Embedded Linux

Have a question? Enter a search term SEARCH

Popular Search: USB GPIO (), USB Relay (), FPGA ()

Saturn, Microblaze and Linux – How to Run Linux on Saturn Spartan 6 FPGA Module – Part I

4729 views March 9, 2016 admin 10

(https://numato.com/help/wp-content/uploads/2016/03/microblaze_Linux.png)

Part I
Part II (https://numato.com/kb/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-ii/)
Part III (http://docs.numato.com/knowledge/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-iii/)
Part IV (https://numato.com/kb/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-iv/)

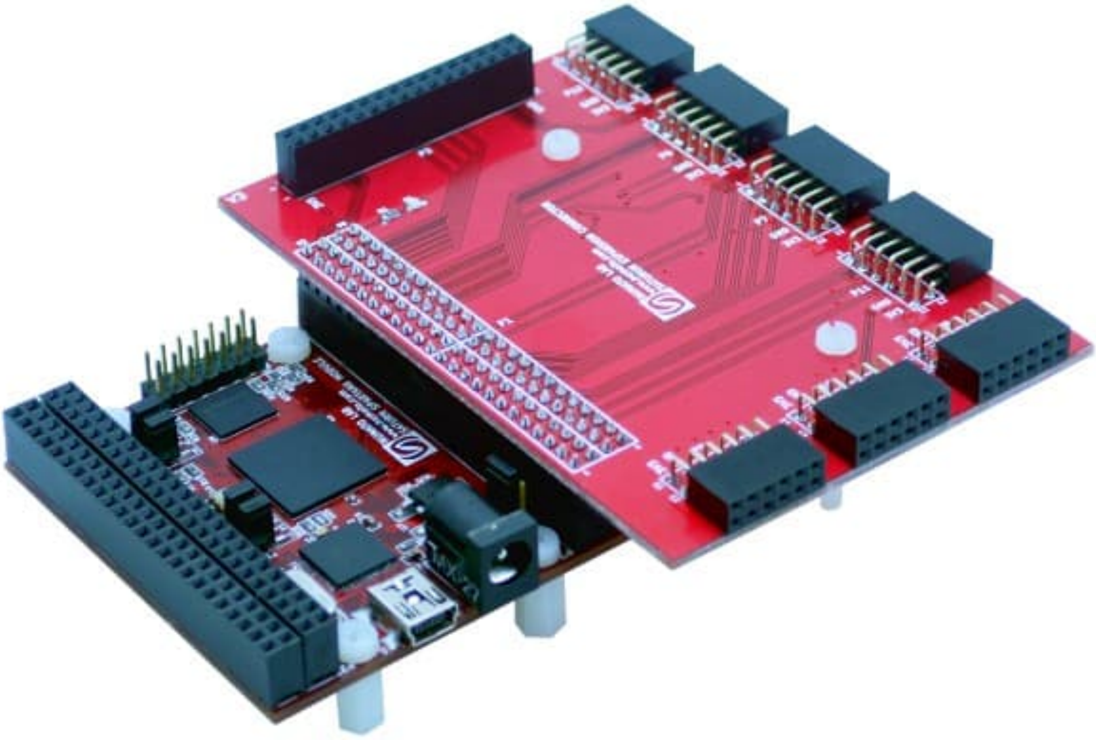
Introduction

Though a low cost FPGA board, Saturn Spartan 6 FPGA Module (https://numato.com/saturn-spartan-6-fpga-development-board-with-ddr-sdram) is perfectly capable of running Linux with Xilinx's popular Microblaze 32 bit soft processor. Recently we spent some time trying to figure out how to build Linux for Saturn. Surprisingly (or not!) it turned out to be fairly easy to do. Building the Linux Kernel was very easy, but creating a bootloader to boot Linux off of the on board SPI flash was a little more involved. We are excited to share what we learned with our readers. Though this article series is mainly focused on building Linux for Saturn a Xilinx Spartan 6 based FPGA module with DDR Memory (https://numato.com/saturn-spartan-6-fpga-development-board-with-ddr-sdram), the general workflow is applicable to any Xilinx Microblaze based embedded system.

In this article series, we will cover the following topics. Each of these topics will be a separate article.

- Building Microblaze based embedded platform for Linux using Xilinx EDK
- Creating Device Tree and building Linux
- Booting Linux using XMD
- Creating a bootloader to load Linux kernel from SPI flash

You can read more about Saturn at the product page (https://numato.com/saturn-spartan-6-fpga-development-board-with-ddr-sdram). User manual and tools can be downloaded from the same page.



(https://docs.numato.com/wp-content/uploads/2016/03/SaturnExpansionConnector_41.jpg)

Saturn Spartan 6 FPGA Module with IO Breakout board

Tool Version Compatibility and Help

This article series uses Xilinx EDK shipped with Xilinx ISE 14.6. So the screenshots and directions/commands may be different on your system if you are using a different version. Please adapt the steps mentioned in this article to make it work with your specific tool version. The best place to ask questions about ISE, EDK (XPS/SDK), Microblaze and Spartan 6 FPGA is Xilinx community forums at http://forums.xilinx.com/. You may optionally open a web case at http://www.xilinx.com/support/clearexpress/websupport.htm. Questions specific to Saturn Spartan 6 FPGA Development Board may be asked at the comments section on the product page or contact Numato Lab (https://numato.com/contact-us/). Please note that author uses Windows to run Xilinx tools and Ubuntu 14.04 or compatible for building Linux kernel.
This article series was written in June 2014. Over time, the tools and Linux kernel itself will evolve and some of the specific steps included in this article series may become incorrect/invalid. If you come across such a situation, please let us know and we will try to fix it. If you have a fix, please share with fellow readers in the comments section below. It is recommended that you have the following tools/software available to follow this tutorial successfully.

1. Saturn Spartan 6 FPGA Module with DDR (https://numato.com/saturn-spartan-6-fpga-development-board-with-ddr-sdram)
2. Saturn configuration tool (Download from product page (https://numato.com/saturn-spartan-6-fpga-development-board-with-ddr-sdram))
3. Xilinx ISE installed with Xilinx EDK (Download from www.xilinx.com. This tutorial uses ISE 14.6)
4. Buildroot (This tutorial uses 2014.5 release (https://buildroot.uclibc.org/downloads/buildroot-2014.05.tar.gz). You can download the latest release from here (https://buildroot.uclibc.org/))
5. Xilinx Platform Cable – USB (Optional but very useful for debugging)

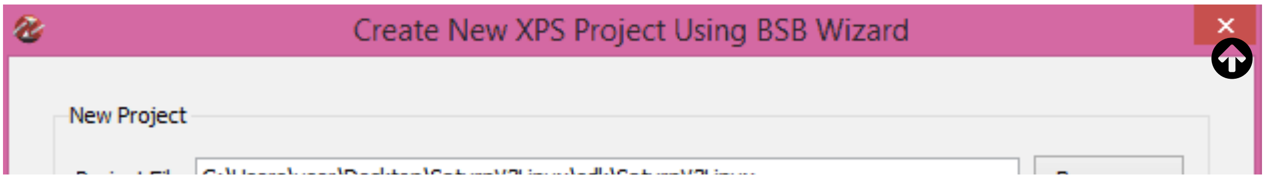
All files that is used/generated while writing this article series can be downloaded here (https://github.com/numato/samplecode/tree/master/FPGA/Saturn/SaturnLinux).

Building Microblaze based embedded platform for Linux using Xilinx EDK

Since Saturn being a FPGA platform which is not a microprocessor rather a collection configurable logic elements, we need to create a microprocessor out of the logic elements/resources available. Doing this from the scratch is going to be very difficult. Fortunately Xilinx already did this for us and the processor design is called Microblaze. All that we need to do is to use Microblaze in our design and make appropriate connections.

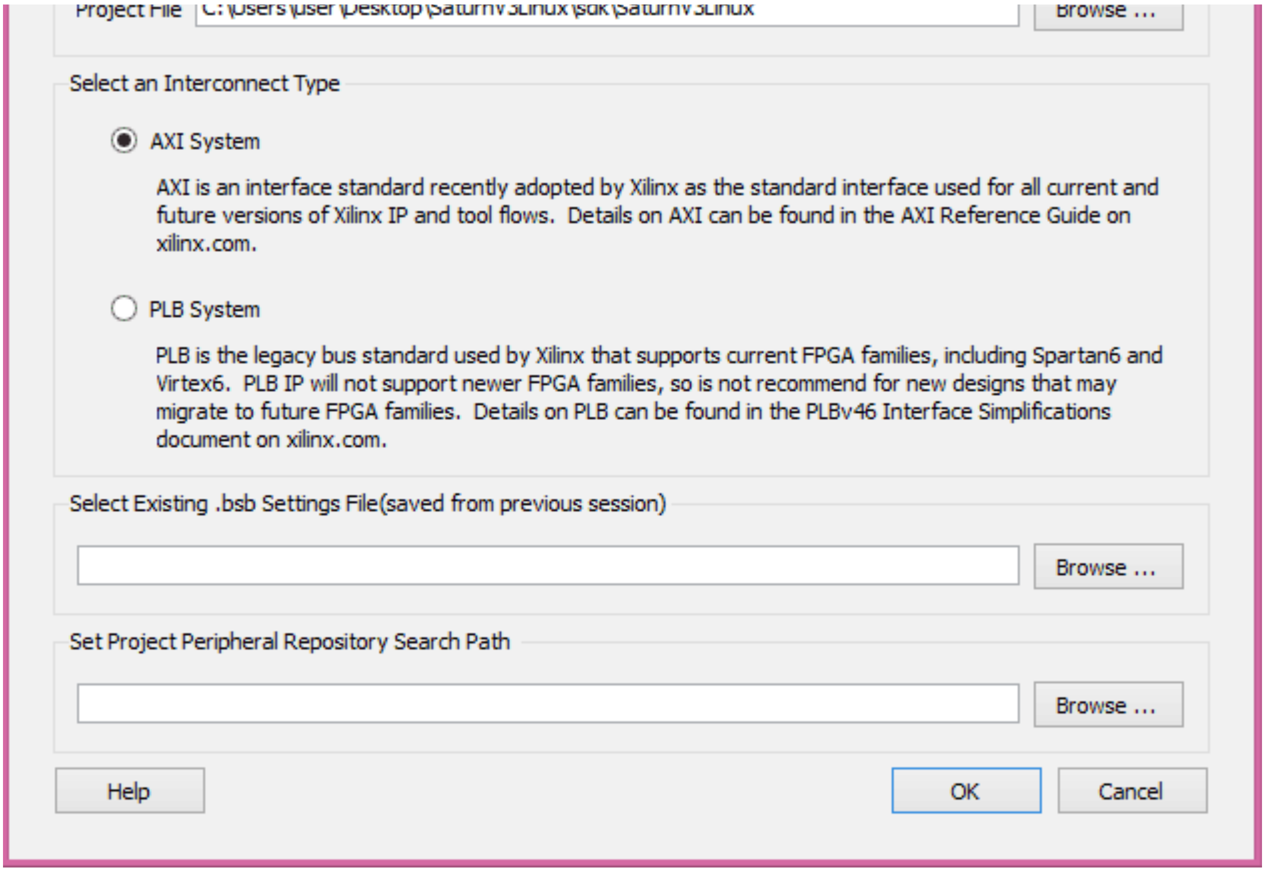
Before continuing to build the Microblaze embedded platform for Linux, it may be worth reading this article (https://numato.com/kb/creating-xilinx-edk-test-project-saturn-your-first-microblaze-processor-based-embedded-desig/) to get familiar with Xilinx EDK and Saturn.

Now let's get started. We will need to use XPS base System Builder (BSB) to create a new project. To make the process easier, download BSB Wizard files from here (https://github.com/numato/samplecode/tree/master/FPGA/Saturn/edk/bsb/SaturnV3) and extract to the folder "C:\Xilinx\xx.xx\ISE_DS\EDK\board". Once extracted, the complete path to the file should look like "C:\Xilinx\xx.xx\ISE_DS\EDK\board\NumatoLab\ipxact\Saturn_LX45_V3\data". Now go to All Programs > Xilinx Design Tools > EDK and start Xilinx Platform Studio. Once XPS is loaded, click on "New BSB Project" under File menu or press Ctrl + Shift + B to invoke the BSB wizard. If everything went fine, you should be presented with BSB wizard window as shown below.



Create New XPS Project Using BSB Wizard

New Project



Project File C:\Users\user\Desktop\SaturnV3Linux\sdk\SaturnV3Linux

Select an Interconnect Type

● AXI System

AXI is an interface standard recently adopted by Xilinx as the standard interface used for all current and future versions of Xilinx IP and tool flows. Details on AXI can be found in the AXI Reference Guide on xilinx.com.

○ PLB System

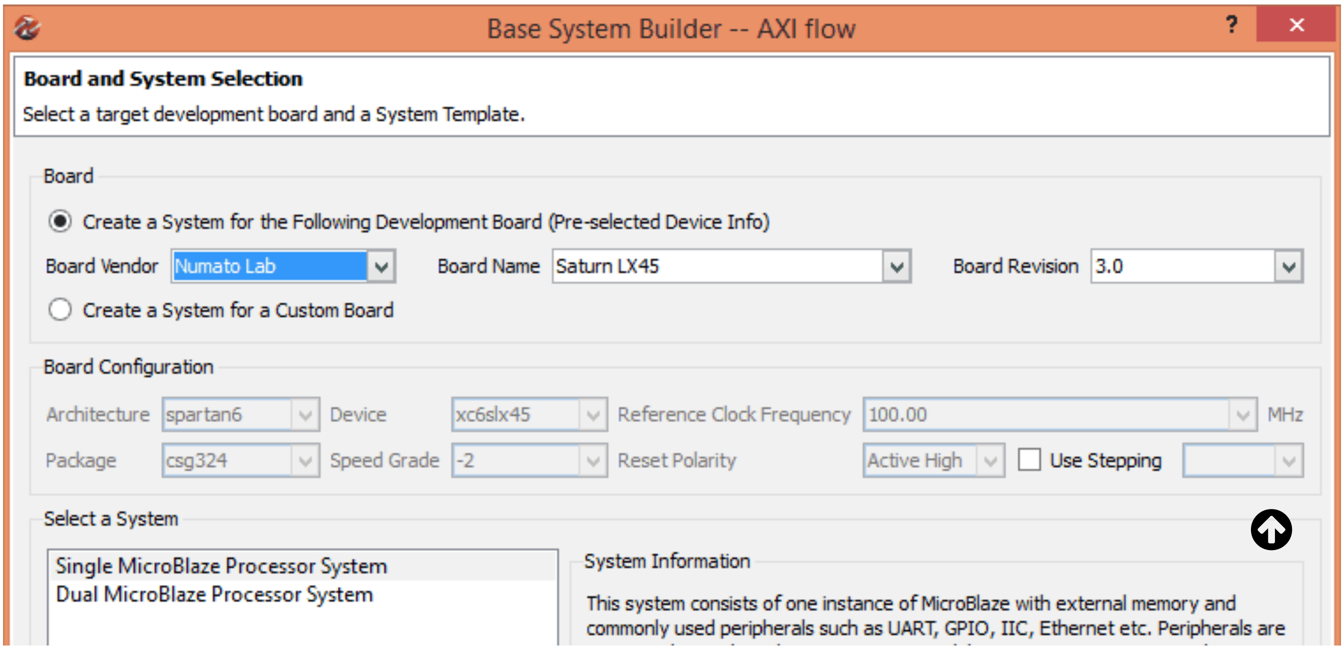
PLB is the legacy bus standard used by Xilinx that supports current FPGA families, including Spartan6 and Virtex6. PLB IP will not support newer FPGA families, so is not recommend for new designs that may migrate to future FPGA families. Details on PLB can be found in the PLBv46 Interface Simplifications document on xilinx.com.

Select Existing .bsb Settings File(saved from previous session)

Set Project Peripheral Repository Search Path

(https://numato.com/help/wp-content/uploads/2016/03/bsbwizard_create_project1.png)

Enter path to a folder where you want to store the project and a project name (Here the project name is SaturnV3Linux in the image above). Select AXI System in the interconnect Type section and click OK. In the next page of the wizard, select “Numato Lab” from the board Vendor combo box (This option should be available if you have placed the BSB wizard support files for Saturn properly as mentioned earlier). Select Saturn LX45 and 3.0 from Board Name and Board version combo boxes respectively and click “Next” button to go to the next page.



Base System Builder -- AXI flow

Board and System Selection

Select a target development board and a System Template.

Board

● Create a System for the Following Development Board (Pre-selected Device Info)

Board Vendor: Numato Lab   Board Name: Saturn LX45   Board Revision: 3.0

○ Create a System for a Custom Board

Board Configuration

Architecture: spartan6   Device: xc6slx45   Reference Clock Frequency: 100.00 MHz

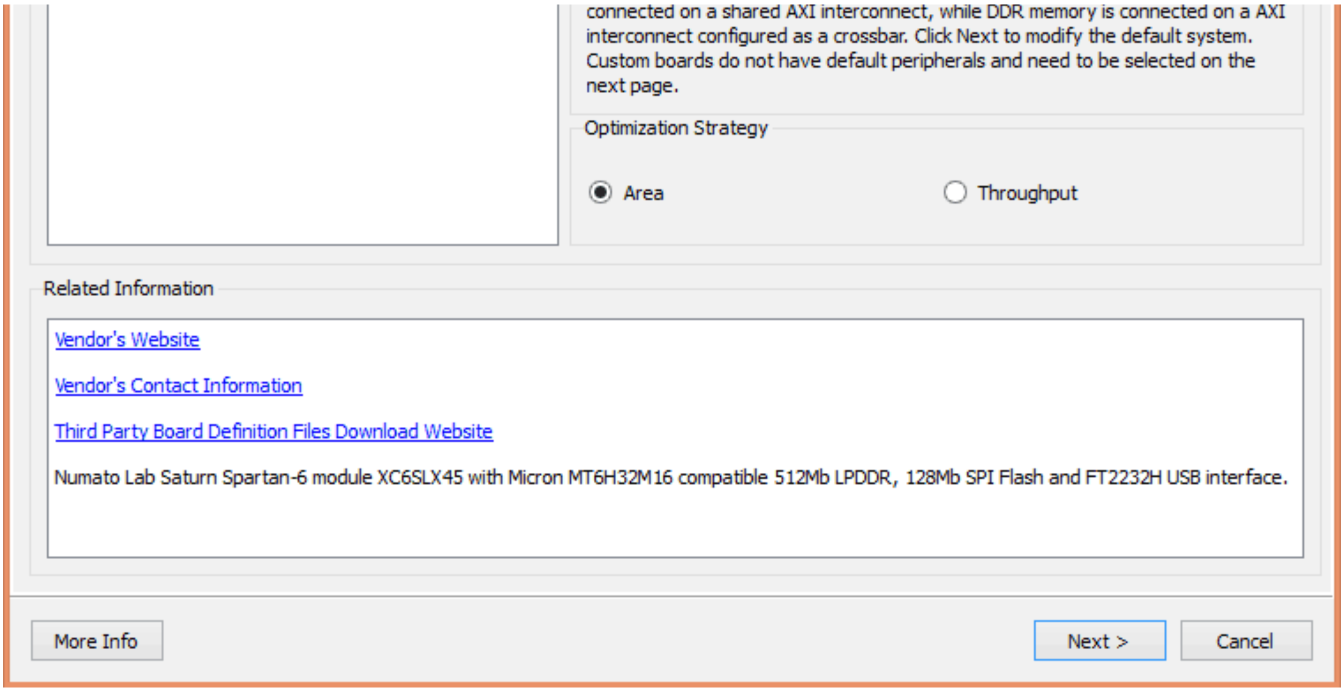
Package: csg324   Speed Grade: -2   Reset Polarity: Active High   □ Use Stepping

Select a System

Single MicroBlaze Processor System
Dual MicroBlaze Processor System

System Information

This system consists of one instance of MicroBlaze with external memory and commonly used peripherals such as UART, GPIO, IIC, Ethernet etc. Peripherals are



connected on a shared AXI interconnect, while DDR memory is connected on a AXI interconnect configured as a crossbar. Click Next to modify the default system. Custom boards do not have default peripherals and need to be selected on the next page.

Optimization Strategy

● Area ○ Throughput

Related Information

Vendor's Website

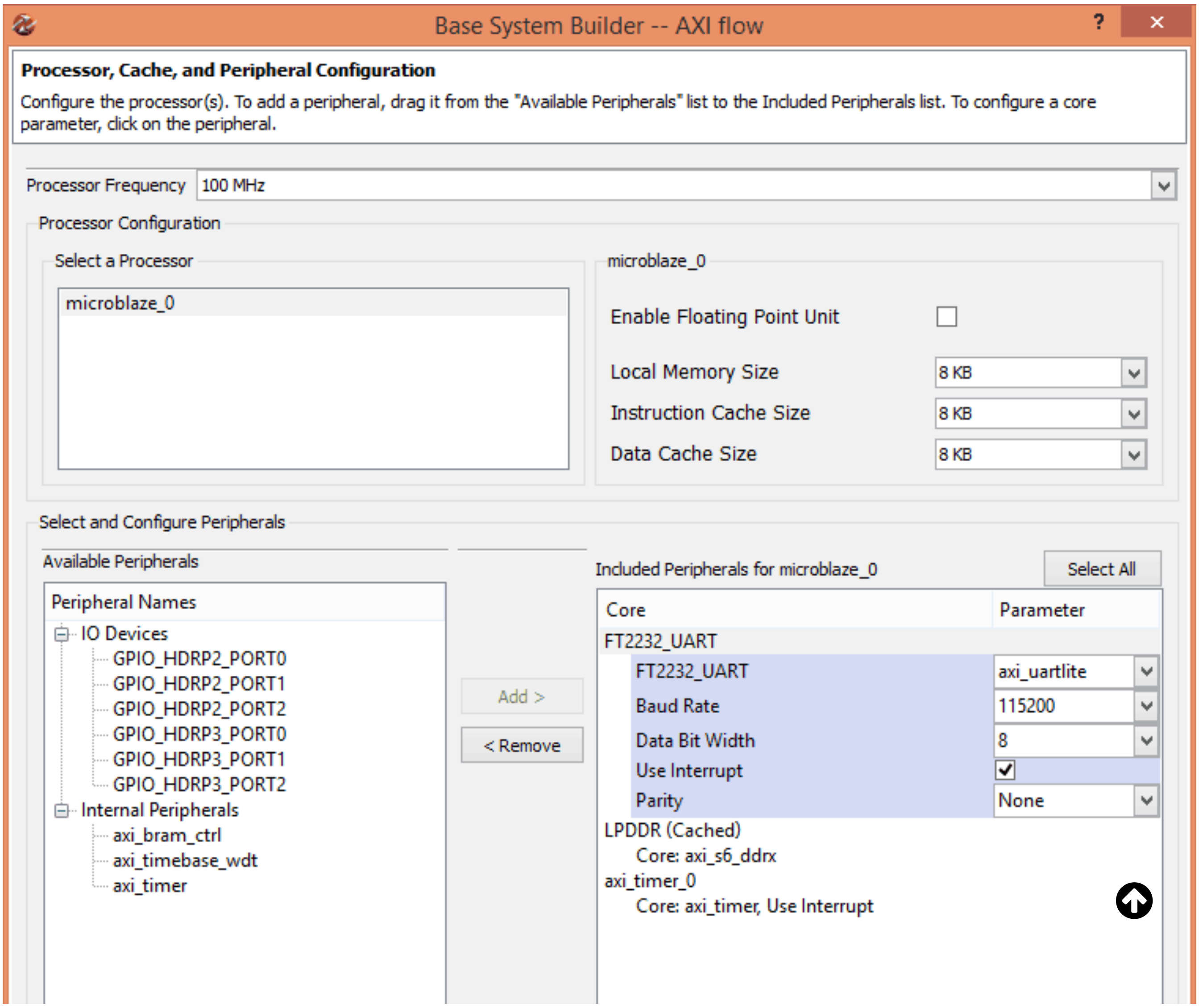
Vendor's Contact Information

Third Party Board Definition Files Download Website

Numato Lab Saturn Spartan-6 module XC6SLX45 with Micron MT6H32M16 compatible 512Mb LPDDR, 128Mb SPI Flash and FT2232H USB interface.

(https://numato.com/help/wp-content/uploads/2016/03/bsbwizard_select_board1.png)

Leave the Optimization Strategy to its default value and click next.



Processor, Cache, and Peripheral Configuration

Configure the processor(s). To add a peripheral, drag it from the "Available Peripherals" list to the Included Peripherals list. To configure a core parameter, click on the peripheral.

Processor Frequency: 100 MHz

Processor Configuration

Select a Processor: microblaze_0

microblaze_0

Enable Floating Point Unit □

Local Memory Size: 8 KB

Instruction Cache Size: 8 KB

Data Cache Size: 8 KB

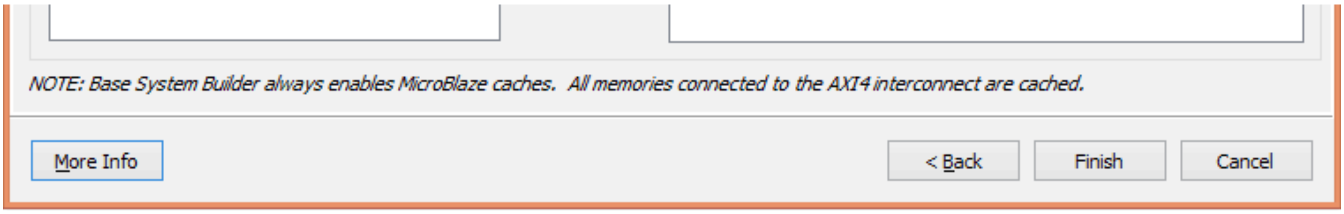
Select and Configure Peripherals

Available Peripherals

Peripheral Names
- IO Devices
  - GPIO_HDRP2_PORT0
  - GPIO_HDRP2_PORT1
  - GPIO_HDRP2_PORT2
  - GPIO_HDRP3_PORT0
  - GPIO_HDRP3_PORT1
  - GPIO_HDRP3_PORT2
- Internal Peripherals
  - axi_bram_ctrl
  - axi_timebase_wdt
  - axi_timer

Included Peripherals for microblaze_0

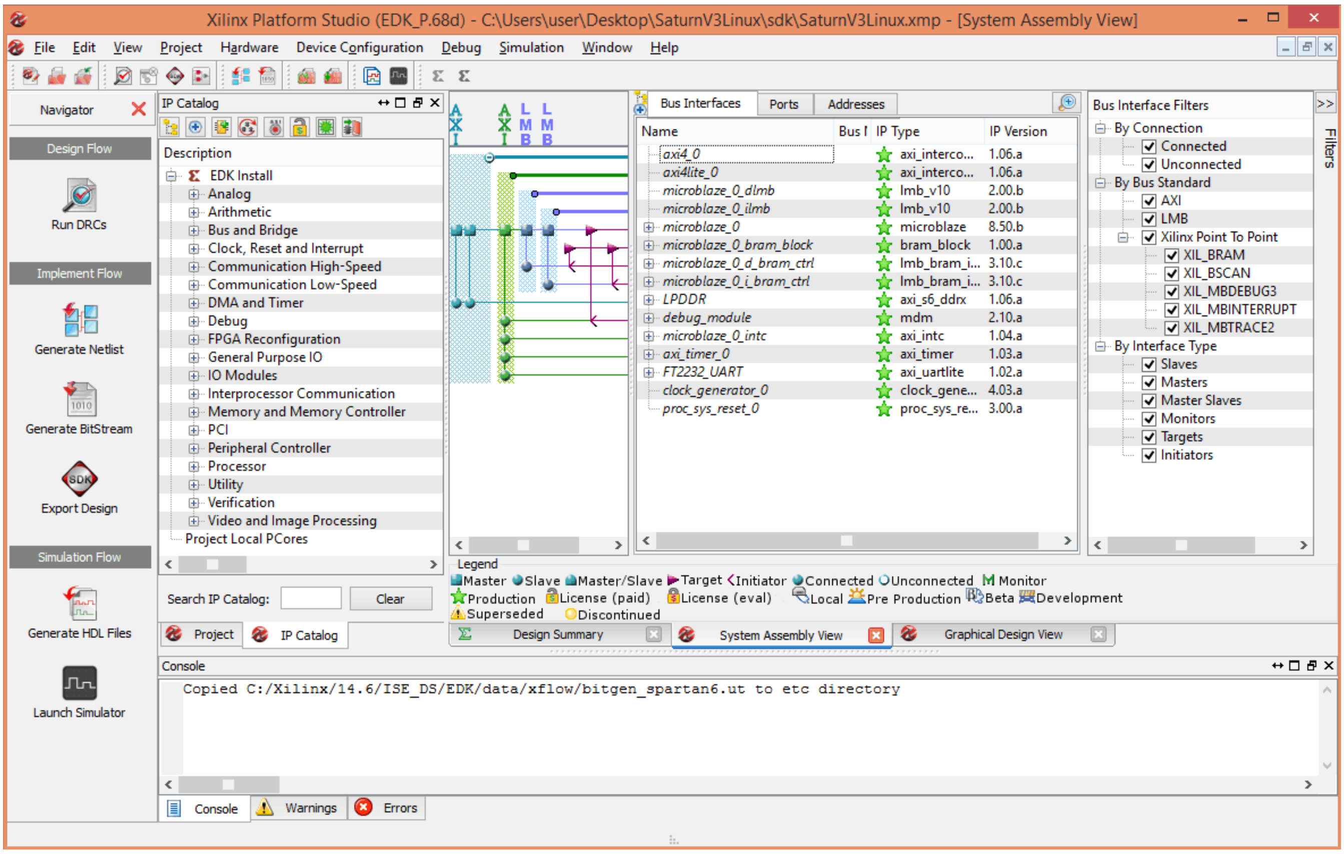
Core	Parameter
FT2232_UART	
FT2232_UART	axi_uartlite
Baud Rate	115200
Data Bit Width	8
Use Interrupt	☑
Parity	None
LPDDR (Cached)	
Core: axi_s6_ddrx	
axi_timer_0	
Core: axi_timer, Use Interrupt	



NOTE: Base System Builder always enables MicroBlaze caches. All memories connected to the AXI4 interconnect are cached.

(https://numato.com/help/wp-content/uploads/2016/03/bsbwizard_select_peripherals1.png)

By default, FT2232 UART, LPDDR and GPIO peripherals will be selected. Select and remove all GPIO peripherals since we won't use them in this project. Select FT2232_UART peripheral and set the baud rate to **115200** and check the "Use Interrupt" check box. Then select "axi_timer" in the "Available Peripherals" list and add that to the system. Select the timer and check the "Use Interrupt" check box. Make sure everything is setup as shown in the image above and click "Finish" button. Now Xilinx Platform Studio will generate a project for you based on the selections you made in the BSB wizard. If everything went well, you will be presented with a window as shown below.



Xilinx Platform Studio (EDK_P.68d) - C:\Users\user\Desktop\SaturnV3Linux\sdk\SaturnV3Linux.xmp - [System Assembly View]

Name	Bus I	IP Type	IP Version
axi4_0		axi_interco...	1.06.a
axi4lite_0		axi_interco...	1.06.a
microblaze_0_dlmb		lmb_v10	2.00.b
microblaze_0_ilmb		lmb_v10	2.00.b
microblaze_0		microblaze	8.50.b
microblaze_0_bram_block		bram_block	1.00.a
microblaze_0_d_bram_ctrl		lmb_bram_i...	3.10.c
microblaze_0_i_bram_ctrl		lmb_bram_i...	3.10.c
LPDDR		axi_s6_ddrx	1.06.a
debug_module		mdm	2.10.a
microblaze_0_intc		axi_intc	1.04.a
axi_timer_0		axi_timer	1.03.a
FT2232_UART		axi_uartlite	1.02.a
clock_generator_0		clock_gene...	4.03.a
proc_sys_reset_0		proc_sys_re...	3.00.a

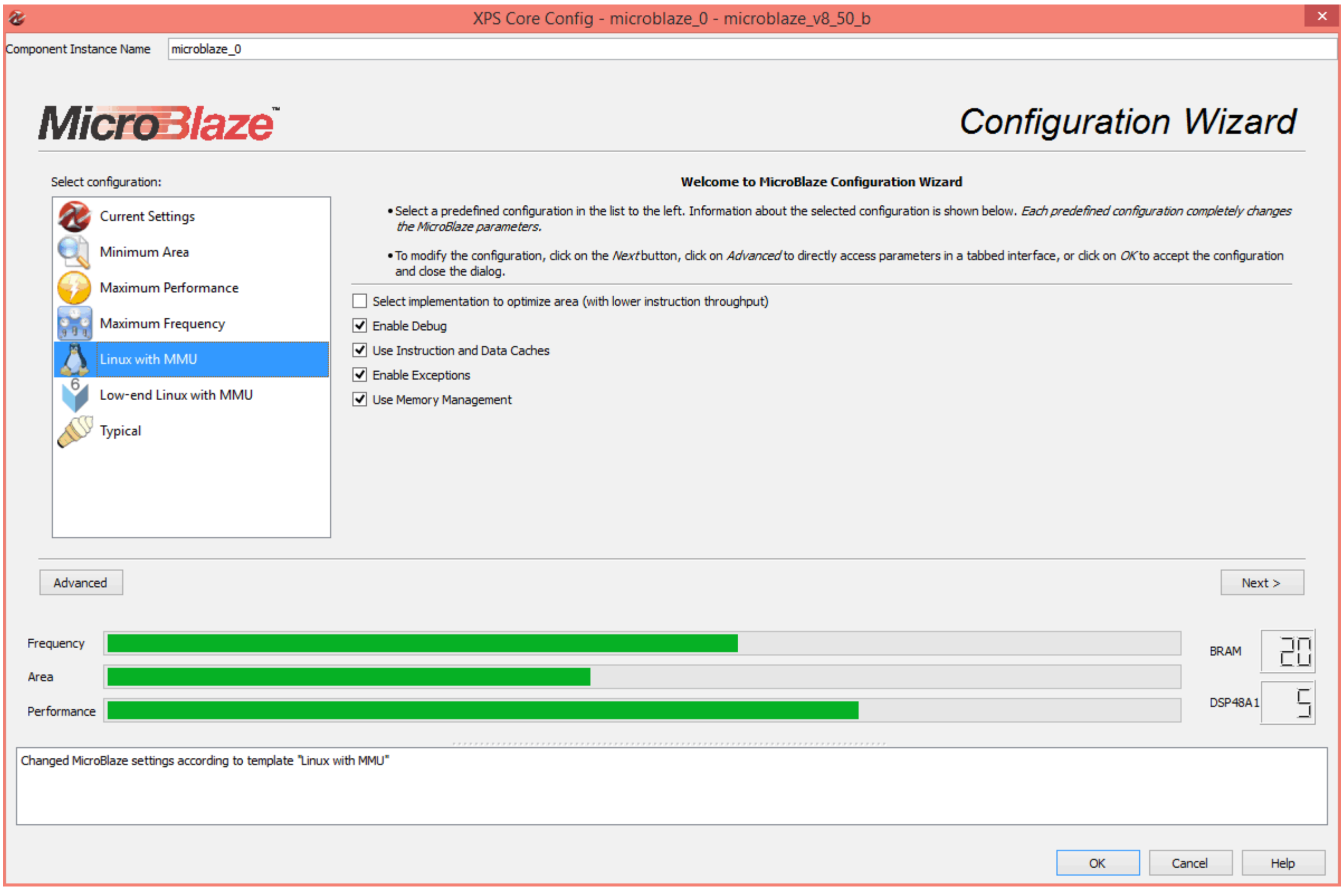
- By Connection
  - ☑ Connected
  - ☑ Unconnected
- By Bus Standard
  - ☑ AXI
  - ☑ LMB
  - ☑ Xilinx Point To Point
    - ☑ XIL_BRAM
    - ☑ XIL_BSCAN
    - ☑ XIL_MBDEBUG3
    - ☑ XIL_MBINTERRUPT
    - ☑ XIL_MBTRACE2
- By Interface Type
  - ☑ Slaves
  - ☑ Masters
  - ☑ Master Slaves
  - ☑ Monitors
  - ☑ Targets
  - ☑ Initiators

```
Copied C:/Xilinx/14.6/ISE_DS/EDK/data/xflow/bitgen_spartan6.ut to etc directory
```

(https://numato.com/help/wp-content/uploads/2016/03/xilinx_platform_studio_new_project1.png)

Now we will need to add one more peripheral IP to complete our system design. From the IP catalog on the left, drag and drop the AXI SPI Interface (under "Communication Low-Speed" category) to the "System Assembly View" and connect the IP to microblaze_0 when asked.

Now we will configure the peripheral IPs. Right click on microblaze_0 in the “System Assembly View > Bus Interfaces” tab and select “Configure IP” from the pop up menu. The Microblaze Configuration Wizard will pop up. Select “Linux with MMU” under the “Select Configuration” options (Please see image below).



XPS Core Config - microblaze_0 - microblaze_v8_50_b

Component Instance Name: microblaze_0

Welcome to MicroBlaze Configuration Wizard

- Select a predefined configuration in the list to the left. Information about the selected configuration is shown below. *Each predefined configuration completely changes the MicroBlaze parameters.*
- To modify the configuration, click on the *Next* button, click on *Advanced* to directly access parameters in a tabbed interface, or click on *OK* to accept the configuration and close the dialog.

Select configuration: Linux with MMU

☐ Select implementation to optimize area (with lower instruction throughput)
☑ Enable Debug
☑ Use Instruction and Data Caches
☑ Enable Exceptions
☑ Use Memory Management

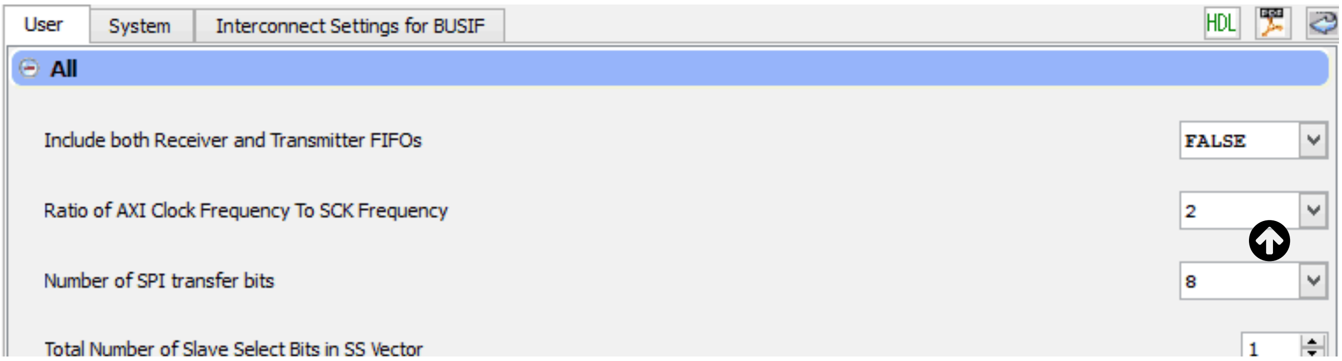
BRAM 20
DSP48A1 5

Changed MicroBlaze settings according to template "Linux with MMU"

(https://numato.com/help/wp-content/uploads/2016/03/microblaze_configuration_wizard1.png)

Click OK to accept the new settings. You can click on the “Advanced” button to tweak Microblaze IP settings further (optional).

Next right click on axi_spi_0 peripheral and select “Configure IP” from the popup menu. Disable Input/Output FIFO and set the clock frequency ratio to 2. Leave all other settings to defaults and click OK. Please see the image below for reference.

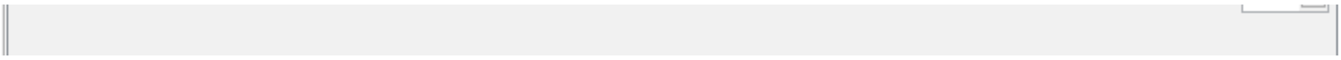


Include both Receiver and Transmitter FIFOs: FALSE

Ratio of AXI Clock Frequency To SCK Frequency: 2

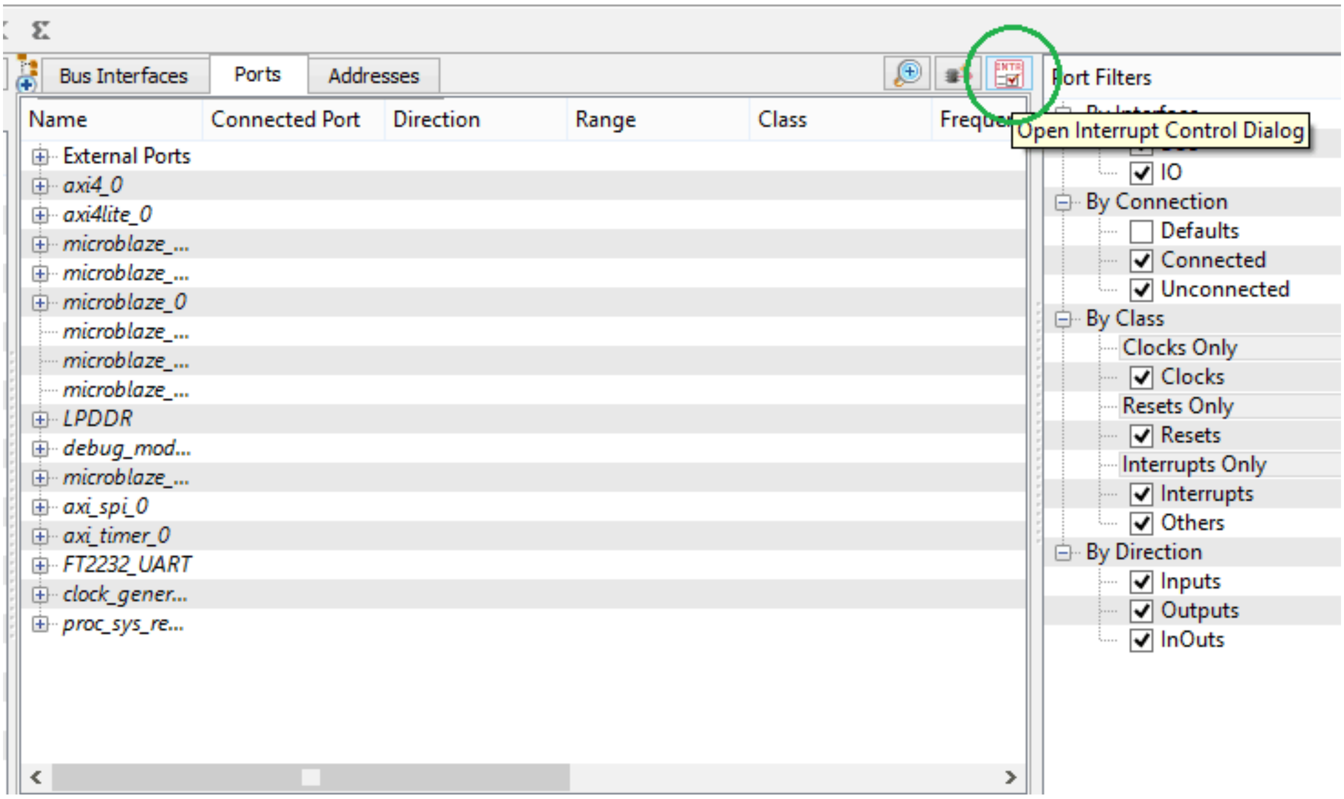
Number of SPI transfer bits: 8

Total Number of Slave Select Bits in SS Vector: 1



(https://numato.com/help/wp-content/uploads/2016/03/spi_configuration1.png)

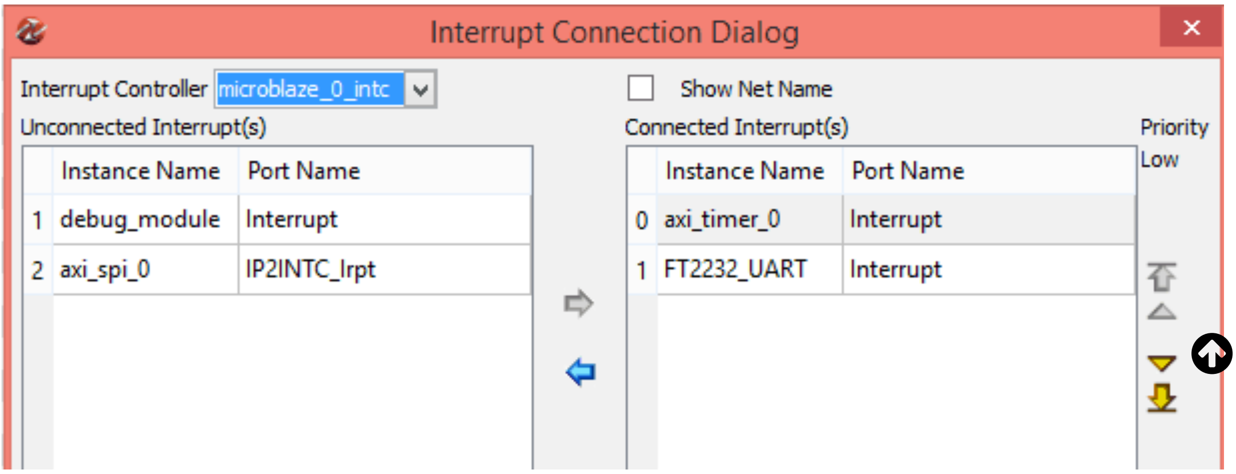
At this point, we have finished configuring Microblaze CPU and all peripherals. The next step is to route all peripheral interrupts to the microblaze CPU through the interrupt controller. To configure interrupts, go to the “Ports” tab in “System Assembly View” and locate the “Open interrupt Control Dialog” button. This button is a little tricky to locate. Please see the image below for help.



Name	Connected Port	Direction	Range	Class	Freque
External Ports					
axi4_0					
axi4lite_0					
microblaze_...					
microblaze_...					
microblaze_0					
microblaze_...					
microblaze_...					
microblaze_...					
LPDDR					
debug_mod...					
microblaze_...					
axi_spi_0					
axi_timer_0					
FT2232_UART					
clock_gener...					
proc_sys_re...					

(https://numato.com/help/wp-content/uploads/2016/03/interrupt_control_dialog1.png)

In the interrupt control dialog box, make sure that both FT2232_UART and axi_timer_0 are connected to the interrupt controller and debug_module and axi_spi_0 are not connected to the interrupt controller as shown in the image below.



Interrupt Controller: microblaze_0_intc

□ Show Net Name

Unconnected Interrupt(s)

	Instance Name	Port Name
1	debug_module	Interrupt
2	axi_spi_0	IP2INTC_Irpt

Connected Interrupt(s)

	Instance Name	Port Name
0	axi_timer_0	Interrupt
1	FT2232_UART	Interrupt

(https://numato.com/help/wp-content/uploads/2016/03/interrupt_settings1.png)

Now we need to add a few lines to the user Constraints File to map FPGA IOs for accessing SPI Flash. Click on the “project” tab next to “IP Catalog” tab and double click on the UCF file “SaturnV3Linux.ucf” to open it. Add the following lines to the end of the UCF file and save.

```
NET axi_spi_0_SCK_pin LOC = "R15"   |  IOSTANDARD = "LVCMOS33";
NET axi_spi_0_SS_pin LOC = "V3"     |  IOSTANDARD = "LVCMOS33";
NET axi_spi_0_MOSI_pin LOC = "T13"  |  IOSTANDARD = "LVCMOS33";
NET axi_spi_0_MISO_pin LOC = "R13"  |  IOSTANDARD = "LVCMOS33";
```

Go back to the “System Assembly View” and select “Addresses” tab and make sure all peripherals has a base address assigned (XPS will take care of this automatically, but just in case). Note the base address of LPDDR memory, we will need it when building Linux kernel. At this point, we are ready to build the project. Click on the “generate bitstream” button the left side pane to start build process. If build fails due to timing closure issues, go to project menu in XPS, select Project Options and uncheck “Treat timing failures as errors”. This should help the build complete without errors.

Read Part II of this article series here. (https://numato.com/kb/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-ii/)

(https://twitter.com/home?status=https://numato.com/kb/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-i/) (https://www.facebook.com/sharer/sharer.php?u=https://numato.com/kb/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-i/) (https://pinterest.com/pin/create/button/?url=https://numato.com/kb/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-i/&media=&description=) (https://plus.google.com/share?url=https://numato.com/kb/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-i/) (https://www.linkedin.com/shareArticle?mini=true&url=https://numato.com

/kb/saturn-microblaze-and-linux-how-run-linux-saturn-spartan-6-fpga-module-part-i/)

Was this helpful?

👍 10 Yes

👎 1 No

Leave A Comment

Comment

Name *

Email *

Website

POST COMMENT

Knowledge Base Category

Callisto Kintex 7 USB 3.1 FPGA Module (https://numato.com/kb-category/callisto-kintex-7-usb-3-1-fpga-module/) (2)

Embedded Linux (https://numato.com/kb-category/embedded-linux/) (12)

FPGA Applications (https://numato.com/kb-category/fpga-applications/) (13)

Galatea PCI Express Spartan 6 FPGA Module (https://numato.com/kb-category/galatea-pci-express-spartan-6-fpga-module/) (4)

Getting Started With FPGA (https://numato.com/kb-category/getting-started-with-fpga/) (32)

Intercore SDK Framework (https://numato.com/kb-category/intercore-sdk-framework/) (1)

Mimas A7 Mini FPGA Development Board (https://numato.com/kb-category/mimas-a7-mini-fpga-development-board/) (3)

Mimas Artix 7 FPGA Development Board (https://numato.com/kb-category/mimas-artix-7-fpga-development-board/) (10)

Narvi Spartan 7 FPGA Module (https://numato.com/kb-category/narvi-spartan-7-fpga-module/) (2)

Nereid Kintex 7 PCI Express FPGA Board (https://numato.com/kb-category/nereid-kintex-7-pci-express-fpga-board/) (3)

Neso Artix 7 FPGA Module (https://numato.com/kb-category/neso-artix-7-fpga-module/) (10)

Opsis: FPGA-based open video platform (https://numato.com/kb-category/opsis-fpga-based-open-video-platform/) (4)

Prodigy Series Automation Devices (https://numato.com/kb-category/prodigy-series-automation-devices/) (1)

Proteus Kintex 7 FPGA Development Module (https://numato.com/kb-category/proteus-kintex-7-fpga-development-module/) (1)

Quick Start Guides (https://numato.com/kb-category/quick-start-guides/) (2)

Rhea Device Management Tool (https://numato.com/kb-category/rhea-device-management-tool/) (2)

Saturn Spartan 6 FPGA Module (https://numato.com/kb-category/saturn-spartan-6-fpga-module/) (7)

Skoll Kintex 7 FPGA Module (https://numato.com/kb-category/skoll-kintex-7-fpga-module/) (6)

Styx Xilinx Zynq FPGA Module (https://numato.com/kb-category/styx-xilinx-zynq-fpga-module/) (8)

Tagus – Artix 7 PCI Express Development Board (https://numato.com/kb-category/tagus-artix-7-pci-express-development-board/) (1)

Telesto MAX 10 FPGA Module (https://numato.com/kb-category/telesto-max-10-fpga-module/) (5)

Tenagra FPGA System Management Software (https://numato.com/kb-category/tenagra-fpga-system-management-software/) (3)

Theia Android Application (https://numato.com/kb-category/theia-android-application/) (1)

USB GPIO Modules (https://numato.com/kb-category/usb-gpio-modules/) (2)

USB Relay Modules (https://numato.com/kb-category/usb-relay-modules/) (1)

Vivado Design Suit (https://numato.com/kb-category/vivado-design-suit/) (4)

Waxwing Spartan 6 FPGA Development Board (https://numato.com/kb-category/waxwing-spartan-6-fpga-development-board/) (3)

White Papers (https://numato.com/kb-category/white-papers/) (1)

Working With Xilinx EDK (https://numato.com/kb-category/working-with-xilinx-edk/) (1)

XO-Bus Framework (https://numato.com/kb-category/xo-bus/) (2)

Don't miss out on new articles! Subscribe to get valuable insights.

email address

Subscribe

(https://twitter.com/numatolab)

(https://www.facebook.com/numato/)

(https://www.youtube.com/user/NumatoLab)

(https://plus.google.com/+Numatosystems)