SmartFusion cSoC: Building Executable Image in Release Mode and Loading into eNVM

Tutorial

Table of Contents

SmartFusion cSoC: Building Executable Image in Release Mode and Loading into eNVM 3
Introduction 3
Building an Executable Image in Release Mode 4
Loading the Executable Image into eNVM 6

List of Changes 14

Product Support 15
Customer Service 15
Customer Technical Support Center 15
Technical Support 15
Website 15
Contacting the Customer Technical Support Center 15
ITAR Technical Support 16

SmartFusion cSoC: Building Executable Image in Release Mode and Loading into eNVM

Introduction

SmartFusion® customized system-on-chip (cSoC) devices contain a hard embedded microcontroller subsystem (MSS), programmable analog circuitry, and FPGA fabric consisting of logic tiles, SRAM, and PLLs. The MSS consists of 100 MHz ARM® Cortex™-M3 processor, advanced high-performance bus (AHB) matrix, system registers, Ethernet MAC, DMA engine, real-time counter (RTC), embedded nonvolatile memory (eNVM), embedded SRAM (eSRAM), fabric interface controller (FIC), I²C, SPI, and EMC. Figure 1 · Figure 1 · shows the SmartFusion cSoC block diagram.

This tutorial explains how to build an application executable image in 'release mode' and load it into eNVM for executing code in place in eNVM of the 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 IAP or FlashPro programming software. In the release mode, you cannot use SoftConsole debugger to load the executable image into eNVM.

This tutorial is in continuation of the SmartFusion tutorials and explains you how to build the applications in release mode after completion of application debugging. Refer to the *Using UART with SmartFusion - Microsemi Libero SoC and SoftConsole Flow Tutorial* to understand the flow for debugging mode.

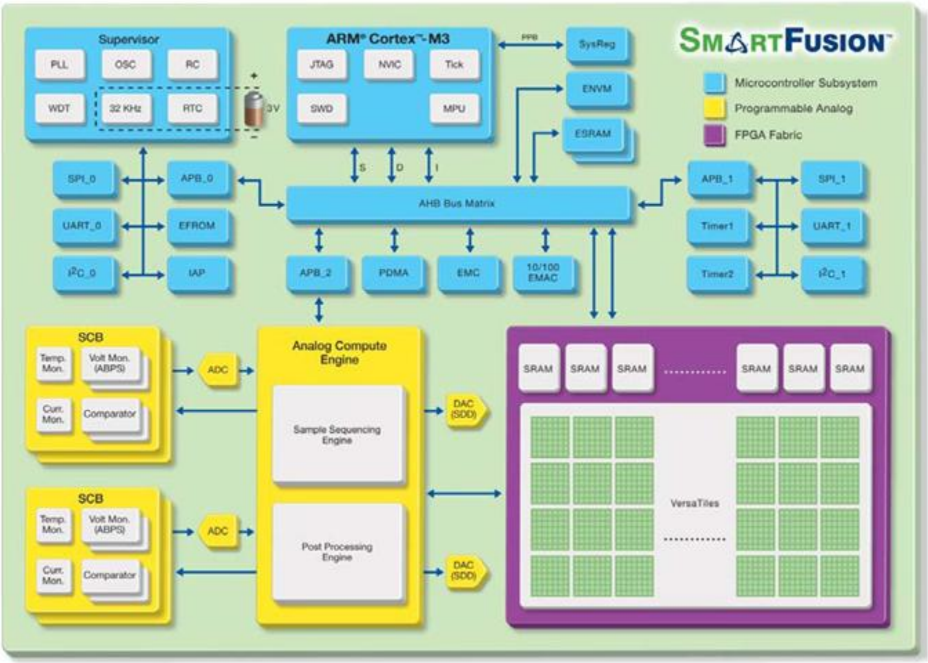
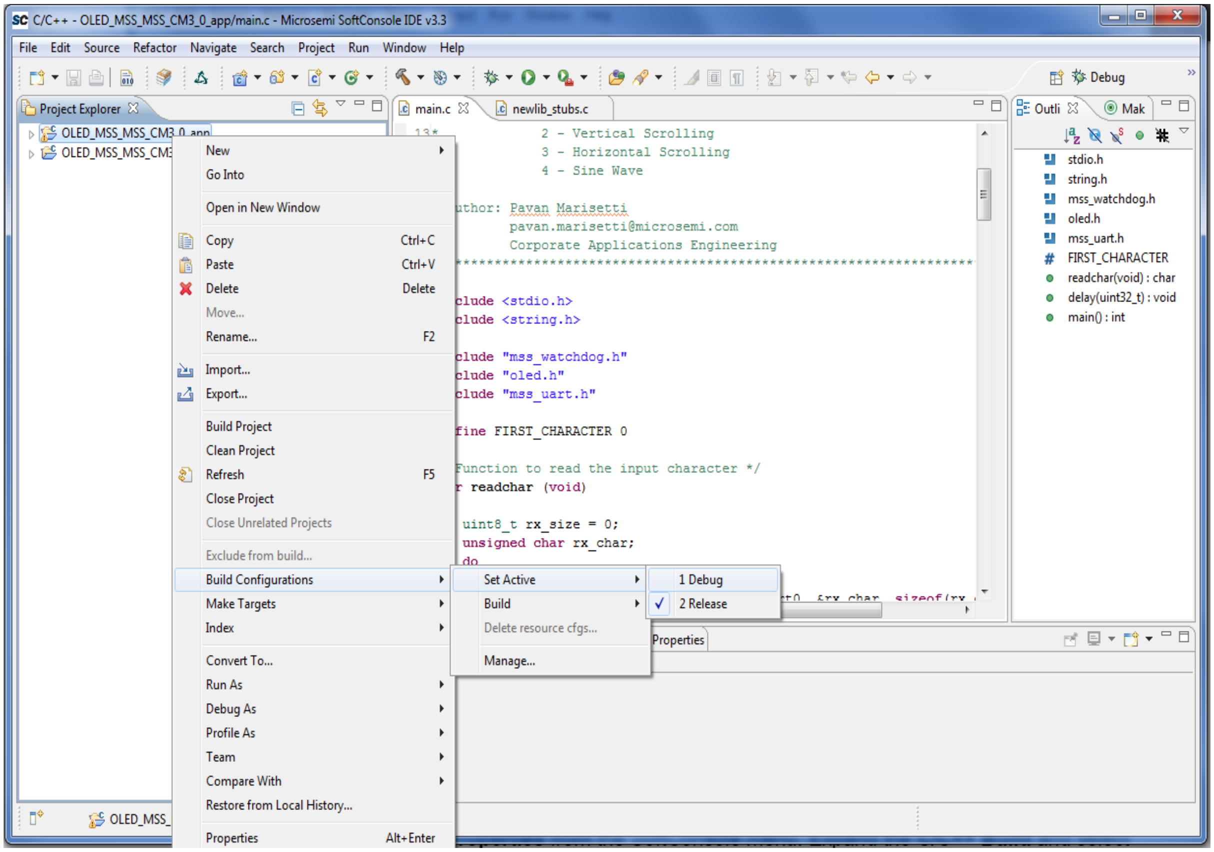

Supervisor
PLL
OSC
RC
WDT
32 KHz
RTC
3V
ARM® Cortex™-M3
JTAG
NVIC
Tick
SWD
MPU
PPB
SysReg
ENVM
ESRAM
SMARTFUSION™
Microcontroller Subsystem
Programmable Analog
FPGA Fabric
S
D
I
SPI_0
APB_0
UART_0
EFROM
I²C_0
IAP
AHB Bus Matrix
APB_1
SPI_1
Timer1
UART_1
Timer2
I²C_1
APB_2
PDMA
EMC
10/100 EMAC
SCB
Temp. Mon.
Volt Mon. (ABPS)
Curr. Mon.
Comparator
ADC
Analog Compute Engine
Sample Sequencing Engine
Post Processing Engine
DAC (SDD)
SRAM
VersaTiles

Figure 1 · SmartFusion cSoC Block Diagram

Building an Executable Image in Release Mode

This section outlines the steps required to build an executable image in release mode for executing the code in place in eNVM of the SmartFusion cSoC device.

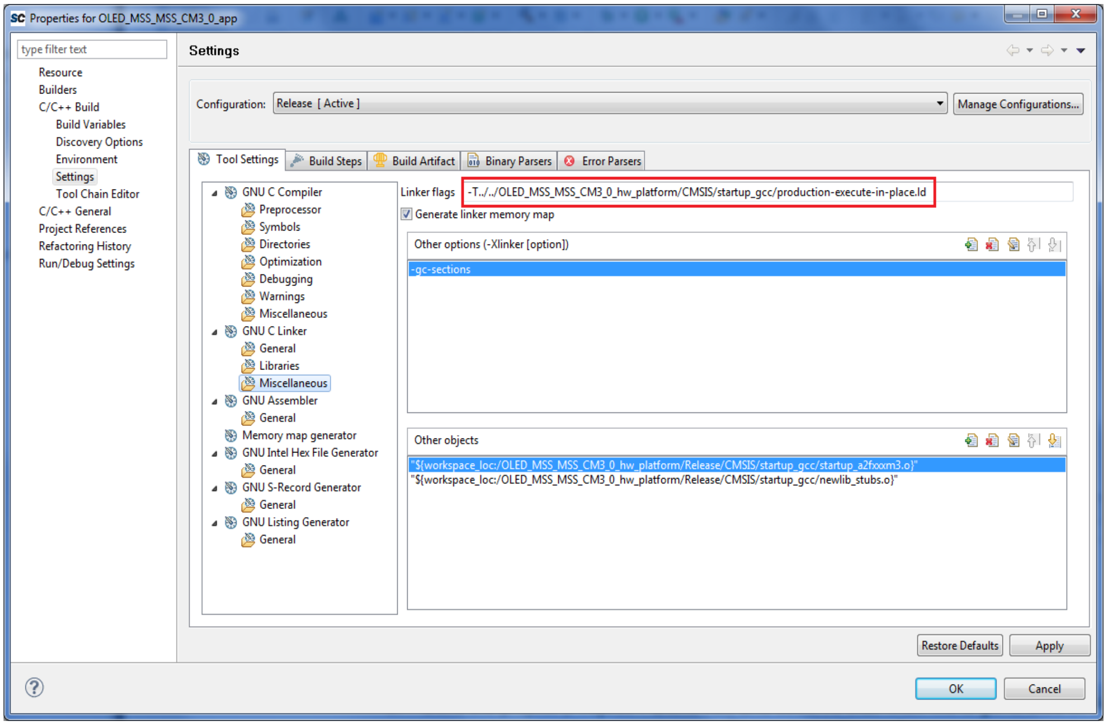
1. Open the SoftConsole project workspace provided in the design files of this tutorial.
2. Right click on both project names in the Project Explorer view and select **Build Configurations > Set Active > Release** as shown in Figure 2 ·


C/C++ - OLED_MSS_MSS_CM3_0_app/main.c - Microsemi SoftConsole IDE v3.3
File Edit Source Refactor Navigate Search Project Run Window Help
Debug
Project Explorer
OLED_MSS_MSS_CM3_0_app
OLED_MSS_MSS_CM3
main.c
newlib_stubs.c
New
Go Into
Open in New Window
Copy Ctrl+C
Paste Ctrl+V
Delete Delete
Move...
Rename... F2
Import...
Export...
Build Project
Clean Project
Refresh F5
Close Project
Close Unrelated Projects
Exclude from build...
Build Configurations
Make Targets
Index
Convert To...
Run As
Debug As
Profile As
Team
Compare With
Restore from Local History...
Properties Alt+Enter
Set Active
Build
Delete resource cfgs...
Manage...
1 Debug
2 Release
Properties
Outli
Mak
stdio.h
string.h
mss_watchdog.h
oled.h
mss_uart.h
FIRST_CHARACTER
readchar(void) : char
delay(uint32_t) : void
main() : int
OLED_MSS_

```
              2 - Vertical Scrolling
              3 - Horizontal Scrolling
              4 - Sine Wave

uthor: Pavan Marisetti
       pavan.marisetti@microsemi.com
       Corporate Applications Engineering
*******************************************************************

clude <stdio.h>
clude <string.h>

clude "mss_watchdog.h"
clude "oled.h"
clude "mss_uart.h"

fine FIRST_CHARACTER 0

Function to read the input character */
r readchar (void)

 uint8_t rx_size = 0;
 unsigned char rx_char;
 do
```

Figure 2 · Configuring Release Mode

3. Select **Project > Properties** from the SoftConsole menu. Expand the **C/C++ Build** and select ‘**Settings**’.
4. Provide the release mode linker script file to the linker by changing the ‘**Linker flags**’ field to “-*T../../OLED_MSS_MSS_CM3_0_hw_platform/CMSIS/startup_gcc/production-execute-in-place.ld*” as shown in the Figure 3 · .


Properties for OLED_MSS_MSS_CM3_0_app
type filter text
Resource
Builders
C/C++ Build
Build Variables
Discovery Options
Environment
Settings
Tool Chain Editor
C/C++ General
Project References
Refactoring History
Run/Debug Settings
Settings
Manage Configurations...
Tool Settings
Build Steps
Build Artifact
Binary Parsers
Error Parsers
GNU C Compiler
Preprocessor
Symbols
Directories
Optimization
Debugging
Warnings
Miscellaneous
GNU C Linker
General
Libraries
Miscellaneous
GNU Assembler
General
Memory map generator
GNU Intel Hex File Generator
General
GNU S-Record Generator
General
GNU Listing Generator
General
Restore Defaults
Apply
OK
Cancel

Configuration: Release [Active]

Linker flags `-T../../OLED_MSS_MSS_CM3_0_hw_platform/CMSIS/startup_gcc/production-execute-in-place.ld`

☑ Generate linker memory map

Other options (-Xlinker [option])

```
-gc-sections
```

Other objects

```
"${workspace_loc:/OLED_MSS_MSS_CM3_0_hw_platform/Release/CMSIS/startup_gcc/startup_a2fxxxm3.o}"
"${workspace_loc:/OLED_MSS_MSS_CM3_0_hw_platform/Release/CMSIS/startup_gcc/newlib_stubs.o}"
```

Figure 3 · Linker Script Settings

5. Click **Apply** and then click **OK**.
6. Build the project (**Project > Build All**) and observe that '.hex' file is generated in 'Release' folder created in project folder as shown in Figure 4 · and Figure 5 ·

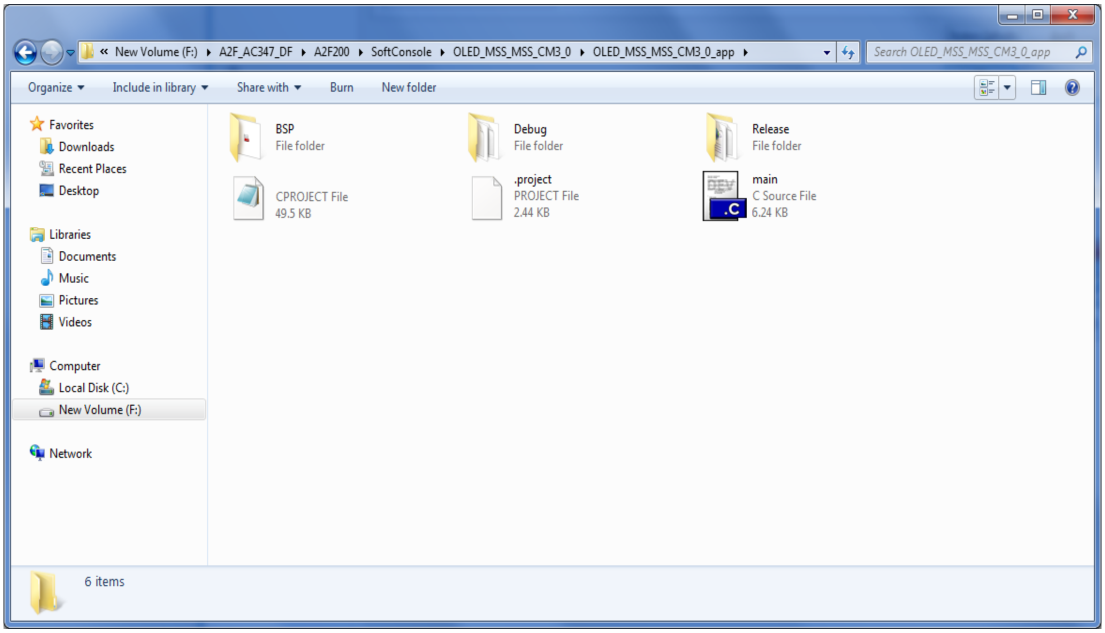

« New Volume (F:) ▸ A2F_AC347_DF ▸ A2F200 ▸ SoftConsole ▸ OLED_MSS_MSS_CM3_0 ▸ OLED_MSS_MSS_CM3_0_app ▸
Search OLED_MSS_MSS_CM3_0_app
Organize
Include in library
Share with
Burn
New folder
Favorites
Downloads
Recent Places
Desktop
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
New Volume (F:)
Network
BSP
File folder
Debug
File folder
Release
File folder
CPROJECT File
49.5 KB
.project
PROJECT File
2.44 KB
main
C Source File
6.24 KB
6 items

Figure 4 · Release Folder Location in Project Folder

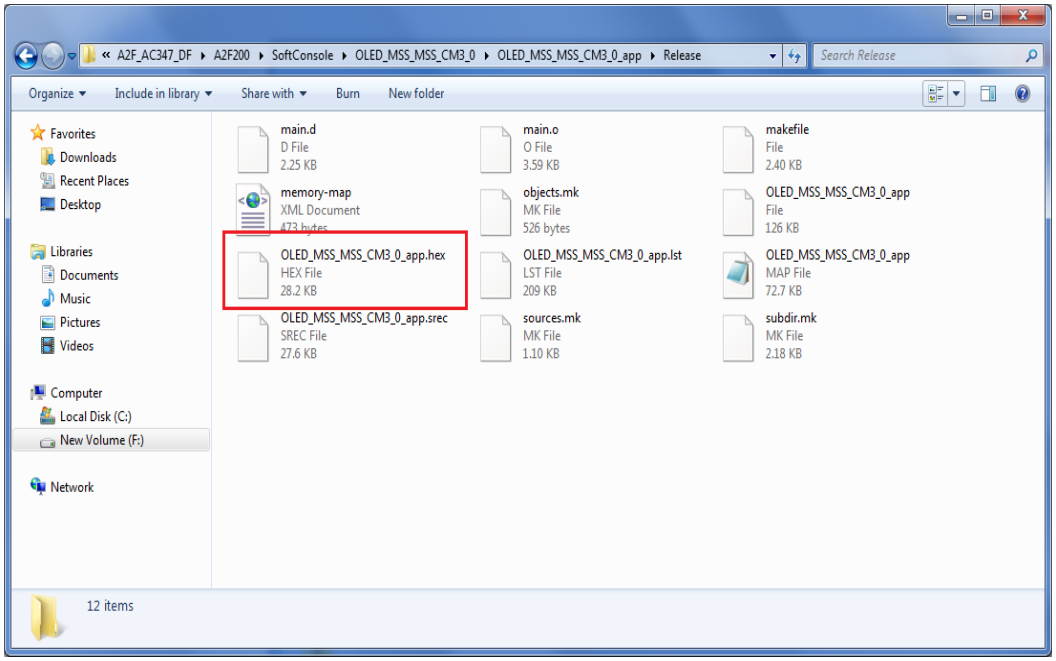

« A2F_AC347_DF ▸ A2F200 ▸ SoftConsole ▸ OLED_MSS_MSS_CM3_0 ▸ OLED_MSS_MSS_CM3_0_app ▸ Release
Search Release
Organize ▾
Include in library ▾
Share with ▾
Burn
New folder
Favorites
Downloads
Recent Places
Desktop
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
New Volume (F:)
Network
main.d D File 2.25 KB
main.o O File 3.59 KB
makefile File 2.40 KB
memory-map XML Document 473 bytes
objects.mk MK File 526 bytes
OLED_MSS_MSS_CM3_0_app File 126 KB
OLED_MSS_MSS_CM3_0_app.hex HEX File 28.2 KB
OLED_MSS_MSS_CM3_0_app.lst LST File 209 KB
OLED_MSS_MSS_CM3_0_app MAP File 72.7 KB
OLED_MSS_MSS_CM3_0_app.srec SREC File 27.6 KB
sources.mk MK File 1.10 KB
subdir.mk MK File 2.18 KB
12 items

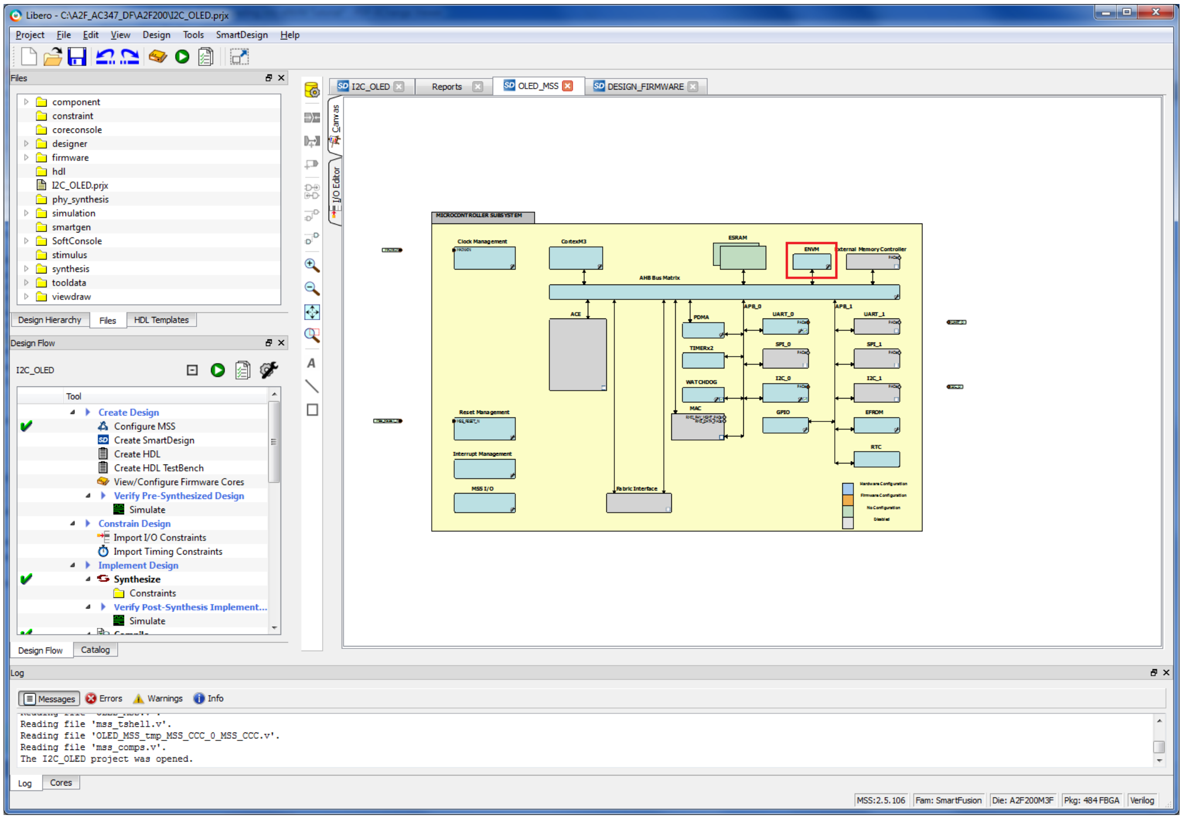
Figure 5 · Hex File Location in Release Folder

Loading the Executable Image into eNVM

This section outlines the steps required to load the generated executable image into eNVM of the SmartFusion cSoC device using the SmartDesign MSS Configurator.

1. Open the Libero SoC hardware project provided in the design files of this tutorial.
2. Open the MSS component created using the SmartDesign MSS Configurator by double clicking on the MSS component present in the 'Design Hierarchy' window. The **SmartDesign MSS Configurator** tab is displayed as shown in Figure 6 · .

If you are using the SoftConsole standalone flow then the created MSS component can be launched from the SoftConsole project workspace (**Run > External Tools > MSS_Configurator**).

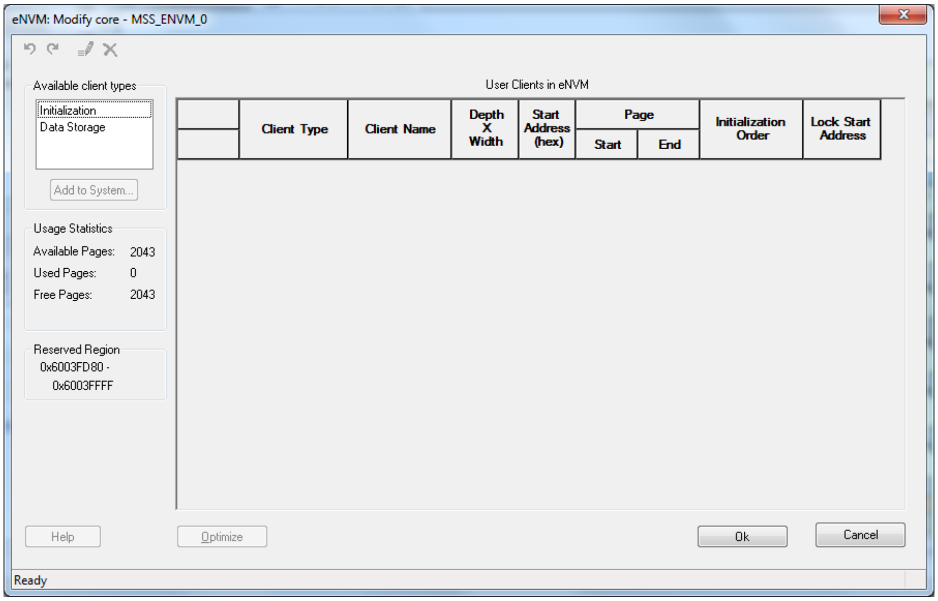

Libero - C:\A2F_AC347_DF\A2F200\I2C_OLED.prjx
Project File Edit View Design Tools SmartDesign Help
Files
component
constraint
coreconsole
designer
firmware
hdl
I2C_OLED.prjx
phy_synthesis
simulation
smartgen
SoftConsole
stimulus
synthesis
tooldata
viewdraw
Design Hierarchy
Files
HDL Templates
Design Flow
I2C_OLED
Tool
Create Design
Configure MSS
Create SmartDesign
Create HDL
Create HDL TestBench
View/Configure Firmware Cores
Verify Pre-Synthesized Design
Simulate
Constrain Design
Import I/O Constraints
Import Timing Constraints
Implement Design
Synthesize
Constraints
Verify Post-Synthesis Implement...
Simulate
Design Flow
Catalog
I2C_OLED
Reports
OLED_MSS
DESIGN_FIRMWARE
Canvas
I/O Editor
MICROCONTROLLER SUBSYSTEM
Clock Management
CortexM3
ESRAM
ENVM
External Memory Controller
AHB Bus Matrix
APB_0
APB_1
ACE
PDMA
UART_0
UART_1
TIMERx2
SPI_0
SPI_1
WATCHDOG
I2C_0
I2C_1
MAC
GPIO
EFROM
Reset Management
RTC
Interrupt Management
MSS I/O
Fabric Interface
Log
Messages
Errors
Warnings
Info
Log
Cores

```
Reading file 'mss_tshell.v'.
Reading file 'OLED_MSS_tmp_MSS_CCC_0_MSS_CCC.v'.
Reading file 'mss_comps.v'.
The I2C_OLED project was opened.
```

MSS:2.5.106 | Fam: SmartFusion | Die: A2F200M3F | Pkg: 484 FBGA | Verilog

Figure 6 · SmartDesign MSS Configurator

3. Double click on the ENVM module available in the SmartDesign MSS Configurator to add eNVM data storage client. The MSS_ENVM_0 configurator window is displayed as shown in the Figure 7 · .


eNVM: Modify core - MSS_ENVM_0
Available client types
Initialization
Data Storage
Add to System...
User Clients in eNVM
Help
Optimize
Ok
Cancel

	Client Type	Client Name	Depth X Width	Start Address (hex)	Page		Initialization Order	Lock Start Address
					Start	End		

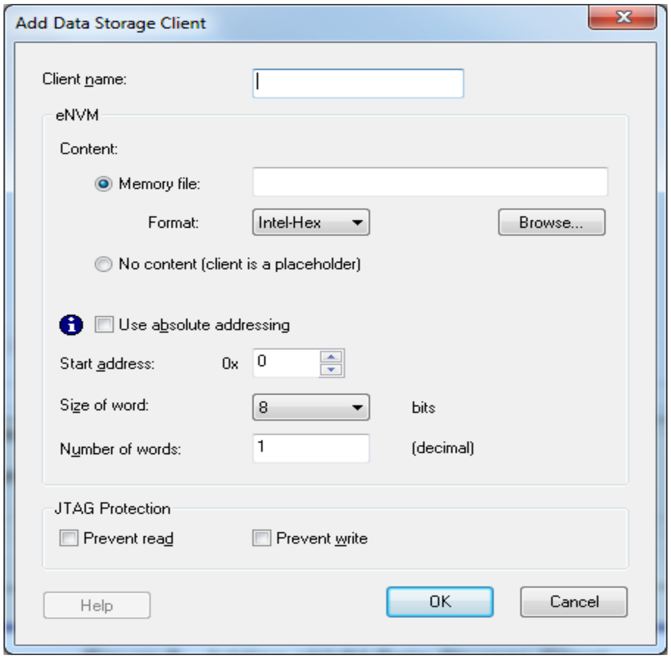
Usage Statistics
Available Pages: 2043
Used Pages: 0
Free Pages: 2043

Reserved Region
0x6003FD80 -
0x6003FFFF

Ready

Figure 7 · MSS_ENVM_0 Configurator

4. Select **Data Storage** under the **Available client types** tab and click **Add to System**. The **Add to Data Storage Client** window opens as shown in Figure 8 · .


Add Data Storage Client
Client name:
eNVM
Content:
Browse...
○ No content (client is a placeholder)
☐ Use absolute addressing
JTAG Protection
☐ Prevent read
☐ Prevent write
Help
OK
Cancel

● Memory file:
Format: Intel-Hex
Start address: 0x 0
Size of word: 8 bits
Number of words: 1 (decimal)

Figure 8 · Adding eNVM Data Storage Client

5. Enter a data storage client name in the **Add to Data Storage Client** window.
6. Browse for .hex file generated in the Building an Executable Image in Release Mode section. The generated executable image can be found in the Release folder under SoftConsole project workspace as shown in the Figure 9 · .

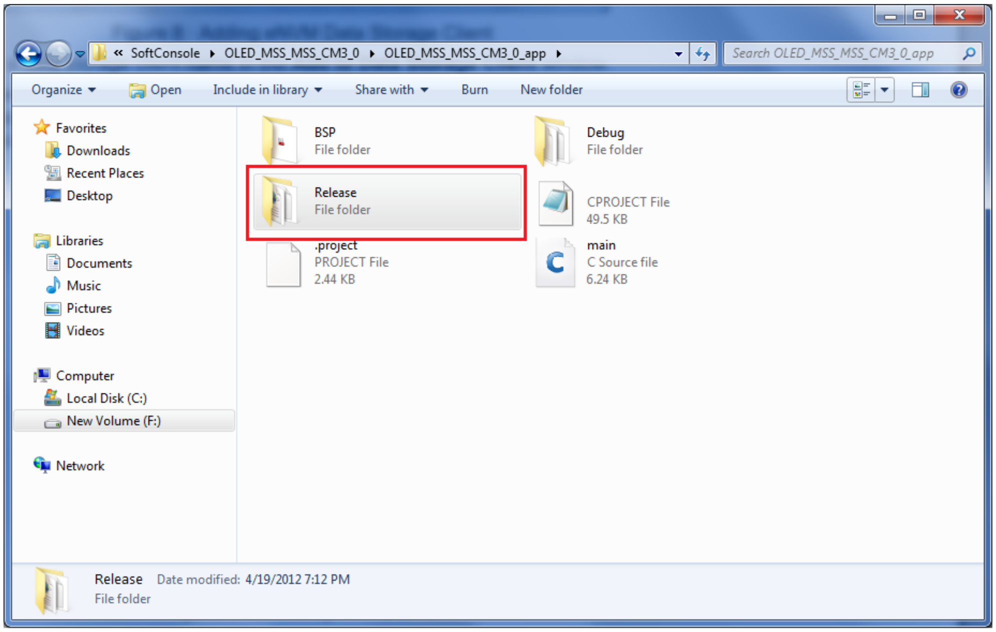
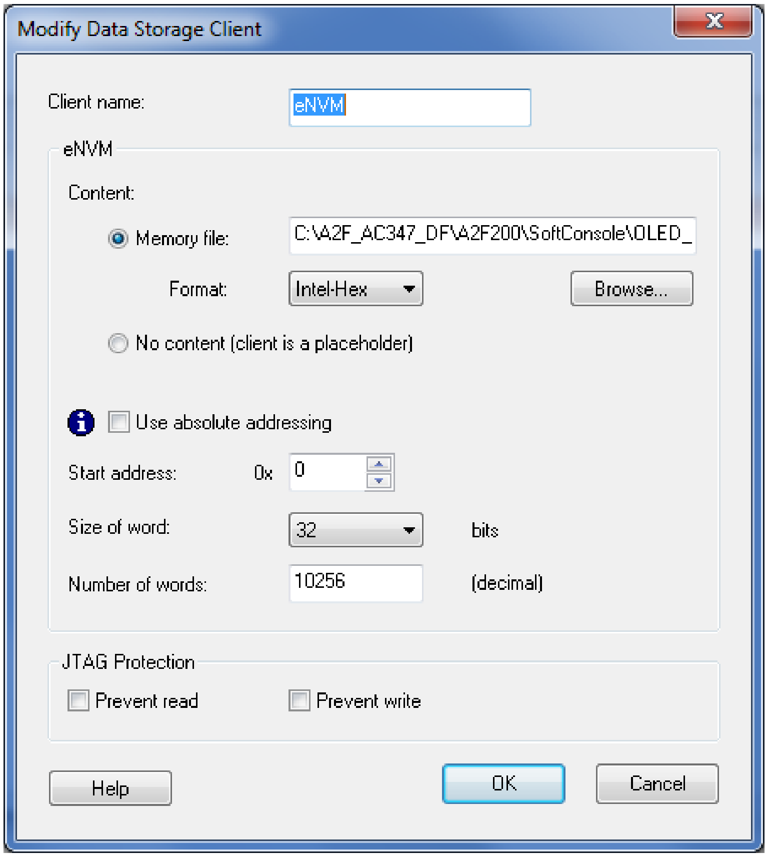

« SoftConsole ▸ OLED_MSS_MSS_CM3_0 ▸ OLED_MSS_MSS_CM3_0_app ▸
Search OLED_MSS_MSS_CM3_0_app
Organize
Open
Include in library
Share with
Burn
New folder
Favorites
Downloads
Recent Places
Desktop
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
New Volume (F:)
Network
BSP
File folder
Debug
File folder
Release
File folder
CPROJECT File
49.5 KB
.project
PROJECT File
2.44 KB
main
C Source file
6.24 KB
Release Date modified: 4/19/2012 7:12 PM
File folder

Figure 9 · Browsing for .hex File

7. Change the size of word to 32 bits as shown in the Figure 10 · .



Client name: eNVM

● Memory file: C:\A2F_AC347_DF\A2F200\SoftConsole\OLED_

Format: Intel-Hex

○ No content (client is a placeholder)

☐ Use absolute addressing

Start address: 0x 0

Size of word: 32 bits

Number of words: 10256 (decimal)

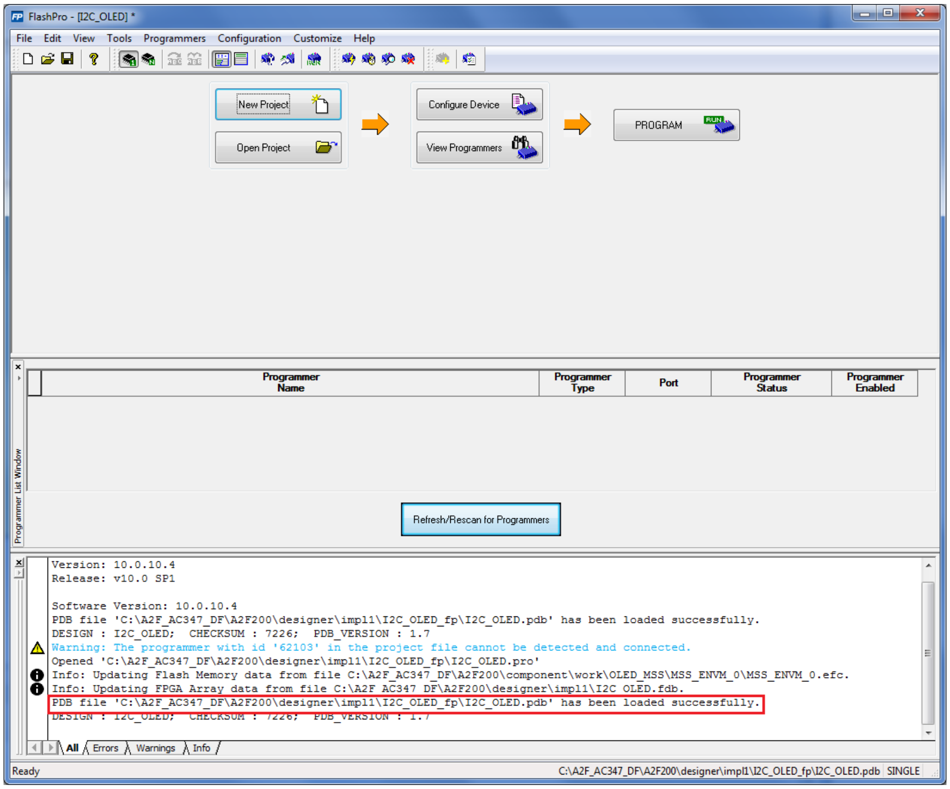
☐ Prevent read ☐ Prevent write

Figure 10 · Loading *.hex File

8. Click **OK** in the **Add Data Storage Client** window and the **MSS_ENVM_0 Configurator** window.
9. Regenerate the MSS component by clicking **SmartDesign > Generate Component**.

   Note: If you are following SoftConsole Standalone flow then skip step 11.
10. Click **Generate Programming Data** to complete the remaining steps to generate fdb file. (synthesis, place-and-route).
11. Open the FlashPro project by right-clicking on the **Program Device** icon in the Libero SoC Design Flow window and select **Open Interactively**. If you are following the SoftConsole Standalone flow then open the created FlashPro project and modify the **Programming file** by **Import** the updated Embedded Flash Memory Configuration file (.efc file) from the generated MSS component folder. Refer to the SoftConsole Standalone Flow Tutorial for more information on importing .efc file.

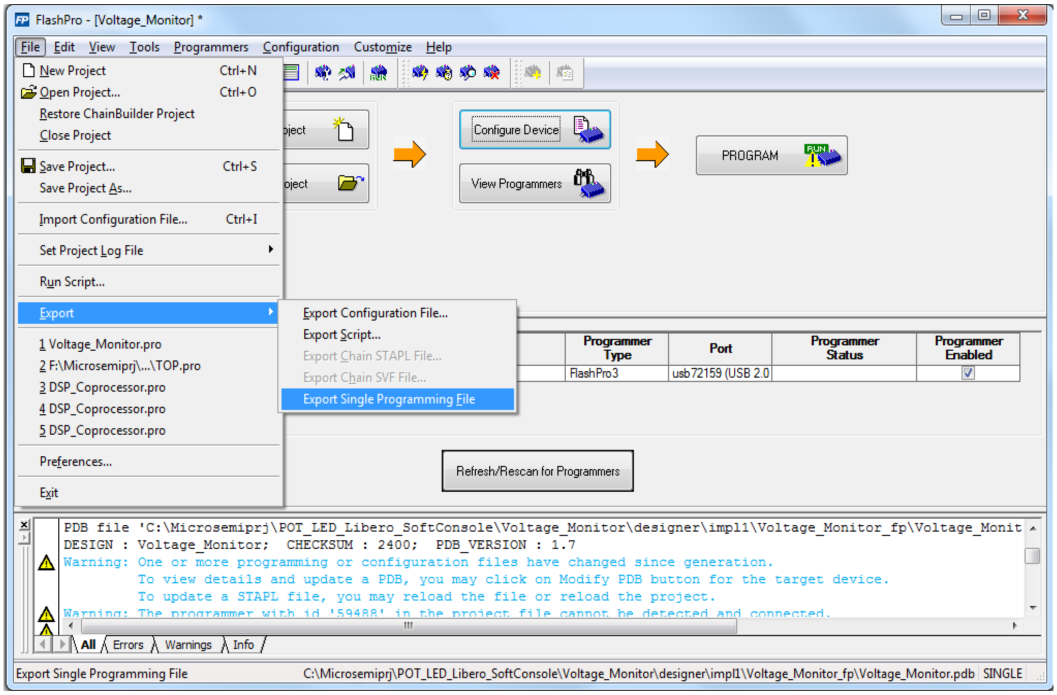
12. The FlashPro tool will be displayed as shown in the Figure 11 · .


FlashPro - [I2C_OLED] *
File Edit View Tools Programmers Configuration Customize Help
New Project
Open Project
Configure Device
View Programmers
PROGRAM
Programmer Name
Programmer Type
Port
Programmer Status
Programmer Enabled
Programmer List Window
Refresh/Rescan for Programmers
All
Errors
Warnings
Info
Ready
C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pdb
SINGLE

```
Version: 10.0.10.4
Release: v10.0 SP1

Software Version: 10.0.10.4
PDB file 'C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pdb' has been loaded successfully.
DESIGN : I2C_OLED;  CHECKSUM : 7226;  PDB_VERSION : 1.7
Warning: The programmer with id '62103' in the project file cannot be detected and connected.
Opened 'C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pro'
Info: Updating Flash Memory data from file C:\A2F_AC347_DF\A2F200\component\work\OLED_MSS\MSS_ENVM_0\MSS_ENVM_0.efc.
Info: Updating FPGA Array data from file C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED.fdb.
PDB file 'C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pdb' has been loaded successfully.
DESIGN : I2C_OLED;  CHECKSUM : 7226;  PDB_VERSION : 1.7
```

Figure 11 · PDB File Loaded by FlashPro

13. Click **File > Export > Export Single Programming File** for generating the STAPL file as shown in Figure 12 · .


FlashPro - [Voltage_Monitor] *
File
Edit
View
Tools
Programmers
Configuration
Customize
Help
New Project Ctrl+N
Open Project... Ctrl+O
Restore ChainBuilder Project
Close Project
Save Project... Ctrl+S
Save Project As...
Import Configuration File... Ctrl+I
Set Project Log File
Run Script...
Export
1 Voltage_Monitor.pro
2 F:\Microsemiprj\...\TOP.pro
3 DSP_Coprocessor.pro
4 DSP_Coprocessor.pro
5 DSP_Coprocessor.pro
Preferences...
Exit
Export Configuration File...
Export Script...
Export Chain STAPL File...
Export Chain SVF File...
Export Single Programming File
Configure Device
View Programmers
PROGRAM
RUN
Refresh/Rescan for Programmers
All
Errors
Warnings
Info

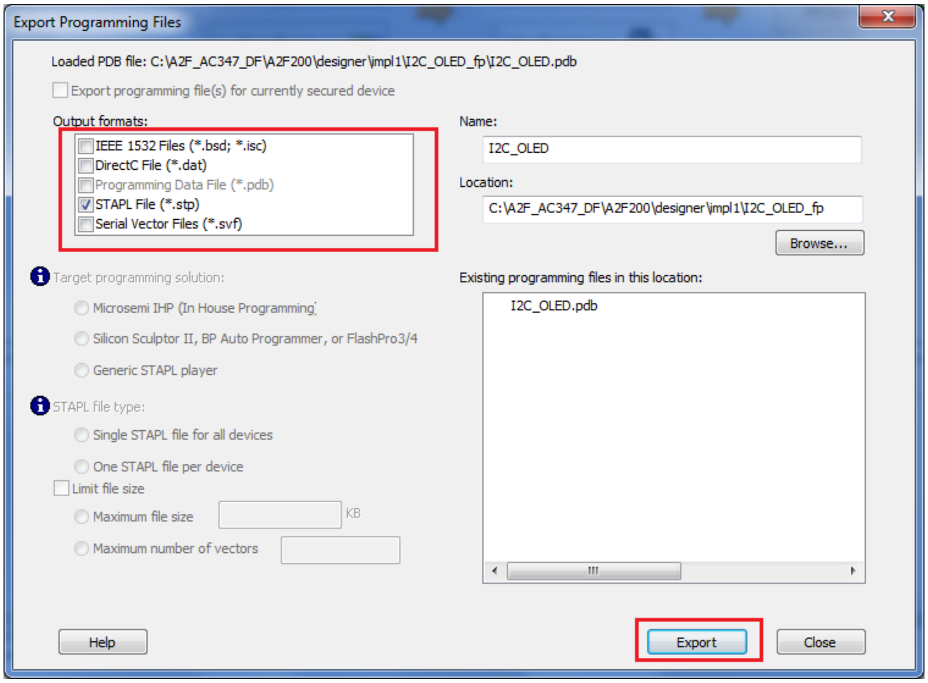
Programmer Type	Port	Programmer Status	Programmer Enabled
FlashPro3	usb72159 (USB 2.0		☑

```
PDB file 'C:\Microsemiprj\POT_LED_Libero_SoftConsole\Voltage_Monitor\designer\impl1\Voltage_Monitor_fp\Voltage_Monit
DESIGN : Voltage_Monitor;  CHECKSUM : 2400;  PDB_VERSION : 1.7
Warning: One or more programming or configuration files have changed since generation.
         To view details and update a PDB, you may click on Modify PDB button for the target device.
         To update a STAPL file, you may reload the file or reload the project.
Warning: The programmer with id '59488' in the project file cannot be detected and connected.
```

Export Single Programming File C:\Microsemiprj\POT_LED_Libero_SoftConsole\Voltage_Monitor\designer\impl1\Voltage_Monitor_fp\Voltage_Monitor.pdb SINGLE

Figure 12 · Exporting The Programming File

14. Clear all the **Output formats** except the STAPL File as shown in Figure 13 · . Click **Export** and then **Close**.


Export Programming Files
Browse...
Help
Export
Close

Loaded PDB file: C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pdb

☐ Export programming file(s) for currently secured device

Output formats:
- ☐ IEEE 1532 Files (*.bsd; *.isc)
- ☐ DirectC File (*.dat)
- ☐ Programming Data File (*.pdb)
- ☑ STAPL File (*.stp)
- ☐ Serial Vector Files (*.svf)

Target programming solution:
- ○ Microsemi IHP (In House Programming)
- ○ Silicon Sculptor II, BP Auto Programmer, or FlashPro3/4
- ○ Generic STAPL player

STAPL file type:
- ○ Single STAPL file for all devices
- ○ One STAPL file per device

☐ Limit file size
- ○ Maximum file size KB
- ○ Maximum number of vectors

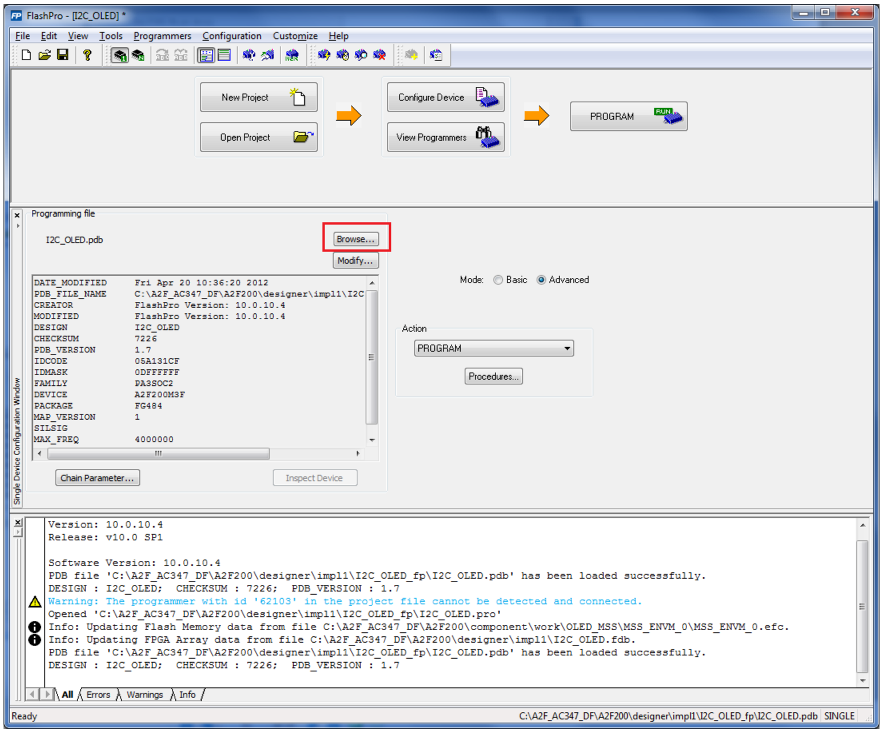
Name: I2C_OLED

Location: C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp

Existing programming files in this location:
I2C_OLED.pdb

Figure 13 · Generating STAPL File

15. Click **Configure Device** and then **Browse**.


FlashPro - [I2C_OLED] *
File Edit View Tools Programmers Configuration Customize Help
New Project
Open Project
Configure Device
View Programmers
PROGRAM
Programming file
Browse...
Modify...
Chain Parameter...
Inspect Device
Single Device Configuration Window
Action
Procedures...
All
Errors
Warnings
Info

I2C_OLED.pdb

```
DATE_MODIFIED     Fri Apr 20 10:36:20 2012
PDB_FILE_NAME     C:\A2F_AC347_DF\A2F200\designer\impl1\I2C
CREATOR           FlashPro Version: 10.0.10.4
MODIFIED          FlashPro Version: 10.0.10.4
DESIGN            I2C_OLED
CHECKSUM          7226
PDB_VERSION       1.7
IDCODE            05A131CF
IDMASK            0DFFFFFF
FAMILY            PA3SOC2
DEVICE            A2F200M3F
PACKAGE           FG484
MAP_VERSION       1
SILSIG
MAX_FREQ          4000000
```

Mode: ○ Basic ● Advanced

Action: PROGRAM

```
Version: 10.0.10.4
Release: v10.0 SP1

Software Version: 10.0.10.4
PDB file 'C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pdb' has been loaded successfully.
DESIGN : I2C_OLED;  CHECKSUM : 7226;  PDB_VERSION : 1.7
Warning: The programmer with id '62103' in the project file cannot be detected and connected.
Opened 'C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pro'
Info: Updating Flash Memory data from file C:\A2F_AC347_DF\A2F200\component\work\OLED_MSS\MSS_ENVM_0\MSS_ENVM_0.efc.
Info: Updating FPGA Array data from file C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED.fdb.
PDB file 'C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pdb' has been loaded successfully.
DESIGN : I2C_OLED;  CHECKSUM : 7226;  PDB_VERSION : 1.7
```

Ready   C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pdb SINGLE

Figure 14 · Configure the Device

16. Select the .stp file and click **Open** as shown in Figure 15 · .

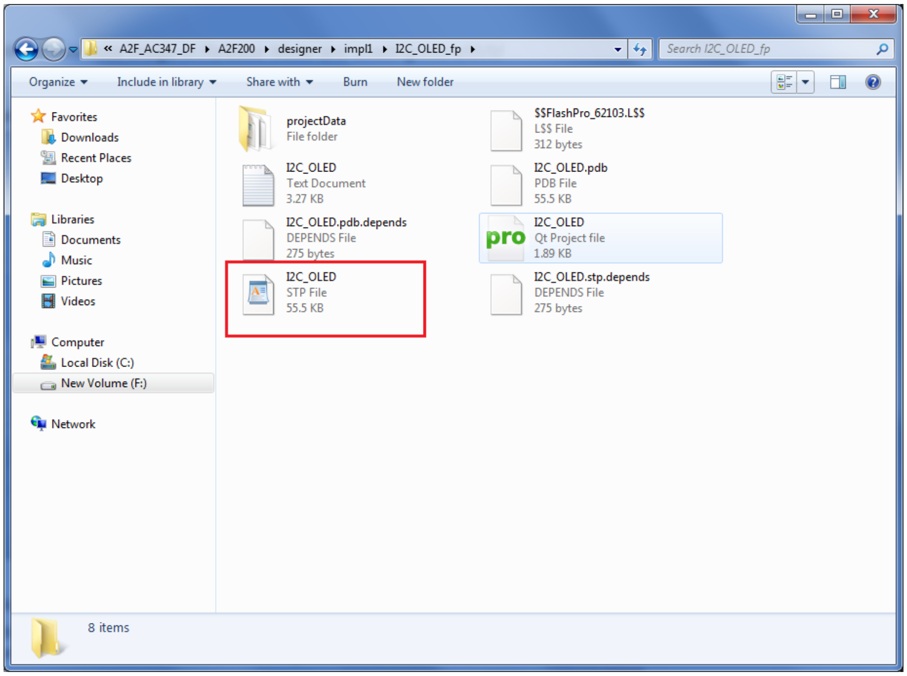
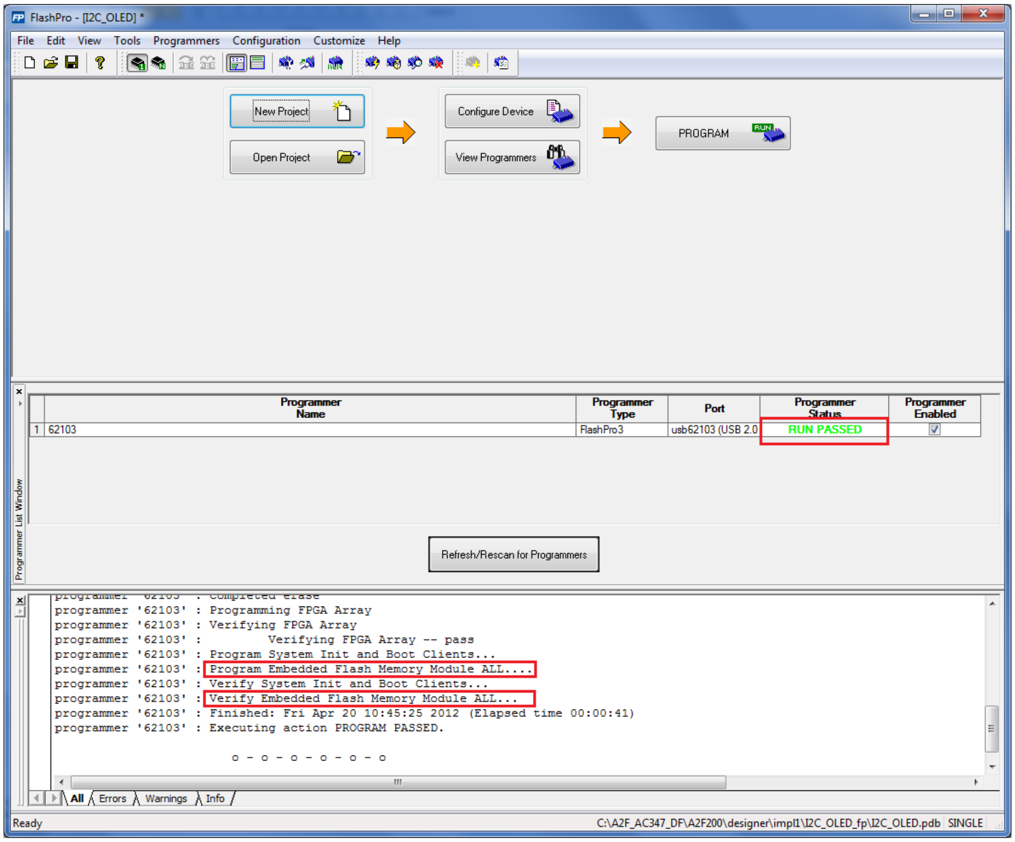

« A2F_AC347_DF ▸ A2F200 ▸ designer ▸ impl1 ▸ I2C_OLED_fp ▸
Search I2C_OLED_fp
Organize
Include in library
Share with
Burn
New folder
Favorites
Downloads
Recent Places
Desktop
Libraries
Documents
Music
Pictures
Videos
Computer
Local Disk (C:)
New Volume (F:)
Network
projectData
File folder
I2C_OLED
Text Document
3.27 KB
I2C_OLED.pdb.depends
DEPENDS File
275 bytes
I2C_OLED
STP File
55.5 KB
$$FlashPro_62103.L$$
L$$ File
312 bytes
I2C_OLED.pdb
PDB File
55.5 KB
I2C_OLED
Qt Project file
1.89 KB
I2C_OLED.stp.depends
DEPENDS File
275 bytes
8 items

Figure 15 · Selecting .stp File

17. Click **PROGRAM** to program the device. Observe the FlashPro log window showing ‘Program Embedded Flash Memory Module ALL...’ command execution as shown in the Figure 16 · .


FlashPro - [I2C_OLED] *
File
Edit
View
Tools
Programmers
Configuration
Customize
Help
New Project
Open Project
Configure Device
View Programmers
PROGRAM
RUN
Programmer List Window
Refresh/Rescan for Programmers
All
Errors
Warnings
Info
Ready
C:\A2F_AC347_DF\A2F200\designer\impl1\I2C_OLED_fp\I2C_OLED.pdb
SINGLE

	Programmer Name	Programmer Type	Port	Programmer Status	Programmer Enabled
1	62103	FlashPro3	usb62103 (USB 2.0	RUN PASSED	☑

```
programmer '62103' : Completed erase
programmer '62103' : Programming FPGA Array
programmer '62103' : Verifying FPGA Array
programmer '62103' :         Verifying FPGA Array -- pass
programmer '62103' : Program System Init and Boot Clients...
programmer '62103' : Program Embedded Flash Memory Module ALL....
programmer '62103' : Verify System Init and Boot Clients...
programmer '62103' : Verify Embedded Flash Memory Module ALL...
programmer '62103' : Finished: Fri Apr 20 10:45:25 2012 (Elapsed time 00:00:41)
programmer '62103' : Executing action PROGRAM PASSED.

                          o - o - o - o - o - o
```

Figure 16 · FlashPro Programming

18. Once programming is done, save the project and close FlashPro project.
19. Remove the USB cable for the programmer and restart the board to run the application.

    Note: You need not run the debugger using SoftConsole since the code is already programmed into eNVM using FlashPro.

List of Changes

Revision	Changes	Page
Revision 2 (April 2012)	Updated screenshots and content for Libero SoC SP1 (SAR 38221).	NA
Revision 1 (November 2011)	Updated screenshot for Libero SoC and support for STAPL File generation (SAR 34981).	7, 11

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.