Example Component

Component Design Document

1 Description

This is the example component.

2 Requirements

The requirements for the Example Component are specified below.

1. The component shall be passive and have less than three connectors.
2. The component shall send a packet to any connected downstream component whenever it is scheduled to run.
3. The packet contents shall include a counter in the first data byte that increments with every schedule tick received.

3 Design

3.1 At a Glance

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

- **Execution** - *passive*
- **Number of Connectors** - 3
- **Number of Invokee Connectors** - 1
- **Number of Invoker Connectors** - 2
- **Number of Generic Connectors** - *None*
- **Number of Generic Types** - *None*
- **Number of Unconstrained Arrayed Connectors** - 1
- **Number of Commands** - *None*
- **Number of Parameters** - *None*
- **Number of Events** - *None*
- **Number of Faults** - *None*
- **Number of Data Products** - *None*
- **Number of Data Dependencies** - *None*

- **Number of Packets** - *None*

3.2 Diagram

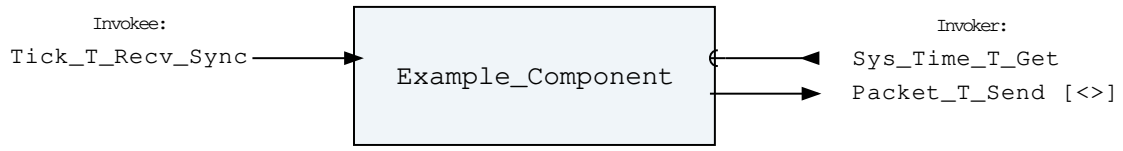

Figure 1: Example Component component diagram.

3.3 Connectors

Below are tables listing the component's connectors.

3.3.1 Invokee Connectors

The following is a list of the component's *invokee* connectors:

Table 1: Example Component Invokee Connectors

Name	Kind	Type	Return_Type	Count
Tick_T_Recv_Sync	recv_sync	Tick.T	-	1

Connector Descriptions:
- **Tick_T_Recv_Sync** - This connector provides the schedule tick for the component.

3.3.2 Invoker Connectors

The following is a list of the component's *invoker* connectors:

Table 2: Example Component Invoker Connectors

Name	Kind	Type	Return_Type	Count
Sys_Time_T_Get	get	-	Sys_Time.T	1
Packet_T_Send	send	Packet.T	-	<>

Connector Descriptions:
- **Sys_Time_T_Get** - This connector is used to fetch the current system time.
- **Packet_T_Send** - This connector is used to send out a telemetry packet.

3.4 Initialization

Below are details on how the component should be initialized in an assembly.

3.4.1 Component Instantiation

This component contains no instantiation parameters in its discriminant.

3.4.2 Component Base Initialization

This component achieves base class initialization using the `init_Base` subprogram. This subprogram requires the following parameters:

Table 3: Example Component Base Initialization Parameters

Name	Type
`Packet_T_Send_Count`	`Connector_Count_Type`

Parameter Descriptions:

- **`Packet_T_Send_Count`** - The size of the Packet_T_Send invoker connector array.

3.4.3 Component Set ID Bases

This component contains no commands, events, packets, faults or data products that need base indentifiers.

3.4.4 Component Map Data Dependencies

This component contains no data dependencies.

3.4.5 Component Implementation Initialization

This component contains no implementation class initialization, meaning there is no `init` subprogram for this component.

4 Unit Tests

The following section describes the unit test suites written to test the component.

4.1 *Example_ Component_ Tests* Test Suite

This is a set of unit tests for the Example Component.

Test Descriptions:

- **`Test_That_Should_Pass`** - This test should pass.
- **`Test_That_Should_Fail`** - This test should not pass.

4.2 *Example_ Component_ Tests* Test Suite

This is a set of unit tests for the Example Component.

Test Descriptions:

- **`Test_That_Should_Pass`** - This test should pass.
- **`Test_That_Should_Fail`** - This test should not pass.

4.3 *Example_ Component_ Tests* Test Suite

This is a set of unit tests for the Example Component.

Test Descriptions:

- **Test_That_Should_Pass** - This test should pass.
- **Test_That_Should_Fail** - This test should not pass.

4.4 *Example_ Component_ Tests* Test Suite

This is a set of unit tests for the Example Component.

Test Descriptions:

- **Test_That_Should_Pass** - This test should pass.
- **Test_That_Should_Fail** - This test should not pass.

5 Appendix

5.1 Packed Types

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

Packet.T:

Generic packet for holding arbitrary data

Table 4: 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_Header.T:

Generic packet header for holding arbitrary data

Table 5: 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

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 6: 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.

Tick.T:

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

Table 7: 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.