Displaying POT Level with LEDs

Libero SoC and SoftConsole Flow Tutorial for a SmartFusion cSoC



Table of Contents

Introduction	3
Tutorial Requirements	
Objective	
Working with Libero SoC and SoftConsole	5
Step 1 - Creating a Libero SoC Project	5
Step 2 - Configuring MSS Peripherals	7
Step 3 - Generating the MSS Component	15
Step 4 - Generating the Program File	17
Step 5 - Programming SmartFusion Board Using FlashPro	19
Step 6 - Building the Software Application Through SoftConsole	21
Step 7 - Configuring Serial Terminal Emulation Program	26
Step 8 - Installing Drivers for the USB to RS232 Bridge	28
Step 9 - Debugging the Application Project using SoftConsole	29
Step 10 - Building Executable Image in Release mode	35
Appendix A – Libero SoC Catalog Settings	37
Appendix B – Firmware Catalog Settings	39
Appendix – C	41
List of Changes	49
Product Support	51
Customer Service	51
Customer Technical Support Center	51
Technical Support	51
Website	51
Contacting the Customer Technical Support Center	51
ITAR Technical Support	



Introduction

This tutorial shows you how to use Microsemi tools to develop an application that can be implemented on 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 SmartFusion cSoC device.
- Configure 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 SoftConsole from Libero SoC and writing application code.
- Compiling application code.
- · Creating and launching a debug session.
- Debugging and running the code using SoftConsole.

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.
- Microsemi SoftConsole v3.3 (or later), which is installed as a part of Libero SoC installation or can be downloaded from www.microsemi.com/soc/download/software/softconsole/default.aspx.

Hardware Requirements

You will need 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_LiberoSC_POTlevel_tutorial_DF

You can download the programming file (*.stp) in release for this tutorial from the Microsemi website: www.microsemi.com/soc/download/rsc/?f=SmartFusion_LiberoSC_POTlevel_tutorial_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 instruct how to configure SmartFusion cSoC analog channels and ACE that is used to monitor the voltage across the potentiometer (POT). The UART is used to send the ADC results to a terminal program.

Design Steps

- Create a Libero SoC project and use the SmartFusion cSoC MSS Configurator to configure the ACE, adding a voltage monitor with flags.
- Perform synthesis and layout, and generate a programming file to program the SmartFusion cSoC device.
- Open the software project in SoftConsole and write the application program.
- Run an application to monitor the voltage across the POT on the SmartFusion Evaluation Kit Board or the SmartFusion 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 SoftConsole

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 SoftConsole 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_SoftConsole)
 - 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

Leave others as default.

New Project				?
Project				
Name:	Voltage_Monitor			
Location:	C:/Microsemipri/POT_LED_Libero	_SoftConsole		Browse
Prefered HDL type:				
Description:				
Edit Tool Profiles				
Device				
Family:	SmartFusion			
Die:	A2F200M3F			
Package:	484 FBGA 🗸			
Speed:	STD 💌			
Die Voltage:	1.5 💙			
Operating Conditions:	сом 💌			
Temper	rature (in degrees Celsius)	Best 0	Typical 25	Worst 85
VCCA V VCCI 1	/oltage (in volts) .5 Voltage (in volts)	1.575 1.6	1.5 1.5	1.425 1.4
VCCI 1	.8 Voltage (in volts) .5 Voltage (in volts)	1.9 2.7	1.8 2.5	1.7 2.3
	.3 Voltage (in volts)	3.6	3.3	3
Design Template				
Use template				
	Core			Version
SmartFusion Microcontro	ller Subsystem (MSS)			2.5.106
			🗹 Sł	now only latest version
Help			_	OK Cancel

Figure 1 · New project Dialog Box



O Tool Profiles				? 🛛
■ Tools Software IDE Synthesis Simulation Programming	Software ID	E profiles Name	Path	
	Add Profile Name: Tool integration: Version: Location:	SoftConsole SoftConsole Unknown psemi\SoftConsole v3.3\Eclip	se\eclipse.exe Browse	Porkbench 6.0
Help			OK Cancel).:: OK Cancel

3. Click Edit Tool Profiles and add SoftConsole by clicking on Software IDE as shown in Figure 2 · .

Figure 2 · Selecting SoftConsole as Software IDE

4. After adding the Profile, click **OK** to close the **Add Profile** dialog window.

Repeat the steps (3 and 4) above for Synthesis, Simulation, and Programming and then click **OK** to close the Tool Profiles dialog window.

- 5. Select the MSS core in New Project dialog box and click OK.
- 6. The project is created and the Libero SoC window appears as shown in Figure 3 · . The SmartDesign "Voltage_Monitor" is created with the instantiation of MSS component.

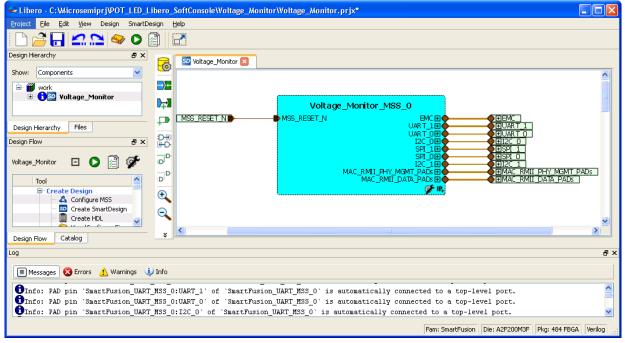


Figure 3 · The Libero Window After Completing New Project Wizard

Step 2 - Configuring MSS Peripherals

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

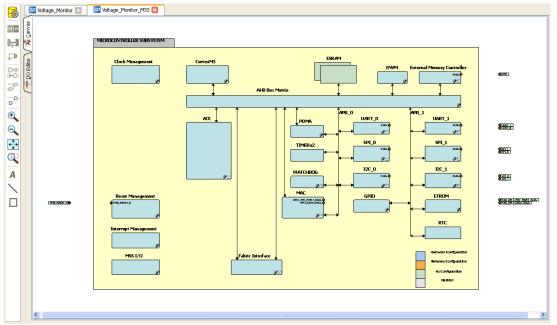


Figure 4 · MSS in the SmartDesign Canvas

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

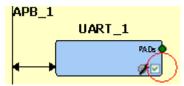


Figure 5 · Enabling the Peripheral

This example uses only the clock management, ACE, GPIO, and UART_0 peripherals.



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

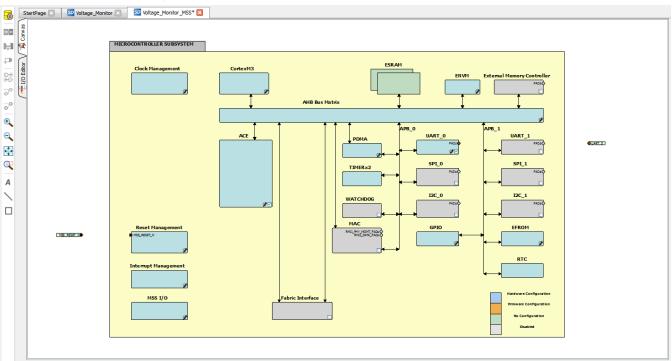


Figure 6 · Used MSS Peripherals

- 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.

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



Microcontroller Subsystem Clock(s) Configurator	
CLKA /20 PEPCLK Control Actual data is shown in blue. CLKA /20 PEPCLK MSS_LOCK CON-Chip RC Oscillator BLL 0.919 ns /16 0.919 ns /16 0.919 ns /16 0.919 ns /16 0.000 MHz /0.005 ms / 2.911 ns /1756	FCIX /I V ACUK (ACE) B0.000 MHz /I V PCUK0 (AP8_0) 80.000 MHz /I V PCUK1 (AP8_1) 80.000 MHz
CLKB	
	External 10 /100 CLK
On-Chip RC Oscillator 100.000 MHz Main Crystal Oscillator 20.000 MHz Low-Power 32 KHz Crystal Oscillator 0.032 MHz Clock Conditionina	Microcontroller Subsystem Clock Conditioning ok Cancei

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 the SmartFusion 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 the SmartFusion Development Kit Board.

Configure ACE							
Config ADC Config	ure ACE	Controller	Flags				
ACL <u>K</u> :	80 MHz	ADC Clock: 1	0 MHz Resolution:	12 v bits		Advanced Opt	ions
Add service							
Differential Temperatu LVTTL Inpu Analog Con Sigma Delta	re Input t nparator	Service		Signal	Sample time (us)	Package Pin	\$CB
		ADC Block 0 ADC Block 1					
						ок	Cancel

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
- 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

Configure ADC Dire	ect Input						
тмо	<u>S</u> ignal na	ame:					
	TM0_j/ol	ltage		Send raw A	DC result to D	MA	
Acguisition time	: 10	.000 us					
Digital filtering				Linear trans	formation		
Filtering facto	or: Nor	ne 🔽		Scale factor	:		
Ini <u>t</u> ial value:	0.0	V 000		Offset:			
Send filter	red result to) DMA		Send trar	nsformed resu	ilt to DMA	
		Thr	eshold Detectio	n		1	×
Fla	g Name	Flag Type	Threshold (V)	Hysteresis (mV)	Assert Samples	De-assert Samples	
over_1		OVER	1	1			
over_1		OVER	1.5	1			_
over_2		OVER OVER	2.5	1			~
				I			
Assert flag v	vhen post p	rocessing complete	3				
					ОК	Can	cel

Figure 9 · MSS ADC Direct Input Configuration

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 Figure 10 · :

ïgure ACE			
Configure ACE	Controller	Flags	
ACL <u>K</u> : 80 MHz	ADC Clock: 10 MHz	Regolution: 12 vits	Advanced Options
Add service:			_/ ×
Active Bipolar Prescaler Inpu ADC Direct Input Current Input Differential Input	Service ADC Direct Input	Signal TM0_Voltage	Sample time (us) Package Pin SCB 10.000 W8 (TM0) ✓ 0(4)
Temperature İnput LVTTL Input Analog Comparator Sigma Delta DAC			
Add >>			
Edd XX			
	ADC Block 0		
	ADC Block 1		
			OK Cancel

Figure 10 \cdot 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.



Configure ACE						X
Configure ACE Procedures	Controller					
ADC0_MAIN	Name		ADC Block 0 ADC Block 1			
Operating sequence entry: Details of procedure: Available signals: Operating sequence Operating sequence	Auto Manual ADCO_MAIN Sampling rate	Signal Iotal samp	Actual Rate (ksps)	ilot		
					ОК	Cancel

Figure 11 · MSS ACE Configuration to Enable Sampling Sequence

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

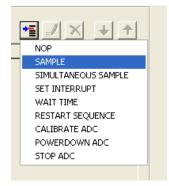


Figure 12 · Select SAMPLE

10. The **Configure SAMPLE** window is displayed. Select **TM0_voltage** as shown in Figure 13 · and click **OK**.

Configure 'SAMP	LE		×
Analog Pad: 👖	M0_Voltage		•
	Descripti	ion	
Sample Analog Peri Only the peripheral	pheral. Is that are assigned to this ADC (can be selected.	
Help		ок	Cancel



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





Figure 14 · Select Restart Sequence

12. Click Calculate Actual Rate.

nfigure ACE	X
Configure ACE Controller Flags	
Procedures Name ADC0 MAIN ADC1 MAIN ADC Block 0 ADC Block 1	
Operating sequence entry: Auto Manual Details of procedure: ADC0_MAIN Available signals: Sampling rate Available signals: Signal Actual Rate 0.000 Calculate a round robin operating sequence of the specified signals. Calculate Actual Rate Operating sequence	
Instruction SSE Execution Sample TMQ_Voltage 11.225 Restarts the execution sequence for this timestot 0.100	
ОК Салсе	

Figure 15 · MSS ACE Configuration: Final Controller Tab

13. The **Controller** tab window is displayed as shown in Figure $16 \cdot :$

ifigure ACE			
Configure ACE Contro	ler Flags		
Procedures	• • / ×		
Name		ADC Block 0	
ADC0_MAIN ADC1_MAIN		ADC Block 1	
Operating sequence entry: O Auto	Manual		
Details of procedure: ADC0_MAIN Available signals:	Sampling rate		
1.51	Signal	Actual Rate	
	TM0_Vokage	[ksps] 84.200	
<<+			
Calculate Act	Iotal sar	npling rate: 84.200 ksps	
Operating sequence			
	Instruction	SSE Execution Time (us)	
Sample TM0_Vokage Restarts the execution sequence to	e this teneslot	11.725	
Help			OK Cancel

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 userdefined 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.

Configure ACE Controller	Flag	·			
elect a register to view ACE flag mapping: - Fabric	Bit	bits of PPE_FLAG50 register:	ACE	Cortex-M3	~
Flag Registers			Interrupt	Interrupt	
		TM0_Voltage:over_1p0v	54 55	118 119	
— PPE_FLAGS1 (0x40021454)		TM0_Voltage:over_1p5v TM0_Voltage:over_2p0v	56	120	
PPE_FLAGS2 (0x40021458)		TMU_voltage:over_2puv TM0_Voltage:over_2p5v	57	120	
— PPE_FLAG53 (0x4002145c)		<pre>//mu_voitage:over_2pov //none></pre>	58	121	
PPE_SFFLAGS (0x40021460)		<none></none>	59	122	
		<none></none>	60	123	
		<none></none>	61	125	
		<none></none>	62	126	
		<none></none>	63	127	
		<none></none>	64	128	
		<none></none>	65	129	
		<none></none>	66	130	
	13	<none></none>	67	131	
		<none></none>	68	132	
		<none></none>	69	133	
		<none></none>	70	134	
		<none></none>	71	135	
		<none></none>	72	136	
		<none></none>	73	137	
		<none></none>	74	138	
		<none></none>	75	139	
		<none></none>	76	140	
		<none></none>	77	141	
		<none></none>	78	142	
		<none></none>	79	143	
		<none></none>	80	144	
		<none></none>	81	145	
		<none></none>	82	146	
	29	<none></none>	83	147	~
	, 101		114	. all	

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 $18 \cdot$, and click **OK**.

🔀 Configuring MSS_GPIO_0 (MSS_GPIO - 1.0.101)	
Configuration	^
Multiplexed With I2C1 Peripheral Dedicated I/Os	
GPIO_31: Use as MSS I/O Pad Output V20 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 V22 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: Lise as MSS I/O Pad Not Lised V AA22 or connect to Fabric Not Lised	
	OK Cancel

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 (A2F-EVAL-KIT Rev 2).
- 3. Click **File > Save** to save the Voltage_Monitor_MSS.

Step 3 - 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 19 · .

Working with Libero SoC and SoftConsole

6	SO Voltage_Monitor 🔀 SO Voltage_Monitor_MSS 🗵	
		-
⊳	Voltage_Monitor_MSS_0	
₽		
0-9 9-0		
0 0		
D	Configure Configure	
Ð	Replace Component for Instance(s) Update Instance(s) with Latest Component	
Q	Clear User Connections	
		~
×		

Figure 19 · Updating the MSS

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





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.

SD	/oltage_Moni	tor 🖂 🛛 🔤 Voltage_Monitor_f	MSS 🖂 🛛 🕺 DESIGN_FIRM	1WARE* 🗵	
	Generate	Instance Name	Core Type	Version	Compatible Hardware Instance
1	V	🜮 HAL_0	HAL	2.1.102	Voltage_Monitor_MSS
2	v	MSS_ACE_Driver_0	MSS_ACE_Driver	2.2.101	Voltage_Monitor_MSS:MSS_ACE_
3	v	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_PD	; SmartFusion_CMSIS_P/		Voltage_Monitor_MSS:MSS_DMA_
7		MSS_RT	, sinan (1 2010) [_sin515_1		Voltage_Monitor_MSS:MSS_RTC_
8		MSS_Tim Configuration —		· · · · · · · · · · · · · · · · · · ·	Voltage_Monitor_MSS:MSS_TIMEI
9	v	MSS_UA Software 1	Fool Chain SoftConsole	✓	Voltage_Monitor_MSS:MSS_UART
10		SmartFu Help -	ОК		Voltage_Monitor_MSS

Figure 21 · Configuring SmartFusion_CMSIS_PAL_0

- 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 A – Libero SoC Catalog Settings.
- 5. Click **File > Save** to Save the Design_Firmware.
- 6. Save the design and generate the component by clicking Generate Component or by selecting SmartDesign > Generate Component.

6	🕺 Volkage_Monitor* 🔀 🛛 Son Volkage_Monitor_MSS 🔣 🛛 Reports 🔛	
	Generate Component	Î
₽₽₽		
₽	Voltage_Monitor_MSS_0	
₽⊕ ⊕Đ	MSS_RESET_N MSS_RESET_N GPIO_31_OUT GPIO_31_OUT	
:D ^D	TMO_Voltage OPIO_30_OUT VAREF0 VAREF0 GPIO_29_OUT	
0 ^D	GPIO_28_OUT UART 0 ⊕	
Ð		
Q		
\Leftrightarrow		
Q		
Α		
×		

Figure 22 · Generating the MSS Component

- After successful generation of project 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.
- 8. Confirm that the SoftConsole folder is created with the folders and files as shown in Figure 23 · .

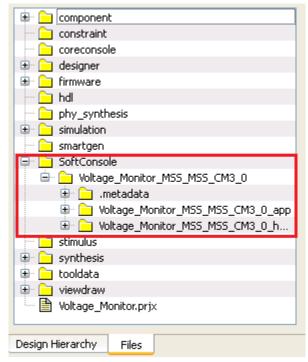


Figure 23 · Files Window

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 generating 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 Figure 24 · to complete the place and route, verify timing, and generate the programming file. This completes the fdb file generation.



Figure 24 · Generating Programming Data

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

Voltage_Monit	tor		🗆 🕒 📄 🜮
	Tool		
	⊿	►	Create Design
V			🖧 Configure MSS
			SD Create SmartDesign
			Create HDL
			Create HDL TestBench
			liew/Configure Firmware Cores
		\triangleright	Verify Pre-Synthesized Design
	\triangleright	⊁	Constrain Design
	⊿	►	Implement Design
V		\triangleright	Synthesize
		4	Verify Post-Synthesis Implementation
			Simulate
V		\triangleright	📸 Compile
		\triangleright	Constrain Place and Route
2			Place and Route
V		4	Verify Post Layout Implementation
			Simulate
v			🖏 Verify Timing
			🚯 Verify Power
			Export Back Annotated Files
V			🚓 Generate Programming Data
	⊿	•	Program Design
			🚆 Program Device
	⊳		Debug Design
	4	•	Handoff Design for Production
			Export Programming File
			Export Pin Report
			Export IBIS Model
	4	•	Develop Firmware
			Write Application Code

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 the *SmartFusion Development Kit User's Guide* for additional information.

Jumper Settings for SmartFusion Evaluation Kit Board

- JP10: Short pin 1 and 2 using a jumper.
- JP7: Short pin 1 and 2 using a jumper for LCPS mode.
- J6: Connect pin 1 and 2 using the jumper.
- JP6: Connect pin 2 and 3 using the jumper.
- J13: Connect the USB cable to J13 connector. Install the FlashPro4 or FlashPro drivers if they are not already installed.
- J14: Connect second USB cable for power.
- JP11, JP12, JP13, and JP14: Short pin 2 and 3 using a jumper (in A2F EVAL REV 2).

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:

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	_

Table 2 · Jumper Settings for Development Kit Board

Programming the Device

Double click **Program Device** under **Program Design** in the **Design Flow** window to program the SmartFusion cSoC device. Click **Yes** when it prompts that the 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 Tech Support at soc_tech@microsemi.com.



Voltage_Mo	nitor					-	0		Ý
	Tool								
	4	►	Cre	ate Design					
V				Configure M	ISS				
			_	Create Smar					
				Create HDL	-				
				Create HDL	TestBenck	h			
			S	View/Config	ure Firm	ware Cor	es		
		\triangleright	•	Verify Pre-S	ynthesiz	ed Desig	jn		
	\triangleright	۲	Со	istrain Desig	n				
	4	►	Im	plement Des	ign				
V		\triangleright	G	Synthesize					
		⊿	►	Verify Post-	Synthesi	s Implen	ientat	ion	
				🗮 Simulate	:				
v		\triangleright	Ro	Compile					
		\triangleright	•	Constrain Pl	ace and I	Route			
5			0 _{lo}	Place and R	oute				
v		4	•	Verify Post		nplemen	tation	1	
				🚆 Simulate					
V				🖄 Verify Ti	_				
				🛕 Verify Po					
				Export Back					
v				Generate P	_	ning Data	1		
	4	•	_	gram Desigi					
		J.		Program Dev	vice				
	⊳			bug Design					
	4	•		ndoff Design					
			_	Export Progr	_	File			
				Export Pin R					
				Export IBIS N					
	⊿	•	_	velop Firmw					
				Write Applic	ation Co	de			

Figure 26 · Design Flow Window

You can also run flash pro interactively by right clicking on **Program Device** in **Design Flow** window and selecting **Open Interactively**. For more information on FlashPro refer *FlashPro user's guide*.

Step 6 - Building the Software Application Through SoftConsole

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

Voltage_Monito	or		•) 🗿	Ý
	Tool				
	4	Þ	Create Design		
V			🖧 Configure MSS		
			SD Create SmartDesign		
			Create HDL		
			Create HDL TestBench		
			♀ View/Configure Firmware Cores		
		\triangleright	Verify Pre-Synthesized Design		
	\triangleright	۲	Constrain Design		
	4	Þ	Implement Design		
V		\triangleright	Synthesize		
		⊿	Verify Post-Synthesis Implement	ntation	
			Simulate		
V		\triangleright	Compile		
		\triangleright	Constrain Place and Route		
V			Place and Route		
v		4	Verify Post Layout Implementa	ation	
			Simulate Simulate		
v			🖏 Verify Timing		
			💫 Verify Power		
			🐘 Export Back Annotated Files		
V			🐘 Generate Programming Data		
***	4	►	Program Design		
V			Program Device		
	\triangleright	►	Debug Design		
	4	►	Handoff Design for Production		
			🛃 Export Programming File		
			Export Pin Report		
			Export IBIS Model		
	4	۲	Develop Firmware		
			Write Application Code		

Figure 27 · Invoking SoftConsole from Libero SoC

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 perspective looks similar to Figure 28 · :



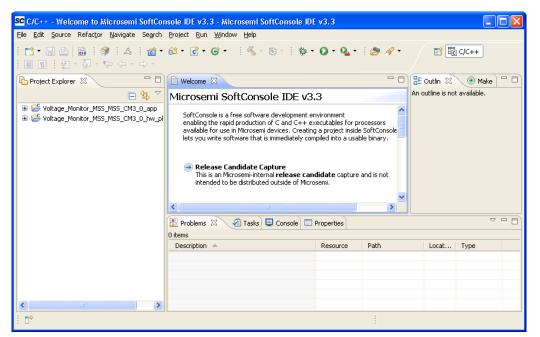


Figure 28 · SoftConsole Workspace

 Copy the code provided below and paste it in main.c file under Voltage_Monitor_MSS_MSS_CM3_0_app project in the SoftConsole editor and delete the existing 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( TMO Voltage );
for (;;)
{
   uint8 t display buffer[32];
   uint16 t adc result;
   int32 t adc value mv;
   adc result = ACE get ppe sample( TMO Voltage );
   adc value mv = ACE convert to mV( TMO 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(TMO Voltage over 2p5v);
           int32 t flag status 2p0v = ACE get flag status(TMO Voltage over 2p0v);
           int32 t flag status 1p5v = ACE get flag status(TMO Voltage over 1p5v);
           int32 t flag status 1p0v = ACE get flag status(TMO 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 as shown in Figure 29 \cdot .



SC C/C++ - Voltage_Monitor_MSS_MSS_CM File Edit Source Refactor Navigate Search	3_0_app/main.c - Microsemi SoftConsole IDE v3.3
	63 • € • € • • ∮ % • ⊗ • ∮ % • Q • Q • } Ø • ∮ Ø • ↓ I I E E c(c++
Project Explorer Project Exp	C main.c X 2 G#define Microsemi_logo \ 2 7"\n\r \ 2 8" t t t t t t t t t t t t t t t t t t
<	
8 □◆	Writable Smart Insert 54:76

Figure 29 · SoftConsole Workspace

5. Perform a clean build by selecting **Project** > **Clean**. Accept the default settings in the **Clean** dialog box and click **OK**.

SC 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 Selected below
Start a build immediately Suild the entire workspace Build only the selected projects
OK Cancel

Figure 30 · Settings for a Clean Build

6. Make sure there are no errors and warnings. Use the next steps to configure the HyperTerminal.



Step 7 - Configuring 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 31 · , proceed to the next step; otherwise follow the
- 3. Step 8 Installing Drivers for the USB to RS232 Bridge to install drivers for USB to RS232 Bridge.

A Device Manager
<u>File Action View H</u> elp
⊿ 🛃 w7-Donthus
Image: Second
Disk drives
🔉 📲 Display adapters
DVD/CD-ROM drives
Human Interface Devices
De 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
July System devices
D - Universal Serial Bus controllers

Figure 31 · 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 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.
- 5. 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:	
<u>N</u> ame:	
Hyperterminal	
lcon:	
	2
OK Car	ncel

Figure 32 · 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**.

Connect To 🔹 🤶 🔀					
Nyperterminal					
Enter details for	the phone number that you want to dial:				
<u>C</u> ountry/region:	India (91)				
Ar <u>e</u> a code:	91				
<u>P</u> hone number:					
Co <u>n</u> nect using:	СОМЗ				
	OK Cancel				

Figure 33 \cdot 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

COM3 Properties		?
Port Settings		
<u>B</u> its per second:	57600	~
<u>D</u> ata bits:	8	~
<u>P</u> arity:	None	*
<u>S</u> top bits:	1	~
<u>F</u> low control:	None	~
	<u>R</u> estr	ore Defaults
	K Cancel	Apply

Figure 34 · Setting the COM Properties

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: You must have full administrative rights for your system to install the USB-RS232 drivers.

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.

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

Step 9 - Debugging the Application Project using SoftConsole

Use the following steps 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 clicking on Microsemi Cortex-M3 RAM target displays an image similar to Figure 35 · :

SC Debug Configurations		
Create, manage, and run confi	gurations	Ť
Image: Section 2014 Image: Section 2014 Image: Section 2014 Image: Section 2014 <td>Name: Voltage_Monitor_MSS_MSS_CM3_0_app Debug Main Image: Debugger Commands Source Common Project (optional): Voltage_Monitor_MSS_MSS_CM3_0_app C/C++ Application: C/C++ Application: ebug\ Voltage_Monitor_MSS_MSS_CM3_app Search Project Search Project Application console Search Project</td> <td>Browse Browse</td>	Name: Voltage_Monitor_MSS_MSS_CM3_0_app Debug Main Image: Debugger Commands Source Common Project (optional): Voltage_Monitor_MSS_MSS_CM3_0_app C/C++ Application: C/C++ Application: ebug\ Voltage_Monitor_MSS_MSS_CM3_app Search Project Search Project Application console Search Project	Browse Browse
Filter matched 8 of 11 items	Apply	Revert
?	Debug	Close

Figure 35 · Debug 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 36 · :

SC Debug Configurations		
Create, manage, and run config	gurations	5.
Image: Section of the section of th	Name: Voltage_Monitor_MS5_MS5_CM3_0_app Debug Main Main Debugger Commands Initialize' commands Source Common Initialize' commands Image: Source Common Image: Source im	
?	Debug Clos	;e

Figure 36 · Debugger Commands

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



Figure 37 · Confirm Perspective Switch

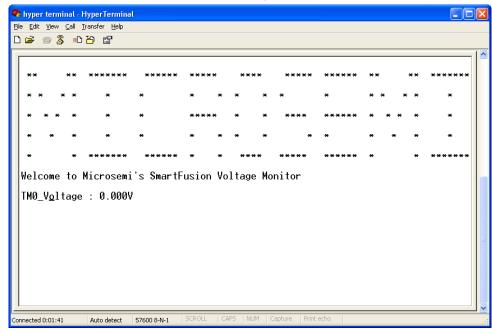
SC Debug - Voltage_Moniter_MSS_MSS_CM3_0_app/main.c - Microsemi Soft	Console IDE v	/3.3				
<u>File Edit Source Refactor Navigate Search Project Run Window Help</u>						
📬 ▾ 📰 🖮 🐘 券 ▾ O ▾ 🏊 ▾ 🥭 🖋 ▾ 🕖 ½ ▾ 🖓 ▾ 🤃					😭 🏇 Debug	€ C/C++
🏇 Debug 🕱 🖓 🖓	🗆 🕪= Variable	es 🖾 🔎 Bri	eakpoints 🚻 Re	gisters 🛋 Mod	lules	
🎇 🕼 🚸 🕪 💷 🛋 🔀 🐟 📖 👹 🖗	7			۴.	🔹 💿 🔄	× ¾ ▽
SC Voltage_Moniter_MSS_MSS_CM3_0_app Debug [Microsemi Cortex-M3 RAM Target]	Name					Value
🖨 🔗 Embedded GDB (9/15/11 8:03 PM) (Suspended)	🕀 🥭 gre	eeting				0x2000ffaC
Thread [1] (Suspended)		annel_name				0x0000000
I main() c:\microsemiprj\pot_led_libero_softconsole\voltage_moniter\softco C:\Program Files\Microsemi\SoftConsole v3.3\Sourcery-G++\bin\arm-none-eabi-gdt						
Signal Company Files (Microseningson Console vs. 5) sourcery-G++ (pinternetione-eabi-gat)						
	<					>
						<u>~</u>
						~
	> <				~	2
📄 Welcome 🚺 C main.c 🛛 C -1 <symbol available="" is="" not=""></symbol>				E Outline 🛛	3	
25			~		Jaz 😿 🗙	● # ▽
26 #define Microsemi_logo \				.\di	rivers\mss_uart\mss_u	Jart.h
27"\n\r \				🖬 .\di	rivers\mss_ace\mss_a	ce.h
28** ** ******* ****** ****** ***** 29* * * * * * * * * * * *	****	* * *	*		rivers\mss_gpio\mss_	gpio.h
30* * * * * * ***** * * ****	*****	* * *	*		io.h	
31* * * * * * * * *	* *	* *	*		in() : int	
32* * ******* ***** * * ****	***** *	* *	*****	# Mic	rosemi_logo	
33						
34						
35 const uint8_t greeting[] = "\n\rWelcome to Micros	emi's Smar	tFusion Vo	ltage M			
36 const uint8 t * channel name;			>			
Sconsole X A Tasks Problems C Executables A Memory			< 🔌 🕞 🚮		- <u>-</u>	•••
Voltage Moniter MSS MSS CM3 0 app Debug [Microsemi Cortex-M3 RAM Target] C:\Program F	:					
Loading section .data, size 0x5b8 lma 0x2000ab90	nestmicroseniitz	OLCOHSOIR A3.31	500rcery-G++(Din	(ann-none-eaoi-	-yuuvexe (9/15/11 o:u	12 PM)
Start address 0x20000298, load size 45384						
Transfer rate: 31 KB/sec, 5673 bytes/write.						-
set \$sp = *0x60080000						~
						>
. □* v	/ritable	Smart Insert	35:1			

7. Your Debug Prospective should resemble Figure 38 · :

Figure 38 · Debug Perspective

- 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.
- Turn the potentiometer (POT) on the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board. The voltage measurement will be 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.

Figure 39 · Voltage Monitor

**	*	* *	*****	*****	****	*	***;	****	*****	** *	* ******
* *	*	×	*	*	*	*	*	* *	*	** *	* *
* *	*	×	×	*	****	×	×	* ****	*****	* * *	* *
*)	×	×	×	×	*	×	×	* *	×	* *	* *
×		* *	*****	*****	×	*	***;	*****	*****	*	* ******
MØ_V¢	olta	ge :	2.560V								

Figure 40 · 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.

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

🗱 Variables 🤷 Breakpoints	s 👬 Registers 🖾 🛋 Modules
Name	Value
🖃 👬 Main	
8181 rO	0x2000b4f4
0101 r1	0x2000ff80
1111 r2	0x60
888 r3	0x2000b4f4
1010 r4	0x0
888 r5	0xe0042020
1919 r6	0xc
8181 r7	0×2000ff58
8181 r8	0xe004201c
888 r9	0x33f236b5
100 r10	0×1da60880
0101 r11	0×18001666
1010 4 0	a according to

Figure 41 · The Registers Tab

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

🏇 Debug 🗄 Outline 🚧 Variables 🕴 💊 Breakpoints 👯 Registers 🛋	Modules
Name	Value
(X)= rx_size	0
🗆 🥭 rx_buff	0x2000ffeb
(X)= rx_buff[0]	0
🖃 🥭 greeting	0x2000ffba
(x)= greeting[0]	'W'
(X)= greeting[1]	'e'
(x)= greeting[2]	Ψ
(X)= greeting[3]	'c'
·· ··	
<u><</u>	

Figure 42 · The Variables Tal

15. Choose **Window > Show View > Disassembly** to display the assembly level instructions. The Assembly window is displayed on the right side of the Debug perspective.



SC Debug - Voltage_Moniter_MSS_MSS_CM3_0_app/main.c - I Elle Edit Source Refactor Navigate Search Project Run Window	
📬 • 🔚 🖻 📾 🎄 • 💽 • 💁 • 🥭 🛷 • 🌛	עַר - אָר - אָר - אָר
🏇 Debug 🕱 📃 🗖	🛋 Modules 💊 Breakpoints 🚻 Registers 🕬= Variables 励 Disassembly 🛛 🗸 🗸 🖓
Image: Solution of the second seco	<pre>{ (</pre>
Image: Construction Image: Construction 25 26 26 Image: Construction 27 Image: Construction 28 Image: Construction 29 Image: Construction 29 Image: Construction 20 Image: Construction 30 Image: Construction 31 Image: Construction 31 Image: Construction 20 Image: Construction	<pre>0x20000444 <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) 0x200004ed <main+45>: stmia.w r12, (r0, r1) MSS_GPIO_init(); 0x200004e4 <main+52>: bl 0x20001880 <mss_gpio_init> MSS_GPIO_config(MSS_GPIO_27 , MSS_GPIO_0UTPUT_MODE); 0x200004es <main+56>: mov.w r0, #27 0x200004ec <main+66>: mov.w r1, #5 0x200004ec <main+66>: bl 0x200018ec <mss_gpio_config> MSS_GPIO_config(MSS_GPIO_26 , MSS_GPIO_0UTPUT_MODE); 0x200004f4 <main+68>: mov.w r0, #26 </main+68></mss_gpio_config></main+66></main+66></main+56></mss_gpio_init></main+52></main+45></main+44></main+40></main+36></pre>
Console Consol	** ** **

Figure 43 · Assembly Window

- 16. 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.
- 17. Click the Instruction Stepping it icon and then perform Step Into operations. Observe that Step Into now executes a single line of assembly code.
- 18. 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.
- 19. Resume execution of the code by choosing **Run > Resume** or by clicking the **Resume** icon. You can even add breakpoints in the application for further debugging.



 Once you made the required changes, terminate the debugger by selecting Voltage_Monitor_MSS_MSS_CM3_0_app Debug in the Debug view, then right-clicking and selecting

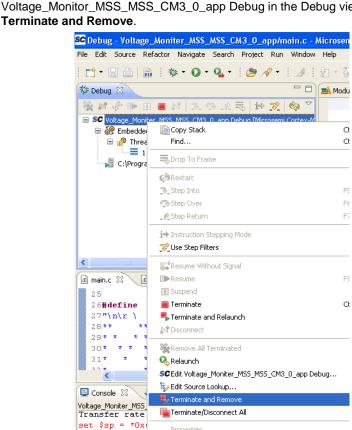


Figure 44 · Terminating the Program

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

Step 10 - Building 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 in-application programming (IAP) or FlashPro programming software. In release mode, you cannot use 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 Catalog Settings

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

1. On the Catalog window, click Options.





- 2. The **Options** window is displayed. Click **Repositories** under **Vault/Repositeries Settings** 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.

Options		? 🛛
 Vault/Repositories Settings Repositories Vault location Vault location View Settings Display Filters 	www.actel-ip.com/repositories/SgCore www.actel-ip.com/repositories/DirectCore www.actel-ip.com/repositories/Firmware	Add Remove
Help	ОК	Cancel

Figure 46 · Setting Repositories



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

Options	? 🛛
Vault/Repositories Settings	Current vault location: C:\Documents and Settings\donthus\Application I
View Settings Display	Select new <u>vault location</u> : C:\Documents and Settings\donthus\Application Data\Actel\tools Default
Help	OK Cancel

Figure 47 · 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.

e View <mark>Tools</mark> Help				
😋 Reload Catalog				
Vault/Repositories Settings			Search by all fields (27/27):	
display only the latest version of a core				<u> </u>
Jame	Version	Size (MB)	Status	^
ore10100_AHBAPB Driver	2.0.103	0.44		
ore16550 Driver	2.1.100	0.48		
oreAI Driver	3.0.101	0.43		
oreAhbNvm Driver	2.1.102	0.23		
oreGPIO Driver	3.0.101	0.55		
oreI2C Driver	2.0.103	0.43		
oreInterrupt Driver	2.1.102	0.2		
oreLPC Driver	2.0.100	0.54		
orePWM Driver	2.1.107	0.45		
oreSPI Driver	2.3.103	0.31		
oreTimer Driver	2.1.101	0.3		
oreUARTapb Driver	3.0.105	0.44		
oreWatchdog Driver	2.1.101	0.26		
lardware Abstraction Layer (HAL)	2.1.102	0.21		
martFusion CMSIS-PAL	2.2.100	0.34		
martFusion MSS ACE Driver	2.2.101	1.9		
martFusion MSS Ethernet MAC Driver	2.0.103	0.28		
martFusion MSS GPIO Driver	2.0.105	0.57		
martFusion MSS I2C Driver	2.0.101	0.55		
martFusion MSS IAP Driver	2.2.101	1.4		
martFusion MSS Peripheral DMA Driver	2.0.102	0.7		
martFusion MSS RTC Driver	2.0.100	0.95		
martFusion MSS SPI Driver	2.1.100	0.8		
martFusion MSS Timer Driver	2.1.101	0.93		
martFusion MSS UART Driver	2.2.101	0.94		~
ocumentation:				~
IAL RN.pdf				
ree software license.pdf				
<u> </u>				
escription: The Hardware Abstraction Laye	er is used by drivers to a	access the hardwa	are. It also allow the control of interrupt	:s.
he HAL is normally required when using bare				
ompatible IP cores:	~ ~			~
		1 I		
New cores are available for download	Download them now!		🔛 Download 🛛 🟥 G	enerate

Figure 48 · 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 $49 \cdot$, and click **OK**.

🗟 Cont	figuring MSS_G	PIO_0 (MSS_GPIO	- 1.0.101)				
Con	figuration						<u>^</u>
	-Multiplexed Wit	h I2C1 Peripheral Dedic	ated I/Os				
	GPIO_31:	Use as MSS I/O Pad	Not Used	U20	or connect to Fabric	Output 💌	
	GPIO_30:	Use as MSS I/O Pad	Not Used	V22	or connect to Fabric	Output 💌	
	-Multiplexed Wit	h UART1 Peripheral Dec	dicated I/Os				
	GPIO_29:	Use as MSS I/O Pad	Not Used 💉	W22	or connect to Fabric	Output 💌	
	GPIO_28:	Use as MSS I/O Pad	Not Used 💊	V20	or connect to Fabric	Output 💌	
	-Multiplexed Wit	h SPI1 Peripheral Dedic	ated I/Os				
	GPIO_27:	Use as MSS I/O Pad	Not Used 🛛 👻	W21	or connect to Fabric	Not Used 🛛 💌	
	GPIO 26:	Lise as MSS I/O Pad	Not Used 🔍 🗸	AA22	or connect to Fabric	Not Lised 🗸	×
Hel	₽ ▼					ОК	Cancel

Figure 49 · Configure MSS_GPIO_0

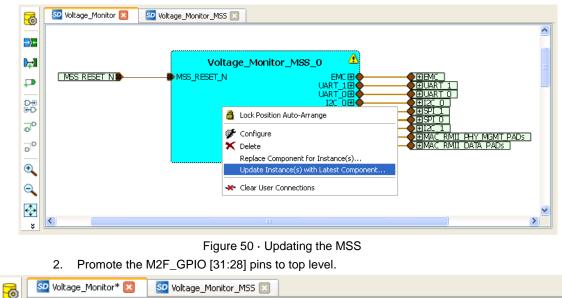
This example requires GPIO_31, GPIO_30, GPIO_29, and GPIO_28 to be connected to LED_4 to LED_1 on the SmartFusion Evaluation Kit Board, and D4 to D1 on the SmartFusion Development Kit Board. These signals will be routed through the fabric to I/O pins H17, C19, B20, and B19, respectively.

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 50 · .





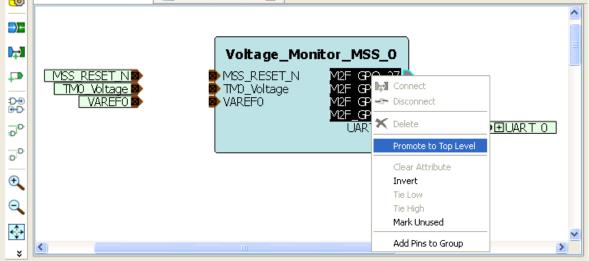


Figure 51 · GPIO Pins Promoted to Top Level

3. Click **Design > Configure Firmware** as shown in Figure 52 · .

Project File Edit View	Design SmartDesign Help
l 🗅 🚔 🔚 🚄 S	🎯 Configure Firmware
	🜔 Generate Programming Data
Design Hierarchy	Reports
_,	

Figure 52 · Opening Design_Firmware

4. On the **DESIGN_FIRMWARE** tab, clear the **Generate** check box 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.



SD	voltage_Moni	tor 🗈	🛛 🛛 🕺 Voltage_Monitor_MSS 🖾	SO DESIGN_FIRMWARE*		
	Generate		Instance Name	Core Type	Version	Compatible Hardware Instance
1	V	Ť	HAL_0	HAL	2.1.102	Voltage_Monitor_MSS
2	v		MSS_ACE_Driver_0	MSS_ACE_Driver	2.2.101 🗸	Voltage_Monitor_MSS:MSS_ACE_0
3			MSS_IAP_Driver_0	MSS_IAP_Driver	2.2.101 🗸	Voltage_Monitor_MSS
4			MSS_NVM_Driver_0	MSS_NVM_Driver	2.2.102	Voltage_Monitor_MSS:MSS_ENVM_0
5			MSS_PDMA_Driver_0	MSS_PDMA_Driver	2.0.102	Voltage_Monitor_MSS:MSS_DMA_0
6			MSS_RTC_Driver_0	MSS_RTC_Driver	2.0.100	Voltage_Monitor_MSS:MSS_RTC_0
7			MSS_Timer_Driver_0	MSS_Timer_Driver	2.1.101 🗸	Voltage_Monitor_MSS:MSS_TIMER_0
8	~		MSS_UART_Driver_0	MSS_UART_Driver	2.2.101 🗸	Voltage_Monitor_MSS:MSS_UART_0
9		Ø,	SmartFusion_CMSIS_PAL_0	SmartFusion_CMSIS_PAL	2.3.102 🗸	Voltage_Monitor_MSS

Figure 53 · Configuring SmartFusion_CMSIS_PAL_0

- 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**.

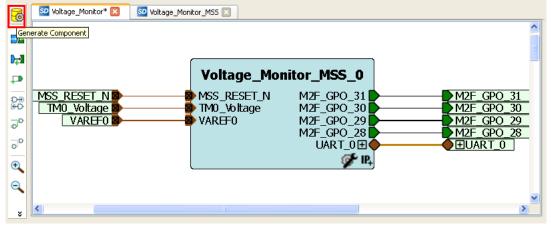


Figure 54 · Generating the MSS Component

After successful generation of project 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.



9. Confirm that the SoftConsole folder is created with the folders and files as shown in Figure 55 · .

F	iles		8	×
				_
	🕀 🔁	component		
	🗀	constraint		
		coreconsole		
	🕀 📄	designer		
	🕀 📄	firmware		
		hdl		
	- 🗀	phy_synthesis		
	🕀 🔁	simulation		
	<u>-</u>	smartgen		
		SoftConsole		
	_	Voltage_Moniter_MSS_MSS_CM3_0	_	
		🕀 🔁 .metadata		
		projects	_	
		Voltage_Moniter_MSS_MSS_CM3_0_app		
		Voltage_Moniter_MSS_MSS_CM3_0_hw_platform	n	
		project		
		🖨 🧰 drivers		
		mss_ace		
		□ drivers_config ★ □ mss_ace		
		⊕ ⊡ mss_ace ⊕ ⊡ hal		
		Voltage_Moniter_hw_platform.h		
		stimulus		
		suntatus synthesis		
	⊕ ∩	viewdraw		
		YICYYGI GW		

Figure 55 · 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 **Generating Programming Data**, you can complete 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 additional information, 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.



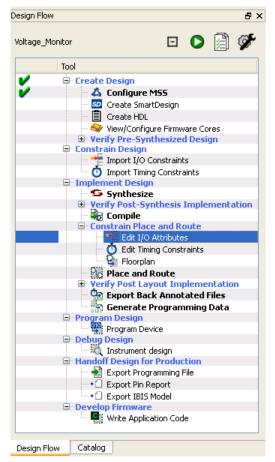


Figure 56 · Edit I/O Attributes



- 2. Make the following pin assignments in MultiView Navigator window as shown in Figure 57 · :
 - GPO_28 to B19
 - GPO_29 to B20
 - GPO_30 to C19
 - GPO_31 to H17

🐊 MultiView Navigator [Voltage]	_Monitor *] - [I/O Attri	bute Editor]						
➡ Eile Edit Yiew Logic Format	<u>T</u> ools <u>W</u> indow <u>H</u> elp						-	₽ ×
© ∯ ⊇ ⊆ M ? .	Ø Ø Ø Ø Ø 🖉 🗏	+D	a to to	⊿ @ ₩	🔁 🎢	🖾 📲 🔺	71 1	* n
E Logical B M2F_GP0_28_pad	Port Name/	Group	Macro Cell	Pin Number	Locked	Bank Name	1/0 Standard	Out _l Drive
B	1 M2F_GP0_28		ADLIB:OUTB	B19	~	BankO	LVTTL	12
	2 M2F_GP0_29		ADLIB:OUTB	B20	~	BankO	LVTTL	12
	3 M2F_GP0_30		ADLIB:OUTB	C19		BankO	LVTTL	12
	4 M2F_GP0_31		ADLIB:OUTB	H17 ⊻		Bank1	LVTTL	12
	5 MSS_RESET		ADLIB:INBUF	R1		Bank2	LVTTL	•
	<	Package Pins	/					>
		esults 👌 Find 1	1					
Ready			row 4, col 4 i	FAM: SmartFusio	on DIE: A2	F200M3F PAC	KAGE: 484 FE	GA

Figure 57 · MultiView Navigator GUI

- Commit and check the edits using File > Commit and Check. Connect 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.

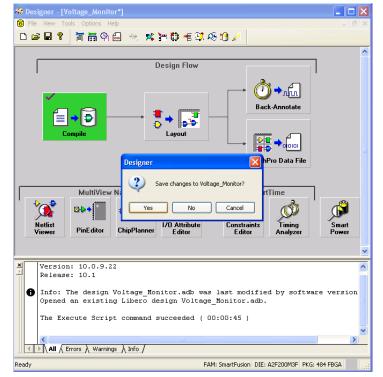


Figure 58 · Designer Window



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

Voltage_Monit	tor									-	0	Ĩ	Ş,
	Tool			_					_				
	4	Þ	Сгеа	ate	Des	sian							
V		ĺ.			nfigu	_	ASS						
·					eate S			ian					
					eate H			Ĩ.,					
			-		eate H		Test	Benc	h				
			9	Vie	w/Co	onfig	gure	Firm	war	e Cor	es		
		\triangleright	•	Vei	rify F	Pre-S	- Synt	hesiz	zed	Desi	gn		
	\triangleright	۲	Con	str	ain D	Desig	gn				-		
	4	۲	Imp	oler	nent	t De	sign						
V		\triangleright	S	Syr	nthe	size							
		۵	•	Vei	rify F	Post-	Syn	thes	is In	npler	nenta	tion	
				ê	Sim	nulate	e						
V		\triangleright	R (Сог	mpile	e							
		⊿			nstra								
			-	+	Crea	ate/E	Edit I	/O A	ttrik	outes			
			(Ö	Crea	ate/E	Edit 1	Timir	ng C	onsti	raints		
			Ŧ	₽,	Floo	orpla	in						
				Pla	ce ai	nd R	oute	e					
		⊿			-		-	out I	mpl	emei	ntatio	n	
				_	Sim								
					Veri	-		-					
			_ 1	þ,	Verit	ify Po	ower						
				-						Files			
								ramn	ning	j Dat	a		
	4	Þ	Prog	-									
			<u>610</u>	Pro	ogran	m De	vice						
	⊳		Deb	_		-							
	\triangleright		Han					Pro	duc	tion			
	4	۲	Dev		-								
			C	Wri	ite A	pplic	catio	n Co	de				

Figure 59 · Generating Programming Data



7. The **Design Flow** window looks similar to Figure $60 \cdot .$

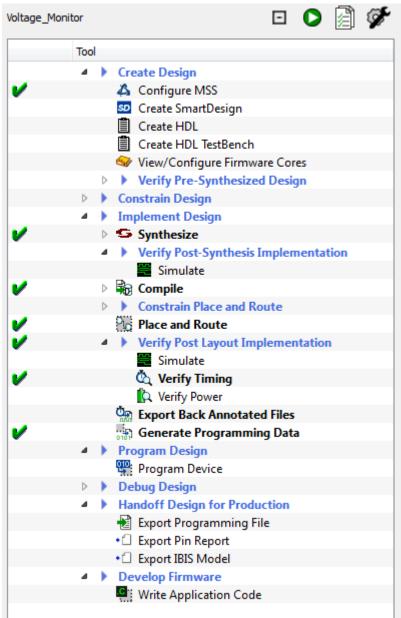


Figure 60 · Design Flow Window After Building The Project

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



List of Changes

Revision	Changes	Page			
Revision 6	Modified Associated Project Files section (SAR 38282).	3			
(May 2012)	Modified Step 1 - Creating a Libero SoC Project section (SAR 38282).	5			
	Updated Figure 6 (SAR 38282).	8			
	Updated Figure 16 (SAR 38282).	13			
	Updated Figure 25 (SAR 38282).	18			
	Updated Figure 26 (SAR 38282).	20			
	Updated Figure 27 (SAR 38282).	21			
	Modified Step 6 - Building the Software Application Through SoftConsole section (SAR 38282).	21			
	Updated Figure 59 (SAR 38282).	46			
	Updated Figure 60 (SAR 38282).	48			
Revision 5 (February 2012)	Modified Associated Project Files section (SAR 36900).				
	Modified Step 6 - Building the Software Application Through SoftConsole section (SAR 36900)	21			
	Modified Step 7 - Configuring Serial Terminal Emulation Program section (SAR 36900).	26			
	Updated Figure 34 (SAR 36900).	26			
	Modified Step 8 - Installing Drivers for the USB to RS232 Bridge section (SAR 36900).	28			
	Modified Step 9 - Debugging the Application Project using SoftConsole section (SAR 36900).	29			
Revision 4	Modified Step 2 - Configuring MSS Peripherals section. (SAR 36492)	8 and 15			
(January 2012)	Modified Step 6 - Building the Software Application Through SoftConsole section (SAR 36492)				
	Modified Step 9 - Debugging the Application Project using SoftConsole section (SAR 36492)	29			
Revision 3 (November 2011)	Updated the document for Libero SoC v10.0 (SAR 35045).				

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 <u>My Cases</u>, select Yes in the ITAR dropdown list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the <u>ITAR</u> 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 Microsemi Corporation (NASDAQ: MSCC) offers a comprehensive portfolio of semiconductor solutions for: aerospace, defense and security; enterprise and communications; and industrial and alternative energy markets. Products include high-performance, high-reliability analog and RF devices, mixed signal and RF integrated circuits, customizable SoCs, FPGAs, and complete subsystems. Microsemi is headquartered in Aliso Viejo, Calif. Learn more at **www.microsemi.com**.

© 2012 Microsemi Corporation. All rights reserved. Microsemi and the Microsemi logo are trademarks of Microsemi Corporation. All other trademarks and service marks are the property of their respective owners.