Displaying POT Level with LEDs

Libero SoC and IAR Embedded Workbench Flow Tutorial for SmartFusion cSoC



Table of Contents

Introduction	3
Tutorial Requirements	3
Objective	4
Working with Libero SoC and IAR	5
Step 1 - Creating a Libero SoC Project	5
Step 2 - Configuring MSS Peripherals	7
Step 3 - Generating the MSS Component	
Step 4 - Generating the Program File	16
Step 5 - Programming SmartFusion Board Using FlashPro	17
Step 6 - Building the Software Application Through IAR Embedded Workbench	
Step 7 - Configuring the Serial Terminal Emulation Program	28
Step 8 - Installing Drivers for the USB to RS232 Bridge	30
Step 9 - Debugging the Application Project Using IAR	
Appendix A – Libero SoC Catalog Settings	
Appendix B – Firmware Catalog Settings	
	37
Appendix B – Firmware Catalog Settings	37 39
Appendix B – Firmware Catalog Settings Appendix C List of Changes	37 39 45
Appendix B – Firmware Catalog Settings	37 39 45 47
Appendix B – Firmware Catalog Settings Appendix C List of Changes Product Support	37 39 45 47 47
Appendix B – Firmware Catalog Settings Appendix C List of Changes Product Support Customer Service.	37 39 45 47 47 47
Appendix B – Firmware Catalog Settings. Appendix C. List of Changes. Product Support Customer Service. Customer Technical Support Center.	37 39 45 47 47 47 47
Appendix B – Firmware Catalog Settings. Appendix C. List of Changes. Product Support Customer Service. Customer Technical Support Center. Technical Support	37 39 45 47 47 47 47 47



Introduction

This tutorial demonstrates how 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.
- Configuring the peripherals using SmartDesign
- Configuring the analog compute engine (ACE)
- · Generating the microcontroller subsystem (MSS) component
- Generating the programming file to program the SmartFusion cSoC device
- Opening the project in Embedded Workbench[®] for ARM[®] (EWARM) IDE from Libero SoC and writing application code
- · Compiling application code
- · Creating and launching a debug session
- Debugging and running the code using IAR

Tutorial Requirements

Software Requirements

This tutorial requires the following software installed on your PC:

- Libero SoC v10.0 (or later) that can be downloaded from: www.microsemi.com/soc/download/software/libero/files.aspx.
- IAR Embedded Workbench[®] v6.30.7 or later.

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 the PC.
- IAR supplied J-Link debugger hardware (not supplied with the SmartFusion Kit Board).

Associated Project Files

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

The IAR_Debugger_files folder contains the following files:

• SmartFusion_dss.ddf: DDF files allows you to see the MSS register map for all peripherals.

Note: Extract the Design Files to root directory (C:\).

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



MSS Components Used

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

Target Board

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

Objective

The objective of this tutorial is to instruct how to configure the SmartFusion analog channels and ACE, used to monitor the voltage across the potentiometer. The UART is used to send the ADC results to a terminal program.

Design Steps

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

- Create a Libero SoC v10.0 project and use the SmartFusion cSoC MSS configurator to configure ACE, adding a voltage monitor with flags.
- Generate the SmartFusion cSoC MSS component.
- Perform synthesis and layout, and generate a programming file to program the SmartFusion cSoC device.
- Program the SmartFusionA2F200M3F or A2F500M3F cSoC device.
- Open the software project in IAR Embedded Workbench IDE and write application code.

Run an application to monitor the voltage across the POT on the SmartFusion Evaluation Kit Board or Development Kit Board.

The hardware configuration has four flags:

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

The design monitors voltage across a potentiometer (POT) and the 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 IAR

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

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_IAR)
 - 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.

Project				
Name:	Voltage_Monitor			
.ocation:	C:/Microsemiprj/POT_LED_Libero	IAR		Browse
Prefered HDL type:	💿 Verilog 🔘 VHDL			
Description:				
1 Edit Tool Profiles				
Device				
Family:	SmartFusion 💌			
Die:	A2F200M3F			
Package:	484 FBGA 💙			
Speed:	STD 💙			
Die Voltage:	1.5 💙			
Operating Conditions:	сом 💌			
Tempe	erature (in degrees Celsius)	Best 0	Typical 25	Worst 85
VCCI	Voltage (in volts) 1.5 Voltage (in volts)	1.575 1.6	1.5 1.5	1.425 1.4
VCCI	1.8 Voltage (in volts) 2.5 Voltage (in volts) 3.3 Voltage (in volts)	1.9 2.7 3.6	1.8 2.5 3.3	1.7 2.3 3
Design Template				
🗹 Use template				
	Core			Version
SmartFusion Microcontr	oller Subsystem (MSS)			2.5.106
			🗹 si	how only latest versio

Figure 1 · New Project Dialog Box



🗢 Tool Profiles	? 🗙
Tools Software IDE Synthesis Simulation Programming Add Pro	
Name: Tool integrati Version: Location:	IAR IAR EWARM IAR EWARM Incommon\bin\IarIdePm.exe Browse IAR IAR EWARM I
Нер	OK Cancel
Help	Export Profiles) OK Cancel

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

Figure 2 · Selecting IAR as Software IDE

3. 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.

- 4. Select the MSS core in New Project Dialog Box and click OK.
 - Note: If SmartFusion cSoC MSS does not appear in the list, refer to Appendix A Libero SoC Catalog Settings to find out how to set your repositories. If your vault does not have MSS core, download the core by double clicking on the core name in **Design template** in the **New Project Dialog Box**.
- 5. The project is created and the Libero SoC window is displayed, as shown in Figure 3 · . The SmartDesign "Voltage_Monitor" is created with the instantiation of MSS component.

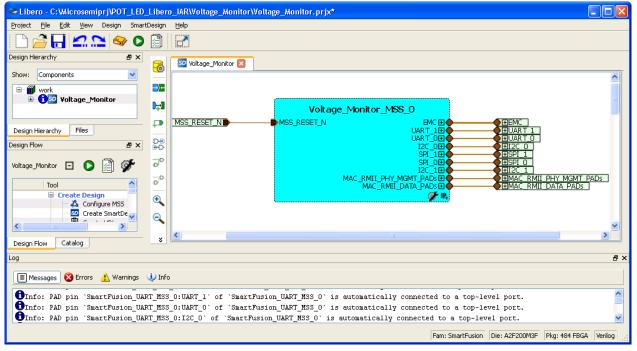


Figure 3 · The Libero Window After Creating New Project Wizard

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 in a new tab, as shown in Figure 4 · .

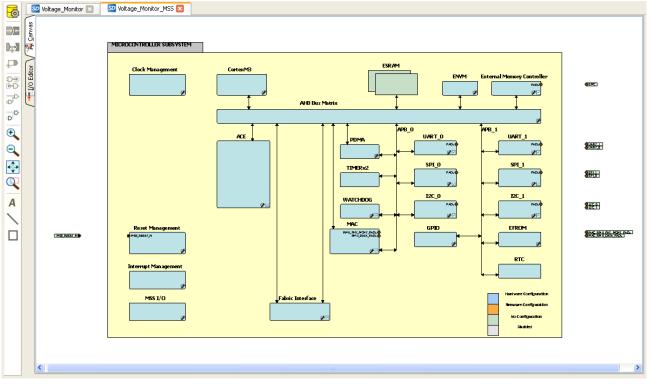


Figure 4 · MSS in the SmartDesign Canvas

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 .

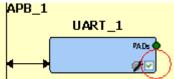
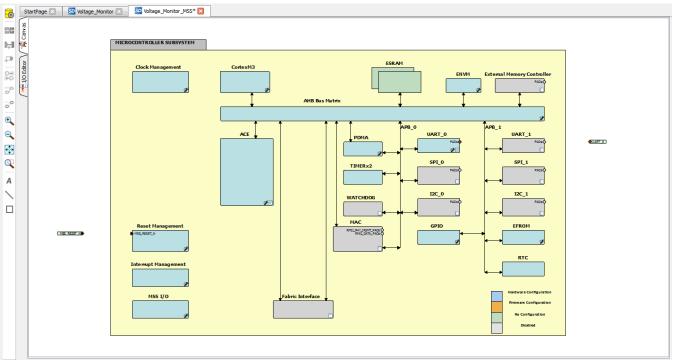


Figure 5 · Enabling the Peripheral

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





2. Disable the following peripherals: MAC, 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.

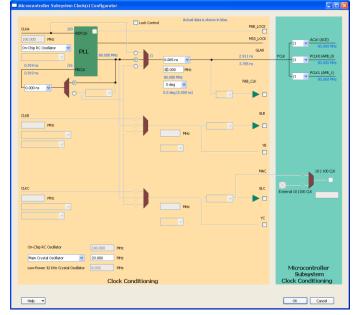


Figure 7 · MSS Clock Configuration



Configuring ACE

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				
Configure ACE	Controller	Flags		
ACL <u>K</u> : 80 MHz	ADC Clock: 10 MHz	Regolution: 12 vits		Advanced Ogtions
Add service:				
Active Bipolar Prescalar Inpl ACC Overset Input Current Input Differential Input TVTTI Input Analog Comparator Sigma Delta DAC	Service	Signal	Sample time (us) Pack	age Pin SCB
	ADC Block 0 ADC Block 1			
				OK Cancel

Figure 8 · MSS ACE Configuration

- 2. Select ADC Direct Input and click 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

Configure ADC Dire	ct Input						
тмо	<u>S</u> ignal na	me:					
	TM0_j/ol	tage		Send raw A	DC result to D	MA NA	
Acguisition time:	10	000 us					
Digital filtering				Linear trans	formation		
Filtering facto	r: Nor	ie 🔽		Scale factor	:		
Ini <u>t</u> ial value;	0.0	00 V		Offset:			
Send filter	ed result to	DMA		Send trar	nsformed resu	ilt to DMA	
		Thre	eshold Detectio	n		*	×
Flag	g Name	Flag Type	Threshold (¥)	Hysteresis (mV)	Assert Samples	De-asse Sample	
over_1	рОу	OVER	1	1			
over_1		OVER	1.5	1			
over_2		OVER	2	1			
over_2	рбу	OVER	2.5	1			×
📃 Assert flag w	hen post p	ocessing completed	ł				
					ОК		ancel

Figure 9 · MSS ADC Direct Input Configuration



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

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

- 4. Click OK.
- 5. Assign the ADC Direct Input Signal to the package pin W8 in the Configure ADC dialog box. The **Configure ACE** tab is displayed as shown in Figure 10 · .

Configure ACE				
Configure ACE	Controller	Flags		
ACLL: 80 MHz	ADC Clock: 10 MHz	Regolution: 12 Vits	Advance	d Options
Add service:				_/ ×
Active Bodar Proceed Input ACC Drevet Input Control Input Temporative Structure Analog Comparator Signis Delta DAC	Service (ADC Direct Input	Signal TM0_Voltege	Sample time (us) 100000 W8 (1140)	SCB V 0(4)
	ADC Block 0 ADC Block 1			
			ОК	Cancel

Figure 10 · MSS ACE Configuration With ADC Direct Input

- 6. The next step in configuring the ACE is to enable the sampling sequence. This configuration dialog is launched by clicking on the **Controller** tab (next to the **Configure ACE** tab).
- 7. Select Manual as the Operating sequence entry in the Controller tab and click Insert operating sequence slot button (see Figure 11 ·).

Configure ACE	Controller	Flags			
configure Acc	Concord				
Procedures		* = / ×			
	Name		ADC Block 0		
ADC0_MAIN ADC1_MAIN			ADC Block 1		
, and the second s					
1		,			
Operating sequence entry:	🔿 Auto 💿 Ma	nuali			
Details of procedure:	ADC0_MAIN				
Available signals:	Sampling r	ate			
		Signal	Actual Rate (ksps)		
	->>		,		
	<				
			Insert operating sequence	alat	
	Calculate Actual Rate	<u>T</u> otal san	P Insercoperacing sequence	sion	
Operating sequence					
			SSE Execution		
	Instruct	on	Time (us)		

Figure 11 · MSS ACE Configuration to Enable Sampling Sequence



8. Select SAMPLE.

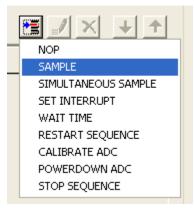


Figure 12 · Select SAMPLE

This displays the Configure SAMPLE window.

9. Select TM0_voltage and click OK.

Configure 'SAMPLE'
Analog Pad: TM0_Voltage
Description
Sample Analog Peripheral. Only certain peripherals are allowed on this procedure. For more information refer to Sampling Sequence Configuration in ACE Configuration user guide.
OK Cancel



10. Click Insert operating sequence slot and select RESTART SEQUENCE.

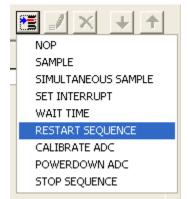


Figure 14 · Select Restart Sequence



11. Click Calculate Actual Rate.

Configure ACE
Configure ACE Controller Flags
Procedures
Name ADC0_MAIN ADC1_MAIN ADC Block 0 ADC1_MAIN ADC Block 1
Operating sequence entry: O Auto O Manual
Details of procedure: ADC0_MAIN
Available signals: Sampling rate
->> Signal Actual Rate (ksps)
->> TM0_Voltage 0.000
Calculate a round robin operating sequence of the specified signals.
Calculate Actual Rate
Operating sequence
Instruction SSE Execution Time [us]
Sample TM0_Voltage 11.900
Restarts the execution sequence for this timeslot 0.080
OK Cancel

Figure 15 · MSS ACE Configuration: Calculate Actual Rate

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

Configure ACE	×
Configure ACE Controller Flags	
Procedures	
Name	
ADCD MAIN ADC Block 0 ADC Block 1	
, Operating sequence entry: O Auto O Manual	
Details of procedure: ADC0_MAIN	
Available signals: Sampling rate	
->> Signal Actual Rate (ksps) ->> TM0_Voltage 83200	
Iotal sampling rate: 83.200 ksps	
Operating sequence	
Instruction SSE Execution Time (us)	
Sample TM0_Voltage 11,900 Restarts the execution sequence for this timeslot 0,080	
OK Cancel	

Figure 16 · MSS ACE Configuration: Controller Tab

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



15. 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.

igure ACE				
Configure ACE Controller	Flags			
Select a register to view ACE flag mapping:	Available bits of PPE_FLAG50 register:			
Fabric Flag Registers	Bit Source	ACE Cortex-M Interrupt Interrup	3 Lock	
	0 TM0_Voltage:over_1p0v	54 118		
PPE FLAGS1 (0x40021454)	1 TM0_Voltage:over_1p5v	55 119		
PPE_FLAG52 (0x40021458)	2 TM0_Voltage:over_2p0v	56 120		
PPE_FLAG53 (0x4002145c)	3 TM0_Voltage:over_2p5v	57 121		
PPE_SFFLAGS (0x40021460)	4 <none></none>	58 122		
	5 <none></none>	59 123		
	6 <none></none>	60 124		
	7 <none></none>	61 125		
	8 <none></none>	62 126		
	9 <none></none>	63 127		
	10 <none></none>	64 128		
	11 <none></none>	65 129		
	12 <none></none>	66 130		
	13 <none></none>	67 131		
	14 <none></none>	68 132		
	15 <none></none>	69 133		
	16 <none></none>	70 134		
	17 <none></none>	71 135		
	18 <none></none>	72 136		
	19 <none></none>	73 137		
	20 <none></none>	74 138		
	21 <none></none>	75 139		
	22 <none></none>	76 140		
	23 <none></none>	77 141		
	24 <none></none>	78 142		
	25 <none></none>	79 143		
	26 <none></none>	80 144		
	27 <none></none>	81 145		
	28 <none></none>	82 146		
	29 <none></none>	83 147		
			ок	Cancel
				Cancer

Figure 17 · ACE Flag Mapping – PPE Flag Registers

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

Configuring the General Purpose Input/Output (GPIO) Peripheral

- Note: If you are not using the SmartFusion Evaluation Kit Board Revision E 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 and configure as shown in Figure 18 · ; and click **OK**.

🕵 Confi	iguring MSS_G	PIO_0 (MSS_GPIO	- 1.0.101)				
Confi	iguration						<u>^</u>
	Multiplexed With	n I2C1 Peripheral Dedic	ated 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	n UART1 Peripheral Dec	licated I/Os				Ξ.
	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	n 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 Used	<u>~</u>
Help	•					ОК	Cancel

- Figure 18 · MSS GPIO Configuration
- 2. This example requires GPIO_31, GPIO_30, GPIO_29, and GPIO_28 to be connected to LED_8 to LED_5 on the SmartFusion Evaluation Kit Board.
- 3. Click **File > Save** to save the Voltage_Monitor_MSS.



Step 3 - Generating the MSS Component

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

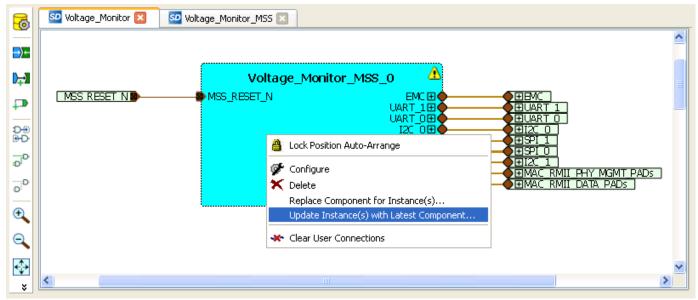


Figure 19 · Updating the MSS

2. Click **Design > Configure Firmware** as shown in Figure $20 \cdot .$

Project File Edit View	Design SmartDesign Help
E 🕞 🔁 🔲 💋 S	😔 Configure Firmware
	Generate Programming Data
Design Hierarchy	-
	🗐 Reports

Figure 20 · Opening Design_Firmware

3. 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 IAR Embedded Workbench as the configuration.

	Instance Name	Core Type	Version	Compatible Hardware Instance
Ø 🖣	HAL_0	HAL	2.2.102 🗸	Voltage_Monitor_MSS
-	MSS_ACE_Driver_0	MSS_ACE_Driver	2.3.105 🗸	Voltage_Monitor_MSS:MSS_ACE_0
-	MSS_GPIO_Driver_0	MSS_GPIO_Driver	2.0.105 🗸	Voltage_Monitor_MSS:MSS_GPIO_0
-	MSS_IAP_Driver_0	MSS_IAP_Driver	2.2.101 🗸	Voltage_Monitor_MSS
-	MSS_NVM_Driver_0	MSS_NVM_Driver	2.2.102 🗸	Voltage_Monitor_MSS:MSS_ENVM_0
-	MSS_PDMA_Driver_0	MSS_PDMA_Driver	2.0.102	Voltage_Monitor_MSS:MSS_DMA_0
-	MSS_RTC_Driver_0	MSS_RTC_Driver	2.0.100	Voltage_Monitor_MSS:MSS_RTC_0
-	MSS_Timer_Driver_0	MSS_Timer_Driver	2.1.101 🗸	Voltage_Monitor_MSS:MSS_TIMER_0
-	MSS_UART_Driver_0	MSS_UART_Driver	2.3.101 🗸	Voltage_Monitor_MSS:MSS_UART_0
F	Configuring SmartFusion_CMSIS_PAL	SmartFusion_CMSIS_PAL	2.3.103 🗸	Voltage_Monitor_MSS
	Configuration Software Tool Chain IAR Embedded Workbench V Help V OK Cance	_		

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 the Appendix B – Firmware Catalog Settings.
- 5. Click **File > Save** to save the **Design_Firmware**.

Filos

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

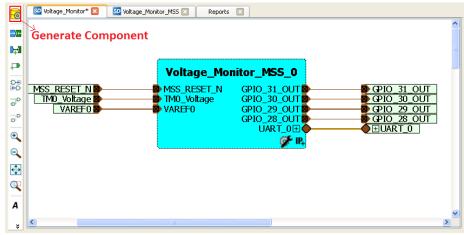


Figure 22 · Generating the MSS Component

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

Image: Component constraint constraint coreconsole image:	riles E	γ.
 constraint coreconsole designer firmware hdl CMSIS drivers drivers drivers_config hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MSS_MSS_CM3_0.eww Voltage_Monitor_MSS_MSS_CM3_0_app.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation smartgen synthesis viewdraw 		
 coreconsole designer firmware hdl CMSIS drivers drivers drivers_config hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MS5_MS5_CM3_0.eww Voltage_Monitor_MS5_MS5_CM3_0_app.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus yithesis viewdraw 		
 designer firmware hdl CMSIS drivers drivers drivers_config hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MS5_MS5_CM3_0.eww Voltage_Monitor_MS5_MS5_CM3_0_app.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis viewdraw 	constraint	
 firmware hdl CMSIS drivers drivers_config hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MS5_MS5_CM3_0.eww Voltage_Monitor_MS5_MS5_CM3_0_app.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis viewdraw 	coreconsole	
hdl CMSIS drivers drivers drivers.config hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MSS_MSS_CM3_0.eww Voltage_Monitor_MSS_MSS_CM3_0_app.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation smartgen simulus viewdraw	🖶 🔁 designer	
 IAR CMSIS drivers drivers hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MSS_MSS_CM3_0.eww Voltage_Monitor_MSS_MSS_CM3_0_app.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis viewdraw 		
CMSIS drivers drivers drivers.config hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MSS_MSS_CM3_0.eww Voltage_Monitor_MSS_MSS_CM3_0_app.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.icf Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus yinthesis viewdraw	a transferred to the second se	
 drivers drivers_config hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MSS_MSS_CM3_0.eww Voltage_Monitor_MSS_MSS_CM3_0_app.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.icf Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus yinthesis viewdraw 	📮 🗀 IAR	1
 drivers_config hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MSS_MSS_CM3_0.eww Voltage_Monitor_MSS_MSS_CM3_0_app.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.ewp Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.icf Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis viewdraw 		
 hal main.c Voltage_Monitor_hw_platform.h Voltage_Monitor_MS5_MS5_CM3_0.eww Voltage_Monitor_MS5_MS5_CM3_0_app.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.icf Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis viewdraw 		1
 main.c Woltage_Monitor_hw_platform.h Woltage_Monitor_MSS_MSS_CM3_0.eww Woltage_Monitor_MSS_MSS_CM3_0_app.ewp Woltage_Monitor_MSS_MSS_CM3_0_hw_platform.ewp Woltage_Monitor_MSS_MSS_CM3_0_hw_platform.icf Woltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus yinthesis viewdraw 		
 Woltage_Monitor_hw_platform.h Woltage_Monitor_MSS_MSS_CM3_0.eww Woltage_Monitor_MSS_MSS_CM3_0_app.ewp Woltage_Monitor_MSS_MSS_CM3_0_hw_platform.ewp Woltage_Monitor_MSS_MSS_CM3_0_hw_platform.icf Woltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation simulation synthesis viewdraw 	🖶 🖶 🧮 hal	1
 Voltage_Monitor_MS5_MS5_CM3_0.eww Voltage_Monitor_MS5_MS5_CM3_0_app.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.icf Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus yinthesis viewdraw 		
 Voltage_Monitor_MS5_MS5_CM3_0_app.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis viewdraw 		1
 Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.ewp Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.icf Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis viewdraw 		
 Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.icf Voltage_Monitor_MS5_MS5_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis viewdraw 	Voltage_Monitor_MSS_MSS_CM3_0_app.ewp	1
 Woltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac phy_synthesis simulation smartgen stimulus synthesis yinthesis 		
phy_synthesis simulation smartgen stimulus synthesis viewdraw		1
simulation smartgen stimulus synthesis viewdraw	Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.mac	
smartgen stimulus synthesis viewdraw		
stimulus synthesis 	🖶 🔁 simulation	
synthesis	smartgen	
• viewdraw	📄 🖳 stimulus	
Design Hierarchy Files Cores	🗄 💼 viewdraw	
Design Hierarchy Files Cores		
Design Hierarchy Files Cores		
Design Hierarchy Files Cores		
Design Hierarchy Files Cores		_
	Design Hierarchy Files Cores	

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 Build button, 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 to the *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.

	Project	File	Edit	View	Design	Tools	Help	
		3			≥ 0	â		
0)esign Hi	erarchy	/			Generat	e Programming Data	₽×

Figure 24 · Generating Programming Data

2. The **Design Flow** window looks similar Figure $25 \cdot .$

Voltage_Mor	nitor				-	0		Ø,
	Тоо							
	4	•	Cre	ate Design				
V				Configure MSS				
			SD	Create SmartDesign				
			Ê	Create HDL				
				Create HDL TestBench				
			9	View/Configure Firmw	are Co	res		
		\triangleright	•	Verify Pre-Synthesize	ed Desi	gn		
	\triangleright	•	Со	nstrain Design				
	4	€	Im	plement Design				
V		\triangleright	G	Synthesize				
		⊿	•	Verify Post-Synthesis	Impler	nenta	tion	
				Simulate				
V		\triangleright		Compile				
		\triangleright	•	Constrain Place and R	oute			
V			$O_{l_{O}}$	Place and Route				
V		4	•	Verify Post Layout Im	pleme	ntatio	n	
				🗮 Simulate				
v				🖎 Verify Timing				
				陷 Verify Power				
			O	Export Back Annotate	ed Files			
V			0101	Generate Programmi	ing Dat	а		
	⊿	∢	Pro	gram Design				
			010	Program Device				
	\triangleright	€	De	bug Design				
	⊿	€	Ha	ndoff Design for Prod	uction			
			-2	Export Programming F	ile			
			•[]	Export Pin Report				
			•[]	Export IBIS Model				
	⊿	€	De	velop Firmware				
				Write Application Cod	e			

Figure 25 · Design Flow Window After Building the Project



Step 5 - Programming SmartFusion Board Using FlashPro

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

Jumper Settings for SmartFusion Evaluation Kit Board

- 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 E).

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

- 1. Double click **Program Device** under **Program Design** in the **Design Flow** window to program the SmartFusion cSoC device.
- 2. Click Yes when it prompts that the I/O and timing constraints 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.

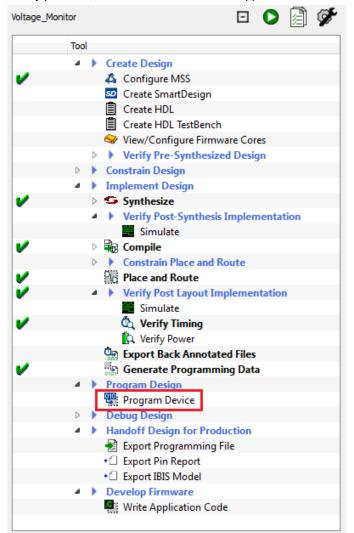


Figure 26 · Design Flow Window

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



Step 6 - Building the Software Application Through IAR Embedded Workbench

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



Figure 27 · Invoking IAR from Libero SoC

2. Your IAR workbench perspective will look like Figure 28 · .

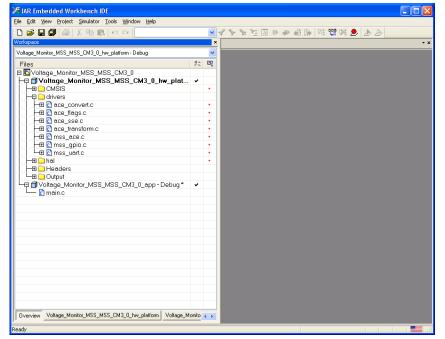


Figure 28 · IAR Workbench



 Copy the code provided below and paste it in main.c file under SmartFusion_UART_HW_MSS_MSS_CM3_0_app project in the IAR 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 \
*
           ******
                       *****
                                          ****
                                                 ****
                                                          *****
                                                                           * ******
...
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( TM0_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\b", channel name, ((float)(-adc value mv) /
(float)(1000)));
       }
       else
       {
         snprintf((char*)display buffer, sizeof(display buffer),
                "%s : %.3fV\r\b", 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(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);

4. To configure the options for the project, right-click the project name (*Voltage_Monitor_MSS_MSS_CM30_app*) and click **Options**.

File Edit View Project Sinulator Tools Window Help Image: Monitor_MSS_MSS_CM3_0_two_platform-Debug Image: Monitor_MSS_MSS_CM3_0_two platform-Debug Image: Monitor_MSS_MSS_CM3_0_two platform-De	🔀 IAR Embedded Workbench IDE	
Workspace × Voltage_Monitor_MSS_MSS_CM3_0 * © Phone * © Output * © Make * © Output * © Stap Build * * * * © Output * * © Monitor_MSS_MSS_CM3_0 * © Containing Foder * © Monitor_MSS_MSS_CM3_0	File Edit View Project Simulator Tools Window Help	
Voltage_Monitor_MSS_MSS_CM3_0 (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Comportation All rights reserved. * (c) Copyright 2011 Microsemi Comportation All rights reserved. * (c) Copyright 2011 Microsemi Comportation All rights reserved. * (c) Copyright 2011 Microsemi Comportation All rights reserved. * (c) Copyright 2011 Microsemi Comportation All rights reserved. * (c) Copyright 2011 Microsemi Comportation All rights reserved. * (c) Copyright 2011 Microsemi Comportation Microsemi Comportation Microsemi Comportation All rights reserved. * (c) Copyright 2011 Microsemi Comportation Microsemi Comportation Microsemi Comportation Microsemi Comportation Microsemi Comportation Microsem	D 📽 🖬 🕼 👗 🖻 💼 🗠 🖂	🗹 🗸 🍾 独 函 🖻 🗭 🦛 🏟 🏠 🕼 🔛 🕭
Files * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * Sample test program for the SmartFusion ACE. DMO is used to send the AL * CMSIS * ace_conventc * ace_flags c * ace_stesc * ace_stesc * B mss_copic * B moutc * Comple * Comple * Stop Build * * * * * * * * * * * * * * * * * * *	Workspace ×	main.c ace_transform.c **
# Output #include "drivers\mss_uart.h" Woltage_Monitor_MSS_MSS_CM3_0 • # Make ude "drivers\mss_ace\mss_ace.h" ude "drivers\mss_gpio\mss_gpio.h" ude "drivers\mss_gpio.h" ude "drivers\mss_gpio.h" ude "drivers\mss_gpio.h" ude "drivers\mss_gpio.h" ude 'drivers\mss_gpio.h" ude 'drivers\mss_gpio.h" ude 'drivers\mss_gpio.h" ude 'drivers\mss_gpio.h" ude <stdio.ho< td=""> Compile ne Microsemi_logo \ Rebuild All Glean Add Version Control System Open Containing Folder File Properties Set as Active </stdio.ho<>	Voltage_Monitor_MSS_MSS_CM3_0_hw_platform - Debug Files © Voltage_Monitor_MSS_MSS_CM3_0 Image Voltage_Monitor_MSS_MSS_CM3 Image Voltage Voltage_Monitor_MSS_MSS_CM3 Image Voltage Voltage Voltage Image Voltage Image Voltage Image Voltage	<pre>* (c) Copyright 2011 Microsemi Corporation. All rights reserved. * * (c) Copyright 2011 Microsemi Corporation. All rights reserved. * * Sample test program for the SmartFusion ACE. IN0 is used to monit * voltage across the potentiometer. The UAET is used to send the Al * to a terminal program. The hardware configuration has four flags: * - over 1.0v * - over 1.5v * - over 1.5v * - over 2.5v * The flag values are displayed on the SmartFusion Evaluation Kit by * Author: Rmit Garg * Last modified : September 12th, 2011 * Tested with : IAF 06.21 Last modified is is knowing from Libero SoC</pre>
Build Debug Log	Image Monitor_MSS_MSS_CM3_0_a Imain.c Options Imain.c Options Imain.c Complex Imain.c Complex Imain.c Complex Imain.c Stop Build Add Remove Rename Version Control Sys Overview Voltage_Monitor_MSS_MSS_CM3_0	<pre>ude "drivers\mss_ace\mss_ace.h" ude "drivers\mss_gpio\mss_gpio.h" ude <stdio.h> ne Microsemi_logo \ `` `** ******* **********************</stdio.h></pre>

Figure 29 · Selecting Options for the Project

The **Options** available for the node Voltage_Monitor_MSS_MSS_CM30_app window are displayed in Figure $30 \cdot$.



Options for node "Vo	oltage_Monitor_MSS_MSS_CM3_0_app"	×
Options for node "Vo Category: General Options C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel GDB Server IAR ROM-monitor	Target Output Library Configuration Library Options MISRA-C:200 Processor variant Ore Core Cortex-M3 Et	>
J-Link/J-Trace TI Stellaris FTDI Macraigor	Endian mode EPU Solutile None	
PE micro RDI ST-LINK Third-Party Driver TI XDS100	● BE32 ● BE <u>8</u>	
	OK Cancel	

Figure 30 · Accessing the Options for Your Project

5. Click Linker under Category and add the Linker configure file

(*Voltage_Monitor_MSS_MSS_CM3_0_hw_platform.icf* file,). Add the file that is available at C:\Microsemiprj\POT_LED_Libero_IAR\Voltage_Monitor\IAR folder by enabling the **Override** default. The *.icf file is a linker configuration file, which defines the segmentation of memory.

Options for node "Vo	ltage_Monitor_MSS_MSS_CM3_0_app"
Category: General Options C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel GDB Server IAR ROM-monitor J-Link/J-Trace TI Stellaris FTDI Macraigor PE micro RDI ST-LINK Third-Party Driver TI XDS100	Factory Settings Config Library Input Optimizations Output List #define Image: Additional content of the symbol content of the symbol content of the symbol definitions: (one per line) Image: Configuration file Image: Configuration file Image: Configuration file Image: Configuration file
	OK Cancel

Figure 31 · Adding the Linker Configuration File



- 6. Click Edit to display the Linker configuration file editor, as shown in Figure 32 · . This window displays the Vector Table, Memory Regions, and Stack/Heap Sizes tabs.
- 7. Click on the Memory Regions tab to view the RAM/ROM regions.

Linker configuration file editor						
Vector Table	Memory Regions	Stack/Heap Sizes				
ROM	Start: 0x20000000	End: 0x20007FFF				
RAM	0×20008000	0x2000FFFF				
Save Cancel						

Figure 32 · The Linker Configuration File Editor – Memory Regions Tab

8. Click the Stack/Heap Sizes tab to view the Stack/Heap sizes. Click Save.

Linker confi	Linker configuration file editor						
Vector Table	Memory Regions	Stack/Heap Sizes					
CSTACK	0×400						
HEAP	0×800						
		Save	Cancel				

Figure 33 · The Linker Configuration File Editor – Stack/Heap Sizes Tab

9. Click the **Debugger** category. Under the **Setup** tab, select the **Driver** as **J-Link/J-Trace** using the dropdown list.

Options for node "Vo	ltage_Monitor_MSS_MSS_CM3_0_app"	
Category: General Options C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel GDB Server IAR ROM-monitor J-Link/J-Trace TI Stellaris FTDI Macraigor PE micro RDI ST-LINK Third-Party Driver TI XDS100	Setup Download Images Extra Options Plugins Driver Diver Simulator Angel GDB Server HAR ROM-monitor JLink/J-Trace TI Stellaris FTDI Macreigor PE micro RDI ST-LINK Third-Party Driver TI XDS100	ary Settings
	OK Ca	ncel

Figure 34 · Selecting the Driver as J-Link/J-Trace for the Debugger



- Select the Use macro file(s) listed under Setup macros (see Figure 35 ·). Browse to the Voltage_Monitor _MSS_MSS_CM3_0_hw_platform.mac file which is available in which is available in C:\ Microsemiprj\POT_LED_Libero_IAR\Voltage_Monitor\IAR folder. The purpose of this file is to remap the ESRAM to the 0th location.
- Select the Override default option listed under the Device description file (see Figure 35 ·). Browse to the SmartFusion_dss.ddf file which is available in C:\ Microsemiprj\POT_LED_Libero_IAR\Voltage_Monitor\IAR folder. DDF files allows user to see the MSS Register map for all peripherals.
 - Note: The Debugger files provided in the attached zip folder have been extracted to the project directory where you will store the EWARM project. For example, C:\Microsemiprj\UART_Libero_IAR\SmartFusion_UART_HW\IAR.

Options for node "Vo	oltage_Monitor_MSS_MSS_CM3_0_app"	×
Category:		
General Options C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel GDB Server IAR ROM-monitor J-Link/J-Trace TI Stellaris FTDI Macraigor PE micro RDI ST-LINK Third-Party Driver TI XDS100	Setup Download Images Extra Options Plugins Driver	ngs
	OK Cancel)

Figure 35 · Setting up Macros for the Debugger

12. Under the Download tab, select Verify download.

Options for node "Vo	oltage_Monitor_MSS_MSS_CM3_0_app"	\mathbf{X}
Category: General Options C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel GDB Server IAR ROM-monitor J-Link/J-Trace TI Stellaris FTDI Macraigor PE micro RDI ST-LINK Third-Party Driver TI XDS100	Factory Settings Setup Download Attach to program Verifu download Suppress download Use flash loader(s) Override default.board file \$TOOLKIT_DIR\$\config\flashloader\ Edit	
	OK Cancel	

Figure 36 · Download Verification Settings for the Debugger

13. Click J-Link/J-Trace category. Under the Setup tab, select Reset as Core as shown in Figure 37 · :

Options for node "Vo	oltage_Monitor_MSS_MSS_CM3_0_app"
Category: General Options C/C++ Compiler Assembler Output Converter Custom Build Build Actions Linker Debugger Simulator Angel GDB Server IAR ROM-monitor J-Link/J-Trace TI Stellaris FTDI Macraigor PE micro RDI ST-LINK Third-Party Driver TI XDS100	Factory Settings Setup Connection Breakpoints Core JTAG/SWD speed Initial 32 KHz Clock setup CPU clock: 72.0 MHz SWO clock: Auto 2000 KHz CPU clock:
	OK Cancel

Figure 37 · J-Link/J-Trace Setup

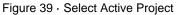
- 14. Click **OK** to close this window and build the project.
- 15. Right-click on Voltage_Monitor_MSS_MSS_CM3_0_hw_platform and select Make.

AR Embedded Workbench IDE	
	✓ ✓ > > > E B > > A B B B B W S D D D
/orkspace ×	main.c
Voltage_Monitor_MSS_MSS_CM3_0_Hw_platform - Debug Image: Monitor_MSS_MSS_CM3_0 Image: Monitor_MSS_MSS_CM3_0 Image: Monitor_MSS_MSS_CM3_0 Image: Monitor_MSS_MSS_CM3_0_hw_platform Image: Monitor_MSS_MSS_CM3_0_hw_platform Image: Monitor_MSS_MSS_CM3_0_hw_platform Image: Monitor_MSS_MSS_CM3_0_hw_platform Image: Monitor_MSS_MSS_CM3_0_hw_platform Image: Monitor_MSS_MSS_CM3_0_hw_platform	<pre>* (c) Copyright 2011 Microsemi Corporation. All rights reserved. * Sample test program for the SmartFusion ACE. TH0 is used to monito. * voltage across the potentiometer. The UART is used to send the ADC * to a terminal program. The hardware configuration has four flags: * - over 1.0v * - over 1.0v * - over 2.0v * - over 2.0v * - over 2.0v * Last modified : September 12th, 2011 * Tested with : 1AR V6.21 * Last modification : Running from Libero SoC ************************************</pre>

- Figure 38 · Make the Hardware Platform
- 16. Right-click on Voltage_Monitor_MSS_MSS_CM3_0_app project name and select Set as Active.



🔀 IAR Embedded Workbench IDE	
File Edit View Project Simulator Tools Window Help	
D 🖆 🖬 🕼 🐇 🐘 💼 🗠 🖂	✓ ✓ > > > 2 2 3 4 5 3 4 5 4 5 5 6 5 <
Workspace	× main.c
Voltage_Monitor_MSS_MSS_CM3_0_hw_platform - Debug	7************************************
Files # <td><pre>* (c) Copyright 2011 Microsemi Corporation. All rights reserved. * * Sample test program for the SmartFusion ACE. INO is used to monitor * voltage across the potentiometer. The UART is used to send the ADC * over 1.0v * - over 1.5v * - over 1.5v * - over 2.5v * - over 2.5v * The flag values are displayed on the SmartFusion Evaluation Kit boa: * Author: Amit Garg * Last modified : September 12th, 2011 * Tested with : IAR v6.21 * Last modification : Running from Libero SoC ************************************</pre></td>	<pre>* (c) Copyright 2011 Microsemi Corporation. All rights reserved. * * Sample test program for the SmartFusion ACE. INO is used to monitor * voltage across the potentiometer. The UART is used to send the ADC * over 1.0v * - over 1.5v * - over 1.5v * - over 2.5v * - over 2.5v * The flag values are displayed on the SmartFusion Evaluation Kit boa: * Author: Amit Garg * Last modified : September 12th, 2011 * Tested with : IAR v6.21 * Last modification : Running from Libero SoC ************************************</pre>
Overview Voltage_Monitor_MSS_MSS_CM3_0Stop Build	** ****** ***** ***** ***** ***** *****
Messages Add startup_a2fxxm3.s system a2fxxm3.c Rename	File Line 🔺
Building library Version Cont	crol System 🔸
Total number of warnings: 0	
Set as Active	
Set as active project	Errors 0, Warnings 0



17. Right-click on the Voltage_Monitor_MSS_MSS_CM3_0_app project name and select Clean.

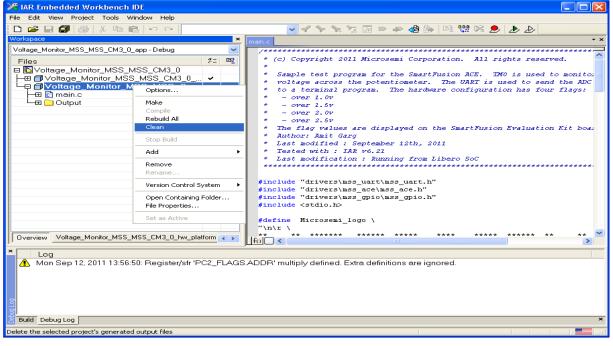


Figure 40 · Building a Clean Project



18. After cleaning the project, the Messages log window shows that some files were deleted.



Figure 41 · Files Deleted Message

- 19. Right-click on the Voltage_Monitor_MSS_MSS_CM3_0_app project name and click Rebuild All.
- 20. Ensure that no errors appear in the **Messages** log window and follow the next steps to configure the HyperTerminal.



Figure 42 · Error-Free Message Log

Step 7 - Configuring the Serial Terminal Emulation Program

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

- 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.
- If the SFE USB to RS232 Controller drivers are automatically detected (this can be verified in Device Manager), as shown in Figure 43 · , proceed to next step; otherwise follow the Step 8 - Installing Drivers for the USB to RS232 Bridge.

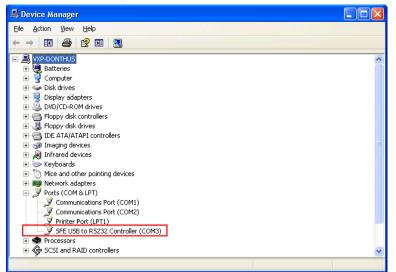


Figure 43 · Device Manager Listing SFE USB to RS232 Controller Drivers



- From the Windows Start menu, select Programs > Accessories > Communications > HyperTerminal. This opens HyperTerminal. If your PC does not have HyperTerminal, use any free serial terminal emulation program like PuTTY or Tera Term. Refer to the *Configuring Serial Terminal Emulation Programs* tutorial for configuring the HyperTerminal, Tera Term, and PuTTY.
- 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	
lcon:	
	>
OK Ca	ncel

Figure 44 · 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	? 🔀
Nyperter	minal
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 45 · 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

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> estore Defaults
	K Cancel Apply

Figure 46 · Setting the COM Properties

7. Click OK to close the UART_Hyperterminal Properties dialog box.

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

Step 8 - Installing Drivers for the USB to RS232 Bridge

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

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

- 1. Unzip the USB_Drivers.zip file and browse to the USB Drivers folder. Refer to the instructions in the readme.txt file contained in the zip file.
- 2. Double click (run) **Preinstaller.exe**.
- 3. Accept the default installation location and click Install.
- 4. Click Continue Anyway if prompted.
- 5. When the installation is complete, click **OK**. The Ports (COM & LPT) section of the Device Manager lists SFE USB to RS232 Controller under the Ports section of the Device Manager.

Step 9 - Debugging the Application Project Using IAR

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

- 1. Connect J-link Box to Board and PC using the following connection tips:
 - Connect the J-Link ARM to the RVI-Header of the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board.
 - JP7: Connect pin 2 and 3 for IAR debugging mode.
 - JP10: Connect pin 2 and 3.
 - J6: Connect pin 1 and 2.



2. In the IAR Workbench click Download and Debug:



Figure 47 · Download and Debug the Project

3. The Workbench window is displayed as shown in Figure $48 \cdot$.

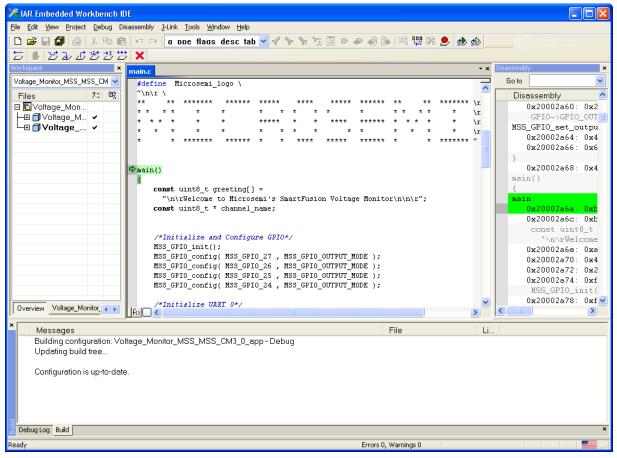


Figure 48 · IAR Workbench After Downloading the Project

4. Click Go.

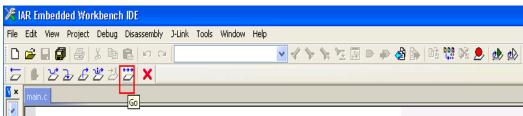


Figure 49 · Run the Project



5. Observe the HyperTerminal window. It should display the greeting message with Microsemi name as shown in Figure 50 \cdot .

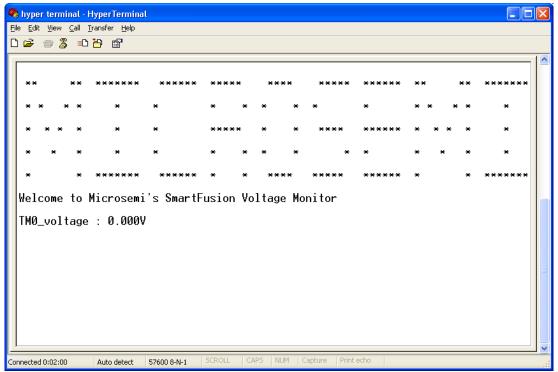


Figure 50 · UART HyperTerminal Window

Note: If you try typing some alphabetic characters, you will find they are echoed in the terminal window.

6. To observe the state of the UART0 registers, click **View > Register**. In the **Register** pane, select **UART_0** from the drop-down list.

Register	×
-	1
Current CPU Registers 🛛 👻	
Current CPU Registers	
CPU Registers	
UART 0	
SPI_0	
L 12C_0	
Ethernet MAC	
Peripheral DMA	
Timer	
Watchdog	
H2F Interrupt Controller	
UART_1	
SPI_1	
12C_1	
GPIO	
RTC	
FROM	
AP Controller	
Analog Conversion Engine	
⊟ System Register ⊐ NVIC	
±EPSR =	J
PC =	
PRIMASK =	
BASEPRI =	
BASEPRI_MAX =	
FAULTHASK =	
THCONTROL =	

Figure 51 · Current CPU Registers



Register		×
UART	0	✓
RBR	=	0x4E
THR	=	0x4E
DLR	=	0x4E
DMR	=	0x00
🗄 I E R	=	0x00
HIIR	=	0xC1
ICR	=	0x03
⊞∎CR	=	0x00
ISR	=	0x00
⊞∎SR	=	0xF0
SR	=	0x00

Figure 52 · UART_0 Registers

- 7. Move the POT on the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board. The voltage measurement is displayed on HyperTerminal and the LEDs are illuminated on the SmartFusion Evaluation Kit Board or the SmartFusion Development Kit Board when one of the voltage monitor flags is asserted.
- 8. Adjust the POT and observe that the voltage measurement is continuously updated.

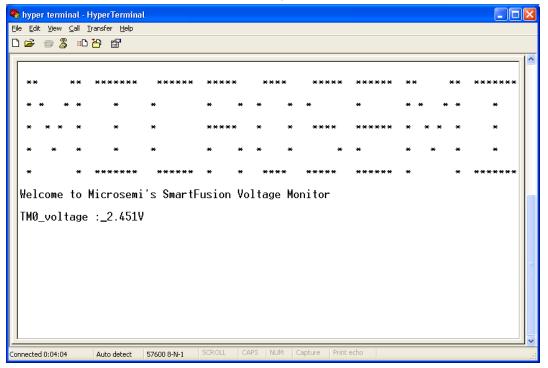


Figure 53 · HyperTerminal Window (2.451 V)

- 9. Observe the state of the LEDs as the POT is adjusted. Confirm that the flags work as specified in the ACE configurator.
- 10. To stop debugging click Stop Debugging.

🔀 IAR Embedded Workbench IDE								
File	Edit	View	Project	Debug	Disa	issembly	J-Link	Tools
D	2	8	@	<u>ж</u> е	a	00	r	
D	6	B	2 £	₩ <u>≯</u>	·	X		
×	main	.c				Sto	p Debug	ging

Figure 54 · Stop Debugging

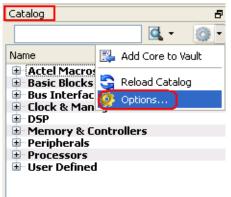
This concludes the tutorial.

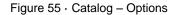


Appendix A – Libero SoC Catalog Settings

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** 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 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		OK Cancel

Figure 56 · 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	?	\times
	Current vault location: C:\Documents and Settings\donthus\Application I	
	Select new <u>vault location</u> :	
	C:\Documents and Settings\donthus\Application Data\Actel\tools ()]
Help	OK Cancel	

Figure 57 · 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):	
	sicones			A -
display only the latest version of a core				<u> </u>
ame	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		
ardware 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:				^
AL RN.pdf				
ree software license.pdf				
escription: The Hardware Abstraction Laye	er is used by drivers to	access the hardwa	are. It also allow the control of interrup	its.
he HAL is normally required when using bare				
ompatible IP cores: stal:DiractCore:CODE90E15: from uproion 1				~
New cores are available for download	Download them now		💽 Download 🛛 🟥 🤇	Generate

Figure 58 · 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 59 · , and click OK.

🕵 Confi	guring MSS_G	PIO_0 (MSS_GPIO	- 1.0.101)					
Config	guration							^
	-Multiplexed With	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 With	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 With	h SPI1 Peripheral Dedic	ated I/Os					
	GPIO_27:	Use as MSS I/O Pad	Not Used	✓ V	V21	or connect to Fabric	Not Used 💌	
		Use as MSS I/O Pad	Not Used	Y A	A22	or connect to Fabric		
Help							ОК	Cancel .:

Figure 59 · Configure MSS_GPIO_0

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

2. Click File > Save to save the Voltage_Monitor_MSS.

Generating the MSS Component

1. Right-click on Voltage_Monitor_MSS_0 component on the Voltage_Monitor tab and select Update Instance(s) with Latest Component.

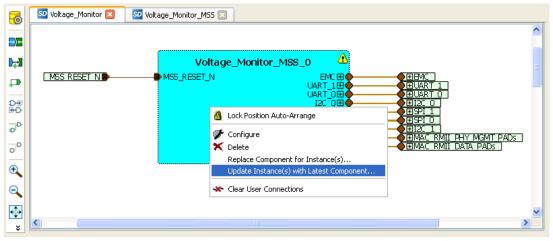


Figure 60 · Updating the MSS



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

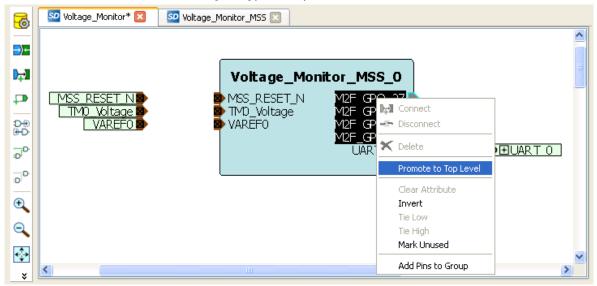


Figure 61 · GPIO Pins Promoted to Top Level

3. Click **Design > Configure Firmware** as shown in Figure $62 \cdot .$



Figure 62 · Opening Design_Firmware

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

SD	Voltage_Moni	tor 🗈	🕄 📔 🕺 Voltage_Monitor_MSS 🔀	🕺 DESIGN_FIRMWARE* 🔀			
	Generate	enerate Instance Name		Core Type	Version	Compatible Hardware Instance	
1		Ø,	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_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 63 · Firmware Configuration Settings – CMSIS Peripheral

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

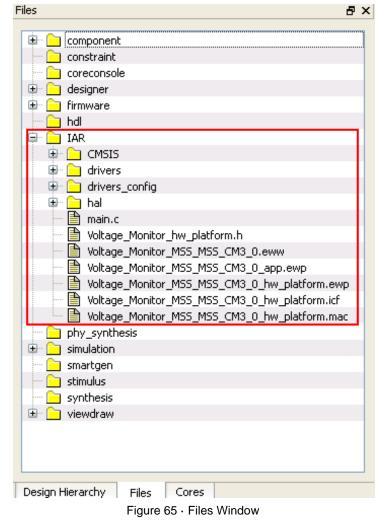


7. Save the design and generate the component by clicking **Generate Component** or by selecting **SmartDesign > Generate Component**.

Gen	SO Voltage_Monitor* SO Voltage_Mon erate Component	nitor_MSS 🔀		
₽		Voltage_Mor	iitor_MSS_0	
0 0 0 0	MSS_RESET_N TM0_Voltage VAREF0	MSS_RESET_N TM0_Voltage VAREF0	M2F_GPO_31 M2F_GPO_30 M2F_GPO_29 M2F_GPO_28	M2F GPO 31 M2F GPO 30 M2F GPO 29 M2F GPO 28
₽° •			UART_0⊞∳ IP↓	● EUART_0
⊇ ×		IIII		>

Figure 64 · Generating the MSS Component

- 8. 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 IAR folder is created with the folders and files as shown in Figure 65 · .





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 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.

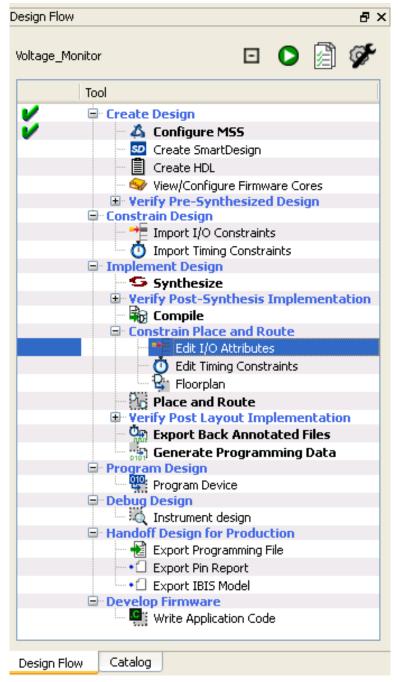


Figure 66 · Edit I/O Attributes



- 2. Make the following pin assignments in MultiView Navigator window as shown in Figure 67 · .
 - GPO_28 to B19
 - GPO_29 to B20
 - GPO_30 to C19
 - GPO_31 to H17

🐊 MultiView Navigator [Voltage_	Monitor *] - [I/O Attrib	ute Editor]						
➡ Eile Edit View Logic Format T	ools <u>W</u> indow <u>H</u> elp						-	Ξ×
	5 ゆ A 図 例 f	o Đ 🐴 🕯		⊿ 0 ▦	0 7	🕺 📲 🤺	176 1	⁺n
Logical	Port Name A	Group	Macro Cell	Pin Number	Locked	Bank Name	1/0 Standard	Out _i Drive
⊡ B M2F_GP0_29_pad	1 M2F_GP0_28		ADLIB:OUTB	B19	V	BankO	LVTTL	12
	2 M2F_GP0_29		ADLIB:OUTB	B20	V	BankO	LVTTL	12
	3 M2F_GP0_30		ADLIB:OUTB	C19	V	BankO	LVTTL	12
	4 M2F_GP0_31		ADLIB:OUTB	H17 ⊻		Bank1	LVTTL	12
	5 MSS_RESET		ADLIB:INBUF	R1		Bank2	LVTTL	
	<	ackage Pins	1					>
×	× ◆ ◆ Output / Res	ults 👌 Find 1	1					
Ready			row 4, col 4 F	FAM: SmartFusio	DIE: A2	F200M3F PAC	KAGE: 484 FB	GA

Figure 67 · MultiView Navigator GUI

- 3. 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.

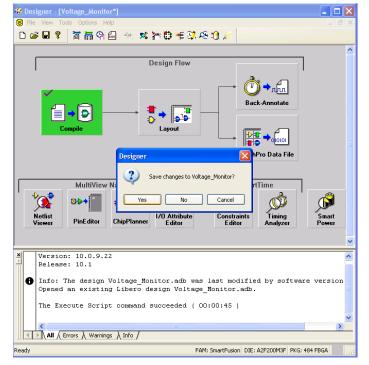


Figure 68 · Designer Window



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

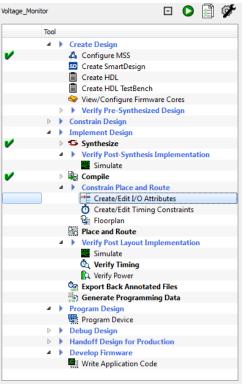
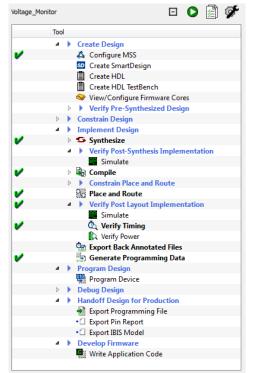


Figure 69 · Generating Programming Data

7. The **Design Flow** window looks as shown in Figure $70 \cdot .$



- Figure 70 · 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 Software Requirements under Tutorial Requirements section (SAR 38349)			
(May 2012)	Replaced Figure 6 · (SAR 38349)			
	Replaced Figure 21 · (SAR 38349)			
	Replaced Figure 25 · (SAR 38349)	16		
	Replaced Figure 26 · (SAR 38349)	18		
	Replaced Figure 27 · (SAR 38349)	19		
	Modified Step 6 - Building the Software Application Through IAR Embedded Workbench (SAR 38349)	19		
	Replaced Figure 69 · and Figure 70 · (SAR 38349)	44		
Revision 5 (November 2011)	Updated the document for Libero SoC v10.0 (SAR 35044).			

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



Product Support

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

Customer Service

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

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

Customer Technical Support Center

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

Technical Support

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

Website

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

Contacting the Customer Technical Support Center

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

Email

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

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

My Cases

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



Outside the U.S.

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

ITAR Technical Support

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



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