Displaying POT Level with LEDs

Libero SoC and µVision4 IDE Flow Tutorial for SmartFusion cSoC

Table of Contents

Introduction

This tutorial shows you how to develop an application that can be implemented on a SmartFusion® customized system-on-chip (cSoC) device. After completing this tutorial you will be familiar with the following:

- Creating and implementing a Libero® system-on-chip (SoC) v10.0 project using a SmartFusion cSoC
- Configuring the peripherals using SmartDesign
- Configuring the analog compute engine (ACE)
- Generating the microcontroller subsystem (MSS) Component
- Generating the programming file to program the SmartFusion cSoC device
- Opening the project in µVision®4 IDE and Debugger tool from Libero SoC and writing application code
- Compiling application code
- Creating and launching a debug session
- Debugging and running the code using Keil µVision®4 IDE

Tutorial Requirements

Software Requirements

This tutorial requires the following software installed on your PC:

- Libero SoC v10.0 (or later) can be downloaded from www.microsemi.com/soc/download/software/libero/default.aspx.
- Keil µVision®4.11 version or later

Hardware Requirements

This tutorial requires the following hardware:

- SmartFusion Evaluation Kit Board or SmartFusion Development Board.
- Two USB cables (programming and communication)—one for connecting the programmer to your PC and the other to connect the universal asynchronous receiver/transmitter (UART) interface on the board to the PC.
- Keil supplied ULINK2 or ULINK-ME debugger hardware (not supplied with the SmartFusion Kit Board)

Associated Project Files

You can download the associated project files for this tutorial from the Microsemi website: www.microsemi.com/soc/download/rsc/?f=SmartFusion_Libero_Keil_POTlevel_tut_DF.

Note: Extract design files to root directory.

You can download the programming file (*.stp) in release for this tutorial from the Microsemi website:

www.microsemi.com/soc/download/rsc/?f=SmartFusion_Libero_Keil_POTlevel_tut_PF.

MSS Components Used

- ARM® Cortex™-M3 processor
- Clock conditioning circuitry (CCC)
- General purpose input/output (GPIO)
- UART_0
- Analog compute engine (ACE)

Target Board

SmartFusion Evaluation Kit Board (A2F-EVAL-KIT) or SmartFusion Development Kit Board (A2F-DEV-KIT).

Objective

The objective of this tutorial is to instruct how to configure SmartFusion analog channels and ACE that is used to monitor the voltage across the potentiometer (POT). The UART is used to send the analog-to-digital converter (ADC) results to a terminal program.

Design Steps

Following are the major steps to be executed for this tutorial:

- Create a Libero SoC v10.0 project for SmartFusion cSoC.
- Configure the SmartFusion cSoC peripherals.
- Generate the SmartFusion cSoC MSS component.
- Perform synthesis and layout, and generate a programming file to program the SmartFusion cSoC device.
- Program the SmartFusion A2F200M3F or A2F500M3F cSoC device.
- Open the software project in µVision®4 IDE and write the application program.
- Run an application to monitor the voltage across the POT on the SmartFusion Evaluation Kit Board or Development Kit Board.

The hardware configuration has four flags:

- Over 1.0 V
- Over 1.5 V
- Over 2.0 V
- Over 2.5 V

The design monitors the voltage across a POT and four flags are included for the voltage monitoring. These flags are used to drive the four LEDs on the board.

Working with Libero SoC and µVision

This section describes how to create a Libero SoC project, configure the microcontroller subsystem (MSS), program the design on the SmartFusion board, and run an application program in the µVision4 IDE.

Step 1 - Creating a Libero SoC Project

1. Launch Libero SoC v10.0 (or later).
2. From the **Project** menu, select **New Project**. Enter the information as displayed in Figure 1 · .
   - Name: Voltage_Monitor
   - Location: <..> (For example, C:\Microsemiprj\ POT_LED_Libero_Keil)
   - Family: SmartFusion
   - Die: If you are using SmartFusion Evaluation Kit Board, enter A2F200M3F; if you are using SmartFusion Development Kit Board, enter A2F500M3F.
   - Package: 484 FBGA
   - Speed: STD and leave others as default

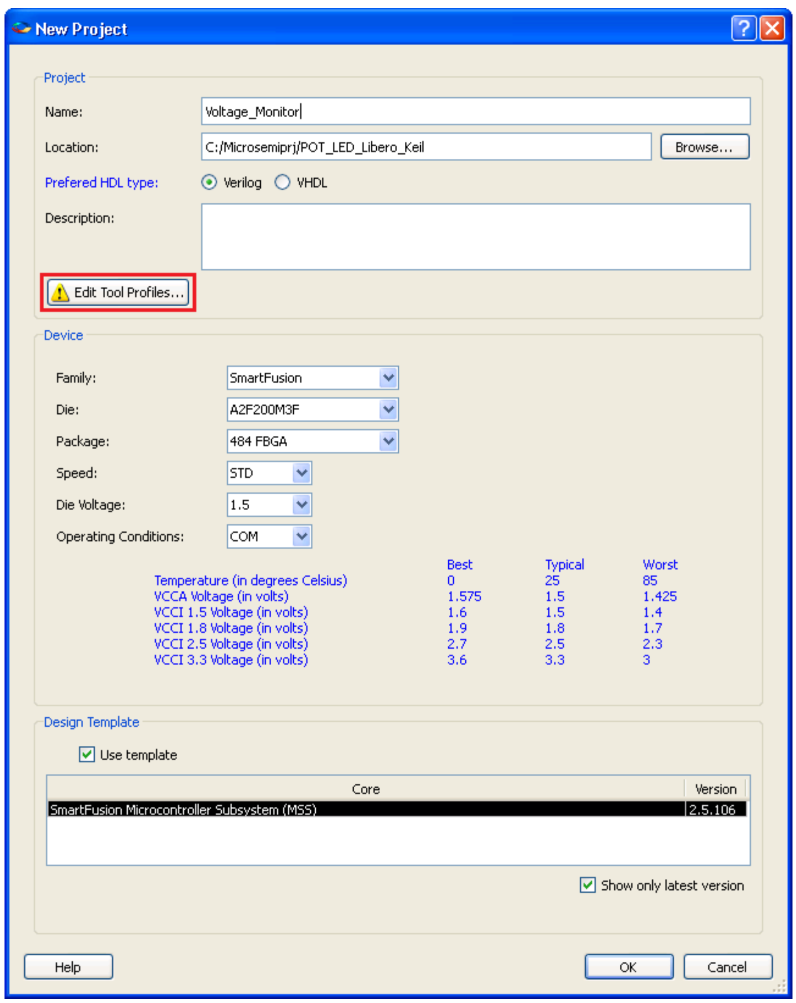

Figure 1 · New Project Dialog Box

3. Click **Edit Tool Profiles** and add Keil by clicking on Software IDE as shown in **Error! Reference source not found.**.

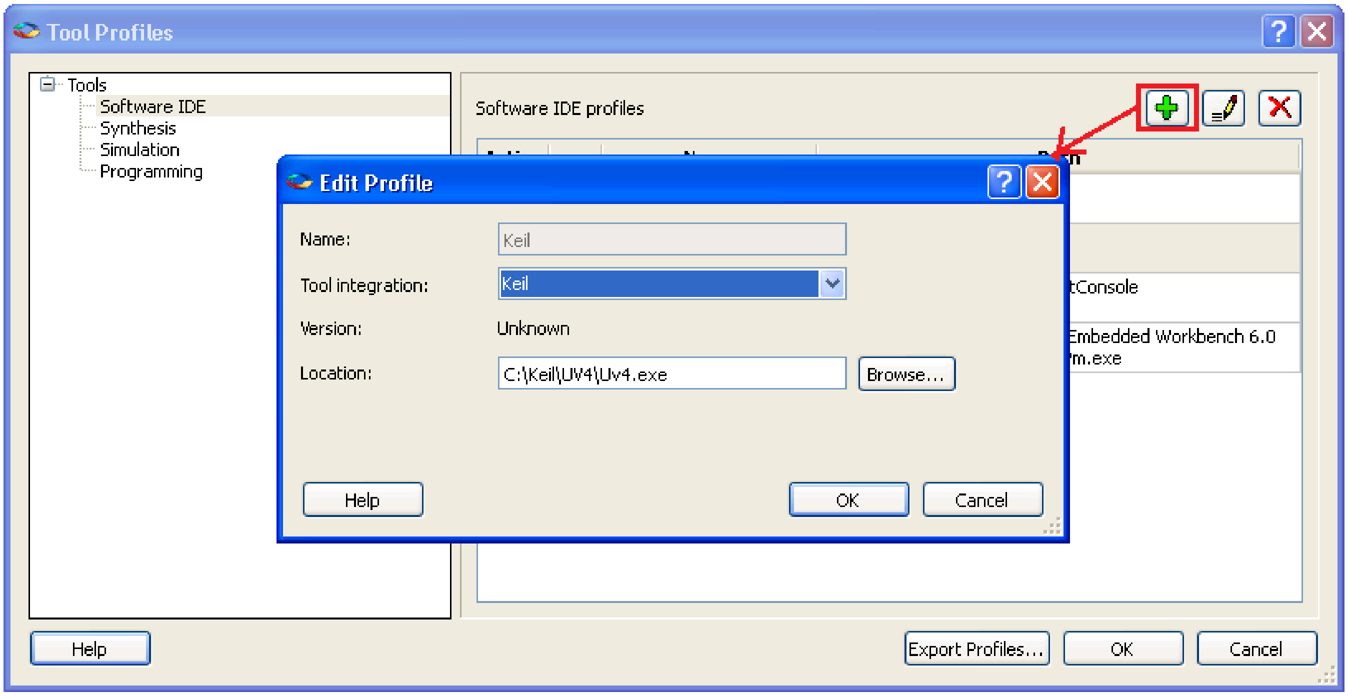

Figure 2 · Selecting Kiel as Software IDE

4. After adding the Profile, click **OK** to close the Tool Profiles Dialog Window. Repeat the steps (3 and 4) above for Synthesis, Simulation, and Programming and then click **OK** to close the Tool Profile dailog window.
5. Select the MSS core in New Project Dialog Box and click **OK.**

Note: If SmartFusion cSoC MSS does not appear in the list, refer to the Appendix A – Libero SoC Catalog Settings to find out how to set your repositories. If your vault does not have the MSS core, download the core by double clicking on the core name in Design template in New Project Dialog Box.

6. The project is created and the Libero SoC window is displayed as shown in Figure 3 · . The SmartDesign “Voltage_Monitor” is created with the instantiation of the MSS component.

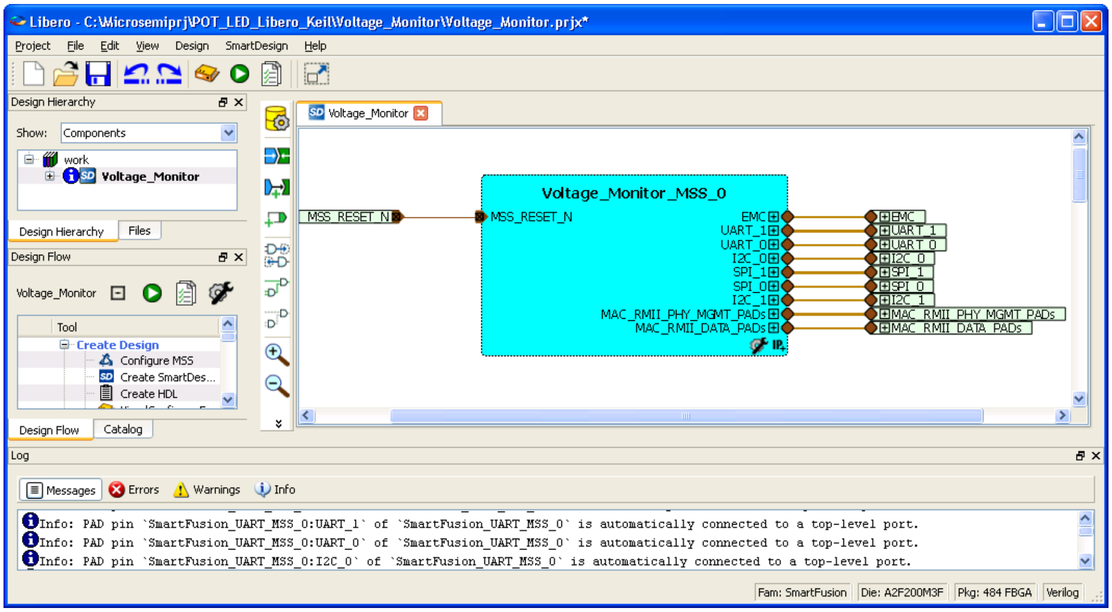

Figure 3 · The Libero Window After Completing New Project Wizard

Step 2 - Configuring MSS Peripherals

1. Double-click on **Voltage_Monitor_MSS_0** to configure the MSS**.** The MSS is displayed in the SmartDesign Canvas in a new tab as shown in Figure 4 · .

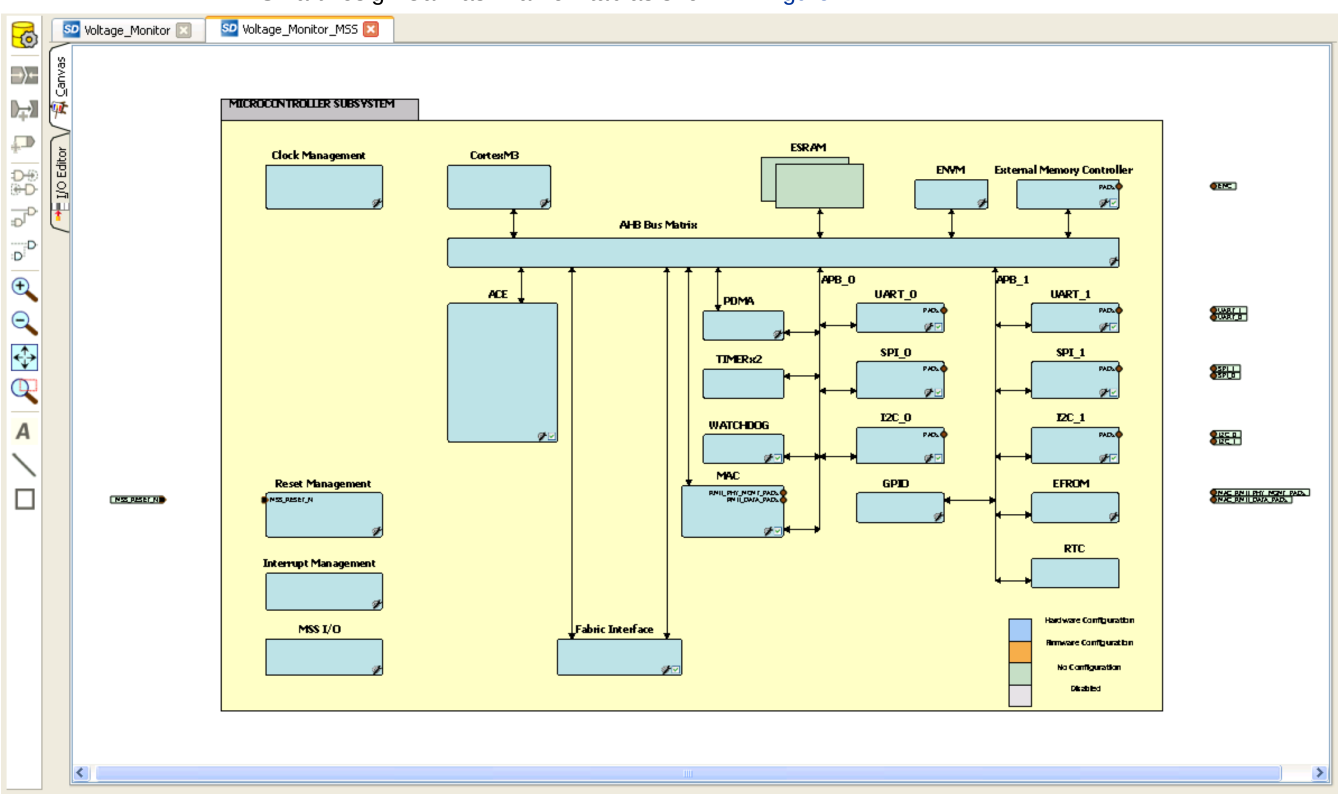

Figure 4 · MSS in the SmartDesign Canvas

The enabled MSS are highlighted in blue, and can be configured in the hardware. The disabled peripherals are shown in gray.

To disable a peripheral that is not required, select the peripheral, right-click, and clear the Enabled check box, or clear the check box in the lower right corner of the peripheral box. The box turns grey to indicate that the peripheral has been disabled. Disabled peripherals can be enabled by repeating the procedure. An enabled peripheral is shown in Figure 5 · .

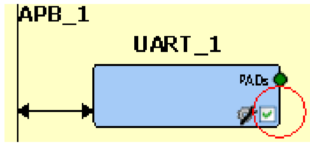

Figure 5 · Enabling the Peripheral

This example uses only the clock management, analog compute engine (ACE), GPIO, and UART_0 peripherals.

2. Disable the following peripherals: MAC, WATCHDOG, Fabric Interface, SPI0, SPI1, I2C0, I2C1, UART1, and EMC.

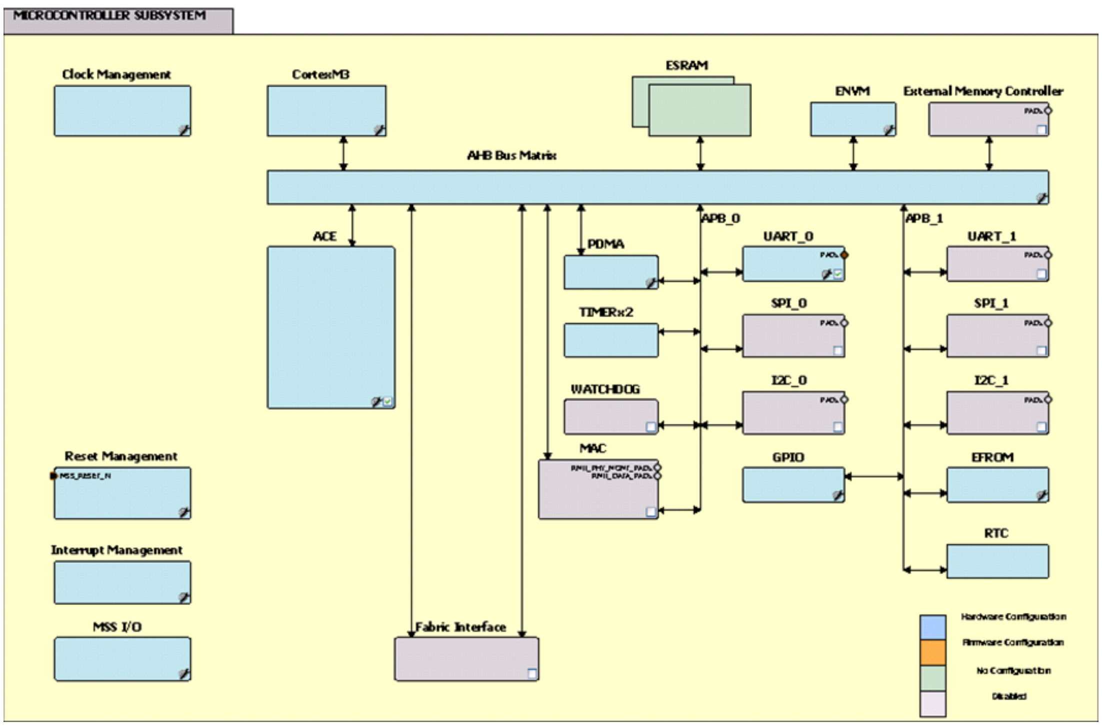

Figure 6 · Used MSS Peripherals

3. Double-click the **Clock Management** block and configure as shown below:
   - CLKA: On-chip RC Oscillator
   - MSS clock source: PLL output
   - MSS clock frequency: 80 MHz

   Use default settings for all other fields.

4. After completing the configuration, click **OK**

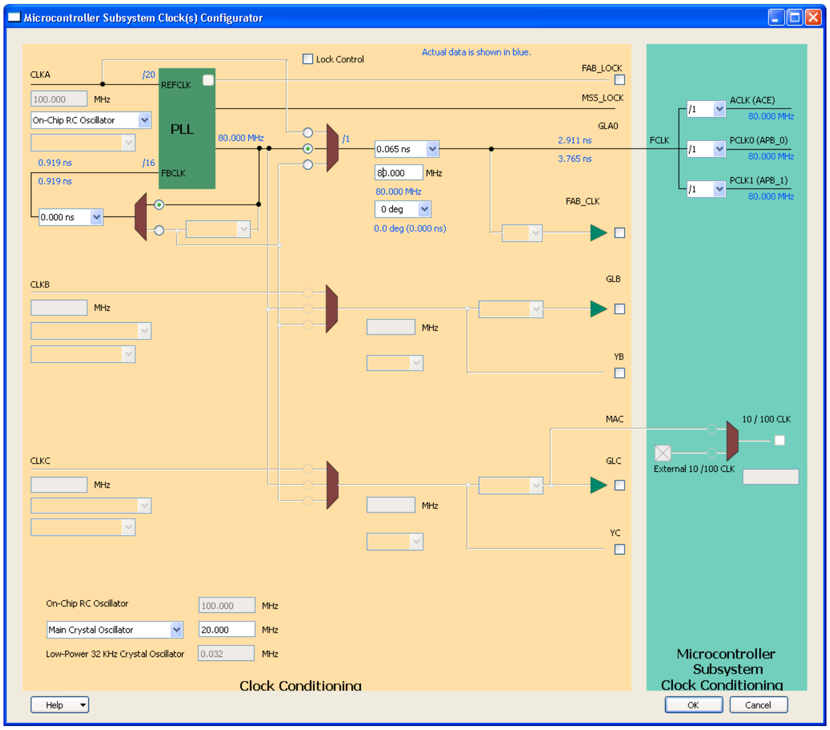

Figure 7 · MSS Clock Configuration

Configuring ACE

1. To configure ACE, double-click the ACE peripheral block and configure as follows:

   Connect TM0 to the POT on the SmartFusion Evaluation Kit Board or Development Kit Board. Configure a voltage monitor to measure the voltage across the POT and also to create flags to indicate when the voltage is greater than 1.0 V, 1.5 V, 2.0 V, and 2.5 V. These flags are used to illuminate the LEDs on the SmartFusion Evaluation Kit Board or Development Kit Board.

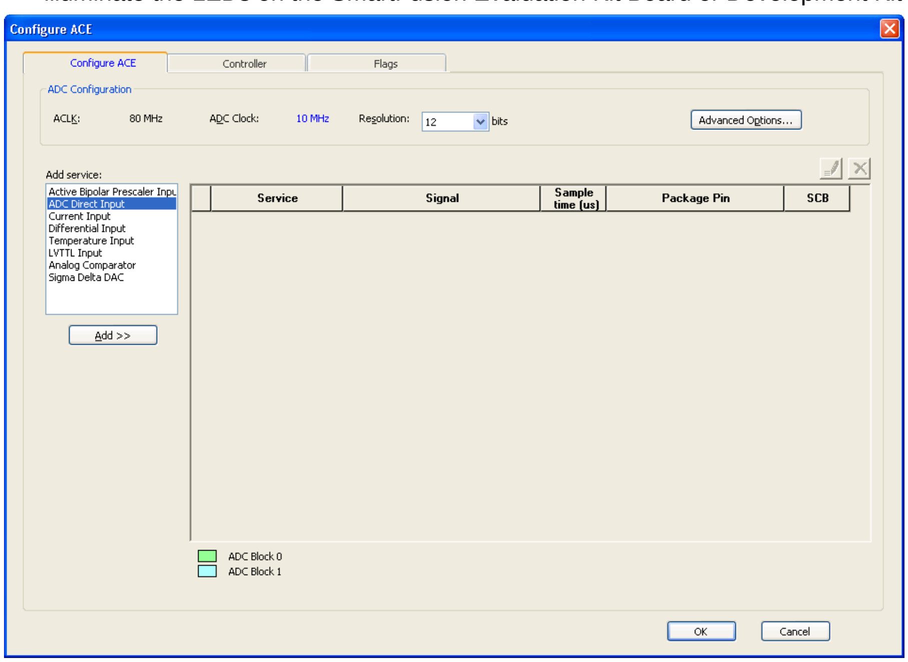

Figure 8 · MSS ACE Configuration

2. Select **ADC Direct Input** > **Add** (or, double-click ADC Direct Input) and enter the parameters as shown in Figure 9 · :
   - Signal name: TM0_Voltage
   - Send raw results to DMA: Cleared check box
   - Acquisition time: 10 µs
   - Filtering factor: None

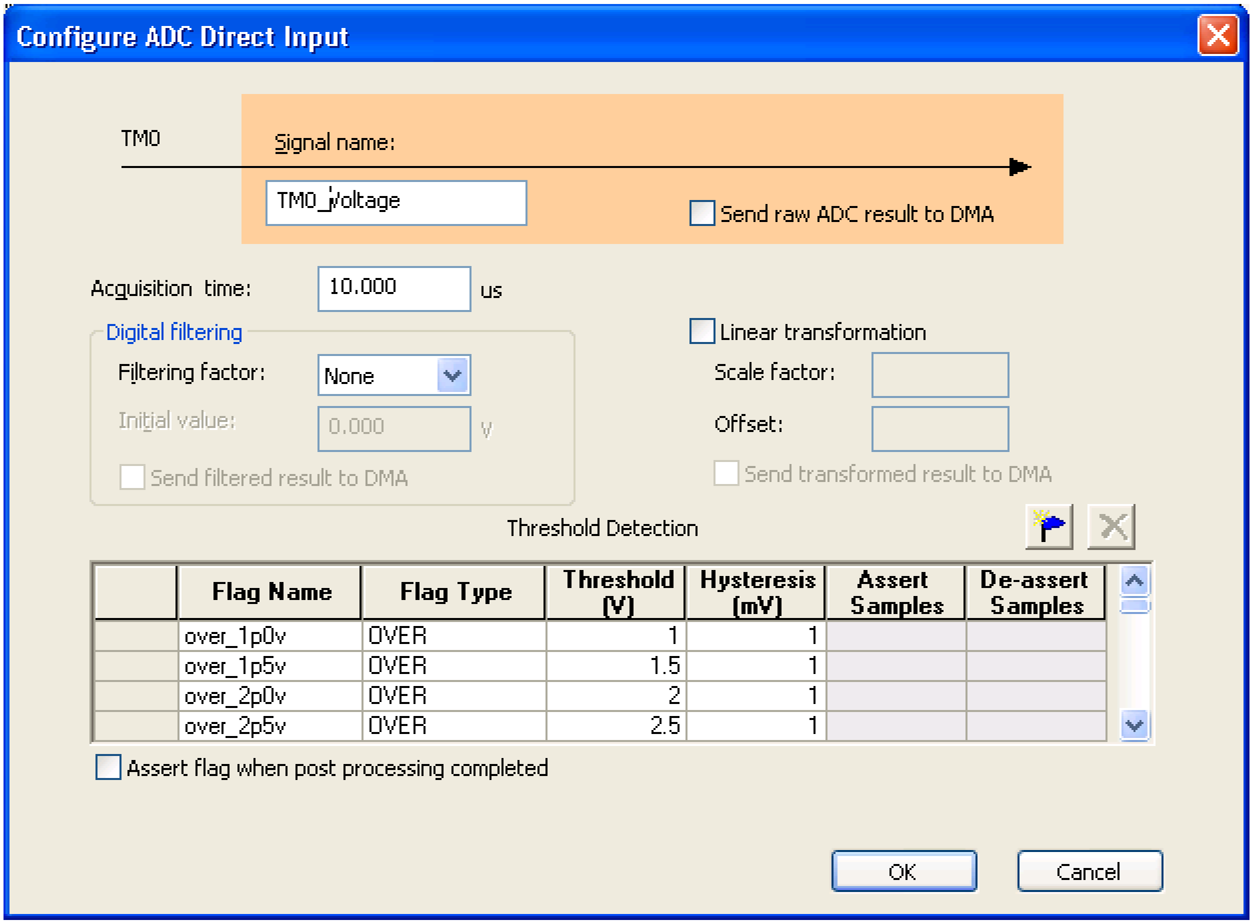

Figure 9 · MSS ADC Direct Input Configuration

3. Next, add the flags as shown in Table 1:

Table 1 · Flag Definitions

Flag Name	Flag Type	Threshold (V)	Hysteresis (mV)
over_1p0v	OVER	1	1
over_1p5v	OVER	1.5	1
over_2p0v	OVER	2	1
over_2p5v	OVER	2.5	1

4. Click **OK.**

5. Assign the ADC Direct Input Signal to package pin W8 in the Configure ACE dialog box. The **Configure ACE** tab is displayed as shown in **Error! Reference source not found.**

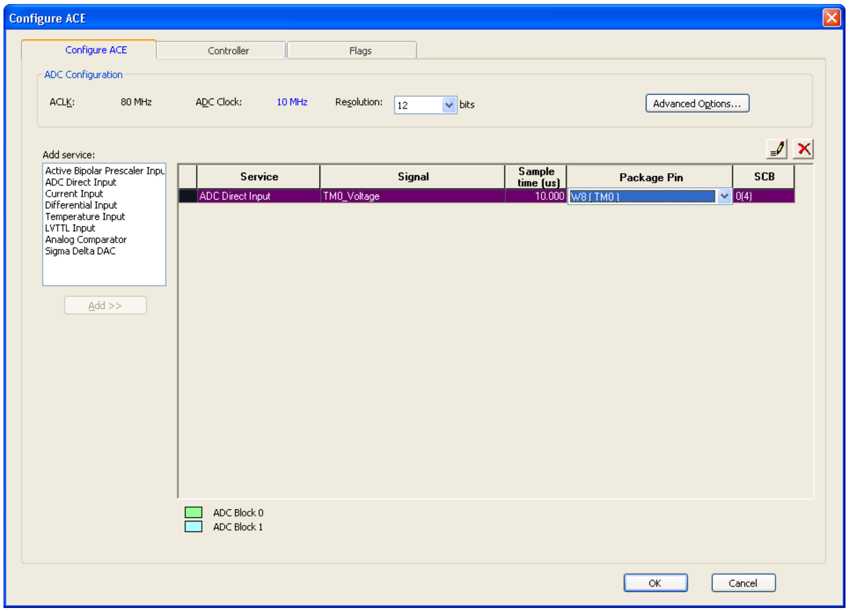

Figure 10 · MSS ACE Configuration With ADC Direct Input

6. The next step in configuring the ACE is to enable the sampling sequence. This configuration dialog is launched by clicking on the **Controller** tab (next to the **Configure ACE** tab).
7. Select **Manual** as the **Operating sequence** entry in the **Controller** tab.

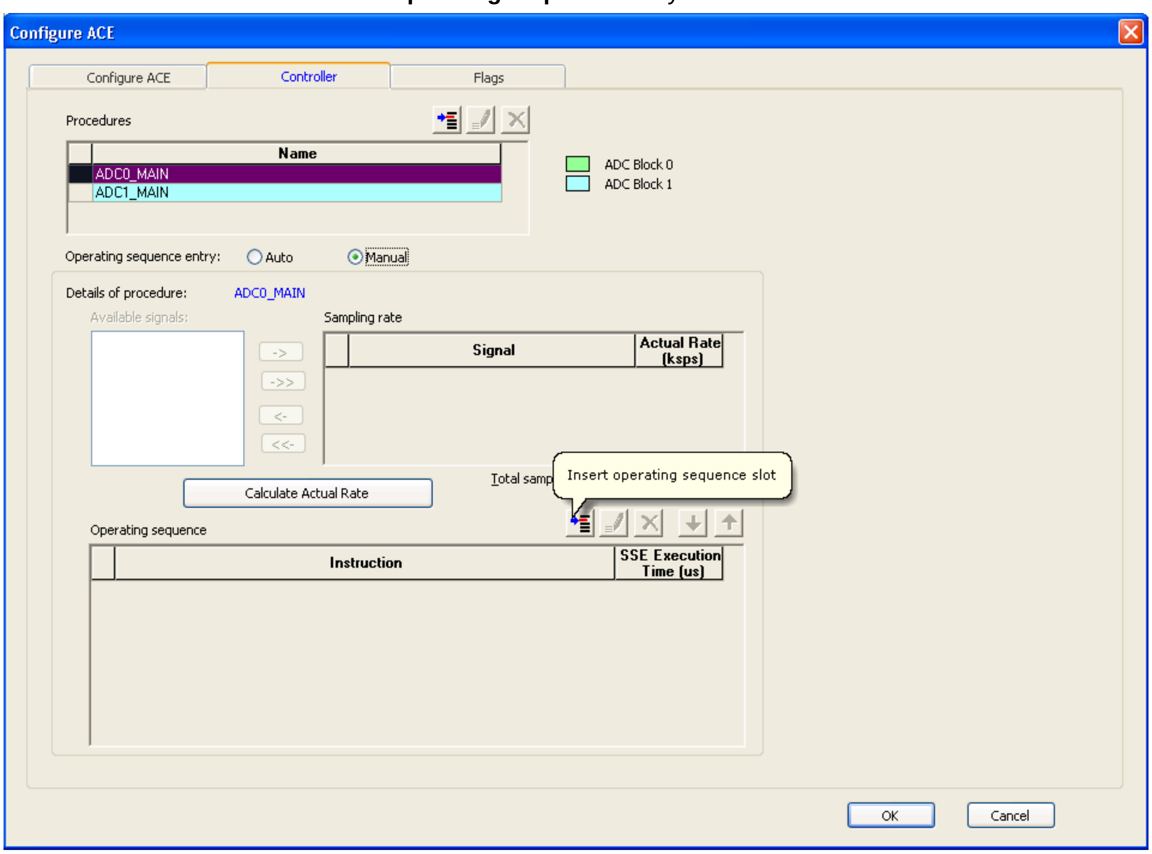

Figure 11 · MSS ACE Configuration to Enable Sampling Sequence

8. Click **Insert operating sequence slot** as shown in Figure 11 · .
9. Select **SAMPLE**.

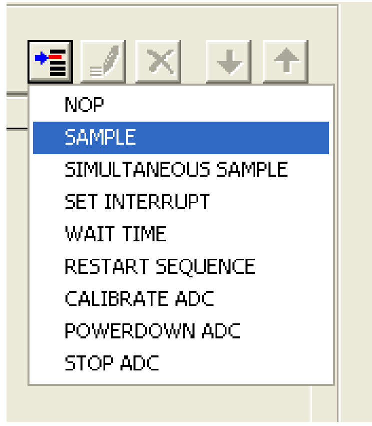

Figure 12 · Select SAMPLE

10. The Configure SAMPLE window is displayed. Select **TM0_voltage** and click **OK**.

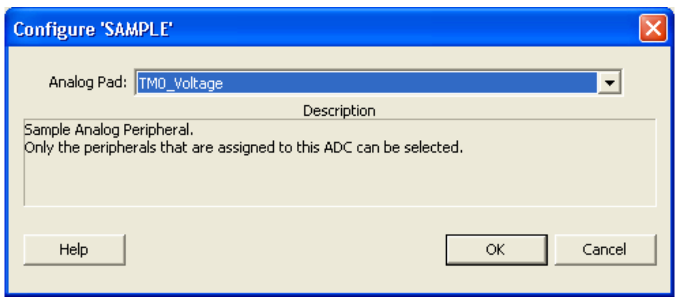

Figure 13 · Configure SAMPLE

11. Click **Insert operating sequence slot** again and select **RESTART SEQUENCE**.

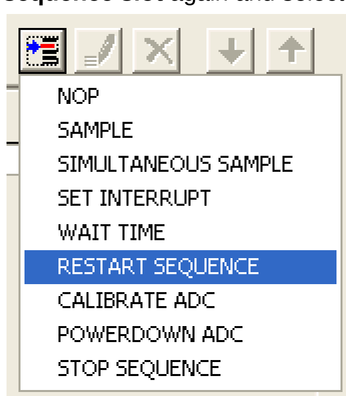

Figure 14 · Select Restart Sequence

12. Click **Calculate Actual Rate**.

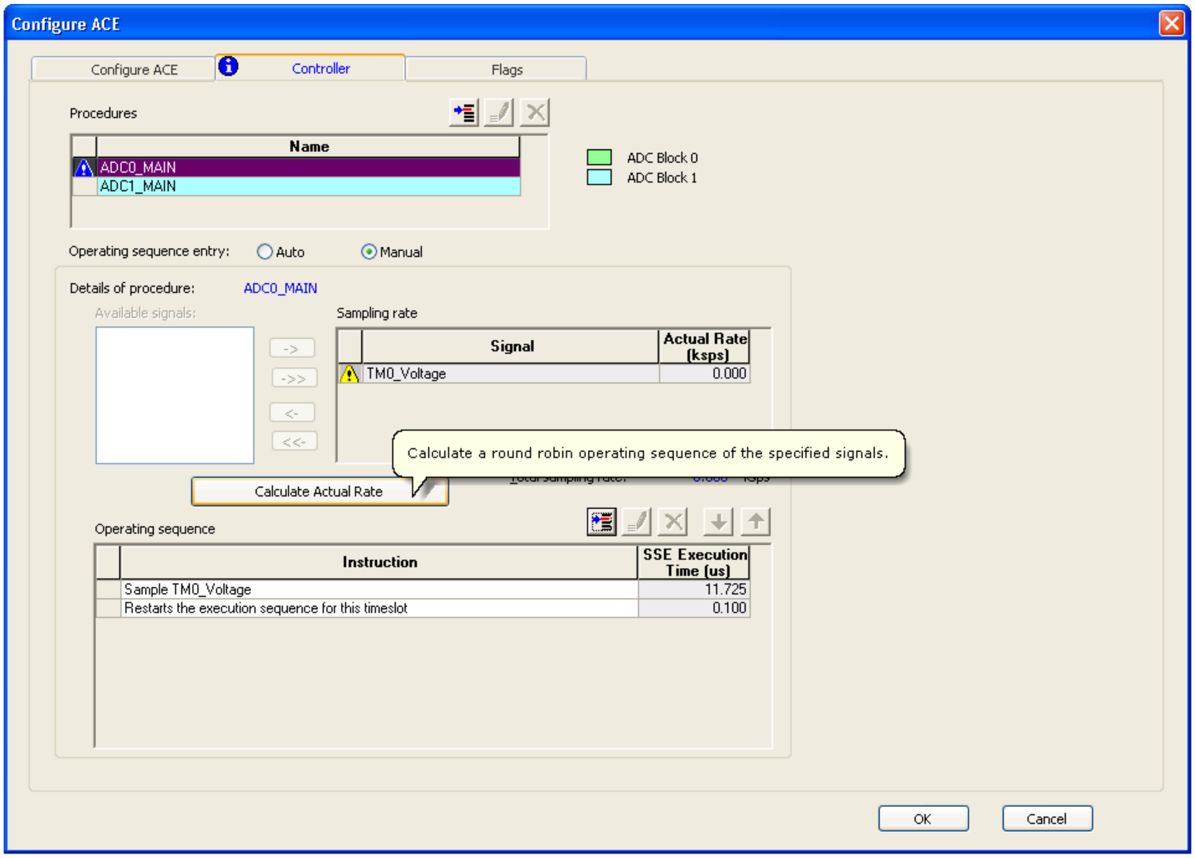

Figure 15 · MSS ACE Configuration: Final Controller Tab

13. The **Controller** tab window is displayed as shown in Figure 16 · :

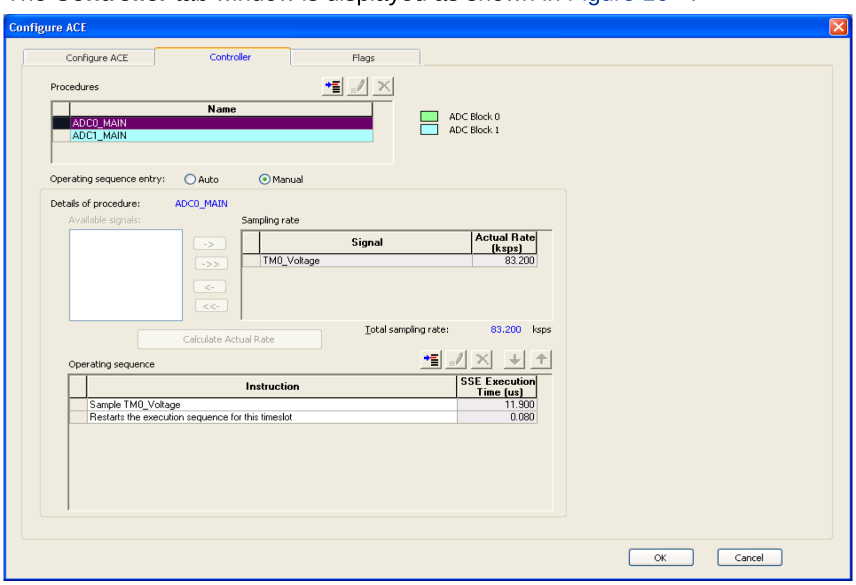

Figure 16 · MSS ACE Configuration: Controller Tab

14. Click the **Flags** tab in the Configure ACE window. This tab lists the flags set from PPE registers.
15. Click the **+** sign to expand the Flag registers group. The PPE_FLAGSn registers contain the user-defined flags.

16. Select **PPE_FLAGS0** (FLAGBANK0). PPE_FLAGS0 contains the 4 threshold flags assigned earlier. These are the flags that were defined when the direct input voltage service was configured. The flag register can be read by the Cortex-M3 processor. The flags also generate interrupts to the Cortex-M3 processor.

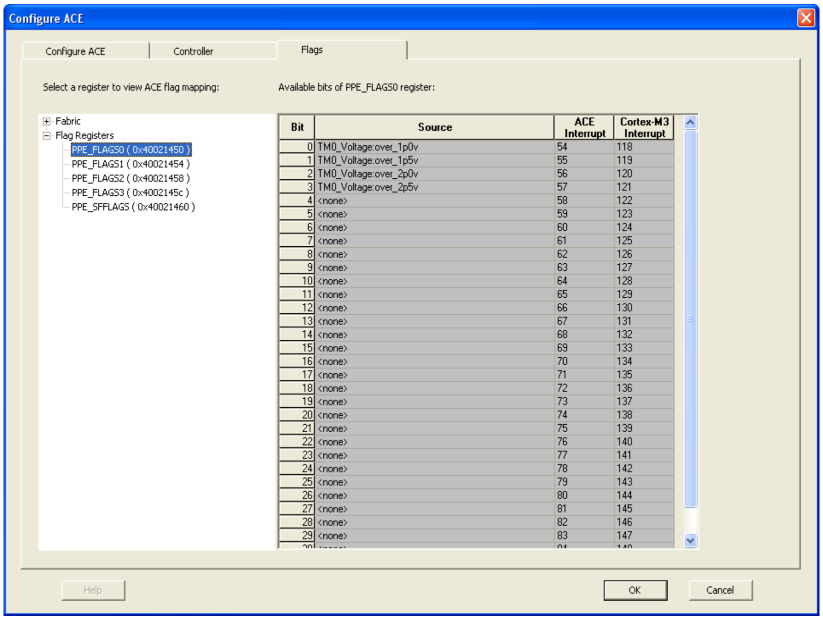

Figure 17 · ACE Flag Mapping - PPE Flag Registers

17. Click **OK** to close the ACE configuration window.

Configuring the GPIO Peripheral

Note: If you are not using the SmartFusion Evaluation Kit Board Revision 2 or later, or using the SmartFusion Development Kit Board, follow Appendix C, Skip Step 3 - Generating the MSS Component and Step 4 - Generating the Program File.

1. Double-click the **GPIO** block in the **MSS component**, configure as shown in Figure 18 · , and click **OK**.

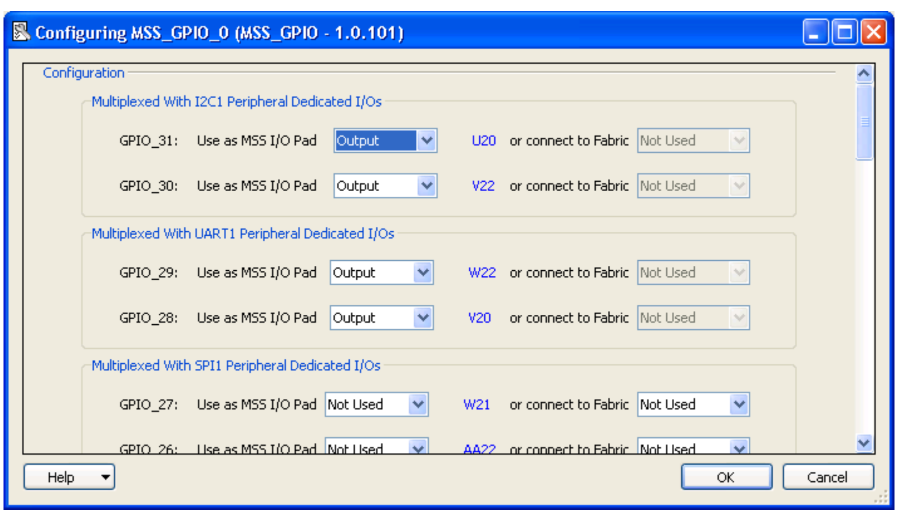

Figure 18 · Configure MSS_GPIO_0

2. This example requires GPIO_31, GPIO_30, GPIO_29, and GPIO_28 to be connected to LED_8 to LED_5 on the SmartFusion Evaluation Kit Board.
3. Click **File > Save** to save the Voltage_Monitor_MSS.

Step 3 - Generating the MSS Component

1. Right-click on **Voltage_Monitor_MSS_0** on the **Voltage_Monitor_MSS** tab and select **Update Instance(s) with Latest Component…** as shown in Figure 19 · .

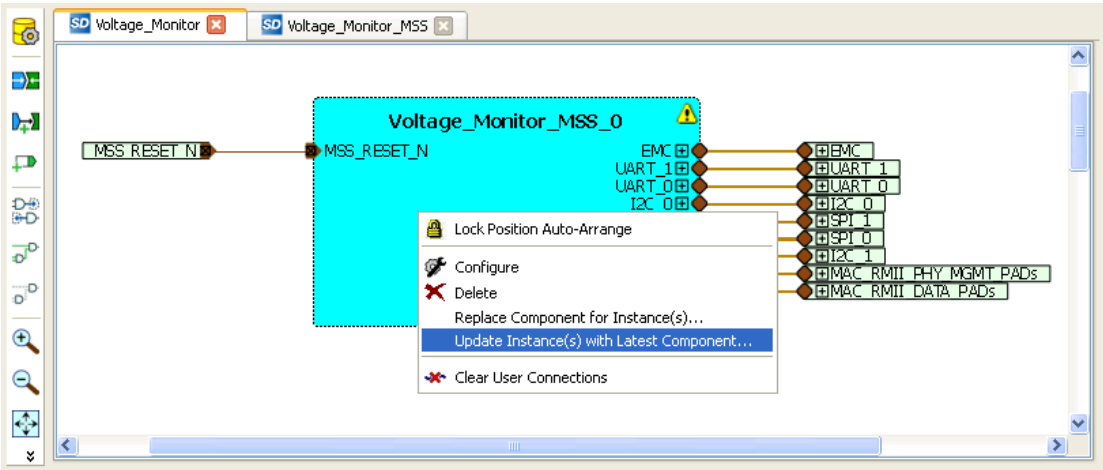

Figure 19 · Updating the MSS

2. Click **Design > Configure Firmware** as shown in Figure 20 · .

Figure 20 · Opening Design_Firmware

On the **DESIGN_FIRMWARE** tab, clear the Generate check boxes for all the peripherals for which you do not need to generate the firmware. Click **Configuration** on the SmartFusion_CMSIS_PAL_0 instance and select **Keil-MDK** as the configuration.

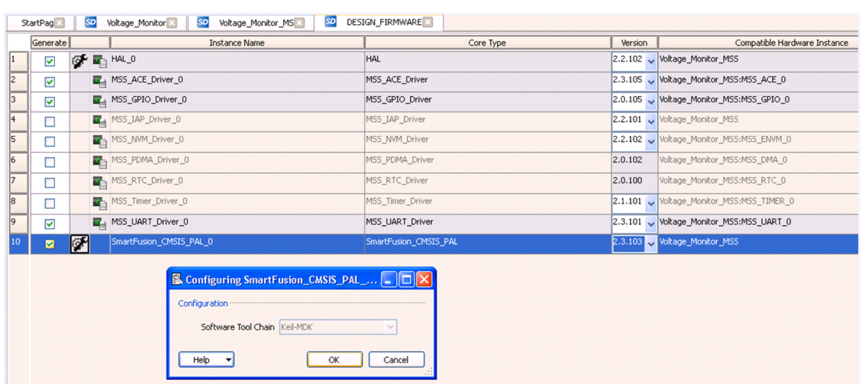

Figure 21 · Configuring SmartFusion_CMSIS_PAL_0

3. Check whether or not you are able to see the latest version of the drivers without any warning or error indicating that firmware is missing from the Vault. If missing, refer to Appendix B – Firmware Catalog Settings.Appendix B – Firmware Catalog Settings
4. Click **File > Save** to save the **Design_Firmware**.

5. **Save** the design and generate the component by clicking **Generate Component** or by selecting **SmartDesign > Generate Component** from menu.

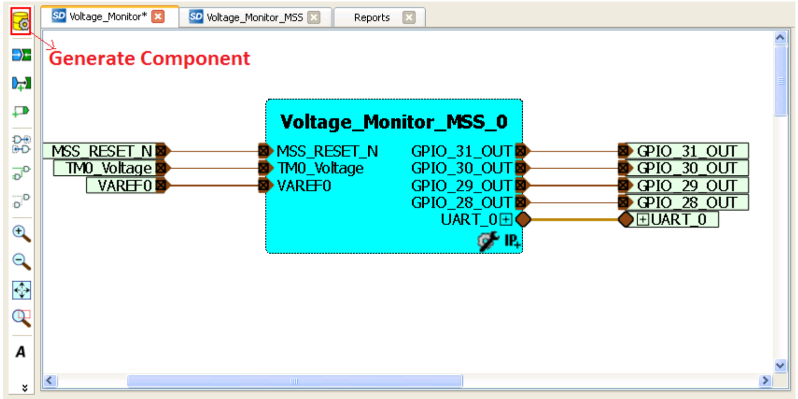

Figure 22 · Generating the MSS Component

6. After successful generation of MSS component the log window displays the message "**Info: 'Voltage_Monitor' was successfully generated. Open datasheet for details**". The datasheet has the Project information like Generated files, used IO's and Memory map etc.
7. Confirm that the Keil folder is created with the subfolders and files shown in Figure 23 · .

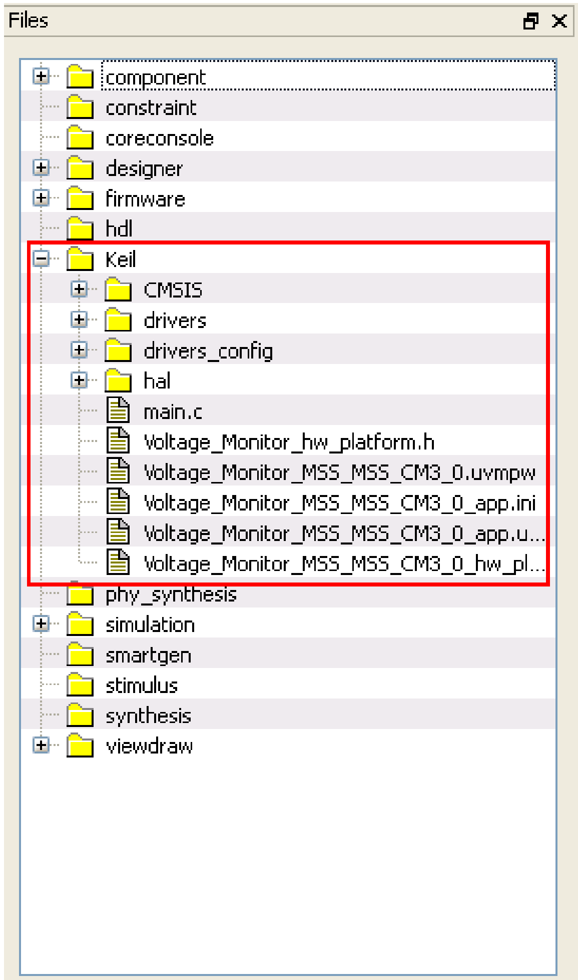

Figure 23 · Files Window

Step 4 - Generating the Program File

Libero SoC provides the push button flow for Generating programming data of the project in a single step. By clicking **Generate Programming Data**, you can complete the synthesis, place and route, verify timing, and generate the programming file. You can also complete the flow by running the synthesis and place and route tools in interactive mode (step-by-step), for more information refer *Libero SoC Quick Start Guide*.

Push-Button Design Flow

1. Click **Generate Programming Data** as shown in **Error! Reference source not found.** to complete the place and route, verify timing, and generate the programming file. This completes the.fdb file generation.

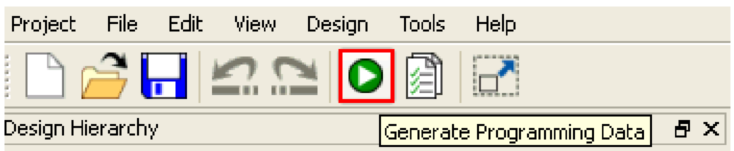

Figure 24 · Build the Project

2. The **Design Flow** window looks as shown in Figure 25 · .

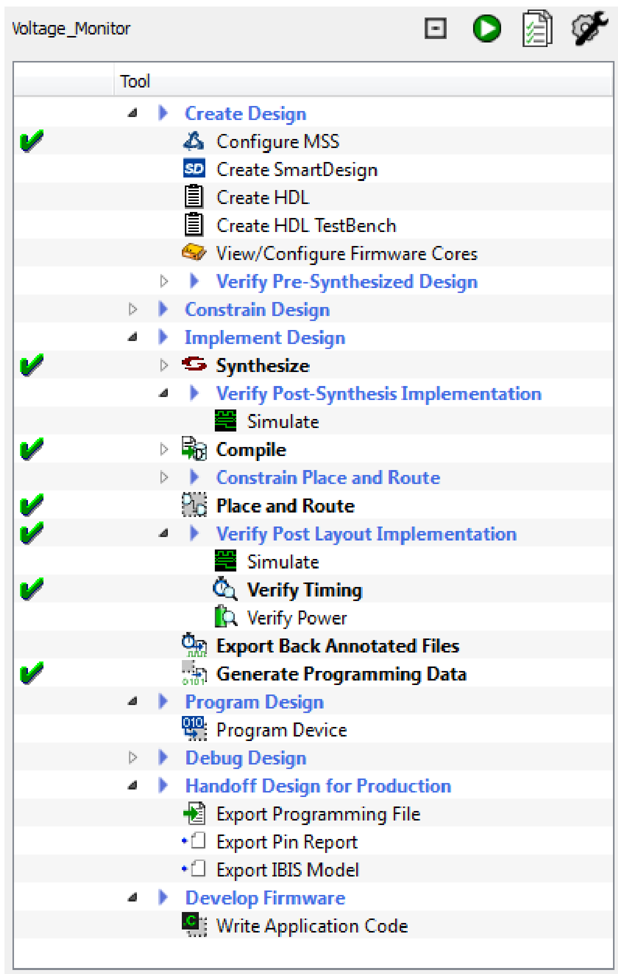

Figure 25 · Design Flow Window After Building the Project

Step 5 - Programming SmartFusion Board Using FlashPro

Before you proceed with programming the device, ensure that the low cost programming stick (LCPS) or FlashPro4 is properly connected to the board. Use the following details to ensure the correct jumper settings. Refer to the *SmartFusion Evaluation Kit User's Guide* and *SmartFusion Development Kit User's Guide* for additional information.

Jumper Settings for SmartFusion Evaluation Kit Board

1. JP10: Short pin 1 and 2 using a jumper
2. JP7: Short pin 1 and 2 using a jumper for LCPS mode
3. J6: Connect pin 1 and 2 using the jumper
4. JP6: Connect pin 2 and 3 using the jumper
5. J13: Connect the USB cable to J13 connector. Install the FlashPro drivers if they are not already installed
6. J14: Connect second USB cable for power
7. JP11, JP12, JP13, and Jp14: Short pin 2 and 3 using a jumper (in A2F - EVAL - REV 2)

Jumper Settings for SmartFusion Development Kit

SW9 must be off (JTAGSEL = H) in order to program the SmartFusion device. SW9 remains in the off position for Libero SoC and SoftConsole programming. Make the jumper settings as shown in Table 2:

Table 2 · Jumper Settings for Development Kit Board

Factory Default	Factory Default	Factory Default
JP1: 1–2	JP12: 1–2	JP21: 1–2
JP2: 1–2	JP13: 1–2	JP22: 2–3
JP4: 1–3; 7–9	JP14: 1–2	JP23: 1–2
JP5: 1–2; 3–4	JP15: 1–2	JP24: 1–2
JP6: 2–3	JP16: 2–3	JP27: 1–2
J7: 2–3; 6–7; 10–11; 14–15	JP17: 2–3	JP28: 1-2
JP7: 1–2	JP18: 1–2	J32: 1–2; 3–4; 5–6
JP8: 3-4; 7-8; 11-12; 15-16	JP19: 2–3	–
JP11: 1-2	JP20: 1–2	–

Programming the Device

1. Double click **Program Device** under **Program Design** in the **Design Flow** window to program the SmartFusion cSoC device.
2. Click **Yes** when it prompts that the I/O and timing constraints are not yet set.

Note: Do not interrupt the programming sequence; it may damage the device or the programmer. If you face any problems, contact Microsemi SoC Products Group Tech Support at soc_tech@microsemi.com.

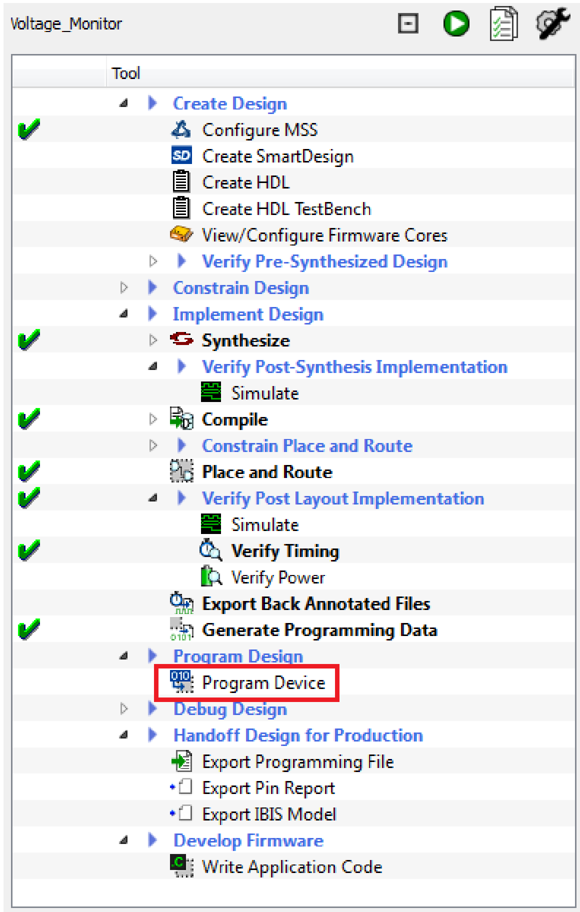

Figure 26 · Design Flow Window

You can also run FlashPro interactively by right clicking on **Program Device** in **Design Flow** window and selecting **Open Interactively**. For more information on FlashPro refer to the *FlashPro User's Guide*.

Step 6 - Building the Software Application through Keil µVision®4 IDE

1. From the Libero SoC, open the Keil project by double clicking on **Write Application Code** under **Develop Firmware** in **Design Flow** window.

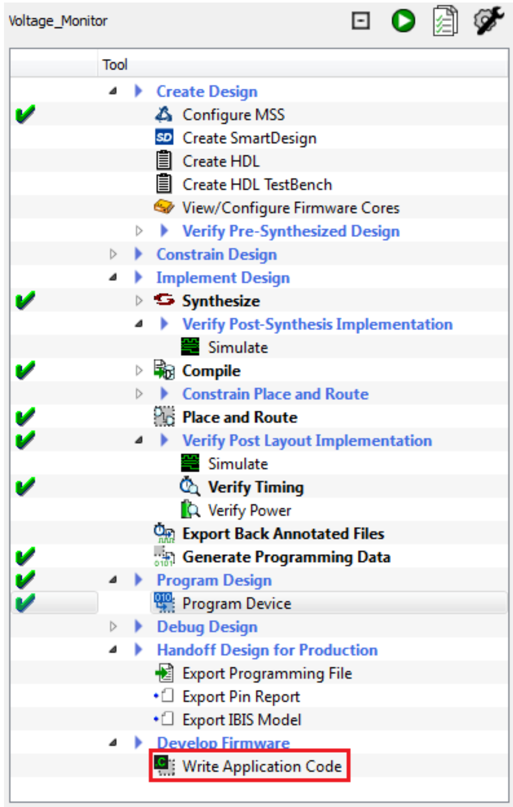

Figure 27 · Invoking µvision4 Project from Libero SoC

2. The µVision perspective looks like Figure 28 ·

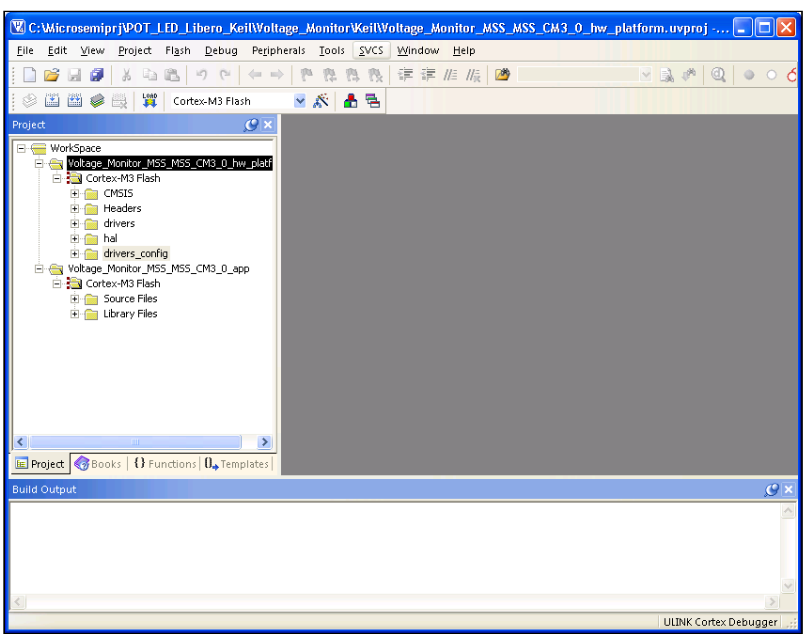

Figure 28 · µVision4 Project

3. Copy the code provided below and paste it in **main.c** file under the Voltage_Monitor_MSS_MSS_CM3_0_app project in the µVision editor and delete the existing code.

```
/*******************************************************************************
 * (c) Copyright 2011 Microsemi Corporation.  All rights reserved.
 *
 *  Sample test program for the SmartFusion ACE.  TM0 is used to monitor the
 *  voltage across the potentiometer.  The UART is used to send the ADC results
 *  to a terminal program.  The hardware configuration has four flags:
 *   - over 1.0v
 *   - over 1.5v
 *   - over 2.0v
 *   - over 2.5v
 *  The flag values are displayed on the SmartFusion eval board LEDs.
 ******************************************************************************/

#include "mss_uart.h"
#include "mss_ace.h"
#include "mss_gpio.h"
#include "ace_handles.h"
#include <stdio.h>

int main()
{
    size_t rx_size;
    uint8_t rx_buff[1];
    const uint8_t greeting[] = "Welcome to Microsemi's SmartFusion Voltage
Monitor\n\r";
    const uint8_t instruction[] = "\n\rPress Any Key\n\r";

int32_t flag_status_2p5v,  flag_status_2p0v, flag_status_1p5v, flag_status_1p0v;
    uint32_t gpio_output;

/*Initialize and Configure GPIO*/
MSS_GPIO_init();
MSS_GPIO_config( MSS_GPIO_31 , MSS_GPIO_OUTPUT_MODE );
MSS_GPIO_config( MSS_GPIO_30 , MSS_GPIO_OUTPUT_MODE );
MSS_GPIO_config( MSS_GPIO_29 , MSS_GPIO_OUTPUT_MODE );
MSS_GPIO_config( MSS_GPIO_28 , MSS_GPIO_OUTPUT_MODE );

/*Initialize UART_0*/
MSS_UART_init
(
&g_mss_uart0,
MSS_UART_57600_BAUD,
MSS_UART_DATA_8_BITS | MSS_UART_NO_PARITY | MSS_UART_ONE_STOP_BIT
);

/*Initialize ACE*/
ACE_init( );

MSS_UART_polled_tx( &g_mss_uart0, greeting, sizeof(greeting) );
MSS_UART_polled_tx( &g_mss_uart0, instruction, sizeof(instruction) );
```

```
for (;;)
{
rx_size = MSS_UART_get_rx( &g_mss_uart0, rx_buff, sizeof(rx_buff));
if (rx_size > 0)
{
uint8_t display_buffer[32];
uint16_t adc_result;
int32_t adc_value_mv;

adc_result   = ACE_get_ppe_sample( TM0_Voltage );
adc_value_mv = ACE_convert_to_mV( TM0_Voltage, adc_result );

if ( adc_value_mv < 0 )
{
snprintf( display_buffer, sizeof(display_buffer),
"-%d.%d V\n\r", -adc_value_mv / 1000, -adc_value_mv % 1000);
}
else
{
snprintf( display_buffer, sizeof(display_buffer),
"%d.%d V\n\r", adc_value_mv / 1000, adc_value_mv % 1000);
}
MSS_UART_polled_tx_string( &g_mss_uart0, display_buffer );

           /* Checking the status of Voltage flags */
           flag_status_2p5v = ACE_get_flag_status(TM0_Voltage_over_2p5v);
           flag_status_2p0v = ACE_get_flag_status(TM0_Voltage_over_2p0v);
           flag_status_1p5v = ACE_get_flag_status(TM0_Voltage_over_1p5v);
           flag_status_1p0v = ACE_get_flag_status(TM0_Voltage_over_1p0v);

           /* Voltage flags are displayed on the LEDs through GPIO */

if ( flag_status_2p5v == FLAG_ASSERTED )
    gpio_output = ~(
                   MSS_GPIO_28_MASK |
                   MSS_GPIO_29_MASK |
                   MSS_GPIO_30_MASK |
                   MSS_GPIO_31_MASK );
else
if ( flag_status_2p0v == FLAG_ASSERTED )
    gpio_output = ~(
                   MSS_GPIO_28_MASK |
                   MSS_GPIO_29_MASK |
                   MSS_GPIO_30_MASK );
else
if ( flag_status_1p5v == FLAG_ASSERTED )
    gpio_output = ~(
                   MSS_GPIO_28_MASK |
                   MSS_GPIO_29_MASK );
else
if ( flag_status_1p0v == FLAG_ASSERTED )
```

```
    gpio_output = ~(
                   MSS_GPIO_28_MASK );
else
    gpio_output = (
                   MSS_GPIO_28_MASK |
                   MSS_GPIO_29_MASK |
                   MSS_GPIO_30_MASK |
                   MSS_GPIO_31_MASK );

MSS_GPIO_set_outputs( gpio_output );

    MSS_UART_polled_tx( &g_mss_uart0, instruction, sizeof(instruction) );
}
}
return 0;
}
/*************************************************************************/
```

4. Right-click on **Cortex – M3 Flash** under **Voltage_Monitor_MSS_MSS_CM3_hw_platform** and select **Build**.

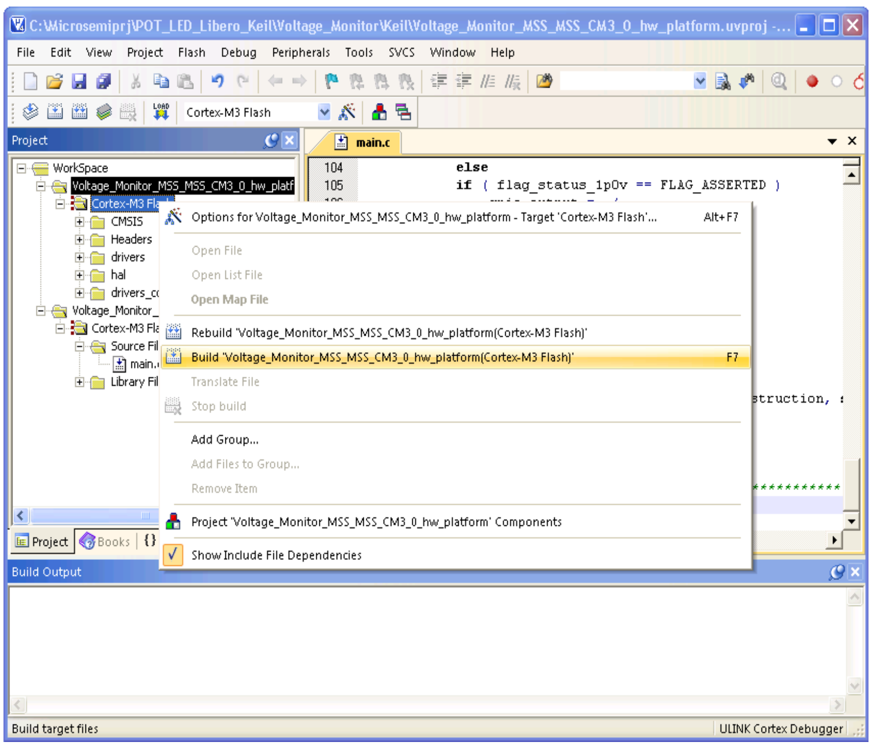

Figure 29 · Building the Hardware Platform

5. Right-click on **Voltage_Monitor_MSS_MSS_CM3_app** and select **Set as Active Project**.

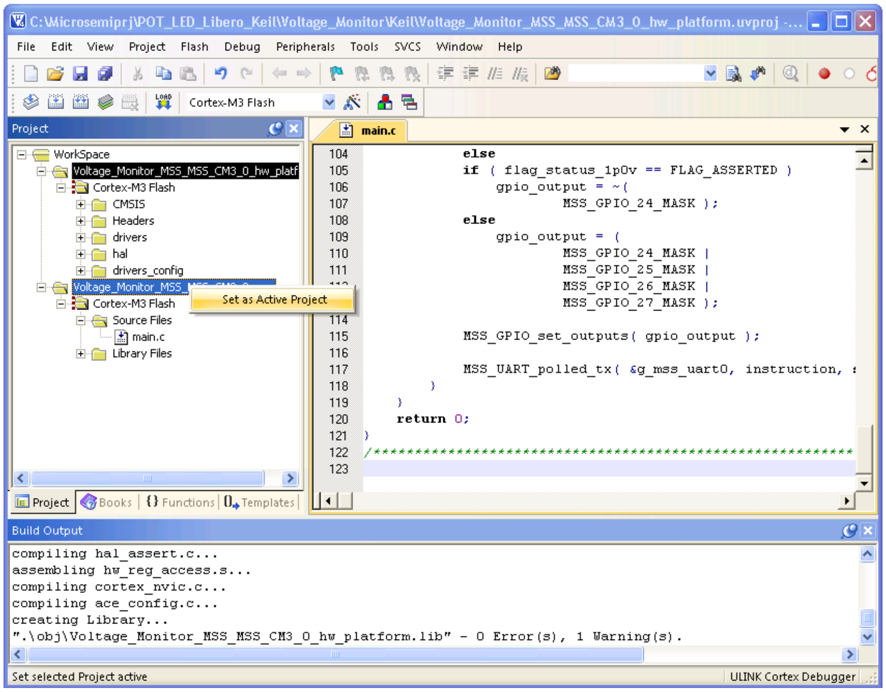

Figure 30 · Setting the Application Project as Active

6. Right-click on **Cortex – M3 Flash** under **Voltage_Monitor_MSS_MSS_CM3_app** and click **Options for project**.

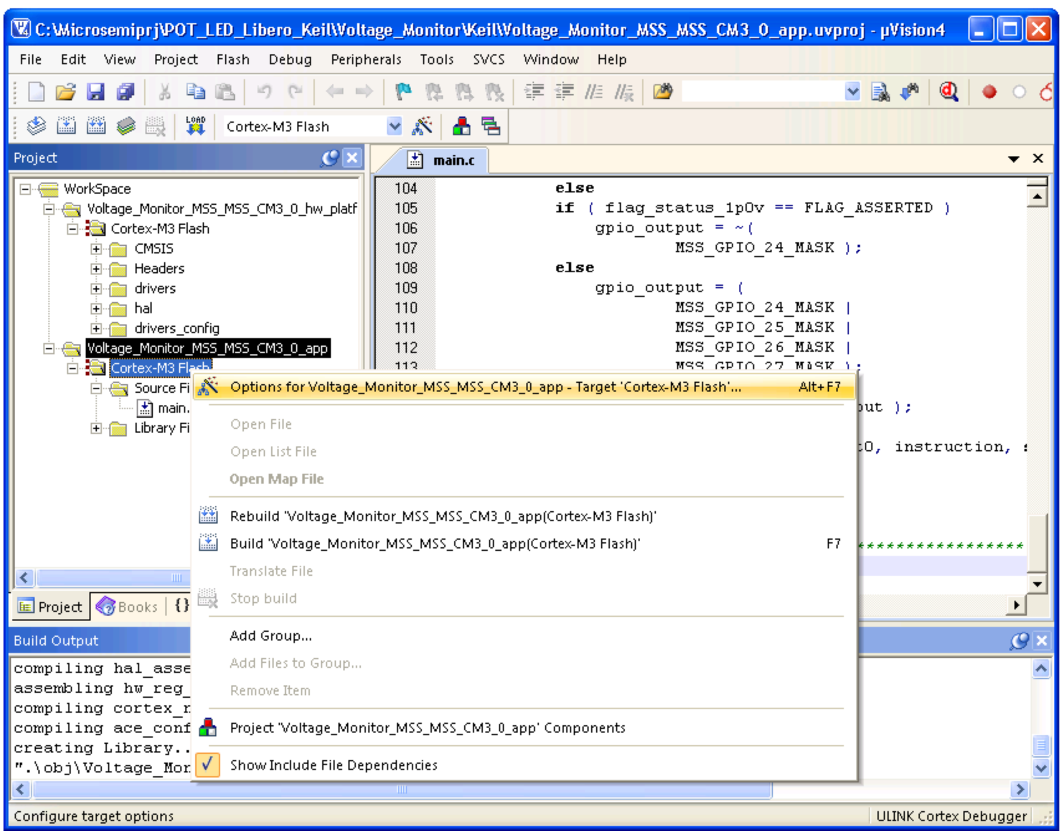

Figure 31 · Target Options

7. Go to the **Target** tab and change XTAL (MHZ) clock to 80 as shown in Figure 32 · .

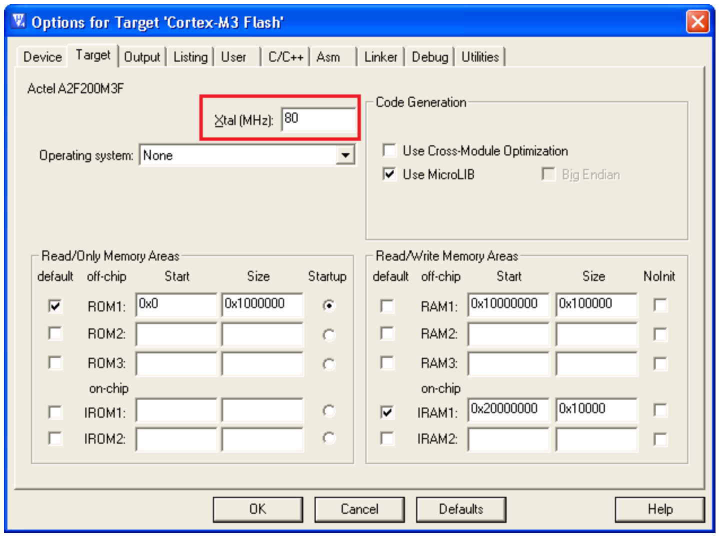

Figure 32 · Target Option for Target Cortex-M3 Flash

8. Close the options dialog by clicking **OK**.
9. Select **rebuild all target files** from the **Project** drop-down menu. This action compiles all of the source files and links the object files into an AXF file for debug. Correct any syntax errors and re-build if necessary.

   The following messages are displayed in the console:

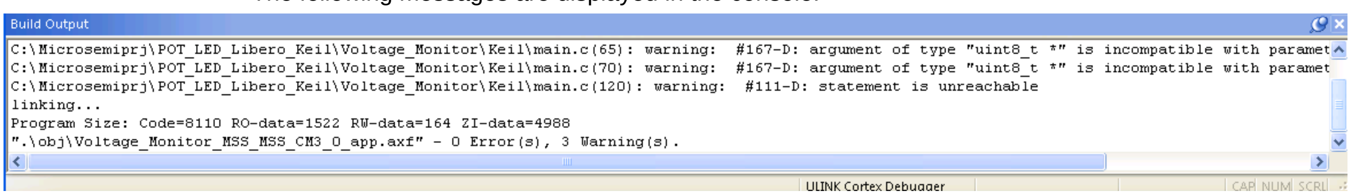

Figure 33 · Build Output

10. Open the **Target Options** by right-clicking **Cortex – M3 Flash** under Voltage_Monitor_MSS_MSS_CM3_app and select Options.
11. Click the **Utilities** tab on the **Options for Target** dialog box and clear the “**Update Target before Debugging**” checkbox.

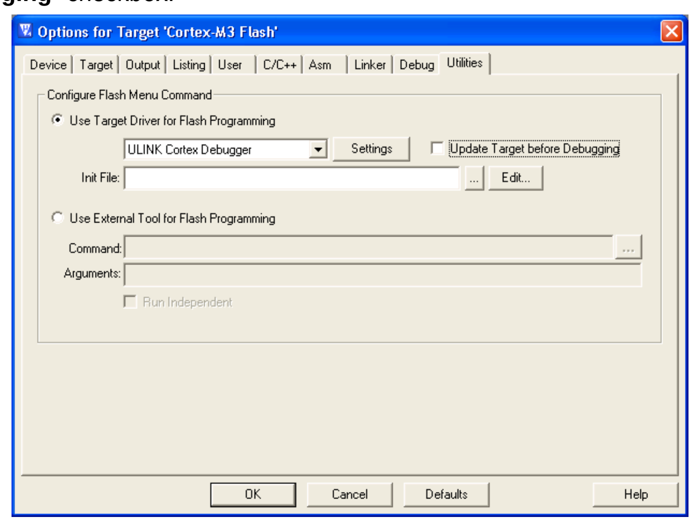

Figure 34 · Clear Update Target Before Debugging

12. Click **Settings**.

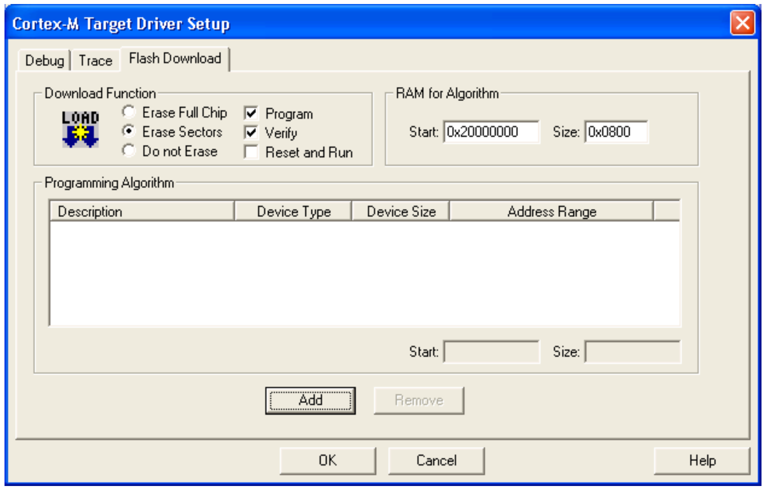

Figure 35 · Cortex – M Target Driver Setup

13. Click **Add**. The Add Flash Programming algorithm is displayed as shown in Figure 36 · .

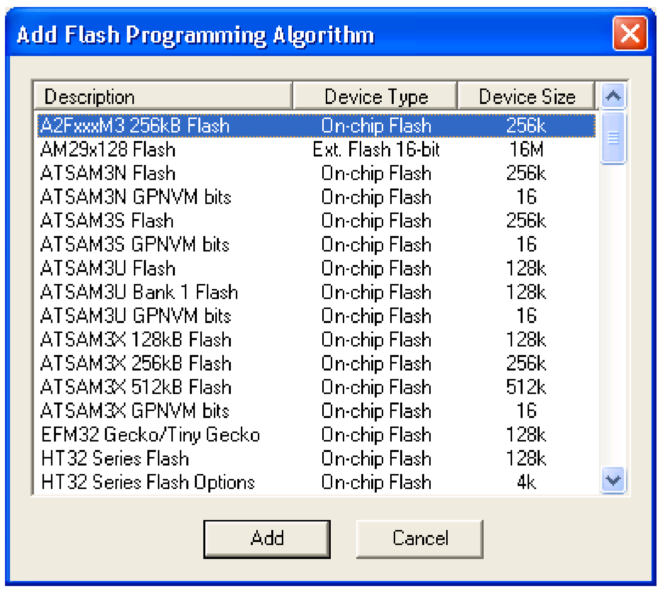

Figure 36 · Add Flash Programming Algorithm

14. After adding the required algorithm, click **OK** in the Target Driver Setup window.
15. Click **OK** to close the Options for Target dialog box (Figure 34 ·).

Step 7 - Configuring the Serial Terminal Emulation Program

Prior to running the application program, you need to configure the terminal emulator program (HyperTerminal, included with Windows®) on your PC. Perform the following steps to use the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board:

1. Connect a second mini USB cable between the USB connector on the SmartFusion Evaluation Kit Board (or the SmartFusion Development Kit Board) and a USB port of your PC. If Windows prompts you to connect to Windows Update, select No, not at this time and click Next.
2. If the Silicon Labs CP210x USB to UART Bridge drivers are automatically detected (this can be verified in Device Manager), as shown in Figure 37 · . Proceed to next step, otherwise follow
3. Step 8 - Installing Drivers for the USB to RS232 Bridge.

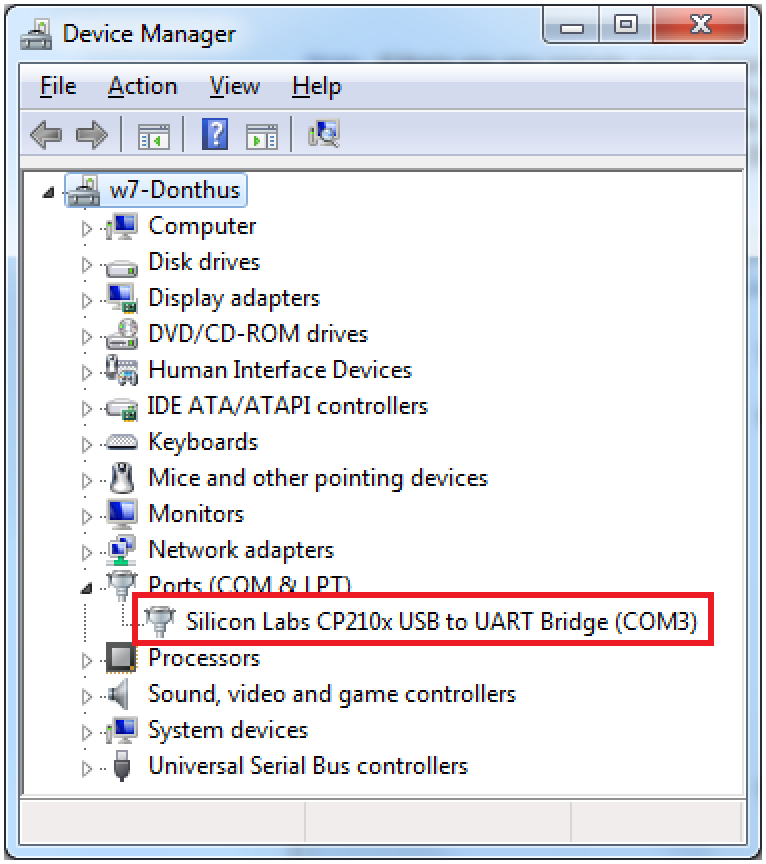

Figure 37 · Device Manager Listing Silicon Labs CP210x USB to UART Bridge Drivers

4. From the Windows **Start** menu, select **Programs > Accessories > Communications > HyperTerminal.** This opens HyperTerminal. If your PC does not have HyperTerminal, use any free serial terminal emulation program like PuTTY or Tera Term. Refer to the *Configuring Serial Terminal Emulation Programs* tutorial for configuring the HyperTerminal, Tera Term, and PuTTY.
5. Enter **Hyperterminal** in the **Name** field in the **Connection Description** dialog box and click **OK**.

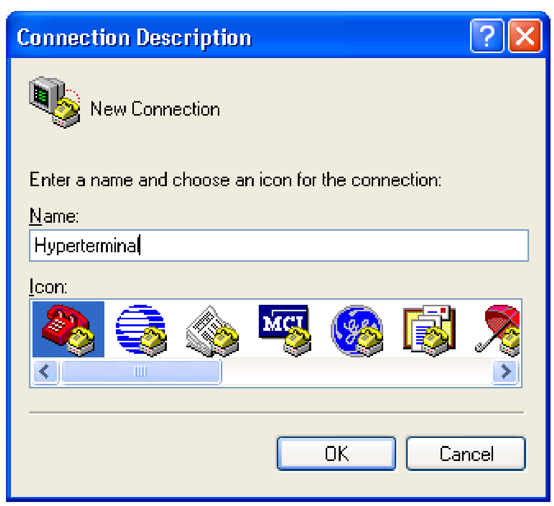

Figure 38 · New Connection

6. Select the appropriate COM port (to which USB-Rs232 drivers are pointed) from the **Connect using** drop-down list and click **OK**.

Figure 39 · Selecting the COM Port

7. Set the following in the **COM Properties** window and click **OK**:
   - Bits per second: 57600
   - Data bits: 8
   - Parity: None
   - Stop Bits: 1
   - Flow control: None

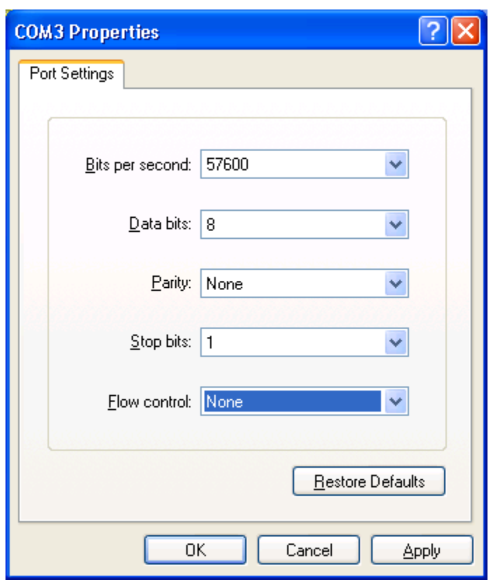

Figure 40 · Setting the COM Properties

8. Click **OK** to close the Hyperterminal Properties dialog box.

   Next time you can directly open HyperTerminal (without configuring) by selecting **Programs > Accessories > Communications > HyperTerminal > Hyperterminal**.

Step 8 - Installing Drivers for the USB to RS232 Bridge

Note: To install the USB-RS232 drivers, you should have administrative privileges for your PC.

Use the following steps to install drivers for the USB to RS232 Bridge:

1. Download the USB to RS232 bridge drivers from www.microsemi.com/soc/documents/CP2102_driver.zip.
2. Unzip the CP2102_driver.zip file.
3. Double-click (Run) the CP210x_VCP_Win_XP_S2K3_Vista_7.exe file.
4. Accept the default installation location and click Install.
5. Click Continue Anyway if prompted.
6. When the installation is complete, click OK. The Ports (COM & LPT) section of the Device Manager lists Silicon Labs CP210x USB to UART Bridge under the Ports section of Device Manager.

Step 9 - Connecting the ULINK-ME to the Board and PC

This section describes the connection between the SmartFusion Evaluation Kit Board or SmartFusion Development Kit Board, ULINK-ME, and PC. Make sure to use the appropriate setting for the board that is being used.

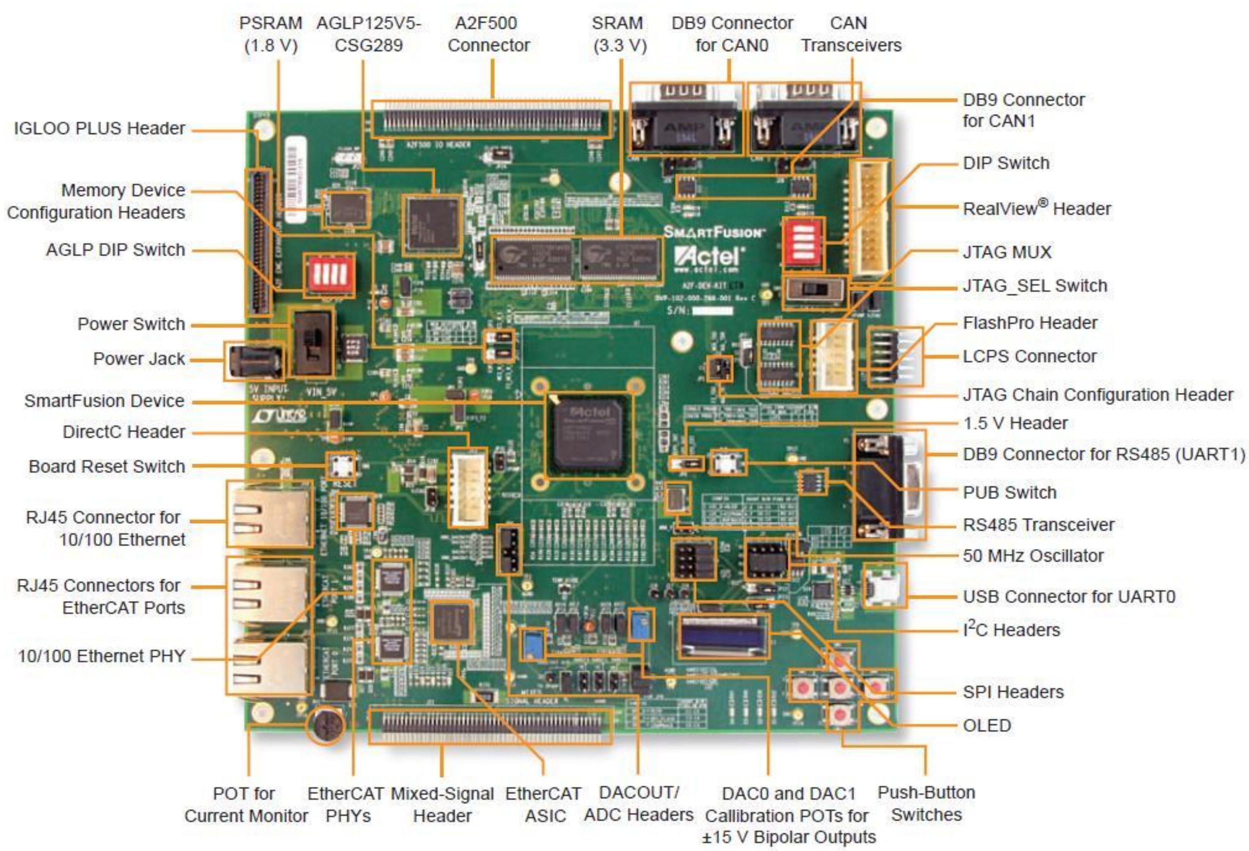

Figure 41 · SmartFusion Development Kit Board

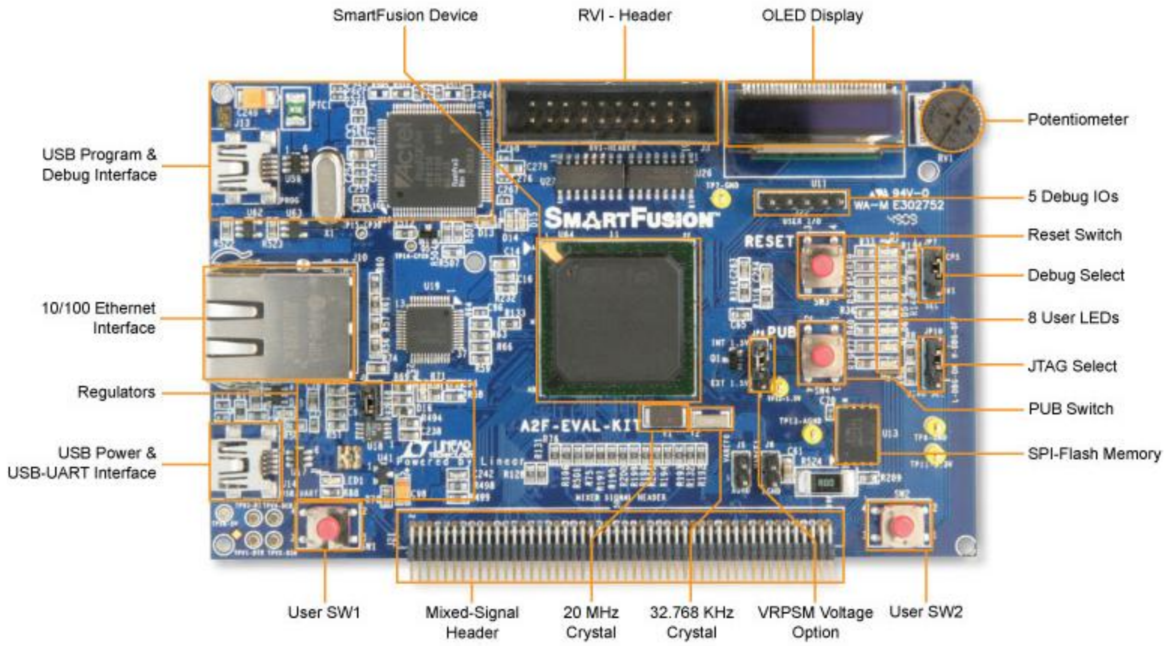

Figure 42 · SmartFusion Evaluation Kit Board

Figure 43 · ULINK-ME Debugger

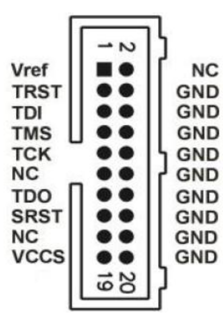

Figure 44 · ULINK-ME Header

Using the SmartFusion Development Kit Board

1. Connect a USB A-Mini B cable between your PC and the SmartFusion Development Kit Board (J9). This is used to display the HyperTerminal communications.
2. Verify that the ULINK-ME debugger is connected to the SmartFusion Development Kit Board (J3) and also to your PC via USB A-Mini B cable. The ULINK-ME adapter has one LED that indicates connection status in the following ways:
   - Slow flashing indicates that ULINK-ME is ready to communicate with the debugger.
   - Fast flashing indicates that the target board is executing the program under debugger control.

- ON during debugging indicates that the debugger has halted the target board.
- ON during download indicates that target download/verification is in progress.

3. Change the switch SW9 to ON position.
4. Connect pin 2 and 3 on the jumper JP7 on the SmartFusion Development Kit Board.

Using the SmartFusion Evaluation Kit Board

1. Connect a USB A-Mini B cable between your PC and the SmartFusion Evaluation Kit Board (RVI-Header). This is used to display the HyperTerminal communications.
2. Verify that the ULINK-ME debugger is connected to the SmartFusion Evaluation Kit Board (J3) and also to your PC via USB A-Mini B cable. The ULINK-ME adapter has one LED that indicates connection status in the following ways:
   - Slow flashing indicates that ULINK-ME is ready to communicate with the debugger.
   - Fast flashing indicates that the target board is executing the program under debugger control.
   - ON during debugging indicates that the debugger has halted the target board.
   - ON during download indicates that target download/verification is in progress.
3. Connect pin 2 and 3 on the jumper JP10 on the SmartFusion Evaluation Kit Board.
4. Connect pin 1 and 2 on the jumper JP6 on the SmartFusion Evaluation Kit Board.
5. Connect pin 2 and 3 on the jumper JP7 for Keil debugging mode.

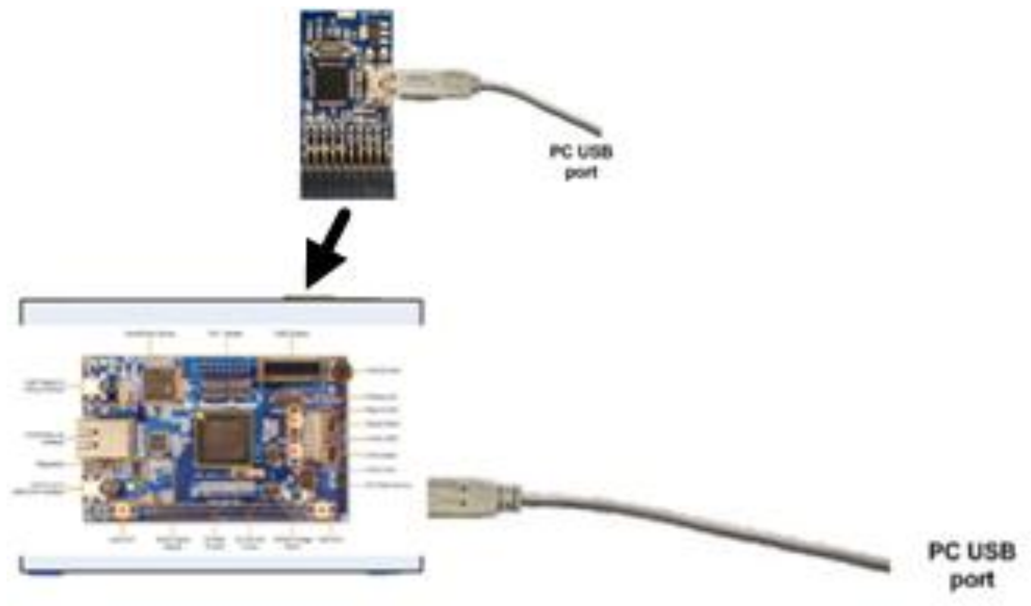

Figure 45 · SmartFusion Evaluation Kit Board, ULINK Debugger, and USB Connections

Step 10 - Debugging the Application Project Through µVision4

1. Download the processor code to the SmartFusion eSRAM and execute it via the debug hardware by selecting **Start/Stop Debug Session** from the **Debug** drop-down menu. The code will automatically ‘run to main’ and then stop.

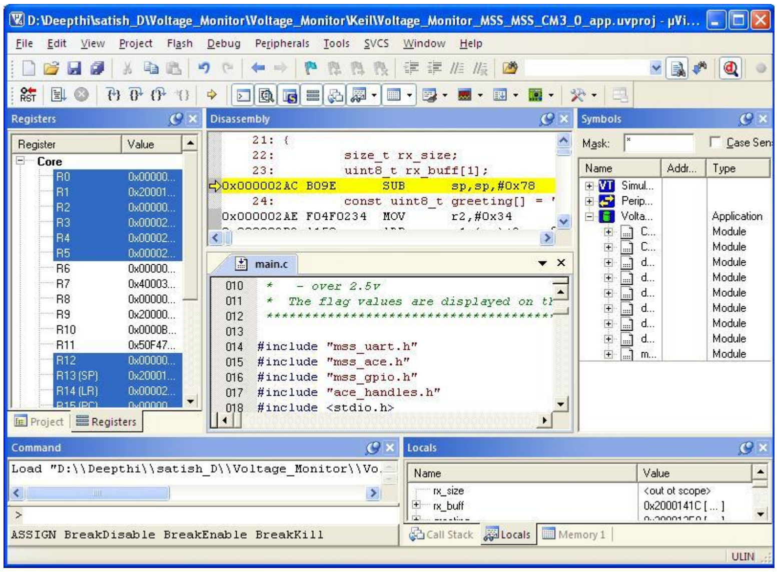

Figure 46 · Debug Session

The Registers Window is displayed by default in the Project Window. To modify a register value, select the value and press F2, or double-click the value.

2. Click the **Run** icon from the debug toolbar, or select **Run** from the **Debug** drop-down menu, it should display the following message in HyperTerminal:

```
Welcome to Microsemi’s SmartFusion Voltage Monitor
Press any Key
```

3. Move your cursor into the HyperTerminal window and press any key on your PC keyboard. The voltage measurement is displayed. In addition, also observe the LEDs on the SmartFusion Evaluation Kit Board. They are illuminated when one of the voltage monitor flags is asserted.
4. Adjust the POT while pressing a key and observe that the voltage measurement is continuously updated

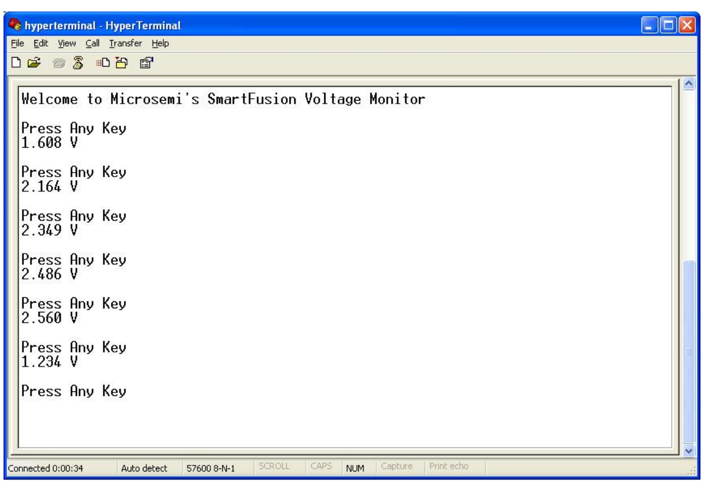

Figure 47 · HyperTerminal Window

Observe the state of the LEDs as the POT is adjusted. Confirm that the flags work as specified in the ACE configurator. To stop the program, click on the ‘Stop’ icon ( ) on the debug toolbar or **Stop** from **Debug** menu.

Appendix A – Libero SoC Catalog Settings

The following steps show how to configure your vault location and set up the repositories in Libero SoC.

1. On the **Catalog** window, click **Options**.

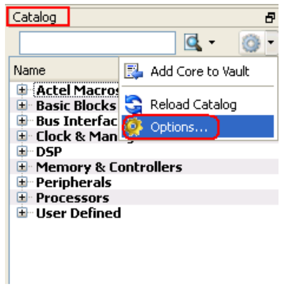

Figure 48 · Catalog – Options

2. The **Options** window is displayed. Click **Repositories** under **Vault/Repositories Settings** add the following in the address field:
   - www.actel-ip.com/repositories/SgCore
   - www.actel-ip.com/repositories/DirectCore
   - www.actel-ip.com/repositories/Firmware

   Note: Click **Add** after entering each path.

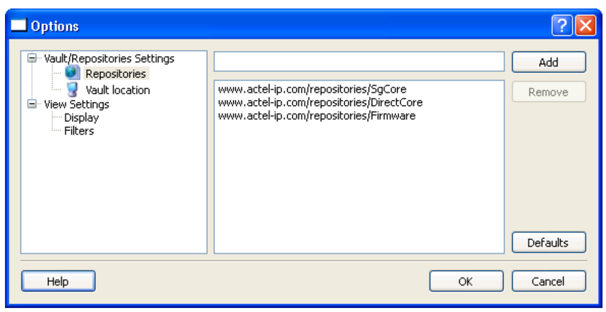

Figure 49 · Setting Repositories

3. Click on the **Vault location** under **Vault/Repositories Settings** in the **Options** window. Browse to a location on your PC to set the vault location where the IPs can be downloaded from the repositories.

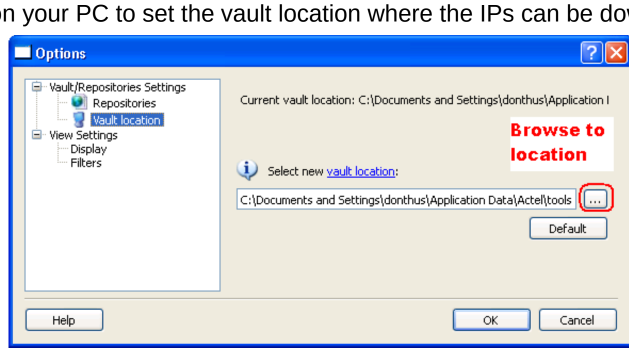

Figure 50 · Setting the Vault Location

4. Click **OK.**

Appendix B – Firmware Catalog Settings

1. Open the **<Libero Installation directory>\Designer\bin\catalog.exe**.
2. Select **Tools > Vault/Repositories Settings**, from the **Firmware Catalog** window.

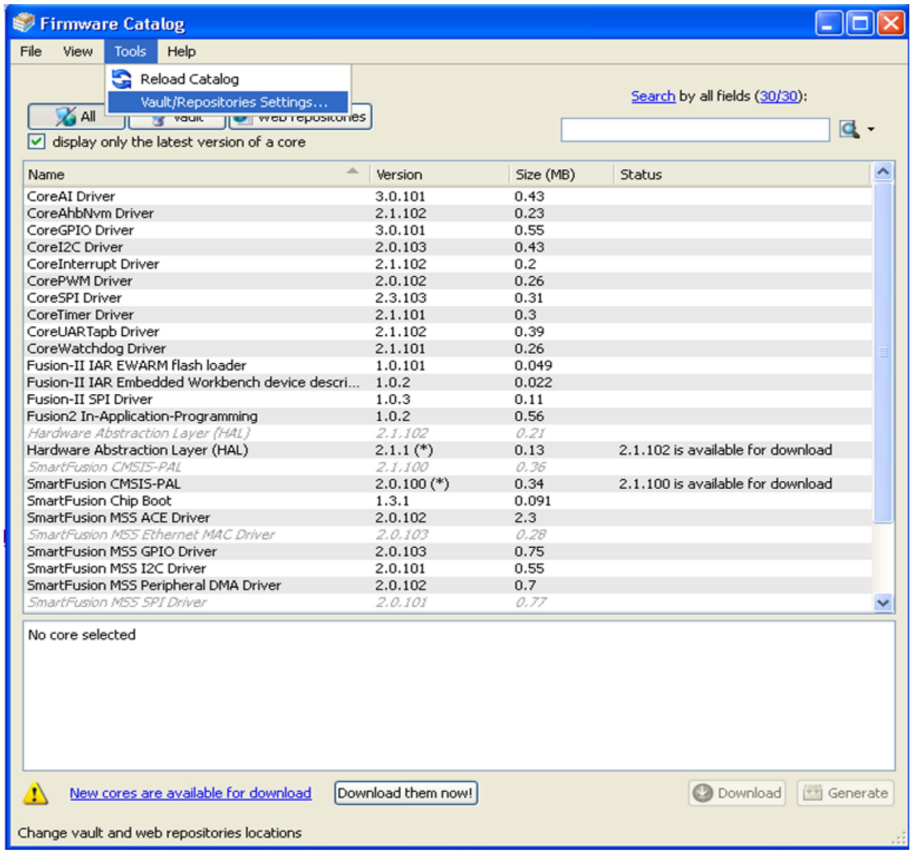

Figure 51 · Firmware Catalog Settings

3. Select **Repositories** under **Vault/Repositories Settings** in the **Options** dialog box.
4. Confirm that the following repositories are displayed (add them if needed):
   - www.actel-ip.com/repositories/SgCore
   - www.actel-ip.com/repositories/DirectCore
   - www.actel-ip.com/repositories/Firmware
5. Add the above mentioned paths in the address field if required by selecting the repository and clicking **Add**.

   If new cores are available for download, click **Download them now!** to download the new cores to the vault.

Appendix – C

Configuring the GPIO Peripheral

1. Double-click the **GPIO** block in the **MSS component**, configure as shown in Figure 52 · , and click **OK**.

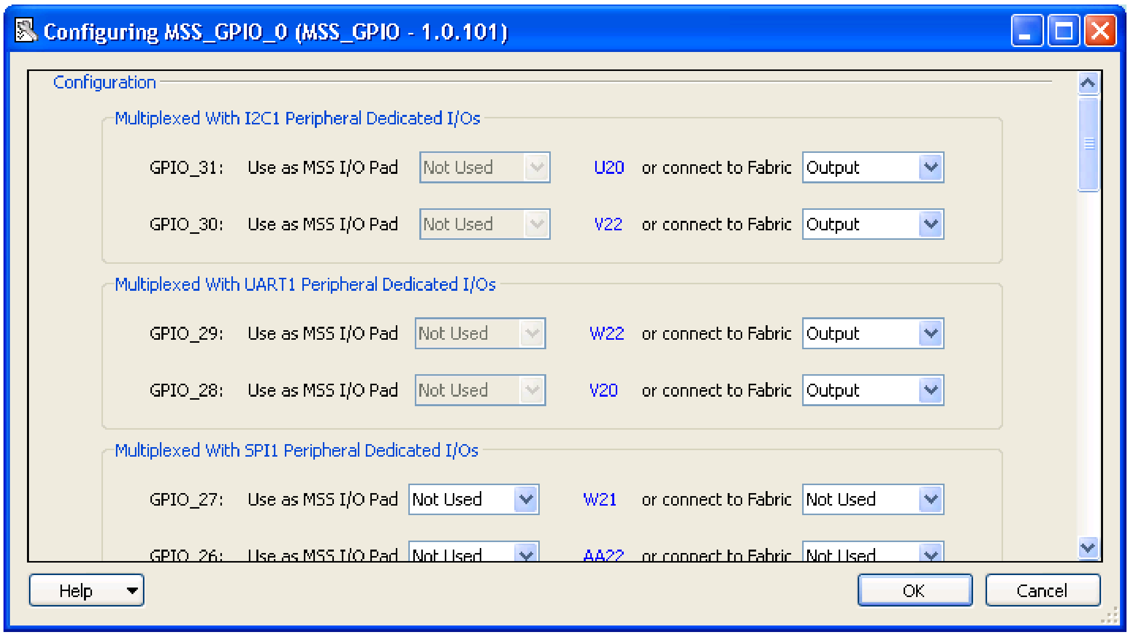

Figure 52 · Configure MSS_GPIO_0

This example requires GPIO_31, GPIO_30, GPIO_29, and GPIO_28 to be connected to LED_4 to LED_1 on the SmartFusion Evaluation Kit Board (D4 to D1 on the SmartFusion Development Kit Board). These signals will be routed through the fabric to I/O pins H17, C19, B20, and B19, respectively.

2. Click **File > Save** to save the Voltage_Monitor_MSS.

Generating the MSS Component

1. Right-click on **Voltage_Monitor_MSS_0** component on the **Voltage_Monitor** tab and select **Update Instance(s) with Latest Component…** as shown in Figure 53 · .

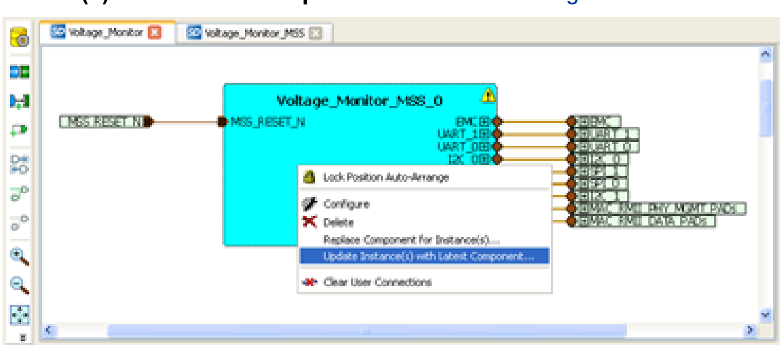

Figure 53 · Updating the MSS

2. Promote the M2F_GPIO [31:28] pins to top level.

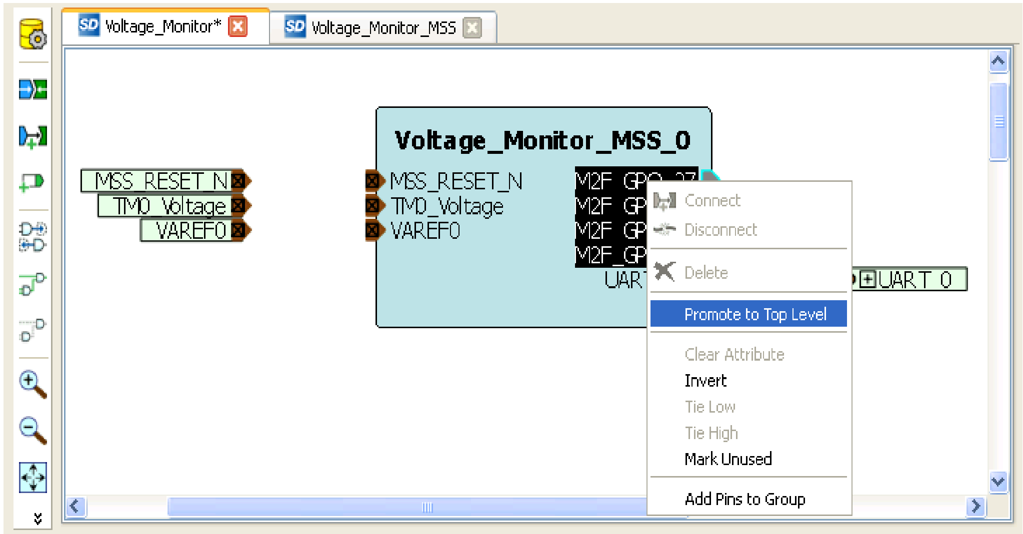

Figure 54 · GPIO Pins Promoted to Top Level

3. Click **Design > Configure Firmware** as shown in Figure 55 · .

Figure 55 · Opening Design_Firmware

4. On the **DESIGN_FIRMWARE** tab, clear the Generate check boxes for all the peripherals for which you do not need to generate the firmware. Click **Configuration** on the SmartFusion_CMSIS_PAL_0 instance and select **SoftConsole** as the configuration.

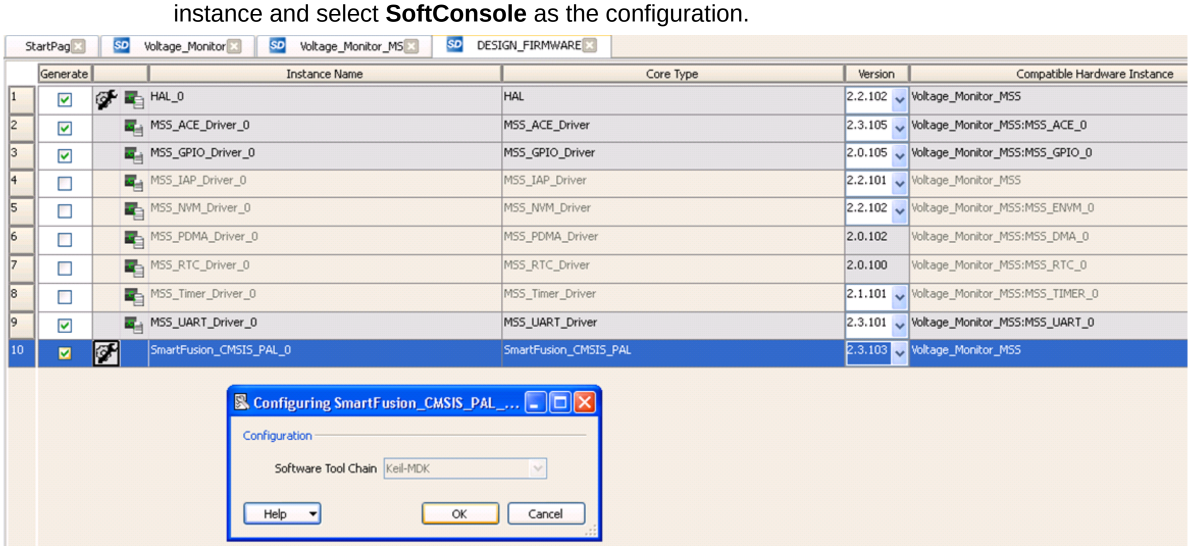

Figure 56 · Firmware Configuration Settings – CMSIS Peripheral

5. Check whether or not you are able to see the latest version of the drivers without any warning or error indicating that firmware is missing from the Vault. If missing, refer to Appendix B – Firmware Catalog Settings.

6. Click **File > Save** to Save the Design_Firmware.

7. **Save** the design and generate the component by clicking **Generate Component** or by selecting **SmartDesign > Generate Component**.

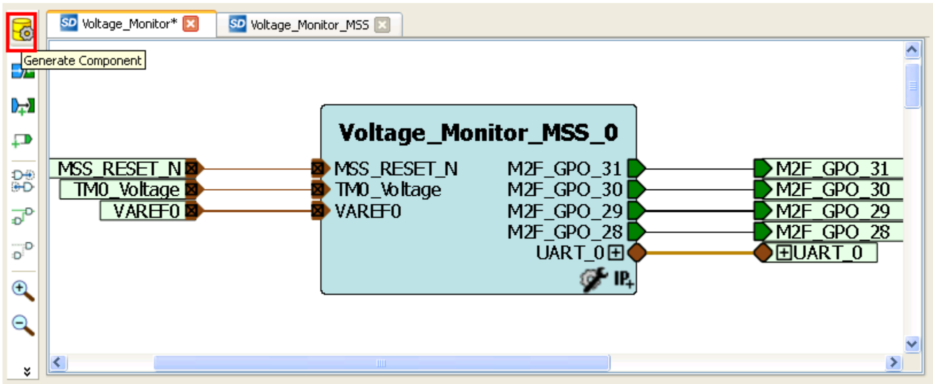

Figure 57 · Generating the MSS Component

8. After successful generation of project, the log window displays the message “**Info: 'Voltage_Monitor' was successfully generated. <u>Open datasheet for details</u>**”. The datasheet has the Project information such as the generated files, used IO’s, memory map, etc.
9. Confirm that the Keil folder is created with the folders and files as shown in Figure 58 · .

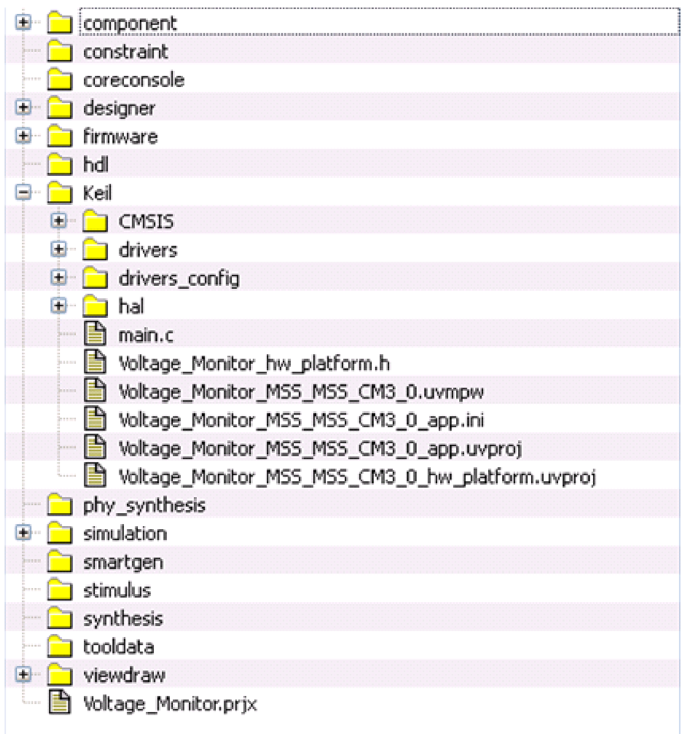

Figure 58 · Files Window

Generating the Program File

Libero SoC provides the push button flow for generating programming data of the project in a single step. By clicking the **Generate Programming Data**, you can complete the synthesis, place and route, verify timing, and generate the programming file. You can also complete the flow by running the synthesis and place and route tools in interactive mode (step-by-step), for more details; refer to the *Libero SoC Quick Start Guide*.

Push-button Design Flow

1. Click **Edit I/O Attributes** under **Constrain Place and Route** in the **Design Flow** window.

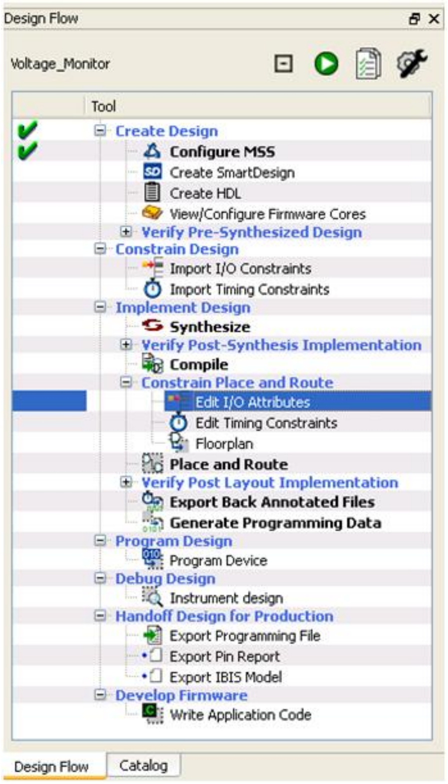

Figure 59 · Configuring SmartFusion_CMSIS_PAL_0

2. Make the following pin assignments in **MultiView Navigator** window as shown in Figure 60 · :
   - GPO_28 to B19
   - GPO_29 to B20
   - GPO_30 to C19
   - GPO_31 to H17

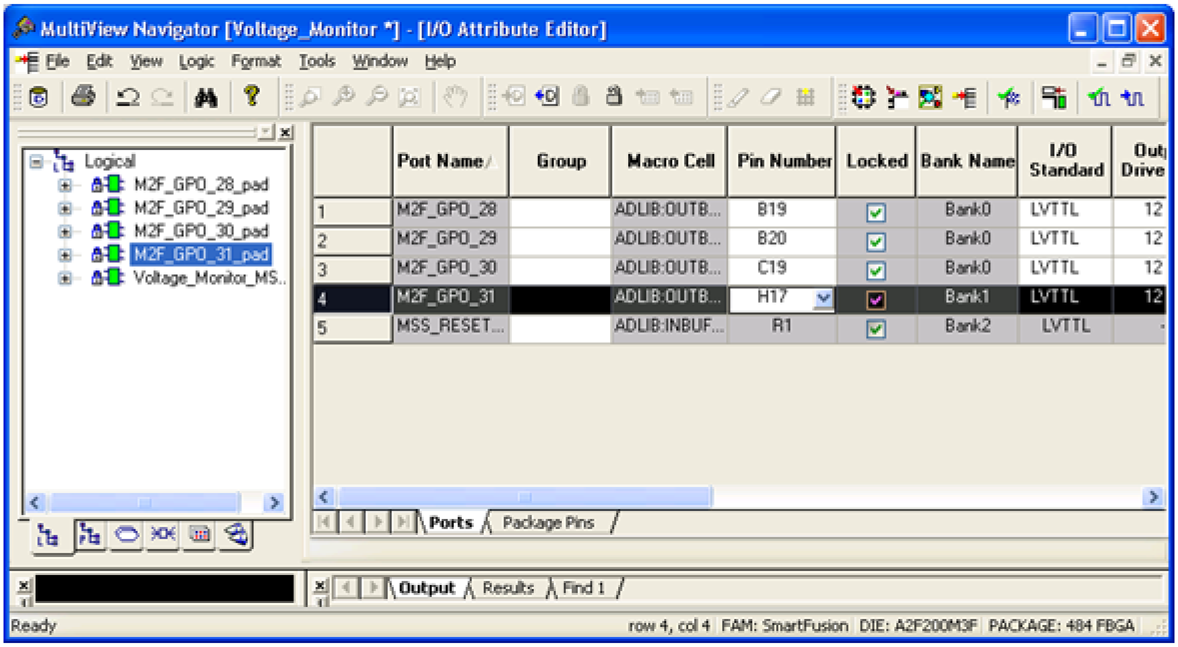

Figure 60 · MultiView Navigator GUI

3. Commit and check the edits using **File > Commit and Check**. Correct any errors that are reported in the MVN log window.
4. Close the MultiView Navigator using **File > Exit**.
5. Close the Designer window and select **Yes** when it prompts to save changes.

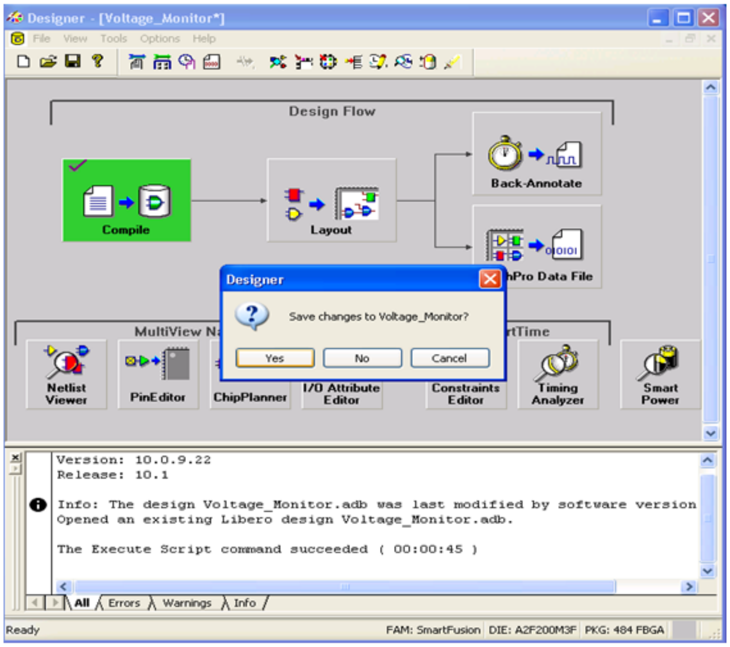

Figure 61 · Designer Window

6. Click **Generate Programming Data** to complete the place and route, verify timing and generate the programming file. This completes the.fdb file generation.

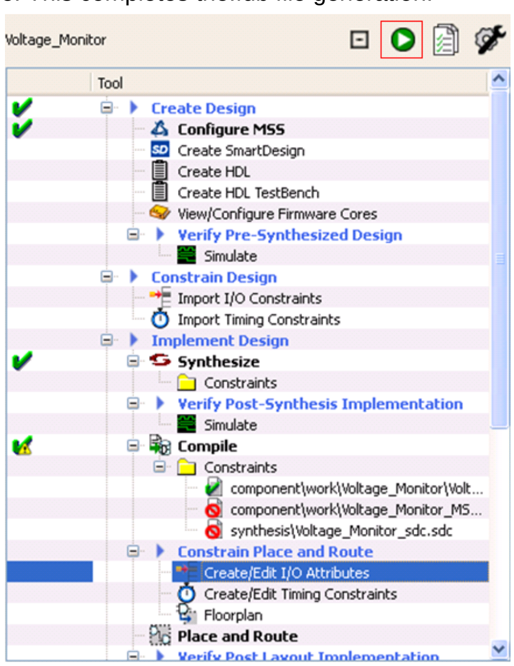

Figure 62 · Generating Programming Data

The **Design Flow** window looks as shown in Figure 63 · .

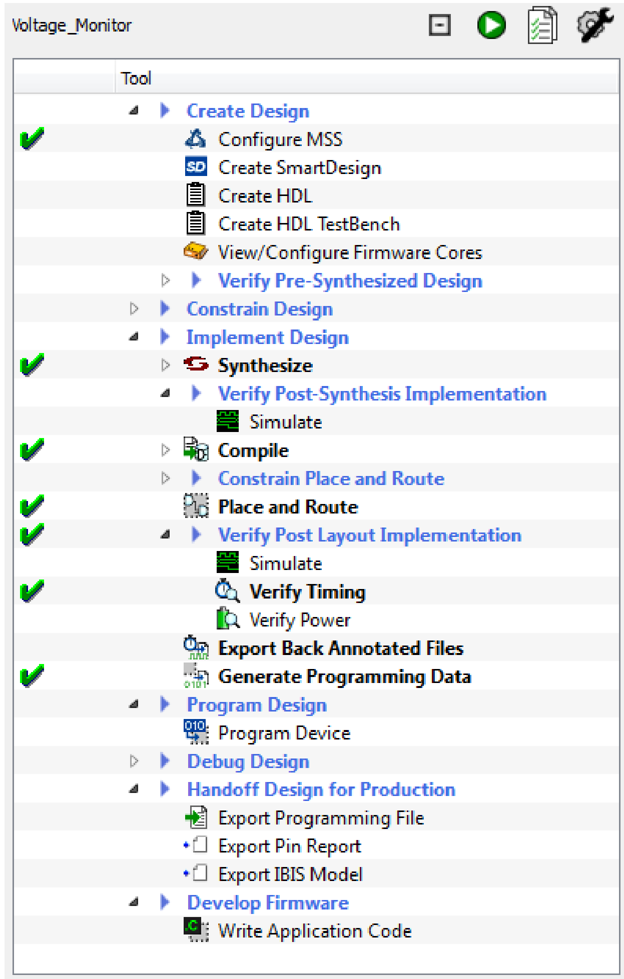

Figure 63 · Design Flow Window After Building the Project

7. Follow Step 5 - Programming SmartFusion Board Using FlashPro.

List of Changes

Revision	Changes	Page
Revision 3 (May 2012)	Replaced Figure 6 · (SAR 38348)	8
	Replaced Figure 21 · (SAR 38348)	15
	Replaced Figure 25 · (SAR 38348)	17
	Replaced Figure 26 · (SAR 38348)	19
	Replaced Figure 27 · (SAR 38348)	20
	Modified Step 6 - Building the Software Application through Keil µVision®4 IDE (SAR 38348)	20
	Modified Appendix – C (SAR 38348)	37
	Replaced Figure 56 · (SAR 38348)	38
	Replaced Figure 58 · (SAR 38348)	39
	Replaced Figure 62 · (SAR 38348)	41
	Replaced Figure 63 · (SAR 38348)	42
Revision 2 (February 2012)	Modified Associated Project Files section (SAR 36902).	3
	Modified Step 2 - Configuring MSS Peripherals section (SAR 36902).	8
	Modified Step 7 - Configuring the Serial Terminal Emulation Program section (SAR 36902).	28
	Updated Figure 38 (SAR 36902).	29
Revision 1 (November 2011)	Updated the document for Libero SoC v10.0 (SAR 35043).	NA
	Corrected the signal name from TM0_Voltage to TM0_voltage in point 10 listed below Figure 12 · (SAR 30307)	12

Note: The revision number is located in the part number after the hyphen. The part number is displayed at the bottom of the last page of the document. The digits following the slash indicate the month and year of publication.

Product Support

Microsemi SoC Products Group backs its products with various support services, including Customer Service, Customer Technical Support Center, a website, electronic mail, and worldwide sales offices. This appendix contains information about contacting Microsemi SoC Products Group and using these support services.

Customer Service

Contact Customer Service for non-technical product support, such as product pricing, product upgrades, update information, order status, and authorization.

From North America, call **800.262.1060**
From the rest of the world, call **650.318.4460**
Fax, from anywhere in the world **408.643.6913**

Customer Technical Support Center

Microsemi SoC Products Group staffs its Customer Technical Support Center with highly skilled engineers who can help answer your hardware, software, and design questions about Microsemi SoC Products. The Customer Technical Support Center spends a great deal of time creating application notes, answers to common design cycle questions, documentation of known issues and various FAQs. So, before you contact us, please visit our online resources. It is very likely we have already answered your questions.

Technical Support

Visit the Microsemi SoC Products Group Customer Support website for more information and support (http://www.microsemi.com/soc/support/search/default.aspx). Many answers available on the searchable web resource include diagrams, illustrations, and links to other resources on website.

Website

You can browse a variety of technical and non-technical information on the Microsemi SoC Products Group home page, at http://www.microsemi.com/soc/.

Contacting the Customer Technical Support Center

Highly skilled engineers staff the Technical Support Center. The Technical Support Center can be contacted by email or through the Microsemi SoC Products Group website.

Email

You can communicate your technical questions to our email address and receive answers back by email, fax, or phone. Also, if you have design problems, you can email your design files to receive assistance. We constantly monitor the email account throughout the day. When sending your request to us, please be sure to include your full name, company name, and your contact information for efficient processing of your request.

The technical support email address is soc_tech@microsemi.com.

My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to My Cases.

Outside the U.S.

Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office. Sales office listings can be found at www.microsemi.com/soc/company/contact/default.aspx.

ITAR Technical Support

For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech_itar@microsemi.com. Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.

Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo CA 92656 USA
Within the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996

50200270-3/05.12