I:\New Released\Cable Drawings\CA05R-MD15-0001ra.zip](I:\New Released\Cable Drawings\CA05R-MD15-0001ra.zip)

CA05-1512-0100 Cable (to Acutime antenna)

P1 PIN
1
9
10
8
11
12
5
4
3
2
7
6
—

CA05-1512-0100 Cable (to Acutime antenna)

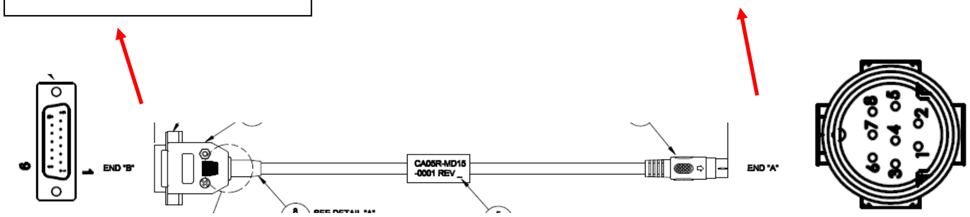
PIN-3	PIN-1
PIN-5	PIN-2
PIN-14	PIN-3
PIN-10	PIN-4
PIN-13	PIN-5
PIN-9	PIN-6
PIN-11	PIN-7
PIN-12	PIN-8
SHELL	SHELL

CA05R-MD15-0001 (supplied 1 foot adapter cable)

- refer to partial drawing below or full drawing in I\\rocfnp01\idrivedata\Engineering\Archive\New Released\Cable Drawings\ (CA05R-MD15-0001) is included in the TsyncE ancillary kit (1191-0000-0703)
- One side pof this **CA05R-MD15-0001** adapter cable plugs into the 8 pin MINI-DIN connector on the edge of the TsyncE board. Then, the other end of the **CA05-1512-100** cable is connected to the D Sub 15 pin connector on the other end of the supplied 100 ft adapter cable.

(DB15 connector) Attach to **CA05-1512-0100 (100 foot Antenna cable**) or to optional 100 ft extension cable

(**DIN connector**) Attach to edge of TsyncE-PCIe board

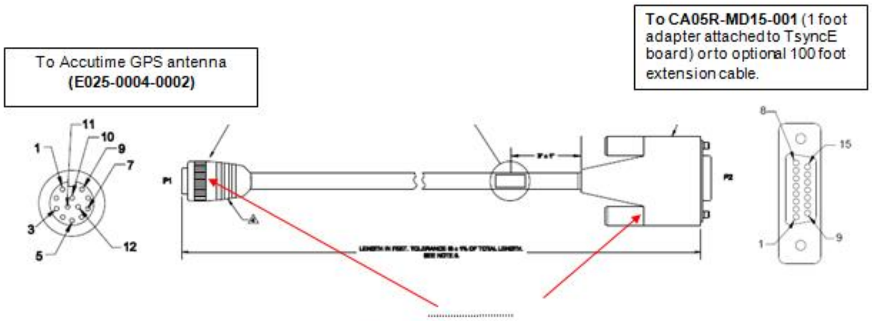


Signal Direction	End B (DB 15)	Signal	Wire Color	END A (mini DIN)
←	Pin 3	+12vdc to GPS receiver	Lt. Blue	Pin 1
↔	Pin 5	Ground	Lt. Green	Pin 2
→	Pin 9	"PPS +" (from GPS receiver)	Yellow	Pin 6
→	Pin 14	"PPS – RX" (from GPS receiver)	Purple	Pin 3
→	Pin 10	"Rcvr TX +" (from GPS receiver)	White	Pin 4
→	Pin 11	"Rcvr TX –" (from GPS receiver)	Brown	Pin 7
←	Pin 12	"Rcvr RX+" (to GPS receiver)	Red	Pin 8
←	Pin 13	"Rcvr RX –" (to GPS receiver)	Orange	Pin 5
↔	Shell		Braid	Shell

CA05-1512-0100 (100 foot Antenna cable) and CA05-1512-0200 (200 foot Antenna cable)

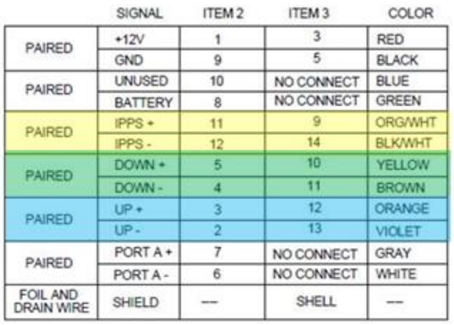
GPS (Acutime) antenna cable used with TSyncE-PCIe boards (CA05-1512-0100) and (CA05-1512-0200).

pin-out \\rocfnp02\idrive\Engineering\Archive\New Released\Cable Drawings\ (CA05-1512-0XXX)



To Accutime GPS antenna
(E025-0004-0002)

To CA05R-MD15-001 (1 foot adapter attached to TsyncE board) or to optional 100 foot extension cable.



	SIGNAL	ITEM 2	ITEM 3	COLOR
PAIRED	+12V	1	3	RED
	GND	9	5	BLACK
PAIRED	UNUSED	10	NO CONNECT	BLUE
	BATTERY	8	NO CONNECT	GREEN
PAIRED	IPPS +	11	9	ORG/WHT
	IPPS -	12	14	BLK/WHT
PAIRED	DOWN +	5	10	YELLOW
	DOWN -	4	11	BROWN
PAIRED	UP +	3	12	ORANGE
	UP -	2	13	VIOLET
PAIRED	PORT A +	7	NO CONNECT	GRAY
	PORT A -	6	NO CONNECT	WHITE
FOIL AND DRAIN WIRE	SHIELD	---	SHELL	---

Note: For pin-out info on this cable, refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

Mini-DIN Schematic (1191-1001-0200) from: I:\Engineering\Schematic\1191-1001-0200

GNSS (Glonass/GPS capability)

- All Tsync boards with GNSS receiver will ship pre-licensed for Glonass, with Glonass enabled
- New Res-SMT-GG receiver for TSyncI (internal receiver) cut-in ~Apr 2014 (ECN 3412) with Trimble receiver firmware version 1.06.
- Note: As of Dec, 2014, we are still shipping the receiver with v1.06 software installed

Requirements to use Glonass satellites

- TSyncI (internal receiver) requires Model 8230 antenna and RES-SMT-GG receiver be attached to the board.

Note: Approximate cut-over to RES-SMT-GG receivers was 1 Apr, 2014. Receiver firmware version at that time was 1.06. As of Dec, 2014, we are still shipping the receiver with v1.06 software installed

- SyncE (External receiver) need to use newer Acutime GG antenna (Refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf)
- Both TSyncI and TSyncE boards require firmware versions 2.2.1 (or v2.2.2 for PTP boards) or higher to support Glonass capability

Enabling /Disabling Constellations (GPS and/or Glonass)

- Can enable just GPS, just Glonass, or both constellations.
- There are only two commands associated with Glonass (GR_GetConstSet and GR_SetConstSet)

Speeding up time it takes to sync after each boot-up

I wanted to let you know there is a command available to be able to speed up the time it takes to sync the TSync-PCIe board after each boot-up. The delay to sync after each boot-up is the time it takes for the receiver to read the GPS to UTC offset (currently set to 16 seconds). This piece of data is transmitted by the GPS satellites every 13.5 minutes. Depending on when the receiver starts listening to the GPS signal in relation to when this information was last transmitted, it can take up to 13.5 minutes to read this value (worse-case scenario. it if just missed it- so it has to wait for it to be transmitted again). The TSync board doesn't declare sync until it knows this value after each boot-up.

This value can be programmed into the board after each boot-up, as desired (this value is not stored inside the receiver or inside the timing board). Note this particular value only changes if a leap second is asserted to UTC time (which typically only happens every couple of years).

The command used to set the offset value to the current value of 16 seconds, to allow sync to occur without waiting for it to be read from GPS is **./CS_SetTimeScaleOff 0 2 16**. To check if it is set correctly, run **./CS_GetTimeScaleOff 0 2**. It should report back "**16**". Sync can then occur!

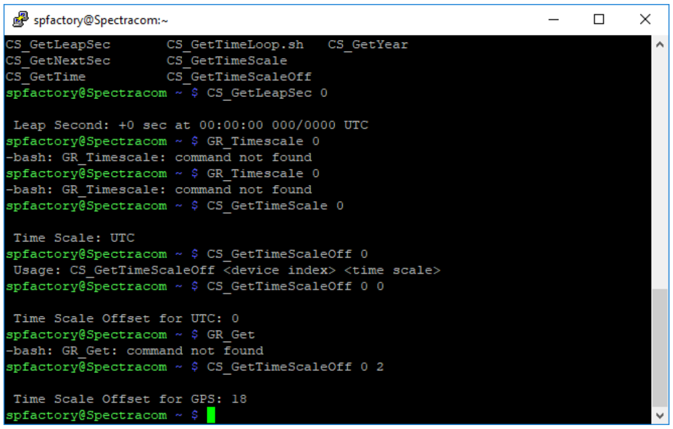
Where:

UTC=0 (power-up default)

TAI=1

GPS=2

Local=3



```
CS_GetLeapSec       CS_GetTimeLoop.sh   CS_GetYear
CS_GetNextSec       CS_GetTimeScale
CS_GetTime          CS_GetTimeScaleOff
spfactory@Spectracom ~ $ CS_GetLeapSec 0

 Leap Second: +0 sec at 00:00:00 000/0000 UTC
spfactory@Spectracom ~ $ GR_Timescale 0
-bash: GR_Timescale: command not found
spfactory@Spectracom ~ $ GR_Timescale 0
-bash: GR_Timescale: command not found
spfactory@Spectracom ~ $ CS_GetTimeScale 0

 Time Scale: UTC
spfactory@Spectracom ~ $ CS_GetTimeScaleOff 0
 Usage: CS_GetTimeScaleOff <device index> <time scale>
spfactory@Spectracom ~ $ CS_GetTimeScaleOff 0 0

 Time Scale Offset for UTC: 0
spfactory@Spectracom ~ $ GR_Get
-bash: GR_Get: command not found
spfactory@Spectracom ~ $ CS_GetTimeScaleOff 0 2

 Time Scale Offset for GPS: 18
spfactory@Spectracom ~ $ 
```

TSyncE-PCIe boards (using Acutime GPS antenna)

- Refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

TSIP

- TsyncE-PCIe boards use TSIP packets to communicate with the GPS receiver located inside the GPS antenna housing.

Calls for setting the TSync-PCI104 GPS receiver Mode/Dynamics code:

TSync-PCI104 command to switch between Stationary, Mobile or single sat modes:
GR_SetMode 0 0 “A” “B” (Where “A” defines the receiver mode and “B” defines the dynamics code, as listed below)

```
C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API>gr_getmode 0 0

  GR (0) Rcvr Mode: STANDARD (1) Dynamics: STATIONARY (3)
```

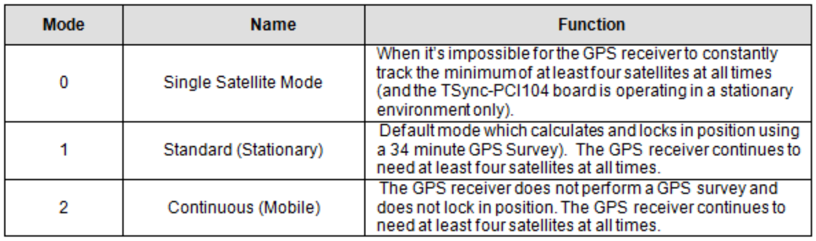
Example: GR_SetMode 0 0 2 2 (sets the Mobile Mode for Airborne use**)**

“A” values
0= Single Sat mode
1= Standard (Stationary) mode
2= Continuous (Mobile) mode
3= Averaging
4= Time-only

“B” values
0= Land
1= Sea
2= Air
3= Stationary
4= Unknown
5= Standby
6= Self

Note: Modes 3, 4, 5 and 6 are supported by the GPS receiver but are not used with the TSync-PCI104 boards. They are for use in SecureSync, which uses the same GPS receiver and similar software.

There are seven receiver mode available for the GPS receiver, though four of which are not used with the TSyncI-PCIe timing boards. The modes that are used with the TSync boards are as follows:



Mode	Name	Function
0	Single Satellite Mode	When it's impossible for the GPS receiver to constantly track the minimum of at least four satellites at all times (and the TSync-PCI104 board is operating in a stationary environment only).
1	Standard (Stationary)	Default mode which calculates and locks in position using a 34 minute GPS Survey). The GPS receiver continues to need at least four satellites at all times.
2	Continuous (Mobile)	The GPS receiver does not perform a GPS survey and does not lock in position. The GPS receiver continues to need at least four satellites at all times.

- ➢ For more info on the Standard, Mobile and Single Sat modes for a Res-T receiver, refer to: Mobile/Stationary/Single Satellite modes with a Resolution-T receiver (NetClock, TSync and SecureSync)
- ➢ This GPS Mode configuration persists in the GPS receiver through power cycles and reboots. It does not need to be issued each time the timing board starts-up.

(5/17/11) Known issue exists with TSyncE-PCIe boards when used with Acutime antenna (at least firmware versions 2.1.0 and 2.0.0- maybe newer if not resolved in the next update – Version 2.00 added the mobile mode, so earlier versions not applicable).

Like SecureSync, the default setting for the TSync-PCI104's Dynamics code is "Land". Also like SecureSync, when the GPS mode is set to Standard (Stationary mode) and after the GPS survey has been completed, the dynamics code is supposed to automatically switch to "Stationary". An issue with the TSync-PCI104 board's firmware is causing the dynamics code to continue being reported as "Land", even after the survey has completed. This appears to be a reporting issue only, as it acts like it it's in stationary mode.

GPS Mobile Mode

- ➢ Requires at least four satellites be tracked at all times.
- ➢ Must requires UT1 correction bit before going into sync (transmitted by sats every 13.5 minutes)
- ➢ GPS survey is not performed while in Mobile mode.

GR_SetMode 0 0 "A" "B" (Where "A" defines the receiver mode and "B" defines the dynamics code, as listed below)

GR_SetMode 0 0 2 B

Examples:
GR_SetMode 0 0 2 0 (sets the Mobile Mode for land use**)**

GR_SetMode 0 0 2 2 (sets the Mobile Mode for Airborne use**)**

"A" value	"B" values
2= Continuous (Mobile) mode	0 = Land
	1 = Sea

2 = Air
3 = Stationary
4 = Unknown

Note: Modes 3, 4, 5 and 6 are supported by the GPS receiver but are not used with the TSync-PCI104 boards. They are for use in SecureSync, which uses the same GPS receiver and similar software.

GPS Mobile mode with TSyncE-PCIe timing boards (GPS receiver located in the Acutime antenna- not on the board)

- Refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

1/03/11 KW- According to Denis Reilly and Paul Myers, mobile mode has not been tested with the TSyncE-PCIe boards and is not currently supported (Currently, mobile mode is only supported with TSyncI-PCIe boards). It is not known at this time whether the Acutime antennas will operate with the same command structure as the Resolution-T receivers that are used in the TSyncI-PCIe boards. And even if the commands are compatible, it's not known if the Acutime will operate normally while moving.

Email from Dave Sohn
Normal stationary mode requires 4 satellites to start a position survey, which is ~30 minutes. There is a single satellite mode, but it still requires a position survey, which is ~30 minutes. Based on what you're looking to do, you will want to run the GPS receiver in mobile mode, which doesn't require a position survey. However, mobile mode will be slightly less accurate than stationary mode.

To enter **mobile mode**, you need to set the GPS reference to continuous mode (mode = 2). We also recommend then changing the dynamics mode to land (dynamics = 0). With example programs, it would be the following:
GR_SetMode 0 0 2
GR_SetDyn 0 0 0

GPS single satellite mo

- Must still complete the GPS survey (requires tracking at least four satellite for 34 minutes) before the minimum requirement drops to only one satellite to prevent going into holdover mode
- Once the GPS survey has been completed, min satellites then drops to just one.

GR_SetMode 0 0 "A" "B"   (Where "A" defines the receiver mode and "B" defines the dynamics code, as listed below)

Example: GR_SetMode 0 0 0 0 (sets the Single sat Mode for land use)

A" values	"B" values
0= Single Sat mode	0= Land
1= Standard (Stationary) mode	1= Sea
2= Continuous (Mobile) mode	2= Air
3= Averaging	3= Stationary
4= Time-only	4= Unknown
5= Standby	
6= Self	

Note: Modes 3, 4, 5 and 6 are supported by the GPS receiver but are not used with the TSync-PCI104 boards. They are for use in SecureSync, which uses the same GPS receiver and similar software.

In single satellite mode, it is not enough just to manually enter the position information (lat and long). Each time the board is started-up, it has to be commanded to single sat mode and must be able to track at least four satellites in order to obtain the UT1 correction factor before it will go into initial sync (about 13 minutes or so). However, after that, unlike the standard mode, it only needs one satellite to maintain sync (instead of the normal four satellites).

Email from Dave Sohn:

My initial testing seems to show that in order to get initial sync, we need to acquire 4 or more satellites and download the almanac info. Once that has happened, we can drop down and sync with a single satellite. No survey is necessary to be completed for single satellite mode.

Satellite mode will have to be set each time the board is restarted. Setting an accurate position should help the timing and should be stored across power cycles.

Email from Dave Sohn
Normal stationary mode requires 4 satellites to start a position survey, which is ~30 minutes. There is a single satellite mode, but it still requires a position survey, which is ~30 minutes. Based on what you're looking to do, you will want to run the GPS receiver in mobile mode, which doesn't require a position survey. However, mobile mode will be slightly less accurate than stationary mode.

To enter single sat mode, you need to set the GPS reference to continuous mode (mode = 2). We also recommend then changing the dynamics mode to land (dynamics = 0). With example programs, it would be the following:
GR_SetMode 0 0 2
GR_SetDyn 0 0 0

Email Keith sent to Steve Rodgers (8/31/12)
I was wondering if the TSync-PCI104 timing board had declared sync yet (SS_GetSync 0 should report "TRUE")? It may have, by now.

We normally only recommend the single satellite mode be used when absolutely necessary, so we don't deal with it very often. One thing that I had forgotten about this seldom used mode is the initial requirement that the TSync-PCI104 board must still be able to track at least four satellites at the beginning. Then, once sync has been declared)sync is true) , this minimum requirement is then relaxed to just one satellite tracked from that point forward (unless the timing board is relocated somewhere else). Normally, it requires at least four satellites be tracked all the time to remain in sync.

The issue with not initially requiring four satellites is that without a good 3D fix being calculated (latitude/longitude and antenna height), timing from a GPS receiver without this positional information can be very incorrect. The GPS receiver needs a good position to know where it's at, relative to the satellites in order to provide good timing information. It takes a minimum of four satellites to achieve this 3D fix.

In the single satellite mode, the GPS survey does still need to be initially performed before sync status goes true. The total (cumulative) time for a survey to be completed is about 34 minutes, as long as the receiver can continue to track at least four satellites. When tracking at least four satellites continuously, the survey will continue to progress and will then complete about 34 minutes after tracking at least four satellites. Each time the survey is interrupted by dropping below four satellites, the survey halts until the receiver is tracking at least four satellites again. So if it's often dropping to less than four satellites, it will take much longer for the GPS survey to complete.

The GPS survey percentage complete can be read using the **GR_GetSurveyProg 0 0** API call or example program. The closer the GPS survey is to 100%, the closer it is to sync status going true.

Antenna cable delay/1PPS offset.

For LMR-400 or equivalent antenna cable, the antenna cable delay is 1.2ns/100 feet of antenna cable. Multiply total cable distance x 1.2 and the result is the cable delay.

Enter the calculated delay as a negative value (this moves the PPS input back, in order to account for the forward advancing that occurs as it goes through the antenna cable).

Use the **TSync-GR_SetOffset** to enter the desired cable delay / PPS offset.

GPS PDOP/HDOP/VDOP/TDOP FOR TSync-PCI104

PDOP/HDOP/VDOP/TDOP

The parameters provided by GR_GetFixData are provided by the GPS receiver itself. Depending on the type of fixes being done, and the mode of the receiver, we may only receive data for some of those variables, and zeroes for the others. We don't receive FOM and TFOM from the receiver currently, so those will always be zero. Below are generic definitions:

HDOP (Horizontal Dilution Of Precision) is a measure of how well the positions of the satellites, used to generate the Latitude and Longitude solutions, are arranged. PDOP less than 4 gives the best accuracy, between 4 and 8 gives acceptable accuracy and greater than 8 gives unacceptable poor accuracy. Higher HDOP values can be caused if the satellites are at high elevations.

VDOP (Vertical Dilution Of Precision) is a measure of how well the positions of the satellites, used to generate the vertical component of a solution, are arranged. Higher VDOP values mean less certainty in the solutions and can be caused if the satellites are at low elevations.

TDOP (Time Dilution Of Precision) is a measure of how the satellite geometry is affecting the ability of the GPS receiver to determine time.

PDOP (Positional Dilution Of Precision) is a measure of overall uncertainty in a GPS position solution with TDOP not included in the estimated uncertainty. The best PDOP (lowest value) would occur with one satellite directly overhead and three others evenly spaced about the horizon.
The most common issue we see when customers use an external 1PPS input ("EPP0") as the 1PPS input reference is due to ringing of the signal because of improper termination. The input impendence of the 1PPS signal is high impedance. However, the 1PPS source may desire a 50 ohm termination into the TSync-PCI104 -PCIe board (instead of high impedance). If the source desires a 50 ohm load termination, the input 1PPS may likely "ring" inside of the TSync-PCI104 board. The ringing of the 1PPS signal prevents a "crisp" point at which the TSync-PCI104 board can trigger on. With ringing of the signal present, the detected trigger point may coincidentally be at the right point, but it could also trigger too early or too late. So, with each 1PPS input when the signal is ringing, it may trigger at a different point on the signal each time. This prevents the disciplining circuit from being able to receive a stable 1PPS that is can accurately discipline to, and so the TFOM values will also be much higher than expected.

We have seen a few other similar cases where an external 1PPS input was causing a jittery 1PPS input because of improper termination of the signal at the input of the board. These were all resolved with a 50 ohm input being applied. To see this affect, if you have access to an oscilloscope, input the 1PPS signal from the source with a high input impedance setting on the scope. Look at the rising edge. You will likely see a ringing of the signal with no defined rising edge. Then, switch the scope to 50 ohm termination and you will likely notice the signal a become a crisp 1PPS with a very defined rising edge. This is what the board is also likely receiving and seeing, as well. To prevent the ringing of the 1PPS input from occurring and affecting the TSync-PCI104 -PCIe board, terminate the 1PPS input from the source with a 50 ohm load resistor. This will often "clean up the signal" so that it can be a better reference into the board.

Let me know if adding the 50 ohm resistor from the input to ground allows the TFOM values to significantly decrease (TFOM values of 3 or 4 would be expected). For your information, the TFOM value is errecalculated/updated about every 20 seconds or so.

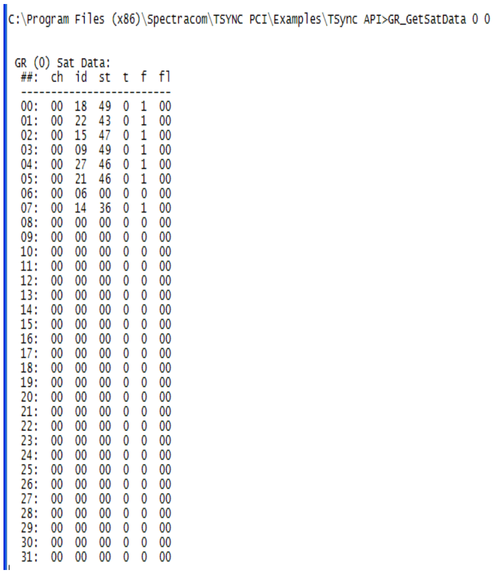
****GPS reception issues with TSync boards**

For more info, refer to: GPS\GPS reception Ap Notes\TSync-PCI104_timing cards

How to verify if the GPS receiver is tracking any satellites

A) Use GR_GetSatData 0 0 **<enter> call/example program**

This is an API Call/example program for number of satellites being tracked (and signal strengths of those sats) This call returns a table with 32 rows (00 through 31 labeled as ##) (note- only the first 12 rows are used at this time). The "ST" column is the strengths for each satellite. The total number of rows where the ST is not 00 is the number of satellites currently being tracked.



```
C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>GR_GetSatData 0 0


 GR (0) Sat Data:
  ##:  ch  id  st  t  f  fl
  -------------------------
  00:  00  18  49  0  1  00
  01:  00  22  43  0  1  00
  02:  00  15  47  0  1  00
  03:  00  09  49  0  1  00
  04:  00  27  46  0  1  00
  05:  00  21  46  0  1  00
  06:  00  06  00  0  0  00
  07:  00  14  36  0  1  00
  08:  00  00  00  0  0  00
  09:  00  00  00  0  0  00
  10:  00  00  00  0  0  00
  11:  00  00  00  0  0  00
  12:  00  00  00  0  0  00
  13:  00  00  00  0  0  00
  14:  00  00  00  0  0  00
  15:  00  00  00  0  0  00
  16:  00  00  00  0  0  00
  17:  00  00  00  0  0  00
  18:  00  00  00  0  0  00
  19:  00  00  00  0  0  00
  20:  00  00  00  0  0  00
  21:  00  00  00  0  0  00
  22:  00  00  00  0  0  00
  23:  00  00  00  0  0  00
  24:  00  00  00  0  0  00
  25:  00  00  00  0  0  00
  26:  00  00  00  0  0  00
  27:  00  00  00  0  0  00
  28:  00  00  00  0  0  00
  29:  00  00  00  0  0  00
  30:  00  00  00  0  0  00
  31:  00  00  00  0  0  00
```

Breakdown of the Satellite Data table (Trimble Res-T/Res-SMT-GG and ublox M8T receivers)

ch: (aka "**chnum**") receiver channel Number

Id: (aka "**SVN**" - Space Vehicle Number) Space Vehicle Number / ID Number of each satellite being tracked

Distinguishing between GPS satellites and Glonass satellites, when the Glonass Option is available (ID field)

- ➢ The Satellite ID number will indicate if that particular satellite is a GPS satellite or a Glonass satellite.
- ➢ GPS satellites have a Satellite ID Number of 0 to 59
- ➢ Glonass satellites have a Satellite ID Number of 60 and above

Note about ID number: Refer to info further below about a condition which causes **6 digit ID numbers** and "**out of range**" to be reported (in summary- ublox receiver installed, custoemer's application software originally written for Trimble receiver and not yet modified to work with a ublox M8T receiver),

st: signal strength of the satellite being tracked (reported in dB/Hz, as provided by the receiver).

t: TRAIM (aka "**bTraim**") Is this satellite accepted by the receiver's TRAIM algorithm? (always reported as "00" with RES-T, RES-SMT and RES-SMT-GG receivers).

UBlox M8T receiver: **from Paul M (3 May 17)** The TRAIM is not used at this time, in some receivers when set. it indicates the algorithm is active.

f: Fix status (aka "**bInfix**") Is the receiver using this satellite in its positional fix?

- "**0**" if not in fix,
- "**1**" if satellite is being used in fix)

fl: **Flags** reported by the receiver (always reported as rere "00" with RES-T, RES-SMT and RES-SMT-GG receivers. Used only with SAASM receivers)

UBlox M8T receiver: **from Paul M (3 May 17)**
The Flag value indicates:

= ABC

A: left most value indicates tracking state of receiver (per u-blox spec, 7 is best)

B: Middle value indicates frequency. **For u-blox it is L1 or 0.**

C: Right value is a 4 bit mask with 0 the right most bit and indicates if satellite is used in PPS and/or time solution.
Bit 0: Time 0=Yes, 1=NO
Bit 1: PPS 0= Yes, 1=NO

SO 103 from the satellite.means for a satellite 1 – not tracking, Frequency 0 means L1, 4bit flag of 3 means PPS and Time NOT used

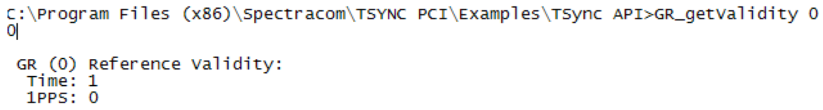
B) By looking at the Status LEDs on the edge of the board

(refer to: Status LED's/Status LEDs upon reboot for more info)

C) Using the SS_GetSync and SS_GetHoldver API calls/example programs to determine current sync status

The current sync status is also available via a couple of API calls/example programs. The **SS_GetSync** API call indicates True when TSync-PCI104 is synced or in holdover mode, or false when not in sync. To differentiate between fully Synced mode and Holdover mode, the **SS_GetHoldover** API responds with True if in Holdver mode only. It will return a False if the Board is in full Sync or if not in sync.

D) Verify GPS input validity using the GR_Getvalidity API call/example program (shown below)



```
C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>GR_getValidity 0
0

 GR (0) Reference Validity:
  Time: 1
  1PPS: 0
```

 As long as at least four satellites are being tracked, the **GR_GetValidity** API call/Example program response will indicate a "1" for both Time and 1PPS. If the receiver is not tracking at least four satellites, GPS is not valid, and both responses will be a "0" instead of a "1".

Driver issue associated with ublox receiver- Satellite ID numbers reported out of valid range (invalid 6 digit

numbers instead of two digits)

- Refer to Salesforce case 24614
- Due to using a pre-version 3.2.0 driver with TSync board having a ublox receiver onboard (driver version 3.2.0 added support for thwe ublox receiver) OR due to customer's application software using "GR" calls not being written for just Trimble receiver and not updated yet to work with both Trimble and uBlox recevers.

Customer Report: We have seen a few TSync cards reporting satellite IDs that are outside the valid range. Our software reports it with the following warning statement:

```
W0302 09:34:58.248358 495051 spectracom.cc:650] Satellite ID 393471 out of range!
W0302 09:35:01.273039 495051 spectracom.cc:650] Satellite ID 393471 out of range!
W0302 09:35:04.298158 495051 spectracom.cc:650] Satellite ID 393471 out of range!
W0302 09:35:07.323003 495051 spectracom.cc:650] Satellite ID 393471 out of range!
W0302 09:35:10.354477 495051 spectracom.cc:650] Satellite ID 393471 out of range!
W0302 09:35:13.382165 495051 spectracom.cc:650] Satellite ID 393471 out of range!
W0302 09:35:16.412534 495051 spectracom.cc:650] Satellite ID 393471 out of range!
```

Cause: Using an earlier version TSync driver (prior to versions 3.2.0) that was intended only for a Trimble receiver, with a newer TSync board having a ublox receiver onboard (Serial Numbers 4601 and above).

Email from Keith (10 Apr 17) You should only see this condition occur if using an earlier version TSync driver (versions prior to version 3.2.0 which added support for the ublox receiver) with a more recent TSync board which has a ublox receiver onboard (instead of a Trimble receiver- TSync boards with Serial Numbers 4601 and above, which started shipping March, 2016).

Fix: They need to update their "**GR**" access calls to include the additonal "**Else**" code for the ublox receiver which we added to the end of the example programs in the newer versions of the Linux driver

Note: (attach the example program of a GR call - such as "**GR_GetRcvInfo.c**" for example, from an earlier version of the Linux driver and also the same file from a more recent version of the linux driver (so customers can see the difference between the two files).

Email Keith sent to Morgan (4 May 17) This customer's application software was originally written to communicate with a Res-SMT-GG or Res-T receiver onboard the TSync board. Now that the boards have a ublox receiver installed, the responses from this receiver are formatted differently than when a Trimble receiver responds. So the "GR" calls are responding with ublox data which is formatted for a Trimble response (like trying to copy dots from a square into the same location of a circle).

As we needed to update our GR calls to accommodate both Trimble and ublox receivers, customers need to make similar changes to their application software to reflect the newer formatting of the responses to these calls (if they use our example GR programs in the latest TSync drivers, 0 instead of their application software- they will see valid responses).

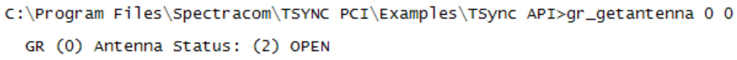
Below (and attached) is the email I sent to Chris Schrier at Google on this same topic. The same info applies to anyone with earlier application code and now using a newer TSync board with a ublox receiver (instead of a Trimble receiver):

Steps to troubleshoot GPS reception issues:

1. Send the GPS reception troubleshooting guide (discusses the below items)
2. Verify one instance of GPS is present (if not, GPS input not available on the board)

   Gr_GetNumInst 0. **Response should indicate GR Number of instances 1.**
3. Look at how many sats are being tracked

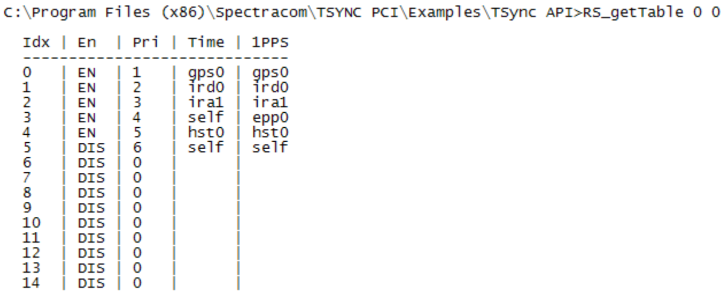
   GR_GetSatData 0 0 <enter> (example is shown just prior to these steps).
4. The **GR_GetSatData 0 0** API call/example program reports the number of satellites currently being tracked. If this value is less than 4, GPS reception is not qualified.
5. Verify Antenna Problem alarm not asserted: **GR_GetAntenna 0 0**



```
C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API>gr_getantenna 0 0

  GR (0) Antenna Status: (2) OPEN
```

The **GR_GetAntenna** API call/example program can be used to determine if a loss of GPS reception is due to an open or short being detected in the Antenna cable. The antenna status will respond with “Open” or “short” if a cable issue has been detected.

- Verify “**gps0**” is listed as an input in the Input Reference Priority table using the RS_GetTable 0 2 API call/example program (shown below):



```
C:\Program Files (x86)\Spectracom\TSYNC PCI\Examples\TSync API>RS_getTable 0 0

  Idx | En  | Pri | Time | 1PPS
  ----------------------------
  0   | EN  | 1   | gps0 | gps0
  1   | EN  | 2   | ird0 | ird0
  2   | EN  | 3   | ira1 | ira1
  3   | EN  | 4   | self | epp0
  4   | EN  | 5   | hst0 | hst0
  5   | DIS | 6   | self | self
  6   | DIS | 0   |      |
  7   | DIS | 0   |      |
  8   | DIS | 0   |      |
  9   | DIS | 0   |      |
  10  | DIS | 0   |      |
  11  | DIS | 0   |      |
  12  | DIS | 0   |      |
  13  | DIS | 0   |      |
  14  | DIS | 0   |      |
```

Make sure a row of the table lists “gps0” in both the Time and 1PPS columns, and that row indicates “EN” (Enabled) in the “EN” column

- If its tracking at least 4 satellites, look at the GPS survey progression status

GR_GetSurveyProg 0 0 <enter>. 50% is about 17 minutes remaining before Sync occurs.

Specific GPS reception scenarios:

A) GPS receiver is tracking at least 4 satellites, but green Sync LED not lit

If this happens, verify the input reference priority has a row (index) where “GPS 0” is the Time and PPS reference and make sure this row is enabled. If it’s disabled or the entry has been removed from the table, the board can’t sync to GPS, even if it’s tracking at least four satellites.

To verify GPS is still an enabled reference in the saved (User) table (or the default table, if the saved table doesn’t exist because a working table hasn’t ever been saved), first perform a **RS_GetTable 0 0** (shows the default factory table configuration). Then perform an **RS_GetTable 0 1** (shows the default User table configuration).

Make sure both tables contain a row that has “GPS 0” in both the Time and 1PPS columns. If it does, make sure this row is set to Enabled. If this row is either disabled or does not exist, even if **GR_GetValidity** is true, the TSync-PCI104 board can’t sync to GPS input.

Verify GPS survey- it may still be in progress- **GR_GetSurveyProg 0 0** <enter>. 50% is about 17 minutes remaining before Sync occurs.

R) GPS receiver is tracking at least 4 satellites, but green and yellow LEDs are both lit (indicating it’s in Holdover mode).

TSync is in Holdover mode.

- Verify that if the board is moving, The TSync-PCI104 been reconfigured for Mobile mode. If not, the movement of the platform will cause the receiver to go into Holdover mode.

S) GPS receiver is not syncing to Spectracom GSG GPS simulator

Default GSG output power level is too low. GSG defaults to -125dBm output power, while the Res-T receivers need -115dBm output power for satellite acquisition.

**DAGR GPS receiver ("Defense Advanced GPS Receiver" interface)

- Refer to "**Model 1204-02"** in the Option Card information document: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\SecureSync\Option Cards

- Refer to the TSync-PCIe section of this document for more info (same as the TSync-PCIe)

**IRIG input synchronization

- Refer to the TSync-PCIe section of this document for more info (same as the TSync-PCIe)

****Setting the year value upon each TSync-PCI104 power-up (IRIG input, for instance)

- Refer to the Tsync-PCIe section of this document

**"external 1PPS Input Reference ("epp 0")

- Refer to the TSync-PCIe section of this document for more info (same as the TSync-PCIe)

**UT1 correction factor (leap seconds/TAI offet)

- Refer to the TSync-PCIe section of this document for more info (same as the TSync-PCIe)

Oscillators / Osc disciplining to PPS ref / TFOM / Holdover

- Refer to the TSync-PCIe section of this document for more info (same as the TSync-PCIe)

** Oscillator calibration / HR (Hertz range)

- Calibrated once at the factory and then stored in EEPROM (not erased with a Clean).
- A patch is required to reset HR back to "0", so that another oscillator calibration can be automatically performed the next time a reference is applied.
- *LS_Message 0* command will display the HR value when its performed, but when its read out with this commabd or the board is power cycled/rebooted, there is no way to retrieve it again (there is no API call for this value).

** 1PPS and 10MHz alignment

Q. How closely are the 10MHZ and the 1PPS slaved together?
A. Except during power-up, there are always 10 million cycles of 10MHz between each 1PPS.

Additional response from Denis Reilly (8 Oct 201) "Further info (probably more than the customer needs at the moment):"

Except during power-up, there are always 10 million cycles of 10MHz between each 1PPS. After a reboot or power cycle, when the TSync syncs to a reference for the first time, the 1PPS and 10MHz are not guaranteed to be locked together during tracking setup.

When the Disciplining State transitions to "Locked", we lock the 1PPS and 10MHz together, and guarantee that there will be 10 million clock cycles between PPS's as long as the Disciplining State remains "Locked". (We can keep both the 10 MHz and 1PPS synchronized to the reference in this state without slipping the signals relative to each other.)

If a unit loses all of its references, and the holdover timeout expires, and the unit enters the Freerun state for a long time, there is a chance that upon acquiring a reference again the relationship between the 10MHz and 1PPS may drift again. But once the disciplining state becomes "Locked" again, the 10 MHZ and 1PPS will be locked together again, and there will be exactly 10 million 10MHz cycles in between PPS's.

***Oscillator disciplining

Firmware version changes associated with disciplining

A) firmware version 3.4.7 ("Oscillator disciplining improvements")

- (ECO 1625 for 3.4.7 released 29 March 2018)

- The command to get the current D/A Value is **XO_GetDac 0**

1PPS disciplining, when an TCXO oscillator is installed

DAC values (the command to get the D/A Value is **XO_GetDac 0)**

Min D/A: 0000
Max D/A: 0xFFFF

TCXO D/A vdc steering range: 0-3vdc

Valid D/A range : 0x2AAA to 0xD555

Oscillator alarm is asserted when D/A is: +/- 10% (0.2vdc) of the valid D/A range

1PPS disciplining, when an OCXO oscillator is installed

DAC values:

Min D/A: 0x0000
Max D/A: 0xFFFF

OCXO D/A Vdc steering range: 0-5vdc

HR (Hertz range) is calculated once at the factory and then stored in EEPROM (not erased with a Clean).

Time it takes for OCXO oscillator to lock

- TFOM exceeding MaxTFOM is 10 times faster slew rate than if MaxTFOM is not exceeded.
- it only stays at the increased slew rate while MaxTFOM is exceeded. As the System PPS is pulled in, TFOM will continue to decrease. It may eventually fall below MaxTFOM, at which point is slows down to the slower slew rate until its back into alignment.
- Also keep in mind that oscillator disciplining is dependent upon a very stable input PPS. If there is excessive jitter in the reference PPS, oscillator disciplining will be trying to keep up with the jitter. This will inherently help dampen out the input jitter so the System PPS is not as jittery as the reference. But the TFOM may not be able to ever decrease, because it can never actually be able to become aligned with the Reference PPS.

Email after talking to Mark Goodlein
After a cold-start, The OCXO oscillators take about 4 minutes to warm-up. A quick-phase oscillator adjustment then occurs as soon as the 1PPS reference has been detected and is considered valid. This quick phase-align adjustment takes about a minute to complete. Then, it takes around 2 minutes for the oscillator to be disciplined "right on".

Total oscillator alignment process time, after a cold start (assuming the input reference is connected at power-up) is about 6 to 7 minutes from power-up.

For the TSync with an OCXO oscillator, the 1PPS is always slewed during re-alignment.
The rate at which it is slewed depends upon how the "Maximum TFOM" setting has been configured. This setting allows the user to define how accurate the 1PPS must be in order for the box to report that it is "In Sync".

- If the current TFOM value is less than or equal to the Max TFOM setting, then the frequency of the oscillator is offset by a maximum of 0.4 Hz (40 ns/s) until the 1PPS is realigned.
- If the current TFOM value is more than the Max TFOM setting, then the frequency of the oscillator is offset by a maximum of 4 Hz (400 ns/s) until the 1PPS is realigned.

As you can see by these slew rates, it may take a long time to re-align the 1PPS if it is very far off. However, there are always 10 million cycles between each 1PPS (except during power-up).

Keep in mind that it only stays at the increased slew rate while MaxTFOM is exceeded. As the System PPS is pulled in, TFOM will continue to decrease. It may eventually fall below MaxTFOM, at which point is slows down to the slower slew rate until its back into alignment.

80 microseconds of phase error is 80,000 nanoseconds. So if TFOM is not exceeding MaxTFOM, at 40ns/second of slew rate, it will take 2000 seconds (about 30 minutes) for the System PPS to be aligned. However, if TFOM exceeds MAXTFOM, it will speed up to 400ns/second (until TFOM no longer exceeds MaxTFOM, which causes it to slow back down to 40ns/second). Assuming TFOM exceeds MaxTFOM the entire time until it was back into alignment, at 400ns/second, it will take 200 seconds for it to be realigned.

Also keep in mind that oscillator disciplining is dependent upon a very stable input PPS. If there is excessive jitter in the reference PPS, oscillator disciplining will be trying to keep up with the jitter. This will inherently help dampen out the input jitter so the System PPS is not as jittery as the reference. But the TFOM may not be able to ever decrease, because it can never actually be able to become aligned with the Reference PPS. So the TFOM will remain a much higher number than usual.

Using MAX TFOM to go into Holdover Mode/ cause oscillator coarse adjust to occur again

sides resetting just the timing board itself in order for the oscillator coarse adjustment to automatically occur again (if the TFOM is excessively high), you can also take advantage of the "Max TFOM" value, if you want. If the estimated TFOM value ever exceeds the configured "Max TFOM" value, the coarse adjustment will occur again, until the TFOM value no longer exceeds Max TFOM.

The default Max TFOM is a value of "15". With Max TFOM still set to the highest possible value of "15", TFOM is never exceeded. However, you can lower the Max TFOM, so that if estimated timing errors exceed the desired values, the oscillator adjustment will occur much faster, instead of it being slewed very slowly. As the 1PPS is brought it, the TFOM value will decrease., Then, when its much closer, the TFOM will be within its set limits, and it will then switch from coarse adjust to software fine-tune slewing mode.

The Max TFOM value is set using **SS_SetMaxTfom**.

Example: The Max TFOM is changed to "5". If the estimated TFOM is 6 or greater, the TSync will go into Holdover mode and a coarse adjustment occurs, until TFOM becomes 5. Then, it switches back to slewing mode, until TFOM becomes the lower possible value.

- Exceeding MaxTFOM is 10 times faster slew rate than if MaxTFOM is not exceeded.

Holdover mode/ Oscillator free-run stability

Important note: Drift is non-linear

Email from Dave Sohn (17 Oct 2013) Drift in holdover is not linear as implied here. The phase drift will be exponential as the frequency drifts. So, no oscillator will provide <1ms per day indefinitely. For example, the TSync OCXO will provide less than 1ms per day for around 6-7 days, but more than that afterwards. Also, all of our models are based on good starting values starting out from GPS synchronization, which won't be available here.

Email from Tony Diflorio to a potential customer looking for 1ms/day accuracies A couple of other ideas to help you: You could use a portable (battery operated) GPS synchronization device to provide re-synchronization to the TSync-PCI104 -Opt-OCXO card, such as our model GPS-12/12R (see attached). This re-synchronization would have to be done every few days though to maintain the TSync-PCI104 -OCXO to <1ms per day of drift. Or, you might be able to use our popular "SecureSync", rack mounted GPS Synchronization system (see attached) with a Rubidium internal oscillator. This depends on whether you must have specific features that only a Bus-Level card can provide. Using our SecureSync with a Rubidium internal oscillator will provide the holdover accuracy you are looking for, over a much longer period of time (almost indefinitely).

Holdover mode operation

Email Keith sent (19 Dec 2019) In case you weren't already aware, TSync boards have an automatic allotted amount of time they will remain "in sync" after losing its IRIG (or GPS) input. This allotted amount of time is called "Holdover mode", and has a boot-up default timeout of 7200 seconds (2 hours).

Holdover applies to the TSync board syncing to a reference (IRIG or GPS)after initial power-up, but then the input reference is lost anytime thereafter (the board cant boot-up into Holdover mode, without first syncing to an input reference. Then it can go into Holdover mode again). The moment the input is lost, Holdover mode begins and the timeout counter starts. While in Holdover mode, "Sync" status remains true, with the System Time and System 1PPS being maintained by the stability of its now free-running oscillator.

After all input references are lost, the on-board oscillator will start to very slowly drift, with disciplining compensation no longer being applied to the oscillator. So, the System Time and System 1PPS being maintained by the oscillator will also inherently very slowly drift. The type of oscillator (TCXO vs OCXO) installed on the board generally determines how quickly it will drift (an OCXO will drift less than a TCXO) and the longer the TSync remains in Holdover/oscillator free run mode with no input references present, the more the Time/PPS will have drifted since the refence was first lost.

Holdover mode (Sync status true) continues from the moment no references are available, until one of the following occurs:

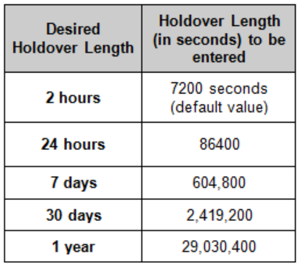
1) A reference (IRIG or GPS) is restored again, before Holdover mode expires. Board remain in Sync and Holdover state changes from true to false

2) The allotted Holdover period (2 hours after each power-up or user configurable for 1 second – up to 5 years, as configured in "seconds") ends with no references being restored. The oscillator will remain in free-run and Sync status changes to False.

3) The TSync board is rebooted. If the input signal isn't available when it powers back up again, the board remains not in sync (Sync status false) until IRIG or GPS becomes available again.

The Holdover period can be changed from two hours as desired, after each boot-up, using the following API call/Example Program: **SS_SetHoldoverTo 0 x (**where x represents the number of seconds of allotted time it can remain in Holdover mode upon loss of all input references

.



Desired Holdover Length	Holdover Length (in seconds) to be entered
2 hours	7200 seconds (default value)
24 hours	86400
7 days	604,800
30 days	2,419,200
1 year	29,030,400

For instance. Hanging the Holdover period to 1 second will cause the TSync board to declare Sync is False, just one second after the IRIG signal has been lost (instead of first waiting two hours to declare loss if sync).

To differentiate between "full" Sync status (external reference still present/valid) and in Sync (because its currently in Holdover mode with the reference having been lost), query both Sync status and Holdover status:

If Sync is true and Holdover is false = a reference is present/valid (board is in "full" Sync to its IRIG source)
If Sync is true and Holdover is true- the board is in sync, because its currently in Holdover mode (the oscillator is no longer being disciplined to IRIG and the oscillator is maintaining the TSync's Time/1PPS

- **SS_GetSync*:*** Get the current sync state.
- ***SS_GetHoldover:*** Get the current holdover state.

FAQs about the TSync specs

Q The user manual says that the GPS output will be within +/- 50 ns of the Internal 1 PPS source. If it's true that the 1 PPS output will also be +/- 50 ns within of the Internal 1 PPS source, then that would mean that the 1 PPS output will be within +/- 100 ns of the GP output. Are the variations of GP output and 1 PPS output with respect to the Internal 1 IPP source constant and do they track with each other? If so, would it be possible to use the 5 ns offset capability on each of the outputs to reduce the time difference between them to something less than +/- 100 ns?
A The spec Rich is referring to is +/-50ns for the 1PPS and GPO to the reference source (GPS, IRIG, etc.), not the internal 1PPS of the TSync. The GPO and 1PPS should be within +/- 10ns of each other.

TCXO/OCXO comparison chart

The following is a link to a chart which shows accuracy when locked and free run mode:
I:\Engineering\Products\ProductPerformanceSpecs5.xls

Default Holdover period is 7200 seconds (2 hours)

API calls to get current Sync and Holdover states:
TSYNC_SS_GetSync*:* Get the current sync state.
TSYNC_SS_GetHoldover: Get the current holdover state.
TSYNC_SS_GetHoldoverTO: Get the current holdover timeout.
TSYNC_SS_SetHoldoverTO: Set the current holdover timeout.
SYNC_SS_GetFreeRun: Get the current freerun state.

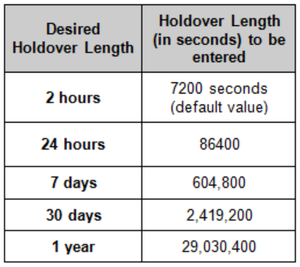
Oscillator free-run associated API calls

TSYNC_SS_GetFreeRun: Get the current freerun state.

The holdover mode begins when all input references are considered invalid and ends when either at least one input is considered valid or until the holdover period expires.

The length of holdover is set using the **TSYNC_SS_SetHoldoverTO** API call (the current value is read using TSYNC_SS_GetHoldoverTO. The value is configured in seconds with a valid range of 1 second up to 5 years. The default holdover period is 2 hours.

When a valid input is connected, the board is in sync, and not in the holdover mode. Once the input is lost, for a certain duration, the timing board will go into the holdover mode of operation. During this interval (default Holdover is 2 hours), the time is being derived from the board's oscillator and so it also remains in the sync mode in addition to the holdover mode being active. If the external reference is restored before the holdover period expires, the board remains in sync, but holdover is turned off. If two hours elapses with the reference not returning, both sync and holdover are then removed.



Desired Holdover Length	Holdover Length (in seconds) to be entered
2 hours	7200 seconds (default value)
24 hours	86400
7 days	604,800
30 days	2,419,200
1 year	29,030,400

FAQs

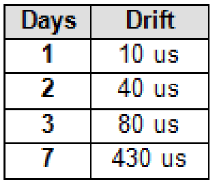
Q Why is the default holdover value 2 hours

A As for 7200 seconds of default holdover- this equates to 2 hours of holdover, which just happened to be the same value as the "worst-case" NetClock (With TCXO osc). Because the default Holdover value is not linked to the type of on-board oscillator, they went with the default of the NetClock/TCXO being 2 hours of holdover. This value is customer configurable for up to 5 years.

Oscillator stability with reference removed

1PPS output holdover specs with OCXO oscillator

Email from Dave Sohn (10/3/11) regarding low phase noise drift (no GPS) after at least two weeks of lock:
Estimate for holdover at constant temperature after 2 weeks of GPS lock



Days	Drift
1	10 us
2	40 us
3	80 us
7	430 us

1PPS output holdover specs with TCXO oscillator
Email from Tim Tetreault to Will Hickey:
The Holdover spec's for the TCXO is the same between the TSync and SecureSync:

Standard TCXO:
450usec/24hr (constant temp after 2 weeks of GPS lock)

The Holdover spec's for the OCXO is not the same because they use different OCXO's. The spec for the TSync is:

Optional OCXO:
90usec/24hr (constant temp after 2 weeks of GPS lock)

So if the customer wanted to calculate the drift per seconds:
The TCXO would be:
45E-5/(60sec x 60min x 24 hours) = 5.2 ppb

The error for the OCXO would be:
9E-5/(60sec x 60min x 24 hours) = 1.0 ppb

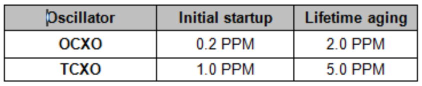
Disciplined to Timecode: 2 x 10–7
Undisciplined Freewheel Drift: 1 x 10–6

Oscillator stability when the board is in holdover mode:

- TCXO our oscillator stability is 1E-6 (1 ppm),
- OCXO our oscillator stability is 5E-8 (50 ppb).

There are two main error factors for oscillator stability: initial startup + aging over time.

The initial startup and lifetime aging specs are as follows:



Oscillator	Initial startup	Lifetime aging
OCXO	0.2 PPM	2.0 PPM
TCXO	1.0 PPM	5.0 PPM

With an external input reference (such as GPS and IRIG), we can account for these errors.

To calculate the estimated amount of Freewheel drift with a TCXO:
Convert the time to seconds and multiply by 1.5E-6.

Examples (where 1 hr = 3600sec):
Amount of drift per hour = 3600 * 1.5E-6 = 0.0054 seconds per hour (or **5.4 milliseconds**) of estimated drift

Amount of drift per second= 0.0054sec / 3600 (seconds) = .0000001 seconds per second (or **0.1 microseconds)** of estimated drift

Amount of drift per day= 0.0054sec x 24 (hours) = .0123 seconds per day (or **12 milliseconds**) of estimated drift

To calculate the estimated amount of Freewheel drift with an OCXO:
Convert the time to seconds and multiply by 5E-8.

Examples (where 1 hr = 3600sec):
Amount of drift per hour = 3600 * 5E-8= 0.00018 seconds per hour (or **0.18 milliseconds**) of estimated drift

Amount of drift per second= 0.00018sec / 3600 (seconds) = 0.00000005 seconds per second (**or 0.05 microseconds)** of estimated drift

Amount of drift per day= 0.00018 sec x 24 (hours) = 0.00432 seconds per day (or **4.32 milliseconds**) of estimated drift

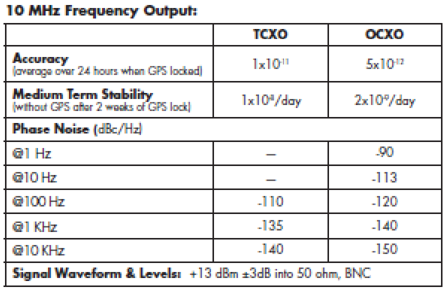
“1PPS not in Specification” / “1PPS Restored to Specification” log entries

These are events that occur and alarm entries that are asserted. “1PPS not in Specification” occurs if the current TFOM values exceed the user-configurable max TFOM value.   When the TFOM falls below Max TFOM, the Restored to Specification event/alarm entry is asserted.

Regarding the “1PPS Not in Specification” alarm, this alarm only occurs if the current TFOM exceeds the user-configurable max TFOM value. The SecureSync going into Holdover mode will not initially and automatically cause the TFOM to increase and exceed the max TFOM value. However, if the unit continues to operate for an extended period of time without any valid input references, the oscillator will begin to drift. As long as the max TFOM is not set to 15, eventually, the max TFOM will be exceeded and the alarm is asserted. The type of oscillator (OCXO or Rubidium) and the setting of max TFOM will determine the typical amount of time in holdover required to exceed the max TFOM.

Outputs

**10MHz output



10 MHz Frequency Output:

	TCXO	OCXO
Accuracy (average over 24 hours when GPS locked)	$1x10^{-11}$	$5x10^{-12}$
Medium Term Stability (without GPS after 2 weeks of GPS lock)	$1x10^{-8}$/day	$2x10^{-9}$/day
Phase Noise (dBc/Hz)		
@1 Hz	–	-90
@10 Hz	–	-113
@100 Hz	-110	-120
@1 KHz	-135	-140
@10 KHz	-140	-150
Signal Waveform & Levels: +13 dBm ±3dB into 50 ohm, BNC		

AM output only (no 10 MHz DCLS output available)

Output level

- Output level is not adjustable
- +13 dBm +/-3dB into 50 ohm, BNC   (10dBm=2.00vpp. 16dBm=3.99vpp)
- 5Vp-p nominal into high impedance, +/-3dB

Output impedance: 50 ohms

Output Amplitude 12 dBm (2.5Vp-p), nominal into 50 ohm, +/-3dB,
5Vp-p nominal into high impedance, +/-3dB,

By default, The 10MHz output is enabled. With no external input, the signal will be derived from the free-running internal oscillator. When the board is synchronized to itself, the oscillator is not disciplined. In order to be disciplined, the board needs an external 1PPS reference, such as GPS or external 1PPS input. This output configuration is limited to disabling the output when the board is not synced (this is known as Signature Control") or disabled altogether (no output, whether or not the board is synced).

The calls to reconfigure when the 10 MHz is not present are listed in the driver guide as "FP" calls /commands. If you search the TSYNC driver guide for this value, it will show you all of the available configuration calls that can be issued to reconfigure the 10MHz output as desired.

Note: Futher below is additional information on Signature Control.

Phase alignment of 10 MHz outputs from multiple TSync-PCI104 boards or SecureSyncs

Because KTS disciplining performs phase alignment of the 10 MHz output, unlike earlier Netclocks which only perform Frequency adjustments, if more than one TSync or SecureSync are synced to the same IRIG or GPS signal, the 10 MHZ outputs will be in phase with each other (not just corrected for frequency).

Harmonics and Spurs of the 10 MHz output

Refer to the TSync data sheet for specs on harmonics. ..\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards

10MHz Harmonics and spurs
For plots of the 10 MHz harmonics and spurs with an OCXO installed, refer to the email from Tom Richardson (8/2/12) in the following folder: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync-PCI104 \10MHz output

10 MHz output Signature Control

Email KW sent to a customer:
The TSync-PCI104 timing board has a feature that can be enabled, which squelches the 10 MHz output, when either the TSync-PCI104 board's Sync state is not "true" OR when the TSync-PCI104 has no valid input references (depending on how this mode is configured). This mode is called "Signature Control".

Signature Control can be enabled each time the TSync-PCI104 board is powered-up (it is disabled by default) using either an API call or an example program. Attached for your reference is a copy of the TSync-PCI104 driver guide (in case you don't already have a copy of it). Please refer to Section 4.2.9 (starting on page 4-153) for a list of API calls/example programs that are associated with the 10MHz output.

Specifically, the API call/example program to enable Signature Control for the 10MHz output is **FP_SetSigCtrl** (as discussed in Section 4.2.9.2 of the attached driver guide), where "SIG_CTL":

0: No Signature Control – 10 MHz always present

1: Signature Control based on SYNC status (10 MHz output is only outputted when the TSync board is either in the Sync or Holdover mode).

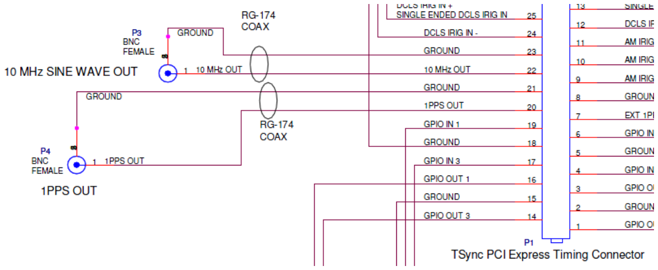
2: Signature Control based on Reference // status (10 MHz output is only outputted when the TSync board has a valid input reference present).

3: Ultimate Signature Control - always OFF (10 MHz never outputted)

FYI- The call to read the current status of Signature Control is **FP_GetSigCtrl**.

10MHz output pin-outs

- ➢ Not available on Basic/Standard Breakout cable
- ➢ Using Premium Breakout cable (BNC Connector P3)



10MHZ Signal	Connector P3 - pin	Goes to Pin on Timing connector
10MHz	Center pin	22
Ground	Shield	23

**1PPS output

Note: The 1PPS/ "Heartbeat" output is always 1PPS, but the pulse width of the output can be edited. For true heartbeat output (ability to change the frequency) refer to GPO outputs.

1PPS output impedance: The 1PPS output is high impedance (The 1PPS input is also high input impedance)

1PPS output Specs from manual:

- 1PPS output at timing interface connector
- 1Hz Pulse, Rising Edge or Falling Edge Active (selectable) o 40ns - 900ms Active Pulse Width (selectable, 200ms default)
- +0.55V VOL, +2.2V VOH

Two methods to output 1PPS

1. Dedicated 1PPS output is on pin 20 of the Timing connector.
2. Configure one or more of the GPO pins

API calls for 1PPS/Heartbeat output: The 1PPS output calls for TSync-PCI104 are the "**TSync _ PP"** calls. (Refer to the TSync family driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec

Available configuration API calls for 1PPS output on pin 20

- ➢ Get/Set Signature Control
- ➢ Get Frequency (1Hz)
- ➢ Get/Set offset (account for cable delays to external device)
- ➢ Get/Set the active edge (defines which edge is aligned to the on-time point)

- Get/Set the pulse width of the 1PPS signal
- Get the number of instances (usually just one on the TSync board)

Per the TSync-PCI104 user manual:
The TSync-PCI104 board generates a digital 1PPS output from the internal 1PPS of the system. Several parameters of the 1PPS can be controlled. The active edge can be set to either rising or falling edge, the pulse width can be adjusted, and an offset can be applied to adjust its relationship to the internal system, 1PPS from -500 msec to +500 msec in 5 nsec increments. The 1PPS output provides signature control similar to the IRIG outputs.

The TSync-PCI104 timing board has legacy TPRO API calls available to configure the "Heartbeat" output (this is the 1PPS output on pin 20 of the timing connector). Unlike the TPRO/TSAT boards, TSync itself does not have a dedicated output called "Heartbeat". This output was replaced by just a "1PPS output" instead. So the legacy heartbeat calls just configure this 1PPS output pin. There are no "TSync Heartbeat" API calls available in the TSync-PCI104 drivers (just the legacy "TPRO heartbeat" calls. The 1PPS output calls for TSync-PCI104 are the TSync-PCI104 _ **PP** calls. (Refer to the TSync-PCI104 driver guide).

1PPS output holdover specs with OCXO oscillator
Email from Dave Sohn (10/3/11) regarding low phase noise drift (no GPS) after at least two weeks of lock:
Estimate for holdover at constant temperature after 2 weeks of GPS lock

Days	Drift
1	10 us
2	40 us
3	80 us
7	430 us

**IRIG output

IRIG output defaults

- Refer to the TSync-PCIe section of this document for more info (same as the TSync-PCIe)

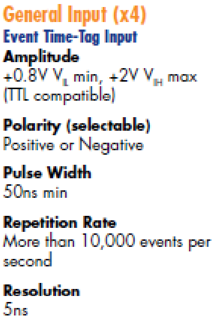
Log/Messages

LS_Get Message command

- If this command returns "**Error: opt err (RC_QUERE_ ERR).**" the log is empty. It needs an input reference present for log entries to be asserted.

GPIO (General Purpose Input Output) pins

**GPI pins (General Purpose Inputs- "GPI")



General Input (x4)

Event Time-Tag Input

Amplitude
+0.8V V_{IL} min, +2V V_{IH} max
(TTL compatible)

Polarity (selectable)
Positive or Negative

Pulse Width
50ns min

Repetition Rate
More than 10,000 events per second

Resolution
5ns

- Four GPI pins are available (GPI 0 through GPI 3) on the Timing connector (in order - pins 6, 19, 4, 17)

Note: The standard breakout cable only provides connection to one GPI pin (the first GPI pin - GPI 0). The premium breakout cable is required for use if two or more GPI pins are needed.

- Function: GPI pins can be used for:

Time-tag/Event input (interrupts can also be generated when an event occurs).

- Accuracy: The General I/O output signals timing are accurate relative to the Input reference's 1PPS signal to within +/- 50 nsec.

Input impedance of GPI pins: Per Mark McGregor: The GPI pins (General Purpose Input) are high impedance input (but if the source has 50 ohm output impedance, we can handle it – just add a 50 ohm load on an external BNC T connector to prevent reflections.

GPI API associated calls:

GPI inputs are configured with the "**Tsync_GI_"** commands. Refer to the TSync-PCI104 driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121 for more info on associated API calls.

- GI_GetEdge and GI_SetEdge are for defining which edge is active
- GI_GetValue gets the current input value
- GI_GetTSEnable and GI_SetTSEnable gets/sets the timestamp enable state when time stamping input changes.
- GI_GetNumInst gets the number of GPI pins present in the system

Q. Can we individually compensate the propagation delays to each GI input channel?
A. There are no delay commands for the GPI input pins

**Desire to have an event on GPI (input) pin generate timestamps (known as timetag)

- Refer to: TSync-PCI104 time tag (timestamping/Time Stamping/Timetag/Time Tag)/External Event Input

---

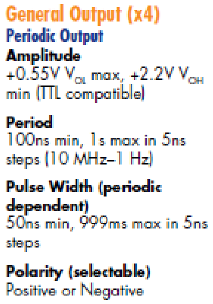
Time tag (timestamping/Time Stamping/Timetag/Time Tag)/External Event Input

Refer to timestamping in the TSync-PCIe Section of this document/

**GPO pins (General Purpose Outputs- "GPO")

- Four GPO pins are available (GPO 0 through GPO 3) on the Timing connector (in order- pins 3, 16, 1, 14).

Note: The standard breakout cable only provides connection to the first two GPO pins (GPO 0 and GPO 1). The premium breakout cable is required if three or all four GPO's are needed.



General Output (x4)

Periodic Output

Amplitude

+0.55V V_{OL} max, +2.2V V_{OH} min (TTL compatible)

Period

100ns min, 1s max in 5ns steps (10 MHz–1 Hz)

Pulse Width (periodic dependent)

50ns min, 999ms max in 5ns steps

Polarity (selectable)

Positive or Negative

Output impedance of GPO pins: The GPO pins (General Purpose Output) are high impedance.

GPO API calls:
GPO pins are configured with the "**Tsync_GO_"** commands. Refer to the TSync-PCI104 driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121 for more info on associated API calls.

- GPO pins are enabled with TSYNC_GO_SetEnable API call.
- Output mode is configured with the TSYNC_GO_SetMode API call.
- TSYNC_GO_GetValue: Gets the GPOs current output value.
- Can enable Signature Control of GPO pins (TSYNC_GO_SetSigCtrl)

Specs: The GPO outputs and the 1PPS output (on-time point) should be within +/- 10ns of each other.

Function: Each GPO pin can be configured to (as described in more detail further below):

Operate in Direct Value mode (signal level changed at user's discretion).

In **Direct mode**, an event is triggered when the output Value in the GPIO output control / status register is changed and creates the active edge selected by the GPIO direct mode output interrupt active edge bit in that same register. This can be used to generate a "software" interrupt by setting the GPIO output appropriately.

Output a Square wave signal (1Hz to 10 MHz) (not sine wave).

In **square wave** mode, an interrupt is generated whenever the GPIO output generates the active edge as selected by the GPIO output square wave active edge bit in the GPIO output control / status register. This can be used to generate a periodic interrupt at the rate of the square wave.

Operate as Match Time Event pin (become active at one time and inactive at another time).

In **match time** mode, an interrupt is generated whenever the GPIO output high match time or GPIO output low match time registers are enabled and subsequently matched against the current system time.

Generate interrupts at a particular interval.

Q Is there a way to command an output pulse (on GPIO output) to start at a time of my choosing and have a frequency that I specify? I'd like to create a 10 msec pulse but I need to control when that pulse starts.

A (Dave Sohn, 8 Jan 2014) You can generate a single 10ms pulse to start at their time of choosing by setting up and enabling both a match high and match low with 10 msec in between.

Q. I am on Linux and when I search the full set of tsync directories that were created when I followed the installation intructions, I only find GO_VALUE_RECIPE in the tsync_Go.c file where it is an extern so it would need to actually be declared somewhere else. Perhaps GO_VALUE_RECIPE is in the driver though that seems to be rather unconventional driver code. If it is in the driver then it is not clear how to link to get access to that variable and your manuals have no instructions on how to link to the driver.

A. GO_VALUE_RECIPE is an extern in the tsync_Go.c file and defined in the tsync_Go_Services_recipes.c file by the line:

```
RECIPE(GO_VALUE)
```

The GO_VALUE fields are defined in the tsync_Go_Services.h file:

```
#define GO_VALUE_FIELDS                    \
        TSYNC_X(        uint32_t,   value)
```

Default state of the GPOs

Q. What is the default state of the General Purpose Outputs on power-up, before any calls are made?

A. All general purpose outputs default to low outputs at system startup.

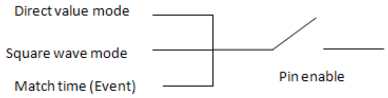
Ability to Start/Stop the GPO outputs

Q.Is there any way to start/stop a square wave output on a GPO pin at a given absolute time? The only thing I can think of is to use a match-time to generate an interrupt and then enable the GPO pin's output in response to the interrupt (assuming that the GPO is already set up for the square wave). Although, this method wastes a GPO pin just to generate the interrupt.

A. They are correct that there is no way currently to stop/start the GPO pin at a given absolute time. Their idea of using the match-time was the same idea we had. The only other think I could think of would be for them to use some external logic and use the match time pin to turn on and off the square wave.

GPO pin enable/disable

- GPO pin can be used to determine when the output signal will be present and when it's not present.
- The output is present as soon as the pin is enabled (whatever state the configured GPO signal happens to be in, at that particular moment.



FAQs about GPO pin enable (Email to Erik Johansson of NOAO 7/22/11)

Q. The pin is enabled and set in direct mode with an output value of 0 (low) and then disabled. Now I set the value to 1 but do not enable the pin. The output is still 0. At some later time I enable the pin. The output now changes from 0 to 1 as soon as the pin is enabled, correct?

A. Keith's reply- This is absolutely correct. In direct mode, with the pin disabled and with it changed from a 0 to a 1, the output won't change. But the output will change from a 0 to a 1 as soon as the pin is enabled.

Q. The pin is set in square wave mode and is enabled for a while so that the output pulse train is active. I now disable the pin. The output will be high or low depending on the state of the square wave at the time the pin was disabled. I leave the pin disabled for some time. Now I enable the pin. The output immediately changes to whatever state the square wave would have been in had the pin remained enabled the entire time.

A Keith's reply- Correct, in square wave mode and with the pin disabled, the 1PPS pulse train continues to be the same signal, but it's not present on the pin. When the signal is re-enabled, the output will go to the same place in the signal that it would be in had the pin not been disabled.

There is no "1PPS alignment change" of the signal when the pin is re-enabled. The 1PPS signal continues to occur, just like the pin had never been disabled to begin with. If the pin is re-enabled while the pulse is current active, part of the pulse width will be cut-off, so the pulse may have a smaller pulse-width than all of the others. If the signal is already a "high" when the pin is enabled, the output will start off as a "high" at that moment. This first active edge is not aligned to 1PPS.

In order to prevent this from occurring, we would recommend ensuring that the pin is not enabled at the on-time point itself. It should be enabled right after or anytime before the next on-time point occurs. The 1PPS interrupt could be used to let you know that the on-time point has just occurred, so it's now OK to now enable the pin without the signal being currently active and preventing an un-aligned edge from occurring.

3. The pin is enabled and set to direct mode and driven to an output value of 0 and then disabled. The output remains at 0. The mode is now changed to a square wave with a rising edge and some pulse configuration. When I enable the pin, the first pulse will go active on the next coincidence of the 1 PPS (this assumes no offset) and then continue according to the pulse configuration.

A Keith's reply- the pin enable /disable places no control on the signal as to the active edge being aligned to the on-time poin. If the mode is changed from direct mode to square wave mode with the pin disabled, the signal will start to be generated (the edge is aligned to the on-time point). When the pin is then enabled, the signal will be outputted as it is at that particular moment. The pin enable does not cause any delay in the signal so that the first active edge is aligned to the on-time point. If the pin is enabled anytime after one pulse width has occurred and before the next active edge occurs, only then will the first edge be aligned to the on-time point.

As described above, if it's desired to prevent any false edges from occurring when the pin is first enabled, the pin should not be enabled during the active pulse. The enabling of the pin needs to be delayed until the pulse has returned to its non-active state and before the next active edge occurs. Otherwise, enabling the pin during the active state will cause the "active edge" to occur at the moment that the pin is enabled- not necessarily at the on-time point.

Modes of GPO pin operation

There are three available modes of GPO pins (each listed/described below)

- Direct Value mode
- Square Wave mode

- Match time mode

Excerpt from the TSync user manual
*In **match time** mode, an interrupt is generated whenever the GPIO output high match time or GPIO output low match time registers are enabled and subsequently matched against the current system time. In **square wave** mode, an interrupt is generated whenever the GPIO output generates the active edge as selected by the GPIO output square wave active edge bit in the GPIO output control / status register. This can be used to generate a periodic interrupt at the rate of the square wave.*

T) Direct Value mode (DVM) (for GPO output pins)

Direct Value Mode allows the user to configure the GPO pin to be a high or a low.

In **Direct mode**, an event is triggered when the output Value in the GPIO output control / status register is changed and creates the active edge selected by the GPIO direct mode output interrupt active edge bit in that same register. This can be used to generate a “software” interrupt by setting the GPIO output appropriately.

GO API calls associated with direct value mode (DVM):

TSYNC_GO_SetDvmValue: Sets the GPOs direct value state
TSYNC_GO_GetDvmValue: Gets the GPOs direct value state

FAQs about DVM

Q. What is the Direct Value Mode for the GPO pins? It is not really described in the driver manual, except for the API description. Is this a way to use the GPO pins manually (i.e., setting directly by making driver calls)? The user manual does not describe direct value mode either, but discusses the possibility of configuring the GPOs as generic output pins (section 5.5.2). Is this the same as direct value mode?
A Direct value mode allows a user to directly set the value presented on a GPO pin. You can manually set an output pin as either high or low when in this mode. The "mode" of the GPO must be set to "direct value" mode and the GPO must be enabled for it to work.

Q. Is there a difference between TSYNC_GO_GetValue and TSYNC_GO_GetDvmValue? I am guessing there is and that TSYNC_GO_GetValue returns the current value of the specified GO pin (for any configured output mode), whereas TSYNC_GO_GetDvmValue returns the current "set" value for DVM mode. Is this correct
A. Get Value gets the current value of the GPO independent of the mode. Get DVM Value gets the currently set direct mode value.

Q: Will the GPIO pins output 5vdc. I need 5vdc for the application I am using the TSync board in. The manual says they are TTL compatible.
A: To answer your question about the TSync timing board outputs, the input and output levels are TTL compatible, as stated in the TSync manual. The allowable input voltage range is 0 to 5.5vdc with a logic high being determined by about 2 vdc up to 5.5vdc. The output voltage of a logic high is guaranteed to be at least 2vdc (into high impedance), but can be as high as about 3vdc. The TTL output levels are never at 5vdc.

If 5vdc is needed when the output is “active”, you may be able to somehow convert the levels from 2 to 5vdc external to the TSync board itself.

U) Square Wave mode

- GPO pins can be configured to output a continuous square wave.

In **square wave** mode, an interrupt is generated whenever the GPIO output generates the active edge as selected by the GPIO output square wave active edge bit in the GPIO output control / status register. This can be used to generate a periodic interrupt at the rate of the square wave.

“GO” API calls associated with Square wave output

TSYNC_GO_SetSquareWave: Sets square wave configuration

- ➢ Period is from 100 nsec to 1 sec. Pulse width is from 10 nsec to 999,999,990 nsec.

(Note: Offset, period, and duty cycle are in nanoseconds)

TSYNC_GO_GetSquareWave: Gets square wave configuration

Example: **Desire to generate a squarewave on GPO pin 0.**

To generate an interrupt at a desired interval, the GPIO Output Event Interrupt (type 8) should be used. One of the GPIO pins (such as GPIO pin 0) needs to be configured for 500ms square wave output (per your desired ½ second interrupt interval).

To configure the GPIO pin 0 for a 500msec square wave, 1 msec pulse width, positive-going pulse, use the following API calls (or example programs):

./GO_SetMode 0 0 2 (GPIO “0” set to square wave mode)
./GO_SetSqWave 0 0 0 5000000 1 1000000 1 (GPIO “0”, 0 offset, 500msec period, usec scale, 1msec pulse width, positive pulse)
GO_SetEnable 0 0 1 (GPIO “0” enabled)

Note: The square wave mode has a limit on the pulse width of 900ms (900 milliseconds). If it’s required to have a wider pulse width wider than 900ms, look at using Match time output mode instead of Square wave mode. This will require them to write their own software in order to constantly change the start and stop times.

**Desire to convert the GPO outputs to 1PPS outputs

- ➢ **Refer to the TSync family driver guide** (1219-5001-0050 in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002)

Note: Be aware that there are some considerations to take into account when performing this operation. The square wave output as currently designed is only aligned to the 1PPS once when configured. That means that during initial alignment of the board’s internal 1PPS to an input using HW smoothing, before the SW takes over with disciplining, the square wave outputs lose their alignment. Once SW takes over, which can be determined by looking at the disciplining state, if a square wave is setup as a 1PPS, it will always be aligned from that point on.

Email from Tim Tetreault to Insup to have 1PPS on GPO 0 (1/27/12)
Here is what you need to do:

- ➢ Setup Pin0 to a “square wave”:
- ➢ ./GO_SetMode 0 0 2
- ➢ Enable Pin0 output:
- ➢ ./GO_SetEnable 0 0 1

The default settings for the square wave output are 1PPS with a positive pulse width of 200msec.
If you want to change the frequency or pulse width, use the command:
“GO_SetSqWave A B C D E F G”
Where:
A = card index
B = I/O pin

C = offset from 1pps in nsec
D = period
E = scale for period 0 = nsec, 1 = usec
F = pulse width in nsec
= 0 is neg pulse, 1 is pos pulse

So for example, if you want GPIO “0” to be set up for a 1pps, 100msec pulse width, positive pulse:

```
./GO_SetSqWave 0 0 0 1000000 1 100000000 1
```

FAQs about GPO pins

- I am confused about the synchronization of a square wave pulse on a GPO pin to the 1 PPS and how that relates to when the GPO pin is first enabled. Let's say I program a square wave pulse and set the GPO mode for square wave, but leave the pin disabled. Later on I come back and enable the GPO pin at some arbitrary time "T". When does the first active edge of the square wave occur? Is it coincident with the next 1PPS transition (assuming no offset) or does it occur at the next time that will allow for an eventual synchronization with 1 PPS?

  I guess another way to ask the question is: if I program a square wave pulse, when does the active edge that is coincident with 1PPS occur relative to when I enable the output on the pin?

- When the GPO pin is disabled and then re-enabled sometime thereafter, it should return to the same state that it was in when it was last stopped. From a signal perspective, the next active edge will be coincident (like it was never disabled). However, to the system that its connected to, depending on the previous state (which will be based on what state the signal is in at the time the pin is disabled), it could look like an active edge occurred as soon as the pin was re-enabled.

As this starting state could be in either state, depending on when the pin is disabled, a safer state, instead of enabling/disabling the pin, would be to switch the GPO between square wave mode to Direct Value Mode (DVM) when its desired to stop the output. Instead of just stopping the signal in a potentially unknown state, the DVM mode allows you to be able to command the pin to be either a constant low or a constant high. This can switch the square wave to a constant low for example, so that when the GPO is switched back to square wave mode, you can be certain on what state it’s starting at. Then, it should transition only when it’s coincident.

Match Time mode

Refer to the TSync family driver guide for more info) (1219-5001-0050in Arena at: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002 (Section 5.3 gives an example configuration).

The General I/O is configurable as a Match Time Event pin, which will activate at a preset time and become inactive at another preset time. The Match Time Event provides two user settable times to make the General I/O pin active and inactive. The Match Time Event configured General I/O pin has a programmable edge, allowing the selection of “Low to High” or “High to Low”.

The following are commands utilized for match time:

GO_SetEnable: (General enable of a general purpose output)

GO_SetMode: (Set the general purpose output to match time mode (mode = 1))

GO_SetMatchEnable: (Enable/disable the time match of a general purpose output. Each output must be enabled for matching high level times and low level times.

HW_SetMatchTimeHi: (Set the time at which the specified general purpose output should go high. Time can be set up to 100 days ahead and as close as 50 msec.)

HW_SetMatchTimeLo: (Set the time at which the specified general purpose output should go low. Time can be set up to 100 days ahead and as close as 50 msec).

Note: The match times for lo and hi are in the format of: <GPO> <day> <hr> <min> <sec> <ns>

Desire to use Match Time and have the GPO pin start off as a high after each time the board is started (instead of the default low that the pin goes to when the mode is selected)

The default setting for the GPO pins at board start-up is a low, until the first setMatchTimeLo occurs. However, it may instead be desired to have the GPO pin remain a high until the first match time occurs (the opposite). The Direct Value Mode (DVM) cannot be temporarily used to direct the pin high because when the mode is switched back from DVM to Match Time, the pin will inherently go back to a low.

To set the GPO pin high before the first real match occurs, keep the GPO pin disabled while performing a setMatchTimeHi (without performing a setMatchTimeLo afterwards). This will cause the pin to transition from a low to a high. Then, enable the GPO pin. At that point, the GPO pin will remain high until the first setMatchTimeLo time occurs.

I just spoke to one of our timing board engineers regarding your first question on Match Time. The reason that using the DVM mode to set the pin to high is not keeping the pin high after switching to Match time mode is because the mode switching is acting like a three-way switch (DVM, Match Time or pulse out). Each of the three modes starts out as a low, so while it it's in DVM mode, it can be set to high, but as soon as the "switch" is changed to Match Time, the pin then inherently goes to a low again. This is the reason that temporarily using the DVM mode won't provide you with the desired operation.

FYI- In addition to the "three-way mode switch", there is also a second "pin enable/disable switch" as well. The pin enable/disable switch applies to all three modes. If the GPO pin is not enabled, the GPO pin itself won't change states, even if a match time occurs. Once the pin is enabled, the effects of the selected mode will then be able to be sensed by the equipment connected to the pin.

To use the Match time mode and have the GPO pin start off with a "masked" starting transition of default low to a high ("masking" the transition from the default low to a high prevents this particular transition from being detected as a "false" match time occurring). We recommend performing a two step process, in a specific order, right after the TSync-PCI104 board is started.

The GPO pins are disabled at each boot-up. While the GPO pin is still disabled, switch the pin to Match Time mode. Then, use the setMatchTimeHi to have the pin go high, even before the pin has been enabled (by not following it with a setMatchTimeLo it will then remain high until the setMatchTimeLo is eventually performed). Then, once this Match has occurred, just enable the GPO pin.

From that point, the pin will be a high until the match time sets it low for the first time. And since the pin was disabled while it was being set to a starting high, the equipment won't detect the "setup" as a Match.

FAQs about Match Time
Q**.** I noticed that after I use setMatchTimeHi or setMatchTimeLo, and then read the info back using getMatchTimeHi or getMatchTimeLo, the value for the **year** returned by the API call is always "0001". Is this
a bug?

A (From Dave Sohn) Match time does not use the year information. Time matches can only be set within one year of the current time. The year information read back is not relevant and should be ignored.

Q. Is it possible to submit an array of match times to be used to generate a series of one-shot pulses on a GPO?

A. **(Response from Tim Tetreault)** Our TSYNC board does not generate a series of one-shot pulses like they are asking for. The only thing we have that could do something like that is our new CTC (Chronos) board that we are making for Lockheed Martin.

Desire to generate two interrupts at two different frequencies (with the interrupts synchronous to the PPS reference).

From Dave Sohn- The customer would want to use the GPIO Square Wave Output. They can set one GPIO Output to generate interrupts at a 50Hz rate and one at a 128Hz rate. The only caveat is that the square wave output will only remain aligned once we're disciplining to a reference. If you start up the board and the GPIO Outputs, and then connect your reference, it won't remain synchronized with the 1PPS. This has to do with how the first time alignment of the board to the reference and the alignment of the GPIO Output Square Wave occur. I can provide more detail into that if necessary.

In summary:

- The customer can set up two GPO Output Square Waves running at 50Hz and 128Hz.
- These Outputs can be setup to generate interrupts.
- If the outputs are setup after the board reports it is in sync, they will forever be aligned with the 1PPS even if sync is lost.

FIRMWARE/SOFTWARE/DRIVERS (Windows Linux Solaris)

API	Area of the board (Hardware, supervisory software, etc)	Description
CS	Clock Service	Provides an abstract interface to the timing subsystem.
EC	LED Component	Drive the LED indicators.
FP	Frequency Output	Configures the output frequency.
FS	Flash Manager Service	Provides access to the images stored in all flash memory devices.
GI	General Purpose Input Component	Configure and monitor the general purpose input pins.
GO	General Purpose Output Component	Configure and monitor the general purpose output pins.
GR	GPS Reference Component	Execute the GPS receiver's protocol and determine 1PPS and time validity.
HA	Host Agent	Obtain the capabilities of the TSync.
HR	Host Reference	Uses information from the host PC to determine 1PPS and time validity.
HW	Hardware	Provide access to the direct HW accessible control/status of the timing subsystem.
IN	Initializer Service	Perform initial configuration and setup of each software module.
IP	IRIG Output	Generate and output IRIG streams.

	Component	
IR	IRIG Reference Component	Control and process decoded IRIG input streams to determine 1PPS and time validity.
LS	Log Service	Provides a queue for errors and maintains system alarms.
PP	PPS Output Component	Control a 1Hz output.
PR	PPS Reference Component	Monitor the 1PPS input reference.
PTR	PTP Reference Component	Control and process decoded PTP network packets (either as an input reference or a time output).
RS	Reference Monitor Service	Determine the best available input reference and maintain the reference priority table.
SS	Supervisor Service	Maintain the time source, 1PPS source, Sync and Holdover states of the system.
US	Upgrade Service	Upgrade the firmware and FPGA images in the external flash memories.
XO	Oscillator Component	Analyze frequency measurements and make corrective adjustments to the timing system oscillator.
XS	Oscillator Monitor Service	Measure and provide the accuracy and stability of the timing system oscillator.

Board handle and device index, index (Instance) Number

Example API call/example program syntax: **IR_SetMode <device index> <index> <mode>.**

The **device index** is an index into the number of TSync-PCI104 boards present in the system. The first board installed is assigned the device index value of “0”. Any additional installed boards are assigned an incremental device index.

The **board handle** is different from the **device index.** The **board handle** is a void pointer to an internal board information structure that is returned when the driver is opened by a user of a specific board.

The **index** is an “instance” number used in some calls to indicate either a potential or currently present similar input or output configuration. The TSync-PCI104 board was designed as a timing engine that other products could be built upon. So many inputs and outputs can have more than one of it. The first instance of that particular “feature” is an index of 0. Then, each duplicate input/output is incremented by one, to identify another specific similar “feature”.

For example, there is only one IRIG DCLS input on a TSync-PCI104 board, so its index is 0. Since it’s entirely possible for other products to have more than 1 IRIG DCLS input, the index number allows configuration for the multiple inputs of this reference. For the TSync-PCI104 boards specifically, most of the index values are going to be “0” since it’s the only one of that particular circuitry. By specifying a unique index number, even though there may only currently be one instance, this allows the ability to be easily able to add-on additional “features” with little change to the driver calls (provides flexibility).
The **mode** is used in some calls as a “variable” in order to select a particular configuration related to that particular API call.

Per your question, in each API/example program the first number after the name of the call is the device index. It indicates the command should be sent to the first TSync-PCI104 board installed. The first board is assigned a device index value of “0”. If there happened to be two or more TSync-PCI104 boards installed in the same machine, they would each be assigned an ascending device index. So if there is only one board installed, the first number will always be a "0".

A second number follows the device index in certain API calls/example programs. This is the instance number. The TSync-PCI104 board is designed to be a full timing engine that can be used in other products (such as the Spectracom SecureSync). Certain functions of the TSync-PCI104 board or other products can potential have more than one of the same function. An example of this is GPS input. Though there is only one available GPS input currently available on the TSync, it's always possible that someday, the TSync-PCI104 or another product based on the TSync board, could have two GPS inputs. The instance number specifies which of the specific instances of the same function the API call or example program is addressing. The first instance of a function is always assigned an instance of "0". In the case of GPS on the TSync-PCI104 board, the instance will be a 0.

Processor time used when performing GPS calls

Q. I've noticed that the TSync-PCI104 driver function GR_GetPosition(nInstance=0) uses a significant chunk of CPU time even when invoked only once per second; on a 2.33 GHz Intel Xeon, a single call occupies between 0.04 and 0.15 seconds of CPU time. Any idea why this is, and whether there's an equivalent but quicker API function for reading the position?

- ➢ The CPU time used during this call is not unexpected operation as there are several steps involved. When the host tries to read the GPS position from the Tsync board, it sends a query to the software running on the board. That software in turn makes a query to the GPS component of the board for that information to pass back to the host.

However, the information is not always immediately available for access. Since both the software and the host are looking at the same data set, access to it needs to be controlled. When the GPS component needs access to the data set to run the communications protocol to the GPS receiver, it puts a lock on the data set. That means if the data set is locked, then the host has to wait to access that data until the component is done with it. The times reported for accesses are not out of line with the time that the component holds the data set in order to receive and process messages using the protocol.

VxWorks or other custom drivers not supplied by Spectracom

- ➢ Spectracom does not supply a VxWorks driver.
- ➢ The Linux driver for Tsync board contains Source code that can be used as a reference to see how functions are performed.
- ➢ Refer to section 2 of the Application programmers manual for register Memory mapping.
- ➢ Refer to Section 8 of the Application programmers manual for info on performing different functions.
- ➢ Unlike legacy TPRO/TSAT boards, the TSync register reads need to be read through the FIFO. The registers for the FIFO buffer are defined under '**Shared Memory Interface**" in the App Programmers guide.

Reading/writing to registers directly "Direct Memory Access" using a Spectracom driver (Peek and Poke API calls)

Refer to the Application Programmers Manual for more information on direct memory access (1191-5002-0050): I:\New Released\Manuals\1191-xxxx-xxxx.   (**Note**: the first page of Section 2 provides the Memory Map of all of the registers).

Values from the TSync-PCIe boards can be directly read from or written directly to the board's registers, if desired.

"Peek" API call: Allows registers to be read directly
"Poke" API call: Allows registers to be written to directly (Caution: Make sure to write to the correct register. Otherwise bad things can happen)

The PC BIOS assigns a unique Base address for each PCIe bus card installed. Then, each Memory Register is assigned a unit Base address extension to identify each particular register.

The TSync Application Programmers manual provides a Memory Map for the memory locations (Base address extensions) which store specific contents. The customer can then create a custom driver to read the contents directly. Refer to the Application Programmers manual for more details I:\New Released\Manuals\1191-xxxx-xxxx (1191-5002-0050).

The source code supplied with the Linux driver can assist them in identifying the Base address that is assigned by Bios and shows how we access the registers.

Pages 2-1 and 2-2 of the Application Programmers manual (latest version is attached, just in case) provides the Memory Map (Base address extension) for all of the Memory Registers on the TSync-PCIe board.

I recommend you also refer to the source code that is provided with the Spectracom TSync-PCIe Linux driver. This source code will help you determine the unique Base address that is assigned by the PC BIOS to the installed TSync-PCIe board. The source code also provides good information that can be used to help create your custom driver for accessing these memory registers.

Tsync registers reads need to be read through the FIFO buffer

Email Keith sent based on input from Tim Tetreault Please be aware that the TSync family of timing boards are setup differently than the earlier, legacy TPRO/TSAT timing boards. Time can be read from a register but all API commands that don't start with a "**HW_**" pass through a FIFO buffer to send and receive commands with the TSync boards.

All commands will use the "TSYNC_SET", "TSYNC_GET" & "TSYNC_WAIT", including even the earlier, legacy "TPRO" API commands.

The registers for the FIFO buffer are defined under '**Shared Memory Interface**" in the App Programmers guide.

---

tsync.dll and tpro.dll files

- When the Tsync driver is installed, it generates two .dll files, tpro.dll and tsync.dll
- The tsync.dll file support all Tsync calls.
- The tpro.dll file allows customers upgrading from TSAT-PCI to be able to use the legacy commands from the Tpro/tsat-PCIU driver, without needing to rewrite their application software.
- They still have to install the Tsync driver and recompile their software to the new driver.

Q. We are upgrading some kit for a customer who use to use a TPro PMC board. The current software uses the supplied TPRO.dll for the API.

Now they have purchased a TSync-PCI104 and wish to upgrade the software, and as a consequence we downloaded the drivers and API supplied on your web site for the TSync-PCI104.

However we have noticed that the TSync-PCI104 driver have both a TSYNC.dll and the TPRO.dll included. Question is, can we use the TPRO.dll to control and communicate with the TSync-PCI104 card. Note we are not interested in any new features that the TSync-PCI104 card may provide at this time, therefore to minimise the porting effort we would happily use the TPRO.dll if this will work?

- (Keith 19 Mar 2014) Thanks for your email and question. I have some information for you.

The tpro.dll file generated when the TSync-PMC driver is installed is actually intended for the legacy TPRO/TSAT-PCI (not PMC) timing boards. However, almost all of the calls from the TPRO/TSAT–PCI and PMC boards (and therefore this tpro.dll file) are the same.

We are in the process of releasing a new TSync-cPCI/PMC board and its driver will also contain a tpro.dll file which is for all of the legacy TPRO/TSAT-PMC/cPCI calls. We are also in the process of releasing a new version of the TSync-PMC driver to support both TSync-PMC and TSync-PCI104.

Two options for you- We should be releasing the new TSync-PCI104 boards and this new combined driver very shortly. If you like, I can let you know when it's available. Or, you can also use the cPCI-PMC driver currently available on our website. To obtain this driver, please visit us at:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=22

Static and dynamic TSync.lib file (library file)
.lib files (Linker files)

What is a .lib file?
From: http://file.org/extension/lib

Files that contain the .lib file extension usually hold a static data library of information. The information in this file is associated with a specific computer application. The LIB files usually store the functions that are referenced by the application, which is then associated with the library.

LIB files may also contain objects like images, media and text clippings. LIB files are an alternative to DLL files, which contain dynamic link libraries. The LIB files contain static libraries that can be used as import libraries for DLL files, with the imported library files being marked with the .lib file suffix.

From Wikipedia: http://stackoverflow.com/questions/3250467/what-is-inside-lib-file-of-static-library-statically-linked-dynamic-library-an

Static library
In a static library, the .lib file contains all the actual object code and data for the functions provided by the library. In the shared version (what you referred to as statically linked dynamic library), there is just enough code to establish the dynamic linkage at runtime. The linker then identifies the bits it needs and puts them in the final executable.
If the code of the library is accessed during the build of the invoking program, then the library is called a static library. An alternative is to build the executable of the invoking program and distribute that, independently from the library implementation. The library behavior is connected after the executable has been invoked to be executed, either as part of the process of starting the execution, or in the middle of execution. In this case the library is called a dynamic library. A dynamic library can be loaded and linked as part of preparing a program for execution, by the linker. Alternatively, in the middle of execution, an application may explicitly request that a module be loaded.

Dynamic library
For a dynamic library, the .lib file contains a list of the exported functions and data elements from the library, and information about which DLL they came from. When the linker builds the final executable then if any of the functions or data elements from the library are used then the linker adds a reference to the DLL (causing it to be automatically loaded by Windows), and adds entries to the executable's import table so that a call to the function is redirected into that DLL.
You don't need a .lib file to use a dynamic library, but without one you cannot treat functions from the DLL as normal functions in your code. Instead you must manually call LoadLibrary to load the DLL (and FreeLibrary when you're done), and GetProcAddress to obtain the address of the function or data item in the DLL. You must then cast the returned address to an appropriate pointer-to-function in order to use it.

Refer to Salesforce case 11052
Q. I received the TSync card this week. One thing I notice is that there is no static library version of TSync.lib; all of our apps are statically linked, and I'm hoping you guys support a build of your API in this mode as well. All I see in the distribution is a dynamically linked version.

- As of at least July 2013, the TSync-PCI104 Linux driver does support both static and dyamic lib files. However, as of at least version 2.4.1 Windows driver does not support both static and dynamic .lib files.

Compiling Customer's application software

(Note: For Visual Studio info, refer to the Windows driver section further below)

Borland

- Our TSync.lib file was written for Visual C+ +. Customer desires to use Borland C + + instead:
- Q. The customer has begun to write his software application. However, he is using Borland C++ and Tsync.lib seems to be wrote for Visual C++. We have found a solution however we are not sure his approach. We used Tsync.dll and applied "implib" utility in order to have Tsync.lib compatible with Borland. It seems to work but we do not have the board to verify it. Is it the good approach? If it is not, can you give us the procedure? Or do you have a Tsync.lib compatible for Borland C++?
-
- A. Email from Tim T - The approach they are using is correct. Here is a link with more information on how to use Visual C++ DLLs with Boland C++ using the IMPLIB utility.

http://bcbjournal.org/articles/vol4/0012/Using_Visual_C_DLLs_with_CBuilder.htm?PHPSESSID=046f3a5a4298523cb724d9dcf777ec0f

C# (called "C Sharp", "Managed Code C #" or "Managed Code C Sharp")

2) Our TSync.lib file was written for Visual C+ +. Customer desires to use C# (C #) instead:

- (email from Jack Loui 3/17/11) I am trying to convert TSync C++ code to C# by using [DllImport("Tsync.dll"), CallingConvention = CallingConvertion.Cdecl)]. So far I have trouble to get this to work. Do you have any sample code wh A newer version of Microsoft Visual C++ 2010 ch is in C# to work with the TSync card?

- Keith's reply:

Unfortunately, we don't have any sample C# code that we can provide you with. We know that we have had one or two other customers that were able to use our dll files for the conversion, but we don't have any information on how they were able to make the conversion, nor have we done this ourselves.

Installing TSync-PCI104 drivers on an ARM platform (ARM processor)
Q. Is it difficult to port the driver to ARM platform?
A. Keith's response- At this time, we are not aware of any compatibility issues with the TSync-PCI104 Linux driver and an ARM platform. However, please keep in mind that a TSync-PCI104 board does not need to be installed in order to test the driver for compiling. The TSync-PCI104 drivers can be downloaded from the Spectracom website at no cost. After downloading the driver, your customer can then try installing and compiling the driver to see if there are any compatibility issues, even before they receive and install a TSync-PCI104 board.

Email from Dave Sohn (11/7/11)

I haven't heard of anyone using the TSync with an ARM processor. I don't know of any limitations with that. There may be some endian issues, but I don't know for sure. I'm assuming this is a Linux platform. They can download the drivers and try it. If there are endian issues, there is a flag in the driver for setting endian that could be tried.

**TROUBLESHOOTING:
****TSync-PCI104 board doesn't appear to be operational after install
Questions to Ask/Info to help troubleshoot (steps to troubleshoot further below)

- What is the Serial Number of the board (via the silver P/N sticker affixed to the board or with the LS_GetSerialNo API call/example program).
- Is this the only TSync-PMC board that you have? If not, have you tried another timing board in its place, yet?
- Have you tried reseating the board in the slot?
- Have you tried this TSync-PMC board in another computer, yet? If not, I recommend you try it in a different computer, to determine if there is an issue with the PC's PCIe slot.
- What are the LEDs on the edge of the board doing (Do they flash once at power-up? Are they flashing a pattern? Are they constantly on)?
- What about the FPGA LED… is that lit? Is the bootloader jumper in the correct position?
- TSync Board firmware version and driver versions.
- Is the board installed in Windows PC or Linux system?

Linux systems only
A. What is the specific linux distribution (such as Redhat, CentOS, Scientific linux, etc) and kernel version?

Notes:

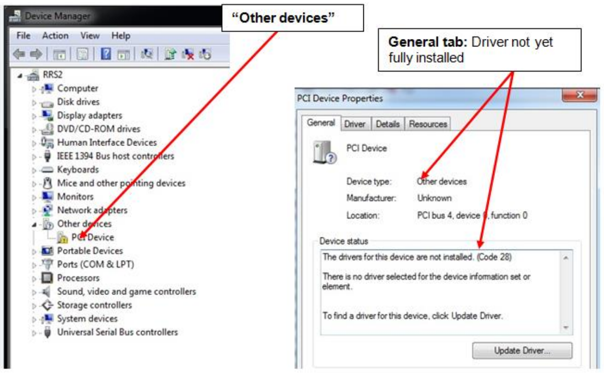
- ASPM may need to be disabled
- Updating the linux distribution to v6.4 or higher may help address issus with ASPM issues.
- Class code may need to be updated (especially if the board shipped before ~ April 2013).
- System BIOS may need to be updated

B. Has the earlier version 2.3.0/2.4.1 drivers ever been installed on this machine. If so, the Rules files need to be updated.

Windows PCs only
With the board installed in a Windows PC (whether or not the driver has been installed yet) "TSync-PCI" is it seen in "Device Manager" (Start-> All Programs -> System -> Hardware-> Device Manager).   Device Manager should list "Timing Boards" (with "TSync-PCI Timing board" below it).

Note: Before the Windows driver is fully installed (with the board installed), it may show up as simply "PCI device" (under "other devices"as shown below:

- With the board installed in a Windows PC (whether or not the driver has been installed yet) "TSync-PCI" should be seen in "Device Manager" (Start-> All Programs -> System -> Hardware-> Device Manager). Device Manager should list "Timing Boards"(with "TSync-PCI Timing board" below it).



"Other devices"

General tab: Driver not yet fully installed

PCI Device

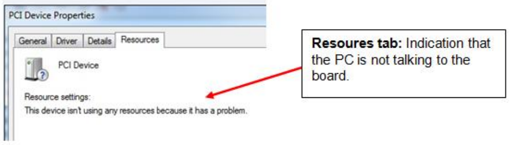
Device type: Other devices
Manufacturer: Unknown
Location: PCI bus 4, device 0, function 0

Device status

The drivers for this device are not installed. (Code 28)

There is no driver selected for the device information set or element.

To find a driver for this device, click Update Driver.



PCI Device

Resource settings:
This device isn't using any resources because it has a problem.

Resoures tab: Indication that the PC is not talking to the board.

- ➢ If displayed, click on Properties -> Details-> Hardware IDs and see what is listed (work with Tim Tetreault as this list may indicate the class code assigned to the board is causing issues with the particular PC).

- ➢ Try the board in another PC (preferably on a different Model motherboard) to see if theother PC can recongnize it's installed.

- ➢ Refer to Class Code update fix and (ASPM) Power Saver/Power Monitor feature

- ➢ Verify the PC BIOS settings for the PCie bus. Make sure the bus is not configured for "Video Only", ASPM (autopower save mode, USB) enabled, etc.

Email KW sent to a customer.
Have you tried going into "Device Manager" and running "Scan for hardware changes" from the "Action" menu?

Another idea is to verify that the system can see the board. In "Device Manager", select "Devices by connection" in the "View menu". Look under the "Microsoft ACPI-Compliant System->PCI bus" group for any unknown device. If you look at properties for that unknown device you can check to see if one of them is the board. The TSync board is Vendor ID "1AD7" and Device ID "8000".

Do you have any other cards installed? Are they visible in Device Manager? Do you have any other PCIe boards to verify that the system is properly seeing the PCIe card slots?

- Specific motherboard issues

ASUS motherboard
Salesforce Case 6738- The ASUS motherboard came with a utility called 'ASRock X-Fast USB' that was removed and the computer was re-booted.   Immediately, the installation program operated normally and the Tsync driver was installed successfully.

Checks to make:

- Check the Status LEDs on the edge and the green "FPGA load" LEDs on the back side of the TSync-PMC board:

- Verify all three Status LEDs (red, yellow and green) on the edge of the TSync board turn on and then go out (and remain not lit until an input reference becomes valid).

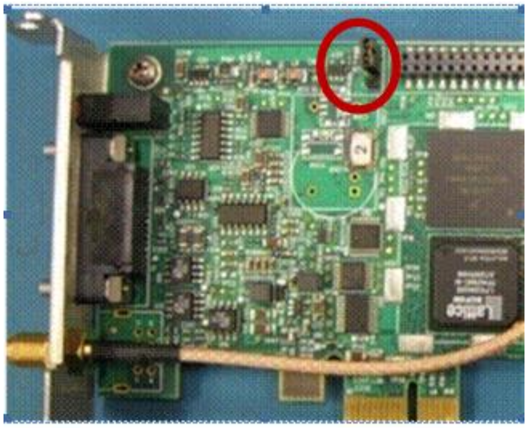
Right after the PC with the TSync-PMC board powers-up, the three LEDs on the edge of the TSync-PMC (the red, yellow and green LEDs), should momentarily illuminate and then turn off (until an input reference is connected). Do all of these LEDs momentarily turn on when the PC is first booted-up?

If the LEDs are not lit at power-up /all three constantly lit/exhibit the correct pattern:

- First, try to re-seat the TSync-PMC board in the system to make sure that all the contacts are solid.

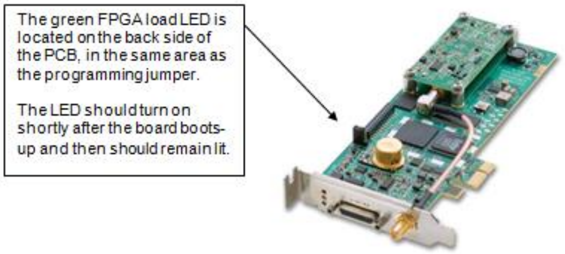
- If the green and red LEDs are blinking on your PCIe card, it is likely that the board is in the bootloader waiting for image downloads from the host. This could signify that either the board needs to be reprogrammed or that there was a failure in the flash forcing it to fail to load the run-time or backup default images, landing back into the bootloader.

Verify the green "FPGA" loaded LED is lit (verify Jumper J25 is in the correct position)
Check jumper setting on J25 and make sure that it is configured on the pins to the outside edge of the board, as seen in the red circle in the photo below:



If this jumper is in the wrong position, on a reboot the timing board will never look for the new images and will always load the bootloader (Green and Red LED s will be blinking and the Yellow LED will be not lit). Please place this two position jumper as shown in the picture above (with the jumper across the two outer pins (closest to the edge of the TSync-PMC board.

There is also a **green** ”load” LED on the back-side of the TSync-PMC boards (in the same area of the board as the jumper described above, but on the other side of the timing board. This LED should light shortly after the TSync-PMC board powers-up and then should remain lit, from that point forward. Please let me know if this LED turns on and then remains lit green while the PC is on.



The green FPGA load LED is located on the back side of the PCB, in the same area as the programming jumper.

The LED should turn on shortly after the board boots-up and then should remain lit.

If the green LED still doesn’t turn on and remain lit with the jumper in the correct position

- Verify +3.3 vdc is present from the PCIe bus. If this Dc voltage is low, the board may be stuck in constant reset state. Refer to: Power (+3.3vdc +12vdc and -12vdc from PCI bus)/logic.

- Try the board in another slot, or another system altogether and see if the gren LED remains lit.

Labview driver/Labview wrapper

- As of at least Jan 2014, the TSync-PMC boards do not have an available Labview driver /Labview wrapper. There are no plans at this time to develop one
- With the shared library (xxx.so) file provided with our driver, a customer can create their own wrapper routines using this provided file. For additional info on calling .so files in Labview, refer to http://zone.ni.com/reference/en-XX/help/371361J-01/lvhowto/import_Shared_library/

Note: The website above discusses how to import functions from a shared Library file (.so file , as provided in the TSync-PMC Windows driver) into Labview.

- The libtsync.so shared library file is generated during the driver build. Its located in the TSync/Linux/Lib directory.

For more information on wrappers, refer to websites such as:
http://digital.ni.com/public.nsf/allkb/06ECDC689DDA0F3D862574440074CD95

Problem:
What is a wrapper DLL and when do I need one?

Solution:
A wrapper is a piece of software that provides a compatibility layer to another piece of software. One is often necessary when developing LabVIEW applications because third-party DLLs were usually designed to be accessed from C (or

similar low-level languages) and not LabVIEW. Such a DLL may, for example, return pointers or complex data structures which LabVIEW cannot easily handle.

Writing a wrapper DLL can be compared to writing a completely separate program in C that accesses the original DLL in the way the original author intended. In turn, this wrapper program has been specifically designed to be accessed from LabVIEW. In this sense, the new C program "wraps" around the original C program (DLL) and provides a layer of compatibility. The benefit of a wrapper is that the source code for the original DLL is not necessary, as it does not need to be modified in any way.

Q. OK, this is a concern. Are you saying that there is no shared library (xxx.so) that I can use to access the functionality of the time card? That is all I would need to access the card from LabVIEW (I can create my own wrapper routines if I have a shared library).

- My response to this email was No need to fret! The libtsync.so shared library file is generated during the driver build. Its located in the TSync/Linux/Lib directory.

Reply Keith sent to a customer (5 Nov 2013)
To answer your question, there are no plans at this time to develop and release a Labview driver/wrapper for the TSync-PMC timing boards.

However, please note that we provide a **libtsync.so** shared library file with the TSync drivers which a customer can use to create their own wrapper routines.   This file is located on the **TSync/Library/Lib** directory. Please note that we do not provide any technical support on how to do this, but your customer is free to try this if they wish. They can download the TSync drivers from our website at no cost if they would like to look into this. The TSync-PMC drivers can be downloaded at:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=37.

NI LabWindows/CVI 9.0,
Q.Our customer has purchased our TSync-PMC timing board, but when they write its testing program based on NI LabWindows/CVI 9.0, (Windwos XP), they found the program can not be building. I was wondering could you please help advise for this issue.

A. (reply from Keith 27 Jan 2014)
For your information, Labview programs such as LabWindows require a wrapper to be used in conjunction with the Spectracom TSync-PMC Windows driver. Spectracom does not provide a Labview wrapper with the TSync-PMC board drivers, so customers that wish to use the TSync-PCI board with programs such as LabWindows will need to create their own custom wrapper, consisting in part of the API calls from the TSync-PMC board which they want to use with the LabWindows program.

However, please note that we provide a **libtsync.so** shared library file with the TSync-PMC drivers, which can be used to create custom wrapper routines. For more information on wrappers, refer to websites such as:
http://digital.ni.com/public.nsf/allkb/06ECDC689DDA0F3D862574440074CD95.

Problem:
What is a wrapper DLL and when do I need one?

Solution:
A wrapper is a piece of software that provides a compatibility layer to another piece of software. One is often necessary when developing LabVIEW applications because third-party DLLs were usually designed to be accessed from C (or

similar low-level languages) and not LabVIEW. Such a DLL may, for example, return pointers or complex data structures which LabVIEW cannot easily handle.

Writing a wrapper DLL can be compared to writing a completely separate program in C that accesses the original DLL in the way the original author intended. In turn, this wrapper program has been specifically designed to be accessed from LabVIEW. In this sense, the new C program "wraps" around the original C program (DLL) and provides a layer of compatibility. The benefit of a wrapper is that the source code for the original DLL is not necessary, as it does not need to be modified in any way.

Windows Driver/ Visual Studio

Shortcut to TSync-PMC driver release notes: I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\TSync driver updates

Paths to the TSync Windows driver and example programs
Path to the drivers/example programs:

Driver: C: \Program Files\Spectracom\TSYNC PCI\drivers\

Example programs
For TSync example programs: C: \Program Files\Spectracom\TSYNC PCI\Examples\TSYNC API\
For Legacy TPRO example programs: C: \Program Files\Spectracom\TSYNC PCI\Examples\TPro API

TSync.lib file

- (July 2013) refer to Salesforce Case 11052 As of at least Verion 2.4.1 of the Windows driver, this driver only support dynamically linked .lib file. It does not currently support statically linked .lib file (unlike the linux driver, which supports both).
- Per Tim Teatrault, we will need to compile the Windows driver differently when we build it, in order to support this capability.

TROUBLESHOOTING WINDOWS INSTALL
General-type questions to ask.
Here are some questions for you:

- What is the Serial Number of the TSync-PMC board (its indicated on the silver Spectracom Part Number sticker affixed to the TSync-PMC board)?
- What is the version of the TSync-PMC Windows driver you have?
- Has this TSync-PMC board been installed and operating normally in this same machine for a while, but has since stopped working with these errors just now being seen?
- Or, Is this a TSync-PMC board that was installed previously in another machine and then recently moved over to this IBM Windows server?
- Or, is this the first time being installed in a machine since it was purchased?
- At what stage of the installation process (assuming it's a new install in this machine), did the server errors start occurring? Was it when the board was first installed, while installing the driver, while working with it in conjunction with custom application software, etc?
- The TSync-PMC board can be installed without the driver being installed, and should be recognized in Device Manager. Is this board being detected as being installed in Device Manager.

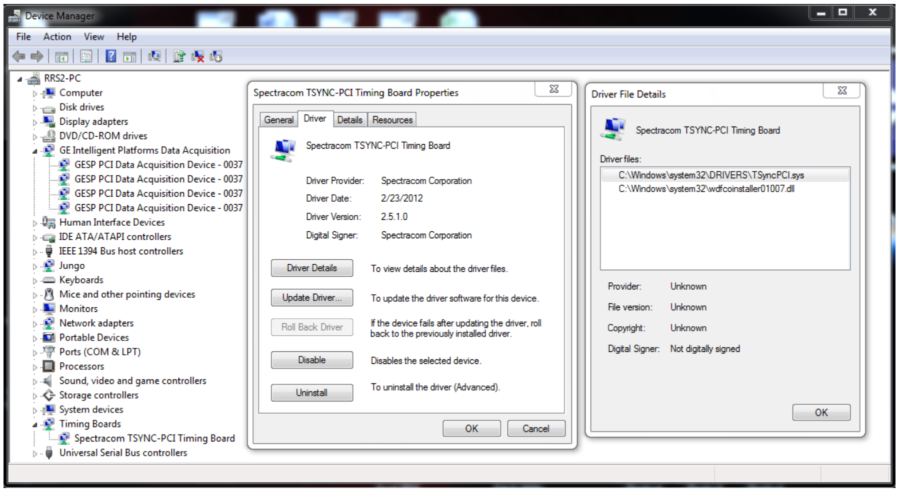
- If you haven’t already, are you able to temporarily install this TSync-PMC board in another Linux or Windows machine to see if it can be detected and able to communicate via the bus?
- Do you have any other PCIe products installed in the same system?

**Windows Driver compatibility/install issues/conflicts with other PCIe boards
Apparent possible conflicts with other installed PCIe boards in the same system.
Driver issues could cause conflicts. If boards are resetting, could also be an issue with the +3.3vdc from the bus dropping too low.

To troubleshoot possible conflicts between TSync and other installed boards, go into Hardware tab / Device Manager. For all PCIe installed products (TSync-PMC and others).

- Right click on each product and select “Properties”.
- Select the Driver tab.
- Select “Driver Details”.

This should list which driver(s) each installed board is linked to. We want to make sure the TSync-PMC board is not linked to any other drivers and the other products are not linked to the TSync-PMC driver. Please send a screenshot of this list to us.



RRS2-PC
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- GE Intelligent Platforms Data Acquisition
  - GESP PCI Data Acquisition Device - 0037
  - GESP PCI Data Acquisition Device - 0037
  - GESP PCI Data Acquisition Device - 0037
  - GESP PCI Data Acquisition Device - 0037
- Human Interface Devices
- IDE ATA/ATAPI controllers
- IEEE 1394 Bus host controllers
- Jungo
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Portable Devices
- Ports (COM & LPT)
- Processors
- Sound, video and game controllers
- Storage controllers
- System devices
- Timing Boards
  - Spectracom TSYNC-PCI Timing Board
- Universal Serial Bus controllers

Spectracom TSYNC-PCI Timing Board

Driver Provider: Spectracom Corporation
Driver Date: 2/23/2012
Driver Version: 2.5.1.0
Digital Signer: Spectracom Corporation

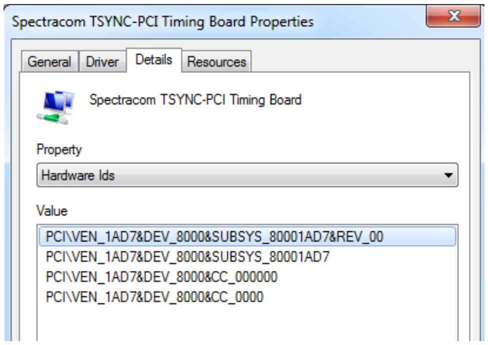
Driver Details — To view details about the driver files.
Update Driver... — To update the driver software for this device.
Roll Back Driver — If the device fails after updating the driver, roll back to the previously installed driver.
Disable — Disables the selected device.
Uninstall — To uninstall the driver (Advanced).

Spectracom TSYNC-PCI Timing Board

Driver files:
C:\Windows\system32\DRIVERS\TSyncPCI.sys
C:\Windows\system32\wdfcoinstaller01007.dll

Provider: Unknown
File version: Unknown
Copyright: Unknown
Digital Signer: Not digitally signed

- Now click on the Details tab.
- In the drop-down in the Details tab, choose “Hardware Ids”).
- This tab shows the sub-system ID information. Send us a screenshot of this list for all of the products.



Spectracom TSYNC-PCI Timing Board Properties

Spectracom TSYNC-PCI Timing Board

Property: Hardware Ids

Value

```
PCI\VEN_1AD7&DEV_8000&SUBSYS_80001AD7&REV_00
PCI\VEN_1AD7&DEV_8000&SUBSYS_80001AD7
PCI\VEN_1AD7&DEV_8000&CC_000000
PCI\VEN_1AD7&DEV_8000&CC_0000
```

If the Drivers for the TSync-PMC and other devices don't appear to be conflicting (as determined by the info above) and the TSync-PMC board is unexpectedly rebooting, using a scope (not a mulitmeter) check the +3.vdc going into the TSync-PMC board via its Test point to see if this voltage is dropping below the min, even momentarily.

Note: This may be too quick or too small of a drop to catch with a multimeter. Really need to look at this test point with a scope so that it can be triggered if it drops below about 3.16vdc (resulting in a reset).

TSync-PMC Driver support for earlier versions of Windows (Win 95, 98, CE, etc)

Info based on talk with Tim Tetreault:
The driver development tools we use start with Windows 2000 and go up from there. They do not list support for any earlier versions of Windows, including CE. Based on this information alone, the drivers are not likely compatible with CE. But, the only way to know for sure would be to install both the freely available driver in addition to the timing board (a demo board, if you have one available for them to try first, before buying??).
The driver contains individual folders for the particular version of Windows. When they install the driver, it may ask them which OS it is, since it may not be able to detect the version. They could select either XP or 2000 as alternate selections.

If they want to first try just installing the driver, they could try this, as the driver is available for free download (however, ultimately they would need to install both the driver and a timing board together to know it's going to work fine in CE).

TSync Driver version 2.41 can't complete the driver install because a newer version of Visual Studio 2010 is already installed on the PC

- Fogbugz case 1842 http://fogbugz/default.asp?pre=preMultiSearch&pg=pgList&pgBack=pgSearch&search=2&searchFor=1842&x=0&y=0
- Error message displayed when trying to install the driver: "A newer version of Microsoft Visual C++ 2010 Redistributable has been detected on the machine."
- Does not allow the driver install to proceed past this point.

  Dec 2012- Refer to Salesforce case 6784/Fogbugz case 1842 for more info)

As reported by NovaSol, if Visual Studio 2010 is installed and has had updates applied to it, the Windows driver install process will halt and there is no wasy to force it to proceed (apparently it doesn't like the fact that a newer version is already installed).

We will need to release a newer version of the TSync-PMC Windows driver that allows the upgrade proces to proceed, if a newer version is already installed on the PC.

Temporary work-around

Email from Dave Lorah to Tim T (67 Jan 2014) We had a customer last fall with a dpendent vc80.crt error and he was able to resolve the issue by totally uninstalling the PCI Driver and then reinserted the CD. When the CD Autostarts it wants to load C++ for you. They selected NO - and did not install anything. They manually selected the XP 86 and loaded just our Driver software. Then after that loaded C++2005 from the web. It then worked OK.

Is there a better workaround for this?

Tim responded back with Dave,I don't have a better work around. We will probably be looking into a Windows driver update that will include this issue but I don't have a time schedule.

**Example programs included with the TSync Windows driver

The path to the drivers is: C:\Program Files\Spectracom\TSync PCI\examples\Tsync API

If the Windows Example programs can't run, try using the Windows Control Utility to communicate with the TSync-PMC board.

Windows 32 bit OS
The example programs were initially compiled for **32** bit OS. They should work fine on any **32** bit Windows PC.

Windows 64 bit OS
Example programs run fine on 32 bit but won't work on 64 bit.

- To use the examples on a 64 bit system, they need to be rebuilt for 64 bit OS.

**Windows Control Utility GUI interface

- To open the Control Utility: Start / All Programs / Spectracom Corp / TSync PCI / TSync Control utility
- Provides a basic interface with the TSync-PMC board via a GUI (such as reading the current Sync status, for example)
- The TSync-PMC board does not need to be installed to use this GUI. Install the Windows driver and this GUI can be run in a "demo" mode to demonstrate the capabilities of the utility

**Clock Daemon program / Clock Daemon service (for all timing boards)

Refer to:
TSync-PMC boards: TSync-PMC driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121 for more information on the Clock daemon:

TPRO/TSAT-PCI boards: Section 2.2 of the PCI Application Programmers manual ("*2.2 Clock Daemon Utilities")* discusses the Clock Daemon program (refer to1186-5002-0050 (in Arena at: https://app.bom.com/items/detail-spec)

- Consists of the “Clock Daemon” and the “Clock Daemon Service”. Only one of these two can be used at a time.

Difference between the Clock Daemon and Clock daemon service:

- Clock daemon program needs to be started after each PC boot.
- Clock daemon service is a Windows service so it can start automatically after each PC boot.

Accuracy of the Clock daemon program
The PMC/cPCI timing board will try to set the time to within 1 microsecond (the board’s resolution). However, due to latencies and processing in Windows, typical expected accuracies is around 1 to 2 ms.

Desire to sync a Windows PC to a TSync-PMC board
A Windows PC won’t automatically sync to an installed TSync-PMC-PTP board. Instead, the TSync-PMC’s Windows driver contains a free utility program that is automatically installed as part of the driver. This program allows a Windows PC to be able to synchronize to the timing board. This utility is called “clock daemon”.
With the TSync-PMC synced to an input reference (such as PTP, as indicated by the green LED on the edge of the board being lit), the clock daemon allows the PC to sync to the time of the TSync-PMC board at scheduled intervals.

Attached you should find a copy of the TSync-PMC PCI driver guide. This provided utility is discussed in Section 2.3.5 of this document (starting on page 2-7). There are two variations of this utility (clock daemon and clock daemon service) discussed in this document. You can use either one, but not both simultaneously (the second one runs as a Windows Service instead of as a program).

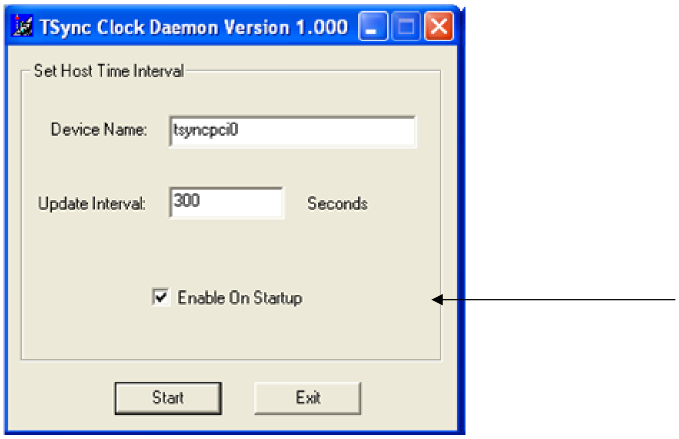
Desire to sync a Windows PC each time the PC boots-up
There are two supplied versions of this program. These are the “Clock Daemon” and the “Clock Daemon Service”. Only one of these two can be used at a time. Only the “Clock Daemon Service” is treated as an actual Windows service that starts up each time the PC starts. The “Clock daemon” program is not a Windows service and won’t automatically start upon a Windows reboot (user interaction is required when using this program). We recommend you switch to using the “Clock Daemon Service” if you wish for the program to start automatically after each Windows start-up.

Update interval / Update on Windows startup

- Default update interval is every 300 seconds

Per the Clock Daemon section of the driver guide, it can optionally sync at Windows start-up, instead of waiting for the first interval after the program has started:

For the Clock Daemon program:



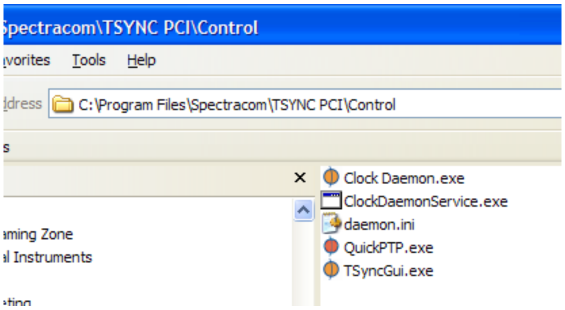
Device Name: tsyncpci0

Update Interval: 300 Seconds

☑ Enable On Startup

Enable On Startup: Checking this box will automatically start system clock synchronization by the clock daemon whenever the program itself starts. Alternatively, the synchronization can be started or stopped manually.

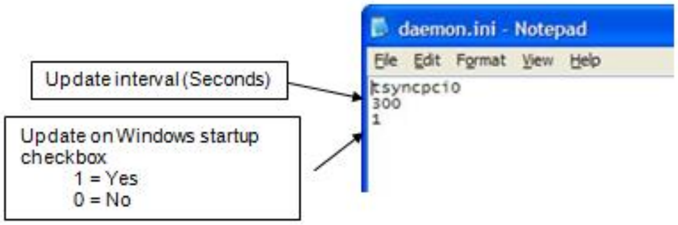
For the Clock Daemon Service:
The controls for device name and update interval are setup by the “daemon.ini” located in the “**Spectracom\TSYNC PCI\Control**” folder with the clock daemon executables. This initialization file is shared by both clock daemon executables.



C:\Program Files\Spectracom\TSYNC PCI\Control

Clock Daemon.exe
ClockDaemonService.exe
daemon.ini
QuickPTP.exe
TSyncGui.exe

The “1” is Enable on Startup selected. When not selected, the PC will wait the Update Interval after the program starts before its first sync.


daemon.ini - Notepad
File Edit Format View Help
Update interval (Seconds)
Update on Windows startup checkbox
1 = Yes
0 = No

```
tsyncpci0
300
1
```

Logging/information about PC drift rates

Q. Does the Clock Daemon create any logs for its time syncs?

- The Clock Daemon program/service by default, syncs the Windows PC to the timing board every 300 seconds. However, this value can be modified by a user as desired, via it's GUI interface. The only disadvantage of setting the time very often is that Windows sends a signal to all running programs each time the time has been synced. Syncing it too often (such as once a minute for example may cause issues with programs due to this signal often being sent. Otherwise, its fine to decrease the interval to sync the PC more often (to minimize the time that the computer is drifting).

This basic program/service just resets the time of the computer at the specified interval, based on PC drift since the last sync occurred. It does not log each time it syncs the PC or display any statistics on the PCs drift that is occurring,

However, another method of syncing Windows PCs which can provide time sync information is to use one or more of our Spectracom NTP servers connected to the network (instead of a bus timing board in each PC) in conjunction with our PresenTense suite of NTP Client software. PresenTense software is also a GUI-based software solution, but it also provides logging and alert notification capabilities if computers are not able to sync to the NTP server, and therefore begin to drift. A Spectracom NTP server can provide time stamps over Ethernet connections to all machines on the network. And if more than one NTP server is on the network, this client software can be configured to get time from up to three different NTP servers for redundancy.

Attached you should find a document that discusses how to sync Windows PCs. The last section of this document (Section 11) discusses this NTP client software, including what each program in the suite can provide.

Troubleshooting the clock daemon not working correctly

Note: the clock deamon in earlier versions of the PCI Windows drivers may have issues with 64 bit Windows (rerfer to Ken Moran from NOAA in Salesforce- email below was from him, also). We recommend updating the PCI driver to the latest version **(with Windows PCI driver v2.3.0, the TSync-PMC driver must also be installed before the PCI driver. Unistall PCI driver, install PCIe driver, and then re-install the PCI driver)**

Admin rights/ user rights
I'm fairly certain the user needs to be logged in with administrative rights in order for Windows to allow the Windows clock to be updated (this is a requirement of Windows- not the driver).

Set system time failed error 87"- Seems to be caused by using the Clock daemon utility in the Windows driver when using IRIG input to a board without having the year set to the correct value. Verify the year value is set correctly in the Windows Control Utility (See Emmett McCabe record in Customer Service database- around 2/5/08). Saw this on a TPRO-PMC card with Windows driver.

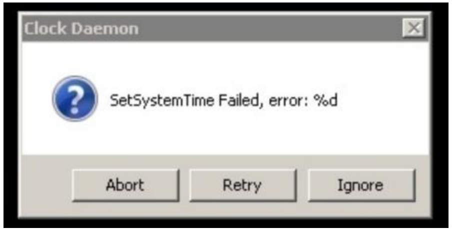
- Configuration window not displayed

Q. We loaded version 2.3 from your website, then removed the old version. When version 2.3 is installed there is now no-way to get the old daemon window to become active(this old window allowed us to change interval synchronization time.) We tried using the daemon .exe file and never see the old style window. We expected the old window would still be available. Presently to change the sync interval we go into the control folder and edit the daemon.ini file (which contains the start-up values for the time sync = 300s) and change it to 10s. We then need to boot the PC. We verified that the sync update is now 10s.

So our question is what happened to the old sync window that allowed us to change the rate. Is it possible we are doing something wrong?

Answer (10/26/12): When using the Clock daemon program in Windows PCI driver v2.3.0, the TSync-PCI driver must also be installed BEFORE installing the PCI driver. Otherwise, the clock daemon (and the Control Utility won't operate correctly, as described in the email above).

"Unable to set time!!"



SetSystemTime Failed, error: %d

Email from Keith (2 May 13) Three key pointers/requirements for you, when using either of the Windows clock daemons:

1) Make sure the TSync board is in sync to IRIG or GPS input, or to itself. The green LED on the edge of the card needs to be lit in order for the timing board to be used as a reference (Sync status can also be verified in the Control Utility using the Sync -> Status menu. If this box is not selected or if the green LED on the edge of the board is not lit, let me know if its connected to a GPS antenna that is installed outdoors with a good view of the sky, or to an IRIG generator, for its input synchronization, Then we can go from there!

2) When using IRIG input for its sync (instead of GPS), the year needs to be set in the TSync-PMC board each time it powers-up. With GPS input, the current year is automatically set after each boot-up. With IRIG input, if the IRIG generator provides the current year, the TSync-PMC board needs to be programmed after each boot-up to know where to look for the year in the IRIG time code message. Until its programmed to look for the year, or if the IRIG generator doesn't provide it, a user can manually enter the year using the Windows Control Utility (Date -> Set Year menu) or with a API call/example program using the driver. To verify the year is set correctly after each boot-uop, use the Date -> Retrieve Gregorian date menu in the Control Utility).

3) The Windows PC must be able to successfully talk to the board. the Control Utility can be used to verify this requirement is being met. If you successfully opened the Date -> Retrieve Gregorian date menu mentioned in the previous step, this requirement has been verified.

- (N/A for PCI boards - applies to PMC, cPCI and PC104) With Windows 7, 2008, 8, etc, make sure its not 64 bit Windows (driver is only compatible with Windows 32 bit)

Please also note the Windows driver is only compatible with Windows 7, 32 bit. It is not compatible with 64 bit Windows. At this time, there are no plans to make this driver compatible with 64 bit Windows

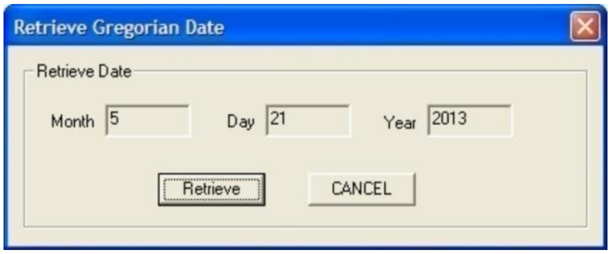
- Year needs to be set in the board

Set system time failed error 87"- Seems to be caused by using the Clock daemon utility in the Windows driver when using IRIG input to a board without having the year set to the correct value. Verify the year value is set correctly in the Windows Control Utility. (See Emmett McCabe record in Customer Service database- around 2/5/08). Saw this on TPRO-PMC card with Windows driver.

Email from Keith (30 Apr 13) Also when using IRIG input (instead of GPS input) to sync the TSync board, the timing board needs to either be configured to read the current year value from the IRIG generator (if it provides year information) or it needs to be set in the TSync-PMC board each time it boots-up. This can be accomplished with an API call/example program or via the Date-> Set Year menu in Control Utility (note this menu only sets the year-it does not read the year. The Date-> Retrieve Gregorian Date menu will read the current year set in the TSync-PMC board. The default year at boot-up is 2000, if the year is not set by IRIG input. GPS input does automatically set the year when the TSync-PMC board is synced by GPS instead of IRIG input.

- Windows must be able to communicate with the timing board

To verify the computer can communicate/access the board (and to also verify the year is set to 2013), open the Windows Control Utility (Start /All Programs/Spectracom Corp) and perform a **Date** -> **Retrieve Gregorian Date.** The Control Utility should respond with the date and correct year (example below)



Retrieve Gregorian Date

Retrieve Date

Month 5

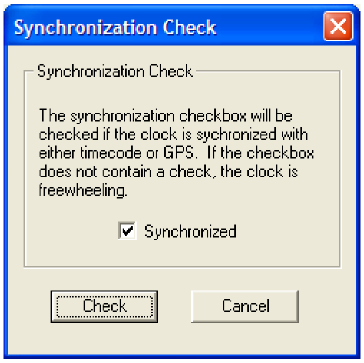
Day 21

Year 2013

- Timing board needs to be synced

Timing board is not synced: The timing Board must be synced in order for it to be used to sync the clock utility. With IRIG input, the year must be manually set (not applicable with GPS input) before the timing board will sync to the IRIG data. Refer to IRIG input synchronization section above.

If the date is correct, the system can access the timing board. Now check sync state. In the Control Utility, click on **Sync** -> **Status.** In the Synchronization Check window that opens, make sure the “Synchronized” box is selected (as shown below)



The synchronization checkbox will be checked if the clock is sychronized with either timecode or GPS. If the checkbox does not contain a check, the clock is freewheeling.

☑ Synchronized

TSync “QuickPTP Viewer / “TSync Viewer” GUI

When the Windows driver is installed, it also installs the “Clock Utility” (basic GUI interface), QuickPTP (PTP module interface) and the Clock Daemon program/service for syncing the PC to the TSync board

(Free PTP application for TSync-PMC-PTP boards and SecureSync=PTP)

- A graphical interface (GUI) installed as part of the Windows driver for the TSync-PMC-PTP boards.
- Version 2.3.0 of the TSync-PMC Windows driver adds a free GUI interface (called “QuickPTP”). This GUI is automatically installed in the Windows PC when the TSync Windows driver is installed.
- Time of the PTP module is displayed in HH:MM:SS.
- To access it: Start/All Programs/Spectracom Corp/TSync PTP/QuickPTP (a demo mode is not available; a TSync-PTP board needs to be installed in order to open it).
- Two screens are displayed. One is for the TSync-PCIe board (shown below on top) and the other is for the PTP module (shown below on the bottom).

Known issues with QuickPTP

- The “Clock Quality” value is not currently supported, but is planned to be fixed in a new TSync-PCIe release to be made available sometime after the SecureSync PTP module has been released. Currently, this value is displayed as “ERR” to indicate an error occurred while trying to read this value.
- As of at least 2/15/11 (and prior), the “Current NetMask” and “Current Gateway” fields are two enhanced feature fields that have been already been added to the QuickPTP GUI in anticipation of newer reporting capabilities with the integration of the next anticipated TSync-PTP firmware upgrade. These fields being blank, at this time, does not indicate that there is a problem with the PTP port. However, you should be able to ping the port and receive a response from it.
- As of at least 2/15/11 (and prior), “Clock Properties” fields (“Steps Removed”, “Offset from Master” and “Mean Path Delay”) show “ERR” (Error). This is also supposed to be fixed in the next version of the TSync-PCIe firmware.

PTP module shows Information displays all "0"'s after cold reset (example screenshot below

- Issues running on Windows 2012

Email from Richard Hochron (15 aug 2013) Can you confirm there is support for this card for Windows 2012 Server. I have noticed that while running the QuickPTP app that the server becomes unstable and blue screens. I have been running the app under 2008R2 for a while and it does seem more stable.

Reply from,Dave Lorah- As far as I know, this has never been tested on a Win 2012 server. I do know it works on Win 2008 and Windows 7.

Denis Reilly commented QuickPTP is that application that Hervé wrote that we included on the driver disc for customers. It hadn't been updated since we first released it. I wouldn't be surprised if it is now out of date with the latest Windows stuff. We should decide what to do with it.

FAQS about QuickPTP

Q. (From Chuck Beck of XRtrading 2/11/11) I think I had it working but was unsure, so I click the Reset PTP Cold button and after that I was unable to set anything on the card. The GUI only shows zeros on all the fields. See attached JPG.

A. (Email from Denis- don't send this as-is to customers)
When the GUI shows all '0''s, it means that the module is not communicating with the TSync.
When the PTP module on the TSync-PTP gets cold reset, the module itself can take a minute or so initialize. (Particularly if DHCP is enabled). During initialization, the module does not communicate, and the TSync deletes all the prior PTP state information, so you will get stuck at all '0''s until communication comes back. In rare circumstances, if there is a lot of non-PTP network traffic when the card is reset, the module could be spending so much of its time filtering that that it doesn't have time to communiate properly. I saw this in development, and I really hoped they fixed it before shipping! :)

I would recommend a cold restart of the system the TSync card is in with the network cable unplugged. Communication should come back by the time the system boots. If that doesn't help, I'll have to think about it some more.

Questions from Sales about PTP:

1. Regarding boundary clocks – is there general information here or do we need to understand the customer's application? I'm sure a customer can support NTP and PTP both from the SecureSync, but if there are any specific comments related to this, please let me know.
2. Any switch that accepts PTP is compliant with Spectracom products, correct? However. I do know that the configuration of the network itself does impact accuracy, so not sure we can easily answer the "what is the accuracy" question. Please add comments here.
3. Not sure about this PTP with Linux Red Hat or if we support at chip level. Comments?

Email response from Denis Reilly (10/18/11) for all three questions
I'll start with just some general PTP information. I apologize if it's too basic, but you say you are "coming up to speed". Let me know if I forgot to answer anything. ;)

1st basic point: on SecureSync, NTP comes out of the main network port (and the additional Gigabit network ports). PTP only comes out of the network port on the PTP card. There is no problem running NTP and PTP over the same network, but the customer will have to take care of configuring the NTP and PTP interfaces properly for the network.

Even though PTP is a master/slave protocol, PTP communication is bi-directional. PTP makes a fundamental assumption that the delays on a network are symmetrical – that the delay in one direction is the same as in the other direction. PTP also works best when the delay does not change much with time. Modern switching equipment has all manner of queues and delay sources that can foul this up – if packets are queued in one direction and not the other, this will affect timing performance.

To help compensate for this, there are two special types of PTP equipment defined in the standard. One type or the other is implemented in "PTP Enabled" switches:

Regarding Linux, our SecureSync product and our TSync product implement the entire PTP stack in “hardware”. (Really a microprocessor, but close enough to hardware). If a customer wants to run a SecureSync PTP server and run PTP clients in Linux, they have several software-based alternatives. The best is to buy TimeKeeper from us. TimeKeeper will synchronize the Linux system clock with PTP. It’s expensive, but is much more accurate than any other software solution: we are selling it into that financial market where they want their Linux kernel synchronized to under 1 us. Timekeeper will work with a normal NIC card, but will work better with a PTP-enabled NIC card with “Hardware PTP timestamping”, and will work best with a TSync PTP card (although that is very expensive – ask Will about that!)

**Spectracom batch file to capture responses from a specified API call

- KW created an API call batch file in Dec 2012.
- Link to batch file: [EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\API Call batch files](EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSYNC-PCIe\API Call batch files).

(12-03-12 KW email to Evan Dietz) Attached you should find the zipped batch file. The file extension has been change to .aok in order to send it to you. Just rename the file to .zip and unzip it to a directory on the hard drive (such as C:\temp for example) of the Windows PC which has the TSync-PMC board installed. Then, run this batch file. The batch file will walk you through the process. I

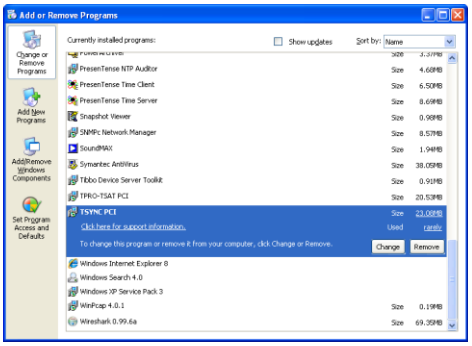
This batch file will capture the results of the desired API call (such as HW_GetPhaseError for your desired test) to a text file in c:\temp. Each time the API call is sent, a hardware time stamp from the TSync board is sent to another file, also in c:\temp. If desired, you can correlate the two text files to know when each API call was issued and the TSync board responded.

The batch file is looped, so it will continue to capture the results of the API call each time a key is pressed. When you want to complete the capture, simply "X" out of the batch. Then, if desired, you can either convert the text file to .xls or copy/paste the responses in the tsync.txt file to an excel spreadsheet. Then, you can also easily create a line or bar graph for the responses!

Uninstalling the current version of the Windows TSync-PMC driver:

To begin, the older Windows driver should be removed before installing the latest version of the driver. The original version can be installed using the standard Windows uninstall program (depending on the version of Windows, such as Control Panel/ Add or Remove programs, for example). When clicking on this Windows program, "TSync PCI" driver will be listed as one of the installed programs that can be removed from the PC. Click on the "Remove" button.

Below is an example of using the windows Add or Remove Programs to uninstall the original driver.


Add or Remove Programs
Change or Remove Programs
Add New Programs
Add/Remove Windows Components
Set Program Access and Defaults
Currently installed programs:
□ Show updates
Sort by: Name
Change
Remove

PowerArchiver	Size	3.37MB
PresenTense NTP Auditor	Size	4.68MB
PresenTense Time Client	Size	6.50MB
PresenTense Time Server	Size	8.69MB
Snapshot Viewer	Size	0.98MB
SNMPc Network Manager	Size	8.57MB
SoundMAX	Size	1.94MB
Symantec AntiVirus	Size	38.05MB
Tibbo Device Server Toolkit	Size	0.91MB
TPRO-TSAT PCI	Size	20.53MB
TSYNC PCI	Size	23.08MB
Click here for support information.	Used	rarely
To change this program or remove it from your computer, click Change or Remove.		
Windows Internet Explorer 8		
Windows Search 4.0		
Windows XP Service Pack 3		
WinPcap 4.0.1	Size	0.19MB
Wireshark 0.99.6a	Size	69.35MB

Installing a new Windows driver
Per the install instructions, we recommend the customer reboots the PC after installing driver. The driver reports it has been installed correctly after install, but may not be able to talk to the board until the PC has been rebooted.

Email from Tim T: In the install instructions, we recommend that you reboot the computer after installing the driver so that the driver gets loaded properly. This is all part of Windows Plug & Pray. This is not an install failure.

The driver can be manually installed through the control panel. I think it was Dave L. I was trying to explain this to earlier this week. When I get some time I will look into the driver install to see if Windows can be forced to scan for new hardware.

Windows driver example programs:
There is an example program that can be used to obtain the current DAC value. Open Windows Explorer and navigate to the following directory: C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API. You will see all of the available example programs that can be run.

To run an example program:
Open a command prompt window and navigate to the "**C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API**" directory. Then, run the desired example program (such as XO_ GetDac.exe to get the current DAC value, for example).

Cygwin
(12/17/10 kw) Cygwin is a Linux shell that runs in a Windows environment (like DOS running in Windows). Arnaud from France has a customer that desires to compile and perform API calls right in Cygwin. Based on info from Denis and Tim T, I emailed him the following information:
Official response from Tim T about this desired operation:
I downloaded and installed Cygwin last night on my computer to test our driver. Our driver will not compile as is.
This is not a true Linux environment. This is a Linux shell.
Our driver compiles differently depending on what kernel is installed. When you prompt Cygwin to report what kernel is running, it reports its own revision, not the kernel.
If this customer has experience with Cygwin, they might be able to setup the environment with the correct source files. They will also need to modify our driver to compile.

Since Cygwin is already running on Windows, I would recommend that they use our Windows driver.

My response to this customer:
I have been speaking to our Engineers regarding your customer's desire to compile and run API calls from Cygwin. We have just tried our Linux driver in this environment and found that the driver is not directly compatible. It will not compile, as it is (this is because Cygwin is only a Linux shell and not a true Linux environment). In order for our driver to be able to compile, it needs to know the version of the linux kernel that is installed. When Cygwin is prompted for its linux kernel version, it reports its own version instead of the kernel version, preventing the driver from being able to compile.

If your customer has experience with Cygwin and desires to operate the TSync-PMC board in Cygwin, they may able to setup Cygwin with the correct source files. They are certainly welcome to use the source code that we provide to try and create their own driver from the source code.

If possible, since Cygwin is already running on Windows, we recommend they use our Windows driver, instead. Otherwise, they will have to create to their own driver from our provided source code. We have no experience with this desired operation and so we won't be able to provide them any additional assistance with this desired task.

Visual Studio (VS)/ Microsoft Redistributables

Visual Studio
Used for developing application software on Windows PCs.

There are several versions of Visual Studio, including Visual Studio 2003, Visual Studio 2005, Visual Studio 2008 and now Visual Studio 2010.

We use Visual Studio 2005 (VS2005)
Email to a customer based on info from Tim Tetreault (~Oct 2011)

Below is the information from Engineering regarding Visual Studio 2010 with the PCI boards:

We only have a Visual Studio 2005 C++ static library.

Have you tried the driver we have on our website (see link below) to see if it can be imported into Studio 2010? If not, can you give it a shot and let us know if it works for you? The driver is free from our web site. Once downloaded, you can attempt to install it, even without the timing board installed. If there is a compatibility issue with installing it, you will know this even without having to have a timing board on-hand.

The Timing board drivers can be downloaded from our website at:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=20

Visual Studio 2010 (VS2010)

Email from Tim Tetreault (30 Oct 2013) I know we have other customers that are using VS2010 but I don't know for sure if they are using it in the what your customer intends on using it. The only way for the customer to know for sure if their applications will work with our board is to give them a demo board to try.

Resource wise we are thin right now so it's hard for me to find the time to do a test setup. And if we did, it's still no guarantee that the way we test it is the way the customer intends on testing it. Is the customer open to demo the board so he can try it out?

Q. Is there a Visual Studio 2010 C++ static library .lib available for your TSync PCIe card? When I go to your website it appears that there is only a Visual Studio 2005 version available under the latest SDK 2.41.
A. Email from Tim Tetreault (9/30/11)
We only have a Visual Studio 2005 C++ static library.
We currently have no plans in the near future to update this.

Has the customer tried the driver to see if it might work or if it can be imported into Studio 2010? The driver is free from our web site.

FAQs about Visual Studio
Q. Do you know if the TSync-PMC is compatible with visual studio 2003?
NOTE: Refer to the "**Update**" to the answer below. The answer is no. They need to upgrade to a newer version of Visual Studio

A. just spoke to one of our timing board engineers to confirm my reply. He said that the TSync-PMC Windows driver is likely compatible with Visual Studio 2003. However, we haven't tested against this earlier version of Visual Studio (We use Visual Studio 2005, instead).

He brought up a good point that all of the TSync-PMC drivers are readily available for free download from the Spectracom website. And, as the timing board does not need to actually be installed in the PC order to install the driver, they can pre-test this before they purchase and receive the TSync-PMC timing boards.
To download any of the freely available TPRO/TSAT or TSync-PMC timing board drivers, just have them visit us at: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=20. To obtain the TSync-PMC drivers, click on "PCIe". The "Documents" folder contains the TSync-PMC manual and driver guides, while the "software" folder contains individual folder for each of the available drivers.

Update (10/26/11 KW):
We have now had two customers (refer to Ken Liske with Rockwell Collins) getting warning and error messages when using Visual Studio 2003. The warning message is ignored but the error messages prevent the code from successfully compiling. The fix is to update to Visual 2005(or higher)

As of at least 2/3/11, the TSync-PMC version 2.3.0 Windows driver appears to not be fully compatible with application programs built with Visual Studio 2003. There will likely be tpro.lib and get link errors present. Example below:

Compiling...

```
stdafx.cpp
Compiling...
IrigMgr.cpp
BtiKsiTpro_DLL.cpp
BtiKsiTPro.cpp
Generating Code...
Compiling resources...
Linking...
tpro.lib(tprodll.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32' name='Microsoft.VC80.CRT'
version='8.0.50727.762' processorArchitecture='x86' publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
LINK : warning LNK4075: ignoring '/EDITANDCONTINUE' due to '/OPT:ICF' specification
tpro.lib(tsync_hw.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tpro_capabilities.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_Gr.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tpro_date_time.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_cs.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_ls.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_Ss.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_Go.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_pp.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_ip.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_ir.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_pr.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_rs.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_Generic.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_base_transaction.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_Gr_Services_recipes.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
```

```
tpro.lib(tsync_misc_Services_recipes.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_cs_Services_recipes.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
name='Microsoft.VC80.CRT' version='8.0.50727.762' processorArchitecture='x86'
publicKeyToken='1fc8b3b9a1e18e3b'' encountered; ignored
tpro.lib(tsync_ls_Services_recipes.obj) : warning LNK4229: invalid directive '/manifestdependency:type='win32'
```

Ken Liske (with Rockwell Collins) also received the following using Visual Studio 2003:

- Warning LNK 4229 "invalid directive'/manifestdependency' encountered; ignored" This is an ignored warning message, so doesn't cause any issues

- lnk2001 unresolved external symbol __imp___Snwprintf_S" and "fatal error LNK1120: 1 unresolved externals" : these are both error messages that are preventing their application code from compiling. We recommended he update to Visual Studio 2005 or higher.

Linux driver specific
Shortcut to TSync-PMC driver release notes: I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\TSync driver updates

Note: CentOS and Scientific Linux are Linux distributions that are based on the Red Hat linux distribution. Issues that affect Red Hat Linux will also likely affect these other distributions as well.

**Notes about Linux drivers

- (Nov 2011 Version 2.5.0) The PCIe Linux driver does not support any 2.4.x kernels. It now supports 32 bit or 64 bit 2.6.x kernels from 2.6.9 through 2.6.39 and 3.0.0 (see note below)

Unofficial kernel support with version 2.5.0 linux driver
Salesfore Case 7453 Keith, the driver was able to build and load without issue on our 3.5.7 kernel.

Lisa Perdue was also able to compile this driver on kernel version 3.2.0.

Note (5 Sept 2013): Tim Tetreault as tested v2.5.0 driver on kernel versions to 3.8.0 with no problems. A minor patch is required for compiling on version 3.9, as described below (this patch will be included in the next driver update)

I found we have tested Kernel 3.8 and it works fine. Version 3.9 has one slight problem which will be addressed in a future Driver update. This should be released in early 2014. In the mean time there is a fix:

I believe the only change required is to the file "tsync_drv_2_6_13_1.c" located in "/tsync/linux/driver/" folder in our driver. The change is

```
/*
**  tsyncpci_driver PCI driver information structure
*/
```

```
static struct pci_driver tsyncpci_driver =
{
   name:     TSYNC_PCI_DRV_NAME,
   id_table: tsyncpci_pci_tbl,
   probe:    tsyncpci_init_one,
#if LINUX_VERSION_CODE >= KERNEL_VERSION(3,8,0)
   remove:   tsyncpci_remove_one,
#else
   remove:   __devexit_p(tsyncpci_remove_one),
#endif
   shutdown: tsyncpci_remove_one,
};
```

Once you make this change, you should be able to compile the driver without any errors.

Previous Version 2.4.1 driver
(4/17/09 Version 2.4.1) The PCIe Linux driver does not support any 2.4.x kernels. It supports 32 bit or 64 bit 2.6.x kernels from 2.6.13 to 2.6.25.

5/26/09 –Compatible with x86 processor only. Does not support Sparq processor.

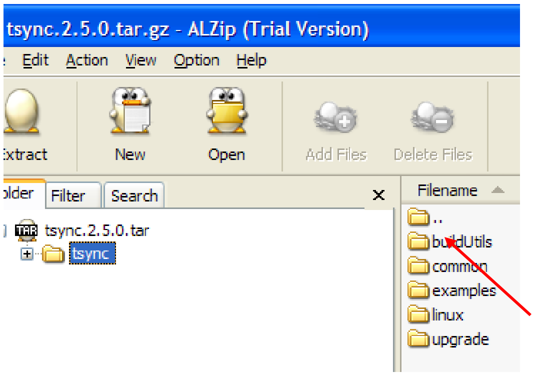
Driver needs to be built before use. Requires GCC and MAKE utilities and the GNU C library.

Email from Tim Tetreault to a customer: The driver its self was written using C/C++ but that doesn't mean you can't use other software languages to interface to it. I know we have other customers that have written software in Java to interface to our boards. I do not have example Java code that I can give you though.

Q. Can the card be used with a PowerPC Board (P2020 by Freescale) or just on a x86 Platform.
A. **(From Tim Tetreault 6/21/12)** The card has been used on PowerPC machines running Solaris but I am not sure about Linux. Are they running a version of Linux on the PowerPC machine?

**Determining the current linux driver version

- The version of the driver is indicated in the name of the tsync .tar file (such as tsync.2.5.0.tar indicates driver version 2.5.0 for example).



Installing the Linux driver (no previous version of the driver already installed

Steps to install the linux driver

- Open a terminal window.
- Make sure you are logged in as a root user.
- Copy the driver file to a convenient directory location.
- Change to the directory in which the driver files were copied.
- Extract the driver using the following command: gunzip –c tsync.<rev>.tar.gz | tar –xvf –
- Build the driver by issuing the commands below:

```
cd tsync
cd linux
make clean
make
make install
```

- Load the driver by issuing the command: modprobe tsyncpci
- To verify that the driver has been installed, type at the prompt: lsmod
- Verify that the driver “tsyncpci” is present.

Notes:
“rmmod tsyncpci” = unload loadable module
“modprobe tsyncpci” = rebuilds and installs the driver
“lsmod” = prints the contents of the /proc/modules file. It shows which loadable kernel modules are currently loaded.

Example successful customer driver install

“Here is the final steps that I took on the machine that is working:”

```
[root@vr2b linux]# make install
make -C driver install
make[1]: Entering directory `/home/vruser/gps_tsync/tsync/tsync/linux/driver'
rm -f /etc/udev/rules.d/25-tsyncpci.rules
install -D -m 644 25-tsyncpci.rules /etc/udev/rules.d/25-tsyncpci.rules
rm -f /lib/modules/2.6.32-358.el6.x86_64/kernel/drivers/tsyncpci.ko
install -D -m 644 tsyncpci.ko /lib/modules/2.6.32-358.el6.x86_64/kernel/drivers/tsyncpci.ko
depmod -a || true
make[1]: Leaving directory `/home/vruser/gps_tsync/tsync/tsync/linux/driver'

[root@vr2b linux]# modprobe tsyncpci

in /var/log/messages:
Nov 22 07:34:49 vr2b kernel: Spectracom TSync-PMC Timing Board - version 2.50
Nov 22 07:34:49 vr2b kernel: Copyright (c) 2009 Spectracom Corporation
Nov 22 07:34:49 vr2b kernel: tsyncpci 0000:05:00.0: PCI INT A -> GSI 34 (level, low) -> IRQ 34
Nov 22 07:34:49 vr2b kernel: TSYNC version of board: tsyncpci0
Nov 22 07:34:49 vr2b kernel: IRQ 34/tsyncpci0: IRQF_DISABLED is not guaranteed on shared IRQs

[root@vr2b vruser]# modprobe -l | grep -i tsync
kernel/drivers/tsyncpci.ko

[root@vr2b linux]# ls -al /dev/|grep -i tsync
crw-rw-rw-   1 root root    248,   0 Nov 22 07:34 tsyncpci0
```

Everything looks good at this point, and the card is in fact working.

****Updating the driver (Uninstalling a previous version Linux driver and installing a newer version)**

Note: When changing the linux kernel version, the Linux driver install steps needs to be repeated.

Email from Tim Tetreault: If the kernel has changed, you will need to repeat the driver installation. Make sure you uninstall and "make clean" before you rebuild the driver.

- Uninstall the earlier driver first
- Unload the driver by issuing the following command:

```
rmmod tsyncpci
```

- Change to the directory in which the driver files were copied.
- Unload the driver by issuing the following commands:

```
cd linux
make uninstall
```

B. Install the newer version driver

- Open a terminal window.
- Make sure you are logged in as a root user.
- Copy the driver file to a convenient directory location.
- Change to the directory in which the driver files were copied.
- Extract the driver using the following command: gunzip –c tsync.<rev>.tar.gz | tar –xvf –
- Build the driver by issuing the commands below:

```
cd tsync
cd linux
make clean
make
make install
```

- Load the driver by issuing the command: modprobe tsyncpci
- To verify that the driver has been installed, type at the prompt: lsmod
- Verify that the driver "tsyncpci" is present.

Notes:
"rmmod tsyncpci" = unload loadable modules
"modprobe tsyncpci" = rebuilds and installs the driver
"lsmod" = prints the contents of the /proc/modules file. It shows which loadable kernel modules are currently loaded.

Attached is a copy of the TSync-PMC driver version 2.41 release notes. The first paragraph provides the link to download the latest version of the driver, at no cost. Also attached is the latest version of the TSync-PMC driver guide. Refer to Section 2.1.2 (page 2-2) for information on uninstalling the current driver and then refer to Section 2.1 (starting on page 2-1) for information on installing this new driver.

Specific to RT Preempt patch

Here are some RT-Preempt patch links:
https://rt.wiki.kernel.org/index.php/Main_Page
https://rt.wiki.kernel.org/index.php/RT_PREEMPT_HOWTO

- Specifically, we added the RT-Preempt patch to the RHEL 6.3 (see below) kernel, and want to run with that. We use kernel 3.2.13 and apply the RT Preempt patch, giving us kernel 3.2.13-rt23. Do I just simply need to just boot into 3.2.13-rt23, and then do the "make" and "make install" again? How can I build two `tsyncpci.ko` drivers, one for 2.6.32 and one for 3.2.13-rt23?

Email from Tim Tetreault (9/14/12) If the kernel has changed, you will need to repeat the driver installation. Make sure you uninstall and "make clean" before you rebuild the driver.

Be aware that we have seen other customers that have patched Red Hat with a RT patch that weren't the same kernel that ended up having "make" issues because kernel paths were broken. You will have to sort those out if they happen.

For each kernel, you will need to go through the building of the driver to get the correct "tsyncpci.ko". So you will need to do this for the 2.6.32 and again for the 3.2.13-rt23.

****Best way to verify Linux driver has been properly/successfully installed.**

Email from Tim Tetreault (9/18/12) The first thing you can do to verify that the driver is installed and working correctly is to type:
Lspci –v –d:8000

This will identify our card on the bus. If the driver is installed correctly, it will also show that kernel driver in use and kernel module is "tsyncpci".

If all you are trying to do is verify that the driver is talking to the board correctly, you can try the following commands:

```
./HW_GetTime 0
./CS_GetTime 0
./LS_GetVersion 0
```

There are many other API calls but without knowing what you application is, I don't know what to suggest.

Note: The lspci command is discussed in more detail in the linux driver troubleshooting section just a little ways further below. Refer to**:** **ISPCI commmands**

Uninstalling, Loading and Unloading the Linux driver

- **"rm tsyncpci"** = uninstalls the current TSync-PCI driver
- "**make uninstall**" & "**make clean**" = to remove any old driver files.
- **"modprobe tsyncpci"** = rebuilds and installs the driver

---

32 Bit application software on our T64 bit driver (compat_ioctl function)

- Refer to Salesforce case 11668 and Fogbugz case 1856
- Applicable to at linux drivers at least version 2.5.0 and below.

Email from David Qi with Raytheon We would like our 32-bit application to work with the TSync-PMC Linux driver when it is built to run on a 64-bit Linux kernel.

We know that a 32-bit application cannot work with the TSync-PMC Linux driver when it is built to run on a 64-bit Linux kernel, because the TSync-PMC code does not use provide driver support -- in the form of a compat_ioctl function -- that would allow a 32-bit application to work with the driver when it is built to run on a 64-bit kernel.

So, the question we have is this ...

Would Spectracom be able to provide us (and its other customers) with a version of the TSync-PMC Linux driver that allows a 32-bit application to work with its driver, when its driver is built for a 64-bit Linux kernel?

.lib file (libtsync.so)

- As of TSync-PMC Linux driver version 2.3.0 (Dec 2010 time-frame), the linux driver supports both statically linked and dynamically linked

  From The linux driver Release Notes- **Added shared library support (libtsync.so).**

- The libtsync.so shared library file is generated during the driver build. Its located in the Root/Desktop/TSync/Linux/tsync/Lib directory

Q. I am now trying to link to your driver on my 64-bit linux system since previously I actually had the driver code compiled into my library and that was leading to the undefined symbols. Unfortunately when I link I get the error message near the bottom of the following:

A. Currently our libraries are not built as shared object libraries. They are intended to be statically linked. The lowest level Makefile for our driver library with its CFLAGS is located here: \tsync\linux\tsync\lib\obj\Makefile.

(Note that this was resolved in the version 2.3.0 driver update, Dec 2010)

**Example programs for Linux driver

Linux Driver example programs:

With the TSync-PMC board and driver installed, the provided Example programs need to first be built/generated, before they can be used. Refer to “Generating the Example Programs” in the driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121 for the instructions.

Generating the Example programs to be able to use them

A “**makefile**” file is supplied in the **/tsync/examples** directory of the Linux driver. A “make” command is performed to build the examples. The “makefile” can be either used as-is or the customer can modify it as necessary (potentially breaking the example programs if it’s incorrectly edited).

The path to the makefile (“mkDriver”) is C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API\Src.

Generating the Example Programs (From the TSync-PMC driver guide)

The TSync driver provides both a static library (libtsync.a) and a shared library (libtsync.so). The example programs are built linked with the static library. To use the example programs with the shared library, modify the example “makefile” by replacing the libtsync.a with libtsync.so and rebuild.

1) Open a terminal window.

2) Change to the directory in which the driver and its sources were extracted.

3) Build the example programs by issuing the commands below:

```
> cd tsync
> cd examples
> make clean
> make
```

Example programs are all located in: Desktop/tsync/examples

To use the example programs: Either open a terminal window (In “Applications”) or a shortcut to the executable needs to be created and then modified with parameter to specify the board number (normally always “**0**”) and the instance number (as applicable for each command). Just clicking on the ICON for a particular example won’t run that example program.

Notice: Must type **“./ “** before each command in order for the call to process. **Example**: **./LS_GetMessage 0**

- Terminal window will provide a command line prompt.
- Use the ls command to list the example programs.
- Use an astericks (“*”) for a wildcard
- Commands are case-sensitive.

Email Keith sent to customer (5/31/12) We supply the source code for all of the example programs that can be used to communicate with the TSync-PMC board. The example programs need to first be generated using the “make” command, before they can be used.

Attached you should find the latest version of the TSync-PMC driver guide. Please refer to Section 2.1.3 (page 2-2) of this guide to generate the Example programs (if you haven’t already).

Example programs for 64 bit Linux

The example programs were initially compiled for 32 bit. In order to use them with 64 bit, all of the examples need to be rebuilt for 64 bit.

The make file needs to be modified to use 64 bit examples. The path to the makefile “mkDriver” is C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API\Src. Refer to Section 2.1.3 (page 2-2 of the attached TSync-PMC driver guide).

Once the examples are built for 64 bit, they should work fine for you.

Email Keith sent to customer (5/31/12) We supply the source code for all of the example programs that can be used to communicate with the TSync-PMC board. The example programs need to first be generated using the “make” command, before they can be used.

Attached you should find the latest version of the TSync-PMC driver guide. Please refer to Section 2.1.3 (page 2-2) of this guide to generate the Example programs (if you haven’t already).

Example programs provided with Linux driver don’t work

If the following two statements are present in a “LSPCI -v” response, the TSync-PMC board/driver are installed, but the Example programs may not have been built yet. Or the “makefile” file supplied in the Linux driver may have been modified from the default settings. Send any changes to this file to engineering for review. If lspci doesn’t show these statements, the board and driver aren’t properly installed yet, so the Example programs can’t run. Refer to the Linux driver section of this document for specific troubleshooting guidance.

Kernel driver in use: tsyncpci”
“Kernel modules: tsyncpci”

If the TSync-PMC board and linux driver are installed, but the Example programs still don't work, refer to: Example programs included with the drivers (in this document).

With the TSync-PMC board and driver installed, the provided Example programs need to first be built/generated, before they can be used. Refer to "Generating the Example Programs" in the driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121 for the instructions.

Note: The example programs with the linux driver need to be generated (built) separate from the driver itself.

Refer to the instructions further above on Generating the Example programs.

Multi-core / Multi-thread applications

Q. Can the TSYNC card be opened and operated via the Linux device driver by more than one program or thread of a program at a time? In other words, can multiple programs use the card at the same time and read time, set alarms, etc.?

A. Yes our Linux driver is designed to support Multicore & Multi Thread applications. (All three drivers support multi-threaded applications running on multiprocessor machines)

**TROUBLESHOOTING linux driver:

A) Difficulties installing/building the driver

- Refer to Salesforce case 10992
- Driver install requires GCC and MAKE utilities and the GNU C library.

Email/Questions Keith sent to a customer to get additional info to diagnose

In order to better assist you with the TSync-PMC driver install, I have some questions for you:

- What is the Serial Number of the TSync-PMC board (its indicated on the silver Spectracom Part Number sticker affixed to the TSync-PMC board)?
- What is the version of the TSync driver you are trying to install? Note the latest driver version is version 2.5.0 and the version of the driver is indicated in the name of the .tar file (such as tsync.2.5.0.tar indicates version 2.5.0 for example).
- What is the linux kernel version installed in this particular machine? Note the latest driver (version 2.5.0) is officially compatible with Linux kernel versions 2.6.9 to 2.6.39, and 3.0.0. However, it has been successfully installed on kernel versions 3.2.0 and 3.5.7
- Are you getting any reported specific or general compile error message being displayed?
- Is the TSync board already installed? If it is, temporarily uninstall it. Then, try to build the driver again with it removed. Does it successfully build this time?
- Do you have GCC and Make utilities as well as the GNU C library installed

If you are getting any compile errors, please send me a copy/paste or screenshots of the error messages for our review. Based on the responses above, we may need to get our Engineering team involved with the diagnosis of why the driver isn't building for you.

V) TSync-PMC board is unexpectedly rebooting

- **If the TSync-PMC board is unexpectedly rebooting for any reason:**

  - Verify the +3.3vdc supply voltage from the PCIe bus. It may be dropping below the minimum of about 3.17 vdc, even for just a moment, causing the low voltage detection circuit to reset the board. Refer to Salesforce case 10297 for an example. For troubleshooting, refer to: Power (+3.3vdc +12vdc and -12vdc from PCI bus)/logic in this document.
  - Verify there are no generic class clode/driver conflict issues, where something else installed is causing the board to reboot. Refer to Salesforce case 10297 for an example. For troubleshooting with a Windows PC, refer to: Windows Driver compatibility in this document.

W) Two or more TSync-PMC boards installed, but only one being detected

Troubleshoot: Try swapping the known good board with one that is not being recognized (put it in the slot that is recognizing the board being installed). If the other board is recognized, the board is OK. It's a system issue preventing the other board(s) from being detected.

- lspci commands
- Checks to see if the timing board is visible on the PCI bus
- Checks to see if PCIe driver is installed.
- Our Vendor ID: note from Denis Reilly (12 Feb 2013) "Our vendor ID is 1AD7".

Perform an lspci –vvv

- Open a terminal window and type the commands.

Issues that appear to be associated with the hardware (such as interrupts): Have the customer perform and send to us an lspci –vvv. This lists modules installed in the machine.

Even without the TSync-PMC driver installed, this command should show the TSync board being installed.

In Linux, you can run the following command as root to look for the PCIe card specifically on the PCI bus:

"lspci -v" = Lists the devices founds on the PCI bus (-v gives minimal info, -vv give more info and -vvv gives all available info)

lspci -d 1ad7:8000

"**lspci –d:8000 –vv**" If the lspci response above has no indication of "tsyncpci", the TSync-PMC board is not being found. This command should then be issued to see if the board can be found using the sub system ID number assigned to the TSync board.

If still no indication of "tsyncpci" in the response, may be a bad board or a major linux system issue.

Note: The following entries should also be present, if the driver is installed:

Kernel driver in use: tsyncpci
Kernel modules: tsyncpci

Results of lspci:

1. Original/Generic Class Code (prior to ~March 2013)

First line of response should be either:

```
nn.nn.n Non-VGA unclassified device: Spectracom Corporation Device 8000
```

- nn.nn.n Non-VGA unclassified device: Unknown device 1ad7:8000

(where **nn.nn.n** is the bus ID number and is system-dependent, and **1ad7:8000** is our Vendor and Device ID for the TSync card)

Note: With the TSync board installed and the driver loaded, the following two lines should be present in an lspci response (likely at the very end of the response).

```
Kernel driver in use: tsyncpci”
“Kernel modules: tsyncpci”
```

The command will return nothing if the system bus can't see the card.

2. Updated Class Code (ECN 3159- released 4 Apr 2013)

- Approximate Manufacturing cut-in Serial Number to have the new class code: 01998

The new class code will now report:
Base Class = 11h (Communications synchronizer) Sub-Class = 10h Interface = 00h

First line of response should be:

```
nn.nn.n 1a:00.0 Communication synchronizer: Spectracom Corporation Device 8000
```

(where **nn.nn.n** is the bus ID number and is system-dependent, and 1ad7:8000 is our Vendor and Device ID for the TSync card)

Note: With the TSync board installed and the driver loaded, the following two lines should be present in an lspci response (likely at the very end of the response).

```
Kernel driver in use: tsyncpci”
“Kernel modules: tsyncpci”
```

3. lspci shows only the limited info below (“unknown header type”)

```
07:00.0 Communication synchronizer: Spectracom Corporation TSync-PMC Time Code Processor (rev ff) (prog-if ff)
 !!! Unknown header type 7f
```

- Refer to Salesforce case 11923 (https://na8.salesforce.com/500C000000U0ZmA)
- lspci shows board is still installed and was working fine. Now, it can no longer communicate.
- Likely due to a recent (manual or automatic) update to the OS being applied, but the driver has been rcompiled to the new kernel version yet.
- Need to recompile the driver with a make clean and normal make install.

Thanks for bringing your customer’s observations to our attention.

Please ask your customer if they have performed any recent updates to their system, or if the linux distribution is setup to provide automatic updates to the OS.

When updates are performed, it can affect the header files. We recommend they just recompile their driver with a make clean and make install (they are likely very familiar with how to recompile the driver, but if they need any information on this, just let me know).

Please also let me know the lspci –vv command shows all of the normal info after the driver has been recompiled. I want to make sure that they are all set with communicating with it again (I appreciate it)!

- lsmod commands (Shows driver version and if the driver has been installed correctly)

“lsmod” = Lists the drivers that are currently installed:

Example:

```
Spectracom TSync-PMC Timing Board - version 2.50
Copyright (c) 2009 Spectracom Corporation
tsyncpci 0000:0a:00.0: PCI INT A -> GSI 16 (level, low) -> IRQ 16
TSYNC version of board: tsyncpci0
```

```
[root@vr2a linux]# modprobe tsyncpci
```

Note: The Linux driver should be installed in the ***/Dev*** directory

Example

```
[root@vr2b linux]# ls -al /dev/|grep -i tsync
crw-rw-rw-   1 root root    248,   0 Nov 22 07:34 tsyncpci0
```

“Please make sure the TSync driver is located in the /Dev folder. “TSyncpci 0” should be displayed in this folder.

If /Dev directory does not show: “`tsyncpci0`”

- See if ASPM has been disabled.
- Has the driver been updated from earlier version (leaves behind a Rules file)

X) (ASPM) Power Saver/Power Monitor feature causing TSync board to hang

- **Also refer to document Tim Tetreault created: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync-PMC\ASPM power monitor**

- 4/19/12 KW – Newer kernel versions are enabling a power saver mode on the PCIe bus that may either cause the TSync-PMC boards to hang-up right away or stop running after a short period of time.

Note: Refer to Rich Schmidt with USNO in Salesforce for more info on this kernel “feature”.

Symptoms of ASPM potentionally being an issue include:

- Intermittent system crashes when the reset command is issued (refer to Salesforce case 9560 https://na8.salesforce.com/500C000000Qvoku

- System never being able to communicate with the board.,

- Make sure ASPM is disabled in BIOS

- Follow the ASPM document to make sure ASPM is disabled in the kernel (grub.conf file).

ASPM configuration change may not persist first time its performed
Note that we have heard of instances in the past where the ASPM configuration change didn’t persist the first time through the procedure. This procedure may need to be repeated if it doesn’t take the first time through. To confirm the change did persist, after a system reboot, the response for the “cat /sys/module/pcie_aspm/parameters/policy” command will be “**performance**” as below (bolded for emphasis):

```
[root@gieib19853 ~]# cat /sys/module/pcie_aspm/parameters/policy
default [performance] powersave
```

Note: in RedHat, the line to add to the kernel is in /Boot/Grub and is called **grub.conf**

ASPM issues with CentOS
(KW 3/19/12) Updating from CentOS 5.6 to CentOS 5.7 is one example which can result in abnormal operation with the TSync-PMC board (such as interrupt conflicts), due to the assigned Class Code (programmed into the TSync-PMC board).

CentOS 6.2/ RHEL 6.2/scientific linux 6.2

1) CentOS version 5.8 works fine, but version 6.2 causes the TSync-PMC board to stop responding

Notice: We highly recommend updating to version 6.4, if possible. In March 2013, **CentOS and Scientific Linux released version 6.4**. Version 6.4 has kernel fixes that address some ASPM related issues (We have even noticed here that ASPM not being completely disabled can cause intermittent boot-up issues). The changes in the ASPM document still need to be performed successfully once, but would it be possible to update the system to version 6.4 to see if it completely resolves the system you are observing? If you can update to the newer version, we have noticed that the ASPM changes made in 6.2 persist through the update to 6.4, so they don’t need to be performed again after upgrading to 6.4.

Fix: Refer to “ASPM” in “power saving feature” section below. CentOS/RHEL Version 6.2 enables this feature (for decreased laptop power consumption). We want it disabled, so the TSync-PMC board doesn’t eventually go into a “sleep” mode. Otherwise, there will be lags in functionality, while the board is waking up.

I have some EXCELLENT news for you!! One of our timing board engineers was able to determine what was causing the issues you were observing with the installed timing boards. The issue is not related at all to either the timing boards or the TSync driver. It has to do with a power saver function being used on the PCIe bus incorporated in the newer kernel versions. With a couple of changes he made, he has been running the system with 2 TSync boards installed and NTP running since yesterday with no problems!!!!

Before sending the system back to you, he would like you to try the changes he has made, just to ensure this has fixed the issues you reported. When you get a moment, can you perform the following steps on a similar system and let us know how it goes? This process changes how the machine boots up.

1) Go to the following directory: #cd boot/grub/
2) Edit the following file: "grub.conf"
3) Add the following to the end of the command line: "pcie_aspm=off" (SEE EXAMPLE BELOW)

```
title CentOS (2.6.32-220.7.1.el6.x86_64)
        root (hd0,0)
        kernel /vmlinuz-2.6.32-220.7.1.el6.x86_64 ro root=/dev/mapper/VolGroup-lv_root rd_NO_LUKS
LANG=en_US.UTF-8 rd_NO_MD rd_LVM_LV=VolGroup/lv_Swap SYSFONT=latarcyrheb-sun16 rhgb
crashkernel=128M quiet rd_LVM_LV=VolGroup/lv_root KEYBOARDTYPE=pc KEYTABLE=us rd_NO_DM
pcie_aspm=off
        initrd /initramfs-2.6.32-220.7.1.el6.x86_64.img
```

Once you do this, reboot the system and load the driver.

Another email from Tim Tetreault (4/20/12) about this topic.
There are 2 other things they can do.

1) I updated the BIOS in the machine they set us. Its new BIOS is:
ProLiant BIOS: 026
BIOS Version: 0260414
BIOS Date: 04/14/11

2) Have them check the following settings in his BIOS:
Under "Advanced Chipset Control"
PCIe Gen2 - Gen2
Enable ASPM - DISABLED

Email from Rich about CentOS 6.2
I discovered that in CentOS 6.2 /sys/modules/pcie_aspm/parameters/policy:
default performance [powersave]

I will test setting this to [disable].

In CentOS 5.8 there is no /sys/modules/pcie_aspm

Q. (from Tim Tetreault to Rich Schmidt with USNO)
What did you do to force the OS to "performance" mode on reboots? I found an app note that recommended that we add the following kernel command line parameter:
pcie_aspm.policy=performance
Is that what you did or did you do something different?

- (from Rich) We simply followed your lead and added pcie_aspm=off to the grub.conf. Now each time we upgrade the kernel we must add this! That is what we forgot to do last time.

E) Class Code assigned to the TSync board

- Some sytems, especially newer HP boxes can have problems with our original/generic Class Code
- Updating from CentOS 5.6 to CentOS 5.7 is one example which can result in abnormal operation with the TSync-PMC board (such as interrupt conflicts), due to the assigned Class Code (programmed into the TSync-PMC board). A patch is available to update the class code to a newer value, which helps the OS work with the TSync-PMC.
- For more info on this issue, refer to Salesforce cases for MIT LL (case 4986) and NOAO (case 5048).

Status Update (4 Apr 2013): ECN 3159 was released today to incorporate the Class Code update to all TSync-PMC boards. Firmware version 2.11.

Obtaining the current Class Code assigned to the TSync-PMC board
Perform an lspci (refer to “lspci” section above).
This command should respond with either the original class code or the newer class code. See the **Results** below.

Note: The command will return nothing if the system bus can't see the card.

Results of lspci:

A. Original/Generic Class Code (prior to ~March 2013)

First line of response should be either:

```
nn.nn.n Non-VGA unclassified device: Spectracom Corporation Device 8000

nn.nn.n Non-VGA unclassified device: Unknown device 1ad7:8000
```

(where **nn.nn.n** is the bus ID number and is system-dependent, and **1ad7:8000** is our Vendor and Device ID for the TSync card)

```
04:00.0 Non-VGA unclassified device: Spectracom Corporation TSync-PMC Time Code Processor
        Subsystem: Spectracom Corporation TSync-PMC Time Code Processor
        Control: I/O+ Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr- Stepping- SERR- FastB2B- DisINTx-
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0, Cache Line Size: 256 bytes
        Interrupt: pin A routed to IRQ 16
        Region 0: Memory at fbcfe000 (32-bit, non-prefetchable) [size=512]
        Capabilities: <access denied>
        Kernel driver in use: tsyncpci
```

Note: With the TSync board installed and the driver loaded, the following two lines should be present in an lspci response (likely at the very end of the response).
Kernel driver in use: tsyncpci”
“Kernel modules: tsyncpci”

The command will return nothing if the system bus can't see the card.

Upgrading the firmware to apply the more specific Class Code
Email from Tim Tetreault to a customer (3/16/12)
Here is the patch and instructions on how to **update** your boards to fix the Class Code issue. If you install the card in a system that was having a problem, you should do the following:
Uninstall the current driver. “**rm tsyncpci**”
Run the “**make uninstall**” & “**make clean**” to remove any old driver files.
Reboot the computer to that the system can identify the card with the new class code.
Rebuild the driver and install using the “**modprobe tsyncpci**” command.

In Linux, you can run the following command as root to look for the PCIe card specifically on the PCI

B. Updated Class Code (ECN 3159- released 4 Apr 2013)

- Firmware version 2.11.

- Approximate Manufacturing cut-in Serial Number to have the new class code: 01998

The new class code will now report:
Base Class = 11h (Communications synchronizer) Sub-Class = 10h Interface = 00h

First line of response should be:

```
nn.nn.n 1a:00.0 Communication synchronizer: Spectracom Corporation Device 8000
```

(where **nn.nn.n** is the bus ID number and is system-dependent, and 1ad7:8000 is our Vendor and Device ID for the TSync card)

Note: With the TSync board installed and the driver loaded, the following two lines should be present in an lspci response (likely at the very end of the response).
Kernel driver in use: tsyncpci"
"Kernel modules: tsyncpci"

E) BIOS Settings / Issues with "AMI UEFI core" in BIOS
System's BIOS settings for the PCIe bus may be causing an issue. May need to update or change the BIOS settings.
Email from Alan Leung with DSPcon
Got the issue resolved. Looks like it was a bios setting that needed to be changed from 16x/8x to 8x/8x in the PCI configuration.

AMI UEFI core:
Email from Tim Tetreault (2 Apr 2013) I spent a little time looking on line about the computer they using and found some people reporting issues related to the bios "AMI UEFI core" that this system uses. I have come across other customers in the past that had issue that were caused by BIOS issues and not card or hardware problems.
Ask the customer if they have check to see if there system has a newer BIOS available.

Also, I want the customer to with just the board installed without the driver installed, run "lspci –vv" and send us the output.

F) Issues with specific linux distributions

- Earlier Linux drivers Versions 2.4.1 and 2.3.0 have an issue with the Rules file (Redhat/CentOS or Scientific Linux distributions)

The Rules files are located in /**etc/udev/rules.d**:
Note: Refer to: Versions 2.4.1 and 2.3.0 have an issue with the Rules file (Redhat/CentOS or Scientific Linux distributions) below for more information.

- The correct Rules file for driver version 2.5.0 (or above) is "tsyncpci.rules"
- The older Rules file from the versions 2.4.1 and below drivers (for older kernels) was "tsyncpci 2_6_9". (Note that this Rules file should be deleted, if present with the version 2.50 or higher driver installed)

The Rules files should be located in **etc/udev/rules.d**:

- The correct Rules file for driver version 2.5.0 (or above) is "tsyncpci.rules"
- The older Rules file from the versions 2.4.1 and below drivers (for older kernels) was tsyncpci 2_6_9 (this second rules file should be deleted, if present with the version 2..50 or higher driver installed)

Symptoms: Error message **"!Could not open </dev/tsyncpci0> : rc <1>"** is displayed when trying to communicate with the TSync-PMC board (such as trying to perform a HW_GetTime call or any other call). "The board is plugged in and we see that the lights on the board itself are on (LED lights in front are off). When we do a "lsmod | grep tsyncpci" it shows that the driver is installed".

- Linux Red Hat distribution/ earlier driver being updated to 2.5.0 or higher

Symptom: I believe that I have solved the issue I was seeing. I was getting a permissions problem when attempting to access /dev/tsyncpci0. Every time I rebooted the machine the permission would prohibit me from utilizing the card. So, I edited the ‘/etc/udev/rules.d/25-tsyncpci.rules’ file to read ‘KERNEL==”tsyncpci*”, OPTIONS+=”last_rule”, MODE=”0666”’. And now upon boot the permissions are set so that the card is usable.

(7/06/11) The TSync-PMC Linux driver versions 2.4.1 (and possibly earlier or later) has a known issue when using Linux Red Hat distribution (Red Hat versions 5 and 6 only – not applicable to Red Hat version 4).

A simple modification of the driver is required for the timing board to be recognized by the OS (Before making this simple mod, you can’t communicate with the timing board, as shown with “comm error” being returned, can’t run example programs, etc).

Email I sent to Gigi Mathew and Steven Lockhart:
Regarding the Spectracom TSync-PMC timing board driver issue you are seeing, I have received some additional information from our timing board engineers that will likely resolve this issue! There is a simple change to the driver that needs to be made when using the driver with the Red Hat distribution.

Please note: Before you make this mod, make sure that you uninstall the current driver before recompiling and installing the driver with the mod to the make file!

In the makefile under “tsync/linux/driver”, change the following line:

```
#-------------------------------------------------
# Test if kernel version is 2.6.13 or later
#-------------------------------------------------
ifeq ($(K_VERSION), post)
    RULEFILE:=25-tsyncpci.rules
else
    RULEFILE:=25-tsyncpci_2_6_9.rules
endif
```

Once you have done this, run “make clean”, “make”, “make install” and “modprobe tsyncpci”. Verify with “lsmod” that “tsyncpci” is present.

You should probably remake the examples.

Once you have made this minor change to the driver, please let me know that this mod resolved the issue and that you are now able to successfully run the example programs.

I was discussing this situation with one of our timing board engineers. We believe you may have an earlier version of the TSync-PMC Linux driver currently installed. There is an issue with CentOS (and other Linux distributions) associated with a Rules file. This issue can be fixed by completely removing the earlier version of the driver and then installing the latest version (rmmod of TSync-PCI and then make uninstall). The error message you reported (“could not open…”) indicates the driver can’t talk to the TSync-PMC board. With this Rules issue, everything looks fine, but this symptom occurs.

Attached are the TSync-PMC driver Release Notes. The latest version of the Linux driver is version 2.5.0 (released in November 2011). If you are currently running previous versions 2.4.1 or 2.3.0 removing the driver and then installing the latest will likely resolve this issue. The latest version of the Linux driver can be downloaded at no cost from the Spectracom website. The link to the latest version of the Linux driver is: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=101.

Please let me know that uninstalling the current driver and installing the latest driver resolves this issue for you!!!

Another email sent to a different customer
A question for you- Was an earlier version of the TSync-PMC driver ever installed on this machine? Or, is the version 2.5.0 driver the only TSync driver that has been installed?

If an earlier version of the TSync-PMC driver was ever installed, it leaves behind a Rules file that will prevent the driver from being able to work with the TSync-PMC board. The Rules files are stored in the /etc/Dev folder. The correct Rules file is "tsyncpci.rules". However, if a file called "tsyncpci 2_6_9" is also located in this same directory, this file should be removed.

G) Issues with input power (+3.3vdc and +12vdc)

- Issues with Input power from the PCI bus (especially with +3.3vdc) can hold the board in constant reset.
- Verify Test points for input voltages. Refer to: Power (+3.3vdc +12vdc and -12vdc from PCI bus)/logic

Missing header file reported when earlier linux version 2.3.0 driver is installed

- This is due to a change to a newer kernel version relocating where this header file is placed (so its not able to be found).
- The fix is to update the Linux driver from 2.3.0 to a newer version driver (v2.4.1 or beyond).

Email from Jacques Kronbauer
I am having a problem while building linux driver caused by a missing header file that was supposed to be in file tsync.1.30.tar.gz.

Specifically the problem is when I try to build it by using make, I run into compilation error caused by a missing header file that was supposed to be in file tsync.1.30.tar.gz.

It seems that there should be a directory 'asm' under linux and it should have a file called semaphore.h.

I checked Spectracom's web site for a newest file but could not find it.

The problem is highlighted with ******* below.

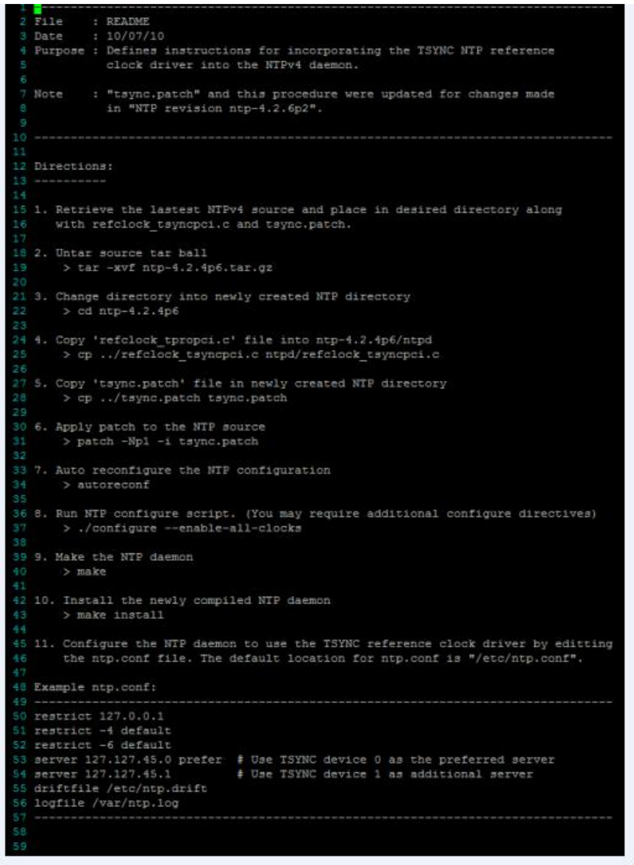
```
make -C /lib/modules/2.6.32.12-0.7-default/build M=/home/DS/kro55737/tsync/tsync/linux/driver modules
make[2]: Entering directory `/usr/src/linux-2.6.32.12-0.7-obj/x86_64/default'
make -C ../../../linux-2.6.32.12-0.7 O=/usr/src/linux-2.6.32.12-0.7-obj/x86_64/default/. modules
  CC [M]  /home/DS/kro55737/tsync/tsync/linux/driver/tsync_drv_2_6_13_1.o
In file included from /home/DS/kro55737/tsync/tsync/linux/driver/tsync_drv_2_6_13_1.c:66:
****** /home/DS/kro55737/tsync/tsync/linux/driver/tsyncpci.h:29:27: error: asm/semaphore.h: No such file or directory
*****
```

My response, with assist from Tim Tetreault (9/18/12 KW)
This header file issue is resolved by upgrading the TSync linux driver to the latest version, version 2.4.1. In summary, this issue is due to a change in the location of the header file in newer kernel versions.

Attached is a copy of the TSync-PMC driver version 2.41 release notes. The first paragraph provides the link to download the latest version of the driver, at no cost. Also attached is the latest version of the TSync-PMC driver guide. Refer to Section 2.1.2 (page 2-2) for information on uninstalling the current driver and then refer to Section 2.1 (starting on page 2-1) for information on installing this new driver.

**Desire to synchronize a Linux PC to an installed TSync-PMC board:

- Refer to the README file contained in the Linux driver (TSync/linux/NTP folder) for more details.



```
--------------------------------------------------------------------------------
File     : README
Date     : 10/07/10
Purpose  : Defines instructions for incorporating the TSYNC NTP reference
           clock driver into the NTPv4 daemon.

Note     : "tsync.patch" and this procedure were updated for changes made
           in "NTP revision ntp-4.2.6p2".

--------------------------------------------------------------------------------

Directions:
----------

1. Retrieve the lastest NTPv4 source and place in desired directory along
   with refclock_tsyncpci.c and tsync.patch.

2. Untar source tar ball
    > tar -xvf ntp-4.2.4p6.tar.gz

3. Change directory into newly created NTP directory
    > cd ntp-4.2.4p6

4. Copy 'refclock_tpropci.c' file into ntp-4.2.4p6/ntpd
    > cp ../refclock_tsyncpci.c ntpd/refclock_tsyncpci.c

5. Copy 'tsync.patch' file in newly created NTP directory
    > cp ../tsync.patch tsync.patch

6. Apply patch to the NTP source
    > patch -Np1 -i tsync.patch

7. Auto reconfigure the NTP configuration
    > autoreconf

8. Run NTP configure script. (You may require additional configure directives)
    > ./configure --enable-all-clocks

9. Make the NTP daemon
    > make

10. Install the newly compiled NTP daemon
    > make install

11. Configure the NTP daemon to use the TSYNC reference clock driver by editting
    the ntp.conf file. The default location for ntp.conf is "/etc/ntp.conf".

Example ntp.conf:
--------------------------------------------------------------------------------
restrict 127.0.0.1
restrict -4 default
restrict -6 default
server 127.127.45.0 prefer  # Use TSYNC device 0 as the preferred server
server 127.127.45.1         # Use TSYNC device 1 as additional server
driftfile /etc/ntp.drift
logfile /var/ntp.log
--------------------------------------------------------------------------------
```

link to the README file: (1191-5003-6001, in Arena at https://app.bom.com/items/detail-spec) (Navigate to the "ISS Source" folder and then to the "linux" folder (in the .tar file, go to tsync/linux/ntp)

The TSync-PMC board can be used to sync the Linux machine it's installed in. The Linux driver includes a patch to apply to NTP running on that machine and the instructions on how to do this. After they compile the new NTP, they have to run NTP from the new patched NTP and not the original compile of NTP.

**Also, refer to Section 2.1 of the Unix Ap Note: I:\Customer Service\Software\UNIX. There are limitations to being able to use the timing board as an NTP reference, as described in this other document (and below).

Answer: In addition to installing the Linux driver, the customer needs to download, install and compile NTPv4 software onto the PC that the TSync-PMC board is installed in. The NTPv4 software is a freeware program that is available. To obtain the software and instructions on installing/building the software, have them visit http://www.ntp.org/

After they install NTP, the Linux driver contains a README file that explains how to patch the NTP software in order for the NTP software to use the TSync board as an available NTP reference clock driver to sync NTP. Once the patch is applied, the "patched" NTP software can be run to sync the PC.

(3/2/11 KW) Our current reference table (part of the NTP patch included with the linux driver) does not allow NTP to sync to the board if the board is synced with **Self/EPP0** (the board can declare sync with this, but NTP won't sync to the board). The minimum requirement for NTP to sync to the TSync-PMC board is **HST1/EPP0** (so that SOME user interaction is necessary in order for NTP to just sync to whatever the board powers up with for the time).

Issues with compiling NTP in SUSE linux

- TimTetreault found issues with compiling NTP is SUSE Enterprise linux , even before installing our patch. Not an issue with our code.
- Refer to Salesforce Case 8916 for more info on this customer: https://na8.salesforce.com/500C000000Pvj7X

Email from Tim Tetreault to Leisa Butler
I setup a computer with the version of SLES the customer told us they were running. Then I installed our TSync board with driver. That works just fine. Then I loaded 3 different versions of NTP. I tried our patch and that works ok but when I try to compile them, I start getting errors. I removed our patch and tried just compiling the NTP source. I got the same errors.

I started looking in to what the errors were and it seem to be issues related to missing or old header files that are required for NTP to compile correctly.

I don't think the problem is our driver or patch but a resource issue with the OS and the NTP source. I think if we figure out what the resource issue is, it will resolve the NTP issue and our board will work just fine.

I have it as a task to see if I can fine a fix but I don't have a time that I can give you on when I will have a fix.

SLES come with full support for customer that pay for it so you might let the customer know what we have found so they can try to use their Linux support. SUSE Linux might have a fix.

Troubleshooting NTP patch not syncing the system

If the linux kernel can't sync to the TSync-PMC board:

- Make sure the green Sync LED is lit (TSync-PMC board needs to be synced).

Note: You can also perform an **SS_GetSync 0** API call/example program to remotely confirm status (it should report "True", if it's in sync)

- Verify the TSync-PMC board and driver can successfully tak to the system by running example programs or API calls. Perform a HW_GetTime 0 API call/example program to both verify nomal operation of the TSync board and to verify the time/date information is correct.

  (HW_Gettime responds with: Year, DOY, Hr, Min Sec, nanosecond, Sync status**)**
- Make sure that all of the steps in the NTP README file are being performed.
- Make sure the version of NTP installed is the one that is specified in the README file. Installing a different version of NTP is not supported and may require customer modify the patch to get it to work

“**Couldn't initialize device**".
If this error is reported, follow the troubleshooting steps above to diagnose.

NTP Reach value not incrementing

Once they have it running, there is a known issue with the NTP Reach value not incrementing. According to Tim Tetreault on 11/4/11. This is a known issue that appears to be NTP

Update Note (5/30/12): Tim Tetreault informed me yesterday the NTP User’s group found what was causing this minor issue to occur. The fix for this issue will be included in the next release of the TSync-PMC Linux driver, as it requires a change to the clock driver software contained in the driver (not a change to NTP)

KW- Email I sent to Richard Schmidt about this (11/4/11) Thanks for your reply and reporting your findings to us. I really appreciate it!

We are aware of the NTP Reach counter not incrementing correctly and have looked into this reporting error already. Other than the Reach value not incrementing, NTP is fully functional. As far as we were able to determine with our investigation, this is a factor of NTP itself and not a factor of the Spectracom driver.

I am going to flag your record in our database and will let you know if we are able to resolve this minor indication error in the future, with either a newer version of NTP or other work-around.

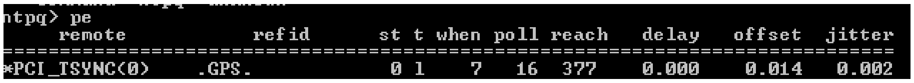
Measuring how well the TSync-PMC board is maintaining the kernel time (using the NTPQ peers and ntpdate commands)

- Can use the NTQ pe (peers command) from NTP running on the linux box to report the offset between the tsync board and the kernel time.
- Can also use the ntpdate –d command to compare the kernel time to other machines.

Email Keith sent to a dealer for his customer (12 Dec 2013) The NTP software running on his machine has two available “functions” called NTPQ and NTPDC that are used to query NTP for information. Specifically, NTPQ has a “peers” command (ntpq -p) which reports the offset between the Linux kernel time versus other time references (such as the TSync-PMC board for example).

He can use the peers command to see how closely aligned the kernel time is to the TSync board. And, if he has any other NTP servers on the network with this machine or can point to any Internet Time servers, he can specify the peers command to compare the time against them as well. He can use this command without NTP syncing to the timing board (as compared to other references) and then run the command again while the kernel is being synced by the TSync board.

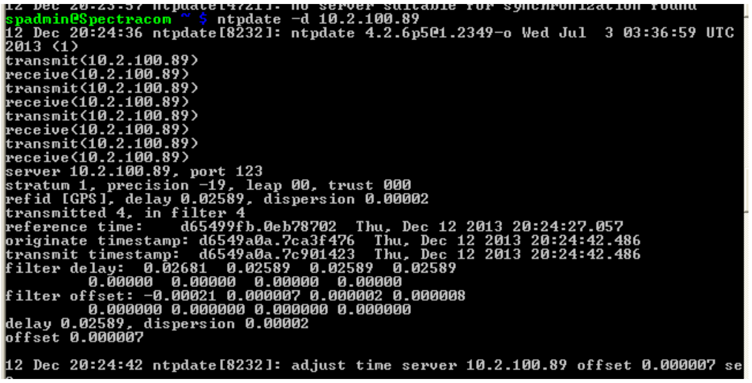
Below is an example response from a peers command, showing the time offset between the TSync-PMC board and the kernel time (in this example, its 14 microseconds of offset).



```
ntpq> pe
     remote           refid      st t when poll reach   delay   offset  jitter
==============================================================================
*PCI_TSYNC(0)    .GPS.            0 l    7   16  377    0.000    0.014   0.002
```

For more information on NTPQ peers command, NTPDC and the **ntpdate –d** command (another command he can use to compare the kernel time to other references, note the –d runs the command in debug mode so that its not actually setting the time), please refer to sites such as: http://www.eecis.udel.edu/~mills/ntp/html/ntpq.html and http://www.novell.com/coolsolutions/trench/418.html.

Below is an example response from the **ntpdate –d** command comparing the offset between the NTP server (10.2.100.89) versus the kernel time of this machine. In this example, the offset is 7 microseconds (from the last line of the response).



```
12 Dec 20:23:57 ntpdate[4721]: no server suitable for synchronization found
spadmin@Spectracom ~ $ ntpdate -d 10.2.100.89
12 Dec 20:24:36 ntpdate[8232]: ntpdate 4.2.6p5@1.2349-o Wed Jul  3 03:36:59 UTC
2013 (1)
transmit(10.2.100.89)
receive(10.2.100.89)
transmit(10.2.100.89)
receive(10.2.100.89)
transmit(10.2.100.89)
receive(10.2.100.89)
transmit(10.2.100.89)
receive(10.2.100.89)
server 10.2.100.89, port 123
stratum 1, precision -19, leap 00, trust 000
refid [GPS], delay 0.02589, dispersion 0.00002
transmitted 4, in filter 4
reference time:    d65499fb.0eb78702  Thu, Dec 12 2013 20:24:27.057
originate timestamp: d6549a0a.7ca3f476  Thu, Dec 12 2013 20:24:42.486
transmit timestamp:  d6549a0a.7c901423  Thu, Dec 12 2013 20:24:42.486
filter delay:  0.02681  0.02589  0.02589  0.02589
         0.00000  0.00000  0.00000  0.00000
filter offset: -0.00021 0.000007 0.000002 0.000008
         0.000000 0.000000 0.000000 0.000000
delay 0.02589, dispersion 0.00002
offset 0.000007

12 Dec 20:24:42 ntpdate[8232]: adjust time server 10.2.100.89 offset 0.000007 se
```

**Real-time (RT) linux (MRG Linux)

- Refer to case 00002331 for Andrew Pearson with Raytheon)
- (See also newer, similar instance of this- Billy Guan (also from Raytheon). Refer to Salesforce case 00003436)
- Will likely need to work with Redhat directly to resolve issues related to the Real Time patch preventing our driver from being installed/able to run (Likely to see a fatal error when running modprobe – more on this further below)

I spoke to our primary timing board engineer. Everything on the Spectracom side looks fine. He recommends that you contact Redhat directly regarding how the RT patch affects the kernel versioning.

We have had a few other customers have the exact same problem until they worked with Redhat directly to resolve some issues associated with this Real Time patching. And if you are paying for Enterprise edition of Redhat, this support is provided by Redhat as part of this purchase. Just let them know you are working with the Spectracom TSync-PMC driver and have the RT patch installed. They can help you with it from there.

You shouldn't need anything else from us. After working with Redhat to resolve the kernel versioning, you should be all set! But if you happen to need anything else from us as you work with Redhat, or anytime thereafter, please let me know!

A runtime patch ("rt") can be installed onto a standard non-real time linux, to make Linux real-time. The problem this causes with our linux driver is installing this "rt" patch results in two kernel versions being created, instead of just one version, (the earlier un-patched version still exists and the newer patched version is also generated). The build command builds our driver to the earlier un-patched version of the kernel, but when they try to load the driver, it's against the newer version, which isn't the same as it was built for, so an "invalid module format error message is displayed.

The resolution to this issue is for the customer to contact Red Hat directly to assist them with installing a non-real time driver (our driver) onto a patched real time linux OS.

"Invalid module format" error message displayed while trying to install the Linux driver onto a machine that has had this runtime patch installed

Q. (1/20/11) Can you just confirm that whether or not your driver is compatible with Linux real-time OS (MRG)? (Customer – Andrew Pearson from Raytheon - was running 3.6.33.7 MRG)
A. **Reply from DL**- I had to check on this and found we had tested a Red Hat real time setup about a year ago and it did work. It was on an earlier version/patch of Red Hat though.

Email to Billy Guan on 9/26/11 (based on research/info received from Dave Sohn)
Note: Refer to the PDF document that was sent with this email - EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync-PMC\MRG real-time Linux

I wanted to pass along some information to you, regarding how to make your MRG Linux compatible with the Spectracom TSync-PMC driver. For your reference, the information below and attached was obtained from an errata report from Red Hat, against their MRG 2.0 patch.

* A missing directory in the kernel-rt-devel packages, which contains header files, prevented third-party modules from being built using only the kernel-rt-devel package. Adding these missing files now allows third-party modules to be successfully built. (BZ#718940).

The fix for this Red Hat issue updates the kernel version to 2.6.33.9-rt31.74.el6rt.x86_64. Please also make sure you have installed the kernel-rt-devel package, which is necessary for module builds. Once you have added the missing Red Hat files, try compiling/running the driver again. Please let us know that adding the files resolved the issue you initially reported!

“modprobe tsyncpci” = rebuilds and installs the driver

Modprobe failures

- “Invalid module format”

Refer to **Real-time linux (MRG Linux)

- Real Time (RT) patch is installed.
- Tim Tetreault recommended they contact Redhat directly to resolve issues with the Real Time patch.

```
            root@pascal linux]# pwd
/hadas/install/tsync/linux
[root@pascal linux]# make clean > ./clean.log
[root@pascal linux]# make  > ./make.log
ar: creating libtpro.a
[root@pascal linux]# make install > ./install.log
[root@pascal linux]# modprobe tsyncpci > ./modprobe.log
FATAL: Error inserting tsyncpci (/lib/modules/3.6.11.2-rt33.39.el6rt.x86_64/kernel/drivers/tsyncpci.ko): Invalid module format
```

I’ve checked that the tsyncpci.ko is regenerated:

```
[root@pascal linux]# date
Mon Oct 14 20:46:48 CEST 2013
[root@pascal linux]# ll /lib/modules/3.6.11.2-rt33.39.el6rt.x86_64/kernel/drivers/tsyncpci.ko
-rw-r--r-- 1 root root 800990 Oct 14 20:45 /lib/modules/3.6.11.2-rt33.39.el6rt.x86_64/kernel/drivers/tsyncpci.ko
```

Solaris driver

Shortcut to TSync-PMC driver release notes: I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\TSync driver updates

prtconf -D example

```
System Configuration:  Sun Microsystems  i86pc
Memory size: 131045 Megabytes
System Peripherals (Software Nodes):
i86pc (driver name: rootnex)
    scsi_vhci, instance #0 (driver name: scsi_vhci)
    pci, instance #0 (driver name: npe)
        pci8086,3c00
        pci8086,3c02, instance #0 (driver name: pcieb)
            pci8086,1d74, instance #5 (driver name: pcieb)
                pci8086,1d3f, instance #6 (driver name: pcieb)
                    pci8086,1d68
                    pci8086,1d70
                    pci8086,1d71
        pci8086,3c03, instance #1 (driver name: pcieb)
            pci15d9,1521, instance #0 (driver name: igb)
            pci15d9,1521, instance #1 (driver name: igb)
        pci8086,3c04 (driver name: pcieb)
        pci8086,3c08 (driver name: pcieb)
        pci8086,3c0a (driver name: pcieb)
        pci8086,3c28
        pci8086,3c2a
        pci8086,3c2c
        pci15d9,628
        pci15d9,628
        pci15d9,628, instance #0 (driver name: ehci)
            hub, instance #1 (driver name: hubd)
                device, instance #0 (driver name: usb_mid)
                    mouse, instance #0 (driver name: hid)
                    keyboard, instance #1 (driver name: hid)
        pci15d9,628, instance #1 (driver name: ehci)
            hub, instance #0 (driver name: hubd)
                device, instance #2 (driver name: usb_mid)
                    keyboard, instance #4 (driver name: hid)
                    mouse, instance #13 (driver name: hid)
        pci8086,244e, instance #0 (driver name: pci_pci)
            display, instance #0 (driver name: vgatext)
        isa, instance #0 (driver name: isa)
            motherboard
            asy, instance #0 (driver name: asy)
            asy, instance #1 (driver name: asy)
            motherboard
            motherboard
            motherboard
            pit_beep, instance #0 (driver name: pit_beep)
        pci15d9,628, instance #0 (driver name: ahci)
            disk, instance #0 (driver name: sd)
            disk, instance #1 (driver name: sd)
            disk, instance #2 (driver name: sd)
            disk, instance #3 (driver name: sd)
        pci15d9,628
    ioapics
        ioapic, instance #0
        ioapic, instance #1
    pci, instance #1 (driver name: npe)
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,3ca0
        pci8086,3c46
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
        pci8086,0
    pci, instance #2 (driver name: npe)
        pci8086,3c01 (driver name: pcieb)
        pci8086,3c02, instance #8 (driver name: pcieb)
            pci8086,3, instance #0 (driver name: ixgbe)
            pci8086,3, instance #1 (driver name: ixgbe)
        pci8086,3c04, instance #9 (driver name: pcieb)
            pci10b5,8617, instance #11 (driver name: pcieb)
                pci10b5,8617, instance #12 (driver name: pcieb)
                    pci10ee,7, instance #0 (driver name: gandalf)
                pci10b5,8617, instance #13 (driver name: pcieb)
                    pci1823,1770, instance #0 (driver name: focalpoint)
                pci10b5,8617, instance #14 (driver name: pcieb)
                    pci1b65,abba, instance #0 (driver name: xel_hx)
            pci8086,3c08, instance #3 (driver name: pcieb)
                pci1ad7,8000, instance #1 (driver name: tsync)
            pci8086,3c28
            pci8086,3c2a
            pci8086,3c2c
        fw, instance #0 (driver name: acpinex)
            sb, instance #1 (driver name: acpinex)
                socket, instance #2 (driver name: acpinex)
                    cpu, instance #0 (driver name: cpudrv)
                    cpu, instance #1 (driver name: cpudrv)
                    cpu, instance #2 (driver name: cpudrv)
                    cpu, instance #3 (driver name: cpudrv)
                    cpu, instance #4 (driver name: cpudrv)
                    cpu, instance #5 (driver name: cpudrv)
                    cpu, instance #6 (driver name: cpudrv)
                    cpu, instance #7 (driver name: cpudrv)
                    cpu, instance #8 (driver name: cpudrv)
                    cpu, instance #9 (driver name: cpudrv)
                    cpu, instance #10 (driver name: cpudrv)
                    cpu, instance #11 (driver name: cpudrv)
                socket, instance #3 (driver name: acpinex)
                    cpu, instance #12 (driver name: cpudrv)
                    cpu, instance #13 (driver name: cpudrv)
                    cpu, instance #14 (driver name: cpudrv)
                    cpu, instance #15 (driver name: cpudrv)
                    cpu, instance #16 (driver name: cpudrv)
                    cpu, instance #17 (driver name: cpudrv)
                    cpu, instance #18 (driver name: cpudrv)
                    cpu, instance #19 (driver name: cpudrv)
                    cpu, instance #20 (driver name: cpudrv)
                    cpu, instance #21 (driver name: cpudrv)
                    cpu, instance #22 (driver name: cpudrv)
                    cpu, instance #23 (driver name: cpudrv)
        used-resources
        iscsi, instance #0 (driver name: iscsi)
        fcoe, instance #0 (driver name: fcoe)
        options, instance #0 (driver name: options)
        pseudo, instance #0 (driver name: pseudo)
        agpgart, instance #0 (driver name: agpgart)
        xsvc, instance #0 (driver name: xsvc)
```

/dev/ (partial example)

tsync0   ("tsync" should be listed once for each installed TSync-PMC board)

Solaris driver version 2.4.1 DOES support PTP calls

(7 Mar 2013) Denis and Matt thought they found an issue with driver version 2.41 not supporting PTP calls. Update from Denis: No, we fixed that at some point, we just didn't realize it at the time.
The Driver datasheet (which we used as the basis for this in the first place) is incorrect.
Our Solaris driver is a few versions old, but includes the PTP calls.

I will find out if Tim K ever found out about this and can update the datasheet

Q. Scott would like to know if you have a driver for the TPRO-PCI-U-2 that supports Trustin Solaris 8
A. (email from Tim Tetreault 7/28/11)
Keith, I think we have had other customers using Solaris 8 but I am not 100% sure. We don't have a system setup here with Solaris 8 so I can't test it to make sure but I think they should be ok.
You can point them to the link on our web site where they can down load it to try.
Q. Can the card be used with a PowerPC Board (P2020 by Freescale) or just on a x86 Platform.
A. (From Tim Tetreault 6/21/12) The card has been used on PowerPC machines running Solaris but I am not sure about Linux. Are they running a version of Linux on the PowerPC machine?

Solaris driver support for more than one installed TSync board in a
Email from Tim Tetreault to Matt Loomis (11 Mar 2013) I did a setup this weekend with Solaris and 2 TSync cards, one of which was a PTP card, and both worked fine. So we know the driver will support multiple cards and works fine in Solaris.

**Troubleshooting

A) Two or moreTSync-PMC boards installed, but only one being detected

Troubleshoot: Try swapping the known good board with one that is not being recognized (put it in the slot that is recognizing the board being installed). If the other board is recognized, the board is OK. It's a system issue preventing the other board(s) from being detected.

Y) PowerPC

Q. Can the card be used with a PowerPC Board (P2020 by Freescale) or just on a x86 Platform.
A. (From Tim Tetreault 6/21/12) The card has been used on PowerPC machines running Solaris but I am not sure about Linux. Are they running a version of Linux on the PowerPC machine?

In summary

A "**makefile**" file ("mkDriver") file is supplied in the **/tsync/examples** directory of the Linux driver. A "make" command is performed to build the examples. The "makefile" can be either used as-is or the customer can modify it as necessary (potentially breaking the example programs if it's incorrectly edited).

The path to the makefile ("mkDriver") is C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API\Src.

The example programs were initially compiled for 32 bit. In order to use them with 64 bit, all of the examples need to be rebuilt for 64 bit.

The make file needs to be modified to use 64 bit examples. The path to the makefile "mkDriver" is C:\Program Files\Spectracom\TSYNC PCI\Examples\TSync API\Src. Refer to Section 2.1.3 (page 2-2 of the attached TSync-PMC driver guide).

Once the examples are built for 64 bit, they should work fine for you.

Desire to use VMWARE (virtual OS/virtual machine)

VMWare allows a “Virtual OS” (such as Linux –known as the Guest OS) to be able to run on top of another OS (such as Windows- known as the Host OS). The problem with this configuration is there is limited capabilities for the Guest OS to be able to access the Host OS’s hardware devices (i.e. the Linux driver on the guest OS can’t talk to the TSync-PMC board). In summary – this can’t work with two custom drivers that the customer would have to create. See the email below that was sent by Keith (12/21/10):

Regarding the configuration of using VMWare, I needed to work with one of our timing board Engineers to get some additional information about this. This is the first time we've seen this desired configuration being attempted. The driver install appears to be fine, but the guest OS is not able to directly communicate with the Host's PCI bus. This is a limitation of using VMWare and not a limitation of the TSync-PMC board itself.

We found the following link that provides additional information about the limitations of using VMware to communicate with a hardware device on the Host OS http://communities.vmware.com/thread/89286. In summary of these findings, it doesn't look like there is a way to allow access to the PCI bus through the virtual OS. The Virtual OS is intended to abstract the hardware for the most part, with a few exceptions for generic operation (such as networking, hard drives, sound, video, etc).

To make this configuration plausible, you would probably need to create some sort of additional custom driver that would abstract the hardware properly to the virtual OS, and then use another custom driver within the virtual OS to talk to that. We're not even sure if this is possible or not. However, we can say that the Spectracom drivers wouldn't be able to provide this capability (just installing the Windows driver wouldn't allow the Linux driver to be able to access the PCI bus because the drivers are not designed to inter-operate with each other). We can provide the source code for the drivers, but this is something that you would have to create, if its possible.

Besides creating your own inter-operating custom drivers, our recommendations is to either install the latest version of the Windows driver (the latest version of this driver adds support for Windows 7) and perform the example programs provided in the Windows driver or to install the timing board into a true Linux machine and use the linux driver to perform example programs or API calls, bypassing the use of the Virtual OS altogether.

For your information, if you would like to use the board in a Windows environment, the Windows driver can be downloaded from our website. Please visit:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=37.
Besides example programs and API calls via the driver, the Windows driver also provides a control utility that allows manual control of many of the TSync-PMC board's functions (Start/All Programs/Spectracom Corp/TSync PCI).

Interrupt generation/Interrupt Counter

Refer to the TSync-PMC driver guide for details (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121

Refer to the following document from another company which discusses Linux interrupt performance: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TSync-PMC\Interrupts

Interrupt descriptions:

From the TSync-PMC manual
5.9 Interrupts
The host bus has one interrupt line available from the TSync-PMC. All interrupt sources destined for the host bus are multiplexed on the single interrupt line. All interrupts are masked on startup, but can be unmasked using the host bus interrupt mask register. Whether an interrupt is masked or not, the current state of the interrupt is available by reading the Host Bus Interrupt Status register. All interrupt sources are latched based on an edge transition. All interrupts are cleared in the host bus interrupt status

Note: The five interrupts below that are in red are only used when a customer is creating their own driver, instead of using a Spectracom-provided driver. The ones in black can be used with the Spectracom driver.

- **1PPS Received**

  This interrupt is driven on the incident edge of the PPS.

- **Timing System Service Request**

  This interrupt is used by the micro to request attention from the local bus.

- **Local / µC Bus FIFO Empty**

  This interrupt is driven when the FIFO from the local bus to the microcontroller bus becomes empty. It is based on the rising edge of the FIFO's empty flag.

- **Local / µC Bus FIFO Overflow**

  This interrupt is driven when the FIFO from the local bus to the microcontroller bus is overflowed. It is based on the rising edge of the FIFO's overflow flag.

- **µC / local bus FIFO Data Available**

  This interrupt is driven when the FIFO from the microcontroller bus to the local bus is no longer empty. It is based on the falling edge of the FIFO's empty flag.

- **µC / local bus FIFO Overflow**

  This interrupt is driven when the FIFO from the microcontroller bus to the local bus is overflowed. It is based on the rising edge of the FIFO's overflow flag.

- **GPIO Input x Event (GPI)**

  This interrupt is driven when the active edge of the GPIO input signal is received.

Note: For more info on using the GPI pins to generate interrupts, refer to: Information about the GPI pins (General Purpose Inputs- "GPIO")

- **GPIO Output x Event (GPO)**

The interrupt is driven when an event occurs in the GPIO output. An event depends on the mode of operation of the GPIO output. In **Direct mode**, an event is triggered when the output Value in the GPIO output control / status register is changed and creates the active edge selected by the GPIO direct mode output interrupt active edge bit in that same register. This can be used to generate a “software” interrupt by setting the GPIO output appropriately. In **match time** mode, an interrupt is generated whenever the GPIO output high match time or GPIO output low match time registers are enabled and subsequently matched against the current system time. In **square wave** mode, an interrupt is generated whenever the GPIO output generates the active edge as selected by the GPIO output square wave active edge bit in the GPIO output control / status register. This can be used to generate a periodic interrupt at the rate of the square wave.

Note: For more info on using the GPO pins to generate interrupts (using either Match time or Square wave), refer to: Information about the GPO pins (General Purpose Outputs- “GPIO”)

- **Time Stamp Data Available**

  This interrupt is driven when the time stamp FIFO goes non-empty. Time stamp data is available in the time stamp FIFO when this interrupt occurs.

Linux kernel thread priorities for interrupt generation

Question from Loredan Neagu (1/19/12) I would like to ask you what is the thread priority at TSYNC driver level for the interrupt generation?
Reply from Dave Sohn: We do not change any default thread priorities of the kernel.

Enabling interrupts:

The best reference for Interrupt operation is section 5.1 of the TSync-PMC driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121

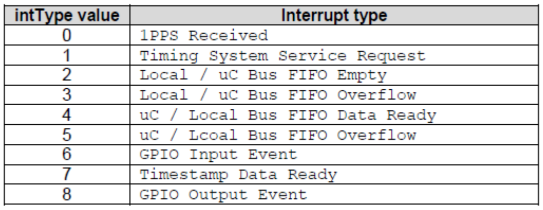
- Interrupts are **enabled** by **disabling the Interrupt mask** of the desired interrupt type (intType).
- Interrupts are **disabled** by **enabling the Interrupt mask** of the desired interrupt type (intType).

TSYNC_**HW_SetIntMask** (TSYNC_HW_SetIntMask 0 intType index bEnable) (see breakdown below)
API call used to enable/disable interrupts of a specific type (such as GPI, GPO or 1PPS)

Example: HW_SetIntMask 0 8 0 0   (Disable GPO output interrupts mask, which enables the interrupts)

intType:
Use the desired value in the following table, based on desired reason for interrupts

Note: Interrupt Types 2 thru 5 are only used if creating a custom driver. They aren’t used when using a Spectracom driver.



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

Index: The interrupt index info, used for certain interrupts (such as a 0 for GPO 0, for instance)

bEnable
The **en** variable on this line should be set to **0** (FALSE) in order to disable the interrupt mask, thereby enabling the interrupt.

The **en** variable on this line should be set to **1** (TRUE) in order to enable the interrupt mask, thereby disabling the interrupt.

Email from Tim Tetreault
To use the GPIO to generate an interrupt, do the following:

- Setup Pin0 to a “square wave”:
- ./GO_SetMode 0 0 2
- Enable Pin0 output:
- ./GO_SetEnable 0 0 1

Desire to generate a 1PPS interrupt (Interrupt generated at the top of each system PPS)

To generate a 1PPS interrupt, perform one of the following:

- Configure a GPO to output 1Hz square wave
- Enable the “1PS Received” interrupt (the easier of the two methods)

Email Keith sent to a customer (to enable the 1PPS received interrupt)
An interrupt can be generated at the top of each second. This particular “interrupt generator” is known as the “**1PPS Received**” interrupt (designated by “intType value “0”). Attached for your reference is a copy of the TSync-PMC driver guide. Section 5.1 (starting on page 5-238) discusses how to enable interrupts to be generated for different situations, such as each time the system PPS occurs.

In summary of the Driver guide, generating this interrupt consists of **disabling** the interrupt mask for “1PPS Received” (all interrupt masks are **enabled** by default). To disable the 1PPS Received mask (thereby enabling the “1PPS Received” interrupt), use the following call: **HW_SetIntMask 0 0 0 0** (where the first 0 is for the first/only TSync board installed in the system, the second 0 is for the “1PPS Received” interrupt, the third 0 is the index value and the last 0 disables the interrupt mask).

With the 1PPS interrupt now being generated at the very top of each second, one available option is to use the available “**waitFor**” API blocking call, to halt the application software until the “1PPS Received” interrupt has occurred. The API call to wait for this interrupt before the application software proceeds in **./HW_WaitforInt 0 0 0** (where the first 0 is for the first/only TSync board installed in the system, the second 0 is to wait for the “1PPS Received” interrupt and the third 0 is the index value).

Interrupt Counter

- Use HW_clrIntCnt to reset the Interrupt counter
- Use HW_GetIntCnt to read the Interrupt counter

Email from Tim Tetreault (Where “8” is for interrupts based on the GPO pin)
Before enabling the interrupt, reset the interrupt counter by using the command:
./HW_ClrIntCount 0 8 0

When an interrupt has been seen, they can us the following command to read the interrupt count:
./HW_GetIntcount 0 8 0

---

waitFor (blocking call)

waitfor is a call/example program used to stop application software from running, until a specified type of interrupt has occurred. (such as to stop the application software, until an event timestamp has occurred, for instance).

Note about "TSync_waitFor" versus "HW_WaitfForInt":

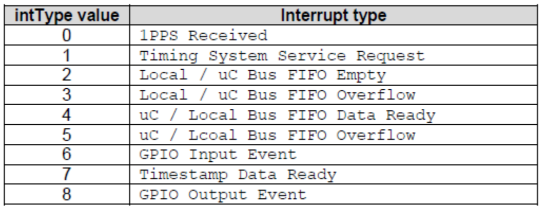
Email from Dave Sohn 1/18/12

- TSYNC_waitFor is the driver API call.
- HW_WaitForInt is the example program, which shows how to use TSYNC_waitFor

TSYNC_waitFor (TSYNC_waitFor 0 <intType> <index>) (see breakdown below)
Blocking call to wait for specified interrupt (such as GPI, GPO or 1PPS)

Example: TSYNC_waitFor 0 8 0   (Call in your application software to wait for an interrupt on GPIO "0").

<intType>
 Use the desired value in the following table, based on desired reason for interrupts



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

<Index>: The interrupt index info, used for certain interrupts (such as a 0 for GPO 0, for instance)

Email from Tim Tetreault (Where "8" is for interrupts based on the GPO pin)
Once the interrupt is running, they can wait for it using the command:
./HW_WaitforInt 0 8 0

Waitfor FAQs

Q. ("**Waitfor" Timeout**) I just thought of another TSync question: if I call the TSYNC_waitFor routine in the driver, it blocks until an interrupt of the specified type occurs. Is there any way to cancel the call? There is no timeout, so this routine has the possibility of causing the calling thread to hang if the interrupt never occurs.

A Excellent question! I hadn't thought about this one before, but it certainly makes sense to think about this condition possibly happening. Though it's not mentioned in our documentation, there is a built-in time-out in this API call to prevent this condition from occurring. The time-out value is based on a set number of Operating System clock ticks, so it's not necessarily an exact, defined amount of time from one PC to the next, but a time-out does exist. The timeout is pre-defined as 100,000 OS ticks.

The waitFor call is one of the example programs included with the TSync-PMC driver. If you wanted, you could run the example program to see that it does time-out after a period of time and it would also give you an idea how long of a duration 100,000 OS ticks would be. This pre-defined value is not adjustable with an API call. However, if it needed to be modified (shortened or lengthened) for your particular application, it can be modified. Just let us know and we can give you additional information on how to modify it.

A1. Response back from customer: Doing the subtraction, that's ~99.9879 seconds. A second test was 99.9878 seconds. Sounds like 100 to me.

Example scenarios for using interrupts

Note: Refer to Example below for:

- **A**= Interrupt to occur once-per-second
- **B=** Interrupt to occur at a specified frequency (such as every 500ms, 1kHz, 1kHz, etc).
- **C=** Interrupt to occur when time stamps are taken

A) Desire to generate a interrupt once-per-second(coincident with the on-time point)

Attached are copies of both the TSync manual and driver guide, which discuss interrupt generation. There is a 1PPS interrupt available. It just needs to be unmasked. These two documents discuss this capability. Refer to Section 5.1 of the driver guide and Section 5.9 of the manual for information on interrupt generation.

FYI:

- Interrupts are enabled by disabling the Interrupt mask of the desired interrupt type (intType).
- Interrupts are disabled by enabling the Interrupt mask of the desired interrupt type (intType)

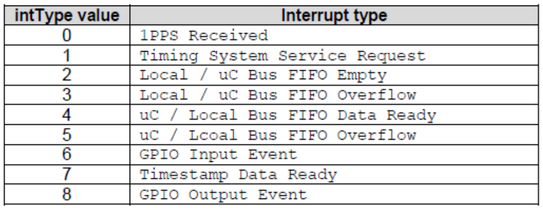
The API call/example program used for interrupt generation is: **HW_SetIntMask.**

The actual call to generate an interrupt each second is as follows: **HW_GetIntMask 0 0 0** <enter> (as broken down below).

HW_GetIntMask <board handle> <intType> <index> <benable>

intType

The "intType" value in this command is selected from the following table. (Value "0" is selected for a once-per-second interrupt to occur).



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

bEnable

The **en** variable in this command should be set to **0** (FALSE) in order to disable the interrupt mask, thereby enabling the interrupt.

The **en** variable in this command should be set to **1** (TRUE) in order to enable the interrupt mask, thereby disabling the interrupt.

waitFor blocking call

If desired, there is an available blocking call that can be used to halt the application code, until this interrupt occurs. This blocking call is the **waitFor** command. When the waitFor command is used in application software, and its desired to wait for the 1PPS received interrupt, the full waitFor syntax is **waitFor 0 0 0** (for the first TSync board installed in the system and to wait for the “1PPS received” interrupt to occur) before allowing the application software to proceed.

Z) Desire to generate an interrupt at a specified frequency rate other than once-per-second

Attached are copies of both the TSync manual and driver guide, which discuss interrupt generation. These two documents discuss this capability. Refer to Section 5.1 of the driver guide and Section 5.9 of the manual for information on interrupt generation.

There is a GPIO interrupt available. A GPO output pin can be configured for “squarewave” mode in order to generate a squarewave output. An interrupt is then generated at the active edge of each sqaurewave. The rate of the interrupt is controlled by the frequency of this square wave output.

FYI:

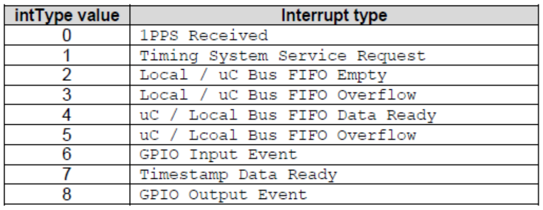
- Interrupts are enabled by disabling the Interrupt mask of the desired interrupt type (intType).
- Interrupts are disabled by enabling the Interrupt mask of the desired interrupt type (intType)

The API call/example program used for interrupt generation is: **HW_SetIntMask.**

The actual call to generate an interrupt at a specified interval using the GPIO is: **HW_SetIntMask 0 8 0 0** <enter> (as broken down below).

HW_SetIntMask <board handle> <intType> <index> <benable>

intType
The “intType” value in this command is selected from the following table. (Value “8” is selected for the GPIO interrupt to occur).



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

bEnable
The **en** variable in this command should be set to **0** (FALSE) in order to disable the interrupt mask, thereby enabling the interrupt.

The **en** variable in this command should be set to **1** (TRUE) in order to enable the interrupt mask, thereby disabling the interrupt.

To generate an interrupt at a desired interval, the GPIO Output Event Interrupt (type 8) should be used. The rate of the interrupt is controlled by the rate of the square wave.

To use the GPIO to generate an interrupt, do the following:

- Setup Pin0 to a “square wave”:
- ./GO_SetMode 0 0 2
- Enable Pin0 output:
- ./GO_SetEnable 0 0 1

To configure the interrupt rate, us the following command:
“GO_SetSqWave A B C D E F G”
Where:
A = card index
B = I/O pin
C = offset from 1pps in nsec
D = period
E = scale for period 0 = nsec, 1 = usec
F = pulse width in nsec
G = 0 is neg pulse, 1 is pos pulse

So for example, if you want GPIO “0” to generate interrupts at a rate of 1kHz, :

```
./GO_SetSqWave 0 0 0 1000 1 10000 1
```

For 10kHz:

```
 /GO_SetSqWave 0 0 0 100 1 10000 1
```

AA) Desire to generate a periodic interrupt every 10 msec

Email from Dave Sohn (2 Oct 13) Using a general purpose output, you can set up a 10ms period signal and receive interrupts every time that signal triggers.

BB) Desire to generate a periodic interrupt every 500 msec (using a squarewave on GPO pin)

One of the GPIO pins (such as GPIO pin 0) needs to be configured for 500ms square wave output (per your desired ½ second interrupt interval).

To configure the GPIO pin 0 for a 500msec square wave, 1 msec pulse width, positive-going pulse, use the following API calls (or example programs):

./GO_SetMode 0 0 2 (GPIO “0” set to square wave mode)
.**/GO_SetSqWave 0 0 0 5000000 1 1000000 1** (GPIO “0”, 0 offset, 500msec period, usec scale, 1msec pulse width, positive pulse)
GO_SetEnable 0 0 1 (GPIO “0” enabled)
2. Desire to generate a periodic interrupt every 1kHz (using a squarewave on GPO pin)

So for example, if you want GPIO “0” to generate interrupts at a rate of 1kHz:

```
./GO_SetSqWave 0 0 0 1000 1 10000 1
```

- ➢ Desire to generate a periodic interrupt every 10kHz (using a squarewave on GPO pin)

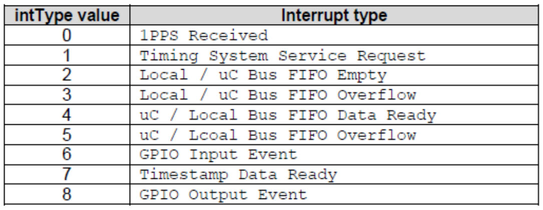
So for example, if you want GPIO “0” to generate interrupts at a rate of 10kHz:

```
./GO_SetSqWave 0 0 0 100 1 10000 1
```

To generate interrupts, the interrupts need to "unmasked", using the HW_SetIntMask API call (set to not true, to unmask interrupts- default configuration is the interrupts being “masked”). With interrupts now being generated (unmasked), you application software can now use the TSYNC_waitFor blocking call for your software to wait for a specific type of interrupt to occur.

intType

The “intType” value in this command is selected from the following table. (Value “8” is selected for the GPIO interrupt to occur).



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

HW_SetIntMask 0 8 0 0   (Disable GPO output interrupts mask, which enables the interrupts)

TSYNC_waitFor 0 8 0   (Call in your application software to wait for an interrupt on GPIO “0”).

waitFor blocking call

If desired, there is an available blocking call that can be used to halt the application code, until this interrupt occurs.   This blocking call is the **waitFor** command. When the waitFor command is used in application software, and its desired to wait for the 1PPS received interrupt, the full waitFor syntax is **waitFor 0 0 8** (for the first TSync board installed in the system and to wait for the “GPIO” interrupt to occur) before allowing the application software to proceed.

CC) Desire for Interrupts to be generated when timestamps are taken (TimeStamp data ready)

Email from Dave Sohn: It is possible to be interrupted when timestamps are taken. He could then disable timestamping, read one timestamp, clear the timestamps, reenable timestamping and then wait for the next event interrupt. Below are the APIs to he would use:

Setting active edge for input events for a given general purpose input
TSYNC_GI_SetEdge(
    TSYNC_BoardHandle hnd,
    ID_PIN index,
    EDGE edge)

Where:
0 = Falling edge
1 = Rising edge
2 = Both edges (Note: As also mentioned in Triggering on event input, we don’t support triggering on both edges. An “opt” error message is displayed when selecting this value).

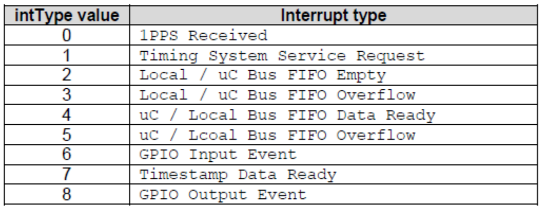
Enabling/disabling timestamps on input events for a given general purpose input
TSYNC_GI_SetTsEnable(
    TSYNC_BoardHandle hnd,
    ID_PIN index,
    int bEnable)

Enabling/disabling interrupt mask for the timestamp event
TSYNC_HW_SetIntMask(
    TSYNC_BoardHandle handle,
    INT_TYPE intType,
    unsigned int index,
    int bEnable)

Note: Enabling interrupts can be confusing. In order to **enable** the interrupt, you need to **disable** the mask. The following is incorrect code for enbabling interrupts

err = TSYNC_HW_SetIntMask(hnd, src, index, **1**);

intType
The “intType” value in this command is selected from the following table. (Value “8” is selected for the GPIO interrupt to occur).



intType value	Interrupt type
0	1PPS Received
1	Timing System Service Request
2	Local / uC Bus FIFO Empty
3	Local / uC Bus FIFO Overflow
4	uC / Local Bus FIFO Data Ready
5	uC / Lcoal Bus FIFO Overflow
6	GPIO Input Event
7	Timestamp Data Ready
8	GPIO Output Event

You should be calling the following to disable the mask and enable the interrupt:

err = TSYNC_HW_SetIntMask(hnd, src, index, **0**);

Waiting for the timestamp event input
TSYNC_waitFor(
    TSYNC_BoardHandle handle,
    INT_TYPE intType,
    uint32_t index)

Enabling/disabling hardware timestamping
TSYNC_HW_SetTsEnable(
    TSYNC_BoardHandle handle,
    int bEnable)

Retrieve a single timestamp
TSYNC_HW_GetTsSingle(
    TSYNC_BoardHandle handle,
    TMSTMP_SRC source,
    TSYNC_HWTimeObj *pObj)

Clear remaining timestamps in FIFO
TSYNC_HW_TsClear(
    TSYNC_BoardHandle handle,
    TMSTMP_SRC source)

O goes non-empty. Time stamp data is available in the time stamp FIFO when this interrupt occurs.

NOTE: The rest of the response refers to sections in the Driver Manual.
Interrupt usage is setup by unmasking the interrupt using the TSYNC_HW_SetIntMask API (4.2.213). The customer would then wait on an interrupt utilizing the TSYNC_waitFor API (4.2.5).

Assuming they would use the General Purpose Outputs (GPO) in square wave mode to generate periodic interrupts, the customer would need to set the output mode using the TSYNC_GO_SetMode API (4.2.191). Then they would need to set the square wave configuration using the TSYNC_GO_SetSquareWave API (4.2.197). Finally, they would enable the GPO using the TSYNC_GO_SetEnable API (4.2.188).

After all this, the customer would receive an interrupt based on the period they set the GPO to.

TSYNC_hwSetIntMask

The index of TSYNC_hwSetIntMask is used for interrupt types that have more than one source, like GPIO. The index is then used to indicate which instance of that interrupt type to mask/unmask. If you wanted to unmask the GPI input 1, for example, then your interrupt type would be GPI and your index would be 1 to indicate general purpose input 1.

waitFor blocking call

If desired, there is an available blocking call that can be used to halt the application code, until this interrupt occurs. This blocking call is the **waitFor** command. When the waitFor command is used in application software, and its desired to wait for the 1PPS received interrupt, the full waitFor syntax is **waitFor 0 0 8** (for the first TSync board installed in the system and to wait for the "GPIO output event" interrupt to occur) before allowing the application software to proceed.

Interrupt FAQs

Q. Is there a way to determine how many interrupts we have missed from one call to 'wait *_for_*interrupt' to the next?

A. Not through the driver itself. You may be able to determine it through the OS. In linux, the OS keeps a running total of the interrupts generated by any interrupt line in /proc/interrupts. Just be aware that it is collecting all interrupts on that line, so if it is shared with other devices, those will be shown as well. Also, all interrupts from the TSync will be aggregated in the tally.

Q. Interrupts don't appear to be generated at exactly the same moment, every time. "Also we have noticed inconsistencies in the time that our application is "woken up" after TSYNC_waitFor. This is a deal breaker for us. We run the interrupts at 128Hz. The interrupt period is 7812500 nanoseconds. With the use of some debug code we have timed the "wake ups" and we see a certain pattern. The cards generally interrupt at precisely the right period, but around once a second, the interrupt takes longer. I typically see it take somewhere in the 9000000 nanoseconds with the following interrupt being in the 5000000 nanoseconds. The average of these two interrupts is always precisely the right period but that isn't good enough. Falling behind on the interrupts means certain functionality is postponed which defeats our realtime simulation."

A. Regarding the report that the TSync interrupts don't appear to be happening "when expected", I have some information for you that I hope may be able to help you with this condition:

The TSync-PMC board's interrupts themselves are occurring exactly when they are supposed to occur, based on the configuration of the GPO output. The hardware interrupts are occurring "real time" but only at the kernel level. Unfortunately, factors beyond the control of the TSync-PMC board and its associated driver may prevent the application software threads from being able to wake up at the same moment that the interrupt occurred. This is partially because Linux is not a Real Time operating system, and can also be affected by the loading of the PC. The OS may have many processes and other interrupts occurring at the same moment of a TSync interrupt occurring that will prevent the software from being able to handle the interrupt in a controlled time-frame each time the interrupt occurs. The application software thus appears to be erratic, but at the hardware level, the "average" time for the interrupts to occur is the expected times for the interrupts to be occurring. At the application software level though, the thread may wake-up sooner than it did previously, or it may take longer to wake-up than previously, depending on what else is occurring on the machine at that particular moment. At this level, the timing cannot be precisely controlled.

As for possible solutions to help improve the "real time" operational needs of your application threads, extensions may be available for the Operating System to make it more resemble a "Real Time Operating System". Changing the priorities of other application software programs that may be running at the same time may also help. Networking is a large generator for system interrupts, which can significantly affect how long it takes for the interrupts to be handled by your application software, so if you can somehow minimize the network activity, this may help as well.

In summary, the board is generating the interrupts at exactly the right time, every time. However, your application software will not likely be able to react to the interrupts and wake-up the threads at the same exact interval that the interrupts are occurring, due to factors beyond the control of the card. The wake-ups may occur earlier than normal or may take longer than normal. But the average time is as expected, based on the configuration of the GPO.

Issues with Interrupt generation

Interrupts weren't being generated on one platform but interrupts did occur when the board was moved to a different platform:

Email from David Higgins (4/21/11)
After a bit of Google research and a bit of poking around with the Linux lspci command, I found the underlying reason for why one particular machine here – one using the Intel S5520HC motherboard, and the one I was using for development work, naturally – wasn't generating interrupts when it should. What I found was the PCI 2.3 specs (see: http://tinyurl.com/442xxfh) added an "Interrupt Disable" bit to the PCI configuration space Command register. lspci showed that this bit was set for the TSync-PMC card when installed in the Intel server motherboard system, and cleared on two other systems.

Since the tsyncpci driver is GPL'ed, I can use a one-line addition calling pci_intx() to clear this bit. The pci_intx() call has been around since at least 2.6.15, but I can't say exactly when it came into being. Here's the code change I made to tsync_drv_2_6_13_1.c, starting around line 620 or so:

```
/*
**  set PCI configuration register to enable PCI I/O and Memory access
*/
rv = pci_enable_device(pdev);
if (rv) {
    if (tsyncpci_devp[idx]) {
        kfree(tsyncpci_devp[idx]);
        tsyncpci_devp[idx] = NULL;
    }
    TPRO_LOG(TPRO_LOG_ERROR,
             (KERN_ALERT "[J%lu]%s: pci_enable_device()\n",
             jiffies, __FUNCTION__));
    return -EIO;
}
/*
**  CCUR: make sure traditional PCI interrupts are enabled
*/
pci_intx(pdev,1);
```

The last few lines are my addition. Perhaps I should have added "non-MSI" after "traditional". Speaking of which, does the TSync-PMC hardware support MSI and/or MSI-X interrupts?

Anyway, I can't answer why the Interrupt Disable bit is set on the Intel based machine and not on the others – perhaps the TSync-PMC card generated an interrupt when the BIOS touched it and the BIOS decided to shut it up. And I don't know why it's being set for the TSync-PMC card but not for the PCIe IRIG card made by those-other-guys-who's-name-starts-with-S.

But I know a workaround, and now so do you.

TSync-PMC Uptime

TSYNC_SS_GetUptime: Get the board's total uptime in minutes.

**Time/date reads / (HW_GetDate, HW_GetTime and CS_GetTime) / Time epoch

HW_GetDate call

HW_GetDate is one of the legacy TPRO calls in the TSync driver. It is in the TSync-PMC in the TPRO calls to support the legacy TPRO/TSAT timing boards.

Time stamps

- There are two main available calls for time reads: cs_Gettime and hw_Gettime
- During these calls, the current time is latched in the FPGA. There are several 16 bit registers to be read for each time read (Refer to Section 2 of the Application Programmers guide)
- Host Super-Second System Time Low (Seconds and minutes data)

Note: Reading this register latches all of the Host Sub-Second System Time, Host Super-Second System Time, and Host Sub-Second BCD Time registers.

- Host Super-Second System Time Mid Low (Hours and some of the day info)
- Host Super-Second System Time Mid High (Some of the day info and some of the year value)
- Host Super-Second System Time High (Some of the year value).
- Host Sub-Second System Time Low
- Host Sub-Second System Time High (Syste)
- Host Super-Second Binary Time Low (Binary time data)
- Host Super-Second Binary Time High (Binary time data)
- Host Sub-Second System BCD Time Low (BCD data)
- Host Sub-Second System BCD Time High (BCD data)
- Sync Status: cs_Gettime doesn't report sync status. hw_Gettime does report the sync status.
- hw_GetTime reads time directly from FPGA with no processing performed on the timing board. However, with cs_Gettime calls, the FPGA works with the micro to format the time responses. So, cs_Gettime is much slower at responding. We recommend using hw_GetTime calls for faster responses.
- cs_Gettime Instance value of 0 returns Year, DOY, Hr, Min Sec, nanoseconds and DST
- cs_Gettime Instance value of 1 returns Year, DOY, Hr, Min Sec, milliseconds microseconds and DST.
- cs_Gettime Instance value of 2 returns Seconds, nanoseconds and DST.
- hw_Gettime responds with: Year, DOY, Hr, Min Sec, nanosecond, Sync status.
- Typical response times for hw_GetTime time reads is about 3 to 4 microseconds.
- The gettime API/Example programs can read the time of the board with 1 microsecond resolution.
- Hw_Gettime reports sync status (from the high order bits register) as well as years, days, hours, minutes seconds, and sub-seconds (all from the low order bits register) with 1 microsecond resolution. The seconds are floating point numeric.

HW_GetTime call

- **HW_Gettime responds with**: Year, DOY, Hr, Min Sec, nanosecond, Sync status ("bSync")

bSync" field: As it is likely important to know whether or not the board is currently synchronized to an external reference when the time was read, the
TSYNC_HWTimeSecondsObj struct's **bSync** field will indicate:

- "**TRUE**" when synced (the board is either currently synced to its external reference or in holdover mode) OR
- "**FALSE**, when "not in sync" (the board hasn't synced to its external reference since it was last powered up or it went into holdover mode and the holdover period has since expired).

CS_GetTime calls

- Sync Status: cs_Gettime doesn't report sync status (hw_Gettime does report the sync status)
- With the cs_Gettime calls, the FPGA works with the micro to format the time responses. So, cs_Gettime is much slower at responding. We recommend using hw_GetTime calls for faster responses.
- cs_Gettime Instance value of 0 returns Year, DOY, Hr, Min Sec, nanoseconds and DST
- cs_Gettime Instance value of 1 returns Year, DOY, Hr, Min Sec, milliseconds microseconds and DST.
- cs_Gettime Instance value of 2 returns Seconds, nanoseconds and DST.

HW_GetTime versus CS_GetTime

The **CS_GetTime** command comes from the FIFO buffer whereas the **HW_ GetTime** command is directly from a register controlled by the microprocessor. This results in fast response to the **HW_GetTime** command (only when the external reference is connected) but delays in the response from a CS_GetTime command (It was reported that the delays are not present when the card is free-wheeling).

Email from a customer (discussing getDate, but it also applies to CS_GetTime calls)
This delay in getDate (or CS_GetTime) seems to occur regardless of call frequency – I made my original tests at 100Hz (10ms between calls), but the delay is apparent even down to 10Hz (100ms between calls). Once a second, a call to getDate will block for 100ms, but only while the external synchronization signal is connected to the card.

The HW_GetTime command shows no delay, only the getDate (or CS_GetTime) command. It's my thought that synchronization is interfering with FIFO operation in some way.

Normal recovery time for the FIFO buffer operations is about 200 microseconds. Gettimes can read times like every microsecond. This can lead to large time corrections. This issue is even worse when IRIG time code is present. With IRIG signal present, the micro has to process the timecode, slowing down the FIFO buffer operations, resulting in even larger time corrections occurring.

Latentcies of HW_GetTime and CS_Gettime calls:
Refer to the next section below for more info (Latentices of API calls)

**Latentices of API calls/Cable delays (including Hardware and Software GetYime calls)

Latencies in the TSYNC board are inherent with the operation

Accounting for Cable delays

Latencies in the TSYNC board are inherent with the operation of the TSync board and can also be caused by latency of the external input:

If latency (offset) is coming from the input reference, there are API calls specific to those references to apply an input offset:

TSYNC_IR_SetOffset (IRIG Input) Used to enter an offset for the IRIG input (entered in the number of nanoseconds, as a positive or negative value)

TSYNC_GR_SetOffset (GPS Input) Used to set the GPS antenna cable delay.

TSYNC_PR_SetOffset (EPP 0- External PPS Input)

HW_GetTime/ CS_GetTime call latencies

- Hardware time reads (reading time from the registers) have VERY low latencies (in the low microseconds range), but the process of asking for and receiving these hardware time stamps at the application layer adds quite a bit of latency to the call itself (possibly measured in the milliseconds range).
- Windows PCs will have even higher latencies than a Linux box will have.
- There are several "layers" for latency to be added to a hardware time read, as shown below (the timing board itself, PCIe bus, driver, OS kernel and Application software).

The lowest latency is with the reading of the registers on the TSync-PMC board.
The highest latency is from the OS kernel and Application software layers.

- Hardware time reads are a "two-way" process (asking for and then receiving the time stamps). So latencies in each layer are added in both directions for each time read.
- One potential work-aound/suggestion for much lower latentcies is to use the "1PPS Received" interrupt to generate an interrupt at the top of each second. Since the interrupt is not generated through the driver and other layers that case latentcies, it's a "one-way direct trigger" that happens at a specific moment each second without a need to request it with an API call. Refer to the email below in red for more on this suggestion.

Email from Denis Reilly (19 Mar 2013) We experimented a bit when we first released the TSync. I don't remember the exact data, but I do remember some things:

- round-trip HW_GetTime call latency was on the order of several microseconds in Linux.
- In Windows, it could be twice as long.

Inherent latencies with the board (delays between each time read) versus timing accuracies

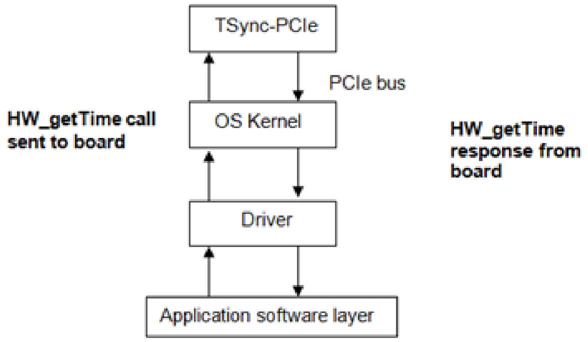
Solaris time read latency from TSYNC_HW_GetTimeSec should be in the range of 8-10 usecs. We don't have any information on latency of the Solaris systems calls gettimeofday() or gethrtime().

Q (from a customer) We are finding that it is taking approximately 3.58 microseconds to read the time of day from the TSync card. We are of the understanding that we should be able to read time from the card with 1 microsecond time of day resolution, which is what we need for our application (all of our computers, software, and data are expecting to be sync'd to 1 usec). Do you have any test programs, code snippets, or anything else we could use to try and replicate a 1 usec time of day read accuracy?

A. There is confusion between latency and accuracy. The latency between time reads can easily be the 3.58 usecs they are seeing. However, that doesn't mean the accuracy of the read is 3.58 usecs. The accuracy will depend on the accuracy of their input and the variation of the latency of their time reads. This is pretty system dependent. They could model the variation in the latency of their time reads, and that should be representative of the accuracy they are getting from the card.

There are a couple of factors to keep in mind when performing time reads of the TSync-PMC timing board. One of these is inherent latencies between time reads that occur and another is the accuracy of each time read (as compared to the input reference for the board). Latency is the amount of time required between each time read occurring (as determined by the board, the computer, software, etc). The amount of elapsed time between each read explains why you are seeing 3.58 microseconds to read the time from the Tsync board. If you try to read the time twice in the same microsecond for example, the second read will lag the first read by the amount of time required betweens reads to occur (the latency). Responses to time reads cannot occur faster than the latency time permits. To prevent seeing delays in the time reads, allow a delay of about 4 microseconds between each time read. This will negate the ~4microsecond delays you are seeing.

The accuracy of each individual time read will be dependent upon the accuracy of the input reference to the TSync-PMC board combined with the latencies you are measuring on each time read. The latencies may always be a set length of time, or they may vary from one read to the next. The more consistent the latency measurements are, the better the representation of the accuracy of the time reads (consistent latencies indicates more accurate time reads).


TSync-PCIe
PCIe bus
HW_getTime call sent to board
OS Kernel
HW_getTime response from board
Driver
Application software layer

Email KW sent to customer, with input from Denis Reilly (11/27/12)
To begin, reading the hardware time of the TSync-PMC board (using the HW_GetTime call) provides the least latency for accurate time reads of the timing board. However, there are several areas where latencies/variable latencies can be introduced into the actual time read.

HW_GetTime calls are “two-way” calls (from the request, to the reply). In between, the timing board hardware, its associated driver, the PCIe bus, the OS kernel, and the application software all come into play. Each one of these areas can introduce latencies, especially at the OS kernel and Application software levels (Windows is even worse than Linux, at these two areas) and with the TSync-PMC hardware having the least latency.

As Windows is not a real time system, its “priorities” can play a big part in latencies affecting the time reads. If a time read is sent, but the OS is busy when the time stamp is returned, the time will have induced latencies, with this being one of the biggest factors. Latencies for reading hardware registers are typically measured in the low microseconds. But when brought to the application layer, the latencies can then be measured in milliseconds (especially in a Windows environment).

Specific to the “10ms” clock, the problem with this testing is the fact that this clock is not extremely accurate, so the time read commands aren’t necessarily going to occur at the same time, each time. The time reads at the time of the read itself is typically accurate to 1 microsecond, but the processing delays/latencies for when the call is sent, to when the time is returned, is going to appear as jitter/offset to the 10ms clock.

One recommendation we have for you, that may help with both testing and with your application software, is to utilize the 1PPS interrupt capabilities of the timing board. The TSync-PMC boards can generate an interrupt at the top of each second. As these PPS interrupts don’t rely on all of the API interfaces mentioned above (Application software, OS, driver, etc), a low latency “marker” can then be used as a “hardware time stamp”. Generated interrupts, unlike time-reads, are a “one-way direct communication” between the timing board and the application software, thereby reducing all of the two-way latencies inherent with API calls and hardware time reads. And unlike “software only” approaches to time synchronization, these very low latency hardware interrupts can be generated by the TSync-PMC timing boards, when they are installed.

An interrupt can be generated at the top of each second. This particular “interrupt generator” is known as the “**1PPS Received**” interrupt (designated by “intType value “0”). Attached for your reference is a copy of the TSync-PMC driver guide. Section 5.1 (starting on page 5-238) discusses how to enable interrupts to be generated for different situations, such as each time the system

PPS occurs.

In summary of the Driver guide, generating this interrupt consists of **disabling** the interrupt mask for "1PPS received" (all interrupt masks are **enabled** by default). To disable the 1PPS Received mask (thereby enabling the "1PPS Received" interrupt), use the following call: **HW_SetIntMask 0 0 0 0** (where the first 0 is for the first/only TSync board installed in the system, the second 0 is for the "1PPS Received" interrupt, the third 0 is the index value and the last 0 disables the interrupt mask).

With the 1PPS interrupt now being generated at the very top of each second, one available option is to use the available "**waitFor**" API blocking call, to halt the application software until the "1PPS Received" interrupt has occurred. The API call to wait for this interrupt before the application software proceeds in **./HW_WaitforInt 0 0 0** (where the first 0 is for the first/only TSync board installed in the system, the second 0 is to wait for the "1PPS Received" interrupt and the third 0 is the index value).

You may also be able to use the generated 1PPS interrupt in other ways, as a VERY low latency marker for your testing (unlike using the very high latency 10ms clock at the application software layer to send and receive the hardware time reads – which are affected by high latency factors in between).

Desire for faster time reads (hw_Gettime takes too long)

Instead of delays to read all of the TSync-PMC's "Host" registers, the machine can have NTP sync to the TSync-PMC board- this will then sync the Linux kernel. Then, the kernel time can be read directly (this is much faster than reading the TSync board, with time reads taking only about 1 microsecond, instead of about 4 microseconds)

The results you are observing are not unexpected. For low latency time reads, we recommend using the HW_GetTime command, instead of using the CS_GetTime command. Here's why:

The CS_GetTime command comes from the FIFO buffer (as the time/date data is formatted before being reported) whereas the HW_ GetTime command is directly from a register controlled by the microprocessor. This results in VERY fast response to the HW_GetTime command (only when the external reference is connected) but delays in the response from a CS_GetTime command. HW_GetTimes have about a 4 microsecond delay.

Normal recovery time for the FIFO buffer operations is about 200 microseconds. HW_Gettimes can read times every microsecond. This can lead to large time corrections. This issue is even worse when IRIG time code is present. With IRIG signal present, the micro has to process the timecode, slowing down the FIFO buffer operations, resulting in even larger time corrections occurring.

If its desired to have even lower latency than the HW_GetTime call (about 1 microsecond versus about 4 microseconds) and the board is installed in a Linux machine, instead of reading the time directly from the TSync board, the TSync board can sync the Linux machine. Then, the kernel time can be read directly, instead of reading the time of the TSync boards. Attached you should find a document that discusses how to sync the Linux kernel to the time of the TSync board

Syncing a linux system/kernel

➢ Sync the Linux kernel time to the TSync-PMC board using TimeKeeper software

Note: Not applicable for Windows or Solaris

➢ For more info on Timekeeper software, refer to the Timekeeper software section towards the beginning of the the VelaSync/Geo Tech note. I:\Customer Service\1- Cust Assist documents\VelaSyncAndGeoCustAssist.pdf

Syncing a Linux system with a TSync-PMC board and TimeKeeper software is the **most optimum** and **accurate** method to sync a Linux system. According to Denis, this method will provide sub 1 microsecond synchronization of the linux kernel

➢ Sync the Linux kernel time to the TSync-PMC board (when not using TimeKeeper software)

Earlier NTP iversions prior to v4.2.8 in the system needs to be patched to install a reference clock driver that allows NTP to sync to the installed TSync-PMC-PCIe board(s).

- Refer to the UNIX Application Note for general information on syncing Linux: Software\UNIX\UNIX Application Note.pdf

- Refer to the README file in the TSync Linux driver for specific information (tsync\linux\ntp folder) on patching NTP to use the TSync-PMC board as a reference clock driver.

How to read the Linux kernel time, once it's being synced to the TSync board via the Reference Clock driver

(Email KW sent to Jeffries on 3/28/12)
As for reading the system time, we use the **gettimeofday** call in our software. We only need the precision to 1 microsecond that this call provides, so this call works well for us. There is also an available **clockgettime** call and **clockgetres** (reports the resolution of clockgettime) call that may be able to provide you with even greater precision than 1 microsecond.

We don't directly support these calls, but with a Google search, you should be able to readily find additional information and c code on these calls (as well as any others that may also exist). For examples, refer to sites such as:
http://souptonuts.sourceforge.net/code/gettimeofday.c.html

http://stackoverflow.com/questions/5362577/c-gettimeofday-for-computing-time

Email KW sent to Jeffries about this (3/26/12) about this concept
To begin, the time, date and sync status are maintained in more than one "Host" register on the TSync-PMC board. These time registers in the TSync-PMC board need to be latched each time a time read needs to be performed. The first time read latches the registers (to stop the other registers from continuing to increment). Then additional, consecutive register reads are performed to obtain the rest of the time/date/sync status. The hw_Gettime call allows the time registers to be latched and all of the necessary registers read to obtain this information from the timing board. Because the time and sync status are contained in more than one register, it's not the same as just reading the CPU time. The time it takes to latch and read all of the registers is about 4 usecs, on average.

If faster time responses is more vital to your application than the accuracy of the time stamps, the only way to speed up the response time would be to use NTP software to sync the Linux kernel. Then, you could read the kernel time directly. This would alleviate the delays of the TSync-PMC's time registers from needing to be read during each gettime request. You would then be reading "CPU" time, instead. Typically, NTP can be synchronized directly to the TSync-PMC board installed in that machine, to within just a few microseconds of the TSync-PMC's time. The accuracy of the time stamps will be slightly degraded but the elapsed time to perform them will be less.

If you aren't familiar with NTP software syncing to the installed TSync-PMC board, the Linux driver for the TSync-PMC board contains a README file that discusses how to patch the NTP software on the machine to have it sync to the TSync-PMC timing board. This reference clock driver then allows NTP to sync the kernel, which would then allow you to read the kernel time, much faster than performing time reads directly from the TSync-PMC board.

Please let me know if you need any additional information on using the TSync-PMC board to sync the linux kernel, as an alternate method to obtain time reads. I understand from Engineering that this method should allow time reads to be performed in about 1 microsecond or so.

HW-getTimeSec()

The **TSYNC_HW_GetTimeSec()** call is normally the best command for reading the time. As it is likely important to know whether or not the board is currently synchronized to an external reference when the time was read, the

bSync" field: As it is likely important to know whether or not the board is currently synchronized to an external reference when the time was read, the
TSYNC_HWTimeSecondsObj struct's **bSync** field will indicate:

- "**TRUE**" when synced (the board is either currently synced to its external reference or in holdover mode) OR
- "**FALSE**, when "not in sync" (the board hasn't synced to its external reference since it was last powered up or it went into holdover mode and the holdover period has since expired).

Q On average the time reported by the Spectracom is 0.465 second late compared to the time we got from the other GPS base device. Time in the Spectracom is retrieve via API call TPRO_GetTime()
A. (Email from Dave Sohn): The most accurate time reads will be using the HW_GetTime calls. I wouldn't use the legacy TPRO calls

Time epoch

Q. This is regarding time in sec, ns form. I wanted to know what the starting time, or epoch, is for time in this form. So, I set the time on my board to zero as follows: CS_SetTime 0 2 0

Then I read back the time. Turns out the epoch is Jan 1, 1970, which is the same as that used by Unix/Linux.

So, onto my question: This epoch information is not documented anywhere in the Spectracom docs that I can find. Will the epoch always be Jan 1, 1970 for time in this format for Spectracom TSync products?

A. This is the same epoch you referred to in your email for Linux (Jan 1, 1970). The TSync-PMC-PCIe boards will always use this epoch for its time base.

From Wikipedia (http://en.wikipedia.org/wiki/Unix_time)

Unix time, or POSIX time, is a system for describing instants in time, defined as the number of seconds elapsed since midnight Coordinated Universal Time (UTC) of Thursday, January 1, 1970 (Unix times are defined, but negative, before that date), not counting leap seconds, which are declared by the International Earth Rotation and Reference Systems Service and are not predictable. It is used widely in Unix-like and many other operating systems and file formats. It is neither a linear representation of time nor a true representation of UTC (though it is frequently mistaken for both), as it cannot unambiguously represent UTC leap seconds (e.g. December 31, 1998 23:59:60), although otherwise the times it repI:\Customer Service\1- Cust Assist documents\VelaSyncAndGeoCustAssist.pdf
resents are UTC. Unix time may be checked on some Unix systems by typing date +%s on the command line.

**Clock System ("CS" calls for Timescales, TZO/DST offsets)

CS calls provide an abstract interface to the timing subsystem.

Where <time scale>:
UTC=0
TAI= 1
GPS= 2
LOCAL **=** 3,

Related CS calls/functions:
Get/Set local time offsets

Get/Set Time
Get/Set TimeScale Get or Set board's current time scale.
Get/Set Leap second
DST rules/DST state
Get/Set Year value

Get/Set Local time conversion (TZO and DST correction)
The TSync-PMC's system time can be converted to local time, as desired. Local time is configured using two separate values, Time Zone Offset and DST.

Time Zone Offset (TZO)
Time Zone Offset is configured using the **SYNC_CS_SetTimeZoneOff** API call. The offset is entered as the number of total seconds of TZO offset from UTC, for that particular region's Standard time. Example: Pacific is 8 hour offset during Standard time. 3600 seconds in an hour x 8 = 28,800 seconds for the Time Zone Offset

Desire to change timescales

- The factory default timescale for the TSync-PMC board is UTC. It can be reconfigured to be GPS, TAI or Local timescale.
- For a visual indications of the time differences between each of the the time scales, go to:http://www.leapsecond.com/java/gpsclock.htm

Important Note: Unlike SecureSyncs (Which persist the GPS and TAI timescale offsets -such as 15 for GPS and 34 for TAI after they are first obtained/unit rebooted), TSync boards do not persist these values through power-ups. So, after each reboot, the TSync-PMC board needs to either obtain the offset values from an external reference (such as GPS providing the UT1 value), or the offset needs to be entered using the **CS_SetTimeScaleOff 0 X** API call/example program (where x is the timescale being offset)

X values
UTC=0
TAI=1
GPS=2
Local=3

Until the timescale offsets are either automatically obtained from an external time source or have been manually entered, the timescale offsets will be a value of "0". So when the HW_Gettime or CS_Gettime call is performed, it will continue to report UTC, no matter what the timescale is specified.

Email from Keith 6/8/12 (just before the June leap second)
I just spoke to one of our engineers, who reminded me that the TSync-PMC boards do not store what the offset values for the other timescales besides UTC (GPS, TAI and local). So, after each power-up, the TSync-PMC board needs to first be synced to GPS in order to automatically obtain the current UTC to GPS time scale offset value (the UT1 correction factor, which is currently "15"). Or the current offset value needs to be set after each reboot. Based on this, your TSync-PMC board likely hasn't synced to GPS since it was last powered-up. If you happen to be using IRIG input, instead of GPS, the correction factor has to always be "manually entered". Until the offset values are obtained either automatically or manually, they are all 0s. So the HW_GetTimes are reporting UTC – "0", which is still UTC. The "0" in this case needs to be changed to a "15", then the Time Stamps will reported in GPS time.

Everything else your doing is correct. No additional steps are required after each boot-up, as far as the time read itself is concerned. You just either need to connect it to GPS and allow it to obtain the offset from GPS automatically after each boot-up (Note this can take up to 12.5 minutes, after the GPS receiver has started tracking GPS satellites for this information to be transmitted in the GPS message). Or, set the UTC timescale offset to the current value of 15 using the **CS_SetTimeScaleOff 0 2 15** call/example program (just remember this value will increment after each leap second, such as the one occurring on June 30th.

Desire to change the TSYNC board’s timescale from UTC (Default) to GPS

- See “Important Note” above.

Use the **CS_SetTimeScale** API call/example program

Where:
UTC=0
TAI=1
GPS=2
Local=3

Example: **CS_SetTimeScale 0 2** (where 0 is the first TSync-PMC board installed and where 2 is the value for GPS timescale).

Desire to change the TSYNC board’s timescale to TAI instead of UTC

- See “Important Note” above.

Use the **CS_SetTimeScale** API call/example program

Where
UTC=0
TAI=1
GPS=2
Local=3

Example: **CS_SetTimeScale 0 1** <enter> (where 0 is the first TSync-PMC board installed and where 1 is the value for TAI timescale).

Desire to change the TSYNC board’s timescale to local time instead of UTC

Refer to the TSync-PMC driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121

The default timescale for the TSync board is UTC time. The TSync-PMC board can be set to use a local timescale for the core clock instead of being configured for UTC. The local timescale needs to be setup and then the system needs to be set to utilize that timescale.

First, you need to set the local Time Zone Offset from UTC using the **TSYNC_CS_SetTimeZoneOff** call. The offset should be provided in seconds offset from UTC.

If you want to provide DST information, use the **TSYNC_CS_SetDstRule** library call. For example, the DST rule for the Eastern Time Zone is:

Reference: 0 (Local time reference)
In: Week: 2
Day: 0 (Sunday)
Month: 3 (March)
Hour: 2
Out: Week: 1
Day: 0 (Sunday)
Month: 11 (November)

Hour: 2
Offset: 3600 (1 hour)

The final step is to tell the clock to utilize the local timescale with the **TSYNC_CS_SetTimescale** library call. Once you change the board's time scale to local, you can enter the match times in local time as well as they will then correlate to the same time base.

Known TSync-PMC issues

1) There are currently two known issues with the TSync firmware V2.0.0 release (limited release for Mobile Mode support):
- The 10MHz output was inadvertently disabled.
- The default reference table incorrectly listed IRIG AM as a higher priority reference than IRIG DCLS.

These issues do not affect anything else in TSync, and if a customer with 2.0.0 is not using these features they don't have to upgrade if they don't want to.

Both of these defects will be corrected with the TSync 2.1.0 update. However, these defects are in the EEPROM portion of the upgrade bundle. Therefore, ALL customers are recommended to apply the latest EEPROM patch files when upgrading to TSync 2.1.0, INCLUDING customers who are already at TSync 2.0.0.

- Issue with the get of GPO square wave's pulse width configuration in firmware versions of at least 2.1.0 and prior. From Erik Johansson of NOAO: I noticed the following interesting output from the getSqWave example code shipped with the driver. The input values are just fake ones that I made up: offset 100ns, period 400 ns, 0 width 100 ns, active edge 0.

All I want to do is set the SqWave config for a pin and then read it back. Turns out the pulse width comes back from the TSync board with the upper bit set. If you mask off the upper bit, the returned value is correct. Is this the proper behavior?

Dave Sohn responded with: There is a bug right now in the released TSync firmware that the high bit of the pulse width may be set. The next TSync firmware release will fix that. For now, the highest order bit of the returned pulse width should be ignored.
(Note: as far as I am aware, this issue was resolved in the version 2.3.0 driver update release, Dec 2010).

Ping not responding in Solaris (reported 14 March 2013)

Refer to Mantis case 1967. Customer verified ping from Windows but not from Solaris.

TSync-PCIe/104 (Caution: this is not an available/viable product)

Q Email from a customer I currently have several (20+) PC/104 stacks in operation; these stacks use a PC/104 IRIG time decoder. I have been directed to upgrade these stacks; we will probably retain the PC/104 form factor but use modules with a PCI Express bus instead of the ISA bus (PCie/104). I see you make TSync time code processors in a PC/104 format and was wondering if you had or plan to produce a product with a PCIe/104 interface. Any help that you can provide would be appreciated.

A reply from Dave Sohn to Steve Visosky (3 Nov 2016)
We don't currently have plans for PCIe/104, but our current offering is actually PCI/104, not PC/104. That means our backend is PCI, not ISA.

LEGACY TSync-PCIE-PTP (TSync-PTP) timing boards (discontinued)

RJ-45 connector ribbon cable (TSync-PTP boards only)

Q. (From Matt Loomis) Can you provide some detailed information on the connector type for the RJ45 ribbon connector on the TSync-PCIe PTP card? Pluribus networks has requested this in case we they need to come up with a special cable to connect within our box.

A. (from Denis Reilly 15 Feb 2013) The part number for the flex cable that is used to connect the PTP module to the little Ethernet board is Molex 98266-0127.

The little Molex connector P/N is 52746-1271 or 52746-1270.

Keep in mind that we are going to be limited in how much we can support them if they start messing with the connector and the cable. About the most we can do is provide them the normal half-height TSync bracket, and then leave them on their own for the Ethernet connection.

The little Ethernet board is very basic, though. It only has the RJ45 connector and the Molex connector, and is trivial to reverse engineer. They may be thinking about making their own connector board (and we really have no reason to stop them, other than letting them know we won't support it.)

General info on PTP and BMC (Best Master Clock algorithm)

- Refer also to PTP info in: ..\CustomerServiceAssistance.pdf

PTP CONFIGURATION/SUPPORT

Specific TSync-PCIe-PTP support:

- Refer to an **earlier version of the TSync-PCI driver guide in the new released drive(not from Arena), such as Rev J or below** for an extensive discussion on how to set up the PTP boards (1191-5001-0050) : I:\Engineering\Archive\New Released\Manuals\1191-xxxx-xxxx

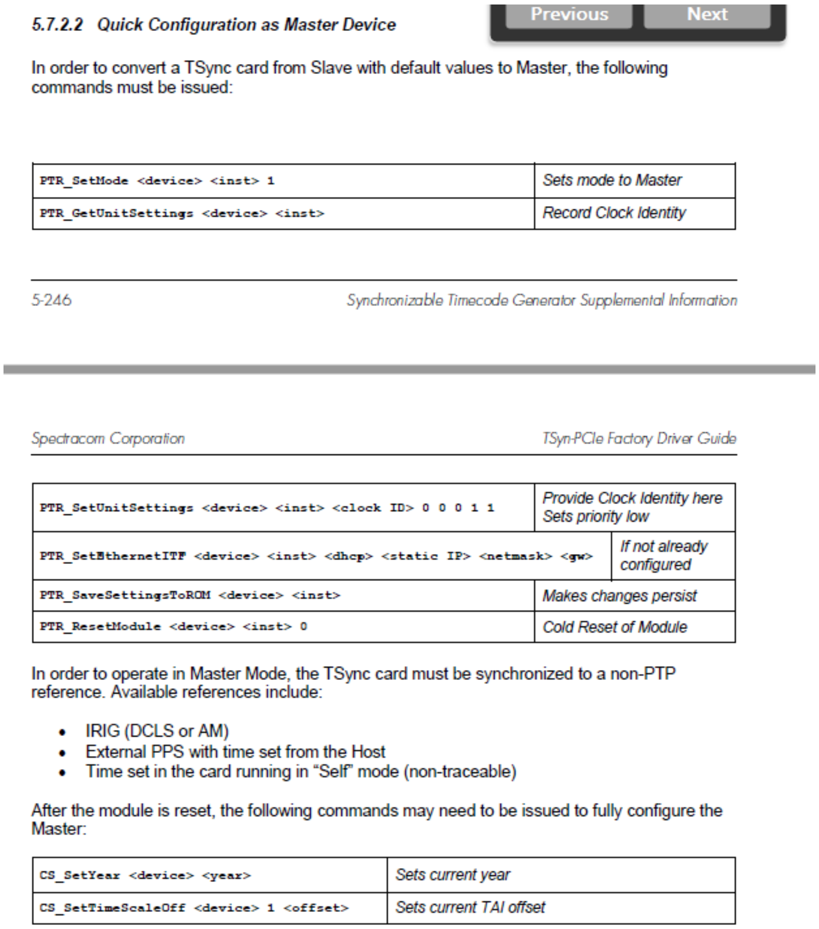
Note: All info on the earlier PTP module has since been removed from the driver guide. So can't use the guide in rena.

Configuring the network settings of a TSync PTP module (DHCP, IP address, netmask, etc) via the PTR_Set example programs/ API calls

Notes:

1) Need to configure the network settings in the PTP Master and all PTP slaves, if there are any switches in betwee the Master and slaves.

2) Refer to "**Changing the network settings of an PTP Option Card or TSync-PCIe-PTP board**" in an earlier version of the Tsync driver guide (as excerpted further below):



5.7.2.2 Quick Configuration as Master Device

In order to convert a TSync card from Slave with default values to Master, the following commands must be issued:

`PTR_SetMode <device> <inst> 1`	*Sets mode to Master*
`PTR_GetUnitSettings <device> <inst>`	*Record Clock Identity*

5-246 Synchronizable Timecode Generator Supplemental Information

Spectracom Corporation TSyn-PCIe Factory Driver Guide

`PTR_SetUnitSettings <device> <inst> <clock ID> 0 0 0 1 1`	*Provide Clock Identity here* *Sets priority low*
`PTR_SetEthernetITF <device> <inst> <dhcp> <static IP> <netmask> <gw>`	*If not already configured*
`PTR_SaveSettingsToROM <device> <inst>`	*Makes changes persist*
`PTR_ResetModule <device> <inst> 0`	*Cold Reset of Module*

In order to operate in Master Mode, the TSync card must be synchronized to a non-PTP reference. Available references include:

- IRIG (DCLS or AM)
- External PPS with time set from the Host
- Time set in the card running in "Self" mode (non-traceable)

After the module is reset, the following commands may need to be issued to fully configure the Master:

`CS_SetYear <device> <year>`	*Sets current year*
`CS_SetTimeScaleOff <device> 1 <offset>`	*Sets current TAI offset*

Save settings to ROM: **PTR_SaveSettingsToROM()**

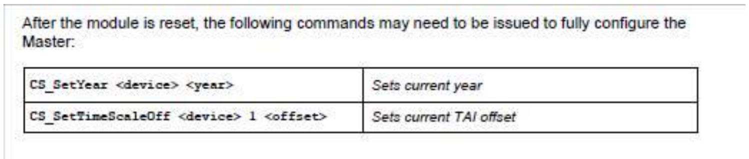
Must **reboot the PTP module** after changing the settings for the new settings to take effect:
TSYNC_PTR_resetModule

Note: We have seen that if all devices are all on the same subnet, the gateway address could be left with all zeroes added and the Slaves still synced to the PTP Master.

Set the TAI offset in the TSync Master

The PTP Module on the TSync-PTP Master needs to know the UTC to TAI offset value. When using a SecureSync as the PTP Master, the PTP module can obtain the UTC to TAI time scale offset automatically from the GPS antenna. When using a TSync-PCIe board as the PTP Master, the TAI offset value needs to manually configured.

From page 5-245 of the driver guide:



After the module is reset, the following commands may need to be issued to fully configure the Master:

`CS_SetYear <device> <year>`	*Sets current year*
`CS_SetTimeScaleOff <device> 1 <offset>`	*Sets current TAI offset*

"Operation of the TSync-PCIe-PTP" (Section 5 of the earlier driver guide versions).

Initial steps needed to sync a TSync-PCIe-PTP Slave to a PTP Master:

1. Make sure the PTP Master is in either Holdover mode or in Sync (to external input or to itself).
   - If PTP Master is a TSync-PTP, sync Master via IRIG input or use self mode
   - If PTP Master is a SecureSync, sync Master as normal (with GPS, NTP, IRIG, Self, etc)

2. PTP boards can only sync via IRIG inout (GPS input is not available). If the IRIG master doesn't provide the year info in its IRIG outpu., set the current year in the board after ezch boot-up.
   - The example program/API call to set the current year is **CS_Setyear 0 xxxx** <enter> (where xxxx is the current year value)

****Desire to use two TSync-PCIe boards to make a "GPS synced" TSync-PCIe-PTP Master

- ➢ Requires one TSync-PCIe board and one TSync-PCIe-PTP board.

For easiest install, requires one standard breakout cable and one premium cable (**Note**: Premium cable must be purchased with the two TSync-PCIe boards). But it can also be done with two Standard breakout cables with some additional extra effort.

TSync-PCIe board syncs via GPS and the TSync-PTP board syncs to the TSync-PCIe board via either:

IRIG DCLS input only (when at least one Premium breakout cable is available), or

via IRIG AM (Time reference) and 1PPS input (PPS reference) supplied from GPI 0 pin on the TSync-PCIe that is configured as a 1PPS output (Requires GPIO 0 on the TSync-PCIe be configured as 1PPS output).

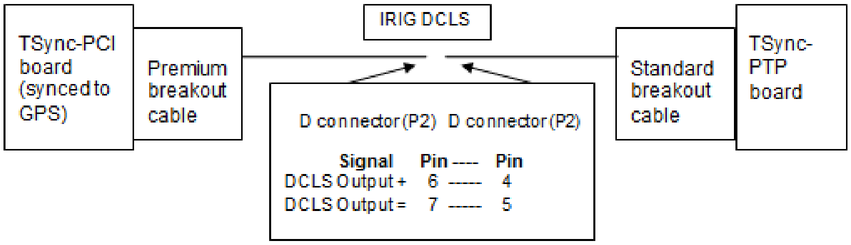
Since the TSync-PCIe boards can't have both a GPS receiver and a PTP module, without an external IRIG reference available to sync the TSync-PTP board , it takes two TSync-PCIe boards (one a TSync-PCIe-PTP board and a second TSync-PCIe with a GPS receiver) to make the TSync-PTP board a PTP Master. Both boards can be installed in the same system.

The TSync-PCIe board with the GPS receiver syncs to GPS. The TSync-PTP syncs to the TSync-PCIe board in one of two methods:

Using IRIG DCLS to sync the TSync-PTP board (Requires at least one Premium breakout cable)

Using the one Premium breakout cable attached to the TSync-PCIe board, the IRIG DCLS output is connected to the TSync-PTP board (via the Standard breakout cable). The IRIG DCLS signal provides both the "Time" and "PPS" references needed for the TSync-PTP board to declare sync.

Note: This requires a special connection between the two D connectors (can't just plug them in to each other) See the diagram below.



Necessary TSync board configurations

Note: All configurations need to be performed after each boot-up of the boards.

Note: There is no need to edit the Reference Priority table or to change the IRIG format (both are IRIG B000 by default) in this particular configuration.

Set the year in the TSync-PTP board:

By factory default, IRIG output does not provide year information. To set the year in the TSync-PTP board, need to either:

Use the **CS_SetYear 0 xxxx** API call/Example program in the TSync-PCIe-PTP board after each boot-up **OR**

Reconfigure the IRIG output of the TSync-PCIe board to include year information in the control field. Then, also need to reconfigure the TSync-PTP board to be able to read the year in the Control field. Refer to: Setting the year value upon each TSync-PCIe power-up (In TSync "Inputs" section).

Refer to the TSync Driver Guide for information on configuring the TSync-PTP board as a PTP Master.

To Verify TSync-PTP is synced to the TSync-PCIe board

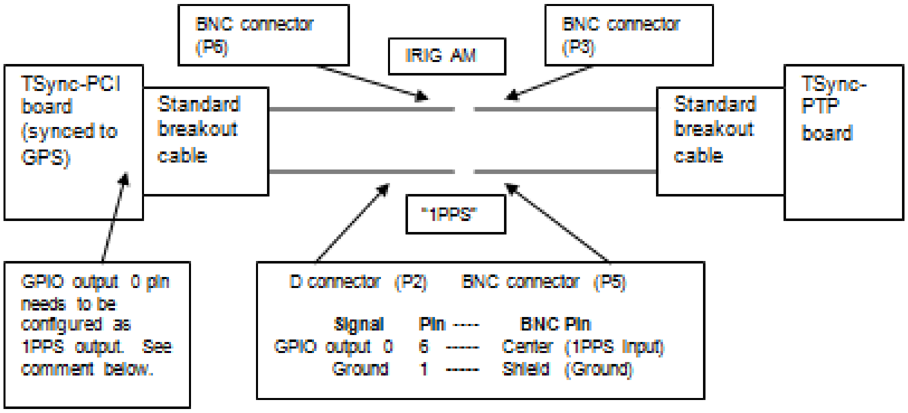
To verify the TSync-PTP board is synced to the TSync-PCIe board:

2. Verify the Green LED on the edge of the board is lit.
3. Use the example program **SS_GetSync 0** on the TSync-PTP board. The response should be "True", if it's synced to the TSync-PCIe board.

Using IRIG AM and 1PPS output (from GPIO pin) to sync the TSync-PTP board (using two Standard breakout cables)

The IRIG AM and GPIO 0 output of the standard breakout cable is connected to the IRIG AM and GPIO input of the TSync-PTP board (via the Standard Breakout cable). The IRIG AM signal provides the "Time" reference and the PPS signal from the GPIO output pin provides the "PPS" reference that are needed for the TSync-PTP board to declare snc.

Note: The reason for the IRIG AM being combined with 1PPS from the GPO output is the IRIG AM signal does not have a crisp ontime point, so the 1PPS from the GPIO pin is used to provide a good ontime point for the TSync-PTP board.


BNC connector (P6)
IRIG AM
BNC connector (P3)
TSync-PCI board (synced to GPS)
Standard breakout cable
Standard breakout cable
TSync-PTP board
"1PPS"
GPIO output 0 pin needs to be configured as 1PPS output. See comment below.
D connector (P2) BNC connector (P5)
Signal Pin ---- BNC Pin
GPIO output 0 6 ----- Center (1PPS Input)
Ground 1 ----- Shield (Ground)

Necessary TSync configurations

Note: All configurations need to be performed after each boot-up of the boards.

In the TSync-PCIe board, need to reconfigure the GPIO output 0 as a 1PPS output

In the TSync "GPO" Section of this document, refer to: **Desire to convert the GPO outputs to 1PPS outputs

Need to add a new entry of **IRIG AM (ira 0)** / **GPI 0** to the Reference Priority table

Set the year in the TSync-PTP board:

By factory default, IRIG output does not provide year information. To set the year in the TSync-PTP board, need to either:

Use the **CS_SetYear 0 xxxx** API call/Example program in the TSync-PCIe-PTP board after each boot-up **OR**

Reconfigure the IRIG output of the TSync-PCIe board to include year information in the control field. Then, also need to reconfigure the TSync-PTP board to be able to read the year in the Control field. Refer to:Setting the year value upon each TSync-PCIe power-up (In TSync "Inputs" section).

Refer to the TSync Driver Guide for information on configuring the TSync-PTP board as a PTP Master.

To verify TSync-PTP is synced to the TSync-PCIe board:

Verify the Green LED on the edge of the board is lit.

Use the example program **SS_GetSync 0** on the TSync-PTP board. The response should be "True", if it's synced to the TSync-PCIe board.

Accuracy of IRIG input (ontime point of TSync-PTP board in relation to the ontime point of the TSync-PCIe board) **Refer to:** "IRIG input Accuracy" in the custserviceassistance document (when the TSync-PTP board is indicating that it's in sync with the TSync-PCIe board, as determined by the methods listed above).

**MAC Address for TSync-PCIe-PTP

Email from Denis (4/25/12)
The MAC Address is stored in the EEPROM. We can change it, but customers can't. We generate it uniquely based on the Serial Number of the PTP card.

Don't be confused by the Clock Identity, which is based on the MAC address.

MAC Address:
11:22:33:44:55:66

Generates a Clock Identity of:
11:22:33:FF:FE:44:55:66

Issue: MAC address is all 0's (zeroes)/MAC address has been lost

From the same email above, from Denis
On the other hand, if the MAC address is all 0's (and other fields have bad data), it may be a sign that the EEPROM is bad.

A second email from Denis (4/25/12)
The PTP module is out to lunch. It is no longer communicating with the KTS, and is probably halted. I'm not really sure why.

The KTS resets all the module parameters to "0" when it boots up. It then asks the PTP module to fill that all in. If the PTP module never responds, all that stuff stays "0".

One thing I will say is that when the module first boots up, if DHCP is configured it will try to find a DHCP address right away for 30 seconds to a minute, before the rest of the module boots up. This means that right after power-up, it may take some time for those fields to fill. But I am assuming this is not your problem, since the fields will eventually fill in.

I'm afraid there's not much to be done for it. Reboot it?

I see that it has recent firmware. During development we had a few PTP modules stop communicating with the pre-release firmware but I haven't seen this problem since we've gone to production. I will ask Michel for his thoughts.

General notes

I believe version 2.10 and above for the Linux driver support the **PTP (PTR) calls/example** programs.

A TSync-PCIe-PTP Master can't ever be synced to GPS because the PTP module replaces the GPS receiver (can't have both a GPS receiver and a PTP module installed)

Refer to the TSync-PCI driver guide for an extensive discussion on how to set up the PTP boards (1219-5001-0050) in Arena: https://app.bom.com/items/detail-spec?item_id=1203375449&version_id=10222044888&orb_msg_Single_Search_p=1&redirect_Seqno=7069578002 ("Operation of the TSync-PCIe-PTP" in Section 5).

TSync-PCIe boards are **factory configured as PTP Slaves** (can be re-configured in the field as PTP Masters with the use of PTP API calls).

TSync-PTP Master must be "in sync" for TSync- PTP Slave to sync to it. TSync-PTP Master can be synced to IRIG input. Or, it can also be synced using either the Host/Self or Self/Self input references.

Get PTP module firmware version: **PTR_GetModuleInfo** API call/example program (or use the QuickPTP GUI).

Save Settings to ROM: After making any configurations specifically to the PTP Option Card configuration pages that you wish to persist through reboots/power cycles, the settings should be "Saved to ROM". API call for this function is **PTR_SaveSettingsToROM()**

TTL (Time To Live) Default is "1". This value is decremented on the input of each switch and router. If the TTL value is "0" after it's decremented, this will likely prevent the switch from forwarding the PTP packets.

Email to a customer (on 11/23/11)
The PTP outputs are multicast messages. The specs recommend multicast traffic remain on the immediate subnet only, to prevent potentially flooding the entire network with PTP packets. For this reason, the factory default TTL (Time To Live) value is set to "1". Depending on the internal operation of the switch, this will often prevent the switch from passing any PTP traffic. You may need to increase the TTL value of the PTP Master to a higher number than 1 (up to a max of 255), to allow the switch to pass the multicast traffic to other subnets.

Delay mechanisms (P2P/E2E Transparent clock delay compensation)

Delay mechanism was implemented in PTP Module firmware version 1.10.

Supports both E2E and P2P delay mechanisms.

Earlier firmware versions of the TSync-PCIe board's PTP module (when the TSync-PCIe board is configured as a PTP Slave) did not support Transparent Clock delay compensations. This capability to account for the calculated delays added in the PTP module firmware version 1.1.0.

To determine if the TSync-PCIe PTP module's firmware supports transparent clock delay compensations, perform a **TSYNC_PTR_GetModuleInfo** API call. If it responds with version 1.0.0, the PTP module will need to be update to version 1.1.0 or higher in order to be able to observe the affects using different PTP aware transparent clocks with the TSync-PCIe boards.

If the PTP firmware version is 1.0.0, the update file can be sent. The instructions to perform the firmware upgrade are included in the latest version of the driver guide (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611475121

(Refer to Section 5.7.4 (page 5-249) (in Rev H, anyways) for more information on the firmware update process. Refer to I:\Customer Service\Product Service Bulletin, PSP and other software updates\TSYNC boards\PTP Module updates\V1.1.0 (ECN 2740 transparent clock) for the link to the 1.1.1 upgrade file.
Note: The PTP update can be applied to any TSync-PTP board without needing to update either the driver or the TSync-PCIe's firmware.

P2P/E2e Support

➢ API Call to enable/configure delay mechanism: "**PTR_SetPortSettings**"

Delay Mechanism: o

- 0x01 = End-to-End Delay Mechanism (**E2E**)
- 0x02 = Peer-to-Peer Delay Mechanism (**P2P**)
- 0xFE = Delay Mechanism disabled

TSync-PCIe-Issue with P2P delay mechanism when in One step Mode only- two step mode is OK) According to Denis (as of least 11/29/11) Les Ulis found a bug with P2P (Peer to Peer mode), when PTP is in the one step mode (not an issue if in "E2E" - End to End mode).

Email from Michel (4 March 2013)
Yes, I confirm that TSync-PTP supports both P2P and E2E delay mechanisms.

No, the issue in P2P/One step mode is not fixed and will not be.

The purpose of the P2P delay mechanism is to reach a higher accuracy than the E2E delay mechanism, using specific IEEE1588 compatible switches/routers.
The present implementation of the One Step mode in the Zürich PTP Stack is based on a software timestamping of the PTP events.
The timestamps, so produced, are not accurate at all.
So the One Step mode, as presently implemented, invalidates the interest in using the P2P delay mechanism.
It doesn't make sense using both P2P delay mechanism and One Step mode with our PTP Stack.

That's why we decided not to fix the issue.
The only two step mode makes sense to be used with the P2P delay mechanism at this time.

Input references for a PTP Master/ Clock source field

The PTP module will send valid time packets whether TSync-PCIe is synced to a hardware reference (IRIG input) or if it's manually set using the host or self references.

However, when the TSync is synced to a hardware reference (IRIG input), the clock source field will report "GPS" in the packets to the slaves. However, when the TSync-PCIe is synced to self, the Clock PTP module source a different value in the clock source field.

Yes! The TSync-PTP Master can successfully distribute PTP time with the Self reference. However, the Clock source field won't indicate "GPS" as it normally does with other input references besides self reference. But because this field is so low in the BMC algorithm, it's highly unlikely to have any effect on the operation of the PTP slaves.

PTP Unicast/Multicast modes

As of (10/24/11), we only support PTP multicast mode, but John Fisher believes we will support unicast mode in the TSyncs by end of the 201**?** (We do plan to support it with SecureSync and believe it will carry over to TSyncs, also).

Update (4 March 2013) TSync-PTP still only supports PTP Multicast mode. Not known if/when Unicast mode will be implemented in TSync-PCIe.

PTP associated API Calls (from TSync-PCIe Driver Guide)

The API calls associated with PTP Input and output (Master and Slave modes) are the "PTR" calls. PTR calls control and process decoded PTP network packets (either as an input reference or a time output).

Typical response times for the PTR-associated API calls/example programs.

The PTP module runs a continuous "software super-loop" of all the info reported in a PTR call. It typically takes about one millisecond or so for this "super-loop" to get to the requested "PTR_" data.

Q My second question is related to the time needed by the getPortState function. Is it normal for it to take around 1.2 ms? If yes, why is the time taken so long for only retrieving a status?

A Keith's response: Yes. This is expected/typical response time for all of the PTP-associated calls/functions (the "PTR_ calls). The "PTR" calls obtain responses from the on-board PTP module. The PTP module runs a continuous software loop that contains the PTR-associated information. Its takes about 1 ms for the loop to continue to where the requested data is in the loop and to provide the requested info to the TSync board (unlike the hardware calls and all of the other non-PTR software calls that are generated within the TSync board itself).

Example PTR calls

1. **TSYNC_PTR_GetModuleInfo** Gets the PTP module's version and build date.
2. **TSYNC_PTR_GetEthernetItf** Gets Ethernet settings for the PTP Module.

3. **TSYNC_PTR_SetEthernetItf** Sets Ethernet settings (IP address, Subnet mask and gateway addres) for the PTP Module.

Note: When enabling DHCP, the rest of the **PTPEthernetItfObj** structure still needs to be populated. This data will be retained by the TSync-PCIe, but it will not be used as long as DHCP is enabled.

Note: After changing Ethernet Settings, please reset the module (using **TSYNC_PTR_resetModule**)

4. **TSYNC_PTR_GetUnitSettings** Gets general PTP settings for the PTP Module.
5. **TSYNC_PTR_SetUnitSettings** Sets general PTP settings for the PTP Module.

6. **Clock Identity:** A unique identifier for every PTP device. Consists of eight 8-bit octet fields. By default, fields 1, 2, and 3 match the first three octets of the six-octet MAC address, fields 4 and 5 are “FF” and “FE”, respectively, and fields 6, 7, and 8 match the last three octets of the MAC address.

7. **One-Step Mode**:
   - “N”: operates in two-step mode (default value).
   - “Y”: operates in one-step mode (this will reduce accuracy).
8. **Unicast**: Set to “N” by default, operating in Multicast mode. O (*Unicast mode is not currently supported).*

9. **Domain Number**: Reports the PTP Domain (defaults to 0) (See PTPv2 Sec. 7.1)

Priority1: Reports the PTP Priority1 value (defaults to 128) (See PTPv2 Sec. 7.6.2.2)

Priority2: Reports the PTP Priority2 value (defaults to 128) (See PTPv2 Sec. 7.6.2.3)

TSYNC_PTR_GetPortState: Gets the current state of the PTP port.

10. Port Number (always a “1”),
11. Port enabled (“always a “Y”),

Port state Reports the current state of the PTP State Machine (one of the following).

- **Initializing**: Cable is unplugged / power-up state. Listening: TSync/PTP is looking for a Master.
- **Master:** TSync/PTP Master has become the active master on the network.
- **Passive**: TSync/PTP Master has become a passive master. (There is another Master on the network with higher priority)
- **Uncalibrated**: TSync/PTP Slave has detected a Master on the network.
- **Slave**: TSync/PTP Slave is actively synchronizing to a Master on the network.
- **Link connected** (“Y” is Ethernet cable connect. “N” is if not connected.

TSYNC_PTR_GetPortSettings Gets configuration information for the PTP port.

TSYNC_PTR_SetPortSettings Sets configuration information for the PTP port.

12. **Port Number**: For this product, will always be “1”.

Announce Reception Timeout: In order for a slave to synchronize to a Master, it must see at least one announce message coming in during an interval defined as the slave’s Announce Interval times the slave’s Announce Reception Timeout. Example: if a slave’s Announce Reception Timeout is 3, and its Log Announce Interval is 1, it must see an Announce message within 3 * (21) = 6 seconds.

Log (base 2) of Announce Interval: A slave uses this value combined with the Announce Reception Timeout to determine its announce timeout. A Master uses this value to determine the rate at which it sends out Announce messages.

Log (base 2) of Sync Interval: A Master uses this value to determine the rate at which Sync messages are transmitted.

Log (base 2) of Delay Request interval: A Master will broadcast this value to Slaves to determine the rate at which Delay Request messages are transmitted. (when the End-to-End Delay Mechanism is chosen).

Log (base 2) of Peer Delay Request interval: A Master will broadcast this value to Slaves to determine the rate at which Peer Delay Request messages are transmitted (when the Peer-to-Peer Delay Mechanism is chosen).

Delay Mechanism: o

- 0x01 = End-to-End Delay Mechanism (**E2E**)
- 0x02 = Peer-to-Peer Delay Mechanism (**P2P**)
- 0xFE = Delay Mechanism disabled

Several parameters are represented as the logarithm to base 2 of the number of seconds between packets. For exampleS:

- **Value 0 = 20** = 1 second between packets
- **Value 1 = 21** = 2 seconds between packets
- **Value 0 = 24** = 16 seconds between packets
- **Value -2 = 2(-2) = (1/4)** = .25 seconds between packets

TSYNC_PTR_GetClkQuality: Gets the PTP module's reported clock quality information.

TSYNC_PTR_GetTimeProperties Gets the module's reported time properties information

TSYNC_PTR_GetParentProperties Gets the module's parent properties dataset

TSYNC_PTR_GetGrandmasterProperties Gets the module's Grandmaster Properties dataset.

Clock Identity: the GrandMaster's Clock Identity (see description in Unit Settings).

Clock Class: a number describing the state of the clock (see PTPv2 Table 5 of Section 7.6.2.4 which is pasted below.

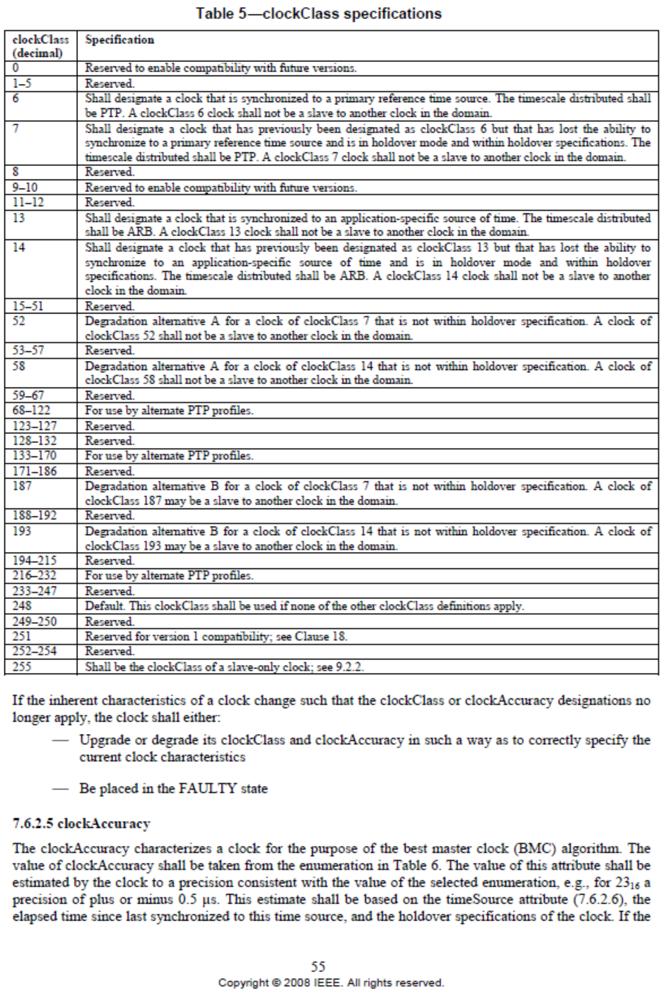


Table 5—clockClass specifications

clockClass (decimal)	Specification
0	Reserved to enable compatibility with future versions.
1–5	Reserved.
6	Shall designate a clock that is synchronized to a primary reference time source. The timescale distributed shall be PTP. A clockClass 6 clock shall not be a slave to another clock in the domain.
7	Shall designate a clock that has previously been designated as clockClass 6 but that has lost the ability to synchronize to a primary reference time source and is in holdover mode and within holdover specifications. The timescale distributed shall be PTP. A clockClass 7 clock shall not be a slave to another clock in the domain.
8	Reserved.
9–10	Reserved to enable compatibility with future versions.
11–12	Reserved.
13	Shall designate a clock that is synchronized to an application-specific source of time. The timescale distributed shall be ARB. A clockClass 13 clock shall not be a slave to another clock in the domain.
14	Shall designate a clock that has previously been designated as clockClass 13 but that has lost the ability to synchronize to an application-specific source of time and is in holdover mode and within holdover specifications. The timescale distributed shall be ARB. A clockClass 14 clock shall not be a slave to another clock in the domain.
15–51	Reserved.
52	Degradation alternative A for a clock of clockClass 7 that is not within holdover specification. A clock of clockClass 52 shall not be a slave to another clock in the domain.
53–57	Reserved.
58	Degradation alternative A for a clock of clockClass 14 that is not within holdover specification. A clock of clockClass 58 shall not be a slave to another clock in the domain.
59–67	Reserved.
68–122	For use by alternate PTP profiles.
123–127	Reserved.
128–132	Reserved.
133–170	For use by alternate PTP profiles.
171–186	Reserved.
187	Degradation alternative B for a clock of clockClass 7 that is not within holdover specification. A clock of clockClass 187 may be a slave to another clock in the domain.
188–192	Reserved.
193	Degradation alternative B for a clock of clockClass 14 that is not within holdover specification. A clock of clockClass 193 may be a slave to another clock in the domain.
194–215	Reserved.
216–232	For use by alternate PTP profiles.
233–247	Reserved.
248	Default. This clockClass shall be used if none of the other clockClass definitions apply.
249–250	Reserved.
251	Reserved for version 1 compatibility; see Clause 18.
252–254	Reserved.
255	Shall be the clockClass of a slave-only clock; see 9.2.2.

If the inherent characteristics of a clock change such that the clockClass or clockAccuracy designations no longer apply, the clock shall either:

— Upgrade or degrade its clockClass and clockAccuracy in such a way as to correctly specify the current clock characteristics

— Be placed in the FAULTY state

7.6.2.5 clockAccuracy

The clockAccuracy characterizes a clock for the purpose of the best master clock (BMC) algorithm. The value of clockAccuracy shall be taken from the enumeration in Table 6. The value of this attribute shall be estimated by the clock to a precision consistent with the value of the selected enumeration, e.g., for 23_{16} a precision of plus or minus 0.5 µs. This estimate shall be based on the timeSource attribute (7.6.2.6), the elapsed time since last synchronized to this time source, and the holdover specifications of the clock. If the

Clock Accuracy: a number describing the accuracy of the oscillator in the Grandmaster (see PTPv2 Spec Section 7.6.2.5).

Offset Scaled log variance of the Master (see PTPv2 Spec Section 7.6.3).

Priority1 setting of the Grandmaster.

Priority2 setting of the Grandmaster.

TSYNC_PTR_SaveSettingsToROM Saves any settings that have been changed in the PTP module to the module's ROM.

TSYNC_PTR_ResetModule Resets the PTP Module.
0 = Cold reset
1 = Warm reset
2 = Restore Factory Defaults and Reset

TSYNC_PTR_GetNumInst Gets the number of PTP instances in the system.

TSYNC_PTR_reinitModule Reinitializes the PTP stack in the PTP module.

TSYNC_PTR_GetValidity Get the PTP validity structure (when used as a PTP Slave).

TSYNC_PTR_GetMode Gets the PTP module's current operational mode (Master or Slave)
0 = PTP Slave Mode
1 = PTP Master Mode

TSYNC_PTR_SetMode Sets the PTP module's current operational mode
0 = PTP Slave Mode
1 = PTP Master Mode

Note: After changing the operational mode, please reset the module.

TSYNC_PTR_GetMacAddr Gets the PTP module's current MAC Address
TSYNC_PTR_GetModuleStatus Gets the PTP module's status information.

Module Status reports the cause of the last reset operation.

TSYNC_PTR_GetUserDesc Gets the PTP User Description strings

TSYNC_PTR_SetUserDesc Gets the PTP User Description strings

**PTP configurations for TSync-PCIe

Changing the network settings of an PTP Option Card or TSync-PCIe-PTP board

When changing the PTP network settings, Save Settings to Rom. Then perform a cold reset of the PTP module (or reboot the SecureSync) to start using the new network settings:

The first step that should always be performed after one or more configuration changes have been made to the PTP Option Card (if it's desired for the configuration changes to persist through a power cycle of the SecureSync) is to "Save Settings to ROM". To perform this step, navigate to the "Module Setup" tab and change the "Save Settings to ROM" field to "Enabled". Then hit Submit. This step saves all of the current PTP card configurations to memory, so that they are restored after each reboot of the SecureSync.

The second step needed to be performed after changing specifically the network configuration in order for the Option Card to start using the new network settings is either the SecureSync needs to be rebooted or the PTP Module just needs a "Cold Reset" be performed. Resetting only the PTP Module allows the new network settings to take effect without having to reboot the entire unit. The PTP Module can be cold reset in the same page of the browser as the "Save Settings to ROM" ("Module Setup" tab).

To perform a cold rest of the PTP module, simply change the "Reset the Module" field to Enabled and change the "Reset Type" field to "Cold Reset". Then, press the Submit button. This will cause the PTP module alone to reset. Moments later when it's operational again, it will be using the new network settings you had assigned it.

Default PTP configuration recommendations from Michel Reyverand (Les Ulis Engineer) as of 11/28/11 (these may change when Peer-to-Peer becomes available):

The appropriate PTP parameters should be:
PTP protocol : Two Steps

Announce interval (multicast)	: 1 (log$_2$ seconds)
Sync interval (multicast)	: 0 (log$_2$ seconds)
Delay Req interval (multicast)	: 4 (log$_2$ seconds)
Min Pdelay Req interval	: 0 (log$_2$ seconds)
Announce receipt timeout	: 3 (announce intervals)
Delay mechanism	: End-to-End

This configuration can be set by commands through the PTP Management serial link, or using the quickPTP tool.

Log Sync message- The interval at which the sync message is sent

N^2 or N^{-2} power = how often sync message is sent.
2^2 = once every 4 seconds
2^{-2} = 4 times per seconds

Note: Per Denis Reilly, the recommended max is about one Sync message per second. Otherwise just flooding the network and possibly overwhelming the Slaves with too much math/calculations.

As a PTP Slave (factory default mode) and with no network switches in between it and the PTP Master, the SecureSync/TSync-PCIe-PTP board should be able to sync to a PTP Master with no configuration changes necessary. A switch will require network settings to be configured.

"**Save Settings to ROM" (applicable to archive versions 4.8.6 and below)** Use this selection when its desired for PTP module configurations to persist through reboots/power cycles

With the exception of network settings, other PTP module configuration changes do not require a reboot of either the module or the TSync-PCIe/SecureSync. After reconfiguring network settings, at a minimum, the PTP module needs to be reset (or the unit rebooted/power cycled).

Network Port value: This is not the UDP port number for PTP (ports 319 and 320). This field is a variable in the PTP specs. It should normally be left as the default value of 1. Do not change this to 319 or 320

Network Domain Number: Allows the ability to set up VLAN (virtual LAN) on the PTP network (PTP Masters and Slaves on "0" and other PTP Masters and Slaves on "1" for "isolation").

Delay mechanism (choices are **P2P, E2E** or **disabled)** P2P stands for Peer to Peer mode. E2E stands for End to End mode. Both are methods for calculating packet delays.

**PTP Multicast Ethertype/Transport protocol (IPv4 or 802.3/Ethernet)

Q. We are seeing ethertype IP (0x800) on the PTP mcast and are curious why isn't this ethertype PTP (0x88F7)? Is this expected? IE: does the device send 0x800 versus 0x88F7? Is that something that can be altered?

Email from Michel:
There are two ways to transmit the PTP protocol:

- PTP over IPv4 protocol (also referred as the layer 3 PTP protocol). In this case, the Ethernet Type of the PTP frames is 0x0800.
- PTP over 802.3/Ethernet protocol (also referred as the layer 2 PTP protocol). In this case, the Ethernet Type of the PTP frames is 0x88F7.

A Email from Denis: The Annex F that Paul mentions refers only to the PTP over 802.3/Ethernet protocol.

In the PTP web pages (PTP SETUP), the way to transmit the PTP protocol can be configured in the field “Transport Protocol” of the “Network” tab.

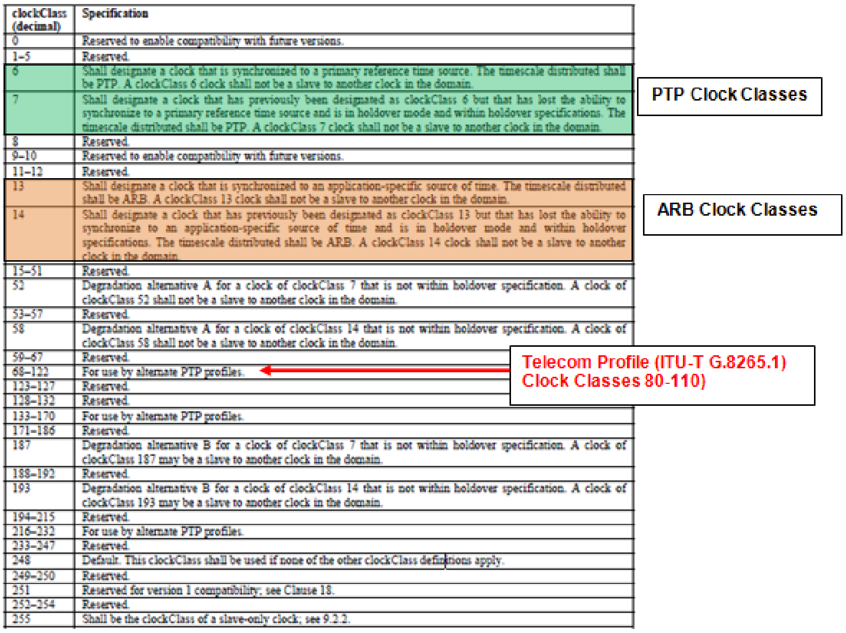
Email from Denis: The Spec states that the Ethertype when using Layer 3 / IPV4 is 0x0800, and when using 802.3/Ethernet the Ethertype is 0x88F7

Active Master Selection process (Passive Master when not selected)

Consists of the following three factors (in descending order of precedence)

Quality level (ClockClass attribute of the Announce message)

Table 5—clockClass specifications



clockClass (decimal)	Specification
0	Reserved to enable compatibility with future versions.
1–5	Reserved.
6	Shall designate a clock that is synchronized to a primary reference time source. The timescale distributed shall be PTP. A clockClass 6 clock shall not be a slave to another clock in the domain.
7	Shall designate a clock that has previously been designated as clockClass 6 but that has lost the ability to synchronize to a primary reference time source and is in holdover mode and within holdover specifications. The timescale distributed shall be PTP. A clockClass 7 clock shall not be a slave to another clock in the domain.
8	Reserved.
9–10	Reserved to enable compatibility with future versions.
11–12	Reserved.
13	Shall designate a clock that is synchronized to an application-specific source of time. The timescale distributed shall be ARB. A clockClass 13 clock shall not be a slave to another clock in the domain.
14	Shall designate a clock that has previously been designated as clockClass 13 but that has lost the ability to synchronize to an application-specific source of time and is in holdover mode and within holdover specifications. The timescale distributed shall be ARB. A clockClass 14 clock shall not be a slave to another clock in the domain.
15–51	Reserved.
52	Degradation alternative A for a clock of clockClass 7 that is not within holdover specification. A clock of clockClass 52 shall not be a slave to another clock in the domain.
53–57	Reserved.
58	Degradation alternative A for a clock of clockClass 14 that is not within holdover specification. A clock of clockClass 58 shall not be a slave to another clock in the domain.
59–67	Reserved.
68–122	For use by alternate PTP profiles.
123–127	Reserved.
128–132	Reserved.
133–170	For use by alternate PTP profiles.
171–186	Reserved.
187	Degradation alternative B for a clock of clockClass 7 that is not within holdover specification. A clock of clockClass 187 may be a slave to another clock in the domain.
188–192	Reserved.
193	Degradation alternative B for a clock of clockClass 14 that is not within holdover specification. A clock of clockClass 193 may be a slave to another clock in the domain.
194–215	Reserved.
216–232	For use by alternate PTP profiles.
233–247	Reserved.
248	Default. This clockClass shall be used if none of the other clockClass definitions apply.
249–250	Reserved.
251	Reserved for version 1 compatibility; see Clause 18.
252–254	Reserved.
255	Shall be the clockClass of a slave-only clock; see 9.2.2.

PTP Clock Classes

ARB Clock Classes

Telecom Profile (ITU-T G.8265.1)
Clock Classes 80-110)

PTSF (Packet Timing Signal Fail)

(PTSF-lossSync, PTSF-lossAnnounce, PTSF-unusable);
This clause defines the notion of packet timing signal fail (PTSF), which corresponds to a signal indicating a failure of the PTP packet timing signal received by the slave.

Three types of PTSF may be raised in a slave implementation:

[PTSF-lossSync], lack of reception of PTP timing messages from a master (loss of the packet timing signal): if the slave does not receive anymore the timing messages sent by a master (i.e., *Sync* and eventually *Follow_up* and *Delay_Resp* messages), then a

PTSF-lossSync associated to this master must occur. A timeout period (e.g., "syncReceiptTimeout" and "delayRespReceiptTimeout") for these timing messages must be implemented in the slave before triggering the PTSF-lossSync (the range and default value of this timeout parameter are for further study).

[PTSF-lossAnnounce], lack of reception of PTP *Announce* messages from a master (loss of the channel carrying the traceability information): if the slave does not receive anymore the *Announce* messages sent by a master, then a PTSF-lossAnnounce associated to this master must occur. A timeout period for these *Announce* messages must be implemented in the slave before triggering the PTSF-lossAnnounce (the range and default value of this timeout parameter are as per [IEEE 1588]). This timeout corresponds to the "announceReceiptTimeout" attribute specified in [IEEE 1588], clause 7.7.3.1.

Priority values (1 to 128, with 1 being the highest priority)

**Delay Mechanism

Q. Is there a downside to using one method vs. the other (P2P versus E2E)? From your documentation it seemed like P2P was preferred?

A. Reply from Denis Reilly: P2P can be better, because in theory you are measuring the delay of each path separately, which will minimize problems if packets take different paths. But this requires that all of your network elements in the path know how to exchange peer delay messages. I don't think your equipment can do that.

Given your network topology, I don't think it will matter for you, but it's my experience that E2E is a bit more common, and would probably interoperate with more of the Slaves that are available right now

***PTP Network Access list (to limit the PTP clients that can get time from the PTP Master)

Q .Dave has a customer (Josh Foshee from Logix) that is interested in setting up network access from the Master to its respective slaves. I've left a message with him trying to gather more information regarding his request and stated at this time the only way we can control access is by setting up the same domain numbers on the masters and slaved and possibly setting up a narrow / restrictive network mask.

With PTP do we have the ability to update the software to setup an access control list similar to SNMP or NTP, if so we can enter a Mantis Case?

Reply from Denis Reilly (9 Sept 13) There is no capability to add an access control list in either of our PTP implementations. You can add a ticket if you want, but I don't know whether it would be possible.

Forced Holdover mode (for PTP Slave mode only- no longer available)

Added in SecureSync version 4.8.6 (June 2012)

Places Slave in Holdover mode, even though its receiving synced packets from the PTP Master

Removed in SecureSync version 4.8.7

Email KW sent to Rob Amos with ASX/Open Access

I have some additional information for you regarding the “Forced Holdover” mode that has been added in the SecureSync version 4.8.6 software update. In summary, this mode was added to the software for a very specific application.

The “Forced Holdover” mode can be enabled when the SecureSync PTP Option Card is configured to be a PTP Slave. With normal PTP operation, the PTP Slave will discipline itself to the PTP Master. When “Forced Holdover” mode is enabled in the PTP slave, the PTP slave will then stop disciplining to the PTP Master, placing the PTP Slave into a Holdover mode. The PTP Master is still indicating to the PTP Slave that it's in sync, but this mode causes the PTP Slave to treat the Master as it being out of sync. The PTP Slave will no longer make any adjustments via disciplining.

The specific application this was designed for was for select customers that desire not to have any correction applied to the PTP slave's outputs, based on inputs from a PTP Master. When using the PTP Slave's outputs as a side-by-side comparison to an output from another device, these customers want the “reference” to not make any deviations, so any variations between the two signals can be determined to be a direct factor of the other output, and not a factor of the reference being supplied by the PTP slave being disciplined to its reference, as well.

I asked Michel with Spectracom France what type of customers would use this mode. He responded with the following:

For instance, there's a special need for STA-61 project where PTP protocol is used for time measurements.
Some time measurement procedures need to have a PTP Slave module in Holdover mode.

DXC (Circuit switches, working on TDM streams)

1) **Email from Emmanuel to Frédéric (24 Mar 2014)**

The network drawing shows the presence of DXC (Circuit switches, working on TDM streams), which impact / compatibility with PTP is difficult to evaluate. There probably the same difficulty, in terms of interoperability, with a MPLS switch.

So the customer support team will do his best to answer the questions, but I would strongly suggest that the customer tests the operation of both SecureSync master and slave when they are on the same LAN (connected to the same Ethernet swith – a simple, straight forward configuration), and then come back to their network configuration.

We cannot drive any conclusion before we understand the impact of the network (in particular this DXC network elements) on the PTP operation, I believe ; This will also allow us to evaluate if there's a failure on one of the SecureSync.

DSCP (Differentiated Services Code Point) / QoS (Quality of Service)

Refer to sites such as http://technet.microsoft.com/en-us/library/cc787218%28v=ws.10%29.aspx

From the link above: Differentiated Services Code Point (DSCP) is a field in an IP packet that enables different levels of service to be assigned to network traffic. This is achieved by marking each packet on the network with a DSCP code and appropriating to it the corresponding level of service.

How DSCP works
Quality of Service (QoS)-enabled programs request a specific service type for a traffic flow through the generic QoS (GQoS) application programming interface (API). The available service types are:

Guaranteed service Guaranteed service provides high quality, quantifiable guarantees with bounded (guaranteed minimum) latency.

Controlled load service Controlled load service provides high quality, quantifiable guarantees without bounded latency.

Email from Denis Reilly (16 Sept 2013) referring to 1204-12 (this is prior to release of the 1204-32) A quick google on DSCP indicates that it might have something to do with tagging packets for Quality of Service purposes. We only support VLAN tagging, we don't support any QoS stuff inside the card. Customers may be able to add it themselves in their routers, though.

Response from Amancio Salas (16 Sept 2013) DSCP Differenciated Service Code Point. This is some bits in the IP_Header of IPv4. We used this to give priority to some traffic from another on the network.

QoS (Quality of Service)

From Wikipedia: Quality of service is the ability to provide different priority to different applications, users, or data flows, or to guarantee a certain level of performance to a data flow.

PTP Management API interface-Management Message packet /Agilent LXI

Refer to Salesforce Case 7055.

As of at least Oct 2014, we do not support Management messages with any of our products. We just ignore/drop these packets.

Agilent LXI Appears to be an API interface for Test and Measurement equipment.

As of at least Oct 2014, we do not support LXI commands.

Link to LXI Device Specification that Denis Reilly found (9 Jan):

\\rocfnp01\idrivedata\Customer Service\PTP\LXI Device Specification 2011 rev 1.4.pdf

Q. I am currently working with one of your SecureSync units. I am sending PTP management packets to the unit but the unit does not seem to respond. Does the SecureSync reply to PTP management packets (for example, "GET Default Dataset")?

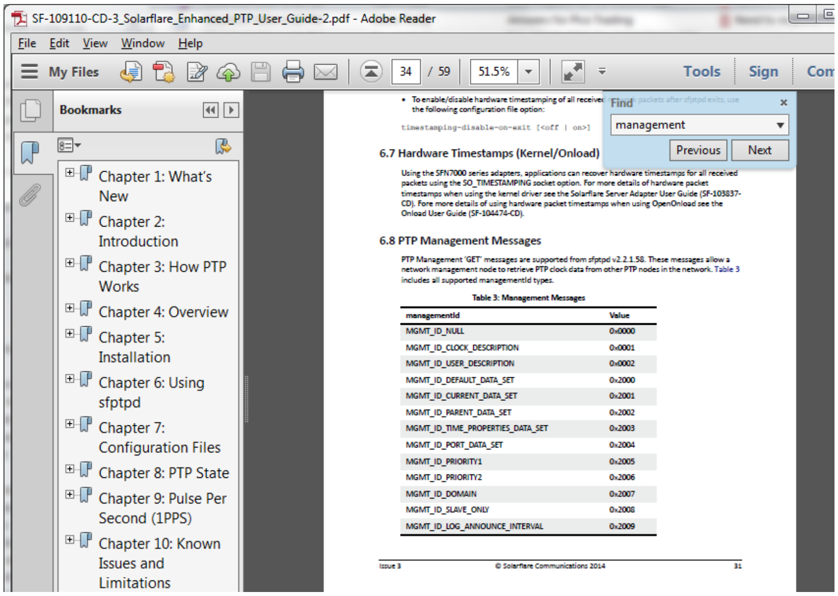
A. Keith's email to Denis Reilly (2 Jan 2013): From what I could gather, LXI appears to be an Agilent-based API interface for Test/Measurement devices. As far as I am aware, we do not currently support this APIU interface. I wanted to confirm this belief with you, before I respond to the customer.
From the link: http://cp.literature.agilent.com/litweb/pdf/5989-6147EN.pdf

LAN eXtensions for Instrumentation (LXI) is a measurement platform based on widely used standards such as Ethernet (IEEE802.3 standard), TCP/IP, Web browsers and IVI drivers. LXI combines the measurement functionality and PC-standard I/O connectivity of standalone instruments with the modularity and compact size of plug-in cards—but without the size or cost of a cardcage.

Programmatic Interface
Because the LXI standard requires that all devices have an Interchangeable Virtual Instrument (IVI) driver, it allows you to use whichever programming language or development environment you prefer. IVI-COM and IVI-C are well-established industry standard drivers that instrument makers supply with their products.

Email from Denis Reilly (9 Jan 2013)
I found the LXI specification, and I believe LXI management messages are the same thing as IEEE 1588 management messages. At least, the Spec makes reference to IEEE 1588 management messages several times, with no apparent reference to any management protocol of its own. 1588 is an integeral part of LXI so it makes sense to me. I am attaching the spec here, and also putting it on Sharepoint in the General PTP directory.

List of PTP Management commands (from a Solarflare manual)



- To enable/disable hardware timestamping of all received network packets after sfptpd exits, use the following configuration file option:

```
timestamping-disable-on-exit [<off | on>]
```

6.7 Hardware Timestamps (Kernel/Onload)

Using the SFN7000 series adapters, applications can recover hardware timestamps for all received packets using the SO_TIMESTAMPING socket option. For more details of hardware packet timestamps when using the kernel driver see the Solarflare Server Adapter User Guide (SF-103837-CD). Fore more details of using hardware packet timestamps when using OpenOnload see the Onload User Guide (SF-104474-CD).

6.8 PTP Management Messages

PTP Management 'GET' messages are supported from sfptpd v2.2.1.58. These messages allow a network management node to retrieve PTP clock data from other PTP nodes in the network. Table 3 includes all supported managementId types.

Table 3: Management Messages

managementId	Value
MGMT_ID_NULL	0x0000
MGMT_ID_CLOCK_DESCRIPTION	0x0001
MGMT_ID_USER_DESCRIPTION	0x0002
MGMT_ID_DEFAULT_DATA_SET	0x2000
MGMT_ID_CURRENT_DATA_SET	0x2001
MGMT_ID_PARENT_DATA_SET	0x2002
MGMT_ID_TIME_PROPERTIES_DATA_SET	0x2003
MGMT_ID_PORT_DATA_SET	0x2004
MGMT_ID_PRIORITY1	0x2005
MGMT_ID_PRIORITY2	0x2006
MGMT_ID_DOMAIN	0x2007
MGMT_ID_SLAVE_ONLY	0x2008
MGMT_ID_LOG_ANNOUNCE_INTERVAL	0x2009

Issue 3 © Solarflare Communications 2014 31

In the case of an error in the execution of a management message, a management error status TLV is to be returned as defined in 15.5.4.

“Custom” Management messages

IEEE 1588 specs provide a designated area in Management messages for “custom” Management messages

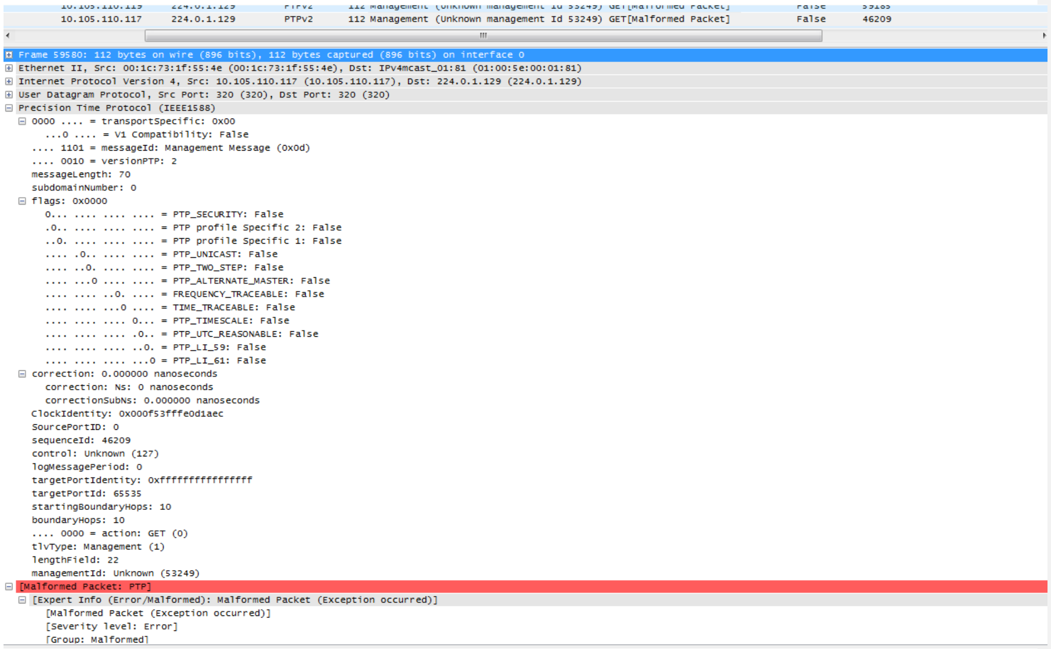
Wireshark/PCAP doesn’t recognize the custom info, so it marks the custom Management packets as “Malformed Packet”

Note: PTP Clients that support Management packets but not the custom message received will send an Error message to the PTP Multicast address when they receive a custom message that wasn’t intended for it to understand. Refer to the example response further below regarding TimeKeeper software customer Management message.

FSMLabs Timekeeper custom Management Message for reporting its status

Info field- “Management (unknown management ID 53249 (GET [malformed packet]”

Example Timekeeper custom Management message from a PTP Client running Timekeeper below:

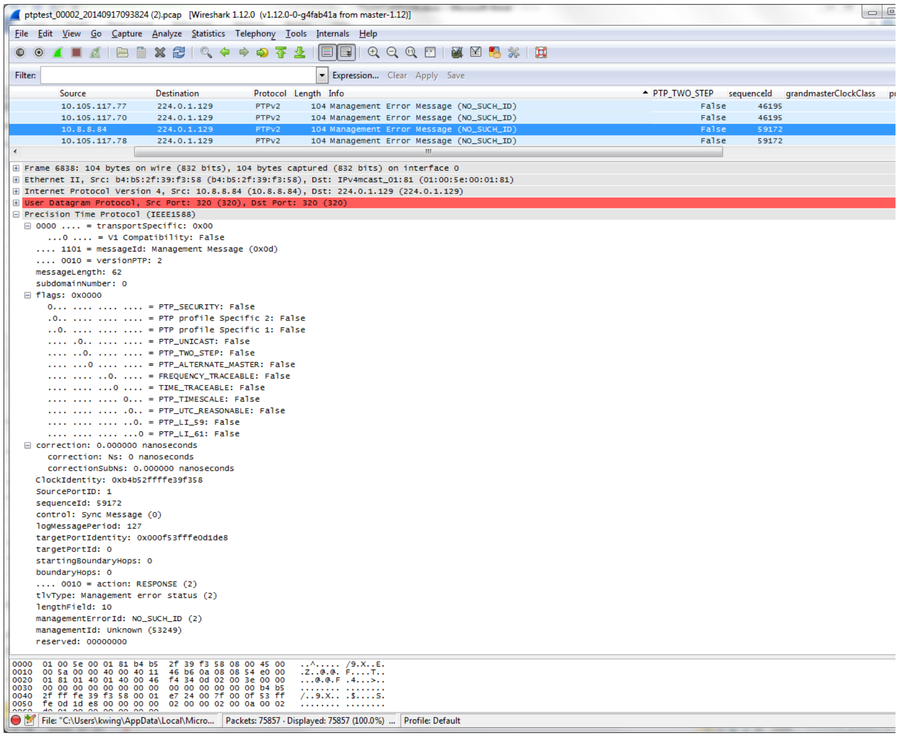


```
    10.105.110.119    224.0.1.129    PTPv2    112 Management (Unknown management Id 53249) GET[Malformed Packet]    False    59185
    10.105.110.117    224.0.1.129    PTPv2    112 Management (Unknown management Id 53249) GET[Malformed Packet]    False    46209

⊞ Frame 59580: 112 bytes on wire (896 bits), 112 bytes captured (896 bits) on interface 0
⊞ Ethernet II, Src: 00:1c:73:1f:55:4e (00:1c:73:1f:55:4e), Dst: IPv4mcast_01:81 (01:00:5e:00:01:81)
⊞ Internet Protocol Version 4, Src: 10.105.110.117 (10.105.110.117), Dst: 224.0.1.129 (224.0.1.129)
⊞ User Datagram Protocol, Src Port: 320 (320), Dst Port: 320 (320)
⊟ Precision Time Protocol (IEEE1588)
  ⊟ 0000 .... = transportSpecific: 0x00
      ...0 .... = V1 Compatibility: False
    .... 1101 = messageId: Management Message (0x0d)
    .... 0010 = versionPTP: 2
    messageLength: 70
    subdomainNumber: 0
  ⊟ flags: 0x0000
      0... .... .... .... = PTP_SECURITY: False
      .0.. .... .... .... = PTP profile Specific 2: False
      ..0. .... .... .... = PTP profile Specific 1: False
      .... .0.. .... .... = PTP_UNICAST: False
      .... ..0. .... .... = PTP_TWO_STEP: False
      .... ...0 .... .... = PTP_ALTERNATE_MASTER: False
      .... .... ..0. .... = FREQUENCY_TRACEABLE: False
      .... .... ...0 .... = TIME_TRACEABLE: False
      .... .... .... 0... = PTP_TIMESCALE: False
      .... .... .... .0.. = PTP_UTC_REASONABLE: False
      .... .... .... ..0. = PTP_LI_59: False
      .... .... .... ...0 = PTP_LI_61: False
  ⊟ correction: 0.000000 nanoseconds
      correction: Ns: 0 nanoseconds
      correctionSubNs: 0.000000 nanoseconds
    ClockIdentity: 0x000f53fffe0d1aec
    SourcePortID: 0
    sequenceId: 46209
    control: Unknown (127)
    logMessagePeriod: 0
    targetPortIdentity: 0xffffffffffffffff
    targetPortId: 65535
    startingBoundaryHops: 10
    boundaryHops: 10
    .... 0000 = action: GET (0)
    tlvType: Management (1)
    lengthField: 22
    managementId: Unknown (53249)
⊟ [Malformed Packet: PTP]
  ⊟ [Expert Info (Error/Malformed): Malformed Packet (Exception occurred)]
      [Malformed Packet (Exception occurred)]
      [Severity level: Error]
      [Group: Malformed]
```

Example Error message from a PTP Client not running Timekeeper that received a TimeKeeper custom Management message being sent out to the multicast address.

Per Michel (22 Sept 2014) The malformed Management GET packets (sent by 10.105.110.117 and10.105.110.119) have no impact because the master is able to parse and reject them (because these packets are not so malformed).

Info field: "Management Error Message (No_Such_ID)"


ptptest_00002_20140917093824 (2).pcap [Wireshark 1.12.0 (v1.12.0-0-g4fab41a from master-1.12)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear Apply Save

```
Source          Destination    Protocol Length Info                                      PTP_TWO_STEP sequenceId grandmasterClockClass
10.105.117.77   224.0.1.129    PTPv2       104 Management Error Message (NO_SUCH_ID)            False      46195
10.105.117.70   224.0.1.129    PTPv2       104 Management Error Message (NO_SUCH_ID)            False      46195
10.8.8.84       224.0.1.129    PTPv2       104 Management Error Message (NO_SUCH_ID)            False      59172
10.105.117.78   224.0.1.129    PTPv2       104 Management Error Message (NO_SUCH_ID)            False      59172
```

```
⊞ Frame 6838: 104 bytes on wire (832 bits), 104 bytes captured (832 bits) on interface 0
⊞ Ethernet II, Src: b4:b5:2f:39:f3:58 (b4:b5:2f:39:f3:58), Dst: IPv4mcast_01:81 (01:00:5e:00:01:81)
⊞ Internet Protocol Version 4, Src: 10.8.8.84 (10.8.8.84), Dst: 224.0.1.129 (224.0.1.129)
⊞ User Datagram Protocol, Src Port: 320 (320), Dst Port: 320 (320)
⊟ Precision Time Protocol (IEEE1588)
  ⊟ 0000 .... = transportSpecific: 0x00
      ...0 .... = V1 Compatibility: False
    .... 1101 = messageId: Management Message (0x0d)
    .... 0010 = versionPTP: 2
    messageLength: 62
    subdomainNumber: 0
  ⊟ flags: 0x0000
      0... .... .... .... = PTP_SECURITY: False
      .0.. .... .... .... = PTP profile Specific 2: False
      ..0. .... .... .... = PTP profile Specific 1: False
      .... .0.. .... .... = PTP_UNICAST: False
      .... ..0. .... .... = PTP_TWO_STEP: False
      .... ...0 .... .... = PTP_ALTERNATE_MASTER: False
      .... .... ..0. .... = FREQUENCY_TRACEABLE: False
      .... .... ...0 .... = TIME_TRACEABLE: False
      .... .... .... 0... = PTP_TIMESCALE: False
      .... .... .... .0.. = PTP_UTC_REASONABLE: False
      .... .... .... ..0. = PTP_LI_59: False
      .... .... .... ...0 = PTP_LI_61: False
  ⊟ correction: 0.000000 nanoseconds
      correction: Ns: 0 nanoseconds
      correctionSubNs: 0.000000 nanoseconds
    ClockIdentity: 0xb4b52ffffe39f358
    SourcePortID: 1
    sequenceId: 59172
    control: Sync Message (0)
    logMessagePeriod: 127
    targetPortIdentity: 0x000f53fffe0d1de8
    targetPortId: 0
    startingBoundaryHops: 0
    boundaryHops: 0
    .... 0010 = action: RESPONSE (2)
    tlvType: Management error status (2)
    lengthField: 10
    managementErrorId: NO_SUCH_ID (2)
    managementId: Unknown (53249)
    reserved: 00000000
```

```
0000  01 00 5e 00 01 81 b4 b5  2f 39 f3 58 08 00 45 00   ..^..... /9.X..E.
0010  00 5a 00 00 40 00 40 11  46 b6 0a 08 08 54 e0 00   .Z..@.@. F....T..
0020  01 81 01 40 01 40 00 46  f4 34 0d 02 00 3e 00 00   ...@.@.F .4...>..
0030  00 00 00 00 00 00 00 00  00 00 00 00 00 00 b4 b5   ........ ........
0040  2f ff fe 39 f3 58 00 01  e7 24 00 7f 00 0f 53 ff   /..9.X.. .$....S.
0050  fe 0d 1d e8 00 00 00 00  02 00 00 02 00 0a 00 02   ........ ........
```

File: "C:\Users\kwing\AppData\Local\Micro... | Packets: 75857 · Displayed: 75857 (100.0%) ... | Profile: Default

Monitoring PTP performance/operation

Monitoring PTP performance/operation

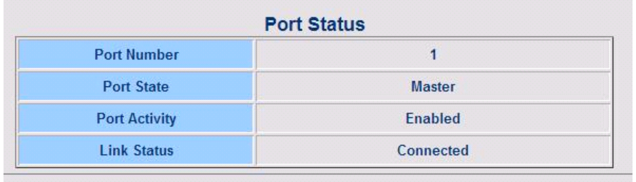
- ➢ Refer to “Monitoring PTP performance/operation” in custserviceassistance doc: ..\CustomerServiceAssistance.pdf

PTP troubleshooting

****Basic troubleshooting info**

Classic interface web browser

Status -> Inputs (or Outputs) -> **PTP**, **Network** tab



Port Status

Port Number	1
Port State	Master
Port Activity	Enabled
Link Status	Connected

- Does “Port State” of the PTP Option Card indicate “Master” or Slave”?
- Does “Port Activity” indicate “Enabled”?
- Does “Link Status” of the PTP Option Card indicate “Connected”?

- Get wireshark packet captures whenever possible (refer to: **WireShark/TCP dump packet capture)
- PTP MULTICAST destination address is **224.0.1.129** (any other address indicates UNICAST mode)
- Are there Sync, Follow-up, announce, Delay Request and Delay Response packets in the capture?
- Max recommended network cable run 100 meters (from one device to another). Longer runs may result in packet losses.
- Is the PTP Slave and all switches in between configured for 10/100 base-T. Make sure none are set to 1000 Base-T when using the Model 1204-12 Option Card or the TSync-PCIe-PTP board.
- Is the PTP Slave in sync?

For a TSync-PTP board as a PTP Slave: is the green LED on the edge of the board lit?

- Is PTP Master in sync (to itself or an external reference)? Is the SecureSync synced to GPS (is the front panel Sync LED solid green?
- (With the exception of NTP sync or SecureSync synced to itself) Does PTP Master have a TFOM better than TFOM 15? If TFOM is 15 when synced to a refernce, PTP packets are not outputted. What is the SecureSync’s TFOM value (As reported in the **Status** -> **Time and Frequency** page of the web browser)?

Note (KW 12/17/12): According to Denis Reilly, the code is written in a way that if SecureSync is synced to Self or to NTP, the PTP Option Card knows the TFOM will be TFOM 15. So if TFOM of 15 is due to these conditions, this won’t prevent the PTP card from outputting packets. But if it’s synced to another reference and TFOM is 15, the PTP Option Card won’t output any packets.

- Network topography (switches in between the Master and Slave)?
- Note that when using PTP Multicast, some switches need special configuration to allow the PTP multicast traffic to get though. Make sure the switch is configured to pass PTP multicast traffic.
- Refer to the following specific indications/conditions below for more info.
- Are the PTP Master, Slaves (and any Cisco switches) all configured to be on the same PTP domain.

One simple thing that came to me while reviewing your configuration: please make sure that your GrandMaster, Slave, and Cisco switch are all set to the same PTP domain. The Cisco switch is set to Clock Domain 0. I believe our equipment defaults to Domain 0 as well, but you may want to check to make sure this hasn't changed – a PTP device will ignore traffic on domains other than the one it is set to.

Some questions for you and your customer, to get a better idea on exactly what is occurring:

- Is the SecureSync synced to GPS (is the front panel Sync LED solid green?
- What is the SecureSync's TFOM value (As reported in the **Status** -> **Time and Frequency** page of the web browser)?
- Have they tried temporarily connecting the TSync directly to the SecureSync using a network cross-over cable to see if the TSync-PCIe board can sync to the SecureSync? If they haven't, I recommend they do so first. This will verify the equipment is operational.
- Have they configured the network settings for both the SecureSync and TSync-PCIe will applicable values for the network they are connected to? Note that with at least one switch in between, the TSync board has to be configured with an appropriate address for the network.
- Are they basing their interpretation on packets not getting through on just the fact that the TSync-PCIe board is not syncing? Or have they verified with a packet capture performed from a PC on that particular subnet that lacks any PTP packets (note that they can't capture packets from a subnet on the other side of the switch)? If they haven't performed a packet capture yet (using wireshark for instance), I recommend they do so. If they can send us the capture, we'll be happy to also review it for them.
- I'm confirming with engineering, but I do not believe the switch being 1000 base T would prevent packets from getting through at all. It may affect the optimal timing of the TSync, but I don't believe this would block any packets from getting through.
- What is the TTL (Time To Live) value configured in the SecureSync (**Setup** -> **Outputs** -> **PTP** page of the browser, Network tab)? Can you have them send us a screenshot for each of the tabs on this page of the browser. If it's still set to 1, I recommend they try increasing it to a number slightly higher than the number of switches in between (if there are only two switches in between, try changing it to a value of "3" instead of "1".
- As the TSync board does not yet support PTP Unicast mode, can you verify the SecureSync is configured for PTP "**Multicast**" mode and not set to Unicast mode (**Network** tab)?

FYI- in their packet captures, the PTP packets from the SecureSync should have a destination address of "224.0.1.129". Any other Destination address value indicates the SecureSync is configured for Unicast mode and should be reconfigured for Multicast mode in order for the TSync-PCIe packets to be able to receive PTP packets (otherwise there will be no packets on the network).

More specific PTP troubleshooting (in addition to the basic troubleshooting discussed above)

Condition 1: PTP Master is in sync but PTP packets are being lost between the PTP Master and Slave (as verified with Wireshark captures performed on same subnet as the Master and the Slave.

Summary:

- A network switch in between may be blocking packets
- A network switch in between may be a misconfigured PTP Boundary clock (not a transparent clock as expected)
- The PTP Master may be in Unicast mode, while the PTP client is in Multicast mode
- Have they configured the network settings for both the SecureSync and TSync-PCIe will applicable values for the network they are connected to? Note that with at least one switch in between, the TSync board has to be configured with an appropriate address for the network.
- What is the TTL (Time To Live) value configured in the PTP Master?
- **Is the PTP Master configured for Multicast mode when using** Some more sophisticated switches have special configuirations to allow multicast traffic to pass.

For a SecureSync-PTP as the Master -As the TSync board does not yet support PTP Unicast mode, can you verify the SecureSync is configured for PTP "**Multicast**" mode and not set to Unicast mode (**Network** tab)?

- What are the Models of all switches/devices in between? If it's actually a PTP boundary switch that isn't configured correctly, it will inherently block the original PTP packets and may not generate/output its PTP packets, resulting in no output packets being sent from that network device.

Condition 2: PTP Master is synced but PTP Slave is not (PTP packets are getting exchanged)

- If the PTP Slave from another vendor isn't syncing to our PTP Master, verify their PTP Slave supports PTPv2 (not PTPv1). We don't support PTPv1 Slaves.
- Verify Sync status of the PTP Master:

What is the Master synced to?
What is its TFOM value?

Condition 3: Delay responses always has 0 for nanoseconds

Issue: Delay response message always has 0 for nanoseconds, and the seconds indicates that the delay request was received BEFORE the master sent out the follow-up message that caused the slave to send the delay request.

Summary: This can be an issue with earlier versions of the PTPd software.

Example report Masataka with TOYO (~2/20/12) SecureSync as PTP Master, using Symmetricom SCi300 as their slaves. The Delay response message sent from the SecureSync PTP Master was indicating it was compatible with PTPv1. This was preventing the SecureSync from being able to fill in the "receivetimestamp (nanosecond)" field with just "0". Told him he would need to disable the delay request message flag that was indicating the PTP slave supports PTPv1.

Email from Denis Reilly:
If the transportSpecific field in a Delay Request is set to a value other than 0, the timestamp that is transmitted in the corresponding Delay Response packet may be incorrect. It is recommended that customers who are using their own clients make sure the value of this field is set to 0 in their Delay Response packets. Customers who are using ptpd from Sourceforge should upgrade to the latest version, where the transportSpecific is set to the proper value. See http://sourceforge.net/p/ptpd/bugs/25/ for more information about this issue in Sourceforge.

Email from Denis (2/21/12)
In case they ask for more details, the flag in question is the "Hardware Compatability" bit in the "transportSpecific" field, as documented in Annex D.4 / Table 123 in the IEEE 1588 specification. We are expecting all bits in the transportSpecific field to be 0.

Known software issues with TSync-PCIe and/or SecureSync PTP

1) **TSync-PCIe -Issue with P2P delay mechanism when in One step Mode only- two step mode is OK) According to Denis (as of least 11/29/11) Les Ulis found a bug with P2P (Peer to Peer mode), when PTP is in the one step mode (not an issue if in "E2E" - End to End mode).**

Update (4 March 2013, email from Michel Reyverand)
Yes, I confirm that TSync-PTP supports both P2P and E2E delay mechanisms.

No, the issue in P2P/One step mode is not fixed and will not be.
The purpose of the P2P delay mechanism is to reach a higher accuracy than the E2E delay mechanism, using specific IEEE1588 compatible switches/routers.
The present implementation of the One Step mode in the Zürich PTP Stack is based on a software timestamping of the PTP events.
The timestamps, so produced, are not accurate at all.
So the One Step mode, as presently implemented, invalidates the interest in using the P2P delay mechanism.
It doesn't make sense using both P2P delay mechanism and One Step mode with our PTP Stack.

- **PTP module stuck in “initializing”** (6/25/12) Bradley Wilson from GE Energy reported “just updated a SecureSync to firmware version 4.8.6 and the PTP module no longer multicast out synch messages. When I look at the output status the port is “initializing”.

Denis Reily emailed me with: I know one way the card can stay in “initializing” state, and it’s bad: if somehow the update hosed the card. Or if the rest software got upgraded, but for some reason the card didn’t – I don’t think the new PTP card software plays well with the old software in the rest of the system.

Potential fix (from Denis): I haven’t tried it yet, but assuming we have new NWP software and old PTP software:
- A forced update to 4.8.P would revert the NWP software without attempting to touch the PTP card
- Then, a forced update to 4.8.6 ***should*** update everything.

- Issues where delay requests are being sent, but delay responses not being returned.

(8 Nov 2012 KW per discussion with Denis Reilly), Denis said there is an issue with PTPd, where they are doing something incorrect (though technically, our PTP Masters should be able to tolerate) which prevents delay responses from being able to be processed. This results in Delay responses not being able to be processed by the PTP clients.

IGMP (v1, v2 and v3)

Source-specific multicast (**SSM**) is a method of delivering multicast packets in which the only packets that are delivered to a receiver are those originating from a specific source address requested by the receiver. By so limiting the source, SSM reduces demands on the network and improves security.

Per Denis Reilly (2 May 14) The SSM feature in v3 means that only source specific multicast messages, which the switch has been told to pass through, will be delivered. All the other multicast packets will be dropped, and that is on purpose, because the intent of SSM seems to be to get rid of unneeded multicast traffic. In other words, the switch assumes that if you didn't program it to specifically allow for that specific traffic, then you must not want it.

When we've seen this happen in the past, the symptom is that delay requests aren't making it back to the master. Packet traces show everything looks good, but the master can't link up with the slave. And if they switched to v2 and that resolved the problem, then it's absolutely a v3 configuration issue on their end.

To make things even more complicated, this SSM (source-specific multicast) has nothing to do with the Sync-E SSM (synchronization status message)

***IGMPv3

IGMPv3 multicast configuration

- Refer to http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipmulti/command/imc-cr-book/imc_i1.html

The following example shows how to configure the interface (loopback 0) to join the PTP multicast group.

```
 Device(config)# interface loopback 0
 Device(config-if)# ip igmp join-group 224.0.1.129
```

**IGMPv3 Snooping:

- Refer to: http://en.wikipedia.org/wiki/IGMP_Snooping

The feature allows a network switch to listen in on the IGMP conversation between hosts and routers. By listening to these conversations the switch maintains a map of which links need which IP multicast streams. Multicasts may be filtered from the links which do not need them and thus controls which ports receive specific multicast traffic

Hi Keith,
It's not that we don't support IGMPv3. It's a matter of configuring the switch to not ignore and drop the multicast packets which will happen if IGMPv3 isn't configured to route them. IGMPv2 improves over IGMPv1 by adding the ability for a host to signal desire to leave a multicast group. IGMPv3 improves over IGMPv2 mainly by supporting source-specific multicast (SSM)

Q. Email from Keith to Denis/Jeremy (2 May 2014) I found the following at this link:
http://www.cisco.com/c/en/us/td/docs/ios-xml/ios/ipmulti/command/imc-cr-book/imc_i1.html
The following example shows how to configure the interface (loopback 0) to join the PTP multicast group.

```
Device(config)# interface loopback 0
Device(config-if)# ip igmp join-group 224.0.1.129
```

Is this all that needs to be done to support IGMPv3 on both PTP boards?

A Reply from Jeremy-The SSM feature in v3 means that only source specific multicast messages, which the switch has been told to pass through, will be delivered. All the other multicast packets will be dropped, and that is on purpose, because the intent of SSM seems to be to get rid of unneeded multicast traffic. In other words, the switch assumes that if you didn't program it to specifically allow for that specific traffic, then you must not want it.

When we've seen this happen in the past, the symptom is that delay requests aren't making it back to the master. Packet traces show everything looks good, but the master can't link up with the slave. And if they switched to v2 and that resolved the problem, then it's absolutely a v3 configuration issue on their end.

**Interface issues between Spectracom and non-Spectracom PTP devices

- Symmetricom SCi300 PTP Slaves (refer to Salesforce case 4810)

Delay responses are all zeroes

Masataka with TOYO (~2/20/12) SecureSync as PTP Master, using Symmetricom SCi300 as their slaves.
The Delay response message sent from the SecureSync PTP Master was indicating it was compatible with PTPv1. This was preventing the SecureSync from being able to fill in the "receivetimestamp (nanosecond)" field with just "0". Told him he would need to disable the delay request message flag that was indicating the PTP slave supports PTPv1.

Email from Denis (2/21/12)
In case they ask for more details, the flag in question is the "Hardware Compatability" bit in the "transportSpecific" field, as documented in Annex D.4 / Table 123 in the IEEE 1588 specification. We are expecting all bits in the transportSpecific field to be 0.

PTP Traffic not getting through a switch/router

Email sent to Bluefirecapital on 11/23/11:
I may be able to help explain why you may not be seeing any PTP traffic on the output side of the switch. There are a couple of reasons that this could happen.

To begin and just to confirm, the Ethernet cable to the switch is connected directly to the PTP Option Card on the SecureSync (and not to the base Ethernet port (Eth 0) ot the Gigabit Option Card- if installed). Also, when using a switch between the Master and Slave, the PTP Option Card needs to be configured with the applicable network settings in order to be on the same subnet as the switch.

If you haven't already, I recommend you first try to capture the network traffic directly from the PTP Master's side of the switch. This will confirm the PTP Master is outputting PTP traffic, as expected. If there is no traffic from the Option Card, there may be a problem with either the PTP Option Card or with its connection to the SecureSync.

The PTP outputs are multicast messages. The specs recommend multicast traffic remain on the immediate subnet only, to prevent potentially flooding the entire network with PTP packets. For this reason, the factory default TTL (Time To Live) value is set to "1". Depending on the internal operation of the switch, this will often prevent the switch from passing any PTP traffic. You may need to increase the TTL value of the PTP Master to a higher number than 1 (up to a max of 255), to allow the switch to pass the multicast traffic to other subnets.

Another potential reason for not having PTP traffic on the output side of the switch is if there is a firewall installed on the switch. The PTP traffic need ports 319 and 320 to be continuously left open, in order for the traffic to pass.

We also recommend verifying that the SecureSync has been indeed re-configured, since it was initially received, to be a PTP Master. By factory default, the PTP Option cards are configured to be a PTP Slave, unless they are reconfigured by a user to be a PTP Master.

To verify that the Option Card is re-configured to be a PTP Master, open the SecureSync's web browser and navigate to the Setup/Inputs page and select the PTP Option Card. In the Clock Setup tab, please make sure the Mode field is set to "Master Only" (not "Slave only" or "Master/Slave").

He clarified that there will be packets broadcasted, even without PTP Slaves being present, as long the PTP Option Card is an ACTIVE PTP Master and not just a passive PTP Master (this makes a difference). If the Option Card is a passive Master, no packets will be observed.

The other webpage that will confirm the port is in the Master mode and is an ACTIVE master (not just a passive Master) is in the Status/Inputs page, Network tab. Verify the Port State indicates "Master", This indicates the Option Card is an Active Master and not just a passive master. In this same table, the "Port Enable" and "Link Connected" fields should indicate "Enabled".

Below are the possible indications for the “Port State” field:

- **Initializing:** Cable is unplugged / in power-up state.
- **Listening:** PTP is looking for a Master.
- **Master:** PTP Master has become the active master on the network.
- **Passive:** PTP Master has become a passive master. (There is another Master on the network with higher priority.) This Master will wait until such time as the Best Master Clock Algorithm determines it should be the Best Master, then it will transition to the Master state.
- **Uncalibrated:** PTP Slave has detected a Master on the network.
- **Slave:** PTP Slave is actively synchronizing to a Master on the network.

“**Port Enable**” indicates the port is enabled for PTP.

“**Link Connected**” indicates whether or not the Ethernet Link is connected.

No Delay Responses being sent from the PTP Master

- Make sure there are Delay Requests (sent from the PTP Clients) reaching the PTP Master
- Request a packet capture on the same side of the switch as the Master
- If there are no Delay Requests present, the clients are sending them or they are being blocked in between) (possibly an issue with IGMP or the TTL value configured in the PTP Client)

PTP Delay Requests present but no other PTP packets present/PTP Master sending Delay Request packets

- Every PTP device (Master and Slave) sends Delay Requests, even if there aren’t any Sync packets present.

Email from Denis Reilly I was puzzled by why our Master would be sending Peer Delay Requests. Don’t Delay Requests come from the Slaves?

The answer is that Peer Delay Requests are different. Each port is supposed to exchange Peer Delay Requests to all other ports it is connected to, and the requests can go both ways. So it actually is expected a P2P Master could send Peer Delay Requests.

Issues with PTP Delay Response/Peer Delay response messages

To better assist you with this condition, I have some questions and a request for you:

To begin, in order for the SecureSync to be able to send delay responses, a properly formatted delay request needs to be received.

If you can send us a wireshark packet capture from a device on the same subnet as the SecureSync, we will be happy to evaluate the capture to ensure a valid delay request is being sent to the SecureSync. If the PTP slave is on a different subnet, please also send a capture of the PTP Slave’s traffic from its network, as well. This can help show delay requests that may not be getting to the Master.

Questions for you:

- Are the PTP slaves using One step or Two step mode?
- Is the PTP Master configured as Multicast or Unicast mode?
- What are you using for the PTP Client? Are you using Timekeeper software from Spectracom, PTPd freeware or some other program?

- If you are running PTPd freeware program, what is the version of the software?
- Are there any firewalls in between the device? If there are, make sure ports 319 and 320 are both open.
- If there are any switches in between, make sure IGMP is not set to V3, which will block the packets from getting through.
- Are the PTP clients configured to send the delay request messages on port 319?

Issues with Delay Response messages not present, even though Delay Request messages are present

1. Make sure that the PTP Delay Request messages are getting all the way to the PTP Master.
2. **Verify Ethernet port speed settings-** What is the Ethernet port speed of all devices on the network (are any switches hard-set, or are they set to auto-negotiate)? Please note that the PTP Option Card is a 10/100 Base-T device only. It is not compatible with faster 1 Gbit/s or 10Mbit/s devices, so devices should not be hard set to these values.
3. Make sure the PTP Slave is configured to send the PTP Delay Request on the correct PTP port. Example of this being an issue is below:

   "The problem was the port number used by the delay request. It was using the general udp port rather than the event udp port (319)"

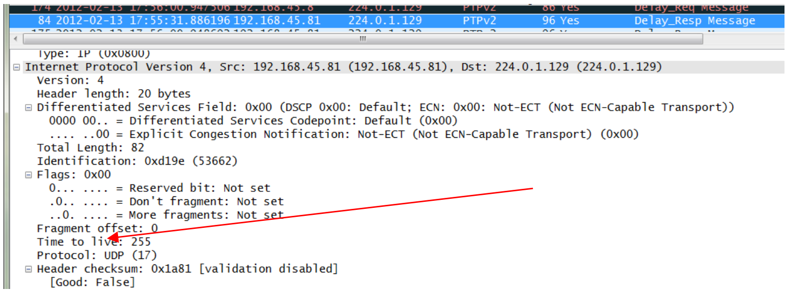
4. Verify the PTP Slave is configured/sending PTP Delay Requests

   (grab a packet capture from a system on the same subnet as the slave and verify the packets are present in the capture). If a firewall is located between the two, make sure the PTP ports are left open (ports 319 and 320).

5. Check packet captures to see if SecureSync is sending out Peer Delay requests, instead of delay responses.

   Email from Denis Reilly- In the packet traces, I could see that the Slave was sending out Delay Requests, and the Master was sending Peer Delay Requests. Delay Request/Responses are for E2E, and Peer Delay Request/responses are for P2P, and they don't mix well. Solution was to change the Slave to E2E instead of P2P.

6. If PTP Slave is sending Delay Request messages, verify they are being sent through any or all network switches.
7. Verify TTL level is configured large enough in the PTP master to allow the packets to get through ALL switches and routers in between Master and Clients.



```
 174 2012-02-13 17:56:00.947506 192.168.45.8    224.0.1.129    PTPv2    86 Yes    Delay_Req Message
  84 2012-02-13 17:55:31.886196 192.168.45.81   224.0.1.129    PTPv2    96 Yes    Delay_Resp Message

    Type: IP (0x0800)
⊟ Internet Protocol Version 4, Src: 192.168.45.81 (192.168.45.81), Dst: 224.0.1.129 (224.0.1.129)
    Version: 4
    Header length: 20 bytes
  ⊟ Differentiated Services Field: 0x00 (DSCP 0x00: Default; ECN: 0x00: Not-ECT (Not ECN-Capable Transport))
      0000 00.. = Differentiated Services Codepoint: Default (0x00)
      .... ..00 = Explicit Congestion Notification: Not-ECT (Not ECN-Capable Transport) (0x00)
    Total Length: 82
    Identification: 0xd19e (53662)
  ⊟ Flags: 0x00
      0... .... = Reserved bit: Not set
      .0.. .... = Don't fragment: Not set
      ..0. .... = More fragments: Not set
    Fragment offset: 0
    Time to live: 255
    Protocol: UDP (17)
  ⊟ Header checksum: 0x1a81 [validation disabled]
      [Good: False]
```

Issues with IGMPv3 (Internet Group Management Protocol) on network switches (IGMP v2 is OK, IGMP V3 is not)

Per Jeremy Onyan- It's not that we don't support IGMPv3. It's a matter of configuring the switch to not ignore and drop the multicast packets which will happen if IGMPv3 isn't configured to route them.

IGMPv2 improves over IGMPv1 by adding the ability for a host to signal desire to leave a multicast group. IGMPv3 improves over IGMPv2 mainly by supporting source-specific multicast (SSM)

Source-specific multicast (**SSM**) is a method of delivering multicast packets in which the only packets that are delivered to a receiver are those originating from a specific source address requested by the receiver. By so limiting the source, SSM reduces demands on the network and improves security.

So in other words, IGMPv3 is filtering out the packets because it's not been configured to allow the PTP multicast packets to go through. They can use IGMPv3, and we have customers that do, they just need to configure it to allow multicast (on port 224 I think…).

Note: Refer to Mantis case 1776 for more info

- IGMP is a network switch configuration that may be present (it's not in the Spectracom devices).
  - IGMP v3 is blocking Delay Request messages.
  - Change IGMP in the network switches to IGMP V2.

(**Note:** reported by Stewart Smith with Virtu Financial)
Apparently, they were having trouble with the PTP slaves having high jitter and offset values. Delay response messages apparently stopped getting passed though a network switch, after a short period of time

Well — I'm finding out that it responds with Delay_Response packets for about 5 minutes. Then they all go unanswered again.

Update status email from Stewart (9/29/12) about this issue
I brainstormed the issue with our network engineer Rex. He changed our switch configuration from IGMPv3 to IGMPv2 and it immediately started working. In other words, with no other changes the clients immediately started seeing Delay_Response messages from the Spectracom SecureSync PTP option card.

I believe the timeout I referred to in my previous message was probably imposed by the switch.
I think some additional work could be put into the implementation of IGMP on the PTP option cards — but there is an obvious work-around which we are using now. Just use IGMP v2.

From Mantis case 1776 (note added by Denis Reilly): We have a customer who has reported successfully using SecureSync PTP with a router that was configured with IGMPv3. When asked how they did it, they replied that they forced Multicast Joining for 224.0.1.129 on the port of the switch connected to the SecureSync.

Perhaps this is an IGMP configuration issue after all?

DD) PTP_TWO_STEP flag and "transportspecific" field settings in the Delay Request

Email from Denis (2/22/13) -Please advise Todd that he needs to set the **PTP_TWO_STEP flag** to 0 and the "**transportspecific**" field to 0x00 in his delay requests. This is why he is seeing problems with the return timestamps in the Delay Responses.

Our hardware timestamper is very particular and will not timestamp delay request packets with these issues. But our PTP stack will send the Delay Response anyway, sometimes with a stale timestamp.

EE) (Model 1204-12 10/100 card only) make sure the PTP slave is NOT configured to use Telecom or Power Profile.

- PTP Slaves that are configured to use other than the default profile (Such as Telecom or Power profiles) send multiple TLVs. This is an issue as the 1204-12 card only supports one TLV.

Email from Denis Reilly, 13 Mar 2014) I think Exfo is the client that tries to connect with Multiple TLV's in one message. That's explicitly called out as a possibility in the Telecom profile, but not in the main spec.
The Actarus 1204-12 card does not support this, and the Korusys 1204-32 card does.
As far as I know, there are no plans to put that function into Actarus, because Actarus is not sold as being Telecom-profile compatible. I've copied Laurent and Michel in to confirm.

FF) Timestamp is missing in the Delay Response message

Email from John Fleck
I have a Secure Sync device with GPS and a PTP option module card.
When the delay mechanism is set to E2E, it seems the timestamp in the delay response message is invalid.
The field is called the receive timestamp. The seconds of the field seem stuck, at an early time, and the nanoseconds are always zero. See his example below (bold value towards the bottom indicates the issue).

Transmission of delay response

```
Frame 9 (96 bytes on wire, 96 bytes captured)
Ethernet II, Src: Spectrac_08:04:86 (00:0c:ec:08:04:86), Dst: IPv4mcast_00:01:81 (01:00:5e:00:01:81)
Internet Protocol, Src: 192.168.1.221 (192.168.1.221), Dst: 224.0.1.129 (224.0.1.129)
User Datagram Protocol, Src Port: ptp-general (320), Dst Port: ptp-general (320)
Precision Time Protocol (IEEE1588)
   0000 .... = transportSpecific: 0x00
   .... 1001 = messageId: Delay_Resp Message (0x09)
   .... 0010 = versionPTP: 2
   messageLength: 54
   subdomainNumber: 0
   flags: 0x0000
      0... .... .... .... = PTP_SECURITY: False
      .0.. .... .... .... = PTP profile Specific 2: False
      ..0. .... .... .... = PTP profile Specific 1: False
      .... .0.. .... .... = PTP_UNICAST: False
      .... ..0. .... .... = PTP_TWO_STEP: False
      .... ...0 .... .... = PTP_ALTERNATE_MASTER: False
      .... .... ..0. .... = FREQUENCY_TRACEABLE: False
      .... .... ...0 .... = TIME_TRACEABLE: False
      .... .... .... 0... = PTP_TIMESCALE: False
      .... .... .... .0.. = PTP_UTC_REASONABLE: False
      .... .... .... ..0. = PTP_LI_59: False
      .... .... .... ...0 = PTP_LI_61: False
   correction: 0.000000 nanoseconds
   ClockIdentity: 0x000cecfffe080486
   SourcePortID: 1
   sequenceId: 8
   control: Delay_Resp Message (3)
   logMessagePeriod: 4
   receiveTimestamp (seconds): 1357564928
   receiveTimestamp (nanoseconds): 0
   requestingSourcePortIdentity: 0x000da8fffe23d768
   requestingSourcePortId: 1
```

To troubleshoot this condition:

- Have the customer send a packet capture of the Delay Request and delay Response messages. Make sure that all values in the PTP Delay Request (sent from the PTP client) have all of the correct values. All other values NEED to be False (0). If the Delay Request is not completely per the specs, the Delay Response is still sent to the Client, but the timestamp data is not included in the reply.

The IEEE Standard 1588-2008 defines each of the fields for the Delay Request packet (as shown below):

- The API calls associated with PTP Input and output (Master and Slave modes) are the “PTR” calls. PTR calls control and process decoded PTP network packets (either as an input reference or a time output).

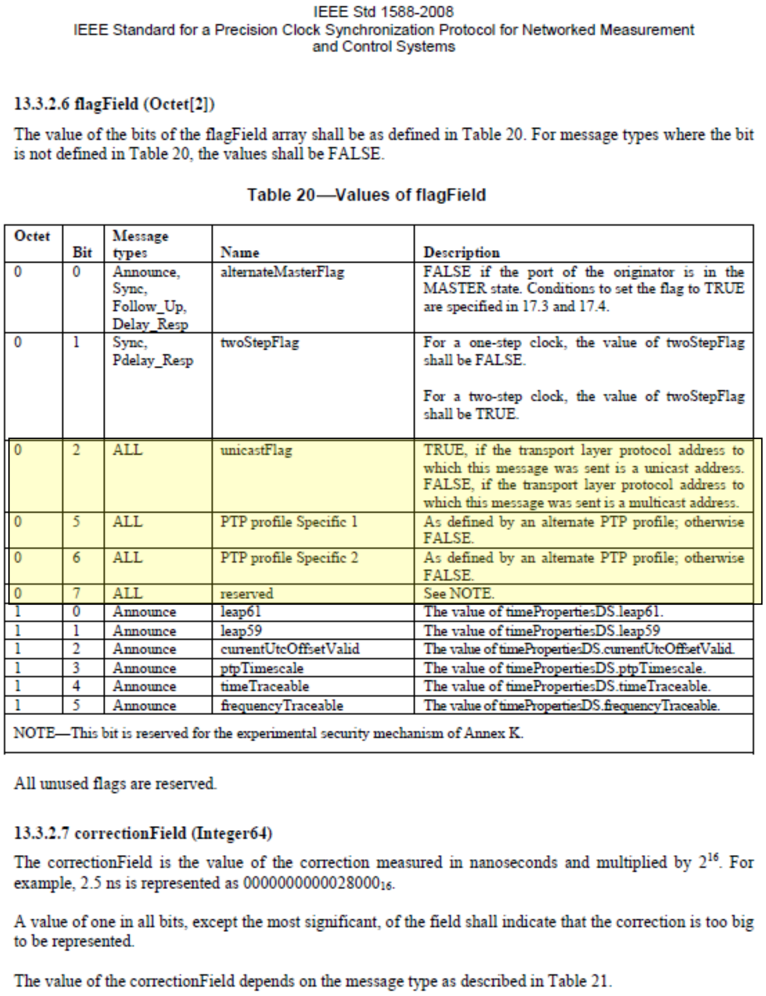
With specific example above, reply from Denis Reilly: Our timestamper is likely not timestamping the packets because the **PTP_TIMESCALE flag** in the Delay Request is enabled.

After mentioning this to Denis, he replied to me with: Keith, This makes sense. Packets sent to the general udp port are not timestamped, packets sent to the event port are. I need to remember this in the future.

- Make sure that the PTP Delay Request messages are getting all the way to the PTP Master. If a firewall is located between the two, make sure the PTP ports are open.
- Make sure the PTP Slave is configured to send the PTP Delay Request on the correct PTP port. Example of this being an issue is below (Reply from the customer, regarding the root cause of the issue above)

The problem was the port number used by the delay request. It was using the general udp port rather than the event udp port

.



IEEE Std 1588-2008
IEEE Standard for a Precision Clock Synchronization Protocol for Networked Measurement and Control Systems

13.3.2.6 flagField (Octet[2])

The value of the bits of the flagField array shall be as defined in Table 20. For message types where the bit is not defined in Table 20, the values shall be FALSE.

Table 20—Values of flagField

Octet	Bit	Message types	Name	Description
0	0	Announce, Sync, Follow_Up, Delay_Resp	alternateMasterFlag	FALSE if the port of the originator is in the MASTER state. Conditions to set the flag to TRUE are specified in 17.3 and 17.4.
0	1	Sync, Pdelay_Resp	twoStepFlag	For a one-step clock, the value of twoStepFlag shall be FALSE. For a two-step clock, the value of twoStepFlag shall be TRUE.
0	2	ALL	unicastFlag	TRUE, if the transport layer protocol address to which this message was sent is a unicast address. FALSE, if the transport layer protocol address to which this message was sent is a multicast address.
0	5	ALL	PTP profile Specific 1	As defined by an alternate PTP profile; otherwise FALSE.
0	6	ALL	PTP profile Specific 2	As defined by an alternate PTP profile; otherwise FALSE.
0	7	ALL	reserved	See NOTE.
1	0	Announce	leap61	The value of timePropertiesDS.leap61.
1	1	Announce	leap59	The value of timePropertiesDS.leap59
1	2	Announce	currentUtcOffsetValid	The value of timePropertiesDS.currentUtcOffsetValid.
1	3	Announce	ptpTimescale	The value of timePropertiesDS.ptpTimescale.
1	4	Announce	timeTraceable	The value of timePropertiesDS.timeTraceable.
1	5	Announce	frequencyTraceable	The value of timePropertiesDS.frequencyTraceable.

NOTE—This bit is reserved for the experimental security mechanism of Annex K.

All unused flags are reserved.

13.3.2.7 correctionField (Integer64)

The correctionField is the value of the correction measured in nanoseconds and multiplied by 2^{16}. For example, 2.5 ns is represented as 0000000000028000_{16}.

A value of one in all bits, except the most significant, of the field shall indicate that the correction is too big to be represented.

The value of the correctionField depends on the message type as described in Table 21.

Bits 2, 5, 6, and 7 (highlighted) are specific to the Delay Request to add time to the Delay Response.

In the example issue above, the issue was due to the “PTP_TimeScale” field indicating “true” (should be false) as shown as bolded in the delay request message below).

Transmission of delay request

```
Frame 7 (86 bytes on wire, 86 bytes captured)
Ethernet II, Src: Teletron_23:d7:68 (00:0d:a8:23:d7:68), Dst: IPv4mcast_00:01:81 (01:00:5e:00:01:81)
Internet Protocol, Src: 192.168.1.5 (192.168.1.5), Dst: 224.0.1.129 (224.0.1.129)
User Datagram Protocol, Src Port: ptp-general (320), Dst Port: ptp-general (320)
Precision Time Protocol (IEEE1588)
    0000 .... = transportSpecific: 0x00
    .... 0001 = messageId: Delay_Req Message (0x01)
    .... 0010 = versionPTP: 2
    messageLength: 44
    subdomainNumber: 0
    flags: 0x0008
        0... .... .... .... = PTP_SECURITY: False
        .0.. .... .... .... = PTP profile Specific 2: False
        ..0. .... .... .... = PTP profile Specific 1: False
        .... .0.. .... .... = PTP_UNICAST: False
        .... ..0. .... .... = PTP_TWO_STEP: False
        .... ...0 .... .... = PTP_ALTERNATE_MASTER: False
        .... .... ..0. .... = FREQUENCY_TRACEABLE: False
        .... .... ...0 .... = TIME_TRACEABLE: False
         .... .... .... 1... = PTP_TIMESCALE: True
        .... .... .... .0.. = PTP_UTC_REASONABLE: False
        .... .... .... ..0. = PTP_LI_59: False
        .... .... .... ...0 = PTP_LI_61: False
    correction: 0.000000 nanoseconds
    ClockIdentity: 0x000da8fffe23d768
    SourcePortID: 1
    sequenceId: 8
    control: Delay_Req Message (1)
    logMessagePeriod: 0
    originTimestamp (seconds): 1357564981
    originTimestamp (nanoseconds): 322428487
```

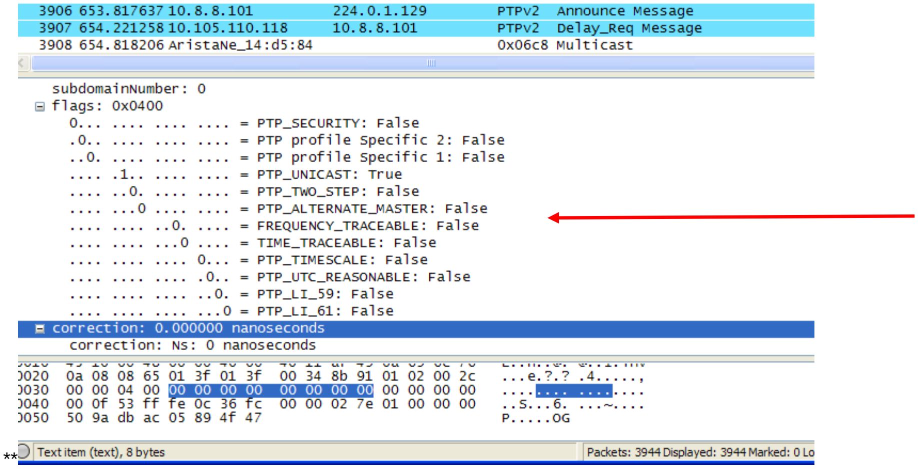
GG) Issue with PTPd freeware client software

(8 Nov 2012 KW per discussion with Denis Reilly), Denis said there is an issue with PTPd, where they are doing something incorrect (though technically, our PTP Masters should be able to tolerate) which prevents delay responses from being able to be processed. This results in Delay responses not being able to be processed by the PTP clients.

- Make sure PTP Client is operating in Multicast mode, if the PTP Master is in in Multicast mode (and vice-versa).

Results: a wireshark packet capture showed delays requests were present but it showed the PTP client was in Unicast mode, while the SecureSync PTP card only supports Master mode. The destination address of 10.8.8.101 indicated it was being sent via Unicast mode. A multicast packet is always sent to the address of 224.0.1.129. So Denis could tell that it was sending it in Unicast mode, causing the PTP Master to completely ignore the delay requests.

Example wireshark screenshot of the delay request packet because of this configuration issue (Notice the flags are false).



```
3906 653.817637 10.8.8.101          224.0.1.129         PTPv2   Announce Message
3907 654.221258 10.105.110.118      10.8.8.101          PTPv2   Delay_Req Message
3908 654.818206 AristaNe_14:d5:84                       0x06c8 Multicast

  subdomainNumber: 0
⊟ flags: 0x0400
    0... .... .... .... = PTP_SECURITY: False
    .0.. .... .... .... = PTP profile Specific 2: False
    ..0. .... .... .... = PTP profile Specific 1: False
    .... .1.. .... .... = PTP_UNICAST: True
    .... ..0. .... .... = PTP_TWO_STEP: False
    .... ...0 .... .... = PTP_ALTERNATE_MASTER: False
    .... .... ..0. .... = FREQUENCY_TRACEABLE: False
    .... .... ...0 .... = TIME_TRACEABLE: False
    .... .... .... 0... = PTP_TIMESCALE: False
    .... .... .... .0.. = PTP_UTC_REASONABLE: False
    .... .... .... ..0. = PTP_LI_59: False
    .... .... .... ...0 = PTP_LI_61: False
⊟ correction: 0.000000 nanoseconds
    correction: Ns: 0 nanoseconds

0020  0a 08 08 65 01 3f 01 3f  00 34 8b 91 01 02 00 2c   ...e.?.? .4.....,
0030  00 00 04 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0040  00 0f 53 ff fe 0c 36 fc  00 00 02 7e 01 00 00 00   ..S...6. ...~....
0050  50 9a db ac 05 89 4f 47                            P.....OG

Text item (text), 8 bytes                Packets: 3944 Displayed: 3944 Marked: 0 Lo
```

**

PTP module (Port State field) is stuck in "initializing"

Q. When I look at the output status the port is "initializing"…

Does the reported version information for the 1204-12 card look correct? If it looks valid, the Option Card is likely "awake".

Email from Denis Reilly about a card stuck in "Initializing"
Everything looks like it upgraded properly. It's on 4.8.6, and the PTP firmware is on B105. In particular, the face that the Version information is present means that the card is awake!

He should look at the "Network" PTP Status page and confirm that the "Link Connected" field says "Enabled". "Disabled" means that the PTP card does not detect a physical network connection.

I assume he has tried rebooting the unit. He may also try unplugging and re-plugging the Ethernet from the back of the unit, and cold restarting the card.
I can see two possible scenarios:

- The cable is not plugged in all the way!
- A longshot, but somehow, the KTS is confused about the state of the PTP module. So the module may be operating properly, but the KTS component does not realize it. We actually built in a feature where when the Ethernet cable is plugged in, the KTS detects this and will re-query the state of the PTP card. Of course, there needs to be a certain amount of communication present for this to work at all, but since he has version information it's worth a shot.

PTP Option Card reverted from PTP Master to Slave Mode (or vice-versa)

Examples

- PTP Option Card reverted to PTP slave mode from PTP Master mode after update was performed
- After a power cycle, PTP Card switches from PTP Master to PTP slave.

The following statement was from a customer: Just before performing a software update from 4.7.0 to 4.8.7, *as* directed by the update instructions, the PTP card was reconfigured from PTP Master to PTP Slave mode. At the end of the update, the PTP card was automatically back in PTP Master mode.

The PTP card returned to a "Master Only" mode after the upgrade - was that supposed to happen? I thought I'd have to change it back from slave mode myself.
Keith's explanation (10/11/12): Not technically, but I can now easily see how this could occur (nothing was adversely affected by this happening). In all earlier versions of the PTP Option Card firmware, whenever a configuration change was made by a user to the PTP card, the settings had to then be manually "Saved to ROM" as a second step, if you wished for the settings to persist through power cycles of the SecureSync. If changes were made without the "Saved to ROM" step being performed, the changes that were just made did not persist through any reboots. The changed values that weren't saved to ROM reverted back to the values that were configured just before the more recent changes were made by the user.

Specific to this instance, if you didn't perform a "Save Settings to ROM" before the update process was started, it was in Slave Mode initially, as you had configured it, but this setting reverted back to the Master Mode during the internal reboots that occur during the update process, because the change wasn't Saved to ROM first.

"Save Settings to ROM" only applied to the PTP Option Card setting. The Version 4.8.7 software update now alleviates the need to save the PTP configuration changes to ROM. This now happens automatically, when changes are made to the PTP card.

**Excessive PTP/1PPS jitter from a Tsync-PTP Master (as exhibited on the PTP Slave)

One way to test PTP accuracy is to compare a 1PPS output from the PTP Master and one from the PTP Slave to see how closely they are aligned to each other. Due to certain conditions, there may be fairly high jitter/wander between them.

This is because the 1PPS output from the PTP slave is not tied directly to the PTP input time stamps. The better the oscillator in the PTP Master and PTP Slave, the lower the 1PPS jitter will likely be (and vice versa) A TCXO oscillator in a PTP Slave can potentially see like 50-70ns of jitter, as a factor of the oscillator- not a factor of the time stamps. And TCXO oscillator can be greatly affected by even small, ambient temperature changes.

- 1PPS input on the TSync?
- Is an external 1PPS input reference applied to the TSync?
- If a PTP Slave that is synced to a PTP Master is exhibiting higher than expected jitter (besides factors such as Delay mechanism) if the SecureSync is ever using NTP/External PPS input as the selected references, NTP Expert Mode must be used to apply the external PPS input to the SecureSync. Otherwise, the SecureSync's output (such as PTP outputs) will have more jitter than expected. The external PPS can only be utilized by manually editing the ntp.conf file in Expert Mode. Refer to the Example Use Cases in the SecureSync NTP Peering document for proper configuration of the NTP Servers page of the browser.

- How good is the oscillator on the PTP Slave

Q. What type of oscillator is installed in the PTP Slave?
A. The better the oscillator type, the less 1PPS jitter that will be present.
Regarding the amount of jitter your customer is observing/reporting this is likely an expected amount of jitter, as a factor of the internal operation of the PTP Slave. if the PTP Slave has a low-end oscillator installed, such as a TCXO oscillator, this is an expected amount of jitter. The PTP input is likely VERY good, but the 1PPS output is not directly tied to the accuracy of the PTP packets. We have observed this with our own testing, when using a PTP Slave with a TCXO oscillator (we have seen similar measurements as he is observing). The 1PPS jitter of a PTP slave could be minimized by using a PTP slave with a more stable, high-end oscillator installed such as an ovenized OCXO or Rubidium oscillator.

Especially with TCXO oscillators, which aren't ovenized, the stability of the oscillator can be significantly affected by just the ambient temperature. Even slight changes to the temperature changes how these oscillators operate.

Regarding the specific question:

The idea is if the TSync can tell us what the PTP output jitter is (let say 30ns), than, when we measure the slave device and see 91ns, we can say the slave generate 61ns jitter. (Does it work this way?)

The engineer mentioned this is a likely a good estimation, as the PTP Slave is the device that handles the majority of the math needed for the ontime point in the PTP Slave to be established (not the PTP Master). The PTP Master is primarily just providing an ontime point for the PTP Slave to reference. Then it's up to the PTP Slave to perform the math/calculations necessary to align its ontime point. This means that like 70% of this process is up to the Slave, as was mentioned.

***** Get Packet Captures if at all possible!*****

**VRRP (Virtual Router Redundancy Protocol) packets included in captures

- **Email from Michel (22 Sept 14)** VRRP packets have no impact because they are to a specific multicast address to which the master doesn't join;

**Symantec Endpoint Protection (SEP)

Q. On the PTP case (multicasting problem):

- I have WireShark proof (from one system, a server, without Symantec Endpoint Protection, a. k. a. SEP) that there is PTP traffic being broadcast, every two seconds, from 00:0c:ec:08:09:22 (the MAC address of the SecureSync's PTP option card).
- Unfortunately, the PTP client (a PC) has SEP and not even WireShark on that system can see the PTP traffic, so I assume this is why the PTP card in it cannot synchronize with the PTP master.

I have tried several things with changing SEP settings and adding "allow" rules to the Windows Firewall (which is controlled by SEP, when SEP is installed), but have had no luck.

Do you have any suggestions? Ports to allow, Windows Firewall adjustment, SEP adjustments, etc.? (SEP is NASA-mandatory on PCs.)

A Reply from Denis Reilly (7 Aug 2014) How to troubleshoot blocked network traffic with the Symantec Endpoint Protection firewall

- http://www.symantec.com/business/support/index?page=content&id=TECH203497

**Cisco slaves not syncing

Email from Denis (18 Jul 2014) What is the **Sync** rate on the PTP Master? Cisco gear may not sync properly if the Sync rate is less than **4 per sec.**

**PTP Master not sending Delay Response Messages to Slaves

3) Check **transportspecific** field in **Delay Request** message to ensure its set to 0x00.
4) Verify the PTP card is actually receiving the Delay Request (it may be being blocked)
5) See if the switch(es) that the card is connected to is using IGMP. If it is, verify the PTP Master has joined the multicast group properly.

Note: If using IGMPv3, special configuration is necessary
Per Jeremy Onyan- it's not that we don't work with IGMPv3, it's that v3 has multi cast packet filtering that needs to be configured to allow the packets. So they can use it if they know how to set that up.
Note: Ping is unicast, so a successful ping doesn't verify IGMP group join.

**PTP Master sending more than one delay response per received delay request/ “UP” packets present on ports 319/320

- There should be only one delay response per delay request
- Pico trading has two timekeeper slaves that may be affecting delay responses. Blocking the two slaves from the network to see if it stops sending two responses per request. Refer to Salesforce cases 15800/15817

Email from Michel (22 Sept 14) The issue is that sometimes extra Delay_Resp packets are sent by the master through the UDP port 319. The 2 points are that for one Delay_req packet received by the master, a right Delay_resp packet is sent through the UDP port 320, but sometimes 2 extra Delay_resp packets or more are sent through the UDP port 319 (whereas a Delay_resp shall be sent through the UDP port 320).

I don’t know yet why, but I guess that it is related to the order in which Delay_req and Delay_resp packets are sent and received. I mean that sometimes 2 Delay_req packets are received by the master before it sends the Delay_resp packet of the first received Delay_req packet; It’s just an hypothesis at this time;
The “UDP” packets could be another manifestation of the same issue.

Suggestion: to know finally whether the wrong Delay_resp packets sent by the master are du, or not, to the wrong Delay Req packets sent by the two slaves sending these wrong packets (10.105.110.117 and 10.105.110.119), we could suggest to the customer to remove them from the network and then, to capture new wireshark traces during few minutes.

Applicable to all TPRO/TSAT Series legacy Timing boards

Discontinuance/Availability of LEGACY TPRO/TSAT series timing boards

A) Updated info, as of Dec 2020 (per presentation from Emmanuel)

we will obsolete the TSAT/TPRO as well as some non PCIe variants, offering a last time buy opportunity to customers, starting in Oct. 2020, with deliveries ending Sept 2021. TSYNC-PCIe and VPX will be the only offered variants afterwards.

Tech support/repairs for TSAT/TPRO Refer to 295291 (3 Apr 2023)

B) Previous to Dec 2020

- Per Dave Sohn (as of at least Sept 2019), with the exception of TPRO/TSAT-PCI, all other variants of the legacy TPRO/TSAT Boards (such as PMC and cPCI) are discontinued/no longer available.
- Customers should consider replacing a legacy TPRO/TSAT board with a similar variant of the **TSync** board (such as replacing a **TPRO/TSAT-cPCI** board with a **TSync-cPCI** board)

Slightly modified email from Dave Sohn (25 Sept 2019) Only the **TPRO/TSAT-PCI** boards have **not been discontinued**. TPRO/TSAT-cPCI/PMC bords are no longer available. I suggest looking at the TSync-PMC/TSync-cPCI boards instead.

Replacing a legacy TPRO/TSAT-PMC/cPCI board with a TSync-PMC/TSync-cPCI board

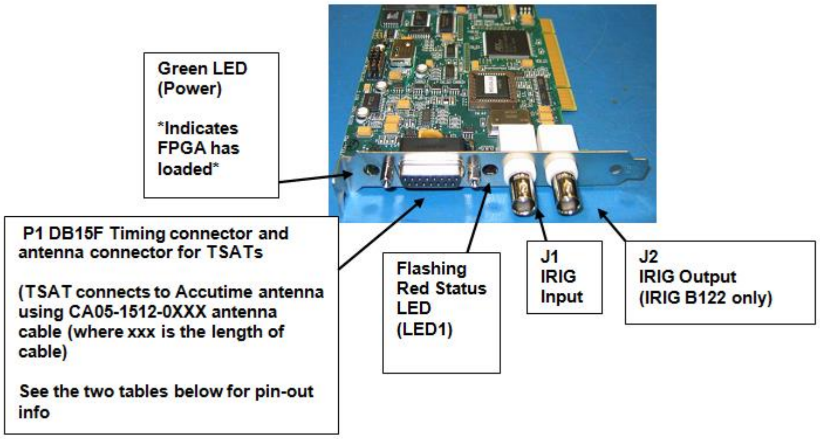
Slightly modified email Keith sent to Nidal (25 Sept 2019)
Having your customer consider "upgrading" from TPRO-cPCI to TSync-cPCI (or TPRO-PMC to TPRO-PMC) is a great suggestion, which I didn't think of. The TSync-cPCI (and TSync-PMC) board is not "exactly" a "complete plug-n-play" swap for TPRO-cPCI (and TSync-PMC), but it's pretty close to being a "direct replacement" for the older timing board.

The TSync-PMC/cPCI will work in the same slot as the TPRO-PMC/cPCI board was installed.

The customer will still need to install the associated TSync driver (available at no cost from our website). The TSync drivers contain all the legacy TPRO/TSAT API calls/example programs. The TPRO.lib and TPRO.h files are compatible with the newly installed TSync board, as long as there is no need/desire to take advantage of the newer API calls (such as holdover, for instance), customer application software doesn't need to be changed. But, the application software needs to be re-compiled with the TSync driver before it will work with the TSync-PMC board.

Note: Customer must install the TSync driver whenever a TSync-xxx board is installed. The previously installed TPRO/TSAT driver won't be able to communicate with the installed TSync-cPCI (or TSync-PMC) board (the device ID is different). So, the TSync driver also needs to be installed (the TPRO/TSAT driver can either be uninstalled or installed simultaneously with the TSync driver, as desired).

TPRO/TSAT-PCI, PCI-U, PCI-66U and PCI-U-2


Green LED (Power)
Indicates FPGA has loaded
P1 DB15F Timing connector and antenna connector for TSATs
(TSAT connects to Accutime antenna using CA05-1512-0XXX antenna cable (where xxx is the length of cable)
See the two tables below for pin-out info
Flashing Red Status LED (LED1)
J1 IRIG Input
J2 IRIG Output (IRIG B122 only)

Links/shortcuts

Shortcut to data sheet: I:\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards

Shortcut to Customer Service folder EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT

Manuals/driver guides

Shortcut to TPRO/TSAT-PCI (5v) user manual: I:\Engineering\Archive\Released\1141 - TPRO-TSAT PCI 5V\Manual

Shortcut to TPRO/TSAT-PCI-U-2 series user manual (1186-5001-0050) in Arena:

https://app.bom.com/items/detail-spec?item_id=1202833262&version_id=10212776478&orb_msg_Single_Search_p=1

Shortcut to Linux Driver Guide (1186-5003-0050) in Arena: https://app.bom.com/items/detail-spec?item_id=1202833264&version_id=10212776498

Shortcut to Windows Driver Guide (1186-5002-0050) in Arena: https://app.bom.com/items/detail-spec?item_id=1202833263&version_id=10212776488&orb_msg_Single_Search_p=1&redirect_Seqno=7546148528

Schematics/BOMs/Engineering docs

Shortcut to PCI-U schematics: (1159-0002-2000) I:\Engineering\Archive\New Released\PCB Documentation

Shortcut to PCI-66U/U-2 schematics (1186-1001-0200, in 1186-0000-F000) in Arena: https://app.bom.com/items/detail-spec

Shortcut to PCI-66U/U-2 BOM: (1186-0000-F000) in Arena: https://app.bom.com/items/detail-whereused?item_id=1202838971&version_id=10338419768

Link to design specs/test results: I:\Company Wide\Project Folders\JumpStart(PCIU-66)

PCI series drivers:

- **Window driver (1186-SD07-xxxx)** where xxxx is the version of the driver (search PLM for “1186-SD07”)
- **Linux driver (1186-SD09-xxxx)** where xxxx is the version of the driver (search PLM for “1186-SD09”)

Hardware

Primary variants of TPRO/TSAT-PCI series boards

TPRO Model	Part Number
TPRO-PCI	1141-0002-0600
TPRO-PCI-U	1159-0002-0600
TPRO-PCI-U-2	1186-3002-0600
TPRO-PCI-66U	1186-6002-0600

TPRO Model	Part Number
TSAT-PCI	1141-0001-0600
TSAT-PCI-U	1159-0002-0600
TSAT-PCI-U-2	1186-3001-0600
TSAT-PCI-66U	1186-6001-0600

Distinguishing between TPRO-PCI and TSAT-PCI (original 5v cards)

- Go by either Part Number or sub-system ID

*Certifications (CE, FCC, etc)

CE Declaration of conformity

The newer TPRO/TSAT-PCI-U-2 and TPRO/TSAT-PCI-66U cards are CE approved (The earlier TPRO/TSAT-PCI and TPRO/TSAT-PCI-U boards are not CE approved

PVC/plastic components, / flame retardant standard

Q. (from Rusbel Perez with Invensys- 12/18/12)
I'm looking for the datasheets on the for Time Reader/Generator Model TPRO-PCI-U2. Specifically, I'm looking for if your product contains any PVC and/or other plastic components, and what is the flame retardant standard of your product.

A (Reply from Scott Holmes) The BNC and D-Dub connectors are made from PVC material. The shunts, pots, many of the IC components and other connectors on the board may be made with PVC material or other plastics.

The bare PC board is rated UL-94V0 but the completed product has not been tested for any flame rating.

*On-board battery

TPRO/TSAT-PCI-U-2 and TPRO/TSAT-PCI-66U timing boards do not have any batteries onboard.

On-board Memory

There is no non-volatile memory on the timing board (it contains only volatile memory, so no customer-unique data is stored upon loss of power). This is why the year value needs to be set each time it powers up (unless using GPS input to a TSAT board, because it can obtain it from the GPS signal).

Refer to: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PCI and PC104 VME

*Conversion of a board from PCI-66U to PCI-U-2

When replacing a legacy PCI board with a PCI-66U board, the original PCI driver has to be uninstalled /new version of the driver installed and their application software will need to be recompiled to the newly installed driver. If they desire not to update the driver and don't need the faster time reads provided by the PCI-66U), they should use the PCI-U-2 boards, instead.

The only difference between the two board variants is the FPGA software. If it's desired to **reprogram** the PCI-66U board as a PCI-U-2 board, new software has to be pushed into the board using a lattice programmer and special programming cable.

It is unlikely that Spectracom UK or France will have the ability to reprogram the boards, so they need to be returned to the US for re-programming.

Email to a customer (11/23/11 KW)
The TPRO-PCI-66U and TPRO-PCI-U-2 are both currently available products and both replace the earlier TPRO-PCI and TPRO-PCI-U timing boards.
The reason that both these timing boards are concurrently available has to do with the timing board driver. The difference between these two Models is the TPRO-PCI-66U boards specifically allows for faster time reads than all of the other timing boards in the PCI family. We had certain customers that wanted to be able to perform more time reads in the same amount of time. So, the PCI-66U

version of the timing boards are able to perform twice as many time reads in the same amount of time.

However, because of the differences in this operation from the earlier series, the driver needs to be updated when replacing the earlier series boards with a TPRO-PCI-66U boards. Because the faster time reads is not normally required, the TPRO-PCI-U-2 boards don't have the faster time reads and do not require a new version of the driver to be installed. Other than the faster time reads, both boards are identical!

So, if you don't need to perform twice as many time reads (this is rare), the TPRO-PCI-U-2 is the recommended replacement timing board to alleviate the requirements of having to install a new driver.

*Replacing a TPRO/TSAT-PCI (original 5vdc board) or TPRO/TSAT-PCI-U board with a TPRO/TSAT-PCI-U-2 board (compatibility between PCI/PCI-U and PCI-U-2)

When updating either a TPRO/TSAT-PCI or TPRO/TSAT-PCI-U timing board to a TPRO-PCI-U-2 timing board, the new board was intentionally designed to be a drop-in compatible replacement. The driver from the TPRO-PCI-U board does not need to be updated to be compatible with the newer board. The driver just needs to be updated when installing a TPRO-PCI-66U board (instead of the PCI-U-2 board) or when switching to a 64 bit OS. Otherwise, the driver does not need to be updated.

However, since the driver was updated, there was a change to the newer driver. The new driver uses all of the same they just simply need to recompile their software to the new driver. They do not need to recompile the driver- just their application software. If they haven't performed this step yet, it explains the symptoms they are seeing.

Once they recompile their application software to the new driver, they should be all set!!

With install of a new PCI-U-2 board in a Windows PC, Windows will report new hardware has been detected and will ask for the new driver to be installed. Just let it automatically search for the driver and it should automatically find it.

Schneider electric (used to be “Invensys”) Support

Note: Invensys wants to directly support their customers, whenever possible. Invensys employees or customers of Invensys should contact Ed Pittsley with Invensys for help. His contact info is directly below. They typically use PCI series boards.

Contact
Ed Pittsley
(508) 549-3068
I believe his email address may now be ed.pittsley@schneider-electric.com

Schneider electric / Invensys packages

- Includes TSAT series timing board, sometimes an Acutime series GPS antenna and other accessories

K0204A (*K0204AX, K0204AY, K0204AZ)* kits

Summary (details further below)

- Created ~8 Aug 2018
- created to change Acutime-GG antennas with Acutime 360 antenna
- ~~**1186-7000-0650** (**Rev 1)** is the most recent kit (since 4/25/2015) with a TSAT-PCI-U-2 board (no GPS antenna)~~
- ~~**1159-7000-0650** (**Rev A**) is the same original rev, but with TSAT-PCI-U (no antenna)~~
- ~~**1159-7000-0650 (Rev A):** earliest package with TSAT-PCI-U, antenna (?)~~

Regarding the K0204AX, K0204AY, K0204AZ BoM’s, there are essentially two product changes required for Schneider’s application:

1) As with the previous kits, for compatibility with the Fiber RX and TX kits (that utilize serial data communication), the eprom in TSync-PCIe-002 needs to be re-programed from the default TSIP (allowing bi-directional communication) to NMEA (which is a one-way communication). This modification will create a new part number with a “1191-“ prefix.

Email from Sadie (8 Aug 18) There has been confusion on our side on what we needed to support and thus what external connection cables that we needed. I think we’ve finally resolved that. I believe we need the following external cables (from link https://spectracom.com/sites/default/files/document-files/TSync_External%20Connections_Guide_revG.pdf)

CA08R-DMD6-0001 Rev B	Micro DB25 to high density DB26
CA08R-0000-0006 Rev B	I/O Breakout cable
CA05R-MD15-0001	Mini DIN to DB15 Adapter

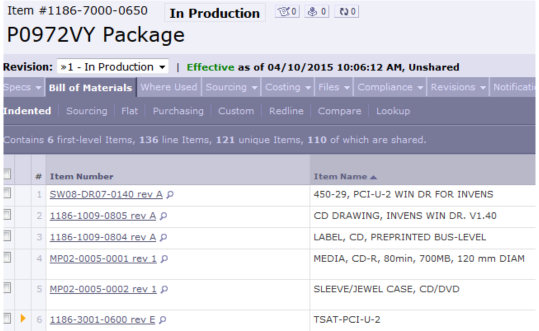
P0972VY packages

Summary (details further below)

- **1186-7000-0650** (**Rev 1)** is the most recent kit (since 4/25/2015) with a TSAT-PCI-U-2 board (no GPS antenna)
- **1159-7000-0650** (**Rev A**) is the same original rev, but with TSAT-PCI-U (no antenna)
- **1159-7000-0650 (Rev A):** earliest package with TSAT-PCI-U, antenna (?)

P0972VY (1186-7000-0650) (Rev 1) : Invensys package with TSAT-PCI-U-2 board (**no antenna included**)

- **In Arena at** https://app.bom.com/items/detail-spec?item_id=1209127787&version_id=10319695788&orb_msg_single_search_p=1
- **Rev 1 on** 4/10/2015



Item #1186-7000-0650 **In Production** 0 0 0

P0972VY Package

Revision: »1 - In Production | **Effective as of 04/10/2015 10:06:12 AM, Unshared**

Contains **6** first-level Items, **136** line Items, **121** unique Items, **110** of which are shared.

#	Item Number	Item Name
1	SW08-DR07-0140 rev A	450-29, PCI-U-2 WIN DR FOR INVENS
2	1186-1009-0805 rev A	CD DRAWING, INVENS WIN DR. V1.40
3	1186-1009-0804 rev A	LABEL, CD, PREPRINTED BUS-LEVEL
4	MP02-0005-0001 rev 1	MEDIA, CD-R, 80min, 700MB, 120 mm DIAM
5	MP02-0005-0002 rev 1	SLEEVE/JEWEL CASE, CD/DVD
6	1186-3001-0600 rev E	TSAT-PCI-U-2

P0972VY (1159-7000-0650) (Rev A): Invensys package with TSAT-PCI- board (no antenna ?)

- In Visual
- **Switched to P/N 1186-7000-0650 (Rev 1) (above)** on 4/10/2015

P0972VY (1159-7000-0650) (Rev A): Invensys package with TSAT-PCI-U, antenna (?)

- In Visual

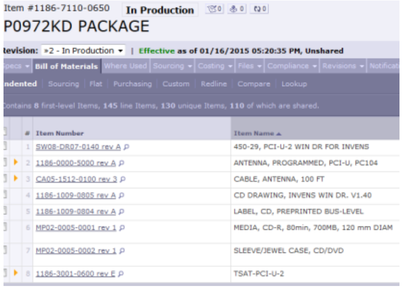
P0972KD packages

Summary (details further below)

- **1186-7110-0650 (Rev 2)** is the more recent kit (since 16 Jan 2015) with a **PCI-U-2 FXA** board (and **Acutime GG antenna**)
- **1159-7110-0650 (Rev A**) is the same original rev but with **TSAT-PCI-U-FXA** (and **Acutime Gold antenna**)
- **1159-7110-0650** (**Rev A):** is the earliest kit with **TSAT-PCI-U opt FXA** (and **Acutime Gold antenna)**

P0972KD (1186-7110-0650) (Rev 2): Invensys package with TSAT-PCI-U2 -FXA and ***Acutime GG*** GNSS antenna (P/N 1186-0000-5000)

- **In Arena at**: https://app.bom.com/items/detail-spec?item_id=1203664186&version_id=10258976728&orb_msg_single_search_p=1
- **Rev 2 effective on** 16 Jan 2015



Item #1186-7110-0650 **In Production**

P0972KD PACKAGE

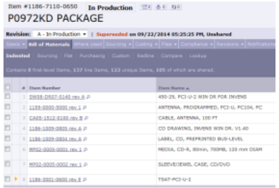
Revision: »2 - In Production | **Effective as of 01/16/2015 05:20:35 PM, Unshared**

Contains 8 first-level Items, 145 line Items, 130 unique Items, 110 of which are shared.

#	Item Number	Item Name
1	SW08-DR07-0140 rev A	450-29, PCI-U-2 WIN DR FOR INVENS
2	1186-0000-5000 rev A	ANTENNA, PROGRAMMED, PCI-U, PC104
3	CA05-1512-0100 rev 3	CABLE, ANTENNA, 100 FT
4	1186-1009-0805 rev A	CD DRAWING, INVENS WIN DR. V1.40
5	1186-1009-0804 rev A	LABEL, CD, PREPRINTED BUS-LEVEL
6	MP02-0005-0001 rev 1	MEDIA, CD-R, 80min, 700MB, 120 mm DIAM
7	MP02-0005-0002 rev 1	SLEEVE/JEWEL CASE, CD/DVD
8	1186-3001-0600 rev E	TSAT-PCI-U-2

P0972KD (1186-7110-0650 (Rev A): Invensys package with TSAT-PCI-U2 -FXA ,and **Acutime Gold** GPS antenna (P/N 1159-0000-5000)

- In Arena at: https://app.bom.com/items/detail-spec?item_id=1203664186&version_id=10258976728&orb_msg_single_search_p=1
- **Rev A effective** ~2 Jun 2014 (until 9/22/2015)
- **Switched to** 1186-7110-0650 Rev 2 (Above) on 16 Jan 2015



Item #1186-7110-0650 **In Production**

P0972KD PACKAGE

Revision: A - In Production | **Superseded on 09/22/2014 05:25:25 PM, Unshared**

Contains 8 first-level Items, 137 line Items, 122 unique Items, 105 of which are shared.

#	Item Number	Item Name
1	SW08-DR07-0140 rev A	450-29, PCI-U-2 WIN DR FOR INVENS
2	1159-0000-5000 rev 1	ANTENNA, PROGRAMMED, PCI-U, PC104, PC
3	CA05-1512-0100 rev B	CABLE, ANTENNA, 100 FT
4	1186-1009-0805 rev A	CD DRAWING, INVENS WIN DR. V1.40
5	1186-1009-0804 rev A	LABEL, CD, PREPRINTED BUS-LEVEL
6	MP02-0005-0001 rev 1	MEDIA, CD-R, 80min, 700MB, 120 mm DIAM
7	MP02-0005-0002 rev 1	SLEEVE/JEWEL CASE, CD/DVD
8	1186-3001-0600 rev E	TSAT-PCI-U-2

P0972KD (1159-7110-0650) (Rev A): Invensys package with **TSAT-PCI-U** opt FXA and **Acutime Gold** GPS Antenna (P/N 1159-0000-5000) and other accessories

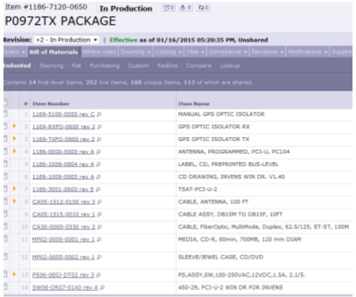
P0972TX

Summary (details further below)

- **1186-7120-0650 (Rev 2):** is the more recent kit with **TSAT-PCI-U-2 opt FXA** (and **Acutime GG Smart** antenna)
- **1159-7120-0650 (Rev A**): is the same original rev but with **TSAT-PCI-U-2 opt FXA** (and **Acutime Gold** GPS antenna)
- **1159-7120-0650** (**Rev A): i**s the earliest kit with **TSAT-PCI-U opt FXA** (and **Acutime Gold** GPS antenna

P0972TX (1186-7120-0650) (Rev 2): Invensys package with TSAT-PCI-U-2 opt FXA, **Acutime GG Smart** antenna (P/N 1186-0000-5000)

- **in Arena**: https://app.bom.com/items/detail-spec?item_id=1203664187&version_id=10258976738&orb_msg_single_search_p=1
- **Rev 2 effective:** 16 Jan 2015



Item #1186-7120-0650 In Production

P0972TX PACKAGE

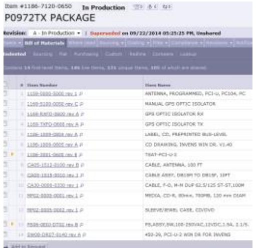
Revision: »2 - In Production | Effective as of 01/16/2015 05:20:35 PM, Unshared

Contains 14 first-level Items, 252 line Items, 188 unique Items, 113 of which are shared.

#	Item Number	Item Name
1	1169-5100-0050 rev C	MANUAL GPS OPTIC ISOLATOR
2	1169-RXFO-0600 rev 2	GPS OPTIC ISOLATOR RX
3	1169-TXFO-0600 rev 2	GPS OPTIC ISOLATOR TX
4	1186-0000-5000 rev A	ANTENNA, PROGRAMMED, PCI-U, PC104
5	1186-1009-0804 rev A	LABEL, CD, PREPRINTED BUS-LEVEL
6	1186-1009-0805 rev A	CD DRAWING, INVENS WIN DR. V1.40
7	1186-3001-0600 rev E	TSAT-PCI-U-2
8	CA05-1512-0100 rev 3	CABLE, ANTENNA, 100 FT
9	CA05-1515-0010 rev 1	CABLE ASSY, DB15M TO DB15F, 10FT
10	CA30-0000-0330 rev 2	CABLE, FiberOptic, MultiMode, Duplex, 62.5/125, ST-ST, 100M
11	MP02-0005-0001 rev 1	MEDIA, CD-R, 80min, 700MB, 120 mm DIAM
12	MP02-0005-0002 rev 1	SLEEVE/JEWEL CASE, CD/DVD
13	PS06-0E0J-DT02 rev 3	PS,ASSY,SW,100-250VAC,12VDC,1.5A, 2.1/5.
14	SW08-DR07-0140 rev A	450-29, PCI-U-2 WIN DR FOR INVENS

P0972TX (1186-7120-0650) (Rev A): Invensys package with TSAT-PCI-U-2 opt FXA, **Acutime Gold** GPS antenna (P/N 1159-0000-5000)

- **in Arena**: https://app.bom.com/items/detail-spec?item_id=1203664187&version_id=10258976738&orb_msg_single_search_p=1
- **Rev A effective:** 2 Jun 2014 (until 16 Jan 2015)



Item #1186-7120-0650 In Production

P0972TX PACKAGE

Revision: A - In Production | Superseded on 09/22/2014 05:25:25 PM, Unshared

Contains 14 first-level Items, [illegible] line Items, [illegible] unique Items, [illegible] of which are shared.

#	Item Number	Item Name
1	1159-0000-5000 rev 1	ANTENNA, PROGRAMMED, PCI-U, PC104, PC
2	1169-5100-0050 rev C	MANUAL GPS OPTIC ISOLATOR
3	1169-RXFO-0600 rev A	GPS OPTIC ISOLATOR RX
4	1169-TXFO-0600 rev A	GPS OPTIC ISOLATOR TX
5	1186-1009-0804 rev A	LABEL, CD, PREPRINTED BUS-LEVEL
6	1186-1009-0805 rev A	CD DRAWING, INVENS WIN DR. V1.40
7	1186-3001-0600 rev B	TSAT-PCI-U-2
8	CA05-1512-0100 rev B	CABLE, ANTENNA, 100 FT
9	CA05-1515-0010 rev 1	CABLE ASSY, DB15M TO DB15F, 10FT
10	CA30-0000-0330 rev 1	CABLE, F-O, M-M DUP 62.5/125 ST-ST,100M
11	MP02-0005-0001 rev 1	MEDIA, CD-R, 80min, 700MB, 120 mm DIAM
12	MP02-0005-0002 rev 1	SLEEVE/JEWEL CASE, CD/DVD
13	PS06-0E0J-DT02 rev B	PS,ASSY,SW,100-250VAC,12VDC,1.5A, 2.1/5.
14	SW08-DR07-0140 rev A	450-29, PCI-U-2 WIN DR FOR INVENS

P0972TX (Old version 1159-7120-0650) (Rev A): Invensys package with TSAT-PCI-U opt FXA and **Acutime Gold** antenna (P/N 1159-0000-5000)

- **In Visual:**
- **Switched to 1186-7120-0650 (Rev A) (above)** 2 Jun 2014

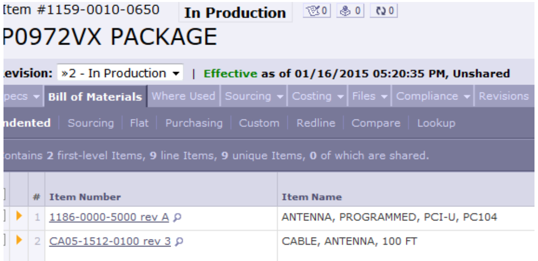
P0972VX

Summary (details further below)

- **1159-0010-0650 (Rev 2)** is the more recent kit (since ~16 Jan 2015) with antenna cable and **Acutime GG Smart antenna** (no Timing board)
- **1159-0010-0650 (Rev A**) is the original rev of the kit with antenna cable and **Acutime Gold antenna**

P0972VX (1159-0010-0650) (Rev 2): Invensys package (no timing board) antenna cable and **Acutime GG Smart** antenna (P/N 1186-0000-5000) (no Timing board)

- in Arena: https://app.bom.com/items/detail-spec?item_id=1202841009&version_id=10258976748&orb_msg_single_search_p=1
- Rev 2 effective on ~16 Jan 2015



Item #1159-0010-0650 **In Production** 0 0 0

P0972VX PACKAGE

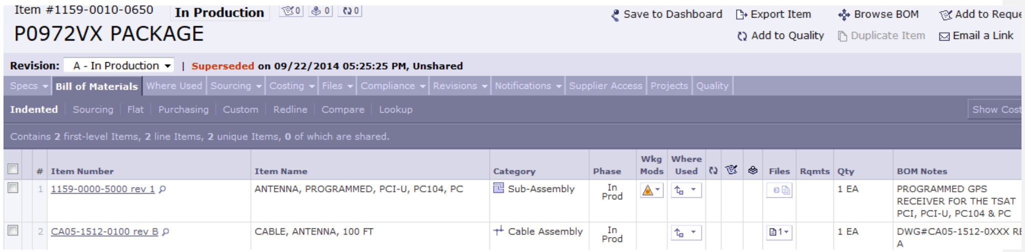
Revision: »2 - In Production | **Effective as of 01/16/2015 05:20:35 PM, Unshared**

ontains **2** first-level Items, **9** line Items, **9** unique Items, **0** of which are shared.

#	Item Number	Item Name
1	1186-0000-5000 rev A	ANTENNA, PROGRAMMED, PCI-U, PC104
2	CA05-1512-0100 rev 3	CABLE, ANTENNA, 100 FT

P0972VX (1159-0010-0650) (Rev A): Invensys package, (no timing board) antenna cable and **Acutime Gold** antenna (P/N 1159-0000-5000)

- **in Arena**: https://app.bom.com/items/detail-spec?item_id=1202841009&version_id=10258976748&orb_msg_single_search_p=1
- Rev A effective ~3 Apr 2014 (until ~ 22 Sept 2014)



Item #1159-0010-0650 **In Production** 0 0 0

P0972VX PACKAGE

Revision: A - In Production | **Superseded on 09/22/2014 05:25:25 PM, Unshared**

Contains **2** first-level Items, **2** line Items, **2** unique Items, **0** of which are shared.

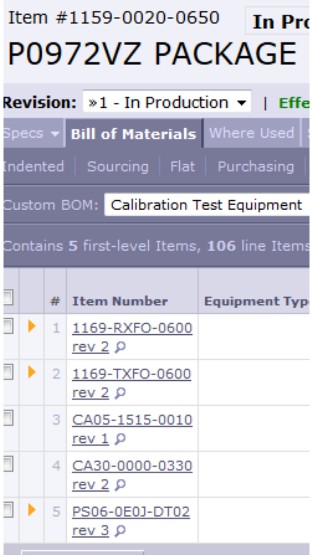
□	#	Item Number	Item Name	Category	Phase	Wkg Mods	Where Used				Files	Rqmts	Qty	BOM Notes
□	1	1159-0000-5000 rev 1	ANTENNA, PROGRAMMED, PCI-U, PC104, PC	Sub-Assembly	In Prod	⚠							1 EA	PROGRAMMED GPS RECEIVER FOR THE TSAT PCI, PCI-U, PC104 & PC
□	2	CA05-1512-0100 rev B	CABLE, ANTENNA, 100 FT	Cable Assembly	In Prod						1		1 EA	DWG#CA05-1512-0XXX RE A

P0972VZ

P0972VZ (1159-0020-0650) (**Rev A):** Invensys package, GPS Optic isolators (TX And RX), fiber cable and power supply (**no timing board and no GPS antenna provided**)

- (in Arena: https://app.bom.com/items/detail-bom-custom?item_id=1202841010&version_id=10299430268&&redirect_Segno=9933902681)

- Effective since 10 Apr 2015



Item #1159-0020-0650 **In Pro**

P0972VZ PACKAGE

Revision: »1 - In Production | **Effe**

Custom BOM: Calibration Test Equipment

Contains **5** first-level Items, **106** line Items

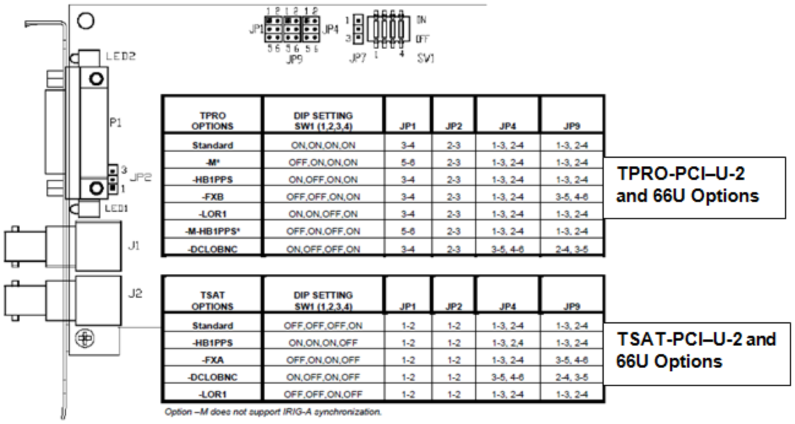
#	Item Number	Equipment Typ
1	1169-RXFO-0600 rev 2	
2	1169-TXFO-0600 rev 2	
3	CA05-1515-0010 rev 1	
4	CA30-0000-0330 rev 2	
5	PS06-0E0J-DT02 rev 3	

Options

(TPRO/TSAT-PCI-U-2 and TPRO/TSAT-PCI-66U boards)

- Also refer to the manual for more info on available Options.
- In the earlier PCI series, options need to be added at the factory.
- With PCI-66U and PCI-U-2 boards, the Options can be configured via jumper/switch settings.

All available options can now be added /removed in the field, using Jumper and DIP switches on the timing board. Refer to Section 9 of the user manual for more information on configuring the options.



TPRO OPTIONS	DIP SETTING SW1 (1,2,3,4)	JP1	JP2	JP4	JP9
Standard	ON,ON,ON,ON	3-4	2-3	1-3, 2-4	1-3, 2-4
-M*	OFF,ON,ON,ON	5-6	2-3	1-3, 2-4	1-3, 2-4
-HB1PPS	ON,OFF,ON,ON	3-4	2-3	1-3, 2-4	1-3, 2-4
-FXB	OFF,OFF,ON,ON	3-4	2-3	1-3, 2-4	3-5, 4-6
-LOR1	ON,ON,OFF,ON	3-4	2-3	1-3, 2-4	1-3, 2-4
-M-HB1PPS*	OFF,ON,OFF,ON	5-6	2-3	1-3, 2-4	1-3, 2-4
-DCLOBNC	ON,OFF,OFF,ON	3-4	2-3	3-5, 4-6	2-4, 3-5

TPRO-PCI–U-2 and 66U Options

TSAT OPTIONS	DIP SETTING SW1 (1,2,3,4)	JP1	JP2	JP4	JP9
Standard	OFF,OFF,OFF,ON	1-2	1-2	1-3, 2-4	1-3, 2-4
-HB1PPS	ON,ON,ON,OFF	1-2	1-2	1-3, 2,4	1-3, 2-4
-FXA	OFF,ON,ON,OFF	1-2	1-2	1-3, 2-4	3-5, 4-6
-DCLOBNC	ON,OFF,ON,OFF	1-2	1-2	3-5, 4-6	2-4, 3-5
-LOR1	OFF,OFF,ON,OFF	1-2	1-2	1-3, 2-4	1-3, 2-4

TSAT-PCI–U-2 and 66U Options

Option –M does not support IRIG-A synchronization.

A) Option -LOR1

- Automatically available with PCI-U-2 and PCI-66U boards (configured with DIP switches)
- Was a purchased Option with PCI-U timing boards

This option provides the board with three different outputs on a three-pin header;

- **JP11 Pin 6** - 1 Megahertz Output (1MHZ)
- **JP11 Pin 7** - 1 Pulse Per Second Output (1PPS)
- **JP11 Pin 8** - Ground Output.

The 3-pin header is located on JP11 Pins 6, 7, and 8.

B) Option -M (1PPS input)

- Automatically available with PCI-U-2 and PCI-66U boards (configured with DIP switches)
- Was a purchased Option with PCI-U timing boards

1PPS input Wiring:

Option M (1PPS input synchronization) pin-outs:

DB15 (P1) pins 15 (1PPS) and 5 or 7 (ground).

1PPS is automatically detected and front panel LED will indicate "Synced" when the signal is present. Time can be manually set using the Windows application.

Yes the JP2 pin #1 is on the bottom.

Jumper configuration:

PRO-PCI OPTION	JP1	JP2	JP4	JP7	JP9
-M	5-6	2-3	1-3, 2-4	None	1-3, 2-4

1PPS input requirements:

I wonder if there could be a problem with the 1PPS Input. If it is intermittent or unstable it could be causing this problem. Here is an excerpt from the manual concerning the time being set to zero:

The board expects the 1PPS input to be continuous. If the 1PPS signal stops pulsing after the board establishes initial sync, the board will continue to increment time ("freewheel"). However, if the 1PPS signal resumes after a period of freewheeling, the board may reset the clock time to 000:00:00:00.000000. This is because the 1PPS occurs outside of a narrow window in which the board expects it, either because the 1PPS has moved or because the board's time has drifted during freewheeling.

The arm command (0x004c) must be sent no sooner than 50 mS after the previous 1PPS occurred, and no later than 50 mS before the next (arming) 1PPS occurs.

From what kind of device are you receiving the 1PPS signal? Is it stable enough to meet the spec?

Here are the specs for the 1PPS Input:

1 PPS Sync Input Specifications (Option –M Only)
Input Voltage 2.4 V min, 16.0 V max (high)
(500 mA max at 5 Vin, 12 mA max at 16 Vin)
Rise/Fall Time 500 nS max
Trigger Edge: Rising
1PPS Accuracy Must be 100 ppm or better

Notice the accuracy of the 1PPS must be 100 ppm or better, or the sync will be lost and the time reset to 00:00:00.

Email from Wade Sober to Elotek (11 Feb 2015) The TPRO-PCI-66U card can be configured through dip switches to be able to accept 1PPS as an input reference. This would be connected through the 15 pin D connector on the edge of the board. To configure this use the dip switch settings for Option M. With this option enabled you could connect either an IRIG B signal or 1PPS to synchronize the card. If both signals are connected the board will automatically use IRIG B as the default. If IRIG B goes away the board would then use the 1PPS input as the reference.

Option -FXA

Note: Options FXA and FXB are often purchased by Invensys or Invensys customers. These customers having issues with PCI series boards should contact Ed Pittsley with Invensys for help. Refer to: Invensys Support in this document.

- Automatically available with PCI-U-2 and PCI-66U boards (configured with DIP switches)
- Was a purchased Option with PCI-U timing boards
- **Process Detail**- Refer to (1159-0001-0602): I:\New Released\Process Details\1159-xxxx-xxxx (TPRO-TSAT-PCI-U) Process Details\
- **TSAT-PCI-U P/N:** 1159-0001-0602
- **TSAT boards Only (not available with TPRO);** Cannot Be Combined with Options -HB1PPS, -HDRV
- Provides balanced RS-422 Heartbeat output on pins 6 (normally low, pulses high) and 8 (normally high, pulses low) of the standard DB15 connector.

This option provides an RS-422 driver for the Heartbeat Output, and provides the same range of pulse rates as the –HB1PPS Option. The 15-pin, D-type connector provides the Heartbeat Output on Pin 6 (normally low, pulses high) and Pin 8 (normally high, pulses low). The rising edge of Pin 6 (falling edge of Pin 8) is *on-time*. The pulse width is approximately 1 mS. The power-on default pulse rate is once per second (1PPS). The Match function can still be used to provide an interrupt, if desired. However, the Match Output Pin has been eliminated. There are *no jumpers* installed on Header JP9.

Option -FXB

Note: Options FXA and FXB are often purchased by Invensys or Invensys customers. These customers having issues with PCI series boards should contact Ed Pittsley with Invensys for help. Refer to: Invensys Support in this document.

- Automatically available with PCI-U-2 and PCI-66U boards (configured with DIP switches)
- Was a purchased Option with earlier PCI-U timing boards
- TPRO-PCI-U P/N: 1159-0002-0603
- TPRO Only; Cannot Be Combined with Options -HB1PPS, -HDRV
- Provides balanced RS-422 Heartbeat output on pins 6 (normally low, pulses high) and 8 (normally high, pulses low) of the standard DB15 connector.

This option provides an RS-422 driver for the Heartbeat Output, and provides the same range of pulse rates as the –HB1PPS Option. The 15-pin, D-type connector provides the Heartbeat Output on Pin 6 (normally low, pulses high) and Pin 8 (normally high, pulses low). The rising edge of Pin 6 (falling edge of Pin 8) is *on-time*. The pulse width is approximately 1 mS. The power-on default pulse rate is once per second (1PPS). The Match function can still be used to provide an interrupt, if desired. However, the Match Output Pin has been eliminated. There are *no jumpers* installed on Header JP9.

Option -DCLOBNC (IRIG DCLS output)

- Automatically available with PCI-U-2 and PCI-66U boards (configured with DIP switches)
- Was a purchased Option with earlier PCI-U timing boards

From the TPRO/TSAT user's manual

This option provides an IRIG-B, DCLS output on the J2 BNC connector and P1 D-type connector Pin 9. The BNC or the 15-pin D-type connector provides a DC level shifted output with the rising edge on time. The modulated IRIG-B output is eliminated with this option.

- Provides IRIG DCLS output (does not provide IRIG DCLS input capability). The standard output is IRIG B122. With this option added (earlier TPRO/TSAT-PCI series boards) or with the option enabled (newer TPRO/TSAT-PCI-U-2 and 66U series boards), the signal becomes IRIG B022

Email from Tim Tetreault (8/3/12)
The-DCLOBNC option on the TSAT/TPRO-PCI boards is a DCLS output. The boards just can't read DCLS. If they setup the jumper and DIP switch as per the manual, they can source DCLS.

Status LEDS

**Green LED on edge of card

- The green LED on the edge of the card indicates power is applied and the FPGA has been loaded (It does not indicate sync status).
- As of at least 9/1/1, this LED is not currently discussed in the PCI-U-2/66-U user manual (it just shows a picture of it).

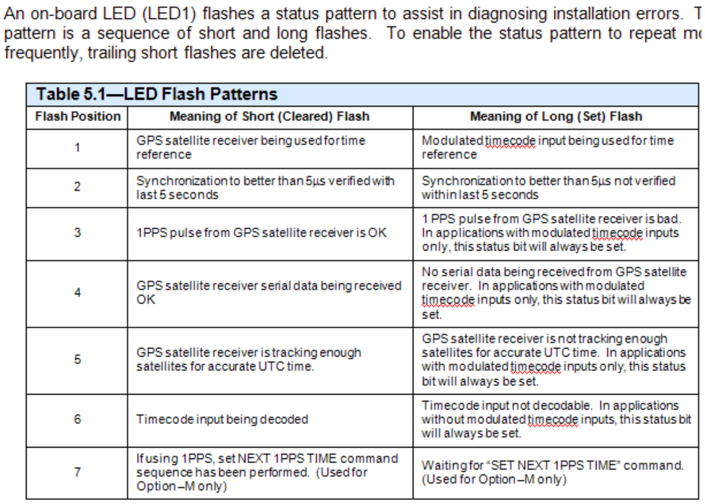
**Sync Status (red LED1 status LED) and API call

A) Determine sync status via the Windows Control Utility (windows only)

B) Determine sync status via API call

- API call to obtain sync status: **TPRO_SynchStatus**

C) Red Status LED on edge of the board (to the right of the two BNC connectors)



An on-board LED (LED1) flashes a status pattern to assist in diagnosing installation errors. T pattern is a sequence of short and long flashes. To enable the status pattern to repeat m frequently, trailing short flashes are deleted.

Table 5.1—LED Flash Patterns

Flash Position	Meaning of Short (Cleared) Flash	Meaning of Long (Set) Flash
1	GPS satellite receiver being used for time reference	Modulated timecode input being used for time reference
2	Synchronization to better than 5µs verified with last 5 seconds	Synchronization to better than 5µs not verified within last 5 seconds
3	1PPS pulse from GPS satellite receiver is OK	1 PPS pulse from GPS satellite receiver is bad. In applications with modulated timecode inputs only, this status bit will always be set.
4	GPS satellite receiver serial data being received OK	No serial data being received from GPS satellite receiver. In applications with modulated timecode inputs only, this status bit will always be set.
5	GPS satellite receiver is tracking enough satellites for accurate UTC time.	GPS satellite receiver is not tracking enough satellites for accurate UTC time. In applications with modulated timecode inputs only, this status bit will always be set.
6	Timecode input being decoded	Timecode input not decodable. In applications without modulated timecode inputs, this status bit will always be set.
7	If using 1PPS, set NEXT 1PPS TIME command sequence has been performed. (Used for Option–M only)	Waiting for "SET NEXT 1PPS TIME" command. (Used for Option–M only)

Where:

- **Short =** 10% duty cycle on/off (lit for about 0.1 seconds, not lit for about 0.9 seconds)
- **Long =** 50% duty cycle on/off (lit for about 0.5 seconds, not lit for 0.5 seconds)

Analysis of the expected flash pattern with a good input

- **TSAT-PCI (GPS input)** = *pause* Short Short Short Short Short Short *pause*
  (IRIG input) = *pause* Long Short Long Long Long Short *pause*

- **TPRO-PCI =** *pause* Long Short Long Long Long Short *pause*

Note: The flash pattern with "Option –m" installed/enabled may add a 7th long (but not a 7th short, because a last short is deleted)

What is meant by the "trailing short flashes are deleted", is that there may or may not be a total of 7 flash/blinks in a row. For instance, if the last two positions (positions 6 and 7 in the table above) both happen to be a long flash (instead of a quick blink), these two long flashes will not be displayed and therefore, there will only be a series of 5 blinks/flashes displayed. However, if positions 6 and 7 are classified as both being blinks, there will be a total of 7 LED lights in a row, instead, with the last two positions being a quick blink.

Inputs / Time Sync / Holdover

**Input Sync accuracies

The PCI and PC104 boards have the following accuracy specs of the timing boards syncing to their input reference:

- **With GPS input:** the timing boards will discipline to within +/- 1 microsecond of the GPS input.
- **With IRIG input:** the timing boards will discipline to within +/- 5 microseconds of the IRIG time source.

The reason for the difference between the two specs is because the GPS reference is much more stable (has less jitter) than an IRIG input. If your customer wants the PC104 timing board to be synced within 1 microsecond, they will need to synchronize the board using its GPS antenna (installed outdoors with a good view of the sky so that it can track at least four satellites at all times). Otherwise, if they sync the board using an IRIG generator, it will be synced with less accuracy to the IRIG time generator.

**Default Year value/ Manually setting the Year (GPS or IRIG input)

- TPRO/TSAT boards default to the year value of "0000" after each boot-up (TSync series boards default to "2000")

The TPRO boards do not (and cannot) read the year from the IRIG input. The year must be set by the user or via the driver after each power-up when using IRIG input. To set the year automatically, use a TSAT board synced via GPS (The TSync-PCIe timing boards can be programmed to both read and distribute year information on the IRIG output).

Default year value at power-up:

From the PCI-U-2 manual: The year is reset to **0000** when power is first applied, or when any of the following occur: system reset, firmware reset (command 0x004f), or writing to the "Assert Reset" or "De-assert Reset" addresses.

Note per Tim Tetreault (15 Jan 2013): the default year of **0000** is treated as a non Leap Year (day 1 through 365).

If the year is not set after each boot-up, the Day of Year will increment on December 31st, but the year will remain 0000, until it's been set by GPS (TSAT board) or being commanded to the correct year (when using IRIG input) (refer to notes in the Lear Year section further below).

Need to set the year with IRIG input

- API call to set the year: **TPRO_SetYear** (as listed in the Application Programmers Guide)

Note: Apparently, the year can also be set in Windows with a command that can be run in a custom batch file

- Is there a command I could run to automate this procedure on startup via batch script or vbs script?
- Reply from Dave Lorah (16 Jul 14) We do not have a batch file but if you run this command: setyear –y %date:10,4% on startup this will set the year to what the computers year is set to.

TPRO boards (IRIG input only):

- The current year has to be set after every reboot of the timing board.
- The board can never read the year information from the IRIG input signal.
- The year can be set using the Windows Control Utility, or via the driver API call
- The year value can be commanded to the correct year (refer to “Setting the year” in the user manual).

TSAT boards (IRIG or GPS input):

- The current year has to be set after every reboot of the timing board.
- When synced via GPS, the year information is automatically set.
- When using IRIG input only (no GPS input), the board cannot read the year information from the IRIG input signal.
- When using IRIG input only, the year can be set using the Windows Control Utility, or commanded to the correctly year via the driver (refer to “Setting the year” in the user manual).

The Spectracom TPRO-PCI-66U board can accept an IRIG AM (Amplitude Modulated) input signal for synchronization (IRIG A132, B122 or NASA36). The IRIG input signal should be around 3.5Vp-p into high impedance (IRIG AM has two amplitudes, one is for Mark and one is for Space). The larger of these two amplitudes should be approximately 3.5Vp-p).

When using a unbalanced IRIG AM input signal (one IRIG signal and one ground), make sure the IRIG time code generator is connected to the BNC connector “J1” on the PCI-66U timing board (The BNC connector that is closest to the DB15 connector which is also on the edge of the board). When using a balanced IRIG AM signal (two signals and ground) you can also provide the IRIG timecode to Pins 1 and 2 of the DB15 connector (P1). If using this connector, try switching the + and – pins (pins 1 and 2) and see if the timing board declares sync. It may be reverse-pinned.

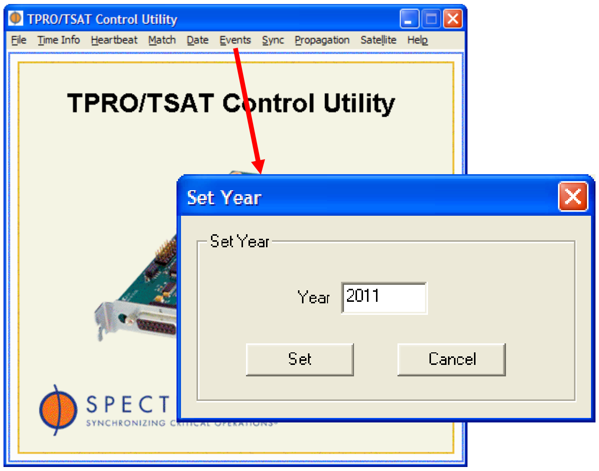
Using the oscilloscope, make sure there are actually two sine wave amplitudes present (and not just one continuous 1kHz sine wave). One is larger than the other, with a ratio of one amplitude to the other being 3.3 to1. The higher of the two amplitudes determines the reported amplitude of the IRG signal.

Note: I can’t vouch for other manufacturer’s IRIG generators, but ours have a feature called Signature Control which, when enabled, will stop the modulation on the IRIG signal if the generator is not synced to an external reference, such as GPS. In this mode, the carrier frequency of a 1kHz signal will be present, but there won’t be a second, different amplitude signal present until it has synced to GPS. Then, two different amplitudes will be present. If there is only one amplitude present, there is no time code modulation present on the signal and the board won’t be able to sync to the IRIG input.

The IRIG input signal needs to be fairly stable in order to allow the PCI-66U’s oscillator to discipline to the input signal. The IRIG generator should have an external input applied (such as GPS) or should be using a very stable internal oscillator to ensure there is very little jitter on the IRIG input signal. Otherwise, the PCI-66U board may not be able to synchronize to the input signal.

Because the TPRO/TSAT-PCI-66U boards do not read the present year value from the IRIG data stream (even if this optional value happens to be present in the IRIG signal), the present year value (such as “2011”) must be set into the timing board each time the board powers-up, before it can go into sync (Note: When using GPS input with a TSAT timing board, the year is set automatically by the GPS signal).

There are a couple of ways the year can be set in a TPRO timing board, each time it powers up. One of these methods is via the Windows Control Utility, when the timing board is installed in a Windows PC (and the Windows PCI driver has been installed).



Year 2011

Use the: “**Date**” –> “**Set Year**” menus

The present year value can also be set via an API call using the Windows or Linux driver. The **TPRO_SetYear** call can be performed after each boot-up to set the current year value. Refer to Section 3.2.14 (page 3-9) of the attached Linux Driver Application Programmer’s guide for more information on using the **TPRO_SetYear** API call**.**

Lastly, the motherboard needs to provide certain DC voltages to the TPRO-PCI-66U board (via the bus interface) in order for the timing board to operate normally. All of the voltages that need to be present on the PC bus are +5V, +12V and -12V. The +12V and -12V are especially important for IRIG input and output, as the timing board cannot sync to IRIG if this voltage is not present on the bus connector and the IRIG output will be distorted if both rails are not present and correct.

** Desire to manually set the Date or Time / Default Day of Year and Time at boot-up

The PCI board starts counting at 000 days, 00 hours, 00 minutes, 00 seconds at power-on.
The clock automatically synchronizes DOY, hours, minutes and seconds to specified timecode signals (GPS also set the current year- IRIG input does not set the current year- which defaults to 0000 at power-up).
The clock time can also be set using the Windows Control Utility or by user command. (The manual provides two sets of instructions on setting the year. One is used if Option –M is not installed and the other is used if Option –M is installed).

The Time cannot be manually set successfully on a TPRO/TSAT-PCI board if GPS and/or IRIG input is present, unless the “Disable Sync” command has been executed:

Manually setting the Date/Time

- The **TPRO_SetTime** API call sets the DOY, hours, minutes and seconds.
- The **TPRO_SetYear** API call sets the year.
- Refer to Status LEDS further below to verify sync status

Note: Apparently, the year can also be set in Windows with a command that can be run in a custom batch file

Q Is there a command I could run to automate this procedure on startup via batch script or vbs script?
A Reply from Dave Lorah (16 Jul 14) We do not have a batch file but if you run this command: **setyear –y %date:10,4%** on startup this will set the year to what the computers year is set to.

One important item to keep in mind- unlike our TSync-PCIe timing boards, the TPRO/TSAT-PCI timing boards do not have the ability to declare sync status via manually set time. A user can manually set the time of the TPRO/TSAT-PCI boards as desired, but the TPRO/TSAT-PCI board will not declare its in sync based on manually set time. In order for the board to be in sync, it needs either an IRIG or GPS input (TSAT-PCI) OR an IRIG input (TPRO-PCI) be applied.
Depending on the intentions for this timing board, the lack of Sync status may or may not be an issue. For example, if they want to generate an IRIG signal, as the IRIG output does not report Sync status, manually setting the time won't be an issue. But if they are trying to sync a PC to the time of the board, this requires the board to be in sync. So the board will need to be synced to IRIG or GPS (which override manually set time)

Manually setting the time of the TPRO/TSAT-PCI board requires an API call be sent to the board (or it can also be set via the Windows Control Utility, using the **Time Info**-> **Set** menu). The **TPRO_SetTime** API call specifies the desired time to set the board to (Specifies Day of Year, Hours, Minutes and seconds. The **TPRO_SetYear** API call is a separate API call for setting just the desired year.

Overriding GPS/IRIG input, if present, in order to be able to manually set the time/DOY

From the PCI manual:

7.13 Enable/Disable Sync

The board can be forced to freewheel by sending the "Disable Sync" command. Sending this command causes the board to ignore the timecode input, the 1PPS input (Option –M only), and the GPS input (TSAT-PCI-U only).To restore normal operation; send the "Enable Sync" command. To send the Disable Sync command, write 0x004e to the command port. To send the Enable Sync command, write 0x004d to the command port. The power-on default is to Enable Sync

Desire to manually set the time when an external trigger occurs

Q. Does the card have the capability to receive a trigger and set the IRIG time to any time I want. For example, I have a signal generator in my PXI system that sends out a trigger when it starts waveform generation. I'd like this device to go to a preset time upon receiving that trigger.
A Reply from Keith (22 Apr 13) This may be able to do this desired functionality via their custom application software. However, the timing board itself doesn't have any provisions to go to a pre-set time based solely on an input trigger.

An input trigger can be used to generate a time stamp of when the time stamp was received.   Your customer may be able to create their own software that sends a pre-configured **SetTime** (and **setYear**, if the year needs to be set) API call at particular moment. But the board doesn't have the abilty to have these value pre-entered but not used until a particular input trigger. The time and/or year is set as soon at the API call is issued.

****Year rollover issue (every year- not just leap years) in firmware versions prior to 2.00

Ed McFadden with DSPCon reported some of their customers with Windows being updated by the Clock daemon are not properly rolling over at the end of the year.

- **Refer to** Salesforce case 9791 and ECN 2562 (Dec 2010).
- Firmware issue with at least the newer PCI-U-2 and 66-U board. Likely also with earlier PCI series boards also, But Tim Tetreault didn't confirm these.
- Fixed with firmware version 2.00 (ECN 2562, Dec 2010) . Refer to: PSB, PSP software updates\TPRO-TSAT boards\board firmware\TPROTSAT-PCI-u-2 and PCI-66U boards

As indicated on IC **U19** as "**0200**"

As indicated in Windows Control Utility (**Help** -> **About**) (See Tim Tetreault)
- Firmware is in IC U19, which is soldered onto the board (it's not a socketed IC).

Only applies to IRIG input (TSATs using GPS input are not affected).

Email Keith sent (4 June 2010) I wanted to provide you with a quick status update for the New Year rollover issue you had reported to us earlier, regarding the Spectracom TPRO-PCI timing boards. You had reported that the TPRO-PCI timing boards were incrementing by two years instead of just one year, after each New Year rollover.

We have now been able to determine the cause of this rollover anomaly. A change that resolves this issue is being incorporated into the software. The new software is currently being tested/verified and will soon be released. At this time, we expect the ECN releasing the incorporated changes to be completed by the week of December 13th.

The software update can only be applied to the TPRO-PCI timing boards as a firmware update. The IC that contains the software associated with this condition is a one-time only programmable IC and cannot be updated in the field via a software update process. The timing boards will need to be returned to the factory for a new IC to be installed onto the board.

As the completion of the ECN will be shortly before the Christmas Holiday (and the next new year rollover), in order to have your timing board(s) updated and returned to you before the Holiday, and the next rollover, we will need to have your timing board(s) returned to us on or about December 13th. If you would like the timing boards to be updated before the holidays and the next year rollover, we will also send you another status update email a few days prior to the update being released so that an RMA number can be generated for the modification and to allow time for the timing board(s) to arrive at the factory. Once the new firmware has been released, the timing boards can be updated and returned to you with the new firmware installed.

**Leap year rollover

Default year value at power-up:

- **From the PCI-U-2 manual:** The year is reset to **0000** when power is first applied, or when any of the following occur: system reset, firmware reset (command 0x004f), or writing to the "Assert Reset" or "De-assert Reset" addresses.
- **Note per Tim Tetreault (15 Jan 2013):** The default year of 0000 is treated as a non-Leap Year (Day 1 through 365).

From the TPRO user manual:
Setting the Year (Not Applicable to Option -M)

Whether the IRIG-B output is synchronized to GPS or timecode, or is freewheeling, it counts from day 365 to day 001, unless the year is set to a leap year. Once set to a leap year, the IRIG-B output counts from 365 to 366, then to 001.

Since TPRO/TSAT boards can’t read the year from IRIG, without GPS applied, the year needs to be set manually each time it’s powered-up or reset. Without GPS input, if the year is not manually set each time it’s rebooted/reset, it will assume it’s not a leap year and it will not add day 366 to the end of the year. (The year will roll over on day 365 instead of on day 366).

The year can be set with API call or with the Windows Control Utility. But if the board is not operating in a Windows /Linux environment, the year can be set manually into a register with a “poke”: From the PCI user manual:
Setting the Day of Year to 366, without setting the year (year is still at default value of “0000”)

- Calendar day goes to Dec 32nd –not Jan 1st.
- If the year is not set, the Year will not rollover to the next year on January 1st- it remains the default year value of “0000”.

Email from Dave Lorah to Masataka with TOYO (21 Jan 2013) This morning we were able to setup a TPRO-PCI-U-2 board on a Windows PC to verify what happens when you set the day manually to 366 and do not set the year value.

We powered up the computer, we did not set the year, so it was 0 and then manually set the day of year to 366. The date indicated it was Dec 32, year 0. When the card rolled over the year at Dec 32, 11:59:59 the board went to Jan 1 00:00:00 and the year was still set to 0. The board will not increment the year unless the year is initially set after power up.

Year 0 is not a valid leap year so day 366 must be manually set in this case.

Setting the Year (Not Applicable to Option -M)

To function properly, the TPRO requires that the year be set by command at the end of a leap year. The board will increment the year when the day rolls over to 001. Also, setting the year enables the board (TPRO and TSAT) to compute the Gregorian date using the year and Julian date.

Set the year by sending the command sequence "0x006n 0x007n 0x008n 0x009n 0x00ea" to the command port. For example, to set the year to 2003, send "0x0062 0x0070 0x0080 0x0093 0x00ea".
The year is reset to 0000 when power is first applied, or when any of the following occur: system reset, firmware reset (command 0x004f), or writing to the "Assert Reset" or "De-assert Reset" addresses.

Additional Note: Commands are sent to the TPRO-PC104 command register (Base Address+2) as a sequence of bytes. All commands should be spaced at least 100µs apart so the TPRO-PC104 firmware has sufficient time to handle each command.

Email KW sent to Eric Girard in France (1/26/11)
The TPRO/TSAT user manual is apparently a bit confusing about the Leap Year rollover, as your customer pointed out. However, to clarify your customer's understanding, when IRIG is the only input reference (no GPS) the year only needs to be set when the board is first powered-up. After power-up, the year will automatically increment to the new year value on each midnight on January 1st, whether or not it's a Leap year.

The timing board software knows what years are leap years and so it can compensate by knowing if the current year has 365 days or instead 366 days in a "Leap Year" year. Because it knows what years are leap years, it can automatically rollover the year value on the correct day, with no user intervention required. After every New Year rollover (leap year, or not leap year), NTP can continue getting the correct year from the timing board, without any necessary user intervention.

FYI- We have tested (and just re-verified) that the TPRO/TSAT-PC-U-2 and 66U boards do automatically handle the leap year rollover correctly, without any user intervention.

**issue with Year/date jump to Jan,1997 (in Sept, 2016) with TSAT-PCI-33U

- Reported 6 Sept 2016 by Schneider Electric (used to be Invensys)
- Refer to case 6653 in SAP

**Forced Holdover mode

7.13 Enable/Disable Sync (from PCI user manual)

The board can be forced to freewheel by sending the "Disable Sync" command. Sending this command causes the board to ignore the timecode input, the 1PPS input (Option –M only), and the GPS input (TSAT-PCI-U only).To restore normal operation, send the "Enable Sync" command. To send the Disable Sync command, write 0x004e to the command port. To send the Enable Sync command, write 0x004d to the command port. The power-on default is to Enable Sync.

1PPS input "Option –m"

- Refer to the details in the "Options" section further above for "Option –m" –M (1PPS input)

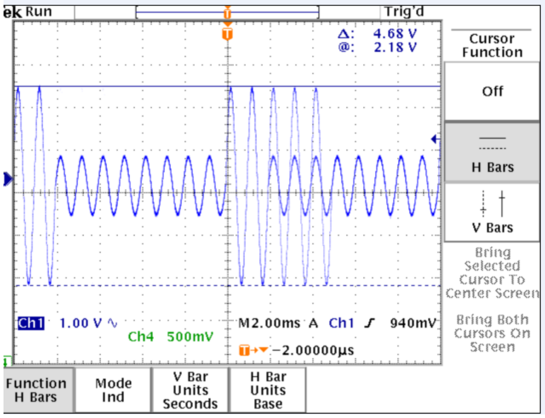
**IRIG Input (J1)

IRIG input for PCI, PCI-U, PCI-U-2 and PCI-66U

- IRIG input on connector **J1 (**BNC connector that is closer to the DB15 connector. The BNC near the edge of the board is the IRIG output)

- IRIG B AM input only (does not accept IRIG DCLS)
- Automatically detects (): **IRIG A132, IRIG B122 or NASA36.** (According to Tim Tetreault) Should also accept **B120**.

Note about B120: IRIG B122 is BCD data only (no CF control field or SBS bits).   IRIG B120 contains the CF and SBS bits, but the card should also be able to ignore these additional bits and sill sync to a B120 input.



- Refer to Status LEDS further below to verify sync status

Note: Option "-DCLOBNC" is for IRIG DCLS OUTPUT only.

Note: The motherboard needs to supply all the voltages necessary to power the board. The board requires +5V, +12V and -12V to operate. If the +12V and/or -12V is missing, the IRIG input and output will not work.

Note: If GPS and IRIG input are simultaneously present, the board defaults to IRIG synchronization.

IEEE-1344 extensions

TPRO-TSAT-PCI series boards cannot receive IEEE-1344 extensions.   They only accept IRIG A132, IRIG B122 or NASA36 which do not contain the IEEE-1344 extension. Need to use TSync-PCIe boards instead, if the extensions are required in the IRIG input.

****Desire to sync one PCI board via the IRIG output of another PCI board**

Q. Would the TPRO/TSAT-PCI board be able to accept the external IRIG B input(s) fed to another IRIG B reader card?
A. It should, just connect the IRIG output to the IRIG Input?

- Though the second board can sync via IRIG A or IRIG B, the first board can only output IRIG B122. The second board has to sync via IRIG B input.
- Based on the accuracy specs of the IRIG B input (100PPM or +/- 100 microseconds/sec) there can be up to an on offset between the two board's 1PPS.
- The amount of offset between the two boards is likely to vary each time the boards power-up, due to the IRIG

input specs mentioned above.

- If this offset between the two boards is too large, the boards should be synced to GPS instead (with GPS sync accuracy).

Sync status

- Refer to Status LEDS further below to verify sync status

***Troubleshooting IRIG input to PCI series

A) PCI series Timing boards not syncing to IRIG in higher ambient temperatures (known potential issue with PCI series and PC104 boards)

- Refer to: Known “temperature issue” with IRIG input synchronization of PCI series and PC-104 boards (in the PC104 section of this document)

General Questions for troubleshooting IRIG input issues

Just to make sure that this isn’t an issue that can be resolved without the need to return it for repair, a few quick questions for you:

- Has another TPRO-PCI board been installed in its place, and synced to the IRIG input?

Note: if the answer to this question is yes, no need to answer the remaining questions. The Timing board does likely need to be returned for eval/repair

- Can the system communicate with the timing board:
- in Windows, via either the driver/API calls or the available Windows Control utility?
- in Linux, using the driver/API calls?
- What was the specific flash pattern of the red LED that was displayed when the board was powered-up and IRIG was connected (it’s a specific pattern of shorts and/or longs- determined by how long the LED is lit for)? Did the pattern change after the IRIG cable was connected?
- The TPRO-PCI boards only accept IRIG AM input (not DCLS). Just to confirm, was it being provided IRIG AM input?
- These timing boards can only accept only IRIG A132, B120 and B122 and NASA36 formatting. Is the IRIG source providing one of these four IRIG formats? Or, is it outputting a different IRIG format?
- Was the IRIG source synced to an external reference, such as GPS? Or, was it in a free-run state? Note that if it’s a free-running IRIG reference, the stability of the IRIG output may not be stable enough for the board’s oscillator to lock/discipline to. IRIG references typically need to be locked to a reference for the IRIG signal to be stable enough to provide a useable signal.

Reply as best you can to the above questions and then we can go from there. If you would rather just send it back for us to test, let me know and I’ll assign the RMA Number for you.

**GPS input (TSAT boards only)

****Acutime GPS antennas (such as Acutime 360, Acutime-Gold, etc) for TSAT-PCI series boards

- Refer also to "**Acutime antennas**" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- Acutime Antenna/Receiver is an eight (8) Channel receiver.
- Refer to Status LEDS further below to verify GPS sync status

Specific to Acutime 360 antennas for TSAT-PCI-33U/66U timing boards

Q Can an Acutime 360 antenna be used with a TPro card? Is it an officially supported configuration?
A Reply from Ron Dries (20 Sept 2019): The Acutime 360 will work with the PCI_TSat boards it just need to be preprogrammed for NMEA.

General to Acutime antennas

The Acutime antennas are programmed at the factory to operate with ONLY TSAT-PCI-U, TSAT-U-2, TSAT-66U, TSAT-PC104 and TSAT-PC boards. They will not operate with any other timing board. More recently shipped Acutime antennas will have a label applied which indicates what timing boards that it is programmed to operate with.

Note: GPS antennas for TSAT-PCI-U, TSAT-U-2, TSAT-66U, TSAT-PC104 and TSAT-PC boards should be annotated as: "1159-0000-5000"

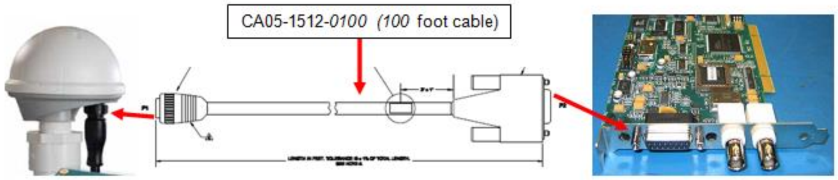
Modified Email from Tim T to Wade Sober (22 Feb 16) referring to the TSAT-PCI-U and TSAT-PCI-66U boards) How old of an antenna is the customer talking about? The current software has been tested against the Trimble Acutime Gold, Acutime 2000 & Acutime GG.

Important Note: Do not connect the Acutime antenna to a TSync-PCIe board unless it's for a permanent change, as the TSync-PCIe board will reprogram the antenna to operate with the TSync-PCIe board only. It won't work with the TSAT-PCI/PC-104 board again, without factory re-programming of the antenna.

The TSAT-PCI boards connect to Acutime antenna using 100 ft **CA05-1512-0100** antenna cable (with one end attached to the DB15 timing connector.

TSAT-PCI's connection to GPS antenna

The TSAT connects to the Acutime antenna using the supplied **CA05-1512-0100** (100 ft antenna cable). Refer to: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PCI and PC104 VME for a full drawing of this cable (partial drawing below)



The PCI series timing boards do not have a breakout cable for the DB15 connector. We provide a standard open DB15 connector (and hood) in the TSAT ancillary kit so that a customer can create their own breakout cable, if they need to use any other pins besides the GPS connection.

GPS receiver power
12vdc -12vdc on a TSAT-PCI-U is used to power the GPS receiver only. If they don't have this power available on their PC bus, they should be able to use the card as TPRO with IRIG as the input. ???????

"1997" year issue now occuring with Acutime 2000 antenna's

- (Oct, 2016) Invensys/Schneider Electric reported TSAT boards with an Acutime 2000 antena are now being set to a year of "1997".
- Solution is to replace the Acutime 2000 antennas with Acutime GG antennas. This requires any TSAT board having firmware version prior to 2.11 be returned to the factory for firmware update. Versions prior to v2.11 are not compatible with Acutime GG antennas.
- Refer to the next section down for more details on this antenna swap-out.

TSAT Firmware update needed when using a newer Acutime GG antenna with a TSAT-PCI-U or PCI-66U board (v2.11 added compatibility of Acutime GG)

- Info below pertains to the Acutime-GG antenna (**Our P/N**: E025R-0004-0003 **Trimble P/N** 55238-00) in arena at: https://app.bom.com/items/detail-whereused?item_id=1202842906&version_id=10221223478

Note: The info below is also in the Acutime-GG section of ..\CustomerServiceAssistance.pdf

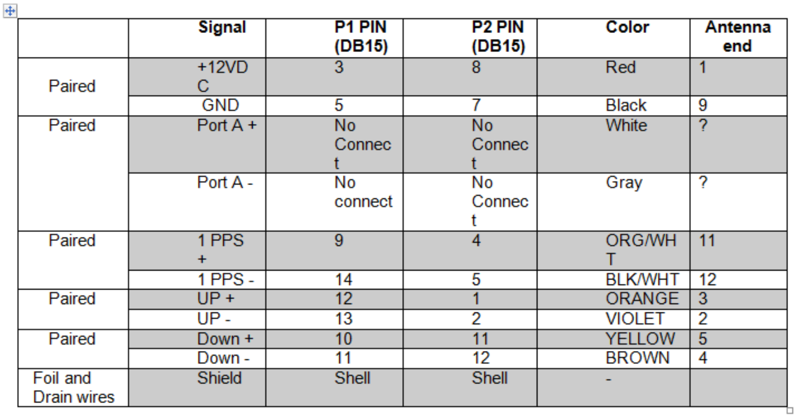
A Firmware upgrade will be likely required for earlier timing boards to be compatible with the newer Acutime-GG antenna. This newer Acutime GG antenna is compatible with:

- **TPRO/TSAT-PCI-U-2 and TPRO/TSAT-PCI-66U** boards with firmware version 2.11 or higher installed (Refer to ECN 3257, July 2013)

Email from Tim T (14 Sept 16)
Sadie,
We could update their cards for them as long as those cards are either a PCI-U2 cards. Invensys has been buying the PCI cards back before Spectracom bought the KSI card line from them. Those cards that were made by KSI cannot be updated.
I would say that if the card has a model number of "1186", we could update it. Anything older than that, we shouldn't do.

Now just because we can do it doesn't mean that we should do it. To update the PCI cards, the boards would have to come back to us. We would need to program a new EEPROM and replace it on the board. Some of the older boards had that IC socketed but most of the boards now have that IC soldered to the board.

Antenna cable (CA05-1512-XXXX) pin-out:



	Signal	P1 PIN (DB15)	P2 PIN (DB15)	Color	Antenna end
Paired	+12VDC	3	8	Red	1
	GND	5	7	Black	9
Paired	Port A +	No Connect	No Connect	White	?
	Port A -	No connect	No Connect	Gray	?
Paired	1 PPS +	9	4	ORG/WHT	11
	1 PPS -	14	5	BLK/WHT	12
Paired	UP +	12	1	ORANGE	3
	UP -	13	2	VIOLET	2
Paired	Down +	10	11	YELLOW	5
	Down -	11	12	BROWN	4
Foil and Drain wires	Shield	Shell	Shell	-	

For editing

	Signal	Wire color	P1 Pin (DB15)	P2 Pin (DB15)	To Acutime Antenna Pin number (12 pin connector)
Paired (power)	+12vdc	Red	3	8	1
	Ground	Black	5	7	9
Paired (Port A)	Port A: Receive -	white	No connection	No connection	6
	Port A: Receive +	Gray			7
Paired (Port A)	Port A: Transmit -	Green	No connection	No connection	8
	Port A: Transmit +	Blue			10
Paired (Port B)	Port B+	B	No connection	No connection	13
	Port B-	Gray			11
Paired (1PPS from antenna to receiver)	1PPS +	Orange/White stripe	9	4	11
	1PPS -	Black/White stripe	14	5	12
Paired (GPS comms	UP +	Orange	12	1	3

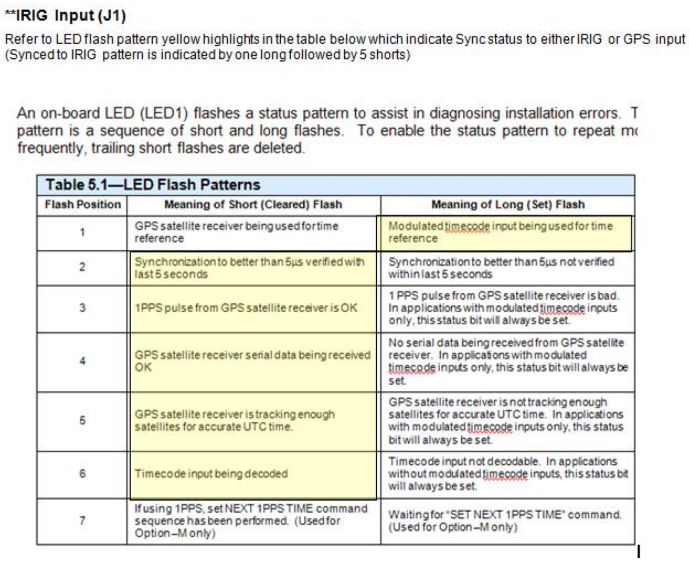
from receiver to antenna)	UP -	Violet	13	2	2
Paired (receiver comms from antenna to receive)	DOWN +	Yellow	10	11	5
	DOWN -	Brown	11	12	4
Foil and Drain wires	Shield	N/A	Shell	Shell	

Desire to use TSAT-PCI series boards in Mobile mode
12/3/10 per Tim Tetreault- The TSAT-PCI timing boards themselves require no change from the current design to operate in a mobile environment. However, the Acutime antennas for the TSAT boards need to be programmed differently at the factory to support this mode.

If a customer desires to obtain this capability with a TSAT board, a new P/N for the Acutime antenna will be required to be created and the programming of the antenna changed to support the mobile mode operation only. The current programming of the antenna for the TSAT board does not support mobile mode operation.

LED flash pattern

Refer to LED flash pattern yellow highlights in the table below which indicate Sync status to either IRIG or GPS input (Synced to IRIG pattern is indicated by one long followed by 5 shorts)



****IRIG Input (J1)**

Refer to LED flash pattern yellow highlights in the table below which indicate Sync status to either IRIG or GPS input (Synced to IRIG pattern is indicated by one long followed by 5 shorts)

An on-board LED (LED1) flashes a status pattern to assist in diagnosing installation errors. T pattern is a sequence of short and long flashes. To enable the status pattern to repeat m frequently, trailing short flashes are deleted.

Table 5.1—LED Flash Patterns

Flash Position	Meaning of Short (Cleared) Flash	Meaning of Long (Set) Flash
1	GPS satellite receiver being used for time reference	Modulated timecode input being used for time reference
2	Synchronization to better than 5µs verified with last 5 seconds	Synchronization to better than 5µs not verified within last 5 seconds
3	1PPS pulse from GPS satellite receiver is OK	1 PPS pulse from GPS satellite receiver is bad. In applications with modulated timecode inputs only, this status bit will always be set.
4	GPS satellite receiver serial data being received OK	No serial data being received from GPS satellite receiver. In applications with modulated timecode inputs only, this status bit will always be set.
5	GPS satellite receiver is tracking enough satellites for accurate UTC time.	GPS satellite receiver is not tracking enough satellites for accurate UTC time. In applications with modulated timecode inputs only, this status bit will always be set.
6	Timecode input being decoded	Timecode input not decodable. In applications without modulated timecode inputs, this status bit will always be set.
7	If using 1PPS, set NEXT 1PPS TIME command sequence has been performed. (Used for Option–M only)	Waiting for "SET NEXT 1PPS TIME" command. (Used for Option–M only)

Oscillator stability/Estimated timing accuracies with reference removed.

Stability

- Disciplined to Timecode: 2 x 10–7
- Undisciplined Freewheel Drift: 1 x 10–6

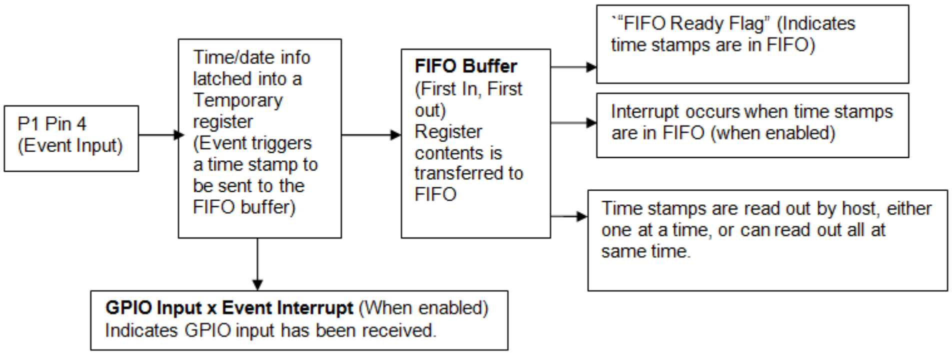
To calculate the estimated amount of Freewheel drift, convert the time to seconds and multiply by 1.5E-6. Examples (where 1 hr = 3600sec):

Amount of drift per hour = 3600 * 1.5E-6 = 0.0054 seconds per hour (or 5.4 milliseconds) of estimated drift

Amount of drift per second= 0.0054sec / 3600 (seconds) = .0000001 seconds per second (or .1 microsecond) of estimated drift

Amount of drift per day= 0.0054sec x 24 (hours) = .0123 seconds per day (or 12 milliseconds) of estimated drift

**TimeTag (Timetagging/time stamping/Timestamping)


P1 Pin 4 (Event Input)
Time/date info latched into a Temporary register (Event triggers a time stamp to be sent to the FIFO buffer)
FIFO Buffer (First In, First out) Register contents is transferred to FIFO
`"FIFO Ready Flag" (Indicates time stamps are in FIFO)
Interrupt occurs when time stamps are in FIFO (when enabled)
Time stamps are read out by host, either one at a time, or can read out all at same time.
GPIO Input x Event Interrupt (When enabled) Indicates GPIO input has been received.

Notes about Timetag (for both PC104 and PCI series timing boards)

- Time Tag for both PC104 and PCI series timing boards is the same.
- When the rising edge of the Time Tag Input (P1 Pin 4) occurs, the clock time is latched into a temporary register, and this register is then loaded into the FIFO buffer.
- Since the clock time is latched when the rising edge occurs, the Time Tag data will be correct regardless of how long it takes to read the FIFO.
- The same FIFO buffer is also use for driver calls (not HW_GetTime), including the getDate call.
- An interrupt, if enabled, is generated on the PCI bus to indicate a time stamp has occured.
- Unlike PMC and cPCI boards, PC104 and PCI series boards time stamps are not stored one at a time in the FPGA. Instead, they are stored in a FIFO buffer (First In / First Out).
- There may be a delay of up to 200 µS before time tag data is available in the FIFO
- Erratic operation may result if the rate of the time tags exceeds 1000 per second.
- The operator can *disable* external events by jumpering JP6 Pin '2-3' unless there is a source of external event pulses connected to P1 Pin 4. To *enable* external events jumper JP6 Pin 1–2.
- The user's software establishes that a time tag has occurred in one of two ways: either by examining the least significant bit of the Status Register (FIFO Ready flag) to determine if it is a zero, or by waiting for an interrupt. Since the clock time is latched when the rising edge occurs, the Time Tag data will be correct regardless of how long it takes to read the FIFO.
- Time tags are read by first sending a command to the board, then reading ten 32-bit words from a first-in-first-out (FIFO) register.
- The user's software must wait a short amount of time after sending each command. This requirement is necessary because the on-board processor places a higher priority on maintaining the time than on processing commands. The actual amount of time needed to process a command depends on when the command was received relative to the on-board time. Entries in Chapter Six of the user manual specify the amount of time needed to assure that the command was received. There is no restriction on reading from the board.

Time Tag input

- TimeTag input is on **pin 4** of the Timing connector (P1) on edge of the board.

- Time Tag occurs on the **Rising** edge of the input signal.

Table 2.3—Time Tag Input Specifications	
Input Voltage	-0.5V min, +0.8V max for logic 0 +2.0V min, +5.5 max for logic 1 Tags rising edge
Input Current	<5 µA for logic 0 <5 µA for logic 1
Rise/Fall Time	500 nS max
Repetition Rate	1000 events per second max
Timing Resolution	1 µS

Q. I am working on an IRIG 106 CH 10 standard recording time tagging of data messages using spectracom PC 104 timing card.   IRIG 106 CH 10 standard specifies 48 bit relative time counter to be used for time tagging this data type (MIL STD 1553).Does this card have a register with this information which I can pull to time stamp?

KW reply on 10/28/11 based on info from Tim Tetreault
The Spectracom PC104 timing board uses a FIFO buffer to read the time tag register. The FIFO buffer consists of 16 bits, but the time tag register itself is 48 bits long. The time tag is in a BCD format with 1 microsecond resolution. Please refer to the following table from the PC104 user manual for the format of the FIFO message.

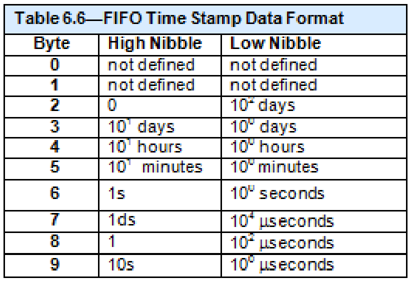


Table 6.6—FIFO Time Stamp Data Format

Byte	High Nibble	Low Nibble
0	not defined	not defined
1	not defined	not defined
2	0	10^2 days
3	10^1 days	10^0 days
4	10^1 hours	10^0 hours
5	10^1 minutes	10^0 minutes
6	1s	10^0 seconds
7	1ds	10^4 µseconds
8	1	10^2 µseconds
9	10s	10^0 µseconds

Kernel lockup issues when performing both Time Tags and getDate calls

Our software queries the time at a rate of approximately 1-2 times per second. The software calls are:

Q. Email from Bob Clemmons with Aeroflex (8/24/12)
TPRO_peek (offsex 0x04) to query the sync status
TRO_SetPropDelayCorr(m_boardHandle, NULL) // not really needed since there is only one user of the board, but just cuz the example has this
TPRO_GetTime
TPRO_GetDate

Our “actual” software makes the Wait_Event call, but we are seeing the symptoms in either case, so I’ve simplified the logic for this test.

What we observe is that, when the Time Tag Input receives pulses, we occasionally (on the order of 3 times in 5 minutes) get invalid time/date from TPRO_GetTime, TPRO_GetDate.

Typically, the year is returned as 4 and the month is returned as 22 or 23. The day is usually 0.

If we disconnect the TTL signal to the Time Tag Input, we do not experience this issue.

The status register is typically 0x17, which indicates that the PCI bus interrupts are disabled (power on default, according to the manual).

If we run this for a long time with the Time Tag Input receiving pulses, we see the kernel lock up after 1-3 hours.   If the Time Tag Input is not receiving pulses, then the system runs without lockup (we monitored it for 12+ hours without a problem).

Dave Lorah's response, after working with Tim Tetreault (9/11/12)
I was discussing the issue with our engineer this morning and he had some information for me. It appears the board is having a problem properly processing the data when reading the FIFO the way the software commands are used. The problem occurs if the date is read at the same time an event is input to the card. The date reading and the event capture both go through the FIFO. If an event occurs at the time the date read is being processed by the micro, the event data takes precedence and the event data is actually read from the FIFO instead of the event time capture data.

This is a result of the card architecture design and cannot be changed. The resolution would be to read the date at the same time an event is captured. This is how the time and date are typically used.

I would recommend changing the code to read the date only when the event time is read to avoid any FIFO misreads.

Another alternative would be to use a TPRO-PCIe type board if a PCIe slot is available in the system. The TPRO or TSAT- PCIe board does not have this limitation of the FIFO. However, the driver and software commands are different.

We are not exactly sure what the application of your system is. If you can elaborate maybe we can recommend a better way to use the -66U board commands. We would be happy to give advice if we know the purpose of the application.

Outputs

**IRIG output (J2)

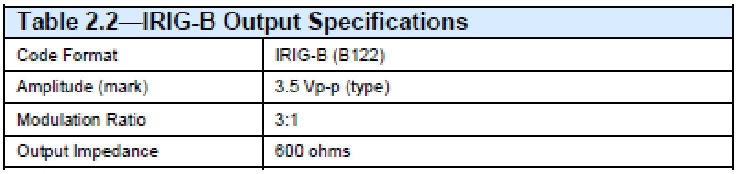


Table 2.2—IRIG-B Output Specifications	
Code Format	IRIG-B (B122)
Amplitude (mark)	3.5 Vp-p (type)
Modulation Ratio	3:1
Output Impedance	600 ohms

Only available output formats:

- **IRIG B122** (IRIG B AM, no year) OR
- **IRIG B022** (with Option "**-DCLOBNC**" added/enabled. Additional info on this available Option further below)

Note: The TPRO/TSAT-PCI boards cannot output other IRIG signals other than B122 or B022 (such as B000, for instance). These legacy timing boards are limited to either B122 or B022. For other IRIG formats, refer to the TSync-PCIe boards, which allow input and output of many different IRIG strings, such as B000.

Year information: Not provided in either B122 or B022 outputs. So the PCI boards don't output year info in IRIG out.

- This means that its required to set the year in the device receiving either B122 or B022 (if Option **DCLOBNC**" is added/enabled)

IRIG output modulation: The IRIG AM output is always AM. It cannot be configured for DCLS output

IRIG output amplitude: The IRIG AM output is about 3.5vp-p into 600 ohms.

The output amplitude is not adjustable on the timing board itself (it's a fixed value). To decrease the IRIG output amplitude, decrease the end termination value.
From the manual, the typical output amplitude is 3.5vpp into 600 ohms. An email from a customer indicated:

I just tested the output signal. However, one problem I saw is it's about 4.8Vp-p when a 600 ohms resistor is added as a termination. It's not 3.5Vp-p as expected. Please let me know why it has so big difference. I also tried to reduce the resistance value. When it's 75 Ohms, the PCI card output amplitude is about 3.2Vp-p. Please let me know if you see any problem with a 75 Ohms termination.

my reply:
Regarding your email, I spoke to one of our engineers who indicated the 3.5Vp-p spec mentioned in the manual is only a typical output amplitude spec (as indicated by "typ" displayed after the value). We do not specify a min or max IRIG output amplitude value, as this actual value will vary from one timing board to the next. We just provide the typical output value into 600 ohms. The actual amplitude at 600 ohms may be lower or higher than the typical spec.

The engineer also stated that if a lower amplitude signal is desired, you can just decrease the termination value (such as going to 75 ohms, instead of 600 ohms) until you find the desired amplitude of the output signal. He said there is no problem leaving the 75 ohm termination at the end of the IRIG distribution cable to provide the 3.2vpp amplitude your equipment desires to see.

The IRIG output is always present:
There is no way to enable/disable the IRIG output of the TPRO-PCI timing board. The IRIG output is always present, as long as the timing board is up and running. The user can enable/disable IRIG input synchronization, but there is no way to disable the IRIG output signal from the timing board.

J2 always outputs IRIG B122 only (IRIG B AM 1000 Hz, BCD only- no control field or year value). There is no configuration available in the drivers or the Control Panel for IRIG out.

The TPRO boards do not distribute year information on the IRIG output. The year must be set by the user when using IRIG input. To set the year automatically, use GPS. However, the TSYNC-PCIe boards can be programmed to both read and distribute year information on the IRIG output.

Note: The motherboard needs to supply all the voltages necessary to power the board. The board requires +5V, +12V and -12V to operate. If the -12V is missing the IRIG output will not work. If the =12 v and/or -12V is missing the IRIG output will be distorted.

****IRIG IEEE-1344 extensions

- Refer to Salesforce case 10054

PCI series boards cannot output IEEE-1344 extensions.   Architecture of the boards prohibits this capability from even being added to these boards as a special mod. Need to use TSync-PCIe boards if the extensions are required in the IRIG input/output.

**Option "–DCLOBNC" (IRIG DCLS output)

- For more info, refer to: Options

Provides IRIG DCLS output (does not provide IRIG DCLS input capability). The standard output is IRIG B122. With this option added (earlier TPRO/TSAT-PCI series boards) or with the option enabled (newer TPRO/TSAT-PCI-U-2 and 66U series boards), the signal becomes IRIG B022.

Email from Tim T (8/2/12)
The-DCLOBNC option on the TSAT/TPRO-PCI boards is a DCLS output. The boards just can't read DCLS. If they setup the jumper and DIP switch as per the manual, they can source DCLS.

**Heartbeat output

- **For more info, refer to**: Options

Output pin: The Heartbeat output is on Pin 6 of the DB15 timing connector.

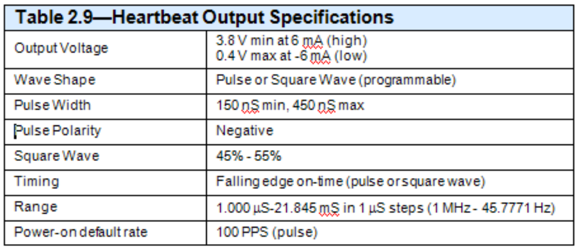


Table 2.9—Heartbeat Output Specifications	
Output Voltage	3.8 V min at 6 mA (high) 0.4 V max at -6 mA (low)
Wave Shape	Pulse or Square Wave (programmable)
Pulse Width	150 nS min, 450 nS max
Pulse Polarity	Negative
Square Wave	45% - 55%
Timing	Falling edge on-time (pulse or square wave)
Range	1.000 µS-21.845 mS in 1 µS steps (1 MHz - 45.7771 Hz)
Power-on default rate	100 PPS (pulse)

Default heartbeat output: The power-on default is 100 PPS, pulse mode.

Allowable Heartbeat frequency ranges:

- **Without HB1PPS option enabled ("Standard" default configuration):** 45.7771Hz to 1.000MHZ
- **With HB1PPS option enabled:** 0.152593Hz to 500Hz (in exact multiples of 1ms)

Programming the heartbeat output frequency

Note: The POKE API call is used to set the register (PEEK API call is used to read the register).

- **Refer to Section 7.18** of the PCI user manual for **heartbeat frequencies above 1 Hz**
- **Refer to Section 7.19** for **heartbeat frequencies of 1 Hz or less**

Pulse or square wave output

The heartbeat output can be programmed for a square wave or a pulse, and can be programmed either to start immediately, or at the beginning of the next cycle. Send one of the following commands to command port:

```
0x00e5          ; Pulse mode, starts at beginning of next cycle
0x00e6          ; Pulse mode, starts immediately
0x00e7          ; Square wave, starts at beginning of next cycle
0x00e8          ; Square wave, starts immediately
```

Need to wait 8 seconds after setting heartbeat settings

Email from Tim T

Whenever you change the heartbeat to the PCI board, you should wait 8 seconds then retest for SYNC before trying to set the heartbeat again. When the board is using an IRIG as its time source, extra time is required to SYNC after the heartbeat is set due to a software filter and decoding the IRIG signal.

Desire to stop the Heartbeat output

- Refer to Salesforce case 118344

**HB1PPS Option

- Desire for 1PPS (or slower) signal on the Heartbeat output (otherwise heartbeat can't go to this low of a frequency)
- **Note:** Refer to "HB1PPS" in the "Options" section above

Email from Tim T: To get a 1Hz heartbeat, they need to change the DIP switches to configure the board for the –HP1PPS option.

Email to customer: In order to provide a 1Hz heartbeat output, you just need to enable the "–HB1PPS" option. This option allows for the 1Hz frequency to be available on the Heartbeat output.

The option is enabled using DIP switches/jumpers on the board. This option is described in Section 7.19, page 7-15 of the TPRO/TSAT manual (I have attached a copy of this manual, for your reference). Section 4, page 4.1 of the manual shows the location and switch positions used to enable each of the available options.

The first table on page 4.1 is for TPRO board options and the second table is for TSAT board options. Both tables contain a row for the HB1PPS option. Follow the "-HB1PPS" row in the applicable table to determine the switch and jumper settings for 1PPS output.

**FXA Option

- RS-422 Heartbeat output.
- **Refer to**: -FXA

**FXB Option

- RS-422 Heartbeat output.
- Refer to: -FXB

**Match time (Start/Stop)

- Refer to Section 5 of the TSync-PCIe driver guide for more info on "Match Time, including the series of API calls to perform, in order for Match time to occur: (1219-5001-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833277&version_id=10212776628&orb_msg_Single_Search_p=1&redirect_Seqno=6611466902

Match time output is on:

- **PMC boards:** Pin **3** of the DB25 **Timing** connector.
- **cPCI boards:** Pin **6** of the DB15 **Timing** connector

Jumper setting for Match output (jumpers located on the edge of the board)

- Jumper 15 to 16 connects the **Match** output from the on-board circuitry to the **Timing** connector. This jumper needs to be installed for the signal to be present on the Timing connector.
- Refer to Section 2 (Jumper settings) in the user manual for more information.

TSYNC_HW_SetMatchTimeHi:"communication error (TSYNC_COMM_ERR)

- Eric Girard had a customer report this message with the Windows driver version 2.4.1 installed.

My reply to Eric, based on working with Tim Tetreault (6 May 15)
Tim believes that this message is likely due to either the Match Time function not being enabled after the board last booted-up, or no Matches have yet occurred since power-up. He recommended your customer verify if they have enabled the Match Time function yet, since the Tsync board was last powered-up. By default, it is disabled in the TSync board. It needs to be enabled each time the TSync board is booted-up, if they wish to use this available function.

The API call/example program to determine if Match Time has been enabled is **GO_GetMatchEnable 0.** If it responds with False (Match Time not enabled), it can be enabled with the **GO_SetMatchEnable 0** API call/example program.

I recommend your customer first enable the Match Time function if they wish to use it. Then, have them repeat the HW_SetMatchTimeHi command to see if the communication error message is still returned.

FAQs about Match Time

Q. I noticed that after I use setMatchTimeHi or setMatchTimeLo, and then read the info back using getMatchTimeHi or getMatchTimeLo, the value for the year returned by the API call is always "0001". Is this a **bug**?
A (Reply from Dave Sohn on 9/26/11) Match time does not use the year information. Time matches can only be set within one year of the current time. The year information read back is not relevant and should be ignored.

**Option –LOR1 output

- Refer to (in “Options” section further above): -FXB

FIRMWARE/SOFTWARE/DRIVERS (Windows Linux VxWorks LynxOS)

Firmware/ Driver versions

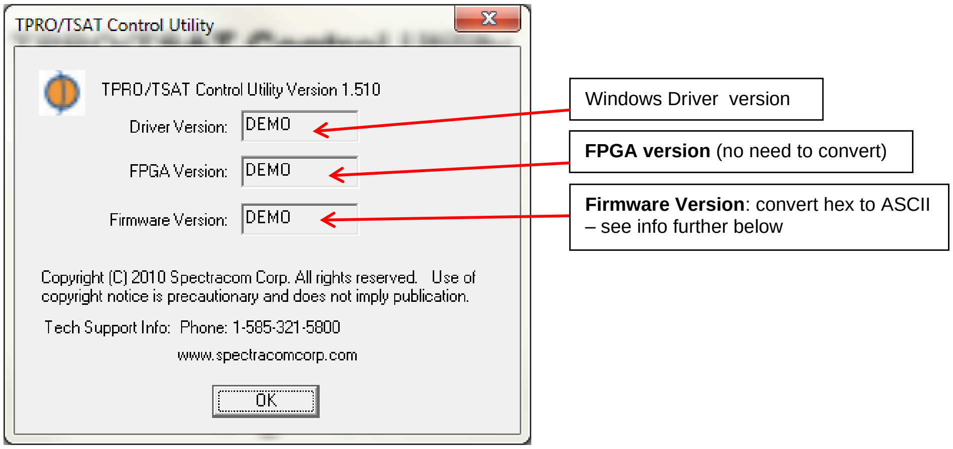
Firmware info

Refer to: PSB, PSP software updates\TPRO-TSAT boards\board firmware

Determining current firmware version installed

A) Use the Windows Control Utility, if board is installed in Windows PC

- **Help** -> **About** menu



TPRO/TSAT Control Utility

TPRO/TSAT Control Utility Version 1.510

Driver Version: DEMO

FPGA Version: DEMO

Firmware Version: DEMO

Copyright (C) 2010 Spectracom Corp. All rights reserved. Use of copyright notice is precautionary and does not imply publication.

Tech Support Info: Phone: 1-585-321-5800

www.spectracomcorp.com

Windows Driver version

FPGA version (no need to convert)

Firmware Version: convert hex to ASCII – see info further below

B) Use the TPRO_GetFirmware API call

- This call will respond with a value such as "**02-11-10-23**".
- The first two sets of numbers (**"02**" and "**11**" in the example above) are the board's firmware version (no conversions necessary). In this example, the firmware version is 2.11.
- The second two sets of number (**"10"** and "**23**" in the example above) are related to the EEPROM and are dependent on which DIP switches for Options are on and which are off.

Specific notes about firmware versions

- Version 2.11 (ECN 3257, July 2013) added support for compatibility with newer Trimble Acutime GG Smart GNSS antenna. This software is also backwards compatible with earlier Trimble Acutime antennas.
- Name change from "PCI-33U" to "PCI-U-2" occurred on ECN 2316, 9 Jan 2009

C) FPGA version

- Same as the **FS_GetVersion** Example program/API call

- ➢ No need to convert the value. Tim Tetreault says this field probably won't work correctly on newer versions because the Control Utility hasn't been updated to reflect the newer FPGA versions *(such as 2.2.2 or 2.2.3 for instance).
- ➢ Use the FS_GetVersion 0 Example program/API to ascertain this value.

D) Firmware version

- ➢ Same as the **LS_GetVersion** Example program/API call
- ➢ Reports the version of the KTS software sub-assembly
- ➢ Convert the Hex to ASCII

To convert the Firmware version, refer to a:

- "Hex to an ASCII" table, such as http://www.asciitable.com/
- or to a "Hex to ASCII converter", such as http://www.dolcevie.com/js/converter.html (Use colons between each set of two numbers)

- ➢ Where "2E" is a decimal point

In this example:
32" Hex is "2" in ASCII
"2E" Hex is a "." in ASCII
"32" Hex is "2" in ASCII
"31" Hex is "1" in ASCII

Specific conversions:
33-2E-30-34 (33:2E:30:34) = version 3.0.4

- ➢ **Using the API calls/example programs**

  FS_GetVersion (API calls for version info)

TPRO/TSAT Drivers

All Drivers (not driver specific)

PCI series drivers:

- **Window driver (1186-SD07-xxxx)** where xxxx is the version of the driver (search PLM for “1186-SD07”)
- **Linux driver (1186-SD09-xxxx)** where xxxx is the version of the driver (search PLM for “1186-SD09”)

PCI series Driver guides:

- **Shortcut to Windows Driver Guide (1186-5002-0050) in Arena:** https://app.bom.com/items/detail-spec?item_id=1202833263&version_id=10212776488&orb_msg_Single_Search_p=1&redirect_Seqno=7546148528
- **Shortcut to Linux Driver Guide (1186-5003-0050) in Arena:** https://app.bom.com/items/detail-spec?item_id=1202833264&version_id=10212776498&orb_msg_Single_Search_p=1&redirect_Seqno=7546145786

Determining current driver version installed

- Use the Windows Control Utility, if board is installed in Windows PC
- Use the **TPRO_GetDriver** API call

Release Notes for Driver versions

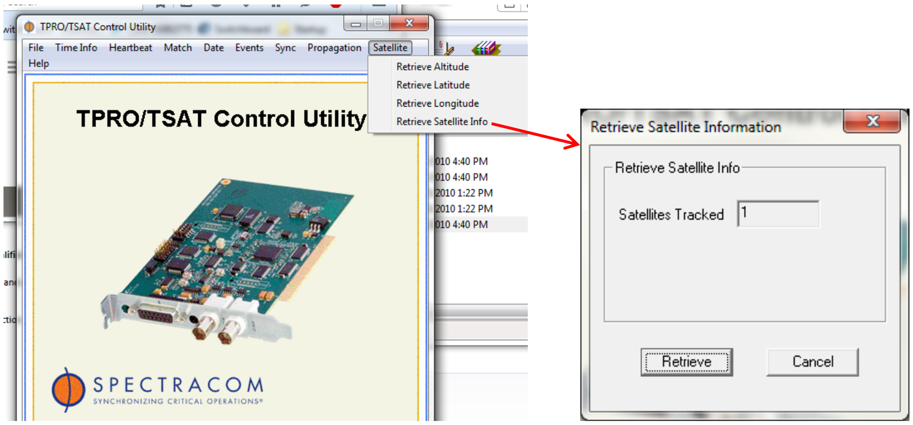
- **Refer to: PSB, PSP software updates\TPRO-TSAT boards\Driver updates**

GPS info for TSAT boards (not applicable to TPRO boards)

- Refer the TPRO/TSAT-PCI-U-2/66U user guide/driver guide for details on all of the available “TPRO” calls.

A) GPS info via Windows Control Utility

Satellite -> **Retrieve Satellite info** menu



Satellites Tracked 1

B) TPRO_GetSatInfo example program/API call

- retrieves the number of satellites tracked from the TSAT device.

**Known issue associated with GPS/Acutime antenna

- Refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

C) TSAT-PCI-U-2 boards have time-reads “freezing-up”

- Refer to SRs 6653/7116 (for Invensys)
- Apparently due to using an earlier Model Acutime antenna

We have several customers reporting time “freezing” on their systems. I see evidence of this in the log files from our timekeeper application from those sites; time read from the card doesn’t change for multiple reads. In one case, they have to reboot to ‘unfreeze” the time. In another case, the customer stripped our software off the system in an attempt to isolate the issue. They’re using an HPZ420 running Windows 7. They installed a TSAT/TRPRO PCI-U-2 GPS card with firmware version 02-11-40-25, and loaded GPS driver version 2.3. They’re using an Acutime GG antenna. And they were able to see the time “freeze” using the TPRO Control Utility.

D) ***TSAT-PCI-U-2 (33U) and TSAT-PCI-66U boards reporting 0 satellites being tracked (when tracking more than 9 satellites)

(1-25-10 KW) There is a software bug in the PCI boards where it cannot report more than 9 satellites. Any double-digit satellite count will have the first digit excluded in the logs, i.e., if the card is tracking 10 satellites, it will be reported as 0 satellites and give a false “insufficient satellites” error. The board will be functioning properly, but it will not report the correct number of satellites if it is tracking more than 9.

At this time, there is no scheduled fix, though the reason this is happening is known. Because of this issue, a customer should not use the number of satellites tracked to indicate Sync Status of the board. One way to work around the bug is to use the “syncStatus” API call. This command will always tell you if the board is in sync or not.

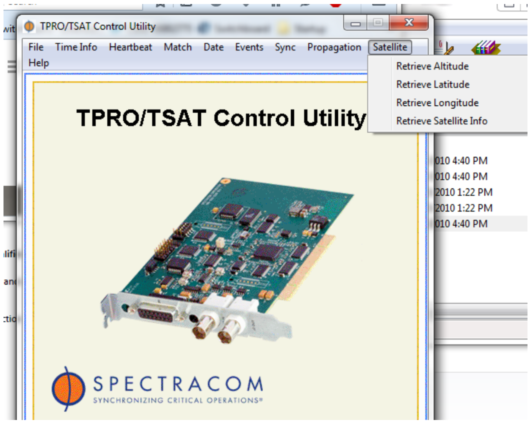
Update Note (12/01/10 KW): This issue was actually a driver issue (instead of a firmware issue with the boards, as was initially thought). It is being fixed in the driver versions about to be released around 12/5/10 (Resolved in ECN 2552 for Windows driver version 2.3.0 and Linux driver version 3.2.0.

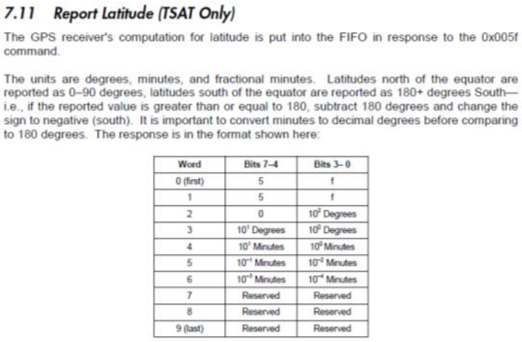
TPRO_GetLatitude and TPRO_GetLongitude (TSAT board only)

- Refer the TPRO/TSAT-PCI-U-2/66U user guide for details (info excerpted below)
- Unlike the newer "TSync" calls for the TSync board, the legacy TPRO_GetLatitude and TPRO_GetLongitude calls do not report "signs" (+/- , N/S, E/W).

Lat and long can be obtained by Windows Control utility (Windows driver only)

Satellite -> **Retrieve Satellite info** menu



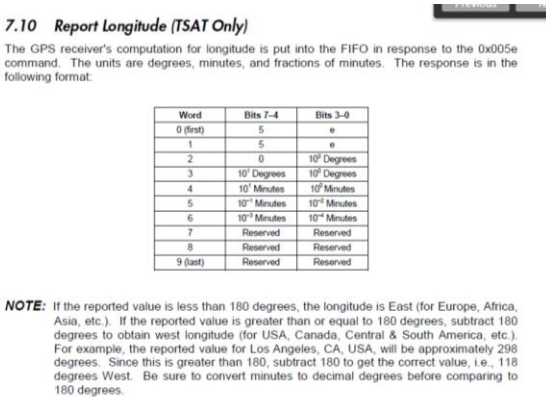


7.11 Report Latitude (TSAT Only)

The GPS receiver's computation for latitude is put into the FIFO in response to the 0x005f command.

The units are degrees, minutes, and fractional minutes. Latitudes north of the equator are reported as 0–90 degrees, latitudes south of the equator are reported as 180+ degrees South—i.e., if the reported value is greater than or equal to 180, subtract 180 degrees and change the sign to negative (south). It is important to convert minutes to decimal degrees before comparing to 180 degrees. The response is in the format shown here:

Word	Bits 7–4	Bits 3– 0
0 (first)	5	f
1	5	f
2	0	10^2 Degrees
3	10^1 Degrees	10^0 Degrees
4	10^1 Minutes	10^0 Minutes
5	10^{-1} Minutes	10^{-2} Minutes
6	10^{-3} Minutes	10^{-4} Minutes
7	Reserved	Reserved
8	Reserved	Reserved
9 (last)	Reserved	Reserved



7.10 Report Longitude (TSAT Only)

The GPS receiver's computation for longitude is put into the FIFO in response to the 0x005e command. The units are degrees, minutes, and fractions of minutes. The response is in the following format:

Word	Bits 7–4	Bits 3–0
0 (first)	5	e
1	5	e
2	0	10^2 Degrees
3	10^1 Degrees	10^0 Degrees
4	10^1 Minutes	10^0 Minutes
5	10^{-1} Minutes	10^{-2} Minutes
6	10^{-3} Minutes	10^{-4} Minutes
7	Reserved	Reserved
8	Reserved	Reserved
9 (last)	Reserved	Reserved

NOTE: If the reported value is less than 180 degrees, the longitude is East (for Europe, Africa, Asia, etc.). If the reported value is greater than or equal to 180 degrees, subtract 180 degrees to obtain west longitude (for USA, Canada, Central & South America, etc.). For example, the reported value for Los Angeles, CA, USA, will be approximately 298 degrees. Since this is greater than 180, subtract 180 to get the correct value, i.e., 118 degrees West. Be sure to convert minutes to decimal degrees before comparing to 180 degrees.

Time reads

- Example program/API call to read the current time : **getTime** **(**Not like **HW_GetTime** or like **CS_GetTime** like with the TSync series timing boards)

Issues associated with time reads

A) TSAT-PCI-U-2 boards reported by Invensys/Schneider electic as having time-reads "freezing-up"

- Apparently due to using an older Model Acutime antenna with the TSAT-PCI-U-2 boards
- Refer to (in this doc) "issues wth GPS): **Known issue associated with GPS/Acutime antennas

**Interrupt generation for PCI board

- Interrupts can be generated with three different conditions (listed below):
- When the Heartbeat output occurs.
- With Match time (interrupt occurs at the "Start Time").
- When data is available in the FIFO buffer.

From the TPRO-PCI-66U manual

7.15 Interrupt Control Port
The board can generate interrupts when the FIFO has data, when each heartbeat occurs, and/or when the commanded Start Time occurs. Each of these is enabled separately via the Interrupt Control Port, and any, all, or none can be enabled simultaneously.

- By default, all interrupts are disabled (until the user's application software enables them after each boot-up)

- Three bits of the “Interrupt Control Port” defines when the interrupts will occur.
- Any combination of types of Interrupts can be enabled, or all can be enabled at the same time.
- Only one interrupt is generated, no matter the condition that generated it. The user’s application software has to interrogate the board to find out which of the three conditions caused the interrupt to be generated.

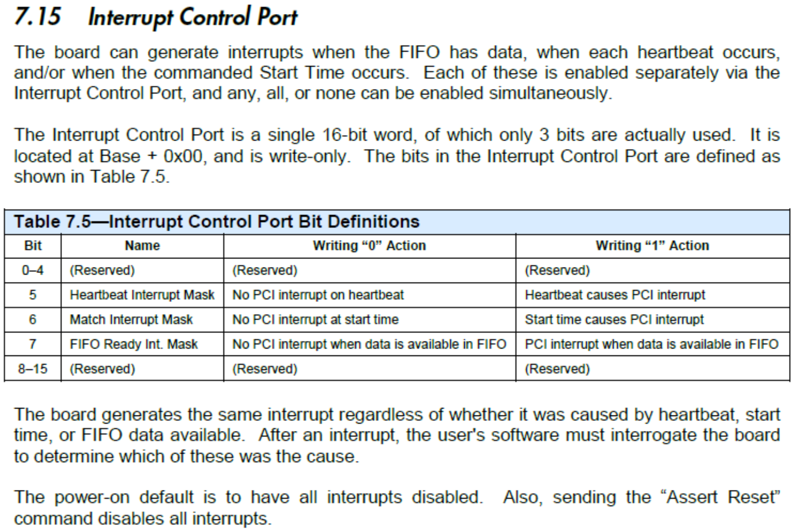
Interrupt Request level (IRQ)

From Section 4.2 of the PCI user’s manual: The host computer's PCI BIOS software configures the interrupt request (IRQ) level

Interrupt Control Port

(From Section 7.15 of the PCI user’s manual)

- All interrupts are disabled by default (after each boot-up)
- Interrupt Control Port Allows each interrupt type to be individually enabled/disabled



7.15 Interrupt Control Port

The board can generate interrupts when the FIFO has data, when each heartbeat occurs, and/or when the commanded Start Time occurs. Each of these is enabled separately via the Interrupt Control Port, and any, all, or none can be enabled simultaneously.

The Interrupt Control Port is a single 16-bit word, of which only 3 bits are actually used. It is located at Base + 0x00, and is write-only. The bits in the Interrupt Control Port are defined as shown in Table 7.5.

Table 7.5—Interrupt Control Port Bit Definitions

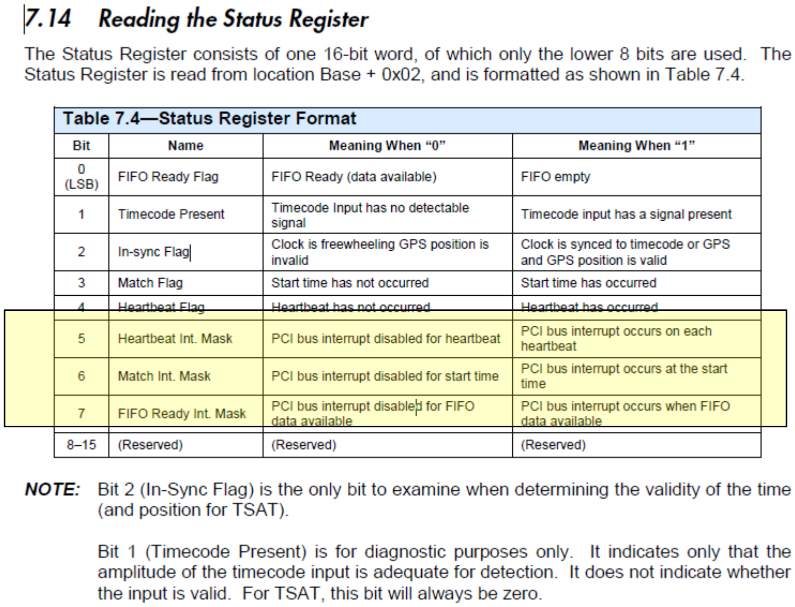
Bit	Name	Writing “0” Action	Writing “1” Action
0–4	(Reserved)	(Reserved)	(Reserved)
5	Heartbeat Interrupt Mask	No PCI interrupt on heartbeat	Heartbeat causes PCI interrupt
6	Match Interrupt Mask	No PCI interrupt at start time	Start time causes PCI interrupt
7	FIFO Ready Int. Mask	No PCI interrupt when data is available in FIFO	PCI interrupt when data is available in FIFO
8–15	(Reserved)	(Reserved)	(Reserved)

The board generates the same interrupt regardless of whether it was caused by heartbeat, start time, or FIFO data available. After an interrupt, the user's software must interrogate the board to determine which of these was the cause.

The power-on default is to have all interrupts disabled. Also, sending the “Assert Reset” command disables all interrupts.

Reading the Status register

(From Section 7.14 of the PCI user’s manual)



7.14 Reading the Status Register

The Status Register consists of one 16-bit word, of which only the lower 8 bits are used. The Status Register is read from location Base + 0x02, and is formatted as shown in Table 7.4.

Table 7.4—Status Register Format

Bit	Name	Meaning When "0"	Meaning When "1"
0 (LSB)	FIFO Ready Flag	FIFO Ready (data available)	FIFO empty
1	Timecode Present	Timecode Input has no detectable signal	Timecode input has a signal present
2	In-sync Flag	Clock is freewheeling GPS position is invalid	Clock is synced to timecode or GPS and GPS position is valid
3	Match Flag	Start time has not occurred	Start time has occurred
4	Heartbeat Flag	Heartbeat has not occurred	Heartbeat has occurred
5	Heartbeat Int. Mask	PCI bus interrupt disabled for heartbeat	PCI bus interrupt occurs on each heartbeat
6	Match Int. Mask	PCI bus interrupt disabled for start time	PCI bus interrupt occurs at the start time
7	FIFO Ready Int. Mask	PCI bus interrupt disabled for FIFO data available	PCI bus interrupt occurs when FIFO data available
8–15	(Reserved)	(Reserved)	(Reserved)

NOTE: Bit 2 (In-Sync Flag) is the only bit to examine when determining the validity of the time (and position for TSAT).

Bit 1 (Timecode Present) is for diagnostic purposes only. It indicates only that the amplitude of the timecode input is adequate for detection. It does not indicate whether the input is valid. For TSAT, this bit will always be zero.

Generating a heartbeat interrupt

- Refer to: PCI Heartbeat output in this document for information on configuring the heartbeat output
- Enable the heartbeat interrupt (refer to the Interrupt Control Port information directly above).

Email Keith sent to a customer (5 Feb 2013)
In general:
For your information, the Spectracom TPRO/TSAT-PCI-U (and U-2) timing board has three available types of interrupt generation. The three reasons for interrupt generation include Heartbeat, Match (Time Tagging) and FIFO ready (indicating the FIFO buffer is ready to be read). Attached you should find a copy of the TPRO/TSAT-PCI-U-2 user's manual. Refer to page 7-12 (Section 7.15) of this manual for more information on the Interrupt generation. From this section of the manual:

The timing board can generate interrupts when the FIFO buffer has data, when each heartbeat occurs, and/or when the commanded Start Time occurs. Each of these is enabled separately via the Interrupt Control Port, and any, all, or none can be enabled simultaneously.

All three of these reasons that interrupts can be generated for will generate one interrupt. When an interrupt occurs, the Status Register can be read to determine which of these three reasons was the one that triggered the interrupt to occur. Refer to page 7-11 (Section 7.14) of the attached user manual for more information on reading the Status Register.

Specific to the Heartbeat output interrupt

Note: The information below needs to be performed each time the timing board powers-up (these configurations don't persist through reboots/power cycles).

To use the Heartbeat interrupt, the Heartbeat output needs to be configured to be at the same frequency as the desired interrupt interval. If this rate is above 1Hz, the HB1PPS option does not need to be enabled. Section 7.18 (page 7-14) of the attached user manual discusses how to configure the Heartbeat (and therefore also the interrupt) rate. However, if the desired interrupt rate is 1 Hz or lower, the HB1PPS Option needs to be enabled/installed.

With the latest TPRO/TSAT-PCI-U-2 and TPRO/TSAT-PCI- 66U timing boards, the HB1PPS Option can be enabled in the field. With the earlier TPRO/TSAT-PCI-U and TPRO/TSAT-PCI timing boards, this Option needs to be purchased and installed at the factory.   Refer to Section 9.1.2, page 9-1 of the user manual to configure the HB1PPS Option) Section 7.19 of the user manual discusses how to configure the Heartbeat output with frequencies of 1Hz or slower.

With the Heartbeat output configured for the desired interrupt frequency, the heartbeat interrupt needs to be enabled (after each boot-up, all interrupt types are disabled). This is controlled by the Interrupt Control Port. Refer to Section 7.15 (page 7-12) of the user manual for information on enabling/disabling the Heartbeat Interrupt.

Interrupts are enabled by disabling the Interrupt Mask. Conversely, Interrupts are disabled by enabling the interrupt mask. Per Table 7-5 on this page of the manual. Bit 5 needs to be set to a “1” to cause the Heartbeat PCI interrupt to start occurring.

If desired, the Status Register can be read at each interrupt to find out what event caused the interrupt to be generated (this allows the same interrupt to be generated for more than one type of event. Each time an interrupt occurs, the Status Register will indicate the reason the interrupt was generated, based on the specific bit being read and its current state. For the three types of events that can cause interrupts to occur, the status register bits for these interrupts (from table 7-14, page 7-11) are Bits 5, 6, and 7, with Bit 5 indicating the Heartbeat output status.

FAQs about interrupts

Q. I have a customer question on the TSAT-PCI 2 card. How many separate interrupts are allowed on the timing card?
A To answer your question, the Spectracom TPRO/TSAT-PCI-U-2 timing board has three available types of interrupt generation. The three reasons for interrupt generation include Heartbeat, Match (Time Tagging) and FIFO ready (indicating the FIFO buffer is ready to be read). Attached you should find a copy of the TPRO/TSAT-PCI-U-2 users manual. Refer to page 7-12 (Section 7.15) of this manual for more information on the Interrupt generation. From this section of the manual:

The board can generate interrupts when the FIFO has data, when each heartbeat occurs,and/or when the commanded Start Time occurs. Each of these is enabled separately via the Interrupt Control Port, and any, all, or none can be enabled simultaneously.

All three of these reasons that interrupts can be generated for will generate one interrupt. When an interrupt occurs, the Status Register can be read to determine which of these three reasons was the one that triggered the interrupt to occur. Refer to page 7-11 (Section 7.14) for more information on reading the Status Register.

“Wait” calls (“Waitfor” type calls for the PCI series timing boards)

- Refer to the Windows or Linux API guides for info on the wait calls.
- There are a total of three “**wait**” calls that can be used to temporarily halt application software:
- **TPRO_waitEvent:** This routine will report the time an external event was detected on the Time Tag Input pin. The routine will block for a given number of ticks (in milliseconds) until an event occurs or the timeout period has been reached.
- **TPRO_waitHeartbeat:** This routine will report the condition of the heartbeat output. The routine will block for a given number of ticks (in milliseconds) until a heartbeat occurs or the timeout period has been reached.

Note: This API call uses the Heartbeat interrupt (not the heartbeat signal itself). So the Heartbeat interrupt needs to be enabled each time the board powers- up in order to use this command.

- **TPRO_waitMatch:** This routine will report the condition of the match start time. The routine will block for a given number of ticks (in milliseconds) until a match start time occurs or the timeout period has been reached.

32 Bit application software on our 64 bit driver (compat_ioctl function) for TPRO boards

- Refer to Salesforce case **25415** for request to add this feature to TPRO boards (this request was from Eventide)
- For earlier request to add this function to TSync boards, refer to Salesfor caswe 11668 and Fogbugz case 1856 (for request from Raytheon).

Email from David Qi with Raytheon We would like our 32-bit application to work with the TSync-PCIe Linux driver when it is built to run on a 64-bit Linux kernel.

We know that a 32-bit application cannot work with the TSync-PCIe Linux driver when it is built to run on a 64-bit Linux kernel, because the TSync-PCIe code does not use provide driver support -- in the form of a compat_ioctl function -- that would allow a 32-bit application to work with the driver when it is built to run on a 64-bit kernel.

So, the question we have is this ...

Would Spectracom be able to provide us (and its other customers) with a version of the TSync-PCIe Linux driver that allows a 32-bit application to work with its driver, when its driver is built for a 64-bit Linux kernel?

Status update (18 Jul 17 KW) I spoke to Dave Sohn and he said we are not adding this functionality to the TPRO/TSAT timing board product line.

**PowerPC:

Q. Is PowerPC included in the supported architectures list of your Linux device-drivers?
A **Email I sent to Florise on 8/25/10, after talking to Tim Tetreault**: We have not tested either of these Linux drivers (cPCI and PCI-66U) on PowerPC and don't have any feedback from customers as to whether they will work on PowerPC. We do not have a PowerPC system to test this on and this configuration is not very common. Engineering informed me that with the cPCI board, it's not very likely to be compatible with PowerPC because it's a fairly old driver. It is more likely to be compatible with the TPRO-PCI-66U boards, as this driver is much newer.

However, engineering reminded me that the timing board does not need to be installed in the computer before installing the driver. Your customer is free to download and try installing either driver, even before they purchase any of the boards. The drivers are available at no cost from the Spectracom website address of:
http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=20

**TPRO/TSAT Windows Driver

Note: **Dave Sohn does not intend to release any newer versions of the TPRO/TSAT boards (28 Sept 17)** At this time, we are not spending effort on modifications for the TPRO/TSAT drivers. In lieu of that, it may be possible to provide the source project if that could help the customer. Anything else would likely require a greater investment from the customer.

- **Window driver (1186-SD07-xxxx)** where xxxx is the version of the driver (search PLM for "**1186-SD09**")
- **Link to the PCI series Windows driver guide (1186-5002-0050) in PLM**: https://app.bom.com/items/detail-spec?item_id=1202833263&version_id=10212776488
- **Shortcut to driver version upgrades in custservice**: I:\Customer Service\PSB, PSP software updates\TPRO-TSAT boards\Driver updates

***Desire for 32/64 bit signed TPRO/TSAT Windows drivers

- (8 Feb 2019) We have no intentions to release a newer version of Windows driver for Digital Signatures
- (8 Feb 2019) All variants of the TPRO/ TSAT boards (including newer variants such as PCI-U-2 and PCI-66U) can use the TSync Windows driver which supports digital signatures (as of 8 FEB 2019, this newer version of the TSync Windows driver is not yet available). Expecting it to be available sometime soon.

- TPRO/TSAT boards can use limited functionality of the TSync driver (instead of using the TPRO/TSAT driver) because the TSync driver supports all the "TPRO" calls included in the TSync driver.
- TPRO/TSAT boards **cannot** use any of the "enhanced" functionality of the TSync calls also included in the TSync driver- these require the TPRO/TSAT board be replaced by a TSync series board/

A) For Windows 2010

Q What is the plan/timeline for 32/64 Win10 signed drivers, and are the PCI-U2 cards planned for win10 or only 66U?

A Reply from Dave Lorah (10 Sept 15) At this time Spectracom has no plans to update the TPRO/TSAT –U2 or -66U Drivers. We are working on the TSync-PCIe drivers but no updates are planned for the older platforms.

Q. I don't currently have a Win10/64 environment to test with yet - are the 64bit windows drivers signed? We're going to be forced into Win10/64 sooner rather than later, but a large-ish hardware purchase is in the works that includes TPRO U-2 cards (from the same folks that are currently trying to return the batch of -66U cards)...

A from Dave L (18 Feb 2016) I discussed the future of the U-2 and 66U board Windows Driver updates with our design engineer this morning. We do think we will be updating the drivers for the boards in the future but do not at this time have a set date.

We have not tested the current driver on a Windows 10 system yet. It is signed for Win 7 64 bit so there is a chance it will work.
You could try it and see if it works in your systems. If you do and find any results I would be interested to hear what they are.

Special Invensys Windows driver for the smallest possible Windows driver to support the TPRO/TSAT-PCI-U-2 (SW08-DR07-0140) timing boards

- **Link to this Invensys driver:** I:\New Released\Media\SW08-DR07-0140

The smallest and most up to date version of the Windows driver for the TPRO/TSAT-PCI-U-2 boards is software SW07-DR07-0140. This driver was created specifically for Invensys but can also be used by other customers that can install the normal PCI windows drivers, because they can’t fit the full driver on their PC

****Paths to the TPRO/TSAT Windows driver and example programs

Path to the drivers/example programs~~:~~

path to the Driver~~:~~

Example programs

Note: **Dave Sohn does not intend to release any newer versions of the TPRO/TSAT boards (28 Sept 17)** At this time, we are not spending effort on modifications for the TPRO/TSAT drivers. In lieu of that, it may be possible to provide the source project if that could help the customer. Anything else would likely require a greater investment from the customer.

****Errors/problems experienced when installing the PCI Windows driver

- In general, uninstall any previous versions of the driver prior to installing a new version and reboot the PC after installing the new version.

A) Digital signature required for newer versions of Windows

install a newer version of the TSync Windows driver (in place of the TPRO/TSAT Windows driver) which supports digital signatures

- TPRO/TSAT boards can use limited functionality of the TSync driver (instead of using the TPRP/TSAT driver) because the TSync driver supports all the "TPRO" calls included in the TSync driver
- TPRO/TSAT boards **cannot** use any of the "enhanced" functionality of the TSync calls also included in the TSync driver- these require the TPRO/TSAT board be replaced by a TSync series board

B) The Windows Control utility can't be fully opened (doesn't allow you to even be able to let you select which card to open (or selected demo mode). References to Windows "SideBySide". Stereology gives references to Microsoft.vc80.crt.dll (such as Microsoft.VC80.CRT,processorArchitecture="x86",publicKeyToken="1fc8b3b9a1e18e3b",type="win32",version="8.0.50727.4053")

With a brand new install of Vista, we were able to exactly duplicate the symptoms and error messages that you had reported with regards to the Windows driver install. After our engineer ran Windows Update, the issue was resolved!!!

Windows Update added necessary.net files. In order to obtain all of the necessary upgrades, don't select the Windows Update in the lower-right system tray. Instead, select "Windows Update" from the Start menu and then ensure that you force it to check for new updates (important). The engineer informed me that Vista found more than 100 updates that needed to be applied. We just checked again, and there were several more new ones (though the driver worked fine) without needing to install the second batch of updates.

Also, after all of the updates have been installed, make sure to reboot the PC. After performing the Windows Update (forced to check for updates), you should be able to access and control the timing board with the control utility!!

C) TPRO/TSAT-PCI Windows Driver version 2.3.0 can't complete the driver install because a newer version of Visual Studio 2010 is already installed on the PC "A newer version of Microsoft Visual C++ 2010 Redistributable has been detected on the machine."

Note: All the info below is from the Tsync section of this document. Needs to be updated for TPRO/TSAT

- Refer to Fogbugz case 1842 http://fogbugz/default.asp?pre=preMultiSearch&pg=pgList&pgBack=pgSearch&search=2&searchFor=1842&x=0&y=0
- Error message displayed when trying to install the driver: "A newer version of Microsoft Visual C++ 2010 Redistributable has been detected on the machine."
- Does not allow the driver install to proceed past this point.

Dec 2012- Refer to Salesforce case 6784/Fogbugz case 1842 for more info)

As reported by NovaSol, if Visual Studio 2010 is installed and has had updates applied to it, the Windows driver install process will halt and there is no easy way to force it to proceed (apparently it doesn't like the fact that a newer version is already installed).

We will need to release a newer version of the TSAT-PCI Windows driver that allows the upgrade process to proceed, if a newer version is already installed on the PC.

Temporary work-around

Email from Dave Lorah to Tim T (67 Jan 2014) We had a customer last fall with a dependent vc80.crt error and he was able to resolve the issue by totally uninstalling the PCI Driver and then reinserted the CD. When the CD Autostarts it wants to load C++ for you. They selected NO - and did not install anything. They manually selected the XP 86 and loaded just our Driver software. Then after that loaded C++2005 from the web. It then worked OK.

Q. Is there a better workaround for this?

A Tim responded back with Dave,I don't have a better work around. We will probably be looking into a Windows driver update that will include this issue but I don't have a time schedule.

D) "An error occurred while installing system components for the TPRO-TSAT PCI. Setup cannot continue until all the system components have been successfully installed.

- Refer to Salesforce case 24368

****Uninstalling/upgrading the TPRO/TSAT-PCI Windows driver

To uninstall the driver from Windows 7, go to “**Control Panel** > **Programs and Features**” and uninstall “TPRO-TSAT PCI”. We also recommend going to the “Device Manager”, find under “Timing Board” the TPRO board and uninstall it. Once you have done this, you can go ahead and install the new driver by running “setup.exe” in the new driver.

Two partial emails from Tim T to a customer (regarding update to a newer version of the driver):
Check in the Device Manager and make sure the card is attached to the driver. Unless you reboot the machine after installing the driver, you sometimes need to force Windows to scan for new hardware.

Also, once the driver is installed, did you force a “Scan for hardware changes” in the Device Manager?

Email resolution Dave L 2-9-2017

Symptom:
The Driver will not will not link to the board and Control Utility not working.
1. After the driver install, the bios screen on boot up reports unknown card in PCI slot 3, irq10. If i move the TPRO card, the bios reports unknown card in PCI slot4 irq5. So the bios sees the card.
2. After driver install, a log file is copied to the documents folder, which reports that the Tsync driver was successfully installed, but there is no message or log file for the 2.3.0 Tsat PCI driver install.

Resolution:
1. Install both sets of drivers the tsat was no issue.
2. To get the TPRO to install:
open control panel - device manager- see "unknown timing board" under "other" click on the unknown timing board/device
select driver - do automatic install - it will not find driver- when using the automatic install method!!!
then select: let me browse for the driver location - choose the TPRO/TSAT 230 driver for TPRO - open it, then go to drivers folder and open it, then go to x64 folder open it - the select the winxp64 folder and let windows do the driver install from here. After this windows was happy and saw the "spectracom tpro pci timing board" in the device manager.

****Windows TPRO/TSAT Control Utility

TPRO/TSAT Control Utility
File
Time Info
Heartbeat
Match
Date
Events
Sync
Propagation
Satellite
Help
TPRO/TSAT Control Utility
SPECTRACOM
SYNCHRONIZING CRITICAL OPERATIONS®

Control Utility not working correctly in Windows driver version 2.3.0

With the Windows version 2.3.0 PCI driver, the TSync-PCIe driver needs to also be installed **BEFORE** installing the PCI driver. Uninstall the PCI driver, install TSync driver, reinstall the PCI driver.

To begin, the latest version of the Spectracom PCI driver is compatible with Windows 7 (its compatible with both 32 and 64 bit Operating Systems. It's also compatible with 32 and 64 bit application software). For your reference, attached you will find a copy of the Release Notes for the TPRO/TSAT-PCI driver. This document contains the link to be able to download the latest PCI drivers from the Spectracom website, if you haven't already (refer to the last page of the document).

However, there is a minor issue with latest version of the TPRO/TSAT driver, where a necessary component (which is installed as part the TSync-PCIe timing board driver, but not currently as part of the TPRO/TSAT driver) is missing. Without this necessary component installed, the Windows Control Utility for the PCI board is unable to be opened and run.

There is a temporary solution to this issue. We recommend you first uninstall the TPRO/TSAT driver. Then, install the Spectracom TSync-PCIe driver (also available from our website). With the TSync-PCIe driver now installed, reinstall the TPRO/TSAT-PCI driver. Both drivers will now be installed on the same PC. However, even though the PC won't have a TSync-PCIe card installed, it won't hurt anything to have both drivers installed (they won't conflict with each other).

For now, we recommend keeping the TSync-PCIe driver installed. When we release the next version of the TSync-PCIe driver, you can update the PCI driver and uninstall the TSync-PCIe driver at that time, if you wish. I am flagging your record in our Customer Service database and will let you know when the next version of the PCI driver has been released to address this limitation.

The TSync-PCIe driver can also be downloaded at no cost from the Spectracom website. To download this driver, please visit: http://spectracom.com/support/tsync-bus-level-timing-product-support (scroll down to "**TSync Windows Driver version 3.12**" in the "**Software**" section).

TPRO/TSAT Windows driver install issues/errors

General Suggestion: Have the customer try installing/using the TSync Windows driver instead of the TPRO/TSAT driver. For backwards compatibility purposes, the TSync driver contains all the TPRO/TSAT calls and example programs from the TPRO/TSAT driver. And as the TSync drivers are typically updated more often than the TPRO/TSAT drivers, the TSync drivers are more likely to successfully compile.

The TSync Windows driver can be downloaded from our website at: https://files.spectracom.com/public-downloads/tsync-linux-driver

> **Example Email Keith sent to customer (3 Jul 2019)** While we are looking into this, and as the **TSync** Linux driver contains (for backwards compatibility) all the example programs and API calls for the **TPRO/TSAT** boards, one suggestion is to try installing the latest TSync-PCIe Linux driver in the same system (instead of using the dedicated TPRO driver).
>
> The latest version of this TSync-PCIe driver may contain changes not included in the TPRO driver which will allow it to successfully install this driver on your kernel version, and allow you to start using the TPRO card!!!
> The latest TSync Linux driver can be downloaded at no cost from our website at: https://files.spectracom.com/public-downloads/tsync-linux-driver
>
> Please let me know if the TSync Windows driver successfully and allows communications with the timing board.

**Clock Daemon program

- **Refer to (in this document):** Clock Daemon / Clock Daemon service (for all timing boards)

Set heartbeat command/heartbeat interrupts

Q. I successfully implemented the set heartbeat & get time example code for the card. I noticed there was a set heartbeat example and not a get heartbeat example. Is it possible to read the frequency, signal type, and output type from the card?

Reply from Keith, based on input from Tim Tetreault (July 2012): We looked into this and found there isn't a way to read the current values from the Timing board. The only indication of the current settings is when the "set heartbeat" command is issued, it echoes back with how it was just set.

Double heartbeat interrupts

- Reported by Joe Rossolll/Joshua Allen with NASA (Salesforce case 7930)
- Tim Tetreault found issue is in the Windows driver.
- Released a Beta version of Windows driver (March 2013) with full release to follow.

**Labview wrapper

Q. Can you or someone at Spectracom speak to the usability of these drivers with later versions of LabVIEW (2010 +)?
A Reply from Tim Tetreault (19 Apr 2013) We haven't tested the Labview wrapper on 2010+ so I don't know if it works. The wrapper and driver are free for the customer to download and try out.

**TPRO/TSAT Linux Driver

- **Linux driver (1186-SD09-xxxx)** where xxxx is the version of the driver (search PLM for “1186-SD09”)
- **Link to the PCI series Linux driver guide (1186-5003-0050) in PLM**: https://app.bom.com/items/detail-spec?item_id=1202833264&version_id=10212776498

TPRO/TSAT linux driver compile errors

General Suggestion: Have the customer try installing/using the TSync Linux driver instead of the TPRO/TSAT driver. For backwards compatibility purposes, the TSync driver contains all the TPRO/TSAT calls and example programs from the TPRO/TSAT driver. and as the TSync drivers are typically updated more often than the TPRO/TSAT drivers, the TSync drivers are more likely to successfully compile.

The TSync Linux driver can be downloaded from our website at: https://files.spectracom.com/public-downloads/tsync-linux-driver

Needing a newer version of TPRO/TSAT driver than we provide, due to newer kernel version not being supported by our Linux driver

- **install a newer version of the TSync Linux driver (in place of the TPRO/TSAT linux driver) which supports digital signatures**
  - TPRO/TSAT boards can use limited functionality of the TSync linux driver (instead of using the TPRO/TSAT driver) because the TSync driver supports all the “TPRO” calls included in the TSync driver
  - TPRO/TSAT boards **cannot** use any of the “enhanced” functionality of the TSync calls also included in the TSync driver- these require the TPRO/TSAT board be replaced by a TSync series board

Example Email Keith sent to customer (3 Jul 2019) While we are looking into this, and as the **TSync** Linux driver contains (for backwards compatibility) all the example programs and API calls for the **TPRO/TSAT** boards, one suggestion is to try installing the latest TSync-PCIe Linux driver in the same system (instead of using the dedicated TPRO driver).

The latest version of this TSync-PCIe driver may contain changes not included in the TPRO driver which will allow it to successfully install this driver on your kernel version, and allow you to start using the TPRO card!!!
The latest TSync Linux driver can be downloaded at no cost from our website at: https://files.spectracom.com/public-downloads/tsync-linux-driver

Please let me know if the TSync Linux driver successfully installs (with the suggested workaround for the other compile error applied) and allows communications with the timing board.

Specific examples of Linux driver compile errors

1) **“error: implicit declaration of function” and “error: initializer element is not constant”**

- Refer to Salesforce Cases such as **202042** and **17976**

```
datadog@xilinx-T7400:~/Downloads/IRIG_Card_Driver/tpropci$ make
make -C driver
make[1]: Entering directory `/home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver'


make -C /lib/modules/3.13.0-53-generic/build M=/home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver modules
make[2]: Entering directory `/usr/src/linux-headers-3.13.0-53-generic'
CC [M] /home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver/tpro_drv_2_6_13_1.o
/home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver/tpro_drv_2_6_13_1.c:200:5: error: implicit declaration of function
‘__devexit_p’ [-Werror=implicit-function-declaration]
```

```
remove: __devexit_p(tpropci_remove_one),
^
/home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver/tpro_drv_2_6_13_1.c:200:5: error: initializer element is not constant
/home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver/tpro_drv_2_6_13_1.c:200:5: error: (near initialization for ‘tpropci_driver.remove’)
cc1: some warnings being treated as errors
make[3]: *** [/home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver/tpro_drv_2_6_13_1.o] Error 1
make[2]: *** [_module_/home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver] Error 2
make[2]: Leaving directory `/usr/src/linux-headers-3.13.0-53-generic'
make[1]: *** [all] Error 2
make[1]: Leaving directory `/home/datadog/Downloads/IRIG_Card_Driver/tpropci/driver'
make: *** [all] Error 2
```

PCI board not working in system (LSPCI, ASPM, BIOS settings)

- Verify ASPM (Power Management) is disabled in both Bios and Kernel
- **Refer to: [(ASPM) Power Saver/Power Monitor feature]** (In TSync-PCIe section)
- Verify LSPCI response

lspci commands

- *Checks to see if the timing board is visible on the PCI bus*
- Checks to see if PCI driver is installed.

Issues that appear to be associated with the hardware (such as interrupts): Have the customer perform and sent to us an LSPCI –vvv.   This lists modules installed in the machine.

Even without the PCI driver installed, this command should show the board being installed.

In Linux, you can run the following command as root to look for the PCI card specifically on the PCI bus:

“lspci -v” = Lists the devices founds on the PCI bus (-v gives minimal info, -vv give more info and -vvv gives all available info)

Example LSPCI response for PCI board using the original/generic Class Code (March 2013)

```
12:01.0 Class ff00: PLX Technology, Inc. PCI9030 32-bit 33MHz PCI <-> IOBus Bridge (rev 01)
Subsystem: Odetics Unknown device 9050
Control: I/O+ Mem+ BusMaster- SpecCycle- MemWINV- VGASnoop- ParErr+ Stepping- SERR+ FastB2B-
Status: Cap- 66MHz+ UDF- FastB2B+ ParErr- DEVSEL=slow >TAbort- <TAbort- <MAbort- >SERR- <PERR-
Interrupt: pin A routed to IRQ 11
Region 0: Memory at fbff0000 (32-bit, non-prefetchable) [size=64]
Region 1: I/O ports at 5000 [size=64]
Region 2: Memory at fbfe0000 (32-bit, non-prefetchable) [size=64]
Region 3: I/O ports at 5040 [size=64]
```

Class Code for all PCI series boards

ECN 3093 (Nov 2012) is releasing new firmware for the PCI-U-2 and PCI-66U boards, which updates the class code for the boards to a more specific value. This will likely help the installation of the timing board in some systems (especially Linux, but potentially in Windows, also).

Original/generic class code
(As reported in LSPCI response) 12:01.0 **Class ff00**: PLX Technology, Inc. PCI9030 32-bit 33MHz PCI <-> IOBus Bridge (rev 01)

Note: (March 2013) Tim Tetreault has an Action Item to update this class code in a future Firmware update.

- If BIOS has a setting for this (some may), make sure that PCI devices with a Class Code of “FF” are allowed (original class code of the boards is “FF00”)

**lsmod commands

“lsmod” = Lists the drivers that are currently installed:

“rm tsyncpci” = uninstalls the current TSync-PCI driver “**make uninstall**” & “**make clean**” = to remove any old driver files.

“modprobe tsyncpci” = rebuilds and installs the driver

(See email from Anthony Willimitis)
The ntp distribution in this latest driver (SW08-DR09-0301) doesn't work. The new driver source shows an extra field called **flags** in the TimeObj structure. To make NTP function, this extra field must be added to the TimeObj structure in **refclock_tpropci.c** as shown below.

```
/*
** TPRO/TSAT Time Object
*/
typedef struct TimeObj {

  unsigned char  syncOption; /* -M option */
  unsigned int   secsDouble; /* seconds floating pt */
  unsigned char  seconds;    /* seconds whole num */
  unsigned char  minutes;    /* minutes */
  unsigned char  hours;      /* hours */
  unsigned short days;       /* days */
  unsigned short year;       /* year for CPCI board */
  unsigned short flags;      /* bit 2 SYNC, bit 1 TCODE; all others 0 */

} TimeObj;
```

Q. When upgrading a PCI-U to a PCI-U-2 board, do I need to upgrade the driver?
A. No! When replacing a PCI-U timing board with an PCI-33U board (also now referred to as the PCI-U-2 board), the new PCI-U-2 (PCI-33U) board was specifically designed to be a drop-in replacement to a PCI-U board. There is no need to update the driver for this replacement. The only time the driver needs to be updated is if you update from a PCI-U to the PCI-66U board. This board is similar in hardware as the 33U board, but contains different firmware that interacts with changes made to the driver to add enhanced capabilities (even faster time-reads).

**DoAll and test.c example routines

- **DoAll** is included in the **Windows** driver: Located in the “**Windows**” folder –> “**Examples**” folder.
- **Test.c** is included in the **Linux** driver:

These are both a sample routine included with the driver. They are both the same as all of the other examples. These two routines perform various API calls to talk to the board. There is no documentation on these two routines.

**Reading registers directly “Direct memory access” (Peak and Poke)

Values from the TPRO-PCI boards can be read from or written directly to registers on the board with the use of the Spectracom driver.

Peek: Reading the registers directly (Generally, there are no cautions for reading registers directly. Reading the incorrect register will result in wrong info, but unlike a poke of the wrong register, it doesn’t affect the operation of the board).

Poke: Writing to the registers directly (**Caution**: make sure to write to the correct register or bad things can happen).

Advantage of peek and poke: allows the ability to grab more than one value at a time and then the data is parsed out accordingly.
The PC’s BIOS automatically assigns a unique Base address for each PCI bus card installed. Then, each Memory Register is assigned a unit Base address extension to identify each particular register.

Regarding the ability to peak the registers directly, this can certainly be done, if desired. However, please keep in mind that the register values for the earlier TPRO, TPRO-U-2 and the TPRO-33U is not the same register location as the TPRO-66U timing board (as discussed in the TPRO user manual). If the register value is not changing, either the board is not synced to the IRIG input, even with the IRIG cable connected OR, you are reading the wrong register location.

The TPRO user's manual provides a Memory Map for the memory locations (Base address extensions) which store specific contents. The customer can then create their own custom driver to read the contents directly. Refer to the TPRO-PCI user's manual for more details (1186-5001-0050).

The source code supplied with the Linux driver can assist them in identifying the Base address that is assigned by Bios and shows how we access the registers.

Email sent to John Napoleone from Raytheon on 8/08/11
Essentially, the information for reading registers directly is the same for the TPRO/TSAT-PCI-66U boards as it is As the TPRO/TSAT--PCI-U-2 timing board is 32 bit while the TPRO/TSAT-PCI-66U timing board is 64 bit, there are two different sections of the User manual for information, one for each type of most recent PCI timing boards. Section 7 (starting on page 7-1) is specifically for the PCI-U-2 timing boards while Section 8 (starting on page 8-1) is specifically for the TPRO/TSAT-PCI-66U timing boards.

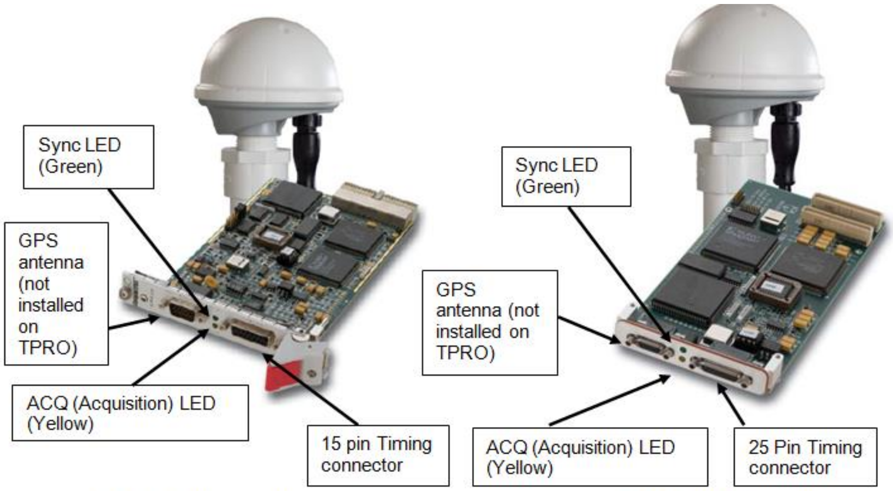
As with the TSync-PCIe boards, we recommend you refer to the Linux source code that we provide with the Linux driver for information to help point you in the right direction to create your own driver (such as how to obtain the Base address automatically assigned by the PC's BIOS and for examples on how we read the registers).

Odetics TPRO-BI timing board

- Refer to Salesforce case 24555
- This is apparently a predecessor to the original KSI TPRO-PCI (5v) timing boards
- It was sold by Odetics.
- We don’t support it or accept it for repair.
- Refer customer to our sales team to see if a newer TPRO/TSAT or TSync board may potentially be a replacement.

Note: The only hit I found with a Google search is: http://www.computerlocators.com/485/price-$2150/TPRO-BI.html

TPRO-cPCI/TSAT-cPCI (compactPCI) and TPRO/TSAT-PMC (mezzanine) / PXI/XMC/FMC boards (discontinued/no longer available)


Sync LED (Green)
GPS antenna (not installed on TPRO)
ACQ (Acquisition) LED (Yellow)
15 pin Timing connector
Sync LED (Green)
GPS antenna (not installed on TPRO)
ACQ (Acquisition) LED (Yellow)
25 Pin Timing connector

TSAT-cPCI board **TSAT-PMC board**

- Shortcut to TPRO/TSAT-PMC manual (1153-5001-0050) in arena at: https://app.bom.com/items/detail-spec?item_id=1202833226&version_id=10212776118&orb_msg_Single_Search_p=1&redirect_Seqno=7069008262
- Shortcut to TPRO/TSAT-cPCI manual (1152-5001-0050) in Arena at : https://app.bom.com/items/detail-spec?item_id=1202833221&version_id=10212776068&orb_msg_Single_Search_p=1&redirect_Seqno=7069003135
- **Shortcut to PMC and cPCI data sheets:** I:\Marketing_Product Data Sheets\Bus-Level Timing Boards
- Shortcut to PMC schematics: (1153-0002-F000) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202840998&version_id=10241794088&orb_msg_Single_Search_p=1&redirect_Seqno=7069041378
- **Shortcut to driver guides:** (1186-5003-0050) in Arena at: https://app.bom.com/items/detail-spec?item_id=1202833264&version_id=10212776498&orb_msg_Single_Search_p=1&redirect_Seqno=7069045653
- Shortcut to PMC/cPCI drivers (1152-0001-6000) in Arena: https://app.bom.com/items/detail-spec?item_id=1202840995&version_id=10212756678&orb_msg_Single_Search_p=1&redirect_Seqno=7069051586
- Link to drivers on our website:

http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=24

Note about Linux driver (15 July 2013) The last update we did to the Linux driver, we combined the cPCI, PMC & PCI drivers into one driver.

Shortcut to Customer Service folder I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT

Note:

- **cPC**I: stands for Compact PCI
- **PMC:** stands for PCI Mezzanine Card
- **XMC**: also called a "Switched Mezzanine Card"

Obsolescence/Discontinued TPRO/TSAT-cPCI board and associated parts

email from Dave Sohn (25 Sept 2019) Only the TPRO/TSAT-PCI boards have not been discontinued. TPRO/TSAT cPCI are unavailable. I suggest looking at the TSync-cPCI boards instead.

TPRO/TSAT-PMC and TPRO/TSAT-cPCI boards

- Refer to: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI
- TPRO/TSAT-PMC boards discontinued ~28 Feb, 2014
- TPRO/TSAT-cPCI boards discontinued ~5 Oct 2013

Discontinued parts for TPRO/TSAT-cPCI boards

- **U14** and **U24** are obsolete/no longer available

Email from Dave L to Taylor (with Dongjin) 22 Mar According to the repair technician the **U14** FPGA device is obsolete and no longer available

Email from Dave L to Taylor (with Dongjin) 23 Mar 2018 The U16 IC is still available to us but the **U24** IC is obsolete and unavailable.

TSync-cPCI as an available replacement for legacy TPRO/TSAT-cPCI boards

Replaced by TSync-cPCI

- Refer to (in this document) CTC-cPCI (CTC/Blitz) and TSync-cPCI (Commercial version of the "blitz" cPCI board) 1205-xxxx

Link to a good comparison of TPRO/TSAT-cPCI versus new TSync-cPCI:
http://partner.spectracomcorp.com/Portals/7/Other/Compact%20PCI%20Timing%20Board%20Update%20-%20Model%20TSync-cPCI.pdf

Email Keith sent to Nidal (25 Sept 2019) Having your customer consider "upgrading" from TPRO-cPCI to TSync-cPCI is a great suggestion, which I didn't think of. The TSync-cPCI is not "exactly" a "complete plug-n-play" swap for TPRO-cPCI, but it's pretty close to being a "direct replacement" for the older timing board.

The TSync-cPCI will work in the same slot as the TPRO-cPCI board was installed.

The customer will still need to install the associated TSync driver (available at no cost from our website). The TSync drivers contain all the legacy TPRO/TSAT API calls. The TPRO.lib and TPRO.h files are compatible with the newly installed TSync board, as long as there is no need/desire to take advantage of the newer API calls (such as holdover, for instance), customer application software doesn't need to be changed. But, the application software needs to be re-compiled with the TSync driver before it will work with the TSync-cPCI board.

Note: Customer must install the TSync driver when TSync-PCIe board is installed. The previously installed TPRO/TSAT driver won't be able to communicate with the installed TSync-cPCI board (the device ID is different). So, the TSync driver also needs to be installed (the TPRO/TSAT driver can either be uninstalled or installed simultaneously with the TSync driver, as desired).

Included with TPRO/TSAT-cPCI timing board (NOTE TPRO/TSAT-PMC and TPRO/TSAT-cPCI boards are no longer available. Refer to TSync-PMC and TSync-cPCI)

TPRO/TSAT-cPCI BOARDS

TPRO-cPCI (1152-0002-0600)

0810545 cPCI breakout cable

TSAT-cPCI: (1152-0001-0600)

- **0810545** cPCI breakout cable
- **1159-0000-5000** Acutime GPS antenna

  Refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- **CA05-1512-0100** ANTENNA CABLE, 100 FT
- **CA05-1515-1003** CABLE ASSY, 15 F D-SUB TO 15 F HD D-SUB
- **1159-0000-0700 ANTENNA MAST** (in Arena) https://app.bom.com/items/detail-spec?item_id=1202841005&version_id=10221291538&orb_msg_Single_Search_p=1&redirect_Seqno=6678014486

TPRO-PMC /TSAT-PMC timing boards (discontinued/no longer available)

TPRO/TSAT-PMC boards are no longer available/they were replaced by TSync-PMC

slightly modified email from Dave Sohn (25 Sept 2019) Only the PCI boards have not been discontinued. TPRO/TSAT PMC are unavailable. I suggest looking at the TSync-PMC boards instead.

Replacing a legacy TPRO/TSAT-PMC with a TSync-PMC board

Slightly modified email Keith sent to Nidal (25 Sept 2019)
Having your customer consider "upgrading" from TPRO-cPCI to TSync-cPCI is a great suggestion, which I didn't think of. The TSync-cPCI is not "exactly" a "complete plug-n-play" swap for TPRO-cPCI, but it's pretty close to being a "direct replacement" for the older timing board.

The TSync-PMC will work in the same slot as the TPRO-PMC board was installed.

The customer will still need to install the associated TSync driver (available at no cost from our website). The TSync drivers contain all the legacy TPRO/TSAT API calls/example programs. The TPRO.lib and TPRO.h files are compatible with the newly installed TSync board, as long as there is no need/desire to take advantage of the newer API calls (such as holdover, for instance), customer application software doesn't need to be changed. But, the application software needs to be re-compiled with the TSync driver before it will work with the TSync-PMC board.

Note: Customer must install the TSync driver when TSync-PCIe board is installed. The previously installed TPRO/TSAT driver won't be able to communicate with the installed TSync-cPCI board (the device ID is different). So, the TSync driver also needs to be installed (the TPRO/TSAT driver can either be uninstalled or installed simultaneously with the TSync driver, as desired).

TPRO-PMC (1153-0002-0600)

0810555 PMC breakout cable

TSAT-PMC (1153-0001-0600)

- **0810555** PMC breakout cable
- **1152-0000-5000** Acutime GPS antenna
  - Refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf
- **CA05-1512-0100** ANTENNA CABLE, 100 FT
- **CA05-1515-2003** ABLE ASSY, 15 F D-SUB TO 15 F MICRO-D
- **1159-0000-0700 ANTENNA MAST**(in Arena) https://app.bom.com/items/detail-spec?item_id=1202841005&version_id=10221291538&orb_msg_Single_Search_p=1&redirect_Seqno=6678014486

CERTS / HARDWARE (Specs and FAQs about the board)

****CPCI / PXI general FAQs**

- PCI Industrial Manufacturers Group (PICMG): governs CompactPCI (cPCI)
- PXI Systems Alliance (PXISA), governs PXI

****PMC / XMC Mezzanine general FAQs**

- **PMC**: Stands for “PCI Mezzanine Card”
- **XMC**: also called a “Switched Mezzanine Card”
- **FMC**: also called a” FPGA Mezzanine Card”

**MTBF

Refer to: **MTBF/MTTR (for all products)**

**Certifications (CE, FCC, etc)

CE Declaration of conformity

As of at least October 2012, the TPRO/TSAT-cPCI, TPRO/TSAT-PMC, TPRO/TSAT-PC104 and TPRO/TSAT-VME timing boards are not CE approved and therefore do not have an available CE Declaration of conformity.

**Hardware interface requirements for the PMC board to be recognized

Q. Anyway can I get a copy of the TPRO-PMC schematic? At least the PMC section? Your card is not being detected by XP when put in a PMC 64/133mhz PMC slot. I would like to know how you have the PLX 9050 strapped. There are also some jumpers and strapping on the TPRO-PMC. These don’t seem to be in the manual. Can you tell me what they do?

A. I have attached a portion of the TPRO-PMC board schematic to a following email that shows the PMC Bus configuration. Hopefully this will help you determine what the problem might be. We were concerned that the bus level was not 5V but 3.3V. It has to be 5V for the board to operate properly. Otherwise the board should be detected as new found hardware. The jumpers on the board are for programming, test and chip selection and do not need to be configured in the field.

PCI Mezzanine® (PMC) Interface
PCI Mezzanine® Interface Standard 32-bit (PMC1J1, PMC1J2)
PCI Mezzanine® Spec IEEE Std 1386.1 - 2001 Compliant
Memory Map 64 consecutive 32-bit words (256 bytes)
I/O Map (none)
Chipset Vendor ID (PLX Technology, Inc.) 0x10b5
Chipset Device ID (PLX 9050 Chip) 0x9050
Subsystem Vendor ID (Spectracom) 0x1347
Subsystem Device ID (TPRO-PMC) 0x 7200
Subsystem Device ID (TSAT-PMC) 0x 7300

Schematic showing the PMC connectors for interface with a PMC bus- Refer to EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI\Schematic TPRO-PMC BUS.pdf

**PXI Express (CPCI board with an additional connector)

Refer to: http://www.ni.com/white-paper/2876/en

Ensuring the successful integration of PCI Express technology into the PXI and CompactPCI backplanes, engineers within the PCI Industrial Manufacturers Group (PICMG), which governs CompactPCI, and the PXI Systems Alliance (PXISA), which governs PXI, have worked to ensure that the PCI Express technology can be integrated into the backplane while still preserving compatibility with the large installed base of existing systems. With PXI Express, users benefit from significantly increased bandwidth, ensured backward compatibility, and additional timing and synchronization features; thus improving upon an already established platform.

- Per Tim Tetreault - PXI Express is an extension of cPCI.
- PXI is an industrial/ruggedized specification based on the cPCI specification).
- PXI significantly improves bandwidth for new applications.
- PXI Express increases the available PXI bandwidth from 132 MB/s to 8 GB/s for a more than 60X improvement in bandwidth while still maintaining software and hardware compatibility with PXI modules
- PXI uses the cPCI connector, plus one additional connector (this second connector is not currently supported but the cPCI connector works with the cPCI timing boards).
- PXI Express provides the additional timing and synchronization features of a differential system clock, differential signaling, and differential star triggers. These provide increased noise immunity for instrumentation clocks and the ability to transmit at higher frequency clocks
- The cPCI board will work as a standard cPCI board in a PXI system (Time stamps, interrupts, etc) but the PXI functionality won’t be available
- You can quote the cPCI board for PXI systems, but need to first find out what functions are desired for the board to perform. If they are associated with the PXI extensions, this board will not work for this application.

**XMC/FMC (based on PMC functionality)

XMC

- Also called “Switched Mezzanine Card”
- XMC is PMC with high-speed serial fabric interconnect.
- PMC with an extra connector.

FMC

- *FPGA Mezzanine Card

(from http://us.kontron.com/technologies/xmc-pmc/) FPGA Mezzanine Card, or FMC, as defined in VITA 57 provides a specification describing a new I/O mezzanine module that will connect to, but not be limited to, 3U and 6U form factor cards. **FMC** modules use a smaller form factor compared with PMC or XMC modules, and assume connection to an FPGA or other device with reconfigurable I/O capability. It is expected that the FMC will be used in a wide range of markets, environments, and carrier card form factors supporting a wide range of I/O interfaces. The standard describes options to create modules for operating in a range of environments from passively cooled to fully ruggedized conduction cooled.

#PMC_PRESENT

Q. The TPRO-PMC card does not pull the #PMC_PRESENT (Pn1 pin 7) PMC signal to gnd. This is required by the VPX3-127 SBC as this signal switches a mux between the XMC & PMC interfaces on the PMC/XMC site. We need a fix from Spectracom to remedy this.

The TPRO-PMC gets correctly enumerate on a VPX3-215 carrier card but the registers cannot be accessed from within lynxos – I cannot determine why yet. Other PMC modules (TTE/UCM) can be correctly accessed in the VPX3-127 PMC site as well as the VPX3-215 carrier so I do not expect problems with SBC/Carrier HW. With the TPRO-PMC on the carrier I could read the registers ok in the CW firmware – which means that lynxos178 BSP still has a bug in the PCI function. (Probably the PCI/PCIe bridges are not setup correctly). This is something we should be able to fix given some time. Note that we will then still have a problem with the TPRO-PMC in the VPX3-127 PMC site until the #PMC-PRESENT signal is pulled low.

A. Email from Tim T (4/11/12) The fix for the #PMC_PRESENT line to GND was done over a year ago. (ECN 2578)

**PMC/CPCI boards don't appear to be operating after install

I'll start by saying that the TPRO-PMC timing boards are compatible with VxWorks version 5.5.1. So the issues that you are observing shouldn't be related to the version of the OS. I'm not sure why the timing boards don't seem to be operating for you.

A few thoughts for you that may help:

1) When the timing board was first installed, were the boards detected as "new hardware"?

2) The TPRO-PMC timing boards are only compatible with 5vdc/5v logic level motherboards (PICMG 2.0 specs). They are not compatible with 3.3v systems.

   Note: Each of these timing boards should have a blue key installed to help prevent them from being plugged into a 3.3v system. If this key is not installed, please let me know as it was inadvertently left off of the bus connector (no key installed indicates they are compatible with 3.3v or 5v systems).

3) The power requirements from the motherboard's PCI bus are as follows:

4) +5 Vdc @ 425 mA max

5) +12 Vdc @ 225 mA max

6) –12 Vdc @ 50 mA max

7) When the TPRO-PMC boards are first powered-up, the yellow ACQ LED should be lit, if the IRIG input signal is attached. With IRIG input not yet connected, this LED may intermittently flicker on/off. During the power-on sequence, this LED and the green Sync LED should momentarily illuminate (then turn off).

8) The driver contains example programs that can be run. The path to the example programs is: **C:\Program Files\KSI\Examples.** These are the "command line" processes that can be run.

9) Lastly, please take a look at the "PLX" IC that is installed in reference designator location "U21". Please let me know if this IC is annotated with "**PC10950**". This component should install be annotated as either "**PC10950-1**" or as "**PC10950-1F**". The "**PC10950**" is not compatible with all newer machines.

Note: For more info on this IC issue, refer also to "**Known issues with both cPCI and PMC boards"** (about 35 pages below).

**Bus interface/ Input voltages / bus logic levels

The specs for the cPCI board's bus interface is "PICMG 2.0" (refer to sites such as: http://www.picmgeu.org/specs/available_Specifications.htm) for more info on the bus specs. It is only compatible with 5vdc power and 5v logic systems. It is not compatible with 3.3vdc power or 3.3v logic systems. Installing the cPCI/PMC boards in a 3.3v logic/power system can cause the computer to hang/crash.

The boards should have a blue "key" installed on the connector that prevents them from even being plugged into a 3.3v system. However, if the key is inadvertently left of the board, allowing them to be installed in a 3.3v system, this is not likely to damage either the motherboard or the timing board.

FYI: Blue key indicates the board supports 5v logic/power. Yellow key indicates *the board supports 3.3v logic/power. No key installed indicates the board supports both 3.3 and 5v systems.*

Email from Tim T: The part number for the cPCI key is in Visual: KSI1-0041-1698

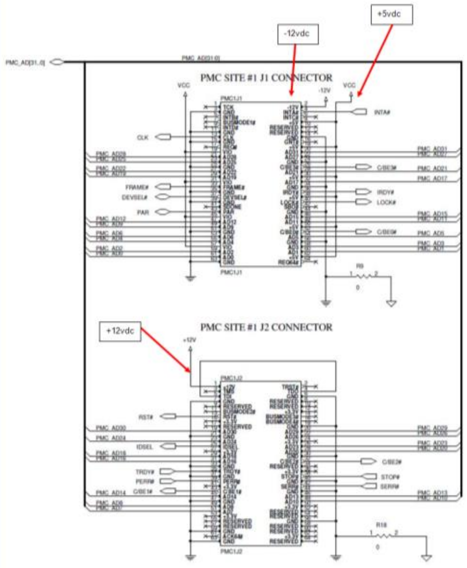
Input voltage/bus logic levels

- PMC and cPCI boards are only compatible with 5vdc/5v logic levels. These boards are **not** compatible with 3.3v systems.
- The boards should have a **blue** key installed to help prevent them from being plugged into a 3.3v system.

Power (from PCI bus):

+5 Vdc @ 425 mA max+
+12 Vdc @ 225 mA max
–12 Vdc @ 50 mA max

Link to drawing below: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI



- Is this a 32-bit cPCI card? Yes. It is a 32-bit card with 16 bit data reads.
- Does this card support a 33 MHz cPCI bus frequency at a 3.3V I/O voltage? It is a 33 MHz bus card, but is

operates on 5vdc only. It is not 3.3v compatible.

- Does this card support a 66 MHz cPCI bus frequency at a 3.3V I/O voltage? It is 33MHz only at 5vdc.

**IEEE 1386 specs

(1/27/12 KW) Alan Turner asked if the TPRO-PMC timing boards are IEEE-1386 compliant. Tim Tetreault verified that we are compliant with this spec.

Tim Tetreault said this spec is regarding the physical dimensions/connectors, etc of a PMC card.

Per Wikipedia: This standard combines the electrical characteristics of the PCI bus with the mechanical dimensions of the Common Mezzanine Card or CMC format (IEEE 1386 standard).

**VME adapter board

Q Could we use the TPRO-PMC in a PMC to VME adapter card?

A Yes this is possible, but the VME driver will have to be modified to work with the PMC board calls. Either that or the PMC driver will need to be adapted to work from a VME computer card. This would take some programming to do and it would have to be done by the customer. Spectracom has no plans to do this at this time. We may change our plans according to demand. If the customer is willing to help fund development costs or commit to a large order we could possible get engineering resources to work on it

Q If so, is there an adapter card that you would recommend for use with the TPRO-PMC module?

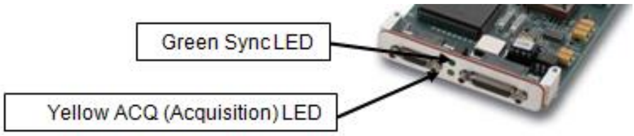
A We have no specific brand adapter card recommended. We have no experience with this application, so we do not know which VME adapter is OK. A simple PMC carrier board should work.

Q What VME address size is the TPRO-PMC compatible with?

A I do not know the answer at this time. I can research this further if necessary.

Input references / Time Sync / Holdover

Edge panel LEDs (ACQ and Sync):



The edge panel has two LED indicator lights:

ACQ Indicator Light (Yellow LED)

The yellow ACQ indicator is lighted when the board is in the process of acquiring either the GPS satellite signals or the incoming IRIG timecode. The ACQ indicator is not lighted if the timecode input is not present, nor if the serial communication to the GPS receiver has errors, or if the board is in-sync.

The ACQ indicator is also lighted momentarily during power-on reset, or when a Forced Reset or Lamp Test command is issued, or when any command is sent to the board when the Blink Yellow Mode is enabled.

Note: With no inputs (IRIG or GPS) connected, the inputs can float, potentially causing the yell ACQ light to flicker or illuminate. This does not affect the operation of the board (including Sync status). To prevent this condition, connect either GPS or IRIG so at least one input is connected.

SYNC Indicator Light (Green LED)

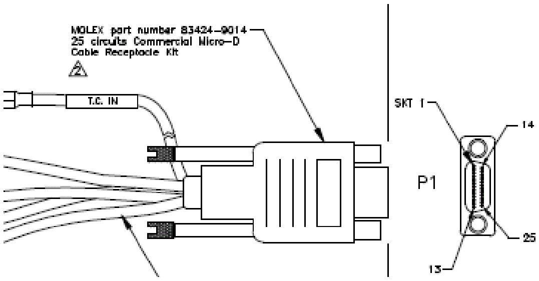
The green SYNC indicator lights when the board has established synchronization with the GPS satellite signal, or input timecode.

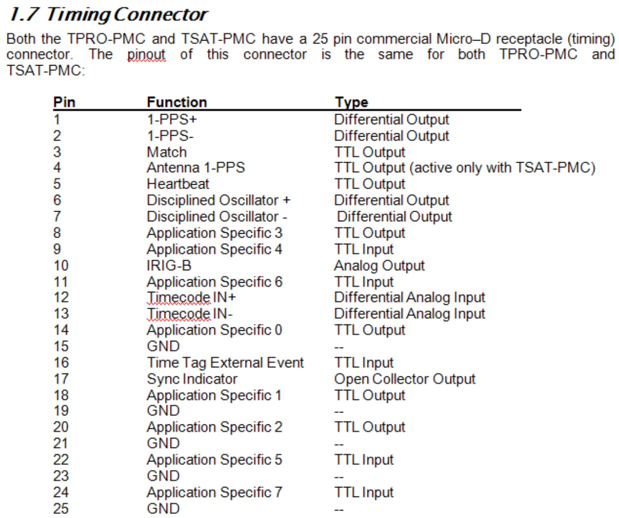
The SYNC indicator is also lighted momentarily during power-on reset, or when a Forced Reset or Lamp Test command is issued.

**Timing connector

TPRO/TSAT-PMC boards

- Timing connector is a **25 pin** female subminiature D connector
- Mating connector for PMC Timing connector = **Molex P/N 83424-9014** or equivalent





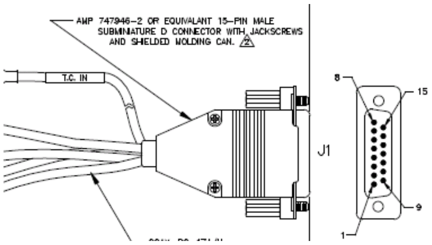
1.7 Timing Connector

Both the TPRO-PMC and TSAT-PMC have a 25 pin commercial Micro–D receptacle (timing) connector. The pinout of this connector is the same for both TPRO-PMC and TSAT-PMC:

Pin	Function	Type
1	1-PPS+	Differential Output
2	1-PPS-	Differential Output
3	Match	TTL Output
4	Antenna 1-PPS	TTL Output (active only with TSAT-PMC)
5	Heartbeat	TTL Output
6	Disciplined Oscillator +	Differential Output
7	Disciplined Oscillator -	Differential Output
8	Application Specific 3	TTL Output
9	Application Specific 4	TTL Input
10	IRIG-B	Analog Output
11	Application Specific 6	TTL Input
12	Timecode IN+	Differential Analog Input
13	Timecode IN-	Differential Analog Input
14	Application Specific 0	TTL Output
15	GND	--
16	Time Tag External Event	TTL Input
17	Sync Indicator	Open Collector Output
18	Application Specific 1	TTL Output
19	GND	--
20	Application Specific 2	TTL Output
21	GND	--
22	Application Specific 5	TTL Input
23	GND	--
24	Application Specific 7	TTL Input
25	GND	--

TPRO/TSAT-cPCI boards

- Timing connector is a **15 pin** female subminiature D connector
- Mating connector for cPCI timing connector = **Amphenol 747946-2** or equivalent


AMP 747946-2 OR EQUIVALANT 15-PIN MALE SUBMINIATURE D CONNECTOR WITH JACKSCREWS AND SHIELDED MOLDING CAN. 2
T.C. IN
J1
8
15
9
1

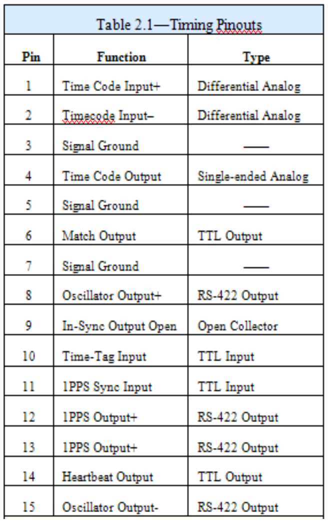


Table 2.1—Timing Pinouts		
Pin	**Function**	**Type**
1	Time Code Input+	Differential Analog
2	Timecode Input–	Differential Analog
3	Signal Ground	——
4	Time Code Output	Single-ended Analog
5	Signal Ground	——
6	Match Output	TTL Output
7	Signal Ground	——
8	Oscillator Output+	RS-422 Output
9	In-Sync Output Open	Open Collector
10	Time-Tag Input	TTL Input
11	1PPS Sync Input	TTL Input
12	1PPS Output+	RS-422 Output
13	1PPS Output+	RS-422 Output
14	Heartbeat Output	TTL Output
15	Oscillator Output-	RS-422 Output

**Breakout cables

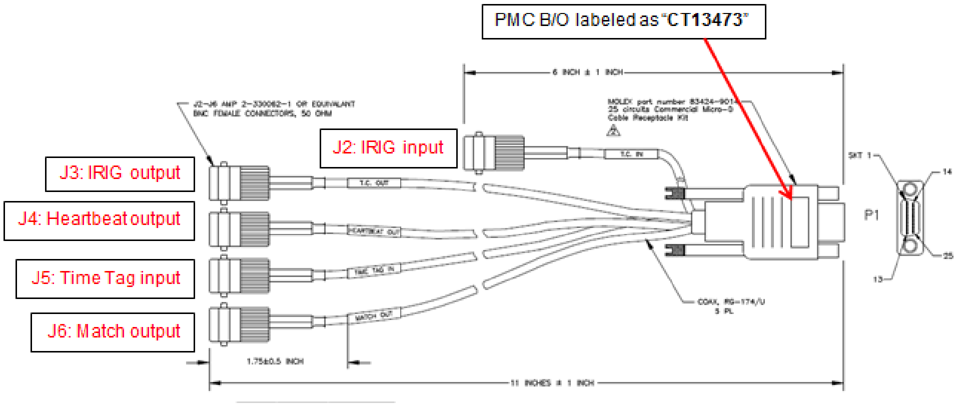
- The PMC and cPCI breakout cables are called out in the Master Pack list. They aren't part of the ancillary kit.
- The PMC and the cPCI boards use a different cable from each other (it's not the same cable for both Models)

A) TPRO-TSAT PMC breakout cable (not cPCI cable)

- Our P/N: **0810555** (labeled on the cable ASSY as MFG P/N note that the cPCI breakout cable has no P/N sticker on it).

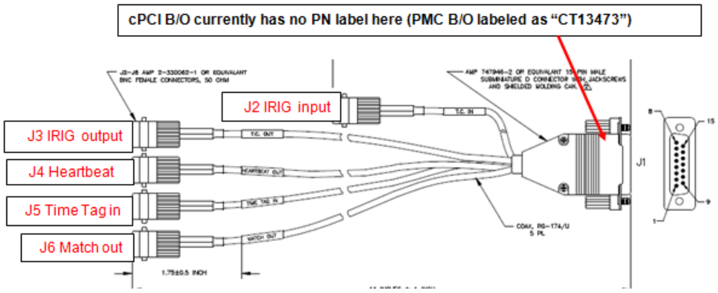
- **Link to cable drawing:** I:\New Released\Cable Drawings\0810555 or I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI

Email from Dave L regarding a PMC board - “The correct cable you should have received is labeled CT13473 on the Micro-D connector. Can you please tell me what the numbers are on the cable you received? It is either labeled on the Micro-D connector or around the bunch of wires leading from the Micro-D connector. I will be able to identify what you have from these numbers. The correct cable has just 5 BNC Connectors. There should be no DB-9 connectors.”



E) TPRO/TSAT-cPCI breakout cable (not PMC cable)

- Our P/N: 0810545 (note the cPCI cable has no label on the DB15 connector- the PMC cable has a sticker with a “CTxxxxx” number on it)
- Link to cable drawing: I:\New Released\Cable Drawings\0810545 or I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI


cPCI B/O currently has no PN label here (PMC B/O labeled as “CT13473”)
J2–J6 AMP 2-330062-1 OR EQUIVALANT BNC FEMALE CONNECTORS, 50 OHM
AMP 747946-2 OR EQUIVALANT 15 PIN MALE SUBMINATURE D CONNECTOR WITH JACKSCREWS AND SHIELDED MOLDING CAN. 2
J2 IRIG input
T.C. IN
J3 IRIG output
T.C. OUT
J4 Heartbeat
HEARTBEAT OUT
J5 Time Tag in
TIME TAG IN
J6 Match out
MATCH OUT
8
15
J1
9
1
COAX, RG-174/U 5 PL
1.75±0.5 INCH

**External 1PPS input

(Email from Tim Tetreault) The cPCI board cannot sync to an external 1PPS reference.

*Manually set Date and Time / Default Day of Year and Time at boot-up

- The cPCI board starts counting at 000 days, 00 hours, 00 minutes, 00 seconds at power-on.
- The clock automatically synchronizes DOY, hours, minutes and seconds to specified timecode signals (GPS also set the current year- IRIG input does not set the current year- which defaults to 0000 at power-up).
- The clock time can be set using the Windows Control utility or it can also be set by user command.
- The Time cannot be manually set successfully on a TSAT board unless the “Disable Sync” command has been executed (if IRIG and/or GPS input is present:

Manually setting the Day of Year / Time

- The TPRO_SetTime API call sets the DOY, hours, minutes and seconds.
- The TPRO_SetYear API call sets the year.

Note: Apparently, the year can also be set in Windows with a command that can be run in a custom batch file (see below)

- Is there a command I could run to automate this procedure on startup via batch script or vbs script?
- Reply from Dave Lorah (16 Jul 14) We do not have a batch file but if you run this command: setyear –y %date:10,4% on startup this will set the year to what the computers year is set to.

One important item to keep in mind- unlike our TSync-PCIe timing boards, the TPRO/TSAT-cPCI timing boards do not have the ability to declare sync status via manually set time. A user can manually set the time of the TPRO/TSAT cPCI boards as desired, but the TPRO/TSAT-cPCI board will not declare its in sync based on manually set time. In order for the board to be in sync, it needs either an IRIG or GPS input (TSAT-cPCI) OR an IRIG input (TPRO-cPCI) be applied.

Depending on their intentions for this timing board, the lack of Sync status may or may not be an issue. For example, if they want to generate an IRIG signal, as the IRIG output does not report Sync status, manually setting the time won’t be an issue. But if they are trying to sync a PC to the time of the board, this requires the board to be in sync. So the board will need to be synced to IRIG or GPS (which override manually set time)

Manually setting the time of the TPRO/TSAT-cPCI board requires an API call be sent to the board (or the time can also be set via the Windows Control Utility, using the **Time Info**-> **Set** menu) . The **TPRO_SetTime** API call specifies the desired time to set the board to (specifies Day of Year, Hours, Minutes and seconds. The **TPRO_SetYear** API call is a separate API call for setting just the desired year.

Ignoring IRIG and/or GPS input to be able to manually set the time

(From “Set Time” section of the cPCI manual)
When not synced to GPS or incoming time code, the host computer can set the time. The time then continues to increment from the set value (freewheel). However, if the GPS receiver begins to track satellites, or if a time code input is applied, time jumps to the GPS or time code time (unless synchronization has been disabled by the *Disable Sync* command).

Desire to manually set the time when an external trigger occurs
Q. Does the card have the capability to receive a trigger and set the IRIG time to any time I want. For example, I have a signal generator in my PXI system that sends out a trigger when it starts waveform generation. I'd like this device to go to a preset time upon receiving that trigger.

A. Reply from Keith (22 Apr 13) This may be able to do this desired functionality via their custom application software. However, the timing board itself doesn't have any provisions to go to a pre-set time based solely on an input trigger.

An input trigger can be used to generate a time stamp of when the time stamp was received. Your customer may be able to create their own software that sends a pre-configured **SetTime** (and **setYear)** if the year needs to be set) API call at particular moment. But the board doesn't have the abilty to have these value pre-entered but not used until a particular input trigger. The time and/or year is set as soon at the API call is issued.

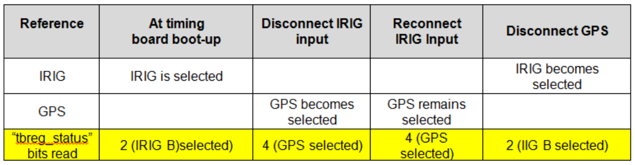
GPS or IRIG input preference / determining if IRIG or GPS is selected (when both are present)

When both GPS and IRIG inputs are present, there are a few different scenarios as to which input reference is selected.

Regarding your question about how to tell which input reference is selected, there isn't an API call specifically available for this information. However, you can perform a register "peek" command to read a stratus register flag directly to obtain this information from the timing board.

Attached you should find a copy of the TPRO/TSAT-PMC User manual. Section 3.45.9.16 "**Synchronization Source Indicator**" on page 3-5 discusses how to read three register bits to determine which input is currently selected. This entails reading bit 18, 17 and 16 in the "**tbreg_Status**" table using the peek command (as shown in Section 3.4.9. page 3-3).

The table below shows which input reference is selected in different scenarios, and what the three bits will indicate, in each scenario (the bits below assume the IRIG input is IRIG B- Not IRIG A).



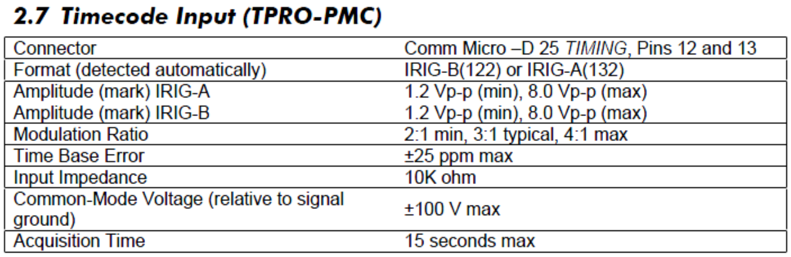
Reference	At timing board boot-up	Disconnect IRIG input	Reconnect IRIG Input	Disconnect GPS
IRIG	IRIG is selected			IRIG becomes selected
GPS		GPS becomes selected	GPS remains selected	
"tbreg_status" bits read	2 (IRIG B)selected)	4 (GPS selected)	4 (GPS selected)	2 (IIG B selected)

As you can see from the table above, IRIG is only preferred over GPS at system boot-up. If it subsequently loses the IRIG input, it will switch to GPS. But just because IRIG is restored, does not cause it to switch back to IRIG input. Instead, it will stay with GPS unless GPS is subsequently lost (or if the board reboots).

**IRIG input synchronization

- The PMC/cPCI board will work with either IRIG AM B122 or IRIG AM A132 input.

IRIG B122= IRIG B, AM, 1000 Hz, BCD_{TOY} (does not read Control Field or year data)
IRIG A132= IRIG A, AM, 10 kHz, BCD_{TOY} (does not read Control Field or year data)



2.7 Timecode Input (TPRO-PMC)

Connector	Comm Micro –D 25 *TIMING*, Pins 12 and 13
Format (detected automatically)	IRIG-B(122) or IRIG-A(132)
Amplitude (mark) IRIG-A Amplitude (mark) IRIG-B	1.2 Vp-p (min), 8.0 Vp-p (max) 1.2 Vp-p (min), 8.0 Vp-p (max)
Modulation Ratio	2:1 min, 3:1 typical, 4:1 max
Time Base Error	±25 ppm max
Input Impedance	10K ohm
Common-Mode Voltage (relative to signal ground)	±100 V max
Acquisition Time	15 seconds max

GPS and IRIG input accuracy

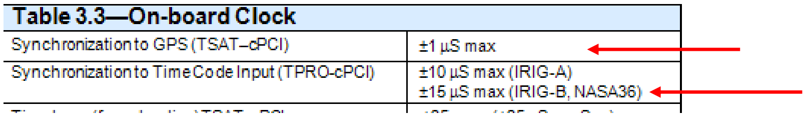


Table 3.3—On-board Clock	
Synchronization to GPS (TSAT–cPCI)	±1 µS max
Synchronization to Time Code Input (TPRO-cPCI)	±10 µS max (IRIG-A) ±15 µS max (IRIG-B, NASA36)
[illegible]	[illegible]

- ➢ The input impedance of the PMC board is 10K ohms.
- ➢ The IRIG Source must also be within stability specifications. The signal should be stable within +/- 25ppm to allow the oscillator to lock onto the IRIG frequency. Usually if the IRIG source is locked to GPS it will be OK.
- ➢ The input circuit of the PMC board is a differential amp so it is very tolerant of DC offsets and noise.
- ➢ IRIG input issue could also be a faulty breakout cable. Is there any signal coming out of the P5 connector?
- ➢ The IRIG Input connects through the breakout cable to pins 12 and 13 of the Micro-D connector on the PMC Board.

**Current Year value (GPS or IRIG input)

Need to set the year with IRIG input

Note: Apparently, the year can also be set in Windows with a command that can be run in a custom batch file (see below)

Q. Is there a command I could run to automate this procedure on startup via batch script or vbs script?
A Reply from Dave Lorah (16 Jul 14) We do not have a batch file but if you run this command: **setyear –y %date:10,4%** on startup this will set the year to what the computers year is set to.

TPRO boards (IRIG input only):

- ➢ The current year has to be set after every reboot of the timing board.
- ➢ The board can never read the year information from the IRIG input signal.
- ➢ The year can be set using the Windows Control Utility, or via the driver (API Call is **TPRO_SetYear**)
- ➢ The year value can either be set by the host machine (refer to "**Set Time**" in the user manual) or it can be commanded to a specific value (refer to "**Set Year**" in the user manual).

TSAT boards (IRIG or GPS input):

- ➢ The current year has to be set after every reboot of the timing board.
- ➢ When synced via GPS, the year information is automatically set.

- When using IRIG input only (no GPS input), the board cannot read the year information from the IRIG input signal.
- When using IRIG input only, the year can be set using the Windows Control Utility, or via the driver.

The year value can either be set by the host machine (refer to "**Set Time**" in the user manual) or it can be commanded to a specific value (refer to "**Set Year**" in the user manual).

There are a couple of different ways to set the year in the PMC/cPCI boards. The easiest method with the Windows driver installed is to use the Windows Control Utility itself. The current Year value can be set under Date/Set Year. The current year can also be set "automatically" using the Windows driver. Page 4-3 of the attached manual discusses the API used to set the year using the driver.

Q: I have connected an external clock with **AM IRIG B B122/B123** through the **AM IRIG INPUT P5** terminal but aparently the board is not getting time as the upper orange led keeps on blinking every 15 seconds or so, but the lower one is never set to ON. I know that my time reference is OK as I have tested it in another device with IRIGB time receiving capabilities as well.
A. (response from Keith) Have you manually set the year in the PMC board since it was last powered-up? As per page 4-3 of the attached TPRO/TSAT-PMC user manual, "Timecodes (IRIG-A, IRIG-B and NASA36) do not convey the year". The year can be set automatically when using GPS as the input to a TSAT-PMC/cPCI bus-level timing board. However, since the IRIG input data stream does not contain the year, the year must be manually set in the board. And since the year is not stored upon power-cycle or power-down of the board, the year must be set each time the board has been powered-up.

Until the year has been set, the green "Time Sync" light cannot illuminate. The yellow light indicates the IRIG signal is being received. But until the year has been set, the board can't sync to the incoming IRIG data stream. Without the year being set and without the green Time Sync light lit, the Windows Clock Daemon program will not be able to sync to the board.

There are a couple of different ways to set the year in the PMC/cPCI boards. The easiest method with the Windows driver installed is to use the Windows Control Utility itself. The current Year value can be set under Date/Set Year. The current year can also be set "automatically" using the Windows driver. Page 4-3 of the attached manual discusses the API used to set the year using the driver.

Q. I have found that the year is not set when I try to set it manually from the utility. I guess that at least the year should be changed in windows and it does not. Moreover, next time I open the Set Year window it does also suggests the value 0. I do not know if this is normal.
A. Regarding the year value being displayed as "0" (instead of the current year) in the Control Utility, I recalled that a screenshot from one of your earlier emails had the year value being correctly displayed as "2010". This indicates you were able to manually set it at one point.

I have a question for you. Has the TSAT-PMC board or the (machine its installed in) been power cycled since you manually set the year? If so, the year value needs to be set again. The reason for this is the timing boards do not save the year during a power cycle. Every time the timing board is powered back up, it needs to obtain the current year value again (either from its input reference or it needs to be manually set, each time). When using GPS as an input to a TSAT board, the timing board is able to automatically obtain the input from the GPS signal. However, when using IRIG as the only input to the board, it can't obtain the current year value from the IRIG signal. So, it must be manually set each time the board is powered back up.

Also, the Date/"Set Year" field in the Control Utility will always show a value of "0" every time you select this drop-down, even if the year was just set and the board has not since been power cycled. The "Set Year" drop-down does not read the year value. In order to confirm the year is set correctly, select the "Date"/"Retrieve Gregorian Date" drop-down to show the date and year, instead.

**Desire to sync one PMC/CPCI board via the IRIG output of another PMC/cPCI board

Q. Would the TPRO-PMC be able to accept the external IRIG B input(s) fed to another IRIG B reader card that resides on the same VME bus?

A. It should, just connect the IRIG output to the IRIG Input–why would you want to do this?
Though the second board can sync via IRIG A or IRIG B, the first board can only output IRIG B. The second board has to sync via IRIG B input.
Based on the accuracy specs of the IRIG B input (+/- 15 us) there can be up to a 15 microsecond offset between the the two board's 1PPS.
The amount of offset between the two boards is likely to vary each time the boards power-up, due to the IRIG input specs mentioned above.

If this offset between the two boards is too large, the boards should be synced to GPS instead (with GPS sync accuracy (+/- 1us).

IEEE-1344 extensions/local time input

- The TPRO/TSAT-PMC boards don't support IEEE 1344 extensions/

Q. The question is if the TPRO/TSAT-PMC timing board supports IEEE 1344 basic only or also the extension.
A Keith responded with (14 May 13)
To answer your question, unlike our TSync-PCIe timing boards, the TPRO/TSAT-PMC timing boards do not support the IEEE1344 extensions. These timing boards can only accept IRIG formats B122 and A132, which do not support IEEE1344 extensions (as shown below):

IRIG B122= IRIG B, AM, 1000 Hz, BCD_{TOY} (does not read Control Field or year data)
IRIG A132= IRIG A, AM, 10 kHz, BCD_{TOY} (does not read Control Field or year data)

As the TPRO/TSAT timing boards only accept IRIG B122 and A132, if the IRIG generator provides the IEEE extensions, the PMC timing boards will ignore the input and not declare sync to the IRIG input. In order for them to sync to the IRIG generator, the IRIG generator has to provide IRIG formats B122 or A132. If the generator provides these formats with local time, the timing boards will sync to the time provided, but will treat the local time as UTC time scale. These timing boards can't convert received local time to UTC time scale.

**GPS input /GPS antenna (Acutime antenna) for TSAT-cPCI and TSAT-PMC

- Refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

GPS and IRIG input accuracy

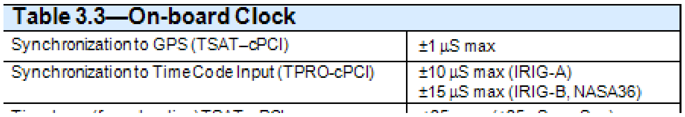


Table 3.3—On-board Clock

Synchronization to GPS (TSAT–cPCI)	±1 µS max
Synchronization to Time Code Input (TPRO-cPCI)	±10 µS max (IRIG-A) ±15 µS max (IRIG-B, NASA36)
[illegible]	[illegible]

- 12 Channel GPS receiver

Acutime antenna programming

The Acutime antennas are programmed at the factory to operate with ONLY TSAT-cPCI and TSAT-PMC boards. They will not operate with any other timing board. More recently shipped Acutime antennas will have a label applied which indicates what timing boards that it is programmed to operate with.

Important Note: Do not connect the Acutime antenna to a TSync-PCIe unless it's for a permanent change, as the TSync-PCIe board will reprogram the antenna to operate with the TSync-PCIe board. It won't work with the PMC or cPCI board again, without factory re-programming of the antenna.

- GPS Antennas for both TSAT-PMC and TSAT-cPCI boards should be annotated as: 1152-0000-5000

Note: P/N 1159-0000-5000 is the Acutime antenna for the PCI series boards. This antenna is NOT compatible with the TSAT-PMC and TSAT cPCI timing boards.

Antenna cables shipped with TSAT-PMC/CPCI boards

There are two cables shipped with the TSAT-PMC and TSAT-cPCI boards. The CA05-1512 cable is a 100 foot antenna cable and the CA05-1515 is a 3 foot adapter cable which adapts the antenna cable to the TSAT-PMC or TSAT-cPCI board.

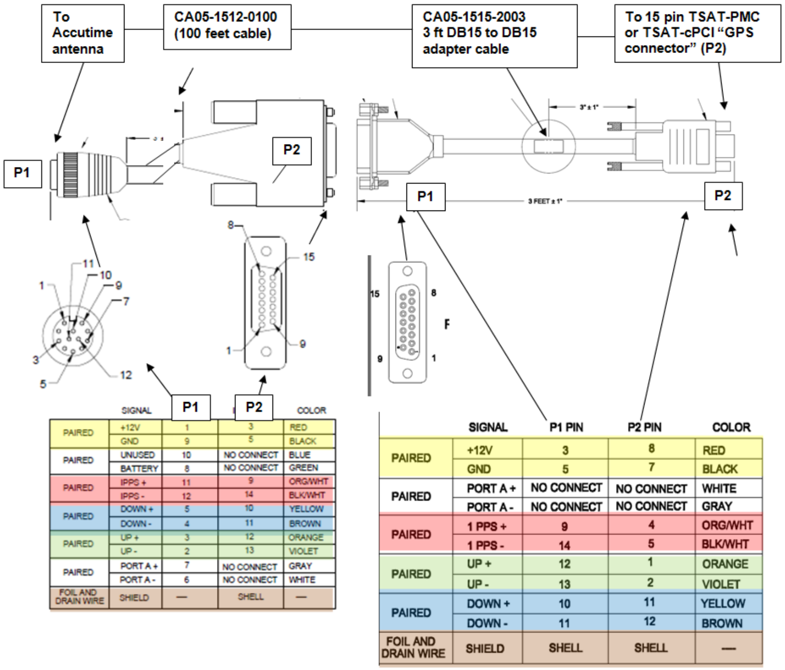
Wiring diagrams

Wiring diagrams for the GPS Antenna cables showing the connector pin-outs. (Refer to I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT for the two “CA05” documents).

Note: Refer to following figures below.

TSAT-PMC and TSAT-cPCI ship with (as shown further below):

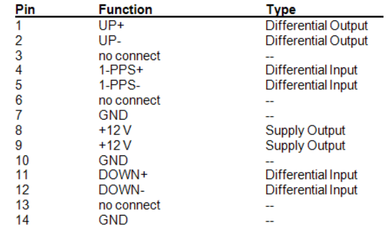
- a 100ft CA05-1512-0100 which connects to the Acutime GPS antenna.
- a short 3 ft adapter cable (CA05-1515-2003) which connects the 100 cable to the TSAT board.


To Accutime antenna
CA05-1512-0100 (100 feet cable)
CA05-1515-2003 3 ft DB15 to DB15 adapter cable
To 15 pin TSAT-PMC or TSAT-cPCI “GPS connector” (P2)
P1
P2
P1
P2
3" ± 1"
3 FEET ± 1"
11
10
9
7
1
3
5
12
8
15
1
9
15
8
9
1
F
P1
P2

	SIGNAL	P1	P2	COLOR
PAIRED	+12V	1	3	RED
	GND	9	5	BLACK
PAIRED	UNUSED	10	NO CONNECT	BLUE
	BATTERY	8	NO CONNECT	GREEN
PAIRED	IPPS +	11	9	ORG/WHT
	IPPS -	12	14	BLK/WHT
PAIRED	DOWN +	5	10	YELLOW
	DOWN -	4	11	BROWN
PAIRED	UP +	3	12	ORANGE
	UP -	2	13	VIOLET
PAIRED	PORT A +	7	NO CONNECT	GRAY
	PORT A -	6	NO CONNECT	WHITE
FOIL AND DRAIN WIRE	SHIELD	—	SHELL	—

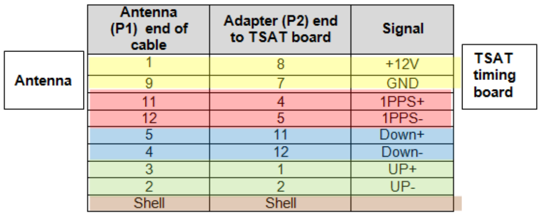
	SIGNAL	P1 PIN	P2 PIN	COLOR
PAIRED	+12V	3	8	RED
	GND	5	7	BLACK
PAIRED	PORT A +	NO CONNECT	NO CONNECT	WHITE
	PORT A -	NO CONNECT	NO CONNECT	GRAY
PAIRED	1 PPS +	9	4	ORG/WHT
	1 PPS -	14	5	BLK/WHT
PAIRED	UP +	12	1	ORANGE
	UP -	13	2	VIOLET
PAIRED	DOWN +	10	11	YELLOW
	DOWN -	11	12	BROWN
FOIL AND DRAIN WIRE	SHIELD	SHELL	SHELL	—

Pin-out for the GPS antenna connector (P2) on TSAT-PMC or cPCI board



Pin	Function	Type
1	UP+	Differential Output
2	UP-	Differential Output
3	no connect	--
4	1-PPS+	Differential Input
5	1-PPS-	Differential Input
6	no connect	--
7	GND	--
8	+12 V	Supply Output
9	+12 V	Supply Output
10	GND	--
11	DOWN+	Differential Input
12	DOWN-	Differential Input
13	no connect	--
14	GND	--

Full End to End (Acutime GPS antenna to TSAT board) cable pin-out:


Antenna
TSAT timing board

Antenna (P1) end of cable	Adapter (P2) end to TSAT board	Signal
1	8	+12V
9	7	GND
11	4	1PPS+
12	5	1PPS-
5	11	Down+
4	12	Down-
3	1	UP+
2	2	UP-
Shell	Shell	

Note: The other pins are no connection.

Also Please ensure the connections are tight and the +12V is present at the antenna connector.

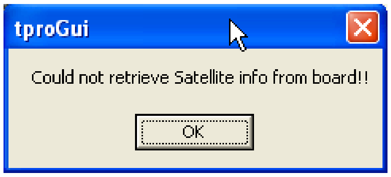
**PC hangs/freezes when GPS antenna is first connected to the TSAT board

Email from Tim Tetreault The only thing I can think of is:

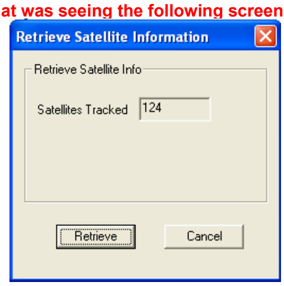
- The board was jarred loose when they connected the cable to the board causing a problem with the PCI bus.
- Their PC does not have the +12V supply that is required to power the antenna.

Somehow the cable they are using is shorting the +12v supply which causes the PC to hang

Email Keith sent on 11 Jan 2013 to a customer that was seeing the following screenshots in the Control Utility"



Could not retrieve Satellite info from board!!



Satellites Tracked 124

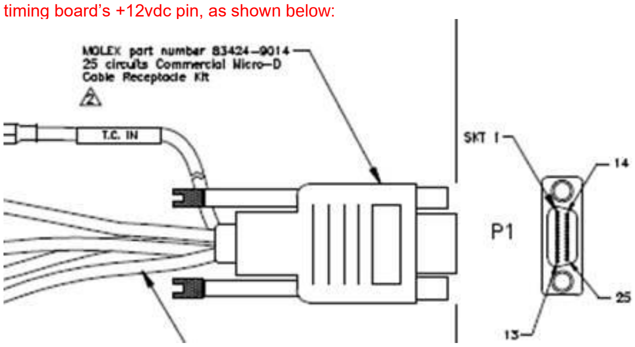
Based on the screenshots you had sent from the Control Utility (showing "124" satellites being tracked), and other error intermittent "error" messages being reported when trying to communicate to the GPS antenna (which also houses the GPS receiver), we are suspecting there is some type of issue with the 12vdc that is initially supplied by the bus connector for the timing board, and then passed to the antenna to power the antenna/receiver.

If this voltage is low or shorting out somehow, it can steal the 12vdc from the timing board, potentially causing the timing board to “hang”. If the card were to “freeze” because of the low voltage condition, this could affect the system bus and other devices on the bus, resulting in the need to have to reboot the system (I believe this along the line of the original failure symptoms).

Based on this theory, and if you haven’t tried this yet, try the normal operation of the timing board and the system it’s installed in, but this time, temporarily leave the GPS antenna cable disconnected from the timing board altogether. The board won’t be able to sync during this test, but what we want to see is if this operation prevents the need to reboot the system, as you have been observing. If these specific symptoms no longer occur, we know that the specific issue is due to the +12vdc supplied by the bus, through the PMC board, to power the antenna. With the issue narrowed down to +12vdc/antenna connection, it will be much easier to find the root cause of these issues.

If the system is able to continue running in the configuration above (antenna cable completely disconnected):

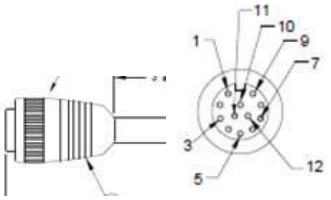
With the GPS antenna cable still disconnected from the timing board, if you have access to a multimeter, measure the 12vdc on the timing board’s +12vdc pin, as shown below:



Might be wrong picture. Should be DB15 not DB25??

+12vdc is on pin 8
Ground is on pin 7

Next, connect only the GPS cable to the timing board (leave the GPS antenna at the other end disconnected from the antenna cable). Does the system continue to operate?
Note: in this particular configuration and if you have access to a multimeter, you could check the +12vdc on the antenna end of the cable. Verify that its very close to +12vdc.



+12vdc is on pin 1
Ground is on pin 9

What is the voltage measured on this end of the coax cable?

Next, connect the GPS antenna at the other end of the antenna cable. Does the system continue to operate?

**Sync Status

- Indicated with the green LED on the edge of the board being lit or by using the **TPRO_SynchStatus** API call.

**Synchronization Source Indicator

These three bits indicate which input time source is being used. This is intended for diagnostic purposes only.

tbstat_Ssi[2:0]	Input Time Source
000	Searching for time code input (TPRO-cPCI) Acquiring GPS satellites (TSAT-cPCI)
001	Time Code Input (IRIG-A autodetected)
010	Time Code Input (IRIG-B autodetected)
011	Time Code Input (NASA36 autodetected)
100	GPS Satellites
Others	(reserved for future use)

**Intermittent loss of sync when IRIG or GPS input is first applied:
The cPCI boards will drop in and out of sync during the first minutes of operation after power up. The boards need to stabilize the oscillator during warm up and it is not uncommon to see the sync go in and out for a while from a cold start. Eventually the time sync should stabilize, but it can take a while sometimes.

Important Note: ECN 2295 (12/23/08) was incorporated to improve the IRIG input operation. Before the ECN was cut-in, the minimum amplitude of the cPCI IRIG-B input was about 3.6 vp-p(Apparently, IRIG A input is slightly better). This is way above the spec mentioned in the manual. Original data sheet spec'd min IRIG signal as 1.2vp-p but this was inaccurate before this ECN.

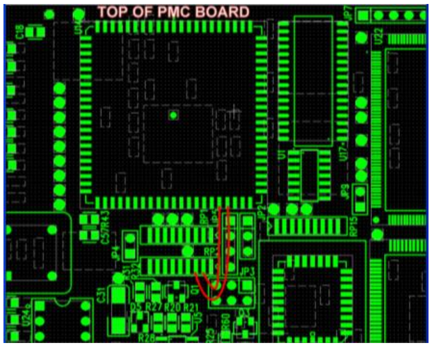
From the ECN: This ECN adds a 220pF capacitor across R24 to the cPCI, and across R24 to the PMC timing cards. Adding a 220pF "!206" sized NPO dielectric SMT capacitor across AM IRIG input fixed gain stage amplifier 47k-ohm feedback resistor to from a 15 kHz low pass filter to the AM IRIG input for the purpose of filtering off high frequency noise to allow the AM IRIG-B input decoding/digitizing hardware to work to original minimum input amplitude specification of 1.2Vp-p.

Also, before the ECN was added, the cPCI/PMC boards would drop in and out of sync during the first minutes of operation after power up. The boards need to stabilize the oscillator during warm up and it is not uncommon to see the sync go in and out for a while from a cold start. Eventually the time sync should stabilize, but it can take a while sometimes.

After the design change was incorporated, the boards were able to sync up much faster and were also able to accept a much lower amplitude signal. The minimum amplitude they could then accept is about 1.5vp-p.

The modification to add this design improvement change consisted of two modifications being added to the PC board. We want to make sure that both of these medications have been added to your TPRO board.

The first change adds two small jumper wires to the top of the board. The two jumper wires should go between the IC labeled “U14” and the resistor pack labeled as “RP2. Below is a picture of the top of the board. The two red lines indicate the location of where these two jumper wires should be added on your board.



Picture is from I:\New Released\PCB Documentation\1153-0002-F000 Rev D\Assembly Drawing

The other modification that needs to be performed is the two sides of one component are soldered onto the two leads of another component. Resistor "R24" should have one side of a capacitor connected to each side of the resistor (the capacitor is in parallel with the resistor).
With the ECN changes added, the PMC/cPCI boards sync up moments after IRIG input of about 1.2vp-p or higher is applied.

**Time base

Q. Be able to keep GPS timebase.
A. (From Wade Sober, based on feedback from Tim T). Yes this is stored in memory.

Oscillators / Osc disciplining to PPS ref / Oscillator free-run capability

**Oscillator disciplining / system on-time point

Q. How is the 1 PPS aligned to the time?
The time is internally aligned to the rising edge of the 1PPS on-time point

**TPRO/TSAT-PMC and TPRO/TSAT-cPCI timing board accuracy specs

The Oscillator on the cPCI/PMC board is 1.5 parts per million (ppm). It is hand adjusted to better than .2ppm at the factory. The ppm is a drift standard meaning that in 1.5 million seconds the oscillator will be off 1 second

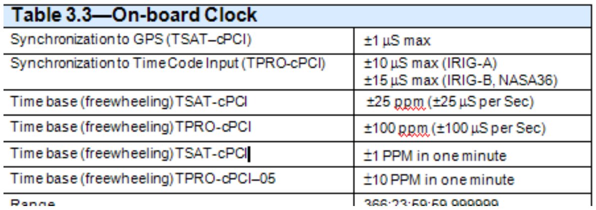


Table 3.3—On-board Clock	
Synchronization to GPS (TSAT–cPCI)	±1 µS max
Synchronization to Time Code Input (TPRO-cPCI)	±10 µS max (IRIG-A) ±15 µS max (IRIG-B, NASA36)
Time base (freewheeling) TSAT-cPCI	±25 ppm (±25 µS per Sec)
Time base (freewheeling) TPRO-cPCI	±100 ppm (±100 µS per Sec)
Time base (freewheeling) TSAT-cPCI	±1 PPM in one minute
Time base (freewheeling) TPRO-cPCI–05	±10 PPM in one minute
Range	366:23:59:59.999999

Q. If synchronization disappears can the card keep time by flywheel?
A. Yes (per Tim Tetreault)

Q. Is 1 PPS output synchronized to card time even when external synchronization is lost or not exists?

A. Yes (per Tim Tetreault)

Inputs

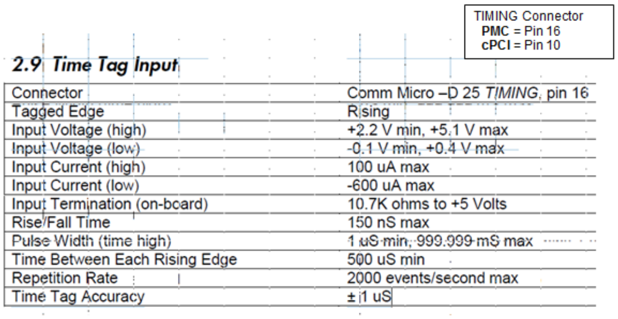
**Timetag / Time Tag (Timetagging'Timestamps/Timestamping)

Notes about timestamping

- Unlike PCI and PC104 boards, the PMC and cPCI boards don't use a FIFO buffer to store timestamps. Instead, one timestamp at a time is stored in the FPGA, until it is read out, one at a time.
- The board only latches one time tag event. If another event occurs before the user reads the time tag registers, the second event will be lost (not latched).
- 2000 events/second max
- Triggers on rising edge of TTL input

Event Input pin

- **TPRO/TSAT-PMC=** Pin 16 of the 25 pin Timing connector on edge of board
- **TPRO/TSAT-cPCI=** Pin 10 of the 25 pin Timing connector on edge of board
- This is a TTL input with an on-board 10K pull-up resistor to +5V.



TIMING Connector
PMC = Pin 16
cPCI = Pin 10

2.9 Time Tag Input

Connector	Comm Micro –D 25 *TIMING*, pin 16
Tagged Edge	Rising
Input Voltage (high)	+2.2 V min, +5.1 V max
Input Voltage (low)	-0.1 V min, +0.4 V max
Input Current (high)	100 uA max
Input Current (low)	-600 uA max
Input Termination (on-board)	10.7K ohms to +5 Volts
Rise/Fall Time	150 nS max
Pulse Width (time high)	1 uS min, 999.999 mS max
Time Between Each Rising Edge	500 uS min
Repetition Rate	2000 events/second max
Time Tag Accuracy	± 1 uS

Enable/Disable Time Tag Input

This bit enables (1) or disables (0) the Time Tag Input on the TIMING connector. Write "1" or "0" to the corresponding bit in the Interrupt Enable Register. The user can read-back this bit by reading the Status Register. The power-on default is "0" (disabled).

Rising or falling edge

The board latches the on-board clock time into a holding register on the rising edge of this TTL signal.

Status Register flag
Flag - Time Tag
This is asserted when a Time Tag event occurs. The user acknowledges the Time Tag event, and de-asserts this bit, by reading the tbreg_ttag_date. This bit might be set at power-on reset; the user must clear this bit before using it.

Interrupt when event occurs

IRQ Enable - Time Tag

Enables an interrupt when Flag - Time Tag is asserted

Time Tag Event Counter

This is intended for diagnostic purposes only. It counts the number of time tag events that have occurred since the time tag registers were read. If it reaches maximum count (0xf) it will remain at maximum count.

The board only latches one time tag event. If another event occurs before the user reads the time tag registers, the second event will be lost (not latched). This counter can be used to determine if time tag events are being lost. Read this counter immediately prior to reading the time tag registers. If the count is zero, no events have occurred. If it is one, the time tag registers contain the latest event time, and no events have been lost. If it is greater than one, some time tag events have been lost.

The user's software must be able to read the time tag register faster than the event repetition rate. External hardware can be used to divide the time tag signal, so only every fifth event is tagged (for example). Such hardware is the user's responsibility.

Blink Yellow Mode (blink the Acq LED each time a timetag is received)

- This diagnostic command blinks the yellow *ACQ* panel light briefly each time the board *finishes* processing *any* command. If the *ACQ* indicator is already lighted, it will extinguish briefly when a command is processed. This can be helpful during software debugging. This mode is disabled when a power-on reset or Forced Reset occurs. There is no response.

NOTE: This command can also be used to provide a visual cue. For example, the user software can be written to send the enable command, immediately followed by a disable command, when a time tag event is detected…

FAQs about Time Tag

Q. Can you provide us with a part number for the required connector?
TPRO/TSAT-PMC= Our P/N for the TPRO/TSAT-PMC breakout cable supplied with timing board is: 0810555 (labeled on the cable ASSY as MFG P/N CT1343). If you desire to fabricate your own breakout cable, the Timing connector mating connector is a Molex P/N 83424-9014 or equivalent 15 female subminature D connector.
OR
TPRO/TSAT-cPCI= Our P/N for the TPRO/TSAT-cPCI breakout cable supplied with timing board is: 0810545 (labeled on the cable ASSY as MFG P/N ???). If you desire to fabricate your own breakout cable, the Timing connector mating connector is an Amphenol 747946-2 or equivalent 15 female subminature D connector.
Note: Refer to I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI for the breakout cable pinout.

Q. When connected to IRIG and GPS, both front panel connectors are busy. Can we still access the time tag input

- TPRO/TSAT-PMC Yes! The supplied breakout cable has indiviual connections. Our P/N for the TPRO/TSAT-PMC breakout cable is: 0810555 (labeled on the cable ASSY as MFG P/N CT1343).

OR
TPRO/TSAT-cPCI Yes! The supplied breakout cable has indiviual connections. Our P/N for the TPRO/TSAT-cPCI breakout cable is: 0810545

Note: Refer to: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI for the breakout cable pinout.

- Be able to provide timestamps for external input pulses.

A (From Wade Sober, based on feedback from Tim T) With TSAT-PMC can provide time stamps for the leading (rising) edge of an incoming pulse (TTL signal).

Outputs

**1 PPS Output

Timing connector pins:
12 = 1PPS Output+
13= 1PPS Output+

This one pulse per second output comes from the on-board clock. It is present regardless of whether the board is synchronized or freewheeling. An RS-422 driver and series 10-ohm resistors in each line are on-board. The recommended termination is 120-ohms, ½ watt, line-to-line (not to ground). The 1PPS Output can be used as a single-ended TTL signal.

**Heartbeat /Programmable output

Specs:

Table 3.12—Heartbeat Output	
Connector	DB-15 TIMING, pin 14
Wave Shape	Pulse
Pulse Polarity	Programmable
Pulse Width	100 nS, 333 nS, 1 uS, or 1 mS (Programmable)
Output Voltage (high)	2.4 V min at 2.5 mA
Output Voltage (low)	0.4 V max at –2.5 mA
Output Current (high or low)	2.5 mA max
Range	200 nS to 65.5 Seconds
Power-on Default	Disabled
Connector	DB-15 TIMING, pin 14

Heartbeat output pin

- **TPRO/TSAT-PMC=** Pin **5** of the 25 pin **Timing** connector on edge of board
- **TPRO/TSAT-cPCI=** Pin **14** of the 25 pin **Timing** connector on edge of board

Jumper setting for Heartbeat output (jumpers located on the edge of the board)

- Jumper 13 to 14 connects the **Heartbeat** output from the on-board circuitry to the **Timing** connector this jumper needs to be installed for the signal to be present.
- Refer to Section 2 (Jumper settings) in the user manual for more information

FAQ'S on programmable outputs

Q. Possibility of creating periodic pulse (frequency) synchronized to 1 PPS (in the customer application 100 Hz)
A. Email from Tim Tetreault (2/9/12) Yes, the cPCI board has a programmable output pin. See spec's (above)

Q. Do we get a 1 PPS signal exactly when the seconds field increments? If not, is there some way for us to coordinate time to the 1 PPS pulse (for the purpose of timestamping)? It doesn't seem an interrupt is generated when a 1 PPS pulse is generated, so it doesn't seem possible for us to query the time based off an interrupt.

A. There isn't an available interrupt generated with the 1PPS signal directly. However, the cPCI board has a Heart Beat output that can be configured for a frequency of 1Hz. There is an available interrupt that can be generated from the Heart Beat output. Setting the Heart Beat output to 1Hz and enabling the Heart Beat interrupt will generate an interrupt that is coincident with the on-time point, which is aligned to the time.

- Be able to specify pulse train period e.g. 1-30 Hz
- (From Wade Sober, based on feedback from Tim T) Can vary the output pulse width.
- Be able to specify pulse width e.g. 100 microseconds, 10 milliseconds
- **(From Wade Sober, based on feedback from Tim T**) Pulse Width: 100 ns, 333 ns, 1 us, or 1 ms (Programmable)
- Be able to start and stop pulse train
- (From Wade Sober, based on feedback from Tim T) Yes we can start and stop output pulse.
- Be able to know what GPS time corresponds to center of pulse width (direct or derived)
- (**From Wade Sober, based on feedback from Tim T)** With TSAT-PMC can only determine the time that the leading or lagging edge occurs. TTL signal.

**Match Output

- Timing connector pin 6
- The Match Output is a TTL output. It goes high at a pre-set time and low at another pre-set time, much like an alarm clock.

**Oscillator Output

Timing connector pin

8= (+)
15= (–)

Software selects whether this signal is 10 MHz, 5 MHz, 1 MHz, 1 kHz, or Off. It is an RS-422 signal with 10-ohm resistors in each line on the board. The recommended termination is 120-ohms, ½ watt, line-to-line (not to ground). The driver is enabled (not tri-stated), held in the "zero" condition, when in the Off mode.

**NMEA output for TSAT-PMC

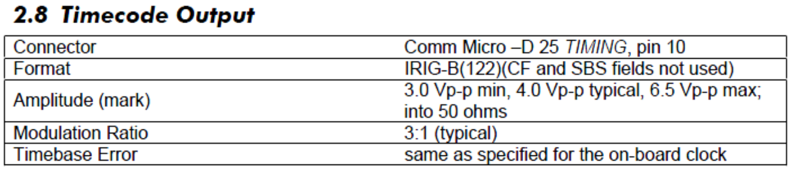
Q.Does TSAT-PMC provide also the receiver position (latitude, longitude and altitude)? Does TSAT-PMC provide it by means of NMEA string format?
A. The PMC driver (for VXWorks) contains separate API calls to retrieve the latitude, longitude, altitude or number of satellites tracked. This information is not available as an ASCII string output from the board. The attached driver guide provides the available TSAT-PMC API calls. I have also attached a copy of the PMC manual that you can also forward to this customer.

**IRIG B Output (PMC and cPCI)

- PMC and cPCI timing boards output only **IRIG B122** (AM 1kHz modulation)

IRIG output pin

- **TPRO/TSAT-PMC=** Pin 10 of the 25 pin Timing connector on edge of board
- **TPRO/TSAT-cPCI=** Pin 4 of the 15 pin Timing connector on edge of board



2.8 Timecode Output

Connector	Comm Micro –D 25 *TIMING*, pin 10
Format	IRIG-B(122)(CF and SBS fields not used)
Amplitude (mark)	3.0 Vp-p min, 4.0 Vp-p typical, 6.5 Vp-p max; into 50 ohms
Modulation Ratio	3:1 (typical)
Timebase Error	same as specified for the on-board clock

IRIG output FAQs

Q. Are BNC output connectors available on the TPRO-PMC? If so, how many maximum outputs would be available.
A. The TPRO-PMC has one available IRIG AM output from the breakout cable. The breakout cable has BNC connectors and attaches to the D-connector on the TPRO-PMC faceplate

Q. Could we augment these outputs with outputs from a secondary PMC module on the same adapter board?
A. In theory this should work, but again there may be something we are not aware of which could cause a problem. Again we have not tried this application.

Q. If so, could we augment those outputs with other PMC modules on additional adapter boards?
A. I assume you are trying to get multiple IRIG outputs from a single VME chassis. An IRIG distribution amplifier would also work.

**Desire to sync a PMC/CPCI board via the IRIG output of another PMC/cPCI board

Q. Would the TPRO-PMC be able to accept the external IRIG B input(s) fed to another IRIG B reader card that resides on the same VME bus?
A. It should, just connect the IRIG output to the IRIG Input–why would you want to do this?
Though the second board can sync via IRIG A or IRIG B, the first board can only output IRIG B. The second board has to sync via IRIG B input.
Based on the accuracy specs of the IRIG B input (+/- 15 us) there can be up to a 15 microsecond offset between the the two board's 1PPS.
The amount of offset between the two boards is likely to vary each time the boards power-up,due to the IRIG input specs mentioned above.
If this offset between the two boards is too large, the boards should be synced to GPS instead (with GPS sync accurary being +/- 1 us).

**Interrupts

These six bits are asserted or de-asserted by writing to the corresponding bits in the Interrupt Enable Register. The user can read-back these bits by reading the Status Register. Writing "1" enables the interrupt. The power-on default is "0".

The interrupt will be asserted as long as the Interrupt Enable and Flag bits are both "1". Be sure that the corresponding flag bit is not asserted before setting the Interrupt Enable bit; otherwise, an unexpected interrupt will occur. The user's interrupt handler software acknowledges the interrupt by clearing the corresponding flag bit.
IRQ Enable–Match
This bit enables an interrupt when Flag–Match is asserted.
IRQ Enable–Heartbeat
This bit enables an interrupt when Flag–Heartbeat is asserted.
IRQ Enable–Time Tag
This bit enables an interrupt when Flag–Time Tag is asserted.
IRQ Enable–Command Complete
This bit enables an interrupt when Flag–Command Complete is asserted.

NOTE: Use this interrupt carefully. The only way to clear Flag-Command Complete is to send another command. This interrupt might be useful if a series of commands is to be sent, but most applications will not use this feature.

IRQ Enable–Sync Change
This bit enables an interrupt when Flag–Sync Change is asserted. This is useful for determining that synchronization has been established or lost.
Enable/Disable Time Tag Input
This bit enables (1) or disables (0) the Time Tag Input on the TIMING connector. Write "1" or "0" to the corresponding bit in the Interrupt Enable Register. The user can read-back this bit by reading the Status Register. The power-on default is "0" (disabled).
Synchronization Source Indicator

Desire to have an interrupt generated once-per-second:
There is no interrupt available that is linked to the 1PPS directly. However, there is one for the Heart Beat output. The heartbeat is configurable to be a 1Hz output. Set the Heartbeat for 1Hz and enable the Heartbeat interrupt and an interrupt will be generated once-a- second, coincident to the on-time point.

FIRMWARE/SOFTWARE/DRIVERS (Windows Linux VxWorks LynxOS)

Firmware/ Driver versions

Firmware info

Refer to: PSB, PSP software updates\TPRO-TSAT boards\board firmware

Determining current firmware version installed

- Use the Windows Control Utility, if board is installed in Windows PC
- Use the **TPRO_GetFirmware** API call
  - This call will respond with a value such as “**02-11-10-23**”.
  - The first two sets of numbers (**“02**” and “**11**” in the example above) are the board’s firmware version (no conversions necessary). In this example, the firmware version is 2.11.
  - The second two sets of number (**“10”** and “**23**” in the example above) are related to the EEPROM and are dependent on which DIP switches for Options are on and which are off.

Timing board Driver info

Driver versions

- Refer to: PSB, PSP software updates\TPRO-TSAT boards\Driver updates

Determining current driver version

- Use the Windows Control Utility, if board is installed in Windows PC
- Use the **TPRO_GetDriver** API call

Memory/Sanitization

Refer to: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI\CPCI-PMC memory

Email from Tim Tetreault (5/31/12)
Nothing is stored on power-down for the PMC/cPCI boards.

I have verified with our Engineering team that it's totally acceptable to share with you the functions of the four components of the Spectracom TPRO-PMC boards.

Below is this information for your reference:

- (U17) SRAM – Used as memory for microprocessor
- (U16) EEPROM – Holds the software that runs on the microprocessor (U16)
- (U23) Serial EEPROM – Holds PCI configuration information used by the PCI bridge chip (U23)
- (U24) Serial PROM – Holds firmware that is loaded into the FPGA at power-up (U24)

**Example programs

- The drivers contain Example programs for showing how to code the calls in C language.
- Path to example programs C:\Program Files\KSI\Examples

**Linux Driver

Drivers are located in: I:\New Released\Media (1152-0001-6000)

Linux driver update for version 2.6.x/3.0 linux kernel support
(9/25/12) Version 4.00 of the PCI series Linux driver was updated (reference ECN 3035) to support the PMC and cPCI timing boards. PCI linux driver now supports kernel versions to version 3.4.6.

Refer customers to the PCI Linux driver, for PMC/cPCI with kernel versions beyond 2.4.

Previous Driver Version 1.2.1

(1/25/12 KW) The version 1.2.1 Linux driver for the PMC and cPCI driver is missing some files that weren't included in the last release. These can be emailed to a customer, if they happen to need them when they are trying to create their own Linux driver (because the current version of this driver does not support kernel version beyond version 2.4)

(Email from Luigi Andrea on 12/16/11)
I'm trying to compile the driver but I'm having some problems. The biggest one is that into the tarball there isn't the file tpro.h included in tprocpci.c and in tprocpci.h.

Note: Missing files are in a folder at: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PMC and cPCI

**LynxOS (Lynx OS)

- **Drivers are located in:** I:\New Released\Media (1152-0001-6000)

Linux versions beyond what the Spectracom driver currently supports
The Linux driver includes source code, so a customer can build their own custom driver for supporting newer versions of VxWorks

Q. We have a customer who is testing one of our Timing cards (TPRO-PMC) and the end customer needs Lynx OS Drivers, we have checked our website, but don't seem to have any, do you know if our Timing cards can support Lynx OS and if possible point us in the right direction for drivers?

Keith's reply (3/22/12)
We don't currently have any TPRO/TSAT-PMC drivers available for LynxOS. With this particular bus-level timing board, we only have Spectracom drivers available for Windows, Linux (currently, version 2.4 kernels only- but will support for version 2.6 kernels in a few months- Sept 2012) as well as VXWorks5.5.1/Tornado 2.2.1.

However, we do provide the source code with the available Linux driver for the TPRO/TSAT-PMC timing boards (which can be downloaded at no cost at http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=24). Using this source code for the Linux PMC driver, your customer may be able to create their own custom LynxOS driver for the TPRO-PMC timing board.

The only timing boards that we currently offer with an available Spectracom LynxOS driver is the TPRO/TSAT-PCI-U-2 timing boards (for a PCI slot on the machine).

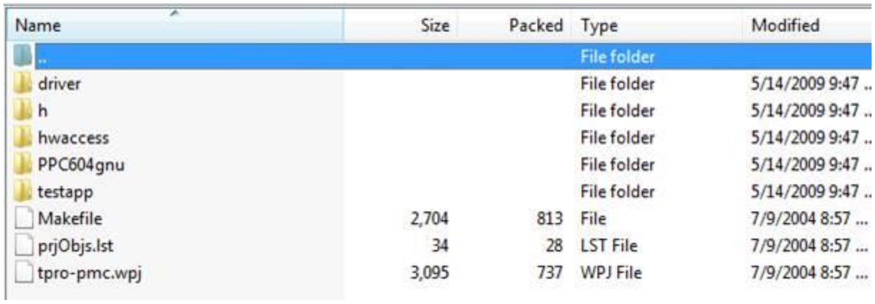
**VxWorks driver

- **Drivers are located in:** I:\New Released\Media (1152-0001-6000)
- **Note**: For TPRO/TSAT-PMC (not cPCI) need to edit the tpro.c file in the VxWorks driver to change the device ID from "7300" (cPCI board) to "7100" (PMC board) and them re-compile the driver (details below):

.c and .h files for VxWorks driver

Email Keith sent to Sylvain after working with Tim T,
I just received some clarification from Tim Tetreault about their request for the .c files. As I was already aware, we include the source code (including the .c and .h files) in the cPCI-PMC board's VxWorks driver they can download from our website.: http://spectracom.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=24

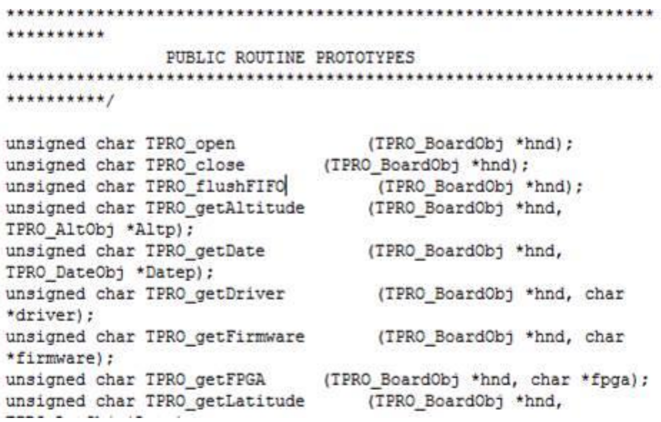
After they download the "**TPRO-TSAT cPCI-PMC VxWorks Driver v1.00"** file on this page, it includes the following items:



Name	Size	Packed	Type	Modified
..			File folder	
driver			File folder	5/14/2009 9:47 ..
h			File folder	5/14/2009 9:47 ..
hwaccess			File folder	5/14/2009 9:47 ..
PPC604gnu			File folder	5/14/2009 9:47 ..
testapp			File folder	5/14/2009 9:47 ..
Makefile	2,704	813	File	7/9/2004 8:57 ...
prjObjs.lst	34	28	LST File	7/9/2004 8:57 ...
tpro-pmc.wpj	3,095	737	WPJ File	7/9/2004 8:57 ...

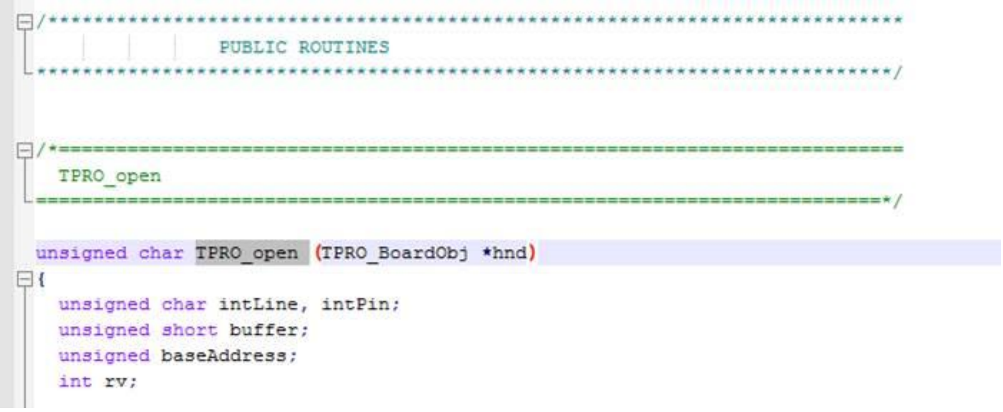
The "driver" folder contains the .c file ("tpro.c") and the "h" folder contains the associated header files (tpro.h).

In the tpro.h file is a section called "**public routine prototypes**" (partial screenshot below). These are a list of the available calls in the tpro.c file. They can use this list in the tpro.h file to find the applicable calls in the tpro.c file



```
*******************************************************************
**********
                PUBLIC ROUTINE PROTOTYPES
*******************************************************************
**********/

unsigned char TPRO_open             (TPRO_BoardObj *hnd);
unsigned char TPRO_close        (TPRO_BoardObj *hnd);
unsigned char TPRO_flushFIFO             (TPRO_BoardObj *hnd);
unsigned char TPRO_getAltitude      (TPRO_BoardObj *hnd,
TPRO_AltObj *Altp);
unsigned char TPRO_getDate          (TPRO_BoardObj *hnd,
TPRO_DateObj *Datep);
unsigned char TPRO_getDriver         (TPRO_BoardObj *hnd, char
*driver);
unsigned char TPRO_getFirmware       (TPRO_BoardObj *hnd, char
*firmware);
unsigned char TPRO_getFPGA      (TPRO_BoardObj *hnd, char *fpga);
unsigned char TPRO_getLatitude      (TPRO_BoardObj *hnd,
```

For example, from the screenshot above. If they want to know how to open the board (it's important to know how to do this in order to talk to the board), this is the first item in the list. they can search for "TPRO_open" in the tpro.c file. This will provide them with the following:



```
/***************************************************************************
                PUBLIC ROUTINES
***************************************************************************/


/*=========================================================================
  TPRO_open
=========================================================================*/

unsigned char TPRO_open (TPRO_BoardObj *hnd)
{
  unsigned char intLine, intPin;
  unsigned short buffer;
  unsigned baseAddress;
  int rv;
```

KW - On 10/1/08, Kent Brinkley from Ophir found our VxWorks driver for PMC card only references the subsystem ID for a TSAT-CPCI card (7100) and not the required ID for a TSAT-PMC card (7300). The driver was supposed to support both the TSAT-PMC and TSAT-CPCI cards, but was only written to support the TSAT-CPCI card). This prevents the GPS satellite information from being able to be retrieved when using our driver with a PMC card. The customer should be able to find/replace the "7100" values located in the TPRO.C driver file with the value of "7300" instead. Then they need to re-compile the driver. We are not intending to update this here because we can't recompile VXWorks (the driver has to be out-sourced to be fixed).

Q. What version(s) of VxWorks is supported?
A. The cPCI driver is compatible with VXWorks versions to 5.5.1.

****VxWorks versions beyond what the Spectracom driver currently supports**

The VxWorks driver includes source code, so a customer can build their own custom driver for supporting newer versions of Vxworks

**PowerPC

Email from Dave Sohn 8/17/12
The VxWorks driver we currently have is available for download from our website, including the source code. The VxWorks driver supports a PowerPC architecture, but not up to VxWorks 6.6.

FAQs
Email from Tim Tetreault to Sam Otto (3 July 2013)
Guys,
This is how I would reply to the customer:

Does the manufacturer of the card (Spectracom) provide support for using the card with VxWorks beyond just providing a VxWorks driver? Symmetricom (the BC635 manufacturer) provided a VxWorks driver for their BC635 card as well, but they did not provide support for using the BC635 with VxWorks. Our experience with Symmetricom was akin to, “You’re using VxWorks? Here’s a VxWorks driver. Good luck.” Additionally, the VxWorks driver Symmetricom provided had 3rd-party licensing restrictions that would have complicated our using it. That leads to the next 2 questions.
Our support is very limited when it comes to VxWorks drivers. The tools required to support the driver are very expensive and we don’t have many customers that use it. We do supply the source for the driver so the customer can modify it to fit their applications if they choose to.

Q. Who owns the license for the supplied VxWorks driver?
A. Spectracom owns the license for the supplied VxWorks driver.

Q. What are the restrictions placed on how we can use that driver?
A. There are no restrictions that we are aware of with our VxWorks driver.

**Windows driver

- **Drivers are located in:** I:\New Released\Media (1152-0001-6000)

Desire for 32/64 bit signed Windows drivers

Q What is the plan/timeline for 32/64 Win10 signed drivers, and are the PCI-U2 cards planned for win10 or only 66U?
A Reply from Dave Lorah (10 Sept 15) At this time Spectracom has no plans to update the TPRO/TSAT –U2 or -66U Drivers. We are working on the TSync-PCIe drivers but no updates are planned for the older platforms.

Driver guide

Windows driver versus Windows NT driver.

There are two different Windows drivers:
Slightly modified Email from Tim Tetreault (9/25/12)
Windows NT driver only supports Windows NT. The other Windows driver supports all the same versions as the PCI or TSync Windows driver except that it can only work on 32-bit systems. It does not support 64 bit.

32 bit / 64 bit Operating Systems
(8/2/11. Modified email from Tim T)
Keith

Windows 7 can be 32bit or 64bit. Our current driver will only support 32bit **(The driver is not digitially-signed,as required for 64 bit install)**. I am not aware of any plans to update the driver to support 64bit. Do we know if they would pay for an update? Dick has one customer that is willing to pay for an update to our VxWorks driver for the PMC/cPCI boards.

Update to this (per Tim Tetreault, 5 May 2013) The driver is still only 32 bit compatible. There are still no plans to release a 64 bit version of the driver. Instead, we plan on releasing a TSync-cPCI timing board later in 2013.

Windows Driver install

Email DL sent on 12/21/10
Installing the driver from a flash drive should not cause any problems. Make sure you have transferred all the files from the CD, though.

It sounds like your TPRO-PMC board is not being recognized by the computer. After the driver is installed the PC must be rebooted or instructed to find new hardware so the driver will attach to the PMC board.

If you open your computers Device Manager you should see TIMING BOARDS listed and the TPRO-PMC board listed there. Select the appropriate board from the list. If you change from a TPRO-PMC to a TSAT-PMC they are different so you will need to reboot or scan for hardware changes using device manager. Then the driver will recognize the board and will be able to open it and communicate.

If the board still will not open make sure it is supplied with the required power of +12V, -12V and 5 V. I know we discussed running the board from 3.3.V last week so I wanted to remind you of this.

****Clock Daemon program

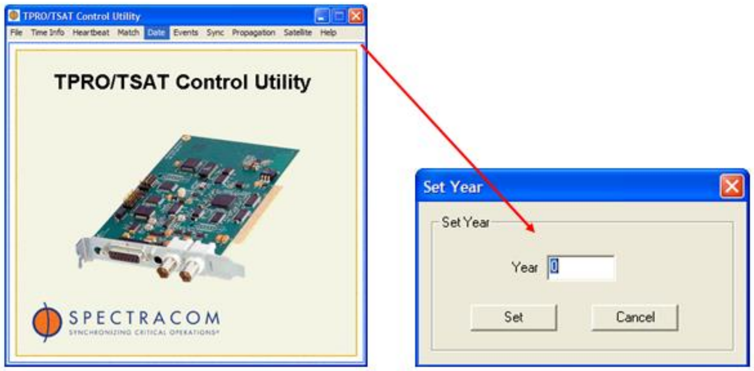
Refer to (in this document): Clock Daemon / Clock Daemon service (for all timing boards)

Email Keith sent to a customer 21 May 2013
I have some information for you that I hope will help!

To begin, the Windows driver for the TPRO/TSAT-PMC timing board contains two clock daemons and either one can be used to sync a Windows PC to a **synchronized** timing board. One is called the Clock daemon program and the other is the Clock daemon Service (unlike the Clock program, the Clock Service can automatically run at Windows startup).

In order for the Clock daemon to be able to set the Windows time, the TPRO/TSAT-PMC board needs to be synced to an IRIG generator (as indicated the green LED on the edge of the board being lit or by using the **TPRO_SynchStatus** API call . Also, as the TPRO/TSAT-PMC board can't obtain year information from the IRIG generator, the year must also be set after each boot-up of the timing board, before the Clock Daemon can sync the PC. Note that if you have a TSAT timing board, the year is automatically set when the board is synced to GPS.

The current year can be manually set in the TPRO timing boards (or the TSAT timing board when only using IRIG input instead of GPS input) using the Windows Control Utility (also installed as part of the Windows driver) - in the **Date** -> **Set Year** menu (the year can be read back via the **Date** -> **Retrieve Gregorian Date** menu). Or, the year can also be set using the API call of **TPRO_SetYear** (this command can be run in application software to automatically set the year).


TPRO/TSAT Control Utility
File Time Info Heartbeat Match Date Events Sync Propagation Satellite Help
TPRO/TSAT Control Utility
SPECTRACOM
SYNCHRONIZING CRITICAL OPERATIONS
Set Year
Set Year
Year
Set
Cancel

With the TPRO/TSAT-PMC timing board synced to the IRIG generator and with the year set using either of the methods mentioned above, the Clock Daemon program or Service can be used to periodically sync the Windows PC. Note that both Clock Daemons can be accessed via **Start / All Programs / Spectracom Corp / PCI**.

**RTX driver

Email from Tim Tetreault (2/29/12)
We do not have a RTX driver for the cPCI board. We can supply them with the source code for our Linux driver but they would have to write the RTX driver.

FUNCTIONS

**Time reads/getTime calls (such as HW_Gettime)

- The Gettime API/Example program can read the time of the board with 1 microsecond resolution.
- Reports days, hours, minutes, seconds, and sub-seconds (down to 1 microsecond resolution). The seconds are floating point numeric.
- Unlike all other calls, the HW_GetTime call doesn't use the FIFO buffer (CS get times and get dates all use the FIFO buffer).

Q. Set time & Read time with zero latency?
A. From Tim Tetreault (2/9/12)
The time reading is zero latency but you still have the OS latencies that are still present.

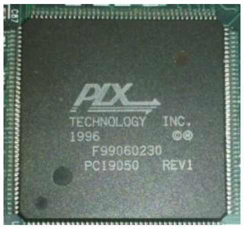
Q Is the TPRO-PMC designated End of Life?
~~No –it is still in the current catalog, but it may become obsolete in the future~~ YES!

Known issues with both cPCI and PMC boards

- Bus interface issue/ machine won't even boot

IC "U21" (PLX bridge chip) may be too old for the newer cPCI buses. Darron Nielson with Lawrence Livermore Nat labs purchased a second batch of boards that had an older PLX chip than the first batch of boards that were initially purchased. The new ones prevented the machines from booting up.

The “working” (newer rev) PLX has an indication of “PC10950-1” (non-rohs version) or the even newer ROHS version of “**PC10950-1F**”. The PLX on the non-working (earlier rev) boards was annotated as “**PC10950**”.   Below is a picture of an earlier rev of the PLX that may not be compatible with all machines (the “F…” number is a batch code, not a rev or build date).



Another email from Tim T:
Keith,
Be aware that this part is also used on the PMC board so it is possible that those customers could see a similar problem as the cPCI boards. I am working with Jen to go through our stock and purge those IC. The boards with the problem have a date code of “1996”.

- BLUE Key indicator for 5v only systems not installed for a long period of time

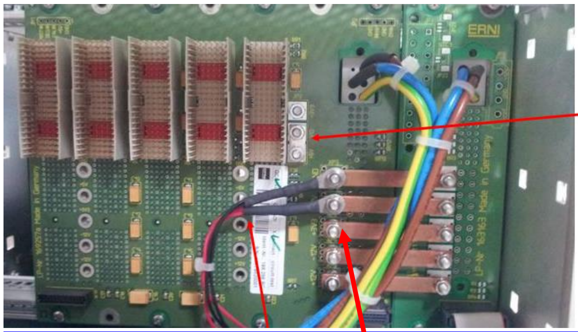
cPCI boards shipping between 2009 time-frame and March/Apr 2011 did not have the blue key installed to indicate these boards are 5v only boards (not compatible with 3.3v systems)

Email from Tim T: The part number for the cPCI key is in Visual: KSI1-0041-1698.
I went and looked at the last F000 BOM and the KEY wasn’t on it. I did a BOM conversion in OCT 2009 for the new “KSI1-xxxx-xxxx” part numbers and must have missed it.

The original BOM had the KEY so only the boards that were built from the Rev C F000 would be missing the KEY. It was released under ECN 2385.

Kontron chassis for Dongjin in Korea (and possibly others)

- The KONTRON chassis needs to be configured for 5vdc operation (the other jumper setting is 3.3vdc operation). There is likely a metal jumper in the chassis for this configuration, as shown in the photo below of the backplane.

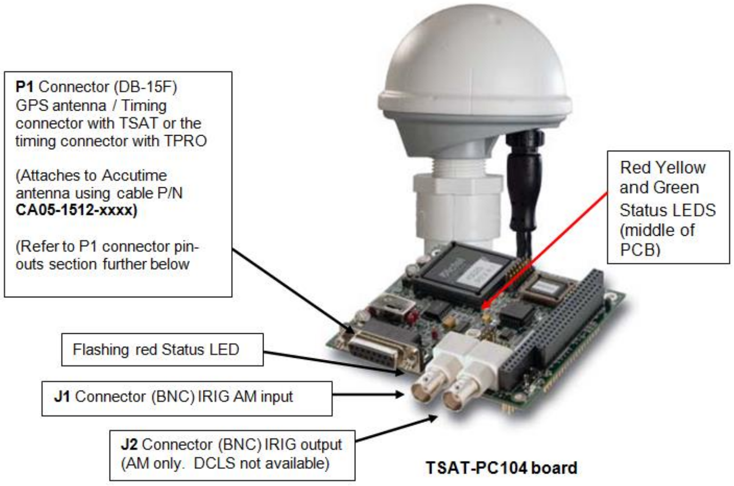


The metal jumper is configured for 5vdc operation (if it was across the other terminal, it would be configured for 3.3vdc).

- With the TPRO/TSAT-cPCI board installed and the chassis powered-up, make sure all rail voltages are present on the back-plane. In this particular Kontron backplane, there are 5 test point holes in the board for verifying voltages are present

- The Single Board Computer (the device that talks to the cPCI board) installed in the same Kontron chassis needs to be ordered for 5vdc operation (it can’t be reconfigured between 3.3 and 5vdc operation. Make sure this module was purchased for 5vdc operation. Otherwise, the SBC won’t be able to communicate with the timing board.

TPRO-PC104/TSAT-PC104 (PC-104)


P1 Connector (DB-15F)
GPS antenna / Timing connector with TSAT or the timing connector with TPRO
(Attaches to Accutime antenna using cable P/N CA05-1512-xxxx)
(Refer to P1 connector pin-outs section further below
Red Yellow and Green Status LEDS (middle of PCB)
Flashing red Status LED
J1 Connector (BNC) IRIG AM input
J2 Connector (BNC) IRIG output (AM only. DCLS not available)

TSAT-PC104 board

Link to PC-104 data sheet (TSAT-PC104 and TPRO-PC104)**:** I:\Marketing_Product Data Sheets (archive)\Bus-Level Timing Boards\TSAT-PC104_revD.pdf

Link to TPRO/TSAT-PC104 user manual: I:\New Released\Manuals\1155-xxxx-xxxx (1155-5001-0050)

Link to Windows/Linux driver guides: I:\New Released\Manuals\1155-xxxx-xxxx
(1155-5002-0050 for Windows)
(1155-5003-0050 for Linux)

Link to TPRO/TSAT-PC104 drivers: I:\New Released\Media (1155-0001-6000)

Link to TPRO/TSAT-PC104 in Customer Service: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PCI and_PC104 VME

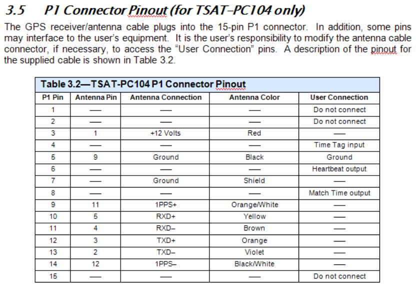
PCB documentation / schematics (1155-0002): I:\New Released\PCB Documentation

---

Certifications (CE, FCC, etc)

CE Declaration of conformity
As of at least October 2012, the TPRO/TSAT-cPCI, TPRO/TSAT-PMC, TPRO/TSAT-PC104 and TPRO/TSAT-VME timing boards are not CE approved and therefore do not have an available CE Declaration of conformity.

Connector P1 (DB15 connector) pin-outs



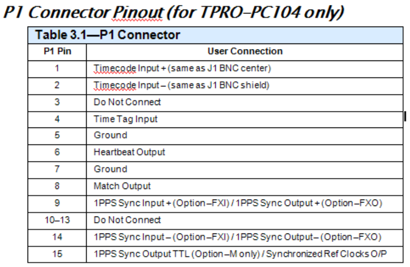
3.5 P1 Connector Pinout (for TSAT–PC104 only)

The GPS receiver/antenna cable plugs into the 15-pin P1 connector. In addition, some pins may interface to the user's equipment. It is the user's responsibility to modify the antenna cable connector, if necessary, to access the "User Connection" pins. A description of the pinout for the supplied cable is shown in Table 3.2.

Table 3.2—TSAT-PC104 P1 Connector Pinout

P1 Pin	Antenna Pin	Antenna Connection	Antenna Color	User Connection
1	—	—	—	Do not connect
2	—	—	—	Do not connect
3	1	+12 Volts	Red	—
4	—	—	—	Time Tag input
5	9	Ground	Black	Ground
6	—	—	—	Heartbeat output
7	—	Ground	Shield	—
8	—	—	—	Match Time output
9	11	1PPS+	Orange/White	—
10	5	RXD+	Yellow	—
11	4	RXD–	Brown	—
12	3	TXD+	Orange	—
13	2	TXD–	Violet	—
14	12	1PPS–	Black/White	—
15	—	—	—	Do not connect

TSAT-PC104 board



P1 Connector Pinout (for TPRO–PC104 only)

Table 3.1—P1 Connector

P1 Pin	User Connection
1	Timecode Input + (same as J1 BNC center)
2	Timecode Input – (same as J1 BNC shield)
3	Do Not Connect
4	Time Tag Input
5	Ground
6	Heartbeat Output
7	Ground
8	Match Output
9	1PPS Sync Input + (Option –FXI) / 1PPS Sync Output + (Option –FXO)
10–13	Do Not Connect
14	1PPS Sync Input – (Option –FXI) / 1PPS Sync Output – (Option –FXO)
15	1PPS Sync Output TTL (Option –M only) / Synchronized Ref Clocks O/P

TPRO-PC104 board

(Email from Tim Tetreault regarding P1 connector, pin 15, 8/8/12)
Here is the answer to the DB15, pin 15 question. After looking at the manual, I see there is a typo in the labeling. It should read:
1PPS Sync IN TTL (Option –M only) / Synchronized Ref Clocks O/P

This pin can be used for 2 functions:

- Synchronized Ref Clocks O/P: The standard function is as a Sync Ref Clocks output that is a TTL signal. The clocks are selected by using the jumper JP1. (Refer to section 3.3 in the manual for frequencies that are available). **Note**: pasted below:

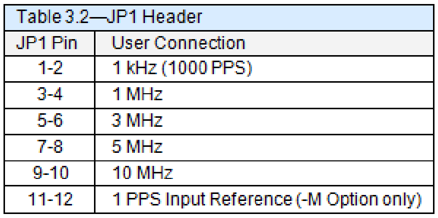


Table 3.2—JP1 Header

JP1 Pin	User Connection
1-2	1 kHz (1000 PPS)
3-4	1 MHz
5-6	3 MHz
7-8	5 MHz
9-10	10 MHz
11-12	1 PPS Input Reference (-M Option only)

Breakout cables

- We don't ship a breakout cable with PC104 boards- just the connector/hood to connect to the connector on the edge of the board.

Note (19 Nov 2012 KW): Section 11-2 "Accessories" of the PC-104 manual (Rev B) discusses two different breakout cable assemblies for the TPRO boards (One for standard boards and one for Option –M boards).

Apparently, these two assemblies don't exist. The same ancillary kit is included with all PC104 boards and the kit includes the bare connector and hood to connect to the timing connector. But apparently we don't offer any breakout cables.

Power requirements / fuses/ board reset

- DC power needs to be provided by the PC (via the bus connector)

Input Power specs / power glitches

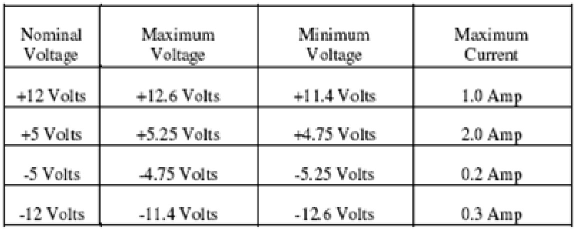
- The Timing boards are designed to meet the PC104 bus specifications, such as the min and max power specs.

Email from Tim Tetreault (10 Jan 2013) It is possible that glitches on the +5v supply could cause the board to reset. If the supply rails conform to the PC104 spec's, there should be no problems.

The current spec's for PC/104 power requirements can be found in section 4.8 of the spec's (at websites such as the following http://www.versalogic.com/support/pdf/pc104-23.pdf)

Note from Keith about the above link: I don't know if this link it to the current version of the PC104 specs, but it has the same table Tim had sent to me about this. These are the specs that we design to.

Table 1: Module Power Requirements



Nominal Voltage	Maximum Voltage	Minimum Voltage	Maximum Current
+12 Volts	+12.6 Volts	+11.4 Volts	1.0 Amp
+5 Volts	+5.25 Volts	+4.75 Volts	2.0 Amp
-5 Volts	-4.75 Volts	-5.25 Volts	0.2 Amp
-12 Volts	-11.4 Volts	-12.6 Volts	0.3 Amp

Power specs below are from our PC104 manual

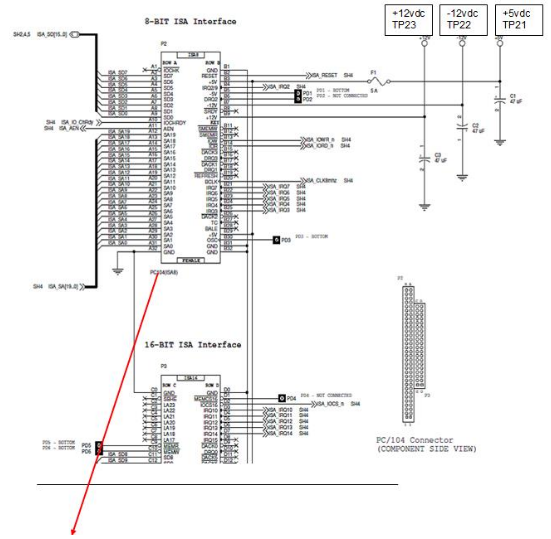
Power (from ISA bus):   Test Point on board
+5 VDC @ 0.7 mA max   (TP21)
+12 VDC @ 175 mA max   (TP23)   (+12vdc is used for IRIG input and GPS antenna on TSATs)
–12 VDC @ 20 mA max   (TP22)   (-12vdc is only used for IRIG input and IRIG output)

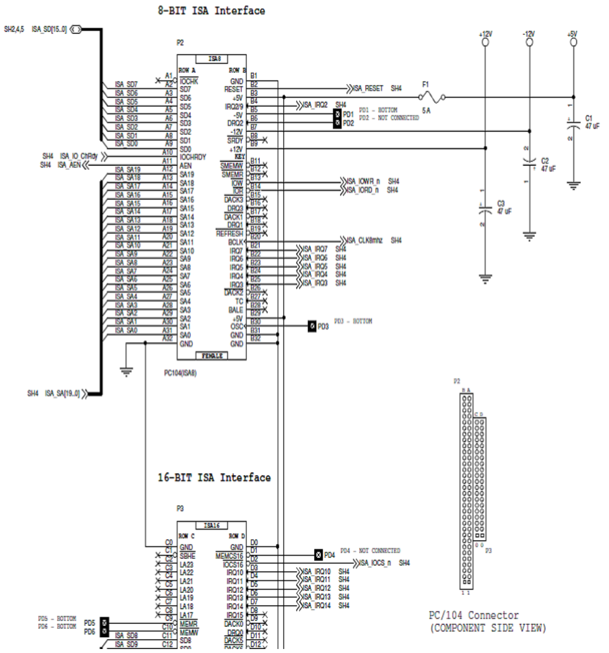
Ground (TP19 and TP20)

Power Fuses

F1 (5A) for +5vdc input from PC104 bus
F2 (0.75A) for +12vdc to Timing connector pin 3

+5, -12v and +12vdc power connector pins for PC104 bus:





Board reset due to lower voltage input/noise

(Salesforce case 7219) Partial email from Conrad Seymour (regarding the +5vdc line) "…and my board does reset it the +5V drops below +3.9V, for even nanoseconds"

Conrad's findings (13 Jan 2013) I believe, we were finally were able to the source of our problem were you help. We measured our +12V supply at +12.74V. It turns out that the TPRO is very sensitive to noise when the +12V supply if higher than 12.6V. It is the combination of the high +12V and low level noise on the supplies that was causing the time to reset. The length of time to reset was statistical, but related to these factors. Once we decreased the +12V to 12.55V, we no longer saw the time resetting in our system. We are not sure of the exact relationship between noise sensitivity and +12V setting, but we plan on keeping the 12V supply below 12.5V in the future.

If -12vdc is not provided by the motherboard:

- **TSAT**- If the motherboard does not provide -12vdc, can't sync to IRIG input and can't use IRIG output. If using GPS for sync and not using he IRIG output, not having -12vdc isn't a problem as no other operation besides IRIG in/out is affected.
- **TPRO**- If the motherboard does not provide -12vdc, can't sync to IRIG input and can't use IRIG output.   So, the board can't even be synced, even if it's not desired to use the IRIG output. So, -12vdc is needed when using a TPRO board. If there is no way to get -12vdc and IRIG output is not needed, switch to a TSAT board, as GPS does not need- 12vdc for input sync.

**Desire to power-up the board in a stand-alone configuration (not installed in a system)

- Need to provide +12vdc, -12vdc and +5vdc
- Looks like also need to provide a reset input, after boot-up (Darryl Dugan reported the status LED is not illuminating after powering up with the necessary rails).

Email Dave sent to Darry about the reset requirement (1 May 13) I I was looking at the reset on the schematics and found an ISA Reset on pin B2 of the PC014 connector. If you short B2 to B1 (GND) it should give the board a reset. Do this after power up and see if the status LED on the front panel starts flashing. Attached is a ISA bus diagram.

Then if the board is given a IRIG B 122 signal, the status LED should flash a sequence of Long, Short,Long,Long,Long,Pause flashes.

Darryl's reply to this: As advised I installed a permanent short on B1-B2 on P2 connector and all is well. The card powers up and immediately synchronizes to in input IRIG time.

Red, Yellow and Green Status LEDs (in center of the board)

- There are also three "centrally-located" LEDs (Near the middle of the board) that should also be flashing when power is applied.
- Stated in manual as being used for "Factory Test".

From page 12, table 2.8 of the attached User Manual:

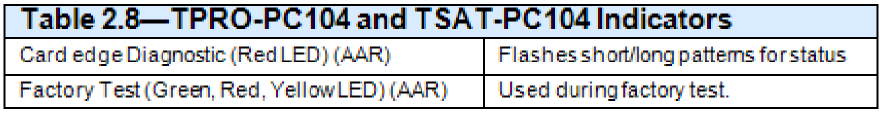


Table 2.8—TPRO-PC104 and TSAT-PC104 Indicators

Card edge Diagnostic (Red LED) (AAR)	Flashes short/long patterns for status
Factory Test (Green, Red, Yellow LED) (AAR)	Used during factory test.

Q. We currently have several boards in house. Most of the time the three board LED alternately flash with the green and yellow on state alternating with the red LED. On one of our boards, these LED are on solid all of the time. Is this something we should be concerned about? The board seems to function correctly otherwise.

A. Email from Tim Tetreault (4/27/12)

Those LED's are only intended for factory test so they shouldn't be concerned about them as long as the board is working. We indicate this in the user manual on page 12, Table 2.8.

Email Keith sent to Frederic
As mentioned in table 2-8 of the manual, there are also three "centrally-located" red, yellow and green LEDs (Near the middle of the board) that should also be flashing when power is applied. These LEDs are for factory test purposes only. With power applied to the timing boards, these three LEDS will continuously alternate between both the green and yellow LEDs being illuminated together, and then the red status LED by itself will illuminate.

TPRO/TSAT-PC104 timing board does not seem like it's installed (after installing driver)

The board may not be running, or the Interrupt configuration (DIP switch SW2) may not match the value the BIOS is assigning to the board.

- Make sure the timing board is displayed in Control Panel/ Device Manager after driver installed:

With the installation of the TSAT-PC104 driver, there should now be a new timing board ICON present in the Control Panel/ Device Manager. If the driver installed successfully, this ICON should be present. If there were any problems with the driver install, the ICON should be a question mark "?". If the driver wasn't able to install at all (such as a potential interrupt issue), there won't be a new ICON present for the newly installed timing board.

- Make sure Status LED is flashing. This indicates +5Vdc is supplied by the ISA bus and that the board is running.

The first thing that needs to be verified is that the Status LED on the edge of the TSAT board, between the Timing connector and the BNC connector (not the three little LEDs in the middle of the board- the ones that look like "airport runway lights") is flashing a pattern of blinking/flashing. If this Status LED on the edge of the board is not flashing a pattern, let me know.

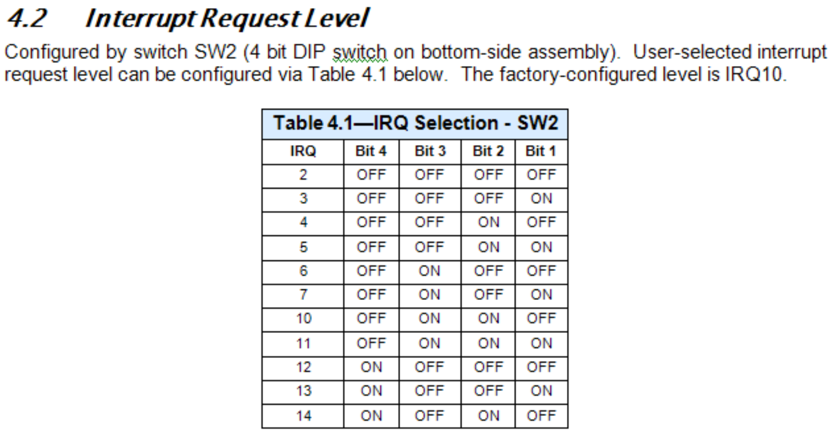
- Make sure all socketed ICs are fully seated on the board

Especially if the Status LED on the TSAT board is not flashing, please also make sure that all of the socketed ICs on the TSAT board are fully seated in their sockets. This can be checked by putting a slight amount of pressure on the center of the ICs. It is possible for a socketed IC to become partially lifted out of its socket during shipping.

If any of the socketed ICs feel like they may have been partially lifted, reinstall the TSAT board and check its operation again.

- Make sure the Interrupt of the PCs BIOS matches the settings of the board, as configured with SW2 on the board.

If the Status LED is flashing a pattern, next it's important to verify that the Interrupt that is assigned to the TSAT board in the machine's BIOS is the same interrupt number that the TSAT board is configured to use. The TSAT board's Interrupt Request Level (as described in Section 4.2 on page 4-1 of the PC104 user manual - pasted below for your reference), needs to be set to the same value that the BIOS is configured for the TSAT board to use. DIP Switch "SW2" on the TSAT board is used to configure this value, with IRQ 10 being the factory default value (For IRQ 10, the switch settings of DIP Switch SW2, the order left to right should be as follows: OFF, ON, ON, OFF). If they don't currently match, you can either manually change the BIOS value or reconfigure the TSAT board to use the one that the BIOS is assigning to the board.



4.2 Interrupt Request Level

Configured by switch SW2 (4 bit DIP switch on bottom-side assembly). User-selected interrupt request level can be configured via Table 4.1 below. The factory-configured level is IRQ10.

Table 4.1—IRQ Selection - SW2

IRQ	Bit 4	Bit 3	Bit 2	Bit 1
2	OFF	OFF	OFF	OFF
3	OFF	OFF	OFF	ON
4	OFF	OFF	ON	OFF
5	OFF	OFF	ON	ON
6	OFF	ON	OFF	OFF
7	OFF	ON	OFF	ON
10	OFF	ON	ON	OFF
11	OFF	ON	ON	ON
12	ON	OFF	OFF	OFF
13	ON	OFF	OFF	ON
14	ON	OFF	ON	OFF

Specifically, the IRQ Resources in BIOS should indicate the assigned IRQ value of SW2 is set to “Legacy ISA” (not PNP). This interrupt needs to be dedicated to the TPRO board (it can’t be a shared interrupt).

Please let me know if the configured IRQ value of the board didn’t match the BIOS setting, or if the Interrupt BIOS configuration wasn’t configured as a dedicated interrupt. This could definitely result in abnormal operation, if it’s not configured correctly.

**PC104 Firmware versions / Driver versions

Firmware info

Determining current firmware version installed

Windows

Note: There is an issue with the Widows Control Utility reading the Firmware version of a PC 104 board. The version 2.00 firmware supports this, so issue is with either the Windows driver or with the Windows Control Utility itself. Refer to the n

Linux

May be able to use the **TPRO_GetFirmware API** call (We’re not sure if it will work).

Note: The BEST way to confirm PC104 firmware version for both Windows and **Linux is to** look at the label affixed to IC “U16”. It will indirectly indicate firmware version (for example 1155-SE01-0200 – where the last four digits indicates the version, in this case, firmware version 2.00).

Firmware version changes

Refer to: PSB, PSP software updates\TPRO-TSAT boards\board firmware

A) Version 2.10

- ECN 3283 (Aug 2013)
- Adds support for newer Acutime GG (GNSS) Smart Antenna (our P/N E025R-0004-0003)

F) Version 2.0.0

- ECN 2569 (Jan 2011)
- Fixes a year rollover issue

G) Version 1.0.0

Driver info (Windows and Linux)

Determining current driver version

- Use the Windows Control Utility, if board is installed in Windows PC
- Use the TPRO_GetDriver API call

Driver versions

- **Link to TPRO/TSAT-PC104 drivers:** I:\New Released\Media (1155-0001-6000)
- **Info on versions- Refer to:** PSB, PSP software updates\TPRO-TSAT boards\Driver updates

---

Memory / DMA (Direct Memory Access) / Memory Controllers

BAR Memory

ECN 3093 (Nov 2012) is releasing new FPGA firmware for the PCI-U-2 and PCI-66U boards, which adds support for 8k BAR memory.   (**Note**: It also updates the class code for the boards to a more specific value)

This is a special firmware version that is being created for China Lake. This change increases the amount of system memory allocated to the timing board when its installed (from 64 bytes (?) to 8k).

Note: Per Tim Tetreault, this function is recommended for Linux OS only – It's NOT recommended for use with Windows OS (he ran into problems with the board installed in a Windows PC).

DMA (Direct Memory Access

(KW 2/7/12) Per Tim Tetreault: "The PC-104 does not require DMA".

Answers below (in red) are from Tim Tetreault on 6/23/11

- DMA is not supported. Therefore, AEN signal is set to a constant logical zero

Not a problem

- Bus mastering is not supported. Therefore, do not connect any other master add-on board to the TITAN PC/104 interface

Not a problem

- Shared interrupts are not supported. Therefore, do not connect more than one add-on board to the same interrupt signal line

Our PC104 board does not support shared interrupts so this should not be a problem

- BALE signal is set to a constant logical one as the address is valid over the entire bus cycle. Only add-on PC/104 boards that implement transparent latch on address lines LA17-LA23 are compatible with the TITAN implementation of BALE.

This should not a problem

- The PXA270 PCMCIA memory controller does not support 8-bit memory read accesses for common memory space

We don't think this should be a problem

- The PXA270 PCMCIA memory controller does not support PC/104 MEMCS# signal, so there is no support for dynamic bus sizing.

Not a problem

- The address space on our master is different than on x86 platforms.
- The PC/104 I/O space is from 0x30000000 to 0x3000003FF, supporting 8bit and 16bit reads and writes
- The PC/104 memory space is from 0x3C000000 to 0x3C1FFFFF, supporting 16 bit reads and writes, but 8 bit writes only.

It looks like these addresses can be easily changed in the driver source, so this is likely not an issue.

Replacement as a TSync board

Important Note: The original TPRO/TSAT-PC104 board (ISA Bus) is becoming a "PCI104" bus (PCI bus) card when its replaced as a TSync board (instead of remaining as a "PC104" board). This is not the same form factor as the legacy timing boards.

Operation / Not operational / Abnormal operation

TPRO/TSAT-PC104 timing board does not seem like it's installed (after installing driver) or it is not operating normally (such as intermittently resetting)
The board may not be running, or the Interrupt configuration (DIP switch SW2) may not match the value the BIOS is assigning to the board.

1) **Make sure the timing board is displayed in Control Panel/ Device Manager after driver installed:**

   With the installation of the TSAT-PC104 driver, there should now be a new timing board ICON present in the Control Panel/ Device Manager. If the driver installed successfully, this ICON should be present. If there were any problems with the driver install, the ICON should be a question mark "?". If the driver wasn't able to install at all (such as a potential interrupt issue), there won't be a new ICON present for the newly installed timing board.

2) **Make sure Status LED is flashing. This indicates +5Vdc is supplied by the ISA bus and that the board is running.**

The first thing that needs to be verified is that the Status LED on the edge of the TSAT board, between the Timing connect and the BNC connector (not the three little LEDs in the middle of the board- the ones that look like “airport runway lights”) is flashing a pattern of blinking/flashing. If this Status LED on the edge of the board is not flashing a pattern, let me know.

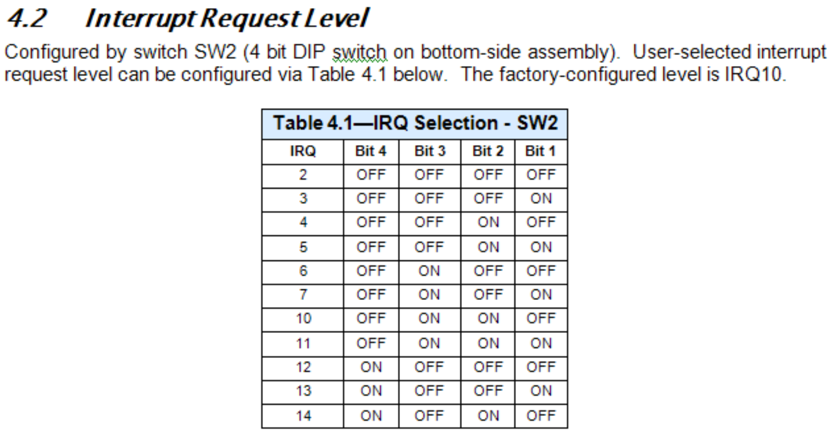
3) **Make sure all socketed ICs are fully seated on the board**

Especially if the Status LED on the TSAT board is not flashing, please also make sure that all of the socketed ICs on the TSAT board are fully seated in their sockets. This can be checked by putting a slight amount of pressure on the center of the ICs. It is possible for a socketed IC to become partially lifted out of its socket during shipping.

If any of the socketed ICs feel like they may have been partially lifted, reinstall the TSAT board and check its operation again.

4) **Make sure the Interrupt of the PCs BIOS matches the settings of the board, as configured with SW2 on the board.**

If the Status LED is flashing a pattern, next it’s important to verify that the Interrupt that is assigned to the TSAT board in the machine’s BIOS is the same interrupt number that the TSAT board is configured to use. The TSAT board’s Interrupt Request Level (as described in Section 4.2 on page 4-1 of the PC104 user manual - pasted below for your reference), needs to be set to the same value that the BIOS is configured for the TSAT board to use. DIP Switch “SW2” on the TSAT board is used to configure this value, with IRQ 10 being the factory default value (For IRQ 10, the switch settings of DIP Switch SW2, the order left to right should be as follows: OFF, ON, ON, OFF). If they don’t currently match, you can either manually change the BIOS value or reconfigure the TSAT board to use the one that the BIOS is assigning to the board.



4.2 Interrupt Request Level

Configured by switch SW2 (4 bit DIP switch on bottom-side assembly). User-selected interrupt request level can be configured via Table 4.1 below. The factory-configured level is IRQ10.

Table 4.1—IRQ Selection - SW2

IRQ	Bit 4	Bit 3	Bit 2	Bit 1
2	OFF	OFF	OFF	OFF
3	OFF	OFF	OFF	ON
4	OFF	OFF	ON	OFF
5	OFF	OFF	ON	ON
6	OFF	ON	OFF	OFF
7	OFF	ON	OFF	ON
10	OFF	ON	ON	OFF
11	OFF	ON	ON	ON
12	ON	OFF	OFF	OFF
13	ON	OFF	OFF	ON
14	ON	OFF	ON	OFF

Specifically, the IRQ Resources in BIOS should indicate the assigned IRQ value of SW2 is set to “Legacy ISA” (not PNP). This interrupt needs to be dedicated to the TPRO board (it can’t be a shared interrupt).

Please let me know if the configured IRQ value of the board didn’t match the BIOS setting, or if the Interrupt BIOS configuration wasn’t configured as a dedicated interrupt. This could definitely result in abnormal operation, if it’s not configured correctly.

Inputs / Synchronization

****Default date/Time/Year at start-up.**

001:00:00:00.000000 (Jan 1 00:00:00, year 0000)

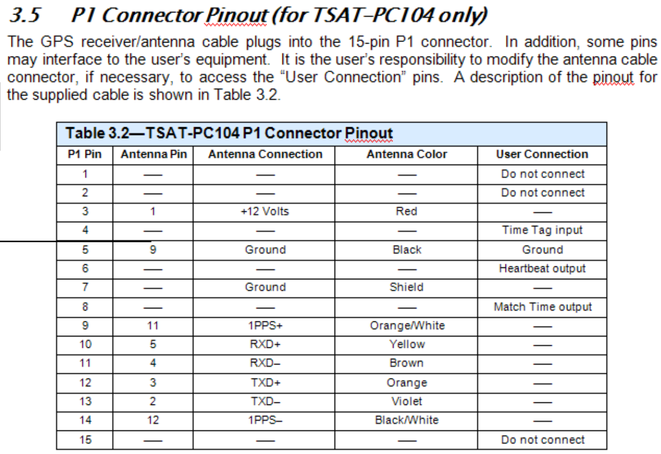
**GPS input (TSAT-PC104 boards only)

P1 Connector (DB-15F)
GPS antenna / Timing connector with TSAT or the timing connector with TPRO

(Attaches to Accutime antenna using cable P/N **CA05-1512-xxxx)**

(Refer to P1 connector pin-outs section further below

KSI Acutime antenna



3.5 P1 Connector Pinout (for TSAT–PC104 only)

The GPS receiver/antenna cable plugs into the 15-pin P1 connector. In addition, some pins may interface to the user's equipment. It is the user's responsibility to modify the antenna cable connector, if necessary, to access the "User Connection" pins. A description of the pinout for the supplied cable is shown in Table 3.2.

Table 3.2—TSAT-PC104 P1 Connector Pinout

P1 Pin	Antenna Pin	Antenna Connection	Antenna Color	User Connection
1	—	—	—	Do not connect
2	—	—	—	Do not connect
3	1	+12 Volts	Red	—
4	—	—	—	Time Tag input
5	9	Ground	Black	Ground
6	—	—	—	Heartbeat output
7	—	Ground	Shield	—
8	—	—	—	Match Time output
9	11	1PPS+	Orange/White	—
10	5	RXD+	Yellow	—
11	4	RXD–	Brown	—
12	3	TXD+	Orange	—
13	2	TXD–	Violet	—
14	12	1PPS–	Black/White	—
15	—	—	—	Do not connect

Using a newer Acutime GG antenna with a TPRO/TSAT-PCI-104 board

➢ Refer to "Acutime antennas" in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

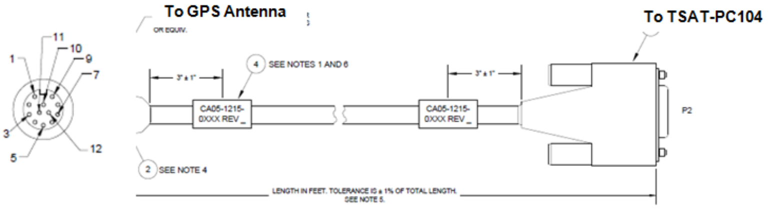
- Info below pertains to the Acutime GG antenna (**Our P/N**: E025R-0004-0003. **Trimble P/N** 55238-00).
- Info below is also in the Acutime GG section of the custserviceassistance document

A Firmware upgrade will be likely required for earlier timing boards to be compatible with the newer antenna. This newer Acutime antenna is compatible with:

- **TPRO/TSAT-PC104** boards with firmware version 2.11 or higher installed (Refer to ECN 3283, Aug 2013)

Antenna cabling

- TSAT-PC104 ships with a standard 100 ft antenna cable (**CA05-1512-0100**) to attach the antenna to the board



	SIGNAL	ITEM 2	ITEM 3	COLOR
PAIRED	+12V	1	3	RED
	GND	9	5	BLACK
PAIRED	UNUSED	10	NO CONNECT	BLUE
	BATTERY	8	NO CONNECT	GREEN
PAIRED	IPPS +	11	9	ORG/WHT
	IPPS -	12	14	BLK/WHT
PAIRED	DOWN +	5	10	YELLOW
	DOWN -	4	11	BROWN
PAIRED	UP +	3	12	ORANGE
	UP -	2	13	VIOLET
PAIRED	PORT A +	7	NO CONNECT	GRAY
	PORT A -	6	NO CONNECT	WHITE
FOIL AND DRAIN WIRE	SHIELD	—	SHELL	—

- 100 foot extender cable=
- **Max recommended cable length** (GPS antenna to TSAT-PC104 board) = 500 feet

---

GPS antenna (Acutime antenna) for TSAT-PC104 boards

- Refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

The Acutime antennas are programmed at the factory to operate with ONLY TSAT-PCI-U, TSAT-U-2, TSAT-66U, TSAT-PC104 and TSAT-PC boards. They will not operate with any other timing board. More recently shipped Acutime antennas will have a label applied which indicates what timing boards that it is programmed to operate with.

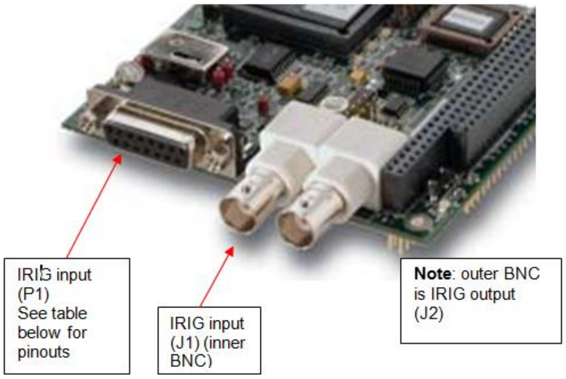
GPS antennas for TSAT-PCI-U, TSAT-U-2, TSAT-66U, **TSAT-PC104** and TSAT-PC boards should be annotated as: P/N **1159-0000-5000.**

Important Note: Do not connect the Acutime antenna to a TSync-PCIe board unless it’s for a permanent change, as the TSync-PCIe board will reprogram the antenna to operate with the TSync-PCIe board only. It won’t work with the TSAT-PC104 board again, without factory re-programming of the antenna.

For more info on Acutime antennas, refer to: Acutime GPS Antennas

****IRIG Input (J1 connector)**

- IRIG input automatically detects and accepts IRIG A, IRIG B and NASA36
- Can connect to either:
  - BNC connector (J1)
  - DB15 connector (P1)


IRIG input (P1) See table below for pinouts
IRIG input (J1) (inner BNC)
Note: outer BNC is IRIG output (J2)

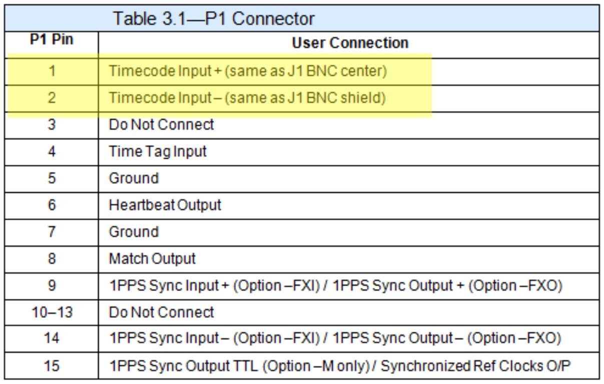


Table 3.1—P1 Connector

P1 Pin	User Connection
1	Timecode Input + (same as J1 BNC center)
2	Timecode Input – (same as J1 BNC shield)
3	Do Not Connect
4	Time Tag Input
5	Ground
6	Heartbeat Output
7	Ground
8	Match Output
9	1PPS Sync Input + (Option –FXI) / 1PPS Sync Output + (Option –FXO)
10–13	Do Not Connect
14	1PPS Sync Input – (Option –FXI) / 1PPS Sync Output – (Option –FXO)
15	1PPS Sync Output TTL (Option –M only) / Synchronized Ref Clocks O/P

In order for the PC104 to be able to read and Sync to IRIG input, the motherboard must supply +12vdc, -12vdc and +5vdc. The + and -12vdc are both required to be present for the IRIG input op amp to work correctly.

IRIG AM requirements: 1.2 to 8.0vp-p

Note: PC104 boards can accept/sync to IRIG A input, but cannot output IRIG A.

Oscillator Calibration and IRIG Input accuracy specs

- **IRIG accuracy to UTC**: Consists of the synchronization to the IRIG source PLUS the stability/accuracy of the IRIG source.
- **IRIG input** = Better than 100 ppm

TPRO/TSAT-PC104 Oscillator calibration procedure

- specified in the latest version of the TPRO and TSAT-PC104 test procedures (In Arena):
  - **TPRO-PC104 (1155-0002-0600-TP)**: https://app.bom.com/items/detail-spec?item_id=1202833233&version_id=10221203788&orb_msg_single_search_p=1
  - **TSAT-PC104 (1155-0001-0600-TP)**: https://app.bom.com/items/detail-spec?item_id=1202833231&version_id=10221289058&orb_msg_single_search_p=1
- Also see the info directly below (**Kown "temperature issue"**)

****Known "temperature issue" with IRIG input synchronization of PCI series and PC-104 boards**
An issue was found where the PC104 boards were having difficulty locking to IRIG input in higher temperature environments. The issue was found to be associated with a PLL circuit for IRIG synchronization. R46 was not adjusted correctly during factory test, causing the boards to lose IRIG sync when the temperature was over **50°C**.

Important Note: Some PC104 boards shipped without the PLL alignment performed duing factory test, while others still shipped even though they couldn't be correctly tuned during test (the resistor was needed but not installed. This issue only applies to these two conditions. It is not a lot/batch issue, or a design issue that requires a product recall. It will not normally be seen by customers, as most boards don't normally operate at this high of a temperature.

- **ECN 3012 (signed-off 17 July 2012):** Revised the PC104 test procedures to ensure R46 is adjusted correctly.
- **Temporary Deviation "ECN 3022" (started ~25 Sept 2012**) Fixes a tolerance issue with the PC104. Some boards are failing the 640kHz cal when pot doesn't have enough adjustment. This is due to tolerance issues with U8. As needed, when this test fails, R37 can be changed from a 75k to a 62.0k resistor. (R116R-0623-SJ0B)

Update info (March 2013 after talking to Tim Tetreault) There was another change to the test procedures after the resistor change from ECN 3022. Don't have a good cut-in for when the boards were "garunteed" to be properly cal'd, but boards sold before **Sept/Oct 2012 time-frame** are definitely susceptible to this issue.

ECN 3218 (~2 June 2013) (Per Tim Tetreault written in ECN) This is a deviation to fix a tolerance issue with the R30 calibration on the TPRO/TSAT PCI boards.

Changing R30 & R29 will give more adjustability for the 640KHz calibration without losing resolution. See Tim T. for parts to do the modifications. I will follow-up with an ECN to make this change permanent.

CORRECT procedure to calibrate the oscillator/ IRIG input (from Dave West, 8 March 2013)

- Oscillator alignment adjustments.
- Perform a warm reset on the Timing Board

NOTE: Ensure IRIG Source or GPS is not connected prior to these steps. IF UUT has started tracking satellites, or is in sync, adjustments will not be correct causing errors in the field. If in Sync or tracking, power cycle PC, remove source(s) and power back up before continuing. Repeat warm reset from previous step

- Using a probe connected to frequency counter;
- Measure OSC1 pin 8, and adjust R22 to 10MHz ±2 Hz.
- Measure TP8, and adjust R38 to 640kHz ± 2kHz.

Email Keith sent to Frederic (10/19/12) Regarding the suspected issue they appear to be observing, back around July 2012 (after this PC104 board was shipped), we discovered the PC104 boards were losing lock to the IRIG input signal, when operated in environmental temperatures above **50°C.** Specifically, there was an issue with the tuning of a potentiometer in the IRIG input PLL circuit. If this pot was not tuned correctly, the PLL may not remain locked at higher temperatures. In some case, a resistor may need to be added to the circuit to allow the PLL to be tuned to the correct voltage.
The deviation to start fixing this issue was implemented in Sept 2012. Therefore PC104 boards shipped prior to Sept 2012 are susceptible to this specific issue, when they are operated at temperatures above **50°C.**

Email Keith sent to Frederic (10/22/12)
First, to clarify my earlier report on the root cause of this issue, this issue is only experienced at higher than normal temperatures, and is only due to a procedural issue during factory test (which was recently discovered). This is not, in any way, a hardware (or software) issue with these boards.

During the factory test procedure, a potentiometer in the IRIG input PLL circuit needs to be tuned for a specific frequency (without the IRIG input being connected). This particular frequency will allow the PLL circuit to be able to remain locked to IRIG input, even at the higher operating temperature range. Periodically, a resistor needs to be installed in this PLL circuit, to allow the circuit to be tuned to the correct frequency. Due to an issue with the test procedure (the IRIG input was connected to the boards during this alignment) the PLL circuit wasn't being tuned correctly. So some of the boards may not remain locked to IRIG input at higher temperatures. The fix, in many cases is to simply tune this circuit for the correct frequency. In some cases, a resistor may also be needed, in order to obtain this desired frequency.

I just checked with our Production Manager to see if we happened to have any TPRO-PC104 boards in stock. She responded that we do not have any currently in stock, and won't be able to have any available until at least 28 November.

Based on this, I see two possible solutions for your customer. One is for your customer to simply try operating each of the five boards that they likely have in their possession. Any of these boards that don't remain locked at the higher temperatures they are operating them at can then be returned to us for the correct calibration process. Or, since the boards need to be powered-up in order to calibrate them properly, if your customer has a good frequency counter, they can likely calibrate them locally, instead of needing to return them to us. Essentially, the board needs to be powered-up without IRIG connected to perform this adjustment (it is very important that the IRIG input not be connected during this calibration). Then, without an IRIG input signal being applied, a **pot (R38)** needs to be turned to allow **TP8** to provide **640 kHz** output.

If they have access to a frequency county, the hardest part of this procedure may be just being able to access this test point and the potentiometer, while the board is installed in the system. If the board is installed in an open area of the system, this will likely be an easy process for them. However, if there are any other boards installed just above this board (and that board can't be temporarily to gain access to the pot) they may not be able to perform this procedure.

5vdc input and ambient temperature requirements for this adjustment
I checked with our primary timing board engineer for the answers to your questions. In summary, the adjustment of the 640kHz is just a centering process to give the IRIG input a full range to remain locked at the entire temperature range. This centering adjustment is designed to work through the full range of the ATX specifications for 5vdc power input, so it can be performed using any input voltage of +5.25 to +4.75. It does not need to be set to a specific value in between these two values in order for this centering adjustment to be performed.

As for ambient temperature at which the centering process should be performed, it should be performed at a typical room temperature. FYI- Our factory room temperature is usually between 70 and 75 degrees F. Just like the input voltage, the engineer said there is plenty of range for the IRIG lock to be maintained throughout the entire temperature range, by centering it in a typical room temperature. What you don't want to do is center it with the ambient temperature being near the min or max specified temperatures of -30 and +70 degrees C, because these temperatures are at the edges of the lock circuit (instead of it being centered, it will be near the rail). But centering it at a typical room temperature of around 70 degrees F, there is plenty of operating range in either direction for the IRIG input to remain locked throughout the entire temperature specs range for the PC104 boards.

Please let me know if you have any timing boards that are unable to be adjusted to within the specified frequency tolerances I had sent to you, when applying 4.75 to 5.25vdc input at ambient temperature.

Input timing accuracies

- same as the TPRO-TSAT-PCI series

GPS input – disciplines to within +/- 1 microsecond of the reference.
IRIG input- disciplines to within +/- 5 microseconds of the IRIG input reference.

The reason for the difference between the two specs is because the GPS reference is much more stable (has less jitter) than an IRIG input. If your customer wants the PC104 timing board to be synced within 1 microsecond, they will need to synchronize the board using its GPS antenna (installed outdoors with a good view of the sky so that it can track at least four satellites at all times). Otherwise, if they sync the board using an IRIG generator, it will be synced with less accuracy to the IRIG time generator.

**Option -M (1PPS input) when installed

From the user manual
Option –M provides a 1PPS synchronization input on Pin 9 and Pin 14 of the D-type connector. The 1PPS input can be RS-485 differential. The +/A signal connects to P1 Pin 9 and the -/B signal connects to P1 Pin 14. The 1PPS input can also be ground referenced (1PPS signal on P1 Pin 9, ground to P1 Pin 5 or Pin 7). A differential timecode input (IRIG-B, IRIG-A, NASA-36 or autodetect) can be applied to the BNC connector.

If both 1PPS and Timecode inputs are present, the board syncs to the incoming timecode and ignores the 1PPS. Note that commands 4D and 4E (enable/disable sync) apply to both the 1PPS and the timecode input. Commands E0 and E9 are not supported.

- Available with TPRO-PC104 boards only (not TSAT-PC104 boards)
- 1PPS is connected to pins 9 and 14 of the DB15 connector.
- Pins 9 and 14 of the DB15 connector for 1PPS input are high impedance inputs.

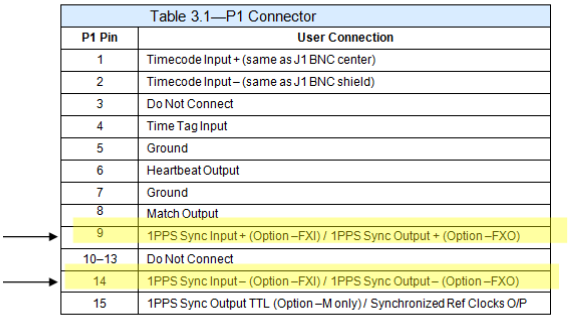


Table 3.1—P1 Connector

P1 Pin	User Connection
1	Timecode Input + (same as J1 BNC center)
2	Timecode Input – (same as J1 BNC shield)
3	Do Not Connect
4	Time Tag Input
5	Ground
6	Heartbeat Output
7	Ground
8	Match Output
9	1PPS Sync Input + (Option –FXI) / 1PPS Sync Output + (Option –FXO)
10–13	Do Not Connect
14	1PPS Sync Input – (Option –FXI) / 1PPS Sync Output – (Option –FXO)
15	1PPS Sync Output TTL (Option –M only) / Synchronized Ref Clocks O/P

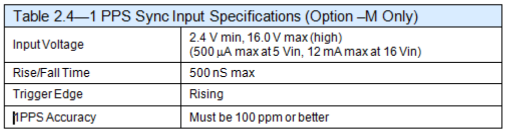


Table 2.4—1 PPS Sync Input Specifications (Option –M Only)

Input Voltage	2.4 V min, 16.0 V max (high) (500 µA max at 5 Vin, 12 mA max at 16 Vin)
Rise/Fall Time	500 nS max
Trigger Edge	Rising
1PPS Accuracy	Must be 100 ppm or better

Specs for the 1PPS input on pins 9 and 14

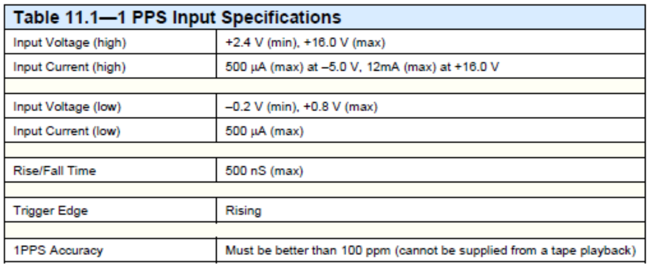


Table 11.1—1 PPS Input Specifications

Input Voltage (high)	+2.4 V (min), +16.0 V (max)
Input Current (high)	500 µA (max) at –5.0 V, 12mA (max) at +16.0 V
Input Voltage (low)	–0.2 V (min), +0.8 V (max)
Input Current (low)	500 µA (max)
Rise/Fall Time	500 nS (max)
Trigger Edge	Rising
1PPS Accuracy	Must be better than 100 ppm (cannot be supplied from a tape playback)

1PPS input pin termination

Q A customer from MIT Lincoln Labs is asking for the DB15 termination of a PC104 board with Option –M installed (the message he left with me is attached).

Reply from Tim Tetreault- When the 1pps reference is connected to Pin 9/14 for the PC104 with –M, it is a hi impedance input.

If time issues are observed with option -M

NOTE: The board expects the 1PPS input to be continuous. If the 1PPS signal stops pulsing after the board establishes initial sync, the board will continue to increment time (“freewheel”). However, if the 1PPS signal resumes after a period of freewheeling, the board may reset time to 001:00:00:00.000000. This is due to the fact that the 1PPS occurs outside of a narrow window in which the board expects it, either because the 1PPS has moved or because the board’s time has drifted during freewheeling.

Email from Tim Tetreault 8/14/12 regarding time not incrementing each second
If the time keeps being reset, it might be due to a noisy 1PPS causing the time to be reset all the time. Using a 50ohm load might help. They also can check to make sure that JP2 and JP3 are set to Pin 2/3.

**Time Tag (Event timestamping/Timetagging/TimeStamping/Time Stamping)

- Time Tag functionality for PC104 is the same as the PCI series timng boards (such as PCI, PCI-U-2 and PCI-66U).

Notes about Timetag (for both PC104 and PCI series timing boards)

- Time Tag for both PC104 and PCI series timing boards is the same.
- When the rising edge of the Time Tag Input (P1 Pin 4) occurs, the clock time is latched into a temporary register, and this register is then loaded into the FIFO buffer.
- Since the clock time is latched when the rising edge occurs, the Time Tag data will be correct regardless of how long it takes to read the FIFO.
- The same FIFO buffer is also use for drive calls (not HW_GetTime), including the getDate call.
- An interrupt, if enabled, is generated on the PCI bus to indicate a time stamp has occured.
- Unlike PMC and cPCI boards, PC104 and PCI series boards time stamps are not stored one at a time in the FPGA. Instead, they are stored in a FIFO buffer (First In / First Out).
- There may be a delay of up to 200 µS before time tag data is available in the FIFO
- Erratic operation may result if the rate of the time tags exceeds 1000 per second.

- The operator can *disable* external events by jumpering JP6 Pin ‘2-3’ unless there is a source of external event pulses connected to P1 Pin 4. To *enable* external events jumper JP6 Pin 1–2.
- The user's software establishes that a time tag has occurred in one of two ways: either by examining the least significant bit of the Status Register (FIFO Ready flag) to determine if it is a zero, or by waiting for an interrupt. Since the clock time is latched when the rising edge occurs, the Time Tag data will be correct regardless of how long it takes to read the FIFO.
- Time tags are read by first sending a command to the board, then reading ten 32-bit words from a first-in-first-out (FIFO) register.
- The user's software must wait a short amount of time after sending each command. This requirement is necessary because the on-board processor places a higher priority on maintaining the time than on processing commands. The actual amount of time needed to process a command depends on when the command was received relative to the on-board time. Entries in Chapter Six of the user manual specify the amount of time needed to assure that the command was received. There is no restriction on reading from the board.

Time Tag input

- TimeTag input is on pin 4 of the Timing connector (P1) on edge of the board.
- Time Tag occurs on the Rising edge of the input signal.

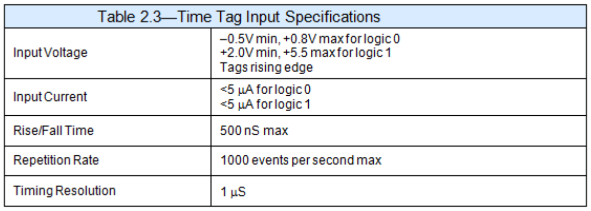


Table 2.3—Time Tag Input Specifications

Input Voltage	–0.5V min, +0.8V max for logic 0 +2.0V min, +5.5 max for logic 1 Tags rising edge
Input Current	<5 µA for logic 0 <5 µA for logic 1
Rise/Fall Time	500 nS max
Repetition Rate	1000 events per second max
Timing Resolution	1 µS

Pin-out

- Timetag input connects to pin 4 of DB15 connector P1

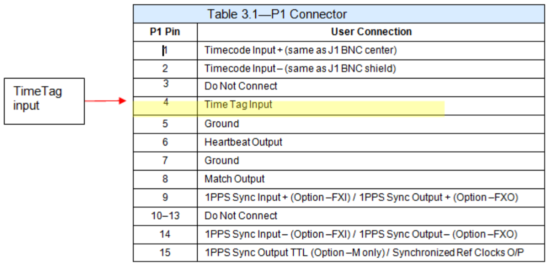

TimeTag input

Table 3.1—P1 Connector

P1 Pin	User Connection
1	Timecode Input + (same as J1 BNC center)
2	Timecode Input – (same as J1 BNC shield)
3	Do Not Connect
4	Time Tag Input
5	Ground
6	Heartbeat Output
7	Ground
8	Match Output
9	1PPS Sync Input + (Option –FXI) / 1PPS Sync Output + (Option –FXO)
10–13	Do Not Connect
14	1PPS Sync Input – (Option –FXI) / 1PPS Sync Output – (Option –FXO)
15	1PPS Sync Output TTL (Option –M only) / Synchronized Ref Clocks O/P

JP6 Event input jumper- Enable event timestamping

- The operator can *disable* external events by jumpering JP6 Pin ‘2-3’ unless there is a source of external event pulses connected to P1 Pin 4.
- To *enable* external events jumper JP6 Pin 1–2.

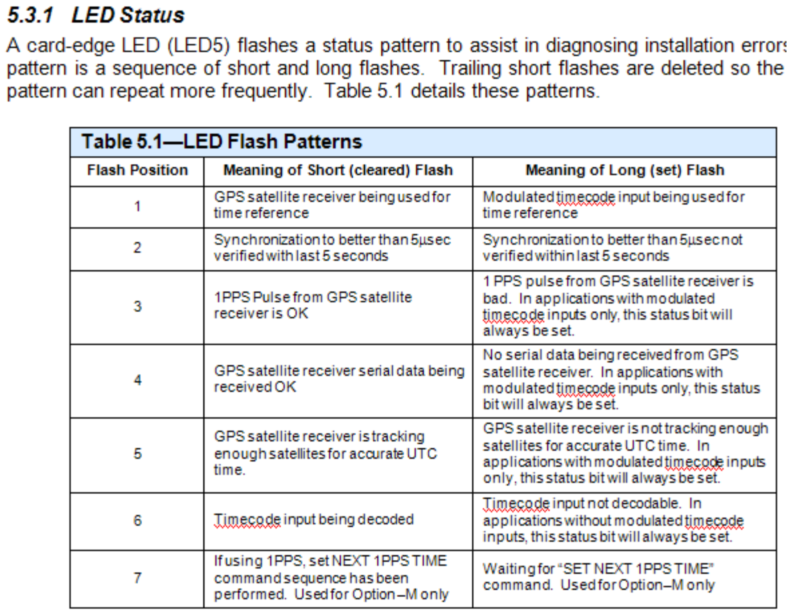
PC104 board misreading register data

I am having an intermittent problem when working with your TPRO-PC104 boards. When our software starts up, it writes the Report Firmware ID command (0xE9) to the command register and then reads back the version. Normally, I see 0xE9, 0xE9, 0xFF, 0xFF, 0xFF, 0xFF, 0xF3, 0xC3, 0x68, 0x01. However, I will occasionally not get back those bytes. When this happens, I see something like 0 six times, 0x8, 0x64, 0x01, 0x96. When this happens, the 3 LEDs on the board are not cycling at all.

I have also had an instance where I successfully read the firmware version and then after letting the board run for about ten minutes noticed that the LEDs were no longer cycling. I resent the firmware command and did not receive the normal response.
We thought this issue was resolved but it recently resurfaced. What I have found is that we were leaving JP6 in the default 1-2 setting. We are not using this input and it seems that noise is causing it to intermittently trigger causing the response that David described below. You may want to keep this in mind for future customer issues. In my opinion, it would be safer to make the default for JP6 to be disabled.

We currently have several boards in house. Most of the time the three board LED alternately flash with the green and yellow on state alternating with the red LED. On one of our boards, these LED are on solid all of the time. Is this something we should be concerned about? The board seems to function correctly otherwise.

**Status LED (LED5)



5.3.1 LED Status

A card-edge LED (LED5) flashes a status pattern to assist in diagnosing installation errors pattern is a sequence of short and long flashes. Trailing short flashes are deleted so the pattern can repeat more frequently. Table 5.1 details these patterns.

Table 5.1—LED Flash Patterns

Flash Position	Meaning of Short (cleared) Flash	Meaning of Long (set) Flash
1	GPS satellite receiver being used for time reference	Modulated timecode input being used for time reference
2	Synchronization to better than 5µsec verified with last 5 seconds	Synchronization to better than 5µsec not verified within last 5 seconds
3	1PPS Pulse from GPS satellite receiver is OK	1 PPS pulse from GPS satellite receiver is bad. In applications with modulated timecode inputs only, this status bit will always be set.
4	GPS satellite receiver serial data being received OK	No serial data being received from GPS satellite receiver. In applications with modulated timecode inputs only, this status bit will always be set.
5	GPS satellite receiver is tracking enough satellites for accurate UTC time.	GPS satellite receiver is not tracking enough satellites for accurate UTC time. In applications with modulated timecode inputs only, this status bit will always be set.
6	Timecode input being decoded	Timecode input not decodable. In applications without modulated timecode inputs, this status bit will always be set.
7	If using 1PPS, set NEXT 1PPS TIME command sequence has been performed. Used for Option–M only	Waiting for "SET NEXT 1PPS TIME" command. Used for Option–M only

Where:
Short = 10% duty cycle on/off (lit for about 0.1 seconds, not lit for about 0.9 seconds)

Long = 50% duty cycle on/off (lit for about 0.5 seconds, not lit for 0.5 seconds)

Analysis of the expected flash pattern with a good input
TSAT-PC104 (GPS input) = *pause* Short Short Short Short Short Short *pause*
(IRIG input) = *pause* Long Short Long Long Long Short *pause*

TPRO-PC104 = *pause* Long Short Long Long Long Short *pause*

Note: The flash pattern with Option –m installed may add a 7th long (but not a 7th short, because a last short is deleted)

What is meant by the "trailing short flashes are deleted", is that there may or may not be a total of 7 flash/blinks in a row. For instance, if the last two positions (positions 6 and 7 in the table above) both happen to be a long flash (instead of a quick blink), these two long flashes will not be displayed and therefore, there will only be a series of 5 blinks/flashes displayed. However, if positions 6 and 7 are classified as both being blinks, there will be a total of 7 LED lights in a row, instead, with the last two positions being a quick blink.

**Sync status- Reading the current Sync status with the driver/API interface

The current sync status of the board can be read using the **TPRO_SynchStatus** API call (via the installed Windows or Linux driver).

Email from Tim T to Dave L (12 Jun 13) There is no pin on the PC104 design that can be used to indicate SYNC. The only thing available is the status flags that can be read through software driver.
**Oscillator disciplining/freerun

Q. I see the oscillator stability specs on our data sheet listed as Disciplined to timecode: 2 x 10–7 and Undisciplined: 1 x 10–6, but can you help me understand how that translates into time error/drift per day? At the specified rates, what kind of time error would a customer see in 30 days if IRIG is unavailable? Thank you for your help.

- To calculate the amount of Freewheel drift, convert the time to seconds and multiply by 1.0E-6.

EX. 1 hr = 3600sec. Amount of drift = 3600 * 1.0E-6 = 0.0036sec
For 30 days, 3600sec*24hr*30days*1.0E-6 = **2.592 sec**

Note: To get less than a 1 sec error over 30 days of holdover would require at least an OCXO oscillator. Our PC104 board does not support using an OCXO oscillator. Our TSync board does, if a PCIe slot is available.

Outputs

**IRIG output

IRIG Output specs

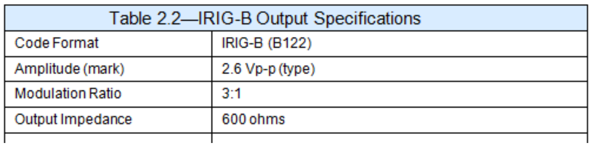
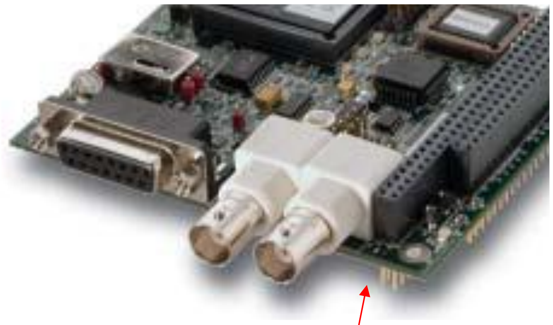


Table 2.2—IRIG-B Output Specifications	
Code Format	IRIG-B (B122)
Amplitude (mark)	2.6 Vp-p (type)
Modulation Ratio	3:1
Output Impedance	600 ohms



IRIG Output (J2)

- IRIG output BNC connector (J2) provides only IRIG B122 (AM modulation) at about 2.6vpp into 600 ohms

Note: PC104 boards can accept/sync to IRIG A input, but cannot output IRIG A. The output is always IRIG B122.

Leap year rollover

Note: This is the same as the TPRO/TSAT-PCI family functionality (PCI and PC104 use the same firmware)

Since TPRO/TSAT boards can't read the year from IRIG, without GPS applied, the year needs to be set manually each time it's powered-up or reset. Without GPS input, if the year is not manually set each time its rebooted/reset, it will assume it's not a leap year and it will not add day 366 to the end of the year. (The year will roll over on day 365 instead of on day 366).

The year can be set with API call or with the Windows Control Utility. But if the board is not operating in a Windows /Linux environment, the year can be set manually into a register with a "poke": From the PCI user manual (as of 4/27/11, this info is not currently in the TPRO/TSAT-PC104 manual).

Setting the Year (Not Applicable to Option -M)
To function properly, the TPRO requires that the year be set by command at the end of a leap year. The board will increment the year when the day rolls over to 001. Also, setting the year enables the board (TPRO and TSAT) to compute the Gregorian date using the year and Julian date.

Set the year by sending the command sequence "0x006n 0x007n 0x008n 0x009n 0x00ea" to the command port. For example, to set the year to 2003, send "0x0062 0x0070 0x0080 0x0093 0x00ea".
The year is reset to 0000 when power is first applied, or when any of the following occur: system reset, firmware reset (command 0x004f), or writing to the "Assert Reset" or "De-assert Reset" addresses.

Additional Note: Commands are sent to the TPRO-PC104 command register (Base Address+2) as a sequence of bytes. All commands should be spaced at least 100µs apart so the TPRO-PC104 firmware has sufficient time to handle each command.

Email KW sent to Eric Girard in France (1/26/11)
The TPRO/TSAT user manual is apparently a bit confusing about the Leap Year rollover, as your customer pointed out. However, to clarify your customer's understanding, when IRIG is the only input reference (no GPS) the year only needs to be set when the board is first powered-up. After power-up, the year will automatically increment to the new year value on each midnight on January 1st, whether or not it's a Leap year.

The timing board software knows what years are leap years and so it can compensate by knowing is the current year has 365 days or instead 366 days in a “Leap Year” year. Because it knows what years are leap years, it can automatically rollover the year value on the correct day, with no user intervention required. After every New Year rollover (leap year, or not leap year), NTP can continue getting the correct year from the timing board, without any necessary user intervention.

FYI- We have tested (and just re-verified) that the TPRO/TSAT-PC-U-2 and 66U boards do automatically handle the leap year rollover correctly, without any user intervention.

Multi-processor/Multi-core machines/ 32 bit OS

The TSAT-PC104 driver will run on a multi-processor machine, but it does not support multi-threaded/multi-processor application software.

Note: the driver currently supports 32 bit only. It is not certified as 64 bit compliant.

For more info on the PC104 driver, refer to: \\Exchange\empshare$\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\Driver support spreadsheets

**Synchronized Reference Clock Output (JP1)

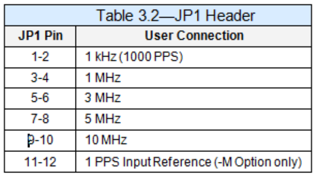


Table 3.2—JP1 Header

JP1 Pin	User Connection
1-2	1 kHz (1000 PPS)
3-4	1 MHz
5-6	3 MHz
7-8	5 MHz
9-10	10 MHz
11-12	1 PPS Input Reference (-M Option only)

The optional reference output clocks are user configurable via JP1 header adjacent to J1 BNC connector. Connection to reference clock outputs are obtainable via connector P1 pin 15.

- 1PPS Sync IN TTL (Option –M only): If the user specifies option –M when ordering, that pin will be used as a 1PPS input reference to the board. When this option is used, the reference clocks are not available.

**Heartbeat output / 1PPS output

Specs

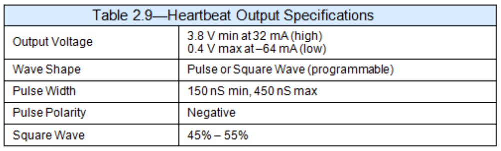


Table 2.9—Heartbeat Output Specifications

Output Voltage	3.8 V min at 32 mA (high) 0.4 V max at –64 mA (low)
Wave Shape	Pulse or Square Wave (programmable)
Pulse Width	150 nS min, 450 nS max
Pulse Polarity	Negative
Square Wave	45% – 55%

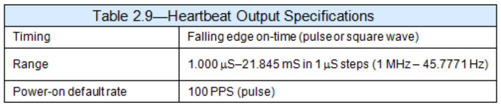


Table 2.9—Heartbeat Output Specifications

Timing	Falling edge on-time (pulse or square wave)
Range	1.000 µS–21.845 mS in 1 µS steps (1 MHz – 45.7771 Hz)
Power-on default rate	100 PPS (pulse)

Pin-out

- Heartbeat output is on **pin 6** of the DB15 (P1) connector

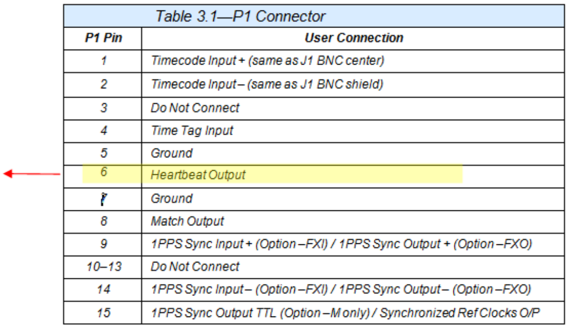


Table 3.1—P1 Connector

P1 Pin	User Connection
1	Timecode Input + (same as J1 BNC center)
2	Timecode Input – (same as J1 BNC shield)
3	Do Not Connect
4	Time Tag Input
5	Ground
6	Heartbeat Output
7	Ground
8	Match Output
9	1PPS Sync Input + (Option –FXI) / 1PPS Sync Output + (Option –FXO)
10–13	Do Not Connect
14	1PPS Sync Input – (Option –FXI) / 1PPS Sync Output – (Option –FXO)
15	1PPS Sync Output TTL (Option –M only) / Synchronized Ref Clocks O/P

Desire for a 1PPS heartbeat output

- Requires HB1PPS Output Option be installed at the factory.
- 1PPS output is available on the Heartbeat Output pin, with HB1PPS Option Installed.
- HB1PPS Option allows for the low frequency of 1Hz.
- With HB1PPS option installed, configure the heartbeat for 1Hz.

Email from Dave Lorah (12/7/12)
I looked into what it would take to get a 1PPS output from your TRPO-PC104 boards.

The Option HB1PPS will need to be installed. We can do this on your boards but they must be returned to Spectracom. The EEPROM will need to be reprogrammed and there are some board modifications that must be done.

The cost to add the HB1PPS (parts and labor) is $375.00 per board (plus shipping fees). If you would like to return the board(s) to add the HB1PPS option please let me know and I can set up a RMA to perform the work.

I will forward your contact info to our sales department who will provide you a quote on a new TPRO-PC104 – HB1PPS board.

Jam Sync/ No-Jam Option sync of the Heartbeat output

- **Jam Sync option:** - When they set the heartbeat, the new heartbeat will start as soon as the board gets the command (not likely at all to be coincident with the system PPS)
- **Non-jam sync option**: If they want the heartbeat out to be in sync with the system 1pps, they need to set it with this option.

Note: both of these are configurations of the PC104 board, as configured by either the driver or the Windows Control Utility. See email further below from Tim Tetreault.

- (as of at least Dec 2012) There is a known firmware issue that cuases the configuration of these two modes to be “backwards”. Refer to the section below about Jam/No-jam further below for more info about this.
- There isn’t much documentation about these two modes (except the info below). The manual doesn’t mention it and the APG for Windows only provides the info directly below:

From the PC104 Application Programmers guide for Windows. These two configurations are mentioned on pages 3-1 and 3-3 (as shown below)

```
#define SIG_NO_JAM (0)            // output type for PCI card
#define SIG_JAM        (1)        // output type for PCI card



/*===========================================================================
                    TPRO HEARTBEAT OBJECT
===========================================================================*/

typedef struct TPRO_HeartObj
{ /*----------------------------------------------------------------------*/
  unsigned char signalType;                    /*-- heart signal type ---*/
  unsigned char outputType;                    /*-- jamming option ------*/

  unsigned char clockFreq;                     /*-- clock freq for CPCI -*/

  double  frequency;                           /*-- heartbeat freq ------*/
} /*----------------------------------------------------------------------*/
TPRO_HeartObj;
```

Q. The Heartbeat seems to have a significant drift. I am comparing it to an external 1pps and it seems to shift in time by 60us. The spec does not offer info on this … is this normal as it seems quite high?

Response from Tim Tetreault (8/19/11)

Dave,

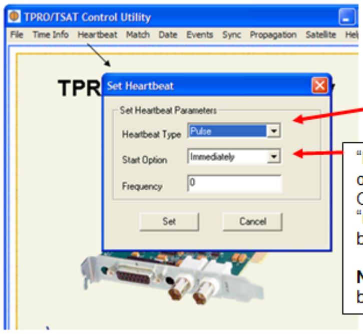
I am guessing the problem the customer is seeing is that they are setting the heartbeat using the Jam sync option. What this means is when they set the heartbeat, the new heartbeat will start as soon as the board gets the command. If they want it to be in sync with the 1pps, they need to set it with the non-jam sync option.

Unfortunately, this flag is backwards in the PC104 board. For example, if they are using the Control Utility program, they need to set the jam function to “Immediately” for it to be in sync with the 1pps.

The same goes for the example that we supply with the Windows driver. When they want to have the heartbeat in sync with the 1pps, the flag should be set to “SIG_JAM”.

When they are using an IRIG source to the board, the heartbeat error should be no greater then 5usec to the on time point. If they were using GPS or 1PPS reference, the error would be less than 1usec. 1PPS has a sharp edge, making it a more accurate edge to trigger to.

Also, tell the customer that every time they set the heartbeat, it will take it 5 to 8 seconds to settle in to its accuracy.



Heartbeat Type: Pulse
Start Option: Immediately
Frequency: 0

"**Pulse**" or "**Square**" (defines output signal

"**Immediately**: coincident with system PPS (non-jam option)
Or
"**next cycle:** occurs immediately- instead of waiting to be coincident with system PPS.

Note: these are "backwards" to a firmware issue in the board!

**Match time output (Start/Stop)

Match Output specs

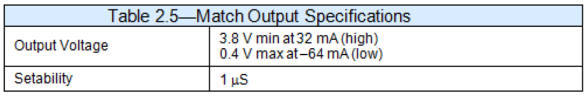


Table 2.5—Match Output Specifications

Output Voltage	3.8 V min at 32 mA (high) 0.4 V max at –64 mA (low)
Setability	1 µS

Pin-out

- Match time output is on pin 8 of the DB 15 (P1) connector

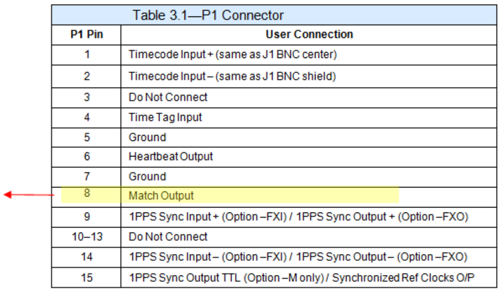


Table 3.1—P1 Connector

P1 Pin	User Connection
1	Timecode Input + (same as J1 BNC center)
2	Timecode Input – (same as J1 BNC shield)
3	Do Not Connect
4	Time Tag Input
5	Ground
6	Heartbeat Output
7	Ground
8	Match Output
9	1PPS Sync Input + (Option –FXI) / 1PPS Sync Output + (Option –FXO)
10–13	Do Not Connect
14	1PPS Sync Input – (Option –FXI) / 1PPS Sync Output – (Option –FXO)
15	1PPS Sync Output TTL (Option –M only) / Synchronized Ref Clocks O/P

PC-104 Drivers

List of all available API calls

- Refer to the Table of Contents in the Windows APG (1155-5002-0050): I:\Engineering\Archive\New Released\Manuals\1155-xxxx-xxxx

**Windows driver

Desire for 32/64 bit signed Windows drivers

Q What is the plan/timeline for 32/64 Win10 signed drivers, and are the PCI-U2 cards planned for win10 or only 66U? **A Reply from Dave Lorah (10 Sept 15)** At this time Spectracom has no plans to update the TPRO/TSAT –U2 or -66U Drivers. We are working on the TSync-PCIe drivers but no updates are planned for the older platforms.

- TPRO/TSAT-PC104 driver is 32 bit only. It is not 64 bit compatible (not digitally signed, as required by 64 bit Windows).

Email from Dave Lorah to Dave Deveau with Scisol on 8/11/11
We have tested the PC104 board you returned and found it is operating normally. But when installing the driver it was not recognizing the board just like you experienced. We think we figured out what the problem is. These ISA bus boards (PC-104) need to be manually installed. They do not work as plug and play with the find new hardware.

The application programmer's guide has some info on installing these boards:

2 Installing the Driver
2.1 Installing the TPRO/TSAT-PC104 Win 98/00/ME/XP Driver
1. Install TPRO/TSAT-PC104 card in a vacant ISA slot of desired Personal Computer.
2. Power on the PC and select Add/Remove Hardware from the Control Panel. When prompted, select "Have Disk" and choose the location of the **tpro.inf** on the distribution CD. The default parameters for the TPRO-PC104 card are a base address of 0x0300 and an IRQ of 10. If these parameters need to be different from your system, be sure to make the necessary changes.
3. Reboot machine when prompted.
4. Browse to the distribution CD, and execute the **setup.exe** program to install the development files and control utility.
5. Follow the on-screen prompts, making sure to accept defaults. The utility commences copying application files. When this process is finished, the control utility installation along with development files are installed.
6. Click the "Finish" button on the screen.

If that doesn't work try this. We had to do this here on our Win 2000 test PC which is pretty much the same as above.

Go to Control panel and select ADD HARDWARE
Choose Next on Hardware Wizard
Select Add a new hardware device from the list
Select Install the hardware that I manually select from a list (Advanced)
Select Other device from the list
Choose Have Disc
Browse to the driver on the CD
Select tpro.inf and OPEN
The board should show up now. Choose next etc. and finish
Device manager should show the board now as TPRO/TSAT ISA Timing board
The control Utility program should now operate.

Please give this a try and let me know how it goes.

**Windows driver Control Utility

Note: There is an issue with the driver that causes the year to be read incorrectly from either the control utility or their application.

**Desire for a customer to create their own custom driver (instead of using our drivers)

Address decoding- 16 bit registers, but normally use only 8 bits. (16 bit reads but only 8 bit writes).

Refer to "Register assignments" section in the PC-104 manual for breakdown of address decoding for the commands. Also, refer to the Linux driver source code (provided free with the Linux driver) as this source code provides examples on how we reference the registers for our Linux driver.

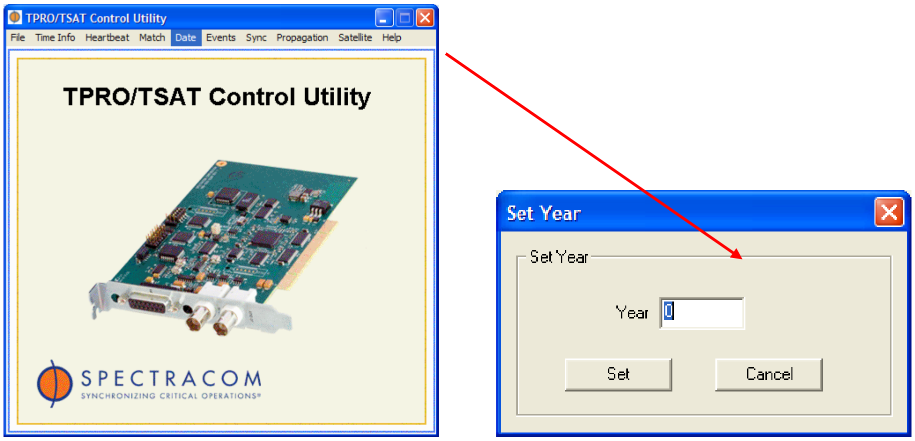
****Clock Daemon

Refer to (in this document): Clock Daemon / Clock Daemon service (for all timing boards)

To begin, the Windows driver for the TPRO-PC104 timing board contains two clock daemons and either one can be used to sync a Windows PC to a **synchronized** timing board. One is called the Clock daemon program and the other is the Clock daemon Service (unlike the Clock program, the Clock Service can automatically run at Windows startup).

In order for the Clock daemon to be able to set the Windows time, the TPRO/TSAT-PC104 board needs to be synced to an IRIG generator (as indicated the green LED on the edge of the board being lit or by using the **TPRO_SynchStatus** API call . Also, as the TPRO/TSAT-PC104 board can’t obtain year information from the IRIG generator, the year must also be set after each boot-up of the timing board, before the Clock Daemon can sync the PC.   Note that if you have a TSAT timing board, the year is automatically set when the board is synced to GPS.

The current year can be manually set in the TPRO timing boards (or the TSAT timing board when only using IRIG input instead of GPS input) using the Windows Control Utility (also installed as part of the Windows driver) - in the **Date** -> **Set Year** menu (the year can be read back via the **Date** -> **Retrieve Gregorian Date** menu).   Or, the year can also be set using the API call of **TPRO_SetYear** (this command can be run in application software to automatically set the year).



Year 0

With the TPRO/TSAT-PC104 timing board synced to the IRIG generator and with the year set using either of the methods mentioned above, the Clock Daemon program or Service can be used to periodically sync the Windows PC. Note that both Clock Daemons can be accessed via **Start / All Programs / Spectracom Corp / PCI**.

Functions

Status via API call

****Reading Sync** Uses the **TPRO_SynchControl** API call.

**Interrupts

The TPRO-PC104 can be programmed to request interrupts at the IRQ level configured by P2 upon selectable conditions. Interrupts can be enabled by writing a "1" into the corresponding interrupt enable bit in Base Address+1.

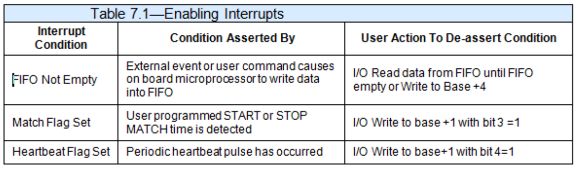


Table 7.1—Enabling Interrupts

Interrupt Condition	Condition Asserted By	User Action To De-assert Condition
FIFO Not Empty	External event or user command causes on board microprocessor to write data into FIFO	I/O Read data from FIFO until FIFO empty or Write to Base +4
Match Flag Set	User programmed START or STOP MATCH time is detected	I/O Write to base +1 with bit 3 =1
Heartbeat Flag Set	Periodic heartbeat pulse has occurred	I/O Write to base+1 with bit 4=1

**Manually setting the time

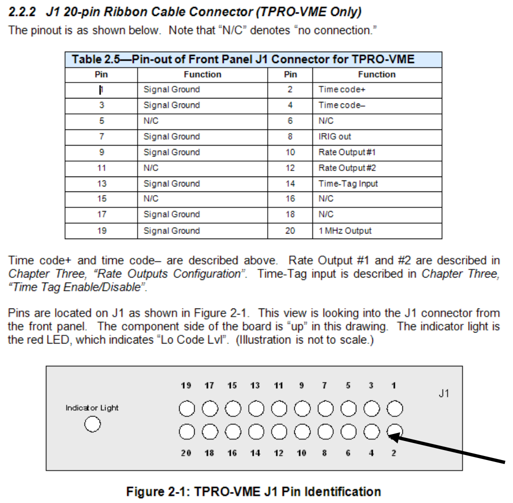
The operator can set the TPRO-PC104 to ***use*** an input reference for applications where the input reference is ***not used***. If the operator does not preset a time, a default preset of 0 days through seconds is used. The operator must disable input synchronization before setting time, or the TPRO-PC104 firmware will just switch back from the commanded time to the input time as soon as the input is validated.

Note: Refer to “**Setting Time**” in the PC104 user manual for more info.

TPRO/TSAT-VME timing boards (discontinued/no longer accepted for RMA)

Note: no longer supported or accepted back for Repair.

- Refer to: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/RetiredProducts/tabid/120/Default.aspx



2.2.2 J1 20-pin Ribbon Cable Connector (TPRO-VME Only)

The pinout is as shown below. Note that “N/C” denotes “no connection.”

Table 2.5—Pin-out of Front Panel J1 Connector for TPRO-VME

Pin	Function	Pin	Function
1	Signal Ground	2	Time code+
3	Signal Ground	4	Time code–
5	N/C	6	N/C
7	Signal Ground	8	IRIG out
9	Signal Ground	10	Rate Output #1
11	N/C	12	Rate Output #2
13	Signal Ground	14	Time-Tag Input
15	N/C	16	N/C
17	Signal Ground	18	N/C
19	Signal Ground	20	1 MHz Output

Time code+ and time code– are described above. Rate Output #1 and #2 are described in *Chapter Three, “Rate Outputs Configuration”*. Time-Tag input is described in *Chapter Three, “Time Tag Enable/Disable”*.

Pins are located on J1 as shown in Figure 2-1. This view is looking into the J1 connector from the front panel. The component side of the board is “up” in this drawing. The indicator light is the red LED, which indicates “Lo Code Lvl”. (Illustration is not to scale.)

Figure 2-1: TPRO-VME J1 Pin Identification

Shortcut to TPRO/TSAT-VME user manual (1156-5001-0050): I:\Engineering\Archive\New Released\Manuals\1156-xxxx-xxxx

Shortcut to TPRO/TSAT-VME VXWorks driver guide (1156-5002-0050) I:\Engineering\Archive\New Released\Manuals\1156-xxxx-xxxx

Shortcut to TPRO/TSAT-VME in Customer Service: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PCI PC104 VME

MTBF: Refer to: MTBF/MTTR (for all products)

Certifications (CE, FCC, etc)

CE Declaration of conformity

As of at least October 2012, the TPRO/TSAT-cPCI, TPRO/TSAT-PMC, TPRO/TSAT-PC104 and TPRO/TSAT-VME timing boards are not CE approved and therefore do not have an available CE Declaration of conformity.

AVAILABLE OPTIONS:

-**MJ5:** This option adds a 1PPS input on the BNC connector J5 in addition to the one on a second connector that is added with Option M.   9MJ5 is a slight variation of Option M).

- 32P2: This option removes the top and bottom rows of the P2 bus connector to prevent BCD time data from being passed to the back plane card.

- Drilled boards: Some boards around serial number 03275 had a board defect where one of the pins in P14 was drilled out because a square pad on an internal layer was shorting 5 V to 12V (GROUND?). On these boards two wire jumpers were added to correct for the drilled out via. These are in the wrong place if a non-default P14 location is used like Rockwell Collins is using. Both wires should go to the via the wire toward the back of the board is tied to.

GPS antenna (Acutime antenna) for TSAT-VME boards

The Acutime antennas are programmed at the factory to operate with ONLY TSAT-VME boards. They will not operate with any other timing board. More recently shipped Acutime antennas will have a label applied which indicates what timing boards that it is programmed to operate with.

Acutime antennas for TSAT-VME boards should be annotated as P/N **1156-0000-5000.**

Important Note: Do not connect the Acutime antenna to a TSync-PCIe board unless it's for a permanent change, as the TSync-PCIe board will reprogram the antenna to operate with the TSync-PCIe board only. It won't work with the TSAT-VME board again, without factory re-programming of the antenna.

The Acutime antenna attaches to the TSAT-VME board using **CA05-1512-0XXX** antenna cable (where xxx is the length of the antenna cable)
For more info on Acutime antennas, refer to: Acutime GPS Antennas

LEDS

Red panel LED

The LOW CODE INPUT LED indicates the IRIG input signal amplitude is too low. The LED will also light if there is no IRIG input to the board. The IRIG Input specs for the TSAT-VME board are 1.2 to 8Vp-p.

LINUX DRIVER

From Mathew Siegel 11/9/07:
A couple things worth noting:

1) The README file does not explain that mknod must be run after doing the modprobe (no big deal but worth updating before more disks go out that fail to create the character device).

2) In order to build on that 2.4 kernel, I had to add '-mcmodel=kernel' to the compile options. Honestly, no idea why but when I ran insmod it failed and then asked if I compiled the module with that option.

RAM and NVRAM for security concerns

Regarding your request for the RAM and NVRAM memory contained on the TSAT-VME timing card, I have the following information for you:

- The card contains both RAM and NVRAM.
- The RAM memory is 2K bytes
- The NVRAM memory is 8K bytes
- Neither memory (RAM and NVRAM) are user accessible.

Reading latitude, Longitude, elevation and number of satellites tracked from the TSAT-VME boar

(Email from Neil Barton 9/12/11)
Q. it appears that every ~10 seconds the TSAT board reports an incorrect longitude. The latitude, altitude, and time are still reported correctly. After reporting 1 incorrect longitude, the board goes back to reporting the proper longitude for another 10 seconds.

We have tried using both of our TSAT-VME boards as well as multiple different Acutime Gold antennas and we are getting the same result. Additionally, we have used a VME bus analyzer to look at the GPS data coming directly from the TSAT-VME board. We saw the same longitude error when reviewing the data from the VME bus analyzer.

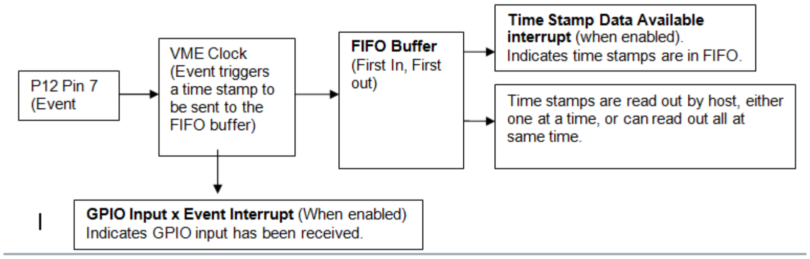
I just wanted to check and see if you could offer any guidance on what we are seeing with this issue. I will be in Alaska all next week with the system, so if you have anything for me to try I should be able to do it.

A (Reply from Dave Lorah) Hi Neil,
I have some information for you. I was not sure what could be causing the problem so I consulted an engineer who told me there is a requirement when reading that information, a minimum of 100usec must be allowed to complete the transfer of info. (See section 6.5 in the VME User manual) . So when reading the Lat, Long, Elev and #SAT, the software must wait for each set of data.

Negative Elevation in a TSAT-VME

-----Original Message-----
From: Fred Zwarts [F.Zwarts@KVI.nl]
Sent: Wednesday, May 14, 2008 3:38 AM
Hi Keith,
I have tried my modified code for some time now. What I see is that the altitude reading varies between 0.0 and a few meters positive. The variation, I think, is due to the precision of the GPS signals. The value 0.0 is seen sometimes for a long time. It would be very unlikely that with this variation of altitude readings a negative altitude would have been missed by accident.
I see that the least significant nibble of the altitude is always 0, as documented, so it does not contain the sign information.
Apparently your first guess was right: Negative altitudes are reported as 0.0.

TimeTag (Time Tag/Time stamping/ timestamping)


P12 Pin 7 (Event
VME Clock (Event triggers a time stamp to be sent to the FIFO buffer)
FIFO Buffer (First In, First out)
Time Stamp Data Available interrupt (when enabled). Indicates time stamps are in FIFO.
Time stamps are read out by host, either one at a time, or can read out all at same time.
GPIO Input x Event Interrupt (When enabled) Indicates GPIO input has been received.

Typo Error in the TPRO/TSAT-VME manual regarding time tag

As of 7/8/10 kw, there is a typo in the Rev C manual about enabling Time Tag.
“The default setup is for Time Tag to be disabled (not enabled by default as indicated in the manual). However, for your configuration, you want the Time Tag to be enabled, instead. The picture of the jumper being configured for the “disable” configuration (Figure 3-12 is correct). So, to enable the Time Tag, just move the jumper to the “W7 to W8” position (instead of the default “W7 to W6” position).”
The Rev C manual shows both enable and disable as being the default configuration for the VME boards, but it should state disabled as the only default value.

Notes about Timetagging

- Timetag is disabled by default
- Is enabled with a jumper change on the VME board.

Note: 4/1/10 KW- The TPRO-VME and the TSAT-VME should be able to process the same number of time tags. I did find a discrepancy in the maximum number of events per second listed on the data sheet vs. the user manual.

The data sheet lists 1000 events per second is the max, but the user manual states the unit could have a problem if more than 200 events per second are tagged. Dave is checking with Tim T to see what the correct answer is.

Palisade GPS antenna for old TSAT-VME boards

- Refer to “Acutime antennas” in the custserviceassist doc: ..\CustomerServiceAssistance.pdf

The very old TSAT-VME boards from Odetics/KSI used a Palisade antenna from Trimble. This antenna was replaced by the Acutime 2000. An older board that had this Palisade antenna will not sync to a new antenna. I updated the U44 firmware from version 981624N to current version 1156-SE01-0100. The board would then work on the new antenna.

1PPS Interrupt

To begin, attached you should find a copy of the TSAT-VME users manual in case you don’t have a copy of this guide.

The SEL1 or SEL2 pins available on the terminal strip connector (Connector P12, pins 5 and 6, as shown on page 3-16 of the attached manual) can be configured for a 1PPS output. Refer to Section 3.11 of the attached manual for additional information on using the jumpers to configure the output rate of SEL 1 or 2 as 1PPS. Either of these two outputs can then be connected back into the “External Event” input pin, (Connector P12, pin 7) to generate a FIFO interrupt. With the SEL1 or SEL2 output pins configured for a 1PPS rate, the FIFO buffer interrupt will generate a 1 Hz

interrupt that you can use as a reference. Sections 3.4 through 3.13 of the attached manual discuss this particular configuration.
If you are just interested in the interrupt occurring each second, you can configure the FIFO Clear configuration jumper to clear the interrupt when the host acknowledges the interrupt. (Refer to Section 3.7 in the manual for more information on clearing the FIFO interrupt).

KSI Demo utility program

Compatible only with TPRO/TSAT PC, PCI, PC-104 and cPCI. NOT compatible with VME cards. Windows based- Only provided on the Windows driver CD when the Windows driver is purchased In the program, select the type of card installed. Then, below will show the “name of the card” starting at 0. If more than one card is installed on the PC, it will increment to 1, 2 , 3 , etc. If the program displays “ an error message that card can’t be found”, verify that the card is installed. Go to Control panel/ system/System Hardware/Device Manager and confirm the timing card is in the list. If it is not, verify the card is seated and verify the drivers are installed.

Special KSI customer requirements

Email from Jim Rall to Wade Sober: Here is all I know about the special customer requirements that KSI had. I apologize that some of these notes appear cryptic but this is what I copied from their “cheat sheet”.

> **Steatite**: TPRO-PMC boards must have a capacitor add done to the board. This rework fixes lock-up problems on **Concurrent Technologies PP220/01x Single board computer**. Windows OS (Win2000 Server edition) locked-up due ringing, incorrect load capacitance and degraded rise-time of the TPRO-PMC PCICLK signal.
>
> **Timing Solutions:** Do not ship with board manuals. Software drivers must use release “X” and labeled using the Timing Solutions label. I did scan through the KSI drive and found some examples of the Release X and what I think is the Timing Solutions CD label.
>
> **Colsa and Digital Wizards:** Always get FREE drivers and shipping.
>
> **Invensys Systems Inc**: As you already know they have issued a spec control drawing that lists their specific part numbers for various packages.
>
> **MDA Systems**: They are the only customer (according to Tom Barbone) that has purchased an extended warranty. Reference KSI order number 10661. This was a 3-year warranty extension. I do not recall the actual ship date.

This is all I have on special requirements.

3vdc/5vdc applications

- KSI boards are 5vdc only. They will not operate on 3vdc.
- SGI systems use 3vdc.

TPRO-VME-D / TSAT-VME-D (discontinued/no longer supported)

- This is a time display option for the VME timing boards

Email from Dave Lorah to a customer (7/18/12)
~~We can perform limited repair service on a TPRO-VME-D board. This product has been discontinued and is obsolete. We no longer have parts for the display board.~~

~~We can take a look at it if you wish but cannot guarantee it can be repaired. The evaluation fee is $200. If repairs are needed it could be more. Return shipping is added to the repair cost.~~

~~If you wish to receive a RMA to return this board for evaluation/repair please reply with your billing and shipping addresses and I will reply with a RMA number and shipping instructions.~~

Questions/answers
Q. Is operation of the TPRO-PCI with Windows XP a problem?
A. No, we have drivers for almost every OS, including Solaris.

Q. I have a customer interested in a TPRO-PCI board. He is also interested in GPS. Are the TPRO-PCI and TSAT-PCI boards physically the same so that I could sell him a TSAT-PCI and he could choose to use IRIG or GPS?
A. The TSAT-PCI is a superset of the TPRO-PCI and will accept IRIG-A/B or GPS as an Input.

Q. Another quick question if you don't mind. What about ARINC 429, was there ever a plan to incorporate this capability into the boards?
A. Not at the current time

Q. Can the TPRO-PCI board both generate and read IRIG signals?
A. Yes, all of the TPRO boards (except the TPRO-IP) can both generate, read and regenerate (sync to incoming IRIG-B and reproduce and output) IRIG-B signal.

Q. The only thing in the guide related to the Time Tag I found was TPRO_waitEvent. Is this correct? I think it would be good if the card could raise a hardware interrupt to Windows XP whenever a Time Tag event is detected. Does the card support this? What happens if you call TPRO_waitEvent a second time before a new Time Tag event occurs?
A. Our timing boards do generate a hardware interrupt when a Time Tag event is detected. The app "TPRO_waitEvent" will wait for this interrupt or until the timeout has occurred. Unless you have a dual core processor and you are running 2 applications at the same time, you can't call the app a second time until the first one has completed.

- It is indicated that the "falling edge of the heartbeat" is the "on time" edge. Does this imply that the falling edge of the heartbeat would be when the PCI interrupt is generated if that interrupt option is programmed?

A. The PCI interrupt is generated on the falling edge of the heartbeat.

- What is the timing relationship between the heartbeat edge and the PCI interrupt?

A. The time difference between the heartbeat edge and the PCI interrupt is less then a microsecond. If they need a more accurate measurement, we will have to setup a board and measure it.

Q It looks like the TSAT version will not accept an IRIG input. Is that correct?
A. The TSAT-PC104 board will take both IRIG and satellite inputs. If both connected at the same time, the IRIG is the default.

Q. I cannot see any way to set the year (in the IRIG product) or to read the year (in the GPS product).
A. We are not sure if the year can be set in the TPRO. It looks like an undocumented function that we need to look in to. The year can be read TSAT by reading the Altitude. The year info is included in that information. The user manual shows the order of the bytes for the Altitude command.

Q. There does not seem to be a Time Tag interrupt capability. Is that correct?

A. Whenever a Time Tag event happens, the FIFO Output Ready Interrupt is generated if enabled. Since the clock time is latched when the rising edge occurs, the Time Tag data will be correct regardless of how long it takes to read the FIFO.

Q. The contents of some of the registers (eg. tbreg_ttag_Status, tbreg_clk_date, tbreg_ttag_date) are not described.
A. tbreg_clk_date & tbreg_ttag_date have the same format as tbreg_cmd[2] in the “Set Time” function. tbreg_ttag_Status has the same format as tbreg_Status.

Q There does not seem to be any way to command the board to switch between IRIG and GPS sources (if both are present at the same time). Is this true or am I missing something?
A. You are correct. If both sources are connected, the board will default to IRIG. There is no way to switch between the two. There is a SYNCHRONIZATION SOURCE INDICATOR that indicates which input time source is being used.

TPRO/TSAT-PC (discontinued/no longer supported)

This is a board we inherited when we purchased the KSI timing board line.

- Very early, discontinued product
- Refer to KSI manual in: EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PC

FAQs

Q. I successfully implemented the set heartbeat & get time example code for the card. I noticed there was a set heartbeat example and not a get heartbeat example. Is it possible to read the frequency, signal type, and output type from the card?
A. We looked into this and found there isn't a way to poll the current values from the Timing board. The only indication of the current settings is when the command is issued, it echoes back how it's was just set.

AITG-VME from KSI (discontinued/not supported/not accepted for repair)

- **For limited available info/docs/schematics, refer to**: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\KSI AITG boards

Per Wade Sober (17 Nov 2017) This is a board we inherited when we purchased the KSI timing board line. We have never produced it ourselves. We had some inventory when we purchased the boards but got rid of any excess boards when we moved to this facility

Function

From Chapter 1 of the AITG manual

The AITG-VME is basically an accurate clock that automatically synchronizes to external time signals and whose time can be captured and read by the host VME bus computer. It is used in VME bus systems for time tagging, synchronizing multiple systems, and generating puts rates at exact times and exact frequencies. Optional operating system handlers can be used for user interface to the AITG-VME or users may directly access the AITG-VME with their own code (see appendices for sample code used in factory testing). Time capture can be caused by a logic pulse from the outside world (an "external" event).

Q. I'm looking for technical specifications, programming details, drivers etc, and schematic for the Odetics/KSI systems AITG-VME card rev E.4.
This particular rev level is a bit different than the ones we are using.
Also is this card still available?
We still use a bunch of these cards on the Eastern Test Range at KSC and CCAFS.
Any help would be appreciated and maybe I can send some business your way?

A. The AITG-VME board is discontinued/obsolete and no longer sold or supported by Spectracom. We have no reference material on this product available.
Sorry for the bad news.

TPRO-IP (discontinued)

This is a board we inherited when we purchased the KSI timing board line.

Note: (~5 Oct 2013) This product is now discontinued.

- **Shortcut to datasheet**: I:\Marketing_Product Data Sheets\Bus-Level Timing Boards\TPRO-IP_revE.pdf
- **Shortcut to manual CD assy (1157-0001-6000) in Arena:** https://app.bom.com/items/detail-spec?item_id=1203165686&version_id=10221246158
- **Shortcut to manual (1157-5001-0050) in Arena:** https://app.bom.com/items/detail-spec?item_id=1203165688&version_id=10221246178
- **Shortcut to TPRO-IP folder in Released drive (1157 family):** ..\..\..\Engineering\Archive\Released\1157 - TPRO IP:
- **Shortcut to TPRO-IP items in New Released drive (1157 family):** ..\..\..\Engineering\Archive\New Released

ECNs associated with TPRO-IP board

1. **ECN 1044 (Jan 2006) to release the KSI documentation as our own, Rev A (from the older ECN database)**

   "internal" Email from Keith (20 Sept 2019) ...I went back into the very old ECN database to see if I could happen to find any Engineering changes incorporated for this device (though I didn't recall any). I found exactly one (**ECN 1044**, opened Jan 2006, to release this old KSI device's documentation to our system as Rev A) .

....I also checked the next newer ECN database to see if there were any other ECNs for TPRO-IP and found two additonal

2. **ECN 2076 (May 2007) "BOARD WOULD LATCH WHEN THE IRIG SIGNAL WAS BELOW 100mVp-p"**

   - CUSTOMER FOUND TPRO-IP BOARD WOULD LATCH WHEN THE IRIG SIGNAL WAS BELOW 100mVp-p. FOUND FIRMWARE BUG.

   Update BOM in Visual. Release new firmware for EEPROM (U3). Update test procedure. Update TPRO-IP Traveler. Updated TPRO-IP Bring-Up Instructions. Created a Drill Drawing for the PCB. Created new ZIP release files. All of our current inventory should be updated and retested. All customer returns should be updated.

~~~~~~~~~~~~~~~

3. **ECN 2088 (June 2007) "Current FPGA is obsolete and must be replaced."**

   Import old FPGA project to new Actel Designer and convert to newer FPGA A42MX09. Test all functions related to FPGA and verified equal or better then old FPGA. Update Visual BOM calling out FPGA and FPGA image. Updated traveler with new CheckSum info for FPGA and assigned P/N. Update assembly ZIP file. Save FPGA project in RCS. Updated schematic.
~~~~~~~~~~~~~~~

Memory/Sanitization

- Refer to the TPRO/TSAT-PCI series memory document: I:\Customer Service\EQUIPMENT\SPECTRACOM EQUIPMENT\Timing boards\TPRO-TSAT\PCI and PC104 VME\PCI series memory

Q. Is there any information available aside from the TPRO-IP datasheet that indicates the onboard memory components such as RAM, SDRAM, ROM, etc...?   For security purposes, it would be helpful to know 1. The type of memory onboard. 2. Is the memory volatile or NVM? 3. What is the process to sanitize the memory? 4. Is the memory user modifiable? 5. What is the size of the memory.

A Reply from Dave Lorah (11 Sept 15) The TPRO-IP Timing Board is similar in architecture to the TPRO-PCI Timing boards and has the same memory and will not retain any user settings or information after power is removed. I have attached a document for the TPRO/TSAT PCI board. We have no documentation like this for the TPRO-IP but the same information applies as the TPRO-PCI Timing board.

FAQs

Q. (From Adam Elovitz with Elotek) Benjamin has an embedded project where he is hoping to use IRIG time from the IP board. Benjamin has already downloaded the manual; is there further documentation we can email over?

A. (Reply from Dave Lorah on 9/19/11) We do not have a lot on the TPRO-IP board. Aside from the user manual all we have is some sample code that is downloadable from our website: http://www.spectracomcorp.com/Support/HowCanWeHelpYou/Library/tabid/59/Default.aspx?EntryId=27

Options for TRO-IP boards

-O (the letter- not the number): **High performance oscillator**