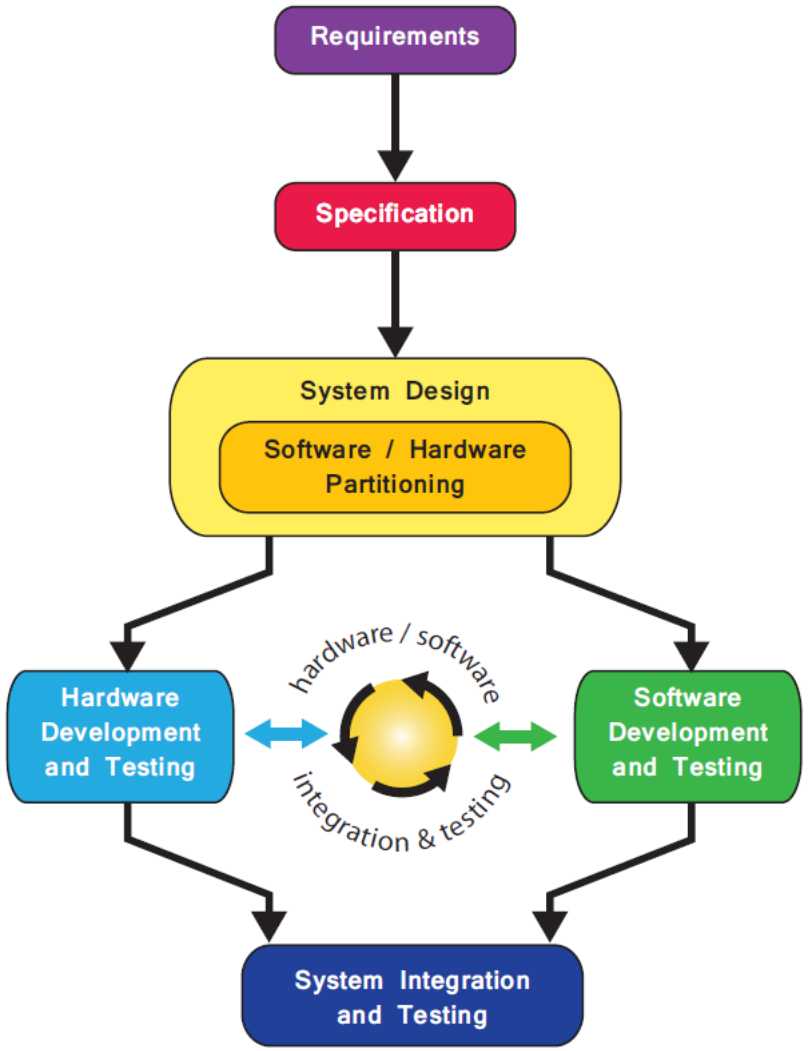
Zynq SoC 설계

Xilinx VIVADO

VHDL/Verilog
HLS(SDSoC)

AXI HW IP



Xilinx SDK

Xilinx, ARM
라이브러리

C/C++ SW IP

Bare Metal App
freeRTOS, Linux

HUINS
Human Intelligent System

Zynq를 활용한 SoC / FPGA 설계

HDL 및 HLS를 이용한 설계 이해 및 실습

HUINS
Human Intelligent System

ZYNQ를 이용한 설계 실습(HDL)

CONTENTS

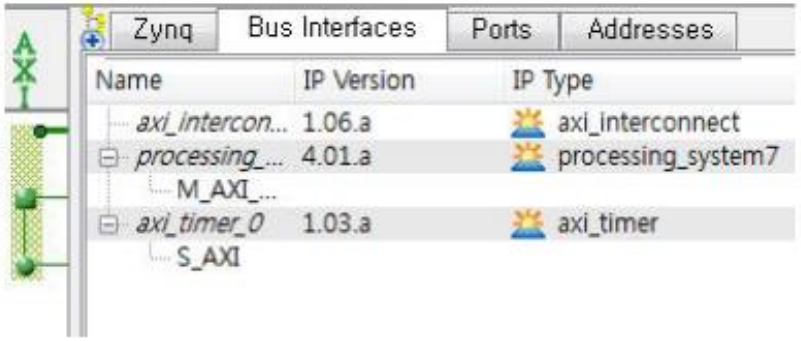
HUINS
Human Intelligent System

Vivado

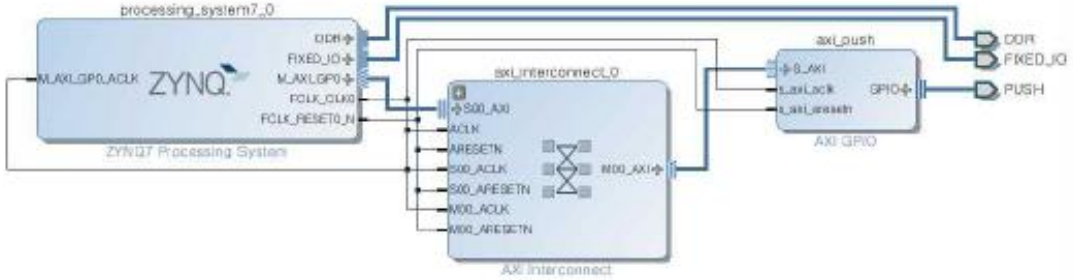
- Xilinx에서 개발한 개발 도구
- ISE 대체
- 7-Series 부터 지원(Spartan7, Artix7, Kintex7, Virtex7, Zynq7000)
- IP 재사용(IP Integrator → 사용, 관리, 생성) 디자인
- GUI 기반 블록 디자인

ISE – Vivado 주요 차이점

	ISE	Vivado
Simulator	ISIM	XSIM
Debug	Chipscope analyzer	Vivado(Integrated Chipscope)
Design	RTL + XPS	RTL + Block Design



< PlanAhead XPS design >

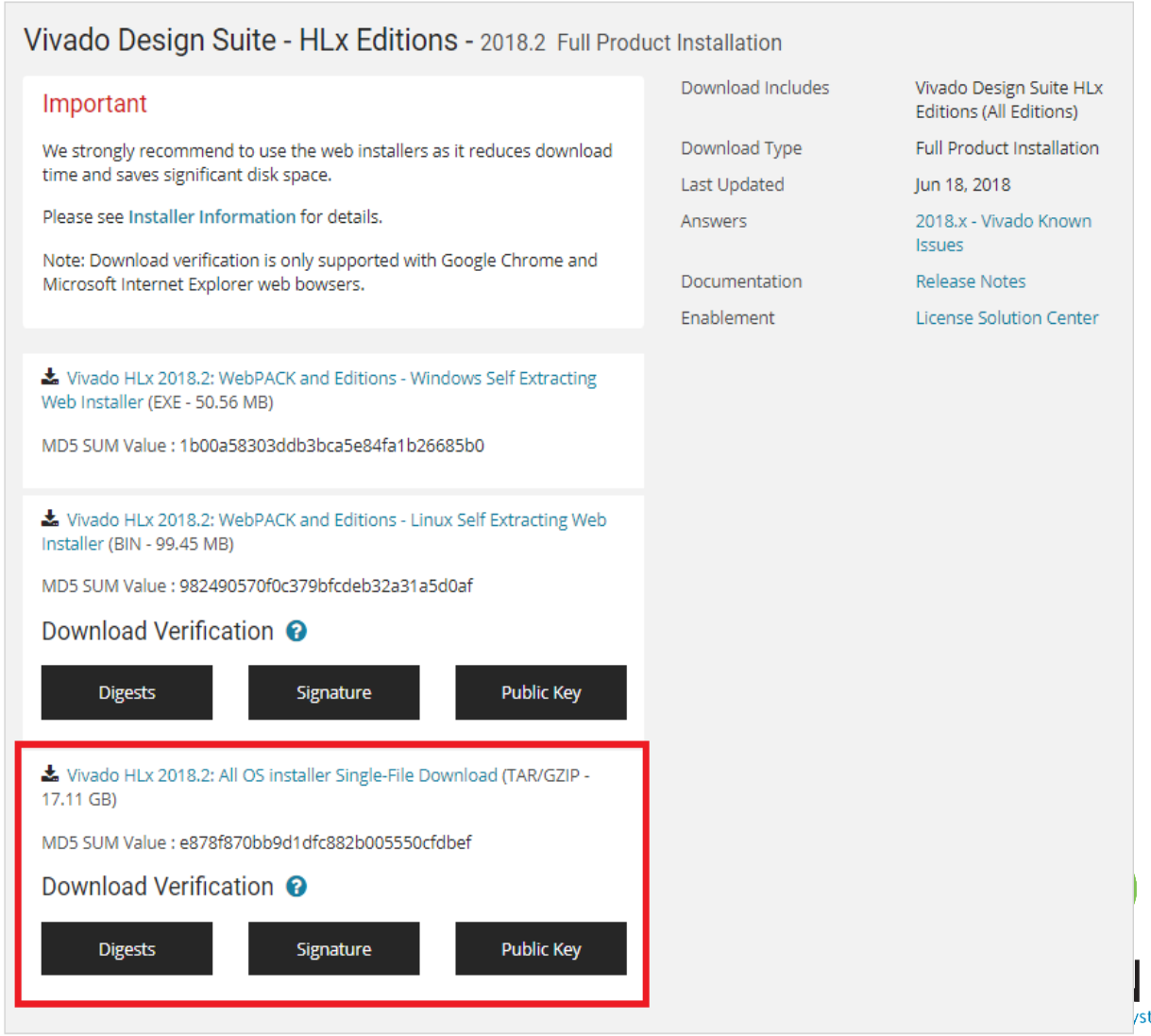


< Vivado Block design >

HUINS
Human Intelligent System

Vivado 다운로드/설치

자일링스 홈페이지 (www.Xilinx.com)에서 Vivado 설치 파일을 다운로드 받는다.



Vivado 다운로드/설치

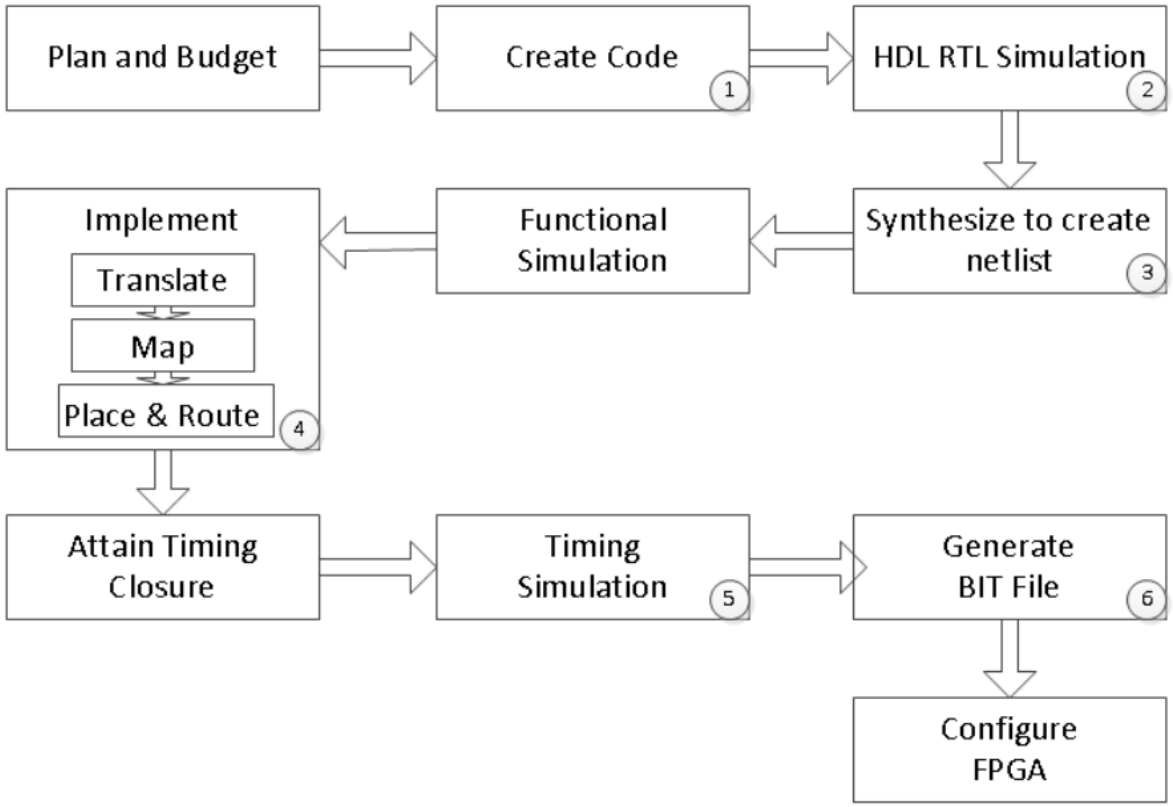
WebPack 에디션 설치

SDK 및 SoC Device,

Cable Driver 선택 및 설치

Vivado Tutorial

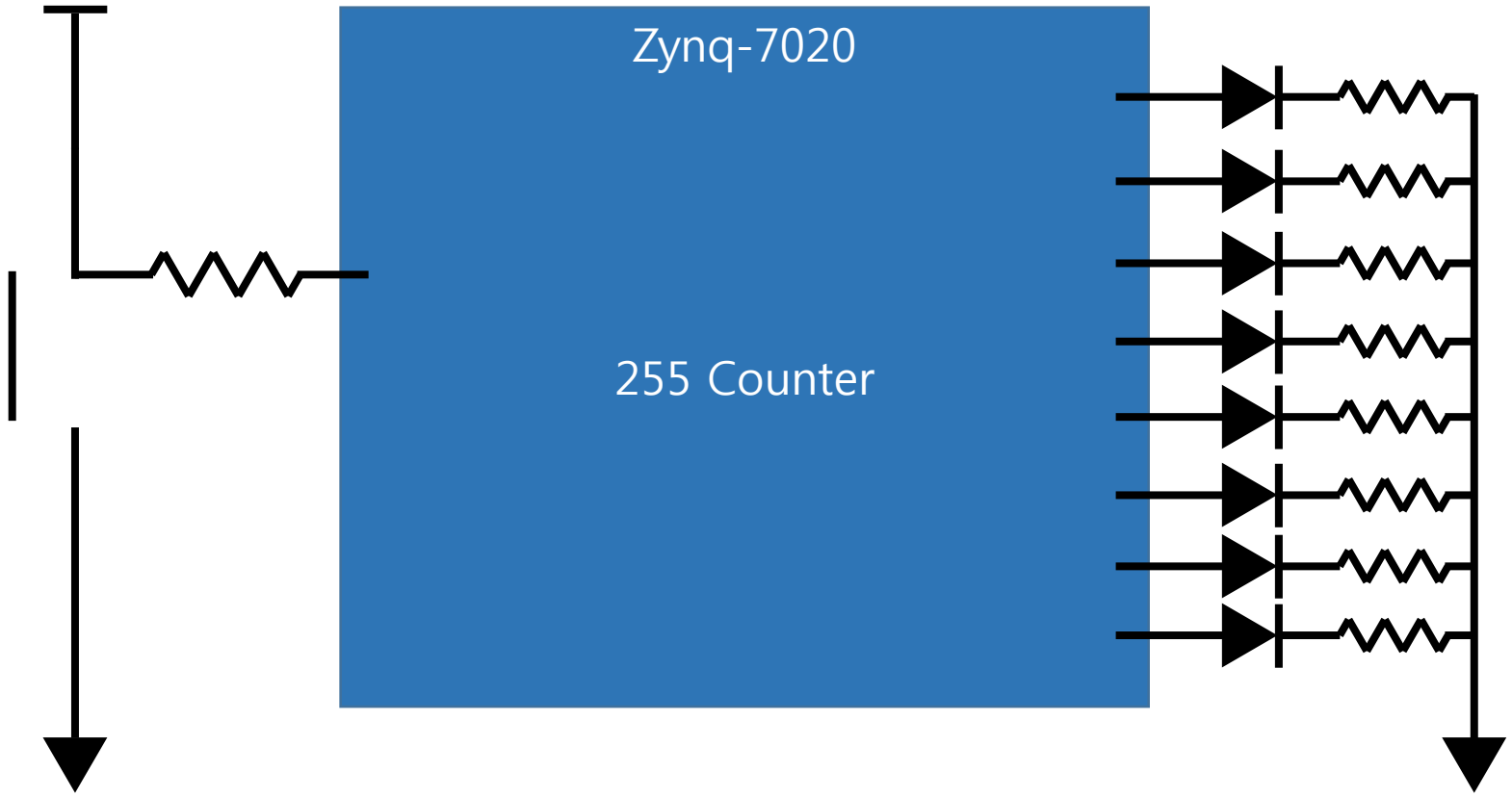
- 일반적인 FPGA 프로젝트(ISE)
  - 프로젝트 계획 수립
  - 소스코드 생성
  - 시뮬레이션
  - 합성 / 구현
  - 타이밍 시뮬레이션
  - BIT 파일 생성



HUINS
Human Intelligent System

Design Description

- 스위치 입력에 따라 0-255 카운터 값 LED에 출력

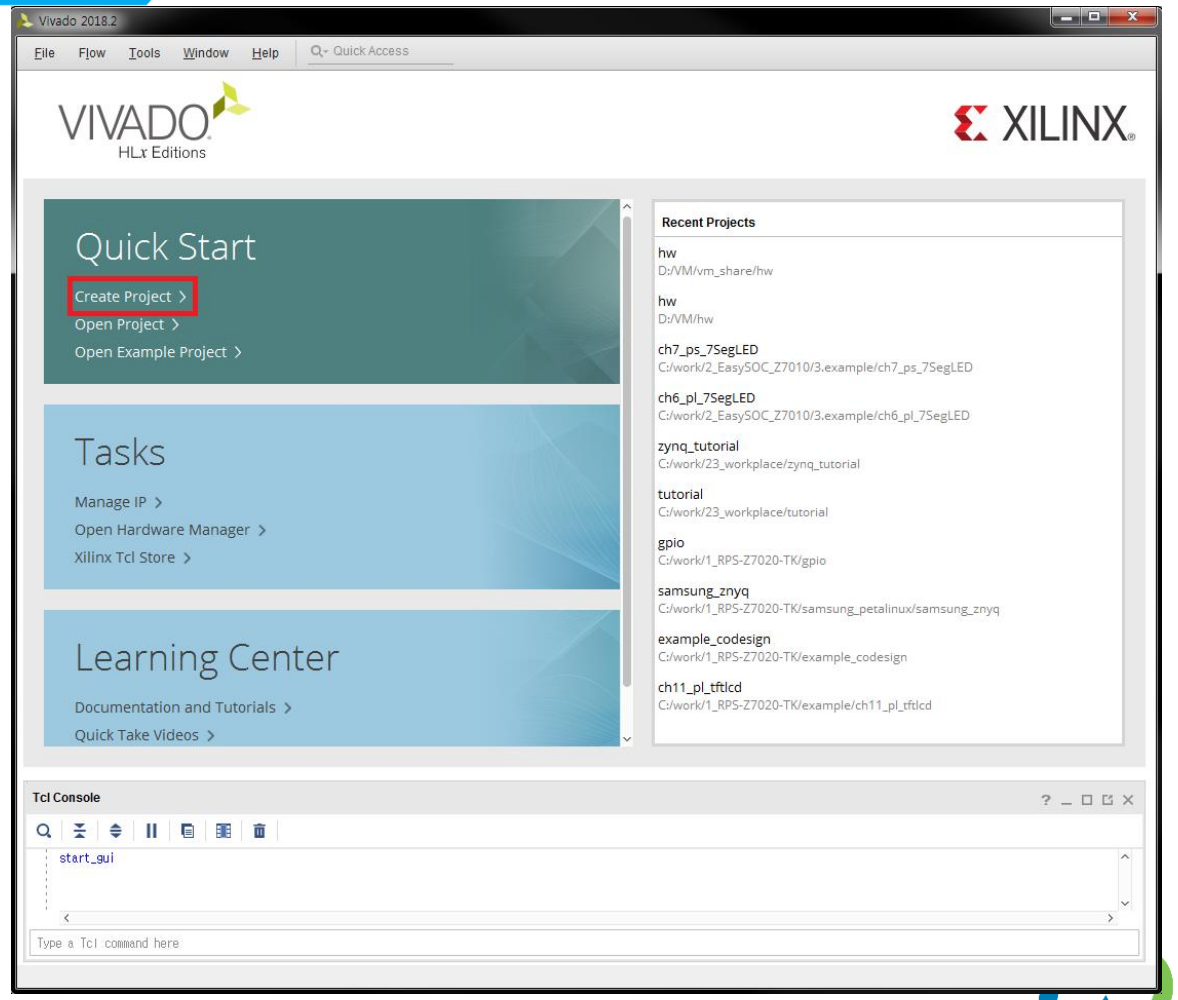


HUINS
Human Intelligent System

프로젝트 생성

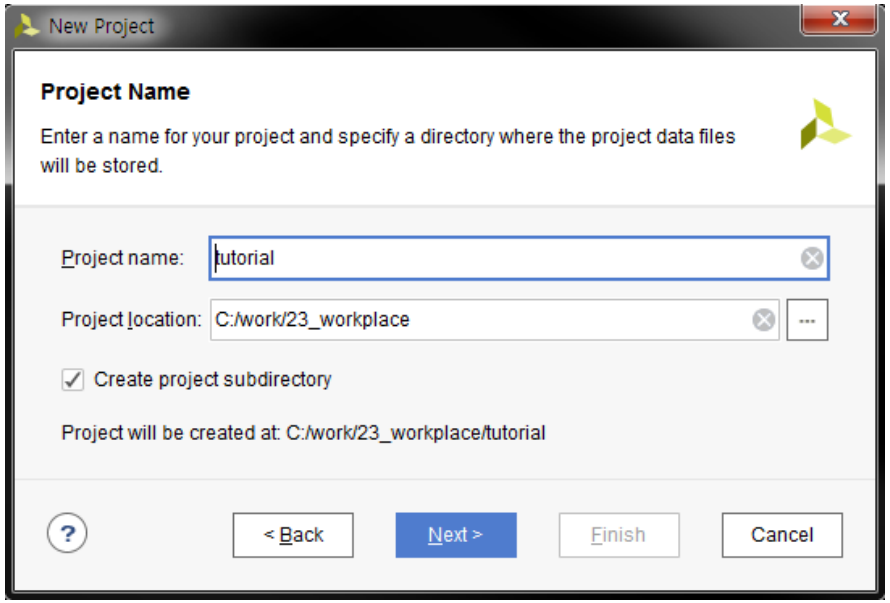
- Vivado를 실행하여 프로젝트를 생성한다.

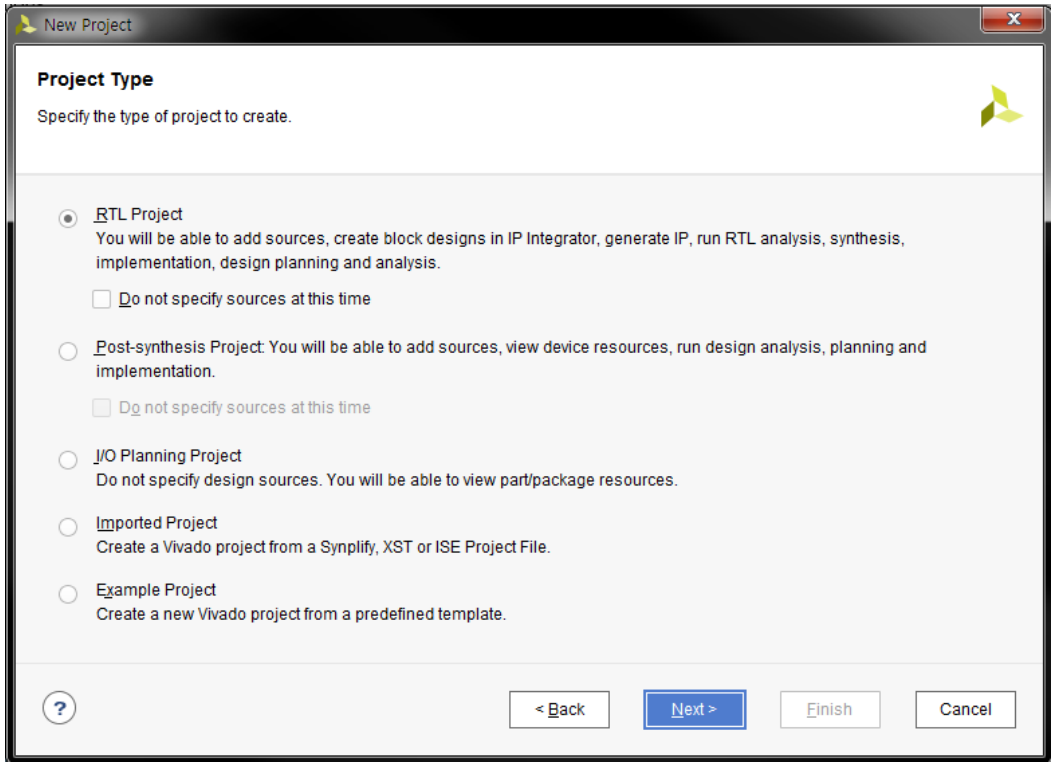
- File - Project – New
- Quick Start - Create Project 실행



HUINS
Human Intelligent System

프로젝트 생성



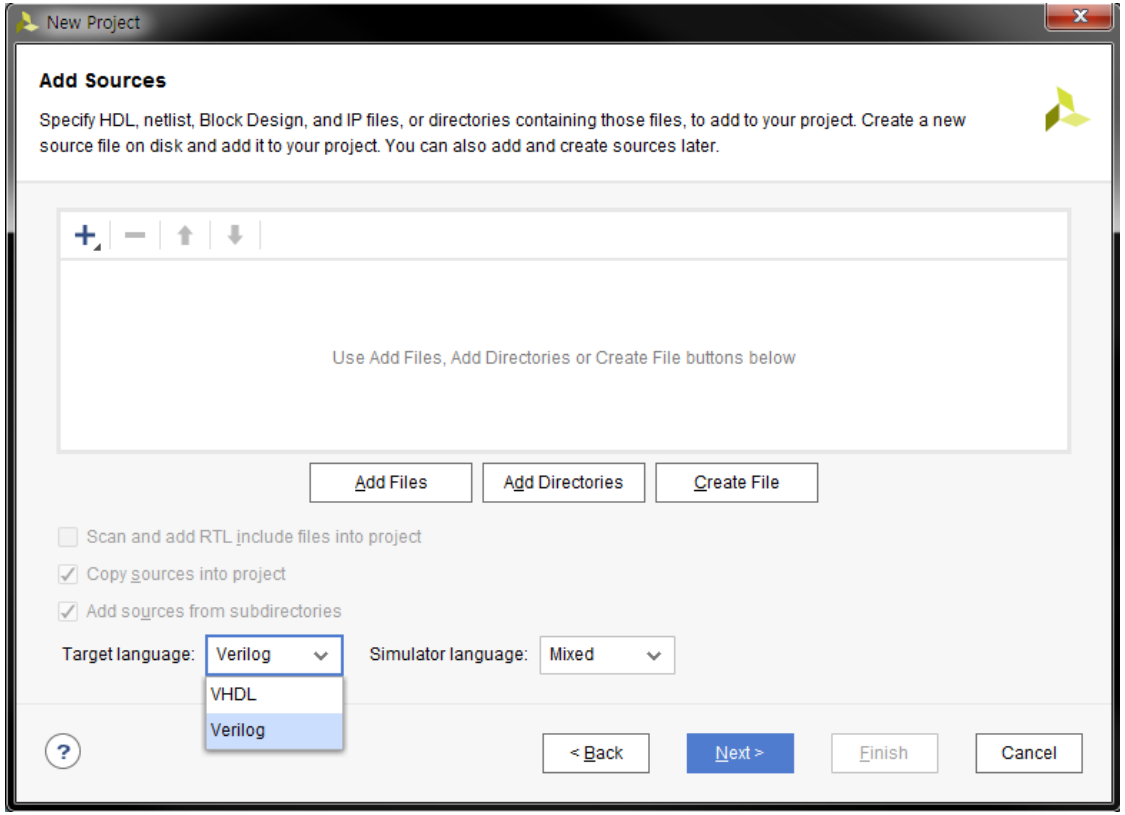


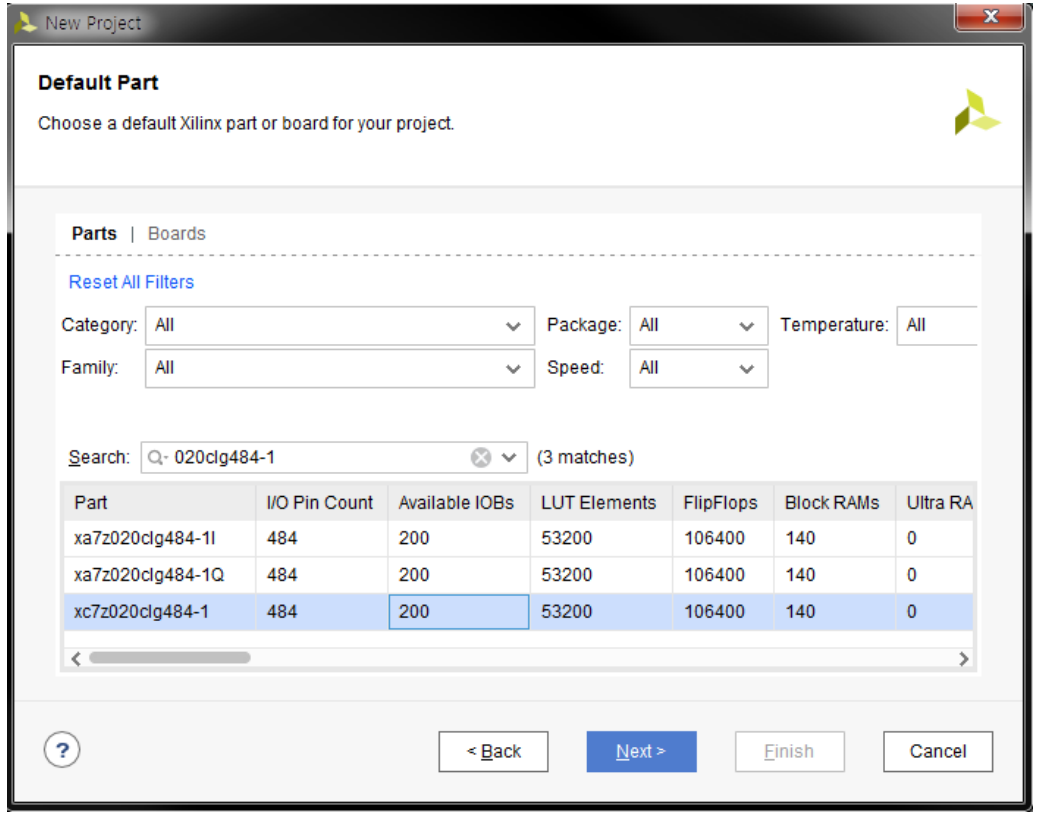
프로젝트 이름, 경로 지정

프로젝트 타입 지정 → RTL

HUINS
Human Intelligent System

프로젝트 생성





소스 추가(파일, 폴더, 생성)
default Language 선택

사용 FPGA/SoC 파트 선택
xc7z020clg484-1

HUINS
Human Intelligent System

Vivado 실행 창

소스

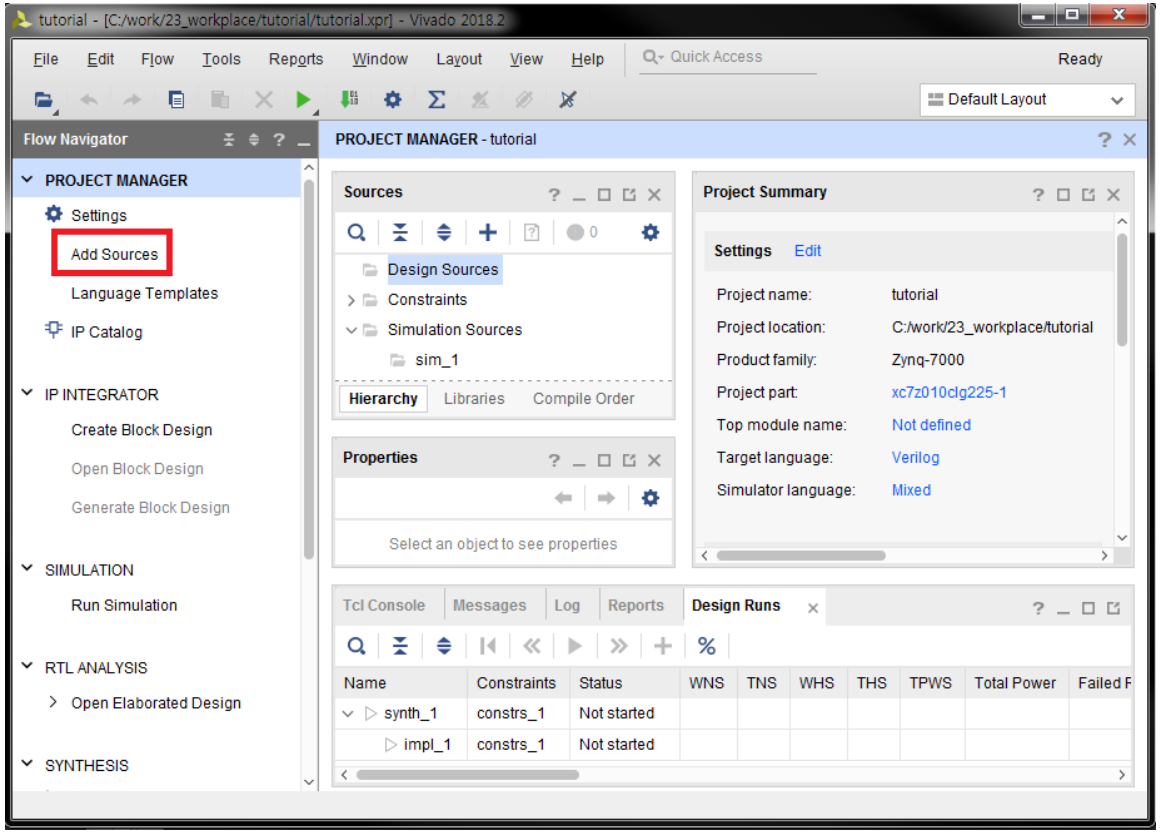
Flow Navigator
- 설계 흐름 순서

속성

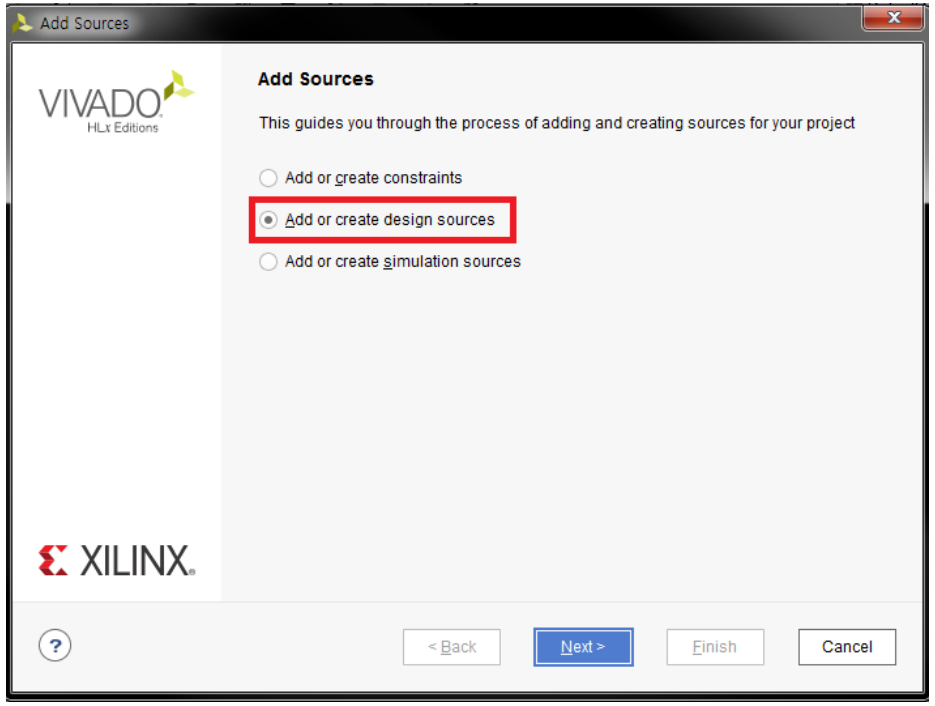
작업 창
Workspace

결과 창
Results Windows Area
실행 로그 및 에러 출력

소스 추가



소스 추가(Add Sources)

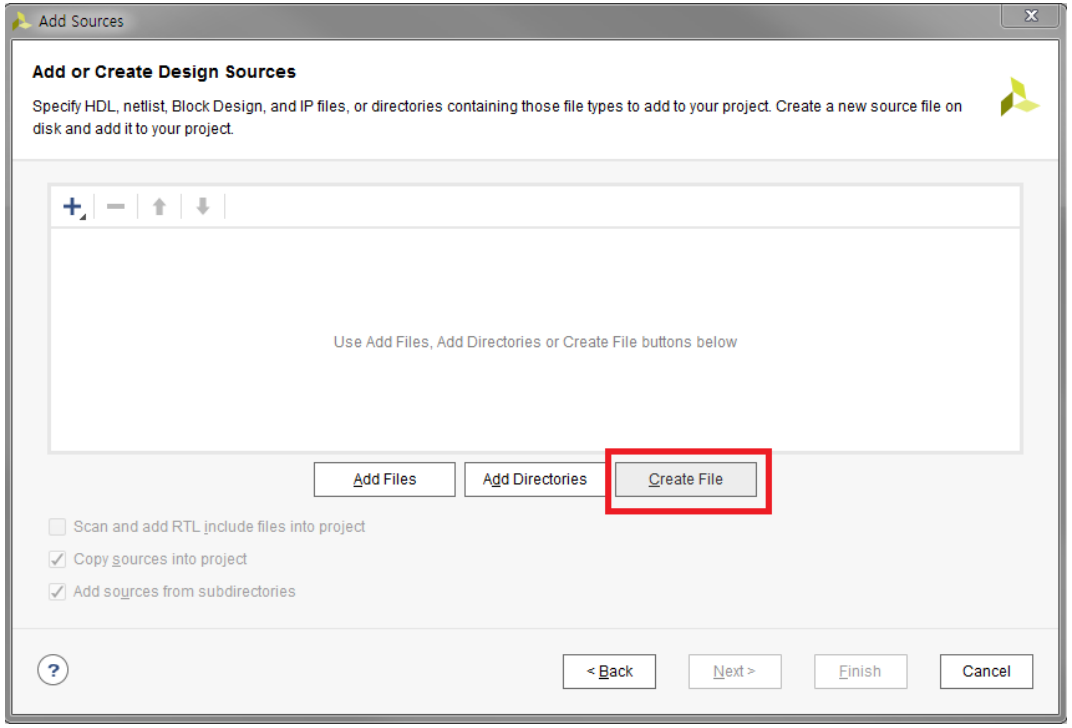


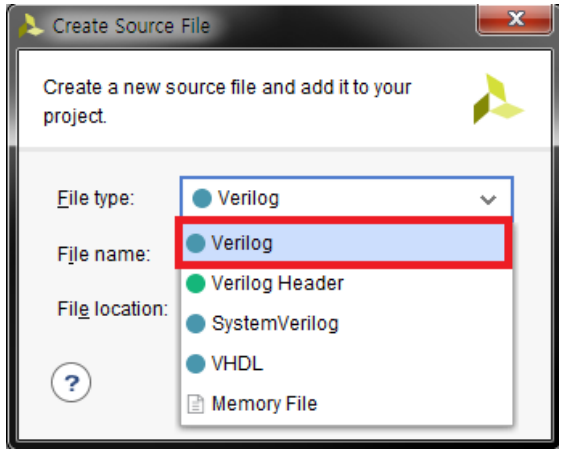
Constraints → Timing, Clock, Pin map

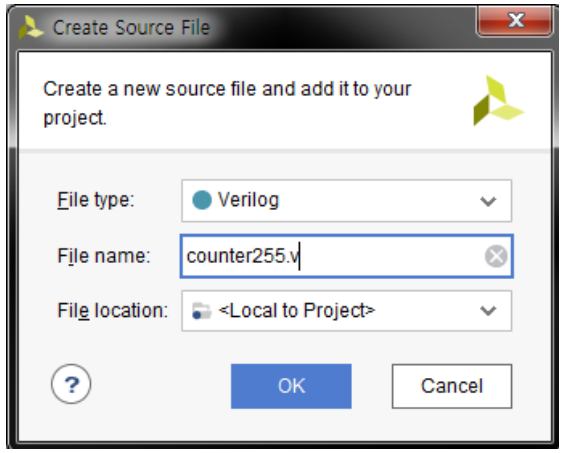
Design Source → HDL, Block Design, IP

Simulation → Test Bench

소스 추가







Add → 파일, 디렉토리 추가

Create File → 소스 파일 생성(counter255.v)

Copy Sources into project → 소스코드 사본 추가

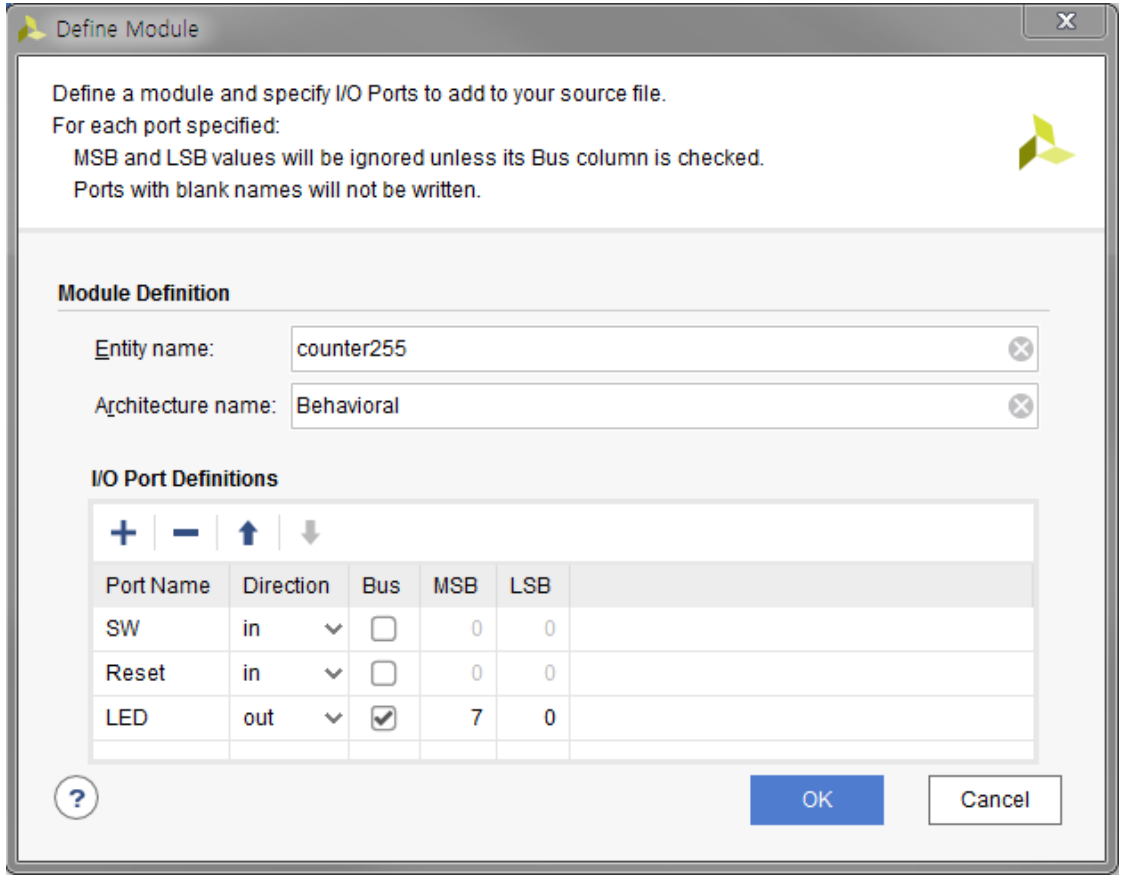
HUINS
Human Intelligent System

소스 추가

모듈 이름 및 I/O 설정

- input: SW, Reset
- output: [7:0] LED

미 설정시 빈 모듈 생성



HUINS
Human Intelligent System

소스 코드 작성(counter)

스위치 입력 시(posedge)
- 레지스터 값(LED) 1 증가
리셋 입력 시(Negative Reset)
- 0으로 초기화

0-255 값 LED 출력

Project Summary | counter255.v

C:/work/10_SDx/tutorial/tutorial.srcs/sources_1/new/counter255.v

```
1   `timescale 1ns / 1ps
2   //////////////////////////////////////////////////////////////////////////////////
21
22
23  module counter255(
24      input SW,
25      input Reset,
26      output reg [7:0] LED
27  );
28
29  always@(posedge SW) begin
30      if(!Reset)
31          LED <= 0;
32      else
33          LED <= LED + 1;
34  end
35
36  endmodule
```

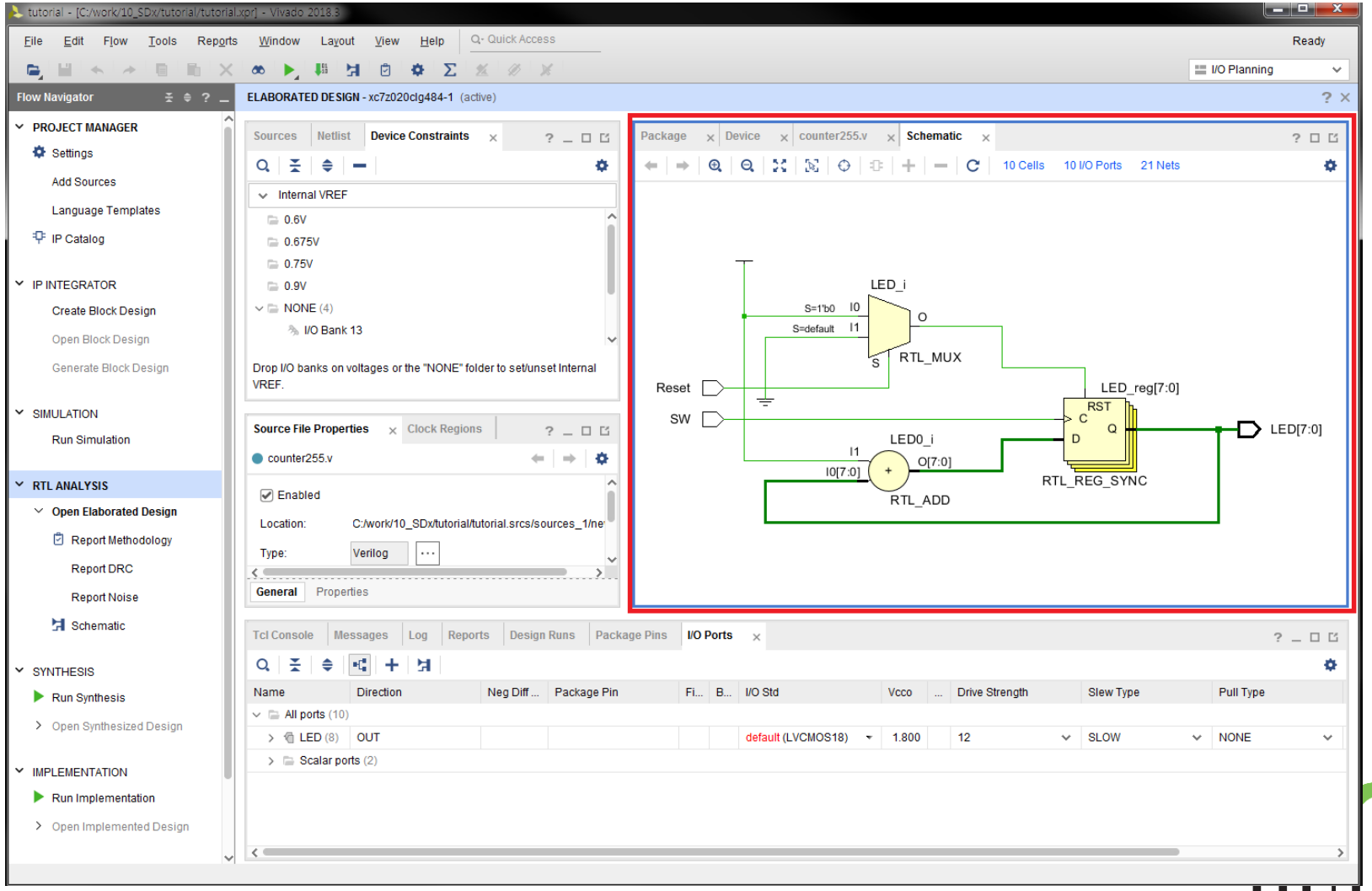

HUINS
Human Intelligent System

RTL Analysis

RTL ANALYSIS – Schematic

RTL 소스 코드
→ Schematic 표현

REG 값 LED 출력
SW 입력시 1 ADD
Reset 시 초기화



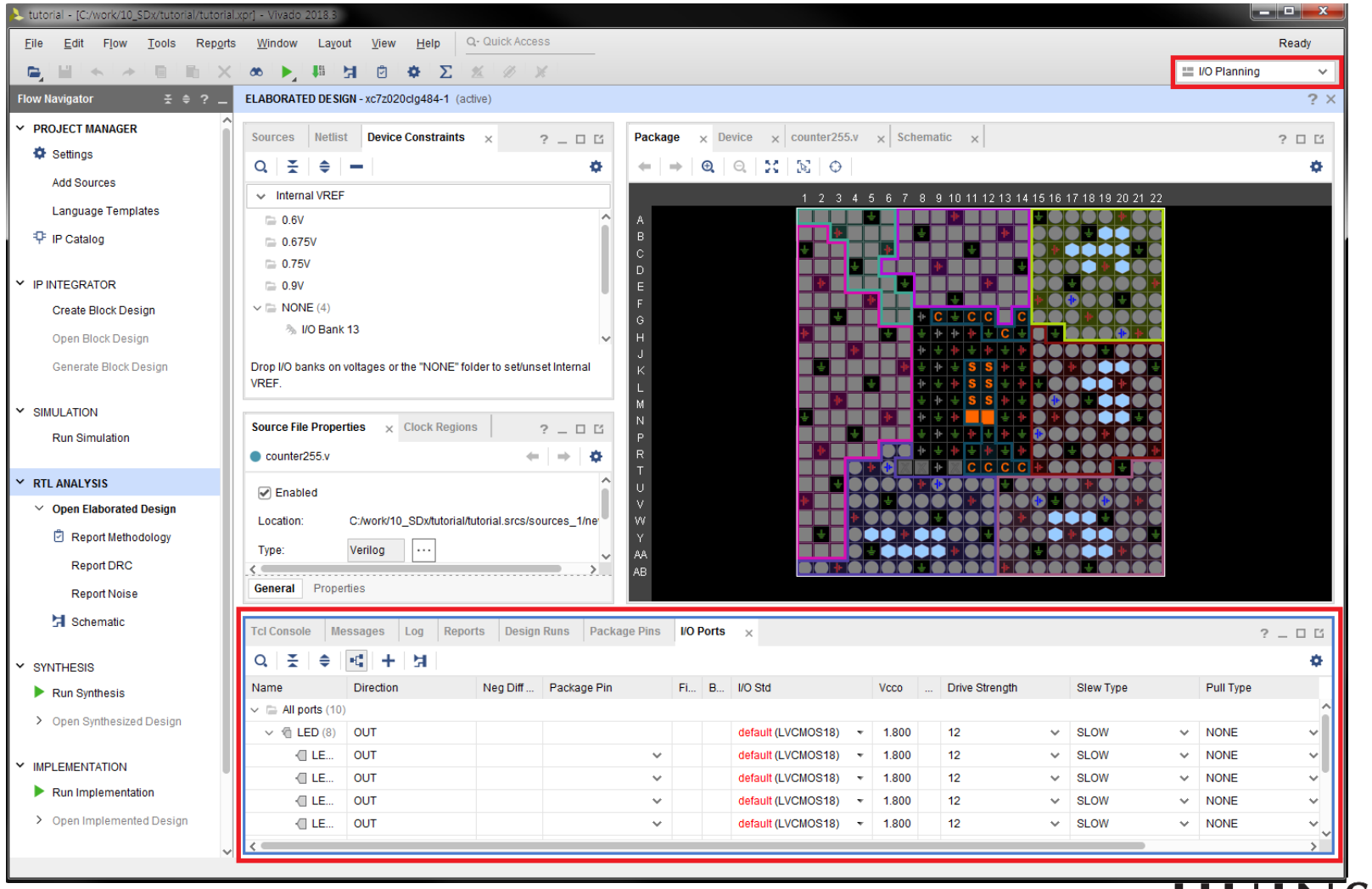
HUINS
Human Intelligent System

I/O Planning

Package 탭

I/O의 핀 맵핑 설정

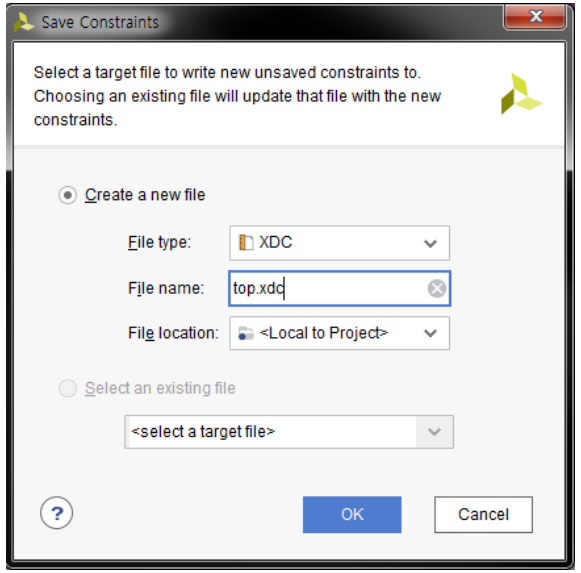
T16	LED[7]	LVCMOS3.3V
T17	LED[6]	LVCMOS3.3V
R19	LED[5]	LVCMOS3.3V
T19	LED[4]	LVCMOS3.3V
R18	LED[3]	LVCMOS3.3V
T18	LED[2]	LVCMOS3.3V
P16	LED[1]	LVCMOS3.3V
R16	LED[0]	LVCMOS3.3V
AA18	SW	LVCMOS3.3V
Y18	Reset	LVCMOS3.3V



Constraints

저장 시 constraints 생성

Add Sources → xdc 파일 생성, 추가 또한 가능



top.xdc | counter255.v

C:/work/10_SDx/tutorial/tutorial.srcs/constrs_1/new/top.xdc

```
1  set_property PACKAGE_PIN T16 [get_ports {LED[7]}]
2  set_property PACKAGE_PIN T17 [get_ports {LED[6]}]
3  set_property PACKAGE_PIN R19 [get_ports {LED[5]}]
4  set_property PACKAGE_PIN T19 [get_ports {LED[4]}]
5  set_property PACKAGE_PIN R18 [get_ports {LED[3]}]
6  set_property PACKAGE_PIN T18 [get_ports {LED[2]}]
7  set_property PACKAGE_PIN P16 [get_ports {LED[1]}]
8  set_property PACKAGE_PIN R16 [get_ports {LED[0]}]
9  set_property PACKAGE_PIN Y18 [get_ports Reset]
10 set_property PACKAGE_PIN AA18 [get_ports SW]
11 set_property IOSTANDARD LVCMOS33 [get_ports {LED[7]}]
12 set_property IOSTANDARD LVCMOS33 [get_ports {LED[6]}]
13 set_property IOSTANDARD LVCMOS33 [get_ports {LED[5]}]
14 set_property IOSTANDARD LVCMOS33 [get_ports {LED[4]}]
15 set_property IOSTANDARD LVCMOS33 [get_ports {LED[3]}]
16 set_property IOSTANDARD LVCMOS33 [get_ports {LED[2]}]
17 set_property IOSTANDARD LVCMOS33 [get_ports {LED[1]}]
18 set_property IOSTANDARD LVCMOS33 [get_ports {LED[0]}]
19 set_property IOSTANDARD LVCMOS33 [get_ports Reset]
20 set_property IOSTANDARD LVCMOS33 [get_ports SW]
21
22 set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets SW]
```

SW 입력이 외부 클럭과 같이 작동하므로 내용 추가

set_property CLOCK_DEDICATED_ROUTE FALSE [get_nets SW]

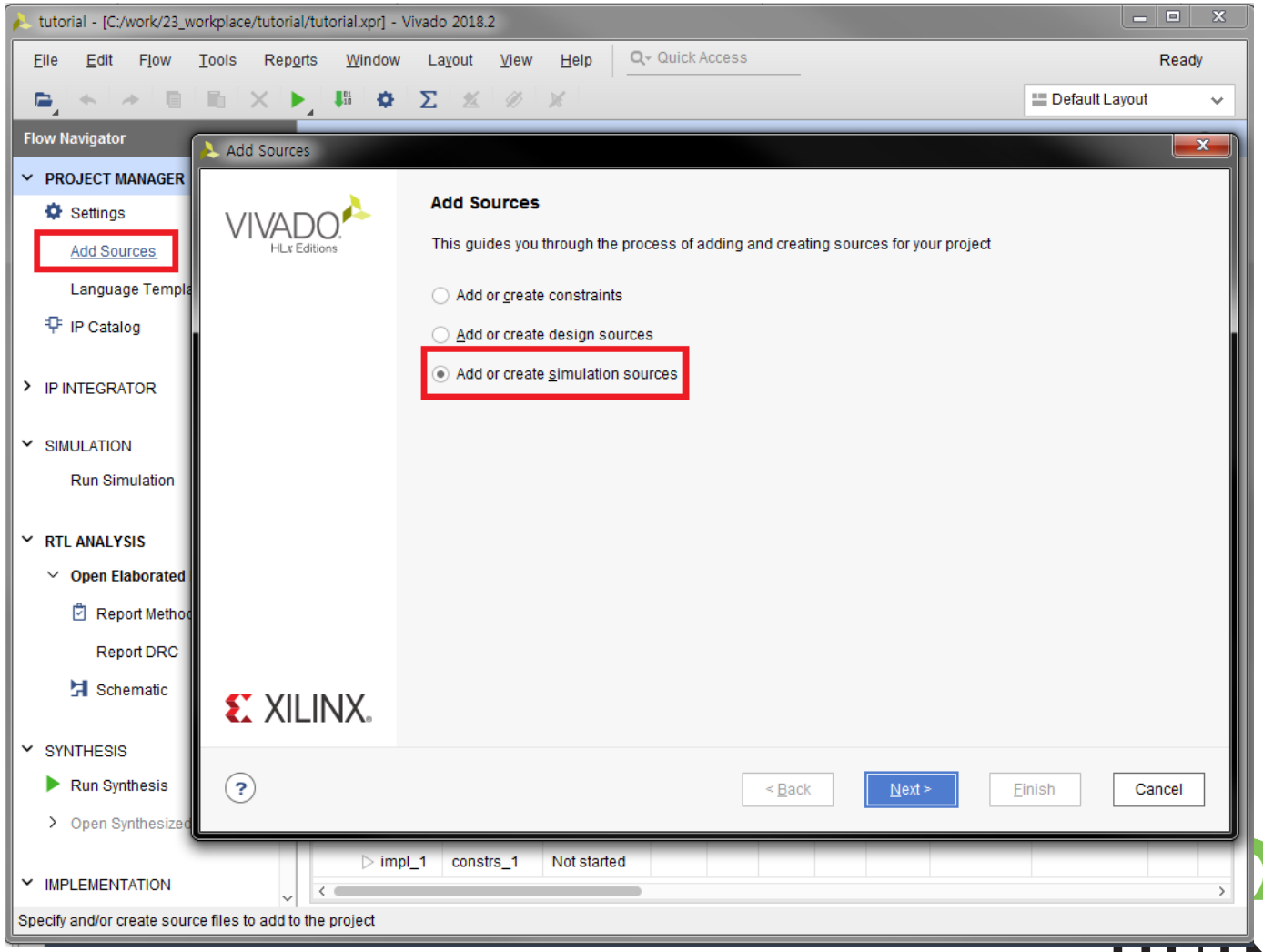
HUINS
Human Intelligent System

Simulation

Add Sources

- Simulation Sources

시뮬레이션 소스 추가



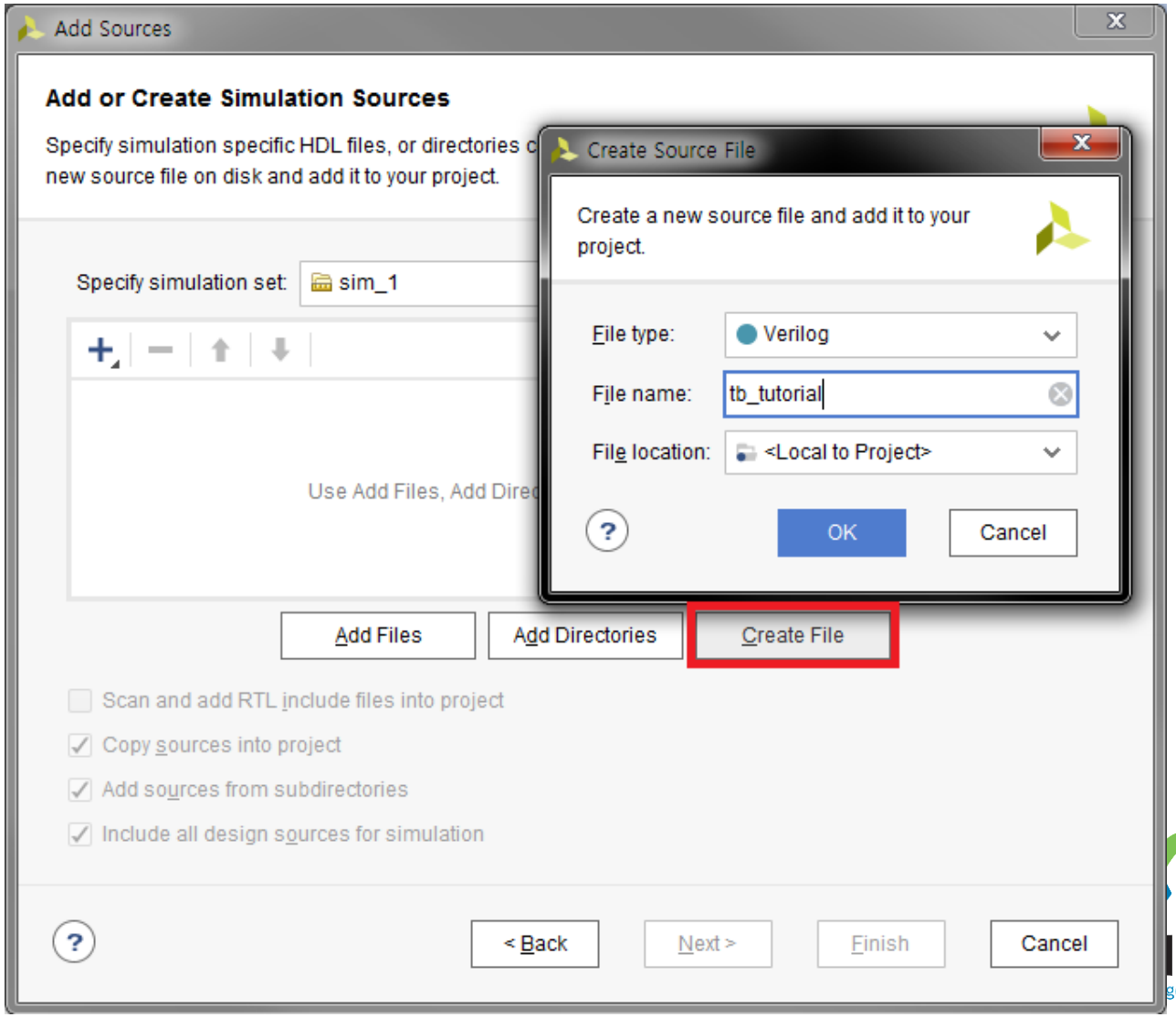
HUINS
Human Intelligent System

Simulation

시뮬레이션 소스 생성

File type: Verilog

File name: tb_tutorial



Simulation Source

시뮬레이션 소스코드 작성

SW 입력 0, 1 toggle(clk)
Reset = 0 일 때 0으로 초기화
LED 값 출력

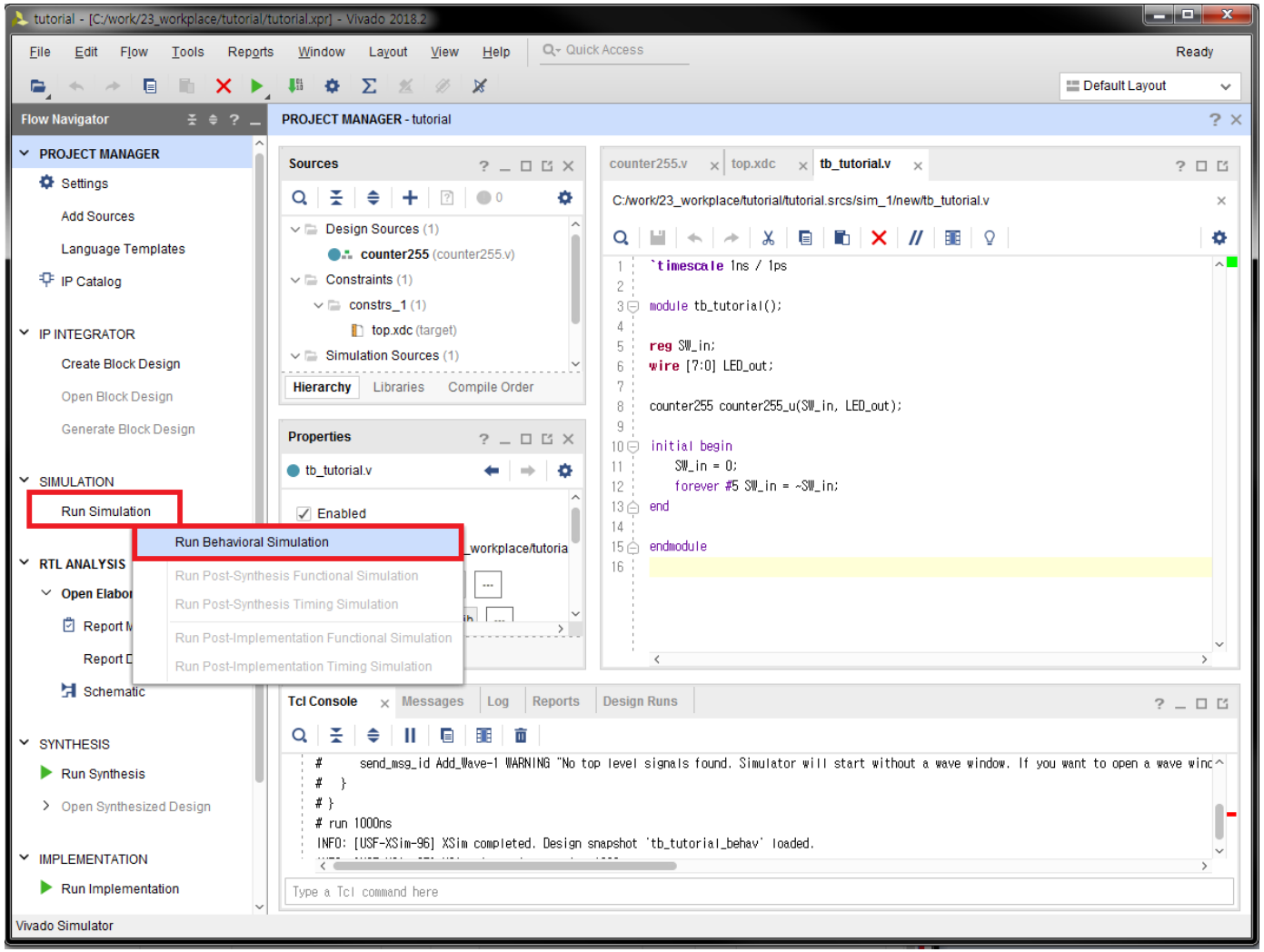
```
1   `timescale 1ns / 1ps
2   ////////////////////////////////////////////////////////////////////////////////
21
22
23  module tb_tutorial();
24
25  reg SW, Reset;
26  wire [7:0] LED;
27
28  counter255 inst_counter255(SW, Reset, LED);
29
30  initial begin
31  SW = 0;
32  forever #5 SW=~SW; //SW Toggle
33  end
34
35  initial begin
36  Reset=0;
37  #20;
38  Reset=1;
39  #100;
40  Reset=0;
41  #20;
42  Reset=1;
43  end
44  endmodule
```

Simulation

SIMULATION

-Run Behavioral Simulation

시뮬레이션을 실행한다



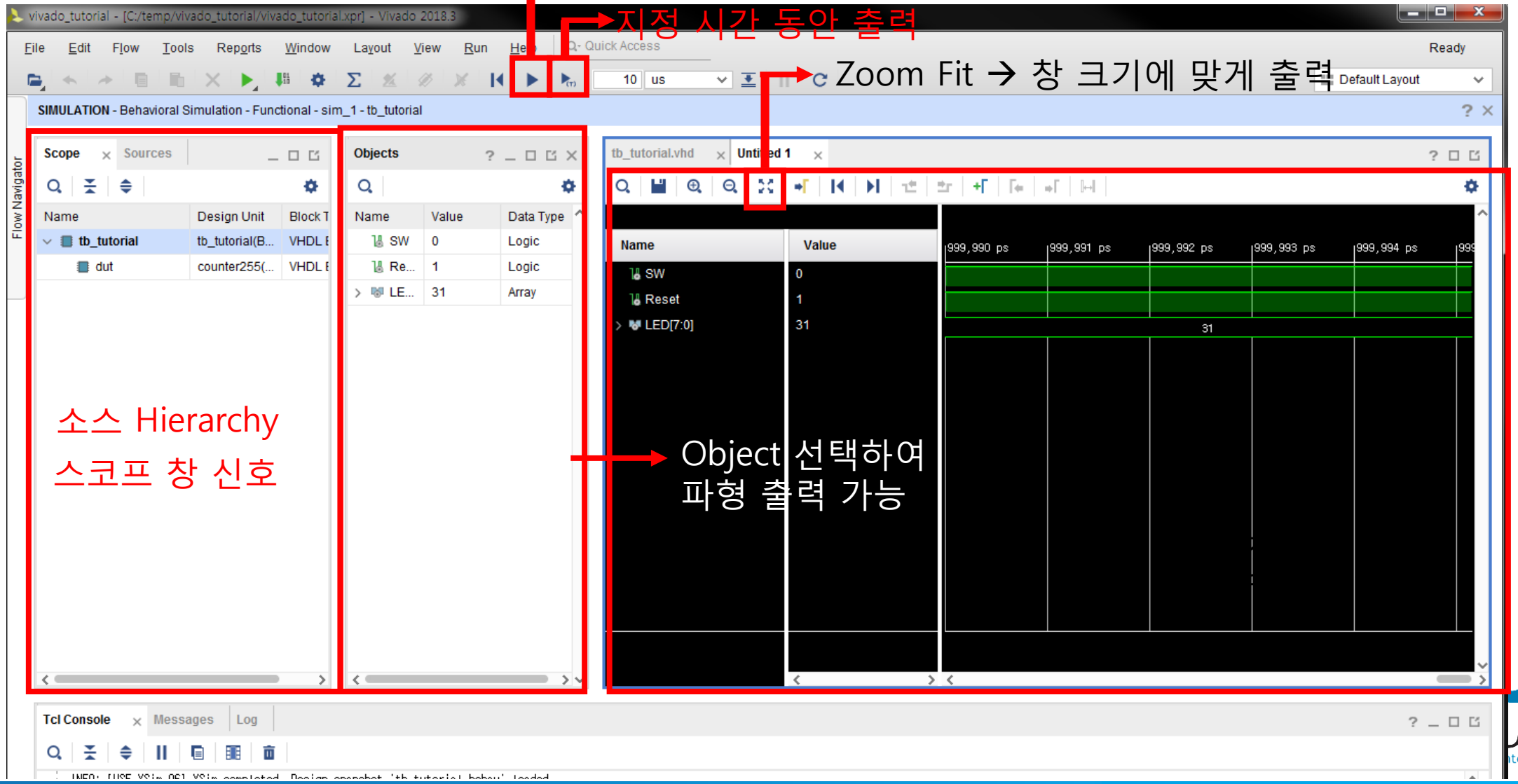
HUINS
Human Intelligent System

Simulation Window

Run All(end 까지 출력)

지정 시간 동안 출력

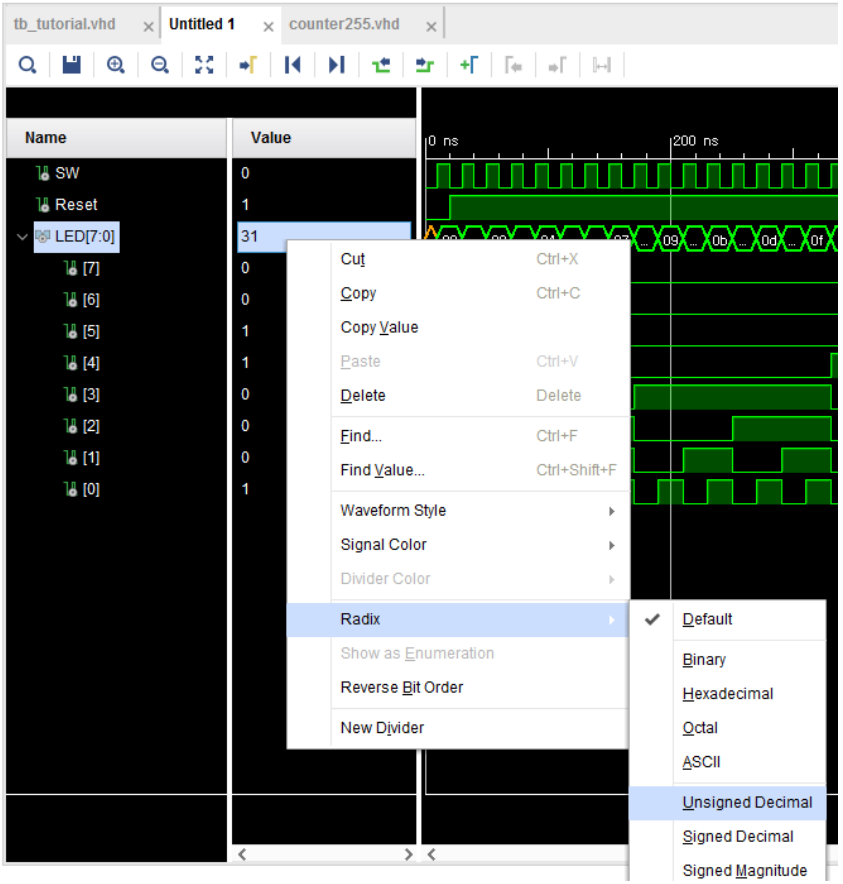
Zoom Fit → 창 크기에 맞게 출력

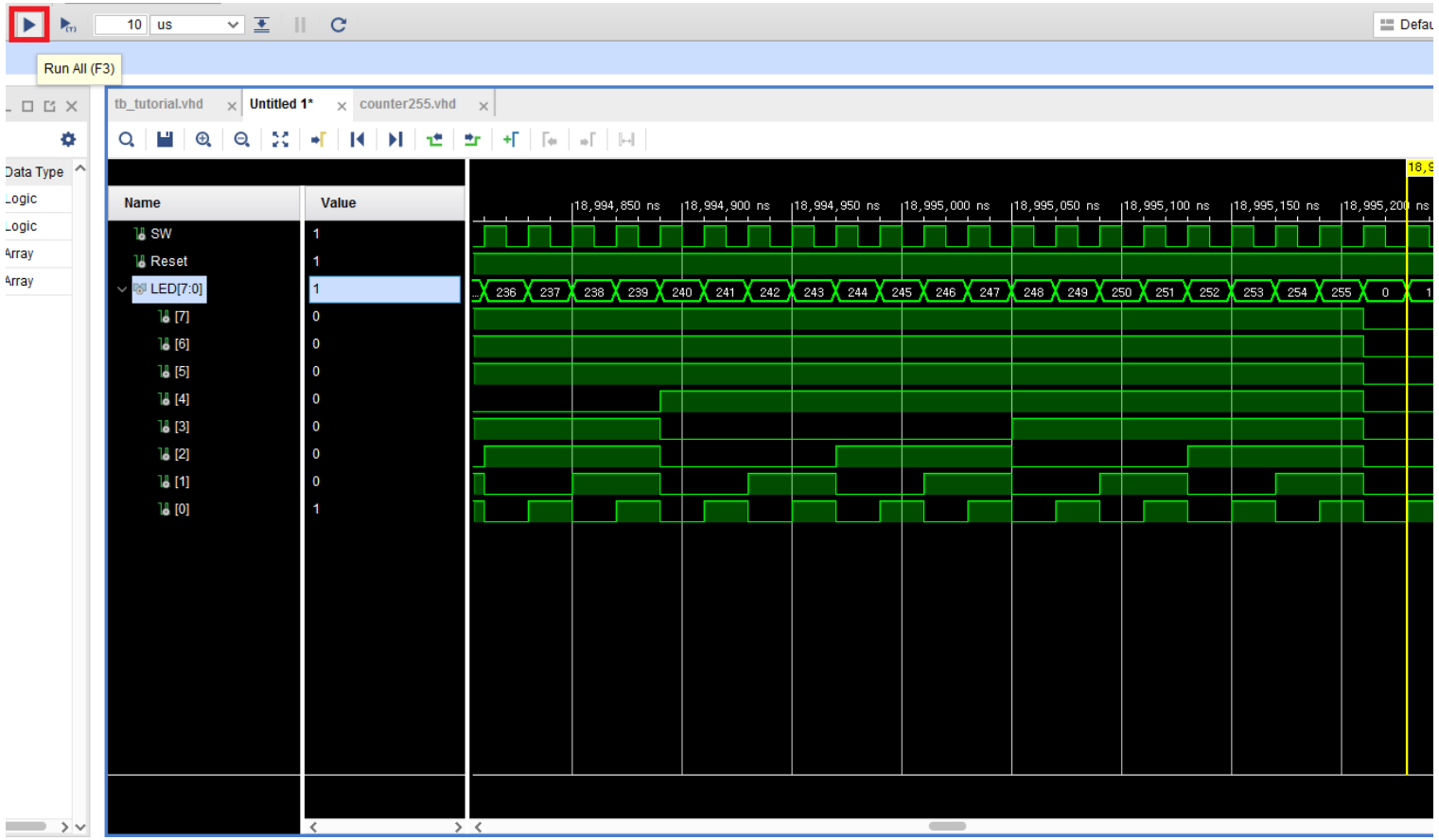


소스 Hierarchy
스코프 창 신호

Object 선택하여
파형 출력 가능

Simulation 출력 파형 확인





출력 파형 추가, 제거

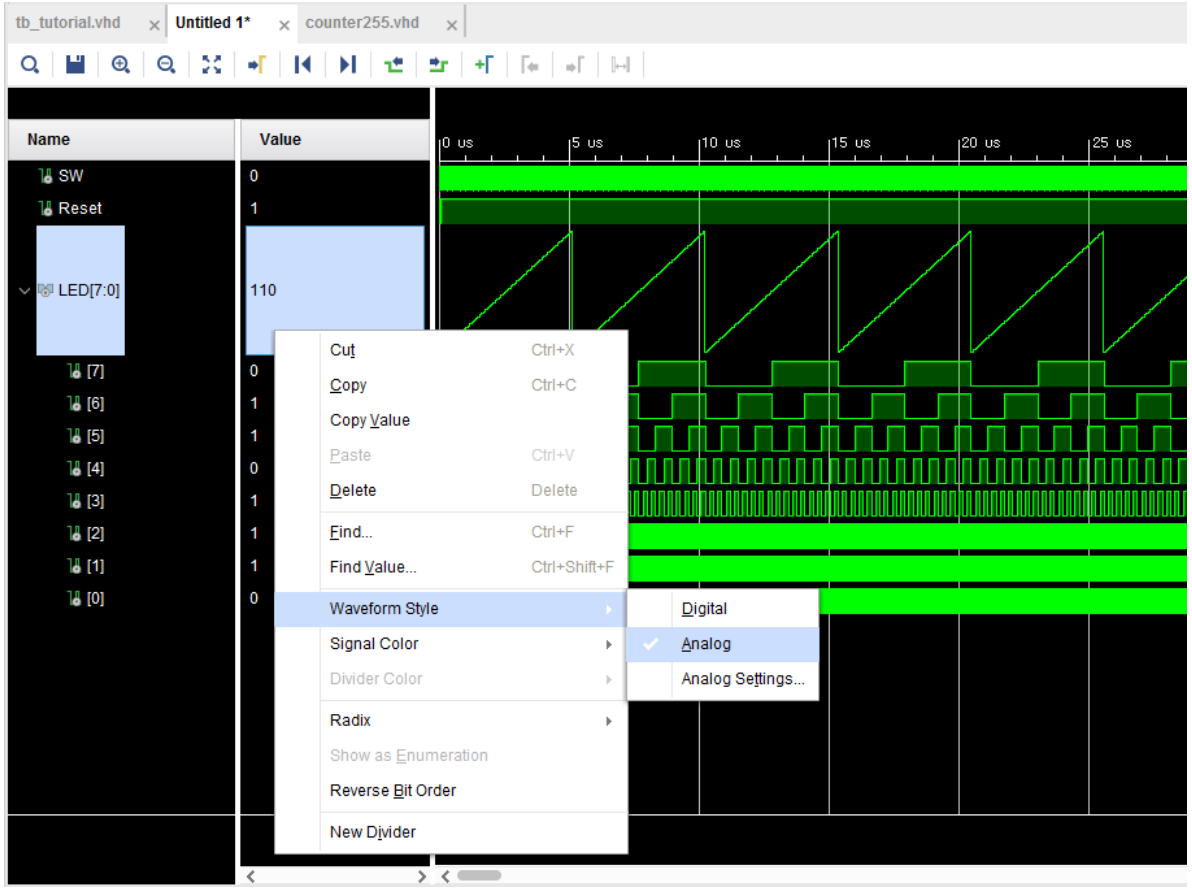
표기(16진수, 10진수), 색상 등 변경

Run All 클릭 시 추가 시뮬레이션

0 – 255 카운터

HUINS
Human Intelligent System

Simulation 출력 파형 확인



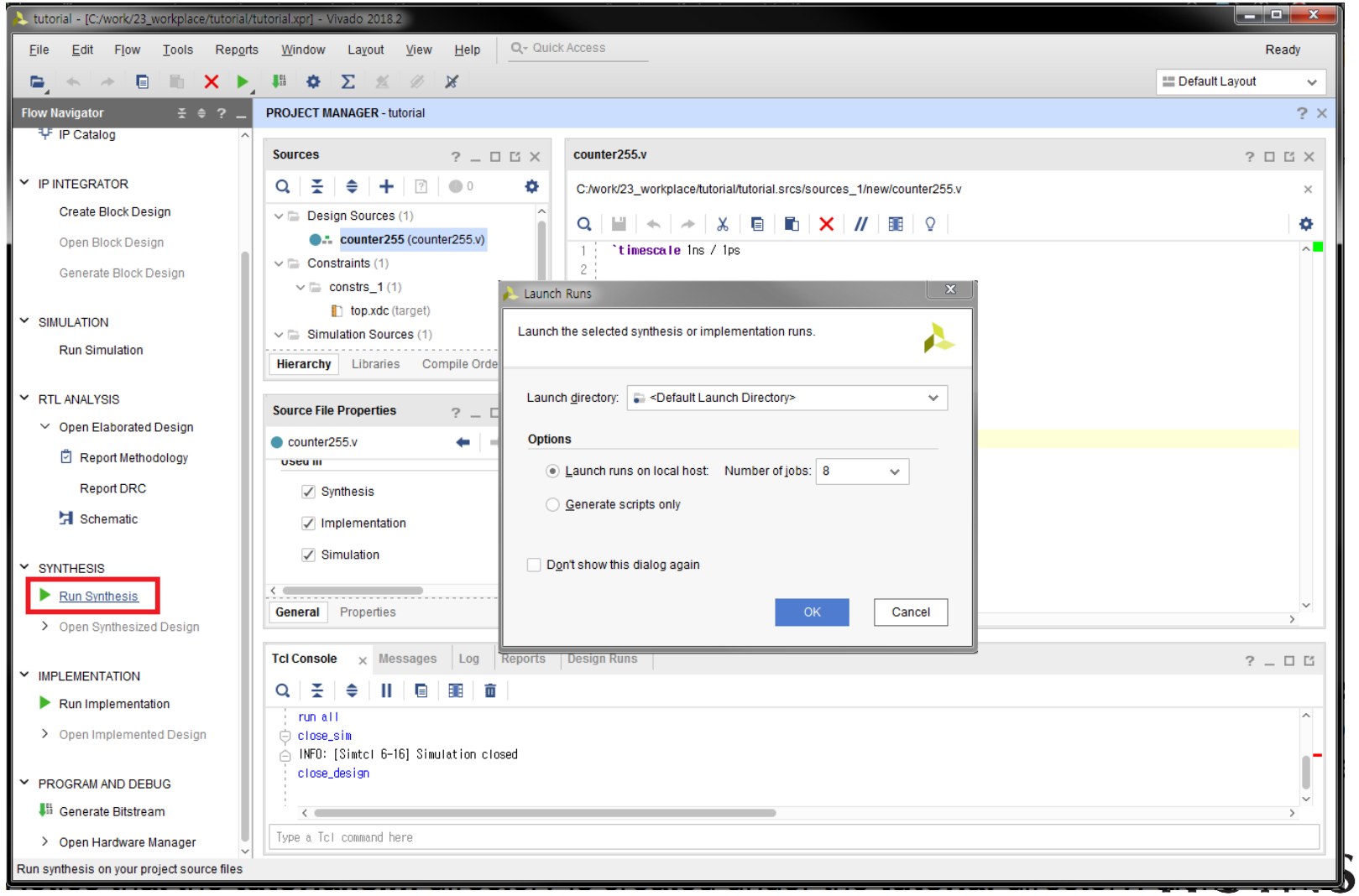
신호를 아날로그 형태로도 출력할 수 있음

HUINS
Human Intelligent System

Synthesis – 합성

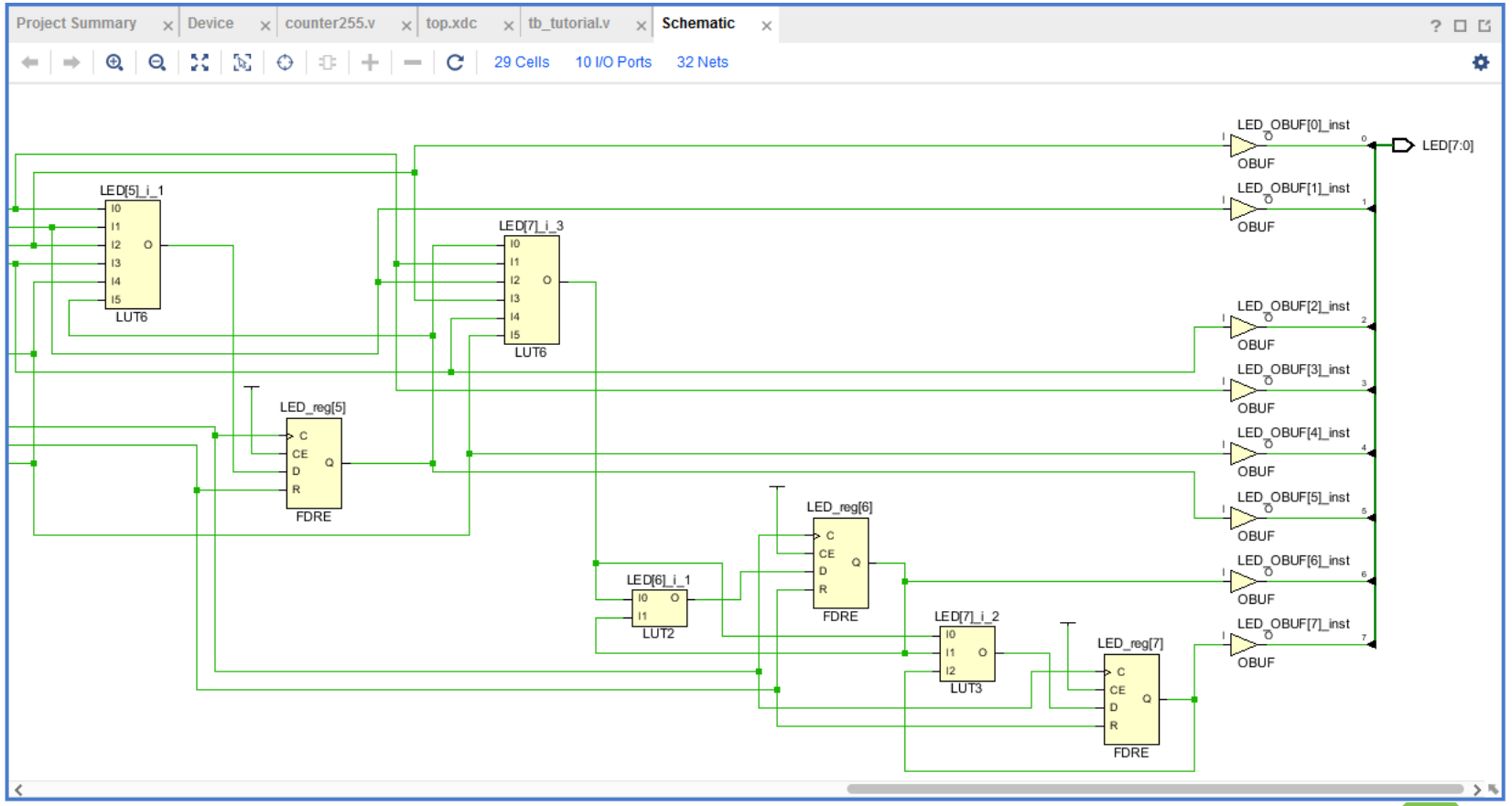
SYNTHESIS –
Run Synthesis 실행

Schematic
→ 넷 리스트로 변환



Synthesis – 합성

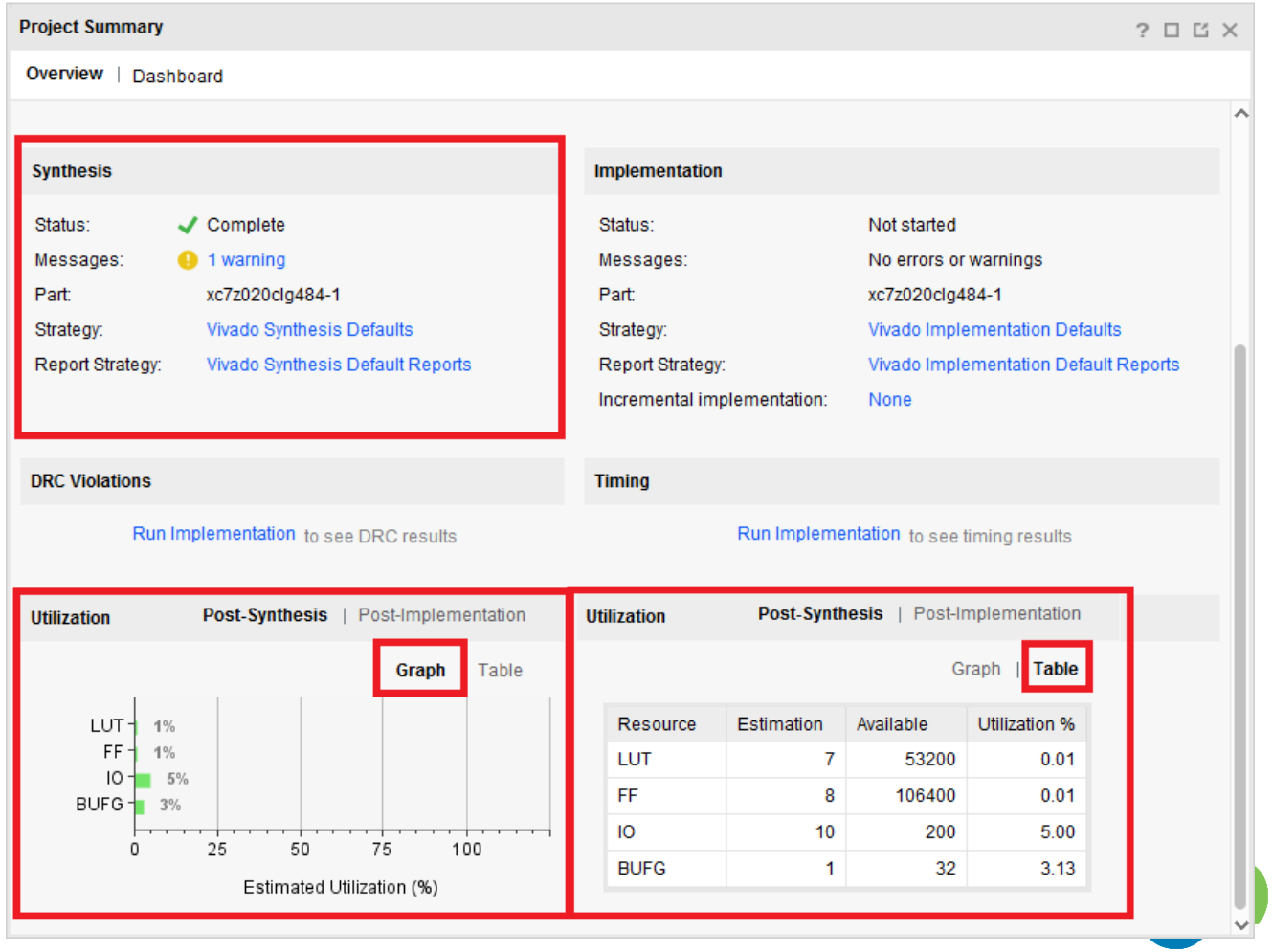
Schematic 창
- LUT, FF으로 구현
- 입력, 출력 버퍼
- I/O 포트(LED, SW)



HUINS
Human Intelligent System

Synthesis – 합성

Project Summary에서
사용된 리소스 정보 확인
- LUT
- FF
- IO
- BUFG

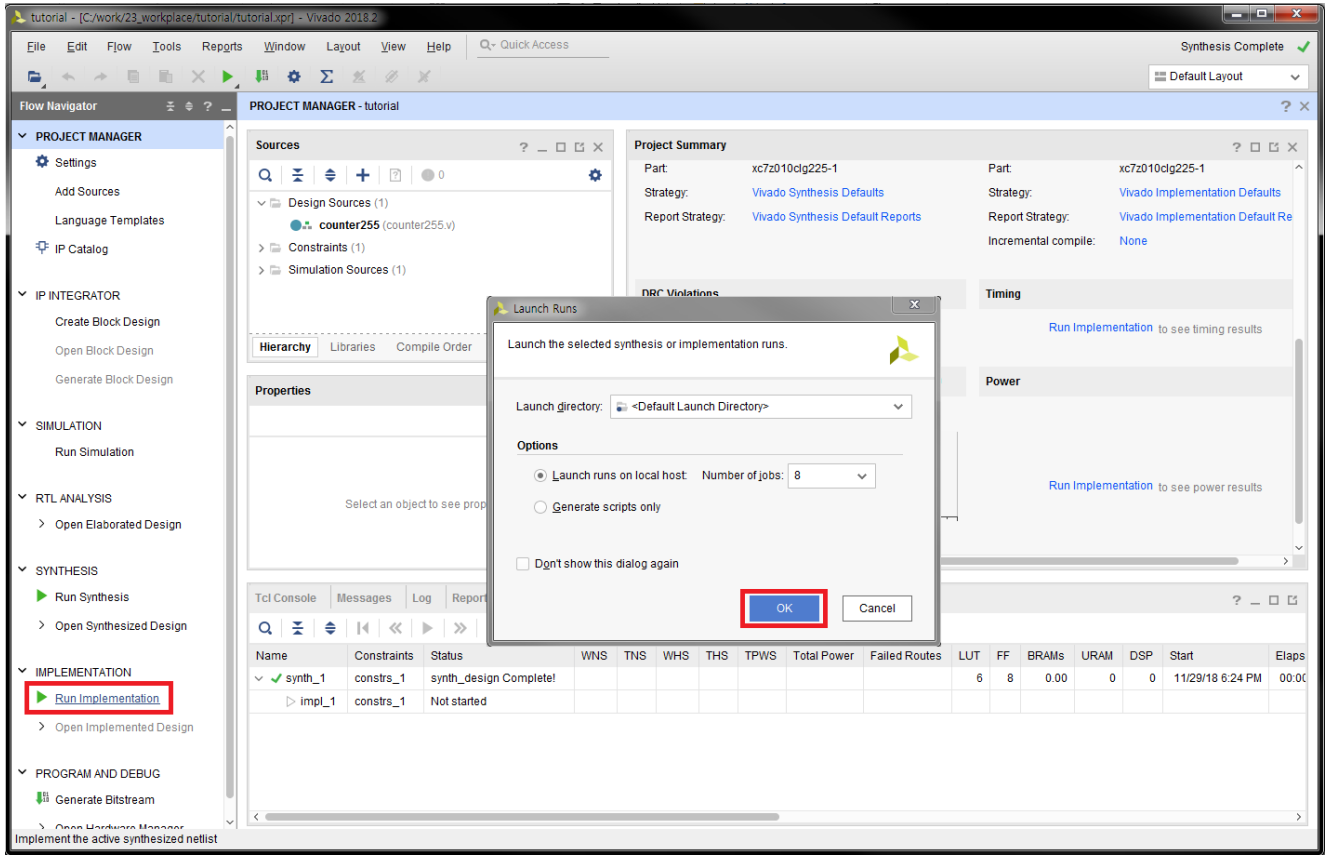


HUINS
Human Intelligent System

Implementation

Xilinx FPGA Implementation

- Translate
  - Netlist Translate(NGC → NGD)
- Map
  - Netlist 해당 장치 리소스 매핑
  - LUT, FF, BRAM
- Place & Route
  - 실제 FPGA 리소스 연결, 배치



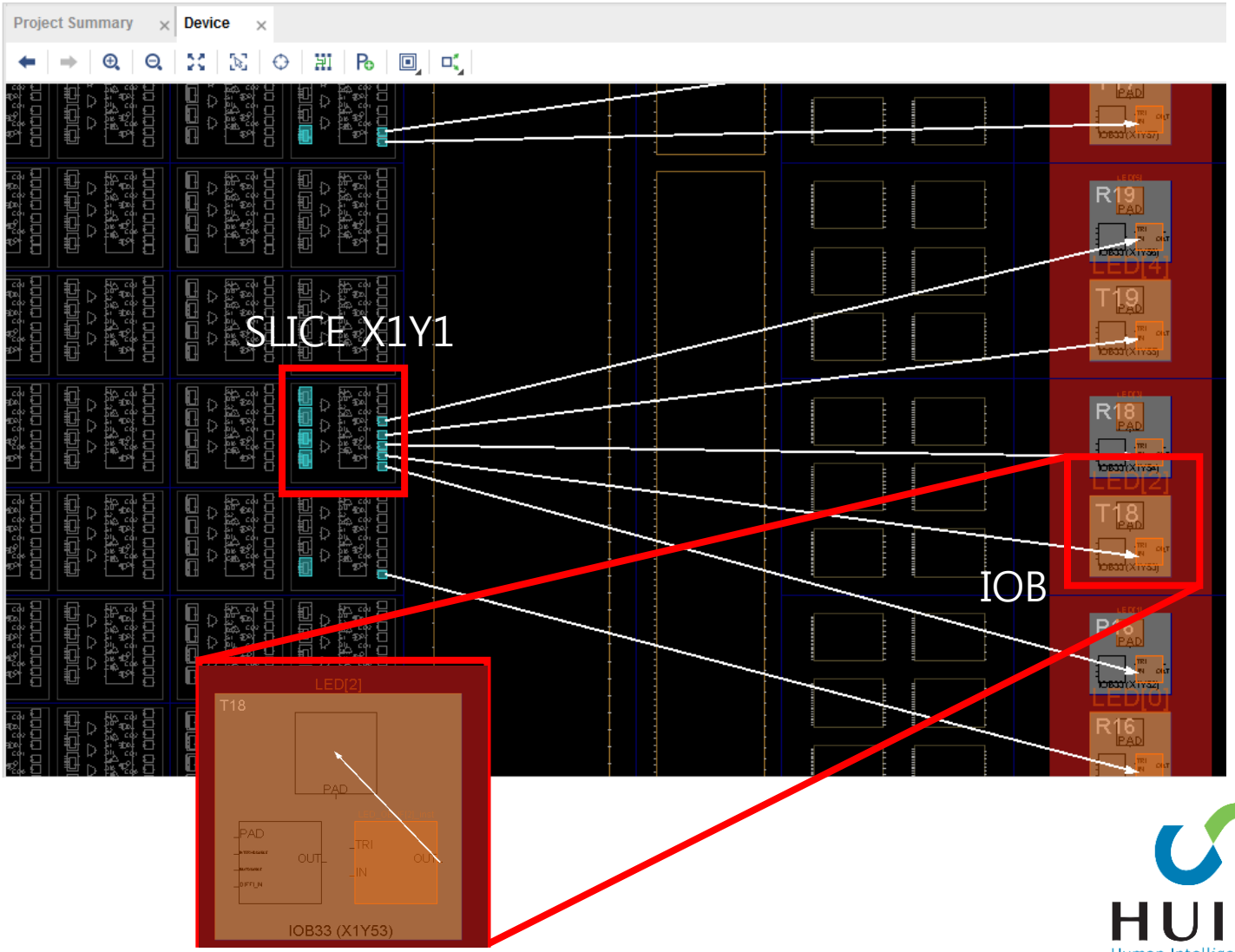
HUINS
Human Intelligent System

Implementation

Netlist의 연결, 배치 확인

LED_OBUF 선택 시
SLICE X1Y1에 배치된 것 확인 가능

SW → SW_IBUF
SLICE(LUT → REG)
REG → LED_OBUF→ PAD

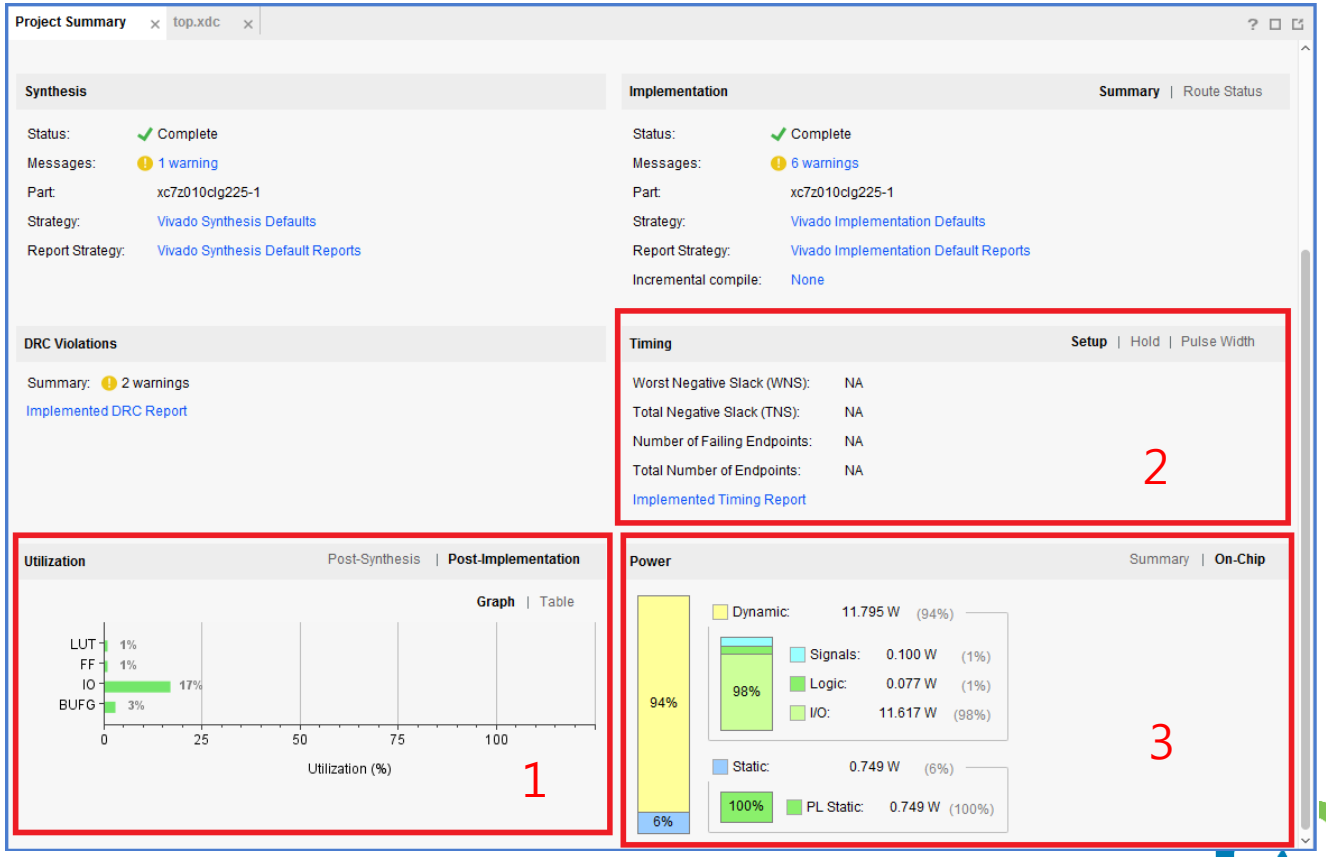


HUINS
Human Intelligent System

Implementation

Project Summary

1. 실제로 배치에 사용된 리소스
2. 타이밍 정보
3. 예상 소모 전력

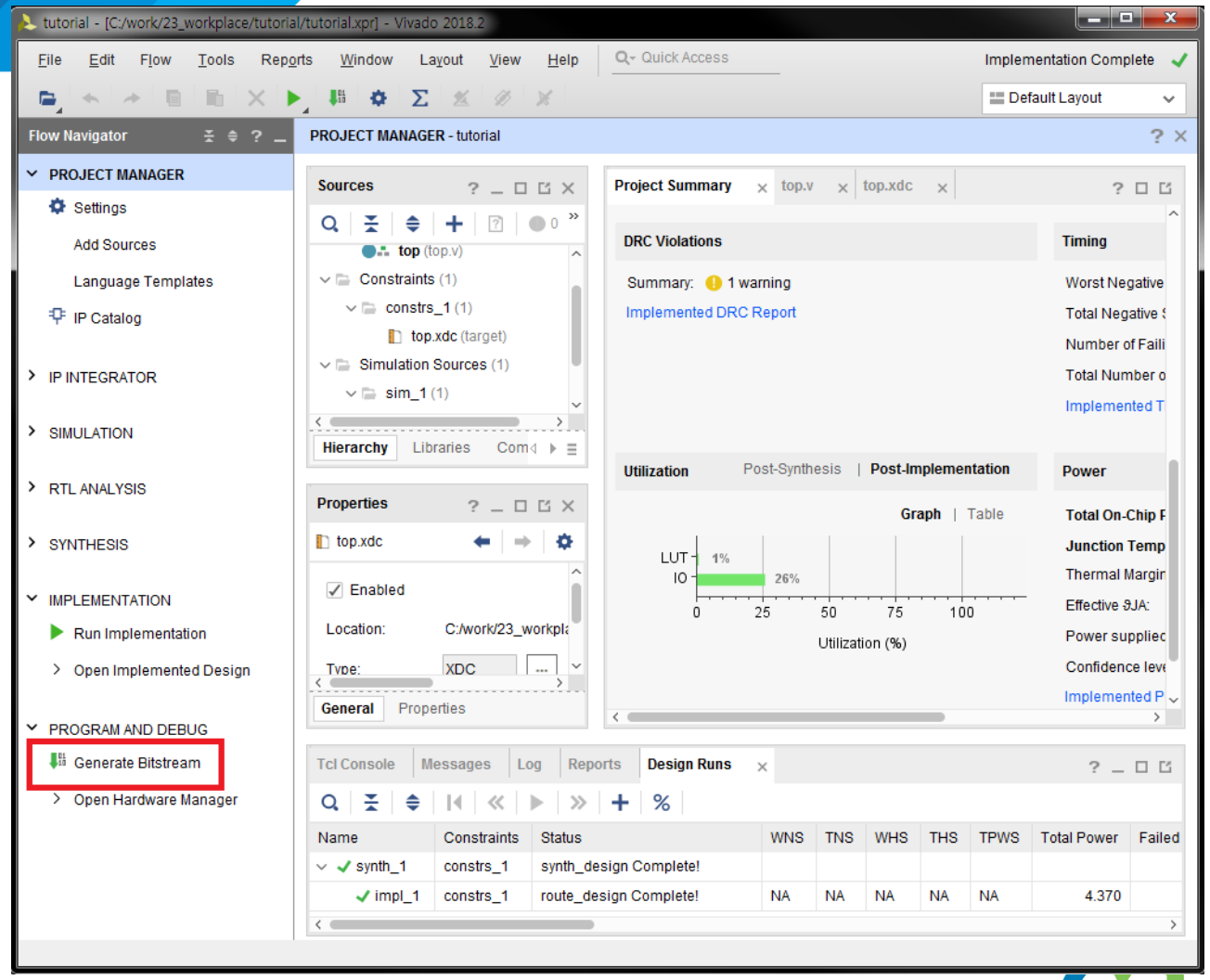


HUINS
Human Intelligent System

Bit 파일 생성

PROGRAM AND DEBUG –
Generate Bitstream

FPGA에 다운로드 할
Bitstream 파일을 생성한다.



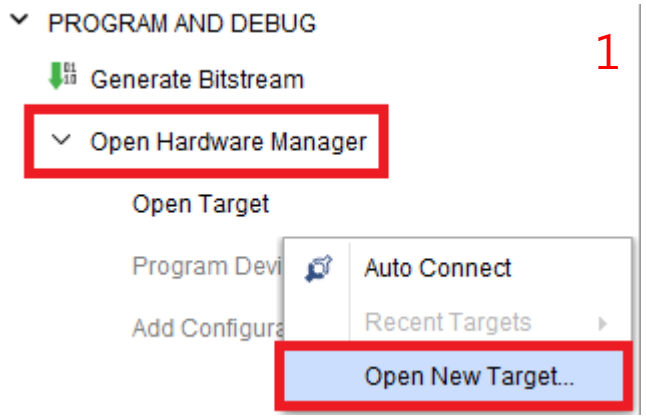
HUINS
Human Intelligent System

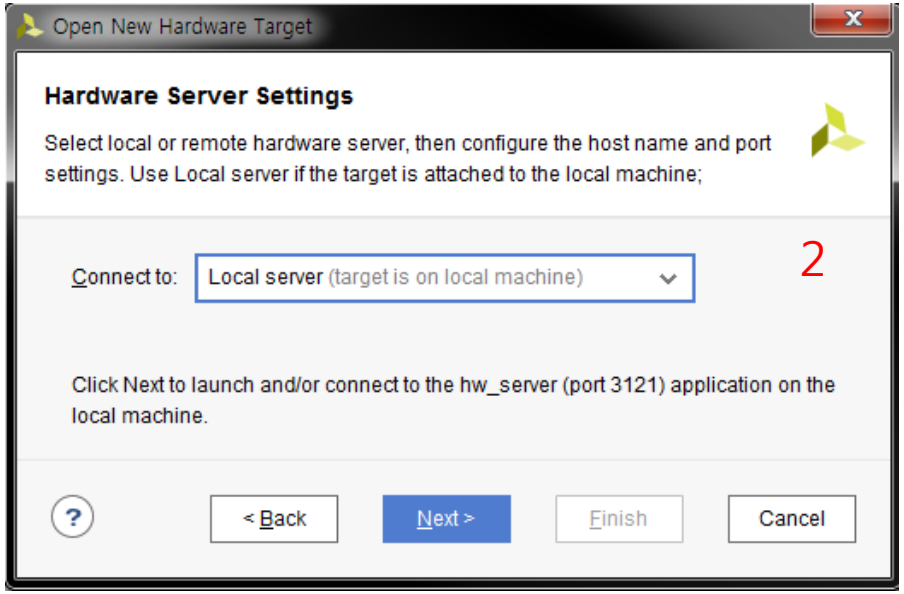
PC에 보드 연결

JTAG 케이블로 PC와 연결

Hardware Manager
- Open New Target 실행
- Local Server로 연결

Auto Connect 실행 시
자동으로 연결

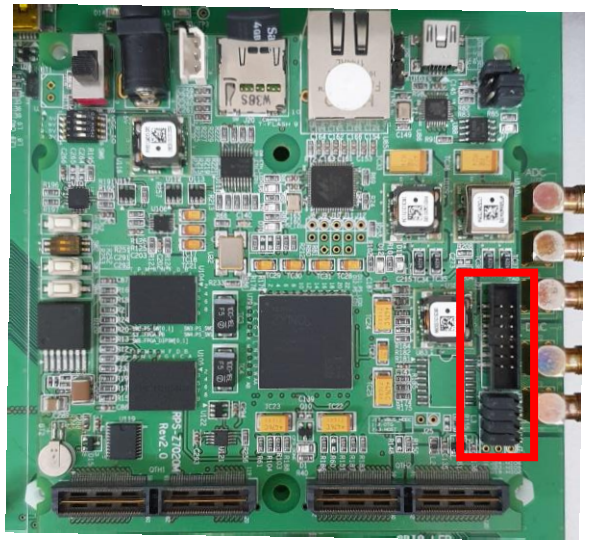




HUINS
Human Intelligent System

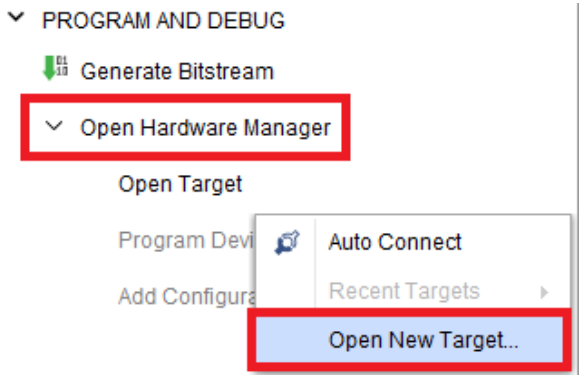
PC에 보드 연결

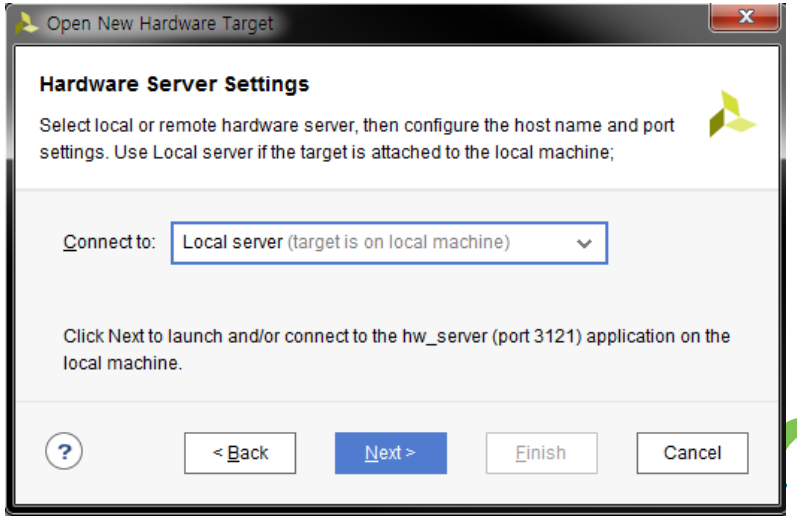
JTAG 케이블로 PC와 연결



Hardware Manager
- Open New Target 실행
- Local Server로 연결

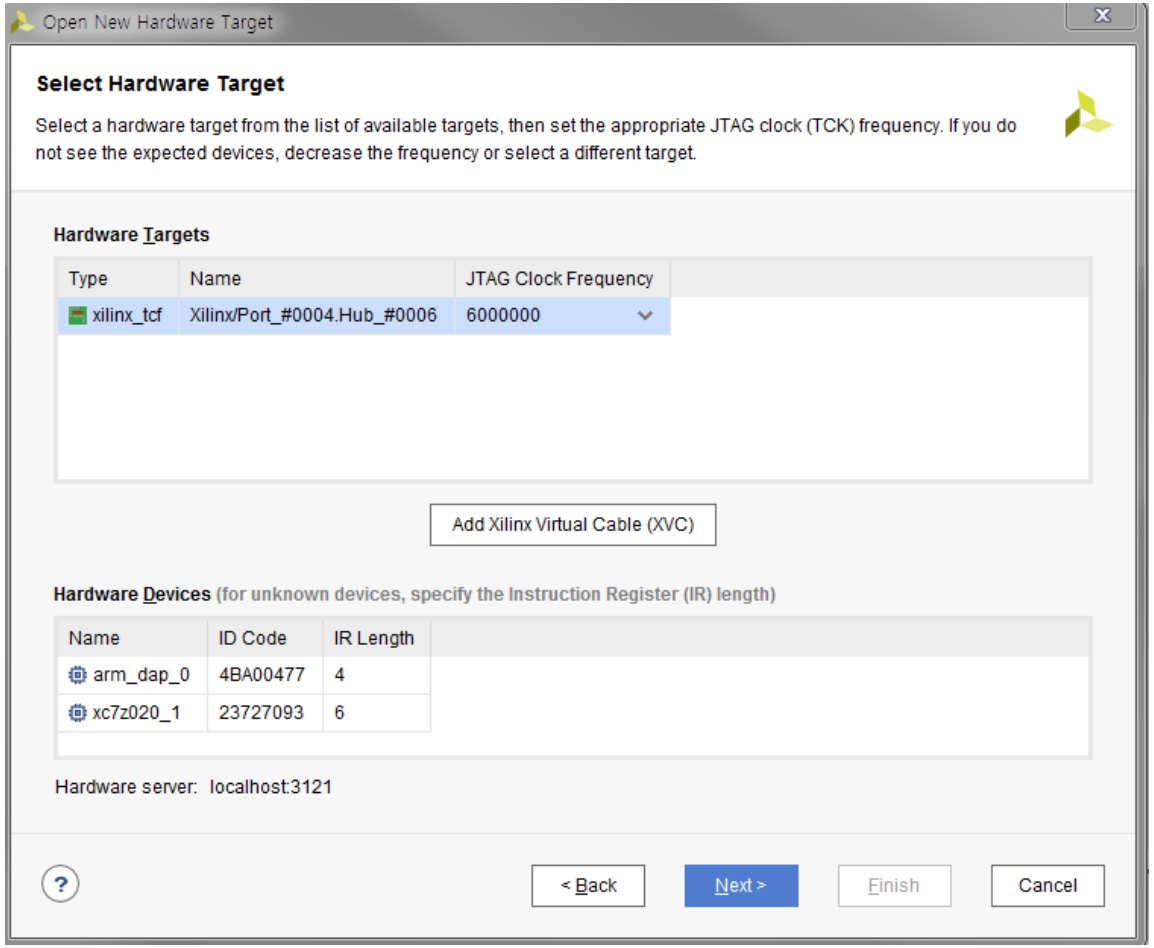
Auto Connect 실행 시
자동으로 연결





HUINS
Human Intelligent System

PC에 보드 연결

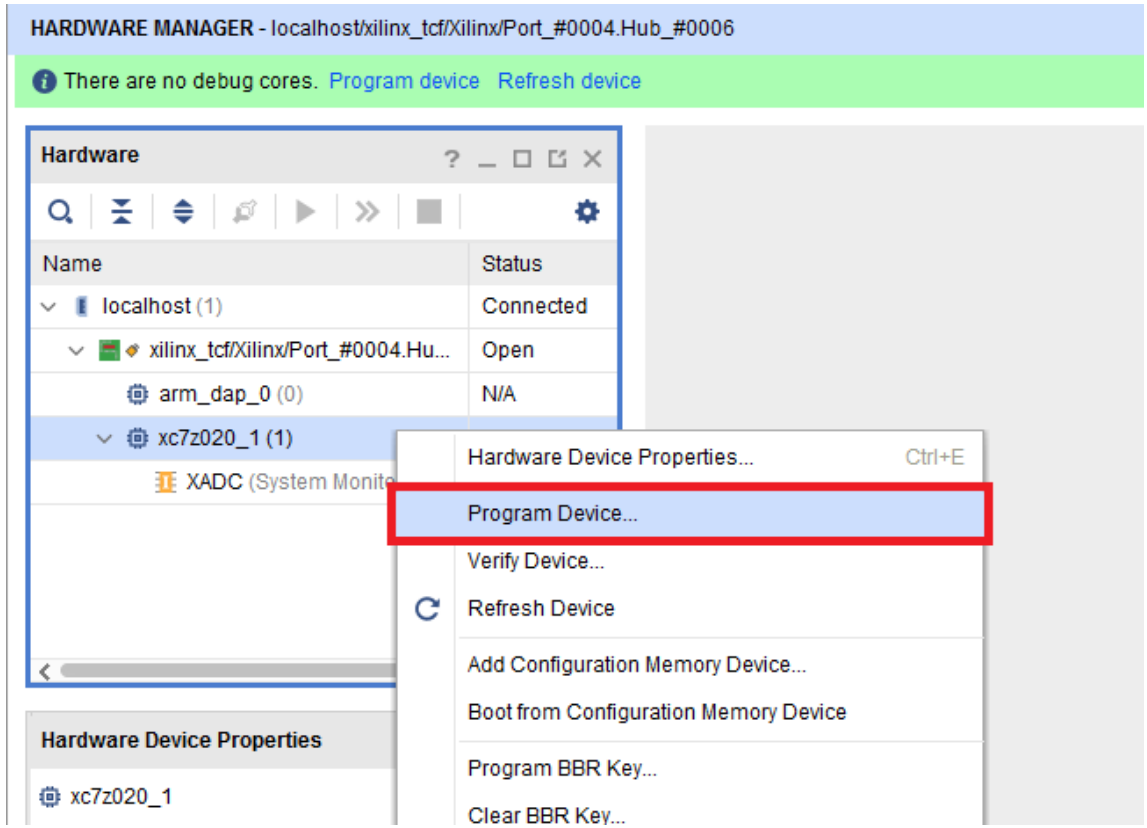


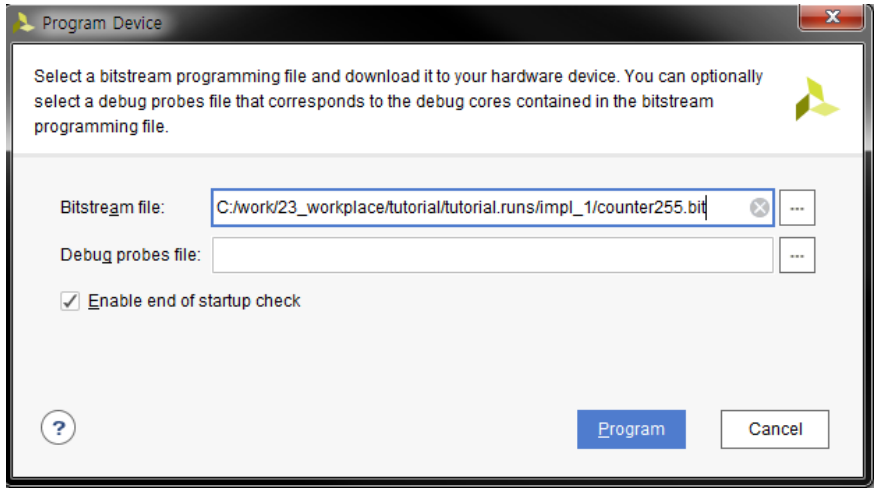
여러 디바이스 연결 시
Hardware Devices
정보 확인 후 선택
Part: xc7z020_1

JTAG Clock 설정 후 연결

HUINS
Human Intelligent System

Program Device





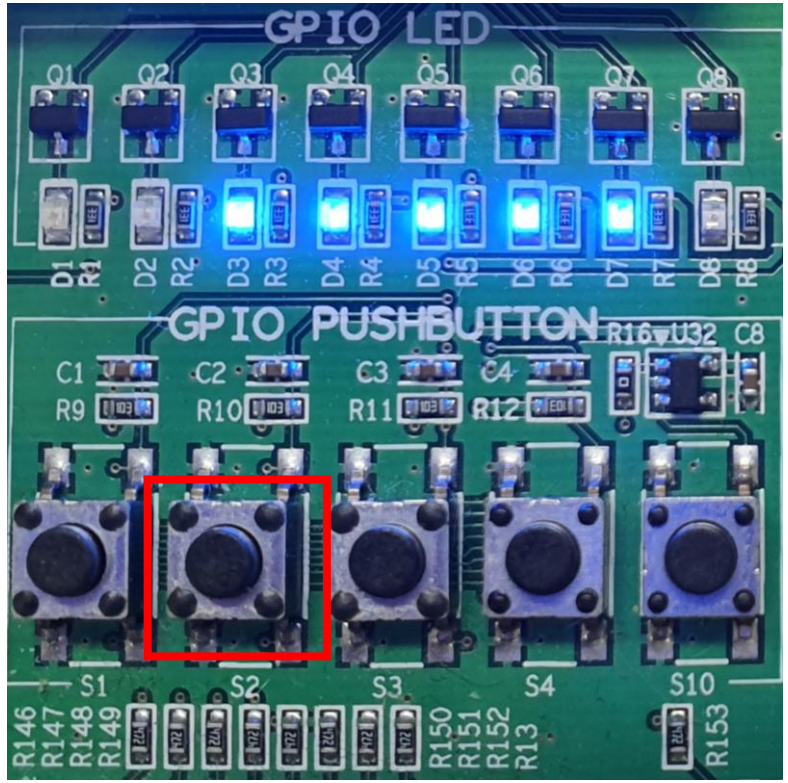
Bit 파일 선택 후 FPGA에 다운로드

HUINS
Human Intelligent System

작동 확인

스위치(SW2) 입력 시
LED 출력이 1씩 증가

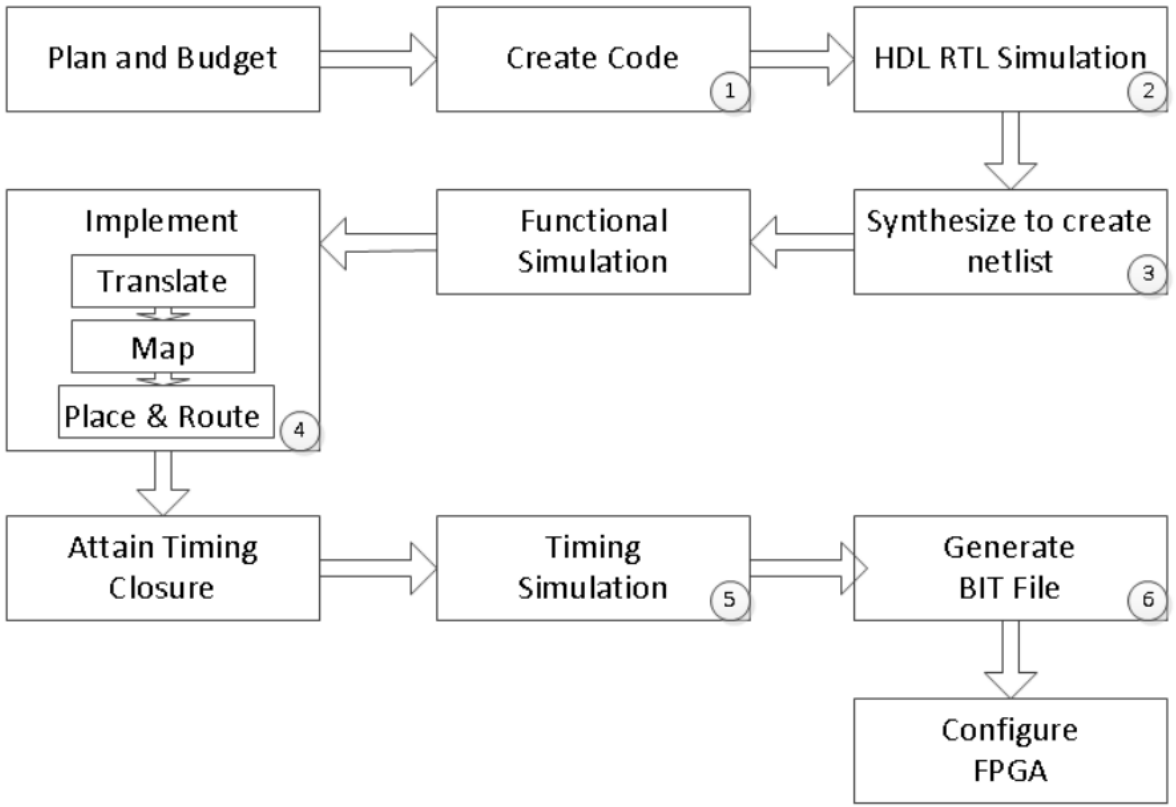
스위치(SW1)을 누른 상태에서(Reset=0)
S2를 입력 시(Posedge SW) 0으로 초기화



HUINS
Human Intelligent System

Vivado Tutorial Summary

- Vivado 통한 HDL Design Flow
- 프로젝트 생성 HDL 소스 작성
- 시뮬레이션을 통한 작동 테스트
- Synthesize → Implement → Bit 파일 생성



HUINS
Human Intelligent System

2. Processing System Tutorial

Vivado

A. 프로젝트 생성

B. Processing System 설정

- I/O Peripheral(UART)
- DDR3 Memory Controller
- HW Export

SDK

C. 애플리케이션 프로젝트 생성

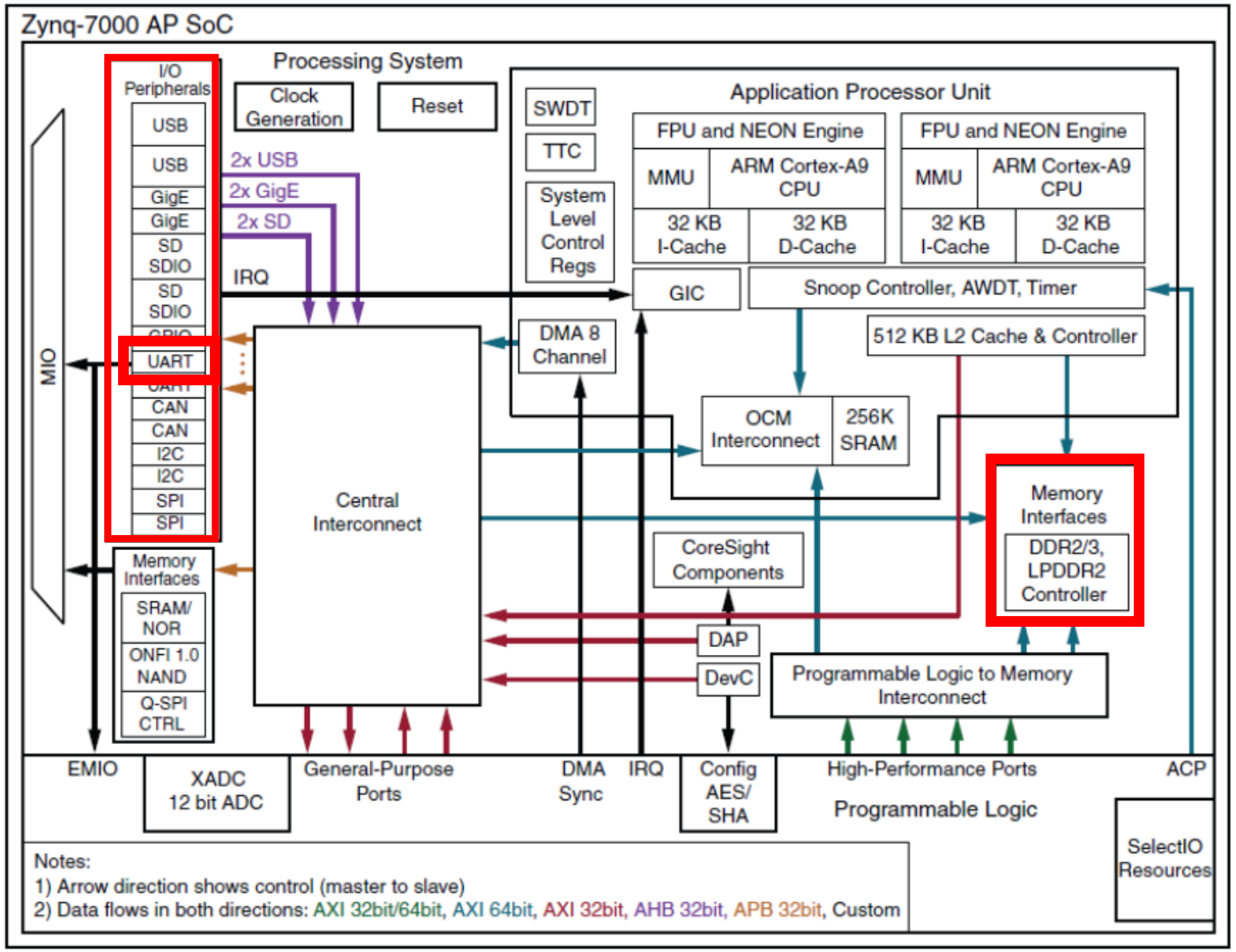
D. “Hello World” 출력

Zynq Processing System

APU
- ARM Cortex-A9
- L2 Cache
- OCM(On Chip Memory)

I/O Peripherals
- MIO(PS), EMIO(PL)

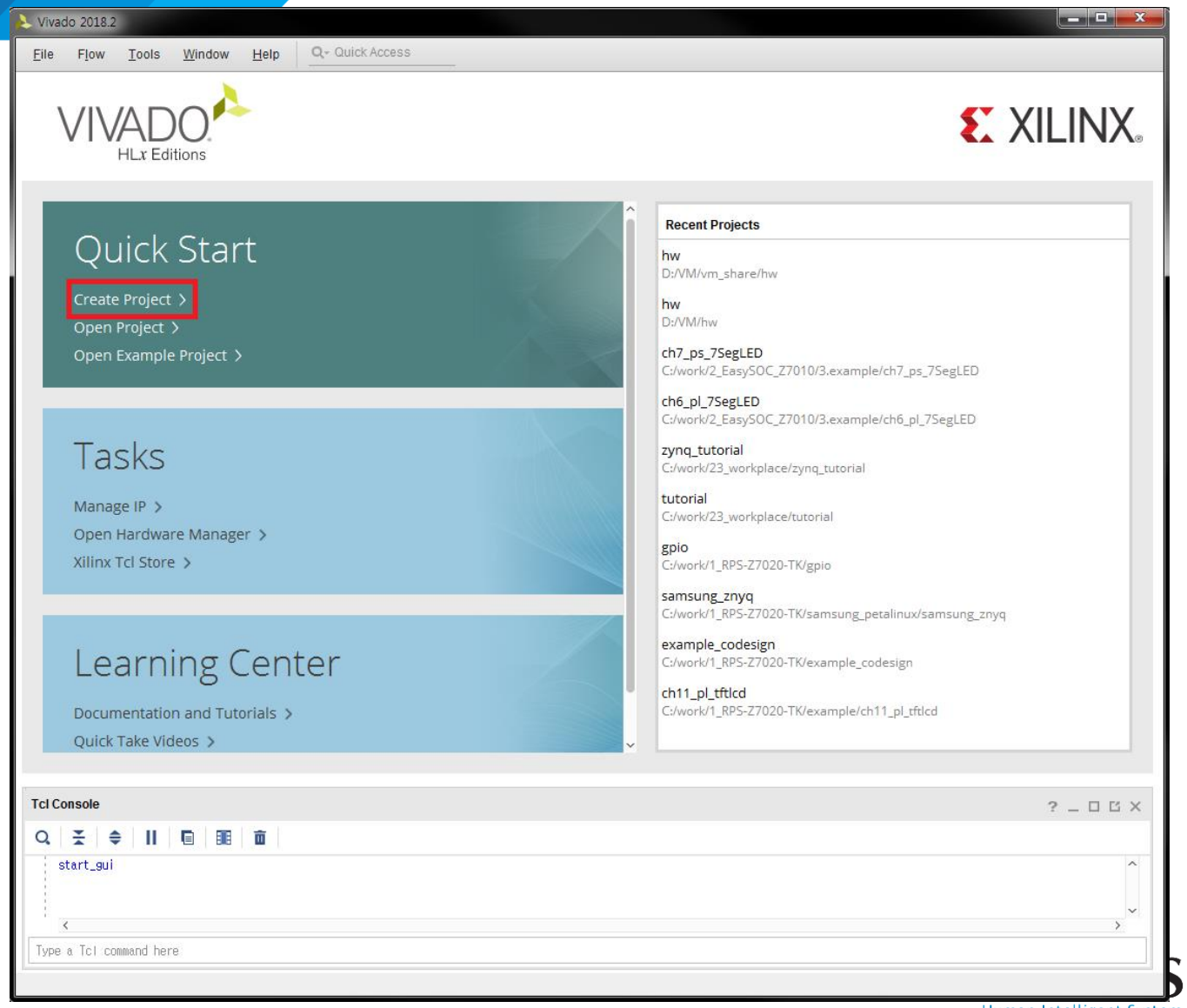
DRAM Controller



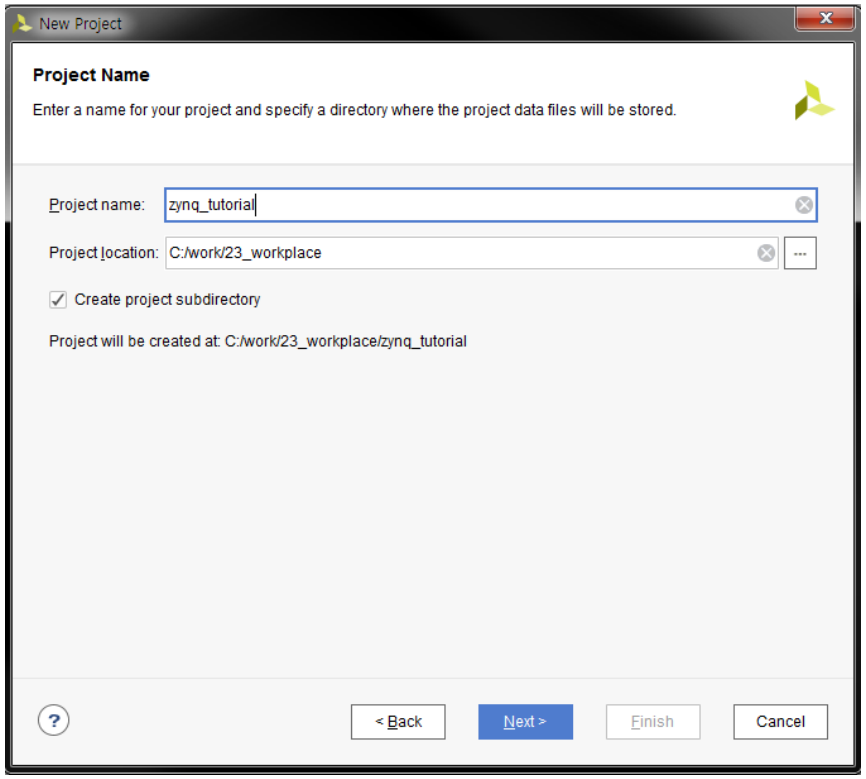
A. 프로젝트 생성

Vivado를 실행하여
프로젝트를 생성한다.

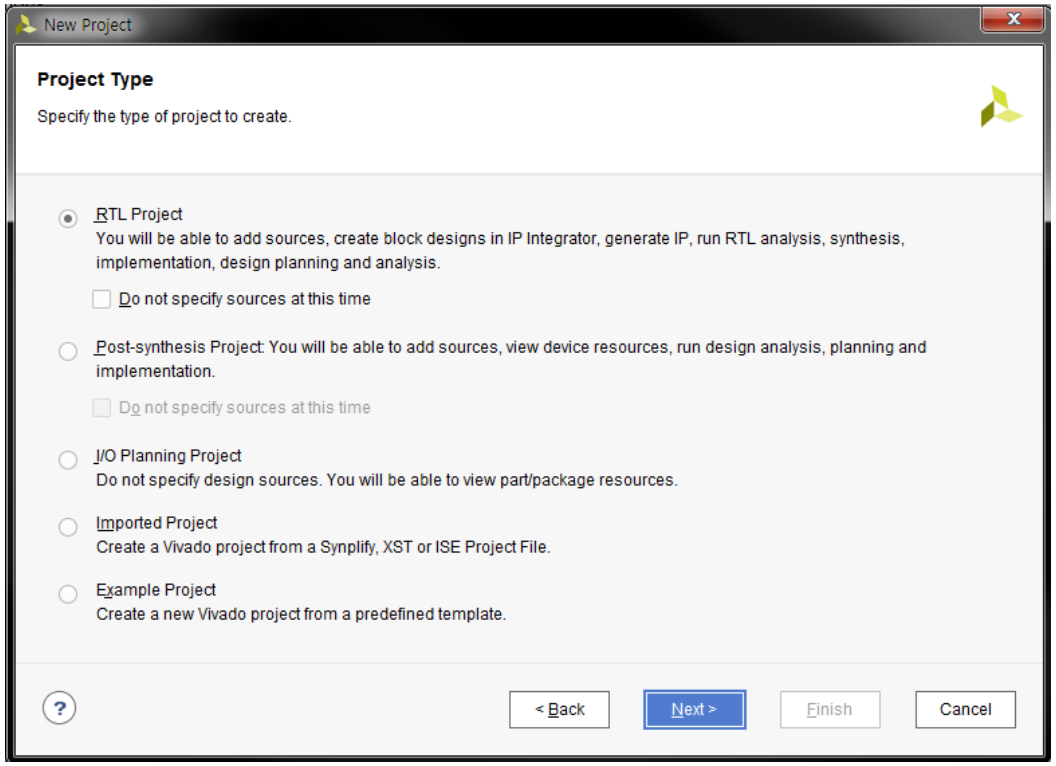
File → Project → New
또는
Quick Start → Create Project



프로젝트 생성



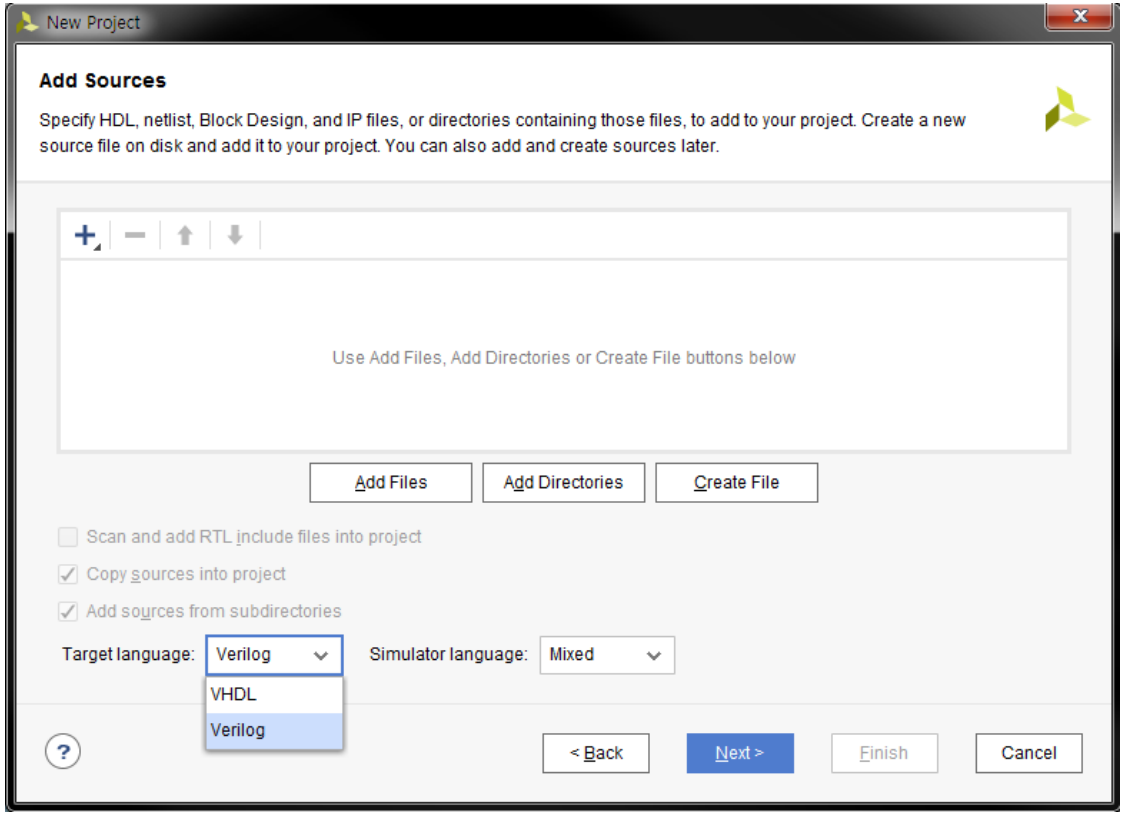
프로젝트 이름, 경로 지정

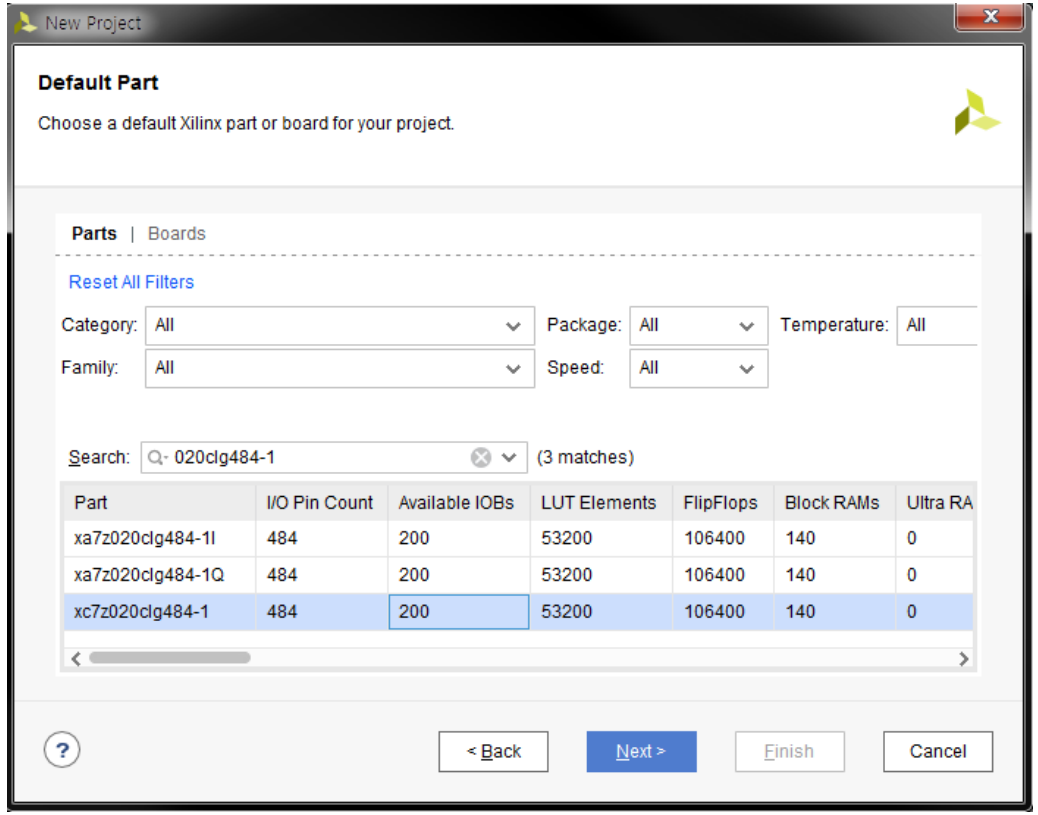


프로젝트 타입 지정 → RTL

HUINS
Human Intelligent System

프로젝트 생성



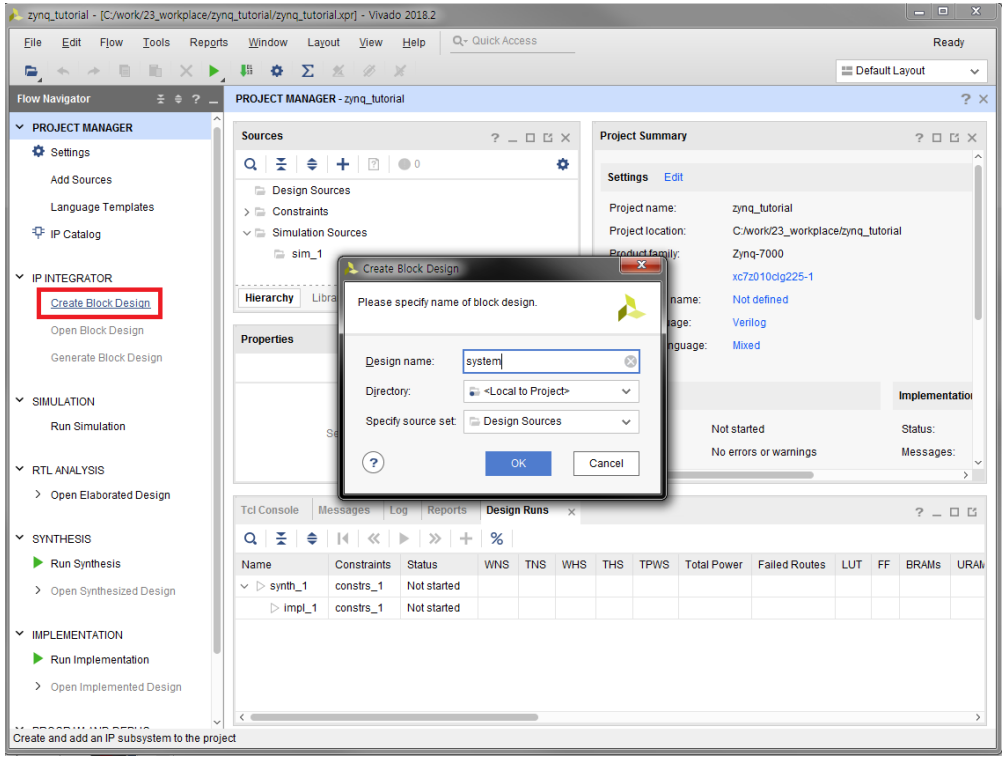


소스 추가(파일, 폴더, 생성)
default Language 선택

사용 FPGA/SoC 파트 선택
xc7z020clg484-1

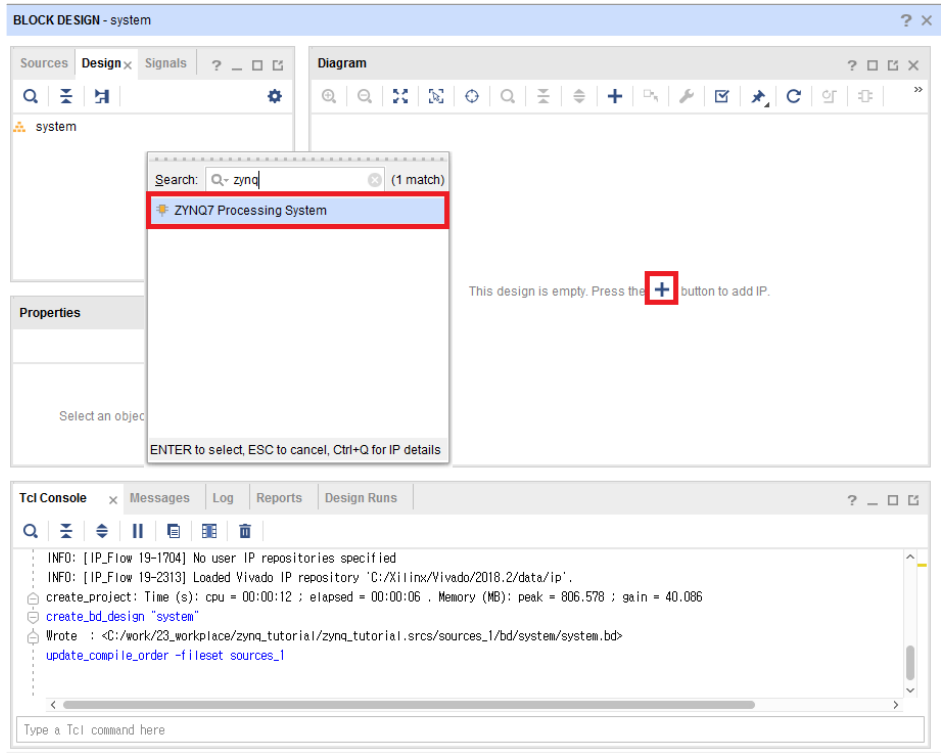
HUINS
Human Intelligent System

Processing System IP 추가



블록 디자인 생성

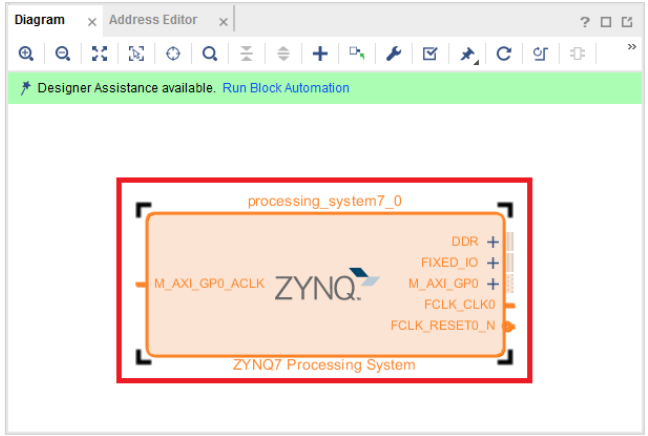
GUI 기반 설계



Zynq PS IP 추가

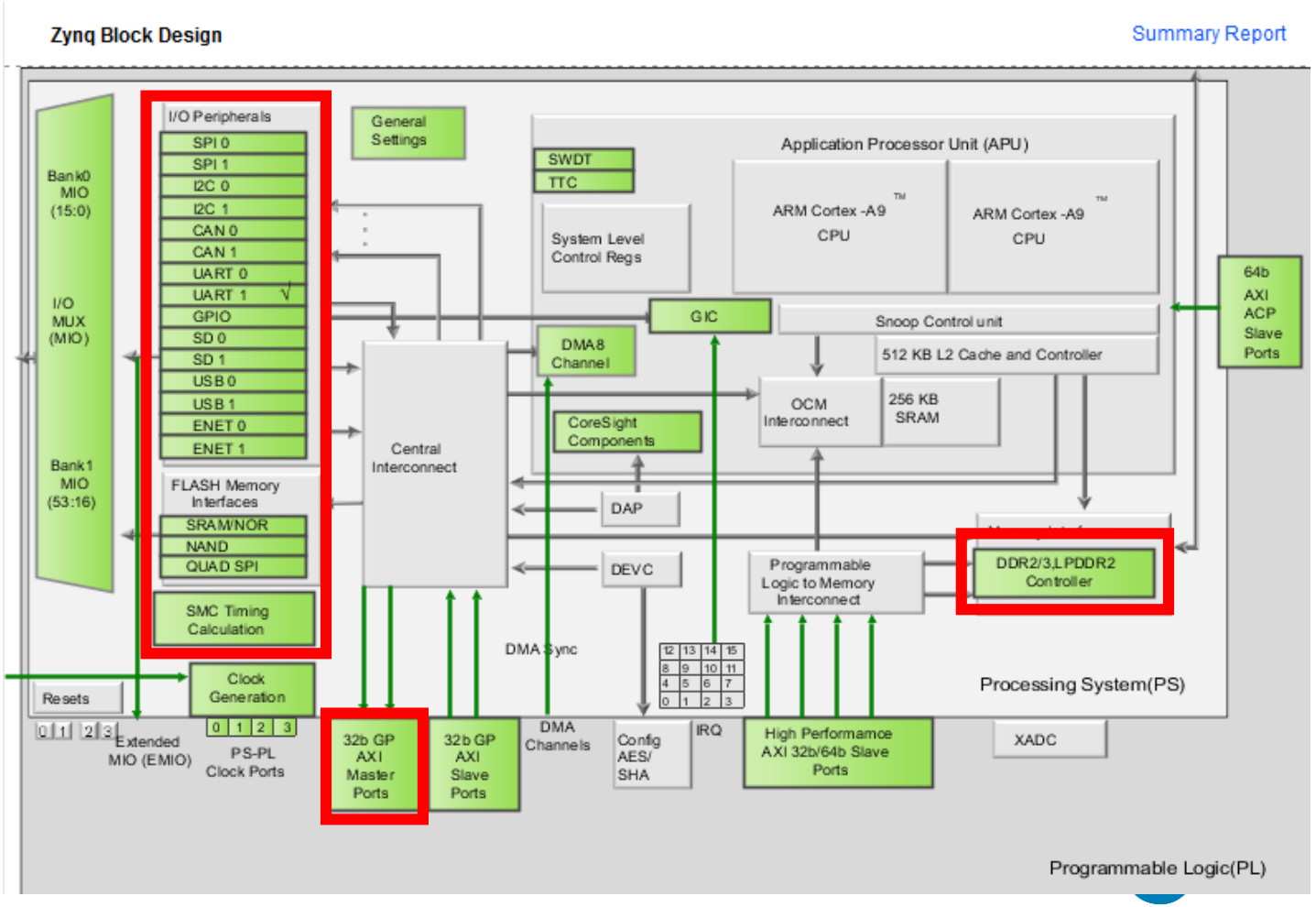
HUINS
Human Intelligent System

B. Processing System 설정



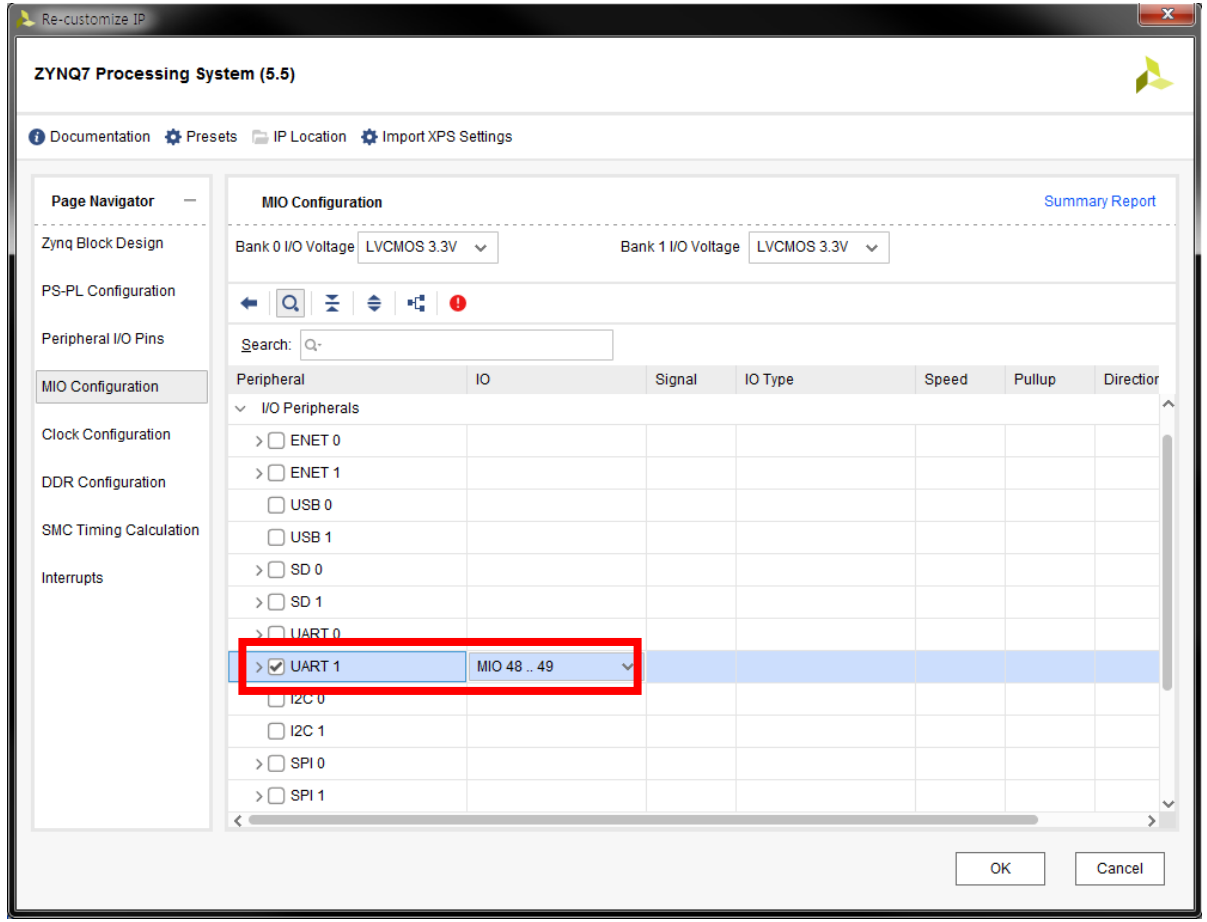
Zynq PS 설정

- I/O Peripheral → UART
- Memory Controller
- 32b GP AXI Master



HUINS
Human Intelligent System

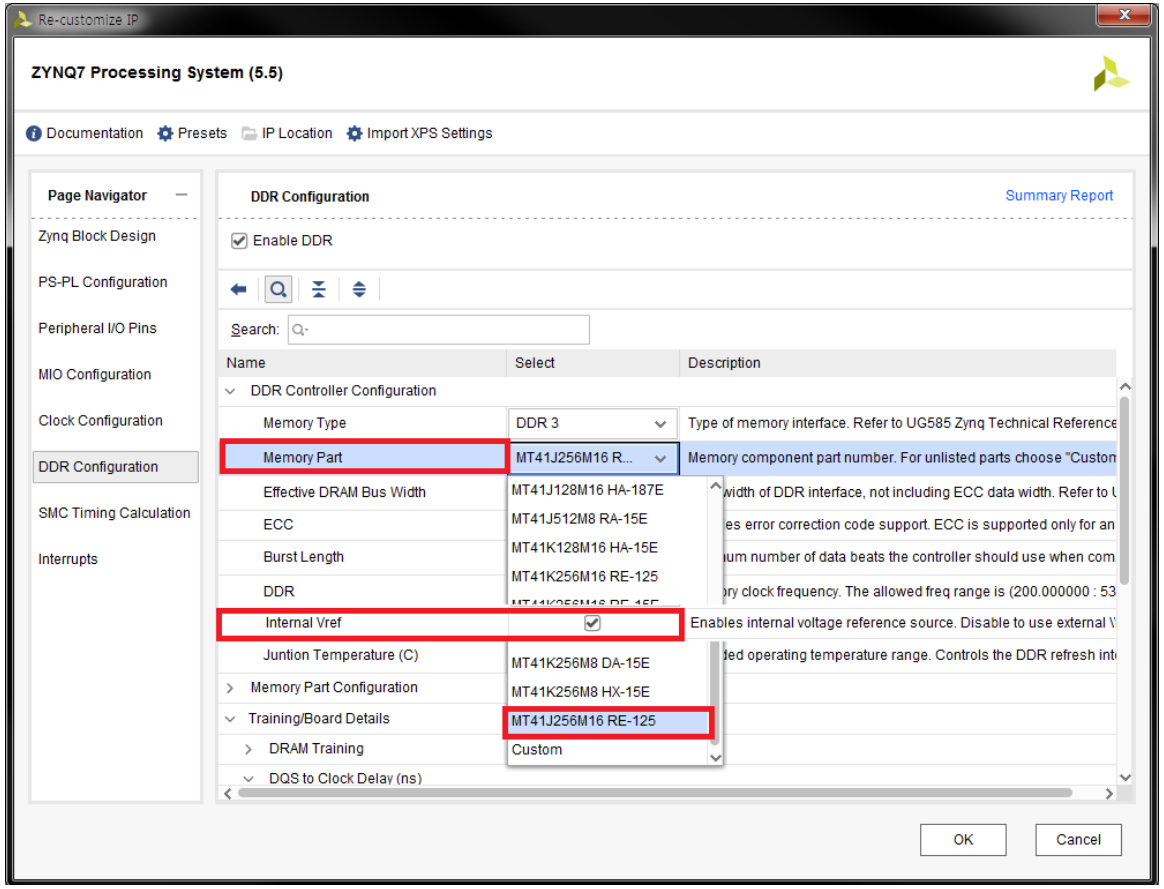
B. Processing System 설정

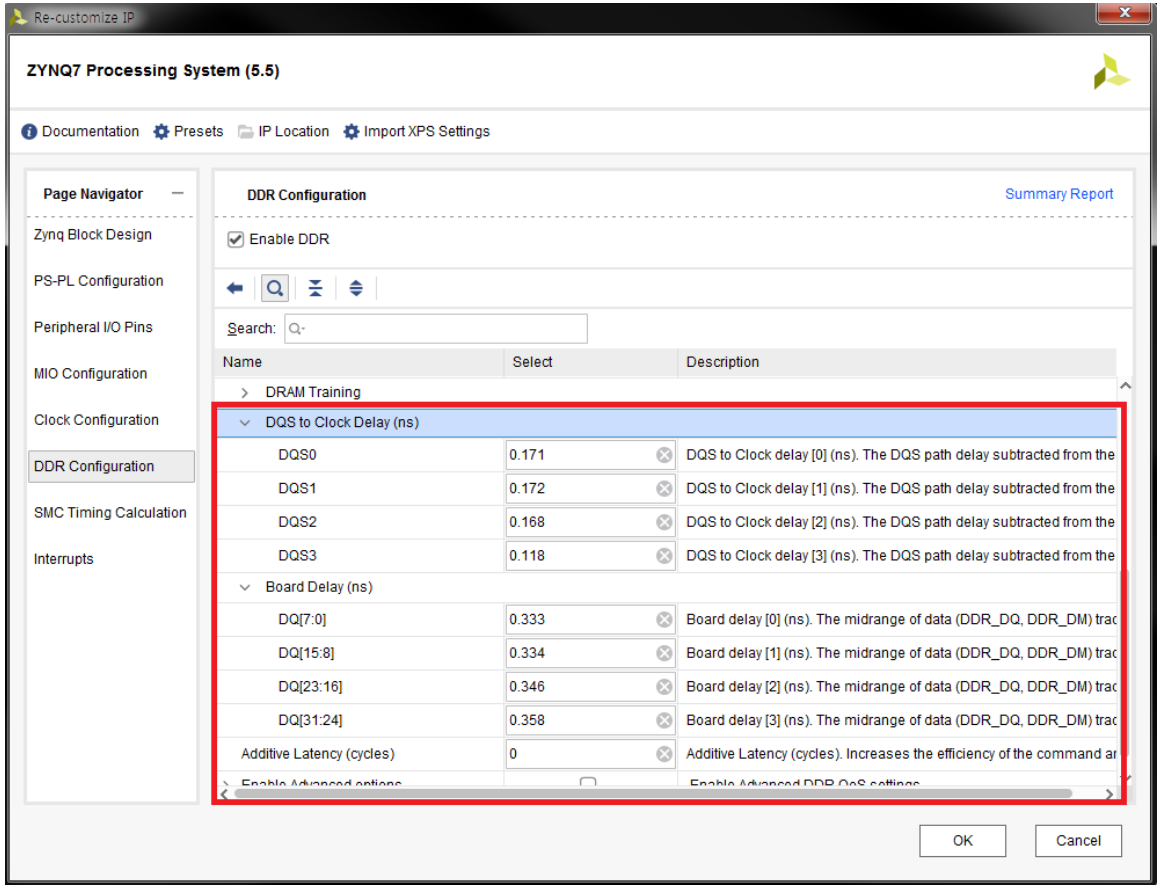


UART1을 Enable 하며, MIO 48-49번 핀 할당

HUINS
Human Intelligent System

B. Processing System 설정





DRAM Controller의 값을 변경한다
(다음 페이지 표 참조)

HUINS
Human Intelligent System

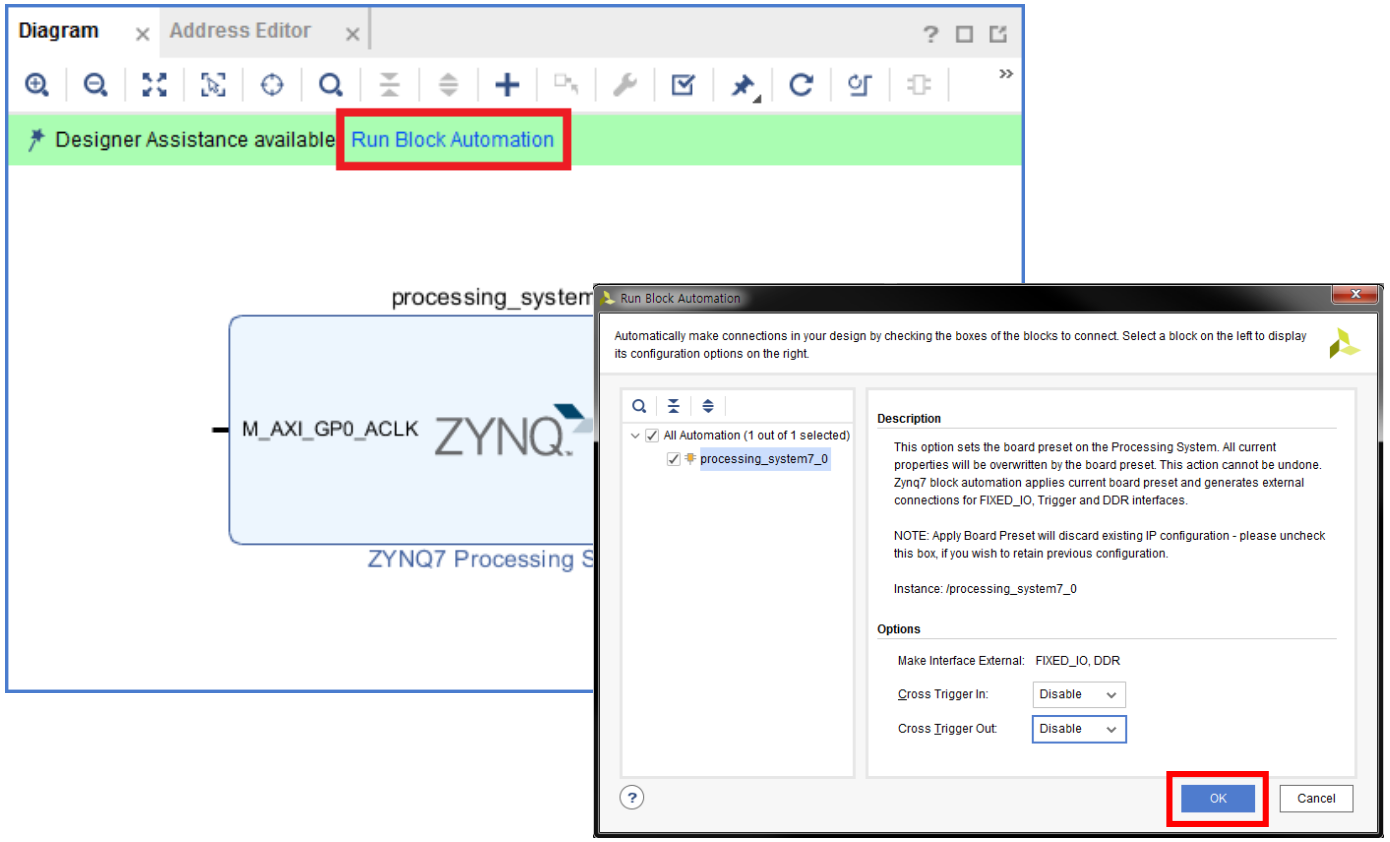
B. Processing System 설정

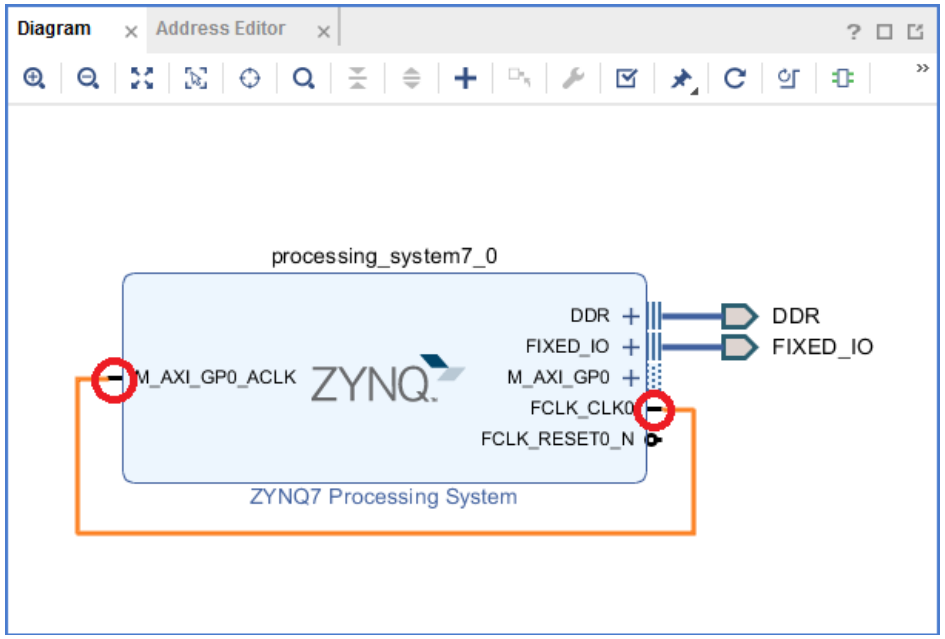
[표] Memory 설정 내용

항목		내용	항목		내용
Memory Type		DDR3	DQS to Clock Delay (ns)	DQS0	0.171
Memory Part		MT41J256M 16 RE-125		DQS1	0.172
Internal Vref		Check		DQS2	0.168
DRAM Training	Write leveling	Check		DQS3	0.118
	Read gate	Check	Board Delay (ns)	DQ[7:0]	0.333
	Read data eye	Check		DQ[15:8]	0.334
				DQ[23:16]	0.346
				DQ[31:24]	0.358

HUINS
Human Intelligent System

B. Processing System 설정



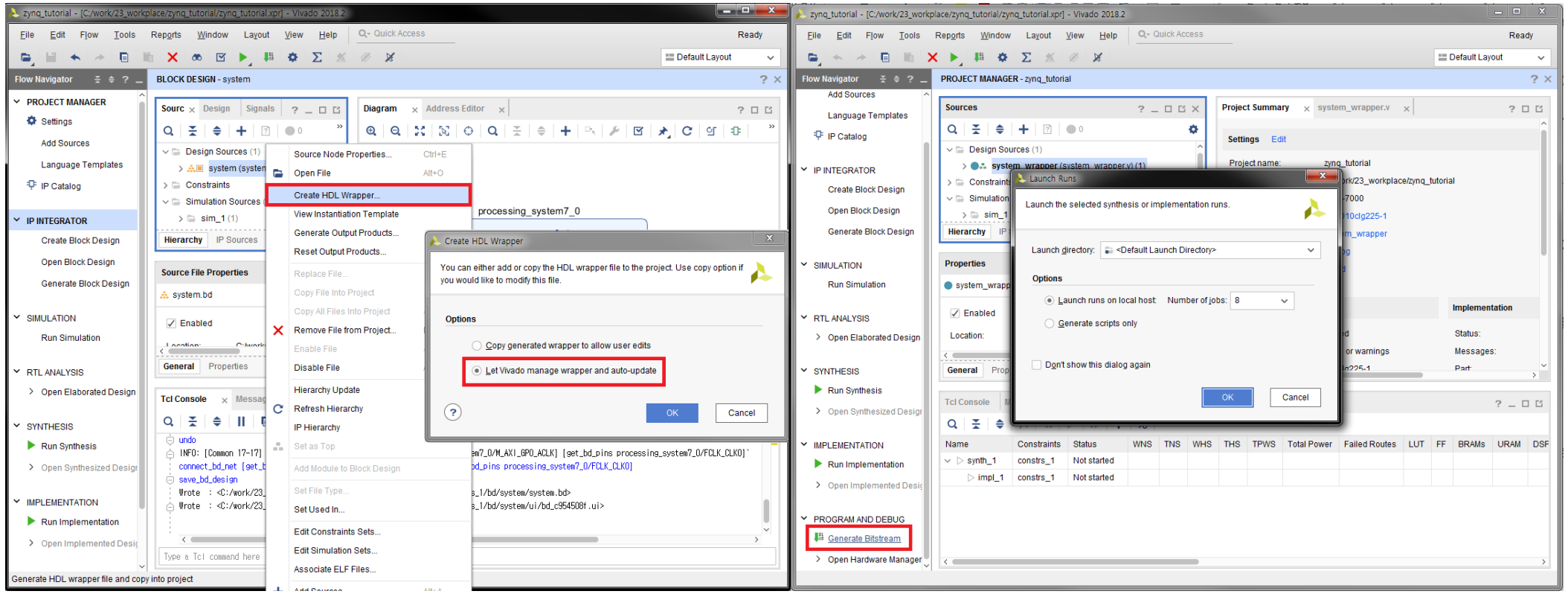


Run Block Automation → FIXED_IO, DDR 연결

ACLK(AXI CLK)과 FCLK(Fabric CLK, PS→PL)을 연결

HUINS
Human Intelligent System

HDL Wrapper

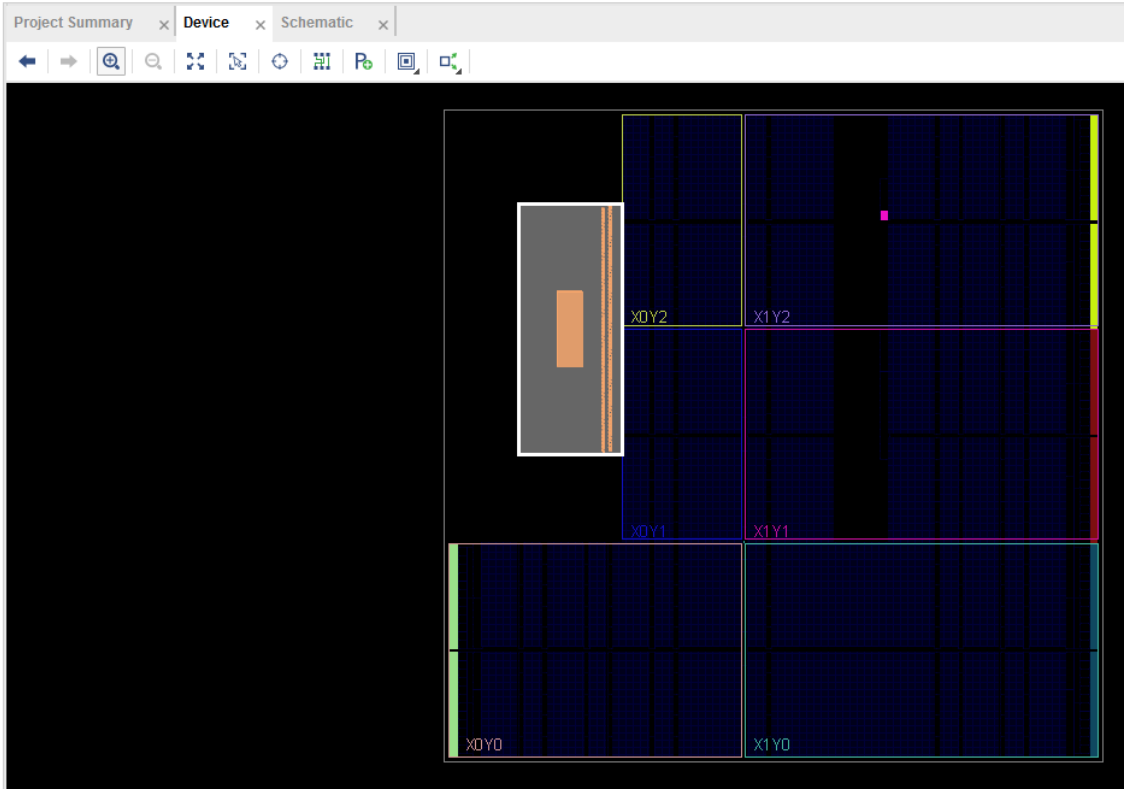


HDL wrapper 생성 → 블록 디자인의 HDL 인스턴스 생성

Generate Bitstream → Synthesis, Implement 거쳐 실행

HUINS
Human Intelligent System

PS – IMPLEMENTATION – Device

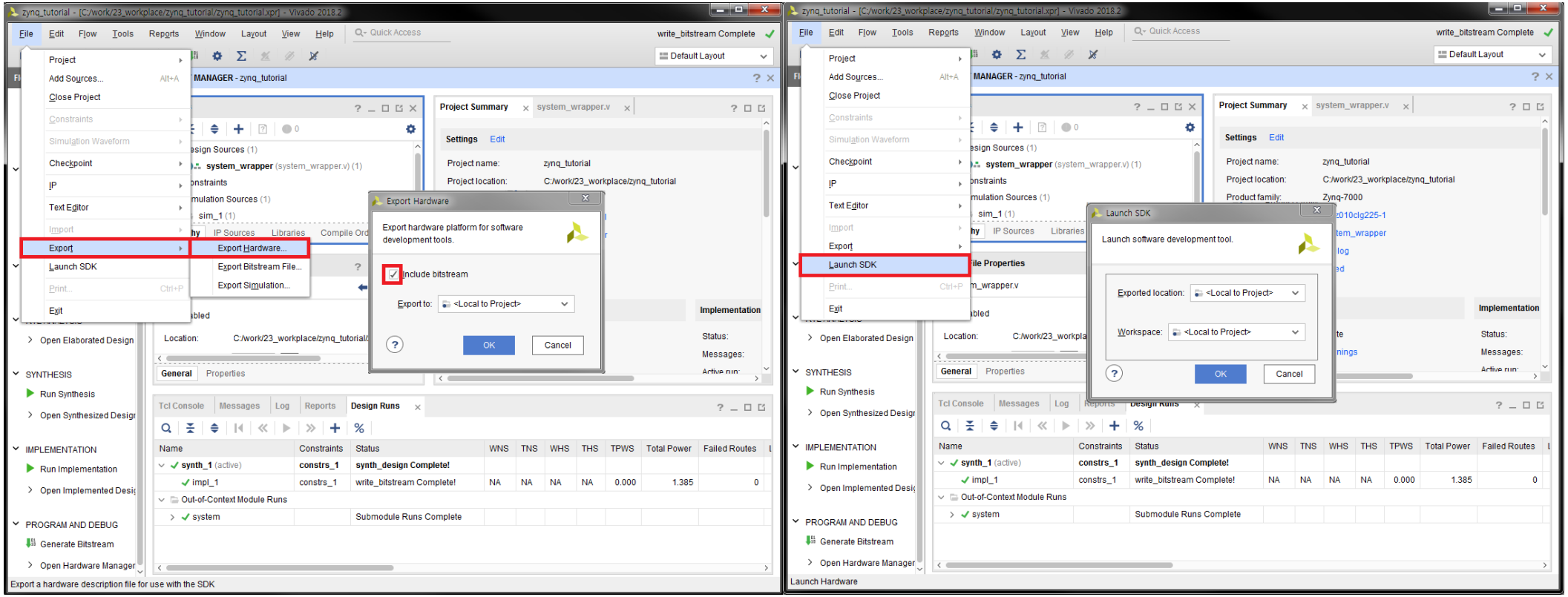


IMPLEMENTATION → Device

PL 영역의 경우 사용된 영역이 없고, PS 영역만 사용됨

HUINS
Human Intelligent System

Export HW & Launch SDK

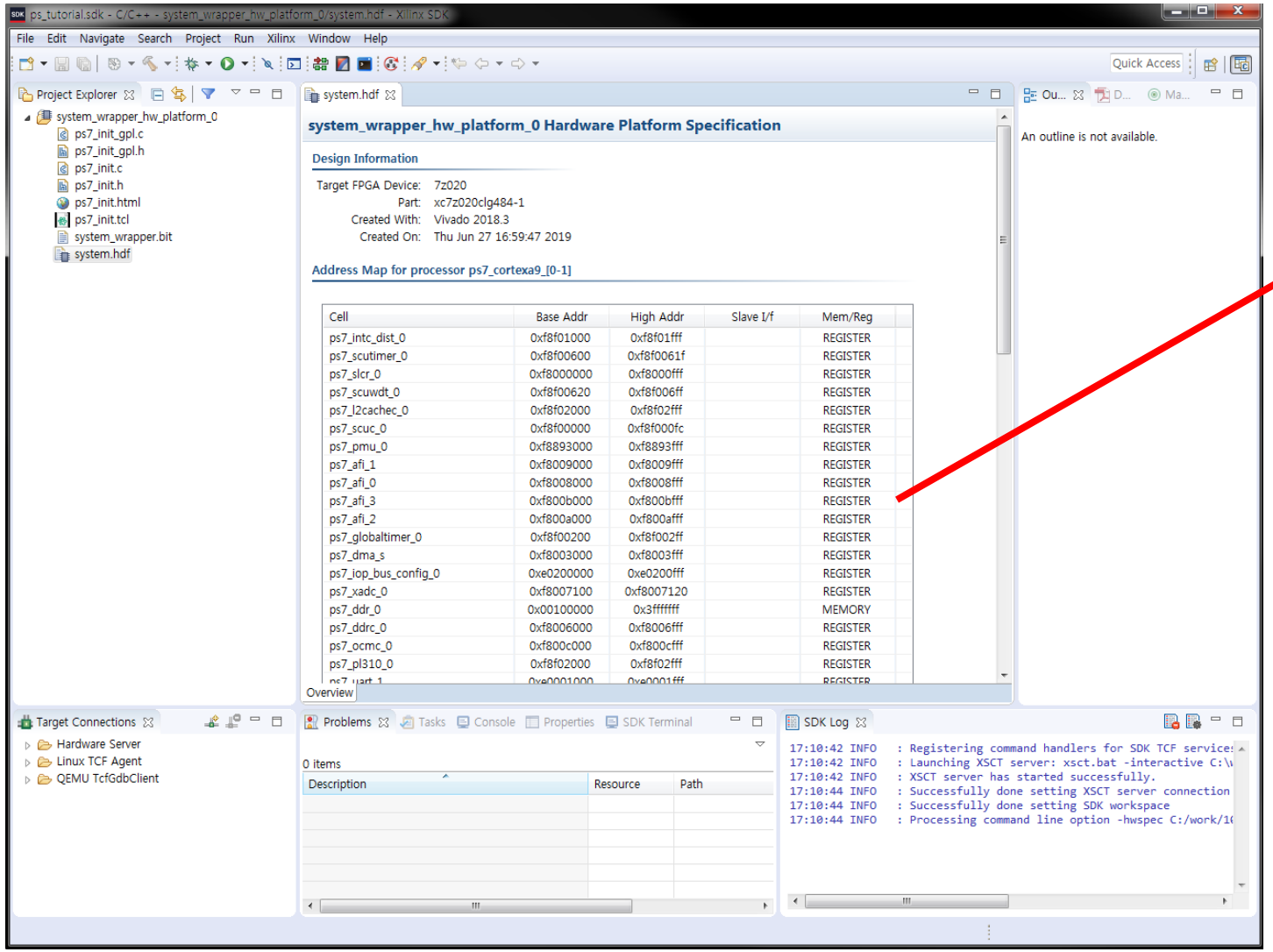


Export Hardware → PS설정 정보 및 bitstream 출력

출력된 하드웨어 정보 기반으로 SDK 실행

HUINS
Human Intelligent System

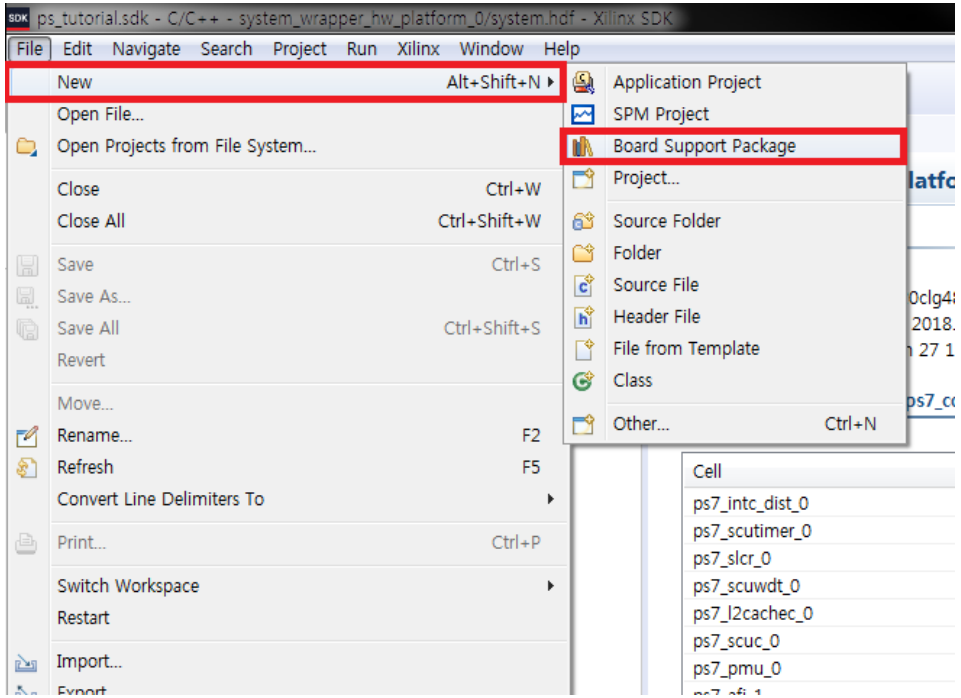
C. 애플리케이션 프로젝트 생성

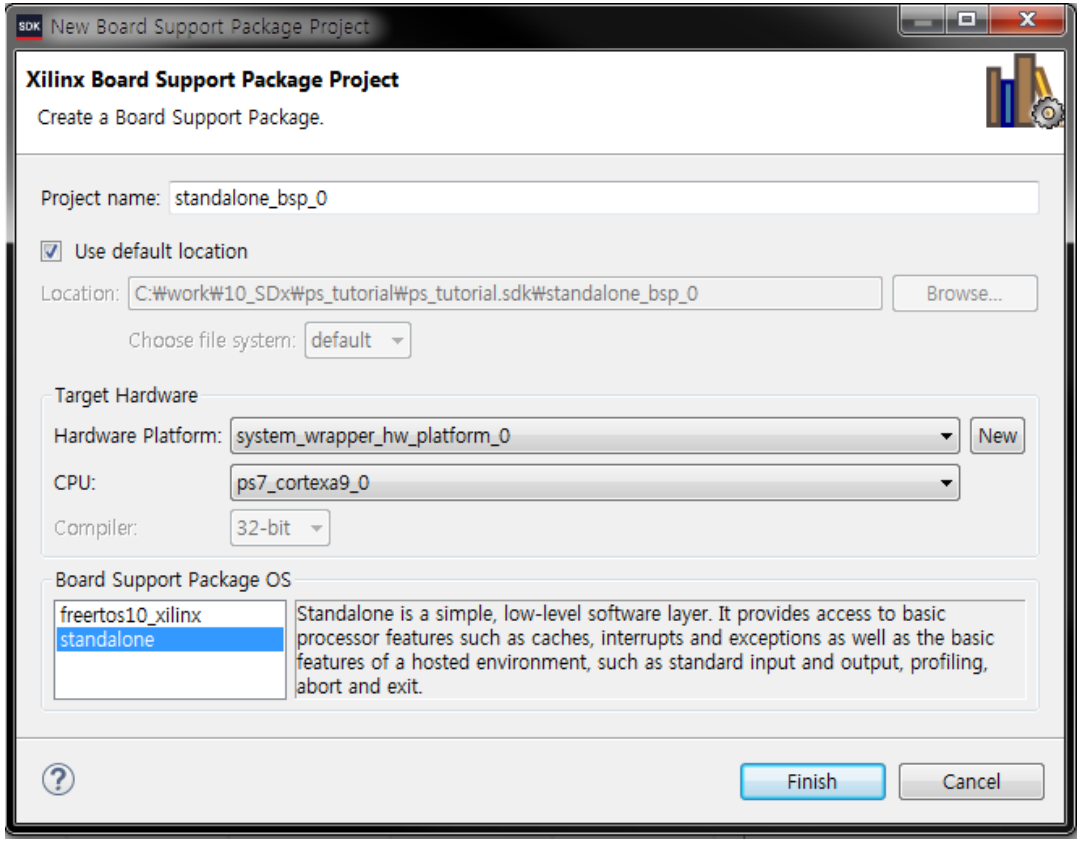


FFFC_0000 to FFFF_FFFF	OCM
FD00_0000 to FFFB_FFFF	Reserved
FC00_0000 to FCFF_FFFF	Quad SPI linear address
F8F0_3000 to FBFF_FFFF	Reserved
F890_0000 to F8F0_2FFF	CPU Private registers
F801_0000 to F88F_FFFF	Reserved
F800_1000 to F880_FFFF	PS System registers,
F800_0C00 to F800_0FFF	Reserved
F800_0000 to F800_0BFF	SLCR Registers
E600_0000 to F7FF_FFFF	Reserved
E100_0000 to E5FF_FFFF	SMC Memory
E030_0000 to E0FF_FFFF	Reserved
E000_0000 to E02F_FFFF	IO Peripherals
C000_0000 to DFFF_FFFF	Reserved
8000_0000 to BFFF_FFFF	PL (MAXI _GP1)
4000_0000 to 7FFF_FFFF	PL (MAXI _GP0)
0010_0000 to 3FFF_FFFF	DDR(address not filtered by SCU)
0004_0000 to 000F_FFFF	DDR(address filtered by SCU)
0000_0000 to 0003_FFFF	OCM

HUINS
Human Intelligent System

C. 애플리케이션 프로젝트 생성



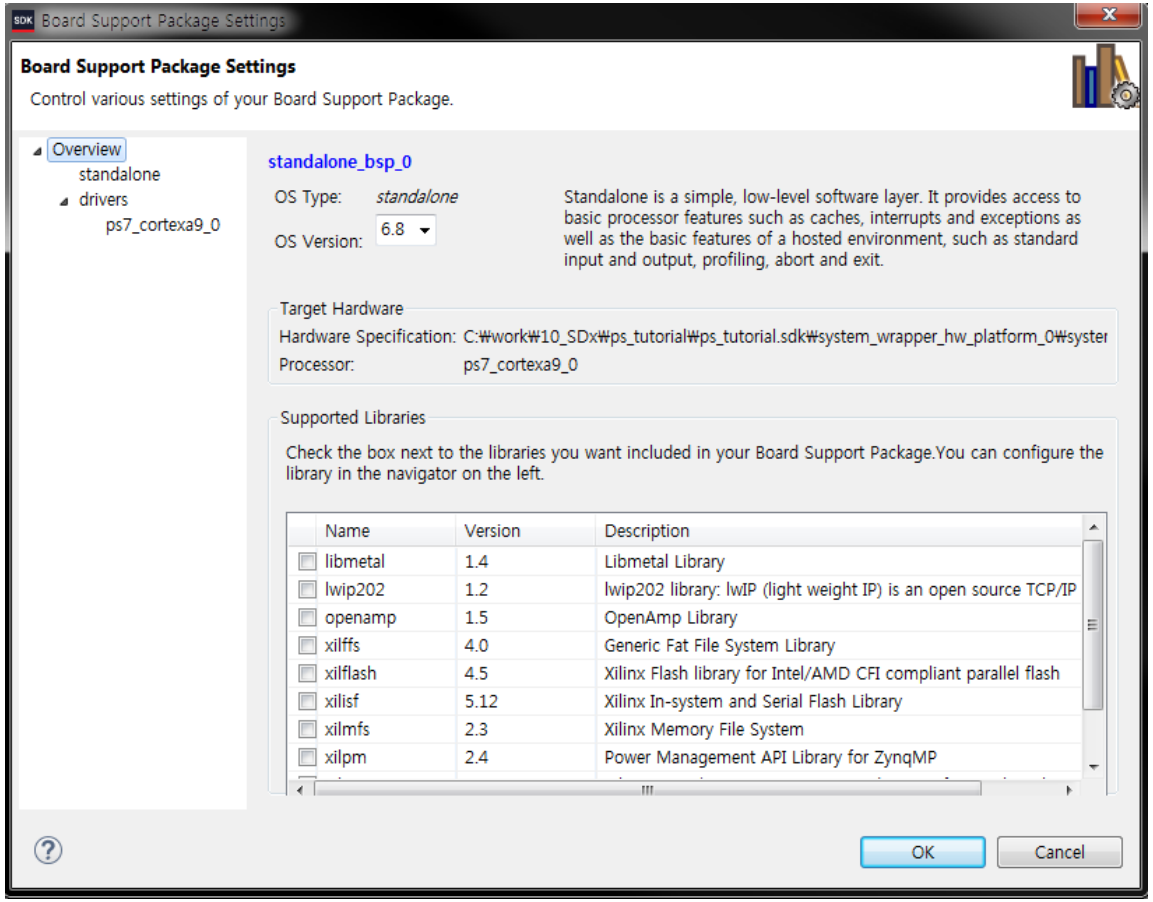


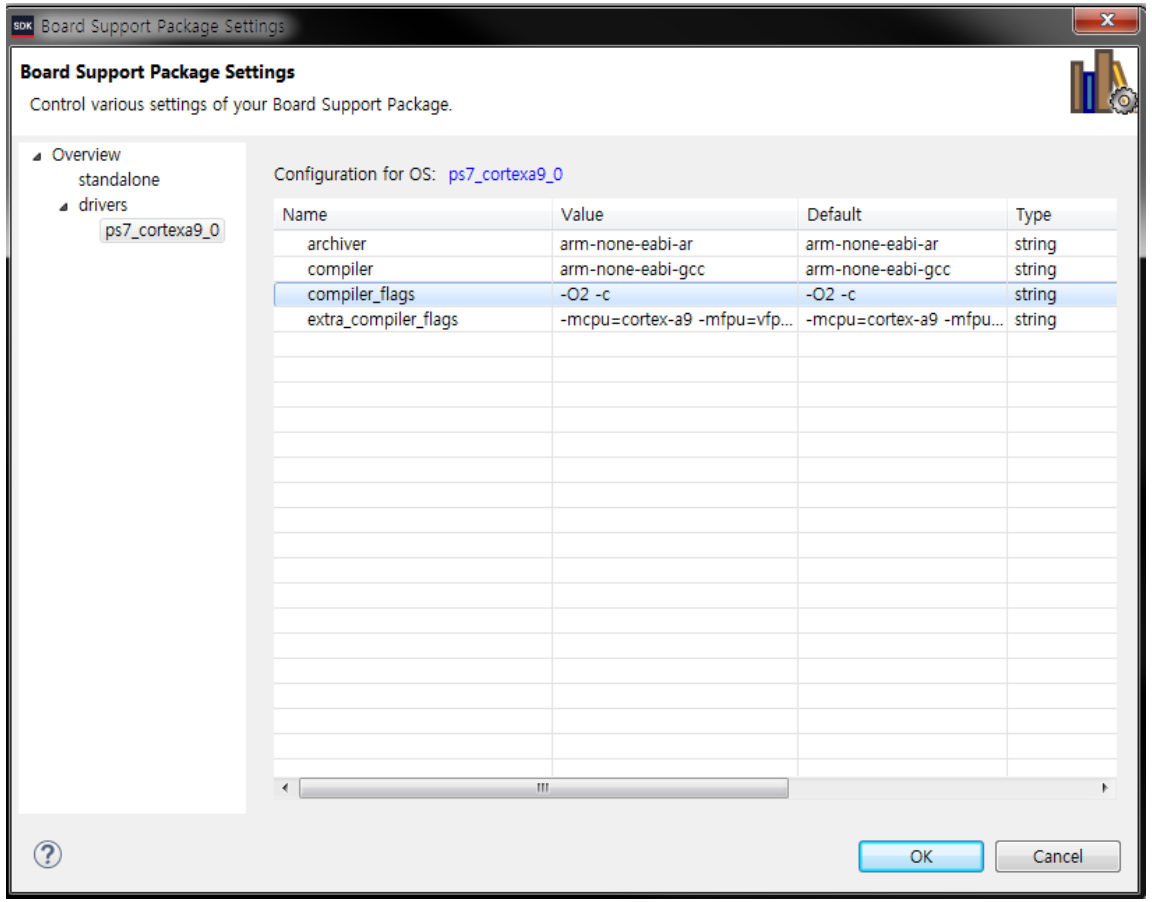
선택한 HW 플랫폼의 BSP 생성

BSP: 장치 드라이버, 예외 처리, 파일 및 메모리 관리

HUINS
Human Intelligent System

C. 애플리케이션 프로젝트 생성



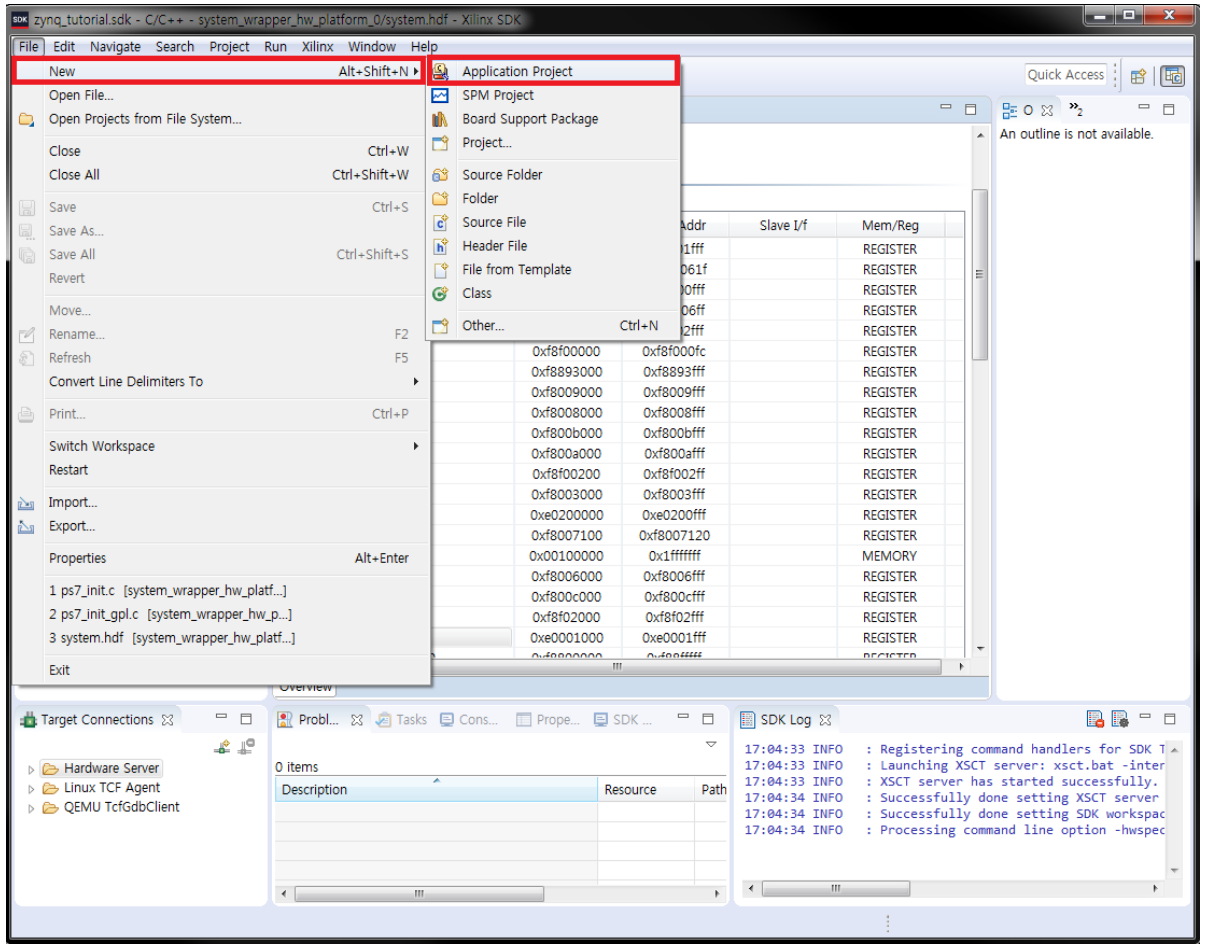


BSP 생성 시 라이브러리 옵션, 컴파일 옵션 등 설정 가능

Default로 OK 클릭

HUINS
Human Intelligent System

C. 애플리케이션 프로젝트 생성



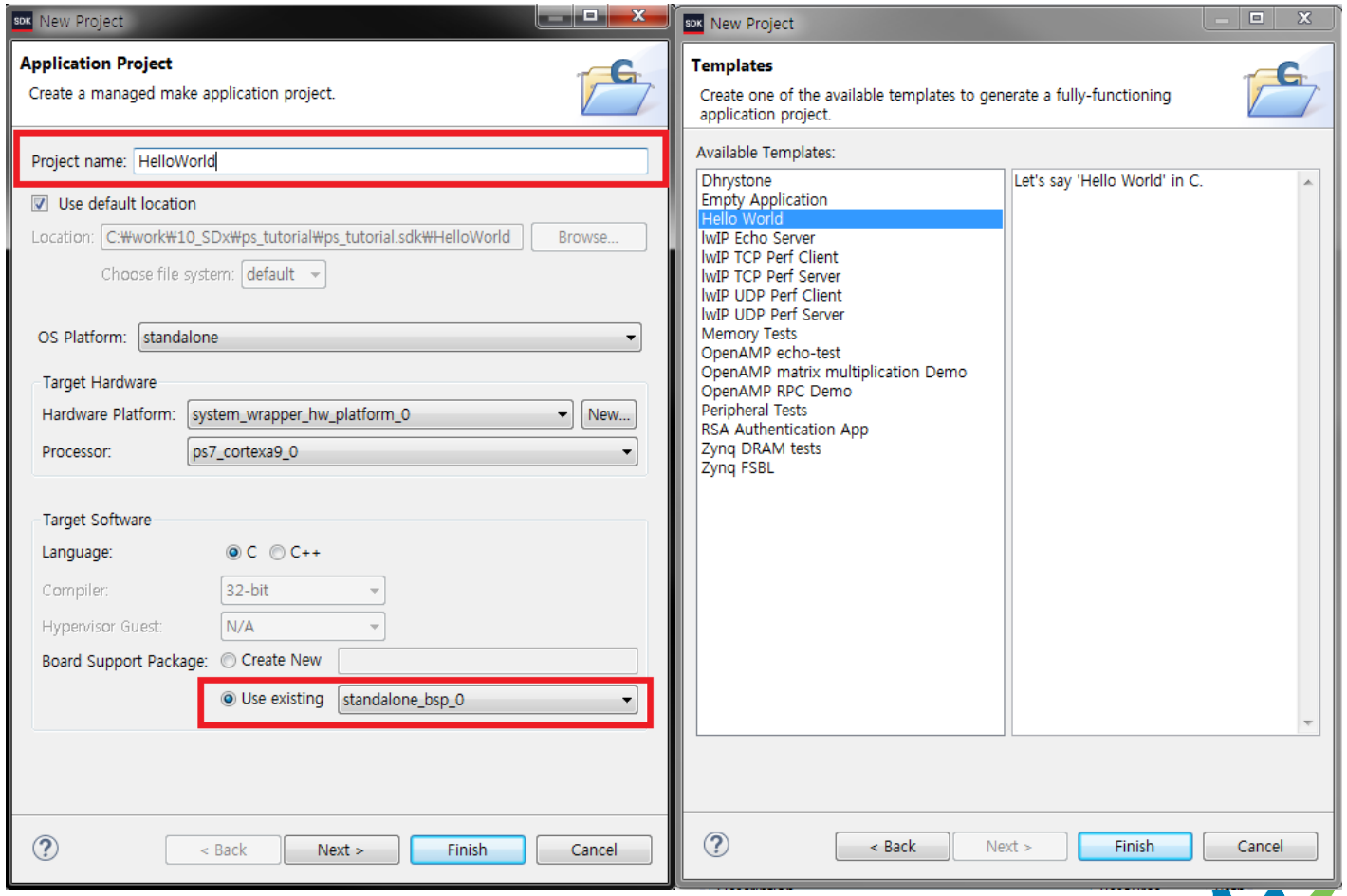
Application Project 생성

HUINS
Human Intelligent System

애플리케이션 프로젝트 생성

프로젝트 생성

- OS: standalone
- Hardware Platform
  → Export된 HW 선택
- Target Language: C
- BSP: standalone_bsp_0
- Template: Hello World



HUINS
Human Intelligent System

애플리케이션 프로젝트 생성

HelloWorld
애플리케이션 생성

HelloWorld.c
→문자열 “Hello World” 출력

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

int main()
{
    init_platform();

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

    cleanup_platform();
    return 0;
}
```

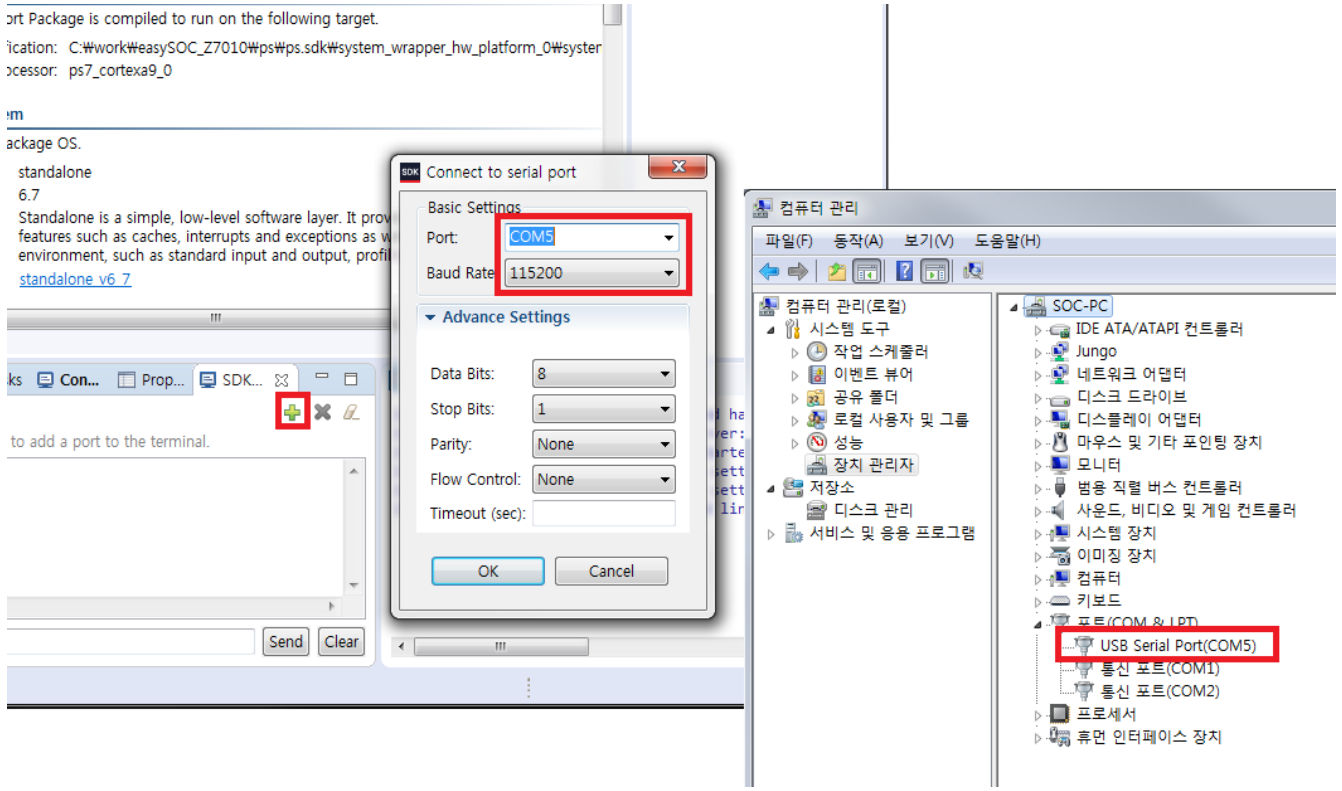
D. “Hello World” 출력

PC와 보드 UART 연결

장치관리자 – 시리얼 포트(USB Serial Port)

터미널 연결

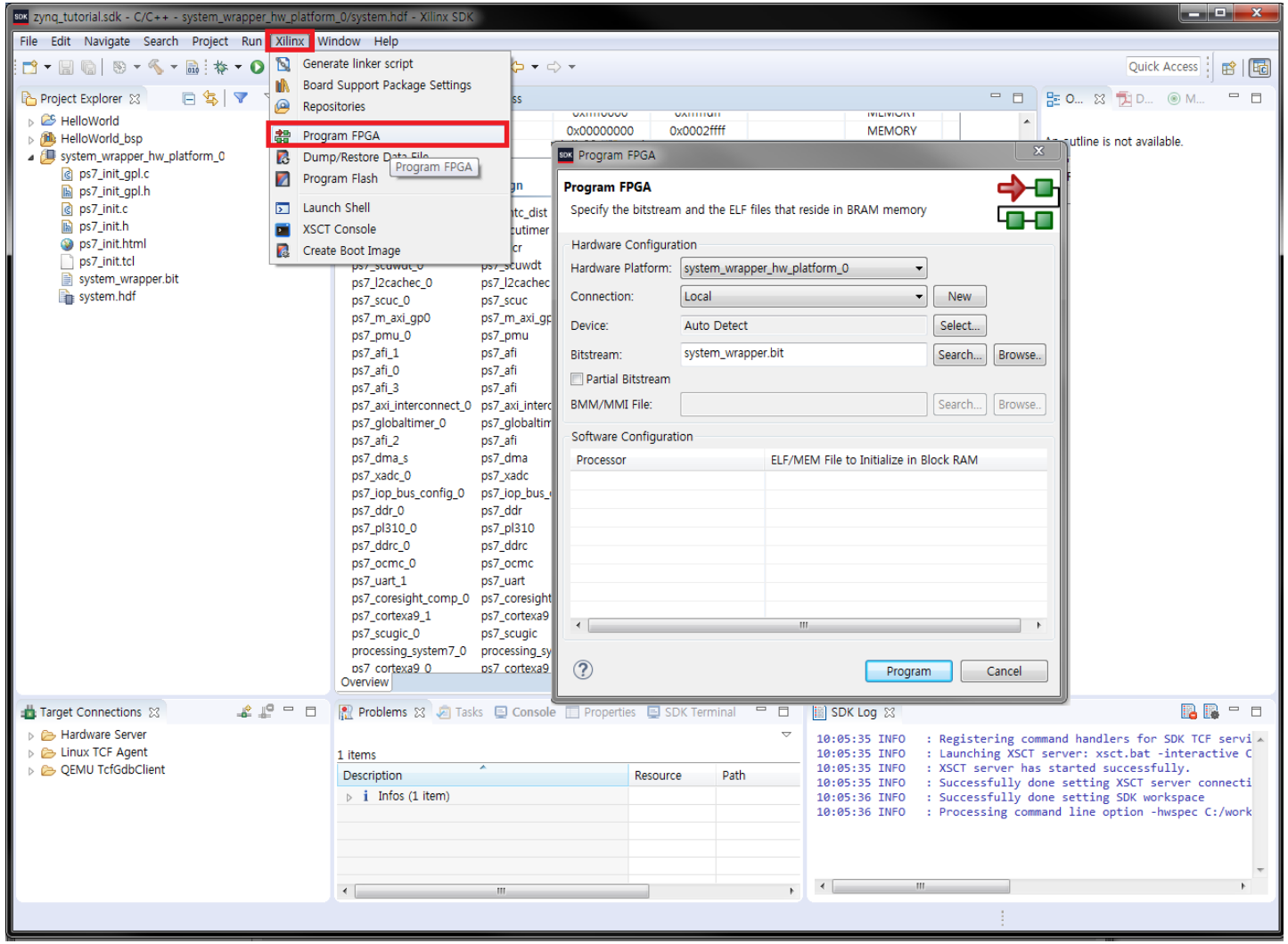
- Baud Rate: 115200
- Data Bits: 8
- Parity: None



HUINS
Human Intelligent System

Program FPGA

Program FPGA
PL에 Bitstream
파일 다운로드

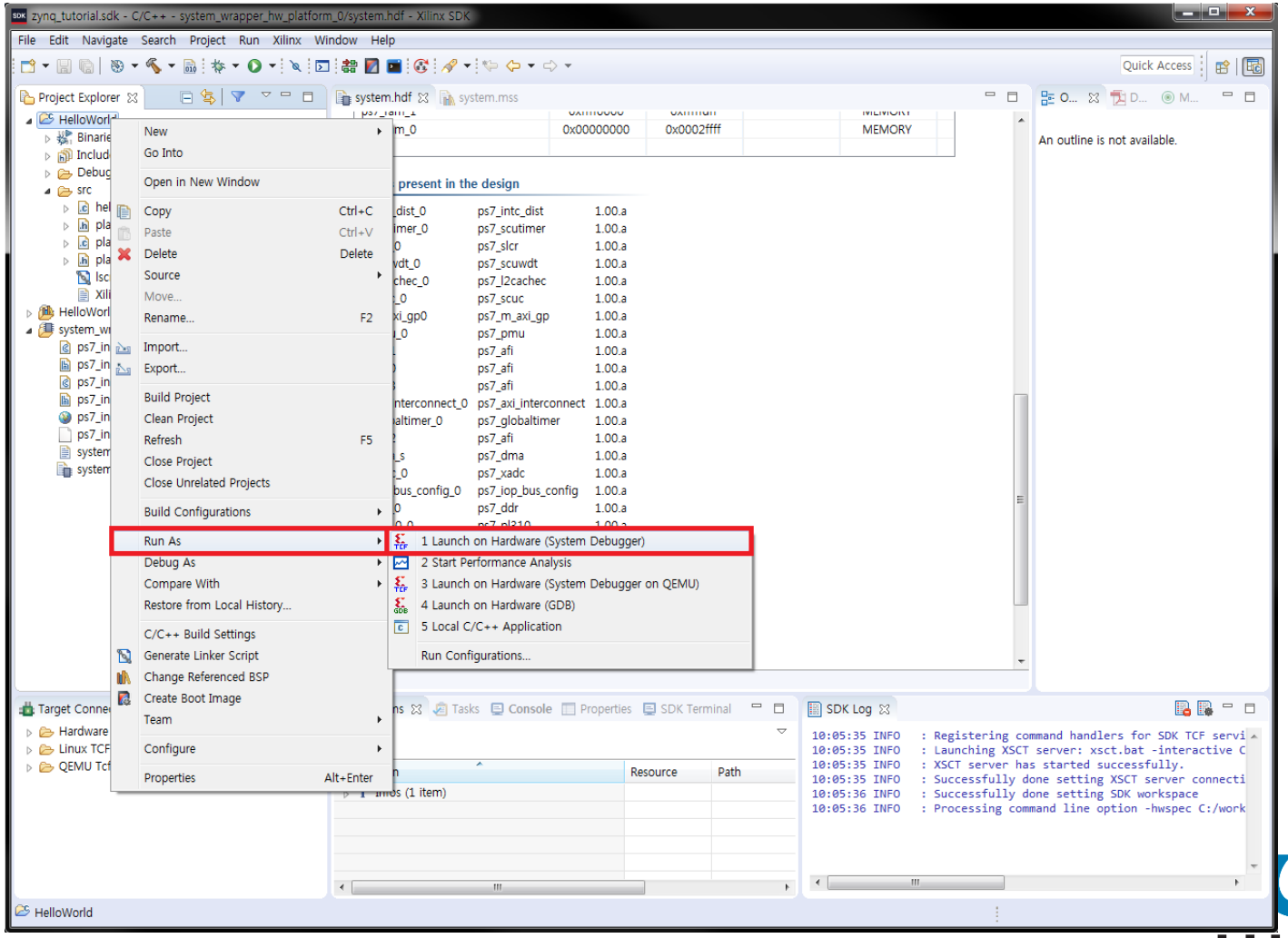


HUINS
Human Intelligent System

애플리케이션 실행

Run As
Launch On Hardware
애플리케이션 실행

디버그 모드 시
Debug As 실행

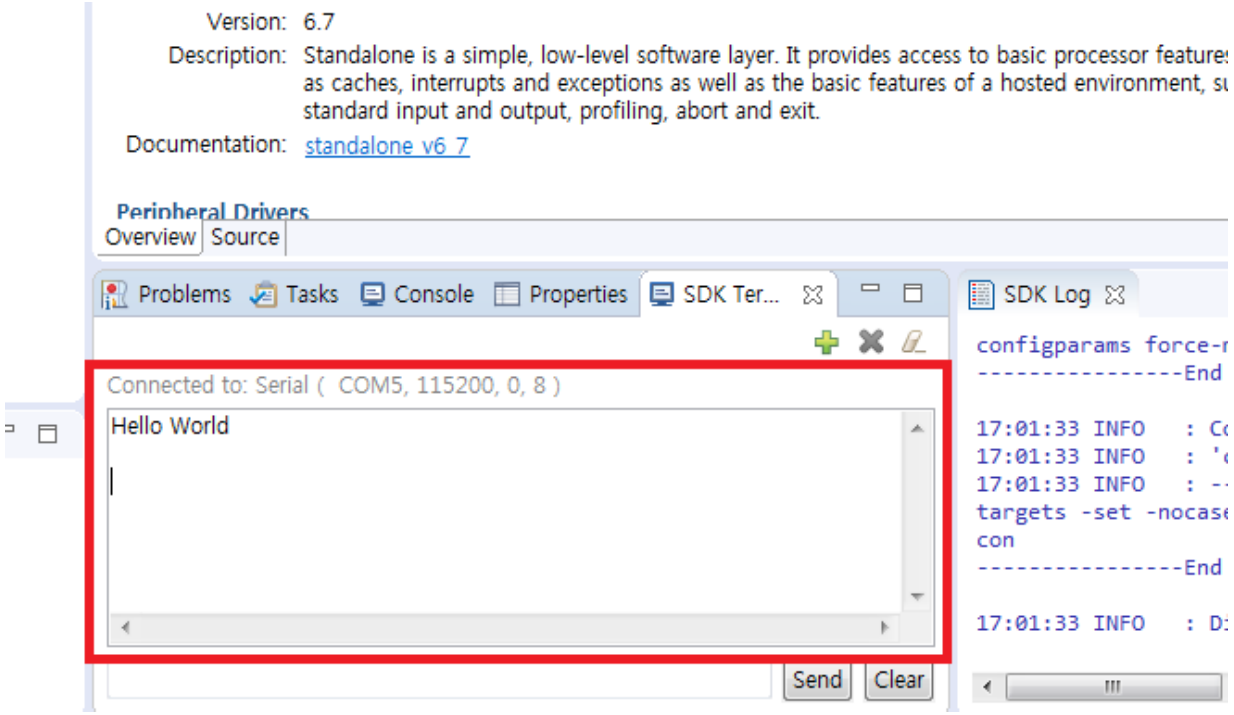


HUINS
Human Intelligent System

애플리케이션 실행

시리얼 터미널 출력
"Hello World"

컴파일 → 실행파일(elf) 생성
프로세서, 주변장치 및
DDR3 메모리 초기화
(0x0010_0000 – 3FFF_FFFF)
애플리케이션 실행



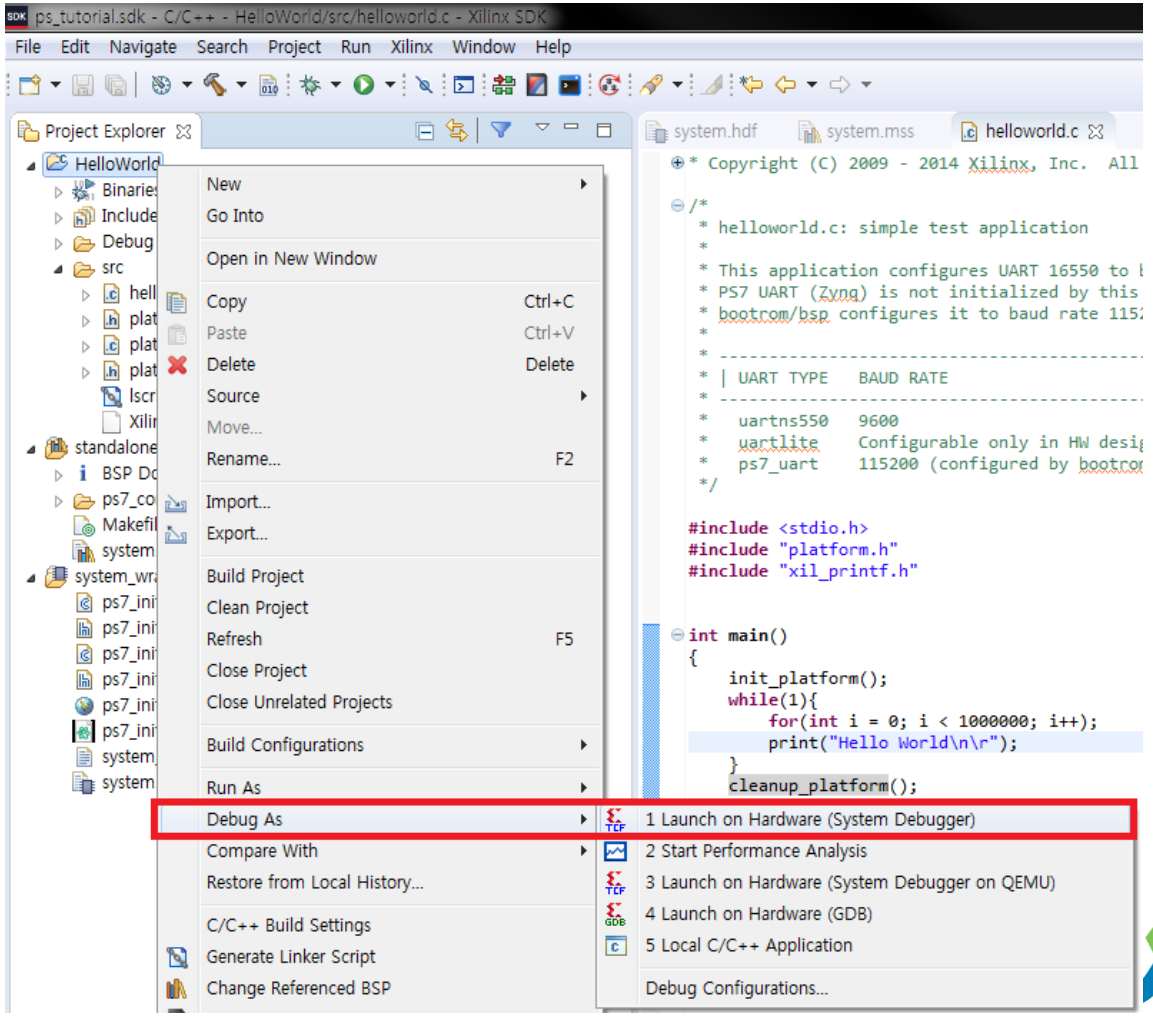
HUINS
Human Intelligent System

애플리케이션 디버그

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

int main()
{
    init_platform();
    while(1){
        for(int i = 0; i < 1000000; i++);
        print("Hello World\n\r");
    }
    cleanup_platform();
    return 0;
}
```

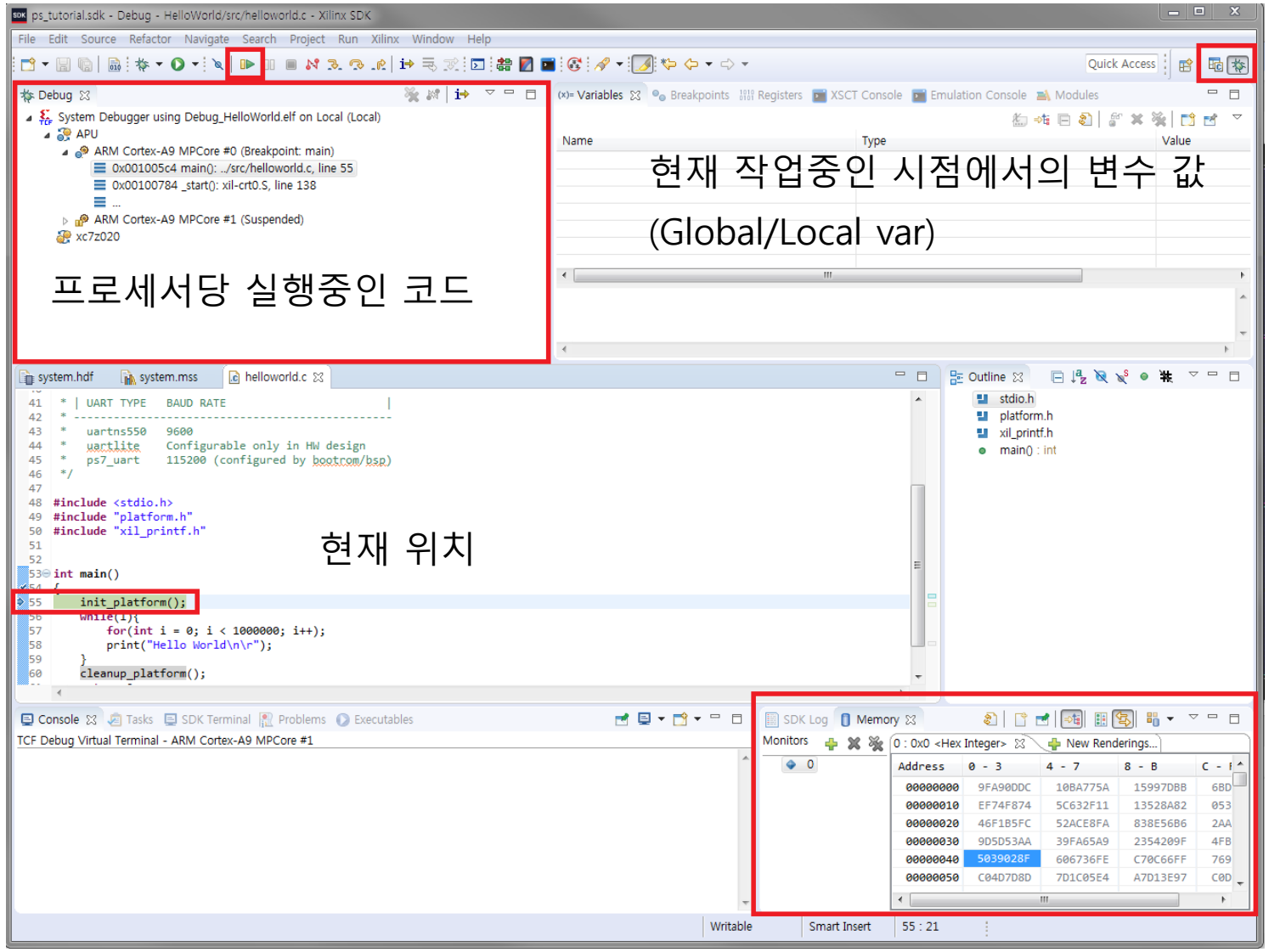
종료되지 않게 소스코드 수정

디버그 모드로 실행



HUINS
Human Intelligent System

애플리케이션 디버그



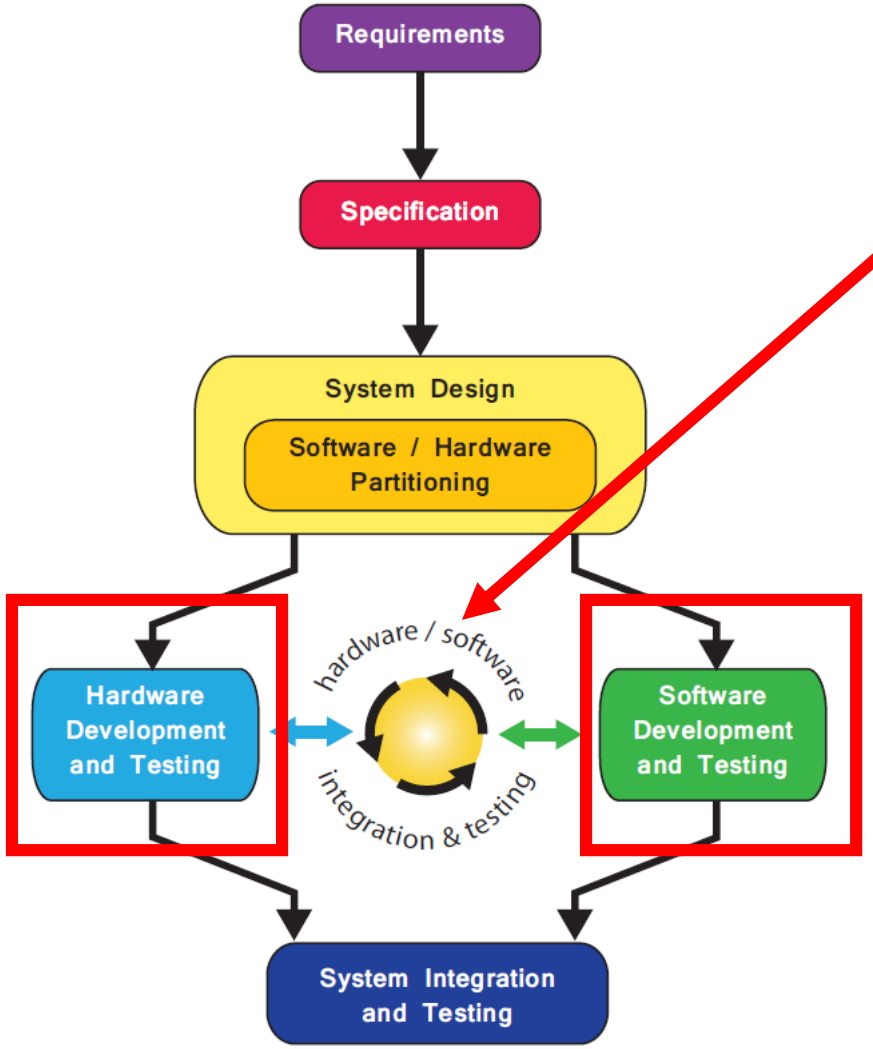
로그 출력

메모리 주소의 값

Zynq SoC 설계

HW + SW
→ AXI Protocol

HW(FPGA)



SW(ARM)

2. S/W & H/W Co-Design Tutorial

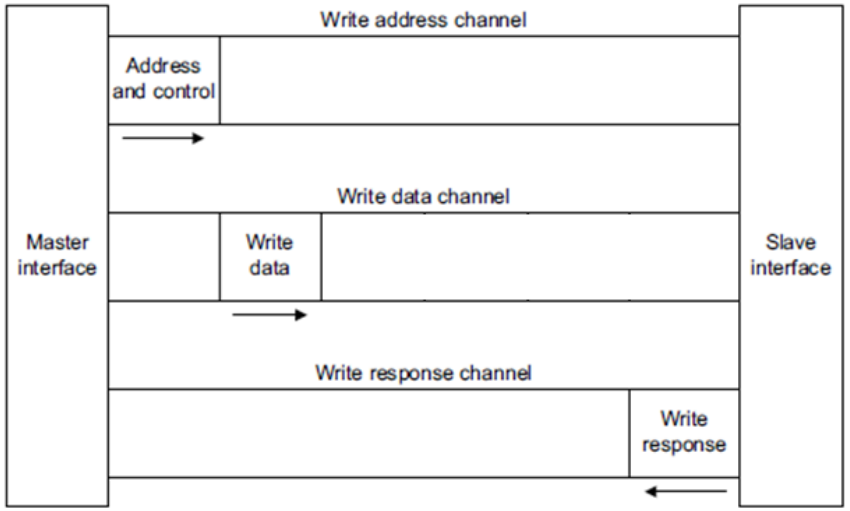
A. AXI-4 Protocol IP 생성
   - 7-Segment → Write Transaction
   - DIP SW → Read Transaction

B. PS 설정(MIO/AXI Interface/DDR Controller)

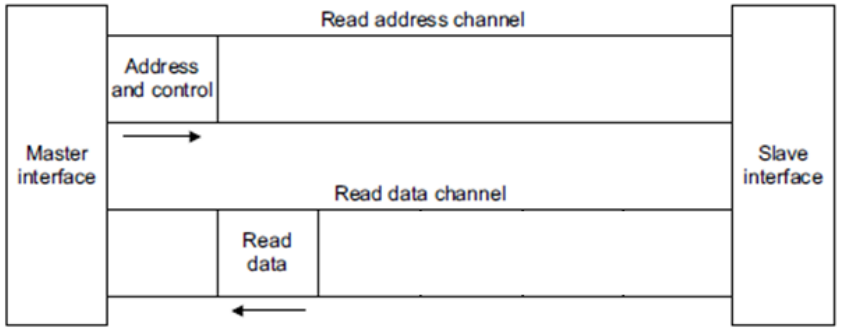
C. PS – PL 연결(Block Design)

D. PS Application → AXI-4 Peripheral IP 제어(Read/Write)

AXI-4 Read/Write



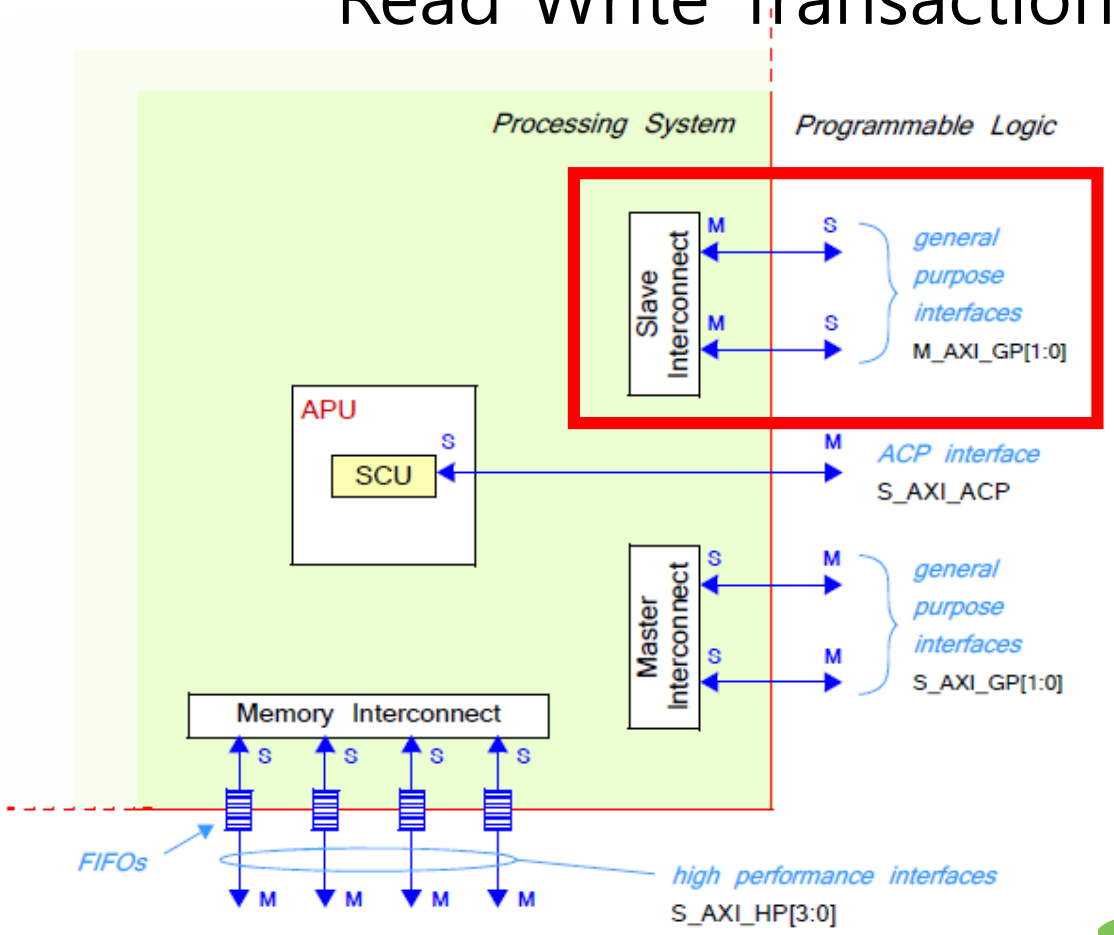
Write



Read

PS(Master)

Read Write Transaction



HUINS
Human Intelligent System

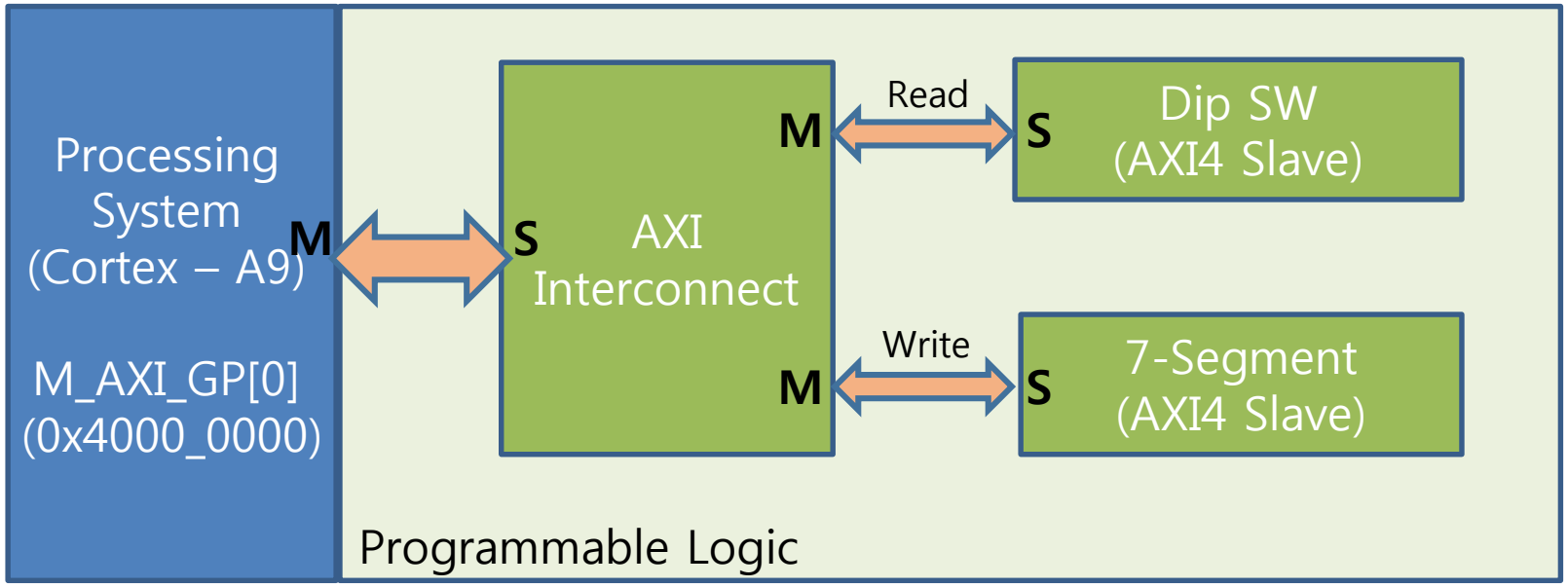
전체 시스템 구성

PL

- AXI4 Slave IP
  - DIP 스위치 (Read)
  - 7-Segment(Write)
- AXI Interconnect
  - M_AXI_GP[0]
  - AXI4 Slave IP

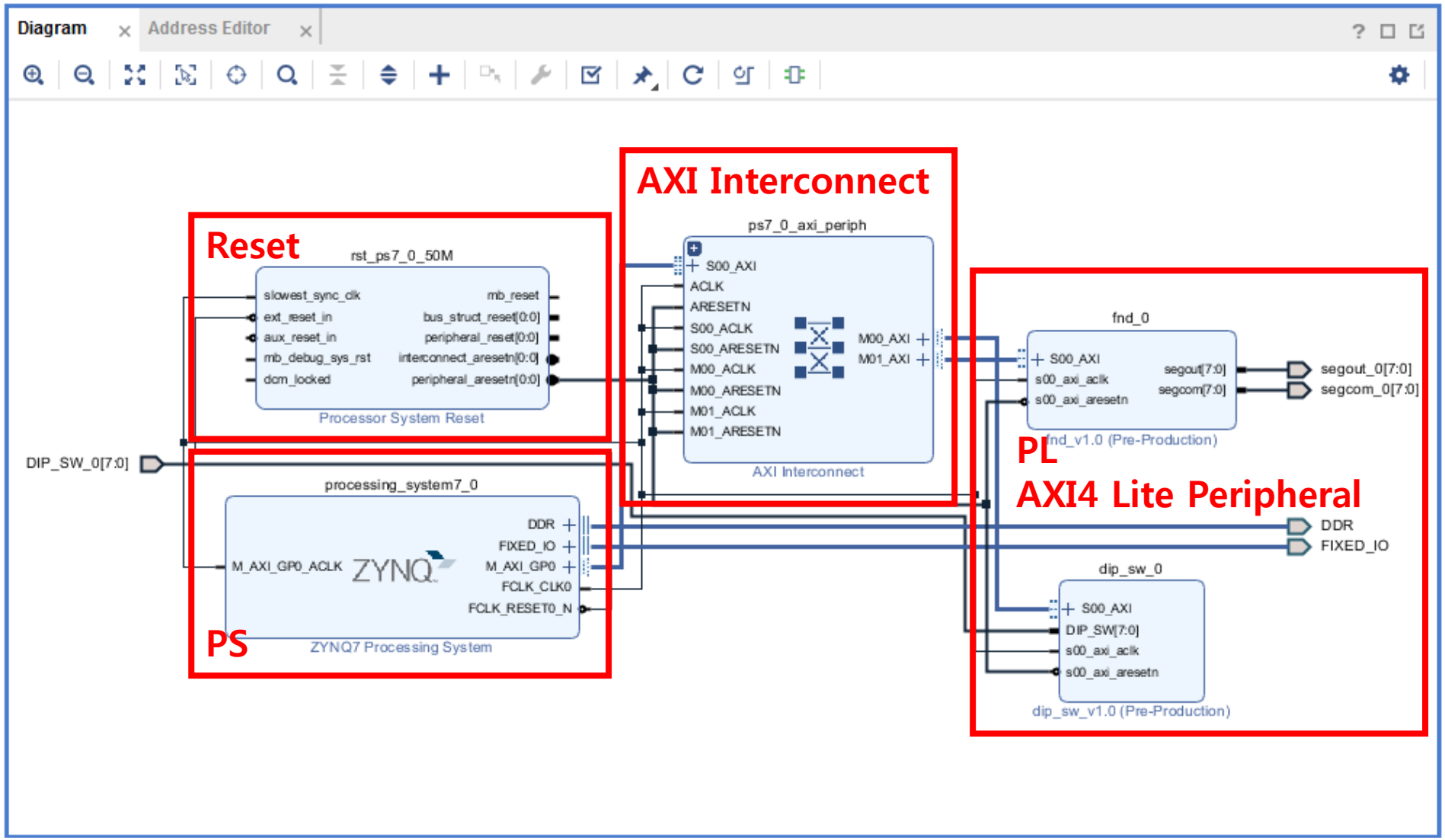
PS

- M_AXI_GP[0]



HUINS
Human Intelligent System

시스템 블록 디자인

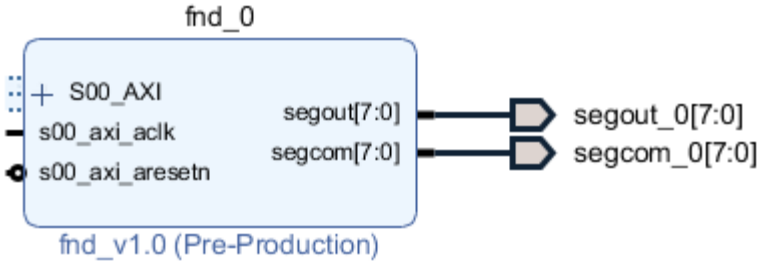


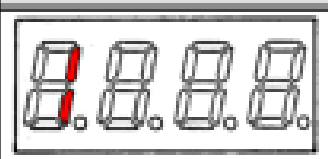
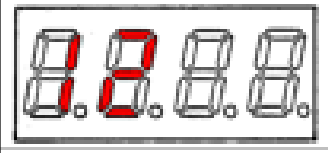
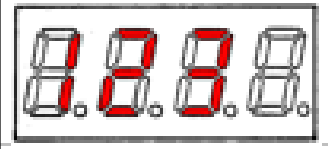
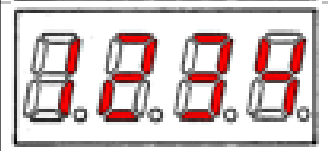
HUINS
Human Intelligent System

A. AXI-4 Protocol IP 생성(7-Segment)

7-Segment
입력된 데이터를 표시

순차적으로 출력
(DIG 1 → 2 → 3 → 4 ..)



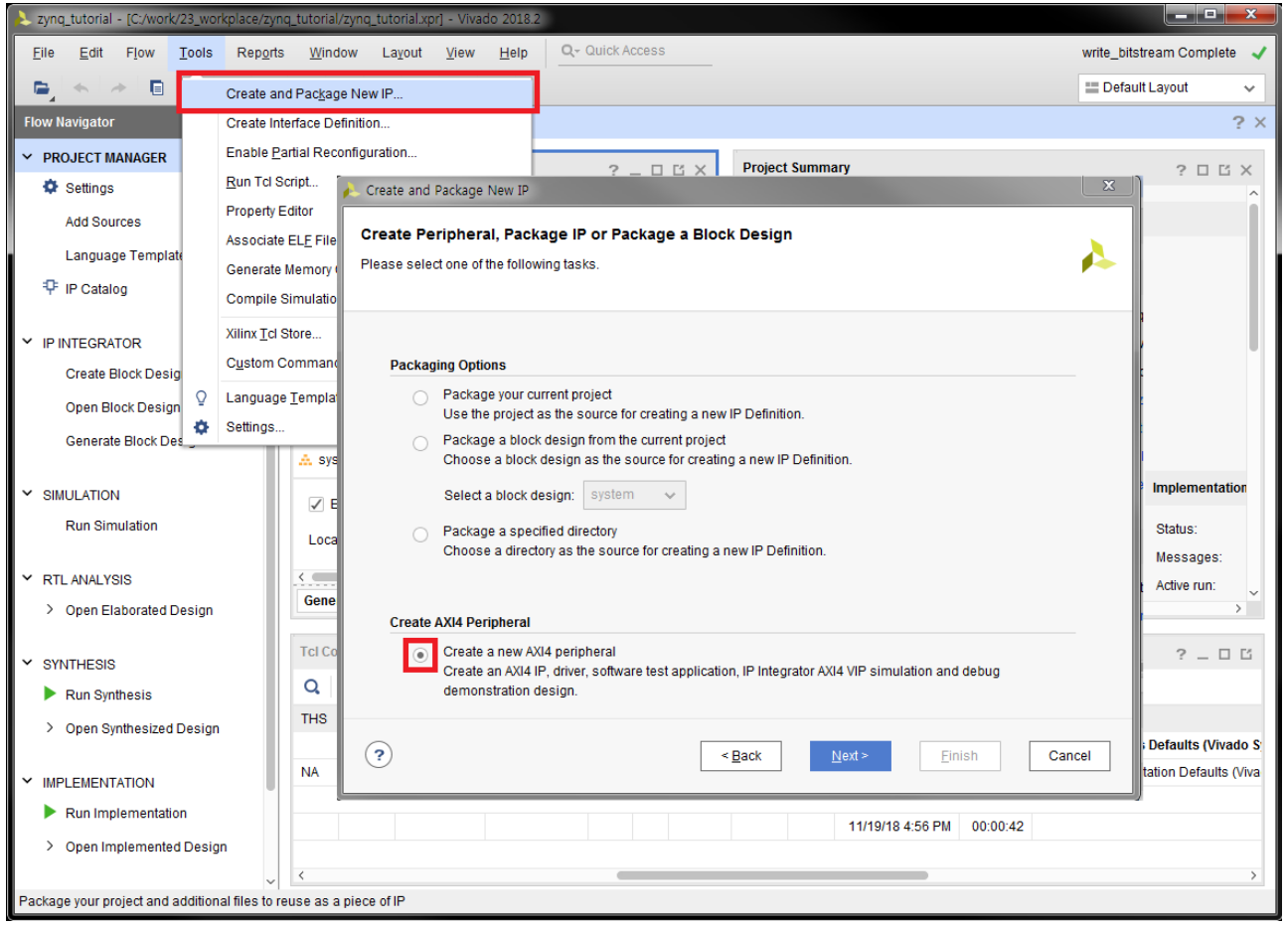
순서	숫자	DATA 설정 값								COM 설정 값			
		A	B	C	D	E	F	G	DP	DIG 1	DIG 2	DIG 3	DIG 4
1		0	1	1	0	0	0	0	0	0	1	1	1
2		1	1	0	1	1	0	1	0	1	0	1	1
3		1	1	1	1	0	0	1	0	1	1	0	1
4		0	1	1	0	0	1	1	0	1	1	1	0

AXI-4 Protocol IP 생성(7-Segment)

Create and
Package New IP 실행

Create a new
AXI4 peripheral
→AXI4 Peripheral IP 생성

AXI4 프로토콜 스켈레톤 IP

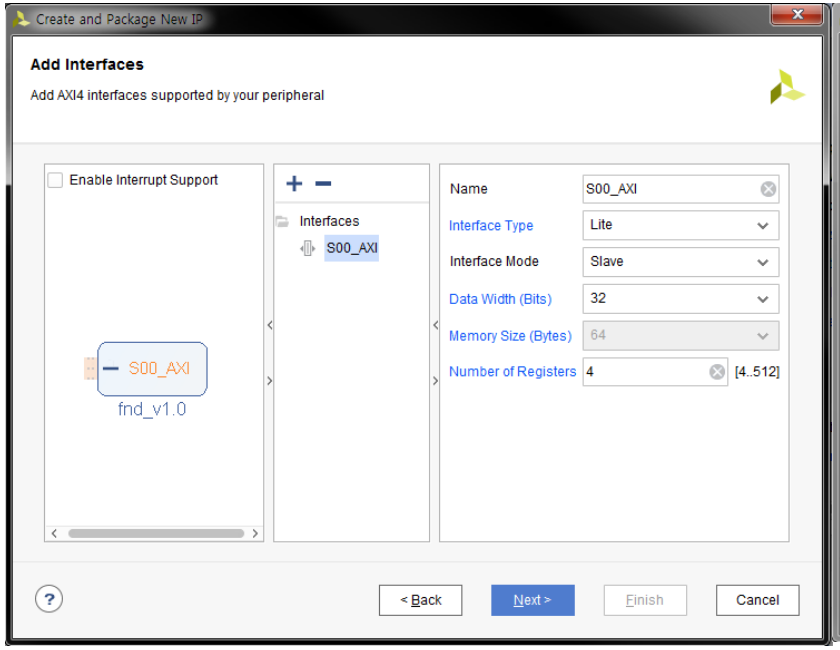


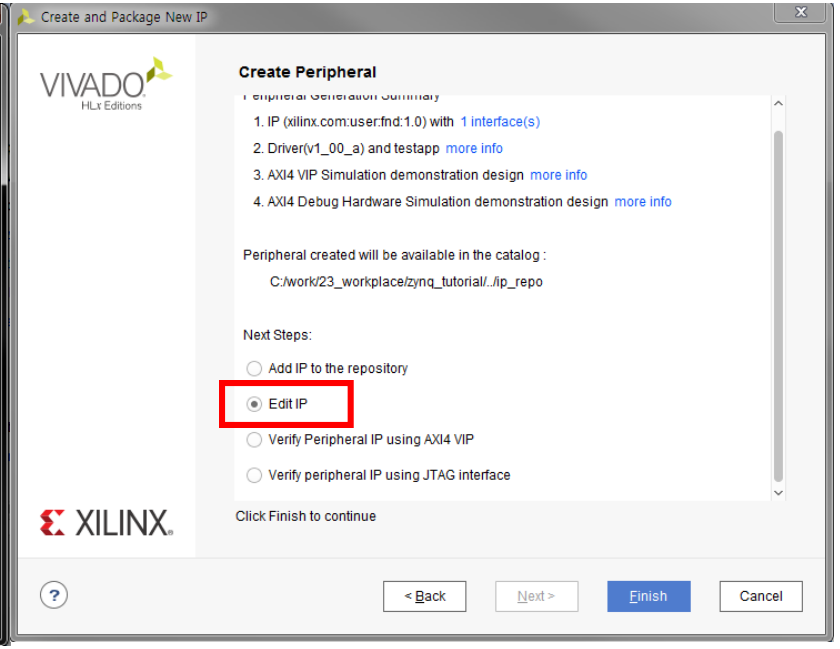
HUINS
Human Intelligent System

AXI-4 Protocol IP 생성(7-Segment)

인터페이스 설정
- Type: AXI4-Lite
- Mode: Slave

AXI 인터페이스 Skeleton IP 생성됨
- Edit-IP 실행→수정





HUINS
Human Intelligent System

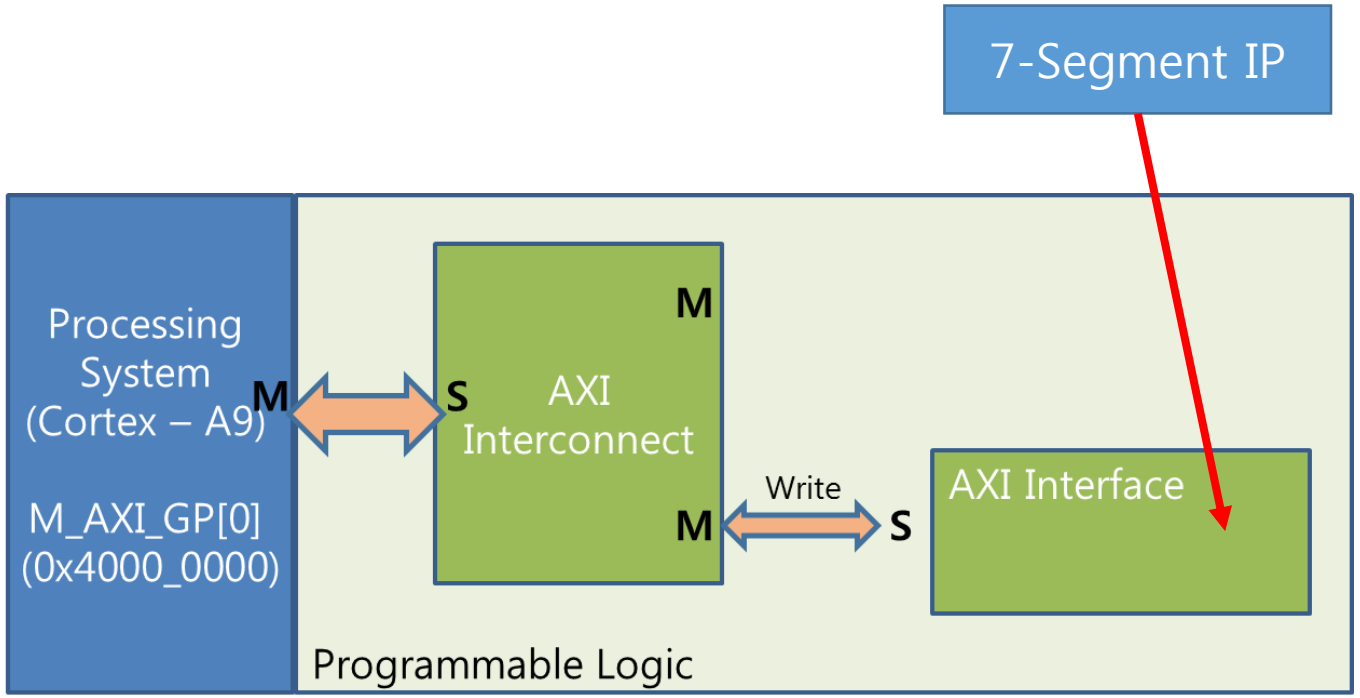
AXI 인터페이스 모듈

IP Packer 프로젝트 생성됨

AXI4-Slave Skeleton IP 생성됨

AXI 인터페이스 모듈 내부에

7-Segment IP를 넣고 연결



HUINS
Human Intelligent System

AXI 인터페이스 모듈

데이터 Width: 32bit

데이터 레지스터 수: 4

```
    // Width of S_AXI data bus
    parameter integer C_S_AXI_DATA_WIDTH = 32,
    // Width of S_AXI address bus
    parameter integer C_S_AXI_ADDR_WIDTH = 4
)
```

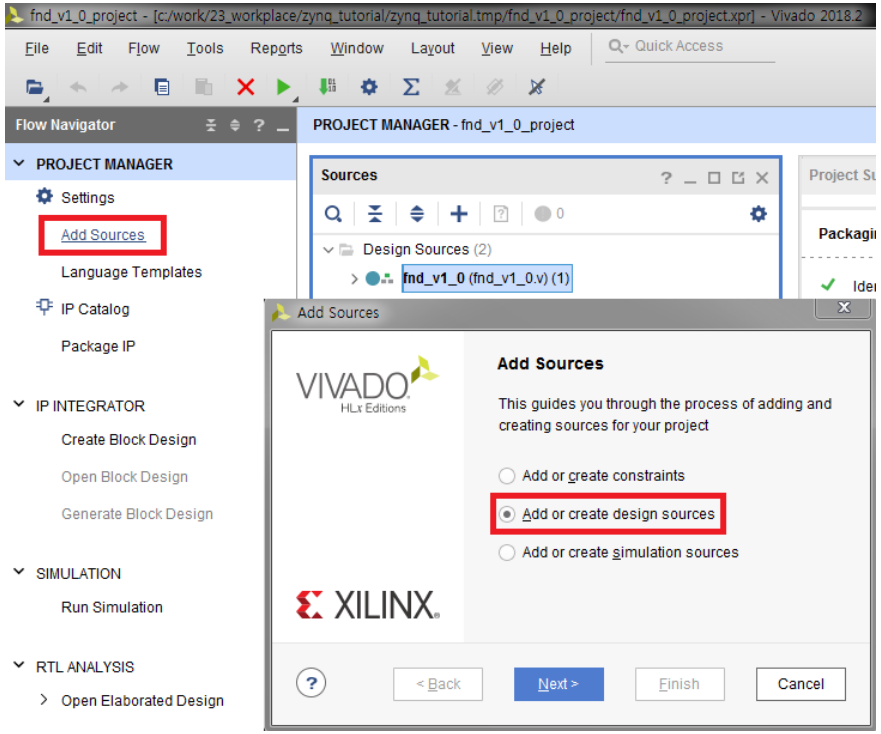
fnd_v_1_0_S00_AXI.v

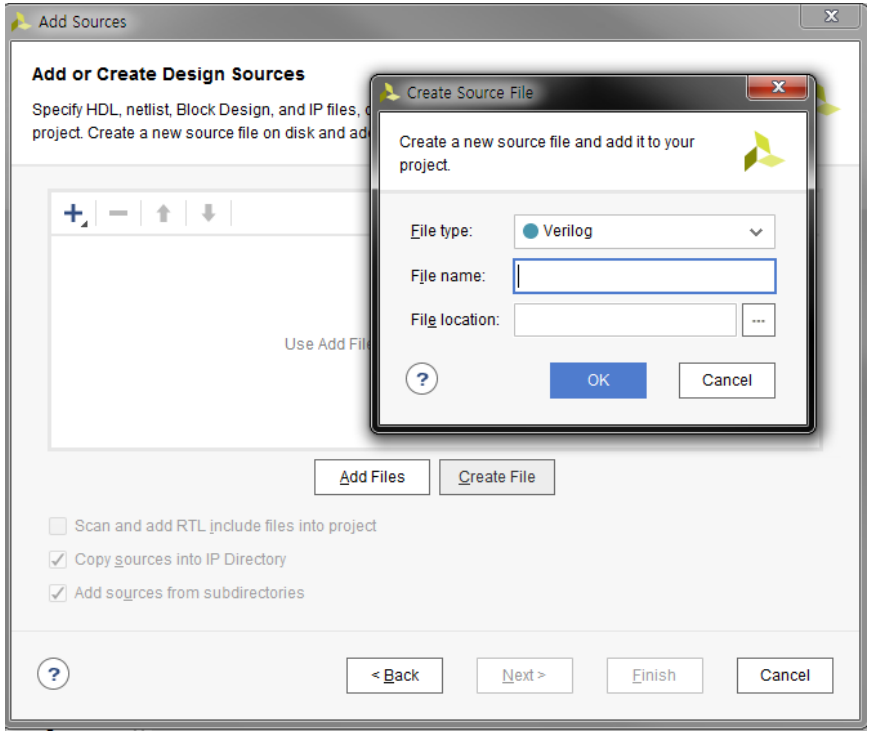
- AXI 인터페이스 모듈
- read/write 위한 hand shake
- write enable시 address에 해당하는 register에 data write

```
230     if (slv_reg_wren)
231       begin
232         case ( axi_awaddr[ADDR_LSB+OPT_MEM_ADDR_BITS:ADDR_LSB] )
233           2'h0:
234             for ( byte_index = 0; byte_index <= (C_S_AXI_DATA_WIDTH/8)-1; byte_index = byte_index+1 )
235               if ( S_AXI_WSTRB[byte_index] == 1 ) begin
236                 // Respective byte enables are asserted as per write strobes
237                 // Slave register 0
238                 slv_reg0[(byte_index*8) +: 8] <= S_AXI_WDATA[(byte_index*8) +: 8];
239               end
240           2'h1:
241             for ( byte_index = 0; byte_index <= (C_S_AXI_DATA_WIDTH/8)-1; byte_index = byte_index+1 )
242               if ( S_AXI_WSTRB[byte_index] == 1 ) begin
243                 // Respective byte enables are asserted as per write strobes
244                 // Slave register 1
245                 slv_reg1[(byte_index*8) +: 8] <= S_AXI_WDATA[(byte_index*8) +: 8];
246               end
```


HUINS
Human Intelligent System

AXI-4 Protocol IP 생성(7-Segment)





AXI4 인터페이스만 구현되어 있으므로
7-Segment를 작동하는 소스코드를 추가한다.
seg_top.v, hex2seg.v, seven_seg.v

HUINS
Human Intelligent System

hex2seg.v

```
module hex2seg(
    input [3:0] hex,
    output reg [7:0] seg_data
);
```

```
always @ (*) begin
    case(hex)
        4'd0 : seg_data <= 8'b11111100 ;    //0
        4'd1 : seg_data <= 8'b01100000 ;    //1
        4'd2 : seg_data <= 8'b11011010 ;    //2
        4'd3 : seg_data <= 8'b11110010 ;    //3
        4'd4 : seg_data <= 8'b01100110 ;    //4
        4'd5 : seg_data <= 8'b10110110 ;    //5
        4'd6 : seg_data <= 8'b10111110 ;    //6
        4'd7 : seg_data <= 8'b11100100 ;    //7
        4'd8 : seg_data <= 8'b11111110 ;    //8
        4'd9 : seg_data <= 8'b11110110 ;    //9
        4'd10 : seg_data <= 8'b00000010 ;   //-
        default : seg_data <= 8'b00111010;
    endcase
end
```

입력된 Hex값을 Segment data로 변환하는 모듈

seven_seg.v

```
module seven_seg(
    input clk_in,
    input reset_n,
    input [31:0] data,
    output reg [7:0] segout,
    output reg [7:0] segcom
);

wire [7:0] seg1;
wire [7:0] seg2;
wire [7:0] seg3;
wire [7:0] seg4;
wire [7:0] seg5;
wire [7:0] seg6;
wire [7:0] seg7;
wire [7:0] seg8;

reg[15:0] clk_cnt;
reg[2:0] com_cnt;
hex2seg hex2seg_1 (.hex(data[31:28]), .seg_data(seg1));
hex2seg hex2seg_2 (.hex(data[27:24]), .seg_data(seg2));
hex2seg hex2seg_3 (.hex(data[23:20]), .seg_data(seg3));
hex2seg hex2seg_4 (.hex(data[19:16]), .seg_data(seg4));
hex2seg hex2seg_5 (.hex(data[15:12]), .seg_data(seg5));
hex2seg hex2seg_6 (.hex(data[11:8]), .seg_data(seg6));
hex2seg hex2seg_7 (.hex(data[7:4]), .seg_data(seg7));
hex2seg hex2seg_8 (.hex(data[3:0]), .seg_data(seg8));
```

```
always @ (negedge reset_n or posedge clk_in) begin
    if (!reset_n) begin
        clk_cnt <= 16'd0;
        com_cnt <= 3'd0;
    end
    else begin
        if (clk_cnt == 16'd32768)
            begin
                clk_cnt <= 16'd0;
                if (com_cnt == 3'd7) com_cnt <= 3'd0;
                else com_cnt <= com_cnt + 3'd1;
            end
        else begin
            clk_cnt <= clk_cnt + 16'd1;
        end
    end
end
```

입력된 Data값 Segment값으로 변경

클럭 입력을 분주

HUINS
Human Intelligent System

seven_seg.v

```
always @ (*) begin
    case (com_cnt)
        3'd0: begin
            segcom = 8'b10000000;
            segout = ~seg1; end
        3'd1: begin
            segcom = 8'b01000000;
            segout = ~seg2; end
        3'd2: begin
            segcom = 8'b00100000;
            segout = ~seg3; end
        3'd3: begin
            segcom = 8'b00010000;
            segout = ~seg4; end
        3'd4: begin
            segcom = 8'b00001000;
            segout = ~seg5; end
        3'd5: begin
            segcom = 8'b00000100;
            segout = ~seg6; end
        3'd6: begin
            segcom = 8'b00000010;
            segout = ~seg7; end
        default: begin
            segcom = 8'b00000001;
            segout = ~seg8; end
    endcase
end

endmodule
```

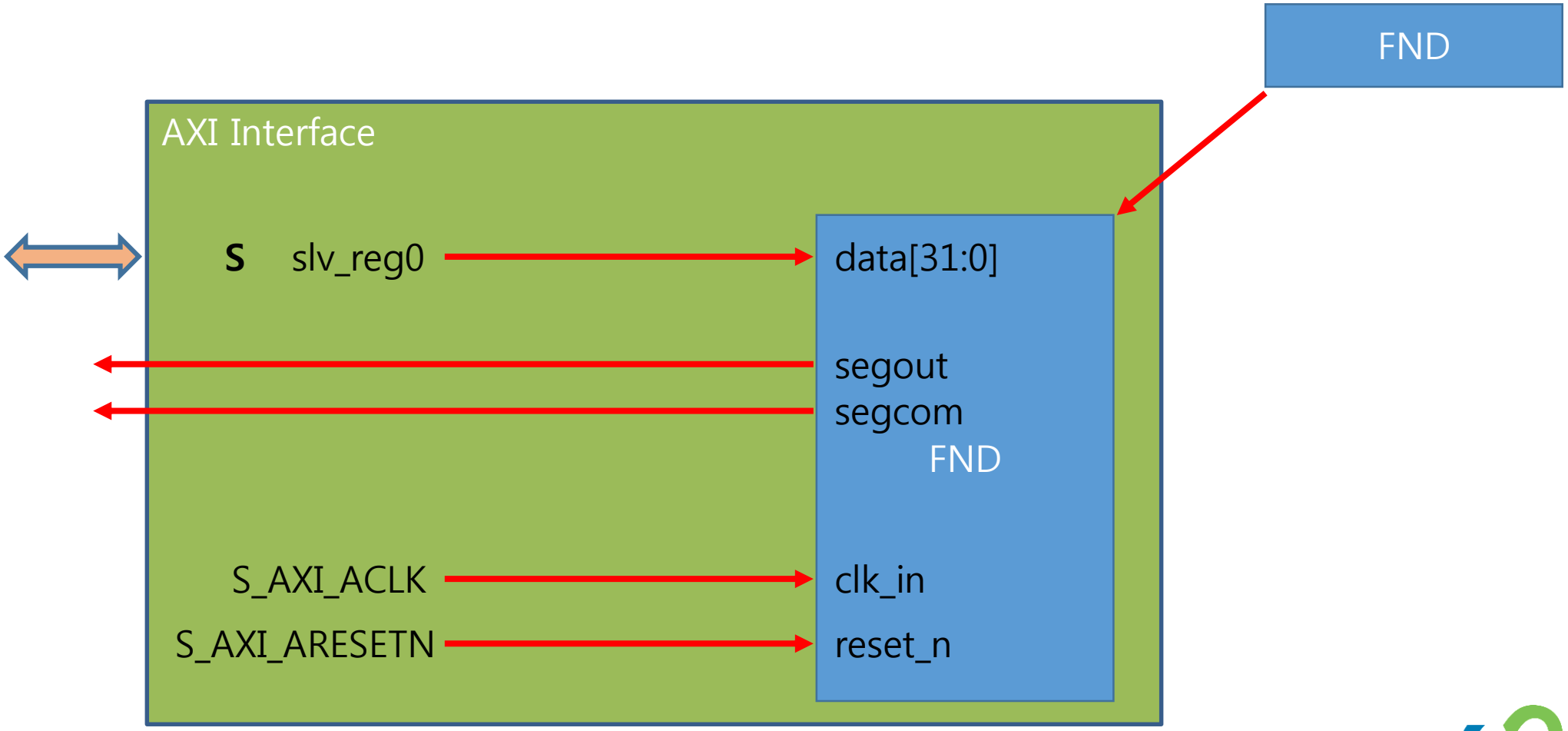
8개의 7-segment를 번갈아 동작

seg_top.v

외부 입력/출력 연결

```
module seg_top(
    input clk_in,
    input reset_n,
    input [31:0] data,
    output [7:0] segout,
    output [7:0] segcom
);

seven_seg seven_seg_u (
    .clk_in(clk_in),
    .reset_n(reset_n),
    .data(data),
    .segout(segout),
    .segcom(segcom)
);

endmodule
```

AXI 인터페이스 모듈 수정



HUINS
Human Intelligent System

AXI 인터페이스 모듈 수정

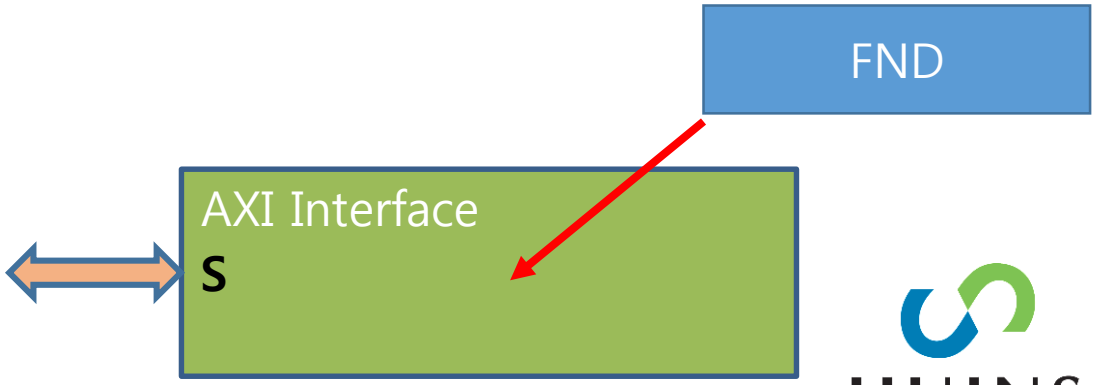
AXI 인터페이스 모듈
fnd_v1_0_S00_AXI.v

7-Segment 모듈 선언
출력 포트 선언

AXI Interface의 slv_reg0와
7-Segment 연결

```
17          // Users to add ports here
18          output wire [7:0] segout,
19          output wire [7:0] segcom,
20          // User ports ends
```

```
401     // Add user logic here
402     seg_top seg_top_inst(
403         .clk_in(S_AXI_ACLK),        //AXI CLK
404         .reset_n(S_AXI_ARESETN),    //AXI Negative Reset
405         .data(slv_reg0),            //Slave Register 0
406         .segout(segout),
407         .segcom(segcom)
408     );
409     // User logic ends
```



FND AXI IP 탑 모듈 수정

FND AXI IP 최상위 모듈
fnd_v1_0.v

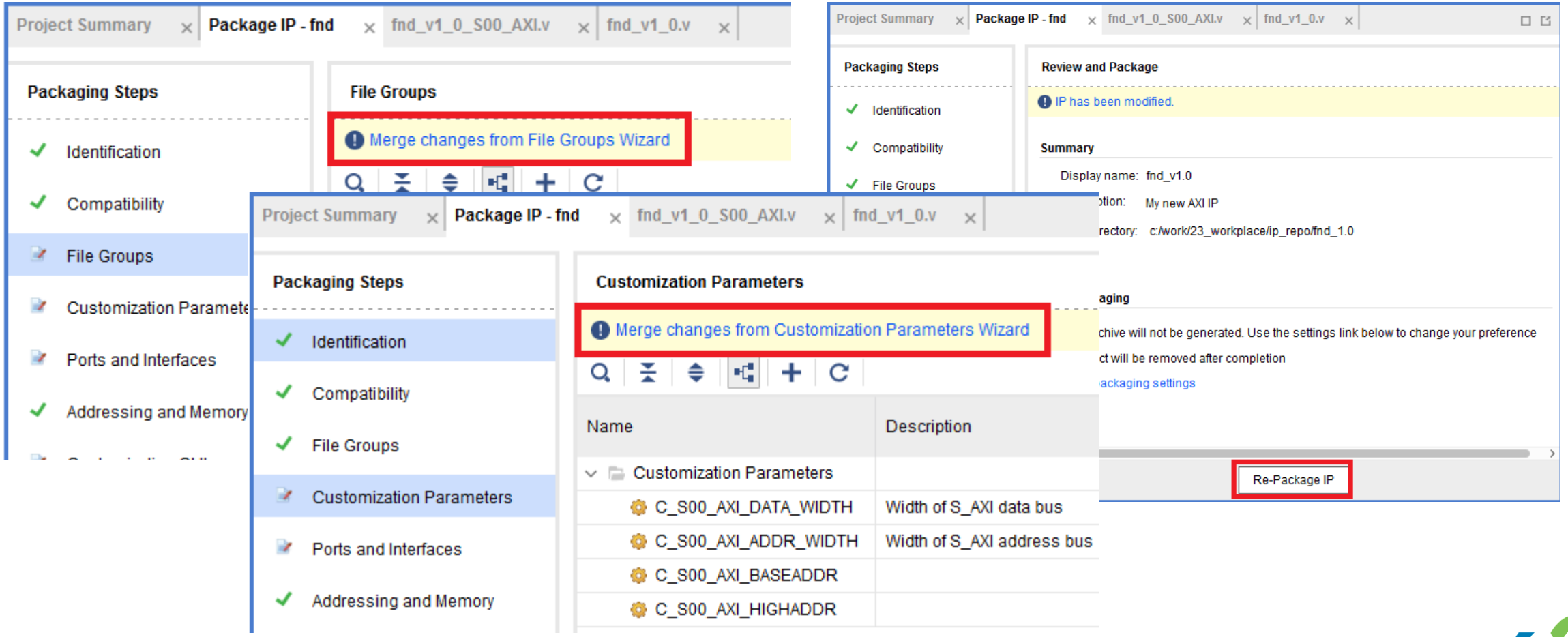
인터페이스 모듈에서
선언된 포트 선언, 외부 연결

```
17          // Users to add ports here
18          output wire [7:0] segout,
19          output wire [7:0] segcom,
20          // User ports ends
```

```
47  // Instantiation of Axi Bus Interface S00_AXI
48      fnd_v1_0_S00_AXI # (
49          .C_S_AXI_DATA_WIDTH(C_S00_AXI_DATA_WIDTH),
50          .C_S_AXI_ADDR_WIDTH(C_S00_AXI_ADDR_WIDTH)
51      ) fnd_v1_0_S00_AXI_inst (
52          .segout(segout),
53          .segcom(segcom),
54          .S_AXI_ACLK(s00_axi_aclk),
55          .S_AXI_ARESETN(s00_axi_aresetn),
```


HUINS
Human Intelligent System

변경 내역 반영



IP 수정을 완료한 뒤, 변경 내역을 반영한다.

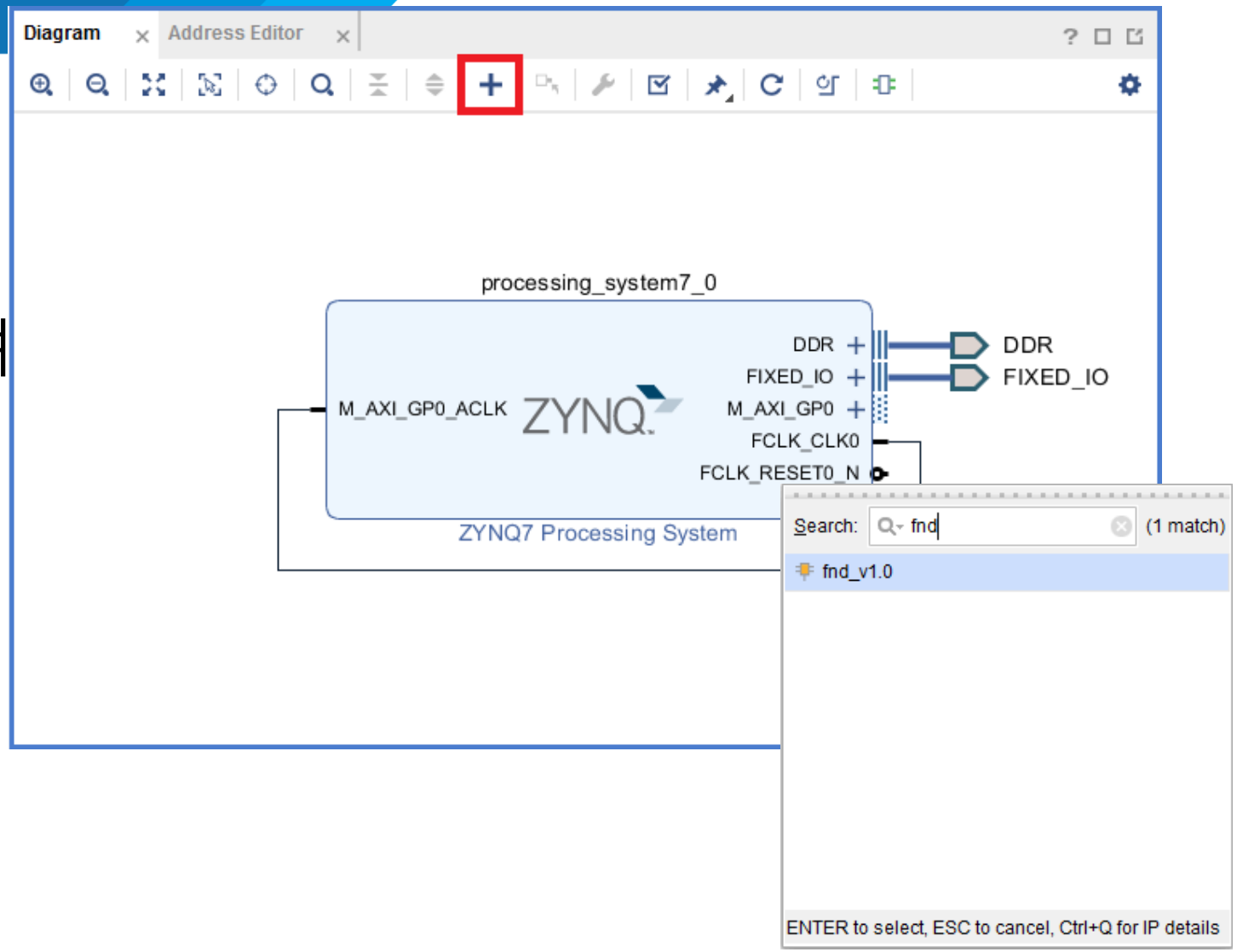
HUINS
Human Intelligent System

7-Segment IP

IP Packer로 생성한 IP

Block Design에서 검색하여 추가 및 연결 가능

필요시 IP Packer로 IP 수정



HUINS
Human Intelligent System

A. AXI-4 Protocol IP 생성(DIP SW)

DIP_SW

외부 스위치 8bit 입력

PS에서 AXI Read Transacation으로
Data Read

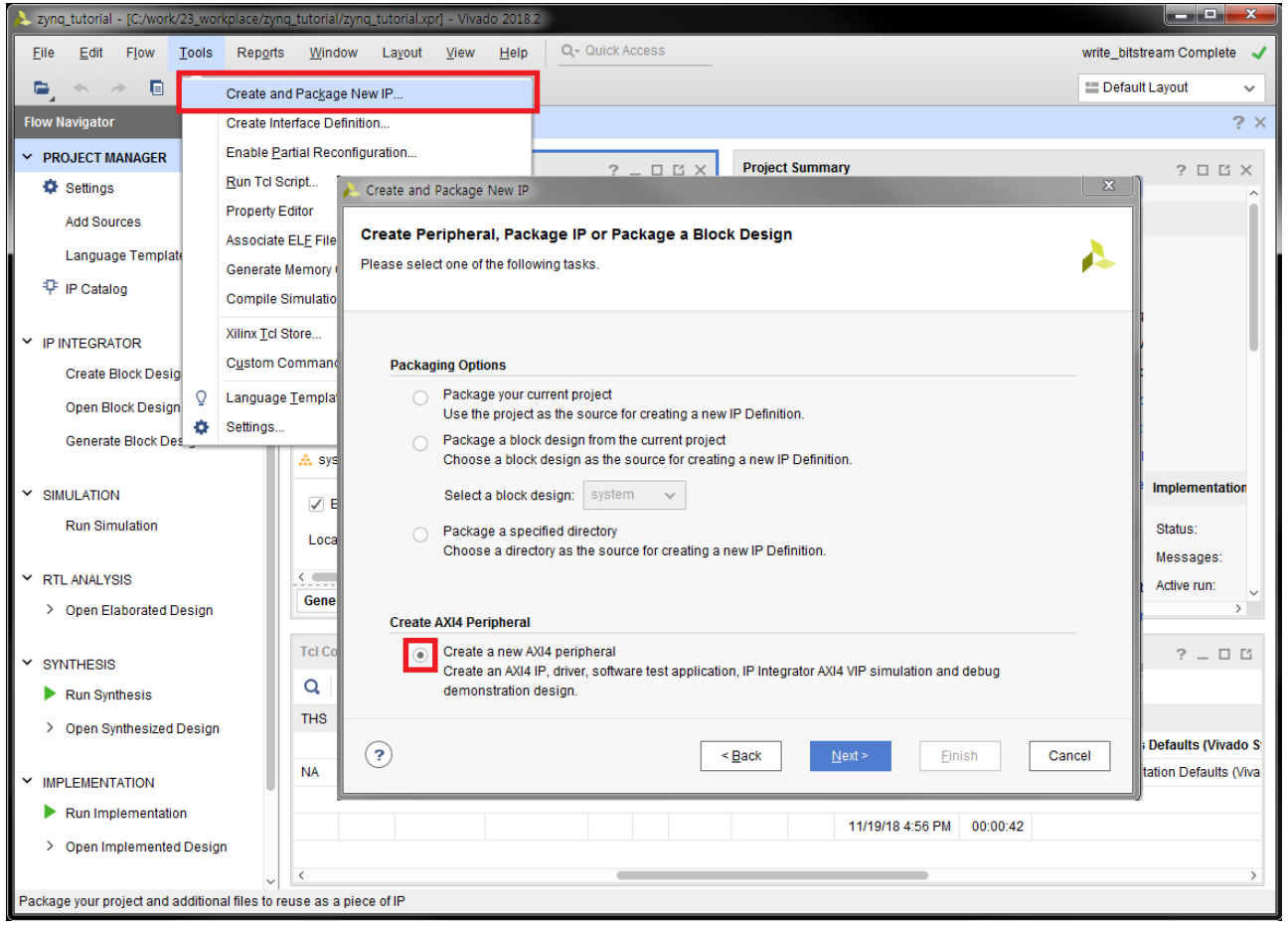
HUINS
Human Intelligent System

AXI-4 Protocol IP 생성(DIP SW)

Create and
Package New IP 실행

Create a new
AXI4 peripheral
→AXI4 Peripheral IP 생성

AXI4 프로토콜 스켈레톤 IP

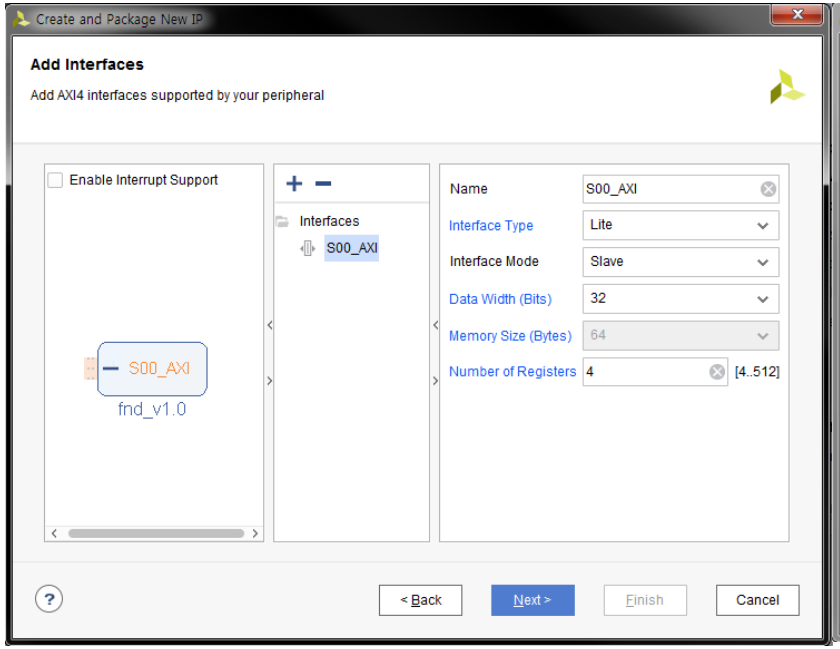


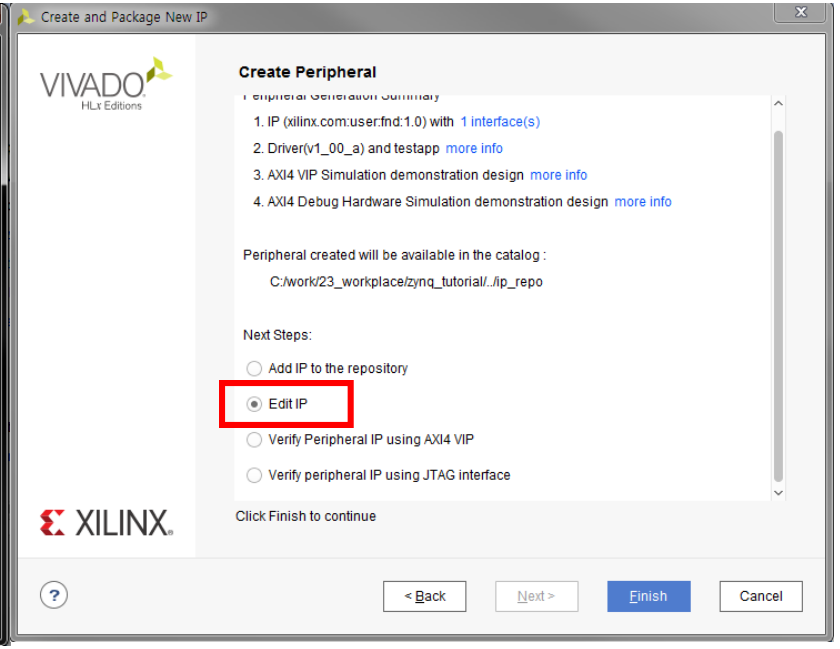
HUINS
Human Intelligent System

AXI-4 Protocol IP 생성(DIP SW)

인터페이스 설정
- Type: AXI4-Lite
- Mode: Slave

AXI 인터페이스 Skeleton IP 생성됨
- Edit-IP 실행→수정





HUINS
Human Intelligent System

AXI 인터페이스 모듈

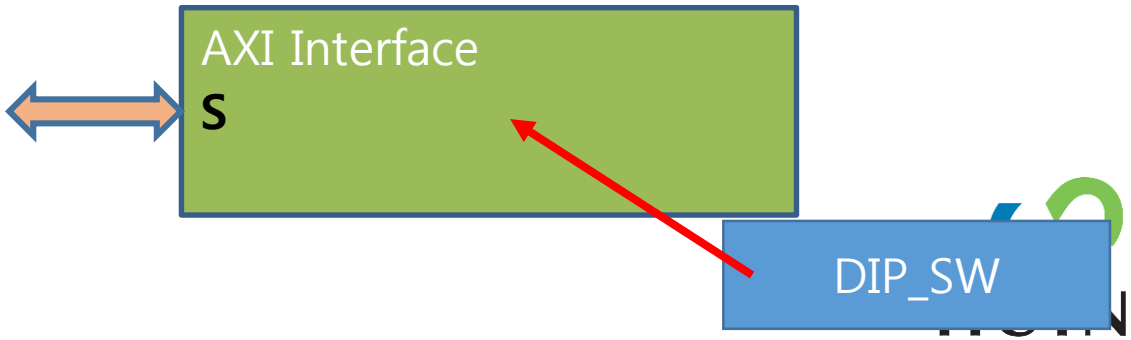
IP Packer 프로젝트 생성됨

AXI4-Slave Skeleton IP 생성됨

u_sonic_v_1_0_S00_AXI.v

- AXI 인터페이스 모듈
- read/write 위한 hand shake
- read enable시 address에 해당하는 register의 data를 출력

```
373     assign slv_reg_rden = axi_arready & S_AXI_ARVALID & ~axi_rvalid;
374     always @(*)
375     begin
376         // Address decoding for reading registers
377         case ( axi_araddr[ADDR_LSB+OPT_MEM_ADDR_BITS:ADDR_LSB] )
378           2'h0   : reg_data_out <= slv_reg0;
379           2'h1   : reg_data_out <= slv_reg1;
380           2'h2   : reg_data_out <= slv_reg2;
381           2'h3   : reg_data_out <= slv_reg3;
382           default : reg_data_out <= 0;
383         endcase
384     end
393         if (slv_reg_rden)
394           begin
395             axi_rdata <= reg_data_out;     // register read data
396           end
```



AXI 인터페이스 모듈 수정

AXI 인터페이스 모듈
u_sonic_v1_0_S00_AXI.v

외부 DIP SW 입력 선언

0번 Reading Register에 연결

```
17        // Users to add ports here
18        input wire [7:0] DIP_SW,
19        // User ports ends
```

```
369   always @(*)
370   begin
371         // Address decoding for reading registers
372         case ( axi_araddr[ADDR_LSB+OPT_MEM_ADDR_BITS:ADDR_LSB] )
373           2'h0   : reg_data_out <= DIP_SW;
374           2'h1   : reg_data_out <= slv_reg1;
375           2'h2   : reg_data_out <= slv_reg2;
376           2'h3   : reg_data_out <= slv_reg3;
377           default : reg_data_out <= 0;
378         endcase
379   end
380
```


HUINS
Human Intelligent System

AXI 탑 모듈 수정

```
17        // Users to add ports here
18        input wire [7:0] DIP_SW,
19        // User ports ends
```

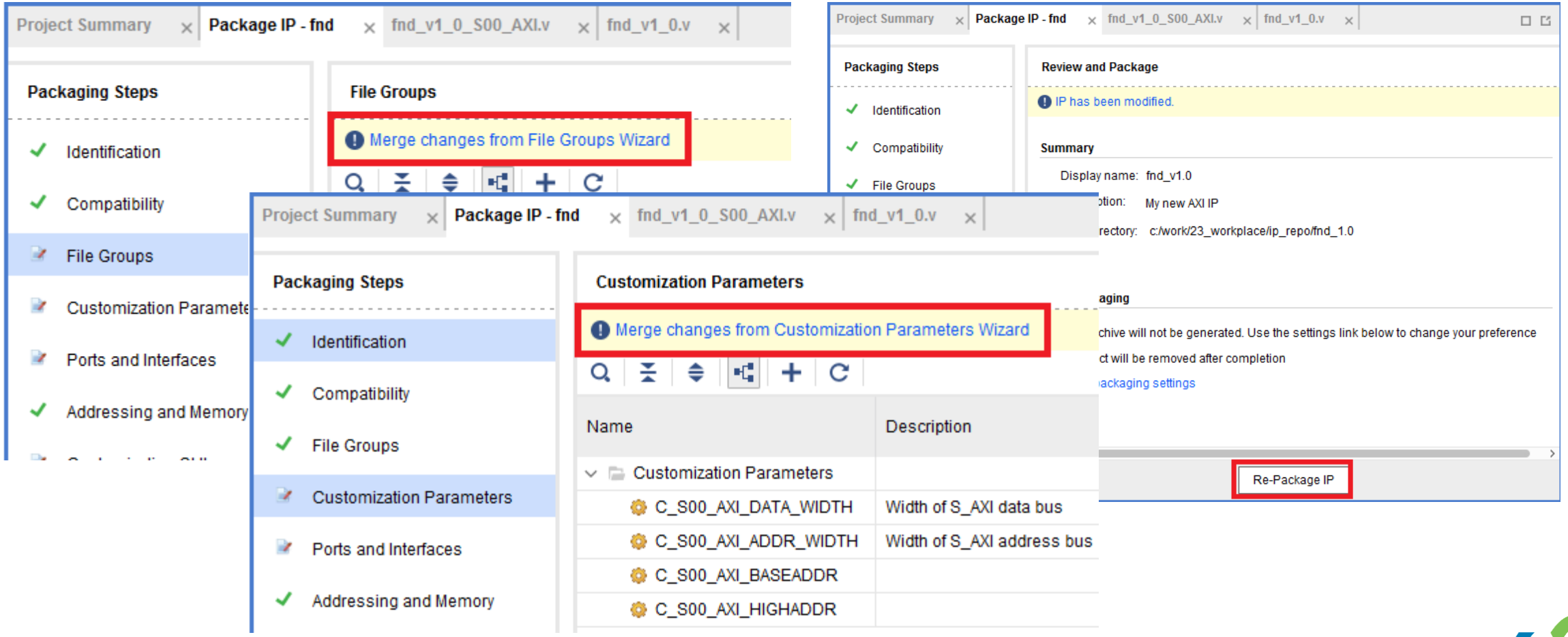
AXI 탑 모듈 u_sonic_v1_0.v

인터페이스에서 선언한
포트 선언 및 외부 연결

```
47    dip_sw_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    ) dip_sw_v1_0_S00_AXI_inst (
51        .DIP_SW(DIP_SW),
52        .S_AXI_ACLK(s00_axi_aclk),
53        .S_AXI_ARESETN(s00_axi_aresetn),
54        .S_AXI_AWADDR(s00_axi_awaddr),
55        .S_AXI_AWPROT(s00_axi_awprot),
```


HUINS
Human Intelligent System

변경 내역 반영

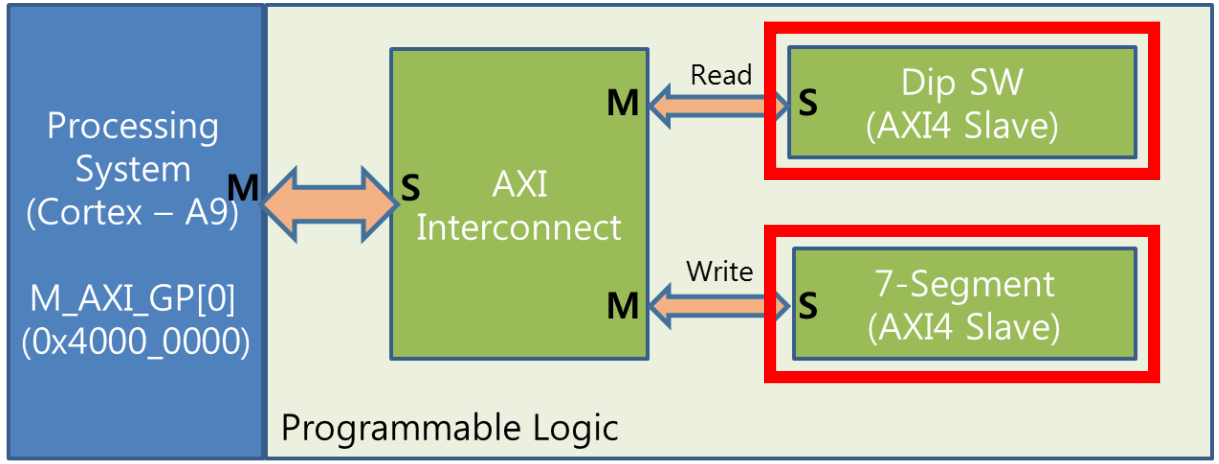


IP 수정을 완료한 뒤, 변경 내역을 반영한다.

HUINS
Human Intelligent System

2. S/W & H/W Co-Design Tutorial

A. *AXI-4 Protocol IP 생성*
   - *7-Segment → Write Transaction*
   - *U-Sonic → Read Transaction*



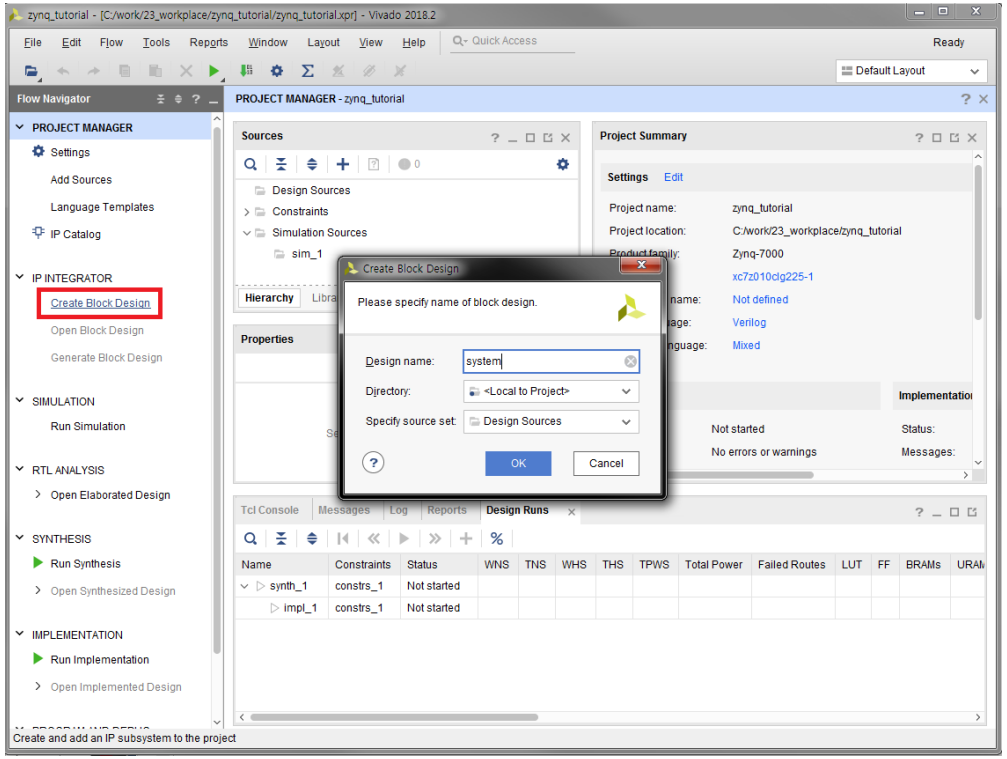
B. PS 설정(MIO/AXI Interface/DDR Controller)

C. PS – PL 연결(Block Design)

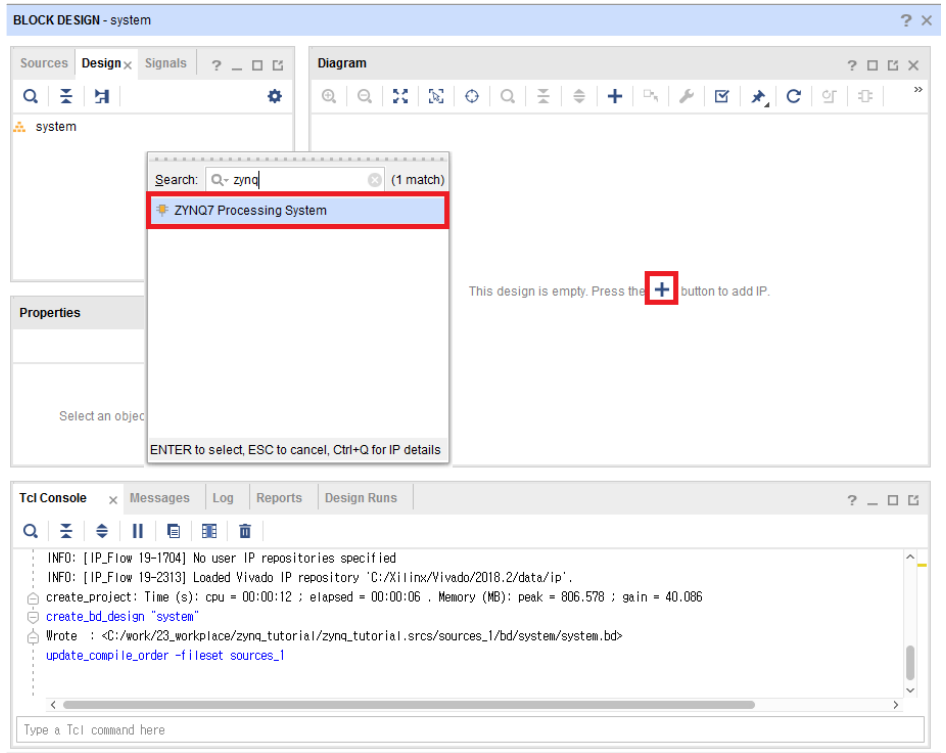
D. PS Application → AXI-4 Peripheral IP 제어(Read/Write)

HUINS
Human Intelligent System

B. Processing System 설정



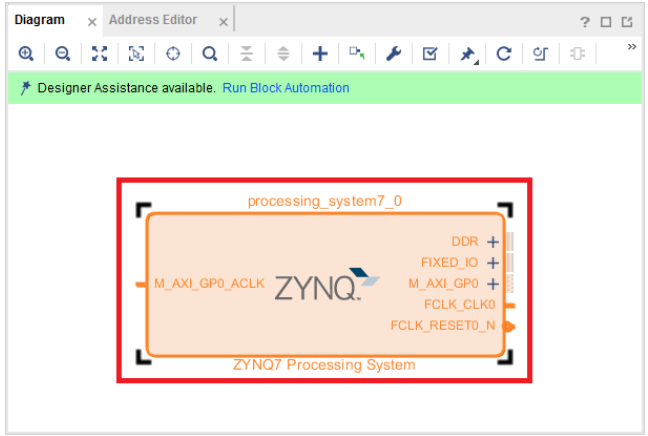
블록 디자인 생성



Zynq PS IP 추가

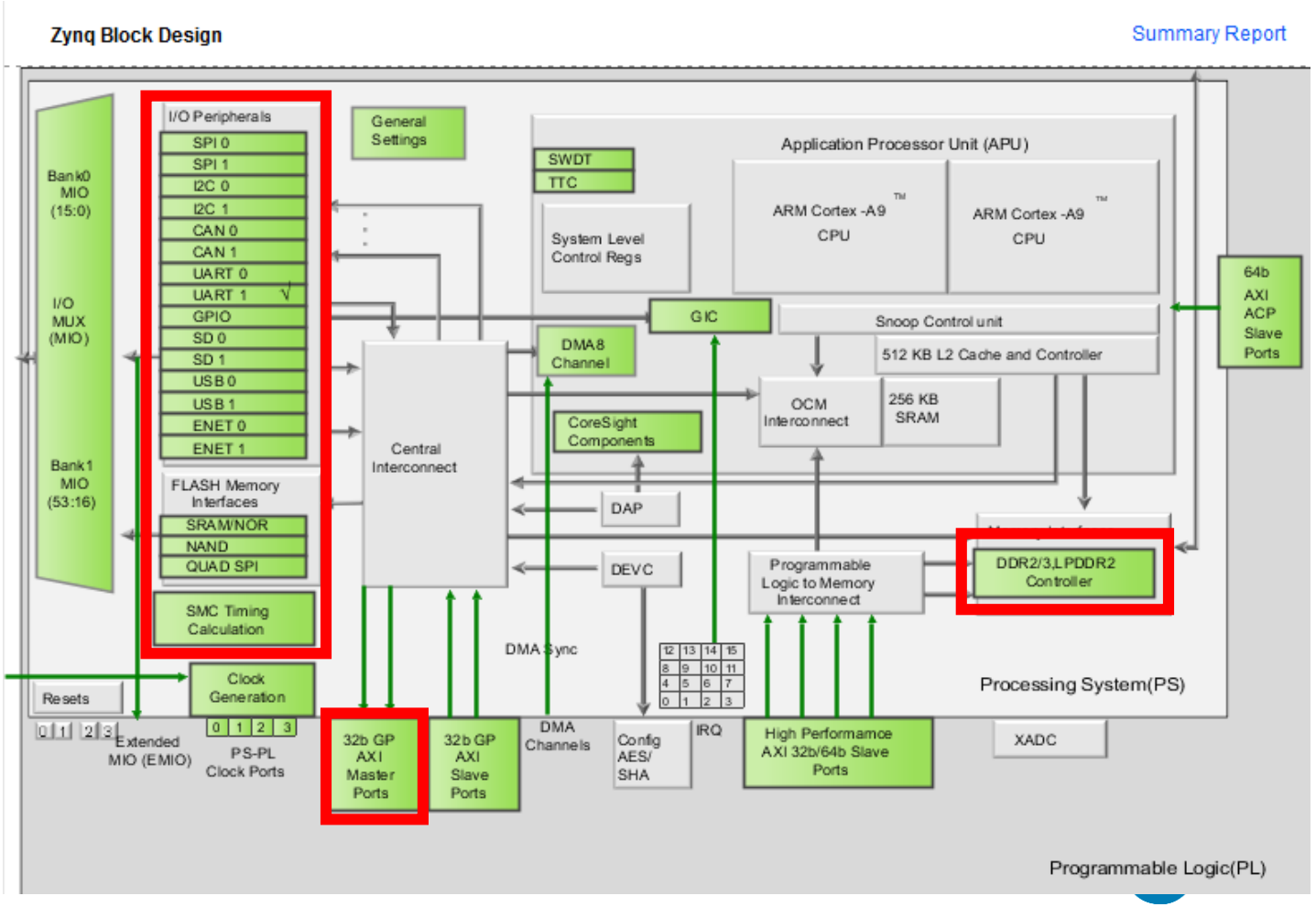
HUINS
Human Intelligent System

Processing System 설정



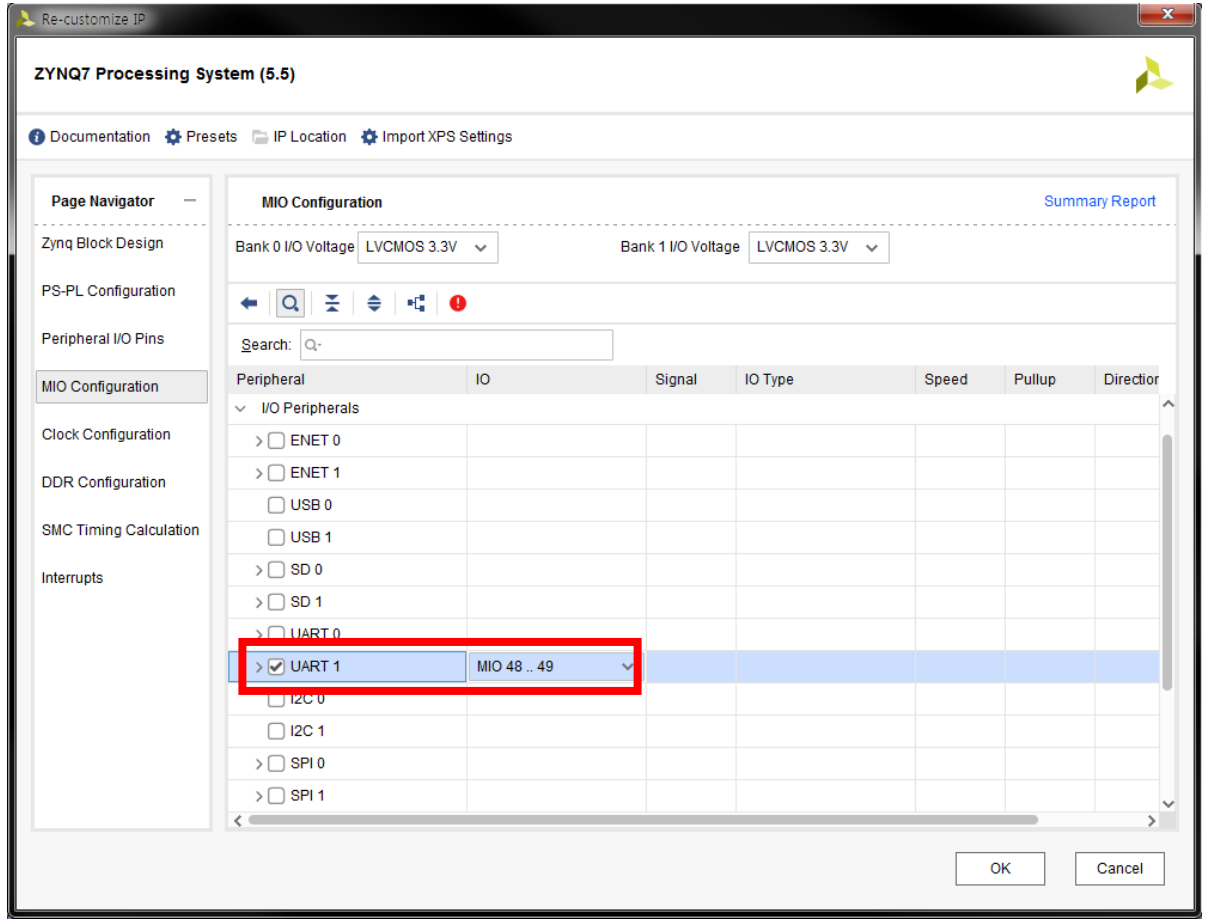
Zynq PS 설정

- I/O Peripheral → UART
- Memory Controller
- 32b GP AXI Master



HUINS
Human Intelligent System

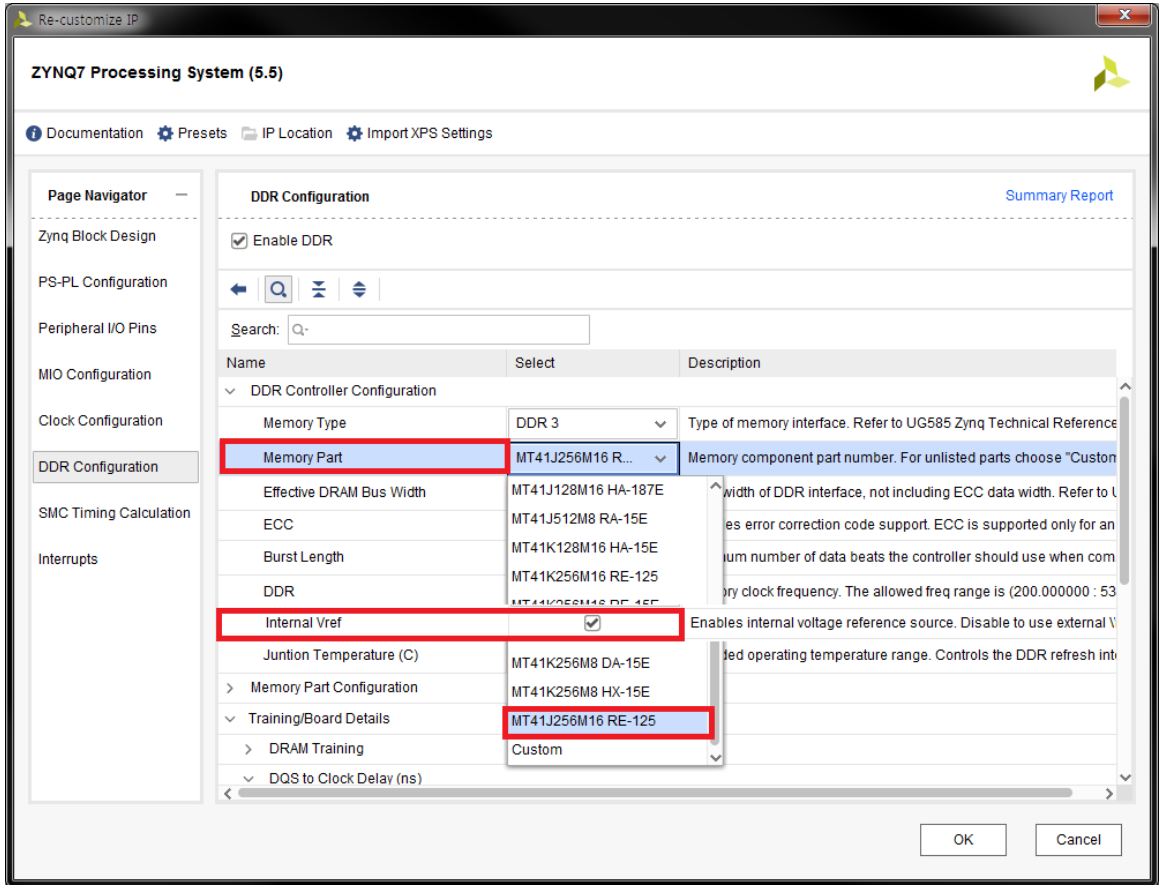
B. Processing System 설정

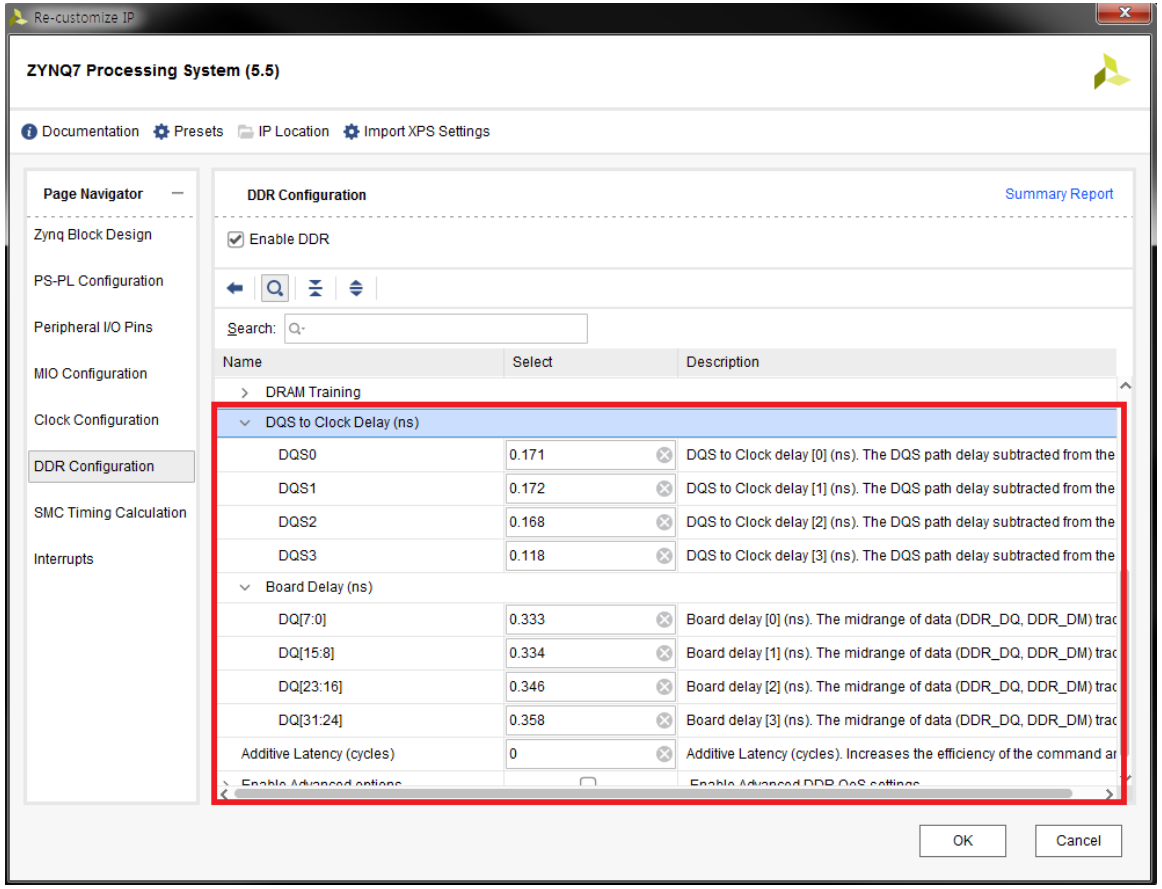


UART1을 Enable 하며, MIO 48-49번 핀 할당

HUINS
Human Intelligent System

B. Processing System 설정





DRAM Controller의 값을 변경한다
(다음 페이지 표 참조)

HUINS
Human Intelligent System

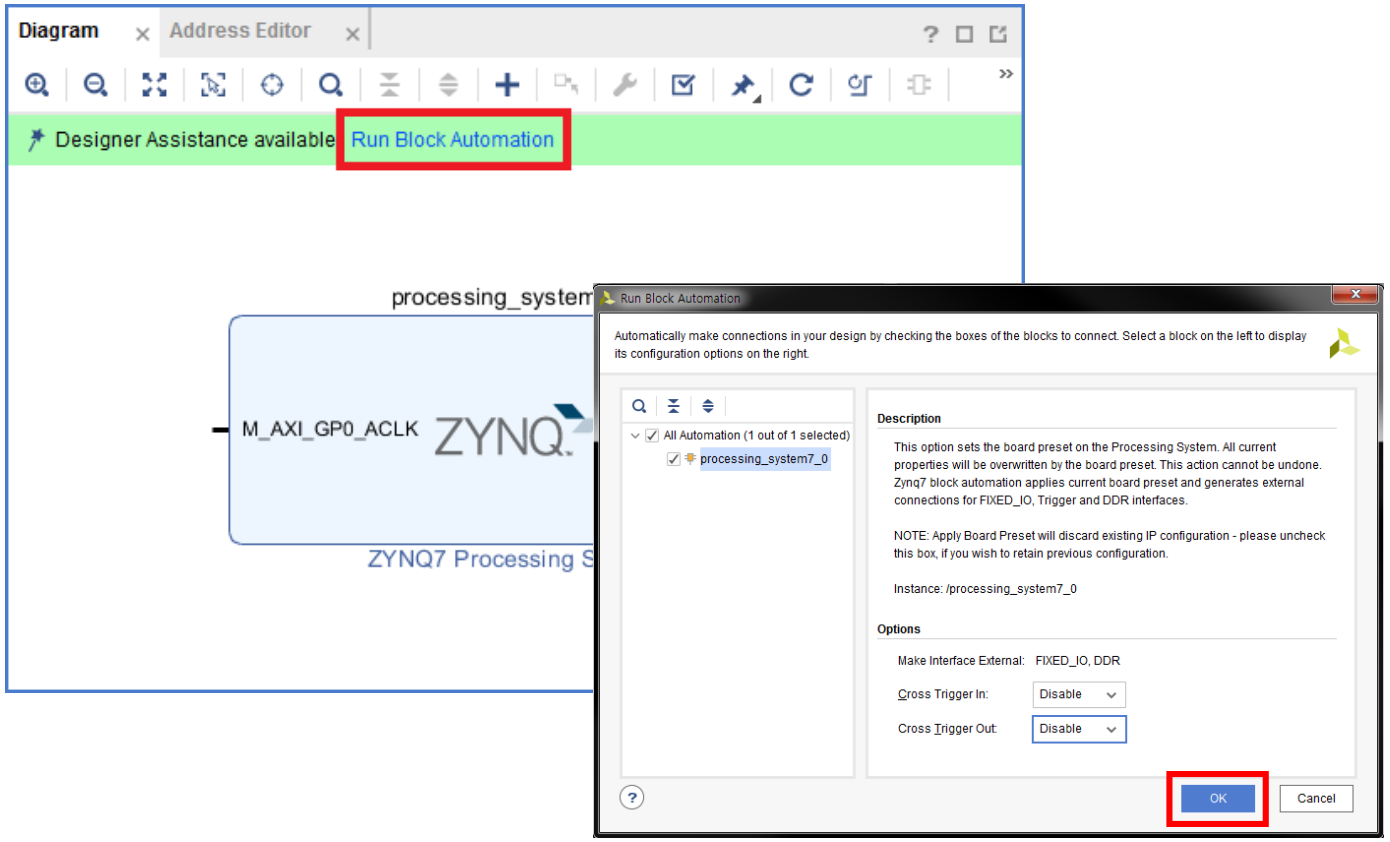
B. Processing System 설정

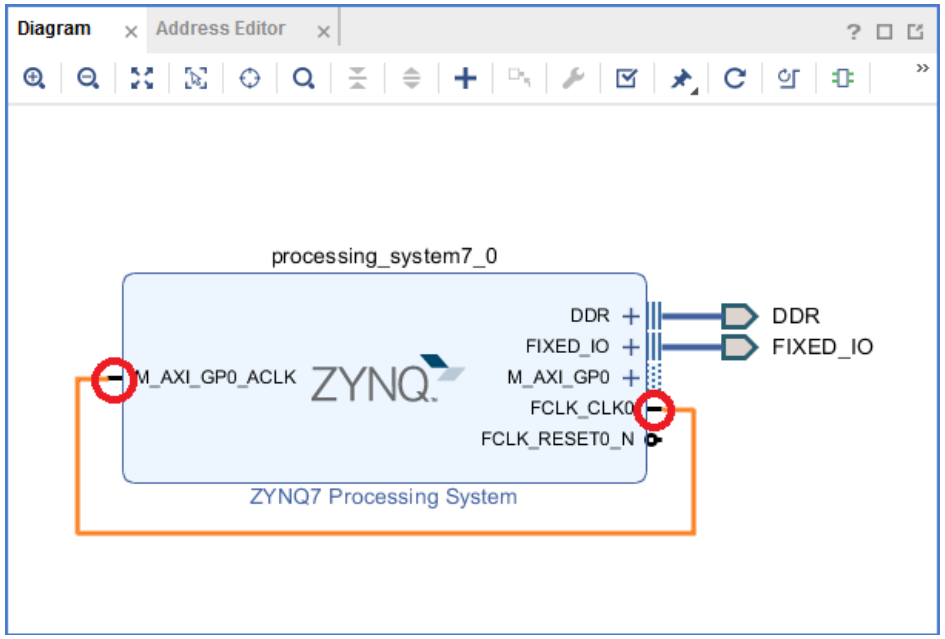
[표] Memory 설정 내용

항목		내용	항목		내용
Memory Type		DDR3	DQS to Clock Delay (ns)	DQS0	0.171
Memory Part		MT41J256M 16 RE-125		DQS1	0.172
Internal Vref		Check		DQS2	0.168
DRAM Training	Write leveling	Check		DQS3	0.118
	Read gate	Check	Board Delay (ns)	DQ[7:0]	0.333
	Read data eye	Check		DQ[15:8]	0.334
				DQ[23:16]	0.346
				DQ[31:24]	0.358

HUINS
Human Intelligent System

Processing System 설정





Run Block Automation → FIXED_IO, DDR 연결

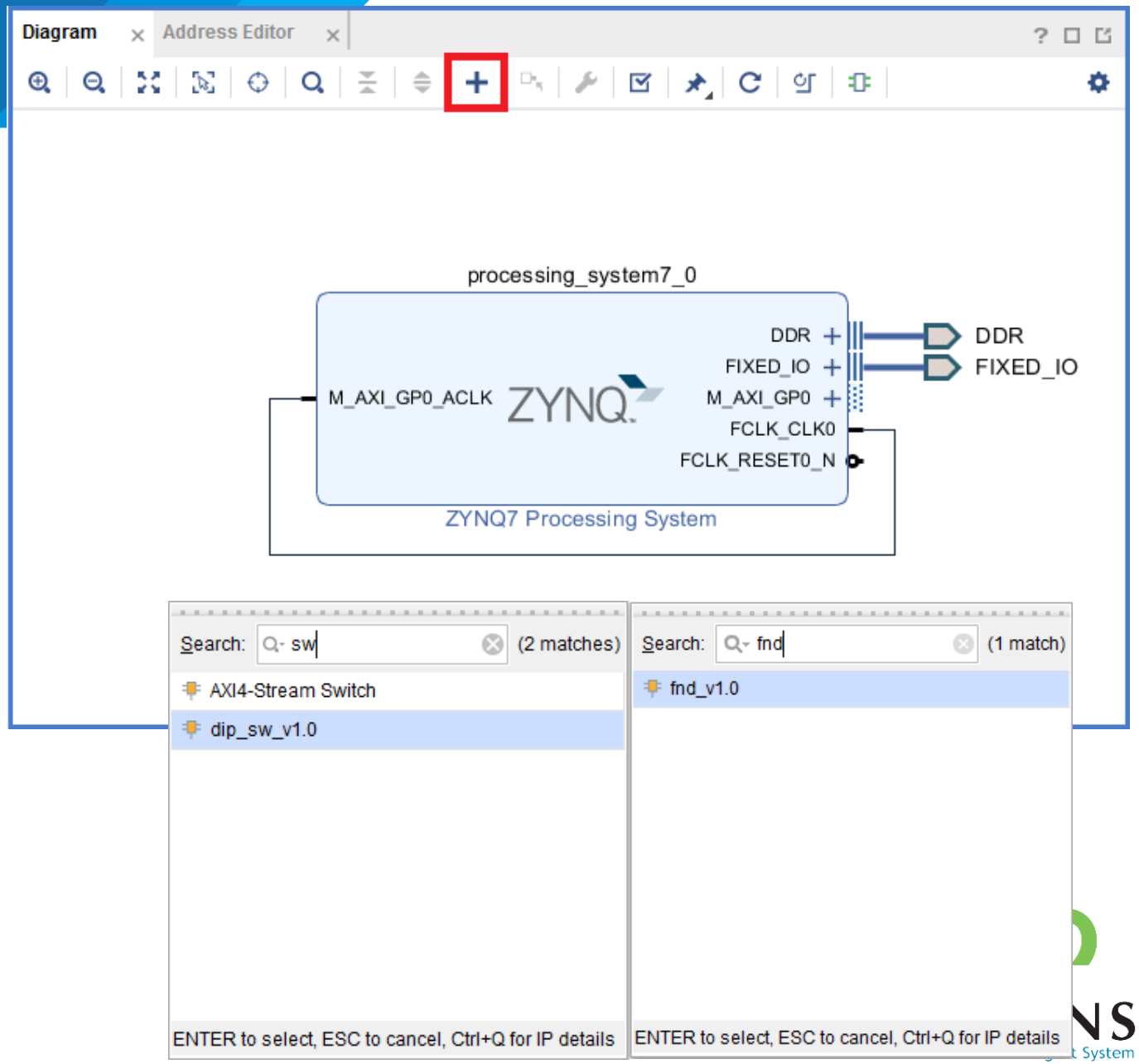
ACLK(AXI CLK)과 FCLK(Fabric CLK, PS→PL)을 연결

HUINS
Human Intelligent System

시스템 구성

시스템 블록디자인에서
생성한 AXI IP 추가

- 7-Segment
- DIP_SW



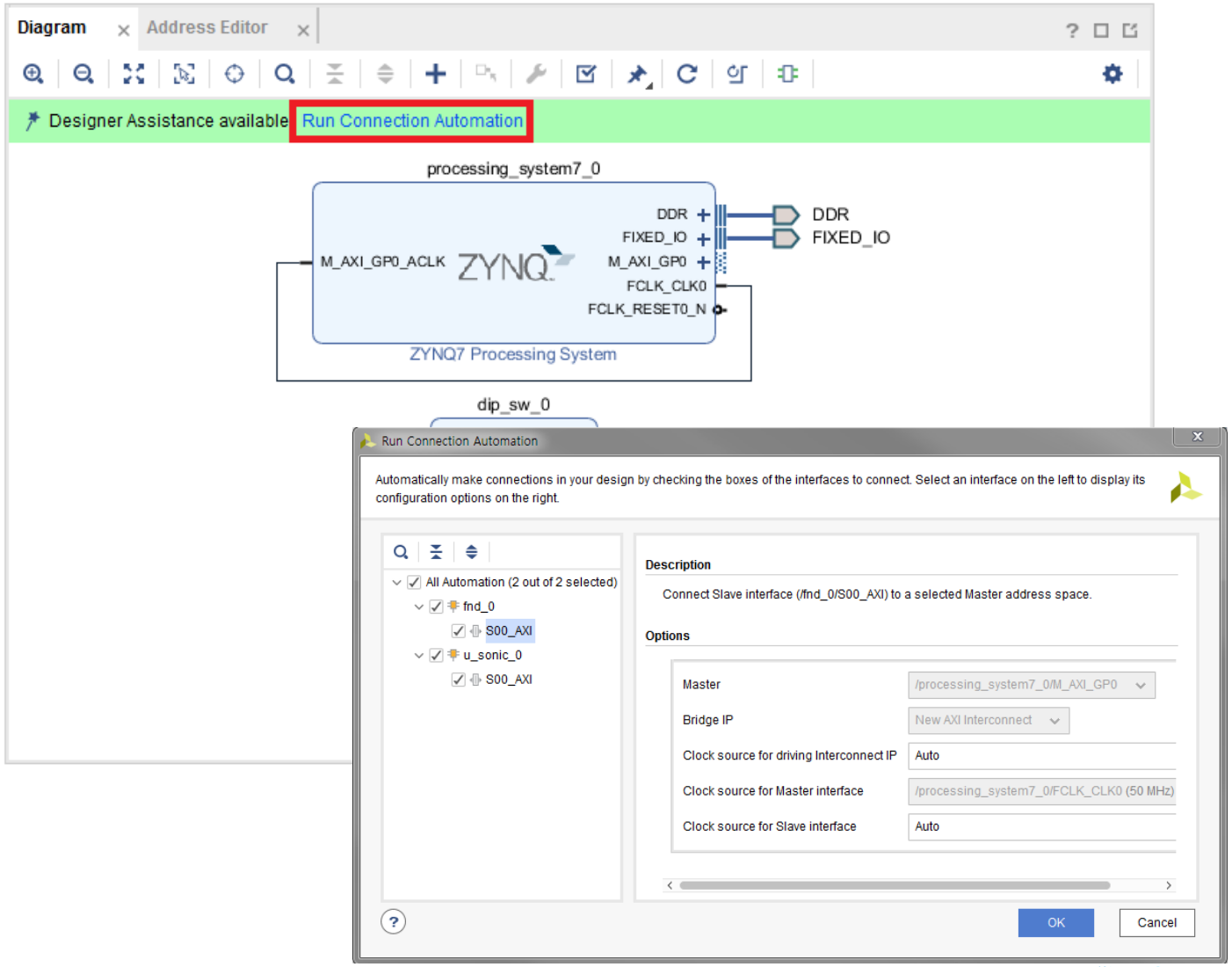
C. PS – PL 연결

Run Connection Automation

연결할 AXI IP 선택 연결

시스템 리셋 IP,

AXI Interconnect IP 생성됨



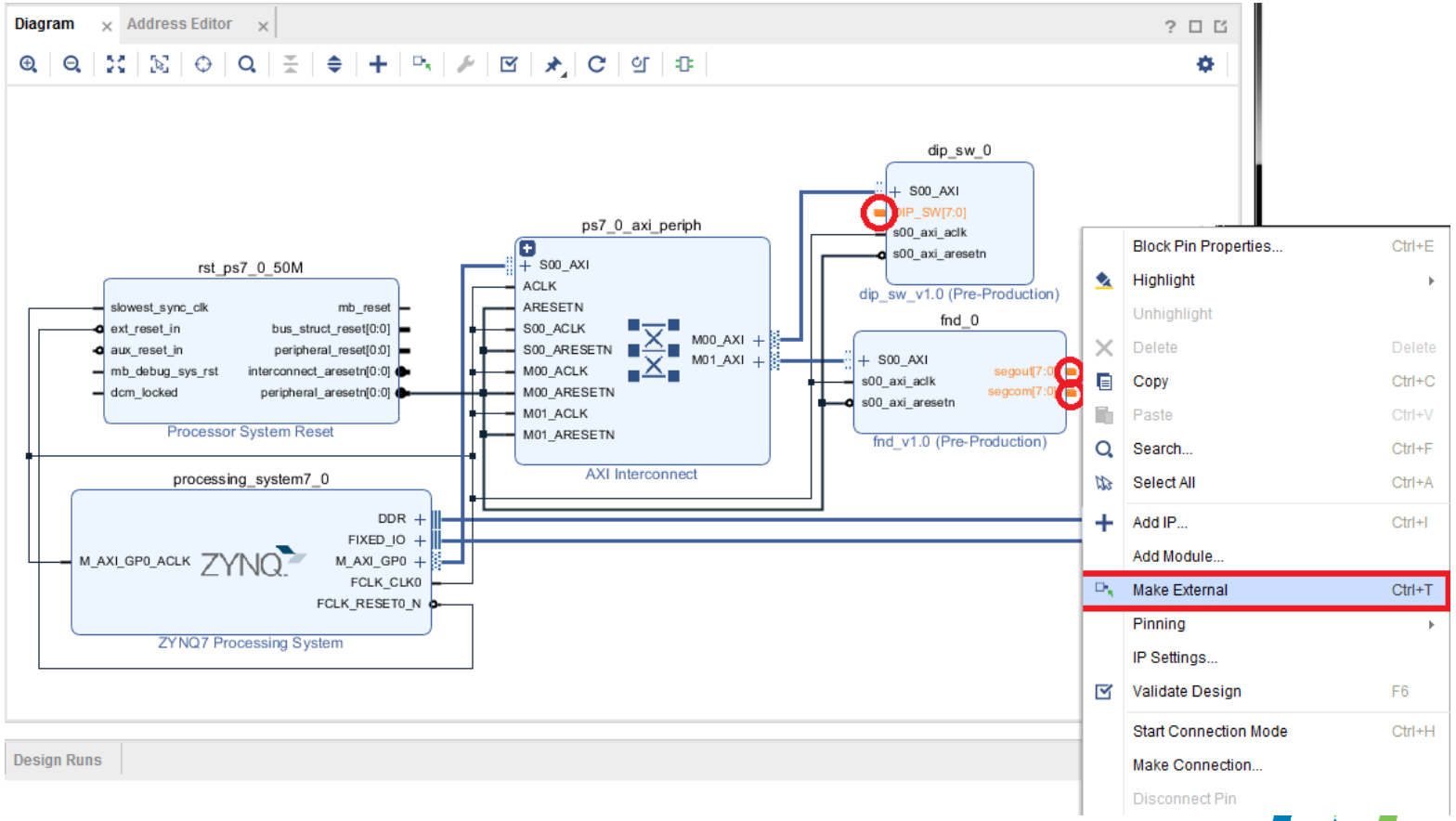
IP 포트 외부 연결

IP의 외부 포트 우클릭

Make External 실행

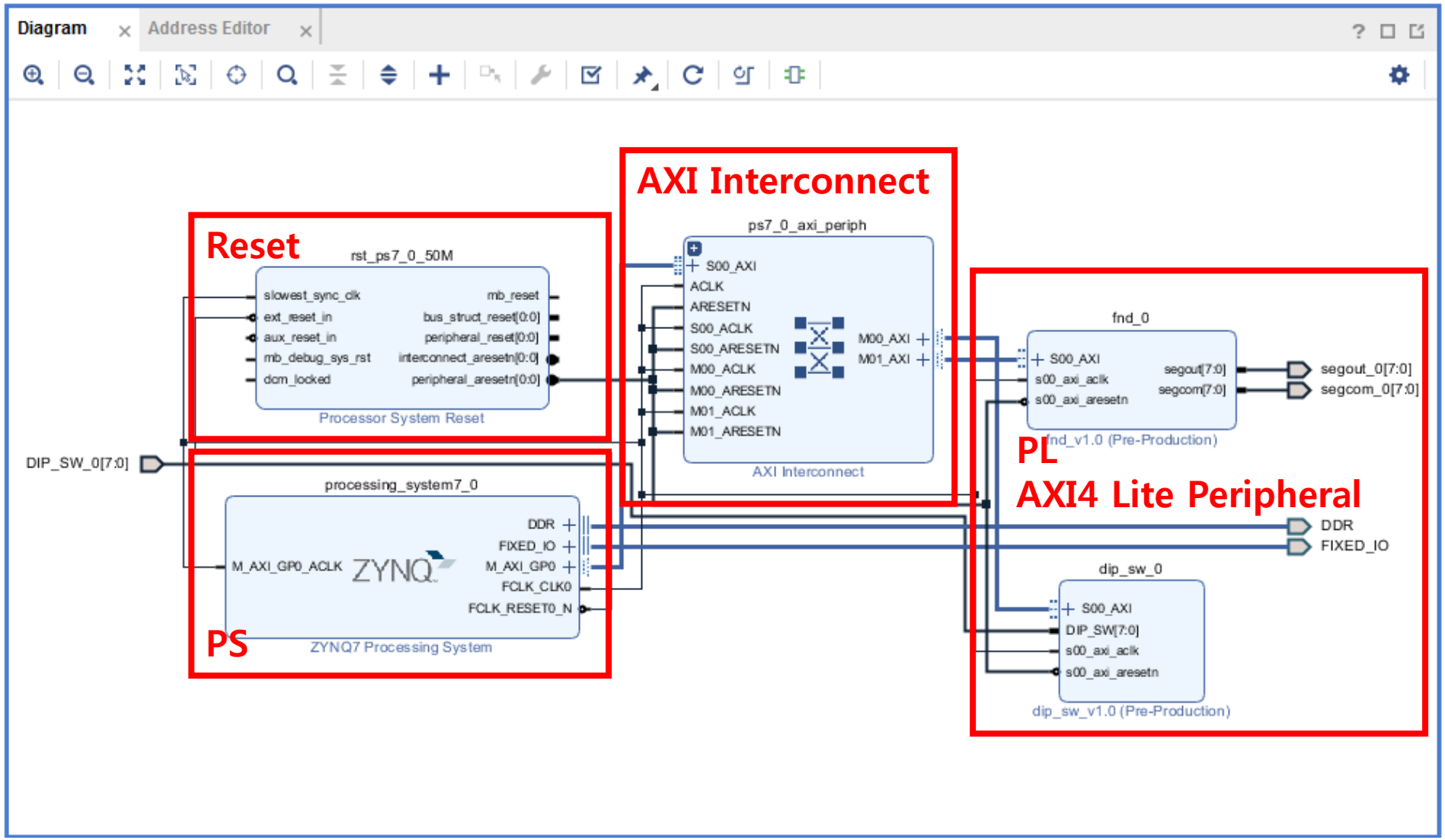
포트 외부 연결

- segout
- segcom
- DIP_SW



HUINS
Human Intelligent System

전체 시스템 구성



HUINS
Human Intelligent System

AXI IP Address

Base Address

DIP SW IP: 0x43C0_0000

- 0x43C0_0000 → DIP_SW
- 0x43C0_0004 → slv_reg1
- 0x43C0_0008 → slv_reg2
- 0x43C0_000C → slv_reg3

7-Segment IP: 0x43C1_0000

- 0x43C1_0000 → slv_reg0(DATA)
- 0x43C1_0000 → slv_reg1
- 0x43C1_0000 → slv_reg2
- 0x43C1_0000 → slv_reg3

Diagram | Address Editor

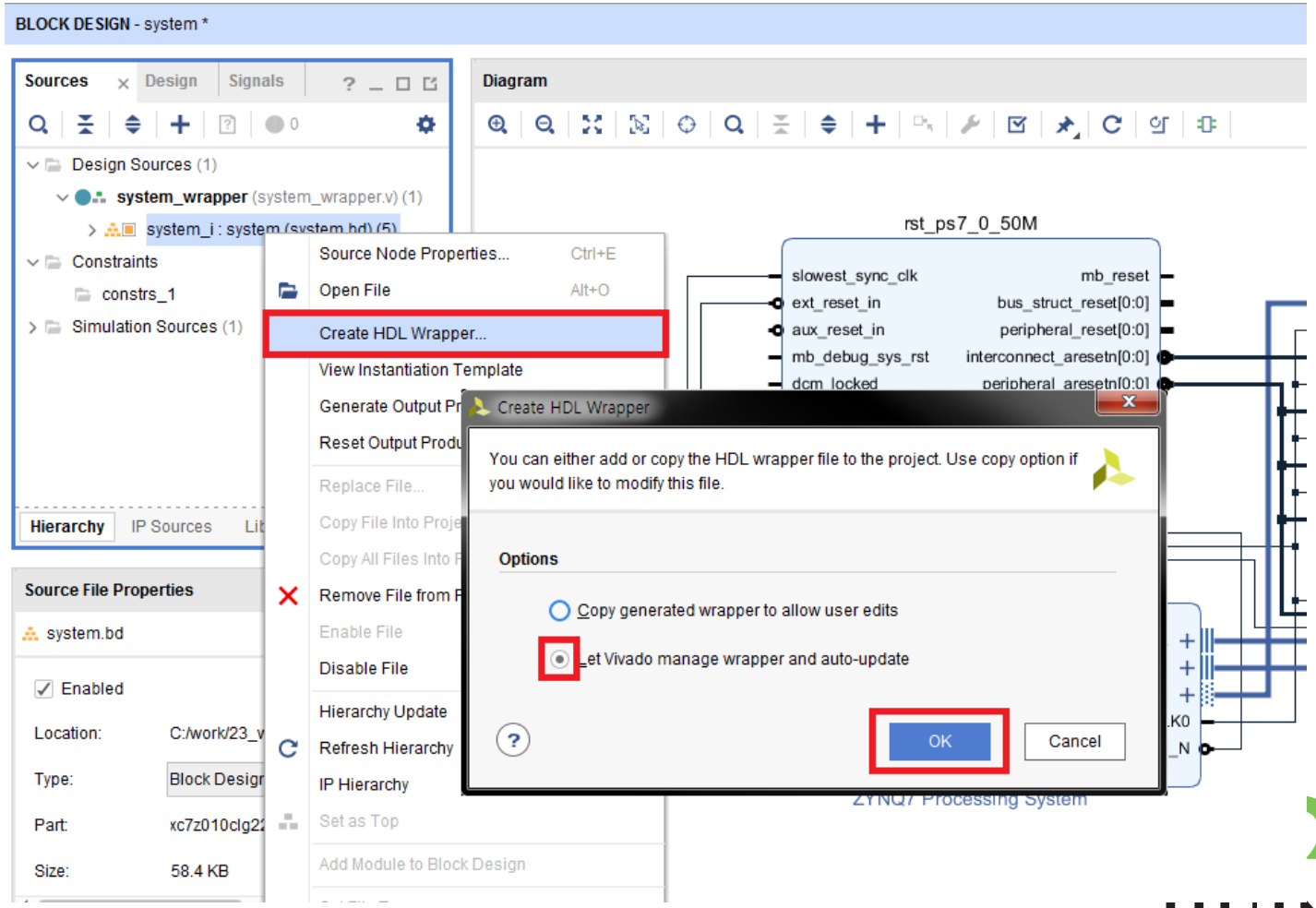
Cell	Slave Interface	Base Name	Offset Address	Range	High Address
processing_system7_0					
Data (32 address bits : 0x40000000 [1G])					
dip_sw_0	S00_AXI	S00_AXI_reg	0x43C0_0000	64K	0x43C0_FFFF
fnd_0	S00_AXI	S00_AXI_reg	0x43C1_0000	64K	0x43C1_FFFF

4000_0000 to 7FFF_FFFF	PL		PL	General Purpose Port #0 to the PL, M_AXI_GP0
8000_0000 to BFFF_FFFF	PL		PL	General Purpose Port #1 to the PL, M_AXI_GP1

HDL Wrapper 생성

HDL wrapper 생성

- 블록 디자인의
  HDL 인스턴스 생성



HUINS
Human Intelligent System

Constraints 작성

Constraints 파일(xdc) 생성

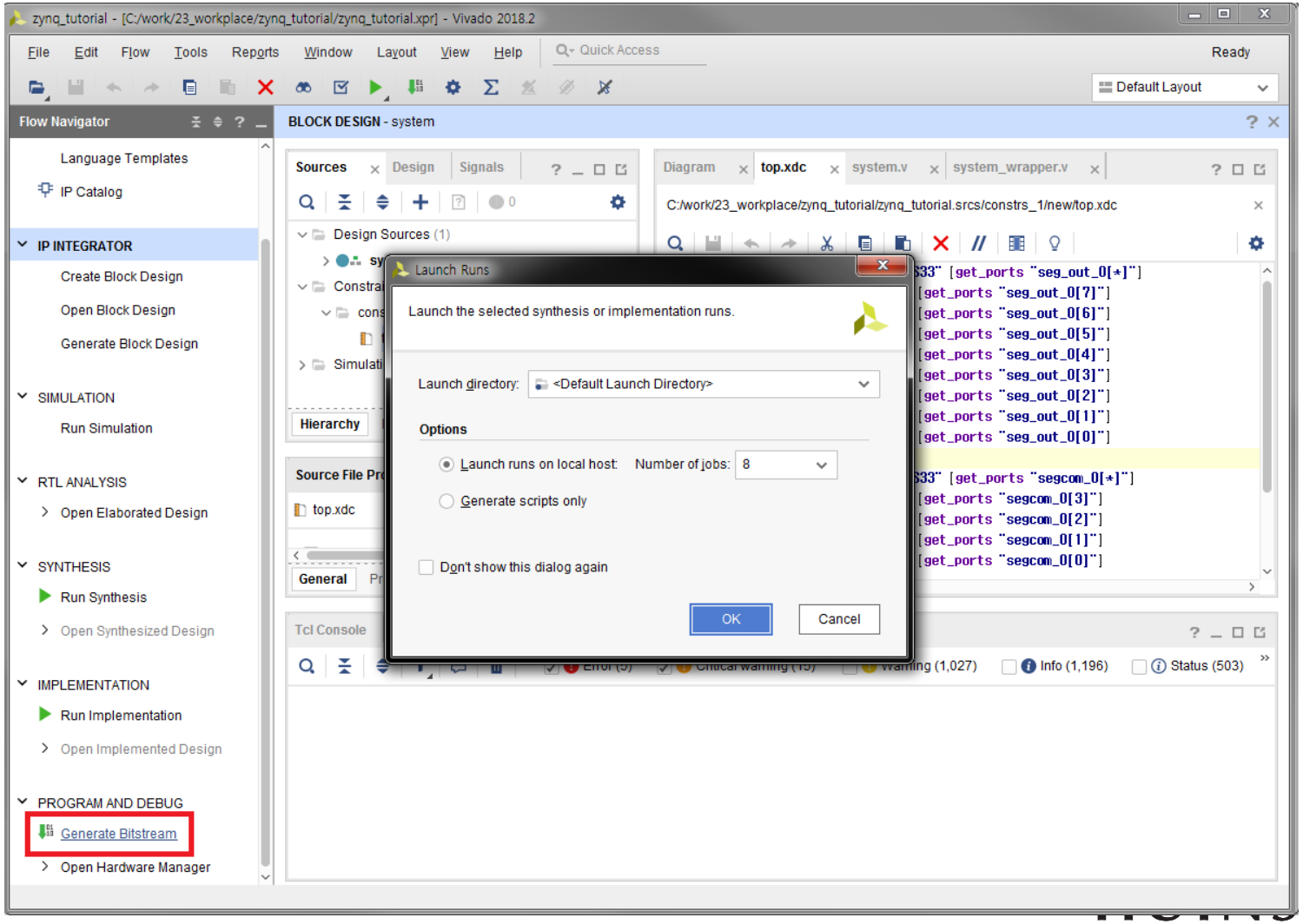
FPGA 핀과 입출력 포트 맵핑
- 7-Segment
- DIP SW

LVCMOS 3.3V 설정

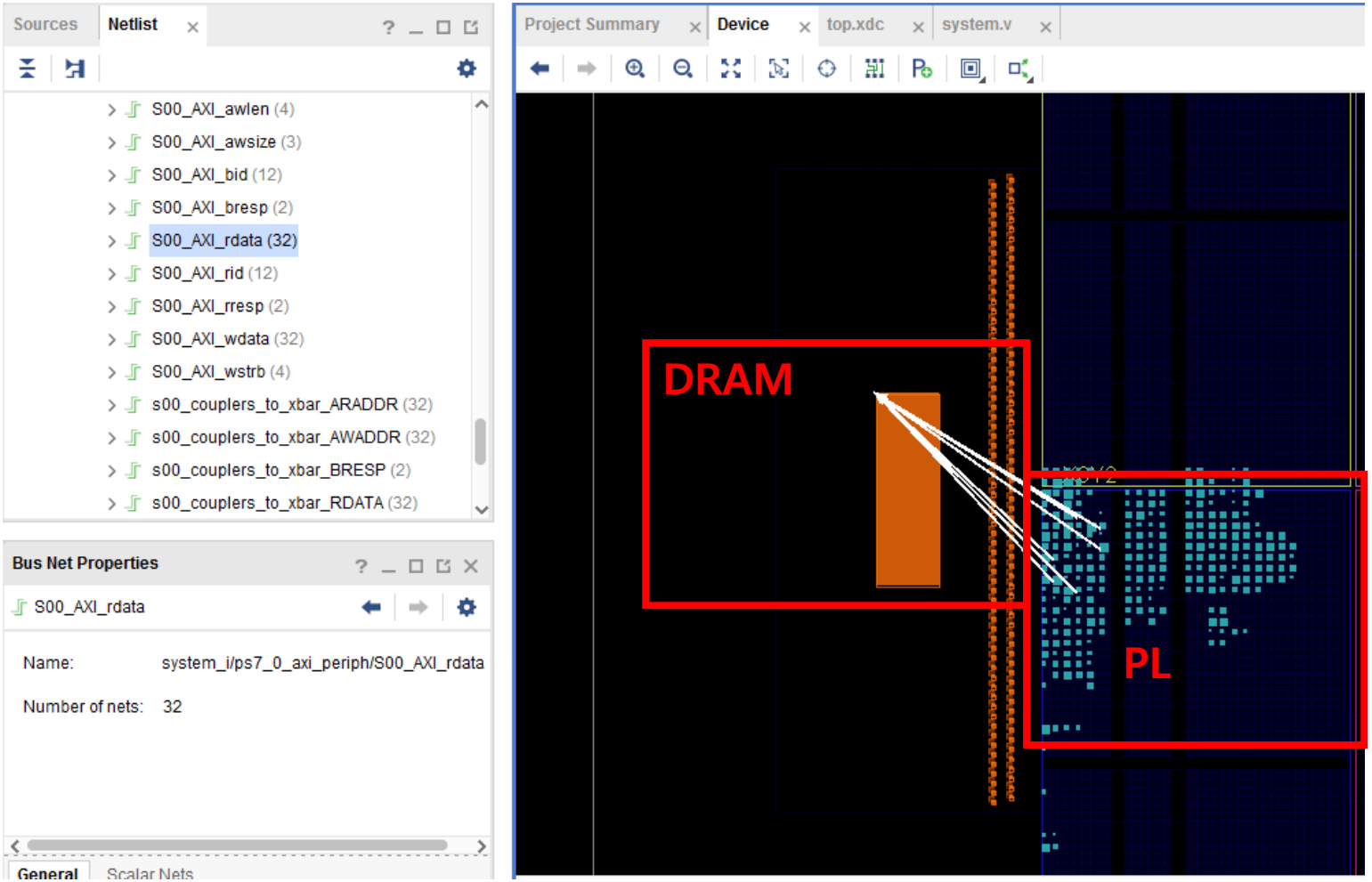
```
set_property PACKAGE_PIN "Y20" [get_ports "DIP_SW_0[7]"]
set_property PACKAGE_PIN "Y21" [get_ports "DIP_SW_0[6]"]
set_property PACKAGE_PIN "AB19" [get_ports "DIP_SW_0[5]"]
set_property PACKAGE_PIN "AB20" [get_ports "DIP_SW_0[4]"]
set_property PACKAGE_PIN "AA22" [get_ports "DIP_SW_0[3]"]
set_property PACKAGE_PIN "AB22" [get_ports "DIP_SW_0[2]"]
set_property PACKAGE_PIN "AA21" [get_ports "DIP_SW_0[1]"]
set_property PACKAGE_PIN "AB21" [get_ports "DIP_SW_0[0]"]
set_property IOSTANDARD "LVCMOS33" [get_ports "DIP_SW_0[*]"]

set_property PACKAGE_PIN "Y6" [get_ports "segout_0[7]"]
set_property PACKAGE_PIN "Y5" [get_ports "segout_0[6]"]
set_property PACKAGE_PIN "AA7" [get_ports "segout_0[5]"]
set_property PACKAGE_PIN "AA6" [get_ports "segout_0[4]"]
set_property PACKAGE_PIN "AB2" [get_ports "segout_0[3]"]
set_property PACKAGE_PIN "AB1" [get_ports "segout_0[2]"]
set_property PACKAGE_PIN "AB5" [get_ports "segout_0[1]"]
set_property PACKAGE_PIN "AB4" [get_ports "segout_0[0]"]
set_property IOSTANDARD "LVCMOS33" [get_ports "segout_0[*]"]

set_property PACKAGE_PIN "AB7" [get_ports "segcom_0[7]"]
set_property PACKAGE_PIN "AB6" [get_ports "segcom_0[6]"]
set_property PACKAGE_PIN "Y4" [get_ports "segcom_0[5]"]
set_property PACKAGE_PIN "AA4" [get_ports "segcom_0[4]"]
set_property PACKAGE_PIN "R6" [get_ports "segcom_0[3]"]
set_property PACKAGE_PIN "T6" [get_ports "segcom_0[2]"]
set_property PACKAGE_PIN "T4" [get_ports "segcom_0[1]"]
set_property PACKAGE_PIN "U4" [get_ports "segcom_0[0]"]
set_property IOSTANDARD "LVCMOS33" [get_ports "segcom_0[*]"]
```

Generate Bitstream

Generate Bitstream
- Synthesis
- Implementation
- Generate Bitstream

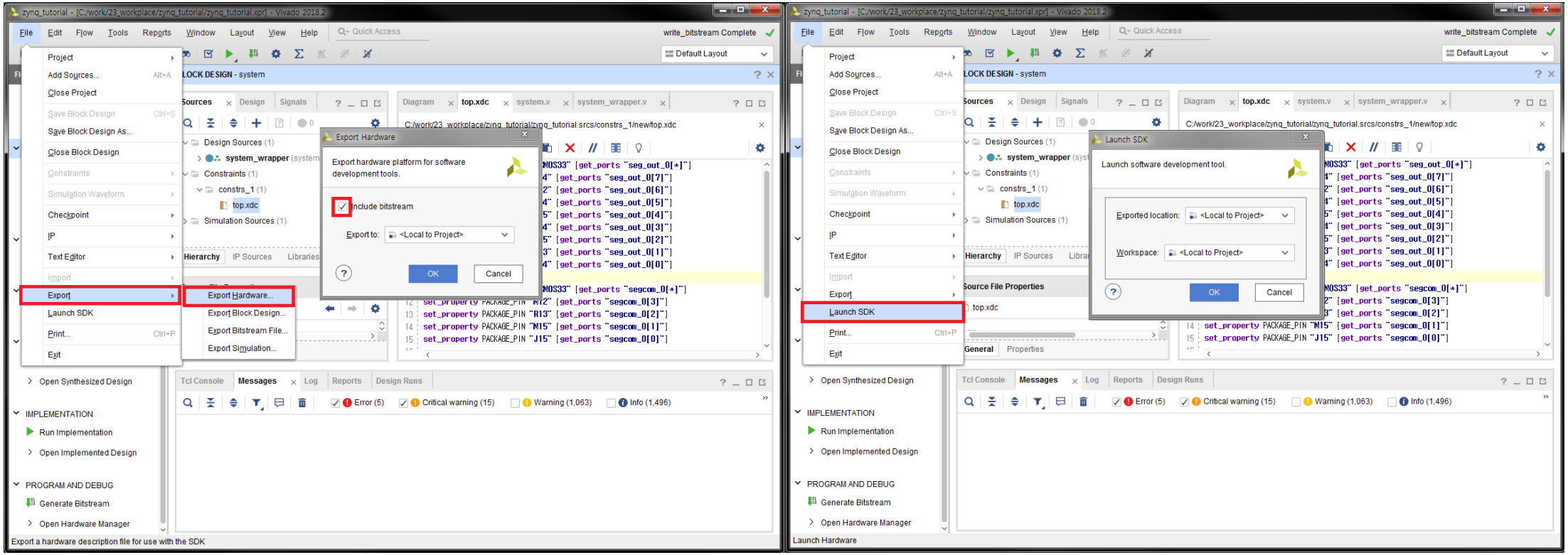


IMPLEMENTATION – Device



HUINS
Human Intelligent System

Hardware Export & Launch SDK



Export Hardware → PS설정 정보 및 bitstream 출력

출력된 하드웨어 정보 기반으로 SDK 실행

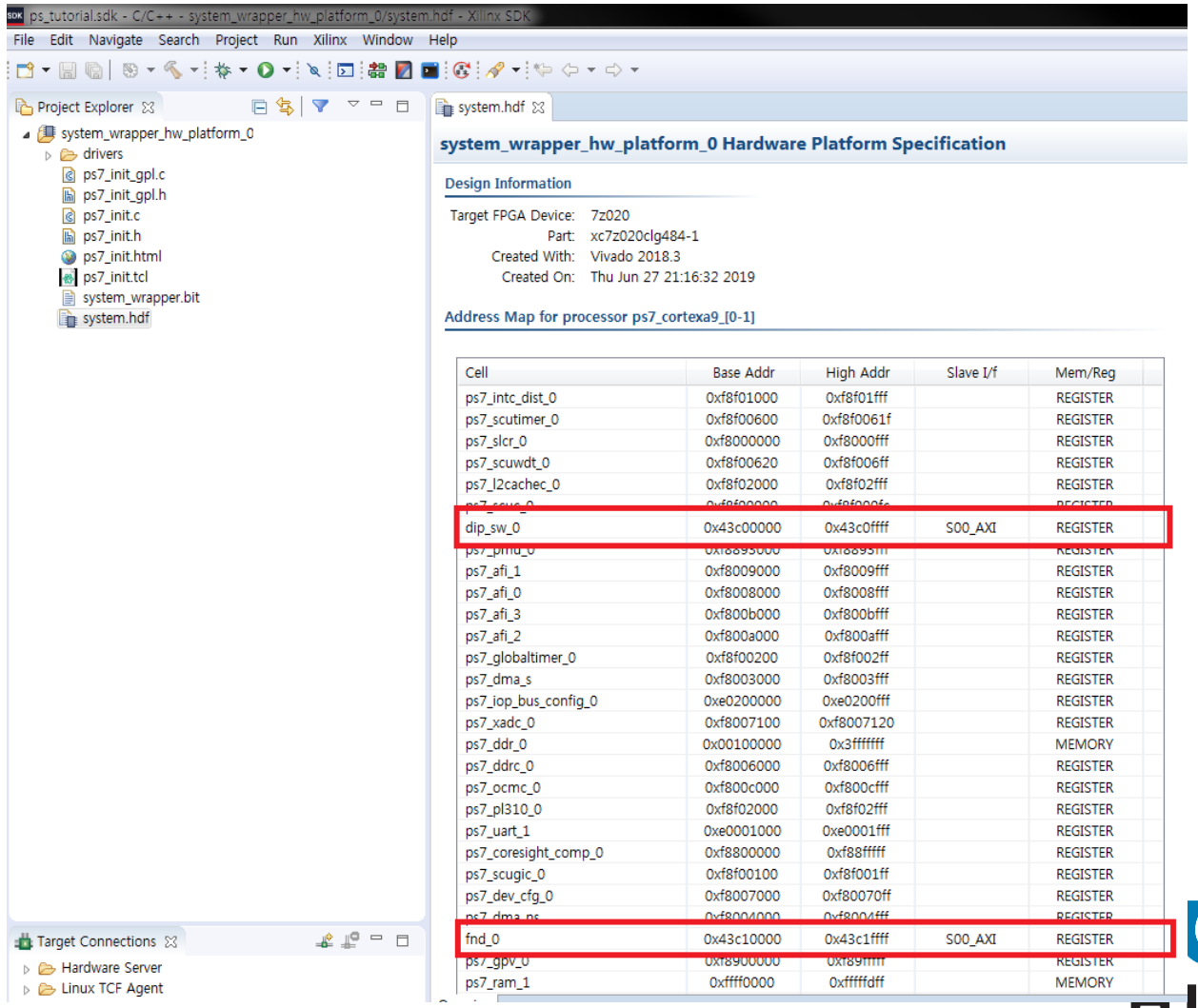
HUINS
Human Intelligent System

D. PS Application(SDK)

SDK 실행 시

HDF 파일의 내용 표시

- Part name: xc7z020clg484-1
- Tool: Vivado 2018.3
- Processor Address Map
- 디자인 내 IP Block 종류
  - AXI4 Slave 인터페이스
  - FND, DIP_SW



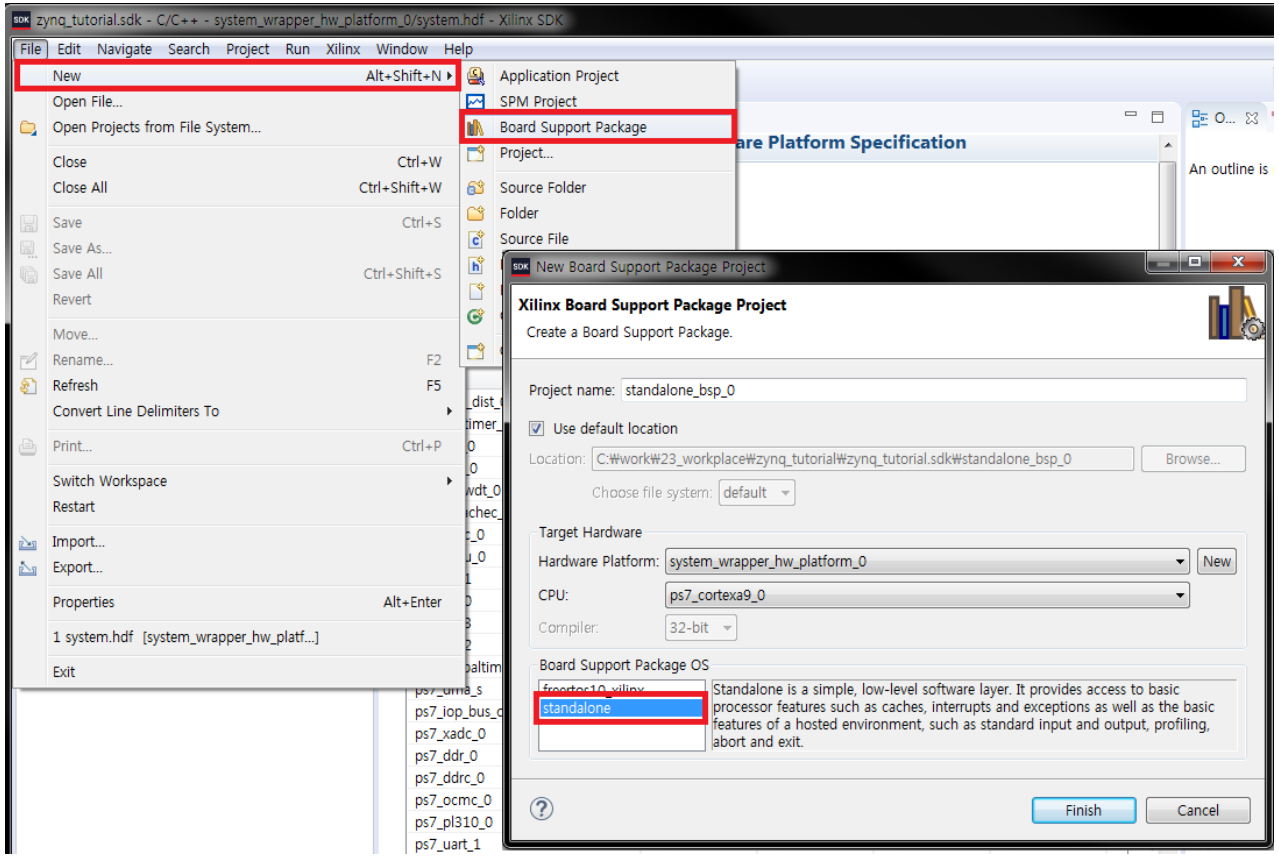
Cell	Base Addr	High Addr	Slave I/f	Mem/Reg
ps7_intc_dist_0	0xf8f01000	0xf8f01fff		REGISTER
ps7_scutimer_0	0xf8f00600	0xf8f0061f		REGISTER
ps7_slcr_0	0xf8000000	0xf8000fff		REGISTER
ps7_scuwdt_0	0xf8f00620	0xf8f006ff		REGISTER
ps7_l2cachec_0	0xf8f02000	0xf8f02fff		REGISTER
dip_sw_0	0x43c00000	0x43c0ffff	S00_AXI	REGISTER
ps7_afi_1	0xf8009000	0xf8009fff		REGISTER
ps7_afi_0	0xf8008000	0xf8008fff		REGISTER
ps7_afi_3	0xf800b000	0xf800bfff		REGISTER
ps7_afi_2	0xf800a000	0xf800afff		REGISTER
ps7_globaltimer_0	0xf8f00200	0xf8f002ff		REGISTER
ps7_dma_s	0xf8003000	0xf8003fff		REGISTER
ps7_iop_bus_config_0	0xe0200000	0xe0200fff		REGISTER
ps7_xadc_0	0xf8007100	0xf8007120		REGISTER
ps7_ddr_0	0x00100000	0x3fffffff		MEMORY
ps7_ddrc_0	0xf8006000	0xf8006fff		REGISTER
ps7_ocmc_0	0xf800c000	0xf800cfff		REGISTER
ps7_pl310_0	0xf8f02000	0xf8f02fff		REGISTER
ps7_uart_1	0xe0001000	0xe0001fff		REGISTER
ps7_coresight_comp_0	0xf8800000	0xf88fffff		REGISTER
ps7_scugic_0	0xf8f00100	0xf8f001ff		REGISTER
ps7_dev_cfg_0	0xf8007000	0xf80070ff		REGISTER
fnd_0	0x43c10000	0x43c1ffff	S00_AXI	REGISTER
ps7_ram_1	0xffff0000	0xffffdff		MEMORY

HUINS Human Intelligent System

Create Board Support Package

BSP 생성

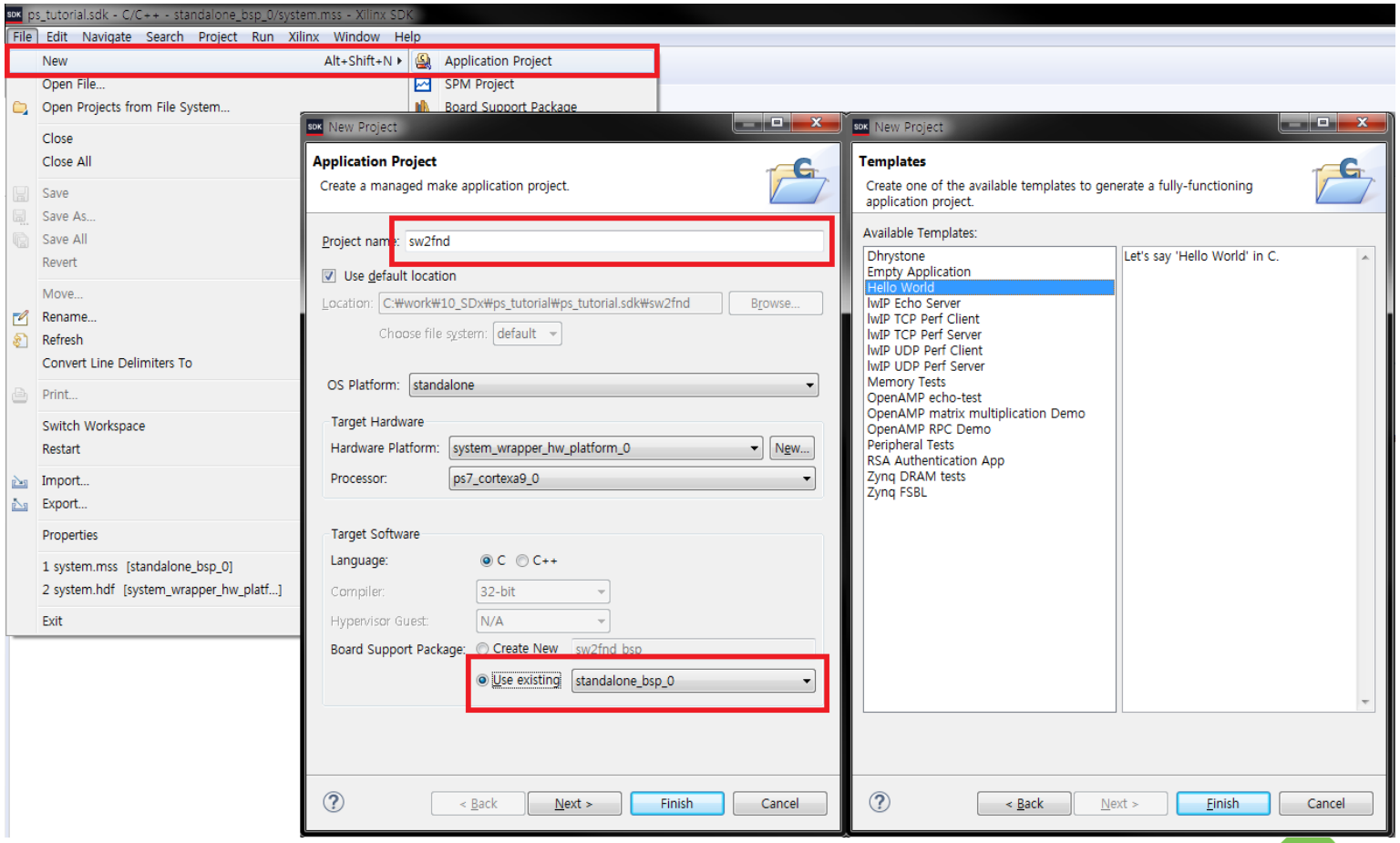
BSP - 장치 드라이버, 예외 처리, 파일 및 메모리 관리



HUINS
Human Intelligent System

Create Application

Application
프로젝트 생성
- OS: standalone
- Language: C
- BSP: standalone_bsp_0
- Template: Hello World



HUINS
Human Intelligent System

helloworld.c

```c
#include <stdio.h>
#include "platform.h"
#include "xil_printf.h"
#include "xil_io.h"
#include "xparameters.h"
#include "fnd.h"
#include "dip_sw.h"

int main(){
        u8 sw_data = 0;
        u32 seg_data = 0;

        while(1){
                sw_data = (DIP_SW_mReadReg(XPAR_DIP_SW_0_S00_AXI_BASEADDR,0) & 0xFF);
                xil_printf("SW: %d\n\r", sw_data);

                seg_data = ((sw_data & 0x80) << 24)
                                + ((sw_data & 0x40) << 20)
                                + ((sw_data & 0x20) << 16)
                                + ((sw_data & 0x10) << 12)
                                + ((sw_data & 0x08) << 12)
                                + ((sw_data & 0x04) << 8)
                                + ((sw_data & 0x02) << 4)
                                + (sw_data & 0x01);
                xil_printf("segdata: %u\n\r", seg_data);

                FND_mWriteReg (XPAR_FND_0_S00_AXI_BASEADDR, 0, seg_data);

                for(int i= 0; i<6000000 ; i++);
        }

}
```

Application 프로젝트 소스 내용 변경

HUINS
Human Intelligent System

Program FPGA

PC에 보드 연결

Bitstream

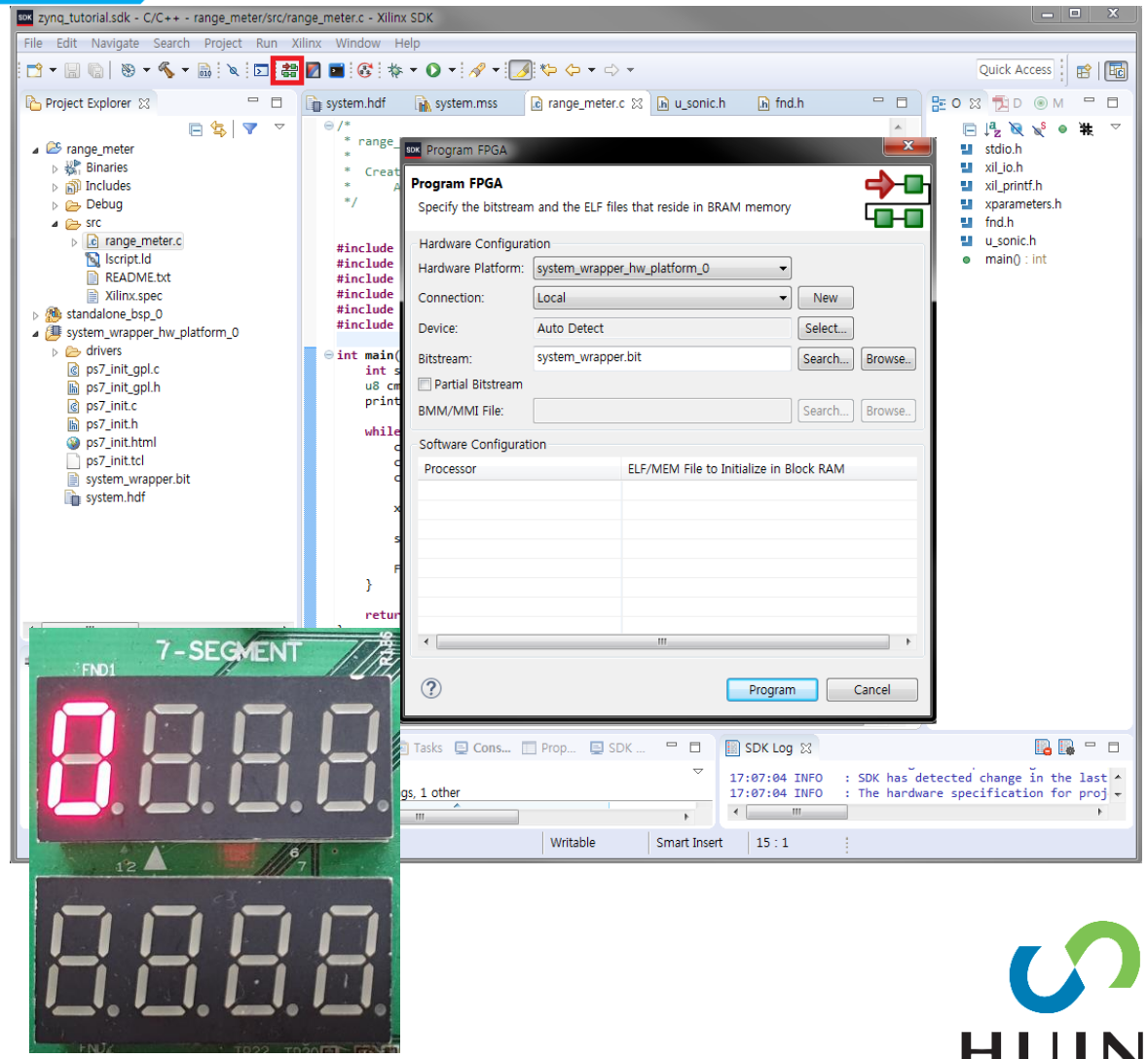
FPGA에 Program

→7-Segment 초기화

7-segment IP는 데이터 write

현재 입력 데이터 0,

클럭이 없어 0번 7-Segment만 켜짐

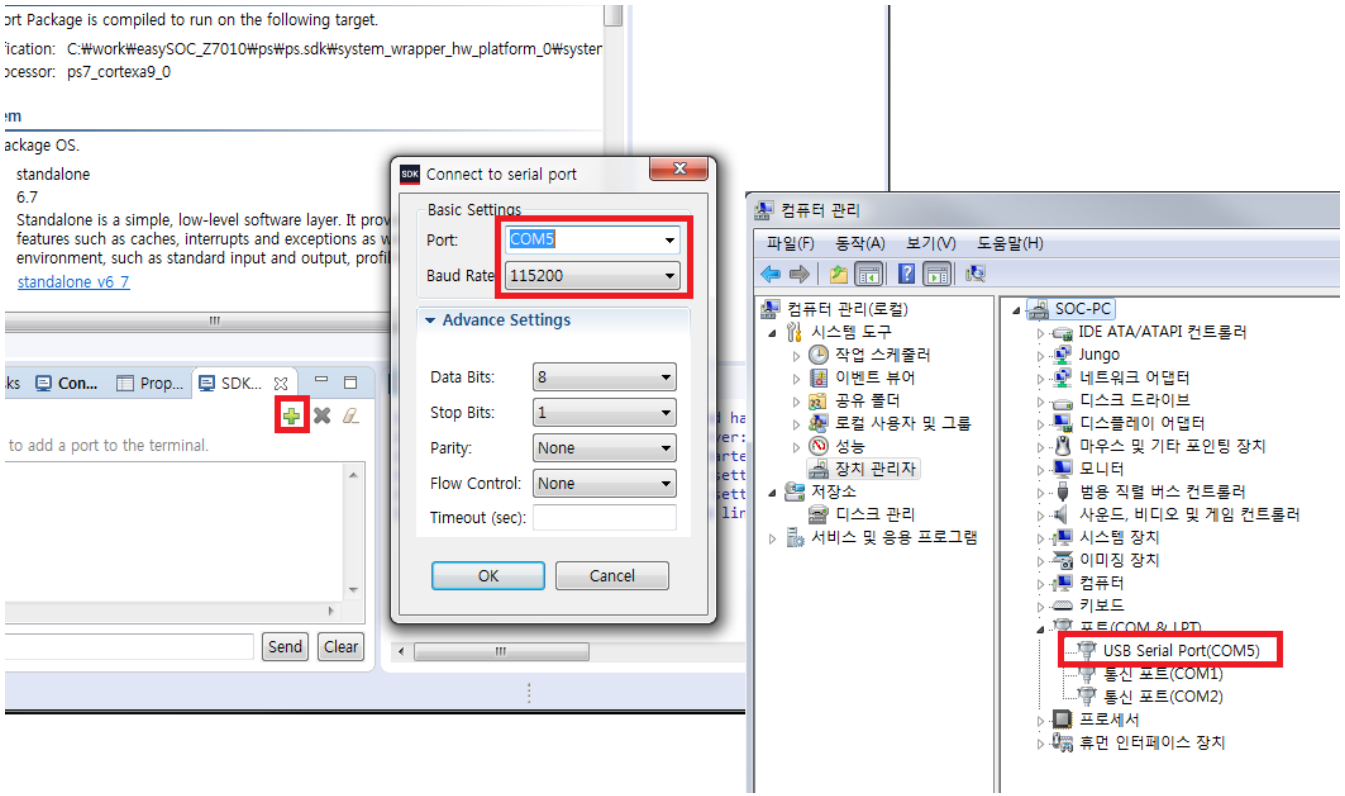


HUINS
Human Intelligent System

Terminal 연결

시리얼 포트 연결

- Baud Rate: 115200
- Data Bits: 8
- Stop Bits: 1
- Parity: None
- Flow Control: None



HUINS
Human Intelligent System

애플리케이션 실행

Run As

→Launch on Hardware

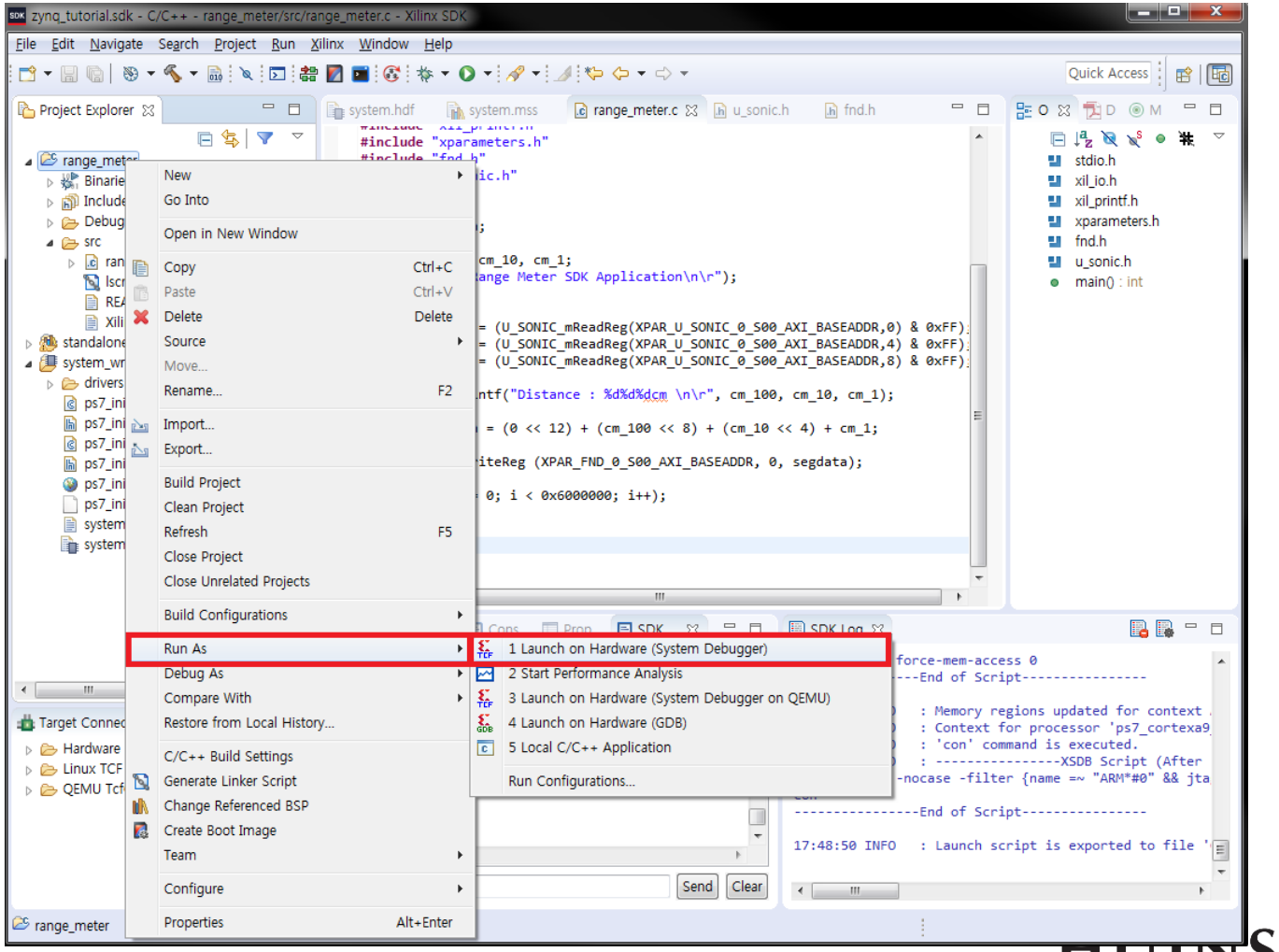
소스코드 컴파일

시스템을 초기화

애플리케이션 탑재 후 실행

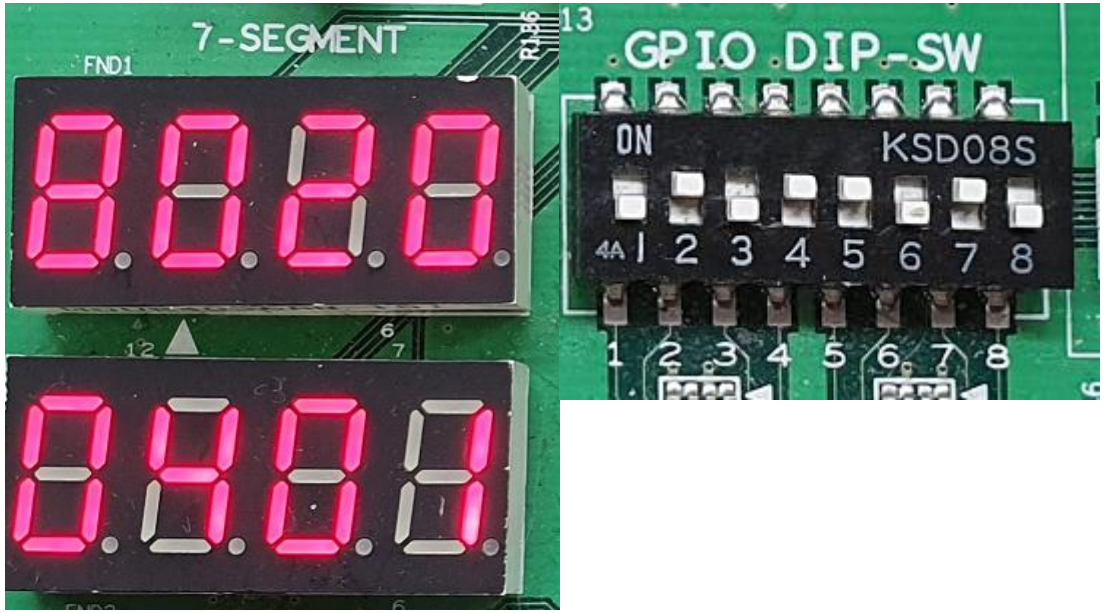
DDR3 Memory

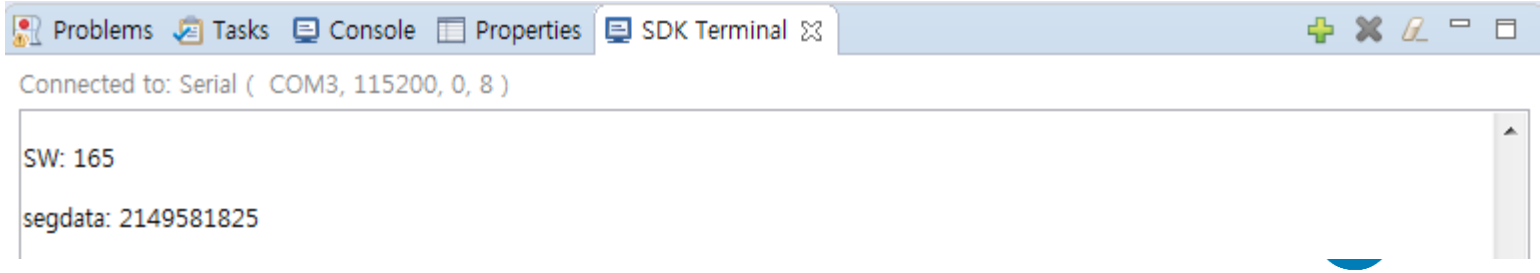
(0x0010_0000 – 3FFF_FFFF)



애플리케이션 실행

DIP SW의 왼쪽 4비트
→상단 7-Segment
DIP SW의 오른쪽 4비트
→하단 7-Segment





HUINS
Human Intelligent System

시스템 동작 설명(Read Transaction)

```
sw_data = (DIP_SW_mReadReg(XPAR_DIP_SW_0_S00_AXI_BASEADDR,0) & 0xFF);
```

PS에서 Read Transaction 시

AXI_GP_Master0 – Interconnect IP 거쳐

DIP_SW IP 레지스터 값 Read

0x43C00000+0 → sw_data

0x43C00000+4 → slv_reg1

0x43C00000+8 → slv_reg

```
369     always @(*)
370     begin
371         // Address decoding for reading registers
372         case ( axi_araddr[ADDR_LSB+OPT_MEM_ADDR_BITS:ADDR_LSB] )
373           2'h0   : reg_data_out <= DIP_SW;
374           2'h1   : reg_data_out <= slv_reg1;
375           2'h2   : reg_data_out <= slv_reg2;
376           2'h3   : reg_data_out <= slv_reg3;
377           default : reg_data_out <= 0;
378         endcase
379     end
380
```

ex)sw_data = 8'b 01011100

HUINS
Human Intelligent System

시스템 동작 설명(Data Packing)

unsigned 8bit 데이터를
32bit 데이터로 패키징

```
seg_data = ((sw_data & 0x80) << 24)
         + ((sw_data & 0x40) << 20)
         + ((sw_data & 0x20) << 16)
         + ((sw_data & 0x10) << 12)
         + ((sw_data & 0x08) << 12)
         + ((sw_data & 0x04) << 8)
         + ((sw_data & 0x02) << 4)
         + (sw_data & 0x01);
```

	[31:24]	[23:16]	[15:8]	[7:0]	
0X43C10000	seg7/seg6	seg5/seg4	seg3/seg2	seg1/seg0	slv_reg0

HUINS
Human Intelligent System

시스템 동작 설명(Write Transaction)

seg_data: 32비트 packing

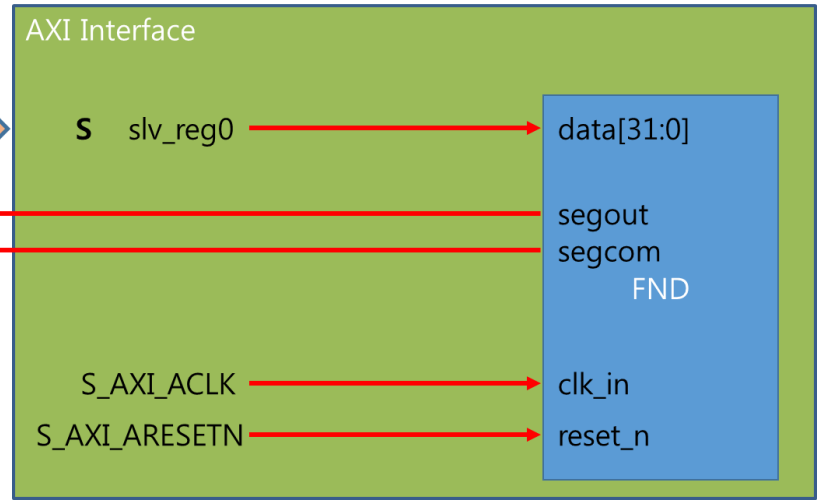
PS에서 Write 시
AXI_GP_Master0 – Interconnect IP 거쳐
7-Segment IP에 Write
slv_reg0에 segdata 입력

slv_reg0 32비트가
7-Segment에 출력됨

```
FND_mWriteReg (XPAR_FND_0_S00_AXI_BASEADDR, 0, seg_data);
```

```
case ( axi_awaddr[ADDR_LSB+OPT_MEM_ADDR_BITS:ADDR_LSB] )
  2'h0:
    for ( byte_index = 0; byte_index <= (C_S_AXI_DATA_WIDTH/8)-1; byte_index = byte_index+1 )
      if ( S_AXI_WSTRB[byte_index] == 1 ) begin
        // Respective byte enables are asserted as per write strobes
        // Slave register 0
        slv_reg0[(byte_index*8) +: 8] <= S_AXI_WDATA[(byte_index*8) +: 8];
      end
```

```
401        // Add user logic here
402        seg_top seg_top_inst(
403            .clk_in(S_AXI_ACLK),
404            .reset_n(S_AXI_ARESETN),
405            .data(slv_reg0),
406            .segout(segout),
407            .segcom(segcom)
408        );
409        // User logic ends
```



	[31:24]	[23:16]	[15:8]	[7:0]	
0X43C10000	seg7/seg6	seg5/seg4	seg3/seg2	seg1/seg0	slv_reg0

HUINS
Human Intelligent System

Summary

FPGA IP를 AXI-4 Protocol IP로 생성할 수 있다.

- 7-Segment → Write Transaction
- DIP-SW → Read Transaction

Vivado에서 Processing System의 설정을 할 수 있다.

- MIO/AXI Interface/DDR Controller

Block Design에서 AXI IP를 추가하여 시스템 구성을 할 수 있다.

- PS – PL 연결(Block Design)

시스템을 Export 하여 SDK에서 HW 기반의 SW를 개발할 수 있다.

PS에서 AXI IP의 Read/Write Transaction을 수행할 수 있다.

References

- ZYNQ Technical Reference Guide(UG-585)
- THE ZYNQ BOOK(http://www.zynqbook.com)
- Vivado-Intro-FPGA-Design-HLS(UG998)