

# Extending Embedded System into PL

Zynq  
Vivado 2017.1 Version



# Objectives

- > **After completing this module, you will be able to:**
  - >> Identify the IP supplied as part of Vivado
  - >> Describe how to add hardware to an existing Vivado project
  - >> Explain how IP is added to extend processing system functionality



# Outline

- > ***IP Catalog***
- > IP directory
- > IP device files
- > GP Interfaces
- > Adding IP to extend PS into PL
- > Bitstream generation
- > Summary



# The PS and the PL

## > The Zynq-7000 AP SoC architecture consists of two major sections

### >> PS: Processing system

- Dual ARM Cortex-A9 processor based
- Multiple peripherals
- Hard silicon core

### >> **PL: Programmable logic**

- Shares the same 7 series programmable logic as
  - Artix™-based devices: Z-7010, Z-7015 and Z-7020 (high-range I/O banks only)
  - Kintex™-based devices: Z-7030, Z-7035, Z-7045, and Z-7100 (mix of high-range and high-performance I/O banks)

## > This section focuses on the PL

# Communicating with PL

## > Processing system master

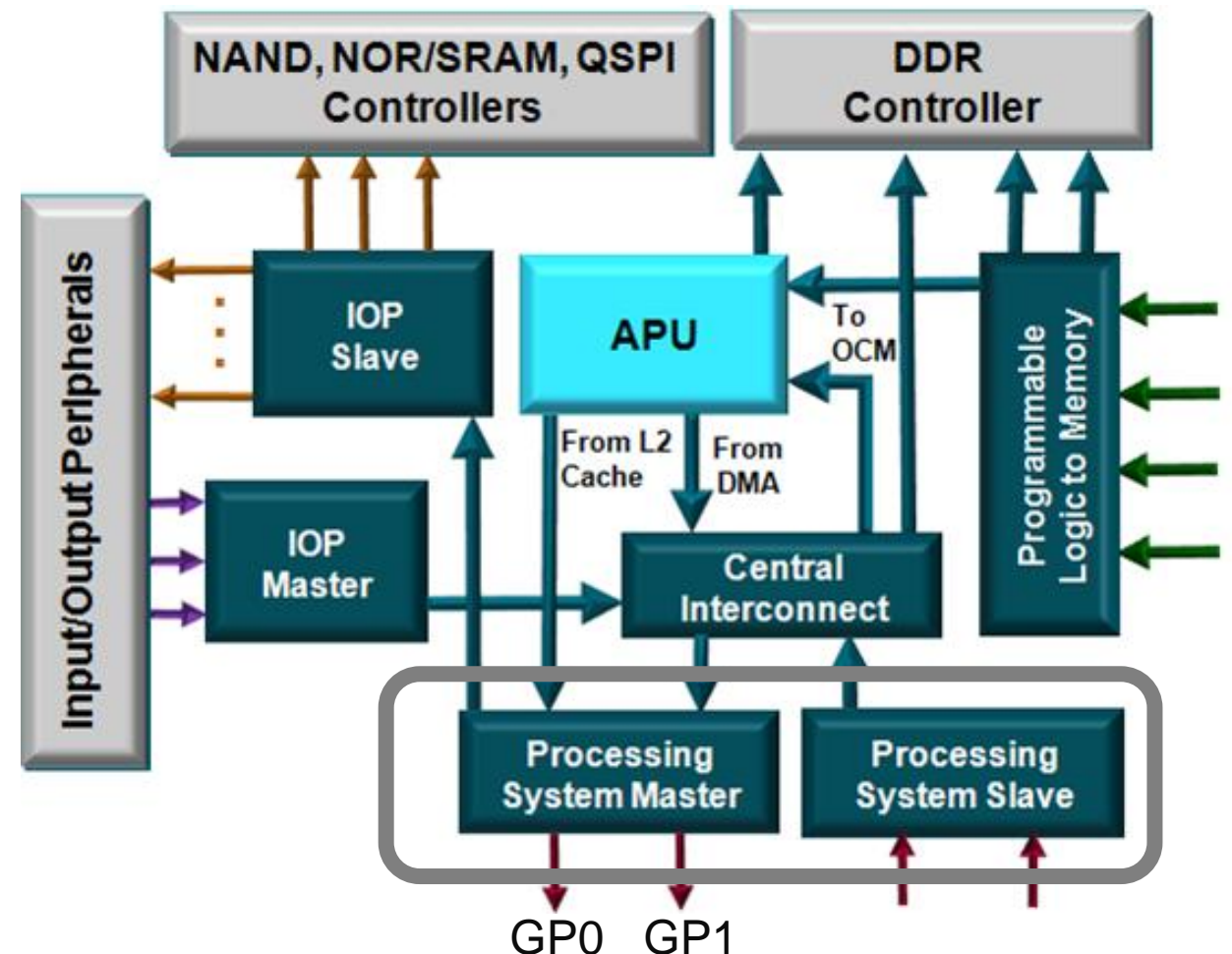
- >> Two ports from the processing system to programmable logic
- >> Connects the CPU block to common peripherals through the central interconnect

## > Processing system slave

- >> Two ports from programmable logic to the processing system

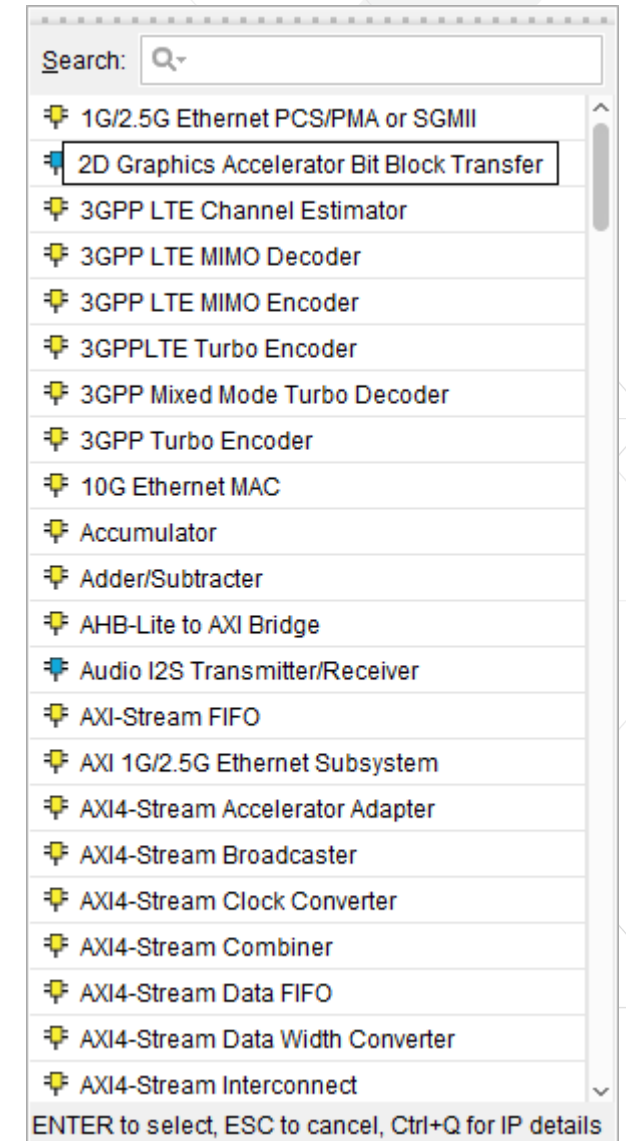
## > Slave PL peripherals address range

- >> 4000\_0000 and 7FFF\_FFFF (connected to GP0) and
- >> 8000\_0000 and BFFF\_FFFF (connected to GP1)



# IP Catalog

- > The IP Catalog contains a collection of IP that can be used to assemble the (embedded) system
- > Supported by IPI
- > Facilitates quick system construction
- > Each IP block has its own configuration parameters
- > Most of the IP is free, some require licenses
- > Stored as source code in the install directory
  - >> Always synthesized with the latest tools
  - >> Some proprietary source code is encrypted
- > Peripherals in the PS are always present and can be dynamically enabled or disabled through PS Configuration wizard



# IP Peripherals Included as Source (Free)

## > **Bus and bridge controllers**

- >> AXI to AXI connector
- >> Local Memory Bus (LMB)
- >> AXI Chip to Chip
- >> AHB-Lite to AXI
- >> AXI4-Lite to APB
- >> AXI4 to AHB-Lite

## > **Debug cores**

- >> Integrated Logic Analyzer

## > **DMA and Timers**

- >> Watchdog, fixed interval

## > **Inter-processor communication**

## > **External peripheral controller Memory and memory controller**

## > **High-speed and low-speed communication peripherals**

- >> AXI 10/100 Ethernet MAC controller
- >> Hard-core tri-mode Ethernet MAC
- >> AXI IIC
- >> AXI SPI
- >> AXI UART

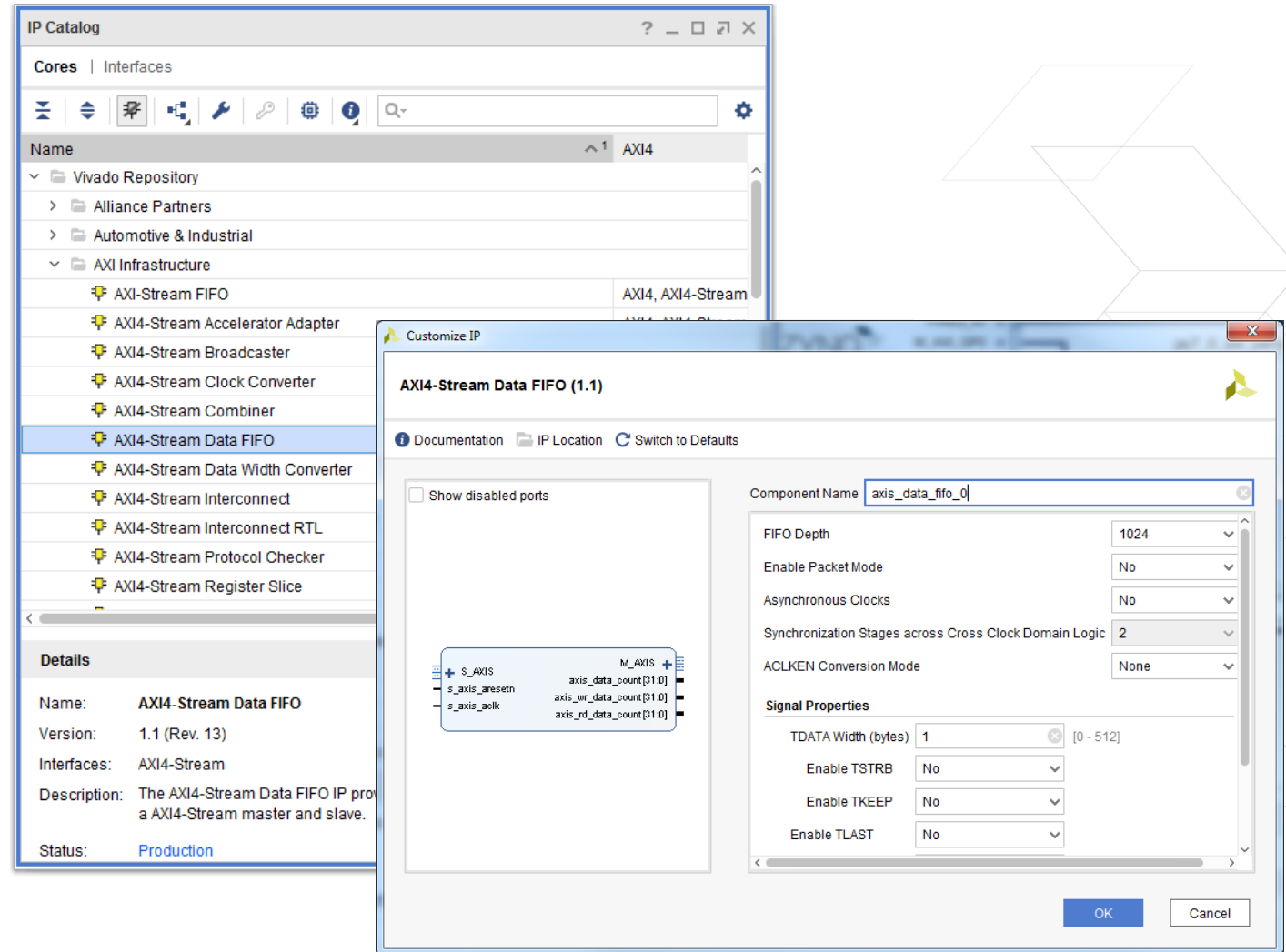
## > **Other cores**

- >> System monitor
- >> Xilinx Analog-to-Digital Converter (XADC)
- >> Clock generator, System reset module
- >> interrupt controller
- >> Traffic Generator, Performance monitor

# Vivado IP Catalog

## > Integrated IP Support

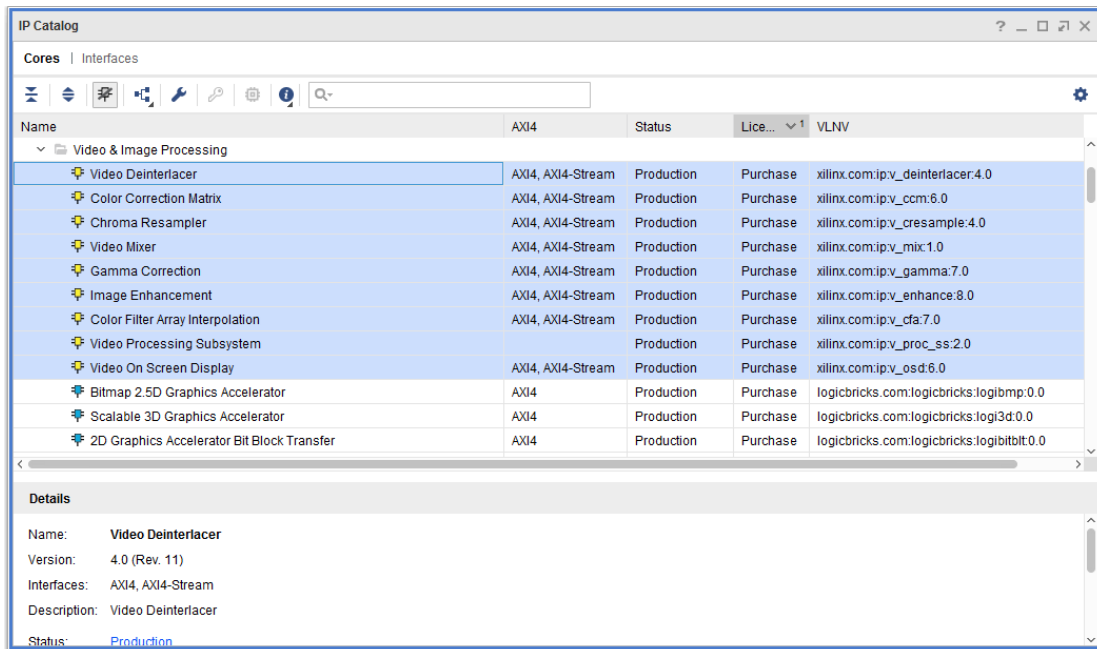
- >> Instant access to IP customization
- >> Vivado IP GUI look and feel
- >> Support for Vivado synthesis and implementation
- >> Selectable IP output products
- >> Full Tcl support





# IP Cores Included as Evaluation

- > Xilinx IP
- > 3<sup>rd</sup> Party IP

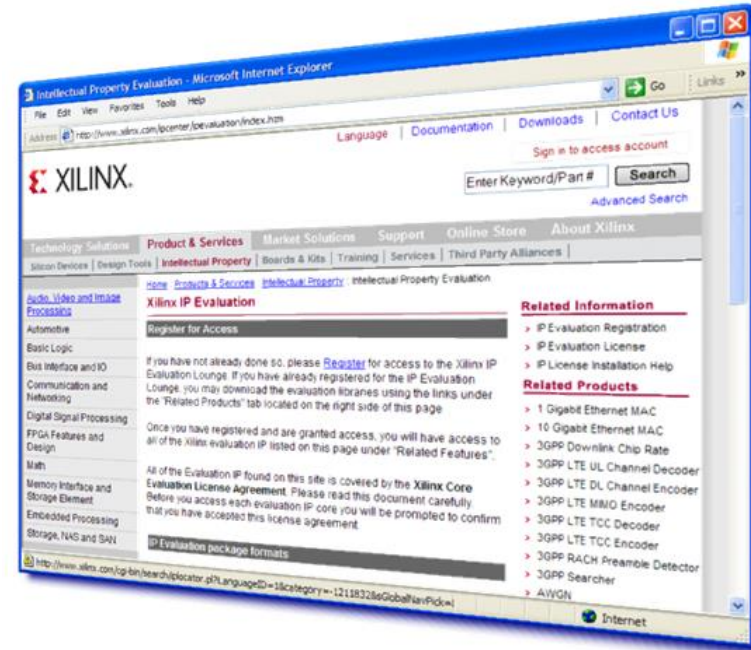


The screenshot shows the Xilinx IP Catalog interface. A table lists various Video & Image Processing cores. The table has columns for Name, AXI4, Status, License, and VLN#.

Name	AXI4	Status	Lice...	VLN#
Video Deinterlacer	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_deinterlacer:4.0
Color Correction Matrix	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_ccm:6.0
Chroma Resampler	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_cresample:4.0
Video Mixer	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_mixer:1.0
Gamma Correction	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_gamma:7.0
Image Enhancement	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_enhance:8.0
Color Filter Array Interpolation	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_cfa:7.0
Video Processing Subsystem	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_proc_ss:2.0
Video On Screen Display	AXI4, AXI4-Stream	Production	Purchase	xilinx.com:ip_v_osd:6.0
Bitmap 2.5D Graphics Accelerator	AXI4	Production	Purchase	logicbricks.com:logicbricks:logibmp:0.0
Scalable 3D Graphics Accelerator	AXI4	Production	Purchase	logicbricks.com:logicbricks:logi3d:0.0
2D Graphics Accelerator Bit Block Transfer	AXI4	Production	Purchase	logicbricks.com:logicbricks:logibitb:0.0

Details for Video Deinterlacer:

- Name: Video Deinterlacer
- Version: 4.0 (Rev. 11)
- Interfaces: AXI4, AXI4-Stream
- Description: Video Deinterlacer
- Status: Production

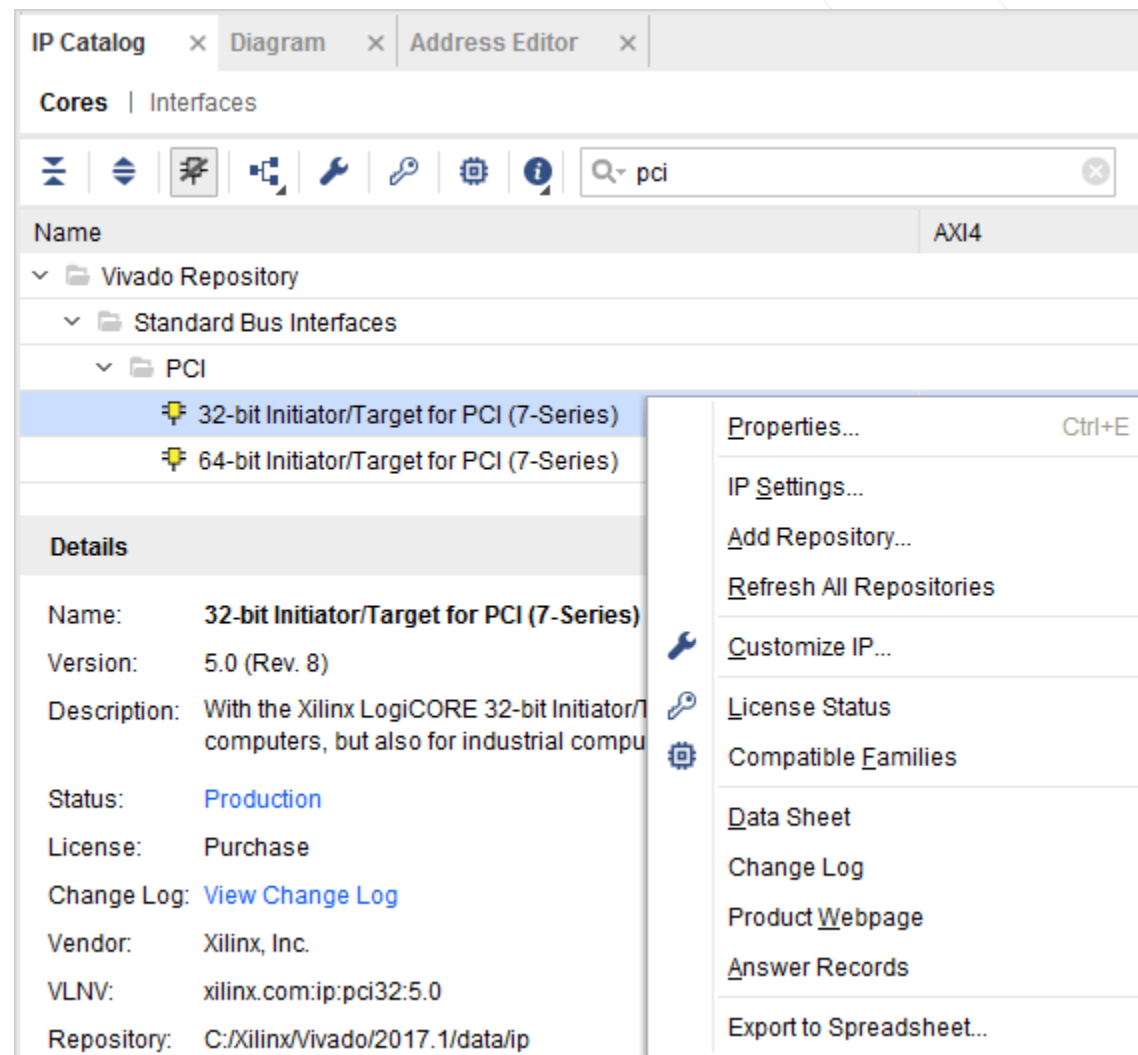


Xilinx developed, delivered, and supported Evaluation IP installs with a 90-day evaluation license

# IP Cores

## > Right click to

- >> Add/customize
  - IP Settings
- >> Check License
- >> Determine compatibility
- >> Datasheet
- >> Change Log
- >> Product Webpage
- >> Answer record



# IP Core Information

Data sheet provided for each core (right-click on core in IP catalog to access)

- > The size of each core is available in the data sheet
- > For example, the axi\_timer\_v2\_00\_a data sheet contains the following table:

*Table 2-2: Performance and Resource Utilization: Artix-7 FPGA (XC7A355TDIE) and Zynq-7000 Devices*

Parameter Values		Device Resources		
Width of Timer/Counter	Enable Timer2	Slices	Flip-Flops	LUTs
8	False	49	53	96
16	False	61	69	120
32	False	84	101	181
8	True	50	74	123
16	True	74	106	161
32	True	97	170	256

# Outline

- > IP Catalog
- > ***IP directory***
- > IP device files
- > GP Interfaces
- > Adding IP to extend PS into PL
- > Bitstream generation
- > Summary

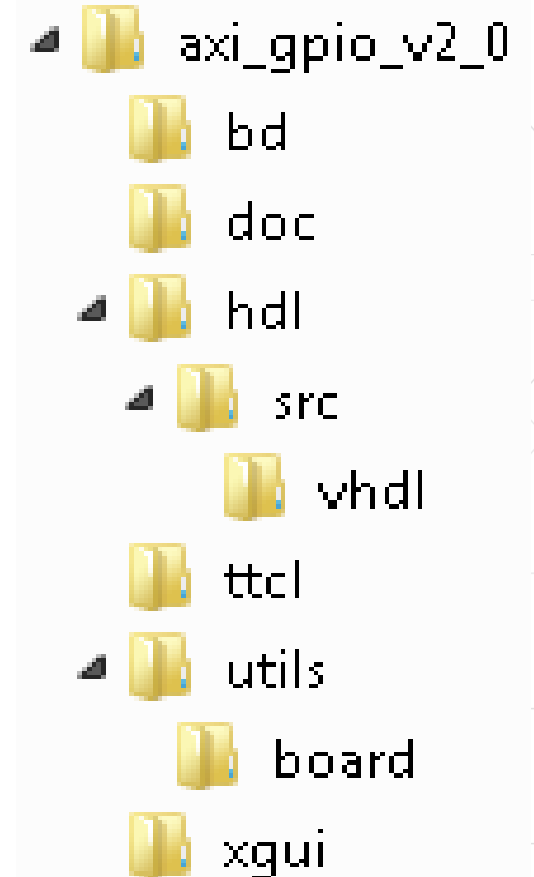


# Peripheral Storage

User peripherals can be located in the project directory or a peripheral repository

## > IP Core directory (located in the project directory)

- >> {component}.xml
- >> MyProcessorIPLib directory (user defined)
  - Repository Directory listed using  
**Project** → **Project Options** → **Device and Repository Search** tab
- >> %XILINX\_INSTALL%\Vivado\2017.X\data\ip



# Outline

- > IP Catalog
- > IP directory
- > ***IP device files***
- > GP Interfaces
- > Adding IP to extend PS into PL
- > Bitstream generation
- > Summary



# IP Core files

## > component.xml

- >> XML format
- >> Top level folder
- >> Provides ports description, parameters and options for IP
- >> Links to source files

## > xgui folder

- >> .tcl file for IPI GUI

```
<?xml version="1.0" encoding="UTF-8"?>
<spirit:component xmlns:xilinx="http://www.xilinx.com" xmlns:spirit="http://www.spiritconsortium.org/XMLSchema/SPIRIT/1685-2009" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <spirit:vendor>xilinx.com</spirit:vendor>
  <spirit:library>XUP</spirit:library>
  <spirit:name>led_ip</spirit:name>
  <spirit:version>1.0</spirit:version>
  <spirit:busInterfaces>
    <spirit:busInterface>
      <spirit:name>S_AXI</spirit:name>
      <spirit:busType spirit:vendor="xilinx.com" spirit:library="interface" spirit:name="aximm" spirit:version="1.0"/>
      <spirit:abstractionType spirit:vendor="xilinx.com" spirit:library="interface" spirit:name="aximm_rtl" spirit:version="1.0"/>
      <spirit:slave>
        <spirit:memoryMapRef spirit:memoryMapRef="S_AXI"/>
      </spirit:slave>
      <spirit:portMaps>
        <spirit:portMap>
          <spirit:logicalPort>
            <spirit:name>AWADDR</spirit:name>
          </spirit:logicalPort>
          <spirit:physicalPort>
            <spirit:name>s_axi_awaddr</spirit:name>
          </spirit:physicalPort>
        </spirit:portMap>
      </spirit:portMaps>
    </spirit:busInterface>
  </spirit:busInterfaces>

```

# Outline

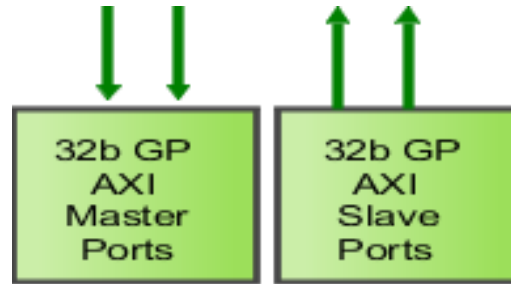
- > IP Catalog
- > IP directory
- > IP device files
- > ***GP Interfaces***
- > Adding IP to extend PS into PL
- > Bitstream generation
- > Summary





# GP Ports

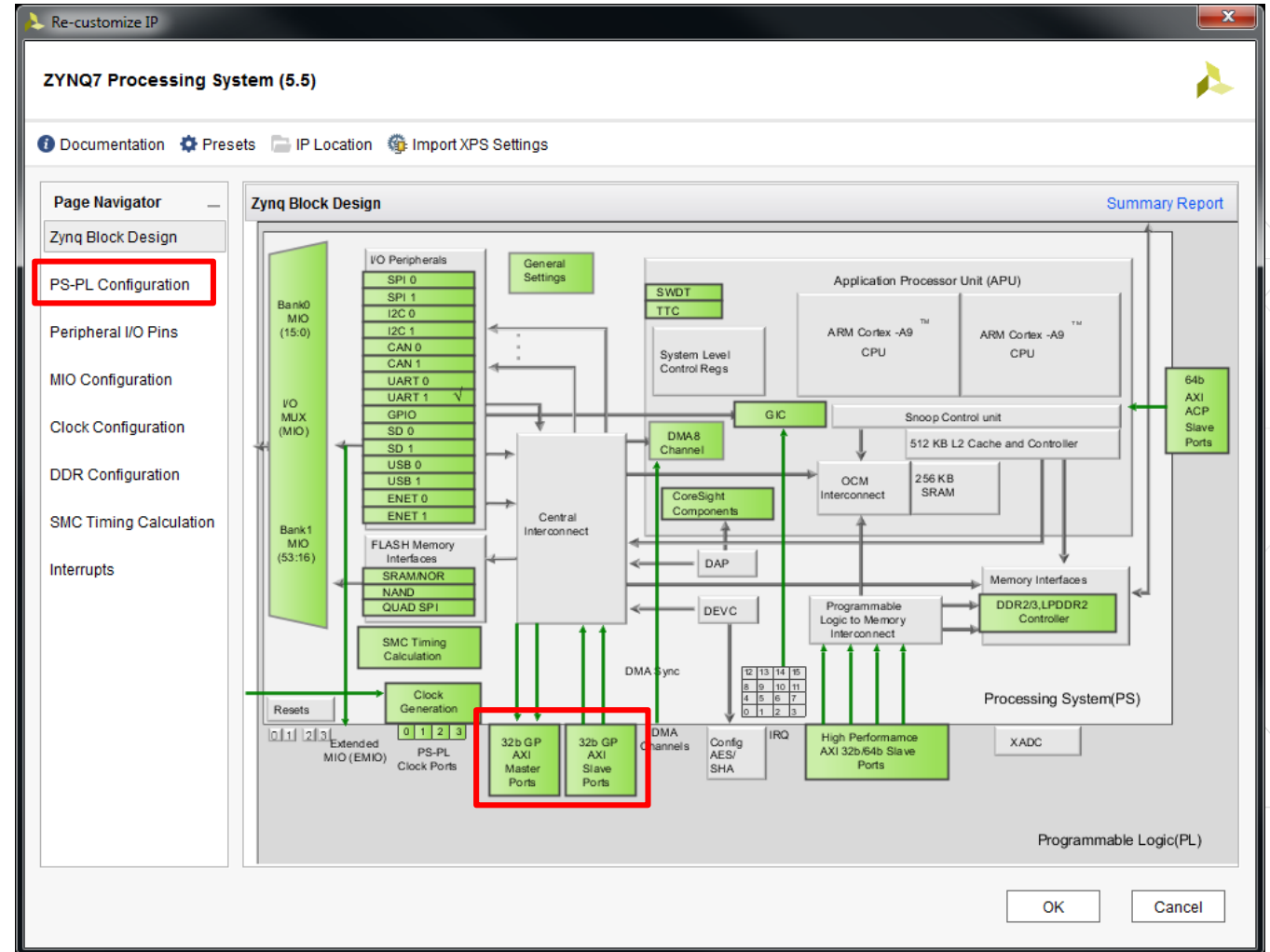
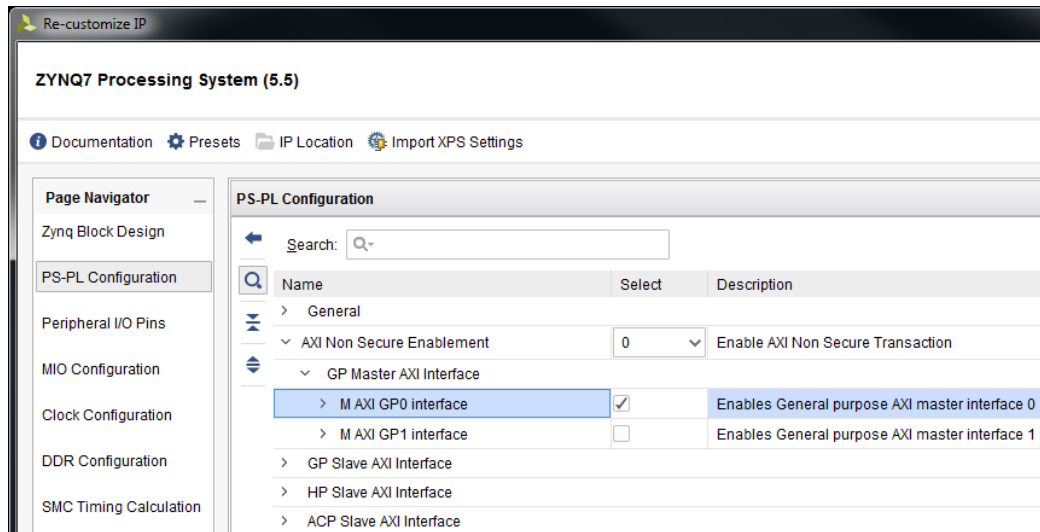
- > By default, GP Slave and Master ports are disabled



- > Enable GP Master and/or Slave ports depending on whether a slave or a master peripheral is going to be added in PL
- > axi\_interconnect block is required to connect IP to a port with different protocols
  - >> Automatically convert Protocols
  - >> Can be automatically added when using Block Automation in IPI

# Configuring GP Ports

- > Click on the menu or green GP Blocks to configure



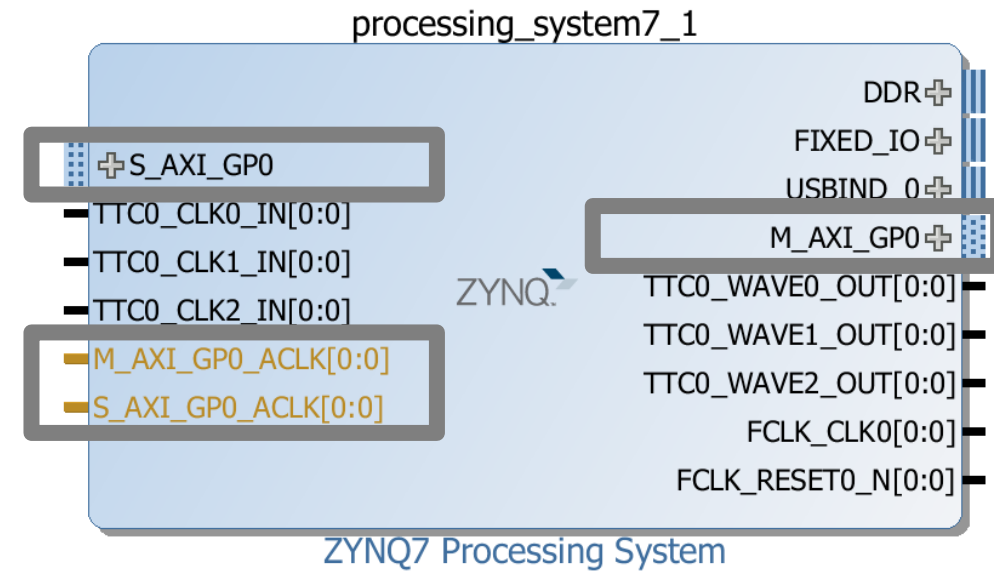
# Outline

- > IP Catalog
- > IP directory
- > IP device files
- > GP Interfaces
- > ***Adding IP to extend PS into PL***
- > Bitstream generation
- > Summary



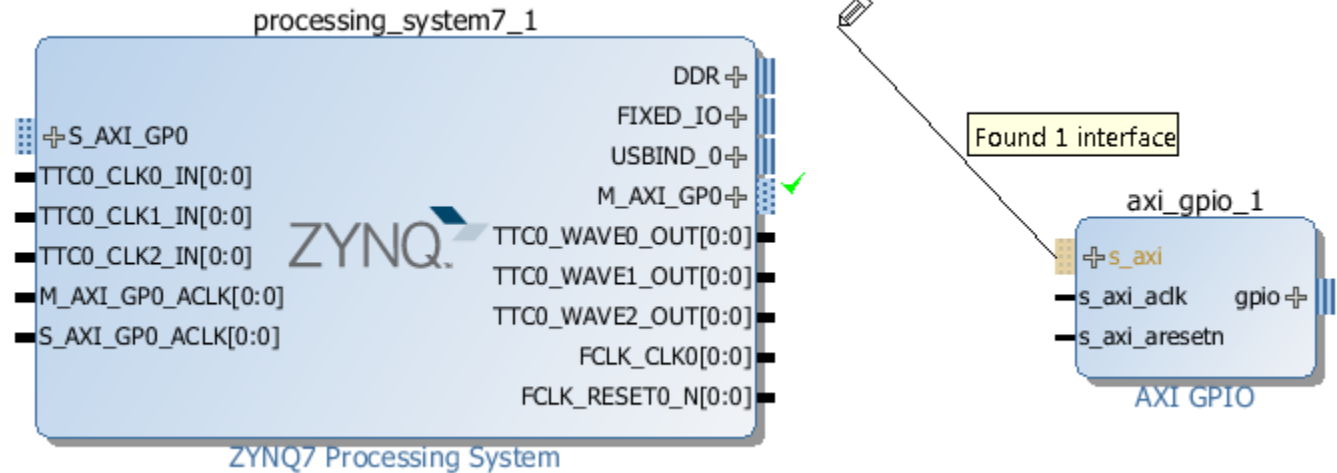
# Add IP in the PL

- > **Configure GP ports from PS Customization GUI**
- > **PS-PL Configuration**
  - E.g. Enable M\_AXI\_GP0/1 or S\_AXI\_GP0/1
- > **Ports will then be available on Zynq Block Diagram**
- > **Connect the added IP to the appropriate port**
- > **Assign address to the added IP, if unmapped**
- > **Configure the IP, if necessary**
- > **Make external connections, if needed**
  - Add external ports/interfaces if the added IP is interacting with external devices



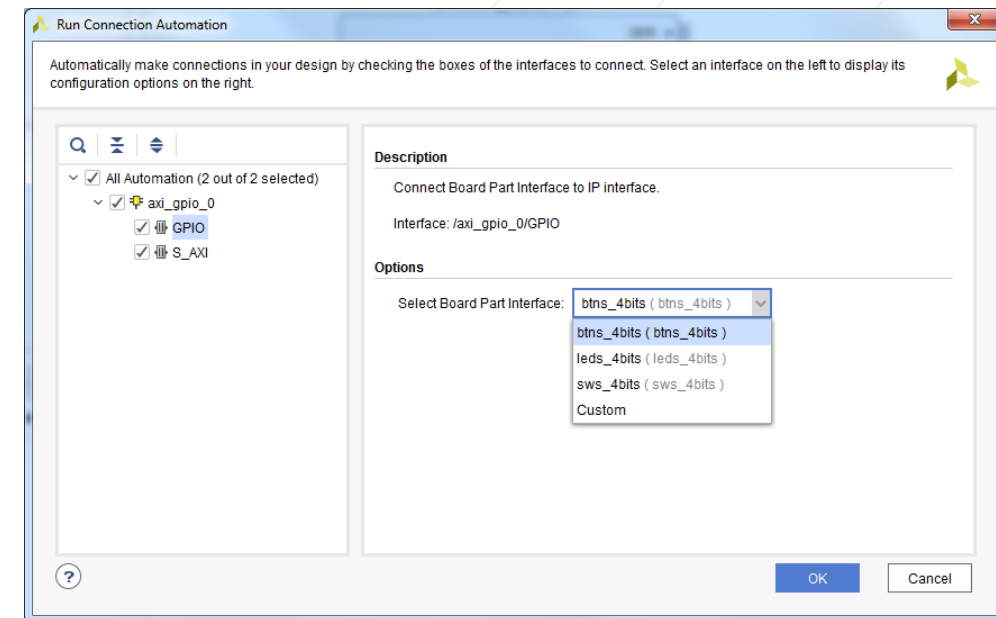
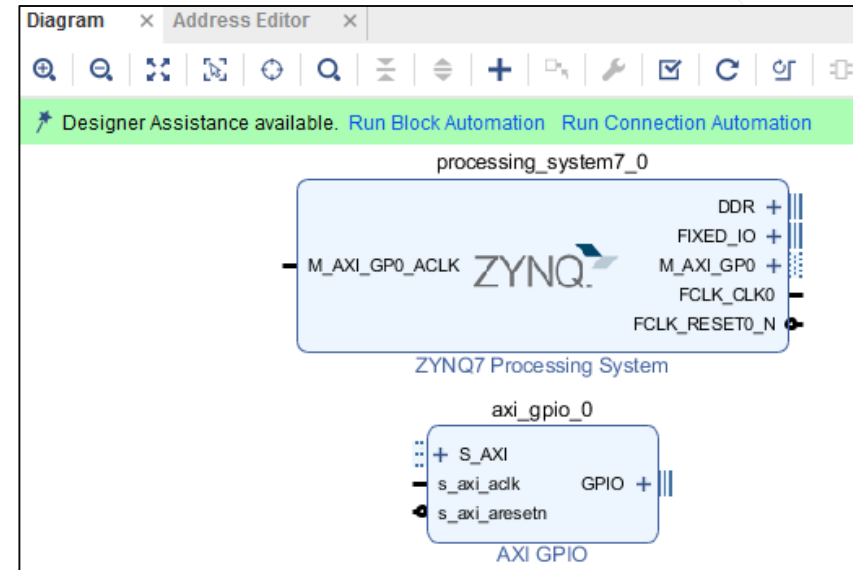
# Connecting IP

- > Add IP from the IP Catalog
- > Click and drag to find connections
- > Valid connections Highlighted
- > Designer Assistance, Connection automation
  - If Board Support available, IP can be connected to external pins
- > Or manually create and connect (external) ports



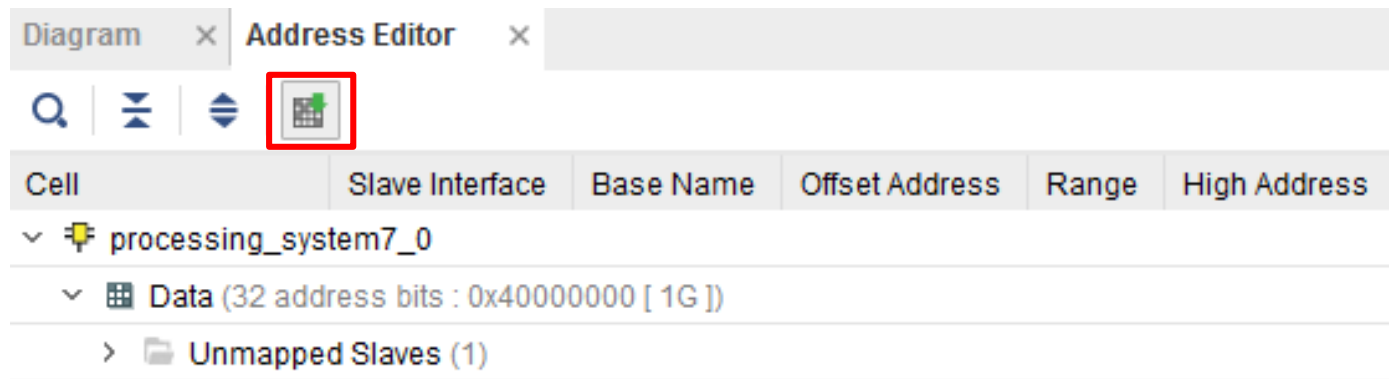
# Designer assistance; Block Automation, Connection Automation

- > **Block, Connection**
- > **Can automatically connect IP blocks**
- > **Automatically insert required blocks**
- > **E.g. Add BRAM; Automation will insert and connect BRAM controller and Reset logic**
- > **If Board Support is available, IP can automatically be connected to top level ports**



# Assign Addresses

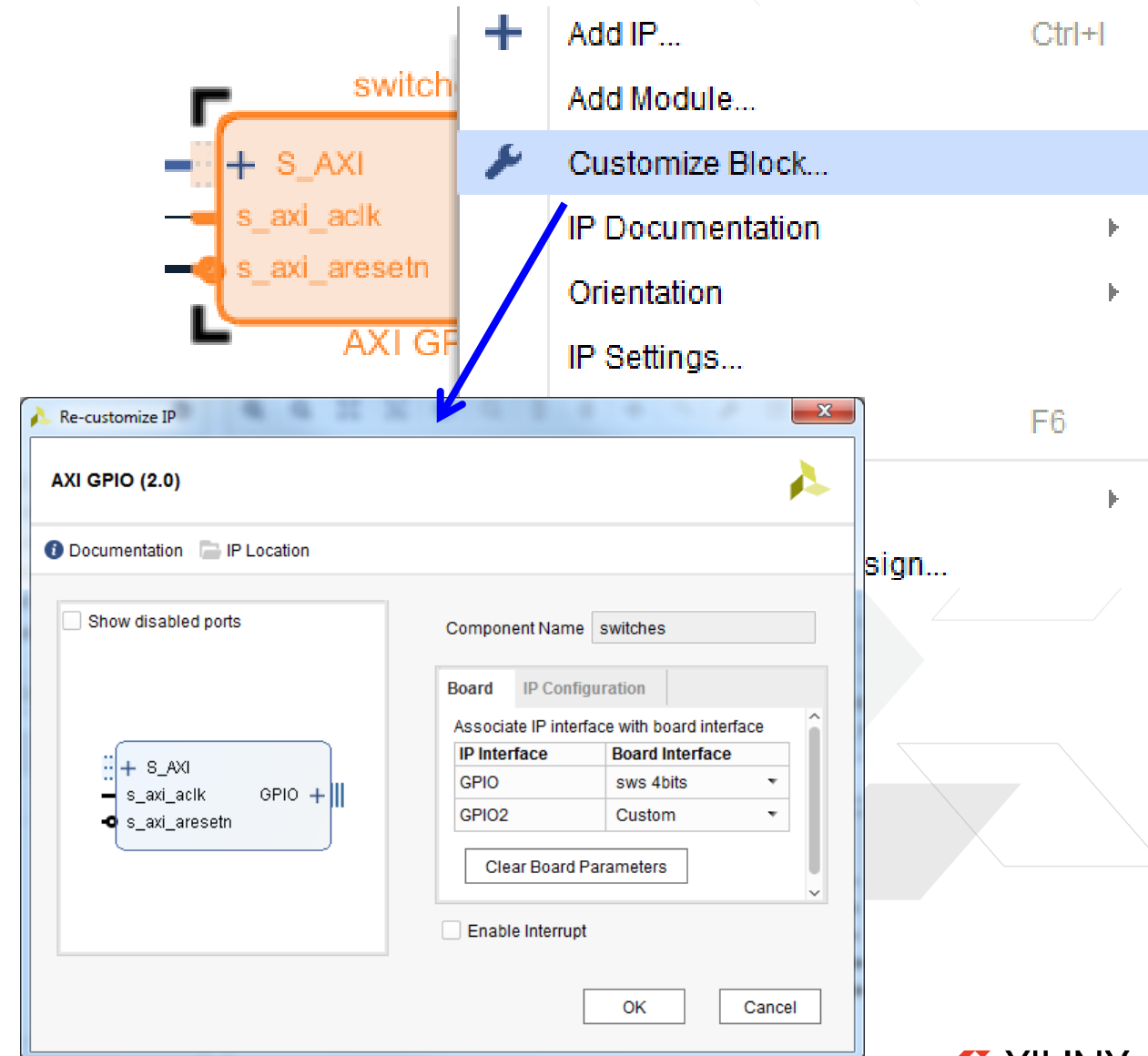
- > Peripherals in the Zynq™ AP SoC PS have fixed addresses and do not appear in the address map when an IP is added to the system
- > For PS peripherals Click on the Auto Assign Addresses button



- > The address will be generated and show the generated addresses of the added IP
- > The fixed addresses of the configured peripherals of the PS

# Parameterize IP Instances

- > **Double-click or right click the instance and select Customize Block to open the configurable parameters dialog box (refer to the datasheet if needed)**
- > **Default values are shown**
  - >> Customize the parameters that you want





# Extending the IP Catalog

- > **IP Packager**
  - >> Packages into IP Integrator Format
- > **Specify repository (local/global)**
- > **IP can then be used in IP Integrator**
- > **More on IP Packager later**

Project Summary x axi\_lite\_slave.v x Package IP - axi\_lite\_slave x

**IP Identification**

- ✓ IP Compatibility
- ✓ IP File Groups
- IP Customization Parameters
- ✓ IP Ports
- ✓ IP Interfaces
- ✓ IP Addressing and Memory
- ✓ IP GUI Customization Layout
- IP Licensing and Security
- ✓ Review and Package

IP Identification

0 errors 0 warnings 0 info messages

Fill in and modify the information fields below that will be used to identify your IP. Set the categories in which your IP will appear in the IP Catalog by modifying the Taxonomy.

**Identification**

Vendor : xilinx.com

Library : user

Name : axi\_lite\_slave

Version : 1.0

Display Name : axi\_lite\_slave\_v1\_0

Description : axi\_lite\_slave\_v1\_0

Vendor Display Name :

Company Url :

Categories : /Basic\_Elements

Root Directory : c:/xup/embedded/labs/led\_ip

Xml File Name : c:/xup/embedded/labs/led\_ip/component.xml

☐ Show advanced information

# Outline

- > IP Catalog
- > IP directory
- > IP device files
- > GP Interfaces
- > Adding IP to extend PS into PL
- > ***Bitstream generation***
- > Summary



# Bitstream Generation

- > **After defining the system hardware, the next step is to create hardware netlists if the system hardware has logic in PL**
- > **A HDL wrapper for the block diagram must be generated**
  - >> Additional logic can be added to the HDL, or the Processor system can be used as a sub block in a HDL design
- > **The design and block diagram must be open before synthesise and implementation can be carried out**
- > **If the system contains hardware in the PL, the bitstream must be generated**
- > **The PL (FPGA) must be programmed before application can be downloaded and executed**

# Outline

- > IP Catalog
- > IP directory
- > IP device files
- > GP Interfaces
- > Adding IP to extend PS into PL
- > Bitstream generation
- > ***Summary***



# Summary

- > **PS functionality can be extended by instantiating peripherals in PL**
- > **Adding IP in PL involves**
  - >> Enabling interface(s) in PS
  - >> Selecting IP from the IP Catalog and configuring IP for desired functionality
  - >> Connecting the (PL) IP to the PS using IP Integrator
  - >> Assigning address
  - >> Connecting IP ports to ports of other peripherals and/or to external pins
- > **HDL Wrapper is needed for IP Integrator Block**
- > **Bitstream must be generated when PL has any IP**
- > **The FPGA must be programmed with the generated hardware bitstream before an application can be run**