Building a RIFFA 2.0 design with ISE: Xilinx Virtex-6 ML605

Matt Jacobsen (mdjacobs@cs.ucsd.edu)

This is a step by step guide to building a RIFFA 2.0 reference design for a Xilinx Virtex-6 ML605 development board using ISE. Though it is likely that this guide will work for other Virtex-6 based FPGA development boards.

RIFFA 2.0 provides a simple to use interface for communicating between a workstation and FPGA cores. It uses a Xilinx PCIe Endpoint IP core to drive the transceivers. The PCIe Endpoint core for Spartan 6 FPGAs is the *Virtex 6 Integrated Block for PCI Express*. This core is licensed by the Xilinx End User License Agreement and is provided with the Xilinx ISE Design suite with no additional charge. A prebuilt design is provided and ready for download to your Xilinx ML605 board. Building your own RIFFA 2.0 design requires generating the PCIe Endpoint core and then merging it with the RIFFA 2.0 source HDL.

To create a RIFFA 2.0 design with ISE:
1. Use Xilinx Coregen to generate the PCIe Endpoint core.
2. Combine the PCIe Endpoint core's source HDL with the RIFFA 2.0 HDL.
3. Create a project in Xilinx ISE with the combined source HDL and `.ucf`.
4. Synthesize and implement.

Detailed instructions on how to do each step follow.

1. Use Xilinx Coregen to generate the PCIe Endpoint core.

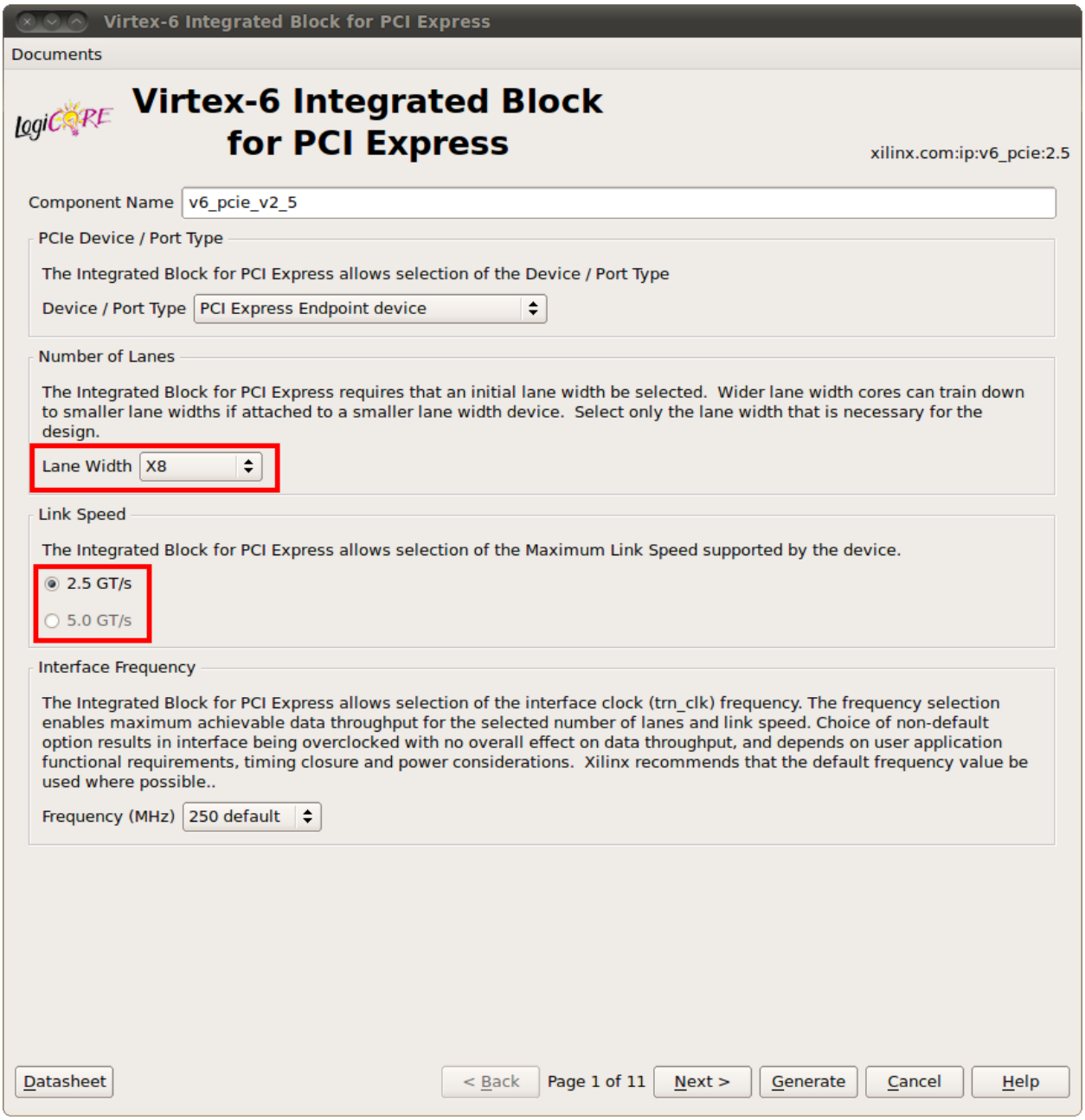
Use Coregen to generate Verilog source for the *Virtex 6 Integrated Block for PCI Express ver. 2.5*. This is the latest production version of the core at the time of this writing.

Start Coregen and make sure to set the project settings to generate Verilog code for the XC6VLX240t-1FFG1156. Use the Coregen wizard to generate the core. Unless otherwise described, the default values on each wizard screen should be left as they are presented.

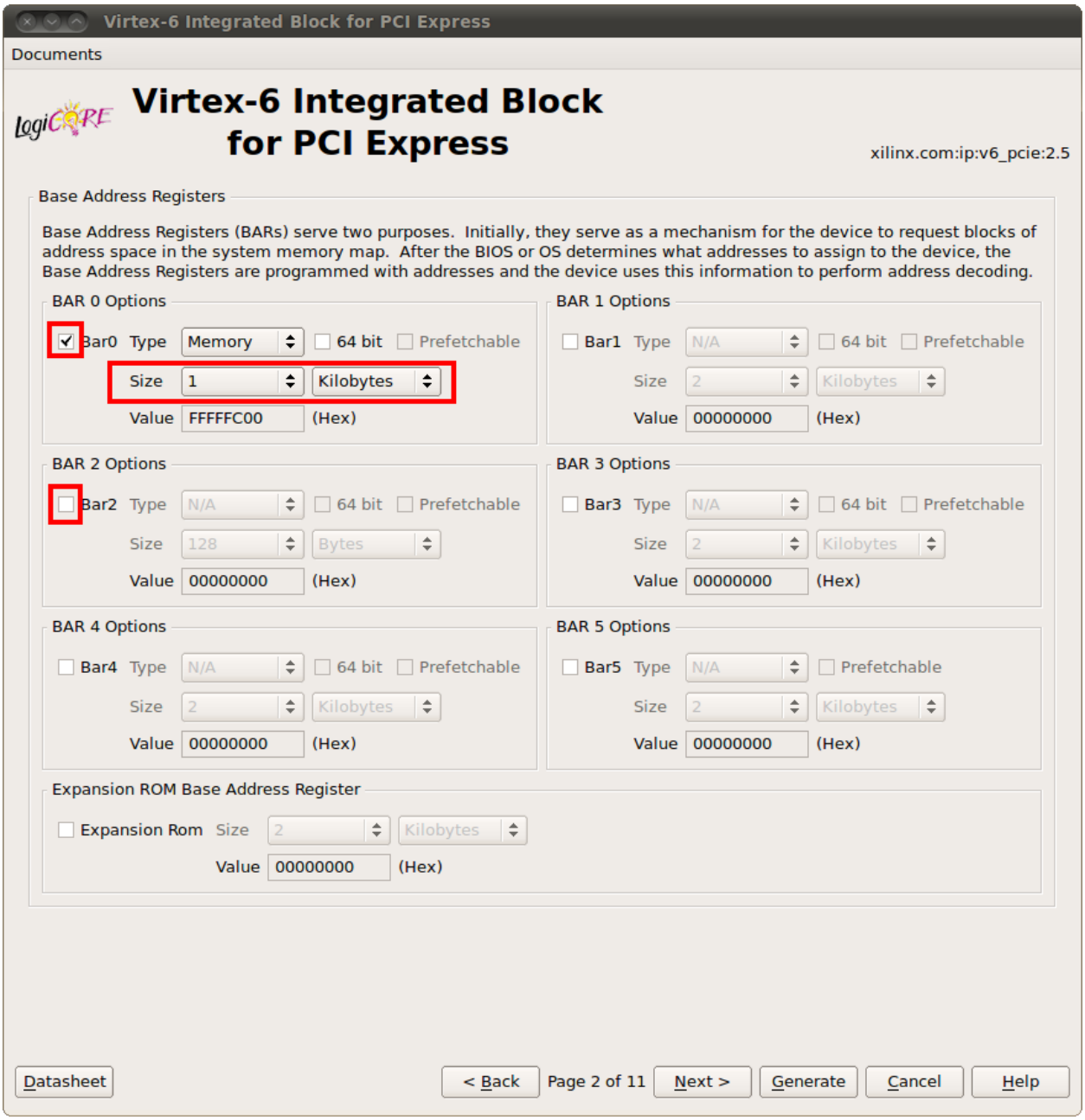
On the first screen, select the desired lane width and link speed. RIFFA 2.0 supports a 32, 64, and 128 bit interface. So any lane width and link speed selection you make will be supported. You can use any interface frequency the options allow. We keep the default component name. Below are maximum theoretical bandwidths for PCIe 1.0 and PCIe 2.0:

Gen1 (2.5 GT/s):
- x1 = 250 MB/s
- x2 = 500 MB/s
- x4 = 1000 MB/s
- x8 = 2000 MB/s

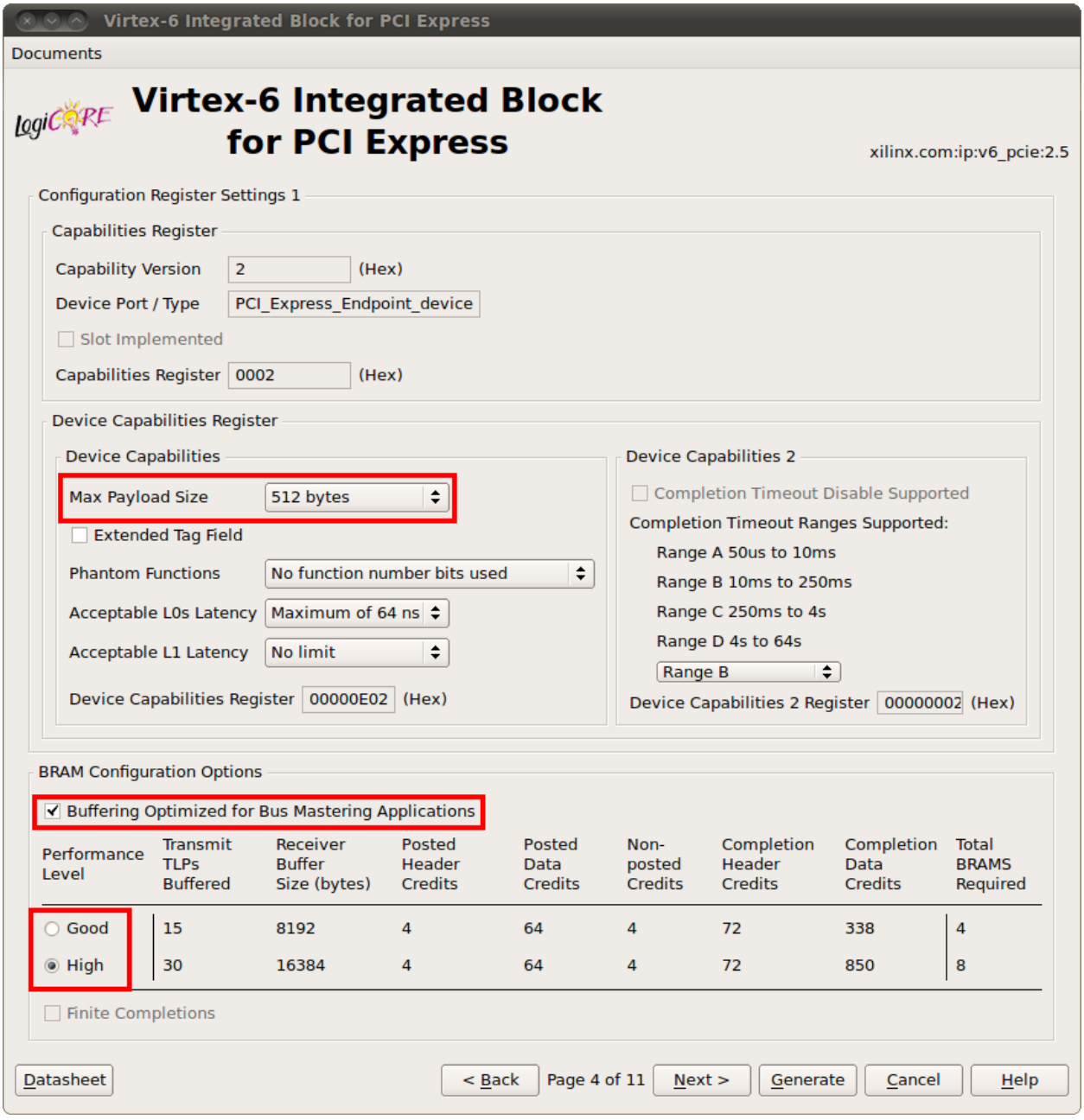
Gen2 (5.0 GT/s):
- x1 = 500 MB/s
- x2 = 1000 MB/s
- x4 = 2000 MB/s
- x8 = 4000 MB/s



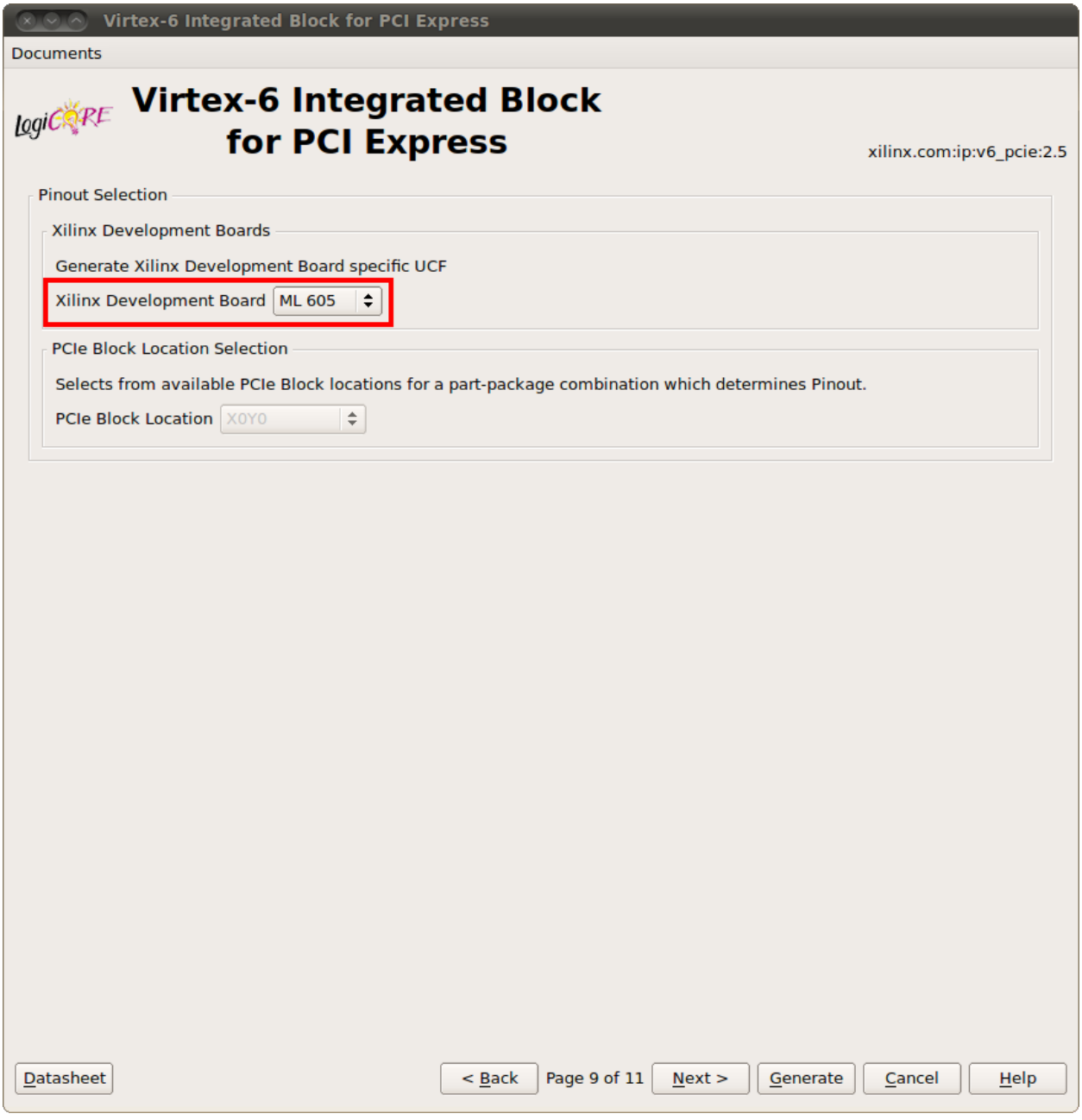
On this screen, make sure only Bar0 is selected and is set to a size of 1 KB. You will need to deselect Bar2.



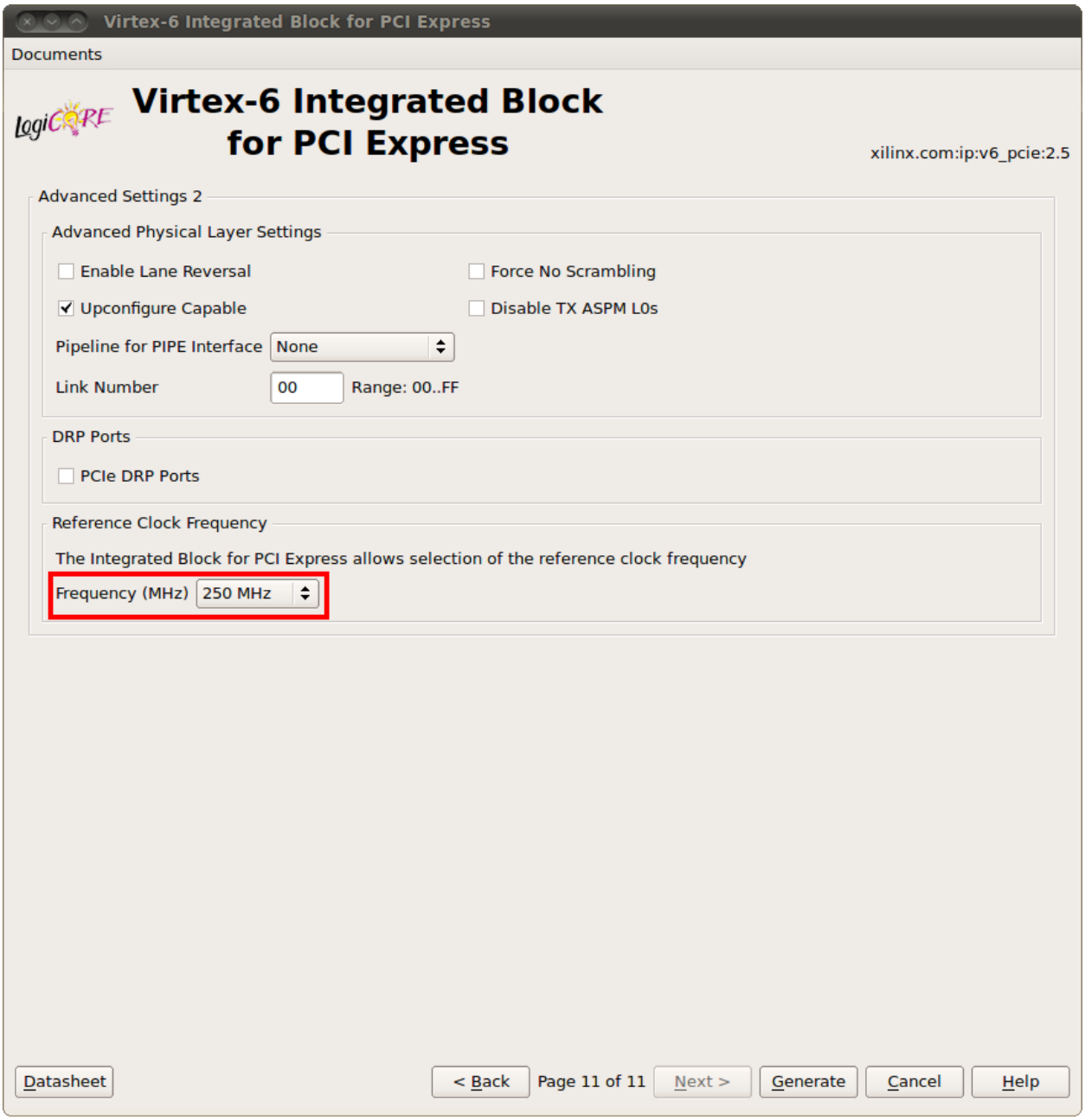
On this screen, select Buffering Optimized for Bus Mastering Applications and Performance Level High. Additionally, set the Max Payload Size to the maximum value offered. These changes are not necessary for RIFFA 2.0 to function. They are required to achieve maximum performance.



On this screen, select the development board that you are using. If your board is not in the list, you'll need to know the PCIe Block location for your part-package combination. Additional modifications to the generated `.ucf` may also be necessary if your board is not in the list.

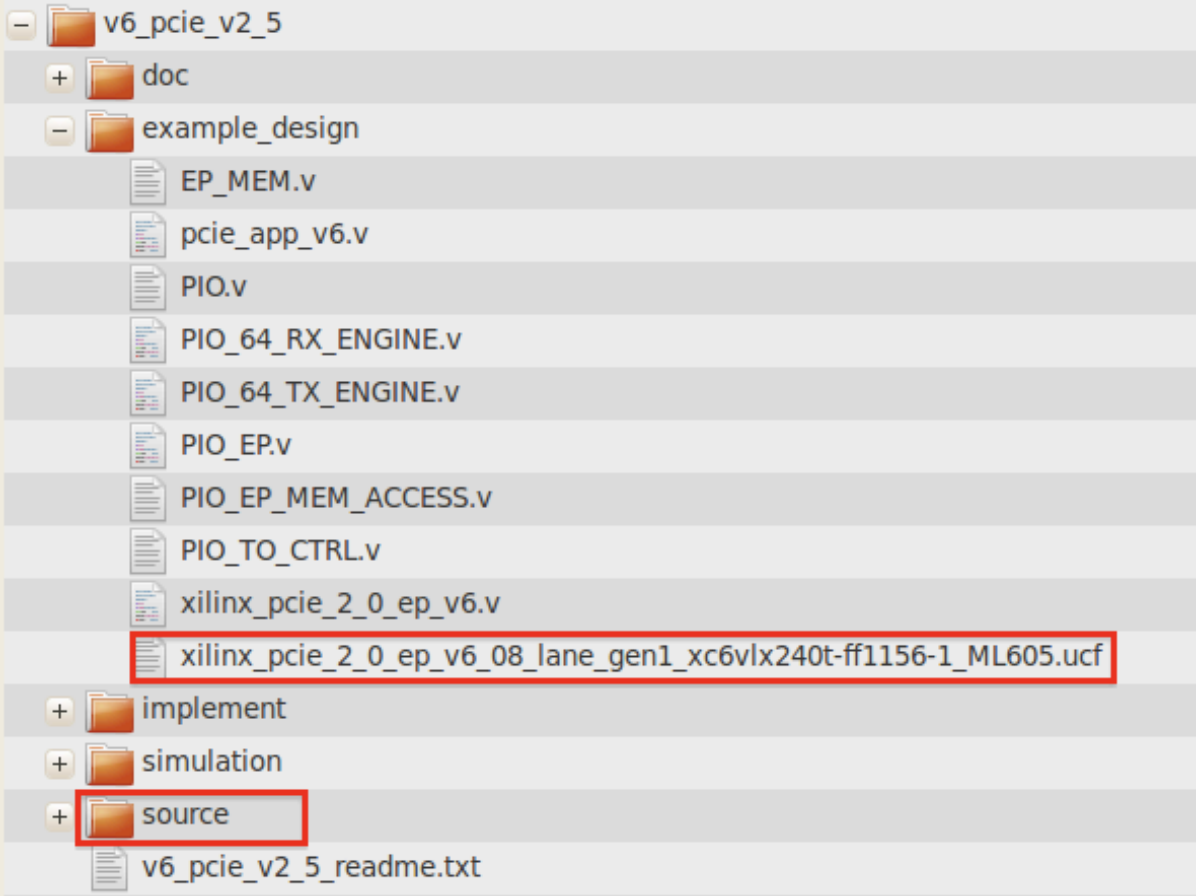


On this screen, set the Reference Clock Frequency to 250 MHz. This is the recommended setting from Xilinx document XTP044. Then complete the wizard and generate the core.



2. Combine the PCIe Endpoint core's source HDL with the RIFFA 2.0 HDL.

Coregen will produce a directory structure similar to what is pictured below. Once completed, combine all the source HDL files from the `source` directory with the RIFFA 2.0 HDL files from the distribution into a new directory of your choosing. Also, into this new directory, copy the top level and adapter module HDL files for this board from the RIFFA 2.0 distribution. Lastly use the `.ucf` file from `example_design` directory. Do not use the top level module from the `example_design` directory as it may not have the necessary modifications to work correctly.



3. Create a project in ISE with the combined HDL and `.ucf`.

I expect you know how to create a new project in ISE. So I won't provide step by step instructions.

4. Synthesize and implement.

At this point, if you attempt to synthesize you'll encounter an error: `C_NUM_CHNL` is not defined. This is intentional. It is done to get you to open the RIFFA 2.0 adapter module file and edit it as needed. There are instructions in that file. However, all you need to do is uncomment the line that defines the `C_NUM_CHNL` parameter, set it to the number of channels you need (1-12), and you should be able to implement the design completely.

The adapter module instantiates a `chnl_tester` module for each channel. Sample user application software in the RIFFA 2.0 distribution can be used to send and receive data to/from the `chnl_tester` modules. The `chnl_tester` is meant to be an example. Your design will need to replace the `chnl_tester` modules with your own modules. You may also need to modify the `.ucf`, top level, and RIFFA adapter modules to bring in additional signals as dictated by your design.

That's it for the HDL design. See the RIFFA website to setup the driver and get started with software programming.