ELECTRONICS & DEFENSE

TSync

Time Code Processor with Optional GPS

Application Programmer's Guide

1219-5002-0050, Rev: 2

8 January 2026

© 2026 Safran. All rights reserved.

Information furnished by Safran is believed to be accurate and reliable. However, no responsibility is assumed by Safran for its use, nor for any infringements of patents or other rights of third parties that may result from its use. Safran reserves the right to make changes without further notice to any products herein. Safran makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Safran assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. No license is granted by implication or otherwise under any patent or patent rights of Safran. Trademarks and registered trademarks are the property of their respective owners. Safran products are not intended for any application in which the failure of the Safran product could create a situation where personal injury or death may occur. Should Buyer purchase or use Safran products for any such unintended or unauthorized application, Buyer shall indemnify and hold Safran and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable legal fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Safran was negligent regarding the design or manufacture of the part.

Safran Electronics & Defense

safran-navigation-timing.com

Safran Trusted 4D

• 45 Becker Road, Suite A, West Henrietta, NY 14586 USA
• 3, Avenue du Canada, 91974 Les Ulis, France

The industry-leading Spectracom/Orolia products you depend on are now brought to you by Safran.

Do you have questions or comments regarding this User Manual?
➔ E-mail: techpubs@nav-timing.safrangroup.com

Table of Contents

TSYNC 1
1 OVERVIEW 1-1
2 HOST BUS MEMORY MAP OF REGISTERS 2-1
2.1 System Clock 2-3
2.1.1 Host Super-Second System Time Low 2-3
2.1.1.1 Address: 0x000 2-3
2.1.2 Host Super-Second System Time Mid Low 2-3
2.1.2.1 Address: 0x002 2-3
2.1.3 Host Super-Second System Time Mid High 2-3
2.1.3.1 Address: 0x004 2-3
2.1.4 Host Super-Second System Time High 2-3
2.1.4.1 Address: 0x006 2-3
2.1.5 Host Sub-Second System Time Low 2-4
2.1.5.1 Address: 0x008 2-4
2.1.6 Host Sub-Second System Time High 2-4
2.1.6.1 Address: 0x00A 2-4
2.1.7 Host Super-Second Binary Time Low 2-4
2.1.7.1 Address: 0x00C 2-4
2.1.8 Host Super-Second Binary Time High 2-4
2.1.8.1 Address: 0x00E 2-4
2.1.9 Host Sub-Second BCD Time Low 2-4
2.1.9.1 Address: 0x010 2-4
2.1.10 Host Sub-Second BCD Time High 2-5
2.1.10.1 Address: 0x012 2-5
2.2 Time Stamping 2-6
2.2.1 Time Stamping Control / Status 2-6
2.2.1.1 Address: 0x020 2-6
2.2.2 Time Stamping FIFO Super-Seconds Low 2-6
2.2.2.1 Address: 0x022 2-6
2.2.3 Time Stamping FIFO Super-Seconds Mid Low 2-7
2.2.3.1 Address: 0x024 2-7
2.2.4 Time Stamping FIFO Super-Seconds Mid High 2-7
2.2.4.1 Address: 0x026 2-7
2.2.5 Time Stamping FIFO Super-Seconds High 2-7
2.2.5.1 Address: 0x028 2-7
2.2.6 Time Stamping FIFO Sub-Seconds Low 2-7
2.2.6.1 Address: 0x02A 2-7
2.2.7 Time Stamping FIFO Sub-Seconds High 2-8
2.2.7.1 Address: 0x02C 2-8
2.3 GPIO Output 2-9
2.3.1 GPIO Output 0 High Match Time Sub-Second Low 2-9
2.3.1.1 Address: 0x040 2-9
2.3.2 GPIO Output 0 High Match Time Sub-Second High 2-9
2.3.2.1 Address: 0x042 2-9
2.3.3 GPIO Output 0 High Match Time Super-Second Low 2-9
2.3.3.1 Address: 0x044 2-9
2.3.4 GPIO Output 0 High Match Time Super-Second Mid 2-10
2.3.4.1 Address: 0x046 2-10

2.3.5 GPIO Output 0 High Match Time Super-Second High 2-10
2.3.5.1 Address: 0x048 2-10
2.3.6 GPIO Output 0 Low Match Time Sub-Second Low 2-10
2.3.6.1 Address: 0x050 2-10
2.3.7 GPIO Output 0 Low Match Time Sub-Second High 2-10
2.3.7.1 Address: 0x052 2-10
2.3.8 GPIO Output 0 Low Match Time Super-Second Low 2-11
2.3.8.1 Address: 0x054 2-11
2.3.9 GPIO Output 0 Low Match Time Super-Second Mid 2-11
2.3.9.1 Address: 0x056 2-11
2.3.10 GPIO Output 0 Low Match Time Super-Second High 2-11
2.3.10.1 Address: 0x058 2-11
2.3.11 GPIO Output 1 High Match Time Sub-Second Low 2-11
2.3.11.1 Address: 0x060 2-11
2.3.12 GPIO Output 1 High Match Time Sub-Second High 2-12
2.3.12.1 Address: 0x062 2-12
2.3.13 GPIO Output 1 High Match Time Super-Second Low 2-12
2.3.13.1 Address: 0x064 2-12
2.3.14 GPIO Output 1 High Match Time Super-Second Mid 2-12
2.3.14.1 Address: 0x066 2-12
2.3.15 GPIO Output 1 High Match Time Super-Second High 2-13
2.3.15.1 Address: 0x068 2-13
2.3.16 GPIO Output 1 Low Match Time Sub-Second Low 2-13
2.3.16.1 Address: 0x070 2-13
2.3.17 GPIO Output 1 Low Match Time Sub-Second High 2-13
2.3.17.1 Address: 0x072 2-13
2.3.18 GPIO Output 1 Low Match Time Super-Second Low 2-13
2.3.18.1 Address: 0x074 2-13
2.3.19 GPIO Output 1 Low Match Time Super-Second Mid 2-14
2.3.19.1 Address: 0x076 2-14
2.3.20 GPIO Output 1 Low Match Time Super-Second High 2-14
2.3.20.1 Address: 0x078 2-14
2.3.21 GPIO Output 2 High Match Time Sub-Second Low 2-14
2.3.21.1 Address: 0x080 2-14
2.3.22 GPIO Output 2 High Match Time Sub-Second High 2-14
2.3.22.1 Address: 0x082 2-14
2.3.23 GPIO Output 2 High Match Time Super-Second Low 2-15
2.3.23.1 Address: 0x084 2-15
2.3.24 GPIO Output 2 High Match Time Super-Second Mid 2-15
2.3.24.1 Address: 0x086 2-15
2.3.25 GPIO Output 2 High Match Time Super-Second High 2-15
2.3.25.1 Address: 0x088 2-15
2.3.26 GPIO Output 2 Low Match Time Sub-Second Low 2-16
2.3.26.1 Address: 0x090 2-16
2.3.27 GPIO Output 2 Low Match Time Sub-Second High 2-16
2.3.27.1 Address: 0x092 2-16
2.3.28 GPIO Output 2 Low Match Time Super-Second Low 2-16
2.3.28.1 Address: 0x094 2-16
2.3.29 GPIO Output 2 Low Match Time Super-Second Mid 2-16
2.3.29.1 Address: 0x096 2-16
2.3.30 GPIO Output 2 Low Match Time Super-Second High 2-17

2.3.30.1 Address: 0x098 ... 2-17
2.3.31 GPIO Output 3 High Match Time Sub-Second Low ... 2-17
2.3.31.1 Address: 0x0A0 ... 2-17
2.3.32 GPIO Output 3 High Match Time Sub-Second High ... 2-17
2.3.32.1 Address: 0x0A2 ... 2-17
2.3.33 GPIO Output 3 High Match Time Super-Second Low ... 2-17
2.3.33.1 Address: 0x0A4 ... 2-17
2.3.34 GPIO Output 3 High Match Time Super-Second Mid ... 2-18
2.3.34.1 Address: 0x0A6 ... 2-18
2.3.35 GPIO Output 3 High Match Time Super-Second High ... 2-18
2.3.35.1 Address: 0x0A8 ... 2-18
2.3.36 GPIO Output 3 Low Match Time Sub-Second Low ... 2-18
2.3.36.1 Address: 0x0B0 ... 2-18
2.3.37 GPIO Output 3 Low Match Time Sub-Second High ... 2-18
2.3.37.1 Address: 0x0B2 ... 2-18
2.3.38 GPIO Output 3 Low Match Time Super-Second Low ... 2-19
2.3.38.1 Address: 0x0B4 ... 2-19
2.3.39 GPIO Output 3 Low Match Time Super-Second Mid ... 2-19
2.3.39.1 Address: 0x0B6 ... 2-19
2.3.40 GPIO Output 3 Low Match Time Super-Second High ... 2-19
2.3.40.1 Address: 0x0B8 ... 2-19
2.4 Host Bus Interface ... 2-20
2.4.1 FPGA Control / Status ... 2-20
2.4.1.1 Address: 0x0C0 ... 2-20
2.4.2 FPGA Revision ID ... 2-20
2.4.2.1.1 Address: 0x0C2 ... 2-20
2.4.3 Host Bus Interrupt Mask ... 2-21
2.4.3.1 Address: 0x0C4 ... 2-21
2.4.4 Host Bus Interrupt Status ... 2-22
2.4.4.1 Address: 0x0C6 ... 2-22
2.5 Shared Memory ... 2-23
2.5.1 Host / Microcontroller Bus FIFO Control / Status ... 2-23
2.5.1.1 Address: 0x160 ... 2-23
2.5.2 Host / Microcontroller Bus FIFO Data ... 2-23
2.5.2.1 Address: 0x180 ... 2-23
2.5.3 Microcontroller / Host Bus FIFO Data ... 2-23
2.5.3.1 Address: 0x1C0 ... 2-23
2.6 Interrupts ... 2-24
2.6.1 Interrupt Descriptions ... 2-24
2.6.1.1 1PPS Received ... 2-24
2.6.1.2 Timing System Service Request ... 2-24
2.6.1.3 Local / µC Bus FIFO Empty ... 2-24
2.6.1.4 Local / µC Bus FIFO Overflow ... 2-24
2.6.1.5 µC / local bus FIFO Data Available ... 2-24
2.6.1.6 µC / local bus FIFO Overflow ... 2-24
2.6.1.7 GPIO Input x Event ... 2-24
2.6.1.8 GPIO Output x Event ... 2-24
2.6.1.9 Timestamp Data Available ... 2-25
3 HOST INTERFACE PROTOCOL (HIP)—INTRODUCTION ... 3-1
3.1 Application Layer ... 3-2
3.2 Connection Layer ... 3-3

3.3 Transport Layer 3-3
3.4 Data Link Layer 3-3
4 HOST INTERFACE PROTOCOL (HIP)—DETAILS 4-1
4.1 Application Layer 4-1
4.1.1 Features 4-1
4.1.2 Layer Details 4-1
4.1.3 Message Details 4-1
4.2 Connection Layer 4-2
4.2.1 Features 4-2
4.2.2 Layer Details 4-2
4.2.3 Message Details 4-2
4.2.3.1 Data Types and Definitions 4-2
4.2.3.2 Message Formats 4-3
4.3 Transport Layer 4-5
4.3.1 Features 4-5
4.3.2 Layer Details 4-5
4.3.3 Message Details 4-5
4.3.3.1 Data Types and Definitions 4-5
4.3.3.2 Message Formats 4-7
5 MESSAGE TRANSACTIONS 5-1
6 DEPENDENCIES 6-1
7 DATA TYPES AND DEFINITIONS 7-1
7.1 General 7-1
7.2 Host Agent 7-15
7.3 Initializer 7-16
7.4 Oscillator Monitor 7-16
7.5 Shared Memory 7-17
7.6 Reference Monitor 7-18
7.7 Supervisor 7-19
7.8 Upgrade 7-20
7.9 Flash Manager 7-20
7.10 GPS Reference 7-21
7.11 IRIG Reference 7-24
7.12 ASCII Output 7-24
7.13 ASCII Reference 7-24
7.14 IRIG Output 7-24
7.15 Variable Frequency 7-25
7.16 Display Output 7-25
7.17 Oscillator 7-25
7.18 LED Control 7-26
7.19 General Purpose Input 7-26
7.20 General Purpose Output 7-27
7.21 Log Service 7-28
7.22 E1/T1 Output 7-28
7.23 PTP Reference 7-29
8 PUBLIC MESSAGE API 8-35
8.1 Example Message Transaction 8-35
8.2 Host Agent API 8-36
8.2.1 Get Capabilities Transaction (`HA_CMD_GET_CAPS`) 8-36
8.3 Component API 8-38

8.3.1 Services ... 8-38
8.3.1.1 Initializer ... 8-38
8.3.1.1.1 Status Transaction (IN_CA_STATUS) ... 8-38
8.3.1.2 Discovery and Configuration Service ... 8-39
8.3.1.2.1 Option Card Information Transaction (DCS_CA_OC_INFO) ... 8-39
8.3.1.2.2 Feature by Index Transaction (DCS_CA_OC_FEAT_IDX) ... 8-40
8.3.1.2.3 Feature Instance Transaction (DCS_CA_INSTANCE) ... 8-42
8.3.1.2.4 Feature Slot Transaction (DCS_CA_SLOT) ... 8-43
8.3.1.3 Oscillator Monitor ... 8-44
8.3.1.3.1 Register Transaction (XS_CA_REGISTER) ... 8-44
8.3.1.3.2 Unregister Transaction (XS_CA_UNREGISTER) ... 8-45
8.3.1.3.3 Meter Command Transaction (XS_CA_METER_CMD) ... 8-46
8.3.1.3.4 Window Size Transaction (XS_CA_WINDOW_SIZE) ... 8-47
8.3.1.3.5 Meter Data Transaction (XS_CA_METER_DATA) ... 8-48
8.3.1.4 Clock ... 8-49
8.3.1.4.1 Time Scale Transaction (CS_CA_TIME_SCALE) ... 8-49
8.3.1.4.2 Time Scale Offset Transaction (CS_CA_TIME_SCALE_OFF) ... 8-50
8.3.1.4.3 Sub-second Adjustment Transaction (CS_CA_SUBSEC_ADJ) ... 8-51
8.3.1.4.4 Time Transaction (CS_CA_TIME) ... 8-52
8.3.1.4.5 Leap Second Transaction (CS_CA_LEAP_SEC) ... 8-53
8.3.1.4.6 Time Zone Offset Transaction (CS_CA_TIME_ZONE_OFF) ... 8-55
8.3.1.4.7 DST Rule Transaction (CS_CA_DST_RULE) ... 8-56
8.3.1.4.8 Year Transaction (CS_CA_YEAR) ... 8-57
8.3.1.4.9 DST State Transaction (CS_CA_DST_STATE) ... 8-59
8.3.1.5 Reference Monitor ... 8-60
8.3.1.5.1 Factory Default Transaction (RS_CA_FACT_DEFAULT) ... 8-60
8.3.1.5.2 User Default Transaction (RS_CA_USER_DEFAULT) ... 8-61
8.3.1.5.3 Save User Default Transaction (RS_CA_SAVE_USER_DEFAULT) ... 8-62
8.3.1.5.4 Best Reference Transaction (RS_CA_BEST_REFERENCE) ... 8-63
8.3.1.5.5 Table Transaction (RS_CA_TABLE) ... 8-64
8.3.1.5.6 Table Entry Transaction (RS_CA_TABLE_ENTRY) ... 8-65
8.3.1.5.7 Table Entry Add Transaction (RS_CA_TABLE_ENTRY_ADD) ... 8-66
8.3.1.5.8 Table Entry Delete Transaction (RS_CA_TABLE_ENTRY_DEL) ... 8-67
8.3.1.5.9 Priority Transaction (RS_CA_PRIORITY) ... 8-68
8.3.1.5.10 Enable Transaction (RS_CA_ENABLE) ... 8-69
8.3.1.5.11 Get Reference State Table Transaction (RS_CA_REF_STATE_TABLE) ... 8-70
8.3.1.6 Supervisor ... 8-71
8.3.1.6.1 Reference Transaction (SS_CA_REF) ... 8-71
8.3.1.6.2 Maximum TFOM Transaction (SS_CA_MAX_TFOM) ... 8-72
8.3.1.6.3 TFOM Transaction (SS_CA_TFOM) ... 8-73
8.3.1.6.4 Sync Transaction (SS_CA_SYNC) ... 8-74
8.3.1.6.5 Holdover Transaction (SS_CA_HOLDOVER) ... 8-75
8.3.1.6.6 Holdover Timeout Transaction (SS_CA_HOLDOVER_TIMEOUT) ... 8-76
8.3.1.6.7 Timestamp Transaction (SS_CA_TIME_STAMP) ... 8-77
8.3.1.6.8 Uptime Transaction (SS_CA_UPTIME) ... 8-78
8.3.1.6.9 Reset Transaction (SS_CA_RESET) ... 8-79
8.3.1.6.10 Freerun Transaction (SS_CA_FREERUN) ... 8-80
8.3.1.7 Upgrade ... 8-81

8.3.1.7.1 Start Transaction (US_CA_START) ... 8-81
8.3.1.7.2 Data Transaction (US_CA_DATA) ... 8-82
8.3.1.7.3 End Transaction (US_CA_END) ... 8-83
8.3.1.7.4 Cancel Transaction (US_CA_CANCEL) ... 8-84
8.3.1.7.5 State Transaction (US_CA_STATE) ... 8-85
8.3.1.7.6 Start Option Card Transaction (US_CA_START_OC) ... 8-86
8.3.1.8 Persistent Data ... 8-87
8.3.1.9 Flash Manager ... 8-87
8.3.1.9.1 Get CRC Transaction (FS_CA_GET_CRC) ... 8-87
8.3.1.9.2 Calculate CRC Transaction (FS_CA_CALC_CRC) ... 8-88
8.3.1.9.3 Image Header Transaction (FS_CA_IMG_HEADER) ... 8-89
8.3.1.9.4 Get Version Transaction (FS_CA_GET_VERSION) ... 8-90
8.3.1.10 Log Service ... 8-91
8.3.1.10.1 Error Log Transaction (LS_CA_ERROR_LOG) ... 8-91
8.3.1.10.2 Alarm Command Transaction (LS_CA_ALARM) ... 8-92
8.3.1.10.3 Firmware Version Transaction (LS_CA_VERSION) ... 8-93
8.3.1.10.4 Serial Number Transaction (LS_CA_SERIAL_NO) ... 8-94
8.3.2 Components ... 8-95
8.3.2.1 GPS Reference ... 8-95
8.3.2.1.1 Offset Transaction (GR_CA_OFFSET) ... 8-95
8.3.2.1.2 Validity Transaction (GR_CA_VALIDITY) ... 8-96
8.3.2.1.3 Position Transaction (GR_CA_POSITION) ... 8-97
8.3.2.1.4 Receiver Mode Transaction (GR_CA_RCVR_MODE) ... 8-98
8.3.2.1.5 Dynamics Mode Transaction (GR_CA_DYNAMICS) ... 8-100
8.3.2.1.6 Fix Data Transaction (GR_CA_FIX_DATA) ... 8-101
8.3.2.1.7 Satellite Data Transaction (GR_CA_SAT_DATA) ... 8-102
8.3.2.1.8 Survey Progress Transaction (GR_CA_SURVEY_PROG) ... 8-103
8.3.2.1.9 Manufacturer Model Transaction (GR_CA_MFR_MDL) ... 8-104
8.3.2.1.10 Receiver Info Transaction (GR_CA_RCVR_INFO) ... 8-105
8.3.2.1.11 Custom Message Transaction (GR_CA_CUSTOM_MSG) ... 8-105
8.3.2.1.12 Get Number of Instances Transaction (GR_CA_NUM_INST) ... 8-106
8.3.2.1.13 Delete Stored Position Transaction (GR_CA_DEL_POS) ... 8-108
8.3.2.1.14 Reference Identifier Transaction (GR_CA_REF_ID) ... 8-108
8.3.2.1.15 GPS Reset Transaction (GR_CA_RESET) ... 8-109
8.3.2.1.16 Antenna Status Transaction (GR_CA_ANTENNA) ... 8-110
8.3.2.2 IRIG Reference ... 8-112
8.3.2.2.1 Offset Transaction (IR_CA_OFFSET) ... 8-112
8.3.2.2.2 Validity Transaction (IR_CA_VALIDITY) ... 8-113
8.3.2.2.3 Mode Transaction (IR_CA_MODE) ... 8-114
8.3.2.2.4 Format Transaction (IR_CA_FORMAT) ... 8-115
8.3.2.2.5 Modulation Transaction (IR_CA_MOD) ... 8-116
8.3.2.2.6 Frequency Transaction (IR_CA_FREQ) ... 8-117
8.3.2.2.7 Coded Expression Transaction (IR_CA_CODED_EXP) ... 8-118
8.3.2.2.8 Control Field Transaction (IR_CA_CTRL_FLD) ... 8-119
8.3.2.2.9 Message Transaction (IR_CA_MESSAGE) ... 8-120
8.3.2.2.10 Get Number of Instances Transaction (IR_CA_NUM_INST) ... 8-122
8.3.2.2.11 Control Field Data Transaction (IR_CA_CFDATA) ... 8-123
8.3.2.2.12 Local Clock Transaction (IR_CA_LOCAL) ... 8-124

8.3.2.2.13 Time Scale Transaction (IR_CA_TIME_SCALE) 8-126
8.3.2.2.14 Reference Identifier Transaction (IR_CA_REF_ID) 8-127
8.3.2.3 HaveQuick Reference 8-128
8.3.2.3.1 Offset Transaction (QR_CA_OFFSET) 8-128
8.3.2.3.2 Validity Transaction (QR_CA_VALIDITY) 8-129
8.3.2.3.3 Format Transaction (QR_CA_FORMAT) 8-130
8.3.2.3.4 Local Clock Transaction (QR_CA_LOCAL) 8-131
8.3.2.3.5 Time Scale Transaction (QR_CA_TIME_SCALE) 8-133
8.3.2.3.6 Reference Identifier Transaction (QR_CA_REF_ID) 8-134
8.3.2.3.7 Get Number of Instances Transaction (QR_CA_NUM_INST) 8-135
8.3.2.4 ASCII Reference 8-136
8.3.2.4.1 Offset Transaction (AR_CA_OFFSET) 8-136
8.3.2.4.2 Validity Transaction (AR_CA_VALIDITY) 8-137
8.3.2.4.3 UART Configuration Transaction (AR_CA_UART_CFG) 8-137
8.3.2.4.4 Leap Second Pending Transaction (AR_CA_LEAP_PENDING) 8-139
8.3.2.4.5 Local Clock Transaction (AR_CA_LOCAL) 8-140
8.3.2.4.6 Time Scale Transaction (AR_CA_TIME_SCALE) 8-141
8.3.2.4.7 Reference Identifier Transaction (AR_CA_REF_ID) 8-143
8.3.2.4.8 Format Transaction (AR_CA_FORMAT) 8-143
8.3.2.4.9 Get Number of Instances Transaction (AR_CA_NUM_INST) 8-145
8.3.2.4.10 Host Reference 8-146
8.3.2.4.11 Valid Transaction (HR_CA_VALID) 8-146
8.3.2.4.12 Time Transaction (HR_CA_TIME) 8-147
8.3.2.4.13 Local Clock Transaction (HR_CA_LOCAL) 8-148
8.3.2.4.14 Time Scale Transaction (HR_CA_TIME_SCALE) 8-149
8.3.2.4.15 Reference Identifier Transaction (HR_CA_REF_ID) 8-151
8.3.2.4.16 Get Number of Instances Transaction (HR_CA_NUM_INST) 8-152
8.3.2.5 PPS Reference 8-153
8.3.2.5.1 Offset Transaction (PR_CA_OFFSET) 8-153
8.3.2.5.2 Edge Transaction (PR_CA_EDGE) 8-154
8.3.2.5.3 Validity Transaction (PR_CA_VALIDITY) 8-155
8.3.2.5.4 Number of Instances Transaction (PR_CA_NUM_INST) 8-155
8.3.2.5.5 Reference Identifier Transaction (PR_CA_REF_ID) 8-157
8.3.2.6 Frequency Reference 8-158
8.3.2.6.1 Frequency Transaction (FR_CA_FREQUENCY) 8-158
8.3.2.6.2 Validity Transaction (FR_CA_VALIDITY) 8-159
8.3.2.6.3 Number of Instances Transaction (FR_CA_NUM_INST) 8-160
8.3.2.6.4 Reference Identifier Transaction (FR_CA_REF_ID) 8-161
8.3.2.7 IRIG Output 8-162
8.3.2.7.1 Signature Control Transaction (IP_CA_SIG_CTL) 8-162
8.3.2.7.2 Offset Transaction (IP_CA_OFFSET) 8-163
8.3.2.7.3 Local Transaction (IP_CA_LOCAL) 8-164
8.3.2.7.4 Format Transaction (IP_CA_FORMAT) 8-166
8.3.2.7.5 Amplitude Transaction (IP_CA_AMPLITUDE) 8-167
8.3.2.7.6 Modulation Transaction (IP_CA_MOD) 8-168
8.3.2.7.7 Frequency Transaction (IP_CA_FREQ) 8-169
8.3.2.7.8 Coded Expression Transaction (IP_CA_CODED_EXP) 8-170
8.3.2.7.9 Control Field Transaction (IP_CA_CTRL_FLD) 8-171

8.3.2.7.10 Message Transaction (IP_CA_MESSAGE) ... 8-172
8.3.2.7.11 Control Field Data Transaction (IP_CA_CFDATA) ... 8-173
8.3.2.7.12 Number of Instances Transaction (IP_CA_NUM_INST) ... 8-175
8.3.2.7.13 Phase Transaction (IP_CA_PHASE) ... 8-175
8.3.2.7.14 Phase Error Transaction (IP_CA_PHASE_ERR) ... 8-176
8.3.2.7.15 Time Scale Transaction (IP_CA_TIME_SCALE) ... 8-177
8.3.2.8 HaveQuick Output ... 8-179
8.3.2.8.1 Signature Control Transaction (QP_CA_SIG_CTL) ... 8-179
8.3.2.8.2 Offset Transaction (QP_CA_OFFSET) ... 8-180
8.3.2.8.3 Local Transaction (QP_CA_LOCAL) ... 8-181
8.3.2.8.4 Format Transaction (QP_CA_FORMAT) ... 8-183
8.3.2.8.5 Time Scale Transaction (QP_CA_TIME_SCALE) ... 8-184
8.3.2.8.6 Get Number of Instances Transaction (QP_CA_NUM_INST) ... 8-185
8.3.2.9 Fixed Frequency Output ... 8-186
8.3.2.9.1 Signature Control Transaction (FP_CA_SIG_CTL) ... 8-186
8.3.2.9.2 Frequency Transaction (FP_CA_FREQ) ... 8-187
8.3.2.9.3 Get Number of Instances Transaction (FP_CA_NUM_INST) ... 8-188
8.3.2.10 Display Output ... 8-189
8.3.2.10.1 Local Transaction (DP_CA_LOCAL) ... 8-189
8.3.2.10.2 Format Transaction (DP_CA_FORMAT) ... 8-191
8.3.2.10.3 Time Scale Transaction (DP_CA_TIME_SCALE) ... 8-192
8.3.2.10.4 Get Number of Instances Transaction (DP_CA_NUM_INST) ... 8-193
8.3.2.10.5 Mode Transaction (DP_CA_MODE) ... 8-194
8.3.2.11 ASCII Output ... 8-195
8.3.2.11.1 Signature Control Transaction (AP_CA_SIG_CTL) ... 8-195
8.3.2.11.2 Local Transaction (AP_CA_LOCAL) ... 8-196
8.3.2.11.3 Time Scale Transaction (AP_CA_TIME_SCALE) ... 8-198
8.3.2.11.4 Format Transaction (AP_CA_FORMAT) ... 8-199
8.3.2.11.5 Mode Transaction (AP_CA_MODE) ... 8-200
8.3.2.11.6 Request Transaction (AP_CA_REQ_CHAR) ... 8-201
8.3.2.11.7 UART Configuration Transaction (AP_CA_UART_CFG) ... 8-202
8.3.2.11.8 Get Number of Instances Transaction (AP_CA_NUM_INST) ... 8-203
8.3.2.12 Variable Frequency Output ... 8-204
8.3.2.12.1 Signature Control Transaction (VP_CA_SIG_CTL) ... 8-204
8.3.2.12.2 Frequency Transaction (VP_CA_FREQ) ... 8-206
8.3.2.12.3 Range Transaction (VP_CA_RANGE) ... 8-207
8.3.2.12.4 Get Number of Instances Transaction (VP_CA_NUM_INST) ... 8-208
8.3.2.13 PPS Output ... 8-209
8.3.2.13.1 Signature Control Transaction (PP_CA_SIG_CTL) ... 8-209
8.3.2.13.2 Frequency Transaction (PP_CA_FREQ) ... 8-210
8.3.2.13.3 Offset Transaction (PP_CA_OFFSET) ... 8-211
8.3.2.13.4 Edge Transaction (PP_CA_EDGE) ... 8-212
8.3.2.13.5 Pulse Width Transaction (PP_CA_PW) ... 8-213
8.3.2.13.6 Get Number of Instances Transaction (PP_CA_NUM_INST) ... 8-214
8.3.2.14 E1/T1 Output ... 8-215
8.3.2.14.1 Signature Control Transaction (ETP_CA_SIG_CTL) ... 8-215
8.3.2.14.2 Offset Transaction (ETP_CA_OFFSET) ... 8-216
8.3.2.14.3 Mode Transaction (ETP_CA_MODE) ... 8-217

8.3.2.14.4 Format Transaction (ETP_CA_FORMAT) ... 8-218
8.3.2.14.5 Get Number of Instances Transaction (ETP_CA_NUM_INST) ... 8-219
8.3.2.15 Oscillator ... 8-220
8.3.2.15.1 Disciplining State Transaction (XO_CA_DISC_STATE) ... 8-220
8.3.2.15.2 Mode Transaction (XO_CA_MODE) ... 8-221
8.3.2.15.3 DAC Transaction (XO_CA_DAC) ... 8-222
8.3.2.15.4 Alarm Transaction (XO_CA_ALARM) ... 8-223
8.3.2.15.5 Serial Number Transaction (XO_CA_SERIAL_NUM) ... 8-224
8.3.2.15.6 Manufacturer Model Transaction (XO_CA_MFR_MDL) ... 8-225
8.3.2.15.7 Custom Message Transaction (XO_CA_CUSTOM_MSG) ... 8-226
8.3.2.15.8 Disciplining Transaction (XO_CA_DISC) ... 8-227
8.3.2.15.9 Estimated Phase Error Transaction (XO_CA_PHASE_ERR) ... 8-228
8.3.2.15.10 Estimated Frequency Error Transaction (XO_CA_FREQ_ERR) ... 8-229
8.3.2.15.11 Oscillator Type Transaction (XO_CA_OSC_TYPE) ... 8-230
8.3.2.16 LED Control ... 8-231
8.3.2.16.1 Mode Transaction (EC_CA_MODE) ... 8-231
8.3.2.16.2 State Transaction (EC_CA_STATE) ... 8-232
8.3.2.17 General Purpose Input ... 8-233
8.3.2.17.1 Value Transaction (GI_CA_VALUE) ... 8-233
8.3.2.17.2 Edge Transaction (GI_CA_EDGE) ... 8-234
8.3.2.17.3 Timestamp Enable Transaction (GI_CA_TIMESTAMP_EN) ... 8-235
8.3.2.17.4 Get Number of Instances Transaction (GI_CA_NUM_INST) ... 8-236
8.3.2.18 General Purpose Output ... 8-237
8.3.2.18.1 Signature Control Transaction (GO_CA_SIG_CTL) ... 8-237
8.3.2.18.2 Output Enable Transaction (GO_CA_OUTPUT_EN) ... 8-238
8.3.2.18.3 Value Transaction (GO_CA_VALUE) ... 8-239
8.3.2.18.4 Mode Transaction (GO_CA_MODE) ... 8-241
8.3.2.18.5 DVM Value Transaction (GO_CA_DVM_VALUE) ... 8-241
8.3.2.18.6 Match Enable Transaction (GO_CA_MATCH_EN) ... 8-243
8.3.2.18.7 Square Wave Transaction (GO_CA_SQUARE) ... 8-244
8.3.2.18.8 Get Number of Instances Transaction (GO_CA_NUM_INST) ... 8-246
8.3.2.19 PTP Reference Component API ... 8-247
8.3.2.19.1 Module Information Transaction (PTR_CA_MODULE_INFO) ... 8-247
8.3.2.19.2 Ethernet ITF Transaction (PTR_CA_ETHERNET_ITF) ... 8-248
8.3.2.19.3 Clock Settings Transaction (PTR_CA_CLOCK_SETTINGS) ... 8-249
8.3.2.19.4 Unit Settings Transaction (PTR_CA_UNIT_SETTINGS) ... 8-251
8.3.2.19.5 Port State Transaction (PTR_CA_PORT_STATE) ... 8-252
8.3.2.19.6 Port Settings Transaction (PTR_CA_PORT_SETTINGS) ... 8-253
8.3.2.19.7 Clock Quality Transaction (PTR_CA_CLOCK_QUALITY) ... 8-255
8.3.2.19.8 Time Properties Transaction (PTR_CA_TIME_PROPERTIES) ... 8-257
8.3.2.19.9 Parent Properties Transaction (PTR_CA_PARENT_PROP) ... 8-258
8.3.2.19.10 GM Properties Transaction (PTR_CA_GM_PROP) ... 8-259
8.3.2.19.11 TOD Enable Transaction (PTR_CA_TOD_ENABLED) ... 8-260
8.3.2.19.12 PPS Enable Transaction (PTR_CA_PPS_ENABLED) ... 8-261
8.3.2.19.13 PPS Edge Transaction (PTR_CA_PPS_EDGE) ... 8-262
8.3.2.19.14 Save Settings (PTR_CA_SAVE_SETTINGS_TO_ROM) ... 8-263
8.3.2.19.15 Reset Module (PTR_CA_RESET_MODULE) ... 8-264
8.3.2.19.16 Get Number of Instances Transaction (PTR_CA_NUM_INST) ... 8-265

8.3.2.19.17 ReInit Module Transaction (PTR_CA_NUM_INST) 8-266
8.3.2.19.18 PTP Validity Transaction (PTR_CA_VALIDITY) 8-267
8.3.2.19.19 PTP Mode Transaction (PTR_CA_MODE) 8-268
8.3.2.19.20 PTP MAC Address Transaction (PTR_CA_MAC_ADDR) 8-269
8.3.2.19.21 Master Active Transaction (PTR_CA_MASTER_ACTIVE) 8-270
8.3.2.19.22 Module Status Transaction (PTR_CA_MODULE_STATUS) 8-271
8.3.2.19.23 PTP User Description Transaction (PTR_CA_PTP_USER_DESCRIPTION) 8-272
8.3.2.20 EBU Output 8-273
8.3.2.20.1 Signature Control Transaction (EP_CA_SIG_CTL) 8-273
8.3.2.20.2 Offset Transaction (EP_CA_OFFSET) 8-274
8.3.2.20.3 Local Transaction (EP_CA_LOCAL) 8-276
8.3.2.20.4 Format Transaction (EP_CA_FORMAT) 8-278
8.3.2.20.5 Time Scale Transaction (EP_CA_TIME_SCALE) 8-279
8.3.2.20.6 Amplitude Transaction (EP_CA_AMPLITUDE) 8-280
8.3.2.20.7 Get Number of Instances Transaction (EP_CA_NUM_INST) 8-281

1 Overview

The TSync architecture essentially consists of input references. These references are used as sources of 1PPS synchronization and/or time-of-day (TOD) and date information, as well as for disciplining an internal oscillator. The TSync device takes the synchronous clock, a 1PPS, time-of-day (TOD), and the date to create output references that act as time references for other devices. Also provided are other interfaces used for time stamping external events, creating precisely timed external signals, debugging, upgrades, and access from a host computer.

The TSync device uses the internal disciplined oscillator and the time reference acquired from the input references to implement a system clock. This clock maintains time-of-day, date, a 1PPS, and an internal synchronous clock. The TSync device uses this system clock to create output references, which are time code outputs, a 1PPS output, and 1PPS synchronous clocks. The capabilities of this architecture can be modified by changing the type of oscillator used. Depending on the application requirements, either TCXO or OCXO oscillators may be used.

2 Host Bus Memory Map of Registers

Address	Name
	System Clock
0x000	Host Super-Second System Time Low
0x002	Host Super-Second System Time Mid Low
0x004	Host Super-Second System Time Mid High
0x006	Host Super-Second System Time High
0x008	Host Sub-Second System Time Low
0x00A	Host Sub-Second System Time High
0x00C	Host Super-Second Binary Time Low
0x00E	Host Super-Second Binary Time High
0x010	Host Sub-Second BCD Time Low
0x012	Host Sub-Second BCD Time High
	Time Stamping
0x020	Time Stamping Control / Status
0x022	Time Stamping FIFO Super-Seconds Low
0x024	Time Stamping FIFO Super-Seconds Mid Low
0x026	Time Stamping FIFO Super-Seconds Mid High
0x028	Time Stamping FIFO Super-Seconds High
0x02A	Time Stamping FIFO Sub-Seconds Low
0x02C	Time Stamping FIFO Sub-Seconds High
	GPIO Output
0x040	GPIO Output 0 High Match Time Sub-Second Low
0x042	GPIO Output 0 High Match Time Sub-Second High
0x044	GPIO Output 0 High Match Time Super-Second Low
0x046	GPIO Output 0 High Match Time Super-Second Mid
0x048	GPIO Output 0 High Match Time Super-Second High
0x050	GPIO Output 0 Low Match Time Sub-Second Low
0x052	GPIO Output 0 Low Match Time Sub-Second High
0x054	GPIO Output 0 Low Match Time Super-Second Low
0x056	GPIO Output 0 Low Match Time Super-Second Mid
0x058	GPIO Output 0 Low Match Time Super-Second High
0x060	GPIO Output 1 High Match Time Sub-Second Low
0x062	GPIO Output 1 High Match Time Sub-Second High
0x064	GPIO Output 1 High Match Time Super-Second Low
0x066	GPIO Output 1 High Match Time Super-Second Mid
0x068	GPIO Output 1 High Match Time Super-Second High
0x070	GPIO Output 1 Low Match Time Sub-Second Low
0x072	GPIO Output 1 Low Match Time Sub-Second High
0x074	GPIO Output 1 Low Match Time Super-Second Low
0x076	GPIO Output 1 Low Match Time Super-Second Mid
0x078	GPIO Output 1 Low Match Time Super-Second High
0x080	GPIO Output 2 High Match Time Sub-Second Low
0x082	GPIO Output 2 High Match Time Sub-Second High
0x084	GPIO Output 2 High Match Time Super-Second Low
0x086	GPIO Output 2 High Match Time Super-Second Mid

Address	Name
0x088	GPIO Output 2 High Match Time Super-Second High
0x090	GPIO Output 2 Low Match Time Sub-Second Low
0x092	GPIO Output 2 Low Match Time Sub-Second High
0x094	GPIO Output 2 Low Match Time Super-Second Low
0x096	GPIO Output 2 Low Match Time Super-Second Mid
0x098	GPIO Output 2 Low Match Time Super-Second High
0x0A0	GPIO Output 3 High Match Time Sub-Second Low
0x0A2	GPIO Output 3 High Match Time Sub-Second High
0x0A4	GPIO Output 3 High Match Time Super-Second Low
0x0A6	GPIO Output 3 High Match Time Super-Second Mid
0x0A8	GPIO Output 3 High Match Time Super-Second High
0x0B0	GPIO Output 3 Low Match Time Sub-Second Low
0x0B2	GPIO Output 3 Low Match Time Sub-Second High
0x0B4	GPIO Output 3 Low Match Time Super-Second Low
0x0B6	GPIO Output 3 Low Match Time Super-Second Mid
0x0B8	GPIO Output 3 Low Match Time Super-Second High
	Host Bus Interface
0x0C0	FPGA Control / Status
0x0C2	FPGA Revision ID
0x0C4	Host Bus Interrupt Mask
0x0C6	Host Bus Interrupt Status
	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.

2.1 System Clock

2.1.1 Host Super-Second System Time Low

2.1.1.1 Address: 0x000

Bits	Description	R/W	Reset Val
3:0	Super-second system time 1s of seconds	R	0000
7:0	Super-second system time 10s of seconds	R	0000
11:8	Super-second system time 1s of minutes	R	0000
15:12	Super-second system time 10s of minutes	R	0000

Notes: Reading this register latches all of the Host Sub-Second System Time, Host Super-Second System Time, and Host Sub-Second BCD Time registers.

2.1.2 Host Super-Second System Time Mid Low

2.1.2.1 Address: 0x002

Bits	Description	R/W	Reset Val
3:0	Super-second system time 1s of hours	R	0000
7:4	Super-second system time 10s of hours	R	0000
11:8	Super-second system time 1s of days	R	0000
15:12	Super-second system time 10s of days	R	0000

Notes: This register is latched when the Host Super-Second System Time Low register is read.

2.1.3 Host Super-Second System Time Mid High

2.1.3.1 Address: 0x004

Bits	Description	R/W	Reset Val
3:0	Super-second system time 100s of days	R	0000
7:4	Super-second system time 1s of year	R	0000
11:8	Super-second system time 10s of year	R	0000
15:12	Super-second system time 100s of year	R	0000

Notes: This register is latched when the Host Super-Second System Time Low register is read.

2.1.4 Host Super-Second System Time High

2.1.4.1 Address: 0x006

Bits	Description	R/W	Reset Val
3:0	Super-second system time 1000s of year	R	0000
15:4	Reserved	-	-

Notes: This register is latched when the Host Super-Second System Time Low register is read.

2.1.5 Host Sub-Second System Time Low

2.1.5.1 Address: 0x008

Bits	Description	R/W	Reset Val
15:0	Sub-second system time low order bits (5ns resolution)	R	x0000

Notes: This register is latched when the Host Super-Second System Time Low register or the Host Super-Second Binary Time Low register is read.

2.1.6 Host Sub-Second System Time High

2.1.6.1 Address: 0x00A

Bits	Description	R/W	Reset Val
11:0	Sub-second system time high order bits (5ns resolution)	R	x000
14:12	Reserved	-	-
15	System synchronization status: 0 = System Not in Sync 1 = System in Sync	R	0

Notes: This register is latched when the Host Super-Second System Time Low register or the Host Super-Second Binary Time Low register is read.

2.1.7 Host Super-Second Binary Time Low

2.1.7.1 Address: 0x00C

Bits	Description	R/W	Reset Val
15:0	Super-second binary time low order bits	R	x0000

Notes: Reading this register latches all of the Host Sub-Second System Time, Host Super-Second Binary Time, and Host Sub-Second BCD Time registers.

2.1.8 Host Super-Second Binary Time High

2.1.8.1 Address: 0x00E

Bits	Description	R/W	Reset Val
15:0	Super-second binary time high order bits	R	x0000

Notes: This register is latched when the Host Super-Second Binary Time Low register is read.

2.1.9 Host Sub-Second BCD Time Low

2.1.9.1 Address: 0x010

Bits	Description	R/W	Reset Val
3:0	Sub-second BCD time 1s of microseconds	R	0000
7:4	Sub-second BCD time 10s of microseconds	R	0000
11:8	Sub-second BCD time 100s of microseconds	R	0000
15:12	Reserved	-	-

Notes: This register is latched when the Host Super-Second System Time Low register or the Host Super-Second Binary Time Low register is read.

2.1.10 Host Sub-Second BCD Time High

2.1.10.1 Address: 0x012

Bits	Description	R/W	Reset Val
3:0	Sub-second BCD time 1s of milliseconds	R	0000
7:4	Sub-second BCD time 10s of milliseconds	R	0000
11:8	Sub-second BCD time 100s of milliseconds	R	0000
15:12	Reserved	-	-

Notes: This register is latched when the Host Super-Second System Time Low register or the Host Super-Second Binary Time Low register is read.

2.2 Time Stamping

2.2.1 Time Stamping Control / Status

2.2.1.1 Address: 0x020

Bits	Description	R/W	Reset Val
0	Enable time stamping: 0 = Disable 1 = Enable	R/W	0
1	Timestamp request: 0 = No Effect 1 = Generate timestamp request (Automatically cleared by circuit)	W	-
2:3	Reserved	-	-
4	Timestamp FIFO clear: 0 = No Effect 1 = Clear FIFO (Automatically cleared by circuit)	W	-
5	Timestamp FIFO empty: 0 = Data Available 1 = Empty	R	1
6	Timestamp FIFO full: 0 = Space Available 1 = Full	R	0
7	Timestamp FIFO overflow: (Value remains until cleared) 0 = Normal 1 = Overflow (Write a 1 to clear)	R/W	0
8	Timestamp FIFO half full: 0 = Less than Half Full 1 = More than Half Full	R	0
15:9	Reserved	-	-

Notes:

2.2.2 Time Stamping FIFO Super-Seconds Low

2.2.2.1 Address: 0x022

Bits	Description	R/W	Reset Val
3:0	Super-second system time 1s of seconds	R	0000
7:0	Super-second system time 10s of seconds	R	0000
11:8	Super-second system time 1s of minutes	R	0000
15:12	Super-second system time 10s of minutes	R	0000

Notes: The entire timestamp is read out of the FIFO into the registers when the time stamping FIFO super-second low order bits register is read.

2.2.3 Time Stamping FIFO Super-Seconds Mid Low

2.2.3.1 Address: 0x024

Bits	Description	R/W	Reset Val
3:0	Super-second system time 1s of hours	R	0000
7:4	Super-second system time 10s of hours	R	0000
11:8	Super-second system time 1s of days	R	0000
15:12	Super-second system time 10s of days	R	0000

Notes: The entire timestamp is read out of the FIFO into the registers when the time stamping FIFO super-second low order bits register is read.

2.2.4 Time Stamping FIFO Super-Seconds Mid High

2.2.4.1 Address: 0x026

Bits	Description	R/W	Reset Val
3:0	Super-second system time 100s of days	R	0000
7:4	Super-second system time 1s of year	R	0000
11:8	Super-second system time 10s of year	R	0000
15:12	Super-second system time 100s of year	R	0000

Notes: The entire timestamp is read out of the FIFO into the registers when the time stamping FIFO super-second low order bits register is read.

2.2.5 Time Stamping FIFO Super-Seconds High

2.2.5.1 Address: 0x028

Bits	Description	R/W	Reset Val
3:0	Super-second system time 1000s of year	R	0000
4	Manual requested timestamp flag	R	0
5	GPIO input 0 timestamp flag	R	0
6	GPIO input 1 timestamp flag	R	0
7	GPIO input 2 timestamp flag	R	0
8	GPIO input 3 timestamp flag	R	0
15:9	Reserved	-	-

Notes: The entire timestamp is read out of the FIFO into the registers when the time stamping FIFO super-second low order bits register is read. The timestamp requester flags represent the time stamping sources for this timestamp.

2.2.6 Time Stamping FIFO Sub-Seconds Low

2.2.6.1 Address: 0x02A

Bits	Description	R/W	Reset Val
15:0	Timestamp sub-second low order bits (5ns resolution)	R	x0000

Notes: The entire timestamp is read out of the FIFO into the registers when the time stamping FIFO super-second low order bits register is read.

2.2.7 Time Stamping FIFO Sub-Seconds High

2.2.7.1 Address: 0x02C

Bits	Description	R/ W	Reset Val
11:0	Timestamp sub-second high order bits (5ns resolution)	R	x000
14:12	Reserved	-	-
15	System synchronization status: 0 = System Not in Sync 1 = System in Sync	R	0

Notes: The entire timestamp is read out of the FIFO into the registers when the time stamping FIFO super-second low order bits register is read.

2.3 GPIO Output

2.3.1 GPIO Output 0 High Match Time Sub-Second Low

2.3.1.1 Address: 0x040

Bits	Description	R/W	Reset Val
15:0	GPIO output high match time sub-second low order bits (5ns resolution)	R/W	x0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.2 GPIO Output 0 High Match Time Sub-Second High

2.3.2.1 Address: 0x042

Bits	Description	R/W	Reset Val
11:0	GPIO output high match time sub-second high order bits (5ns resolution)	R/W	x0000
15:12	Reserved	-	-

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.3 GPIO Output 0 High Match Time Super-Second Low

2.3.3.1 Address: 0x044

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 1s of seconds	R/W	0000
7:0	GPIO output high match time 10s of seconds	R/W	0000
11:8	GPIO output high match time 1s of minutes	R/W	0000
15:12	GPIO output high match time 10s of minutes	R/W	0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.4 GPIO Output 0 High Match Time Super-Second Mid

2.3.4.1 Address: 0x046

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 1s of hours	R/W	0000
7:0	GPIO output high match time 10s of hours	R/W	0000
11:8	GPIO output high match time 1s of days	R/W	0000
15:12	GPIO output high match time 10s of days	R/W	0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.5 GPIO Output 0 High Match Time Super-Second High

2.3.5.1 Address: 0x048

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 100s of days	R/W	0000
15:4	Reserved	-	-

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.6 GPIO Output 0 Low Match Time Sub-Second Low

2.3.6.1 Address: 0x050

Bits	Description	R/W	Reset Val
15:0	GPIO output low match time sub-second low order bits (5ns resolution)	R/W	x0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.7 GPIO Output 0 Low Match Time Sub-Second High

2.3.7.1 Address: 0x052

Bits	Description	R/W	Reset Val
11:0	GPIO output low match time sub-second high order bits (5ns resolution)	R/W	x0000
15:12	Reserved	-	-

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.8 GPIO Output 0 Low Match Time Super-Second Low

2.3.8.1 Address: 0x054

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 1s of seconds	R/W	0000
7:0	GPIO output low match time 10s of seconds	R/W	0000
11:8	GPIO output low match time 1s of minutes	R/W	0000
15:12	GPIO output low match time 10s of minutes	R/W	0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.9 GPIO Output 0 Low Match Time Super-Second Mid

2.3.9.1 Address: 0x056

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 1s of hours	R/W	0000
7:0	GPIO output low match time 10s of hours	R/W	0000
11:8	GPIO output low match time 1s of days	R/W	0000
15:12	GPIO output low match time 10s of days	R/W	0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.10 GPIO Output 0 Low Match Time Super-Second High

2.3.10.1 Address: 0x058

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 100s of days	R/W	0000
15:4	Reserved	-	-

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.11 GPIO Output 1 High Match Time Sub-Second Low

2.3.11.1 Address: 0x060

Bits	Description	R/W	Reset Val
15:0	GPIO output high match time sub-second low order bits (5ns resolution)	R/W	x0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.12 GPIO Output 1 High Match Time Sub-Second High

2.3.12.1 Address: 0x062

Bits	Description	R/W	Reset Val
11:0	GPIO output high match time sub-second high order bits (5ns resolution)	R/W	x0000
15:12	Reserved	-	-

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.13 GPIO Output 1 High Match Time Super-Second Low

2.3.13.1 Address: 0x064

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 1s of seconds	R/W	0000
7:0	GPIO output high match time 10s of seconds	R/W	0000
11:8	GPIO output high match time 1s of minutes	R/W	0000
15:12	GPIO output high match time 10s of minutes	R/W	0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.14 GPIO Output 1 High Match Time Super-Second Mid

2.3.14.1 Address: 0x066

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 1s of hours	R/W	0000
7:0	GPIO output high match time 10s of hours	R/W	0000
11:8	GPIO output high match time 1s of days	R/W	0000
15:12	GPIO output high match time 10s of days	R/W	0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.15 GPIO Output 1 High Match Time Super-Second High

2.3.15.1 Address: 0x068

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 100s of days	R/W	0000
15:4	Reserved	-	-

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.16 GPIO Output 1 Low Match Time Sub-Second Low

2.3.16.1 Address: 0x070

Bits	Description	R/W	Reset Val
15:0	GPIO output low match time sub-second low order bits (5ns resolution)	R/W	x0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.17 GPIO Output 1 Low Match Time Sub-Second High

2.3.17.1 Address: 0x072

Bits	Description	R/W	Reset Val
11:0	GPIO output low match time sub-second high order bits (5ns resolution)	R/W	x0000
15:12	Reserved	-	-

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.18 GPIO Output 1 Low Match Time Super-Second Low

2.3.18.1 Address: 0x074

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 1s of seconds	R/W	0000
7:0	GPIO output low match time 10s of seconds	R/W	0000
11:8	GPIO output low match time 1s of minutes	R/W	0000
15:12	GPIO output low match time 10s of minutes	R/W	0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.19 GPIO Output 1 Low Match Time Super-Second Mid

2.3.19.1 Address: 0x076

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 1s of hours	R/W	0000
7:0	GPIO output low match time 10s of hours	R/W	0000
11:8	GPIO output low match time 1s of days	R/W	0000
15:12	GPIO output low match time 10s of days	R/W	0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.20 GPIO Output 1 Low Match Time Super-Second High

2.3.20.1 Address: 0x078

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 100s of days	R/W	0000
15:4	Reserved	-	-

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.21 GPIO Output 2 High Match Time Sub-Second Low

2.3.21.1 Address: 0x080

Bits	Description	R/W	Reset Val
15:0	GPIO output high match time sub-second low order bits (5ns resolution)	R/W	x0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.22 GPIO Output 2 High Match Time Sub-Second High

2.3.22.1 Address: 0x082

Bits	Description	R/W	Reset Val
11:0	GPIO output high match time sub-second high order bits (5ns resolution)	R/W	x0000
15:12	Reserved	-	-

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.23 GPIO Output 2 High Match Time Super-Second Low

2.3.23.1 Address: 0x084

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 1s of seconds	R/W	0000
7:0	GPIO output high match time 10s of seconds	R/W	0000
11:8	GPIO output high match time 1s of minutes	R/W	0000
15:12	GPIO output high match time 10s of minutes	R/W	0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.24 GPIO Output 2 High Match Time Super-Second Mid

2.3.24.1 Address: 0x086

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 1s of hours	R/W	0000
7:0	GPIO output high match time 10s of hours	R/W	0000
11:8	GPIO output high match time 1s of days	R/W	0000
15:12	GPIO output high match time 10s of days	R/W	0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.25 GPIO Output 2 High Match Time Super-Second High

2.3.25.1 Address: 0x088

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 100s of days	R/W	0000
15:4	Reserved	-	-

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.26 GPIO Output 2 Low Match Time Sub-Second Low

2.3.26.1 Address: 0x090

Bits	Description	R/W	Reset Val
15:0	GPIO output low match time sub-second low order bits (5ns resolution)	R/W	x0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.27 GPIO Output 2 Low Match Time Sub-Second High

2.3.27.1 Address: 0x092

Bits	Description	R/W	Reset Val
11:0	GPIO output low match time sub-second high order bits (5ns resolution)	R/W	x0000
15:12	Reserved	-	-

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.28 GPIO Output 2 Low Match Time Super-Second Low

2.3.28.1 Address: 0x094

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 1s of seconds	R/W	0000
7:0	GPIO output low match time 10s of seconds	R/W	0000
11:8	GPIO output low match time 1s of minutes	R/W	0000
15:12	GPIO output low match time 10s of minutes	R/W	0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.29 GPIO Output 2 Low Match Time Super-Second Mid

2.3.29.1 Address: 0x096

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 1s of hours	R/W	0000
7:0	GPIO output low match time 10s of hours	R/W	0000
11:8	GPIO output low match time 1s of days	R/W	0000
15:12	GPIO output low match time 10s of days	R/W	0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.30 GPIO Output 2 Low Match Time Super-Second High

2.3.30.1 Address: 0x098

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 100s of days	R/W	0000
15:4	Reserved	-	-

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.31 GPIO Output 3 High Match Time Sub-Second Low

2.3.31.1 Address: 0x0A0

Bits	Description	R/W	Reset Val
15:0	GPIO output high match time sub-second low order bits (5ns resolution)	R/W	x0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.32 GPIO Output 3 High Match Time Sub-Second High

2.3.32.1 Address: 0x0A2

Bits	Description	R/W	Reset Val
11:0	GPIO output high match time sub-second high order bits (5ns resolution)	R/W	x0000
15:12	Reserved	-	-

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.33 GPIO Output 3 High Match Time Super-Second Low

2.3.33.1 Address: 0x0A4

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 1s of seconds	R/W	0000
7:0	GPIO output high match time 10s of seconds	R/W	0000
11:8	GPIO output high match time 1s of minutes	R/W	0000
15:12	GPIO output high match time 10s of minutes	R/W	0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.34 GPIO Output 3 High Match Time Super-Second Mid

2.3.34.1 Address: 0x0A6

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 1s of hours	R/W	0000
7:0	GPIO output high match time 10s of hours	R/W	0000
11:8	GPIO output high match time 1s of days	R/W	0000
15:12	GPIO output high match time 10s of days	R/W	0000

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.35 GPIO Output 3 High Match Time Super-Second High

2.3.35.1 Address: 0x0A8

Bits	Description	R/W	Reset Val
3:0	GPIO output high match time 100s of days	R/W	0000
15:4	Reserved	-	-

Notes: GPIO output will set to high level when system time matches the high match time value.

2.3.36 GPIO Output 3 Low Match Time Sub-Second Low

2.3.36.1 Address: 0x0B0

Bits	Description	R/W	Reset Val
15:0	GPIO output low match time sub-second low order bits (5ns resolution)	R/W	x0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.37 GPIO Output 3 Low Match Time Sub-Second High

2.3.37.1 Address: 0x0B2

Bits	Description	R/W	Reset Val
11:0	GPIO output low match time sub-second high order bits (5ns resolution)	R/W	x0000
15:12	Reserved	-	-

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.38 GPIO Output 3 Low Match Time Super-Second Low

2.3.38.1 Address: 0x0B4

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 1s of seconds	R/W	0000
7:0	GPIO output low match time 10s of seconds	R/W	0000
11:8	GPIO output low match time 1s of minutes	R/W	0000
15:12	GPIO output low match time 10s of minutes	R/W	0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.39 GPIO Output 3 Low Match Time Super-Second Mid

2.3.39.1 Address: 0x0B6

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 1s of hours	R/W	0000
7:0	GPIO output low match time 10s of hours	R/W	0000
11:8	GPIO output low match time 1s of days	R/W	0000
15:12	GPIO output low match time 10s of days	R/W	0000

Notes: GPIO output will set to low level when system time matches the low match time value.

2.3.40 GPIO Output 3 Low Match Time Super-Second High

2.3.40.1 Address: 0x0B8

Bits	Description	R/W	Reset Val
3:0	GPIO output low match time 100s of days	R/W	0000
15:4	Reserved	-	-

Notes: GPIO output will set to low level when system time matches the low match time value.

2.4 Host Bus Interface

2.4.1 FPGA Control / Status

2.4.1.1 Address: 0x0C0

Bits	Description	R/W	Reset Val
7:0	Reserved	-	-
15:8	FPGA part number	R	x00

Notes: FPGA part number is x00 because only one FPGA exists in TSync-PCIe.

2.4.2 FPGA Revision ID

2.4.2.1.1 Address: 0x0C2

Bits	Description	R/W	Reset Val
15:0	FPGA revision ID	R	(Rev ID)

Notes: FPGA revision ID is the revision of the FPGA load currently running.

2.4.3 Host Bus Interrupt Mask

2.4.3.1 Address: 0x0C4

Bits	Description	R/W	Reset Val
0	1PPS received	R/W	1
1	Timing System status	R/W	1
2	Host / µC bus FIFO empty	R/W	1
3	Host / µC bus FIFO overflow	R/W	1
4	µC / host bus FIFO data available	R/W	1
5	µC / host bus FIFO overflow	R/W	1
6	GPIO input 0 event	R/W	1
7	GPIO input 1 event	R/W	1
8	GPIO input 2 event	R/W	1
9	GPIO input 3 event	R/W	1
10	Timestamp occurred	R/W	1
11	GPIO output 0 event	R/W	1
12	GPIO output 1 event	R/W	1
13	GPIO output 2 event	R/W	1
14	GPIO output 3 event	R/W	1
15	Reserved	-	-

Notes: Writing a 1 masks the source from generating an interrupt to the host bus. Writing a 0 allows the interrupt to signal the host bus. The current state of the interrupt source is always available in the Interrupt Status register, whether or not the interrupt is masked to the host bus.

2.4.4 Host Bus Interrupt Status

2.4.4.1 Address: 0x0C6

Bits	Description	R/W	Reset Val
0	1PPS received (Write a 1 to clear)	R/W	0
1	Timing System status (Write a 1 to clear)	R/W	0
2	Host bus / µC FIFO overflow (Write a 1 to clear)	R/W	0
3	Host bus / µC FIFO empty (Write a 1 to clear)	R/W	0
4	µC / host bus FIFO data available (Write a 1 to clear)	R/W	0
5	µC / host bus FIFO overflow (Write a 1 to clear)	R/W	0
6	GPIO input 0 event (Write a 1 to clear)	R/W	0
7	GPIO input 1 event (Write a 1 to clear)	R/W	0
8	GPIO input 2 event (Write a 1 to clear)	R/W	0
9	GPIO input 3 event (Write a 1 to clear)	R/W	0
10	Timestamp occurred (Write a 1 to clear)	R/W	0
11	GPIO output 0 event (Write a 1 to clear)	R/W	0
12	GPIO output 1 event (Write a 1 to clear)	R/W	0
13	GPIO output 2 event (Write a 1 to clear)	R/W	0
14	GPIO output 3 event (Write a 1 to clear)	R/W	0
15:	Reserved	-	-

Notes: The current state of the interrupt source is always available in the Interrupt Status register, whether or not the interrupt is masked to the host bus.

2.5 Shared Memory

2.5.1 Host / Microcontroller Bus FIFO Control / Status

2.5.1.1 Address: 0x160

Bits	Description	R/W	Reset Val
0	µC / host bus FIFO clear: 0 = No Effect 1 = Clear FIFO (Automatically cleared by circuit)	W	-
1	µC / host bus FIFO empty: 0 = Data Available 1 = Empty	R	1
2	µC / host bus FIFO full: 0 = Space Available 1 = Full	R	0
3	µC / host bus FIFO overflow: 0 = Normal 1 = Overflow (Write a 1 to clear)	R/W	0
4	Host / µC bus FIFO clear: 0 = No Effect 1 = Clear FIFO (Automatically cleared by circuit)	W	-
5	Host / µC bus FIFO empty: 0 = Data Available 1 = Empty	R	1
6	Host / µC bus FIFO full: 0 = Space Available 1 = Full	R	0
7	Host / µC bus FIFO overflow: 0 = Normal 1 = Overflow (Write a 1 to clear)	R/W	0
15:8	Reserved	-	-

Notes:

2.5.2 Host / Microcontroller Bus FIFO Data

2.5.2.1 Address: 0x180

Bits	Description	R/W	Reset Val
15:0	Host / µC bus FIFO data	W	-

Notes:

2.5.3 Microcontroller / Host Bus FIFO Data

2.5.3.1 Address: 0x1C0

Bits	Description	R/W	Reset Val
15:0	µC / host bus FIFO data	R	x0000

Notes:

2.6 Interrupts

The Host Bus has one interrupt line available from the TSync device. 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 register.

2.6.1 Interrupt Descriptions

2.6.1.1 1PPS Received

This interrupt is driven on the incident edge of the PPS.

2.6.1.2 Timing System Service Request

This interrupt is used by the micro to request attention from the local bus.

2.6.1.3 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.

2.6.1.4 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.

2.6.1.5 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.

2.6.1.6 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.

2.6.1.7 GPIO Input x Event

This interrupt is driven when the active edge of the GPIO input signal is received.

2.6.1.8 GPIO Output x Event

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.

2.6.1.9 Timestamp Data Available

This interrupt is driven when the timestamp FIFO goes non-empty. Timestamp data is available in the timestamp FIFO when this interrupt occurs.

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.

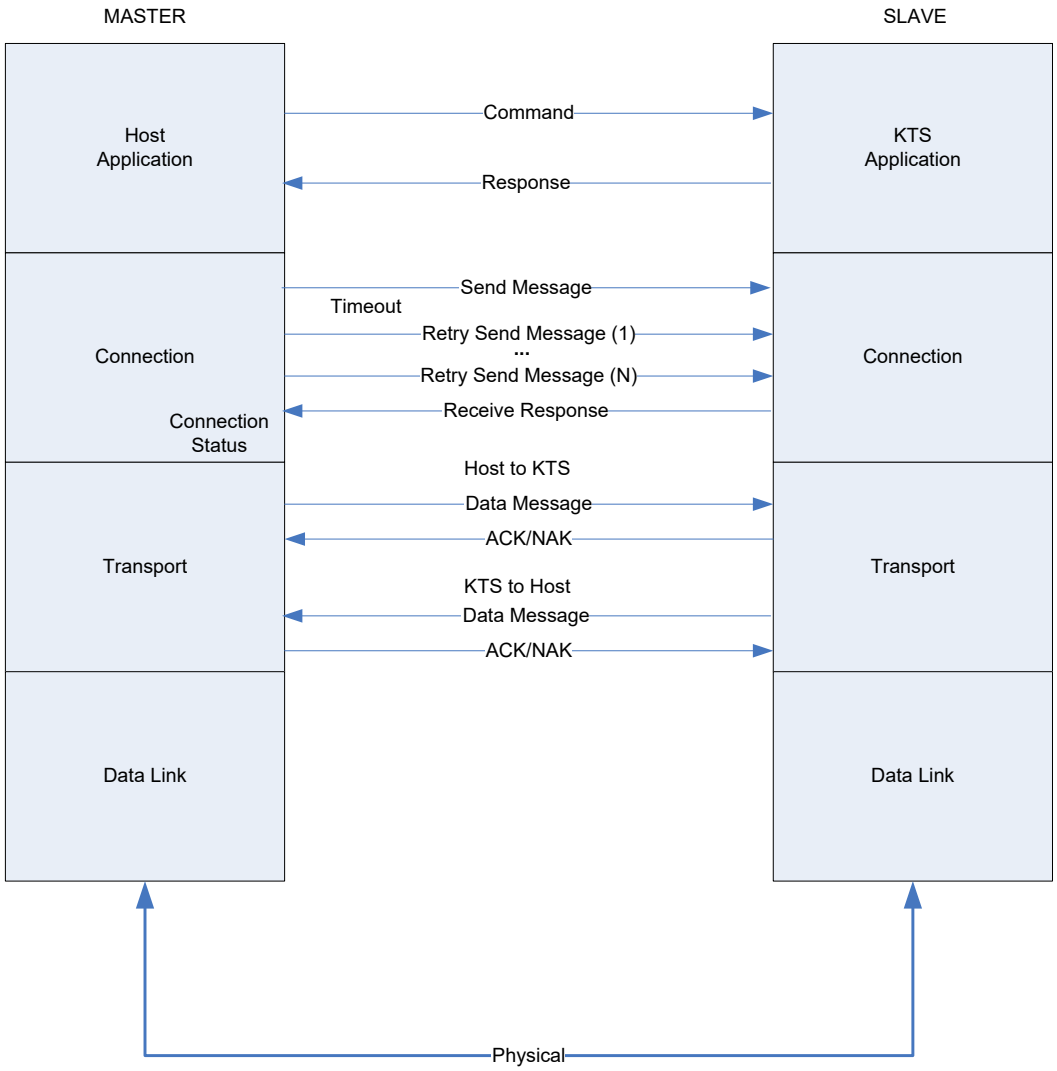

Figure 3-1 — Layers

3.1 Application Layer

The application layer consists of the protocol definition of the commands. The protocol is defined in terms of messages, which are sent from the master to the slave, and responses, which are sent from the slave to the master. This layer requires that a connection be established between the master and the slave that provides a reliable and error free means of communication. "Reliable" means that messages sent are received. "Error free" implies data integrity, which means that the data sent is the same as the data received.

The application layer sends command messages from the host to the slave using the connection layer. It expects the host to receive a response message from the slave or an error condition indicating that the connection layer was unable to communicate with the slave.

3.2 Connection Layer

The connection layer consists of the protocol primitives which allow the transmission and receipt of protocol messages. It is responsible for establishing a channel between the master and the slave and is only concerned with maintaining that connection. It is implemented to send command messages from the master to the slave and expects response messages in all cases. Failure to receive a response message within a defined timeout interval may cause the command message to be resent up to N times.

If the connection layer fails to send a command from the host to the slave, it returns an error to the application layer. If a response message is received on the host from the slave, it is returned to the application layer if it is received before the timeout expires.

The connection layer assumes that the transport layer and the layers beneath it provide both reliable transport and data integrity.

3.3 Transport Layer

The transport layer is responsible for ensuring a reliable transport of data. Command messages from the master to the slave are acknowledged using an ACK/NAK message. Response messages from the slave to the master are also acknowledged using an ACK/NAK message. Data integrity is maintained by requiring all command or response messages to have some form of a checksum. If a checksum is found to be invalid, a NAK message is sent, otherwise an ACK message is sent. If no ACK/NAK message is received or the ACK/NAK message is corrupted, the transaction will fail due to timeout.

The following bus technologies implement the transport layer and below, thus eliminating the need for the HIP to implement these layers: PCI, PCIe, cPCI, cPCIe, PMC, ISA, USB, VME32, VME64, Ethernet, CAN, IP (Industrial Protocol bus).

3.4 Data Link Layer

The data link layer is the low-level communications technology enabling the above layers to communicate across the physical medium. The data link layer consists of the low-level software device drivers and logic devices required to drive the physical layer. This layer is configured during initialization of the above layers to provide the required bandwidth and data transfer characteristics.

The following bus technologies implement only the data link layer and do not provide reliable transport or data integrity: I2C, SPI, Serial (RS-232/RS-485/RS-422). They require the HIP to implement the transport layer; providing reliable transport and data integrity.

4 Host Interface Protocol (HIP)—Details

The HIP is described in detail in this section. The responsibilities and capabilities provided at each level are specified.

4.1 Application Layer

The application layer is the most abstract definition and usage of the HIP. At this level, multiple TSync devices may be controlled by a single host. This layer must define the command messages sent from the host to the TSync device and the response messages received by the host from the TSync device.

4.1.1 Features

- Commands
- Responses
- Connection Status API
  - Read configuration
  - Read connection status
  - Configure
- Error Conditions
  - Command timeouts
  - Loss of connection

4.1.2 Layer Details

1. Protocol is master-slave protocol.
2. The host must open a connection to a TSync device in order to communicate.
3. The host may communicate with multiple TSync devices at a time over separate connections.
4. The host may optionally configure connection-status polling by the connection layer to verify the connection is still active.
5. Half-duplex operation is required because only one command can be performed on a given TSync device at a time.
6. The host expects a response for every command sent.
7. The host must examine response messages for command status.
8. The host only receives timeout errors on failures indicating an inability to communicate with a TSync device or connection lost status.
9. Timeouts are long, relative to lower-level protocol interaction timeouts, to limit the need for collision detection between master and slave devices.

4.1.3 Message Details

Message details for the application layer, including message formats and message definitions, are contained herein.

4.2 Connection Layer

The connection layer is responsible for maintaining a logical connection between the host and the TSync device. It performs retries on messages from the host to the TSync device and returns failures due to timeouts. It provides an optional API to allow the connection status to be monitored via health messages and responses between the host and the TSync device. The connection status is determined from the health response message from the TSync device or if the message times out.

4.2.1 Features

- Send Message API
- Receive Message API
- Detect failures from transport
- Performs N retries in case of timeout
- Configurable command message timeout
- Connection Status API
  - Enable/Disable monitoring of connection status
  - Poll rate for host to contact TSync device
  - Connection status
- Affected by connection timeout
- Return status from health response

4.2.2 Layer Details

1. Health message and health response are defined in this layer.
2. The command timeout and retry count are defined at this level and may be configured from the application layer.
3. Connection Status API allows for enabling, setting poll rate, and configuring connection timeout.

4.2.3 Message Details

4.2.3.1 Data Types and Definitions

```
typedef struct
{
    UINT8  id;                  // Message ID
    UINT8  seqNum;              // Sequence Number
    UINT16 len;                 // Payload Length

} HL_CL_MSG_HDR;
```

The **HL_CL_MSG_HDR** type is used to define the common message header used by all command and response messages. A message identifier contains the message identifier and message direction to identify different messages. The sequence number is simply a byte value that increments upward with every new message sent (not retries). The sequence numbers are used to verify that the command and response messages match. The length field contains the payload length of the message in bytes.

```
typedef enum
{
    HL_CL_APPL = 0x01,          // Application Command/Response
    HL_CL_HLTH = 0x02           // Connection Health Command/Response

} HL_CL_MSG_ID;
```

The **HL_CL_MSG_ID** type defines the valid message IDs found in the least significant nibble of the message ID field.

```
typedef enum
{
    HL_CL_M2S = 0x00,           // Master to slave
    HL_CL_S2M = 0x80            // Slave to master

} HL_CL_MSG_DIR;
```

The **HL_CL_MSG_DIR** type defines the direction codes found in the most significant nibble of the message ID field.

```
typedef struct
{
    HL_CL_MSG_HDR hdr;          // Message header
    UINT8         payload[];    // Message payload
    UINT16        chksum;       // Checksum

} HL_CL_MSG_TYPE;
```

The **HL_CL_MSG_TYPE** type is used to define the common message structure used by all command and response messages. The message consists of a header, an optional payload, and a trailer. The trailer consists of a 2-byte checksum.

4.2.3.2 Message Formats

Application Command

An application command is used to encapsulate application layer packets from the host to the TSync device. The message consists of a message header and a variable length payload. The payload contains an entire application layer packet and is passed to the application layer. The sequence number is incremented for each new application command, while remaining the same to indicate a command retry.

Offset	Field	TYPE	Description
0	0x01	UINT8	Application Command ID
1	seqNum	UINT8	Sequence Number
2	len	UINT16	Payload Length
4	payload	UINT8[len]	Payload
len+4	chksum	UINT16	Checksum

Application Response

An application command is used to encapsulate application layer packets from the TSync device to the host. The response message consists of a message header and a variable length payload. The payload contains an entire application layer response packet that is passed back from the application layer. The sequence number for the response is the same as the command message that it resulted from.

Offset	Field	TYPE	Description
0	0x81	UINT8	Application Response ID
1	seqNum	UINT8	Sequence Number
2	len	UINT16	Payload Length
4	payload	UINT8[len]	Payload
len+4	chksum	UINT16	Checksum

Connection Health Command

A connection health command packet is sent from the host to the TSync device to monitor the health of the connection between them. The message consists of a message header only. The sequence number is incremented for each new application command, while remaining the same to indicate a command retry.

Offset	Field	TYPE	Description
0	0x02	UINT8	Connection Health Command ID
1	seqNum	UINT8	Sequence Number
2	0	UINT16	Payload Length
4	chksum	UINT16	Checksum

Connection Health Response

A connection health response packet is sent from the TSync device to the host in response toa connection health command. The message consists of a message header only. The sequence number for the response is the same as the command message that it resulted from.

Offset	Field	TYPE	Description
0	0x82	UINT8	Connection Health Response ID
1	seqNum	UINT8	Sequence Number
2	0	UINT16	Payload Length
4	chksum	UINT16	Checksum

4.3 Transport Layer

The transport layer is responsible for providing a reliable transport and data integrity between the host and the TSync device. It provides the ability for both the host and the TSync device to send a message and receive an acknowledgement. On transmit; it generates a checksum value which is appended to the message in the transport trailer. On receipt, it verifies messages by generating a checksum and matching it against the checksum in the transport trailer. The transport layer utilizes timeouts and sequence counts to ensure that message acknowledgements correspond to the messages sent.

4.3.1 Features

- Send Message API
- Receive message API
- Message receiver sends an ACK/NAK with matching sequence number
- Timeout if no ACK/NAK received
- Generates checksum for sent messages
- Verifies checksum for received messages
- Provides sequence number in all messages

4.3.2 Layer Details

1. Collision avoidance is built-in for the case when the TSync device sends a response message after the timeout has expired and the host sends another command.
2. Checksums can be implemented as linear checksums or CRCs depending on the transport medium error rates.
3. All transport implementations use network byte order (big-endian).
4. The host and the TSync device maintain separate sequence numbers.

4.3.3 Message Details

4.3.3.1 Data Types and Definitions

```
typedef struct
{
    UINT8  id;                  // Message ID
    UINT8  seqNum;              // Sequence Number
    UINT16 len;                 // Payload Length
    UINT8  payload[];           // Message payload
    UINT16 crc;                 // CRC

} HL_TL_DATA_MSG_TYPE;
```

The **HL_TL_DATA_MSG_TYPE** type is used to define the data message consisting of a header, a payload, and a trailer. The header consists of a message ID, sequence number, and a payload length (in bytes). The trailer consists of a 2-byte checksum or CRC.

```
Typedef struct
```

```
{
    UINT8  id;                  // Message ID
    UINT8  seqNum;              // Sequence Number
    UINT16 crc;                 // CRC

} HL_TL_ACK_MSG_TYPE;
```

The **HL_TL_ACK_MSG_TYPE** type is used to define the ACK message consisting of a header, a payload, and a trailer. The header consists of the message ID. The payload consists of the sequence number of the message being acknowledged. The trailer consists of a 2-byte checksum or CRC.

```
Typedef struct
{
    UINT8  id;                  // Message ID
    UINT8  seqNum;              // Sequence Number
    UINT16 crc;                 // CRC

} HL_TL_NAK_MSG_TYPE;
```

The **HL_TL_NAK_MSG_TYPE** type is used to define the NAK message consisting of a header, a payload, and a trailer. The header consists of the message ID. The payload consists of the sequence number of the message being acknowledged. The trailer consists of a 2-byte checksum or CRC.

```
Typedef enum
{
    HL_TL_DATA = 0x01,          // Data Message
    HL_TL_ACK  = 0x05,          // ACK Message
    HL_TL_NAK  = 0x0E           // NAK message

} HL_TL_MSG_ID;
```

The **HL_TL_MSG_ID** type defines the valid message IDs found in the least significant nibble of the message ID field.

```
Typedef enum
{
    HL_TL_M2S = 0x00,           // Master to Slave
    HL_TL_S2M = 0x80            // Slave to Master

} HL_TL_MSG_DIR;
```

The **HL_TL_MSG_DIR** type defines the direction codes found in the most
// significant nibble of the message ID field.

```
Typedef union
{
    HL_TL_DATA_MSG_TYPE data;   // Data message
    HL_TL_ACK_MSG_TYPE  ack;    // ACK message
    HL_TL_NAK_MSG_TYPE  nak;    // NAK message
```

```
} HL_TL_PKT;
```

The **HL_TL_PKT** type is a union of all the valid message types.

4.3.3.2 Message Formats

Host to TSync Data Message

A host to TSync data message packet is used to encapsulate connection layer packets from the host to the TSync device. The message consists of a message ID, a sequence number, a payload length, a variable length payload, and a checksum or CRC. The sequence number is incremented for each new data message. The sequence number remains the same to indicate a message retry.

Offset	Field	TYPE	Description
0	0x01	UINT8	Host to TSync Data Message ID
1	seqNum	UINT8	Sequence Number
2	len	UINT16	Payload Length
4	payload	UINT8[len]	Payload
len+4	crc	UINT16	Checksum/CRC

TSync to Host Data Message

A TSync to host data message packet is used to encapsulate connection layer response packets from the TSync device to the host. The message consists of a message ID, a sequence number, a payload length, a variable length payload, and a checksum or CRC. The sequence number is incremented for each new data message. The sequence number remains the same to indicate a message retry.

Offset	Field	TYPE	Description
0	0x81	UINT8	TSync to Host Data Message ID
1	seqNum	UINT8	Sequence Number
2	len	UINT16	Payload Length
4	payload	UINT8[len]	Payload
len+4	crc	UINT16	Checksum/CRC

Host to TSync ACK

A host to TSync ACK packet is sent from the host to the TSync device to acknowledge the receipt of a good data message packet. The packet consists of a message ID, a sequence number, and a checksum or CRC. The sequence number is the same as the data message packet being acknowledged.

Offset	Field	TYPE	Description
0	0x05	UINT8	Host to TSync ACK ID
1	seqNum	UINT8	Sequence Number
2	crc	UINT16	Checksum/CRC

TSync to Host ACK

A TSync to host ACK packet is sent from the TSync device to the host to acknowledge the receipt of a good data message packet. The packet consists of a message ID, a sequence number, and a checksum or CRC. The sequence number is the same as the data message packet being acknowledged.

Offset	Field	TYPE	Description
0	0x85	UINT8	Host to TSync ACK ID
1	seqNum	UINT8	Sequence Number
2	crc	UINT16	Checksum/CRC

Host to TSync NAK

A host to TSync NAK packet is sent from the host to the TSync device to acknowledge the receipt of a bad data message packet. The packet consists of a message ID, a sequence number, and a checksum or CRC. The sequence number is the same as the data message packet being acknowledged.

Offset	Field	TYPE	Description
0	0x0E	UINT8	Host to TSync ACK ID
1	seqNum	UINT8	Sequence Number
2	crc	UINT16	Checksum/CRC

TSync to Host ACK

A TSync to host NAK packet is sent from the TSync device to the host to acknowledge the receipt of a bad data message packet. The packet consists of a message ID, a sequence number, and a checksum or CRC. The sequence number is the same as the data message packet being acknowledged.

Offset	Field	TYPE	Description
0	0x8E	UINT8	Host to TSync ACK ID
1	seqNum	UINT8	Sequence Number
2	crc	UINT16	Checksum/CRC

5 Message Transactions

This section describes the messages that are transmitted between the TSync Host and the Host Agent (HA).

The responsibility for the command-response message protocol state machine resides entirely in the HA. The Host Interface message protocol is a master-slave, half-duplex protocol which requires a response for every message received. The Host is the initiator of all message transactions. A message transaction consists of a command sent by the Host to the TSync device and a response sent by the TSync device to the Host. The HA is the recipient of all messages and returns all responses. The HA acts on each received command. Most command messages are designed to act on components. The Command message is received by the HA, which uses the Component Interface (CI) to execute a call to a registered API on the component. Some command messages are intended to be executed by the HA. These commands exercise built-in HA functionality.

The format of commands and responses is component- or agent-operation specific. All messages consist of a header and a payload. The header identifies the message and provides a means to pass control information. The payload is optional and carries parameters specific to each operation. Command message payloads may contain parameters required to execute an operation on a component. Response message payloads may also contain parameters containing data returned to the host. Response messages return a mandatory error code payload on error and set a control field indicating that the component or agent operation failed. Error response messages may optionally return payload parameters following the error code.

The host interface transfers all messages in big-endian (network) byte order. This translation is normally performed at the lowest level in the implementation where the message contents are known. However, because our microprocessor is big-endian and portability is not a concern no translation functions will be used. If this code is ported to a little-endian processor, functions to translate each message to big-endian must be provided. These functions would be used at the component layer, which is knowledgeable of the message types.

6 Dependencies

This document refers to no TSync subsystems; however, it refers to types defined in TSync libraries, components, services and agents to express the contents of each message and improve readability.

7 Data Types and Definitions

TSync is a family of products, and the Public Message API reflects product configurations across the entire family.

TSync supports the following Public Message APIs:

- Host
- Initializer
- Oscillator Monitor
- Reference Monitor
- Supervisor
- Upgrade
- Persistent Data
- Flash Manager
- Log Service
- GPS Reference
- IRIG Reference
- Fixed Frequency Output
- Host Reference
- PPS Reference
- IRIG Output
- PPS Output
- Oscillator
- LED Control
- General Purpose Input
- General Purpose Output
- PTP Reference

The following base message types are defined for explaining the HIDD.

NOTE: These data types and definitions are for example only and do not necessarily reflect the actual implementation.

7.1 General

```
#define HL_AL_DATA_MAX    (1028)  // Largest pkt = 4B dtl + 1kB data
```

The **HL_AL_DATA_MAX** defines the maximum application layer data length transferrable block of data.

```
typedef struct
{
    UINT8  cai;                    // Component Agent Identifier
    UINT8  iid;                    // Item Identifier
    UINT16 ctl;                    // Control Type bit mask
    UINT16 pyldLen;                // Payload Length
```

```
} HL_AL_MSG_HDR;
```

The **HL_AL_MSG_HDR** type is used to define the common message header used by all commands and response messages. A component identifier and item identifier are used to act as a message ID indicating which component item is being acted upon. The control bit mask field provides definition as to the operation being performed. The payload length field provides the length of the message payload.

```
typedef enum
{
    HL_AL_CTL_ERR = (1<<0),         // Error Response (NO_ERR=0, ERR=1)
    HL_AL_CTL_GS  = (1<<1)          // Get or Set (get=0, set=1)

    // remaining bits reserved

} HL_AL_CTL_TYPE;
```

The **HL_AL_CTL_TYPE** type defines the control field bit mask supported for the component messages. This field is used to provide additional configuration and status capability for the messages.

The **HL_AL_CTL_ERR** Error Response bit indicates whether the response returned is the normally expected response message or an error message. The use of this bit allows for success and failure response messages to differ in payload format.

The **HL_AL_CTL_GS** get or set type bit allows the protocol to identify the item's operation state as either get or set.

```
typedef struct
{
    UINT8         pyld[HL_AL_DATA_MAX];   // Optional Data

} HL_AL_MSG_PYLD;
```

The **HL_AL_MSG_PYLD** type is used to define the common message payload used by all commands and responses. The optional payload is operation specific.

```
typedef struct
{
    UINT32 ec;                                      // Error Code
    UINT8  data[HL_AL_DATA_MAX - sizeof(UINT32)]; // Optional Data

} HL_AL_MSG_ERR_PYLD;
```

The **HL_AL_MSG_ERR_PYLD** type is used to define the error response message payload used by all responses messages which indicate an error. The optional data is operation and possibly error specific.

```
typedef union
{
    HL_AL_MSG_PYLD     msg;         // Command/Response message
    HL_AL_MSG_ERR_PYLD errMsg;      // Error message
```

```
} HL_AL_PYLD;
```

The **HL_AL_PYLD** type is a union of all the valid payload types.

```
typedef struct
{
    HL_AL_MSG_HDR hdr;              // Message header
    HL_AL_PYLD    pyld;             // Message payload

} HL_AL_PKT;
```

The **HL_AL_PKT** type us used to define the message packet used by all commands and response messages. It consists of a header and payload.

```
typedef enum
{
    HA_RC_NO_ERROR      = 0x00,
    HA_RC_INVALID_MSG   = 0x01,
    HA_RC_UNKNOWN_ID    = 0x02,
    HA_RC_INVALID_OP    = 0x03,
    HA_RC_INVALID_PARAM = 0x04,
    HA_RC_TIMEOUT       = 0x05,
    HA_RC_CMD_ERR       = 0x06

} HA_RC;
```

The **HA_RC** type enumerates the set of response codes in the HIP.

```
typedef enum
{
    RC_NO_ERROR          = 0,
    RC_NULL              = 1,
    RC_BAD_PARAM         = 2,
    RC_BAD_HANDLE        = 3,
    RC_WRONG_TASK        = 4,
    RC_STUB_FUNC         = 5,
    RC_UNINITIALIZED     = 6,
    RC_READ_ERR          = 7,
    RC_WRITE_ERR         = 8,
    RC_TABLE_FULL        = 9,
    RC_ITEM_NOT_FOUND    = 10,
    RC_ITEM_NOT_SETABLE  = 11,
    RC_ITEM_NOT_GETABLE  = 12,
    RC_MUTEX_ERR         = 13,
    RC_SEMAPHORE_ERR     = 14,
    RC_OPEN_ERR          = 15,
    RC_TABLE_ERR         = 16,
    RC_EVENT_ERR         = 17,
    RC_QUEUE_ERR         = 18,
    RC_TASK_ERR          = 19,
    RC_WAIT_ERR          = 20,
    RC_TIMEOUT           = 21,
    RC_IOCTL_ERR         = 22,
    RC_UNEXPECTED_MSG    = 23,
    RC_IMG_ERR           = 24,
```

```
    RC_TIMER_ERR          = 25,
    RC_DUPLICATE          = 26,
    RC_NOT_ENABLED        = 27,
    RC_BAD_CONFIG         = 28,
    RC_BAD_CHORE          = 29,
    RC_BAD_TSIP_ID        = 30,
    RC_BUF_TOO_SMALL      = 31,
    RC_BAD_LENGTH         = 32,
    RC_BAD_TIME           = 33,
    RC_CONSOLE            = 34,
    RC_PIPE_ERR           = 35,
    RC_PIPE_FULL          = 36,
    RC_PIPE_EMPTY         = 37,
    RC_ANTENNA_ERR        = 38,
    RC_CHG_PENDING        = 39,
    RC_BAD_CHANNEL        = 40,
    RC_TIME_MATCH         = 41,
    RC_PHASE_ERROR        = 42,
    RC_WRONG_MODE         = 43,
    RC_NO_CARD            = 44,
    RC_BAD_SLOT           = 45,
    RC_BAD_CHKSUM         = 46,
    RC_UNSUPPORTED        = 47,
    RC_OSC_PROBLEM        = 48,
    RC_BAD_STATE          = 49,
    RC_BAD_RESPONSE       = 50,
    RC_NO_RESPONSE        = 51,
    RC_MODE_CHANGE        = 52,
    RC_SYNTAX_ERROR       = 53,
    RC_UNKNOWN_MSG        = 54,
    RC_ILLEGAL_OP         = 55,
    RC_BUSY               = 56

} RC;
```

The return codes are described by an enumeration type named **RC** which defines all possible codes that can be returned by functions to the system.

```
typedef struct
{
    FLT64 lat;                      // Latitude in radians
    FLT64 lon;                      // Longitude in radians
    FLT64 alt;                      // Altitude in meters

} POS_LLA;   // See TSYNC_LLAObj
```

The **POS_LLA** type is used to store position information as latitude, longitude, and altitude.

```
typedef enum
{
    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
    SIG_CTL_NUM

} SIG_CTL;
```

The **SIG_CTL** type is used to define the Signature Control Mode.

```
    typedef enum
    {                                   // TFOM based on Estimated Time
                                        // Error (ETE)
        TFOM_MIN       = 0,             // Minimum TFOM value
        TFOM_UNDEFINED = 0,             // TFOM is not defined, ETE is
                                        // unknown
        TFOM_1         = 1,             //              ETE <=   1 nsec
        TFOM_2         = 2,             //    1 nsec < ETE <=  10 nsec
        TFOM_3         = 3,             //   10 nsec < ETE <= 100 nsec
        TFOM_4         = 4,             //  100 nsec < ETE <=   1 usec
        TFOM_5         = 5,             //    1 usec < ETE <=  10 usec
        TFOM_6         = 6,             //   10 usec < ETE <= 100 usec
        TFOM_7         = 7,             //  100 usec < ETE <=   1 msec
        TFOM_8         = 8,             //    1 msec < ETE <=  10 msec
        TFOM_9         = 9,             //   10 msec < ETE <= 100 msec
        TFOM_10        = 10,            //  100 msec < ETE <=   1 sec
        TFOM_11        = 11,            //    1 sec  < ETE <=  10 sec
        TFOM_12        = 12,            //   10 sec  < ETE <=  100 sec
        TFOM_13        = 13,            //  100 sec  < ETE <= 1000 sec
        TFOM_14        = 14,            // 1000 sec  < ETE <= 10000 sec
        TFOM_15        = 15,            //              ETE >  10000 sec
        TFOM_NUM,                       // Number of TFOM values
        TFOM_MAX       = (TFOM_15)

    } 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 time and syntonization of frequency.

```
typedef enum
{
    EDGE_MIN     = 0,
    EDGE_FALLING = 0,
    EDGE_RISING  = 1,
    EDGE_BOTH    = 2,
    EDGE_NUM

} EDGE;
```

The **EDGE** type is used to describe the transitions of a single bit or pin.

```
typedef enum
{
    LEVEL_LOW  = 0,
```

```
    LEVEL_HIGH = 1

} LEVEL;
```

The **LEVEL** type is used to describe the level of a single bit or pin.

```
typedef enum
{
    OSC_UNKNOWN = -1,           // Unknown oscillator type
    OSC_TCXO_1 = 0,             // Lowest quality TCXO
    OSC_TCXO_2 = 1,             // Highest quality TCXO
    OSC_OCXO_1 = 2,             // Lowest quality OCXO
    OSC_OCXO_2 = 3,
    OSC_OCXO_3 = 4,
    OSC_OCXO_4 = 5,             // Highest quality OCXO
    OSC_RBXO_1 = 6,             // Lowest quality Rubidium
    OSC_RBXO_2 = 7,
    OSC_RBXO_3 = 8,             // Highest quality Rubidium
    OSC_NUM

} OSC;
```

The OSC type is used to define the oscillator type used in the timing system.

```
typedef struct
{
    BOOL timeValid;                 // Is the time source valid?
    BOOL ppsValid;                  // Is the 1PPS source valid?

} EL_VALIDITY;
```

The **EL_VALIDITY** type is used to store validity state information for an input reference's time and 1PPS sources.

```
typedef enum
{
    ML_START_TIME_SCALES = 0,       // First time scale in list
    ML_TIME_SCALE_UTC    = 0,       // Universal Coordinated Time
    ML_TIME_SCALE_TAI    = 1,       // International Atomic Time
    ML_TIME_SCALE_GPS    = 2,       // Global Positioning System
    ML_TIME_SCALE_LOCAL  = 3,       // UTC w/local rules for time
                                    // zone/DST
    ML_NUM_TIME_SCALES   = 4,       // Number of time scales
    ML_TIME_SCALE_MAX    = 15       // Maximum number of timescales

} ML_TIME_SCALE;
```

Time scales are ways of keeping time and different time scales typically have different values for the same moment in time. The time scales supported are described by an enumeration type named **ML_TIME_SCALE** which defines each possible time scale for the KTS. The default time scale is UTC.

```
typedef struct
{
    ML_TIME_TYPE  type;
```

```
    ML_TIME_UNION time;

} ML_TIME;
```

The **ML_TIME** structure is used to combine different descriptions of time into a single object that can be passed around easily. The **ML_TIME_UNION** type is defined below.

```
typedef enum
{
    ML_TIME_START_TYPES  = 0,       // First time type in the list
    ML_TIME_DOYTIME      = 0,       // Year, Day of Year, Hour, Min,
                                    // Sec, nsec
    ML_TIME_BCDTIME      = 1,       // BCD Year, Day of Year, Hour,
                                    // Min, Sec, ms, us
    ML_TIME_SECONDS      = 2,       // Total number of seconds in
                                    // epoch, nsec
    ML_TIME_NUM_TYPES               // Number of time types

} ML_TIME_TYPE;
```

Time can be described in more than one way. The supported ways to described it are listed by the enumeration type named **ML_TIME_TYPE**.

```
typedef union
{
    ML_SECTIME st;
    ML_DOYTIME dt;
    ML_BCDTIME bt;

} ML_TIME_UNION;
```

The **ML_TIME_UNION** union is used to combine different descriptions of time into a single object that can be passed around easily.

```
typedef struct
{
    UINT32 seconds;
    UINT32 ns;

} ML_SECTIME;
```

The **ML_SECTIME** structure is used to represent time as a count of seconds and nanoseconds. Not all fields must be used. As a convention, any unused fields should be set to zero.

```
typedef struct
{
    UINT32 year;
    UINT32 doy;
    UINT32 hour;
    UINT32 minute;
    UINT32 second;
    UINT32 ns;
```

```
} ML_DOYTIME;
```

The **ML_DOYTIME** structure is used to represent time as a count of years, days, hours, minutes, seconds, and nanoseconds. Not all fields must be used. As a convention, any unused fields should be set to zero. All 2-digit years are by convention assumed to be after the year 2000. Input references providing 2-digit years should add 2000 to these values.

```
typedef struct
{
    UINT32 year;                    // MSB->LSB defines BCD digit for
                                    // 1000's to 1's of yrs
    UINT32 doy;                     // MSB is unused, then 100's to 1's
                                    // of doy
    UINT32 hour;                    // LSB1 and LSB define 10's and 1's
                                    // of hours
    UINT32 minute;                  // LSB1 and LSB define 10's and 1's
                                    // of minutes
    UINT32 second;                  // LSB1 and LSB define 10's and 1's
                                    // of seconds
    UINT32 ms;                      // MSB is unused, then 100's to 1's
                                    // of milliseconds
    UINT32 us;                      // MSB is unused, then 100's to 1's
                                    // of microseconds

} ML_BCDTIME;
```

The **ML_BCDTIME** type defines the BCD Time down to the microsecond. This structure can be used to store BCD time from conversions of other time types or as read from the Clock subsystem where BCD time to the nearest microsecond is maintained.

```
typedef struct
{
    INT32      offset;
    ML_DOYTIME utcDate;

} ML_LEAP_SEC;
```

The **ML_LEAP_SEC** structure is used to pass around upcoming leap second information, including the number of leap seconds to insert or remove (offset) and the UTC date/time when the leap second offset is applied (utcDate).

```
typedef struct
{
    ML_DST_REF   ref;
    ML_DST_POINT in;
    ML_DST_POINT out;
    INT32        offset;

} ML_DST_RULE;
```

The **ML_DST_RULE** structure is used to describe a specific rule governing the period of the year when daylight savings time is observed. Each rule is described as the reference by which transitions occur (ref), the point in time at which daylight savings

time begins (in), the point in time at which daylight savings time ends (out), and the amount of time that is offset during daylight savings time (offset) in seconds.

```
typedef enum
{
    ML_DST_REF_LCL = 0,             // Reference is local time
    ML_DST_REF_UTC = 1              // Reference is UTC

} ML_DST_REF;
```

The daylight savings time references are described by an enumeration type named **ML_DST_REF** which defines all possible references for daylight savings time transitions.

```
typedef struct
{
    ML_MONTH month;
    ML_WOM   wom;
    ML_DOW   dow;
    UINT32   hour;

} ML_DST_POINT;
```

The **ML_DST_POINT** structure is used to describe a time transition point from standard time to daylight savings time or vice versa. All time transition points are described by the month during which it occurs (month), the week of the month (wom), the day of the week (dow), and the hour of the day (hour).

```
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

} ML_MONTH;
```

The months are described by an enumeration type named **ML_MONTH** which defines all possible months that can be used by the KTS.

```
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

} ML_WOM;
```

The weeks of the month are described by an enumeration type named **ML_WOM** which defines all possible weeks of the month that can be used by the KTS.

```
typedef enum
{
    ML_DOW_SUN    = 0,
    ML_DOW_MON    = 1,
    ML_DOW_TUE    = 2,
    ML_DOW_WED    = 3,
    ML_DOW_THU    = 4,
    ML_DOW_FRI    = 5,
    ML_DOW_SAT    = 6,

} ML_DOW;
```

The days of the week are described by an enumeration type named **ML_DOW** which defines all possible days of the week that can be used by the KTS.

```
typedef struct
{
    ML_DST_RULE dst;
    INT32       tz;

} ML_CLK_LOCAL;
```

The **ML_CLK_LOCAL** structure is used to describe a clock referenced to a localized area. It is described as a daylight savings time rule (dst), and a time zone offset (tz) in seconds.

```
typedef enum
{
    ML_HOUR_START_TYPES  = 0,       // First hour type in the list
    ML_HOUR_12           = 0,       // 12-hour format
    ML_HOUR_24           = 1,       // 24-hour format
    ML_HOUR_NUM_TYPES               // Number of hour types

} ML_HOUR;
```

Hour of day can be described by an enumerated type **ML_HOUR** in 12-hour or 24-hour format.

```
typedef enum
{
    IL_MODE_AUTO   = 0,
    IL_MODE_MANUAL = 1

} IL_MODE;
```

The **IL_MODE** type is used to indicate whether the KTS should attempt to automatically detect the format, frequency, and coded expressions of the incoming IRIG signal (AUTO) or if it should rely on manual settings (MANUAL).

```
typedef enum
{
    IL_FMT_IRIG_A     = 0,
    IL_FMT_IRIG_B     = 1,
    IL_FMT_IRIG_G     = 2,
    IL_FMT_NASA_36    = 3,
    IL_FMT_IRIG_E_100 = 4,
    IL_FMT_IRIG_E_1K  = 5,
    IL_FMT_UNKNOWN    = 7,          // Unknown
    IL_FMT_AUTO                     // Auto-detection

} IL_FMT;
```

The **IL_FMT** type is used to indicate the format of an IRIG time code stream.

```
typedef enum
{
    IL_MOD_DCLS         = 0,        // IRIG DCLS only
    IL_MOD_AM           = 1,        // IRIG AM only
    IL_MOD_MAN          = 2,        // IRIG Manchester coding
    IL_MOD_UNKNOWN      = 3,        // Unknown
    IL_MOD_AM_AND_DCLS = 4          // Port supports both AM and DCLS
    IL_MOD_AM_OR_DCLS  = 5          // Port generates AM or DCLS


} IL_MOD;
```

The **IL_MOD** type is used to indicate the modulation of an IRIG time code stream. .

```
typedef enum
{
    IL_FRQ_NONE    = 0,             // No Carrier/Index count interval
    IL_FRQ_100HZ   = 1,
    IL_FRQ_1KHZ    = 2,
    IL_FRQ_10KHZ   = 3,
    IL_FRQ_100KHZ  = 4,
    IL_FRQ_1MHZ    = 5,
    IL_FRQ_UNKNOWN = 6              // Unknown

} IL_FRQ;
```

The **IL_FRQ** type is used to indicate the carrier frequency of an IRIG time code stream.

```
typedef enum
{
    IL_CE_BCDT_CF_SBS      = 0,     // BCD TOY, Ctrl Func, Binary
                                    // Seconds
    IL_CE_BCDT_CF          = 1,     // BCD TOY, Ctrl Func
    IL_CE_BCDT             = 2,     // BCD TOY
    IL_CE_BCDT_SBS         = 3,     // BCD TOY, Binary Seconds,
    IL_CE_BCDT_BCDY_CF_SBS = 4,     // BCD TOY/Year, Ctrl Func, Binary
```

```
                                    // Seconds
    IL_CE_BCDT_BCDY_CF     = 5,     // BCD TOY/Year, Ctrl Func
    IL_CE_BCDT_BCDY        = 6,     // BCD TOY/Year,
    IL_CE_BCDT_BCDY_SBS    = 7,     // BCD TOY/Year, Binary Seconds
    IL_CE_UNKNOWN          = 8      // Unknown - No fields

} IL_CE;
```

The **IL_CE** type is used to indicate the coded expressions combination of an IRIG time code stream. Formats that place BCD year information overlapping with the beginning of the control function field should use **IL_CE_BCDT_CF** or **IL_CE_BCDT_CF_SBS** in conjunction with **IL_CF_1344**.

```
typedef enum
{
    IL_CF_MIN      = 0,             // Minimum value of CF
    IL_CF_UNKNOWN  = 0,             // Unknown - All bits ignored
    IL_CF_200_04   = 1,             // Fields conform to RCC 200-04
    IL_CF_1344     = 2,             // Fields conform to IEEE
                                    // C37.118-2005
    IL_CF_SPEC     = 3,             // Fields conform to Spectracom
                                    // format
    IL_CF_SPEC_FAA = 4,             // Fields conform to Spectracom FAA
                                    // format
    IL_CF_NASA     = 5,             // Fields conform to NASA formats
    IL_CF_NUM                       // Number of CF types

} IL_CF;
```

The **IL_CF** type is used to indicate the format of the control field bits in the IRIG stream.

```
typedef enum
{
    UD_BR      br;
    UD_STOP    numbits;
    UD_DATA    stopbits;
    UD_PAR     parity;

} UD_CFG;
```

The **UD_CFG** type defines the configuration used by the UD for a UART.

```
typedef enum
{
    UD_BR_1200   = 1200,
    UD_BR_2400   = 2400,
    UD_BR_4800   = 4800,
    UD_BR_9600   = 9600,
    UD_BR_19200  = 19200,
    UD_BR_38400  = 38400,
    UD_BR_57600  = 57600,
    UD_BR_76800  = 76800,
    UD_BR_115200 = 115200
```

```
} UD_BR;
```

The **UD_BR** type defines the bit rates supported by the UD.

```
typedef enum
{
    UD_DATA_8 = 0,                  // 8 Data bits
    UD_DATA_7 = 1,                  // 7 Data bits
    UD_DATA_6 = 2,                  // 6 Data bits
    UD_DATA_5 = 3                   // 5 Data bits

} UD_DATA;
```

The **UD_DATA** type defines the number of data bits supported by the UD.

```
typedef enum
{
    UD_STOP_1  = 0,                 // 1 Stop bit
    UD_STOP_15 = 1,                 // 1.5 Stop bits
    UD_STOP_2  = 2                  // 2 Stop bits

} UD_STOP;
```

The **UD_STOP** type defines the number of stop bits supported by the UD.

```
typedef enum
{
    UD_PAR_NONE = 0,                // Parity none
    UD_PAR_ODD  = 1,                // Odd parity
    UD_PAR_EVEN = 2                 // Even parity

} UD_PAR;
```

The **UD_PAR** type defines the types of parity supported by the UD.

```
typedef enum
{
    QL_FMT_HQ_I     = 0,            // STANAG 4246 HaveQuick I
    QL_FMT_HQ_II    = 1,            // STANAG 4246 HaveQuick II
    QL_FMT_HQ_IIA   = 2,            // STANAG 4372 HaveQuick IIA
    QL_FMT_4430_STM = 3,            // STANAG 4430 Standard Time Message
    QL_FMT_4430_XHQ = 4,            // STANAG 4430 Extended HaveQuick
    QL_FMT_GPS_BCD  = 5,            // ICD-GPS-060A Binary Coded Decimal
    QL_FMT_GPS_HQ   = 6,            // ICD-GPS-060A HaveQuick
    QL_FMT_NUM,                     // Number of formats
    QL_FMT_START    = 0,            // Start of format list
    QL_FMT_UNKNOWN  = -1            // Unknown format

} QL_FMT;
```

The **QL_FMT** type defines the HaveQuick formats.

```
typedef enum
{
    AL_FMT_SPEC_0     = 0x00000000,     // Spectracom format 0
```

```
    AL_FMT_SPEC_1     = 0x00000001,     // Spectracom format 1
    AL_FMT_SPEC_2     = 0x00000002,     // Spectracom format 2
    AL_FMT_SPEC_3     = 0x00000003,     // Spectracom format 3
    AL_FMT_SPEC_4     = 0x00000004,     // Spectracom format 4
    AL_FMT_SPEC_5     = 0x00000005,     // Spectracom format 4
    AL_FMT_SPEC_6     = 0x00000006,     // Spectracom format 6
    AL_FMT_SPEC_7     = 0x00000007,     // Spectracom format 7
    AL_FMT_SPEC_8     = 0x00000008,     // Spectracom format 8
    AL_FMT_SPEC_9     = 0x00000009,     // Spectracom format 9
    AL_FMT_NMEA_GGA   = 0x0000000A,     // NMEA GGA message
    AL_FMT_NMEA_RMC   = 0x0000000B,     // NMEA RMC message
    AL_FMT_NMEA_ZDA   = 0x0000000C,     // NMEA ZDA message
    AL_FMT_BBC_01     = 0x0000000D,     // BBC format 1
    AL_FMT_BBC_02     = 0x0000000E,     // BBC format 2
    AL_FMT_BBC_03     = 0x0000000F,     // BBC format 3 PSTN
    AL_FMT_BBC_04     = 0x00000010,     // BBC format 4
    AL_FMT_153C_BB    = 0x00000011,     // ICD-153C Buffer Box
    AL_FMT_153C_TT    = 0x00000012,     // ICD-153C Time Transfer
    AL_FMT_SPEC_1S    = 0x00000101,     // Format 1 variant
    AFL_FMT_UNKNOWN   = 0x0000FFFF,     // Unknown (auto)

} AL_FMT;
```

The **AL_FMT** type defines the individual ASCII message formats supported by the AP and AR.

```
typedef union
{
    NL_PKT nmeaPkt;          // NMEA Formats Common Packet Struct
    FL_PKT specPkt;          // Spectracom Formats Common Packet Struct

} AL_PKT
```

The **AL_PKT** union is used to generically contain any message format common packet structure.

```
typedef enum
{
    AL_OMODE_BC,              // Broadcast
    AL_OMODE_OT,              // On-Time
    AL_OMODE_IM,              // Immediate
    AL_OMODE_ST               // Streaming (unsupported)

} AL_OUT_MODE
```

The **AL_OUT_MODE** type defines the message transmit behavior supported by the AL.

```
typedef enum
{
    LE_ALL      = -1,

    LE_0        = 0,
    LE_1        = 1,
    LE_2        = 2,
    LE_3        = 3,
```

```
    LE_4        = 4,
    LE_5        = 5,

    LE_LEDS_NUM = 6

} LE_INDEX;
```

The **LE_INDEX** type defines the index used to describe each LED.

7.2 Host Agent

```
typedef struct
{
    UINT32 pgno;
    BOOL   more;
    CI_CAP caps[CI_CAP_PAGE_SIZE];

} CI_CAP_PAGE;
```

The **CI_CAP_PAGE** type defines a page or set of capabilities. This is used for sending fixed-size sets of capability information to an Agent. Each page consists of the set of capabilities (caps), a page number identifier (pgno), and an indication of whether there are additional pages available after the current page (more).

The **CI_CAP_PAGE_SIZE** type defines the number of item capabilities per capabilities page.

```
#define CI_CAP_PAGE_SIZE                (16)

typedef struct
{
    CI_ITEM   item;
    CI_ACCESS access;

} CI_CAP;
```

The **CI_CAP** type defines a specific item that the CI is capable of accessing. In addition to the item identifier (item), it describes the type of access supported (access).

```
typedef UINT32 CI_ITEM;
```

The **CI_ITEM** type is used to identify pieces of data that can be get/set from Components via the CI. Each component is statically assigned a range of values within which it defines all data items exposed as its public API.

```
typedef enum
{
    CI_ACC_NONE = 0,
    CI_ACC_GET  = 1,
    CI_ACC_SET  = 2,
    CI_ACC_BOTH = 3
```

```
} CI_ACCESS;
```

The **CI_ACCESS** type enumerates the levels of access to a particular piece of data.

7.3 Initializer

```
typedef struct
{
    UINT32    pgno;
    BOOL      more;
    IN_RESULT results[TSYNC_INIT_RESULT_NUM];

} IN_STATUS;
```

The **IN_STATUS** type is used to pass around an entire table of initialization results. Each page consists of the page number (pgno, set of results (results), and an indication of whether there are additional pages available after the current page (more).

```
typedef struct
{
    char module[32];                // Module identification string
    RC   result;                    // Return code from initialization

} IN_RESULT;
```

The **IN_RESULT** type is used to store the initialization result for one software module.

7.4 Oscillator Monitor

```
typedef enum
{
    XS_CMD_AUTO    = 0,             // Automatically start a new window
                                    // of measurement when a window
                                    // completes
    XS_CMD_START   = 1,             // Begin a single window of
                                    // measurement
    XS_CMD_STOP    = 2,             // Stop measuring immediately
    XS_CMD_FINISH  = 3,             // Stop after current window
                                    // completes
    XS_CMD_RESTART = 4              // Restart window measurement (Does
                                    // not change meter state)

} XS_CMD;
```

The oscillator meter commands are described by an enumeration type named **XS_CMD**.

```
typedef struct
{
    XS_STATE state;                 // Meter state
    UINT32   size;                  // Window size
    UINT32   elapsed;               // Elapsed duration through the
                                    // window
```

```
    INT32    frAccum;              // Accumulated frequency error of
                                   // current window
    INT32    frPrev;               // Total frequency error of
                                   // previous window
    INT32    phStart;              // Starting phase error of current
                                   // window
    INT32    phAccum;              // Accumulated phase error of
                                   // current window
    INT32    phPrev;               // Total phase error of previous
                                   // window

} XS_METER_DATA;
```

The **XS_METER_DATA** type defines the meter's configuration settings.

```
typedef enum
{
    XS_STATE_STOPPED = 0,          // Not measuring
    XS_STATE_RUNNING = 1,          // Measuring
    XS_STATE_ENDING  = 2           // Measuring until end of window

} XS_STATE;
```

The oscillator meter states are described by an enumeration type named **XS_STATE**.

7.5 Shared Memory

```
typedef enum
{
    MS_DI_NMEA_GGA  = 0,           // NMEA messages
    MS_DI_NMEA_GLL  = 1,
    MS_DI_NMEA_GSA  = 2,
    MS_DI_NMEA_GSV  = 3,
    MS_DI_NMEA_RMC  = 4,
    MS_DI_NMEA_VTG  = 5,
    MS_DI_NMEA_ZDA  = 6,
    MS_DI_POSITION  = 7,           // Position of unit
    MS_DI_NUM_ITEMS = 8,           // Total number of data items
    MS_DI_ALL = -1                 // Refers to all data items

} MS_DI_INDEX;
```

The **MS_DI_INDEX** type defines the index used to access each shared memory data item in the MS.

```
typedef struct
{
    UINT32  seqNum;
    MS_DATA data;

} MS_SHARE_ITEM;
```

The **MS_SHARE_ITEM** structure type is used to hold a shared data and a sequence number. The sequence number is incremented if the shared memory item is changed.

```
typedef union
{
    char    nmea[256];
    POS_LLA pos;

} MS_DATA;
```

The **MS_DATA** type is used to abstract the type of a piece of data as a union of all types that are managed as shared memory data items.

7.6 Reference Monitor

```
typedef struct
{
    BOOL   enab;                    // TRUE=1, FALSE=0
    UINT32 prio;                    // Lower number = higher priority
                                    // (1 - MAX)
    char   time [4];                // Input time source (4-character)
    char   pps  [4];                // Input 1PPS source (4-character)

} RS_TABLE_ENTRY;
```

The RS uses a **RS_TABLE_ENTRY** array as the fixed size reference table. The reference table is used to search the input references components or input reference time and PPS reference pairs to determine the current best available highest priority input reference source to synchronize the CS.

```
typedef enum
{
    RS_TT_START_TYPES = 0,
    RS_TT_FACT        = 0,          // Factory default table (built at
                                    // runtime)
    RS_TT_USER        = 1,          // User-default table (stored in
                                    // EEPROM)
    RS_TT_CURRENT     = 2,          // Current working table

    RS_TT_NUM_TYPES

} RS_TABLE_TYPE;
```

The RS uses a **RS_TABLE_TYPE** to define which type of table.

```
typedef struct
{
    char src [4];                   // Input source (4-character)
    BOOL timeValid;                 // Time source validity
    BOOL ppsValid;                  // 1PPS source validity

} RS_REF_STATE_ENTRY;
```

The **RS_REF_STATE_ENTRY** structure holds a single entry in the reference state tables. Each entry contains a 4-character ASCII source identifier and its current validity.

7.7 Supervisor

```
typedef enum
{
    SS_EVT_SYNC = 0,                // In sync / Out of sync
    SS_EVT_REF  = 1                 // Valid ref(s) / No valid ref(s)

} SS_EVENT;
```

The events that the SS can execute callback functions for are described by an enumerated type named SS_EVENT.

```
typedef enum
{
    SS_RESET_BRD   = 0,             // Reset the entire board except
                                    // for FPGA
    SS_RESET_MIC   = 1,             // Reset the microprocessor &
                                    // peripherals
    SS_RESET_FPGA  = 2,             // Reset the FPGA only
    SS_RESET_SLOTS = 3,             // Reset the FPGA in the slots only
    SS_RESET_NUM

} SS_RESET;
```

The resets that the SS can perform are described by an enumerated type **SS_RESET**. Typically only the board level reset will be performed at the product level by the host and other resets are reserved for development, debug and use under software control.

```
typedef enum
{
    SS_TS_MIN      = 0,         // Minimum Timestamp index
    SS_TS_TIME_REF = 0,         // Timestamp on Time Reference change
    SS_TS_1PPS_REF = 1,         // Timestamp on 1PPS Reference change
    SS_TS_TFOM     = 2,         // Timestamp on TFOM value change
    SS_TS_SYNC     = 3,         // Timestamp on Sync state change
    SS_TS_HOLDOVER = 4,         // Timestamp on Holdover state change
    SS_TS_LOST_REF = 5,         // Timestamp on entering Holdover
    SS_TS_NUM                   // Number of Timestamps
} SS_TS_SRC;
```

The timestamp types supported by the Supervisor are defined by this type.

```
typedef struct
{
    char time[4];
    char pps[4];

} SS_REF;
```

The reference structure defines 2 ASCII arrays for time and 1PPS reference names

```
typedef struct
{
    SS_TS_SRC    src;
    ML_TIME_TYPE type;

} SS_TS_REQ;
```

This SS_TS_REQ type is used to request a specified timestamp in a particular format.

7.8 Upgrade

```
typedef struct
{
    FS_IMG     image;
    US_PROCESS step;
    UINT32     complete;

} US_STATE;
```

The **US_STATE** structure is used to report the current state of the US. It consists of the image being upgraded (image), the current step in the upgrade process (step), and a percentage estimate of how complete the upgrade is (complete).

```
typedef enum
{
    US_PROC_IDLE      = 0,
    US_PROC_DOWNLOAD  = 1,
    US_PROC_CRC_RAM   = 2,
    US_PROC_ERASE     = 3,
    US_PROC_PROGRAM   = 4,
    US_PROC_CRC_FLASH = 5,
    US_PROC_FAILED    = 6

} US_PROCESS;
```

The **US_PROCESS** enumerated type is used to define the supported process steps of an image upgrade.

7.9 Flash Manager

```
typedef enum
{
    FS_IMG_RT_FW        = 0,   // Run-time firmware image (upgradeable)
    FS_IMG_RT_FPGA      = 1,   // Run-time FPGA image (upgradeable)
    FS_IMG_DEF_FW       = 2,   // Default firmware image (read-only)
    FS_IMG_DEF_FPGA     = 3,   // Default FPGA image (read-only)
    FS_IMG_BL           = 4,   // Boot Loader (read-only)
    FS_IMG_CFPGA        = 5,   // Compressed FPGA image (read-only)
    FS_IMG_EE_PATCH     = 6,   // EEPROM Patch image
    FS_IMG_SER_RT_FPGA  = 7,   // Run-time serial flash FPGA image
                               // (upgradeable)
```

```
    FS_IMG_SER_DEF_FPGA = 8,    // Default serial flash FPGA image
                                // (upgradeable)

    FS_IMG_NUM_ITEMS,
    FS_IMG_RSVD_BL = 0xFF       // Reserved for bootloader use

} FS_IMG;
```

The code images that reside in internal and external flash are described by an enumeration type named **FS_IMG** which defines all possible images managed by the FS.

```
typedef struct
{
    UINT32 mark;                        // Image marker
    FS_IMG type;                        // Image type/location
    UINT32 len;                         // Image length

} FS_IMG_HDR;
```

The **FS_IMG_HDR** structure defines a flash image header containing a marker, the image type, and image length.

```
typedef struct
{
    char   ver[4];                      // Image version
    UINT32 crc;                         // Image crc

} FS_IMG_TRL;
```

The **FS_IMG_TRL** structure defines a flash image trailer containing a version and CRC.

7.10 GPS Reference

```
typedef struct
{
    UINT32 chnum;                   // Channel number
    UINT32 svId;                    // ID number of the SV being
                                    // tracked
    UINT32 str;                     // Signal strength
    BOOL   traim;                   // Is accepted by TRAIM algorithm?
    BOOL   infix;                   // Is used for position or time
                                    // fixes?
    UINT32 flags;                   // Bit flags

} GL_SAT_INFO;
```

The **GL_SAT_INFO** type is used to store information about a specific GPS satellite.

```
typedef enum
{
    GL_DYN_LAND = 0,
    GL_DYN_SEA  = 1,
    GL_DYN_AIR  = 2,
```

```
    GL_DYN_STAT = 3

} GL_DYN;
```

The **GL_DYN** type defines the dynamics mode for the algorithms used in the GPS receiver used to model motion.

```
typedef enum
{
    GL_MODE_1SAT = 0,
    GL_MODE_STND = 1,
    GL_MODE_CONT = 2,
    GL_MODE_AVRG = 3
    GL_MODE_TIME = 4,
    GL_MODE_STBY = 5,
    GL_MODE_SELF = 6

} GL_MODE;
```

The **GL_MODE** type defines the operating mode of the GPS receiver.

```
typedef struct
{
    UINT32 nSats;                   // Number of Satellites used in fix
    FLT32  pdop;                    // Position DOP
    FLT32  hdop;                    // Horizontal DOP
    FLT32  vdop;                    // Vertical DOP
    FLT32  tdop;                    // Time DOP
    UINT32 fom;                     // FOM
    UINT32 tfom;                    // Time FOM
    UINT32 herr;                    // Horizontal Error
    UINT32 verr;                    // Vertical Error

} GL_FIX_INFO;
```

The **GL_FIX_INFO** is used to return GPS fix information regarding the position and time accuracy.

```
typedef struct
{
    UINT32      chNum;              // Channel number
    UINT32      svId;               // ID number of SV being tracked
    UINT32      str;                // Signal strength
    BOOL        traim;              // Is accepted by T-RAIM alg?
    BOOL        infix;              // Is used for pos or time fix?
    UINT32      flags;              // Bit flags

} GL_SAT_INFO;
```

The **GL_SAT_INFO** structure is used to contain space vehicle data for one channel of a GPS receiver.

```
typedef struct
{
    GL_SAT_INFO info[32];           // Satellite information structures
```

```
} GR_SAT_DATA;
```

The **GR_SAT_DATA** structure is used to contain space vehicle data for all channels of a GPS receiver.

```
typedef struct
{
    UINT32 length;
    char   msg[256];

} GR_CUSTOM_MSG;
```

The **GR_CUSTOM_MSG** structure is used to contain a custom GPS protocol message. A custom message is a message sent to or received from the GPS receiver in its native communications protocol.

```
typedef struct
{
    char mfr[16];
    char mdl[16];

} GR_MFR_MDL;
```

The **GR_MFR_MDL** structure is used to contain the manufacturer and model strings for the GPS receiver.

```
typedef enum
{
    GL_RESET_COLD = 0,  // Clear data in RAM (like power-cycle)
    GL_RESET_WARM = 1,  // Clear ephemeris and osc. uncertainty
    GL_RESET_HOT  = 2,  // No clear, SW reset, rerun self-test
    GL_RESET_POS  = 3,  // Clear position in receiver flash
    GL_RESET_FACT = 4,  // Cold reset + restore factory settings
    GL_RESET_SAVE = 5,  // Reset GPS + save settings

    // Reserve all other reset numbers…

    GL_RESET_NONE = 0xFF // No Reset required for Web UI Display

} GL_RESET;
```

The **GL_RESET** type is used to define the various GPS Resets supported.

```
typedef enum
{
    GL_ANT_OK    = 0,
    GL_ANT_SHORT = 1,
    GL_ANT_OPEN  = 2,
    GL_ANT_UNK   = 3

} GL_ANT_STATUS;
```

The **GL_ANT_STATUS** type is used to define the various GPS Antenna status values supported.

7.11 IRIG Reference

```
typedef struct
{
    UINT16 subP0;                   // Sub-frame P0 from current second
    UINT16 subP1;                   // Sub-frame P1 from current second
    UINT16 subP2;                   // Sub-frame P2 from current second
    UINT16 subP3;                   // Sub-frame P3 from current second
    UINT16 subP4;                   // Sub-frame P4 from current second
    UINT16 subP5;                   // Sub-frame P5 from current second
    UINT16 subP6;                   // Sub-frame P6 from previous second
    UINT16 subP7;                   // Sub-frame P7 from previous second
    UINT16 subP8;                   // Sub-frame P8 from previous second
    UINT16 subP9;                   // Sub-frame P9 from previous second

} IR_MSG;
```

The **IR_MSG** structure is used to contain an IRIG message received from the IRIG hardware decoder.

7.12 ASCII Output

```
typedef enum
{
    AL_OMODE_BC = 0,                    // Broadcast
    AL_OMODE_OT = 1,                    // On-Time
    AL_OMODE_IM = 2,                    // Immediate
    AL_OMODE_ST = 3                     // Streaming (unsupported)

} AL_OUT_MODE;
```

The **AL_OUT_MODE** type defines the ASCII output mode determining how an output message is transmitted.

7.13 ASCII Reference

```
typedef enum
{
    AR_1PPS_TYPE_DERIVED,   // 1PPS is derived from OTP in time code
    AR_1PPS_TYPE_DISCRETE   // 1PPS is provided on separate input pin

} AR_1PPS_TYPE;
```

The **AR_1PPS_TYPE** type is describe the cases for a 1PPS reference associated with the ASCII time code.

7.14 IRIG Output

```
typedef struct
{
    UINT16 subP0;                   // Sub-frame P0
```

```
    UINT16 subP1;                   // Sub-frame P1
    UINT16 subP2;                   // Sub-frame P2
    UINT16 subP3;                   // Sub-frame P3
    UINT16 subP4;                   // Sub-frame P4
    UINT16 subP5;                   // Sub-frame P5
    UINT16 subP6;                   // Sub-frame P6
    UINT16 subP7;                   // Sub-frame P7
    UINT16 subP8;                   // Sub-frame P8
    UINT16 subP9;                   // Sub-frame P9

} IP_MSG;
```

The **IP_MSG** structure is used to contain an IRIG message to be sent to the IRIG hardware decoder.

7.15 Variable Frequency

```
typedef struct
{
    UINT32 min;                     // Minimum frequency
    UINT32 max;                     // Maximum frequency
    FLT32  step;                    // Minimum frequency step size

} VP_CFG;
```

The **VP_CFG** type is used to define the frequency range and minimum step size.

7.16 Display Output

```
typedef enum
{
    DP_MODE_NORMAL = 0,             // Normal operational mode
    DP_MODE_PHOTO  = 1,             // Photo mode
    DP_MODE_TEST   = 2,             // Display test mode

} DP_MODE;
```

The **DP_MODE** type is used to define the Display Output's operational mode.

7.17 Oscillator

```
typedef enum
{
    XO_MODE_DISC,                   // Disciplining
    XO_MODE_TEST                    // Testing

} XO_MODE;
```

The **XO_MODE** type defines the oscillator modes that may be used in the XO.

```
typedef struct
{
    char mfr[16];
```

```
    char mdl[16];

} XO_MFR_MDL;
```

The **XO_MFR_MDL** structure is used to contain the manufacturer and model strings for the oscillator.

```
typedef struct
{
    UINT32 length;
    char   msg[256];

} XO_CUSTOM_MSG
```

The **XO_CUSTOM_MSG** structure is used to contain a custom external oscillator protocol message. A custom message is a message sent to or received from the external oscillator in its native communications protocol.

7.18 LED Control

```
typedef enum
{
    EC_MODE_SYNC     = 0,          // LED indicates sync status
    EC_MODE_HOLDOVER = 1,          // LED indicates holdover status
    EC_MODE_ALARM    = 2,          // LED indicates alarm status
    EC_MODE_1PPS     = 3,          // LED goes on briefly when 1PPS
                                   // occurs
    EC_MODE_MANUAL   = 4,          // LED manually controlled

    EC_MODE_NUM

} EC_MODE;
```

The **EC_MODE** type lists the different modes of operation for an LED.

```
typedef enum
{
    EC_STATE_OFF   = 0,            // LED off solid
    EC_STATE_ON    = 1,            // LED on solid
    EC_STATE_BLINK = 2,            // LED blinks on/off (2Hz rate)
    EC_STATE_CODE  = 3,            // LED blinks a code (2Hz rate,
                                   // 2sec pause)

    EC_STATE_NUM

} EC_STATE;
```

The **EC_STATE** type lists the allowable action states for an LED.

7.19 General Purpose Input

```
typedef enum
{
```

```
    ID_PIN_ALL = -1,

    ID_PIN_MIN = 0,
    ID_PIN_0   = 0,
    ID_PIN_1   = 1,
    ID_PIN_2   = 2,
    ID_PIN_3   = 3,
    ID_PIN_4   = 4,
    ID_PIN_5   = 5,
    ID_PIN_6   = 6,
    ID_PIN_7   = 7,

    // Add more GPIO Inputs here

    ID_PIN_NUM = 4 // [FUTURE]      // Number of GPI pins

} ID_PIN;
```

The **ID_PIN** type defines the index used to describe each GPI pin.

7.20 General Purpose Output

```
typedef enum
{
    OD_PIN_ALL = -1,
    OD_PIN_MIN = 0,
    OD_PIN_0   = 0,
    OD_PIN_1   = 1,
    OD_PIN_2   = 2,
    OD_PIN_3   = 3,
    OD_PIN_4   = 4,
    OD_PIN_5   = 5,
    OD_PIN_6   = 6,
    OD_PIN_7   = 7,
    OD_PIN_NUM = 4                  // Number of GPO ([FUTURE] 8)

} OD_PIN;
```

The **OD_PIN** type defines the index used to describe each General Purpose Output (GPO).

```
typedef enum
{
    OD_MODE_DIRECT      = 0,        // Direct Output Value
    OD_MODE_MATCH_TIME  = 1,        // Match Time Output
    OD_MODE_SQUARE_WAVE = 2,        // Square Wave
    OD_MODE_RESERVED    = 3

} OD_MODE;
```

The **OD_MODE** type defines the Output Mode used by the OD.

```
typedef struct
{
```

```
    OD_PIN pin;                         // GP Output Selection
    LEVEL  lvl;                         // Low or High Match Time

} GO_MATCH_SEL;
```

The **GO_MATCH_SEL** type defines the selection of the GP output and specifies whether selecting configuration for a low or high match time.

```
typedef UINT32 GO_PER;
```

The **GO_PER** type defines the period of the square wave output. The MSB is used to select the scale of the rest of the period value.
0 = Period value is in ns.
1 = Period value is in us.

```
typedef struct
{
    INT32     off;                      // Offset from internal 1PPS
    GO_PER    per;                      // Period of square wave
    UINT32    pw;                       // Incident Pulse Width
    EDGE      ae;                       // Active Edge

} GO_SQUARE;
```

The **GO_SQUARE** type defines the Square Wave output configuration supported by the GO.

7.21 Log Service

```
typedef enum
{
    LS_ALARM_SYNC     = 0,      // Not in Sync
    LS_ALARM_HOLDOVER = 1,      // In Holdover
    LS_ALARM_FREQ_ERR = 2,      // Frequency Error
    LS_ALARM_SELF_REF = 3,      // Self Reference only
    LS_ALARM_SW_ERR   = 4,      // Software Error
    LS_ALARM_1PPS     = 5,      // 1PPS is not in specification
    LS_ALARM_REF_CHG  = 6,      // Reference Change
    LS_ALARM_HW_ERR   = 7,      // Hardware Error

    LS_ALARM_NUM                // Number of Alarms

} LS_ALARM;
```

The alarms the KTS can raise to the outside world are defined by this type. All alarms are gettable. The only settable alarm is the software error which can be cleared by being set to FALSE.

7.22 E1/T1 Output

```
typedef enum
{
```

```
    ETP_MODE_E1 = 0,
    ETP_MODE_T1 = 1

} ETP_MODE;
```

The **ETP_MODE** defines the mode of output used by the E1/T1 Output card.

```
typedef enum
{
    ETP_FMT_SF      = 0,            // Super Frame
    ETP_FMT_ESF     = 1,            // Extended Super Frame
    ETP_FMT_ESF_SSM = 2             // Extended Super Fram with SSM

} ETP_FMT;
```

The **ETP_FMT** is the format of the frame output by the E1/T1 Output card.

7.23 PTP Reference

```
typedef struct
{
    UINT32 ptpVersionNumber
    UINT32 softwareVersion; // 8 bits
    char   hardwareVersion;
    char   filler[3];       // for byte alignment
    char   softDate[12];
    char   softTime[9];

} PTL_MODULE_INFO;
```

The **PTL_MODULE_INFO** structure is used to store information about the PTP module attached to the system.

```
typedef struct
{
    BOOL            dhcpEnabled;
    UINT8           macAddress[8]; // only [6:0] are used
    UINT8           staticIpAddr;
    UINT8           ipAddress[4];
    UINT8           netMask[4];
    UINT8           defaultGateway[4];

} PTL_ETH_INFO;
```

The **PTL_ETH_INFO** structure is used to store information about the PTP module's Ethernet settings.

```
typedef struct
{
    BOOL            ptpClockRunning;
    BOOL            usingExternalClock;

} PTL_CLK_SETTINGS;
```

The **PTL_CLK_SETTINGS** structure is used to store information about the PTP module's clock.

```
typedef struct
{
    UINT8           clockIdentity[8];
    BOOL            oneStepMode;
    BOOL            unicast;
    UINT32          domainNumber; // 8 bits
    UINT32          priority1; // 8 bits
    UINT32          priority2; // 8 bits

} PTL_UNIT_SETTINGS;
```

The **PTL_UNIT_SETTINGS** structure is used to store general settings for the PTP module.

```
typedef enum
{
    PTL_PTP_STATE_INVALID      = -1
    PTL_PTP_STATE_INITIALIZING = 0,
    PTL_PTP_STATE_FAULTY       = 1,
    PTL_PTP_STATE_DISABLED     = 2,
    PTL_PTP_STATE_LISTENING    = 3,
    PTL_PTP_STATE_PRE_MASTER   = 4,
    PTL_PTP_STATE_MASTER       = 5,
    PTL_PTP_STATE_PASSIVE      = 6,
    PTL_PTP_STATE_UNCALIBRATED = 7,
    PTL_PTP_STATE_SLAVE        = 8,

    PTL_PTP_STATE_COUNT        = 9

} PTL_PTP_STATE;
```

The **PTL_PTP_STATE** enumeration specifies all possible states for a particular PTP port.

```
typedef struct
{
    UINT32          portNumber; // 8 bits
    BOOL            portEnabled;
    PTL_PTP_STATE   portState;
    BOOL            linkConnected;

} PTL_PORT_STATE;
```

The **PTL_PORT_STATE** structure is used to store port information for the PTP module.

```
typedef enum
{
    PTL_DELAY_MECH_E2E      = 0x01,
    PTL_DELAY_MECH_P2P      = 0x02,
    PTL_DELAY_MECH_DISABLED = 0xFE

} PTL_DELAY_MECH;
```

The **PTL_DELAY_MECH** enumeration specifies all available delay mechanisms as specified in the PTP spec.

```
typedef struct
{
    UINT32          portNumber;              // 8 bits
    UINT32          annRcptTimeout;          // 8 bits
    INT32           logAnnInterval;          // 8 bits signed
    INT32           logSyncInterval;         // 8 bits signed
    INT32           logDelayReqInterval;     // 8 bits signed
    INT32           logPeerDelayReqInterval; // 8 bits signed
    PTL_DELAY_MECH  delayMechanism;

} PTL_PORT_SETTINGS;
```

The **PTL_PORT_SETTINGS** structure is used to store settings for a particular PTP port.

```
typedef enum
{
    PTL_CLK_ACC_MIN            = 0x20,

    PTL_CLK_ACC_WITHIN_25_NS   = 0x20,
    PTL_CLK_ACC_WITHIN_100_NS  = 0x21,
    PTL_CLK_ACC_WITHIN_250_NS  = 0x22,
    PTL_CLK_ACC_WITHIN_1000_NS = 0x23,
    PTL_CLK_ACC_WITHIN_2_5_US  = 0x24,
    PTL_CLK_ACC_WITHIN_10_US   = 0x25,
    PTL_CLK_ACC_WITHIN_25_US   = 0x26,
    PTL_CLK_ACC_WITHIN_100_US  = 0x27,
    PTL_CLK_ACC_WITHIN_250_US  = 0x28,
    PTL_CLK_ACC_WITHIN_1_MS    = 0x29,
    PTL_CLK_ACC_WITHIN_2_5_MS  = 0x2A,
    PTL_CLK_ACC_WITHIN_10_MS   = 0x2B,
    PTL_CLK_ACC_WITHIN_25_MS   = 0x2C,
    PTL_CLK_ACC_WITHIN_100_MS  = 0x2D,
    PTL_CLK_ACC_WITHIN_250_MS  = 0x2E,
    PTL_CLK_ACC_WITHIN_1_S     = 0x2F,
    PTL_CLK_ACC_WITHIN_10_S    = 0x30,
    PTL_CLK_ACC_BEYOND_10_S    = 0x31,

    PTL_CLK_ACC_MAX            = 0x31,

    PTL_CLK_ACC_UNKNOWN        = 0xFE

} PTL_CLK_ACC;
```

The **PTL_CLK_ACC** enumeration specifies all applicable clock accuracy values, as defined in the PTP spec.

```
typedef struct
{
    UINT32          clockClass;              //  8 bits
    PTL_CLK_ACC     clockAccuracy;
    UINT32          offsetScaledLogVariance; // 16 bite
```

```
} PTL_CLK_QUALITY;
```

The **PTL_CLK_QUALITY** structure holds information about the clock quality of a PTP module.

```
typedef enum
{
    PTL_TIME_SRC_ATOMIC_CLOCK        = 0x10,
    PTL_TIME_SRC_GPS                 = 0x20,
    PTL_TIME_SRC_TERR_RADIO          = 0x30,
    PTL_TIME_SRC_PTP                 = 0x40,
    PTL_TIME_SRC_NTP                 = 0x50,
    PTL_TIME_SRC_HAND_SET            = 0x60,
    PTL_TIME_SRC_OTHER               = 0x90,
    PTL_TIME_SRC_INTERNAL_OSCILLATOR = 0xA0

} PTL_TIME_SRC;
```

The **PTL_TIME_SRC** enumeration specifies all applicable PTP clock sources, as defined in the PTP spec.

```
typedef struct
{
    INT32           utcOffset; // 16 bits signed
    BOOL            utcOffsetValid;
    BOOL            forwardLeap;
    BOOL            backwardLeap;
    BOOL            timeTraceable;
    BOOL            freqTraceable;
    BOOL            ptpTimescale;
    PTL_TIME_SRC    timeSource;

} PTL_TIME_PROP;
```

The **PTL_TIME_PROP** structure holds information about the PTP module's time properties.

```
typedef struct
{
    UINT8           clockIdentity[8];
    UINT32          portNumber;   //  8 bits
    BOOL            statsCalculated;
    UINT32          observedOSLV; // 16 bits
    UINT32          observedCPCR;

} PTL_PARENT_PROP
```

The **PTL_PARENT_PROP** structure holds information that a PTP module has about it's parent. (The parent is the PTP unit on the network that the PTP module is synchronized to).

```
typedef struct
{
    UINT8           clockIdentity[8];
    PTL_CLK_QUALITY clockQuality;
```

```
    UINT32          priority1; // 32 bits
    UINT32          priority2; // 32 bits

} PTL_GM_PROP;
```

The **PTL_GM_PROP** structure holds information that the PTP module has about the network's grandmaster clock.

```
typedef enum
{
    PTL_RESET_COLD = 0,        // Clear data in RAM (like power-cycle)
    PTL_RESET_HOT  = 1,        // No clear, SW reset, rerun self-test
    PTL_RESET_FACT = 2         // Cold reset + restore factory settings

} PTL_RESET;
```

The **PTL_RESET** enumeration defines the different resets possible on the PTP module.

```
typedef struct
{
    BOOL            todEnabled;
    ML_TIME_SCALE   timeScale;

} PTL_TOD_SETTINGS;
```

The **PTL_TOD_SETTINGS** structure holds information that a PTP unit has about the TOD Settings.

```
typedef enum
{
    PTL_OP_MODE_SLAVE            = 0x00,  // Slave-Only operation
    PTL_OP_MODE_MASTER           = 0x01,  // Master-Only Operation

} PTL_OP_MODE;
```

The **PTL_OP_MODE** enumeration defines the different possible operating modes of the PTP Module.

```
typedef enum
{
    PTL_RESET_CAUSE_LOSS_LOCK  = 0,
    PTL_RESET_CAUSE_LOSS_CLOCK = 1,
    PTL_RESET_CAUSE_EXTERNAL   = 2,
    PTL_RESET_CAUSE_POWER      = 3,
    PTL_RESET_CAUSE_WATCHDOG   = 4,
    PTL_RESET_CAUSE_REQUEST    = 5

} PTL_RESET_CAUSE
```

The **PTL_RESET_CAUSE** enumeration defines different ways the PTP Module can be reset.

```
typedef struct
{
    char            deviceName[16];
```

```
    char            deviceLocation[16];

} PTL_USER_DESC
```

The **PTL_USER_DESC** structure stores the PTP User Description text fields. Note that the size of the deviceName and deviceLocation fields is limited to 16 characters each.

8 Public Message API

The Public Message API is described below. This API consists of the command and response message formats. The response message format can differ if the operation succeeds or fails. No command message operation should ever result in a condition in which a response message is not sent. No timeouts are defined for any command message operation in this API.

8.1 Example Message Transaction

An example HIDD message transaction is described below. A message transaction consists of a command, and a response or an error response message. Failure for the host to receive a response within a specified timeout indicates a potential KTS internal fault condition or a loss of connection.

Command Message

A command message consists of a message header and an optional variable length payload. The operation can be either a get or set indicated by the Control Type Get/Set bit mask.

Offset	Field	TYPE	Description
0	cai	UINT8	Component/Agent Identifier
1	iid	UINT8	Item Identifier
2	ctl	UINT16	Control Type bit mask
4	pyldLen	UINT32	Payload Length
8	pyld	UINT8[]	Command payload

Response Message

A response message consists of a message header and an optional variable length payload. The response message returns the same Component Identifier, Item Identifier and Control Type Get/Set bit received in the command message. The Command Type Error bit is never set. The other reserved control bits default to 0.

Offset	Field	TYPE	Description
0	cai	UINT8	Component/Agent Identifier
1	iid	UINT8	Item Identifier
2	ctl	UINT16	Control Type bit mask
4	pyldLen	UINT32	Payload Length
8	pyld	UINT8[]	Command payload

Error Response Message

An error response message consists of a message header, a mandatory error code and an optional variable length payload. The error response payload, if provided, is specific to component and item, and may even be specific to the error code. The response message returns the same Component Identifier, Item Identifier and Control Type Get/Set bit received in the command message. The Error Response bit is set indicating that this message is the result of an error. The other reserved control bits default to 0.

Offset	Field	TYPE	Description
0	cai	UINT8	Component/Agent Identifier
1	iid	UINT8	Item Identifier
2	ctl | CTL_ERR	UINT16	Control Type bit mask
4	pyldLen	UINT32	Payload Length
8	ec	HA_RC	Error code

The default error response message for all command messages is the message header followed by the mandatory Error Code with no payload unless specified by the HIDD.

Offset	Field	TYPE	Description
0	cai	UINT8	Component/Agent Identifier
1	iid	UINT8	Item Identifier
2	ctl | CTL_ERR	UINT16	Control Type bit mask
4	pyldLen	UINT32	Payload Length
8	ec	HA_RC	Error code
12	data	UINT8[]	Error data

The error response message may provide an operation and error specific payload by the HIDD.

8.2 Host Agent API

The HIDD Host Agent API is defined as the set of built-in command and response messages used to communicate with the HA. The HA provides built-in commands which may differ from other agents. Error codes are defined by the HA.

8.2.1 Get Capabilities Transaction (HA_CMD_GET_CAPS)

The Get Capabilities transaction is used to retrieve a structure defining the version and capabilities information of all component interfaces provided by the KTS. This command could be sent at system startup. The Get Capabilities response returns a page of capabilities structures and a flag indicating if there are more pages. The Host issues Get Capabilities request messages until all the pages are retrieved. The Get Capabilities Error Response consists of a message header and error code indicating a fault in the KTS.

Get Capabilities Command

The Get Capabilities command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x01	UINT8	HA Identifier
1	0x01	UINT8	Get Capabilities item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	page	UINT32	Page Number

Get Capabilities Response

The Get Capabilities response consists of a header and a Capabilities structure.

Offset	Field	TYPE	Description
0	0x01	UINT8	HA Identifier
1	0x01	UINT8	Get Capabilities item
2	0x0000	UINT16	Get
4	16*8+8	UINT32	Payload Length
8	capPage	CI_CAP_PAGE	Capabilities page structure
8	page	UINT32	Page Number
12	more	BOOL	More pages follow this one
16	cap0	CI_CAP	Capabilities structure
16	item	CI_ITEM	Item Identifier
20	access	CI_ACCESS	Access
24	cap1	CI_CAP	Capabilities structure
32	cap2	CI_CAP	Capabilities structure
…	…	…	…
15*8+16	cap15	CI_CAP	Capabilities structure

Get Capabilities Error Response

The Get Capabilities Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x01	UINT8	HA Identifier
1	0x01	UINT8	Get Capabilities item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3 Component API

The HIDD Component API is defined as the set of command and response messages used to communicate with each component or component derived type.

8.3.1 Services

Each service and its messages are defined in this section.

8.3.1.1 Initializer

The messages used by the Initializer (IN) are defined below.

8.3.1.1.1 Status Transaction (IN_CA_STATUS)

The Status transaction is used to retrieve the KTS startup status. A Status response is returned containing the Initializer's **IN_STATUS** structure containing the entire table of initialization results for all objects in the KTS. The Status Error Response consists of a message header and error code indicating a failure of the Initializer to carry out the operation.

Status Command

The Status command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x20	UINT8	IN Identifier
1	0x00	UINT8	Status item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	page	UINT32	Page number

Status Response

The Status response returns the Initializer's **IN_STATUS** structure containing the entire table of initialization results for all objects in the KTS.

Offset	Field	TYPE	Description
0	0x20	UINT8	IN Identifier
1	0x00	UINT8	Status item
2	0x0000	UINT16	Get
4	16*36+8	UINT32	Payload Length
8	status	IN_STATUS	Status structure
8	pgno	UINT32	Page number
12	more	BOOL	More pages follow this one
16	result0	UINT32	Result structure
16	module	char[32]	Module String
48	result	RC	Module Status
52	result1	IN_RESULT	Result structure
88	result2	IN_RESULT	Result structure
15*36+16	result15	IN_RESULT	Result structure

Status Error Response

The Status Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x20	UINT8	IN Identifier
1	0x00	UINT8	Status item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.2 Discovery and Configuration Service

The messages used by the Discovery and Configuration Service (DCS) are defined below.

8.3.1.2.1 Option Card Information Transaction (DCS_CA_OC_INFO)

The Option Card Information transaction is used to retrieve information about the option card in a specified option card slot. An Option Card Information response is returned containing the option card information. The Option Card Information Error Response consists of a message header and error code indicating a failure of the DCS to carry out the operation.

Option Card Information Command

The Option Card Information command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x00	UINT8	Option Card Information item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	slot	UINT32	Option Card slot

Option Card Information Response

The Option Card Information response returns the option card information.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x00	UINT8	Option Card Information item
2	0x0000	UINT16	Get
4	28	UINT32	Payload Length
8	handle	OCL_HDR	Option Card Header structure
8	id	UINT32	Option Card ID
12	version	UINT32	Option Card Version
16	card info	OCL_CISH	Card Info Header structure
16	id	UINT32	Programmable Logic Device ID
20	version	UINT32	Programmable Logic Device Version
24	rsv	UINT32	Unused (reserved)
28	rsv	UINT32	Unused (reserved)
32	numfr	UINT32	Number of Feature Records

Option Card Information Error Response

The Option Card Information Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x00	UINT8	Option Card Information item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.2.2 Feature by Index Transaction (DCS_CA_OC_FEAT_IDX)

The Feature by Index transaction is used to retrieve information about the option card in a specified option card slot. A Feature by Index response is returned containing the option card information. The Feature by Index Error Response consists of a message header and error code indicating a failure of the DCS to carry out the operation.

Feature by Index Command

The Feature by Index command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x01	UINT8	Feature by Index item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	specifier	OCL_SFI	Slot/Index specifier
8	slot	INT32	Option Card slot
12	index	UINT32	Feature index

Feature by Index Response

The Feature by Index response returns the feature information.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x01	UINT8	Feature by Index item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	feature info	OCL_FI	Feature Instance structure
8	id	UINT32	Feature ID
12	instance	UINT32	Instance of feature in the system

Feature by Index Error Response

The Feature by Index Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x01	UINT8	Feature by Index item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.2.3 Feature Instance Transaction (DCS_CA_INSTANCE)

The Feature Instance transaction is used to retrieve information about the option card in a specified option card slot. A Feature Instance response is returned containing the option card information. The Feature Instance Error Response consists of a message header and error code indicating a failure of the DCS to carry out the operation.

Feature Instance Command

The Feature Instance command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x02	UINT8	Feature Instance item
2	0x0000	UINT16	Get
4	12	UINT32	Payload Length
8	specifier	OCL_LFI	Slot/ID/Instance specifier
8	slot	INT32	Option Card slot
12	id	UINT32	Feature ID
16	instance	UINT32	Instance of feature on the card

Feature Instance Response

The Feature Instance response returns the feature instance.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x02	UINT8	Feature Instance item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	INT32	Instance of feature in the system

Feature Instance Error Response

The Feature Instance Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x02	UINT8	Feature Instance item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.2.4 Feature Slot Transaction (DCS_CA_SLOT)

The Feature Slot transaction is used to retrieve information about the option card in a specified option card slot. A Feature Slot response is returned containing the option card information. The Feature Slot Error Response consists of a message header and error code indicating a failure of the DCS to carry out the operation.

Feature Slot Command

The Feature Slot command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x03	UINT8	Feature Slot item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	Feature info	OCL_FI	Feature/Instance structure
8	id	UINT32	Feature ID
12	instance	UINT32	Instance of feature in the system

Feature Slot Response

The Feature Slot response returns the option card slot.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x03	UINT8	Feature Slot item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	slot	INT32	Option Card slot

Feature Slot Error Response

The Feature Slot Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x42	UINT8	DCS Identifier
1	0x03	UINT8	Feature Slot item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.3 Oscillator Monitor

The messages used by the Oscillator Monitor Service (XS) are defined below.

8.3.1.3.1 Register Transaction (XS_CA_REGISTER)

The Register transaction is used to reserve a meter. A Register response is returned containing the meter's handle. The Status Error Response consists of a message header and error code indicating a failure of the XS to carry out the operation.

Register Command

The Register command consists of a header only.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x00	UINT8	Register item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Register Response

The Register response returns the meter's handle.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x00	UINT8	Register item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	handle	UINT32	Meter Handle

Register Error Response

The Register Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x00	UINT8	Register item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.3.2 Unregister Transaction (XS_CA_UNREGISTER)

The Unregister transaction is used to free the specified meter. An Unregister response is returned indicating the meter is freed. The Unregister Error Response consists of a message header and error code indicating a failure of the XS to carry out the operation.

Unregister Command

The Unregister command consists of a header and meter handle.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x01	UINT8	Unregister item
2	0x0002	UINT16	Set
4	4	UINT32	Payload Length
8	handle	UINT32	Meter Handle

Unregister Response

The Unregister response returns a header only.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x01	UINT8	Unregister item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Unregister Error Response

The Unregister Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x01	UINT8	Unregister item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.3.3 Meter Command Transaction (XS_CA_METER_CMD)

The Meter Command transaction is used to execute an oscillator meter command on the specified meter. A Meter Command response is returned indicating the oscillator meter command is performed. The Meter Command Error Response consists of a message header and error code indicating a failure of the XS to carry out the operation.

Meter Command

The Meter command consists of a header, a meter handle and an oscillator meter command.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x02	UINT8	Meter Command item
2	0x0002	UINT16	Set
4	8	UINT32	Payload Length
8	handle	UINT32	Meter Handle
12	command	XS_CMD	Oscillator Meter command

Meter Command Response

The Meter Command response returns a header only.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x02	UINT8	Meter Command item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Meter Command Error Response

The Meter Command Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x02	UINT8	Meter Command item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.3.4 Window Size Transaction (XS_CA_WINDOW_SIZE)

The Window Size transaction is used to get or set the meter's window size. The Meter Command Error Response consists of a message header and error code indicating a failure of the XS to carry out the operation.

Window Size Command

The Window Size command consists of a header, a meter handle and a window size when set and a header only with get.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x03	UINT8	Window Size item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	handle	UINT32	Meter Handle
12	size	UINT32	Window Size (Set only)

Window Size Response

The Meter Command get response returns the meter's windows size. The Meter Command set response consists of the header only.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x03	UINT8	Window Size item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	size	UINT32	Window Size (Get only)

Window Size Error Response

The Meter Command Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x03	UINT8	Window Size item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.3.5 Meter Data Transaction (XS_CA_METER_DATA)

The Meter Data transaction is used to get the meter's state structure. A Meter Data response returns the meter's state data structure. The Meter Data Error Response consists of a message header and error code indicating a failure of the XS to carry out the operation.

Meter Data Command

The Meter Data command consists of a header and a meter handle.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x04	UINT8	Meter Data item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	handle	UINT32	Meter Handle

Meter Data Response

The Meter Data response returns the meter's state data structure.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x04	UINT8	Meter Data item
2	0x0000	UINT16	Get command
4	32	UINT32	Payload Length
8	meterData	XS_METER_DATA	Meter data
8	state	XS_STATE	Meter State
12	size	UINT32	Window Size
16	elapsed	UINT32	Elapsed duration
20	frAccum	INT32	Accumulated frequiency error
24	frPrev	INT32	Total frequency error of previous window
28	phStart	INT32	Starting phase error
32	phAccum	INT32	Accumulated phase error
36	phPrev	INT32	Total phase error of previous window

Meter Data Error Response

The Meter Data Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x21	UINT8	XS Identifier
1	0x04	UINT8	Meter Data item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4 Clock

The messages used by the Clock Service (CS) are defined below.

8.3.1.4.1 Time Scale Transaction (CS_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the clock's Time Scale. A time scale response is returned containing the clock's time scale when a get is issued. The Time Scale Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

Time Scale Command

The Time Scale get command consists of a header only. The Time Scale set command consists of a header and the new time scale.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x01	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Get only) 4 (Set only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time scale (Set only)

Time Scale Response

The Time Scale get response returns with the clocks time scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x01	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x01	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4.2 Time Scale Offset Transaction (CS_CA_TIME_SCALE_OFF)

The Time Scale Offset transaction is used to get or set the clock's Time Scale Offset. A Time Scale Offset response is returned containing the clock's time scale offset when a get is issued. The Time Scale Offset Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

Time Scale Offset Command

The Time Scale Offset get command consists of a header only. The Time Scale Offset set command consists of a header and the new time scale.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x02	UINT8	Time Scale Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time scale
12	tsOffset	INT32	Time scale offset (Set only)

Time Scale Offset Response

The Time Scale Offset get response returns with the clock's time scale offset. The Time Scale Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x02	UINT8	Time Scale Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	INT32	Time scale offset (Get only)

Time Scale Offset Error Response

The Time Scale Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x02	UINT8	Time Scale Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4.3 Sub-second Adjustment Transaction (CS_CA_SUBSEC_ADJ)

The Sub-second Adjustment transaction is used to set the clock's one-time Sub-second Adjustment. The Sub-second Adjustment Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

Sub-second Adjustment Command

The Sub-second Adjustment set command consists of a header and the sub-second adjustment.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x03	UINT8	Sub-second Adjustment item
2	0x0002	UINT16	Set
4	4	UINT32	Payload Length
12	ssAdjust	INT32	Sub-second adjustment

Sub-second Adjustment Response

The Sub-second Adjustment set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x03	UINT8	Sub-second Adjustment item
2	control	UINT16	Get or Set
4	0	UINT32	Payload Length

Sub-second Adjustment Error Response

The Sub-second Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x03	UINT8	Sub-second Adjustment item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4.4 Time Transaction (CS_CA_TIME)

The Time transaction is used to get or set the clock's time. A Time response is returned containing the clock's time when a get is issued. The Time Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

Time Command

The Time get command consists of a header and a **ML_TIME_TYPE** time type. The Time set command consists of a header and the new time defined as a **ML_TIME** structure in the specified format with any unused bytes set to 0x00.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x05	UINT8	Time item
2	control	UINT16	Get or Set
4	4 (Get only) 36 (Set only)	UINT32	Payload Length
8	type	ML_TIME_TYPE	Time type
8	timeInfo	ML_TIME	Time structure (Set only)
8	type	ML_TIME_TYPE	Time type (Set only)
12	time	ML_TIME_UNION	Time (Set only)

Time Response

The Time get response returns with the clock's time in a **ML_TIME** structure in the specified time format with any unused bytes set to 0x00. The Time set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x05	UINT8	Time item
2	control	UINT16	Get or Set
4	0 (Set only) 32 (Get only)	UINT32	Payload Length
8	timeInfo	ML_TIME	Time structure (Set only)
8	type	ML_TIME_TYPE	Time type (Set only)
12	time	ML_TIME_UNION	Time (Set only)

Time Error Response

The Time Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x05	UINT8	Time item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4.5 Leap Second Transaction (CS_CA_LEAP_SEC)

The Leap Second transaction is used to get or set the clock's Leap Second value and adjustment time. A Leap Second response is returned containing the clock's Leap Second value and adjustment time when a get is issued. The Leap Second Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

Leap Second Command

The Leap Second get command consists of a header only. The Leap Second set command consists of a header and the Leap Second adjustment defined as a **ML_LEAP_SEC**.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x07	UINT8	Leap Second item
2	control	UINT16	Get or Set
4	0 (Get only) 28 (Set only)	UINT32	Payload Length
8	leapInfo	ML_LEAP_SEC	Leap Second structure (Set only)
8	offset	INT32	Leap Second value (Set only)
12	utcDate	ML_DOYTIME	Time structure (Set only)
12	year	UINT32	Year (Set only)
16	doy	UINT32	Day of year (Set only)
20	hour	UINT32	Hour (Set only)
24	minute	UINT32	Minute (Set only)
28	second	UINT32	Second (Set only)
32	ns	UINT32	Nanosecond (Set only)

Leap Second Response

The Leap Second get response returns with the clock's Leap Second adjustment defined as a **ML_LEAP_SEC**. The Leap Second set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x07	UINT8	Leap Second item
2	control	UINT16	Get or Set
4	0 (Get only) 28 (Set only)	UINT32	Payload Length
8	leapInfo	ML_LEAP_SEC	Leap Second structure (Set only)
8	offset	INT32	Leap Second value (Set only)
12	utcDate	ML_DOYTIME	Time structure (Set only)
12	year	UINT32	Year (Set only)
16	doy	UINT32	Day of year (Set only)
20	hour	UINT32	Hour (Set only)
24	minute	UINT32	Minute (Set only)
28	second	UINT32	Second (Set only)
32	ns	UINT32	Nanosecond (Set only)

Leap Second Error Response

The Leap Second Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x07	UINT8	Leap Second item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4.6 Time Zone Offset Transaction (CS_CA_TIME_ZONE_OFF)

The Time Zone Offset transaction is used to get or set the clock's Time Zone Offset. A Time Zone Offset response is returned containing the clock's time zone offset when a get is issued. The Time Zone Offset Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

Time Zone Offset Command

The Time Zone Offset get command consists of a header only. The Time Zone Offset set command consists of a header and the new time zone offset.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x08	UINT8	Time Zone Offset item
2	control	UINT16	Get or Set
4	0 (Get only) 4 (Set only)	UINT32	Payload Length
8	tzOffset	INT32	Time zone offset (Set only)

Time Zone Offset Response

The Time Zone Offset get response returns with the clock's time zone offset. The Time Zone Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x08	UINT8	Time Zone Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 8 (Get only)	UINT32	Payload Length
8	tzOffset	INT32	Time zone offset(Get only)

Time Zone Offset Error Response

The Time Zone Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x08	UINT8	Time Zone Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4.7 DST Rule Transaction (CS_CA_DST_RULE)

The DST Rule transaction is used to get or set the clock's DST Rule. A DST Rule response is returned containing the clock's DST rule when a get is issued. The DST Rule Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

DST Rule Command

The DST Rule get command consists of a header only. The DST Rule set command consists of a header and the new DST rule.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0A	UINT8	DST Rule item
2	control	UINT16	Get or Set
4	0 (Get only) 40 (Set only)	UINT32	Payload Length
8	rule	ML_DST_RULE	DST rule structure (Set only)
8	ref	ML_DST_REF	DST reference (Set only)
12	in	ML_DST_POINT	DST in point structure (Set only)
12	month	ML_MONTH	Month (Set only)
16	wom	ML_WOM	Week of Month (Set only)
20	dow	ML_DOW	Day of Week (Set only)
24	hour	UINT32	Hour (Set only)
28	out	ML_DST_POINT	DST out point structure (Set only)
28	month	ML_MONTH	Month (Set only)
32	wom	ML_WOM	Week of Month (Set only)
36	dow	ML_DOW	Day of Week (Set only)
40	hour	UINT32	Hour (Set only)
44	offset	INT32	DST offset (Set only)

DST Rule Response

The DST Rule get response returns with the clock's DST rule. The DST Rule set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0A	UINT8	DST Rule item
2	control	UINT16	Get or Set
4	0 (Set only) 40 (Get only)	UINT32	Payload Length
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)

DST Rule Response

The DST Rule Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0A	UINT8	DST Rule item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4.8 Year Transaction (CS_CA_YEAR)

The Year transaction is used to get or set the clock's year. A Year response is returned containing the clock's Year when a get is issued. The Year Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

Year Command

The Year get command consists of a header only. The Year set command consists of a header and the new year.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0B	UINT8	Year item
2	control	UINT16	Get or Set
4	0 (Get only) 4 (Set only)	UINT32	Payload Length
8	year	UINT32	Year (Set only)

Year Response

The Year get response returns with the clock's Year. The Year set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0B	UINT8	Year item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	year	UINT32	Year (Get only)

Year Error Response

The Year Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0B	UINT8	Year item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.4.9 DST State Transaction (CS_CA_DST_STATE)

The DST State transaction is used to get or set the clock's DST flag. A DST State response is returned containing the clock's DST flag when a get is issued. The DST State Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

DST State Command

The DST State get command consists of a header only. The DST State set command consists of a header and the new DST flag.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0C	UINT8	DST State item
2	control	UINT16	Get or Set
4	0 (Get only) 4 (Set only)	UINT32	Payload Length
8	Flag	BOOL	DST flag (Set only)

DST State Response

The DST State get response returns with the clock's DST flag. The DST State set response consists of the header only.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0C	UINT8	DST State item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	flag	BOOL	DST flag (Get only)

DST State Error Response

The DST State Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x23	UINT8	CS Identifier
1	0x0C	UINT8	DST State item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	Ec	HA_RC	HA Error Code
12	Rc	RC	Optional operation specific return code

8.3.1.5 Reference Monitor

The messages used by the Reference Monitor Service (RS) are defined below.

8.3.1.5.1 Factory Default Transaction (RS_CA_FACT_DEFAULT)

The Factory Default transaction is used to reset the RS reference table to the factory default settings. A Factory Default response is returned when completed. The Factory Default Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Factory Default Command

The Factory Default command consists of a header only.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x00	UINT8	Factory Default item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Factory Default Response

The Factory Default response returns when completed.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x00	UINT8	Factory Default item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Factory Default Error Response

The Factory Default Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x00	UINT8	Factory Default item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.2 User Default Transaction (RS_CA_USER_DEFAULT)

The User Default transaction is used to reset the RS reference table to the user default settings. A User Default response is returned when completed. The User Default Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

User Default Command

The User Default command consists of a header only.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x01	UINT8	User Default item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

User Default Response

The User Default response returns when completed.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x01	UINT8	User Default item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

User Default Error Response

The User Default Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x01	UINT8	User Default item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.3 Save User Default Transaction (RS_CA_SAVE_USER_DEFAULT)

The Save User Default transaction is used to save the user default reference table settings. A Save User Default response is returned when completed. The Save User Default Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Save User Default Command

The Save User Default command consists of a header only.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x02	UINT8	Save User Default item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Save User Default Response

The Save User Default response returns when completed.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x02	UINT8	Save User Default item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Save User Default Error Response

The Save User Default Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x02	UINT8	Save User Default item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.4 Best Reference Transaction (RS_CA_BEST_REFERENCE)

The Best Reference transaction is used to get the current best reference table entry. A Best Reference response is returned when completed. The Best Reference Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Best Reference Command

The Best Reference command consists of a header only.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x03	UINT8	Best Reference item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Best Reference Response

The Best Reference response returns when completed with the current best reference table entry in a **RS_TABLE_ENTRY** structure.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x03	UINT8	Best Reference item
2	0x0000	UINT16	Get
4	16	UINT32	Payload Length
8	refInfo	RS_TABLE_ENTRY	Table Entry structure
8	enab	BOOL	Enable
12	prio	UINT32	Priority
16	time	char[4]	Input Time Source String
20	pps	char[4]	Input 1PPS Source String

Best Reference Error Response

The Best Reference Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x03	UINT8	Best Reference item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.5 Table Transaction (RS_CA_TABLE)

The Table transaction is used to get the entire reference table. A Table response is returned when completed. The Table Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Table Command

The Table command consists of a header only.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x04	UINT8	Table item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	table	RS_TABLE_TYPE	Reference table requested

Table Response

The Table response returns when completed with the entire reference table in an array of **RS_TABLE_ENTRY** structures.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x04	UINT8	Table item
2	0x0000	UINT16	Get
4	256	UINT32	Payload Length
8	entry0	RS_TABLE_ENTRY	Table Entry structure
8	enab	BOOL	Enable
12	prio	UINT32	Priority
16	time	char[4]	Input Time Source String
20	pps	char[4]	Input 1PPS Source String
24	entry1	RS_TABLE_ENTRY	Table Entry 1
40	entry2	RS_TABLE_ENTRY	Table Entry 2
…	…	…	…
15*16+8	Entry15	RS_TABLE_ENTRY	Table Entry N

Table Error Response

The Table Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x04	UINT8	Table item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.6 Table Entry Transaction (RS_CA_TABLE_ENTRY)

The Table Entry transaction is used to get a single reference table entry from the working table. A Table Entry response is returned when completed. The Table Entry Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Table Entry Command

The Table Entry command consists of a header and the table entry index.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x05	UINT8	Table Entry item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	index	UINT32	Table entry index

Table Entry Response

The Table Entry response returns when completed with the requested reference table entry in a **RS_TABLE_ENTRY** structure.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x05	UINT8	Table Entry item
2	0x0000	UINT16	Get
4	16	UINT32	Payload Length
8	entry	RS_TABLE_ENTRY	Table Entry structure
8	enab	BOOL	Enable
12	prio	UINT32	Priority
16	time	char[4]	Input Time Source String
20	pps	char[4]	Input 1PPS Source String

Table Entry Error Response

The Table Entry Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x05	UINT8	Table Entry item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.7 Table Entry Add Transaction (RS_CA_TABLE_ENTRY_ADD)

The Table Entry Add transaction is used to add the provided entry to the reference table. A Table Entry Add response is returned when completed. The Table Entry Add Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Table Entry Add Command

The Table Entry Add command consists of a header and a reference table entry in a **RS_TABLE_ENTRY** structure.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x06	UINT8	Table Entry Add item
2	0x0002	UINT16	Set
4	16	UINT32	Payload Length
8	entry	RS_TABLE_ENTRY	Table Entry structure
8	enab	BOOL	Enable
12	prio	UINT32	Priority
16	time	char[4]	Input Time Source String
20	pps	char[4]	Input 1PPS Source String

Table Entry Add Response

The Table Entry Add response returns when completed.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x06	UINT8	Table Entry Add item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Table Entry Add Error Response

The Table Entry Add Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x06	UINT8	Table Entry Add item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.8 Table Entry Delete Transaction (RS_CA_TABLE_ENTRY_DEL)

The Table Entry Delete transaction is used to delete the indicated entry to the reference table. A Table Entry Delete response is returned when completed. The Table Entry Delete Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Table Entry Delete Command

The Table Entry Delete command consists of a header and the table entry index.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x07	UINT8	Table Entry Delete item
2	0x0002	UINT16	Set
4	4	UINT32	Payload Length
8	index	UINT32	Table entry index

Table Entry Delete Response

The Table Entry Delete response returns when completed.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x07	UINT8	Table Entry Delete item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Table Entry Delete Error Response

The Table Entry Delete Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x07	UINT8	Table Entry Delete item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.9 *Priority Transaction (RS_CA_PRIORITY)*

The Priority transaction is used to get and set the priority of the indicated entry in the reference table. A Priority response is returned when completed. The Priority Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Priority Command

The Priority command consists of a header, a table entry index and a priority.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x08	UINT8	Priority item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	index	UINT32	Table entry index
12	prio	UINT32	Priority (Set only)

Priority Response

The Priority get response returns with the requested table entries priority. The Priority set response consists of the header only.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x08	UINT8	Priority item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	prio	UINT82	Priority (Get only)

Priority Error Response

The Priority Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x08	UINT8	Priority item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.10 Enable Transaction (RS_CA_ENABLE)

The Enable transaction is used to get and set the enable flag of the indicated entry in the reference table. An Enable response is returned when completed. The Enable Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Enable Command

The Enable command consists of a header, a table entry index and an enable flag.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x09	UINT8	Enable item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	index	UINT32	Table entry index
12	en	BOOL	Enable (Set only)

Enable Response

The Enable get response returns with the requested table entries priority. The Enable set response consists of the header only.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x09	UINT8	Enable item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	BOOL	Enable (Get only)

Enable Error Response

The Enable Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x09	UINT8	Enable item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.5.11 Get Reference State Table Transaction (RS_CA_REF_STATE_TABLE)

The Get Reference State Table transaction is used to get the time and 1PPS state for each reference registered with the RS. A Get State Table response is returned when completed. The ReferenceGet State Table Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Get Reference State Table Command

The Get Reference State Table command consists of a header, a table entry index and an enable flag.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x0A	UINT8	Reference State Table item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Get Reference State Table Response

The Get Reference State Table get response returns with the requested table entries priority.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x0A	UINT8	Reference State Table item
2	0x0000	UINT16	Get
4	192	UINT32	Payload Length
8	entry0	RS_REF_STATE_ENTRY	Table Entry structure
8	src	char[4]	Input Source
12	timeValid	BOOL	Time Valid
16	ppsValid	BOOL	1PPS Valid
20	entry1	RS_REF_STATE_ENTRY	Table Entry structure
32	entry2	RS_REF_STATE_ENTRY	Table Entry structure
…	…	…	…
15*12+8	entry15	RS_REF_STATE_ENTRY	Table Entry structure

Get Reference State Table Error Response

The Get Reference State Table Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x24	UINT8	RS Identifier
1	0x0A	UINT8	Ref State Table item
2	0x0001	UINT16	Get Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6 Supervisor

The messages used by the Supervisor Service (SS) are defined below.

8.3.1.6.1 Reference Transaction (SS_CA_REF)

The Reference transaction is used to retrieve the current time and 1 PPS references. A Reference response is returned when completed. The Reference Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Reference Command

The Reference command consists of a header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x00	UINT8	Reference item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Reference Response

The Reference response consists of a header and the current 1PPS and Time reference pair.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x00	UINT8	Reference item
2	0x0000	UINT16	Get
4	16	UINT32	Payload Length
8	time	char[4]	Time Reference Identifier
12	pps	char[4]	1PPS Reference Identifier

Reference Error Response

The Reference Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x00	UINT8	Reference item
2	Control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.2 Maximum TFOM Transaction (SS_CA_MAX_TFOM)

The Maximum TFOM transaction is used to get and set the maximum TFOM value used as a threshold in the Sync Rule. An Enable response is returned when completed. The Enable Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Maximum TFOM Command

The Maximum TFOM command consists of a header, a table entry index and an enable flag.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x01	UINT8	Maximum TFOM item
2	control	UINT16	Get or Set
4	0 (Get only) 4 (Set only)	UINT32	Payload Length
8	max	TFOM	Maximum TFOM (Set only)

Maximum TFOM Response

The Maximum TFOM get response returns with the requested maximum TFOM value. The Maximum TFOM set response consists of the header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x01	UINT8	Maximum TFOM item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	max	TFOM	Maximum TFOM (Get only)

Maximum TFOM Error Response

The Maximum TFOM Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x01	UINT8	Maximum TFOM item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.3 TFOM Transaction (ss_ca_tfom)

The TFOM transaction is used to retrieve the current TFOM state. A TFOM response is returned when completed. The TFOM Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

TFOM Command

The TFOM command consists of a header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x02	UINT8	TFOM item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

TFOM Response

The TFOM response consists of a header and the TFOM state.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x02	UINT8	TFOM item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	tfom	TFOM	TFOM

TFOM Error Response

The TFOM Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x02	UINT8	TFOM item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.4 Sync Transaction (SS_CA_SYNC)

The Sync transaction is used to retrieve the current sync state. A Sync response is returned when completed. The Sync Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Sync Command

The Sync command consists of a header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x03	UINT8	Sync item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Sync Response

The Sync response consists of a header and the sync state.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x03	UINT8	Sync item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	sync	BOOL	Sync

Sync Error Response

The Sync Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x03	UINT8	Sync item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.5 Holdover Transaction (SS_CA_HOLDOVER)

The Holdover transaction is used to retrieve the current holdover state. A Holdover response is returned when completed. The Holdover Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Holdover Command

The Holdover command consists of a header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x04	UINT8	Holdover item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Holdover Response

The Holdover response consists of a header and the holdover state.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x04	UINT8	Holdover item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	holdover	BOOL	Holdover

Holdover Error Response

The Holdover Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x04	UINT8	Holdover item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.6 Holdover Timeout Transaction (SS_CA_HOLDOVER_TIMEOUT)

The Holdover Timeout transaction is used to get and set the Holdover Timeout interval in seconds. A Holdover Timeout response is returned when completed. The Holdover Timeout Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Holdover Timeout Command

The Holdover Timeout command consists of a header and a Holdover Timeout value.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x05	UINT8	Holdover Timeout item
2	control	UINT16	Get or Set
4	0 (Get only) 4 (Set only)	UINT32	Payload Length
8	timeout	UINT32	Holdover Timeout

Holdover Timeout Response

The Holdover Timeout get response returns with the requested Holdover Timeout value. The Holdover Timeout set response consists of the header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x05	UINT8	Holdover Timeout item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	timeout	UINT32	Holdover Timeout (Get only)

Holdover Timeout Error Response

The Holdover Timeout Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x05	UINT8	Holdover Timeout item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.7 Timestamp Transaction (SS_CA_TIME_STAMP)

The Timestamp is used to get the requested timestamp indicating the last change of the timestamp variables state in the specified time format. A Timestamp response is returned containing the timestamp. The Timestamp Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Timestamp Command

The Timestamp get command consists of a header, a timestamp source indicating the state variable change time, and a timestamp time format.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x06	UINT8	Timestamp item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	request	***SS_TS_RE172 Q***	Timestamp Request structure
8	source	***SS_TS_SRC***	Timestamp Source
12	type	***ML_TIME_TYPE***	Timestamp type

Timestamp Response

The Timestamp get response returns with the requested Timestamp value which is an ML_TIME type containing the requested timestamp in the specified format.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x06	UINT8	Timestamp item
2	0x0000	UINT16	Get
4	32	UINT32	Payload Length
8	timestamp	ML_TIME	Timestamp

Timestamp Error Response

The Timestamp Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x06	UINT8	Timestamp item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.8 Uptime Transaction (SS_CA_UPTIME)

The Uptime transaction is used to retrieve the current uptime value in quarter hour ticks. An Uptime response is returned when completed. The Uptime Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Uptime Command

The Uptime command consists of a header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x07	UINT8	Uptime item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Uptime Response

The Uptime response consists of a header and uptime in quarter hour ticks.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x07	UINT8	Uptime item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	uptime	UINT32	Uptime

Uptime Error Response

The Uptime Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x07	UINT8	Uptime item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.9 Reset Transaction (SS_CA_RESET)

The Reset transaction is used to reset the KTS. A Reset response is returned when completed. The Reset Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Reset Command

The Reset command consists of a header and reset type value.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x08	UINT8	Reset item
2	0x0002	UINT16	Set
4	4	UINT32	Payload Length
8	type	SS_RESET	Reset type

Reset Response

The Reset response consists of a header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x08	UINT8	Reset item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Reset Error Response

The Reset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x08	UINT8	Reset item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.6.10 Freerun Transaction (ss_ca_freerun)

The Freerun transaction is used to retrieve the current Freerun state. A Freerun response is returned when completed. The Freerun Error Response consists of a message header and error code indicating a failure of the RS to carry out the operation.

Freerun Command

The Freerun command consists of a header only.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x09	UINT8	***FREERUN ITEM***
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Freerun Response

The Freerun response consists of a header and the Freerun state.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x09	UINT8	***FREERUN ITEM***
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	freerun	BOOL	Freerun

Freerun Error Response

The Freerun Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x25	UINT8	SS Identifier
1	0x09	UINT8	***FREERUN ITEM***
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.7 Upgrade

The messages used by the Upgrade Service (SS) are defined below.

8.3.1.7.1 Start Transaction (US_CA_START)

The Start transaction is used to start the upgrade of the KTS using the specified image. A Start response is returned when completed. The Start Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Start Command

The Start command consists of a header, an image type and header.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x00	UINT8	Start item
2	0x0002	UINT16	Set
4	16	UINT32	Payload Length
8	img	FS_IMG	Image type
12	hdr	FS_IMG_HDR	Image header structure
12	mark	UINT32	Image marker
16	type	FS_IMG	Image type
20	Len	UINT32	Image length

Start Response

The Start response consists of a header only.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x00	UINT8	Start item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Start Error Response

The Start Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x00	UINT8	Start item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.7.2 Data Transaction (US_CA_DATA)

The Data transaction is used to transmit data to the KTS to update the specified image. A Data response is returned when completed. The Data Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Data Command

The Data command consists of a header, an image type and data block.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x01	UINT8	Data item
2	0x0002	UINT16	Set
4	1028	UINT32	Payload Length
8	img	FS_IMG	Image type
12	data	UINT8[1024]	Image data block of fixed size

Data Response

The Data response consists of a header only.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x01	UINT8	Data item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Data Error Response

The Data Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x01	UINT8	Data item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.7.3 End Transaction (US_CA_END)

The End transaction is used to finish the update of the specified image. A Data response is returned when completed. The End Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

End Command

The End command consists of a header, and image type and trailer data block.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x02	UINT8	End item
2	0x0002	UINT16	Set
4	12	UINT32	Payload Length
8	img	FS_IMG	Image type
12	trl	FS_IMG_TRL	Image trailer structure
12	ver	char[4]	Image version
16	crc	UINT32	Image CRC

End Response

The End response consists of a header only.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x02	UINT8	End item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

End Error Response

The End Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x02	UINT8	End item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.7.4 Cancel Transaction (US_CA_CANCEL)

The Cancel transaction is used to cancel the update of the specified image. A Cancel response is returned when completed. The Cancel Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Cancel Command

The Cancel command consists of a header and trailer data block.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x03	UINT8	Cancel item
2	0x0002	UINT16	Set
4	4	UINT32	Payload Length
8	img	FS_IMG	Image type

Cancel Response

The Cancel response consists of a header only.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x03	UINT8	Cancel item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Cancel Error Response

The Cancel Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x03	UINT8	Cancel item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.7.5 State Transaction (US_CA_STATE)

The State transaction is used to get the state of the upgrade service. A State response is returned when completed. The State Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

State Command

The State command consists of a header only.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x4	UINT8	State item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

State Response

The State response consists of a header and the upgrade process state values.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x4	UINT8	State item
2	0x0000	UINT16	Get
4	16	UINT32	Payload Length
8	stateInfo	US_STATE	State structure
8	img	FS_IMG	Image type
12	step	US_PROCESS	Update process step
20	complete	UINT32	Percentage complete

State Error Response

The State Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x4	UINT8	State item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.7.6 Start Option Card Transaction (US_CA_START_OC)

The Start Option Card transaction is used to start the upgrade of an Option Card using the specified image. A Start Option Card response is returned when completed. The Start Option Card Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Start Option Card Command

The Start command consists of a header, an image type and header.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x05	UINT8	Start Option Card item
2	0x0002	UINT16	Set
4	16	UINT32	Payload Length
8	img	FS_IMG	Image type
12	hdr	FS_IMG_HDR	Image header structure
12	mark	UINT32	Image marker
16	type	FS_IMG	Image type
20	Len	UINT32	Image length

Start Option Card Response

The Start Option Card response consists of a header only.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x05	UINT8	Start Option Card item
2	0x0002	UINT16	Set
4	0	UINT32	Payload Length

Start Option Card Error Response

The Start Option Card Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x26	UINT8	US Identifier
1	0x05	UINT8	Start Option Card item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.8 Persistent Data

The messages used by the Persistent Data Service (PS) are defined below.

None are defined at this time.

8.3.1.9 Flash Manager

The messages used by the Flash Manager Service (FS) are defined below.

8.3.1.9.1 Get CRC Transaction (FS_CA_GET_CRC)

The Get CRC transaction is used to get the specified images CRC. A Get CRC response is returned when completed. The Get CRC Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Get CRC Command

The Get CRC command consists of a header and an image type.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x00	UINT8	Get CRC item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	img	FS_IMG	Image type

Get CRC Response

The Get CRC response consists of a header and the specified image's CRC.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x00	UINT8	Get CRC item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	crc	UINT32	Image CRC

Get CRC Error Response

The Get CRC Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x00	UINT8	Get CRC item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.9.2 Calculate CRC Transaction (FS_CA_CALC_CRC)

The Calculate CRC transaction is used to calculate the specified images CRC. A Calculate CRC response is returned when completed. The Calculate CRC Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Calculate CRC Command

The Calculate CRC command consists of a header and an image type.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x01	UINT8	Calculate CRC item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	img	FS_IMG	Image type

Calculate CRC Response

The Calculate CRC response consists of a header and the specified image's calculated CRC.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x01	UINT8	Calculate CRC item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
12	crc	UINT32	Image CRC

Calculate CRC Error Response

The Calculate CRC Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x01	UINT8	Calculate CRC item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.9.3 Image Header Transaction (FS_CA_IMG_HEADER)

The Image Header transaction is used to get the specified image header. An Image Header response is returned when completed. The Image Header Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Image Header Command

The Image Header command consists of a header and an image type.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x02	UINT8	Image Header item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	img	FS_IMG	Image type

Image Header Response

The Image Header response consists of a header and the specified image's header.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x02	UINT8	Image Header item
2	0x0000	UINT16	Get
4	12	UINT32	Payload Length
8	hdr	FS_IMG_HDR	Image header structure
8	mark	UINT32	Image marker
12	type	FS_IMG	Image type
16	Len	UINT32	Image length

Image Header Error Response

The Image Header Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x02	UINT8	Image Header item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.9.4 Get Version Transaction (FS_CA_GET_VERSION)

The Get Version transaction is used to get the specified image version string. A Get Version response is returned when completed. The Get Version Error Response consists of a message header and error code indicating a failure of the SS to carry out the operation.

Get Version Command

The Get Version command consists of a header and an image type.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x03	UINT8	Version String item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	img	FS_IMG	Image type

Get Version Response

The Get Version response consists of a header and the specified image's version string.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x03	UINT8	Version String item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	ver	char[4]	Image version

Get Version Error Response

The Get Version Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x28	UINT8	FS Identifier
1	0x03	UINT8	Version String item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.10 Log Service

The messages used by the Log Service (LS) are defined below.

8.3.1.10.1 Error Log Transaction (LS_CA_ERROR_LOG)

The Error Log transaction is used to get an entry off the error log. An Error Log response is returned when completed containing a log entry string. The Error Log Response consists of a message header and error code indicating a failure of the LS to carry out the operation.

Error Log Command

The Error Log command consists of a header only.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x00	UINT8	Error Log item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Error Log Response

The Error Log response consists of a header and the Error Log string.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x00	UINT8	Error Log item
2	0x0000	UINT16	Get
4	120	UINT32	Payload Length
8	ASCII	char[120]	Error Log Entry string

Error Log Error Response

The Error Log Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x00	UINT8	Error Log item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.10.2 Alarm Command Transaction (LS_CA_ALARM)

The Alarm Command transaction is used to get and set the specified Alarm state. An Alarm Command response is returned when completed. At this time only the Software Error alarm status is settable to FALSE to clear this alarm condition. The Alarm Command Error Response consists of a message header and error code indicating a failure of the LS to carry out the operation.

Alarm Command

The Alarm Command consists of a header and an Alarm index value when a get command. If the Alarm Command is used to set an alarm the command consists of a header, an Alarm Index and a BOOL flag.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x01	UINT8	Alarm item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	index	***LS_ALARM***	Alarm index
12	State	***BOOL***	Alarm state (Set only)

Alarm Command Response

The Alarm Command get response returns with the requested Alarm state value. The Alarm Command set response consists of the header only.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x01	UINT8	Alarm item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	State	***BOOL***	Alarm state (Get only)

Alarm Command Error Response

The Alarm Command Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x01	UINT8	Alarm item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.10.3 Firmware Version Transaction (LS_CA_VERSION)

The Firmware Version transaction is used to get the firmware version string. A Firmware Version Response is returned when completed containing the version string of the firmware. The Firmware Version Error Response consists of a message header and error code indicating a failure of the LS to carry out the operation.

Firmware Version Command

The Firmware Version Command consists of a header only.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x02	UINT8	Firmware Version item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Firmware Version Response

The Firmware Version Response consists of a header and the Error Log string.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x02	UINT8	Firmware Version item
2	0x0000	UINT16	Get
4	6	UINT32	Payload Length
8	ASCII	char[6]	Firmware Version string

Firmware Version Error Response

The Firmware Version Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x02	UINT8	Firmware Version item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.1.10.4 Serial Number Transaction (LS_CA_SERIAL_NO)

The Serial Number transaction is used to get the product serial number string. A Serial Number Response is returned when completed containing the serial number string of the firmware. The Serial Number Error Response consists of a message header and error code indicating a failure of the LS to carry out the operation.

Serial Number Command

The Serial Number Command consists of a header only.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x03	UINT8	Serial Number item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Serial Number Response

The Serial Number Response consists of a header and the serial number string.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x03	UINT8	Serial Number item
2	0x0000	UINT16	Get
4	32	UINT32	Payload Length
8	ASCII	char[32]	Serial Number string

Serial Number Error Response

The Serial Number Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x40	UINT8	LS Identifier
1	0x03	UINT8	Serial Number item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2 Components

Each component and its messages are defined in this section.

8.3.2.1 GPS Reference

The messages used by the GPS Reference Component (GR) are defined below.

8.3.2.1.1 Offset Transaction (GR_CA_OFFSET)

The Offset transaction is used to get or set the GPS reference's 1PPS input offset. The Offset Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the input offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Set only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x00	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.2 Validity Transaction (GR_CA_VALIDITY)

The Validity transaction is used to get the time and 1PPS validity structure. The Validity Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Validity Command

The Validity command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Validity Response

The Validity response returns the time and 1PPS validity structure.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	validInfo	EL_VALIDITY	Validity structure
8	timeValid	BOOL	Time validity state
12	ppsValid	BOOL	1PPS validity state

Validity Error Response

The Validity Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x02	UINT8	Validity item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.3 Position Transaction (GR_CA_POSITION)

The Position transaction is used to get or set the position. The Position Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Position Command

The Position command consists of a header, and a position structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x03	UINT8	Position item
2	control	UINT16	Get or Set
4	4 (Get only) 28 (Set only)	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance
12	posInfo	POS_LLA	Position structure (Set only)
12	lat	FLT64	Latitude (Set only)
20	lon	FLT64	Altitude (Set only)
28	alt	FLT64	Altitude (Set only)

Position Response

The Position get response returns a position structure. The Position set response consists of the header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x03	UINT8	Position item
2	control	UINT16	Get or Set
4	0 (Set only) 24 (Get only)	UINT32	Payload Length
8	posInfo	POS_LLA	Position structure (Get only)
8	lat	FLT64	Latitude (Get only)
16	lon	FLT64	Longitude (Get only)
20	alt	FLT64	Altitude (Get only)

Position Error Response

The Position Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x03	UINT8	Position item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.4 Receiver Mode Transaction (GR_CA_RCVR_MODE)

The Receiver Mode transaction is used to get or set the GPS receiver mode. The GPS Receiver dynamics is also set as well; due to the fact some receivers have a dependency between mode and dynamics. The Receiver Mode Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Receiver Mode Command

The Receiver Mode command consists of a header, and a GPS receiver mode and dynamics when set and a header only with get.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x04	UINT8	Receive Mode item
2	control	UINT16	Get or Set
4	4 (Get only) 12 (Set only)	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance
12	mode	GL_MODE	Receiver Mode (Set only)
16	dyn	GL_DYN	Receiver Dynamics (Set only)

Receiver Mode Response

The Receiver Mode get response returns the GPS receiver mode. The Receiver Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x04	UINT8	Receive Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 8 (Get only)	UINT32	Payload Length
8	mode	GL_MODE	Receiver Mode (Get only)
12	dyn	GL_DYN	Receiver Dynamics (Get only)

Receiver Mode Error Response

The Receiver Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x04	UINT8	Receive Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.5 Dynamics Mode Transaction (GR_CA_DYNAMICS)

The Dynamics Mode transaction is used to get or set the GPS Dynamics mode. The Dynamics Mode Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Dynamics Mode Command

The Dynamics Mode command consists of a header, and a GPS Dynamics mode when set and a header only with get.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x05	UINT8	Dynamics Mode item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance
12	mode	GL_DYN	Dynamics Mode (Set only)

Dynamics Mode Response

The Dynamics Mode get response returns the GPS Dynamics mode. The Dynamics Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x05	UINT8	Dynamics Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	mode	GL_DYN	Dynamics Mode (Get only)

Dynamics Mode Error Response

The Dynamics Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x05	UINT8	Dynamics Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.6 Fix Data Transaction (GR_CA_FIX_DATA)

The Fix Data transaction is used to get GPS Position Fix data. The Fix Data Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Fix Data Command

The Fix Data command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x06	UINT8	Fix Data item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Fix Data Response

The Fix Data get response returns the GPS Position Fix data.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x06	UINT8	Fix Data item
2	0x0000	UINT16	Set
4	36	UINT32	Payload Length
8	fixInfo	GL_FIX_INFO	GPS Fix structure
8	nSats	UINT32	Number of Satellites used for fix
12	pdop	FLT32	Position DOP
16	hdop	FLT32	Horizontal DOP
20	vdop	FLT32	Vertical DOP
24	tdop	FLT32	Time DOP
28	fom	UINT32	FOM
32	tfom	UINT32	Time FOM
36	herr	UINT32	Horizontal error
40	verr	UINT32	Vertical error

Fix Data Error Response

The Fix Data Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x06	UINT8	Fix Data item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.7 Satellite Data Transaction (GR_CA_SAT_DATA)

The Satellite Data transaction is used to get GPS Satellite data. The Satellite Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Satellite Data Command

The Satellite Data command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x07	UINT8	Satellite Data item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Satellite Data Response

The Satellite Data get response returns the GPS Satellite data.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x07	UINT8	Satellite Data item
2	0x0000	UINT16	Set
4	32*24+8	UINT32	Payload Length
8	satInfo	GR_SAT_DATA	Satellite Data structure
8	info0	GL_SAT_INFO	Satellite Info structure
8	chnum	UINT32	Channel Number
12	svId	UINT32	ID of SV
16	str	UINT32	Signal strength
20	traim	BOOL	TRAIM accepted BOOL
24	infix	BOOL	Used for pos or time fix BOOL
28	flags	UINT32	Bit flags
32	info1	GL_SAT_INFO	Satellite Info structure
56	info2	GL_SAT_INFO	Satellite Info structure
…	…	…	…
31*24+8	info31	GL_SAT_INFO	Satellite Info structure

Satellite Data Error Response

The Satellite Data Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x07	UINT8	Satellite Data item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.8 Survey Progress Transaction (GR_CA_SURVEY_PROG)

The Survey Progress transaction is used to get the survey progress. The Survey Progress Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Survey Progress Command

The Survey Progress command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x08	UINT8	Survey Progress item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Survey Progress Response

The Survey Progress response returns the survey progress.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x08	UINT8	Survey Progress item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	prog	UINT32	Survey Progress

Survey Progress Error Response

The Survey Progress Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x08	UINT8	Survey Progress item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.9 Manufacturer Model Transaction (GR_CA_MFR_MDL)

The Manufacturer Model transaction is used to get the GPS receiver manufacturer and model. The Manufacturer Model Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Manufacturer Model Command

The Manufacturer Model command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0B	UINT8	Manufacturer Model item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Manufacturer Model Response

The Manufacturer Model response returns the GPS receiver manufacturer and model.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0B	UINT8	Manufacturer Model item
2	0x0000	UINT16	Get
4	32	UINT32	Payload Length
8	mfrmdl	GR_MFR_MDL	Manufacturer/Model structure
8	mfr	char[16]	Manufacturer string
24	mdl	char[16]	Model string

Manufacturer Model Error Response

The Manufacturer Model Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0B	UINT8	Manufacturer Model item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.10 Receiver Info Transaction (GR_CA_RCVR_INFO)

The Receiver Info transaction is used to get the GPS receiver manufacturer specific information including receiver version, software revision, hardware revision and serial numbers. The Receiver Info Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Receiver Info Command

The Receiver Info command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0C	UINT8	Receiver Info item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Receiver Info Response

The Receiver Info response returns the GPS receiver specific information.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0C	UINT8	Receiver Info item
2	0x0000	UINT16	Get
4	256	UINT32	Payload Length
8	info	UINT8[256]	Receiver Info

Receiver Info Error Response

The Receiver Info Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0C	UINT8	Receiver Info item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.11 Custom Message Transaction (GR_CA_CUSTOM_MSG)

The Custom Message transaction is used to get a custom message response from the GPS or send a Custom Message to the GPS. The Custom Message Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Custom Message Command

The Custom Message command consists of a header, and a custom message when set and a header only with get.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0D	UINT8	Custom Message item
2	control	UINT16	Get or Set
4	4 (Get only) 264 (Set only)	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance
12	msgInfo	GR_CUSTOM_MSG	Custom Message structure (Set only)
12	len	UINT32	Length (Set only)
16	msg	char[256]	Custom Message (Set only)

Custom Message Response

The Custom Message get response returns the Custom Message response. The Custom Message set response consists of the header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0D	UINT8	Custom Message item
2	control	UINT16	Get or Set
4	0 (Set only) 260 (Get only)	UINT32	Payload Length
8	msgInfo	GR_CUSTOM_MSG	Custom Message structure (Get only)
8	len	UINT32	Length (Get only)
12	msg	char[256]	Custom Message response (Get only)

Custom Message Error Response

The Custom Message Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0D	UINT8	Custom Message item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.12 Get Number of Instances Transaction (GR_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of GPS References present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0E	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of GPS References present in the system.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0E	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	Num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0E	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.13 Delete Stored Position Transaction (GR_CA_DEL_POS)

The Delete Stored Position transaction is used to clear any position information that is stored in persistent memory inside the GPS receiver.

Delete Stored Position Command

The Delete Stored Position command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0F	UINT8	Delete Stored Position item
2	control	UINT16	Set
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Delete Stored Position Response

The Delete Stored Position Command get response does not return any data.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0F	UINT8	Delete Stored Position item
2	control	UINT16	Set
4	0	UINT32	Payload Length

Delete Stored Position Error Response

The Delete Stored Position Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x0F	UINT8	Delete Stored Position item
2	control|0x0001	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.14 Reference Identifier Transaction (GR_CA_REF_ID)

The Reference Identifier transaction is used to retrieve the 4-character reference identifier for a specified GPS reference instance.

Reference Identifier Command

The Reference Identifier Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x10	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Reference Identifier Response

The Reference Identifier Response returns the reference identifier string for the specified GPS reference instance.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x10	UINT8	Reference Identifier item
2	control	UINT16	Get
4	0	UINT32	Payload Length
8	ref ID	char[4]	Reference Identifier

Reference Identifier Error Response

The Reference Identifier Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x10	UINT8	Reference Identifier item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.15 GPS Reset Transaction (GR_CA_RESET)

The GPS Reset transaction is used to issue a specified reset type to for a specified GPS reference instance.

GPS Reset Command

The GPS Reset command consists of a header only.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x11	UINT8	GPS Reset item
2	control	UINT16	Set
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance
12	reset	GL_RESET	Reset type

GPS Reset Response

The GPS Reset Command get response does not return any data.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x11	UINT8	GPS Reset item
2	control	UINT16	Set
4	0	UINT32	Payload Length

GPS Reset Error Response

The GPS Reset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x11	UINT8	GPS Reset item
2	control|0x0001	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.1.16 Antenna Status Transaction (GR_CA_ANTENNA)

The Antenna Status transaction is used to retrieve the GPS Antenna Status for a specified GPS reference instance.

Antenna Status Command

The Antenna Status Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x12	UINT8	GPS Antenna Status item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	GPS Reference Instance

Antenna Status Response

The Antenna Status Response returns the GPS Antenna Status for the specified GPS reference instance.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x12	UINT8	GPS Antenna Status item
2	control	UINT16	Get
4	0	UINT32	Payload Length
8	status	GL_ANT_STATUS	GPS Antenna Status

Antenna Status Error Response

The Antenna Status Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x29	UINT8	GR Identifier
1	0x12	UINT8	GPS Antenna Status item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2 IRIG Reference

The messages used by the IRIG Reference Component (IR) are defined below. Multiple IRIG References are indexed by an instance number ranging from 0 to N-1 where N is the number of IRIG References a product has.

8.3.2.2.1 Offset Transaction (IR_CA_OFFSET)

The Offset transaction is used to get or set the IRIG reference's 1PPS input offset. The Offset Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the input offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x00	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.2 Validity Transaction (IR_CA_VALIDITY)

The Validity transaction is used to get the time and 1PPS validity structure of the specified IRIG Reference. The Validity Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Validity Command

The Validity command consists of a header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance

Validity Response

The Validity response returns the time and 1PPS validity structure.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	validInfo	EL_VALIDITY	Validity structure
8	timeValid	BOOL	Time validity state
12	ppsValid	BOOL	1PPS validity state

Validity Error Response

The Validity Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x02	UINT8	Validity item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.3 Mode Transaction (IR_CA_MODE)

The Mode transaction is used to get or set the IRIG mode. The Mode Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Mode Command

The Mode command consists of a header, and the mode when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x03	UINT8	Mode item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	mode	IL_MODE	Mode (Set only)

Mode Response

The Mode get response returns the mode. The Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x03	UINT8	Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	mode	IL_MODE	Mode (Get only)

Mode Error Response

The Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x03	UINT8	Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.4 Format Transaction (IR_CA_FORMAT)

The Format transaction is used to get or set the IRIG Format. The Format Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Format Command

The Format command consists of a header, and the format when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x04	UINT8	Format item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	fmt	IL_FMT	Format (Set only)

Format Response

The Format get response returns the format. The Format set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x04	UINT8	Format item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	fmt	IL_FMT	Format (Get only)

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x04	UINT8	Format item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.5 Modulation Transaction (IR_CA_MOD)

The Modulation transaction is used to get the IRIG Modulation. The Modulation Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Modulation Command

The Modulation command consists of a header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x05	UINT8	Modulation item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance

Modulation Response

The Modulation get response returns the modulation.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x05	UINT8	Modulation item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	mod	IL_MOD	Modulation

Modulation Error Response

The Modulation Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x05	UINT8	Modulation item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.6 Frequency Transaction (IR_CA_FREQ)

The Frequency transaction is used to get or set the IRIG carrier frequency. The Frequency Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Frequency Command

The Frequency command consists of a header, and the IRIG carrier frequency when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x06	UINT8	Frequency item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	freq	IL_FRQ	Frequency (Set only)

Frequency Response

The Frequency get response returns the IRIG carrier frequency. The Frequency set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x06	UINT8	Frequency item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	freq	IL_FRQ	Frequency (Get only)

Frequency Error Response

The Frequency Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x06	UINT8	Frequency item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.7 Coded Expression Transaction (IR_CA_CODED_EXP)

The Coded Expression transaction is used to get or set the IRIG Coded Expression. The Coded Expression Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Coded Expression Command

The Coded Expression command consists of a header, and the IRIG Coded Expression when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x07	UINT8	Coded Expression item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	ce	IL_CE	Coded Expression (Set only)

Coded Expression Response

The Coded Expression get response returns the IRIG Coded Expression. The Coded Expression set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x07	UINT8	Coded Expression item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ce	IL_CE	Coded Expression (Get only)

Coded Expression Error Response

The Coded Expression Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x07	UINT8	Coded Expression item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.8 Control Field Transaction (IR_CA_CTRL_FLD)

The Control Field transaction is used to get or set the IRIG Control Field. The Control Field Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Control Field Command

The Control Field command consists of a header, and the IRIG Control Field when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x08	UINT8	Control Field item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	cf	IL_CF	Control Field (Set only)

Control Field Response

The Control Field get response returns the IRIG Control Field. The Control Field set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x08	UINT8	Control Field item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	cf	IL_CF	Control Field (Get only)

Control Field Error Response

The Control Field Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x08	UINT8	Control Field item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.9 Message Transaction (IR_CA_MESSAGE)

The Message transaction is used to get and set the latest IRIG input message. The Message Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation. The get message is enabled by default. The set message is enabled as part of the product configuration.

Message Command

The Message command consists of a header only for get and the header and the entire sub-frame to send for set.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x09	UINT8	Message item
2	control	UINT16	Get or Set
4	4 (Get only) 24 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	msg	IR_MSG	IRIG message structure (Set only)
12	subframe	UINT16	Subframe P0 (Set only)
14	subframe	UINT16	Subframe P1 (Set only)
16	subframe	UINT16	Subframe P2 (Set only)
18	subframe	UINT16	Subframe P3 (Set only)
20	subframe	UINT16	Subframe P4 (Set only)
22	subframe	UINT16	Subframe P5 (Set only)
24	subframe	UINT16	Subframe P6 (Set only)
26	subframe	UINT16	Subframe P7 (Set only)
28	subframe	UINT16	Subframe P8 (Set only)
30	subframe	UINT16	Subframe P9 (Set only)

Message Response

The Message response returns the time and last IRIG input message on get. The Message response consists of a header only on set.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x09	UINT8	Message item
2	control	UINT16	Get or Set
4	0 (Set only) 20 (Get only)	UINT32	Payload Length
8	msg	IR_MSG	IRIG message structure (Get only)
8	subframe	UINT16	Subframe P0 (Get only)
10	subframe	UINT16	Subframe P1 (Get only)
12	subframe	UINT16	Subframe P2 (Get only)
14	subframe	UINT16	Subframe P3 (Get only)
16	subframe	UINT16	Subframe P4 (Get only)
18	subframe	UINT16	Subframe P5 (Get only)
20	subframe	UINT16	Subframe P6 (Get only)
22	subframe	UINT16	Subframe P7 (Get only)
24	subframe	UINT16	Subframe P8 (Get only)
26	subframe	UINT16	Subframe P9 (Get only)

Message Error Response

The Message Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x09	UINT8	Message item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.10 Get Number of Instances Transaction (IR_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of IRIG References present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of IRIG References present in the system.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.11 Control Field Data Transaction (IR_CA_CFDATA)

The Control Field Data transaction is used to get the current control field data received. The Control Field Data Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Control Field Data Command

The Control Field Data command consists of a header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0B	UINT8	Control Field Data item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance

Control Field Data Response

The Control Field Data response consists of a header and the current control field data.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0B	UINT8	Control Field Data item
2	0x0000	UINT16	Get
4	6	UINT32	Payload Length
8	cfData	UINT16[3]	Control Field Data

Control Field Data Error Response

The Control Field Data Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0B	UINT8	Control Field Data item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.12 Local Clock Transaction (IR_CA_LOCAL)

The Local Clock transaction is used to get or set the Local Time information. The Local Clock Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Local Clock Command

The Local Clock command consists of a header, and the local clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0C	UINT8	Local Clock item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Clock Response

The Local Clock get response returns the local clock structure. The Local Clock set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0C	UINT8	Local Clock item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Clock Error Response

The Local Clock Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0C	UINT8	Local Clock item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.13 Time Scale Transaction (IR_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the IRIG input's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the IR to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the IRIG Input's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0D	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the IRIG Input's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0D	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0D	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.2.14 Reference Identifier Transaction (IR_CA_REF_ID)

The Reference Identifier transaction is used to retrieve the 4-character reference identifier for a specified IRIG reference instance.

Reference Identifier Command

The Reference Identifier Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0E	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	IRIG Reference Instance

Reference Identifier Response

The Reference Identifier Response returns the reference identifier string for the specified IRIG reference instance.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0E	UINT8	Reference Identifier item
2	control	UINT16	Get
4	0	UINT32	Payload Length
8	ref ID	char[4]	Reference Identifier

Reference Identifier Error Response

The Reference Identifier Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2A	UINT8	IR Identifier
1	0x0E	UINT8	Reference Identifier item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.3 HaveQuick Reference

The messages used by the HaveQuick Reference Component (QR) are defined below.

8.3.2.3.1 Offset Transaction (QR_CA_OFFSET)

The Offset transaction is used to get or set the HaveQuick reference's 1PPS input offset. The Offset Error Response consists of a message header and error code indicating a failure of the QR to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	HaveQuick Reference Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the input offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x00	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.3.2 *Validity Transaction (QR_CA_VALIDITY)*

The Validity transaction is used to get the time and 1PPS validity structure. The Validity Error Response consists of a message header and error code indicating a failure of the QR to carry out the operation.

Validity Command

The Validity command consists of a header only.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x01	UINT8	Validity item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	HaveQuick Reference Instance

Validity Response

The Validity response returns the time and 1PPS validity structure.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x01	UINT8	Validity item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	validInfo	EL_VALIDITY	Validity structure
8	timeValid	BOOL	Time validity state
12	ppsValid	BOOL	1PPS validity state

Validity Error Response

The Validity Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x01	UINT8	Validity item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.3.3 Format Transaction (QR_CA_FORMAT)

The Format transaction is used to get the HaveQuick format. The Format Error Response consists of a message header and error code indicating a failure of the QR to carry out the operation.

Format Command

The Format command consists of a header only.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x02	UINT8	Format item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length

Format Response

The Format response returns the time and HaveQuick format.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x02	UINT8	Format item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	fmt	QL_FMT	HaveQuick format

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x02	UINT8	Format item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.3.4 Local Clock Transaction (QR_CA_LOCAL)

The Local Clock transaction is used to get or set the Local Time information. The Local Clock Error Response consists of a message header and error code indicating a failure of the QR to carry out the operation.

Local Clock Command

The Local Clock command consists of a header, and the local clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x03	UINT8	Local Clock item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	HaveQuick Reference Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Clock Response

The Local Clock get response returns the local clock structure. The Local Clock set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x03	UINT8	Local Clock item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Clock Error Response

The Local Clock Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x03	UINT8	Local Clock item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.3.5 Time Scale Transaction (QR_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the HaveQuick reference's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the QR to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the HaveQuick Reference's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x04	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	HaveQuick Reference Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the HaveQuick Reference's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x04	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x04	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.3.6 Reference Identifier Transaction (QR_CA_REF_ID)

The Reference Identifier transaction is used to retrieve the 4-character reference identifier for a specified HaveQuick reference instance.

Reference Identifier Command

The Reference Identifier Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x05	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	HaveQuick Reference Instance

Reference Identifier Response

The Reference Identifier Response returns the reference identifier string for the specified HaveQuick reference instance.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x05	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	ref ID	char[4]	Reference Identifier

Reference Identifier Error Response

The Reference Identifier Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x05	UINT8	Reference Identifier item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.3.7 Get Number of Instances Transaction (QR_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of HaveQuick References present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the QR to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x06	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of HaveQuick References present in the system.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x06	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2B	UINT8	QR Identifier
1	0x06	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4 ASCII Reference

The messages used by the ASCII Reference Component (AR) are defined below.

8.3.2.4.1 Offset Transaction (AR_CA_OFFSET)

The Offset transaction is used to get or set the ASCII reference's 1PPS input offset. The Offset Error Response consists of a message header and error code indicating a failure of the AR to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the input offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x00	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.2 *Validity Transaction (AR_CA_VALIDITY)*

The Validity transaction is used to get the time and 1PPS validity structure. The Validity Error Response consists of a message header and error code indicating a failure of the AR to carry out the operation.

Validity Command

The Validity command consists of a header only.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance

Validity Response

The Validity response returns the time and 1PPS validity structure.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	validInfo	EL_VALIDITY	Validity structure
8	timeValid	BOOL	Time validity state
12	ppsValid	BOOL	1PPS validity state

Validity Error Response

The Validity Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x02	UINT8	Validity item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.3 *UART Configuration Transaction (AR_CA_UART_CFG)*

The UART Configuration transaction is used to get or set the UART reference's configuration. The UART Configuration Error Response consists of a message header and error code indicating a failure of the AR to carry out the operation.

UART Configuration Command

The UART Configuration command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x04	UINT8	UART Configuration item
2	control	UINT16	Get or Set
4	4 (Get only) 20 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance
12	config	UD_CFG	UART Configuration structure (Set only)
12	br	UD_BR	Baud rate (Set only)
16	numbits	UD_DATA	Number Data bits (Set only)
20	stopbits	UD_STOP	Stop bits (Set only)
24	Parity	UD_PAR	Parity (Set only)

UART Configuration Response

The UART Configuration get response returns the input offset. The UART Configuration set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x04	UINT8	UART Configuration item
2	control	UINT16	Get or Set
4	0 (Set only) 16 (Get only)	UINT32	Payload Length
8	config	UD_CFG	UART Config. structure (Get only)
8	br	UD_BR	Baud rate (Get only)
12	numbits	UD_DATA	Number Data bits (Get only)
16	stopbits	UD_STOP	Stop bits (Get only)
20	Parity	UD_PAR	Parity (Get only)

UART Configuration Error Response

The UART Configuration Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x04	UINT8	UART Configuration item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.4 Leap Second Pending Transaction (AR_CA_LEAP_PENDING)

The Leap Second Pending transaction is used to retrieve the leap second pending flag associated with a specified ASCII reference instance.

Leap Second Pending Command

The Leap Second Pending Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x05	UINT8	Leap Second Pending item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance

Leap Second Pending Response

The Leap Second Pending Response returns the reference identifier string for the specified HaveQuick reference instance.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x05	UINT8	Leap Second Pending item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	leap pending	BOOL	Leap Second Pending flag

Leap Second Pending Error Response

The Leap Second Pending Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x05	UINT8	Leap Second Pending item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.5 Local Clock Transaction (AR_CA_LOCAL)

The Local Clock transaction is used to get or set the Local Time information. The Local Clock Error Response consists of a message header and error code indicating a failure of the AR to carry out the operation.

Local Clock Command

The Local Clock command consists of a header, and the local clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x06	UINT8	Local Clock item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Clock Response

The Local Clock get response returns the local clock structure. The Local Clock set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x06	UINT8	Local Clock item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Clock Error Response

The Local Clock Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x06	UINT8	Local Clock item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.6 Time Scale Transaction (AR_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the ASCII reference's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the AR to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the ASCII Reference's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x07	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the ASCII Reference's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x07	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x07	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.7 Reference Identifier Transaction (AR_CA_REF_ID)

The Reference Identifier transaction is used to retrieve the 4-character reference identifier for a specified ASCII reference instance.

Reference Identifier Command

The Reference Identifier Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x08	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance

Reference Identifier Response

The Reference Identifier Response returns the reference identifier string for the specified ASCII reference instance.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x08	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	ref ID	char[4]	Reference Identifier

Reference Identifier Error Response

The Reference Identifier Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x08	UINT8	Reference Identifier item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.8 Format Transaction (AR_CA_FORMAT)

The Format transaction is used to get or set the ASCII time code Format. The Format Error Response consists of a message header and error code indicating a failure of the AR to carry out the operation.

Format Command

The Format command consists of a header, and the format when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x09	UINT8	Format item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance
12	fmt	AL_FMT	Format (Set only)

Format Response

The Format get response returns the format. The Format set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x09	UINT8	Format item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	fmt	AL_FMT	Format (Get only)

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x09	UINT8	Format item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.9 Get Number of Instances Transaction (AR_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of ASCII References present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the AR to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of ASCII References present in the system.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2C	UINT8	AR Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.10 Host Reference

The messages used by the Host Reference Component (HR) are defined below.

8.3.2.4.11 Valid Transaction (HR_CA_VALID)

The Valid transaction is used to get or set the reference validity from the Host. A Valid response is returned on completion. The Valid Error Response consists of a message header and error code indicating a failure of the HR to carry out the operation.

Valid Command

The Valid command consists of a header and the validity structure defined as a **EL_VALIDITY**.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x00	UINT8	Valid item
2	control	UINT16	Get or Set
4	0 (Get only) 12 (Set only)	UINT32	Payload Length
8	instance	UINT32	Host Reference Instance
12	Valid	EL_VALIDITY	Reference validity structure (Set only)
12	timeValid	BOOL	Time Reference Valid (Set only)
16	ppsValid	BOOL	PPS Reference Valid (Set only)

Valid Response

The Valid set response consists of the header only. A valid get response returns the validity structure.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x00	UINT8	Valid item
2	0x0002	UINT16	Set
4	0 (Set only) 12 (Get only)	UINT32	Payload Length
8	instance	UINT32	Host Reference Instance
12	Valid	EL_VALIDITY	Reference validity structure (Get only)
12	timeValid	BOOL	Time Reference Valid (Get only)
16	ppsValid	BOOL	PPS Reference Valid (Get only)

Valid Error Response

The Valid Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x00	UINT8	Valid item
2	0x0003	UINT16	Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.12 Time Transaction (HR_CA_TIME)

The Time transaction is used to set the clock's specified Host Reference time value. Get is not supported. The Time Error Response consists of a message header and error code indicating a failure of the CS to carry out the operation.

Time Command

The Leap Second set command consists of a header and the current time defined as a **ML_DOYTIME**.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x01	UINT8	Time item
2	control	UINT16	Set only
4	28 (Set only)	UINT32	Payload Length
8	instance	UINT32	Host Reference Instance
12	utcDate	ML_DOYTIME	Time structure (Set only)
12	year	UINT32	Year (Set only)
16	doy	UINT32	Day of year (Set only)
20	hour	UINT32	Hour (Set only)
24	minute	UINT32	Minute (Set only)
28	second	UINT32	Second (Set only)
32	ns	UINT32	Nanosecond (Set only)

Time Error Response

The Time Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x01	UINT8	Leap Second item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

Time Response

The Time set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x01	UINT8	Time item
2	0x0002	UINT16	Set
4	0 (Set only)	UINT32	Payload Length

8.3.2.4.13 Local Clock Transaction (HR_CA_LOCAL)

The Local Clock transaction is used to get or set the Local Time information. The Local Clock Error Response consists of a message header and error code indicating a failure of the HR to carry out the operation.

Local Clock Command

The Local Clock command consists of a header, and the local clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x02	UINT8	Local Clock item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Clock Response

The Local Clock get response returns the local clock structure. The Local Clock set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x02	UINT8	Local Clock item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Clock Error Response

The Local Clock Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x02	UINT8	Local Clock item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.14 Time Scale Transaction (HR_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the Host reference's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the HR to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the Host Reference's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x03	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the Host Reference's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x03	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x03	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.15 Reference Identifier Transaction (HR_CA_REF_ID)

The Reference Identifier transaction is used to retrieve the 4-character reference identifier for a specified Host reference instance.

Reference Identifier Command

The Reference Identifier Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	ASCII Reference Instance

Reference Identifier Response

The Reference Identifier Response returns the reference identifier string for the specified ASCII reference instance.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	ref ID	char[4]	Reference Identifier

Reference Identifier Error Response

The Reference Identifier Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.4.16 Get Number of Instances Transaction (HR_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of Host References present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the HR to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x05	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of Host References present in the system.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x05	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2D	UINT8	HR Identifier
1	0x05	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.5 PPS Reference

The messages used by the PPS Reference Component (PR) are defined below.

8.3.2.5.1 Offset Transaction (PR_CA_OFFSET)

The Offset transaction is used to get or set the 1PPS reference's input offset. The Offset Error Response consists of a message header and error code indicating a failure of the PR to carry out the operation.

Offset Command

The Offset command consists of a header, an instance number, and an offset when set and a header and an instance number with get.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	PPS Reference Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the input offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x00	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x00	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.5.2 Edge Transaction (PR_CA_EDGE)

The Edge transaction is used to get or set the 1PPS reference's active edge setting. The Edge Error Response consists of a message header and error code indicating a failure of the PR to carry out the operation.

Edge Command

The Edge command consists of a header, an instance number, and an active edge setting when set and a header and an instance number with get.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x01	UINT8	Edge item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	PPS Reference Instance
12	edge	EDGE	Active Edge (Set only)

Edge Response

The Edge get response returns the active edge setting. The Edge set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x01	UINT8	Edge item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	edge	EDGE	Active Edge (Get only)

Edge Error Response

The Edge Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x01	UINT8	Edge item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.5.3 Validity Transaction (PR_CA_VALIDITY)

The Validity transaction is used to get the time and 1PPS validity structure. The Validity Error Response consists of a message header and error code indicating a failure of the PR to carry out the operation.

Validity Command

The Validity command consists of a header and instance number.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	PPS Reference Instance

Validity Response

The Validity response returns the time and 1PPS validity structure.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	validInfo	EL_VALIDITY	Validity structure
8	timeValid	BOOL	Time validity state
12	ppsValid	BOOL	1PPS validity state

Validity Error Response

The Validity Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x02	UINT8	Validity item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.5.4 Number of Instances Transaction (PR_CA_NUM_INST)

The Number of Instances transaction is used to get the number of PPS Reference instances in the system. The Number of Instances Error Response consists of a message header and error code indicating a failure of the PR to carry out the operation.

Number of Instances Command

The Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x03	UINT8	Number of Instances item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Number of Instances Response

The Number of Instances response returns the number of PPS Reference instances.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x03	UINT8	Number of Instances item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	numInst	UINT32	Number of Instances

Number of Instances Error Response

The Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x03	UINT8	Number of Instances item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.5.5 Reference Identifier Transaction (PR_CA_REF_ID)

The Reference Identifier transaction is used to retrieve the 4-character reference identifier for a specified PPS reference instance.

Reference Identifier Command

The Reference Identifier Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	PPS Reference Instance

Reference Identifier Response

The Reference Identifier Response returns the reference identifier string for the specified PPS reference instance.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	ref ID	char[4]	Reference Identifier

Reference Identifier Error Response

The Reference Identifier Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2E	UINT8	PR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.6 Frequency Reference

The messages used by the Frequency Reference Component (FR) are defined below.

8.3.2.6.1 Frequency Transaction (FR_CA_FREQUENCY)

The Frequency transaction is used to get or set the Frequency reference's frequency setting. The Edge Error Response consists of a message header and error code indicating a failure of the FR to carry out the operation.

Frequency Command

The Frequency command consists of a header, an instance number, and a frequency setting when set and a header and an instance number with get.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x01	UINT8	Frequency item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	Frequency Reference Instance
12	frequency	UINT32	Frequency (Set only)

Frequency Response

The Frequency get response returns the active edge setting. The Frequency set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x01	UINT8	Frequency item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	frequency	UINT32	Frequency (Set only)

Frequency Error Response

The Frequency Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x01	UINT8	Frequency item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.6.2 *Validity Transaction (FR_CA_VALIDITY)*

The Validity transaction is used to get the time and 1PPS validity structure. The Validity Error Response consists of a message header and error code indicating a failure of the FR to carry out the operation.

Validity Command

The Validity command consists of a header and instance number.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	Frequency Reference Instance

Validity Response

The Validity response returns the time and 1PPS validity structure.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x02	UINT8	Validity item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	validInfo	EL_VALIDITY	Validity structure
8	timeValid	BOOL	Time validity state
12	ppsValid	BOOL	1PPS validity state

Validity Error Response

The Validity Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x02	UINT8	Validity item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.6.3 Number of Instances Transaction (FR_CA_NUM_INST)

The Number of Instances transaction is used to get the number of Frequency Reference instances in the system. The Number of Instances Error Response consists of a message header and error code indicating a failure of the FR to carry out the operation.

Number of Instances Command

The Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x03	UINT8	Number of Instances item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Number of Instances Response

The Number of Instances response returns the number of Frequency Reference instances.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x03	UINT8	Number of Instances item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	numInst	UINT32	Number of Instances

Number of Instances Error Response

The Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x03	UINT8	Number of Instances item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.6.4 Reference Identifier Transaction (FR_CA_REF_ID)

The Reference Identifier transaction is used to retrieve the 4-character reference identifier for a specified Frequency reference instance.

Reference Identifier Command

The Reference Identifier Command consists of a header and detail.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	PPS Reference Instance

Reference Identifier Response

The Reference Identifier Response returns the reference identifier string for the specified frequency reference instance.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	ref ID	char[4]	Reference Identifier

Reference Identifier Error Response

The Reference Identifier Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3A	UINT8	FR Identifier
1	0x04	UINT8	Reference Identifier item
2	control	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7 IRIG Output

The messages used by the IRIG Output Component (IP) are defined below.

8.3.2.7.1 Signature Control Transaction (IP_CA_SIG_CTL)

The Signature Control transaction is used to get or set the IRIG output's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Signature Control Command

The Signature Control command consists of a header and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x00	UINT8	Signature Control item
2	**control|0x0001**	**UINT16**	**Get or Set and Error Response**
4	**4 (or 8)**	**UINT32**	**Payload Length**
8	**ec**	**HA_RC**	**HA Error Code**
12	**rc**	**RC**	**Optional operation specific return code**

8.3.2.7.2 Offset Transaction (IP_CA_OFFSET)

The Offset transaction is used to get or set the IRIG output's output offset. The Offset Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x01	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the output offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x01	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x01	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.3 Local Transaction (IP_CA_LOCAL)

The Local transaction is used to get or set the IRIG output's Local Clock including the time zone and DST rule. The Local Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Local Command

The Local command consists of a header, and a Local Clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x02	UINT8	Local item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Response

The Local get response returns the Local Clock structure. The Local set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x02	UINT8	Local item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Error Response

The Local Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x02	UINT8	Local item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.4 Format Transaction (IP_CA_FORMAT)

The Format transaction is used to get or set the IRIG output's Format. The Format Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Format Command

The Format command consists of a header, and the format when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x03	UINT8	Format item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	Fmt	IL_FMT	Format (Set only)

Format Response

The Format get response returns the format. The Format set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x03	UINT8	Format item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	Fmt	IL_FMT	Format (Get only)

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x03	UINT8	Format item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.5 Amplitude Transaction (IP_CA_AMPLITUDE)

The Amplitude transaction is used to get or set the IRIG output's Amplitude. The Amplitude Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Amplitude Command

The Amplitude command consists of a header, and the amplitude when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x04	UINT8	Amplitude item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	amp	UINT32	Amplitude (Set only)

Amplitude Response

The Amplitude get response returns the amplitude. The Amplitude set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x04	UINT8	Amplitude item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	amp	UINT32	Amplitude (Get only)

Amplitude Error Response

The Amplitude Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x04	UINT8	Amplitude item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.6 Modulation Transaction (IP_CA_MOD)

The Modulation transaction is used to get the IRIG output's Modulation. The Modulation Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Modulation Command

The Modulation command consists of a header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x05	UINT8	Modulation item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance

Modulation Response

The Modulation get response returns the modulation.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x05	UINT8	Modulation item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	mod	IL_MOD	Modulation

Modulation Error Response

The Modulation Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x05	UINT8	Modulation item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.7 Frequency Transaction (IP_CA_FREQ)

The Frequency transaction is used to get or set the IRIG output's carrier frequency. The Frequency Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Frequency Command

The Frequency command consists of a header, and the IRIG carrier frequency when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x06	UINT8	Frequency item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	freq	IL_FRQ	Frequency (Set only)

Frequency Response

The Frequency get response returns the IRIG carrier frequency. The Frequency set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x06	UINT8	Frequency item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	freq	IL_FRQ	Frequency (Get only)

Frequency Error Response

The Frequency Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x06	UINT8	Frequency item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.8 Coded Expression Transaction (IP_CA_CODED_EXP)

The Coded Expression transaction is used to get or set the IRIG output's Coded Expression. The Coded Expression Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Coded Expression Command

The Coded Expression command consists of a header, and the IRIG Coded Expression when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x07	UINT8	Coded Expression item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	ce	IL_CE	Coded Expression (Set only)

Coded Expression Response

The Coded Expression get response returns the IRIG Coded Expression. The Coded Expression set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x07	UINT8	Coded Expression item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ce	IL_CE	Coded Expression (Get only)

Coded Expression Error Response

The Coded Expression Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x07	UINT8	Coded Expression item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.9 Control Field Transaction (IP_CA_CTRL_FLD)

The Control Field transaction is used to get or set the IRIG output's Control Field. The Control Field Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Control Field Command

The Control Field command consists of a header, and the IRIG Control Field when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x08	UINT8	Control Field item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	cf	IL_CF	Control Field (Set only)

Control Field Response

The Control Field get response returns the IRIG Control Field. The Control Field set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	Control Field item
1	0x08	UINT8	Control Field item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	cf	IL_CF	Control Field (Get only)

Control Field Error Response

The Control Field Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	Control Field item
1	0x08	UINT8	Control Field item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.10 Message Transaction (IP_CA_MESSAGE)

The Message transaction is used to get and set the latest IRIG input message. The Message Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation. The get message is enabled by default. The set message is enabled as part of the product configuration.

Message Command

The Message command consists of a header only for get and the header and the entire sub-frame to send for set.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x09	UINT8	Message item
2	control	UINT16	Get or Set
4	4 (Get only) 24 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	frame	IP_MSG	IRIG Frame structure (Set only)
12	sub1	UINT16	Subframe 1 (Set only)
14	sub2	UINT16	Subframe 2 (Set only)
16	sub3	UINT16	Subframe 3 (Set only)
18	sub4	UINT16	Subframe 4 (Set only)
20	sub5	UINT16	Subframe 5 (Set only)
22	sub6	UINT16	Subframe 6 (Set only)
24	sub7	UINT16	Subframe 7 (Set only)
26	sub8	UINT16	Subframe 8 (Set only)
28	sub9	UINT16	Subframe 9 (Set only)
30	sub10	UINT16	Subframe 10 (Set only)

Message Response

The Message response returns the time and last IRIG output message on get. The Message response consists of a header only on set.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x09	UINT8	Message item
2	control	UINT16	Get or Set
4	0 (Set only) 20 (Get only)	UINT32	Payload Length
8	frame	IP_MSG	IRIG Frame structure (Get only)
8	sub1	UINT16	Subframe 1 (Get only)
10	sub2	UINT16	Subframe 2 (Get only)
12	sub3	UINT16	Subframe 3 (Get only)
14	sub4	UINT16	Subframe 4 (Get only)
16	sub5	UINT16	Subframe 5 (Get only)
18	sub6	UINT16	Subframe 6 (Get only)
20	sub7	UINT16	Subframe 7 (Get only)
22	sub8	UINT16	Subframe 8 (Get only)
24	sub9	UINT16	Subframe 9 (Get only)
26	sub10	UINT16	Subframe 10 (Get only)

Message Error Response

The Message Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x09	UINT8	Message item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.11 Control Field Data Transaction (IP_CA_CFDATA)

The Control Field Data transaction is used to get or set the current control field data. The Control Field Data Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Control Field Data Command

The Control Field Data command consists of a header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0A	UINT8	Control Field Data item
2	control	UINT16	Get or Set
4	4 (Get only) 10 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	cfData	UINT16[3]	Control Field Data

Control Field Data Response

The Control Field Data response consists of a header and the current control field data.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0A	UINT8	Control Field Data item
2	control	UINT16	Get or Set
4	0 (Set only) 6 (Get only)	UINT32	Payload Length
8	cfData	UINT16[3]	Control Field Data

Control Field Data Error Response

The Control Field Data Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0A	UINT8	Control Field Data item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.12 Number of Instances Transaction (IP_CA_NUM_INST)

The Number of Instances transaction is used to get the number of IRIG Output instances in the system. The Number of Instances Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Number of Instances Command

The Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0B	UINT8	Number of Instances item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Number of Instances Response

The Number of Instances response returns the number of IRIG Output instances.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0B	UINT8	Number of Instances item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	numInst	UINT32	Number of Instances

Number of Instances Error Response

The Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0B	UINT8	Number of Instances item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.13 Phase Transaction (IP_CA_PHASE)

The Phase transaction is used to get or set the IRIG Output's phase adjustment. The Phase Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation. This feature is used only by Safran for product testing.

Phase Command

The Phase command consists of a header, and the IRIG Output's Phase value when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0C	UINT8	Phase item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	phase	UINT32	Phase (Set only)

Phase Response

The Phase get response returns the IRIG Output's phase value. The Phase set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0C	UINT8	Phase item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	phase	UINT32	Phase (Get only)

Phase Error Response

The Phase Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0C	UINT8	Phase item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.14 Phase Error Transaction (IP_CA_PHASE_ERR)

The Phase Error transaction is used to get the IRIG Output's current phase error. The Phase Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation. This feature is used only by Safran for product testing.

Phase Error Command

The Phase Error command consists of a header, and the IRIG Output's instance value when get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0D	UINT8	Phase Error item
2	control	UINT16	Get or Set
4	4 (Get only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance

Phase Error Response

The Phase Error get response returns the IRIG Output's phase value.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0D	UINT8	Phase Error item
2	control	UINT16	Get or Set
4	4 (Get only)	UINT32	Payload Length
8	phErr	INT32	Phase Error (Get only)

Phase Error Error Response

The Phase Error Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0D	UINT8	Phase item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.7.15 Time Scale Transaction (IP_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the IRIG Output's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the IP to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the IRIG Output's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0E	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the IRIG Output's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0E	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x2F	UINT8	IP Identifier
1	0x0E	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.8 HaveQuick Output

The messages used by the HaveQuick Output Component (QP) are defined below.

8.3.2.8.1 Signature Control Transaction (QP_CA_SIG_CTL)

The Signature Control transaction is used to get or set the HaveQuick output's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the QP to carry out the operation.

Signature Control Command

The Signature Control command consists of a header and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	HaveQuick Output Instance
12	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x00	UINT8	Signature Control item
2	**control|0x0001**	**UINT16**	**Get or Set and Error Response**
4	**4 (or 8)**	**UINT32**	**Payload Length**
8	**ec**	**HA_RC**	**HA Error Code**
12	**rc**	**RC**	**Optional operation specific return code**

8.3.2.8.2 Offset Transaction (QP_CA_OFFSET)

The Offset transaction is used to get or set the HaveQuick reference's output offset. The Offset Error Response consists of a message header and error code indicating a failure of the QP to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x01	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	HaveQuick Output Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the output offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x01	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x01	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.8.3 Local Transaction (QP_CA_LOCAL)

The Local transaction is used to get or set the HaveQuick reference's Local Clock including the time zone and DST rule. The Local Error Response consists of a message header and error code indicating a failure of the QP to carry out the operation.

Local Command

The Local command consists of a header, and a Local Clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x02	UINT8	Local item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	HaveQuick Output Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Response

The Local get response returns the Local Clock structure. The Local set response consists of the header only.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x02	UINT8	Local item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Error Response

The Local Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x02	UINT8	Local item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.8.4 Format Transaction (QP_CA_FORMAT)

The Format transaction is used to get or set the HaveQuick output's format state. The Format Error Response consists of a message header and error code indicating a failure of the QP to carry out the operation.

Format Command

The Format command consists of a header and a format state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x03	UINT8	Format item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	HaveQuick Output Instance
12	fmt	QL_FMT	Format (Set only)

Format Response

The Format Command get response returns the format state. The Format set response consists of the header only.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x03	UINT8	Format item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	fmt	QL_FMT	Format (Get only)

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x03	UINT8	Format item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.8.5 Time Scale Transaction (QP_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the HaveQuick Output's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the QP to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the HaveQuick Output's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x04	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	HaveQuick Output Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the HaveQuick Output's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x04	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x04	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.8.6 Get Number of Instances Transaction (QP_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of HaveQuick Output's present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the QP to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x05	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of HaveQuick Outputs present in the system.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x05	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x30	UINT8	QP Identifier
1	0x05	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.9 Fixed Frequency Output

The messages used by the Fixed Frequency Output Component (FP) are defined below.

8.3.2.9.1 Signature Control Transaction (FP_CA_SIG_CTL)

The Signature Control transaction is used to get or set the Fixed Frequency output's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the FP to carry out the operation.

Signature Control Command

The Signature Control command consists of a header and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	Frequency Output Instance
12	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x00	UINT8	Signature Control item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.9.2 Frequency Transaction (FP_CA_FREQ)

The Frequency transaction is used to get the Fixed Frequency output's frequency. The Frequency Error Response consists of a message header and error code indicating a failure of the FP to carry out the operation.

Frequency Command

The Frequency command consists of a header only with get.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x01	UINT8	Frequency item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	Frequency Output Instance

Frequency Response

The Frequency Command get response returns the frequency.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x01	UINT8	Frequency item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	frq	FLT32	Frequency

Frequency Error Response

The Frequency Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x01	UINT8	Frequency item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.9.3 Get Number of Instances Transaction (FP_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of Fixed Frequency Outputs present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the FP to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x02	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of Fixed Frequency Outputs present in the system.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x02	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	4	UINT32	Payload Length
8	Num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x31	UINT8	FP Identifier
1	0x02	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.10 Display Output

The messages used by the Display Output Component (DP) are defined below.

8.3.2.10.1 Local Transaction (DP_CA_LOCAL)

The Local transaction is used to get or set the Display's Local Clock including the time zone and DST rule. The Local Error Response consists of a message header and error code indicating a failure of the DP to carry out the operation.

Local Command

The Local command consists of a header, and a Local Clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x00	UINT8	Local item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	Display Output Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Response

The Local get response returns the Local Clock structure. The Local set response consists of the header only.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x00	UINT8	Local item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Error Response

The Local Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x00	UINT8	Local item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.10.2 Format Transaction (DP_CA_FORMAT)

The Format transaction is used to get or set the Display's time format state. The Format Error Response consists of a message header and error code indicating a failure of the DP to carry out the operation.

Format Command

The Format command consists of a header and a format state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x01	UINT8	Format item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	Display Output Instance
12	fmt	ML_HOUR	Format (Set only)

Format Response

The Format Command get response returns the format state. The Format set response consists of the header only.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x01	UINT8	Format item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	fmt	ML_HOUR	Format (Get only)

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x01	UINT8	Format item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.10.3 Time Scale Transaction (DP_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the Display Output's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the DP to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the Display Output's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x02	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	Display Output Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the Display Output's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x02	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x02	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.10.4 Get Number of Instances Transaction (DP_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of Display Output's present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the DP to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x03	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of Display Outputs present in the system.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x03	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x03	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.10.5 Mode Transaction (DP_CA_MODE)

The Mode transaction is used to get or set the Display Output's mode. The Mode Error Response consists of a message header and error code indicating a failure of the DP to carry out the operation.

Mode Command

The Mode command consists of a header, and the Display Output's Mode when set and a header only with get.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x04	UINT8	Mode item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	Display Output Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Mode Response

The Mode get response returns the Display Output's Mode. The Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x04	UINT8	Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	mode	DP_MODE	Display Mode (Get only)

Mode Error Response

The Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x32	UINT8	DP Identifier
1	0x04	UINT8	Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.11 ASCII Output

The messages used by the ASCII Output Component (AP) are defined below.

8.3.2.11.1 Signature Control Transaction (AP_CA_SIG_CTL)

The Signature Control transaction is used to get or set the ASCII output's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the AP to carry out the operation.

Signature Control Command

The Signature Control command consists of a header and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Output Instance
8	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x00	UINT8	Signature Control item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.11.2 Local Transaction (AP_CA_LOCAL)

The Local transaction is used to get or set the ASCII reference's Local Clock including the time zone and DST rule. The Local Error Response consists of a message header and error code indicating a failure of the AP to carry out the operation.

Local Command

The Local command consists of a header, and a Local Clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x02	UINT8	Local item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Output Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Response

The Local get response returns the Local Clock structure. The Local set response consists of the header only.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x02	UINT8	Local item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Error Response

The Local Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x02	UINT8	Local item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.11.3 Time Scale Transaction (AP_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the ASCII Output's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the AP to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the ASCII Output's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x03	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Output Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the ASCII Output's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x03	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x03	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.11.4 Format Transaction (AP_CA_FORMAT)

The Format transaction is used to get or set the ASCII reference's format state. The Format Error Response consists of a message header and error code indicating a failure of the AP to carry out the operation.

Format Command

The Format command consists of a header and a format state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x04	UINT8	Format item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Output Instance
12	fmt	AL_FMT	Format (Set only)

Format Response

The Format Command get response returns the format state. The Format set response consists of the header only.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x04	UINT8	Format item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	fmt	AL_FMT	Format (Get only)

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x04	UINT8	Format item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.11.5 Mode Transaction (AP_CA_MODE)

The Mode transaction is used to get or set the ASCII references transmit mode used when outputting messages which controls how and when the ASCII output messages are transmitted. The Mode Error Response consists of a message header and error code indicating a failure of the AP to carry out the operation.

Mode Command

The Mode command consists of a header, and a output mode state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x06	UINT8	Mode item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Output Instance
12	mode	AL_OUT_MODE	Mode(Set only)

Mode Response

The Mode Command get response returns the output mode setting. The Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x06	UINT8	Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	mode	AL_OUT_MODE	Mode(Get only)

Mode Error Response

The Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x06	UINT8	Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.11.6 Request Transaction (AP_CA_REQ_CHAR)

The Request transaction is used to get or set the ASCII reference's request character. The Request Error Response consists of a message header and error code indicating a failure of the AP to carry out the operation.

Request Command

The Request Command consists of a header, and the request character when set and a header only with get.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x08	UINT8	Request Character item
2	control	UINT16	Get or Set
4	4 (Get only) 5 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Output Instance
12	req	char	Request Character(Set only)

Request Response

The Request Command get response returns the request character setting. The Request Command set response consists of the header only.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x08	UINT8	Request Character item
2	control	UINT16	Get or Set
4	0 (Set only) 1 (Get only)	UINT32	Payload Length
8	req	char	Request Character(Get only)

Request Error Response

The Request Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x08	UINT8	Request Character item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.11.7 UART Configuration Transaction (AP_CA_UART_CFG)

The UART Configuration transaction is used to get or set the UART reference's configuration. The UART Configuration Error Response consists of a message header and error code indicating a failure of the AP to carry out the operation.

UART Configuration Command

The UART Configuration command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x09	UINT8	UART Configuration item
2	control	UINT16	Get or Set
4	4 (Get only) 20 (Set only)	UINT32	Payload Length
8	instance	UINT32	ASCII Output Instance
12	config	UD_CFG	UART Configuration structure (Set only)
12	br	UD_BR	Baud rate (Set only)
16	numbits	UD_DATA	Number Data bits (Set only)
20	stopbits	UD_STOP	Stop bits (Set only)
24	Parity	UD_PAR	Parity (Set only)

UART Configuration Response

The UART Configuration get response returns the input offset. The UART Configuration set response consists of the header only.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x09	UINT8	UART Configuration item
2	control	UINT16	Get or Set
4	0 (Set only) 16 (Get only)	UINT32	Payload Length
8	config	UD_CFG	UART Configuration structure (Get only)
8	br	UD_BR	Baud rate (Get only)
12	numbits	UD_DATA	Number Data bits (Get only)
16	stopbits	UD_STOP	Stop bits (Get only)
20	Parity	UD_PAR	Parity (Get only)

UART Configuration Error Response

The UART Configuration Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x09	UINT8	UART Configuration item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.11.8 Get Number of Instances Transaction (AP_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of ASCII Outputs present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the AP to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of ASCII Outputs present in the system.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x33	UINT8	AP Identifier
1	0x0A	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.12 Variable Frequency Output

The messages used by the Variable-Frequency Output Component (VP) are defined below.

8.3.2.12.1 Signature Control Transaction (VP_CA_SIG_CTL)

The Signature Control transaction is used to get or set the Variable-Frequency output's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the VP to carry out the operation.

Signature Control Command

The Signature Control command consists of a header and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	Variable-Frequency Output Instance
12	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x00	UINT8	Signature Control item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.12.2 Frequency Transaction (VP_CA_FREQ)

The Frequency transaction is used to get or set the Variable-Frequency output's frequency value. The Frequency Error Response consists of a message header and error code indicating a failure of the VP to carry out the operation.

Frequency Command

The Frequency command consists of a header and a frequency when set and a header only with get.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x01	UINT8	Frequency item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	Variable-Frequency Output Instance
12	frq	UINT32	Frequency (Set only)

Frequency Response

The Frequency Command get response returns the frequency. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x01	UINT8	Frequency item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	frq	UINT32	Frequency (Get only)

Frequency Error Response

The Frequency Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x01	UINT8	Frequency item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.12.3 Range Transaction (VP_CA_RANGE)

The Range transaction is used to get the Variable-Frequency output's range. The Range Error Response consists of a message header and error code indicating a failure of the VP to carry out the operation.

Range Command

The Range command consists of a header only with get.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x02	UINT8	Range item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	instance	UINT32	Variable-Frequency Output Instance

Range Response

The Frequency Command get response returns the frequency range structure.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x02	UINT8	Range item
2	0x0000	UINT16	Get
4	12	UINT32	Payload Length
8	Config	VP_CFG	Configuration structure
8	min	FLT32	Minimum Frequency
12	max	FLT32	Maximum Frequency
16	step	FLT32	Minimum step size

Range Error Response

The Frequency Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x02	UINT8	Range item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.12.4 Get Number of Instances Transaction (VP_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of Variable-Frequency Outputs present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the VP to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x03	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of Variable-Frequency Outputs present in the system.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x03	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x34	UINT8	VP Identifier
1	0x03	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.13 PPS Output

The messages used by the PPS Output Component (PP) are defined below.

8.3.2.13.1 Signature Control Transaction (PP_CA_SIG_CTL)

The Signature Control transaction is used to get or set the 1PPS Output's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the PP to carry out the operation.

Signature Control Command

The Signature Control command consists of a header and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	PPS Output Instance
12	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x00	UINT8	Signature Control item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.13.2 Frequency Transaction (PP_CA_FREQ)

The Frequency transaction is used to get the 1PPS Output's frequency. The Frequency Error Response consists of a message header and error code indicating a failure of the PP to carry out the operation.

Frequency Command

The Frequency command consists of a header only with get.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x01	UINT8	Frequency item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	inst	UINT32	PPS Output Instance

Frequency Response

The Frequency Command get response returns the frequency.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x01	UINT8	Frequency item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	frq	FLT32	Frequency (constant 1PPS)

Frequency Error Response

The Frequency Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x01	UINT8	Frequency item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.13.3 Offset Transaction (PP_CA_OFFSET)

The Offset transaction is used to get or set the 1PPS Output's offset. The Offset Error Response consists of a message header and error code indicating a failure of the PP to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x02	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	PPS Output Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the output offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x02	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x02	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.13.4 Edge Transaction (PP_CA_EDGE)

The Edge transaction is used to get or set the PPS Output's active edge. The Edge Error Response consists of a message header and error code indicating a failure of the PP to carry out the operation.

Edge Command

The Edge command consists of a header, and the PPS output's active edge when set and a header only with get.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x03	UINT8	Edge item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	PPS Output Instance
12	edge	EDGE	Active Edge (Set only)

Edge Response

The Edge command get response returns the PPS output's active edge setting. The Edge set response consists of the header only.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x03	UINT8	Edge item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	edge	EDGE	Active Edge (Get only)

Edge Error Response

The Edge Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x03	UINT8	Edge item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.13.5 Pulse Width Transaction (PP_CA_PW)

The Pulse Width transaction is used to get or set the PPS Output's active pulse width. The Pulse Width Error Response consists of a message header and error code indicating a failure of the PP to carry out the operation.

Pulse Width Command

The Pulse Width command consists of a header, and the PPS output's active pulse width when set and a header only with get.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x04	UINT8	Pulse Width item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	PPS Output Instance
12	pw	UINT32	Active Pulse Width (Set only)

Pulse Width Response

The Pulse Width command get response returns the PPS output's active pulse width setting. The Pulse Width set response consists of the header only.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x04	UINT8	Pulse Width item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	pw	UINT32	Active Pulse Width (Get only)

Pulse Width Error Response

The Pulse Width Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x04	UINT8	Pulse Width item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.13.6 Get Number of Instances Transaction (PP_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of PPS Outputs present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the PP to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x05	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of PPS Outputs present in the system.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x05	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x35	UINT8	PP Identifier
1	0x05	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.14 E1/T1 Output

The messages used by the E1/T1 Output Component (ETP) are defined below.

8.3.2.14.1 Signature Control Transaction (ETP_CA_SIG_CTL)

The Signature Control transaction is used to get or set the E1/T1 Output's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the ETP to carry out the operation.

Signature Control Command

The Signature Control command consists of a header and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	E1/T1 Output Instance
12	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x00	UINT8	Signature Control item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.14.2 Offset Transaction (ETP_CA_OFFSET)

The Offset transaction is used to get or set the E1/T1 Output's offset. The Offset Error Response consists of a message header and error code indicating a failure of the ETP to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x01	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	E1/T1 Output Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the output offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x01	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x01	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.14.3 Mode Transaction (ETP_CA_MODE)

The Mode transaction is used to get or set the E1/T1 Output's mode (E1 or T1). The Mode Error Response consists of a message header and error code indicating a failure of the ETP to carry out the operation.

Mode Command

The Mode command consists of a header, and an output mode state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x02	UINT8	Mode item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	E1/T1 Output Instance
12	mode	ETP_MODE	Mode(Set only)

Mode Response

The Mode Command get response returns the mode setting. The Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x02	UINT8	Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	mode	ETP_MODE	Mode(Get only)

Mode Error Response

The Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x02	UINT8	Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.14.4 Format Transaction (ETP_CA_FORMAT)

The Format transaction is used to get or set the E1/T1 Output's format state. The Format Error Response consists of a message header and error code indicating a failure of the ETP to carry out the operation.

Format Command

The Format command consists of a header and a format state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x03	UINT8	Format item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	E1/T1 Output Instance
12	fmt	ETP_FMT	Format (Set only)

Format Response

The Format Command get response returns the format state. The Format set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x03	UINT8	Format item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	fmt	ETP_FMT	Format (Get only)

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x03	UINT8	Format item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.14.5 Get Number of Instances Transaction (ETP_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of E1/T1 Outputs present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the ETP to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x04	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of E1/T1 Outputs present in the system.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x04	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3D	UINT8	ETP Identifier
1	0x04	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15 Oscillator

The messages used by the Oscillator Component (XO) are defined below.

8.3.2.15.1 Disciplining State Transaction (XO_CA_DISC_STATE)

The Disciplining State transaction is used to get the external oscillator's Disciplining State. The Disciplining State Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Disciplining State Command

The Disciplining State command consists of a header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x00	UINT8	Disciplining State item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Disciplining State Response

The Disciplining State response returns the receiver alarm.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x00	UINT8	Disciplining State item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	disc	INT32	Disciplining State

Disciplining State Error Response

The Disciplining State Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x00	UINT8	Disciplining State item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.2 Mode Transaction (XO_CA_MODE)

The Mode transaction is used to get or set the external oscillator's mode used when disciplining or testing. The Mode Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Mode Command

The Mode command consists of a header, and the oscillator disciplining mode when set and a header only with get.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x01	UINT8	Mode item
2	control	UINT16	Get or Set
4	0 (Get only) 4 (Set only)	UINT32	Payload Length
8	mode	XO_MODE	Mode (Set only)

Mode Response

The Mode Command get response returns the disciplining mode setting. The Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x01	UINT8	Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	mode	XO_MODE	Mode (Get only)

Mode Error Response

The Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x01	UINT8	Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.3 DAC Transaction (XO_CA_DAC)

The DAC transaction is used to get or set the external oscillator's DAC for testing. The DAC Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

DAC Command

The DAC command consists of a header, and the external oscillator's DAC value when set and a header only with get.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x02	UINT8	DAC item
2	control	UINT16	Get or Set
4	0 (Get only) 2 (Set only)	UINT32	Payload Length
8	val	UINT16	DAC value (Set only)

DAC Response

The DAC Command get response returns the current DAC setting. The Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x02	UINT8	DAC item
2	control	UINT16	Get or Set
4	0 (Set only) 2 (Get only)	UINT32	Payload Length
8	val	UINT16	DAC value (Get only)

DAC Error Response

The DAC Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x02	UINT8	DAC item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.4 Alarm Transaction (XO_CA_ALARM)

The Alarm transaction is used to get the external oscillator's Alarm state. The Alarm Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Alarm Command

The Alarm command consists of a header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x03	UINT8	Alarm State item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Alarm Response

The Alarm response returns the external oscillator alarm.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x03	UINT8	Alarm State item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	alarm	UINT32	Alarm State

Alarm Error Response

The Alarm Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x03	UINT8	Alarm State item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.5 Serial Number Transaction (XO_CA_SERIAL_NUM)

The Serial Number transaction is used to get the external oscillator's serial number. The Serial Number Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Serial Number Command

The Serial Number command consists of a header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x04	UINT8	Serial Number item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Serial Number Response

The Serial Number response returns the oscillator serial number.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x04	UINT8	Serial Number item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	sernum	char[32]	Serial Number

Serial Number Error Response

The Serial Number Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x04	UINT8	Serial Number item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.6 Manufacturer Model Transaction (XO_CA_MFR_MDL)

The Manufacturer Model transaction is used to get the external oscillator's manufacturer and model. The Manufacturer Model Error Response consists of a message header and error code indicating a failure of the GR to carry out the operation.

Manufacturer Model Command

The Manufacturer Model command consists of a header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x05	UINT8	Manufacturer Model item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Manufacturer Model Response

The Manufacturer Model response returns the oscillator manufacturer and model.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x05	UINT8	Manufacturer Model item
2	0x0000	UINT16	Get
4	32	UINT32	Payload Length
8	mfrmdl	XO_MFR_MDL	Manufacturer/Model structure
8	mfr	char[16]	Manufacturer string
24	mdl	char[16]	Model string

Manufacturer Model Error Response

The Manufacturer Model Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x05	UINT8	Manufacturer Model item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.7 Custom Message Transaction (XO_CA_CUSTOM_MSG)

The Custom Message transaction is used to get a custom message response from the external oscillator or send a Custom Message to the external oscillator. The Custom Message Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Custom Message Command

The Custom Message command consists of a header, and a custom message when set and a header only with get.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x06	UINT8	Custom Message item
2	control	UINT16	Get or Set
4	0 (Get only) 260 (Set only)	UINT32	Payload Length
8	msgInfo	XO_CUSTOM_MSG	Custom message struct (Set only)
8	len	UINT32	Length (Set only)
12	msg	char[256]	Custom Message (Set only)

Custom Message Response

The Custom Message get response returns the Custom Message response. The Custom Message set response consists of the header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x06	UINT8	Custom Message item
2	control	UINT16	Get or Set
4	0 (Set only) 260 (Get only)	UINT32	Payload Length
8	msgInfo	XO_CUSTOM_MSG	Custom message struct (Get only)
8	len	UINT32	Length (Get only)
12	resp	char[256]	Custom Message response (Get only)

Custom Message Error Response

The Custom Message Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x06	UINT8	Custom Message item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.8 Disciplining Transaction (XO_CA_DISC)

The Disciplining transaction is used to get a disciplining dataset from the XO or send a disciplining command and dataset to the XO. The Disciplining Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Disciplining Command

The Disciplining command consists of a header, and a command and dataset when set and a header and command with get.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x07	UINT8	Disciplining item
2	control	UINT16	Get or Set
4	TBD1 (Get only) TBD1+TBD2 (Set only)	UINT32	Payload Length
8	cmd	UINT8[TBD1]	Command
TBD1+8	dataset	UINT8[TBD2]	Dataset (Set only)

Disciplining Response

The Disciplining get response returns the dataset response. The Disciplining set response consists of the header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x07	UINT8	Disciplining item
2	control	UINT16	Get or Set
4	0 (Set only) TBD2 (Get only)	UINT32	Payload Length
8	dataset	UINT8[TBD2]	Dataset (Get only)

Disciplining Command Error Response

The Disciplining Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x07	UINT8	Disciplining item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.9 Estimated Phase Error Transaction (XO_CA_PHASE_ERR)

The Estimated Phase Error transaction is used to get the external oscillator's estimated phase error. The Estimated Phase Error Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Estimated Phase Error Command

The Estimated Phase Error command consists of a header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x08	UINT8	Estimated Phase Error item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Estimated Phase Error Response

The Estimated Phase Error response returns the estimated phase error.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x08	UINT8	Estimated Phase Error item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	phase_err	INT32	Estimated Phase Error

Estimated Phase Error Error Response

The Estimated Phase Error Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x08	UINT8	Estimated Phase Error item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.10 Estimated Frequency Error Transaction (XO_CA_FREQ_ERR)

The Estimated Frequency Error transaction is used to get the external oscillator's estimated frequency error. The Estimated Phase Error Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Estimated Frequency Error Command

The Estimated Frequency Error command consists of a header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x09	UINT8	Estimated Frequency Error item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Estimated Frequency Error Response

The Estimated Frequency Error response returns the estimated frequency error.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x09	UINT8	Estimated Frequency Error item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	freq_err	FLT32	Estimated Phase Error

Estimated Frequency Error Error Response

The Estimated Frequency Error Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x09	UINT8	Estimated Frequency Error item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.15.11 Oscillator Type Transaction (XO_CA_OSC_TYPE)

The Oscillator Type transaction is used to get the external oscillator's type. The Oscillator Type Error Response consists of a message header and error code indicating a failure of the XO to carry out the operation.

Oscillator Type Command

The Oscillator Type command consists of a header only.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x0A	UINT8	Oscillator Type item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Oscillator Type Response

The Oscillator Type response returns the external oscillator type value.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x0A	UINT8	Oscillator Type item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	type	UINT32	Oscillator Type

Oscillator Type Error Response

The Oscillator Type Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x36	UINT8	XO Identifier
1	0x0A	UINT8	Oscillator Type item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.16 LED Control

The messages used by the LED Control Component (EC) are defined below.

8.3.2.16.1 Mode Transaction (EC_CA_MODE)

The Mode transaction is used to get or set the LED usage mode state. The Mode Error Response consists of a message header and error code indicating a failure of the EC to carry out the operation.

Mode Command

The Mode command consists of a header, and LED handle, and a mode state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x37	UINT8	EC Identifier
1	0x00	UINT8	Mode item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	led	LE_INDEX	LED index
12	md	EC_MODE	Mode (Set only)

Mode Response

The Mode Command get response returns the LED usage mode state. The Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x37	UINT8	EC Identifier
1	0x00	UINT8	Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
12	md	EC_MODE	Mode (Get only)

Mode Error Response

The Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x37	UINT8	EC Identifier
1	0x00	UINT8	Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.16.2 State Transaction (EC_CA_STATE)

The State transaction is used to get or set the LED display state. The State Error Response consists of a message header and error code indicating a failure of the EC to carry out the operation.

State Command

The State command consists of a header and a LED display state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x37	UINT8	EC Identifier
1	0x01	UINT8	State item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	led	LE_INDEX	LED index
12	st	EC_STATE	State (Set only)

State Response

The State Command get response returns the LED display state. The State set response consists of the header only.

Offset	Field	TYPE	Description
0	0x37	UINT8	EC Identifier
1	0x01	UINT8	State item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	st	EC_STATE	State (Get only)

State Error Response

The State Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x37	UINT8	EC Identifier
1	0x01	UINT8	State item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.17 General Purpose Input

The messages used by the General Purpose Input Component (GI) are defined below.

8.3.2.17.1 Value Transaction (GI_CA_VALUE)

The Value transaction is used to get the specified GPI's current input value. The Value State Error Response consists of a message header and error code indicating a failure of the GI to carry out the operation.

Value Command

The Value command consists of a header only.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x00	UINT8	Value item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	gpi	ID_PIN	GPI index

Value Response

The Value State response returns the specified GPI current input value.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x00	UINT8	Value item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	val	BOOL	GPI Input value

Value Error Response

The Value Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x00	UINT8	Value item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.17.2 Edge Transaction (GI_CA_EDGE)

The Edge transaction is used to get or set the GPI's trigger edge used when detecting input changes. The Edge Error Response consists of a message header and error code indicating a failure of the GI to carry out the operation.

Edge Command

The Edge command consists of a header, and the GPI's trigger edge when set and a header only with get.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x01	UINT8	Edge item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	gpi	ID_PIN	GPI index
12	edge	EDGE	GPI Edge (Set only)

Edge Response

The Edge Command get response returns the output mode setting. The Edge set response consists of the header only.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x01	UINT8	Edge item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	edge	EDGE	GPI Edge (Get only)

Edge Error Response

The Edge Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x01	UINT8	Edge item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.17.3 Timestamp Enable Transaction (GI_CA_TIMESTAMP_EN)

The Timestamp Enable transaction is used to get or set the GPI's timestamp enable state used when time stamping input changes. The Timestamp Enable Error Response consists of a message header and error code indicating a failure of the GI to carry out the operation.

Timestamp Enable Command

The Timestamp Enable command consists of a header, and the GPI's timestamp enable state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x02	UINT8	Timestamp enable item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	gpi	ID_PIN	GPI index
12	en	BOOL	Timestamp enable (Set only)

Timestamp Enable Response

The Timestamp Enable Command get response returns the output mode setting. The Timestamp Enable set response consists of the header only.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x02	UINT8	Timestamp enable item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	BOOL	Timestamp enable (Get only)

Timestamp Enable Error Response

The Timestamp Enable Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x02	UINT8	Timestamp enable item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.17.4 Get Number of Instances Transaction (GI_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of GPIO Inputs present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the GI to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x03	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of GPIO Inputs present in the system.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x03	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x38	UINT8	GI Identifier
1	0x03	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.18 General Purpose Output

The messages used by the General Purpose Input Component (GPO) are defined below.

8.3.2.18.1 Signature Control Transaction (GO_CA_SIG_CTL)

The Signature Control transaction is used to get or set the GPO's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the GO to carry out the operation. **OD_PIN_ALL** is invalid on a get. If **OD_PIN_ALL** is requested on a set, the passed signature control setting will be applied to all outputs.

Signature Control Command

The Signature Control command consists of a header, a GPO index and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	gpo	OD_PIN	GPO index
12	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x00	UINT8	Signature Control item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.18.2 Output Enable Transaction (GO_CA_OUTPUT_EN)

The Output Enable transaction is used to get or set the GPO's output enable state. The Output Enable Error Response consists of a message header and error code indicating a failure of the GO to carry out the operation. If **OD_PIN_ALL** is requested on a get, a bit mask indexed by **OD_PIN** is returned with bits set indicating which enables are high when set. If a single input is specified a **TRUE** or **FALSE** is indicating whether the enable is set or cleared. If **OD_PIN_ALL** is requested on a set, the passed **TRUE** or **FALSE** will set or clear all output enables.

Output Enable Command

The Output Enable command consists of a header, a GPO index, and an output enable state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x01	UINT8	Output Enable item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	gpo	OD_PIN	GPO index
12	en	UINT32	Output Enable (Set only)

Output Enable Response

The Output Enable Command get response returns the output enable state. The Output Enable set response consists of the header only.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x01	UINT8	Output Enable item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	UINT32	Output Enable (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x01	UINT8	Output Enable item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.18.3 Value Transaction (GO_CA_VALUE)

The Value transaction is used to get the specified GPO's current output value. The Value Error Response consists of a message header and error code indicating a failure of the GO to carry out the operation. If **OD_PIN_ALL** is requested a bit mask indexed by **OD_PIN** is returned with bits set indicating which inputs are logic high when set. If a single input is specified a **TRUE** or **FALSE** is returned indicating whether the input is set or cleared.

Value Command

The Value command consists of a header, and a GPO index.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x02	UINT8	Value item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	gpo	OD_PIN	GPO index

Value Response

The Value Command get response returns the specified GPO's current output value.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x02	UINT8	Value item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	val	UINT32	Value

Value Error Response

The Value Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x02	UINT8	Value item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.18.4 Mode Transaction (GO_CA_MODE)

The Mode transaction is used to get or set the GPO's output mode state. The Mode Error Response consists of a message header and error code indicating a failure of the GO to carry out the operation. If **OD_PIN_ALL** is requested on a set, the passed mode setting will be applied to all outputs.

Mode Command

The Mode command consists of a header, a GPO index and a mode value when set and a header only with get.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x03	UINT8	Mode item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	gpo	OD_PIN	GPO index
12	mode	OD_MODE	Mode (Set only)

Mode Response

The Mode Command get response returns the mode value. The Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x03	UINT8	Mode item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	mode	OD_MODE	Mode (Get only)

Mode Error Response

The Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x03	UINT8	Mode item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.18.5 DVM Value Transaction (GO_CA_DVM_VALUE)

The DVM Value transaction is used to get or set the GPO's Direct Value Mode (DVM) value state. The DVM Value Error Response consists of a message header and error code indicating a failure of the GO to carry out the operation. If **OD_PIN_ALL** is

requested on a get, a bit mask indexed by **OD_PIN** is returned with bits set indicating which values are logic high when set. If a single input is specified a **TRUE** or **FALSE** is returned indicating whether the value is set or cleared. If **OD_PIN_ALL** is requested on a set, the passed **TRUE** or **FALSE** will set or clear all output values.

DVM Value Command

The DVM Value command consists of a header, a GPO index and a DVM value when set and a header only with get.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x04	UINT8	DVM Value item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	gpo	OD_PIN	GPO index
12	val	UINT32	DVM Value (Set only)

DVM Value Response

The DVM Value Command get response returns the DVM value. The DVM Value set response consists of the header only.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x04	UINT8	DVM Value item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	val	UINT32	DVM Value (Get only)

DVM Value Error Response

The DVM Value Error Response consists of a message header and error code.

Offset	0x39	UINT8	GO Identifier
0	0x04	UINT8	DVM Value item
1	0x03	UINT8	Window Size item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.18.6 Match Enable Transaction (GO_CA_MATCH_EN)

The Match Enable transaction is used to get or set the GPO's match enable state and level. The Match Enable Error Response consists of a message header and error code indicating a failure of the GO to carry out the operation. If **OD_PIN_ALL** is requested on a get, a bit mask indexed by **OD_PIN** is returned with bits set indicating which match time enables are logic high when set. If a single input is specified a **TRUE** or **FALSE** is returned indicating whether the match time enable is set or cleared. If **OD_PIN_ALL** is requested on a set, the passed match enable will be applied to all outputs.

Match Enable Command

The Match Enable command consists of a header, a GPO index, and a match enable state structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x05	UINT8	Match Enable item
2	control	UINT16	Get or Set
4	8 (Get only) 12 (Set only)	UINT32	Payload Length
8	matchInfo	GO_MATCH_SEL	GPO match selection structure
8	gpo	OD_PIN	GPO index
12	lvl	LEVEL	Low or High match time
16	en	UINT32	Match Enable (Set only)

Match Enable Response

The Match Enable Command get response returns the match enable state structure. The Match Enable set response consists of the header only.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x05	UINT8	Match Enable item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	UINT32	Match Enable (Get only)

Match Error Response

The Match Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x05	UINT8	Match Enable item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.18.7 Square Wave Transaction (GO_CA_SQUARE)

The Square Wave transaction is used to get or set the GPO's square wave output configuration structure. The Square Wave Error Response consists of a message header and error code indicating a failure of the GO to carry out the operation. **OD_PIN_ALL** is invalid on a get. If **OD_PIN_ALL** is requested on a set, the passed square wave configuration will be applied to all outputs. Square wave alignment is automatically enabled after configuration during the set.

Square Wave Command

The Square Wave command consists of a header, a GPO index, and a square wave output configuration structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x06	UINT8	Square Wave item
2	control	UINT16	Get or Set
4	4 (Get only) 20 (Set only)	UINT32	Payload Length
8	gpo	OD_PIN	GPO index
12	config	GO_SQUARE	Square wave configuration structure (Set only)
12	off	INT32	Offset (Set only)
16	per	GO_PER	Period of square wave (Set only)
20	pw	UINT32	Pulse Width (Set only)
24	ae	EDGE	Active Edge (Set only)

Square Wave Response

The Square Wave Command get response returns the level and the specified GPO's square wave output configuration structure. The Square Wave set response consists of the header only.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x06	UINT8	Square Wave item
2	control	UINT16	Get or Set
4	0 (Set only) 16 (Get only)	UINT32	Payload Length
8	config	GO_SQUARE	Square wave configuration structure (Get only)
8	off	INT32	Offset (Get only)
12	per	GO_PER	Period of square wave (Get only)
16	pw	UINT32	Pulse Width (Get only)
20	ae	EDGE	Active Edge (Get only)

Square Wave Response

The Square Wave Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x06	UINT8	Square Wave item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.18.8 Get Number of Instances Transaction (GO_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of GPIO Outputs present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the GO to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x07	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of GPIO Outputs present in the system.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x07	UINT8	Get Number of Instances item
2	control	UINT16	Get or Set
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x39	UINT8	GO Identifier
1	0x07	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19 PTP Reference Component API

8.3.2.19.1 Module Information Transaction (PTR_CA_MODULE_INFO)

The Module Information transaction is used to get information about the PTP module attached to the system. The Module Information Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Module Information Command

The Module Information command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x01	UINT8	Module Information item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	inst	UINT32	Instance

Module Information Response

The Module Information get response returns text strings containing the PTP Module Software Serial Number and Build Date.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x01	UINT8	Module Information item
2	0x0000	UINT16	Get
4	33	UINT32	Payload Length
8	moduleInfo	PTL_MODULE_INFO	Module Info Structure
8	ptpVersionNumber	UINT32	PTP Version
12	softwareVersion	UINT32	Software Version
16	hardwareVersion	Char	HW Version (char)
17	Filler	Char[3]	Filler (for byte-alignment)
20	softDate	char[12]	Build Date String
32	softTime	Char[9]	Build Time string

Module Information Error Response

The Module Information Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x01	UINT8	Module Information item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.2 Ethernet ITF Transaction (PTR_CA_ETHERNET_ITF)

The Ethernet ITF transaction is used to get and set information about the current Ethernet configuration of the PTP module. The Ethernet ITF Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Ethernet ITF Command

The Ethernet ITF get command consists of a header only.
The Ethernet ITF set command consists of a header and Ethernet information. The full ethSettings structure should be provided when issuing the set command, even though some elemts fo the structure may be ignored.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x02	UINT8	Ethernet ITF item
2	Control	UINT16	Get or Set
4	4 (Get Only) 32 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	ethSettings	PTL_ETH_INFO	Ethernet Info (Set Only)
12	dhcpEnabled	BOOL	Enable/Disable DHCP (Set Only)
16	macAddress	UINT8[8]	MAC Address (Set Only)
24	staticIpAddr	UINT8[4]	Static IP Addr (Set Only)
24	netMask	UINT8[4]	IP Netmask (Set Only)
28	defaultGateway	UINT8[4]	Default Gateway (Set Only)
32	ipAddress	UINT8[4]	IP Address (Set Only)

Ethernet ITF Response

The Ethernet ITF get response consists of a header and Ethernet information. The Ethernet ITF set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x02	UINT8	Ethernet ITF item
2	Control	UINT16	Get or Set
4	0 (Set only) 24 (Get Only)	UINT32	Payload Length
8	ethSettings	PTL_ETH_INFO	Ethernet Info (Get Only)
8	dhcpEnabled	BOOL	Enable/Disable DHCP (Get Only)
12	macAddress	UINT8[8]	MAC Address (Get Only)
20	staticIpAddr	UINT8[4]	Static IP Addr (Get Only)
24	netMask	UINT8[4]	IP Netmask (Get Only)
28	defaultGateway	UINT8[4]	Default Gateway (Get Only)
	ipAddress	UINT8[4]	IP Address (Get Only)

Ethernet ITF Error Response

The Ethernet ITF Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x02	UINT8	Ethernet ITF item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.3 Clock Settings Transaction (PTR_CA_CLOCK_SETTINGS)

The Clock Settings transaction is used to get and set information about the current Clock settings of the PTP module. The Clock Settings Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Clock Settings Command

The Clock Settings get command consists of a header only.
The Clock Settings set command consists of a header and Ethernet information.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x03	UINT8	Clock Settings item
2	Control	UINT16	Get or Set
4	4 (Get Only) 12 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	ethSettings	PTL_CLK_SETTINGS	Ethernet Info (Set Only)
12	ptpClockRunning	BOOL	PTP Clock Running (Set Only)
16	usingExternalClock	BOOL	External Clock (Set Only)

Clock Settings Response

The Clock Settings get response consists of a header and Ethernet information.
The Clock Settings set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x03	UINT8	Clock Settings item
2	Control	UINT16	Get or Set
4	0 (Set only) 8 (Get Only)	UINT32	Payload Length
8	ethSettings	PTL_CLK_SETTINGS	Ethernet Info (Set Only)
8	ptpClockRunning	BOOL	PTP Clock Running (Set Only)
12	usingExternalClock	BOOL	External Clock (Set Only)

Clock Settings Error Response

The Clock Settings Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x03	UINT8	Clock Settings item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.4 Unit Settings Transaction (PTR_CA_UNIT_SETTINGS)

The Unit Settings transaction is used to get and set general settings for the PTP module. The Unit Settings Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Unit Settings Command

The Unit Settings get command consists of a header only.
The Unit Settings set command consists of a header and Unit Settings information.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x04	UINT8	Unit Settings item
2	Control	UINT16	Get or Set
4	4 (Get Only) 32 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	unitSettings	PTL_UNIT_SETTINGS	Unit Settings (Set Only)
12	clockIdentity	UINT8[8]	Clock Identity (Set Only)
20	oneStepMode	BOOL	One Step Mode (Set Only)
24	unicast	BOOL	Unicast Mode (set only)
28	domainNumber	UINT32	Domain Number (Set Only)
32	priority1	UINT32	Prority 1 (Set Only)
36	priority2	UINT32	Prority 2 (Set Only)

Unit Settings Response

The Unit Settings get response consists of a header and Ethernet information. The Unit Settings set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x04	UINT8	Unit Settings item
2	Control	UINT16	Get or Set
4	0 (Set only) 28 (Get Only)	UINT32	Payload Length
8	unitSettings	PTL_UNIT_SETTINGS	Unit Settings (Set Only)
8	clockIdentity	UINT32[8]	Clock Identity (Set Only)
16	oneStepMode	BOOL	One Step Mode (Set Only)
20	unicast	BOOL	Unicast Mode (set only)
24	domainNumber	UINT32	Domain Number (Set Only)
28	priority1	UINT32	Prority 1 (Set Only)
32	priority2	UINT32	Prority 2 (Set Only)

Unit Settings Error Response

The Unit Settings Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x04	UINT8	Unit Settings item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.5 Port State Transaction (PTR_CA_PORT_STATE)

The Port State transaction is used to get state information about the PTP module port. The Port State Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Port State Command

The Port State command consists of a header only.
The Port State Set command consists of a header and Port State information.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x05	UINT8	Port State item
2	0x0000	UINT16	Get
4	4 (Get Only) 24 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	portState	PTL_PORT_STATE	Port State structure
12	portNumber	UINT32	Port Number
16	portEnabled	BOOL	Port Enable status
20	portState	PTL_PTP_STATE	Port State enumeration
24	linkConnected	BOOL	Port Link Connected

Port State Response

The Port State get response returns Port State information.
The Port State set response returns the Header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x05	UINT8	Port State item
2	Control	UINT16	Get or set
4	16 (Get Only) 0 (Set Only)	UINT32	Payload Length
8	portState	PTL_PORT_STATE	Port State structure
8	portNumber	UINT32	Port Number
12	portEnabled	BOOL	Port Enable status
16	portState	PTL_PTP_STATE	Port State enumeration
20	linkConnected	BOOL	Port Link Connected

Port State Error Response

The Port State Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x05	UINT8	Port State item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.6 Port Settings Transaction (PTR_CA_PORT_SETTINGS)

The Port Settings transaction is used to get and set general settings for the port on the PTP module. The Port Settings Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Port Settings Command

The Port Settings get command consists of a header only.
The Port Settings set command consists of a header and Port Settings information.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x06	UINT8	Port Settings item
2	Control	UINT16	Get or Set
4	4 (Get Only) 32 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	portSettings	PTL_PORT_SETTINGS	Port Settings (Set Only)
16	portNumber	UINT32	Clock Identity (Set Only)
20	annRcptTimeout	UINT32	One Step Mode (Set Only)
24	logAnnInterval	INT32	Slave Only Mode (set only)
28	logSyncInterval	INT32	Unicast Mode (set only)
32	logDelay ReqInterval	INT32	Domain Number (Set Only)
36	logPeerDelay ReqInterval	INT32	Prority 1 (Set Only)
40	delayMechanism	PTL_DELAY_MECH	Prority 2 (Set Only)

Port Settings Response

The Port Settings get response consists of a header and information on port settings. The Port Settings set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x06	UINT8	Port Settings item
2	Control	UINT16	Get or Set
4	0 (Set only) 28 (Get Only)	UINT32	Payload Length
8	portSettings	PTL_PORT_SETTINGS	Port Settings (Set Only)
8	portNumber	UINT32	Clock Identity (Set Only)
12	annRcptTimeout	UINT32	One Step Mode (Set Only)
16	logAnnInterval	INT32	Slave Only Mode (set only)
20	logSyncInterval	INT32	Unicast Mode (set only)
24	logDelay ReqInterval	INT32	Domain Number (Set Only)
28	logPeerDelay ReqInterval	INT32	Prority 1 (Set Only)
32	delayMechanism	PTL_DELAY_MECH	Prority 2 (Set Only)

Port Settings Error Response

The Port Settings Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x06	UINT8	Port Settings item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.7 Clock Quality Transaction (PTR_CA_CLOCK_QUALITY)

The Clock Quality transaction is used to get information about quality of the PTP module's clock. The Clock Quality Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Clock Quality Command

The Clock Quality command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x07	UINT8	Clock Quality item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	inst	UINT32	Instance

Clock Quality Response

The Clock Quality response returns information concerning the quality of the PTP module's clock.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x07	UINT8	Clock Quality item
2	0x0000	UINT16	Get
4	12	UINT32	Payload Length
8	clockQuality	PTL_CLK_QUALITY	Clock Quality structure
8	clockClass	UINT32	Clock Class
12	clockAcc	PTL_CLK_ACC	Clock Accuracy
16	offset ScaledLog Variance	UINT32	Offset Scaled Log Variance

Clock Quality Error Response

The Clock Quality Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x07	UINT8	Clock Quality item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.8 Time Properties Transaction (PTR_CA_TIME_PROPERTIES)

The Time Properties transaction is used to get information about the PTP module's current time. The Time Properties Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Time Properties Command

The Time Properties command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x08	UINT8	Time Properties item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	inst	UINT32	Instance

Time Properties Response

The Time Properties get response returns information about the PTP module's current time.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x08	UINT8	Time Properties item
2	0x0000	UINT16	Get
4	32	UINT32	Payload Length
8	timeProperties	PTL_TIME_PROP	Port State structure
8	utcOffset	INT32	UTC Offset
12	utcOffsetValid	BOOL	UTC Offset Valid
16	forwardLeap	BOOL	Forward Leap Sec
20	backwardLeap	BOOL	Backward Leap Sec
24	timeTraceable	BOOL	Is Time Tracable
28	freqTraceable	BOOL	Is Freq Traceable
32	ptpTimescale	BOOL	PTP Timescale
36	timeSource	PTL_TIME_SRC	Time Source Enumeration

Time Properties Error Response

The Time Properties Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x08	UINT8	Time Properties item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.9 Parent Properties Transaction (PTR_CA_PARENT_PROP)

The Parent Properties transaction is used to get state information about the parent of the PTP module port. (The Parent is the PTP unit on the network that this PTP module is synchronized to.) The Parent Properties Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Parent Properties Command

The Parent Properties command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x09	UINT8	Parent Properties item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	inst	UINT32	Instance

Parent Properties Response

The Parent Properties response returns state information about the parent of the PTP module port.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x09	UINT8	Parent Properties item
2	0x0000	UINT16	Get
4	24	UINT32	Payload Length
8	parentProp	PTL_PARENT_PROP	Port State structure
8	clockId	UINT8[8]	Clock Identity
16	portNumber	UINT32	Port Number
20	stats Calculated	BOOL	Stats Calculated?
24	observed OSLV	UINT32	Observed OSLV
28	observed CPCR	UINT32	Observed CPCR

Parent Properties Error Response

The Parent Properties Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x09	UINT8	Parent Properties item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.10 GM Properties Transaction (PTR_CA_GM_PROP)

The GM Properties transaction is used to get state information about the Grandmaster Clock on the network of the PTP module port. The GM Properties Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

GM Properties Command

The GM Properties command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0A	UINT8	GM Properties item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	inst	UINT32	Instance

GM Properties Response

The GM Properties response returns state information about the Grandmaster Clock on the network of the PTP module port.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0A	UINT8	GM Properties item
2	0x0000	UINT16	Get
4	28	UINT32	Payload Length
8	gmProperties	PTL_GM_PROP	Port State structure
8	clockId	UINT8[8]	Clock Identity
16	clockQuality	PTL_CLK_QUALITY	Clock Quality structure
16	clockClass	UINT32	Clock Class
20	clockAcc	PTL_CLK_ACC	Clock Accuracy enum
24	offset ScaledLog Variance	UINT32	Offset Scaled Log Variance
28	priority1	UINT32	priority 1
32	priority2	UINT32	Priority 2

GM Properties Error Response

The GM Properties Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0A	UINT8	GM Properties item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.11 TOD Enable Transaction (PTR_CA_TOD_ENABLED)

The TOD Enable transaction is used to get and set whether the PTP module will output the current TOD (Time of Day).. The TOD Enable Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

TOD Enable Command

The TOD Enable get command consists of a header only.
The TOD Enable set command consists of a header and the desired TOD Enable value.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0B	UINT8	TOD Enable item
2	Control	UINT16	Get or Set
4	4 (Get Only) 12 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	todEnabled	BOOL	TOD Enable (Set Only)
16	timescale	ML_TIME_SCALE	Time Scale Enumeration

TOD Enable Response

The TOD Enable get response consists of a header and the current TOD Enable value. The TOD Enable set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0B	UINT8	TOD Enable item
2	Control	UINT16	Get or Set
4	0 (Set only) 8 (Get Only)	UINT32	Payload Length
8	todEnabled	BOOL	TOD Enable (Get Only)
12	timescale	ML_TIME_SCALE	Time Scale Enumeration

TOD Enable Error Response

The TOD Enable Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0B	UINT8	TOD Enable item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.12 PPS Enable Transaction (PTR_CA_PPS_ENABLED)

The PPS Enable transaction is used to get and set whether the PTP module will output the PPS as a PTP slave. The PPS Enable Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

PPS Enable Command

The PPS Enable get command consists of a header only.
The PPS Enable set command consists of a header and the desired PPS Enable value..

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0C	UINT8	PPS Enable item
2	Control	UINT16	Get or Set
4	4 (Get Only) 8 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	ppsEnabled	BOOL	TOD Enable (Set Only)

PPS Enable Response

The PPS Enable get response consists of a header and the current TOD Enable value.
The PPS Enable set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0C	UINT8	PPS Enable item
2	Control	UINT16	Get or Set
4	0 (Set only) 4 (Get Only)	UINT32	Payload Length
8	ppsEnabled	BOOL	TOD Enable (Get Only)

PPS Enable Error Response

The PPS Enable Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0C	UINT8	PPS Enable item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.13 PPS Edge Transaction (PTR_CA_PPS_EDGE)

The PPS Edge transaction is used to get and set which edge is the active output PPS edge. Enabling this boolean will use the rising edge as the active PPS edge, and disabling it will use the falling edge. The PPS Edge Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

PPS Edge Command

The PPS Edge get command consists of a header only.
The PPS Edge set command consists of a header and the desired PPS Edge value.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0D	UINT8	PPS Edge item
2	Control	UINT16	Get or Set
4	4 (Get Only) 8 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	ppsRisingEdge	BOOL	PPS Rising Edge Enable (Set Only)

PPS Edge Response

The PPS Edge get response consists of a header and the current PPS Edge value.

The PPS Edge set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0D	UINT8	PPS Edge item
2	Control	UINT16	Get or Set
4	0 (Set only) 4 (Get Only)	UINT32	Payload Length
8	ppsRisingEdge	BOOL	PPS Rising Edge Enable (Get Only

PPS Edge Error Response

The PPS Edge Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0D	UINT8	PPS Edge item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.14 Save Settings (PTR_CA_SAVE_SETTINGS_TO_ROM)

The Save Settings transaction is used to save the current PTP module settings to ROM. The Save Settings Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Save Settings Command

The Save Settings set command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0E	UINT8	Save Settings item
2	Control	UINT16	Set
4	4 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance

Save Settings Response

The Save Settings set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0E	UINT8	Save Settings item
2	Control	UINT16	Set
4	0	UINT32	Payload Length

Save Settings Error Response

The Save Settings Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0E	UINT8	Save Settings item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.15 Reset Module (PTR_CA_RESET_MODULE)

The Reset Module transaction is used to send a reset request to the PTP Module. The Reset Module Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Reset Module Command

The Reset Module set command consists of a header and the desired Reset Type.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0F	UINT8	Save Settings item
2	Control	UINT16	Set
4	8 (Set Only)	UINT32	Payload Length
8	Inst	UINT32	Instance
12	resetType	PTL_RESET	Reset Type

Reset Module Response

The Reset Module set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0F	UINT8	Save Settings item
2	Control	UINT16	Set
4	0	UINT32	Payload Length

PPS Edge Error Response

The Reset Module Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x0F	UINT8	Save Settings item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.16 Get Number of Instances Transaction (PTR_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of PTP reference instances in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x10	UINT8	Get Number of Instances item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances response returns the number of PTP reference instances in the system.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x10	UINT8	Get Number of Instances item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x10	UINT8	Get Number of Instances item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.17 ReInit Module Transaction (PTR_CA_NUM_INST)

The ReInit Module transaction is used to reinitialize the module. The ReInit Module Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

ReInit Module Command

The ReInit Module command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x11	UINT8	Reinit Module item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	Inst	UINT32	Instance

ReInit Module Response

The ReInit Module response consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x11	UINT8	Reinit Module item
2	0x0000	UINT16	Get
4	0	UINT32	Payload Length

ReInit Module Error Response

The ReInit Module Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x11	UINT8	Reinit Module item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.18 PTP Validity Transaction (PTR_CA_VALIDITY)

The PTP Validity transaction is used to get the time and 1PPS reference validity structure for the PTP module when used as a slave. The Validity Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

ReInit Module Command

The ReInit Module command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x12	UINT8	PTP Validity item
2	0x0000	UINT16	Get
4	4	UINT32	Payload Length
8	Inst	UINT32	Instance

ReInit Module Response

The PTP Validity response returns the time and 1PPS validity structure.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x12	UINT8	PTP Validity item
2	0x0000	UINT16	Get
4	8	UINT32	Payload Length
8	validInfo	EL_VALIDITY	Validity Structure
8	timeValid	BOOL	Time validity state
12	ppsValid	BOOL	1PPS validity state

ReInit Module Error Response

The PTP Validity Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x12	UINT8	PTP Validity item
2	0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.19 PTP Mode Transaction (PTR_CA_MODE)

The PTP Mode transaction is used to get and set the current operating mode of the PTP module. The PTP Mode Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

PTP Mode Command

The PTP Mode get command consists of a header only.
The PTP Mode set command consists of a header and the desired PTP Operating Mode value.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x13	UINT8	PTP Mode item
2	Control	UINT16	Get or Set
4	4 (Get Only) 8 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	opMode	PTL_OP_MODE	PTP operating Mode (Set Only)

PTP Mode Response

The PTP Mode get response consists of a header and the current PTP Mode value.
The PTP Mode set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x13	UINT8	PTP Mode item
2	Control	UINT16	Get or Set
4	0 (Set only) 4 (Get Only)	UINT32	Payload Length
8	opMode	PTL_OP_MODE	PTP operating Mode (Set Only)

PTP Mode Error Response

The PTP Mode Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x13	UINT8	PTP Mode item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.20 PTP MAC Address Transaction (PTR_CA_MAC_ADDR)

The PTP MAC Address transaction is used to get and set the current MAC Address of the PTP Module. The PTP MAC Address Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

PTP MAC Address Command

The PTP MAC Address get command consists of a header only.
The PTP MAC Address set command consists of a header, the desired MAC address, and a factory password.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x14	UINT8	PTP Mac Address Item
2	Control	UINT16	Get or Set
4	4 (Get Only) 20 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	macAddr	UINT8[8]	MAC Address (Set Only)
20	password	UINT8[8]	Password (Set Only)

PTP MAC Address Response

The PTP MAC Address get response consists of a header and the current PTP Mode value.
The PTP MAC Address set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x14	UINT8	PTP Mac Address Item
2	Control	UINT16	Get or Set
4	0 (Set only) 8 (Get Only)	UINT32	Payload Length
8	macAddr	UINT8[8]	MAC Address

PTP MAC Address Error Response

The PTP MAC Address Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x14	UINT8	PTP Mac Address Item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.21 Master Active Transaction (PTR_CA_MASTER_ACTIVE)

The Master Active transaction is used to get whether the PTP Master is currently active. The Master Active Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Master Active Command

The Master Active get command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x15	UINT8	Master Active item
2	Control	UINT16	Get
4	4	UINT32	Payload Length
8	inst	UINT32	Instance

Master Active Response

The Master Active get response consists of a header and the current Master Active value.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x15	UINT8	Master Active item
2	Control	UINT16	Get
4	4	UINT32	Payload Length
8	masterActive	BOOL	Master Active status

Master Active Error Response

The Master Active Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x15	UINT8	Master Active item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.22 Module Status Transaction (PTR_CA_MODULE_STATUS)

The Module Status transaction is used to get the current PTP Module status information. The Module Status Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Module Status Command

The Module Status get command consists of a header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x16	UINT8	Module Status item
2	Control	UINT16	Get
4	4	UINT32	Payload Length
8	inst	UINT32	Instance

Module Status Response

The Module Status get response consists of a header and the current Master Active value.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x16	UINT8	Module Status item
2	Control	UINT16	Get
4	4	UINT32	Payload Length
8	resetCause	PTL_RESET_CAUSE	Reset Cause info

Module Status Error Response

The Module Status Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x16	UINT8	Module Status item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.19.23 PTP User Description Transaction (PTR_CA_PTP_USER_DESCRIPTION)

The PTP User Description transaction is used to get and set the PTP User Description fields broadcast by the PTP Module. The PTP User Description Transaction Error Response consists of a message header and error code indicating a failure of the PTR to carry out the operation.

Note that the maximum size of each element of the PTP User Description fields is 16 characters.

PTP User Description Command

The PTP User Description get command consists of a header only.
The PTP User Description set command consists of a header and User Description information.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x17	UINT8	PTP User Description item
2	Control	UINT16	Get or Set
4	4 (Get Only) 36 (Set Only)	UINT32	Payload Length
8	inst	UINT32	Instance
12	userDesc	PTL_USER_DESC	PTP User Description (Set Only)
12	deviceName	char[16]	Device Name (Set Only)
28	DeviceLocation	char[16]	Device Location (Set Only)

PTP User Description Response

The PTP User Description get response consists of a header and User Description information.
The PTP User Description set response consists of the header only.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x17	UINT8	PTP User Description item
2	Control	UINT16	Get or Set
4	0 (Set only) 24 (Get Only)	UINT32	Payload Length
8	userDesc	PTL_USER_DESC	PTP User Description
8	deviceName	char[16]	Device Name
24	deviceLocation	char[16]	Device Location

PTP User Description Error Response

The PTP User Description Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x3B	UINT8	PTR Identifier
1	0x17	UINT8	PTP User Description item
2	Control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.20 EBU Output

The messages used by the EBU Output Component (EP) are defined below.

8.3.2.20.1 Signature Control Transaction (EP_CA_SIG_CTL)

The Signature Control transaction is used to get or set the EBU output's signature control state. The Signature Control Error Response consists of a message header and error code indicating a failure of the EP to carry out the operation.

Signature Control Command

The Signature Control command consists of a header and a signature control state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	EBU Output Instance
12	en	SIG_CTL	Signature Control (Set only)

Signature Control Response

The Signature Control Command get response returns the signature control state. The Signature Control set response consists of the header only.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x00	UINT8	Signature Control item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	en	SIG_CTL	Signature Control (Get only)

Signature Control Error Response

The Signature Control Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x00	UINT8	Signature Control item
2	**control|0x0001**	**UINT16**	**Get or Set and Error Response**
4	**4 (or 8)**	**UINT32**	**Payload Length**
8	**ec**	**HA_RC**	**HA Error Code**
12	**rc**	**RC**	**Optional operation specific return code**

8.3.2.20.2 Offset Transaction (EP_CA_OFFSET)

The Offset transaction is used to get or set the EBU reference's output offset. The Offset Error Response consists of a message header and error code indicating a failure of the EP to carry out the operation.

Offset Command

The Offset command consists of a header, and an offset when set and a header only with get.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x01	UINT8	Offset item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	EBU Output Instance
12	offset	INT32	Offset (Set only)

Offset Response

The Offset get response returns the output offset. The Offset set response consists of the header only.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x01	UINT8	Offset item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	offset	INT32	Offset (Get only)

Offset Error Response

The Offset Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x01	UINT8	Offset item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.20.3 Local Transaction (EP_CA_LOCAL)

The Local transaction is used to get or set the EBU reference's Local Clock including the time zone and DST rule. The Local Error Response consists of a message header and error code indicating a failure of the EP to carry out the operation.

Local Command

The Local command consists of a header, and a Local Clock structure when set and a header only with get.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x02	UINT8	Local item
2	control	UINT16	Get or Set
4	4 (Get only) 48 (Set only)	UINT32	Payload Length
8	instance	UINT32	EBU Output Instance
12	lclClk	ML_CLK_LOCAL	Local clock structure (Set only)
12	rule	ML_DST_RULE	DST rule structure (Set only)
12	ref	ML_DST_REF	DST reference (Set only)
16	in	ML_DST_POINT	DST in point structure (Set only)
16	month	ML_MONTH	Month (Set only)
20	wom	ML_WOM	Week of Month (Set only)
24	dow	ML_DOW	Day of Week (Set only)
28	hour	UINT32	Hour (Set only)
32	out	ML_DST_POINT	DST out point structure (Set only)
32	month	ML_MONTH	Month (Set only)
36	wom	ML_WOM	Week of Month (Set only)
40	dow	ML_DOW	Day of Week (Set only)
44	hour	UINT32	Hour (Set only)
48	offset	INT32	DST offset (Set only)
52	tz	INT32	Time Zone offset (Set only)

Local Response

The Local get response returns the Local Clock structure. The Local set response consists of the header only.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x02	UINT8	Local item
2	control	UINT16	Get or Set
4	0 (Set only) 44 (Get only)	UINT32	Payload Length
8	lclClk	ML_CLK_LOCAL	Local clock structure (Get only)
8	rule	ML_DST_RULE	DST rule structure (Get only)
8	ref	ML_DST_REF	DST reference (Get only)
12	in	ML_DST_POINT	DST in point structure (Get only)
12	month	ML_MONTH	Month (Get only)
16	wom	ML_WOM	Week of Month (Get only)
20	dow	ML_DOW	Day of Week (Get only)
24	hour	UINT32	Hour (Get only)
28	out	ML_DST_POINT	DST out point structure (Get only)
28	month	ML_MONTH	Month (Get only)
32	wom	ML_WOM	Week of Month (Get only)
36	dow	ML_DOW	Day of Week (Get only)
40	hour	UINT32	Hour (Get only)
44	offset	INT32	DST offset (Get only)
48	tz	INT32	Time Zone offset (Get only)

Local Error Response

The Local Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x02	UINT8	Local item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.20.4 Format Transaction (EP_CA_FORMAT)

The Format transaction is used to get or set the EBU output's format state. The Format Error Response consists of a message header and error code indicating a failure of the EP to carry out the operation.

Format Command

The Format command consists of a header and a format state when set and a header only with get.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x03	UINT8	Format item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	EBU Output Instance
12	fmt	QL_FMT	Format (Set only)

Format Response

The Format Command get response returns the format state. The Format set response consists of the header only.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x03	UINT8	Format item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	fmt	QL_FMT	Format (Get only)

Format Error Response

The Format Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x03	UINT8	Format item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.20.5 Time Scale Transaction (EP_CA_TIME_SCALE)

The Time Scale transaction is used to get or set the EBU Output's Time Scale. The Time Scale Error Response consists of a message header and error code indicating a failure of the EP to carry out the operation.

Time Scale Command

The Time Scale command consists of a header, and the EBU Output's Time Scale when set and a header only with get.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x04	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	EBU Output Instance
12	ts	ML_TIME_SCALE	Time Scale (Set only)

Time Scale Response

The Time Scale get response returns the EBU Output's Time Scale. The Time Scale set response consists of the header only.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x04	UINT8	Time Scale item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	ts	ML_TIME_SCALE	Time Scale (Get only)

Time Scale Error Response

The Time Scale Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x04	UINT8	Time Scale item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.20.6 Amplitude Transaction (EP_CA_AMPLITUDE)

The Amplitude transaction is used to get or set the EBU output's Amplitude. The Amplitude Error Response consists of a message header and error code indicating a failure of the EP to carry out the operation.

Amplitude Command

The Amplitude command consists of a header, and the amplitude when set and a header only with get.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x05	UINT8	Amplitude item
2	control	UINT16	Get or Set
4	4 (Get only) 8 (Set only)	UINT32	Payload Length
8	instance	UINT32	IRIG Ouput Instance
12	amp	UINT32	Amplitude (Set only)

Amplitude Response

The Amplitude get response returns the amplitude. The Amplitude set response consists of the header only.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x05	UINT8	Amplitude item
2	control	UINT16	Get or Set
4	0 (Set only) 4 (Get only)	UINT32	Payload Length
8	amp	UINT32	Amplitude (Get only)

Amplitude Error Response

The Amplitude Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x05	UINT8	Amplitude item
2	control|0x0001	UINT16	Get or Set and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

8.3.2.20.7 Get Number of Instances Transaction (EP_CA_NUM_INST)

The Get Number of Instances transaction is used to get the number of EBU Output's present in the system. The Get Number of Instances Error Response consists of a message header and error code indicating a failure of the EP to carry out the operation.

Get Number of Instances Command

The Get Number of Instances command consists of a header only.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x06	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	0	UINT32	Payload Length

Get Number of Instances Response

The Get Number of Instances Command get response returns the number of EBU Outputs present in the system.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x06	UINT8	Get Number of Instances item
2	control	UINT16	Get
4	4	UINT32	Payload Length
8	num	UINT32	Number of Instances

Get Number of Instances Error Response

The Get Number of Instances Error Response consists of a message header and error code.

Offset	Field	TYPE	Description
0	0x44	UINT8	EP Identifier
1	0x06	UINT8	Get Number of Instances item
2	control|0x0001	UINT16	Get and Error Response
4	4 (or 8)	UINT32	Payload Length
8	ec	HA_RC	HA Error Code
12	rc	RC	Optional operation specific return code

Document Revision History			
Rev	*ECN*	*Description*	*Date*
A	3316	*First draft of Spectracom documentation for this product.*	December 2013
2		*Switched to Safran Trusted 4D branding.*	December 2026

Technical Support

Visit Safran's TSync Support Page for more information, or to request product support:

https://safran-navigation-timing.com/support-hub/

Information furnished by Safran is believed to be accurate and reliable. However, no responsibility is assumed by Safran for its use, nor for any infringements of patents or other rights of third parties that may result from its use. Safran reserves the right to make changes without further notice to any products herein. Safran makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Safran assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. No license is granted by implication or otherwise under any patent or patent rights of Safran. Trademarks and registered trademarks are the property of their respective owners. Safran products are not intended for any application in which the failure of the Safran product could create a situation where personal injury or death may occur. Should Buyer purchase or use Safran products for any such unintended or unauthorized application, Buyer shall indemnify and hold Safran and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable legal fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Safran was negligent regarding the design or manufacture of the part.

Safran Electronics & Defense safran-navigation-timing.com