Power Matters.TM

Microsemi

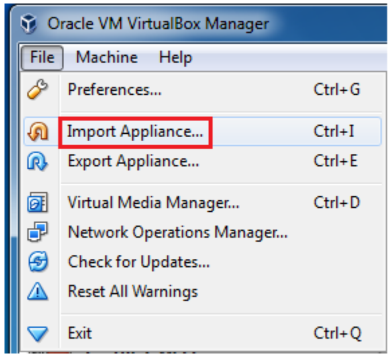
RISC-V Workshop

July 2016

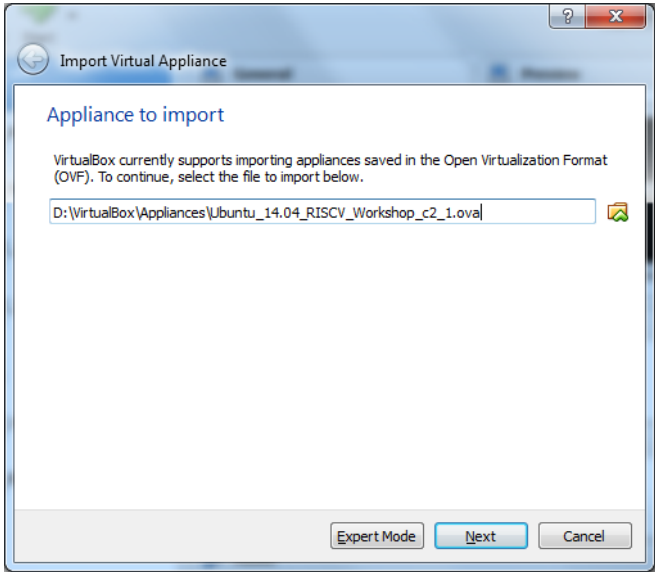
Virtual Machine Setup

Virtual Machine Setup

- VirtualBox virtual machine available from USB drive or download
- Import appliance:
  - File->Import Appliance…



Oracle VM VirtualBox Manager
File
Machine
Help
Preferences... Ctrl+G
Import Appliance... Ctrl+I
Export Appliance... Ctrl+E
Virtual Media Manager... Ctrl+D
Network Operations Manager...
Check for Updates...
Reset All Warnings
Exit Ctrl+Q



VirtualBox currently supports importing appliances saved in the Open Virtualization Format (OVF). To continue, select the file to import below.

D:\VirtualBox\Appliances\Ubuntu_14.04_RISCV_Workshop_c2_1.ova

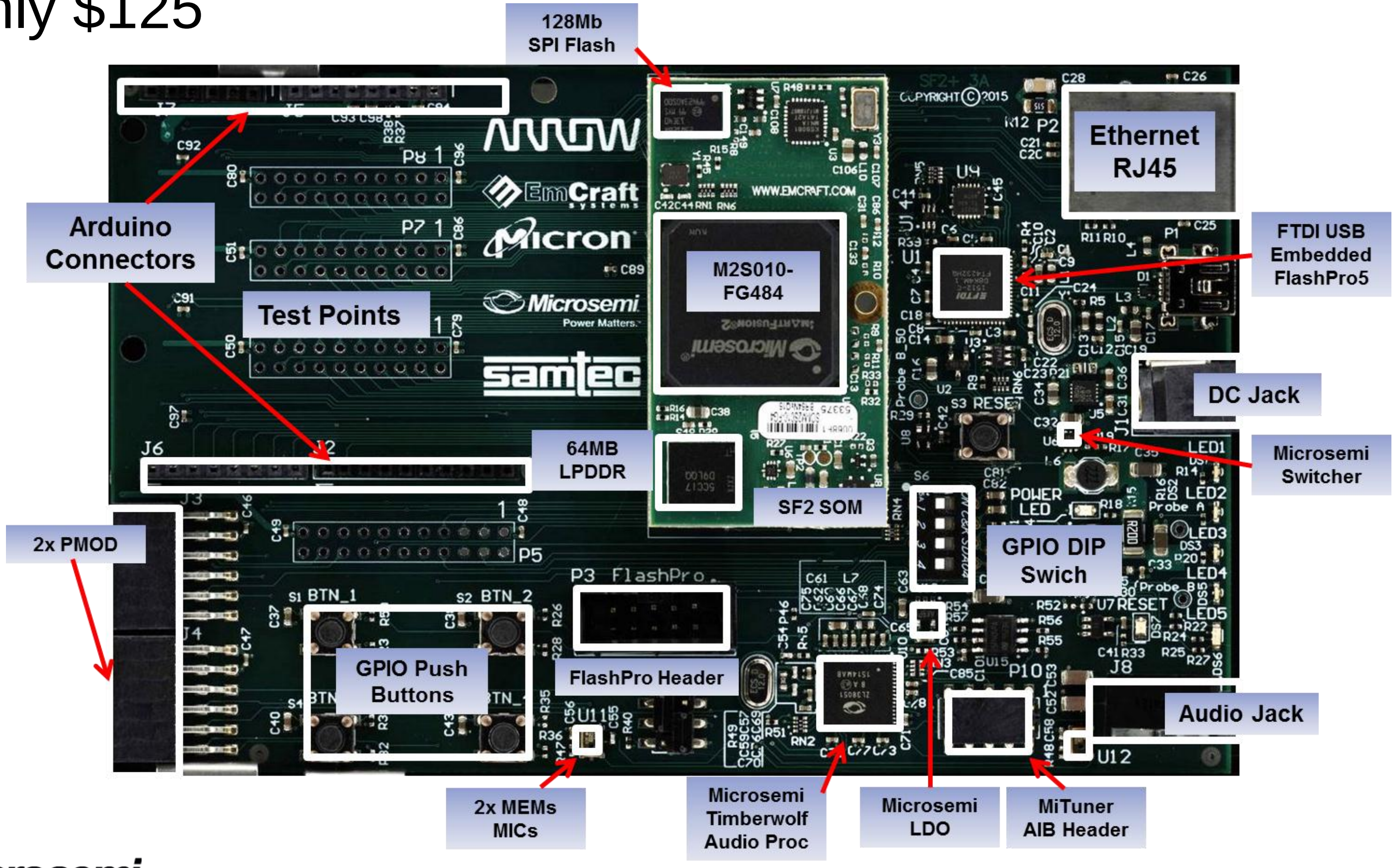
Retrieve Tutorial Material

- Retrieve the tutorial material from within the virtual machine
- Download tutorial material from github:
  - $ cd ~
  - $ git clone https://github.com/riscv/riscv-4th-workshop-tutorials.git
- Update the material if you already downloaded:
  - $ cd ~/riscv-4th-workshop-tutorials
  - $ git pull

Development Board

Arrow SF2+ Kit

- Features SF2-010 & Timberwolf audio processor
- Example designs included
- Only $125

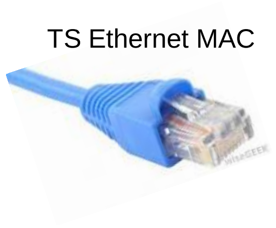


IGLOO2 FPGAs & SmartFusion2 SoCs

More resources in low density devices

with the lowest power, proven security and exceptional reliability

TS Ethernet MAC



Secure Flash

DDR3 Controller

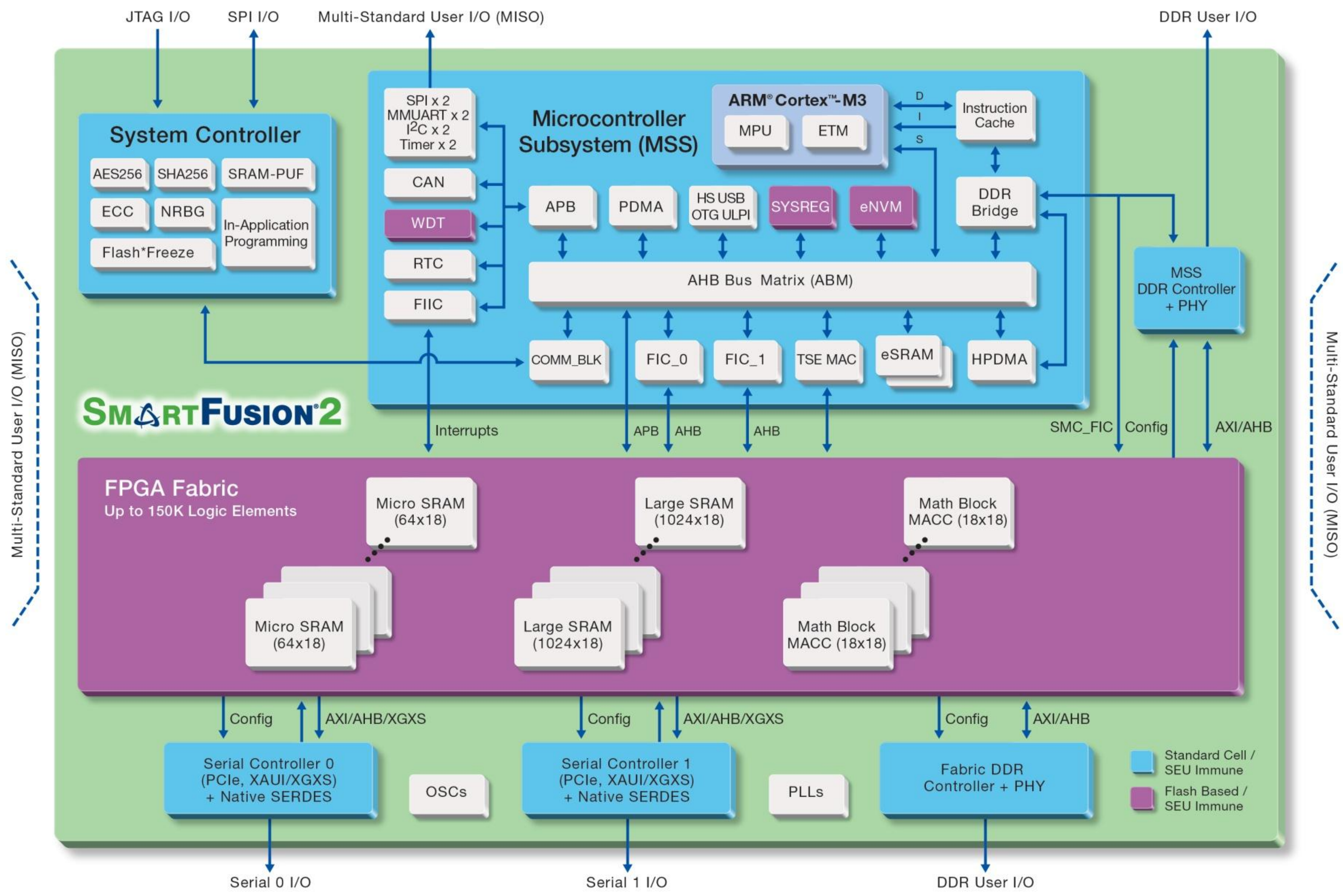
Low Power

Security

Reliability

SmartFusion2 SoC FPGAs

SmartFusion2 Architecture



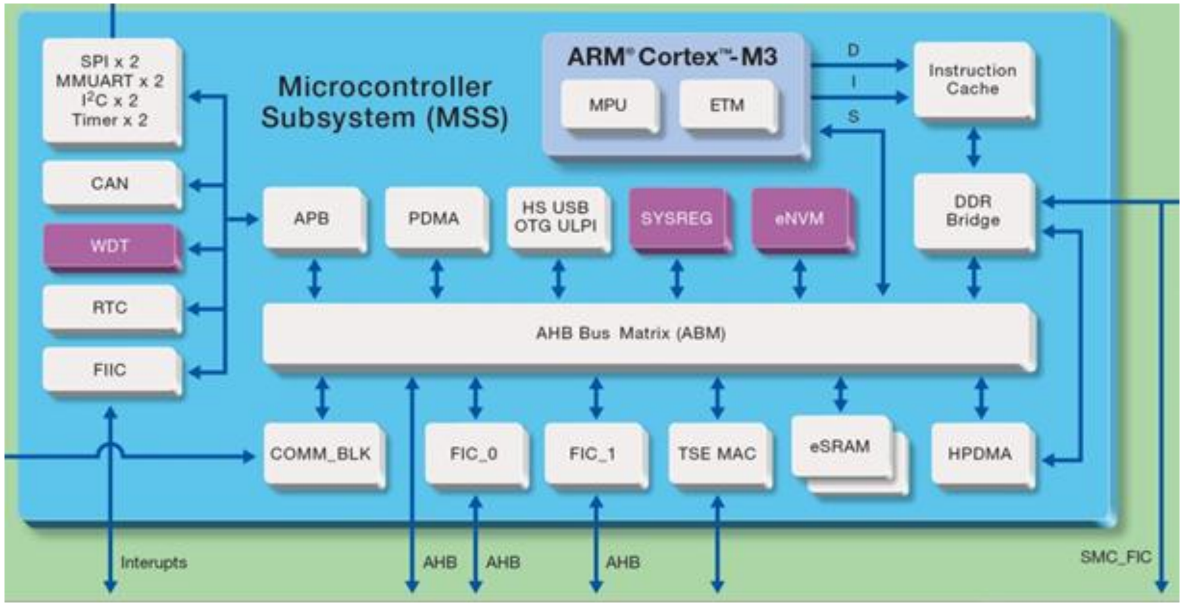
SmartFusion2 Product Family

	Features	M2S005	M2S010	M2S025	M2S050	M2S060	M2S090	M2S150
Logic/DSP	Maximum Logic Elements (4LUT + DFF)[1]	6,060	12,084	27,696	56,340	56,520	86,184	146,124
	Math Blocks (18x18)	11	22	34	72	72	84	240
	Fabric Interface Controllers (FICs)	1			2	1		2
	PLLs and CCCs	2		6				8
	Data Security	AES256, SHA256, RNG				AES256, SHA256, RNG, ECC, PUF		
MSS	Cortex-M3 + Instruction cache	Yes						
	eNVM (K Bytes)	128	256				512	
	eSRAM (K Bytes)	64						
	eSRAM (K Bytes) Non SECDED	80						
	CAN, 10/100/1000 Ethernet, HS USB	1 each						
	Multi-Mode UART, SPI, I2C, Timer	2 each						
Fabric Memory	LSRAM 18K Blocks	10	21	31	69	69	109	236
	uSRAM1K Blocks	11	22	34	72	72	112	240
	Total RAM (K bits)	191	400	592	1314	1314	2074	4488
High Speed	DDR Controllers (Count x Width)	1x18			2x36	1x18	1x18	2x36
	SERDES Lanes (T)	0	4		8	4	4	16
	PCIe End Points	0	1		2			4
User I/Os	MSIO (3.3V)	115	123	157	139	271	309	292
	MSIOD (2.5V)	28	40	40	62	40	40	106
	DDRIO (2.5V)	66	70	70	176	76	76	176
	Total User I/O	209	233	267	377	387	425	574

- Total logic may vary based on utilization of DSP and memories in your design. Please see the IGLOO2 Fabric UG for details
- Feature availability is package dependent

Microsemi

Microcontroller Subsystem

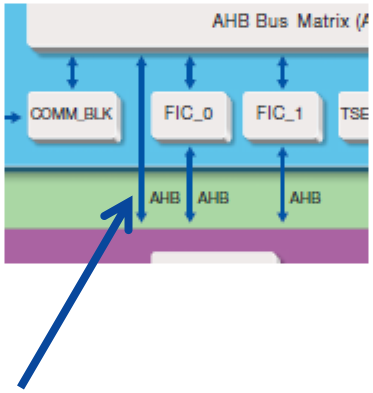


Equivalent to > 70K Luts

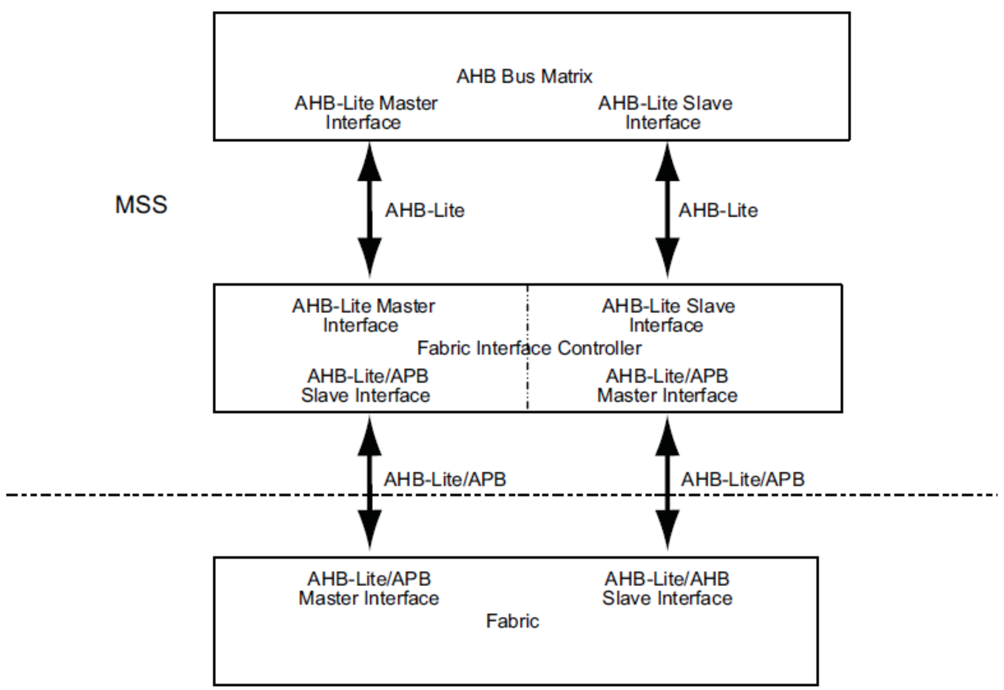
	Features	M2S005	M2S010	M2S025	M2S050	M2S060	M2S090	M2S150
MSS	Cortex-M3 + Instruction cache	Yes						
	eNVM (K Bytes)	128	256				512	
	eSRAM (K Bytes)	64						
	eSRAM (K Bytes) Non SECDED	80						
	CAN, 10/100/1000 Ethernet, HS USB	1 each						
	Multi-Mode UART, SPI, I2C, Timer	2 each						

Fabric Interface Controller (FIC)

1 FIC on the 05 through 025
2 FIC's on the 050 through the 150



Dedicated APB configuration bus in all devices

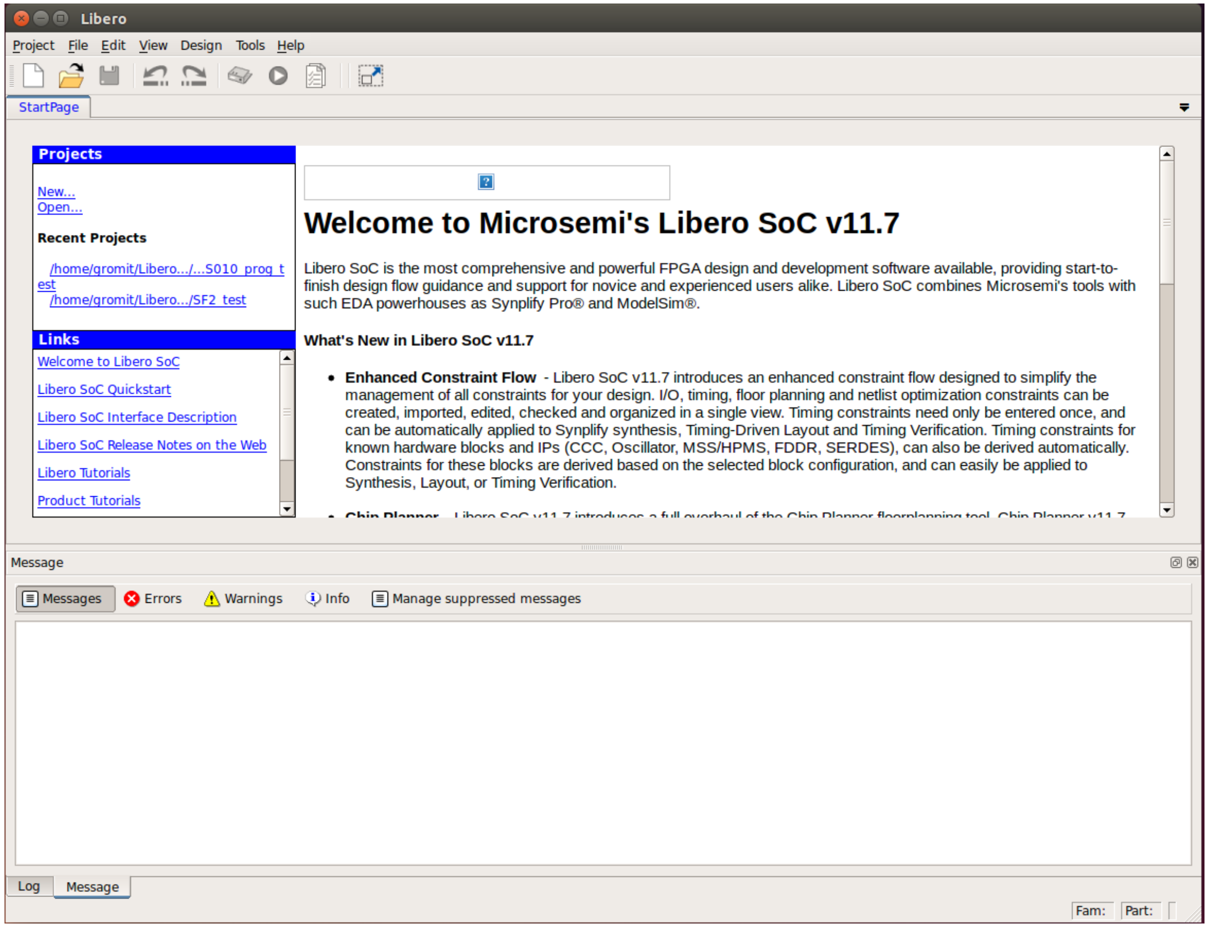

AHB Bus Matrix
AHB-Lite Master Interface
AHB-Lite Slave Interface
MSS
AHB-Lite
AHB-Lite
AHB-Lite Master Interface
AHB-Lite Slave Interface
Fabric Interface Controller
AHB-Lite/APB Slave Interface
AHB-Lite/APB Master Interface
AHB-Lite/APB
AHB-Lite/APB
AHB-Lite/APB Master Interface
AHB-Lite/AHB Slave Interface
Fabric

1 FIC

Microsemi Libero

Libero

- Start Libero using the *start_libero.sh* script located in the home directory


Libero
Project File Edit View Design Tools Help
StartPage
Projects
New...
Open...
Recent Projects
/home/gromit/Libero.../...S010_prog_test
/home/gromit/Libero.../SF2_test
Links
Welcome to Libero SoC
Libero SoC Quickstart
Libero SoC Interface Description
Libero SoC Release Notes on the Web
Libero Tutorials
Product Tutorials
Message
Messages
Errors
Warnings
Info
Manage suppressed messages
Log
Message
Fam:
Part:

Welcome to Microsemi's Libero SoC v11.7

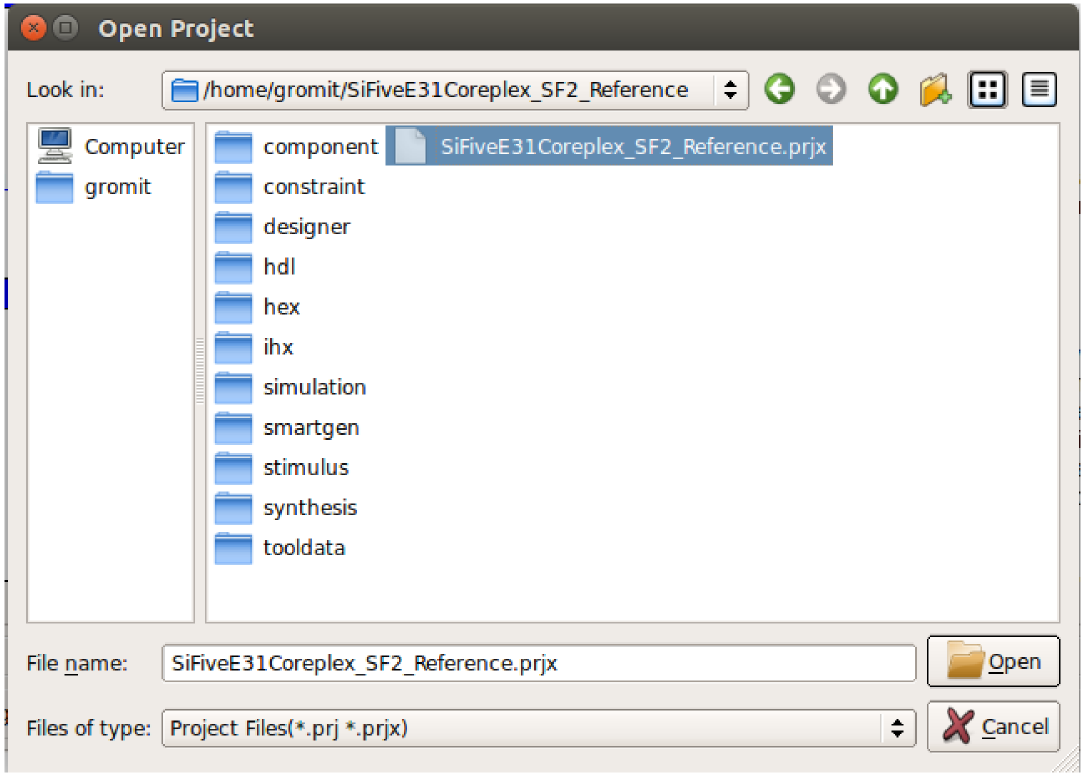
Libero SoC is the most comprehensive and powerful FPGA design and development software available, providing start-to-finish design flow guidance and support for novice and experienced users alike. Libero SoC combines Microsemi's tools with such EDA powerhouses as Synplify Pro® and ModelSim®.

What's New in Libero SoC v11.7

- **Enhanced Constraint Flow** - Libero SoC v11.7 introduces an enhanced constraint flow designed to simplify the management of all constraints for your design. I/O, timing, floor planning and netlist optimization constraints can be created, imported, edited, checked and organized in a single view. Timing constraints need only be entered once, and can be automatically applied to Synplify synthesis, Timing-Driven Layout and Timing Verification. Timing constraints for known hardware blocks and IPs (CCC, Oscillator, MSS/HPMS, FDDR, SERDES), can also be derived automatically. Constraints for these blocks are derived based on the selected block configuration, and can easily be applied to Synthesis, Layout, or Timing Verification.

Libero

- Open existing project located in /home/gromit/SiFiveE31Coreplex_SF2_reference



/home/gromit/SiFiveE31Coreplex_SF2_Reference

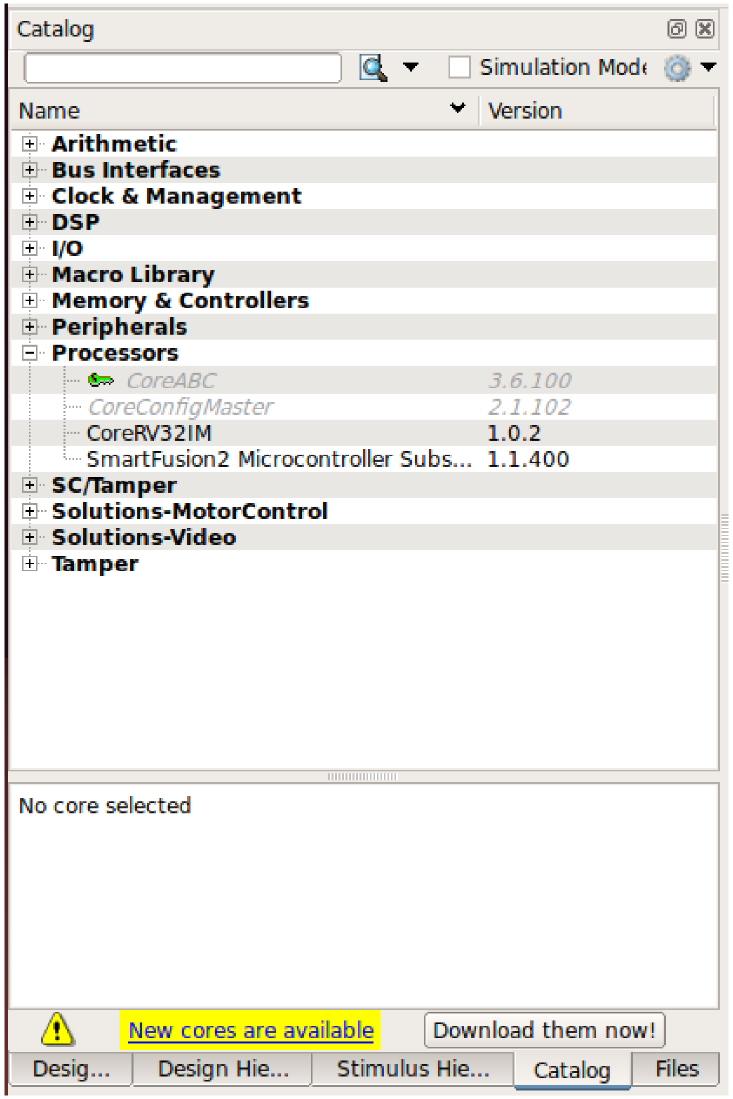
component
constraint
designer
hdl
hex
ihx
simulation
smartgen
stimulus
synthesis
tooldata
SiFiveE31Coreplex_SF2_Reference.prjx

File name: SiFiveE31Coreplex_SF2_Reference.prjx

Files of type: Project Files(*.prj *.prjx)

Libero - IP Catalog

- Libero IP Catalog displays the list of available IP
- IP packages are made available through a remote repository
- IP packages can also be manually added



Catalog

Simulation Mode

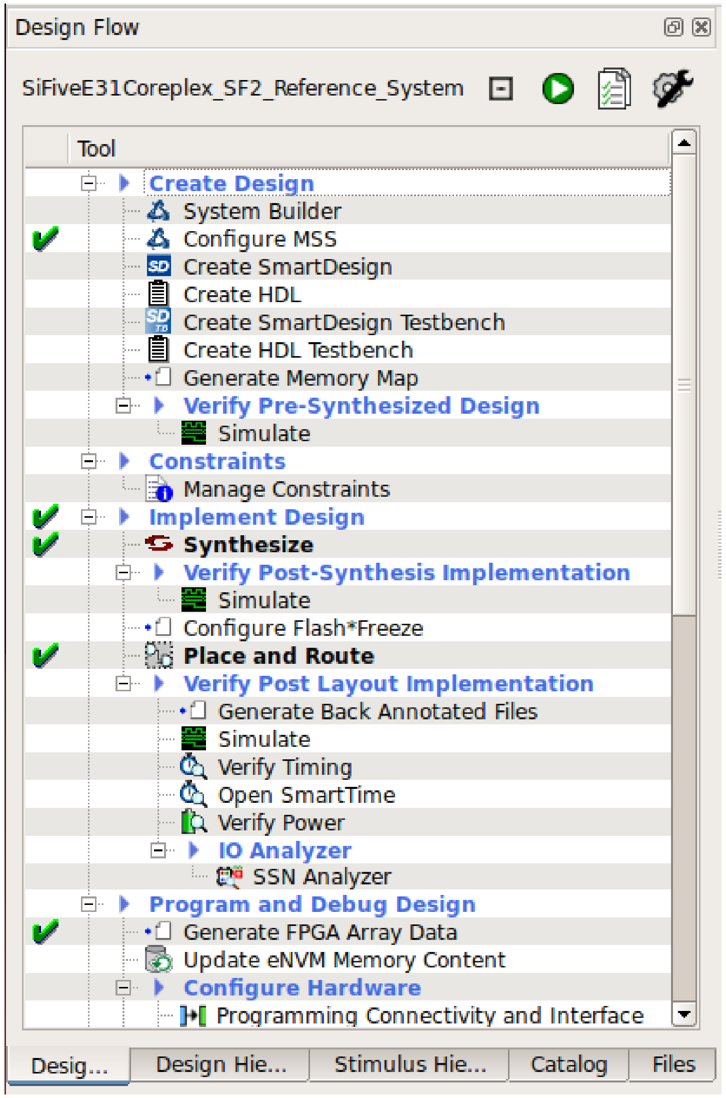
Name	Version
Arithmetic	
Bus Interfaces	
Clock & Management	
DSP	
I/O	
Macro Library	
Memory & Controllers	
Peripherals	
Processors	
CoreABC	3.6.100
CoreConfigMaster	2.1.102
CoreRV32IM	1.0.2
SmartFusion2 Microcontroller Subs...	1.1.400
SC/Tamper	
Solutions-MotorControl	
Solutions-Video	
Tamper	

No core selected

New cores are available

Libero - Design Flow

- Libero Design Flow window guides you through each step of design
  - Design entry
  - Constraints definition
  - Synthesis
  - Place and Route
  - Programming bit stream creation

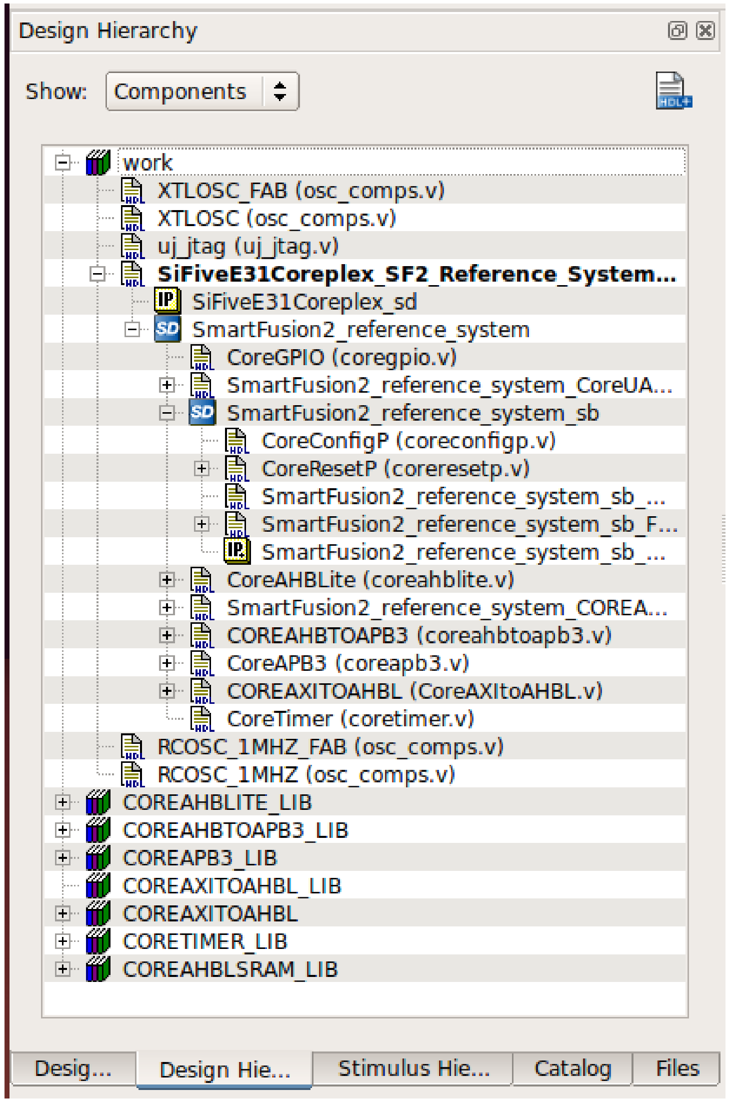

Design Flow

SiFiveE31Coreplex_SF2_Reference_System

Tool
- Create Design
  - System Builder
  - ✓ Configure MSS
  - Create SmartDesign
  - Create HDL
  - Create SmartDesign Testbench
  - Create HDL Testbench
  - Generate Memory Map
  - Verify Pre-Synthesized Design
    - Simulate
- Constraints
  - Manage Constraints
- ✓ Implement Design
  - ✓ Synthesize
  - Verify Post-Synthesis Implementation
    - Simulate
  - Configure Flash*Freeze
  - ✓ Place and Route
  - Verify Post Layout Implementation
    - Generate Back Annotated Files
    - Simulate
    - Verify Timing
    - Open SmartTime
    - Verify Power
    - IO Analyzer
      - SSN Analyzer
- Program and Debug Design
  - ✓ Generate FPGA Array Data
  - Update eNVM Memory Content
  - Configure Hardware
    - Programming Connectivity and Interface

Libero – Design Hierarchy

- Libero Design Hierarchy let's you explore the hierarchy of your design
- Clinking on a hierarchy level will open it in the main canvas

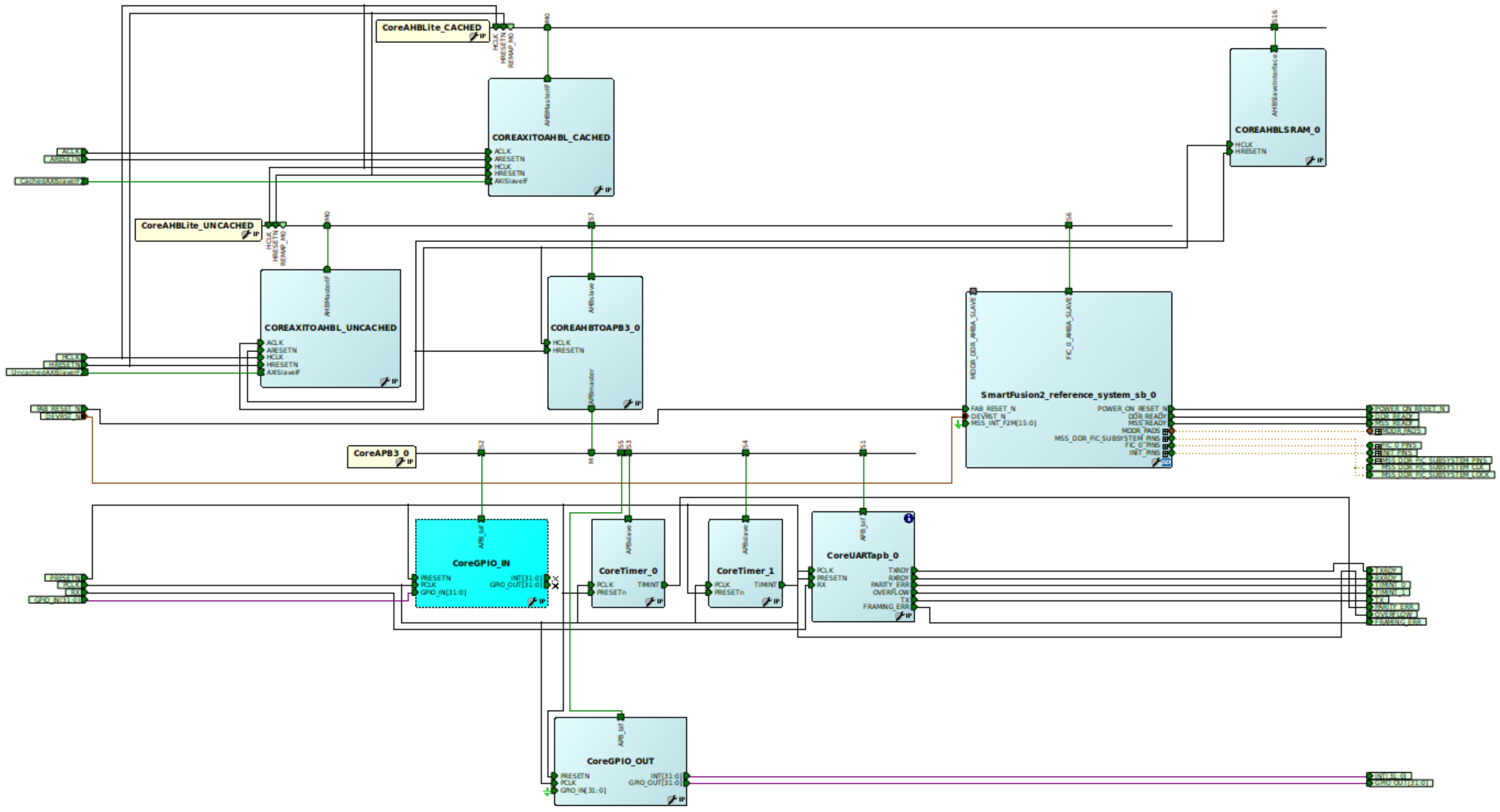


Design Hierarchy

Show: Components

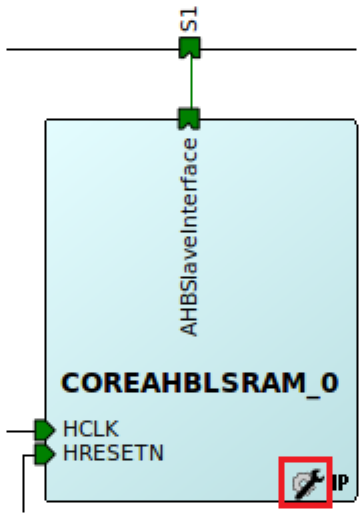
```
work
    XTLOSC_FAB (osc_comps.v)
    XTLOSC (osc_comps.v)
    uj_jtag (uj_jtag.v)
    SiFiveE31Coreplex_SF2_Reference_System...
        SiFiveE31Coreplex_sd
        SmartFusion2_reference_system
            CoreGPIO (coregpio.v)
            SmartFusion2_reference_system_CoreUA...
            SmartFusion2_reference_system_sb
                CoreConfigP (coreconfigp.v)
                CoreResetP (coreresetp.v)
                SmartFusion2_reference_system_sb_...
                SmartFusion2_reference_system_sb_F...
                SmartFusion2_reference_system_sb_...
            CoreAHBLite (coreahblite.v)
            SmartFusion2_reference_system_COREA...
            COREAHBTOAPB3 (coreahbtoapb3.v)
            CoreAPB3 (coreapb3.v)
            COREAXITOAHBL (CoreAXItoAHBL.v)
            CoreTimer (coretimer.v)
    RCOSC_1MHZ_FAB (osc_comps.v)
    RCOSC_1MHZ (osc_comps.v)
COREAHBLITE_LIB
COREAHBTOAPB3_LIB
COREAPB3_LIB
COREAXITOAHBL_LIB
COREAXITOAHBL
CORETIMER_LIB
COREAHBLSRAM_LIB
```

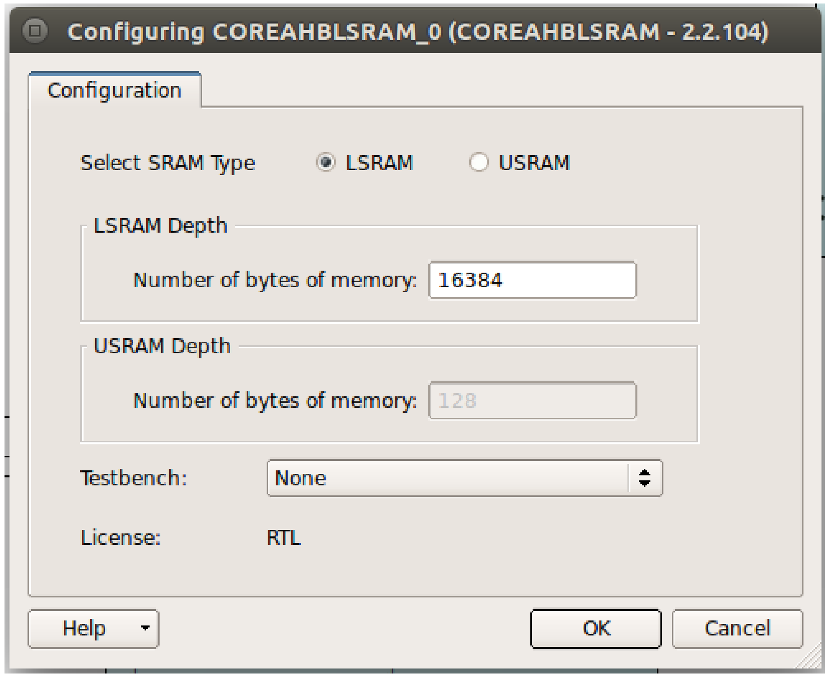
Libero - SmartDesign



Libero – IP Configuration

- Some IP blocks are configurable





Select SRAM Type [✓ LSRAM] USRAM

LSRAM Depth

Number of bytes of memory: 16384

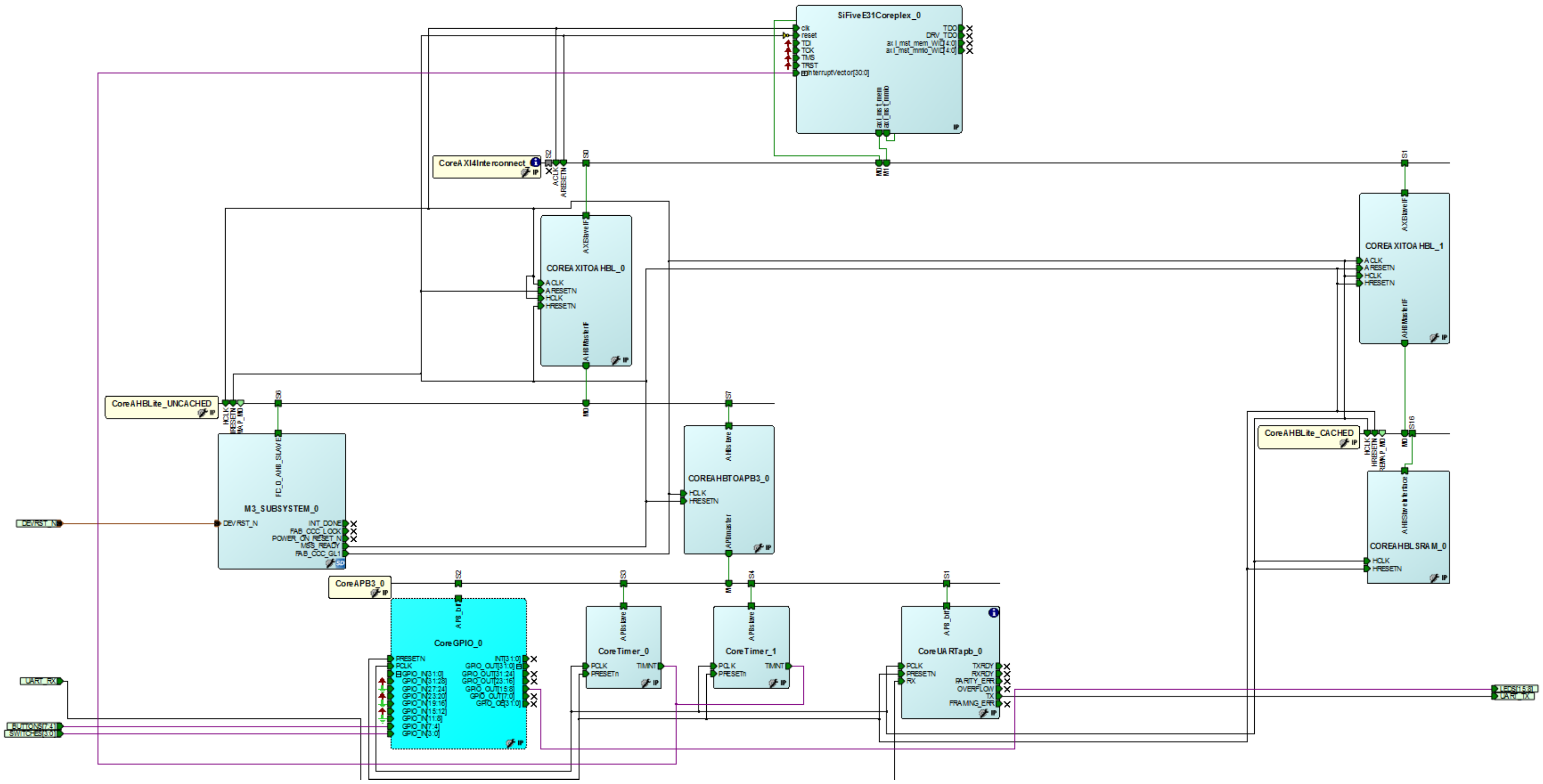
USRAM Depth

Number of bytes of memory: 128

Testbench: None

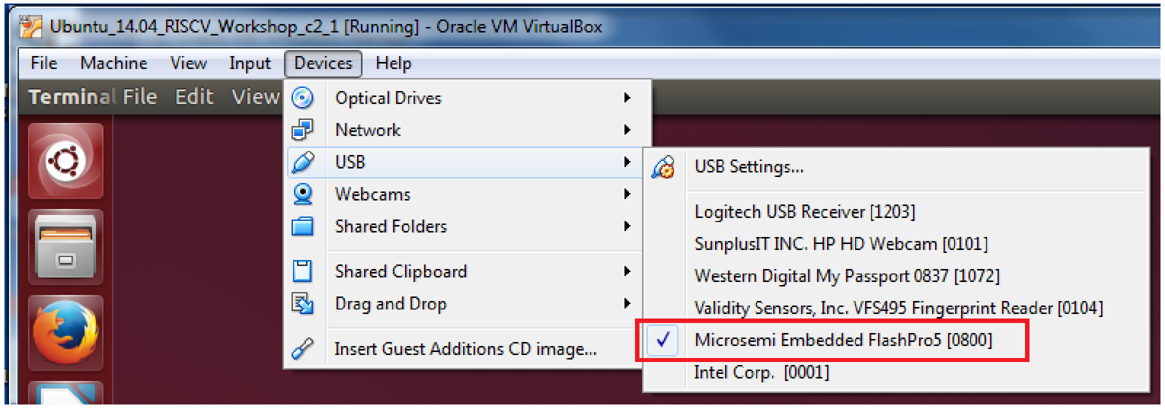
License: RTL

Easy RISC-V Deployment



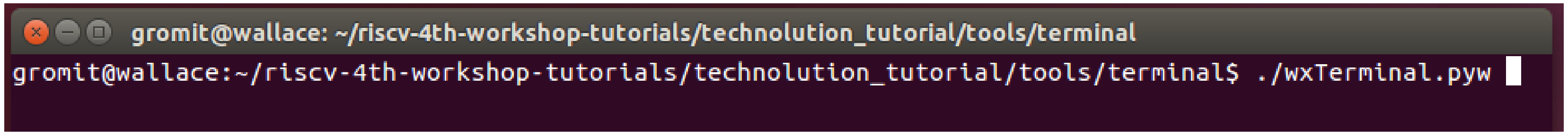
Virtual Machine UART Setup

VM – Capture Board UART Port



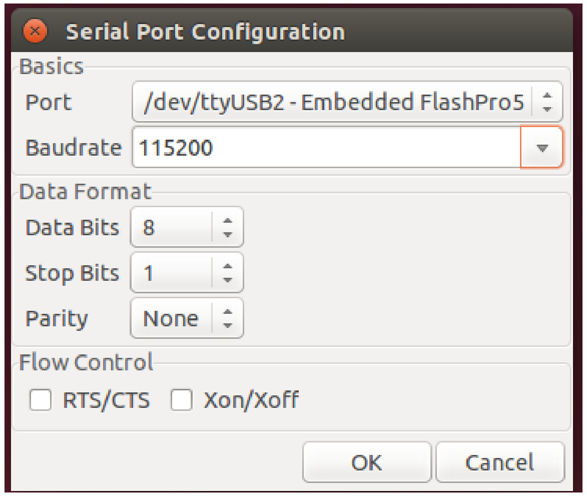
Ubuntu_14.04_RISCV_Workshop_c2_1 [Running] - Oracle VM VirtualBox

- Logitech USB Receiver [1203]
- SunplusIT INC. HP HD Webcam [0101]
- Western Digital My Passport 0837 [1072]
- Validity Sensors, Inc. VFS495 Fingerprint Reader [0104]
- ✓ Microsemi Embedded FlashPro5 [0800]
- Intel Corp. [0001]



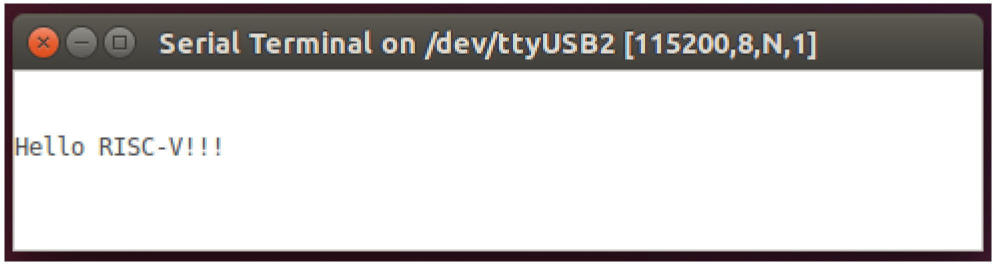
gromit@wallace: ~/riscv-4th-workshop-tutorials/technolution_tutorial/tools/terminal

```
gromit@wallace:~/riscv-4th-workshop-tutorials/technolution_tutorial/tools/terminal$ ./wxTerminal.pyw
```



Serial Port Configuration

Basics	
Port	/dev/ttyUSB2 - Embedded FlashPro5
Baudrate	115200
Data Format	
Data Bits	8
Stop Bits	1
Parity	None
Flow Control	
☐ RTS/CTS	☐ Xon/Xoff



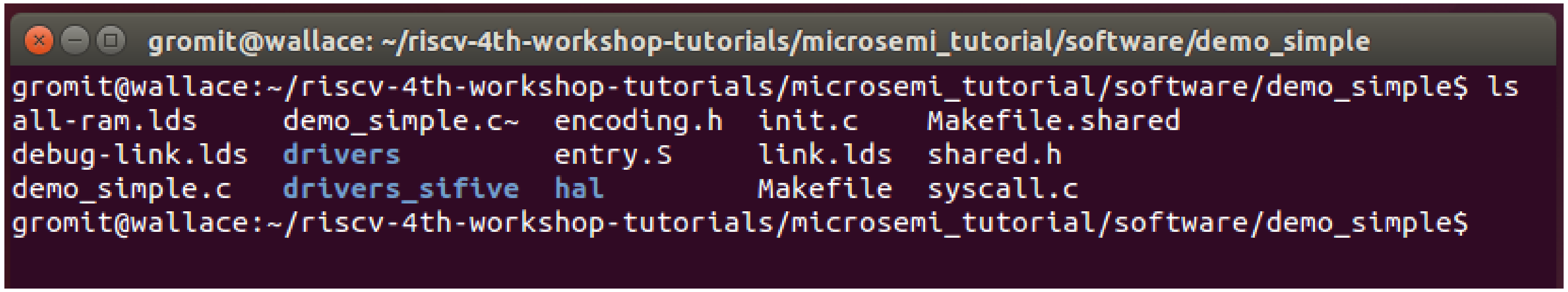
Serial Terminal on /dev/ttyUSB2 [115200,8,N,1]

```
Hello RISC-V!!!
```

Building RISC-V Bare Metal Application

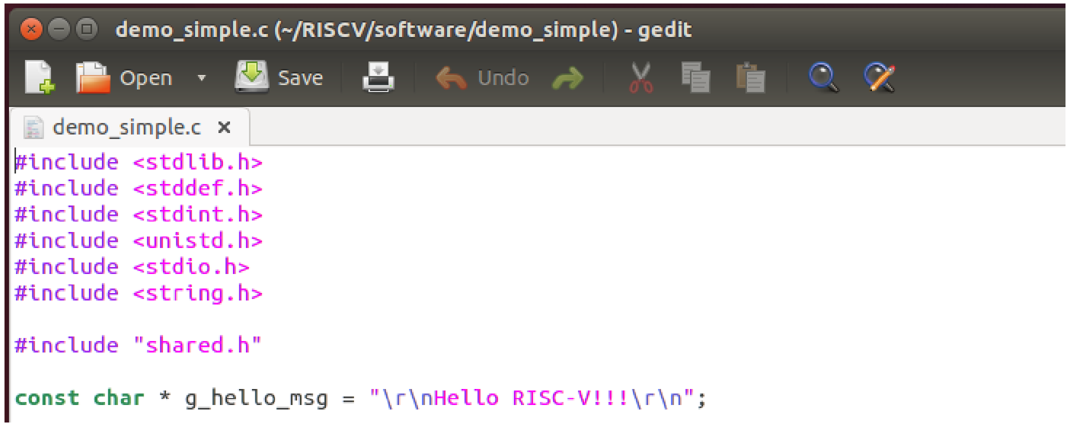
Building RISC-V Bare Metal Application

- Open a terminal and move to the following directory:
  - $ cd ~/riscv-4th-workshop-tutorials/microsemi_tutorial/software/demo_simple



```
gromit@wallace: ~/riscv-4th-workshop-tutorials/microsemi_tutorial/software/demo_simple
gromit@wallace:~/riscv-4th-workshop-tutorials/microsemi_tutorial/software/demo_simple$ ls
all-ram.lds      demo_simple.c~   encoding.h   init.c     Makefile.shared
debug-link.lds   drivers          entry.S      link.lds   shared.h
demo_simple.c    drivers_sifive   hal          Makefile   syscall.c
gromit@wallace:~/riscv-4th-workshop-tutorials/microsemi_tutorial/software/demo_simple$
```

- Edit demo_simple.c with the editor of your choice
  - Replace the g_hello_msg with the text of your choice



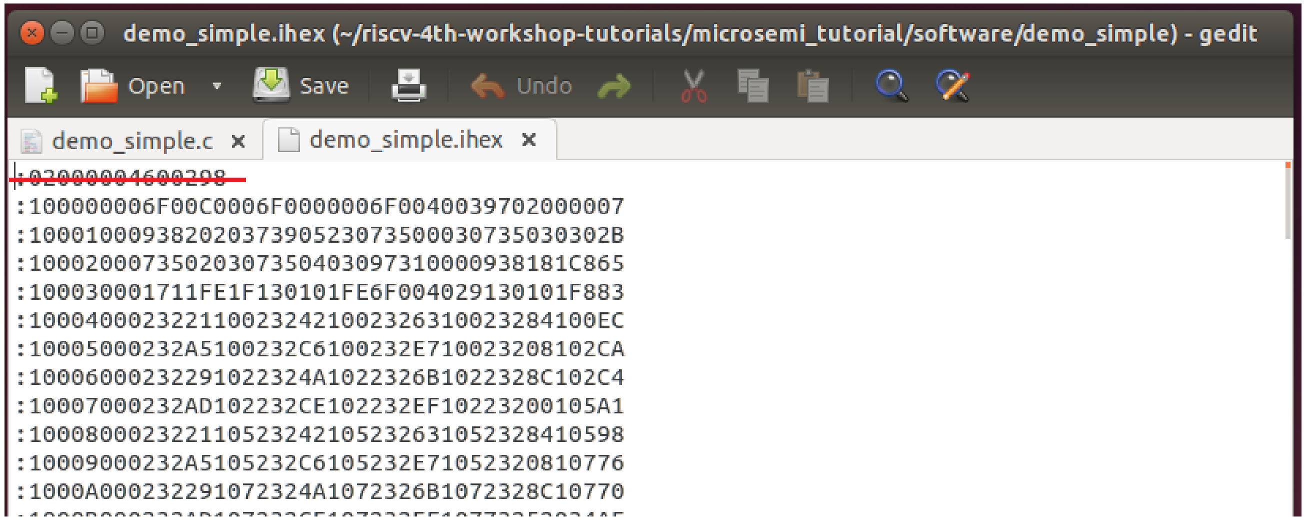
```
#include <stdlib.h>
#include <stddef.h>
#include <stdint.h>
#include <unistd.h>
#include <stdio.h>
#include <string.h>

#include "shared.h"

const char * g_hello_msg = "\r\nHello RISC-V!!!\r\n";
```

Building RISC-V Bare Metal Application

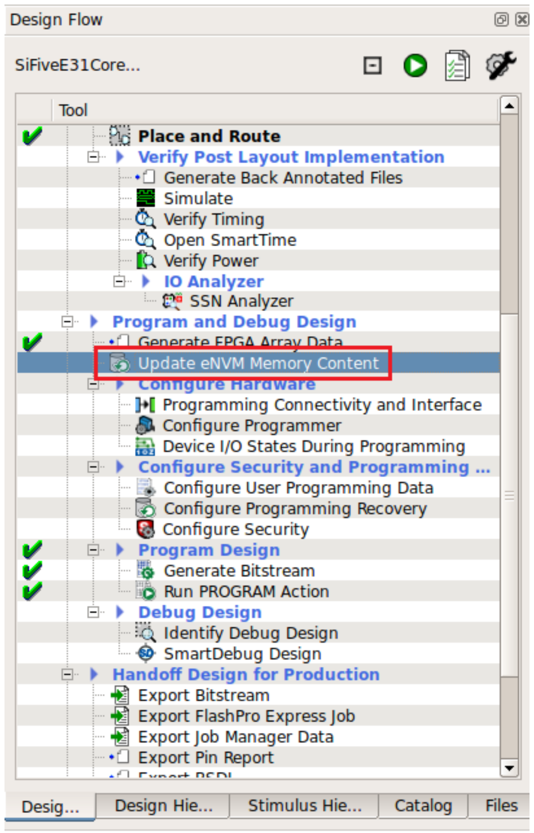
- Go back terminal window and run make:
  - $ make
- Create Intel-Hex file from created executable:
  - $ riscv32-unknown-elf-objcopy demo_simple demo_simple.ihex
- Open demo_simple-ihex in your favorite editor, remove the first line and save as demo_simple.ihx


demo_simple.ihex (~/riscv-4th-workshop-tutorials/microsemi_tutorial/software/demo_simple) - gedit
Open
Save
Undo
demo_simple.c
demo_simple.ihex

```
~~:02000004600298~~
:100000006F00C0006F0000006F0040039702000007
:100010009382020373905230735000307350303028
:10002000735020307350403097310000938181C865
:100030001711FE1F130101FE6F004029130101F883
:1000400023221100232421002326310023284100EC
:10005000232A5100232C6100232E710023208102CA
:10006000232291022324A1022326B1022328C102C4
:10007000232AD102232CE102232EF10223200105A1
:100080002322110523242105232631052328410598
:10009000232A5105232C6105232E71052320810776
:1000A00023229107232A1072326B1072328C10770
```

Program Non Volatile Memory (Flash)

- Select Update eNVM Memory Content in the Libero Design Flow pane



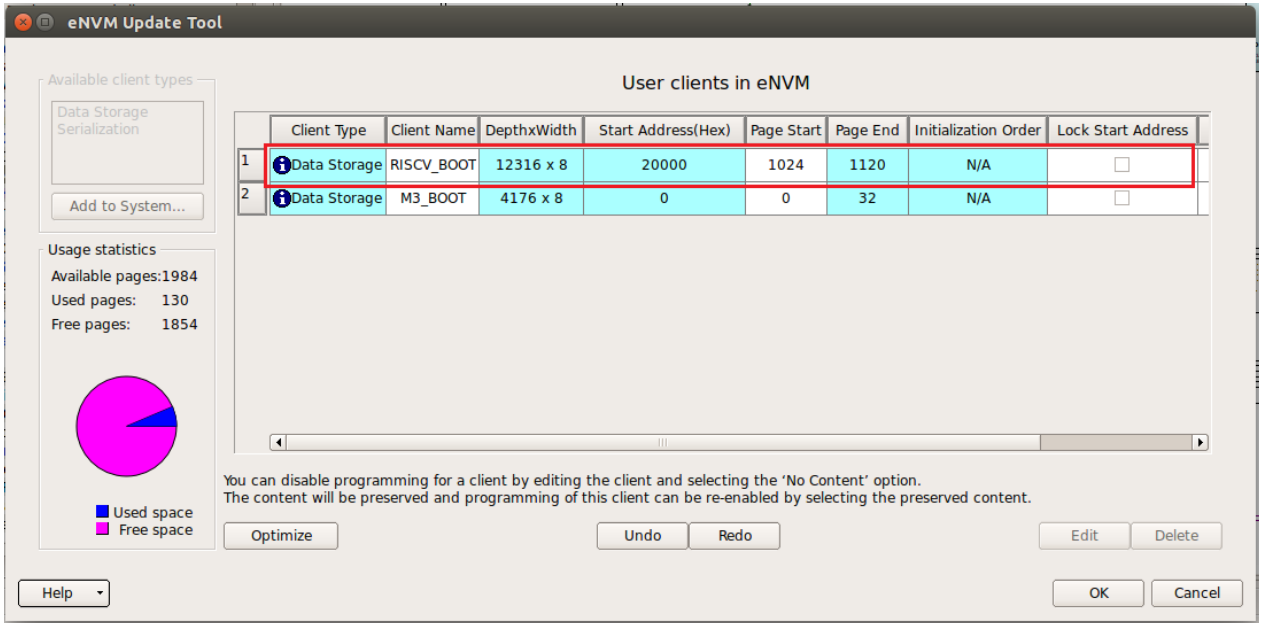
Design Flow

SiFiveE31Core...

Tool

- ✓ Place and Route
  - Verify Post Layout Implementation
    - Generate Back Annotated Files
    - Simulate
    - Verify Timing
    - Open SmartTime
    - Verify Power
    - IO Analyzer
      - SSN Analyzer
- Program and Debug Design
  - ✓ Generate FPGA Array Data
  - Update eNVM Memory Content
  - Configure Hardware
    - Programming Connectivity and Interface
    - Configure Programmer
    - Device I/O States During Programming
  - Configure Security and Programming ...
    - Configure User Programming Data
    - Configure Programming Recovery
    - Configure Security
  - ✓ Program Design
    - ✓ Generate Bitstream
    - ✓ 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

Desig... | Design Hie... | Stimulus Hie... | Catalog | Files



eNVM Update Tool

Available client types

Data Storage
Serialization

Add to System...

User clients in eNVM

	Client Type	Client Name	DepthxWidth	Start Address(Hex)	Page Start	Page End	Initialization Order	Lock Start Address
1	Data Storage	RISCV_BOOT	12316 x 8	20000	1024	1120	N/A	☐
2	Data Storage	M3_BOOT	4176 x 8	0	0	32	N/A	☐

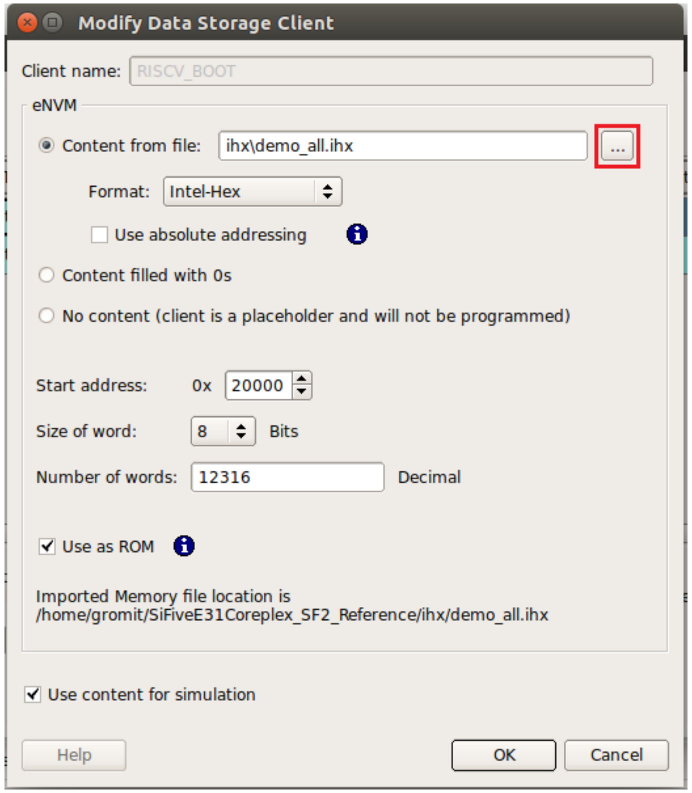
Usage statistics

Available pages: 1984
Used pages: 130
Free pages: 1854

You can disable programming for a client by editing the client and selecting the 'No Content' option.
The content will be preserved and programming of this client can be re-enabled by selecting the preserved content.

Program Non Volatile Memory (Flash)

- Select the demo_simple.ihx file you created. Click OK several time, until all dialog windows are closed



Modify Data Storage Client

Client name: RISCV_BOOT

eNVM

(●) Content from file: ihx\demo_all.ihx ...

Format: Intel-Hex

[ ] Use absolute addressing

( ) Content filled with 0s

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

Start address: 0x 20000

Size of word: 8 Bits

Number of words: 12316 Decimal

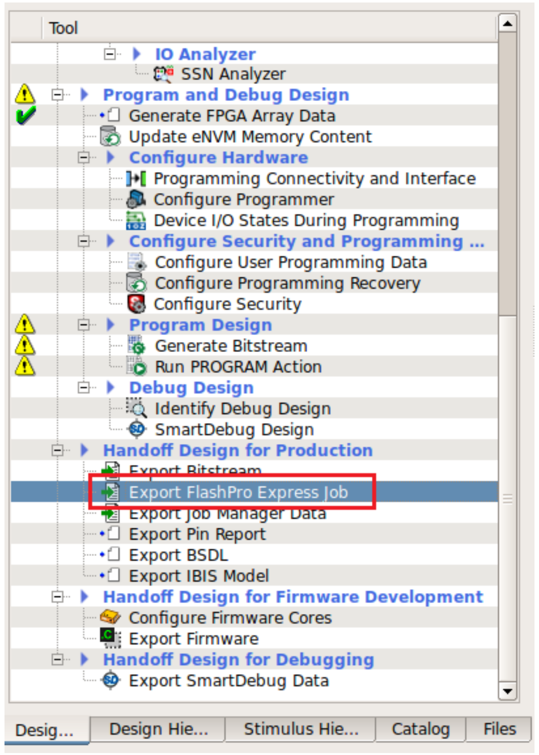
[✓] Use as ROM

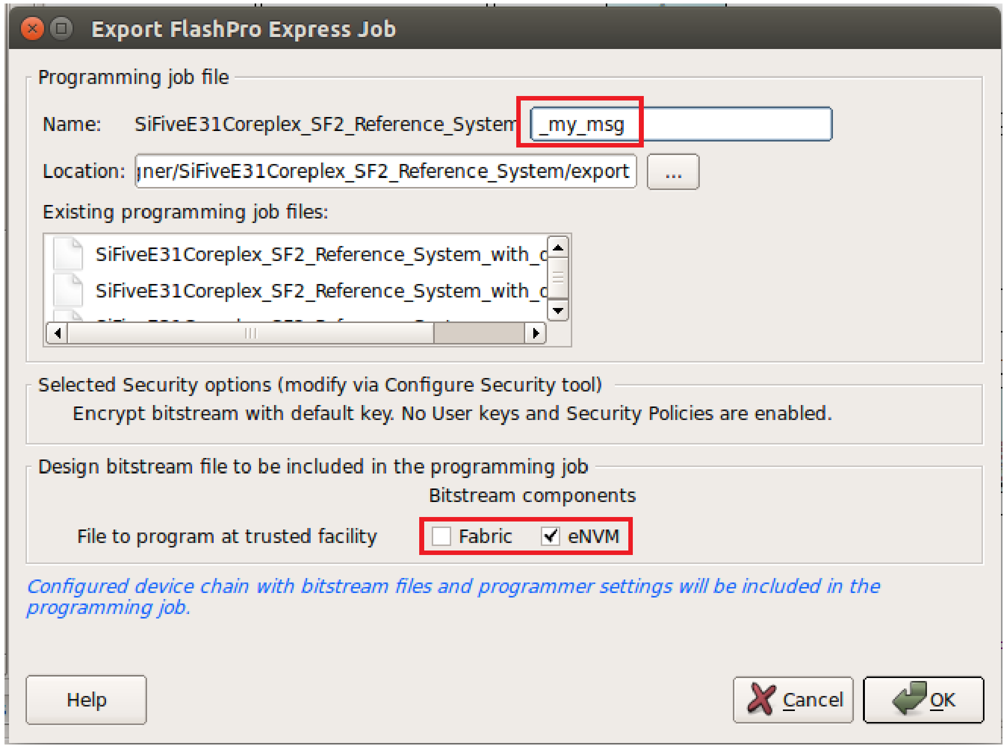
Imported Memory file location is
/home/gromit/SiFiveE31Coreplex_SF2_Reference/ihx/demo_all.ihx

[✓] Use content for simulation

Program Non Volatile Memory (Flash)

- Select *Export FlashPro Express job*
- Add the suffix of your choice to the exportedName
- Untick Fabric
- Make a note of the location where the files will be exported


Tool
- IO Analyzer
  - SSN Analyzer
- Program and Debug Design
  - Generate FPGA Array Data
  - Update eNVM Memory Content
  - Configure Hardware
    - Programming Connectivity and Interface
    - Configure Programmer
    - Device I/O States During Programming
  - Configure Security and Programming ...
    - Configure User Programming Data
    - Configure Programming Recovery
    - Configure Security
  - Program Design
    - Generate Bitstream
    - 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


Export FlashPro Express Job

Programming job file

Name: SiFiveE31Coreplex_SF2_Reference_System _my_msg

Location: gner/SiFiveE31Coreplex_SF2_Reference_System/export

Existing programming job files:
SiFiveE31Coreplex_SF2_Reference_System_with_c
SiFiveE31Coreplex_SF2_Reference_System_with_c

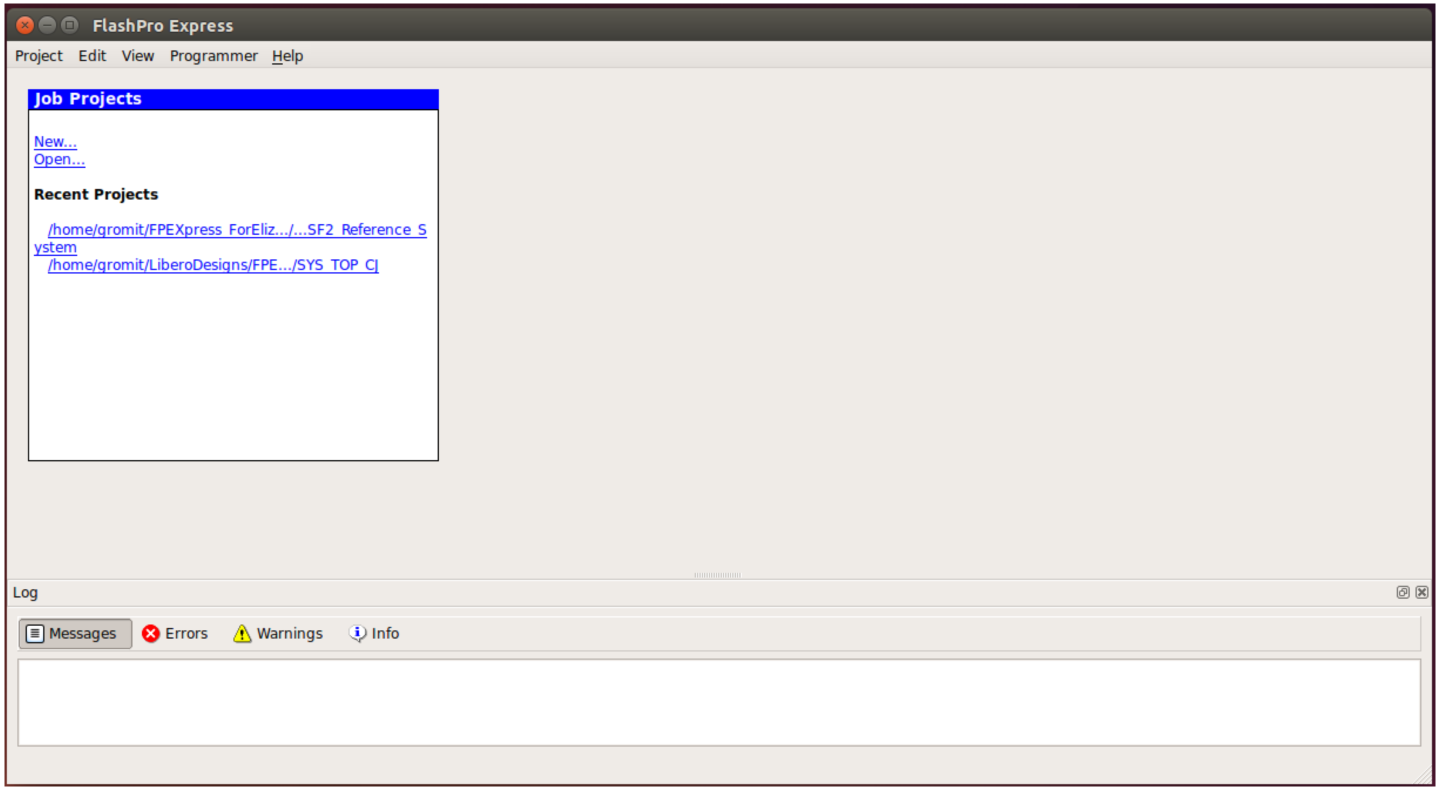
Selected Security options (modify via Configure Security tool)
Encrypt bitstream with default key. No User keys and Security Policies are enabled.

Design bitstream file to be included in the programming job
Bitstream components
File to program at trusted facility ☐ Fabric ☑ eNVM

Configured device chain with bitstream files and programmer settings will be included in the programming job.

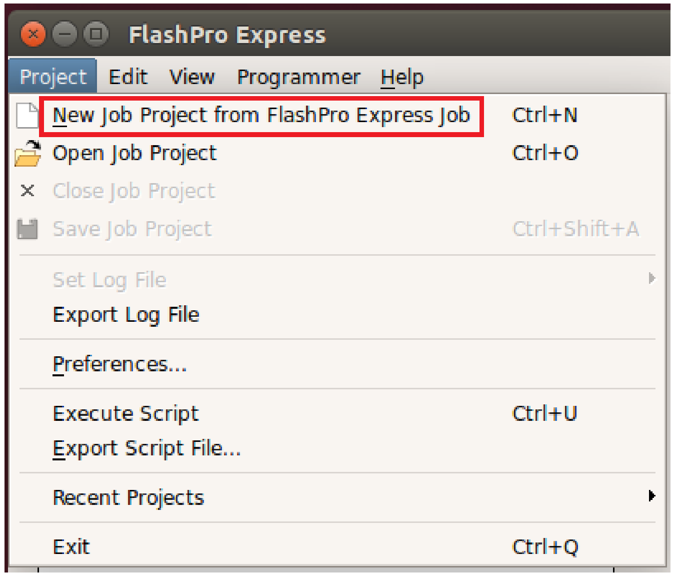
Program Non Volatile Memory (Flash)

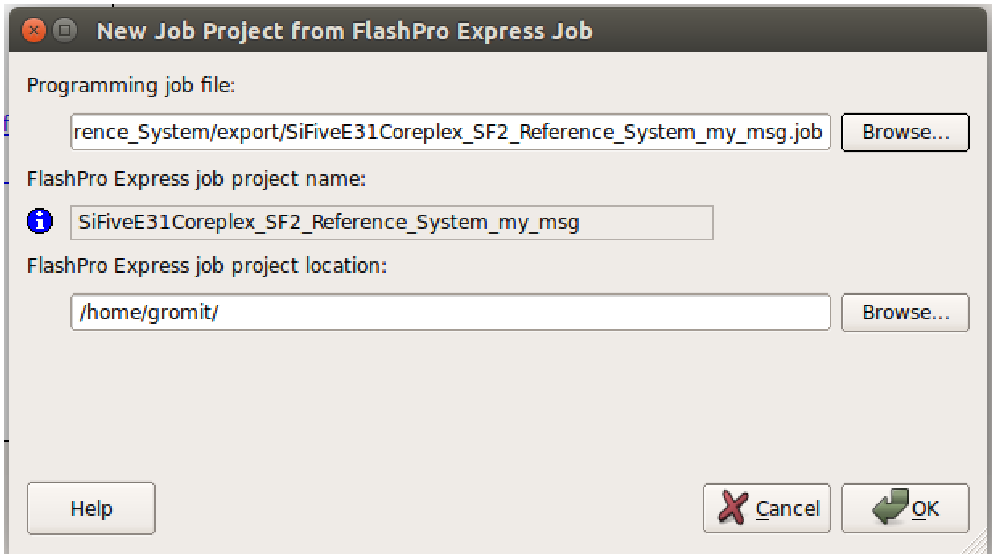
- Start FlashProExpress
  - $ cd ~
  - ./start_FPExpress


FlashPro Express
Project Edit View Programmer Help
Job Projects
New...
Open...
Recent Projects
/home/gromit/FPEXpress_ForEliz.../...SF2_Reference_System
/home/gromit/LiberoDesigns/FPE.../SYS_TOP_CJ
Log
Messages
Errors
Warnings
Info

Program Non Volatile Memory (Flash)

- Select Project->New Job Project from FlashPro Express job
- Select the export FlashProExpress job created in Libero




New Job Project from FlashPro Express Job

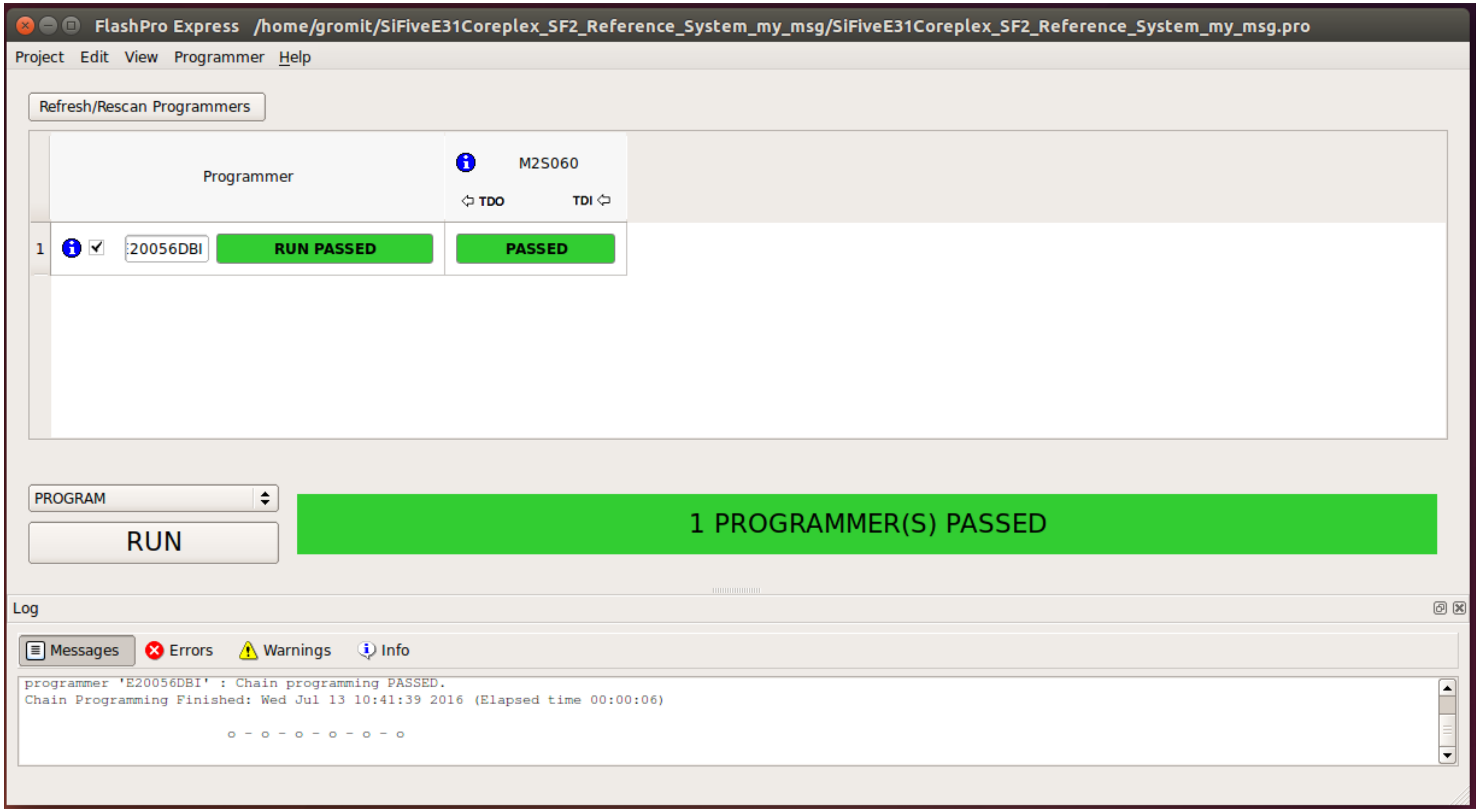
Programming job file:
rence_System/export/SiFiveE31Coreplex_SF2_Reference_System_my_msg.job

FlashPro Express job project name:
SiFiveE31Coreplex_SF2_Reference_System_my_msg

FlashPro Express job project location:
/home/gromit/

Program Non Volatile Memory (Flash)

- Click Run

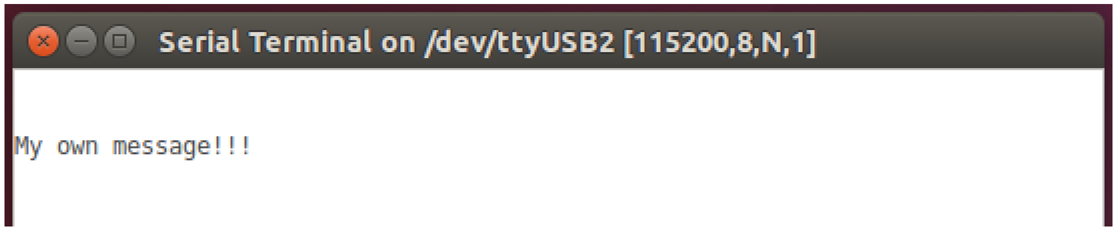

FlashPro Express /home/gromit/SiFiveE31Coreplex_SF2_Reference_System_my_msg/SiFiveE31Coreplex_SF2_Reference_System_my_msg.pro
Project Edit View Programmer Help
Refresh/Rescan Programmers
Programmer
M2S060
TDO
TDI
1
:20056DBI
RUN PASSED
PASSED
PROGRAM
RUN
1 PROGRAMMER(S) PASSED
Log
Messages
Errors
Warnings
Info

```
programmer 'E20056DBI' : Chain programming PASSED.
Chain Programming Finished: Wed Jul 13 10:41:39 2016 (Elapsed time 00:00:06)

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

Program Non Volatile Memory (Flash)

- Click the reset button on the board



```
My own message!!!
```

Programming Next Session’s Design

NATIVE FlashPro Users
Drag & drop
vectorblox_tutorial/FPExpress
folder to host OS

- Start VM and Terminal app

```
$ cd ~
$ git clone https://github.com/riscv/riscv-4th-workshop-tutorials.git
```

- Program FPGA bitstream

```
$ cd riscv-4th-workshop-tutorials/
$ cd vectorblox_tutorial/
$ source env.sh
$ cd software; vi main.c ; make clean all; cd ..
$ ./program.sh fpga      #(FPGA & ELF, 15min)
$ ./program.sh           #(ELF only, 90seconds)
```

- See LEDs blink, “Hello World” and audio pass-thru

```
$ cat /dev/ttyUSB2
```

35