

Xilinx Zynq FPGA, TI DSP, MCU 기반 의 회로 설계 및 임베디드 전문가 과정

강사 - Innova Lee(이상훈)

gcccompil3r@gmail.com

학생 - Hyungjoo Kim(김형주)

mihaelkel@naver.com

주요 내용

1. Custom IP 만드는 방법(PWM, eCAP)
2. 만든 커스텀 IP 불러오는 방법.
3. SDK 상에서의 eCAP 루프백 테스트

1. 커스텀 IP 가 저장될 디렉토리를 생성한다.

```
howard@Howard:~$ cd FPGA_Proj/  
howard@Howard:~/FPGA_Proj$ mkdir custom_IP  
howard@Howard:~/FPGA_Proj$
```

2. Vivado 를 실행하여 프로젝트를 생성한다.

New Project

Project Name

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

Project name: custom_ip

Project location: /home/howard/FPGA_Proj/custom_IP

☐ Create project subdirectory

Project will be created at: /home/howard/FPGA_Proj/custom_IP

New Project

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

New Project

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

Reset All Filters

Search: (3 matches)

Display Name	Vendor	Board Rev	Part	I/O Pin C
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

3.상단바의 Tools 를 눌러 Create and Package new IP 를 누른다.

4.Create a new AXI4 peripheral

Create and Package New IP

Create Peripheral, Package IP or Package a Block Design
Please select one of the following tasks.

Packaging Options

- ☐ Package your current project
Use the project as the source for creating a new IP Definition.
- ☐ Package a block design from the current project
Choose a block design as the source for creating a new IP Definition.
- ☐ Package a specified directory
Choose a directory as the source for creating a new IP Definition.

Create AXI4 Peripheral

- ☒ Create a new AXI4 peripheral
Create an AXI4 IP, driver, software test application, IP Integrator AXI4 VIP simulation and debug demonstration design.

5.Name 과 IP location 을 설정해주고, Overwrite existing 체크박스를 체크한다.
이 예제에서는 모터용 PWM IP 를 생성하고 뒷부분에서 eCAP IP 를 생성할 것이다.

Create and Package New IP

Peripheral Details
Specify name, version and description for the new peripheral

Name: PWM_50Hz

Version: 1.0

Display name: PWM_50Hz_v1.0

Description: My new AXI IP

IP location: /home/howard/FPGA_Proj/custom_IP

☒ Overwrite existing

6.Next 를 눌러 넘어가고, 아래 부분에서 Edit IP 를 선택한다.

VIVADO
HLS Editions

Create Peripheral

Peripheral Generation Summary

- 1. IP (user.org:user:PWM_50Hz:1.0) with 1 interface(s)
- 2. Driver(v1_00_a) and testapp [more info](#)
- 3. AXI4 VIP Simulation demonstration design [more info](#)
- 4. AXI4 Debug Hardware Simulation demonstration design [more info](#)

Peripheral created will be available in the catalog :
/home/howard/FPGA_Proj/custom_IP

Next Steps:

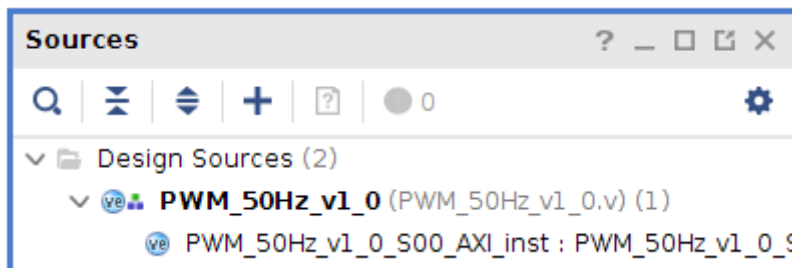
- ☐ Add IP to the repository
- ☒ Edit IP
- ☐ Verify Peripheral IP using AXI4 VIP
- ☐ Verify peripheral IP using JTAG interface

Click Finish to continue

XILINX
ALL PROGRAMMABLE.

< Back Next > Finish Cancel

7.Design Sources 를 보면, 탑모듈과 하위모듈 두 개가 생성되어 있음을 볼 수 있다.
하위모듈을 더블클릭하여 내부 코드를 연다.



8.파라미터 설정 부분이 아래와 같이 보이는 데,

```
1 |
2 | `timescale 1 ns / 1 ps
3 |
4 | module PWM_50Hz_v1_0_S00_AXI #
5 | (
6 |     // Users to add parameters here
7 |
8 |     // User parameters ends
9 |     // Do not modify the parameters beyond this line
10 |
11 |    // Width of S_AXI data bus
12 |    parameter integer C_S_AXI_DATA_WIDTH    = 32,
13 |    // Width of S_AXI address bus
14 |    parameter integer C_S_AXI_ADDR_WIDTH    = 4
15 | )
```

parameter integer PWM_PRESCALE = 2000000,

를 아래와 같이 추가한다. 이는 Zybo 보드의 동작 주파수가 100MHz 이고, 모터를 구동하는 데 필요한 주파수가 50Hz 이기 때문에, Prescaler = 100MHz/50Hz = 2000000 이라는 계산을 통해 나온 수치이다.
만약 Zybo 보드의 동작주파수를 변경하고 싶다면, 위의 수치 또한 변경되어야 할 것이다.

```
1 |
2 | `timescale 1 ns / 1 ps
3 |
4 | module PWM_50Hz_v1_0_S00_AXI #
5 | (
6 |     // Users to add parameters here
7 |     parameter integer PWM_PRESCALE = 2000000,
8 |     // User parameters ends
9 |     // Do not modify the parameters beyond this line
10 |
11 |    // Width of S_AXI data bus
12 |    parameter integer C_S_AXI_DATA_WIDTH    = 32,
13 |    // Width of S_AXI address bus
14 |    parameter integer C_S_AXI_ADDR_WIDTH    = 4
15 | )
```

9. 17 번 라인에 유저 포트 추가 부분이 있는데,

```
17 // Users to add ports here
18
19 // User ports ends
```

output wire pwm,

을 아래와 같이 추가한다.

```
17 // Users to add ports here
18 output wire pwm,
19 // User ports ends
```

10. 400 번 라인쪽을 보면, 실제 로직을 추가하는 부분이 있다.

```
400 // Add user logic here
401
402 // User logic ends
```

```
reg [31:0] counter;
always @(posedge S_AXI_ACLK)
begin
    if(counter == PWM_PRESCALE)
        counter <= 0;
    else
        counter <= counter + 1;
end
assign pwm = counter < slv_reg0 ? 1'b1 : 1'b0;
```

위 코드를 아래와 같이 추가한다.

```
400 // Add user logic here
401 reg [31:0] counter;
402 always @(posedge S_AXI_ACLK)
403 begin
404     if(counter == PWM_PRESCALE)
405         counter <= 0;
406     else
407         counter <= counter + 1;
408 end
409 assign pwm = counter < slv_reg0 ? 1'b1 : 1'b0;
410 // User logic ends
```

11. 탭모듈을 더블클릭하여 코드 파일을 연다.

12.아래와 같이 6 번 라인에 파라미터를 추가하는 부분이 있다.

```
4 module PWM_50Hz_v1_0 #
5 (
6     // Users to add parameters here
7
8     // User parameters ends
9     // Do not modify the parameters beyond this line
10
11
12     // Parameters of Axi Slave Bus Interface S00_AXI
13     parameter integer C_S00_AXI_DATA_WIDTH = 32,
14     parameter integer C_S00_AXI_ADDR_WIDTH = 4
15 )
```

parameter integer PWM_PRESCALE = 2000000,

위 코드를 아래와 같이 추가한다.

```
4 module PWM_50Hz_v1_0 #
5 (
6     // Users to add parameters here
7     parameter integer PWM_PRESCALE = 2000000,
8     // User parameters ends
9     // Do not modify the parameters beyond this line
10
11
12     // Parameters of Axi Slave Bus Interface S00_AXI
13     parameter integer C_S00_AXI_DATA_WIDTH = 32,
14     parameter integer C_S00_AXI_ADDR_WIDTH = 4
15 )
```

13.17 번 라인에 포트 추가 부분을 아래와 같이 수정한다.

```
17 // Users to add ports here
18
19 // User ports ends
20 // Do not modify the ports beyond this line
```

output wire pwm,

```
17 // Users to add ports here
18 output wire pwm,
19 // User ports ends
20 // Do not modify the ports beyond this line
```

14.47 라인부터는 하위 모듈의 파라미터와 포트를 상위 모듈에서 연결하는 부분이다.

```
47 PWM_50Hz_v1_0 S00_AXI # (
48     .C_S_AXI_DATA_WIDTH(C_S00_AXI_DATA_WIDTH),
49     .C_S_AXI_ADDR_WIDTH(C_S00_AXI_ADDR_WIDTH)
```

먼저 위와 같은 파라미터 설정에서 PWM_PRESCALE 값을 추가해준다.

.PWM_PRESCALE(PWM_PRESCALE),

```
47 PWM_50Hz_v1_0 S00_AXI # (
48     .C_S_AXI_DATA_WIDTH(C_S00_AXI_DATA_WIDTH),
49     .C_S_AXI_ADDR_WIDTH(C_S00_AXI_ADDR_WIDTH),
50     .PWM_PRESCALE(PWM_PRESCALE)
```

콤마(,) 와 점(.)의 위치에 유의한다.

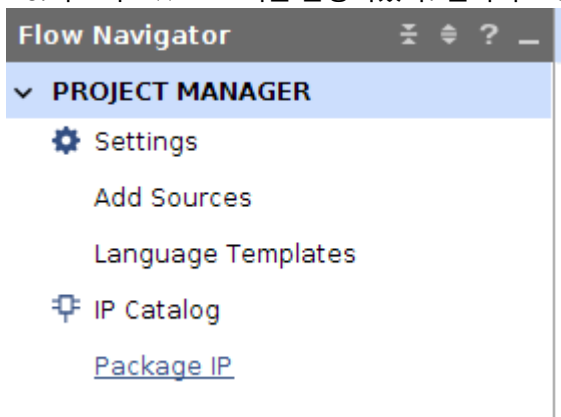
15.아래쪽 포트 부분도 살펴본다.

```
51 |         ) PWM_50Hz_v1_0_S00_AXI_inst (
52 |             .S_AXI_ACLK(s00_axi_aclk),
53 |             .S_AXI_ARESETN(s00_axi_aresetn),
54 |             .S_AXI_AWADDR(s00_axi_awaddr),
```

pwm 포트를 연결한다.

```
.pwm(pwm),
51 |         ) PWM_50Hz_v1_0_S00_AXI_inst (
52 |             .pwm(pwm),
53 |             .S_AXI_ACLK(s00_axi_aclk),
54 |             .S_AXI_ARESETN(s00_axi_aresetn),
55 |             .S_AXI_AWADDR(s00_axi_awaddr),
```

16.이로써 PWM 로직은 완성되었다. 왼쪽의 Flow Navigator 에서 Package IP 를 클릭한다.



17.Compatibility 탭으로 가서, + 버튼을 눌러 artix7 을 추가한다.

Project Summary x Package IP - PWM_50Hz x PWM_50Hz_v1_0_S00_AXI.v x PW

Packaging Steps

✓ Identification

✓ **Compatibility**

File Groups

Customization Parameters

Packaging Steps

✓ Identification

✓ **Compatibility**

File Groups

Compatibility

Family | Simulator

+ - ↑ ↓

Family	Life Cycle
zynq	Pre-Production

✕ □ **Add Family**

File Groups

! Merge changes from File Groups Wizard

🔍 ⚡ ⚙️ 🗑️ + ↺

Name

Standard

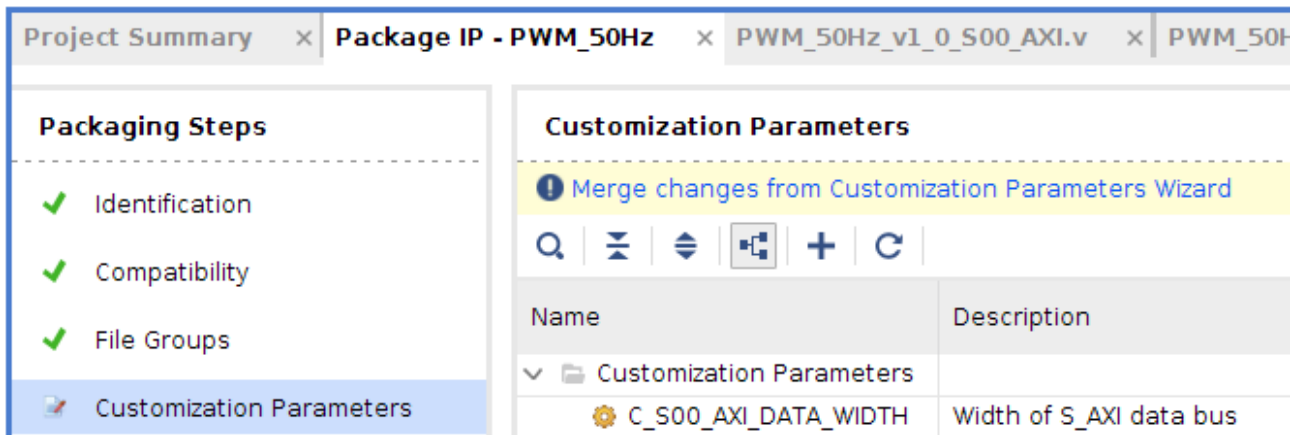
> ☐ aartix7 (Artix-7)

> ☒ artix7 (Artix-7)

> ☐ artix7l (Artix-7)

18. File Groups 탭으로 가서, Merge changes from File Groups Wizard 를 누른다.

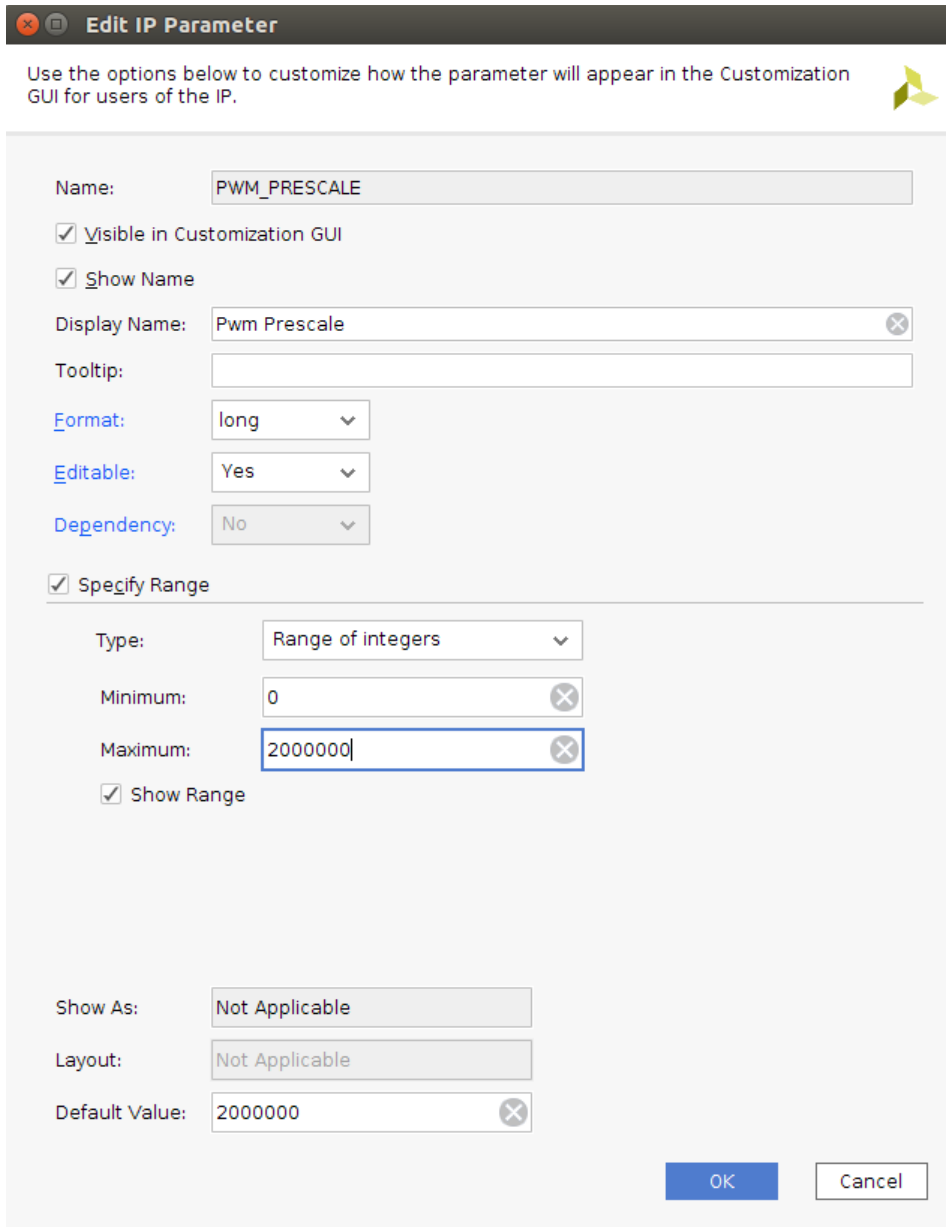
19. Customization Parameters 탭으로 가서, Merge changes from Customization Parameters Wizard 를 클릭한다.



20. Hidden Parameters 의 왼쪽에 화살표 모양을 눌러 확장하면, PWM_PRESCALE 이라는 위쪽에서 설정한 파라미터가 보일 것이다. 더블클릭한다.

Name	Description	Display Name	
Customization Parameters			
C_S00_AXI_DATA_WIDTH	Width of S_AXI data bus	C S00 AXI DATA WIDTH	
C_S00_AXI_ADDR_WIDTH	Width of S_AXI address bus	C S00 AXI ADDR WIDTH	
C_S00_AXI_BASEADDR		C S00 AXI BASEADDR	
C_S00_AXI_HIGHADDR		C S00 AXI HIGHADDR	
Hidden Parameters			
PWM_PRESCALE		Pwm Prescale	

21.아래와 같이 설정한다.



Edit IP Parameter

Use the options below to customize how the parameter will appear in the Customization GUI for users of the IP.

Name: PWM_PRESCALE

☒ Visible in Customization GUI

☒ Show Name

Display Name: Pwm Prescale

Tooltip:

Format: long

Editable: Yes

Dependency: No

☒ Specify Range

Type: Range of integers

Minimum: 0

Maximum: 2000000

☒ Show Range

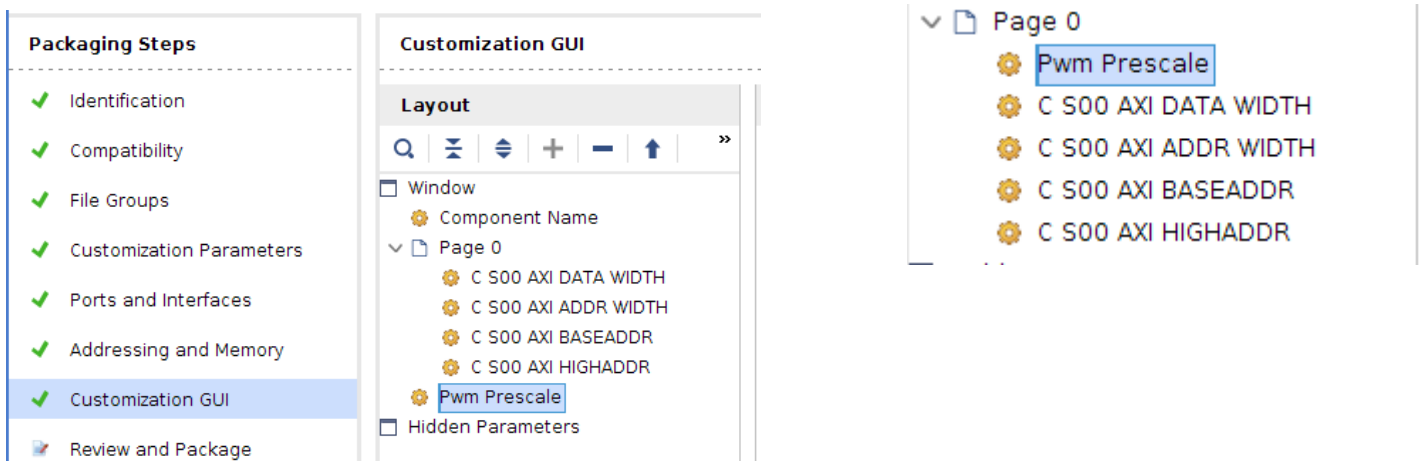
Show As: Not Applicable

Layout: Not Applicable

Default Value: 2000000

OK Cancel

22. Customization GUI 탭으로 가면, Pwm Prescale 이 Page0 바깥 쪽에 있음을 볼 수 있다.
Pwm Prescale 을 드래그하여 오른쪽 그림과 같이 Page 0 밑으로 넣어준다.



Packaging Steps

- Identification
- Compatibility
- File Groups
- Customization Parameters
- Ports and Interfaces
- Addressing and Memory
- Customization GUI**
- Review and Package

Customization GUI

Layout

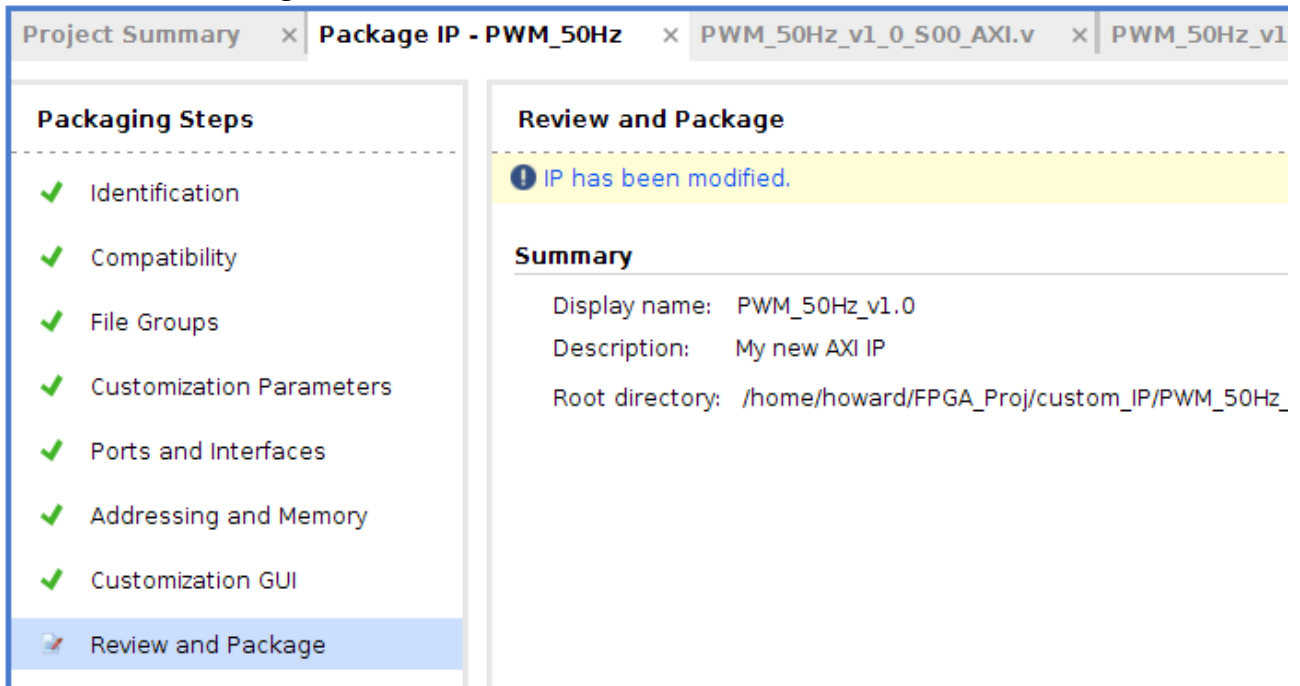
Window

- Component Name
- Page 0
 - C S00 AXI DATA WIDTH
 - C S00 AXI ADDR WIDTH
 - C S00 AXI BASEADDR
 - C S00 AXI HIGHADDR
 - Pwm Prescale**
- Hidden Parameters

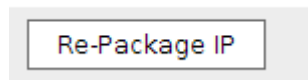
Page 0

- Pwm Prescale**
- C S00 AXI DATA WIDTH
- C S00 AXI ADDR WIDTH
- C S00 AXI BASEADDR
- C S00 AXI HIGHADDR

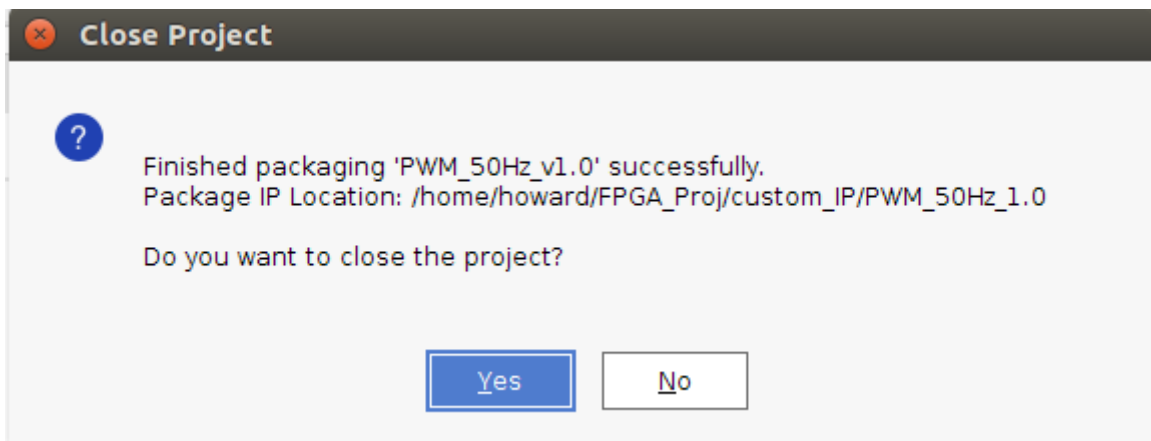
23.Review and Package 탭으로 가서, IP has been modified 를 클릭한다.



24. Re-Package IP 를 클릭하여 IP 를 생성한다.



아래와 같은 메시지가 뜨면, 성공이고 Yes 를 눌러 종료한다.



25.이번엔 eCAP IP 를 생성할 것이다.

마찬가지로 상단의 Tools → Create and Package new IP 를 눌러 IP 를 생성한다.

선택 옵션은 PWM IP 만들 때와 동일하다.


Create AXI4 Peripheral



Create a new AXI4 peripheral

Create an AXI4 IP, driver, software test application, IP Integrator AXI4 VIP simulation and debug demonstration design.

Create and Package New IP

Peripheral Details
Specify name, version and description for the new peripheral

Name:

Version:

Display name:

Description:

IP location:

☒ Overwrite existing

< Back

Next >

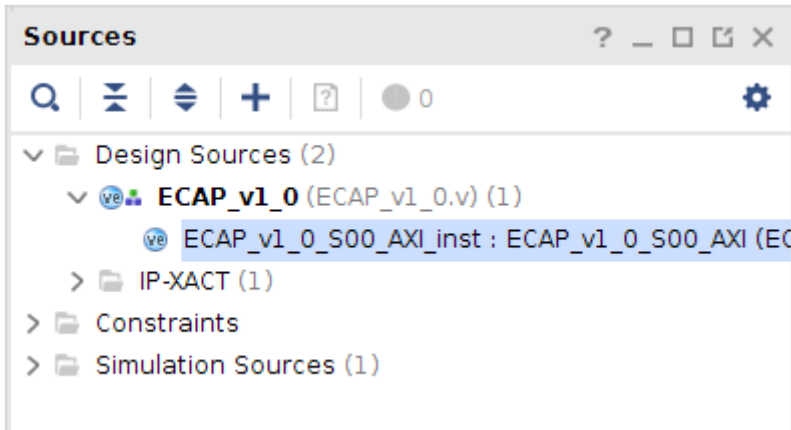
Finish

Cancel

Next Steps:

- ☐ Add IP to the repository
- ☒ Edit IP
- ☐ Verify Peripheral IP using AXI4 VIP
- ☐ Verify peripheral IP using JTAG interface

26.하위 모듈을 더블클릭하여, 소스 코드를 엽니다.



27.eCAP 에서는 파라미터가 필요 없으므로, 바로 18 번 라인에 포트를 추가한다.

```
input wire pwm_in,
```

```
17 | ..... // Users to add ports here
18 |         input wire pwm_in,
19 | ..... // User ports ends
```

28. 84 번 라인에 아래와 같이 7 개의 레지스터들을 선언한다.

```
reg [31:0] period;
reg [31:0] duty;
reg [31:0] counter;
reg edge_state;
reg [31:0] cnt1;
reg [31:0] cnt2;
reg [31:0] cnt3;
```

```
84 |         reg [31:0] period;
85 |         reg [31:0] duty;
86 |         reg [31:0] counter;
87 |         reg edge_state;
88 |         reg [31:0] cnt1;
89 |         reg [31:0] cnt2;
90 |         reg [31:0] cnt3;
91 |
92 |         // AXI4LITE signals
93 |         reg [C_S_AXI_ADDR_WIDTH-1 : 0] axi_awaddr;
94 |         reg axi_awready;
```

29.아래와 같이 381, 382 번 라인에 slv_reg0, slv_reg1 을 각 각 period 와 duty 로 바꾼다.

이는 SDK 혹은 device driver 에서 period 와 duty 값을 읽어오기 위함이다.

```
376 : assign slv_reg_rden = axi_arready & S_AXI_ARVALID & ~axi_rvalid;
377 : always @(*)
378 : begin
379 :     // Address decoding for reading registers
380 :     case ( axi_araddr[ADDR_LSB+OPT_MEM_ADDR_BITS:ADDR_LSB] )
381 :         2'h0 : reg_data_out <= period;
382 :         2'h1 : reg_data_out <= duty;
383 :         2'h2 : reg_data_out <= slv_reg2;
384 :         2'h3 : reg_data_out <= slv_reg3;
385 :         default : reg_data_out <= 0;
386 :     endcase
387 : end
```

30.409 번 라인의 유저 작성 부분에 아래 코드를 작성한다.

```
// Add user logic here
/*
input wire pwm_in;

reg [31:0] period;
reg [31:0] duty;
reg edge_state;
reg cnt1;
reg cnt2;
reg cnt3;
*/
always @(posedge S_AXI_ACLK)
begin
    counter <= counter + 32'd1;
    if(counter == 32'd1000000000)
        counter <= 32'd0;

    if(pwm_in == 1'b1)
    begin
        //edge detection : rising
        if(edge_state == 1'b0)
        begin
            cnt3 <= counter;
            //prevent overflow
            if(cnt3 > cnt1)
                period <= cnt3 - cnt1;
            else
                period <= cnt3 + (32'd1000000000 - cnt1);
            cnt1 <= cnt3;
            edge_state <= 1'b1;
        end
    end

    else if(pwm_in == 1'b0)
    begin
        //edge detection : polling
        if(edge_state == 1'b1)
        begin
```

```

        cnt2 <= counter;
        //prevent overflow
        if(cnt2 > cnt1)
            duty <= cnt2 - cnt1;
        else
            duty <= cnt2 + (32'd1000000000 - cnt1);
        edge_state <= 1'b0;
    end
end

end
// User logic ends

```

31.탑 모듈로 가서, 17 번 라인에 포트를 추가한다.

```
input wire pwm_in,
```

```

17 :           // Users to add ports here
18 :           input wire pwm_in,
19 :           // User ports ends

```

32.51 번 라인에 가서 포트를 연결해준다.

```
.pwm_in(pwm_in),
```

```

50 :           ) ECAP_v1_0_S00_AXI_inst (
51 :               .pwm_in(pwm_in),
52 :               .S_AXI_ACLK(s00_axi_aclk),

```

33.Flow Navigator 로 가서 Package IP 를 누른다.

34. Compatibility 에서 artix7 을 추가한다.

Packaging Steps

- Identification
- Compatibility
- File Groups
- Customization Parameters

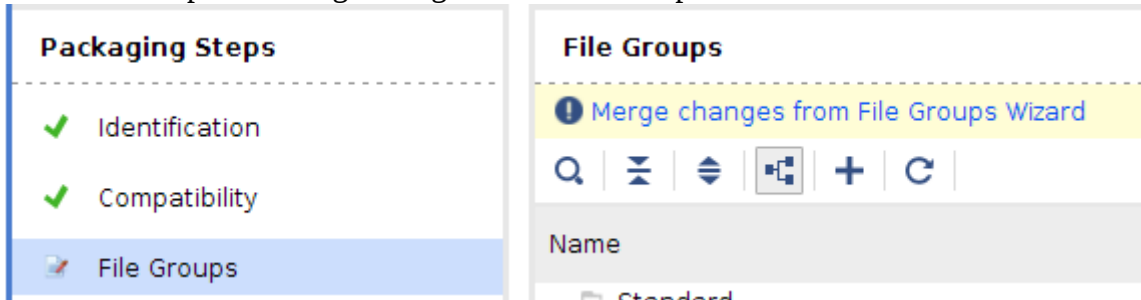
Compatibility

Family | Simulator

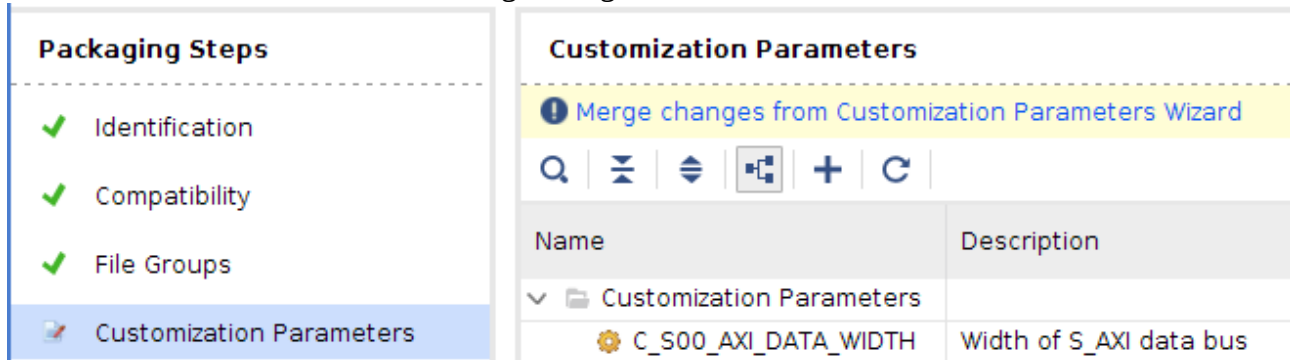
+
-
↑
↓

Family	Life Cycle
artix7	Beta
zynq	Pre-Production
artix7	Beta

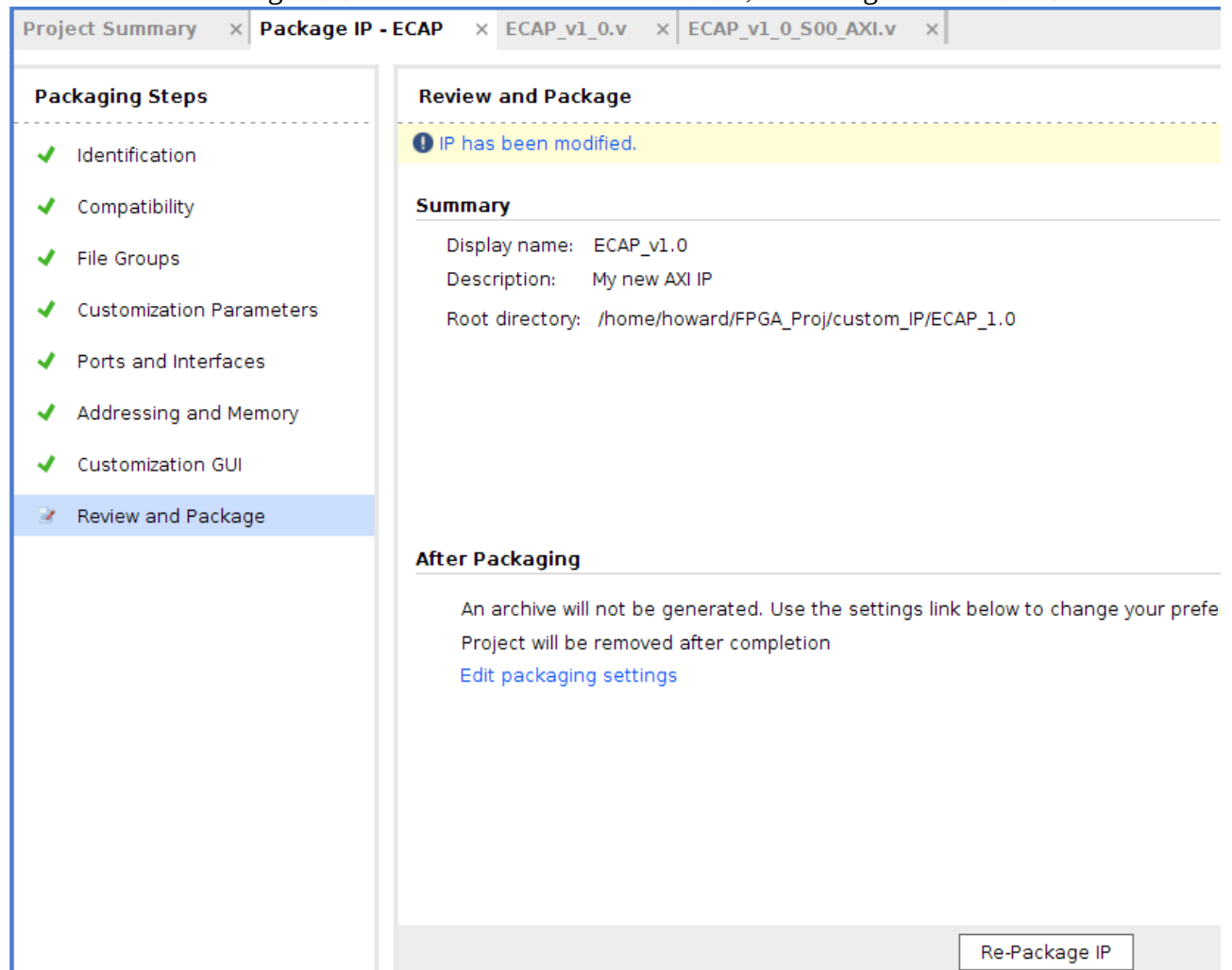
35. File Groups 에서 Merge changes from File Groups Wizard 를 클릭한다.



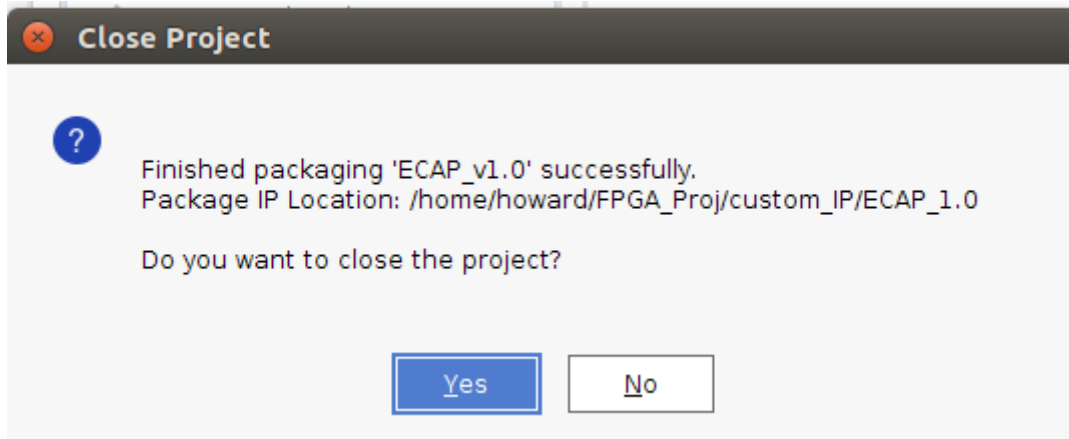
36. Customization Parameters 에서 Merge changes from Customization Parameters Wizard 를 클릭.



37. Review and Package 에서 IP has been modified 를 클릭하고, Re-Package IP 를 클릭한다.



38.아래와 같은 메시지가 뜬다면 성공. Yes 를 눌러 종료한다.



39.생성한 모듈 두개를 터미널에서 확인한 후, Vivado 를 종료한다.

```
howard@Howard:~/FPGA_Proj/custom_IP$ ls
custom_ip.cache      custom_ip.sim      custom_ip.xpr
custom_ip.hw         custom_ip.srcs     ECAP_1.0
custom_ip.ip_user_files custom_ip.tmp      PWM_50Hz_1.0
```

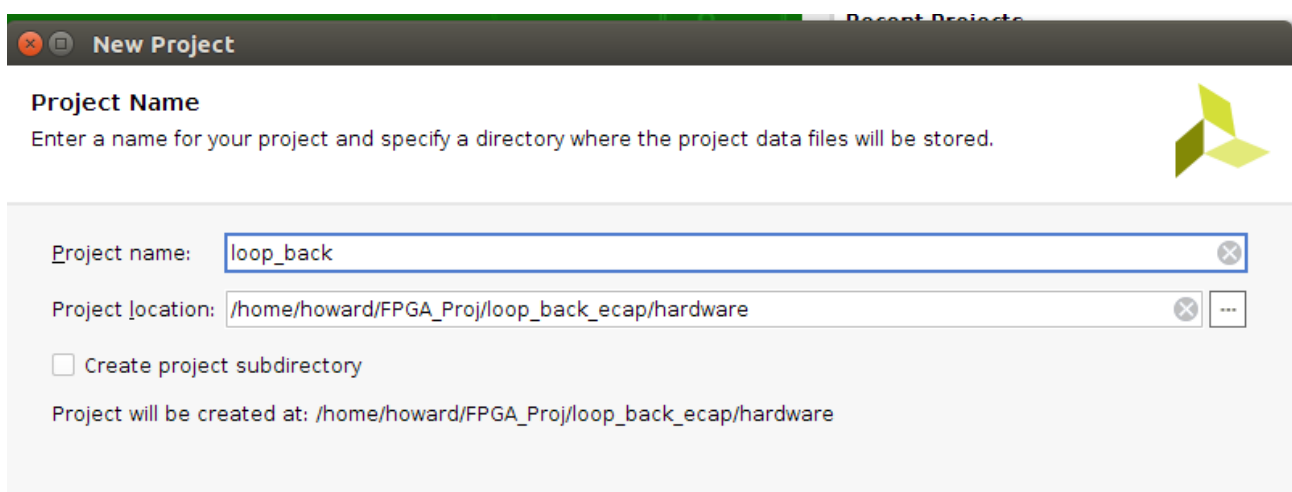
40.이제 PWM IP 와 eCAP IP 를 통해 루프백 테스트를 해볼 것이다.

아래와 같이 새 프로젝트를 저장할 디렉토리를 만들고, 그 안에 hardware 디렉토리를 만든다.

```
howard@Howard:~/FPGA_Proj/custom_IP$ cd ..
howard@Howard:~/FPGA_Proj$ ls
custom_IP  ip_repo  motor_pwm  pwm_ecap  read_reg
howard@Howard:~/FPGA_Proj$ mkdir loop_back_ecap
howard@Howard:~/FPGA_Proj$ cd loop_back_ecap/
howard@Howard:~/FPGA_Proj/loop_back_ecap$ mkdir hardware
howard@Howard:~/FPGA_Proj/loop_back_ecap$ ls
hardware
```

41.Vivado 를 다시 실행한다. 새 프로젝트를 만든다.

경로는 아래와 같이, 위에서 생성한 loop_back_ecap/hardware 로 지정하고, Create project subdirectory 체크박스를 해제한다.

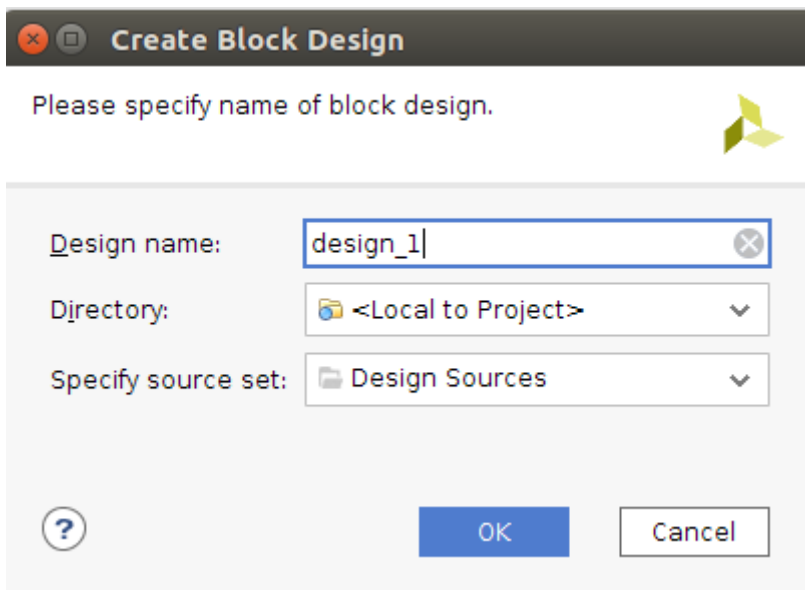


42. RTL Project 를 생성하고, 보드는 Zybo 를 선택한다.

- ☒ 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

43. Create Block Design 을 눌러 블록 디자인을 생성한다. 이름은 상관없다.

- ▼ IP INTEGRATOR
 - Create Block Design
 - Open Block Design
 - Generate Block Design



Create Block Design

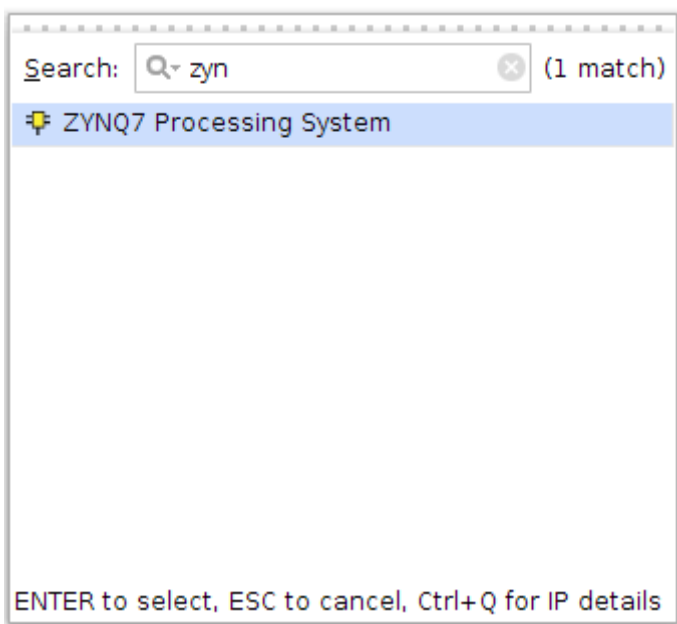
Please specify name of block design.

Design name:

Directory:

Specify source set:


44. Diagram 에서 ZYNQ7 Processing System 을 추가하여 Run Block Automation 을 누른다. 설정은 건드리지 않고, OK 를 누른다.



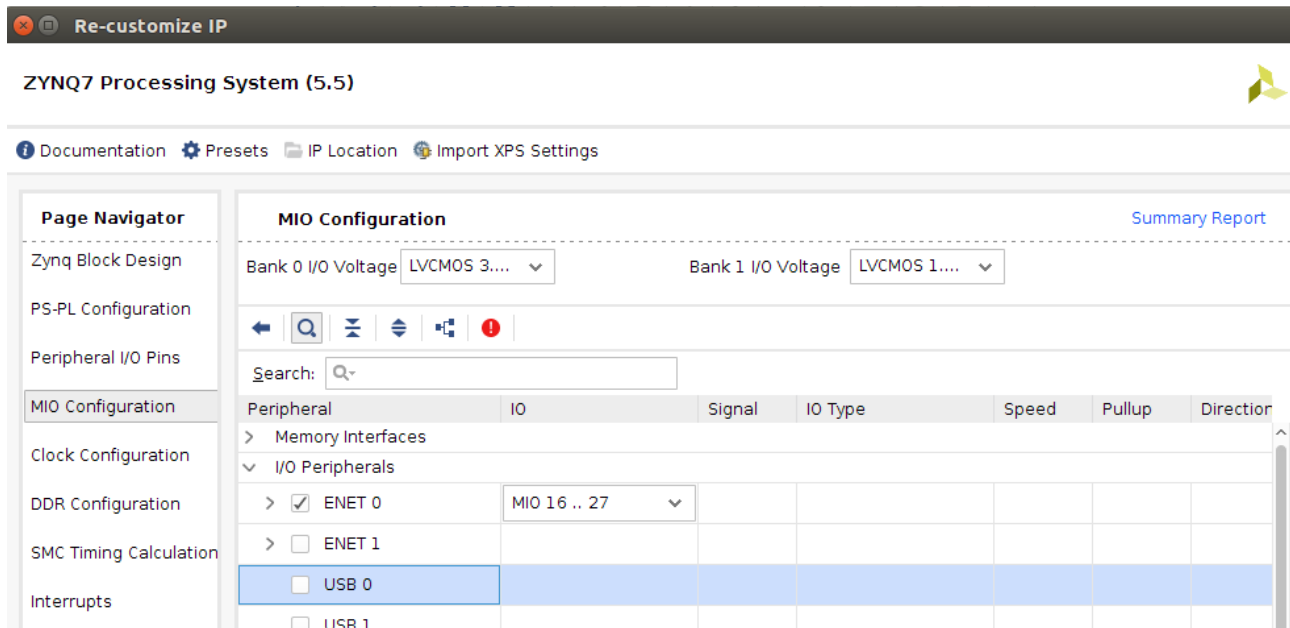
Search: (1 match)

☒ ZYNQ7 Processing System

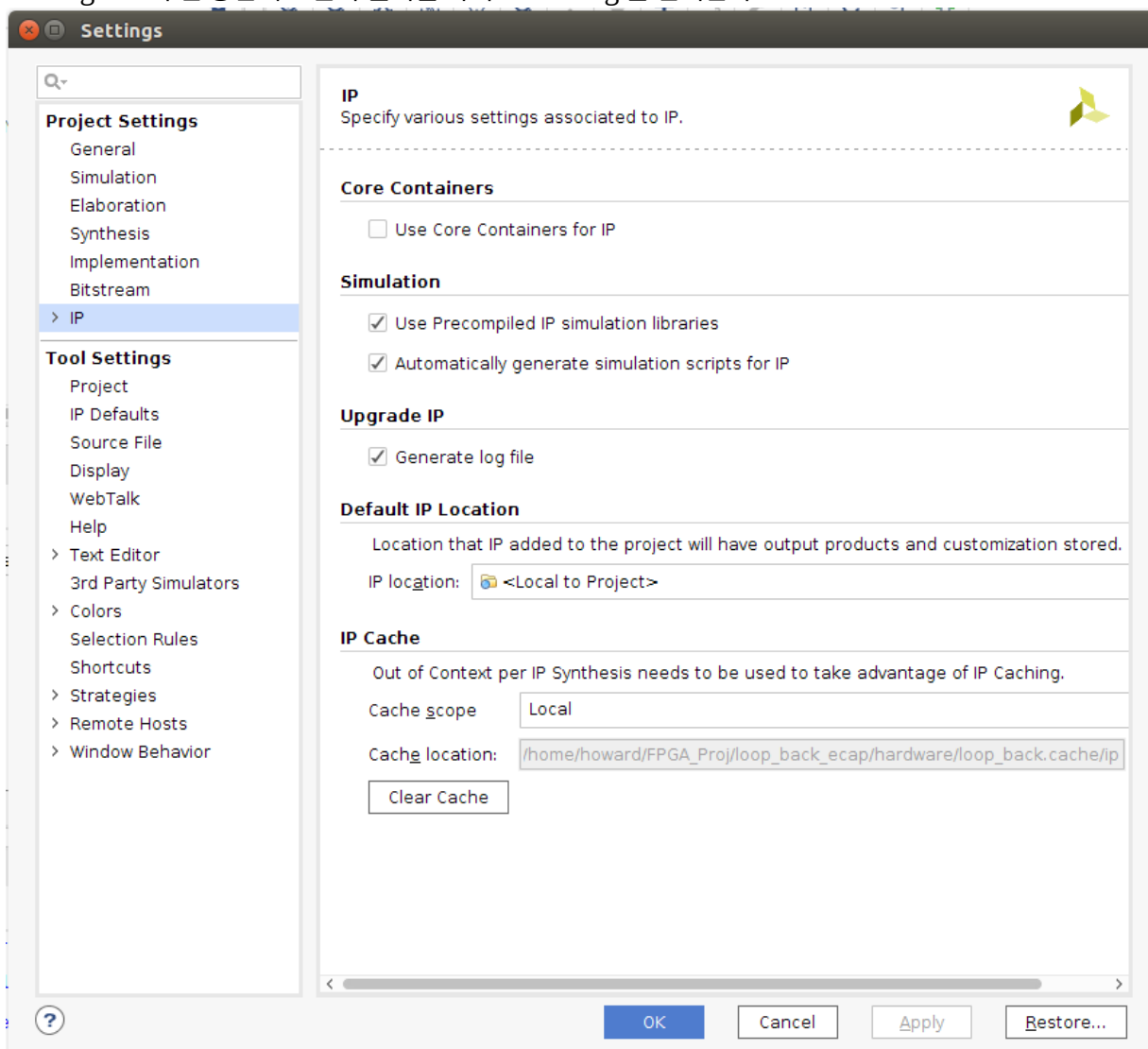
ENTER to select, ESC to cancel, Ctrl+Q for IP details

 Designer Assistance available. [Run Block Automation](#)

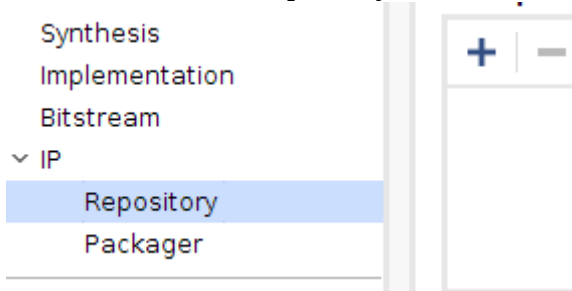
45.ZYNQ7 IP 를 더블클릭하여 MIO Configuration 에서 I/O Peripherals 의 USB0 를 체크 해제한 후 OK 를 누른다.



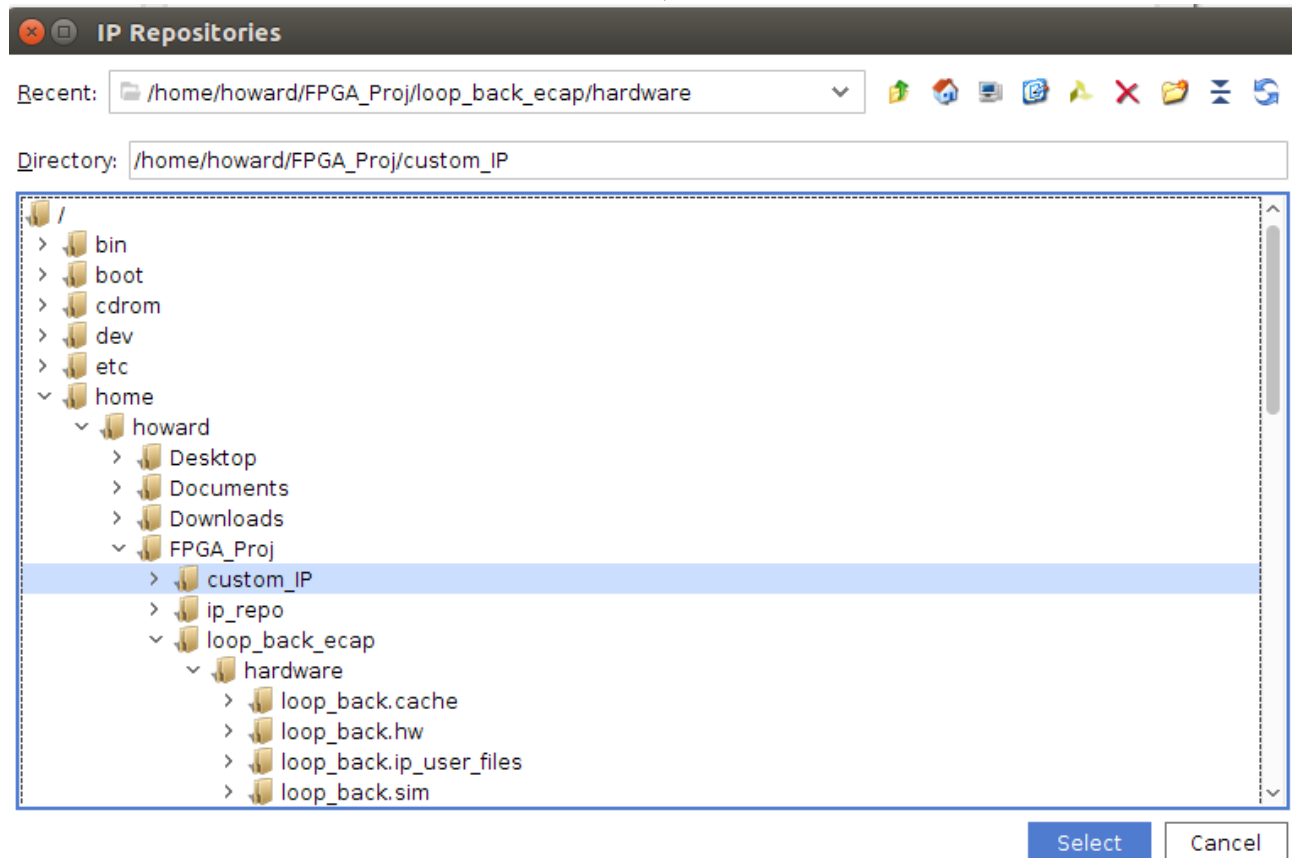
46.Diagram 의 빈 공간에 오른쪽 클릭을 하여 IP Setting 을 선택한다.



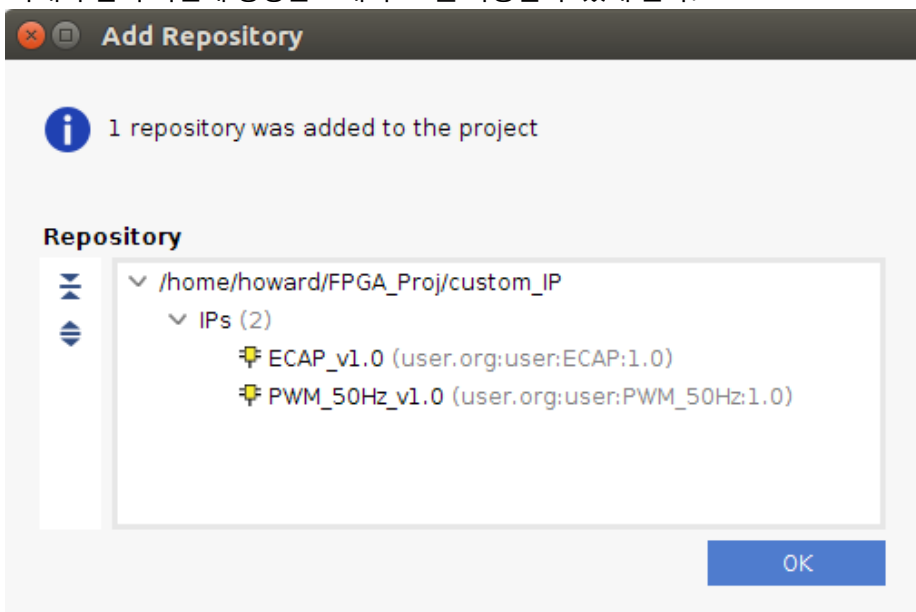
47.IP 탭을 확장하여 Repository 탭으로 가서 + 버튼을 누른다.



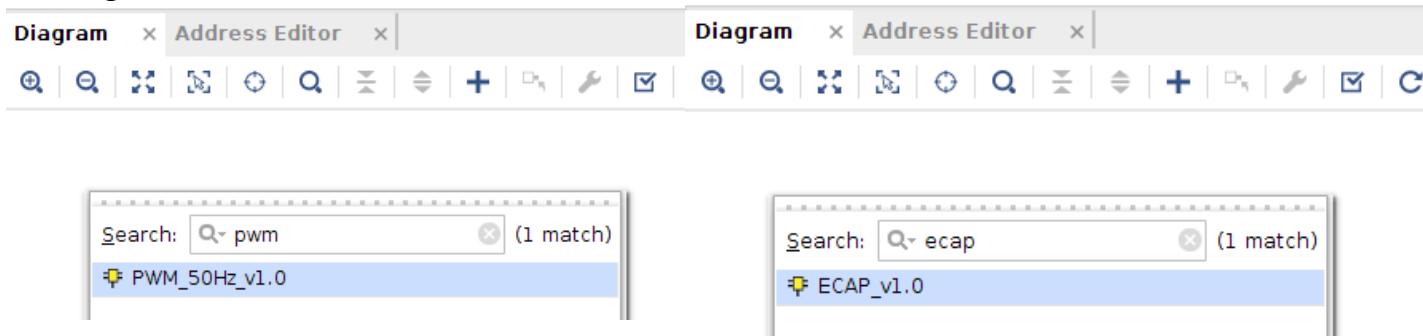
48.디렉토리를 custom_IP 를 선택하여 Select 를 누르면,



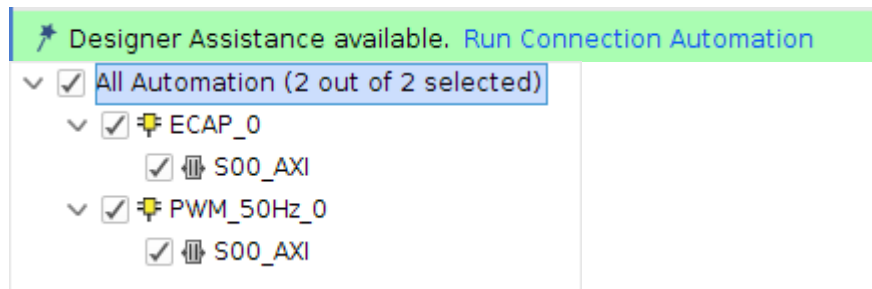
아래와 같이 이전에 생성한 2 개의 IP 를 사용할 수 있게 된다.



49.Diagram 에서 PWM_50Hz 와 ECAP 을 추가한다.

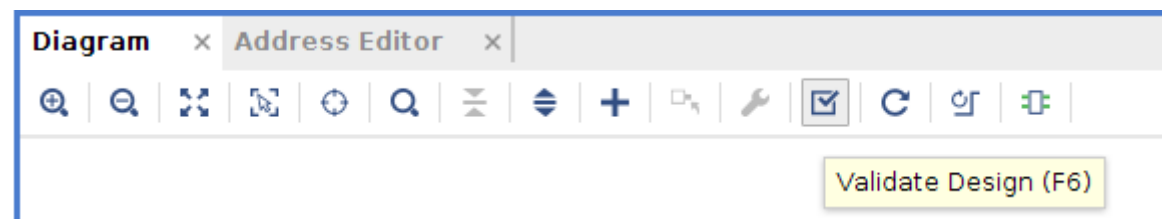


50.Run Connection Automation 을 클릭하고, 모든 옵션을 체크한다.

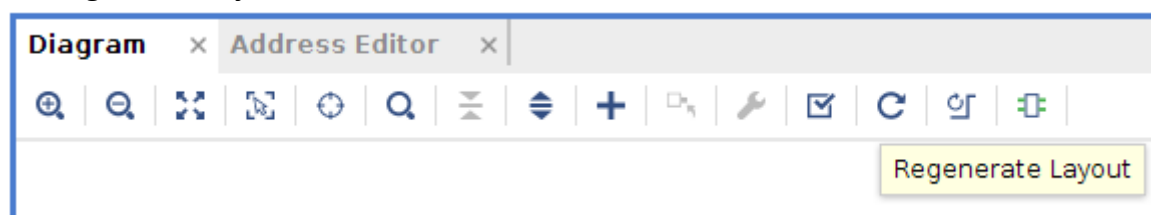


51.ECAP 의 pwm_in 핀과 PWM_50Hz 의 pwm 핀을 오른쪽 클릭하여 Create Port 를 한다.

52.Validate Design 을 한다.



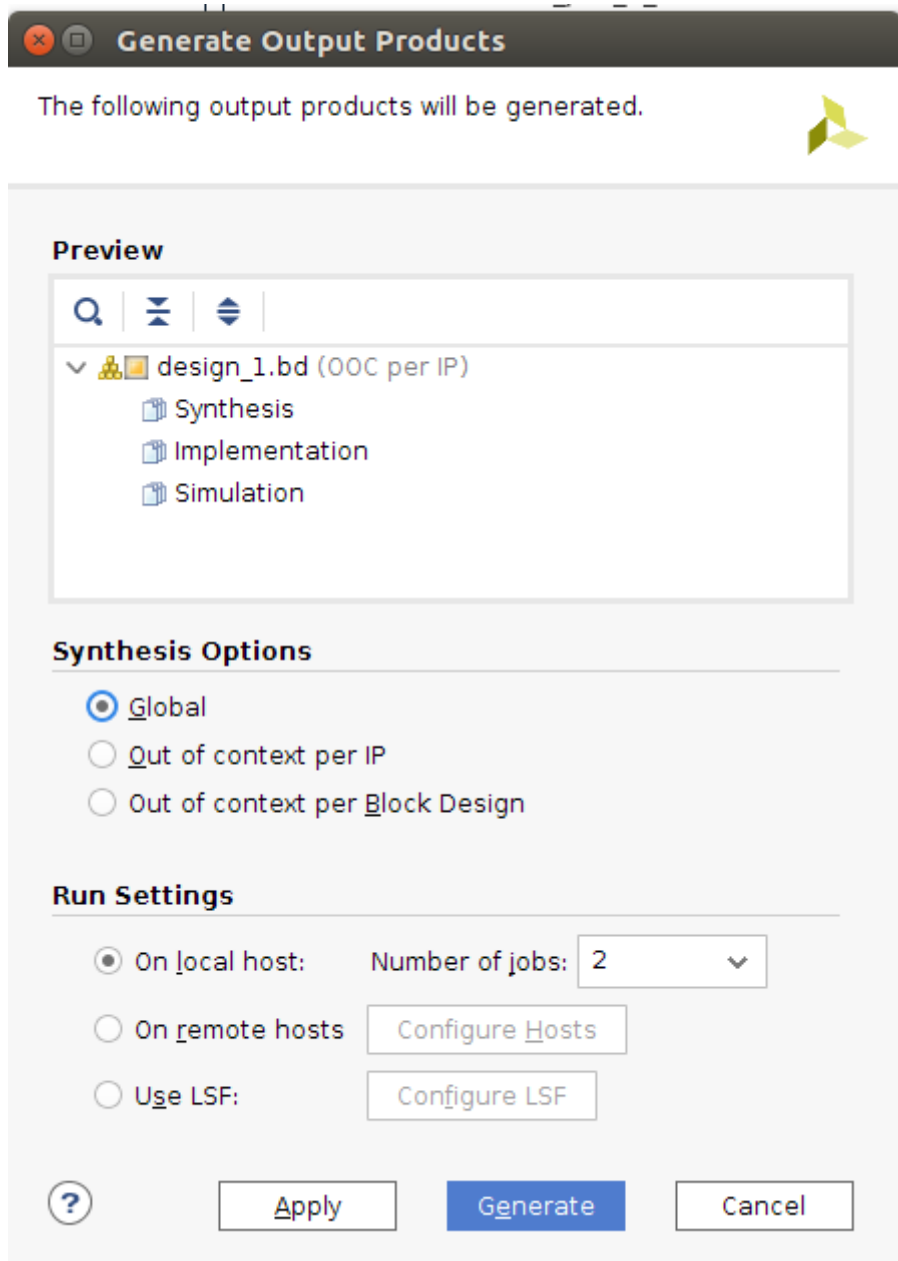
53.Regenerate Layout 을 클릭하여 다이어그램을 정돈한다.



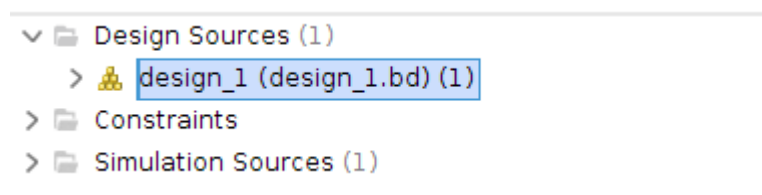
54.design 을 오른쪽 클릭하여 Generate Output Products 를 클릭한다.



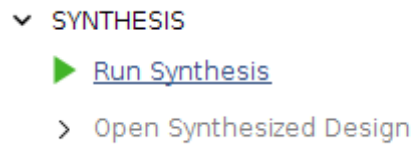
Synthesis Options 를 Global 로 변경하여 Generate 를 누른다.



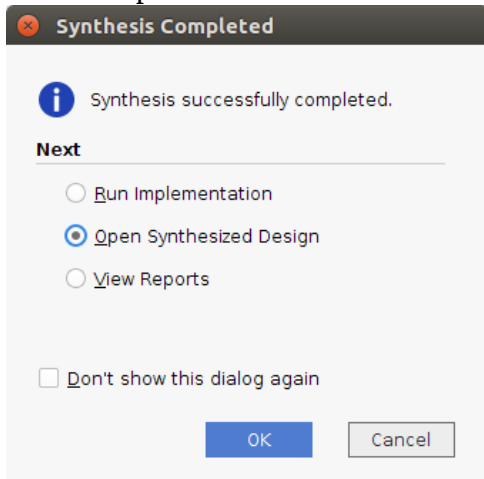
55. 마찬가지로 design 을 오른쪽 클릭하여 Create HDL Wrapper 를 클릭한다.



56. Run Synthesis



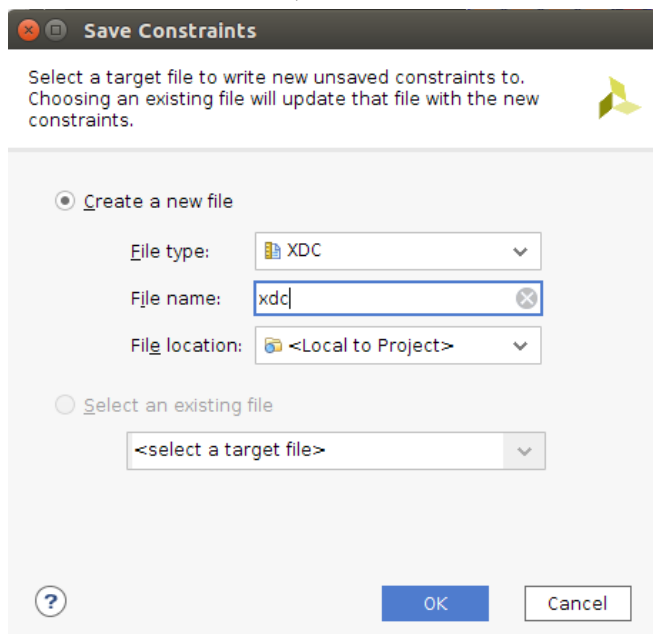
57. Run Implementation 을 바로 하지 않고, Open Synthesized Design 을 선택한다.



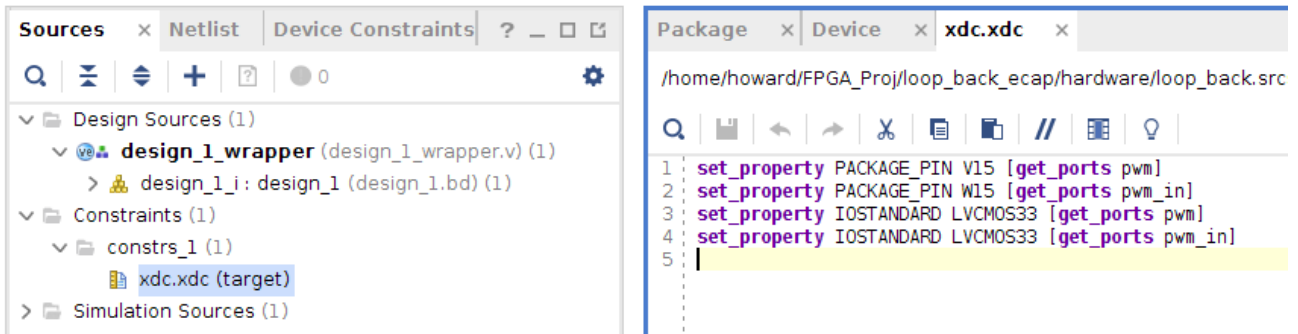
58. 아래와 같이 I/O PORT 설정을 해준다.

All ports (132)									
> DDR_54576 (71)									
	INOUT						✓	502	(Multiple)*
> FIXED_IO_54576 (59)									
	INOUT						✓	(Multiple)	(Multiple)*
Scalar ports (2)									
pwm	OUT				V15	✓	34	LVC MOS33*	
pwm_in	IN				W15	✓	34	LVC MOS33*	

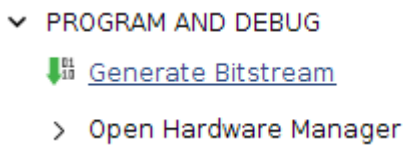
Ctrl + s 로 저장을 누르면, xdc 파일을 생성하라고 뜨는데, 이름은 임의로 정한다.



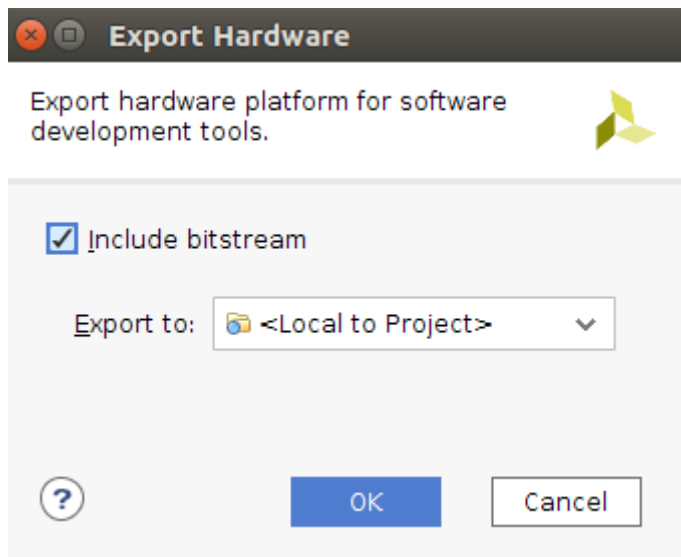
59.constraints 로 가서 xdc 파일에 포트 설정이 잘 되어있나 확인한다.



60.Generate Bitstream



61.File → Export → Export Hardware
include bitstream 옵션 체크



이제 하드웨어 구성은 끝났고, SDK 나 petalinux 를 이용하여 소프트웨어를 구현하면 된다.
본 예제에서는 SDK 를 통해 소프트웨어를 구현할 것이다.

61.File → Launch SDK

SDK 에서 New → Application Project

Project name 을 작성하고, Next 를 클릭한다.

Empty Application 을 선택하여 프로젝트를 생성한다.

New Project

Application Project

Create a managed make application project.

Project name:

☒ Use default location

Location:

Choose file system:

OS Platform:

Target Hardware

Hardware Platform:

Processor:

Target Software

Language: ☒ C ☐ C++

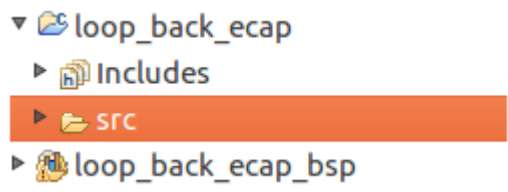
Compiler:

Hypervisor Guest:

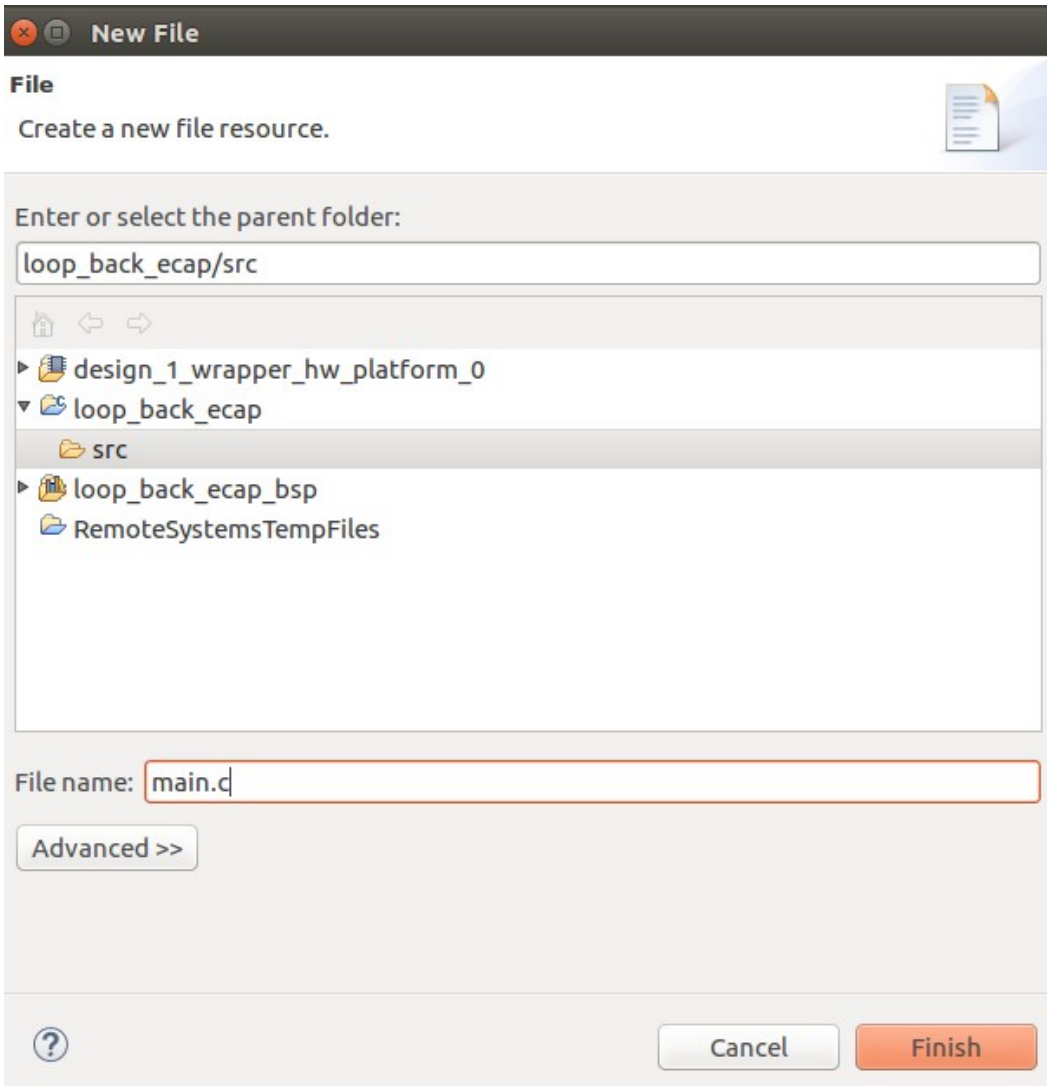
Board Support Package: ☒ Create New

☐ Use existing

62.생성된 프로젝트의 src 폴더를 오른쪽 클릭하여, New → File 을 선택한다.



파일 이름을 main.c 로 작성하여 파일을 생성한다.



63.빈 main.c 파일이 나오는 데, 코드 작성 전에 vivado 의 address editor 로 가서 아래와 같이 IP 들의 주소를 확인한다.

A screenshot of the Vivado Address Editor window. The title bar shows 'Diagram', 'Address Editor', and 'xdc.xdc'. The window contains a table with columns: Cell, Slave Interface, Base Name, Offset Address, Range, and High Address. The table lists two IP blocks: 'processing_system7_0' and its 'Data' block. The 'Data' block is expanded to show two registers: 'ECAP_0' and 'PWM_50Hz_0'.

Cell	Slave Interface	Base Name	Offset Address	Range	High Address
processing_system7_0					
Data (32 address bits : 0x40000000 [1 G])					
ECAP_0	S00_AXI	S00_AXI_reg	0x43C0_0000	64K	0x43C0_FFFF
PWM_50Hz_0	S00_AXI	S00_AXI_reg	0x43C1_0000	64K	0x43C1_FFFF

필자의 경우 ECAP = 0x43C00000, PWM_50Hz = 0x43C10000

64.아래와 같이 코드를 작성한다.

```
system.hdf system.mss main.c
#include <stdio.h>
#include "xil_printf.h"
#include "xparameters.h"
#include "xil_io.h"
#include "xbasic_types.h"

Xuint32 ECAP_ADDR    = 0x43C00000;
Xuint32 PWM_ADDR     = 0x43C10000;
int main(void)
{
    int num = 100000, i;

    while(1)
    {
        Xil_Out32(PWM_ADDR, num);
        if(num == 200000)
            num = 100000;
        else
            num += 100;

        xil_printf("period = %d, duty = %d\r\n",*((Xuint32*)ECAP_ADDR + 0),*((Xuint32*)ECAP_ADDR + 1));

        for(i=0;i<300000;i++)
            ;
    }
    return 0;
}
```

```
#include <stdio.h>
#include "xil_printf.h"
#include "xparameters.h"
#include "xil_io.h"
#include "xbasic_types.h"

Xuint32 ECAP_ADDR    = 0x43C00000;
Xuint32 PWM_ADDR     = 0x43C10000;
int main(void)
{
    int num = 100000, i;

    while(1)
    {
        Xil_Out32(PWM_ADDR, num);
        if(num == 200000)
            num = 100000;
        else
            num += 100;

        xil_printf("period = %d, duty = %d\r\n",*((Xuint32*)ECAP_ADDR +
0),*((Xuint32*)ECAP_ADDR + 1));

        for(i=0;i<300000;i++)
            ;
    }
    return 0;
}
```

65.상단의 Program FPGA 버튼을 클릭한다. 아래와 같이 생겼다.



66.프로젝트 폴더(필자의 경우 loop_back_ecap)을 우클릭 하여 Run as → Launch on Hardware(GDB)를 클릭한다.



67.아래와 같이 시리얼 프로그램을 실행한다.

```
howard@Howard:~/FPGA_Proj/loop_back_ecap$ sudo chmod 666 /dev/ttyUSB1
[sudo] password for howard:
howard@Howard:~/FPGA_Proj/loop_back_ecap$ putty
```

68.Zybo 보드의 V15(JC1)와 W15(JC2)를 연결해주면, pwm 의 주기와 듀티가 잘 측정되는 것을 볼 수 있다.

```
/dev/ttyUSB1 - PuTTY
period = 2000001, duty = 168700
period = 2000001, duty = 168700
period = 2000001, duty = 168700
period = 2000001, duty = 169200
period = 2000001, duty = 169200
period = 2000001, duty = 169200
period = 2000001, duty = 169200
period = 2000001, duty = 169200
period = 2000001, duty = 169700
period = 2000001, duty = 169700
period = 2000001, duty = 169700
period = 2000001, duty = 169700
period = 2000001, duty = 169700
period = 2000001, duty = 169700
period = 2000001, duty = 170200
period = 2000001, duty = 170200
period = 2000001, duty = 170200
period = 2000001, duty = 170200
period = 2000001, duty = 170200
period = 2000001, duty = 170700
period = 2000001, duty = 170700
period = 2000001, duty = 169700
period = 2000001, duty = 170200
period = 2000001, duty = 170200
```