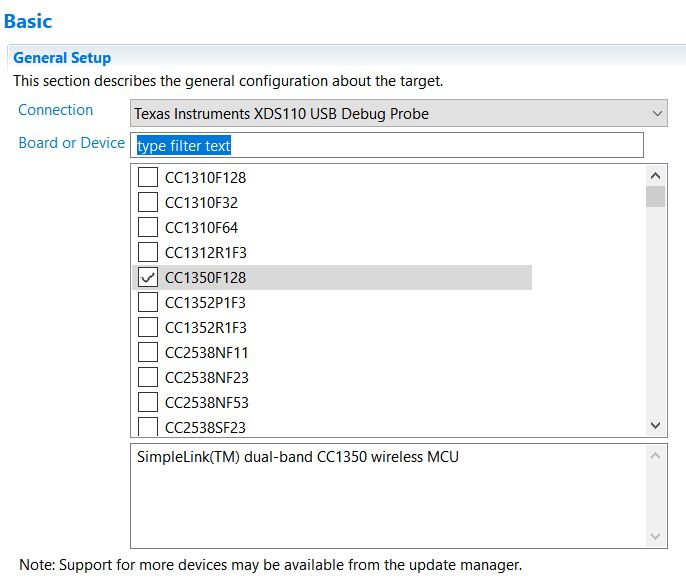
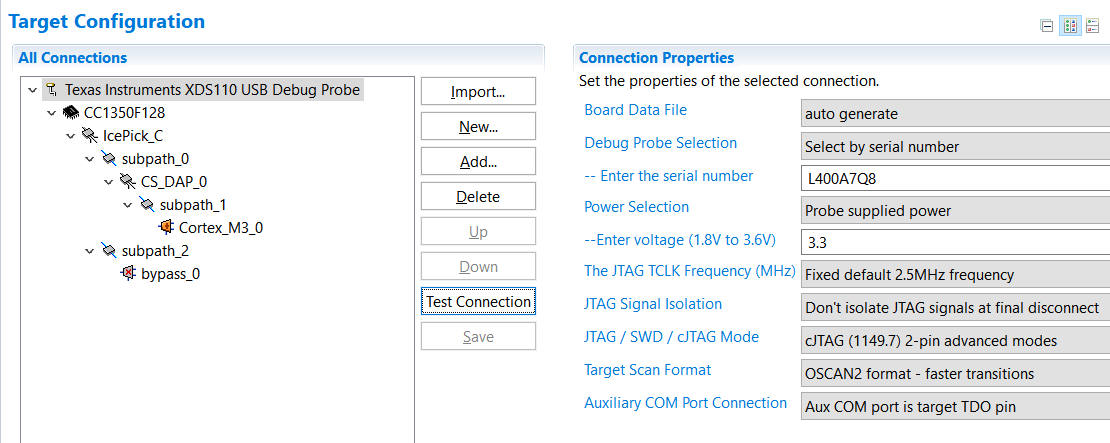
**Date Submitted: 11-25-2018**

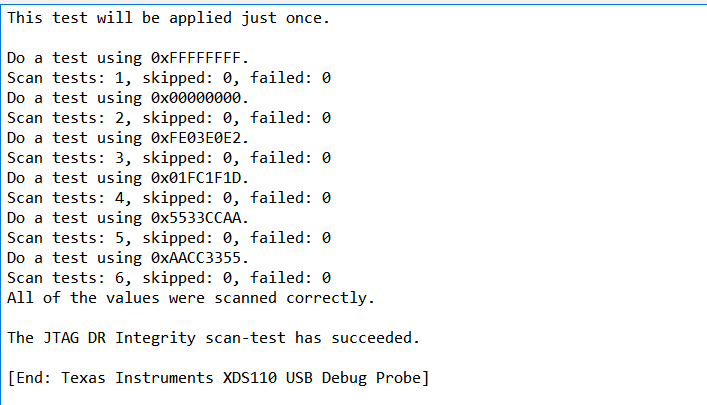
**Task 01: Set up Project in SCS then import to CCS and initialize connection via targetConfigs.**



Defining which board to use through config setup.



Target configuration set. Serial changed.



Testing connection after changing serial being successful.

Youtube Link: No video needed for this part. Only screenshots.

**Modified Code:** No coding in this task.

**------------------------------------------------------------------------------------**

**Task 02: Download and Debug with CCS.**

Youtube Link: No video for this task since it was just building and debug. There is nothing being shown besides placing breakpoints.

**Modified Code:**

Task02.c

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

// SENSOR CONTROLLER STUDIO EXAMPLE: ADC WINDOW MONITOR FOR LAUNCHPAD

// Operating system: TI-RTOS

//

// The Sensor Controller is used to sample a single ADC channel and monitor

// the value. The Sensor Controller updates a bit-vector that indicates

// whether the ADC value is:

// - Below a configurable low threshold

// - Above a configurable high threshold

//

// The Sensor Controller notifies the application when the bit-vector changes

// (triggering scTaskAlertCallback()), and the application sets the LEDs as

// follows:

// - Green LED is set whenever the ADC value is below the low threshold

// - Red LED is set whenever the ADC value is above the high threshold

//

//

// Copyright (C) 2017 Texas Instruments Incorporated - http://www.ti.com/

//

//

// Redistribution and use in source and binary forms, with or without

// modification, are permitted provided that the following conditions

// are met:

//

// Redistributions of source code must retain the above copyright

// notice, this list of conditions and the following disclaimer.

//

// Redistributions in binary form must reproduce the above copyright

// notice, this list of conditions and the following disclaimer in the

// documentation and/or other materials provided with the distribution.

//

// Neither the name of Texas Instruments Incorporated nor the names of

// its contributors may be used to endorse or promote products derived

// from this software without specific prior written permission.

//

// THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS

// "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT

// LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR

// A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT

// OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL,

// SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT

// LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE,

// DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY

// THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT

// (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE

// OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

//\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

**#include** "ex\_include\_tirtos.h"

**#include** "scif.h"

**#define** BV(n) (1 << (n))

// Display error message if the SCIF driver has been generated with incorrect operating system setting

**#if** !(defined(SCIF\_OSAL\_TIRTOS\_H) || defined(SCIF\_OSAL\_TIDPL\_H))

**#error** "SCIF driver has incorrect operating system configuration for this example. Please change to 'TI-RTOS' or 'TI Driver Porting Layer' in the Sensor Controller Studio project panel and re-generate the driver."

**#endif**

// Display error message if the SCIF driver has been generated with incorrect target chip package

**#ifndef** SCIF\_TARGET\_CHIP\_PACKAGE\_QFN48\_7X7\_RGZ

**#error** "SCIF driver has incorrect target chip package configuration for this example. Please change to 'QFN48 7x7 RGZ' in the Sensor Controller Studio project panel and re-generate the driver."

**#endif**

// Task data

Task\_Struct myTask;

Char myTaskStack[1024];

// Semaphore used to wait for Sensor Controller task ALERT event

**static** Semaphore\_Struct semScTaskAlert;

**void** **scCtrlReadyCallback**(**void**) {

} // scCtrlReadyCallback

**void** **scTaskAlertCallback**(**void**) {

// Wake up the OS task

Semaphore\_post(Semaphore\_handle(&semScTaskAlert));

} // scTaskAlertCallback

PIN\_Config pLedPinTable[] = {

Board\_GLED | PIN\_GPIO\_OUTPUT\_EN | PIN\_GPIO\_LOW | PIN\_PUSHPULL | PIN\_DRVSTR\_MAX,

Board\_RLED | PIN\_GPIO\_OUTPUT\_EN | PIN\_GPIO\_LOW | PIN\_PUSHPULL | PIN\_DRVSTR\_MAX,

PIN\_TERMINATE

};

PIN\_State ledPinState;

**void** **taskFxn**(UArg a0, UArg a1) {

PIN\_Handle hLedPins;

// Enable LED pins

hLedPins = PIN\_open(&ledPinState, pLedPinTable);

// Initialize the Sensor Controller

scifOsalInit();

scifOsalRegisterCtrlReadyCallback(scCtrlReadyCallback);

scifOsalRegisterTaskAlertCallback(scTaskAlertCallback);

scifInit(&scifDriverSetup);

scifStartRtcTicksNow(0x00010000 / 8);

// Configure and start the Sensor Controller's ADC window monitor task (not to be confused with OS tasks)

scifTaskData.adcWindowMonitor.cfg.adcWindowHigh = 800;

scifTaskData.adcWindowMonitor.cfg.adcWindowLow = 400;

scifStartTasksNbl(BV(SCIF\_ADC\_WINDOW\_MONITOR\_TASK\_ID));

// Main loop

**while** (1) {

// Wait for an ALERT callback

Semaphore\_pend(Semaphore\_handle(&semScTaskAlert), BIOS\_WAIT\_FOREVER);

// Clear the ALERT interrupt source

scifClearAlertIntSource();

// Indicate on LEDs whether the current ADC value is high and/or low

**if** (scifTaskData.adcWindowMonitor.output.bvWindowState & SCIF\_ADC\_WINDOW\_MONITOR\_BV\_ADC\_WINDOW\_LOW) {

PIN\_setOutputValue(hLedPins, Board\_GLED, 1);

} **else** {

PIN\_setOutputValue(hLedPins, Board\_GLED, 0);

}

**if** (scifTaskData.adcWindowMonitor.output.bvWindowState & SCIF\_ADC\_WINDOW\_MONITOR\_BV\_ADC\_WINDOW\_HIGH) {

PIN\_setOutputValue(hLedPins, Board\_RLED, 1);

} **else** {

PIN\_setOutputValue(hLedPins, Board\_RLED, 0);

}

// Acknowledge the alert event

scifAckAlertEvents();

}

} // taskFxn

**int** **main**(**void**) {

Task\_Params taskParams;

// Initialize the board

Board\_initGeneral();

**#ifdef** Board\_shutDownExtFlash

Board\_shutDownExtFlash();

**#endif**

// Configure the OS task

Task\_Params\_init(&taskParams);

taskParams.stack = myTaskStack;

taskParams.stackSize = **sizeof**(myTaskStack);

taskParams.priority = 3;

Task\_construct(&myTask, taskFxn, &taskParams, NULL);

// Create the semaphore used to wait for Sensor Controller ALERT events

Semaphore\_Params semParams;

Semaphore\_Params\_init(&semParams);

semParams.mode = Semaphore\_Mode\_BINARY;

Semaphore\_construct(&semScTaskAlert, 0, &semParams);

// Start TI-RTOS

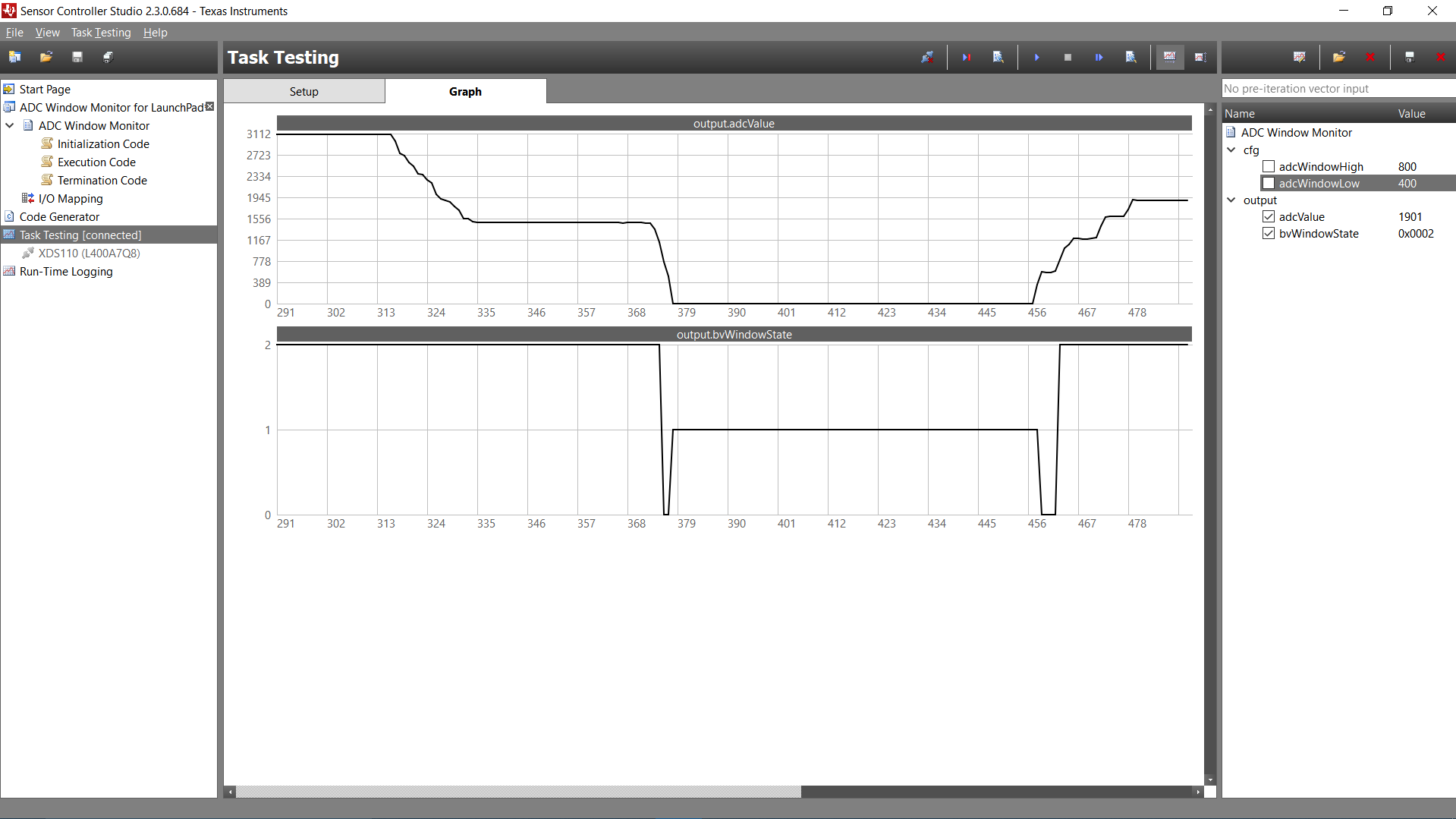
BIOS\_start();

**return** 0;

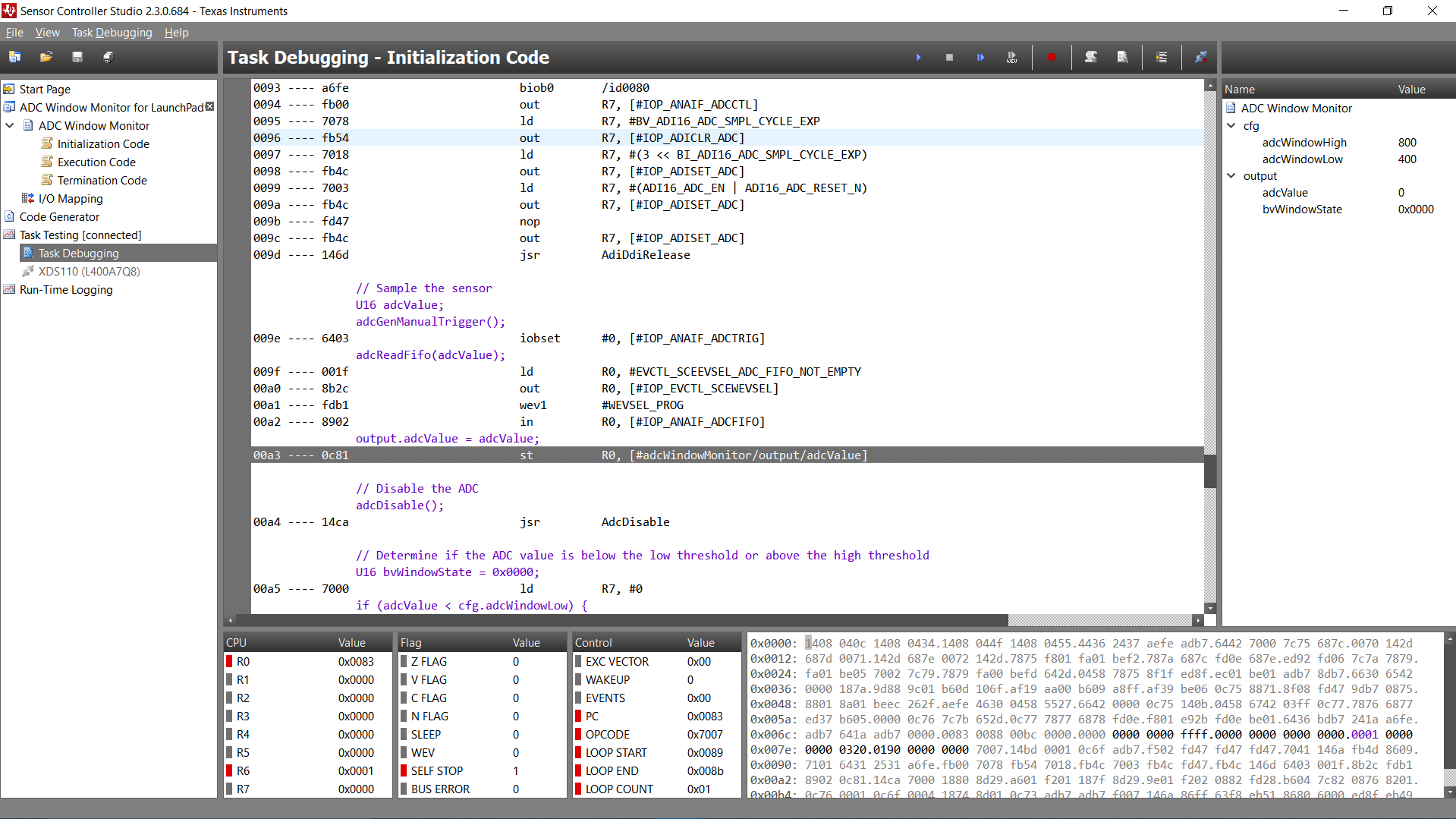
} // main

**------------------------------------------------------------------------------------**

**Task 03: Download and Debug with SCS**



Visual representation of ADC value using potentiometer. It starts with high and slowly goes down to low as I turned the potentiometer.



Task test debugging.

Youtube Link: See screenshots for visual representation.

**Modified Code:** Same code as nothing has changed.