



Microchip FPGA BU, San Jose, CA 95134

© 2020 Microchip Corporation. All rights reserved. Permission granted for Arrow to use this document.

Printed in the United States of America

Release: October 2020

Microchip makes no warranties with respect to this documentation and disclaims any implied warranties of merchantability or fitness for a particular purpose. Information in this document is subject to change without notice. Microchip assumes no responsibility for any errors that may appear in this document.

Trademarks

Microchip, Microchip SmartFusion2, IGLOO2 and the associated logos are trademarks or registered trademarks of Microchip Corporation. All other trademarks and service marks are the property of their respective owners.



Table of Contents

| SmartFusion2 ARM Cortex-M3 Lab Guide1 |
|--|
| Introduction |
| Components of SmartFusion2 Device Used |
| Tutorial Requirements |
| Software Requirements5 |
| Hardware Requirements |
| Extracting the source files |
| Step 1 – Creating a Libero SoC Project |
| Launching Libero SoC6 |
| Checking Tool Profiles and IP Cores7 |
| Creating the Libero SoC Project7 |
| Step 2 – Configuring the SmartFusion2 MSS with System Builder9 |
| Step 3 – Constraining the Design |
| Importing an IO Constraint file27 |
| Generating Timing Constraints29 |
| Step 4 – Synthesis and Layout |
| Step 5 – Running the Application |
| Step 6 – Generating Sample Projects and Exporting the Firmware Configuration Files |
| Generating sample projects |
| Exporting firmware configuration files and firmware drivers37 |
| Step 7 - Debugging with SoftConsole v6.4 41 |
| Running the CorePWM slow blink application41 |
| Confirming the SoftConsole v6.4 Project Settings49 |
| Building the Project |
| Debugging with SoftConsole |
| Step 8 - Running the RTC_time application (time permitting) |
| Step 9 - Debugging code running from the SmartFusion2 eNVM |
| Building the project with the Release build configuration68 |
| List of Figures |



Introduction

This tutorial demonstrates how to implement a basic SmartFusion2 Microcontroller Subsystem (MSS) configuration that includes the GPIO, MMUART_0, the RTC and a soft PWM (CorePWM) in the SmartFusion2 fabric using System Builder.

The SmartFusion2 GPIO[8] will be configured as an input. GPIO[8] will be connected to one of the switches on the SmartFusion2 SMF2000. The fabric PWM[8:1] will be configured as 16 bit PWM; outputs will drive LEDs on the SmartFusion2 SMF2000;

After completing this tutorial, you will be familiar with the following:

- Using a Tcl script to create a Libero SoC project
- Using System Builder to configure the SmartFusion2 MSS, add fabric peripherals and generate the design
- Constraining the design for synthesis and layout
- Synthesize the SmartFusion2 design with Synplify Pro ME
- Run layout
- Generate a bitstream
- Program the SmartFusion2 silicon on the SMF2000 board
- Creating firmware configuration files and Sample projects from Libero SoC
- Using SoftConsole v6.4 to create and debug SmartFusion2 applications



SMF2000 board

Online Reference Manual:

https://wiki.trenz-electronic.de/display/PD/TEM0001+TRM

Schematics:

http://www.trenz-electronic.de/fileadmin/docs/Trenz_Electronic/Modules_and_Module_Carriers/2.5x6.15/TEM0001/REV01/Documents/SCH-TEM0001-01-010C.PDF



SmartFusion2 ARM Cortex-M3 Lab Guide



Figure 1 - SmartFusion2 Block Diagram

Components of SmartFusion2 Device Used

This tutorial uses the SmartFusion2 Cortex-M3, the MSS GPIO block, MMUART_0, the RTC and the FPGA fabric.

Tutorial Requirements

Software Requirements

- Microsemi Libero SoC v12.5
- Synplify Pro Q-2020.03M
- Identify Debugger Q-2020.03M
- FlashPro Express v12.5
- Microchip SoftConsole v6.4
- Serial terminal program, such as PuTTY or TeraTerm

SmartFusion2 MSS Component versions

The table below lists the version of the SmartFusion2 MSS which must be used.

| Libero SoC Version | SmartFusion2 MSS version | System Builder |
|--------------------|--------------------------|----------------|
| 12.5 | 1.1.500 | 1.0 |

 Table 1 – SmartFusion2 MSS component versions



Hardware Requirements

This tutorial targets the Trenz Board SMF2000 with SmartFusion2 on it. The free Libero SoC Silver license can be used for this tutorial. ModelSimPro (mixed language simulation) is supported by a Silver license generated after May 22nd 2020.

Extracting the source files

Use *SMF2000_Cortex_M3_PWM_lab_sources.zip* to extract the required lab files to the *C:> or D:>\Microsemiprj* folder on the HDD of your PC. Confirm that a folder named *SMF2000_Cortex_M3_PWM_lab_sources* containing 3 sub-folders named *constraints, hex_File* and *Script* were extracted.

Step 1 - Creating a Libero SoC Project

In this step, you will create a Libero SoC v12.5 project.

Launching Libero SoC

 Click Start > Programs > Microsemi Libero SoC v12.5 > Libero SoC v12.5 or click the shortcut on your desktop. The Libero SoC Project Manager will open.

| 📀 Libero | - | | 2 |
|--|--|-----------|---|
| Project <u>File Edit V</u> iew Design Tools <u>H</u> elp | | | |
| | | | |
| StartPage | EP Min | | |
| | | | _ |
| Projects | D | • | |
| New | | | |
| Open | Welcome to Microsemi's Libero® SoC v12.5 | | |
| Recent Projects | Microsemi Libero® System-on-Chip (SoC) design suite offers high productivity with its comprehensive, easy to learn, easy to adopt development tools for designing with Microsemi's power efficient flas' FPGAs, SoC FPGAs, and Rad-Tolerant FPGAs. The suite integrates industry standard Synopsys Symplify Pro@ synthesis and Mentor Graphics ModelSm@ simulation with best-in-class constraints | sh | |
| C:/tmp/test67/test67 C:/tmp/test6/test6 | management, debug capabilities, and secure production programming support. | | |
| C:/Microsemipri/PF/Controller Sim C:/Projects/gige/hd/top level | What's New in Libero SoC v12.5 | | |
| C:/tmp/ PE_Core10GBa/PHY_Evaluation | Introducing PolarFire SoC family - for further information, refer to PolarFire SoC FPGA | | |
| | Preliminary timing and power for MPFS250T_ES EXT device along with programming and SmartDebug | | |
| | Preliminary timing and power for MPFS250T/L EXT/IND device | | |
| Links | Advance timing and power for MPFS460T/L EXT/IND device | | |
| Welcome to Libero SoC | Standalone MSS configurator (,bin64/pfsoc_mss) and import of the generated MSS component into Libero SoC | | |
| Libero SoC Quickstart | | | |
| Libero SoC Interface Description | PolarFire Silcon support | | |
| Libero SoC Release Notes on the Web | MIL operating condition - Production timing for the MPF200TS, MPF300TS and MPF500TS 1.05V/STD devices | | |
| Libero Tutorials | New Lidless package FCG784N for MPF300T/S, STD/-1, IND | | |
| Product Tutorials | Transceiver enhancements | | |
| Training Webcasts | Enhanced Receiver Management (ERM) - New static DFE calibration option for Signal Integrity | | |
| Microsemi SoC Website | All RX_CTLE settings fully validated and can be considered as production | | |
| | DDR enhancements | - | |
| | | | |
| Message | | ð | × |
| 🔳 Messages 🔞 Errors 🔺 Warnings 🌐 Info | I Manage suppressed messages | | |
| | | | - |
| | | | |
| 1 | | | |
| Log Message | | | |
| | Fe | am: Part: | / |
| | Figure 2 - Libero SoC Project Manager | | |



Checking Tool Profiles and IP Cores

- Click Project > Tool Profiles from the Libero SoC menu to open the Tool Profiles dialog box. Verify that the following tool profiles exist:
 - Synthesis: Synplify Pro ME contained in <Libero_v12.5_installation>\SynplifyPro\bin\synplify_pro.exe
 - Simulation: ModelSim ME Pro contained in
 <Libero v12.5 installation>\ModelsimPro\win32acoem\modelsim.exe
- 3. Click **OK** to close the Tool Profiles dialog box.
- 4. Open the Libero SoC IP catalog (View > Windows > Catalog). If the message New Cores are available appears at the bottom of the catalog, click Download them now to download the cores.

| Catalog | | | | ₽× |
|------------------|-------------------------|---------|------------------|-----------------|
| _ | | | Simulation Mode | () - |
| Name | | 🛆 🛛 Ver | rsion | |
| Arithmetic | | | | |
| Basic Blocks | | | | |
| Bus Interfaces | | | | |
| 🗄 Clock & Manag | ement | | | |
| ⊕ DSP | | | | |
| Fusion Peripher | als | | | |
| ⊕ I/O | | | | |
| Memory & Con | trollers | | | |
| Peripherals | | | | |
| PolarFire Featur | es | | | |
| Power Manager | nent | | | |
| Processors | | | | |
| SC/Tamper | | | | |
| Solutions-AXI | | | | |
| Solutions-FIL-H | SP-IP | | | |
| Solutions-Moto | rControl | | | |
| Solutions-Video |) | | | |
| Solutions-Wired | lComms | | | |
| Solutions-Wirel | ess | | | |
| Tamper | | | | |
| User Defined | | | | |
| <u> </u> | New cores are available | Do | wnload them now! | |

Figure 3 - IP Catalog indicating new cores are available

Creating the Libero SoC Project

Use the Tcl script contained in the Scripts folder to create the Libero SoC project.

- 5. Select **Project > Execute Script** from the Libero SoC menu. The Execute Script dialog box will open.
- - 7. Navigate to the C:\Microsemiprj\SMF2000_Cortex_M3_PWM_lab_sources\Scripts\ folder. Select *Project_creation.tcl* then click **Open**.
- 8. Click **Run** in the Execute Script dialog box to execute the script.



| Execute Script | ? | \times |
|--|---------|----------|
| Script file: C:/Microsemiprj/SMF2000_Cortex_M3_PWM_lab_sources/Scripts/Project_creat | tion.td | |
| Arguments: | | |
| Show script report | | |
| Help | Cano | el |



9. The script execution report will appear as shown below indicating successful execution of the script.

| Script Execution Report | ? | × |
|---|---------|----------|
| Messages Serrors (0) | Setting | s |
| Info: This version of Libero supports only the enhanced constraint flow. The SMF2000-M3_BaseDesign project was created. The Execute Script command succeeded. | | |
| Help | Close | <u>.</u> |

Figure 5 - Script execution report

10. Close the Script Execution Report.



Step 2 - Configuring the SmartFusion2 MSS with System Builder

In this step, you will configure the SmartFusion2 Microcontroller Subsystem (MSS) using System Builder.

1. Select the Libero SoC Design Flow tab. Expand Create Design and double-click System Builder.

| Design Flow | | | ₽× |
|---|------|-------|----|
| Please select a | • | 0 | ø. |
| Tool | | | |
| 🕀 🕨 🕨 Create Design | | | |
| 🖌 🕰 System Builder | | | |
| 🗛 Configure MSS | | | |
| Create SmartDesign | | | |
| Create HDL | | | |
| - Seate SmartDesign Testbench | | | |
| 🔄 📋 Create HDL Testbench | | | |
| Generate Memory Map | | | |
| Verify Pre-Synthesized Design | | | |
| 🔤 Simulate | | | |
| Constraints | | | |
| Implement Design | | | |
| Program and Debug Design | | | |
| Handoff Design for Production | | | |
| Handoff Design for Firmware Development | | | |
| Handoff Design for Debugging | | | |
| | | | |
| | | | |
| Design Flow Design Hierarchy Stimulus Hierarchy Cat | alog | Files | |

Figure 6 - Libero SoC Design Flow tab

2. Enter *SF2_MSS* when prompted for a name for your system.

| Enter a name for your sy | ? | × |
|--------------------------|----|------|
| Name: | | |
| SF2_MSS | | |
| Неір ОК | Ca | ncel |

Figure 7 - Entering a name for System Builder



3. System Builder will open with the "System Builder - Device Features" page visible as shown below.



Figure 8 – SmartFusion2 System Builder Device Features page

- 4. Enter the following on the "System Builder Device Features" page:
 - Memory
 - o MSS External Memory
 - MSS On-chip Flash Memory (eNVM)
 - Microcontroller Options
 - Watchdog Timer
 - o Peripheral DMA
 - o Real Time Counter

un-checked (default) checked

un-checked (default) un-checked (default) checked



Microsemi

а 🕵 Міскоснір company

Figure 9 - System Builder Device Features after selections

5. Click Next. The "System Builder – Memory" page will open.



Figure 10 - System Builder Memories page



- Select Data Storage under Available client types (highlighted in the figure above) then click Add to System. This will create a partition in the SmartFusion2 Embedded Non-volatile memory (eNVM) which will be used to store the Cortex-M3 application program.
- 7. Enter the following in the Add Data Storage Client dialog box:
 - Client name: PGM_store
 - Content:
 - \circ $\;$ Click the browse button to open the Import Memory File dialog box.
 - Navigate to the C:\Microsemiprj\SMF2000_Cortex_M3_PWM_lab_sources\hex_File folder
 - Select pwm_demo_eNVM.hex and "Use relative path" directory, then click Open

| 🕑 Import M | lemory File | 2 | | | | | | | | ? | | × |
|--------------------|---------------|----------------------|-------------------|----------|------------|----------|-----------|---------|---|---|--------------|-----|
| Look in: | C: Mic | crosemiprj\SMF200 | 0_Cortex_M3_PW | /M_lab_ | _sources\h | ex_File | • | 00 | 0 | Ø | :: | |
| S My Co | omputer | Name | | ∇ | Size | Туре | Date Modi | fied | | | | |
| 🚬 Jens | | pwm_demo | _eNVM.hex | | 20 KB | hex File | 30,11,28 | 3 12:20 | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| | | | | | | | | | | | | |
| File <u>n</u> ame: | pwm_dem | o_eNVM.hex | | | | | | | | Ç | <u>)</u> pen | |
| Files of type: | Intel-Hex | Files (*.hex *.ihx) | | | | | | | • | С | ancel | |
| C Use absolu | ute path (fil | e will not be copier | d if you move the | design) | | | | | | | | |
| 🚹 🖲 Use re | elative path | 1 | | | | | | | | | | |
| C Copy mem | ory file to p | project directory | | | | | | | | | | |
| | | | | | | | | | | | | _// |

Figure 11 - Selecting the hex file for the data storage client

8. The Add Data Storage Client dialog box will appear as shown in the figure below. Click **OK** to close the Add Data Storage Client dialog box.

| Content from file:Cortex_M3_PWM_lab_sources\hex_File\pwm_ | _demo_eNVM.hex |
|--|--------------------|
| Imported Memory file location : C:\Microsemiprj\SMF2000_Corte: \hex_File\pwm_demo_eNVM.hex | _M3_PWM_lab_source |
| Format: Intel-Hex | |
| Use absolute addressing | |
| C Content filled with 0s | |
| O No content (client is a placeholder and will not be programmed) | |
| Start address: 0x 0 📩 | |
| Size of word: 8 💌 Bits | |
| Number of words: 7584 Decimal | |
| | |
| Use as ROM 🚯 | |
| | |
| Use content for simulation | |
| | |
| | |
| | |
| | |
| | |

Figure 12 - Data Storage Client dialog box after importing the hex file



9. The client will be visible in the System Builder Memories page.

| System Builder - Memories | | | | | | — |) × |
|-------------------------------|----------------------|-------------------------|--------------------|-------------------|-------------------------|--------------------|------|
| Device Features Memories | Peripherals Confi | Clocks >> Micro | controller > > SE | CDED Ser | curity >> Interru | ipts >> Memory M | |
| Available client types | | | User clients | in eNVM | | | _ |
| Data Storage Serialization | Client Type (| Client Name DepthxWidth | Start Address(Hex) | Page Start Page B | nd Initialization Order | Lock Start Address | - |
| | 1 Data Storage P | GM_Storage 7584 x 8 | 0 | 0 59 | N/A | | |
| Add to System | | | | | | | |
| Available pages: 2032 | | | | | | | |
| Used pages: 60 | | | | | | | |
| Used space Free space | | | | | | | |
| | Optimize | | | | | Edit Delete | |
| Help | | | | | | Back | lext |

Figure 13 - System Builder Memories page after adding the data storage client

10. Click **Next**. The "System Builder – Peripherals" page will open. Here you can enable or disable MSS peripherals and add fabric peripherals. Configure peripherals by clicking the wrench symbol (*). The number of fabric peripherals used can be changed by editing the number in the Quantity column.

| Device Featur | res Memories | <u>Peripherals</u> <u>Clocks</u> | Microcontroller s and masters for the second s |) ∑ SE or each | ECDI sul | ED > > | Security >> Interrupts | . > > Memory M | |
|-----------------------|----------------------------------|---|--|-------------------|-------------|-----------|----------------------------------|----------------|----------|
| Direct Connection | Mode (FIC interfaces are Fab | exported out of System Builder) ric Slave Cores | | | | | Subsystems | | |
| Core | | Version | | | | MSS I | FIC_0 - MSS Master Subsyste | m | <u> </u> |
| CoreAHBLSRAM | 2.0.113 | | | | | drag a | nd drop here to add to subsystem | | |
| Corel2C | 7.0.102 | | | | | MSS F. | IC 0 - Fabric Master Subsyst | em | |
| CoreSPI | 3.0.156 | | | | | drag a | nd drop here to add to subsystem | | |
| CoreTimer | 1 1 101 | | | | | | MCC Paripharak | | = |
| CorellARTanh | 522 | | | ofigure | able | Name | PISS Peripherals | | |
| CorePWM | 4.1.106 | | | e la | | MM UART 0 | | | |
| Fabric AMBA Slave | 0.0.102 | | | ¢ [| - | MM_UART_1 | | | |
| | 1 | | | ۶ | • | MSS_I2C_0 | | | |
| | Eab | is Mastar Caros | 4 | ۶ | • | MSS_I2C_1 | | | |
| - | | | | ۶ | | MSS_SPI_0 | | | |
| Core | | Version | | ۶ | ~ | MSS_SPI_1 | | | |
| | er 0.0.102 | | | | | MSS_GPIO | | | |
| | | | | | | MSS_USB | | | |
| | | | | | | MSS_MAC | | | |
| | | | | L | | MSS_CAN | | | |
| | | | | | | | | | - |
| move a peripheral fro | m one subsystem to another, o | rag it from its present location and drop it onto the d | esired susbsystem. | | | | | | |
| cannot drag and dro | p onto MSS Peripherals, | | | | | | | | |

icros

Figure 14 - System Builder Peripherals page

- 11. Disable the MM_UART_1, MSS_I2C_0, MSS_I2C_1, MSS_SPI_0 and MSS_SPI_1 peripherals by clicking the box in the Enable column.
- 12. Click the wrench symbol (🌮) next to the MM_UART_0 peripheral to open the MM_UART_0 Configurator. Verify the "Connect to **IO**" configuration. Click **OK** to close the configurator.

| Configurator | — | | × |
|--------------------------------|--------|-----------|---|
| UART Configurato | r | | |
| Microsemi:SystemBuilder:SF2_MS | 55_UAR | T:0.0.101 | |
| Configuration | | | |
| Transmit-Receive | | | |
| Duplex Mode Full Duplex | • | | |
| Transmission Mode Asynchrone | ous 🔻 | | |
| Connect To IO | • | | |
| -Modem | | | |
| Use Modem | | | |
| Connect To 🛛 🖳 | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| Help | Ж | Cano | e |

Figure 15 - MM_UART_0 configurator



- 13. Enable the MSS_GPIO peripheral by clicking the box under the Enable column.
- 14. Click the wrench symbol (*) next to the MSS_GPIO peripheral to open the MSS GPIO Configurator. Configure the GPIOs as shown below.
 - Set/Reset Definition: accept default settings
 - Configure GPIO[31:0] per the table below:

| GPIO ID | Direction | Package Pin | Connectivity |
|-------------|-----------|-------------|--------------|
| GPIO[0:7] | Not Used | NA | NA |
| GPIO_8 | Input | NA | FABRIC_A |
| GPIO_[9:31] | Not Used | NA | NA |

Table 3 – SmartFusion2 GPIO configuration

| MSS | GPIO Configurator | | | | | ? × |
|-----|-----------------------|------------------------|------------------|------------------|--|-----------|
| Cor | nfiguration | | | | Connectivity Preview | ^ |
| Se | et/Reset Definition | n | | | | |
| G | PIO_31_24 Reset Sou | rce SYSREG (MSS_GPIO_3 | 1_24_SOFT_RESET) | Reset State 1 | GPIO_8 | |
| G | PIO_23_16 Reset Sou | rce SYSREG (MSS_GPIO_2 | 3_16_SOFT_RESET) | Reset State 1 💌 | GPIO_8_I | |
| G | PIO_15_8 Reset Source | ce SYSREG (MSS_GPIO_1 | 5_8_SOFT_RESET) | Reset State 1 💌 | | |
| G | PIO_7_0 Reset Source | SYSREG (MSS_GPIO_7 | 0_SOFT_RESET) | Reset State 1 💌 | | |
| | | | | | | |
| G | PIO Assignment | D ' '' | | Advanced Options | | |
| | GPIO ID | Direction | Package Pin | Connectivity | | |
| | | Not Used | | | W | |
| | | Not Used | | | MSS ¹² | |
| | GPIO_2 | Not Used 💌 | | ■ A_OI | 010 | |
| | GPIO_3 | Not Used 💌 | | IO_A | | |
| | GPIO_4 | Not Used 💌 | | 10_A 💌 | | |
| | GPIO_5 | Not Used 💌 | | IO_A 💌 | FPGA FADRIC | |
| | GPIO_6 | Not Used 💌 | | IO_A 💌 | Click on a signal row to see the preview | |
| | GPIO_7 | Not Used 💌 | | IO_A 💌 | | |
| | GPIO_8 | Input 💌 | | FABRIC_A | | |
| | GPIO_9 | Not Used 💌 | | | | |
| - | | | | ► ► | | - |
| 1 | | | | | | |
| He | lp | | | | | OK Cancel |

Figure 16 - SmartFusion2 GPIO Configuration

15. Click **OK** to close the MSS GPIO Configurator.



16. Drag an instance of CorePWM from the Fabric Slave Cores to the MSS FIC_0 - MSS Master Subsystem. This will add a soft PWM core in the FPGA fabric which will be connected to FIC_0.

| | Select the periphe | erals and maste | rs for eac | ch sub | system | | |
|--|--|----------------------------|------------|-------------|---------------|----------------------------|----------|
| Direct Connection | Mode (FIC interfaces are exported out of System Builder) Fabric Slave Cores | | | | | Subsystems | |
| Core | Version | _ | | | MSS FIC_0 | - MSS Master Subsystem | |
| Corel2C | 7.0.102 | | Configure | Quantit | у | Name | |
| CoreSPI | 3.0.156 | | ø | 1 | corepwm_0 | | |
| CoreGPIO | 3.0.120 | | | | MSS FIC_0 - | Fabric Master Subsystem | |
| CoreTimer | 1.1.101 | | | | drag and drop | o here to add to subsystem | |
| CoreUARTapb | 5.2.2 | | | | ſ | ISS Peripherals | |
| CorePWM | 4.1.106 | | Configure | Enable | | Name | _ |
| Fabric AMBA Slav | e 0.0.102 | • | ø | | MM_UART_0 | | |
| | | | | MM_UART_1 | | | |
| Core | Version | | | | MSS_I2C_0 | | |
| Fabric AMBA Mas | ter 0.0.102 | | | | MSS_I2C_1 | | |
| | | | | | MSS_SPI_0 | | |
| | | | | | MSS_SPI_1 | | |
| | | | ø | > | MSS_GPIO | | |
| | | | | | MSS_USB | | |
| | | | | | MSS_MAC | | |
| move a peripheral fr i cannot drag and dra sters are in bold an Help v | om one subsystem to another, drag it from its present location and drop it onl op onto MSS Peripherals. Id blue. Icel | to the desired susbsystem. | | 1 | | Back | Next |

- 17. Click the wrench symbol (*) next to corepwm_0 in the MSS FIC_0 MSS Master Subsystem to open the CorePWM configurator. Enter the following in the Configuring corepwm_0 dialog box.
 - Global Configuration:
 - Configuration Mode: 0 PWM Only Mode (default)
 - Number of PWM Channels: 8
 - APB Data Bus Width / Resolution: 16
 - Global PWM Mode Configuration:
 - Fixed Prescale: un-checked
 - Fixed Period: unchecked (default)
 - Channel 1 8 Configuration: accept the default settings
- 18. Click **OK** to close the CorePWM configuration dialog box.



| Configurator | | | | - | |
|---|-------------------------|----------------------------|---|---|----|
| CorePWM Configurator | | | | | |
| icrosemi:DirectCore:corepwm:4.1.106 | | | | | |
| Configuration | | | | | |
| Global Configuration: | | | | | ור |
| Configuration Mode: 0 - PWM | Only Mode | • | | | |
| Number of PWM Channels: 8 | | • | | | |
| APB Data Bus Width / Resolution: 16 | | _ | | | |
| Global PWM Mode Configuration: | | | | | |
| Fixed Prescale: Fixed Value: 0 | | _ | | | |
| Fixed Period: Fixed Value: 1 | | _ | | | |
| Channel 1 Configuration: | | | | | |
| Low Ripple DAC mode: | | Shadow Update Register: | | | |
| Fixed PWM PosEdge: | $\overline{\mathbf{v}}$ | Fixed PWM PosEdge Value: 0 | | | |
| Fixed PWM NegEdge / DAC LevelOut: | | Fixed PWM NegEdge Value: 0 | _ | | |
| Channel 1 PWM stretch level (HIGH when sele | cted) 🗆 | | | | |
| Channel 2 Configuration: | | | | | |
| Low Ripple DAC mode: | | Shadow Update Register: | | | |
| Fixed PWM PosEdge: | $\overline{\mathbf{v}}$ | Fixed PWM PosEdge Value: 0 | | | |
| Fixed PWM NegEdge / DAC LevelOut: | | Fixed PWM NegEdge Value: 0 | _ | | |
| Channel2 PWM stretch level (HIGH when sele | cted) 🗆 | | | | |
| Channel 3 Configuration: | | | | | |
| Low Ripple DAC mode: | | Shadow Update Register: | | | |
| Fixed PWM PosEdge: | $\overline{\mathbf{v}}$ | Fixed PWM PosEdge Value: 0 | | | |
| Fixed PWM NegEdge / DAC LevelOut: | | Fixed PWM NegEdge Value: 0 | _ | | |
| Channel3 PWM stretch level (HIGH when sele | cted) 🗆 | | | | |
| | | Г | | | 11 |

Figure 18 - corepwm_0 configuration

Note: additional information about CorePWM and the configuration parameters is available in the CorePWM handbook. The CorePWM handbook can be accessed from the CorePWM Configurator by clicking Help.



- 19. Click **Next**. The "System Builder Clocks" page will open. Use this page to specify the clock source and clock frequencies used in the design. Enter the following on the Clock tab:
 - System Clock: Select On-chip 25/50 MHz RC Oscillator from the pull-down menu
 - M3_CLK: 100 MHz (default)
 - APB_0 CLK: M3_CLK/1 (100 MHz) (default)
 - APB_1 CLK: M3_CLK/1 (100 MHz) (default)
 - FIC_0_CLK: M3_CLK/2 (50 MHz)

These frequencies were chosen to maximize the speed of the Cortex-M3 microcontroller.

| at I saturas I | | | | co | Jure Goek requirements | |
|--|---------------------|------------------|------|---------|------------------------|----------------|
| stem Clock 50.0 On-chip 25/50 MHz Ri | MHz C Oscillator | | - | | | Cortex-M3 |
| Cortex-M3 and MSS M | 1ain Clock | | | | | |
| M3_CLK | = | 100.00 | MHz | 100.000 | MSS_CCC HPDI | AHB Bus Matrix |
| IDDR Clocks | | | | | M3_CLK | MMUART |
| MDDR_CLK | = M3_CLK * | 1 | ~ | | | SPI_0 |
| DDR/SMC_FIC_CLK | = MDDR_CLK | / 1 | Ŧ | | APB_0_CLK | |
| ISS APB_0/1 Clocks | | · | | | APB_1_CLK | DMA |
| APB_0_CLK | = M3_CLK / | 1 | • | 100.000 | | WD |
| APB_1_CLK | = M3_CLK / | 1 | • | 100.000 | | |
| abric Interface Clock | s | | | | MSS 🕇 | FIC_0 |
| FIC_0_CLK | = M3_CLK / | 2 | • | 50.000 | | |
| | | AHBLite Bypass | Mode | | | FIC_0 |
| FIC_1_CLK | = M3_CLK / | 1 | Ŧ | | FIC_0_CLK | Subsystēm |
| | | ☐ AHBLite Bypass | Mode | | | |
| abric DDR Clocks | | | | | | |
| FDDR_CLK | = | 100 | MHz | | OSC | |
| EDDR SUBSYSTEM C | TK = EDDR CIK | / 1 | - | | Fabric | |

Figure 19 – Configuring the MSS clocks

20. Click **Next**. The "System Builder - Microcontroller" page will open. This page has multiple tabs, which allow configuring the Cortex-M3 microcontroller, the Cache Controller and the AHB Bus Matrix.

| System Builder - Microcontroller | |
|--|---|
| Davies Fastures Memories Deripherals | Clarke Microsophallar CECDED Coquity Interrupts Mamon Man |
| <u> Device reatures</u> / <u>Memories</u> / <u>Peripherais</u> | Clocks / Microconditioner / Security / Interrupts / Methody Map / |
| | Configure Microcontroller options |
| Real Time Counter Cortex-M3 Cache Controller AHB Bus Matrix | |
| | Clock Source |
| | Source External 32 KHz RTC Crystal Oscillator |
| | WakeUp Interrupt |
| | Enable WakeUp Interrupt to Cortex-M3 |
| | Enable WakeUp Interrupt to FPGA Fabric |
| | |
| | Expose RTC_MATCH port to FPGA Fabric |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| | |
| Help Cancel | Back Next |

Figure 20 - System Builder Microcontroller options page

- 21. Select the Real Time Counter tab in the System Builder Microcontroller pane. This is where options for the RTC are set.
- 22. Select the On-chip 1 MHz RC Oscillator as the RTC clock source from the pull-down menu in the Clock Source field on the Real Time Counter tab. This option matches the RTC sample projects available in the Firmware Catalog.
- 23. Accept the default settings for the Wakeup Interrupt and the RTC_MATCH signal.

| TIONZ ARM CORTEX-M3 | Lab Guiae | а 🏠 Міскоснір сотра |
|--|--|--------------------------|
| m Builder - Microcontroller | | |
| Device Features > Memories > Perip | nerals > Clocks > Microcontroller > SECDED > Security > Ir | nterrupts > Memory Map > |
| | Configure Microcontroller options | |
| I Time Counter Cortex-M3 Cache Controller AHB Bu | s Matrix | |
| | Clock Source | |
| | Source On-chip 1 MHz RC Oscillator | |
| | WakeUp Interrupt Enable WakeUp Interrupt to Cortex-M3 | |
| | Enable WakeUp Interrupt to FPGA Fabric | |
| | RTC_MATCH Expose RTC_MATCH port to FPGA Fabric | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |
| | | |

Figure 21 - System Builder Microcontroller options page after selecting the RTC clock source

- 24. Although we won't be making any other changes, take a moment to become familiar with the contents of the other tabs (Cortex-M3, Cache Controller, AHB Bus Matrix).
- 25. Click **Next** to accept the default settings on the Microcontroller page.



26. The "System Builder – SECDED" (Single Error Correct / Double Error Detect) page will open. Use this page to enable SECDED for various memory blocks within the SmartFusion2 MSS. Although we won't be using SECDED for this design, take a moment to become familiar with the contents of this page.

| System Builder - SECDED | |
|---|-------------------|
| Device Features Memories Peripherals Clocks Microcontroller SECDED Security Interrupt | s >> Memory Map > |
| Configure Single Error Correct / Double Error Detect (SECDED) options | |
| SECDED | |
| | • |
| EDAC Errors | |
| Expose EDAC_ERROR bus | |
| eSRAM_0 | |
| Enable EDAC Interrupt(s) None | |
| eSRAM_1 | |
| Enable EDAC Interrupt(s) None | |
| Ethernet TX RAM | E |
| Enable EDAC Enable EDAC Interrupt(s) None | |
| Ethernet RX RAM | |
| Enable EDAC Enable EDAC Interrupt(s) None | |
| USB | |
| | |
| CAN Enable EDAC Enable EDAC Interrupt(s) None | |
| MDDR | |
| Enable MODE ECC Teters at 177 | - |
| | |
| Help Cancel | Back Next |

Figure 22 - System Builder SECDED options page

- 27. Accept the default settings on the page and click Next.
- 28. The "System Builder Security" page will open. This page is used to set the Master to Slave Read/Write access for devices that support Data Security features. The SmartFusion2 devices on the SMF2000 kit does not support these security features, so these options will be grayed out.
- 29. Click **Next** to accept the default settings.



30. The "System Builder – Interrupts" page will open. This page displays interrupt connections generated from attached fabric peripherals. The CorePWM TACHINT interrupt will be visible as shown in the figure below.

| 🕑 System Builder - Interrupts | | | — [| ⊐ × |
|---------------------------------|-----------------|---|--------|------|
| > Device Features > Me | mories | als Clocks Microcontroller SECDED Security Interrupts | Memory | Map |
| | Interrupt conne | ections generated from attached peripherals and the processor | | |
| Processor Interrupt | Instance Name | Trigger Signals | | Lock |
| SF2_MSS_sb_MSS_0:MSS_INT_F2M[0] | corepwm_0_0 | TACHINT | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| | | | | |
| Help Cancel | | [| Back | Next |
| | | Figure 23 - CorePWM TACHINT interrupt | | |



31. Click Next. The "System Builder – Memory Map" page will open. This page displays the addresses of fabric

peripherals. The address of CorePWM will be visible. Up to six different memory regions can be assigned to each FIC in the MSS memory map. In this design, two memory regions (0x30000000 and 0x50000000) are assigned to FIC_0 for the interface to the FPGA fabric and CorePWM.

| 📀 System Builder - Memory Map | | - 0 | × |
|---|---|-------------|---|
| Device Features Memories Peripherals | Clocks > Microcontroller > SECDED > Security > Interrupts > | Memory Map | |
| Memo | ory Map for peripherals attached to the processor | | |
| Select Bus to View or Assign Peripheral(s) | Assign peripherals to addresses on bus: | | |
| CoreAPB3_0 (MSS FIC_0 - MSS Master Subsystem) | Address Peripheral 0x50000000 corepwm_0_0APBslave | | |
| Help Cancel | | Back Finish | |

Figure 24 - Fabric memory map

32. Click Finish.

33. Confirm that the message "SF2_MSS_sb'" was generated successfully" appears in the Libero SoC Message tab.

| Me | Aessage | | | | | | | |
|----|--|---|-----------------|----------------|--|--|--|--|
| | 🗏 Messages 😵 Errors 🗼 Warnings 🌒 Info 🔳 Manage suppressed messages | | | | | | | |
| Γ | Message ID Source Location | | Source Location | Log Location | | | | |
| | Please refer for details about SF2_MSS_sb | - | - | SF2 MSS sb Log | | | | |
| | SF2_MSS_sb' was generated successfully | - | - | - | | | | |
| | | | | | | | | |
| | Log Message | | | | | | | |

Figure 25 - Message tab after successfully generating the MSS design

34. A component named *SF2_MSS_*sb_0 will be visible in the SmartDesign canvas as shown below. If necessary, click the SF2_MSS tab to display the SmartDesign canvas.





35. Connect the FAB_RESET_N port to VCC by selecting the port, right-clicking and selecting **Tie High**.



Figure 27 – Connecting FAB_RESET_N port to VCC

- 36. If necessary, click the + sign next to corepwm_0_0_PINS to expand the pin group. Promote the PWM[8:1] port to the top level by selecting the port, right-clicking and selecting **Promote to Top Level**. These outputs will drive LEDs on the target board. Connect the TACHIN[1] port to GND by selecting the port, right-clicking and selecting **Tie Low**.
- 37. Mark the POWER_ON_RESET_N and MSS_READY output ports unused by selecting the port, right-clicking and selecting **Mark Unused**.
- 38. If necessary, click the + sign next to FAB_CCC_PINS to expand the pin group. Mark the FAB_CCC_GL0 and FAB_CCC_LOCK pins unused. These pins are not used in the design.
- 39. If necessary, click the + sign next to GPIO_FABRIC to expand the pin group. Promote the GPIO_8_F2M port to the top level by selecting the port, right-clicking and selecting **Promote to Top Level**. The GPIO_8_F2M port will connect to a switch on the SMF2000 board.
- 40. If necessary, click the + sign next to the INIT_PINS output port to expand the group. Mark the INIT_DONE output port unused.
- 41. After making the pin connections the *SF2_MSS*_sb_0 component will look like the figure below.



Figure 28 – SF2_MSS_sb_0 component after making pin connections



42. Execute a Design Rules Check by clicking the Checkmark icon.



Figure 29– Execute Design Rules Check

- 43. Verify the correctness of the SmartDesign Component:
 - Check Messages window for "SmartDesign 'SF2_MSS' design rules check succeeded"
- 44. Generate the design by clicking **SmartDesign > Generate Component** or by clicking the Generate Component icon

on the SmartDesign toolbar (🧐).

45. The message "SF2_MSS' was generated successfully" will appear in the Libero SoC Message window indicating the design was generated without any errors.

| Me | essage | | | 8 × | | | | |
|--|---------------------------------------|-------------------|--------------|----------|--|--|--|--|
| 🗐 Messages 😵 Errors 🗼 Warnings 🌒 Info 🗏 Manage suppressed messages | | | | | | | | |
| | Message Message 1 | D Source Location | Log Location | _ | | | | |
| | | - | - | | | | | |
| | SF2_MSS' was generated successfully - | - | • · | • | | | | |
| | | | | | | | | |
| L | Log Message | | | | | | | |

Figure 30 - Libero SoC Message window after generating the design

40. Close the design (File > Close SF2_MSS).



Step 3 – Constraining the Design

Importing an IO Constraint file

There are multiple ways to make I/O Assignments. In this lab, we will use the I/O Physical Design Constraint (PDC) file that is provided in the lab source files.

1. Expand Constraints in the Libero Design Flow window and double-click Manage Constraints.



Figure 31 - Opening the Constraint Editor

2. The Enhanced constraint manager will open.

| Reports | |
|--|--------------|
| D Attributes Timing Floor Planner Netlist Attributes | |
| New 🔻 Import Link Edit 🚩 Check Help |) |
| Place and Route | |
| | |
| | |
| | |
| | |
| | |
| | |
| I/O Settings | |
| Reserve Pins for Device Migration | |
| Calact the devices you are targetting for migration. Dire not handed on these devices will be received in the device selected for this project. | |
| Select the devices you are targetting for hingration. Fins hot bonded on these devices will be reserved in the device selected for this project. | |
| | |
| Tarent Davisors | |
| M2S005S | |
| | |
| General | |
| Reserve Pins for Probes | |
| | |
| | |





 Click Import on the I/O Attributes tab to open the Import Files dialog box. Navigate to the C:\Microsemiprj\SMF2000_Cortex_M3_PWM_lab_sources\constraints folder. Select *io_constraints.pdc* then click Open.

| 🕑 Import Files | | × |
|--|--------------------------|---|
| $\leftarrow \rightarrow \checkmark \uparrow$ — \leftarrow Arro | owTraining > constraints | ✓ [™] constraints" durchsuchen |
| Organisieren 👻 Neuer | Ordner | 1== - 11 ? |
| og Timing & Synch ^ | Name | Änderungsdatum Typ |
| 👧 Übergabefotos | io_constraints | 11/27/2018 11:17 PDC-Date |
| or USB | | |
| 💻 Dieser PC 🛛 🗸 | < | > |
| Datei <u>n</u> | ame: io_constraints | ✓ I/O Constraint Files (*.pdc) ✓ Ö<u>f</u>fnen Abbrechen |
| | | Abbicenen |

Figure 33 - Importing the I/O constraint file

4. The file will be visible on the I/O Attributes tab of the Constraint Manager. Double-click io_constraints.pdc to open the file in the Libero SoC text editor. Scroll in the file to become familiar with the syntax. The constraint set_io sets the pin number and I/O specific attributes. The # symbol is a comment. Note the internal pull-up setting for the GPIO_8_F2M signal pin.

```
# USER_BTN
set_io GPIO_8_F2M -pinname B19 -fixed yes -iostd LVCMOS33 -RES_PULL Up
```

- 5. Close the editor (File > Close io_constraints.pdc).
- 6. Check the box under Place and Route for constraint\io\io_constraints.pdc in the I/O Attributes tab to use the PDC constraint file for layout.
- 7. Click **Save** to save all changes in Constraint Manager, select **YES** in Warning message about flow invalidation.



Figure 34 - Selecting and saving the I/O constraint file for layout



Generating Timing Constraints

In this step, you derive timing constraints for the design. Libero can generate timing constraints for known blocks (such as the RC oscillators and the PLLs) automatically.

7. Select the Timing tab in the Constraint manager window. Double-click **Derive Constraints**.

| Reports & X StartPage & X Constra | aint Manager 🛛 🗗 🗙 | | | | ; |
|---|------------------------|-----------------|---------------------|---------------------|--------|
| I/O Attributes Timing Floor Planner Netlist Attribu | ites | | | | |
| New Import Link | Fdit with Constraint F | ditor 👻 Check 🔻 | Derive Constraints | Constraint Coverage | Help 🔶 |
| | Synthesis | Place and Route | Timing Verification | | |
| | Synthesis | Place and Route | Timing venication | 1 | |



8. Click on **Yes** in the Message window to automatically associate the derived constraints SDC file to the 'Synthesis', 'Place and Route' and 'Timing Verification'.

| Message | × |
|--|--|
| To automatically associate the derived constraint SDC file to the 'Sy \underline{Y} es | nthesis', 'Place and Route' and 'Timing Verification' tools click 'Yes' else click 'No'. |
| Figure 36 - N | Aessage Window |

9. A constraint file named SF2_MSS_derived_constraints.sdc will be visible. Double-click on the file name to open the file in the Libero text editor to become familiar with the content.

| | Reports 🗗 🗙 | StartPage 🗗 🗙 | Constraint | Manager 🗗 🗙 | | | = | , |
|-----|------------------|----------------------|--------------------|-------------------|-----------------|--|--------|---|
| I/(| O Attributes Tim | ning Floor Planner N | Vetlist Attributes |] | | | | |
| | New | Import | Link Edit | with Constraint E | ditor 👻 Check 👻 | Derive Constraints Constraint Coverage | Help 🗲 | |
| | | | Sy | nthesis | Place and Route | Timing Verification | | |
| Ę | constraint\SF2_ | MSS_derived_cons | traints.sdc 🗹 | | | | | |

Figure 37 - Derived Timing Constraints



Step 4 – Synthesis and Layout

In this step, you will use the push-button flow to synthesize the design with Synplify Pro, run layout and generate the programming file

1. Double-click the Generate Bitstream in the Design Flow window to synthesize the design, run layout using the I/O constraints that were created and generate the programming file.

| Design Flow A | × |
|---|-----|
| Fabric_Top 🖸 🖸 🖗 | F |
| Tool | - |
| 🕀 🕨 Create Design | |
| 🖻 🕨 Constraints | |
| Manage Constraints | |
| 🖻 🕨 Implement Design | |
| 🗣 🚱 Netlist Viewer | |
| Synthesize | |
| Verify Post-Synthesis Implementation | |
| Generate Simulation File | |
| 🔤 Simulate | |
| Configure Flash*Freeze | |
| Configure Register Lock Bits | |
| Place and Route | |
| Verify Post Layout Implementation | |
| Generate Back Annotated Files | |
| Simulate | |
| 🗠 Verify Timing | |
| 🗠 Open SmartTime | |
| 🖳 🔯 Verify Power | |
| 🖻 🕨 IO Analyzer | |
| SSN Analyzer | |
| 🖻 🕨 Program and Debug Design | |
| Generate FPGA Array Data | |
| Update eNVM Memory Content | |
| Configure Hardware | |
| Configure Programming Options | |
| Sconfigure Security | |
| Program Design | |
| Generate Bitstream | |
| Run PROGRAM Action | |
| 🗈 🕨 Debug Design 🦳 | |
| Handoff Design for Production | |
| Handoff Design for Firmware Development | - 1 |
| 🗄 🕨 Handoff Decian for Debugging | |
| Design Flow Design Hierarchy Stimulus Hierarchy Catalog Files | _ |

Figure 38 - Generate Bitstream



The design implementation tools will run in batch mode. Successful completion of a design step will be indicated by a green check mark next to the Implement Design item in the Design Flow window.

| abric_ | Тор | | | - | 0 | Ŷ |
|--------|------|------------------|---|---|---|---|
| | Tool | | | | | |
| | ⊳ | × | Create Design | | | _ |
| | ٨ | Þ | Constraints | | | |
| | | | Manage Constraints | | | |
| 1 | ۵ | × | Implement Design | | | |
| - | | | P Netlist Viewer | | | |
| ~ | | | Synthesize | | | |
| | | \triangleright | Verify Post-Synthesis Implementation | | | |
| | | | Configure Flash*Freeze | | | |
| | | | Configure Register Lock Bits | | | |
| ∕ _ | | | Place and Route | | | |
| | | 4 | Verify Post Layout Implementation | | | |
| | | | Generate Back Annotated Files | | | |
| | | | Simulate | | | |
| | | | 🖄 Verify Timing | | | |
| | | | 💩 Open SmartTime | | | |
| | | | 💫 Verify Power | | | |
| | | | IO Analyzer | | | |
| | | | 💓 SSN Analyzer | | | |
| | 4 | ۲. | Program and Debug Design | | | _ |
| | | | • Generate FPGA Array Data | | | |
| | | | 🐼 Update eNVM Memory Content | | | |
| | | \triangleright | Configure Hardware | | | |
| | | | Configure Programming Options | | | |
| | | | Configure Security | | | |
| | | 4 | Program Design web | | | |
| | | | Generate Bitstream | | | |

Figure 39 - Successful completion of design implementation



2. The Reports tab will display reports for the tools used to implement the design.

| Reports 🗗 🗙 | StartPage 🗗 🗙 📔 | | | | |
|---|-----------------|--|--|--|--|
| Reports T × | StartPage P × | | | | |
| SF2_MSS_layout_combinational_loops.xml SF2_MSS_place_and_route_constraint_coverage.xml SF2_MSS_pinrpt_name.rpt SF2_MSS_bankrpt.rpt SF2_MSS_bankrpt.rpt SF2_MSS_layout_log.log Generate FPGA Array Data SF2_MSS_init_config.xml Generate Bitstream SF2_MSS_generateBitstream.log | | | | | |

Figure 40 - Reports tab after implementing the design

4.Connect the SMF2000 board using a Micro-USB cable to the PC USB port (board power is provided from PC USB port). The Window's Device Manager should display the following entries (port number may differ depending on your PC):



Figure 41 - Device Manager



5. Run program action



Step 5 - Running the Application

COM

The next step is to observe the operation of the Cortex-M3 program in the design.

Determining the COM port setting

This design requires a terminal emulator. In order to configure the terminal emulator, the COM port assignment must be determined.

- 1. Open the Windows Device Manager and expand the Ports (COM & LPT) section.
 - One port will be listed as "FlashPro 5 Port". Record the COM port number below.

USB Serial Port:

| 🛃 Device Manager — | × |
|---|------|
| <u>File</u> <u>Action</u> <u>V</u> iew <u>H</u> elp | |
| | |
| > 🎽 Firmware | ^ |
| > 🛺 Human Interface Devices | |
| > 🧝 IDE ATA/ATAPI controllers | |
| > 🚠 Imaging devices | |
| > 🔤 Keyboards | - 10 |
| > 🥅 Memory technology devices | |
| Mice and other pointing devices | |
| > 🛄 Monitors | |
| > 🖵 Network adapters | |
| V 💭 Ports (COM & LPT) | |
| FlashPro5 Port (COM5) | |
| > 🚍 Print queues | |
| > 🚍 Printers | |
| Decessory | ~ |
| | |
| | |

Figure 43 - Windows Device Manager showing COM port

- 2. Open a terminal emulator program such as TeraTerm or PuTTY and configure a Serial terminal as follows:
 - Port: Select the COM port number recorded above.
 - Baud rate: 115200
 - Data Bits: 8
 - Parity: None
 - Stop Bits: 1
 - Flow Control: None
- 3. Press and release User Button (the one close to the LEDs) at SMF2000 board. The string "Button-Press" will appear in the Terminal Emulator.
- 4. Press and release the reset button. The string "Hi! I am SmartFusion2 :)" will appear in the Terminal.
- 5. The LEDs should appear as follows:
 - LED D9 to LED D2 gradually get dimmer and brighter. The PWM core in the FPGA fabric drives these LEDs.



Step 6 – Generating Sample Projects and Exporting the Firmware Configuration Files

In this step, you will generate sample projects and export the firmware configuration files for the design.

Generating sample projects

1. Expand Handoff Design for Firmware Development on the Design Flow tab. Select Configure Firmware Cores, then right-click and select **Open Interactively**.



Figure 44 - Configuring Firmware cores for the design

2. The DESIGN_FIRMWARE tab will open.

| R | Reports Ø × StartPage Ø × Constraint Manager Ø × SD DESIGN_FIRMWARE Ø × | | | | | | | | | |
|---|---|-----|---|---|-----------|------------------------------|--|--|--|--|
| 0 | | | | | | | | | | |
| | Generate | | Instance Name | Core Type | Version | Compatible Hardware Instance | | | | |
| 1 | v | - | CorePWM_Driver_0 | CorePWM_Driver | 2.4.100 - | SF2_MSS_sb:corepwm_0_0 | | | | |
| 2 | v | Ø 🖣 | SmartFusion2_CMSIS_0 | SmartFusion2_CMSIS | 2.3.105 - | SF2_MSS_sb_MSS | | | | |
| 3 | V | - | SmartFusion2_MSS_GPIO_Driver_0 | SmartFusion2_MSS_GPIO_Driver | 2.1.102 - | SF2_MSS_sb_MSS:GPIO | | | | |
| 4 | Y | - 4 | SmartFusion2_MSS_HPDMA_Driver_0 | SmartFusion2_MSS_HPDMA_Driver | 2.2.100 - | SF2_MSS_sb_MSS | | | | |
| 5 | • | - | SmartFusion2_MSS_MMUART_Driver_0 | SmartFusion2_MSS_MMUART_Driver | 2.1.100 - | SF2_MSS_sb_MSS:MMUART_0 | | | | |
| 6 | • | - 4 | SmartFusion2_MSS_NVM_Driver_0 | SmartFusion2_MSS_NVM_Driver | 2.5.100 - | SF2_MSS_sb_MSS | | | | |
| 7 | ~ | - | SmartFusion2_MSS_RTC_Driver_0 | SmartFusion2_MSS_RTC_Driver | 2.2.100 - | SF2_MSS_sb_MSS:RTC | | | | |
| 8 | ~ | - | SmartFusion2_MSS_System_Services_Driver_0 | SmartFusion2_MSS_System_Services_Driver | 2.9.100 • | SF2_MSS_sb_MSS | | | | |
| 9 | ~ | - | SmartFusion2_MSS_Timer_Driver_0 | SmartFusion2_MSS_Timer_Driver | 2.2.100 - | SF2_MSS_sb_MSS | | | | |

Figure 45 - DESIGN FIRMWARE tab

- 3. Confirm that none of the drivers appears in *italics*. If any drivers appear in *italics*, click the check box in the Generate column for the missing core. Click **Yes** when prompted about downloading the core.
- 4. Click Yes in the Download Required dialog box to download any firmware cores that are missing from the IP vault.





Figure 46 - Downloading missing firmware cores

5. Use the pull-down menu in the SmartFusion2_CMSIS_0 row to select version 2.3.105 if it is not selected.

| R | Reports 🗗 × StartPage 🗗 × Constraint Manager 🗗 × 🕺 DESIGN_FIRMWARE 🗗 × | | | | | | | | | | |
|---|--|----------|---|---|-----------|------------------------------|--|--|--|--|--|
| C | 0 | | | | | | | | | | |
| | Generate | | Instance Name | Core Type | Version | Compatible Hardware Instance | | | | | |
| 1 | ~ | • | CorePWM_Driver_0 | CorePWM_Driver | 2.4.100 - | SF2_MSS_sb:corepwm_0_0 | | | | | |
| 2 | ~ | Ø 🖣 | SmartFusion2_CMSIS_0 | SmartFusion2_CMSIS | 2.3.105 - | SF2_MSS_sb_MSS | | | | | |
| 3 | ~ | - | SmartFusion2_MSS_GPIO_Driver_0 | SmartFusion2_MSS_GPIO_Driver | 2.1.102 - | SF2_MSS_sb_MSS:GPIO | | | | | |
| 4 | ~ | • | SmartFusion2_MSS_HPDMA_Driver_0 | SmartFusion2_MSS_HPDMA_Driver | 2.2.100 - | SF2_MSS_sb_MSS | | | | | |
| 5 | ~ | • | SmartFusion2_MSS_MMUART_Driver_0 | SmartFusion2_MSS_MMUART_Driver | 2.1.100 - | SF2_MSS_sb_MSS:MMUART_0 | | | | | |
| 6 | ~ | - | SmartFusion2_MSS_NVM_Driver_0 | SmartFusion2_MSS_NVM_Driver | 2.5.100 | SF2_MSS_sb_MSS | | | | | |
| 7 | ~ | a | SmartFusion2_MSS_RTC_Driver_0 | SmartFusion2_MSS_RTC_Driver | 2.2.100 | SF2_MSS_sb_MSS:RTC | | | | | |
| 8 | ~ | a | SmartFusion2_MSS_System_Services_Driver_0 | SmartFusion2_MSS_System_Services_Driver | 2.9.100 - | SF2_MSS_sb_MSS | | | | | |
| 9 | ~ | - | SmartFusion2_MSS_Timer_Driver_0 | SmartFusion2_MSS_Timer_Driver | 2.2.100 - | SF2_MSS_sb_MSS | | | | | |

Figure 47 - Selecting SmartFusion2_CMSIS version 2.3.105

 Create the CorePWM sample project by selecting CorePWM_Driver_0 on the DESIGN FIRMWARE tab, then right clicking and selecting Generate Sample Project > Cortex-M3 > SoftConsole v4.0 > PWM slow blink. Note that SoftConsole v4.0 projects can be opened in SoftConsole v6.4.

| Reports & X StartPage & X SD DESIGN_FIRMWARE & X | | | | | | | | | |
|--|---|------------------|-------------------------------|---------------------------|---|---|------------------|---------|--|
| 0 | | | | | | | | | |
| 3enerat€ | | e | Instance Name | | | Core Type | | | Compatible Hardware Instance |
| 1 | | CorePWM_Driver_0 | | | CorePWM_Dri | CorePWM_Driver | | | SF2_MSS_sys_sb:corepwm_0_0 |
| 2 | ~ | Ø 🖣 | SmartFusion2_CMSIS_0 | Disable Generation | SmartFusion2_CMSIS | | | 2.3.105 | SF2_MSS_sys_sb_MSS |
| 3 | 7 | - | SmartFusion2_MSS_GPIC | Show Details | SmartFusion2 | _MSS_GPIO_Driver | | 2.1.102 | SF2_MSS_sys_sb_MSS:GPI0 |
| 4 | 7 | . | SmartFusion2_MSS_HPDI | Open Documentation | SmartFusion2 | _MSS_HPDMA_Driver | | 2.2.100 | SF2_MSS_sys_sb_MSS |
| 5 | 7 | • | SmartFusion2_MSS_MML | Generate Sample Project 🔸 | Cortex-M1 • | MSS_MMUART_Driver | | 2.1.100 | ▼ SF2_MSS_sys_sb_MSS:MMUART_0 |
| 6 | ~ | - | SmartFusion2_MSS_MMU | ART_Driver_1 | er_1 Cortex-M3 SoftConsole v4.0 PWM edge co | | PWM edge control | | SF2_MSS_sys_sb_MSS:MMUART_1 |
| 7 | ~ | - | SmartFusion2_MSS_NVM_Driver_0 | | RISC-V → | MSS_NVM_Driver | PWM slow blink | | SF2_MSS_sys_sb_MSS |
| 8 | ~ | • | SmartFusion2_MSS_RTC_ | Driver_0 | SmartFusion2 | SmartFusion2_MSS_RTC_Driver PWM tachometer m | | sure | SF2_MSS_sys_sb_MSS:RTC |
| 9 | ~ | • | SmartFusion2_MSS_Syste | m_Services_Driver_0 | SmartFusion2 | SmartFusion2_MSS_System_Services_Dr PWM waveform al | | nent | ▼ SF2_MSS_sys_sb_MSS |
| 10 | ~ | - | SmartFusion2_MSS_Time | r_Driver_0 | SmartFusion2 | SmartFusion2_MSS_Timer_Driver | | | SF2_MSS_sys_sb_MSS |

Figure 48 - Generating the CorePWM sample project

- 7. Confirm the following settings in the Generate Sample Options dialog box then click OK:
 - Folder: C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole
 - Show generation report: checked

| Generate Sample Options | ? | × | | | | | |
|---|----------------|---|--|--|--|--|--|
| Samples folder: C:\Microsemiprj\SMF2000-M3_BaseDesi | gn\SoftConsole | | | | | | |
| Files will be generated in: C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole\SF2_GNU_SC4_pwm_slow_blink | | | | | | | |
| Show generation report | | | | | | | |
| Help | OK Cancel | | | | | | |

Figure 49 – CorePWM slow blink sample project options



8. The Report dialog box will list all the files generated and the location.

| Report | ? | × |
|--|------|------|
| Save | 🖨 Pi | rint |
| Files generated in 'C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole': | | |
| SF2_GNU_SC4_pwm_slow_blinkcproject SF2_GNU_SC4_pwm_slow_blinkCMSIS\my_reg_jo.h SF2_GNU_SC4_pwm_slow_blink\CMSIS\mss_assert.h SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc\/debug-in-microsemi-smartfusion2-envm.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc.\/debug-in-microsemi-smartfusion2-envm.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc.\/debug-in-microsemi-smartfusion2-envm.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc.\/debug-in-microsemi-smartfusion2-envm.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc.\/debug-in-microsemi-smartfusion2-envm.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc.\/debug-in-microsemi-smartfusion2-envm.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc.\/production-smartfusion2-excute-in-place.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc.\/production-smartfusion2-relocate-to-external-ram.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\startup_gcc.\/production-smartfusion2-relocate-to-external-ram.ld SF2_GNU_SC4_pwm_slow_blink\CMSIS\system_m2sxxx.c SF2_GNU_SC4_pwm_slow_blink\CMSIS\system_m2sxxx.h SF2_G | | |
| Help | Clos | e |

Figure 50 - GPIO Simple Blink project files

- 9. Click **Close** to close the Report window.
- 10. Create the MSS RTC sample project by selecting SmartFusion2_MSS_RTC_Driver_0 on the DESIGN_FIRMWARE tab, then right clicking and selecting **Generate Sample Project > Cortex-M3 > SoftConsole v4.0 > RTC Time**.

| | Reports | ₽× | StartPage 🗗 × 🛛 SD DESIGN_FIRMWAR | 8× | | | | | | - | | |
|----|----------|---|-----------------------------------|--------------------------------|-------------------------------|-----------------------------|------------------------|---------|------------------------------|---------------------------------------|--|--|
| | 9 | | | | | | | | | | | |
| | Generate | | Instance Name | Core Type | | | Versi | ion | Compatible Hardware Instance | | | |
| 1 | 7 | CorePWM_Driver_0 | | | CorePWM_Driver | | | 2.4.100 | - | <pre>SF2_MSS_sys_sb:corepwm_0_0</pre> | | |
| 2 | 7 | SmartFusion2_CMSIS_0 | | | SmartFusion2_CMSIS | | | 2.3.105 | - | SF2_MSS_sys_sb_MSS | | |
| 3 | v | SmartFusion2_MSS_GPIO_Driver_0 | | | SmartFusion2_MSS_GPIO_Driver | | | 2.1.102 | - | ▼ SF2_MSS_sys_sb_MSS:GPIO | | |
| 4 | 7 | - | SmartFusion2_MSS_HPDMA_Driver_0 | | SmartFusion2_MSS_HPDMA_Driver | | | 2.2.100 | - | · SF2_MSS_sys_sb_MSS | | |
| 5 | 7 | - | SmartFusion2_MSS_MMUART_Driver_0 | SmartFusion2_MSS_MMUART_Driver | | | 2.1.100 | - | SF2_MSS_sys_sb_MSS:MMUART_0 | | | |
| 6 | 6 🔽 🖉 | | SmartFusion2_MSS_MMUART_Driver_1 | SmartFusion2_MSS_MMUART_Driver | | | 2.1.100 | - | SF2_MSS_sys_sb_MSS:MMUART_1 | | | |
| 7 | 7 🔽 🖉 | | SmartFusion2_MSS_NVM_Driver_0 | SmartFusion2_MSS_NVM_Driver | | | 2.5.100 | - | · SF2_MSS_sys_sb_MSS | | | |
| 8 | | | SmartFusion2_MSS_RTC_Driver_0 | Disable Conorat | lonore star | _MSS_RTC_Driver | | 2.2.100 | - | SF2_MSS_sys_sb_MSS:RTC | | |
| 9 | 7 | SmartFusion2_MSS_System_Services_Driver | | Disable General | | _MSS_System_Services_Driver | | 2.9.100 | - | · SF2_MSS_sys_sb_MSS | | |
| 10 | 10 🔽 🖷 | | SmartFusion2_MSS_Timer_Driver_0 | Show Details | | _MSS_Timer_Driver | | 2.2.100 | - | · SF2_MSS_sys_sb_MSS | | |
| | | | | Open Document | tation 🔸 | | | | | | | |
| | | | | Generate Sampl | e Project 🔸 | Cortex-M3 ► | IAR Embedded Workbench | | | | | |
| | | | | | | | Keil-MDK | . | | | | |
| | | | | | | | SoftConsole v3.4 | . | | | | |
| | | | | | | | SoftConsole v4.0 | RTO | C Tim | ne | | |
| | | | | | | | RTO | C Inte | errupt | | | |
| | | | | | | | RTO | C Cale | endar Alarm | | | |

Figure 51 - Generating the RTC Driver sample project

- 11. Confirm the following settings in the Generate Sample Options dialog box then click **OK**:
 - Folder: C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole
 - Show generation report: checked


| Generate Sample Options | | ? | × |
|---|--------------------|---------|----------|
| Samples folder: C:\Microsemiprj\SMF2000-M3_Base | Design\SoftConsole | | |
| Files will be generated in: C: \Microsemiprj \SMF2000-M3_BaseDesign \SoftConso | ole\SF2_GNU_SC4_R | TC_time | |
| Show generation report | | | |
| Help | ОК | Cance | <u>.</u> |

Figure 52 – RTC Time sample project location

12. The Report dialog box will list all the files generated and the location.

| 🕑 Report | ? | \times |
|--|------|----------|
| Save | 🖨 Pr | int |
| Files generated in 'C:\Microsemipr)\SMF2000-M3_BaseDesign\SoftConsole': | | - |
| SF2_GNU_SC4_RTC_time\cyclocct SF2_GNU_SC4_RTC_time\cyclocct SF2_GNU_SC4_RTC_time\cyclocct SF2_GNU_SC4_RTC_time\cyclocct | | |
| SF2_GNU_SC4_RTC_time\CMSIS\mss_assert.h SF2_GNU_SC4_RTC_time\CMSIS\startup_gcc\debug-in-microsemi-smartfusion2-envm.ld SF2_GNU_SC4_RTC_time\CMSIS\startup_occ\debug-in-microsemi-smartfusion2-esram.ld | | |
| SF2_GNU_SC4_RTC_time\CMSIS\startup_gcc\debug-in-microsemi-smartfusion2-external-ram.ld SF2_GNU_SC4_RTC_time\CMSIS\startup_gcc\newlib_stubs.c SF2_GNU_SC4_RTC_time\CMSIS\startup_gcc\newlib_stubs.c | | |
| SF2_GNU_SC4_RTC_time\CMSIS\startup_gcc\production-smartfusion2-relocate-to-external-ram.ld SF2_GNU_SC4_RTC_time\CMSIS\startup_gcc\startup_m2sxxx.s SF2_GNU_SC4_RTC_time\CMSIS\startup_gcc\startup_m2sxx.c | | |
| SF2_GNU_SC4_RTC_time\CMSIS\system_m2sxxx.h SF2_GNU_SC4_RTC_time\CMSIS\sys_int_cfg_types.h SF2_GNU_SC4_RTC_time\driversymss_trchmss_trc. | | |
| SF2_GNU_SC4_RTC_time\drivers\mss_rtc\mss_rtc.h SF2_GNU_SC4_RTC_time\drivers\mss_uart\mss_uart.c SF2_GNU_SC4_RTC_time\drivers\mss_uart\mss_uart.h | | |
| SF2_GNU_SC4_RTC_time\drivers\mss_uart\mss_uart_regs.h SF2_GNU_SC4_RTC_time\drivers_config\sys_config\sys_config.c SF2_GNU_SC4_RTC_time\drivers_config\sys_config\sys_config.h | | |
| SF2_GNU_SC4_RTC_time\drivers_config\sys_conf | | |
| SF2_GNU_SC4_RTC_time\hal\CortexM3\GNU\cpu_types.h SF2_GNU_SC4_RTC_time\hal\CortexM3\GNU\pu_hal.s SF2_GNU_SC4_RTC_time\hal\CortexM3\GNU\hal.s | | <u> </u> |
| Help | Clos | e |

Figure 53 - RTC_time project files

- 13. Click **Close** to close the Report window.
- 14. Close the DESIGN_FIRMWARE tab (File > Close DESIGN_FIRMWARE). Select Yes if prompted about saving changes to DESIGN_FIRMWARE.

Exporting firmware configuration files and firmware drivers

The firmware used in a SoftConsole project must match the target hardware configuration. For SmartFusion2 projects, Libero SoC generates specific firmware files that are required to ensure that the SoftConsole project matches and is compatible with the target hardware. The sample projects created in the previous step contain generic firmware files that may not match the target hardware. In this step, you will export firmware configuration files that match the design configuration.



15. Select Export Firmware under Handoff Design for Firmware Development on the Design Flow tab, then right-click and select **Export Firmware...** to create the firmware drivers for the design.

| Design Flow & | × |
|---|---|
| Top Module(root): SF2_MSS 🛛 D 👔 🌮 | |
| Active Synthesis Implementation: synthesis | |
| Tool | ĺ |
| 🕀 🕨 Constraints | |
| Manage Constraints | |
| 🖌 🖻 🕨 Implement Design | |
| - 🚱 Open Netlist Viewer | |
| V Synthesize | |
| Verify Post-Synthesized Design | |
| Configure Flash*Freeze | |
| -• Configure Register Lock Bits | |
| ✓ Biace and Route | |
| Verify Post Layout Implementation | |
| 🖻 🕨 Program and Debug Design | |
| Generate FPGA Array Data | |
| Update eNVM Memory Content | |
| Configure Hardware | |
| — S Configure Programming Options | |
| Configure Security | |
| 🖌 🖻 🕨 Program Design | |
| V Generate Bitstream | |
| Run PROGRAM Action | |
| 🖻 🕨 Debug Design | |
| - Kalentify Debug Design | |
| 🖙 🐵 SmartDebug Design | |
| Handoff Design for Production | |
| Handoff Design for Firmware Development | |
| - Sonfigure Firmware Cores | |
| Export Firmware | |
| B Handoff Design for Debu | |
| Help | |
| Design Flow Design Hierarchy Stimulus Hierarchy Catalog Files | |

Figure 54 - Exporting the firmware configuration files

Checked

- 16. Enter the following in the Export Firmware dialog box then click **OK**:
 - Location:
 - Software IDE:

Accept the default location

select SoftConsole4.0 from the pull-down menu

- Export hardware configuration:

| Export Firmware | 2 | | ? | × |
|---------------------|--|---------------|-------|------|
| Location: | C:\Microsemiprj\SMF2000-M3_BaseDesign | | Brows | e |
| Software IDE: Soft | Console4.0 | | | |
| Export hardware | configuration and firmware drivers | | | |
| C Create software p | project including hardware configuration and fir | mware drivers | | |
| | | | | |
| | | | | |
| Help | | ОК | Cance | el 🔤 |

Figure 55 - Export Firmware options

17. Click **OK** in the dialog box that indicates the location of the firmware cores.



Figure 56 - Firmware driver location

18. The firmware drivers will be visible on the Libero SoC Files tab. The Project for selected toolchain and the sample projects will be visible on the Files tab in the SoftConsole folder. If the projects are not visible, select View > Refresh Design Hierarchy from the Libero SoC menu.



Figure 57 - Sample projects on Libero SoC Files tab



19. The Data Sheet containing the memory map will be visible in the Libero SoC Report tab.

| Reports 🗗 × StartPage 🗗 × | | , |
|---|--|---|
| Broject Summary St2_MSS reports Components Generate Memory Map St2_MSS_DataSheet.xml SynPiffylog St2_MSS_cm | | ⊡ Data Sheet: SF2_MSS |
| SF2_MSS_dsp_rpt.txt SF2_MSS_dsp_rpt.txt SF2_MSS_ram_rpt.txt SF2_MSS_romile_netlist_resources.xml | | Project Settings |
| | FAM: Die: Package: Speed Grade: Voltage: HDL: Project Descriptio Location: State (Time): | SmartFusion2 M2S010 400 VF STD 1.2 VHDL n: C:/Microsemipri/SMF2000-M3_BaseDesign/component/work/SF2_MSS GENERATED (Thu Aug 20 17:37:20 2020) |
| SF2_MSS_PROGRAM.log | | Table of Contents |
| | Generated Files IO's Hardware Instances Firmware Memory Map | 2 |
| | | Generated Files |
| | Figure | 58 – Data Sheet for the design |

- 20. Scroll in the SF2_MSS data sheet and become familiar with the Generated Files, Firmware and Memory Map sections (click on the hyperlink at the top of the data sheet to move to the section of interest).
- 21. Select the Memory Map to become familiar with the locations of the peripherals.

What is the address of MMUART_0? _____

What is the address of corepwm?

22. Minimize Libero SoC.



Step 7 - Debugging with SoftConsole v6.4

Running the CorePWM slow blink application

In this step, you will launch SoftConsole v6.4, import the sample projects created from Libero SoC and run the CorePWM slow blink application from the SmartFusion2 eSRAM.

1. Click **Start > Programs > Microsemi > SoftConsole v6.4**, or click the shortcut on your desktop. The SoftConsole 6.4 Workspace Launcher may open as shown below.

| sc SoftConsole | - | | × |
|--|---------|-----------|-----|
| SoftConsole | | | |
| Select a directory as workspace | | | |
| Workspace:/extras/workspace.example | | Browse | |
| Populate the new workspace with SoftConsole's recommended settings | | | |
| > Recent Workspaces | | | |
| | Launch | Can | cel |
| Figure 59 - SoftConsole 6.4 Workspace Launcher (locatio | n shown | may diffe | er) |

2. Click the browse button in the Workspace Launcher and navigate to C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole then click **OK**.

| Browse For Folder | × |
|----------------------------|---|
| | |
| SMF2000-M3_BaseDesign |] |
| > component | |
| > constraint | |
| > designer | |
| > firmware | |
| hdl | |
| simulation | |
| smartgen | |
| ✓ SoftConsole | |
| filelist | |
| SF2_GNU_SC4_pwm_slow_blink | |
| > SF2_GNU_SC4_RTC_time 🗸 | |
| | - |
| Eolder: SoftConsole | |
| Make New Folder OK Cancel |] |





3. Click Launch in the Workspace Launcher to open the SoftConsole GUI.

| SC SoftConsole | _ | | × |
|--|-------|---------|----|
| SoftConsole | | | |
| Select a directory as workspace | | | |
| Workspace: C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole | | Browse. | |
| > Recent Workspaces | | | |
| L | aunch | Canc | el |

Figure 61 – Eclipse Launcher with workspace selected.

4. The SoftConsole v6.4 GUI will open.



Figure 62 - SoftConsole v6.4 GUI

If the workspace launcher did not open, select **File > Switch Workspace > Other** and navigate to the workspace location shown on the previous page.



Importing the CorePWM slow blink and RTC Time SoftConsole Projects

 Import the SoftConsole Projects by selecting File > Import... from the SoftConsole menu. The Import dialog box will open. Expand General and select Existing Projects into Workspace then click Next.

| SC Import | | × |
|---|--------|---|
| Select Create new projects from an archive file or directory. | Ľ | 5 |
| Select an import wizard: | | |
| type filter text | | |
| General Archive File Existing Projects into Workspace File System Preferences Projects from Folder or Archive C/C++ Git Install | | * |
| (?) < <u>B</u> ack <u>N</u> ext > <u>F</u> inish | Cancel | |

Figure 63 - SoftConsole Import dialog box

- 6. Enter the following in the Import Projects dialog box then click **Finish**:
 - Select root directory: Browse to C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole Click Select Folder in the Select Folder dialog box.
 - Projects: "SF2_GNU_SC4_pwm_slow_blink" and "SF2_GNU_SC4_RTC_time" checked
 - Options:
 - Search for nested projects: un-checked (default)
 - Copy projects into workspace: un-checked (default)
 - Close newly imported projects upon completion: un-checked (default)
 - Hide projects that already exist in the workspace: un-checked (default)
 - Click **Finish** to import the two projects into the current Workspace.



| SC Import | | | × |
|---|---|--|-----------|
| Import Projects Select a directory to sear | ch for existing Eclipse projects. | | 7 |
| Select root directory: Select archive file: Projects: | C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole | B <u>r</u> owse. | |
| SF2_GNU_SC4_pw | vm_slow_blink (C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole\SF2_GNU_SC4_pwm_slow_blink) C_time (C:\Microsemiprj\SMF2000-M3_BaseDesign\SoftConsole\SF2_GNU_SC4_RTC_time) | <u>S</u> elect A <u>D</u> eselect A | II All |
| Options Searc <u>h</u> for nested pro <u>C</u> opy projects into w Cl <u>o</u> se newly importe H <u>i</u> de projects that all | ojects orkspace d projects upon completion ready exist in the workspace | | |
| Working sets | ting sets | Ne <u>w</u> S <u>e</u> lect | |
| ? | < <u>B</u> ack <u>N</u> ext > <u>F</u> inish | Cancel | |

Figure 64 - Importing the sample projects into the workspace



7. The SoftConsole projects will be visible in the SoftConsole Project Explorer.

| SC SoftConsole - SoftConsole v6.4.0.412 | | | | | | - | | × |
|--|---|-------------------------------------|-----------------|---------------|----------|----------------|----------------|-------|
| <u>F</u> ile <u>E</u> dit <u>S</u> ource Refac <u>t</u> or <u>N</u> avigate Se | arch <u>P</u> roject <u>R</u> un <u>V</u> | <u>N</u> indow <u>U</u> ltraDevelop | <u>H</u> elp | | | | | |
| : 🖻 🕶 🔚 🕞 🔊 🖛 🔨 🖛 : 🏠 🕶 : (| ヨ: 🏊 🝈 🔨 🖷 🛛 🔪 | 🗙 🛛 🚓 🗄 🏪 🗄 合 🗄 ру | i 📸 👻 🔂 👻 🕞 👻 🚱 | 🔋 🛨 🎄 👻 🕥 👻 🖗 | 💁 - : 👝 | 🛷 🚽 🔛 | T I | |
| | | · @. #. ¤. | | | | | ۹ ا | 2 6 |
| Project Explorer 🛛 🗖 🗖 | | | | | | ⊃ ⊠ ® B | » 1 | |
| E SF2_GNU_SC4_pwm_slow_blink > E SF2_GNU_SC4_RTC_time | | | | | The | re is no activ | re editor ine. | that |
| | Problems 🛛 🧔 | Tasks 📮 Console 📃 | Properties | | | ٢ | 8 | |
| | U items | ^ | _ | | | _ | | |
| | Description | | Resource | Path | Location | Туре | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |
| | | | | | | | | |

Figure 65 - SoftConsole projects in the workspace

8. Close the SF2_GNU_SC4_RTC_time project by selecting the project name in the Project Explorer then right-clicking and selecting **Close Project**.



Importing the firmware configuration files

9. Import the firmware configuration files that were exported from Libero SoC into the

SF2_GNU_SC4_pwm_slow_blink project by selecting SF2_GNU_SC4_pwm_slow_blink in the Project Explorer then selecting **File > Import** from the SoftConsole menu. The Import dialog box will open.

10. Expand the General category in the Import dialog box and select **File System** and then click **Next**.

| SC Import | | \times |
|---|--------|----------|
| Select Import resources from the local file system into an existing project. | P | 1 |
| Select an import wizard: | | |
| type filter text | | |
| Existing Projects into Workspace File System Preferences Projects from Folder or Archive C/C++ Git Install Run/Debug Team | | < > |
| ⑦ < <u>Back</u> <u>Next ></u> Einish | Cancel | |

Figure 66 - Importing the firmware configuration files

- 11. The File system dialog box will open. Click the **Browse** button. Navigate to the C:\Microsemiprj\SMF2000-M3_BaseDesign\firmware folder and click **Select Folder**.
- 12. Enter the following in the File system dialog box:
 - From directory: C:\Microsemiprj\SMF2000-M3_BaseDesign\firmware
 - Select
 - Click next to the firmware folder in the left window pane to expand it (circled in the figure below)
 - Make sure the following folders are enabled (checkmark):
 - CMSIS
 - drivers
 - drivers_config
 - hal
 - With firmware folder selected on the left side, un-check "README.txt", "SF2_MSS_cmsis.svd.xml" and "SF2_MSS_hw_platform.h" on the right side
 - Into Folder: SF2_GNU_SC4_pwm_slow_blink

| SC Import | _ | | \times |
|---|--|-----------------|----------|
| File system Import resources from the local file system. | | | |
| From directory: C:\Microsemiprj\SMF2000-M3_Basel | Design\firmware v | B <u>r</u> owse | |
| Firmware CMSIS Chivers Chivers_config Filelist Chief hal | README.txt SF2_MSS_cmsis_svd.xml SF2_MSS_hw_platform.h | | |
| Filter Types Select All Into folder: SF2 GNU SC4 pwm slow blink |] | Browse | |
| Options Options Overwrite existing resources without warning Create top-level folder Advanced >> | | 0.0 <u></u> | |
| ? < <u>B</u> ack | Next > <u>Finish</u> | Cancel | |

Figure 67 - Importing the firmware configuration files into the project

13. Click Finish. Click Yes To All in the Question dialog box when prompted about Overwriting files.



Figure 68 - Question dialog box



14. The project should appear in the Project Explorer as shown in the figure below.



Figure 69 - SF2_GNU_SC4_pwm_slow_blink after importing the firmware configuration files

15. Double click main.c in the Project Explorer window to open the file in the SoftConsole C/C++ editor. Scroll through the file to become familiar with it.

| le main.c ⊠ | |
|--|---|
| 10/************************************ | ^ |
| 2 * (c) Copyright 2015 Microsemi SoC Products Group. All rights reserved. 3 * | |
| 4 * This example project demonstrates control control the duty cycle of | |
| 6 * | |
| 7 * Please refer to the file README.txt for further details about this example. 8 * | |
| 9 * SVN \$Revision: 8042 \$ | |
| 10 * SVN \$Date: 2015-10-15 17:55:12 +0530 (Thu, 15 Oct 2015) \$ | |
| 11 */ 12 #include "platform b" | |
| 13 #include "core pwm.h" | |
| 14 | |
| 15@/************************************ | |
| 16 * Delay count used to time the delay between duty cycle updates. 17 ************************************ | |
| 18 #define DELAY_COUNT 10000 | |
| 20@/************************************ | |
| 21 * PWM prescale and period configuration values. 22 ************************************ | |
| 23 #define PWM PRESCALE 8 | |
| 24 #define PWM_PERIOD 100 | |
| 26 /************************************ | |
| 27 * CorePWM instance data. | |
| 28 ************************************ | |
| 30 | ~ |

Figure 70 - main.c in the SoftConsole editor



16. Double click platform.h in the Project Explorer window to open the file in the SoftConsole C/C++ editor. Compare the COREPWM_BASE_ADDR on line 16 to the address recorded on page 40. Modify and save the file if necessary.

Hint: The default address value shown below is incorrect and needs to be updated according to the memory map.



Confirming the SoftConsole v6.4 Project Settings

The next steps to confirm the sample project settings prior to performing a build. SoftConsole supports multiple build configurations. By default, projects contain two build configurations: *Debug* and *Release*. Some project settings apply to all build configurations while others apply to a specific build configuration.

- 17. Open the Project Properties dialog box by clicking on the project name (SF2_GNU_SC4_pwm_slow_blink) in the Project Explorer and selecting **Project > Properties**.
- 18. Navigate to **C/C++ Build > Settings** in the Properties for SF2_GNU_SC4_pwm_slow_blink dialog box.

| SC Properties for SF2_GNU_SC4_pwm_slow_blink – 🗆 🗙 | | | | | |
|---|--|---------------------|--|---|--|
| type filter text Settings 🗘 🔻 🖒 | | | | | |
| > Resource Builders > C/C++ Build Build Variables Environment Logging Settings | Configuration: Debug [Active] | eps 😤 Build Artifac | Manage Configurations | ^ | |
| Tool Chain Editor | Iarget Processor A Optimization | ARM family | cortex-m3 ~ | | |
| > C/C++ General | 🖉 Warnings | Architecture | Toolchain default ~ | | |
| Linux Tools Path | 🖄 Debugging | Instruction set | Thumb (-mthumb) v | | |
| Project References | S Cross ARM GNU Assembler Preprocessor | Thumb interwo | rk (-mthumb-interwork) | | |
| Run/Debug Settings | Includes | Endianness | Toolchain default ~ | | |
| > Task Repository Task Tase | 🖄 Warnings | Float ABI | Toolchain default V | | |
| > Validation | Miscellaneous | FPU Type | Toolchain default $\!$ | | |
| WikiText | S Cross ARM C Compiler | Unaligned access | Toolchain default | | |
| | Includes | AArch64 family | Generic (-mcpu=generic) | | |
| | Optimization | Feature crc | Toolchain default | | |
| | Warnings Miscellaneous | Feature crypto | Toolchain default | | |
| | v 🗞 Cross ARM C Linker | Easture fr | | | |
| | 🖉 General | Feature ip | rooichain default | | |
| | Libraries Miscellaneous | Feature simd | Enabled (+simd) | | |
| | ✓ S Cross ARM GNU Create Flash Image | Code model | Small (-mcmodel=small) ~ | | |
| | 🖉 General | Strict align (-m | strict-align) | | |
| | V S Cross ARM GNU Print Size | Other target flags | | | |
| | General | | | ~ | |
| ? | | | Apply and Close Cancel | | |

а 🔨 **Місвоснір** company

Figure 72 - Project Properties dialog

Select different Build configurations by using the pull-down menu in the Configuration field of the Project Properties dialog box (highlighted in the figure above). Some project settings are applicable to all build configurations while others are for a specific build configuration.

19. Select *Configuration = [All configurations]* to configure settings applicable to both the Debug and Release build targets.

| Settings | ← ▼ ↔ ▼ ▼ |
|---------------------------------------|-------------------------|
| Configuration: [All configurations] | ▼ Manage Configurations |



- 20. Select *Cross ARM C Compiler > Includes* on the Tool Settings tab. Confirm that the Include paths (-I) field contains the following:
 - ../drivers_config/sys_config
 - ../drivers/CorePWM
 - ../CMSIS
 - ../hal/CortexM3/GNU
 - ../hal/CortexM3
 - ..
 - ../hal



| Settings | | | | - ⇒ • • |
|---|--|-------------------------|--|---------|
| Configuration: [All configurations] | teps 🙅 Build Artifact 🗟 Binary F | Parsers 📀 Error Parsers | Manage Configura | tions |
| Target Processor Optimization Warnings Debugging Cross ARM GNU Assembler Preprocessor Includes Warnings Miscellaneous Scross ARM C Compiler Preprocessor Includes Preprocessor Warnings Miscellaneous Scross ARM C Compiler Preprocessor Miscellaneous | Include paths (-I) /drivers/CorePWM /CMSIS /hal/CortexM3/GNU /hal/CortexM3 /hal | | All All | E |
| | Figure 74 - Project | Include paths | | |

If necessary, add or modify the Include paths by clicking the Add directory path button (🗐) or the Edit directory path

button (🗟).



21. SmartFusion2 projects require a setting in order for the preprocessor to find the toolchain CMSIS header files. Select *Cross ARM C Compiler > Miscellaneous* on the Tool Settings tab. Confirm that the Other compiler flags field contains --specs=cmsis.specs.

| 🛞 Tool Settings 🛞 Toolchains 🎤 Build St | eps 🎐 Build Artifact 🗟 Binary Parsers 😣 Error Parsers |
|---|--|
| 🛞 Target Processor | Generate assembler listing (-Wa,-adhIns="\$@.lst") |
| Optimization | Save temporary files (save-temps Use with caution!) |
| Warnings | Verbose (-v) |
| Debugging | |
| Cross ARM GNU Assembler | Other compiler flagsspecs=cmsis.specs |
| Preprocessor | |
| 🖄 Includes | |
| 🖄 Warnings | |
| 🖄 Miscellaneous | |
| Cross ARM C Compiler | |
| Preprocessor | |
| Includes | |
| Optimization | |
| Warnings | |
| | |
| General | |
| | |
| Miscellaneous | |
| Source Area Contraction C | |
| 🖄 General | |
| a 🛞 Cross ARM GNU Print Size | |
| 🖄 General | |
| Figure 75 Con | i firming the Gross ADM C Compiler Other compiler flags setting |
| rigure 75 - Con | mining the cross Artivi C complier Other complier hdgs setting |

The appropriate linker script must be configured for the project build configuration. The sample projects include linker scripts bundled with the CMSIS/HAL firmware core. Follow the steps below to confirm the linker script setting for the Debug and Release build configurations.

- 22. Select *Configuration = Debug [Active]* from the pull-down menu in the Configuration field of the Project Properties dialog box.
- 23. Select *Cross ARM C Linker > General* on the Tool Settings tab of the Project Properties dialog box. Confirm that the Script files (-T) contains **../CMSIS/startup_gcc/debug-in-microsemi-smartfusion2-esram.ld**. This linker script builds an application that runs from the SmartFusion2 eSRAM.



| Settings | | ⟨¬ ▼ ¬ |
|--|--|--|
| Configuration: Debug [Active] | | ▼ Manage Configurations |
| Tool Settings Toolchains Puild State Target Processor Optimization Warnings Cross ARM GNU Assembler Preprocessor Includes Warnings Miscellaneous Sortiation Miscellaneous Sortiation Warnings Includes Optimization Miscellaneous Cross ARM C Linker Core ARM C Linker Core ARM C Linker | eps Paild Artifact Binary Parsers Error Parsers Script files (-T) /CMSIS/startup_gcc/debug-in-microsemi-smartfusion2-esram.ld | |
| Wiscenarieous | Figure 76 - Debug configuration linker script | |

- 24. Select *Configuration = Release* from the pull-down menu in the Configuration field of the Project Properties dialog box.
- 25. Select *Cross ARM C Linker > General* on the Tool Settings tab of the Project Properties dialog box and confirm that the Script files (-T) field contains **../CMSIS/startup_gcc/debug-in-microsemi-smartfusion2-envm.ld**. This linker script builds an application that runs from the SmartFusion2 eNVM.



26. After confirming the settings listed in the previous steps, click **Apply and Close** in the Properties for SF2_GNU_SC4_pwm_slow_blink dialog box.

Building the Project

After configuring the project settings, the next step is to build the project. The build configuration may need to be set prior to building the project.

27. Select *SF2_GNU_SC4_pwm_slow_blink* in the Project Explorer then and select **Project > Build Configurations > Set Active**. Confirm that a check mark appears next to **Debug** indicating it is the selected build configuration. If necessary click **Debug**.

| sc SoftConsole - SF2_GNU_SC4_pwm_slow_blink/m | ain.c - SoftConsole v6.4.0.412 | 2 | |
|--|--------------------------------|---------------|--|
| File Edit Source Refactor Navigate Search | Project Run Window U | UltraDevelop | Help |
| ☆ + 🔚 🕼 🥸 + 🇞 + 📾 約 + i 🖴 ið 🖗 + 🖗 + 🏷 (> + ◇ + 🛃 | Open Project Close Project | | ☞ _@ ≂ ∞ & 1 = 0 : 0 : 0 : 0 + 6 + 6 + 6 + 6 |
| 🔁 Project Explorer 🐹 🛛 📄 🔄 🏹 🖇 🖻 | 🗟 Build All | Ctrl+B | © 0x2000136e 💽 core_pwm.c |
| ✓ | Build Configurations | > | Set Active > 🗸 1 Debug * |
| > 🎉 Binaries | Build Project | | Manage 2 Release d |
| > 🗊 Includes | Build Working Set | > | Build by Working Set |
| > 🔁 Civisis | Clean | | Set Active by Working Set |
| > 🔁 drivers | Build Automatically | | Manage Working Sets etails about this ex |
| > 🔁 drivers_config | Build Targets | > | b. ¢. |
| > 👝 hal | | | -15 17:55:12 +0530 (Thu, 15 Oct 2015) \$ |
| > C main.c | C/C++ Index | > | |
| > B README.txt | Properties | | |
| SF2_GNU_SC4_RTC_time | 14 | | 2 |
| 1 | Figure 78 - Selecting th | ne build conf | iguration |

28. With the project name still selected in the Project Explorer, select **Project > Build All** or by clicking the Build All icon

() on the SoftConsole toolbar to build the project. The results of the build process can be viewed in the Console view and the Problems view. Confirm that there are no errors or warnings in the Problems view.

| 🖹 Problems 🛿 🧔 Tasks 📮 Console 🔲 Properties | | | | ◎ ▽ □ □ | |
|---|----------|------|----------|----------------|--|
| 0 items | | | | | |
| Description | Resource | Path | Location | Туре | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |
| | | | | | |

Figure 79 - Problems view after building the project

Debugging with SoftConsole

After building the project, the next step is to use the SoftConsole debugger to run the application on the SmartFusion2 Cortex-M3. Before using the On-chip Debugger (OCD), the debug launch configuration must be created.

Creating a debug launch configuration

- 29. Select SF2_GNU_SC4_pwm_slow_blink in the SoftConsole Project Explorer View.
- 30. Select Run > Debug Configurations... from the SoftConsole menu. The Debug Configurations dialog will open.
- 31. Select *GDB OpenOCD Debugging* in the Debug Configurations dialog box then right-click and select **New Configuration** to create a new debug launch configuration for the Debug build configuration.

| SC Debug Configurations | | – 🗆 X |
|---|---|---|
| Create, manage, and run con | figurations | TO- |
| Image: Second | Configure launch settings from thi | is dialog: n' button to create a configuration of the selected type. *ton to create a launch configuration prototype of the selected type. port the selected configurations. |
| Kobot Kobot Kemote Sobot UltraDevelop Agent Sobut UltraDevelop OCD So UltraDevelop Remote | Duplicate | copy the selected configuration. move the selected configuration. figure filtering options. |
| LU | Link Prototype Unlink Prototype Reset with Prototype Values | uration by selecting it. and then select 'Link Prototype' menu item to link a prototype. ay and then select 'Unlink Prototype' menu item to unlink a prototype. |

Figure 80 - Creating a new debug launch configuration

32. Ensure that the C/C++ Application field on the Main tab of the Debug Configurations dialog box contains Debug\SF2_GNU_SC4_pwm_slow_blink.elf

| SC Debug Configurations | - | |
|--|--|---------------|
| Create, manage, and run configu | urations | Ť. |
| Image: Second system Image: Second system | Name: SF2_GNU_SC4_pwm_slow_blink Debug Main Startup Project: SF2_GNU_SC4_pwm_slow_blink SF2_GNU_SC4_pwm_slow_blink C/C++ Application: Debug\SF2_GNU_SC4_pwm_slow_blink.elf Yariables Build (if required) before launching Build Configuration: Select Automatically O Disable auto build Image: Select Settings Configure Workspace Settings | Browse Browse |
| < >> Filter matched 8 of 14 items | Re <u>v</u> ert | Apply |
| ? | Debug | Close |

Figure 81 - Debug launch configuration

- 33. Select the Debugger tab of the Debug Configurations dialog box. The Config Options field must contain the correct command line options/script to be passed to OpenOCD.
- 34. Enter --command "set DEVICE M2S010" --file board/microsemi-cortex-m3.cfg in the Config Options field.

--command "set DEVICE M2SOXX" specifies the target device, modify based on the target silicon. --file board/microsemi-cortex-m3.cfg is a board script for the integrated Cortex-M3 core of SmartFusion2

- 35. Scroll to the GDB Client Setup field on the Debugger tab. Confirm that the following appear:
 - **Executable:** \${cross prefix}gdb\${cross suffix}
 - Commands: set mem inaccessible-by-default off. If necessary, delete the lines "set arch riscv:rv32" and "set riscv use_compressed_breakpoints no"

| sc Debug Configurations | | — 🗆 X |
|--|---|--|
| Create, manage, and run configurat | ions | The second secon |
| | | |
| | Name: SF2_GNU_SC4_pwm_slow_blink Debug | |
| type filter text × | 📄 Main 🕸 Debugger 🌘 Startup 🦃 Source 🔲 Common 🔀 SVD Path | |
| B OpenOCD Debugging | OpenOCD Setup | ^ |
| SF2_GNU_SC4_pwm_slow_blink Debug unch Group | Start OpenOCD locally | |
| bot | Executable path: \${openocd_path}/\${openocd_executable} | Browse Variables |
| bot Remote raDevelop Agent | Actual executable: C:\Microchip\SoftConsole-v6.4\eclipse\//openocd/bin/openo | cd.exe |
| raDevelop OCD | (to change it use the <u>global</u> or <u>workspace</u> preferences pages or th | e <u>project</u> properties page) |
| :raDevelop Remote Target | GDB port: 3333 | |
| | leinet port: 4444 | |
| | | |
| | command set DEVICE M2S010file board/microsemi-corte | k-ms.crg |
| | | |
| | Allocate console for OpenOCD | the telnet connection |
| | GDB Client Setup | |
| | ☑ Start GDB session | |
| | Executable name: \${cross_prefix}gdb\${cross_suffix} | Browse Variables |
| | Actual executable: arm-none-eabi-gdb | |
| | Other options: | |
| | Commands: set mem inaccessible-by-default off | ^ |
| | | |
| | | × |
| | Remote Target | |
| | Host name or IP address: localhost | |
| | Port number: 3333 | |
| | Force thread list update on suspend | |
| | | Restore defaults |
| < >> | | Re <u>v</u> ert Appl <u>v</u> |
| The matched o of 14 items | | |
| ? | | <u>D</u> ebug Close |
| Figure 82 | 2 - SoftConsole v6.4 Debugger tab settings for the SmartFusion2 SN | //F2000 kit |

36. Select the Startup tab and confirm that the default settings are configured as shown in the figure below.

• Disable (un-check) "Pre-run/Restart" under Run/Restart Commands

| SC Debug Configurations | | - 🗆 | × |
|--|--|-----------|---|
| Create, manage, and run configu | Irations | 1 | 5 |
| | | X | |
| Image: point of the second | Name: SF2_GNU_SC4_pwm_slow_blink Debug Main Debugger Startup Source Common SVD Path Initialization Commands Initialization Commands Initial Reset. Type: Initialization Commands Icoad Symbols and Executable Icoad symbols and Executable Icoad executable | le System | |
| < >> Filter matched 8 of 14 items | □ Set program counter at (hex): ☑ Set breakpoint at: ☑ Continue | Apply | ~ |
| ? | <u>D</u> ebug | Close | |

Figure 83 - Startup tab settings

- 37. Click **Apply** to save the changes.
- 38. Click **Debug** to launch the Debugger. Click **Switch** in the **Confirm Perspective Switch** dialog box. The SoftConsole Debug perspective will open. The code will be downloaded to the SmartFusion2 SMF2000 board. The program will be suspended at the first line of main() as shown in the figure below. Use for example F6 key to execute code single lines (Step Over).

| File [af Source Refactor Newsjar Sagch Bojert Ban Window Undowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Window Undowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Window Undowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Window Undowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Window Undowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Window Undowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Window Undowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Window Undowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Windowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Windowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Windowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Windowedor Help File [af Source Refactor Newsjar Sagch Bojert Ban Windowedor Help File [af Source Refactor Newsjar Sagch Ban Windowedor Help File [af Source Refactor Newsjar Sagch Ban Windowedor Help File [af Source Refactor Newsjar Sagch Ban Windowedor Help File [af Source Refactor Newsjar Sagch Ban Windowedor Help File [af Source Refactor News] File [af Source Ref | SC SoftConsole - SF2_GNU_SC4_pwm_slow_blink/main.c - S | oftConsole v6.4.0.412 | | | | - C | × נ |
|--|--|--|---|---------------------------------|--|-----------|---------|
| Consider and the set of t | File Edit Source Refactor Navigate Search Project | <u>Run W</u> indow <u>U</u> ltraDevelop <u>H</u> elp | | | | | |
| <pre>% Debug % Project Environment (%) pigteform (%) Do200038e</pre> | 🗂 - 🔚 🐚 🏔 🏝 - ! 🖻 🥐 🕼 🖫 🖫 🔌 | 🕨 🗉 🖉 🗷 👁 . R 🖬 🗮 🗷 🕹 | 습 🖭 🕸 • 🛈 • 💁 • 🥭 🛷 | - 🍠 🎙 - 👸 | • * • • • • • • | Q E | } 晒 林 |
| <pre> Sty CHU SC Jewn show Jiho Wolf (SBB Check Sy CHU SC Jewn show Jiho Wolf (SBB Check Sy CHU SC Jewn show Jiho Wolf (SBB Check Jewn show Jiho Kalf Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Void); Sy CHU SC Jewn show Jiho Kalf (Ji); Sy CHU SC Jewn show Jiho Kalf (Ji);</pre> | 🎋 Debug 🔀 🏪 Project Explorer 🛛 🗖 🗖 | 🖻 main.c 🛛 🕩 platform.h 💿 0x2000136e | | | (X)= Va X 💁 Br 생약 Ex | 🛋 M 🛛 🖓 P | - 0 |
| Writable Smart Inset 41:11:1655 | <pre>v Copy co</pre> | <pre>2</pre> | ce setting the prosole and peri ADOR, PMP_RESCALE, PM_PERIOD - CorePAN output 1. [1, duty_cycle); CorePAN output 1. . PMP_1, duty_cycle); utable: CobuggerConsole CorePAN output 1. . CorePAN output 1. . PMP_1, duty_cycle); | d values.) ; Monitor 🔹 💥 | Vor Vor <th>my XX</th> <th></th> | my XX | |
| | | | Writable Smart Insert | 41 : 1 : 1655 | | | 5 |

Figure 84 - SoftConsole Debug perspective

Running the pwm_slow_blink Application

39. Start the Cortex-M3 software application by clicking **Run > Resume** from the SoftConsole menu. The LEDs will appear as shown in the table below.

| LED | State | |
|----------------|----------------------|--|
| LED D1 | On | |
| LED D2 | Blinking (PWM), | |
| | gradually get dimmer | |
| | and brighter quickly | |
| LED D3-D9, D10 | Off | |

Table 2 – SMF2000 board LED states

- 40. Suspend the software application by clicking **Run > Suspend** from the SoftConsole menu or by clicking the Suspend icon () on the SoftConsole Toolbar.
- 41. Open the Registers view (Window > Show View > Registers).
- 42. Select the **Registers** view on the lower window pane to view the value of the Cortex-M3 internal registers as shown in the figure below. Your values may differ. If the Registers view is not visible, open it by selecting **Window > Show View > Registers**.

| 🖻 Console 🕮 Registers 🛛 🖹 Problems 📀 Executab | les 📓 Debugger Console 🔋 Memory | |
|---|---------------------------------|---|
| Name V III General Registers | Value | Description General Purpose and FPU Register Group |
| 1010 rO | 0x5000008 | |
| 1010 r1 | 0x1 | |
| 1010 r2 | 0x270f | |
| 1000 r3 | 0xf0f | |
| ¹⁰¹⁰ r4 | 0x0 | |
| 1111 r5 | 0x0 | |
| 1111 r6 | 0x11 | |
| 1010 0101 r7 | 0x2000ffe0 | |
| 3339 r8 | 0x0 | |
| 1111 r9 | 0x0 | |
| 1000 r10 | 0x0 | |
| 1889 r11 | 0x0 | |
| 388 r12 | 0x20e2a368 | ¥ |
| < | | > |

43. Choose **Window > Show View > Disassembly** to display the assembly level instructions. The Disassembly window will open on the right side in the middle of the Debug perspective as shown in the figure below.

Figure 86 – SoftConsole Disassembly view

44. Scroll in main.c and locate PWM_PERIOD on line 24.

| 虎 ma | in.c 🕱 | | |
|------|--|-----|---|
| 2⊕ | * (c) Copyright 2015 Microsemi SoC Products Group. All rights reserved. | | |
| 12 | <pre>#include "platform.h"</pre> | | |
| 13 | <pre>#include "core_pwm.h"</pre> | | - |
| 14 | | | - |
| 15⊝ | /************************************** | l | |
| 16 | * Delay count used to time the delay between duty cycle updates. | | |
| 17 | *************************************** | | |
| 18 | #define DELAY_COUNT 10000 | | |
| 19 | | | |
| 20⊝ | /************************************** | | |
| 21 | PWM prescale and period configuration values. | | |
| 22 | | | |
| 23 | Hade General Discourse and the second s | | |
| 24 | #define PWM_PERIOD 100 | | |
| 25 | /************************************** | | |
| 27 | / * CorePWM instance data | | |
| 28 | *************************************** | | |
| 29 | pwm instance t the pwm: | | |
| 30 | rr, | | - |
| | 4 | - F | |
| | • | P | |

Figure 87 - PWM_PERIOD in main.c

- 45. Changing the value of PWM_PERIOD on line 24 will change the toggle rate of the LED. Try changing the value then save the file (**File > Save**). A larger value for the period will make the LED toggle more slowly.
- 46. Build the project by selecting **Project > Build All**. Select the Problems View and confirm that there are no errors.

47. Select SF2_GNU_SC4_pwm_slow_blink.elf under the Debug tab in the upper left corner of the SoftConsole Debug Perspective. Right-click and choose **Terminate and Relaunch** to stop the debugger and download the new application.

| 🗱 Debug 🖾 Project Explorer | 🎉 🏟 🔻 🗖 🗖 | 🖻 main.c 🛛 🖪 platform.h | |
|--|--|-------------------------|---|
| ✓ | oug [GDB OpenOCD Debugging] | 18 #define DELAY_COUNT | 10000 |
| [®] SF2_GNU_SC4_pwm_slow_blink.e [®] Thread #1 (Suspended : Sign [®] delay() at main c²⁹ 0y200 [®] | Copy Stack Find | 19 Ctrl+C Ctrl+F | ************************************** |
| ■ main() at main.c:63 0x200 | 🗟 Drop To Frame | | ************************************** |
| 📕 openocd.exe | 3. Step Into | F5 | 800 |
| 📕 arm-none-eabi-gdb | 👁 Step Over | F6 | |
| | _ 요 Step Return | F7 | *************************************** |
| | i⇒ Instruction Stepping Mode | | data. *********************************** |
| | Ise Step Filters | | pwm; |
| | Resume Without Signal Resume Suspend | F8 | ************************************** |
| | Terminate | Ctrl+F2 | *************************************** |
| | Terminate and Relaunch | | |
| | ✔ Restart № Disconnect ₩ Connect | | Problems O Everytables Debugger Console O Memon |
| | Ø [♦] Debug New Executable | | 4 nwm slow hlink] |
| | Remove All Terminated Relaunch Edit SF2_GNU_SC4_pwm_slop Terminate and Remove Terminate/Disconnect All Properties | ow_blink Debug | <pre>dec hex filename</pre> |

Figure 88 – Terminating and re-launching the debugger

48. Run the application by clicking **Run > Resume** from the SoftConsole menu. The LED will toggle at a different rate.

Driving two LEDs

The application in the sample project only drives one LED, but CorePWM was configured to drive two outputs. Follow the steps below to modify the application to drive two LEDs.

- 49. Terminate the software application by clicking **Run > Terminate** from the SoftConsole menu.
- 50. Scroll in main to the section with the comment "* Set the initial duty cycle for CorePWM output 1." on line 50. This line configures one CorePWM output.
- 51. Copy the line "PWM_set_duty_cycle(&the_pwm, PWM_1, duty_cycle);"and paste it below the existing line. Modify the line to drive PWM_2.
- 52. Scroll to the section with the comment "* Update the duty cycle for CorePWM output 1." on line 58.
- 53. Copy the line "PWM_set_duty_cycle(&the_pwm, PWM_1, duty_cycle);"and paste it below the existing line. Modify the line to drive PWM_2. Save the file after making the edits above.

Figure 89 - Modified code to drive two PWM outputs

- 54. Repeat the previous steps to build the project. Confirm that there are no errors in the Problems View.
- 55. Re-launch the debugger (Run > Debug History > SF2_GNU_SC4_pwn_slow_blink Debug).
- 56. Run the application (Run > Resume) and observe the LEDs. Both LED D2 and LED D3 should toggle at the same rate. When finished, terminate the application by selecting SF2_GNU_SC4_pwm_slow_blink.elf under the Debug view, then right-clicking and selecting Terminate and Remove to stop the debugger. Click Yes when prompted about Terminating.

| 🎄 Debug 🖾 | | | 🍇 😹 | i> 🕆 🗖 🗖 |
|--|--------|---------------------------------------|------------------|---------------------|
| ▲ C SF2_GNU_SC4_pwm_slow_b | link D | ebug [GDB OpenOCD Debugging] | | |
| ▲ SF2_GNU_SC4_pwm_slo | | Copy Stack Find | Ctrl+C Ctrl+F | |
| arm-none-eabi-gdb | P | Drop To Frame | | |
| le main.c 🕅 | Ъ. | Step Into | F5 | |
| 45 /*************** | P | Step Over | F6 | ***** |
| 46 * Initialize the 47 **************** | .r. | Step Return | F7 | .ues. ****/ |
| 48 PWM_init(&the_pwm | i⇒ | Instruction Stepping Mode | | |
| 49 50⊝ /*************** | P | Use Step Filters | | ***** |
| 51 * Set the initial 52 *************** | | Resume Without Signal | | ****/ |
| 53 PWM_set_duty_cycle | | Resume | F8 | |
| 54 PWM_set_duty_cycle | | Suspend | | re a second LED |
| 56 while (1) | | Terminate | Ctrl+F2 | |
| 57 { 58 /*********** | • | Terminate and Relaunch | | ***** |
| 59 * Update the | 也 | Restart | | |
| 60 ********************** | 14 | Disconnect | | ****/ |
| 62 /********** | 84 | Connect | | **** |
| 63 * Update the 64 ********** | 0° | Debug New Executable | | econd LED *****/ |
| 65 PWM_set_duty_o | × | Remove All Terminated | | |
| 4 | Q | Relaunch | | |
| 🗖 Canaala 🕅 🖨 Taalaa 🔍 Daa | С | Edit SF2_GNU_SC4_pwm_slow_blink Debug | | |
| SE2 CMU SC4 nume claux blink Dab | • | Terminate and Remove | | |
| (22) d t 2 mole ((4) | 6 | Terminate/Disconnect All | | |
| (32) dwt_2_mask (/4) (33) dwt_2_function (/32) (34) dwt 3 comp (/32) | | Properties | | |
| F | igur | e 90 – Terminating the application | | |

- 57. Close the Debug perspective (Window > Perspective > Close Perspective).
- 58. Close the SF2_GNU_SC4_pwm_slow_blink project by selecting the project name in the Project Explorer then rightclicking and selecting **Close Project**.

Step 8 - Running the RTC_time application (time permitting)

In this step, you will open the SF2_GNU_SC4_RTC_time sample project, import the firmware configuration files and run the application. This sample project uses the SmartFusion2 RTC's calendar function.

- 1. Open the SF2_GNU_SC4_RTC_time project by selecting the project name in the Project Explorer then right-clicking and selecting **Open Project**.
- 2. Repeat the steps on pages 46 to 47 to import the firmware drivers and configuration files into the project.
- 3. Double click main.c in the Project Explorer window to open the file in the SoftConsole C/C++ editor. Scroll through the file to become familiar with it. The comments at the top of the file describe what the program does.
- 4. Select SF2_GNU_SC4_RTC_time in the SoftConsole Project Explorer. Build the project by selecting **Project > Build All** or by clicking the Build All icon.
- 5. Confirm that no error messages appear in the SoftConsole Problems view.

Determining the COM port setting

This design requires a terminal emulator. The COM port assignment must be determined to configure the terminal emulator.

- 6. Open the Windows Device Manager and expand the **Ports (COM & LPT)** section.
 - One port will appear as "FlashPro 5 Port". Record the COM port number below.

USB Serial Port: COM_____

| 🛃 Device Manager | - | × |
|--|---|---|
| <u>File</u> <u>Action</u> <u>View</u> <u>H</u> elp | | |
| 🗢 🔿 🖬 🔢 💷 | | |
| > 🎽 Firmware | | ^ |
| > 🛺 Human Interface Devices | | |
| > 🦏 IDE ATA/ATAPI controllers | | |
| > 🚡 Imaging devices | | |
| > 🔤 Keyboards | | |
| > 🥅 Memory technology devices | | |
| > III Mice and other pointing devices | | |
| > 🛄 Monitors | | |
| > 🚽 Network adapters | | |
| Ports (COM & LPT) | | |
| FlashPro5 Port (COM5) | | |
| > 🚍 Print queues | | |
| > 🚍 Printers | | |
| Disassan | | * |
| | | |

Figure 91 - Windows Device Manager showing COM port

Creating a debug launch configuration

- 7. Select SF2_GNU_SC4_RTC_time in the SoftConsole Project Explorer View.
- 8. Select **Run > Debug Configurations...** from the SoftConsole menu. The Debug Configurations dialog will open.
- 9. Select GDB OpenOCD Debugging in the Debug Configurations dialog box then right-click and select **New Configuration** to create a new debug launch configuration for the Debug build configuration.
- 10. Ensure that the C/C++ Application field on the Main tab of the Debug Configurations dialog box contains Debug\SF2_GNU_SC4_RTC_time.elf

| sc Debug Configurations | | | — D |
|--|--------------------------------------|--|-----------------|
| reate, manage, and run configura | tions | | Ś |
| type filter text | Name: SF2_GNU_SC4_RTC_time Debu | | |
| C GDB OpenOCD Debugging C SF2 GNU SC4 RTC time Det | Project: | | |
| Launch Group | SF2_GNU_SC4_RTC_time | | <u>B</u> rowse |
| Robot | C/C++ Application: | | |
| S UltraDevelop Agent | Debug\SF2_GNU_SC4_RTC_time.elf | | |
| S UltraDevelop OCD | | <u>V</u> ariables Searc <u>h</u> Project | B <u>r</u> owse |
| 🄏 UltraDevelop Remote Target | Build (if required) before launching | | |
| | Build Configuration: Select Autom | atically | \sim |
| | O Enable auto build | O Disable auto build | |
| | Use workspace settings | Configure Workspace Settings | |
| | | | |
| | | | |
| < > | | Revert | Apply |
| ilter matched 8 of 15 items | | | 1.464.7 |
| ? | | Debua | Close |
| 0 | | <u>_</u> | |

11. Select the Debugger tab of the Debug Configurations dialog box. Confirm that the Config options field appears as shown in the figure below.

| sc Debug Configurations | | - 0 | × |
|---|--|---------------|----|
| Create, manage, and run configurations | | 5 | Š. |
| Image: Second | Name SF2_GNU_SC4_RTC_time Debug Main | e Variables | |
| Filter matched 8 of 15 items | Reyert | Apply | Ĺ |
| ? | <u>D</u> ebu | ıg Clo | se |

Figure 93 - SoftConsole device and board file settings for the SmartFusion2 SMF2000 kit

12. Select the Startup tab. Confirm that the settings match the figure below.

| SC Debug Configurations | - 🗆 X |
|--|--|
| Create, manage, and run configurations | |
| | |
| 🗋 🖻 💫 🗈 🗙 🖻 🍸 🗸 | Name: SF2_GNU_SC4_RTC_time Debug |
| type filter text C GDB OpenOCD Debugging C SF2_GNU_SC4_RTC_time Debug Robot Robot Remote UltraDevelop Agent UltraDevelop Remote Target | Main Sk Debugger Startup Source ⊆ommon SvD Path Initialization Commands Initialization Commands Initialization Commands Initial Reset. Type: init Initial Reset. Type: init Initialization Commands ✓ Enable ARM semihosting Initial Reset. Type: init Initial Reset. Type: init ✓ Enable ARM semihosting Initial Reset. Type: init Initial Reset. Type: init ✓ Load Symbols and Executable ✓ Ioad symbols Ioad symbols ④ Use project binary: SF2_GNU_SC4_RTC_time.elf Workspace ✓ Load executable Iso offset (hex): Iso offset (hex): ✓ Load executable Iso offset (hex): Iso offset (hex): ✓ Load executable offset (hex): Iso offset (hex): Iso offset (hex): ✓ Load executable offset (hex): Iso offset (hex): Iso offset (hex): Runtime Options Ibebug in RAM Iso offset (hex): Iso offset (hex): Runtime Options Ibebug in RAM Ibebug in RAM Iso offset (hex): Pre-run/Restart Commands Ipre-run/Restart reset Type: halt (slways executed at Restart) |
| Filter matched 8 of 15 items | Set program counter at (hex): Set breakpoint at: Set breakpoint at: Continue Restore defaulter |
| ? | Debug Close |

Figure 94 - Startup tab settings

13. Click **Apply** to save if changes were made.

Launching the Debugger

Click Debug in the Create, manage and run configurations dialog box. If you already closed the dialog box, select
 Run > Debug from the SoftConsole menu to launch the SoftConsole debugger. You can also launch the debugger by

clicking the debug icon (*) in the SoftConsole tool bar.

15. Click **Switch** in the **Confirm Perspective Switch** dialog box.

Configuring a Serial Terminal View

SoftConsole 6.4 includes a built-in serial terminal view, which eliminates the need to run a separate serial terminal emulator when connecting to a target board using a UART.

16. Select Window > Show View > Terminal to open a serial terminal. The Terminals view will be visible.

- 17. Click the **Open a Terminal** button (highlighted in the figure above) to configure the serial terminal.
- 18. Enter the following in the Launch Terminal dialog box then click **OK**:
 - Choose terminal: select Serial Terminal from the pull-down menu
 - Port: Enter the COM port number recorded on page 63.
 - Baud rate: 115200
 - Data Bits: 8
 - Parity: None
 - Stop Bits: 1
 - Flow Control: None
 - Encoding: Default (ISO-8859-1)

The baud rate, data bits, parity, stop bits and flow control settings are based on the MSS_UART_init function in main().

| Choose term | inal: Serial Terminal | | ~ |
|--------------|-----------------------|-------|--------|
| Settings | | | |
| Serial port: | COM5 | | ~ |
| Baud rate: | 115200 | | ~ |
| Data size: | 8 | | ~ |
| Parity: | None | | \sim |
| Stop bits: | 1 | | \sim |
| Encoding: | Default (ISO-8859-1) | | \sim |
| | | | |
| ? | OK | Cance | |

Figure 96 - Serial terminal settings - COM port setting will vary

19. Run the software application by clicking the Resume icon (IP>) or by clicking **Run > Resume** from the SoftConsole menu. The Terminals view will display a message as shown in the figure below.

| 💷 Console 🖹 Problems 📀 Executables 🧬 Terminal 🛛 | |
|--|--|
| ■ COM5 ☎ | |
| * | × |
| ************************************** | me Example ************************************ |
| This example project displays the time read f A simple command line interface allows the fo - Set the RTC time by pressing "t" - Set the RTC date by pressing "d" | from the SmartFusion2 RTC. ollowing operations: |
| Saturday January 1 2000 (week 1) 00: | :00:12 |
| Figur | re 97 - SoftConsole Terminals view with message |

20. Type d to set the date.

| 🗟 Console 🖹 Problems 📀 Executables 🚚 Terminal 🛛 | 🖻 🕅 🖣 🛱 🚮 🗎 🕯 🥵 – 🗖 |
|--|---------------------|
| | |
| This example project displays the time read from the SmartFusion2 RTC. A simple command line interface allows the following operations: | |
| - Set the RTC time by pressing "t" - Set the RTC date by pressing "d" | |
| Saturday January 1 2000 (week 1) 00:01:42 | |
| Change date. Day (1 to 31): 4 | |
| Year (0 to 254): 19 | ~ |
| Figure 98 - Changing the date | |

21. Type t to set the time.

| 🖻 Console 🖹 Problems 💿 Executables 🔊 Terminal 🛛 | 😑 🕅 İn 🔓 🖬 🗈 🗈 🧳 🗝 🗖 |
|---|--------------------------|
| E COM5 🛙 | |
| Saturday January 1 2000 (week 1) 00:01:42 | ^ |
| Change date. Day (1 to 31): 4 Month (1 to 12): 3 Year (0 to 254): 19 | |
| Monday March 4 2019 (week 11) 00:03:12 | |
| Change time. Hours: 21 Minutes: 13 Seconds: 00 | ~ |
| Figure 00 Setting the time | |

Figure 99 - Setting the time

22. After becoming familiar with the Debug perspective and the debug features, terminate the debugger by clicking the Terminate icon (\blacksquare) or by selecting **Run > Terminate** from the SoftConsole menu.

Step 9 - Debugging code running from the SmartFusion2 eNVM

The application can also execute from the SmartFusion2 eNVM. The Cortex-M3 has six hardware breakpoints for debugging from eNVM. This section describes the steps to execute code from the SmartFusion2 eNVM.

Building the project with the Release build configuration

- Select the SoftConsole C/C++ Perspective by clicking the C/C++ button (¹) or by selecting Window > Perspective > Open Perspective > C/C++ from the SoftConsole menu.
- 2. The sample project Release build configuration settings use the *debug-in-microsemi-smartfusion2-envm.ld* linker script to build an executable that will run in the SmartFusion2 eNVM.
- Select SF2_GNU_SC4_RTC_time in the SoftConsole Project Explorer then right-click and select Build Configurations
 Set Active > Release to select the Release configuration as the active build configuration.
- 4. Build the project build by selecting **Project > Build All**. Confirm that there are no errors listed in the Problems view.
- 5. A folder named Release containing the executable for the Release build configuration will be visible in the Project Explorer.
- 6. Select SF2_GNU_SC4_RTC_time.map in the Release folder, press right mousebutton and choose "Open With" "Text Editor" to open the file in the SoftConsole editor. Scroll down to the Memory Configuration section. The section should appear as shown in the figure below.

| SC SoftConsole - SF2_GNU_SC4_RTC_time/Release/SF2_GNU_SC4_RTC_time.map - Microsemi SoftConsole v6.0.0.116 - | \times | |
|---|----------|----|
| ille <u>E</u> dit <u>S</u> ource Refactor <u>N</u> avigate Se <u>a</u> rch <u>P</u> roject <u>R</u> un <u>W</u> indow <u>H</u> elp | | |
| ¹ ▼ | 🔤 🛪 | ¢۶ |
| [™] Main.c [™] SF2_GNU_SC4_RTC_time.map % [™] [™] SF2_GNU_SC4_RTC_time.map % [™] [™] [™] SF2_GNU_SC4_RTC_time.map % [™] | 2 B | ₽ |
| 128/.bs 0x00000000000000000000000000000000000 | , , . | |
| 1294 1295 Name Origin Length Attributes 1296 rom 0x0000000000000 0x0000000000000000 xr 1297 romMirror 0x00000000000000 0x000000000000000 xr 1298 ram 0x0000000000000 0x00000000000000000000000000000000000 | - | |
| 1301Linker script and memory map 1302 1303LOAD c:/microsemi/softconsole_v6.0/arm-none-eabi-gcc/bin//lib/gcc/arm-none-eabi/7.3.1/thumb/v7-m/crti.o 1304LOAD c:/microsemi/softconsole_v6.0/arm-none-eabi-gcc/bin//lib/gcc/arm-none-eabi/7.3.1/thumb/v7-m/crtbegin.o 1305LOAD c:/microsemi/softconsole_v6.0/arm-none-eabi-gcc/bin//lib/gcc/arm-none-eabi/7.3.1////arm-none-eabi/lib/thumb/v7-m/crt0.o <pre></pre> | ~ | |
| Writable Insert 1:1 | 4 | - |
| Figure 100, SE2, CNUL SC4, BTC, time man for the Balance build configuration | | |

- Figure 100 SF2_GNU_SC4_RTC_time.map for the Release build configuration
- 7. Notice that the memory configuration for the Release build includes a rom section at 0x60000000, which is the address of the SmartFusion2 eNVM.

Creating the Release debug launch configuration

Create a debug launch configuration for the Release build.

- 8. Select SF2_GNU_SC4_RTC_time in the SoftConsole Project Explorer View then select **Run > Debug Configurations...** from the SoftConsole menu. The Debug Configurations dialog will open.
- 9. Select GDB OpenOCD Debugging in the Debug Configurations dialog box then right-click and select **New Configuration** to create a new debug launch configuration for the Release build configuration.
- 10. Click the Search Project button under the C/C++ Application field on the Main tab of the Debug Configurations dialog box.

| Name: SF2_GNU_SC4_RTC_time Release | |
|---|-----------------|
| 📄 Main 🕸 Debugger 🕨 Startup 🦆 Source 🔲 <u>C</u> ommon | |
| Project: | |
| SF2_GNU_SC4_RTC_time | <u>B</u> rowse |
| C/C++ Application: | |
| | |
| <u>V</u> ariables Searc <u>h</u> Project | B <u>r</u> owse |

Figure 101 - Search Project button

11. The Program Selection dialog box will open.

Select armle - / SF2_GNU_SC4_RTC_time /Release/SC4_project.elf then click **OK**.

| SC Program Selection |
|---|
| Choose a <u>p</u> rogram to run: |
| |
| Binaries: |
| SF2_GNU_SC4_RTC_time.elf |
| |
| Qualifier: |
| armle - /SF2_GNU_SC4_RTC_time/Debug/SF2_GNU_SC4_RTC_time.elf armle - /SF2_GNU_SC4_RTC_time/Release/SF2_GNU_SC4_RTC_time.elf |
| OK Cancel |

Figure 102 - Selecting the Release configuration application program

12. Select the Debugger tab of the Debug Configurations dialog box. Confirm that the Config options and GDB Client Setup fields contain the settings described previously.

| sc Debug Configurations | | | | - 0 | |
|---|---|--|--------|-----------|----|
| reate, manage, and run configurations | | | | Ċ | Ś |
| Provide a state of the second state of th | Name: SF2_GNU_SC4 | RTC_time Release er Startup jer Source Stoppoord_path/Stoppoord_executable; CiMicrochip/SoftConsole-v6.4teclipse//openocd/bin/openocd.exe to change it use the global or workspace preferences pages or the project properties page) 3333 4444 | Browse | Variables |] |
| | Config options: | command "set DEVICE M25010"file board/microsemi-cortex-m3.cfg for OpenOCD Allocate console for the teinet connection | n | ^ ~ | |
| | GDB Client Setup | n | | | |
| | Executable name: Actual executable: Other options: Commands: | S(cross_prefix)gdbS(cross_suffix) arm-none-eabi-gdb set mem inaccessible-by-default off | Browse | Variables |] |
| ilter matched 9 of 16 items | | | Revert | Apply | ! |
| 3 | | | Debug | Clo | se |

Figure 103 - Config options for the Release Debug launch configuration for the SmartFusion2 SMF2000 kit

- 13. Select the Startup tab. Confirm the setting match those shown on page 65.
- 14. Click **Apply** to save the changes.
- 15. Click **Debug** to launch the Debugger. Click **Switch** in the **Confirm Perspective Switch** dialog box. The SoftConsole Debug perspective will open and the code will be programmed to the SmartFusion2 eNVM. Messages will appear in the Console view while the code is being downloaded. When finished the program will be suspended at the first line of main().

Figure 104 - Console view while downloading to the SmartFusion2 eNVM

- 16. Select the Terminals view in the Debug Perspective. Right click in the Terminal and select **Clear Terminal** to remove any existing text. If necessary, open the Terminals view (**Window > Show View > Terminal**).
- 17. Click the Resume icon to run the application. Note that the application runs the same as previously when it was executing from the SmartFusion2 eSRAM memory.
- Try setting breakpoints while executing the code. When finished, terminate the debugger and close the Debug perspective (Window > Close Perspective). Do not close SoftConsole.
- The C/C++ Perspective will be open. Open the Terminal view (Window > Show View > Terminal) and configure as described earlier in the document.
- 20. Reset the SmartFusion2 SMF2000 kit by pressing and releasing SW3.
- 21. Observe the code execution in the Terminals view. The code is running without the debugger because it was programmed into the SmartFusion2 eNVM.

Note that the serial terminal can be used to display output even when a program is not being debugged - e.g. a program running from eNVM on power on reset that uses UART output.

Figure 105 - Terminals view in C/C++ perspective

This demonstrates how to generate sample SmartFusion2 firmware projects and debug them with SoftConsole 6.4.

End of SmartFusion2 lab


List of Figures

| Figure 1 - SmartFusion2 Block Diagram | 5 |
|--|----|
| Figure 2 - Libero SoC Project Manager | 6 |
| Figure 3 - IP Catalog indicating new cores are available | 7 |
| Figure 4 - Executing the project creation script | 8 |
| Figure 5 - Script execution report | 8 |
| Figure 6 - Libero SoC Design Flow tab | 9 |
| Figure 7 - Entering a name for System Builder | 9 |
| Figure 8 – SmartFusion2 System Builder Device Features page | 10 |
| Figure 9 - System Builder Device Features after selections | 11 |
| Figure 10 - System Builder Memories page | 11 |
| Figure 11 - Selecting the hex file for the data storage client | 12 |
| Figure 12 - Data Storage Client dialog box after importing the hex file | 12 |
| Figure 13 - System Builder Memories page after adding the data storage client | 13 |
| Figure 14 - System Builder Peripherals page | 14 |
| Figure 15 - MM_UART_0 configurator | 14 |
| Figure 16 - SmartFusion2 GPIO Configuration | 15 |
| Figure 17 - System Builder Peripherals page after adding CorePWM | 16 |
| Figure 18 - corepwm_0 configuration | 17 |
| Figure 19 – Configuring the MSS clocks | 18 |
| Figure 20 - System Builder Microcontroller options page | 19 |
| Figure 21 - System Builder Microcontroller options page after selecting the RTC clock source | 20 |
| Figure 22 - System Builder SECDED options page | 21 |
| Figure 23 - CorePWM TACHINT interrupt | 22 |
| Figure 24 - Fabric memory map | 23 |
| Figure 25 - Message tab after successfully generating the MSS design | 23 |
| Figure 26 – SF2_MSS_sys_sb_0 component | 24 |
| Figure 27 – Connecting FAB_RESET_N port to VCC | 24 |
| Figure 28 – SF2_MSS_sb_0 component after making pin connections | 25 |
| Figure 29– Execute Design Rules Check | 26 |
| Figure 30 - Libero SoC Message window after generating the design | 26 |
| Figure 31 - Opening the Constraint Editor | 27 |
| Figure 32 - Libero SoC Enhanced Constraint Manager | 27 |
| Figure 33 - Importing the I/O constraint file | 28 |
| Figure 34 - Selecting and saving the I/O constraint file for layout | 28 |
| Figure 35 - Deriving Timing Constraints | 29 |
| Figure 36 - Message Window | 29 |
| Figure 37 - Derived Timing Constraints | 29 |
| Figure 38 - Generate Bitstream | 30 |
| Figure 39 - Successful completion of design implementation | 31 |
| Figure 40 - Reports tab after implementing the design | 32 |
| Figure 41 - Device Manager | 32 |
| Figure 42 - Run program action | 33 |
| Figure 43 - Windows Device Manager showing COM port | 33 |
| | |



SmartFusion2 ARM Cortex-M3 Lab Guide

| Figure 44 - Configuring Firmware cores for the design | 34 |
|---|----|
| Figure 45 - DESIGN FIRMWARE tab. | |
| Figure 46 - Downloading missing firmware cores | 35 |
| Figure 47 - Selecting SmartEusion? CMSIS version 2 3 105 | 35 |
| Figure 48 - Generating the CorePWM sample project | 25 |
| Figure 49 – CorePWM slow blink sample project ontions | 55 |
| Figure 50 - GPIO Simple Blink project files | 36 |
| Figure 51 - Generating the RTC Driver sample project | 36 |
| Figure 52 – PTC Time sample project location | 50 |
| Figure 52 – RTC time project location | 37 |
| Figure 54 Exporting the firmware configuration files | 57 |
| Figure 54 - Exporting the firmware comparation mes | 0C |
| Figure 55 - Export Firmware options | 0C |
| Figure 55 - Firmware driver location | 39 |
| Figure 57 - Sample projects on Libero Soc Files tab | 39 |
| Figure 58 – Data Sneet for the design | 40 |
| Figure 59 - SoftConsole 6.4 Workspace Launcher (location snown may differ) | 41 |
| Figure 60 - Selecting the SoftConsole workspace | 41 |
| Figure 61 – Eclipse Launcher with workspace selected. | 42 |
| Figure 62 - SoftConsole v6.4 GUI | 42 |
| Figure 63 - SoftConsole Import dialog box | 43 |
| Figure 64 - Importing the sample projects into the workspace | 44 |
| Figure 65 - SoftConsole projects in the workspace | 45 |
| Figure 66 - Importing the firmware configuration files | 46 |
| Figure 67 - Importing the firmware configuration files into the project | 47 |
| Figure 68 - Question dialog box | 47 |
| Figure 69 - SF2_GNU_SC4_pwm_slow_blink after importing the firmware configuration files | 48 |
| Figure 70 - main.c in the SoftConsole editor | 48 |
| Figure 71 - platform.h in the SoftConsole editor | 49 |
| Figure 72 - Project Properties dialog | 50 |
| Figure 73 - Selecting all build configurations | 50 |
| Figure 74 - Project Include paths | 51 |
| Figure 75 - Confirming the Cross ARM C Compiler Other compiler flags setting | 52 |
| Figure 76 - Debug configuration linker script | 53 |
| Figure 77 - Release configuration linker script | 53 |
| Figure 78 - Selecting the build configuration | 54 |
| Figure 79 - Problems view after building the project | 54 |
| Figure 80 - Creating a new debug launch configuration | 55 |
| Figure 81 - Debug launch configuration | 55 |
| Figure 82 - SoftConsole v6.4 Debugger tab settings for the SmartFusion2 SMF2000 kit | 56 |
| Figure 83 - Startup tab settings | 57 |
| Figure 84 - SoftConsole Debug perspective | 58 |
| Figure 85 - Cortex-M3 registers | 58 |
| Figure 86 – SoftConsole Disassembly view | 59 |
| Figure 97 DWM DEDIOD in main c | 59 |
| rigule 07 - PWIVI PERIOD III IIIdiii.C | |



SmartFusion2 ARM Cortex-M3 Lab Guide

| Figure 88 – Terminating and re-launching the debugger | 60 |
|---|----|
| Figure 89 - Modified code to drive two PWM outputs | 61 |
| Figure 90 – Terminating the application | 62 |
| Figure 91 - Windows Device Manager showing COM port | 63 |
| Figure 92 - Debug launch configuration | 64 |
| Figure 93 - SoftConsole device and board file settings for the SmartFusion2 SMF2000 kit | 64 |
| Figure 94 - Startup tab settings | 65 |
| Figure 95 - Terminal view in the Debug perspective | |
| Figure 96 - Serial terminal settings - COM port setting will vary | 67 |
| Figure 97 - SoftConsole Terminals view with message | 67 |
| Figure 98 - Changing the date | 67 |
| Figure 99 - Setting the time | |
| Figure 100 - SF2_GNU_SC4_RTC_time.map for the Release build configuration | |
| Figure 101 - Search Project button | |
| Figure 102 - Selecting the Release configuration application program | 70 |
| Figure 103 - Config options for the Release Debug launch configuration for the SmartFusion2 SMF2000 kit | 71 |
| Figure 104 - Console view while downloading to the SmartFusion2 eNVM | 71 |
| Figure 105 - Terminals view in C/C++ perspective | 72 |