
PSIM Tutorial

**How to Use SCI for Real-Time Monitoring
in TI F28335 Target**

With the SimCoder Module and the TI F28335 Target, PSIM can generate ready-to-run codes for DSP boards that use TI floating-point DSP F28335. By using the Serial Communication Interface (SCI) blocks in the TI F28335 Target library, PSIM offers the capability to display waveforms and change parameters inside the DSP in real time. This makes it extremely easy to test, debug, and adjust DSP control code in a non-disruptive and non-intrusive way.

This tutorial describes the SCI blocks, SCI cable, and the setup procedure in the PSIM schematic and in hardware to achieve the real-time monitoring.

To illustrate the process, we will use the circuit “3-ph sine wave with SCI monitoring.psimsch” as an example. This example is located in the sub-folder “[examples\SimCoder\TI F28335 Target\3-ph sine wave with SCI](#)” in the PSIM directory. We will generate the code from this circuit and run it on the TI TMS320F28335 Experimenter Kit (TI Part Number TMDSDOCK28335).

To keep the original example unchanged, we will copy the whole folder to “[c:\TI SCI](#)”, and use this folder as the working folder in this tutorial.

1. SCI in TI F28335

The TI F28335 DSP has 3 SCI ports: SCIA, SCIB, and SCIC. Each SCI port can use different GPIO ports for communication, as listed below :

For SCIA:

- GPIO28 as the SCI receiving (Rx, or input) pin, and GPIO29 as the SCI transmitting (Tx, or output) pin.
- GPIO35 as the receiving pin, and GPIO36 as the transmitting pin.

For SCIB:

- GPIO9 as the receiving pin, and GPIO11 as the transmitting pin.
- GPIO14 as the receiving pin, and GPIO15 as the transmitting pin.
- GPIO18 as the receiving pin, and GPIO19 as the transmitting pin.
- GPIO22 as the receiving pin, and GPIO23 as the transmitting pin.

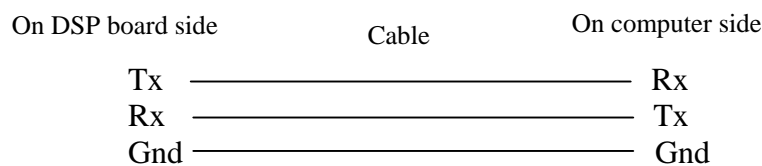
For SCIC:

- GPIO62 as the receiving pin, and GPIO63 as the transmitting pin.

PSIM will support only one of the above SCI port combinations at any time as RS-232.

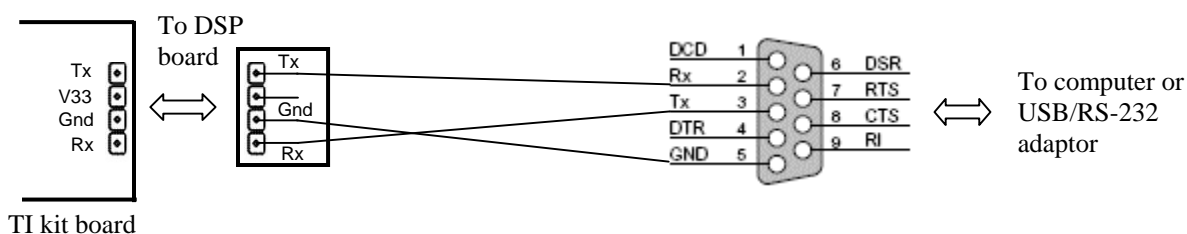
2. RS-232 Cable

To communicate with the computer, a RS-232 cable is needed to connect the DSP board to the computer. This cable needs 3 wires: One for receiving (Rx), one for transmitting (Tx), and another for ground (Gnd). The connections on each end of the cable are shown below:

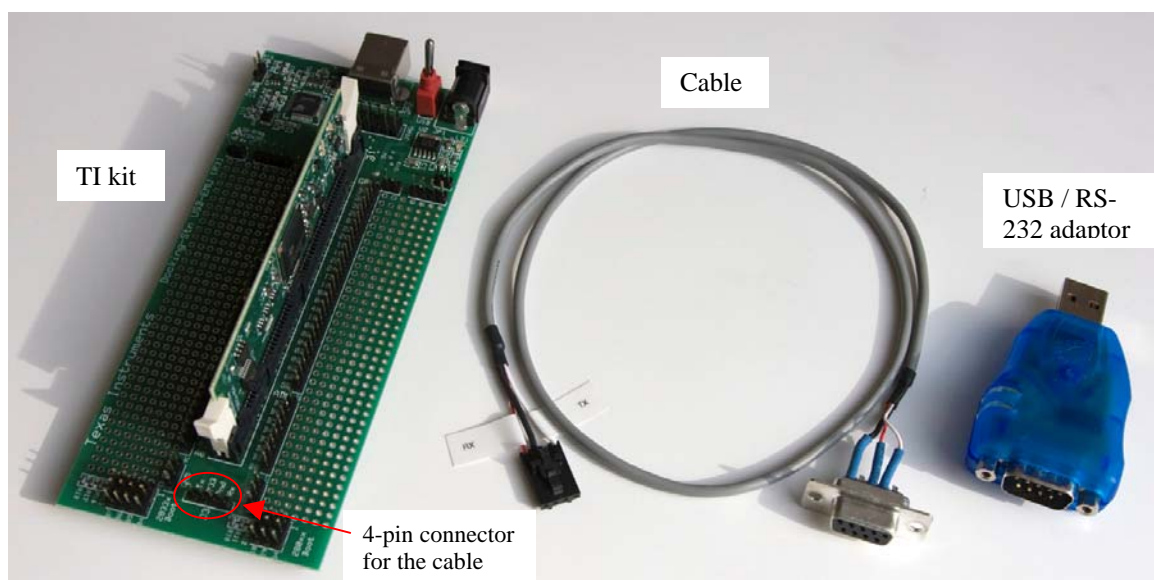


A 9-pin DB-9 connector (female) is used on the cable so that it can be connected to the computer serial port. If a computer does not have the serial port, an USB/RS-232 serial adapter with the DB-9 connector (male) needs to be used.

The figure below shows the connection diagram of the cable made for the TI evaluation kits. The DSP end consists of a 4-pin header connector, and the computer end uses a 9-pin DB-9 connector (female) to connect to the computer serial port. If a computer does not have a serial port, an USB/RS-232 serial adapter with the DB-9 connector (male) can be used.

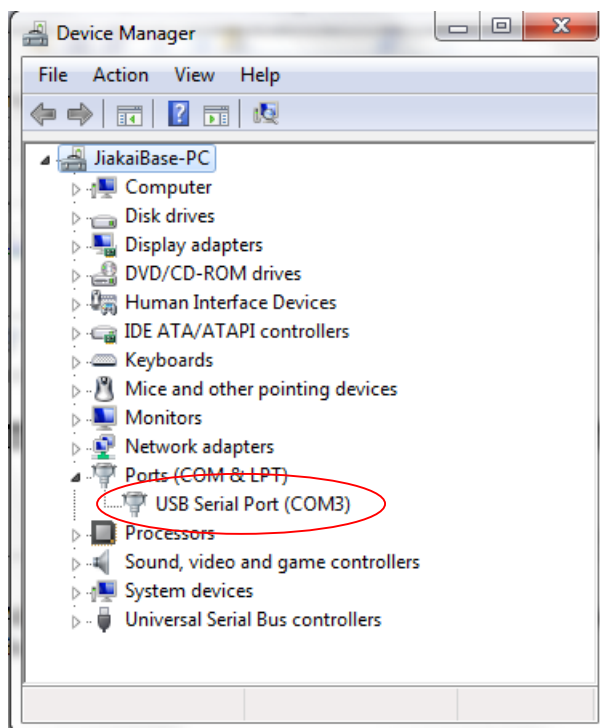


A picture of the TI Experimenter kit, the cable, and the USB/RS-232 serial adaptor is shown below. On the Experimenter kit, the 4-pin connector is labelled as "Tx V33 Gnd Rx".



Note that if the USB/RS-232 adaptor is used, one needs to first install the driver that comes with the adaptor before using it.

After the USB/RS-232 adaptor driver is installed, plug in the adaptor to the computer and open computer Device Manager from the Control Panel to find the RS-232 port number, as shown below. The figure shows that the RS-232 serial port is COM3.



If there are multiple serial devices connected to the computer, try to unplug the USB/RS-232 adaptor and plug it back in to find out the port number. Remember this port number as it is needed in PSIM when defining the communication with DSP.

3. Testing the Communication between DSP and Computer

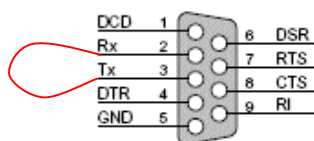
To test if the communication between DSP and the computer works, or to isolate and debug the problem if the communication does not work, perform the three tests below.

Test 1: DSP Only

In this test, a DSP program will send out the data through the SCI port. The RS-232 cable will be shorted so that the data will be looped back and the DSP will receive the same data that it sends out. If the test is successful, it means that the serial communication on the DSP side is working.

Follow the steps below to perform the test:

- Connect the TI Experimenter kit to the computer through the USB cable. Turn on the power on the Experimenter kit. Launch the CodeComposerStudio (CCS) setup, and make sure that "F28335 XDS100 USB Emulator" is selected. Exit the setup and launch CCS. Connect to the DSP board by selecting **Debug -> Connect**.
- Connect the 4-pin connector end of the RS-232 cable to the 4-pin connector (labelled as "Tx V33 Gnd Rx") on the Experimenter kit board.
- Use a wire to short the Rx pin (pin 2) and Tx pin (pin 3) of the DB-9 connector, as shown here:



- In CCS, select **File -> Load Program**, and go to the PSIM folder under "SimCoder\DSP RxTx Test", and select the file "DSP_RxTx_Test.out,out" to load the code to the DSP RAM memory. For the dialog message "Code Composer could not locate DSP2833x_CodeStartBranch.asm. Would you like to browse for it?", choose No.
- In CCS, select **Debug -> Reset CPU** to reset the DSP. Then select **Debug -> Restart** to restart the DSP, and select **Debug -> Go Main** to go to the entry point of the main function. On the dialog message "Code Composer could not locate: DoutSci.c. Would you like to browse for it?", choose No. Then select **Debug -> Run** to run the code.
- If the communication works, the two red LED lights LD2 and LD3 on the F28335 controlCard should be on. The location of these two LED's are shown below (circled in red):



Test 2: Computer Only

In this test, a program running on the computer will send out the data through the serial port. The RS-232 cable will be shorted so that the data will be looped back and the computer will receive the same data that it sends out. If the test is successful, it means that the serial communication on the computer side is working.

Follow the steps below to perform the test:

- Connect the USB adapter to the computer, and the DB-9 end of the RS-232 cable to the USP adapter.
- Use a wire to short the Rx pin and Tx pin of the 4-pin connector of the RS-232 cable.
- In CCS, select **File -> Load Program**, and go to the PSIM folder under "SimCoder\DSP RxTx Test", and select the file "DSP_RxTx_Test.out,out" to load the code to the DSP RAM memory.
- Launch PSIM. Select **Utilities -> DSP Oscilloscope**. Set the correct serial port number. Set the baud rate as 115200, and parity as None.
- Click on the **Test** button. If the communication works, the dialog will show "The test is successful".

Test 3: DSP and Computer

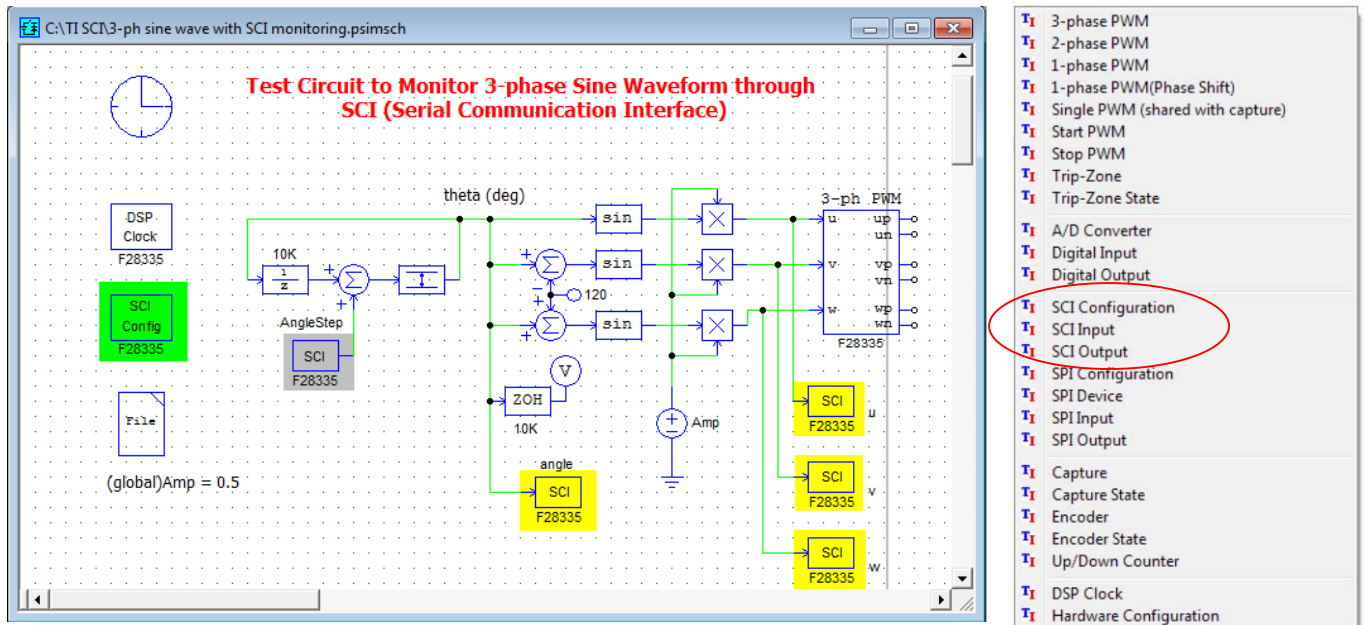
In this test, both the DSP test program and the computer test program will run at the same time. Data sent by DSP will be received by the computer, and data sent by the computer will be received by the DSP. If the test is successful, it means that the serial communication between the DSP and the computer is working.

Follow the steps below to perform the test:

- Set up in the same way as in Test 1. But instead of shorting the Rx and Tx pins of the DB-9 connector, connect the DB-9 connector to the USB adapter, and connect the USB adapter to the computer.
- Run the test program in Test 1.
- Run the test program in Test 2.
- On the DSP Oscilloscope, click on the **Test** button. If the communication works, the two red LED lights on the DSP controlCard will blink for around 3 sec. before they are off, and the computer program will report "The test is successful".

4. Setting up in PSIM

In PSIM, load the schematic file "3-ph sine wave with SCI monitoring.psimsch" in the folder "examples\TI F28335 Target\3-ph sine wave with SCI", as shown on the left below. The TI F28335 Target library menu is shown on the right.



In this circuit, there are 3 types of SCI library elements: *SCI Configuration* (highlighted in green), *SCI Input* (highlighted in grey), and *SCI Output* (highlighted in yellow), as explained in the sections below. These elements can be accessed by going to **Elements -> SimCoder -> TI F28335 Target**.

4.1 SCI Configuration

The SCI Configuration element has four parameters:

- *SCI Port*: Defines the SCI port from the following list:
 - SCIA (GPIO28, GPIO29)
 - SCIA (GPIO35, GPIO36)
 - SCIB (GPIO9, GPIO11)
 - SCIB (GPIO14, GPIO15)
 - SCIB (GPIO18, GPIO19)
 - SCIB (GPIO22, GPIO23)

- SCIC (GPIO62, GPIO63)
- *Speed (bps)*: Define the SCI communication speed, in bps (bits per second). A list of preset speeds is provided at 200000, 115200, 57600, 38400, 19200, or 9600 bps. Or one can specify any other speed manually.
- *Parity Check*: Define the parity check setting for error check in communication. It can be *None*, *Odd*, or *Even*.
- *Output Buffer Size*: Define the size of the data buffer allocated in DSP for SCI. The buffer is located in the RAM area, and each buffer element stores one data point which consists of three 16-bit words (that is, 6 bytes, or 48 bits, per data point).

4.2 SCI Input

To adjust a parameter inside the DSP at run time, one can use a SCI Input element in the schematic. The SCI Input element behaves as a signal source, and can be used in both the main schematic and subcircuits. An initial value can be set for the SCI Input element, and this value will be used until it is changed. Users can change the parameter on the computer. The change will be sent to the DSP through the SCI communication, and the parameter will be updated on the DSP at run time.

In a schematic, up to 127 SCI Input elements can be used.

4.3 SCI Output

To monitor and display the waveform of a signal inside the DSP at run time, one can place a SCI Output element on the schematic. The SCI Output element behaves as an output probe, and can be used in both the main schematic and subcircuits other than event subcircuits and interrupt subcircuits. The waveform of a SCI Output element will be incorrect if the element is inside an event or interrupt subcircuit.

In a schematic, up to 127 SCI Output elements can be used.

The SCI Output element has one parameter: Data Point Step. It defines how frequent the data is sent out. If the Data Point Step is set to 5, for example, only 1 points out of every available 5 points will be sent from DSP to the computer.

To calculate the time that it takes to send one data point, a data point has 6 bytes, and the command and other information take 2 bytes, making it 8 bytes in total for one data point. When sending out one byte of data, one start bit and one stop bit, and possibly one parity check bit, need to be added.

With the parity check, the total bit length for one data byte will be 11 (1 start bit, 8 data bits, 1 parity check bit, and 1 stop bit), and without the parity check, the total bit length for one data byte will be 10 (1 start bit, 8 data bits, and 1 stop bit).

The time that it takes to send one data point can be calculated as follows:

$$\text{Total Bit Length} * 8 / \text{SCI Speed}$$

For example, if the SCI communication speed is 115200 bps, and no parity check is used, sending out one data point will take $10 * 8 / 115200$, or 0.000694 sec. In another word, DSP can send out up to $1/0.000694 = 1440$ data points in one second.

5. Generating and Running Code on DSP

One can simulate the circuit by selecting **Simulate -> Run Simulation**. In the simulation, the values of SCI Input elements will be fixed at their initial values.

To generate the code, and upload and run it on the DSP, follow the steps below:

- Generate code by selecting **Simulate -> Generate code**.
- Connect the DSP board to the computer with the USB cable. Note that in this case, the TI Experimenter Kit has an onboard XDS100 USB emulator, and can be connected to the computer directly without another JTAG emulator in between. If another DSP board is used (for example TI's Digital Power Experimenter's Kit, part number TMSDCDC2KIT), an external JTAG emulator will be needed.
Also, make sure that the USB/RS-232 adaptor is disconnected from the computer. For some reason, if the computer is connected to the DSP board via the RS-232 cable, the CodeComposerStudio cannot establish the connection with the DSP.
- Start CodeComposerStudio (CCS) v3.3. Go to **File -> Launch Setup**. In the CCS Setup window, under **System Configuration**, make sure that "F28335 XDS100 USB Emulator" is listed. If not, on the right, choose the **Fatory Boards** tab, choose **C28xx** under the **Family** pull-down menu, and find and highlight **F28335 XDS100 USB Emulator**, then click on the **Add** button. Click on **Save & Quit** to exit the CCS Setup. If changes are made in the setup, restart CCS.
- In CCS, go to **Project -> Open**, and navigate to the folder "c:\TI SCI\ 3-ph sine wave with SCI monitoring (C code)", and open the file "3_ph_sine_wave_with_SCI_monitoring.pjt".
- Select **Debug -> Connect** to connect to the DSP board. If the connection is successful, a message "The target is now connected" will pop up at the lower left corner of CCS.
- Go to **Project -> Rebuild All** to compile the code. The default configuration of the project is RAM Debug.
- Select **File -> Load Program**, and go to the RamDebug folder and select the file "3_ph_sine_wave_with_SCI_monitoring.out" to load the code to the DSP RAM memory.
- Select **Debug -> Reset CPU** to reset the DSP. Then select **Debug -> Restart** to restart the DSP, and select **Debug -> Go Main** to go to the entry point of the main function.
- Select **Debug -> Run** to run the code.

6. Monitoring Waveforms and Adjusting Parameters

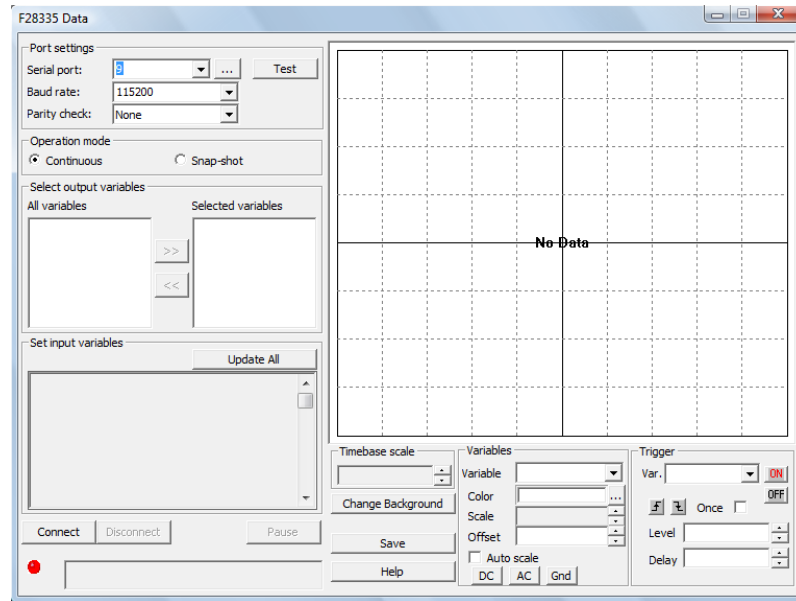
As the code is now running in the DSP, we can monitor waveforms and change parameters following the steps below. Note that the DSP will not send out any data until it is requested.

- Connect the RS-232 cable to the DSP board on one end, and to the computer on the other end. If the computer does not have the serial port, use an USB/RS-232 adapter.
- In PSIM, select **Utilities -> DSP Oscilloscope**. The dialog window is shown below.

Under **Port settings**, set the serial port number to be the same as one found in Step 2. For example, if the serial port is COM3, set the serial port number to 3.

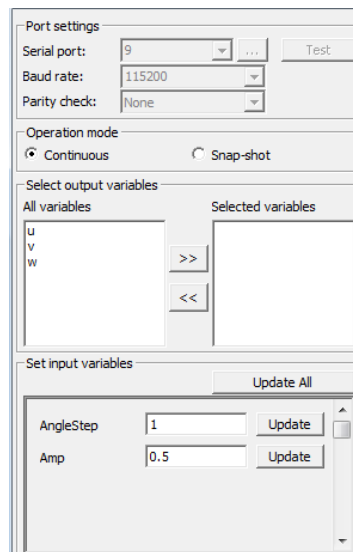
Also, set the baud rate and the parity check to be the same as defined in the **SCI Configuration** element in PSIM. In this example, the baud rate is set to 115200bps, and the parity check is set to None.

Under **Operation mode**, choose *Continuous*. In the *Continuous* operation mode, data are sent from DSP to the computer continuously. In the case where the communication speed is not fast enough to send out all the data, some data will be lost and the Data Integrity (shown at the lower left corner) will be less than 100%.



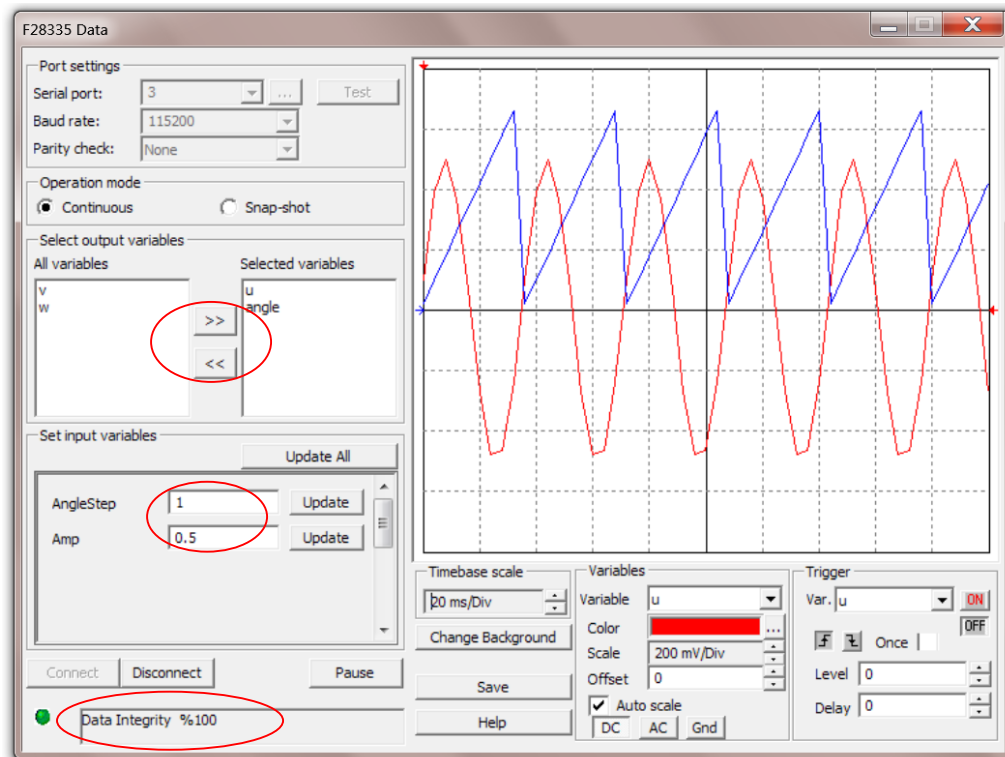
On the other hand, in the *Snap-shot* operation mode, every data point in the DSP buffer will be sent out, and the complete data set will be displayed on the DSP Oscilloscope. After all the data in the buffer are sent out, the buffer will be refilled. In this mode, the data is not continuous, but a snap-shot is captured every time.

- Click on the **Connect** button in the lower left corner to connect the computer to the DSP. If connection is successful, all output and input variables will be displayed in the left side of the window, as shown in the figure below.



If connection cannot be established, please check the following:

- Make sure that the serial port number is correct.
- Make sure that the communication speed and parity check settings in the DSP Oscilloscope are the same as in the **SCI Configuration** element in PSIM.
- Make sure that the RS-232 cable is connected properly on both ends.
- If an USB/RS-232 adaptor is used, make sure that necessary driver is installed properly as instructed.
- Make sure that the DSP external clock on the DSP hardware is same as the setting in the **DSP Clock** element in PSIM.
- Make sure that the DSP code is running.
- If the problem still persists, please run the tests as described in Section 3.
- To display waveforms of output variables defined by the **SCI Output** elements, select the variables from the **All Variables** column, and click on the >> button to move them to the **Selected variables** column. The waveforms of the selected variables will be displayed in the scope, as shown below.



To remove a variable from display, highlight it in the **Selected variables** column, and click on the << button.

- Input variables are the parameters that can be changed in the DSP code. Input variables are defined either by the SCI Input elements (such as the SCI Input element *AngleStep* in this example), or if a parameter is defined as a global variable (such as the variable *Amp* in this example).

To change an input variable, enter the new value in the edit box, and click on **Update**. The new value will be sent to DSP. For example, when the input variable AngleStep is changed from 1 to 0.5, the frequency of the waveforms will be reduced by half.

- The lower left corner of the oscilloscope also shows the **Data Integrity**. If the data integrity is 100%, it means that all the data collected have been successfully transmitted and displayed.

If the data integrity is less than 100%, it means that some data points are lost due to the fact that the requested data amount is too large and the DSP does not have enough resource to send it. If this occurs, show a fewer number of waveforms, or increase the Data Point Step value in the SCI Output elements, so that fewer number of data points need to be transmitted and displayed. Alternatively, one can change the Operation Mode from *Continuous* to *Snap-shot*.

- Click on the **Pause** button to halt the data acquisition, or the **Disconnect** button to disconnect the computer from the DSP.

7. Deciding the SCI Communication Buffer Size

There are two operation modes in DSP Oscilloscope: continuous mode and snap-shot mode. In the continuous mode, the buffer size does not matter as the data integrity is determined by the communication speed. However, in the snap-shot mode, the buffer size determines the amount of data displayed in the DSP Oscilloscope. Therefore, the buffer size should be selected based on the consideration of the snap-shot mode.

In the snap-shot mode, data are saved to the DSP buffer, and are sent to the computer at the same time. New data will continue to be save to available buffer cells until no more free cells are available. This will happen if the rate of receiving data is faster than the rate of sending data. At this point, DSP will stop collecting the data, and it will resume data collection after all the data in the buffer are sent out.

In the snap-shot mode, since the DSP Oscilloscope will display only the complete set of data in the buffer at a time, if the buffer size is too small, the displayed waveform will be very short. On the other hand, the buffer size cannot be too large as it is limited by the physical RAM memory available in the DSP.

To calculate the proper buffer size, assume that there is only one variable to display. Let N_{step} be the Data Point Step of the SCI output block, f_s be the sampling frequency of the variables, $N_{display}$ be the number of data points one intends to display, N_{sci} be number of data points that SCI sends out in one second, and N_{buffer} be the buffer size, we have:

$$N_{buffer} = N_{display} \cdot \left(1 - \frac{N_{step} \cdot N_{sci}}{f_s}\right) + 1$$

For example, if the baud rate is 115200 bps, based on the calculation in Section 4.3, the number of points that SCI can send out in one second is $N_{sci} = 1440$. Also, if the Data Point Step N_{step} is 5, the sampling rate f_s is 10 kHz, and the number of points for display $N_{display}$ is 1000, we can calculated the required buffer size as:

$$N_{buffer} = 1000 \cdot \left(1 - \frac{5 \cdot 1440}{10000}\right) + 1 = 281$$

If the size of the buffer is too big to locate in the RAM memory, CCS will report a link error as below:

```
"C:\TI SCI\3-ph sine wave with SCI monitoring (C code)\F28335_RAM_Lnk.cmd", line 47:  
error: run placement fails for object ".ebss"
```

When CCS reports a link error like this, try to reduce the buffer size, or compile and run the generated code in the "Flash Release" mode.