GVI KXZ7C01 Zynq Connectivity Kit

PCIe x8 Gen2 DMA Demo, ver. 2.2.2

QuickStart

December 2017

Order Hardware:

https://detail.tmall.com/item.htm?id=562769965498

http://www.gvi-tech.com/products-fpga-kits

杭州言曼科技有限公司

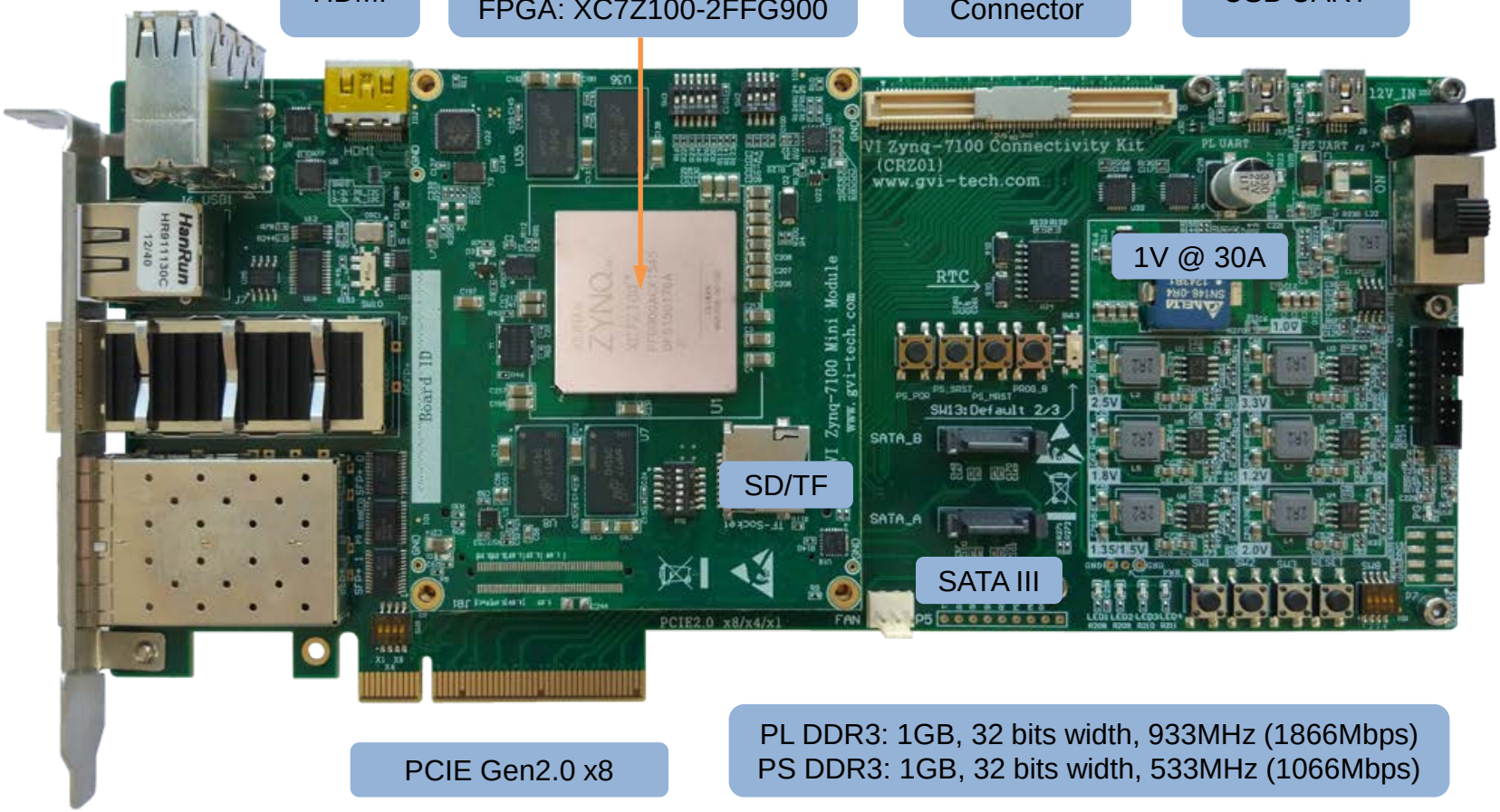
Overview

- Zynq XC7Z100 PCIe Capability
- GVI KXZ7C01 Zynq Connectivity Kit
- Software Requirements
- Design Architecture
- Hardware Setup
- Running the PCIe x8 Gen2 DMA Demo
- References

Zynq XC7Z100 PCIe Capability

- Integrated Block for PCI Express
  - PCI Express Base 2.0 Specification (5.0 Gb/s), compatible with PCIe Gen 1.1 and Gen 1.0.
  - KXZ7C01 supports x1, x4 or x8 Gen 1 and Gen 2
- Configurable for Configurable for Endpoint or Root Port Applications
  - KXZ7C01 is configured for Endpoint Applications
- GTX Transceivers implement a fully compliant PHY
- Configurable BAR spaces
  - Up to 6 x 32 bit, 3 x 64 bit, or a combination
  - Memory or IO
  - BAR and ID filtering
- Management and Statistics Interface

GVI Zynq Connectivity Kit – KXZ7C01



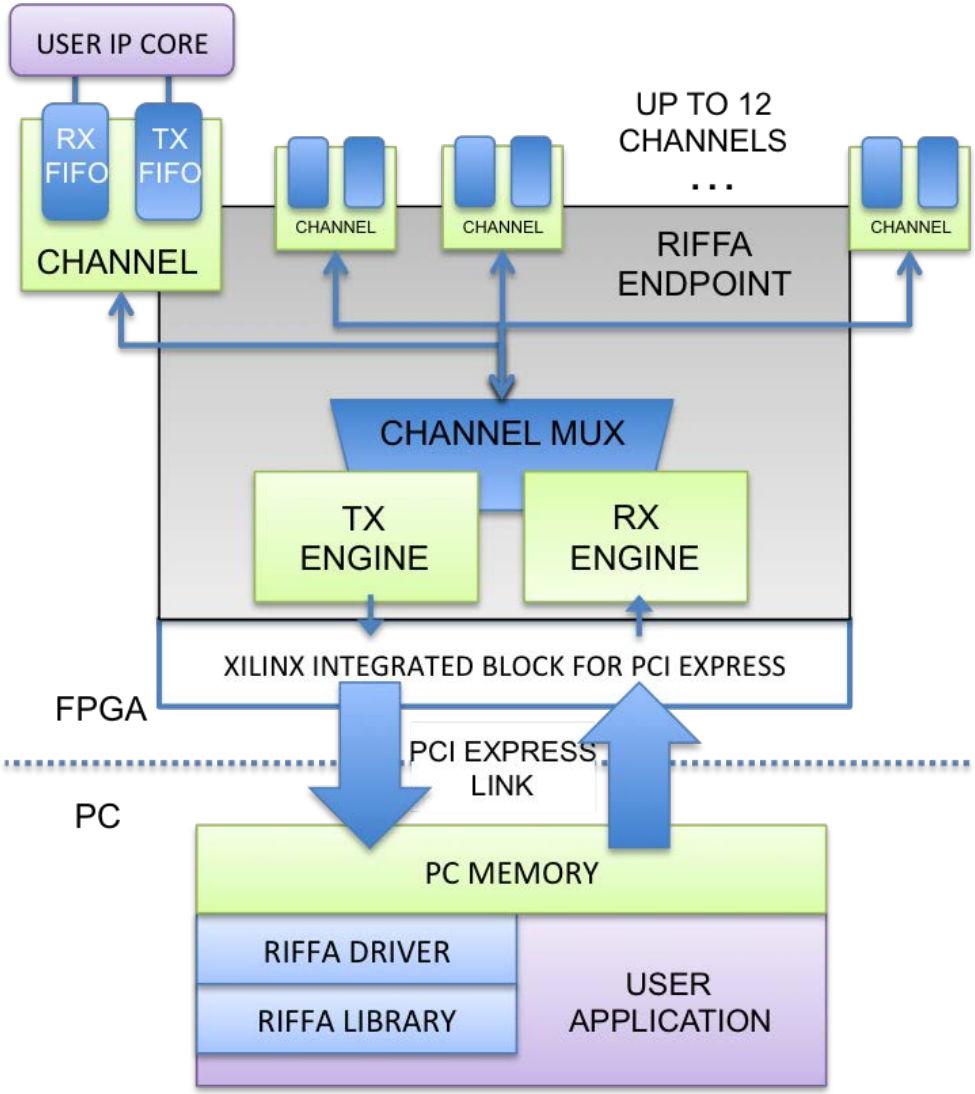
Xilinx Software Requirement

- Windows 7 x64 is used as Operating-System for this PCIe DMA demo.
- Xilinx Vivado Design Suite (version 2016.4 is used for development)

RIFFA

- This demo is based mainly on Xilinx PCIe Endpoint IP core and RIFFA 2.2.2.
- RIFFA
  - RIFFA (Reusable Integration Framework for FPGA Accelerators) is a simple framework for communicating data from a host CPU to a FPGA via a PCI Express bus.
  - RIFFA supports Windows and Linux, Altera and Xilinx, with bindings for C/C++, Python, MATLAB and Java.
  - RIFFA communicates data using direct memory access (DMA) transfers and interrupt signaling.

Architecture

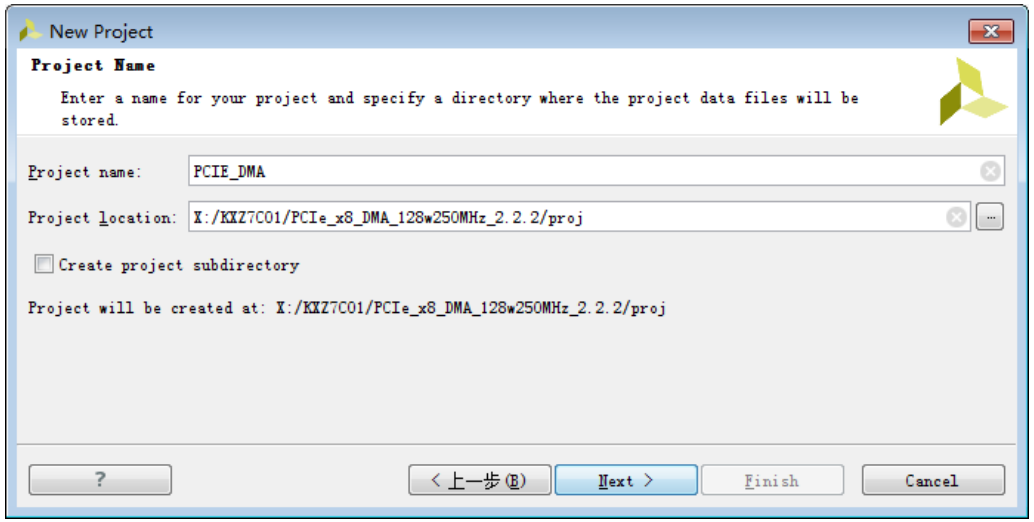


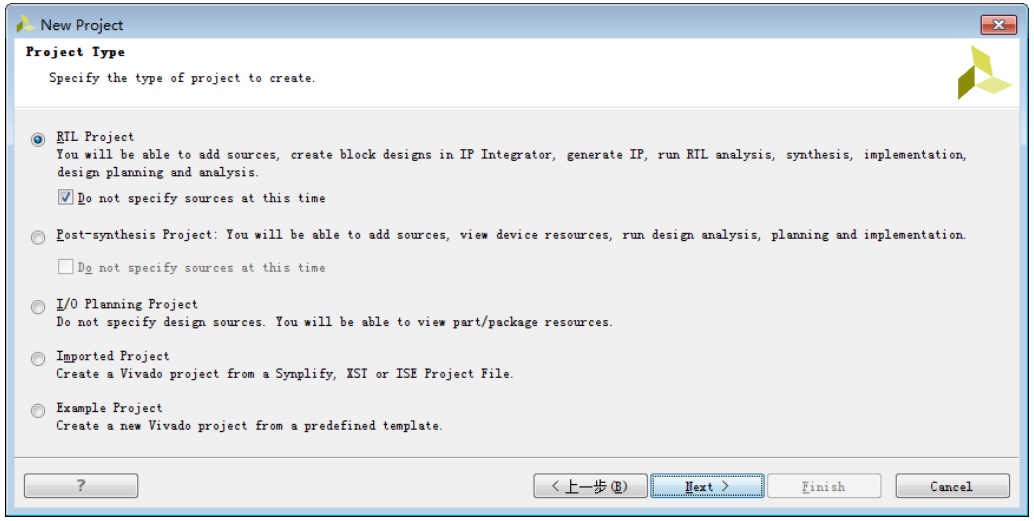
Create Your Own Design

- When you receive the KXZ7C01 Kit, the PCIe_DMA Targeted Reference Design (TRD) is already programed into the on board QSPI flash. So it is already ready to run.
- If you only want to test the PCIe functionality with the default configuration, please **jump to page 58**.
- The following slides (page 9 to 57) show how to creat the PCIe_DMA design from scratch.

Create Vivado Project

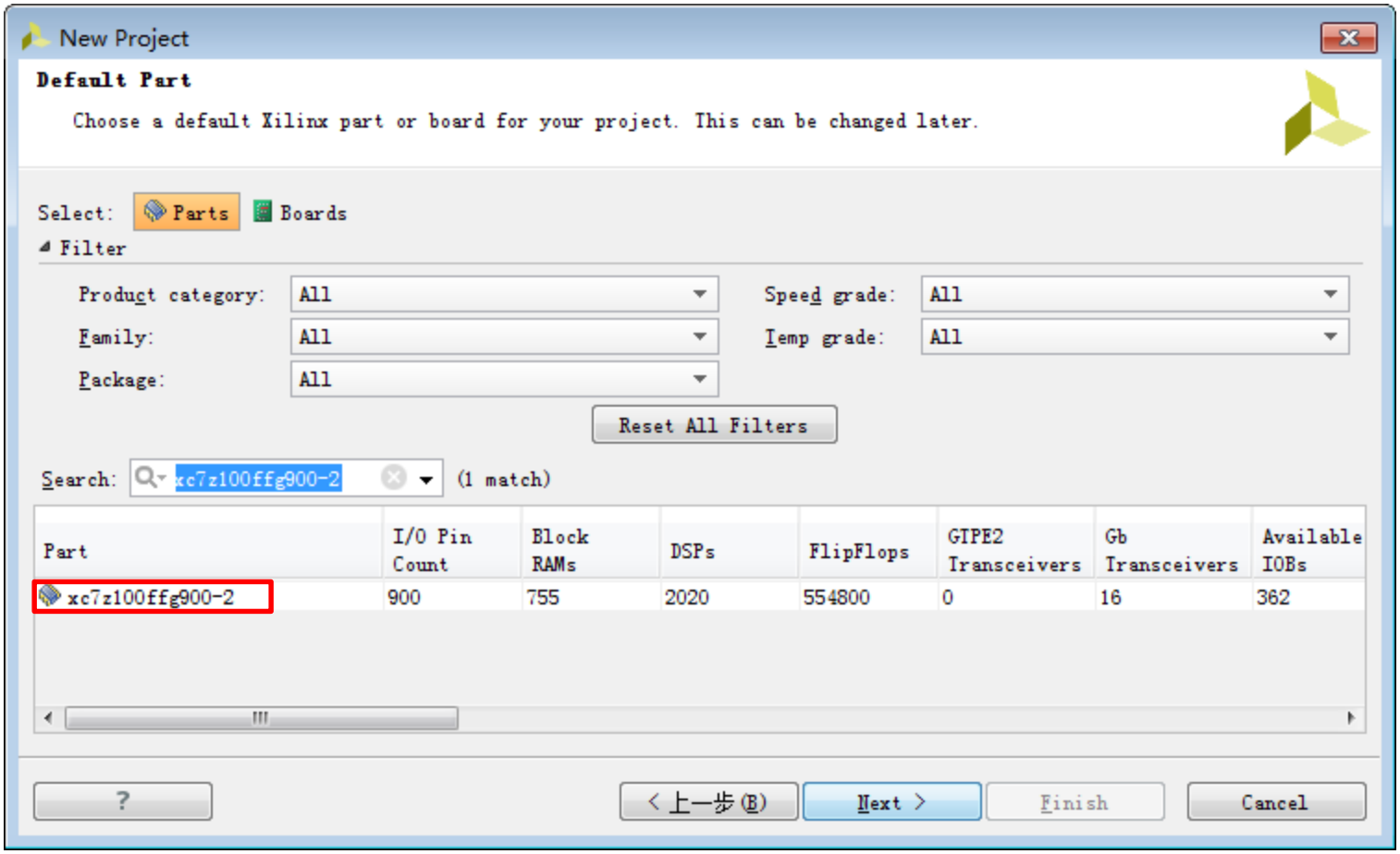
- Create a new FPGA project





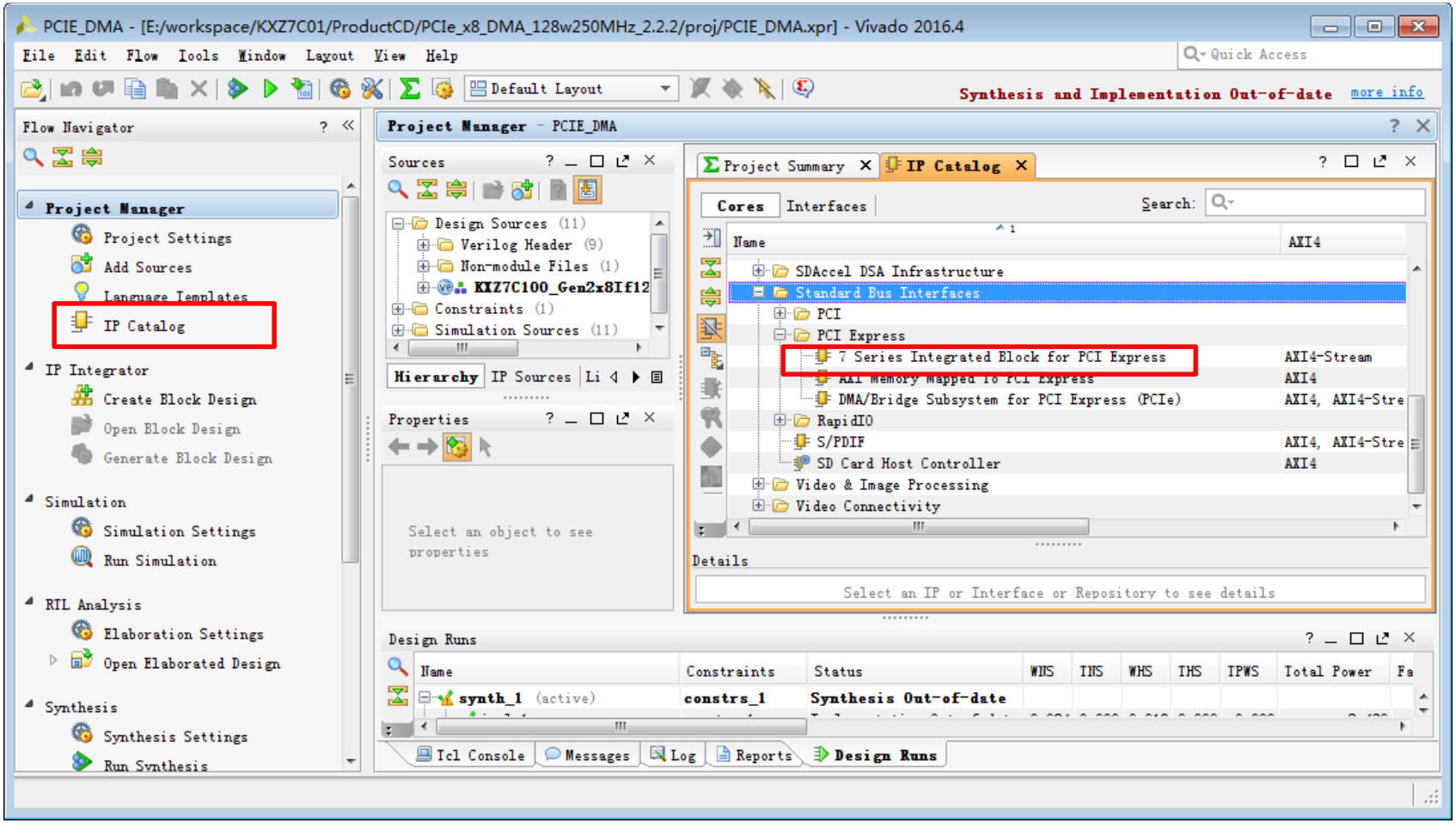
Create Vivado Project

- Select FPGA Part: xc7z100ffg900-2



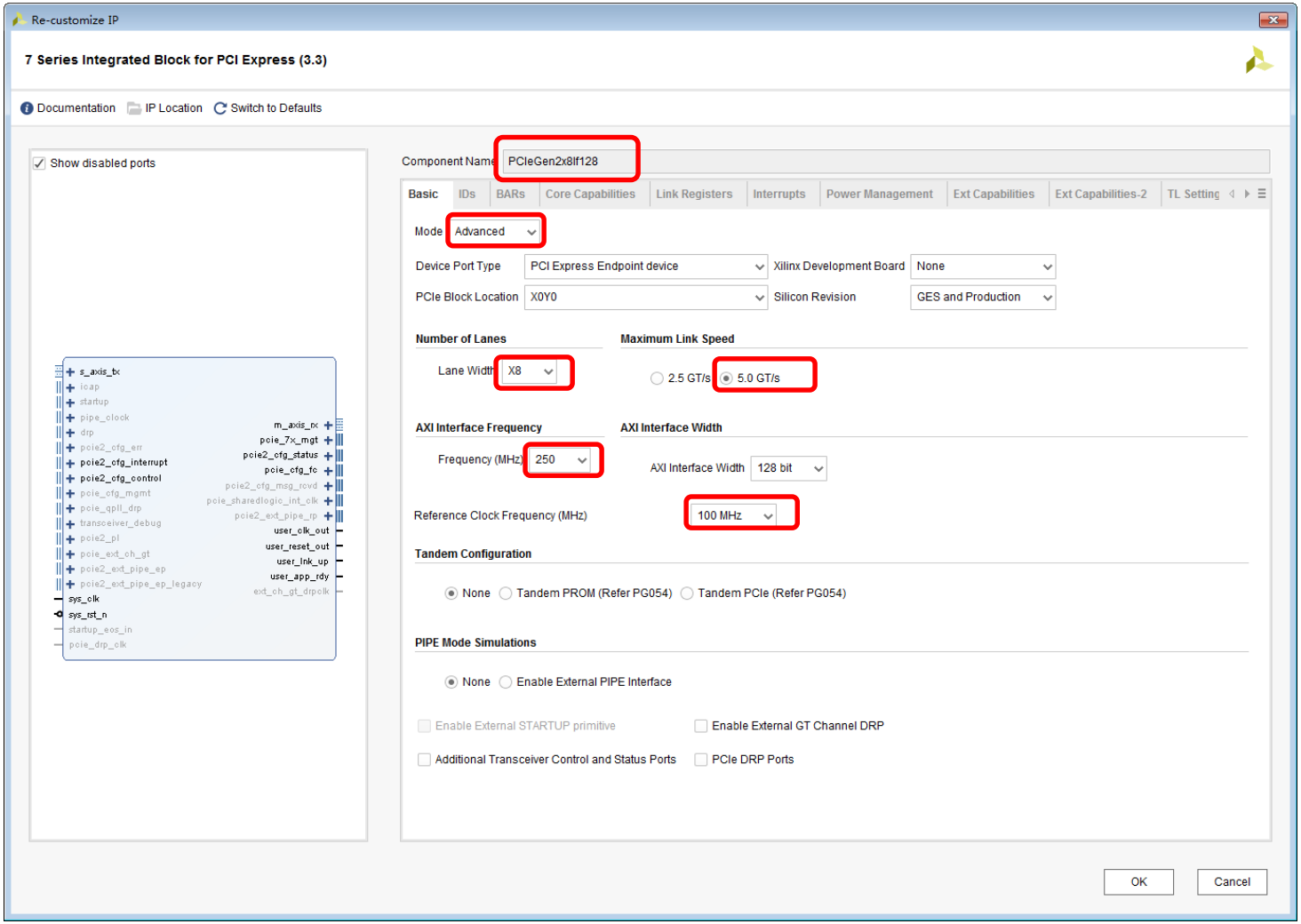
Generate PCIe Core

- Select IP Catalog and double click on the 7 Series Integrated Block for PCI Express core.



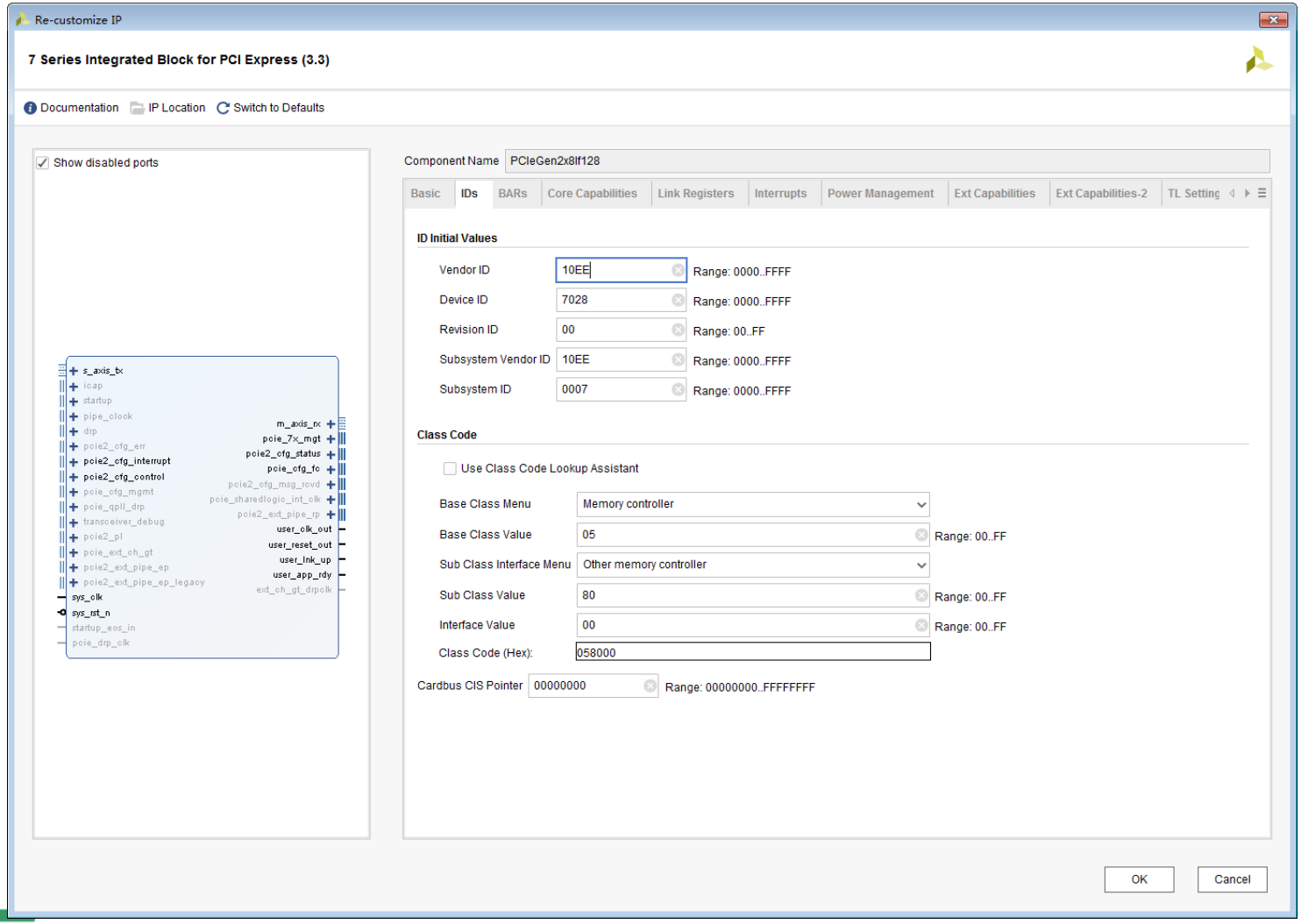
Generate PCIe Core

- On the first screen, set the desired lane width and link speed.



Generate PCIe Core

- On the second screen, use the default settings.



Generate PCIe Core

- On this screen, make sure only Bar0 is selected and is set to a size of 1 KB.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

Show disabled ports

Component Name: PCIeGen2x8If128

Basic | IDs | BARs | Core Capabilities | Link Registers | Interrupts | Power Management | Ext Capabilities | Ext Capabilities-2 | TL Setting

Base Address Registers (BARs) serve two purposes. Initially, they serve as a mechanism for the device to request blocks of address space in the system memory map. After the BIOS or OS determines what addresses to assign to the device, the Base Address Registers are programmed with addresses and the device uses this information to perform address decoding.

Bar0 Enabled
- Type: Memory | 64 bit | Prefetchable
- Size Unit: Kilobytes | Size Value: 1
- Value (Hex): FFFFFC00

Bar1 Enabled
- Type: N/A | 64 bit | Prefetchable
- Size Unit: Kilobytes | Size Value: 2
- Value (Hex): 00000000

Bar2 Enabled
- Type: N/A | 64 bit | Prefetchable
- Size Unit: Kilobytes | Size Value: 2
- Value (Hex): 00000000

Bar3 Enabled
- Type: N/A | 64 bit | Prefetchable
- Size Unit: Kilobytes | Size Value: 2
- Value (Hex): 00000000

Bar4 Enabled

Bar5 Enabled

OK | Cancel

Generate PCIe Core

- Set Performance Level to High.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

Show disabled ports

s_axis_tx
icap
startup
pipe_clock
drp
pcie2_cfg_err
pcie2_cfg_interrupt
pcie2_cfg_control
pcie_cfg_mgmt
pcie_qpll_drp
transceiver_debug
pcie2_pl
pcie_ext_ch_gt
pcie2_ext_pipe_ep
pcie2_ext_pipe_ep_legacy
sys_clk
sys_rst_n
startup_eos_in
pcie_drp_clk

m_axis_rx
pcie_7x_mgt
pcie2_cfg_status
pcie_cfg_fc
pcie2_cfg_msg_rcvd
pcie_sharedlogic_int_clk
pcie2_ext_pipe_rp
user_clk_out
user_reset_out
user_lnk_up
user_app_rdy
ext_ch_gt_drpclk

Component Name: PCIeGen2x8If128

Basic | IDs | BARs | Core Capabilities | Link Registers | Interrupts | Power Management | Ext Capabilities | Ext Capabilities-2 | TL Setting

Capabilities Register

Capability Version (Hex): 2
Device Port / Type: PCI_Express_Endpoint_device
Slot Implemented
Capabilities Register (Hex): 0002

Device Capabilities Register

Max Payload Size: 256 bytes
Extended Tag Field
Extended Tag Default
Phantom Functions: No function number bits used
Acceptable L0s Latency: Maximum of 64 ns
Acceptable L1 Latency: No limit
Device Capabilities Register (Hex): 00000E01

BRAM Configuration Options

Buffering Optimized for Bus Mastering Applications
Finite Completions

Perf Level: High

Performance Level	Transmit TLPs Buffered	Receiver Buffer Size (bytes)	Posted Header/Data Credits	Non-posted Header/Data Credits	C
Good	14	4096	4/32	4/8	72
High	29	8192	4/32	4/8	72

Disable Completion Timeout

OK | Cancel

Generate PCIe Core

- On this page, use the default settings.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

Show disabled ports

s_axis_tx
icap
startup
pipe_clock
drp
pcie2_cfg_err
pcie2_cfg_interrupt
pcie2_cfg_control
pcie_cfg_mgmt
pcie_qpll_drp
transceiver_debug
pcie2_pl
pcie_ext_ch_gt
pcie2_ext_pipe_ep
pcie2_ext_pipe_ep_legacy
sys_clk
sys_rst_n
startup_eos_in
pcie_drp_clk

m_axis_rx
pcie_7x_mgt
pcie2_cfg_status
pcie_cfg_fc
pcie2_cfg_msg_rcvd
pcie_sharedlogic_int_clk
pcie2_ext_pipe_rp
user_clk_out
user_reset_out
user_lnk_up
user_app_rdy
ext_ch_gt_drpclk

Component Name: PCIeGen2x8If128

Basic | IDs | BARs | Core Capabilities | Link Registers | Interrupts | Power Management | Ext Capabilities | Ext Capabilities-2 | TL Setting

Link Capabilities Register

Supported Link Speeds (Hex): 2
Maximum Link Width (Hex): 8
ASPM Optionality
DLL Link Active Reporting Capability
Link Capabilities Register (Hex): 0003F482

Link Control Registers

Hardware Autonomous Speed Disable

Read Completion Boundary: 64 byte / 128 byte
Target Link Speed: 2.5 GT/s / 5.0 GT/s
Compliance De-emphasis: -6 / 0 / dB:

Link Control-1 Register (Hex): 0
Link Control-2 Register (Hex): -

Link Status Register

Enable Slot Clock Configuration

OK | Cancel

Generate PCIe Core

- Disable the interrupt.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

Show disabled ports

s_axis_tx
icap
startup
pipe_clock
drp
pcie2_cfg_err
pcie2_cfg_interrupt
pcie2_cfg_control
pcie_cfg_mgmt
pcie_qpll_drp
transceiver_debug
pcie2_pl
pcie_ext_ch_gt
pcie2_ext_pipe_ep
pcie2_ext_pipe_ep_legacy
sys_clk
sys_rst_n
startup_eos_in
pcie_drp_clk

m_axis_rx
pcie_7x_mgt
pcie2_cfg_status
pcie_cfg_fc
pcie2_cfg_msg_rcvd
pcie_sharedlogic_int_clk
pcie2_ext_pipe_rp
user_clk_out
user_reset_out
user_lnk_up
user_app_rdy
ext_ch_gt_drpclk

Component Name: PCIeGen2x8If128

Basic | IDs | BARs | Core Capabilities | Link Registers | Interrupts | Power Management | Ext Capabilities | Ext Capabilities-2 | TL Setting

Legacy Interrupt Settings

- [ ] Enable IntX

Interrupt PIN: NONE

MSI Capabilities

- [x] Enable MSI Capability Structure
- [x] 64 bit Address Capable

Multiple Message Capable: 1 vector

- [ ] Per Vector Masking Capable

MSIx Capabilities

- [ ] Enable MSIx Capability Structure

MSIx Table Settings

Table Size (Hex): 1 — Range: 1..800

Table Offset: 0 — Range: 0..1FFFFFFF

BAR Indicator: BAR 0

MSIx Pending Bit Array (PBA) Settings

PBA Offset: 0 — Range: 0..1FFFFFFF

PBA BAR Indicator: BAR 0

OK | Cancel

Generate PCIe Core

- On this page, use the default settings.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

☑ Show disabled ports

Component Name: PCIeGen2x8If128

Basic | IDs | BARs | Core Capabilities | Link Registers | Interrupts | **Power Management** | Ext Capabilities | Ext Capabilities-2 | TL Setting

Power Management Registers

☐ Device Specific Initialization

☐ D1 Support ☐ D2 Support

PME Support

☑ D0 ☑ D1 ☑ D2 ☑ D3hot ☐ D3cold

☑ No Soft Reset

Power Consumption

	Power Consumed		Scale Factor		Total Power
D0	0	x	0	=	0.0(Watts)
D1	0	x	0	=	0.0(Watts)
D2	0	x	0	=	0.0(Watts)
D3	0	x	0	=	0.0(Watts)
	Range: 0..255		Range: 0..3		

Power Dissipation

	Power Dissipated		Scale Factor		Tota
D0	0	x	0	=	0.0(
D1	0	x	0	=	0.0(
D2	0	x	0	=	0.0(
D3	0	x	0	=	0.0(
	Range: 0..255		Range: 0..3		

OK | Cancel

Generate PCIe Core

- On this page, use the default settings.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

Show disabled ports

s_axis_tx
ioap
startup
pipe_clock
drp
pcie2_cfg_err
pcie2_cfg_interrupt
pcie2_cfg_control
pcie_cfg_mgmt
pcie_qpll_drp
transceiver_debug
pcie2_pl
pcie_ext_ch_gt
pcie2_ext_pipe_ep
pcie2_ext_pipe_ep_legacy
sys_clk
sys_rst_n
startup_eos_in
pcie_drp_clk

m_axis_rx
pcie_7x_mgt
pcie2_cfg_status
pcie_cfg_fc
pcie2_cfg_msg_rcvd
pcie_sharedlogic_int_clk
pcie2_ext_pipe_rp
user_clk_out
user_reset_out
user_lnk_up
user_app_rdy
ext_ch_gt_drpclk

Component Name: PCIeGen2x8If128

Basic | IDs | BARs | Core Capabilities | Link Registers | Interrupts | Power Management | **Ext Capabilities** | Ext Capabilities-2 | TL Setting

Device Serial Number Capability

The Device Serial Number (DSN) Capability is an optional PCIe Extended Capability, that contains a unique Device Serial Number. This identifier must be presented on the Device Serial Number Input pin of the port.

☑ Enable DSN Capability

Virtual Channel Capability

The Virtual Channel (VC) Capability is an optional PCIe Extended Capability, which when enabled, allows the port to support functionality beyond the default Traffic Class (TC0) over the default Virtual Channel (VC0). Checking this allows Traffic Class (TC) filtering to be supported.

☐ Enable VC Capability

☐ Reject Snoop Transactions

Vendor Specific Capability

The Vendor Specific (VSec) Capability is an optional PCIe Extended Capability, which enables Xilinx specific Loopback Control.

☐ Enable VSEC Capability

User Defined Configuration Capabilities

☐ PCI Configuration Space Enable

PCI Configuration Space Pointer: 3F — Range: 2A..3F

☐ PCI Express Extended Configuration Space Enable

PCI Express Extended Configuration Space Pointer: 3FF — Range : 043..3FF

OK | Cancel

Generate PCIe Core

- On this page, use the default settings.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

Show disabled ports

s_axis_tx
icap
startup
pipe_clock
drp
pcie2_cfg_err
pcie2_cfg_interrupt
pcie2_cfg_control
pcie_cfg_mgmt
pcie_qpll_drp
transceiver_debug
pcie2_pl
pcie_ext_ch_gt
pcie2_ext_pipe_ep
pcie2_ext_pipe_ep_legacy
sys_clk
sys_rst_n
startup_eos_in
pcie_drp_clk

m_axis_rx
pcie_7x_mgt
pcie2_cfg_status
pcie_cfg_fc
pcie2_cfg_msg_rcvd
pcie_sharedlogic_int_clk
pcie2_ext_pipe_rp
user_clk_out
user_reset_out
user_lnk_up
user_app_rdy
ext_ch_gt_drpclk

Component Name: PCIeGen2x8If128

Basic | IDs | BARs | Core Capabilities | Link Registers | Interrupts | Power Management | Ext Capabilities | Ext Capabilities-2 | TL Setting

Enable AER Capability

The Advanced Error Reporting(AER) Capability is an optional PCIe Extended Capability, which when enabled, allows advanced error control and reporting

Multiheader | Permit Root Error Update
ECRC Check Capable | ECRC Generate Capable

Optional Error Support: 000000

Correctable Internal Error | Completion Timeout | Uncorrectable Internal Error
Header Log Overflow | Completer Abort | MC Blocked TLP
Receiver Error | Receiver Overflow | AtomicOp Egress Blocked
Surprise Down | ECRC Error | TLP Prefix Blocked
Flow Control Protocol Error | ACS Violation

RBAR Capabilities

The Resizable BAR(RBAR) Capabilities is an optional PCIe Extended Capability, which when enabled, adds a capability for Functions with BARS to report various options for sizes of their memory mapped resources.

Enable RBAR Capability

OK | Cancel

Generate PCIe Core

- On this page, use the default settings.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

Show disabled ports

s_axis_tx
icap
startup
pipe_clock
drp
pcie2_cfg_err
pcie2_cfg_interrupt
pcie2_cfg_control
pcie_cfg_mgmt
pcie_qpll_drp
transceiver_debug
pcie2_pl
pcie_ext_ch_gt
pcie2_ext_pipe_ep
pcie2_ext_pipe_ep_legacy
sys_clk
sys_rst_n
startup_eos_in
pcie_drp_clk

m_axis_rx
pcie_7x_mgt
pcie2_cfg_status
pcie_cfg_fc
pcie2_cfg_msg_rcvd
pcie_sharedlogic_int_clk
pcie2_ext_pipe_rp
user_clk_out
user_reset_out
user_lnk_up
user_app_rdy
ext_ch_gt_drpclk

Component Name: PCIeGen2x8If128

sic | IDs | BARs | Core Capabilities | Link Registers | Interrupts | Power Management | Ext Capabilities | Ext Capabilities-2 | TL Settings

Transaction Layer Module Advanced Settings

Routing Received Messages to Transaction Interface

Controls if Message TLPs are also received on the Transaction Interface

Endpoint

☐ Unlock ☐ PME_Turn_OFF

Root Port

☐ Error Correctable ☐ Error Non-Fatal ☐ Error Fatal

☐ INTA ☐ INTB ☐ INTC ☐ INTD

☐ PM_PME ☐ PME_TO_ACK

☑ Receive Non-Posted Request

Pipeline Registers for Transaction Block RAM Buffers: None

ATS

☑ UR INV REQ ☑ UR PRS RESPONSE

OK | Cancel

Generate PCIe Core

- On this page, use the default settings.

Re-customize IP

7 Series Integrated Block for PCI Express (3.3)

Documentation | IP Location | Switch to Defaults

Show disabled ports

s_axis_tx
icap
startup
pipe_clock
drp
pcie2_cfg_err
pcie2_cfg_interrupt
pcie2_cfg_control
pcie_cfg_mgmt
pcie_qpll_drp
transceiver_debug
pcie2_pl
pcie_ext_ch_gt
pcie2_ext_pipe_ep
pcie2_ext_pipe_ep_legacy
sys_clk
sys_rst_n
startup_eos_in
pcie_drp_clk

m_axis_rx
pcie_7x_mgt
pcie2_cfg_status
pcie_cfg_fc
pcie2_cfg_msg_rcvd
pcie_sharedlogic_int_clk
pcie2_ext_pipe_rp
user_clk_out
user_reset_out
user_lnk_up
user_app_rdy
ext_ch_gt_drpclk

Component Name: PCIeGen2x8If128

Power Management | Ext Capabilities | Ext Capabilities-2 | TL Settings | DL & PL Settings | Shared Logic | Core Interface Parameters | Add.

Link Layer Module Advanced Settings

Override ACK/NAK Latency Timer

Override Function: Absolute

Override Value: 0000 Range: 0000..7FFF

Override Replay Timer

Override Function: Add

Override Value: 0000 Range: 0000..7FFF

Advanced Physical Layer Settings

Enable Lane Reversal | Force No Scrambling

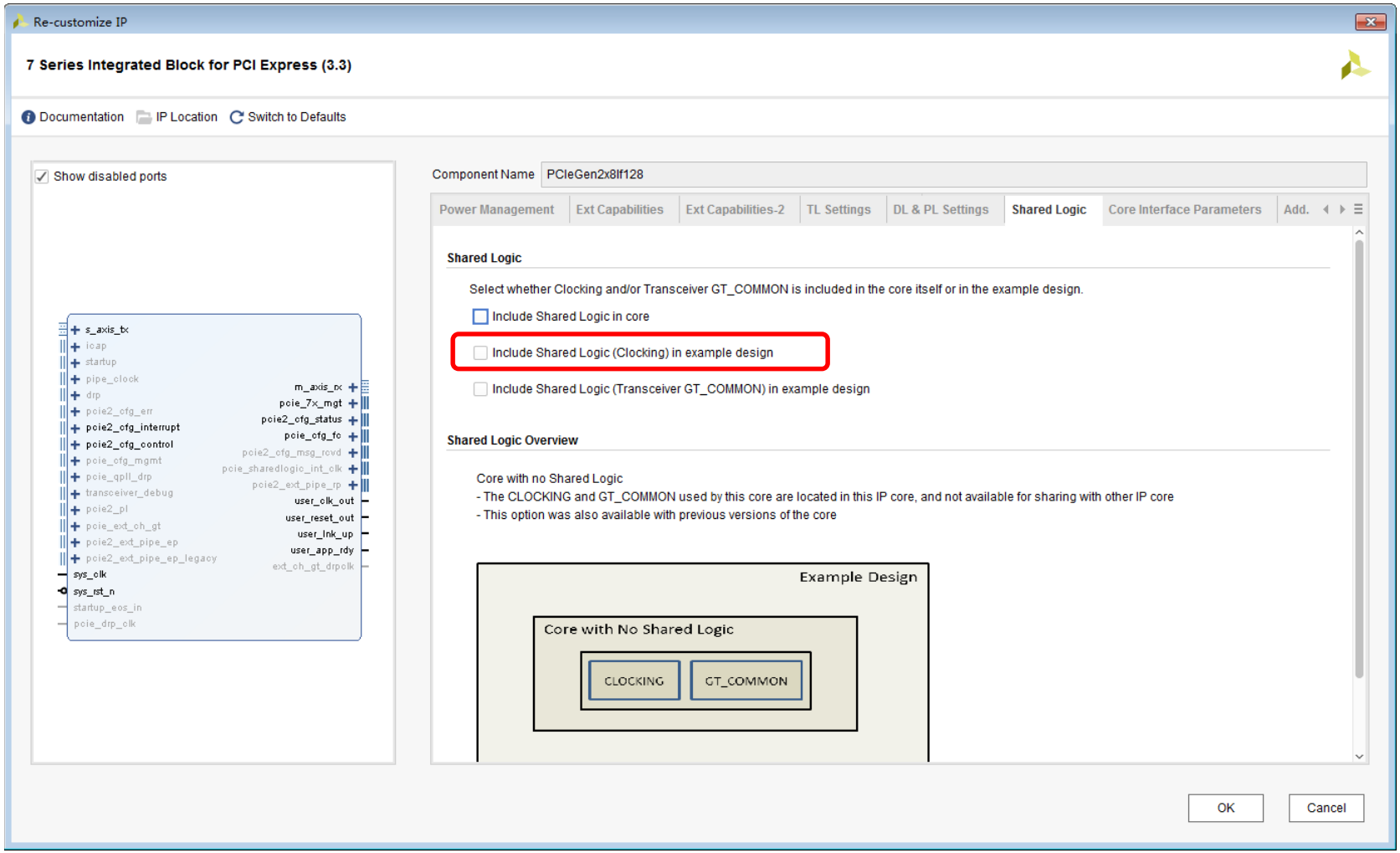
Upconfigure Capable | Disable Tx ASPM L0s

Link Number: 00 Range: 00..FF

OK | Cancel

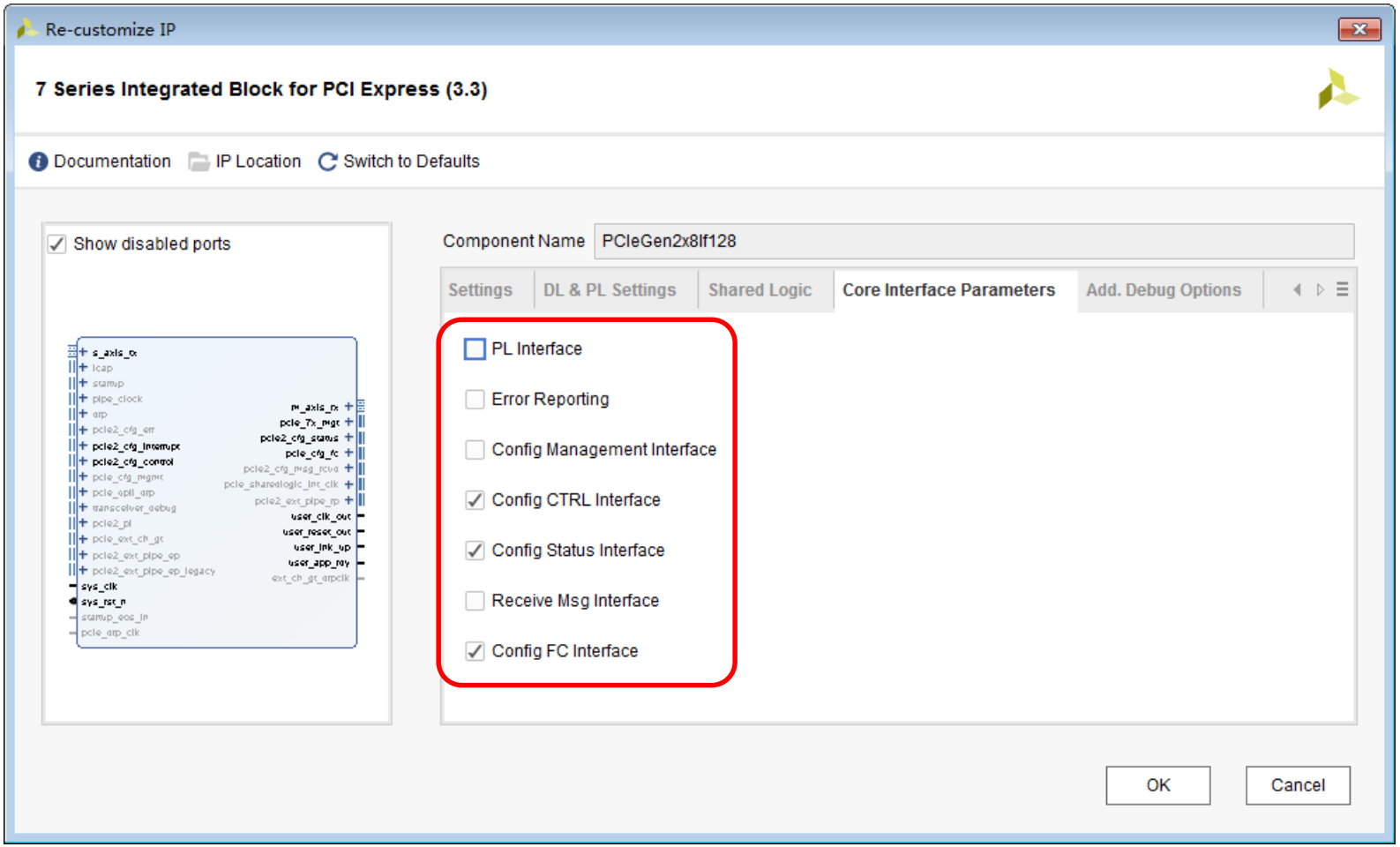
Generate PCIe Core

- Uncheck: Include Shared Logic in example design.



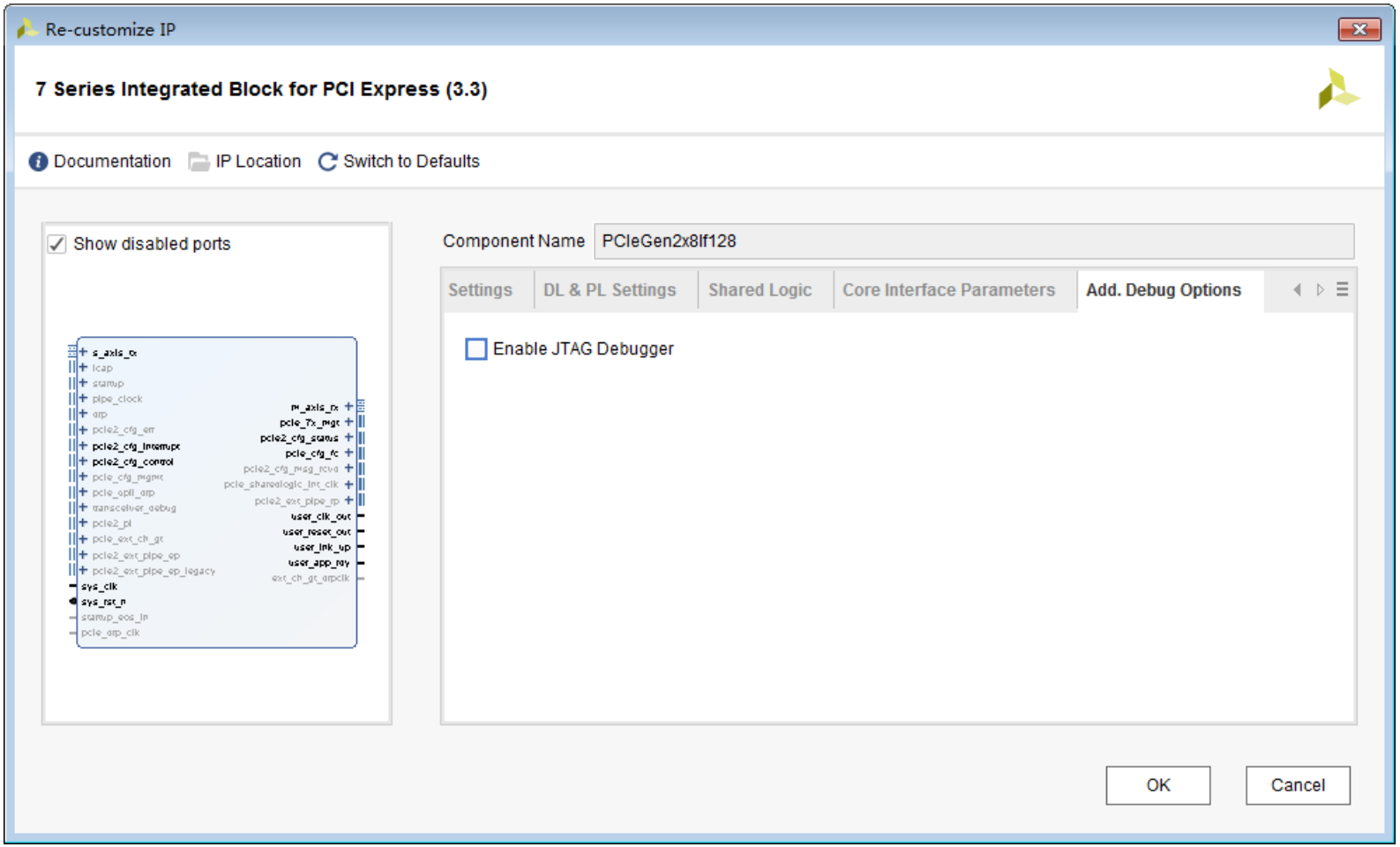
Generate PCIe Core

- Configure the tab as shown below



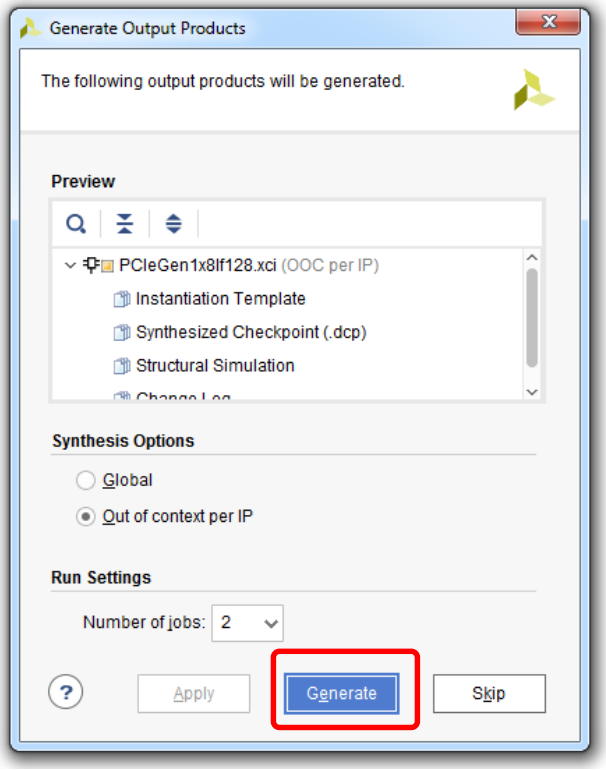
Generate PCIe Core

- Use the default settings, and then click OK.



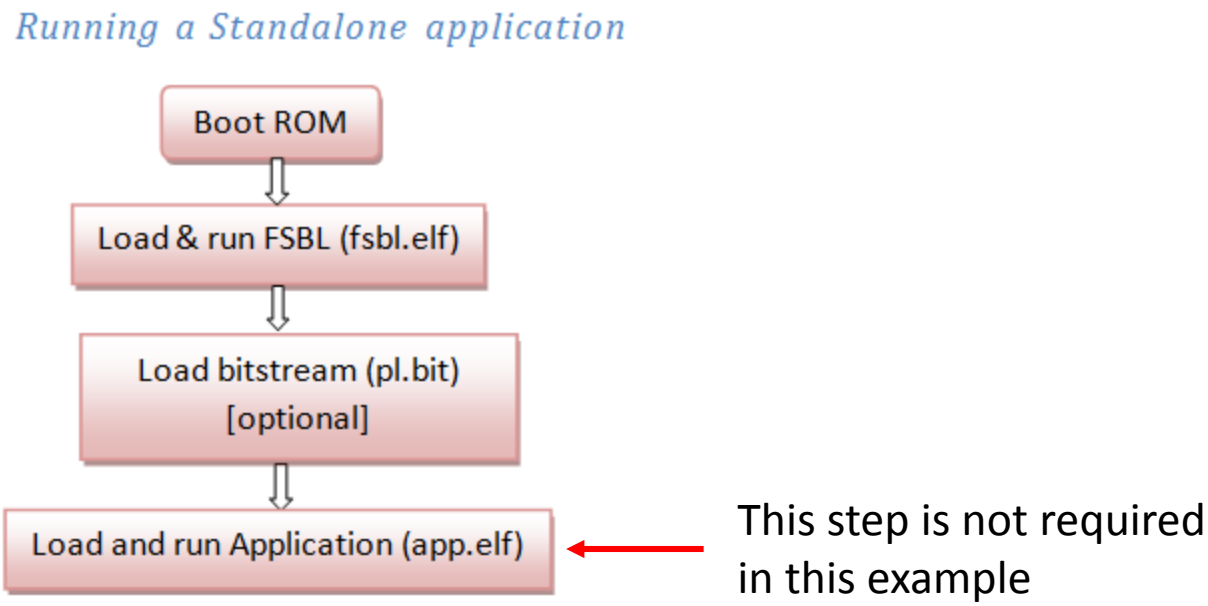
Generate PCIe Core

- Click Generate.



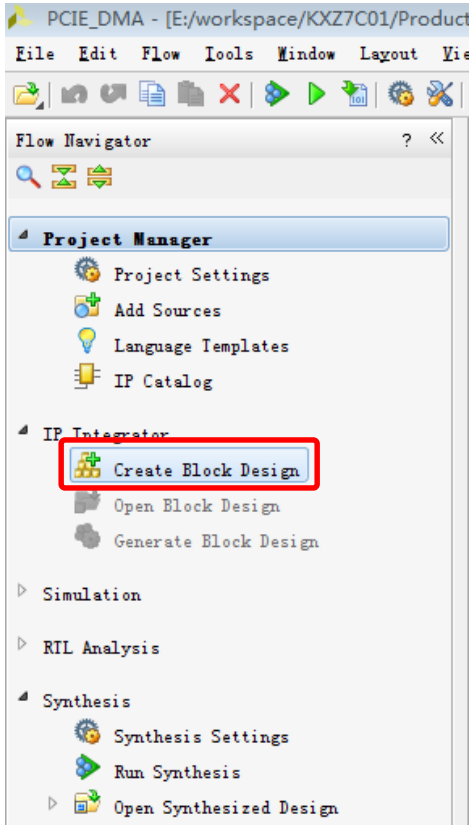
First Stage Boot Loader

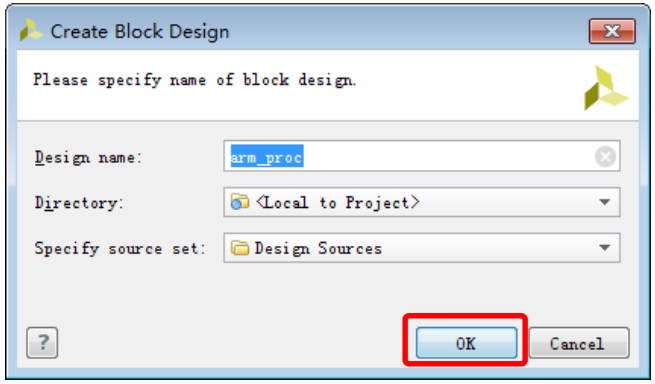
- With Zynq FPGA, in order to load the bitfile from Flash, a First Stage Boot Loader (FSBL) is required. Thus we need to configure the ARM processor although it is not used in this PCIe DMA example.



Configure ARM Processor

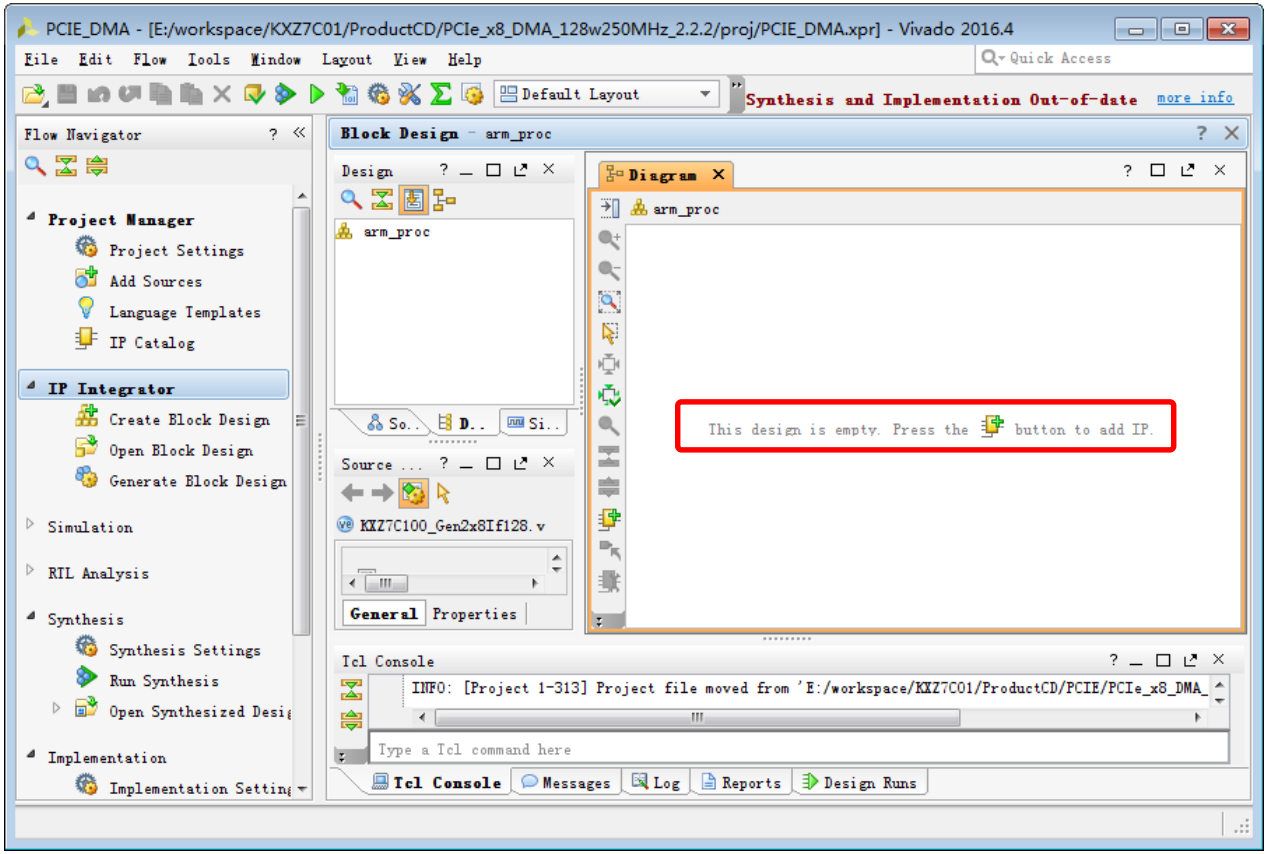
- Create a dummy Block Design.

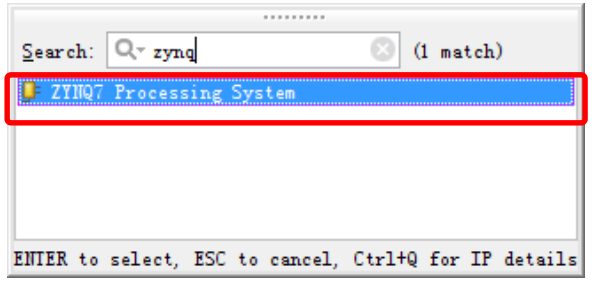




Configure ARM Processor

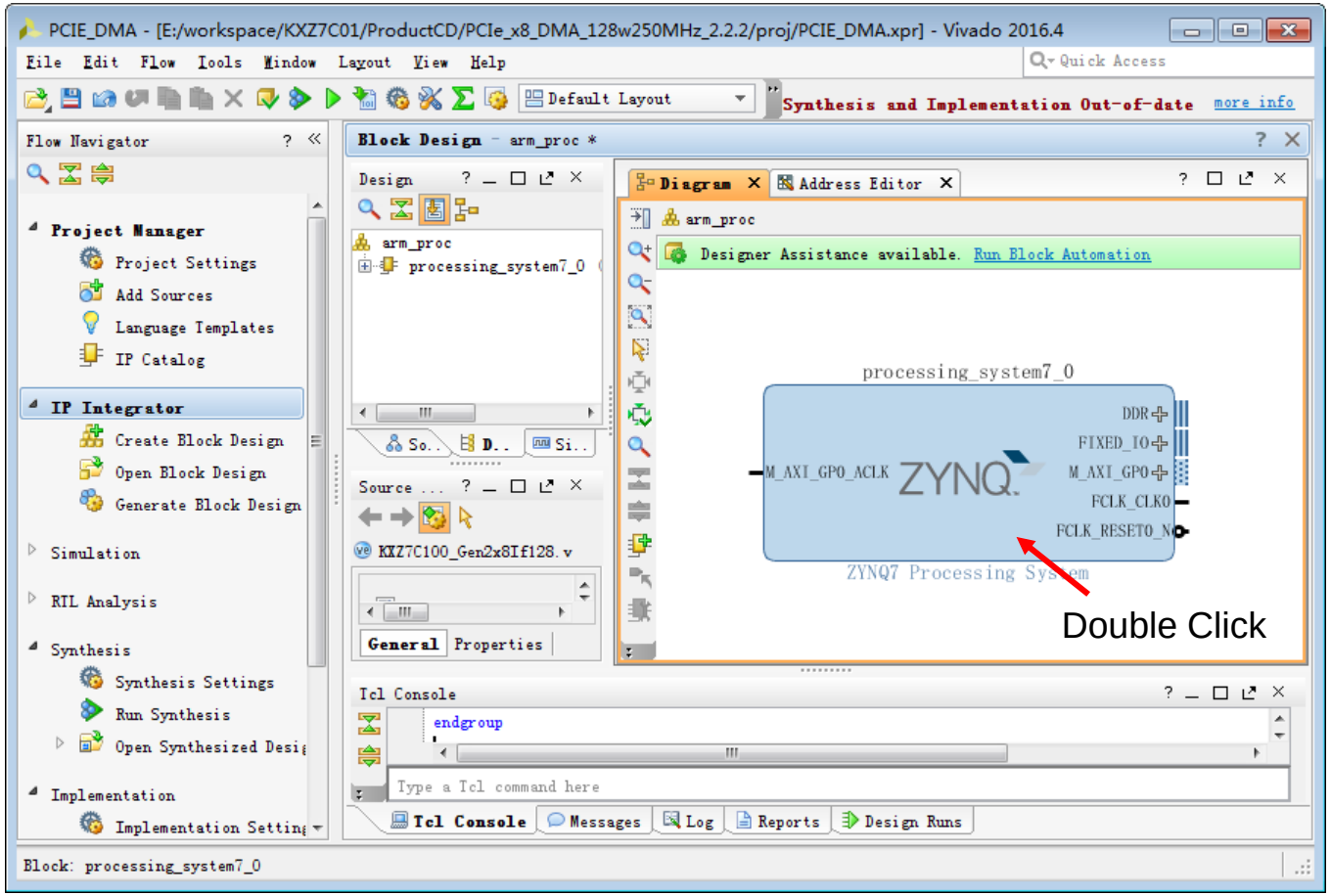
- Click the button to add IP, and choose “ZYNQ7 Processing System”.





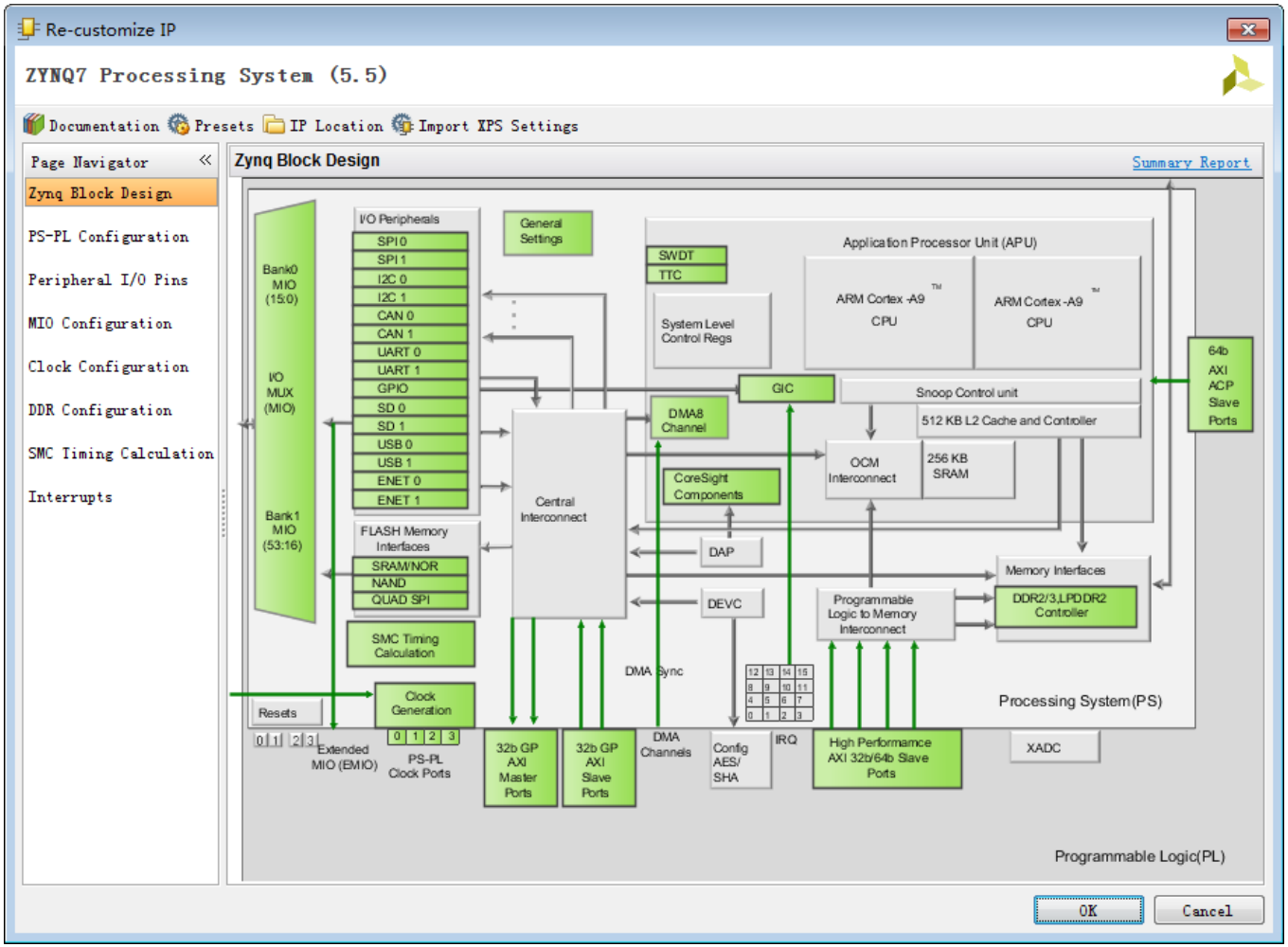
Configure ARM Processor

- Double click on ZYNQ7 Processing System to open its configuration.



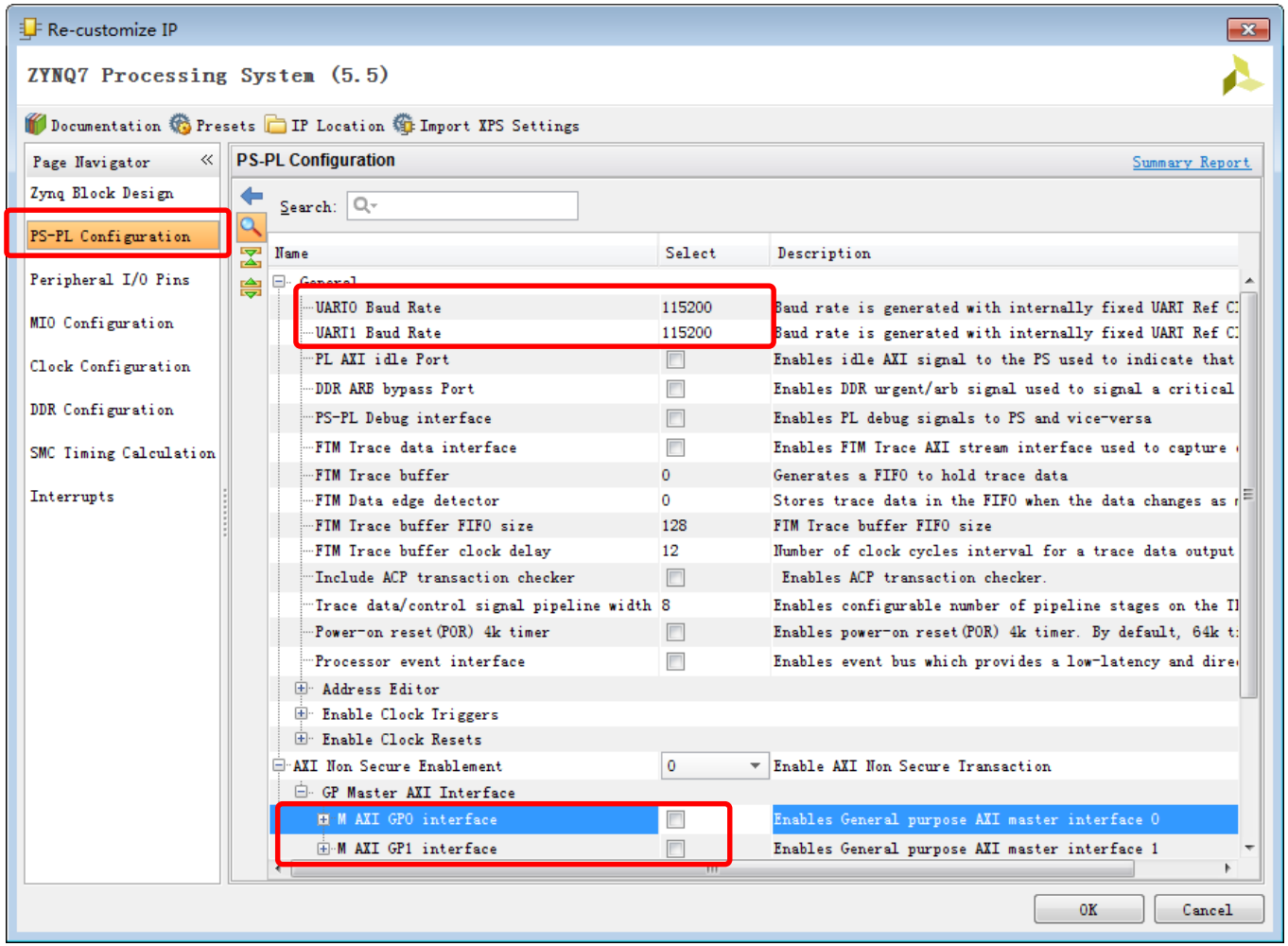
Configure ARM Processor

- On this page, use the default settings.



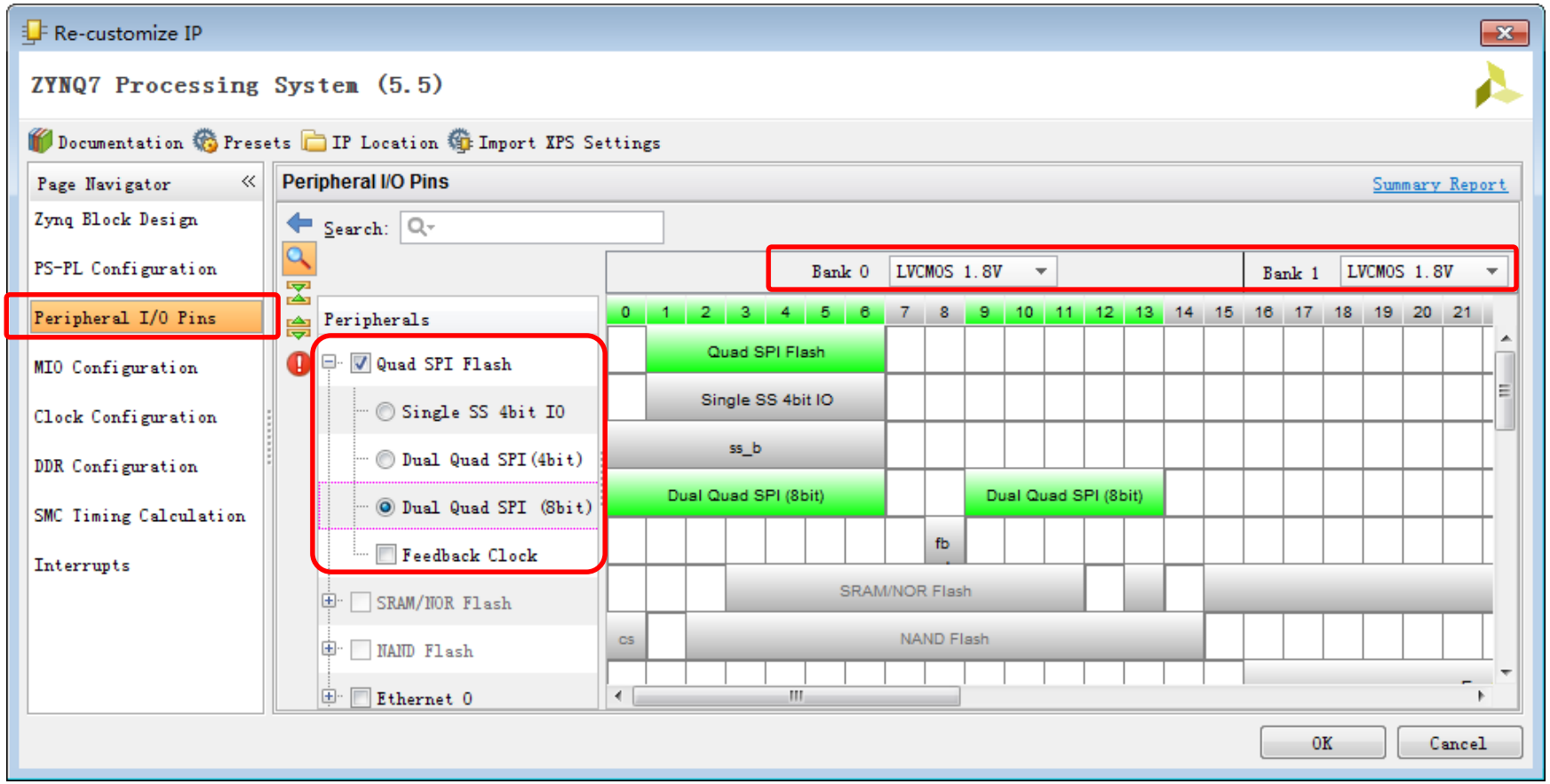
Configure ARM Processor

- “PS-PL Configuration”: set UART Baud rate to 115200.



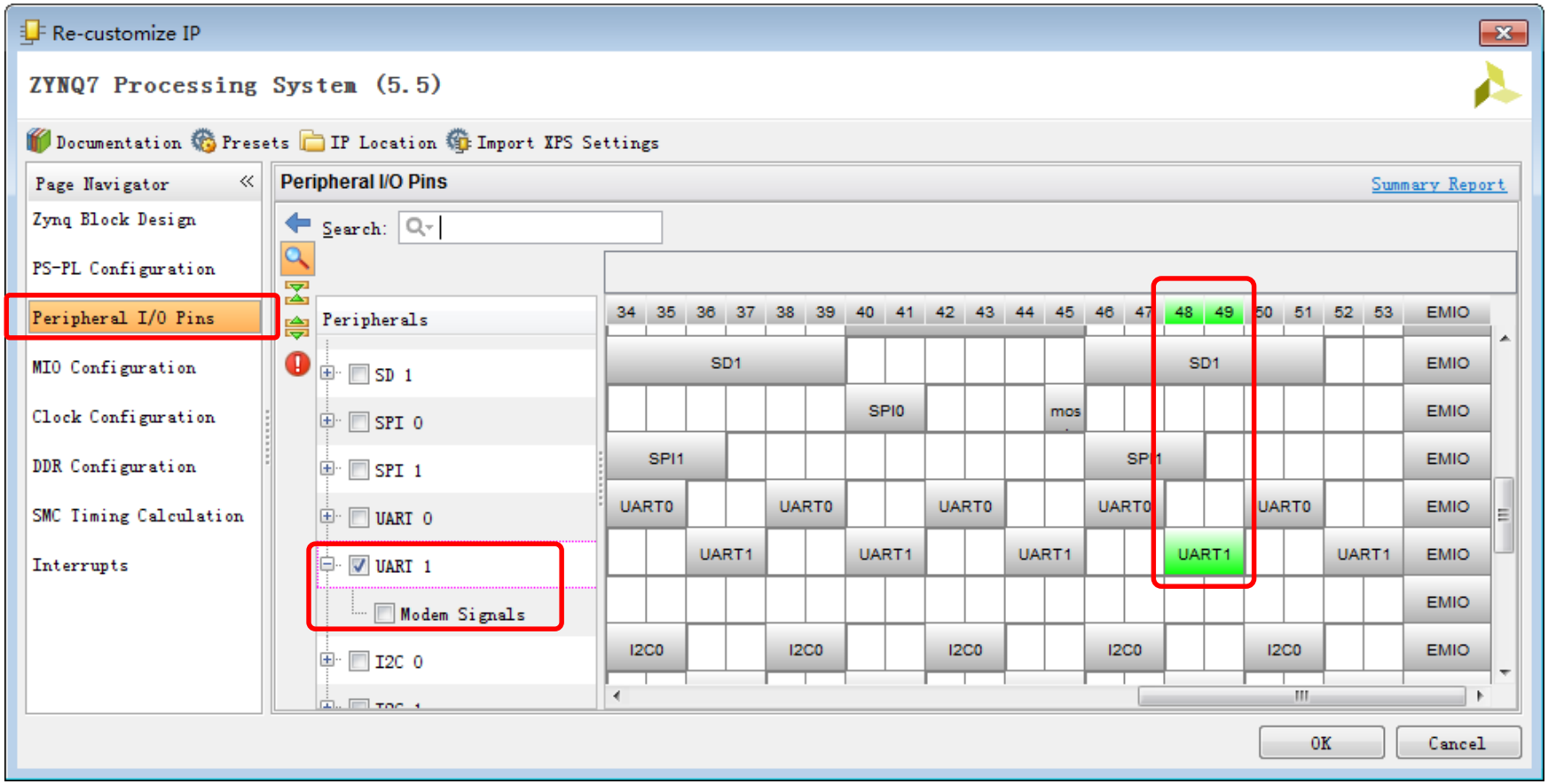
Configure ARM Processor

- "Peripheral I/O Pins":
  - Change voltage level for both Banks to "LVCMOS 1.8V";
  - Enable the "Quad SPI Flash" in "Dual Quad SPI (8bit) mode".



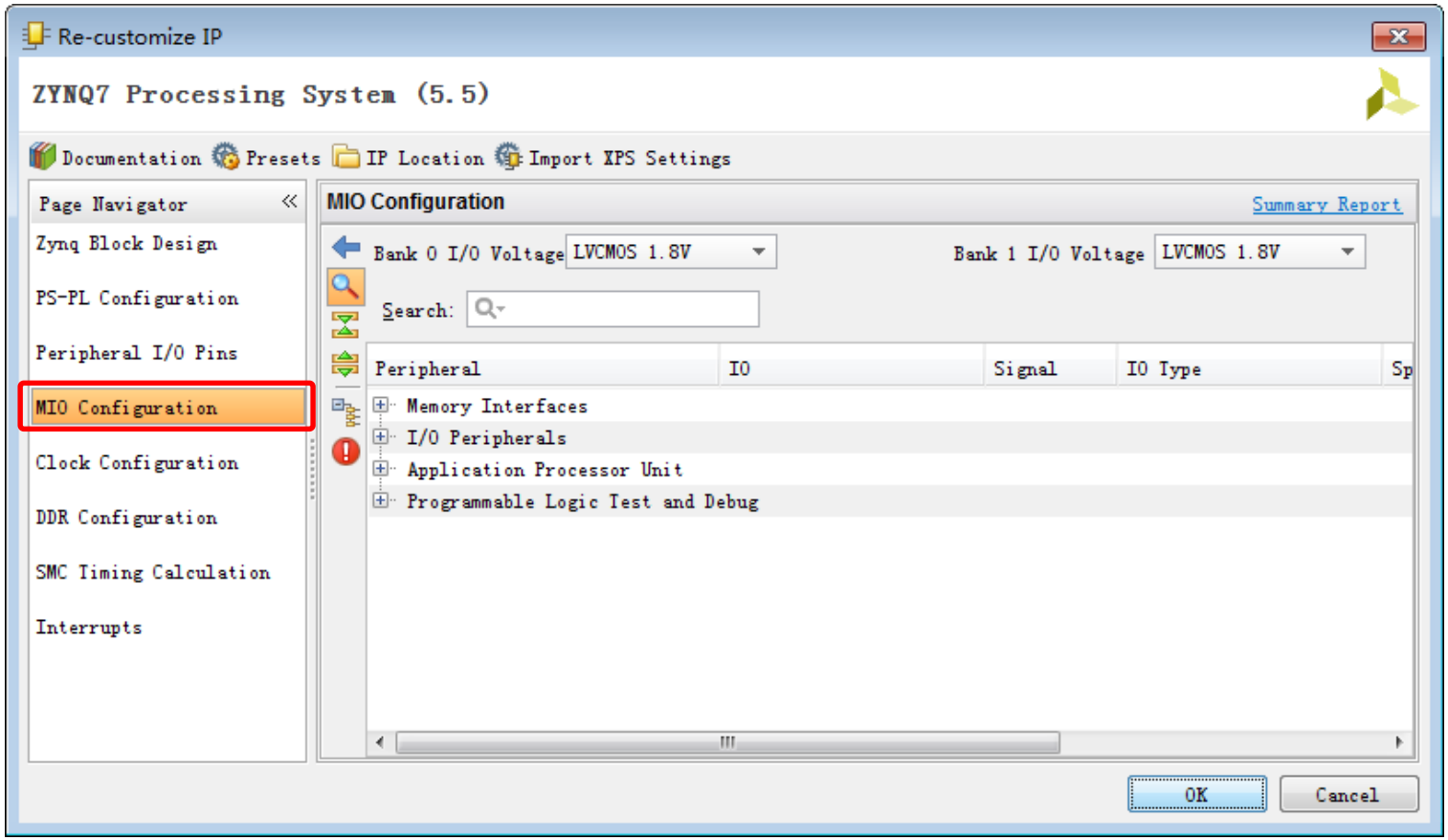
Configure ARM Processor

- "Peripheral I/O Pins":
  - Assign the UART1 to pin 48 and 49 as shown below.



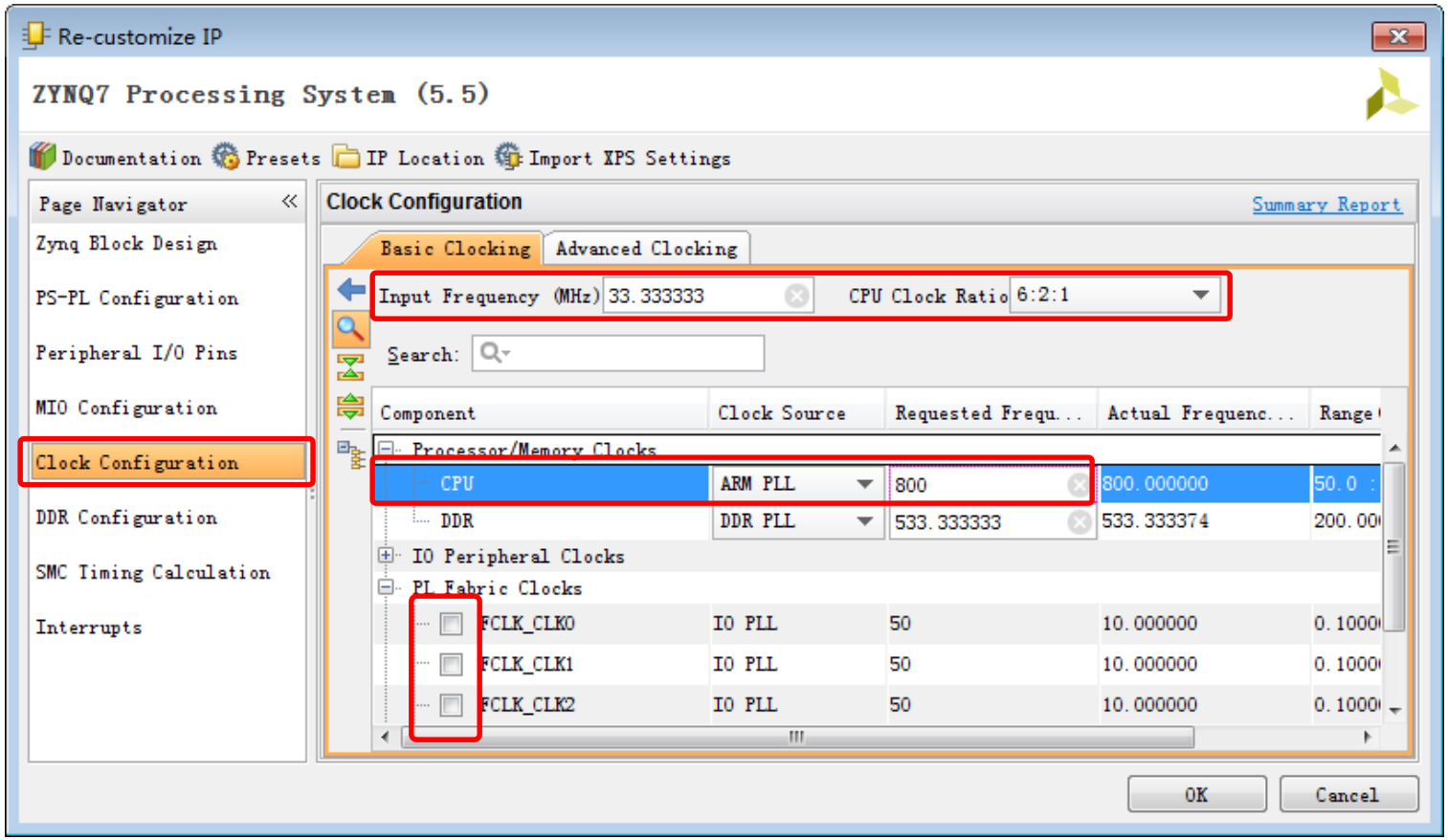
Configure ARM Processor

- “MIO Configuration”: keep the default settings.



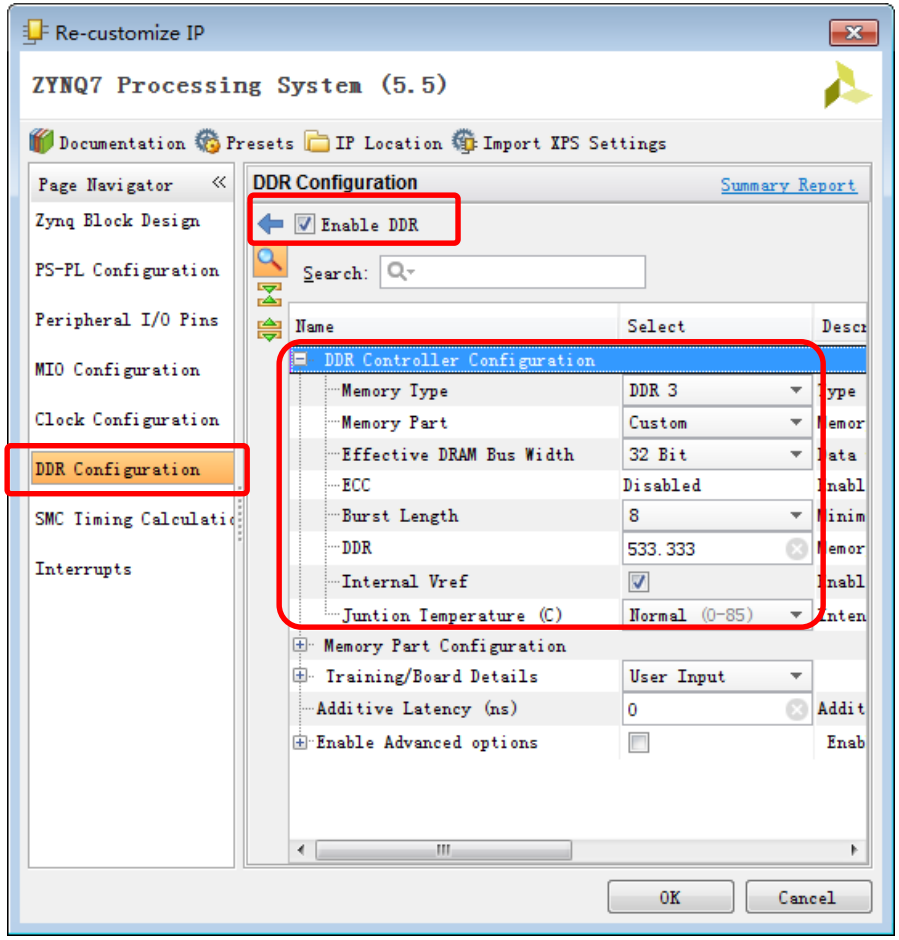
Configure ARM Processor

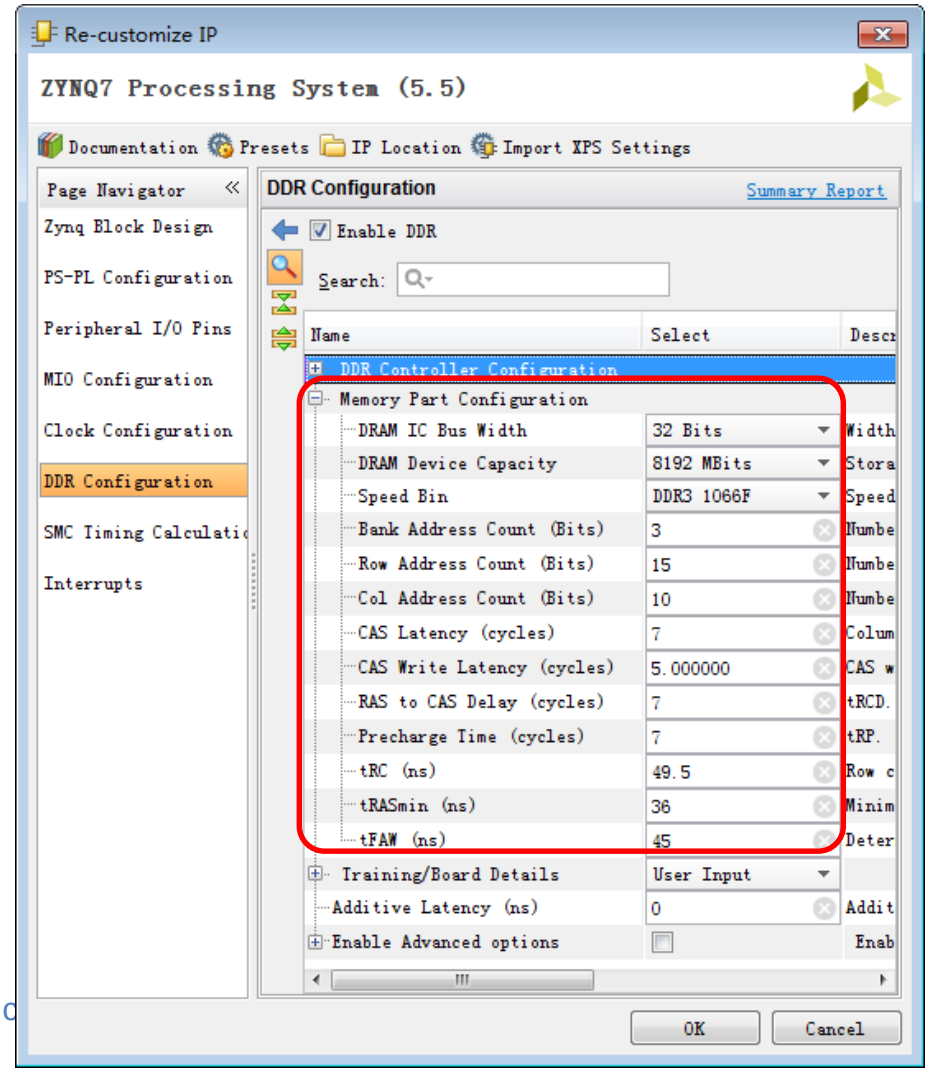
- “Clock Configuration”: use the configuration as follows



Configure ARM Processor

- DDR should be configured as follows:





Configure ARM Processor

- DDR should be configured as follows:

Re-customize IP

ZYNQ7 Processing System (5.5)

Documentation | Presets | IP Location | Import XPS Settings

Page Navigator

- Zynq Block Design
- PS-PL Configuration
- Peripheral I/O Pins
- MIO Configuration
- Clock Configuration
- DDR Configuration
- SMC Timing Calculation
- Interrupts

DDR Configuration — Summary Report

Enable DDR

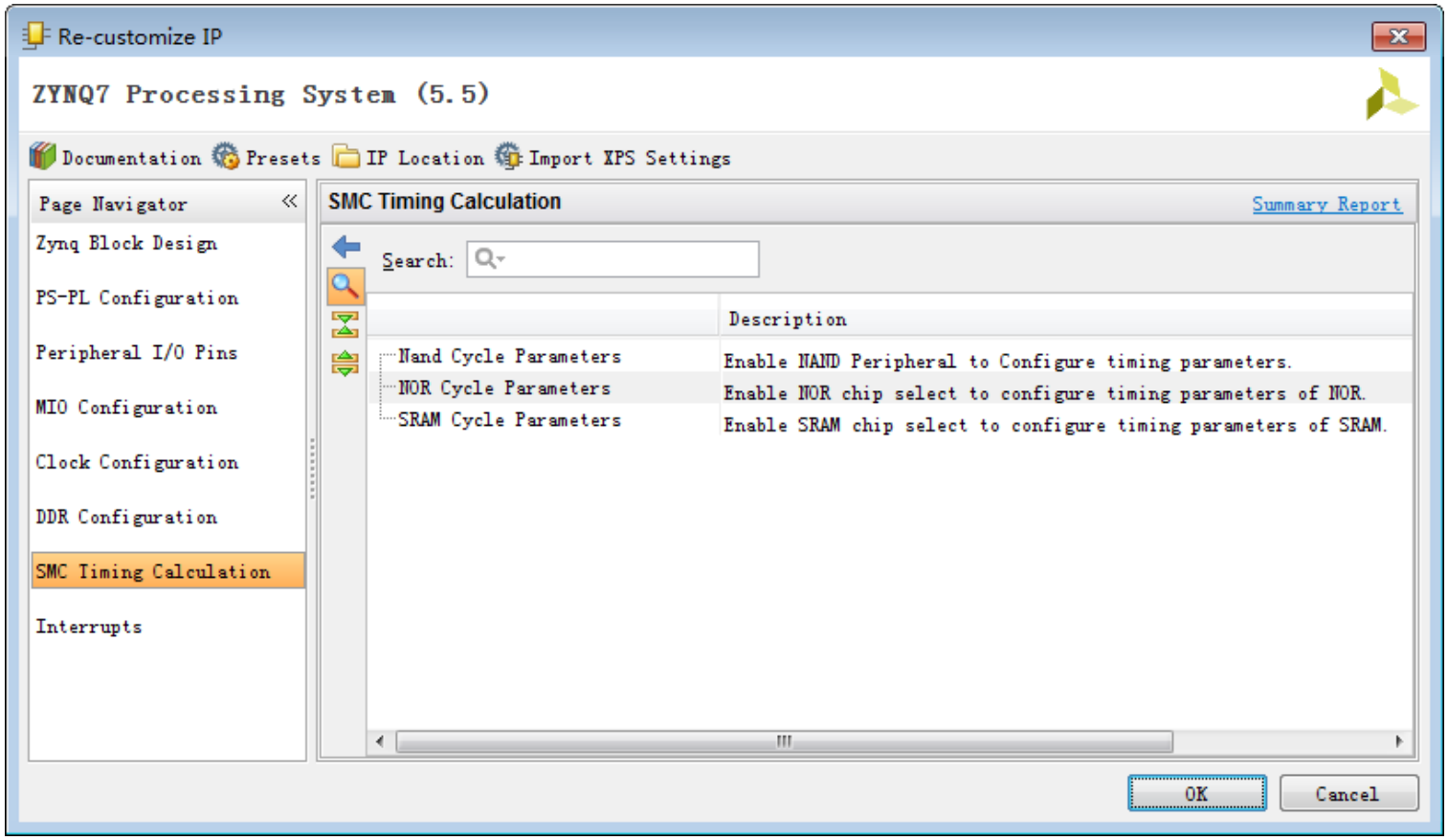
Search:

Name	Select	Description
DDR Controller Configuration		
Memory Part Configuration		
Training/Board Details	User Input	
DRAM Training		
Write leveling	☐	Enables Write Leveling calibration, which adjusts write DQS rel...
Read gate	☐	Enables Read Gate calibration, which adjusts valid RD DQS window.
Read data eye	☐	Enables Read Data Eye calibration, which adjusts the read DQS t...
DQS to Clock Delay (ns)		
DQS0	0.077	DQS to Clock delay [0] (ns). The DQS path delay subtracted from...
DQS1	0.074	DQS to Clock delay [1] (ns). The DQS path delay subtracted from...
DQS2	-0.058	DQS to Clock delay [2] (ns). The DQS path delay subtracted from...
DQS3	-0.054	DQS to Clock delay [3] (ns). The DQS path delay subtracted from...
Board Delay (ns)		
DQ[7:0]	0.482	Board delay [0] (ns). The midrange of data (DDR_DQ, DDR_DM) tra...
DQ[15:8]	0.484	Board delay [1] (ns). The midrange of data (DDR_DQ, DDR_DM) tra...
DQ[23:16]	0.418	Board delay [2] (ns). The midrange of data (DDR_DQ, DDR_DM) tra...
DQ[31:24]	0.416	Board delay [3] (ns). The midrange of data (DDR_DQ, DDR_DM) tra...
Additive Latency (ns)	0	Additive Latency (ns). Increases the efficiency of the command ...
Enable Advanced options	☐	Enable Advanced DDR QoS settings

OK | Cancel

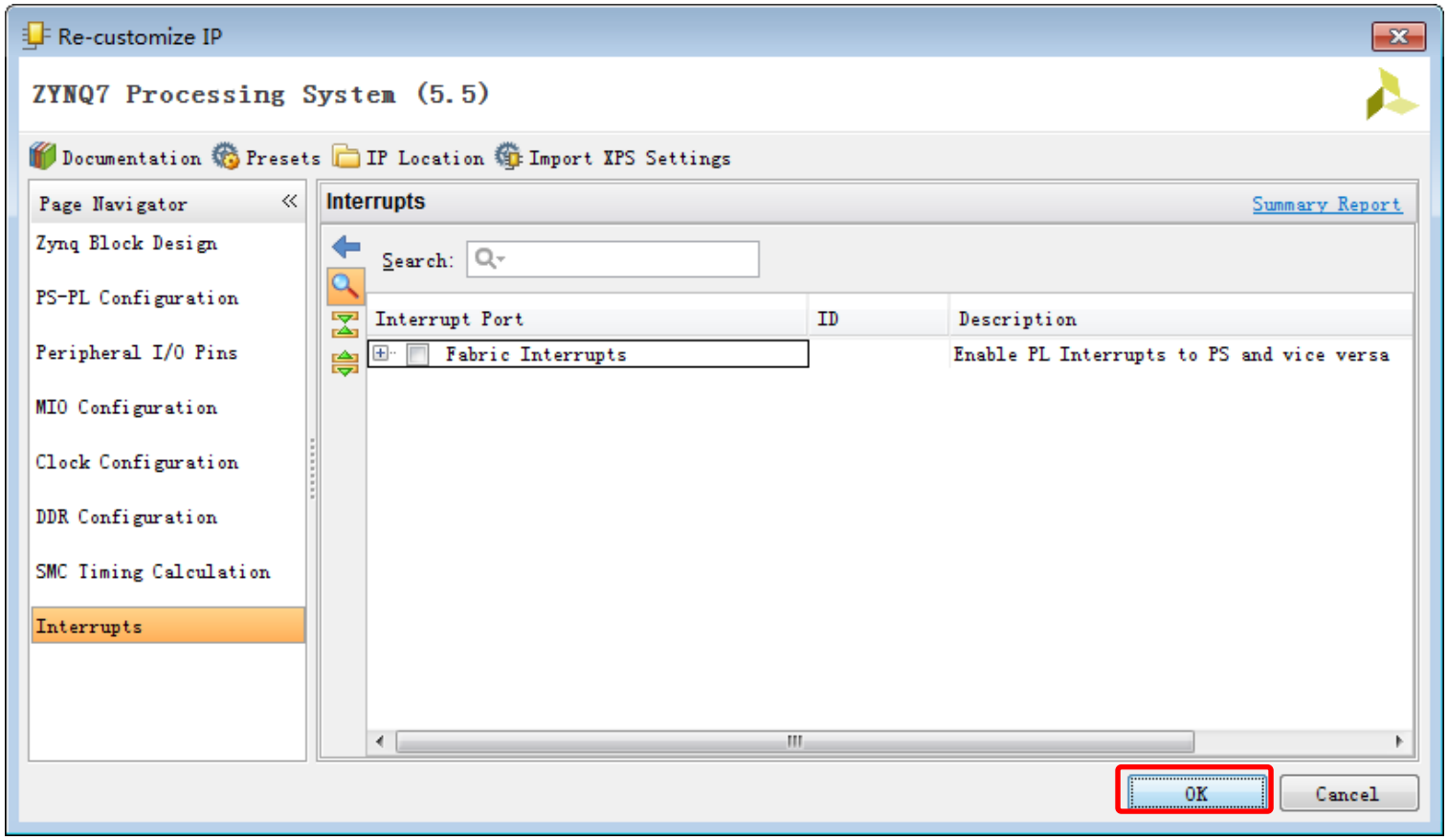
Configure ARM Processor

- “SMC Timing Calculation”: Keep the default settings



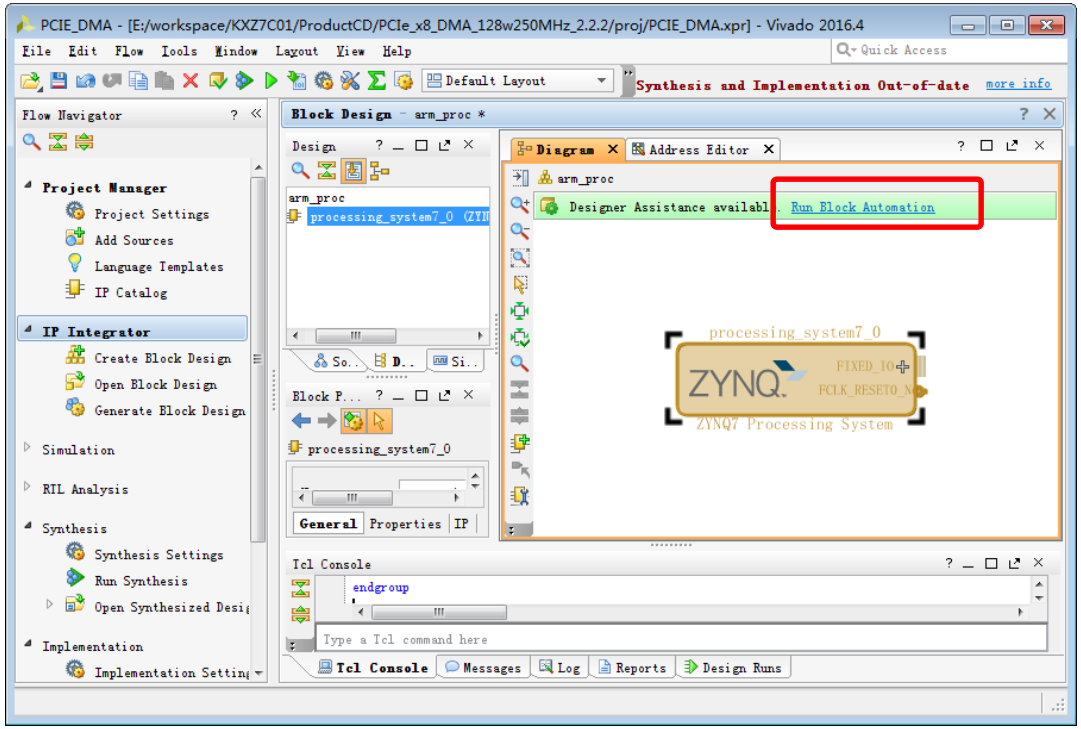
Configure ARM Processor

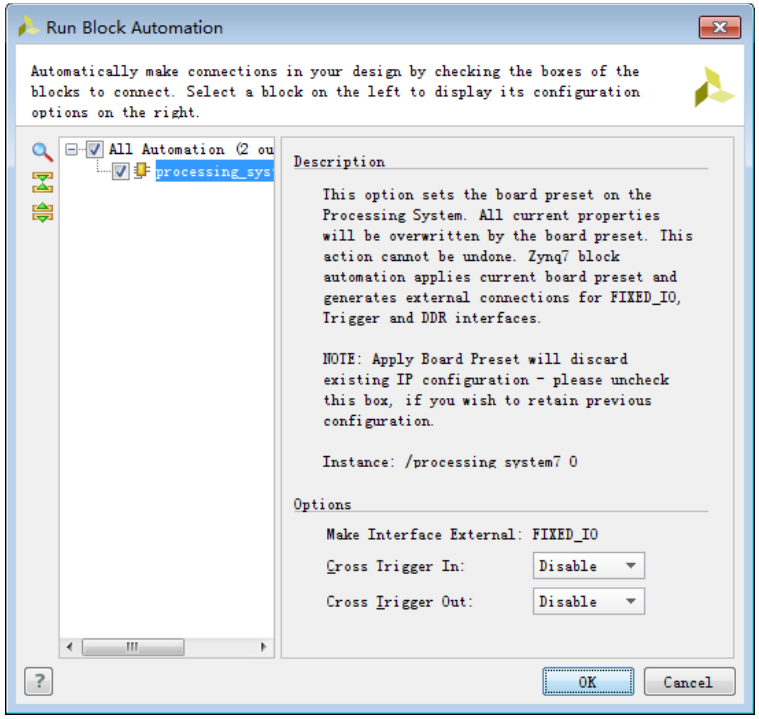
- “Interrupts”: Not used in this example.
- Click “OK” to save the configuration.



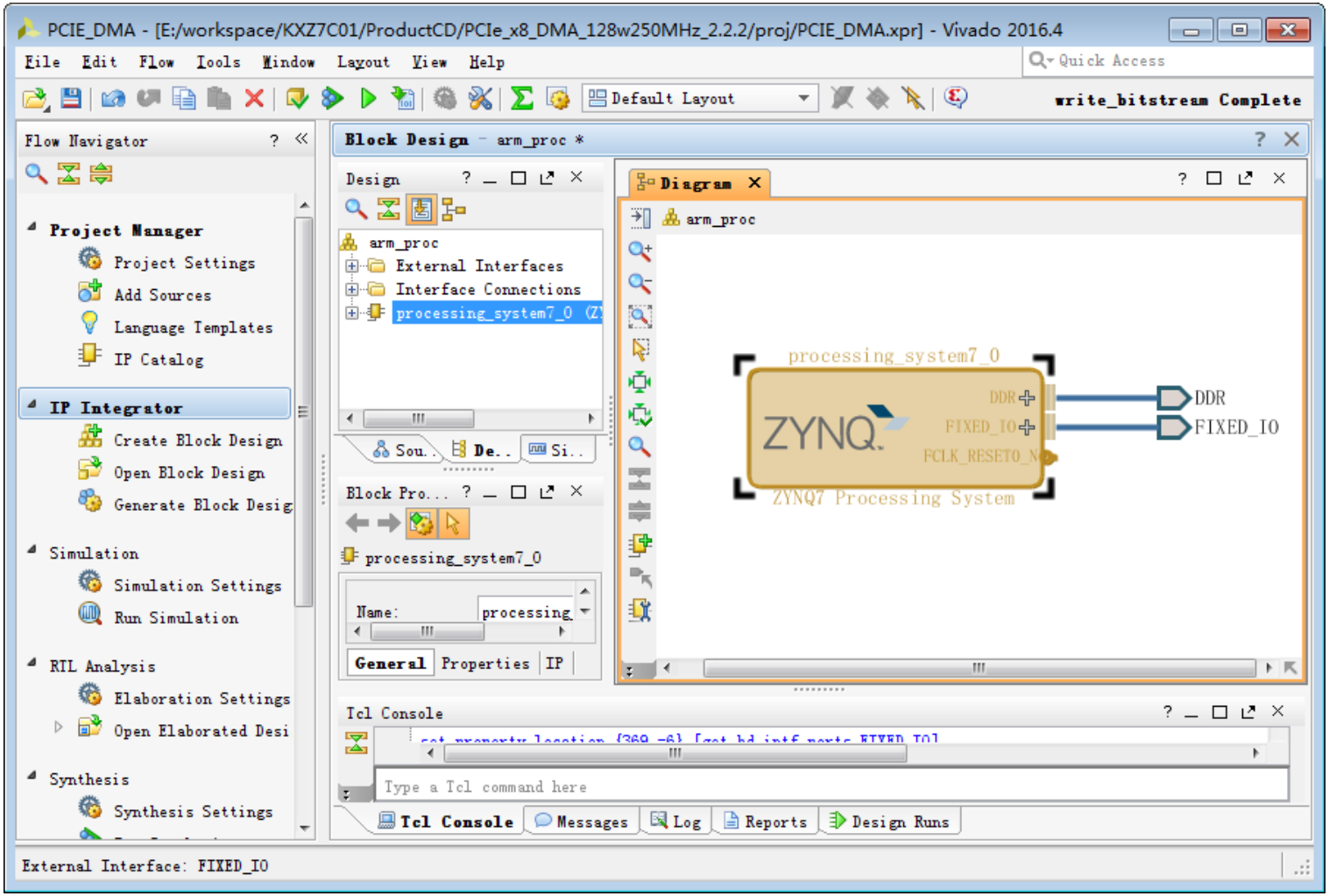
Configure ARM Processor

- Click the “Run Block Automation” to finalize the block design.



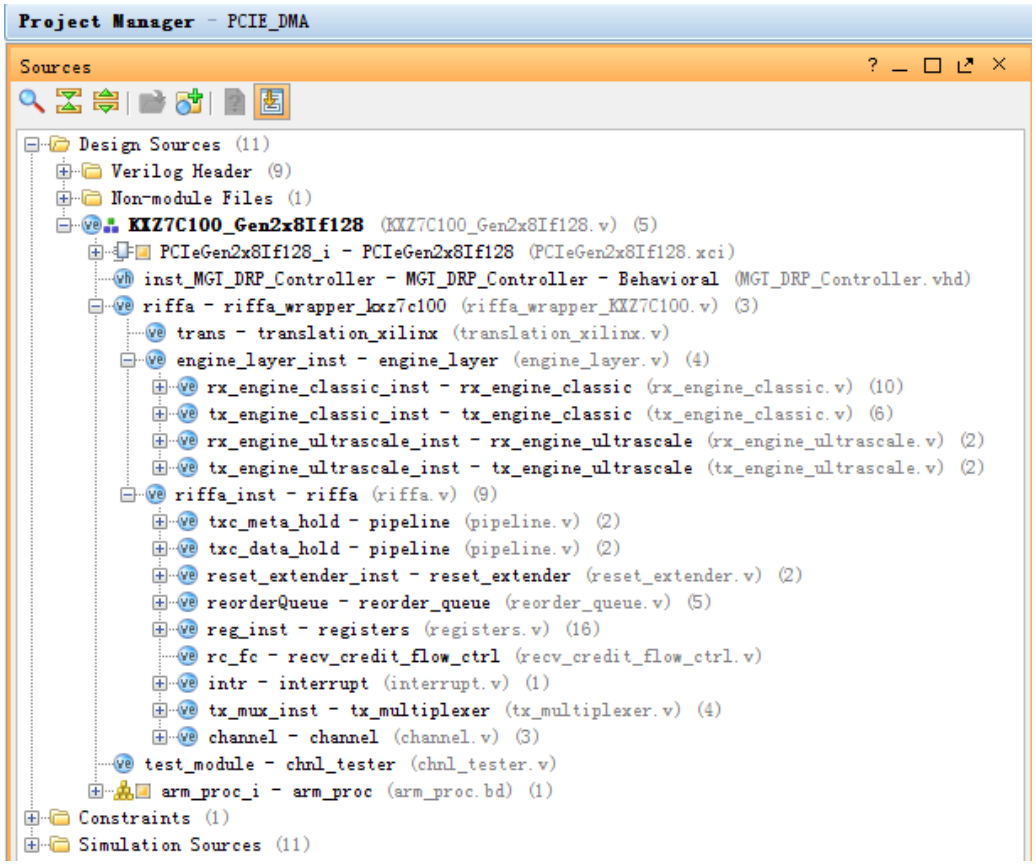


Configure ARM Processor



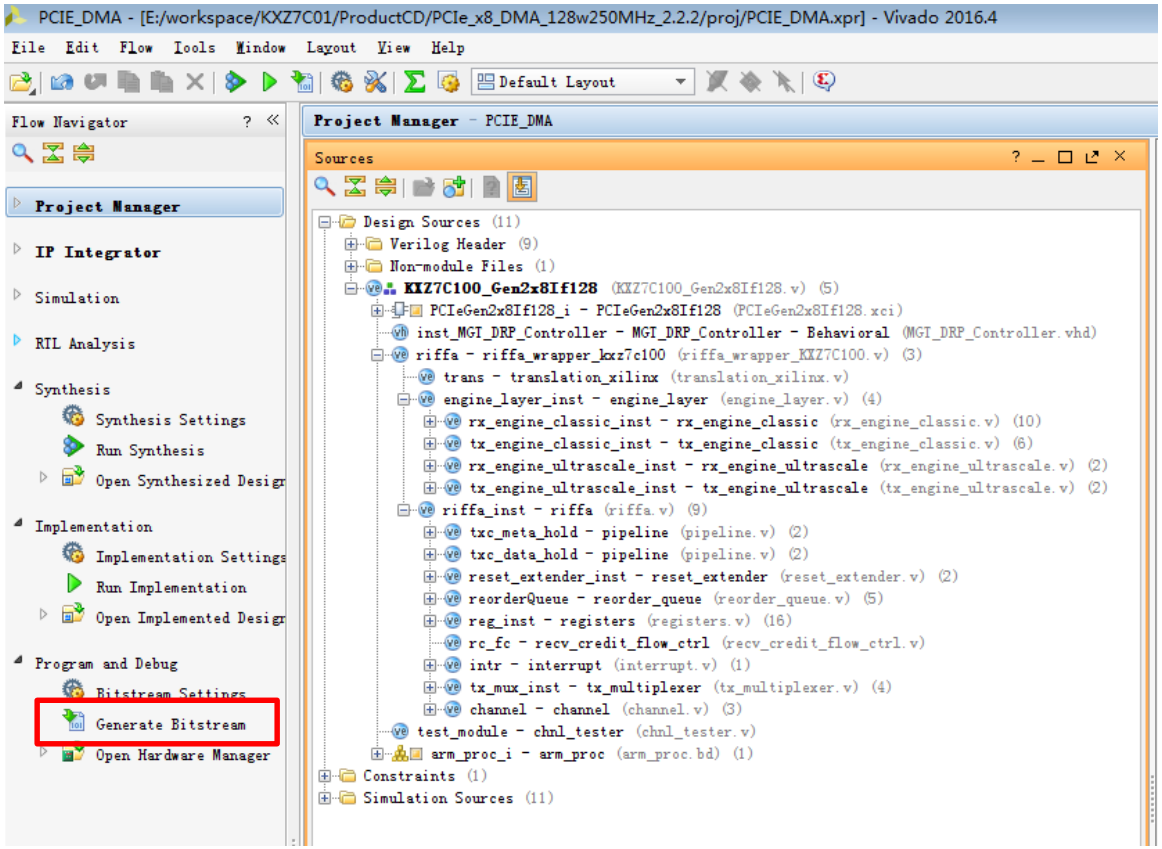
Add Files to Vivado Project

- Add source files and constrain file into the Vivado project (the source files are available in GVI product CD)



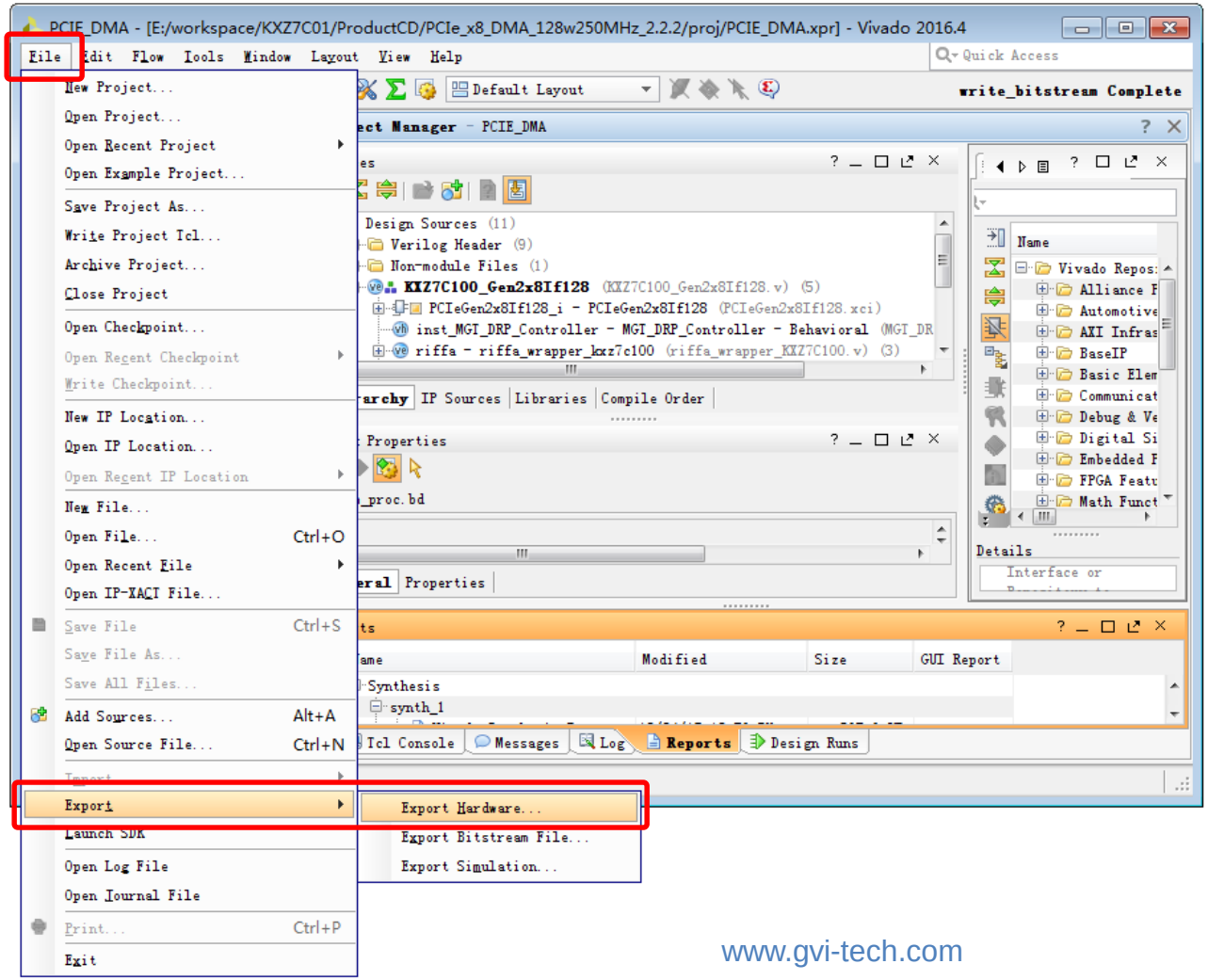
Compile the Demo Project

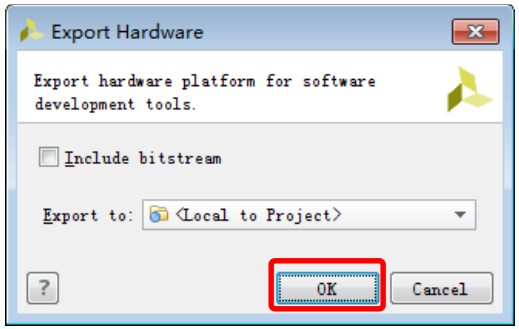
- Set “KXZ7C100_Gen2x8If128.v” to be the top-level design.
- Click “Generate Bitstream” to generate the FPGA configuration file.



First Stage Boot Loader

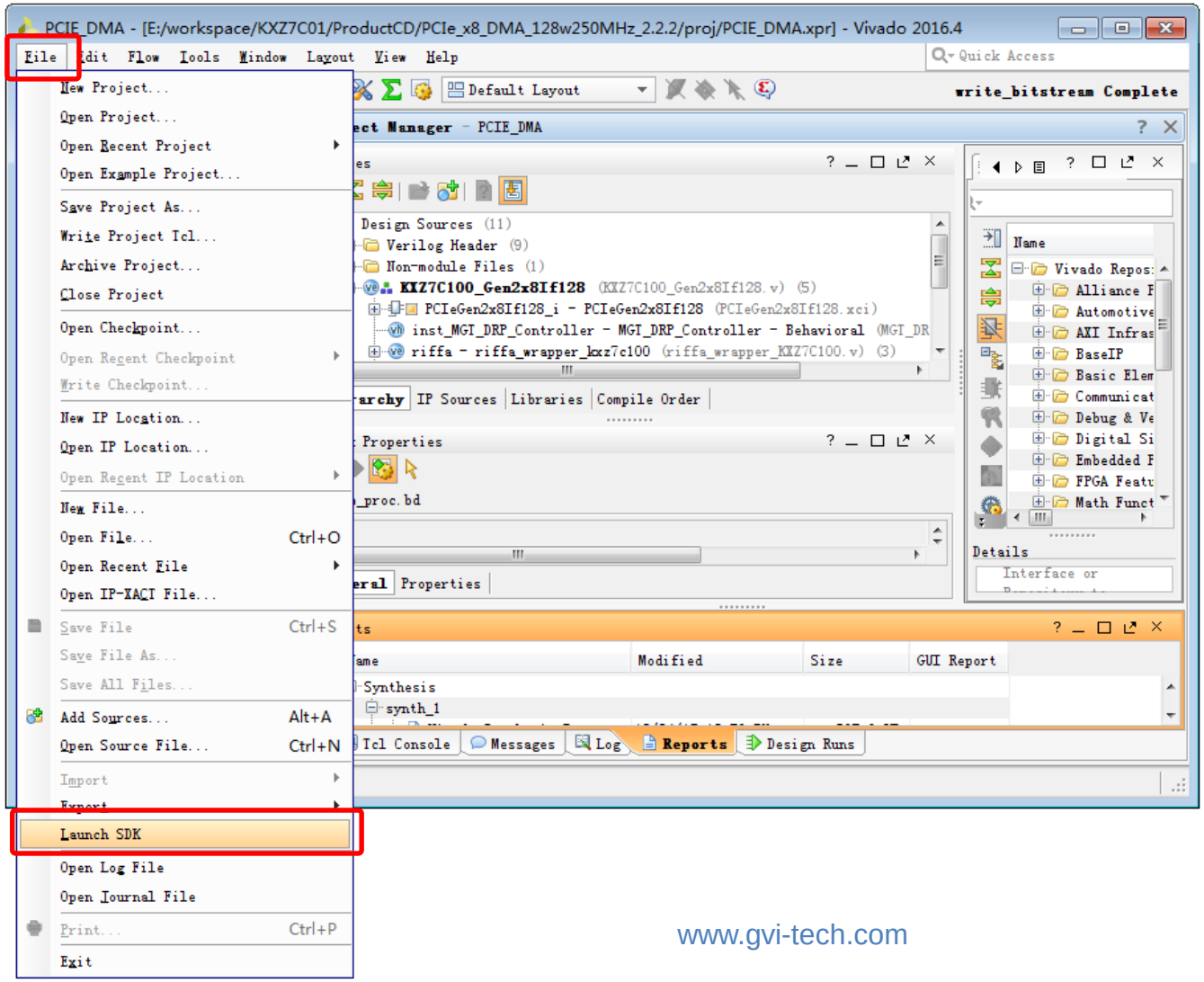
- Export Hardware to SDK.

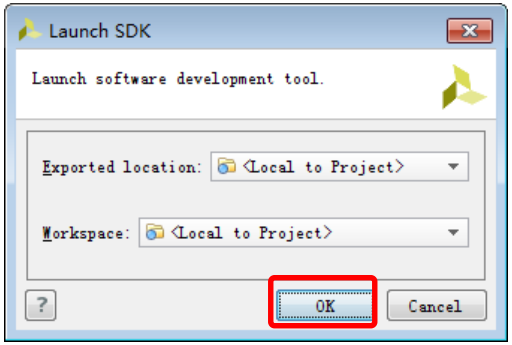




First Stage Boot Loader

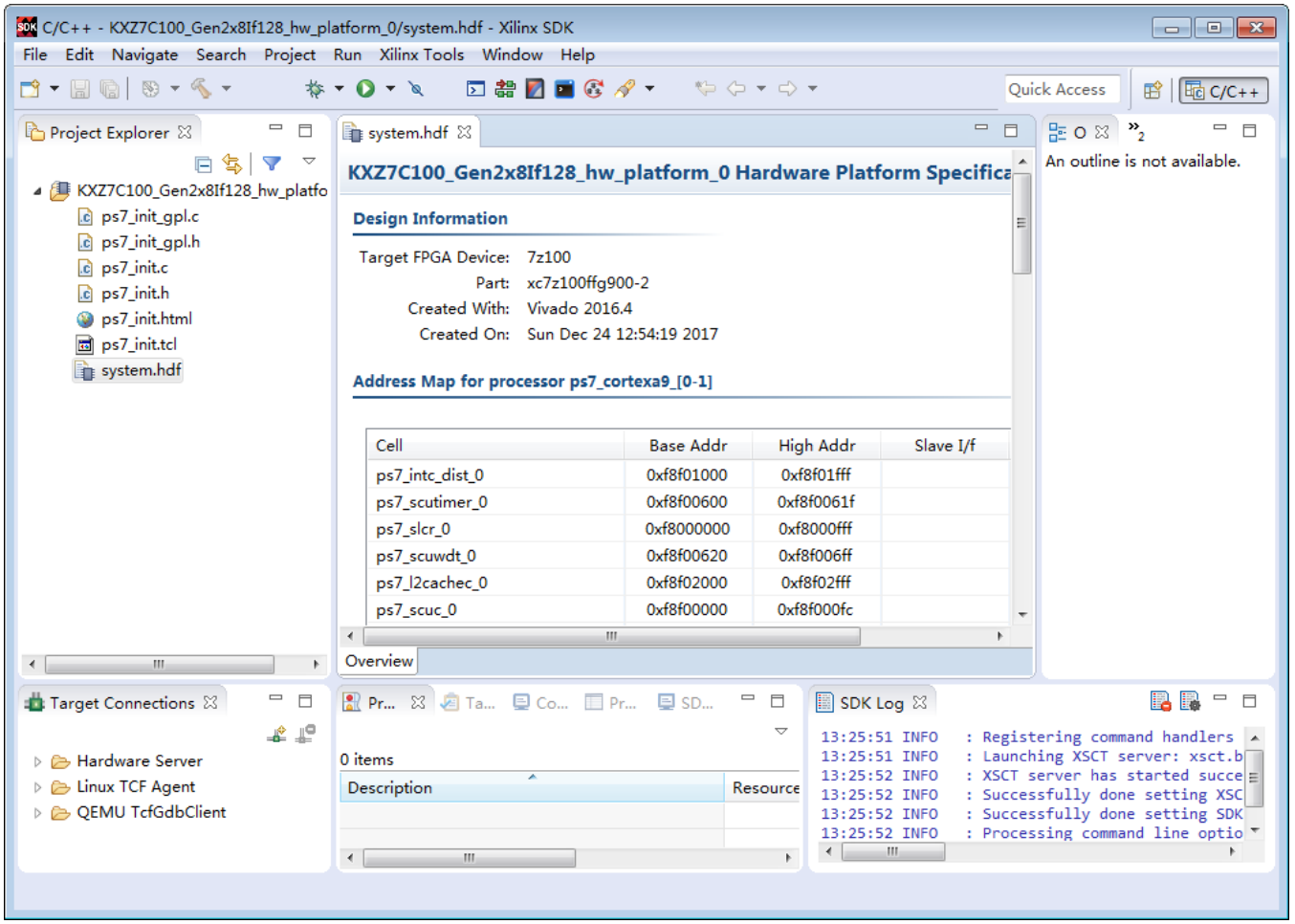
- Lauch SDK.





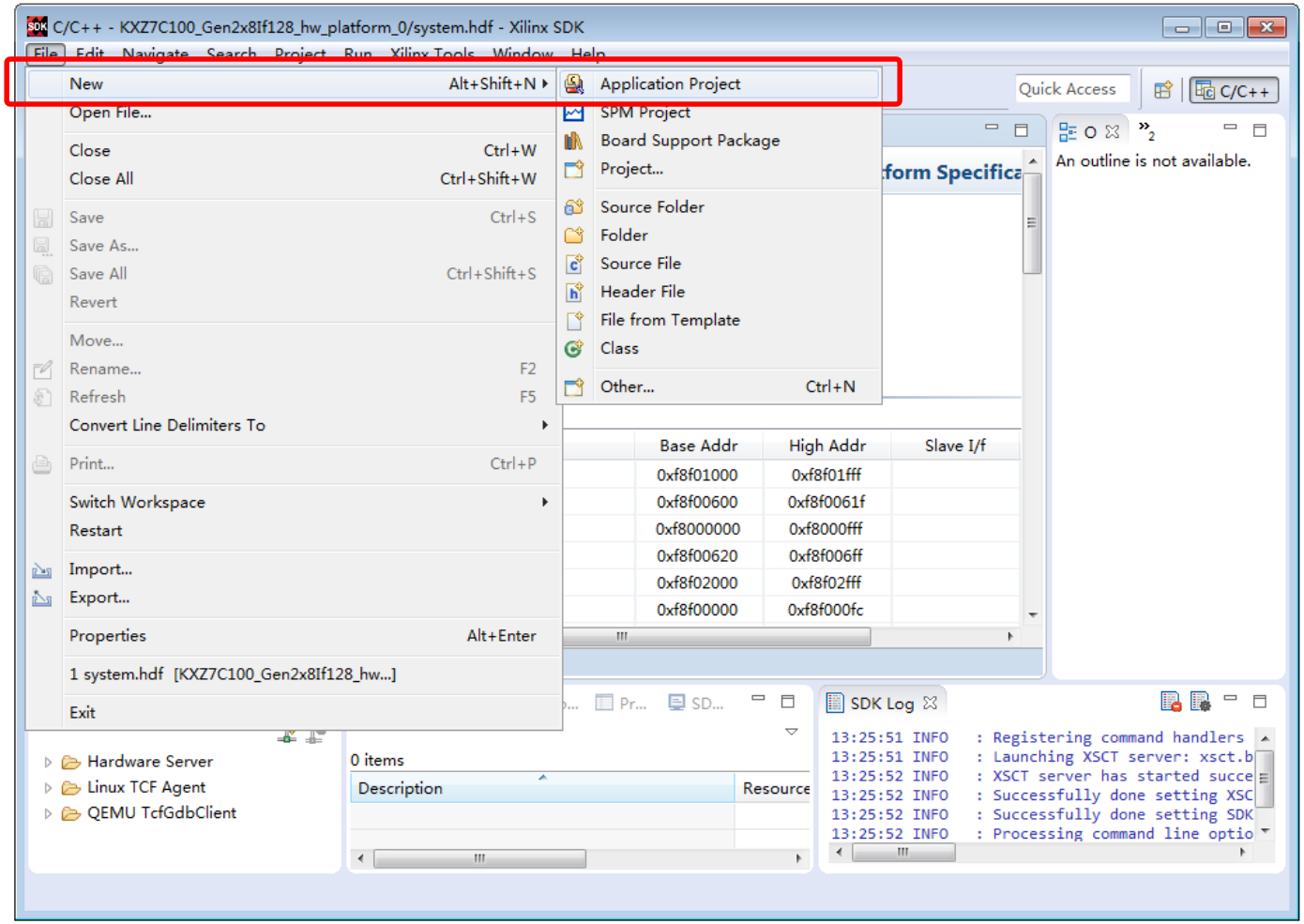
First Stage Boot Loader

- Launch SDK.



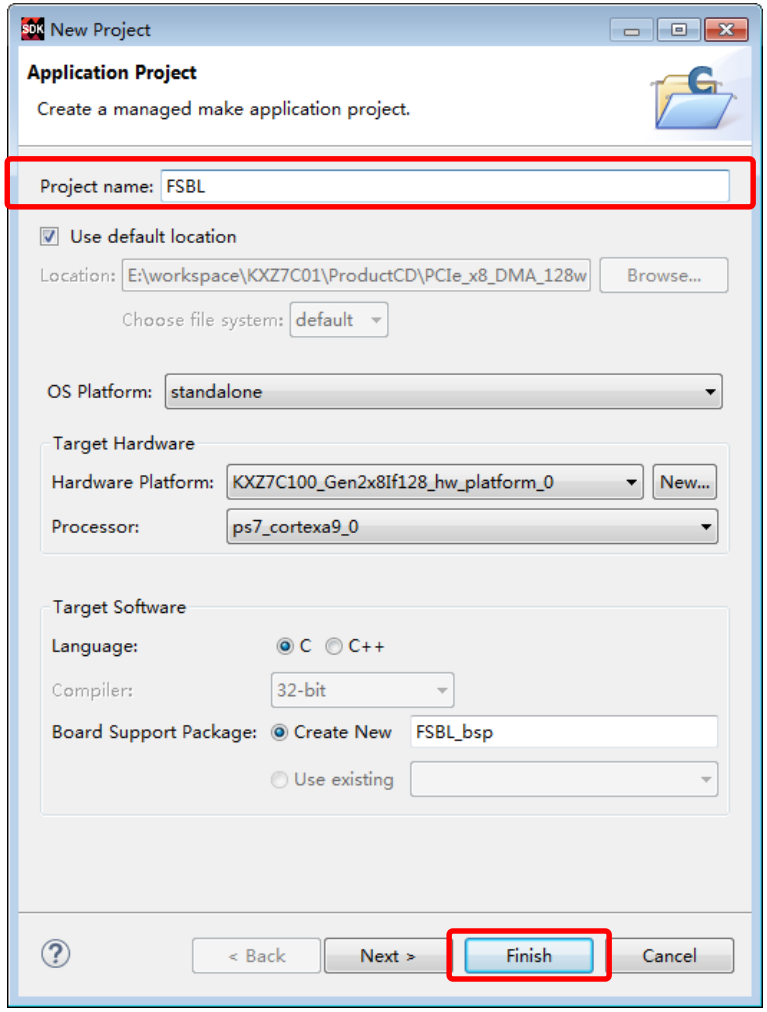
First Stage Boot Loader

- Create new Application Project.



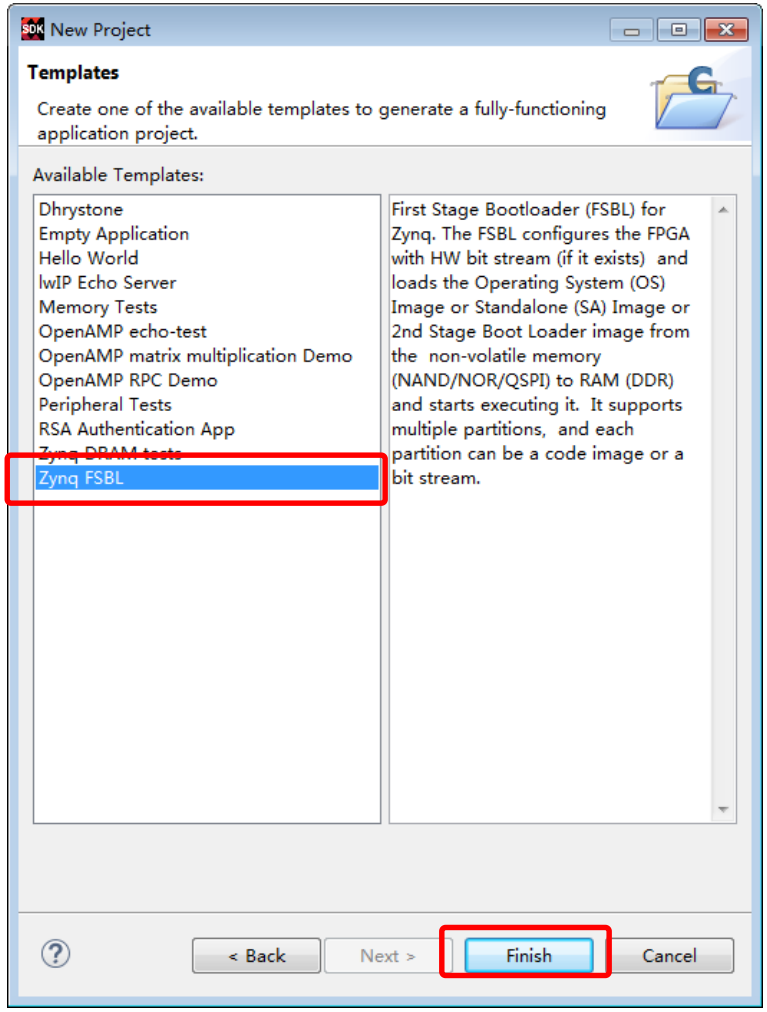
First Stage Boot Loader

- Specify the Application project name: FSBL.



First Stage Boot Loader

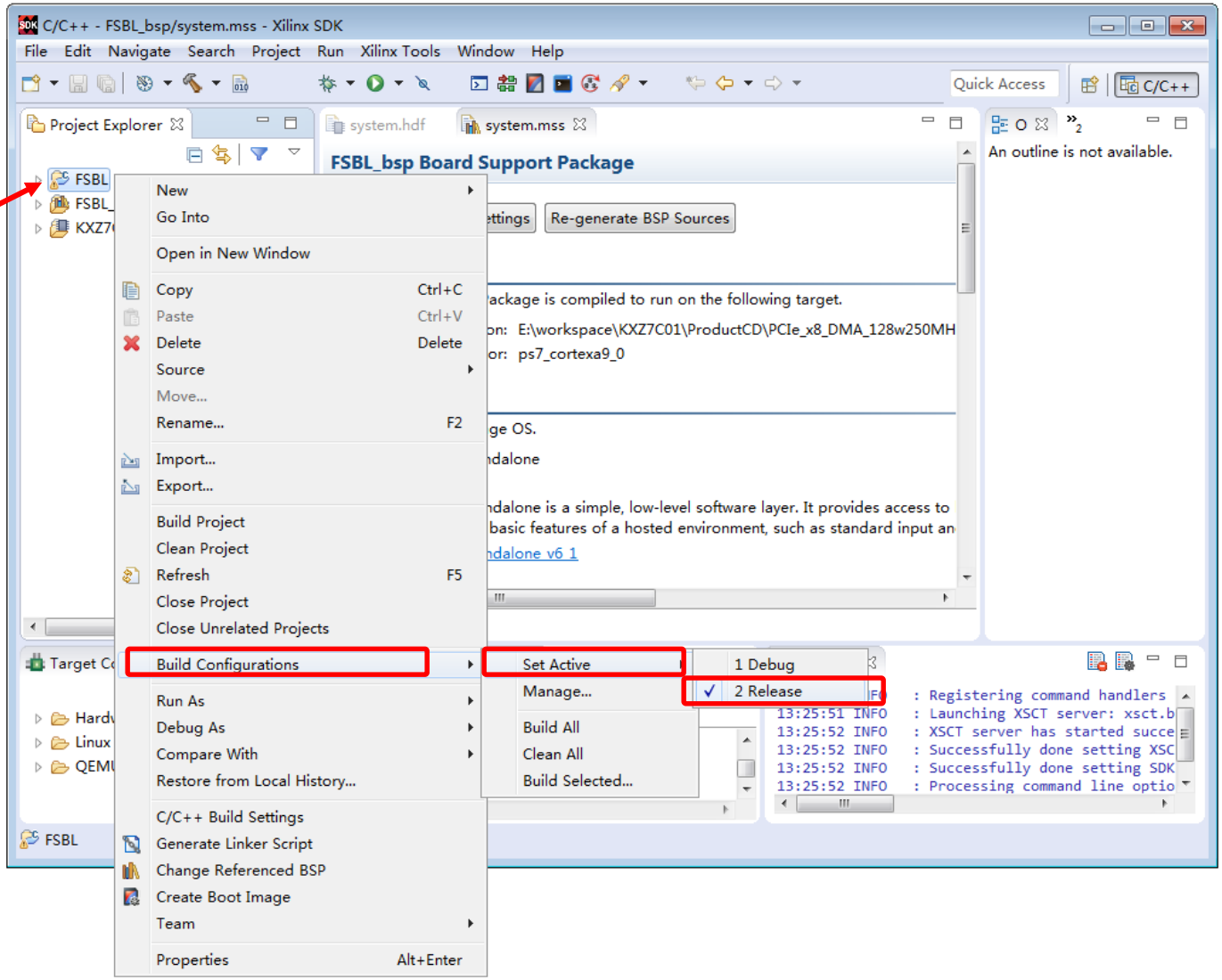
- Select the template: Zynq FSBL.



First Stage Boot Loader

- Set the build configuration as “Release”.

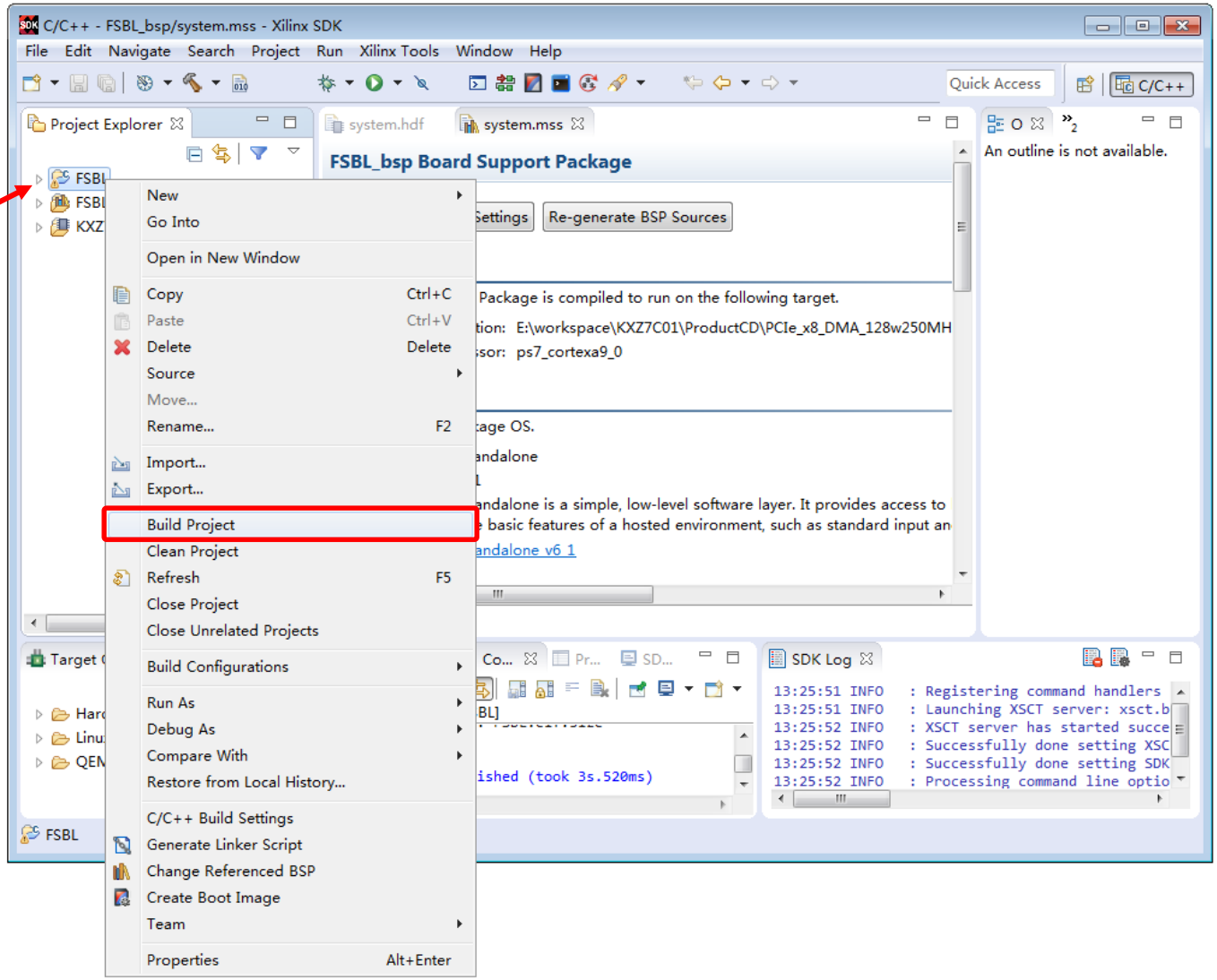
Right click on this application



First Stage Boot Loader

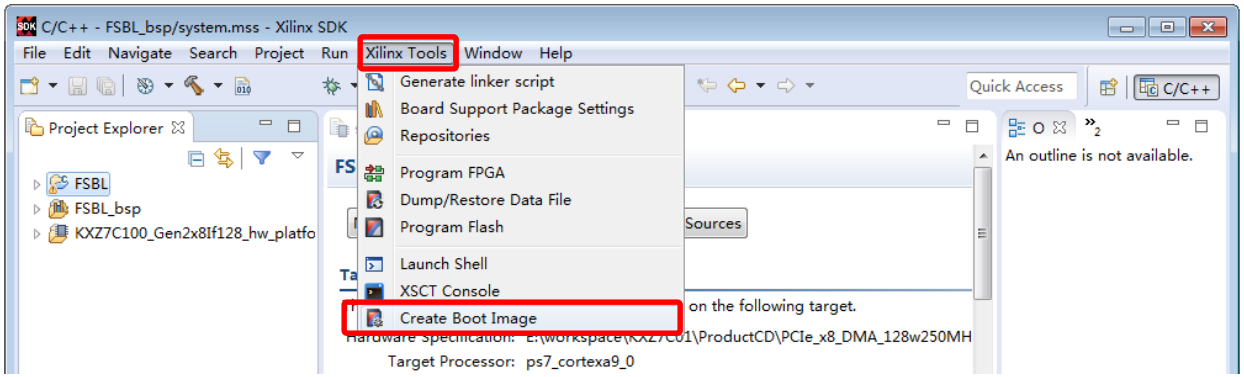
- Rebuild the application.

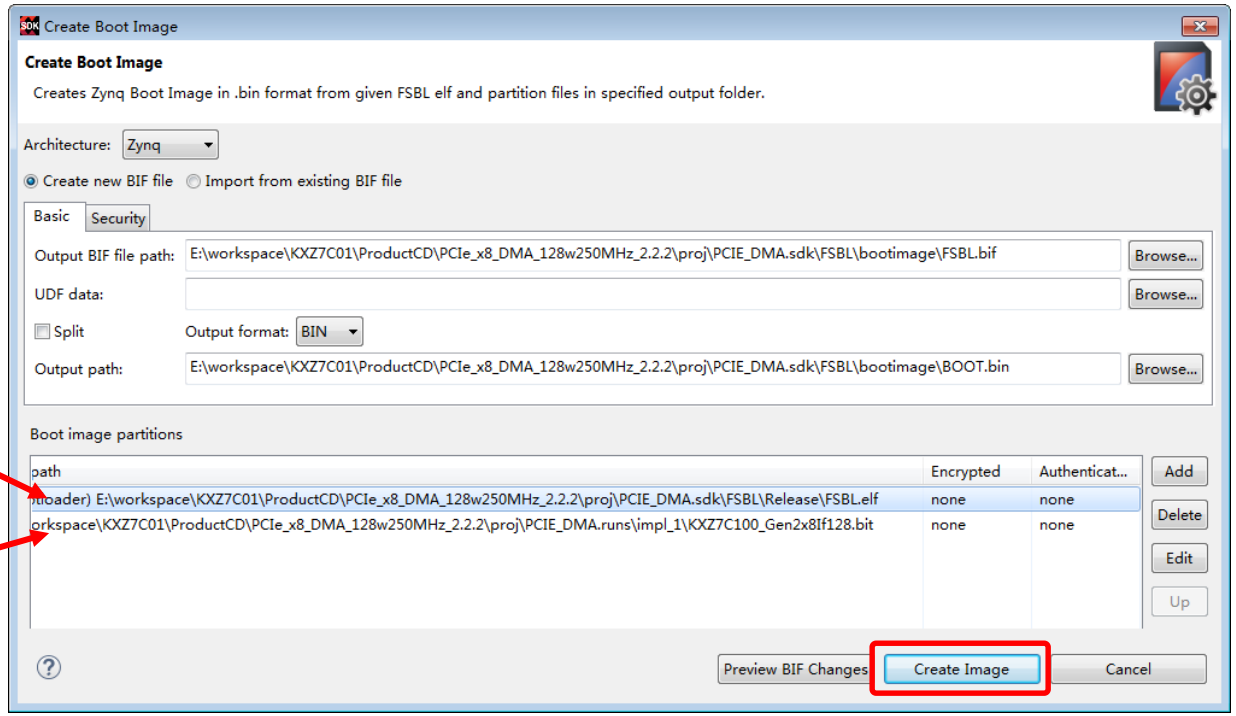
Right click on this application



Create Boot Image

- In SDK, select “Xilinx Tools → Create Boot Image”.





FSBL.elf

FPGA PL bitfile

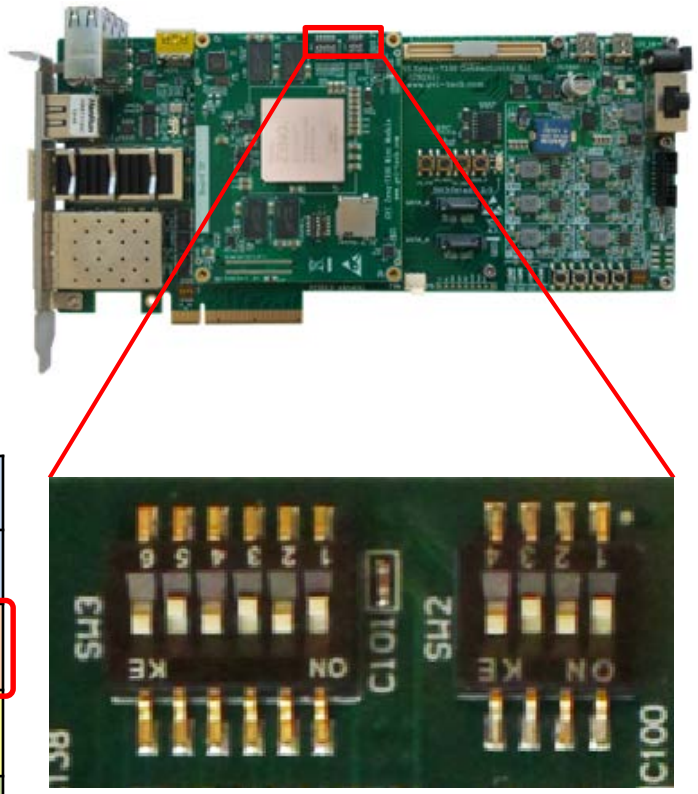
Hardware Setup

- Connect the JTAG cable to the base board.

- Connect the power supplier. You can use an ATX to DC5.5 adapter.

Set Boot Mode to JTAG

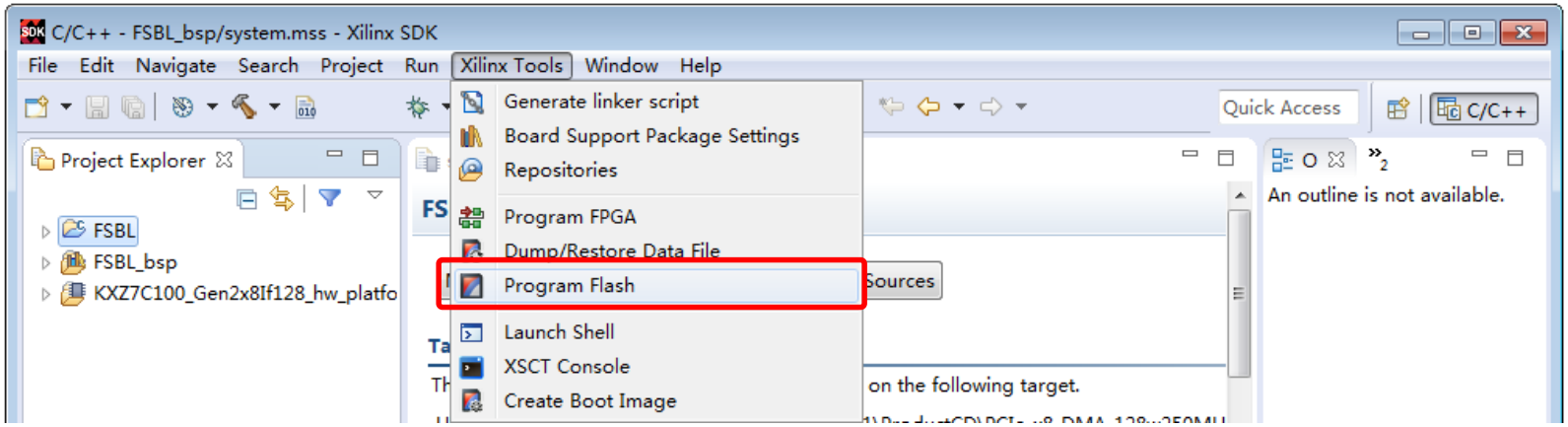
- Before writing the configuration file into Flash, we need to set the boot mode to JTAG mode
- Change the position of SW2 and SW3 to set the boot-mode to JTAG

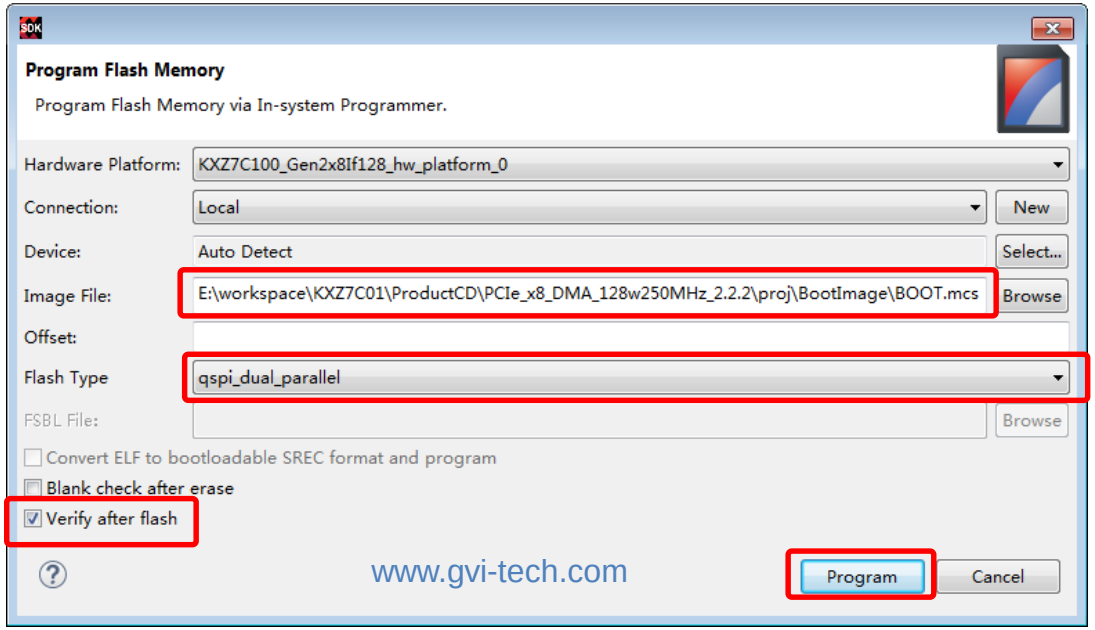


	SW2				SW3					
	1	2	3	4	1	2	3	4	5	6
JTAG	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
QSPI	ON	OFF	OFF	ON	OFF	ON	OFF	ON	OFF	ON
SD	ON	OFF	OFF	ON	OFF	ON	OFF	ON	ON	OFF

Configure QSPI Flash

- In SDK, select “Xilinx Tools → Program Flash”.





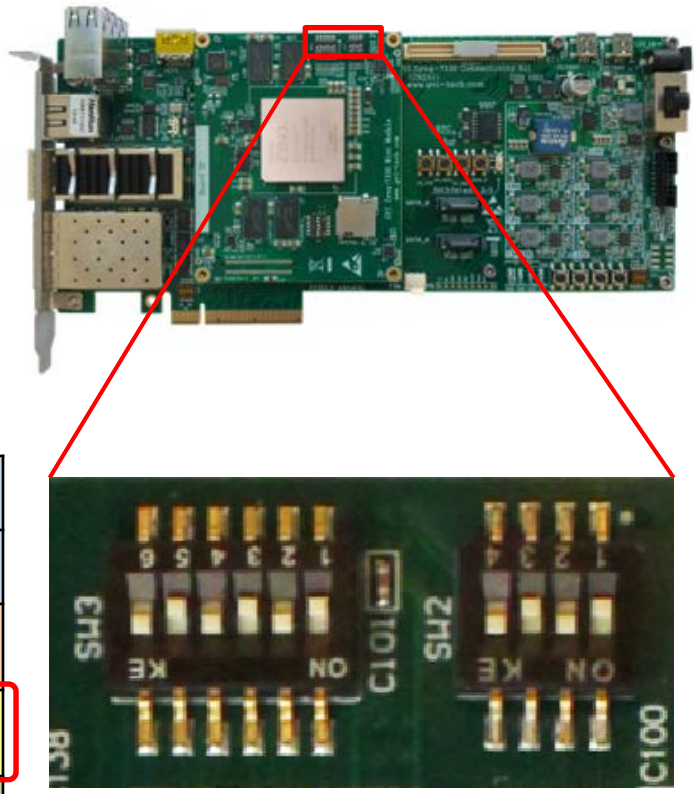
Power off the Board and then Restart

- The flashing process may take few minutes.
- After writing the mcs file into flash, you need to shut down you computer and then start the computer again. Because the new flash configuration file can only be loaded after a new power on sequence of FPGA.

Run the PCIe DMA Data Transfer Demo

Set Boot Mode to QSPI

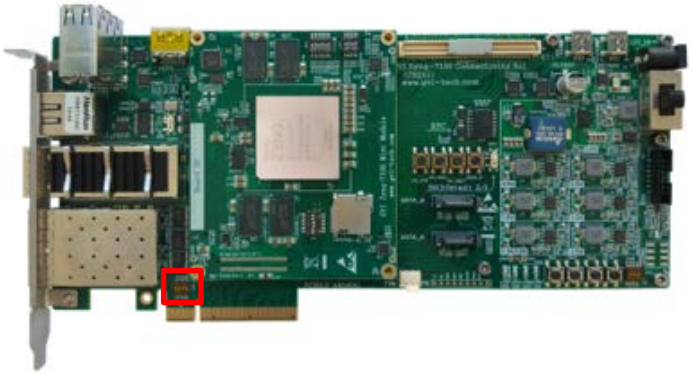
- After writing the configuration file into flash, we need to change the boot mode to QSPI, so it can boot from the flash.
- Change the position of SW2 and SW3 to set the boot-mode to QSPI



	SW2				SW3					
	1	2	3	4	1	2	3	4	5	6
JTAG	OFF	ON	OFF	ON	OFF	ON	OFF	ON	OFF	ON
QSPI	ON	OFF	OFF	ON	OFF	ON	OFF	ON	OFF	ON
SD	ON	OFF	OFF	ON	OFF	ON	OFF	ON	ON	OFF

Run the Demo – Hardware Setup

- Put the switch SW6 to the position of X8 mode.



- Turn off the PC. Insert the board into a PCIe slot. Connect the power supplier to ATX. You can use an ATX to DC5.5 adapter.

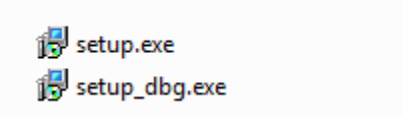
- Turn on the power switch.

Run the Demo – Install Driver

- The driver for the RIFFA reference design can be obtained from the RIFFA website:

  http://riffa.ucsd.edu/

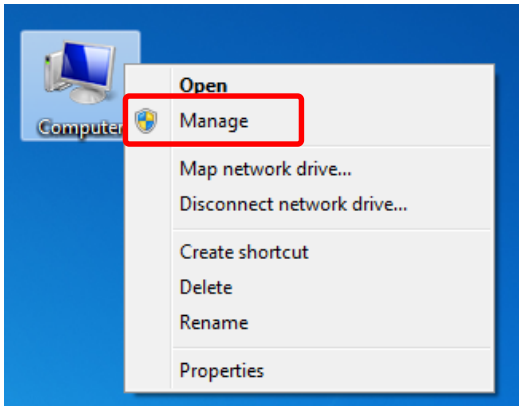
- We also put a copy of the driver in the product CD (driver_software\PCIe_DMA_Driver\windows\win7).



- Install the kernel driver and C/C++ library by running the setup.exe or setup_dbg.exe installer (the debug installer outputs additional debug messages to the Windows debug framework).
- After running the installer, reboot to let the system find your RIFFA 2.2 design on the PCIe bus and let the OS load your driver.

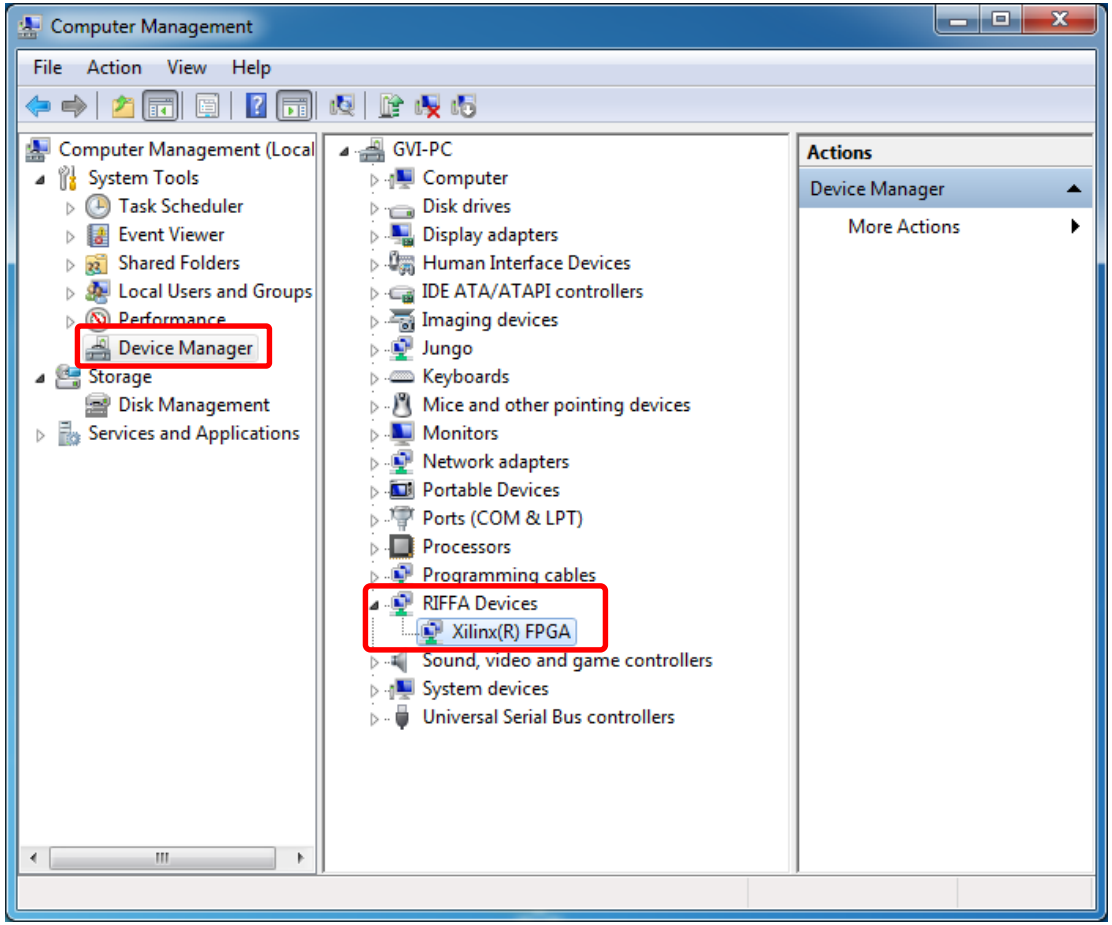
Run the Demo – Install Driver

- After restart your computer, right click on “Computer”, and then select “Manage”.



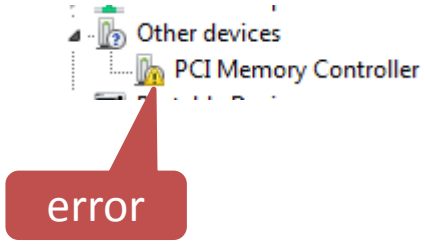
Run the Demo – Install Driver

- Under “Device Manager”, you should be able to see the RIFFA device.



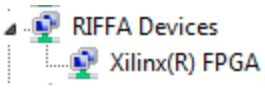
Run the Demo – Bug Fix

- In computer, one may experience problem with the default driver. This an issue with the RIFFA driver. Please check with RIFFA website for new version.
- We have a work around before a new RIFFA release, please refer to page 50 for the work around.



Run the RIFFA Test Utils

- When the driver is installed correctly, you can start to use the RIFFA library.



- Open a command window, and switch to the RIFFA installation directory.

Run the RIFFA Test Utils

- Type the following command:

  testutil.exe 0

- The output should similar to:

Bandwidth Benchmarking

- Type the following command:

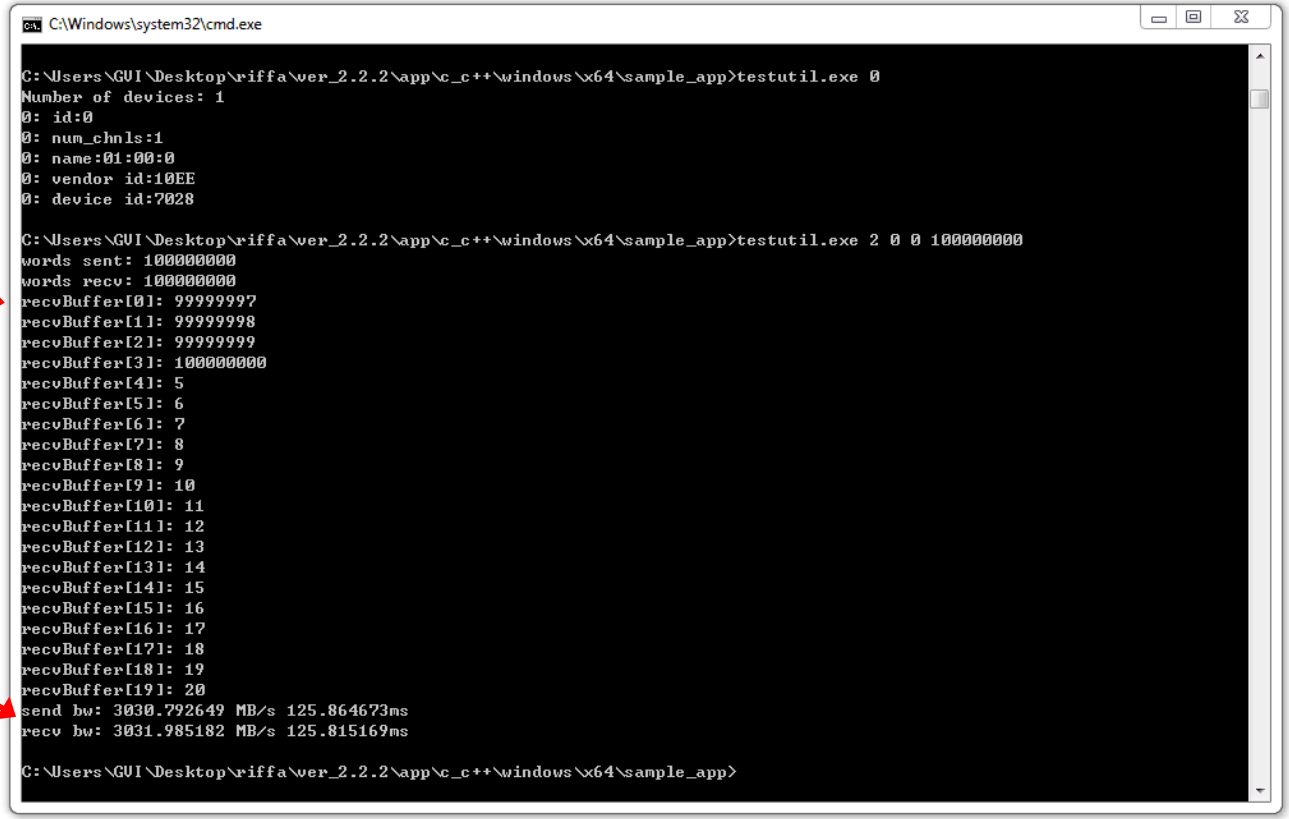
testutil.exe 2 0 0 10000000

Bandwidth-Test | FPGA ID | Channel ID | Number of words to transfer

- The output should similar to:

The last four words

Data transfer bandwidth. This value depends on the computer.



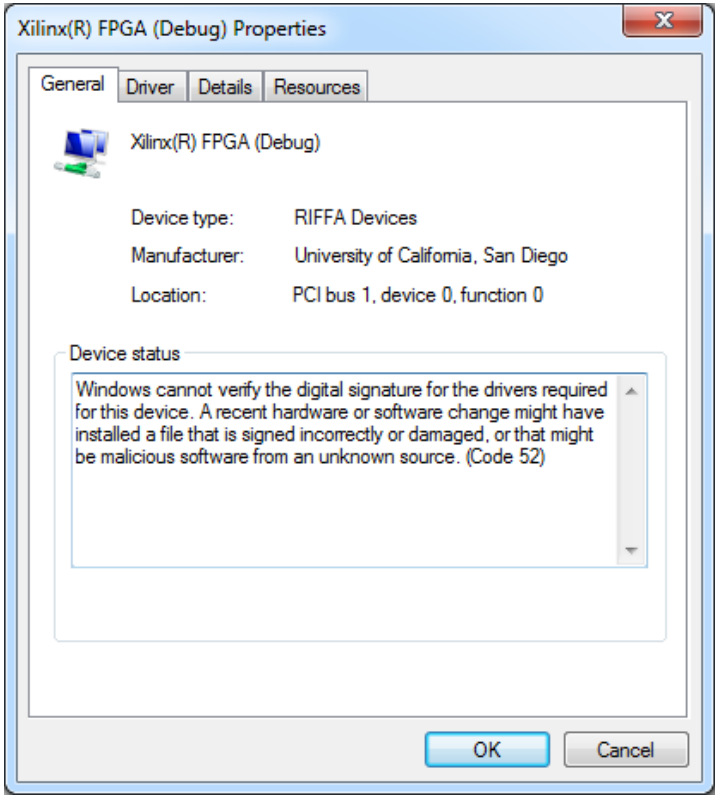
Reference

- [1] Xilinx 7 Series Gen2 Integrated Block for PCI Express (PCIe): http://www.xilinx.com/products/intellectual-property/7_series_pci_express_block.html
- [2] RIFFA 2.2: http://riffa.ucsd.edu/
- More design resources can be found :

  www.gvi-tech.com
- How to Buy：
  - https://detail.tmall.com/item.htm?id=562769965498

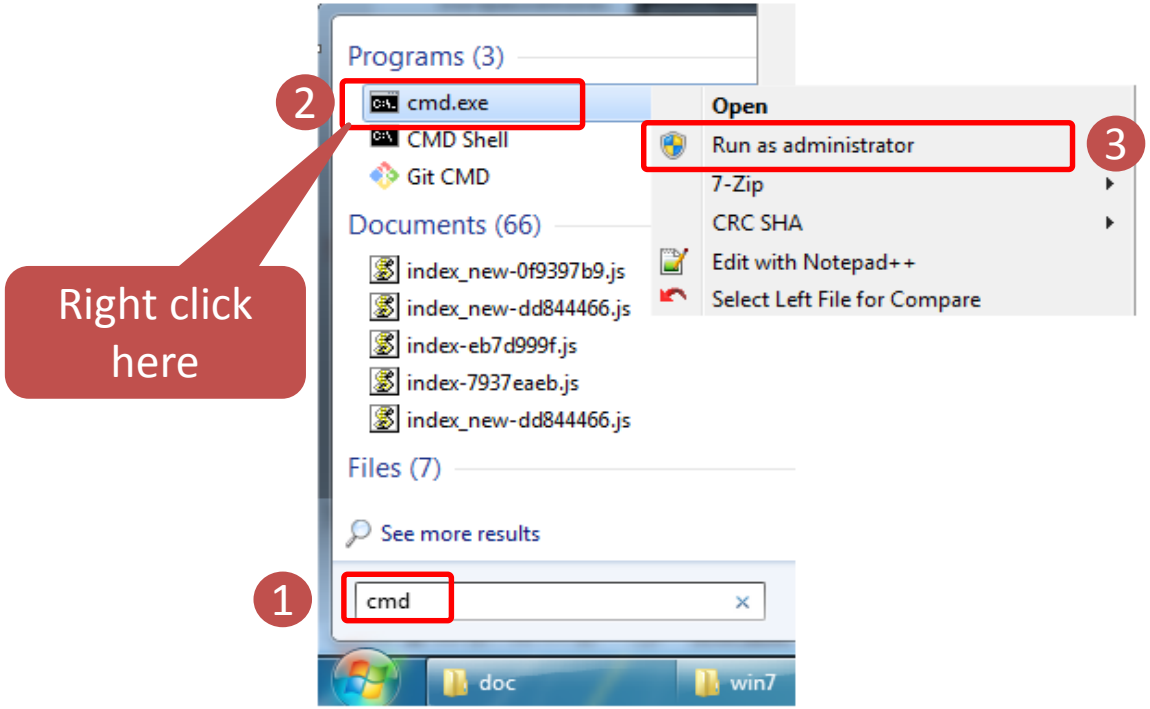
Workaround: Problem Description

- By default, all 64-bit Windows versions, starting from Windows 7, prohibit to install drivers of the devices that do not have a valid digital signature. The digital signature guarantees (to some extent) that the driver has been issued by a certain developer or vendor, and its code hasn't been modified after it was signed.
- However, some computer has the problem to verify the digital signature of the RIFFA driver, and thus the driver does not work properly.



Workaround: Disable Digital Signature

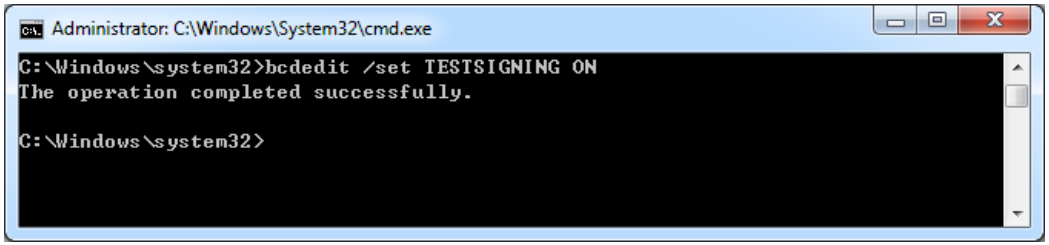
- In order to avoid this problem until the new release of RIFFA with bugfix, we need to disable Windows Driver Digital Signature as a workaround.
- Click **Start**, and then type *cmd* in the **Search** box. Under **Programs,** right-click **cmd.exe**, and then click **Run as administrator**.



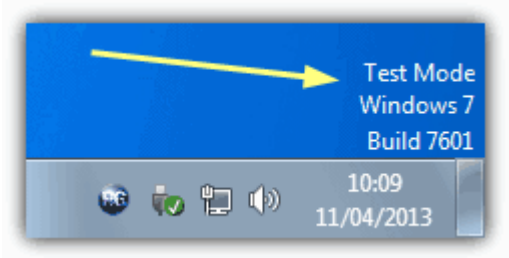
Workaround: Disable Digital Signature

- At the command prompt, type the following text, and then press Enter:

  bcdedit /set TESTSIGNING ON

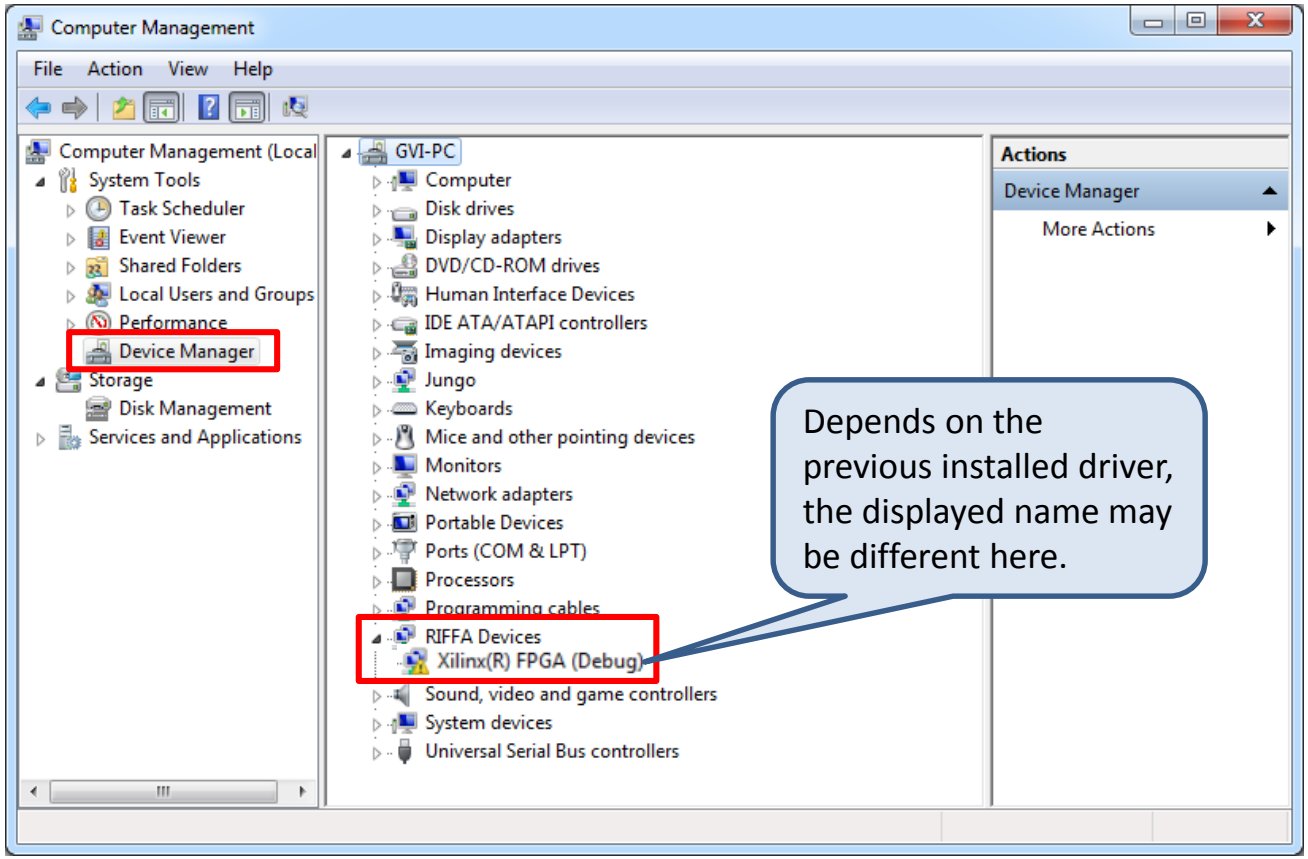


- After enabling Test Mode using one of the above options, you will notice that there is a watermark above the clock at the bottom right of the screen saying "Test Mode, Windows **, Build **

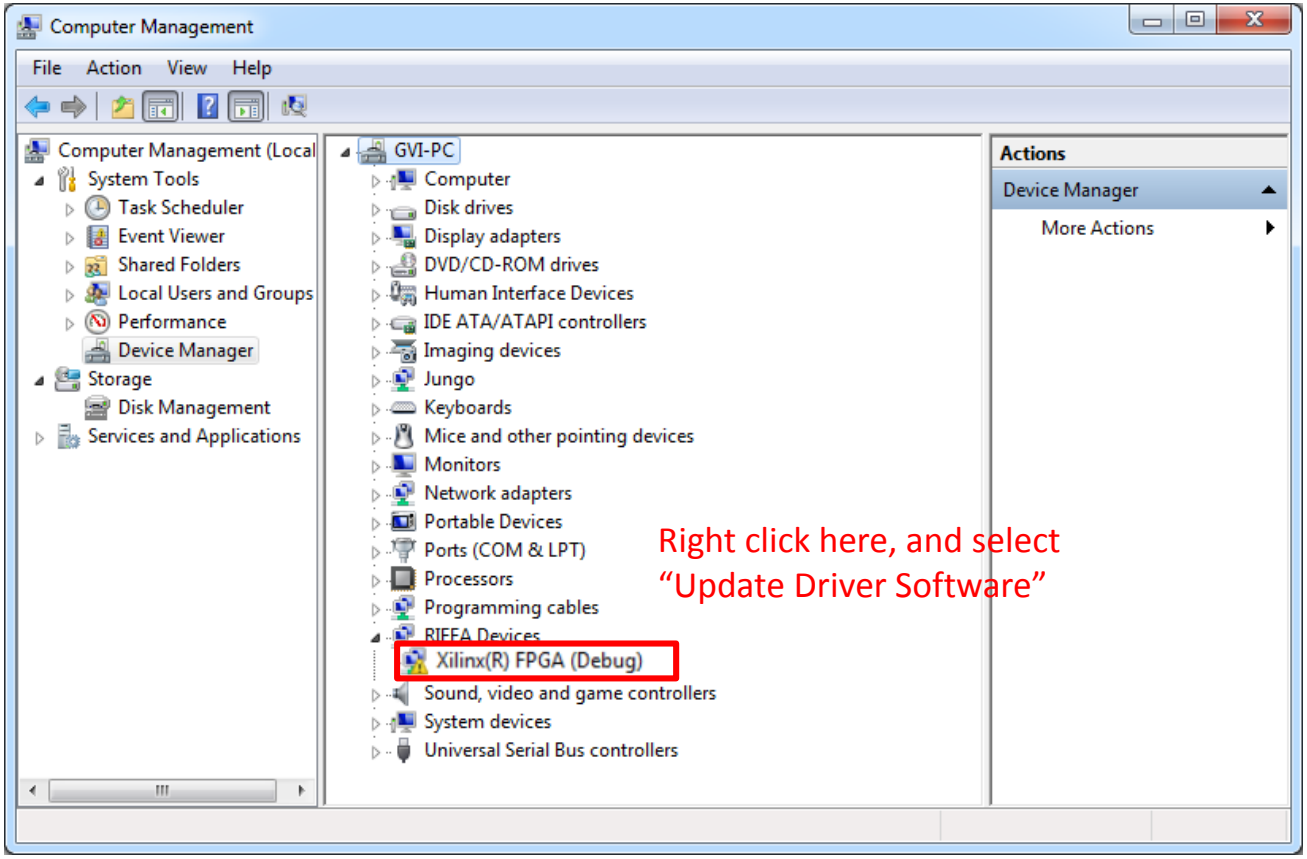


Workaround: Install Driver

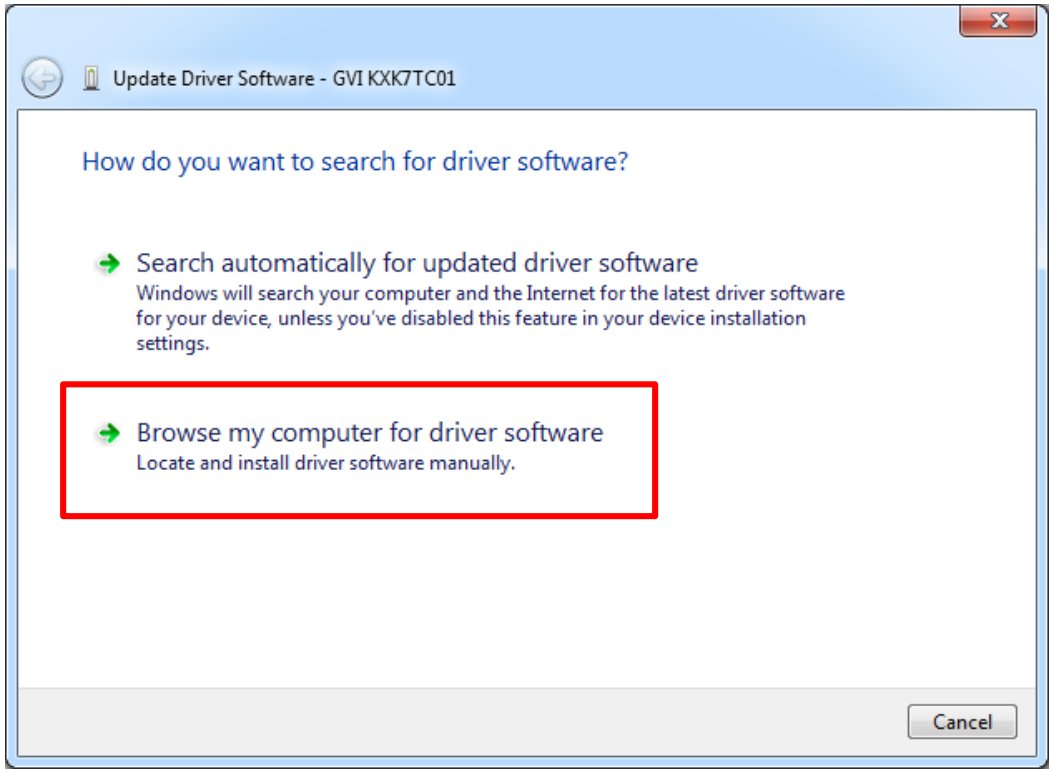
- Right click on “Computer”, select “Manage”.
- Click “Device Manager”, you will see the RIFFA Device in the List.



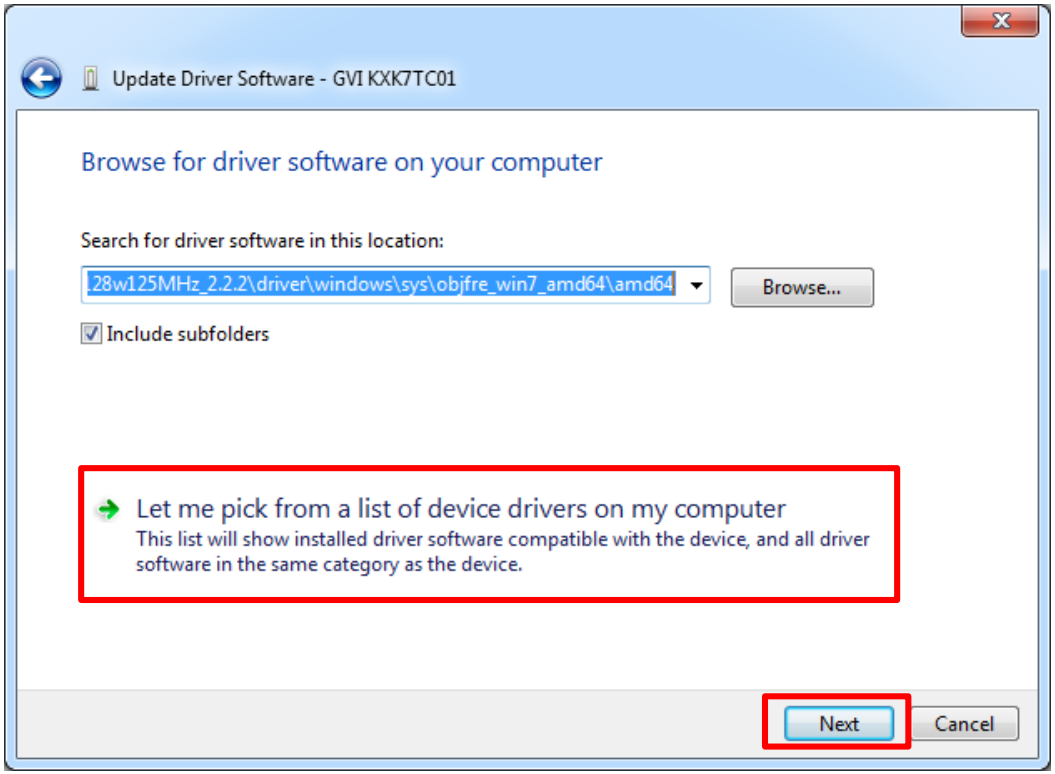
Workaround: Install Driver



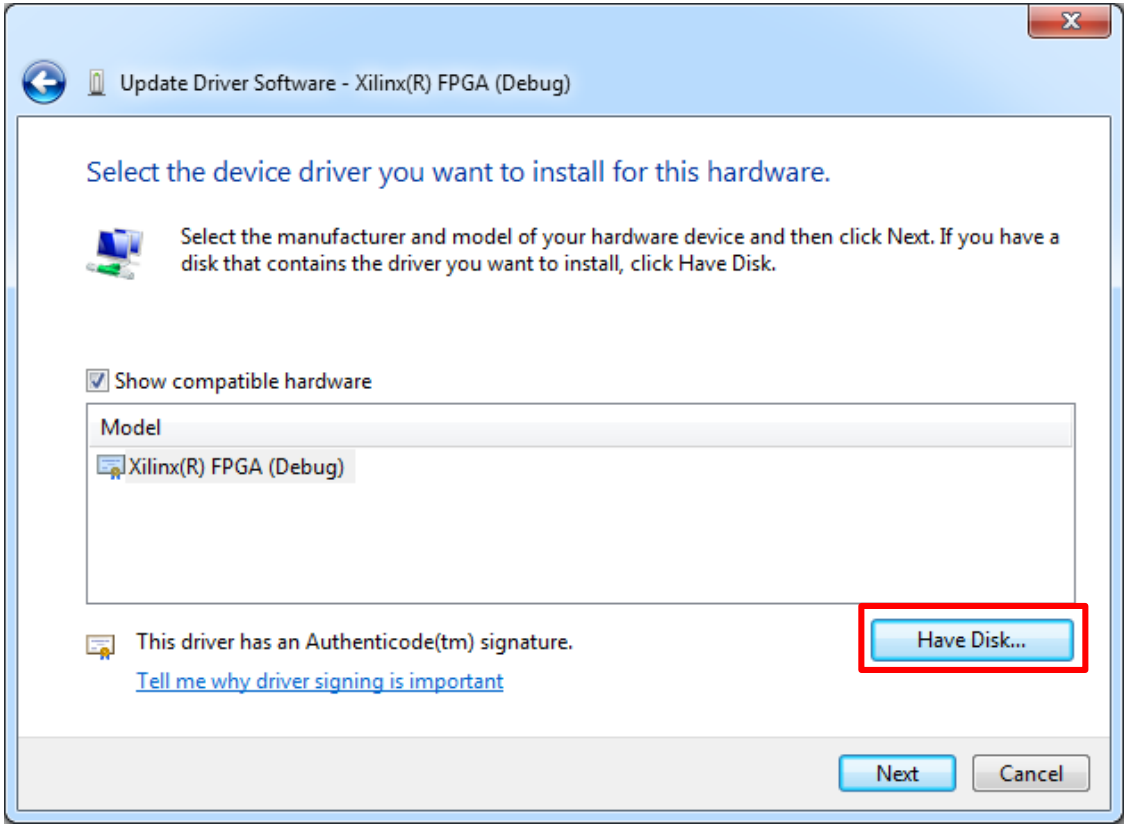
Workaround: Install Driver



Workaround: Install Driver

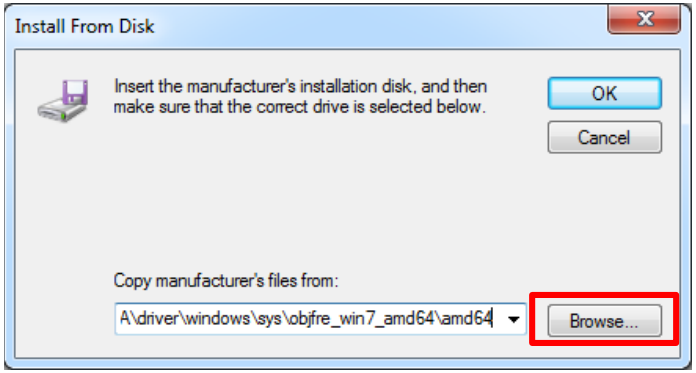


Workaround: Install Driver

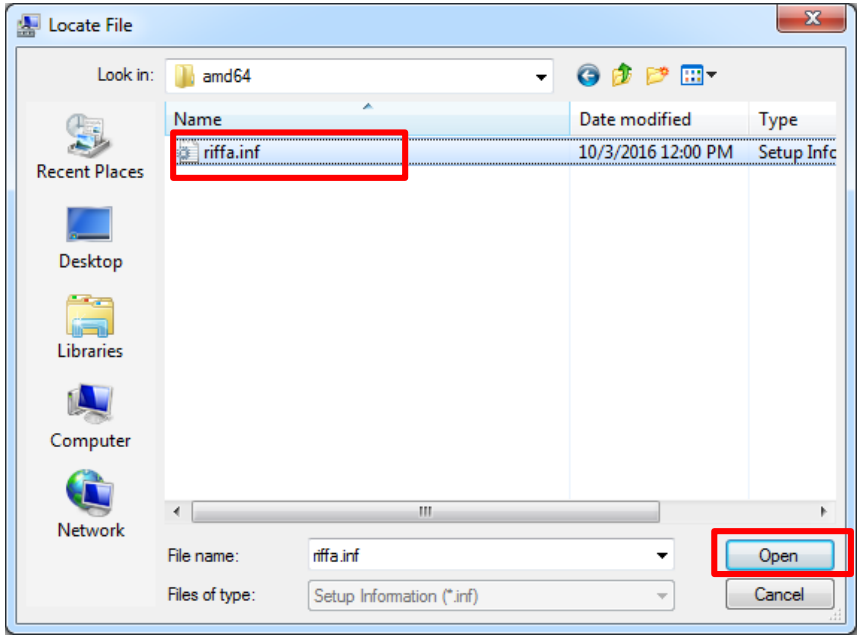


Workaround: Install Driver

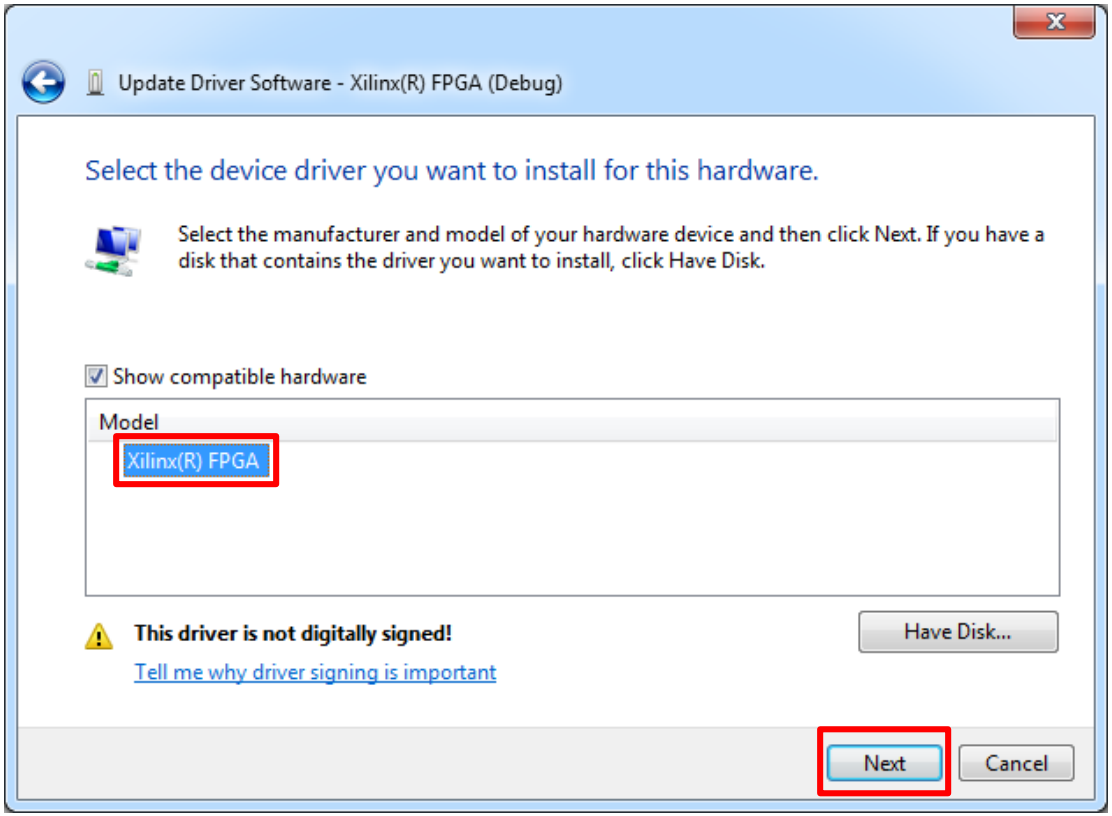
- The driver patch is provided in the folder *driver_patch/windows/x64*



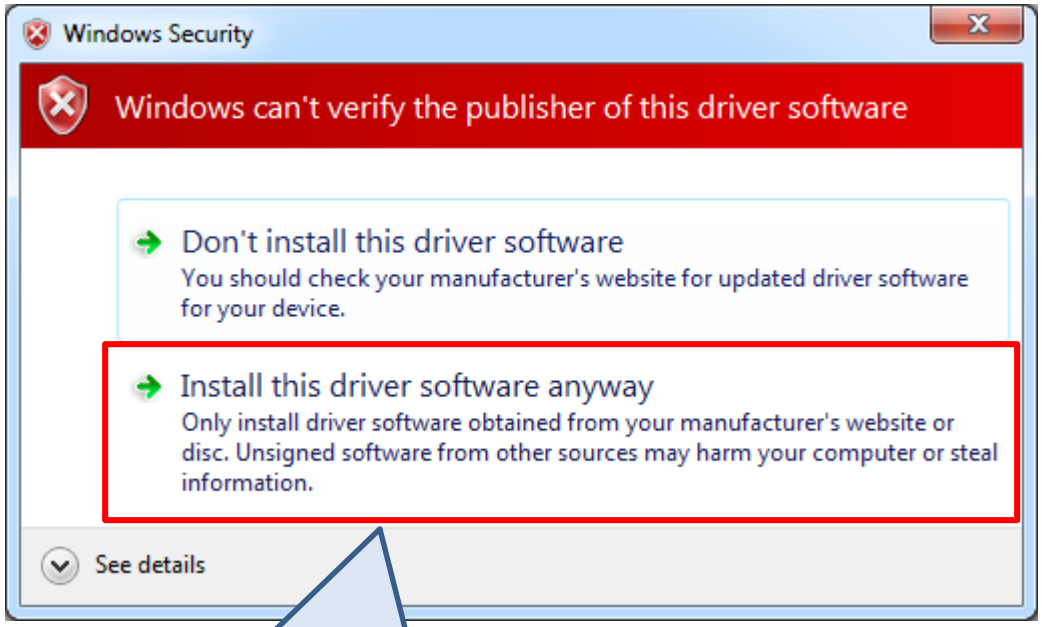
Select the provided driver



Workaround: Install Driver

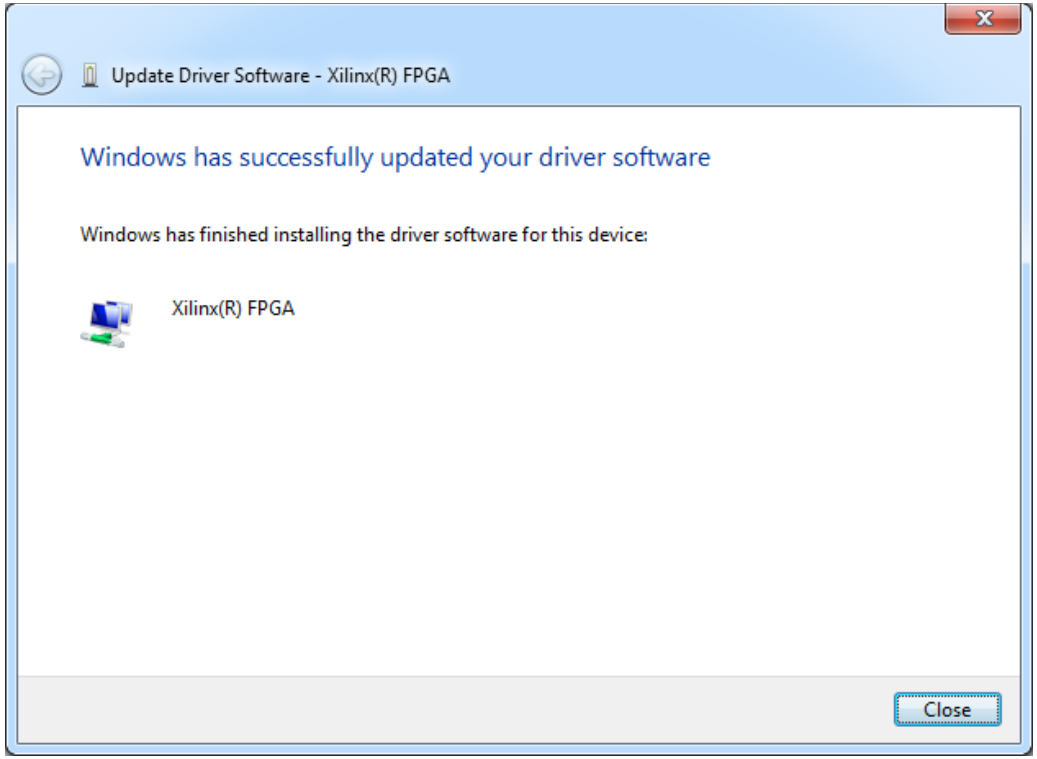


Workaround: Install Driver



In Test Mode, we should select the second item.

Workaround: Install Driver



Workaround: Install Driver

- Copy the generated “riffa.dll” (in the folder *app/c_c++/windows/x64*) to the folder “C:\Windows\System32”. If this file already exist in the destination folder, replace the existing one with the new one.
- Now you should be able to use the RIFFA library.