Linux Example

Assembly Design Document

1 Description

This example assembly is designed to run inside the Adamant Linux build environment. It includes a small collection of components that demonstrate how Adamant can be compiled into a single binary and run on native Linux. In particular, the assembly demonstrates a simple rate group system, commanding, telemetry reporting using events, parameters, data products and packets, and a simple fault detection and correction scheme. Explore the tables and diagrams below to learn more about the design of the example assembly.

2 Design

2.1 At a Glance

Below is a list of useful parameters and statistics that give a quick look into the makeup of the assembly.

- **Number of Components** - 35
- **Number of Component Types** - 31
- **Number of Active Components** - 13
- **Number of Passive Components** - 22
- **Number of Components with Queue** - 15
- **Number of Components without Queue** - 20
- **Number of Components with Events** - 27
- **Number of Components with Data Products** - 18
- **Number of Components with Data Dependencies** - *None*
- **Number of Components with Packets** - 14
- **Number of Components with Commands** - 22
- **Number of Components with Parameters** - 2
- **Number of Components with Faults** - 2
- **Number of Connections** - 172
- **Number of Events** - 192
- **Number of Data Products** - 37
- **Number of Data Dependencies** - *None*
- **Number of Packets** - 14
- **Number of Commands** - 66
- **Number of Parameters** - 6

- **Number of Faults** - 4

2.2 Components

All of the components in the Linux Example assembly are reusable components that can be found within the Adamant repository (in *src/components/*) with a few exceptions:

- **`Counter`** - This is an extremely simple component that produces an incrementing counter as telemetry.
- **`Fault_Producer`** - This is an component that can be used to inject a fault into the system via command.
- **`Oscillator`** - This is a simple component that produces two sinusoidal outputs into telemetry.
- **`Parameter_Manager`** - This component demonstrates how parameter tables can be copied between a non-volatile store and an active (in-use) parameter store.
- **`Interrupt_Responder`** - This is a simple component that receives a tick when an interrupt is received. It processes this interrupt by printing the interrupt's time stamp to the terminal.

These specific component are located in the Example repository in *src/components/*. A full list of the components included in the assembly can be seen below. Note that components marked as active execute on their own task within the Ada runtime.

Table 1: Linux Example Components

Name	**Type**	**Has Queue**	**Execution**
`System_Time_Instance`	`Gps_Time`	no	passive
`Ticker_Instance`	`Ticker`	no	active
`Tick_Divider_Instance`	`Tick_Divider`	no	passive
`Slow_Rate_Group`	`Rate_Group`	yes	active
`Fast_Rate_Group`	`Rate_Group`	yes	active
`Watchdog_Rate_Group`	`Rate_Group`	yes	active
`Ccsds_Command_Depacketizer_Instance`	`Ccsds_Command_Depacketizer`	no	passive
`Command_Router_Instance`	`Command_Router`	yes	active
`Event_Text_Logger_Instance`	`Event_Text_Logger`	yes	active
`Event_Filter_Instance`	`Event_Filter`	no	passive
`Event_Limiter_Instance`	`Event_Limiter`	no	passive
`Event_Splitter_Instance`	`Splitter`	no	passive

Event_Splitter_2_Instance	Splitter	no	passive
Event_Post_Mortem_Logger	Logger	no	passive
Event_Packetizer_Instance	Event_Packetizer	no	passive
Parameters_Instance	Parameters	yes	active
Parameter_Store_Instance	Parameter_Store	yes	active
Parameter_Manager_Instance	Parameter_Manager	yes	active
Product_Database_Instance	Product_Database	no	passive
Product_Packetizer_Instance	Product_Packetizer	yes	passive
Ccsds_Packetizer_Instance	Ccsds_Packetizer	no	passive
Ccsds_Socket_Interface_Instance	Ccsds_Socket_Interface	yes	active
Memory_Packetizer_Instance	Memory_Packetizer	yes	active
Interrupt_Servicer_Instance	Interrupt_Servicer	no	active
Interrupt_Responder_Instance	Interrupt_Responder	no	passive
Counter_Instance	Counter	yes	passive
Oscillator_A	Oscillator	yes	passive
Oscillator_B	Oscillator	yes	passive
Zero_Divider_Instance	Zero_Divider	no	passive
Task_Watchdog_Instance	Task_Watchdog	no	passive
Fault_Producer_Instance	Fault_Producer	no	passive

Fault_Correction_Instance	Fault_Correction	yes	active
Cpu_Monitor_Instance	Cpu_Monitor	no	passive
Queue_Monitor_Instance	Queue_Monitor	no	passive
Stack_Monitor_Instance	Stack_Monitor	no	passive

Table 2: Linux Example Component Item Counts

Component Name	**Connectors**	**Commands**	**Events**	**Data Products**	**Data Dependencies**	**Parameters**	**Packets**	**Faults**
System_Time_Instance	1	0	0	0	0	0	0	0
Ticker_Instance	2	0	0	0	0	0	0	0
Tick_Divider_Instance	4	0	1	0	0	0	0	0
Slow_Rate_Group	6	0	5	1	0	0	0	0
Fast_Rate_Group	6	0	5	1	0	0	0	0
Watchdog_Rate_Group	6	0	5	1	0	0	0	0
Ccsds_Command_Depacketizer_Instance	8	1	7	2	0	0	1	0
Command_Router_Instance	10	4	18	7	0	0	0	0
Event_Text_Logger_Instance	1	0	0	0	0	0	0	0
Event_Filter_Instance	9	7	12	3	0	0	1	0
Event_Limiter_Instance	9	8	14	3	0	0	1	0
Event_Splitter_Instance	2	0	0	0	0	0	0	0
Event_Splitter_2_Instance	2	0	0	0	0	0	0	0
Event_Post_Mortem_Logger	7	7	6	1	0	0	1	0
Event_Packetizer_Instance	7	1	0	2	0	0	1	0
Parameters_Instance	8	2	18	0	0	0	1	0
Parameter_Store_Instance	7	1	8	0	0	0	1	0
Parameter_Manager_Instance	8	1	6	0	0	0	0	0

Product_Database_Instance	8	5	18	2	0	0	1	0
Product_Packetizer_Instance	7	4	9	0	0	0	1	0
Ccsds_Packetizer_Instance	2	0	0	0	0	0	0	0
Ccsds_Socket_Interface_Instance	4	0	4	0	0	0	0	0
Memory_Packetizer_Instance	7	1	4	1	0	0	0	0
Interrupt_Servicer_Instance	2	0	0	0	0	0	0	0
Interrupt_Responder_Instance	3	0	1	0	0	0	0	0
Counter_Instance	6	3	6	0	0	0	1	0
Oscillator_A	7	3	6	1	0	3	0	0
Oscillator_B	7	3	6	1	0	3	0	0
Zero_Divider_Instance	4	1	3	0	0	0	1	0
Task_Watchdog_Instance	9	4	10	4	0	0	0	2
Fault_Producer_Instance	5	2	3	0	0	0	0	2
Fault_Correction_Instance	7	5	11	4	0	0	0	0
Cpu_Monitor_Instance	7	1	2	1	0	0	1	0
Queue_Monitor_Instance	7	1	2	1	0	0	1	0
Stack_Monitor_Instance	7	1	2	1	0	0	1	0

Component Descriptions:

- **System_Time_Instance** - This component provides the system time for the assembly.
- **Ticker_Instance** - This component uses a periodic signal to drive the assembly rate groups.
- **Tick_Divider_Instance** - This component divides a periodic signal into intervals suitable for the assembly rate groups.
- **Slow_Rate_Group** - This component provides a 0.5 Hz task for other components to execute on periodically.
- **Fast_Rate_Group** - This component provides a 5 Hz task for other components to execute on periodically.
- **Watchdog_Rate_Group** - This component provides a 1 Hz task for the watchdog components to execute on periodically.
- **Ccsds_Command_Depacketizer_Instance** - This component converts CCSDS packets

containing commands to valid Adamant formatted command types.

- **`Command_Router_Instance`** - This component provides command and command response routing throughout the assembly.
- **`Event_Text_Logger_Instance`** - This component prints events in human readable format to the terminal.
- **`Event_Filter_Instance`** - This component filters events by ID.
- **`Event_Limiter_Instance`** - This component filters out events that spam the system too frequently.
- **`Event_Splitter_Instance`** - This component splits the event stream into a filtered and limited stream and an unfiltered and unlimited stream for the post mortem log.
- **`Event_Splitter_2_Instance`** - This component splits the event stream between events to printed to the terminal and those fowarded out of the socket interface.
- **`Event_Post_Mortem_Logger`** - This component stores events in a circular buffer on the heap. In an embedded system this log will usually be instantiated in non-volatile memory, so that events can be dumped after a power cycle.
- **`Event_Packetizer_Instance`** - This component gathers events and packetizes them for downlink.
- **`Parameters_Instance`** - This component manages the active parameter table and provides parameter updates to any component in the assembly that uses parameters.
- **`Parameter_Store_Instance`** - This component manages the default parameter table, storing it on the heap. In an embedded system the default parameter table will usually be instantiated in non-volatile memory, so that it can be restored after reset.
- **`Parameter_Manager_Instance`** - This component responds to commands to copy parameter tables from the active parameter table to the default parameter table or vice versa.
- **`Product_Database_Instance`** - This component serves as the database for data products throughout the system.
- **`Product_Packetizer_Instance`** - This component periodically fetches values from the Product Database and packetizes them for downlink.
- **`Ccsds_Packetizer_Instance`** - This component converts Adamant formatted packets to CCSDS for downlink.
- **`Ccsds_Socket_Interface_Instance`** - This component receives commands from and sends events and telemetry packets through a TCP/IP Socket.
- **`Memory_Packetizer_Instance`** - This component packetizes the event post mortem log when a dump is requested.
- **`Interrupt_Servicer_Instance`** - This component waits for interrupt and, when received, passes it along to the assembly for processing.
- **`Interrupt_Responder_Instance`** - This component responds to an interrupt.
- **`Counter_Instance`** - This component periodically produces a count data product.
- **`Oscillator_A`** - This component periodically produces an oscillating data product.
- **`Oscillator_B`** - This component periodically produces an oscillating data product.
- **`Zero_Divider_Instance`** - This component responds to a command that divides by zero if received. This can be used to trigger a fault condition within the processor.
- **`Task_Watchdog_Instance`** - This component monitors other critical active components (tasks) within the assembly to make sure they continue to run.
- **`Fault_Producer_Instance`** - This component can be used to induce a fault into the system by command.

- **`Fault_Correction_Instance`** - This component produces a corrective action (a command) for any fault that is thrown in the system.
- **`Cpu_Monitor_Instance`** - This component produces a packet that includes the CPU usage percentage for each task and interrupt in the system.
- **`Queue_Monitor_Instance`** - This component produces a packet that includes the current and maximum queue usage for each component in the system.
- **`Stack_Monitor_Instance`** - This component produces a packet that includes the maximum stack and secondary stack usage for each component in the system.

2.3 Views

This section shows the example assembly visually as a set of *views*. Each shows a specific set of components and connections (while not showing other components and connections) in order to highlight a particular function of the assembly. Components that are bold are *active*, meaning they have an Ada task assigned to them. Connections are labeled with the type that is passed along them. A dotted line indicates that the connection is asynchronous, meaning the data is put onto a queue for later processing. A solid line indicates that the connection is synchronous, meaning processing of that data occurs right when the data is passed along the connector.

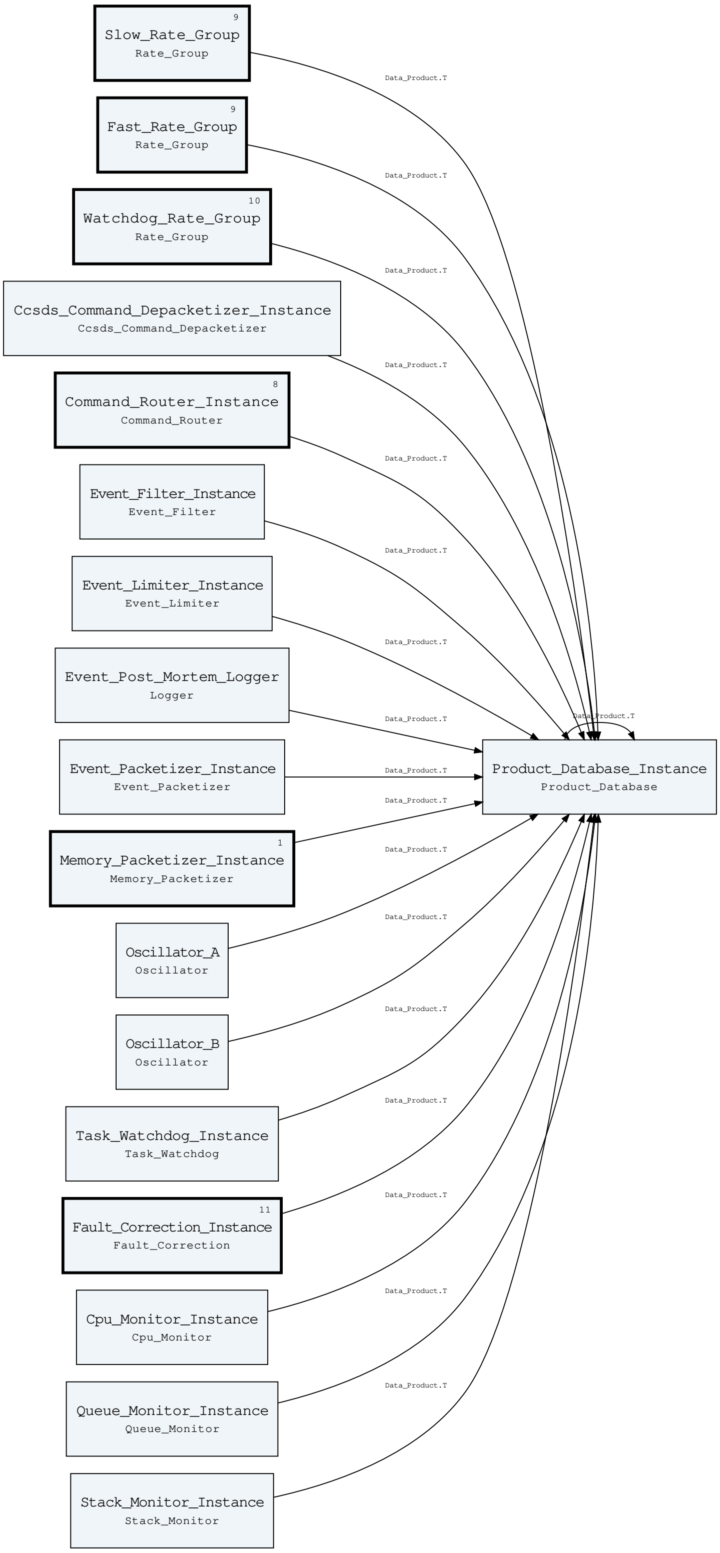

Figure 1: **Data Products View:** Any data product produced by a component during execution is sent synchronously to on onboard storage database, the Product_Database_Instance. This component stores the latest value and timestamp of each data product in the system. Other components, such as the Product_Packetizer_Instance can then fetch these data products at a later time for limit checking, packetization, etc.

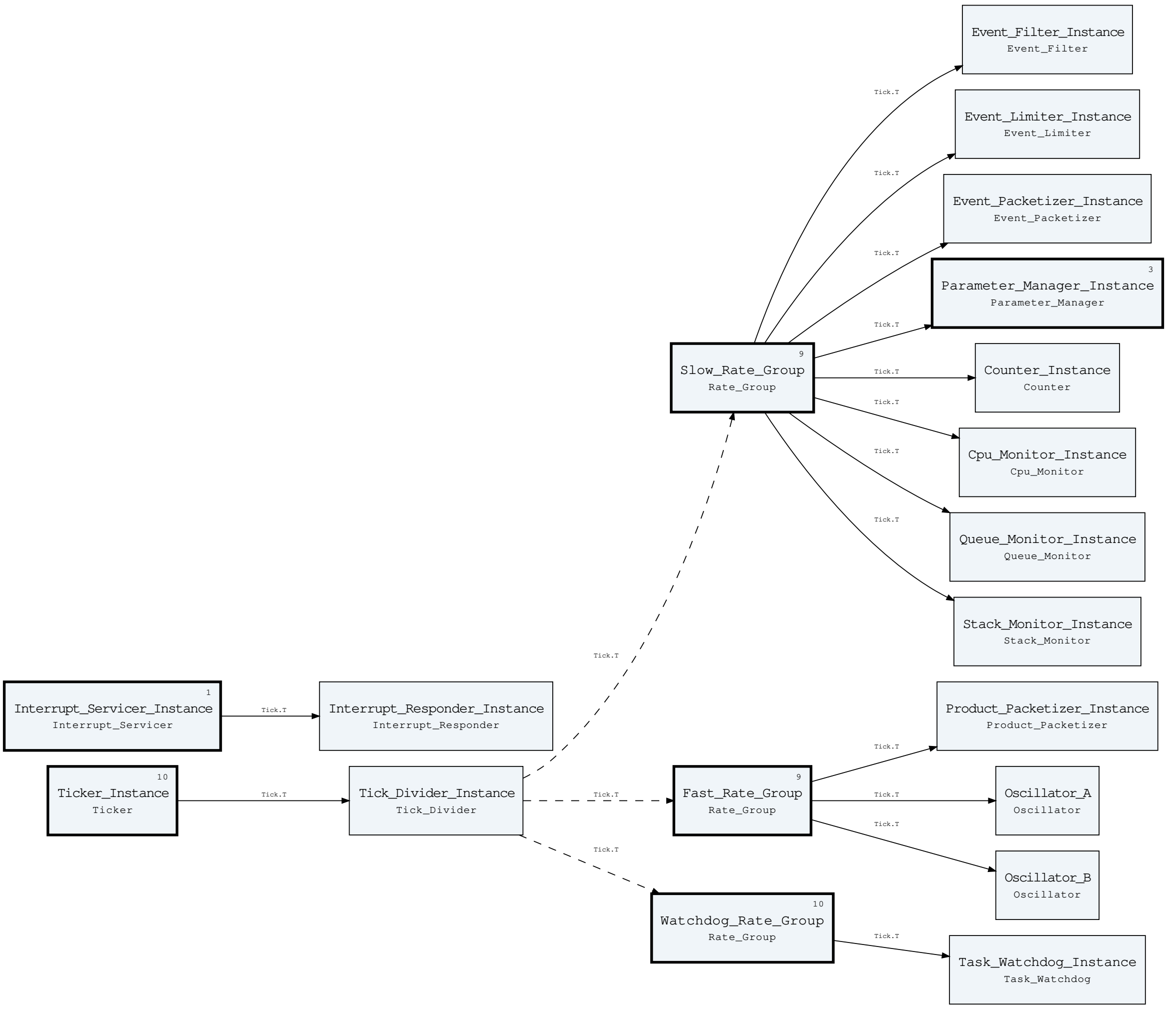

Figure 2: **Tick View:** In this assembly, Tick.T types are used to indicate when periodic or interrupt driven tasks should be run. This view shows the rate groups and interrupts that exist in the assembly. The Interrupt_Servicer_Instance is an active component that sits an waits, ready to execute when a specific interrupt is received. When this interrupt is received, it passes a tick to the Interrupt_Responder_Instance, which performs a task in response to the received interrupt. The Interrupt_Servicer_Instance will then immediately wait again for a subsequent interrupt. The Ticker_Instance runs in a high priority task and produces a Tick.T at 5 Hz. This tick is forwarded to the Tick_Divider_Instance which takes the 5 Hz signal and divides it up amongst 3 different rate groups. The Slow_Rate_Group runs at 0.5 Hz. The Fast_Rate_Group runs a 5 Hz. The Watchdog_Rate_Group runs as 1 Hz. Each of these rate group components will call the components connected to them every time they receive a tick from upstream. This architecture allows for periodic execution of certain functions, such as collecting telemetry or creating packets. Note that the actual execution of each rate group is also interleaved by the Ada scheduler. Each rate group is an active component, which executes on a task with a priority shown in the top right hand corner of the component. Rate groups produce telemetry relating how long it takes them to execute, both in CPU time and in wall clock time.

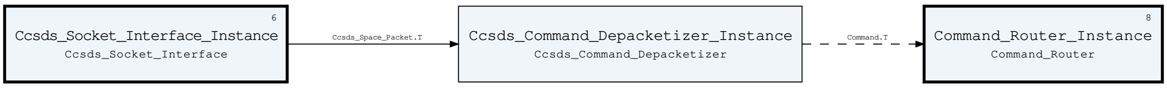
6
Ccsds_Socket_Interface_Instance
Ccsds_Socket_Interface
Ccsds_Space_Packet.T
Ccsds_Command_Depacketizer_Instance
Ccsds_Command_Depacketizer
Command.T
8
Command_Router_Instance
Command_Router

Figure 3: **Uplink View:** The Ccsds_Socket_Interface_Instance receives data from a ground system on a TCP socket. If the component receives a valid CCSDS packet it forwards it along to the Ccsds_Commmand_Depacketizer_Instance. This component looks for commands in the CCSDS packet, extracts them, and then forwards them along the Command_Router_Instance for routing and later execution.

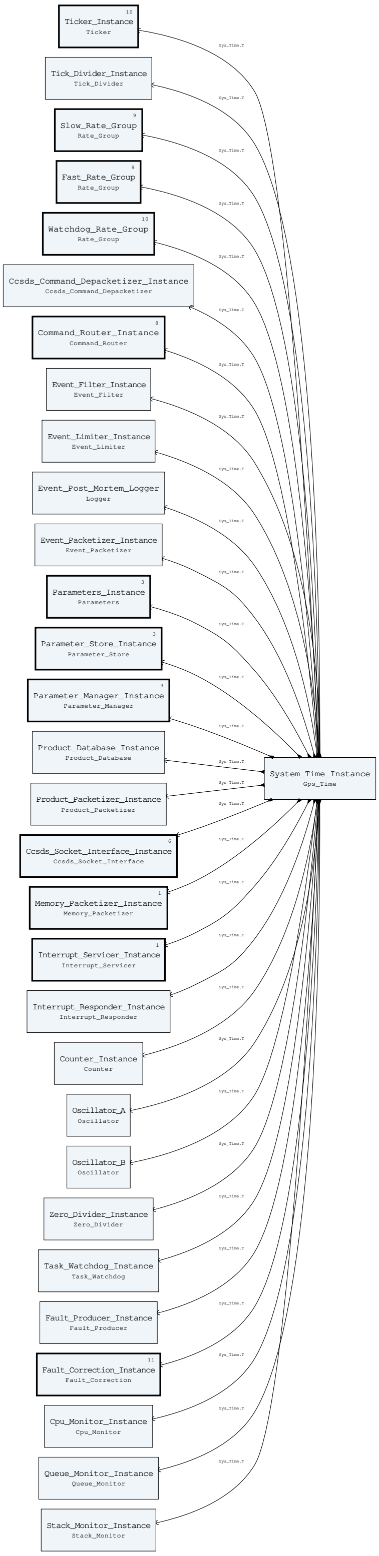

Figure 4: **Time View:** All components in the system fetch time by requesting it from the System_Time_Instance. By implementing the time system at the Adamant architectural level, different time sources and time synchronization schemes can be easily swapped into the system by replacing System_Time_Instance with a more tailored version. Note that synchronous connectors exhibit extremely low overhead, so exposing anything at the architectural level should not be prohibitive in terms of performance.

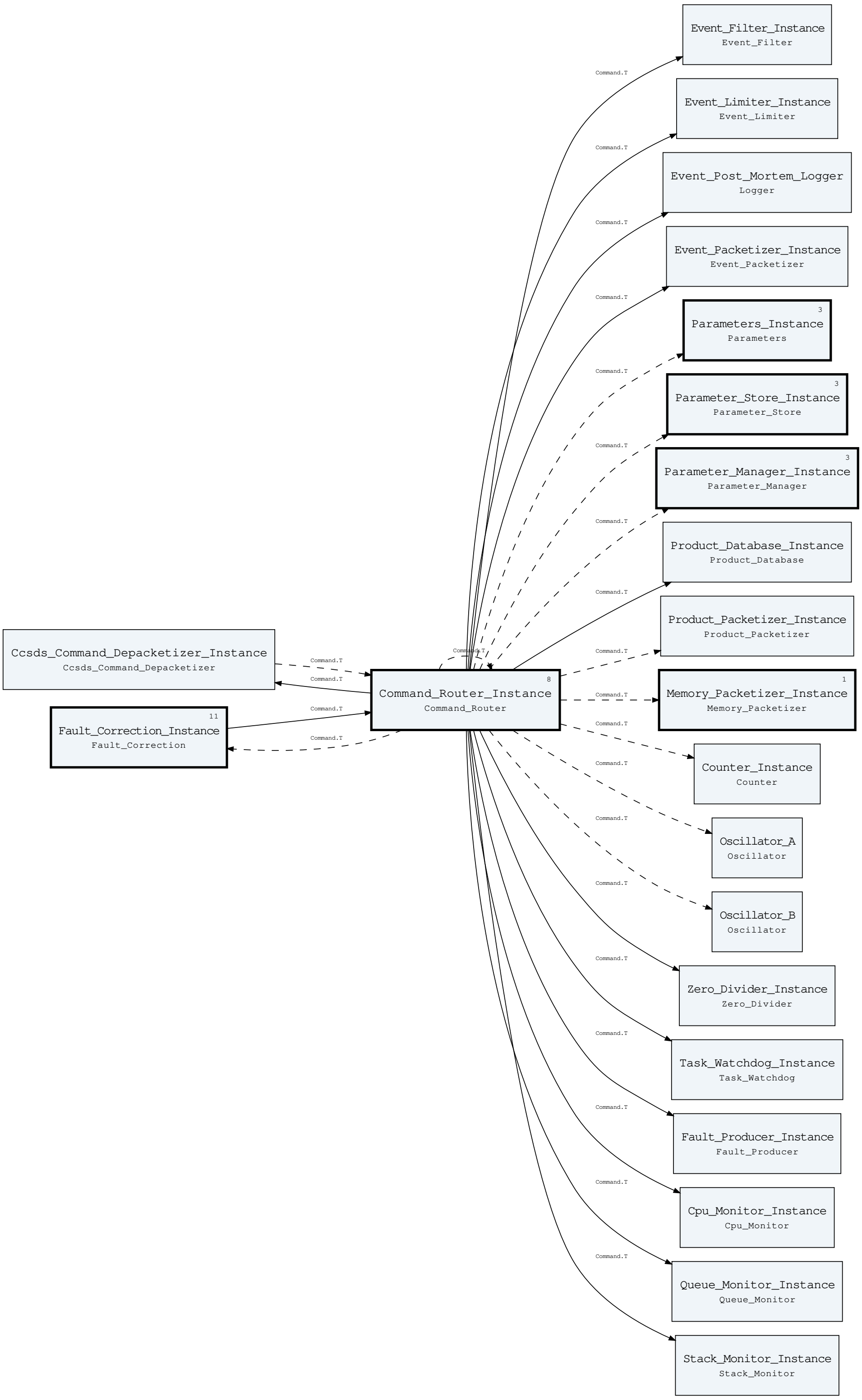

Figure 5: **Command View:** This view shows how commands are routed throughout the assembly. Commands are contained in the Command.T data type. Commands coming from the ground system originate from the Ccsds_Command_Depacketizer_Instance and then get passed the Command_Router_Instance. The router looks at the command's ID, determines the destination component for which the command is intended, and then forwards the command to that appropriate destination component. When a destination component receives a command, it will execute it and pass back a Command_Response.T data type back to the command router (shown in another view). Note that the Fault_Correction_Instance can also produce commands in order to correct a system fault. Commands from the Fault_Correction_Instance are passed to the router synchronously, bypassing the standard command queue that the Ccsds_Command_Depacketizer_Instance uses.

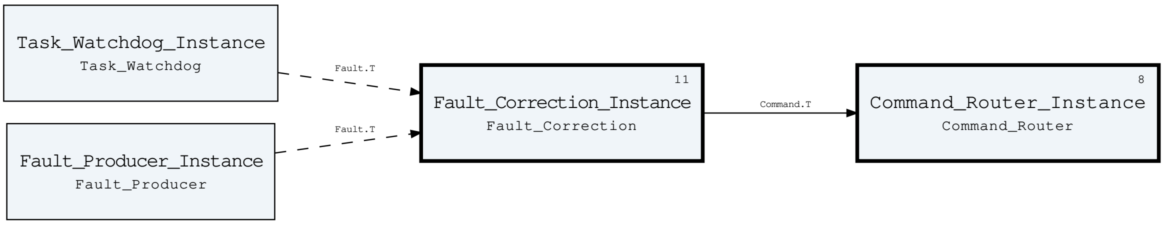

Figure 6: **Faults View:** Any component that produces faults will send them to the Fault_Correction_Instance. This component maps the fault to a response action. This action is in the form of a correction command which is sent to the Command_Router_Instance synchronously for execution. In the Linux assembly, two components can throw faults. The first is the Task_Watchdog_Instance. This component monitors some critical tasks to ensure that they are always running in a timely fashion. If one stops executing, a fault is thrown. The Fault_Producer_Instance is a simple component that throws a fault when commanded to. This can be used to inject a fault into the system for testing purposes.

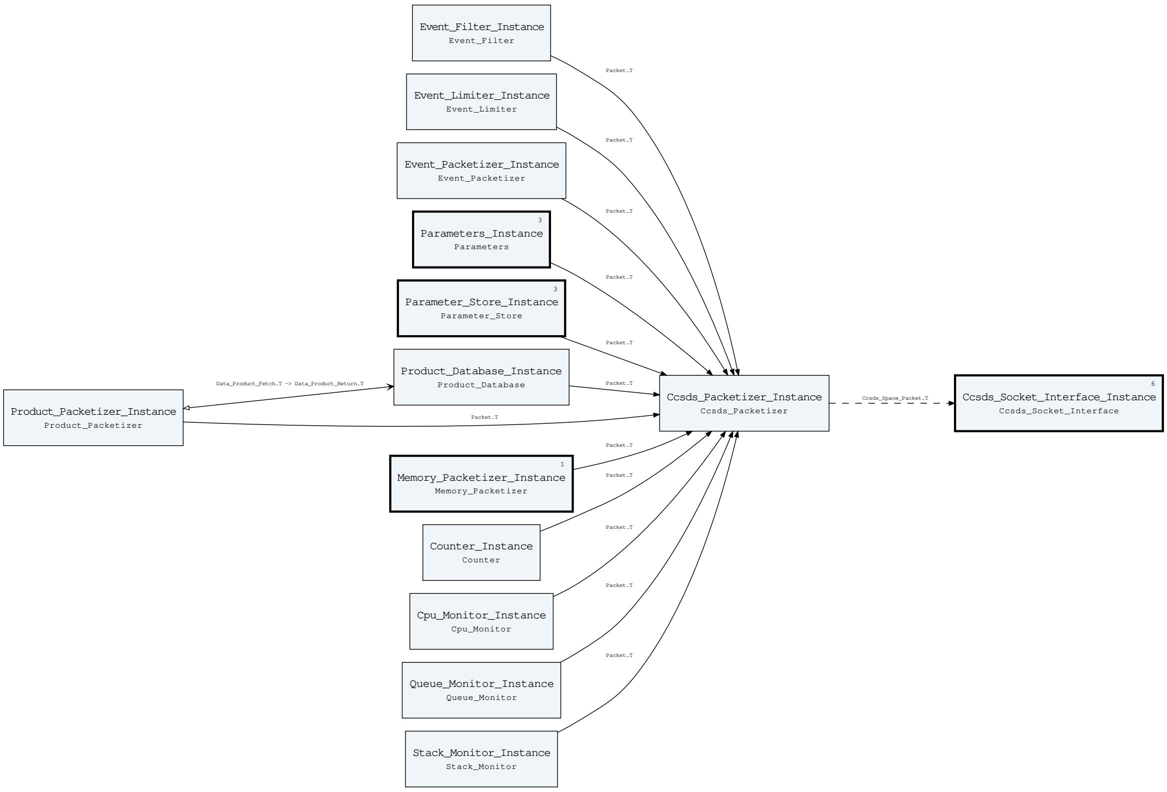

Figure 7: **Downlink View:** Any packet produced by the assembly is sent to the Ccsds_Packetizer_Instance. This component takes the Adamant formatted packets, converts them to CCSDS, and sends them asynchronously to the Ccsds_Socket_Interface_Instance. The socket interface component will take packets stored on its queue and transmit them over a network TCP socket. The Product_Packetizer_Instance periodically requests data products from the Product_Database_Instance to create packets containing data from multiple components. These packets are also forwarded to the Ccsds_Packetizer_Instance for downlink.

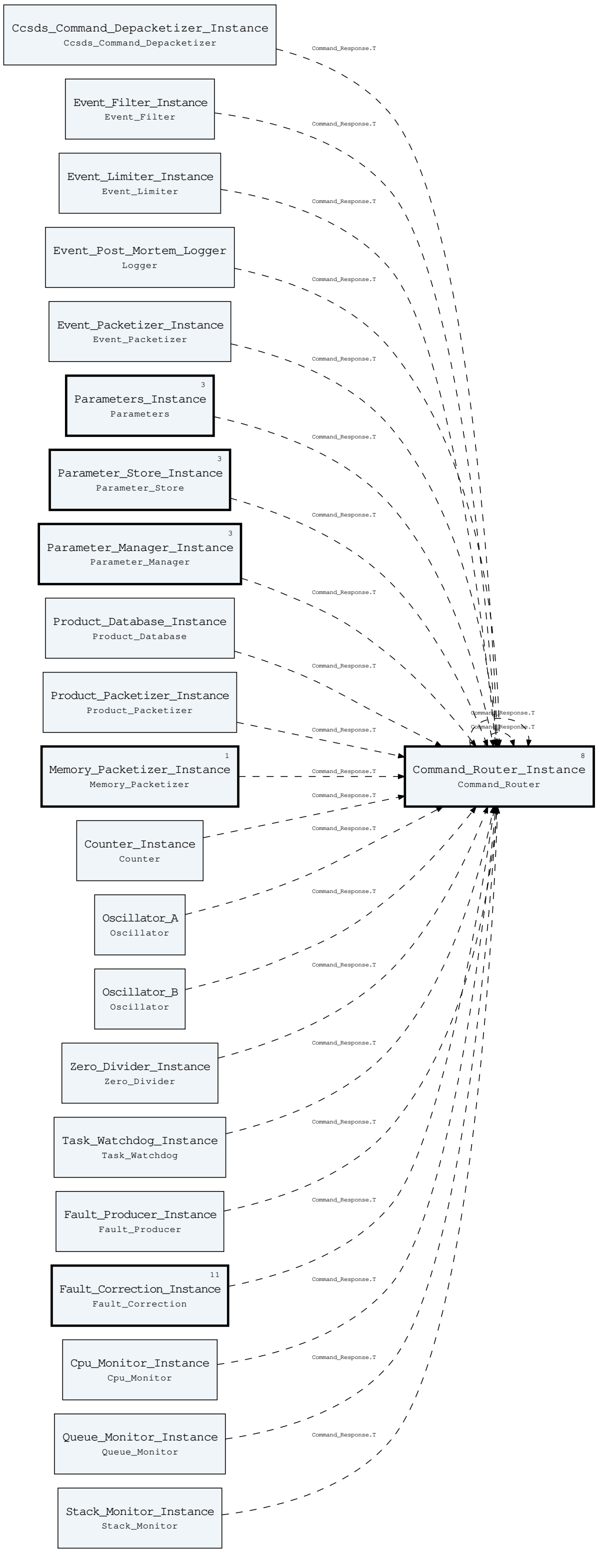

Figure 8: **Command Response View:** After a component executes a command received from the Command_Router_Instance, it passes back a Command_Response.T type to the router letting it know if the command succeeded or failed. This connection is also used to register all the commands with the router an initialization. This allows the Command_Router_Instance to create the internal routing table that it uses to route incoming commands.

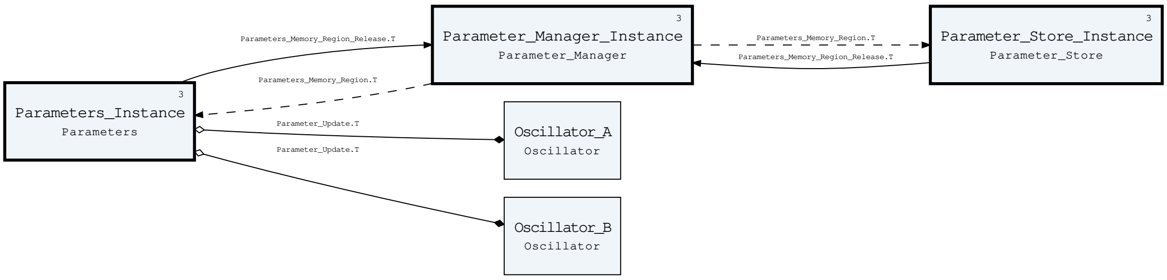

Figure 9: **Parameters View:** This view demonstrates an example usage of the Adamant parameter system. Any component that requires parameters (Oscillator_A and Oscillator_B) receives them from the Parameters_Instance. This component manages the updating and fetching of parameters from any parameter using components in the system. The Parameter_Store_Instance maintains a memory copy of the parameter table for the entire assembly. This table includes values for each individual parameter used by all the components in the system. In an embedded assembly, this parameter store would usually store the parameter table in a non-volatile memory region. The Parameter_Manager_Instance is responsible for managing commands to copy parameter tables between the Parameter_Store_Instance (usually manages what is called the default parameter table) and the Parameter_Instance (usually manages what is called the active parameter table). Any updates to the active parameter table will cause the Parameters_Instance to push individual parameter updates to the downstream components that need the updated parameters. Note - The current Linux assembly does not include code within the Parameter_Manager_Instance for uploading new parameter tables from a ground system yet.

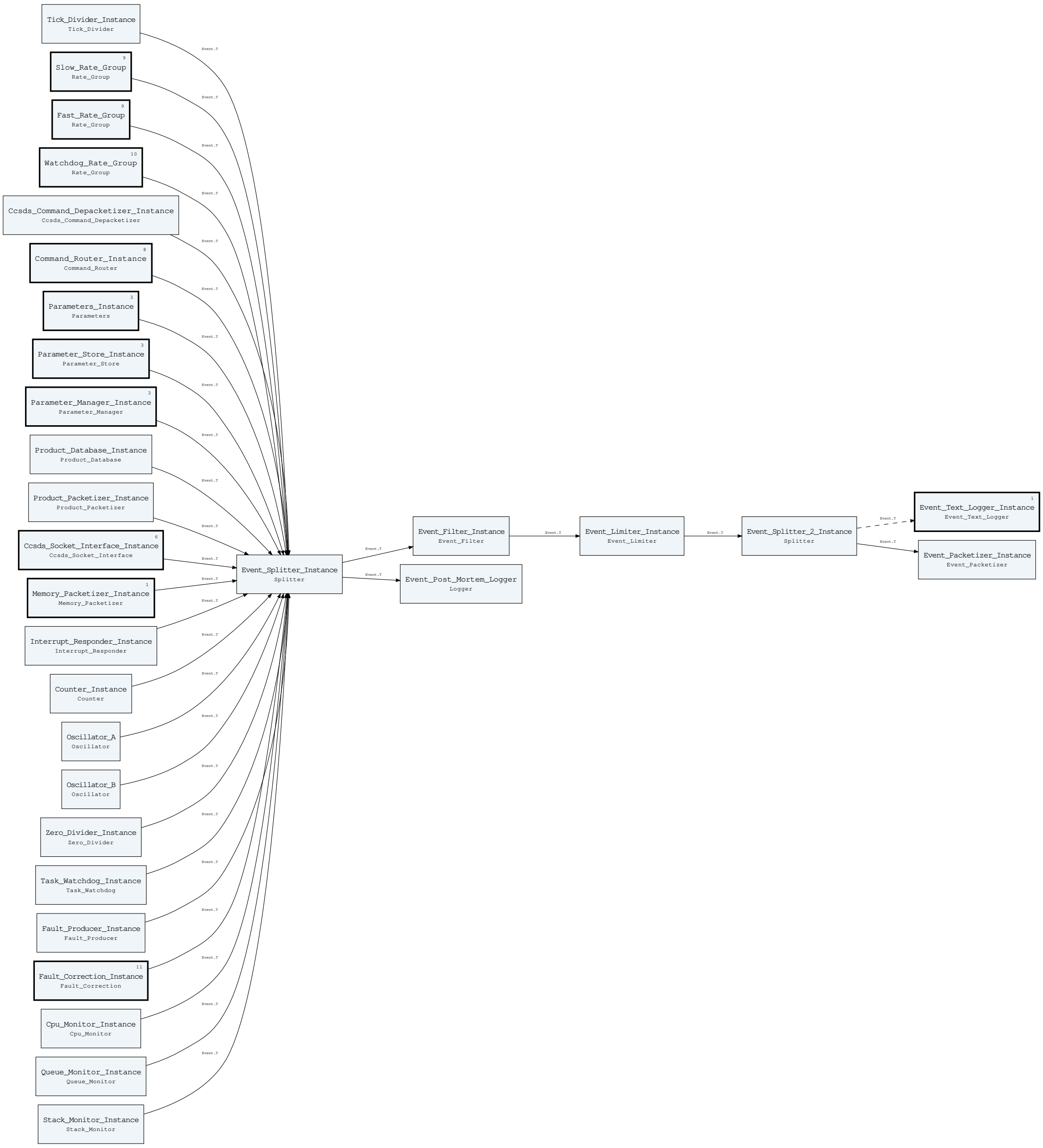

Figure 10: **Events View:** Events are produced by components when something interesting happens. All components send their events first to the Event_Splitter_Instance. This components duplicates the event and passes it to two destination components. This first is the Event_Post_Mortem_Logger, which stores the event into an memory store. On an embedded assembly, this would typically be in non-volatile memory. The event is also passed down a different path to the Event_Filter_Instance. This component can enabled/disable individual events by ID. Next, events are forwarded to the Event_Limiter_Instance. This component will start limiting problematic spam events that might flood the system. Any event that passes both of these filtering components is split by the Event_Splitter_2_Instance to two final destinations. The first is the Event_Text_Logger_Instance which prints out human readable events to the Linux terminal. The second is the Event_Packetizer_Instance which collects incoming events into packets for later downlink. This design allows for all events to be stored (unfiltered and unlimited) in the post mortem logger. A filtered and limited event stream is presented to both the Linux terminal and socket.

2.4 Task Priorities

The table below outlines the system tasks for the Linux Example assembly. Task names are in the form *component_ name.task_ name*. The priority *rank* is a number from 1 to *n* denoting how the priority of a component's task compares to others in the system. A rank of 1 is the highest priority in the system. The *priority value* is the actual priority number provided to the system scheduler. A larger *priority value* value signifies a higher priority task.

Table 3: Linux Example Component Task Priorities

Task Number	**Task Name**	**Priority Rank**	**Priority Value**
0	`Fault_Correction_Instance.Active_Task`	1	11
1	`Ticker_Instance.Active_Task`	2	10
2	`Watchdog_Rate_Group.Active_Task`	2	10
3	`Slow_Rate_Group.Active_Task`	3	9
4	`Fast_Rate_Group.Active_Task`	3	9
5	`Command_Router_Instance.Active_Task`	4	8
6	`Ccsds_Socket_Interface_Instance.Active_Task`	5	6
7	`Parameters_Instance.Active_Task`	6	3
8	`Parameter_Store_Instance.Active_Task`	6	3
9	`Parameter_Manager_Instance.Active_Task`	6	3
10	`Event_Text_Logger_Instance.Active_Task`	7	1
11	`Memory_Packetizer_Instance.Active_Task`	7	1
12	`Interrupt_Servicer_Instance.Active_Task`	7	1
13	`Ccsds_Socket_Interface_Instance.Listener_Task`	8	0

2.5 Commands

The table below shows the commands for the Linux Example assembly.

Table 4: Linux Example Commands

Command ID	**Command Name**	**Argument Type**

0x0001 (1)	Ccsds_Command_Depacketizer_Instance.Reset_Counts	-
0x0002 (2)	Command_Router_Instance.Noop	-
0x0003 (3)	Command_Router_Instance.Noop_Arg	Command_Router_Arg.T
0x0004 (4)	Command_Router_Instance.Noop_Response	-
0x0005 (5)	Command_Router_Instance.Reset_Data_Products	-
0x0006 (6)	Event_Filter_Instance.Filter_Event	Event_Filter_Single_Event_Cmd_Type.T
0x0007 (7)	Event_Filter_Instance.Unfilter_Event	Event_Filter_Single_Event_Cmd_Type.T
0x0008 (8)	Event_Filter_Instance.Filter_Event_Range	Event_Filter_Id_Range.T
0x0009 (9)	Event_Filter_Instance.Unfilter_Event_Range	Event_Filter_Id_Range.T
0x000a (10)	Event_Filter_Instance.Enable_Event_Filtering	-
0x000b (11)	Event_Filter_Instance.Disable_Event_Filtering	-
0x000c (12)	Event_Filter_Instance.Dump_Event_States	-
0x000d (13)	Event_Limiter_Instance.Enable_Event_Limit	Event_Single_State_Cmd_Type.T
0x000e (14)	Event_Limiter_Instance.Disable_Event_Limit	Event_Single_State_Cmd_Type.T
0x000f (15)	Event_Limiter_Instance.Enable_Event_Limit_Range	Event_Limiter_Id_Range.T
0x0010 (16)	Event_Limiter_Instance.Disable_Event_Limit_Range	Event_Limiter_Id_Range.T
0x0011 (17)	Event_Limiter_Instance.Enable_Event_Limiting	-

0x0012 (18)	Event_Limiter_Instance.Disable_Event_Limiting	-
0x0013 (19)	Event_Limiter_Instance.Set_Event_Limit_Persistence	Event_Limiter_Persistence_Type.T
0x0014 (20)	Event_Limiter_Instance.Dump_Event_States	-
0x0015 (21)	Event_Post_Mortem_Logger.Enable	-
0x0016 (22)	Event_Post_Mortem_Logger.Disable	-
0x0017 (23)	Event_Post_Mortem_Logger.Dump_Log	-
0x0018 (24)	Event_Post_Mortem_Logger.Dump_Newest_Data	Packed_Positive_Length.T
0x0019 (25)	Event_Post_Mortem_Logger.Dump_Oldest_Data	Packed_Positive_Length.T
0x001a (26)	Event_Post_Mortem_Logger.Dump_Log_Memory	-
0x001b (27)	Event_Post_Mortem_Logger.Send_Meta_Data_Event	-
0x001c (28)	Event_Packetizer_Instance.Send_Packet	-
0x001d (29)	Parameters_Instance.Update_Parameter	Parameter.T
0x001e (30)	Parameters_Instance.Dump_Parameters	-
0x001f (31)	Parameter_Store_Instance.Dump_Parameter_Store	-
0x0020 (32)	Product_Database_Instance.Clear_Override	Data_Product_Id.T
0x0021 (33)	Product_Database_Instance.Clear_Override_For_All	-
0x0022 (34)	Product_Database_Instance.Override	Data_Product.T

0x0023 (35)	Product_Database_Instance.Dump	Data_Product_Id.T
0x0024 (36)	Product_Database_Instance.Dump_Poly_Type	Data_Product_Poly_Extract.T
0x0025 (37)	Product_Packetizer_Instance.Set_Packet_Period	Packet_Period.T
0x0026 (38)	Product_Packetizer_Instance.Enable_Packet	Packet_Id.T
0x0027 (39)	Product_Packetizer_Instance.Disable_Packet	Packet_Id.T
0x0028 (40)	Product_Packetizer_Instance.Send_Packet	Packet_Id.T
0x0029 (41)	Memory_Packetizer_Instance.Set_Max_Packet_Rate	Packets_Per_Period.T
0x002a (42)	Oscillator_A.Set_Frequency	Packed_F32.T
0x002b (43)	Oscillator_A.Set_Amplitude	Packed_F32.T
0x002c (44)	Oscillator_A.Set_Offset	Packed_F32.T
0x002d (45)	Oscillator_B.Set_Frequency	Packed_F32.T
0x002e (46)	Oscillator_B.Set_Amplitude	Packed_F32.T
0x002f (47)	Oscillator_B.Set_Offset	Packed_F32.T
0x0030 (48)	Zero_Divider_Instance.Divide_By_Zero	Packed_U32.T
0x0031 (49)	Task_Watchdog_Instance.Enable_Watchdog_Pet_Checks	-
0x0032 (50)	Task_Watchdog_Instance.Disable_Watchdog_Pet_Checks	-
0x0033 (51)	Task_Watchdog_Instance.Set_Watchdog_Limit	Linux_Example_Task_Watchdog_List_Watchdog_Limit_Cmd.T
0x0034 (52)	Task_Watchdog_Instance.Set_Watchdog_Action	Linux_Example_Task_Watchdog_List_Watchdog_Action_Cmd.T

0x0035 (53)	Fault_Producer_Instance.Throw_Fault_1	-
0x0036 (54)	Fault_Producer_Instance.Throw_Fault_2	-
0x0037 (55)	Fault_Correction_Instance.Enable_Fault_Response	Linux_Example_Fault_Responses_Packed_Id_Type.T
0x0038 (56)	Fault_Correction_Instance.Disable_Fault_Response	Linux_Example_Fault_Responses_Packed_Id_Type.T
0x0039 (57)	Fault_Correction_Instance.Clear_Fault_Response	Linux_Example_Fault_Responses_Packed_Id_Type.T
0x003a (58)	Fault_Correction_Instance.Clear_All_Fault_Responses	-
0x003b (59)	Fault_Correction_Instance.Reset_Data_Products	-
0x003c (60)	Cpu_Monitor_Instance.Set_Packet_Period	Packed_U16.T
0x003d (61)	Queue_Monitor_Instance.Set_Packet_Period	Packed_U16.T
0x003e (62)	Stack_Monitor_Instance.Set_Packet_Period	Packed_U16.T
0x0063 (99)	Counter_Instance.Set_Count	Packed_U32.T
0x0064 (100)	Counter_Instance.Reset_Count	-
0x0065 (101)	Counter_Instance.Set_Count_Add	Operands.T
0x1450 (5200)	Parameter_Manager_Instance.Copy_Parameter_Table	Packed_Parameter_Table_Copy_Type.T

Command Descriptions:

- **Ccsds_Command_Depacketizer_Instance.Reset_Counts** - This command resets the internal counts for the data products.
- **Command_Router_Instance.Noop** - Simple NOOP command which produces an event saying that it was triggered. This can be used to self test the command routing system and verify system aliveness.
- **Command_Router_Instance.Noop_Arg** - Simple NOOP command which produces an event saying that it was triggered with a certain Arg. This can be used to self test the command argu-

ment validation system. Sending a command with an Arg value of 868 will cause the component to Fail the command. Any other value will produce a successfully executed command.

- **`Command_Router_Instance.Noop_Response`** - A NOOP command which self tests the command response forwarding mechanism. The command handler itself acts as a command sender component, and sends out a NOOP command with a registered Source Id. The Command Router should then send out an event saying that the command response was forwarded and received again by the Command Router.
- **`Command_Router_Instance.Reset_Data_Products`** - This command resets the values of all the component's data product to the values at initialization.
- **`Event_Filter_Instance.Filter_Event`** - Enable the event filtering for a specific event ID.
- **`Event_Filter_Instance.Unfilter_Event`** - Disable the event filtering for a specific event ID.
- **`Event_Filter_Instance.Filter_Event_Range`** - Enable the event filtering for a specific range of event IDs.
- **`Event_Filter_Instance.Unfilter_Event_Range`** - Disable the event filtering for a specific range of event IDs.
- **`Event_Filter_Instance.Enable_Event_Filtering`** - Enable the component to filter events that have been set to be filtered.
- **`Event_Filter_Instance.Disable_Event_Filtering`** - Disable the component so that all events will not be filtered. The event states will be maintained for when re-enabled.
- **`Event_Filter_Instance.Dump_Event_States`** - Dump a packet for the state of all events pertaining to if they are filtered or not.
- **`Event_Limiter_Instance.Enable_Event_Limit`** - Enable the event limiter for a specific event ID.
- **`Event_Limiter_Instance.Disable_Event_Limit`** - Disable the event limiter for a specific event ID.
- **`Event_Limiter_Instance.Enable_Event_Limit_Range`** - Enable the event limiter for a specific range of event IDs.
- **`Event_Limiter_Instance.Disable_Event_Limit_Range`** - Disable the event limiter for a specific range of event IDs.
- **`Event_Limiter_Instance.Enable_Event_Limiting`** - Enable the component to limit events that have been set to be limited.
- **`Event_Limiter_Instance.Disable_Event_Limiting`** - Disable the component so that all events will not be limited. The event states and counts will be maintained for when re-enabled.
- **`Event_Limiter_Instance.Set_Event_Limit_Persistence`** - Change the persistence of the event limiter for all events that are limited. Value must be between 1 and 7.
- **`Event_Limiter_Instance.Dump_Event_States`** - Dump a packet for the state of all events on if they are limited or not.
- **`Event_Post_Mortem_Logger.Enable`** - Enable the logger to start saving data.
- **`Event_Post_Mortem_Logger.Disable`** - Disable the logger from saving received data.
- **`Event_Post_Mortem_Logger.Dump_Log`** - Dump the entire log oldest to newest data.
- **`Event_Post_Mortem_Logger.Dump_Newest_Data`** - Dump the newest X bytes of data from the log.
- **`Event_Post_Mortem_Logger.Dump_Oldest_Data`** - Dump the oldest X bytes of data from the log.

- **Event_Post_Mortem_Logger.Dump_Log_Memory** - Dump the entire region of memory associated with the logger from start to finish in memory byte order.
- **Event_Post_Mortem_Logger.Send_Meta_Data_Event** - Send an event out with the meta data of the log.
- **Event_Packetizer_Instance.Send_Packet** - Send a packet out next tick, unless there are no events stored within the component.
- **Parameters_Instance.Update_Parameter** - Update the active parameter value in a component for a parameter with the given ID, Length, and Value.
- **Parameters_Instance.Dump_Parameters** - Produce a packet with the currently staged parameter values contained within connected components.
- **Parameter_Store_Instance.Dump_Parameter_Store** - Produce a packet with the currently stored parameter values.
- **Product_Database_Instance.Clear_Override** - Clear the override condition for the data product of the provided ID.
- **Product_Database_Instance.Clear_Override_For_All** - Clear the override condition for all data products in the product store.
- **Product_Database_Instance.Override** - Override the value of a data product in the data product store. The value of this data product will be fixed to the commanded value, ignoring all other updates, until the override is cleared.
- **Product_Database_Instance.Dump** - Dump the data product of the provided ID in a packet.
- **Product_Database_Instance.Dump_Poly_Type** - Dump the data product of the provided ID into a poly type based on the provided offset and length.
- **Product_Packetizer_Instance.Set_Packet_Period** - Command to change the period of packet generation for a given packet id.
- **Product_Packetizer_Instance.Enable_Packet** - Command to enable the emission of a packet from the packetizer.
- **Product_Packetizer_Instance.Disable_Packet** - Command to disable the emission of a packet from the packetizer.
- **Product_Packetizer_Instance.Send_Packet** - Command to build specific packet and send it out on the next available tick. The packet is built and sent regardless of the packet being enabled or disabled.
- **Memory_Packetizer_Instance.Set_Max_Packet_Rate** - Set a new value for the max_Packets_Per_Time_Period and the time_Period_In_Seconds to control the output rate of the emitted packets.
- **Oscillator_A.Set_Frequency** - Set the frequency of the oscillator in Hz
- **Oscillator_A.Set_Amplitude** - Set the amplitude of the oscillator
- **Oscillator_A.Set_Offset** - Set the Y offset of the oscillator
- **Oscillator_B.Set_Frequency** - Set the frequency of the oscillator in Hz
- **Oscillator_B.Set_Amplitude** - Set the amplitude of the oscillator
- **Oscillator_B.Set_Offset** - Set the Y offset of the oscillator
- **Zero_Divider_Instance.Divide_By_Zero** - You must provide the correct magic number as argument to this command for it to be executed.
- **Task_Watchdog_Instance.Enable_Watchdog_Pet_Checks** - Command to enable the watchdog component to check all connected components for incoming pets.
- **Task_Watchdog_Instance.Disable_Watchdog_Pet_Checks** - Command to disable the watchdog component to check all connected components for incoming pets.

- **Task_Watchdog_Instance.Set_Watchdog_Limit** - Set the limit value for the watchdog given an index and the new index value.
- **Task_Watchdog_Instance.Set_Watchdog_Action** - Sets the action of a petter given the index of that petter and the updated action. Note that actions cannot be promoted to fault if they were not provided a fault id.
- **Fault_Producer_Instance.Throw_Fault_1** - Throw the first fault.
- **Fault_Producer_Instance.Throw_Fault_2** - Throw the second fault.
- **Fault_Correction_Instance.Enable_Fault_Response** - Enable a fault response for the provided ID. This will only succeed if another response with the same Fault ID is not already enabled.
- **Fault_Correction_Instance.Disable_Fault_Response** - Disable a fault response for the provided ID.
- **Fault_Correction_Instance.Clear_Fault_Response** - Resets a fault response to the Enabled state of the provided ID. If the fault is latched, it unlatches the fault.
- **Fault_Correction_Instance.Clear_All_Fault_Responses** - Resets all fault responses to the Enabled state. Unlatches all latched fault responses.
- **Fault_Correction_Instance.Reset_Data_Products** - This command resets the values of all the component's data product to the values at initialization, except for the Fault_Response_Statuses data product which can be reset by the Clear_All_Fault_Responses command.
- **Cpu_Monitor_Instance.Set_Packet_Period** - Set the period of the packet. A period of zero disables the sending of the packet.
- **Queue_Monitor_Instance.Set_Packet_Period** - Set the period of the packet. A period of zero disables the sending of the packet.
- **Stack_Monitor_Instance.Set_Packet_Period** - Set the period of the packet. A period of zero disables the sending of the packet.
- **Counter_Instance.Set_Count** - Change the current counter value in the counter component
- **Counter_Instance.Reset_Count** - Reset the current counter value in the counter component to zero
- **Counter_Instance.Set_Count_Add** - Change the current counter value in the counter component to the sum of the arguments
- **Parameter_Manager_Instance.Copy_Parameter_Table** - Copy parameter table from source to destination based on the enumeration provided.

2.6 Parameters

The table below shows the parameters for the Linux Example assembly.

Table 5: Linux Example Parameters

Parameter ID	Parameter Name	Type	Default Value
0x0001 (1)	Oscillator_A.Frequency	Packed_F32.T	(Value=>0.175)
0x0002 (2)	Oscillator_A.Amplitude	Packed_F32.T	(Value=>5.0)

0x0003 (3)	Oscillator_A. Offset	Packed_F32.T	(Value=>0.0)
0x0004 (4)	Oscillator_B. Frequency	Packed_F32.T	(Value=>0.175)
0x0005 (5)	Oscillator_B. Amplitude	Packed_F32.T	(Value=>5.0)
0x0006 (6)	Oscillator_B. Offset	Packed_F32.T	(Value=>0.0)

Parameter Descriptions:

- **Oscillator_A.Frequency** - The frequency of the oscillator in Hz
- **Oscillator_A.Amplitude** - The amplitude of the oscillator
- **Oscillator_A.Offset** - The Y offset of the oscillator
- **Oscillator_B.Frequency** - The frequency of the oscillator in Hz
- **Oscillator_B.Amplitude** - The amplitude of the oscillator
- **Oscillator_B.Offset** - The Y offset of the oscillator

2.7 Events

The table below shows the events for the Linux Example assembly.

Table 6: Linux Example Events

Event ID	**Event Name**	**Parameter Type**
0x0001 (1)	Tick_Divider_Instance. Component_Has_Full_Queue	Td_Full_Queue_Param.T
0x0002 (2)	Slow_Rate_Group.Cycle_Slip	Cycle_Slip_Param.T
0x0003 (3)	Slow_Rate_Group.Max_Cycle_ Time_Exceeded	Time_Exceeded.T
0x0004 (4)	Slow_Rate_Group.Max_ Execution_Time_Exceeded	Time_Exceeded.T
0x0005 (5)	Slow_Rate_Group.Component_ Has_Full_Queue	Full_Queue_Param.T
0x0006 (6)	Slow_Rate_Group.Incoming_ Tick_Dropped	Tick.T
0x0007 (7)	Fast_Rate_Group.Cycle_Slip	Cycle_Slip_Param.T

0x0008 (8)	Fast_Rate_Group.Max_Cycle_Time_Exceeded	Time_Exceeded.T
0x0009 (9)	Fast_Rate_Group.Max_Execution_Time_Exceeded	Time_Exceeded.T
0x000a (10)	Fast_Rate_Group.Component_Has_Full_Queue	Full_Queue_Param.T
0x000b (11)	Fast_Rate_Group.Incoming_Tick_Dropped	Tick.T
0x000c (12)	Watchdog_Rate_Group.Cycle_Slip	Cycle_Slip_Param.T
0x000d (13)	Watchdog_Rate_Group.Max_Cycle_Time_Exceeded	Time_Exceeded.T
0x000e (14)	Watchdog_Rate_Group.Max_Execution_Time_Exceeded	Time_Exceeded.T
0x000f (15)	Watchdog_Rate_Group.Component_Has_Full_Queue	Full_Queue_Param.T
0x0010 (16)	Watchdog_Rate_Group.Incoming_Tick_Dropped	Tick.T
0x0011 (17)	Ccsds_Command_Depacketizer_Instance.Invalid_Packet_Checksum	Invalid_Packet_Xor8_Info.T
0x0012 (18)	Ccsds_Command_Depacketizer_Instance.Invalid_Packet_Type	Ccsds_Primary_Header.T
0x0013 (19)	Ccsds_Command_Depacketizer_Instance.Packet_Too_Small	Invalid_Packet_Length.T
0x0014 (20)	Ccsds_Command_Depacketizer_Instance.Packet_Too_Large	Invalid_Packet_Length.T
0x0015 (21)	Ccsds_Command_Depacketizer_Instance.No_Secondary_Header	Ccsds_Primary_Header.T
0x0016 (22)	Ccsds_Command_Depacketizer_Instance.Counts_Reset	-
0x0017 (23)	Ccsds_Command_Depacketizer_Instance.Invalid_Command_Received	Invalid_Command_Info.T

0x0018 (24)	Command_Router_Instance.Command_Received	Command_Header.T
0x0019 (25)	Command_Router_Instance.Command_Execution_Successful	Command_Response.T
0x001a (26)	Command_Router_Instance.Command_Execution_Failure	Command_Response.T
0x001b (27)	Command_Router_Instance.Command_Id_Not_Registered	Command_Header.T
0x001c (28)	Command_Router_Instance.Registration_Id_Conflict	Command_Id.T
0x001d (29)	Command_Router_Instance.Router_Table_Full	Command_Id.T
0x001e (30)	Command_Router_Instance.Outgoing_Command_Dropped	Command_Header.T
0x001f (31)	Command_Router_Instance.Incoming_Command_Dropped	Command_Header.T
0x0020 (32)	Command_Router_Instance.Noop_Command_Dropped	Command_Header.T
0x0021 (33)	Command_Router_Instance.Command_Response_Dropped	Command_Response.T
0x0022 (34)	Command_Router_Instance.Noop_Received	-
0x0023 (35)	Command_Router_Instance.Noop_Arg_Received	Command_Router_Arg.T
0x0024 (36)	Command_Router_Instance.Noop_Response_Received	-
0x0025 (37)	Command_Router_Instance.Noop_Response_Forwarding_Success	Command_Response.T
0x0026 (38)	Command_Router_Instance.Forwarded_Command_Response_Dropped	Command_Response.T
0x0027 (39)	Command_Router_Instance.Invalid_Command_Source_Id	Command_Response.T

0x0028 (40)	Command_Router_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x0029 (41)	Command_Router_Instance.Data_Products_Reset	-
0x002a (42)	Event_Filter_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x002b (43)	Event_Filter_Instance.Filtered_Event	Event_Filter_Single_Event_Cmd_Type.T
0x002c (44)	Event_Filter_Instance.Unfiltered_Event	Event_Filter_Single_Event_Cmd_Type.T
0x002d (45)	Event_Filter_Instance.Filtered_Event_Range	Event_Filter_Id_Range.T
0x002e (46)	Event_Filter_Instance.Unfiltered_Event_Range	Event_Filter_Id_Range.T
0x002f (47)	Event_Filter_Instance.Enable_Event_Filter	-
0x0030 (48)	Event_Filter_Instance.Disable_Event_Filter	-
0x0031 (49)	Event_Filter_Instance.Filter_Event_Invalid_Id	Event_Filter_Single_Event_Cmd_Type.T
0x0032 (50)	Event_Filter_Instance.Unfilter_Event_Invalid_Id	Event_Filter_Single_Event_Cmd_Type.T
0x0033 (51)	Event_Filter_Instance.Filter_Event_Range_Invalid_Id	Event_Filter_Id_Range.T
0x0034 (52)	Event_Filter_Instance.Unfilter_Event_Range_Invalid_Id	Event_Filter_Id_Range.T
0x0035 (53)	Event_Filter_Instance.Dump_Event_States_Recieved	-
0x0036 (54)	Event_Limiter_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x0037 (55)	Event_Limiter_Instance.Events_Limited_Since_Last_Tick	Event_Limiter_Num_Events_Type.T

0x0038 (56)	Event_Limiter_Instance.Event_Limit_Enabled	Event_Single_State_Cmd_Type.T
0x0039 (57)	Event_Limiter_Instance.Event_Limit_Disabled	Event_Single_State_Cmd_Type.T
0x003a (58)	Event_Limiter_Instance.Event_Limit_Range_Enabled	Event_Limiter_Id_Range.T
0x003b (59)	Event_Limiter_Instance.Event_Limit_Range_Disabled	Event_Limiter_Id_Range.T
0x003c (60)	Event_Limiter_Instance.Event_Limiting_Enabled	-
0x003d (61)	Event_Limiter_Instance.Event_Limiting_Disabled	-
0x003e (62)	Event_Limiter_Instance.Event_Limit_Enable_Invalid_Id	Event_Single_State_Cmd_Type.T
0x003f (63)	Event_Limiter_Instance.Event_Limit_Disable_Invalid_Id	Event_Single_State_Cmd_Type.T
0x0040 (64)	Event_Limiter_Instance.Event_Limit_Range_Enabled_Invalid_Id	Event_Limiter_Id_Range.T
0x0041 (65)	Event_Limiter_Instance.Event_Limit_Range_Disabled_Invalid_Id	Event_Limiter_Id_Range.T
0x0042 (66)	Event_Limiter_Instance.Set_New_Persistence	Event_Limiter_Persistence_Type.T
0x0043 (67)	Event_Limiter_Instance.Dump_Event_States_Recieved	-
0x0044 (68)	Event_Post_Mortem_Logger.Log_Attempt_Failed	Logger_Error.T
0x0045 (69)	Event_Post_Mortem_Logger.Log_Disabled	Circular_Buffer_Meta.T
0x0046 (70)	Event_Post_Mortem_Logger.Log_Enabled	-
0x0047 (71)	Event_Post_Mortem_Logger.Log_Info_Update	Logger_Info.T

0x0048 (72)	Event_Post_Mortem_Logger.Dumping_Log_Memory	Memory_Region.T
0x0049 (73)	Event_Post_Mortem_Logger.Invalid_Command_Received	Invalid_Command_Info.T
0x004a (74)	Parameters_Instance.Parameter_Update_Success	Parameter_Id.T
0x004b (75)	Parameters_Instance.Parameter_Update_Id_Not_Recognized	Parameter_Id.T
0x004c (76)	Parameters_Instance.Parameter_Stage_Failed	Parameter_Operation_Status.T
0x004d (77)	Parameters_Instance.Parameter_Update_Failed	Parameter_Operation_Status.T
0x004e (78)	Parameters_Instance.Parameter_Fetch_Failed	Parameter_Operation_Status.T
0x004f (79)	Parameters_Instance.Parameter_Fetch_Length_Mismatch	Invalid_Parameter_Length.T
0x0050 (80)	Parameters_Instance.Parameter_Update_Length_Mismatch	Invalid_Parameter_Length.T
0x0051 (81)	Parameters_Instance.Memory_Region_Length_Mismatch	Invalid_Parameters_Memory_Region_Length.T
0x0052 (82)	Parameters_Instance.Memory_Region_Crc_Invalid	Invalid_Parameters_Memory_Region_Crc.T
0x0053 (83)	Parameters_Instance.Dumping_Parameters	-
0x0054 (84)	Parameters_Instance.Finished_Dumping_Parameters	-
0x0055 (85)	Parameters_Instance.Starting_Parameter_Table_Update	Memory_Region.T
0x0056 (86)	Parameters_Instance.Finished_Parameter_Table_Update	Parameters_Memory_Region_Release.T

0x0057 (87)	Parameters_Instance.Starting_Parameter_Table_Fetch	Memory_Region.T
0x0058 (88)	Parameters_Instance.Finished_Parameter_Table_Fetch	Parameters_Memory_Region_Release.T
0x0059 (89)	Parameters_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x005a (90)	Parameters_Instance.Command_Dropped	Command_Header.T
0x005b (91)	Parameters_Instance.Memory_Region_Dropped	Parameters_Memory_Region.T
0x005c (92)	Parameter_Store_Instance.Memory_Region_Length_Mismatch	Invalid_Parameters_Memory_Region_Length.T
0x005d (93)	Parameter_Store_Instance.Memory_Region_Crc_Invalid	Invalid_Parameters_Memory_Region_Crc.T
0x005e (94)	Parameter_Store_Instance.Dumped_Parameters	-
0x005f (95)	Parameter_Store_Instance.Parameter_Table_Updated	Memory_Region.T
0x0060 (96)	Parameter_Store_Instance.Parameter_Table_Fetched	Memory_Region.T
0x0061 (97)	Parameter_Store_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x0062 (98)	Parameter_Store_Instance.Command_Dropped	Command_Header.T
0x0063 (99)	Parameter_Store_Instance.Memory_Region_Dropped	Parameters_Memory_Region.T
0x0064 (100)	Parameter_Manager_Instance.Starting_Parameter_Table_Copy	Packed_Parameter_Table_Copy_Type.T
0x0065 (101)	Parameter_Manager_Instance.Finished_Parameter_Table_Copy	Packed_Parameter_Table_Copy_Type.T

0x0066 (102)	Parameter_Manager_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x0067 (103)	Parameter_Manager_Instance.Parameter_Table_Copy_Timeout	-
0x0068 (104)	Parameter_Manager_Instance.Parameter_Table_Copy_Failure	Parameters_Memory_Region_Release.T
0x0069 (105)	Parameter_Manager_Instance.Command_Dropped	Command_Header.T
0x006a (106)	Product_Database_Instance.Data_Product_Update_Id_Out_Of_Range	Data_Product_Id.T
0x006b (107)	Product_Database_Instance.Data_Product_Fetch_Id_Out_Of_Range	Data_Product_Id.T
0x006c (108)	Product_Database_Instance.Data_Product_Fetch_Id_Not_Available	Data_Product_Id.T
0x006d (109)	Product_Database_Instance.Override_Cleared	Data_Product_Id.T
0x006e (110)	Product_Database_Instance.Override_Cleared_For_All	-
0x006f (111)	Product_Database_Instance.Data_Product_Overridden	Data_Product_Header.T
0x0070 (112)	Product_Database_Instance.Data_Product_Override_Serialization_Failure	Data_Product_Header.T
0x0071 (113)	Product_Database_Instance.Data_Product_Override_Id_Out_Of_Range	Data_Product_Id.T
0x0072 (114)	Product_Database_Instance.Data_Product_Clear_Override_Id_Out_Of_Range	Data_Product_Id.T
0x0073 (115)	Product_Database_Instance.Data_Product_Dump_Id_Not_Available	Data_Product_Id.T

0x0074 (116)	Product_Database_Instance.Data_Product_Dump_Id_Out_Of_Range	Data_Product_Id.T
0x0075 (117)	Product_Database_Instance.Data_Product_Dumped	Data_Product_Header.T
0x0076 (118)	Product_Database_Instance.Dumping_Data_Product_Poly_Type	Data_Product_Poly_Extract.T
0x0077 (119)	Product_Database_Instance.Dumped_Data_Product_Poly_Type	Data_Product_Poly_Event.T
0x0078 (120)	Product_Database_Instance.Data_Product_Dump_Poly_Id_Not_Available	Data_Product_Id.T
0x0079 (121)	Product_Database_Instance.Data_Product_Dump_Poly_Id_Out_Of_Range	Data_Product_Id.T
0x007a (122)	Product_Database_Instance.Data_Product_Poly_Type_Extraction_Failed	Data_Product_Header.T
0x007b (123)	Product_Database_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x007c (124)	Product_Packetizer_Instance.Invalid_Packet_Id_Commanded	Invalid_Packet_Id.T
0x007d (125)	Product_Packetizer_Instance.Packet_Enabled	Packet_Period.T
0x007e (126)	Product_Packetizer_Instance.Packet_Disabled	Packet_Period.T
0x007f (127)	Product_Packetizer_Instance.Packet_Period_Set	Packet_Period.T
0x0080 (128)	Product_Packetizer_Instance.Data_Product_Missing_On_Fetch	Packet_Data_Product_Ids.T
0x0081 (129)	Product_Packetizer_Instance.Packet_Period_Item_Bad_Id	Packet_Data_Product_Ids.T

0x0082 (130)	Product_Packetizer_Instance.Data_Product_Length_Mismatch	Invalid_Data_Product_Length.T
0x0083 (131)	Product_Packetizer_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x0084 (132)	Product_Packetizer_Instance.Dropped_Command	Command_Header.T
0x0085 (133)	Ccsds_Socket_Interface_Instance.Socket_Connected	Socket_Address.T
0x0086 (134)	Ccsds_Socket_Interface_Instance.Socket_Not_Connected	Socket_Address.T
0x0087 (135)	Ccsds_Socket_Interface_Instance.Packet_Send_Failed	Ccsds_Primary_Header.T
0x0088 (136)	Ccsds_Socket_Interface_Instance.Packet_Recv_Failed	Ccsds_Primary_Header.T
0x0089 (137)	Memory_Packetizer_Instance.Max_Packet_Id_Exceeded	Packet_Id.T
0x008a (138)	Memory_Packetizer_Instance.Memory_Dump_Request_Dropped	Packet_Id.T
0x008b (139)	Memory_Packetizer_Instance.Max_Packet_Rate_Set	Packets_Per_Period.T
0x008c (140)	Memory_Packetizer_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x008d (141)	Interrupt_Responder_Instance.Interrupt_Received	Tick.T
0x008e (142)	Counter_Instance.Set_Count_Command_Received	Packed_U32.T
0x008f (143)	Counter_Instance.Reset_Count_Command_Received	-
0x0090 (144)	Counter_Instance.Set_Count_Add_Command_Received	Operands.T
0x0091 (145)	Counter_Instance.Sending_Value	Packed_U32.T

0x0092 (146)	Counter_Instance.Dropped_Command	Command_Header.T
0x0093 (147)	Counter_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x0094 (148)	Oscillator_A.Frequency_Value_Set	Packed_F32.T
0x0095 (149)	Oscillator_A.Amplitude_Value_Set	Packed_F32.T
0x0096 (150)	Oscillator_A.Offset_Value_Set	Packed_F32.T
0x0097 (151)	Oscillator_A.Dropped_Command	Command_Header.T
0x0098 (152)	Oscillator_A.Invalid_Command_Received	Invalid_Command_Info.T
0x0099 (153)	Oscillator_A.Invalid_Parameter_Received	Invalid_Parameter_Info.T
0x009a (154)	Oscillator_B.Frequency_Value_Set	Packed_F32.T
0x009b (155)	Oscillator_B.Amplitude_Value_Set	Packed_F32.T
0x009c (156)	Oscillator_B.Offset_Value_Set	Packed_F32.T
0x009d (157)	Oscillator_B.Dropped_Command	Command_Header.T
0x009e (158)	Oscillator_B.Invalid_Command_Received	Invalid_Command_Info.T
0x009f (159)	Oscillator_B.Invalid_Parameter_Received	Invalid_Parameter_Info.T
0x00a0 (160)	Zero_Divider_Instance.Dividing_By_Zero	Packed_Natural.T
0x00a1 (161)	Zero_Divider_Instance.Invalid_Magic_Number	Packed_U32.T
0x00a2 (162)	Zero_Divider_Instance.Invalid_Command_Received	Invalid_Command_Info.T

0x00a3 (163)	Task_Watchdog_Instance.Watchdog_Pet_Checks_Enabled	-
0x00a4 (164)	Task_Watchdog_Instance.Watchdog_Pet_Checks_Disabled	-
0x00a5 (165)	Task_Watchdog_Instance.Watchdog_Limit_Set	Watchdog_Limit_Cmd.T
0x00a6 (166)	Task_Watchdog_Instance.Watchdog_Action_Set	Watchdog_Action_Cmd.T
0x00a7 (167)	Task_Watchdog_Instance.Watchdog_Limit_Change_Index_Out_Of_Range	Packed_Connector_Index.T
0x00a8 (168)	Task_Watchdog_Instance.Watchdog_Action_Change_Index_Out_Of_Range	Packed_Connector_Index.T
0x00a9 (169)	Task_Watchdog_Instance.Watchdog_Action_Change_Invalid_Transition_To_Fault	Packed_Connector_Index.T
0x00aa (170)	Task_Watchdog_Instance.Component_Exceeded_Pet_Limit	Packed_Connector_Index.T
0x00ab (171)	Task_Watchdog_Instance.Critical_Task_Not_Petting	Packed_Connector_Index.T
0x00ac (172)	Task_Watchdog_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x00ad (173)	Fault_Producer_Instance.Sending_Fault_1	-
0x00ae (174)	Fault_Producer_Instance.Sending_Fault_2	-
0x00af (175)	Fault_Producer_Instance.Invalid_Command_Received	Invalid_Command_Info.T
0x00b0 (176)	Fault_Correction_Instance.Fault_Received	Fault_Static.T
0x00b1 (177)	Fault_Correction_Instance.Fault_Response_Sent	Command_Header.T

0x00b2 (178)	Fault_Correction_Instance. Fault_Response_Cleared	Packed_Fault_Id.T
0x00b3 (179)	Fault_Correction_Instance. Fault_Response_Disabled	Packed_Fault_Id.T
0x00b4 (180)	Fault_Correction_Instance. Fault_Response_Enabled	Packed_Fault_Id.T
0x00b5 (181)	Fault_Correction_Instance. All_Fault_Responses_Cleared	-
0x00b6 (182)	Fault_Correction_Instance. Unrecognized_Fault_Id	Packed_Fault_Id.T
0x00b7 (183)	Fault_Correction_Instance. Invalid_Command_Received	Invalid_Command_Info.T
0x00b8 (184)	Fault_Correction_Instance. Command_Dropped	Command_Header.T
0x00b9 (185)	Fault_Correction_Instance. Fault_Dropped	Fault_Header.T
0x00ba (186)	Fault_Correction_Instance. Data_Products_Reset	-
0x00bb (187)	Cpu_Monitor_Instance. Packet_Period_Set	Packed_U16.T
0x00bc (188)	Cpu_Monitor_Instance. Invalid_Command_Received	Invalid_Command_Info.T
0x00bd (189)	Queue_Monitor_Instance. Packet_Period_Set	Packed_U16.T
0x00be (190)	Queue_Monitor_Instance. Invalid_Command_Received	Invalid_Command_Info.T
0x00bf (191)	Stack_Monitor_Instance. Packet_Period_Set	Packed_U16.T
0x00c0 (192)	Stack_Monitor_Instance. Invalid_Command_Received	Invalid_Command_Info.T

Event Descriptions:

- **Tick_Divider_Instance.Component_Has_Full_Queue** - The tick divider tried to put a Tick on a component's queue, but the queue was full, so the Tick was dropped.
- **Slow_Rate_Group.Cycle_Slip** - Execution ran long on this cycle.
- **Slow_Rate_Group.Max_Cycle_Time_Exceeded** - A new maximum cycle time was

reached. The event parameter is a Tick type with the maximum cycle time as the Time and the cycle count where the maximum cycle was achieved as the Count.

- **Slow_Rate_Group.Max_Execution_Time_Exceeded** - A new maximum execution time was reached. The event parameter is a Tick type with the maximum cycle time as the Time and the cycle count where the maximum cycle was achieved as the Count.
- **Slow_Rate_Group.Component_Has_Full_Queue** - The rate group tried to put a Tick on a component's queue, but the queue was full, so the Tick was dropped.
- **Slow_Rate_Group.Incoming_Tick_Dropped** - The rate group component's queue is full, so it cannot store the tick coming in. This usually means the rate group is cycle slipping and not running as fast as it needs to.
- **Fast_Rate_Group.Cycle_Slip** - Execution ran long on this cycle.
- **Fast_Rate_Group.Max_Cycle_Time_Exceeded** - A new maximum cycle time was reached. The event parameter is a Tick type with the maximum cycle time as the Time and the cycle count where the maximum cycle was achieved as the Count.
- **Fast_Rate_Group.Max_Execution_Time_Exceeded** - A new maximum execution time was reached. The event parameter is a Tick type with the maximum cycle time as the Time and the cycle count where the maximum cycle was achieved as the Count.
- **Fast_Rate_Group.Component_Has_Full_Queue** - The rate group tried to put a Tick on a component's queue, but the queue was full, so the Tick was dropped.
- **Fast_Rate_Group.Incoming_Tick_Dropped** - The rate group component's queue is full, so it cannot store the tick coming in. This usually means the rate group is cycle slipping and not running as fast as it needs to.
- **Watchdog_Rate_Group.Cycle_Slip** - Execution ran long on this cycle.
- **Watchdog_Rate_Group.Max_Cycle_Time_Exceeded** - A new maximum cycle time was reached. The event parameter is a Tick type with the maximum cycle time as the Time and the cycle count where the maximum cycle was achieved as the Count.
- **Watchdog_Rate_Group.Max_Execution_Time_Exceeded** - A new maximum execution time was reached. The event parameter is a Tick type with the maximum cycle time as the Time and the cycle count where the maximum cycle was achieved as the Count.
- **Watchdog_Rate_Group.Component_Has_Full_Queue** - The rate group tried to put a Tick on a component's queue, but the queue was full, so the Tick was dropped.
- **Watchdog_Rate_Group.Incoming_Tick_Dropped** - The rate group component's queue is full, so it cannot store the tick coming in. This usually means the rate group is cycle slipping and not running as fast as it needs to.
- **Ccsds_Command_Depacketizer_Instance.Invalid_Packet_Checksum** - A packet was received with an invalid checksum
- **Ccsds_Command_Depacketizer_Instance.Invalid_Packet_Type** - A packet was received with an invalid ccsds packet type. The expected packet type is a telecommand, but a telemtry packet was received.
- **Ccsds_Command_Depacketizer_Instance.Packet_Too_Small** - The packet recieved was too small to contain necessary command information.
- **Ccsds_Command_Depacketizer_Instance.Packet_Too_Large** - The packet recieved was too large and is bigger than the size of a command.
- **Ccsds_Command_Depacketizer_Instance.No_Secondary_Header** - A packet was received without a secondary header, but the secondary header is required.
- **Ccsds_Command_Depacketizer_Instance.Counts_Reset** - A command was received to reset the counts.
- **Ccsds_Command_Depacketizer_Instance.Invalid_Command_Received** - A command was received with invalid parameters.

- **Command_Router_Instance.Command_Received** - A command was received by the command router to be routed.
- **Command_Router_Instance.Command_Execution_Successful** - A command was routed, executed, and returned a response saying it was executed successfully
- **Command_Router_Instance.Command_Execution_Failure** - A command execution failed.
- **Command_Router_Instance.Command_Id_Not_Registered** - A command was sent to the router, but it was not found in the router table.
- **Command_Router_Instance.Registration_Id_Conflict** - The command Id has already been registered.
- **Command_Router_Instance.Router_Table_Full** - Cannot add command Id to router table because it is full.
- **Command_Router_Instance.Outgoing_Command_Dropped** - A command was dropped because the recipient's queue was full.
- **Command_Router_Instance.Incoming_Command_Dropped** - A command was dropped because the command router's queue was full.
- **Command_Router_Instance.Noop_Command_Dropped** - A noop command was dropped because the command router's queue was full.
- **Command_Router_Instance.Command_Response_Dropped** - A command response was dropped because the command router's queue was full.
- **Command_Router_Instance.Noop_Received** - A Noop command was received.
- **Command_Router_Instance.Noop_Arg_Received** - A Noop command was received with an argument.
- **Command_Router_Instance.Noop_Response_Received** - A noop response self test command was received.
- **Command_Router_Instance.Noop_Response_Forwarding_Success** - If this event is sent then the noop response self test command succeeded.
- **Command_Router_Instance.Forwarded_Command_Response_Dropped** - A forwarded command response was dropped because the receiving component's queue overflowed.
- **Command_Router_Instance.Invalid_Command_Source_Id** - A command response contained an invalid source id. This is a software bug and should be corrected.
- **Command_Router_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Command_Router_Instance.Data_Products_Reset** - The component's data products have been reset to initialization values.
- **Event_Filter_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Event_Filter_Instance.Filtered_Event** - This event indicates that the state of an event was set to enabled for the filter.
- **Event_Filter_Instance.Unfiltered_Event** - This event indicates that the state of an event was set to disabled for the filter.
- **Event_Filter_Instance.Filtered_Event_Range** - This event indicates that the state of a range of events were set to enabled for the filter.
- **Event_Filter_Instance.Unfiltered_Event_Range** - This event indicates that the state of a range of events were set to disabled for the filter.
- **Event_Filter_Instance.Enable_Event_Filter** - This event indicates that the state of all events were set to enabled for the filter, but kept the internal state.

- **`Event_Filter_Instance.Disable_Event_Filter`** - This event indicates that the state of all events were set to disabled for the filter, but kept the internal state.
- **`Event_Filter_Instance.Filter_Event_Invalid_Id`** - This event indicates that the command to change the event state to enabled failed since the event ID was out of range.
- **`Event_Filter_Instance.Unfilter_Event_Invalid_Id`** - This event indicates that the command to change the event state to disable failed since the event ID was out of range.
- **`Event_Filter_Instance.Filter_Event_Range_Invalid_Id`** - This event indicates that changing the state for the range to enabled, failed due to an invalid id.
- **`Event_Filter_Instance.Unfilter_Event_Range_Invalid_Id`** - This event indicates that changing the state for the range to disabled, failed due to an invalid id.
- **`Event_Filter_Instance.Dump_Event_States_Recieved`** - Event that indicates the process of building the packet that stores the event states has started and will send the packet once we go through a decrement cycle.
- **`Event_Limiter_Instance.Invalid_Command_Received`** - A command was received with invalid parameters.
- **`Event_Limiter_Instance.Events_Limited_Since_Last_Tick`** - An event that indicates how many events have been limited as well as up to the first 10 ids of those events. The event ids listed may not have been dropped, but are at listed since they are at the current max limit.
- **`Event_Limiter_Instance.Event_Limit_Enabled`** - This event indicates that the state of an event was set to enabled for the limiter.
- **`Event_Limiter_Instance.Event_Limit_Disabled`** - This event indicates that the state of an event was set to disabled for the limiter.
- **`Event_Limiter_Instance.Event_Limit_Range_Enabled`** - This event indicates that the state of a range of events were set to enabled for the limiter.
- **`Event_Limiter_Instance.Event_Limit_Range_Disabled`** - This event indicates that the state of a range of events were set to disabled for the limiter.
- **`Event_Limiter_Instance.Event_Limiting_Enabled`** - This event indicates that the state of all events were set to enabled for the limiter.
- **`Event_Limiter_Instance.Event_Limiting_Disabled`** - This event indicates that the state of all events were set to disabled for the limiter.
- **`Event_Limiter_Instance.Event_Limit_Enable_Invalid_Id`** - This event indicates that the command to change the event state to enabled failed since the event ID was out of range.
- **`Event_Limiter_Instance.Event_Limit_Disable_Invalid_Id`** - This event indicates that the command to change the event state to disable failed since the event ID was out of range.
- **`Event_Limiter_Instance.Event_Limit_Range_Enabled_Invalid_Id`** - This event indicates that changing the state for the range to enabled, failed due to an invalid id.
- **`Event_Limiter_Instance.Event_Limit_Range_Disabled_Invalid_Id`** - This event indicates that changing the state for the range to disabled, failed due to an invalid id.
- **`Event_Limiter_Instance.Set_New_Persistence`** - Indicates that the persistence of the number of events until we limit was changed to a new value between 1 and 7.
- **`Event_Limiter_Instance.Dump_Event_States_Recieved`** - Event that indicates the process of building the packet that stores the event states has started and will send the packet once we go through a decrement cycle.
- **`Event_Post_Mortem_Logger.Log_Attempt_Failed`** - A log attempt failed with the following status.

- **Event_Post_Mortem_Logger.Log_Disabled** - The log was disabled. No more data will be stored.
- **Event_Post_Mortem_Logger.Log_Enabled** - The log was enabled. Data will now be stored.
- **Event_Post_Mortem_Logger.Log_Info_Update** - The current meta data of the log was requested.
- **Event_Post_Mortem_Logger.Dumping_Log_Memory** - Currently dumping log memory from the following location.
- **Event_Post_Mortem_Logger.Invalid_Command_Received** - A command was received with invalid parameters.
- **Parameters_Instance.Parameter_Update_Success** - A parameter value was updated.
- **Parameters_Instance.Parameter_Update_Id_Not_Recognized** - A parameter value could not be updated because the ID is not recognized.
- **Parameters_Instance.Parameter_Stage_Failed** - A parameter value could not be updated.
- **Parameters_Instance.Parameter_Update_Failed** - A parameter value could not be updated.
- **Parameters_Instance.Parameter_Fetch_Failed** - A parameter value could not be updated.
- **Parameters_Instance.Parameter_Fetch_Length_Mismatch** - A parameter was fetched but contained an unexpected length.
- **Parameters_Instance.Parameter_Update_Length_Mismatch** - A parameter command was received to update a parameter but it contained an unexpected length.
- **Parameters_Instance.Memory_Region_Length_Mismatch** - A memory region was received with an invalid length. The length of the region must be the same size as the parameter table.
- **Parameters_Instance.Memory_Region_Crc_Invalid** - A memory region parameter table was received with an invalid CRC. The computed CRC does not match the CRC found in the header.
- **Parameters_Instance.Dumping_Parameters** - Producing a packet with the currently staged parameter values contained within connected components.
- **Parameters_Instance.Finished_Dumping_Parameters** - Done dumping the parameters.
- **Parameters_Instance.Starting_Parameter_Table_Update** - Starting updating of the parameters from a received memory region.
- **Parameters_Instance.Finished_Parameter_Table_Update** - Done updating the parameters from a received memory region with following status.
- **Parameters_Instance.Starting_Parameter_Table_Fetch** - Starting updating of the parameters from a received memory region.
- **Parameters_Instance.Finished_Parameter_Table_Fetch** - Done updating the parameters from a received memory region with following status.
- **Parameters_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Parameters_Instance.Command_Dropped** - A command was dropped due to a full queue.
- **Parameters_Instance.Memory_Region_Dropped** - A memory region was dropped due to a full queue.
- **Parameter_Store_Instance.Memory_Region_Length_Mismatch** - A memory region

was received with an invalid length. The length of the region must be the same size as the parameter table.

- **Parameter_Store_Instance.Memory_Region_Crc_Invalid** - A memory region parameter table was received with an invalid CRC. The computed CRC does not match the CRC found in the header.
- **Parameter_Store_Instance.Dumped_Parameters** - Produced a packet with the contents of the parameter store.
- **Parameter_Store_Instance.Parameter_Table_Updated** - Parameter table updated from a received memory region.
- **Parameter_Store_Instance.Parameter_Table_Fetched** - Starting fetching of the parameters into received memory region.
- **Parameter_Store_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Parameter_Store_Instance.Command_Dropped** - A command was dropped due to a full queue.
- **Parameter_Store_Instance.Memory_Region_Dropped** - A memory region was dropped due to a full queue.
- **Parameter_Manager_Instance.Starting_Parameter_Table_Copy** - Starting parameter table copy from source to destination.
- **Parameter_Manager_Instance.Finished_Parameter_Table_Copy** - Finished parameter table copy from source to destination, without errors.
- **Parameter_Manager_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Parameter_Manager_Instance.Parameter_Table_Copy_Timeout** - A timeout occured while waiting for a parameter table copy operation to complete.
- **Parameter_Manager_Instance.Parameter_Table_Copy_Failure** - A parameter table copy failed.
- **Parameter_Manager_Instance.Command_Dropped** - A command was dropped due to a full queue.
- **Product_Database_Instance.Data_Product_Update_Id_Out_Of_Range** - A data product update was received with an ID that was out of range.
- **Product_Database_Instance.Data_Product_Fetch_Id_Out_Of_Range** - A data product fetch was received with an ID that was out of range.
- **Product_Database_Instance.Data_Product_Fetch_Id_Not_Available** - A data product fetch was received with an ID that has not yet been stored in the database.
- **Product_Database_Instance.Override_Cleared** - Override condition cleared for the data product of the provided ID.
- **Product_Database_Instance.Override_Cleared_For_All** - Override condition cleared for all data productd.
- **Product_Database_Instance.Data_Product_Overridden** - Data product overridden by command.
- **Product_Database_Instance.Data_Product_Override_Serialization_Failure** - Data product override could not be completed due to a serialization error.
- **Product_Database_Instance.Data_Product_Override_Id_Out_Of_Range** - A data product override command was received with an ID that was out of range.
- **Product_Database_Instance.Data_Product_Clear_Override_Id_Out_Of_Range** - A data product clear override command was received with an ID that was out of range.

- **Product_Database_Instance.Data_Product_Dump_Id_Not_Available** - A data product dump command was received with an ID that has not yet been stored in the database.
- **Product_Database_Instance.Data_Product_Dump_Id_Out_Of_Range** - A data product dump command was received with an ID that was out of range.
- **Product_Database_Instance.Data_Product_Dumped** - Data product dumped into a packet by command.
- **Product_Database_Instance.Dumping_Data_Product_Poly_Type** - Data product poly type dumped into a packet by command.
- **Product_Database_Instance.Dumped_Data_Product_Poly_Type** - Data product poly type dumped into a packet by command.
- **Product_Database_Instance.Data_Product_Dump_Poly_Id_Not_Available** - A data product dump poly command was received with an ID that has not yet been stored in the database.
- **Product_Database_Instance.Data_Product_Dump_Poly_Id_Out_Of_Range** - A data product dump poly command was received with an ID that was out of range.
- **Product_Database_Instance.Data_Product_Poly_Type_Extraction_Failed** - A data product dump poly command failed because the extraction could not succeed with the provided parameters.
- **Product_Database_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Product_Packetizer_Instance.Invalid_Packet_Id_Commanded** - An invalid packet id was commanded for a given command.
- **Product_Packetizer_Instance.Packet_Enabled** - An packet was enabled.
- **Product_Packetizer_Instance.Packet_Disabled** - An packet was disabled.
- **Product_Packetizer_Instance.Packet_Period_Set** - An packet period was set.
- **Product_Packetizer_Instance.Data_Product_Missing_On_Fetch** - A data product was missing when fetched for packet insertion.
- **Product_Packetizer_Instance.Packet_Period_Item_Bad_Id** - A packet period packet item could not be formed because the ID is invalid.
- **Product_Packetizer_Instance.Data_Product_Length_Mismatch** - A data product was fetched but contained an unexpected length.
- **Product_Packetizer_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Product_Packetizer_Instance.Dropped_Command** - A command was dropped due to a full queue.
- **Ccsds_Socket_Interface_Instance.Socket_Connected** - The socket was successfully connected on the host and port provided.
- **Ccsds_Socket_Interface_Instance.Socket_Not_Connected** - The socket connection failed on the host and port provided
- **Ccsds_Socket_Interface_Instance.Packet_Send_Failed** - Failed to send a packet over the socket because it has an invalid CCSDS header.
- **Ccsds_Socket_Interface_Instance.Packet_Recv_Failed** - Failed to receive a packet over the socket because it has an invalid CCSDS header.
- **Memory_Packetizer_Instance.Max_Packet_Id_Exceeded** - The maximum number of packet ids that the component can keep track of sequence numbers for has been exceeded. Packets of this id will be emitted with a sequence number of 0.
- **Memory_Packetizer_Instance.Memory_Dump_Request_Dropped** - The queue for

memory dump requests overflowed and a request to dump memory with the given packet id was dropped.

- **Memory_Packetizer_Instance.Max_Packet_Rate_Set** - A new maximum rate has been set for the packetizer.
- **Memory_Packetizer_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Interrupt_Responder_Instance.Interrupt_Received** - Received an interrupt.
- **Counter_Instance.Set_Count_Command_Received** - Received a Set_Count command.
- **Counter_Instance.Reset_Count_Command_Received** - Received a Reset_Count command.
- **Counter_Instance.Set_Count_Add_Command_Received** - Received a Set_Count_Add command.
- **Counter_Instance.Sending_Value** - Sending the current value out as data product.
- **Counter_Instance.Dropped_Command** - The component's queue overflowed and the command was dropped.
- **Counter_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Oscillator_A.Frequency_Value_Set** - A new frequency value was set by command
- **Oscillator_A.Amplitude_Value_Set** - A new amplitude value was set by command
- **Oscillator_A.Offset_Value_Set** - A new offset value was set by command
- **Oscillator_A.Dropped_Command** - The component's queue overflowed and the command was dropped.
- **Oscillator_A.Invalid_Command_Received** - A command was received with invalid parameters.
- **Oscillator_A.Invalid_Parameter_Received** - A parameter was received with invalid parameters.
- **Oscillator_B.Frequency_Value_Set** - A new frequency value was set by command
- **Oscillator_B.Amplitude_Value_Set** - A new amplitude value was set by command
- **Oscillator_B.Offset_Value_Set** - A new offset value was set by command
- **Oscillator_B.Dropped_Command** - The component's queue overflowed and the command was dropped.
- **Oscillator_B.Invalid_Command_Received** - A command was received with invalid parameters.
- **Oscillator_B.Invalid_Parameter_Received** - A parameter was received with invalid parameters.
- **Zero_Divider_Instance.Dividing_By_Zero** - A divide by zero command was received, and the magic number was correct. The division will occur in N milliseconds, where N is provided as the event parameter.
- **Zero_Divider_Instance.Invalid_Magic_Number** - A divide by zero command was received, but the magic number was incorrect. The division will not occur.
- **Zero_Divider_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Task_Watchdog_Instance.Watchdog_Pet_Checks_Enabled** - Indicates a command was recieved to enable the checks on upstream pets.
- **Task_Watchdog_Instance.Watchdog_Pet_Checks_Disabled** - Indicates a command was recieved to disable the checks on upstream pets.

- **Task_Watchdog_Instance.Watchdog_Limit_Set** - An event to indicate that the limit was changed by command for a particular index.
- **Task_Watchdog_Instance.Watchdog_Action_Set** - An event to indicate that the action was changed by command for a particular index.
- **Task_Watchdog_Instance.Watchdog_Limit_Change_Index_Out_Of_Range** - Event indicating there was an error for the index range in the set limit command.
- **Task_Watchdog_Instance.Watchdog_Action_Change_Index_Out_Of_Range** - Event indicating there was an error for the index range in the set limit command.
- **Task_Watchdog_Instance.Watchdog_Action_Change_Invalid_Transition_To_Fault** - Event indicating there was an error trying to set the action to fault. The petter did not have a fault declared in the model so the action cannot be set to fault.
- **Task_Watchdog_Instance.Component_Exceeded_Pet_Limit** - Event to indicate a pet connector has not received a pet within the set limits for that component.
- **Task_Watchdog_Instance.Critical_Task_Not_Petting** - Event to indicate that one or more of our critical tasks have not indicated a pet in the maximum limit of ticks. The hardware watchdog will not be pet in this case.
- **Task_Watchdog_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Fault_Producer_Instance.Sending_Fault_1** - The component received a command to send out fault 1.
- **Fault_Producer_Instance.Sending_Fault_2** - The component received a command to send out fault 2.
- **Fault_Producer_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Fault_Correction_Instance.Fault_Received** - A fault was received.
- **Fault_Correction_Instance.Fault_Response_Sent** - A fault response was sent with the included command header.
- **Fault_Correction_Instance.Fault_Response_Cleared** - A fault response was cleared.
- **Fault_Correction_Instance.Fault_Response_Disabled** - A fault response has been disabled
- **Fault_Correction_Instance.Fault_Response_Enabled** - A fault response has been enabled.
- **Fault_Correction_Instance.All_Fault_Responses_Cleared** - Any latched faults have been unlatched by command.
- **Fault_Correction_Instance.Unrecognized_Fault_Id** - A fault response entry with the included fault ID was not found in the table.
- **Fault_Correction_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Fault_Correction_Instance.Command_Dropped** - A command was dropped due to a full queue.
- **Fault_Correction_Instance.Fault_Dropped** - A fault was dropped due to a full queue.
- **Fault_Correction_Instance.Data_Products_Reset** - The component's data products have been reset to initialization values.
- **Cpu_Monitor_Instance.Packet_Period_Set** - A command was received to change the packet period.
- **Cpu_Monitor_Instance.Invalid_Command_Received** - A command was received with

invalid parameters.

- **Queue_Monitor_Instance.Packet_Period_Set** - A command was received to change the packet period.
- **Queue_Monitor_Instance.Invalid_Command_Received** - A command was received with invalid parameters.
- **Stack_Monitor_Instance.Packet_Period_Set** - A command was received to change the packet period.
- **Stack_Monitor_Instance.Invalid_Command_Received** - A command was received with invalid parameters.

2.8 Data Products

The table below shows the data products for the Linux Example assembly.

Table 7: Linux Example Data Products

Data Product ID	Data Product Name	Type
0x0001 (1)	Slow_Rate_Group.Timing_Report	Task_Timing_Report.T
0x0002 (2)	Fast_Rate_Group.Timing_Report	Task_Timing_Report.T
0x0003 (3)	Watchdog_Rate_Group.Timing_Report	Task_Timing_Report.T
0x0004 (4)	Ccsds_Command_Depacketizer_Instance.Rejected_Packet_Count	Packed_U16.T
0x0005 (5)	Ccsds_Command_Depacketizer_Instance.Accepted_Packet_Count	Packed_U16.T
0x0006 (6)	Command_Router_Instance.Command_Receive_Count	Packed_U16.T
0x0007 (7)	Command_Router_Instance.Command_Success_Count	Packed_U16.T
0x0008 (8)	Command_Router_Instance.Command_Failure_Count	Packed_U16.T
0x0009 (9)	Command_Router_Instance.Last_Received_Command	Command_Id.T
0x000a (10)	Command_Router_Instance.Last_Successful_Command	Command_Id.T

0x000b (11)	Command_Router_Instance.Last_Failed_Command	Command_Id_Status.T
0x000c (12)	Command_Router_Instance.Noop_Arg_Last_Value	Command_Router_Arg.T
0x000d (13)	Event_Filter_Instance.Total_Events_Filtered	Packed_U32.T
0x000e (14)	Event_Filter_Instance.Total_Events_Unfiltered	Packed_U32.T
0x000f (15)	Event_Filter_Instance.Component_Filter_State	Event_Component_State_Type.T
0x0010 (16)	Event_Limiter_Instance.Limited_Events_Since_Tick	Packed_U16.T
0x0011 (17)	Event_Limiter_Instance.Total_Events_Limited	Packed_U32.T
0x0012 (18)	Event_Limiter_Instance.Component_Limiting_Enabled_Status	Event_Enable_State_Type.T
0x0013 (19)	Event_Post_Mortem_Logger.Mode	Logger_Status.T
0x0014 (20)	Event_Packetizer_Instance.Events_Dropped_Count	Packed_U32.T
0x0015 (21)	Event_Packetizer_Instance.Bytes_Available	Packed_Natural.T
0x0016 (22)	Product_Database_Instance.Data_Product_Poly_Type_Dump	Data_Product_Poly_Type.T
0x0017 (23)	Product_Database_Instance.Database_Override	Packed_Enable_Disable_Type.T
0x0018 (24)	Memory_Packetizer_Instance.Max_Packets_Per_Time_Period	Packets_Per_Period.T
0x0019 (25)	Oscillator_A.Oscillator_Value	Packed_F32.T
0x001a (26)	Oscillator_B.Oscillator_Value	Packed_F32.T

0x001b (27)	Task_Watchdog_Instance.Watchdog_Component_Petter_State	Packed_Watchdog_Component_State.T
0x001c (28)	Task_Watchdog_Instance.Pet_Connector_Action_States	Linux_Example_Task_Watchdog_List_State_Record.T
0x001d (29)	Task_Watchdog_Instance.Slow_Rate_Group_Limit	Packed_Missed_Pet_Limit.T
0x001e (30)	Task_Watchdog_Instance.Fast_Rate_Group_Limit	Packed_Missed_Pet_Limit.T
0x001f (31)	Fault_Correction_Instance.Fault_Counter	Packed_U16.T
0x0020 (32)	Fault_Correction_Instance.Last_Fault_Id_Received	Linux_Example_Fault_Responses_Packed_Id_Type.T
0x0021 (33)	Fault_Correction_Instance.Time_Of_Last_Fault_Received	Sys_Time.T
0x0022 (34)	Fault_Correction_Instance.Fault_Response_Statuses	Linux_Example_Fault_Responses_Status_Record.T
0x0023 (35)	Cpu_Monitor_Instance.Packet_Period	Packed_U16.T
0x0024 (36)	Queue_Monitor_Instance.Packet_Period	Packed_U16.T
0x0025 (37)	Stack_Monitor_Instance.Packet_Period	Packed_U16.T

Data Product Descriptions:

- **Slow_Rate_Group.Timing_Report** - Data relating timing performance of the component.
- **Fast_Rate_Group.Timing_Report** - Data relating timing performance of the component.
- **Watchdog_Rate_Group.Timing_Report** - Data relating timing performance of the component.
- **Ccsds_Command_Depacketizer_Instance.Rejected_Packet_Count** - The number of packets rejected by the component due to invalid data
- **Ccsds_Command_Depacketizer_Instance.Accepted_Packet_Count** - The number of packets accepted by the component
- **Command_Router_Instance.Command_Receive_Count** - The number of commands received by the component.
- **Command_Router_Instance.Command_Success_Count** - The number of commands that successfully executed.

- **Command_Router_Instance.Command_Failure_Count** - The number of commands that failed to execute.
- **Command_Router_Instance.Last_Received_Command** - The ID of the last received command by the command router.
- **Command_Router_Instance.Last_Successful_Command** - The ID of the last successful command routed by the command router.
- **Command_Router_Instance.Last_Failed_Command** - The ID and status of the last failed command routed by the command router.
- **Command_Router_Instance.Noop_Arg_Last_Value** - The last value sent with the Noop_Arg command. This data product can be useful for testing purposes.
- **Event_Filter_Instance.Total_Events_Filtered** - The total number of events that have been filtered for the components lifetime.
- **Event_Filter_Instance.Total_Events_Unfiltered** - The total number of events that have been passed through for the components lifetime. Does not include out of range IDs.
- **Event_Filter_Instance.Component_Filter_State** - The state of the master switch for filtering events.
- **Event_Limiter_Instance.Limited_Events_Since_Tick** - The number of events that were limited since the last tick.
- **Event_Limiter_Instance.Total_Events_Limited** - The total number of events that have been limited for the components lifetime.
- **Event_Limiter_Instance.Component_Limiting_Enabled_Status** - The current state of the component master switch.
- **Event_Post_Mortem_Logger.Mode** - The current enabled/disabled mode of the component.
- **Event_Packetizer_Instance.Events_Dropped_Count** - The number of events dropped by the component.
- **Event_Packetizer_Instance.Bytes_Available** - The current number of bytes available for event storage within the component.
- **Product_Database_Instance.Data_Product_Poly_Type_Dump** - Data product poly type dumped into a data product by command.
- **Product_Database_Instance.Database_Override** - If set to Enabled then the database contains at least one data product that has been overridden by command.
- **Memory_Packetizer_Instance.Max_Packets_Per_Time_Period** - The current maximum packet sends per time period.
- **Oscillator_A.Oscillator_Value** - The current value of the oscillator.
- **Oscillator_B.Oscillator_Value** - The current value of the oscillator.
- **Task_Watchdog_Instance.Watchdog_Component_Petter_State** - Data product that tracks the global state to enable or disable all checks on the upstream watchdog pets.
- **Task_Watchdog_Instance.Pet_Connector_Action_States** - 2-bit of state for each pet connector indicating the current action that will be taken if there is an error. Note that Packed_U32.T is just a placeholder type for this data product. The actual type of this data product will be autocoded and at assembly model ingest time.
- **Task_Watchdog_Instance.Slow_Rate_Group_Limit** - Slow rate group monitoring.
- **Task_Watchdog_Instance.Fast_Rate_Group_Limit** - Fast rate group monitoring.
- **Fault_Correction_Instance.Fault_Counter** - The number of faults received by the component.
- **Fault_Correction_Instance.Last_Fault_Id_Received** - The ID of the last fault re-

ceived.

- **Fault_Correction_Instance.Time_Of_Last_Fault_Received** - The system time of the last fault received.
- **Fault_Correction_Instance.Fault_Response_Statuses** - 2-bits of status for each fault response that this component is managing. Note that Packed_U32.T is just a placeholder type for this data product. The actual type of this data product will be autocoded and at assembly model ingest time.
- **Cpu_Monitor_Instance.Packet_Period** - The current packet period.
- **Queue_Monitor_Instance.Packet_Period** - The current packet period.
- **Stack_Monitor_Instance.Packet_Period** - The current packet period.

2.9 Packets

The table below shows the packets for the Linux Example assembly.

Table 8: Linux Example Packets

Packet ID	Packet Name	Type
0x0001 (1)	Product_Packetizer_Instance.Housekeeping_Packet	*Undefined*
0x0002 (2)	Ccsds_Command_Depacketizer_Instance.Error_Packet	Ccsds_Space_Packet.T
0x0003 (3)	Event_Filter_Instance.Event_Filter_State_Packet	*Undefined*
0x0004 (4)	Event_Limiter_Instance.Event_Limiter_State_Packet	*Undefined*
0x0005 (5)	Event_Post_Mortem_Logger.Log_Packet	*Undefined*
0x0006 (6)	Parameters_Instance.Active_Parameters	Linux_Example_Parameter_Table_Record.T
0x0007 (7)	Counter_Instance.Counter_Value	Packed_U32.T
0x0008 (8)	Parameter_Store_Instance.Stored_Parameters	Linux_Example_Parameter_Table_Record.T
0x0009 (9)	Product_Database_Instance.Dump_Packet	Data_Product.T
0x000a (10)	Zero_Divider_Instance.Last_Chance_Handler_Packet	Packed_Exception_Occurrence.T

`0x000b (11)`	`Cpu_Monitor_Instance.Cpu_Usage_Packet`	`Linux_Example_Cpu_Monitor_Packet_Type.T`
`0x000c (12)`	`Queue_Monitor_Instance.Queue_Usage_Packet`	`Linux_Example_Queue_Monitor_Packet_Type.T`
`0x000d (13)`	`Stack_Monitor_Instance.Stack_Usage_Packet`	`Linux_Example_Stack_Monitor_Packet_Type.T`
`0x0062 (98)`	`Event_Packetizer_Instance.Events_Packet`	*Undefined*

Packet Descriptions:

- **`Product_Packetizer_Instance.Housekeeping_Packet`** - This packet contains housekeeping data.
- **`Ccsds_Command_Depacketizer_Instance.Error_Packet`** - This packet contains a CCSDS packet that was dropped due to error.
- **`Event_Filter_Instance.Event_Filter_State_Packet`** - The packet used to dump all the state information for which events are filtered and which are not. Each event ID takes a bit and any extra bits beyond the event range will show as not filtered.
- **`Event_Limiter_Instance.Event_Limiter_State_Packet`** - The packet used to dump all the state information for which events are limited and which are not. Each event takes a bit and any extra bits beyond the event range will show as disabled.
- **`Event_Post_Mortem_Logger.Log_Packet`** - This packet contains log data.
- **`Parameters_Instance.Active_Parameters`** - This packet contains a copy of all the active parameters managed by this component.
- **`Counter_Instance.Counter_Value`** - The counter value 1.
- **`Parameter_Store_Instance.Stored_Parameters`** - This packet contains a copy of all the parameter stored and managed by this component.
- **`Product_Database_Instance.Dump_Packet`** - This packet contains dumped data products.
- **`Zero_Divider_Instance.Last_Chance_Handler_Packet`** - This packet contains information regarding an exception occurrence that triggers the Last_Chance_Handler to get invoked. This packet is not produced directly by this component, and should be produced by the last chance handler implementation. This packet definition exists to ensure that the packet gets reflected in the documentation and ground system definitions.
- **`Cpu_Monitor_Instance.Cpu_Usage_Packet`** - This packet contains cpu usage numbers for tasks and interrupts in the system.
- **`Queue_Monitor_Instance.Queue_Usage_Packet`** - This packet contains queue usage numbers for queued components in the system.
- **`Stack_Monitor_Instance.Stack_Usage_Packet`** - This packet contains stack and secondary stack usage numbers for tasks in the system.
- **`Event_Packetizer_Instance.Events_Packet`** - This packet contains events as subpackets.

2.10 Faults

The table below shows the faults for the Linux Example assembly.

Table 9: Linux Example Faults

Fault ID	Fault Name	Parameter Type
0x0001 (1)	Task_Watchdog_Instance.Slow_Rate_Group_Fault	Packed_Connector_Index.T
0x0002 (2)	Task_Watchdog_Instance.Fast_Rate_Group_Fault	Packed_Connector_Index.T
0x0003 (3)	Fault_Producer_Instance.Fault_1	-
0x0004 (4)	Fault_Producer_Instance.Fault_2	Packed_Natural.T

Fault Descriptions:

- **Task_Watchdog_Instance.Slow_Rate_Group_Fault** - Slow rate group monitoring.
- **Task_Watchdog_Instance.Fast_Rate_Group_Fault** - Fast rate group monitoring.
- **Fault_Producer_Instance.Fault_1** - First fault that the component can send.
- **Fault_Producer_Instance.Fault_2** - Second fault that the component can send.

3 Appendix

3.1 Connections

Table 10: Linux Example Connections

Number	From	To	Kind
1	Ticker_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
2	Tick_Divider_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
3	Slow_Rate_Group.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
4	Fast_Rate_Group.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return

5	Watchdog_Rate_Group.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
6	Command_Router_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
7	Counter_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
8	Oscillator_A.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
9	Oscillator_B.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
10	Ccsds_Socket_Interface_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
11	Interrupt_Servicer_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
12	Interrupt_Responder_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
13	Event_Packetizer_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
14	Product_Database_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
15	Product_Packetizer_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
16	Ccsds_Command_Depacketizer_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
17	Event_Post_Mortem_Logger.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
18	Memory_Packetizer_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return

19	Cpu_Monitor_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
20	Stack_Monitor_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
21	Queue_Monitor_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
22	Event_Filter_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
23	Event_Limiter_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
24	Parameters_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
25	Parameter_Store_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
26	Zero_Divider_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
27	Task_Watchdog_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
28	Fault_Correction_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
29	Parameter_Manager_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
30	Fault_Producer_Instance.Sys_Time_T_Get	System_Time_Instance.Sys_Time_T_Return	get-return
31	Ticker_Instance.Tick_T_Send	Tick_Divider_Instance.Tick_T_Recv_Sync	send-recv_sync

32	Tick_Divider_Instance.Tick_T_Send [1]	Watchdog_Rate_Group.Tick_T_Recv_Async	send-recv_async
33	Tick_Divider_Instance.Tick_T_Send [2]	Slow_Rate_Group.Tick_T_Recv_Async	send-recv_async
34	Tick_Divider_Instance.Tick_T_Send [3]	Fast_Rate_Group.Tick_T_Recv_Async	send-recv_async
35	Slow_Rate_Group.Tick_T_Send [1]	Counter_Instance.Tick_T_Recv_Sync	send-recv_sync
36	Slow_Rate_Group.Tick_T_Send [2]	Event_Packetizer_Instance.Tick_T_Recv_Sync	send-recv_sync
37	Slow_Rate_Group.Tick_T_Send [3]	Cpu_Monitor_Instance.Tick_T_Recv_Sync	send-recv_sync
38	Slow_Rate_Group.Tick_T_Send [4]	Queue_Monitor_Instance.Tick_T_Recv_Sync	send-recv_sync
39	Slow_Rate_Group.Tick_T_Send [5]	Stack_Monitor_Instance.Tick_T_Recv_Sync	send-recv_sync
40	Slow_Rate_Group.Tick_T_Send [6]	Event_Filter_Instance.Tick_T_Recv_Sync	send-recv_sync
41	Slow_Rate_Group.Tick_T_Send [7]	Event_Limiter_Instance.Tick_T_Recv_Sync	send-recv_sync
42	Slow_Rate_Group.Tick_T_Send [8]	Parameter_Manager_Instance.Timeout_Tick_Recv_Sync	send-recv_sync
43	Fast_Rate_Group.Tick_T_Send [1]	Oscillator_A.Tick_T_Recv_Sync	send-recv_sync
44	Fast_Rate_Group.Tick_T_Send [2]	Oscillator_B.Tick_T_Recv_Sync	send-recv_sync
45	Fast_Rate_Group.Tick_T_Send [3]	Product_Packetizer_Instance.Tick_T_Recv_Sync	send-recv_sync

46	Watchdog_Rate_Group.Tick_T_Send	Task_Watchdog_Instance.Tick_T_Recv_Sync	send-recv_sync
47	Tick_Divider_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
48	Slow_Rate_Group.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
49	Fast_Rate_Group.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
50	Watchdog_Rate_Group.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
51	Command_Router_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
52	Counter_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
53	Oscillator_A.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
54	Oscillator_B.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
55	Ccsds_Socket_Interface_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
56	Product_Packetizer_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
57	Stack_Monitor_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
58	Queue_Monitor_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
59	Cpu_Monitor_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
60	Event_Filter_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
61	Event_Limiter_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync

62	Parameters_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
63	Parameter_Store_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
64	Zero_Divider_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
65	Task_Watchdog_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
66	Fault_Correction_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
67	Event_Splitter_Instance.T_Send [1]	Event_Filter_Instance.Event_T_Recv_Sync	send-recv_sync
68	Event_Filter_Instance.Event_Forward_T_Send	Event_Limiter_Instance.Event_T_Recv_Sync	send-recv_sync
69	Event_Limiter_Instance.Event_Forward_T_Send	Event_Splitter_2_Instance.T_Recv_Sync	send-recv_sync
70	Event_Splitter_2_Instance.T_Send [1]	Event_Packetizer_Instance.Event_T_Recv_Sync	send-recv_sync
71	Event_Splitter_2_Instance.T_Send [2]	Event_Text_Logger_Instance.Event_T_Recv_Async	send-recv_async
72	Event_Splitter_Instance.T_Send [2]	Event_Post_Mortem_Logger.T_Recv_Sync	send-recv_sync
73	Interrupt_Responder_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
74	Product_Database_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
75	Ccsds_Command_Depacketizer_Instance.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync
76	Event_Post_Mortem_Logger.Event_T_Send	Event_Splitter_Instance.T_Recv_Sync	send-recv_sync

77	Memory_Packetizer_ Instance.Event_T_Send	Event_Splitter_ Instance.T_Recv_Sync	send-recv_sync
78	Parameter_Manager_ Instance.Event_T_Send	Event_Splitter_ Instance.T_Recv_Sync	send-recv_sync
79	Fault_Producer_ Instance.Event_T_Send	Event_Splitter_ Instance.T_Recv_Sync	send-recv_sync
80	Command_Router_ Instance.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
81	Counter_Instance. Command_Response_T_ Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
82	Oscillator_A.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
83	Oscillator_B.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
84	Product_Packetizer_ Instance.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
85	Event_Packetizer_ Instance.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
86	Event_Post_Mortem_ Logger.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
87	Memory_Packetizer_ Instance.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
88	Ccsds_Command_ Depacketizer_ Instance.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
89	Product_Database_ Instance.Command_ Response_T_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async

90	Stack_Monitor_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
91	Queue_Monitor_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
92	Cpu_Monitor_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
93	Event_Filter_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
94	Event_Limiter_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
95	Parameters_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
96	Parameter_Store_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
97	Task_Watchdog_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
98	Fault_Correction_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
99	Zero_Divider_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
100	Parameter_Manager_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
101	Fault_Producer_Instance.Command_Response_T_Send	Command_Router_Instance.Command_Response_T_Recv_Async	send-recv_async
102	Command_Router_Instance.Command_T_Send [1]	Command_Router_Instance.Command_T_Recv_Async	send-recv_async

103	Command_Router_Instance.Command_T_Send [2]	Counter_Instance.Command_T_Recv_Async	send-recv_async
104	Command_Router_Instance.Command_T_Send [3]	Oscillator_A.Command_T_Recv_Async	send-recv_async
105	Command_Router_Instance.Command_T_Send [4]	Oscillator_B.Command_T_Recv_Async	send-recv_async
106	Command_Router_Instance.Command_T_Send [5]	Product_Packetizer_Instance.Command_T_Recv_Async	send-recv_async
107	Command_Router_Instance.Command_T_Send [6]	Event_Packetizer_Instance.Command_T_Recv_Sync	send-recv_sync
108	Command_Router_Instance.Command_T_Send [7]	Event_Post_Mortem_Logger.Command_T_Recv_Sync	send-recv_sync
109	Command_Router_Instance.Command_T_Send [8]	Memory_Packetizer_Instance.Command_T_Recv_Async	send-recv_async
110	Command_Router_Instance.Command_T_Send [9]	Product_Database_Instance.Command_T_Recv_Sync	send-recv_sync
111	Command_Router_Instance.Command_T_Send [10]	Ccsds_Command_Depacketizer_Instance.Command_T_Recv_Sync	send-recv_sync
112	Ccsds_Command_Depacketizer_Instance.Command_T_Send	Command_Router_Instance.Command_T_To_Route_Recv_Async	send-recv_async
113	Command_Router_Instance.Command_T_Send [11]	Stack_Monitor_Instance.Command_T_Recv_Sync	send-recv_sync
114	Command_Router_Instance.Command_T_Send [12]	Queue_Monitor_Instance.Command_T_Recv_Sync	send-recv_sync

115	Command_Router_ Instance.Command_ T_Send [13]	Cpu_Monitor_Instance. Command_T_Recv_Sync	send-recv_sync
116	Command_Router_ Instance.Command_ T_Send [14]	Event_Filter_ Instance.Command_ T_Recv_Sync	send-recv_sync
117	Command_Router_ Instance.Command_ T_Send [15]	Event_Limiter_ Instance.Command_ T_Recv_Sync	send-recv_sync
118	Command_Router_ Instance.Command_ T_Send [16]	Parameters_Instance. Command_T_Recv_Async	send-recv_async
119	Command_Router_ Instance.Command_ T_Send [17]	Parameter_Store_ Instance.Command_ T_Recv_Async	send-recv_async
120	Command_Router_ Instance.Command_ T_Send [18]	Zero_Divider_ Instance.Command_ T_Recv_Sync	send-recv_sync
121	Command_Router_ Instance.Command_ T_Send [19]	Fault_Correction_ Instance.Command_T_ Recv_Async	send-recv_async
122	Command_Router_ Instance.Command_ T_Send [20]	Task_Watchdog_ Instance.Command_ T_Recv_Sync	send-recv_sync
123	Command_Router_ Instance.Command_ T_Send [21]	Parameter_Manager_ Instance.Command_T_ Recv_Async	send-recv_async
124	Command_Router_ Instance.Command_ T_Send [22]	Fault_Producer_ Instance.Command_ T_Recv_Sync	send-recv_sync
125	Command_Router_ Instance.Command_ Response_T_To_ Forward_Send	Command_Router_ Instance.Command_ Response_T_Recv_Async	send-recv_async
126	Oscillator_A.Data_ Product_T_Send	Product_Database_ Instance.Data_ Product_T_Recv_Sync	send-recv_sync
127	Oscillator_B.Data_ Product_T_Send	Product_Database_ Instance.Data_ Product_T_Recv_Sync	send-recv_sync

128	Event_Packetizer_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
129	Ccsds_Command_Depacketizer_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
130	Slow_Rate_Group.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
131	Fast_Rate_Group.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
132	Watchdog_Rate_Group.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
133	Command_Router_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
134	Event_Post_Mortem_Logger.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
135	Memory_Packetizer_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
136	Product_Database_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
137	Stack_Monitor_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
138	Queue_Monitor_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
139	Cpu_Monitor_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
140	Event_Filter_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync

141	Event_Limiter_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
142	Task_Watchdog_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
143	Fault_Correction_Instance.Data_Product_T_Send	Product_Database_Instance.Data_Product_T_Recv_Sync	send-recv_sync
144	Product_Packetizer_Instance.Data_Product_Fetch_T_Request	Product_Database_Instance.Data_Product_Fetch_T_Service	request-service
145	Interrupt_Servicer_Instance.Interrupt_Data_Type_Send	Interrupt_Responder_Instance.Tick_T_Recv_Sync	send-recv_sync
146	Ccsds_Socket_Interface_Instance.Ccsds_Space_Packet_T_Send	Ccsds_Command_Depacketizer_Instance.Ccsds_Space_Packet_T_Recv_Sync	send-recv_sync
147	Event_Post_Mortem_Logger.Memory_Dump_Send	Memory_Packetizer_Instance.Memory_Dump_Recv_Async	send-recv_async
148	Event_Packetizer_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
149	Product_Packetizer_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
150	Counter_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
151	Memory_Packetizer_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
152	Ccsds_Command_Depacketizer_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync

153	Product_Database_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
154	Ccsds_Packetizer_Instance.Ccsds_Space_Packet_T_Send	Ccsds_Socket_Interface_Instance.Ccsds_Space_Packet_T_Recv_Async	send-recv_async
155	Cpu_Monitor_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
156	Queue_Monitor_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
157	Stack_Monitor_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
158	Parameters_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
159	Parameter_Store_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
160	Event_Filter_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
161	Event_Limiter_Instance.Packet_T_Send	Ccsds_Packetizer_Instance.Packet_T_Recv_Sync	send-recv_sync
162	Parameters_Instance.Parameter_Update_T_Provide [1]	Oscillator_A.Parameter_Update_T_Modify	provide-modify
163	Parameters_Instance.Parameter_Update_T_Provide [2]	Oscillator_B.Parameter_Update_T_Modify	provide-modify
164	Slow_Rate_Group.Pet_T_Send	Task_Watchdog_Instance.Pet_T_Recv_Sync	send-recv_sync
165	Fast_Rate_Group.Pet_T_Send	Task_Watchdog_Instance.Pet_T_Recv_Sync	send-recv_sync

166	Task_Watchdog_Instance.Fault_T_Send	Fault_Correction_Instance.Fault_T_Recv_Async	send-recv_async
167	Fault_Producer_Instance.Fault_T_Send	Fault_Correction_Instance.Fault_T_Recv_Async	send-recv_async
168	Fault_Correction_Instance.Command_T_Send	Command_Router_Instance.Command_T_To_Route_Recv_Sync	send-recv_sync
169	Parameter_Manager_Instance.Working_Parameters_Memory_Region_Send	Parameters_Instance.Parameters_Memory_Region_T_Recv_Async	send-recv_async
170	Parameter_Manager_Instance.Default_Parameters_Memory_Region_Send	Parameter_Store_Instance.Parameters_Memory_Region_T_Recv_Async	send-recv_async
171	Parameter_Store_Instance.Parameters_Memory_Region_Release_T_Send	Parameter_Manager_Instance.Parameters_Memory_Region_Release_T_Recv_Sync	send-recv_sync
172	Parameters_Instance.Parameters_Memory_Region_Release_T_Send	Parameter_Manager_Instance.Parameters_Memory_Region_Release_T_Recv_Sync	send-recv_sync

Connection Descriptions:

- **Ticker_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Tick_Divider_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Slow_Rate_Group.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Fast_Rate_Group.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Watchdog_Rate_Group.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Command_Router_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Counter_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Oscillator_A.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*

- **Oscillator_B.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Ccsds_Socket_Interface_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Interrupt_Servicer_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Interrupt_Responder_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Event_Packetizer_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Product_Database_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Product_Packetizer_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Ccsds_Command_Depacketizer_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Event_Post_Mortem_Logger.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Memory_Packetizer_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Cpu_Monitor_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Stack_Monitor_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Queue_Monitor_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Event_Filter_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Event_Limiter_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Parameters_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Parameter_Store_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Zero_Divider_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Task_Watchdog_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Fault_Correction_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Parameter_Manager_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Fault_Producer_Instance.Sys_Time_T_Get-System_Time_Instance.Sys_Time_T_Return** - *No description provided.*
- **Ticker_Instance.Tick_T_Send-Tick_Divider_Instance.Tick_T_Recv_Sync** - *No description provided.*
- **Tick_Divider_Instance.Tick_T_Send[1]-Watchdog_Rate_Group.Tick_T_**

Recv_Async - *No description provided.*

- **Tick_Divider_Instance.Tick_T_Send[2]-Slow_Rate_Group.Tick_T_Recv_Async** - *No description provided.*
- **Tick_Divider_Instance.Tick_T_Send[3]-Fast_Rate_Group.Tick_T_Recv_Async** - *No description provided.*
- **Slow_Rate_Group.Tick_T_Send[1]-Counter_Instance.Tick_T_Recv_Sync** - Schedule connection from the rate group to the counter
- **Slow_Rate_Group.Tick_T_Send[2]-Event_Packetizer_Instance.Tick_T_Recv_Sync** - Schedule connection from the rate group to the packetizer
- **Slow_Rate_Group.Tick_T_Send[3]-Cpu_Monitor_Instance.Tick_T_Recv_Sync** - *No description provided.*
- **Slow_Rate_Group.Tick_T_Send[4]-Queue_Monitor_Instance.Tick_T_Recv_Sync** - *No description provided.*
- **Slow_Rate_Group.Tick_T_Send[5]-Stack_Monitor_Instance.Tick_T_Recv_Sync** - *No description provided.*
- **Slow_Rate_Group.Tick_T_Send[6]-Event_Filter_Instance.Tick_T_Recv_Sync** - *No description provided.*
- **Slow_Rate_Group.Tick_T_Send[7]-Event_Limiter_Instance.Tick_T_Recv_Sync** - *No description provided.*
- **Slow_Rate_Group.Tick_T_Send[8]-Parameter_Manager_Instance.Timeout_Tick_Recv_Sync** - *No description provided.*
- **Fast_Rate_Group.Tick_T_Send[1]-Oscillator_A.Tick_T_Recv_Sync** - Schedule connection from the rate group to the oscillator
- **Fast_Rate_Group.Tick_T_Send[2]-Oscillator_B.Tick_T_Recv_Sync** - Schedule connection from the rate group to the oscillator
- **Fast_Rate_Group.Tick_T_Send[3]-Product_Packetizer_Instance.Tick_T_Recv_Sync** - Schedule connection from the rate group to the packetizer
- **Watchdog_Rate_Group.Tick_T_Send-Task_Watchdog_Instance.Tick_T_Recv_Sync** - *No description provided.*
- **Tick_Divider_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Slow_Rate_Group.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Fast_Rate_Group.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Watchdog_Rate_Group.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Counter_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Oscillator_A.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Oscillator_B.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Ccsds_Socket_Interface_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*

- **Product_Packetizer_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Stack_Monitor_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Queue_Monitor_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Cpu_Monitor_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Event_Filter_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Event_Limiter_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Parameters_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Parameter_Store_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Zero_Divider_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Task_Watchdog_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Fault_Correction_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Event_Splitter_Instance.T_Send[1]-Event_Filter_Instance.Event_T_Recv_Sync** - Split the event stream to create a filtered and limited stream for downlink and printing.
- **Event_Filter_Instance.Event_Forward_T_Send-Event_Limiter_Instance.Event_T_Recv_Sync** - *No description provided.*
- **Event_Limiter_Instance.Event_Forward_T_Send-Event_Splitter_2_Instance.T_Recv_Sync** - *No description provided.*
- **Event_Splitter_2_Instance.T_Send[1]-Event_Packetizer_Instance.Event_T_Recv_Sync** - *No description provided.*
- **Event_Splitter_2_Instance.T_Send[2]-Event_Text_Logger_Instance.Event_T_Recv_Async** - *No description provided.*
- **Event_Splitter_Instance.T_Send[2]-Event_Post_Mortem_Logger.T_Recv_Sync** - Unfiltered and unlimited event stream is passed to the post mortem event logger.
- **Interrupt_Responder_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Product_Database_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Ccsds_Command_Depacketizer_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Event_Post_Mortem_Logger.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Memory_Packetizer_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Parameter_Manager_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*

- **Fault_Producer_Instance.Event_T_Send-Event_Splitter_Instance.T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Counter_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Oscillator_A.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Oscillator_B.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Product_Packetizer_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Event_Packetizer_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Event_Post_Mortem_Logger.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Memory_Packetizer_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Ccsds_Command_Depacketizer_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Product_Database_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Stack_Monitor_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Queue_Monitor_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Cpu_Monitor_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Event_Filter_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Event_Limiter_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Parameters_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Parameter_Store_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Task_Watchdog_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Fault_Correction_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Zero_Divider_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Parameter_Manager_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Fault_Producer_Instance.Command_Response_T_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[1]-Command_Router_Instance.**

Command_T_Recv_Async - *No description provided.*

- **Command_Router_Instance.Command_T_Send[2]-Counter_Instance.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[3]-Oscillator_A.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[4]-Oscillator_B.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[5]-Product_Packetizer_Instance.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[6]-Event_Packetizer_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[7]-Event_Post_Mortem_Logger.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[8]-Memory_Packetizer_Instance.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[9]-Product_Database_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[10]-Ccsds_Command_Depacketizer_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Ccsds_Command_Depacketizer_Instance.Command_T_Send-Command_Router_Instance.Command_T_To_Route_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[11]-Stack_Monitor_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[12]-Queue_Monitor_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[13]-Cpu_Monitor_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[14]-Event_Filter_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[15]-Event_Limiter_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[16]-Parameters_Instance.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[17]-Parameter_Store_Instance.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[18]-Zero_Divider_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[19]-Fault_Correction_Instance.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[20]-Task_Watchdog_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[21]-Parameter_Manager_Instance.Command_T_Recv_Async** - *No description provided.*
- **Command_Router_Instance.Command_T_Send[22]-Fault_Producer_Instance.Command_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Command_Response_T_To_Forward_Send-Command_Router_Instance.Command_Response_T_Recv_Async** - *No description provided.*

- **Oscillator_A.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - Data product connection between the oscillator and packetizer
- **Oscillator_B.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - Data product connection between the oscillator and packetizer
- **Event_Packetizer_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Ccsds_Command_Depacketizer_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Slow_Rate_Group.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Fast_Rate_Group.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Watchdog_Rate_Group.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Command_Router_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Event_Post_Mortem_Logger.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Memory_Packetizer_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Product_Database_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Stack_Monitor_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Queue_Monitor_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Cpu_Monitor_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Event_Filter_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Event_Limiter_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Task_Watchdog_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Fault_Correction_Instance.Data_Product_T_Send-Product_Database_Instance.Data_Product_T_Recv_Sync** - *No description provided.*
- **Product_Packetizer_Instance.Data_Product_Fetch_T_Request-Product_Database_Instance.Data_Product_Fetch_T_Service** - *No description provided.*
- **Interrupt_Servicer_Instance.Interrupt_Data_Type_Send-Interrupt_Responder_Instance.Tick_T_Recv_Sync** - *No description provided.*
- **Ccsds_Socket_Interface_Instance.Ccsds_Space_Packet_T_Send-Ccsds_Command_Depacketizer_Instance.Ccsds_Space_Packet_T_Recv_Sync** - *No description provided.*
- **Event_Post_Mortem_Logger.Memory_Dump_Send-Memory_Packetizer_Instance.Memory_Dump_Recv_Async** - *No description provided.*
- **Event_Packetizer_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*

- **Product_Packetizer_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Counter_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Memory_Packetizer_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Ccsds_Command_Depacketizer_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Product_Database_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Ccsds_Packetizer_Instance.Ccsds_Space_Packet_T_Send-Ccsds_Socket_Interface_Instance.Ccsds_Space_Packet_T_Recv_Async** - *No description provided.*
- **Cpu_Monitor_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Queue_Monitor_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Stack_Monitor_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Parameters_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Parameter_Store_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Event_Filter_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Event_Limiter_Instance.Packet_T_Send-Ccsds_Packetizer_Instance.Packet_T_Recv_Sync** - *No description provided.*
- **Parameters_Instance.Parameter_Update_T_Provide[1]-Oscillator_A.Parameter_Update_T_Modify** - *No description provided.*
- **Parameters_Instance.Parameter_Update_T_Provide[2]-Oscillator_B.Parameter_Update_T_Modify** - *No description provided.*
- **Slow_Rate_Group.Pet_T_Send-Task_Watchdog_Instance.Pet_T_Recv_Sync[1]** - *No description provided.*
- **Fast_Rate_Group.Pet_T_Send-Task_Watchdog_Instance.Pet_T_Recv_Sync[2]** - *No description provided.*
- **Task_Watchdog_Instance.Fault_T_Send-Fault_Correction_Instance.Fault_T_Recv_Async** - *No description provided.*
- **Fault_Producer_Instance.Fault_T_Send-Fault_Correction_Instance.Fault_T_Recv_Async** - *No description provided.*
- **Fault_Correction_Instance.Command_T_Send-Command_Router_Instance.Command_T_To_Route_Recv_Sync** - We connect the fault correction command response to the command router synchronous connector for the fastest, most reliable execution. This bypasses the command router's queue.
- **Parameter_Manager_Instance.Working_Parameters_Memory_Region_Send-Parameters_Instance.Parameters_Memory_Region_T_Recv_Async** - *No description provided.*
- **Parameter_Manager_Instance.Default_Parameters_Memory_Region_Send-Parameter_Store_Instance.Parameters_Memory_Region_T_Recv_Async** -

No description provided.

- **Parameter_Store_Instance.Parameters_Memory_Region_Release_T_Send-Parameter_Manager_Instance.Parameters_Memory_Region_Release_T_Recv_Sync** - *No description provided.*
- **Parameters_Instance.Parameters_Memory_Region_Release_T_Send-Parameter_Manager_Instance.Parameters_Memory_Region_Release_T_Recv_Sync** - *No description provided.*

3.2 Packed Types

The following section outlines any complex data types used in the assembly in alphabetical order. This includes packed records and packed arrays that might be used as connector types, command arguments, or event parameters.

Ccsds_Command_Header.T:

Record for a LASP-specific CCSDS command header.

Table 11: Ccsds_Command_Header Packed Record : 64 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Primary_Header	Ccsds_Primary_Header.T	-	48	0	47
Secondary_Header	Ccsds_Command_Secondary_Header.T	-	16	48	63

Field Descriptions:

- **Primary_Header** - The CCSDS primary header
- **Secondary_Header** - The command secondary header

Ccsds_Command_Secondary_Header.T:

Record for the LASP-specific command secondary header.

Preamble (inline Ada definitions):

```
type Function_Code_Type is mod 2**7;
type One_Bit_Pad_Type is mod 2**1;
```

Table 12: Ccsds_Command_Secondary_Header Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Reserved	One_Bit_Pad_Type	0 to 1	1	0	0
Function_Code	Function_Code_Type	0 to 127	7	1	7
Checksum	Interfaces.Unsigned_8	0 to 255	8	8	15

Field Descriptions:

- **Reserved** - Reserve bit.
- **Function_Code** - The command function code.
- **Checksum** - An 8 bit checksum over the entire command packet

Ccsds_Primary_Header.T:

Record for the CCSDS Packet Primary Header

Preamble (inline Ada definitions):

```ada
subtype Three_Bit_Version_Type is Interfaces.Unsigned_8 range 0 .. 7;
type Ccsds_Apid_Type is mod 2**11;
type Ccsds_Sequence_Count_Type is mod 2**14;
```

Table 13: Ccsds_Primary_Header Packed Record : 48 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Version	Three_Bit_Version_Type	0 to 7	3	0	2
Packet_Type	Ccsds_Enums.Ccsds_Packet_Type.E	0 => Telemetry 1 => Telecommand	1	3	3
Secondary_Header	Ccsds_Enums.Ccsds_Secondary_Header_Indicator.E	0 => Secondary_Header_Not_Present 1 => Secondary_Header_Present	1	4	4
Apid	Ccsds_Apid_Type	0 to 2047	11	5	15
Sequence_Flag	Ccsds_Enums.Ccsds_Sequence_Flag.E	0 => Continuationsegment 1 => Firstsegment 2 => Lastsegment 3 => Unsegmented	2	16	17
Sequence_Count	Ccsds_Sequence_Count_Type	0 to 16383	14	18	31
Packet_Length	Interfaces.Unsigned_16	0 to 65535	16	32	47

Field Descriptions:

- **`Version`** - Packet Version Number
- **`Packet_Type`** - Packet Type
- **`Secondary_Header`** - Does packet have CCSDS secondary header
- **`Apid`** - Application process identifier
- **`Sequence_Flag`** - Sequence Flag

- **Sequence_Count** - Packet Sequence Count
- **Packet_Length** - This is the packet data length. One added to this number corresponds to the number of bytes included in the data section of the CCSDS Space Packet.

Ccsds_Space_Packet.T:

Record for the CCSDS Space Packet

Preamble (inline Ada definitions):

```
use Basic_Types;
subtype Ccsds_Data_Type is Byte_Array (0 ..
↪  Configuration.Ccsds_Packet_Buffer_Size - 1);
```

Table 14: Ccsds_Space_Packet Packed Record : 10240 bits *(maximum)*

Name	Type	Range	Size (Bits)	Start Bit	End Bit	Variable Length
Header	Ccsds_Primary_Header.T	-	48	0	47	-
Data	Ccsds_Data_Type	-	10192	48	10239	Header.Packet_Length

Field Descriptions:

- **Header** - The CCSDS Primary Header
- **Data** - User Data Field

Circular_Buffer_Meta.T:

This record holds meta data associated with a circular buffer data structure.

Table 15: Circular_Buffer_Meta Packed Record : 96 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Head	Interfaces.Unsigned_32	0 to 4294967295	32	0	31
Count	Interfaces.Unsigned_32	0 to 4294967295	32	32	63
Size	Interfaces.Unsigned_32	0 to 4294967295	32	64	95

Field Descriptions:

- **Head** - The head index of the buffer.
- **Count** - The number of bytes currently used in the buffer.
- **Size** - The total size of the buffer in bytes.

Command.T:

Generic command packet for holding arbitrary commands

Table 16: Command Packed Record : 2080 bits *(maximum)*

Name	Type	Range	Size (Bits)	Start Bit	End Bit	Variable Length
Header	Command_ Header.T	-	40	0	39	-
Arg_Buffer	Command_ Types. Command_Arg_ Buffer_Type	-	2040	40	2079	Header.Arg_ Buffer_Length

Field Descriptions:

- **Header** - The command header
- **Arg_Buffer** - A buffer to that contains the command arguments

Command_Header.T:

Generic command header for holding arbitrary commands

Table 17: Command_Header Packed Record : 40 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Source_Id	Command_Types. Command_Source_Id	0 to 65535	16	0	15
Id	Command_Types. Command_Id	0 to 65535	16	16	31
Arg_Buffer_Length	Command_Types. Command_Arg_Buffer_ Length_Type	0 to 255	8	32	39

Field Descriptions:

- **Source_Id** - The source ID. An ID assigned to a command sending component.
- **Id** - The command identifier
- **Arg_Buffer_Length** - The number of bytes used in the command argument buffer

Command_Id.T:

A packed record which holds a command identifier.

Table 18: Command_Id Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Command_Types. Command_Id	0 to 65535	16	0	15

Field Descriptions:

- **Id** - The command identifier

Command_Id_Status.T:

Record for holding a command identifier and command response status.

Table 19: Command_Id_Status Packed Record : 24 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Command_Types. Command_Id	0 to 65535	16	0	15
Status	Command_Enums. Command_ Response_ Status.E	0 => Success 1 => Failure 2 => Id_Error 3 => Validation_Error 4 => Length_Error 5 => Dropped 6 => Register 7 => Register_Source	8	16	23

Field Descriptions:

- **Id** - The command ID for the command response.
- **Status** - The command execution status.

Command_Response.T:

Record for holding command response data.

Table 20: Command_Response Packed Record : 56 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Source_Id	Command_ Types.Command_ Source_Id	0 to 65535	16	0	15
Registration_ Id	Command_ Types.Command_ Registration_ Id	0 to 65535	16	16	31
Command_Id	Command_Types. Command_Id	0 to 65535	16	32	47
Status	Command_Enums. Command_ Response_ Status.E	0 => Success 1 => Failure 2 => Id_Error 3 => Validation_Error 4 => Length_Error 5 => Dropped 6 => Register 7 => Register_Source	8	48	55

Field Descriptions:

- **Source_Id** - The source ID. An ID assigned to a command sending component.
- **Registration_Id** - The registration ID. An ID assigned to each registered component at initialization.
- **Command_Id** - The command ID for the command response.
- **Status** - The command execution status.

Command_Router_Arg.T:

A 32-bit unsigned integer with range 0 to 999.

Preamble (inline Ada definitions):

```
subtype Value_Type is Natural range 0 .. 999;
```

Table 21: Command_Router_Arg Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Value	Value_Type	0 to 999	32	0	31

Field Descriptions:

- **Value** - The 32-bit unsigned integer with range 0 to 999.

Cycle_Slip_Param.T:

This is a type that contains useful information about a cycle slip.

Table 22: Cycle_Slip_Param Packed Record : 112 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Slipped_Tick	Tick.T	-	96	0	95
Num_Slips	Interfaces. Unsigned_16	0 to 65535	16	96	111

Field Descriptions:

- **Slipped_Tick** - The tick during which the cycle slip occured.
- **Num_Slips** - The number of cycle slips that have occured.

Data_Product.T:

Generic data product packet for holding arbitrary data types

Table 23: Data_Product Packed Record : 344 bits *(maximum)*

Name	Type	Range	Size (Bits)	Start Bit	End Bit	Variable Length
Header	Data_Product_ Header.T	-	88	0	87	-
Buffer	Data_Product_ Types.Data_ Product_ Buffer_Type	-	256	88	343	Header.Buffer_ Length

Field Descriptions:

- **Header** - The data product header
- **Buffer** - A buffer that contains the data product type

Data_Product_Fetch.T:

A packed record which holds information for a data product request.

Table 24: Data_Product_Fetch Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Data_Product_Types. Data_Product_Id	0 to 65535	16	0	15

Field Descriptions:

- **Id** - The data product identifier

Data_Product_Header.T:

Generic data_product packet for holding arbitrary data_product types

Table 25: Data_Product_Header Packed Record : 88 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Time	Sys_Time.T	-	64	0	63
Id	Data_Product_Types. Data_Product_Id	0 to 65535	16	64	79
Buffer_Length	Data_Product_Types.Data_Product_Buffer_Length_Type	0 to 32	8	80	87

Field Descriptions:

- **Time** - The timestamp for the data product item.
- **Id** - The data product identifier
- **Buffer_Length** - The number of bytes used in the data product buffer

Data_Product_Id.T:

A packed record which holds a data product identifier.

Table 26: Data_Product_Id Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Data_Product_Types. Data_Product_Id	0 to 65535	16	0	15

Field Descriptions:

- **Id** - The data product identifier

Data_Product_Poly_Event.T:

Data product with 4 byte data buffer.

Table 27: Data_Product_Poly_Event Packed Record : 120 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Header	Data_Product_Header.T	-	88	0	87
Data	Basic_Types.Poly_32_Type	-	32	88	119

Field Descriptions:

- **Header** - The data product header
- **Data** - The polymorphic type.

Data_Product_Poly_Extract.T:

Contains information to extract a poly type from a data product.

Preamble (inline Ada definitions):

```
subtype Data_Product_Bit_Offset_Type is Natural range Natural'First ..
↪  Data_Product_Types.Data_Product_Buffer_Type'Length *
↪  Basic_Types.Byte'Object_Size; subtype Poly_Type_Size_Type is Positive range
↪  Positive'First .. 32;
```

Table 28: Data_Product_Poly_Extract Packed Record : 40 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Data_Product_Types.Data_Product_Id	0 to 65535	16	0	15
Offset	Data_Product_Bit_Offset_Type	0 to 256	16	16	31
Size	Poly_Type_Size_Type	1 to 32	8	32	39

Field Descriptions:

- **Id** - ID of the data product.
- **Offset** - Offset of the data product item (in bits).
- **Size** - Size of the data product item (in bits).

Data_Product_Poly_Type.T:

Data product poly type, for dumping arbitrary data products.

Table 29: Data_Product_Poly_Type Packed Record : 112 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Time	Sys_Time.T	-	64	0	63
Id	Data_Product_Types.Data_Product_Id	0 to 65535	16	64	79
Data	Basic_Types.Poly_32_Type	-	32	80	111

Field Descriptions:

- **Time** - The timestamp for the data product item.
- **Id** - The data product identifier
- **Data** - The polymorphic type.

Data_Product_Return.T:

This record holds data returned from a data product fetch request.

Table 30: Data_Product_Return Packed Record : 352 bits *(maximum)*

Name	Type	Range	Size (Bits)	Start Bit	End Bit	Variable Length
The_ Status	Data_ Product_ Enums. Fetch_ Status.E	0 => Success 1 => Not_Available 2 => Id_Out_Of_Range	8	0	7	-
The_Data_ Product	Data_ Product.T	-	344	8	351	-

Field Descriptions:

- **The_Status** - A status relating whether or not the data product fetch was successful or not.
- **The_Data_Product** - The data product item returned.

Delta_Time.T:

A record which holds a time difference using GPS format including seconds and subseconds.

Table 31: Delta_Time Packed Record : 64 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Seconds	Interfaces. Unsigned_32	0 to 4294967295	32	0	31
Subseconds	Interfaces. Unsigned_32	0 to 4294967295	32	32	63

Field Descriptions:

- **Seconds** - The number of seconds elapsed since epoch.
- **Subseconds** - The number of 1/(2^32) sub-seconds.

Event.T:

Generic event packet for holding arbitrary events

Table 32: Event Packed Record : 344 bits *(maximum)*

Name	Type	Range	Size (Bits)	Start Bit	End Bit	Variable Length
Header	Event_Header.T	-	88	0	87	-
Param_Buffer	Event_Types. Parameter_ Buffer_Type	-	256	88	343	Header.Param_ Buffer_Length

Field Descriptions:

- **Header** - The event header
- **Param_Buffer** - A buffer that contains the event parameters

Event_Component_State_Type.T:

This record is for the data product that indicates if the event filter component is enabled for filtering or disabled all together.

Table 33: Event_Component_State_Type Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Component_Filter_State	Event_Filter_Entry_Enums.Global_Filter_State.E	0 => Disabled 1 => Enabled	8	0	7

Field Descriptions:

- **Component_Filter_State** - Flag to indicate if the component is enabled or disabled at a overriding level from the individual event states

Event_Enable_State_Type.T:

This record contains the definition for a two event ID type for ranges in the event limiter commands as well as an issue packet type for issuing packets

Table 34: Event_Enable_State_Type Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Component_Enable_State	Two_Counter_Entry_Enums.Event_State_Type.E	0 => Disabled 1 => Enabled	8	0	7

Field Descriptions:

- **Component_Enable_State** - Flag to indicate if the component is enabled or disabled at a overriding level

Event_Filter_Id_Range.T:

This record contains the definition for a two event ID type for ranges in the event limiter commands as well as an issue packet type for issuing packets

Table 35: Event_Filter_Id_Range Packed Record : 40 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Start_Event_Id	Event_Id.T	-	16	0	15
Stop_Event_Id	Event_Id.T	-	16	16	31
Issue_State_Packet	Event_Filter_Enums.Issue_Packet_Type.E	0 => No_Issue 1 => Issue	8	32	39

Field Descriptions:

- **Start_Event_Id** - The starting event ID to begin the range
- **Stop_Event_Id** - The last event ID to end the range

- **Issue_State_Packet** - Flag to indicate if we dump the states after the change is complete

Event_Filter_Single_Event_Cmd_Type.T:

This record contains the definition for a two event ID type for ranges in the event limiter commands as well as an issue packet type for issuing packets

Table 36: Event_Filter_Single_Event_Cmd_Type Packed Record : 24 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Event_To_Update	Event_Id.T	-	16	0	15
Issue_State_Packet	Event_Filter_ Enums.Issue_ Packet_Type.E	0 => No_Issue 1 => Issue	8	16	23

Field Descriptions:

- **Event_To_Update** - The starting event ID to begin the range
- **Issue_State_Packet** - Flag to indicate if we dump the states after the change is complete

Event_Header.T:

Generic event packet for holding arbitrary events

Table 37: Event_Header Packed Record : 88 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Time	Sys_Time.T	-	64	0	63
Id	Event_Types.Event_ Id	0 to 65535	16	64	79
Param_Buffer_Length	Event_Types. Parameter_Buffer_ Length_Type	0 to 32	8	80	87

Field Descriptions:

- **Time** - The timestamp for the event.
- **Id** - The event identifier
- **Param_Buffer_Length** - The number of bytes used in the param buffer

Event_Id.T:

A packed record which holds an event identifier.

Table 38: Event_Id Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Event_Types.Event_ Id	0 to 65535	16	0	15

Field Descriptions:

- **Id** - The event identifier

Event_Id_Array.T:

Packed array of events so that it can be used with the event for which ids were limited

Table 39: Event_Id_Array Packed Array : 160 bits

Type	Range	Element Size (Bits)	Length	Total Size (Bits)
Event_Types.Event_Id	0 to 65535	16	10	160

Event_Limiter_Id_Range.T:

This record contains the definition for a two event ID type for ranges in the event limiter commands as well as an issue packet type for issuing packets

Table 40: Event_Limiter_Id_Range Packed Record : 40 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Start_Event_Id	Event_Id.T	-	16	0	15
Stop_Event_Id	Event_Id.T	-	16	16	31
Issue_State_Packet	Event_Limiter_ Enums.Issue_ Packet_Type.E	0 => No_Issue 1 => Issue	8	32	39

Field Descriptions:

- **Start_Event_Id** - The starting event ID to begin the range
- **Stop_Event_Id** - The last event ID to end the range
- **Issue_State_Packet** - Flag to indicate if we dump the states after the change is complete

Event_Limiter_Num_Events_Type.T:

This record contains the definition for the format of the event for this component that contains up to 10 event IDs of those that were limited.

Table 41: Event_Limiter_Num_Events_Type Packed Record : 184 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Num_Events_Limited	Interfaces. Unsigned_16	0 to 65535	16	0	15
Num_Event_Ids	Interfaces. Unsigned_8	0 to 255	8	16	23
Event_Id_Limited_ Array	Event_Id_Array.T	-	160	24	183

Field Descriptions:

- **Num_Events_Limited** - The number of events limited since last issuing the event.
- **Num_Event_Ids** - The number of event IDs contained in the event that indicate that event was limited.
- **Event_Id_Limited_Array** - The Array that list the event IDs that were limited since the last event message.

Event_Limiter_Persistence_Type.T:

This record contains the definition for a packed persistence type

Table 42: Event_Limiter_Persistence_Type Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Persistence	Two_Counter_Entry. Persistence_Type	1 to 7	8	0	7

Field Descriptions:

- **Persistence** - The persistence that is used when limiting events.

Event_Single_State_Cmd_Type.T:

This record contains the definition for a two event ID type for ranges in the event limiter commands as well as an issue packet type for issuing packets

Table 43: Event_Single_State_Cmd_Type Packed Record : 24 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Event_To_Update	Event_Id.T	-	16	0	15
Issue_State_Packet	Event_Limiter_ Enums.Issue_ Packet_Type.E	0 => No_Issue 1 => Issue	8	16	23

Field Descriptions:

- **Event_To_Update** - The starting event ID to begin the range
- **Issue_State_Packet** - Flag to indicate if we dump the states after the change is complete

Fault.T:

Generic fault packet for holding arbitrary faults.

Table 44: Fault Packed Record : 152 bits *(maximum)*

Name	Type	Range	Size (Bits)	Start Bit	End Bit	Variable Length
Header	Fault_Header.T	-	88	0	87	-
Param_Buffer	Fault_Types. Parameter_ Buffer_Type	-	64	88	151	Header.Param_ Buffer_Length

Field Descriptions:

- **Header** - The fault header
- **Param_Buffer** - A buffer that contains the fault parameters

Fault_Header.T:

Generic fault header.

Table 45: Fault_Header Packed Record : 88 bits

Name	**Type**	**Range**	**Size (Bits)**	**Start Bit**	**End Bit**
Time	Sys_Time.T	-	64	0	63
Id	Fault_Types.Fault_Id	0 to 65535	16	64	79
Param_Buffer_Length	Fault_Types.Parameter_Buffer_Length_Type	0 to 8	8	80	87

Field Descriptions:

- **Time** - The timestamp for the fault.
- **Id** - The fault identifier
- **Param_Buffer_Length** - The number of bytes used in the param buffer

Fault_Static.T:

Generic fault packet for holding arbitrary faults. This is the same as the Fault.T type, except that it is not variable sized, it is always maximum sized. This can be useful for sending events with faults in them.

Table 46: Fault_Static Packed Record : 152 bits

Name	**Type**	**Range**	**Size (Bits)**	**Start Bit**	**End Bit**
Header	Fault_Header.T	-	88	0	87
Param_Buffer	Fault_Types.Parameter_Buffer_Type	-	64	88	151

Field Descriptions:

- **Header** - The fault header
- **Param_Buffer** - A buffer that contains the fault parameters

Full_Queue_Param.T:

This is a type that contains useful information about a component full queue.

Table 47: Full_Queue_Param Packed Record : 112 bits

Name	**Type**	**Range**	**Size (Bits)**	**Start Bit**	**End Bit**
Dropped_Tick	Tick.T	-	96	0	95
Index	Connector_Types.Connector_Index_Type	1 to 65535	16	96	111

Field Descriptions:

- **Dropped_Tick** - The tick during which the component's queue was found to be full.
- **Index** - The rate group index number of the component that had a full queue.

Invalid_Command_Info.T:

Record for holding information about an invalid command

Table 48: Invalid_Command_Info Packed Record : 112 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Command_Types.Command_Id	0 to 65535	16	0	15
Errant_Field_Number	Interfaces.Unsigned_32	0 to 4294967295	32	16	47
Errant_Field	Basic_Types.Poly_Type	-	64	48	111

Field Descriptions:

- **Id** - The command Id received.
- **Errant_Field_Number** - The field that was invalid. 1 is the first field, 0 means unkown field, 2**32 means that the length field of the command was invalid.
- **Errant_Field** - A polymorphic type containing the bad field data, or length when Errant_Field_Number is 2**32.

Invalid_Data_Product_Length.T:

A packed record which holds data related to an invalid data product length.

Table 49: Invalid_Data_Product_Length Packed Record : 96 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Header	Data_Product_Header.T	-	88	0	87
Expected_Length	Data_Product_Types.Data_Product_Buffer_Length_Type	0 to 32	8	88	95

Field Descriptions:

- **Header** - The packet identifier
- **Expected_Length** - The packet length bound that the length failed to meet.

Invalid_Packet_Id.T:

A packed record which holds a packet identifier and data product identifier

Table 50: Invalid_Packet_Id Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Packet_Id	Packet_Types.Packet_Id	0 to 65535	16	0	15
Command_Id	Command_Types.Command_Id	0 to 65535	16	16	31

Field Descriptions:

- **Packet_Id** - The packet identifier
- **Command_Id** - The command id

Invalid_Packet_Length.T:

A packed record which holds data related to an invalid command packet length.

Table 51: Invalid_Packet_Length Packed Record : 112 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Ccsds_Header	Ccsds_Primary_Header.T	-	48	0	47
Length	Integer	-2147483648 to 2147483647	32	48	79
Length_Bound	Integer	-2147483648 to 2147483647	32	80	111

Field Descriptions:

- **Ccsds_Header** - The packet identifier
- **Length** - The packet length
- **Length_Bound** - The packet length bound that the length failed to meet.

Invalid_Packet_Xor8_Info.T:

A packed record which holds data related to an invalid checksummed CCSDS command packet.

Table 52: Invalid_Packet_Xor8_Info Packed Record : 80 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Ccsds_Header	Ccsds_Command_Header.T	-	64	0	63
Computed_Checksum	Xor_8.Xor_8_Type	0 to 255	8	64	71
Expected_Checksum	Xor_8.Xor_8_Type	0 to 255	8	72	79

Field Descriptions:

- **Ccsds_Header** - The CCSDS command header.
- **Computed_Checksum** - The computed XOR of the entire packet. This should be 0 if the packet passes.
- **Expected_Checksum** - The XOR included in the CCSDS packet secondary header.

Invalid_Parameter_Info.T:

Record for holding information about an invalid parameter

Table 53: Invalid_Parameter_Info Packed Record : 112 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Parameter_Types.Parameter_Id	0 to 65535	16	0	15
Errant_Field_Number	Interfaces.Unsigned_32	0 to 4294967295	32	16	47
Errant_Field	Basic_Types.Poly_Type	-	64	48	111

Field Descriptions:

- **Id** - The parameter Id received.

- **Errant_Field_Number** - The field that was invalid. 1 is the first field, 0 means unkown field, 2**32 means that the length field of the parameter was invalid.
- **Errant_Field** - A polymorphic type containing the bad field data, or length when Errant_Field_Number is 2**32.

Invalid_Parameter_Length.T:

A packed record which holds data related to an invalid parameter length.

Table 54: Invalid_Parameter_Length Packed Record : 56 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Header	Parameter_Header.T	-	24	0	23
Expected_Length	Natural	0 to 2147483647	32	24	55

Field Descriptions:

- **Header** - The packet identifier
- **Expected_Length** - The packet length bound that the length failed to meet.

Invalid_Parameters_Memory_Region_Crc.T:

A packed record which holds data related to an invalid parameter memory region CRC.

Table 55: Invalid_Parameters_Memory_Region_Crc Packed Record : 168 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Parameters_Region	Parameters_Memory_Region.T	-	104	0	103
Header	Parameter_Table_Header.T	-	48	104	151
Computed_Crc	Crc_16.Crc_16_Type	-	16	152	167

Field Descriptions:

- **Parameters_Region** - The memory region and operation.
- **Header** - The parameter table header stored in the memory region.
- **Computed_Crc** - The FSW computed CRC of the parameter table stored in the memory region.

Invalid_Parameters_Memory_Region_Length.T:

A packed record which holds data related to an invalid parameter memory region length.

Table 56: Invalid_Parameters_Memory_Region_Length Packed Record : 136 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Parameters_Region	Parameters_Memory_Region.T	-	104	0	103
Expected_Length	Natural	0 to 2147483647	32	104	135

Field Descriptions:

- **Parameters_Region** - The memory region and operation.
- **Expected_Length** - The length bound that the memory region failed to meet.

Linux_Example_Cpu_Monitor_Packet_Type.T:

This is the autocoded packet type for the CPU monitor component. It contains CPU utilization information for every task in the assembly.

Table 57: Linux_Example_Cpu_Monitor_Packet_Type Packed Record : 360 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Fault_Correction_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	0	7
Fault_Correction_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	8	15
Fault_Correction_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	16	23
Ticker_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	24	31
Ticker_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	32	39
Ticker_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	40	47
Watchdog_Rate_Group_Active_Usage_1	Basic_Types.Byte	0 to 255	8	48	55
Watchdog_Rate_Group_Active_Usage_2	Basic_Types.Byte	0 to 255	8	56	63
Watchdog_Rate_Group_Active_Usage_3	Basic_Types.Byte	0 to 255	8	64	71
Slow_Rate_Group_Active_Usage_1	Basic_Types.Byte	0 to 255	8	72	79
Slow_Rate_Group_Active_Usage_2	Basic_Types.Byte	0 to 255	8	80	87
Slow_Rate_Group_Active_Usage_3	Basic_Types.Byte	0 to 255	8	88	95
Fast_Rate_Group_Active_Usage_1	Basic_Types.Byte	0 to 255	8	96	103
Fast_Rate_Group_Active_Usage_2	Basic_Types.Byte	0 to 255	8	104	111
Fast_Rate_Group_Active_Usage_3	Basic_Types.Byte	0 to 255	8	112	119
Command_Router_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	120	127
Command_Router_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	128	135
Command_Router_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	136	143

Ccsds_Socket_Interface_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	144	151
Ccsds_Socket_Interface_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	152	159
Ccsds_Socket_Interface_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	160	167
Parameters_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	168	175
Parameters_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	176	183
Parameters_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	184	191
Parameter_Store_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	192	199
Parameter_Store_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	200	207
Parameter_Store_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	208	215
Parameter_Manager_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	216	223
Parameter_Manager_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	224	231
Parameter_Manager_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	232	239
Event_Text_Logger_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	240	247
Event_Text_Logger_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	248	255
Event_Text_Logger_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	256	263
Memory_Packetizer_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	264	271
Memory_Packetizer_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	272	279
Memory_Packetizer_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	280	287
Interrupt_Servicer_Instance_Active_Usage_1	Basic_Types.Byte	0 to 255	8	288	295

Interrupt_Servicer_Instance_Active_Usage_2	Basic_Types.Byte	0 to 255	8	296	303
Interrupt_Servicer_Instance_Active_Usage_3	Basic_Types.Byte	0 to 255	8	304	311
Ccsds_Socket_Interface_Instance_Listener_Usage_1	Basic_Types.Byte	0 to 255	8	312	319
Ccsds_Socket_Interface_Instance_Listener_Usage_2	Basic_Types.Byte	0 to 255	8	320	327
Ccsds_Socket_Interface_Instance_Listener_Usage_3	Basic_Types.Byte	0 to 255	8	328	335
Interrupt_Servicer_Instance_Usage_1	Basic_Types.Byte	0 to 255	8	336	343
Interrupt_Servicer_Instance_Usage_2	Basic_Types.Byte	0 to 255	8	344	351
Interrupt_Servicer_Instance_Usage_3	Basic_Types.Byte	0 to 255	8	352	359

Field Descriptions:

- **Fault_Correction_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Fault_Correction_Instance.Active task.
- **Fault_Correction_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Fault_Correction_Instance.Active task.
- **Fault_Correction_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Fault_Correction_Instance.Active task.
- **Ticker_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Ticker_Instance.Active task.
- **Ticker_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Ticker_Instance.Active task.
- **Ticker_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Ticker_Instance.Active task.
- **Watchdog_Rate_Group_Active_Usage_1** - The period one CPU utilization percentage for the Watchdog_Rate_Group.Active task.
- **Watchdog_Rate_Group_Active_Usage_2** - The period two CPU utilization percentage for the Watchdog_Rate_Group.Active task.
- **Watchdog_Rate_Group_Active_Usage_3** - The period three CPU utilization percentage for the Watchdog_Rate_Group.Active task.
- **Slow_Rate_Group_Active_Usage_1** - The period one CPU utilization percentage for the Slow_Rate_Group.Active task.
- **Slow_Rate_Group_Active_Usage_2** - The period two CPU utilization percentage for the Slow_Rate_Group.Active task.
- **Slow_Rate_Group_Active_Usage_3** - The period three CPU utilization percentage for the Slow_Rate_Group.Active task.
- **Fast_Rate_Group_Active_Usage_1** - The period one CPU utilization percentage for the Fast_Rate_Group.Active task.
- **Fast_Rate_Group_Active_Usage_2** - The period two CPU utilization percentage for the

Fast_Rate_Group.Active task.

- **Fast_Rate_Group_Active_Usage_3** - The period three CPU utilization percentage for the Fast_Rate_Group.Active task.
- **Command_Router_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Command_Router_Instance.Active task.
- **Command_Router_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Command_Router_Instance.Active task.
- **Command_Router_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Command_Router_Instance.Active task.
- **Ccsds_Socket_Interface_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Ccsds_Socket_Interface_Instance.Active task.
- **Ccsds_Socket_Interface_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Ccsds_Socket_Interface_Instance.Active task.
- **Ccsds_Socket_Interface_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Ccsds_Socket_Interface_Instance.Active task.
- **Parameters_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Parameters_Instance.Active task.
- **Parameters_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Parameters_Instance.Active task.
- **Parameters_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Parameters_Instance.Active task.
- **Parameter_Store_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Parameter_Store_Instance.Active task.
- **Parameter_Store_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Parameter_Store_Instance.Active task.
- **Parameter_Store_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Parameter_Store_Instance.Active task.
- **Parameter_Manager_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Parameter_Manager_Instance.Active task.
- **Parameter_Manager_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Parameter_Manager_Instance.Active task.
- **Parameter_Manager_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Parameter_Manager_Instance.Active task.
- **Event_Text_Logger_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Event_Text_Logger_Instance.Active task.
- **Event_Text_Logger_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Event_Text_Logger_Instance.Active task.
- **Event_Text_Logger_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Event_Text_Logger_Instance.Active task.
- **Memory_Packetizer_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Memory_Packetizer_Instance.Active task.
- **Memory_Packetizer_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Memory_Packetizer_Instance.Active task.
- **Memory_Packetizer_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Memory_Packetizer_Instance.Active task.
- **Interrupt_Servicer_Instance_Active_Usage_1** - The period one CPU utilization percentage for the Interrupt_Servicer_Instance.Active task.

- **Interrupt_Servicer_Instance_Active_Usage_2** - The period two CPU utilization percentage for the Interrupt_Servicer_Instance.Active task.
- **Interrupt_Servicer_Instance_Active_Usage_3** - The period three CPU utilization percentage for the Interrupt_Servicer_Instance.Active task.
- **Ccsds_Socket_Interface_Instance_Listener_Usage_1** - The period one CPU utilization percentage for the Ccsds_Socket_Interface_Instance.Listener task.
- **Ccsds_Socket_Interface_Instance_Listener_Usage_2** - The period two CPU utilization percentage for the Ccsds_Socket_Interface_Instance.Listener task.
- **Ccsds_Socket_Interface_Instance_Listener_Usage_3** - The period three CPU utilization percentage for the Ccsds_Socket_Interface_Instance.Listener task.
- **Interrupt_Servicer_Instance_Usage_1** - The period one CPU utilization percentage for the Interrupt_Servicer_Instance interrupt.
- **Interrupt_Servicer_Instance_Usage_2** - The period two CPU utilization percentage for the Interrupt_Servicer_Instance interrupt.
- **Interrupt_Servicer_Instance_Usage_3** - The period three CPU utilization percentage for the Interrupt_Servicer_Instance interrupt.

Linux_Example_Fault_Responses_Packed_Id_Type.T:

This is an autocoded packed ID type for the fault correction component based on the fault IDs that it handles.

Table 58: Linux_Example_Fault_Responses_Packed_Id_Type Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Linux_Example_Fault_Responses_Enums.Fault_Type.E	0 => None 1 => Task_Watchdog_Instance_Slow_Rate_Group_Fault 2 => Task_Watchdog_Instance_Fast_Rate_Group_Fault 3 => Fault_Producer_Instance_Fault_1 4 => Fault_Producer_Instance_Fault_2	16	0	15

Field Descriptions:

- **Id** - The fault type identifier.

Linux_Example_Fault_Responses_Status_Record.T:

This is an autocoded data product status record type for a Fault Correction component.

Table 59: Linux_Example_Fault_Responses_Status_Record Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Task_Watchdog_Instance_Slow_Rate_Group_Fault_Status	Fault_Correction_Enums.Status_Type.E	0 => Disabled 1 => Nominal 2 => Fault_Detected 3 => Fault_Latched	2	0	1

Task_Watchdog_Instance_Fast_Rate_Group_Fault_Status	Fault_Correction_Enums.Status_Type.E	0 => Disabled 1 => Nominal 2 => Fault_Detected 3 => Fault_Latched	2	2	3
Fault_Producer_Instance_Fault_1_Status	Fault_Correction_Enums.Status_Type.E	0 => Disabled 1 => Nominal 2 => Fault_Detected 3 => Fault_Latched	2	4	5
Fault_Producer_Instance_Fault_2_Status	Fault_Correction_Enums.Status_Type.E	0 => Disabled 1 => Nominal 2 => Fault_Detected 3 => Fault_Latched	2	6	7

Field Descriptions:

- **Task_Watchdog_Instance_Slow_Rate_Group_Fault_Status** - The current status of the Task_Watchdog_Instance.Slow_Rate_Group_Fault fault response.
- **Task_Watchdog_Instance_Fast_Rate_Group_Fault_Status** - The current status of the Task_Watchdog_Instance.Fast_Rate_Group_Fault fault response.
- **Fault_Producer_Instance_Fault_1_Status** - The current status of the Fault_Producer_Instance.Fault_1 fault response.
- **Fault_Producer_Instance_Fault_2_Status** - The current status of the Fault_Producer_Instance.Fault_2 fault response.

Linux_Example_Parameter_Table_Record.T:

This is the autocoded parameter table type for the Parameters_Instance component in the Linux_Example assembly.

Table 60: Linux_Example_Parameter_Table_Record Packed Record : 256 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Crc_Calculated	Crc_16.Crc_16_Type	-	16	0	15
Header	Parameter_Table_Header.T	-	48	16	63
Oscillator_A_Frequency	Packed_F32.T	-	32	64	95
Oscillator_A_Amplitude	Packed_F32.T	-	32	96	127
Oscillator_A_Offset	Packed_F32.T	-	32	128	159
Oscillator_B_Frequency	Packed_F32.T	-	32	160	191
Oscillator_B_Amplitude	Packed_F32.T	-	32	192	223
Oscillator_B_Offset	Packed_F32.T	-	32	224	255

Field Descriptions:

- **Crc_Calculated** - The CRC of the parameter table, as computed by the FSW system upon receipt of the uplinked parameter table.
- **Header** - The parameter table header.
- **Oscillator_A_Frequency** - (ID: 1) The frequency of the oscillator in Hz
- **Oscillator_A_Amplitude** - (ID: 2) The amplitude of the oscillator

- **Oscillator_A_Offset** - (ID: 3) The Y offset of the oscillator
- **Oscillator_B_Frequency** - (ID: 4) The frequency of the oscillator in Hz
- **Oscillator_B_Amplitude** - (ID: 5) The amplitude of the oscillator
- **Oscillator_B_Offset** - (ID: 6) The Y offset of the oscillator

Linux_Example_Queue_Monitor_Packet_Type.T:

This is the autocoded packet type for the Queue Monitor component. It contains queue utilization information for queued component in the assembly.

Table 61: Linux_Example_Queue_Monitor_Packet_Type Packed Record : 240 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Slow_Rate_Group_Current_Usage	Basic_Types.Byte	0 to 255	8	0	7
Slow_Rate_Group_Maximum_Usage	Basic_Types.Byte	0 to 255	8	8	15
Fast_Rate_Group_Current_Usage	Basic_Types.Byte	0 to 255	8	16	23
Fast_Rate_Group_Maximum_Usage	Basic_Types.Byte	0 to 255	8	24	31
Watchdog_Rate_Group_Current_Usage	Basic_Types.Byte	0 to 255	8	32	39
Watchdog_Rate_Group_Maximum_Usage	Basic_Types.Byte	0 to 255	8	40	47
Command_Router_Instance_Current_Usage	Basic_Types.Byte	0 to 255	8	48	55
Command_Router_Instance_Maximum_Usage	Basic_Types.Byte	0 to 255	8	56	63
Event_Text_Logger_Instance_Current_Usage	Basic_Types.Byte	0 to 255	8	64	71
Event_Text_Logger_Instance_Maximum_Usage	Basic_Types.Byte	0 to 255	8	72	79
Parameters_Instance_Current_Usage	Basic_Types.Byte	0 to 255	8	80	87
Parameters_Instance_Maximum_Usage	Basic_Types.Byte	0 to 255	8	88	95
Parameter_Store_Instance_Current_Usage	Basic_Types.Byte	0 to 255	8	96	103
Parameter_Store_Instance_Maximum_Usage	Basic_Types.Byte	0 to 255	8	104	111
Parameter_Manager_Instance_Current_Usage	Basic_Types.Byte	0 to 255	8	112	119
Parameter_Manager_Instance_Maximum_Usage	Basic_Types.Byte	0 to 255	8	120	127

Product_Packetizer_ Instance_Current_ Usage	Basic_Types.Byte	0 to 255	8	128	135
Product_Packetizer_ Instance_Maximum_ Usage	Basic_Types.Byte	0 to 255	8	136	143
Ccsds_Socket_ Interface_Instance_ Current_Usage	Basic_Types.Byte	0 to 255	8	144	151
Ccsds_Socket_ Interface_Instance_ Maximum_Usage	Basic_Types.Byte	0 to 255	8	152	159
Memory_Packetizer_ Instance_Current_ Usage	Basic_Types.Byte	0 to 255	8	160	167
Memory_Packetizer_ Instance_Maximum_ Usage	Basic_Types.Byte	0 to 255	8	168	175
Counter_Instance_ Current_Usage	Basic_Types.Byte	0 to 255	8	176	183
Counter_Instance_ Maximum_Usage	Basic_Types.Byte	0 to 255	8	184	191
Oscillator_A_ Current_Usage	Basic_Types.Byte	0 to 255	8	192	199
Oscillator_A_ Maximum_Usage	Basic_Types.Byte	0 to 255	8	200	207
Oscillator_B_ Current_Usage	Basic_Types.Byte	0 to 255	8	208	215
Oscillator_B_ Maximum_Usage	Basic_Types.Byte	0 to 255	8	216	223
Fault_Correction_ Instance_Current_ Usage	Basic_Types.Byte	0 to 255	8	224	231
Fault_Correction_ Instance_Maximum_ Usage	Basic_Types.Byte	0 to 255	8	232	239

Field Descriptions:

- **Slow_Rate_Group_Current_Usage** - The current percent usage of the Slow_Rate_Group internal queue.
- **Slow_Rate_Group_Maximum_Usage** - The maximum percent usage (high water mark) of the Slow_Rate_Group internal queue.
- **Fast_Rate_Group_Current_Usage** - The current percent usage of the Fast_Rate_Group internal queue.
- **Fast_Rate_Group_Maximum_Usage** - The maximum percent usage (high water mark) of the Fast_Rate_Group internal queue.
- **Watchdog_Rate_Group_Current_Usage** - The current percent usage of the Watchdog_Rate_Group internal queue.
- **Watchdog_Rate_Group_Maximum_Usage** - The maximum percent usage (high water mark) of the Watchdog_Rate_Group internal queue.
- **Command_Router_Instance_Current_Usage** - The current percent usage of the Command_Router_Instance internal queue.

- **Command_Router_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Command_Router_Instance internal queue.
- **Event_Text_Logger_Instance_Current_Usage** - The current percent usage of the Event_Text_Logger_Instance internal queue.
- **Event_Text_Logger_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Event_Text_Logger_Instance internal queue.
- **Parameters_Instance_Current_Usage** - The current percent usage of the Parameters_Instance internal queue.
- **Parameters_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Parameters_Instance internal queue.
- **Parameter_Store_Instance_Current_Usage** - The current percent usage of the Parameter_Store_Instance internal queue.
- **Parameter_Store_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Parameter_Store_Instance internal queue.
- **Parameter_Manager_Instance_Current_Usage** - The current percent usage of the Parameter_Manager_Instance internal queue.
- **Parameter_Manager_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Parameter_Manager_Instance internal queue.
- **Product_Packetizer_Instance_Current_Usage** - The current percent usage of the Product_Packetizer_Instance internal queue.
- **Product_Packetizer_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Product_Packetizer_Instance internal queue.
- **Ccsds_Socket_Interface_Instance_Current_Usage** - The current percent usage of the Ccsds_Socket_Interface_Instance internal queue.
- **Ccsds_Socket_Interface_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Ccsds_Socket_Interface_Instance internal queue.
- **Memory_Packetizer_Instance_Current_Usage** - The current percent usage of the Memory_Packetizer_Instance internal queue.
- **Memory_Packetizer_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Memory_Packetizer_Instance internal queue.
- **Counter_Instance_Current_Usage** - The current percent usage of the Counter_Instance internal queue.
- **Counter_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Counter_Instance internal queue.
- **Oscillator_A_Current_Usage** - The current percent usage of the Oscillator_A internal queue.
- **Oscillator_A_Maximum_Usage** - The maximum percent usage (high water mark) of the Oscillator_A internal queue.
- **Oscillator_B_Current_Usage** - The current percent usage of the Oscillator_B internal queue.
- **Oscillator_B_Maximum_Usage** - The maximum percent usage (high water mark) of the Oscillator_B internal queue.
- **Fault_Correction_Instance_Current_Usage** - The current percent usage of the Fault_Correction_Instance internal queue.
- **Fault_Correction_Instance_Maximum_Usage** - The maximum percent usage (high water mark) of the Fault_Correction_Instance internal queue.

Linux_Example_Stack_Monitor_Packet_Type.T:

This is the autocoded packet type for the stack monitor component. It contains stack and secondary stack utilization information for every task in the assembly.

Table 62: Linux_Example_Stack_Monitor_Packet_Type Packed Record : 224 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Fault_Correction_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	0	7
Fault_Correction_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	8	15
Ticker_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	16	23
Ticker_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	24	31
Watchdog_Rate_Group_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	32	39
Watchdog_Rate_Group_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	40	47
Slow_Rate_Group_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	48	55
Slow_Rate_Group_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	56	63
Fast_Rate_Group_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	64	71
Fast_Rate_Group_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	72	79
Command_Router_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	80	87
Command_Router_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	88	95
Ccsds_Socket_Interface_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	96	103
Ccsds_Socket_Interface_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	104	111

Parameters_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	112	119
Parameters_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	120	127
Parameter_Store_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	128	135
Parameter_Store_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	136	143
Parameter_Manager_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	144	151
Parameter_Manager_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	152	159
Event_Text_Logger_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	160	167
Event_Text_Logger_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	168	175
Memory_Packetizer_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	176	183
Memory_Packetizer_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	184	191
Interrupt_Servicer_Instance_Active_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	192	199
Interrupt_Servicer_Instance_Active_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	200	207
Ccsds_Socket_Interface_Instance_Listener_Primary_Stack_Usage	Basic_Types.Byte	0 to 255	8	208	215
Ccsds_Socket_Interface_Instance_Listener_Secondary_Stack_Usage	Basic_Types.Byte	0 to 255	8	216	223

Field Descriptions:

- **Fault_Correction_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Fault_Correction_Instance.Active task.
- **Fault_Correction_Instance_Active_Secondary_Stack_Usage** - The secondary

stack utilization percentage for the Fault_Correction_Instance.Active task.

- **Ticker_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Ticker_Instance.Active task.
- **Ticker_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Ticker_Instance.Active task.
- **Watchdog_Rate_Group_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Watchdog_Rate_Group.Active task.
- **Watchdog_Rate_Group_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Watchdog_Rate_Group.Active task.
- **Slow_Rate_Group_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Slow_Rate_Group.Active task.
- **Slow_Rate_Group_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Slow_Rate_Group.Active task.
- **Fast_Rate_Group_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Fast_Rate_Group.Active task.
- **Fast_Rate_Group_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Fast_Rate_Group.Active task.
- **Command_Router_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Command_Router_Instance.Active task.
- **Command_Router_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Command_Router_Instance.Active task.
- **Ccsds_Socket_Interface_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Ccsds_Socket_Interface_Instance.Active task.
- **Ccsds_Socket_Interface_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Ccsds_Socket_Interface_Instance.Active task.
- **Parameters_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Parameters_Instance.Active task.
- **Parameters_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Parameters_Instance.Active task.
- **Parameter_Store_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Parameter_Store_Instance.Active task.
- **Parameter_Store_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Parameter_Store_Instance.Active task.
- **Parameter_Manager_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Parameter_Manager_Instance.Active task.
- **Parameter_Manager_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Parameter_Manager_Instance.Active task.
- **Event_Text_Logger_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Event_Text_Logger_Instance.Active task.
- **Event_Text_Logger_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Event_Text_Logger_Instance.Active task.
- **Memory_Packetizer_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Memory_Packetizer_Instance.Active task.
- **Memory_Packetizer_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Memory_Packetizer_Instance.Active task.
- **Interrupt_Servicer_Instance_Active_Primary_Stack_Usage** - The primary stack utilization percentage for the Interrupt_Servicer_Instance.Active task.

- **Interrupt_Servicer_Instance_Active_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Interrupt_Servicer_Instance.Active task.
- **Ccsds_Socket_Interface_Instance_Listener_Primary_Stack_Usage** - The primary stack utilization percentage for the Ccsds_Socket_Interface_Instance.Listener task.
- **Ccsds_Socket_Interface_Instance_Listener_Secondary_Stack_Usage** - The secondary stack utilization percentage for the Ccsds_Socket_Interface_Instance.Listener task.

Linux_Example_Task_Watchdog_List_State_Record.T:

This is an autocoded data product status record type for a Task Watchdog component regarding the status of checking each connector for pets.

Table 63: Linux_Example_Task_Watchdog_List_State_Record Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Slow_Rate_Group_State	Task_Watchdog_Enums.Watchdog_Action_State.E	0 => Disabled 1 => Warn 2 => Error_Fault	2	0	1
Fast_Rate_Group_State	Task_Watchdog_Enums.Watchdog_Action_State.E	0 => Disabled 1 => Warn 2 => Error_Fault	2	2	3
Reserved_0	Task_Watchdog_Types.Two_Bit_Padding_Type	0 to 3	2	4	5
Reserved_1	Task_Watchdog_Types.Two_Bit_Padding_Type	0 to 3	2	6	7

Field Descriptions:

- **Slow_Rate_Group_State** - The current state of the Slow_Rate_Group watchdog action.
- **Fast_Rate_Group_State** - The current state of the Fast_Rate_Group watchdog action.
- **Reserved_0** - Padding bits, not used.
- **Reserved_1** - Padding bits, not used.

Linux_Example_Task_Watchdog_List_Watchdog_Action_Cmd.T:

This is an autocoded packed enumeration record which contains an enumeration literal for each task that the task watchdog manages. This record contains information for changing the limit of a specific watchdog connector via this enumeration.

Preamble (inline Ada definitions):

```ada
type Task_Enum_Type is (
  Slow_Rate_Group,
  Fast_Rate_Group
);
for Task_Enum_Type use (
  Slow_Rate_Group => 1,
  Fast_Rate_Group => 2
);
```

Table 64: Linux_Example_Task_Watchdog_List_Watchdog_Action_Cmd Packed Record : 24 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Watchdog_Task	Task_Enum_Type	1 => Slow_Rate_Group 2 => Fast_Rate_Group	16	0	15
New_Action	Task_Watchdog_Enums.Watchdog_Action_State.E	0 => Disabled 1 => Warn 2 => Error_Fault	8	16	23

Field Descriptions:

- **Watchdog_Task** - The task watchdog task enumeration.
- **New_Action** - The new value of the action for the specific associated connector

Linux_Example_Task_Watchdog_List_Watchdog_Limit_Cmd.T:

This is an autocoded packed enumeration record which contains an enumeration literal for each task that the task watchdog manages. This record contains information for changing the limit of a specific watchdog connector via this enumeration.

Preamble (inline Ada definitions):

```
type Task_Enum_Type is (
  Slow_Rate_Group,
  Fast_Rate_Group
);
for Task_Enum_Type use (
  Slow_Rate_Group => 1,
  Fast_Rate_Group => 2
);
```

Table 65: Linux_Example_Task_Watchdog_List_Watchdog_Limit_Cmd Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Watchdog_Task	Task_Enum_Type	1 => Slow_Rate_Group 2 => Fast_Rate_Group	16	0	15
New_Limit	Task_Watchdog_Types.Missed_Pet_Limit_Type	1 to 65534	16	16	31

Field Descriptions:

- **Watchdog_Task** - The task watchdog task enumeration.
- **New_Limit** - The new value of the limit for the specific associated connector

Logger_Error.T:

A packed record which holds status information about a failed log attempt.

Table 66: Logger_Error Packed Record : 40 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Num_Bytes_Logged	Natural	0 to 2147483647	32	0	31
Status	Logger_Enums.Log_Attempt_Status.E	0 => Success 1 => Serialization_Failure 2 => Too_Full	8	32	39

Field Descriptions:

- **Num_Bytes_Logged** - The number of bytes that was attempted to store.
- **Status** - The returned status from the log attempt.

Logger_Info.T:

A packed record which holds information about the internal status of the log.

Table 67: Logger_Info Packed Record : 104 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Meta_Data	Circular_Buffer_Meta.T	-	96	0	95
Current_Mode	Logger_Enums.Logger_Mode.E	0 => Disabled 1 => Enabled	8	96	103

Field Descriptions:

- **Meta_Data** - The current meta data of the internal circular buffer.
- **Current_Mode** - Is the log enabled or disabled?

Logger_Status.T:

A packed record which holds the enabled/disabled state of the logger

Table 68: Logger_Status Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Current_Mode	Logger_Enums.Logger_Mode.E	0 => Disabled 1 => Enabled	8	0	7

Field Descriptions:

- **Current_Mode** - Is the log enabled or disabled?

Memory_Region.T:

A memory region described by a system address and length (in bytes).

Table 69: Memory_Region Packed Record : 96 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Address	System.Address	-	64	0	63
Length	Natural	0 to 2147483647	32	64	95

Field Descriptions:

- **Address** - The starting address of the memory region.
- **Length** - The number of bytes at the given address to associate with this memory region.

Operands.T:

This is a type that contains a left and right side

Table 70: Operands Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Left	Interfaces. Unsigned_16	0 to 65535	16	0	15
Right	Interfaces. Unsigned_16	0 to 65535	16	16	31

Field Descriptions:

- **Left** - The left side
- **Right** - The right side

Packed_Address.T:

A packed system address.

Table 71: Packed_Address Packed Record : 64 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Address	System.Address	-	64	0	63

Field Descriptions:

- **Address** - The starting address of the memory region.

Packed_Connector_Index.T:

Single component record for holding packed connector index.

Table 72: Packed_Connector_Index Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Index	Connector_Types. Connector_Index_ Type	1 to 65535	16	0	15

Field Descriptions:

- **Index** - The 16-bit connector index.

Packed_Enable_Disable_Type.T:

Single component record for holding an enable/disable enumeration.

Table 73: Packed_Enable_Disable_Type Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
State	Basic_Enums.Enable_Disable_Type.E	0 => Disabled 1 => Enabled	8	0	7

Field Descriptions:

- **State** - The 8-bit enable disable enumeration.

Packed_Exception_Occurrence.T:

Packed record which holds information from an Ada Exception Occurrence type. This is the type passed into the Last Chance Handler when running a full runtime.

Preamble (inline Ada definitions):

```ada
type Exception_Name_Buffer is new Basic_Types.Byte_Array (0 .. 99)
  with Size => 100 * 8,
       Object_Size => 100 * 8;
type Exception_Message_Buffer is new Basic_Types.Byte_Array (0 .. 299)
  with Size => 300 * 8,
       Object_Size => 300 * 8;
```

Table 74: Packed_Exception_Occurrence Packed Record : 9632 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Exception_Name	Exception_Name_Buffer	-	800	0	799
Exception_Message	Exception_Message_Buffer	-	2400	800	3199
Stack_Trace_Depth	Interfaces.Unsigned_32	0 to 4294967295	32	3200	3231
Stack_Trace	Stack_Trace_Addresses.T	-	6400	3232	9631

Field Descriptions:

- **Exception_Name** - The exception name.
- **Exception_Message** - The exception message.
- **Stack_Trace_Depth** - The depth of the reported stack trace.
- **Stack_Trace** - The stack trace addresses.

Packed_F32.T:

Single component record for holding packed 32-bit floating point number.

Table 75: Packed_F32 Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Value	Short_Float	-3.40282e+38 to 3.40282e+38	32	0	31

Field Descriptions:

- **Value** - The 32-bit floating point number.

Packed_Fault_Id.T:

A packed record which holds an fault identifier.

Table 76: Packed_Fault_Id Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Fault_Types.Fault_Id	0 to 65535	16	0	15

Field Descriptions:

- **Id** - The fault identifier

Packed_Missed_Pet_Limit.T:

Limit value packed for the task watchdog data products

Table 77: Packed_Missed_Pet_Limit Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Limit	Task_Watchdog_Types.Missed_Pet_Limit_Type	1 to 65534	16	0	15

Field Descriptions:

- **Limit** - The limit value

Packed_Natural.T:

Single component record for holding packed Natural value.

Table 78: Packed_Natural Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Value	Natural	0 to 2147483647	32	0	31

Field Descriptions:

- **Value** - The 32-bit Natural Integer.

Packed_Parameter_Table_Copy_Type.T:

Packed record which holds the Parameter Manager copy type.

Table 79: Packed_Parameter_Table_Copy_Type Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Copy_Type	Parameter_ Manager_ Enums. Parameter_ Table_Copy_ Type.E	0 => Default_To_Working 1 => Working_To_Default	8	0	7

Field Descriptions:

- **Copy_Type** - This enumeration describes the source and destination of the Parameter Manager copy command.

Packed_Positive_Length.T:

Single component record for holding packed Positive value that represents a length.

Table 80: Packed_Positive_Length Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Length	Positive	1 to 2147483647	32	0	31

Field Descriptions:

- **Length** - The 32-bit Positive Integer that represents a length.

Packed_U16.T:

Single component record for holding packed unsigned 16-bit value.

Table 81: Packed_U16 Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Value	Interfaces. Unsigned_16	0 to 65535	16	0	15

Field Descriptions:

- **Value** - The 16-bit unsigned integer.

Packed_U32.T:

Single component record for holding packed unsigned 32-bit value.

Table 82: Packed_U32 Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Value	Interfaces. Unsigned_32	0 to 4294967295	32	0	31

Field Descriptions:

- **Value** - The 32-bit unsigned integer.

Packed_Watchdog_Component_State.T:

State value packed for the task watchdog data products

Table 83: Packed_Watchdog_Component_State Packed Record : 8 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
State	Task_Watchdog_ Enums.Watchdog_ Enabled_State.E	0 => Disabled 1 => Enabled	8	0	7

Field Descriptions:

- **State** - The state of the watchdog component to check all upstream petters.

Packet.T:

Generic packet for holding arbitrary data

Table 84: Packet Packed Record : 10080 bits *(maximum)*

Name	Type	Range	Size (Bits)	Start Bit	End Bit	Variable Length
Header	Packet_ Header.T	-	112	0	111	-
Buffer	Packet_ Types.Packet_ Buffer_Type	-	9968	112	10079	Header. Buffer_Length

Field Descriptions:

- **Header** - The packet header
- **Buffer** - A buffer that contains the packet data

Packet_Data_Product_Ids.T:

A packed record which holds a packet identifier and data product identifier.

Table 85: Packet_Data_Product_Ids Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Packet_Id	Packet_Types. Packet_Id	0 to 65535	16	0	15
Data_Product_Id	Data_Product_Types. Data_Product_Id	0 to 65535	16	16	31

Field Descriptions:

- **Packet_Id** - The packet identifier
- **Data_Product_Id** - The data product identifier

Packet_Header.T:

Generic packet header for holding arbitrary data

Table 86: Packet_Header Packed Record : 112 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Time	Sys_Time.T	-	64	0	63
Id	Packet_Types. Packet_Id	0 to 65535	16	64	79
Sequence_Count	Packet_Types. Sequence_Count_Mod_ Type	0 to 16383	16	80	95
Buffer_Length	Packet_Types. Packet_Buffer_ Length_Type	0 to 1246	16	96	111

Field Descriptions:

- **Time** - The timestamp for the packet item.
- **Id** - The packet identifier
- **Sequence_Count** - Packet Sequence Count
- **Buffer_Length** - The number of bytes used in the packet buffer

Packet_Id.T:

A packed record which holds a packet identifier.

Table 87: Packet_Id Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Packet_Types. Packet_Id	0 to 65535	16	0	15

Field Descriptions:

- **Id** - The packet identifier

Packet_Period.T:

A packed record which holds a packet identifier and period.

Table 88: Packet_Period Packed Record : 48 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Packet_Types. Packet_Id	0 to 65535	16	0	15
Period	Natural	0 to 2147483647	32	16	47

Field Descriptions:

- **Id** - The packet identifier
- **Period** - The packet period, in ticks

Packets_Per_Period.T:

A record which holds information on how to set a maximum packet rate.

Table 89: Packets_Per_Period Packed Record : 64 bits

Name	**Type**	**Range**	**Size (Bits)**	**Start Bit**	**End Bit**
Max_Packets	Natural	0 to 2147483647	32	0	31
Period	Positive	1 to 2147483647	32	32	63

Field Descriptions:

- **Max_Packets** - The maximum number of packets to send out in one time period.
- **Period** - The length of a time period, defined in seconds.

Parameter.T:

Generic parameter packet for holding a generic parameter

Table 90: Parameter Packed Record : 280 bits *(maximum)*

Name	**Type**	**Range**	**Size (Bits)**	**Start Bit**	**End Bit**	**Variable Length**
Header	Parameter_ Header.T	-	24	0	23	-
Buffer	Parameter_ Types. Parameter_ Buffer_Type	-	256	24	279	Header.Buffer_ Length

Field Descriptions:

- **Header** - The parameter header
- **Buffer** - A buffer to that contains the parameter type

Parameter_Header.T:

Generic parameter header for holding arbitrary parameters

Table 91: Parameter_Header Packed Record : 24 bits

Name	**Type**	**Range**	**Size (Bits)**	**Start Bit**	**End Bit**
Id	Parameter_Types. Parameter_Id	0 to 65535	16	0	15
Buffer_Length	Parameter_Types. Parameter_Buffer_ Length_Type	0 to 32	8	16	23

Field Descriptions:

- **Id** - The parameter identifier
- **Buffer_Length** - The number of bytes used in the parameter type buffer

Parameter_Id.T:

Packed record used for holding a parameter Id.

Table 92: Parameter_Id Packed Record : 16 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Id	Parameter_Types.Parameter_Id	0 to 65535	16	0	15

Field Descriptions:

- **Id** - The parameter identifier

Parameter_Operation_Status.T:

A record that can be used to report the operation and status of a certain parameter ID.

Table 93: Parameter_Operation_Status Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Operation	Parameter_Enums.Parameter_Operation_Type.E	0 => Stage 1 => Update 2 => Fetch	8	0	7
Status	Parameter_Enums.Parameter_Update_Status.E	0 => Success 1 => Id_Error 2 => Validation_Error 3 => Length_Error	8	8	15
Id	Parameter_Types.Parameter_Id	0 to 65535	16	16	31

Field Descriptions:

- **Operation** - The parameter operation to perform.
- **Status** - The parameter return status.
- **Id** - The parameter identifier

Parameter_Table_Header.T:

A packed record which holds parameter table header data. This data is asswill be prepended to the table data upon upload.

Preamble (inline Ada definitions):

```ada
-- Declare the start index at which to begin calculating the CRC. The
-- start index is dependent on this type, and so is declared here so that
-- it is easier to keep in sync.
Crc_Section_Length : constant Natural := Crc_16.Crc_16_Type'Length;
Version_Length : constant Natural := 4;
```

Table 94: Parameter_Table_Header Packed Record : 48 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Crc_Table	Crc_16.Crc_16_Type	-	16	0	15
Version	Short_Float	-3.40282e+38 to 3.40282e+38	32	16	47

Field Descriptions:

- **Crc_Table** - The CRC of the parameter table, as computed by a ground system, and uplinked with the table.
- **Version** - The current version of the parameter table.

Parameter_Update.T:

A record intended to be used as a provide/modify connector type for updating/fetching parameters.

Table 95: Parameter_Update Packed Record : 296 bits *(maximum)*

Name	Type	Range	Size (Bits)	Start Bit	End Bit	Variable Length
Operation	Parameter_Enums.Parameter_Operation_Type.E	0 => Stage 1 => Update 2 => Fetch	8	0	7	-
Status	Parameter_Enums.Parameter_Update_Status.E	0 => Success 1 => Id_Error 2 => Validation_Error 3 => Length_Error	8	8	15	-
Param	Parameter.T	-	280	16	295	-

Field Descriptions:

- **Operation** - The parameter operation to perform.
- **Status** - The parameter return status.
- **Param** - The parameter that has been updated or fetched.

Parameters_Memory_Region.T:

A packed record which holds the parameter memory region to operate on as well as an enumeration specifying the operation to perform.

Table 96: Parameters_Memory_Region Packed Record : 104 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Region	Memory_Region.T	-	96	0	95

Operation	Parameter_Enums. Parameter_Table_ Operation_Type.E	0 => Get 1 => Set	8	96	103

Field Descriptions:

- **Region** - The memory region.
- **Operation** - The parameter table operation to perform.

Parameters_Memory_Region_Release.T:

A packed record which holds the parameter memory region to release as well as the status returned from the parameter update operation.

Table 97: Parameters_Memory_Region_Release Packed Record : 104 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Region	Memory_Region.T	-	96	0	95
Status	Parameter_Enums.Parameter_Table_Update_Status.E	0 => Success 1 => Length_Error 2 => Crc_Error 3 => Parameter_Error 4 => Dropped	8	96	103

Field Descriptions:

- **Region** - The memory region.
- **Status** - The return status from the parameter update operation.

Pet.T:

The pet datatype is used for servicing a watchdog. Included in this type is a count.

Table 98: Pet Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Count	Interfaces.Unsigned_32	0 to 4294967295	32	0	31

Field Descriptions:

- **Count** - The cycle number of the pet.

Socket_Address.T:

This is a type that contains the IP address (host) and port number of a socket connection.

Table 99: Socket_Address Packed Record : 64 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Ip_Address	Gnat.Sockets.Inet_ Addr_V4_Type	-	32	0	31
Port	Gnat.Sockets.Port_ Type	0 to 65535	32	32	63

Field Descriptions:

- **Ip_Address** - The host or IP address number.
- **Port** - The port number for the connection.

Stack_Trace_Addresses.T:

An array of packed addresses in big endian. This is sized to easily fit a normal stack trace.

Table 100: Stack_Trace_Addresses Packed Array : 6400 bits

Type	Range	Element Size (Bits)	Length	Total Size (Bits)
Packed_Address.T	-	64	100	6400

Sys_Time.T:

A record which holds a time stamp using GPS format including seconds and subseconds since epoch (1-5-1980 to 1-6-1980 midnight).

Table 101: Sys_Time Packed Record : 64 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Seconds	Interfaces. Unsigned_32	0 to 4294967295	32	0	31
Subseconds	Interfaces. Unsigned_32	0 to 4294967295	32	32	63

Field Descriptions:

- **Seconds** - The number of seconds elapsed since epoch.
- **Subseconds** - The number of 1/(2^32) sub-seconds.

Task_Timing_Report.T:

Record which holds timing reports for the all-time maximum and recent maximum of an executing task

Table 102: Task_Timing_Report Packed Record : 256 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Max	Timing_Report.T	-	128	0	127
Recent_Max	Timing_Report.T	-	128	128	255

Field Descriptions:

- **Max** - The maximum recorded timing report since start up.

- **Recent_Max** - The maximum recorded timing report during some recent time invterval.

Td_Full_Queue_Param.T:

This is a type that contains useful information about a component full queue.

Table 103: Td_Full_Queue_Param Packed Record : 112 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Dropped_Tick	Tick.T	-	96	0	95
Index	Connector_Types. Connector_Index_ Type	1 to 65535	16	96	111

Field Descriptions:

- **Dropped_Tick** - The tick during which the component's queue was found to be full.
- **Index** - The tick divider index number of the component that had a full queue.

Tick.T:

The tick datatype used for periodic scheduling. Included in this type is the Time associated with a tick and a count.

Table 104: Tick Packed Record : 96 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Time	Sys_Time.T	-	64	0	63
Count	Interfaces. Unsigned_32	0 to 4294967295	32	64	95

Field Descriptions:

- **Time** - The timestamp associated with the tick.
- **Count** - The cycle number of the tick.

Time_Exceeded.T:

Datatype used for reporting time deltas at a particular count number.

Table 105: Time_Exceeded Packed Record : 96 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Time_Delta	Delta_Time.T	-	64	0	63
Count	Interfaces. Unsigned_32	0 to 4294967295	32	64	95

Field Descriptions:

- **Time_Delta** - The time delta.
- **Count** - The cycle number of the tick.

Timing_Report.T:

Record which holds wall time and execution time to describe the runtime performance some piece of code.

Table 106: Timing_Report Packed Record : 128 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Wall_Time	Delta_Time.T	-	64	0	63
Execution_Time	Delta_Time.T	-	64	64	127

Field Descriptions:

- **Wall_Time** - The wall time associated with the measurement.
- **Execution_Time** - The time the task spent executing on the CPU.

Watchdog_Action_Cmd.T:

This record contains information for changing the limit of a specific watchdog connector.

Table 107: Watchdog_Action_Cmd Packed Record : 24 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Index	Connector_Types.Connector_Index_Type	1 to 65535	16	0	15
New_Action	Task_Watchdog_Enums.Watchdog_Action_State.E	0 => Disabled 1 => Warn 2 => Error_Fault	8	16	23

Field Descriptions:

- **Index** - The index of the connector that the command wants to change the limit of
- **New_Action** - The new value of the action for the specific associated connector

Watchdog_Limit_Cmd.T:

This record contains information for changing the limit of a specific watchdog connector.

Table 108: Watchdog_Limit_Cmd Packed Record : 32 bits

Name	Type	Range	Size (Bits)	Start Bit	End Bit
Index	Connector_Types.Connector_Index_Type	1 to 65535	16	0	15
New_Limit	Task_Watchdog_Types.Missed_Pet_Limit_Type	1 to 65534	16	16	31

Field Descriptions:

- **Index** - The index of the connector that the command wants to change the limit of
- **New_Limit** - The new value of the limit for the specific associated connector

3.3 Enumerations

The following section outlines any enumerations used in the assembly.

Basic_Enums.Enable_Disable_Type.E:

This enumeration includes enable and disable state.

Table 109: Enable_Disable_Type Literals:

Name	Value	Description
Disabled	0	The state is disabled.
Enabled	1	The state is enabled.

Ccsds_Enums.Ccsds_Packet_Type.E:

This single bit is used to identify that this is a Telecommand Packet or a Telemetry Packet. A Telemetry Packet has this bit set to value 0; therefore, for all Telecommand Packets Bit 3 shall be set to value 1.

Table 110: Ccsds_Packet_Type Literals:

Name	Value	Description
Telemetry	0	Indicates a telemetry packet
Telecommand	1	Indicates a telecommand packet

Ccsds_Enums.Ccsds_Secondary_Header_Indicator.E:

This one bit flag signals the presence (Bit 4 = 1) or absence (Bit 4 = 0) of a Secondary Header data structure within the packet.

Table 111: Ccsds_Secondary_Header_Indicator Literals:

Name	Value	Description
Secondary_Header_Not_Present	0	Indicates that the secondary header is not present within the packet
Secondary_Header_Present	1	Indicates that the secondary header is present within the packet

Ccsds_Enums.Ccsds_Sequence_Flag.E:

This flag provides a method for defining whether this packet is a first, last, or intermediate component of a higher layer data structure.

Table 112: Ccsds_Sequence_Flag Literals:

Name	Value	Description
Continuationsegment	0	Continuation component of higher data structure
Firstsegment	1	First component of higher data structure
Lastsegment	2	Last component of higher data structure
Unsegmented	3	Standalone packet

Command_Enums.Command_Response_Status.E:

This status enumerations provides information on the success/failure of a command through the command response connector.

Table 113: Command_Response_Status Literals:

Name	Value	Description
Success	0	Command was passed to the handler and successfully executed.
Failure	1	Command was passed to the handler not successfully executed.
Id_Error	2	Command id was not valid.
Validation_Error	3	Command parameters were not successfully validated.
Length_Error	4	Command length was not correct.
Dropped	5	Command overflowed a component queue and was dropped.
Register	6	This status is used to register a command with the command routing system.
Register_Source	7	This status is used to register command sender's source id with the command router for command response forwarding.

Data_Product_Enums.Fetch_Status.E:

This status denotes whether a data product fetch was successful.

Table 114: Fetch_Status Literals:

Name	Value	Description
Success	0	The data product was returned successfully.
Not_Available	1	No data product is yet available for the provided id.
Id_Out_Of_Range	2	The data product id was out of range.

Event_Filter_Entry_Enums.Global_Filter_State.E:

This is an enumeration indicating if the filter component is enabled/disabled

Table 115: Global_Filter_State Literals:

Name	Value	Description
Disabled	0	Individual event states will be ignored and no events will be filtered
Enabled	1	Individual event states will be used to determine if the event needs to be filtered

Event_Filter_Enums.Issue_Packet_Type.E:

An enumeration for commands once the state is change for ground to determine if they want to send the state packet or not

Table 116: Issue_Packet_Type Literals:

Name	Value	Description
No_Issue	0	Packet will not be issued

Issue	1	Packet will be issued

Event_Limiter_Enums.Issue_Packet_Type.E:

An enumeration for commands once the state is change for ground to determine if they want to send the state packet or not

Table 117: Issue_Packet_Type Literals:

Name	**Value**	**Description**
No_Issue	0	Packet will not be issued
Issue	1	Packet will be issued

Fault_Correction_Enums.Status_Type.E:

This status enumerations provides the status of a fault response.

Table 118: Status_Type Literals:

Name	**Value**	**Description**
Disabled	0	The fault response is disabled.
Nominal	1	The fault response is enabled and no fault has been detected.
Fault_Detected	2	A fault has been detected and a command response has been issued.
Fault_Latched	3	A fault has been detected and a command response has been issued. The fualt has been latched, so any more faults received with this ID will not issue another response until the this fault response has been unlatched.

Linux_Example_Fault_Responses_Enums.Fault_Type.E:

This enumerations lists all the possible fault conditions possible in the assembly. The value of the enumeration corresponds to the fault identifier for that fault.

Table 119: Fault_Type Literals:

Name	**Value**	**Description**
None	0	No fault present.
Task_Watchdog_Instance_Slow_Rate_Group_Fault	1	Throw noop with value 1 if this fault occurs.
Task_Watchdog_Instance_Fast_Rate_Group_Fault	2	Throw noop with value 2 if this fault occurs.
Fault_Producer_Instance_Fault_1	3	Throw noop with value 3 if this fault occurs.
Fault_Producer_Instance_Fault_2	4	Throw noop with value 4 if this fault occurs.

Logger_Enums.Logger_Mode.E:

This flag denotes whether the log is currently enabled or disabled.

Table 120: Logger_Mode Literals:

Name	Value	Description
Disabled	0	The log is disabled and not currently logging data.
Enabled	1	The log is enabled and currently logging data.

Logger_Enums.Log_Attempt_Status.E:

This enumerations returns the status of a log attempt.

Table 121: Log_Attempt_Status Literals:

Name	Value	Description
Success	0	Log action was successful.
Serialization_Failure	1	Logging failed due to a serialization error.
Too_Full	2	Logging failed because the log was too full to fit the data.

Parameter_Enums.Parameter_Table_Update_Status.E:

This status enumerations provides information on the success/failure of a parameter table update.

Table 122: Parameter_Table_Update_Status Literals:

Name	Value	Description
Success	0	Parameter was successfully staged.
Length_Error	1	Parameter table length was not correct.
Crc_Error	2	The computed CRC of the table does not match the stored CRC.
Parameter_Error	3	Failed to fetch or update an individual parameter within a component.
Dropped	4	The operation could not be performed because it was dropped from a full queue.

Parameter_Enums.Parameter_Operation_Type.E:

This enumeration lists the different parameter operations that can be performed.

Table 123: Parameter_Operation_Type Literals:

Name	Value	Description
Stage	0	Stage the parameter.
Update	1	All parameters are staged, it is ok to update all parameters now.
Fetch	2	Fetch the parameter.

Parameter_Enums.Parameter_Update_Status.E:

This status enumerations provides information on the success/failure of a parameter operation.

Table 124: Parameter_Update_Status Literals:

Name	Value	Description
Success	0	Parameter was successfully staged.
Id_Error	1	Parameter id was not valid.
Validation_Error	2	Parameter values were not successfully validated.
Length_Error	3	Parameter length was not correct.

Parameter_Enums.Parameter_Table_Operation_Type.E:

This enumeration lists the different parameter table operations that can be performed.

Table 125: Parameter_Table_Operation_Type Literals:

Name	Value	Description
Get	0	Retrieve the current values of the parameters.
Set	1	Set the current values of the parameters.

Parameter_Manager_Enums.Parameter_Table_Copy_Type.E:

This enumeration describes the source and destination of the Parameter Manager copy command.

Table 126: Parameter_Table_Copy_Type Literals:

Name	Value	Description
Default_To_Working	0	Copy the parameter table from Default (NVRAM) to Working (RAM).
Working_To_Default	1	Copy the parameter table from Working (RAM) to Default (NVRAM).

Task_Watchdog_Enums.Watchdog_Enabled_State.E:

The state for if each watchdog is enabled or disabled for checking

Table 127: Watchdog_Enabled_State Literals:

Name	Value	Description
Disabled	0	
Enabled	1	

Task_Watchdog_Enums.Watchdog_Action_State.E:

The state for if each watchdog for which action it should take.

Table 128: Watchdog_Action_State Literals:

Name	Value	Description
Disabled	0	
Warn	1	
Error_Fault	2	

Two_Counter_Entry_Enums.Event_State_Type.E:

This is a single bit identifying if the event limiter is enabled for a particular ID.

Table 129: Event_State_Type Literals:

Name	**Value**	**Description**
Disabled	0	Event will not be limited
Enabled	1	Event will be limited