NetFusion-EXP Libero Starter Project Helper

User Guide

Table of Contents

List of Figures

NetFusion-EXP Libero Starter Project Helper

V1.0 – April 2015

About This Document

This specification introduces the NetFusion-EXP baseboard's Libero IDE starter project. Whilst the project ZIP binary is provided as a downloadable target to the NetFusion-EXP PCB, using the open-source IP project as a start point will aid and help the user's intended functional product, and allow for a start-point in their design customization process.

Intended Audience

This document is fully intended to be viewed and reference by Nine Ways customers using the technology for larger designs and projects.

1 Introduction

Microsemi design and manufactured the SmartFusion2 FPGA. *Libero* is the IDE software for programming and synthesizing IP cores. Along with the bundle of uClinux firmware compilation tools, this is the heart of the NetFusion-EXP design process for Users.

1.1 Libero SoC/IDE Version 11.x Derivatives

Microsemi's Libero ® IDE software release for designing with Microsemi Rad-Tolerant FPGAs, Antifuse FPGAs and Legacy & Discontinued Flash FPGAs and managing the entire design flow from design entry, synthesis and simulation, through place-and-route, timing and power analysis. PCN 1108: Silicon Family Support in Libero IDE.

Libero IDE Software Features:

- Powerful project and design flow management
- Full suite of integrated design entry tools and methodologies:
  - SmartDesign graphical SoC design creation with automatic abstraction to HDL
  - IP Core Catalog and configuration
  - User-defined block creation flow for design re-use
- Synplify Pro ME synthesis fully optimizes Microsemi FPGA device performance and area utilization
- Synphony Model Compiler ME performs high-level synthesis optimizations within a Simulink® environment

- Modelsim ME VHDL or Verilog behavioral, post-synthesis and post-layout simulation capability
- Physical design implementation, floor planning, physical constraints, and layout
- Timing-driven and power-driven place-and-route
- SmartTime environment for timing constraint management and analysis
- SmartPower provides comprehensive power analysis for actual and "what if" power scenarios
- Interface to FlashPro programmers
- Post-route On Chip Debug Tools and Identify ME debugging software for Microsemi flash designs
- Silicon Explorer II debugging software for Microsemi antifuse designs

1.2 Installation

Libero software is downloadable for free from http://www.microsemi.com/products/fpga-soc/designresources/design-software/libero-soc#downloads. Some Libero features are optional during installation.

You can minimize the disk space required by only installing tools you use.

You must have a license to run Libero; the license type that you obtain determines what devices you can use and what IP is included. The following license types exist for Libero:

- Libero Platinum (*All devices and RTL IP Bundle*)
- Libero Gold (*Limited devices and Obfuscated IP Bundle*)

View the complete descriptions of the above Libero installations at http://www.microsemi.com/products/fpga-soc/design-resources/design-software/libero soc#licensing.

View the IP Bundle contents at http://www.microsemi.com/products/fpga-soc/design-resources/ip-cores.

Libero installation is covered in "Installing Libero Software" on page 16.

Note: You must have Admin rights on the installation machine to install Libero SoC.

1.3 Starting a Project and Basic Understanding

Before attempting to modify or implement any project in Libero, it is advised that you download and read the following PDF references:

System on Chip installation:

Libero SoC v11.X User's Guide

Libero SoC Quick Start Guide for Software v10.0

Integrated Development Environment installation:

Libero IDE and Software Installation and Licensing Guide

Libero IDE License Troubleshooting Guide

(**Note:** *Press CNTL and click to download the links*)

1.3.1 Libero 11.X IDE First Launch

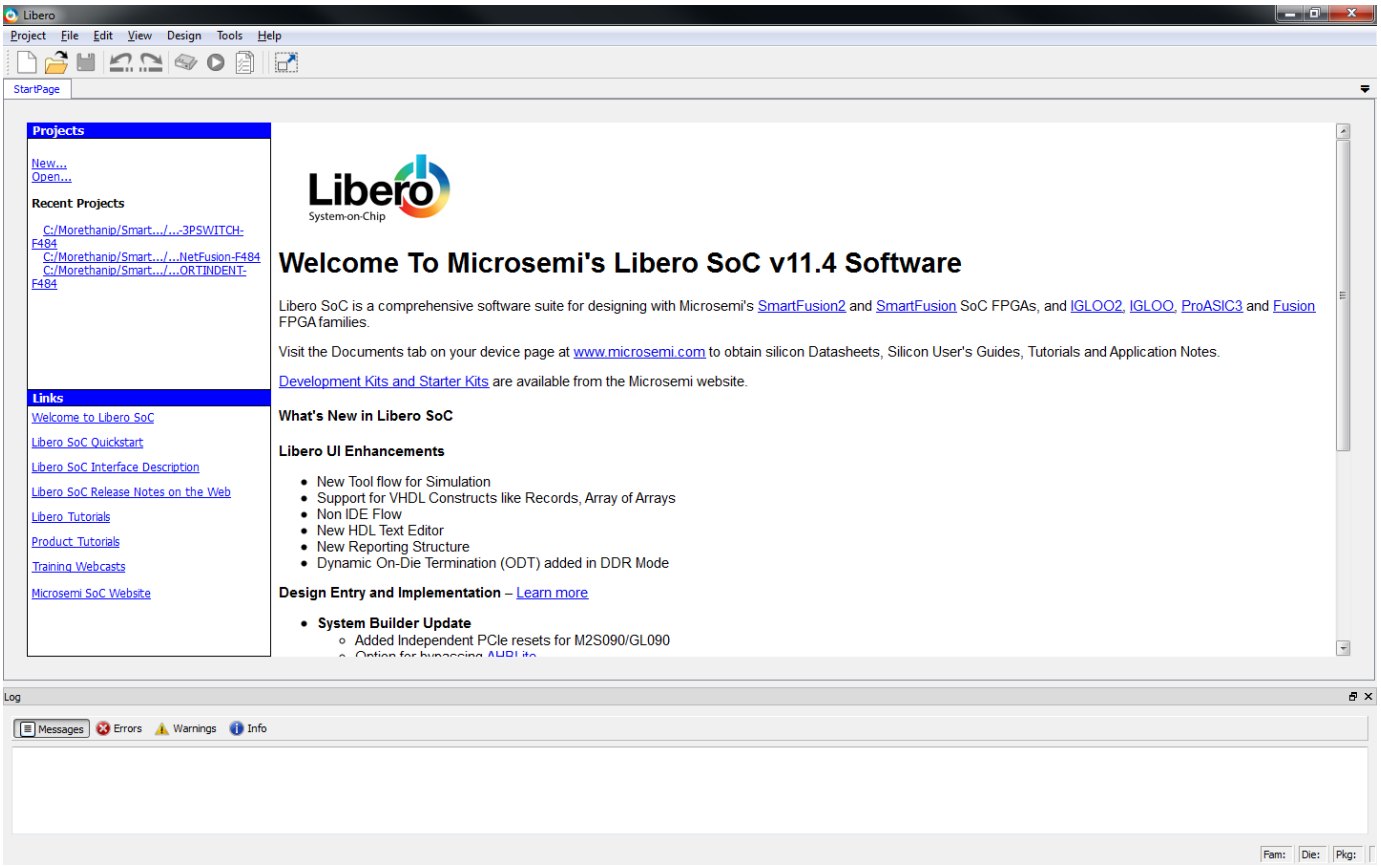


Figure 1 - Libero 11.X IDE Boot-Up Screen

1.4 NetFusion-EXP

When purchasing any NetFusion-EXP variant, an important component of the overall product is the NetFusion-EXP Libero starter project that is provided by Nine Ways Research & Development Ltd. This is downloadable from Nine Ways R&D Ltd and when expanded into a target directory on your development PC, provides an immediate project for your needs. Instead of having to start and debug creating an ARM sub-system with all of the supporting IP cores required to have a functioning NetFusion-EXP PCB – just use the provided project.

From installation, you can program the IDE project once synthesized into your NetFusion-EXP product using the FlashPro4 USB device. This will provide the standard functionality in the Smartfusion2 FPGA fabric on the M2S-SOM-F484 **SOM** (**S**ystem-**O**n-**M**odule) to see a working project. In the software bundle to this product, the uClinux device drivers will drive and control the hardware on the NetFusion-EXP PCB through the IP cores in this project for the FPGA fabric.

Users can contact Nine Ways R&D for special functionality to be developed and deployed, but this serves as an addition to this starter Libero 11.X IDE project. Moreover, in conjunction with MorethanIP GmbH, customized and locked down projects can be tailored for customer requests but that also is separate to this project.

1.5 Product Development

At the point where the Libero IDE/SoC has been installed, the NetFusion-EXP starter project has been downloaded and exploded into a target directory on your PC, the project has been loaded, synthesized, programmed and shown to be running on the NetFusion-EXP PCB product – you are ready to begin your development.

As standard, the main fast Ethernet pathways into the SmartFusion2 FPGA fabric are wired in through the FPGA ball I/O, assigned in the I/O editor, brought down through the Top-Level and then into the SOM level of the design in the project. They then terminate at a dummy IP core for all un-assigned wires – this makes life a lot easier for the developer knowing that all the NetFusion-EXP traces coming into the M2S-SOM-F484 are wired into the SOM level of the fabric design. Changes are therefore quick and easy to then re-assign in that lower level sheet to new instantiated IP cores of the user's choice.

The category of wires left terminated and not used are the GMII Ethernet pathways. Users can download and use Vendor specific MAC/SWITCH cores or chose to privately purchase cores from reputable design-houses such as MorethanIP. All other used hardware on the NetFusion-EXP PCB is wired and connected in the SOM sheet layer to *GPIO, SPI, UART, USB, I2C* etc as standard in the starter project.

If you decide to write and author your own IP cores in Verilog or VHDL, you can drop and place the code into the **\[Project Dir]\hdl** directory.

Note: *Once you have started to customize and tailor the project to your own needs and functionality, obviously renaming the project is easy – just rename the project directory and inside that directory, just rename the **[.prj]** file. Close and re-open the Libero IDE.*

1.6 Documentation Checklist

Libero SoC User's Guide
Explains how to use the Libero SoC Project Manager, including Designer and SmartDesign.

SmartFusion2 and IGLOO2 SmartTime, I/O Editor and ChipPlanner User's Guide
Provides details about using SmartTime for timing analysis, placing macros, floor planning and viewing chip resources for SmartFusion2

MultiView Navigator User's Guide (includes documentation for ChipPlanner, PinEditor, I/O Attribute Editor, and NetlistViewer in MVN)
Provides details about placing macros, floor planning, and viewing chip resources; contains information about using NetlistViewer in the MultiView Navigator to view your netlist; describes how to use the PinEditor in MVN; describes how to use the I/O Attribute Editor tool.

Design Constraints User's Guide
Provides a complete reference for creating and modifying timing, physical, and netlist optimization constraints in Libero SoC, including families and file formats supported for each constraint. It also describes how to create and modify I/O constraints with the I/O Attribute Editor, before compiling your design.

SmartPower User's Guide
Describes how to use SmartPower for power analysis.

SmartTime User's Guide
Describes how to use SmartTime for timing analysis and how to set clock constraints.

Tcl Command Reference
Lists all the Tcl commands and parameters for the Microsemi software tools.

Analog System Builder, FlashROM and Flash Memory System Builder User's Guide
Describes how to use the FlashROM generator, the Analog System Builder, and the Flash Memory System Builder.

SmartGen Cores Reference Guide
Provides descriptions of cores that can be generated from the Catalog using the SmartGen Core Builder.

FlashPro User's Guide
Contains information about how to program your devices using the FlashPro software and device programmer. FlashPro is not available on UNIX.

SmartFusion2 and IGLOO2 Macro Library Guide
Provides descriptions of Microsemi library elements for the Microsemi SmartFusion2 and IGLOO2 device families. Symbols, truth tables, and module counts (if appropriate) are included for all macros.

IGLOO, ProASIC3, SmartFusion and Fusion Macro Library Guide
Provides descriptions of Microsemi library elements for Microsemi SmartFusion, Fusion, ProASIC3 and ProASIC3E device families. Symbols, truth tables, and module counts (if appropriate) are included for all macros.

SmartFusion2 and IGLOO2 Block Flow User's Guide
Describes how to create and integrate Blocks in Libero SoC for SmartFusion2 and IGLOO2.

VHDL Vital Simulation Guide
Contains information about using the ModelSim to simulate designs for Microsemi SoC devices.

Verilog Simulation Guide
Contains information about interfacing the FPGA development software with Verilog simulation tools.

ViewDraw User's Guide
Describes how to create designs in ViewDraw using menu commands, toolbar buttons, and by selecting and entering information on dialog boxes. ViewDraw for Microsemi is not available on UNIX.

ModelSim ME Book Case
Contains a User's Manual, Command Reference, and Tutorial. These guides contain details about using ModelSim ME, Libero's integrated simulation tool. Refer to the documentation included with ModelSim ME for more information. ModelSim ME documentation is also available at:

http://www.microsemi.com/products/fpga-soc/design-resources/design-software/liberosoc#Documents

Synopsys Synplify Pro ME
Documents include release notes, user's guide, tutorial, reference manual, and license configuration and set-up. Refer to the documentation included with Synopsys Synplify Pro for more information. Synopsys Synplify Pro ME documentation is also available at:

http://www.microsemi.com/products/fpga-soc/design-resources/design-software/libero-soc#documents

2 Downloading New IP Core(s) into the Libero IDE Installation

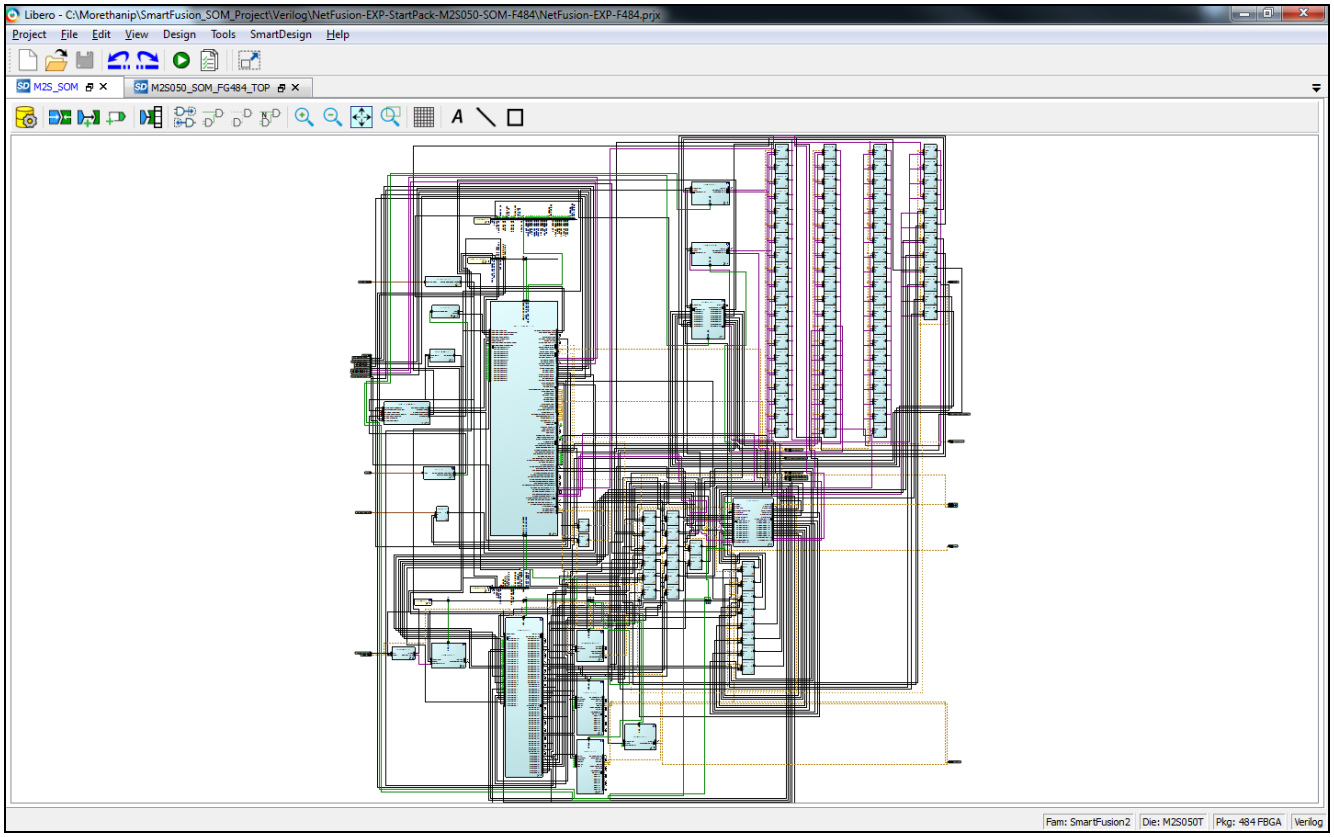


Figure 2 - Libero Updating IP Core in the Vault

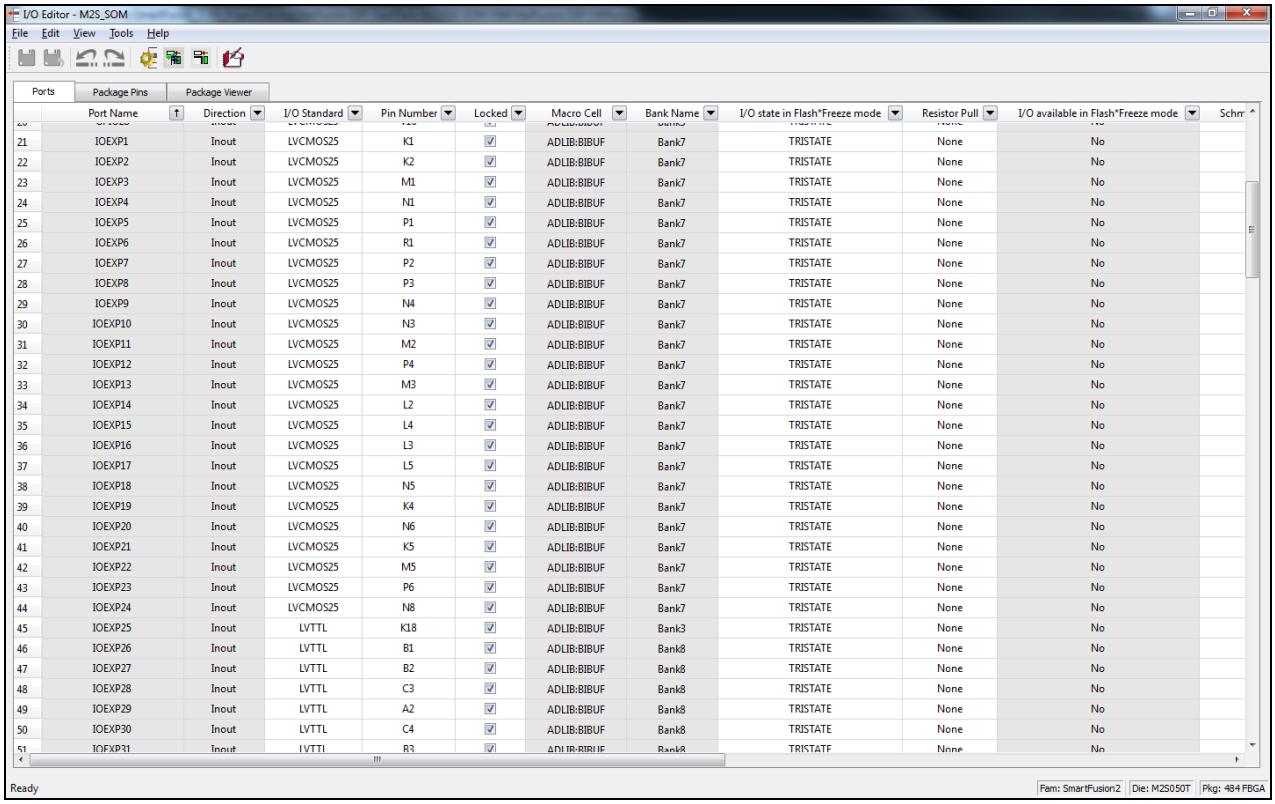
When you first load the NetFusion-EXP starter project into the IDE, the most likely occurrence to happen is a system message warning you to update “New IP Cores”. This is because the design includes cores that you possibly do not have on your vault on your local hard drive.

Click “*YES*” to proceed and let the download process complete. **Note:** *this may take several minutes and you MUST have a network connection and gateway to the internet.*

3 NetFusion-EXP Libero I/O Assignments

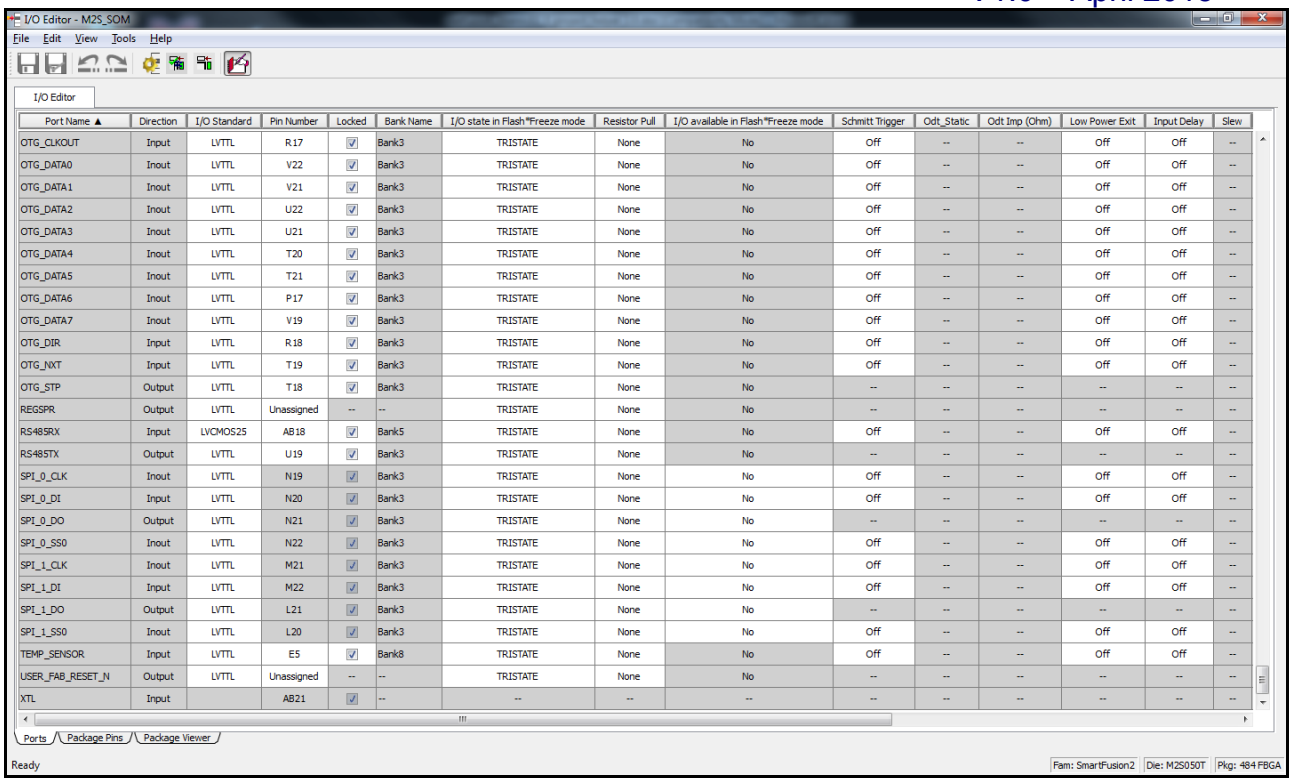
The NetFusion-EXP Libero starter project is delivered already with the I/O balls of the FPGA assigned ready for the user. These all correspond with the pin-out tracking assignments on the NetFusion-EXP PCB product. This means that you will not have to consider any of these assignments unless you are planning any major changes to the inherent default design.

This is very unlikely as you would need then to request changes to the PCB design. However, this section highlights the assignments in case you also want to make minor changes to the direction of the port, default output values and/or change internal SmartFusion2 FPGA pull-up values.



	Port Name	Direction	I/O Standard	Pin Number	Locked	Macro Cell	Bank Name	I/O state in Flash*Freeze mode	Resistor Pull	I/O available in Flash*Freeze mode
21	IOEXP1	Inout	LVCMOS25	K1	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
22	IOEXP2	Inout	LVCMOS25	K2	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
23	IOEXP3	Inout	LVCMOS25	M1	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
24	IOEXP4	Inout	LVCMOS25	N1	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
25	IOEXP5	Inout	LVCMOS25	P1	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
26	IOEXP6	Inout	LVCMOS25	R1	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
27	IOEXP7	Inout	LVCMOS25	P2	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
28	IOEXP8	Inout	LVCMOS25	P3	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
29	IOEXP9	Inout	LVCMOS25	N4	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
30	IOEXP10	Inout	LVCMOS25	N3	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
31	IOEXP11	Inout	LVCMOS25	M2	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
32	IOEXP12	Inout	LVCMOS25	P4	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
33	IOEXP13	Inout	LVCMOS25	M3	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
34	IOEXP14	Inout	LVCMOS25	L2	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
35	IOEXP15	Inout	LVCMOS25	L4	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
36	IOEXP16	Inout	LVCMOS25	L3	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
37	IOEXP17	Inout	LVCMOS25	L5	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
38	IOEXP18	Inout	LVCMOS25	N5	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
39	IOEXP19	Inout	LVCMOS25	K4	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
40	IOEXP20	Inout	LVCMOS25	N6	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
41	IOEXP21	Inout	LVCMOS25	K5	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
42	IOEXP22	Inout	LVCMOS25	M5	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
43	IOEXP23	Inout	LVCMOS25	P6	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
44	IOEXP24	Inout	LVCMOS25	N8	☑	ADLIB:BIBUF	Bank7	TRISTATE	None	No
45	IOEXP25	Inout	LVTTL	K18	☑	ADLIB:BIBUF	Bank3	TRISTATE	None	No
46	IOEXP26	Inout	LVTTL	B1	☑	ADLIB:BIBUF	Bank8	TRISTATE	None	No
47	IOEXP27	Inout	LVTTL	B2	☑	ADLIB:BIBUF	Bank8	TRISTATE	None	No
48	IOEXP28	Inout	LVTTL	C3	☑	ADLIB:BIBUF	Bank8	TRISTATE	None	No
49	IOEXP29	Inout	LVTTL	A2	☑	ADLIB:BIBUF	Bank8	TRISTATE	None	No
50	IOEXP30	Inout	LVTTL	C4	☑	ADLIB:BIBUF	Bank8	TRISTATE	None	No
51	IOEXP31	Inout	LVTTL	B3	☑	ADLIB:BIBUF	Bank8	TRISTATE	None	No

Figure 3 - Example of Top List I/O Ball Assignments



Port Name ▲	Direction	I/O Standard	Pin Number	Locked	Bank Name	I/O state in Flash*Freeze mode	Resistor Pull	I/O available in Flash*Freeze mode	Schmitt Trigger	Odt_Static	Odt Imp (Ohm)	Low Power Exit	Input Delay	Slew
OTG_CLKOUT	Input	LVTTL	R17	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DATA0	Inout	LVTTL	V22	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DATA1	Inout	LVTTL	V21	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DATA2	Inout	LVTTL	U22	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DATA3	Inout	LVTTL	U21	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DATA4	Inout	LVTTL	T20	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DATA5	Inout	LVTTL	T21	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DATA6	Inout	LVTTL	P17	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DATA7	Inout	LVTTL	V19	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_DIR	Input	LVTTL	R18	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_NXT	Input	LVTTL	T19	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
OTG_STP	Output	LVTTL	T18	☑	Bank3	TRISTATE	None	No	--	--	--	--	--	--
REGSPR	Output	LVTTL	Unassigned	--	--	TRISTATE	None	No	--	--	--	--	--	--
RS485RX	Input	LVCMOS25	AB18	☑	Bank5	TRISTATE	None	No	Off	--	--	Off	Off	--
RS485TX	Output	LVTTL	U19	☑	Bank3	TRISTATE	None	No	--	--	--	--	--	--
SPI_0_CLK	Inout	LVTTL	N19	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
SPI_0_DI	Input	LVTTL	N20	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
SPI_0_DO	Output	LVTTL	N21	☑	Bank3	TRISTATE	None	No	--	--	--	--	--	--
SPI_0_SS0	Inout	LVTTL	N22	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
SPI_1_CLK	Inout	LVTTL	M21	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
SPI_1_DI	Input	LVTTL	M22	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
SPI_1_DO	Output	LVTTL	L21	☑	Bank3	TRISTATE	None	No	--	--	--	--	--	--
SPI_1_SS0	Inout	LVTTL	L20	☑	Bank3	TRISTATE	None	No	Off	--	--	Off	Off	--
TEMP_SENSOR	Input	LVTTL	E5	☑	Bank8	TRISTATE	None	No	Off	--	--	Off	Off	--
USER_FAB_RESET_N	Output	LVTTL	Unassigned	--	--	TRISTATE	None	No	--	--	--	--	--	--
XTL	Input		AB21	☑	--	--	--	--	--	--	--	--	--	--

Figure 4 - Example of Bottom List I/O Ball Assignments

The I/O assignment dialog is selected from the I/O Constraints anchor in Place and Route section (*below*)

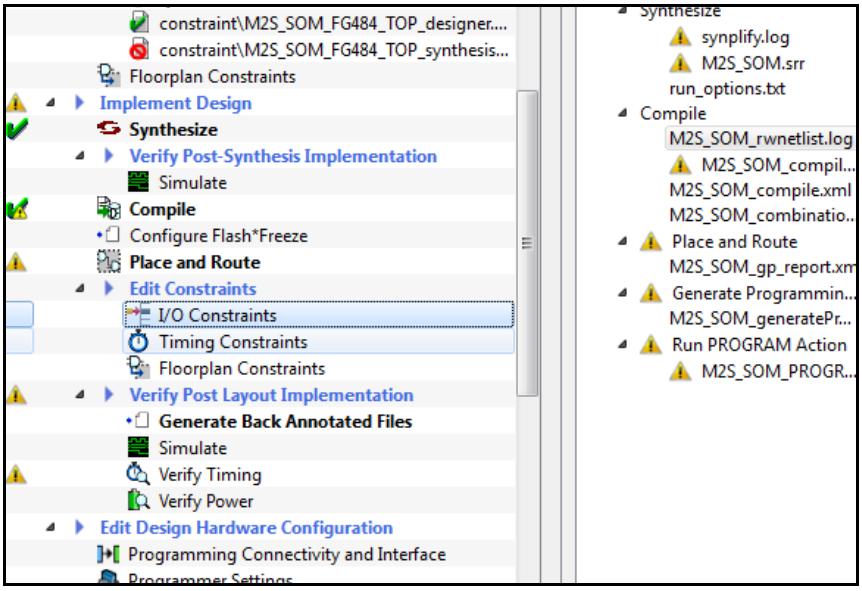


Figure 5 - Selecting the I/O Constraints

4 NetFusion-EXP Libero 11.X IDE/SoC Top-Level

Within the NetFusion-EXP starter Libero project, the M2S_SOM Top-Level sheet illustrates the lower modules to the design just before the I/O ports go out to the real-world PCB. The ports labeled in this sheet correspond to the I/O assignments (*shown in the previous section*). This sheet is effectively the linkage from the inner SOM design (*see next section*) up to the I/O balls of the SmartFusion2 FPGA.

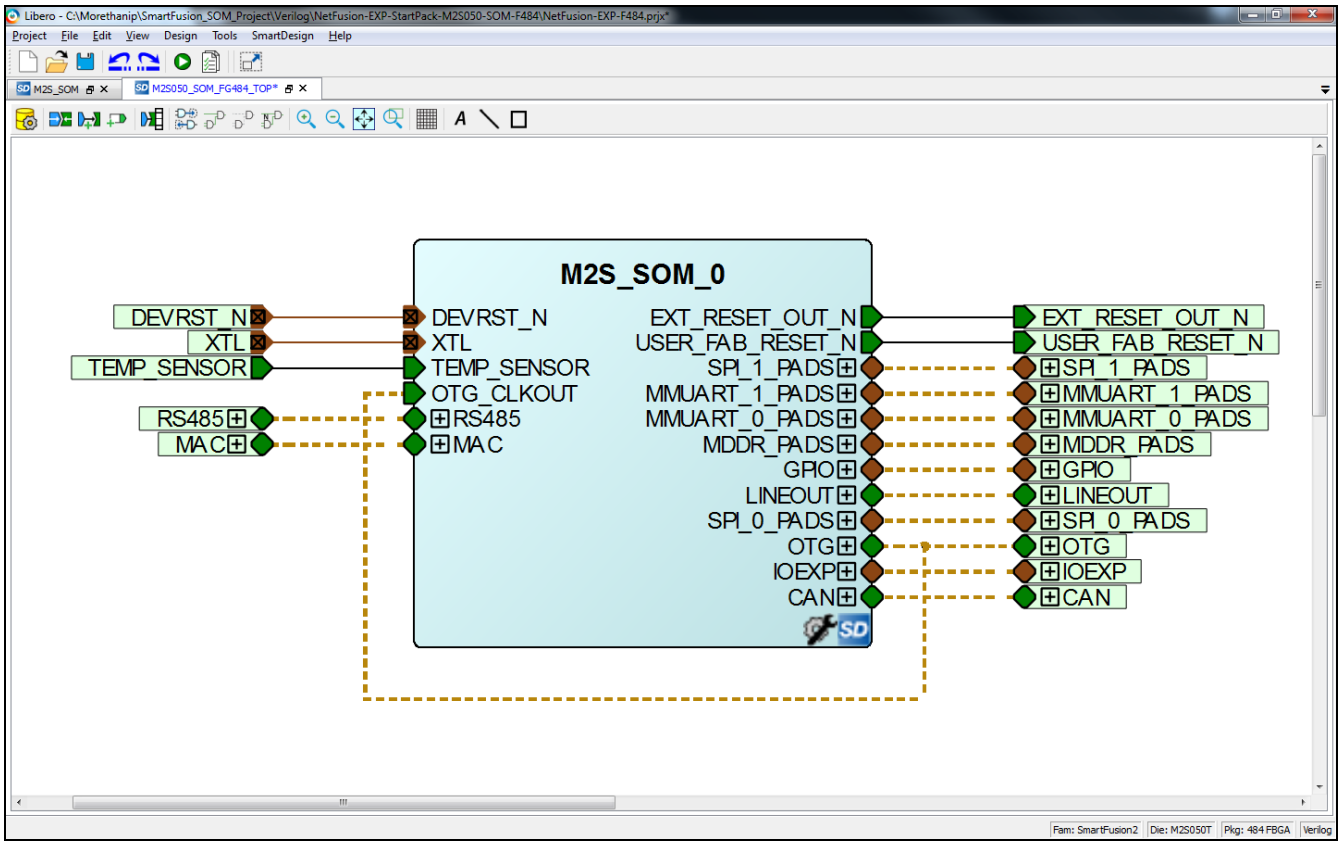


Figure 6 – NetFusion-EXP Top-Level Sheet

When lower level SOM sheets are built and prepared, they propagate information up automatically to this higher sheet. New ports suddenly appear and you must then compile this sheet before the main synthesis.

Note: *you can instantiate normal IP cores into this top sheet if you wish and it makes sense according to your design requirements. This is just the starter project so everything by default is kept in the lower SOM module. But this can change rapidly as your customization starts to take effect.*

The lower SOM sheet of the NetFusion-EXP starter Libero project contains all of the default designs and linkage.

This SOM sheet lower layer contains the ARM Cortex-M3 MSS block and then all of the necessary associated IP core blocks to allow the processor, the Emcraft SOM and the wider surrounding NetFusion-EXP baseboard hardware to operate and be accessed.

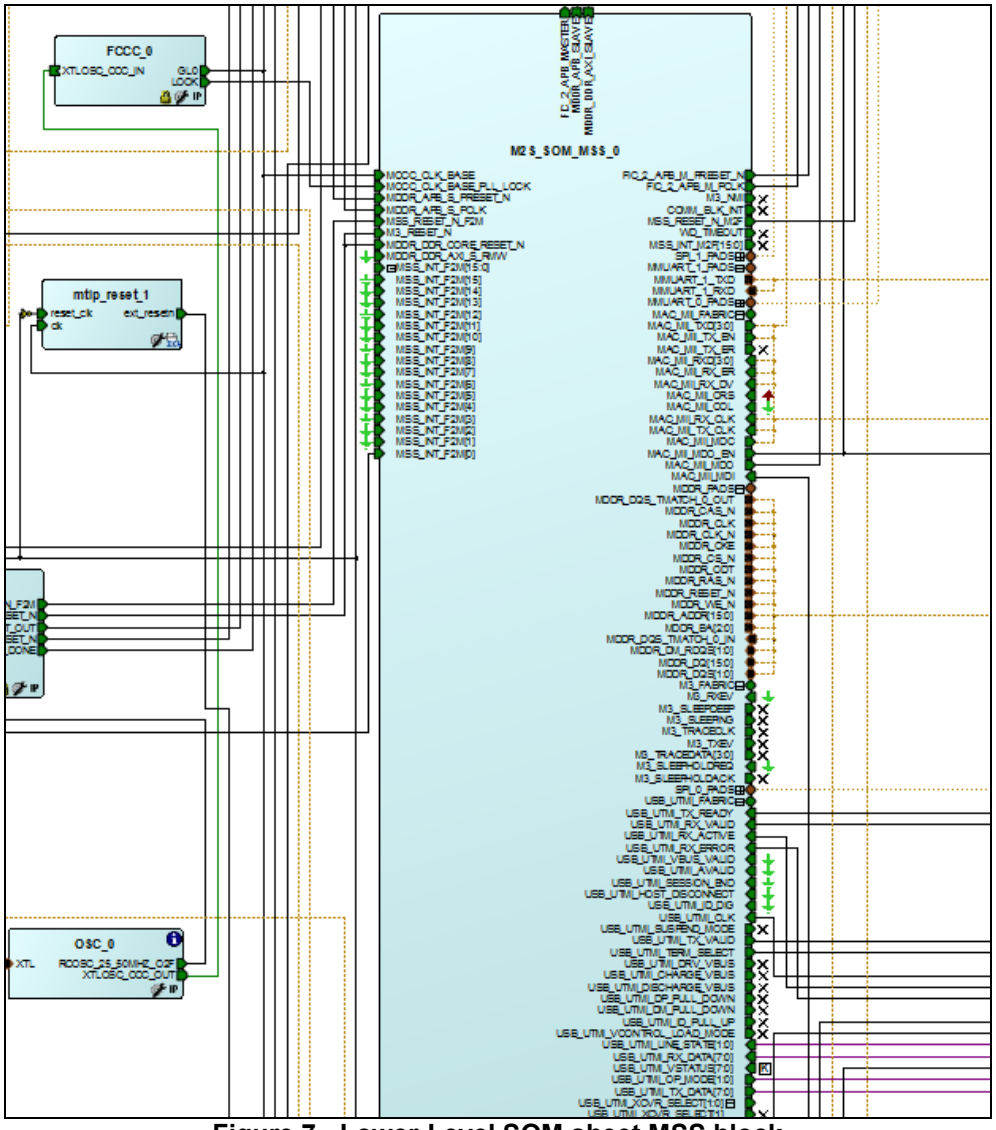


Figure 7 - Lower-Level SOM sheet MSS block

The next screenshot shows a rats-nest but the following sub-sections illustrate the different parts in more detail. **Note:** *Incidentally, the largest module center-left is the MSS ARM cortex processor (zoomed above).*

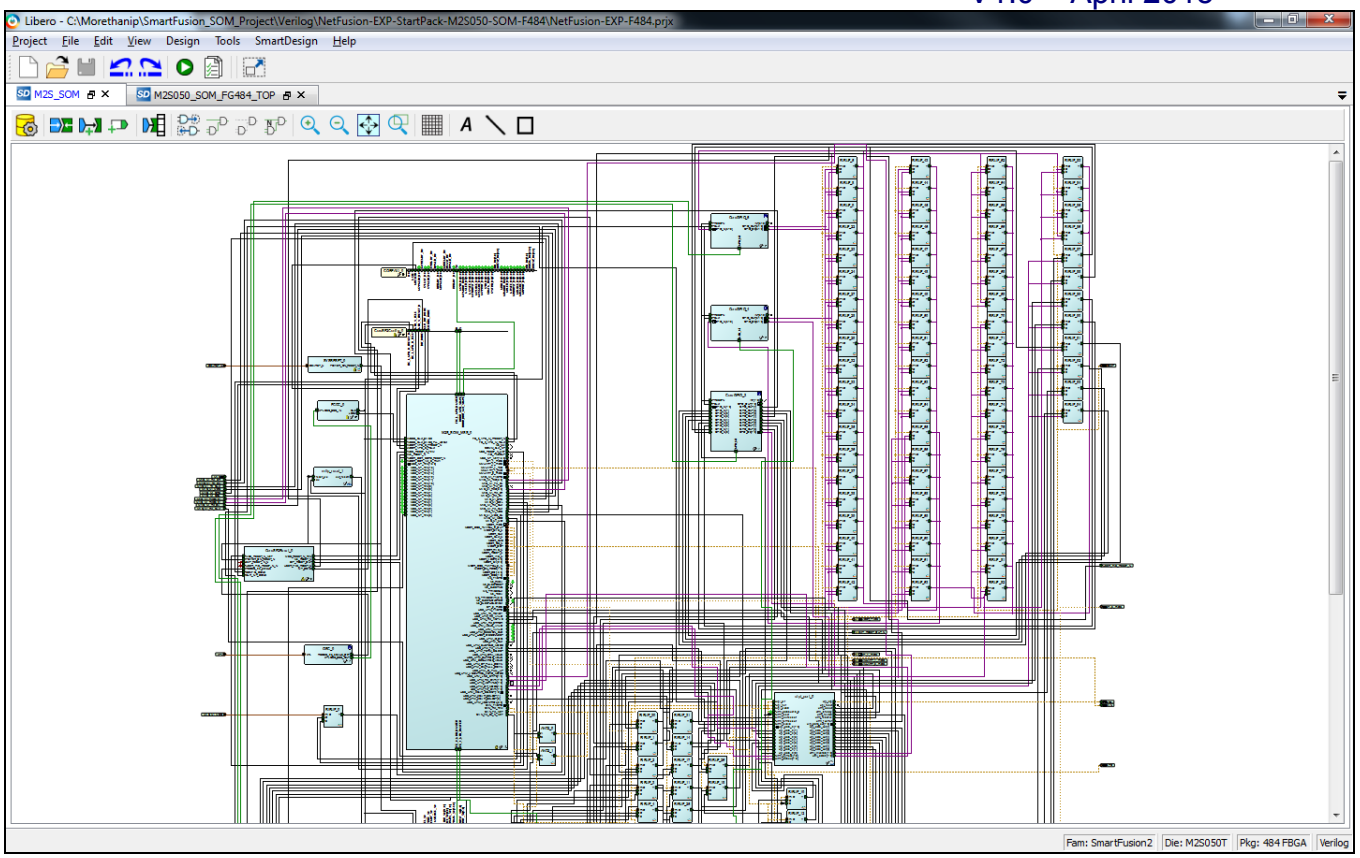


Figure 8 - Screenshot of Lower-Level SOM sheet of the Starter-Pack Libero Project

4.1 Emcraft Systems

The (*above*) figure overview encapsulates the whole of the NetFusion-EXP SOM sheet. However, it was built on the basic project supplied currently by Emcraft. They designed and developed the M2S-SOM-F484 that is housed on the NetFusion-EXP baseboard PCB.

Various IP cores were added to accommodate the need to support and facilitate the vast multitude of NetFusion-EXP's PCB hardware available to the ARM MSS sub-system in the SmartFusion2 FPGA on the Emcraft M2S-SOM-F484 housed on NetFusion-EXP as one product. As the MSS can only drive and support some of the pins on the SOM unit, there was a requirement therefore, to add more IP cores in the fabric to interface through the I/O assignments to the PCB hardware not connected directly to the MSS.

These additional cores and the default statutory parts of the SOM design in the Fabric are described in the sub-sections (*below).*

4.2 ULPI/UTMI OTG USB

On the SmartFusion2 F484 package that is used with Emcraft's M2S-SOM-F484 System-On-Module, the ULPI MSS interface is not supported. However, the UTMI OTG USB signals are supported. As the track routing and the IC USB device on the NetFusion-EXP PCB support ULPI, an IP core in the fabric is require to convert between the two different USB On-The-Go protocols and signals. This has been instantiated as [*ulpi_port_0*].

This OTG core was used from OpenCores at http://www.opencores.com and resides in the NetFusion-EXP starter project's ***hdl/*** sub-directory.

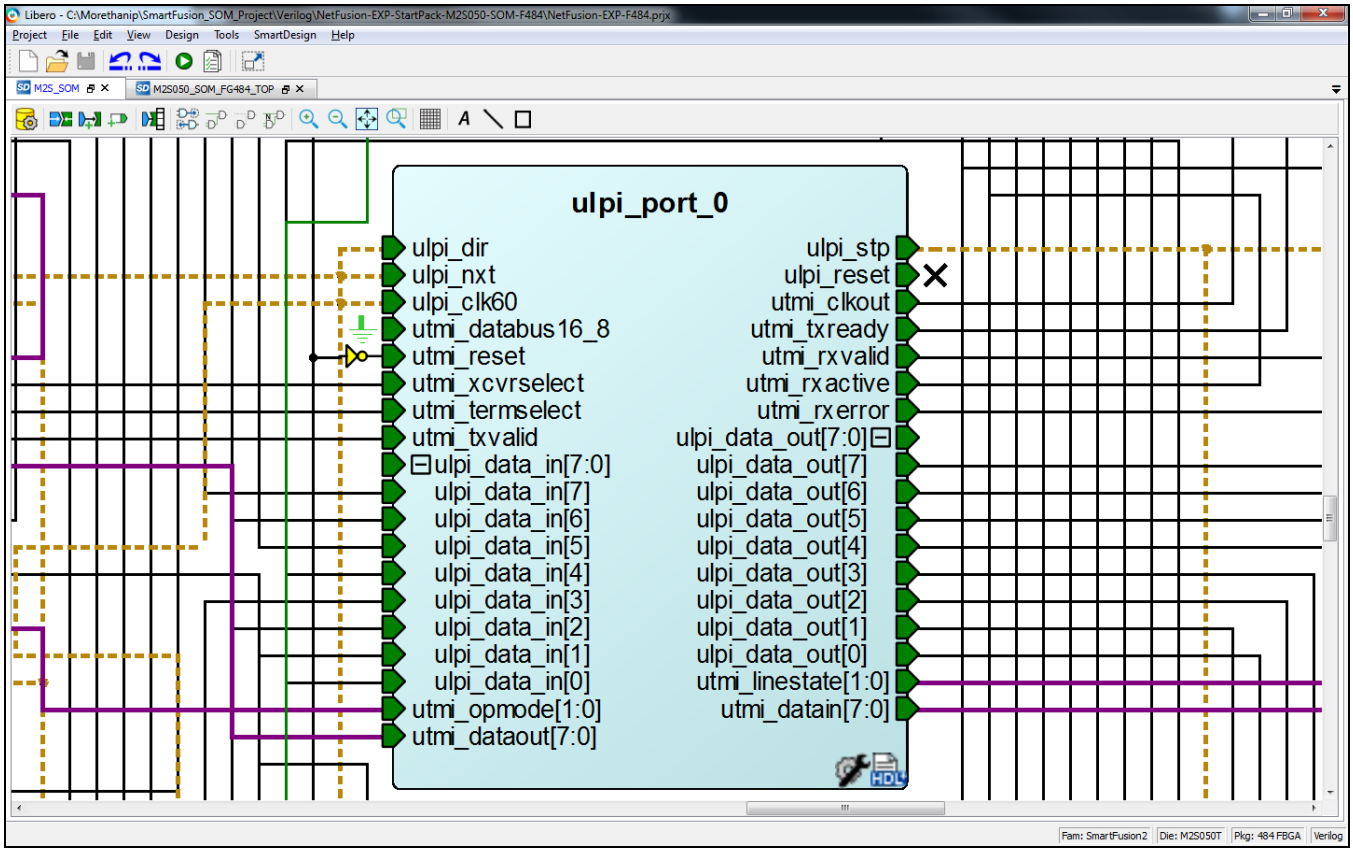


Figure 9 - ULPI/UTMI USB Converter IP Core

4.3 CoreSF2Reset

This core is used to co-ordinate the reset across all of the FPGA in strict timed sequence. It is important for the peripheral logic to be released after the MSS ARM processor and then the fabric logic following the MSS. This is a standard Libero ACTEL core from the vault.

The **EXT_RESET_OUT** signal is routed out of the FPGA device to the NetFusion-EXP PCB. The input to this core is **the POWER_ON_RESET_N**. This core was instantiated as [*CoreSF2Reset_0*].

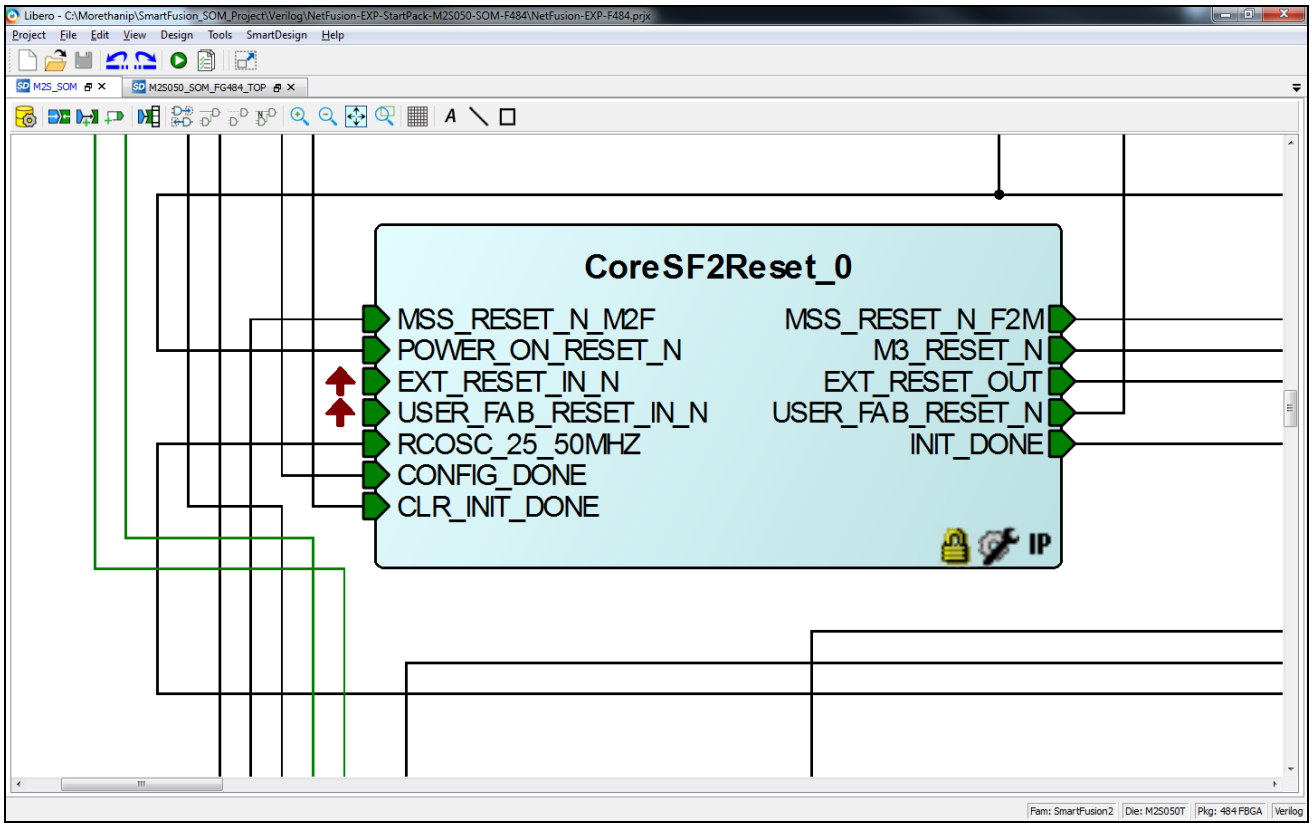


Figure 10 - SmartFusion2 Reset Controller IP Core

4.4 FCCC

The entire FPGA sub-system comprising of the MSS ARM processor and the main fabric run on clocks all generated from this core. It is the main coordination of the clock lines that are distributed. The input is *12MHz* from an off-chip crystal/IC. The CCC PLL divides down the *12MHz* source by **12** to *1MHz*. Then this is multiplied up by differing amounts for **GL0**. You can add more clock PLL lines as you wish when you are modifying the design.

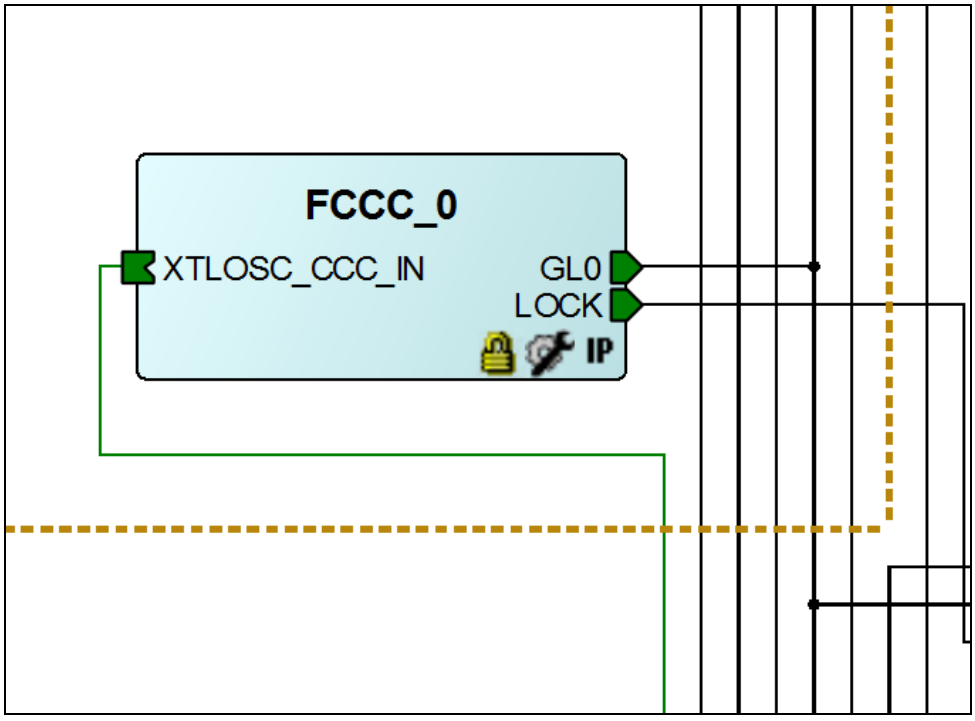


Figure 11 - Clock PLL Macro Core

The core as been instantiated as [F*CCC_0*]. There is also a **LOCK** output signal that is used by the MSS to determine when the PLL has settled and locked onto the desired output frequencies. All clock outputs are digital square waves.

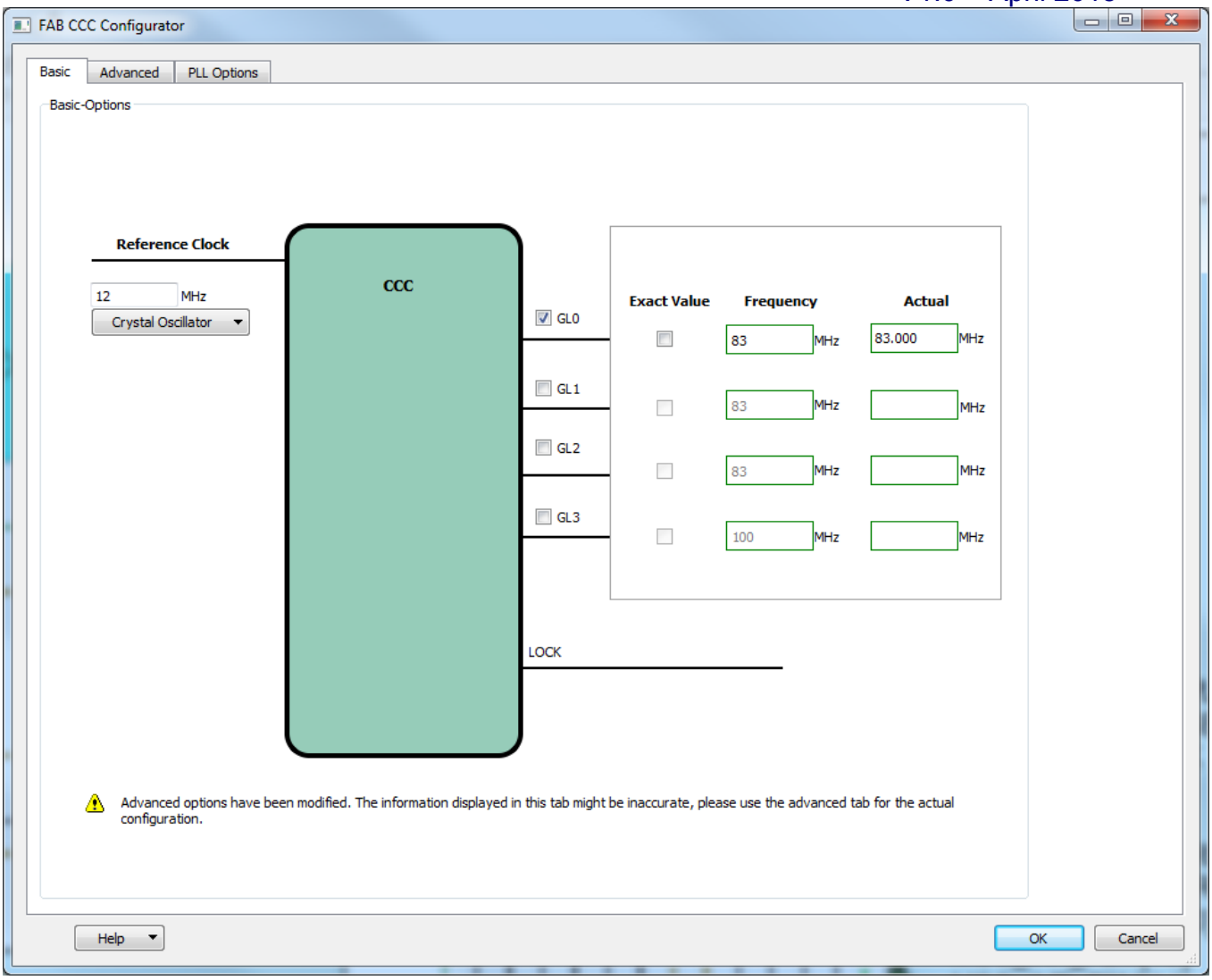


Figure 12 - PLL Clock Macro Settings

Output signal **GL0** is distributed to the memory bus and the MSS ARM block. **T**he main base clock frequency for the *166MHz* ARM processor is scaled up inside the MSS block.

4.5 Counter - CoreGPIO

NetFusion-EXP PCB has an on-board temperature sensor. The output of which is a square wave signal that has a frequency is equivalent to degrees Kelvin (*down to absolute zero*). This signal is routed into the SmartFusion2 FPGA fabric and clocks the *16-bit* counter [*counter16_0*]. The counter's *16-bit* output value is wired into GPIO input [*coreGPIO_1*] which is memory addressable from the ARM uClinux applications. This serves as a simple *32-bit* memory location to read and makes the software algorithm for determining the temperature incredibly simple.

Note: *the memory interface from the ARM MSS is an APB interface.*

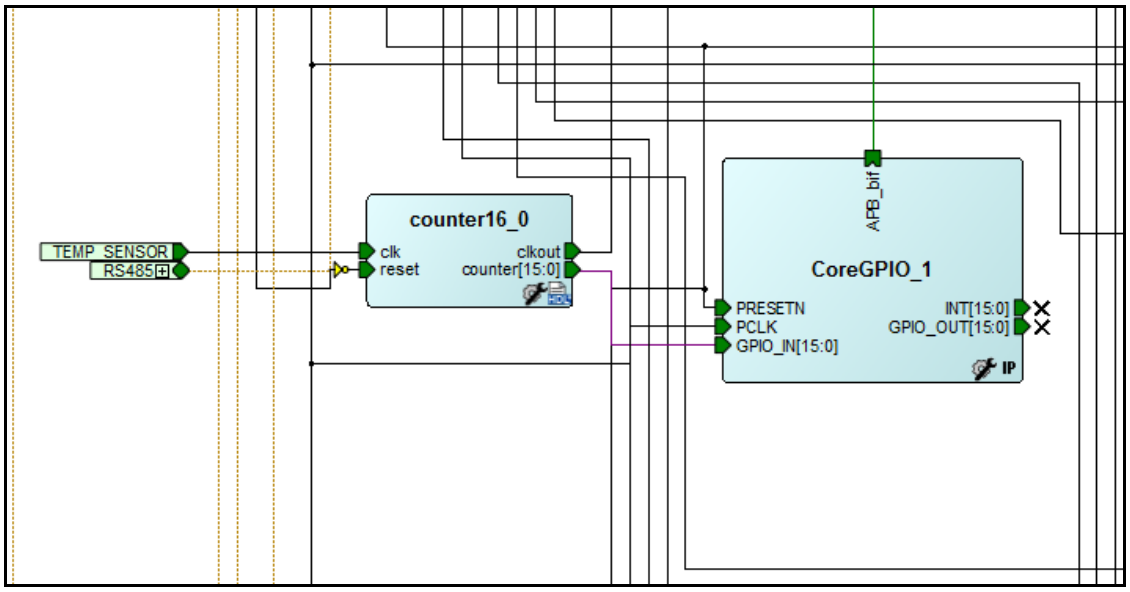


Figure 13 - Temperature Sensor Logic

4.6 NetFusion-EXP Baseboard Peripheral CoreGPIO

The vast majority of the hardware on the NetFusion-EXP PCB is connected through this core. The obvious exceptions are the Ethernet pathways, *RS485 UART* and the *SPI* audio input/output. However, all of the *I2C, Relays, I/O expanders, Real-Time clock IC, voltage monitor, expansion I/O* and all other slow speed communications are handled through this [*coreGPIO_0*] instantiated block.

It is visible to the ARM MSS processor via an APB interface memory bus. Lines can be configured as inputs or outputs and are all individually controlled and observed by the uClinux UIO device driver.

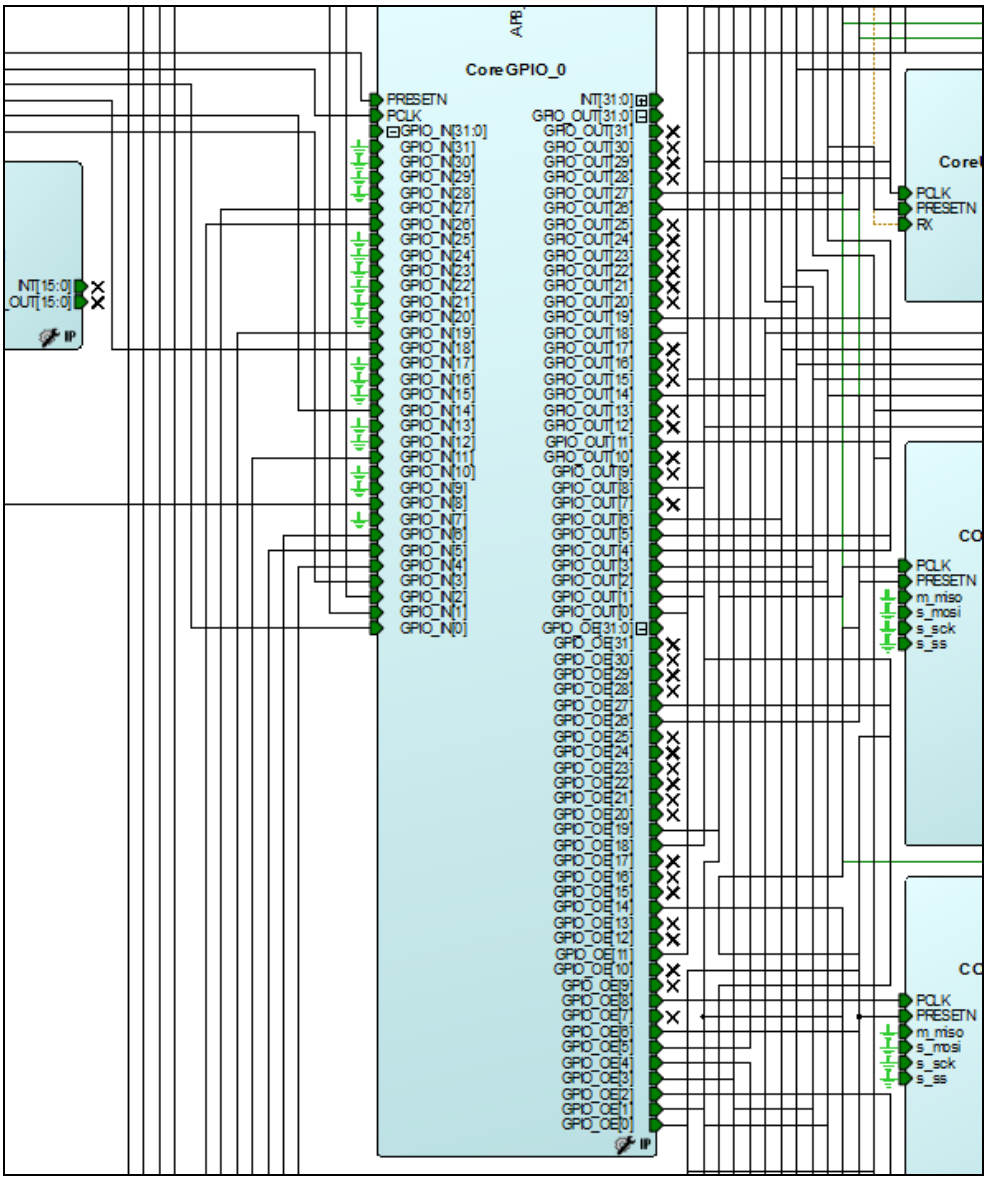


Figure 14 - Main PCB GPIO Interface with APB bus from the ARM Processor MSS block

4.7 NetFusion-EXP IO-EXP Daughter Card GPIO

The IO-EXP Daughter Card expansion slots on the NetFusion-EXP PCB are connected through these cores. The expansion slot functionality is handled through [*coreGPIO_3, coreGPIO_4, coreGPIO_5*] instantiated block and as a basic example for the user and all 72 GPIOs for the expansion daughter-card are connected to these cores for the ARM processor software to handle. It will in most cases of user development, be changed from this processor visible design to a bespoke customer logic design, but the initial design in this provided Libero project allows simple initial software testing and probing before major design changes are implemented.

The CoreGPIO blocks are visible to the ARM MSS processor via an APB interface memory bus. Lines can be configured as inputs or outputs and are all individually controlled and observed by the uClinux UIO device driver.

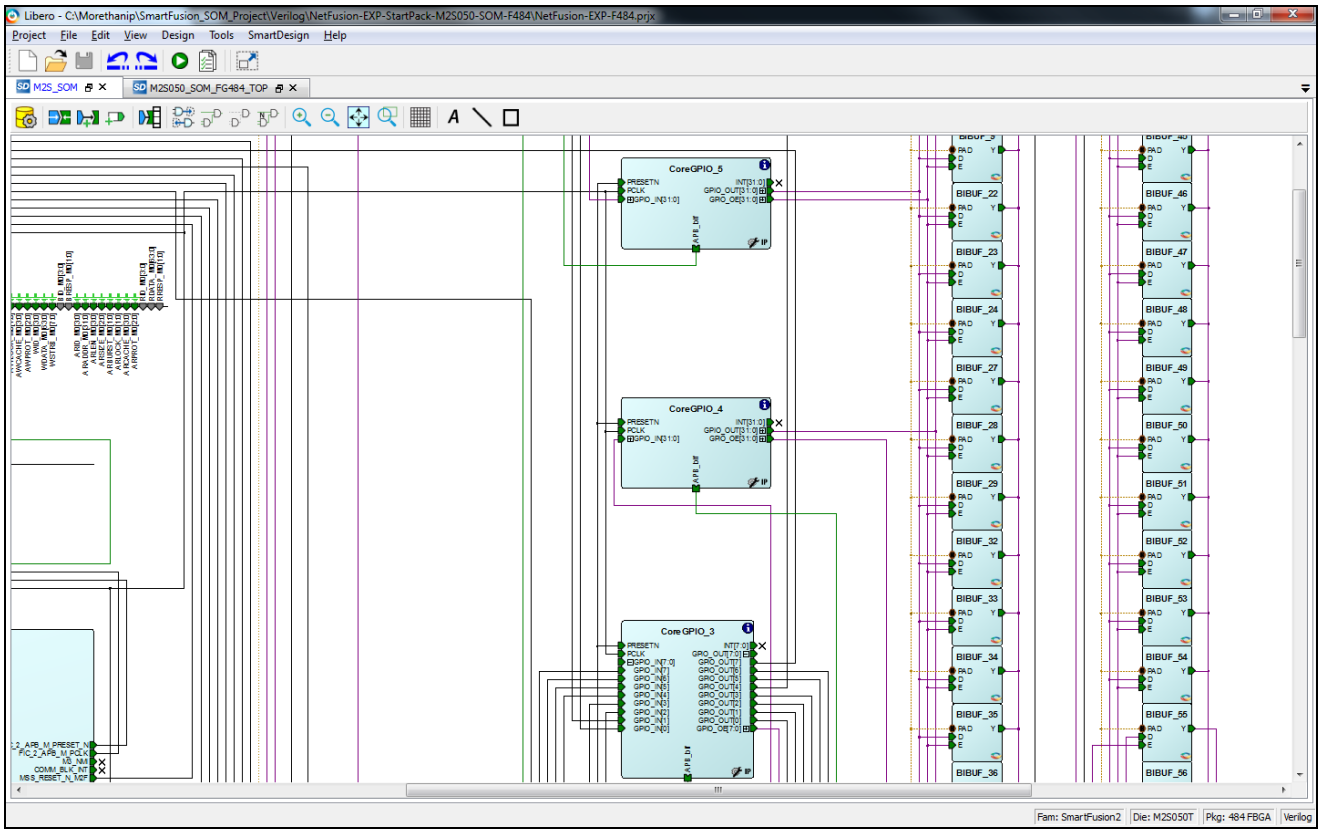


Figure 15 – Main CoreGPIO for the Daughter Card Expansion IO slots

4.8 BIBUF Bi-Direction buffers for Daughter Card GPIO

Throughout the NetFusion-EXP starter project's fabric logic design is scattered around Bi-Directional macros. However, the group shown (*below*) pertains to the bi-directional switching capability of the IO lines for ALL 72 IO-EXP daughter card signals. These convert conveniently any IN/OUT data flow when the RX and TX are kept separate in the IP cores. The output side always routes up out of the SmartFusion2 FPGA to the NetFusion-EXP daughter card slots where the signals are capable of input/output operation using pull-up resistors combining tri-state operation.

The instantiated BIBUFs are illustrated in the Figure (*below).*

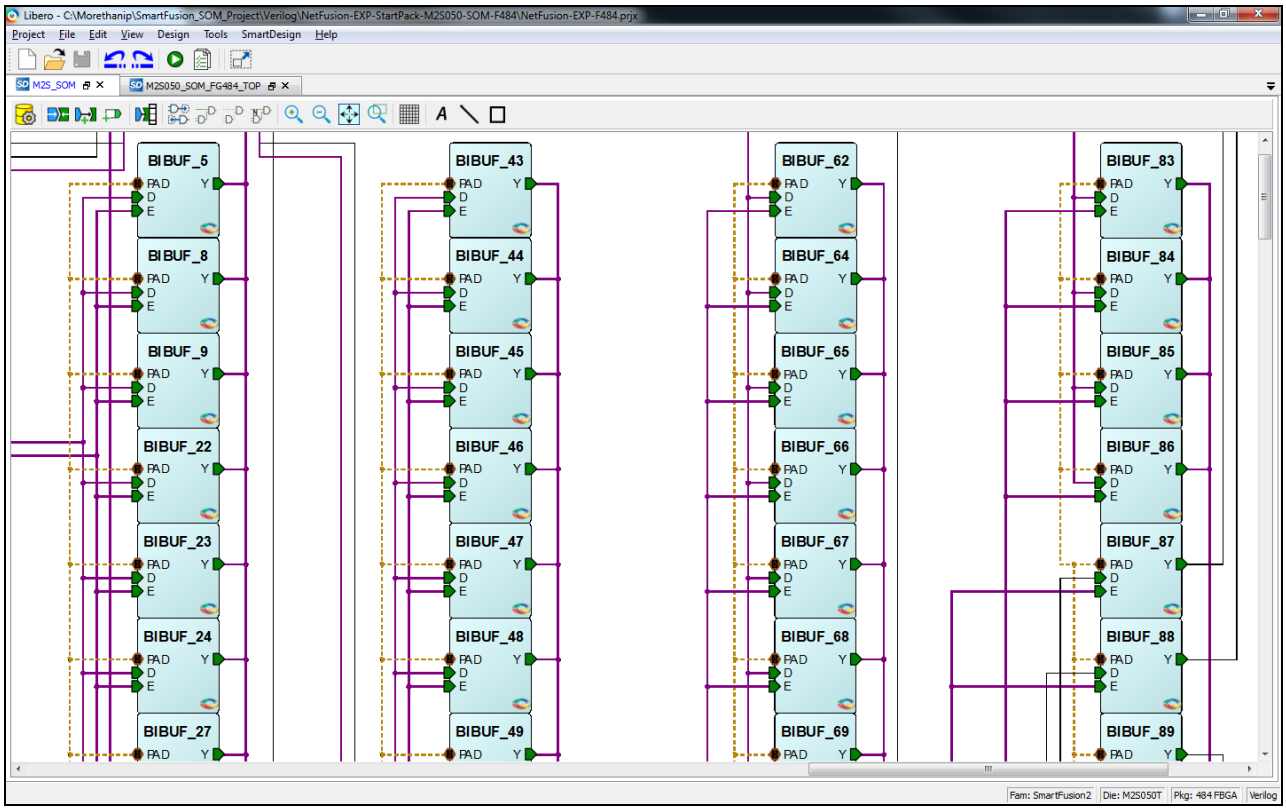


Figure 16 – Bi-Directional Buffers for the Daughter Card Expansion IO slots

Important Note: *the Libero logic design of this NetFusion-EXP start project is capable of ALL 72 IO lines to the daughter card slots to support input or output configuration. As the user customizes and changes the design this may become more specific to input or output depending the required design. Additionally, some daughter card's themselves may have input only or output only circuitry present which means if the user is keeping this starter project design, they will still have to configure the associated GPIO core to reflect the I/O setup.*

4.9 CAN Controller

The CAN transceiver on the NetFusion-EXP baseboard PCB is connected to the CAN Controller (*inside the MSS block*) either directly for the RX and TX, or via OR gate logic for the enable lines. These are instantiated as [*OR2_0, OR2_1*] and serve as a mask to either allow *bus1*, *bus2* or both to be used during transmission. The masks are controlled by signals from the small GPIO core [*CoreGPIO_2*]. The signals from the GPIO core to mask the enable lines ensures that application software controls the dual bus behavior for the CAN interface on NetFusion-EXP. Corresponding GPIO bits that are cleared disable that CAN bus and those which are set enable it.

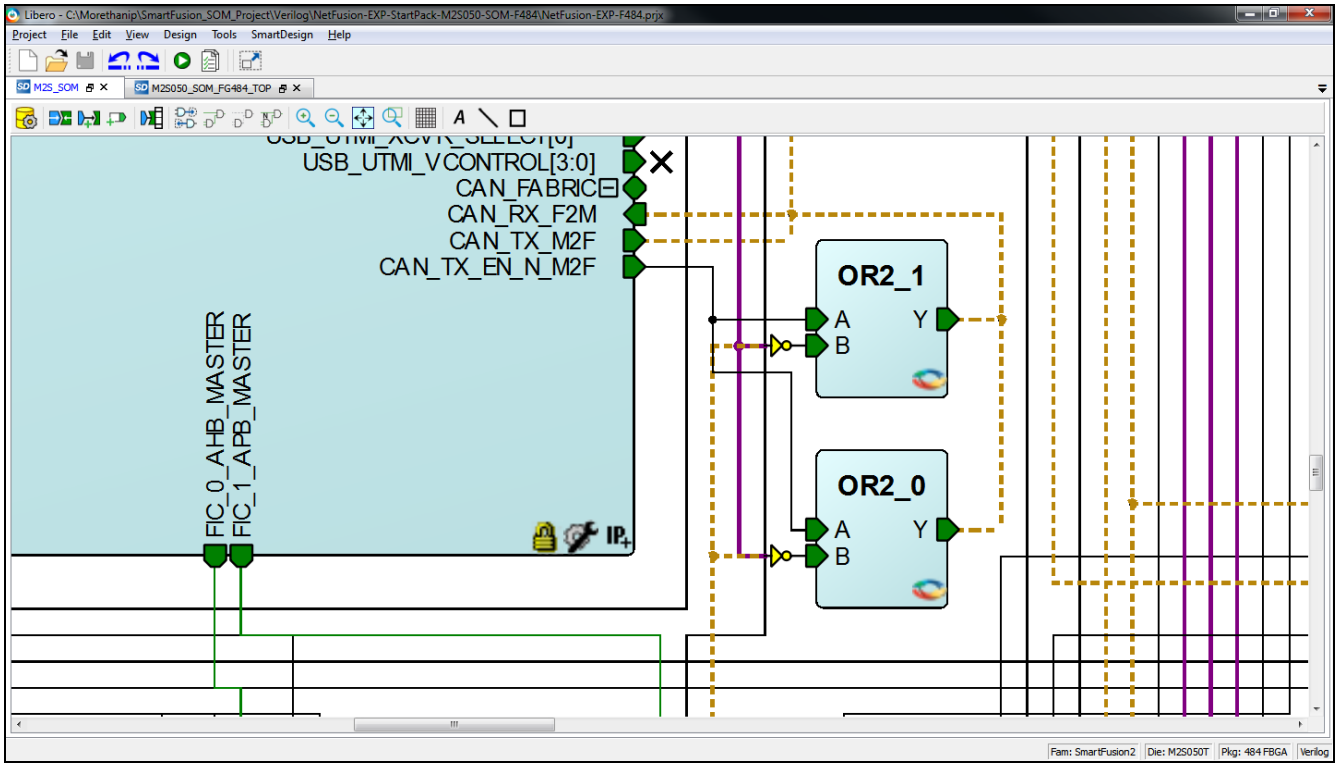


Figure 17 - CAN Controller Interface Logic

4.10 CoreUARTapb

As there is no way in the SmartFusion2 ARM MSS block for standard UART connections *(/dev/ttyS0* and */dev/ttyS1* already used) the RS484 PCB hardware has to be controlled and handled from a fabric UART core. This was instantiated as [*CoreUARTabp_0*] and has an APB connection to the MSS block.

The uClinux device driver for this hardware access the core as a block of memory and the FIFO RX and TX data is stored in the fabric core. The RX ad TX signals to the NetFusion-EXP PCB hardware are TTL levels and then get converted to RS485 voltage signals in the electronics.

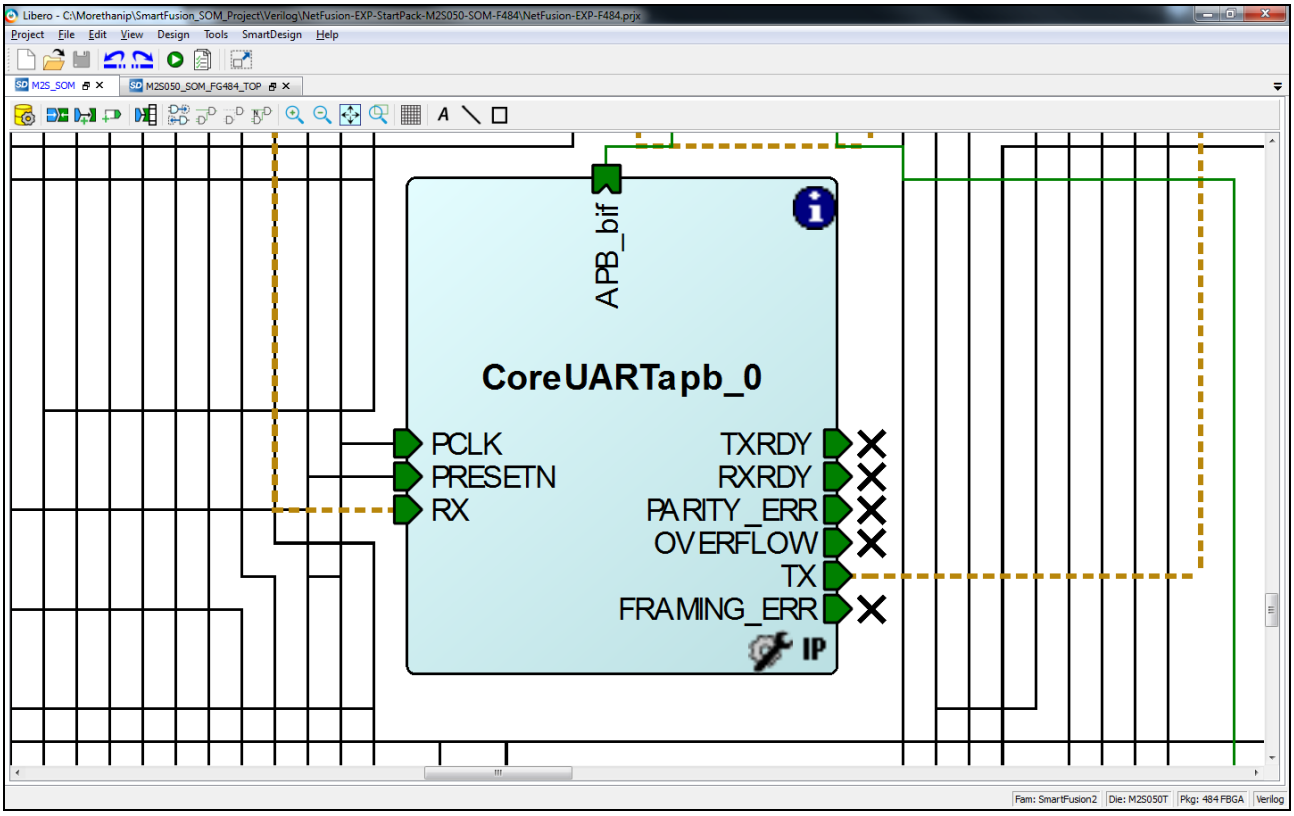


Figure 18 - RS485 UART IP core

4.11 CoreSPI0, 1

Running as separated IP cores instantiated in the fabric to achieve higher speed sampling by SPI software are SPI blocks. These are [*CORESPI_0* and *CORESPI_1*] and use an APB interface to the ARM MSS sub-system. Each core has a *4-wire* SPI bus routed out of the FPGA to the wider NetFusion-EXP PCB hardware.

These SPI interfaces connect to DAC and ADC stereo IC devices. This allows for audio to be sent and received from the PCB and the digital samples can be processed by the ARM processor from network traffic if necessary. SPI is used as it is full duplex and runs at over *1MHz* during operation. However, bottlenecks in the processor application code and also human audible hearing limitations keep realistic sampling operations around < *10KHz*.

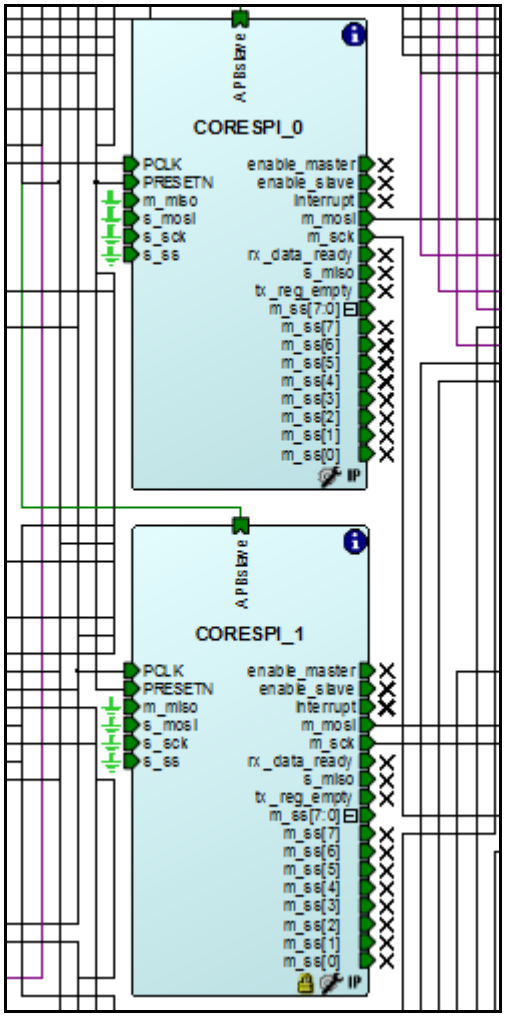


Figure 19 - Stereo Audio Line IN & Line OUT

4.12 Temperature Sensor Glue Logic

The baseboard temperature sensor IC is mounted under the Emcraft **SOM** daughter card and monitors the temperature of the FPGA that can often be demanded to perform high switching tasks in the fabric of the device.

The sensor produces an output square wave in which the frequency in *Hz* is equal to the degrees Kelvin on the baseboard. This is routed into the FPGA and the fabric and is represented on the SOM design sheet as "**TEMP_SENSOR**". This in turn, clocks the [*counter16_0*] which has a *16-bit* register output, and can be read into the ARM software environment via the [*CoreGPIO_1*] APB3 core.

Although this read value wraps around *16-bit* values, the software can use a one second timer to run comparisons and then calculate the temperature in Kelvin.

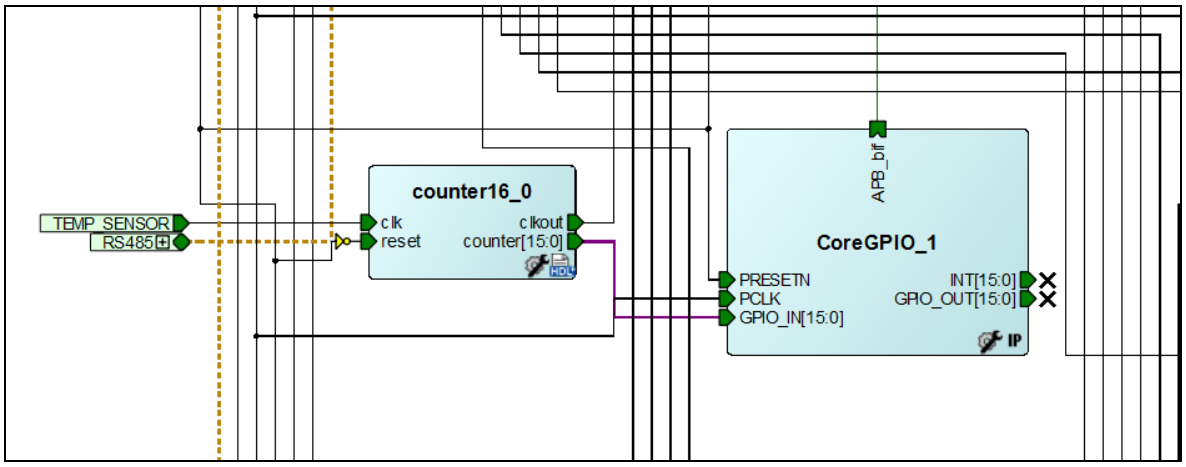


Figure 20 - Temp Sensor Counter Logic

4.13 Coordinated RESET

The customized IP core instantiated as [*mtip_reset_1*] coordinates a longer reset pulse than the power on reset core often provides. The source [*SYSRESET_0*] feeds into the [*mtip_reset_1*] which then provides an output to the whole SOM sheet design. This includes the *ARM MSS block*, *memory cores*, *GPIO* and all peripherals.

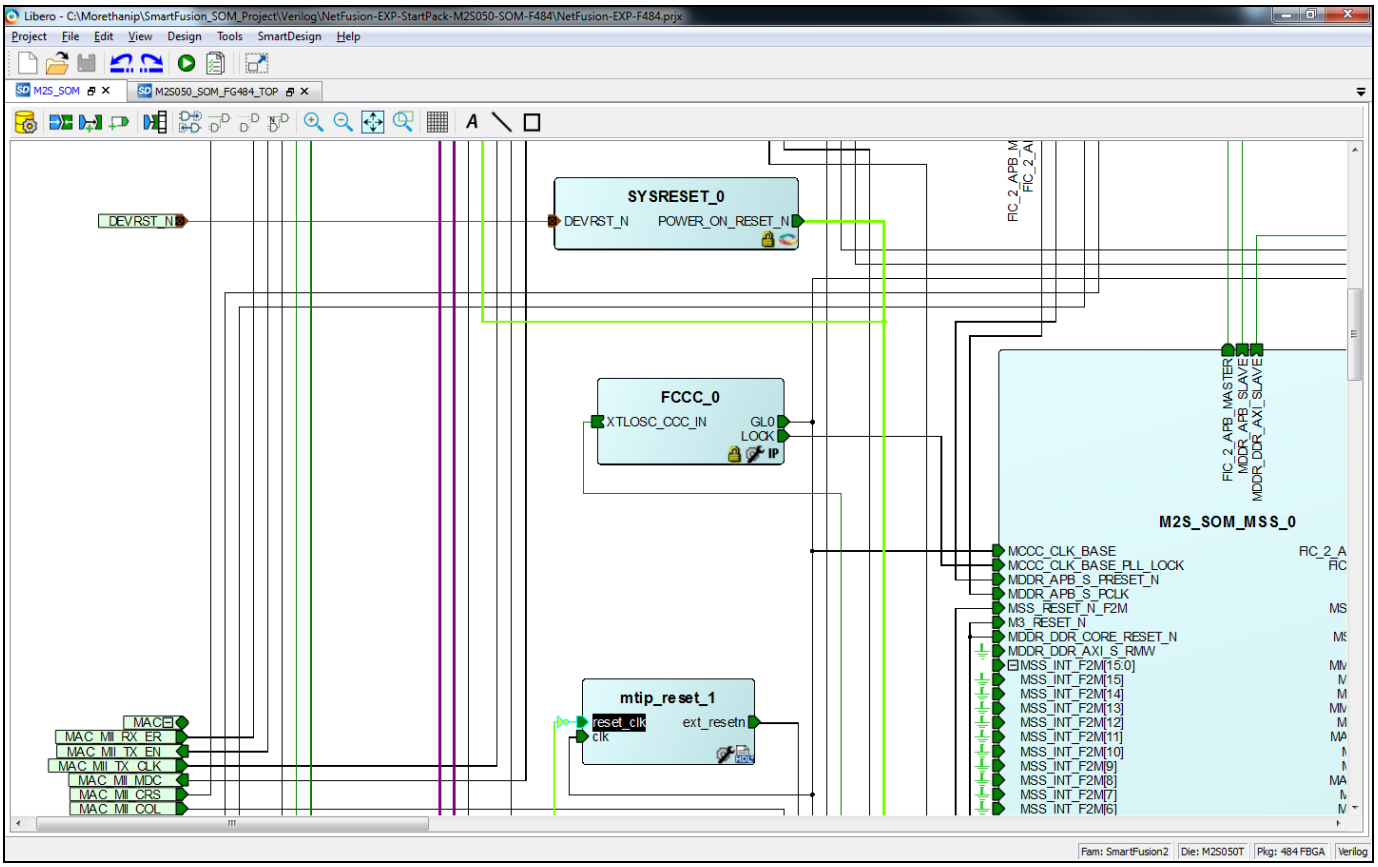


Figure 21- Customized RESET logic

4.14 BIBUF

Throughout the NetFusion-EXP starter project's fabric logic design is scattered around Bi-Directional macros. These convert conveniently any IN/OUT data flow when the RX and TX are kept separate in the IP cores. The output side always routes up out of the SmartFusion2 FPGA to the NetFusion-EXP PCB hardware where the signals are capable of input/output operation using pull-up resistors combining tri-state operation.

The instantiated BIBUFs are illustrated in the Figure (*below).*

They are controlled by an output enable signal that drives the state of the output pin.

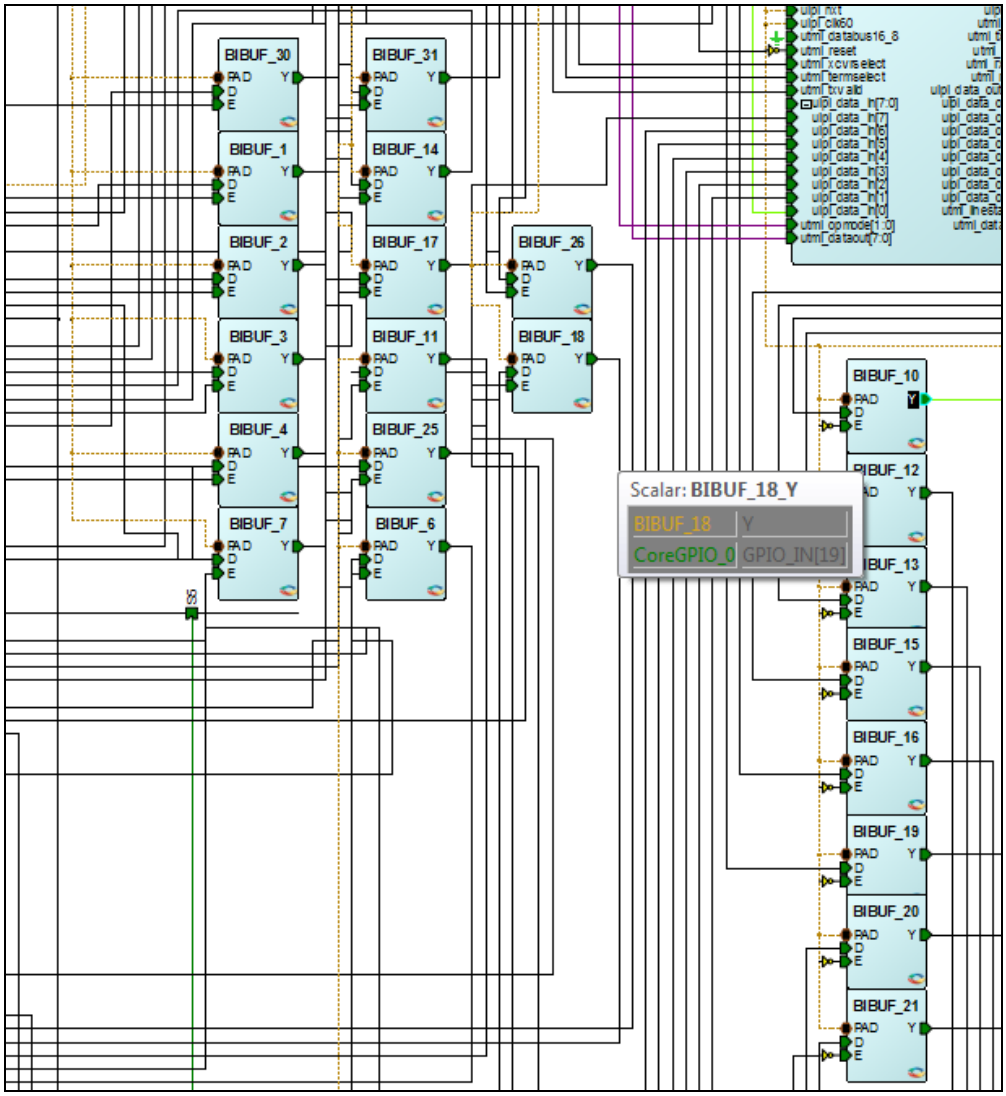


Figure 22 - Bi-Directional Fabric Macros for input/output signals

4.15 CoreSF2Config

The SmartFusion2 MSS processor sub-system always as standard has a default *CoreSF2Config* block which loops back an APB bus out and then back into the MSS block. This seems at first strange and can be very confusing. However, it is the inherent architecture of the FPGA ASIC area that most of the peripheral devices inside the MSS ARM processor core are not actually controlled by the selections made in the Libero 11.X IDE. It is the software in *u-boot* during boot-up that configures. If say for instance, an AHB-Lite interface is selected in Libero, then this does not configure the SmartFusion2 FPGA itself. It saves a configuration file that can be included by software in either bare-metal programming or the *u-boot* from Emcraft uClinux environment.

Only when the boot-up code access the APB feedback bus via [*CoreSF2Config_0*] and manipulates hardware peripheral memory address does the peripherals in the MSS get the correct mode of operation intended for them.

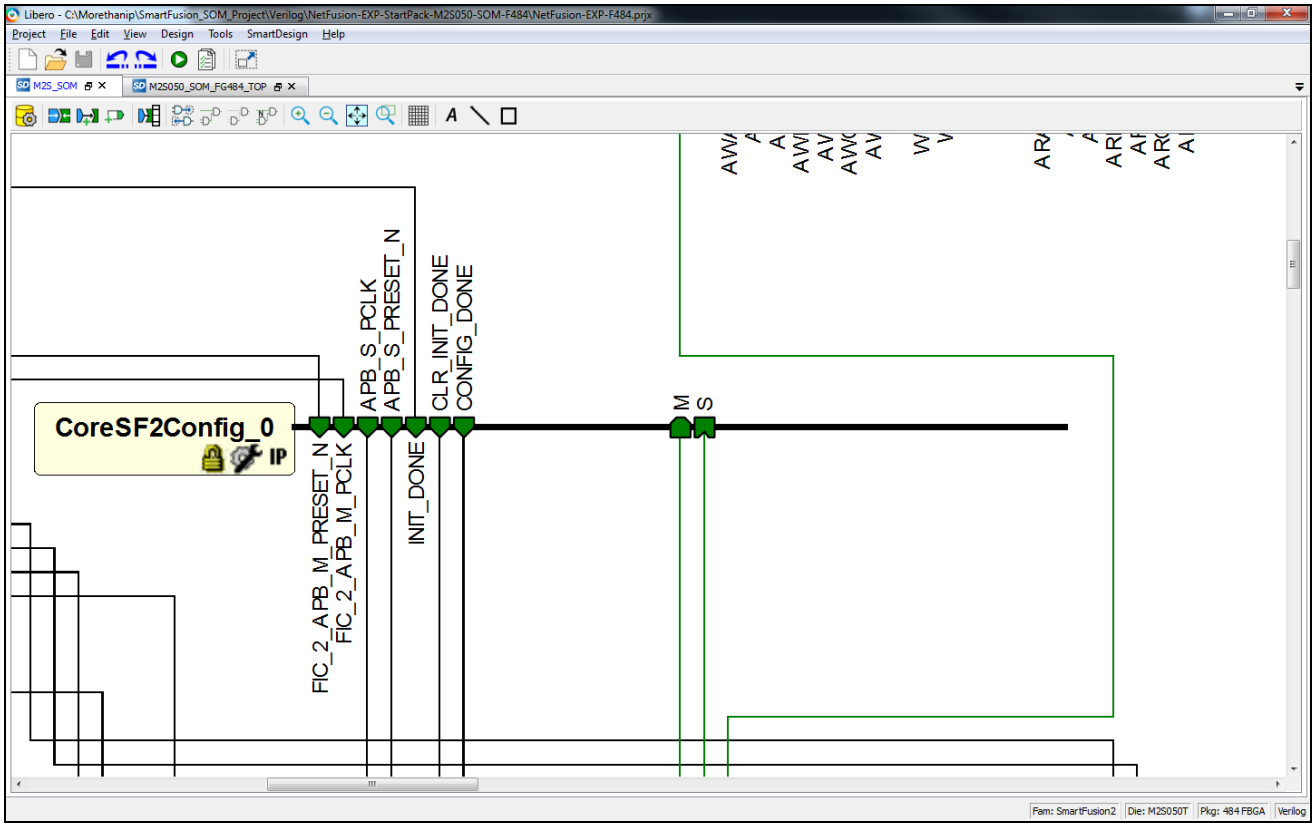


Figure 23 - APB feedback Bus for Peripheral Configuration by Software

4.16 Core AXI

In the SOM sheet of the design, there exists an IP core used for AXI memory bus accessing. It does not have any memory core attached as default when you open and view the project initially. However, this stub is the user to use to attach to any AXI memory bus interface IP core that they wish to use. All of the signals not used are either tied *LOW* (or *HIGH* if required) for input signals, and marked as unused if output signals.

The AXI core is instantiated as [*COREAXI_0]* in the SOM sheet design and currently has no intended functionality other than providing the user with a hook access point to the AXI ARM MSS system. The AXI core has an associated location in the overall ARM AHB matrix memory map and therefore attached IP cores can then have hardware address visibility from the ARM processor software environment.

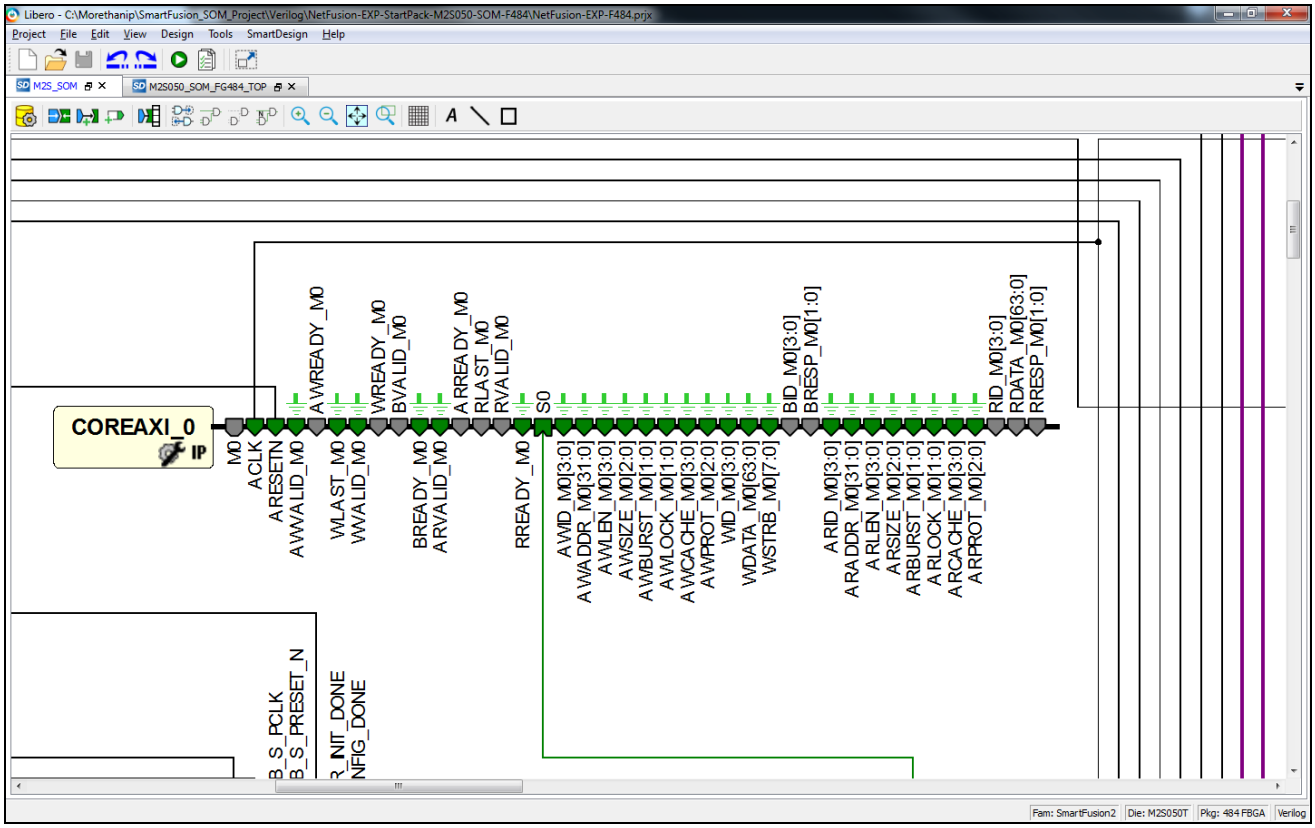


Figure 24 - AMBA Memory Interfaces from ARM for an AXI bus

4.17 Core AHB-Lite

In the SOM sheet of the design, there exists an IP core used for AHB-lite memory bus accessing. It does not have any memory core attached as default when you open and view the project initially. However, this stub is the user to use to attach to any AHB-lite memory bus interface IP core that they wish to use. All of the signals not used are either tied *LOW* (or *HIGH* if required) for input signals, and marked as unused if output signals.

The AHB core is instantiated as [*CoreAHBLite_0]* in the SOM sheet design and currently has no intended functionality other than providing the user with a hook access point to the AHB ARM MSS system. The AHB core has an associated location in the overall ARM AHB matrix memory map and therefore attached IP cores can then have hardware address visibility from the ARM processor software environment.

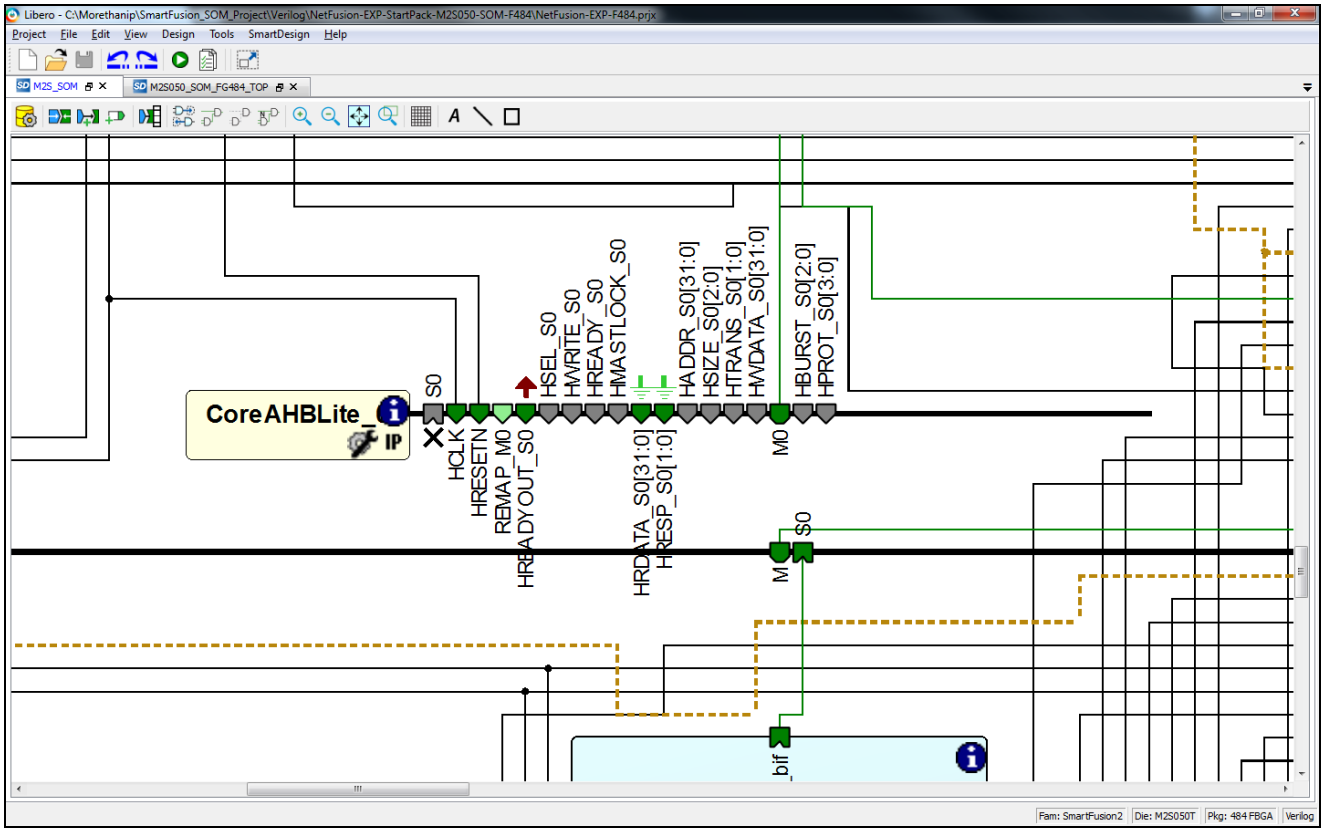


Figure 25 - AMBA Memory Interfaces from ARM for an AHB-Lite bus

4.18 Core APB3

The SOM sheet design has a useful APB3 core instantiated as [*CoreAPB3_0]* which is by default, used to collate and connect various important cores to the ARM MSS memory map. Currently the temperature sensor counter GPIO *(CoreGPIO_1),* the main GPIO core (*CoreGPIO_0*), the UART core (*CoreUARTapb_0*), the two audio SPI cores (*CORESPI_0, CORESPI_1*), the SPI baseboard device IC chip select logic (*CoreGPIO_2*) and finally the IO-Expansion GPIO for the EXP daughter-card slots (*CoreGPIO_3, CoreGPIO_4, CoreGPIO_5*).

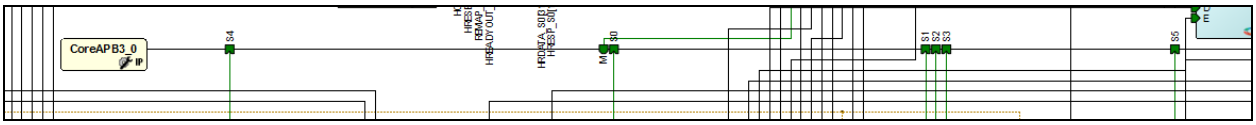


Figure 26 - APB3 fixed bus UART, GPIO, IO-EXP slots and Counter Peripherals

Although the APB3 core is used extensively, this can be extended to allow any user instantiated core to be connected by adding another slave line to the core. This can be obtained by double-clicking on the core and editing the settings.

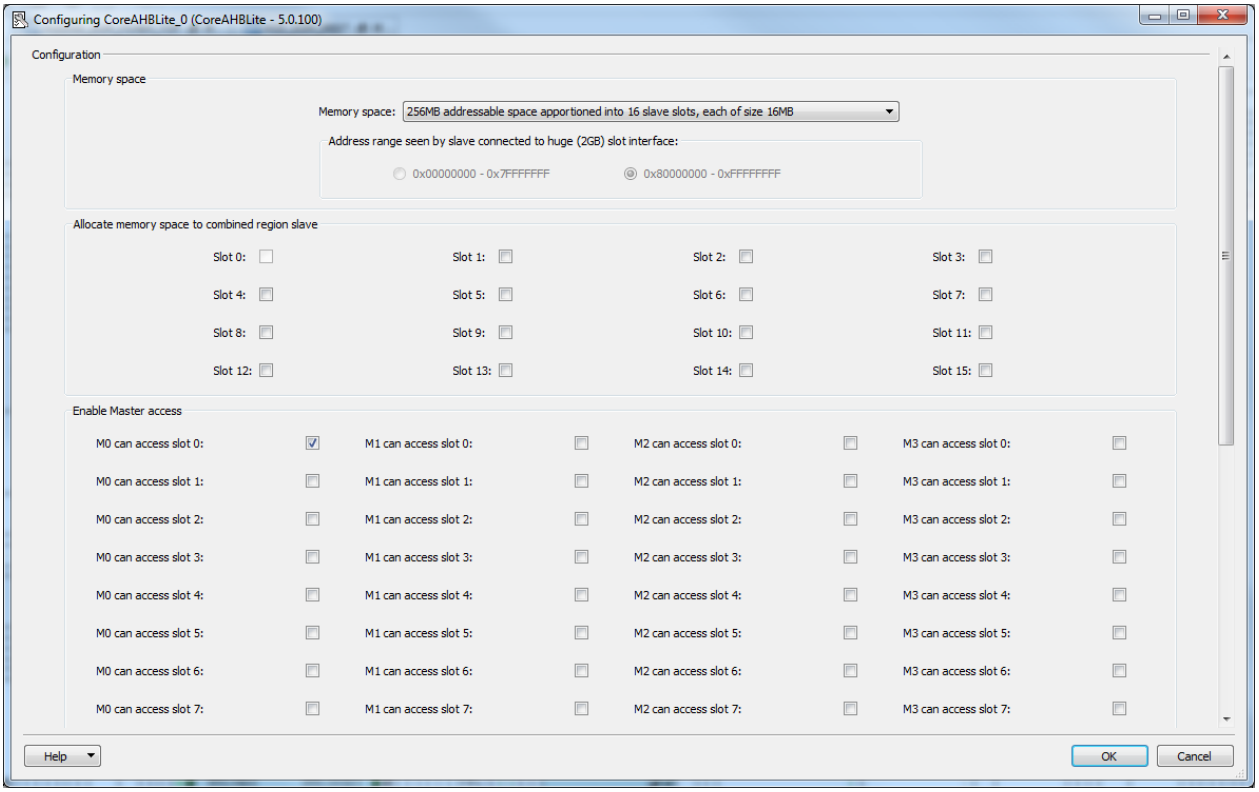


Figure 27 – Editing the APB3 core settings

Please Note: *the previous AXI and AHB-Lite cores can be edited and changed with exactly the same method by double-clicking on the cores themselves.*

4.19 AXI, AHB-Lite & APB3 default Memory Mapped Settings

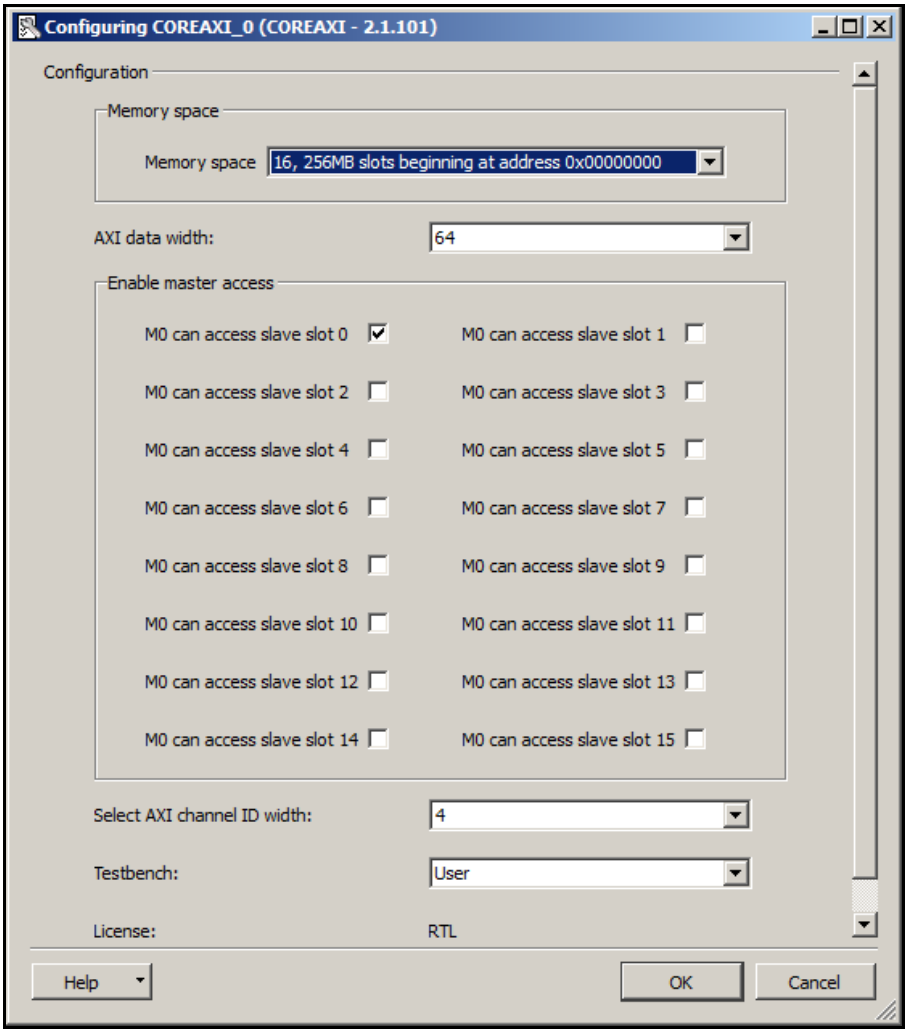


Figure 28 - Default AXI Configuration

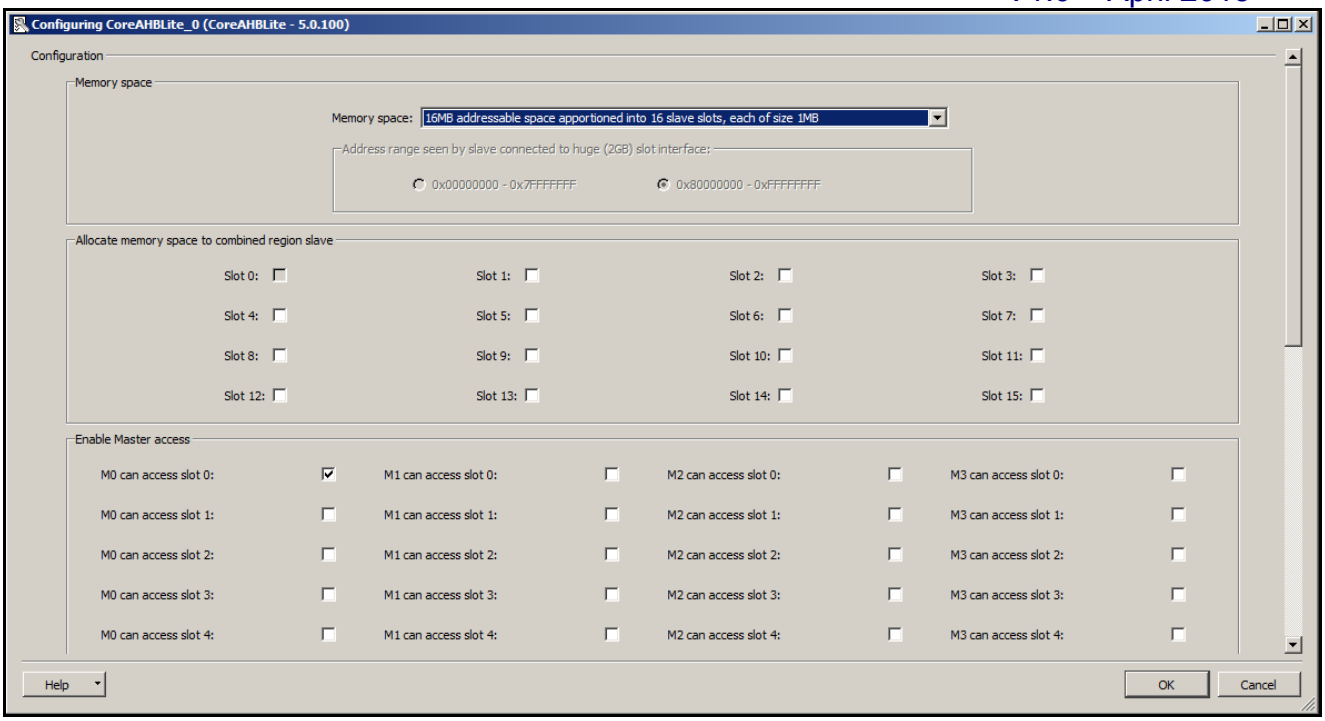


Figure 29 - Default AHB-Lite Configuration

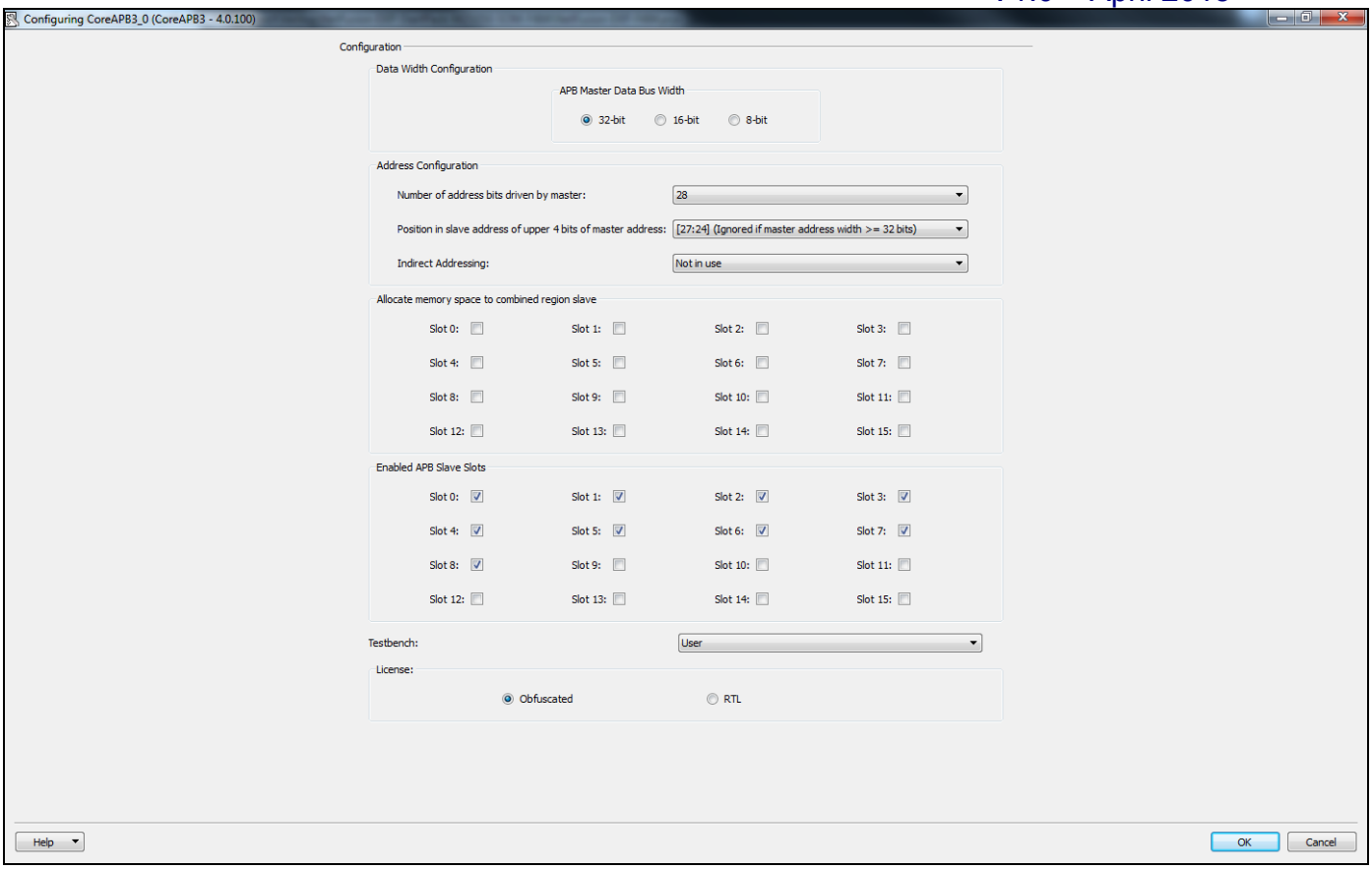


Figure 30 - Default APB Configuration

Important Note: *as a Libero 11.X designer and developer, the user may change these defaults and added or remove memory accessible IP cores to/from the fabric.*

4.20 SmartFusion2 MSS

In the SOM layer sheet of the NetFusion-EXP starter project, the MSS ARM processor main core is situated in the middle and is shown (*below*). If you double-click on the MSS block it will expanded and create another window in Libero 11.X on screen.

Note: *Refer to the next section for the MSS documentation.*

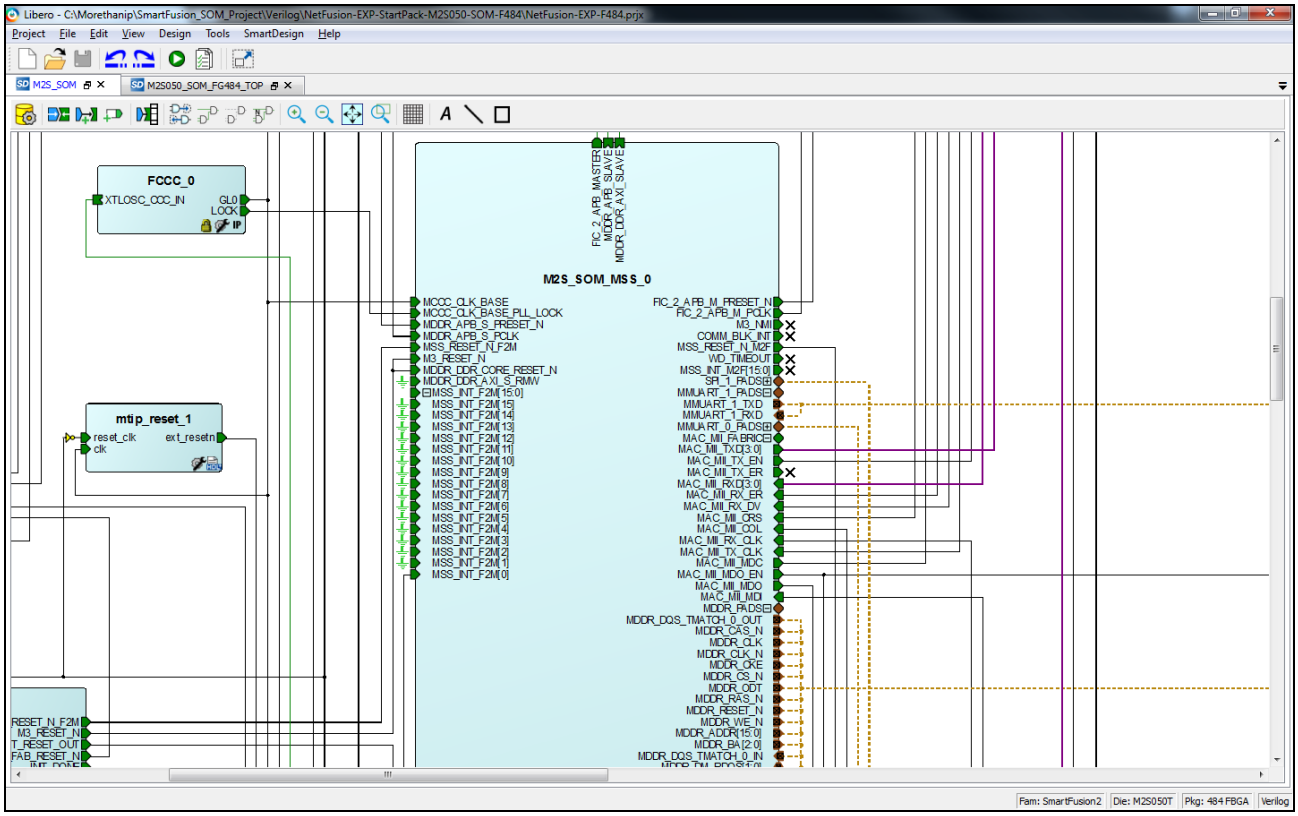


Figure 31 - SOM Module of the ARM MSS

5 NetFusion-EXP ARM Cortex MSS Default Configuration

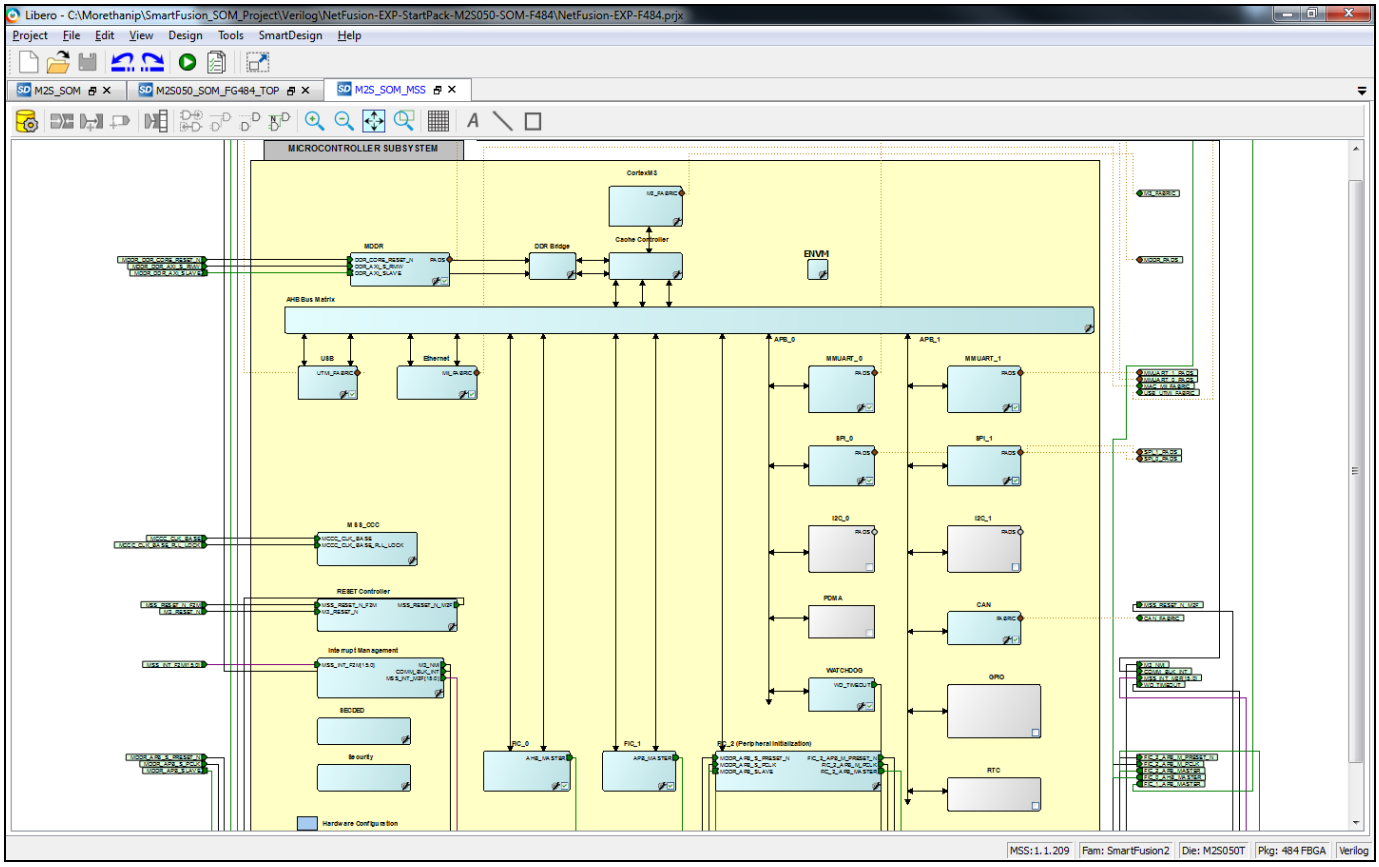


Figure 32 - Exploded View of the Modules in the NetFusion-EXP ARM MSS

The MSS in the heart of the SmartFusion2 NetFusion-EXP design is defaulted and set to the exact current needs of the PCB product. Modules *I2C1, I2C2, GPIO, RTC and PDMA* are disabled currently but there is nothing stopping the user from enabling and wiring out the modules to the SOM sheet.

The blue modules are currently enabled for the NetFusion-EXP design as they have functional requirements and the signals are wired out to the SOM sheet then if necessary up to the I/O assignments then out of the FPGA device itself.

5.1 CAN Controller

The CAN Controller (*shown in the above block*) handles the processing of the data streams in and out of the two transceiver buses. The uCLinux environment is capable of supporting the hardware and is represented as a file descriptor in the typical UNIX way.

5.2 MDDR

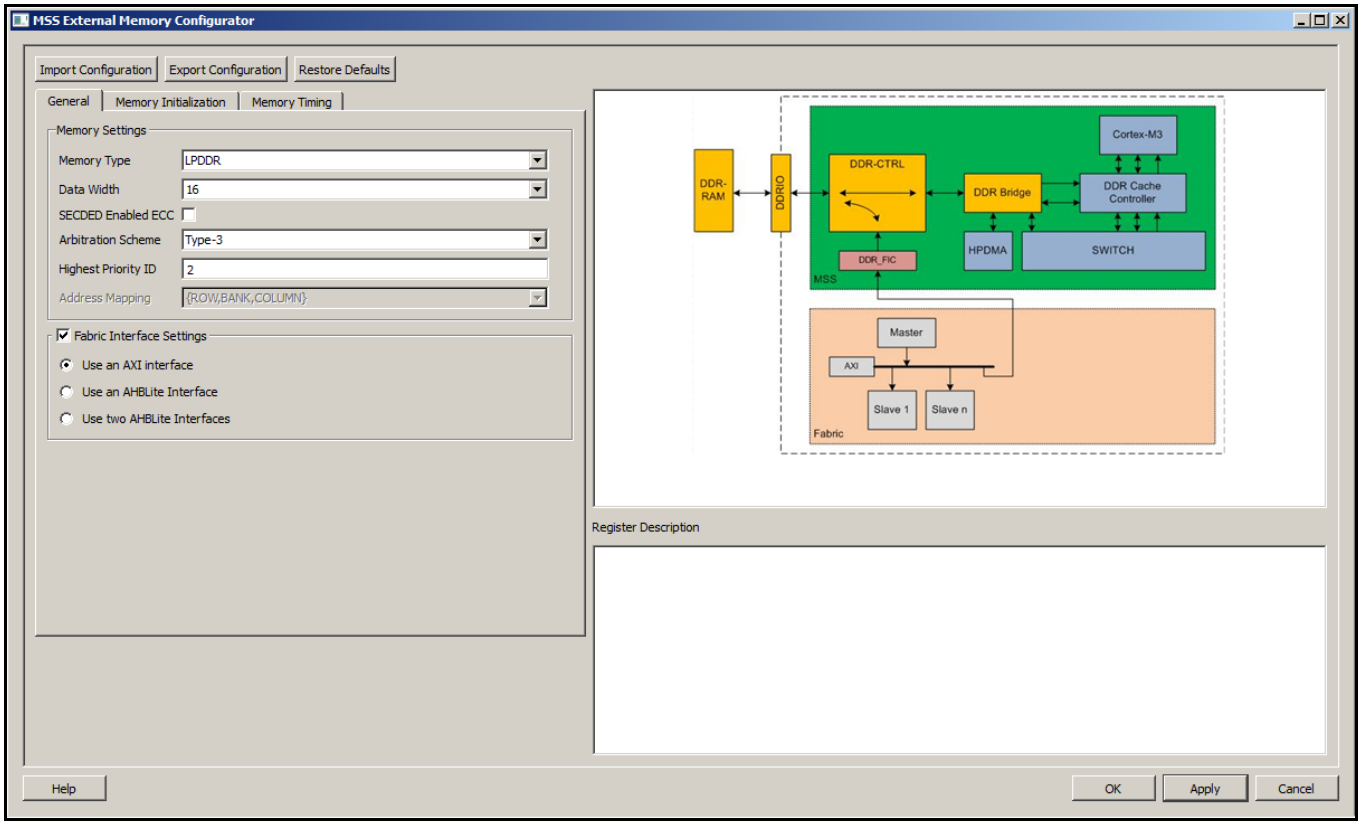


Figure 33 - MDDR MSS Controlling LPDDR Memory with AXI from Fabric

The MDDR has been configured to use a single data rate LPDDR SDRAM device on the M2S-SOM-F484 using 16-bit data width. Priority has been given to the AXI master interface from the fabric where the default connection is to the AMBA DMA Controller for *3rd party MACs and Ethernet SWITCH*.

If you do not wish to keep the DMA Controller or any AXI interface for that matter, then you can de-select the Fabric Interface and the SOM sheet will adjust accordingly. **Important:** *You will have to remove and delete the instantiated DMA Controller however.*

Note: *NetFusion-EXP has to follow the architecture of the SOM-F484 so this is primarily based upon the Emcraft starter project for this block.*

5.3 USB

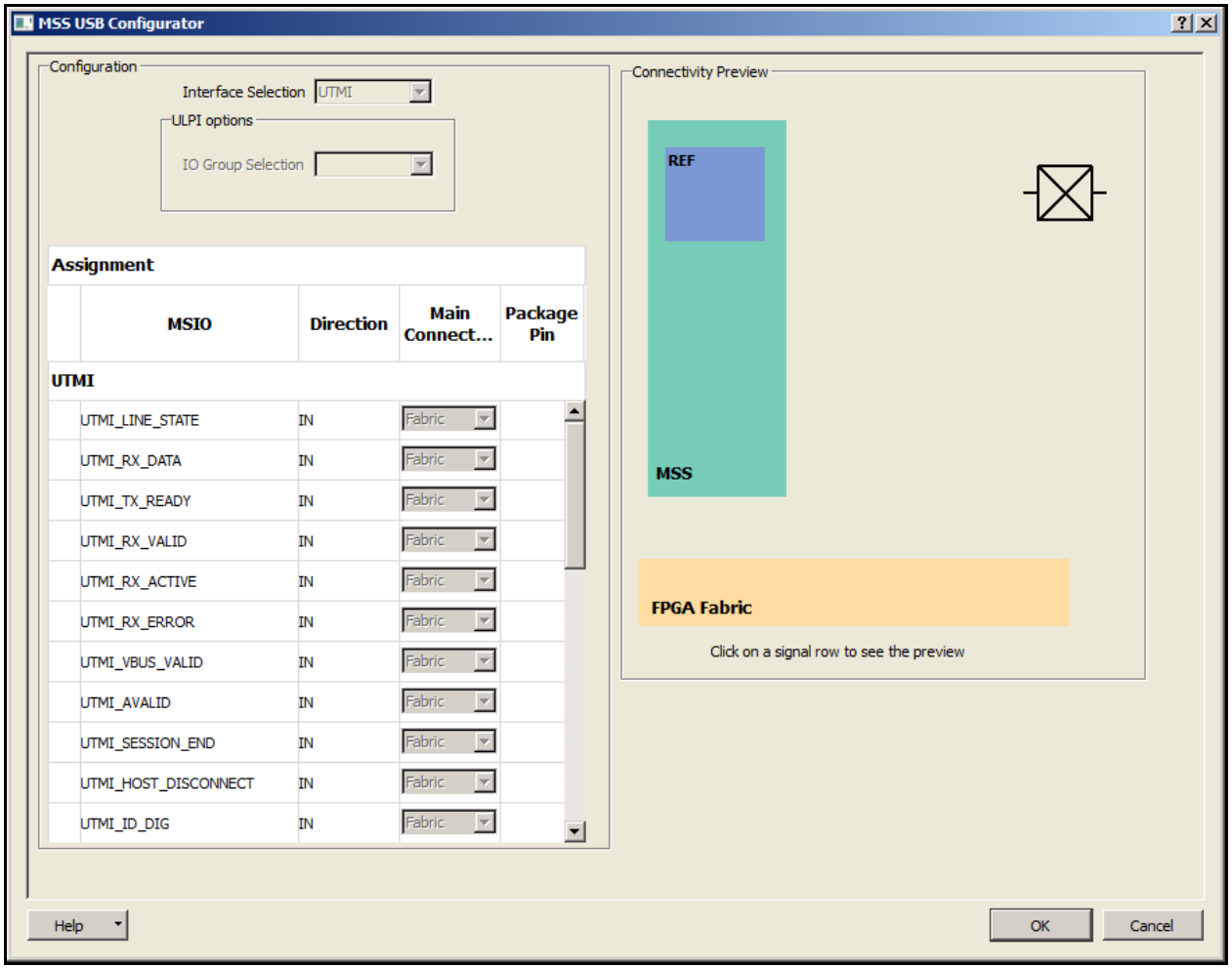


Figure 34 - USB OTG UTMI Host Controller

NetFusion-EXP does utilize on the PCB hardware an *OTG USB* interface. In the fabric, the *ULPI* is converted to UTMI and then connected to this MSS internal block.

The ARM uClinux application code will be able to access this USB block as a block of memory registers and device drivers will be able to control the USB OTG as a USB stack.

Note: *the UTMI signals do not get routed to the I/O pins directly but the fabric for the use of the ULPI/UTMI converter IP core.*

5.4 Ethernet

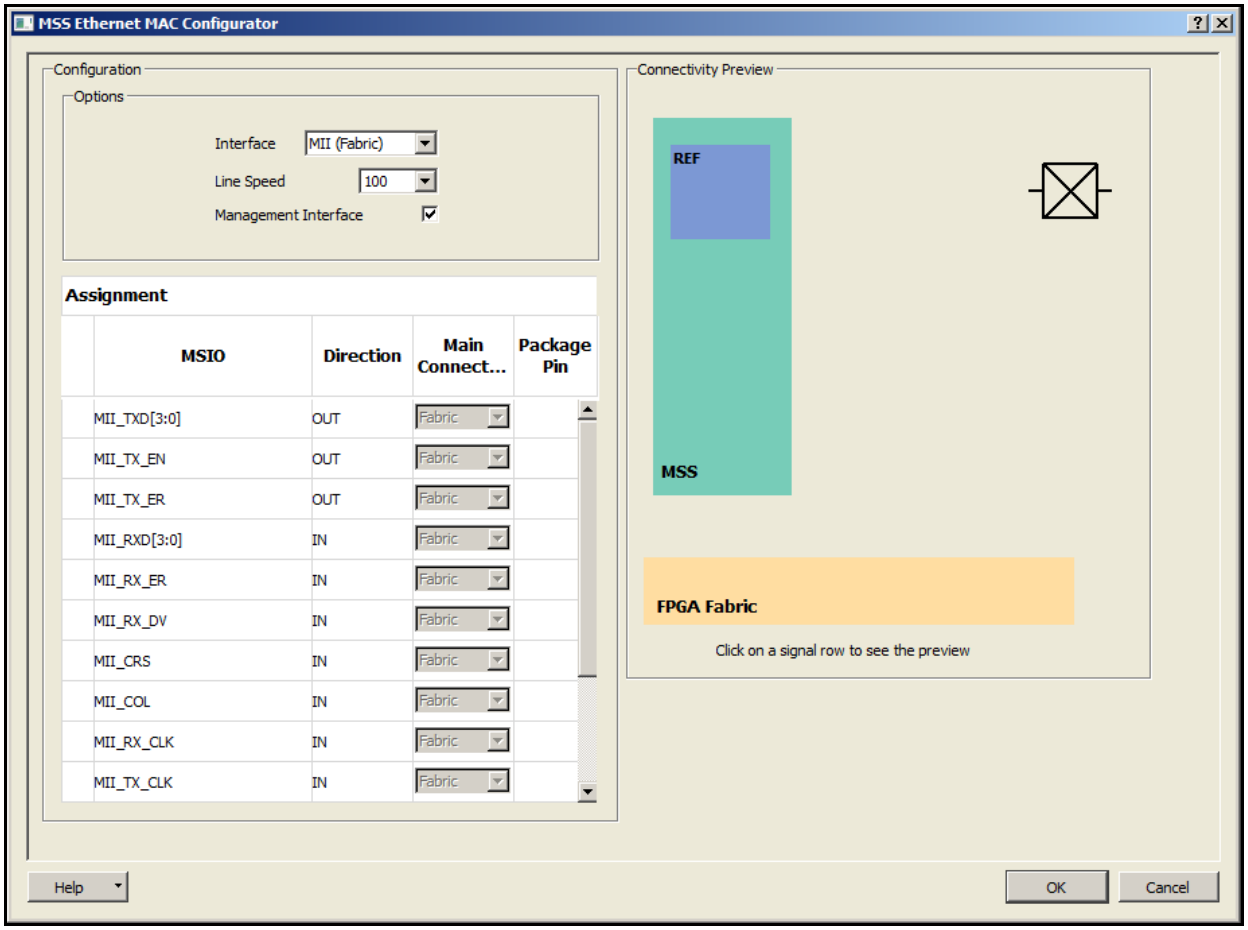


Figure 35 - MSS MAC for 10/100 Ethernet Using SOM PHY

This is a very strategic and important module in the MSS of the SmartFusion2 of NetFusion-EXP. The RJ45 Ethernet connector on the NetFusion-EXP PCB is routed directly to the PHY on the M2S-SOM-F484 which in turn, connects into the SmartFusion2 FPGA and directly into the ARM MSS.

Once in the MSS, the Ethernet MAC receives the connections which then provide a memory interface internally for the ARM uClinux device drivers for the network interfaces.

Note: *The connections route to the fabric before going up through to the I/O assignments.*

It is important to emphasize that with NetFusion-EXP, all of the Daughter Card Expander GPIO that routes into the SOM sheet of the starter project and terminates at [*unused_block_0*].

5.5 MSS_CCC

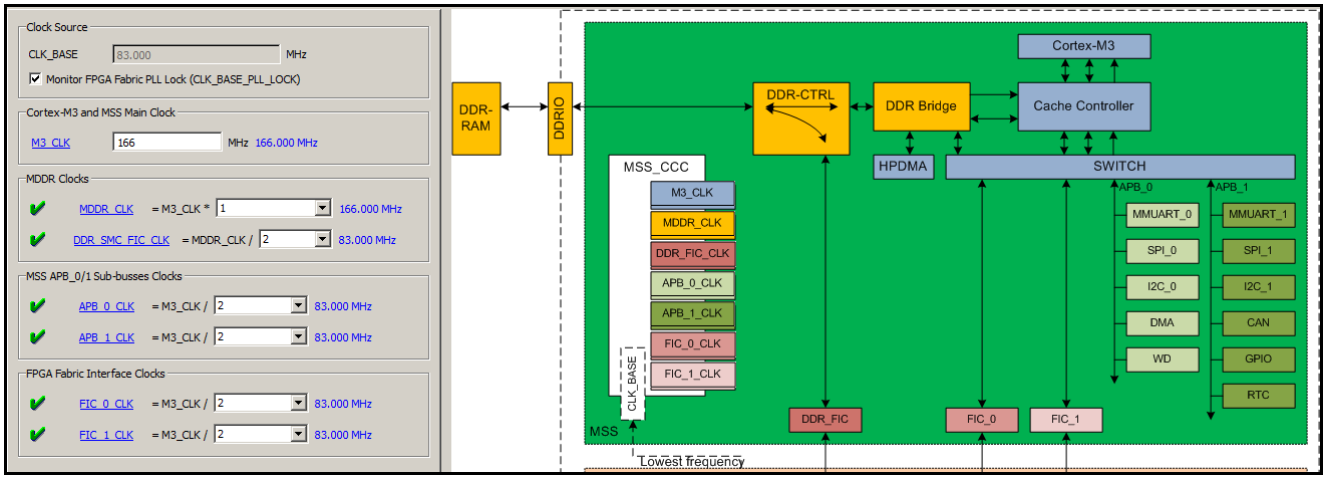


Figure 36 - MSS CCC Divider from the CLK_BASE with-in ARM Sub-System

The main *166MHz* clock into the MSS from the fabric is configured to be split up and if necessary divided down to the different areas of the sub-system. It is derived from the **Div 2** *83MHz* from the PLL in the fabric.

In the case of NetFusion-EXP and the Starter Pack Libero 11.X IDE project, all peripherals, memory controllers and fabric interfaces run at half the base clock frequency at *83MHz*.

5.6 RESET Controller

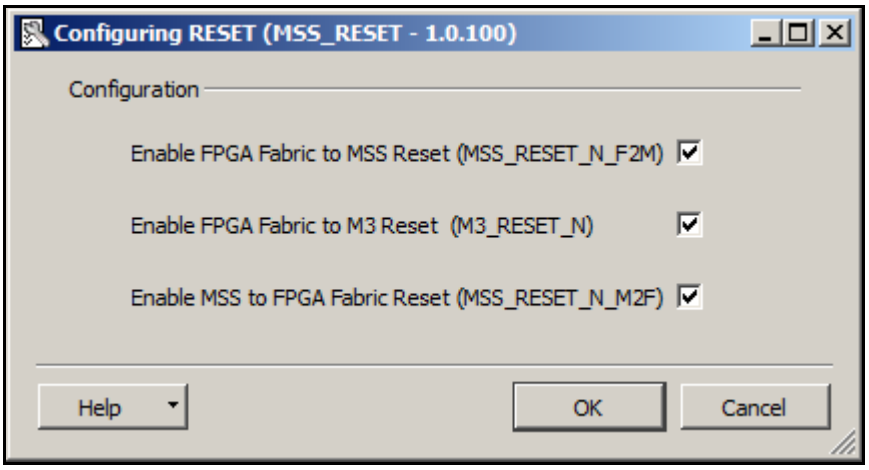


Figure 37 - Reset MSS Module

NetFusion-EXP by default enables the **MSS_RESET_N_F2M**, **M3_RESET_N** and the **MSS_RESET_N_M2F** negative reset signals to come in from the SmartFusion2 FPGA fabric.

This gives more functionality to the user. However, if it is not a required functionality when the design is customized, then they can be de-selected.

5.7 FIC_0

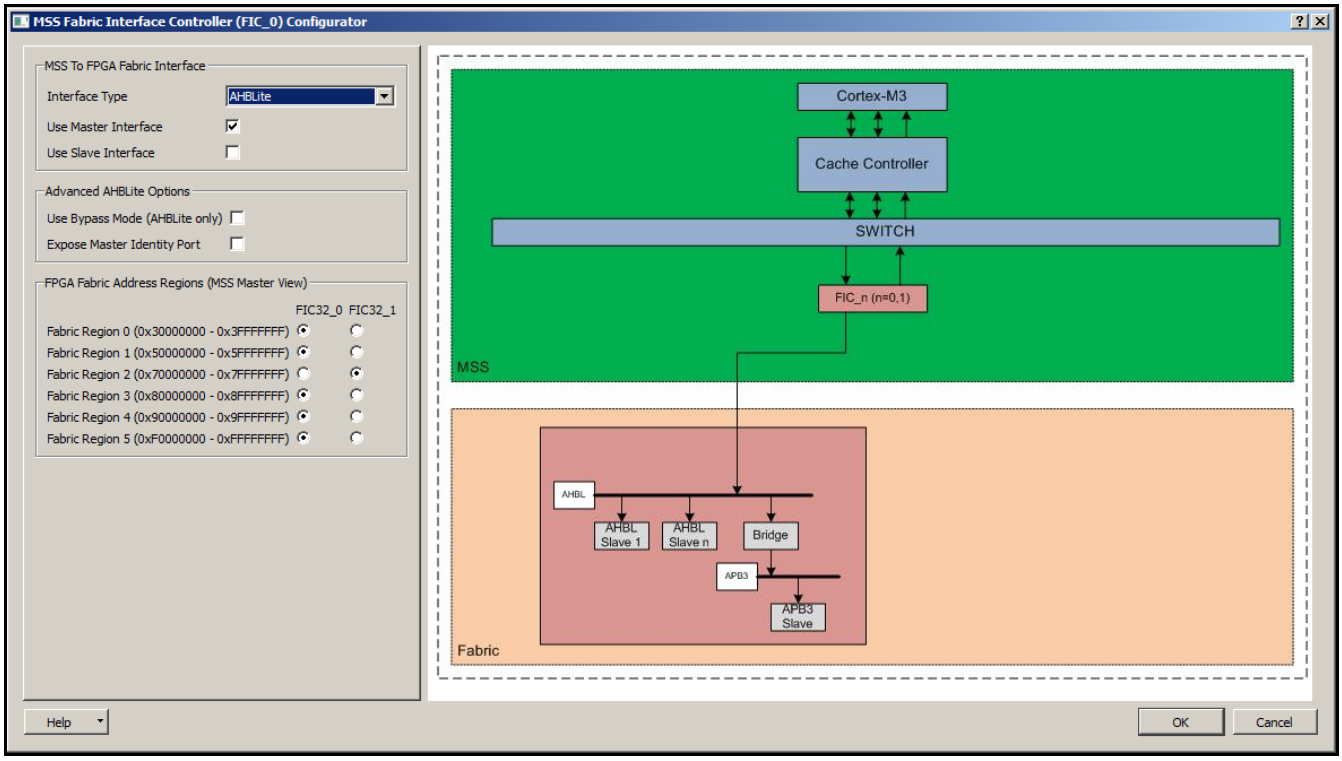


Figure 38 - AHB/APB Fabric Interface [1]

The NetFusion-EXP **FIC_0** has been chosen to be assigned for the AHB-Lite fabric interface. It is a *MASTER* which connects to *the SLAVES* in the fabric as the ARM processor has complete control.

By default, the AHB-Lite only interfaces to the AMBA DMA Controller used for 3rd party Ethernet IP cores. Although the DMA Controller uses *AXI* for the main data throughput, the AHB-Lite is used to access the configuration registers.

Note: *you will observe that Fabric Region 2: 0x7000000 - 0x7FFFFFF has been allocated to the next* ***FIC_1*** *block (next sub-section). The configuration on the left panel (above) selects mapping for both* ***FIC_0*** *and* ***FIC_1****.*

NetFusion-EXP assigns Fabric Region 2 to the APB memory interface so that the *UART, SPI* and *GPIO* can be accessed in the fabric otherwise only this AHB-Lite interface would be mapped which would severely limit the NetFusion-EXP functionality and capability.

5.8 FIC_1

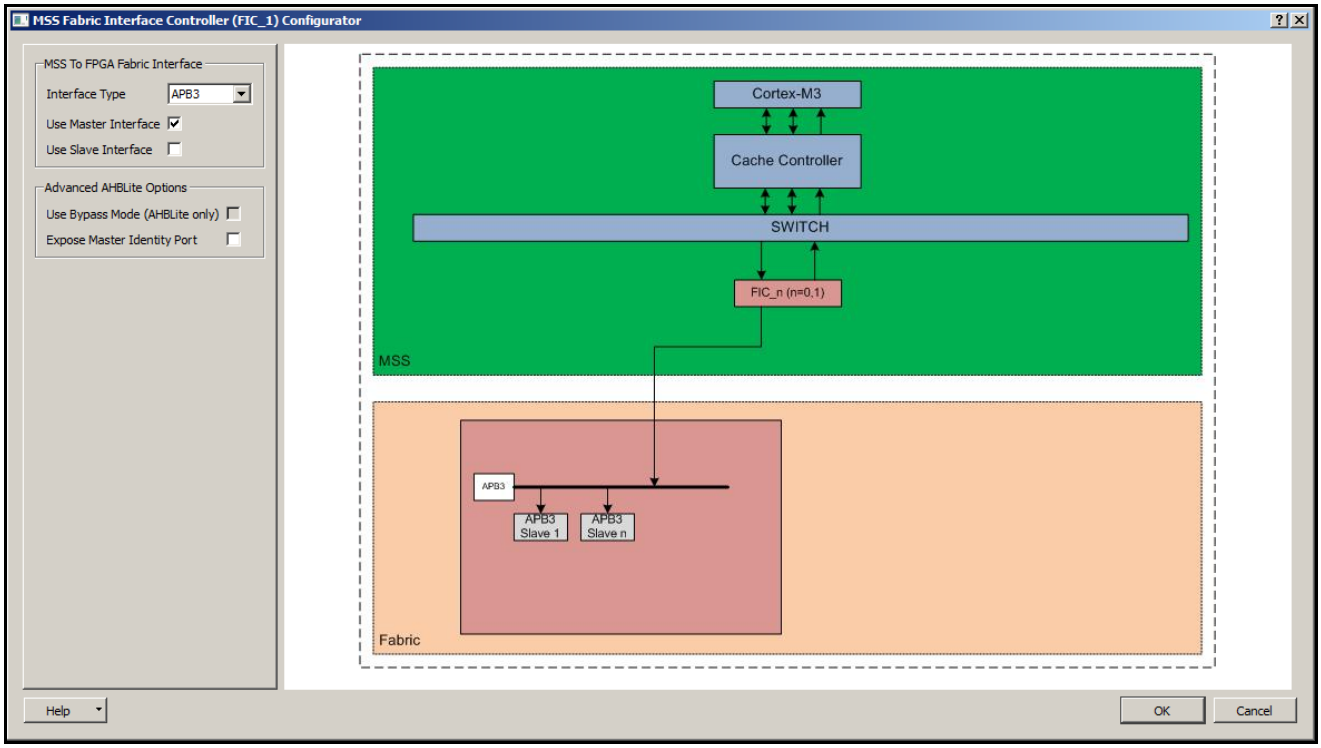


Figure 39 - APB3/AHB-Lite Fabric Interface [2]

Using the configuration from the previous **FIC_0** block *(previous*), this MSS module enables the *APB* interface to access the *GPIO, SPI* and *UART* in the NetFusion-EXP SmartFusion2 fabric.

Note: *this is also an APB Master as the ARM processor has complete control.*

5.9 FIC_2 (Peripheral Initialization)

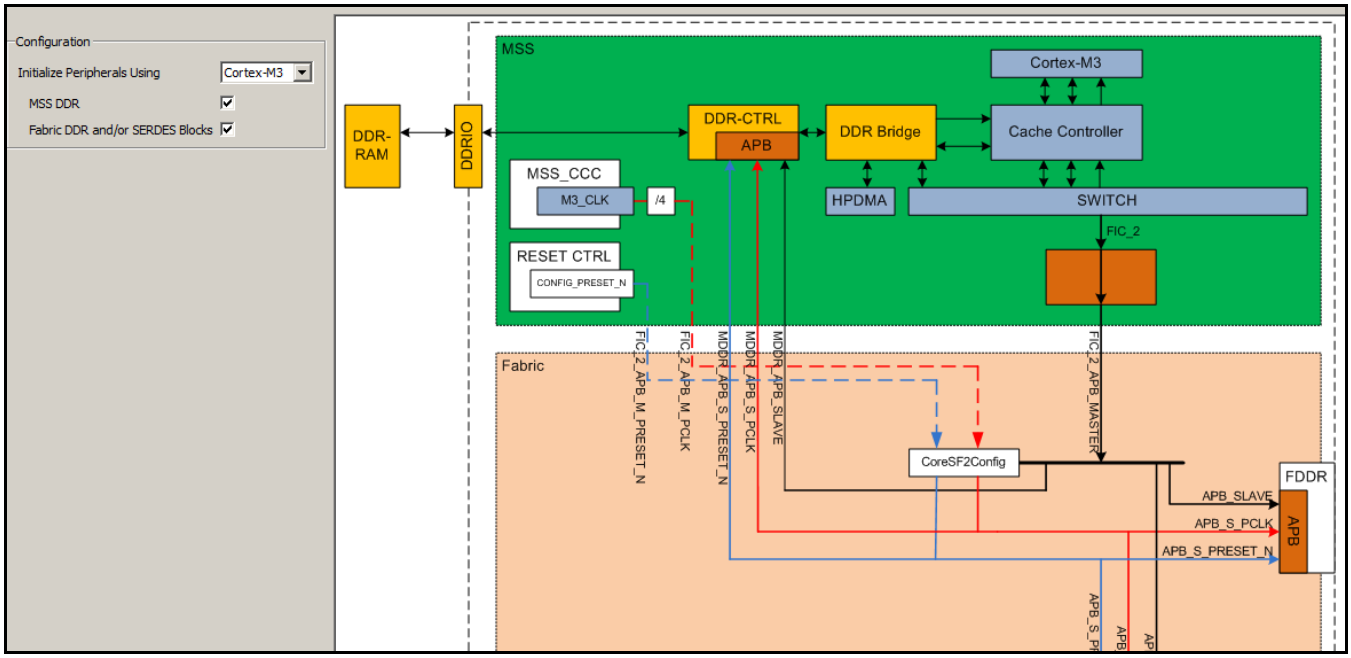


Figure 40 - APB Peripheral Hardware Configuration

As described earlier in this document, the SmartFusion2 FPGA is setup and primarily configured by boot-up software executed by the ARM processor. This can be bare-metal code or early boot code from uClinux called *u-boot*.

This MSS module block enables the signals to connect a feedback *APB* interface from the ARM processor to the other peripherals in the *MSS* itself. It seems overkill but that is the architecture we are ruled by it seems.

In the case of NetFusion-EXP, we configure the **FIC_0** and **FIC_1** fabric interface controllers and also select the AXI Slave interface for the MDDR block.

Important Note: *do not ever remove this feature from the design of NetFusion-EXP. The system will not operate if deleted.*

6 Adding IP Cores from Verilog/VHDL

6.1 Importing Source Files

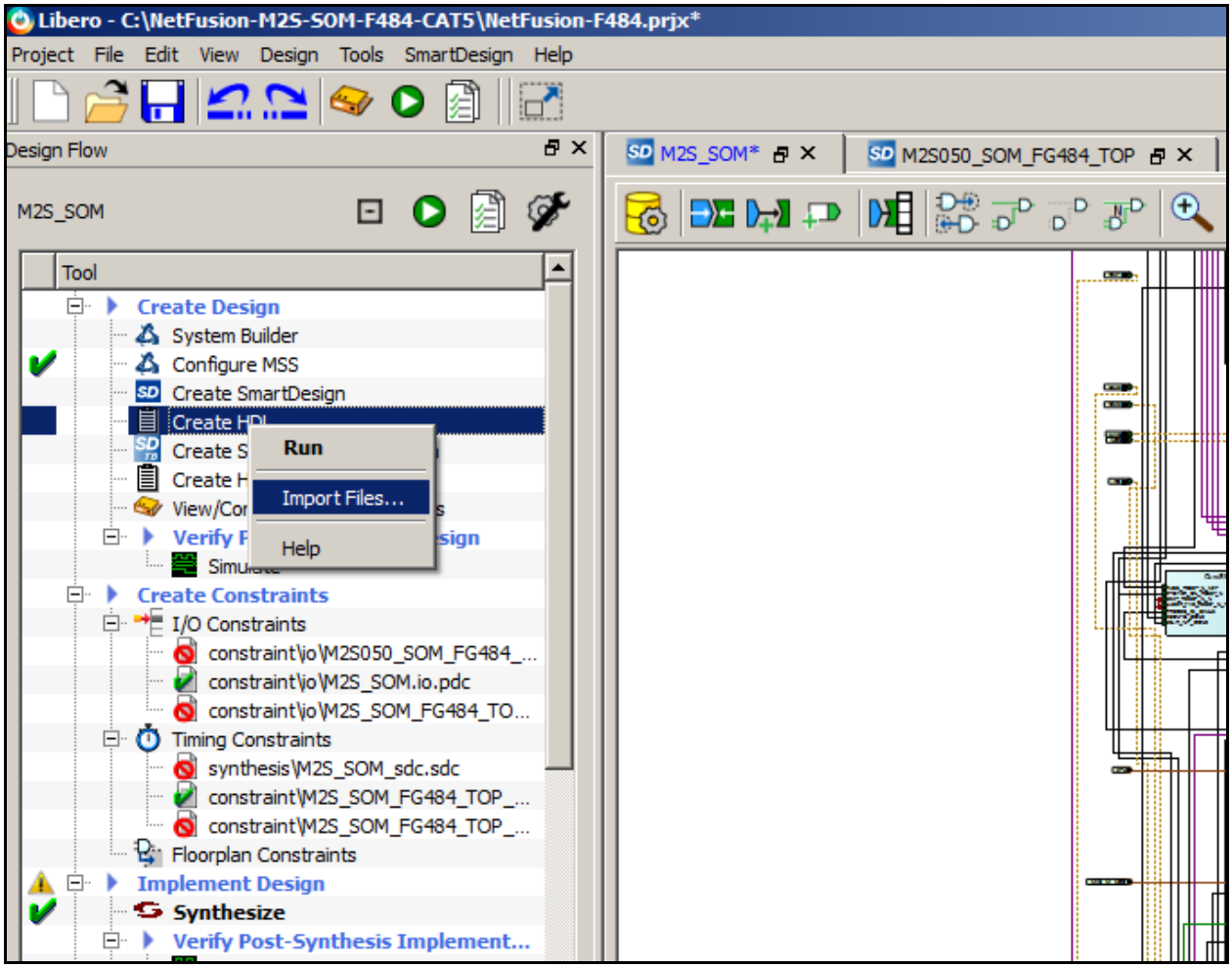


Figure 41 - Importing a single Verilog/VHDL file into the NetFusion-EXP Libero Project

From time to time, you may want to deviate from the default NetFusion-EXP Libero 11.X Starter Project. If you need to add catalogue cores especially for the fabric macros, then download using the vault.

Also, as illustrated *(above*), you also may wish to import 3rd party source IP cores. To do this, on the panel on the left of the Libero IDE, right click on "*Create HDL*" and then select "*Import Files...".* This will bring the Verilog/VHDL into the main system.

You can check if the source code has syntax errors and is valid for the synthesizer (Libero 11.XSynplify Pro) that Libero 11.X uses. This mechanism enables the NetFusion-EXP project to be customized and built on for a user's system requirement.

6.2 Instantiating into the Fabric

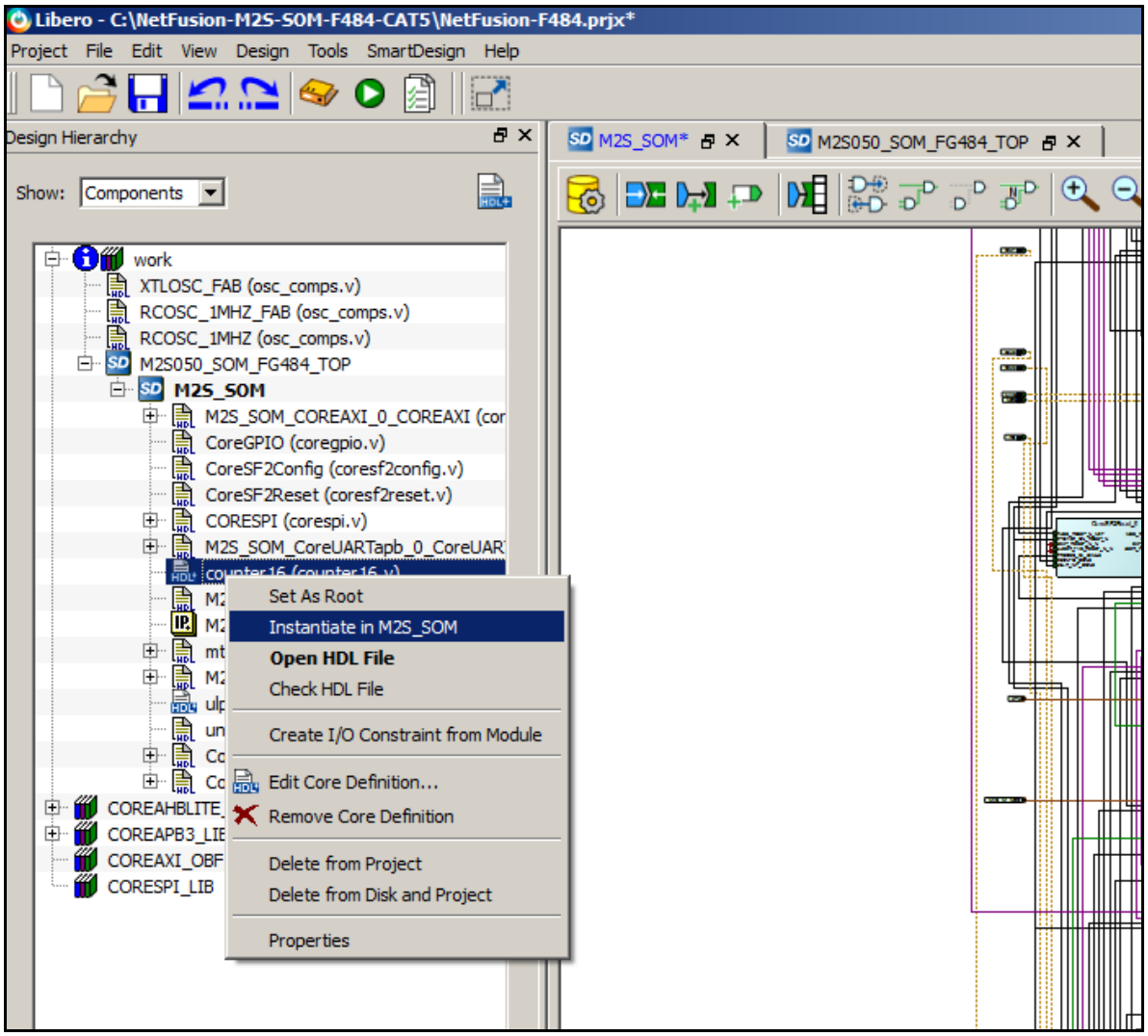


Figure 42 - Instantiating an Imported IP Core in your Fabric Design

Whether you wish to instantiate into the SmartFusion2 fabric an imported 3rd party source code core from VHDL/Verilog or it is from the vault downloaded from ACTEL, the process to get the core into the design sheets is the same and relatively simple. Even if the core is a macro as part of the ASIC area of the FPGA, the process is the same no matter what area of the design it involves. Simply move the left panel to select "*Design Hierarchy*" and then right click on the listed core of your choice. Select *"Instantiate in M2S_SOM"* or whichever sheet is displayed on the right pane of Libero IDE.

The core will appear in the design for you to move and anchor ready for connection routing.

Note: *If the core has errors, the error report page will appear and the instantiation will not occur.*

6.2.1 Building and Synthesizing the NetFusion-EXP Design

- *Open the Top-Level, SOM and the MSS sheets*
- *Start with the MSS and right click "Generate Component"*
- *If errors eliminate then perform previous instruction over again*
- *Once successful, move to the SOM sheet*
- *Right click "Generate Component"*
- *If errors eliminate then perform previous instruction over again*
- *Once successful, move to the Top-Level sheet*
- *Right click "Generate Component"*
- *If errors eliminate then perform previous instruction over again*

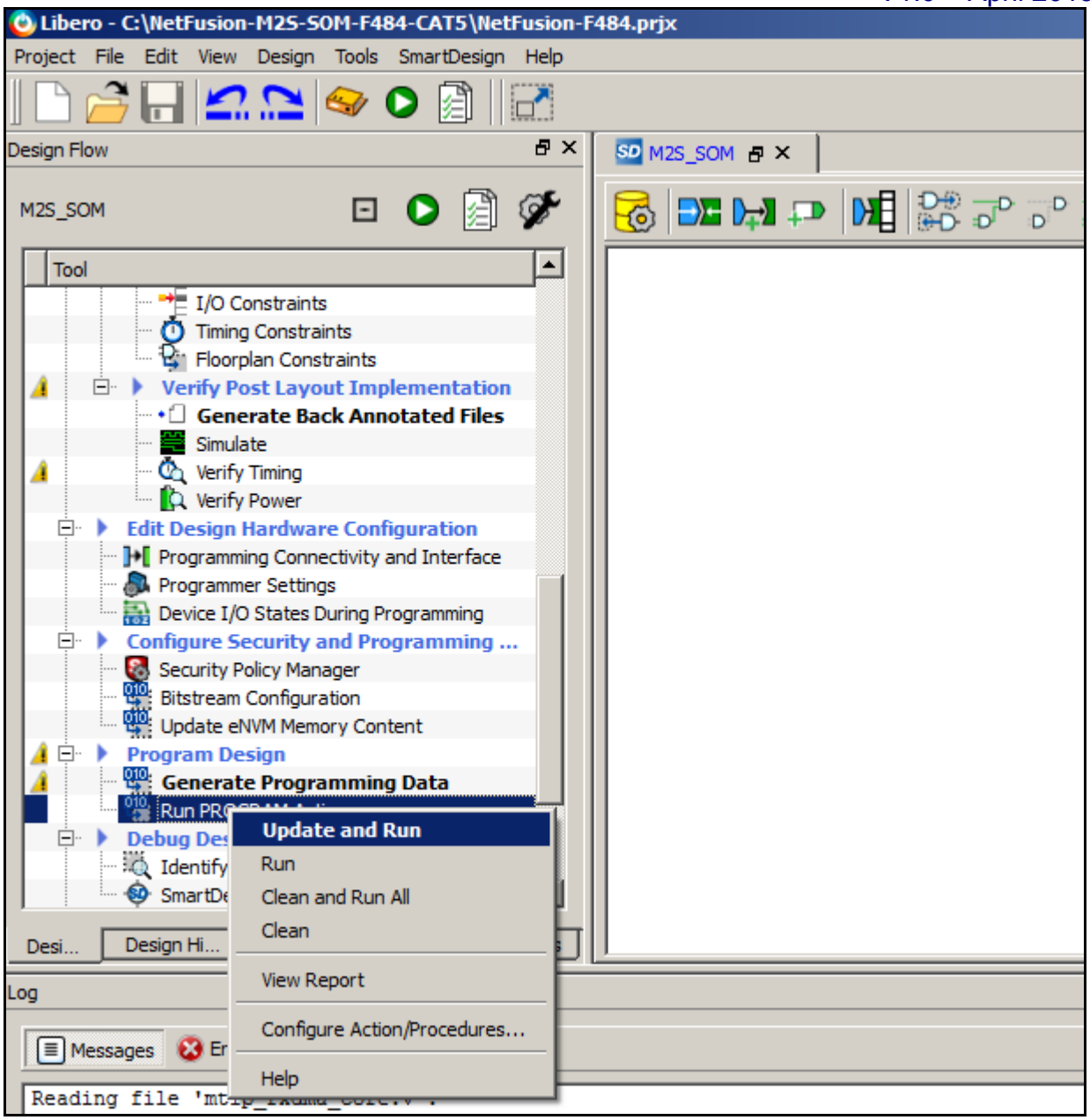


Figure 43 - Selecting Full NetFusion-EXP Synthesis and FPGA Programming

Once you are ready to synthesize and program the SmartFusion2 FPGA on the M2S-SOM-F484 housed on NetFusion-EXP, select the option *(shown above*).

Note: *This should take around 20 minutes on a standard Windows XP/7/8 PC. However, larger customized design will add significant time especially if the design has strict time-constraints.*

7 Using FlashPro4 IDE to Program NetFusion-EXP

Install and make use of *FlashPro IDE* on Windows XP/7/8 OS to allow for the stand-alone programming of the baseboard without the need for launching, installing or using the full **Libero 11.X SoC IDE**.

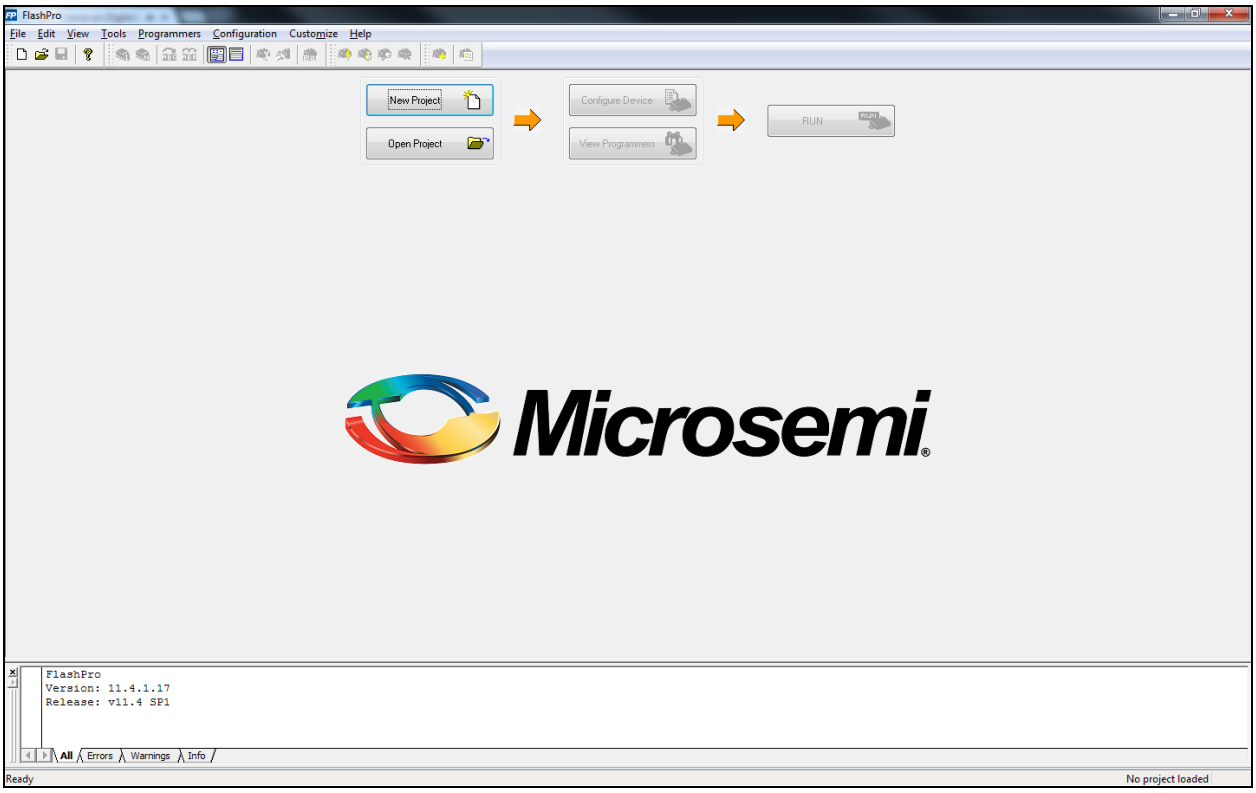


Figure 44 - Opening screen of FlashPro IDE

Launch the IDE so that you see the (*above*) screen. Also, if you have just installed the IDE, ensure that you have also installed the *FlashPro4 USB* device driver in the **Windows Device Manager**.

Note: *the device driver should be automatically installed with the FlashPro installation (which is downloadable from the Microsemi website).*

Ensure that the NetFusion-EXP PCB has the ribbon cable attached from the FlashPro4 USB programmer (*below*)

Figure 45 - Microsemi FlashPro4 ribbon connection

On the FlashPro IDE screen, click on **New Project**.
Enter any project name (*and make sure you have write permissions to the selected directory you choose*).

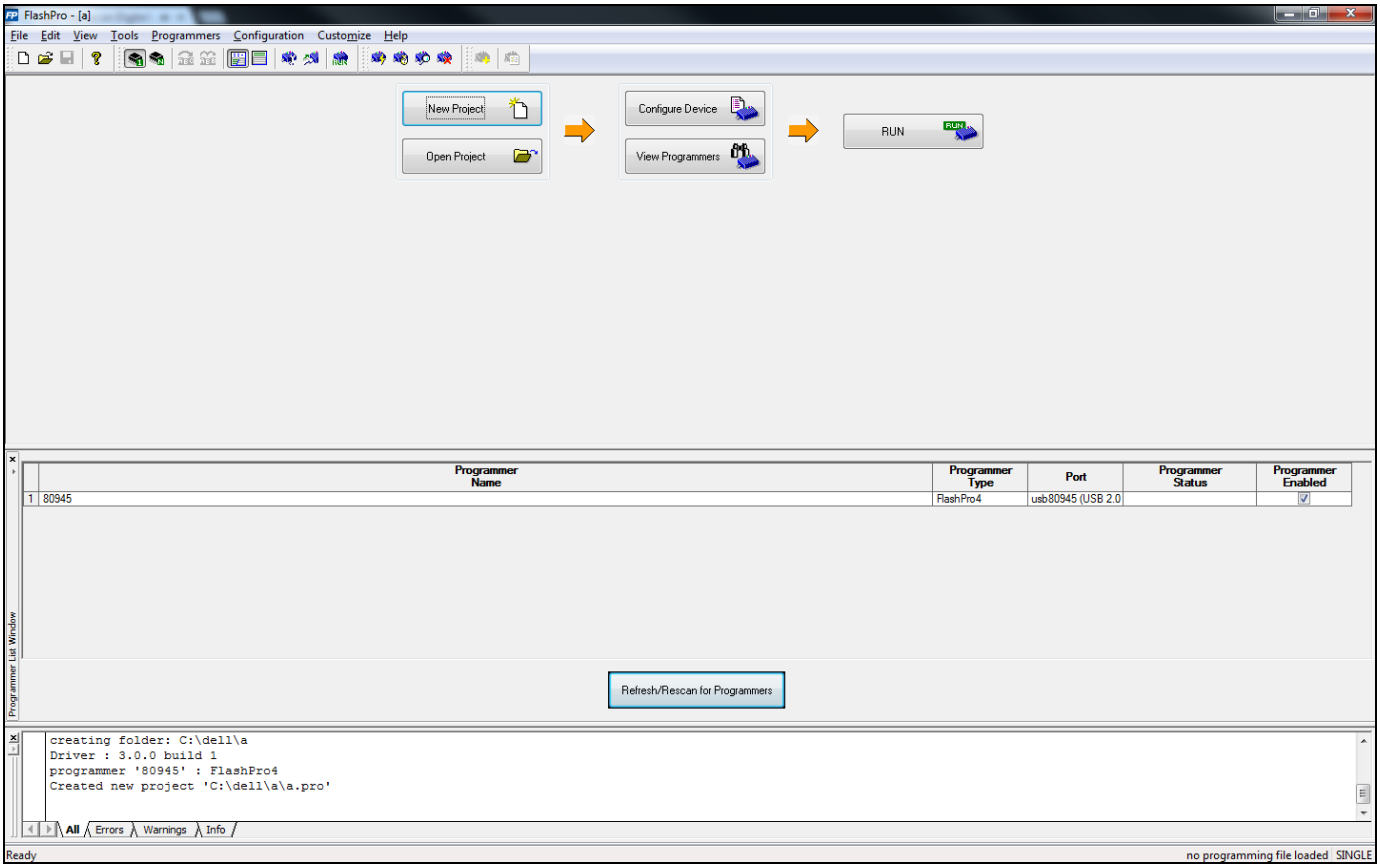


Figure 46 - FlashPro Programming Screen

In the drop-down menus, choose **Configuration**, then select **Load Programming File…**

Chose your **[.STP]** file to program NetFusion-EXP.

Click on the main **PROGRAM** button center right of the (*above*) screen.

Wait for the programming to complete typically after several minutes…

If you cannot select the USB programmer correctly, check it is plugged in and also click on "Refresh/Rescan for Programmers" button towards the bottom of the screen (*above*).

Important Note: *once the operation to program NetFusion-EXP is complete, the SmartFusion2 FPGA will re-power and start automatically.*

8 References

Please refer to online documented support at the Microsemi reference center.

For M2S-SOM-F484 hardware documents please visit the Emcraft hardware for the SOM-F484.

If you need Libero 11.X references from Emcraft, this will illustrate the default designs that NetFusion-EXP was built from in order to achieve the baseboard functionality.

9 Contact

Nine Ways Research & Development Ltd	E-Mail : **pbates@nineways.co.uk** Internet : **www.nineways.co.uk** **UK** Unit A.3, iDCentre, Lathkill House, rtc Business Park London Road, Derby. DE24 8UP United Kingdom Tel : **+44 (0) 1332 258847** FAX : **+44 (0) 1332 258823**

10 Document History

Document Change Notices (DCO)

Version	Description	Created/Changed By	Date
Version 1.0	Initial Release according to Version 1.0	Paul Bates: Nine Ways	**23rd April 2015**