
Using UART with SmartFusion cSoC

SoftConsole Standalone Flow Tutorial



Table of Contents

Introduction	3
Tutorial Requirements	4
Working with the Softconsole	5
Step 1 - Launching the SoftConsole	5
Step 2 - Configuring MSS Peripherals	6
Step 3 - Programming SmartFusion Board Using FlashPro	11
Step 4 - Building the Project	12
Step 5 - Configuring Serial Terminal Emulation Program	15
Step 6 - Installing Drivers for the USB to RS232 Bridge	18
Step 7 - Debugging the Application Using SoftConsole	18
Step 8 - Debugging the Application Using the printf Statement in SoftConsole	22
Step 9 - Building Executable Image in Release mode	26
Appendix A – Libero SoC Vault/Repository Settings	27
Appendix B – Firmware Catalog Settings	29
List of Changes	31
Product Support	33
Customer Service	33
Customer Technical Support Center	33
Technical Support	33
Website	33
Contacting the Customer Technical Support Center	33
ITAR Technical Support	34

Introduction

This tutorial describes the use of Microsemi tools to configure the SmartFusion® customizable system-on-chip (cSoC) microcontroller subsystem (MSS) peripherals using the SmartFusion cSoC MSS configurator outside the Libero® system-on-chip (SoC) v10.0 flow.

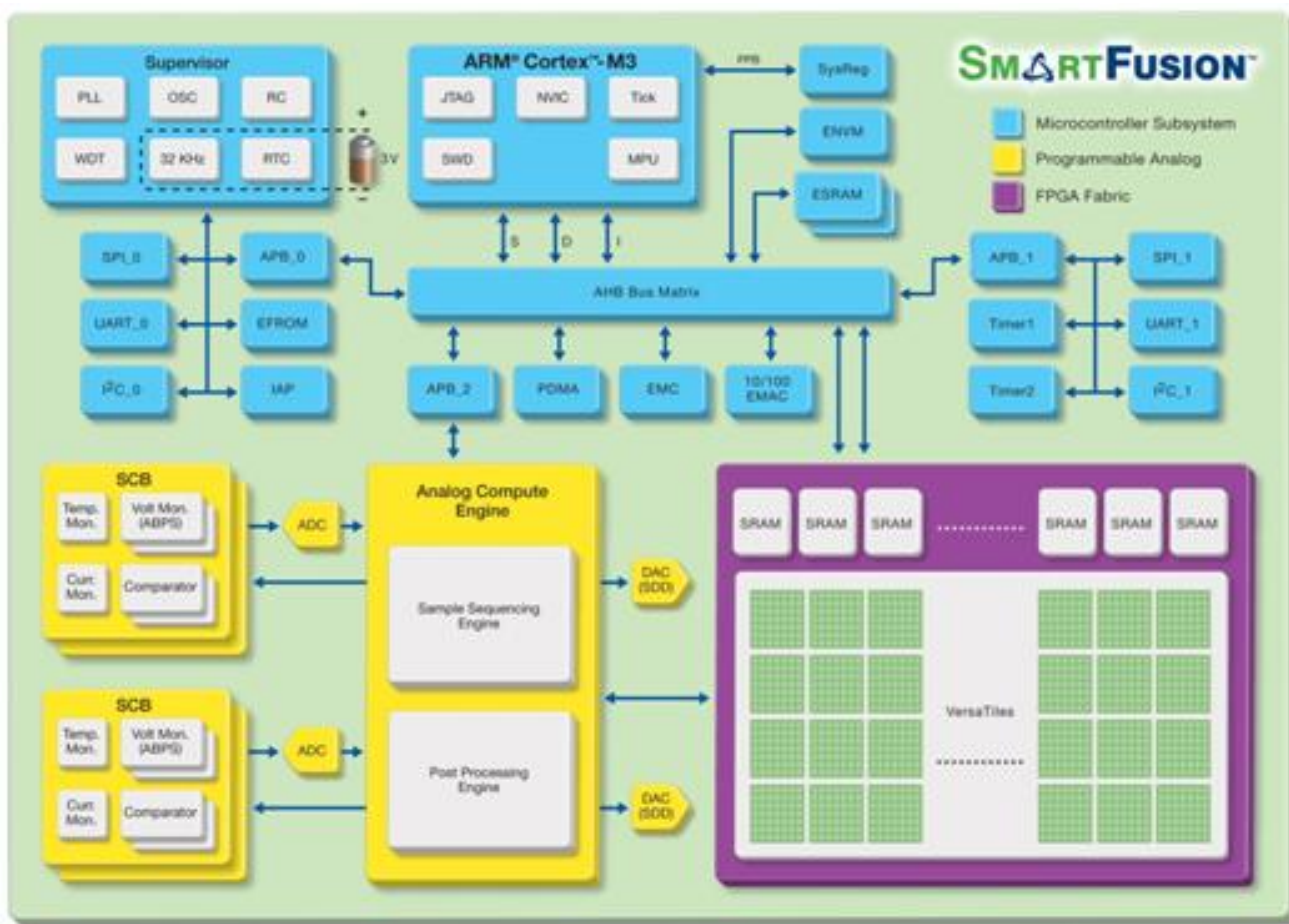


Figure 1 · SmartFusion cSoC Block Diagram

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

- Configuring MSS Configurator.
- Configuring the SmartFusion cSoC MSS
- Generating the programming file on the SmartFusion Kit Board
- Configuring the HyperTerminal program
- Compiling application code
- Creating and launching a debug session
- Debugging the application using the printf statement in SoftConsole

Tutorial Requirements

Software Requirements

This tutorial requires the following software installed on your PC:

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

Hardware Requirements

This tutorial requires the following hardware:

- SmartFusion Evaluation Kit Board or SmartFusion Development Kit Board
- Two USB cables (programming and communication) — One for connecting the programmer to your PC and the other to connect the universal asynchronous receiver/transmitter (UART) interface on the board to PC.

Associated Project Files

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

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

MSS Components Used

- ARM® Cortex™-M3 processor
- Communications matrix
- Clock conditioning circuit (CCC)
- UART

Target Board

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

Objective

The objective of this tutorial is to instruct how to configure the SmartFusion cSoC MSS peripherals using the SmartFusion cSoC MSS configurator outside the Libero SoC v10.0 flow.

Design Steps

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

- Invoke MSS Configurator from SoftConsole.
- Configure the SmartFusion peripherals using SmartFusion cSoC MSS configurator.
- Generate the programming file and program the SmartFusion cSoC using FlashPro.
- Debug the application using SoftConsole.

Working with the Softconsole

Step 1 - Launching the SoftConsole

1. Click **Start > Programs > Microsemi SoftConsole v3.3 > Microsemi SoftConsole IDE**.

The SoftConsole Workspace Launcher is displayed, as shown in Figure 2 :

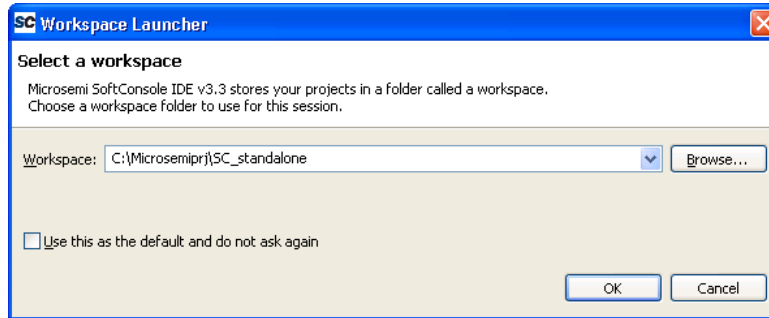


Figure 2 · Specifying the SoftConsole Workspace

2. Click **Browse** on the SoftConsole workspace launcher to navigate to your preferred location (where you will create your project), or enter the location of your workspace. For example, C:\Microsemiprj\SC_standalone.

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

3. Click **OK**. The SoftConsole window is displayed.

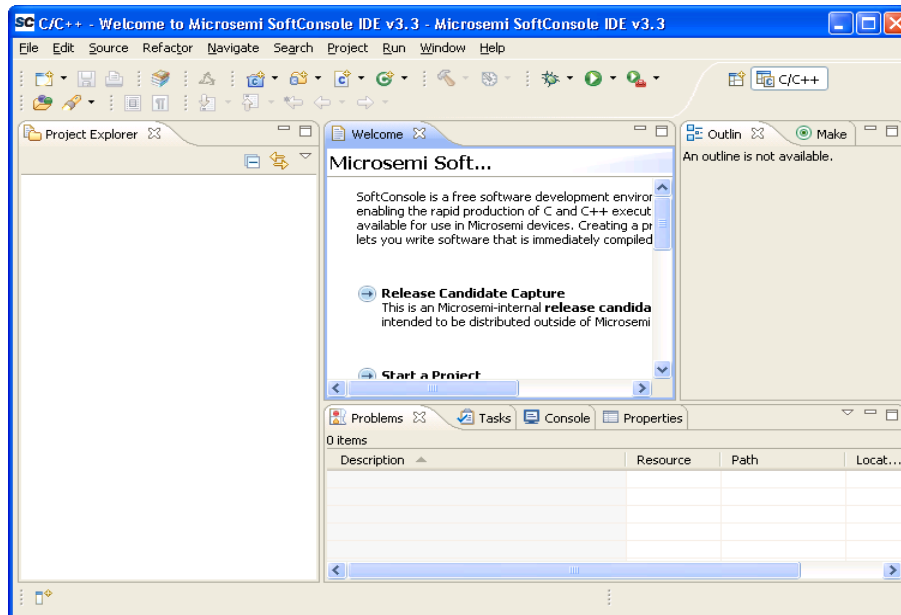


Figure 3 · The SoftConsole GUI

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

Step 2 - Configuring MSS Peripherals

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

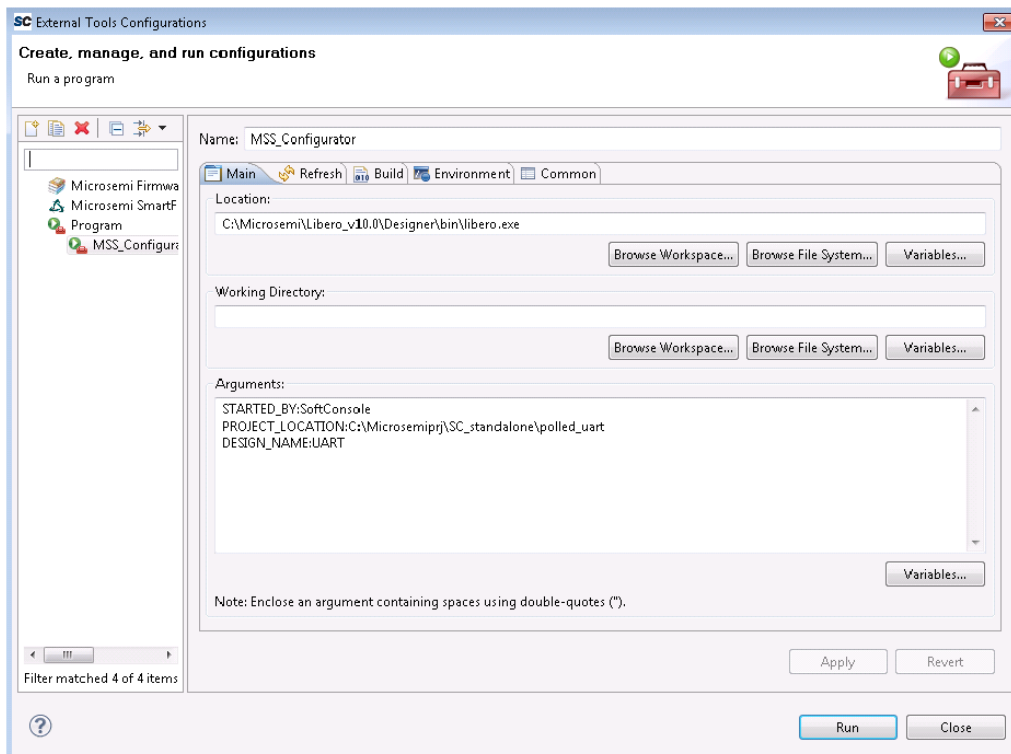
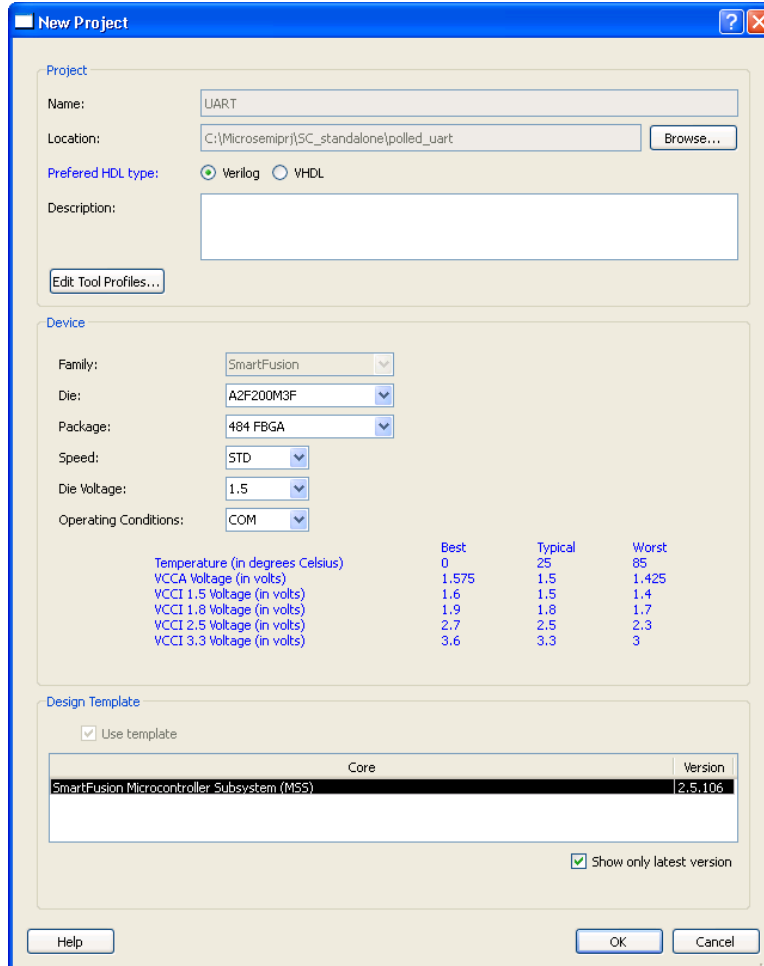


Figure 4 · Running SmartFusion MSS Configurator

8. The Libero SoC is displayed with a **New Project** dialog box, as shown in [Figure 5](#) · . Select:
 - Die: If you are using SmartFusion Evaluation Kit Board, enter A2F200M3F; if you are using SmartFusion Development Kit Board, enter A2F500M3F.
 - Package: 484 FBGA
 - Speed: STD
 - Die Voltage: 1.5

9. The **Project Name** and **Project Directory** parameters are already specified. Click **OK**.



The 'New Project' dialog window is shown with the following settings:

- Project:**
 - Name: UART
 - Location: C:\Microsemi\prj\SC_standalone\polled_uart (with a 'Browse...' button)
 - Preferred HDL type: Verilog (selected), VHDL
 - Description: (empty text box)
 - Edit Tool Profiles... (button)
- Device:**
 - Family: SmartFusion
 - Die: A2F200M3F
 - Package: 484 FBGA
 - Speed: STD
 - Die Voltage: 1.5
 - Operating Conditions: COM

	Best	Typical	Worst
Temperature (in degrees Celsius)	0	25	85
VCCA Voltage (in volts)	1.575	1.5	1.425
VCCI 1.5 Voltage (in volts)	1.6	1.5	1.4
VCCI 1.8 Voltage (in volts)	1.9	1.8	1.7
VCCI 2.5 Voltage (in volts)	2.7	2.5	2.3
VCCI 3.3 Voltage (in volts)	3.6	3.3	3
- Design Template:**
 - Use template: ☒
 - Table:

Core	Version
SmartFusion Microcontroller Subsystem (MSS)	2.5.106
 - Show only latest version: ☒

Buttons: Help, OK, Cancel

Figure 5 · New Project Dialog Window

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

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

The MSS canvas appears as shown in Figure 6 :

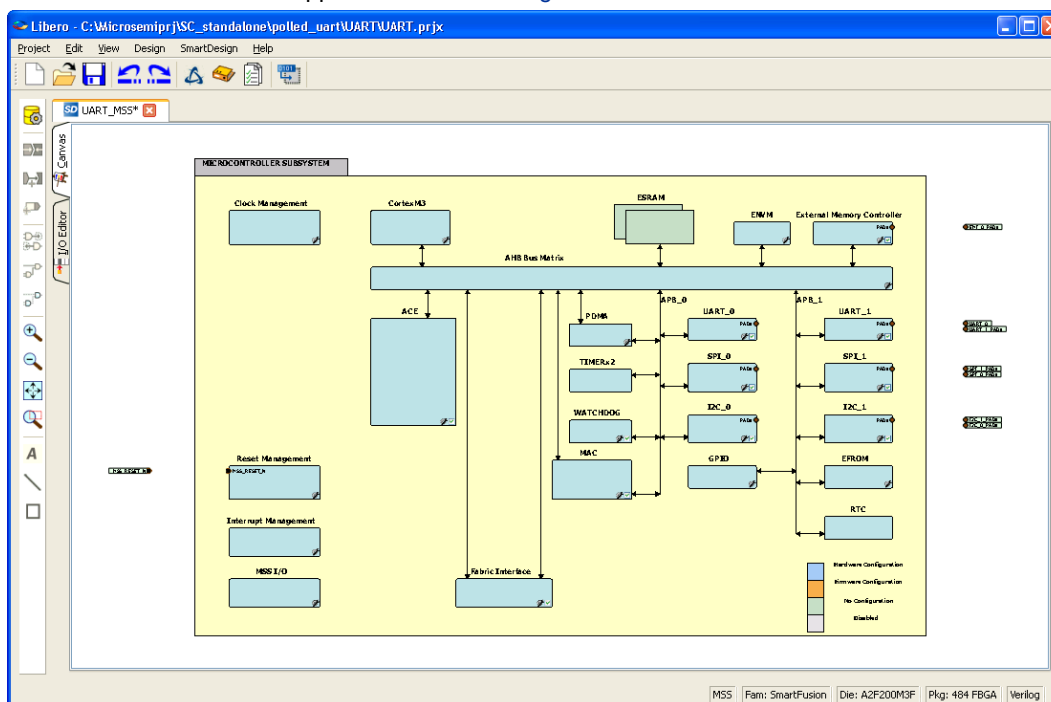


Figure 6 · MSS in the SmartDesign Canvas

The enabled MSS peripherals are highlighted in blue, and can be configured in hardware. The disabled peripherals are shown in gray.

To disable a peripheral that is not required, select the peripheral, right-click, and clear the Enabled check box, in the lower right corner of the peripheral box. The box turns grey to indicate a peripheral has been disabled. Disabled peripherals can be enabled by repeating the procedure.

An enabled peripheral looks as shown in Figure 7 · .

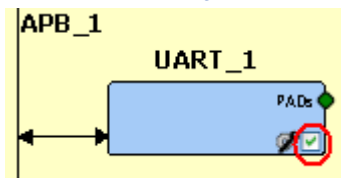


Figure 7 · Enabling a Peripheral

For this design we will be using the ACE, UART_0, and clock management peripherals.

For this tutorial, disable the following peripherals: analog compute engine (ACE), MAC, EMC, fabric interface, UART_1, SPI_0, SPI_1, I2C_0, and I2C_1.

Configure the MSS CCC by double-clicking the Clock Management block. Configure the clocks as shown below.

- CLKA: On-chip RC Oscillator
- MSS clock source: PLL output (VCO output)
- MSS clock frequency: 80 MHz

Use default settings for all other fields. After completing the configuration, click **OK**.

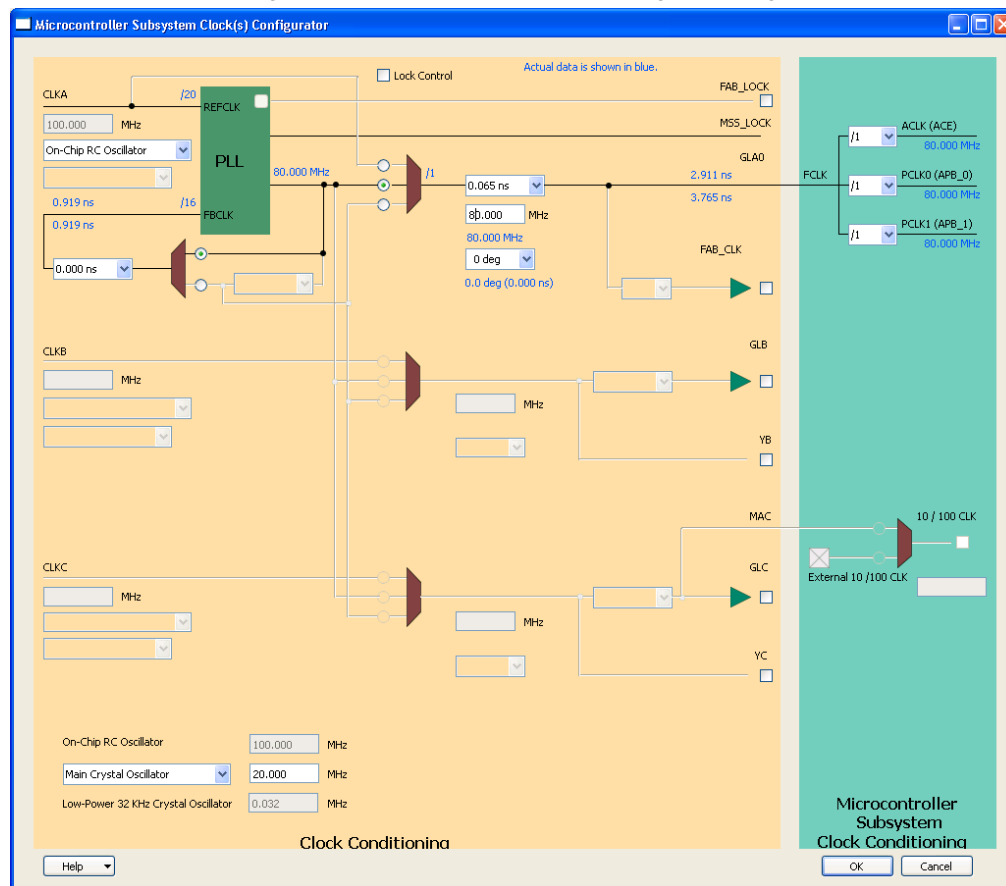


Figure 8 · MSS Clock Configuration

10. Click **Project > Save** to save **SmartFusion_UART_MSS**.

Generating the MSS Component

1. Click **Design > Configure Firmware** as shown in Figure 9 · .



Figure 9 · Firmware Tab

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

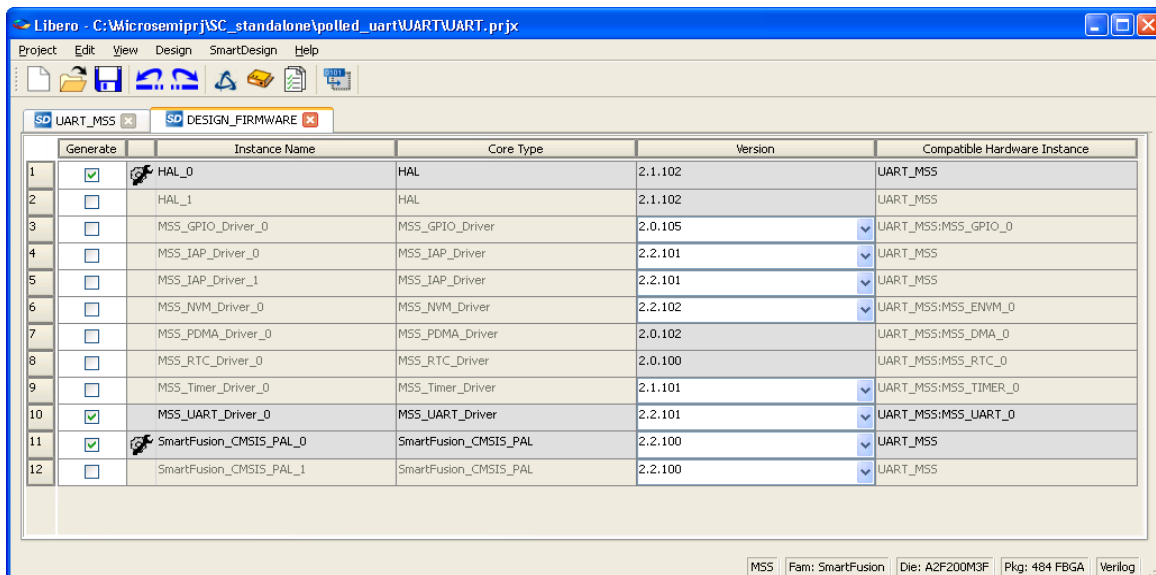


Figure 10 · Configuring SmartFusion_CMSIS_PAL_0

Note: Check whether or not you are able to see the latest version of the drivers without any warning or error indicating that firmware is missing from the Vault. If missing, refer to [Appendix B – Firmware Catalog Settings](#) to set your repositories. If you are opening the SmartDesign for the first time then you need to download the all firmware drivers from the Firmware tab in the SmartDesign.

- Click **Project > Save** to save the **DESIGN_FIRMWARE**.
- Save the design and generate the MSS component by clicking **Generate Component** or click **SmartDesign > Generate Component**.

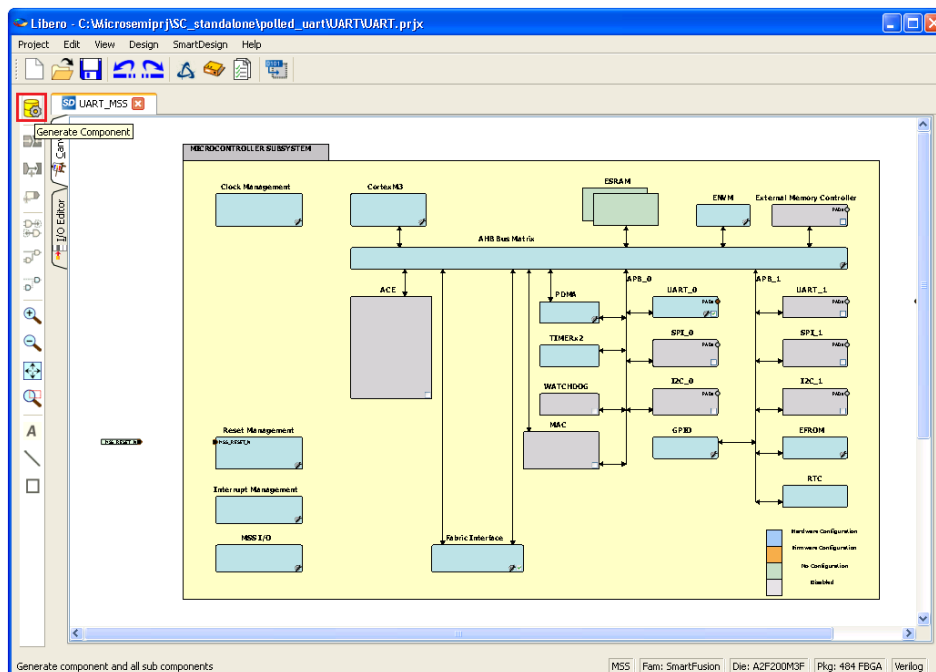


Figure 11 · Select SoftConsole in the Configuration Window

Confirm that the design was successfully generated.

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

Step 3 - Programming SmartFusion Board Using FlashPro

Jumper Settings for SmartFusion Evaluation Kit Board

Before you proceed with programming the device, ensure that LCPS or FlashPro4 is properly connected to the board. Use the following details to ensure the correct jumper settings.

Refer to the [SmartFusion Evaluation Kit User's Guide](#) and [SmartFusion Development Kit User's Guide](#) for additional information.

- JP10: Connect pin 1 and 2.
- JP7: Connect pin 1 and 2 for LCPS programming mode.
- J6: Connect pins 1 and 2 with the jumper.
- JP6: Connect pins 2 and 3 with the jumper.
- J13: Connect USB cable to the J13 connector. When the cable is connected, the FlashPro drivers might be installed if they are not already installed.
- J14: Connect one end of USB mini B cable to J14.

Jumper Settings for SmartFusion Development Kit

SW9 must be off (JTAGSEL = H) in order to program the SmartFusion device. SW9 remains in the off position for Libero SoC and SoftConsole programming. Make the jumper settings as shown in Table 1:

Table 1 · Jumper Settings for Development Kit Board

Factory Default	Factory Default	Factory Default
JP1: 1–2	JP12: 1–2	JP21: 1–2
JP2: 1–2	JP13: 1–2	JP22: 2–3
JP4: 1–3; 7–9	JP14: 1–2	JP23: 1–2
JP5: 1–2; 3–4	JP15: 1–2	JP24: 1–2
JP6: 2–3	JP16: 2–3	JP27: 1–2
J7: 2–3; 6–7; 10–11; 14–15	JP17: 2–3	JP28: 1–2
JP7: 1–2	JP18: 1–2	J32: 1–2; 3–4; 5–6
JP8: 3–4; 7–8; 11–12; 15–16	JP19: 2–3	–
JP11: 1–2	JP20: 1–2	–

Programming the Device

1. Click **Program Device** as shown in [Figure 12](#) · or select **Design > Program Device**.



Figure 12 · Select SoftConsole in the Configuration Window

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

2. Click **Yes** when it prompts **I/O and Timing constraints not set**
3. Close **Libero (Project > Exit)**.

Step 4 - Building the Project

1. Invoke the SoftConsole project generated by Libero SoC v10.0 using **File > Switch Workspace > Other...** and browse to the SoftConsole workspace as shown in [Figure 13](#) . .

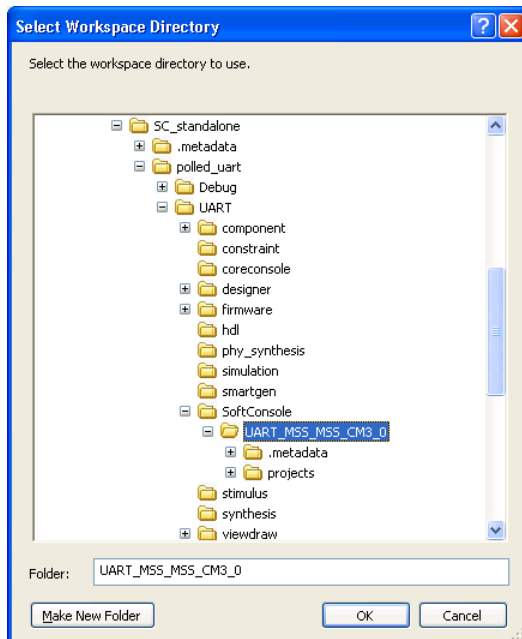


Figure 13 · Specifying the Workspace

The SoftConsole window looks as shown in [Figure 14](#) . .

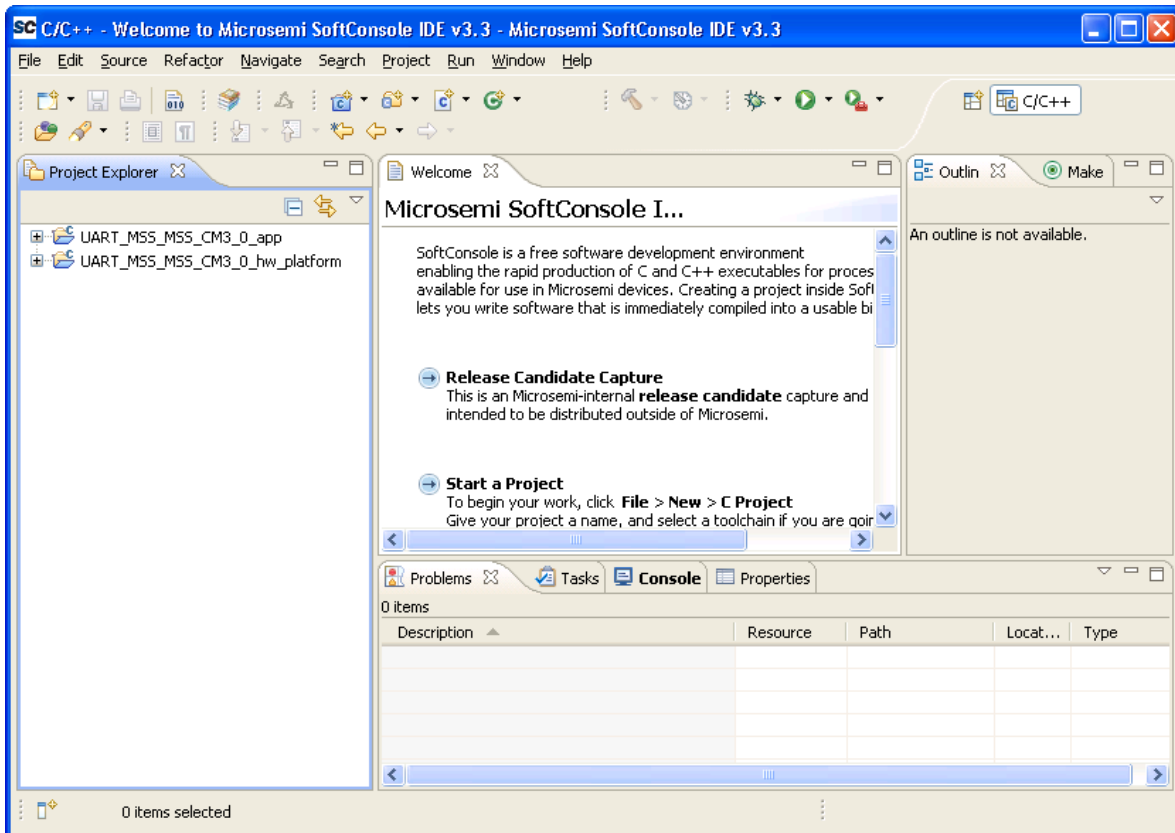


Figure 14 · SoftConsole Window

Enter the code provided below in the `main.c` file in the `UART_MSS_MSS_CM3_0_app` project.

```
#include "mss_uart.h"
#define Microsemi_logo \
"\n\r \
**      **  *****      *****      *****      *****      *****      **      **  *****      \n\r \
* *      *      *      *      *      *      *      *      *      *      *      *      *      *      *      \n\r \
* * *      *      *      *****      *      *      *****      *      *      *      *      *      *      \n\r \
* * *      *      *      *      *      *      *      *      *      *      *      *      *      *      *      \n\r \
*      *      *****      *****      *      *      *****      *****      *      *      *****      "

/* Main function */
int main()
{
    const uint8_t greeting[] = "\n\rWelcome to SmartFusion "
        "- customizable system-on-chip (cSoC) ";

    /* Initialize and configure UART0. */
    MSS_UART_init
    (
        &g_mss_uart0,
        MSS_UART_57600_BAUD,
        MSS_UART_DATA_8_BITS | MSS_UART_NO_PARITY | MSS_UART_ONE_STOP_BIT
    );

    /* Send the Microsemi Logo over the UART_0 */
}
```

```

MSS_UART_polled_tx_string( &g_mss_uart0, (const uint8_t *)Microsemi_logo);

/* Send greeting message over the UART_0 */
MSS_UART_polled_tx( &g_mss_uart0, greeting, sizeof(greeting) );

while(1)
{
}

return 0;
}

```

The SoftConsole Workspace looks as shown in Figure 15 . .

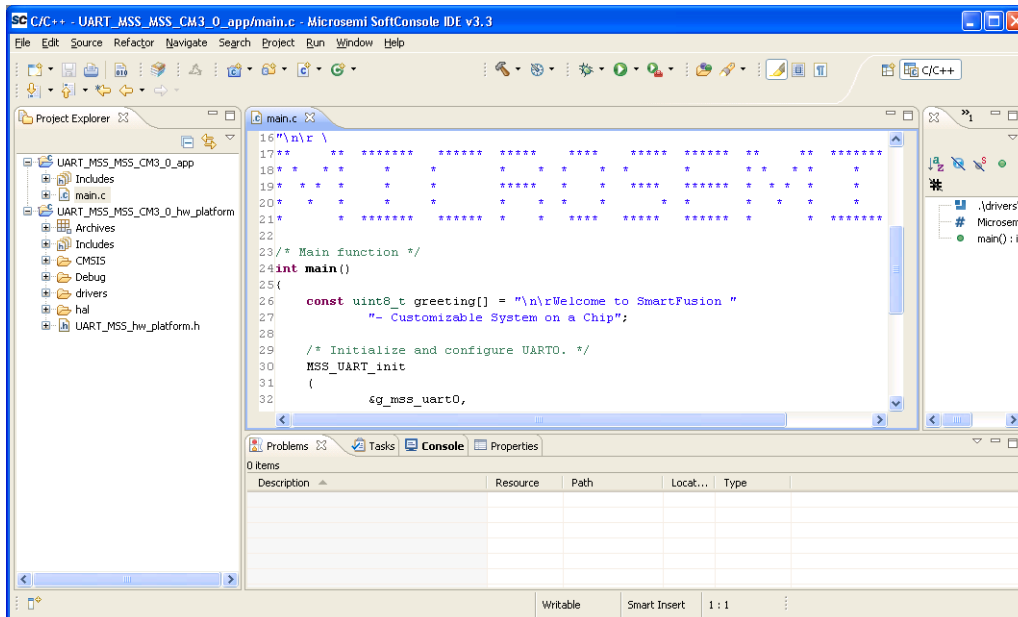


Figure 15 · SoftConsole Workspace

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

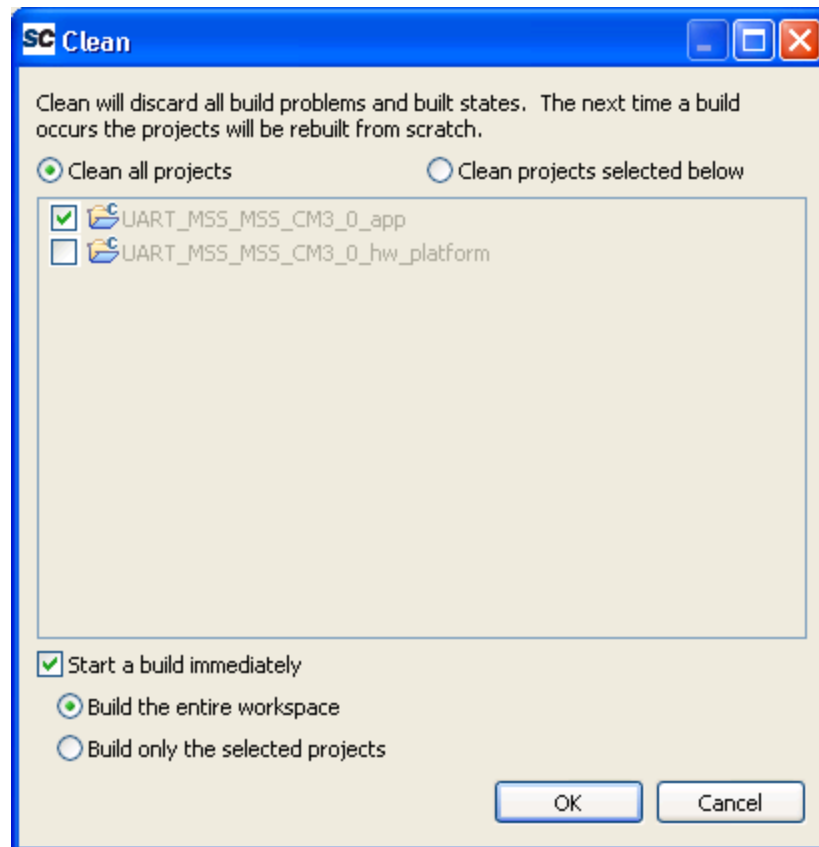


Figure 16 · Setting for a Clean Build

Make sure there are no errors and warnings.

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

Step 5 - Configuring Serial Terminal Emulation Program

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

1. Connect a second mini USB cable between the USB connector on the SmartFusion Evaluation Kit Board (or the SmartFusion Development Kit Board) and a USB port of your PC. If Windows prompts you to connect to Windows Update, select No, not at this time and click Next.

2. If the Silicon Labs CP210x USB to UART Bridge drivers are automatically detected (this can be verified in Device Manager), as shown in [Figure 17](#) . . Proceed to next step; otherwise follow [Step 6 - Installing Drivers for the USB to RS232 Bridge](#).

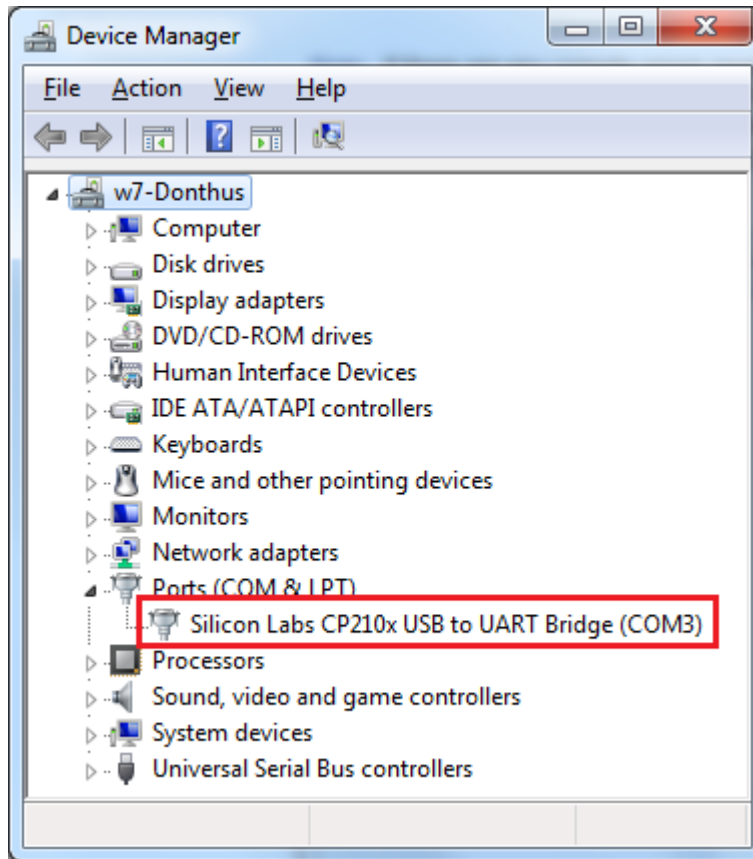


Figure 17 · Device Manager Listing Silicon Labs CP210x USB to UART Bridge

3. From the Windows **Start** menu, select **Programs > Accessories > Communications > HyperTerminal**. This opens HyperTerminal. If your PC does not have HyperTerminal, use any free serial terminal emulation program like PuTTY or Tera Term. Refer to the [Configuring Serial Terminal Emulation Programs](#) tutorial for configuring the HyperTerminal, Tera Term, and PuTTY.
4. Enter **Hyperterminal** in the **Name** field in the **Connection Description** dialog box and click **OK**.

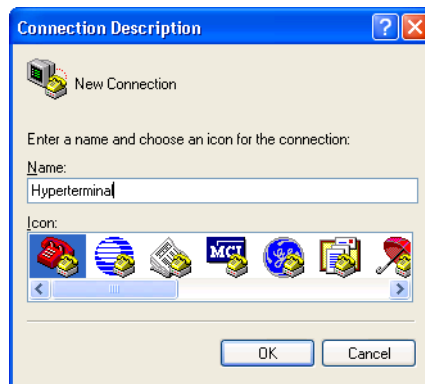


Figure 18 · New Connection

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

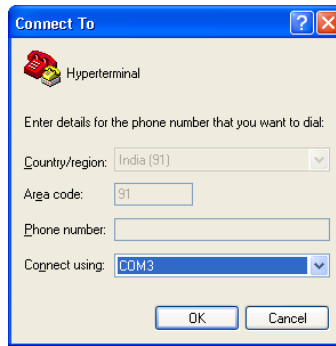


Figure 19 · Selecting the COM Port

6. Set the following in the **COM Properties** window and click **OK**:
 - Bits per second: 57600
 - Data bits: 8
 - Parity: None
 - Stop Bits: 1
 - Flow control: None

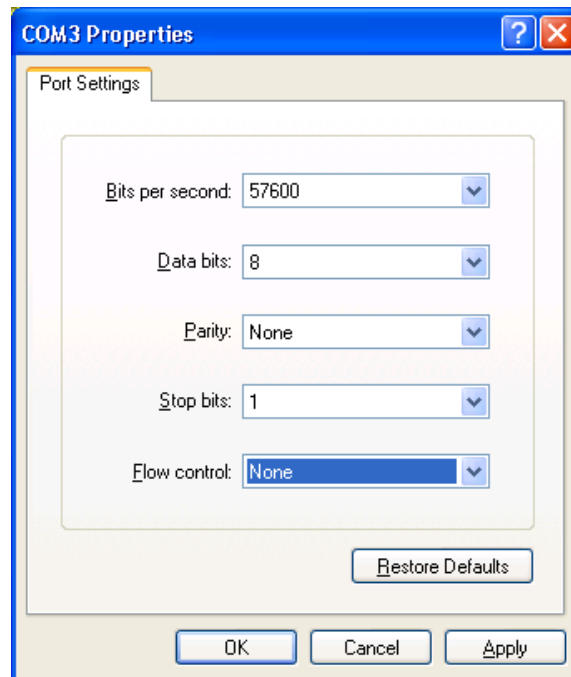


Figure 20 · Setting the COM Properties

7. Click **OK** to close the Hyperterminal Properties dialog box.
Next time you can directly open HyperTerminal (without configuring) by selecting, **Programs > Accessories > Communications > HyperTerminal > Hyperterminal**.

Step 6 - Installing Drivers for the USB to RS232 Bridge

Note: To install the USB-RS232 drivers, you should have administrative privileges for your PC.
Use the following steps to install drivers for the USB to RS232 Bridge:

1. 1. Download the USB to RS232 bridge drivers from www.microsemi.com/soc/documents/CP2102_driver.zip.
2. 2. Unzip the CP2102_driver.zip file.
3. 3. Double-click (Run) the CP210x_VCP_Win_XP_S2K3_Vista_7.exe file.
4. 4. Accept the default installation location and click Install.
5. 5. Click Continue Anyway if prompted.
6. 6. When the installation is complete, click OK. The Ports (COM & LPT) section of the Device Manager lists Silicon Labs CP210x USB to UART Bridge under the Ports section of Device Manager.

Step 7 - Debugging the Application Using SoftConsole

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

1. Select **Debug Configurations** from the **Run** menu of the SoftConsole. The Debug dialog is displayed.
2. Double-click on **Microsemi Cortex-M3 RAM target**. The **Debug Configurations** window is displayed, as shown in Figure 21 . .

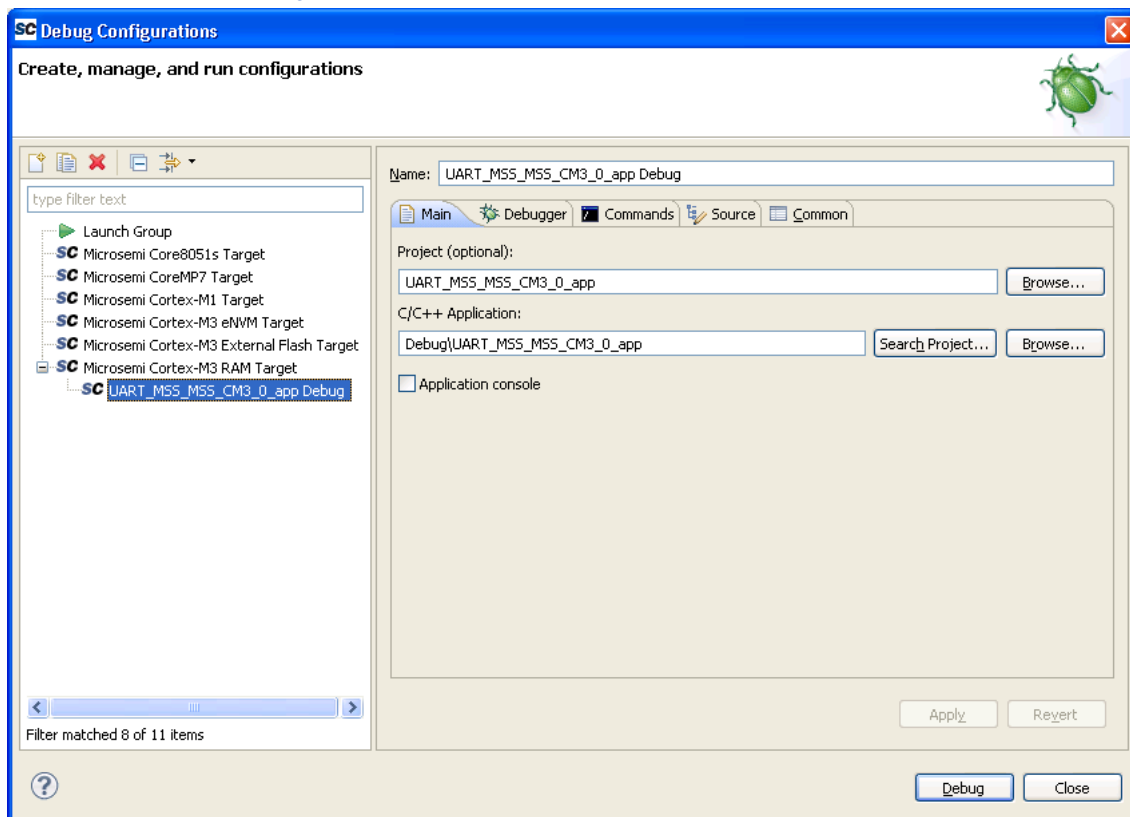


Figure 21 · Debug Configurations Window

3. If in case the **C/C++ Application** project is not displayed (as seen under **C/C++ Application**) click **Search Project** to select the program to run.

Note: If you select the project name listed in project explorer and then invoke the debug window, it displays all the fields automatically.

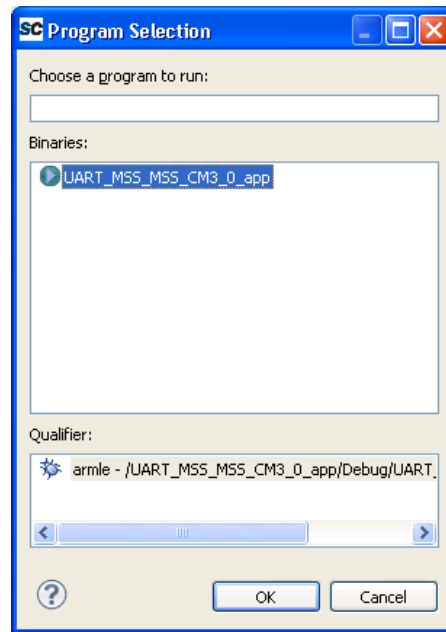


Figure 22 · Select Program

4. On the Debug window, click **Apply** and **Debug**.
5. Click **Yes** when prompted for **Confirm Perspective Switch**. This displays the Debug view mode.

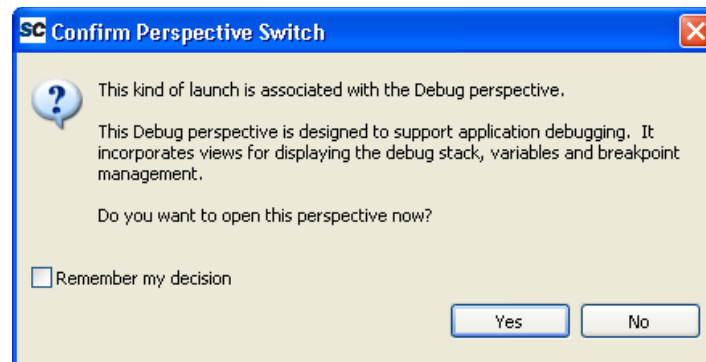


Figure 23 · Confirm Perspective Switch

The SoftConsole Debug Perspective window is displayed as shown in Figure 24 . .

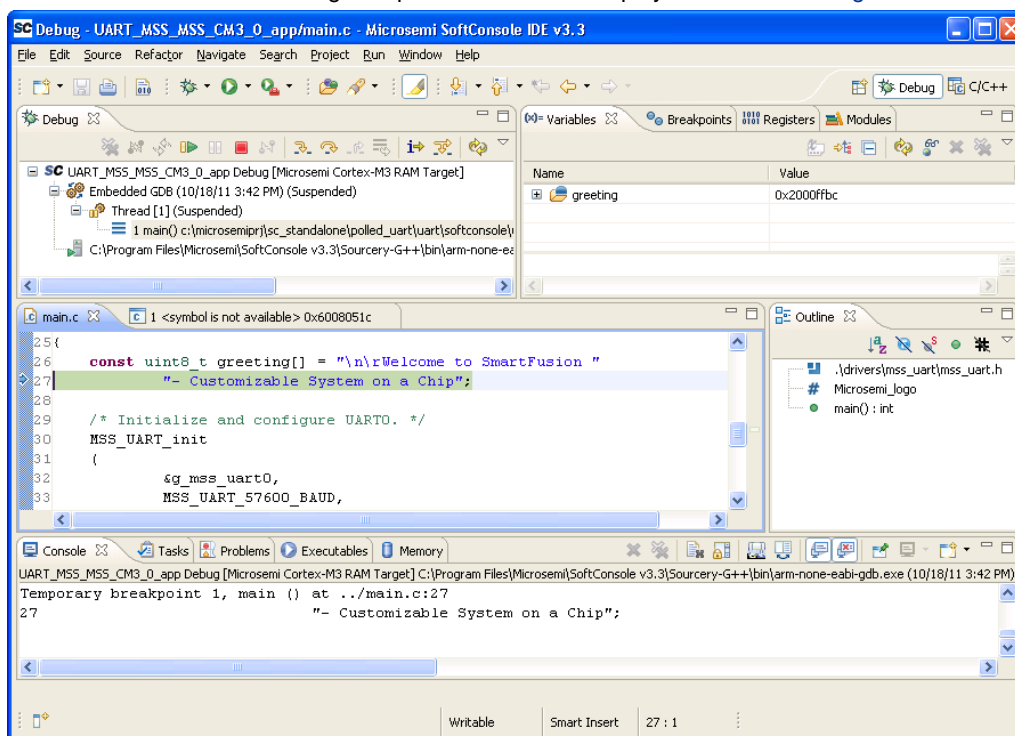


Figure 24 · The SoftConsole Debug Perspective

6. Click **Resume** or press the **F8** key to resume debugging.
7. Observe the HyperTerminal window. It should display the greeting message with the Microsemi name, as shown in Figure 25 . .

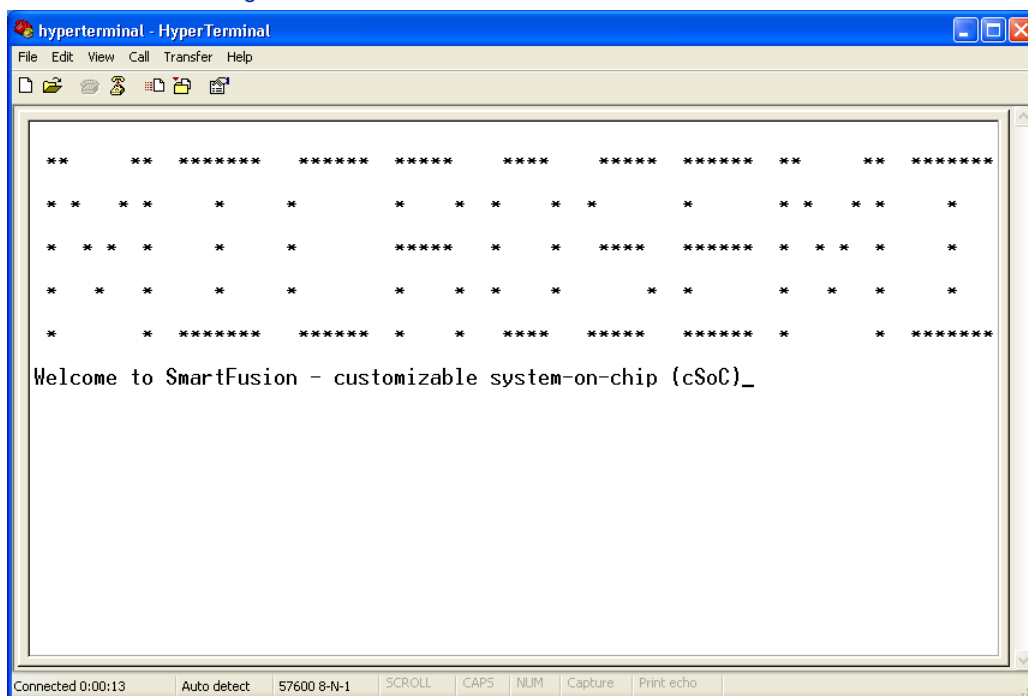
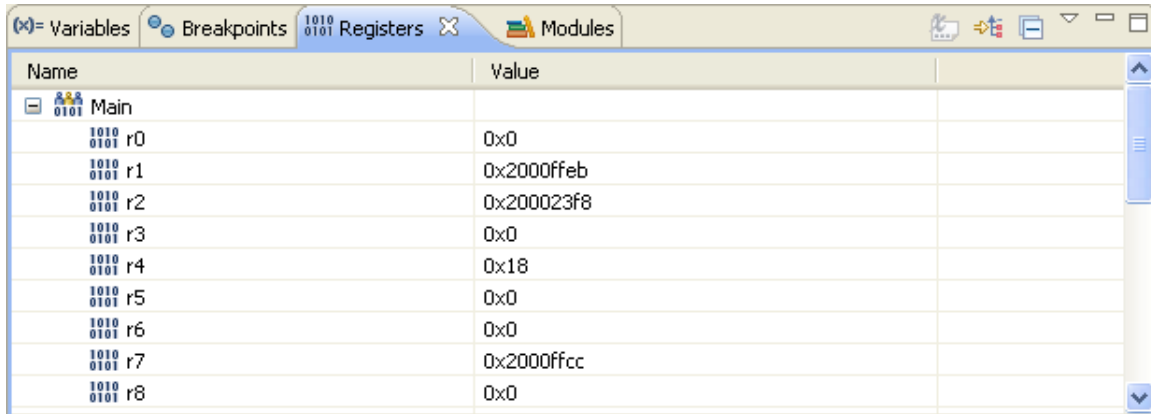


Figure 25 · HyperTerminal with Microsemi Name and Greeting Message

8. Suspend the software application by selecting **Run > Suspend** from the **SoftConsole** menu.

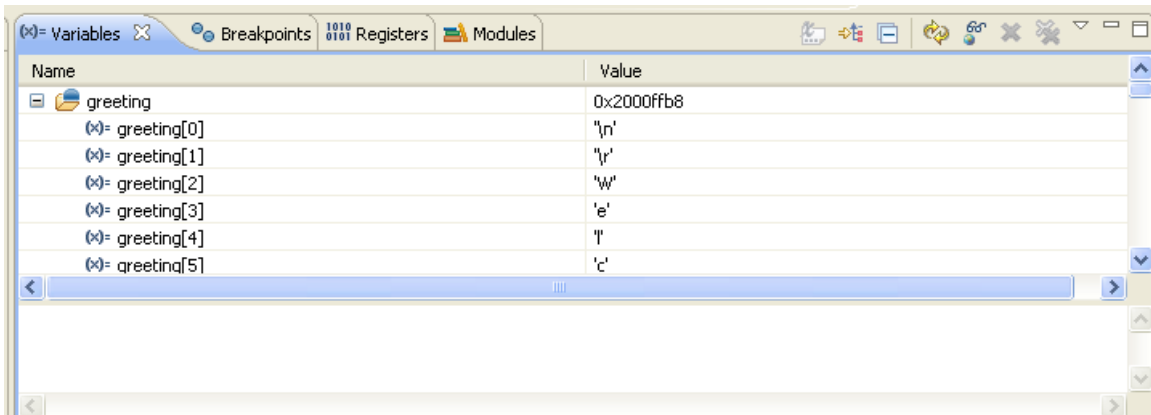
- Select the **Registers** tab on the upper right window pane to view the value of the Cortex-M3 internal registers:



Name	Value
Main	
r0	0x0
r1	0x2000ffeb
r2	0x200023f8
r3	0x0
r4	0x18
r5	0x0
r6	0x0
r7	0x2000ffcc
r8	0x0

Figure 26 · Values of the Cortex-M3 Internal Registers

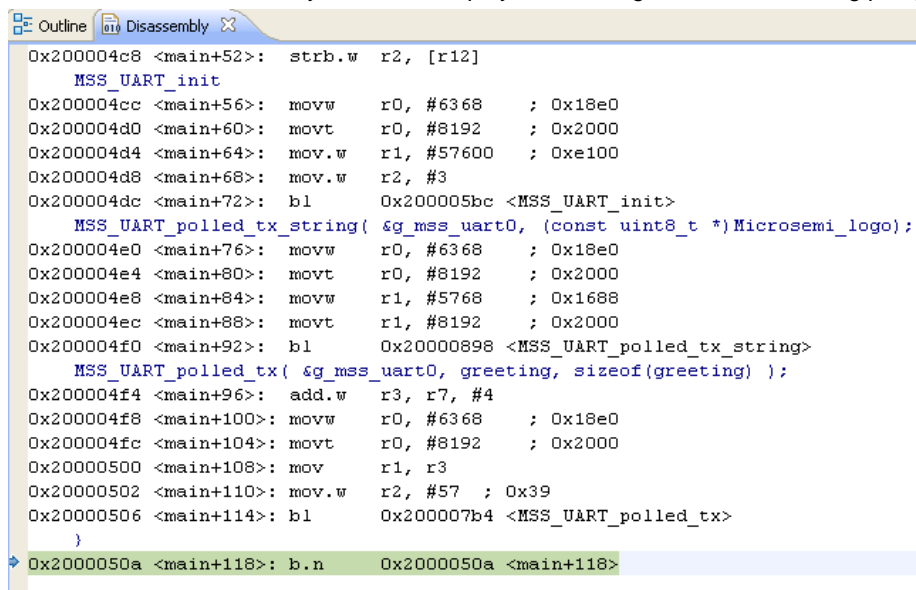
- Select the **Variables** tab in the upper left window pane to view the value of variables in the source code:



Name	Value
greeting	0x2000ffb8
greeting[0]	'n'
greeting[1]	'r'
greeting[2]	'w'
greeting[3]	'e'
greeting[4]	'l'
greeting[5]	'c'

Figure 27 · Values of Variables in the Source Code

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






```

0x200004c8 <main+52>: strb.w  r2, [r12]
    MSS_UART_init
0x200004cc <main+56>: movw    r0, #6368    ; 0x18e0
0x200004d0 <main+60>: movt    r0, #8192    ; 0x2000
0x200004d4 <main+64>: mov.w   r1, #57600   ; 0xe100
0x200004d8 <main+68>: mov.w   r2, #3
0x200004dc <main+72>: bl      0x200005bc <MSS_UART_init>
    MSS_UART_polled_tx_string( &g_mss_uart0, (const uint8_t *)Microsemi_logo);
0x200004e0 <main+76>: movw    r0, #6368    ; 0x18e0
0x200004e4 <main+80>: movt    r0, #8192    ; 0x2000
0x200004e8 <main+84>: movw    r1, #5768    ; 0x1688
0x200004ec <main+88>: movt    r1, #8192    ; 0x2000
0x200004f0 <main+92>: bl      0x20000898 <MSS_UART_polled_tx_string>
    MSS_UART_polled_tx( &g_mss_uart0, greeting, sizeof(greeting) );
0x200004f4 <main+96>: add.w   r3, r7, #4
0x200004f8 <main+100>: movw    r0, #6368    ; 0x18e0
0x200004fc <main+104>: movt    r0, #8192    ; 0x2000
0x20000500 <main+108>: mov     r1, r3
0x20000502 <main+110>: mov.w   r2, #57      ; 0x39
0x20000506 <main+114>: bl      0x200007b4 <MSS_UART_polled_tx>
    }
0x2000050a <main+118>: b.n     0x2000050a <main+118>

```

Figure 28 · Assembly Level Instructions

12. You can single-step through the source code by choosing **Run > Step Into** or **Run > Step Over** or by clicking the **Step Into**  or **Step Over**  icons. Observe the changes in the source code window and Disassembly view. Performing a **Step Over** allows for stepping over functions; the entire function is executed but there is no need to go through each instruction contained in the function.
13. Click the **Instruction Stepping**  icon and then perform **Step Into** operations. Observe that **Step Into** now executes a single line of assembly code.
14. Click the **Instruction Stepping** icon again to exit the instruction stepping mode. Single-step through the application and observe the instruction sequence in the source code window in the middle of the Debug perspective, and the values of the variables and registers.
15. Resume execution of the code by choosing **Run > Resume** or by clicking the **Resume** icon.
16. Add breakpoints in the application. Enter commands in the HyperTerminal window to force the code to halt, then single step and observe the instruction sequence.
17. Select **Embedded GDB** under the **Debug** tab in the upper left corner of SoftConsole. Right-click and choose **Terminate and Remove** to stop the debugger when you are finished.
18. Close the HyperTerminal and SoftConsole using **File > Exit**.

Step 8 - Debugging the Application Using the printf Statement in SoftConsole

Minor modifications need to be made in the above project in order to debug using the printf function. Refer to the steps below to use printf. You will create a new project to implement this function.

1. Follow the steps as described in [Step 2 - Configuring MSS Peripherals](#) and Step 3 - Programming SmartFusion Board Using FlashPro and switch to new SoftConsole workspace. Ensure that you close the previous SoftConsole project and also ensure that the name of the new project is different from the one used earlier.
2. Right click **<project_name>_MSS_MSS_hw_platform** and select **Properties**. Click **Settings** under **C/C++ Build**.
3. Select **Miscellaneous** under **GNU C Linker** in **Tool Settings** tab.
4. To add a symbol, click **Add** and then type **ACTEL_STUDIO_THRU_UART** in the **Add Symbol** dialog box. Click **OK**.

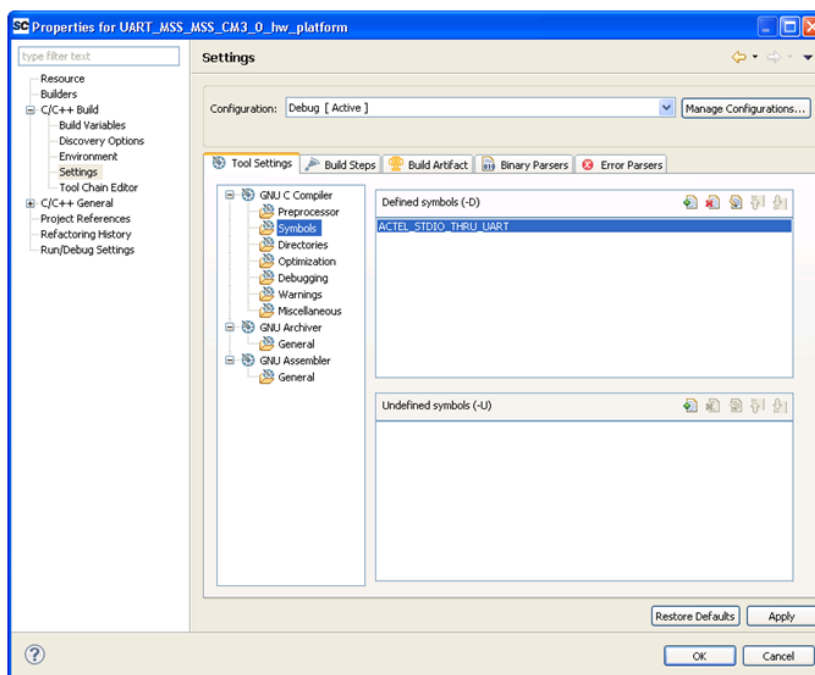


Figure 29 · Properties Window

5. Right click **<project_name>_MSS_MSS_CM3_0_app** and select properties. Click **Settings** under **C/C++ Build**.
6. Select **Miscellaneous** located under the **GNU C Linker** and enter the following in the **Linker flags** field: `-T ../UART_MSS_MSS_CM3_0_hw_platform/CMSIS/startup_gcc/debug-in-actel-smartfusion-envm.ld`

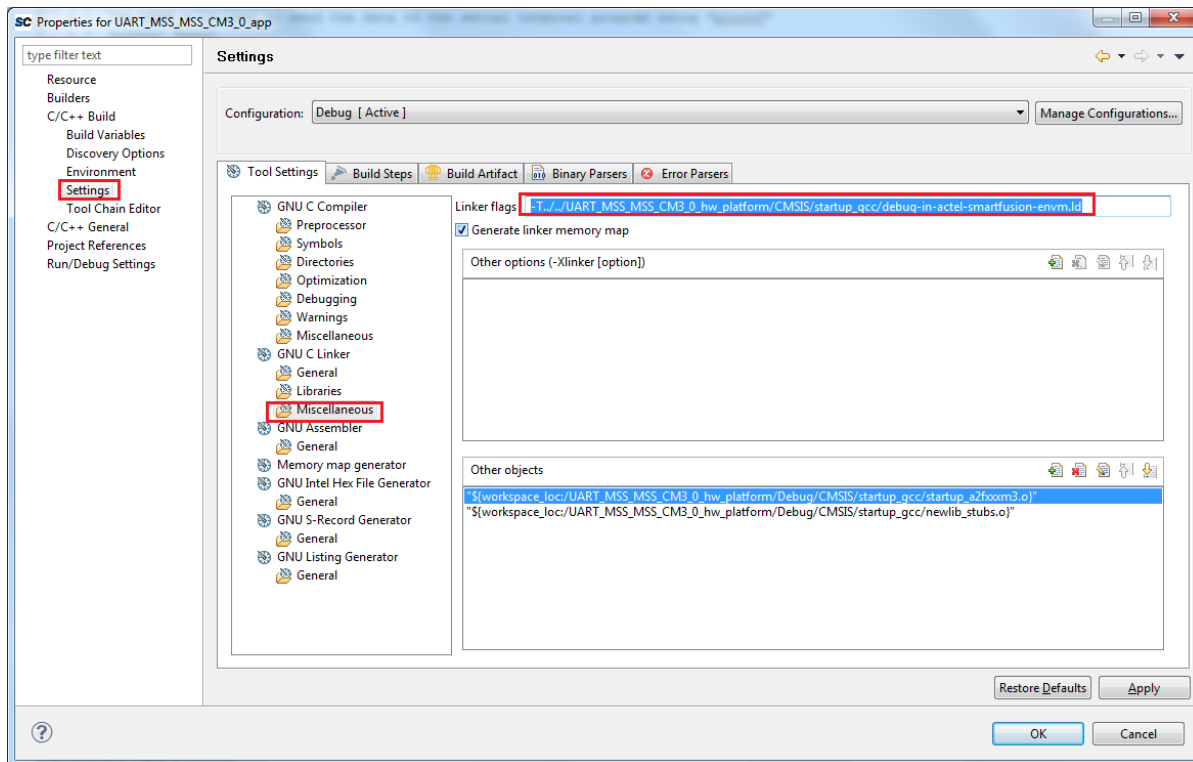


Figure 30 · Specifying Values for the Linker Flags Field

- Click **OK** and close the properties window by clicking **Apply**, then **OK**.
- Replace the code of **main.c** in the **UART_MSS_MSS_CM3_0_app** with the following code below:

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

/* Main function */
int main()
{
    int i = 0, j = 8;

    /* printing Microsemi logo on the serial terminal using "printf" */
    printf ("\n\r \
**      **   *          *          *          *          *          *          *          *          *\n\r \
* *     * *       *        *         *      *      *      *          *          *          *          *\n\r \
*  * *  *         *        *          *          *          *          *          *          *          *\n\r \
*   *   *         *        *          *      *      *      *          *          *          *          *\n\r \
*           *          *          *        *          *          *          *          *          *\n\r \
*/);

    /* printing greeting message on the serial terminal using "printf" */
    printf ("Welcome to SmartFusion-customizable system-on-chip(cSoC)\n\r");

    /* printing 8 Table on UART */
```

```
printf("Times 8 Table: \n\r");
for(i = 0; i <= 10; i++)
{
    printf("    %d x %d = %d\n\r", i, j, j*i);
}

while(1)
{
}

return 0;
```

9. To build your project, select the project name in the **Project Explorer** tab and click **Project > Clean Project**. Enter the settings as shown in the [Figure 31](#) and click **OK**.

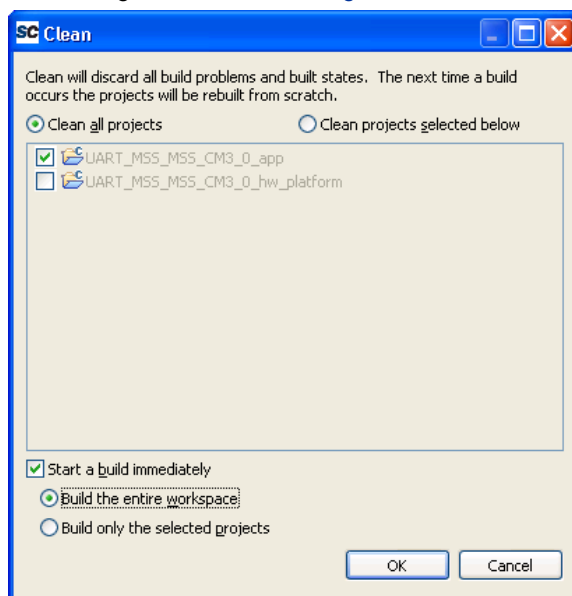


Figure 31 · Building a Clean Project

Ensure that there are no errors and warnings.

10. Select **Open Debug Dialog** or **Debug Configurations** from the **Run** menu. The Debug dialog is displayed.

11. Double-click **Microsemi Cortex-M3 eNVM** target, which displays a similar screen as [Figure 32](#) .

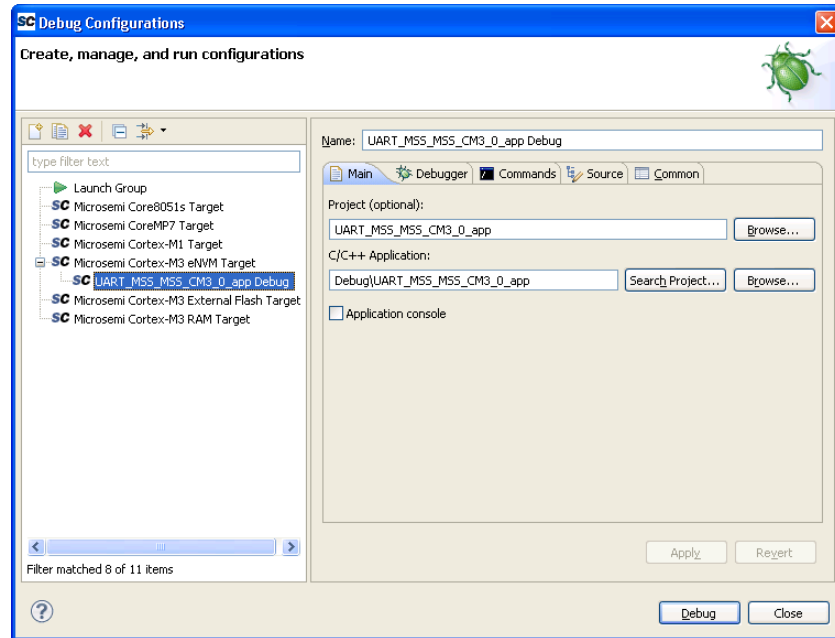


Figure 32 · Microsemi Cortex-M3 eNVM Target

12. Click **Debug**.
13. **Click Yes when prompted to Confirm Perspective Switch.** Ensure that the USB cable is connected to the programmer and the device and also ensure that the HyperTerminal is open and connected.
14. The SoftConsole Debug Perspective should now resemble [Figure 33](#) .

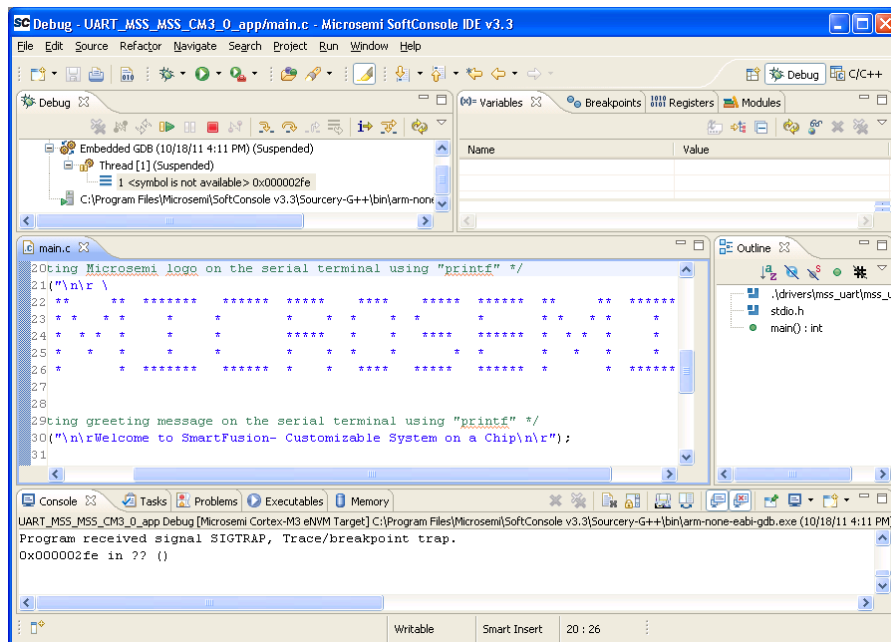


Figure 33 · The SoftConsole Debug Perspective

15. Click **Resume** or press the **F8** key to resume debugging.
Observe the HyperTerminal window; it should display greeting message along with the multiplier table of 8.

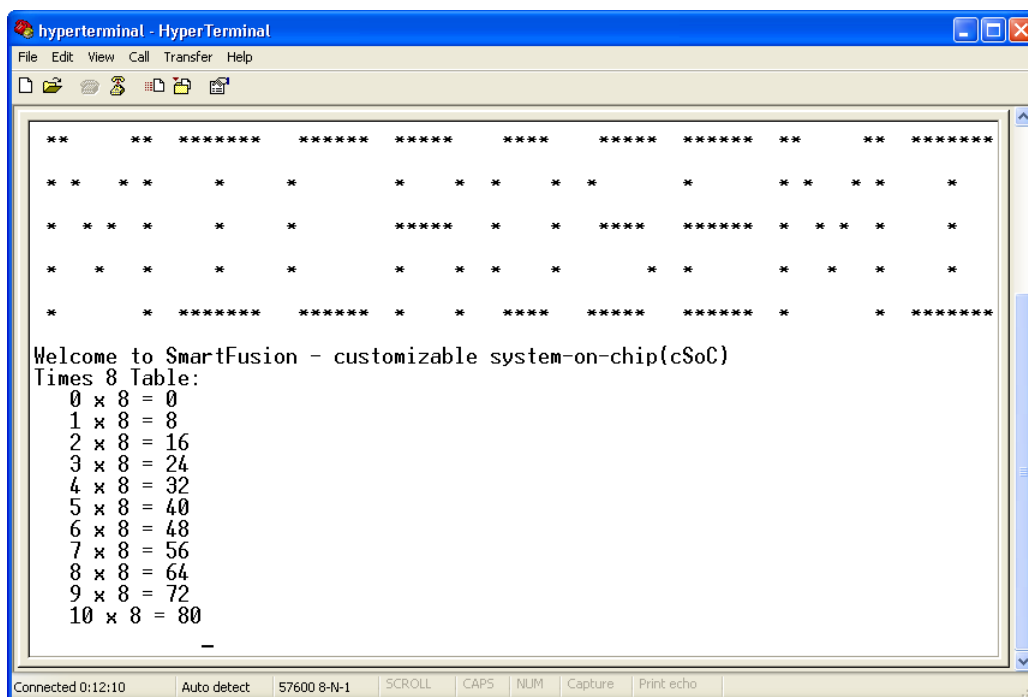


Figure 34 · The HyperTerminal Screen

Refer to [Step 7 - Debugging the Application Using SoftConsole](#) for debugging the application.

Step 9 - Building Executable Image in Release mode

You can build an application executable image in “release mode” and load it into eNVM for executing code in the eNVM of SmartFusion cSoC device. You can load the application executable image into eNVM with the help of eNVM data storage client from SmartDesign MSS Configurator and in-application programming (IAP) or FlashPro programming software. In release mode, you cannot use SoftConsole debugger to load the executable image into eNVM. For steps to build an executable image for our application refer the tutorial [SmartFusion: Building Executable Image in Release Mode and Loading into eNVM](#).

This concludes the tutorial.

Appendix A – Libero SoC Vault/Repository Settings

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

1. Click **Project > Vault/Repositories Settings**.
2. The **Vault/Repositories Settings** window is displayed. Click **Repositories** and add the following in the address field:
 - www.actel-ip.com/repositories/SgCore
 - www.actel-ip.com/repositories/DirectCore
 - www.actel-ip.com/repositories/Firmware

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

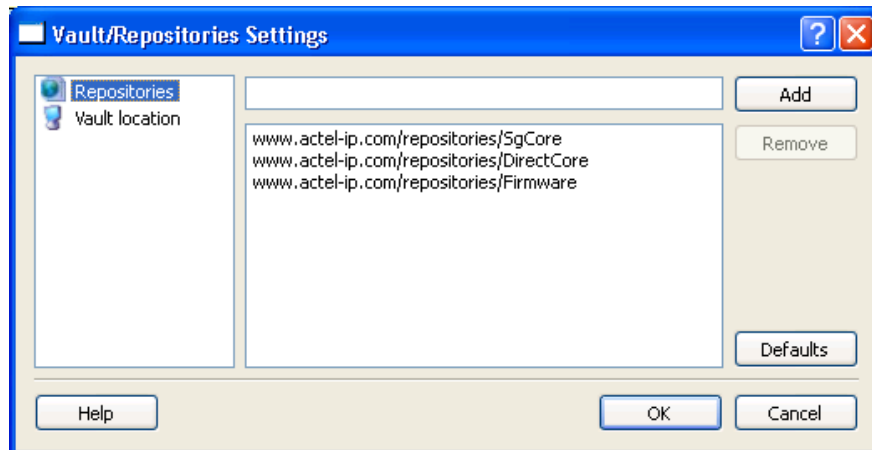


Figure 35 · Setting Repositories

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

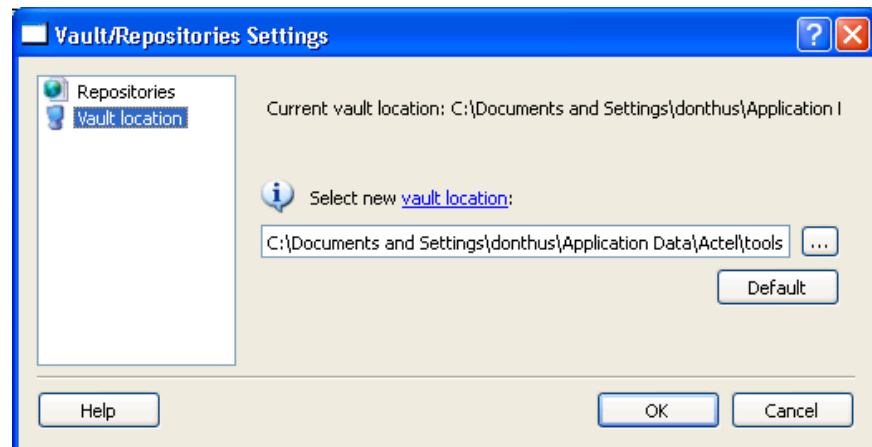


Figure 36 · Setting the Vault Location

Appendix B – Firmware Catalog Settings

1. Open the < Libero Installation directory>\Designer\bin\catalog.exe.
2. Select **Tools > Vault/Repositories Settings**, from the Firmware Catalog widow.

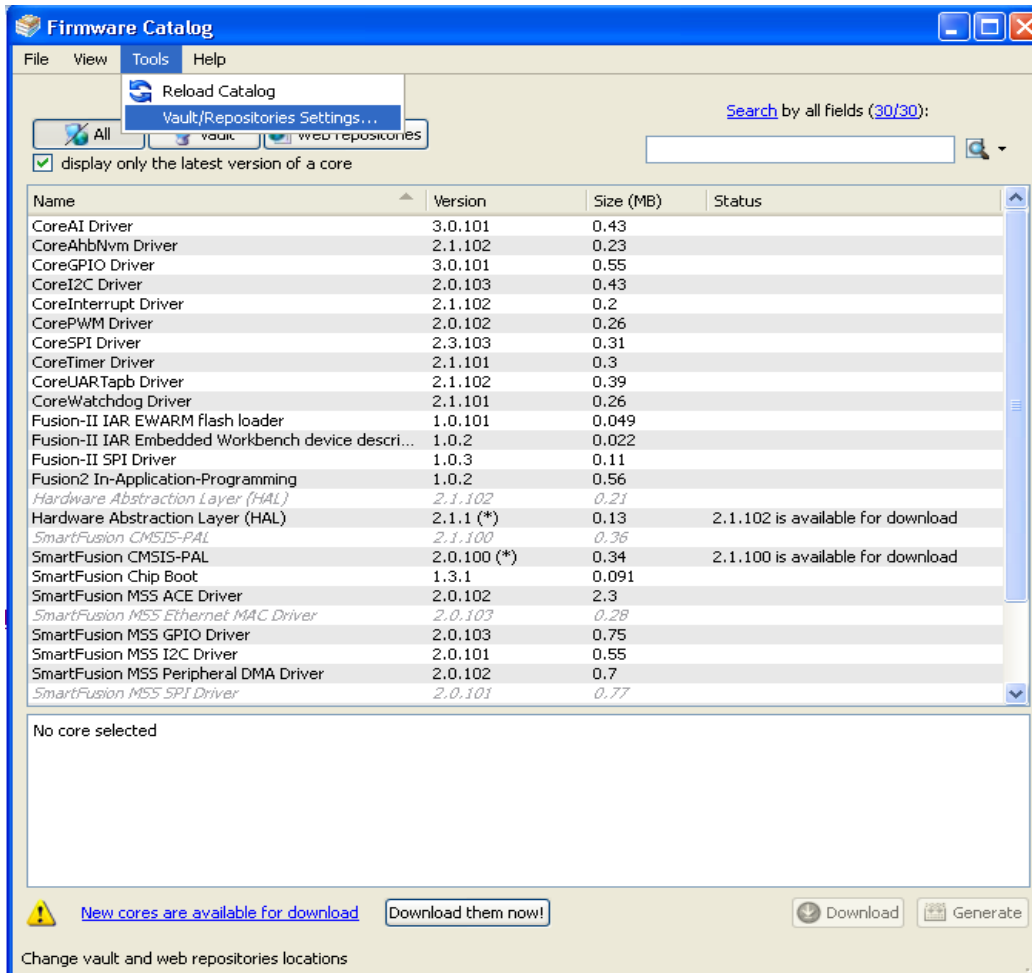


Figure 37 · Firmware Catalog Settings

3. Select **Repositories** in the **Vault/Repositories Settings** dialog box.
4. Confirm that the following repositories are displayed (add them if needed):
 - www.actel-ip.com/repositories/SgCore
 - www.actel-ip.com/repositories/DirectCore
 - www.actel-ip.com/repositories/Firmware
5. Add the above mentioned paths in the address field if required by selecting the repository and clicking **Add**.
If new cores are available for download, click **Download them now!** to download the new cores to the vault.

List of Changes

Revision	Changes	Page
Revision 5 (April 2012)	Modified note under Associated Project Files (SAR 38280)	4
	Modified Step 2 - Configuring MSS Peripherals (SAR 38280)	6
	Replaced Figure 4 · (SAR 38280)	6
	Modified Step 3 - Programming SmartFusion Board Using FlashPro (SAR 38280)	11
	Modified note under Step 4 - Building the Project (SAR 38280)	12
	Replaced Figure 30 · (SAR 38280)	23
Revision 4 (February 2012)	Modified Associated Project Files section (SAR 36889).	4
	Added note below Figure 13 · (SAR 36889).	12
	Replaced Figure 17 and modified Step 5 - Configuring Serial Terminal Emulation Program section (SAR 36889).	15
	Modified Step 6 - Installing Drivers for the USB to RS232 Bridge section (SAR 36889).	18
Revision 3 (November 2011)	Updated the document for Libero SoC v10.0 (SAR 35049).	NA

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

Product Support

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

Customer Service

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

From North America, call **800.262.1060**

From the rest of the world, call **650.318.4460**

Fax, from anywhere in the world **408.643.6913**

Customer Technical Support Center

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

Technical Support

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

Website

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

Contacting the Customer Technical Support Center

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

Email

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

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

My Cases

Microsemi SoC Products Group customers may submit and track technical cases online by going to [My Cases](#).

Outside the U.S.

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

ITAR Technical Support

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



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

Microsemi Corporation (NASDAQ: MSCC) offers a comprehensive portfolio of semiconductor solutions for: aerospace, defense and security; enterprise and communications; and industrial and alternative energy markets. Products include high-performance, high-reliability analog and RF devices, mixed signal and RF integrated circuits, customizable SoCs, FPGAs, and complete subsystems. Microsemi is headquartered in Aliso Viejo, Calif. Learn more at www.microsemi.com.

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