

Getting Started with Zynq

Innova Lee(이상훈)
gcccompil3r@gmail.com



Overview

이 가이드는 Zybo Board 용 Vivado IP Integrator 를 사용하여 HW Design 을 만드는 과정을 단계별로 설명한다.
과정을 단계별로 설명한다.

이 튜토리얼의 마지막에는 아래와 같은 내용이 포함된다:

- * 보드상의 LED 및 스위치를 통합한 간단한 HW Design 을 만든다.
- * 이전 단계에서 보여준 HW Design 을 사용하여 보드 상의 LED 를 묶고 Xilinx Vivado SDK 에 C 프로젝트를 만든다.

Prerequisites

* Hardware

UART 통신 및 JTAG 프로그래밍을 위한 Digilent 의 Zybo 개발 보드 및 micro USB 케이블

* Software

SDK 패키지가 포함된 Xilinx Vivado 2015.X

* Board Support Files

- Zybo Support Files

- 1) 이 파일은 보드의 GPIO 인터페이스를 설명하고 초기 Design 설정에서 Board 를 쉽게 선택하고 Block Design 에 GPIO IP Block 을 추가한다.
- 2) Vivado 2015.X Board Support Files 를 설치하는 방법에 대한 위키 가이드 'Vivado Board Files for Digilent 7-Series FPGA Boards' 를 따르라.
<https://reference.digilentinc.com/learn/software/tutorials/vivado-board-files/start?redirect=1>

Tutorial

General Design Flow

I. Vivado

- Vivado 를 열고 Zybo Board 를 선택한다.
- 새로운 Vivado Project 를 생성한다.
- 새로운 프로젝트에 Empty Block Design Workspace 를 생성한다.
- IP Integrator 를 사용하여 필요한 IP Block 을 추가하고 HW Design 을 구축한다.
- Block Design 을 검증하고 저장한다.
- HDL System Wrapper 를 작성한다.
- Design 합성하고 구현한다.
- 비트 파일을 생성한다.
- 생성 된 비트 스트림 파일을 SDK 도구에 포함하여 HW Design Export 하기
- SDK 시작

이제 HW Design 이 SDK 도구로 Export 된다.

Vivado 에서 SDK 로의 Hand-Off 는 Vivado 를 통해 내부적으로 수행된다.

우리는 SDK 를 사용하여 Vivado 에서 HW Design 정보를 가져와서

Custom Board Interface Data 및 FPGA HW 구성을 사용할 SW 응용 프로그램을 작성한다.

II. SDK

- 새로운 응용 프로그램 프로젝트를 만들고 기본 Hello World 템플릿을 선택한다.
- FPGA 프로그래밍 및 응용 프로그램 실행

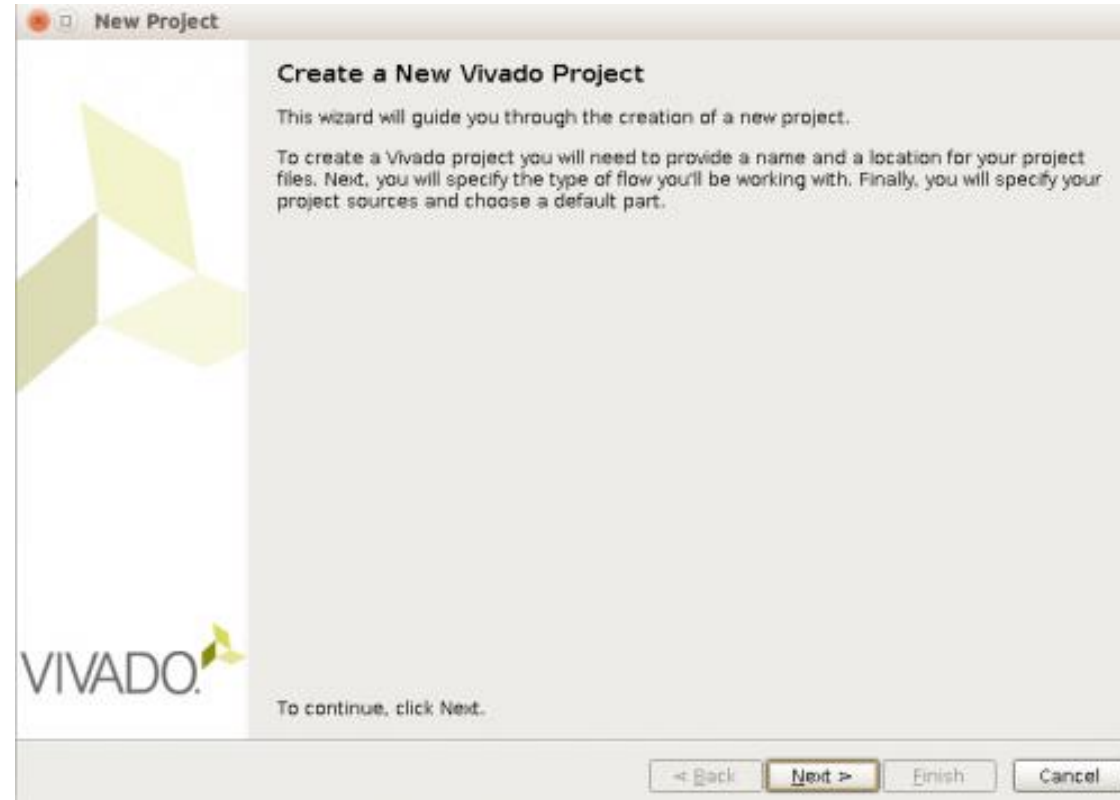
1. Creating a New Project

Vivado 를 처음 실행하면 새 프로젝트를 만들거나 최근 프로젝트를 열 수 있는 기본 시작 창이 보인다.

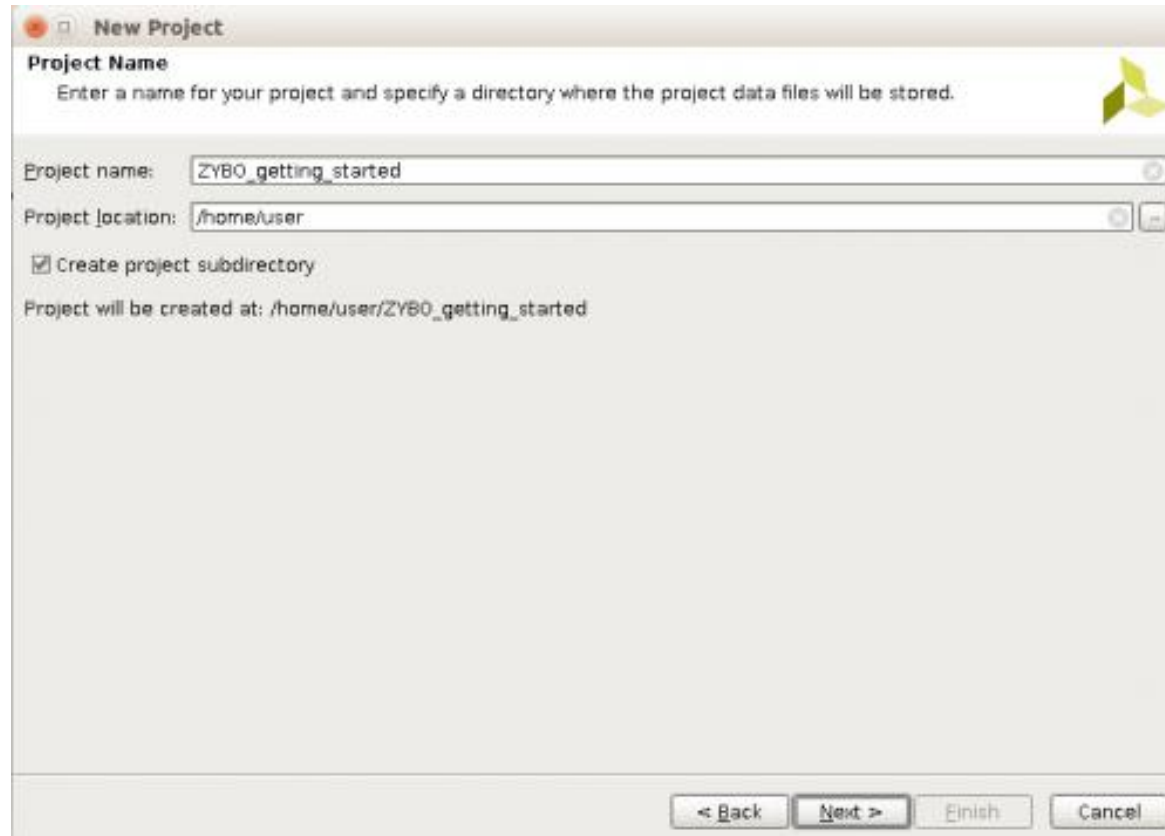
1.1) Create New Project 를 클릭한다.



- 1.2) 프로젝트 생성 마법사가 나타난다.
Next 를 클릭한다.



1.3) 프로젝트 이름과 위치를 입력하고 Next 를 클릭한다.



The image shows a 'New Project' dialog box with a title bar containing a red close button and the text 'New Project'. The main area has a header 'Project Name' followed by the instruction 'Enter a name for your project and specify a directory where the project data files will be stored.' and a small yellow logo. Below this are two text input fields: 'Project name:' containing 'ZYBO_getting_started' and 'Project location:' containing '/home/user'. A checkbox labeled 'Create project subdirectory' is checked. Below the checkbox, it says 'Project will be created at: /home/user/ZYBO_getting_started'. At the bottom, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

New Project

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

Project name: ZYBO_getting_started

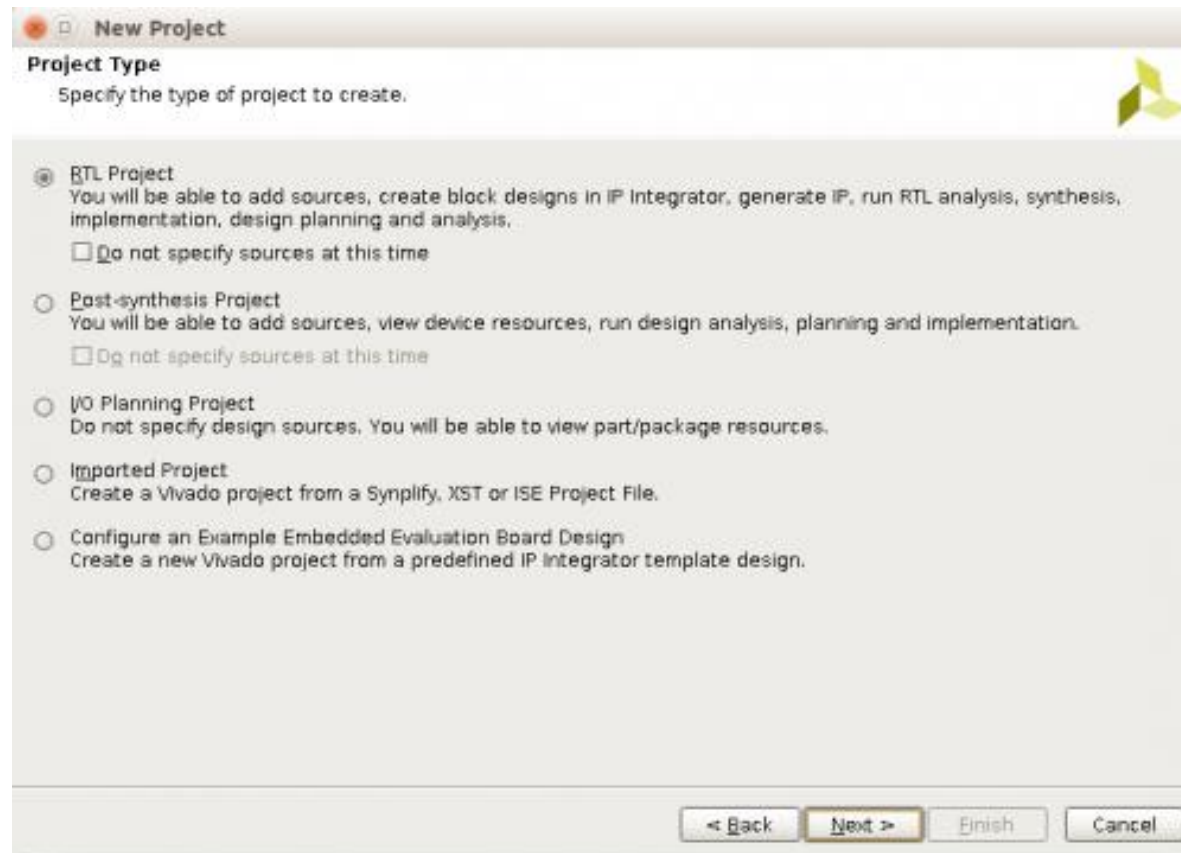
Project location: /home/user

☒ Create project subdirectory

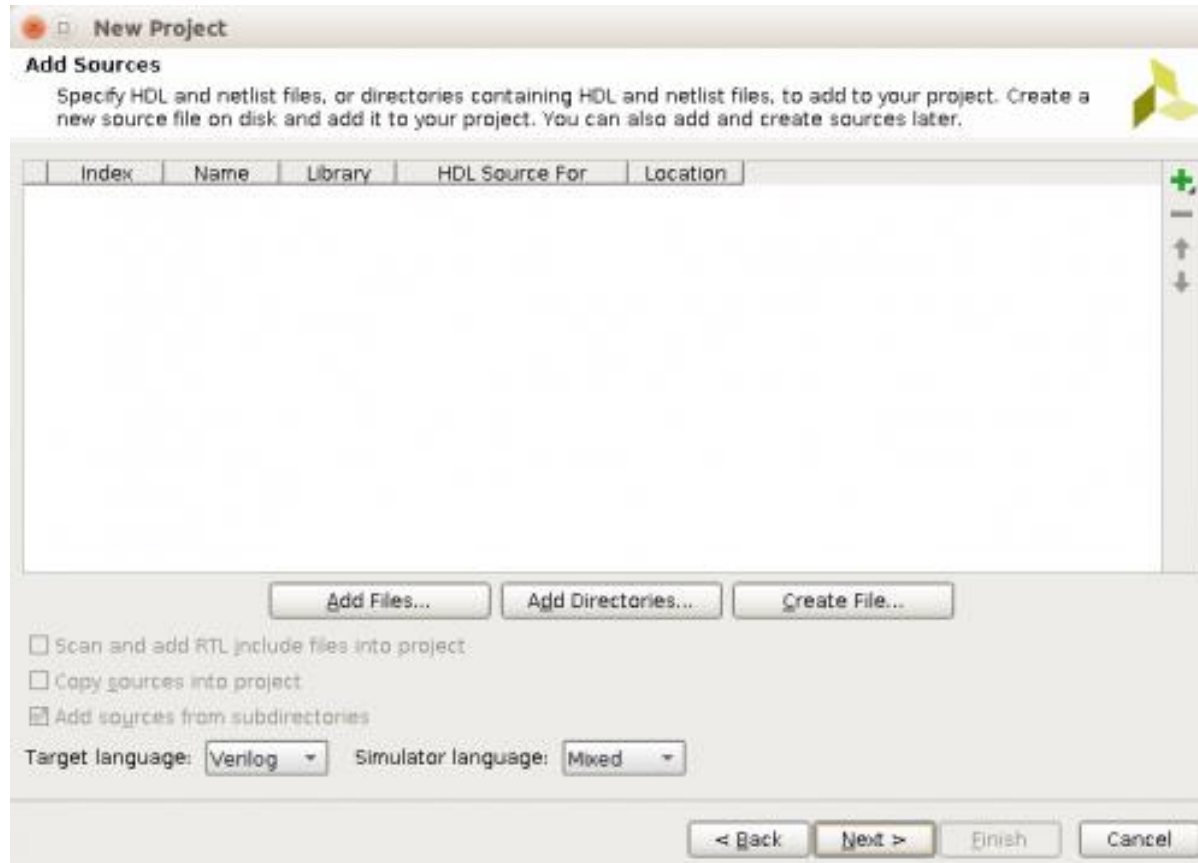
Project will be created at: /home/user/ZYBO_getting_started

< Back Next > Finish Cancel

1.4) RTL Project 를 선택하고 Next 를 클릭한다.



- 1.5) 이 데모에서는 기존 소스, 기존 IP 또는 제약 조건을 사용하지 않는다.
다음과 같은 화면 세 개가 나오면 Next 를 클릭한다.



- 1.6) Board 를 선택하고 Zybo 보드 파일을 선택한다.
Next 를 클릭하고 Finish 를 클릭한다.

New Project

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

Select: ☐ Parts ☒ Boards

Filter

Vendor: All

Display Name: All

Board Rev: Latest

Reset All Filters

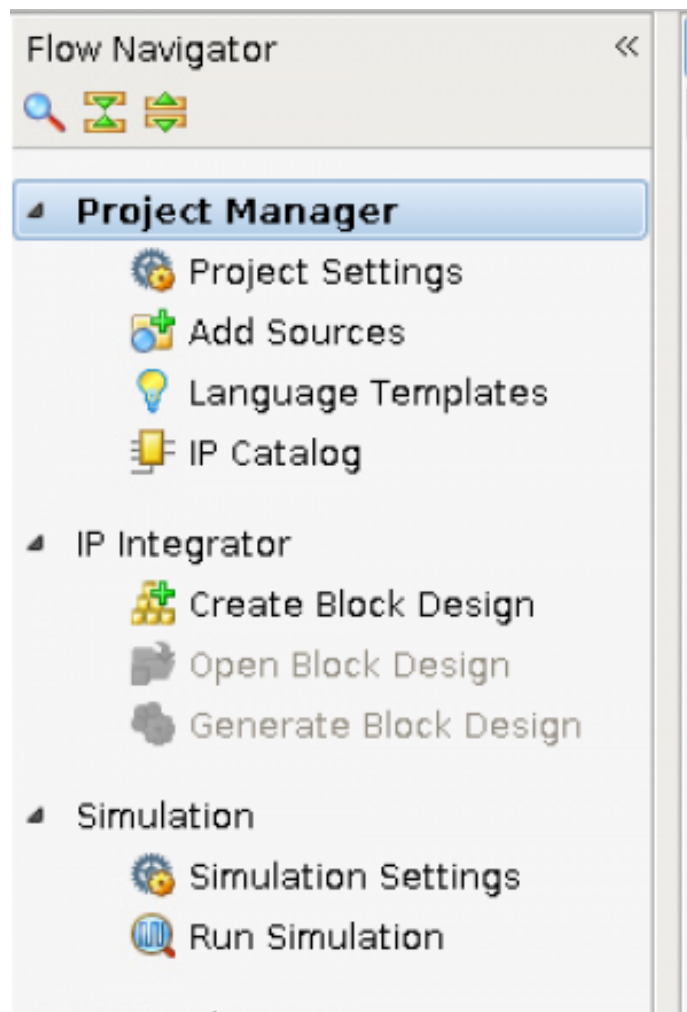
Search: Q-

Display Name	Vendor	Board Rev	Part	I/O Pin Count	File \
<input checked="" type="checkbox"/> Zybo	digilentinc.com	b	xc7z010clg400-1	400	1.0
<input checked="" type="checkbox"/> MicroZed Board	em.avnet.com	f	xc7z010clg400-1	400	1.1
<input checked="" type="checkbox"/> ZedBoard Zynq Evaluation and Development Kit	em.avnet.com	d	xc7z020clg484-1	484	1.2
<input checked="" type="checkbox"/> Artix-7 AC701 Evaluation Platform	xilinx.com	1.1	xc7a200tfg676-2	676	1.1
<input checked="" type="checkbox"/> Kintex-7 KC705 Evaluation Platform	xilinx.com	1.1	xc7k325tfg900-2	900	1.1
<input checked="" type="checkbox"/> Virtex-7 VC707 Evaluation Platform	xilinx.com	1.1	xc7vx485tfg1761-2	1,761	1.1
<input checked="" type="checkbox"/> Virtex-7 VC709 Evaluation Platform	xilinx.com	1.0	xc7vx690tfg1761-2	1,761	1.5
<input checked="" type="checkbox"/> ZYNQ-7 ZC702 Evaluation Board	xilinx.com	1.0	xc7z020clg484-1	484	1.1
<input checked="" type="checkbox"/> ZYNQ-7 ZC706 Evaluation Board	xilinx.com	1.1	xc7z045ffg900-2	900	1.1

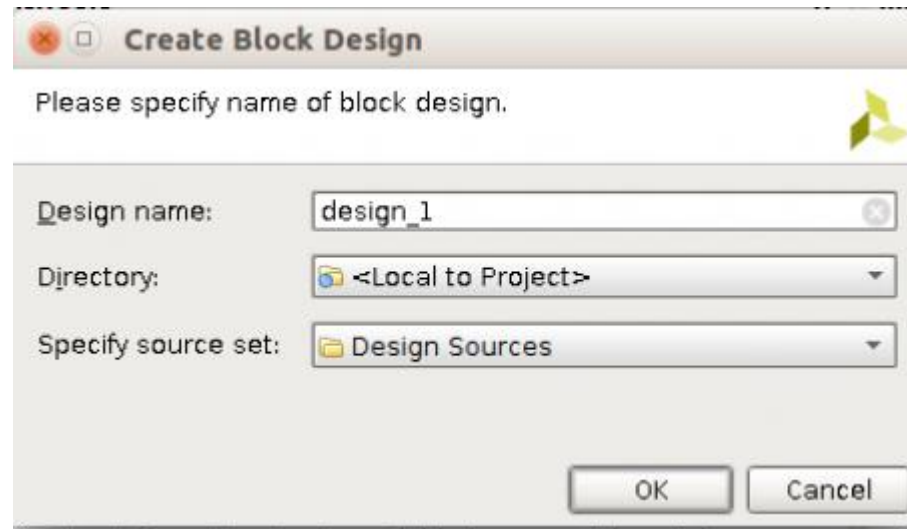
< Back Next > Finish Cancel

2. Creating a New Block Design

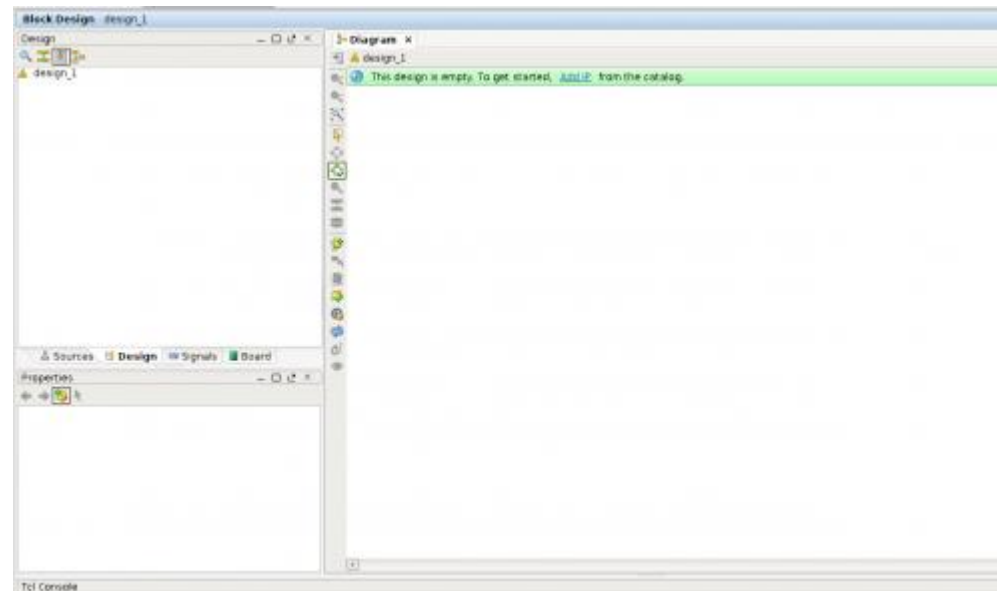
2.1) 일련의 절차가 완료되면 Flow Navigator 에서 Create Block Design 을 클릭한다.



2.2) OK 를 클릭한다.



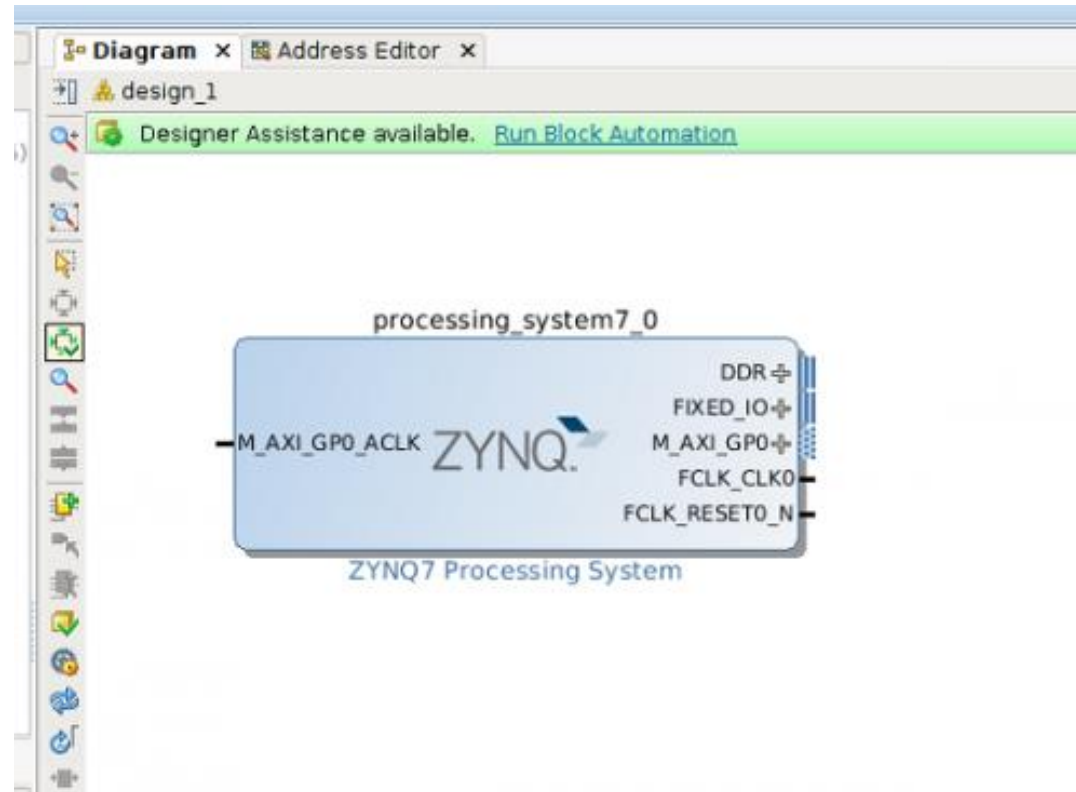
2.3) Blank Block Design 이 열린다.



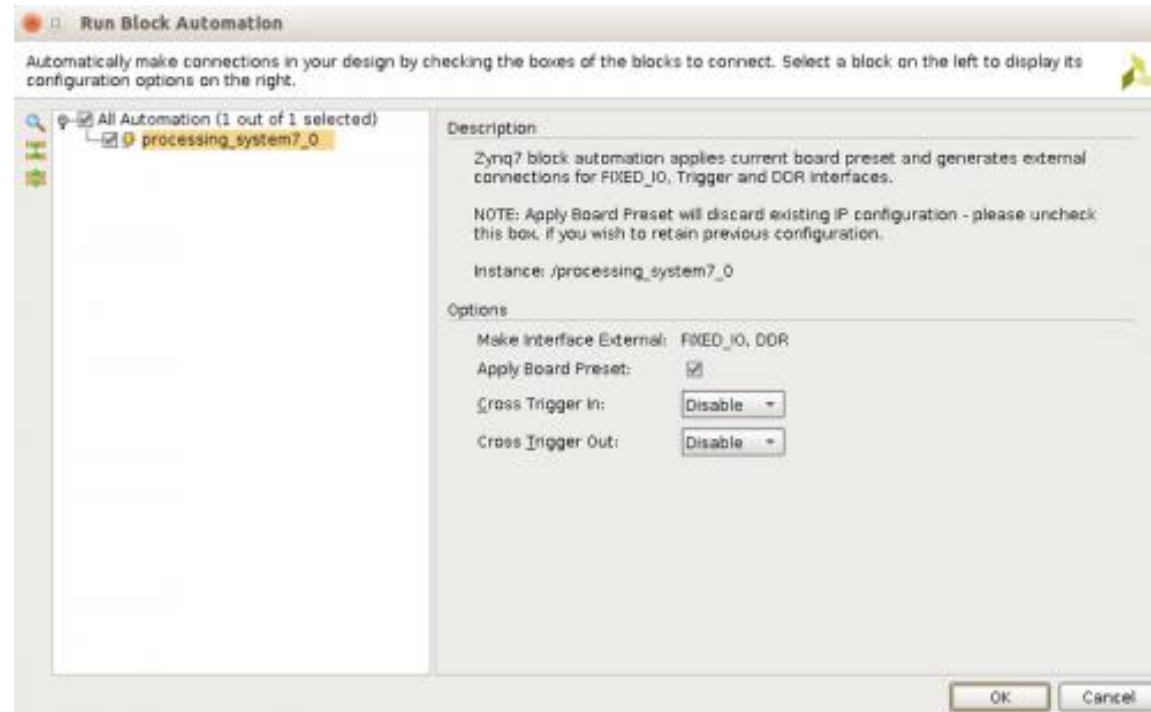
3. Add the Zynq IP & GPIO Blocks

3.1) Add IP 버튼을 클릭하고 ZYNQ 를 검색한다.

ZYNQ7 Processing System 을 더블 클릭하여 bare Zynq Block 을 배치한다.



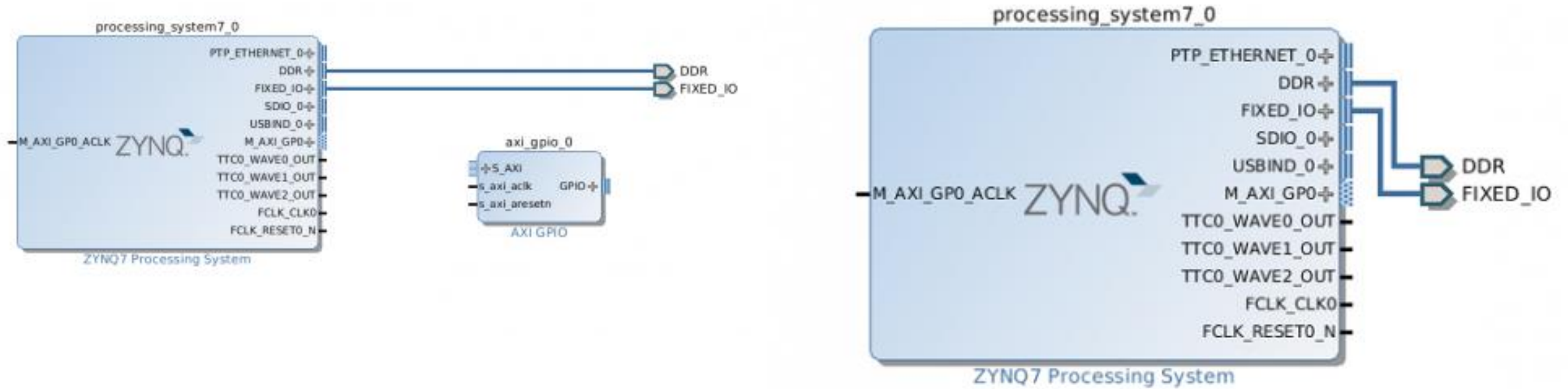
3.2) Run Block Automation 링크를 클릭한다.



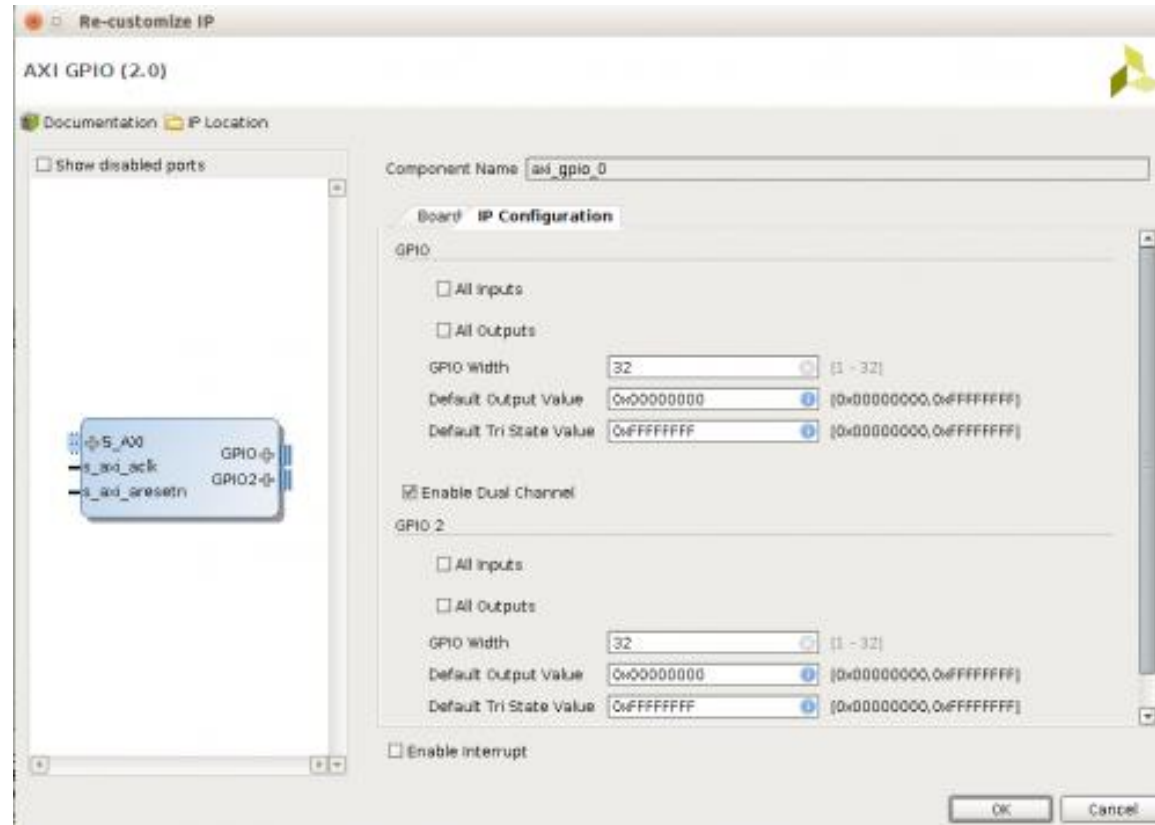
이제 Zynq Block 은 다음 페이지의 아래 그림과 같이 보인다.

3.3) Add IP Icon 을 다시 클릭한다.

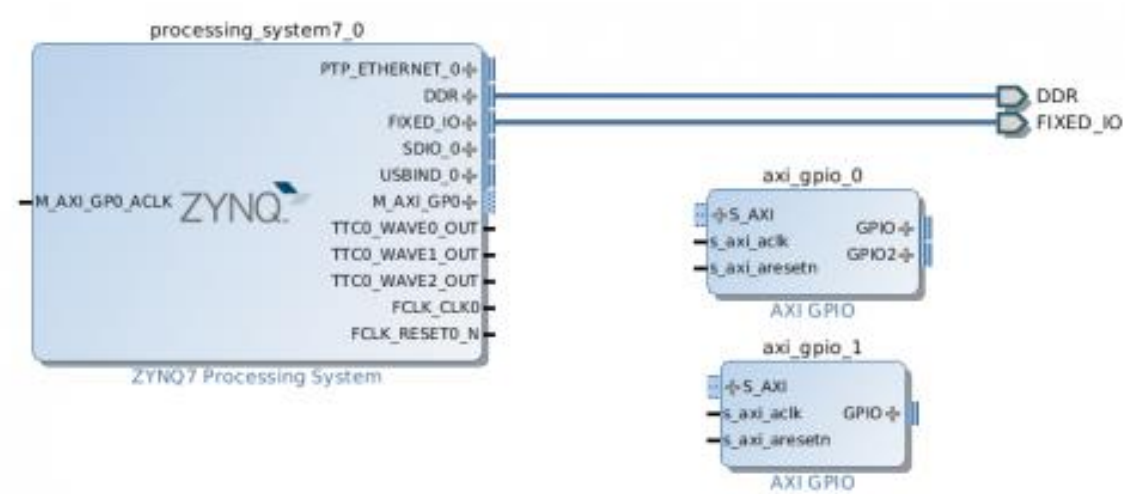
이번에는 "gpio" 를 검색하고 AXI GPIO 코어를 추가한다.



- 3.4) 방금 추가된 새로운 axi_gpio_0 코어를 더블 클릭하여 커스터마이징 창을 연다.
IP Configuration 탭에서 Enable Dual Channel 박스를 선택한다.
OK 를 클릭한다.

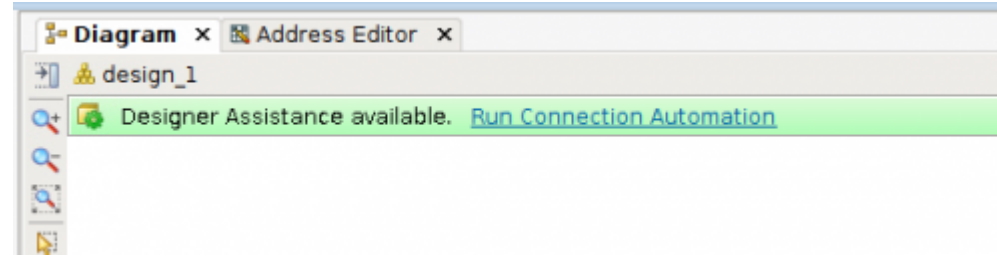


3.5) 단계 3.3 을 반복하여 다른 GPIO 코어를 추가하지만 Dual Channel 을 활성화하지 않도록 한다.

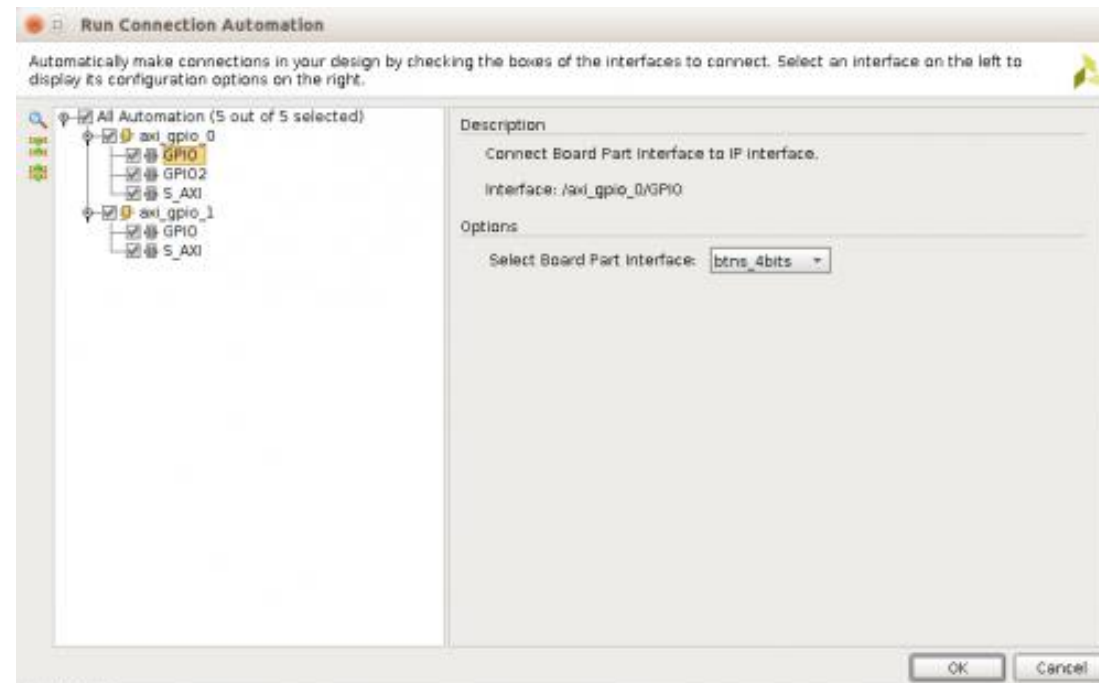


4. Run the Connection Automation Tool

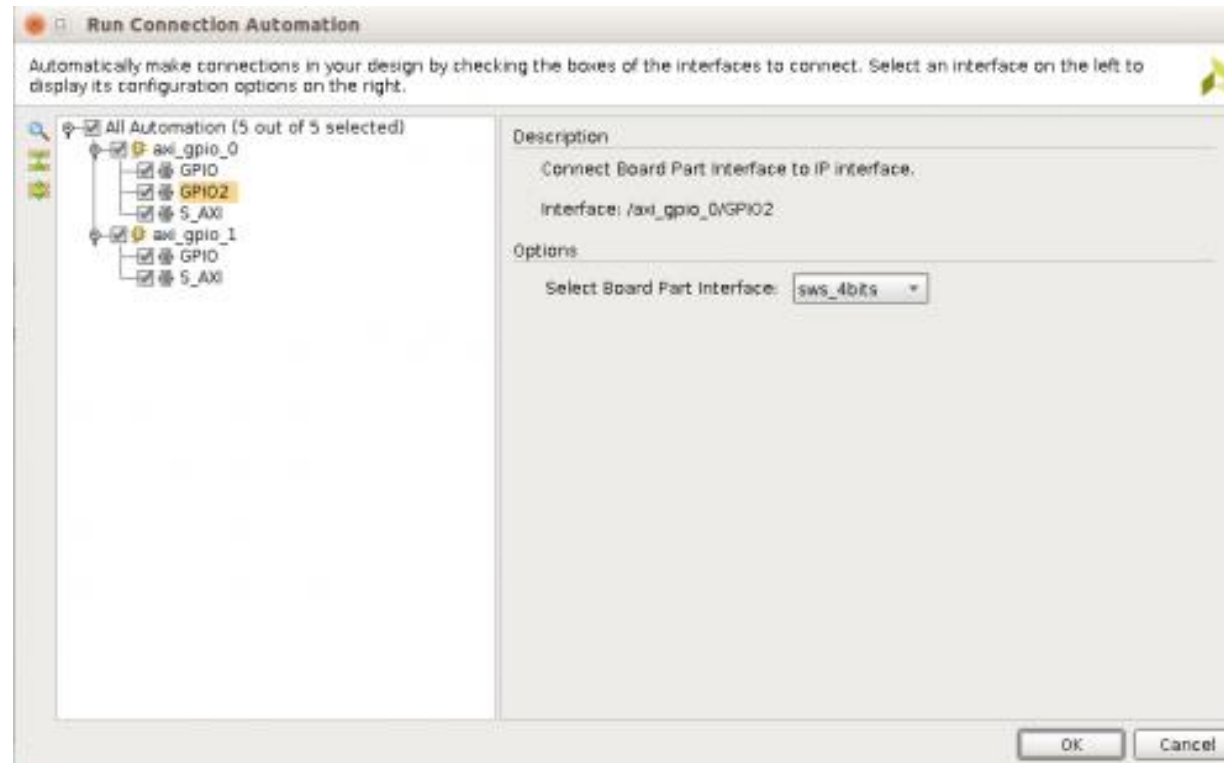
- 4.1) Connection Automation Tool 은 데모에 필요한 Logic Block 을 추가한다.
파란색으로 강조된 Run Connection Automation 을 선택한다.



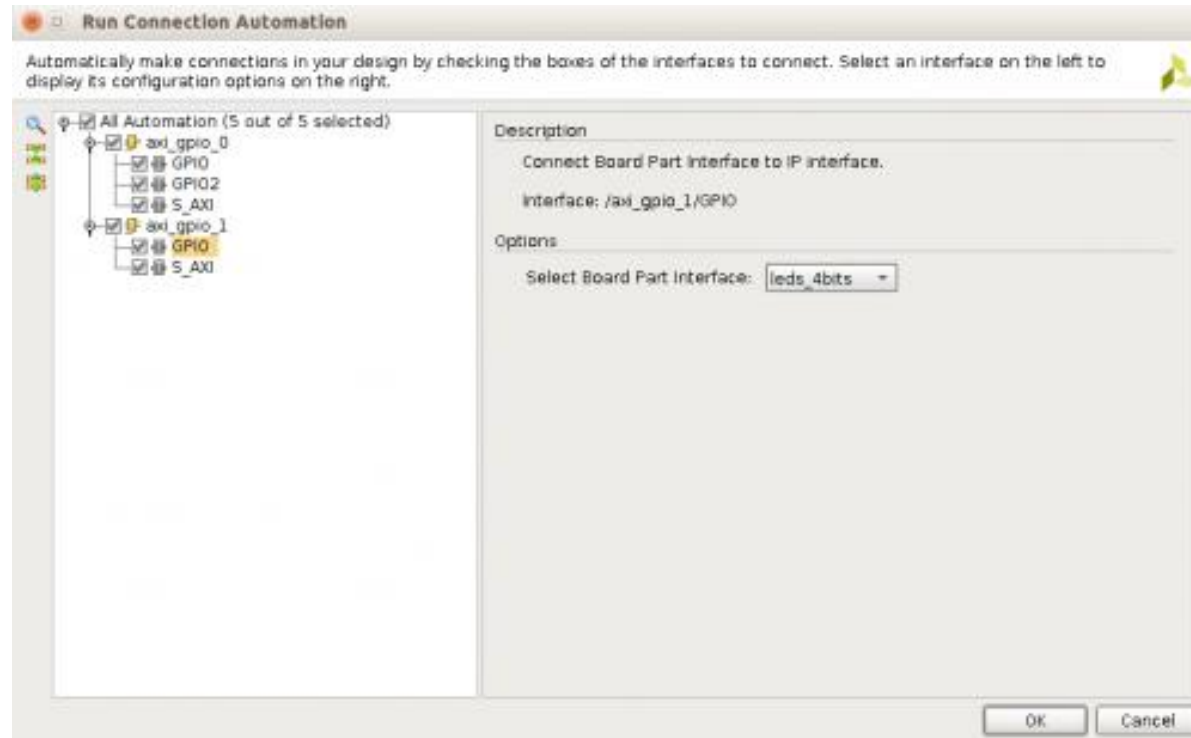
- 4.2) All Automation 확인란을 체크한다.
axi_gpio_0 에서 GPIO 를 선택하고 Board Part Interface 드롭 다운 상자에서 btns_4bits 를 선택한다.



4.3) axi_gpio_0 에서 GPIO2 를 선택하고 drop down 상자에서 swts_4bits 를 선택한다.

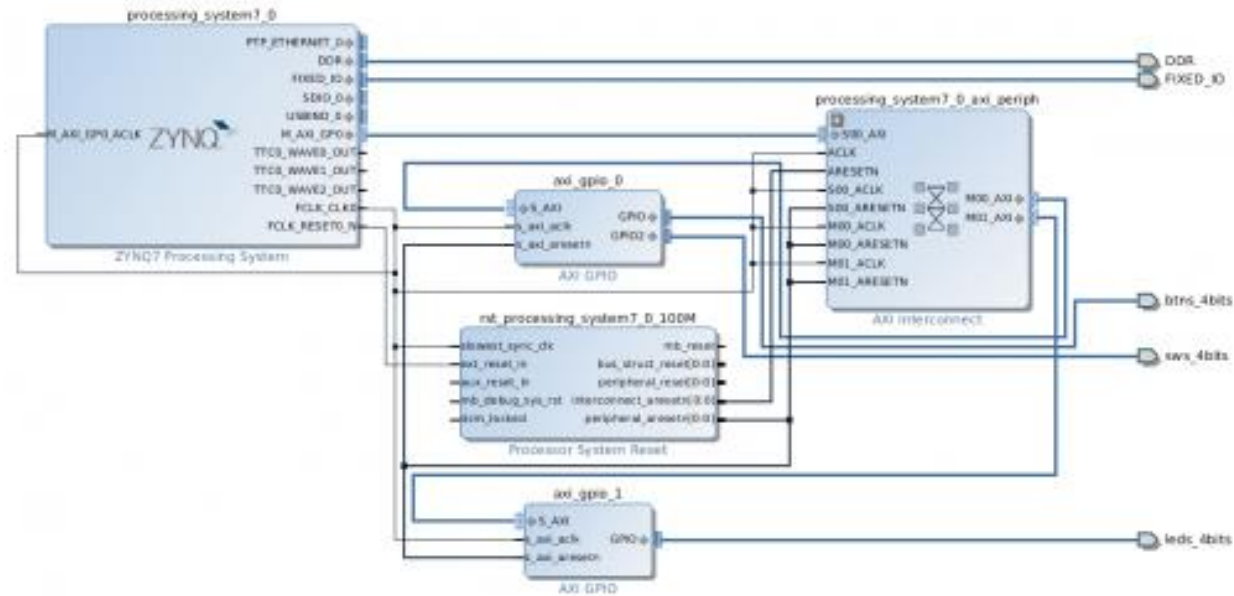


4.4) axi_gpio_1 에서 GPIO 를 선택하고 drop down 상자에서 leds_4bits 를 선택하고 OK 를 누른다.

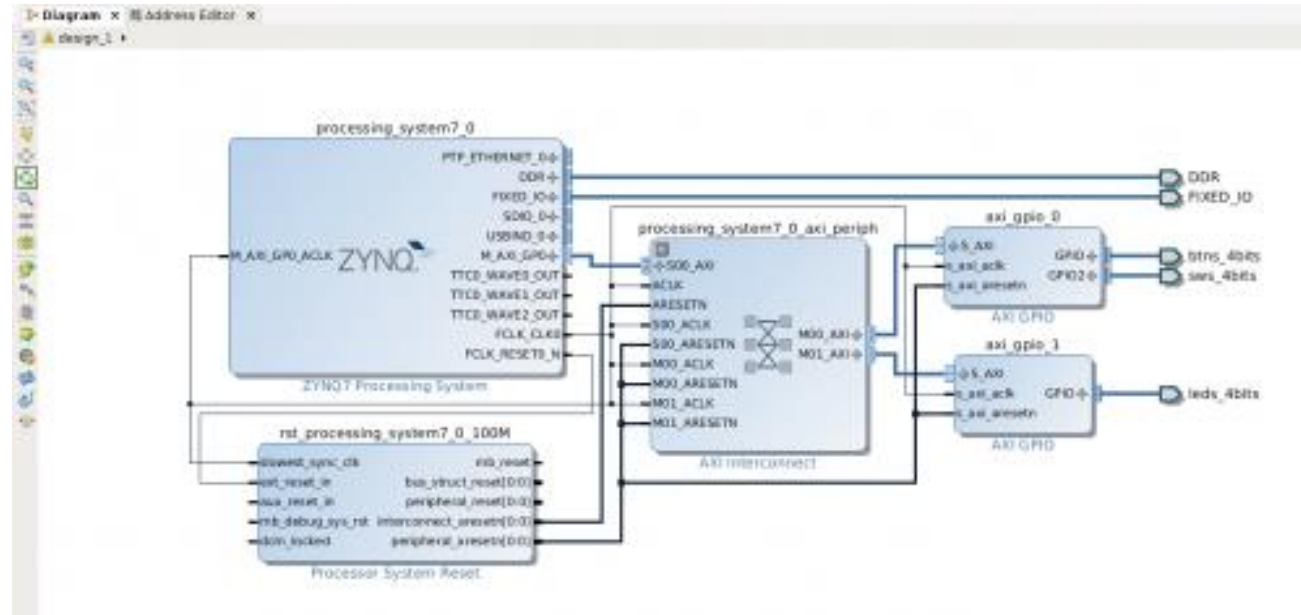


4.5) 이 프로세스는 아래를 추가한다:

- * AXI 상호 연결
- * Processor System Reset
- * 버튼, 스위치 및 LED 보드 부품



- 4.6) 다음으로 Block Design 을 정리한다.
Regenerate Layout 버튼을 클릭하여 Block Design 을 다시 정렬한다.



5. Generate HDL Wrapper and Validate Design

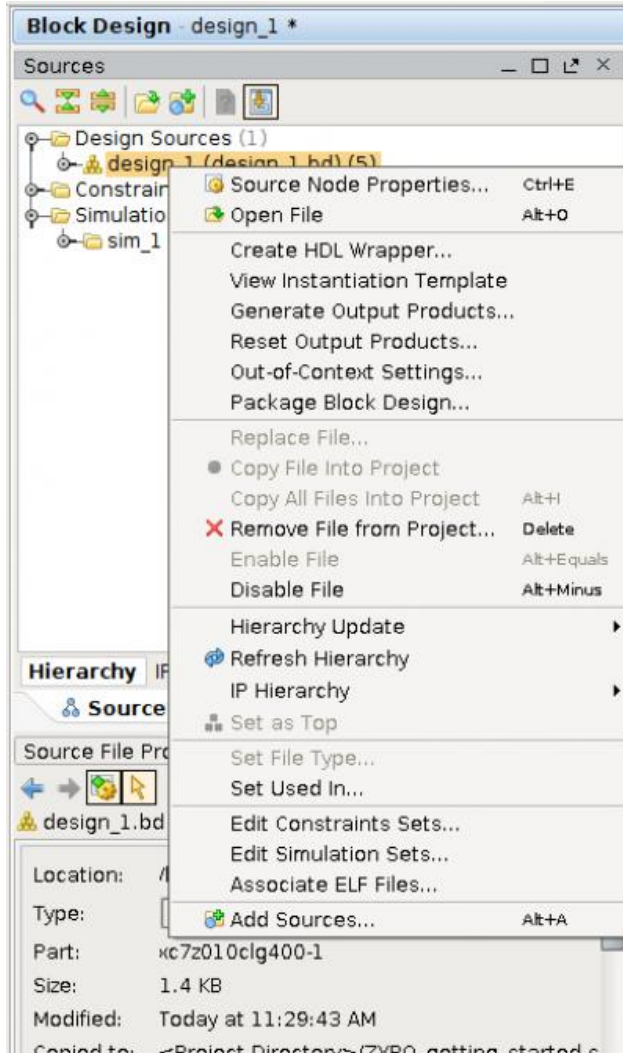
5.1) Validate Design 을 선택한다.

이렇게 하면 Design 및 Connection 오류를 검사한다.

5.2) Design Validation 단계가 끝나면 HDL System Wrapper 를 작성한다.

Block Design 창 아래의 Design Sources 탭에서 Block Diagram 파일을 우클릭한다.

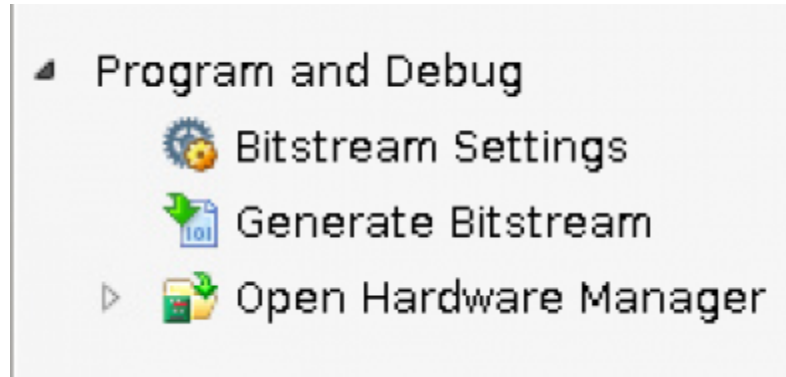
이것을 "design_1.bd" 로 표시하고 Create HDL Wrapper 를 선택한다.



이렇게 하면 VHDL 에 최상위 모듈이 생성되어 bitstream 을 생성 할 수 있다.

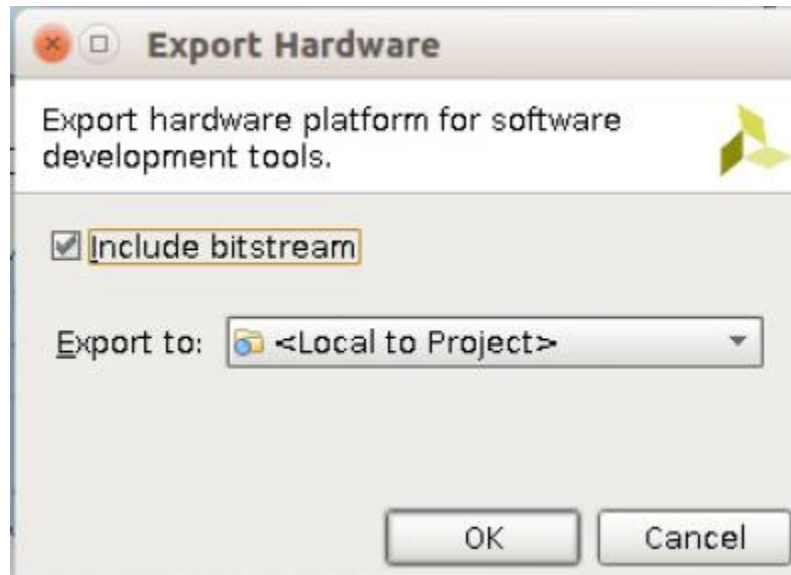
6. Generate the Bitstream

- 6.1) Flow Navigator 하단의 Generate Bitstream 을 클릭한다.
해당 절차가 완료 될 때까지 기다렸다가 OK 를 클릭한다.



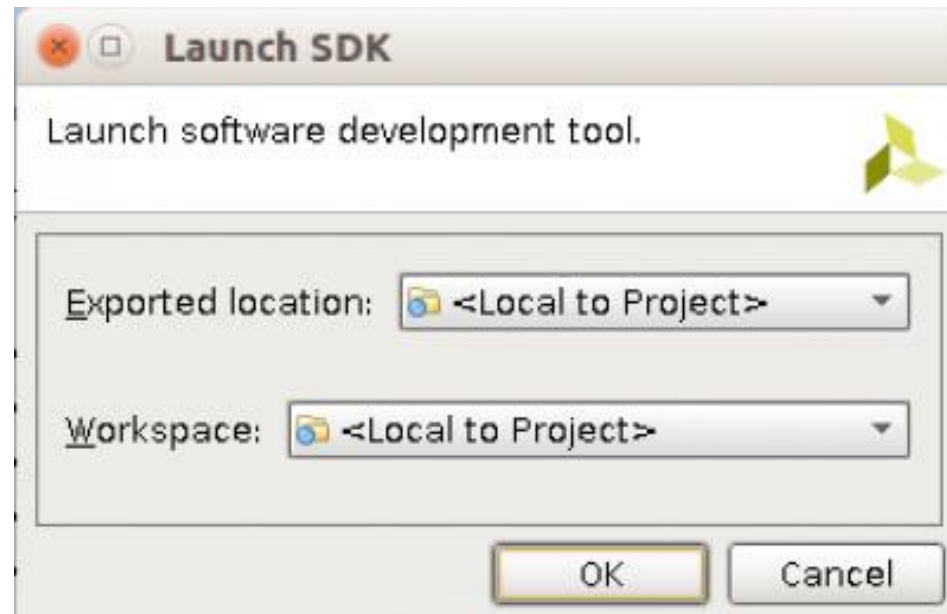
7. Export hardware files for SDK

- 7.1) file -> Export -> Export Hardware... 로 이동한다.
Include bitstream 확인란을 선택하고 OK 를 클릭한다.



8. Launch SDK

8.1) File -> Launch SDK 로 이동해서 OK 를 클릭한다.



9. Create a new Hello World Application Project in SDK

9.1) File -> New -> Application Project 로 이동한다.

New Project

Application Project
Create a managed make application project.

Project name:

☒ Use default location

Location:

Choose file system:

Target Hardware

Hardware Platform:

Processor:

Target Software

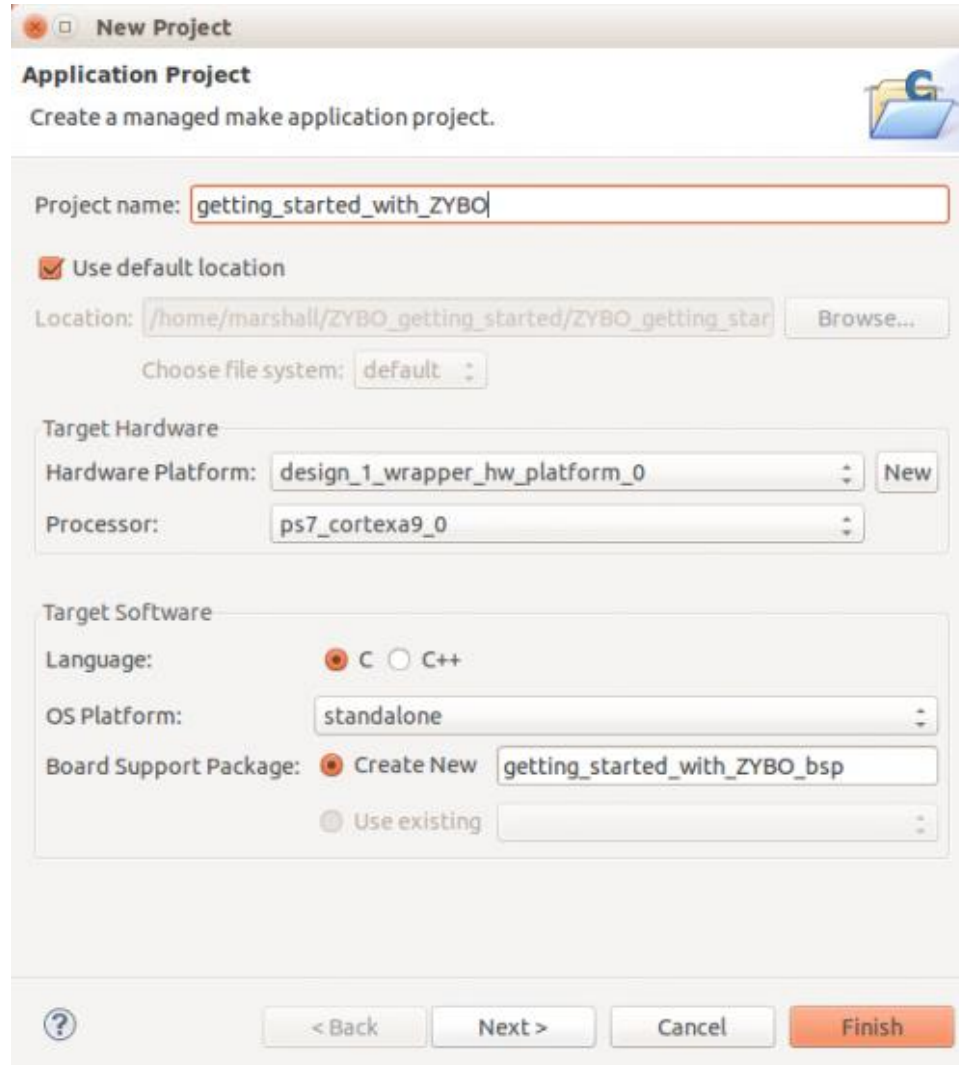
Language: ☒ C ☐ C++

OS Platform:

Board Support Package: ☒ Create New ☐ Use existing

9.2) 프로젝트 세부 정보 입력:

- * 프로젝트 이름: "getting_started_with_ZYBO"
- * HW 플랫폼: design_1_wrapper_hw_platform_0
- * 프로세서: ps7_cortexa9_0
- * 언어: C
- * OS 플랫폼: 독립 실행형
- * 보드 지원 패키지: 신규 작성(기본 이름 그대로)



The image shows a 'New Project' dialog box for an 'Application Project'. The title bar says 'New Project'. Below the title, it says 'Application Project' and 'Create a managed make application project.' with a folder icon. The 'Project name' field contains 'getting_started_with_ZYBO'. The 'Use default location' checkbox is checked. The 'Location' field shows '/home/marshall/ZYBO_getting_started/ZYBO_getting_star' with a 'Browse...' button. The 'Choose file system' dropdown is set to 'default'. Under 'Target Hardware', the 'Hardware Platform' is 'design_1_wrapper_hw_platform_0' and the 'Processor' is 'ps7_cortexa9_0', both with 'New' buttons. Under 'Target Software', the 'Language' is 'C' (selected) and 'C++'. The 'OS Platform' is 'standalone'. The 'Board Support Package' is 'Create New' (selected) with the name 'getting_started_with_ZYBO_bsp', and 'Use existing' is unselected.

New Project

Application Project
Create a managed make application project.

Project name: getting_started_with_ZYBO

☒ Use default location

Location: /home/marshall/ZYBO_getting_started/ZYBO_getting_star Browse...

Choose file system: default

Target Hardware

Hardware Platform: design_1_wrapper_hw_platform_0 New

Processor: ps7_cortexa9_0

Target Software

Language: ☒ C ☐ C++

OS Platform: standalone

Board Support Package: ☒ Create New getting_started_with_ZYBO_bsp
☐ Use existing

? < Back Next > Cancel Finish

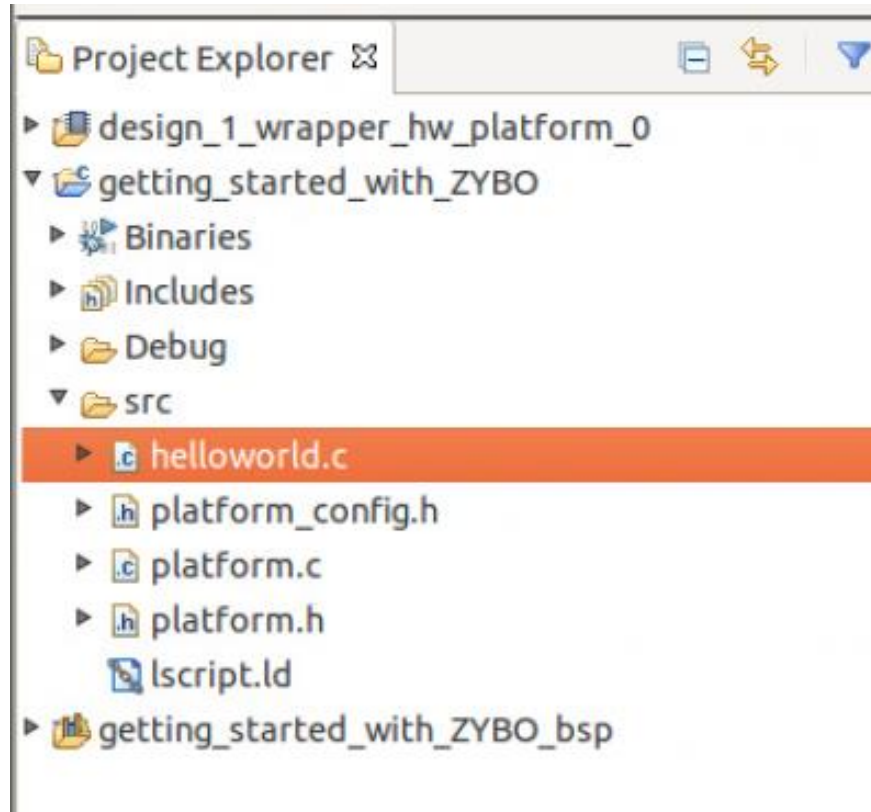
9.3) Hello World 데모는 이 데모의 좋은 출발점이다.

다음을 클릭하고 Hello World 를 선택한 다음 Finish 를 클릭한다.

이 절차는 두 개의 디렉토리를 프로젝트 탐색기에 추가한다.

9.4) getting_started_with_ZYBO 를 확장 한 다음 src 를 열고 "helloworld.c" 를 더블 클릭한다.

이것은 기본 Hello World C 코드다.



10. Creating Our Own Hello World

10.1) 아래 코드를 복사하여 helloworld.c 파일에 붙여 넣으라.

/*****

Getting Started Guide for Zybo

This demo displays the status of the switches on the LEDs and prints a message to the serial communication when a button is pressed.

Terminal Settings:

- Baud: 115200
- Data bits: 8
- Parity: no
- Stop bits: 1

1/6/14: Created by MarshallW

*****/

```
#include <stdio.h>
#include "platform.h"
#include <xgpio.h>
#include "xparameters.h"
#include "sleep.h"
```

```
int main()
```

```
{
```

```
    XGpio input, output;
    int button_data = 0;
    int switch_data = 0;
```

```
    XGpio_Initialize(&input, XPAR_AXI_GPIO_0_DEVICE_ID);
    XGpio_Initialize(&output, XPAR_AXI_GPIO_1_DEVICE_ID);
```

```
//initialize input XGpio variable
//initialize output XGpio variable
```

```
    XGpio_SetDataDirection(&input, 1, 0xF);
    XGpio_SetDataDirection(&input, 2, 0xF);
```

```
//set first channel tristate buffer to input
//set second channel tristate buffer to input
```

```
    XGpio_SetDataDirection(&output, 1, 0x0);
```

```
//set first channel tristate buffer to output
```

```
    init_platform();
```

```

while(1){
    switch_data = XGpio_DiscreteRead(&input, 2);          //get switch data

    XGpio_DiscreteWrite(&output, 1, switch_data);        //write switch data to the LEDs

    button_data = XGpio_DiscreteRead(&input, 1);          //get button data

    //print message dependent on whether one or more buttons are pressed
    if(button_data == 0b0000){} //do nothing

    else if(button_data == 0b0001)
        xil_printf("button 0 pressed\nWr");

    else if(button_data == 0b0010)
        xil_printf("button 1 pressed\nWr");

    else if(button_data == 0b0100)
        xil_printf("button 2 pressed\nWr");

    else if(button_data == 0b1000)
        xil_printf("button 3 pressed\nWr");

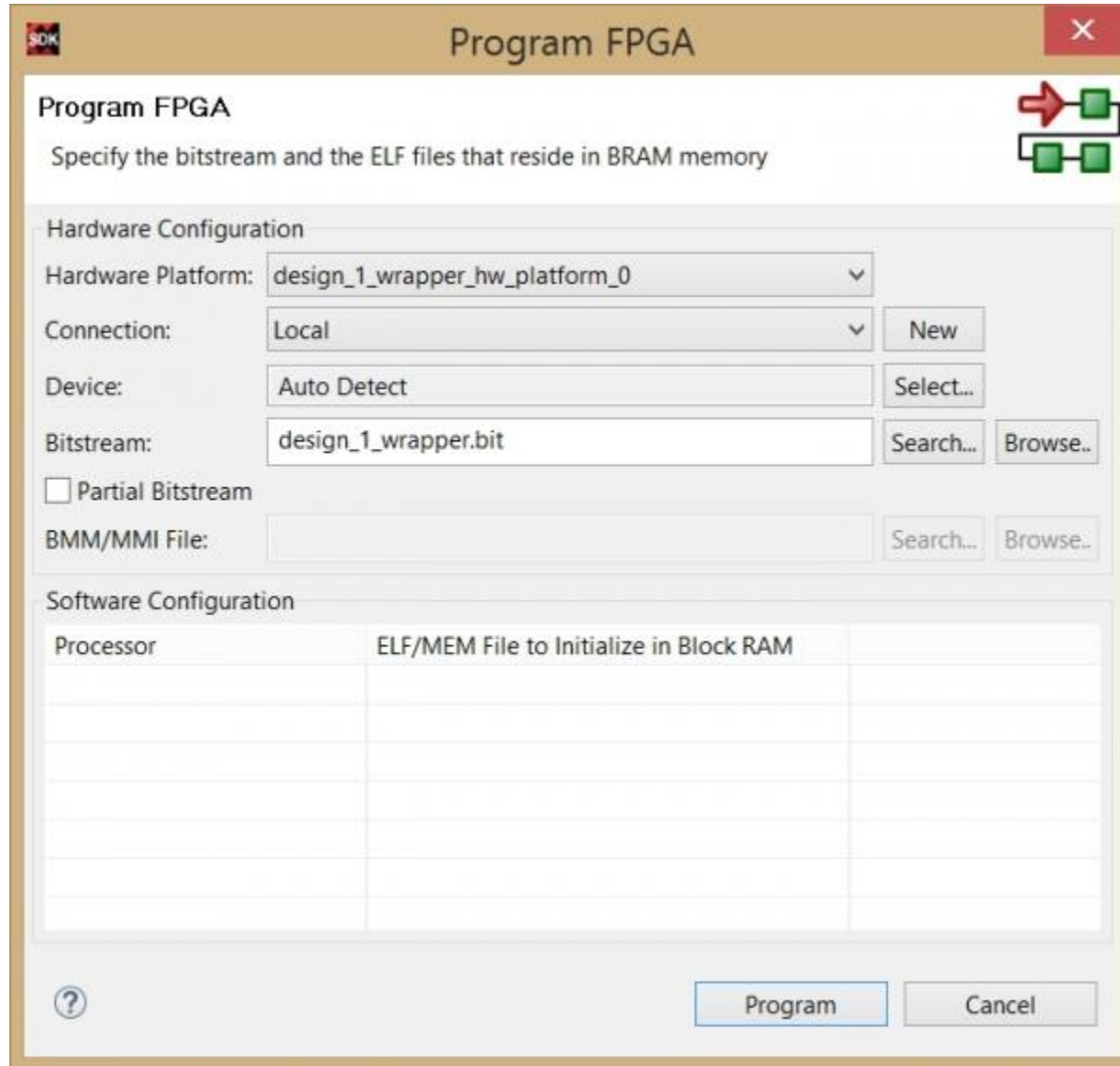
    else
        xil_printf("multiple buttons pressed\nWr");

    usleep(200000);          //delay
}
cleanup_platform();
return 0;
}

```

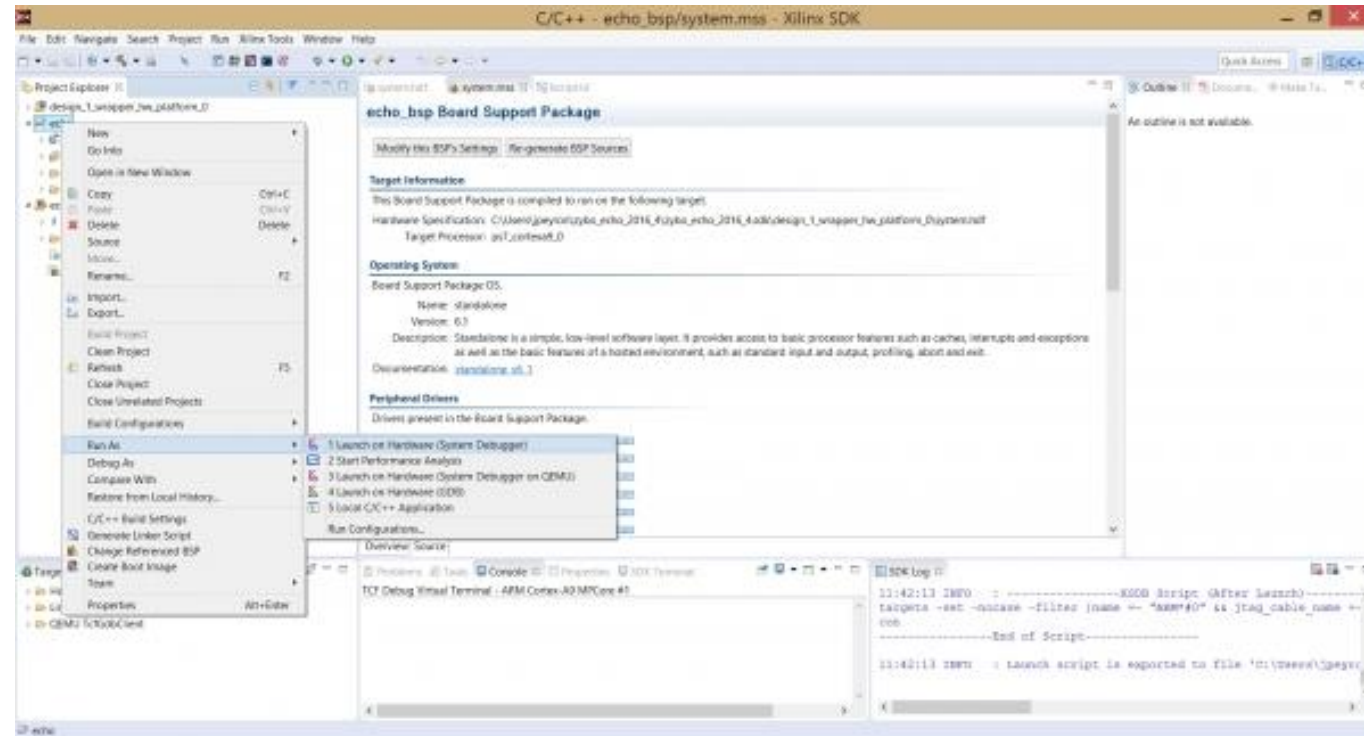
11. Run the Project

- 11.1) Zybo 가 UART USB 포트를 통해 Host PC 에 연결되어 있고 JP5 가 JTAG 로 설정되어 있는지 확인한다.
FPGA 를 프로그래밍 하려면 상단 툴바에서 Program FPGA 버튼을 클릭한다.



11.2) 프로젝트를 저장한다.
프로젝트가 자동으로 Build 된다.

11.3) getting_started_with_ZYBO 디렉토리를 마우스 우클릭하고 run as -> Launch on Hardware(System Debugger) 를 선택한다.



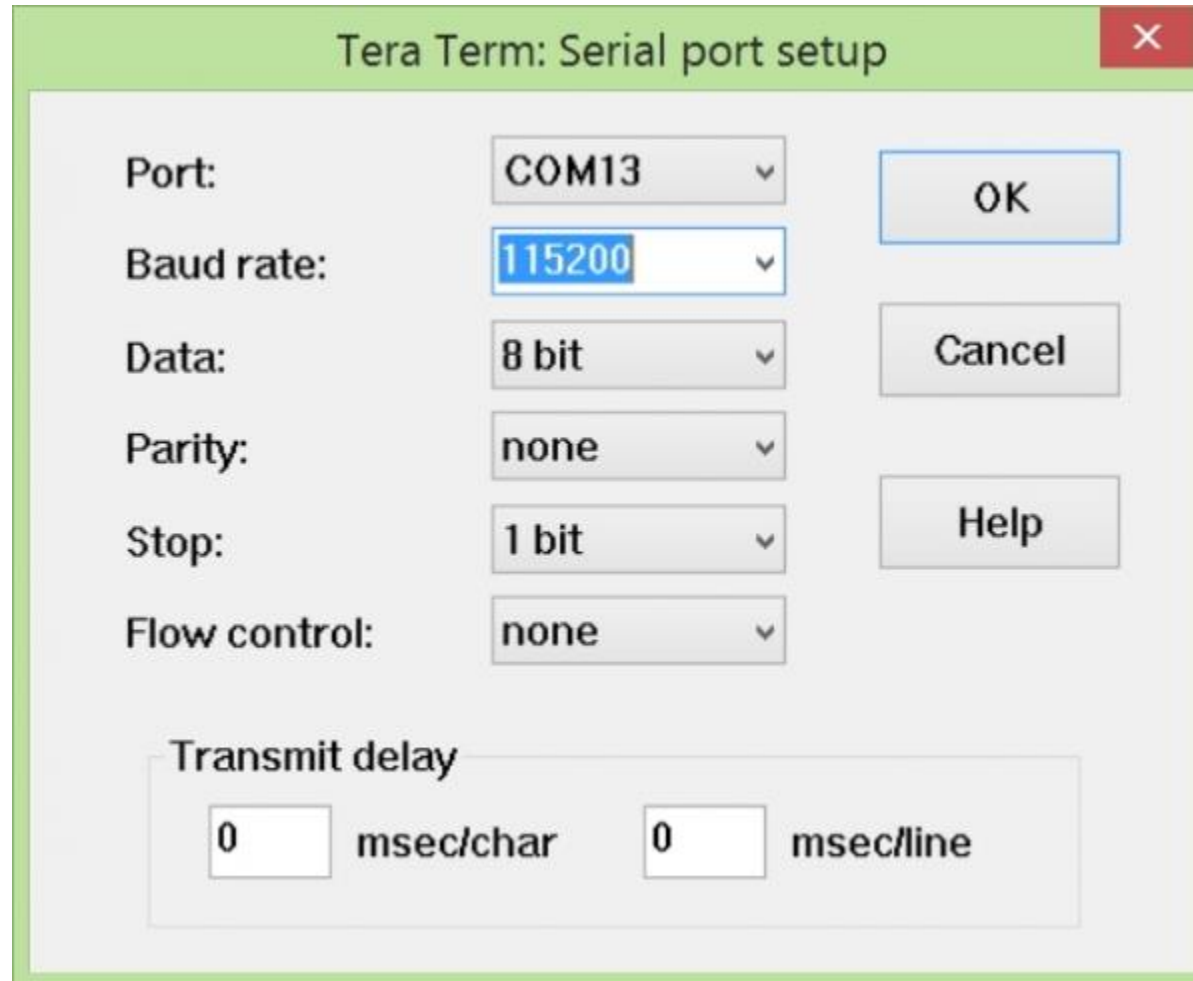
11.4) 데모는 ZYBO 에서 실행된다.

4 개의 스위치 (SW0-SW3) 으로 놀아보자.

또한 직렬 포트를 통해 각 버튼(BTN0-BTN3 으로 표시)을 누르면 "button x pressed" 라는 메시지가 나타난다.

11.5) Tera Term 또는 모든 직렬 터미널은 BTN 의 출력을 표시하기 위한 콘솔로 작동한다.

터미널을 아래와 같이 설정한다.



The image shows a screenshot of the 'Tera Term: Serial port setup' dialog box. The dialog has a green title bar with a red close button. The settings are as follows:

Setting	Value
Port:	COM13
Baud rate:	115200
Data:	8 bit
Parity:	none
Stop:	1 bit
Flow control:	none

Buttons on the right: OK, Cancel, Help.

Transmit delay section:

Unit	Value
msec/char	0
msec/line	0

References

1. <https://reference.digilentinc.com/learn/programmable-logic/tutorials/zybo-getting-started-with-zynq/start?redirect=1>