Practice 10

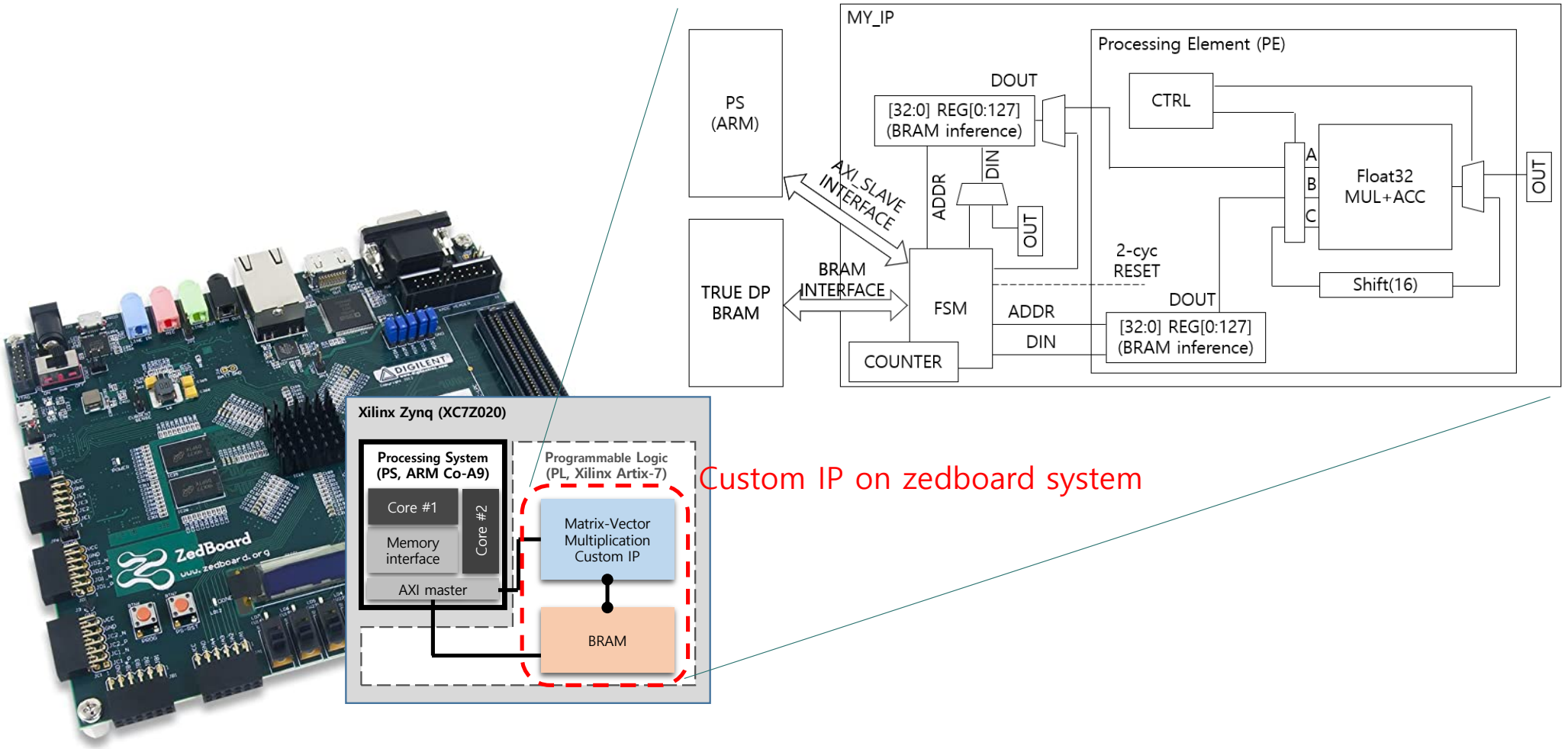
- Custom IP

Computing Memory Architecture Lab.

Overview

- **Custom IP Tutorial**
  - Processing System + BRAM + Connectivity + Custom IP (shifter)
    - Note) Term project = PS + BRAM + Connectivity + Custom IP (M*V multiplier)

Final Project Overview: Matrix Multiplication IP



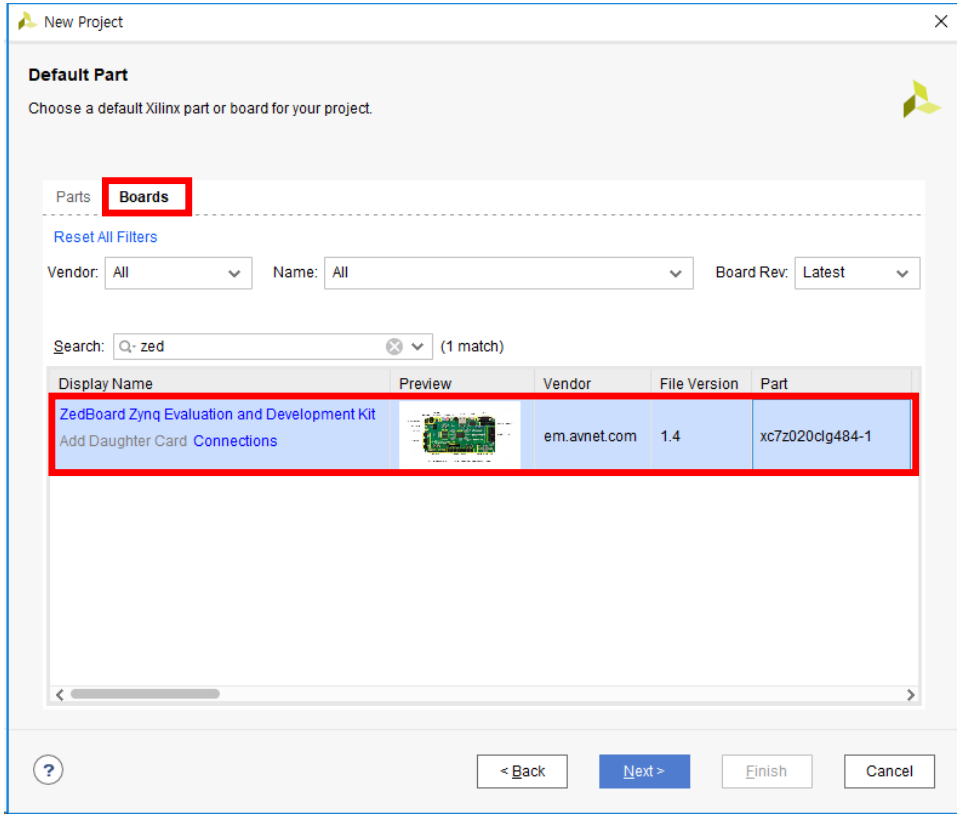
Custom IP Tutorial

Notations

- HOST$ XXX
  - Type XXX @ the terminal of your Ubuntu-PC
- BOARD$ YYY
  - Type YYY @ the terminal of ZedBoard
    - Which is equivalent to the after-terminal of below command
      - HOST$ minicom -D /dev/ttyACM0
- TCL_Console$ ZZZ
  - Type ZZZ @ tcl console of Vivado (details are in the following slides)

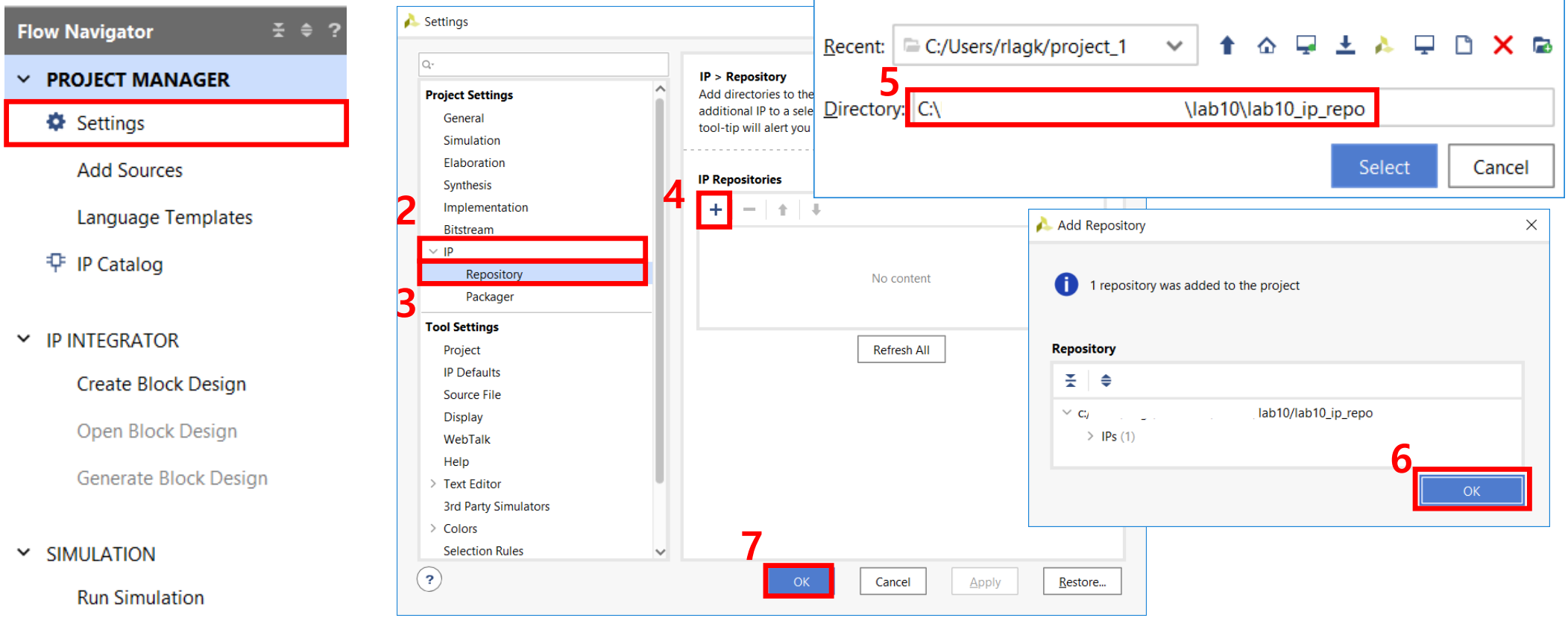
Vivado project creation

- Choose part or board
  - We are going to use ZedBoard



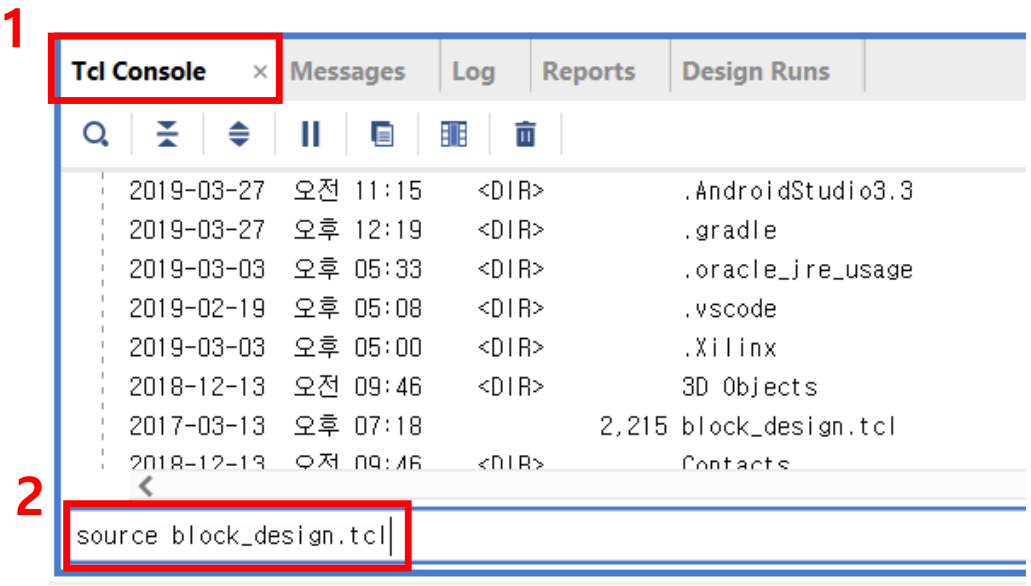
Example IP - Shifter

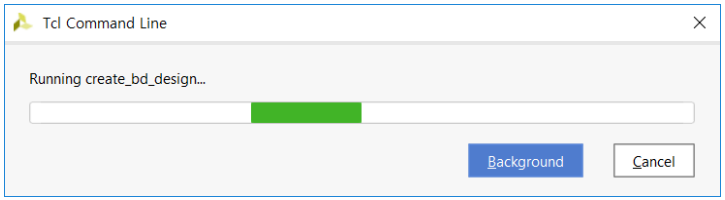
- Download an example IP repository
  - HOST$ tar -zxvf lab10_ip_repo.tar.gz



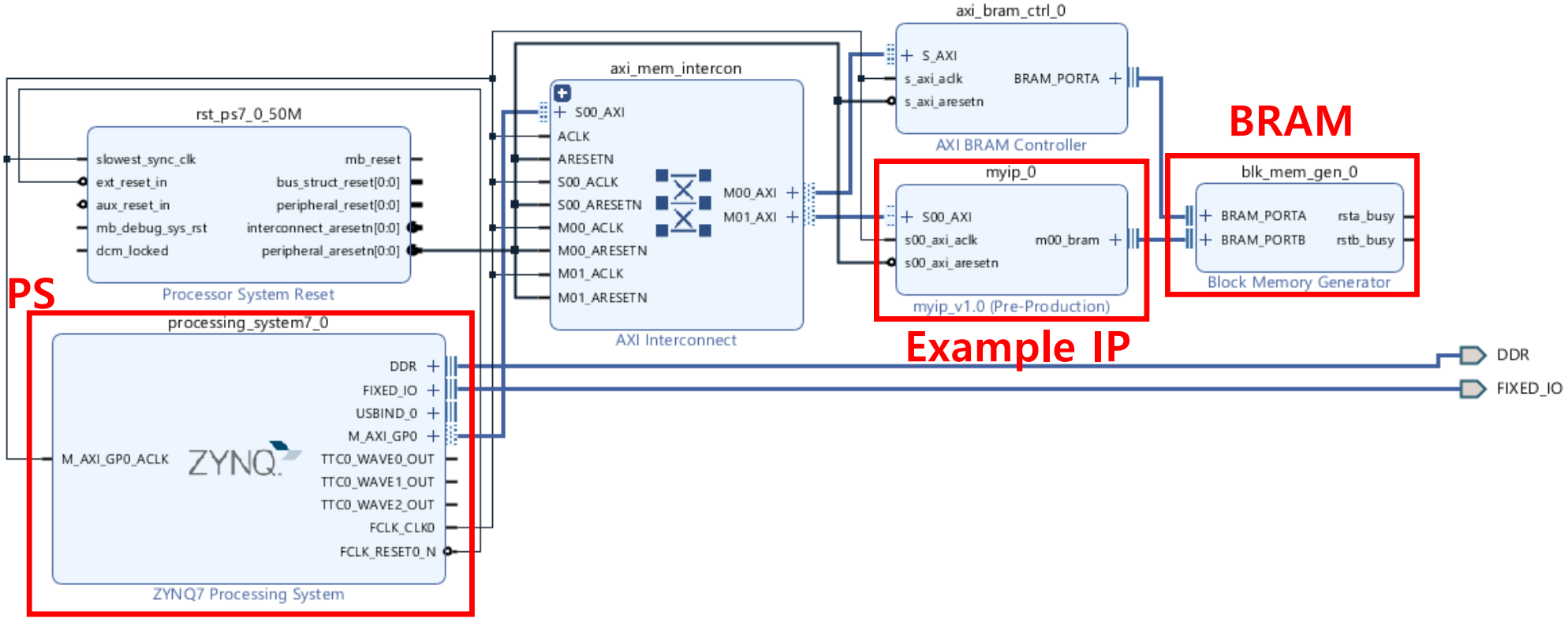
Example IP - Shifter

- Execute TCL scripts of the example design
  - TCL_Console$ cd ${DOWNLOAD_LAB10_DIR}
  - TCL_Console$ source block_design.tcl



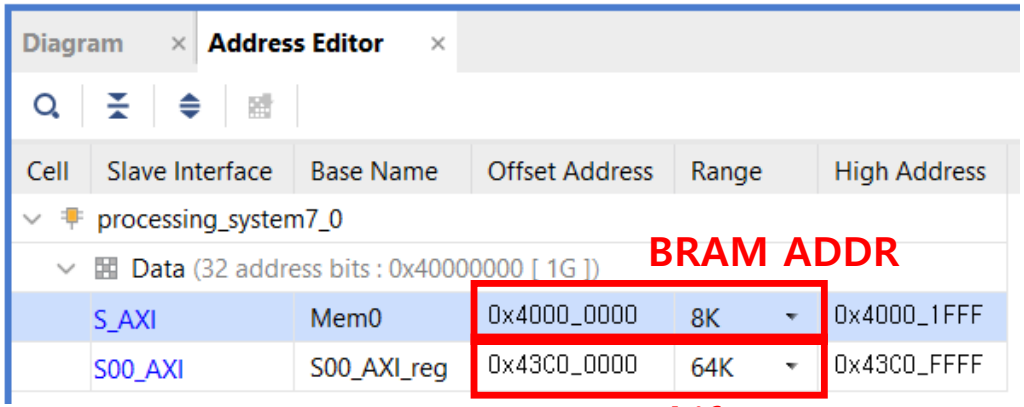


Example IP - Shifter



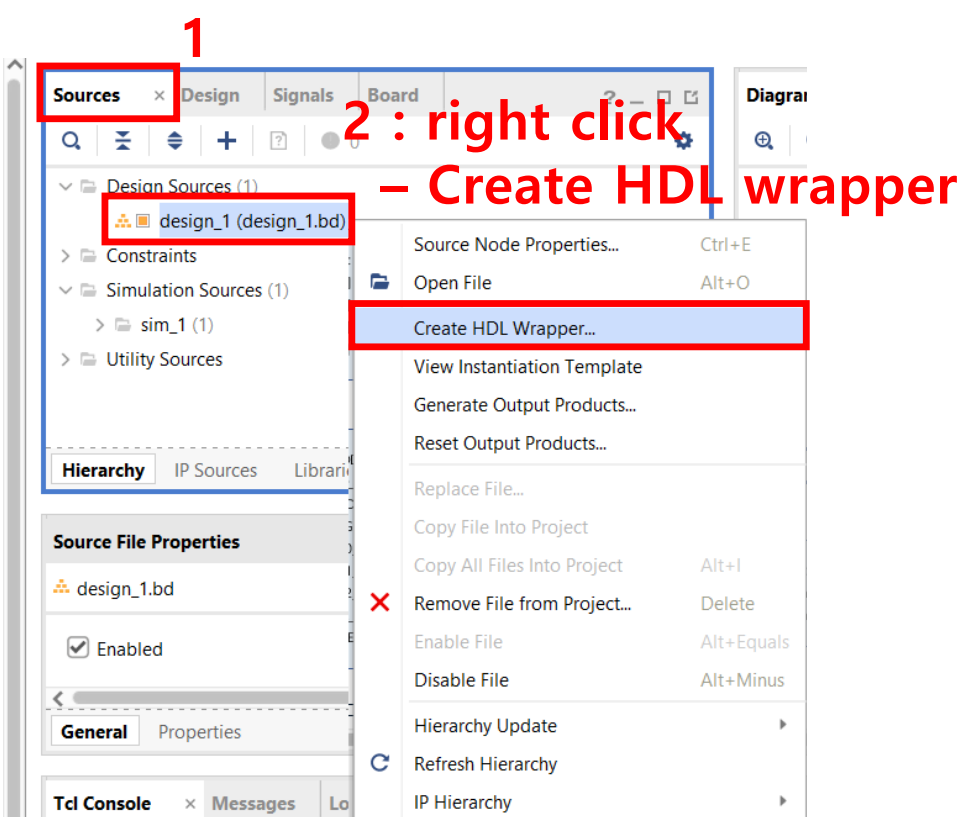
Example IP - Shifter

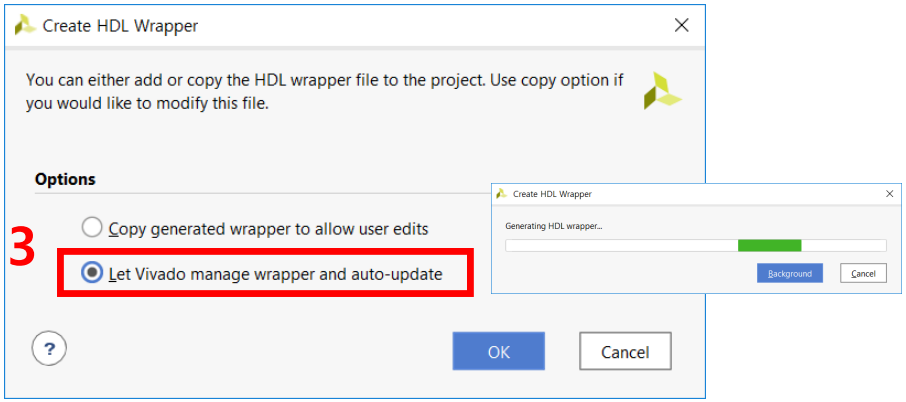
- BRAM is @ address 0x4000_0000 ~ 0x4000_1FFF
- Shifter is @ address 0x43C0_0000 ~ 0x43C0_FFFF

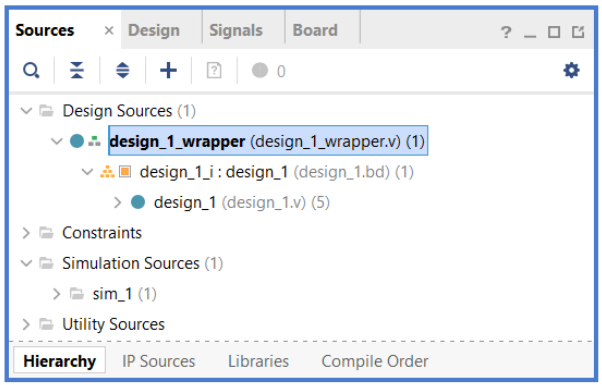


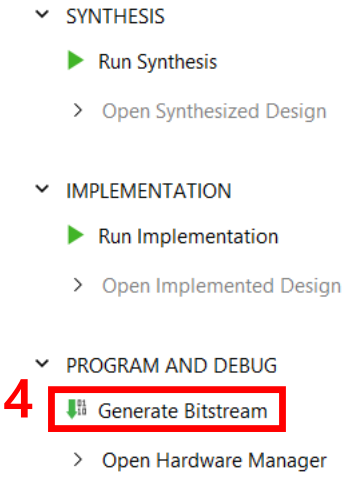

```
422     wire magic_code = (slv_reg0 == 32'h5555);
423
424     always @( posedge S_AXI_ACLK )
425     begin
426       if ( S_AXI_ARESETN == 1'b0 )
427         bram_state <= BRAM_IDLE;
428       else
429         case (bram_state)
430           BRAM_IDLE: bram_state <= (magic_code)? BRAM_READ : BRAM_IDLE;
431           BRAM_READ: bram_state <= BRAM_WAIT;
432           BRAM_WAIT: bram_state <= BRAM_WRITE;
433           BRAM_WRITE: bram_state <= (run_complete)? BRAM_IDLE: BRAM_READ;
434           default  : bram_state <= BRAM_IDLE;
435         endcase
436     end
```

Example IP - Shifter









[ZEDBoard Hardware User's Guide, 2014]

Example IP - Shifter

- Preparing SD card (1)
  - Change **zynq.bit** on host
  - Insert sdcard to board
- Set SD boot mode (2)
  - BD.JP7 ~ BD.JP11
    - **5'b01100**
- Insert LAN Cable (3)
- Insert power cable (4)
  - Yet stay power OFF
- Connect USB cable (5)
  - BD.J14
  - HOST$ dmesg

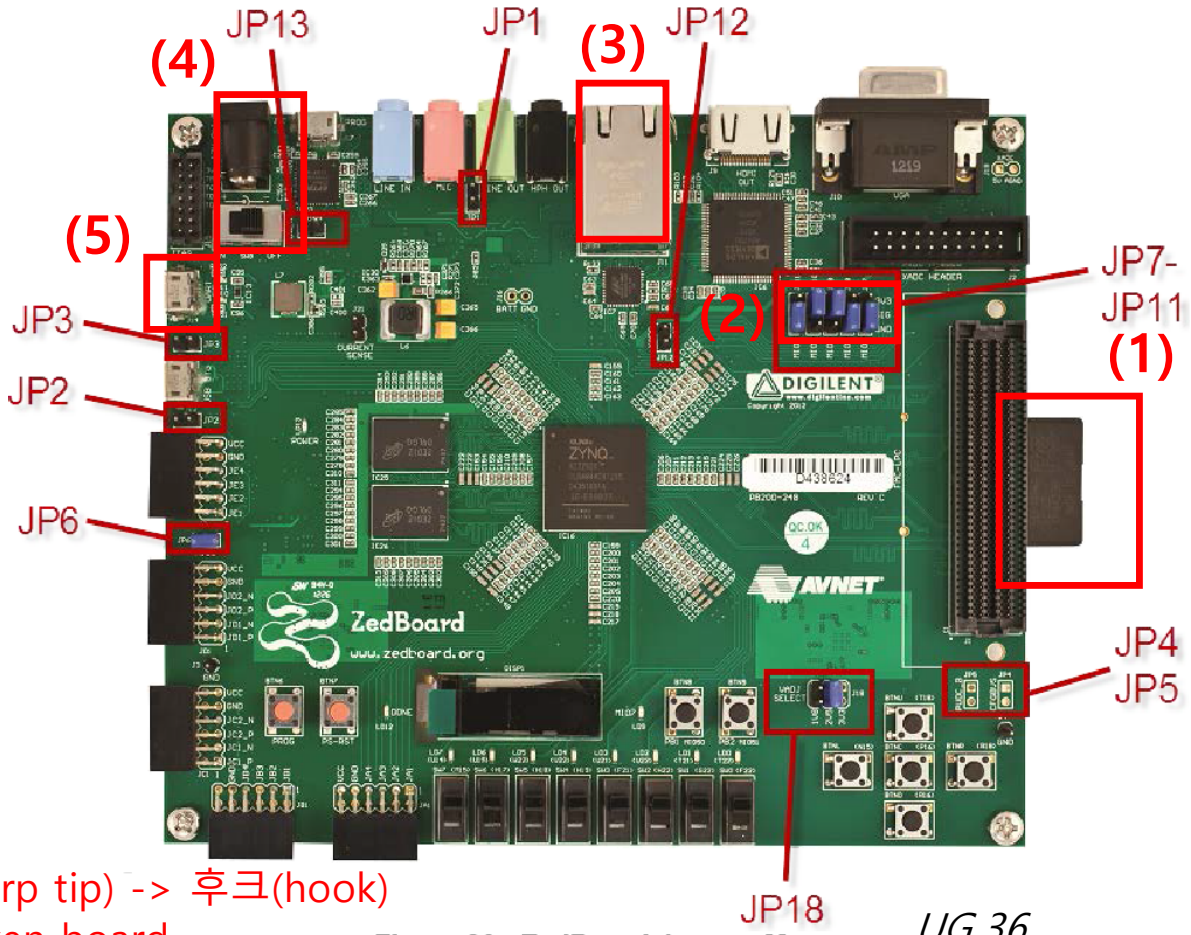


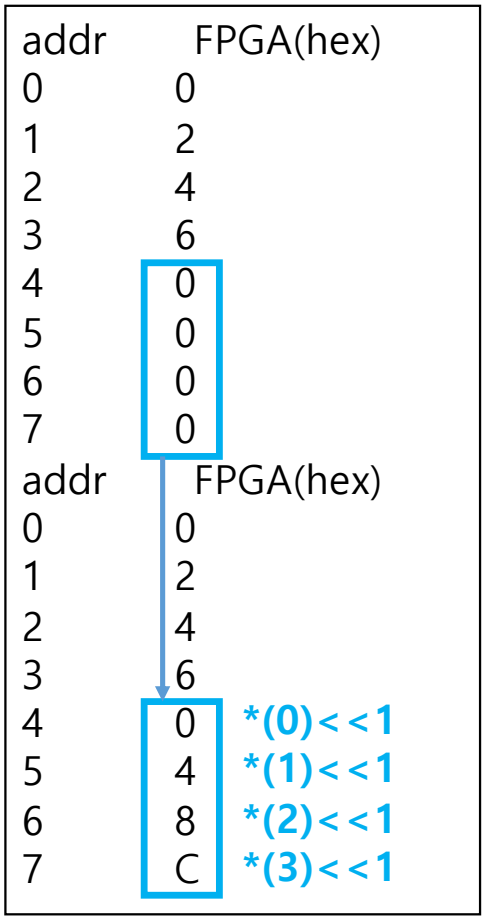
Figure 20 - ZedBoard Jumper Map

UG.36

- Micro 5pin cable
  If 빡빡(stiff) -> 칼팁(sharp tip) -> 후크(hook)
  No further labs for broken board

Example IP - Shifter

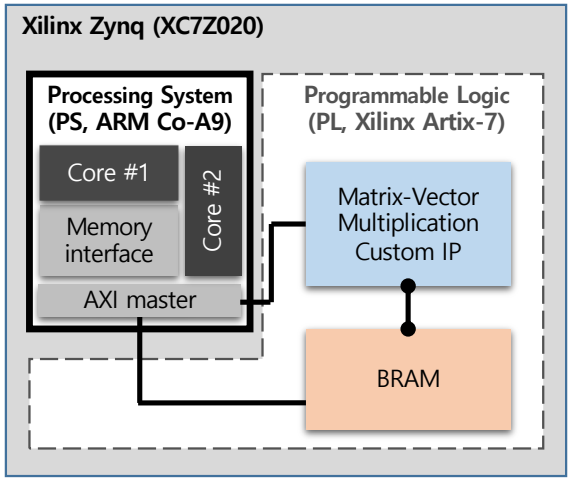
- Board Power ON
- Open terminal @ HOST
  - Login ID/PW: zed/zedzed
  - HOST$ minicom -D /dev/ttyACM0
- Run example program
  - BOARD$ cd ${LAB10_DIR}
  - BOARD$ make
  - (before un-plugging power) BOARD$ sudo poweroff
    - If you do not execute poweroff on terminal and eject the sdcard, your sdcard will not work permanantly. (penalty ☺)



Main Practice

Practice

- **Follow tutorial & understand the functionality**
  - Run a given example IP in your FPGA linux.
    - Run and understand ip customizing of sample project on report.
  - Our practice
    - Processing System + BRAM + Connectivity + Custom IP (shifter)
  - Our project
    - Note) Term project = PS + BRAM + Connectivity + Custom IP (M*V multiplier)



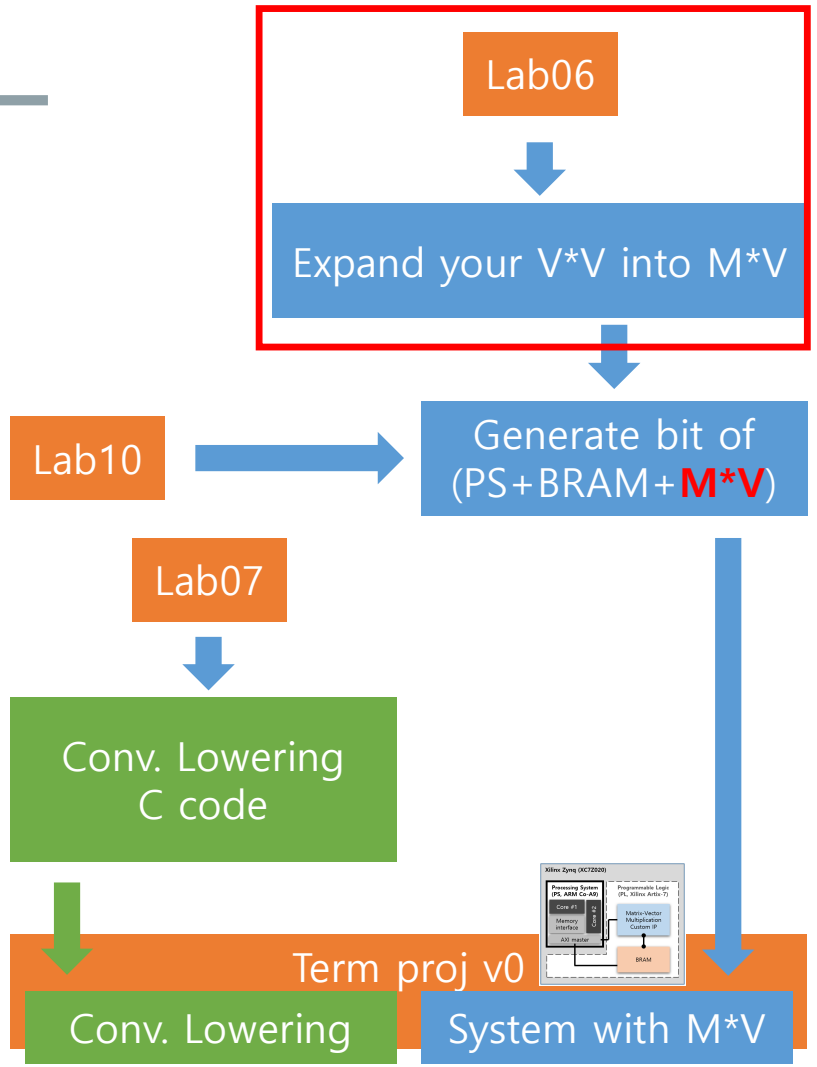
Homework

- Follow tutorial & Understand functionality of MyIP
- No Report
- Nothing to Submit for Lab 10
  - There will be no submit page either.

Term Project v0

Term Project v0 중간제출

- Project Requirement
  - Expanding V*V into M*V by making PE Array
    - M*V by making PE Array
      - No required specs
      - Edit your V*V code(lab06) on your own
  - Demo
    - Waveform

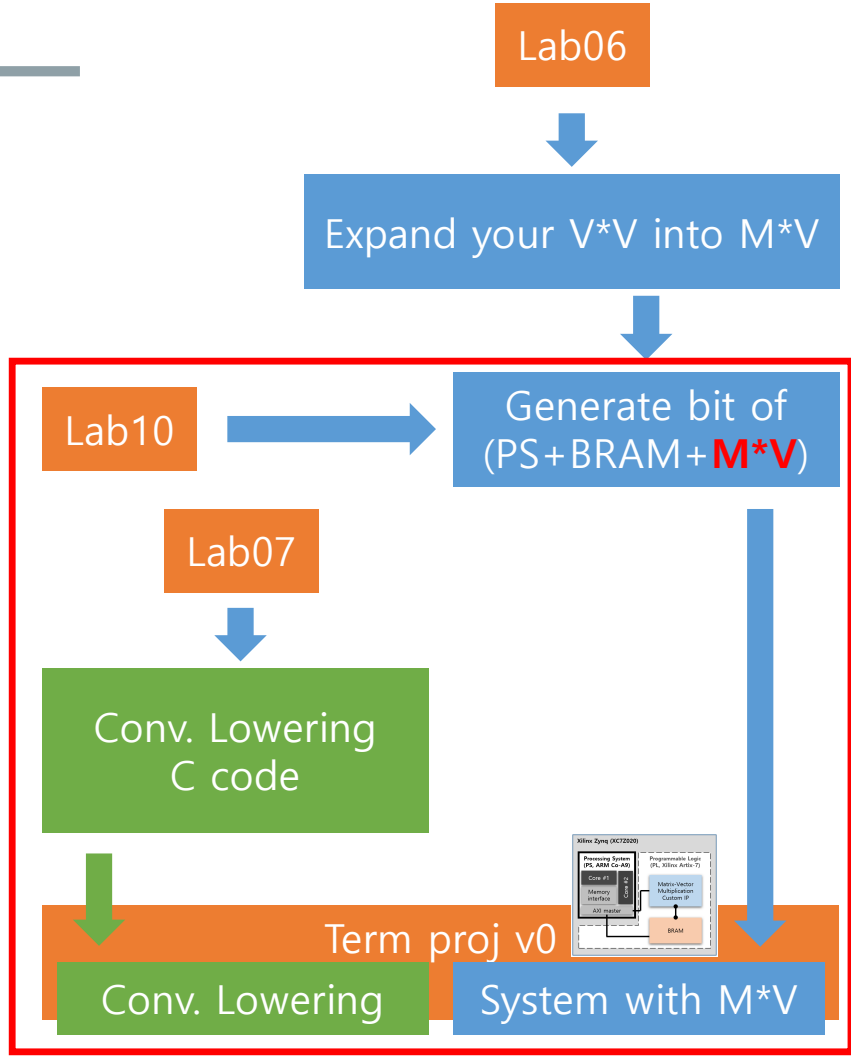


Term Project v0 중간제출

- Requirements
  - Result
    - Attach your project folder with all your verilog codes (e.g., M*V, test bench)
    - Attach your M*V waveform(simulation result) with [student_number, name]
      - The correct waveform should be shown to confirm the operation of your code.
      - Refer to Practice3 about screenshot.
  - Report
    - Explain operation of M*V with waveform that you implemented
    - In your own words
    - Either in Korean or in English
    - # of pages does not matter
    - **PDF only!!**
  - **Result + Report to one .zip file**
- Upload (.zip) file on ETL
  - Submit one (.zip) file
    - zip file name : [Term0-mid]name.zip (ex : [Term0-mid]홍길동.zip)
  - Due: 6/2(TUE) 23:59
    - **No Late Submission**

Term Project v0 최종제출

- Project Requirement
  - Convolution lowering + Custom IP system with M*V
    - Convolution lowering
      - Refer Lab07
    - Custom IP system with M*V
      - Refer Lab10 tutorial(Custom IP) & appendix(editing the custom IP)
  - Demo
    - Zedboard



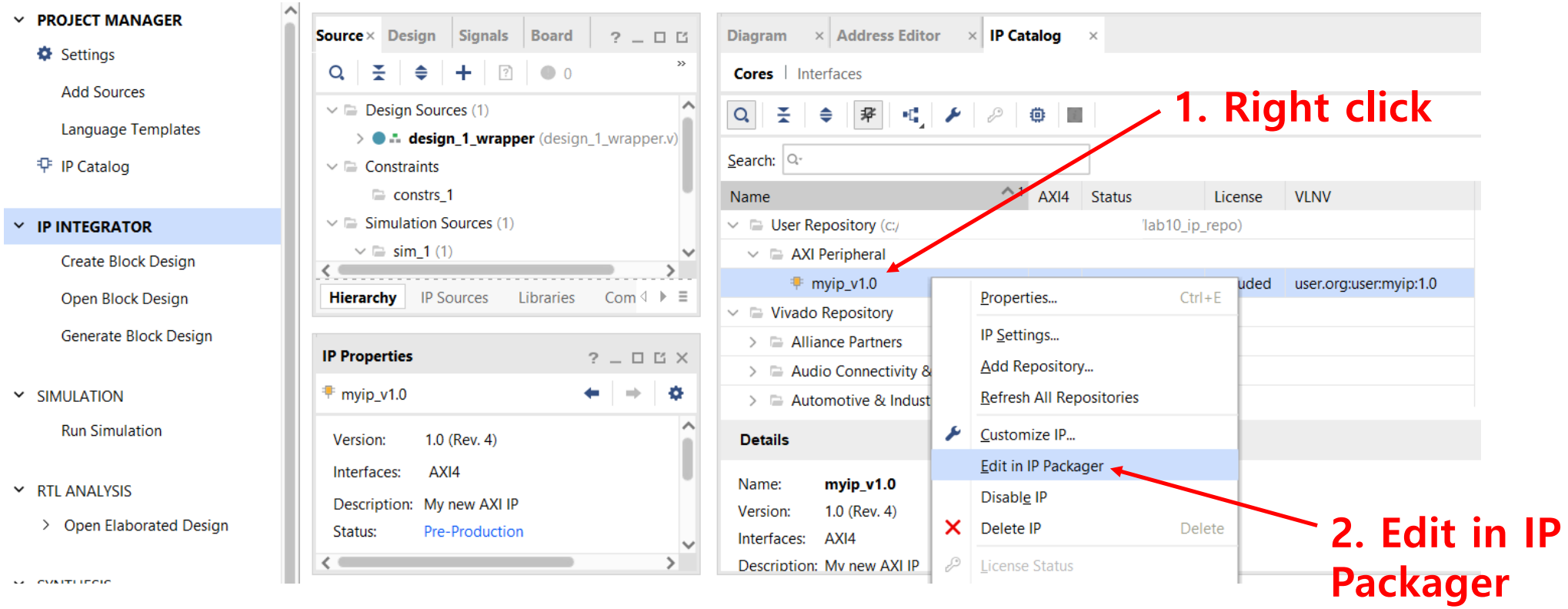
Term Project v0 최종제출

- Due: 6/16(TUE) 23:59
  - **Details will be noticed next week**

Appendix – Editing the Custom IP

Editing Custom IP

- (1) IP Catalog -> Edit in IP Packager -> IP project

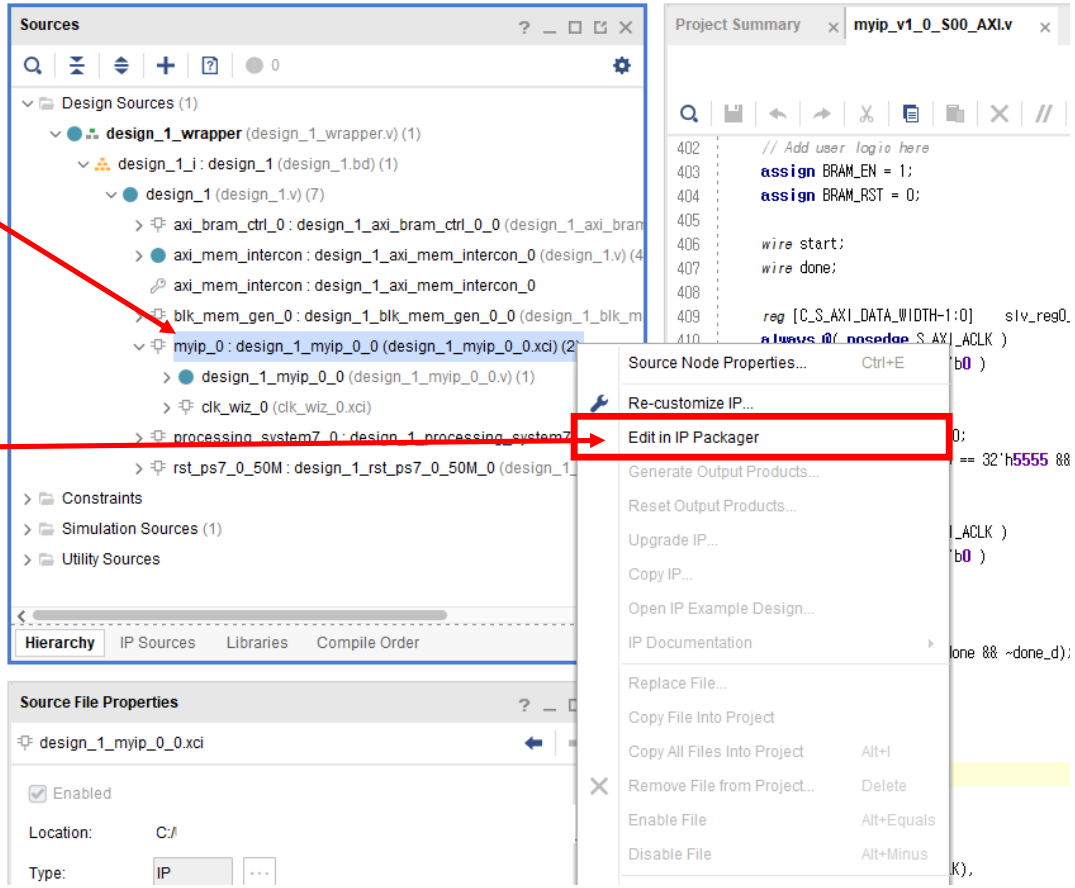


Editing Custom IP

- (2) On your source list -> Edit in IP Packager -> Packaging Custom IP on IP Project

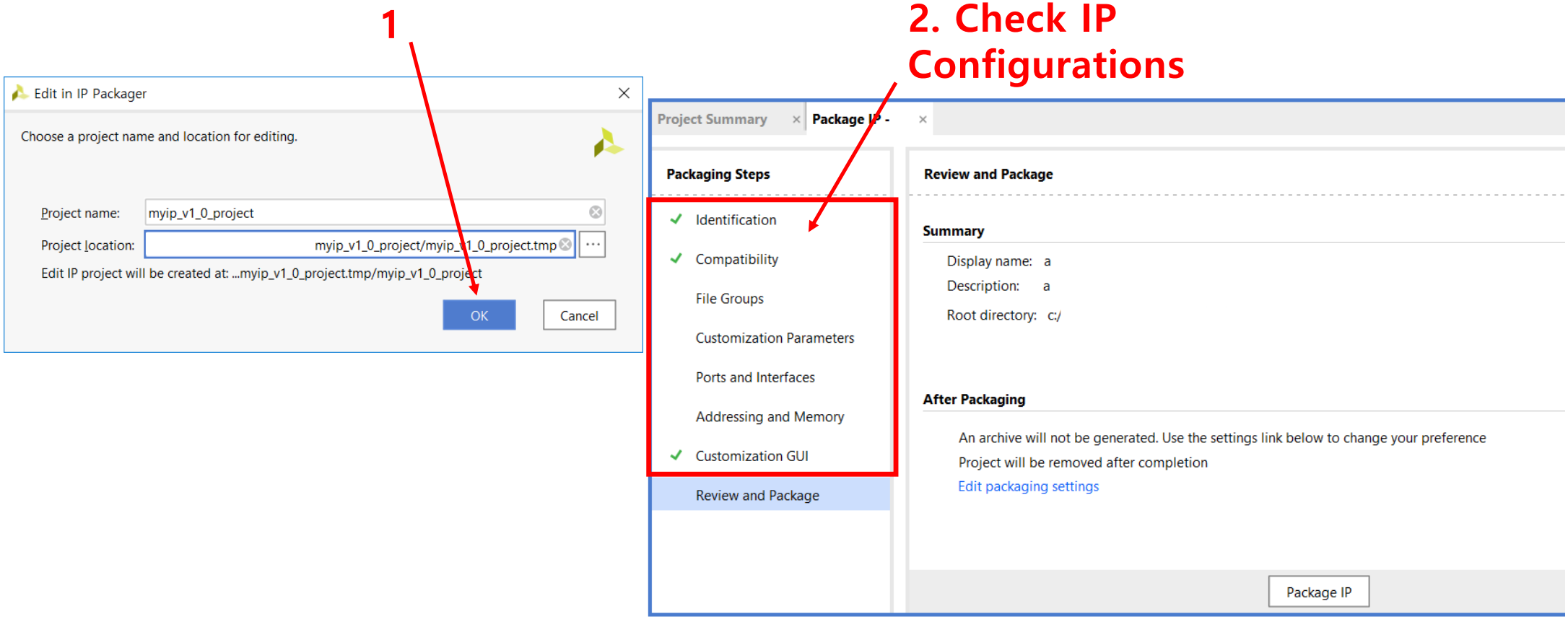
1. Right click on custom IP

2. Edit in IP Packager



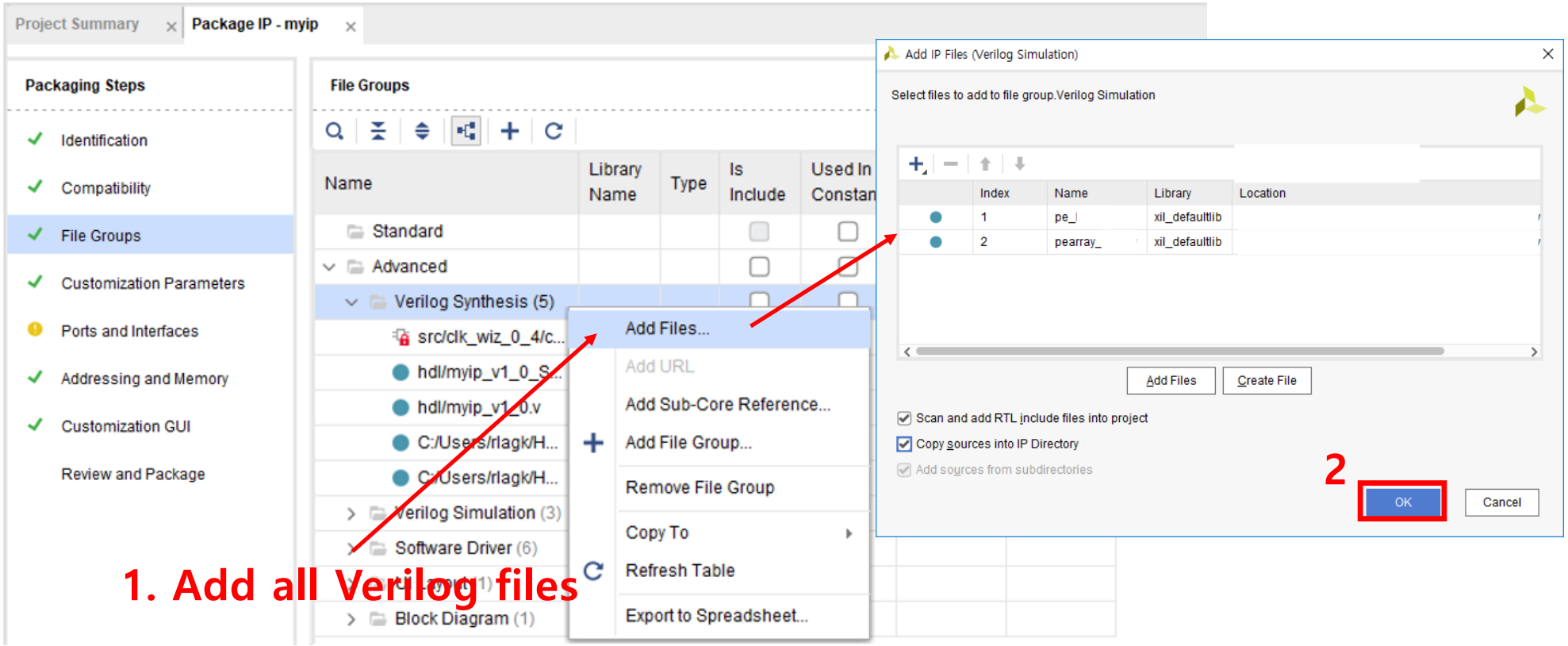
Packaging Custom IP

- Package IP -> Configurations



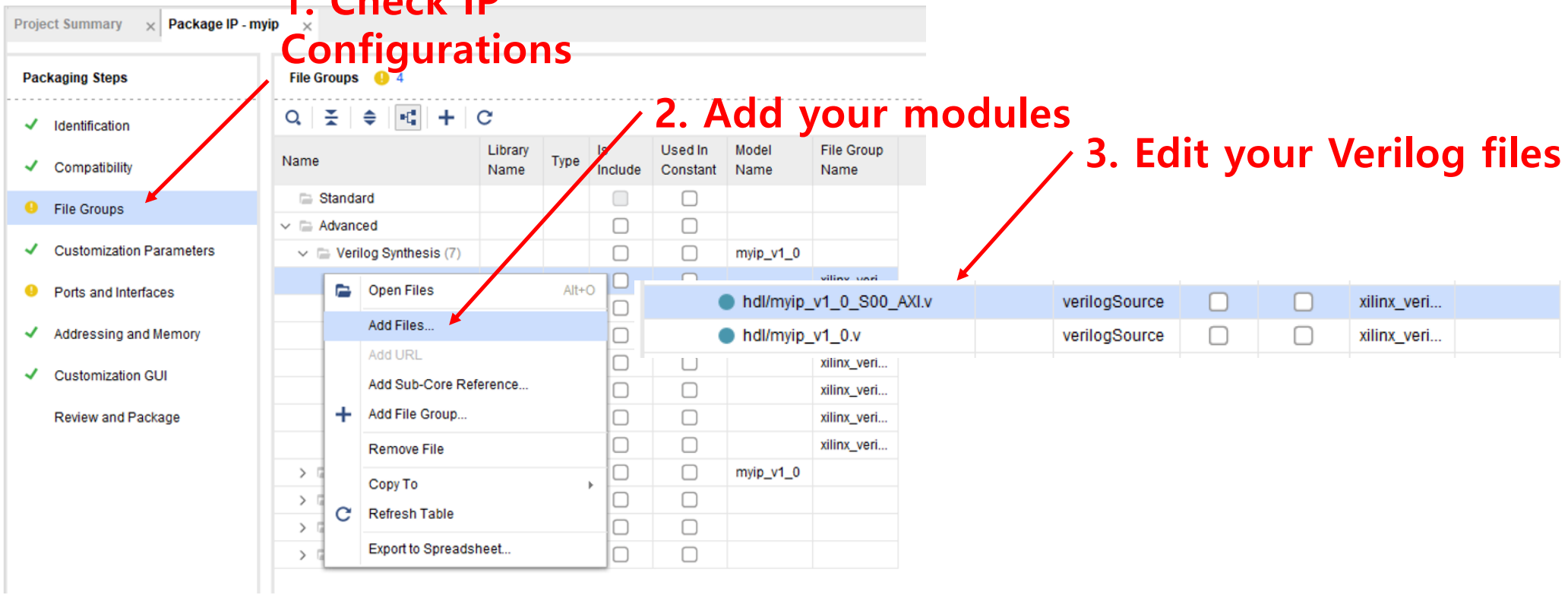
Checking IP Configurations

- Changing IP files
- File Groups -> Add all Verilog files needed for IP



Checking IP Configurations

- Changing IP files
- File Groups -> Add your modules -> Edit Verilog files for IP

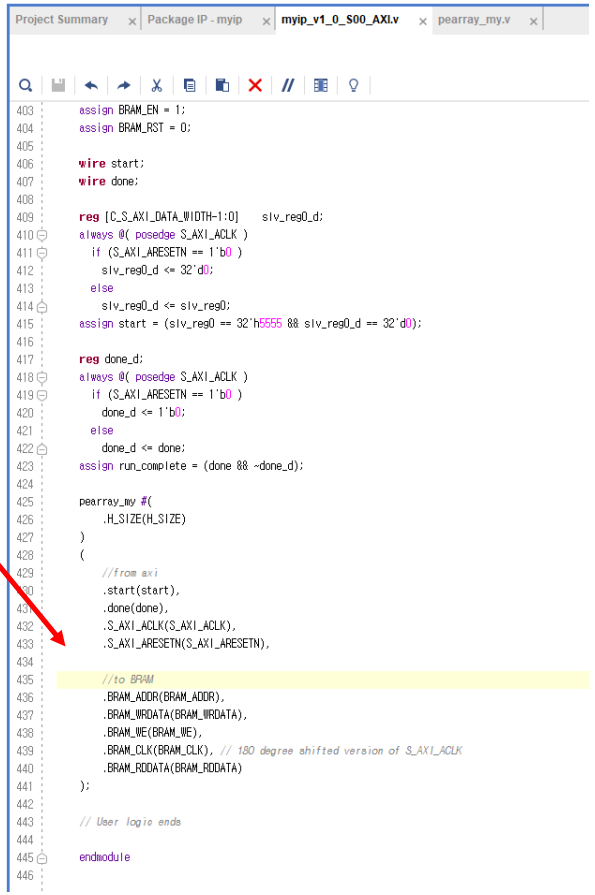


Checking IP Configurations

- Changing IP files
- File Groups -> Add & Edit Verilog files for IP

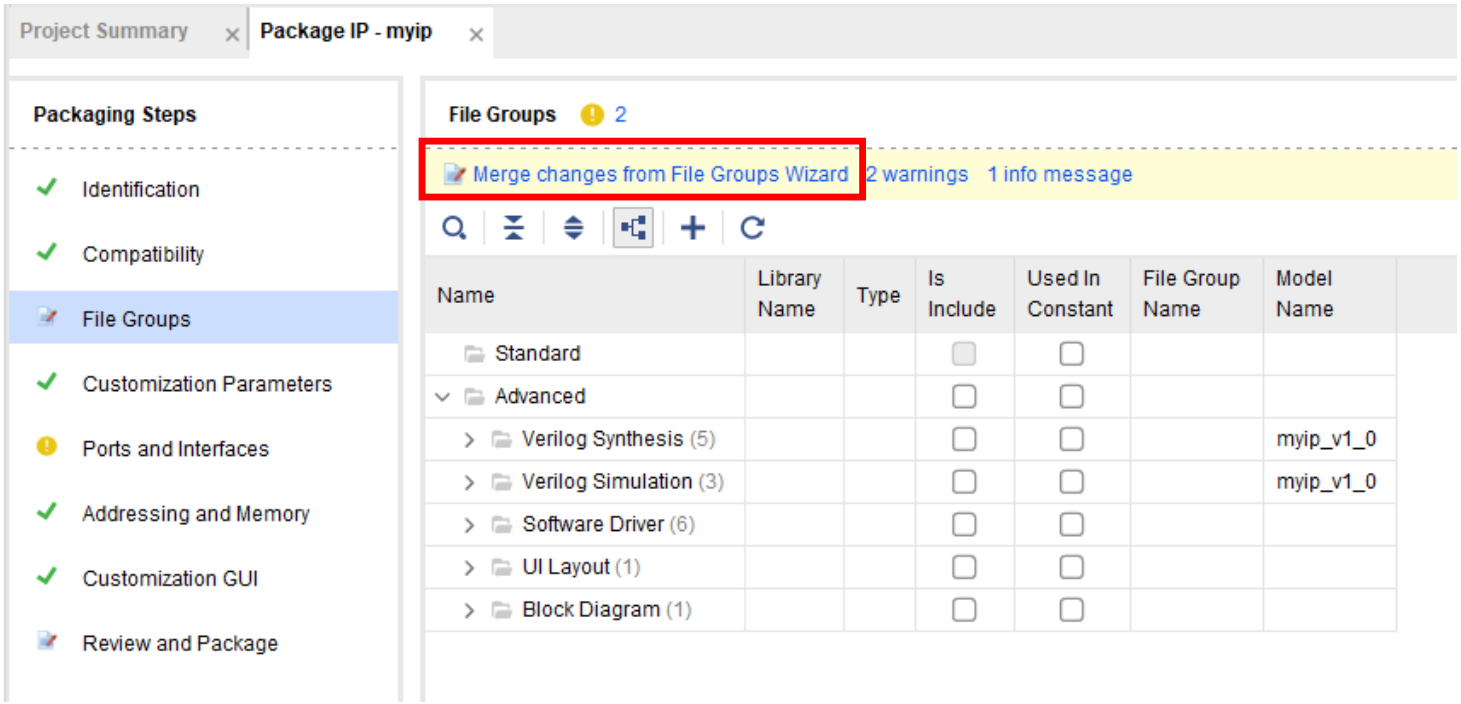
1. Add your modules
My ip, AXI parameter, port modules

2. Revise myip_v1_0_S00_AXI.v & myip_v1_0.v codes
- especially ports, parameters, modules you needed



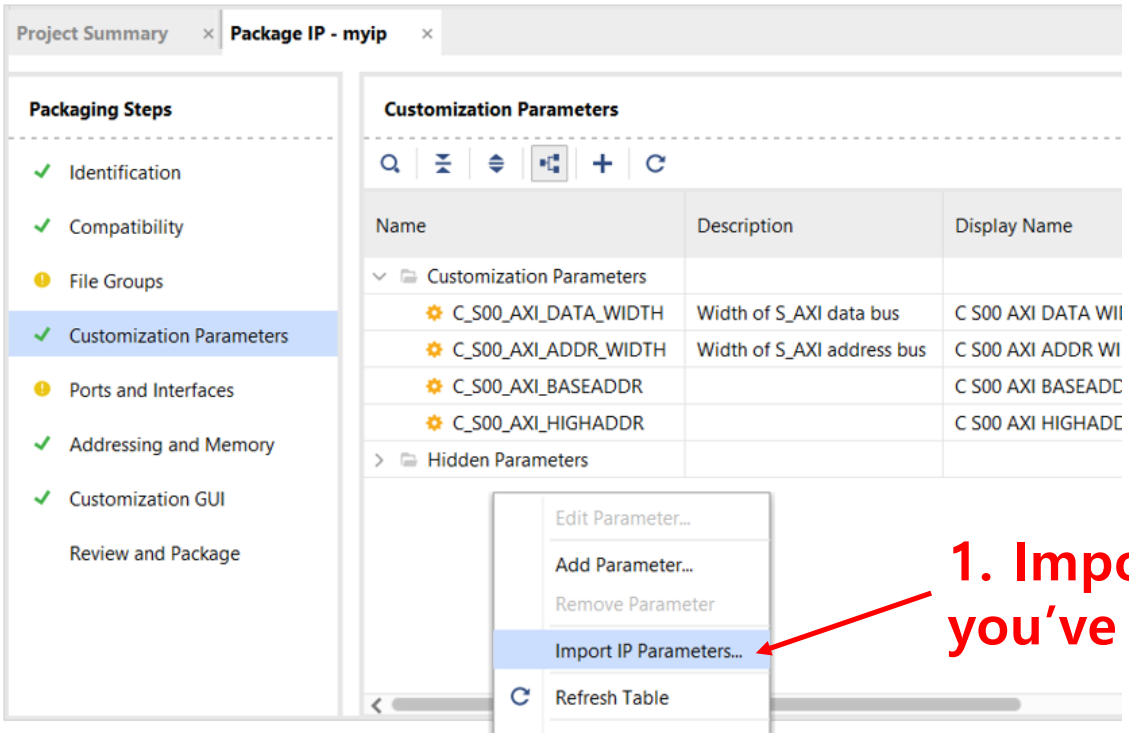
Checking IP Configurations

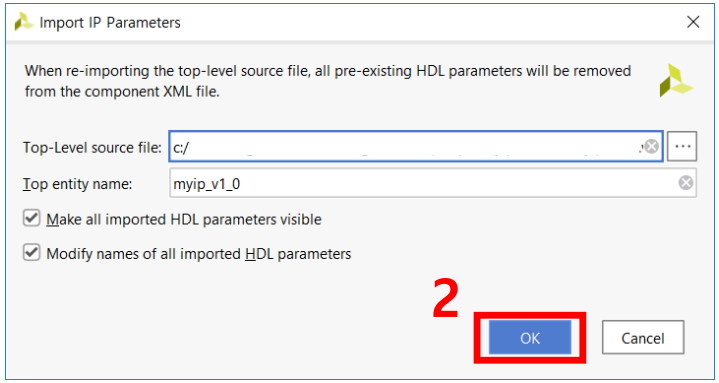
- Changing IP files
- File Groups -> Merge changes



Checking IP Configurations

- Parameter configuration
- Customization Parameters -> import IP parameters

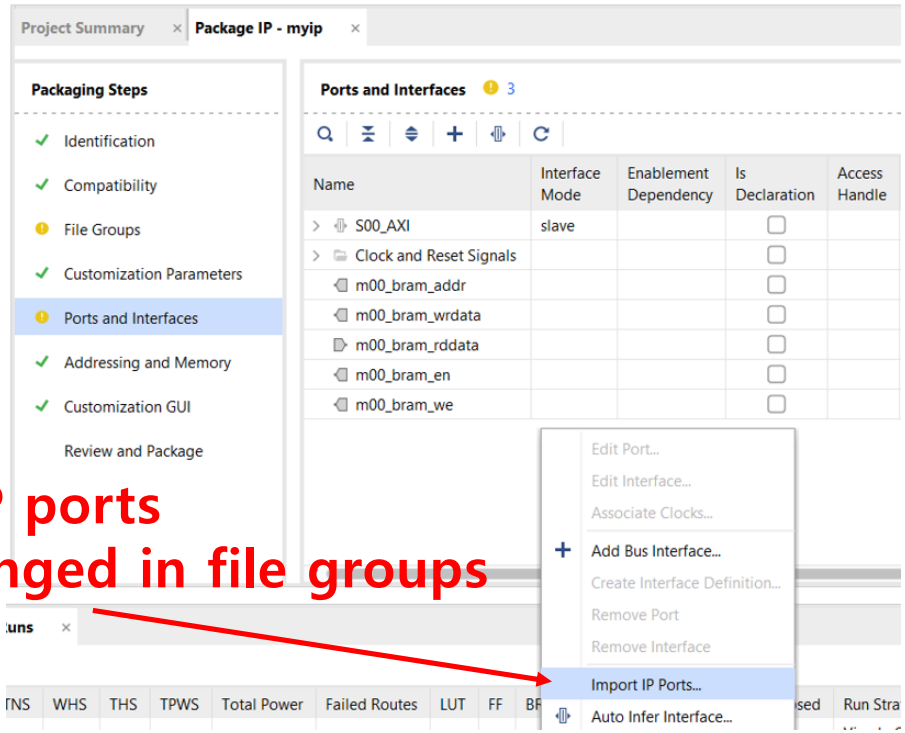




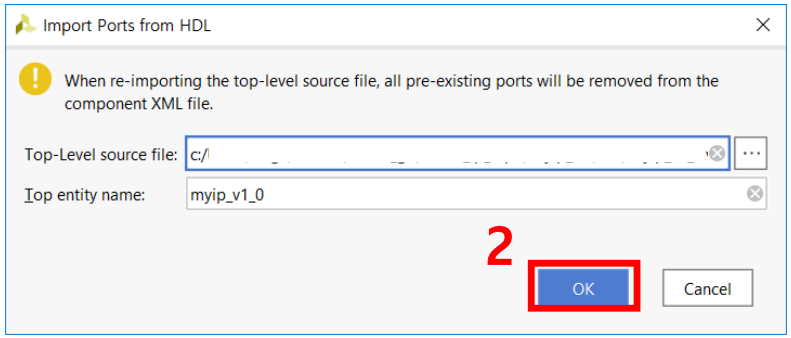
1. Import IP parameters you've changed in file groups

Checking IP Configurations

- Port configuration
- Ports and Interfaces -> Import IP ports

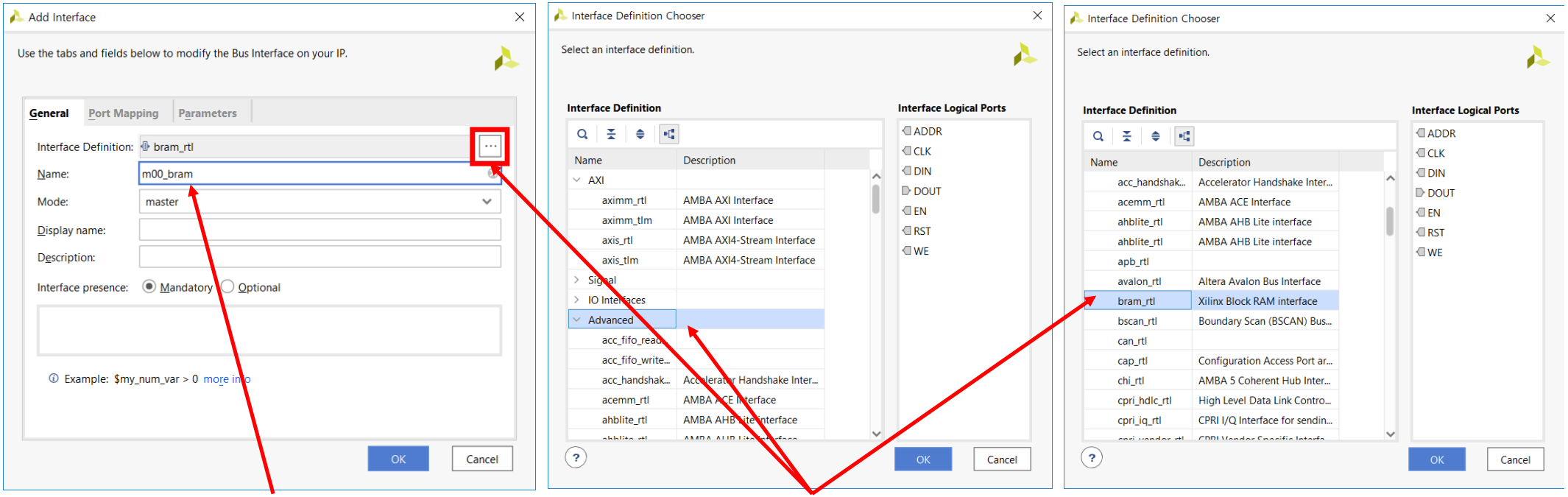


1. Import IP ports you've changed in file groups



Checking IP Configurations

- Port configuration
- Grouping ports - case1 : when the group doesn't exist
- ex) m00_bram

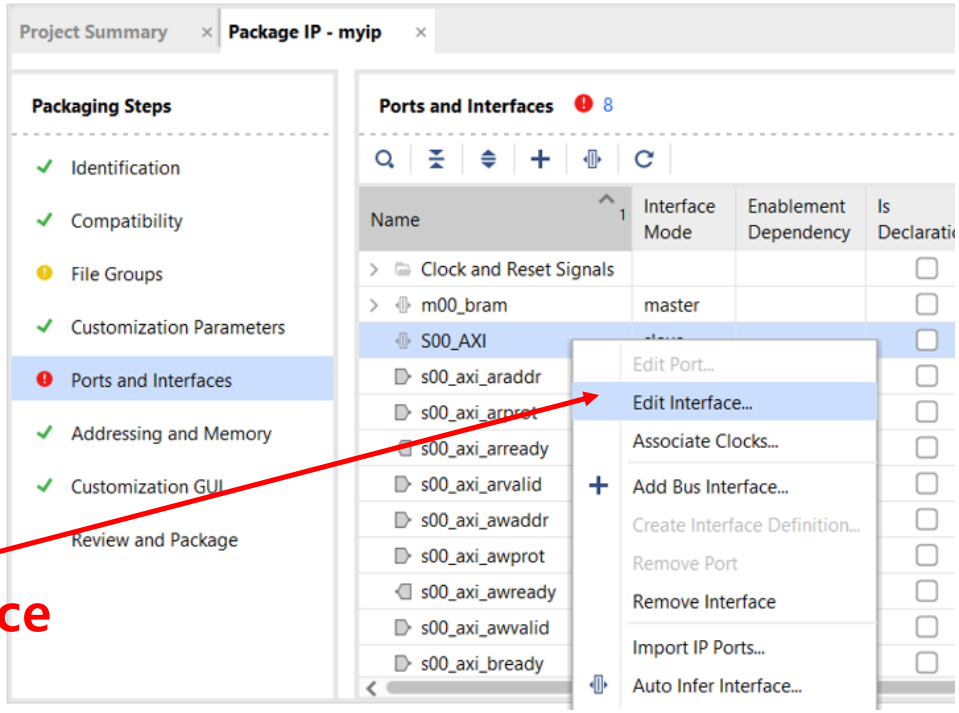


1. Set your port name

2. Advanced – bram_rtl

Checking IP Configurations

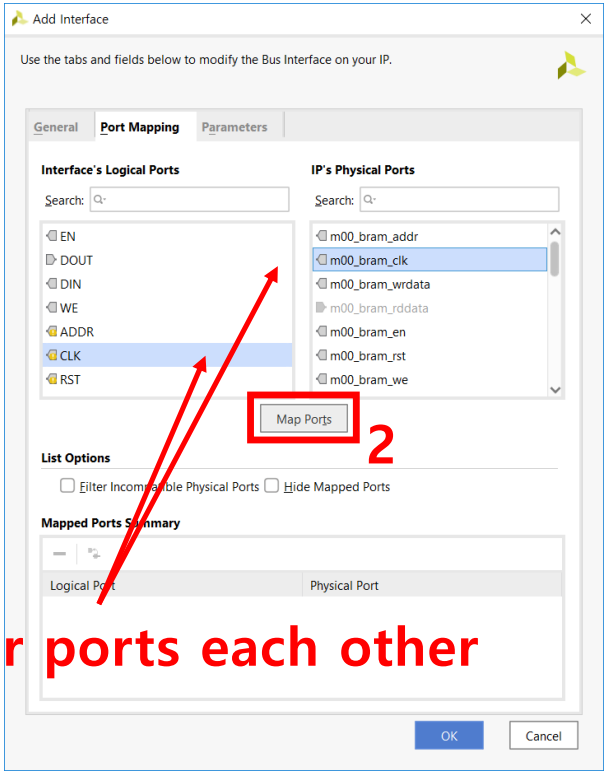
- Port configuration
- Grouping ports – case2 : when the group exists
- ex) S00_AXI



1. Edit Interface

Checking IP Configurations

- Port configuration
- Grouping ports – case1 & case2 both common



1. Match your ports each other

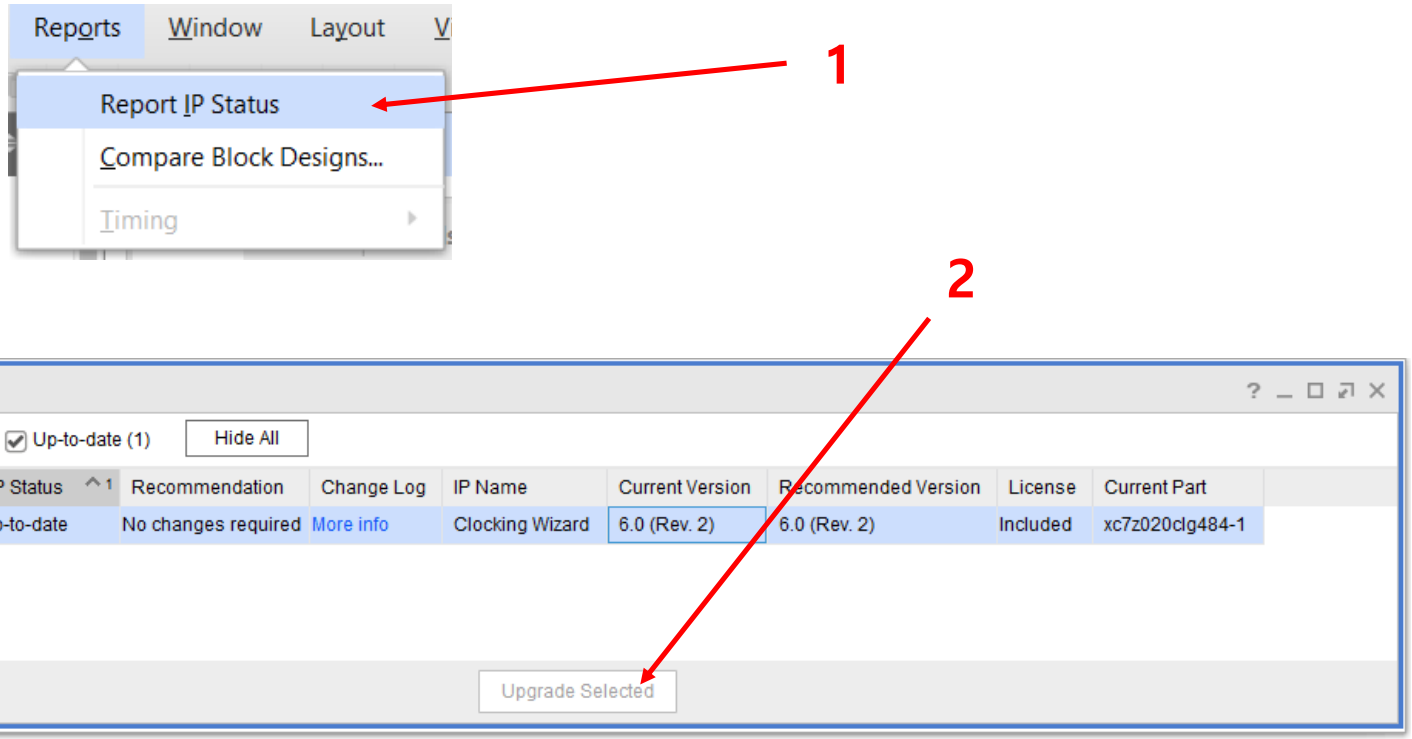
Mapped Ports Summary

Logical Port	Physical Port
CLK	m00_bram_clk
RST	m00_bram_rst
ADDR	m00_bram_addr
WE	m00_bram_we
EN	m00_bram_en
DOUT	m00_bram_rddata
DIN	m00_bram_wrdata

3. Check your port mapping

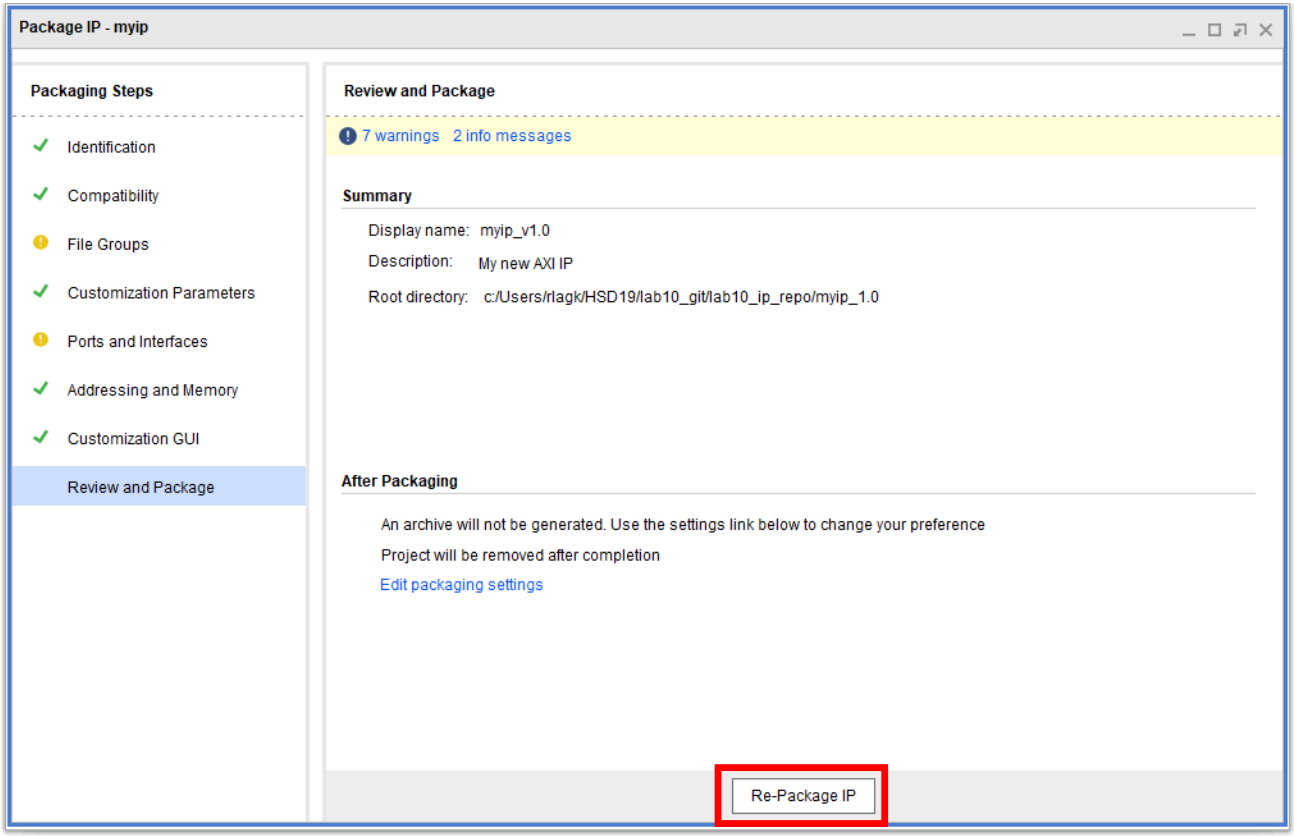
Upgrading Custom IP

- Execute update for existing Ips, if your project needed.



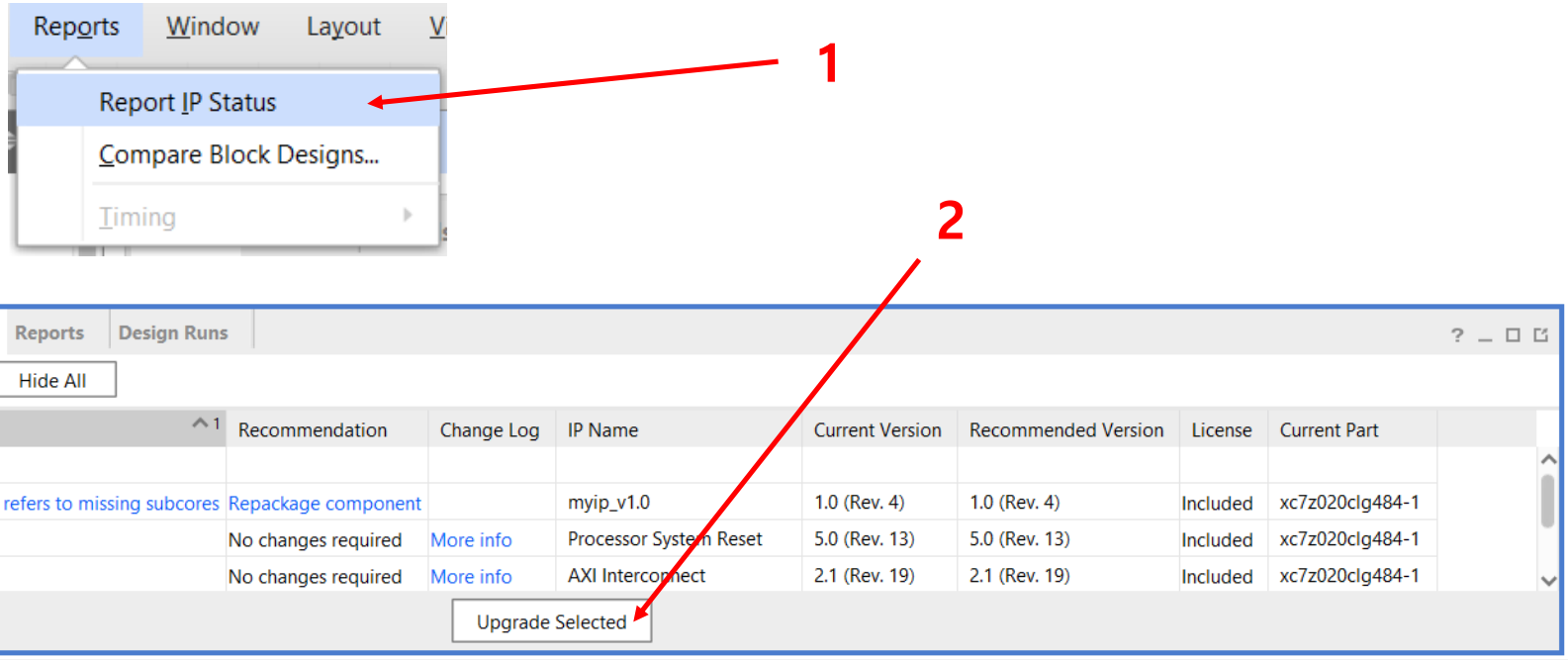
Checking IP Configurations

- Re-Package IP



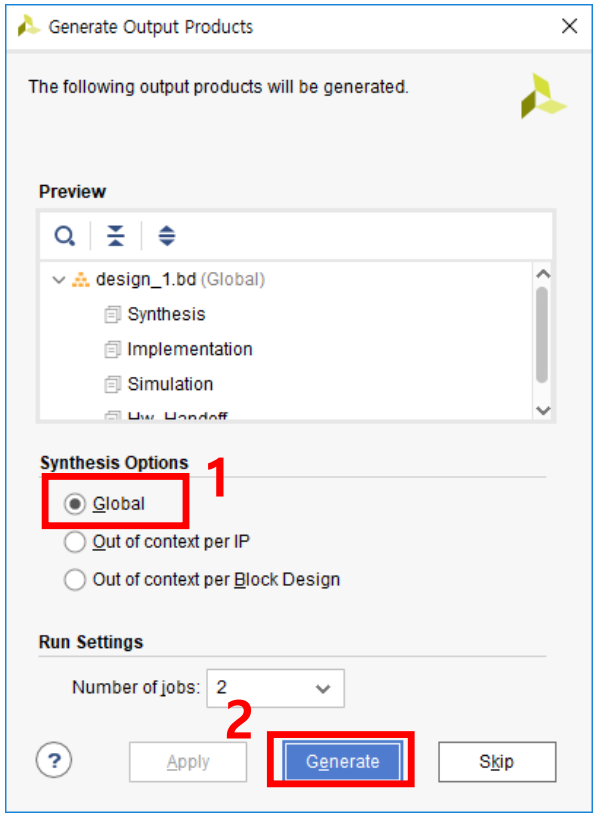
Upgrading Custom IP

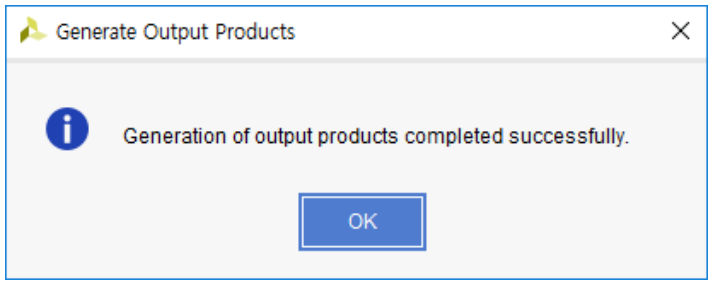
- Report IP Status -> Upgrade Selected
- You always have to update IP Status when you revise your IPs.

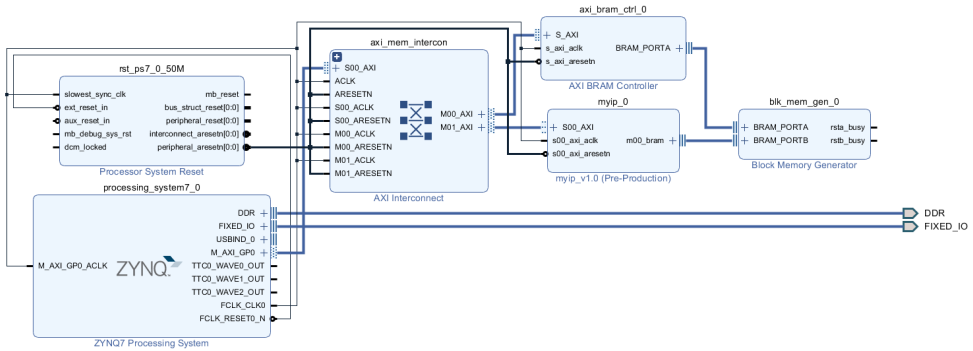


Upgrading Custom IP

- Don’t forget to generate your output products with global synthesis option.

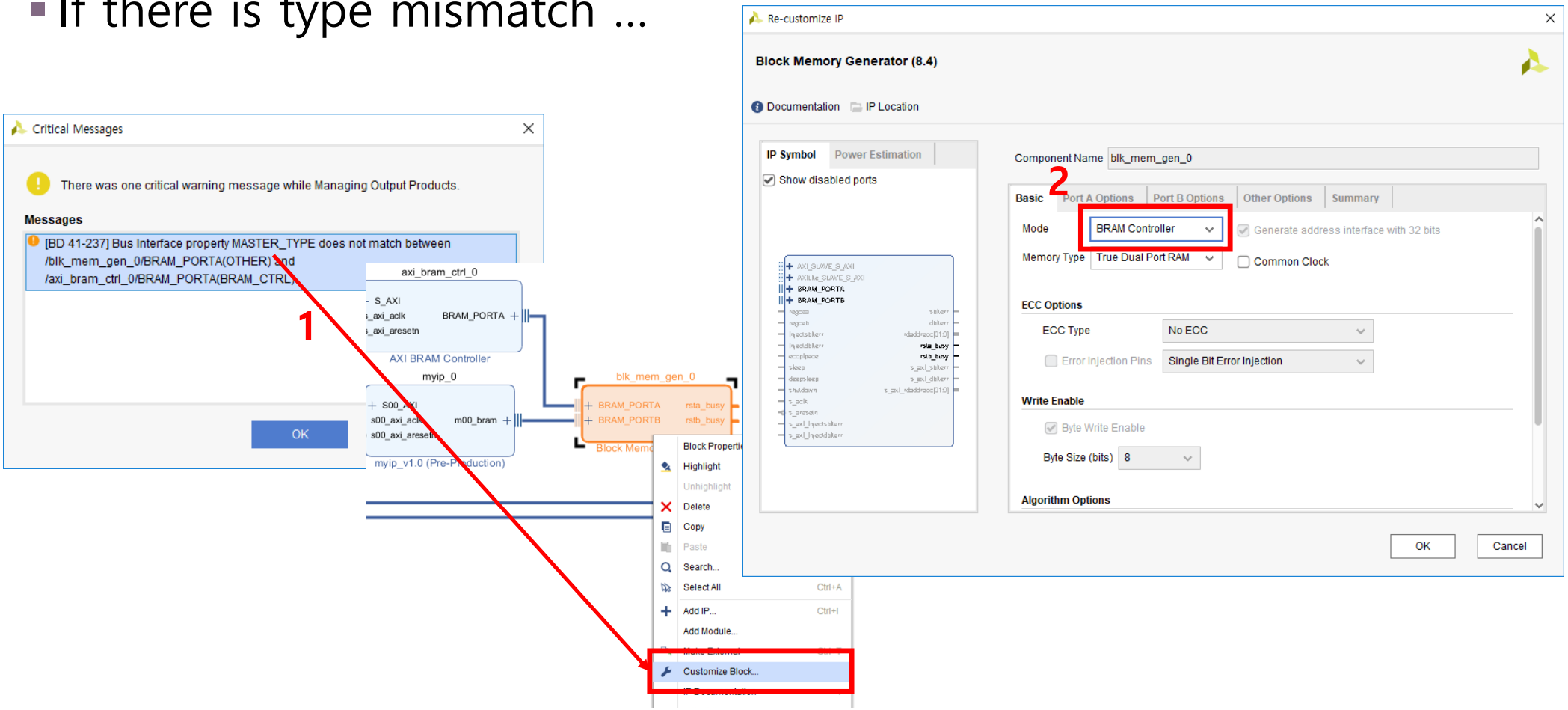






Type mismatch(General IP)

- If there is type mismatch …



Type mismatch(Custom IP)

- In custom IP case, you can solve every problems in IP Packager.