[Microprocessor Applications]
Lab 1: Board Test

Chester Sungchung Park

SoC Design Lab, Konkuk University

Webpage: http://soclab.konkuk.ac.kr

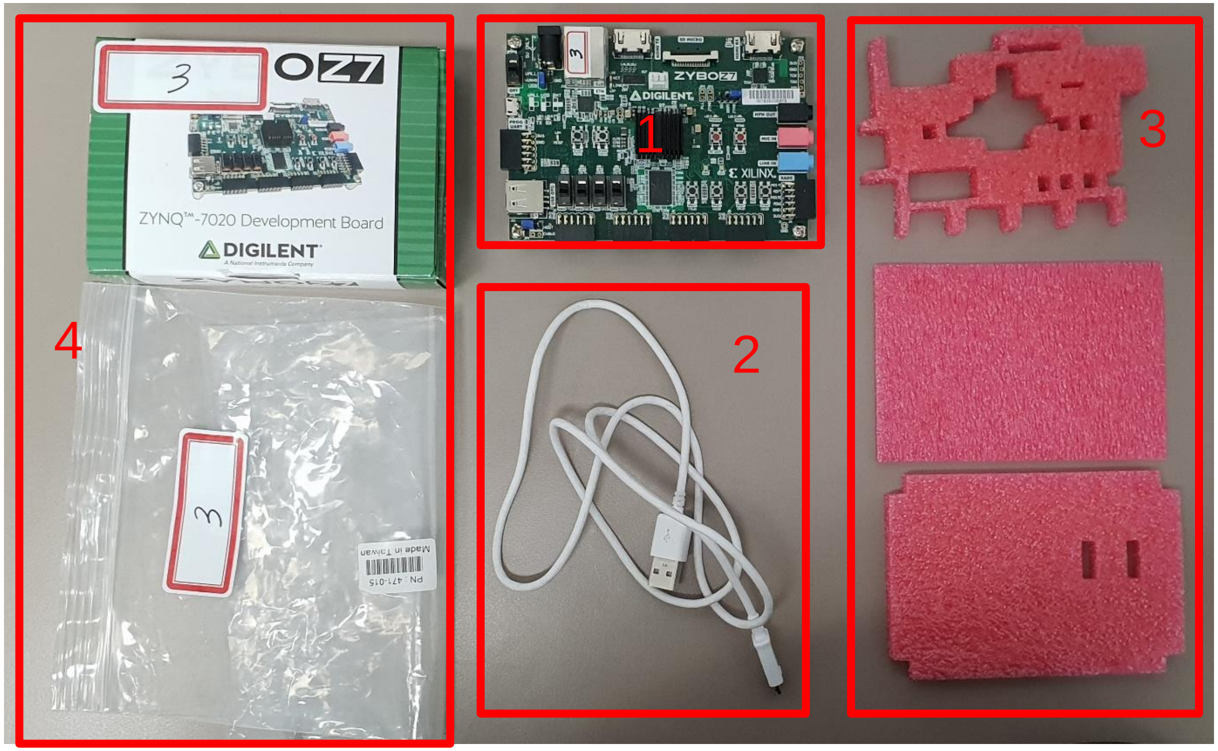
Outline

❑ Introduction

❑ Board test

- Vivado
  - ✓Creating projects
  - ✓Creating block designs
  - ✓Generating bitstream
- SDK (SW Development Kit)
  - ✓Running C applications

What’s Together w/ ZYBO



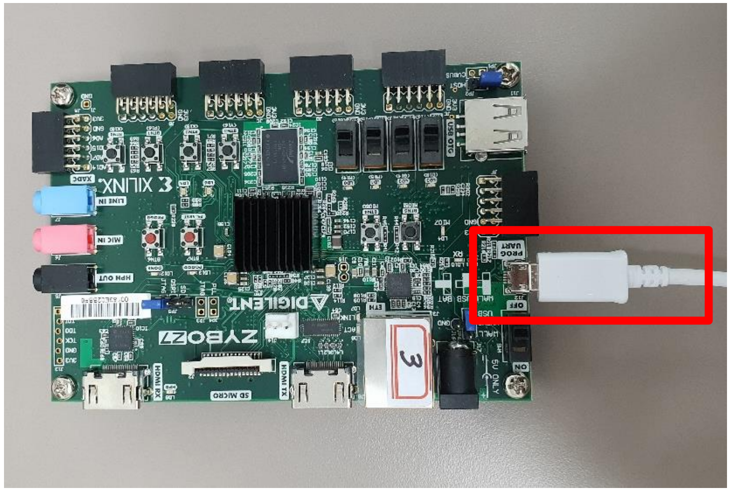
① ZYBO Z7-20 (Z7-10)

② USB-A to Micro-USB-B Cable

③ ZYBO Board Soft Case

④ ZYBO Board Hard Case

Connecting to ZYBO

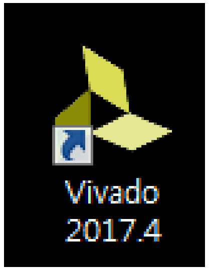


❑ Connect J12 (USB-UART port) to the PC using the Micro USB cable.

Creating Projects

❑ Run the Vivado

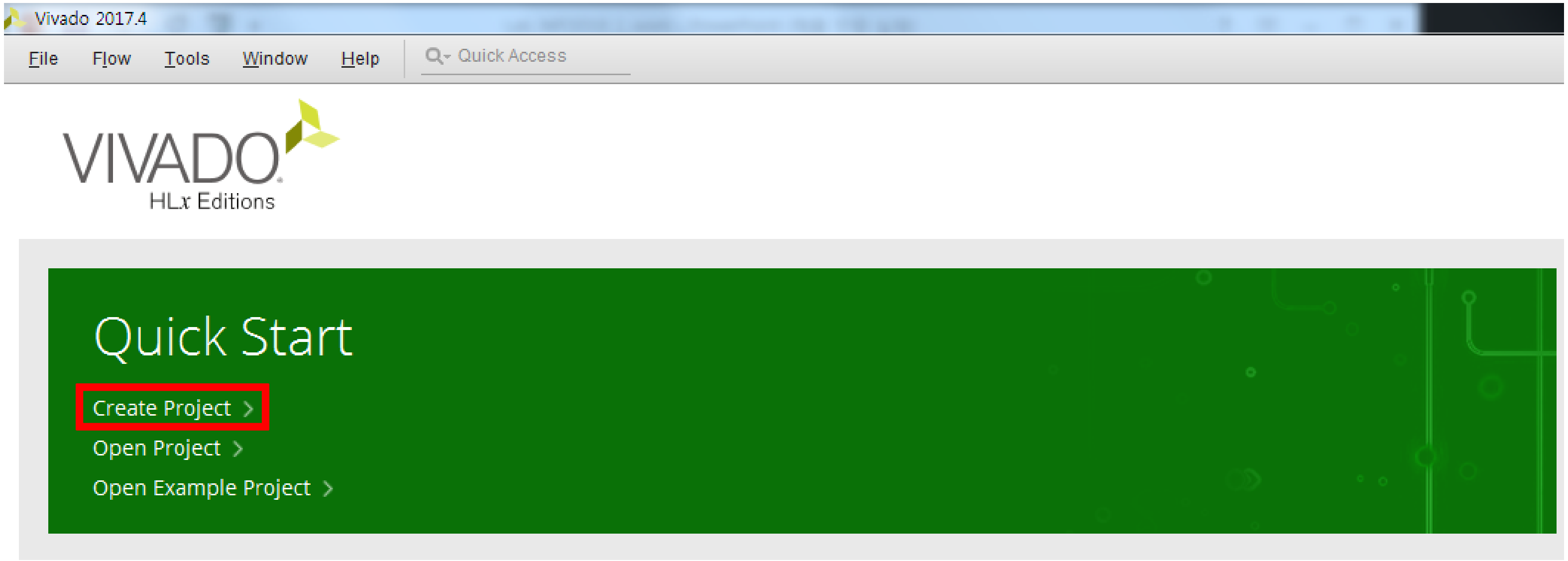
- From the Window desktop, double-click the ***'Vivado 2017.4'*** icon.



Creating Projects

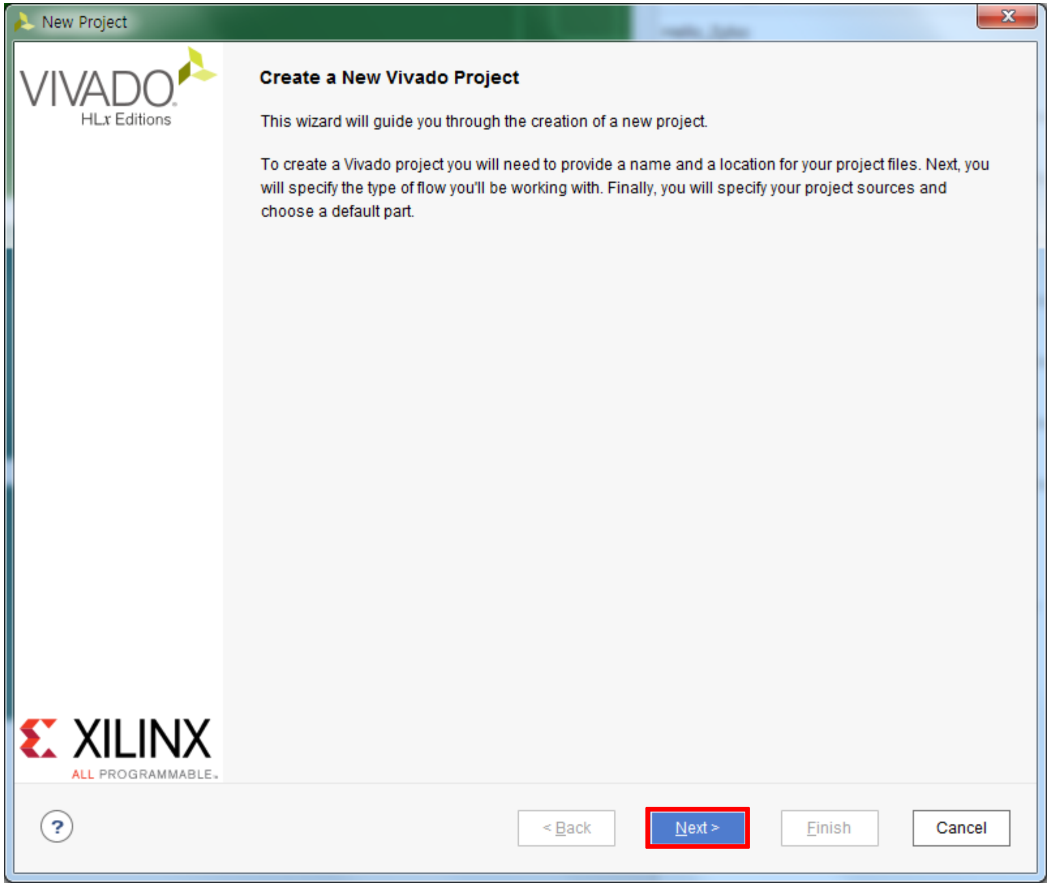
❑ Get Started

- Click ‘***Create Project***’



Creating Projects

- Create a New Vivado Project
  - Click ‘***Next’***



Create a New Vivado Project

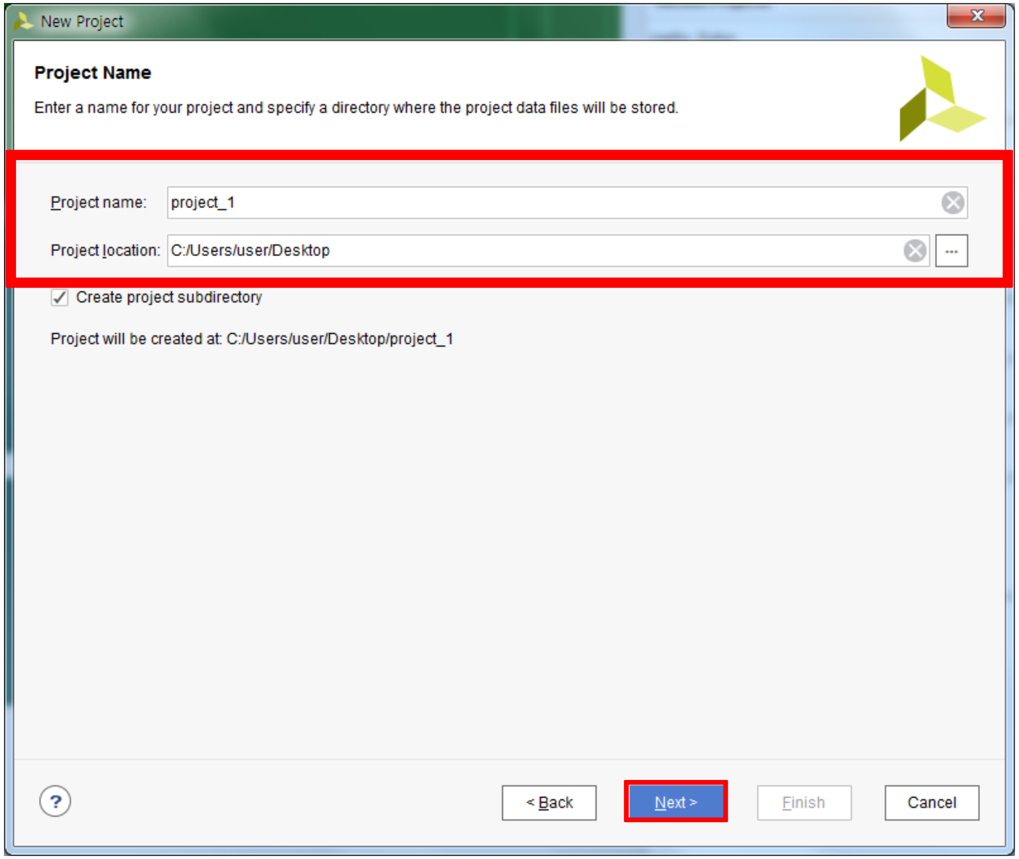
This wizard will guide you through the creation of a new project.

To create a Vivado project you will need to provide a name and a location for your project files. Next, you will specify the type of flow you'll be working with. Finally, you will specify your project sources and choose a default part.

Creating Projects

❑ Enter Project Name

- Type ‘***Project name’*** and choose ***‘Project location’***
- Click ‘***Next’***



Project Name

Enter a name for your project and specify a directory where the project data files will be stored.

Project name: project_1

Project location: C:/Users/user/Desktop

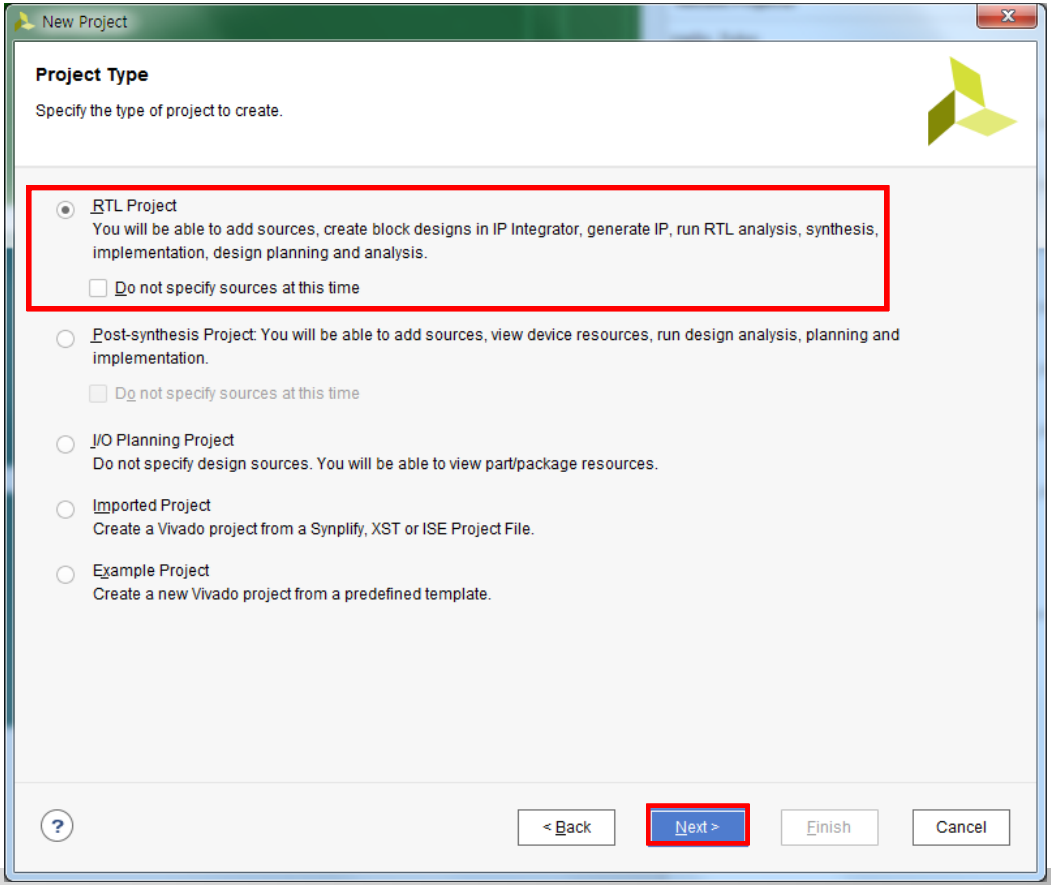
☑ Create project subdirectory

Project will be created at: C:/Users/user/Desktop/project_1

Creating Projects

❑ Choose Project Type

- Click ***'RTL Project'*** and then click ***'Next'***



Project Type

Specify the type of project to create.

● RTL Project
You will be able to add sources, create block designs in IP Integrator, generate IP, run RTL analysis, synthesis, implementation, design planning and analysis.

□ Do not specify sources at this time

○ Post-synthesis Project: You will be able to add sources, view device resources, run design analysis, planning and implementation.

□ Do not specify sources at this time

○ I/O Planning Project
Do not specify design sources. You will be able to view part/package resources.

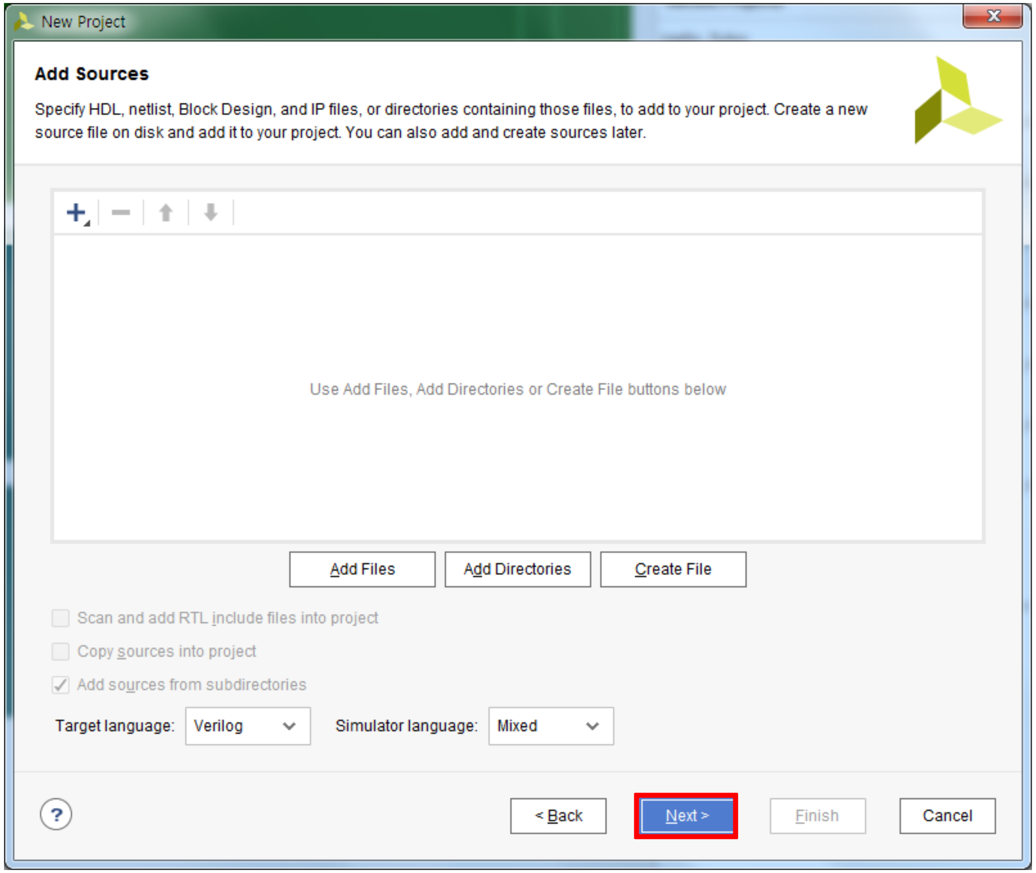
○ Imported Project
Create a Vivado project from a Synplify, XST or ISE Project File.

○ Example Project
Create a new Vivado project from a predefined template.

Creating Projects

❑ Add Sources

- Click ‘***Next***’



New Project

Add Sources

Specify HDL, netlist, Block Design, and IP files, or directories containing those files, to add to your project. Create a new source file on disk and add it to your project. You can also add and create sources later.

Use Add Files, Add Directories or Create File buttons below

□ Scan and add RTL include files into project

□ Copy sources into project

☑ Add sources from subdirectories

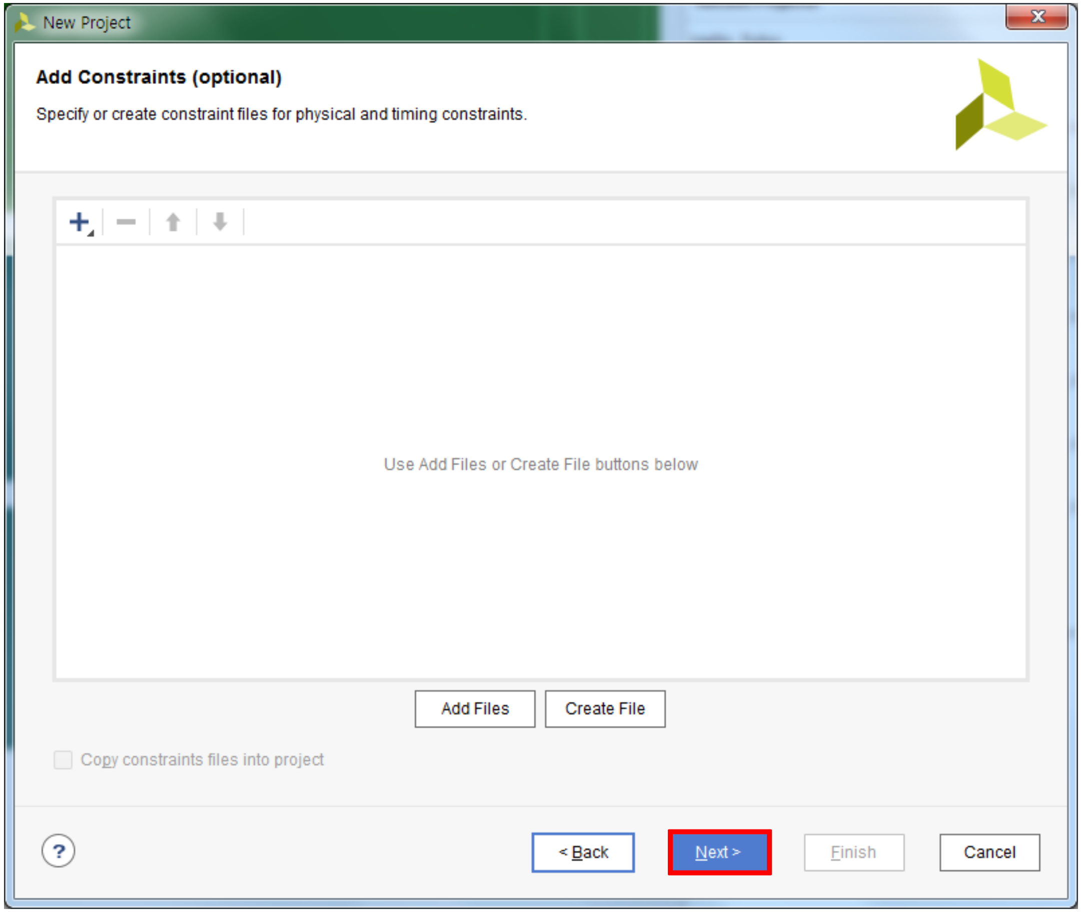
Target language: Verilog

Simulator language: Mixed

Creating Projects

❑ Add Constraints

- Click ‘***Next***’



New Project

Add Constraints (optional)

Specify or create constraint files for physical and timing constraints.

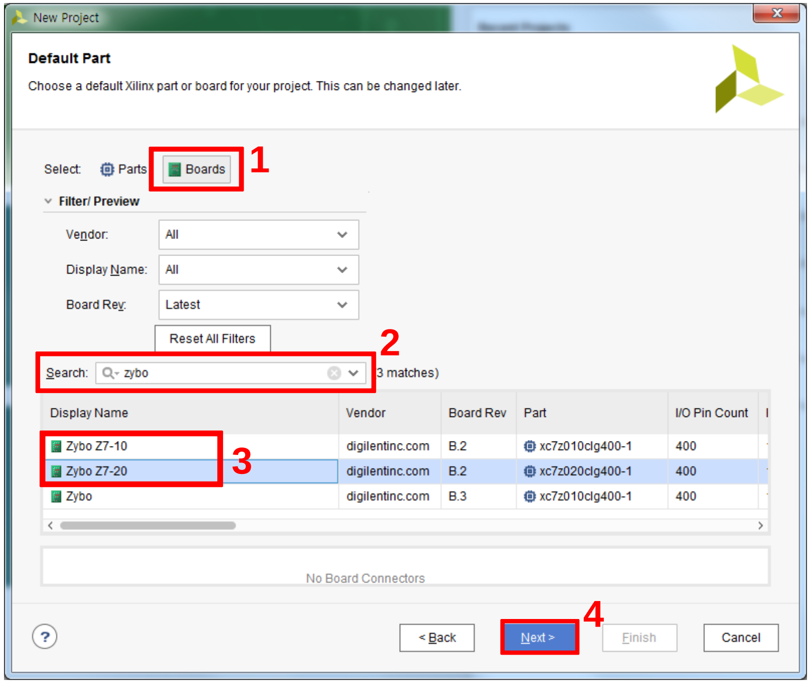
Use Add Files or Create File buttons below

□ Copy constraints files into project

Creating Projects

❑ Choose Default Part

- Select the ***'Boards'***
- Search for the ***'zybo'***
- Select the ***'Zybo Z7-20'*** or ***'Zybo Z7-10'***
- Click ***'Next'***



Default Part

Choose a default Xilinx part or board for your project. This can be changed later.

Select: ○ Parts ● Boards

Filter/ Preview

Vendor: All

Display Name: All

Board Rev: Latest

Search: zybo (3 matches)

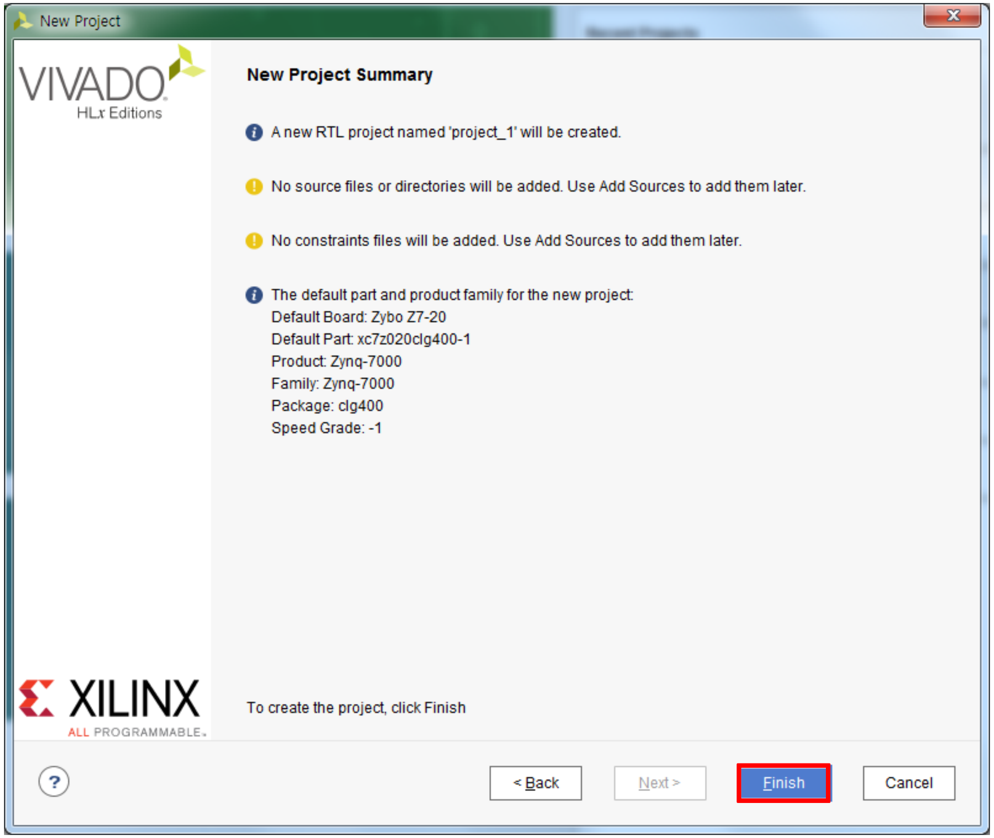
Display Name	Vendor	Board Rev	Part	I/O Pin Count	I
Zybo Z7-10	digilentinc.com	B.2	xc7z010clg400-1	400	
Zybo Z7-20	digilentinc.com	B.2	xc7z020clg400-1	400	
Zybo	digilentinc.com	B.3	xc7z010clg400-1	400	

No Board Connectors

Creating Projects

❑ Check New Project Summary

- Click ‘***Finish***’



New Project Summary

A new RTL project named 'project_1' will be created.

No source files or directories will be added. Use Add Sources to add them later.

No constraints files will be added. Use Add Sources to add them later.

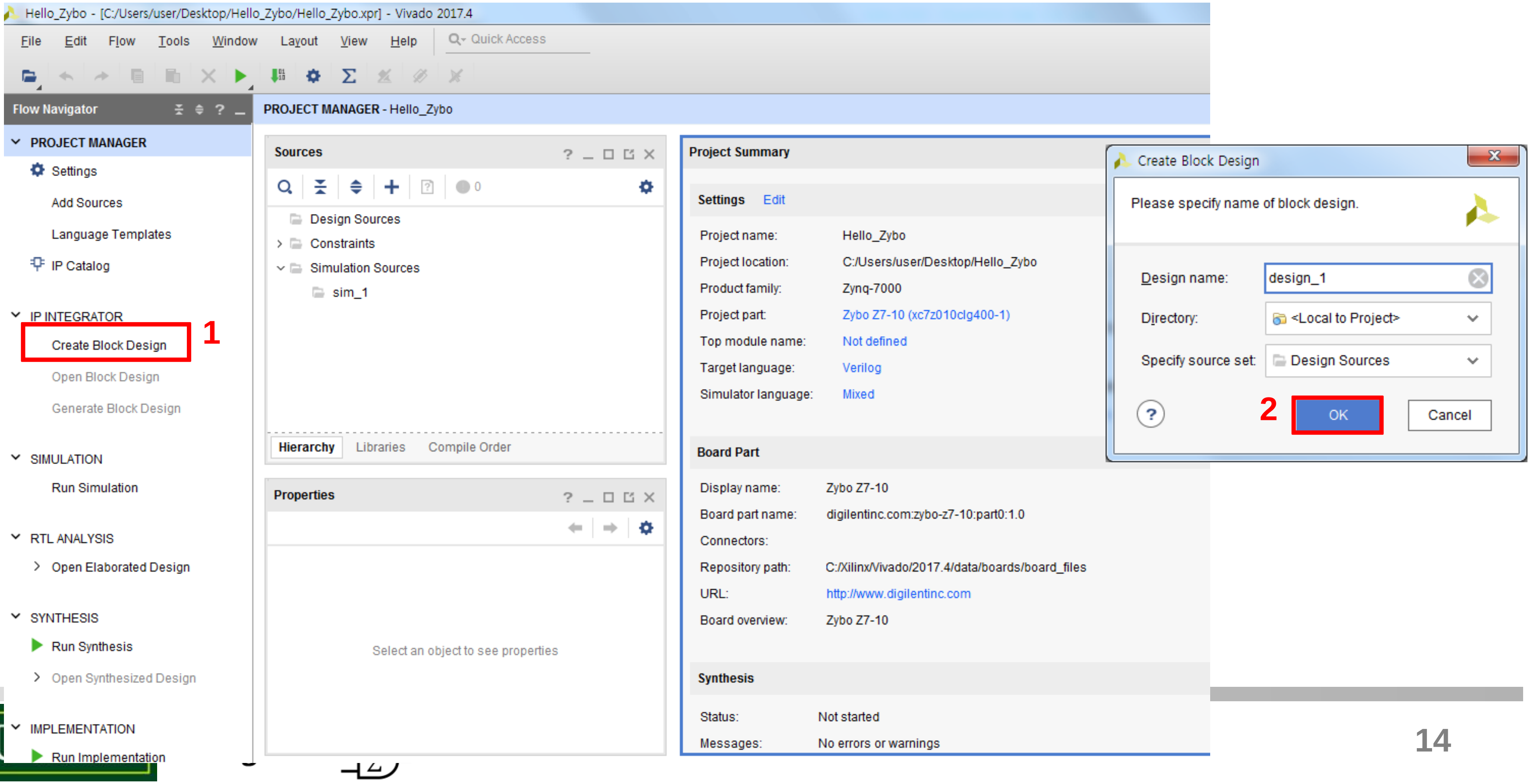
The default part and product family for the new project:
Default Board: Zybo Z7-20
Default Part: xc7z020clg400-1
Product: Zynq-7000
Family: Zynq-7000
Package: clg400
Speed Grade: -1

To create the project, click Finish

Creating Block Designs

❑ Create Block Design

- Click ‘***Create Block Design***’
- Type ‘***Design name***’ and then click ***‘OK’***



Hello_Zybo - [C:/Users/user/Desktop/Hello_Zybo/Hello_Zybo.xpr] - Vivado 2017.4

PROJECT MANAGER - Hello_Zybo

Project name:	Hello_Zybo
Project location:	C:/Users/user/Desktop/Hello_Zybo
Product family:	Zynq-7000
Project part:	Zybo Z7-10 (xc7z010clg400-1)
Top module name:	Not defined
Target language:	Verilog
Simulator language:	Mixed

Display name:	Zybo Z7-10
Board part name:	digilentinc.com:zybo-z7-10:part0:1.0
Connectors:	
Repository path:	C:/Xilinx/Vivado/2017.4/data/boards/board_files
URL:	http://www.digilentinc.com
Board overview:	Zybo Z7-10

Status:	Not started
Messages:	No errors or warnings

Create Block Design

Please specify name of block design.

Design name:	design_1
Directory:	<Local to Project>
Specify source set:	Design Sources

Creating Block Designs

❑ Add Processing System

- Click the '+' ***(Add IP Button)*** and then type '***zynq***' in the search field
- Double-click '***ZYNQ7 Processing System***'

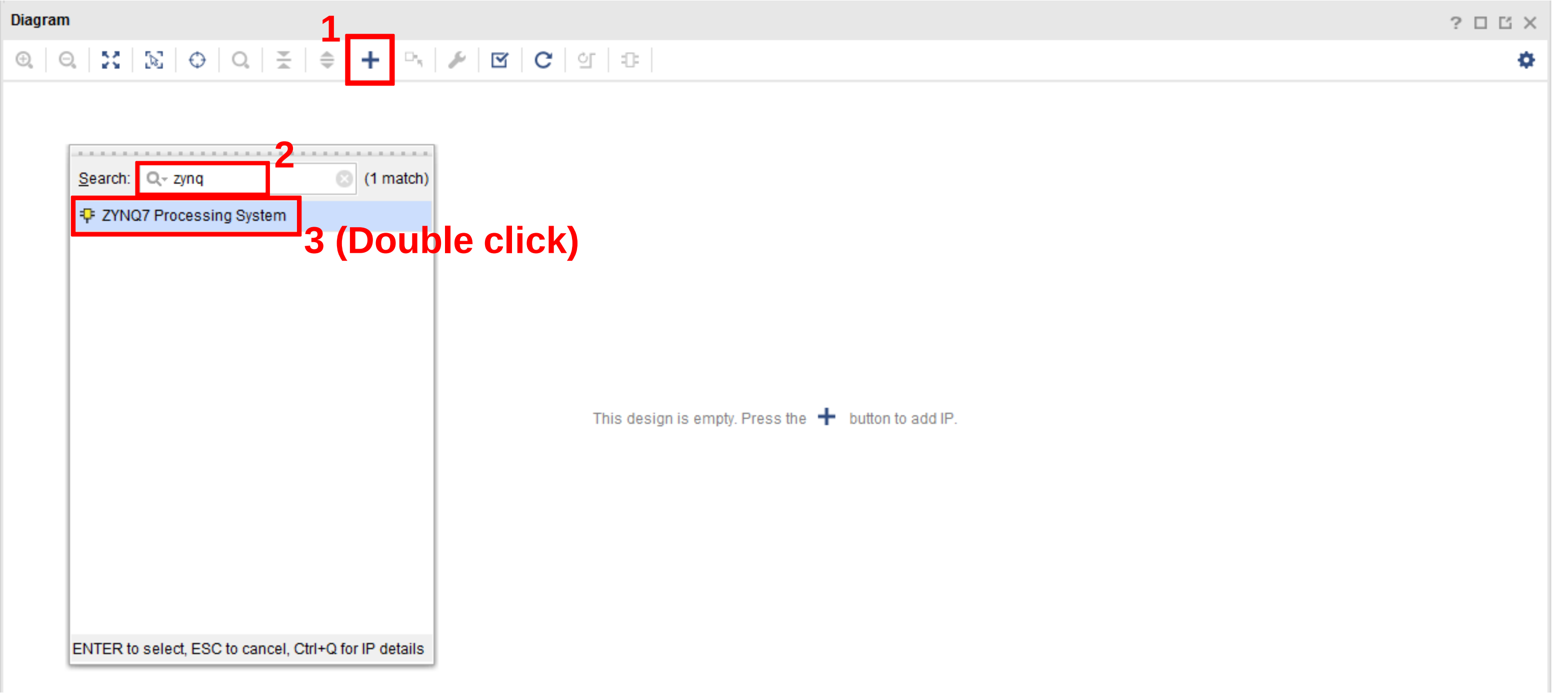
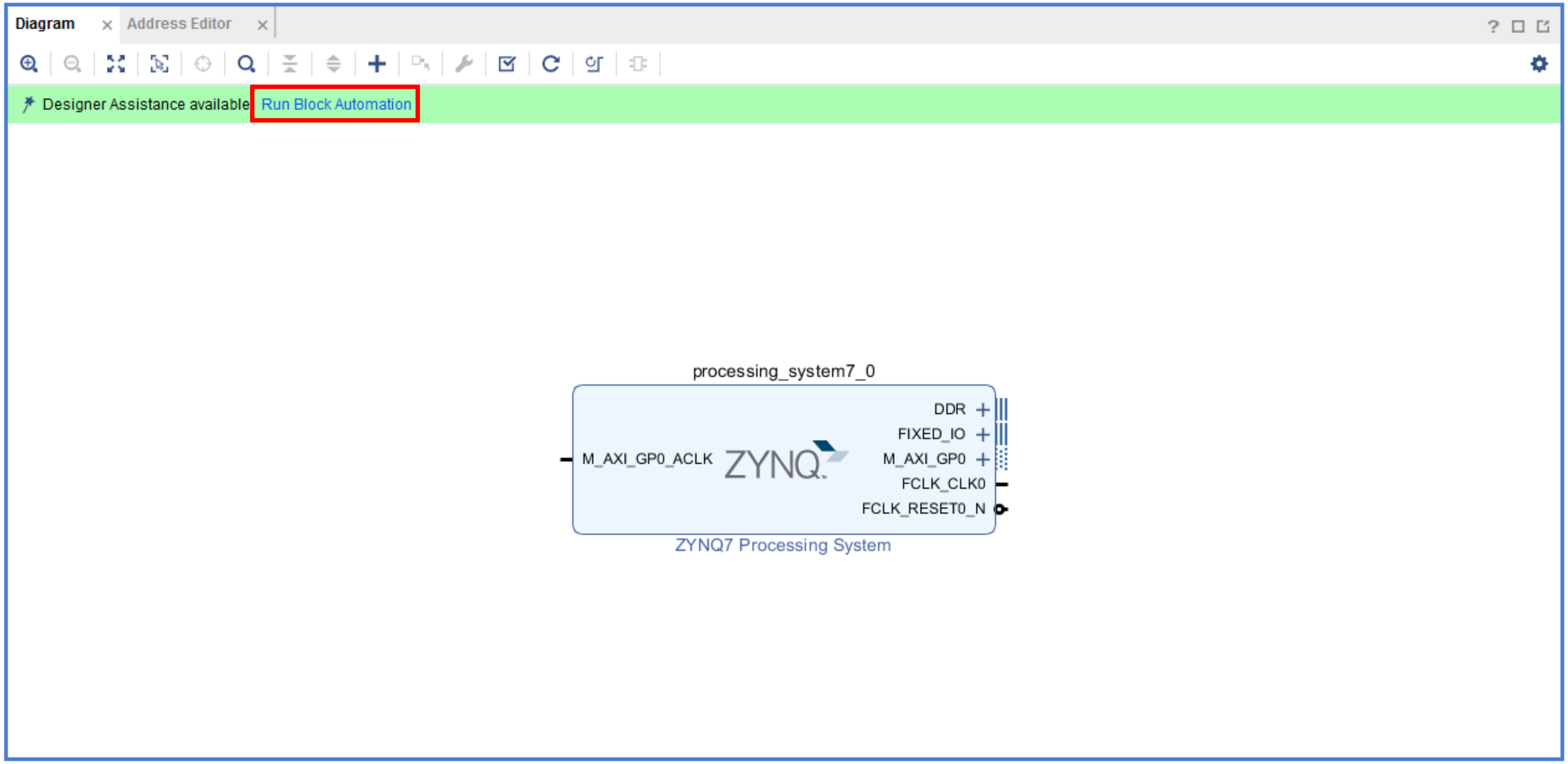


Diagram
1
2
Search: zynq
(1 match)
ZYNQ7 Processing System
3 (Double click)
This design is empty. Press the + button to add IP.
ENTER to select, ESC to cancel, Ctrl+Q for IP details

Creating Block Designs

❑ Make external connection

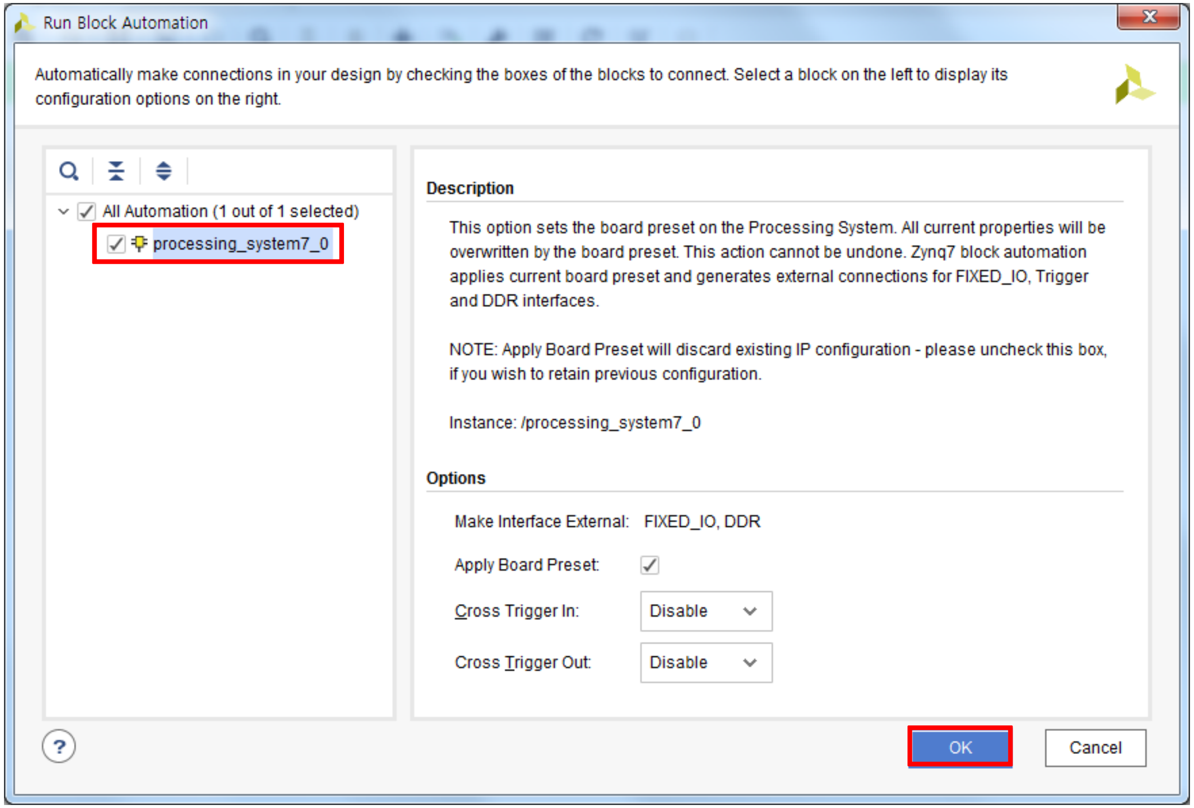
- Click ‘***Run Block Automation***’



Creating Block Designs

❑ Make external connection (cont'd)

- Click ***'processing_system7_0'*** > ***'OK'***



Automatically make connections in your design by checking the boxes of the blocks to connect. Select a block on the left to display its configuration options on the right.

☑ All Automation (1 out of 1 selected)

☑ processing_system7_0

Description

This option sets the board preset on the Processing System. All current properties will be overwritten by the board preset. This action cannot be undone. Zynq7 block automation applies current board preset and generates external connections for FIXED_IO, Trigger and DDR interfaces.

NOTE: Apply Board Preset will discard existing IP configuration - please uncheck this box, if you wish to retain previous configuration.

Instance: /processing_system7_0

Options

Make Interface External: FIXED_IO, DDR

Apply Board Preset: ☑

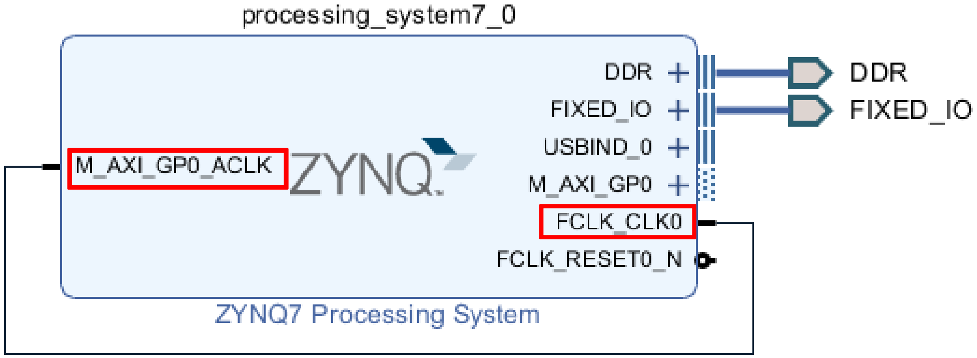
Cross Trigger In: Disable

Cross Trigger Out: Disable

Creating Block Designs

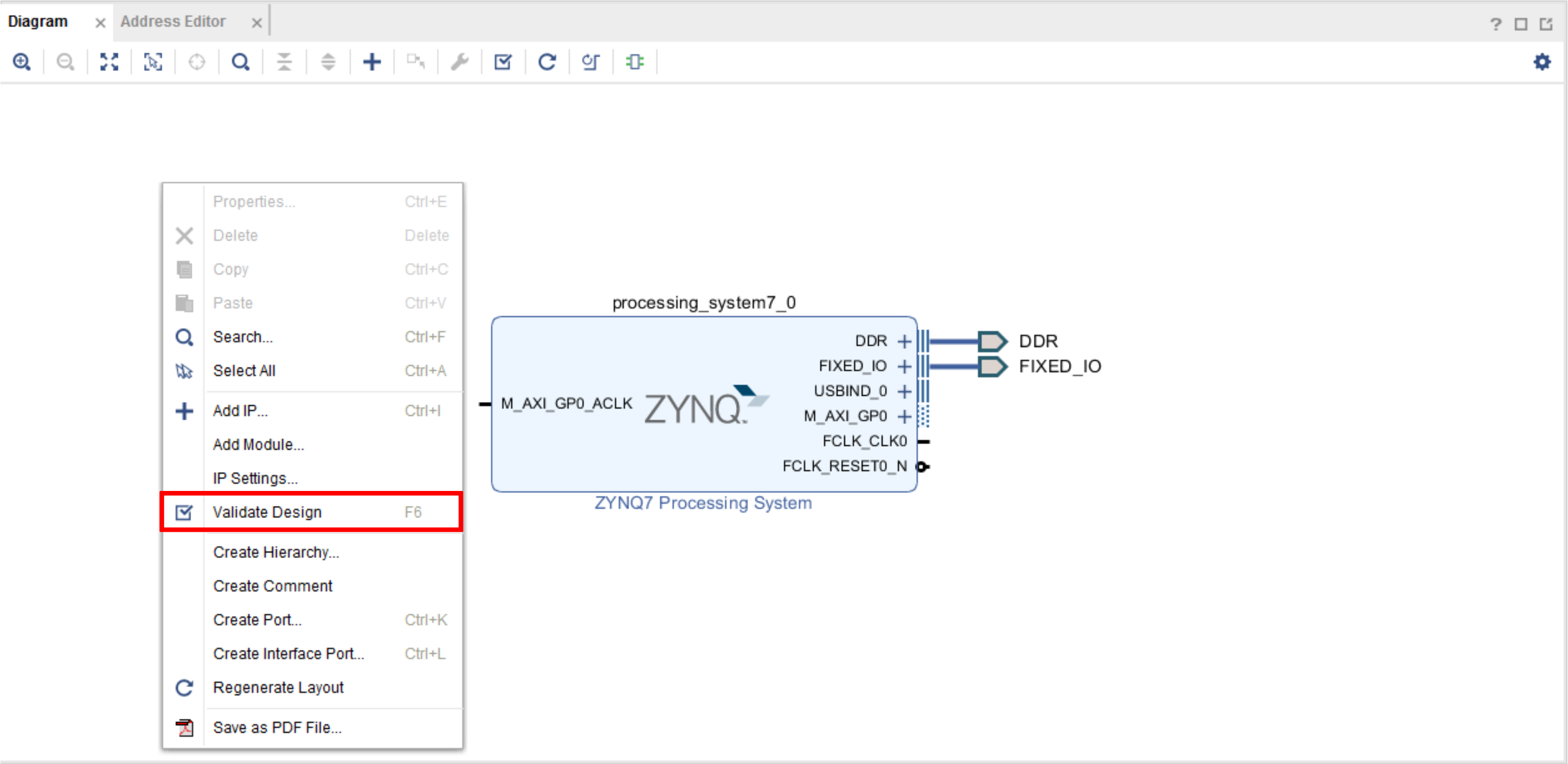
❑ Make connections

- Connect ***‘FLCK_CLK0’*** with ***‘M_AXI_GP0_ACLK’***



Creating Block Designs

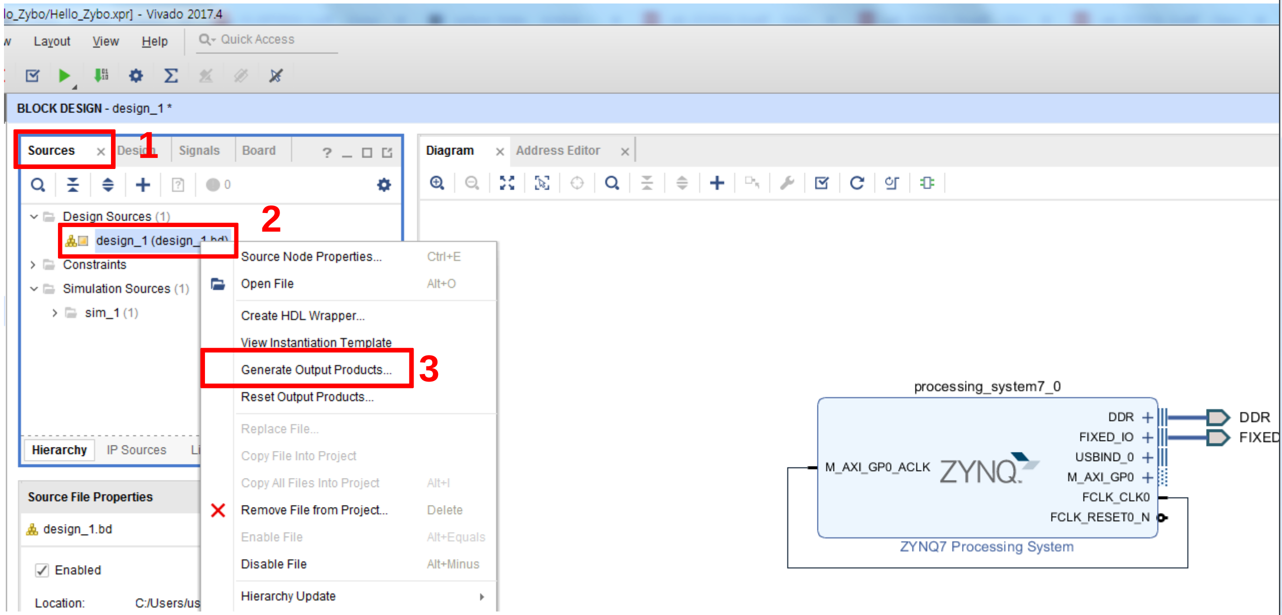
- ❑ Validate Design
  - Right-click and then click '***Validate Design***'

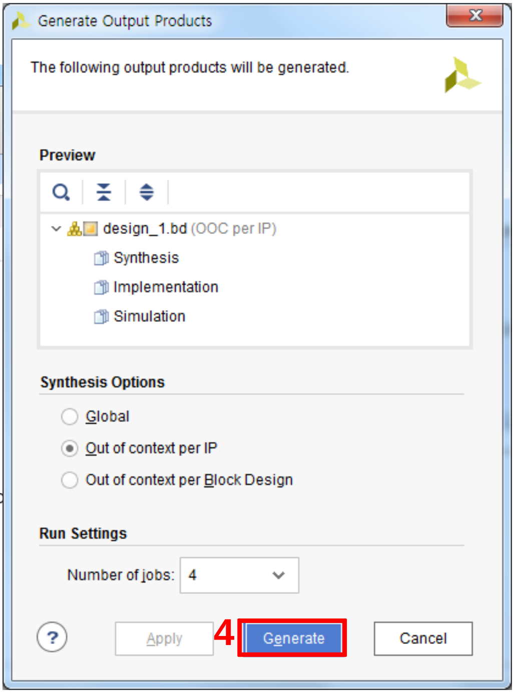


Creating Block Designs

❑ Generate Output Products

- Select the ‘***Design Sources***’ tab and then right-click the block diagram
- Click ‘***Generate Output Products’ > ‘Generate’***





Generate Output Products

The following output products will be generated.

Preview

- design_1.bd (OOC per IP)
  - Synthesis
  - Implementation
  - Simulation

Synthesis Options

○ Global
● Out of context per IP
○ Out of context per Block Design

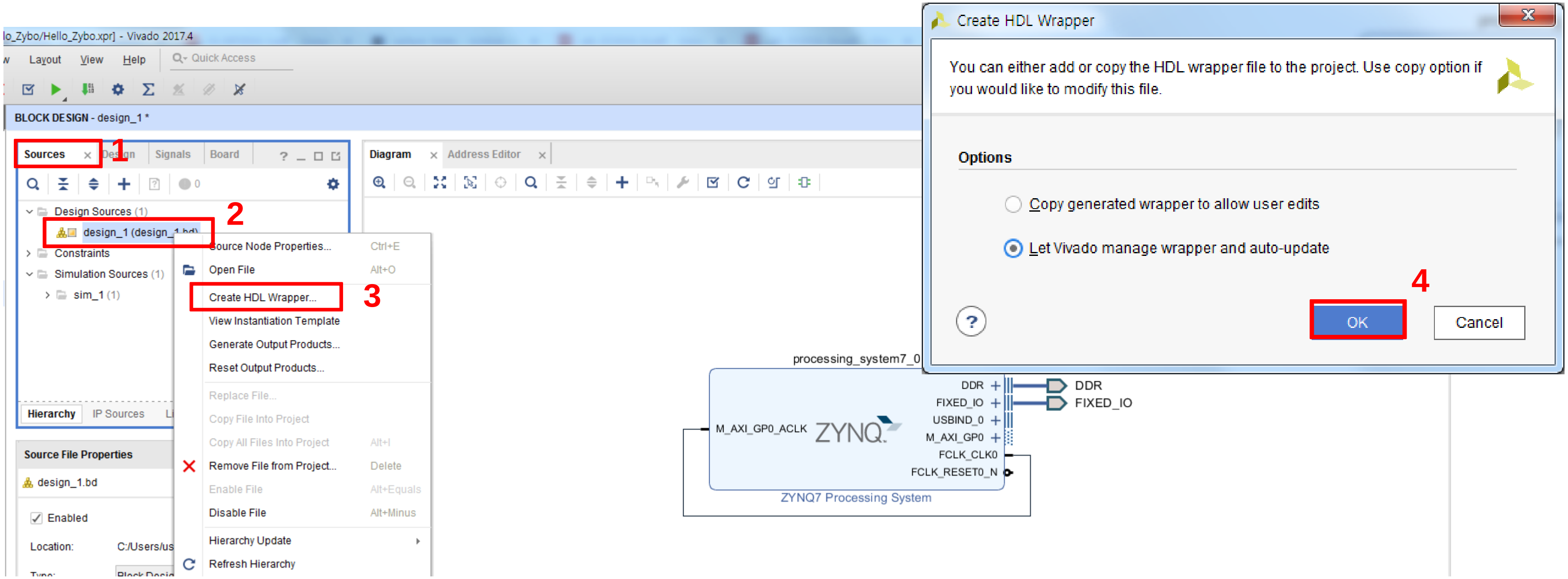
Run Settings

Number of jobs: 4

Creating Block Designs

❑ Create HDL Wrapper

- Select the ***‘Sources’*** tap and then right-click ***‘design_1’***
- Click **‘Create HDL Wrapper’ > ‘OK’**



Create HDL Wrapper

You can either add or copy the HDL wrapper file to the project. Use copy option if you would like to modify this file.

Options

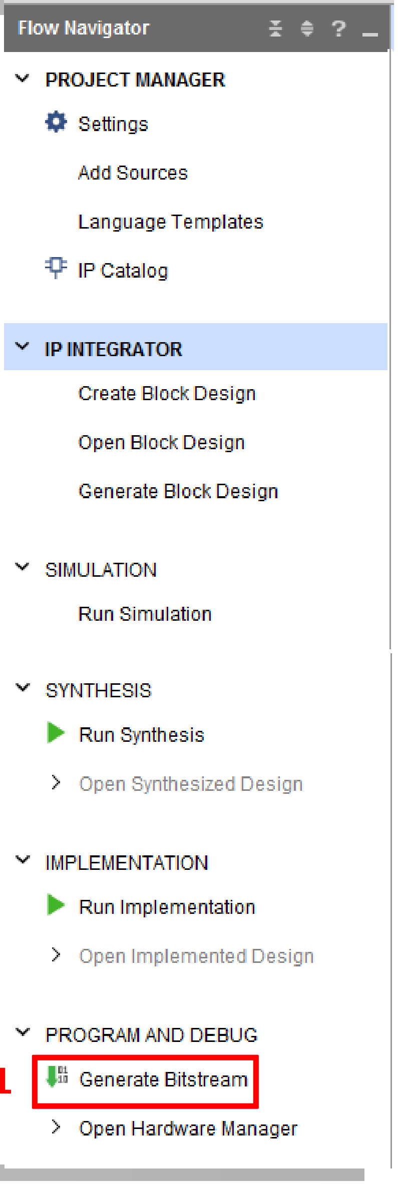
○ Copy generated wrapper to allow user edits

● Let Vivado manage wrapper and auto-update

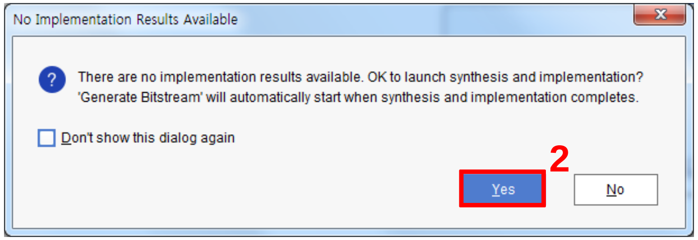
Generating Bitstream

❑ Generate Bitstream

- Click ‘***Generate Bitstream***’ at the bottom of the Flow Navigator.
- Click ‘***Yes***’ > ‘**OK**’

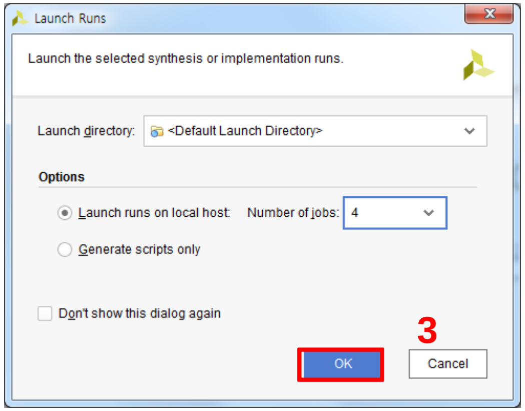


1



There are no implementation results available. OK to launch synthesis and implementation?
'Generate Bitstream' will automatically start when synthesis and implementation completes.

□ Don't show this dialog again



Launch the selected synthesis or implementation runs.

Launch directory: <Default Launch Directory>

Options

● Launch runs on local host: Number of jobs: 4

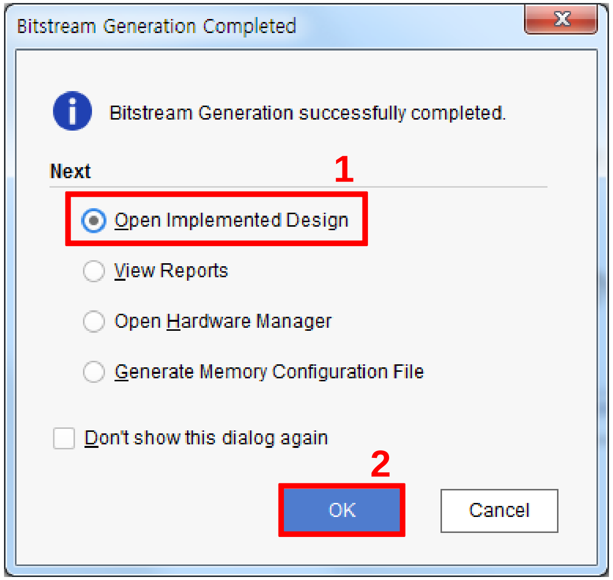
○ Generate scripts only

□ Don't show this dialog again

Generating Bitstream

❑ Generate Bitstream (cont'd)

- Once the Bitstream Generation ends, choose ***'Open Implemented Design' > 'OK'***



Bitstream Generation Completed

Bitstream Generation successfully completed.

Next

1

● Open Implemented Design

○ View Reports

○ Open Hardware Manager

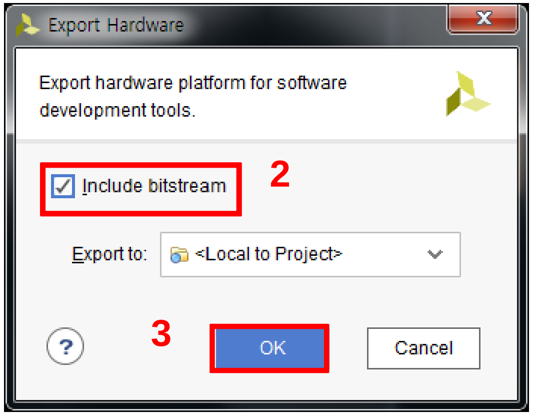
○ Generate Memory Configuration File

□ Don't show this dialog again

2

Generating Bitstream

- ❑ Export Hardware for SDK
  - Open the ***'File'*** menu and choose ***'Export'*** **>** ***'Export Hardware'***
  - Click '***Include bitstream***' **> 'OK'**

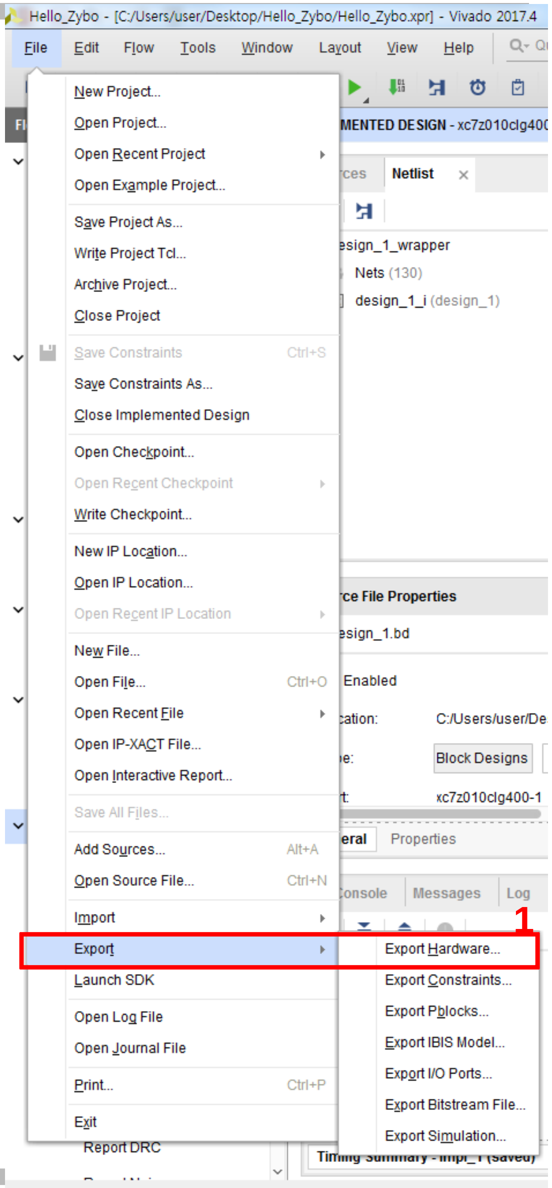


Export Hardware

Export hardware platform for software development tools.

☑ Include bitstream

Export to: <Local to Project>

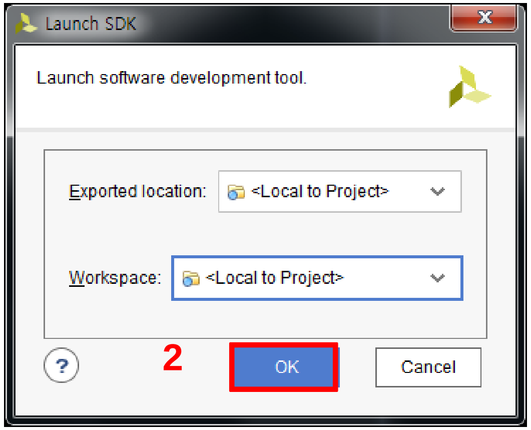


Hello_Zybo - [C:/Users/user/Desktop/Hello_Zybo/Hello_Zybo.xpr] - Vivado 2017.4

Generating Bitstream

❑ Launch SDK

- Open the ***'File'*** menu and then click ***'Launch SDK' > 'OK'***

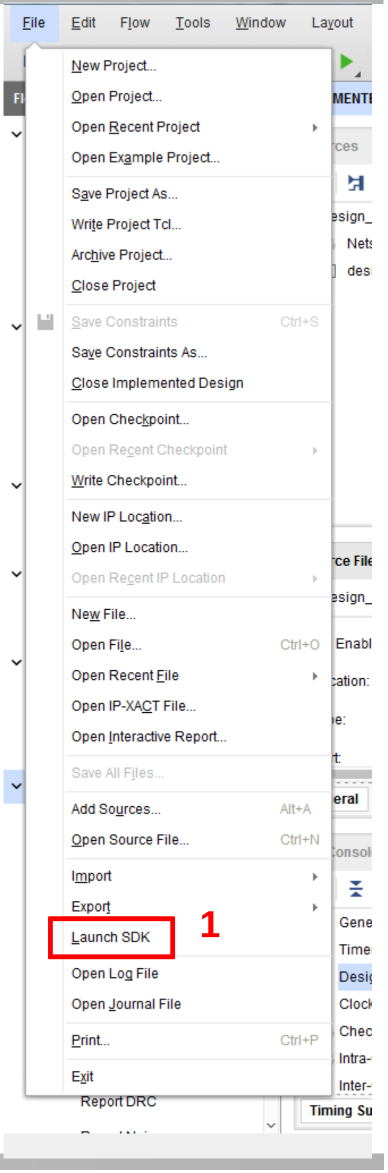


Launch software development tool.

Exported location: <Local to Project>

Workspace: <Local to Project>

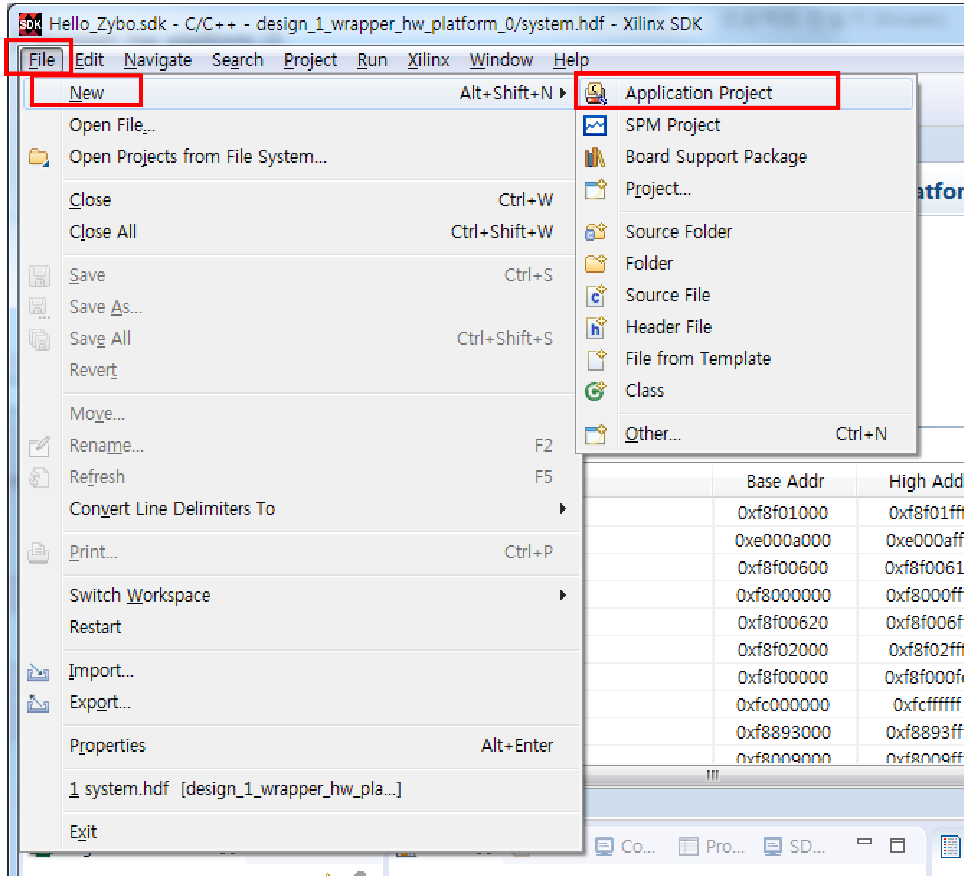
2



1

Running C Applications

- Create a C application project
  - Click ***‘File’ > ‘New’ > ‘Application Project’***

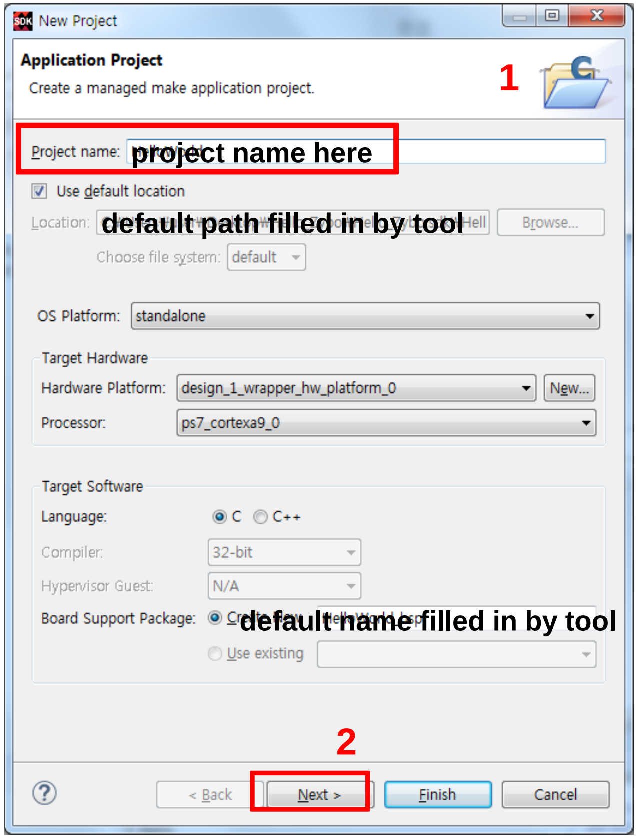


Base Addr	High Add
0xf8f01000	0xf8f01ff
0xe000a000	0xe000aff
0xf8f00600	0xf8f0061
0xf8000000	0xf8000ff
0xf8f00620	0xf8f006f
0xf8f02000	0xf8f02ff
0xf8f00000	0xf8f000f
0xfc000000	0xfcffffff
0xf8893000	0xf8893ff
0xf8009000	0xf8009ff

Running C Applications

❑ Create a C application project (cont'd)

- Type ***<your project name>*** in the Project name field
- The ***'Board Support Package'*** field can be set up to use an existing BSP or a new BSP can be created based on the project name. (Do not modify)



New Project

Application Project

Create a managed make application project.

1

Project name: project name here

☑ Use default location

Location: default path filled in by tool

Choose file system: default

OS Platform: standalone

Target Hardware

Hardware Platform: design_1_wrapper_hw_platform_0

Processor: ps7_cortexa9_0

Target Software

Language: ● C ○ C++

Compiler: 32-bit

Hypervisor Guest: N/A

Board Support Package: ● Create New default name filled in by tool

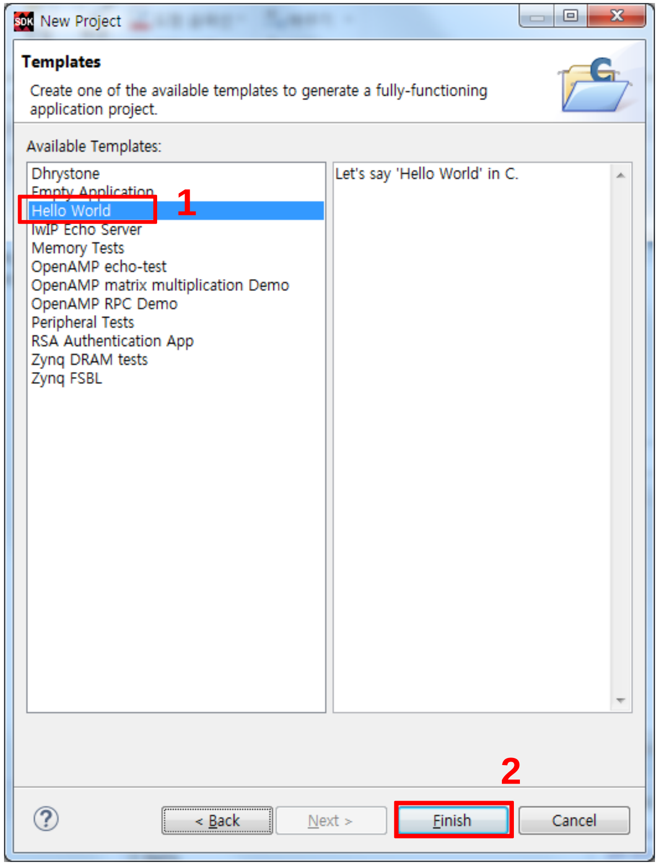
○ Use existing

2

Running C Applications

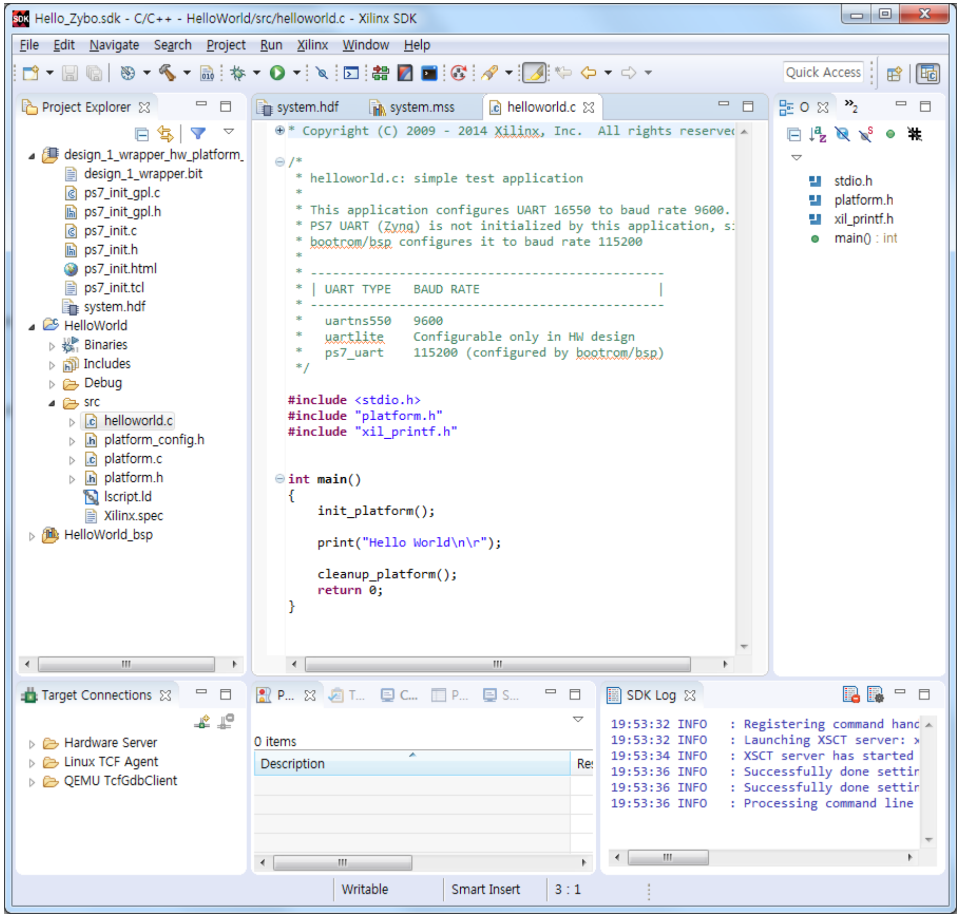
❑ Create a C application project (cont'd)

- Select '***Hello World***' from the template list
- Click '***Finish***'



Running C Applications

- ❑ Review the source code ‘***helloworld.c***’

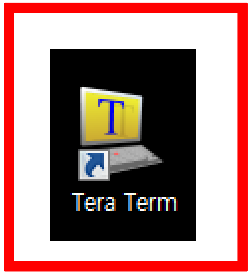


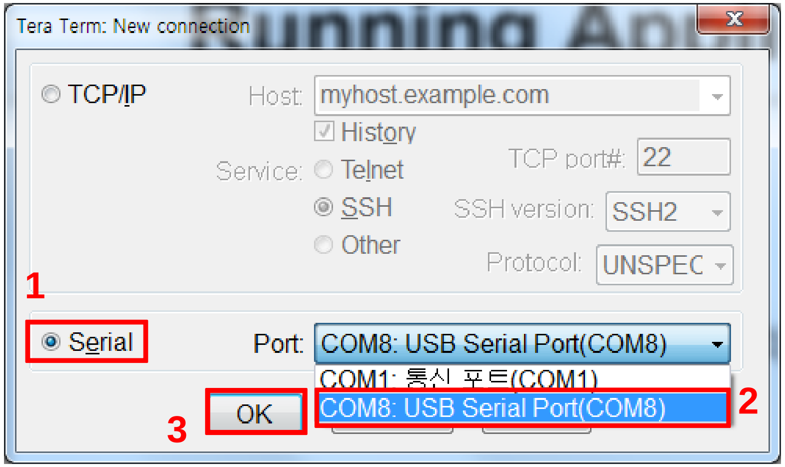
```c
* Copyright (C) 2009 - 2014 Xilinx, Inc.  All rights reservec

/*
 * helloworld.c: simple test application
 *
 * This application configures UART 16550 to baud rate 9600.
 * PS7 UART (Zynq) is not initialized by this application, s:
 * bootrom/bsp configures it to baud rate 115200
 *
 * ------------------------------------------------
 * | UART TYPE   BAUD RATE                        |
 * ------------------------------------------------
 *   uartns550   9600
 *   uartlite    Configurable only in HW design
 *   ps7_uart    115200 (configured by bootrom/bsp)
 */

#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"


int main()
{
    init_platform();

    print("Hello World\n\r");

    cleanup_platform();
    return 0;
}
```

```
19:53:32 INFO   : Registering command hanc
19:53:32 INFO   : Launching XSCT server: x
19:53:34 INFO   : XSCT server has started
19:53:36 INFO   : Successfully done settir
19:53:36 INFO   : Successfully done settir
19:53:36 INFO   : Processing command line
```

Running C Applications

- ❑ Power on the ZYBO
- ❑ Run the Tera Term
  - From the Window desktop, double-click the **‘Tera Term’** icon.
  - Select **‘Serial’ > ‘COM(x): USB Serial Port(COM(x))’** and then click **‘*OK*’**




Tera Term: New connection

○ TCP/IP Host: myhost.example.com
☑ History
Service: ○ Telnet TCP port#: 22
● SSH SSH version: SSH2
○ Other Protocol: UNSPEC

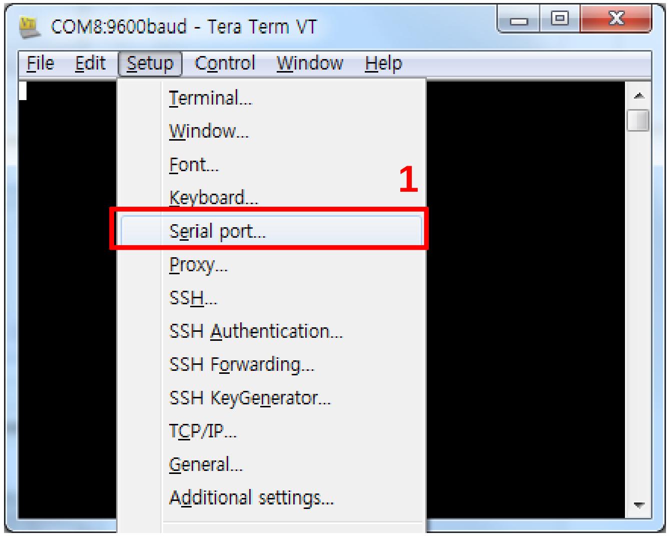
1 ● Serial Port: COM8: USB Serial Port(COM8)
COM1: 통신 포트(COM1)
2 COM8: USB Serial Port(COM8)

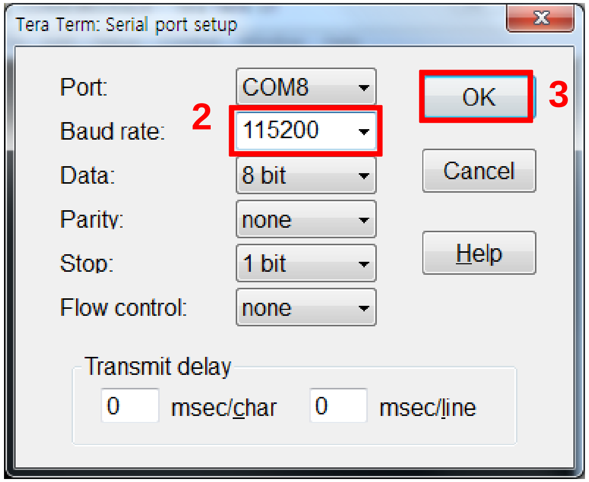
3 OK

Running C Applications

❑ Set up serial port

- Open the ***‘Setup’*** menu and then click ***‘Serial port’***
- Set ***‘Baud Rate’*** to ***115200*** and then click ***‘OK’***




Tera Term: Serial port setup

Port: COM8

2 Baud rate: 115200

Data: 8 bit

Parity: none

Stop: 1 bit

Flow control: none

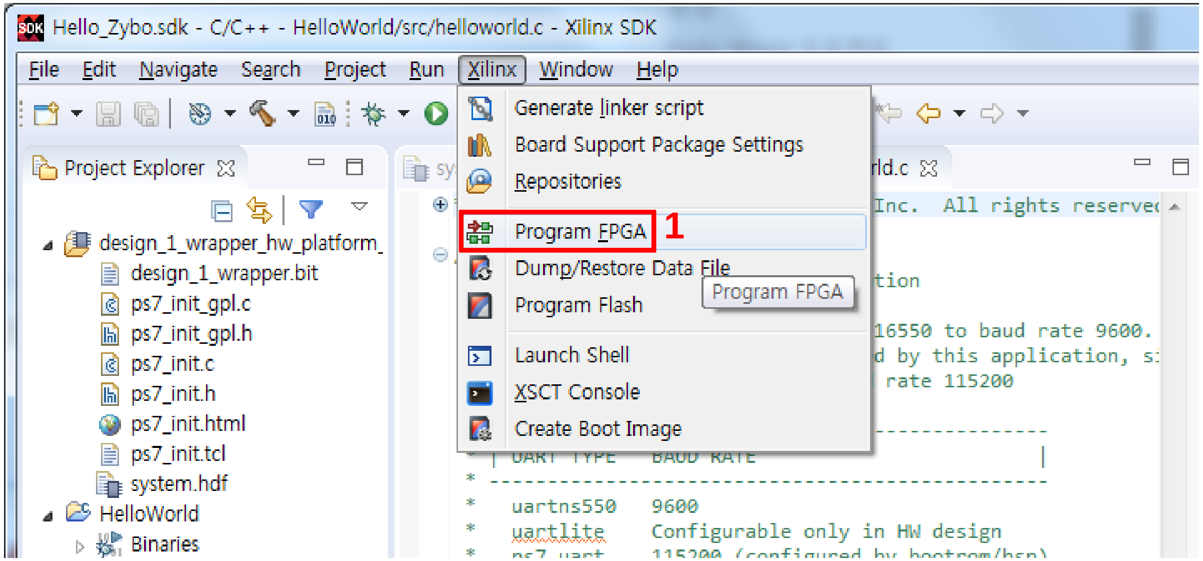
Transmit delay

0 msec/char 0 msec/line

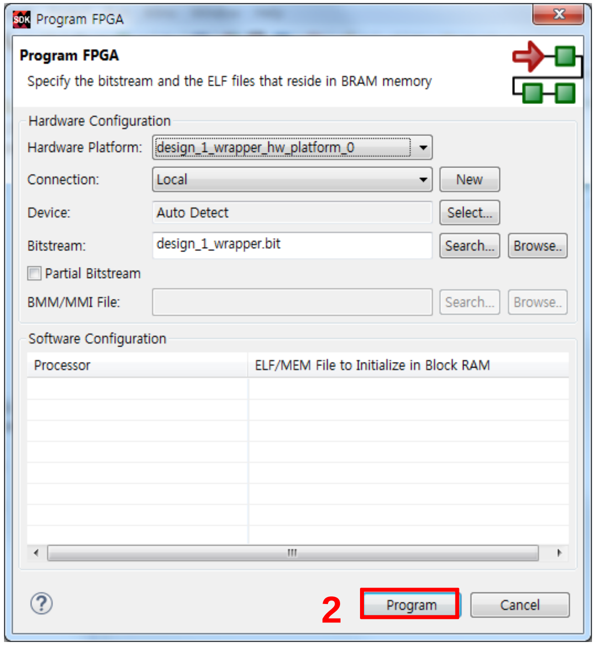
Running C Applications

❑ Program FPGA

- Open the ‘***Xilinx***’ menu and then click ‘***Program FPGA***’
- Click ‘***Program***’



```
Inc.  All rights reserved
tion
16550 to baud rate 9600.
d by this application, s
 rate 115200
---------------
* | UART TYPE    BAUD RATE                            |
* --------------------------------------------------------
*   uartns550    9600
*   uartlite     Configurable only in HW design
*   ps7_uart     115200 (configured by bootrom/bsp)
```



Program FPGA

Specify the bitstream and the ELF files that reside in BRAM memory

Hardware Configuration

Hardware Platform: design_1_wrapper_hw_platform_0

Connection: Local

Device: Auto Detect

Bitstream: design_1_wrapper.bit

□ Partial Bitstream

BMM/MMI File:

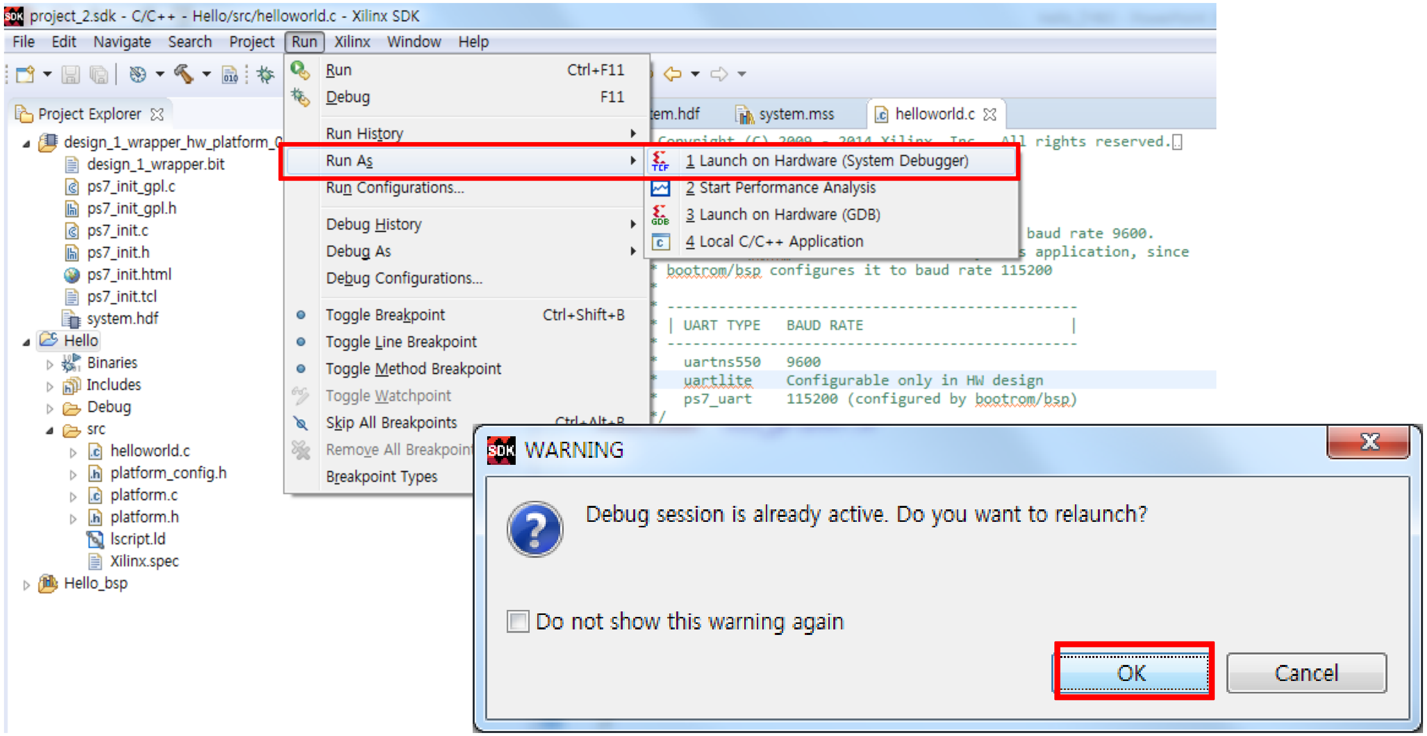
Software Configuration

Processor	ELF/MEM File to Initialize in Block RAM

Running C Applications

❑ Run the application

- Open the ***'Run' > 'Run As'*** menu and then click ***'Launch on Hardware (System Debugger)'***



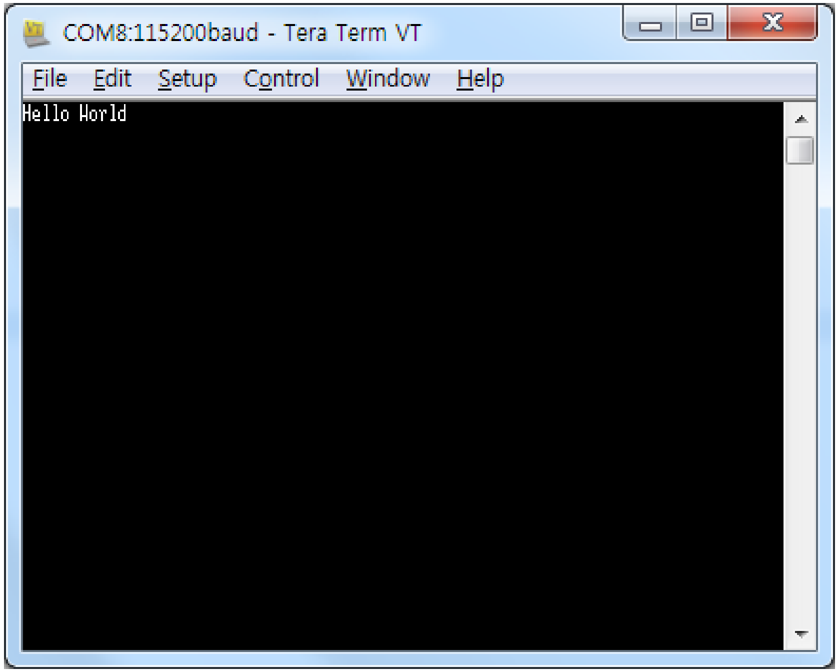
```
Copyright (C) 2009 - 2014 Xilinx, Inc.  All rights reserved.
...
                                     baud rate 9600.
                                     s application, since
 * bootrom/bsp configures it to baud rate 115200
 *
 * ------------------------------------------------
 * | UART TYPE   BAUD RATE                        |
 * ------------------------------------------------
 *   uartns550   9600
 *   uartlite    Configurable only in HW design
 *   ps7_uart    115200 (configured by bootrom/bsp)
 */
```

Debug session is already active. Do you want to relaunch?

☐ Do not show this warning again

Running C Applications

- ❑ Run the application (cont'd)
  - Check the output on ***'Tera Term'***
    - ✓ You should see 'Hello World' as shown below.

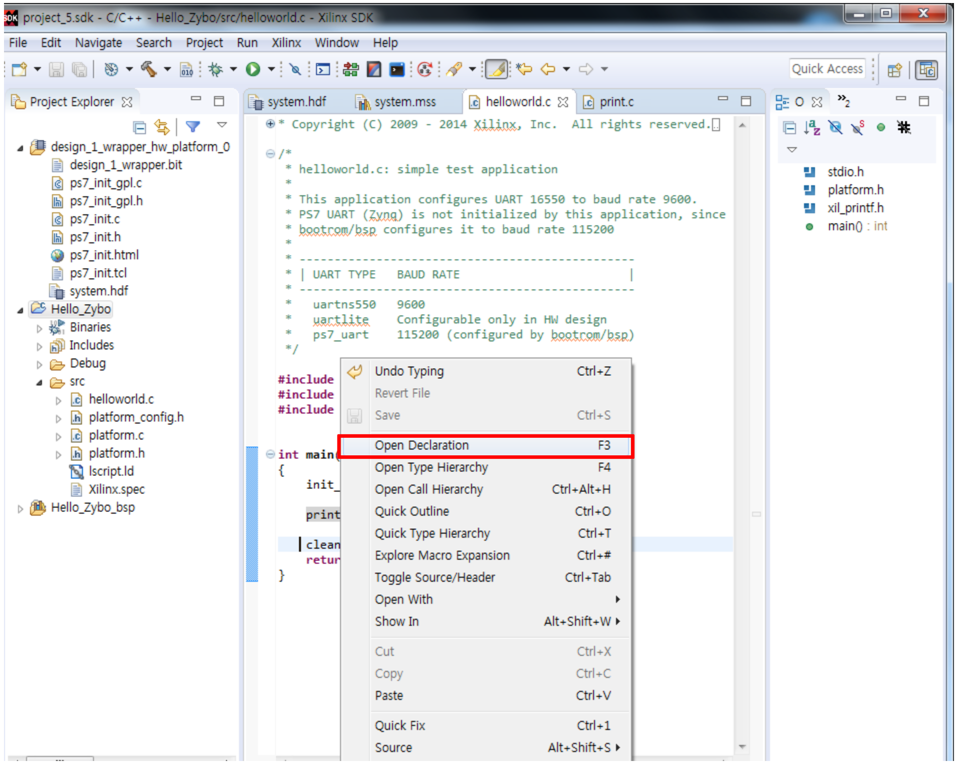


```
Hello World
```

Running C Applications

❑ Modify the application

- Right-click ‘***print’*** in ‘***helloworld.c’*** and then click ‘***Open Declaration’***



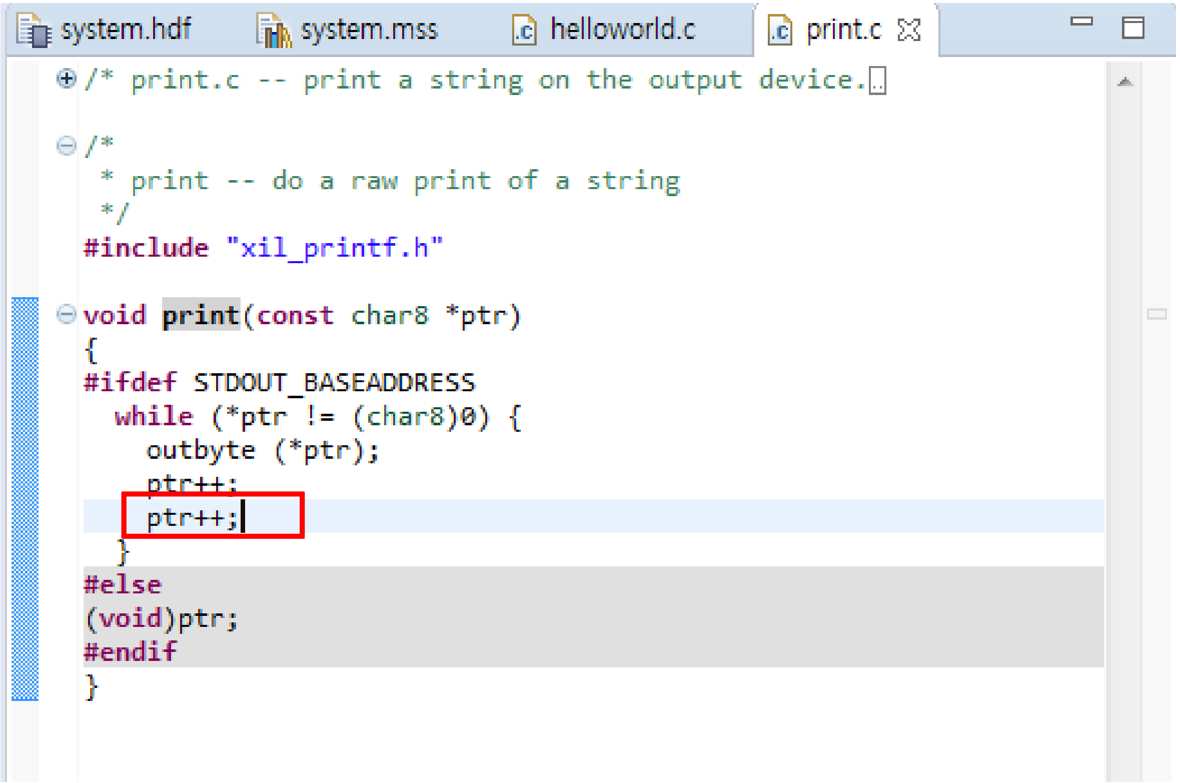
project_5.sdk - C/C++ - Hello_Zybo/src/helloworld.c - Xilinx SDK

```
* Copyright (C) 2009 - 2014 Xilinx, Inc.  All rights reserved.

/*
 * helloworld.c: simple test application
 *
 * This application configures UART 16550 to baud rate 9600.
 * PS7 UART (Zynq) is not initialized by this application, since
 * bootrom/bsp configures it to baud rate 115200
 *
 * ------------------------------------------------
 * | UART TYPE   BAUD RATE                        |
 * ------------------------------------------------
 *   uartns550   9600
 *   uartlite    Configurable only in HW design
 *   ps7_uart    115200 (configured by bootrom/bsp)
 */

#include
#include
#include

int main(
{
    init_

    print

    clean
    retur
}
```

Running C Applications

❑ Modify the application (cont'd)

- Add '***ptr++;***' at the end of the function.

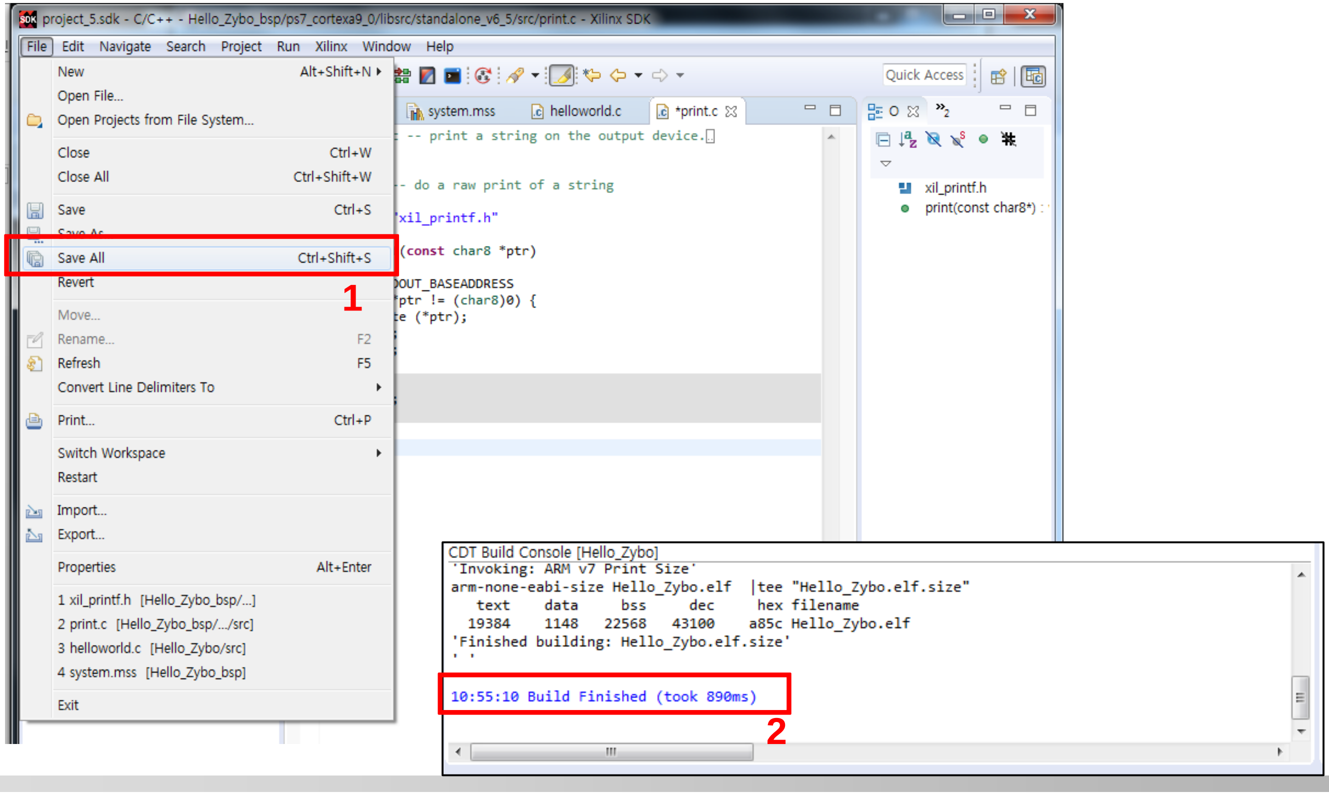


```
/* print.c -- print a string on the output device.

/*
 * print -- do a raw print of a string
 */
#include "xil_printf.h"

void print(const char8 *ptr)
{
#ifdef STDOUT_BASEADDRESS
  while (*ptr != (char8)0) {
    outbyte (*ptr);
    ptr++;
    ptr++;
  }
#else
(void)ptr;
#endif
}
```

Running C Applications

❑ Modify the application (cont’d)

- Click ***‘File > Save All’***
- Check ***‘Build finished’*** on the ***‘Build Console’***



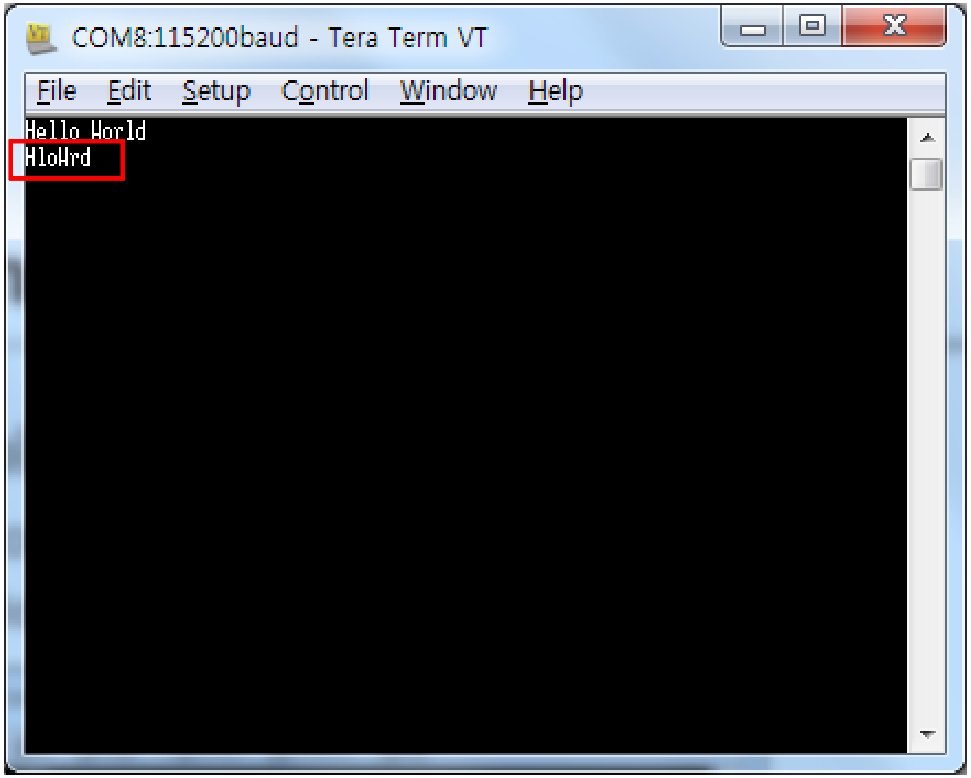
```
-- print a string on the output device.

- do a raw print of a string

'xil_printf.h"

(const char8 *ptr)

OUT_BASEADDRESS
*ptr != (char8)0) {
e (*ptr);
```

```
CDT Build Console [Hello_Zybo]
'Invoking: ARM v7 Print Size'
arm-none-eabi-size Hello_Zybo.elf  |tee "Hello_Zybo.elf.size"
   text    data     bss     dec     hex filename
  19384    1148   22568   43100    a85c Hello_Zybo.elf
'Finished building: Hello_Zybo.elf.size'
' '

10:55:10 Build Finished (took 890ms)
```

Running C Applications

❑ Run the application

- Follow pp. 34~35 of this lab workbook


COM8:115200baud - Tera Term VT
File Edit Setup Control Window Help

```
Hello World
HloWrd
```

Appendix

❑ Rerun the application

- When you need to rerun the existing application after shutting down Vivado/SDK, you don’t have to repeat all the steps shown in this lab workbook.
- You should simply follow p. 25 and pp. 30~38 of this lab workbook