Displaying POT Values over UART

SoftConsole Standalone Flow Tutorial for SmartFusion cSoC

Table of Contents

Introduction

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

1. Creating and implementing a project using SoftConsole and SmartFusion cSoC device.
2. Configuring the peripherals using SmartDesign
3. Configuring the analog compute engine (ACE)
4. Generating the programming file to program the SmartFusion cSoC device
5. Compiling application code
6. Creating and launching a debug session

Tutorial Requirements

Software Requirements

This tutorial requires the following software installed on your PC:

- Libero® system-on-chip (SoC) v10.0 (or later) that can be downloaded from the link below:
  www.microsemi.com/soc/download/software/libero/files.aspx.
- Microsemi SoftConsole v3.3 or later, which is installed as a part of Libero SoC v10.0 installation or can be downloaded from www.microsemi.com/soc/download/software/softconsole/default.aspx.
- FlashPro v10.0 or higher, which is often installed as part of the Microsemi Libero SoC installation and can be launched from within Libero SoC or as a standalone.

Hardware Requirements

This tutorial requires the following hardware:

- SmartFusion Evaluation Kit Board or SmartFusion Development Kit 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 PC.

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_SoftConsole_POTlevel_UART_tutorial_DF.

Note: When launching the SoftConsole from Libero SoC, SoftConsole may open without any workspace or display an error. Refer to the following KB article for the workaround: www.microsemi.com/soc/kb/article.aspx?id=KI8879.

You can download the programming file (*.stp) in release for this tutorial from the Microsemi website: www.microsemi.com/soc/download/rsc/?f=SmartFusion_SoftConsole_POTlevel_UART_tutorial_PF.

MSS Components Used

- ARM® Cortex™-M3 processor
- Communications matrix
- Clock conditioning circuitry (CCC)
- UART_0

- 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 explain how to configure the SmartFusion cSoC MSS peripherals using the SmartFusion cSoC MSS configurator outside the Libero SoC V10.0 flow.

Design Steps

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

- Configure the SmartFusion cSoC MSS peripherals using SmartDesign MSS.
- Generate the programming file and program the SmartFusion cSoC device.
- Create a simple project and run the application using the SoftConsole debugger.

As a part of MSS configuration, this tutorial demonstrates how to configure SmartFusion analog channels and ACE (used to monitor the voltage across the potentiometer).

The UART_0 is used to send the analog to digital converter (ADC) results to a serial terminal program (HyperTerminal).

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 potentiometer (POT) and the four flags are included for voltage monitoring.

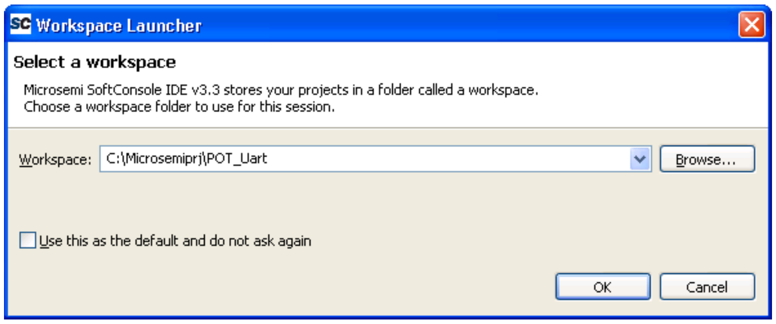
Working with the SoftConsole

Step 1 - Launching SoftConsole

1. Open SoftConsole by clicking **Start > Programs > Microsemi SoftConsole v3.3 > Microsemi SoftConsole IDE**.
2. The **SoftConsole Workspace Launcher** is displayed as shown in Figure 1. Click **Browse** on the SoftConsole Workspace Launcher to navigate to your preferred location (where you will create your project), or enter the location of your workspace. For example: C:\Microsemiprj\POT_Uart). Click **OK**.

   Note: You can also specify the workspace as a default workspace for all your projects by selecting **Use this as the default and do not ask again**.

   You can switch workspaces within SoftConsole by selecting **File > Switch Workspace**.


Workspace Launcher

Select a workspace

Microsemi SoftConsole IDE v3.3 stores your projects in a folder called a workspace.
Choose a workspace folder to use for this session.

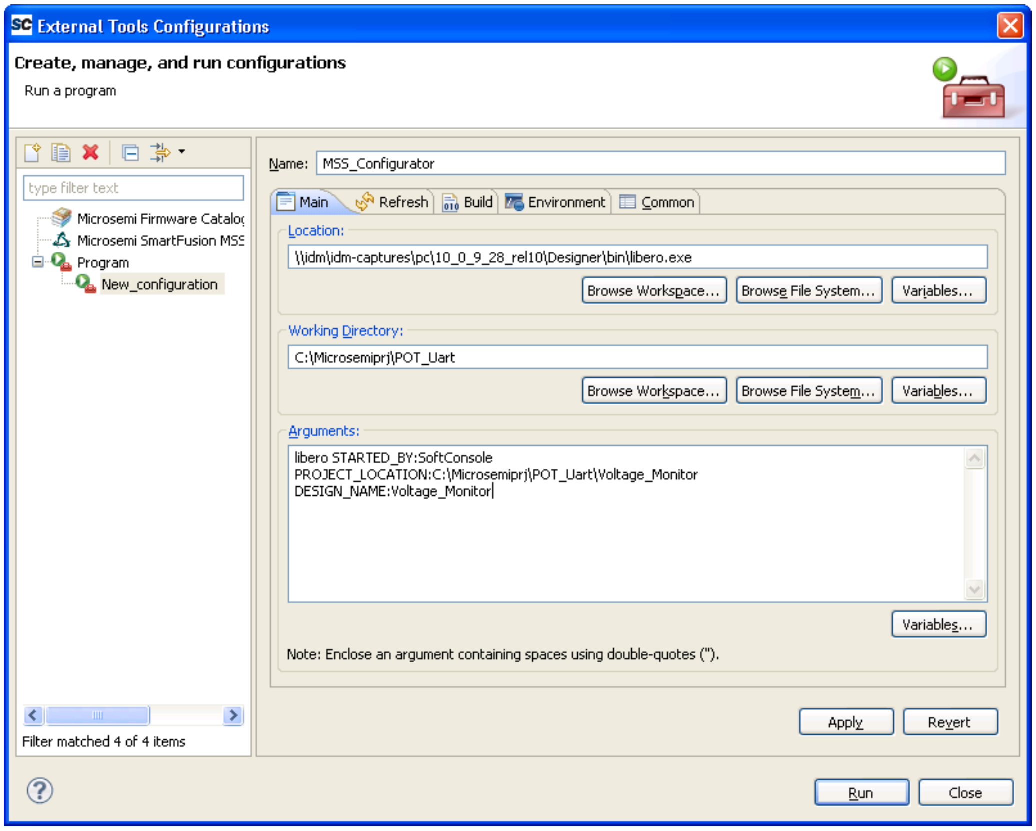
Workspace: C:\Microsemiprj\POT_Uart

Use this as the default and do not ask again

Figure 1 Specifying the SoftConsole Workspace

Step 2 - Configuring MSS Peripherals

1. Open the **External Tools** dialog box using **Run > External Tools > External Tools Configurations**.
2. Double-click **Program** to create a new configuration.
3. Enter **MSS_Configurator** in the **Name** field.
4. Click **Browse File System** in the **Location** field. The **Open** dialog box is displayed.
5. Navigate to the **Libero.exe**.
6. In the **Arguments** field, enter the following:
   - STARTED_BY: SoftConsole
   - PROJECT_LOCATION: C:\Microsemiprj\POT_Uart\Voltage_Monitor
   - DESIGN_NAME: Voltage_Monitor
   - Click **Apply** and then click **Run**.



Name: MSS_Configurator

Location: \\idm\idm-captures\pc\10_0_9_28_rel10\Designer\bin\libero.exe

Working Directory: C:\Microsemiprj\POT_Uart

Arguments:

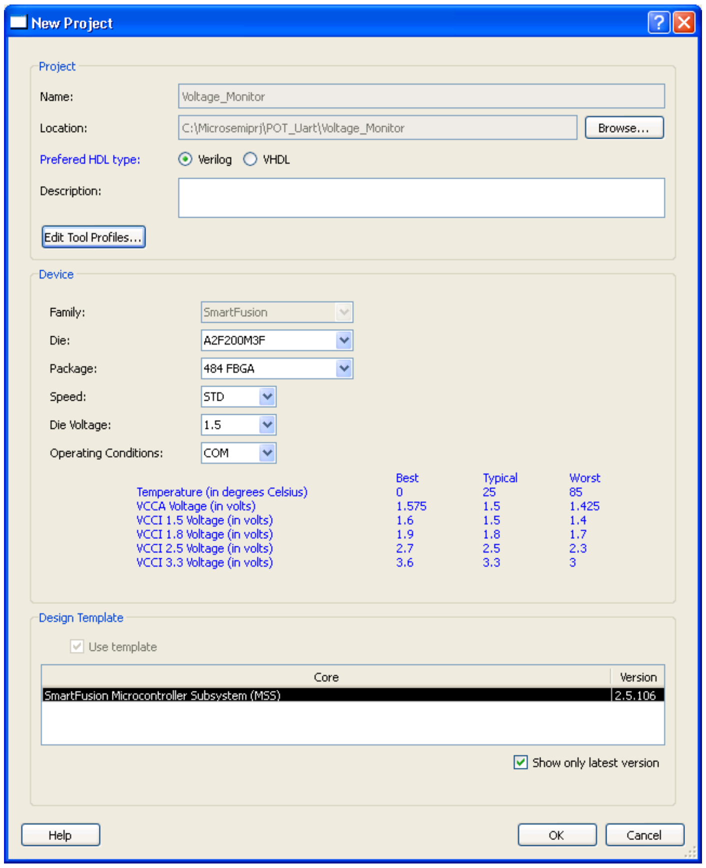
```
libero STARTED_BY:SoftConsole
PROJECT_LOCATION:C:\Microsemiprj\POT_Uart\Voltage_Monitor
DESIGN_NAME:Voltage_Monitor
```

Note: Enclose an argument containing spaces using double-quotes (").

Figure 2 Running SmartFusion MSS Configurator

7. The Libero SoC appears with a **New Project** dialog box, as shown in Figure 3 below. Select the following:
   - 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
   - Die Voltage: 1.5

The **Project Name** and **Project Directory** parameters are already specified.


New Project

Project

Name: Voltage_Monitor

Location: C:\Microsemiprj\POT_Uart\Voltage_Monitor

Prefered HDL type: [✓ Verilog] VHDL

Description:

Device

Family: SmartFusion

Die: A2F200M3F

Package: 484 FBGA

Speed: STD

Die Voltage: 1.5

Operating Conditions: COM

	Best	Typical	Worst
Temperature (in degrees Celsius)	0	25	85
VCCA Voltage (in volts)	1.575	1.5	1.425
VCCI 1.5 Voltage (in volts)	1.6	1.5	1.4
VCCI 1.8 Voltage (in volts)	1.9	1.8	1.7
VCCI 2.5 Voltage (in volts)	2.7	2.5	2.3
VCCI 3.3 Voltage (in volts)	3.6	3.3	3

Design Template

✓ Use template

Core	Version
SmartFusion Microcontroller Subsystem (MSS)	2.5.106

✓ Show only latest version

Figure 3 New Project Dialog Window

Note: If you do not see the latest MSS version (v2.5.106 or later), you need to change your repositories settings. Steps for setting up repositories are described in Appendix A – Libero SoC Vault/Repository Settings.

If your vault does not have MSS core then, double click on MSS name in the Design Template to download the core.

8. Click **OK**. The MSS canvas is displayed, as shown in Figure 4.

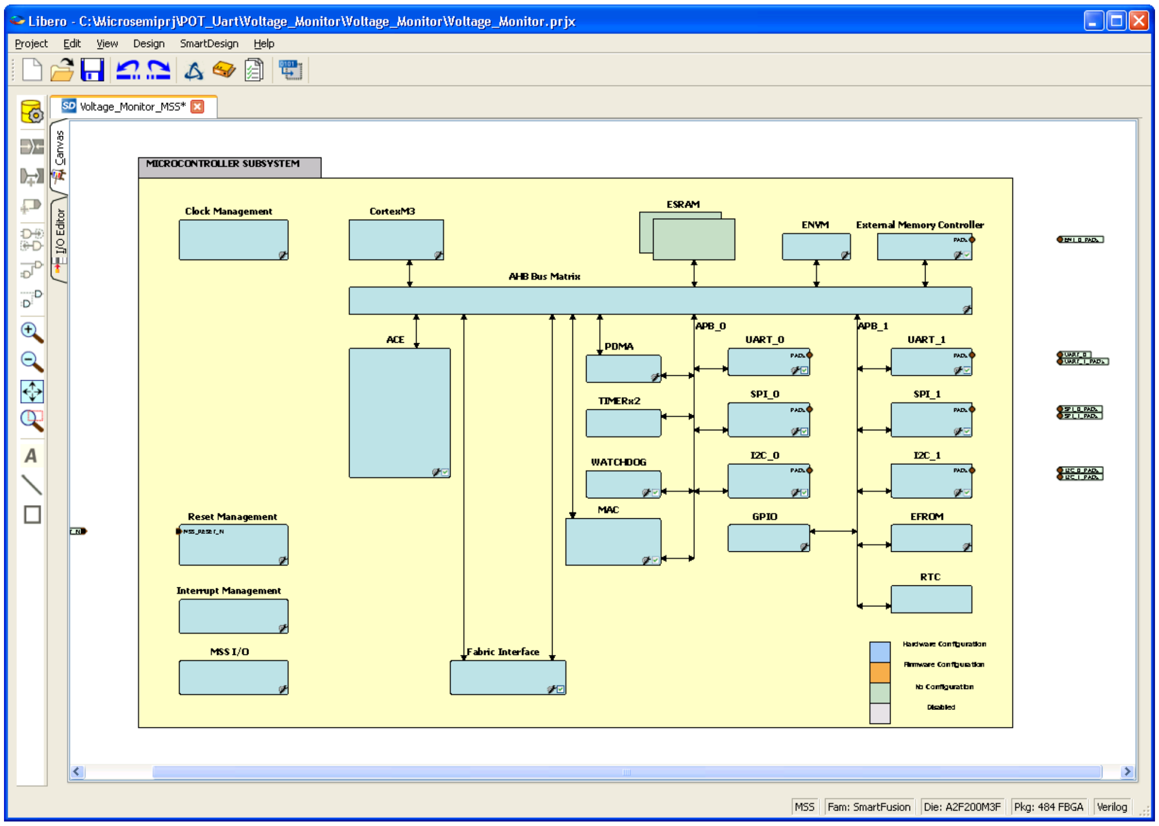


Figure 4 MSS in the SmartDesign Canvas

The enabled MSS peripherals are highlighted in blue, and can be configured in 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 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 looks as shown in Figure 5.

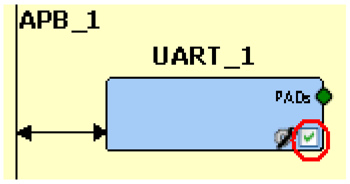


Figure 5 Enabling the Peripheral

For this design we will be using the analog compute engine (ACE), UART_0, and clock management peripherals. Other peripherals must be disabled.

9. 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
   - 10/100CLK: Cleared check box (by default it will be cleared, if MAC is disabled)
   - Use default settings for all other fields.

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

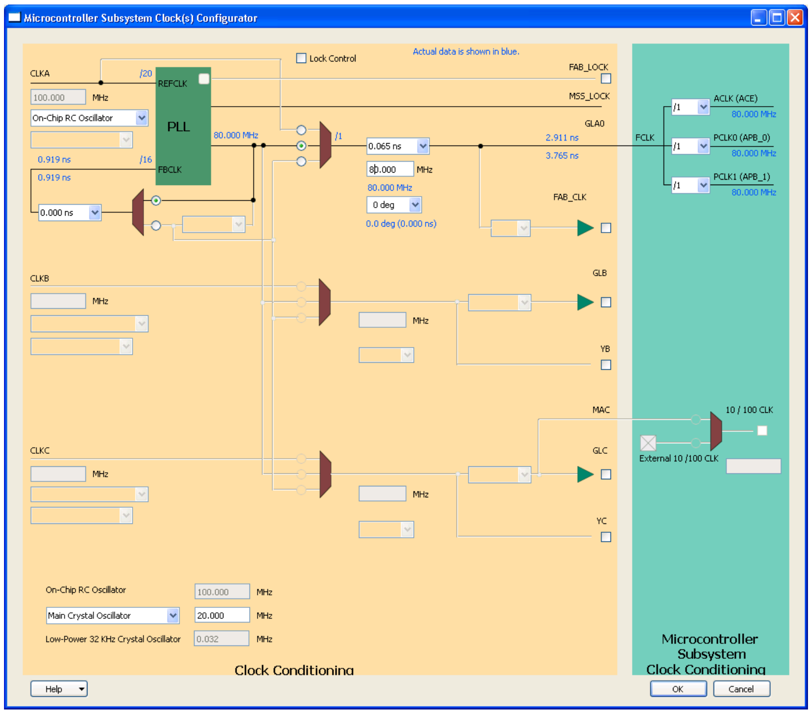


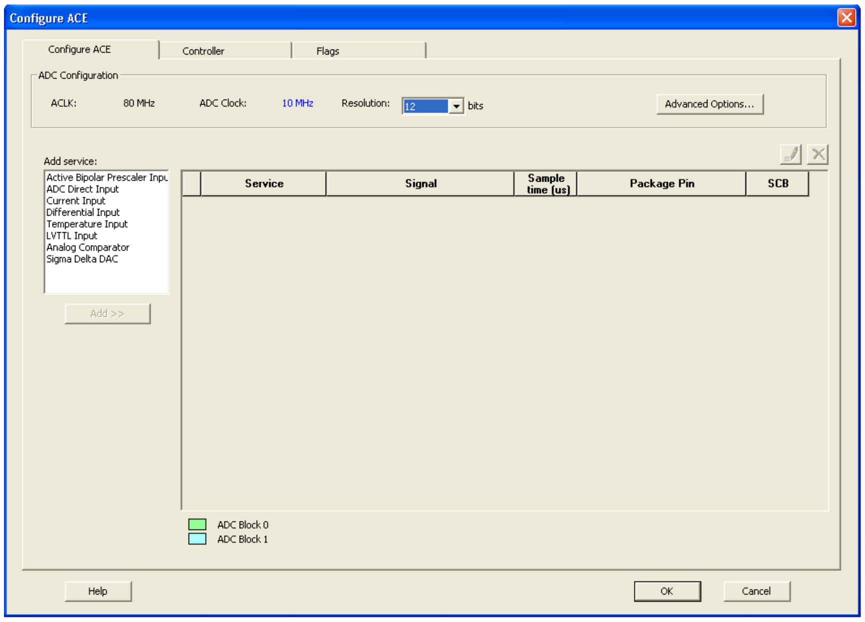
Figure 6 MSS Clock Configuration

11. Click **File > Save** to save SmartFusion_UART_MSS.

Configuring ACE

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

1. Connect TM0 to the potentiometer (POT) on the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board. Configure a voltage monitor to measure the voltage across the POT and also 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 will be used to monitor the voltage.



ADC Configuration

ACLK: 80 MHz ADC Clock: 10 MHz Resolution: 12 bits

Add service:
Active Bipolar Prescaler Input
ADC Direct Input
Current Input
Differential Input
Temperature Input
LVTTL Input
Analog Comparator
Sigma Delta DAC

Service	Signal	Sample time (us)	Package Pin	SCB

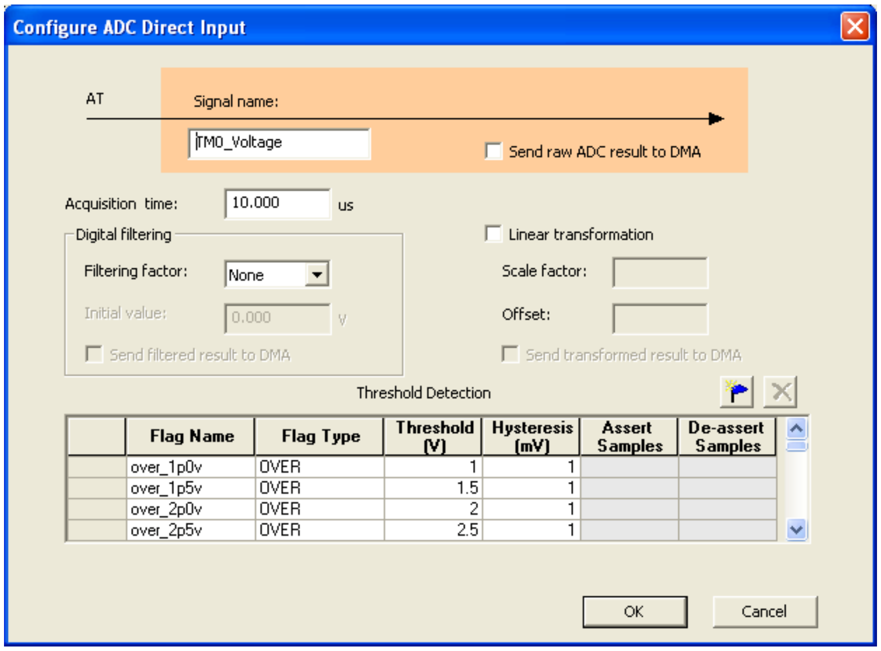
🟩 ADC Block 0
🟦 ADC Block 1

Figure 7 MSS ACE Configuration

2. Select **ADC Direct Input** and click **Add** (or double-click ADC Direct Input) and enter the parameters as shown in Table 1
   - Signal name: TM0_Voltage
   - Send raw results to DMA: un-checked
   - Acquisition time: 10 µs
   - Filtering factor: None

Table 1 · Configure ADC Direct Input Dialog Box

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



Signal name: TM0_Voltage

☐ Send raw ADC result to DMA

Acquisition time: 10.000 us

Digital filtering

Filtering factor: None

Initial value: 0.000 V

☐ Send filtered result to DMA

☐ Linear transformation

Scale factor:

Offset:

☐ Send transformed result to DMA

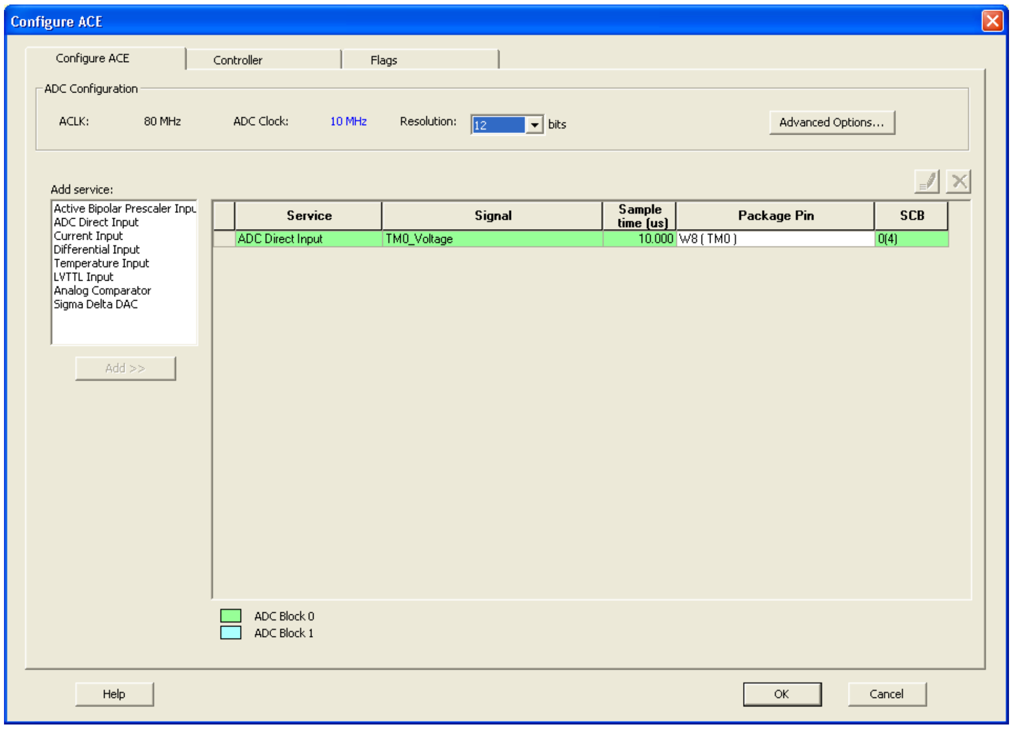
Threshold Detection

Flag Name	Flag Type	Threshold (V)	Hysteresis (mV)	Assert Samples	De-assert Samples
over_1p0v	OVER	1	1		
over_1p5v	OVER	1.5	1		
over_2p0v	OVER	2	1		
over_2p5v	OVER	2.5	1		

Figure 8 MSS ADC Direct Input Configuration

3. Click **OK**.
4. Assign ADC Direct Input Signal to package pin W8 in the Configure ACE dialog box.

5. The **Configure ACE** tab is displayed as shown in Figure 9.

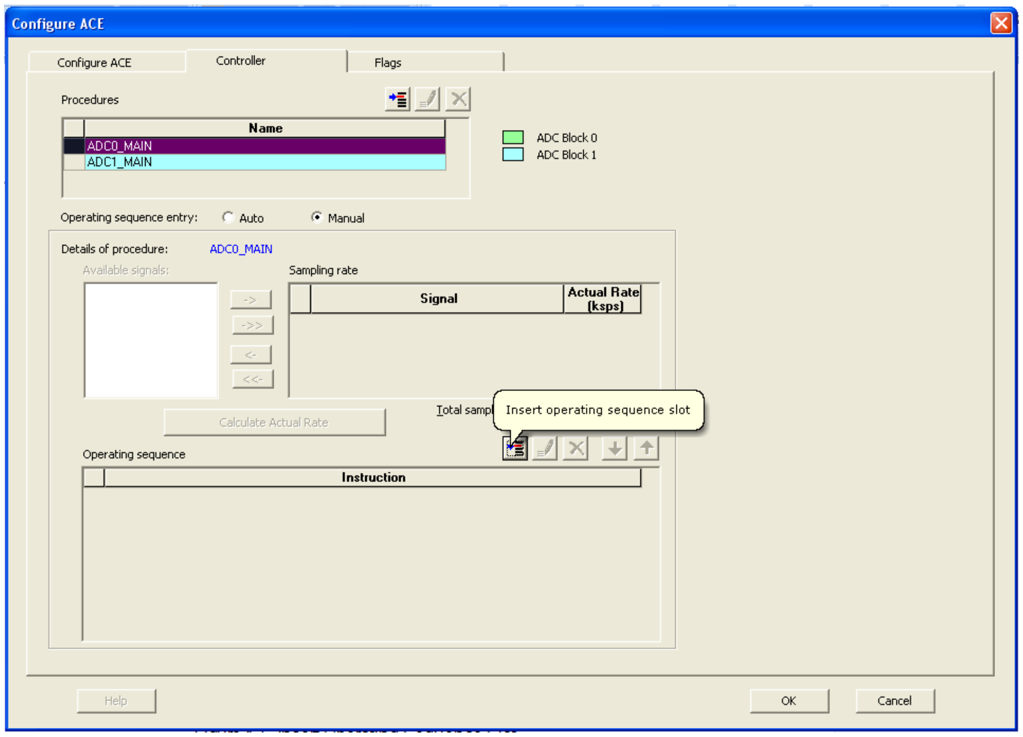


Service	Signal	Sample time (us)	Package Pin	SCB
ADC Direct Input	TM0_Voltage	10.000	W8 (TM0)	0(4)

Figure 9 The Configure ACE Tab

The next step in configuring the ACE is to enable the sampling sequence. This configuration dialog is launched by clicking the **Controller** tab (next to the **Configure ACE** tab).

6. Select **Manual** as the **Operating sequence entry** in the **Controller** tab.



Operating sequence entry: Auto [✓ Manual]

Figure 10 Insert Operating Sequence Slot

7. Click the **Insert operating sequence slot**, highlighted in Figure 10.

8. Select **SAMPLE**.

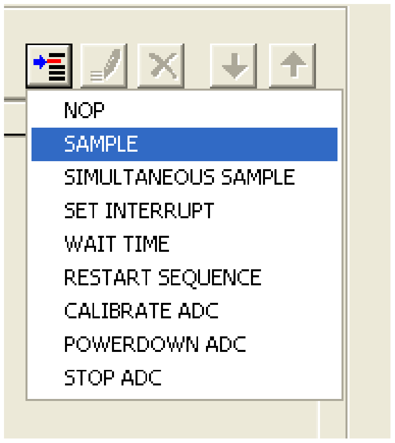
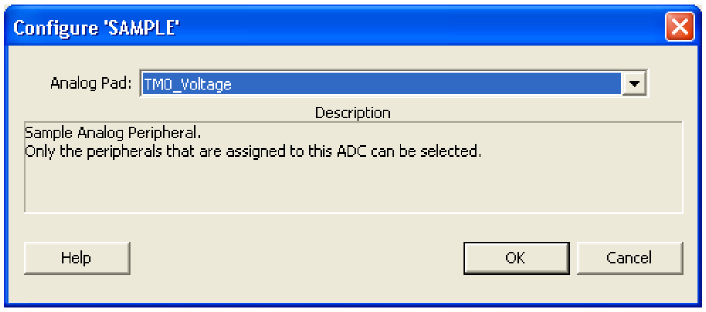


Figure 11 Select SAMPLE

9. The Configure SAMPLE window is displayed. Select **TM0_Voltage** from the drop-down list and click **OK**.

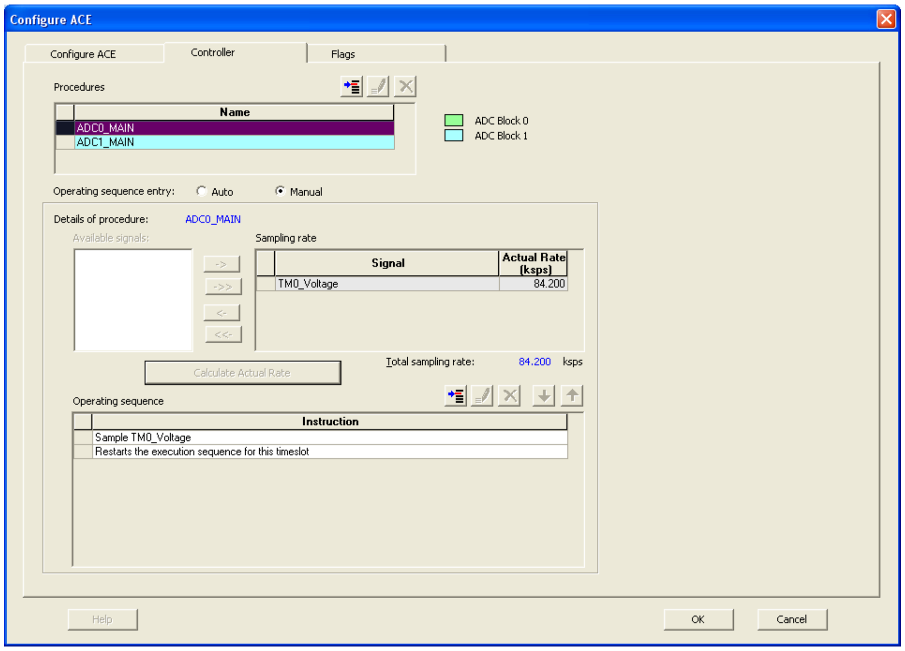


Analog Pad: TM0_Voltage

Description
Sample Analog Peripheral.
Only the peripherals that are assigned to this ADC can be selected.

Figure 12 Configure SAMPLE

10. Click **Insert operating sequence slot** again and select **RESTART SEQUENCE**.
11. Click **Calculate Actual Rate**. The final **Controller** tab window is displayed as shown in Figure 13.



	Name
	ADC0_MAIN
	ADC1_MAIN

🟩 ADC Block 0
🟦 ADC Block 1

Operating sequence entry: Auto [✓ Manual]

Details of procedure: ADC0_MAIN

Sampling rate

	Signal	Actual Rate (ksps)
	TM0_Voltage	84.200

Total sampling rate: 84.200 ksps

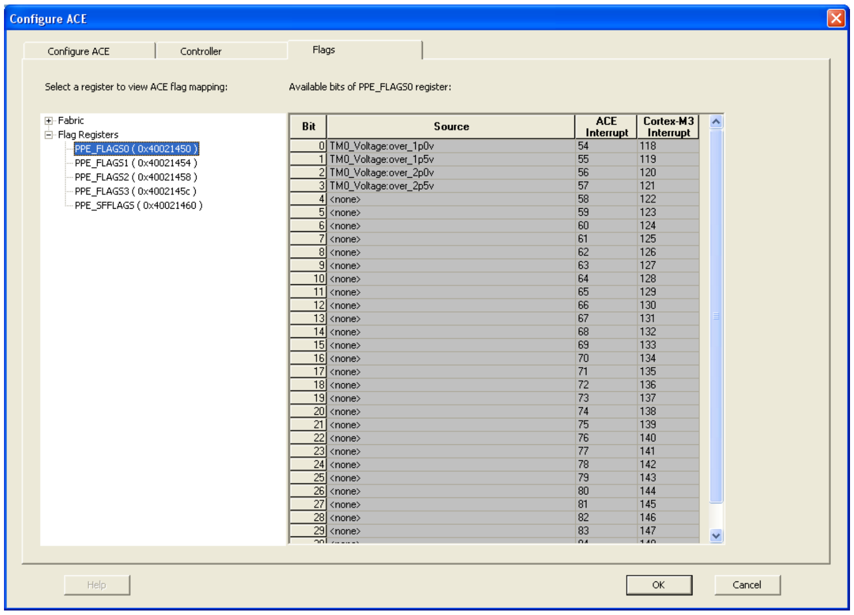
	Instruction
	Sample TM0_Voltage
	Restarts the execution sequence for this timeslot

Figure 13 MSS ACE Configuration: Final Controller Tab

12. Click the **Flags** tab in the Configure ACE window. This tab lists the Flags set from PPE registers.

13. Click the **+** sign to expand the Flag Registers group. The PPE_FLAGSn registers contain the user-defined flags.

    Select PPE_FLAGS0 (FLAGBANK0). Observe that 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.



Select a register to view ACE flag mapping:

- Fabric
- Flag Registers
  - PPE_FLAGS0 (0x40021450)
  - PPE_FLAGS1 (0x40021454)
  - PPE_FLAGS2 (0x40021458)
  - PPE_FLAGS3 (0x4002145c)
  - PPE_SFFLAGS (0x40021460)

Available bits of PPE_FLAGS0 register:

Bit	Source	ACE Interrupt	Cortex-M3 Interrupt
0	TM0_Voltage:over_1p0v	54	118
1	TM0_Voltage:over_1p5v	55	119
2	TM0_Voltage:over_2p0v	56	120
3	TM0_Voltage:over_2p5v	57	121
4	<none>	58	122
5	<none>	59	123
6	<none>	60	124
7	<none>	61	125
8	<none>	62	126
9	<none>	63	127
10	<none>	64	128
11	<none>	65	129
12	<none>	66	130
13	<none>	67	131
14	<none>	68	132
15	<none>	69	133
16	<none>	70	134
17	<none>	71	135
18	<none>	72	136
19	<none>	73	137
20	<none>	74	138
21	<none>	75	139
22	<none>	76	140
23	<none>	77	141
24	<none>	78	142
25	<none>	79	143
26	<none>	80	144
27	<none>	81	145
28	<none>	82	146
29	<none>	83	147

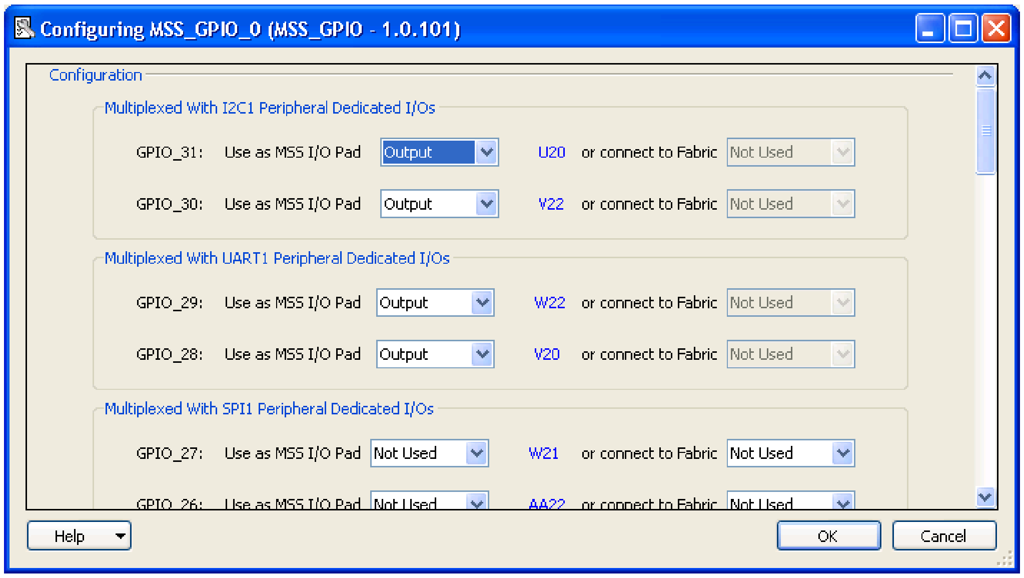
Figure 14 ACE Flag Mapping: PPE Flag Registers

14. Click **OK** to close the ACE configuration box.

Configuring the General Purpose Input/Output (GPIO) Peripheral

Note: If you are not using SmartFusion Evaluation Kit Board – REV 2, skip the configuration of GPIO.

Double-click the GPIO block in the MSS component, configure as shown in Figure 15, and click **OK**.



Multiplexed With I2C1 Peripheral Dedicated I/Os

GPIO_31:	Use as MSS I/O Pad	Output	U20	or connect to Fabric	Not Used
GPIO_30:	Use as MSS I/O Pad	Output	V22	or connect to Fabric	Not Used

Multiplexed With UART1 Peripheral Dedicated I/Os

GPIO_29:	Use as MSS I/O Pad	Output	W22	or connect to Fabric	Not Used
GPIO_28:	Use as MSS I/O Pad	Output	V20	or connect to Fabric	Not Used

Multiplexed With SPI1 Peripheral Dedicated I/Os

GPIO_27:	Use as MSS I/O Pad	Not Used	W21	or connect to Fabric	Not Used
GPIO_26:	Use as MSS I/O Pad	Not Used	AA22	or connect to Fabric	Not Used

Figure 15 Configure MSS_GPIO_0

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 (A2F_EVAL-KIT Rev 2).

Generating the MSS Component

1. The MSS canvas is displayed, as shown in Figure 16.

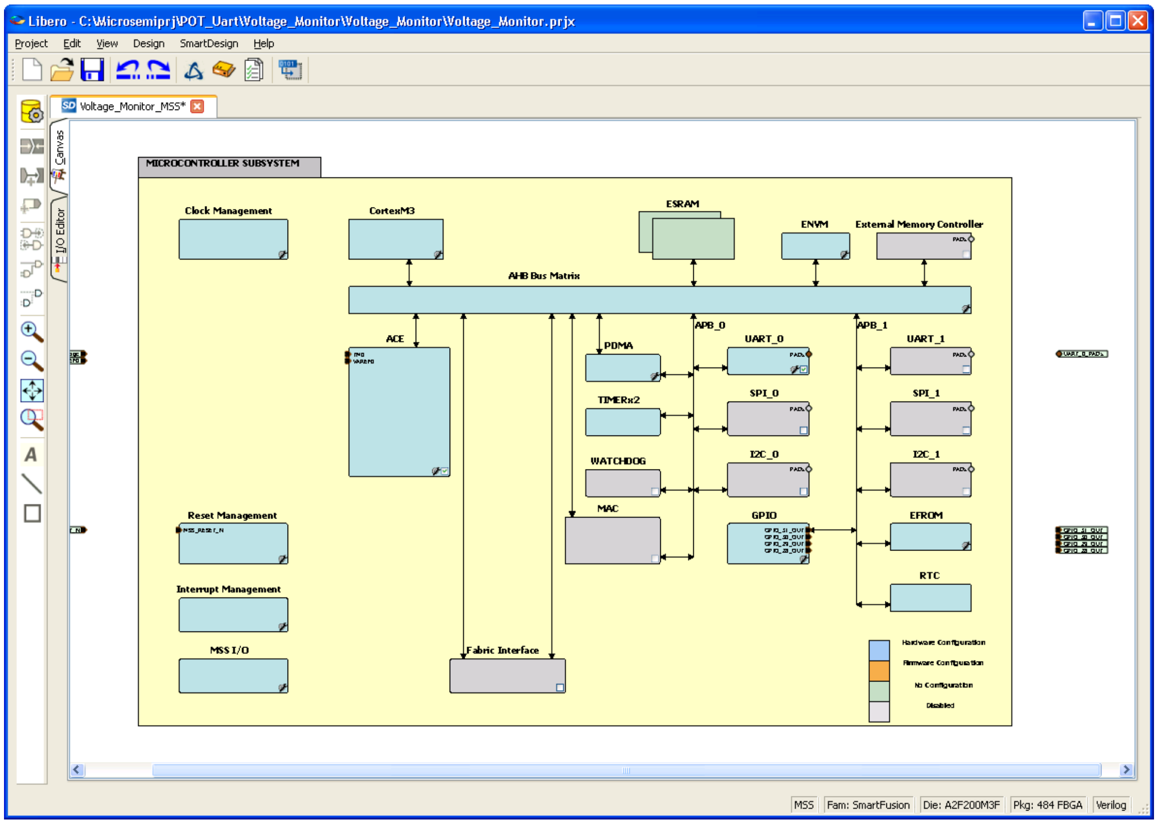


Figure 16 The MSS Canvas

2. Save your design using **File > Save Voltage_Monitor_MSS**.
3. Click **Design > Configure Firmware** as shown in Figure 17.

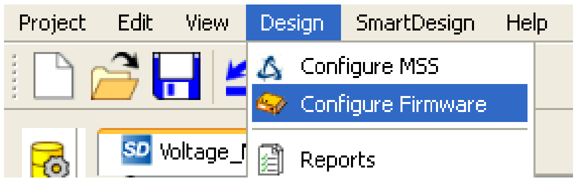
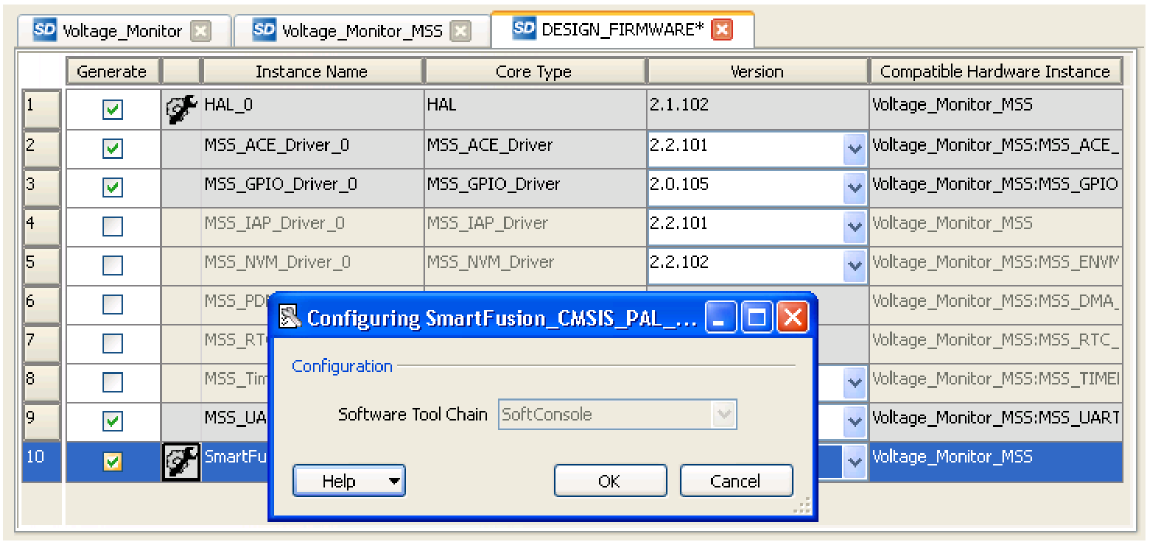


Figure 17 The Firmware Tab

Note: 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 to set your Repositories. If you are opening the SmartDesign for the first time then you need to download the all firmware drivers from the **Firmware** tab in the SmartDesign

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.



	Generate		Instance Name	Core Type	Version	Compatible Hardware Instance
1	✓		HAL_0	HAL	2.1.102	Voltage_Monitor_MSS
2	✓		MSS_ACE_Driver_0	MSS_ACE_Driver	2.2.101	Voltage_Monitor_MSS:MSS_ACE_
3	✓		MSS_GPIO_Driver_0	MSS_GPIO_Driver	2.0.105	Voltage_Monitor_MSS:MSS_GPIO
4	☐		MSS_IAP_Driver_0	MSS_IAP_Driver	2.2.101	Voltage_Monitor_MSS
5	☐		MSS_NVM_Driver_0	MSS_NVM_Driver	2.2.102	Voltage_Monitor_MSS:MSS_ENVM
6	☐		MSS_PDI			Voltage_Monitor_MSS:MSS_DMA_
7	☐		MSS_RT			Voltage_Monitor_MSS:MSS_RTC_
8	☐		MSS_Tim			Voltage_Monitor_MSS:MSS_TIMEI
9	✓		MSS_UA			Voltage_Monitor_MSS:MSS_UART
10	✓		SmartFu			Voltage_Monitor_MSS

Configuring SmartFusion_CMSIS_PAL_...

Configuration

Software Tool Chain: SoftConsole

Help | OK | Cancel

Figure 18 Configuring SmartFusion_CMSIS_PAL_0

5. Click **OK**.
6. Click **File > Save to save the Design_Firmware**.
7. **Save** the design and generate the MSS component by clicking **Generate Component** or clicking **SmartDesign > Generate Component**.

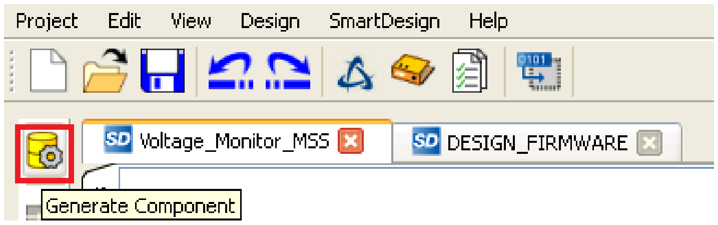


Figure 19 Select SoftConsole in the Configuration Window

8. Confirm that the design was successfully generated.

   Note: If errors are indicated, open the log window (**View > Log Window**) to get additional information.
9. Open the memory map for the design (**Design > Reports**). Scroll the window to become familiar with the locations of the peripherals. Close the window when finished.
10. Close Libero SoC (**File > Exit**)

Step 3 - Programming SmartFusion Board Using FlashPro

Jumper Settings for SmartFusion Evaluation Kit Board

Before you proceed with programming the device, ensure that 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 the *SmartFusion Development Kit User's Guide* for additional information.

- JP10: Connect pin 1 and 2.
- JP7: Connect pin 1 and 2 for LCPS programming mode.
- J6: Connect pins 1 and 2 with the jumper.
- JP6: Connect pins 2 and 3 with the jumper.

- J13: Connect USB cable to the J13 connector. When the cable is connected, the FlashPro4 or FlashPro drivers might be installed if they are not already installed.
- J14: Connect one end of USB mini B cable to J14.

Jumper Settings for SmartFusion Development Kit Board

- 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 the following table:

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. Click **Program Device** as shown in Figure 20 or click **Design > Program Device**.

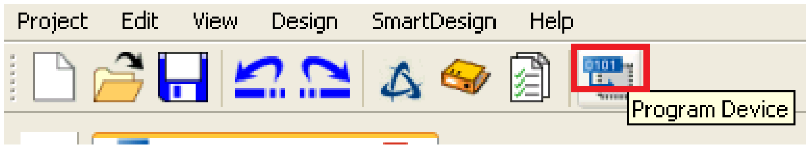


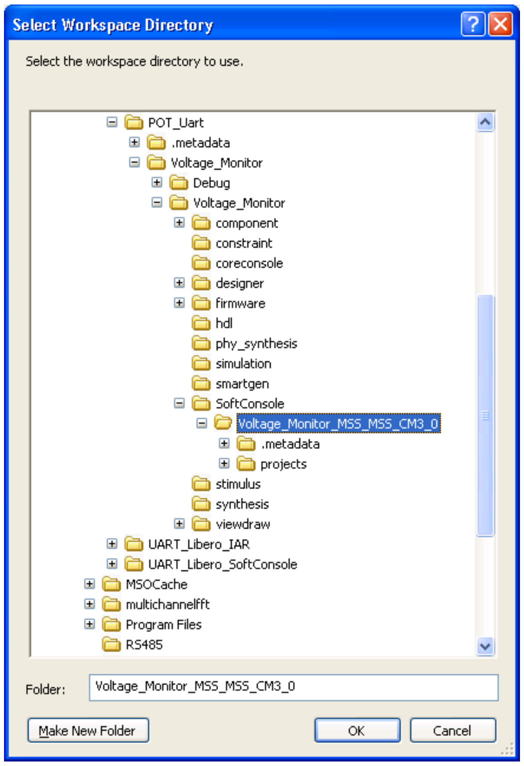
Figure 20 Select SoftConsole in the Configuration Window

Note: If errors are indicated, open the Reports (**Design > Reports**) to get additional information.

2. Close Libero (**File > Exit**).

Step 4 - Building the Project

1. Invoke the SoftConsole project generated by Libero SoC v10.0 using **File > Switch Workspace > Other…** and browse to the SoftConsole workspace as shown in Figure 21.

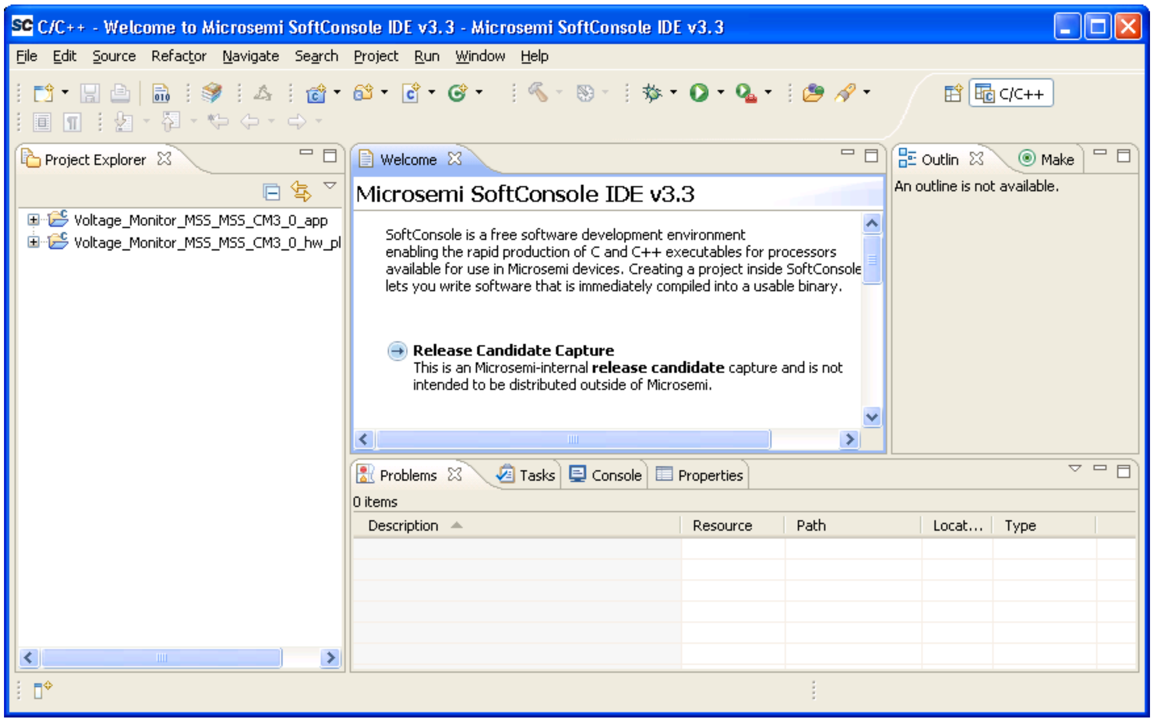


Folder: Voltage_Monitor_MSS_MSS_CM3_0

Figure 21 Specifying the Workspace

Note: If you are using the provided design files and if SoftConsole opens without a workspace or displays an error when opening the project then refer to www.microsemi.com/soc/kb/article.aspx?id=KI8879.

2. The SoftConsole window looks as shown in Figure 22.


C/C++ - Welcome to Microsemi SoftConsole IDE v3.3 - Microsemi SoftConsole IDE v3.3
File Edit Source Refactor Navigate Search Project Run Window Help
C/C++
Project Explorer
Voltage_Monitor_MSS_MSS_CM3_0_app
Voltage_Monitor_MSS_MSS_CM3_0_hw_pl
Welcome
Outlin
Make
Problems
Tasks
Console
Properties
Description
Resource
Path
Locat...
Type

Microsemi SoftConsole IDE v3.3

SoftConsole is a free software development environment enabling the rapid production of C and C++ executables for processors available for use in Microsemi devices. Creating a project inside SoftConsole lets you write software that is immediately compiled into a usable binary.

Release Candidate Capture
This is an Microsemi-internal **release candidate** capture and is not intended to be distributed outside of Microsemi.

An outline is not available.

0 items

Figure 22 SoftConsole Window

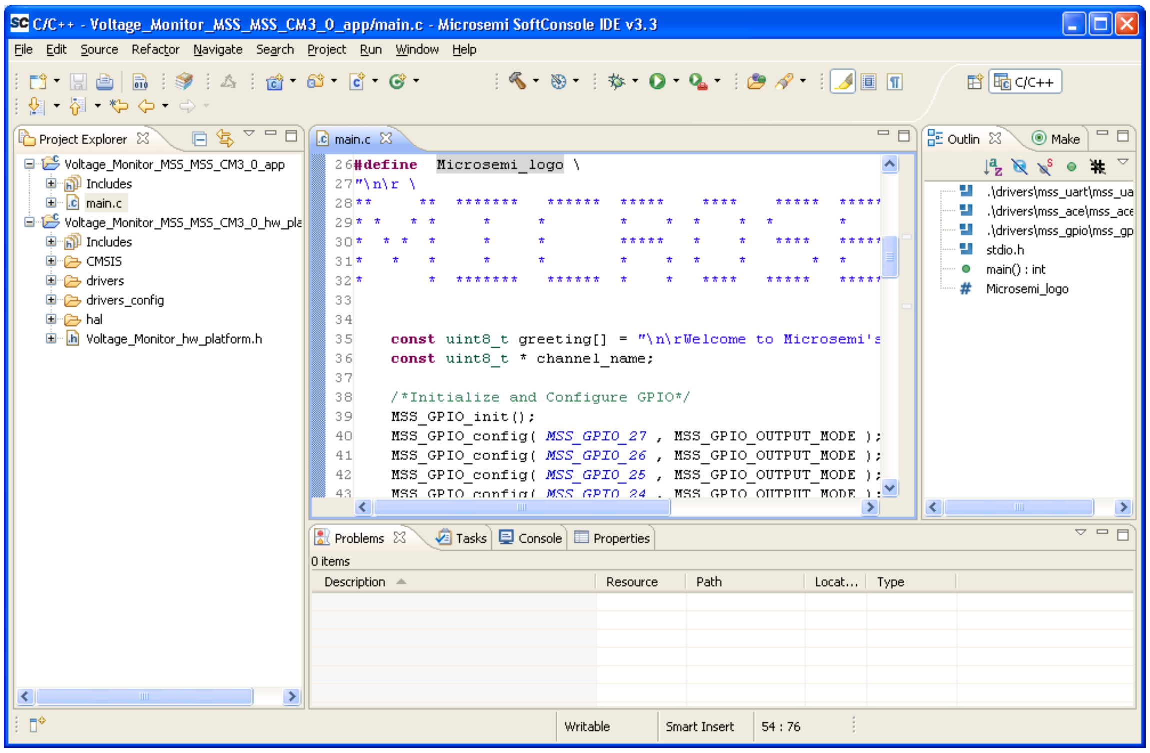
3. Enter the code provided below in **main.c** file in the Voltage_Monitor_MSS_MSS_CM3_0_app project.

Sample C code

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

#define  Microsemi_logo \
"\n\r \
**     **  *******   ******  *****    ****    *****  ******  **     **  ******* \n\r \
* *   * *     *     *        *    *  *    *  *       *       * *   * *     *    \n\r \
*  * *  *     *     *        *****   *    *   ****   ******  *  * *  *     *    \n\r \
*   *   *     *     *        *    *  *    *       *  *       *   *   *     *    \n\r \
*       *  *******   ******  *    *   ****   *****   ******  *       *  ******* "

int main()
{

        const uint8_t greeting[] = "\n\rWelcome to Microsemi's SmartFusion Voltage
Monitor\n\n\r";
    const uint8_t * channel_name;
/*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_string( &g_mss_uart0, (const uint8_t*)Microsemi_logo );
    MSS_UART_polled_tx( &g_mss_uart0, greeting, sizeof(greeting) );

    channel_name = ACE_get_channel_name( TM0_Voltage );

    for (;;)
    {
        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( (char *)display_buffer, sizeof(display_buffer),
                "%s : -%.3fV\r", channel_name, ((float)(-adc_value_mv) / (float)(1000)));
               }
        else
            {
                       snprintf( (char *)display_buffer, sizeof(display_buffer),
                "%s : %.3fV\r", channel_name, ((float)(adc_value_mv) / (float)(1000)));
            }
MSS_UART_polled_tx_string( &g_mss_uart0, display_buffer );

        /* checking the status of Voltage flags */
        int32_t flag_status_2p5v = ACE_get_flag_status(TM0_Voltage_over_2p5v);
               int32_t flag_status_2p0v = ACE_get_flag_status(TM0_Voltage_over_2p0v);
               int32_t flag_status_1p5v = ACE_get_flag_status(TM0_Voltage_over_1p5v);
               int32_t flag_status_1p0v = ACE_get_flag_status(TM0_Voltage_over_1p0v);

               /* Voltage flags are displayed on the LEDs through GPIO */
               uint32_t gpio_output;
```

```
                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 );
      }
    return 0;
}
/****************************************************************************/
```

4. The SoftConsole window looks similar to Figure 23.

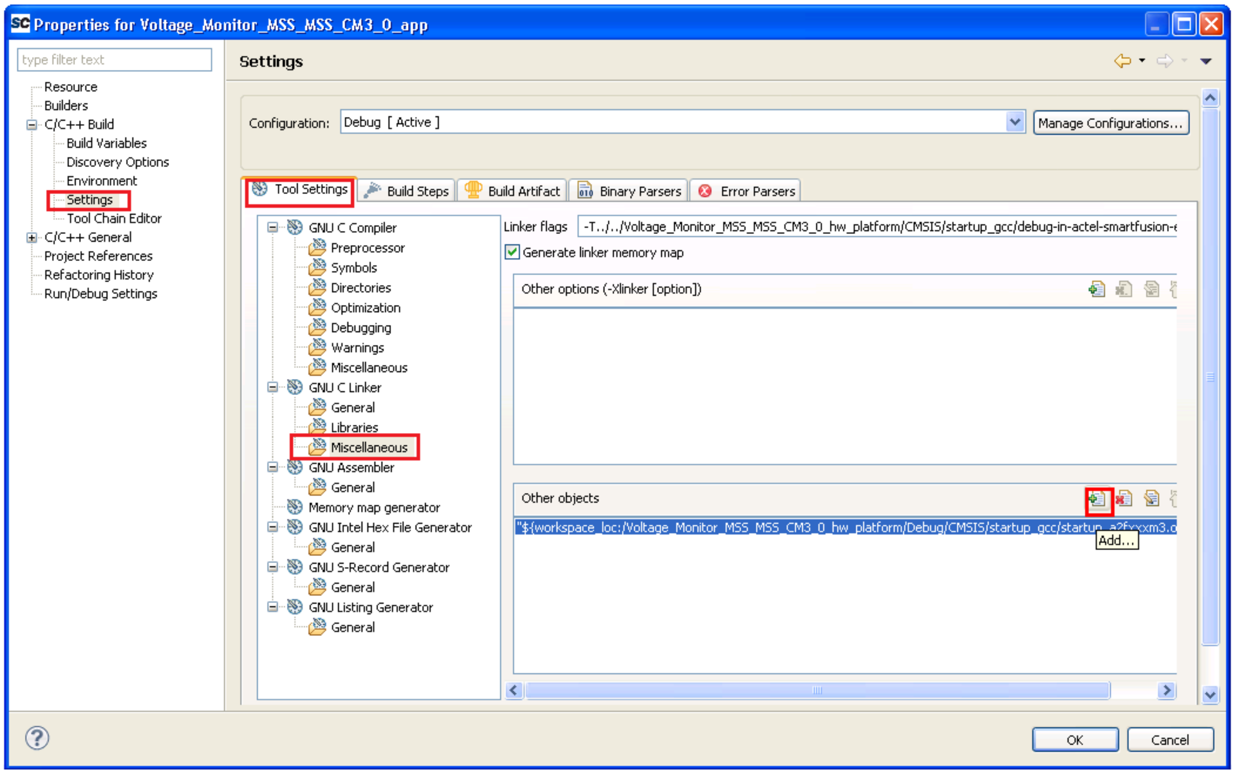

C/C++ - Voltage_Monitor_MSS_MSS_CM3_0_app/main.c - Microsemi SoftConsole IDE v3.3
File Edit Source Refactor Navigate Search Project Run Window Help
C/C++
Project Explorer
Voltage_Monitor_MSS_MSS_CM3_0_app
Includes
main.c
Voltage_Monitor_MSS_MSS_CM3_0_hw_pla
Includes
CMSIS
drivers
drivers_config
hal
Voltage_Monitor_hw_platform.h
main.c
Outlin
Make
.\drivers\mss_uart\mss_ua
.\drivers\mss_ace\mss_ace
.\drivers\mss_gpio\mss_gp
stdio.h
main() : int
Microsemi_logo
Problems
Tasks
Console
Properties
0 items
Description
Resource
Path
Locat...
Type
Writable
Smart Insert
54 : 76

```
#define  Microsemi_logo \
"\n\r \
**     **  *******   ******  *****    ****    *****  *****
* *   * *     *     *        *    *  *    *  *       *
*  * *  *     *     *        *****   *    *   ****   *****
*   *   *     *     *        *    *  *    *       *  *
*       *  *******   ******  *    *   ****   *****   *****


    const uint8_t greeting[] = "\n\rWelcome to Microsemi's
    const uint8_t * channel_name;

    /*Initialize and Configure GPIO*/
    MSS_GPIO_init();
    MSS_GPIO_config( MSS_GPIO_27 , MSS_GPIO_OUTPUT_MODE );
    MSS_GPIO_config( MSS_GPIO_26 , MSS_GPIO_OUTPUT_MODE );
    MSS_GPIO_config( MSS_GPIO_25 , MSS_GPIO_OUTPUT_MODE );
    MSS_GPIO_config( MSS_GPIO_24 , MSS_GPIO_OUTPUT_MODE );
```

Figure 23 SoftConsole workspace

5. Right-click on **Voltage_Monitor_MSS_MSS_CM3_0_app** project and select **Properties**.
6. Click **Settings** under **C/C++ Build**.

7. Select **Miscellaneous** under **GNU C Linker** in **Tool Settings** tab as shown in Figure 24.



Linker flags: -T../../Voltage_Monitor_MSS_MSS_CM3_0_hw_platform/CMSIS/startup_gcc/debug-in-actel-smartfusion-e

☑ Generate linker memory map

Other objects: "${workspace_loc:/Voltage_Monitor_MSS_MSS_CM3_0_hw_platform/Debug/CMSIS/startup_gcc/startup_a2fxxxm3.o

Figure 24 Properties Window

8. Click **Add** in the **Other objects** section. The **Add file path** dialog window is displayed, as shown in Figure 25.

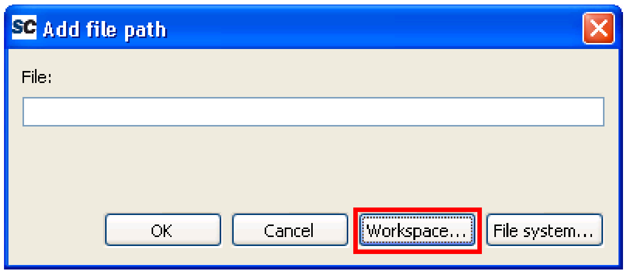


Figure 25 Add File Path Window

9. Click **Workspace** and add the **newlib_stubs.c** file under **Voltage_Monitor_MSS_MSS_CM3_0_hw_platform > CMSIS > startup_gcc > newlib_stubs.c** as shown in Figure 26.

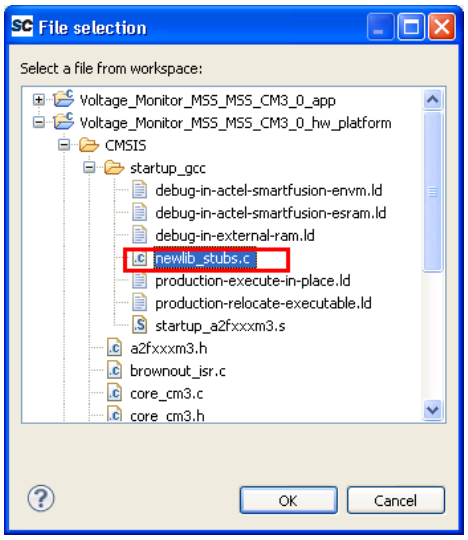
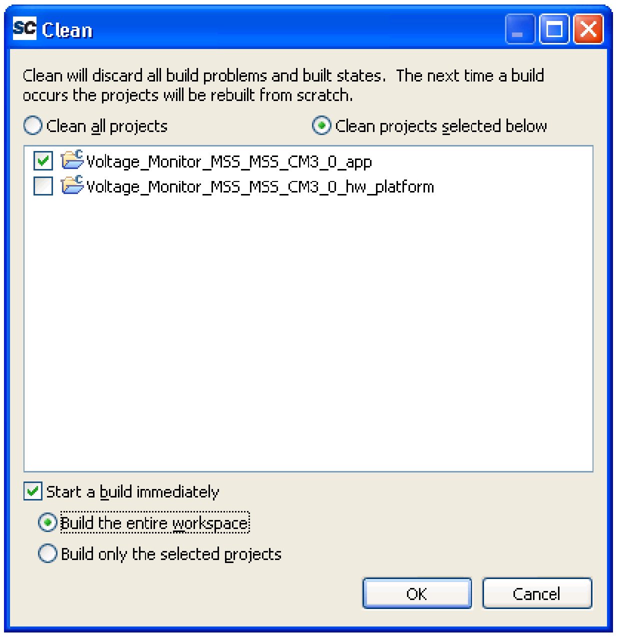

File selection
Select a file from workspace:
Voltage_Monitor_MSS_MSS_CM3_0_app
Voltage_Monitor_MSS_MSS_CM3_0_hw_platform
CMSIS
startup_gcc
debug-in-actel-smartfusion-envm.ld
debug-in-actel-smartfusion-esram.ld
debug-in-external-ram.ld
newlib_stubs.c
production-execute-in-place.ld
production-relocate-executable.ld
startup_a2fxxxm3.s
a2fxxxm3.h
brownout_isr.c
core_cm3.c
core_cm3.h
OK
Cancel

Figure 26 File Selection Window

10. Click **OK** in the properties window.
11. Perform a clean build by selecting **Project** > **Clean**. Accept the default settings in the **Clean** dialog box and click **OK**.


Clean

Clean will discard all build problems and built states. The next time a build occurs the projects will be rebuilt from scratch.

Clean all projects [✓ Clean projects selected below]

- [✓] Voltage_Monitor_MSS_MSS_CM3_0_app
- [ ] Voltage_Monitor_MSS_MSS_CM3_0_hw_platform

[✓] Start a build immediately

[✓ Build the entire workspace] Build only the selected projects

Figure 27 Settings for a Clean Build

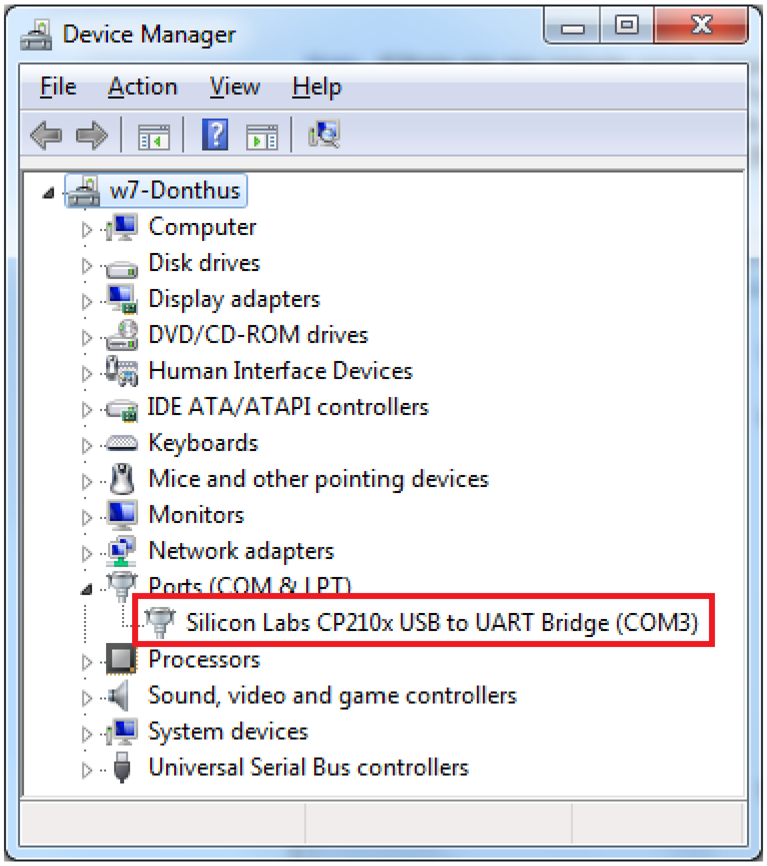
12. Make sure there are no errors and warnings.

Note: If there are any compilation errors regarding the file or directory missing then use the SoftConsole directories including feature from **Project options > C/C++ Build > settings > GNU C Compiler > Directories** to include the header file directories. This enables the compiler to look in to added directories for the included header files.

Step 5 - 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 computer. Perform the following steps to use the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board:

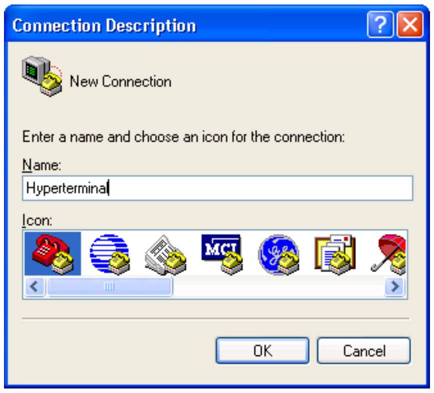
1. Connect a second mini USB cable between the USB connector on the SmartFusion Evaluation Kit Board (or the SmartFusion Development Kit Board) and a USB port of your computer. If Windows prompts you to connect to Windows Update, select No, not at this time and click Next.
2. If the Silicon Labs CP210x USB to UART Bridge drivers are automatically detected (this can be verified in Device Manager), as shown in Figure 28, proceed to next step; otherwise follow the Step 6 - Installing Drivers for the USB to RS232 Bridge.



w7-Donthus
- Computer
- Disk drives
- Display adapters
- DVD/CD-ROM drives
- Human Interface Devices
- IDE ATA/ATAPI controllers
- Keyboards
- Mice and other pointing devices
- Monitors
- Network adapters
- Ports (COM & LPT)
  - Silicon Labs CP210x USB to UART Bridge (COM3)
- Processors
- Sound, video and game controllers
- System devices
- Universal Serial Bus controllers

Figure 28 Device Manager Listing Silicon Labs CP210x USB to UART Bridge Drivers

3. From the Windows **Start** menu, select **Programs > Accessories > Communications > HyperTerminal**. This opens HyperTerminal. If your computer 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.
4. Enter Hyperterminal in the Name field in the **Connection Description** dialog box and click **OK**.


Connection Description

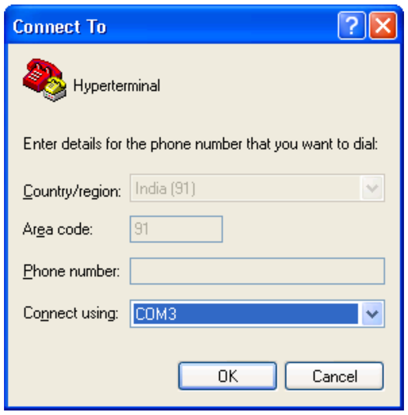
New Connection

Enter a name and choose an icon for the connection:

Name: Hyperterminal

Figure 29 New Connection

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


Connect To

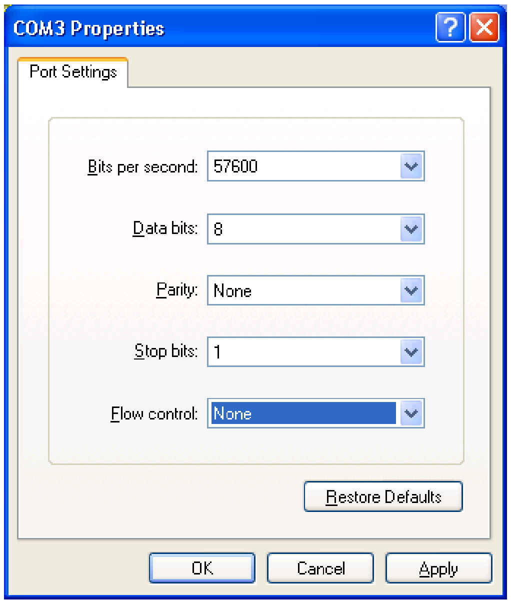
Hyperterminal

Enter details for the phone number that you want to dial:

Country/region: India (91)
Area code: 91
Phone number:
Connect using: COM3

Figure 30 Selecting the COM Port

6. 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



Bits per second: 57600
Data bits: 8
Parity: None
Stop bits: 1
Flow control: None

Figure 31 Setting the COM Properties

7. 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 6 - Installing Drivers for the USB to RS232 Bridge

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

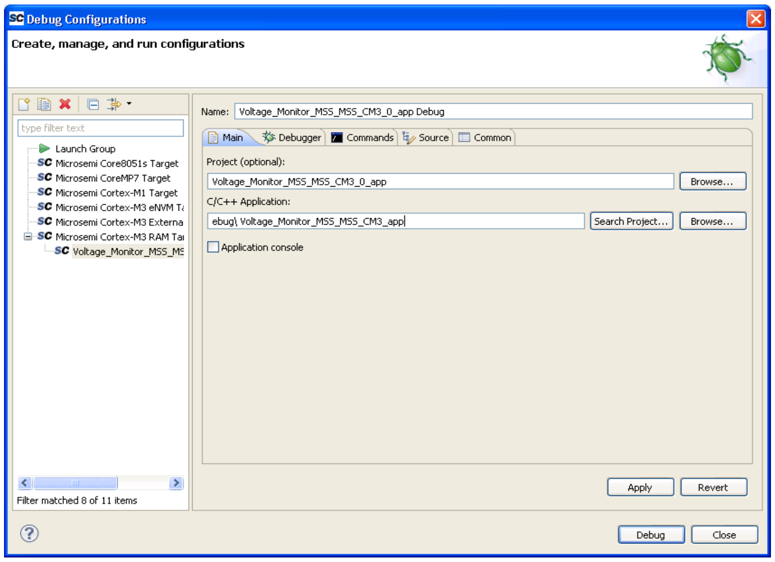
Use the following steps to install drivers for the USB to RS232 Bridge:

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 7 - Running the Application

Follow the steps given below to debug the application project using SoftConsole:

1. Select **Debug Configurations** from the **Run** menu of the SoftConsole. The Debug dialog is displayed.
2. Double click on **Microsemi Cortex-M3 RAM target**. The **Debug Configurations** window is displayed, as shown in Figure 32.


Debug Configurations
Create, manage, and run configurations
type filter text
Launch Group
Microsemi Core8051s Target
Microsemi CoreMP7 Target
Microsemi Cortex-M1 Target
Microsemi Cortex-M3 eNVM Ta
Microsemi Cortex-M3 Externa
Microsemi Cortex-M3 RAM Tar
Voltage_Monitor_MSS_MS
Filter matched 8 of 11 items
Main
Debugger
Commands
Source
Common
Browse...
Search Project...
Apply
Revert
Debug
Close

Name: Voltage_Monitor_MSS_MSS_CM3_0_app Debug

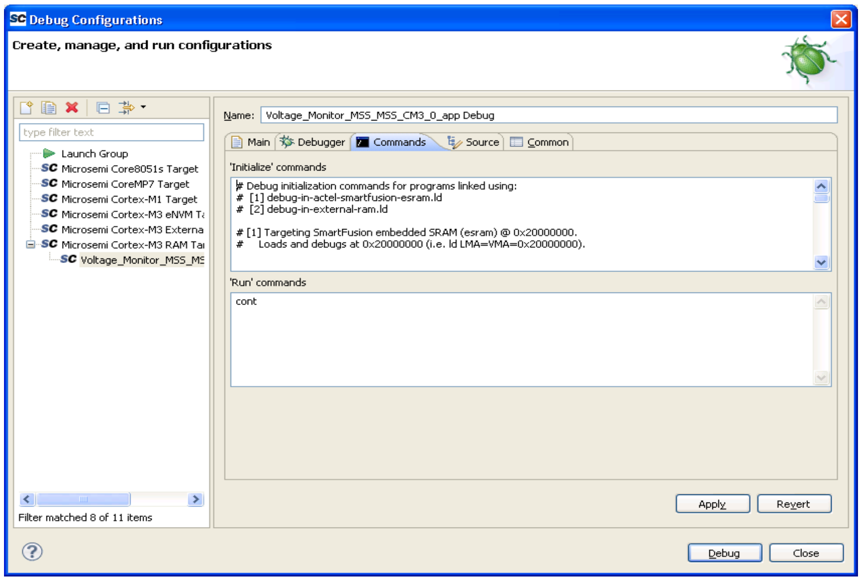
Project (optional): Voltage_Monitor_MSS_MSS_CM3_0_app

C/C++ Application: ebug\ Voltage_Monitor_MSS_MSS_CM3_app

☐ Application console

Figure 32 Debug Configurations Window

3. Confirm that the following appear on the **Main** tab in the Debug window:
   - Name: Voltage_Monitor_MSS_MSS_CM3_0_app Debug
   - Project: Voltage_Monitor_MSS_MSS_CM3_0_app
   - C/C++ application: Debug\ Voltage_Monitor_MSS_MSS_CM3_0_app
4. Select the **Commands** tab. Confirm that commands appear in the Initialize and Run command sections, as shown in Figure 33.


Debug Configurations
Create, manage, and run configurations
type filter text
Launch Group
Microsemi Core8051s Target
Microsemi CoreMP7 Target
Microsemi Cortex-M1 Target
Microsemi Cortex-M3 eNVM Ta
Microsemi Cortex-M3 Externa
Microsemi Cortex-M3 RAM Tar
Voltage_Monitor_MSS_MS
Filter matched 8 of 11 items
Main
Debugger
Commands
Source
Common
Apply
Revert
Debug
Close

Name: Voltage_Monitor_MSS_MSS_CM3_0_app Debug

'Initialize' commands

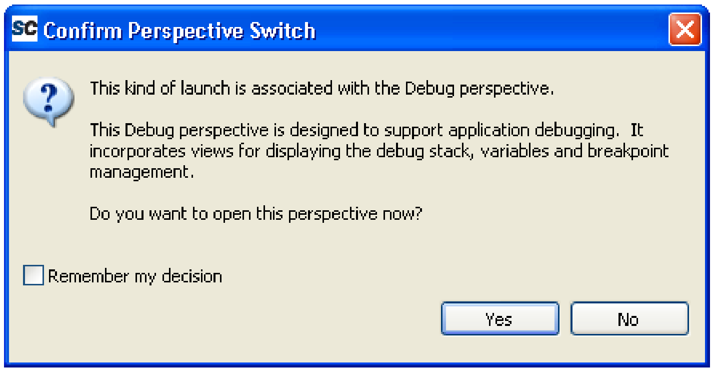
```
# Debug initialization commands for programs linked using:
# [1] debug-in-actel-smartfusion-esram.ld
# [2] debug-in-external-ram.ld

# [1] Targeting SmartFusion embedded SRAM (esram) @ 0x20000000.
#     Loads and debugs at 0x20000000 (i.e. ld LMA=VMA=0x20000000).
```

'Run' commands

```
cont
```

Figure 33 Debugger Commands

5. Click **Apply** and **Debug**.
6. Click **Yes** when prompted for **Confirm Perspective Switch**. This displays the debug view mode.



Confirm Perspective Switch

This kind of launch is associated with the Debug perspective.

This Debug perspective is designed to support application debugging. It incorporates views for displaying the debug stack, variables and breakpoint management.

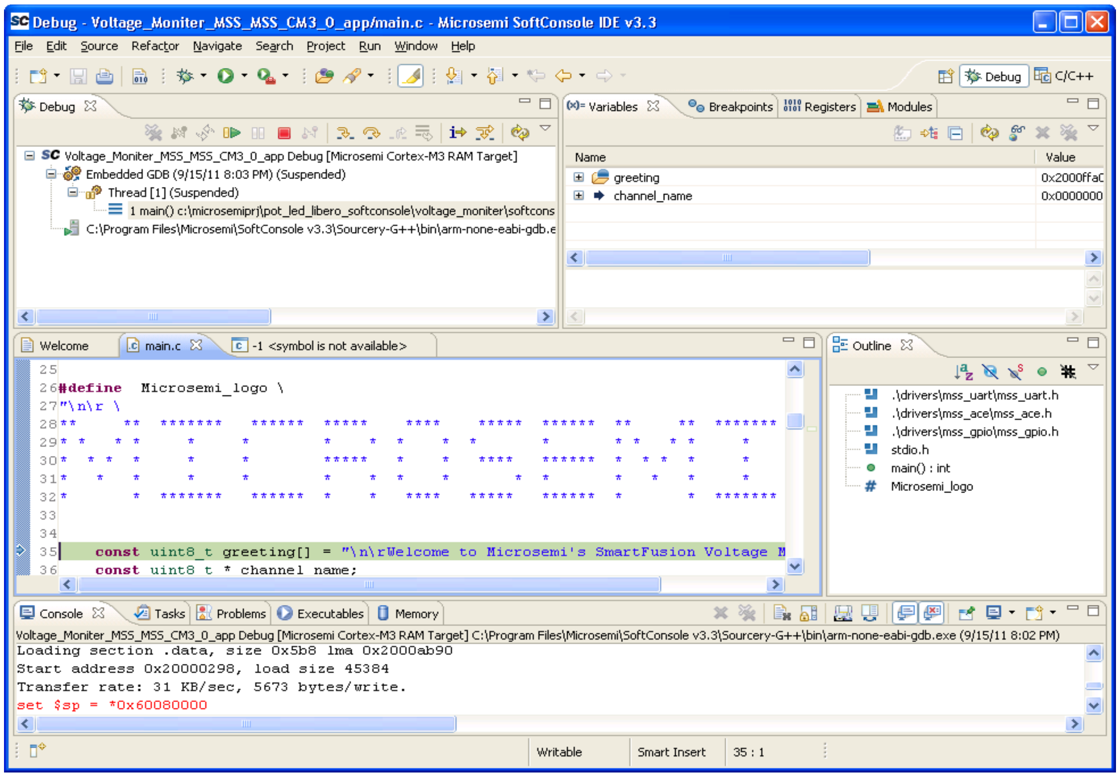
Do you want to open this perspective now?

☐ Remember my decision

Yes | No

Figure 34 Confirm Perspective Switch

7. The Debug Prospective window is displayed, as shown in Figure 35.



Debug - Voltage_Moniter_MSS_MSS_CM3_0_app/main.c - Microsemi SoftConsole IDE v3.3
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
C/C++
Debug
Variables
Breakpoints
Registers
Modules
Welcome
main.c
-1 <symbol is not available>
Outline
Console
Tasks
Problems
Executables
Memory

Voltage_Moniter_MSS_MSS_CM3_0_app Debug [Microsemi Cortex-M3 RAM Target]
Embedded GDB (9/15/11 8:03 PM) (Suspended)
Thread [1] (Suspended)
1 main() c:\microsemiprj\pot_led_libero_softconsole\voltage_moniter\softcons
C:\Program Files\Microsemi\SoftConsole v3.3\Sourcery-G++\bin\arm-none-eabi-gdb.e

Name	Value
greeting	0x2000ffaC
channel_name	0x0000000

```

#define  Microsemi_logo \
"\n\r \
**     **  *******   ******  *****    ****    *****  ******  **     **  *******
* *   * *     *     *        *    *  *    *  *       *       * *   * *     *
*  * *  *     *     *        *****   *    *   ****   ******  *  * *  *     *
*   *   *     *     *        *    *  *    *       *  *       *   *   *     *
*       *  *******   ******  *    *   ****   *****   ******  *       *  *******


    const uint8_t greeting[] = "\n\rWelcome to Microsemi's SmartFusion Voltage M
    const uint8_t * channel_name;
```

.\drivers\mss_uart\mss_uart.h
.\drivers\mss_ace\mss_ace.h
.\drivers\mss_gpio\mss_gpio.h
stdio.h
main() : int
Microsemi_logo

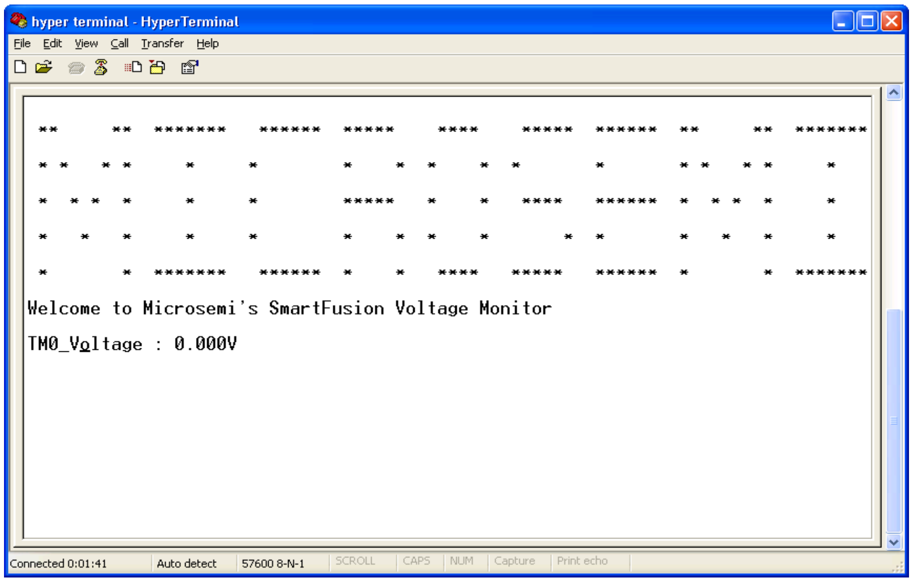
Voltage_Moniter_MSS_MSS_CM3_0_app Debug [Microsemi Cortex-M3 RAM Target] C:\Program Files\Microsemi\SoftConsole v3.3\Sourcery-G++\bin\arm-none-eabi-gdb.exe (9/15/11 8:02 PM)

```
Loading section .data, size 0x5b8 lma 0x2000ab90
Start address 0x20000298, load size 45384
Transfer rate: 31 KB/sec, 5673 bytes/write.
set $sp = *0x60080000
```

Writable | Smart Insert | 35 : 1

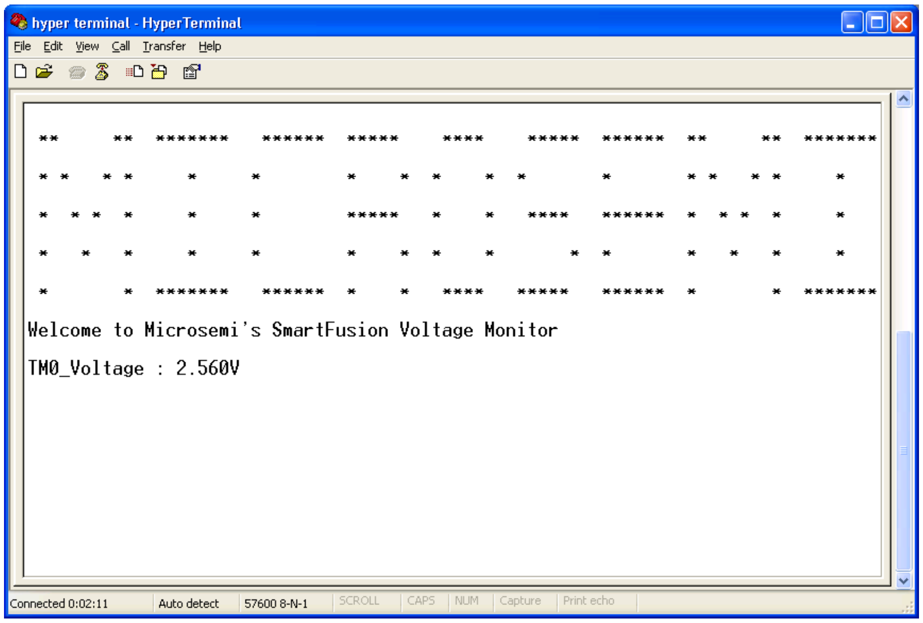
Figure 35 Debug Perspective

8. Run the application by clicking **Run > Resume** or by clicking the **Run** icon on the SoftConsole toolbar. The voltage measurement along with the greeting message is displayed in the terminal program window.
9. Turn the potentiometer (POT) on the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board. The voltage measurement is displayed on HyperTerminal and the LEDs on the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board will illuminate when one of the voltage monitor flags is asserted.
10. Adjust the POT and observe that the voltage measurement is continuously updated.



```
Welcome to Microsemi's SmartFusion Voltage Monitor
TM0_Voltage : 0.000V
```

Figure 36 Voltage Monitor



```
Welcome to Microsemi's SmartFusion Voltage Monitor
TM0_Voltage : 2.560V
```

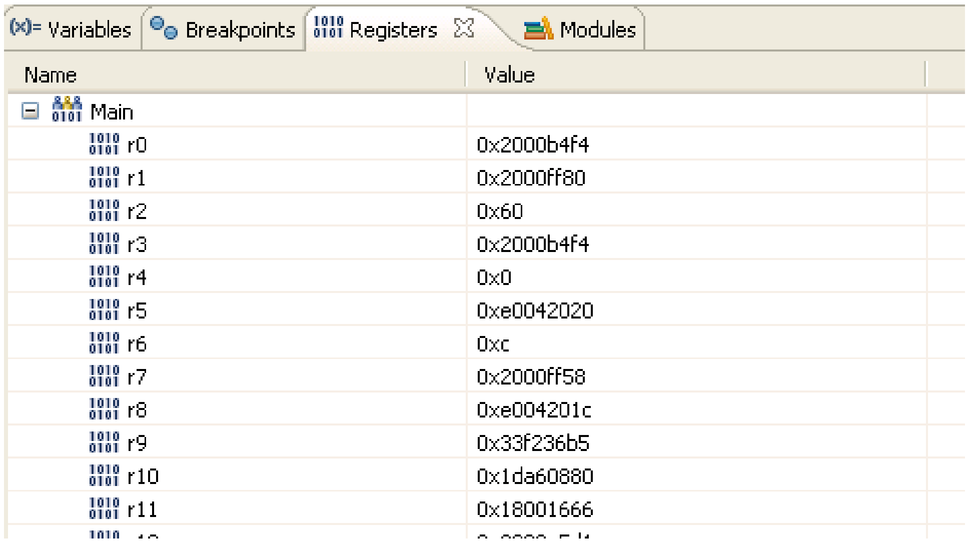
Figure 37 Voltage Measurement Continuous Update

11. Observe the state of the LEDs as the POT is adjusted. Confirm that the flags work as specified in the ACE configurator.
12. Suspend the software application by clicking **Run > Suspend** from the SoftConsole menu.

Step 8 - Debugging the Application

Use the following steps to further debug the application:

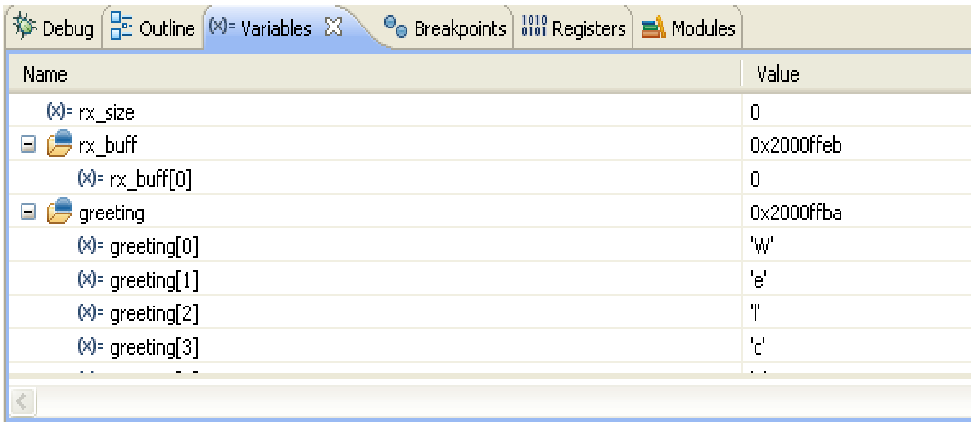
1. Select the **Registers** tab on the upper right window pane to view the value of the Cortex-M3 processor internal registers.



Name	Value
Main	
r0	0x2000b4f4
r1	0x2000ff80
r2	0x60
r3	0x2000b4f4
r4	0x0
r5	0xe0042020
r6	0xc
r7	0x2000ff58
r8	0xe004201c
r9	0x33f236b5
r10	0x1da60880
r11	0x18001666

Figure 38 The Registers Tab

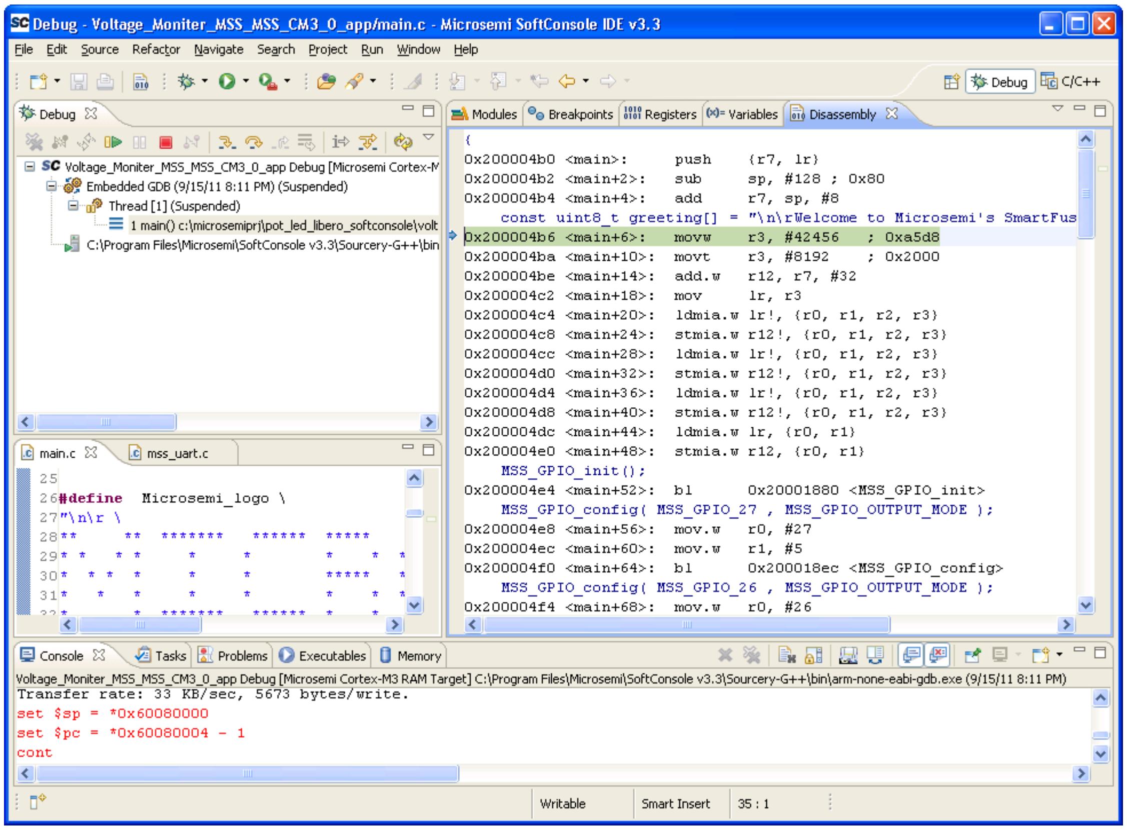
2. Select the **Variables** tab in the upper left window pane to view the value of variables in the source code.



Name	Value
rx_size	0
rx_buff	0x2000ffeb
rx_buff[0]	0
greeting	0x2000ffba
greeting[0]	'W'
greeting[1]	'e'
greeting[2]	'l'
greeting[3]	'c'

Figure 39 The Variables Tab

3. In the **Debug** window, select **Window > Show View > Disassembly** to display the assembly level instructions. The Assembly window opens on the right-side of the Debug perspective.



Debug - Voltage_Moniter_MSS_MSS_CM3_0_app/main.c - Microsemi SoftConsole IDE v3.3

```
Voltage_Moniter_MSS_MSS_CM3_0_app Debug [Microsemi Cortex-M
  Embedded GDB (9/15/11 8:11 PM) (Suspended)
    Thread [1] (Suspended)
      1 main() c:\microsemiprj\pot_led_libero_softconsole\volt
  C:\Program Files\Microsemi\SoftConsole v3.3\Sourcery-G++\bin
```

main.c

```
#define  Microsemi_logo \
"\n\r \
**      **   *******    ******   *****
* *    * *      *        *        *     *   *
*  * *  *       *        *        *****     *
*    *   *      *        *        *     *   *
```

Disassembly

```
{
0x200004b0 <main>:     push    {r7, lr}
0x200004b2 <main+2>:   sub     sp, #128 ; 0x80
0x200004b4 <main+4>:   add     r7, sp, #8
    const uint8_t greeting[] = "\n\rWelcome to Microsemi's SmartFus
0x200004b6 <main+6>:   movw    r3, #42456   ; 0xa5d8
0x200004ba <main+10>:  movt    r3, #8192    ; 0x2000
0x200004be <main+14>:  add.w   r12, r7, #32
0x200004c2 <main+18>:  mov     lr, r3
0x200004c4 <main+20>:  ldmia.w lr!, {r0, r1, r2, r3}
0x200004c8 <main+24>:  stmia.w r12!, {r0, r1, r2, r3}
0x200004cc <main+28>:  ldmia.w lr!, {r0, r1, r2, r3}
0x200004d0 <main+32>:  stmia.w r12!, {r0, r1, r2, r3}
0x200004d4 <main+36>:  ldmia.w lr!, {r0, r1, r2, r3}
0x200004d8 <main+40>:  stmia.w r12!, {r0, r1, r2, r3}
0x200004dc <main+44>:  ldmia.w lr, {r0, r1}
0x200004e0 <main+48>:  stmia.w r12, {r0, r1}
    MSS_GPIO_init();
0x200004e4 <main+52>:  bl      0x20001880 <MSS_GPIO_init>
    MSS_GPIO_config( MSS_GPIO_27 , MSS_GPIO_OUTPUT_MODE );
0x200004e8 <main+56>:  mov.w   r0, #27
0x200004ec <main+60>:  mov.w   r1, #5
0x200004f0 <main+64>:  bl      0x200018ec <MSS_GPIO_config>
    MSS_GPIO_config( MSS_GPIO_26 , MSS_GPIO_OUTPUT_MODE );
0x200004f4 <main+68>:  mov.w   r0, #26
```

Console

```
Voltage_Moniter_MSS_MSS_CM3_0_app Debug [Microsemi Cortex-M3 RAM Target] C:\Program Files\Microsemi\SoftConsole v3.3\Sourcery-G++\bin\arm-none-eabi-gdb.exe (9/15/11 8:11 PM)
Transfer rate: 33 KB/sec, 5673 bytes/write.
set $sp = *0x60080000
set $pc = *0x60080004 - 1
cont
```

Figure 40 Assembly Window

4. You can single-step through the source code by choosing **Run > Step Into** or **Run > Step Over** or by clicking the **Step Into** or **Step Over** icons. Observe the changes in the source code window and Disassembly view. Performing a Step Over allows for stepping over functions. The entire function is executed but there is no need to single step through each instruction contained in the function.
5. Click the **Instruction Stepping** icon and then perform **Step Into** operations. Observe that **Step Into** now executes a single line of assembly code.
6. Click the **Instruction Stepping** icon to exit the instruction stepping mode. Single-step through the application and observe the instruction sequence in the source code window in the middle of the Debug perspective, and the values of the variables and registers.
7. Resume execution of the code by choosing **Run > Resume** or by clicking the **Resume** icon.
8. You can even add breakpoints in the application for further debugging.
9. Once you are done, terminate the debugger by selecting Voltage_Monitor_MSS_MSS_CM3_0_app Debug in the Debug view, then right-clicking and selecting **Terminate and Remove**.

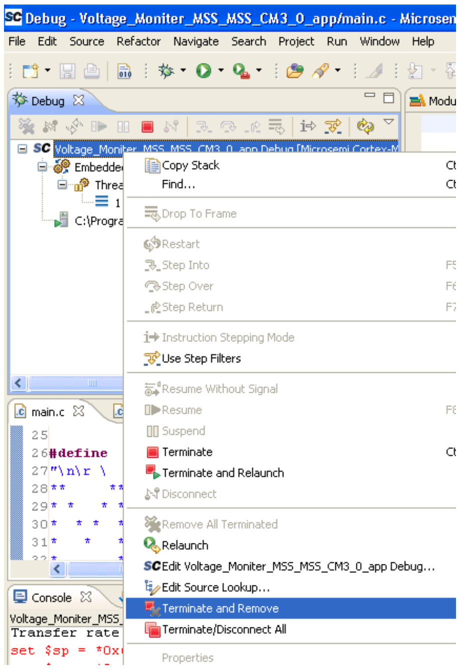


Figure 41 Terminating the Program

10. Close the Debug perspective by selecting **Close Perspective** from the **Window** menu.
11. Close the voltage monitor project by selecting the project name in the SoftConsole Project Explorer view, right–clicking, and selecting **Close Project**.
12. Close SoftConsole using **File > Exit**.
13. Close HyperTerminal using **File > Exit**. Click **Yes,** when prompted for closing immediately.

Step 9 - Building an Executable Image in Release mode

You can build an application executable image in “release mode” and load it into eNVM for executing code in eNVM of SmartFusion cSoC device. You can load the application executable image into eNVM with the help of eNVM data storage client from SmartDesign MSS Configurator and IAP or FlashPro programming software. In the release mode, you cannot use the SoftConsole debugger to load the executable image into eNVM.

For steps to build an executable image for our application refer the tutorial *SmartFusion: Building Executable Image in Release Mode and Loading into eNVM*.

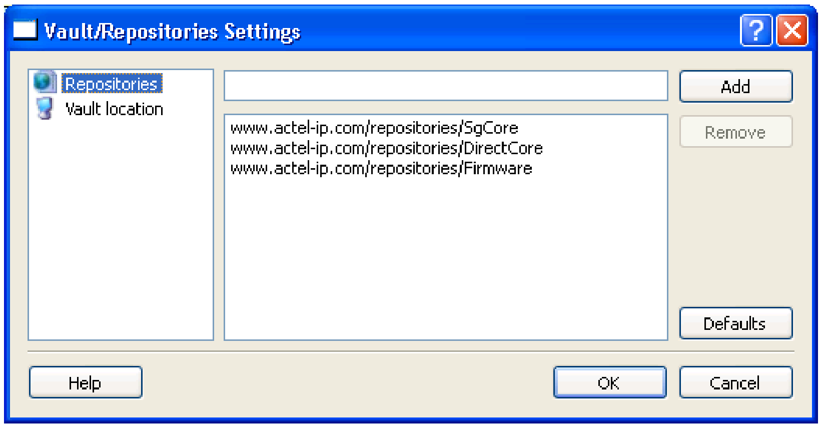
This concludes the tutorial.

Appendix A – Libero SoC Vault/Repository Settings

Listed below are the steps to show how to configure your vault location and set up the repositories in Libero SoC.

1. Click **Project > Vault/Repositories Settings**.
2. The **Vault/Repositories Settings** window is displayed. Click **Repositories** and 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

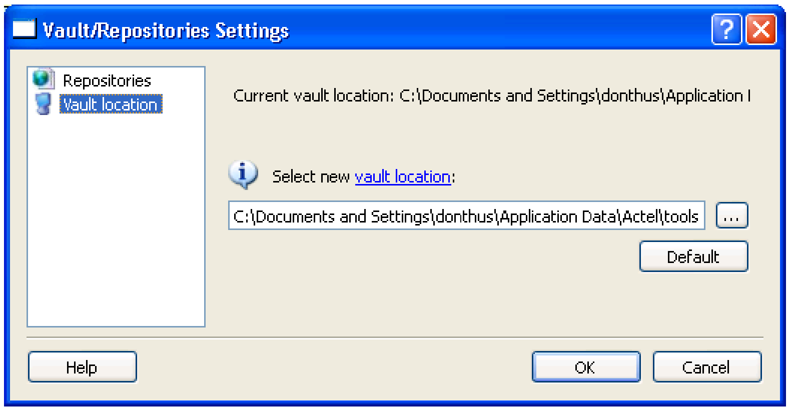
   Note: Click **Add** after entering each path.



www.actel-ip.com/repositories/SgCore
www.actel-ip.com/repositories/DirectCore
www.actel-ip.com/repositories/Firmware

Figure 42 Setting Repositories

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



Current vault location: C:\Documents and Settings\donthus\Application I

Select new vault location:

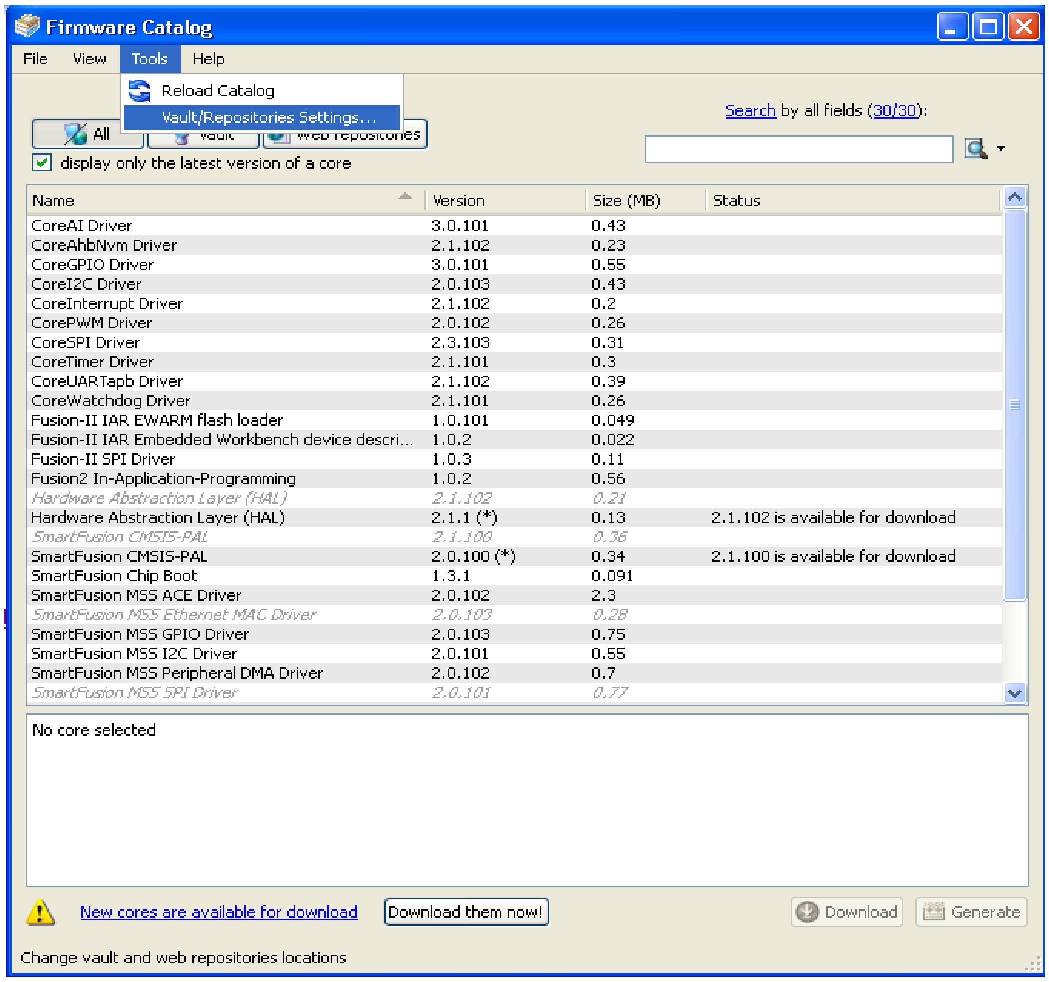
C:\Documents and Settings\donthus\Application Data\Actel\tools

Figure 43 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** widow.


Firmware Catalog
File
View
Tools
Help
Reload Catalog
Vault/Repositories Settings...
All
Vault
Web repositories
Search by all fields (30/30):
display only the latest version of a core
New cores are available for download
Download them now!
Download
Generate

Name	Version	Size (MB)	Status
CoreAI Driver	3.0.101	0.43	
CoreAhbNvm Driver	2.1.102	0.23	
CoreGPIO Driver	3.0.101	0.55	
CoreI2C Driver	2.0.103	0.43	
CoreInterrupt Driver	2.1.102	0.2	
CorePWM Driver	2.0.102	0.26	
CoreSPI Driver	2.3.103	0.31	
CoreTimer Driver	2.1.101	0.3	
CoreUARTapb Driver	2.1.102	0.39	
CoreWatchdog Driver	2.1.101	0.26	
Fusion-II IAR EWARM flash loader	1.0.101	0.049	
Fusion-II IAR Embedded Workbench device descri...	1.0.2	0.022	
Fusion-II SPI Driver	1.0.3	0.11	
Fusion2 In-Application-Programming	1.0.2	0.56	
Hardware Abstraction Layer (HAL)	*2.1.102*	*0.21*	
Hardware Abstraction Layer (HAL)	2.1.1 (*)	0.13	2.1.102 is available for download
SmartFusion CMSIS-PAL	*2.1.100*	*0.36*	
SmartFusion CMSIS-PAL	2.0.100 (*)	0.34	2.1.100 is available for download
SmartFusion Chip Boot	1.3.1	0.091	
SmartFusion MSS ACE Driver	2.0.102	2.3	
SmartFusion MSS Ethernet MAC Driver	*2.0.103*	*0.28*	
SmartFusion MSS GPIO Driver	2.0.103	0.75	
SmartFusion MSS I2C Driver	2.0.101	0.55	
SmartFusion MSS Peripheral DMA Driver	2.0.102	0.7	
SmartFusion MSS SPI Driver	*2.0.101*	*0.77*	

No core selected

Change vault and web repositories locations

Figure 44 Firmware Catalog Settings

3. Select **Repositories** in the **Vault/Repositories Settings** 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**.
6. If new cores are available for download, click **Download them now!** to download the new cores to the vault.

List of Changes

Revision	Changes	Page
Revision 4 (February 2012)	Modified Associated Project Files section. (SAR 36896)	3
	Modified Step 4 - Building the Project section. (SAR 36896)	17
	Modified Step 5 - Configuring the Serial Terminal Emulation Program section. (SAR 36896)	24
	Updated Figure 28. (SAR 36896)	24
	Modified Step 6 - Installing Drivers for the USB to RS232 Bridge section. (SAR 36896)	26
	Modified Step 7 - Running the Application section. (SAR 36896)	26
Revision 3 (November 2011)	Updated the document for Libero SoC v10.0 (SAR 35046).	NA

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 **650. 318.8044**

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, common design cycle questions, 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 (www.microsemi.com/soc/support/search/default.aspx) for more information and support. 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 dropdown list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web page.

Microsemi Corporate Headquarters
One Enterprise Drive, Aliso Viejo CA 92656
Within the USA: (800) 713-4113
Outside the USA: (949) 221-7100
Fax: (949) 756-0308 · www.microsemi.com

50200212-4/02.12