PSIM

TUTORIAL

Auto Code Generation for F2806X Target

August 2017

PSIM's SimCoder Module, combined with the F2806x Hardware Target, can generate ready-to-run code from a PSIM control schematic for hardware based on TI F2806x series fixed-point DSP.

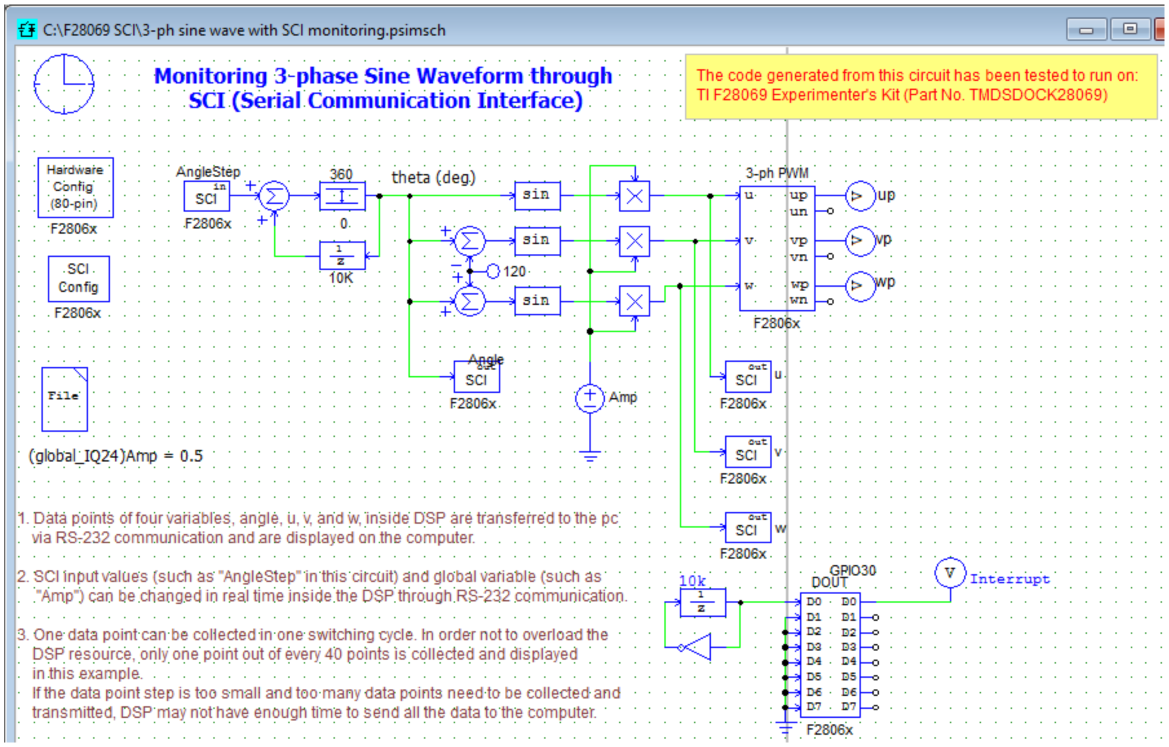
This tutorial describes, in step by step, how to generate code in PSIM, compile and upload the code in Code Composer Studio (CCS 6.1), and run it on F2806x DSP hardware.

To illustrate the process, we use the circuit "*3-ph sine wave with SCI monitoring.psimsch*" as an example. This example is located in the sub-folder "Examples\SimCoder\F2806x Target\3-ph sine wave with SCI" in the PSIM directory.

To keep the original example unchanged, we will copy the folder to "C:\F28069 SCI", and use this folder as the working folder in this tutorial.

1. PSIM Setup for Code Generation

In PSIM, load the schematic file "*3-ph sine wave with SCI monitoring.psimsch*" as shown below.


C:\F28069 SCI\3-ph sine wave with SCI monitoring.psimsch

Monitoring 3-phase Sine Waveform through SCI (Serial Communication Interface)

The code generated from this circuit has been tested to run on: TI F28069 Experimenter's Kit (Part No. TMDSDOCK28069)

(global_IQ24)Amp = 0.5

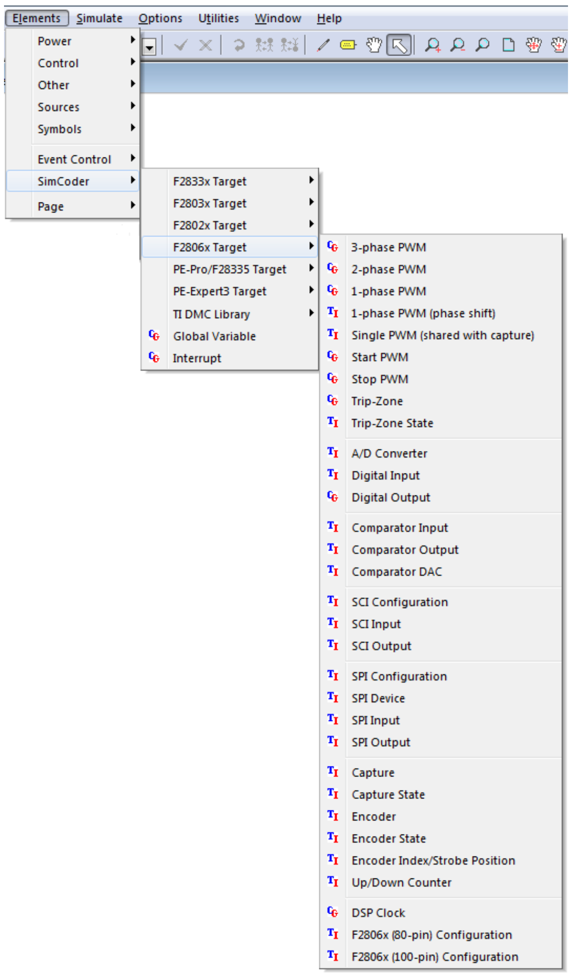
1. Data points of four variables, angle, u, v, and w, inside DSP are transferred to the pc via RS-232 communication and are displayed on the computer.

2. SCI input values (such as "AngleStep" in this circuit) and global variable (such as "Amp") can be changed in real time inside the DSP through RS-232 communication.

3. One data point can be collected in one switching cycle. In order not to overload the DSP resource, only one point out of every 40 points is collected and displayed in this example.
If the data point step is too small and too many data points need to be collected and transmitted, DSP may not have enough time to send all the data to the computer.

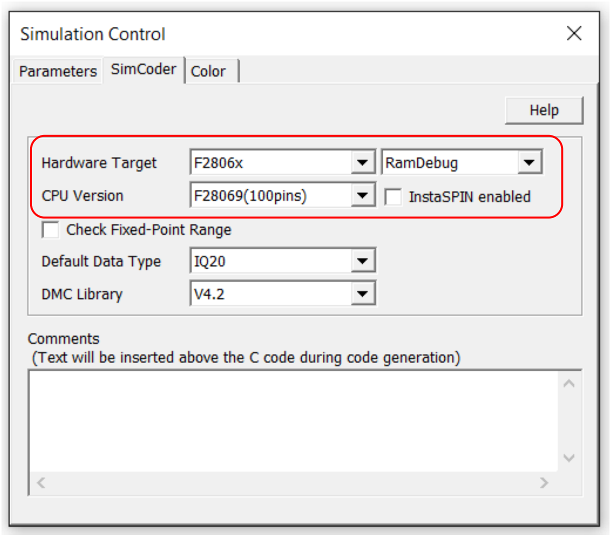
This example generates the code for the F28069 Piccolo Experimenter Kit (TMDSDOCK28069). To simulate this circuit, select **Simulate** >> **Run Simulation**.

For fixed-point code generation, a circuit must contain elements from the F2806x Target library. The library is located at **Elements** >> **SimCoder** >> **F2806x Target**. A list of the library is shown below.



Simulation Control Parameters

Before the code is generated, SimCoder parameters must be properly set in the Simulation Control dialog. Double click on the Simulation Control block (the clock image). Click on the SimCoder tab, and set the parameters as shown below.



Hardware Target: F2806x, RamDebug
CPU Version: F28069(100pins)
InstaSPIN enabled: ☐
Check Fixed-Point Range: ☐
Default Data Type: IQ20
DMC Library: V4.2
Comments
(Text will be inserted above the C code during code generation)

The settings are:

Hardware Target:	Set to *F2806x* and *RamDebug*. RamDebug is selected in this example for debugging.
CPU Version:	Select F28069 (100pin).
InstaSPIN enabled:	Check this box if an InstaSPIN enabled DSP is used, such as F28069M or F28069F. Leave this box unchecked a DSP without InstaSPIN. In this example, since the F28069 controlCARD is used which does not include InstaSPIN, the box will be unchecked.
Check Fixed-Point Range:	Check this box to check the fixed-point range for overflow.
Default Data Type:	Select IQ20 for this example.

Note that checking the fixed-point range for overflow requires additional computation. It is recommended that one perform this check after the circuit is close to be finalized.

Also, if an InstaSPIN enabled DSP is used, the checkbox *InstaSPIN enabled* must be checked. Otherwise the code will not run properly.

Currently, PSIM can generate project files for CCS v3.3 only. When using a CCS version newer than v3.3, one needs to use the function "Import Legacy v3.3 Project".

Defining F2806x Target Elements

In this circuit, several F2806x Target library elements are used. All these elements must be properly configured according to the specific target hardware settings.

F2806x Hardware Configuration: This element defines the I/O ports of the F2806x hardware. One must unlock this element to change the designation of the I/O port, and must lock it afterwards. In this example:

- GPIO1 and GPIO2 are defined as PWM outputs;
- GPIO28 and GPIO29 are defined for SCI communication; and
- GPIO30 is used as a digital output.

DSP Clock: There is no DSP Clock element on this schematic, which means the default clock setting as below will be used in this project.

- *DSP Clock Source*: Internal oscillator 1
- External Clock (MHz): 10
- *DSP Speed* (*MHz*): 90

SCI Configuration: Specify the I/O pins, speed, and parity check of the SCI port. The output buffer size is determined by the desired waveform resolution and available memory space. SCI communication provides a convenient way to utilize PSIM's DSP Oscilloscope function. The DSP Oscilloscope is used to control the buck converter's output voltage and monitor the values inside the DSP. The SCI input allows the reference value to be changed by the DSP Oscilloscope in real time, and SCI outputs are used to display signals it the DSP Oscilloscope in real time.

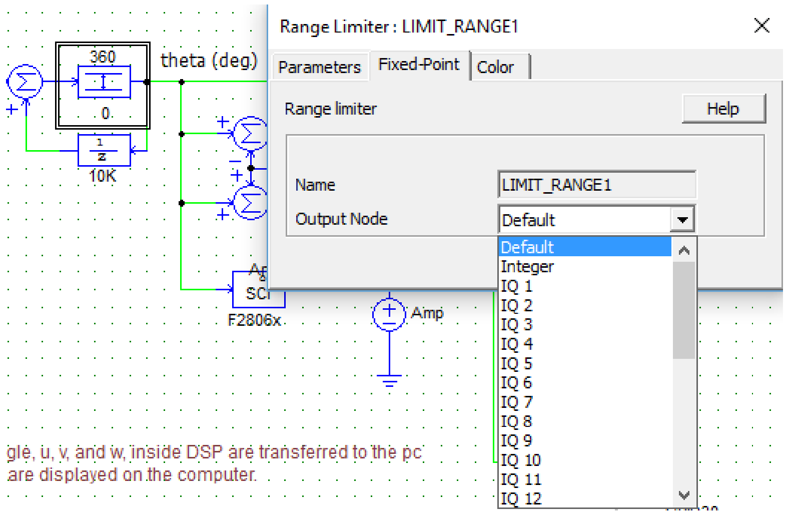
Digital Output (16 channel): Specify the corresponding GPIO port for output. In this example, GPIO30 on D0 is chose to indicate if the PWM interrupt executing is on time.

PWM Module (single phase): Specify the PWM module source, output mode, switching frequency, and many other settings of a single-phase PWM module. For further detailed information, please refer to the SimCoder User Manual.

Defining Fixed-Point Positions

In the F2806x Target code, TI's IQmath library is used for fixed-point calculation. As shown in the figure below, each of the non-hardware-related elements can be set as *Default*, *Integer*, or fixed point with the IQ number from 1 to 30.

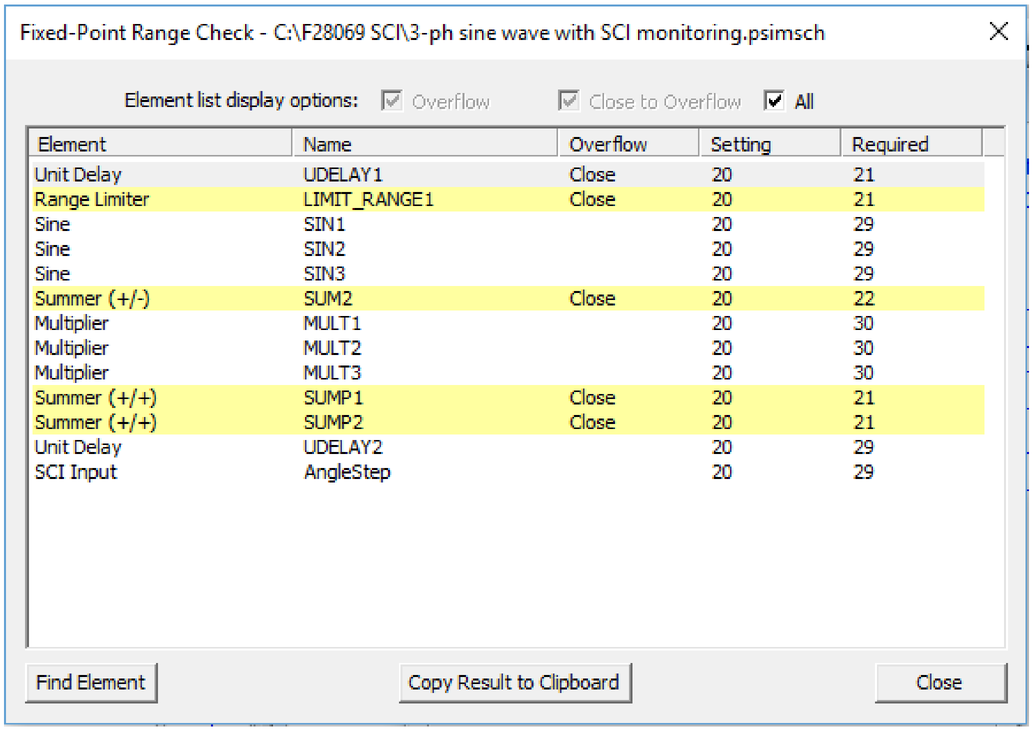
In this example, the *Default* fixed-point position is selected for the range limiter block LIMIT_RANGE1, which means that the output of this block will use the fixed-point position of IQ20 as defined in *DefaultFixed-Point Position* in Simulation Control.



Name LIMIT_RANGE1
Output Node Default

In fixed-point calculation, the selection of the fixed-point position is important. If the fixed-point range is too narrow, the variable value may overflow. But if the fixed-point range is too wide, the calculation resolution will suffer. To address this problem, PSIM provides the function to check the fixed-point range from simulation results, and give indication if a particular output is close to overflow or already overflow, or is too conservative in the fixed-point position setting.

In this example, after the simulation is completed, if the box *Check Fixed-Point Range* under **Simulation Control** >> **SimCoder** is checked, the following dialog window will appear.



Fixed-Point Range Check - C:\F28069 SCI\3-ph sine wave with SCI monitoring.psimsch

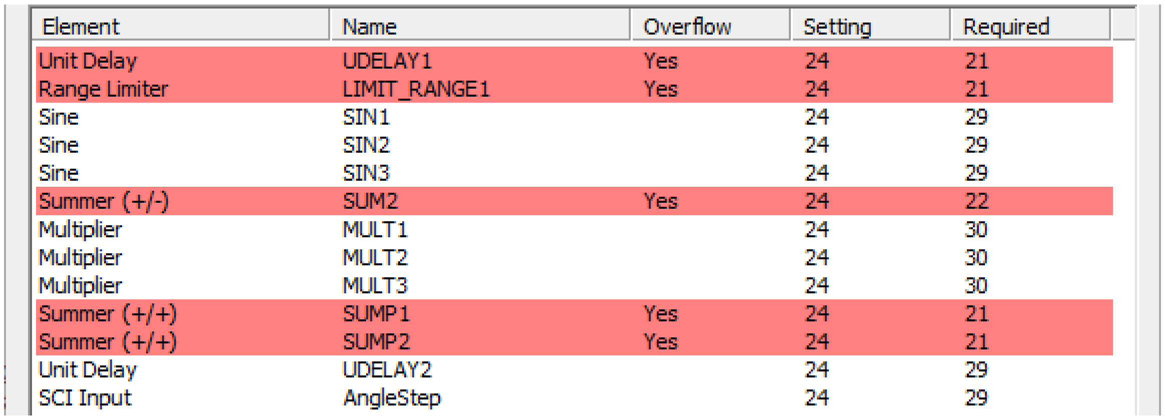
Element list display options: ☑ Overflow ☑ Close to Overflow ☑ All

Element	Name	Overflow	Setting	Required
Unit Delay	UDELAY1	Close	20	21
Range Limiter	LIMIT_RANGE1	Close	20	21
Sine	SIN1		20	29
Sine	SIN2		20	29
Sine	SIN3		20	29
Summer (+/-)	SUM2	Close	20	22
Multiplier	MULT1		20	30
Multiplier	MULT2		20	30
Multiplier	MULT3		20	30
Summer (+/+)	SUMP1	Close	20	21
Summer (+/+)	SUMP2	Close	20	21
Unit Delay	UDELAY2		20	29
SCI Input	AngleStep		20	29

Find Element Copy Result to Clipboard Close

The dialog window shows that, for example, the output of the range limiter LIMIT_RANGE1 is set to IQ20, and the required setting is IQ21, which is close to overflow. These variables are shown yellow in the window and need to decide if a wider range needed.

However, as an example, if the default fixed-position is set to IQ24, it will not be sufficient as some of the variables will overflow. The fixed-point range check results are shown below:



Element	Name	Overflow	Setting	Required
Unit Delay	UDELAY1	Yes	24	21
Range Limiter	LIMIT_RANGE1	Yes	24	21
Sine	SIN1		24	29
Sine	SIN2		24	29
Sine	SIN3		24	29
Summer (+/-)	SUM2	Yes	24	22
Multiplier	MULT1		24	30
Multiplier	MULT2		24	30
Multiplier	MULT3		24	30
Summer (+/+)	SUMP1	Yes	24	21
Summer (+/+)	SUMP2	Yes	24	21
Unit Delay	UDELAY2		24	29
SCI Input	AngleStep		24	29

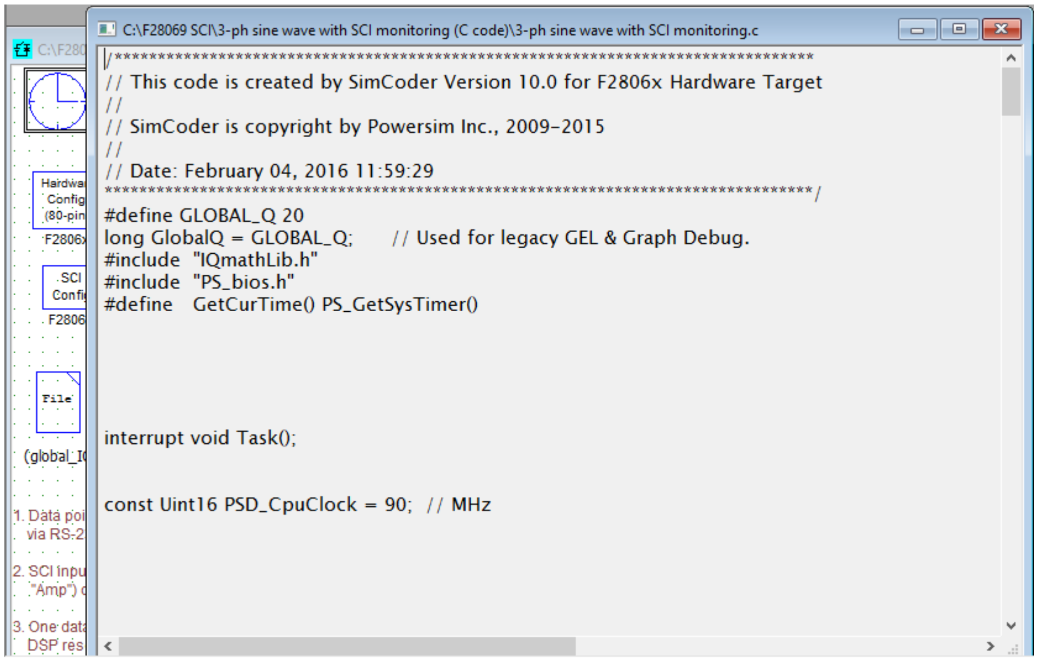
The results show that the outputs of the following elements will overflow:

UDELAY1, LIMIT_RANGE1, SUM2, SUMP1 and SUMP2

The range check function provides a convenient way to handle fixed-point positions.

Generating Code

To generate code, select **Simulate** >> **Generate Code**. The generated code will be displayed in a separate window, as shown below.


C:\F28069 SCI\3-ph sine wave with SCI monitoring (C code)\3-ph sine wave with SCI monitoring.c

```
/*****************************************************************************
// This code is created by SimCoder Version 10.0 for F2806x Hardware Target
//
// SimCoder is copyright by Powersim Inc., 2009-2015
//
// Date: February 04, 2016 11:59:29
*****************************************************************************/
#define GLOBAL_Q 20
long GlobalQ = GLOBAL_Q;     // Used for legacy GEL & Graph Debug.
#include "IQmathLib.h"
#include "PS_bios.h"
#define  GetCurTime() PS_GetSysTimer()



interrupt void Task();


const Uint16 PSD_CpuClock = 90;  // MHz
```

PSIM will create a subfolder inside the folder containing the PSIM circuit file and is named the same as the PSIM circuit but with (C code) added at the end. For this example project, this subfolder is C:\F28069 SCI\3-ph sine wave with SCI monitoring (C code).

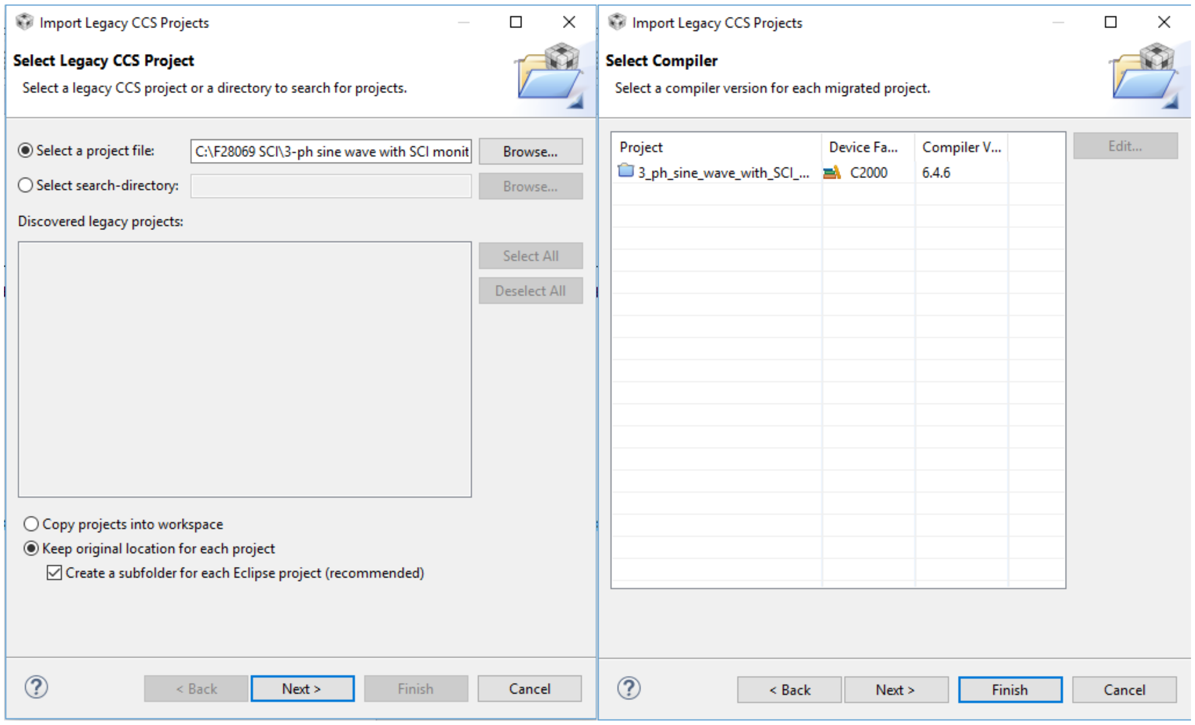
In addition to the C code file, PSIM also generates project files, link command files, as well as all other files necessary for four different project building configurations: RAM Debug, RAM Release, Flash Release, and Flash RAM Release. All these files are stored in the newly created subfolder C:\F28069 SCI\3-ph sine wave with SCI monitoring (C code).

2. CCS Setup

The generated project is for CCS v3.3. If you are using CCS v3.3, the project can be opened directly. For CCS 4 or higher, you must use the *Import Legacy v3.3 Project* function in CCS. Below CCS v6.1 is used to show how to load and debug the program.

Import Project into CCS

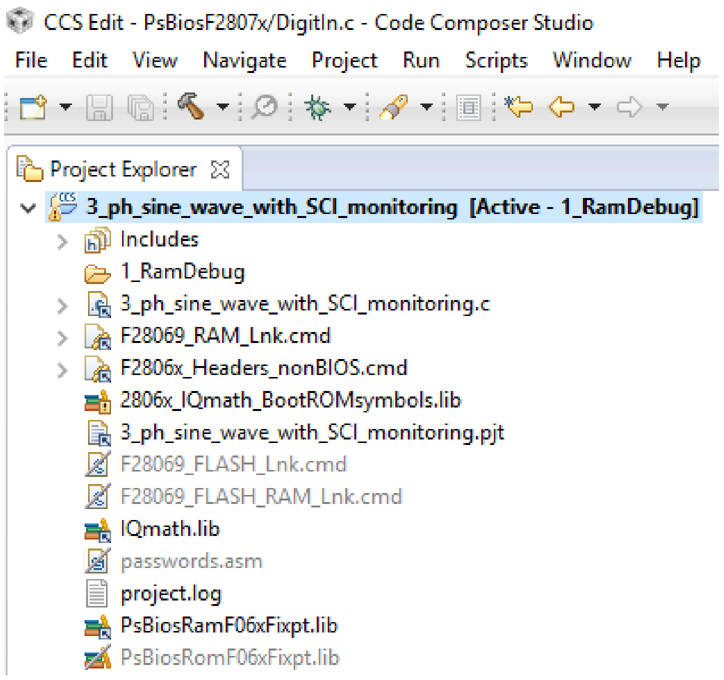
Launch CCS v6.1. If this is the first time you load the project, go to **Project** >> **Import Legacy CCSv3.3 Project**. The dialog is shown below on the left. Click on **Browse...** to choose "C:\F28069 SCI\3-ph sine wave with SCI monitoring (C code)\3_ph_sine_wave_with_SCI_monitoring.pjt", then click on **Next**.The dialog will be shown as below on the right. Click on **Finish** to start importing.


Import Legacy CCS Projects
Select Legacy CCS Project
Select a legacy CCS project or a directory to search for projects.
Browse...
Select All
Deselect All
< Back
Next >
Finish
Cancel
Import Legacy CCS Projects
Select Compiler
Select a compiler version for each migrated project.
Edit...
< Back
Next >
Finish
Cancel

(●) Select a project file: C:\F28069 SCI\3-ph sine wave with SCI monit
( ) Select search-directory:
Discovered legacy projects:
( ) Copy projects into workspace
(●) Keep original location for each project
☑ Create a subfolder for each Eclipse project (recommended)

Project	Device Fa...	Compiler V...
3_ph_sine_wave_with_SCI_...	C2000	6.4.6

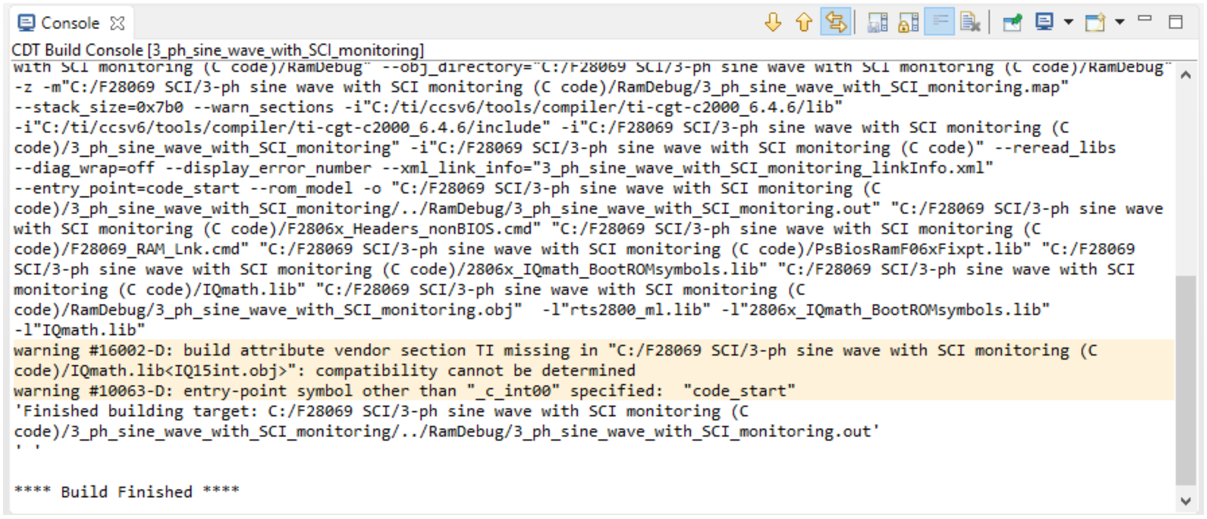
If there is no problem in importing, the converted project will be automatically loaded. Project Explorer of the CCS will appear as shown below.


CCS Edit - PsBiosF2807x/DigitIn.c - Code Composer Studio
File Edit View Navigate Project Run Scripts Window Help
Project Explorer

3_ph_sine_wave_with_SCI_monitoring [Active - 1_RamDebug]
- Includes
- 1_RamDebug
- 3_ph_sine_wave_with_SCI_monitoring.c
- F28069_RAM_Lnk.cmd
- F2806x_Headers_nonBIOS.cmd
- 2806x_IQmath_BootROMsymbols.lib
- 3_ph_sine_wave_with_SCI_monitoring.pjt
- F28069_FLASH_Lnk.cmd
- F28069_FLASH_RAM_Lnk.cmd
- IQmath.lib
- passwords.asm
- project.log
- PsBiosRamF06xFixpt.lib
- PsBiosRomF06xFixpt.lib

Note that the project configuration is set to *RAM Debug*. With this setting, all program and data will be loaded to the RAM memory.

To compile the project, right click on the project name "3_ph_sine_wave_with_SCI_monitoring" in the panel of the Project Explorer. Then click on **Build Project** in the popup menu. Or click on the project name in the panel of Project Explorer to select it as the current project (the project name changes to bold). Select **Project** >> **Build** to build the project or **Project** >> **Rebuild All** to rebuild the whole project. After the compiling is complete, CCS will display the following:


Console

CDT Build Console [3_ph_sine_wave_with_SCI_monitoring]

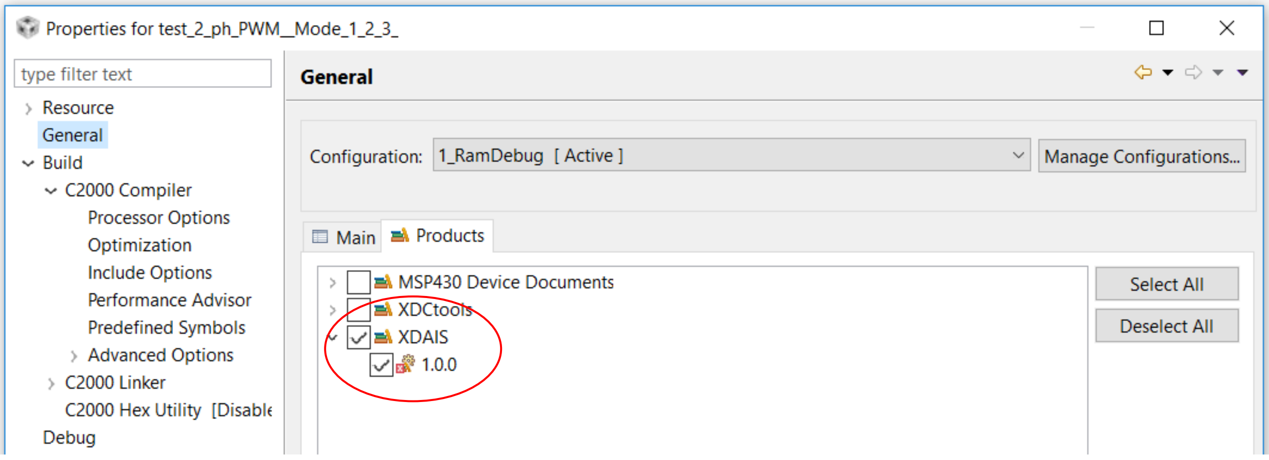
```
with SCI monitoring (C code)/RamDebug" --obj_directory="C:/F28069 SCI/3-ph sine wave with SCI monitoring (C code)/RamDebug"
-z -m"C:/F28069 SCI/3-ph sine wave with SCI monitoring (C code)/RamDebug/3_ph_sine_wave_with_SCI_monitoring.map"
--stack_size=0x7b0 --warn_sections -i"C:/ti/ccsv6/tools/compiler/ti-cgt-c2000_6.4.6/lib"
-i"C:/ti/ccsv6/tools/compiler/ti-cgt-c2000_6.4.6/include" -i"C:/F28069 SCI/3-ph sine wave with SCI monitoring (C
code)/3_ph_sine_wave_with_SCI_monitoring" -i"C:/F28069 SCI/3-ph sine wave with SCI monitoring (C code)" --reread_libs
--diag_wrap=off --display_error_number --xml_link_info="3_ph_sine_wave_with_SCI_monitoring_linkInfo.xml"
--entry_point=code_start --rom_model -o "C:/F28069 SCI/3-ph sine wave with SCI monitoring (C
code)/3_ph_sine_wave_with_SCI_monitoring/../RamDebug/3_ph_sine_wave_with_SCI_monitoring.out" "C:/F28069 SCI/3-ph sine wave
with SCI monitoring (C code)/F2806x_Headers_nonBIOS.cmd" "C:/F28069 SCI/3-ph sine wave with SCI monitoring (C
code)/F28069_RAM_Lnk.cmd" "C:/F28069 SCI/3-ph sine wave with SCI monitoring (C code)/PsBiosRamF06xFixpt.lib" "C:/F28069
SCI/3-ph sine wave with SCI monitoring (C code)/2806x_IQmath_BootROMsymbols.lib" "C:/F28069 SCI/3-ph sine wave with SCI
monitoring (C code)/IQmath.lib" "C:/F28069 SCI/3-ph sine wave with SCI monitoring (C
code)/RamDebug/3_ph_sine_wave_with_SCI_monitoring.obj"  -l"rts2800_ml.lib" -l"2806x_IQmath_BootROMsymbols.lib"
-l"IQmath.lib"
warning #16002-D: build attribute vendor section TI missing in "C:/F28069 SCI/3-ph sine wave with SCI monitoring (C
code)/IQmath.lib<IQ15int.obj>": compatibility cannot be determined
warning #10063-D: entry-point symbol other than "_c_int00" specified:  "code_start"
'Finished building target: C:/F28069 SCI/3-ph sine wave with SCI monitoring (C
code)/3_ph_sine_wave_with_SCI_monitoring/../RamDebug/3_ph_sine_wave_with_SCI_monitoring.out'
' '

**** Build Finished ****
```

The warning message can be ignored. This warning message is displayed when program is not saved in the flash memory.

Note: If you use CCS v7.0 or higher, when compiling the project, you will see an error message as below:

> Product 'XDAIS' v1.0.0 is not currently installed and no compatible version is available. Please install this product or a compatible version.

To fix the error, right click on the project and select **Properties**. On the dialog window, select **General** on the left menu, and click on the **Products** tab, as shown below.


Properties for test_2_ph_PWM__Mode_1_2_3_

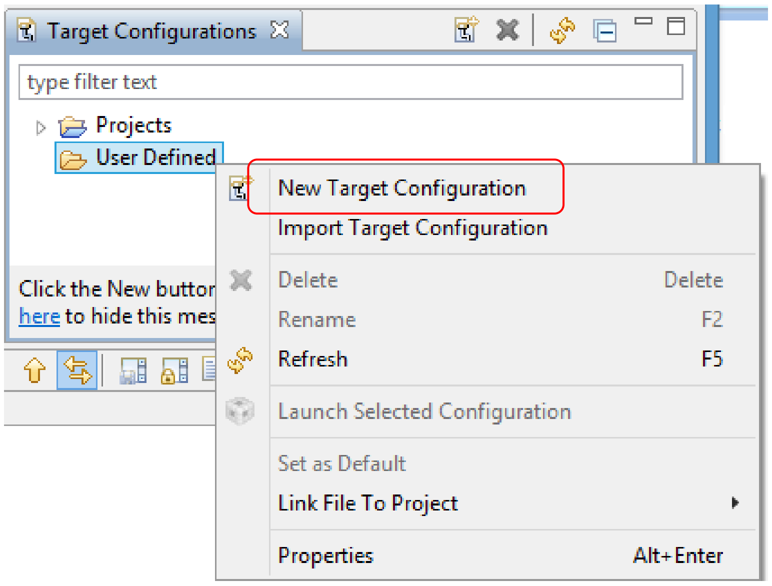
General

Configuration: 1_RamDebug [Active]

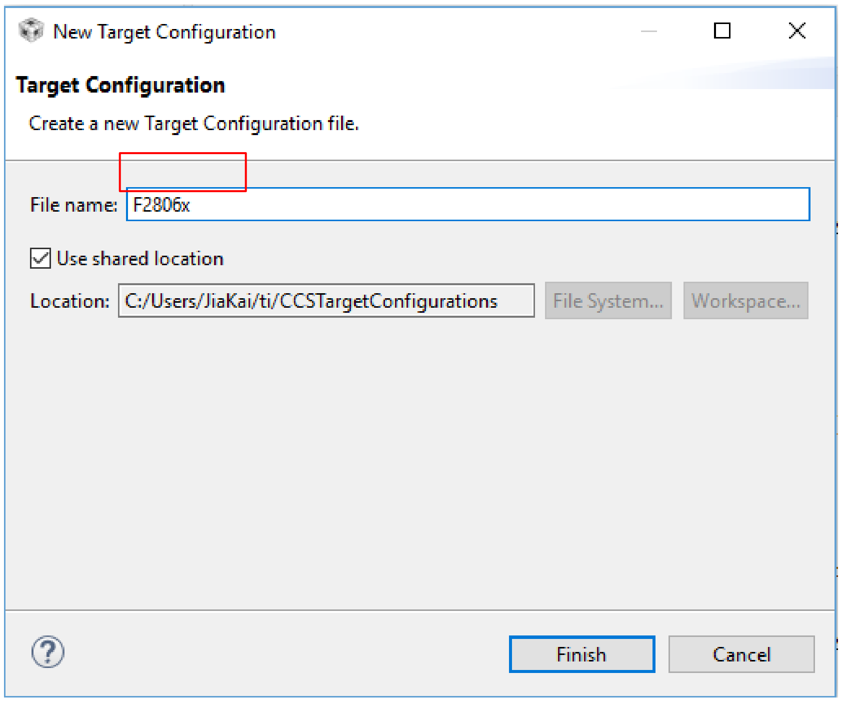
Then uncheck the box XDAIS, and recompile.

CCS Target Configuration

Before loading a program to the target, we need to create a target configuration for the target board. In this example, a TI's TMS320F28069 controlCARD is used. Select **View >> Target Configurations**. A dialog window will appear as follows in CCS.



Right click on **User Defined** in the Target Configuration dialog. Then select **New Target Configuration** in the pop-up menu. A dialog will appear as below.


New Target Configuration

Target Configuration

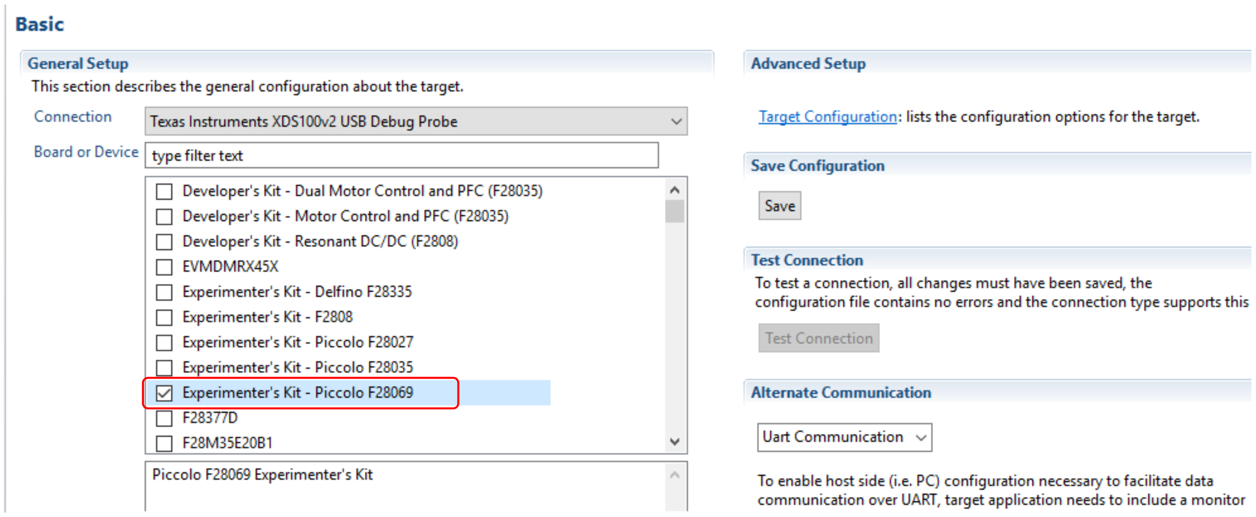
Create a new Target Configuration file.

File name: F2806x

☑ Use shared location

Location: C:/Users/JiaKai/ti/CCSTargetConfigurations

Change the file name as you wish (here it is called "F2806x", and the file extension will be "ccxml"), then click on **Finish**. A dialog named "F2806x.ccxml" will appear as below.



Basic

General Setup

This section describes the general configuration about the target.

Connection: Texas Instruments XDS100v2 USB Debug Probe

Board or Device: type filter text

- ☐ Developer's Kit - Dual Motor Control and PFC (F28035)
- ☐ Developer's Kit - Motor Control and PFC (F28035)
- ☐ Developer's Kit - Resonant DC/DC (F2808)
- ☐ EVMDMRX45X
- ☐ Experimenter's Kit - Delfino F28335
- ☐ Experimenter's Kit - F2808
- ☐ Experimenter's Kit - Piccolo F28027
- ☐ Experimenter's Kit - Piccolo F28035
- ☑ Experimenter's Kit - Piccolo F28069
- ☐ F28377D
- ☐ F28M35E20B1

Piccolo F28069 Experimenter's Kit

Advanced Setup

Target Configuration: lists the configuration options for the target.

Save Configuration

Save

Test Connection

To test a connection, all changes must have been saved, the configuration file contains no errors and the connection type supports this

Test Connection

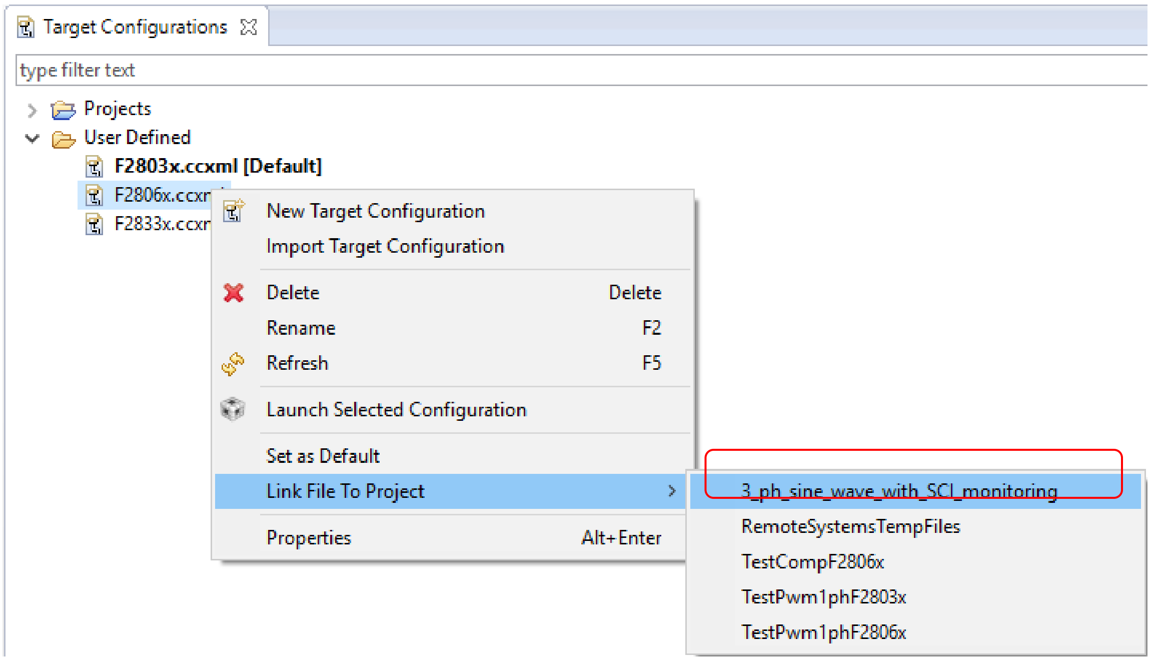
Alternate Communication

Uart Communication

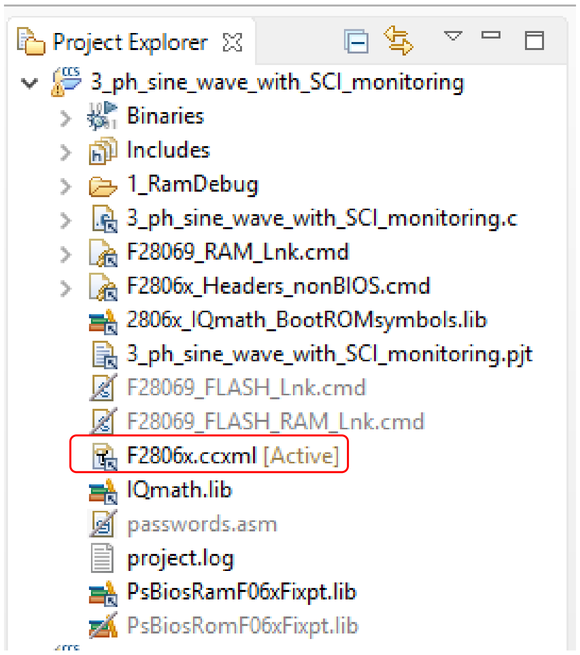
To enable host side (i.e. PC) configuration necessary to facilitate data communication over UART, target application needs to include a monitor

In the "Connection" combo box, choose the emulator you will use (for example, "Texas Instruments XDS100v2 USB Debug Probe"), then check "Experimenter's Kit - Piccolo F28069" in the list box of "Board of Device". Click on **Save** to save the configuration.

Back to the "Target Configurations" dialog, right click on the "F2806x.ccxml" configuration, and move the cursor to "Link File to Project" in the popup menu. All projects will be displayed in the sub-menu. Select the project "3_ph_sine_wave_with_SCI_monitoring" as shown below.



Now, check if "F2806x.ccxml" is added to the project in the Project Explorer, as shown below.



3_ph_sine_wave_with_SCI_monitoring
- Binaries
- Includes
- 1_RamDebug
- 3_ph_sine_wave_with_SCI_monitoring.c
- F28069_RAM_Lnk.cmd
- F2806x_Headers_nonBIOS.cmd
- 2806x_IQmath_BootROMsymbols.lib
- 3_ph_sine_wave_with_SCI_monitoring.pjt
- F28069_FLASH_Lnk.cmd
- F28069_FLASH_RAM_Lnk.cmd
- F2806x.ccxml [Active]
- IQmath.lib
- passwords.asm
- project.log
- PsBiosRamF06xFixpt.lib
- PsBiosRomF06xFixpt.lib

3. Target Hardware Setup

The F28069 Piccolo Experimenter Kit is shown below, with the F28069 controlCARD (non-isolated type).

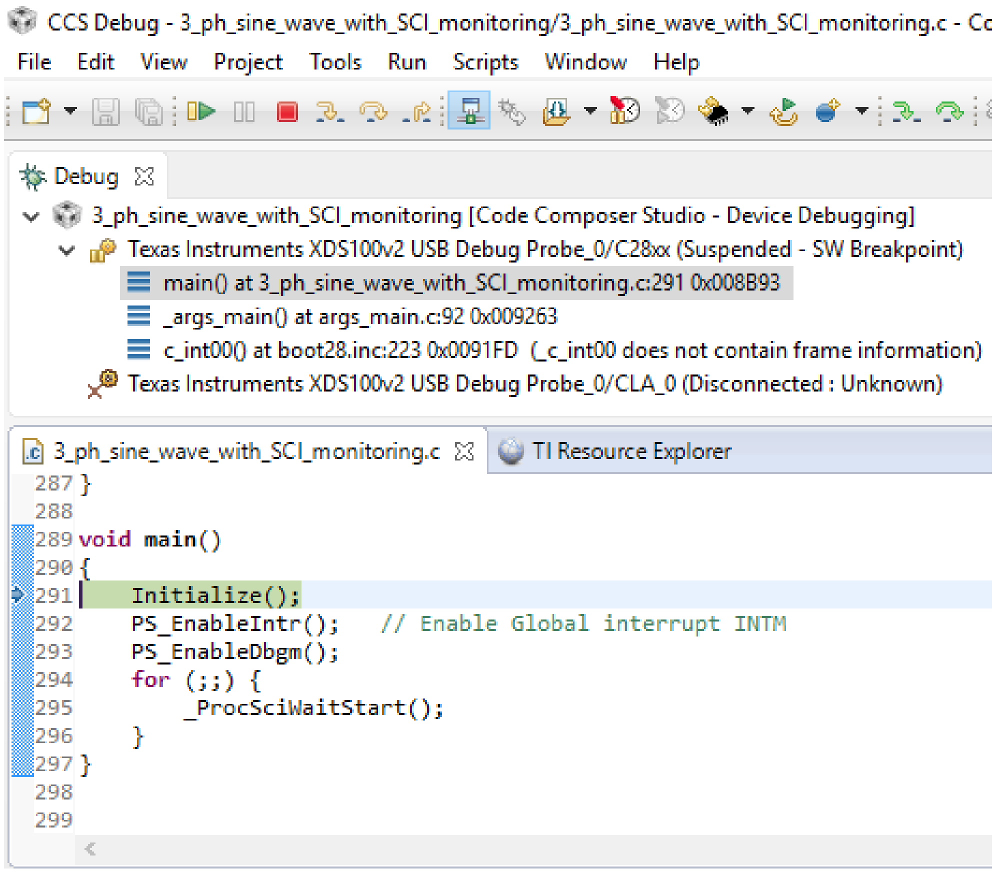
Connect the JP2 connector of the docking station board to the computer’s USB port. The kit provides an on-board XDS100v2 emulator. The USB port also supplies the power to the board when the switch SW1 on the docking station board is at OFF position.

The USB port provides the second channel for RS-232C communication. Note that one needs to short-circuit Jumper J9 on the docking station board in order to use the USB port to communicate with the DSP for waveform monitoring.

4. Running Code in DSP

Loading Code to DSP

In CCS, click on "3_ph_sine_wave_with_SCI_monitoring" project to set it as the current project (the current project name will appear in bold). Then select **Run** >> **Debug** to connect the computer to the DSP. If the connection is successful, the program will be uploaded to the target, and the F28069 DSP will automatically reset and run to the start place of the main function as shown below.


CCS Debug - 3_ph_sine_wave_with_SCI_monitoring/3_ph_sine_wave_with_SCI_monitoring.c - Cc
File Edit View Project Tools Run Scripts Window Help
Debug
TI Resource Explorer

3_ph_sine_wave_with_SCI_monitoring [Code Composer Studio - Device Debugging]
Texas Instruments XDS100v2 USB Debug Probe_0/C28xx (Suspended - SW Breakpoint)
main() at 3_ph_sine_wave_with_SCI_monitoring.c:291 0x008B93
_args_main() at args_main.c:92 0x009263
c_int00() at boot28.inc:223 0x0091FD (_c_int00 does not contain frame information)
Texas Instruments XDS100v2 USB Debug Probe_0/CLA_0 (Disconnected : Unknown)

3_ph_sine_wave_with_SCI_monitoring.c

```
}

void main()
{
    Initialize();
    PS_EnableIntr();   // Enable Global interrupt INTM
    PS_EnableDbgm();
    for (;;) {
        _ProcSciWaitStart();
    }
}
```

Loading Code to DSP (Flash Release Version)

There are 4 configurations in a generated project:

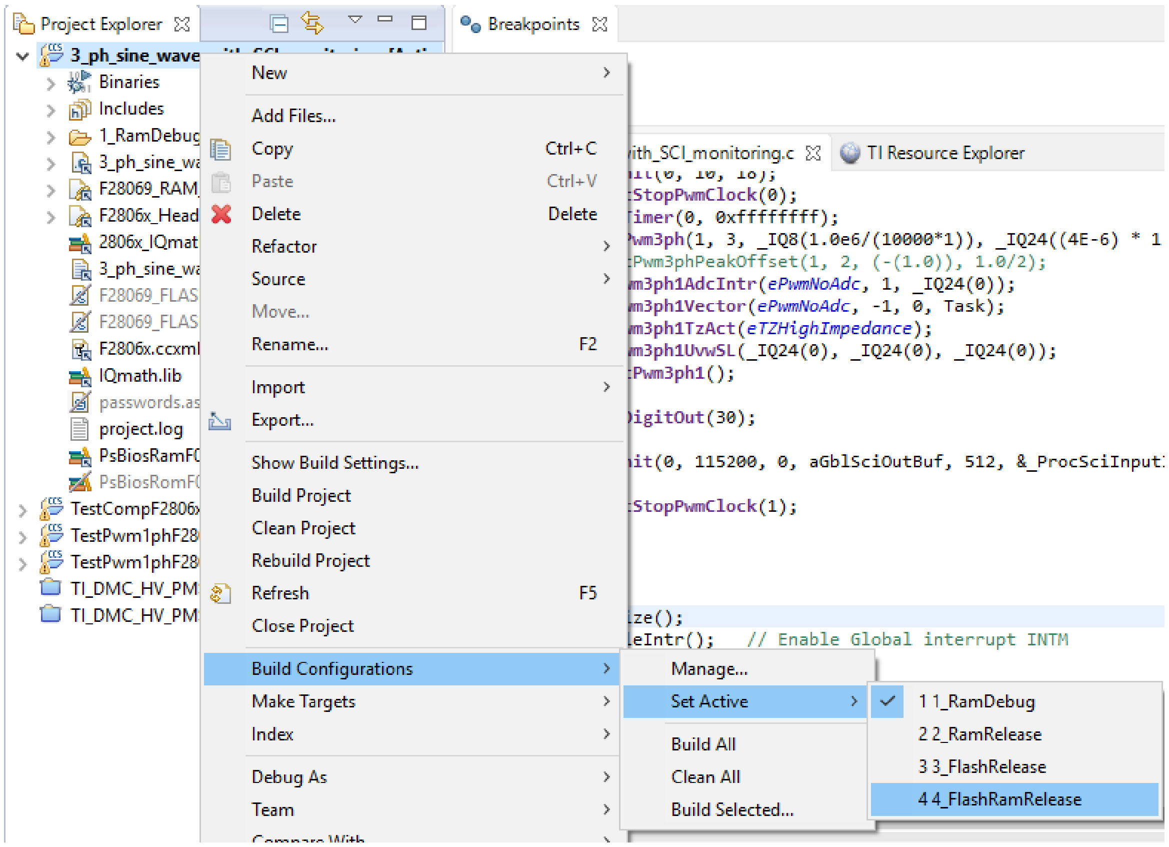
RamRelease:	Load program to RAM area and run in RAM
RamDebug:	Load program to RAM area and run in RAM
FlashRelease:	Load program to flash memory and run in flash
FlashRamRelease:	Load program to flash memory but copy program into RAM and ran in RAM

Note that in order to run the code in either flash release mode or flash RAM release mode so that the code will run when the controlCARD is disconnected from the computer and the DSP board is powered on again, make sure that the controlCARD has the following DIP switch setting:

For the F28069 non-isolated controlCARD, set the switch 2 of the DIP switch SW1 to ON. In this position, the controlCARD can be programmed and can also run in the flash mode.

For the F28069 or F28069M isolated controlCARD, set the switch 1 of the DIP switch SW3 to ON for programming. After the controlCARD is programmed, set the switch to OFF to run in the flash mode.

To use a different configuration, right click on the project name in Project Explorer. Move the cursor to **Build Configurations** in the popup menu and select a configuration of your choice, as shown below.

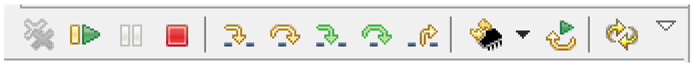


```
:StopPwmClock(0);
Timer(0, 0xffffffff);
Pwm3ph(1, 3, _IQ8(1.0e6/(10000*1)), _IQ24((4E-6) * 1
:Pwm3phPeakOffset(1, 2, (-(1.0)), 1.0/2);
m3ph1AdcIntr(ePwmNoAdc, 1, _IQ24(0));
m3ph1Vector(ePwmNoAdc, -1, 0, Task);
m3ph1TzAct(eTZHighImpedance);
m3ph1UvwSL(_IQ24(0), _IQ24(0), _IQ24(0));
:Pwm3ph1();

DigitOut(30);

it(0, 115200, 0, aGblSciOutBuf, 512, &_ProcSciInput

:StopPwmClock(1);

ze();
eIntr();   // Enable Global interrupt INTM
```

After changing configuration, the project needs to be compiled again, and the code can be uploaded to the DSP with the same procedure as described before.

Run Code in DSP Target

To run the code in the DSP, we can use the toolbar in the "Debug" toolbar as shown below to resume, stop, step into, step over, and step return to run the program.

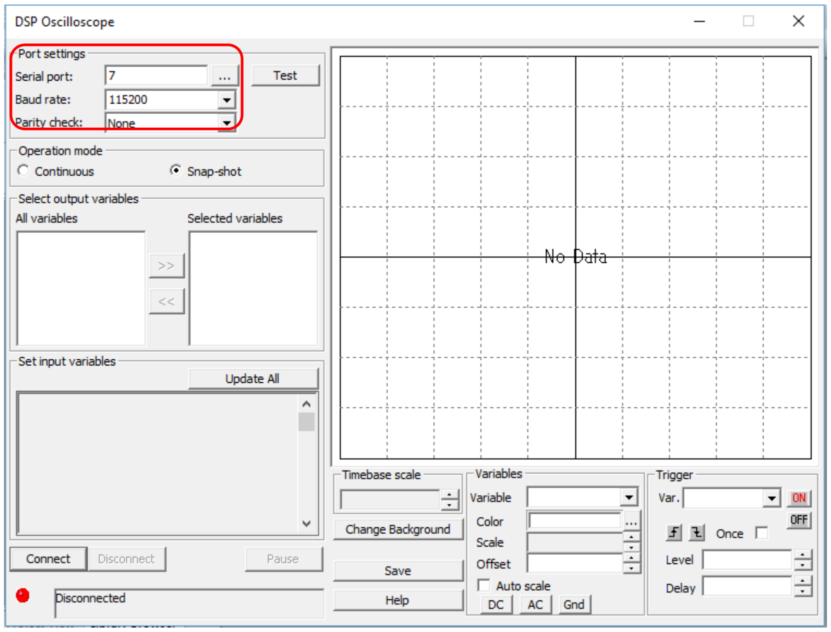


5. Monitoring Waveforms with DSP Oscilloscope

Once the code is running in the targeted DSP, one can use PSIM's DSP Oscilloscope feature to monitor the variables inside the DSP and to control the converter output voltage.

To set up DSP Oscilloscope for waveform monitoring, refer to the tutorial "Tutorial – Using SCI for waveform monitoring.pdf". To launch the DSP Oscilloscope, in PSIM, select **Utilities** >> **DSP Oscilloscope**. The interface is shown below.

Set the correct serial port number, baud rate, and parity check. They must be identical to these in the SCI Configuration block in the PSIM circuit.

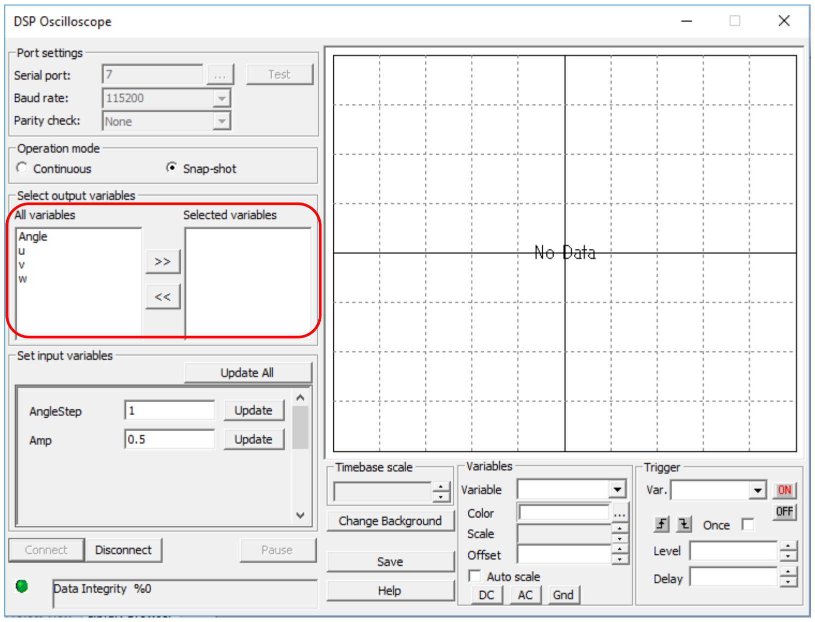

DSP Oscilloscope

Port settings
Serial port: 7
Baud rate: 115200
Parity check: None

Operation mode: Continuous [✓ Snap-shot]

Disconnected

Click the **Connect** button at the left bottom of the scope panel. All names of SCI output and input variables will be listed on the left side of the panel, as shown below.


DSP Oscilloscope

Port settings
Serial port: 7
Baud rate: 115200
Parity check: None

Operation mode: Continuous [✓ Snap-shot]

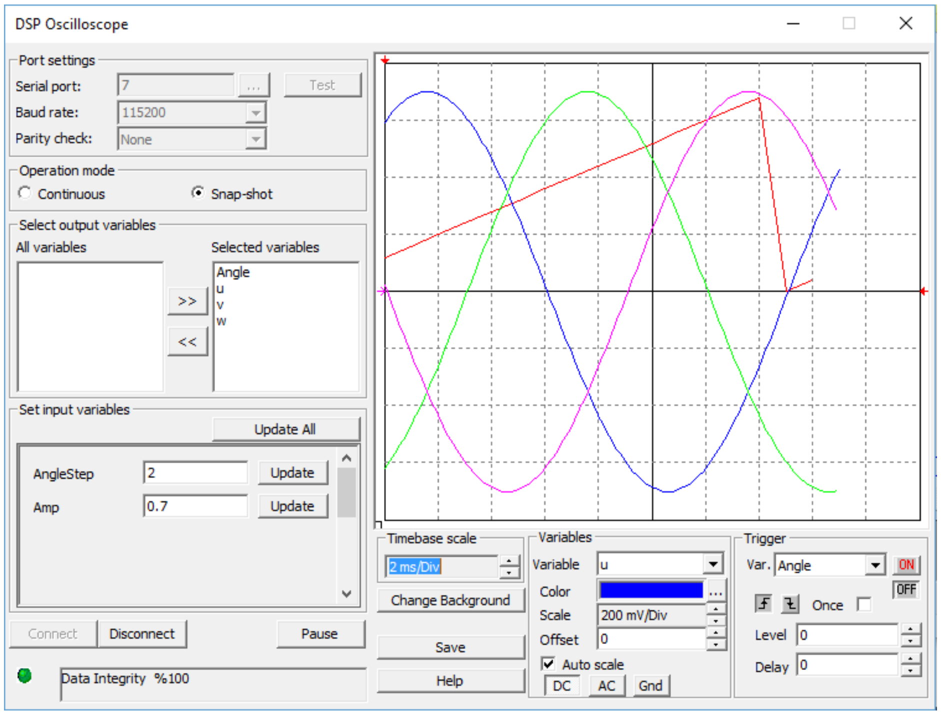
All variables: Angle, u, v, w

Set input variables
AngleStep: 1
Amp: 0.5

Data Integrity %0

The 4 variables available for monitoring are Angle, u, v and w. Select the variables to display on the scope screen.

To change the output of the 3-phase PWM, modify the value AngleStep and/or Amp then click the **Update** button. The Angle waveform will be changed if AngleStep changes; the waveforms of u, v and w will be changed if AngleStep and/or Amp change. The figure below shows the waveforms with the new values for Angle and Amp.


DSP Oscilloscope

Port settings: Serial port: 7; Baud rate: 115200; Parity check: None

Operation mode: Continuous [✓ Snap-shot]

Select output variables — Selected variables: Angle, u, v, w

Set input variables: AngleStep 2; Amp 0.7

Data Integrity %100

Timebase scale: 2 ms/Div

Variables: Variable u; Scale 200 mV/Div; Offset 0; ✓ Auto scale

Trigger: Var. Angle; ON; Once ☐; Level 0; Delay 0