

Blinking LED Tutorial for the Hello FPGA Kit

Written by: Dan Grise

2020-11-16

Purpose:

To provide a simple tutorial to help the user generate a design targeting the Hello FPGA board.

This tutorial is good for people with little to no experience with the Libero SoC software.

Assumptions:

To run this tutorial the following software must be installed:

Libero SoC (Version 12.0 or later is recommended)

Soft Console (Version 4.0 or later is recommended)

Hello FPGA GUI

The user has access to the Hello FPGA development board

Disclaimer:

The author is not affiliated with Microsemi or Microchip and is not an expert with Libero SoC.

This document was written to help users get over some of the learning curve hurdles associated with trying out new development platforms.

And yes, I acknowledge that the formatting of this document is sparse and bare bones. All the same I hope it is helpful to those that read it.

Setting up Project

Open the Libero SoC software

Click Project → New Project

Enter first_processor as the Project name

Choose a suitable folder for Project location

You can leave Description blank

Choose your preferred HDL language

Leave Enable block creation blank

Click Next

The screenshot shows the 'New project' dialog box with the 'Project details' tab selected. The dialog has a sidebar on the left with a vertical list of steps: 'Project Details' (highlighted), 'Device Selection', 'Device Settings', 'Design Template', 'Add HDL Sources', and 'Add Constraints'. The main area contains the following fields and options:

- Project name:** A text box containing 'first_processor'.
- Project location:** A text box containing 'C:/Users/Dan/Documents/Libero/' with a 'Browse...' button to its right.
- Description:** A large empty text box.
- Preferred HDL type:** A dropdown menu with 'VHDL' selected.
- Enable block creation:** An unchecked checkbox.
- Block flow description:** A paragraph of text explaining that block flow enables publishing reusable components that can be instantiated into another design, but may not contain I/O cells and cannot be programmed. It could include timing constraints, physical constraints, placement or routing.

At the bottom of the dialog, there is a 'Help' button on the left and a set of navigation buttons on the right: '< Back', 'Next >', 'Finish', and 'Cancel'.

Select M2S010-1VF256 as your device

Click Next

New project

Device selection
Select a part for your project from the part number list

Selected part: M2S010-1VF256

Project Details

Device Selection

Device Settings

Design Template

Add HDL Sources

Add Constraints

Libero
System-on-Chip

Part filter

Family: SmartFusion2 Die: All Package: All

Speed: All Core voltage: All Range: All

Reset filters

Search part:

Part Number /	4LUT	DFF	User I/Os	uSRAM 1K	LSRAM 18K	Math (18x18)	PLLs and
M2S005-FG484	6060	6060	209	11	10	11	2
M2S005-FG484I	6060	6060	209	11	10	11	2
M2S005-TQ144	6060	6060	84	11	10	11	2
M2S005-TQ144I	6060	6060	84	11	10	11	2
M2S005-VF256	6060	6060	161	11	10	11	2
M2S005-VF256I	6060	6060	161	11	10	11	2
M2S005-VF400	6060	6060	171	11	10	11	2
M2S005-VF400I	6060	6060	171	11	10	11	2
M2S010-1FG484	12084	12084	233	22	21	22	2
M2S010-1FG484I	12084	12084	233	22	21	22	2
M2S010-1TQ144	12084	12084	84	22	21	22	2
M2S010-1TQ144I	12084	12084	84	22	21	22	2
M2S010-1VF256	12084	12084	138	22	21	22	2
M2S010-1VF256I	12084	12084	138	22	21	22	2

< Back Next > Finish Cancel

For Default I/O technology select LVCMOS 3.3V

Check Reserve pins for probes

Select 3.3 for PLL supply voltage

Leave System controller suspended mode unchecked

Click Next

New project

Device settings
Choose device settings for your project

Selected part: M25010-1VF256

Project Details

Device Selection

Device Settings

Design Template

Add HDL Sources

Add Constraints

Libero
System-on-Chip

I/O settings

Default I/O technology: LVCMOS 3.3V Please use the I/O Editor to change individual I/O attributes.

☒ Reserve pins for probes

Power supplies

PLL supply voltage (V): 3.3

VDD Supply Ramp Time: 100ms

☐ System controller suspended mode

Help

< Back

Next >

Finish

Cancel

For Design templates and creators select None

Leave other boxes unchecked

Click Next

New project

Design Template
Choose a design template

Selected part: M25010-1VF256

Project Details

Device Selection

Device Settings

Design Template

Add HDL Sources

Add Constraints

Libero
System-on-Chip

Design templates and creators

- ☒ None
- ☐ Create a system builder based design
- ☐ Create a microcontroller(MSS) based design

Core	Version
SmartFusion2 Microcontroller Subsystem (MSS)	1.1.500

☒ Show only latest version

Design methodology

- ☐ Use standalone initialization for MDDR/FDDR/SERDES peripherals
- ☐ Instantiate SystemBuilder/MSS component in a SmartDesign on creation

Help

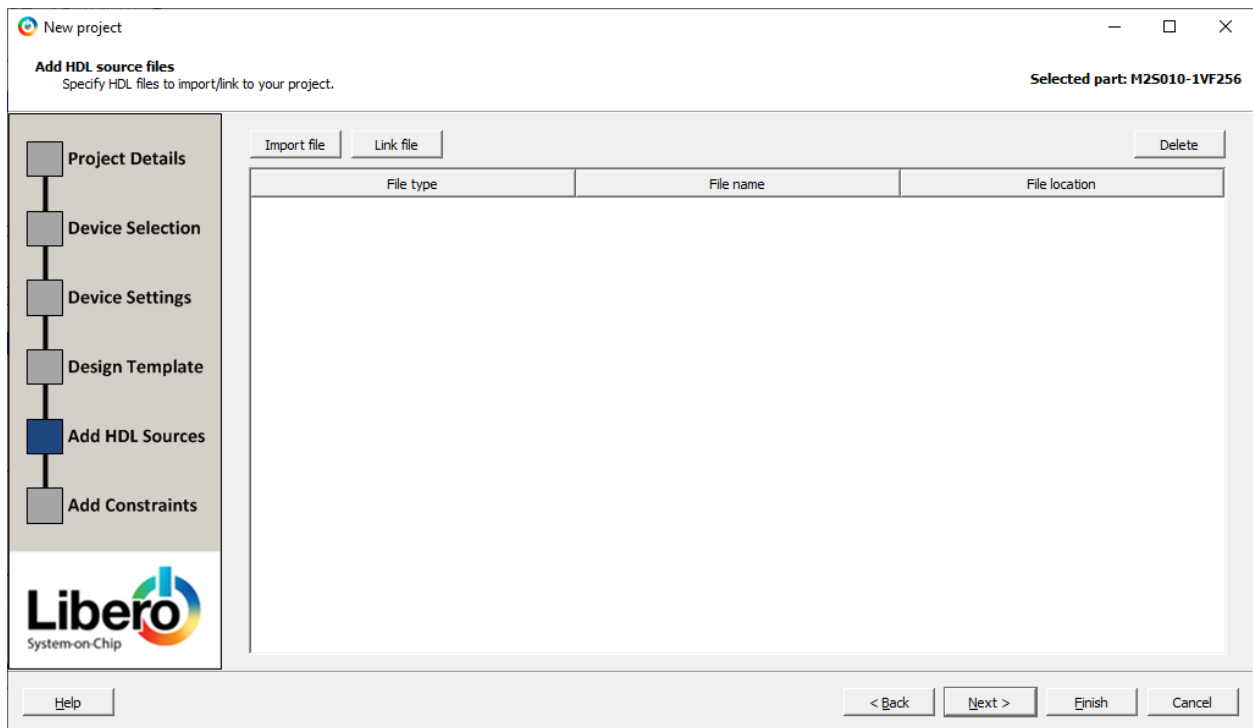
< Back

Next >

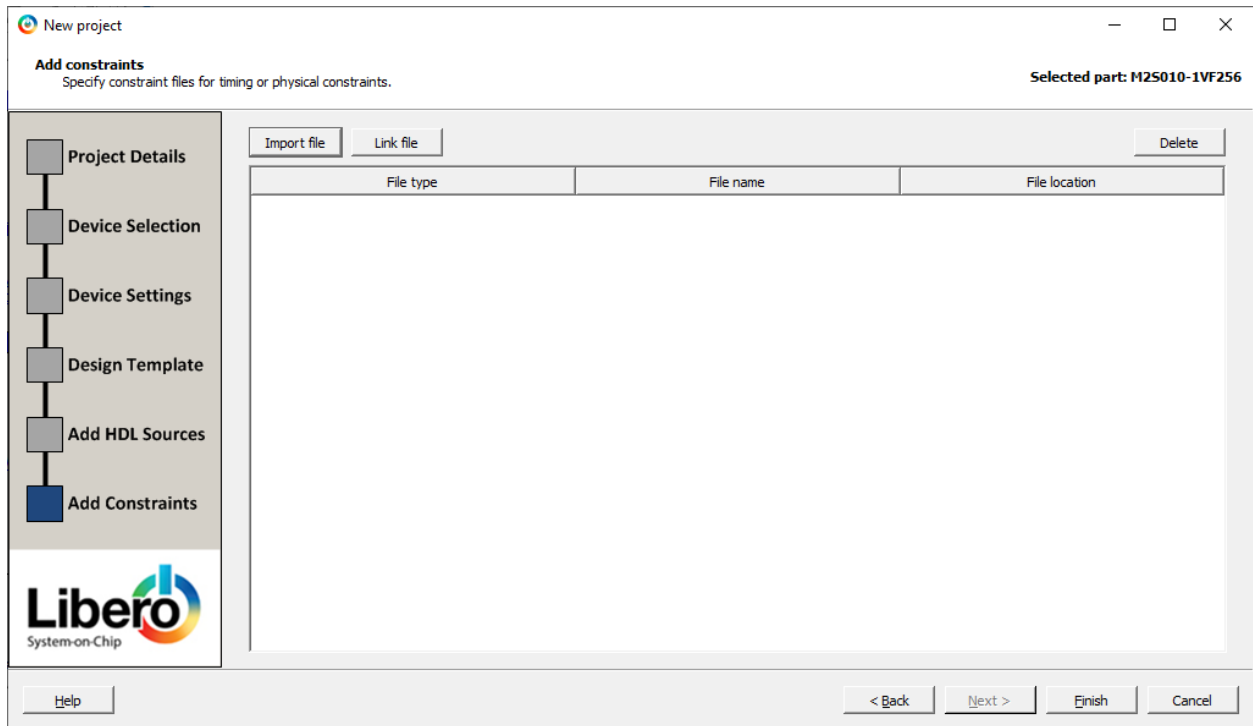
Finish

Cancel

When Add HDL screen emerges, click Next



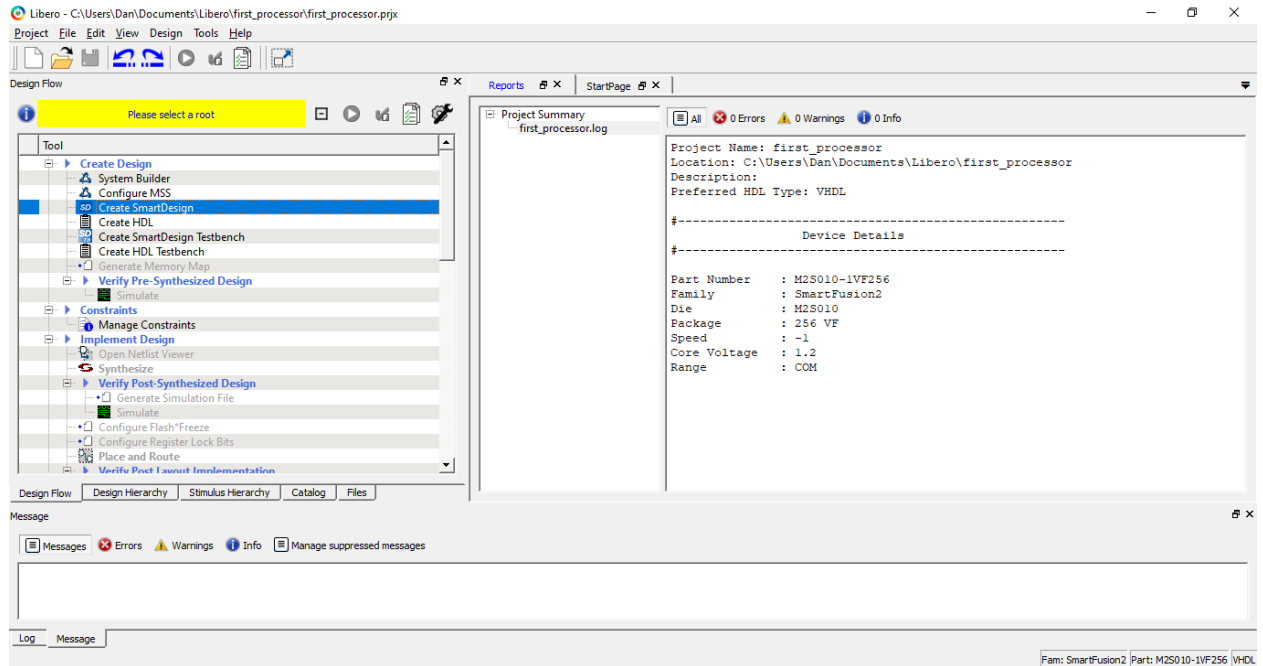
When Add Constraints window emerges, click Finish



Creating Smart Design

In the Design Flow tab double click Create SmartDesign

Enter top as the Design name and click OK



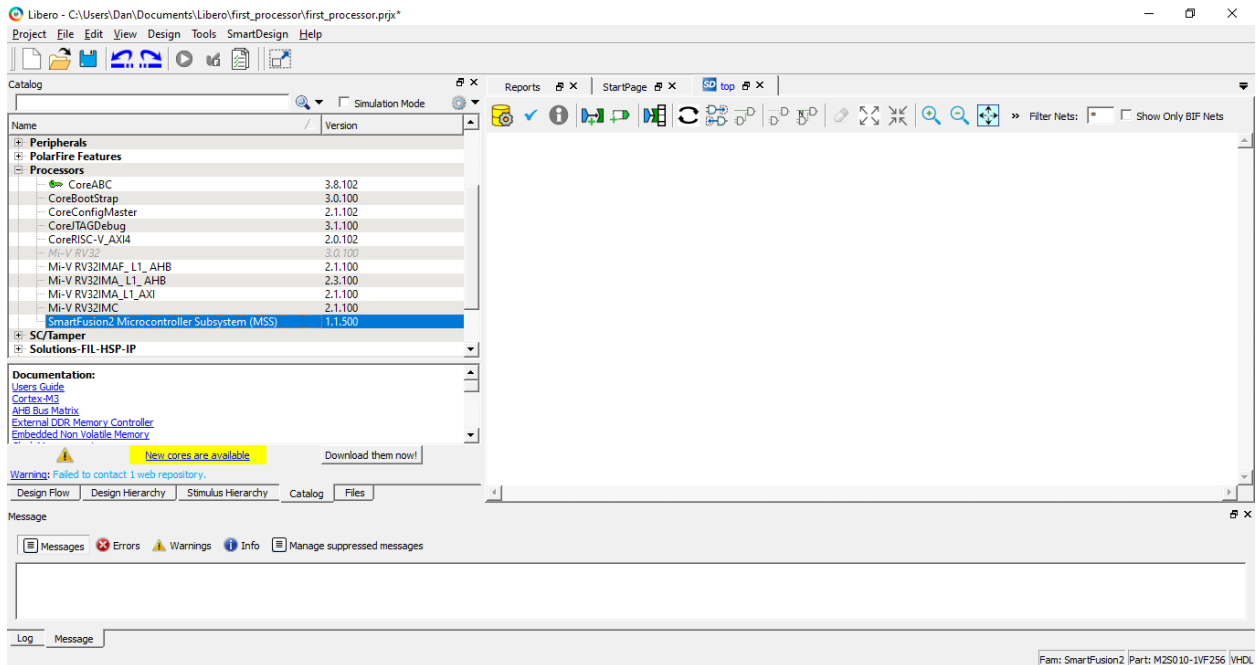
Select Catalog tab

Expand Processors

Select SmartFusion2 Microcontroller Subsystem (MSS)

Note: if SmartFusion2 Microcontroller Subsystem (MSS) is greyed out, select it, right click and click download

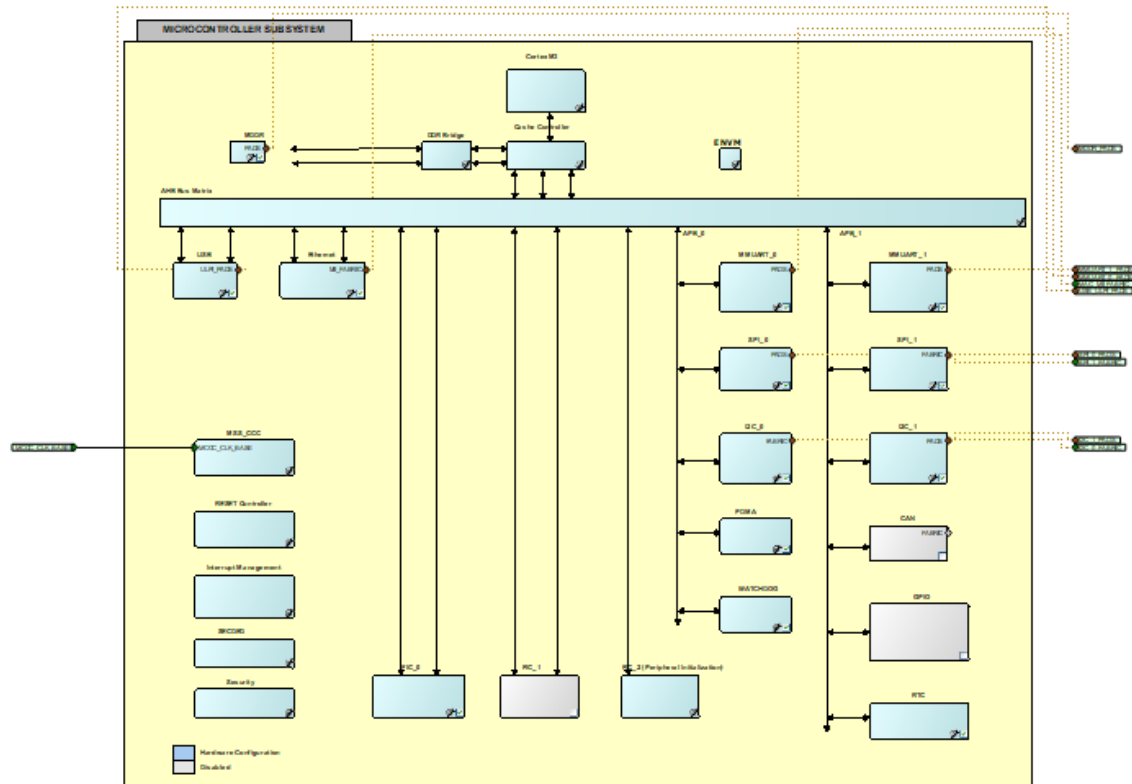
Drag SmartFusion2 Microcontroller Subsystem (MSS) towards the right



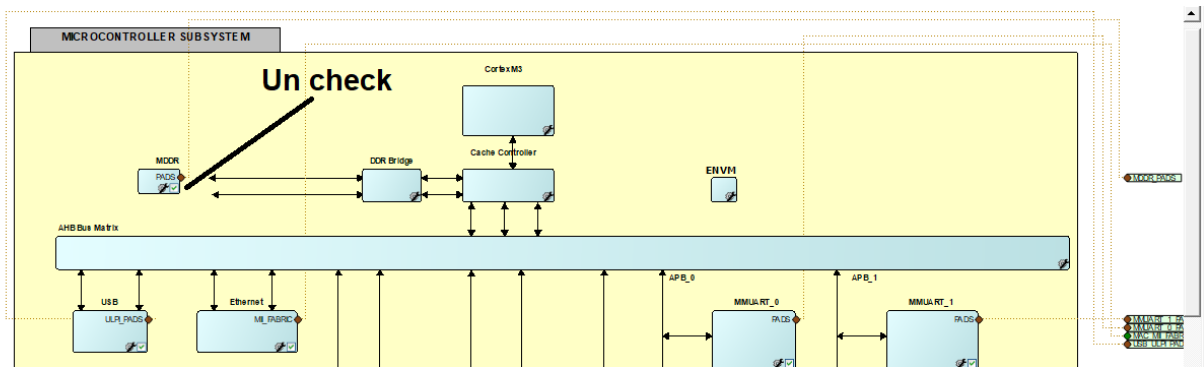
Accept default name and press OK

The MSS_C0 smart design emerges

It should look similar to the image that follows:



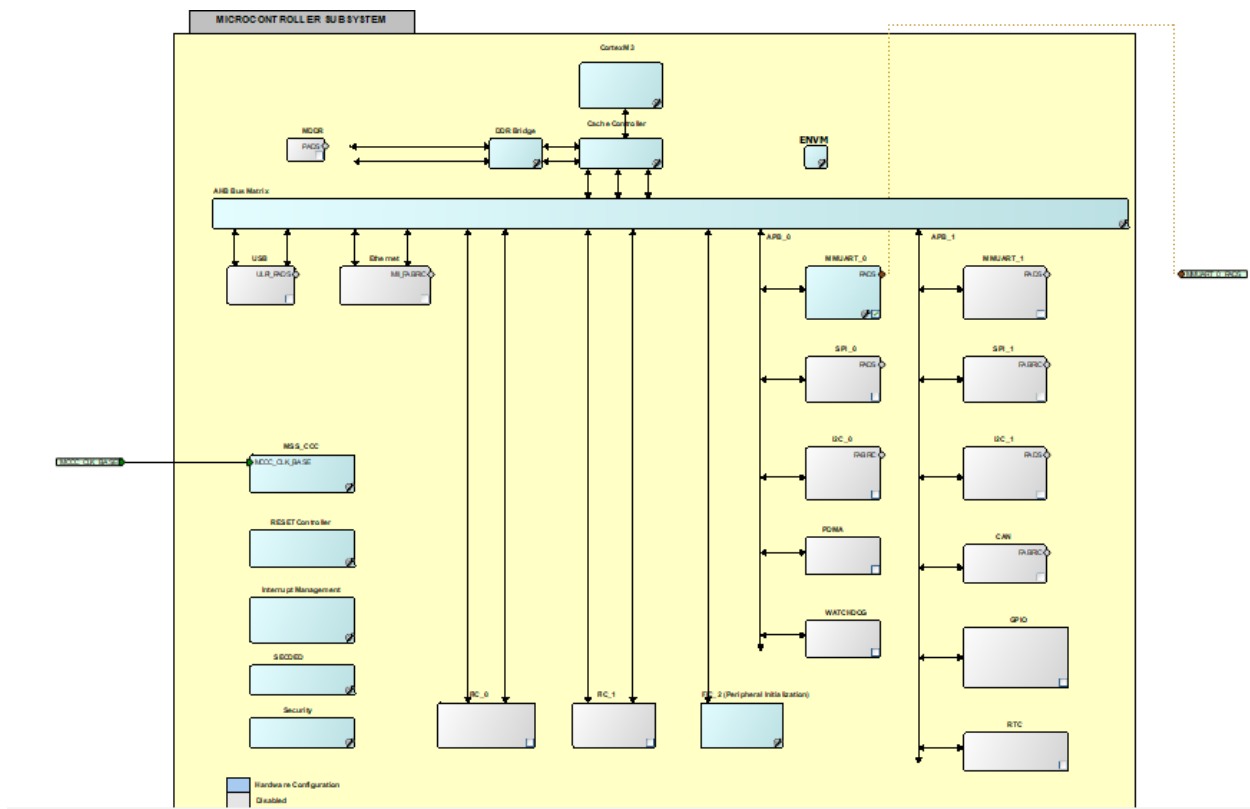
Disable MDDR by unchecking it



In a similar fashion disable the following:


- MMUART_1, SPI_1, I2C_1, CAN, RTC
- SPI_0, I2C_0, PDMA, WATCHDOG
- USB, Ethernet, RC_0, RC_1

When you are finished your MSS should look like the image that follows:



Double click MSS_CCC

Verify that the settings are as follows and press OK to exit

 MSS Clock Conditioning Circuitry Configurator

System Clocks Advanced Options

Clock Source

CLK_BASE 50.000 MHz

☐ Monitor FPGA Fabric PLL Lock (CLK_BASE_PLL_LOCK)

Cortex-M3 and MSS Main Clock

[M3_CLK](#) 100.000 MHz 100.000 MHz

MDDR Clocks

[MDDR_CLK](#) = M3_CLK * 2 ▼

[DDR_SMC_FIC_CLK](#) = MDDR_CLK / 1 ▼

MSS APB_0/1 Sub-busses Clocks

✓ [APB_0_CLK](#) = M3_CLK / 1 ▼ 100.000 MHz

✓ [APB_1_CLK](#) = M3_CLK / 1 ▼ 100.000 MHz

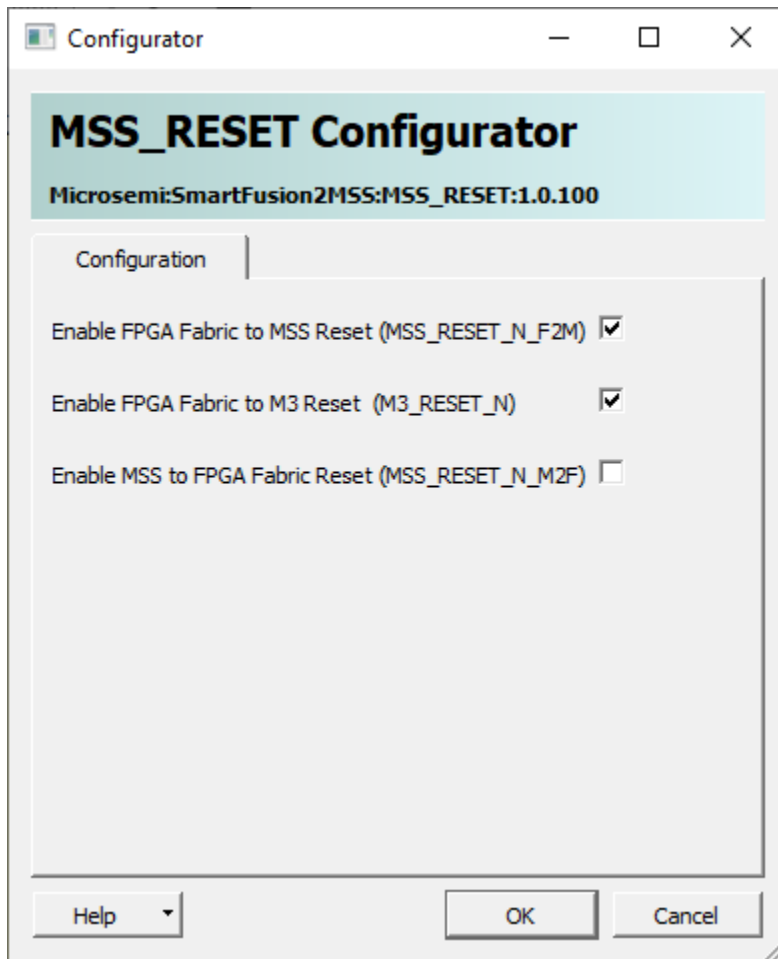
FPGA Fabric Interface Clocks

[FIC_0_CLK](#) = M3_CLK / 1 ▼

[FIC_1_CLK](#) = M3_CLK / 1 ▼

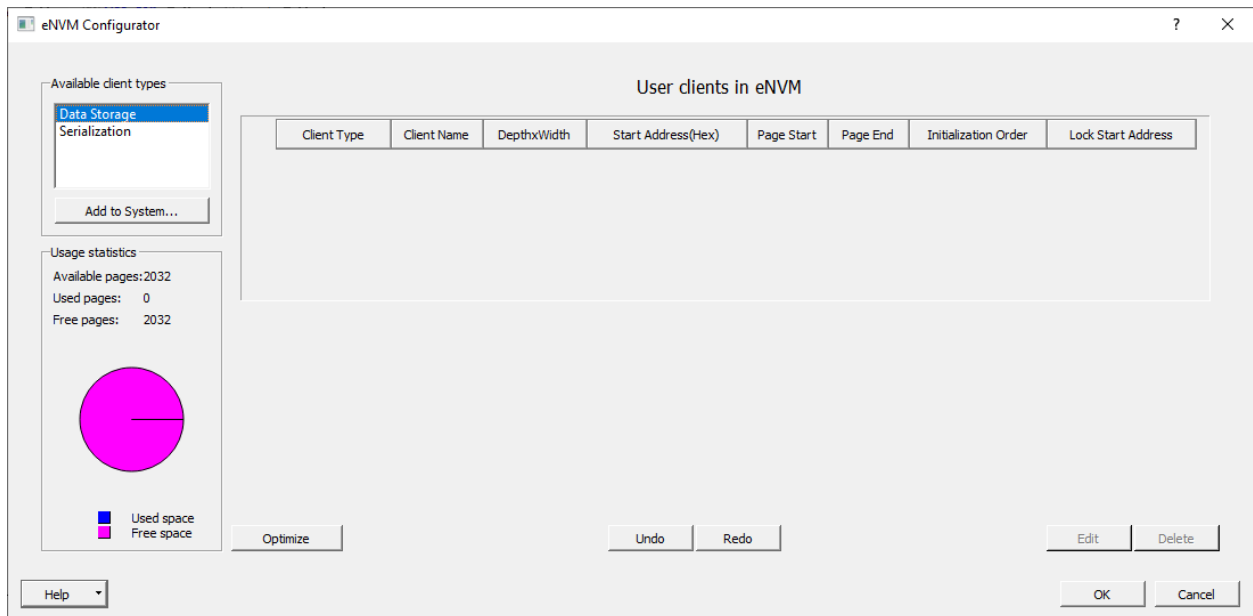
Double click RESET Controller

Apply check box options to match picture below and click OK



Double click ENVN

Double click Data Storage



Enter ENVM as Client Name

Select Intel-Hex format

Un-check Use absolute addressing

Select No content (place holder) option

Leave Start Address as 0

Set size of word to 8 Bits

Set Number of words to 2000

Leave other options unchecked

Click OK

Add Data Storage Client

Client name: ENVM

eNVM

☐ Content from file: ...

Format: Intel-Hex

☐ Use absolute addressing ⓘ

☐ Content filled with 0s

☒ No content (client is a placeholder and will not be programmed)

Start address: 0x 0

Size of word: 8 Bits

Number of words: 2000 Decimal

☐ Use as ROM ⓘ

☐ Use content for simulation

Help OK Cancel

Double click MMUART_0


Ensure Duplex Mode is set to Full Duplex

Ensure Async/Sync Mode is set to Asynchronous

For RXD set main connection to Fabric

For TXD set main connection to Fabric

Click OK

 **MSS MMUART_0 Configurator**

Configuration

Duplex Mode Full Duplex ▼

Async/Sync Mode Asynchronous ▼

Use Modem Interface ☐

Assignment Advanced Options ☐

	MSIO	Direction	Main Connection	Package Pin
	Full Duplex			
	RXD	IN	Fabric ▼	
	TXD	OUT	Fabric ▼	


Apply Checkmark to GPIO to enable it

Double click GPIO

Set GPIO_0 Direction to Output

Set GPIO_0 Connectivity to FABRIC_A

Click OK

 MSS GPIO Configurator

Configuration

Set/Reset Definition

GPIO_31_24 Reset Source	SYSREG (MSS_GPIO_31_24_SOFT_RESET) ▼	Reset State	1 ▼
GPIO_23_16 Reset Source	SYSREG (MSS_GPIO_23_16_SOFT_RESET) ▼	Reset State	1 ▼
GPIO_15_8 Reset Source	SYSREG (MSS_GPIO_15_8_SOFT_RESET) ▼	Reset State	1 ▼
GPIO_7_0 Reset Source	SYSREG (MSS_GPIO_7_0_SOFT_RESET) ▼	Reset State	1 ▼

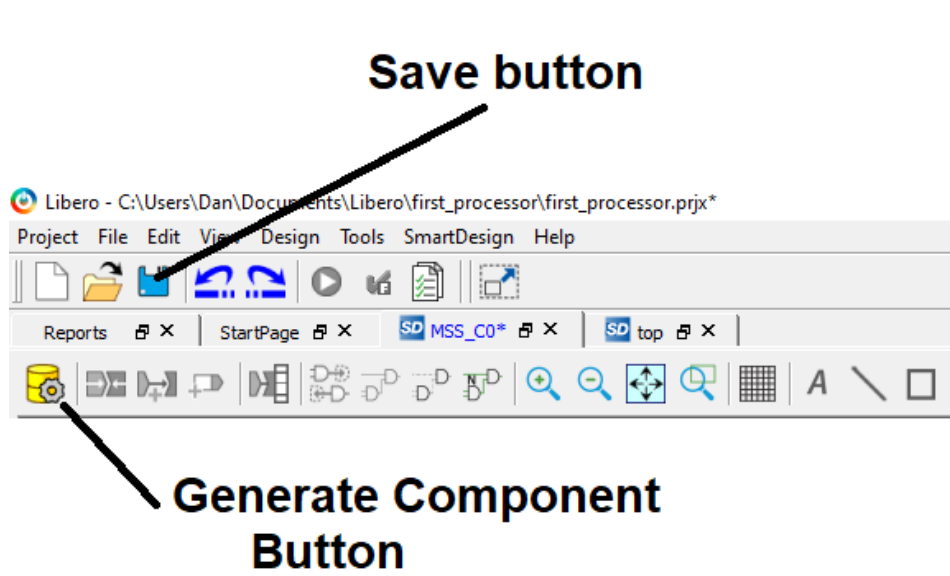
GPIO Assignment

Advanced Options ☐

GPIO ID	Direction	Package Pin	Connectivity
GPIO_0	Output ▼		FABRIC_A ▼
GPIO_1	Not Used ▼		IO_A ▼
GPIO_2	Not Used ▼		FABRIC_A ▼
GPIO_3	Not Used ▼		FABRIC_A ▼
GPIO_4	Not Used ▼		FABRIC_A ▼
GPIO_5	Not Used ▼		IO_A ▼
GPIO_6	Not Used ▼		IO_A ▼
GPIO_7	Not Used ▼		IO_A ▼
GPIO_8	Not Used ▼		IO_A ▼
GPIO_9	Not Used ▼		IO_A ▼

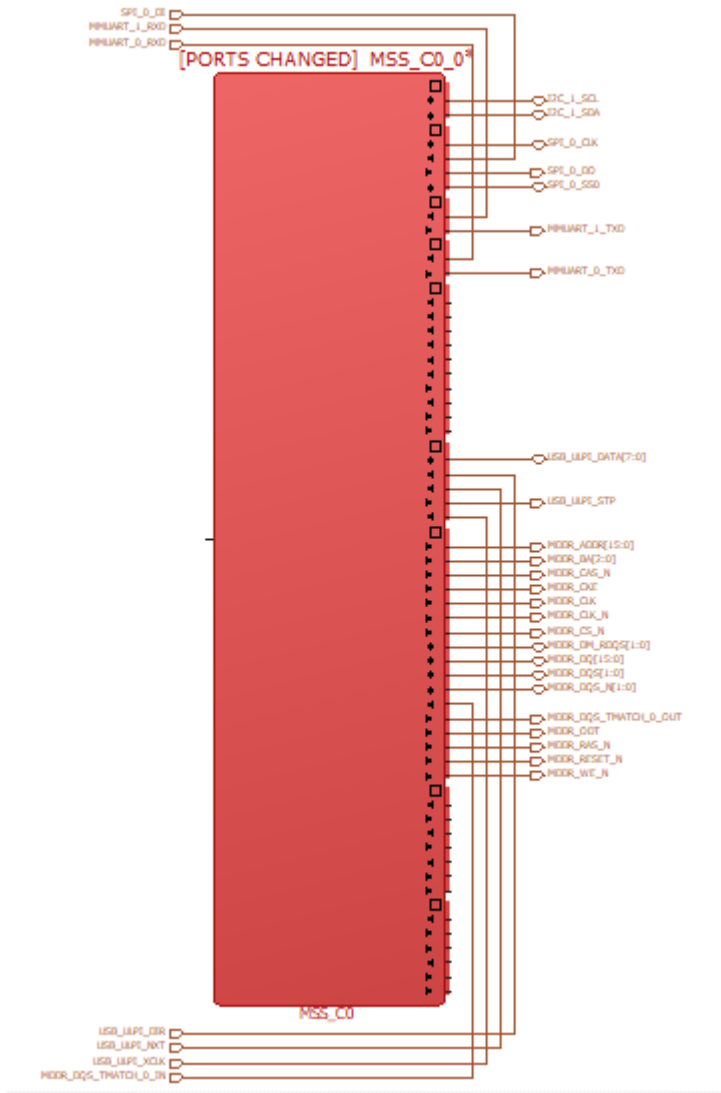
Save MSS_C0

Click Generate Component



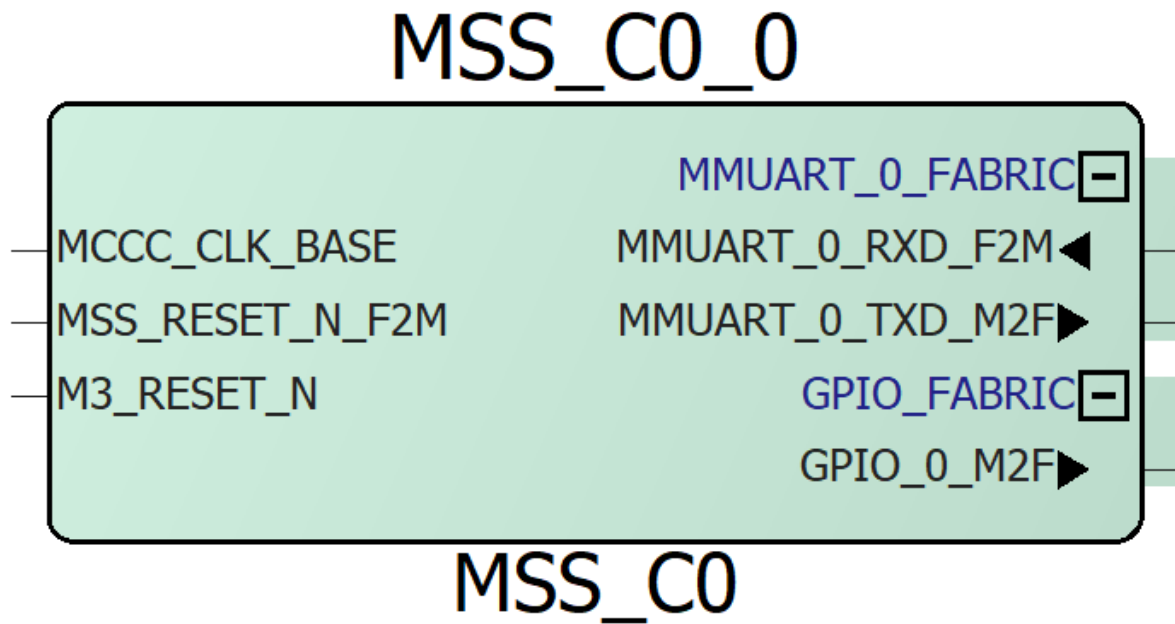
Go to top smart design

The symbol for MSS_CO_0 will be red (because it needs to be updated)



Right click and select Update Component

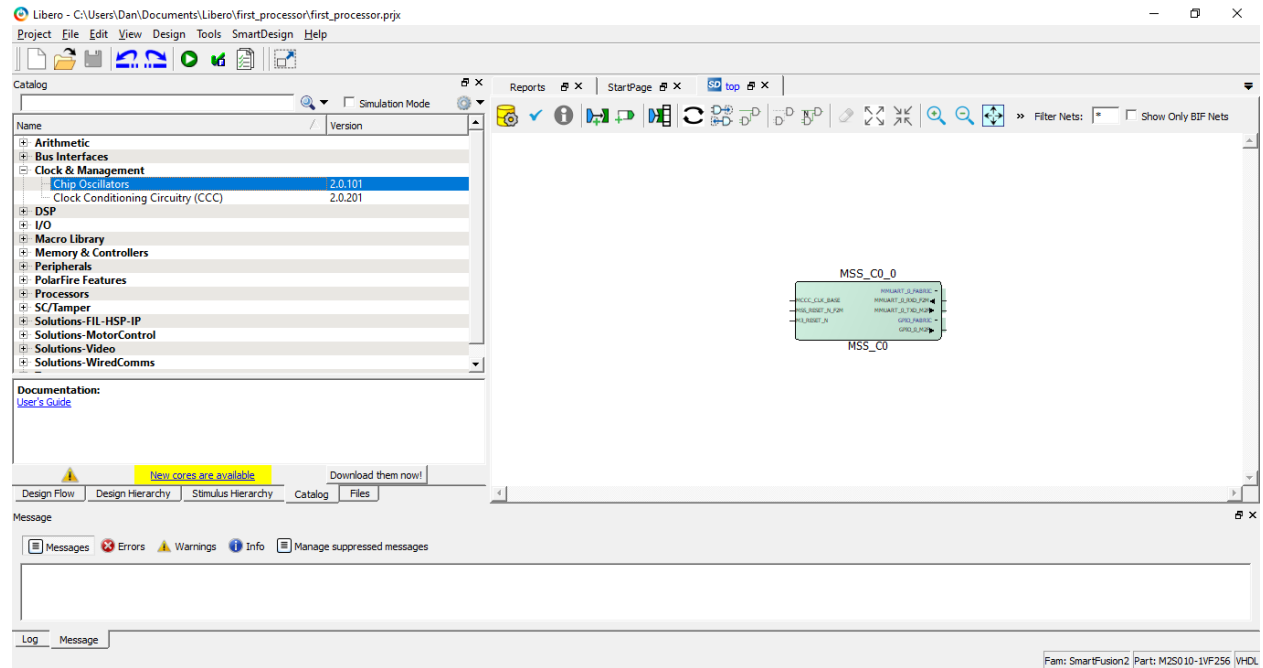
Afterwards the component should look like the image that follows



Select Catalog tab

Expand Clock & Management

Select Chip Oscillators



Drag Chip Oscillators into top

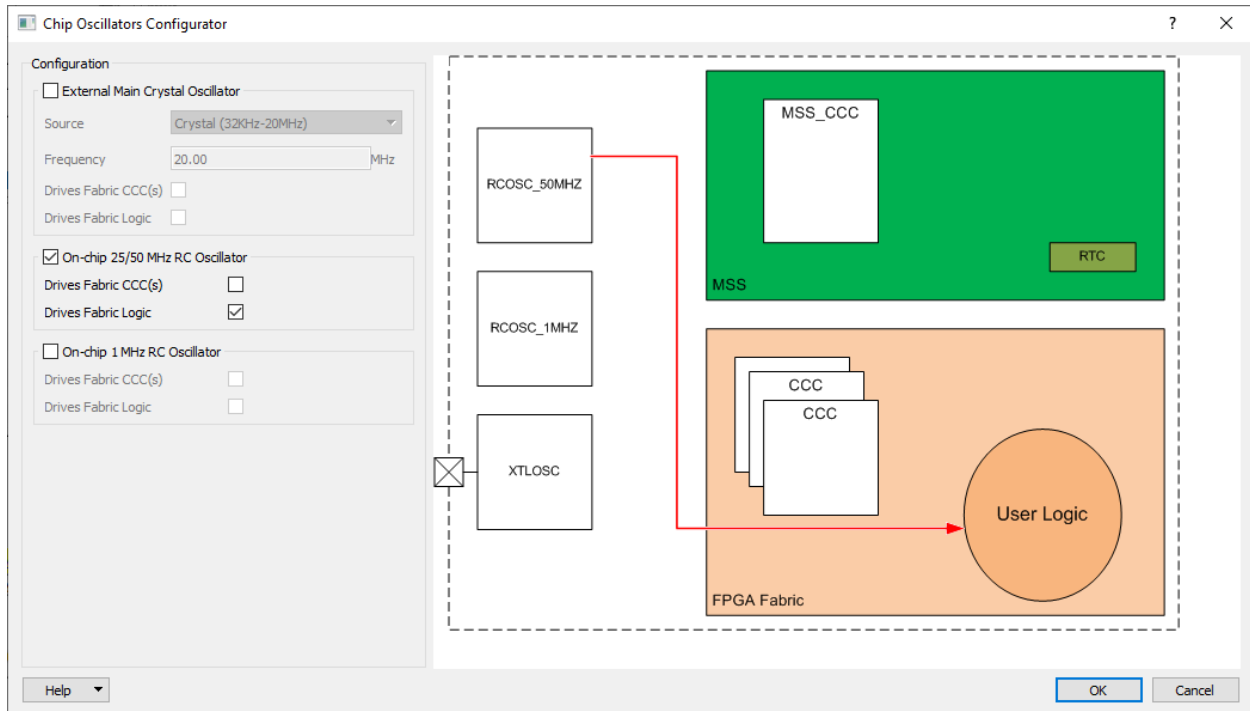
Accept default name (OSC_C0) and click OK

The Chip Oscillators Configurator window will pop up

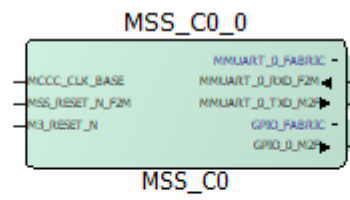
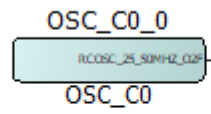
Apply Checkmark to select On-chip 25/50 MHz RC Oscillator

Apply Checkmark to Drives Fabric Logic Option

Press OK



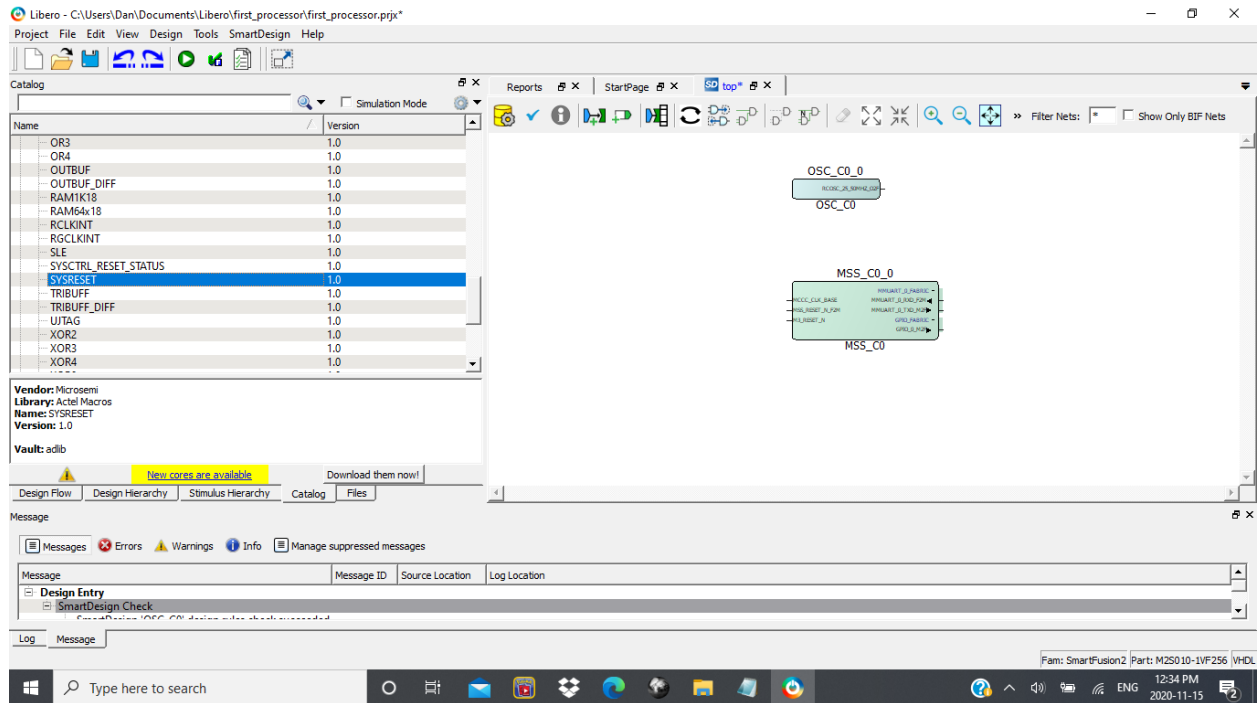
Oscillator OSC_C0_0 will now appear in top



Select Catalog tab

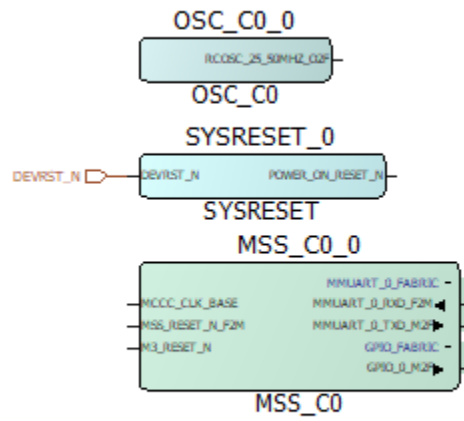
Expand Macro Library

Select SYSRESET



Drag SYSRESET into top

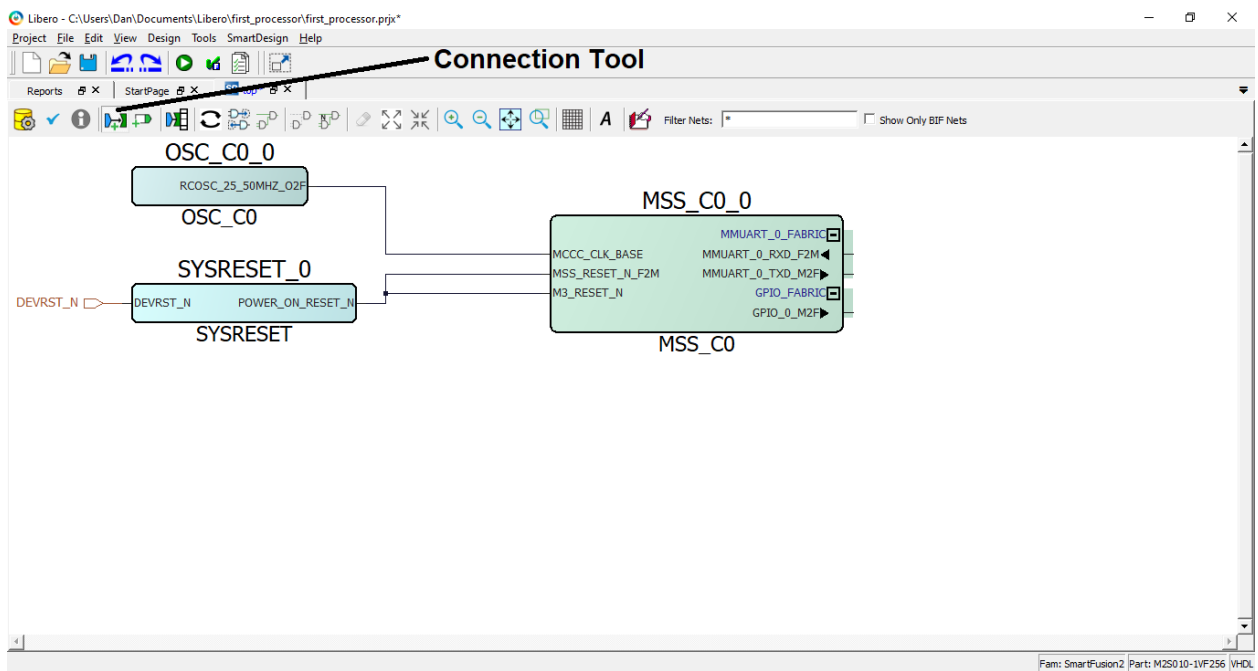
SYS_RESET_0 should now appear in top



Rearrange blocks similar to example pictured below



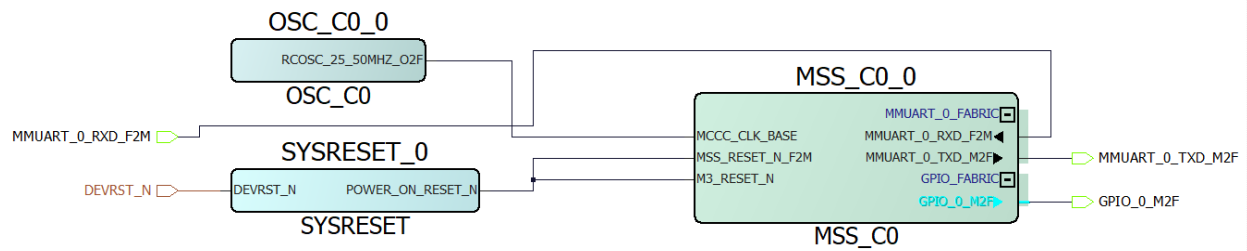
Use connection tool and make connections shown below



Right click the MMUART_0_RXD_F2M pin and select promote to top level

Do the same with the MMUART_0_TXD_M2F and GPIO_0_M2F pins

The Smart Design should now resemble the example below

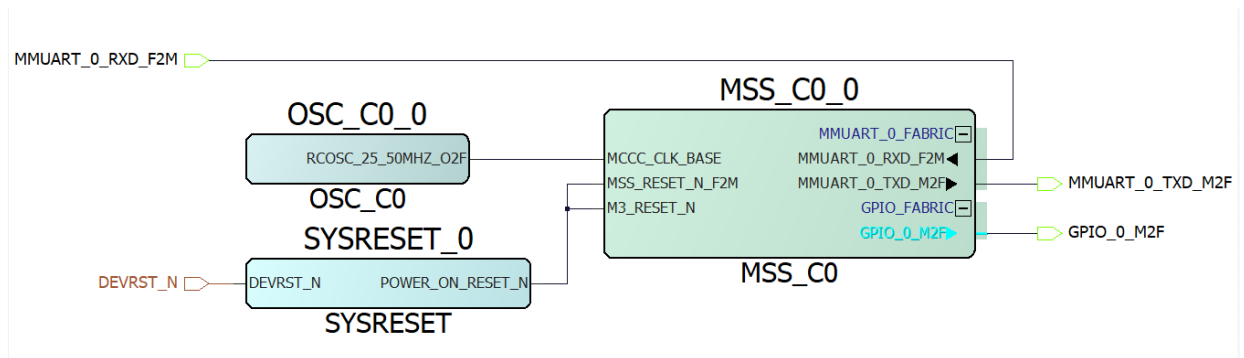


Unselect connection tool

Click auto-arrange



You will notice that things got prettier



Save top

Click generate

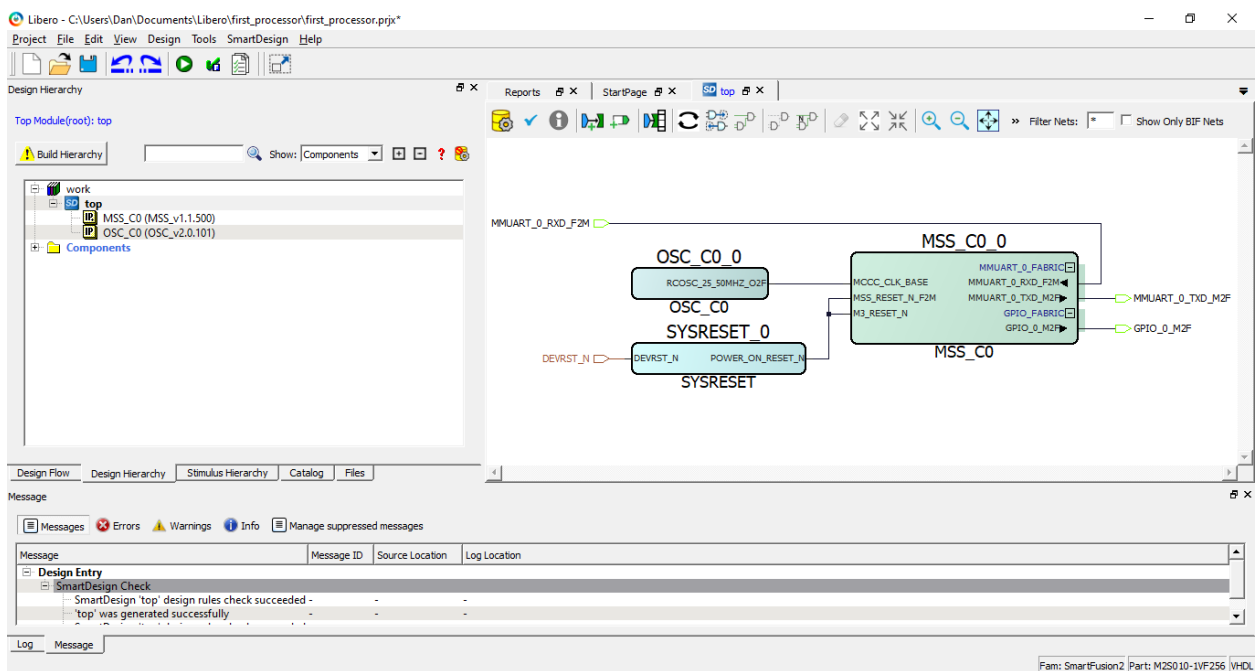


Select Design Hierarchy tab

Right click top

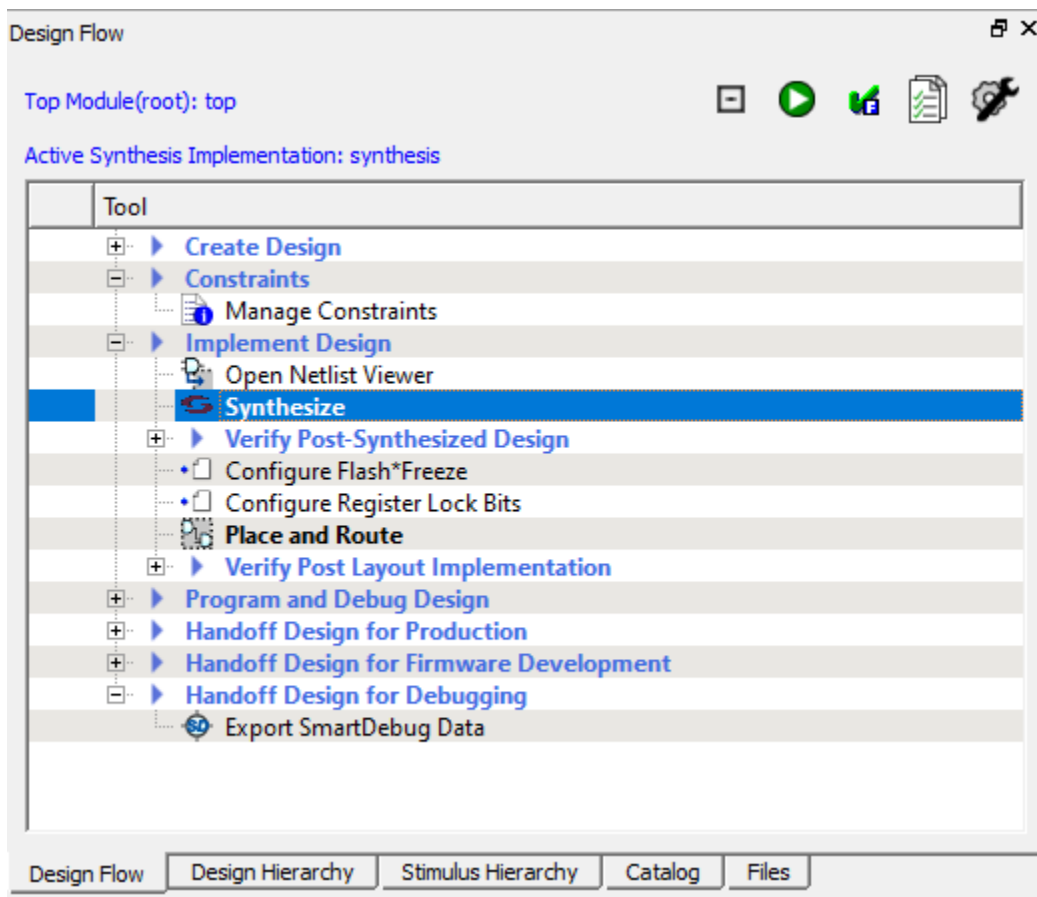
Select Set as Root

Now top should be recognized as the true top level



Double click Synthesize

When complete a check box will appear next to Synthesize



Double click Manage Constraints

In Constraint Manager select I/O Attributes tab

Click Edit

Then click edit with I/O Editor

Set pin number of GPIO_0_M2F to T12

Set pin number of MMUART_0_RXD_F2M to H13

Set pin number of MMUART_0_TXD_M2F to H12

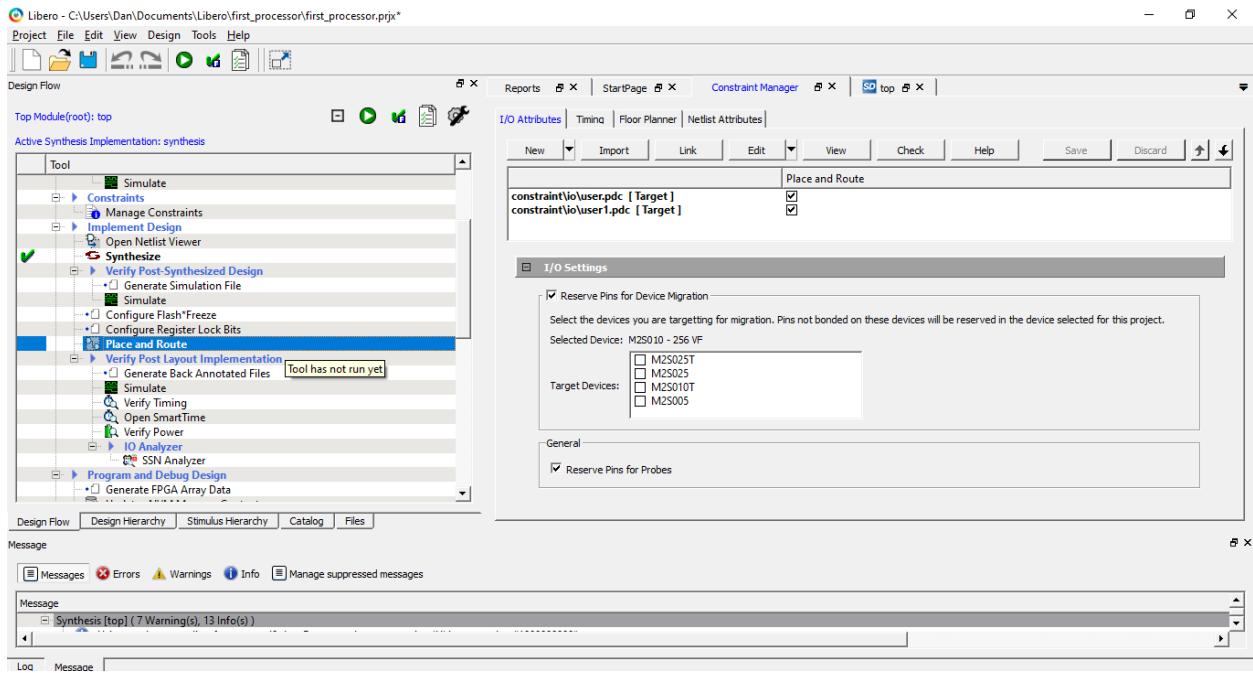
Refer to diagram below to double check pin assignments

Save I/O Editor

Port View [active] Pin View Package View Floorplanner View Netlist Viewer - Hier						
	Port Name	Direction	I/O Standard	Pin Number	Locked	
1	DEVRST_N	INPUT	--	M11	<input checked="" type="checkbox"/>	
2	GPIO_0_M2F	OUTPUT	LVC MOS33	T12	<input checked="" type="checkbox"/>	
3	MMUART_0_RXD_F2M	INPUT	LVC MOS33	H13	<input checked="" type="checkbox"/>	
4	MMUART_0_TXD_M2F	OUTPUT	LVC MOS33	H12	<input checked="" type="checkbox"/>	

Double click Place and Route

When Complete a check mark will appear next to Place and Route



Right click Configure Firmware Cores and select Open Interactively

The screenshot shows the Microsemi Libero IDE interface. The main window displays the 'Global Net Report' for the project 'top'. The report includes a 'Global Nets Information' table and an 'I/O to GB Connections' section.

Global Net Report

Microsemi Corporation - Microsemi Libero Software Release v12.5 (Version 12.900.10.16)
Date: Sun Nov 15 14:00:14 2020

Global Nets Information

	From	GB Location	Net Name	Fanout
1	GB[4]	(222, 54)	OSC_C0_0/OSC_C0_0/I_RCOSC_25_50MHZ_FAB_CLKINT/U0_YNn	1


I/O to GB Connections








(none)

The 'Configure Firmware Cores' item in the Design Flow tree is right-clicked, and the context menu is open, showing the 'Open Interactively' option selected.

Click download

Reports × | StartPage × | Constraint Manager × | SD DESIGN_FIRMWARE × | SD top ×

 **Click download**

	Generate		Instance Name	Core Type	Version	Compatible Hardware Instance
1	<input checked="" type="checkbox"/>		SmartFusion2_CMSIS_0	SmartFusion2_CMSIS	2.3.105 ▾	MSS_C0
2	<input checked="" type="checkbox"/>		SmartFusion2_MSS_GPIO_Driver_0	SmartFusion2_MSS_GPIO_Driver	2.1.102 ▾	MSS_C0:GPIO
3	<input checked="" type="checkbox"/>		SmartFusion2_MSS_HPDMADriver_0	SmartFusion2_MSS_HPDMADriver	2.2.100 ▾	MSS_C0
4	<input checked="" type="checkbox"/>		SmartFusion2_MSS_MMUART_Driver_0	SmartFusion2_MSS_MMUART_Driver	2.1.100 ▾	MSS_C0:MMUART_0
5	<input checked="" type="checkbox"/>		SmartFusion2_MSS_NVM_Driver_0	SmartFusion2_MSS_NVM_Driver	2.5.100 ▾	MSS_C0
6	<input checked="" type="checkbox"/>		SmartFusion2_MSS_System_Services_Driver_0	SmartFusion2_MSS_System_Services_Driver	2.9.100 ▾	MSS_C0
7	<input checked="" type="checkbox"/>		SmartFusion2_MSS_Timer_Driver_0	SmartFusion2_MSS_Timer_Driver	2.2.100 ▾	MSS_C0

Double click Export Firmware

Libero - C:\Users\Dan\Documents\Libero\first_processor\first_processor.prjx*

Project File Edit View Design Tools SmartDesign Help

Design Flow

Top Module(root): top

Active Synthesis Implementation: synthesis

Tool

- Configure I/O States During JTAG Programming
- Configure Programming Options
- Configure Security
- Program Design
 - Generate Bitstream
 - Configure Actions and Procedures
 - Run PROGRAM Action
- Debug Design
 - Identify Debug Design
 - SmartDebug Design
- Handoff Design for Production
 - Export Bitstream
 - Export FlashPro Express Job
 - Export Job Manager Data
 - Export Pin Report
 - Export BSDL
 - Export IBIS Model
- Handoff Design for Firmware Development
 - Configure Firmware Cores
 - Export Firmware
- Handoff Design for Debugging
 - Export SmartDebug Data

Design Flow Design Hierarchy Stimulus Hierarchy Catalog Files

Generate	Instance Name	Core Type	Version	Compatible Hardware Instance
1	SmartFusion2_CMSIS_0	SmartFusion2_CMSIS	2.3.105	MSS_C0
2	SmartFusion2_MSS_GPIO_Driver_0	SmartFusion2_MSS_GPIO_Driver	2.1.102	MSS_C0:GPIO
3	SmartFusion2_MSS_HPDM_A_Driver_0	SmartFusion2_MSS_HPDM_A_Driver	2.2.100	MSS_C0
4	SmartFusion2_MSS_MMUART_Driver_0	SmartFusion2_MSS_MMUART_Driver	2.1.100	MSS_C0:MMUART_0
5	SmartFusion2_MSS_NWM_Driver_0	SmartFusion2_MSS_NWM_Driver	2.5.100	MSS_C0
6	SmartFusion2_MSS_System_Services_Drv	SmartFusion2_MSS_System_Services_Drv	2.9.100	MSS_C0
7	SmartFusion2_MSS_Timer_Driver_0	SmartFusion2_MSS_Timer_Driver	2.2.100	MSS_C0

Message

Messages Errors Warnings Info Manage suppressed messages

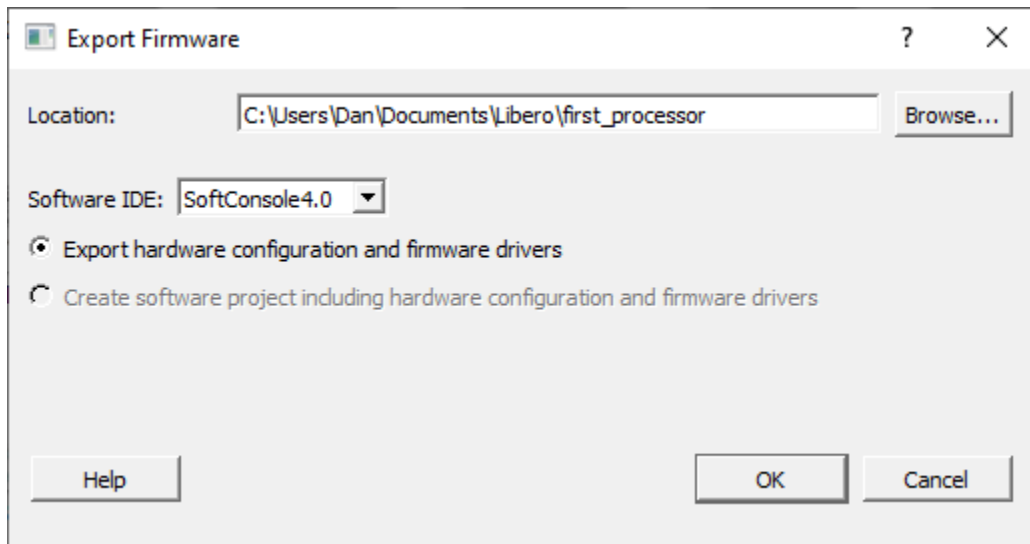
Message

Place and Route [top]

Log Message












The location should be set to your project folder by default

Select SoftConsole4.0 as the Software IDE



**** If Export Firmware fails unselect MSS_GPIO_Driver_0 and double click Export Firmware again

When firmware is successfully exported you should see that the project's folder now includes a firmware subfolder

<input type="checkbox"/> Name	Date modified	Type	Size
 component	2020-11-14 10:17 AM	File folder	
 constraint	2020-11-15 1:37 PM	File folder	
 designer	2020-11-15 1:37 PM	File folder	
 firmware	2020-11-15 2:20 PM	File folder	
 hdl	2020-11-14 8:40 AM	File folder	
 simulation	2020-11-15 1:48 PM	File folder	
 smartgen	2020-11-15 2:20 PM	File folder	
 stimulus	2020-11-14 8:40 AM	File folder	
 synthesis	2020-11-15 1:37 PM	File folder	
 tooldata	2020-11-15 2:20 PM	File folder	
 first_processor.prjx	2020-11-15 1:48 PM	PRJX File	14 KB

It is critical that the firmware subfolder appear here in this location.

If you were able to successfully export the firmware on your first attempt you can skip ahead to page 40.

For Those That Had Trouble Generating Firmware

If you had to de-select MSS_GPIO_Driver_0 in order to export firmware then follow steps below:

Download the Digital_Signal_Processing_demo sub design from the following webpage:

<https://www.microsemi.com/existing-parts/parts/150925#resources>

Extract zip folder

Explore extracted folders and navigate to:












Digital_Signal_Processing_demo\DSP_FIR_FF_LCD_12_2\firmware\drivers

copy mss_gpio folder

Explore your project folder and navigate to the first_processor\firmware\drivers\ folder

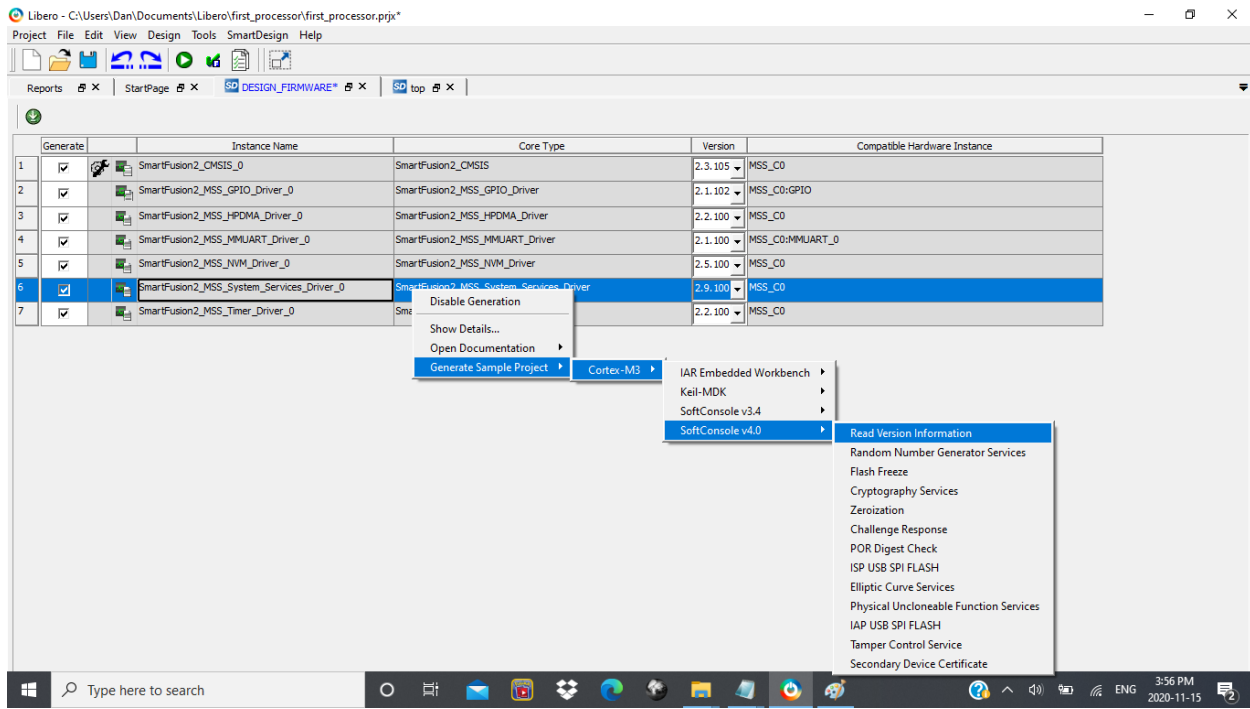
Paste mss_gpio

Your firmware\drivers\ subfolder should now look like:

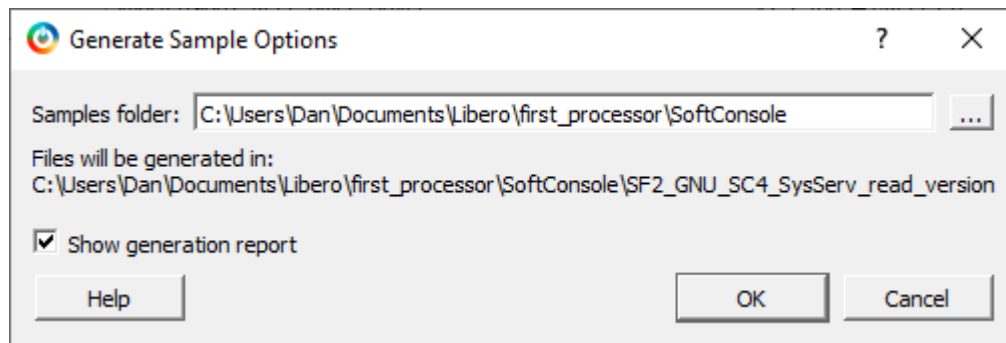
> first_processor > firmware > drivers >			
<input type="checkbox"/>	Name ^	Date modified	Type
	 mss_gpio	2020-11-15 3:52 PM	File folder
	 mss_hpdma	2020-11-15 2:20 PM	File folder
	 mss_nvm	2020-11-15 2:20 PM	File folder
	 mss_sys_services	2020-11-15 2:20 PM	File folder
	 mss_timer	2020-11-15 2:20 PM	File folder
	 mss_uart	2020-11-15 2:20 PM	File folder

In DESIGN_FIRMWARE window select SmartFusion2_MSS_System_Services_Driver_0

Right click and select Generate Sample Project → Cortex M3 → SoftConsole4.0 → Read Version Information



Accept the default location and click OK















When next popup emerges click close

Explore your project folder

Ensure that the SoftConsole folder was created (as seen below)

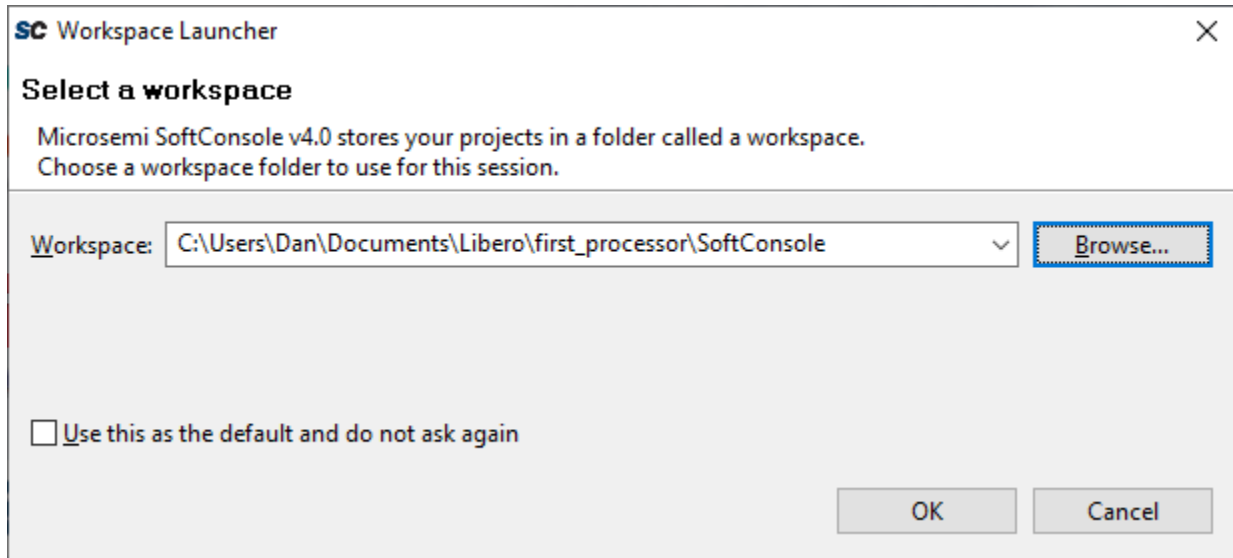
It is critical that this folder was produced in the proper location

<input type="checkbox"/> Name	Date modified	Type	Size
 component	2020-11-14 10:17 AM	File folder	
 constraint	2020-11-15 1:37 PM	File folder	
 designer	2020-11-15 1:37 PM	File folder	
 firmware	2020-11-15 2:20 PM	File folder	
 hdl	2020-11-14 8:40 AM	File folder	
 simulation	2020-11-15 1:48 PM	File folder	
 smartgen	2020-11-15 2:20 PM	File folder	
 SoftConsole	2020-11-15 4:04 PM	File folder	
 stimulus	2020-11-14 8:40 AM	File folder	
 synthesis	2020-11-15 1:37 PM	File folder	
 tooldata	2020-11-15 2:20 PM	File folder	
 first_processor.prjx	2020-11-15 1:48 PM	PRJX File	14 KB

Using SoftConsole 4.0 IDE

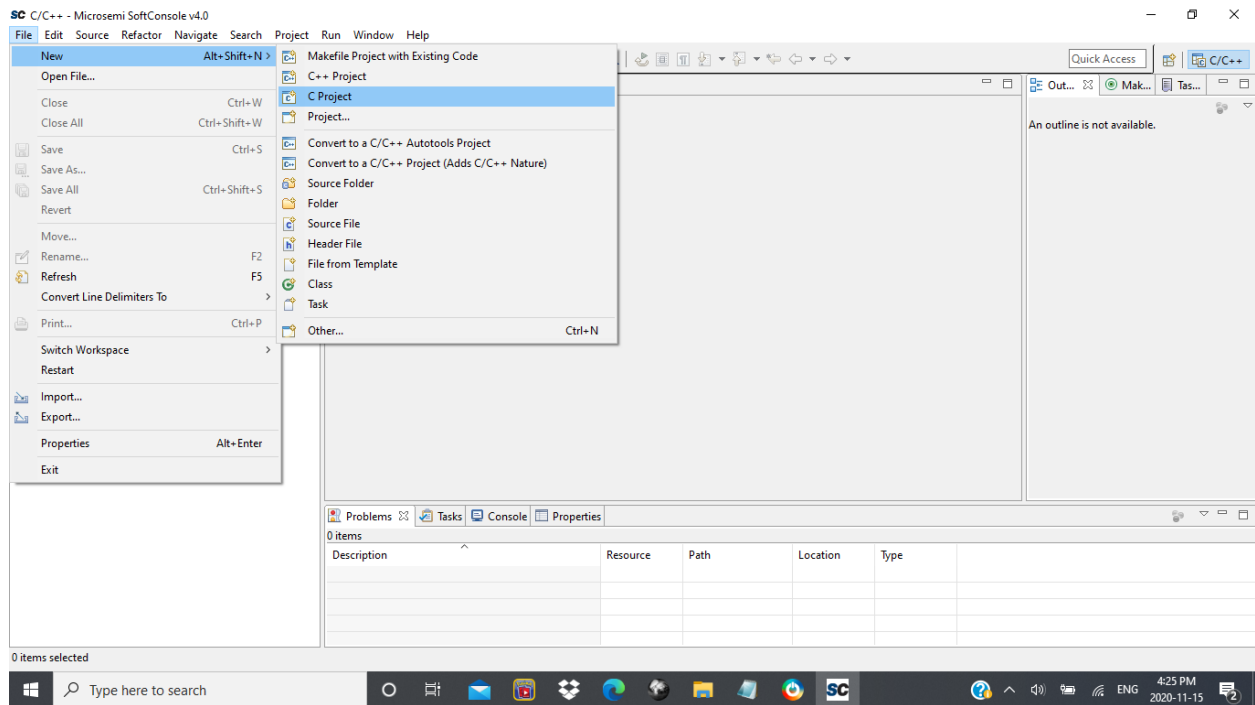
Open Microsemi SoftConsole v4.0 program

For workspace browse to your project's SoftConsole subfolder and click OK



The SoftConsole IDE will open

Click File → New → C Project



For project name type blinking_led

For Project type select Executable -- Empty Project

For tool chain select Cross ARM GCC

Click Next

SC C Project

C Project
Create C project of selected type

Project name:

☒ Use default location

Location:

Choose file system:

Project type:

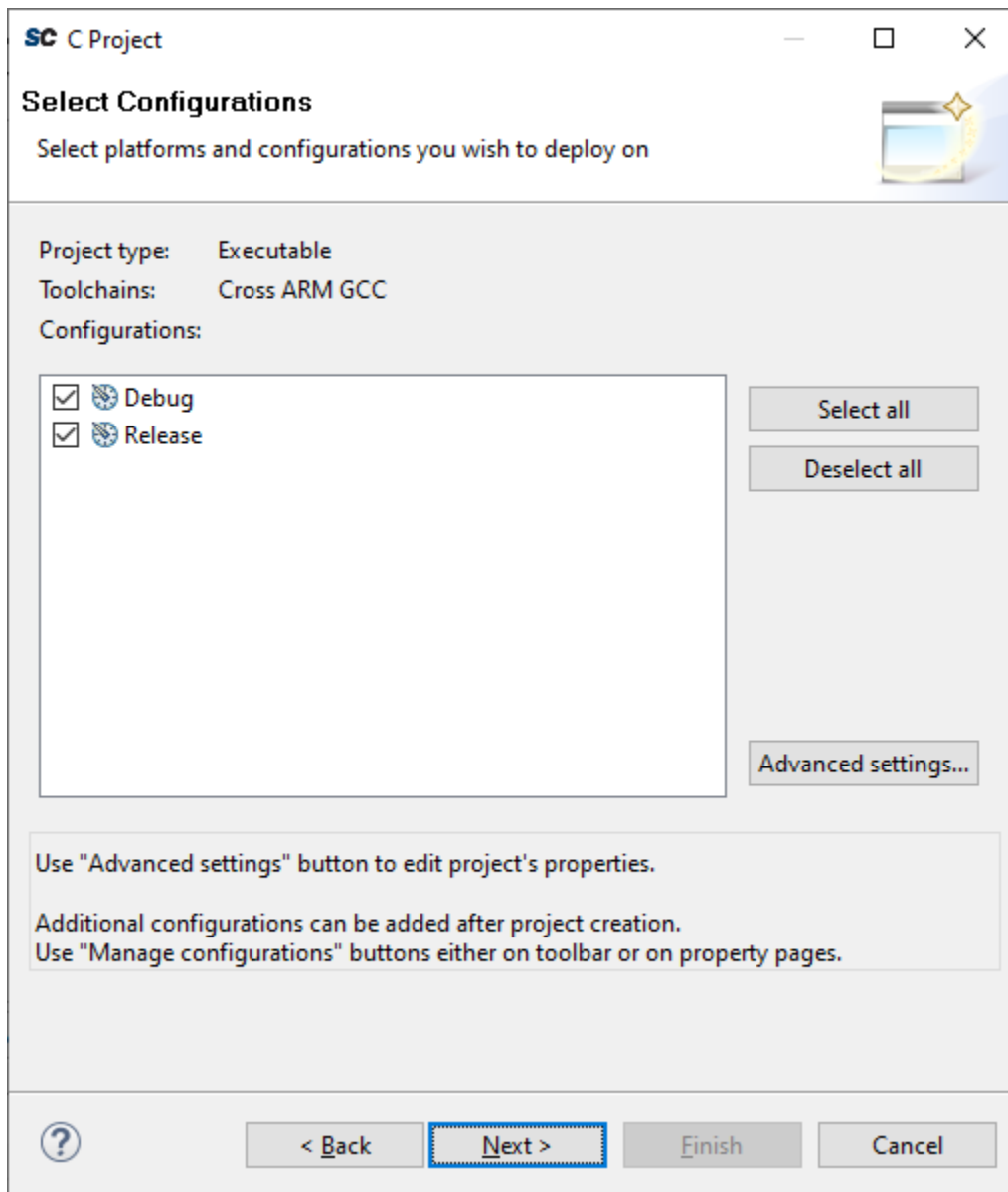
- > GNU Autotools
- ▼ Executable
 - Empty Project
 - Hello World ANSI C Project
 - Hello World ARM C Project
- > Shared Library
- > Static Library
- > Makefile project

Toolchains:

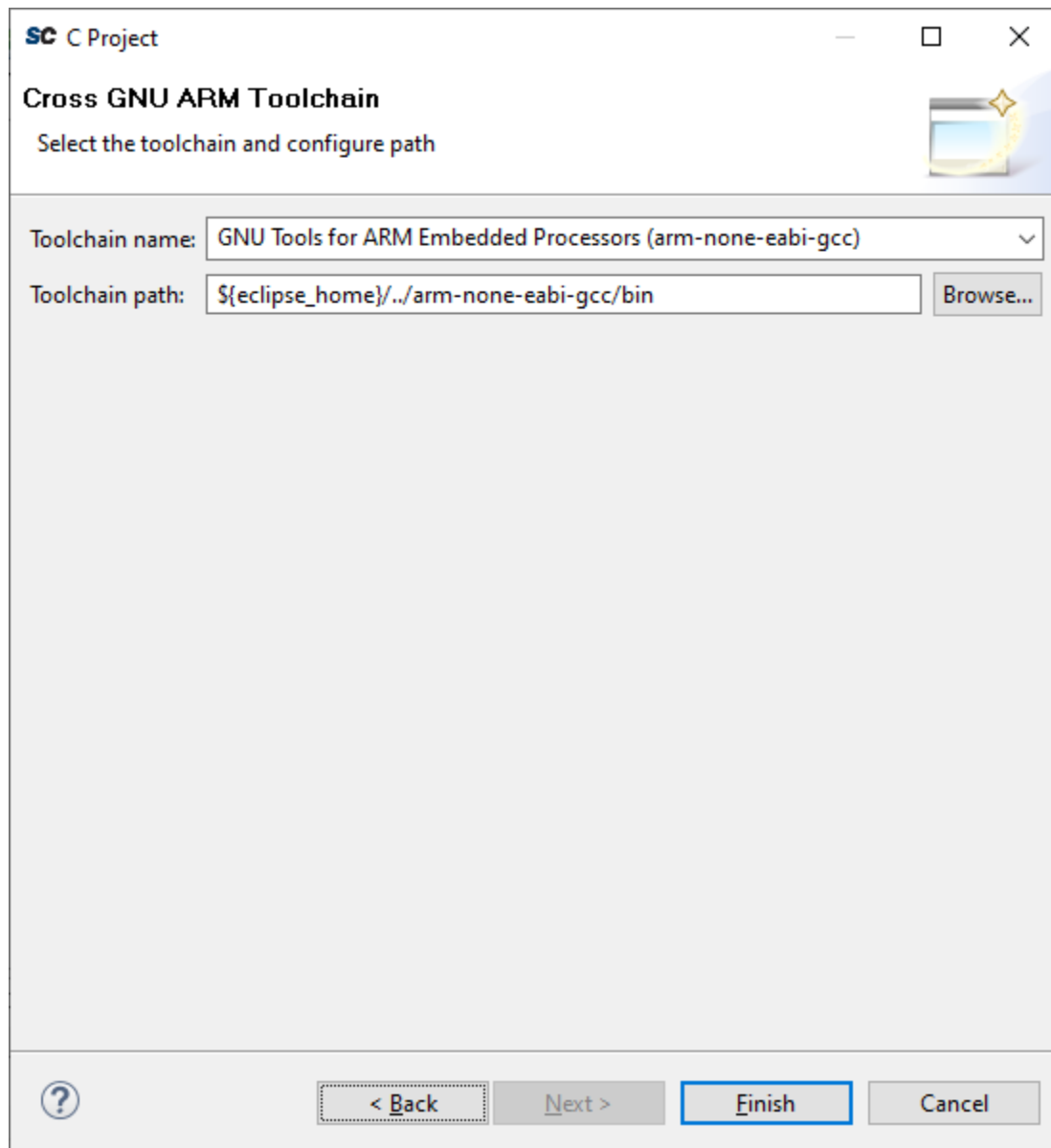
- Cross ARM GCC
- Cross GCC
- Microsoft Visual C++

☒ Show project types and toolchains only if they are supported on the platform

Ensure that Debug and Release are checked and click Next



Accept default toolchain name and path and click Finish



The screenshot shows a dialog box titled "SC C Project" with the subtitle "Cross GNU ARM Toolchain". Below the subtitle is the instruction "Select the toolchain and configure path". The dialog contains two input fields: "Toolchain name:" with a dropdown menu showing "GNU Tools for ARM Embedded Processors (arm-none-eabi-gcc)" and "Toolchain path:" with a text box containing "\${eclipse_home}/../arm-none-eabi-gcc/bin" and a "Browse..." button. At the bottom, there is a question mark icon, a "< Back" button, a "Next >" button, a "Finish" button (highlighted with a blue border), and a "Cancel" button.

SC C Project

Cross GNU ARM Toolchain

Select the toolchain and configure path

Toolchain name: GNU Tools for ARM Embedded Processors (arm-none-eabi-gcc) ▼

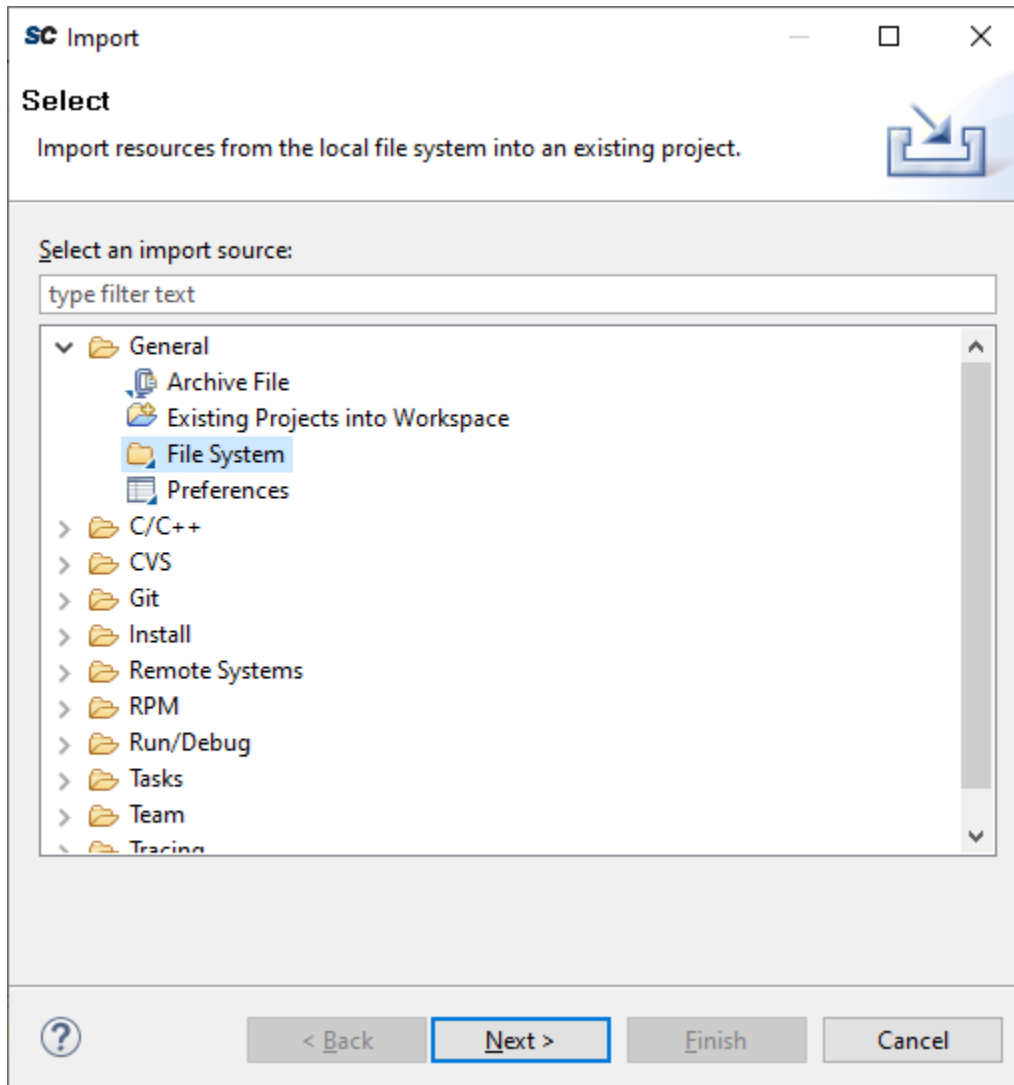
Toolchain path: \${eclipse_home}/../arm-none-eabi-gcc/bin Browse...

? < Back Next > Finish Cancel

Click File → Import

Select General → File System

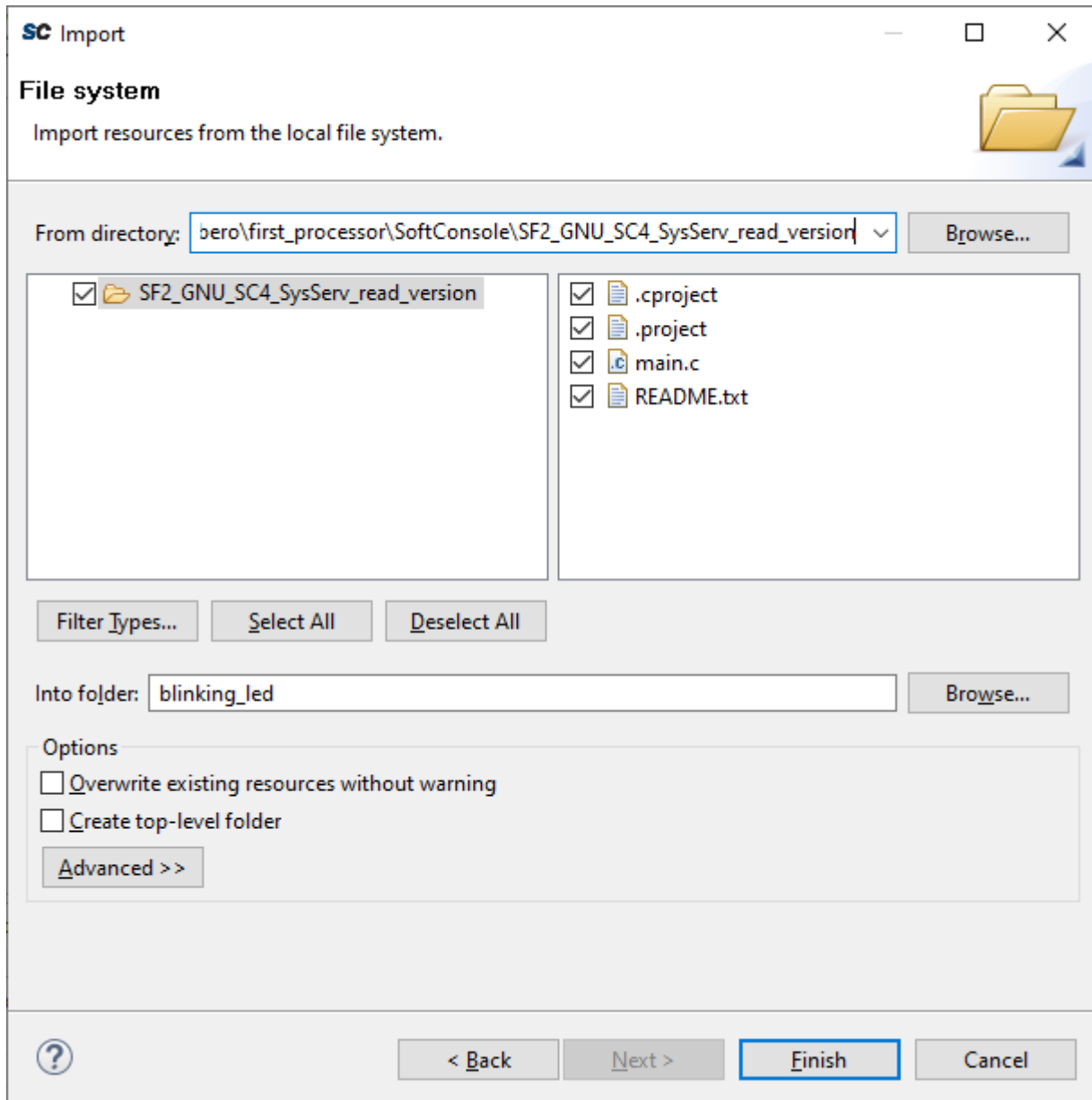
Click Next



Browse to your project's SoftConsole\SF2_GNU_SC4_SysServ_read_version\ subfolder

Apply check boxes like example below

Click Finish



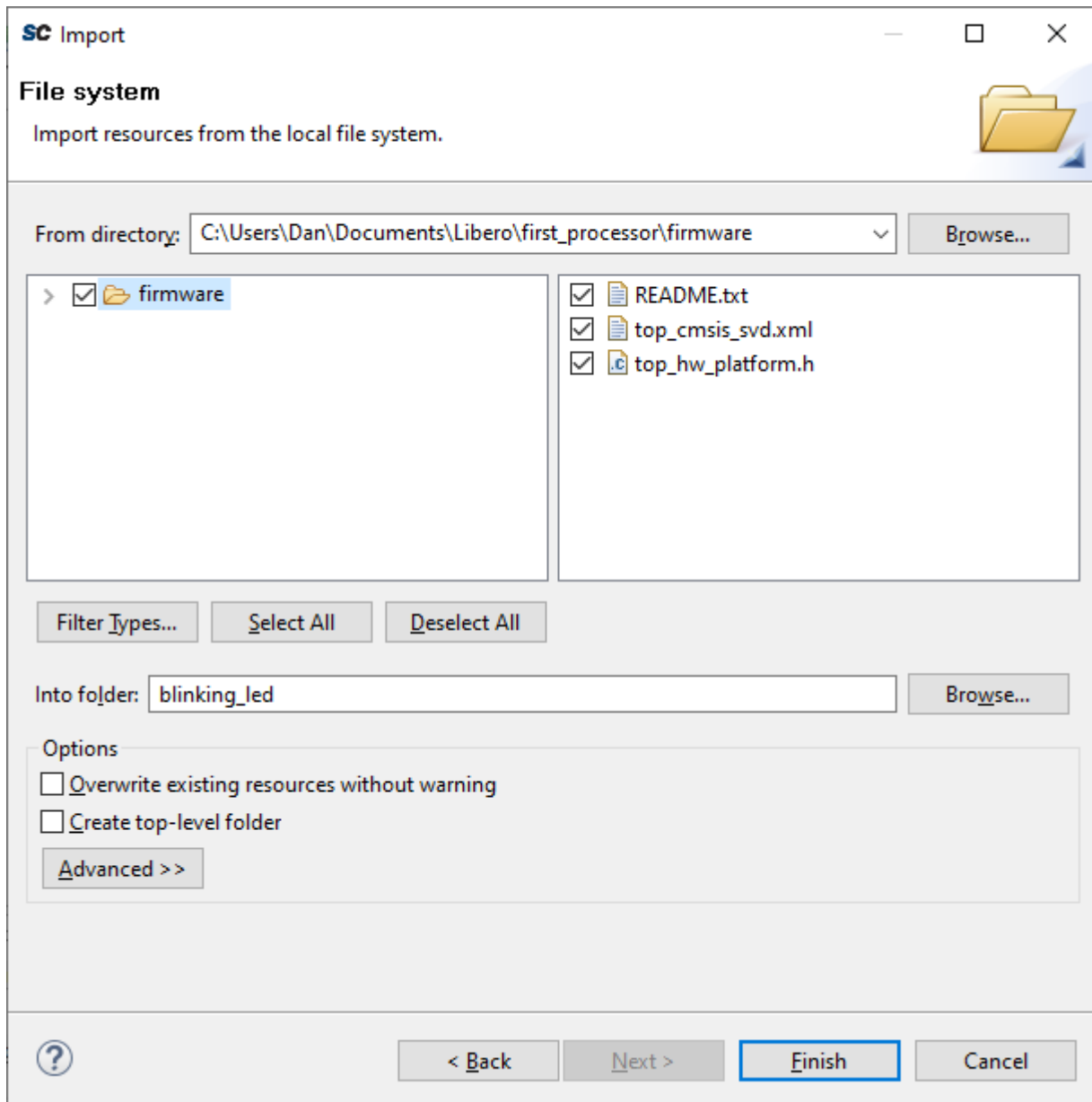
When prompted to overwrite click Yes to All

Click File → Import

Browse to your project's \firmware\ subfolder

Apply check boxes like example below

Click Finish



When prompted to overwrite click Yes to All

Double click main.c to open the file

Overwrite existing code with code on following page

```

#include <stdio.h>

#include <unistd.h>

#include "drivers/mss_sys_services/mss_sys_services.h"

#include "drivers/mss_uart/mss_uart.h"

#include "drivers/mss_gpio/mss_gpio.h"

#include "drivers/mss_timer/mss_timer.h"

//Function Declarations

void timer_delay(uint32_t load_value);

/*=====

Main function.

*/

int main(){

    uint32_t load_value= 100000000;

    //initialize GPIO and timer

    MSS_GPIO_init();

    MSS_TIM1_init(MSS_TIMER_ONE_SHOT_MODE);

    while(1){

        //set GPIO high

        MSS_GPIO_set_output(MSS_GPIO_0, MSS_GPIO_DRIVE_LOW);

        //delay

        timer_delay(load_value);

        //set GPIO low

        MSS_GPIO_set_output(MSS_GPIO_0, MSS_GPIO_DRIVE_HIGH);

        timer_delay(load_value);

        //delay

    };

    return 0;

}

void timer_delay(uint32_t load_value){

    MSS_TIM1_load_immediate(load_value);

    MSS_TIM1_start();

    while (MSS_TIM1_get_current_value()); //while counter has not reached 0

}

```

Save the main.c file by clicking file → Save

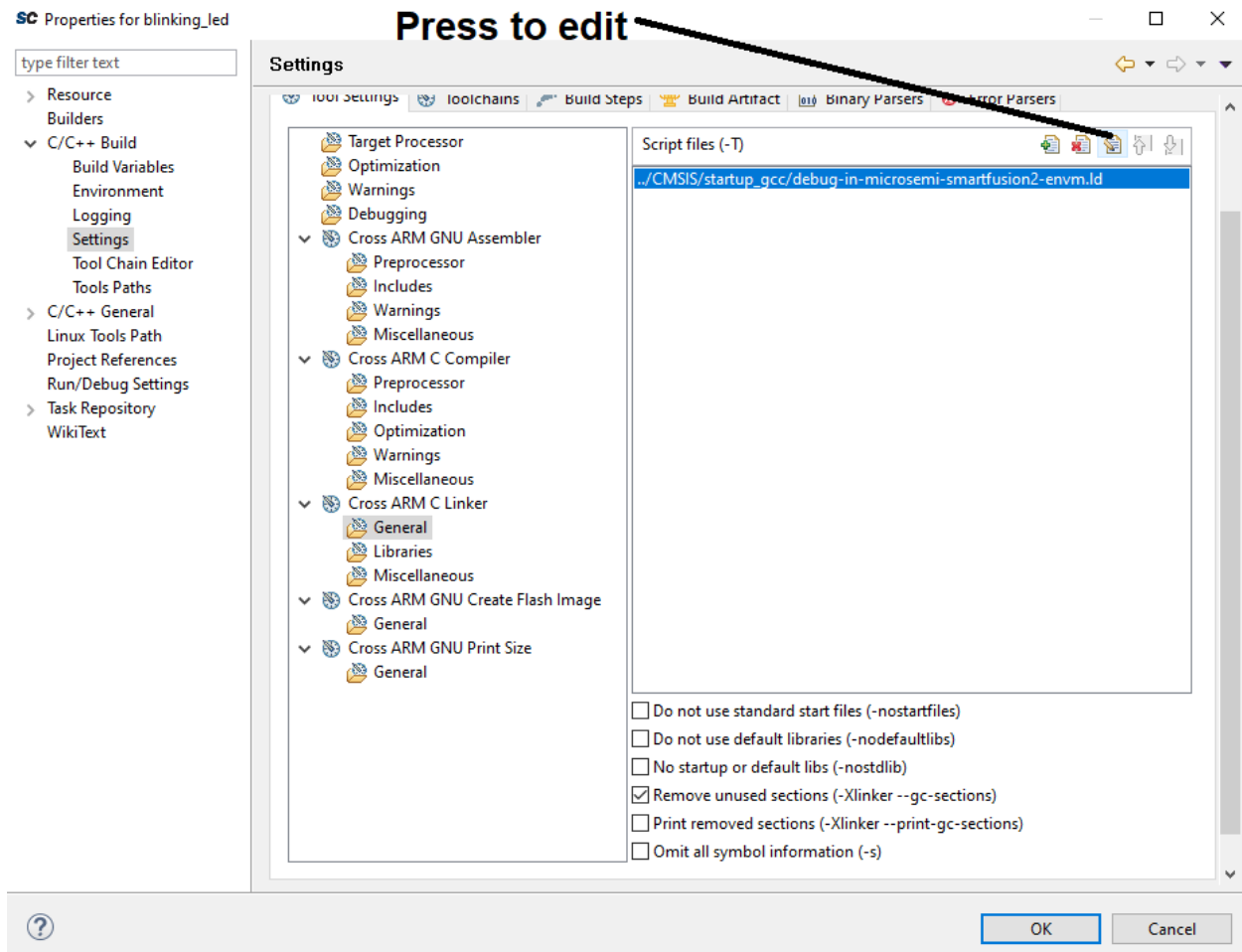
Click Project → Build Configurations → Set Active → Release

Click Project → Properties

Select C/C++ Build → Settings

In Tool Settings tab select Cross ARM C Linker → General

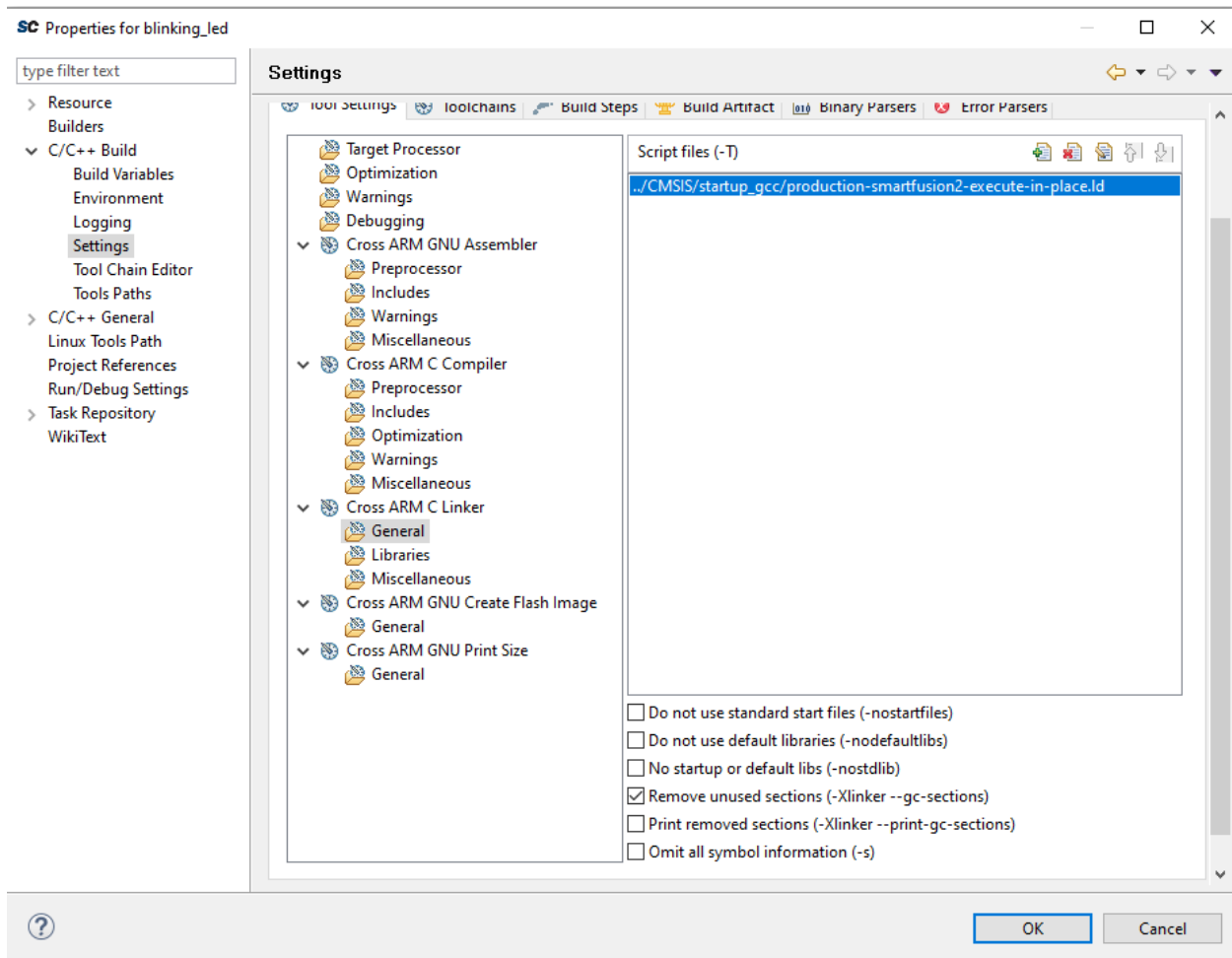
Click edit



Change the existing text to:

`../CMSIS/startup_gcc/production-smartfusion2-execute-in-place.ld`

Click OK

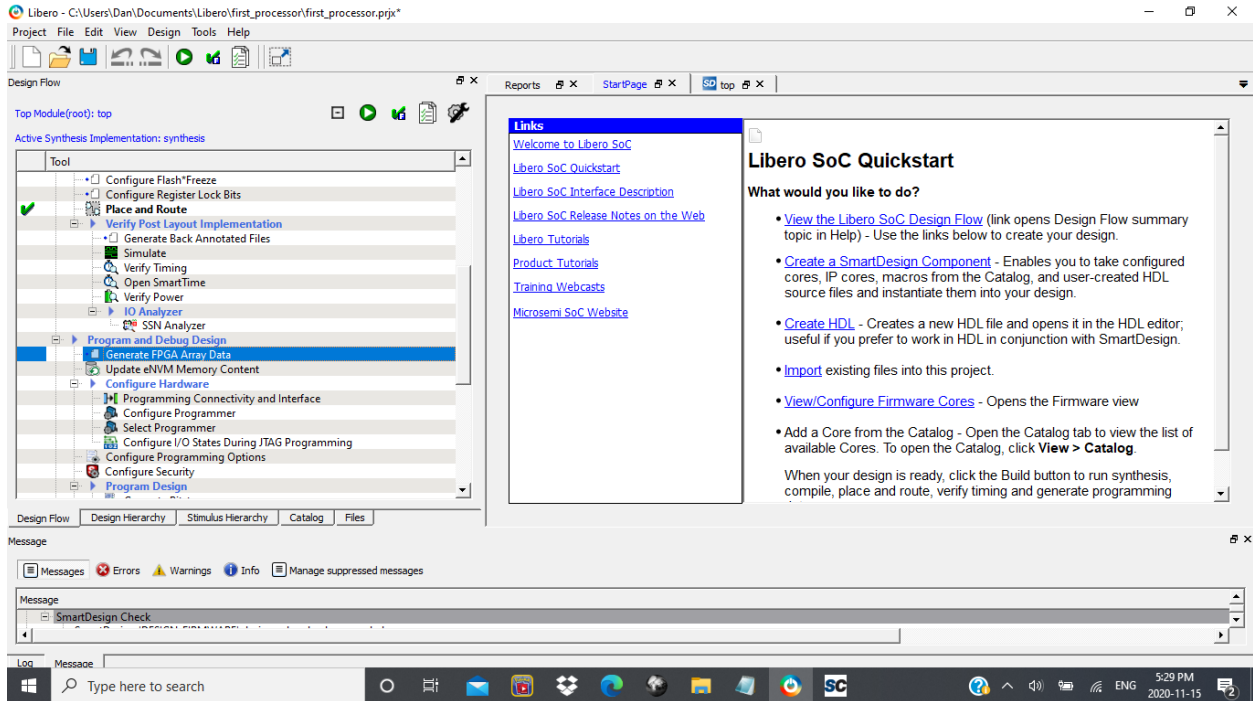


Click Project--> Build All

Go back to Libero SoC

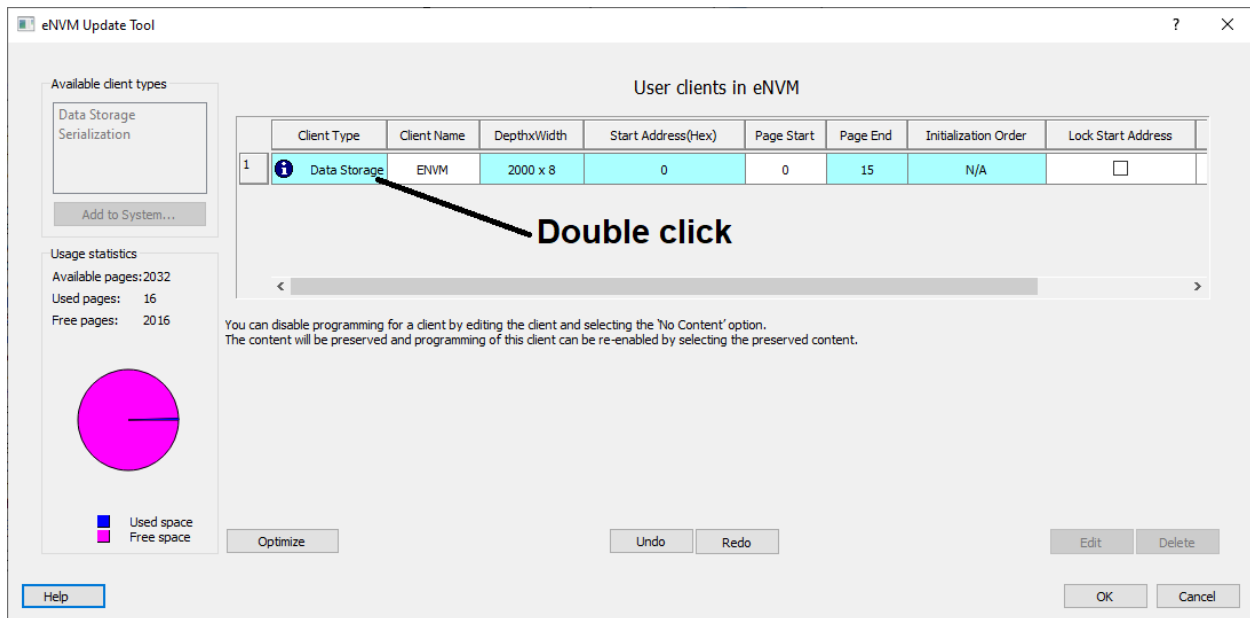
Double click Generate FPGA Array Data

If an information box pops up click OK



Double Click Udate eNVM Memory Content

Double Click Data Storage

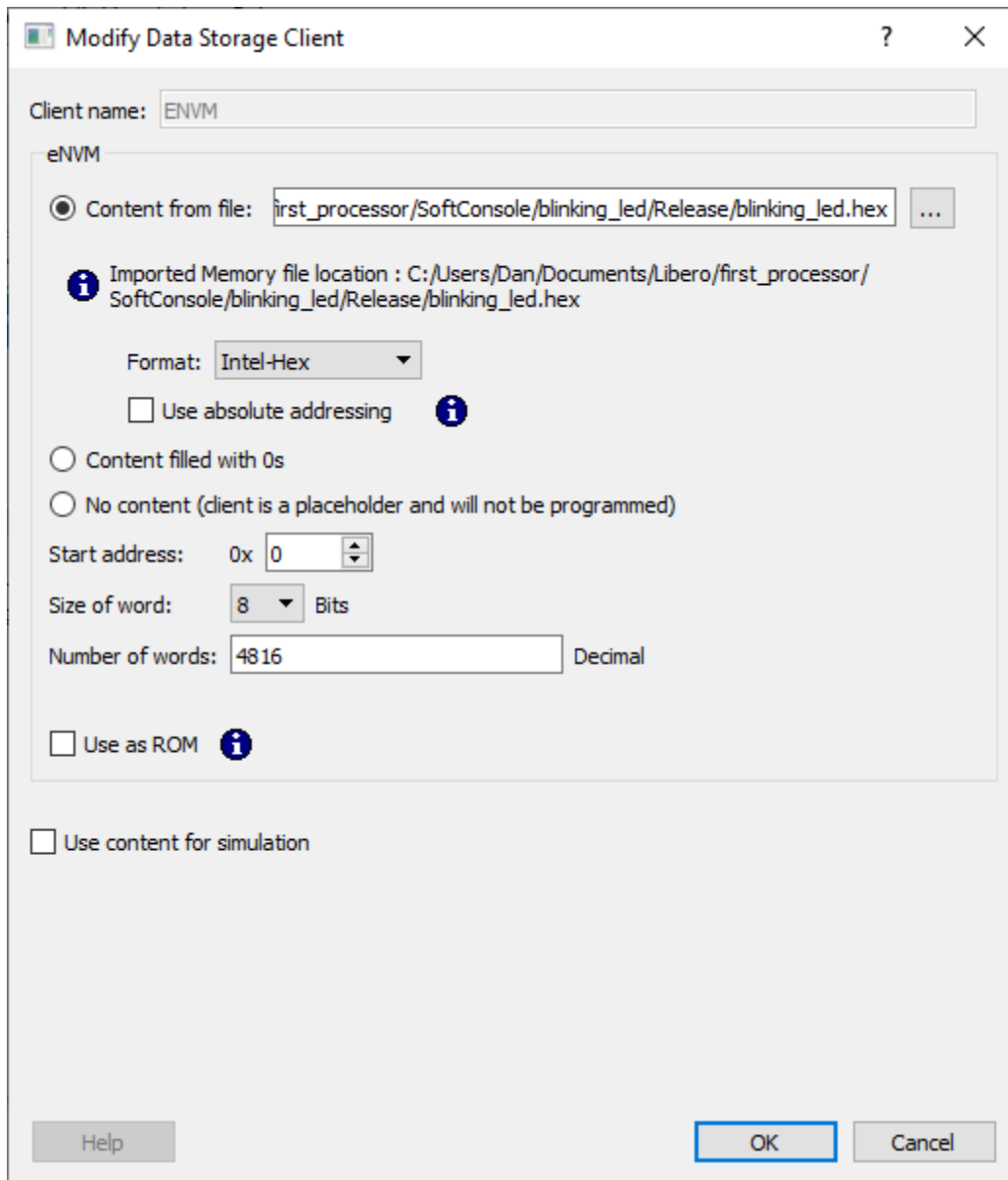


Select Content from file option

Choose the blinking_led.hex file from your project's \SoftConsole\blinking_led\Release\ subfolder

Refer to following image for other settings

Press OK



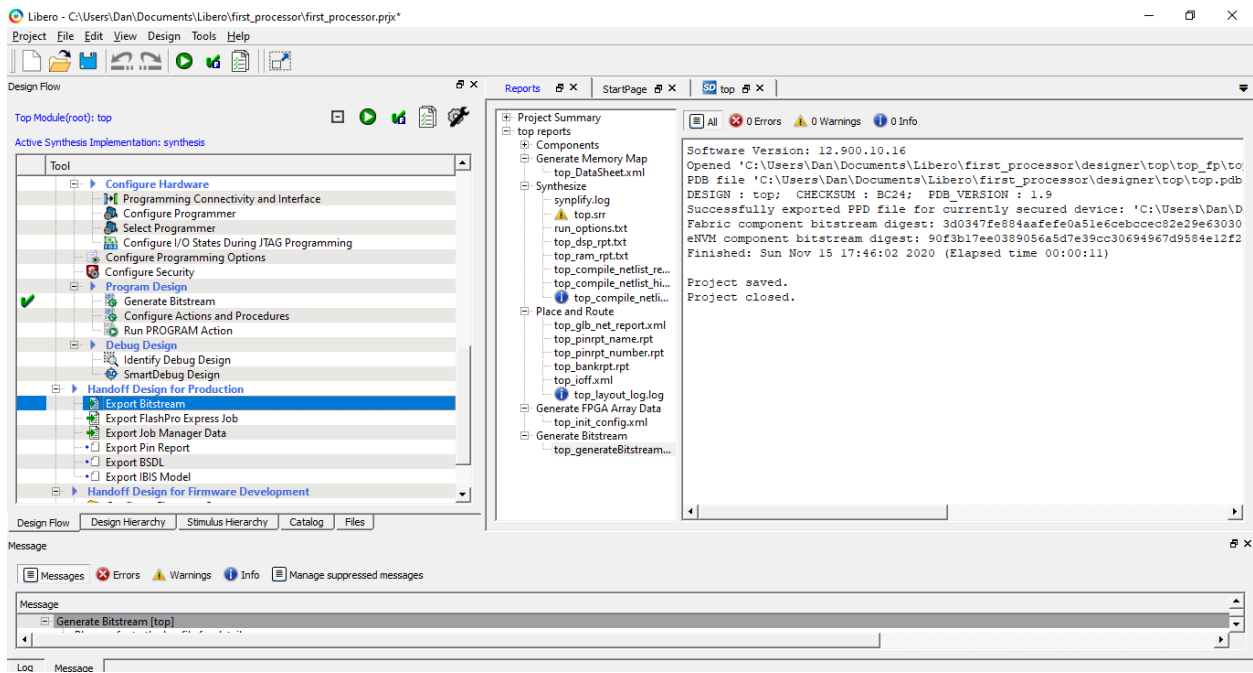
Double click Generate Bitstream

The screenshot shows the Microsemi Libero IDE interface. The 'Design Flow' pane on the left lists various steps, with 'Generate Bitstream' highlighted. The 'Reports' pane on the right displays the 'Configuration Report for MSS, SERDES(s), Fabric DDR and Fabric CCC(s)'. The report includes the Microsemi Corporation logo, version information (v12.5), and a date (Sun Nov 15 17:31:28 2020). Below the title, it states 'CCC-NE0 (Unused)' and provides a table of register values.

Register	Field	INIT	Value
FCCC_RFDIV_CR	RFDIV[7:0]	INIT[7:0]	8'h0
FCCC_FBDIV_CR0	FBDIV[7:0]	INIT[15:8]	8'h0
FCCC_FBDIV_CR1	FBDIV[13:8]	INIT[21:16]	6'h0
FCCC_GPD0_CR	GPDIV[7:0]	INIT[29:22]	8'h0
FCCC_GPD1_CR	GPDIV[7:0]	INIT[37:30]	8'h0
FCCC_GPD2_CR	GPDIV[7:0]	INIT[45:38]	8'h0

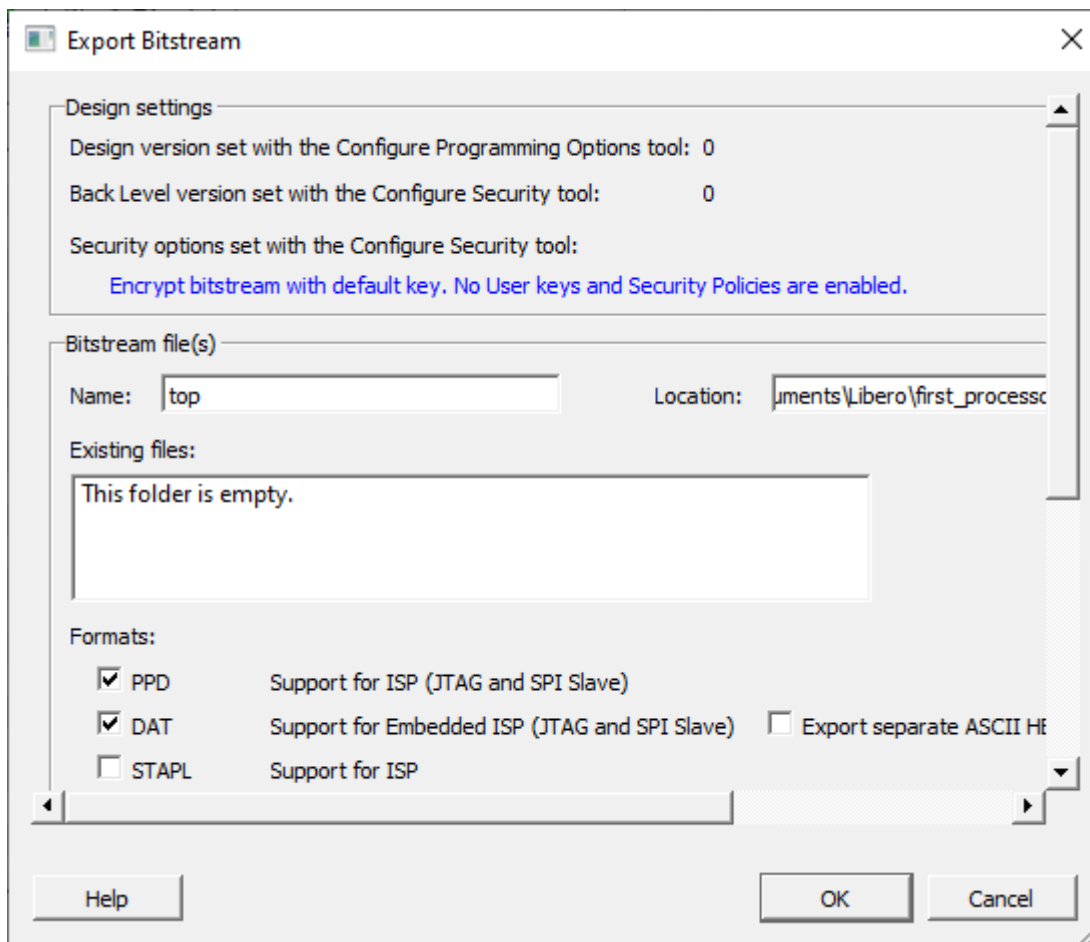
The bottom of the screenshot shows the Windows taskbar with the time 5:44 PM and date 2020-11-15.

Double click Export Bitstream



Ensure that PPD and DAT formats are selected

Click OK



The generated .DAT file can be found in your project's \designer\top\export\ subfolder

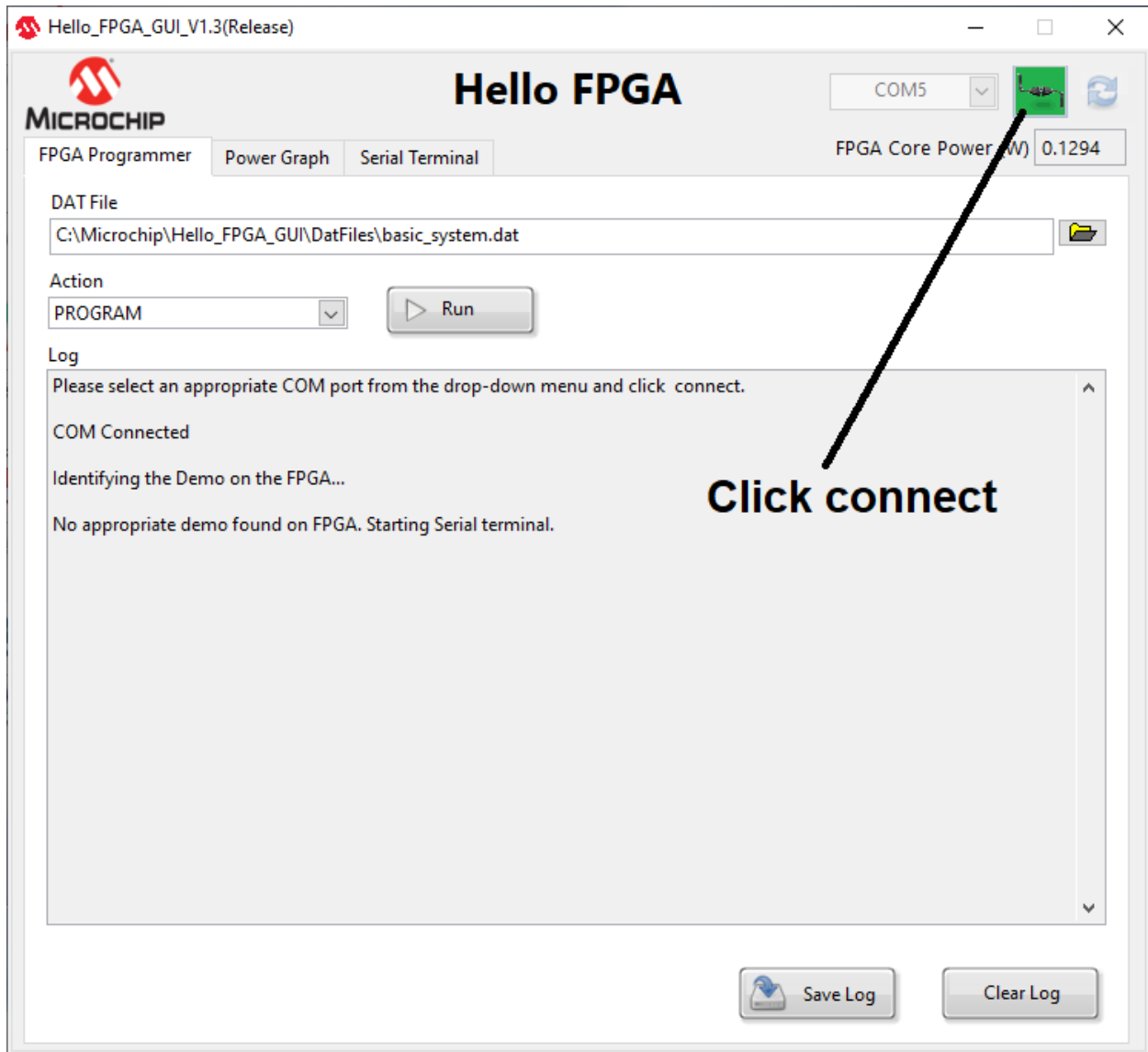
> first_processor > designer > top > export				
<input type="checkbox"/>	Name	Date modified	Type	Size
<input checked="" type="checkbox"/>	top.dat	2020-11-15 5:53 PM	DAT File	563 KB
<input type="checkbox"/>	top.ppd	2020-11-15 5:52 PM	PPD File	720 KB
<input type="checkbox"/>	top_dat.digest	2020-11-15 5:53 PM	DIGEST File	1 KB
<input type="checkbox"/>	top_ppd.digest	2020-11-15 5:52 PM	DIGEST File	1 KB

(Assuming that you installed the Hello FPGA GUI in its default location) I recommend that you copy top.DAT to C:\Microchip\Hello_FPGA_GUI\DatFiles

Connect the Hello FPGA Development board to your computer using the USB cable included with the kit

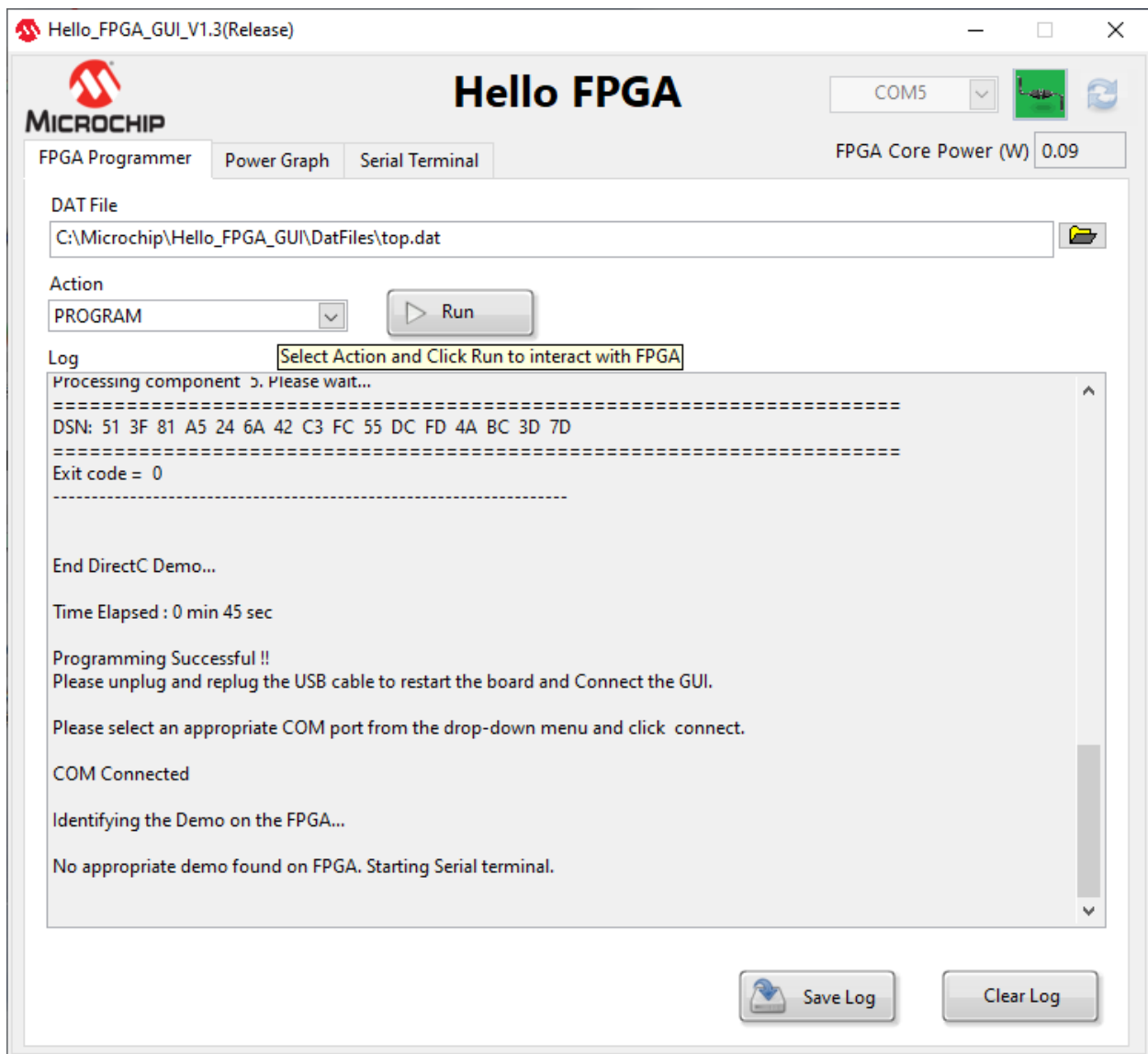
Open the Hello FPGA GUI Software

Click the connect graphic



Choose the top.DAT file

Click Run



Unplug and then reconnect the USB cable

LED1 should now be flashing (toggling between on and off every second)

Congratulations on making your first design targeting the Hello FPGA Design Kit

