

# TI C2000 Toolbox SCI Receive (Generic)

This document describes *SCI Receive (Generic)* component from TI C2000 Toolbox library.

## Short description

*SCI Receive (Generic)* component enables user to configure data transmitting operation of the *Serial Communication Interface* peripheral. It is used to specify the type of connection with the receiver and message configuration and scheduling.

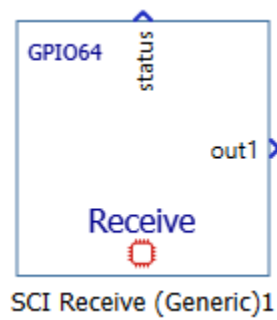


Figure 1. SCI Receive (Generic) - component icon.

## Detailed overview

**NOTE:** It is recommended to select *target platform* on [TI C2000 Setup](#) component before configuring the component.

### Component properties:

- Tab **General:**
  - Access port – specifies type of connection:
    - **DIO** (digital inputs/outputs) – communication is executed between digital outputs of the HIL device and digital inputs of controller board header pins. In this case, serial communication must be configured on the HIL side as well (HIL is a transmitter), by configuring [Serial Setup](#) and [Serial Send](#) components in the model, outside of exported subsystem. It is not required to have an USB connection to the controller after flashing the code.
    - **USB** – communication is executed between controller USB connector and HIL SCADA (the PC application, not the HIL device). In this case, serial communication must be configured on the HIL SCADA side as well (SCADA is a transmitter), by configuring [SCI](#)

[Setup widget](#) and [SCI Write widget](#) on SCADA panel. It is required to have an USB connection with the controller.

- Transmitter (HIL device or HIL SCADA) and receiver (controller) must have the same setup parameters (such as connection type, baud rate, parity, stop bits, data bits) and start-of-frame, end-of-frame, message configuration settings – for both types of connections!
  - Transmitting HIL DO number – visible when *Access port* is set to *DIO*. Specifies the number of HIL digital output from which the controller will receive messages. It lists all available HIL digital outputs that are connected to the digital inputs of the controller that have SCI mux option as receiver (RX). Also, signal *uart tx data* must be applied to selected HIL DO in [Initial Settings component](#).
  - Interface type - select interface board that is used, currently supported boards are '[HIL TI Launchpad Interface](#)' and '[HIL TI uGrid Launchpad Interface](#)', '[HIL DSP 180 Interface](#)' and '[HIL DSP Interface](#)'. Visible when *Access port* is set to *DIO*.
  - Controller index - visible when '[HIL TI uGrid Launchpad Interface](#)' is selected and *Access port* is set to *DIO*. Specifies which MCU slot on the interface board is used. It is dependent on *Interface type* property value and *Target platform* selected in *TI C2000 Setup* component
  - Timeout – time window in seconds after which the controller reports that expected messages are not received by raising a timeout flag and will re-initialize SCI receiving registers. If *Access port* is set to *USB* and SCADA widgets are used for transmitting messages to the controller, timeout should be equal or greater than 250 ms, since SCADA widgets operate with period of 250 ms.
  - Execution rate – desired rate at which the controller will read the received bytes. This value must be compatible with other components of the same subsystem: the value must be a multiple of the fastest execution rate in the circuit. To specify the execution rate, you can use either decimal (e.g. 0.001) or exponential values (e.g. 1e-3) in seconds. The aspect of limited baud rate and amount of data specified for exchange must be considered while determining execution rate. Typically, an execution rate of 1 ms should be sufficient.
- **Tab Data:**
    - Start of frame – bytes used for start of frame which is sent before the message payload and is used for synchronization,
    - End of frame - bytes used for end of frame which is sent after the message payload and is used for synchronization,
    - Message configuration - Defines the message payload for receiving operation. It specifies message configuration that controller expects. For more information on how to configure the payload check the [Message](#)

[Configuration](#) section (from the documentation for core serial communication components).

**NOTE:** Data escape option is not available in SCI Receive (Generic) component! Hence, if [Serial Send](#) component is used on transmitter side, this option must be disabled.

#### Component outputs:

- out1 – for each message defined in *Message configuration* property separate output is created and named after the message. Dimension is also defined separately for each output in *Message configuration*.
  - Supported types: uint, int and real.
  - Vector support: yes, if *Access port* is set to *DIO*, otherwise no.
- status – receiver status flags. It is a vector of 3 elements. Each element can have a value of 0 or 1. Value of 1 indicates that some of the following flags have been raised:
  - Status[0]: Timeout flag. If value of *Timeout* property is greater than 0 this flag will be raised after specified time frame if no bytes are received.
    - Supported types: uint,
    - Vector support: no.
  - Status[1]: Framing error bit. Value of FE bit from SCI RX status register is copied. A mismatch in *start-of-frame* and/or *end-of-frame* bytes will also raise the framing error flag if those properties are specified.
    - Supported types: uint,
    - Vector support: no.
  - Status[2]: Parity error bit. Value of PE bit from SCI RX status register is copied.
    - Supported types: uint,
    - Vector support: no.

**NOTE:** For utilizing serial communication between *HIL SCADA* and controller (through USB connection between PC and controller – if *Access port* is set to **USB**), a SCADA counterpart is available in form of [SCI Write widget](#). *SCI Write* widget writes messages defined by [SCI Receive \(Generic\)](#) component to COM port to which the controller is connected. Number of [SCI Write widgets](#) should match number of messages in *Message configuration* property. **Dimension of every message must be 1 in this case!** *Start of frame*, *End of frame* and *Message configuration* properties should be copied to [SCI Setup widget](#) properties. In this case, target development board **must be configured** to enable the serial communication through USB connection. Details on how to do this can be found [here](#).

**NOTE:** In context of serial communication between HIL device and controller (through digital pins – if *Access port* is set to **DIO**), HIL transmits messages to the controller, hence [Serial Send](#) component must be configured properly in the model (outside of the

exported subsystem) – it must have the same *Start of frame*, *End of frame* and *Message configuration* properties. Also, the same HIL DO number must be used by both sides (specified in *Transmitting HIL DO number* property of SCI Receive (Generic) component and [Initial Settings](#) component).

## Backward incompatibility resolution

With *TI C2000 Toolbox* version 1.0.0 *SCI Receive (Generic)* suffered major refactoring. Many new properties are introduced (see [Component properties](#) section) and as a result, models containing this component might have some backward incompatible changes. In case of *SCI Receive (Generic)* component, the following changes might affect old models:

- **Message configuration changed:** Before 1.0.0 version, all messages defined in *SCI Receive (Generic)* component had fixed configuration:
  - Data type: 32-bit (4-byte) float (real),
  - Dimension: 1,
  - Endianness: little endian
  - Additional *index* byte before EVERY message (variable) defined was expected by the controller (starting from 1).

For example, if message payload was defined with one signal (one output of the component), the MCU expected to receive bytes arranged like on Figure 2:

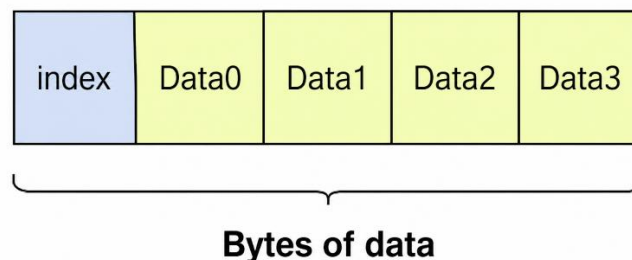


Figure 2. Old message configuration.

On new version, the *index* byte is removed, so data bytes of all signals are “merged” into one sequence, with optional *start-of-frame* and *end-of frame* bytes. Message payload can be customized additionally with several parameters (such as data type, dimension, endianness, etc.), as described [here](#).

Now, expected message payload can have different configurations, as described [here](#).