SoftConsole v4.0 and Libero SoC v11.7

TU0546 Tutorial

Contents

Figures

Tables

1 SoftConsole v4.0 and Libero SoC v11.7

1.1 Introduction

This tutorial describes how to implement an ARM® Cortex®-M3 design using Libero® System-on-Chip v11.7 and build a simple LED blink application using SoftConsole v4.0.

After completing this tutorial, you will be able to perform the following tasks:

- Create a Libero SoC project using System Builder
- Generate the programming file to program the SmartFusion®2 System-on-Chip (SoC) field programmable gate array (FPGA) device
- Creating a SoftConsole v4.0 project
- Compile application code
- Debug and run code using SoftConsole

1.2 Design Requirements

Table 1 • **Design Requirements**

Design Requirements	Description
Hardware Requirements	
SmartFusion2 Security Evaluation Kit*: • FlashPro4 or FlashPro5 programmer • 12 V adapter	Rev D or later
SmartFusion2 Advanced Development Kit*: • FlashPro4 or FlashPro5 programmer • 12 V adapter	Rev B or later
SmartFusion2 Starter Kit*: • FlashPro4 or FlashPro5 programmer • USB A to Mini-B cable	SmartFusion2-484-Starter-Kit (M2S010-FGG484)
Host PC or Laptop	Any 64-bit Windows Operating System
Software Requirements	
Libero SoC	v11.7
SoftConsole	v4.0
FlashPro programming software	v11.7
Note: * This tutorial is applicable for any one of the SmartFusion2 boards listed in this table.	

1.2.1 Associated Project Files

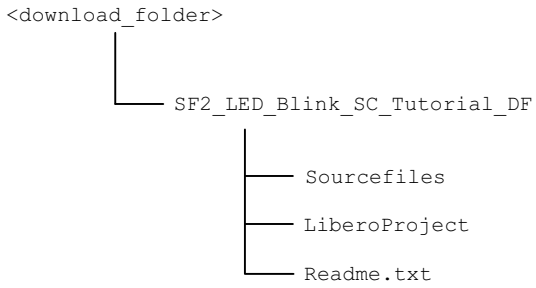
Download design files for this tutorial from the Microsemi website:
http://soc.microsemi.com/download/rsc/?f=m2s_tu0546_liberov11p7_df

The demo design files include:

- LiberoProject
- Sourcefiles
- Readme file

Figure 1 on page 7 shows the top-level structure of the design files. See the `Readme.txt` file provided in the design files for the complete directory structure.

Figure 1 • **Design Files Top-Level Structure**



1.3 Design Overview

This tutorial demonstrates a simple LED blinking application for SmartFusion2 device. Microcontroller subsystem (MSS) general-purpose input/output (GPIOs) are configured as outputs and connected to LEDs using fabric I/Os. This tutorial is applicable on one of the following SmartFusion2 boards:

- SmartFusion2 Security Evaluation Kit
- SmartFusion2 Advanced Development Kit
- SmartFusion2 Starter Kit (M2S010-FGG484)

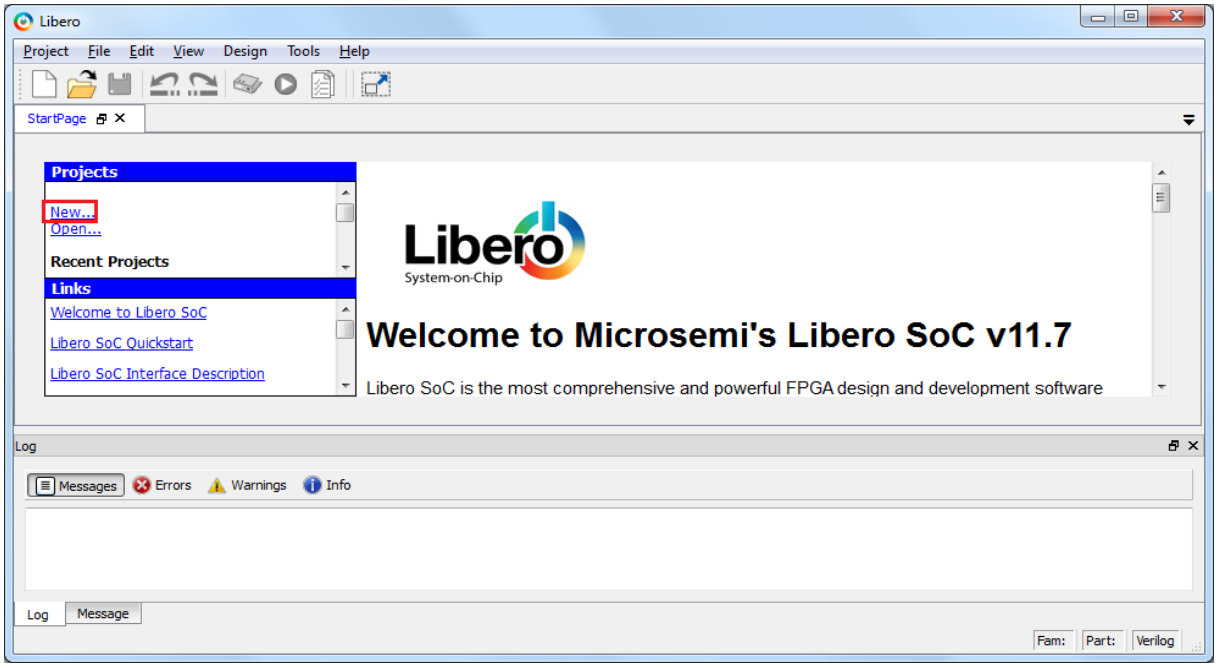
1.4 Step 1: Creating a Libero SoC Project

The following steps describe how to create a Libero SoC project:

Launching Libero SoC

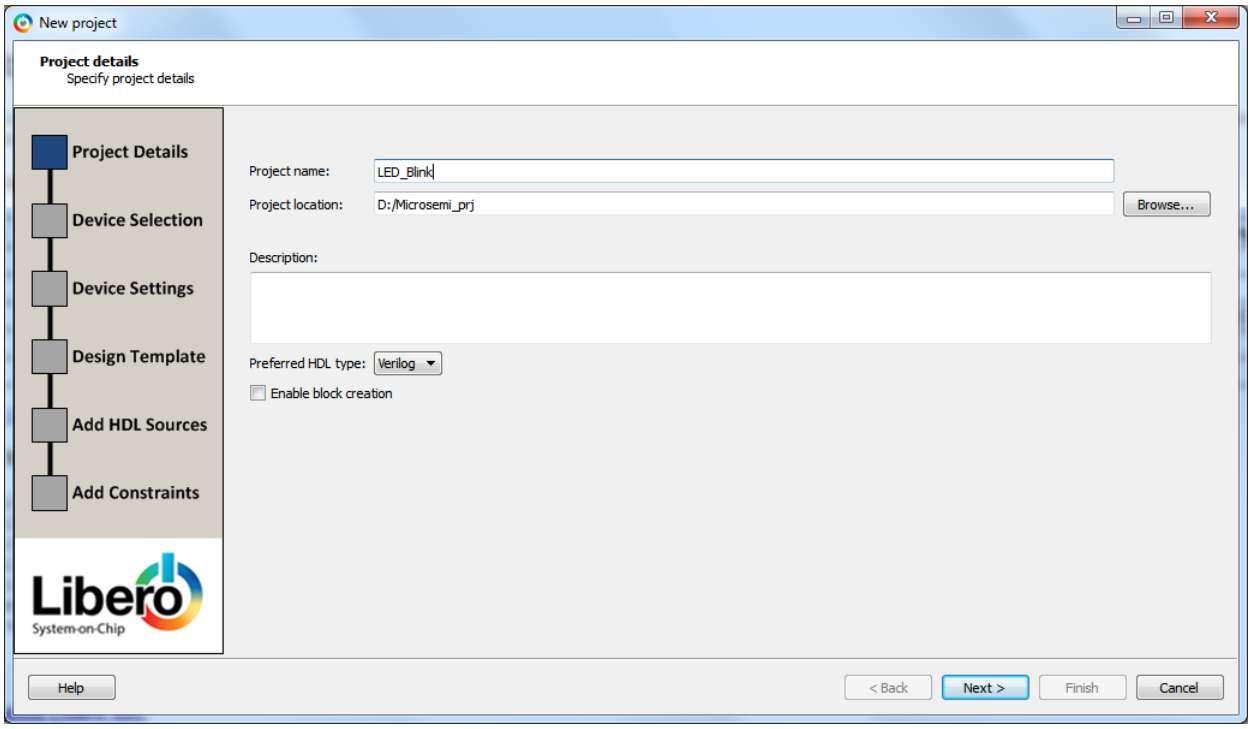
1. Click **Start > Programs > Microsemi > Libero SoC v11.7** or double-click the shortcut on desktop to open the Libero SoC v11.7 Project Manager.
2. Create a new project by selecting **New** on the **Start Page** tab (highlighted in Figure 2) or by clicking **Project > New Project** from the Libero SoC menu.

Figure 2 • **Libero SoC Project Manager**



3. In the **Project Details** window, enter the following information as shown in Figure 3.
   - **Project Name**: LED_Blink
   - **Project Location**: Select an appropriate location (for example, *D:/Microsemi_prj*)
   - **Preferred HDL type**: Verilog or VHDL
   - **Enable Block Creation**: Unchecked

Figure 3 • **Project Details Window**

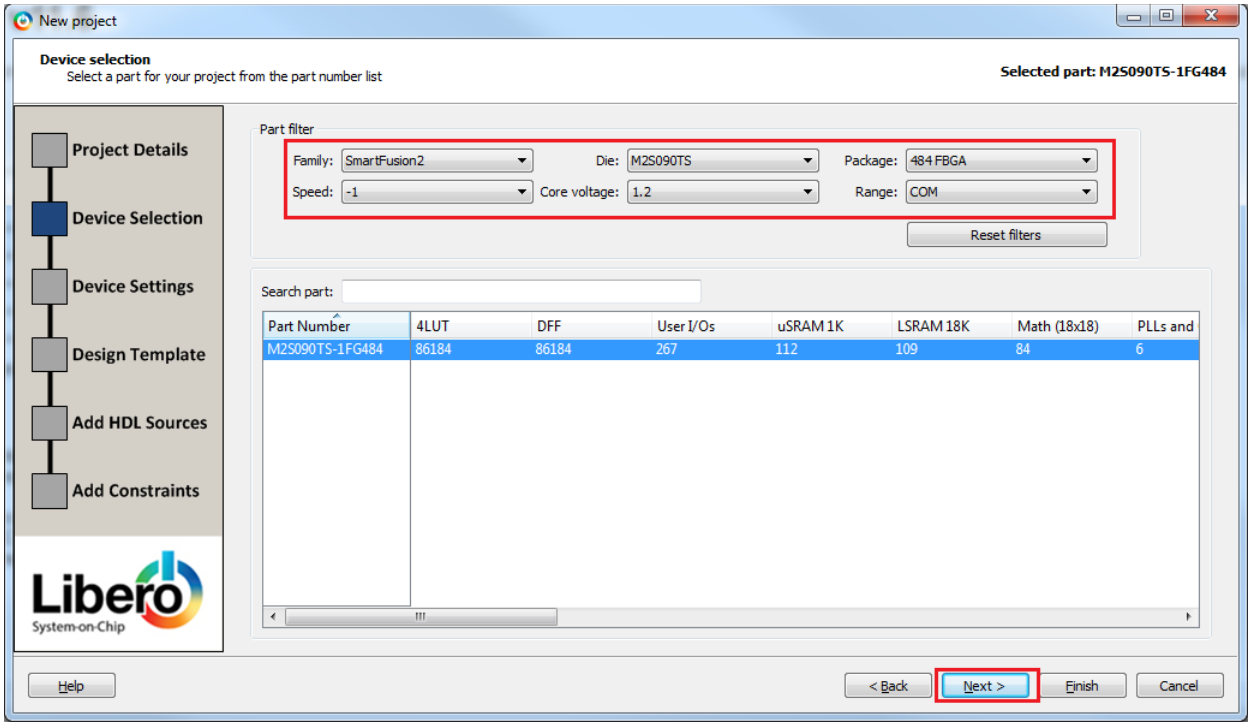


4. Click **Next**. In the **Device Selection** window, select the information as displayed in Figure 4 on page 9. In the Part Filter, select the values using the drop-down lists, as shown in Table 2.
   - **Family**: SmartFusion2

Table 2 • **SmartFusion2 Devices Selection**

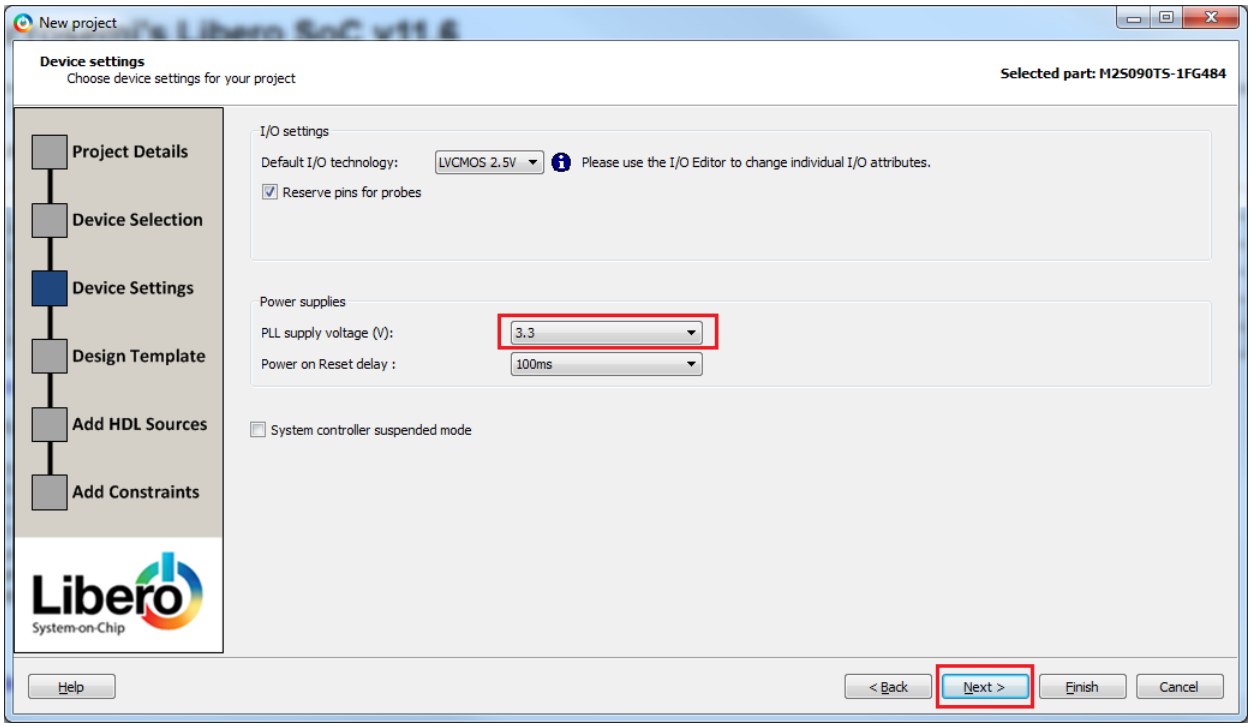
Board	Die	Package	Speed	Core Voltage	Range	PLL Supply Voltage
SmartFusion2 Security Evaluation Kit	M2S090TS	484 FBGA	-1	1.2	COM	3.3
SmartFusion2 Advanced Development Kit	M2S150T	1152 FC	-1	1.2	COM	3.3
SmartFusion2 Starter Kit	M2S010	484 FBGA	STD	1.2	COM	2.5

Figure 4 • **Device Selection Window**



5. Click **Next**. The Device settings window is displayed. Select PLL Supply Voltage (V), as shown in Figure 5.

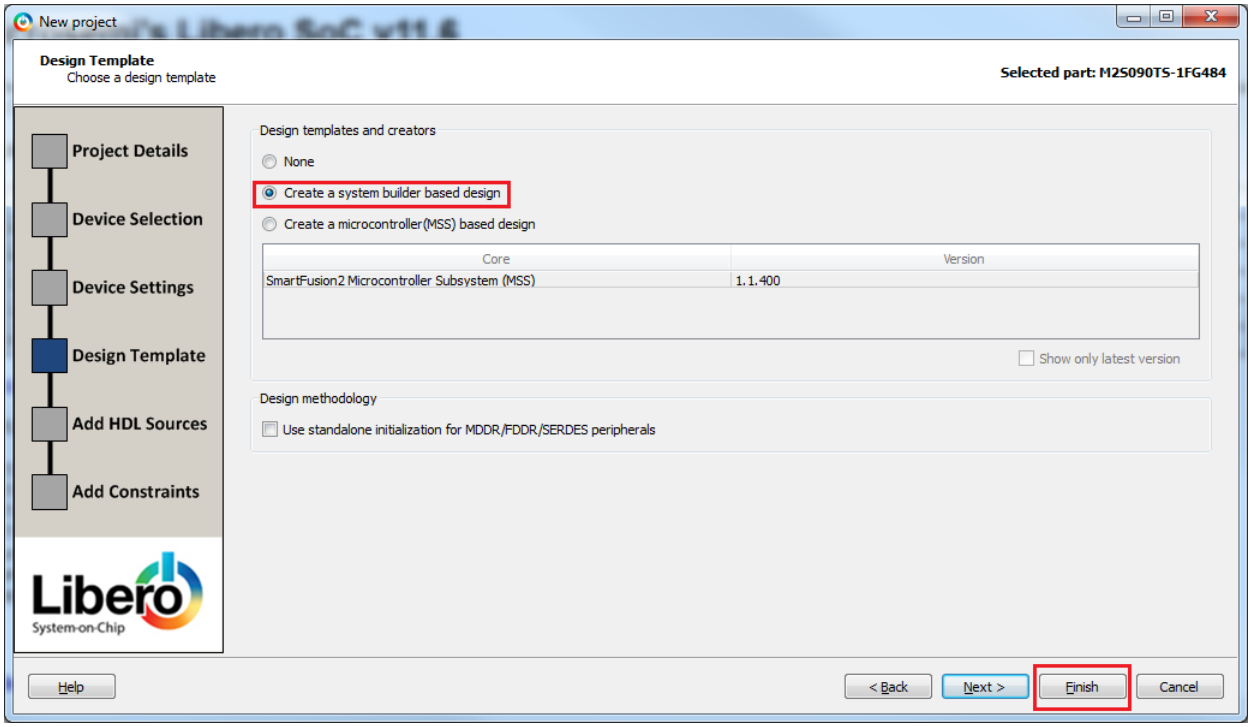
Figure 5 • **Device Settings**



See Table 2 on page 8 for specific board values.

6. Click **Next**. In the **Design Template** page, select **Create a system builder based design** check box under the **Design Templates and Creators** as shown in Figure 6.

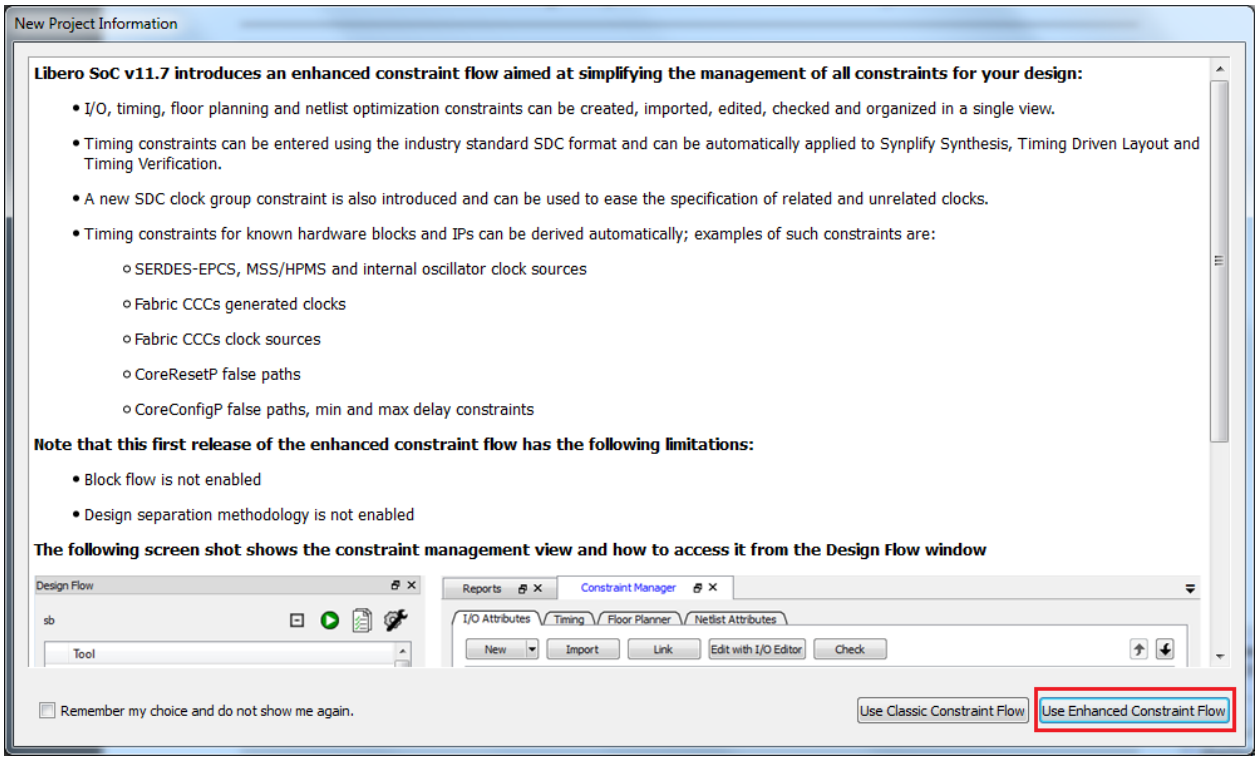
Figure 6 • **Design Template Window**



7. Click **Finish**. A **New Project Information** window is displayed, as shown in Figure 7 on page 11.

8. Click **Use Enhanced Constraint Flow**, as shown in Figure 7.

Figure 7 • **New Project Information Window**

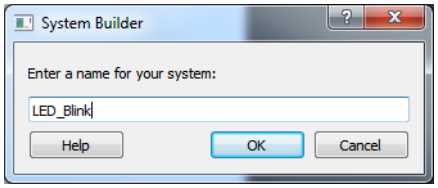


A System Builder dialog box is displayed, as shown in Figure 8.

A System Builder is a graphical design wizard. It creates a design based on high-level design specifications by taking the user through a set of high-level questions that define the intended system.

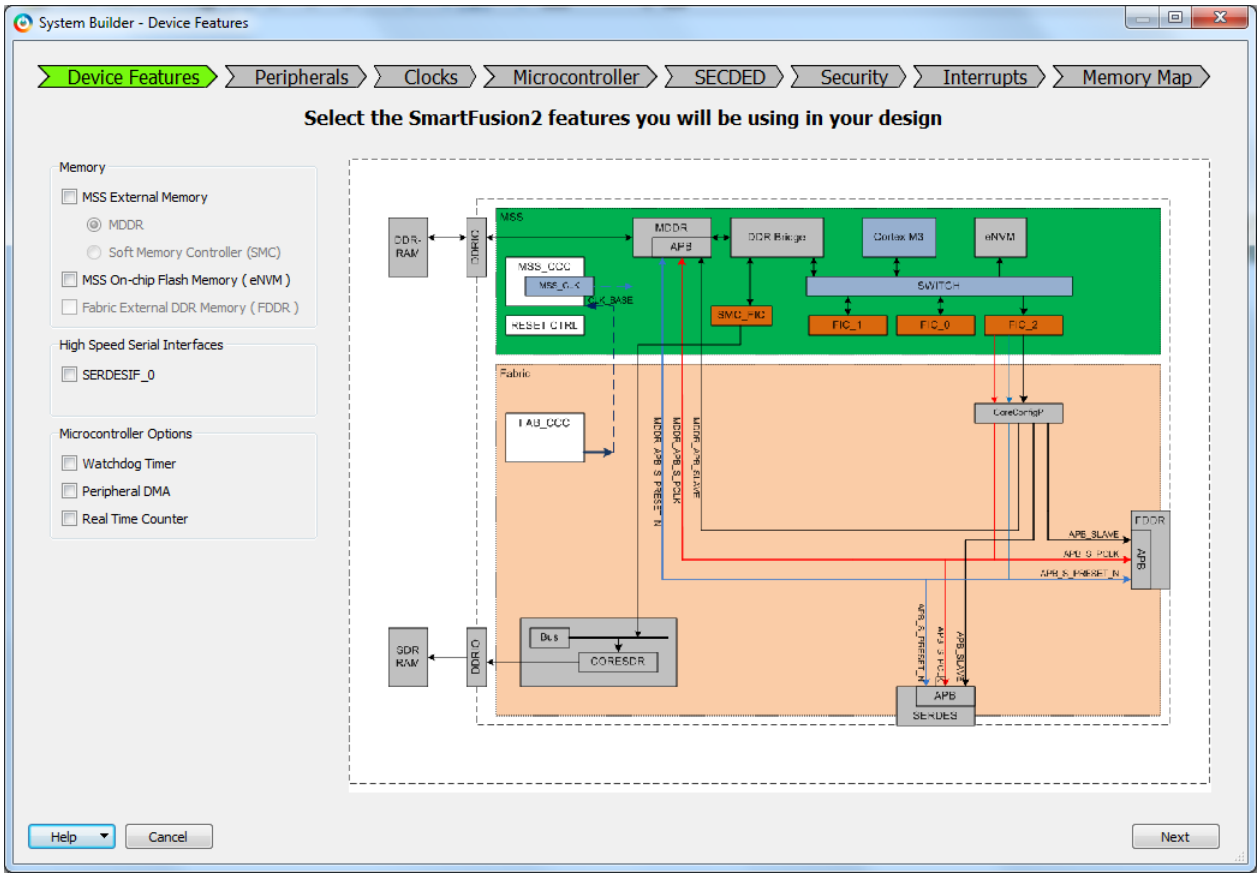
9. Enter a name for your system as **LED_Blink** and then click **OK**, as shown in Figure 8.

Figure 8 • **System Builder Dialog Box**



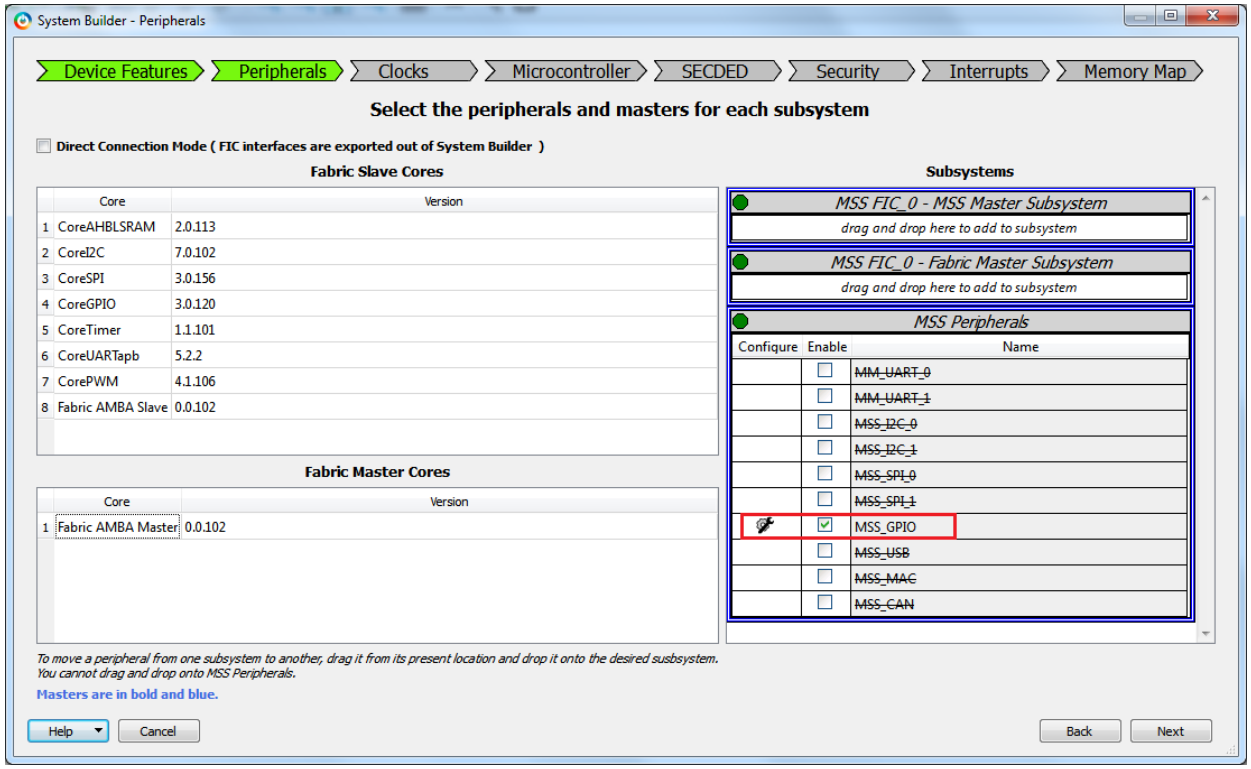
System Builder – Device Features page is displayed, as shown in Figure 9.

Figure 9 • **System Builder – Device Features Page**



10. Retain the default values. Click **Next**, the **System Builder – Peripherals** page is displayed. Under the MSS Peripherals section, uncheck all the check boxes except MSS_GPIO, as shown in Figure 10.

Figure 10 • **System Builder Configurator – Select Peripherals Page**



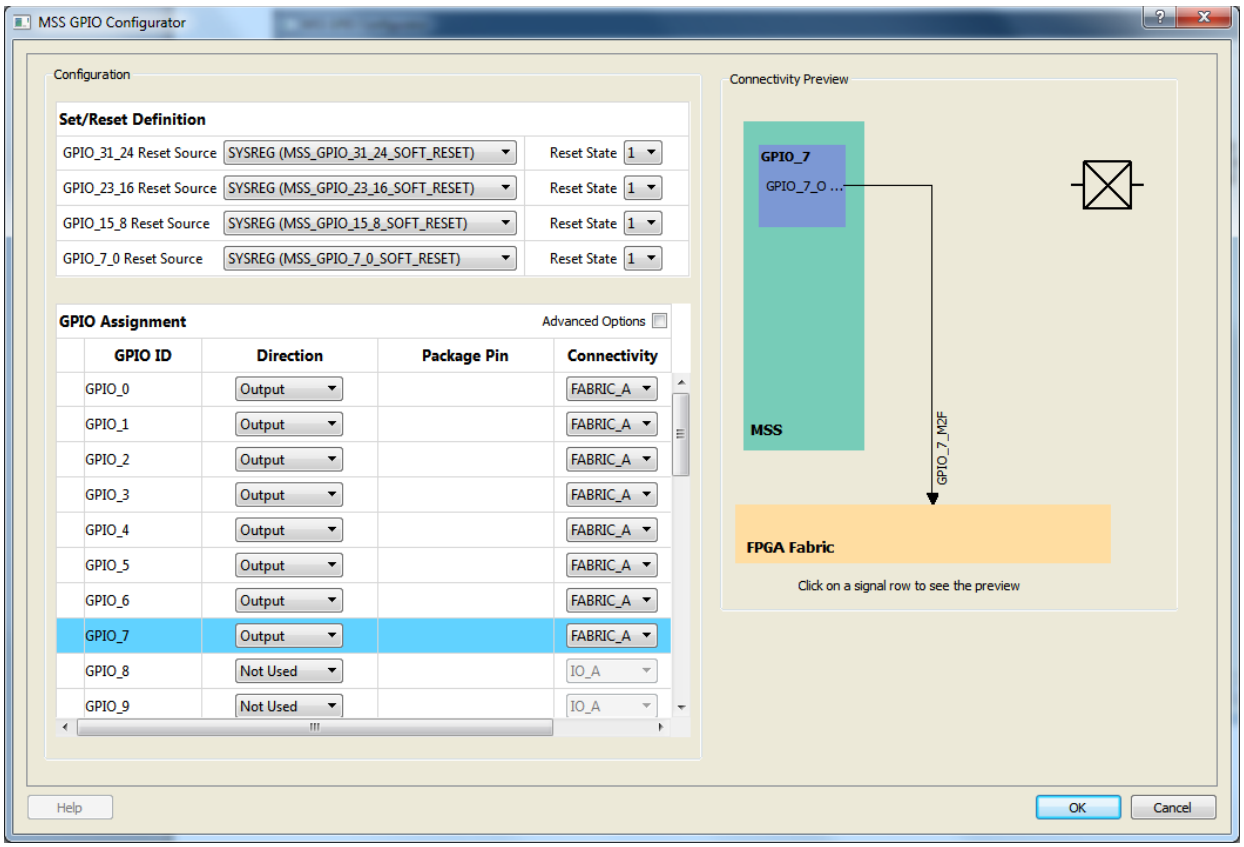
11. Double-click the **wrench** symbol for the MSS_GPIO peripheral to open the MSS_GPIO Configurator.
12. This design requires configuring GPIOs to drive LEDs on the board, configure the GPIOs as shown below:

- **Set/Reset Definition** accept default settings
- Configure GPIO as shown in Table 3

Table 3 • **SmartFusion2 GPIO Configuration**

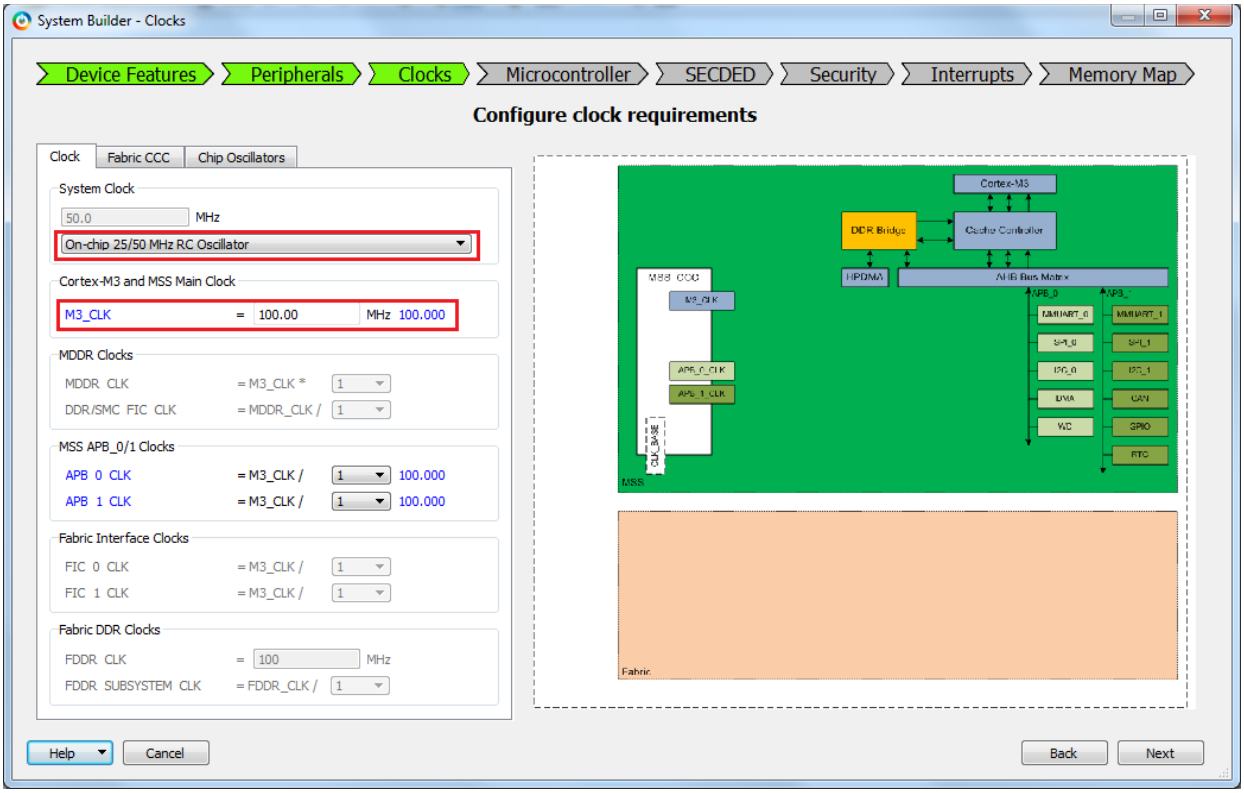
Board	Die	GPIO ID	Direction	Package Pin	Connectivity
SmartFusion2 Security Evaluation Kit	M2S090TS	GPIO_0 to GPIO_7	Output	NA	FABRIC_A
SmartFusion2 Advanced Development Kit	M2S150T	GPIO_0 to GPIO_7	Output	NA	FABRIC_A
SmartFusion2 Starter Kit	M2S010	GPIO_0 to GPIO_1	Output	NA	FABRIC_A

Figure 11 • **MSS_GPIO Configurator**



13. Click **Next**, the **System Builder – Clocks Settings** page is displayed, as shown in Figure 12. Select **System Clock** source as **On-chip 25/50 MHz RC Oscillator**. The M3_CLK is configured to 100 MHz by default.

Figure 12 • **System Builder Configurator – Clock Settings Page**

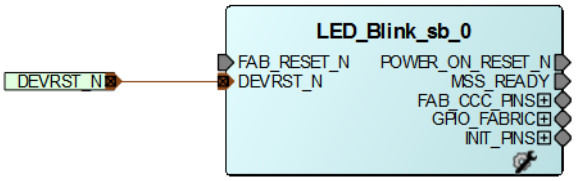


14. Click **Next**, the **System Builder – Microcontroller Options** page is displayed.
    - Retain the default values.
15. Click **Next**, the **System Builder – SECDED Options** page is displayed.
    - Retain the default values.
16. Click **Next**, the **System Builder – Security Options** page is displayed.
    - Retain the default values.
17. Click **Next**, the **System Builder – Interrupts Options** page is displayed.
    - Retain the default values.
18. Click **Next**, the **System Builder – Memory Map Options** page is displayed.
    - Retain the default values.
19. Click **Finish**.

System Builder generates the system based on the selected options.

System Builder block is created and added to the Libero SoC project, as shown in Figure 13.

Figure 13 • **System Builder Generated System**

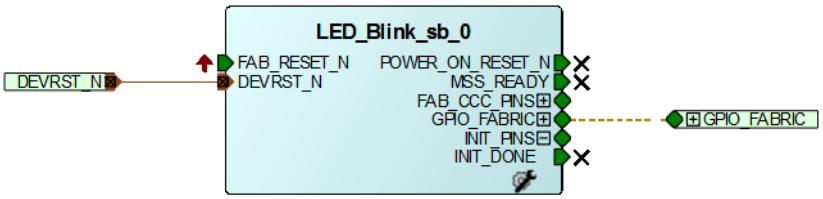


1.4.1 Connecting Components in LED_Blink SmartDesign

The following steps describe connecting components in LED_Blink SmartDesign:

1. Connect the pins as follows:
   - Tie the **FAB_RESET_N** to high by right-clicking and selecting **Tie High**.
   - Mark the output port **POWER_ON_RESET_N** as unused by right-clicking and selecting **Mark Unused**.
   - Mark the output port **MSS_READY** as unused by right-clicking and selecting **Mark Unused**.
   - Expand **INIT_PINS**, right-click **INIT_DONE** and select **Mark Unused**.
   - Expand **FAB_CCC_PINS**, right-click **FAB_CCC_GL0** and select **Mark Unused**.
   - Right-click **FAB_CCC_LOCK** and select **Mark Unused**.
   - Right-click **GPIO_FABRIC** and select **Promote to Top Level**.
2. Click **File** > **Save**. The LED_Blink design is displayed, as shown in Figure 14.

Figure 14 • **LED_Blink Design**

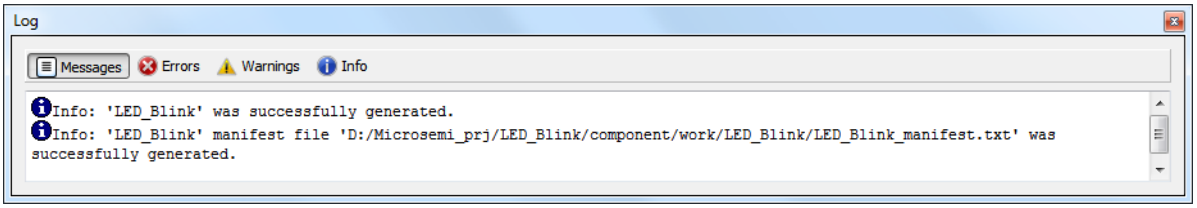


3. Generate the LED_Blink SmartDesign by clicking **SmartDesign** > **Generate Component** or by clicking **Generate Component** on the SmartDesign toolbar.

Figure 15 • **Generate Component**

After successful generation of the system, the message **'info: LED_Blink' was successfully generated** is displayed on the Libero SoC **Log** window, as shown in Figure 16.

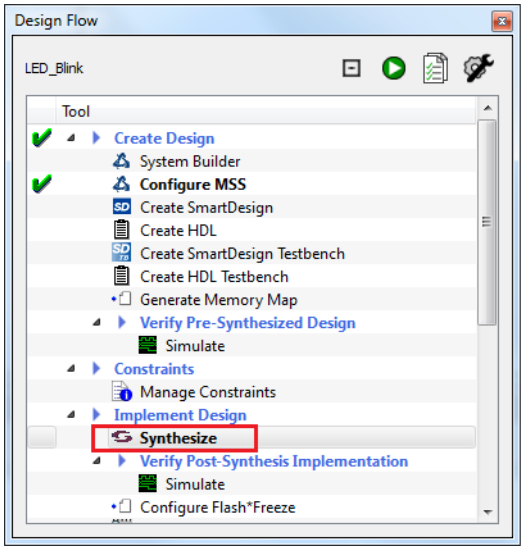
Figure 16 • **Log Window**



1.5 Step 2: Generating the Program File

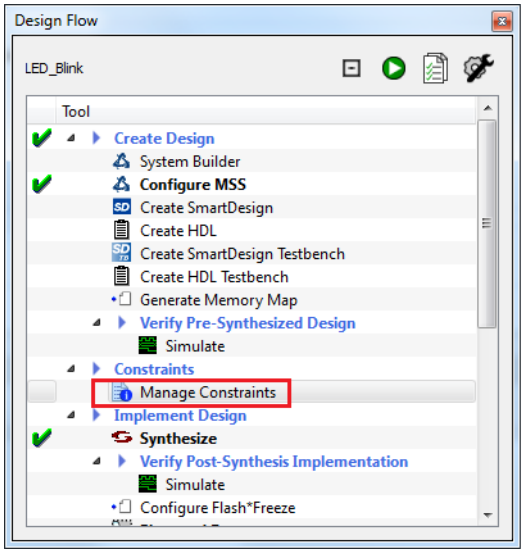
1. Double-click **Synthesize** in the **Design Flow** window, as shown in Figure 17 to complete the synthesis.

Figure 17 • **Design Flow Window**



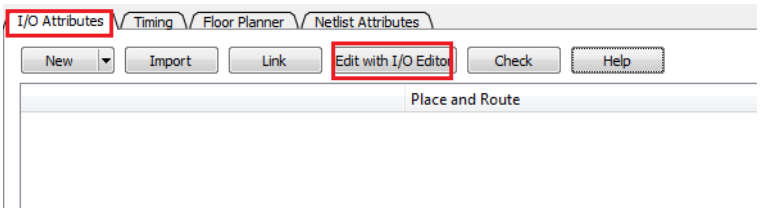
2. Double-click **Manage Constraints** in the **Design Flow** window, as shown in Figure 18.

Figure 18 • **Manage Constraints**



3. Click **Edit with I/O Editor** under **I/O Attributes**, as shown in Figure 19. The I/O Editor window is displayed, as shown in Figure 20 on page 18.

Figure 19 • **I/O Attributes**

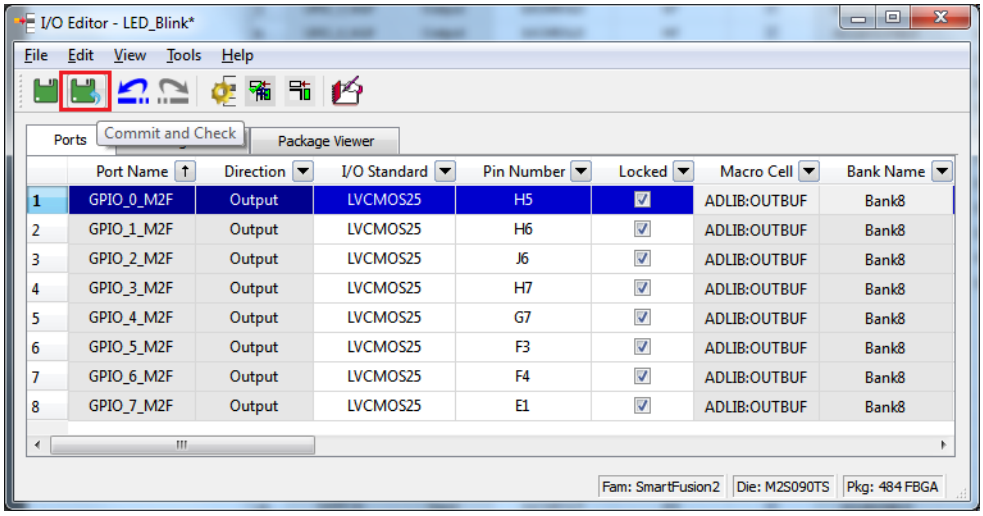


4. Make the pin assignments, as shown in Table 4. After the pins are assigned, the I/O Editor is displayed, as shown in Figure 20.

Table 4 • **Port to Pin Mapping**

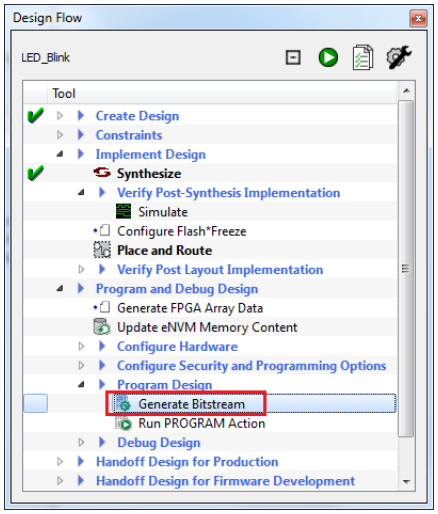
Port Name	Pin Number
SmartFusion2 Security Evaluation Kit	
GPIO_0_M2F	H5
GPIO_1_M2F	H6
GPIO_2_M2F	J6
GPIO_3_M2F	H7
GPIO_4_M2F	G7
GPIO_5_M2F	F3
GPIO_6_M2F	F4
GPIO_7_M2F	E1
SmartFusion2 Advanced Development Kit	
GPIO_0_M2F	D26
GPIO_1_M2F	F26
GPIO_2_M2F	A27
GPIO_3_M2F	C26
GPIO_4_M2F	C28
GPIO_5_M2F	B27
GPIO_6_M2F	C27
GPIO_7_M2F	E26
SmartFusion2 Starter Kit	
GPIO_0_M2F	AB18
GPIO_1_M2F	P1

Figure 20 • **I/O Editor**



5. After updating I/O editor, click **Commit and Check**.
6. Close the **I/O Editor**.
7. Click **Generate Bitstream** in **Design flow** window, as shown in Figure 21 to generate the programming file.

Figure 21 • **Generate Bitstream**



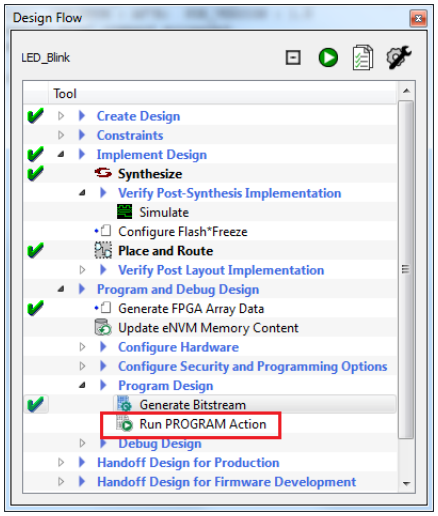
1.6 Step 3: Programming the SmartFusion2 Board Using FlashPro

Jumper settings for the supported target boards and board setup for running the tutorial are given in:

- "Appendix: Board Setup for SmartFusion2 Security Evaluation Kit" section on page 37
- "Appendix: Board Setup for SmartFusion2 Advanced Development Kit" section on page 39
- "Appendix: Board Setup for SmartFusion2 Starter Kit" section on page 41.

1. To program the SmartFusion2 device, double-click **Run PROGRAM Action** in the **Design Flow** window, as shown in Figure 22.

Figure 22 • **Run Programming Action**

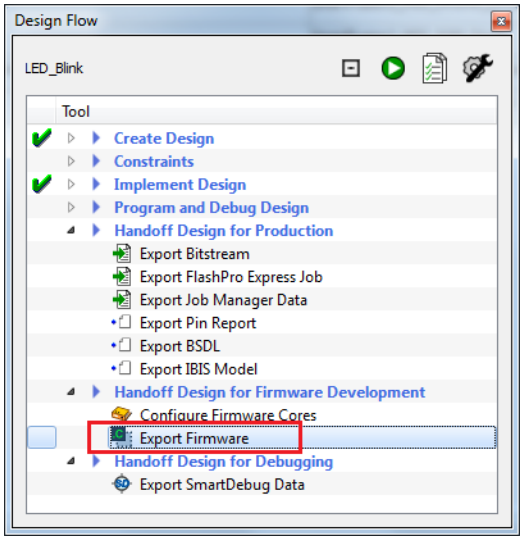


1.7 Step 4: Creating Software Project using SoftConsole 4.0

1.7.1 Export Firmware

1. Double-click **Export Firmware** in **Handoff design for Production** in the **Design Flow** window, as shown in Figure 23.

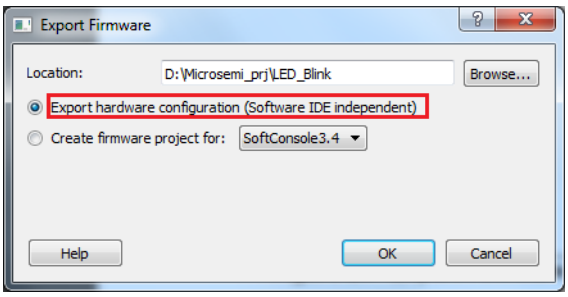
Figure 23 • **Export Firmware**



Export Firmware dialog box is displayed as shown in Figure 24.

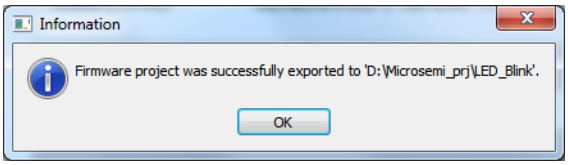
2. Select **Export hardware configuration (Software IDE independent)**, as shown in Figure 24.

Figure 24 • **Export Firmware Dialog Box**



3. Click **OK**, a notification window appears saying **Firmware project was successfully exported to <drive:\>Microsemi_prj\LED_Blink**, as shown in Figure 25.

Figure 25 • **Firmware Export Successful**



4. Click **OK**.

1.7.2 Download Firmware Drivers

The following drivers are used in this tutorial:

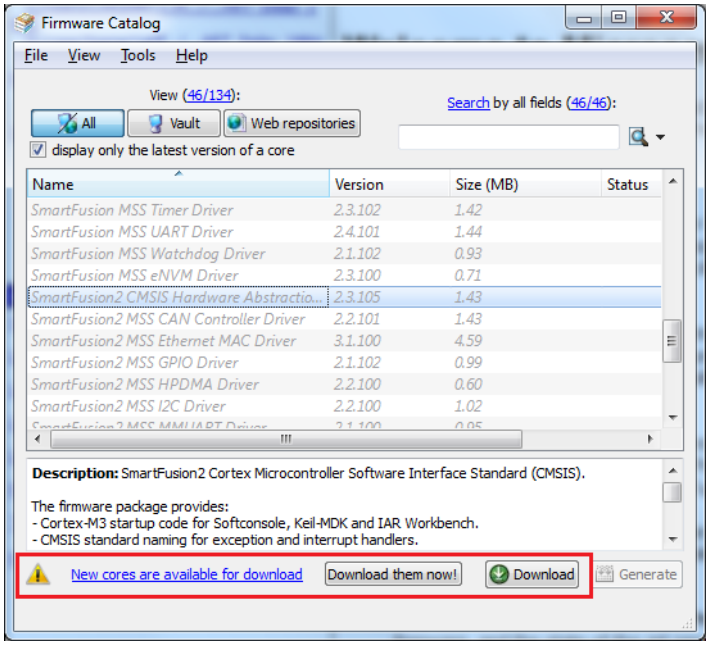
- CMSIS
- GPIO

To generate the required drivers:

1. Open the Microsemi SoC Firmware catalog from: **Start > Programs > Microsemi> Libero SoC v11.7>Firmware Catalog> Firmware Catalog**.

A message **New cores are available for download** at the bottom of the Firmware Catalog is displayed, as shown in Figure 26.

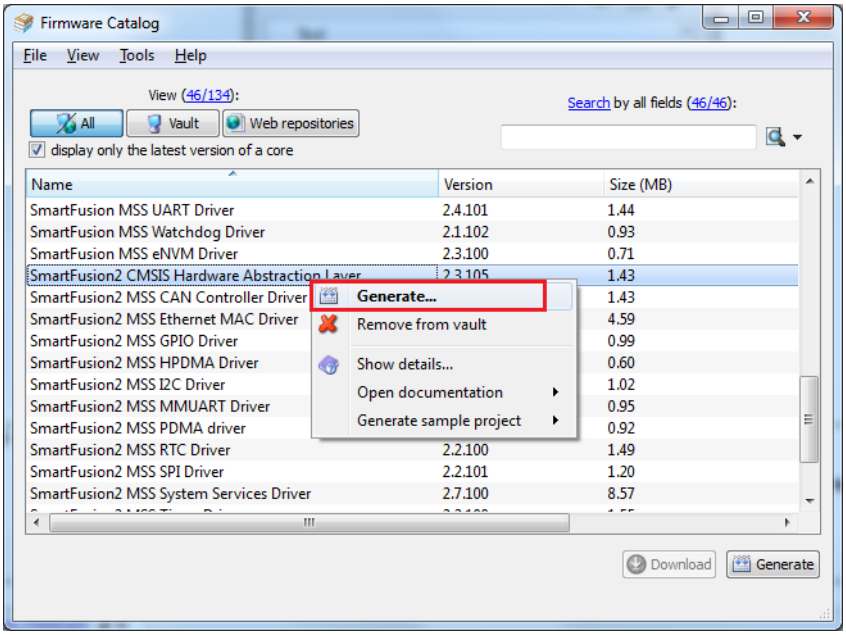
Figure 26 • **Firmware Catalog**



2. Click **Download them now!** to download most recent drivers for peripherals.

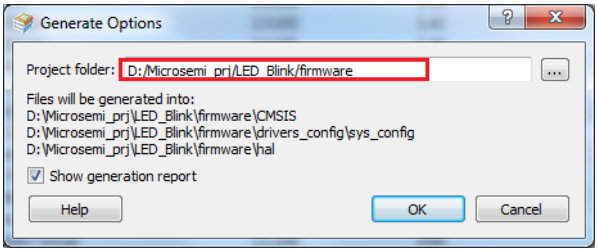
3. In **Firmware catalog** window, right-click **SmartFusion2 CMSIS Hardware Abstraction Layer** and then click **Generate**, as shown in Figure 27.

Figure 27 • **Firmware Catalog**



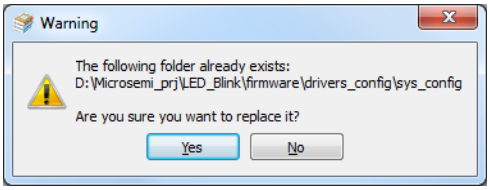
4. In **Generate Options** window, browse **Project folder** at **<drive:\>Microsemi_prj\LED_Blink\firmware** and then click **Ok**.

Figure 28 • **Generate Options**



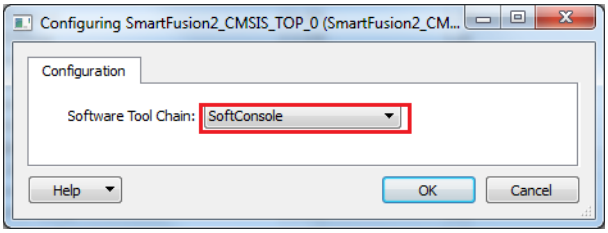
5. A warning message is displayed, as shown in Figure 29. Click **Yes.**

Figure 29 • **Warning Message**



6. **Configuring SmartFusion2_CMSIS_TOP_0** window is displayed, as shown in Figure 30. Select **SoftConsole** from the Software Tool Chain drop-down list and click **Ok**.

Figure 30 • **Software Tool Chain**

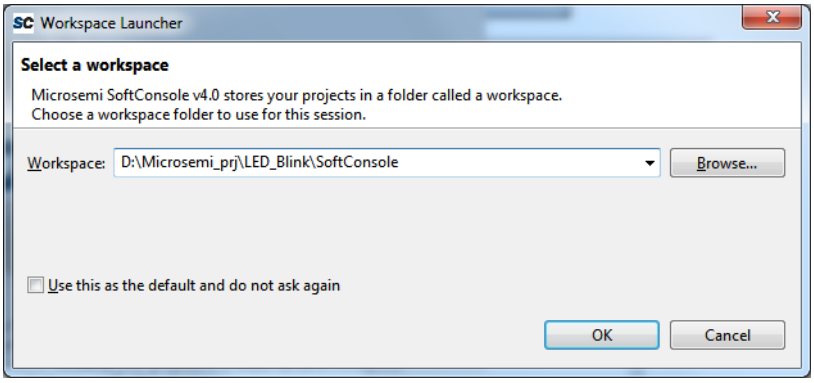


7. Repeat steps 3 to 7 to download SmartFusion2 MSS GPIO driver.

1.7.3 Building Software Application using SoftConsole 4.0

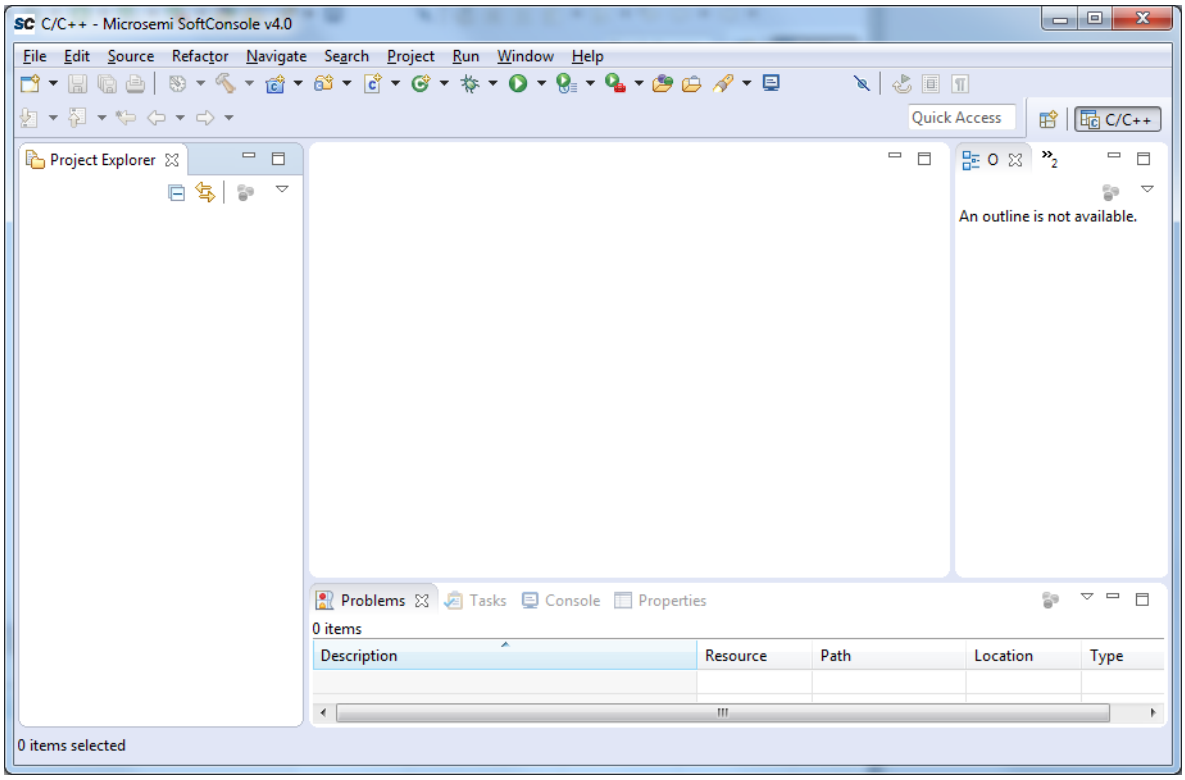
1. Click **Start > Programs > Microsemi SoftConsole v4.0 > Microsemi SoftConsole v4.0** to open the SoftConsole IDE. The SoftConsole **Workspace Launcher** window is displayed.
2. Browse to the location to select D:\Microsemi_prj\LED_Blink\SoftConsole, as shown in Figure 31.

Figure 31 • **Workspace Launcher**



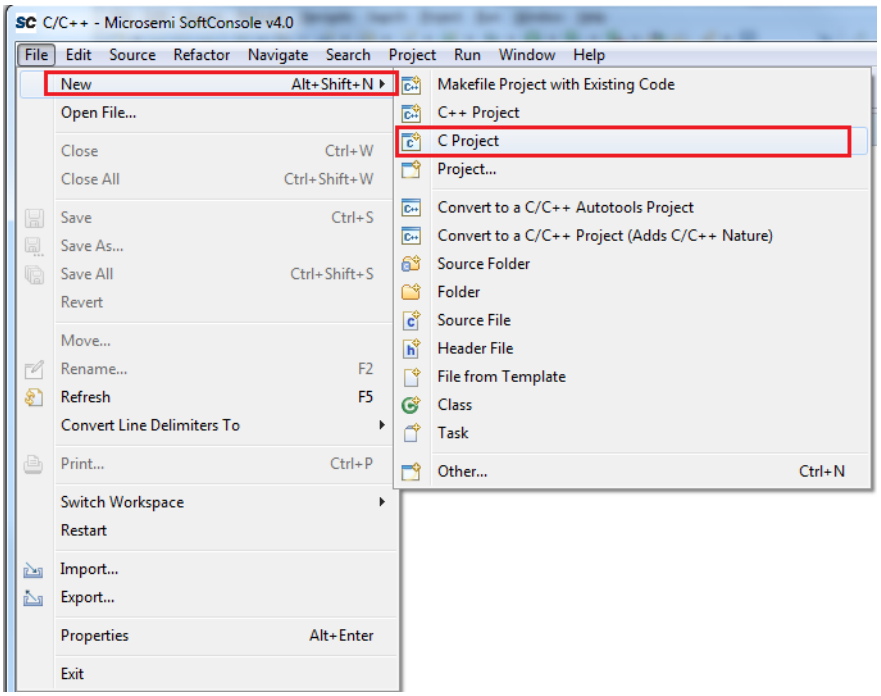
The SoftConsole workspace is displayed, as shown in Figure 32.

Figure 32 • **SoftConsole Window**



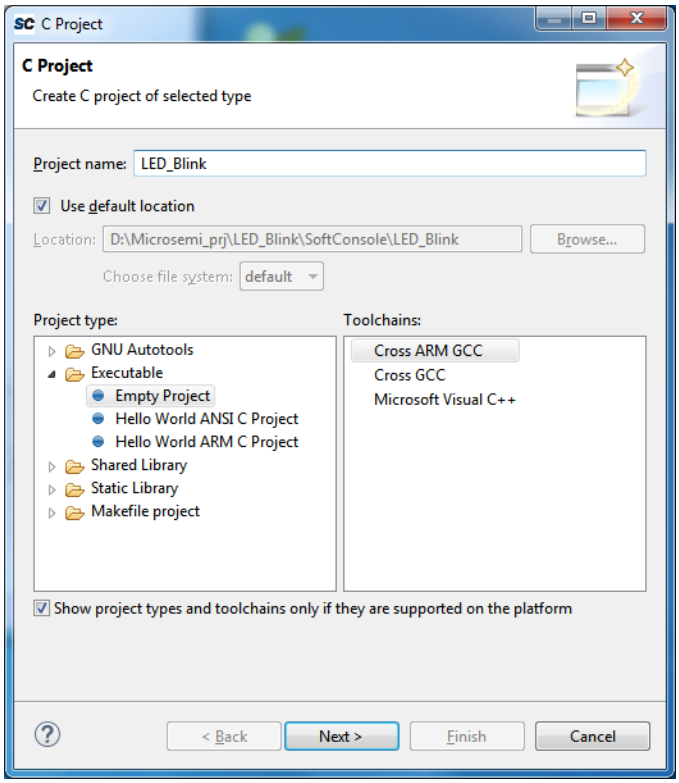
3. Click **File >New >C project** as shown in Figure 33.

Figure 33 • **Creating New C Project**



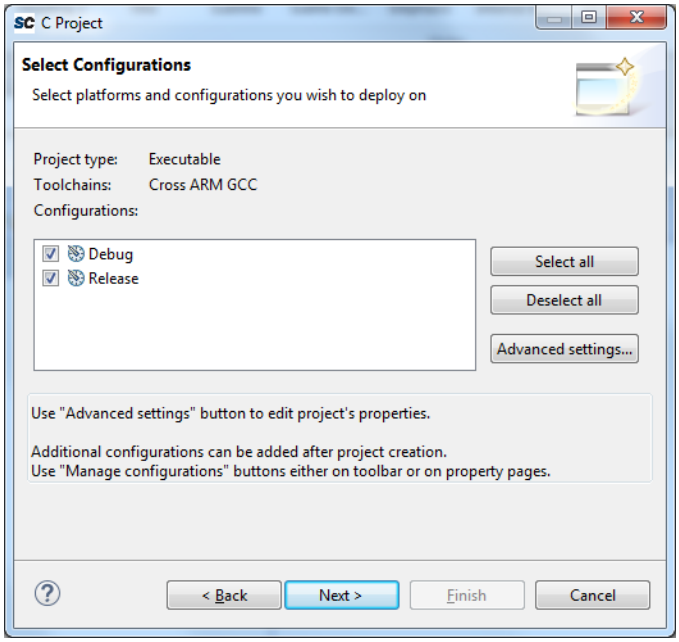
4. Enter Project name as LED_Blink, as shown in Figure 34.

Figure 34 • **C Project Window**



5. Click **Next**, **Select Configurations** window is displayed, as shown in Figure 35.

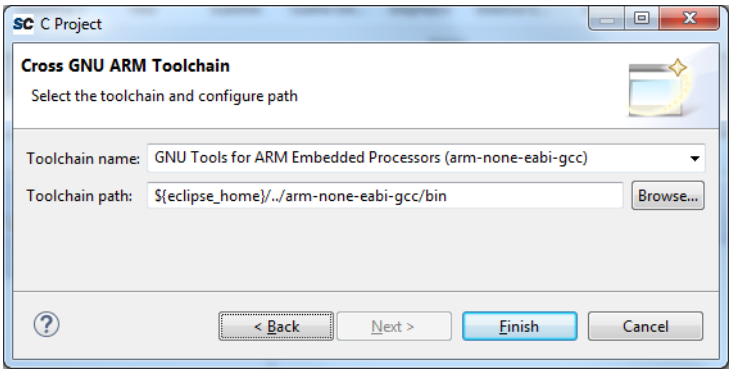
Figure 35 • **C Project - Select Configuration**



6. Do not change default settings. Click **Next**.

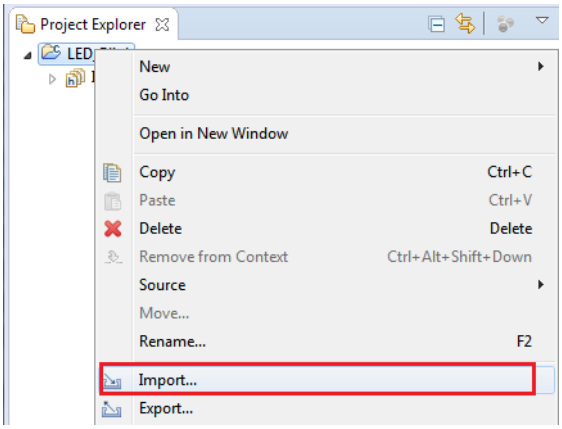
Cross GNU ARM Tool chain window is displayed, as shown in Figure 36.

Figure 36 • **C Project - Cross GNU ARM Tool Chain**



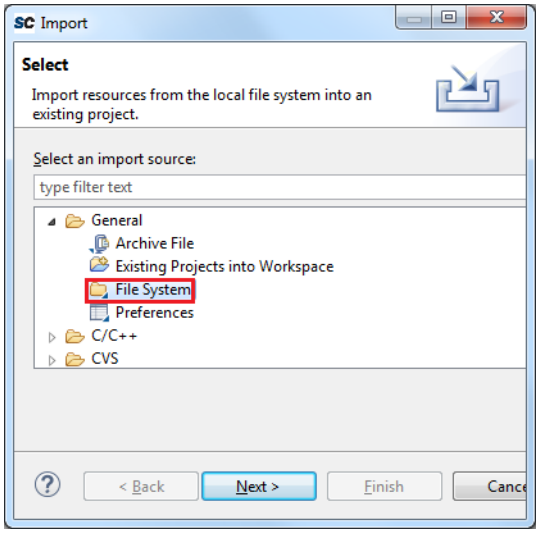
7. Click **Finish**.
8. Right-click **LED_Blink** and click **Import** as shown in Figure 37.

Figure 37 • **Project Explorer - Import**



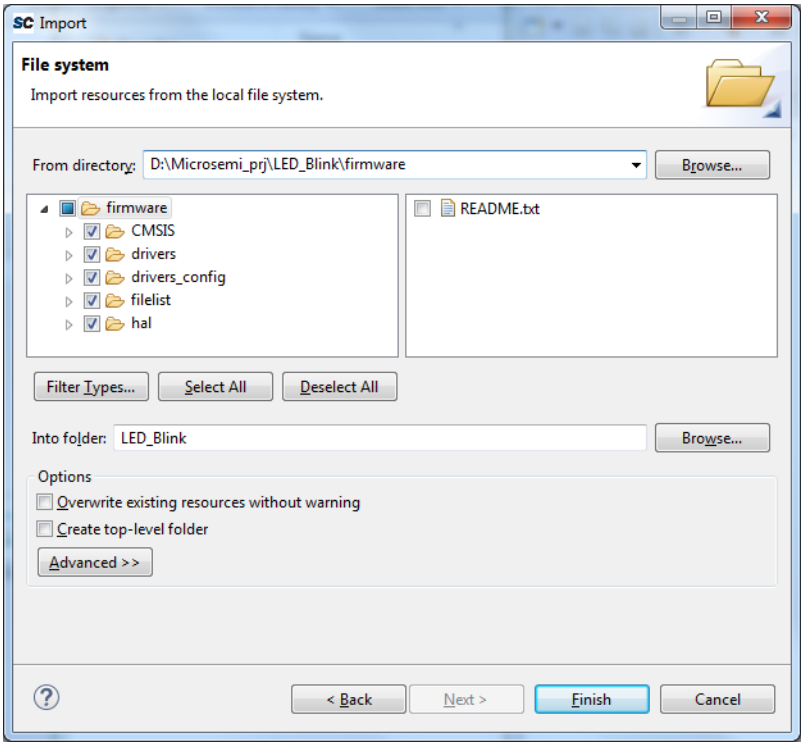
9. **Import** window is displayed, as shown Figure 38.
10. Click **File System** and then click **Next** as shown in Figure 38.

Figure 38 • **Import Window**



11. Browse to **D:\Microsemi_prj\LED_Blink\firmware** and check **firmware** check box, as shown in Figure 39.

Figure 39 • **Import - File System**

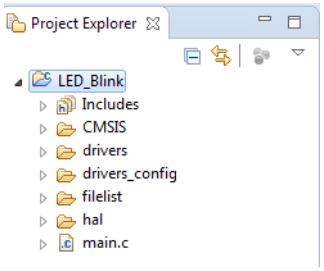


12. Click **Finish**.

Note: If any changes are made to the Libero SoC project, firmware need to be exported from Libero and new firmware must be imported to **LED_Blink**.

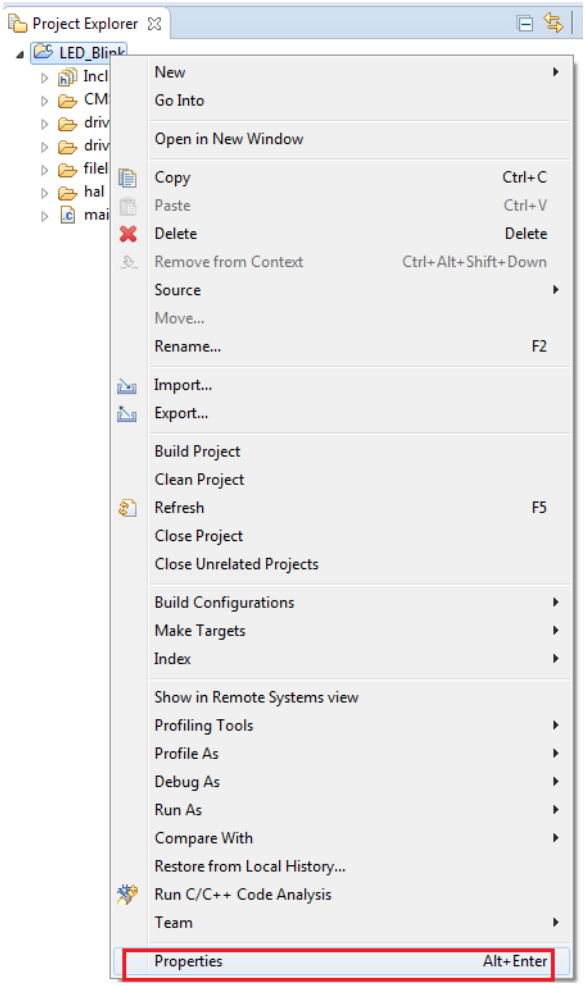
13. Using Windows explorer, browse to the `main.c` file location in the design files folder:
    - For SmartFusion2 Security Evaluation Kit: *<download_folder>\SF2_LED_Blink_SC_Tutorial_DF\Sourcefiles\SF2_Security_Kit*.
    - For SmartFusion2 Advanced Development Kit: *<download_folder>\SF2_LED_Blink_SC_Tutorial_DF\Sourcefiles\SF2_Adv_Dev_Kit*.
    - For SmartFusion2 Starter Kit: *<download_folder>\SF2_LED_Blink_SC_Tutorial_DF\Sourcefiles\SF2_Starter_Kit*.
14. Copy the `main.c` file to the **LED_Blink** project in the SoftConsole workspace, as shown in Figure 40.

Figure 40 • **Project Explorer**



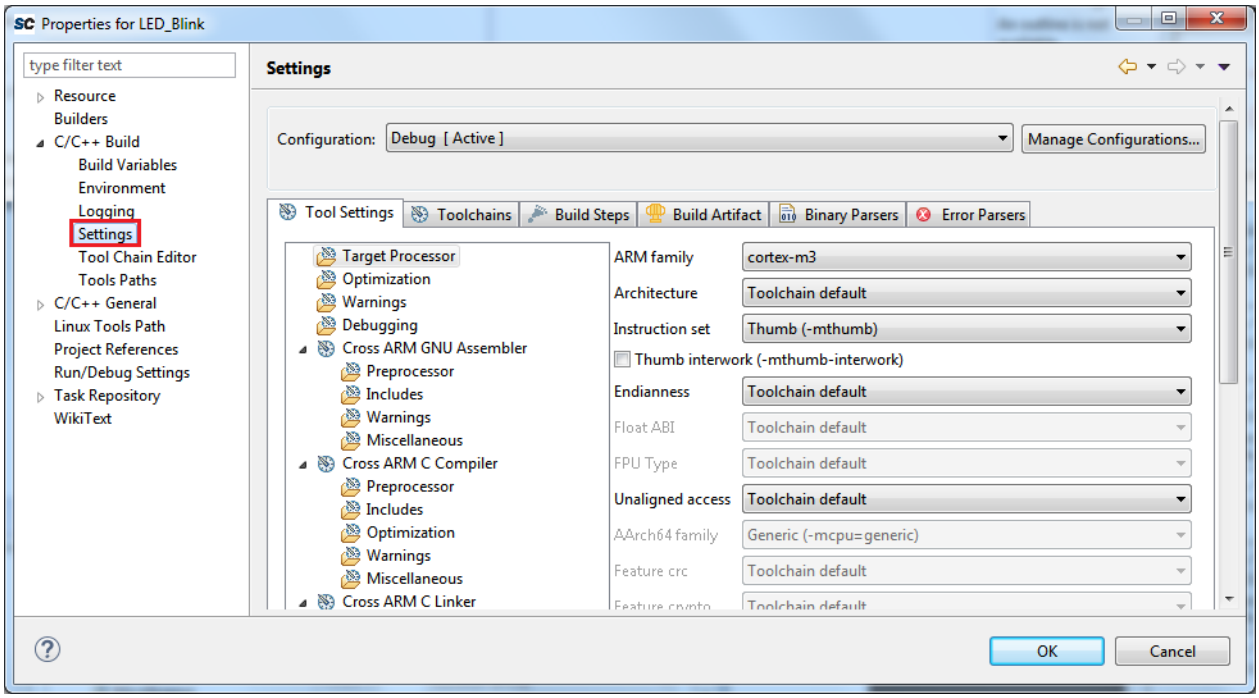
15. Right-click **LED_Blink** and click **Properties,** as shown in Figure 41.

Figure 41 • **Project Explorer window - Properties**



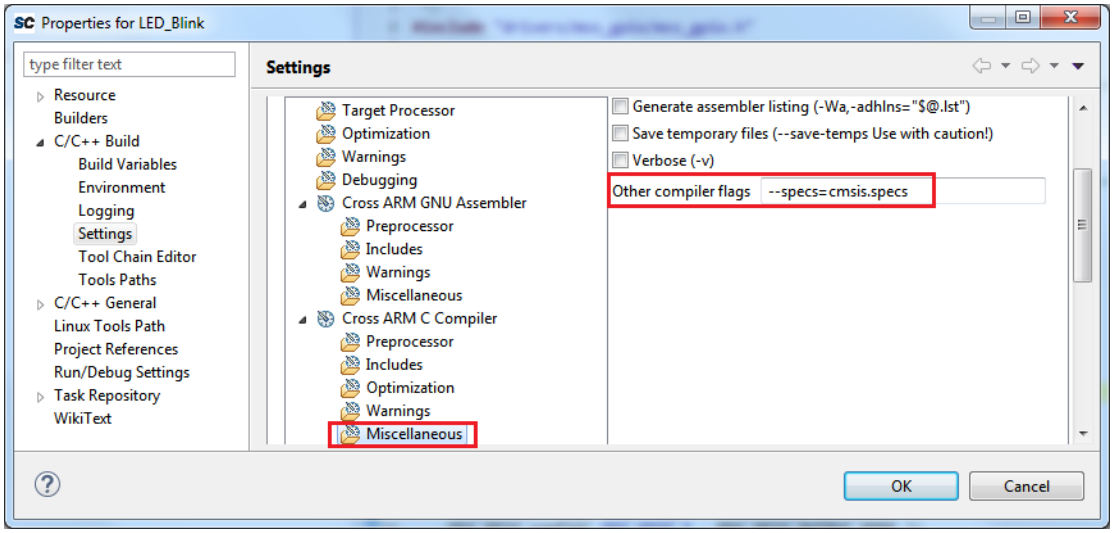
16. Click **Settings** under the **C/C++ Build** tab, as shown in Figure 42.

Figure 42 • **Properties for LED_Blink**



17. Under **Cross ARM C compiler**, click **Miscellaneous** and enter **--specs=cmsis.specs**, in **Other compiler flags** text box as shown in Figure 43.

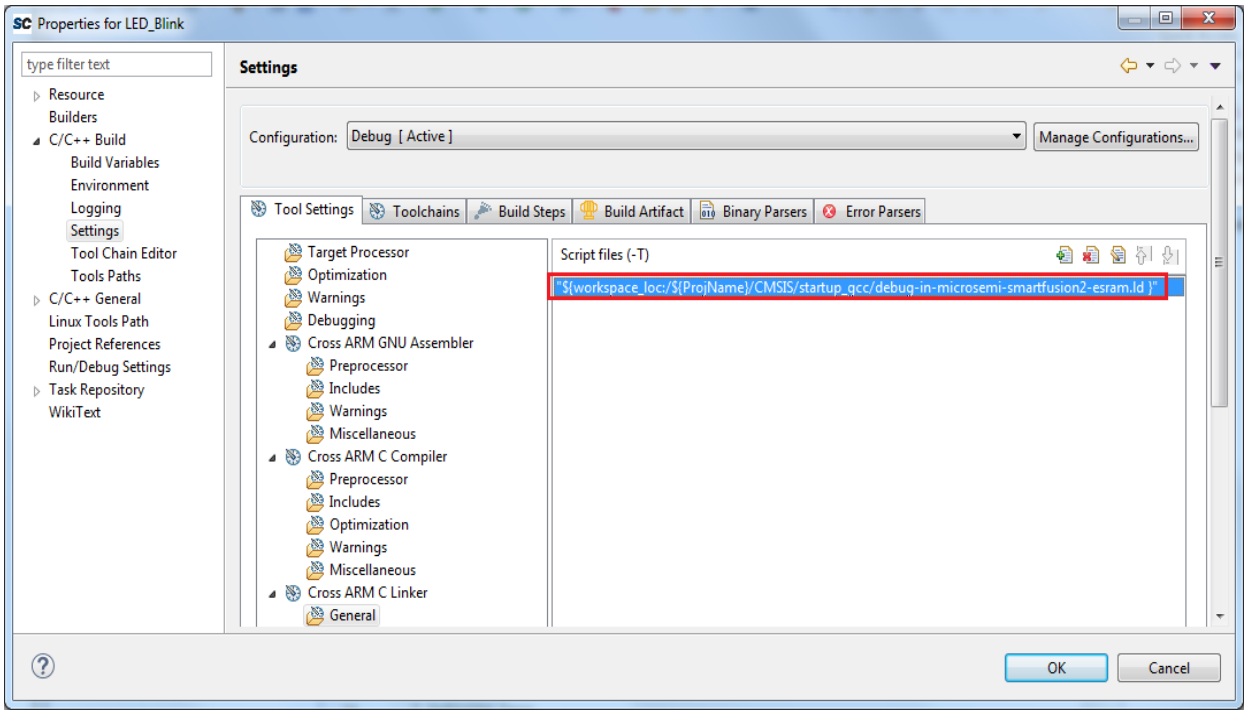
Figure 43 • **Properties for LED_Blink - Miscellaneous**



18. Under **Cross ARM C Linker**, click **General** as shown in Figure 44.
19. Click **add button** and add following linker Script path:
    "${workspace_loc:/${ProjName}/CMSIS/startup_gcc/debug-in-microsemi-smartfusion2-esram.ld }"

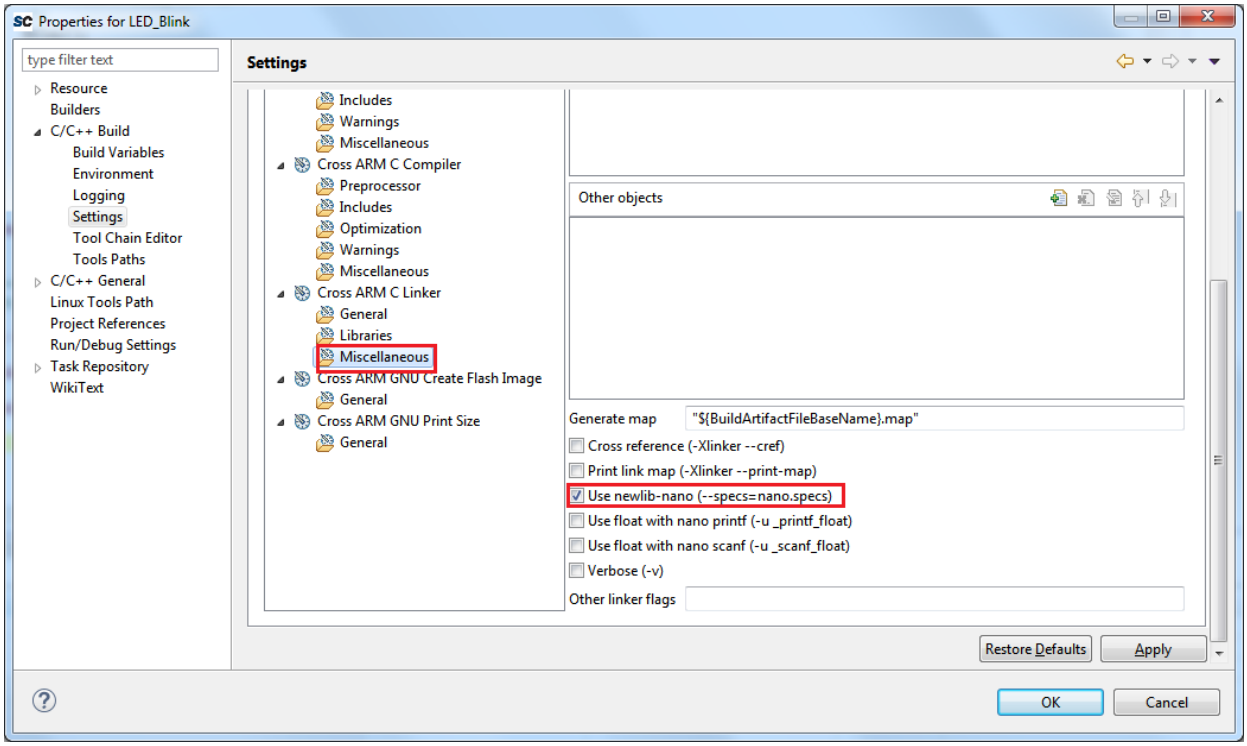
After adding Linker Script, **Properties for LED_Blink** window is displayed, as shown in Figure 44.

Figure 44 • **Properties for LED_Blink - General**



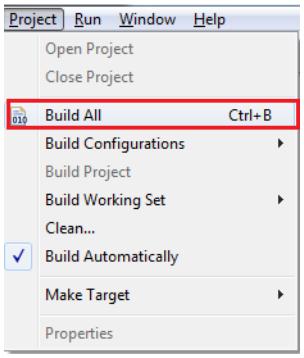
20. Under **Cross ARM C Linker**, click **Miscellaneous**.
21. Check **Use newlib-nano(--specs=nano.specs)** option, as shown in Figure 45.

Figure 45 • **Properties for LED_Blink - Miscellaneous**



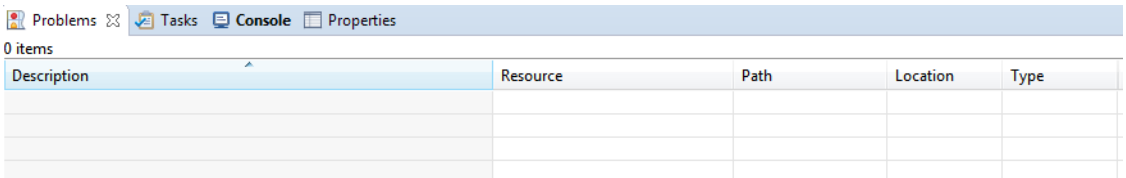
22. Click **Ok**.
23. Click **Project** and click **Build All**, as shown in Figure 46.

Figure 46 • **Project - Build All**



24. Ensure Problems window must not have any errors, as shown in Figure 47.

Figure 47 • **Problem Window**

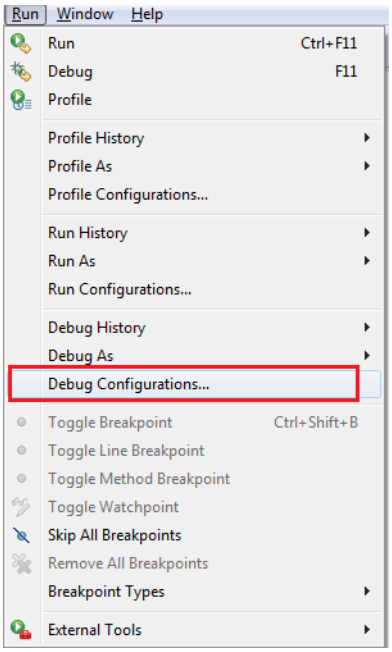


1.7.4 Debugging the Application Project using SoftConsole v4.0

The following steps describe how to debug the application project using SoftConsole v4.0:

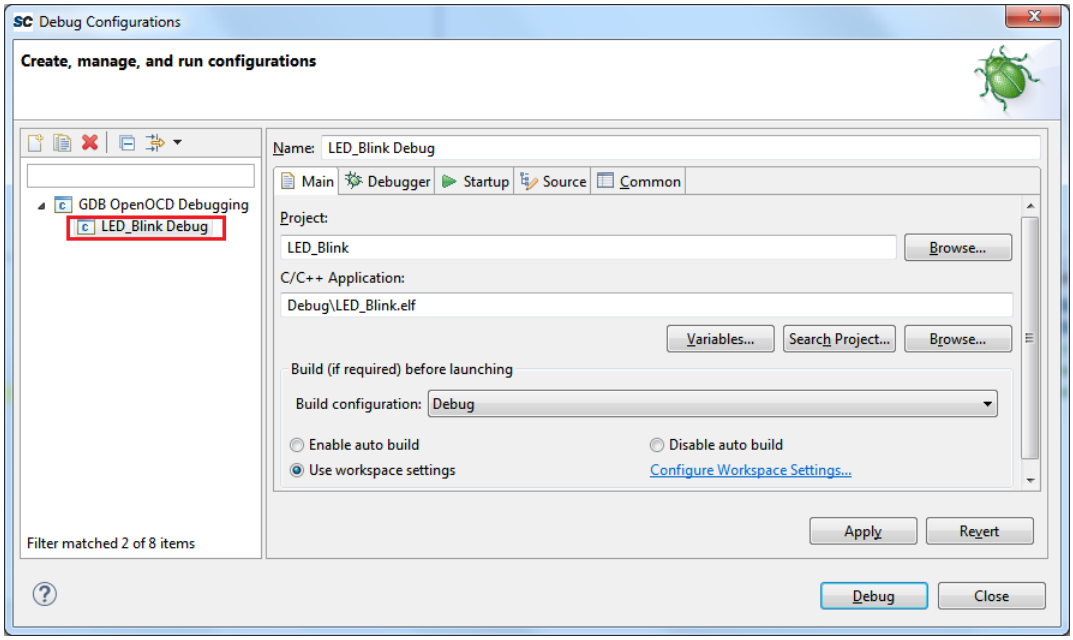
1. Click **Debug Configurations** in the **Run** menu of the SoftConsole, as shown in Figure 48. The **Debug Configurations** window is displayed, as shown in Figure 49.

Figure 48 • **Run - Debug Configurations**



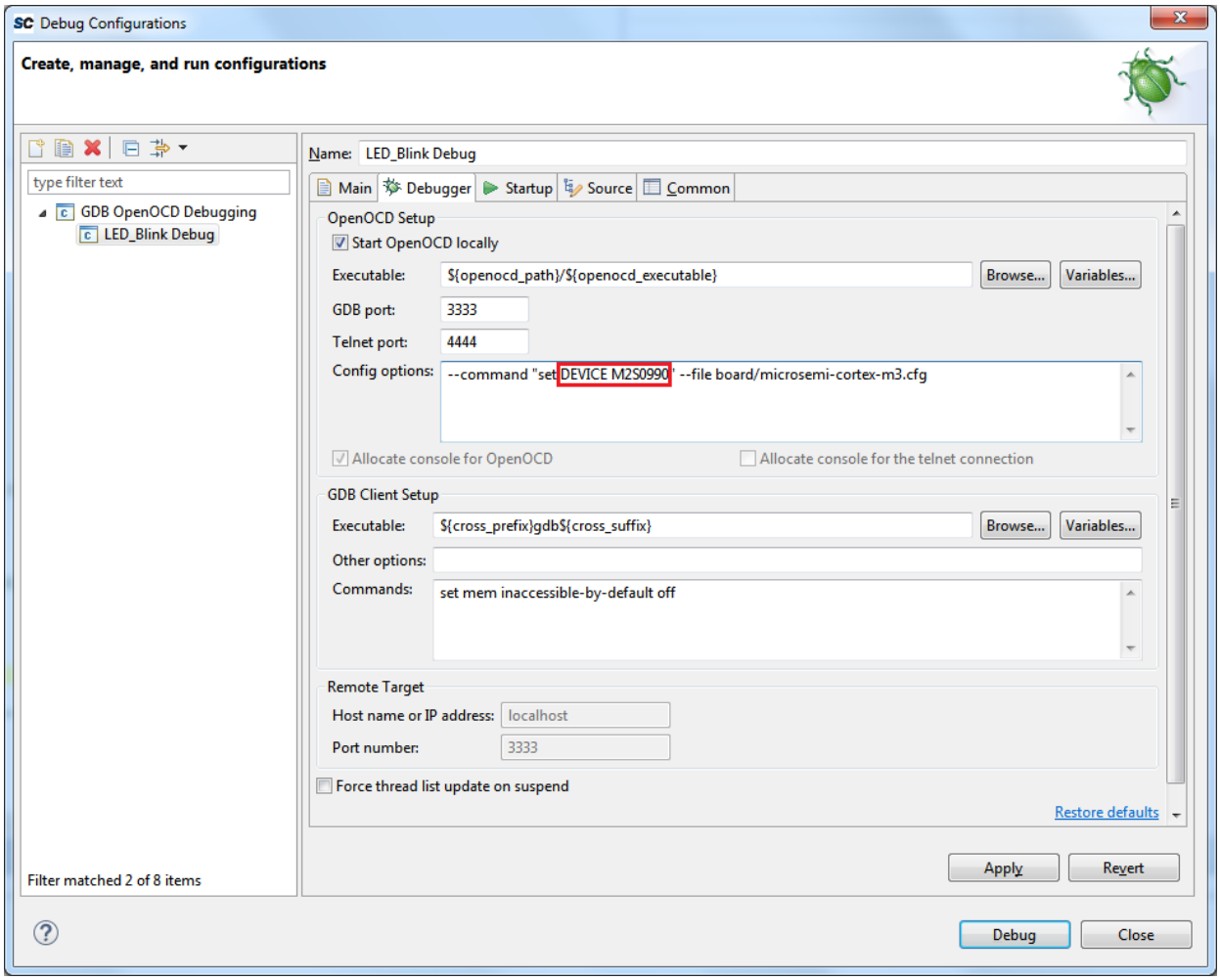
2. Double-click **GDB OpenOCD Debugging** to view the configurations, as shown in Figure 49.

Figure 49 • **Debug Configurations**



3. Ensure that the following values are filled in the corresponding fields:
   - **Name**: LED_Blink Debug
   - **Project**: LED_Blink
   - **C/C++ Application**: Debug\LED_Blink.elf
4. Select the **Debugger** tab in the **Debug Configurations** dialog box. **--command "set DEVICE M2S090"** specifies the target device, as shown in Figure 50. This command needs to be modified based on the target silicon.
   - SmartFusion2 Security Evaluation Kit - set DEVICE M2S090
   - SmartFusion2 Advanced Development Kit - set DEVICE M2S150
   - SmartFusion2 Starter Kit - set DEVICE M2S010

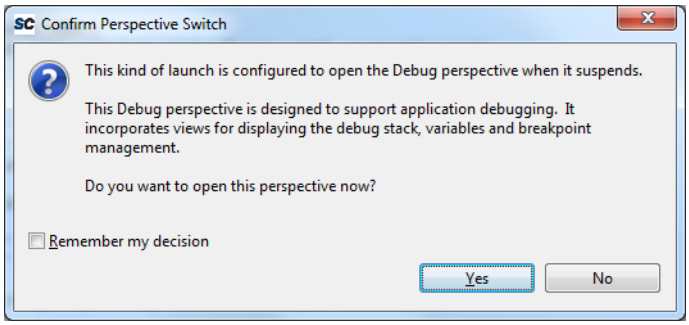
Figure 50 • **Debug Tab**



5. Click **Debug**.

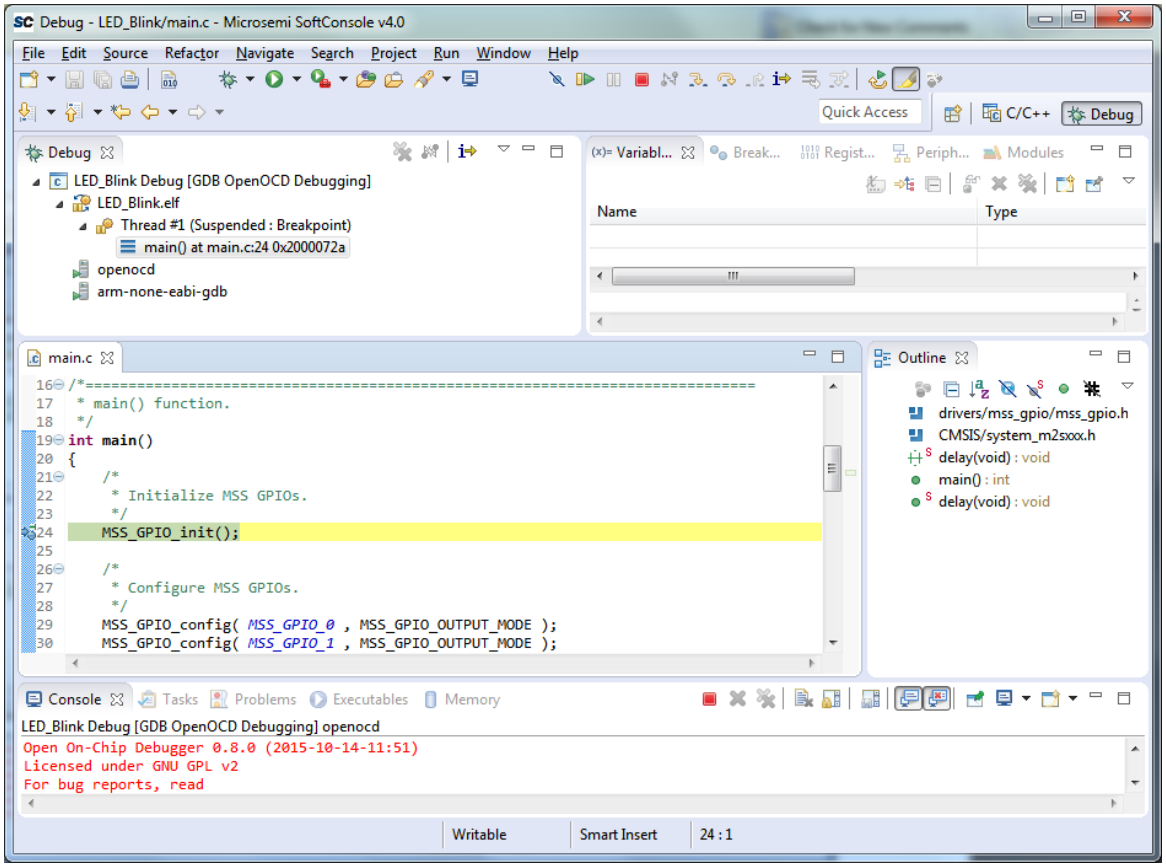
6. On the **Confirm Perspective Switch** window, click **Yes** as shown in Figure 51.

Figure 51 • **Confirm Perspective Switch**



The **SoftConsole Debugger Perspective** window is displayed, as shown in Figure 52.

Figure 52 • **Debugger Perspective Window**



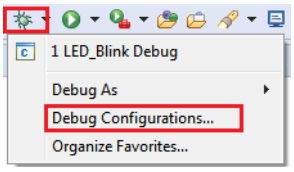
7. Click **Run > Resume** to run the application. LEDs start blinking on the SmartFusion2 target boards. Table 5 shows which LEDs blink for the different SmartFusion2 target boards.

Table 5 • **LED Target Board**

Target Board	LEDs
SmartFusion2 Security Evaluation Kit	H5, H6, J6, H7, G7, F3, F4, and E1
SmartFusion2 Advanced Development Kit	DS0, DS1, DS2, DS3, DS4, DS5, DS6, and DS7
SmartFusion2 Starter Kit	DS4, DS3

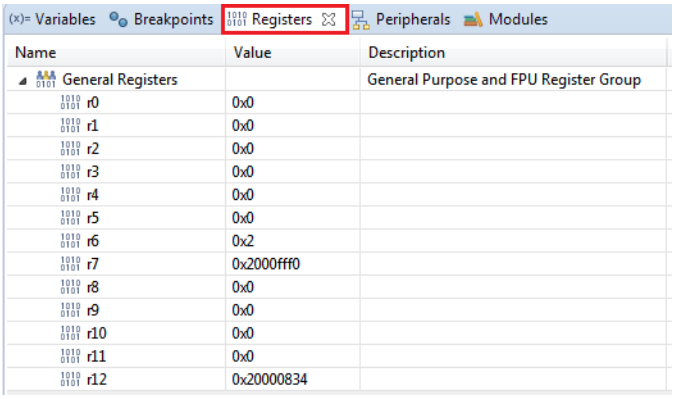
8. Launch the debug session:
   - By selecting **Debug Configurations** from the **Run** menu of SoftConsole.
     or
   - By selecting the **Debug Configurations** using the Debug button as shown in Figure 53.

Figure 53 • **Debug Configurations Option**



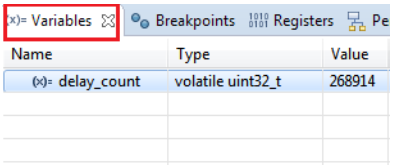
9. Click the **Registers** tab to view the values of the Cortex-M3 processor internal registers, as shown in Figure 54.

Figure 54 • **Values of Cortex-M3 Internal Registers**



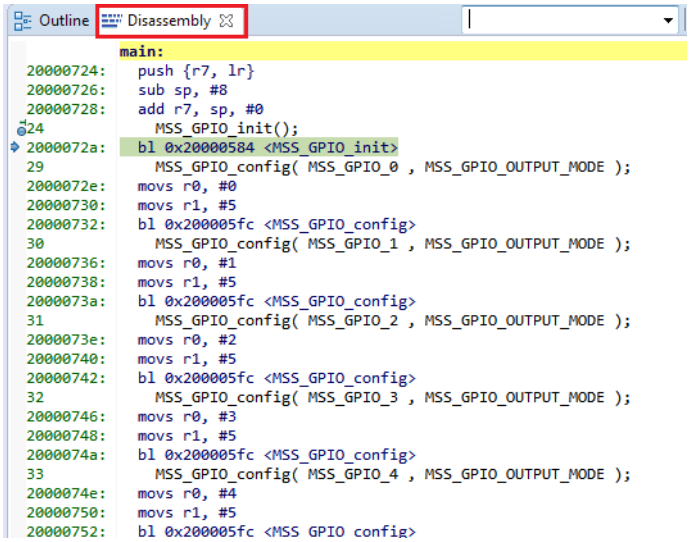
10. Click the **Variables** tab to view the values of variables in the source code, as shown in Figure 55.

Figure 55 • **Values of the Variables in the Source Code**



11. In the **Debug** window, click **Window > Show View > Disassembly** to display the assembly level instructions. The **Disassembly** window with assembly instructions is displayed on the right-side of the Debug perspective, as shown Figure 56.

Figure 56 • **Assembly Level Instructions**



12. Source code can be single-stepped by choosing **Run > Step Into or Run > Step Over**. Observe the changes in the source code window and Disassembly view. Performing a Step Over provides an option for stepping over functions. The entire function is run but there is no need to single-step through each instruction contained in the function.
13. Click **Instruction Stepping** ( ) and perform **Step Into** operations. Observe that **Step Into** executes a single line of assembly code.
14. Click **Instruction Stepping** to exit the instruction stepping mode. Single-step through the application and observe the instruction sequence in the source code window of the Debug perspective, and the values of the variables and registers.
15. Add breakpoints in the application to force the code to halt, single-step, and observe the instruction sequence.
16. When debug process is finished, terminate execution of the code by choosing **Run > Terminate**.
17. Close Debug Perspective by selecting **Close Perspective** from the **Window** menu.
18. Close SoftConsole using **File > Exit**.
19. Close the HyperTerminal using **File > Exit**.

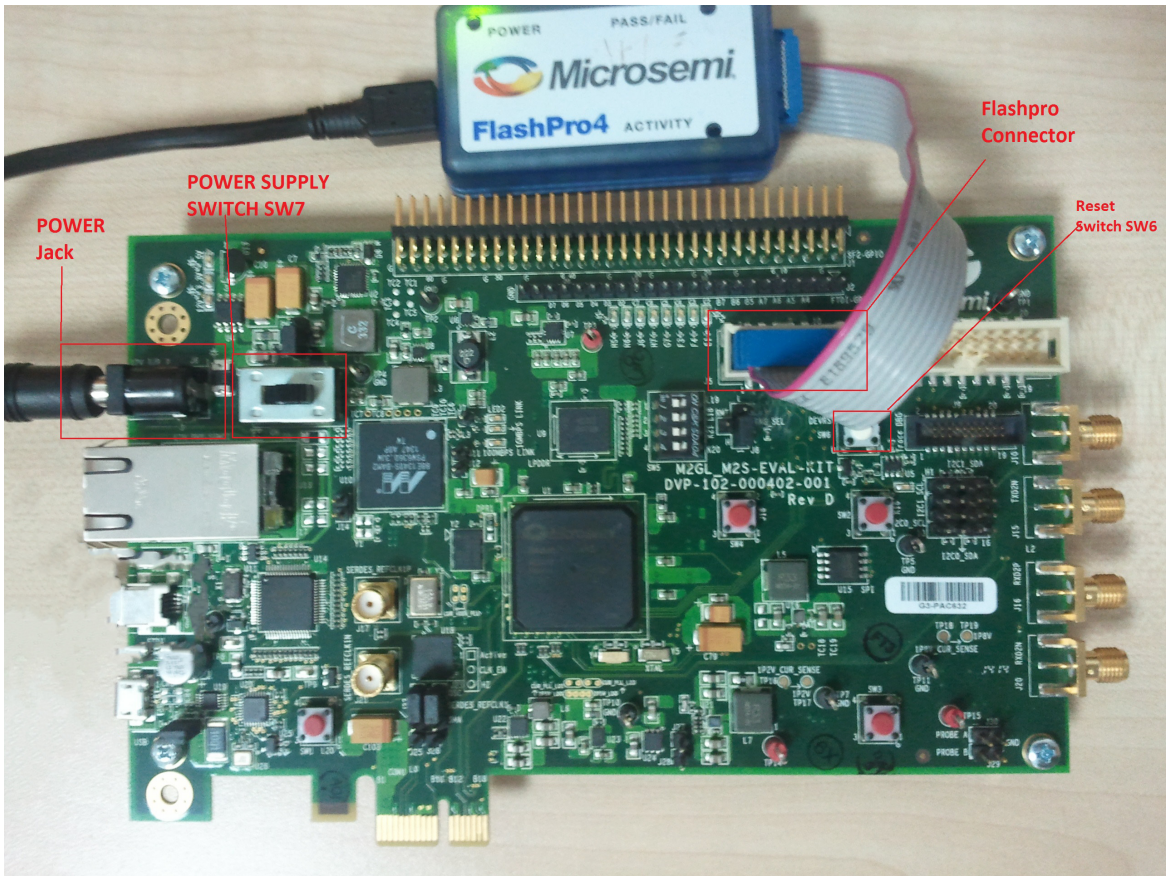
1.8 Conclusion

This tutorial provides steps to create a Libero SoC design using System Builder. It describes the procedure to build, debug, and run a SoftConsole application. It also provides a simple design to blink LEDs.

2 Appendix: Board Setup for SmartFusion2 Security Evaluation Kit

Figure 56 shows the board setup for running the tutorial on the SmartFusion2 Security Evaluation Kit board.

Figure 56 • **SmartFusion2 Security Evaluation Kit Setup**



1. Connect the jumpers on the SmartFusion2 Security Evaluation Kit board as listed in Table 6. For more information on jumper locations, see Figure 57 on page 38 for SmartFusion2 Security Evaluation Kit board Jumper Locations.

CAUTION: While making the jumper connections, the **SW7** power supply switch on the board must be in **OFF** position.

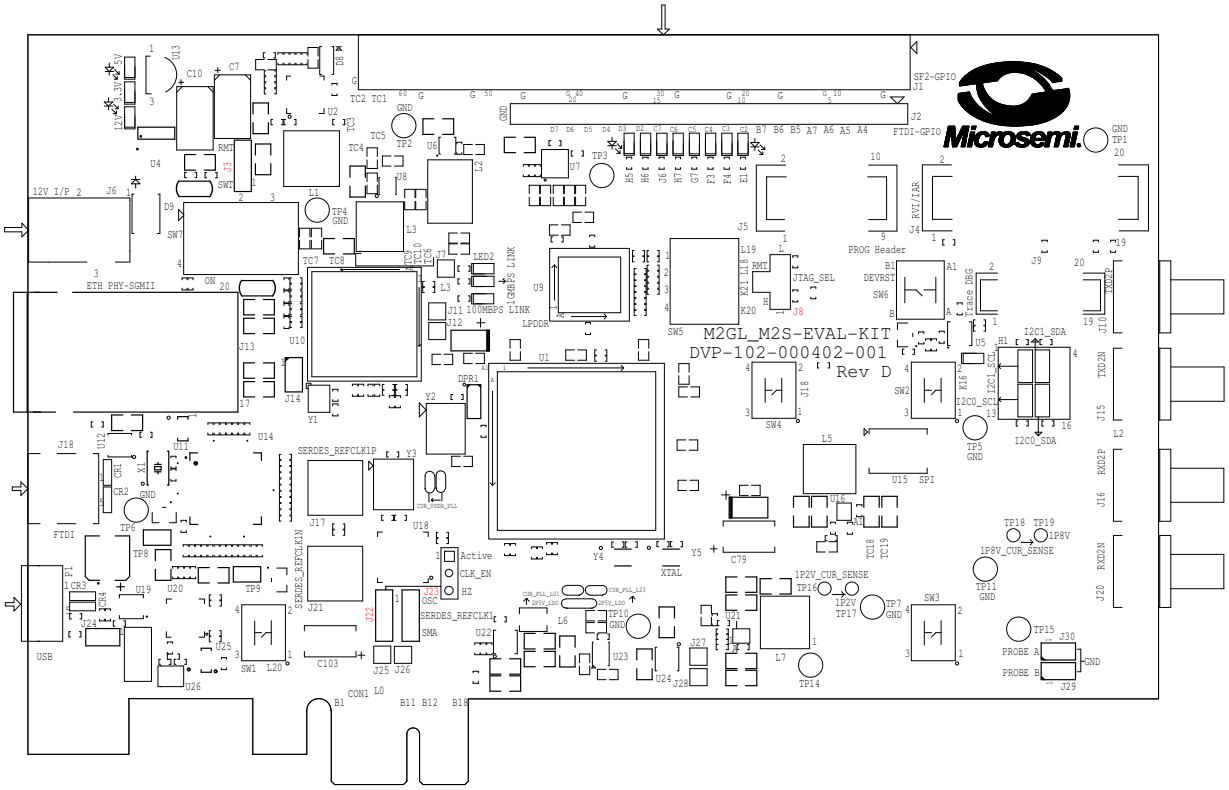
Table 6 • **SmartFusion2 Security Evaluation Kit Jumper Settings**

Jumper	Pin (From)	Pin (To)	Comments
J22, J23, J8, and J3	1	2	These are the default jumper settings of the SmartFusion2 Security Evaluation Kit board. Ensure, these jumpers are set accordingly.

2. Connect the FlashPro4 or FlashPro5 programmer to the **J5** connector of the SmartFusion2 Security Evaluation Kit.
3. Connect the power supply to the **J6** connector.
4. Switch **ON** the **SW7** power supply switch.

Figure 57 shows the jumper locations on the SmartFusion2 Security Evaluation Kit board.

Figure 57 • **SmartFusion2 Security Evaluation Kit Board Jumper Locations**



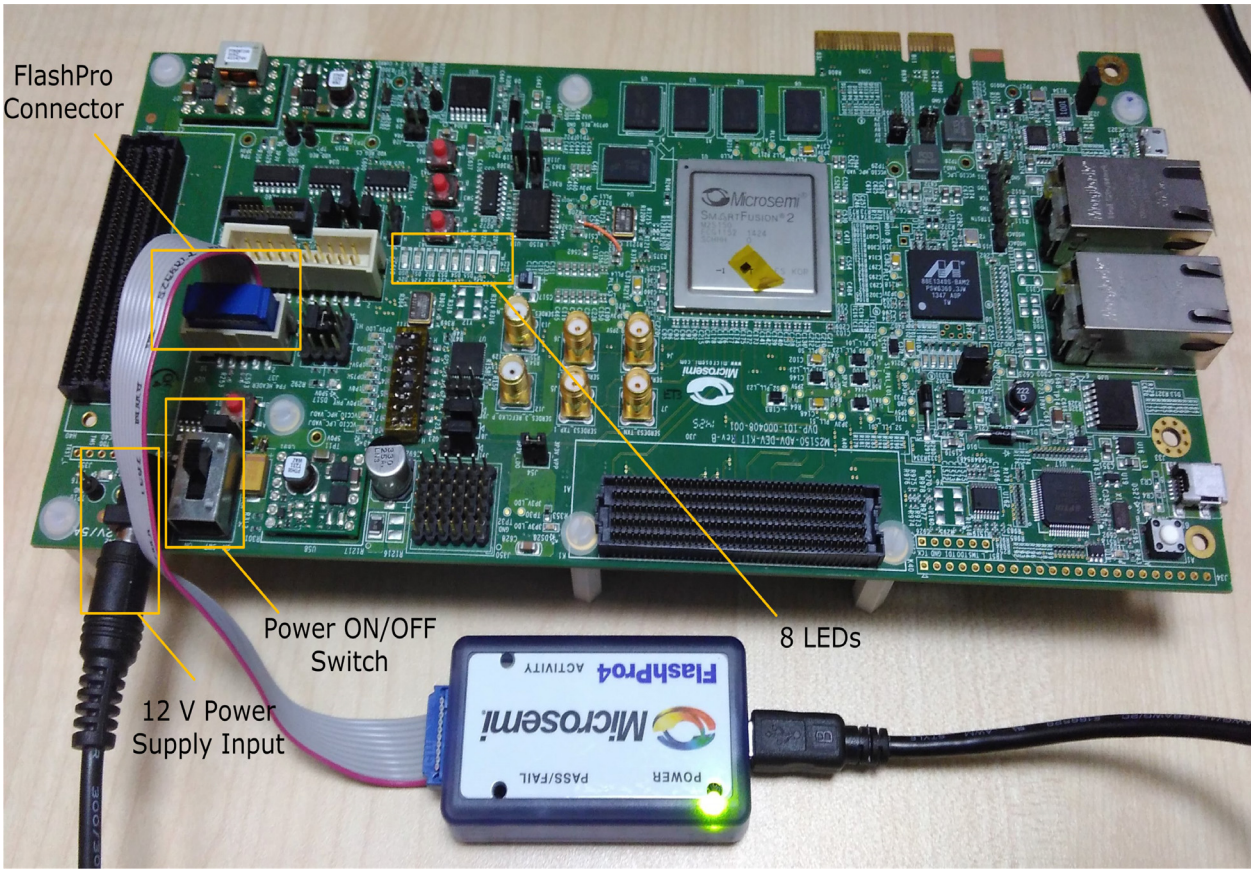
Notes:

- Jumpers highlighted in red are set by default.
- The locations of the jumpers in Figure 57 are searchable.

3 Appendix: Board Setup for SmartFusion2 Advanced Development Kit

Figure 58 shows the board setup for running the demo on the SmartFusion2 Advanced Development Kit board.

Figure 58 • **SmartFusion2 Advanced Development Kit Setup**



1. Connect the jumpers on the SmartFusion2 Advanced Development Kit board as listed in Table 7. For more information on jumper locations, see Figure 59 on page 40 of SmartFusion2 Advanced Development Kit board Jumper Locations.

CAUTION: While making the jumper connections, the **SW7** power supply switch on the board must be in **OFF** position.

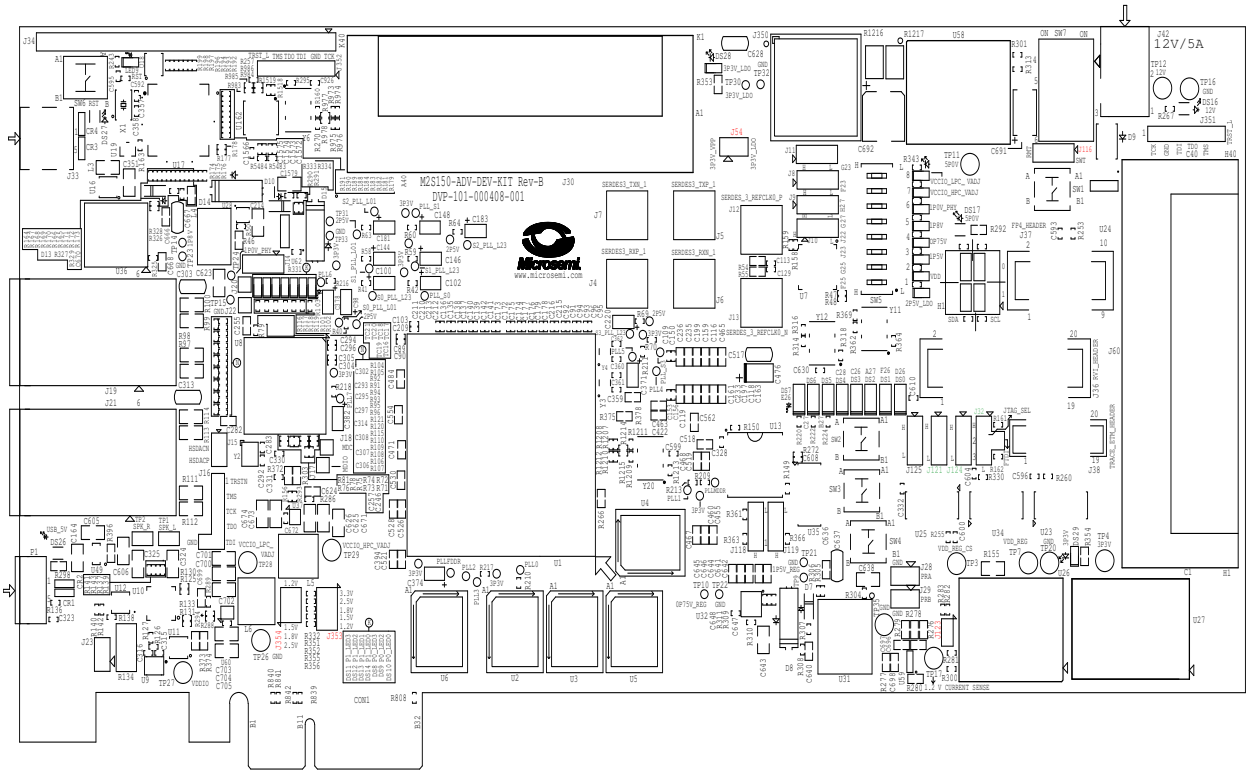
Table 7 • **SmartFusion2 Advanced Development Kit Jumper Settings**

Jumper	Pin (from)	Pin (to)	Comments
J116, J353, J354, and J54	1	2	These are the default jumper settings of the SmartFusion2 advanced development Kit board. Ensure, these jumpers are set accordingly.
J123	2	3	
J124, J121, and J32	2	3	JTAG programming via FTDI

2. Connect the FlashPro4 or FlashPro5 programmer to the **J37** connector of the SmartFusion2 Advanced Development Kit.
3. Connect the power supply to the **J42** connector.
4. Switch **ON** the **SW7** power supply switch.

Figure 59 shows the jumper locations on the SmartFusion2 Advanced Development Kit board.

Figure 59 • **SmartFusion2 Advanced Development Kit Board Jumper Locations**



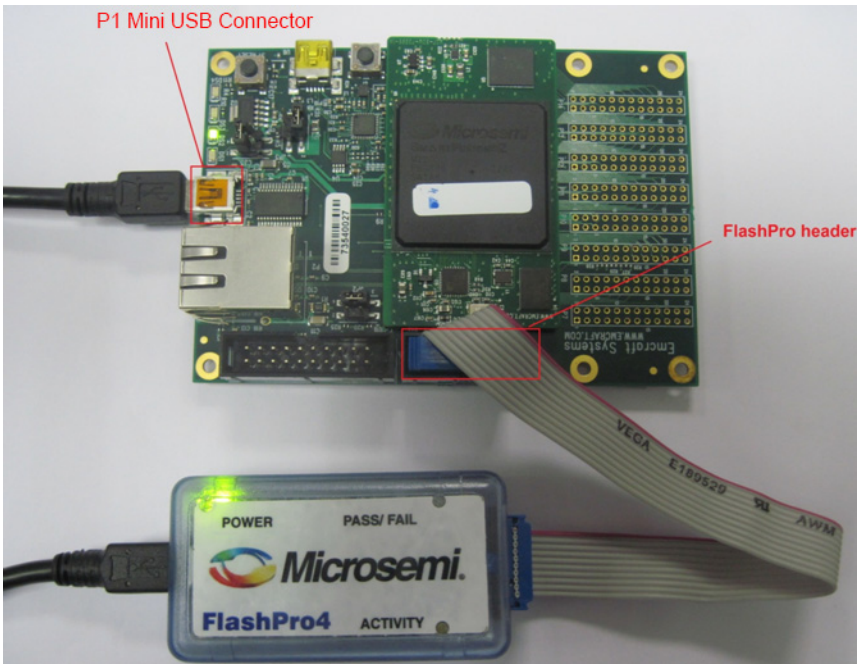
Notes:

- Jumpers highlighted in red are set by default.
- Jumpers highlighted in green must be set manually.
- The locations of the jumpers in Figure 59 are searchable.

4 Appendix: Board Setup for SmartFusion2 Starter Kit

Figure 60 shows the board setup for running the demo on the SmartFusion2 starter Kit board.

Figure 60 • **SmartFusion2 Starter Kit Setup**



1. Connect the jumpers on the SmartFusion2 Starter Kit board as listed in Table 8.

Table 8 • **SmartFusion2 Starter Kit Jumper Settings**

Jumper	Pin (From)	Pin (To)	Comments
JP1	1	2	These are the default jumper settings of SmartFusion2 Starter Kit board. Ensure, these jumpers are set accordingly.
JP2	3	4	
JP3	2	4	

2. Connect the FlashPro4 or FlashPro5 programmer to the **P5** connector of the SmartFusion2 Starter Kit.
3. Connect the host PC USB port to the P1 Mini USB connector on the SmartFusion2 Starter Kit board using the USB Mini-B cable. As soon as the connection to the PC is made, the on-board LED DS2 will illuminate, indicating that the board has power.

5 Revision History

The following table shows important changes made in this document for each revision.

Revision	Changes	Page
Revision 5 (February 2016)	Updated the document for Libero SoC v11.7 software release (SAR 76609).	N/A
Revision 4 (December 2015)	Updated the document for Libero SoC v11.6 and SoftConsole v4.0 software release (SAR 72814)	N/A
Revision 3 (March 2015)	Updated the document for Libero SoC v11.5 software release (SAR 64190).	N/A
Revision 2 (October 2014)	Updated the document for Libero SoC v11.4 software release (SAR 61627).	N/A
Revision 1 (April 2014)	Initial release.	N/A

6 Product Support

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

6.1 Customer Service

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

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

6.2 Customer Technical Support Center

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

6.3 Technical Support

For Microsemi SoC Products Support, visit
http://www.microsemi.com/products/fpga-soc/design-support/fpga-soc-support.

6.4 Website

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

6.5 Contacting the Customer Technical Support Center

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

6.5.1 Email

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

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

6.5.2 My Cases

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

6.5.3 Outside the U.S.

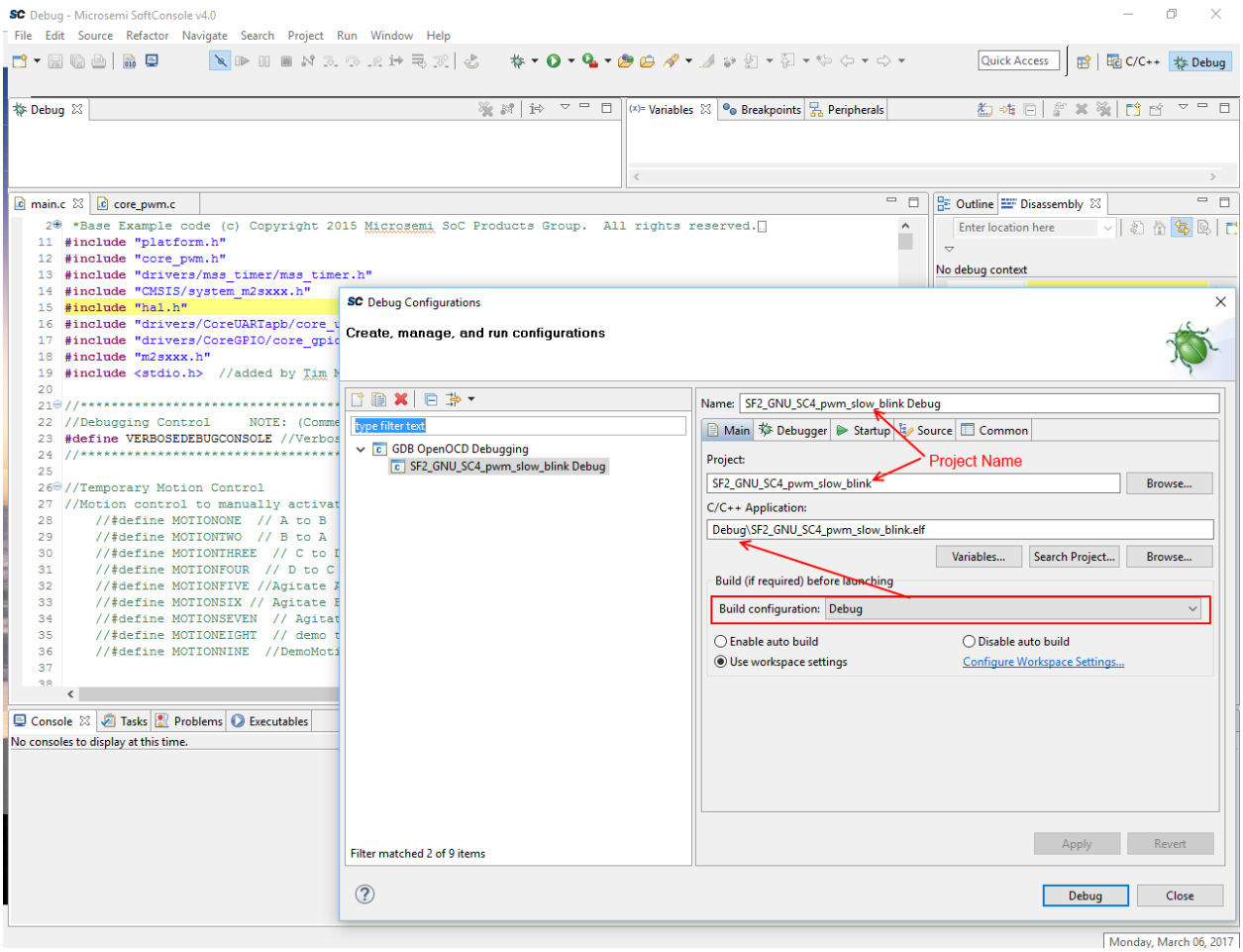
Customers needing assistance outside the US time zones can either contact technical support via email (soc_tech@microsemi.com) or contact a local sales office. Visit About Us for sales office listings and corporate contacts.

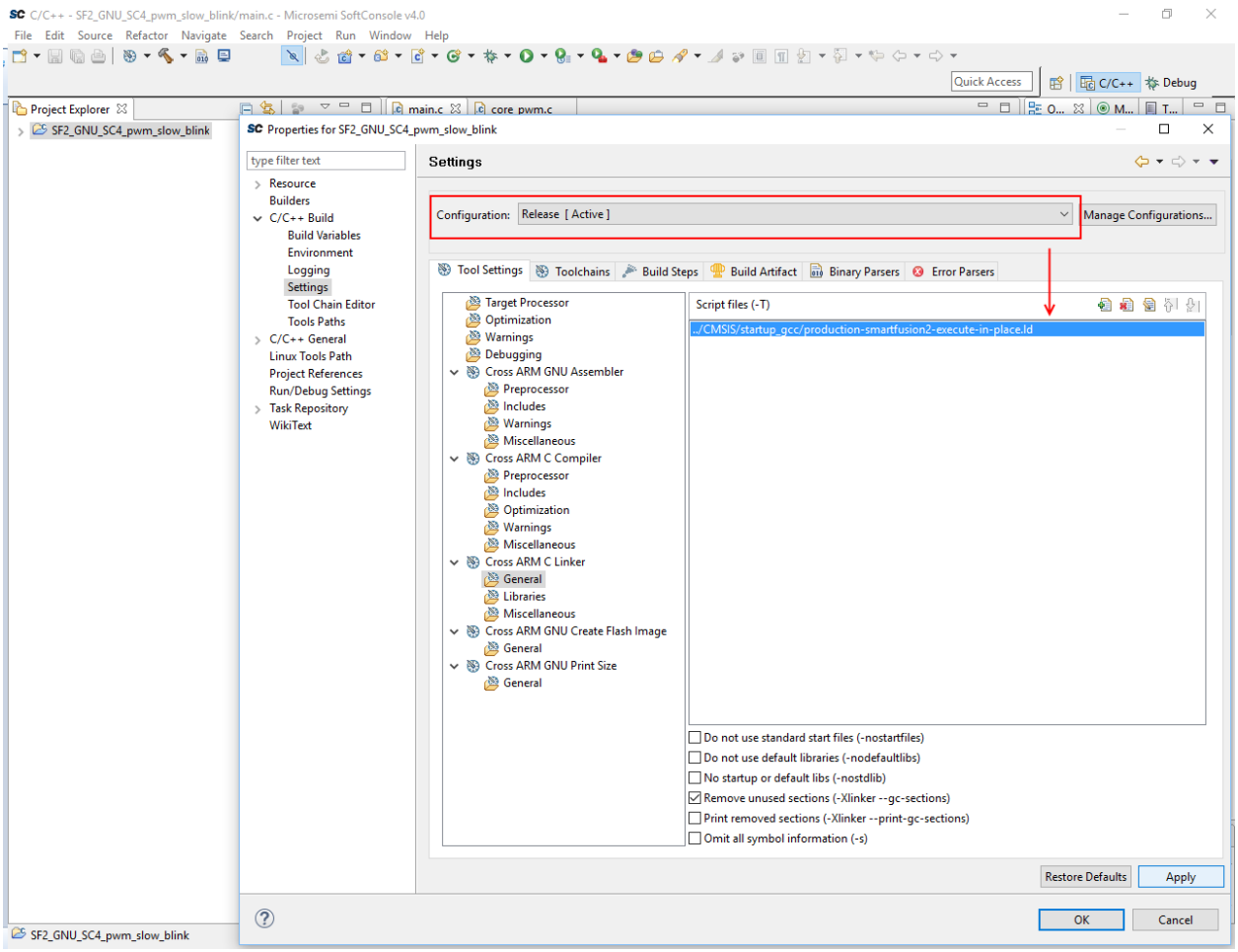
6.6 ITAR Technical Support

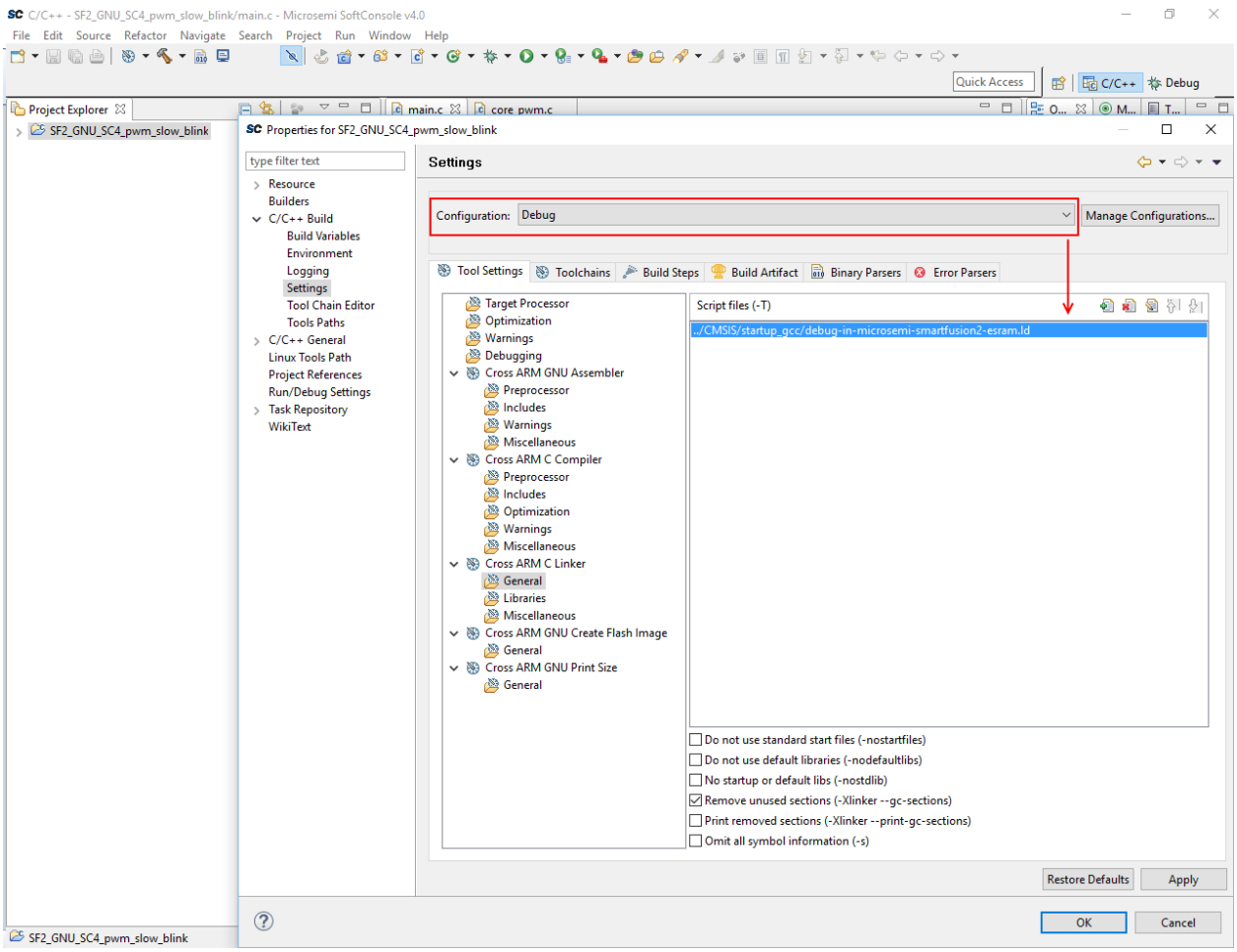
For technical support on RH and RT FPGAs that are regulated by International Traffic in Arms Regulations (ITAR), contact us via soc_tech@microsemi.com. Alternatively, within My Cases, select **Yes** in the ITAR drop-down list. For a complete list of ITAR-regulated Microsemi FPGAs, visit the ITAR web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

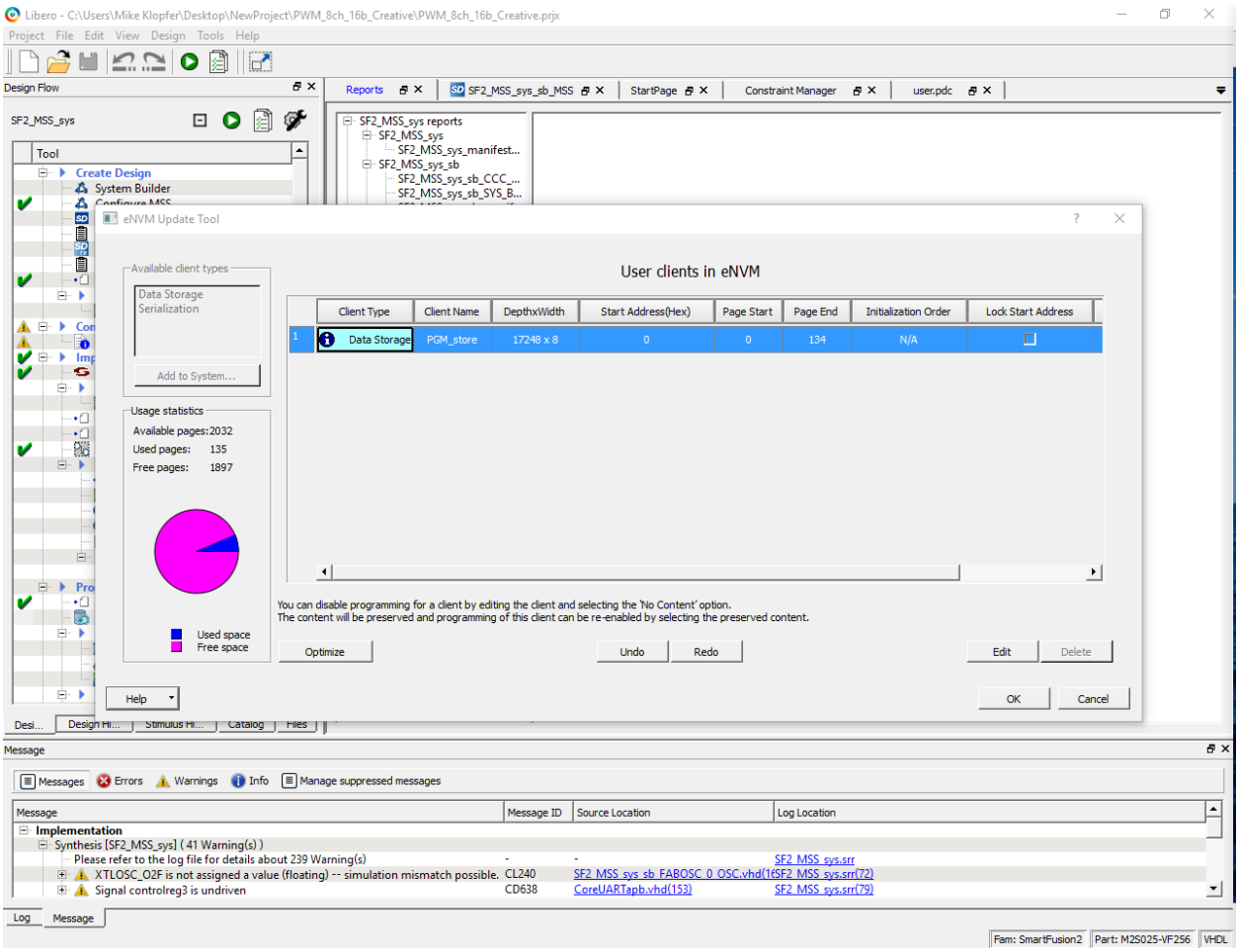
Summary of Softconsole settings for linker scripts and semihosting

Please verify the location (path) for the linker scripts for both the “Production” and “Debug” linker scripts. These are used to map a compiled program to the proper memory location. Without the proper linker script in place for the intended destination, your program will either not load or not execute. Here is an example for the settings for the “Debug” and “Production” linker script and configuration settings in Softconsole.



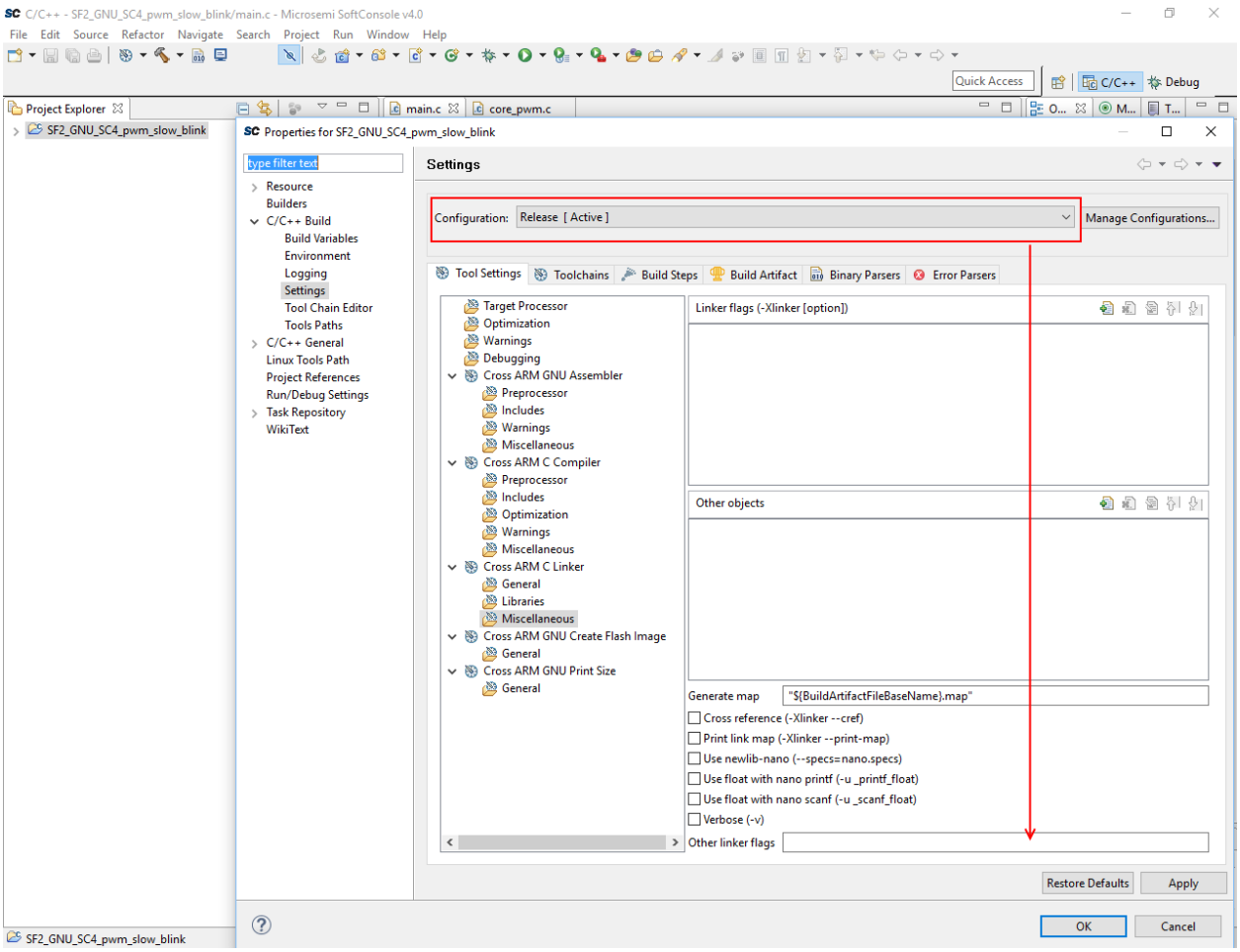

C/C++ - SF2_GNU_SC4_pwm_slow_blink/main.c - Microsemi SoftConsole v4.0
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
C/C++
Debug
Project Explorer
SF2_GNU_SC4_pwm_slow_blink
main.c
core_pwm.c
Properties for SF2_GNU_SC4_pwm_slow_blink
type filter text
Resource
Builders
C/C++ Build
Build Variables
Environment
Logging
Settings
Tool Chain Editor
Tools Paths
C/C++ General
Linux Tools Path
Project References
Run/Debug Settings
Task Repository
WikiText
Settings
Configuration: Release [Active]
Manage Configurations...
Tool Settings
Toolchains
Build Steps
Build Artifact
Binary Parsers
Error Parsers
Target Processor
Optimization
Warnings
Debugging
Cross ARM GNU Assembler
Preprocessor
Includes
Warnings
Miscellaneous
Cross ARM C Compiler
Preprocessor
Includes
Optimization
Warnings
Miscellaneous
Cross ARM C Linker
General
Libraries
Miscellaneous
Cross ARM GNU Create Flash Image
General
Cross ARM GNU Print Size
General
Script files (-T)
../CMSIS/startup_gcc/production-smartfusion2-execute-in-place.ld
Do not use standard start files (-nostartfiles)
Do not use default libraries (-nodefaultlibs)
No startup or default libs (-nostdlib)
Remove unused sections (-Xlinker --gc-sections)
Print removed sections (-Xlinker --print-gc-sections)
Omit all symbol information (-s)
Restore Defaults
Apply
OK
Cancel
SF2_GNU_SC4_pwm_slow_blink


C/C++ - SF2_GNU_SC4_pwm_slow_blink/main.c - Microsemi SoftConsole v4.0
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
C/C++
Debug
Project Explorer
SF2_GNU_SC4_pwm_slow_blink
main.c
core_pwm.c
Properties for SF2_GNU_SC4_pwm_slow_blink
type filter text
Resource
Builders
C/C++ Build
Build Variables
Environment
Logging
Settings
Tool Chain Editor
Tools Paths
C/C++ General
Linux Tools Path
Project References
Run/Debug Settings
Task Repository
WikiText
Settings
Configuration: Debug
Manage Configurations...
Tool Settings
Toolchains
Build Steps
Build Artifact
Binary Parsers
Error Parsers
Target Processor
Optimization
Warnings
Debugging
Cross ARM GNU Assembler
Preprocessor
Includes
Warnings
Miscellaneous
Cross ARM C Compiler
Preprocessor
Includes
Optimization
Warnings
Miscellaneous
Cross ARM C Linker
General
Libraries
Miscellaneous
Cross ARM GNU Create Flash Image
General
Cross ARM GNU Print Size
General
Script files (-T)
../CMSIS/startup_gcc/debug-in-microsemi-smartfusion2-esram.ld
Do not use standard start files (-nostartfiles)
Do not use default libraries (-nodefaultlibs)
No startup or default libs (-nostdlib)
Remove unused sections (-Xlinker --gc-sections)
Print removed sections (-Xlinker --print-gc-sections)
Omit all symbol information (-s)
Restore Defaults
Apply
OK
Cancel
SF2_GNU_SC4_pwm_slow_blink

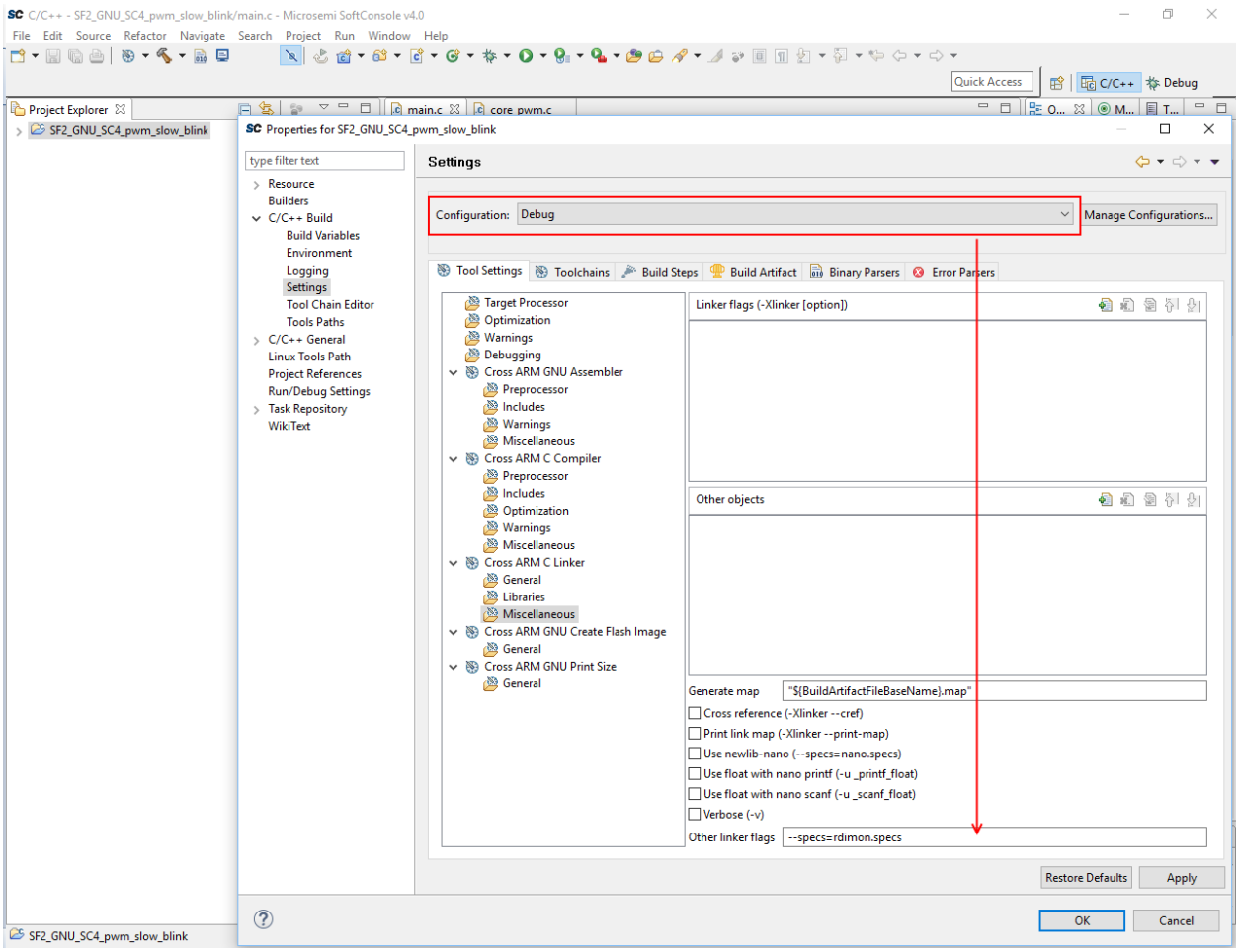


(eNVM Memory update tool where the “Production” build is loaded to be flashed onto the SF2 – notice the image size, and memory location)

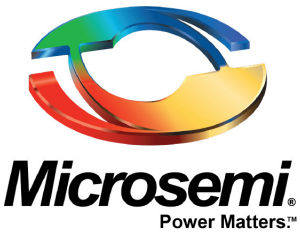
Semihosting Settings – this must be enabled to make semihosting work. Semihosting allows messages to be passed to the console in operation via Open OCD. This is only useful and should be done for the Debug configuration, not the Release configuration loaded onto the eNVM with Libero. Commonly you use a #define and #ifdefined to turn on this "debug" configuration – Comment out the #define to disable Semihosting for production builds. Details: https://mcuoneclipse.com/2014/09/11/semihosting-with-gnu-arm-embedded-launchpad-and-gnu-arm-eclipse-debug-plugins/



(Before)


C/C++ - SF2_GNU_SC4_pwm_slow_blink/main.c - Microsemi SoftConsole v4.0
File Edit Source Refactor Navigate Search Project Run Window Help
Quick Access
C/C++
Debug
Project Explorer
SF2_GNU_SC4_pwm_slow_blink
main.c
core_pwm.c
Properties for SF2_GNU_SC4_pwm_slow_blink
type filter text
Resource
Builders
C/C++ Build
Build Variables
Environment
Logging
Settings
Tool Chain Editor
Tools Paths
C/C++ General
Linux Tools Path
Project References
Run/Debug Settings
Task Repository
WikiText
Settings
Configuration: Debug
Manage Configurations...
Tool Settings
Toolchains
Build Steps
Build Artifact
Binary Parsers
Error Parsers
Target Processor
Optimization
Warnings
Debugging
Cross ARM GNU Assembler
Preprocessor
Includes
Warnings
Miscellaneous
Cross ARM C Compiler
Preprocessor
Includes
Optimization
Warnings
Miscellaneous
Cross ARM C Linker
General
Libraries
Miscellaneous
Cross ARM GNU Create Flash Image
General
Cross ARM GNU Print Size
General
Linker flags (-Xlinker [option])
Other objects
Generate map
"${BuildArtifactFileBaseName}.map"
Cross reference (-Xlinker --cref)
Print link map (-Xlinker --print-map)
Use newlib-nano (--specs=nano.specs)
Use float with nano printf (-u _printf_float)
Use float with nano scanf (-u _scanf_float)
Verbose (-v)
Other linker flags
--specs=rdimon.specs
Restore Defaults
Apply
OK
Cancel
SF2_GNU_SC4_pwm_slow_blink

(After)



Microsemi Corporate Headquarters
One Enterprise, Aliso Viejo,
CA 92656 USA

Within the USA: +1 (800) 713-4113
Outside the USA: +1 (949) 380-6100
Sales: +1 (949) 380-6136
Fax: +1 (949) 215-4996
E-mail: sales.support@microsemi.com

Microsemi Corporation (Nasdaq: MSCC) offers a comprehensive portfolio of semiconductor and system solutions for communications, defense & security, aerospace and industrial markets. Products include high-performance and radiation-hardened analog mixed-signal integrated circuits, FPGAs, SoCs and ASICs; power management products; timing and synchronization devices and precise time solutions, setting the world's standard for time; voice processing devices; RF solutions; discrete components; Enterprise Storage and Communication solutions, security technologies and scalable anti-tamper products; Ethernet solutions; Power-over-Ethernet ICs and midspans; as well as custom design capabilities and services. Microsemi is headquartered in Aliso Viejo, Calif, and has approximately 4,800 employees globally. Learn more at **www.microsemi.com**.

50200546-5/02.16