

DE2 BOARD 101

Electronic Engineering
JAICHANGPARK(Dreamwalker)
Master Degree

➤ 들어가기 전

사실

De2 보드를 사용해 포팅을 하고 코드를 짜고 다 처음 해보는 부분이었습니다.

일주일간의 시간 동안

1. 포팅하는 과정,
2. 프로젝트를 생성하고 파일을 만드는 과정
3. 또 프로그램해서 보드에 올리는 과정 (이 부분이 몇 일 해도 잘 되지 않았습니다.)

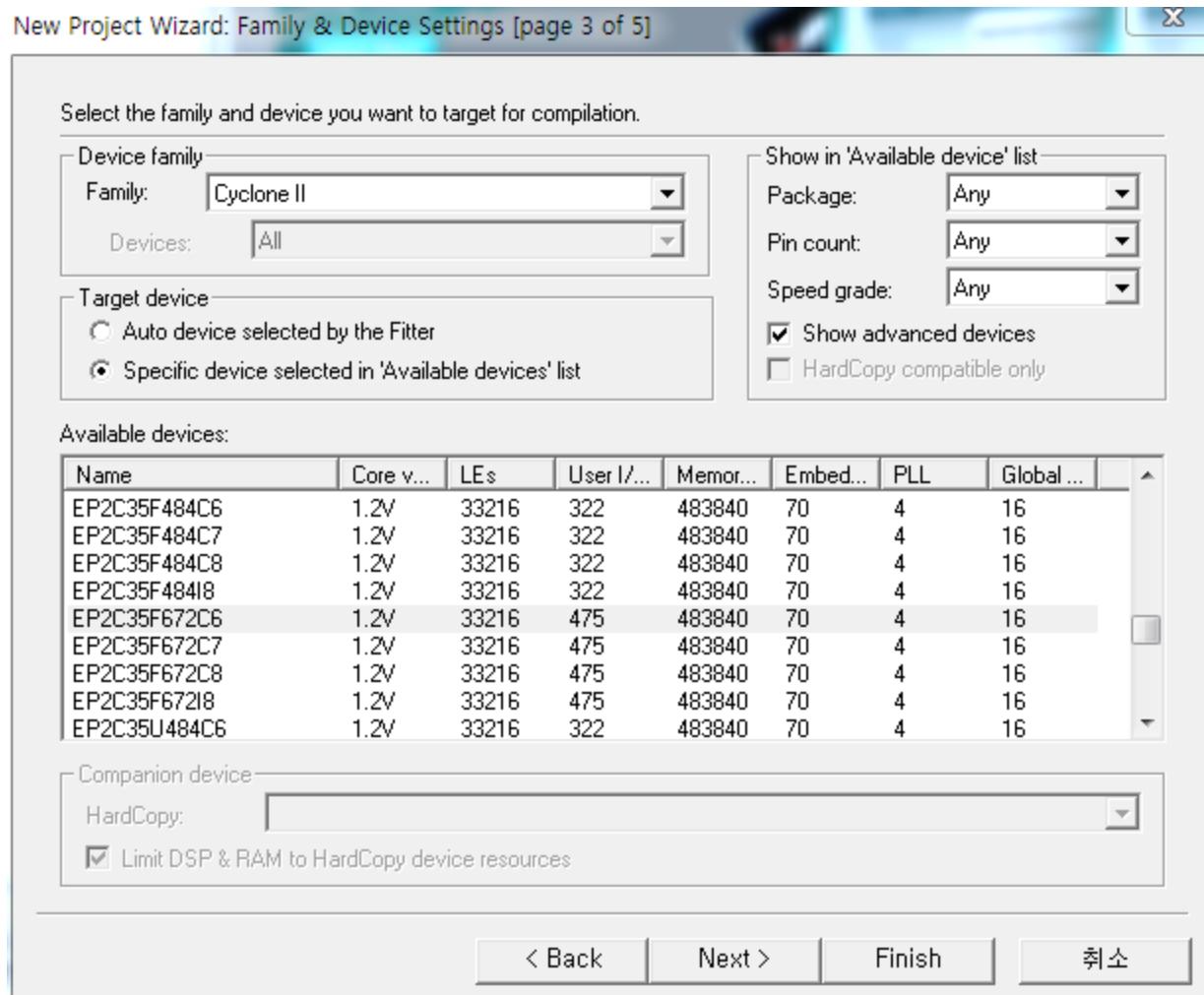
처음이라 쉽지는 않았습니다만

고통신호 제어 등을 위한 제작과정을 이 자료에 담아 봤습니다.

➤ 프로젝트 생성시 Cpu 설정방법



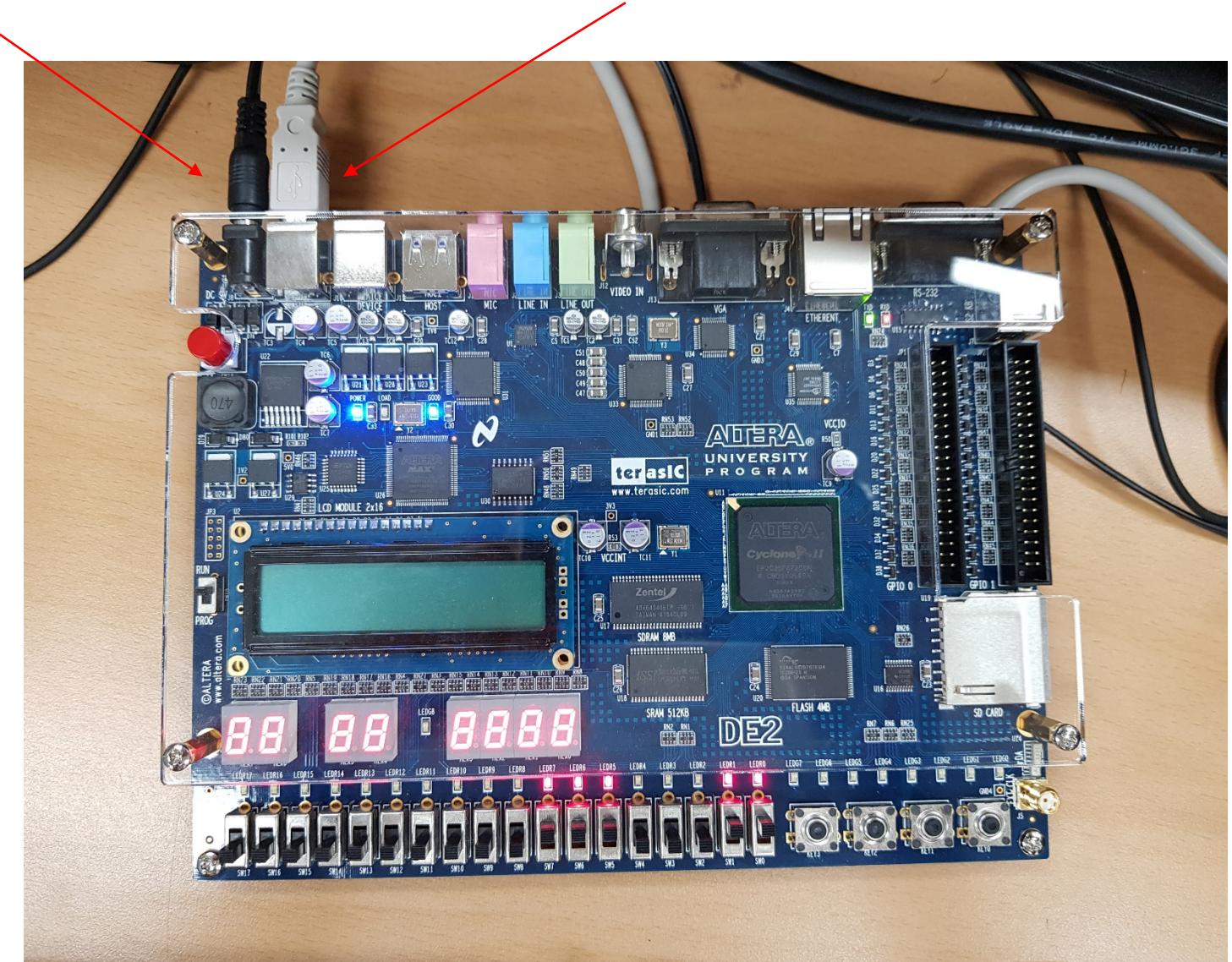
Cpu 설정방법



프로그램 방법

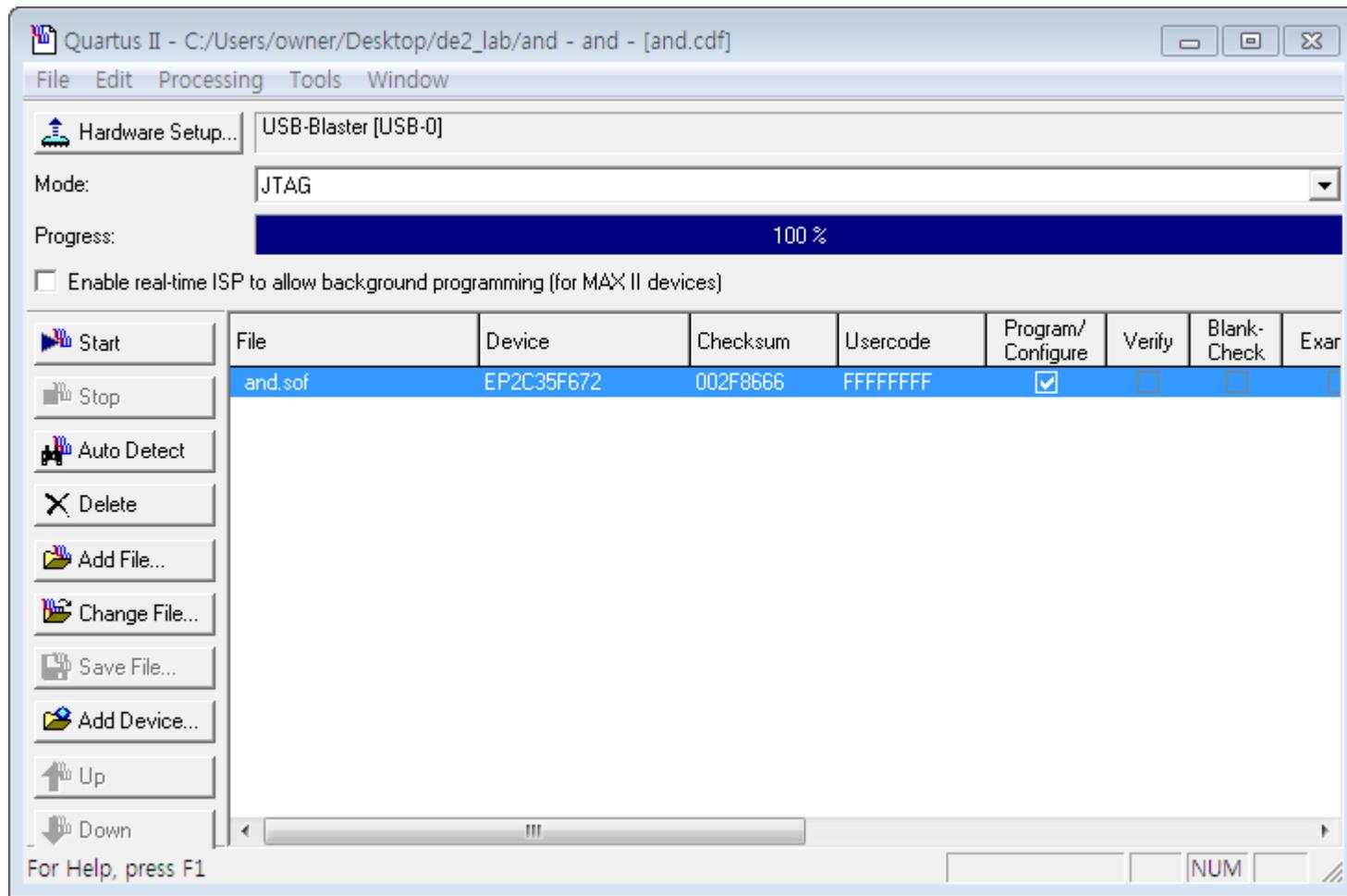
전원 인가는 필수

컴퓨터 다운로드 케이블을
Bluster포트에 연결해야함.



프로그램 방법

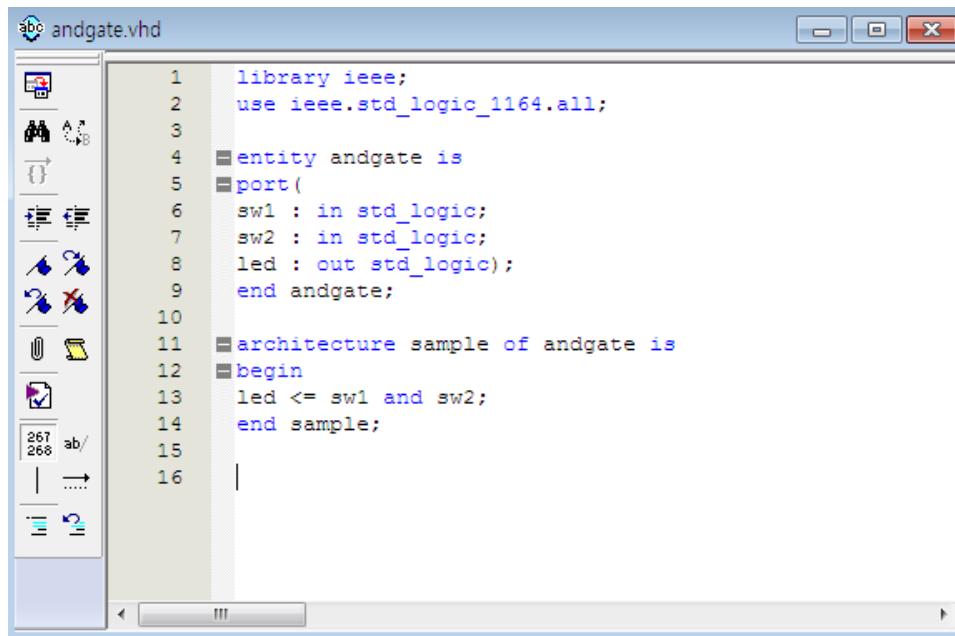
이거 아느라 시간 오래걸렸습니다...
우선 드라이버 설치가 필요합니다.



**드라이버 설치는
장치관리자에서 진행합니다**

드라이버는 큐터스 풀더 내부에 있습니다.

- 첫번째 실험: AND 게이트를 생성하고 led를 우선 켜봤습니다.



The screenshot shows a VHDL code editor window titled "andgate.vhd". The code defines an entity "andgate" with two inputs (sw1, sw2) and one output (led). The architecture "sample" implements a simple AND logic function. The code is as follows:

```
library ieee;
use ieee.std_logic_1164.all;

entity andgate is
port(
    sw1 : in std_logic;
    sw2 : in std_logic;
    led : out std_logic);
end andgate;

architecture sample of andgate is
begin
    led <= sw1 and sw2;
end sample;
```

```
library ieee;
use ieee.std_logic_1164.all;
```

```
entity andgate is
port(
    sw1 : in std_logic;
    sw2 : in std_logic;
    led : out std_logic);
end andgate;
```

```
architecture sample of andgate is
begin
    led <= sw1 and sw2;
end sample;
```

1. 프로젝트생성

2. 파일 추가 및 저장

3. 코드 작성

4. 컴파일하기

5. 핀 포팅

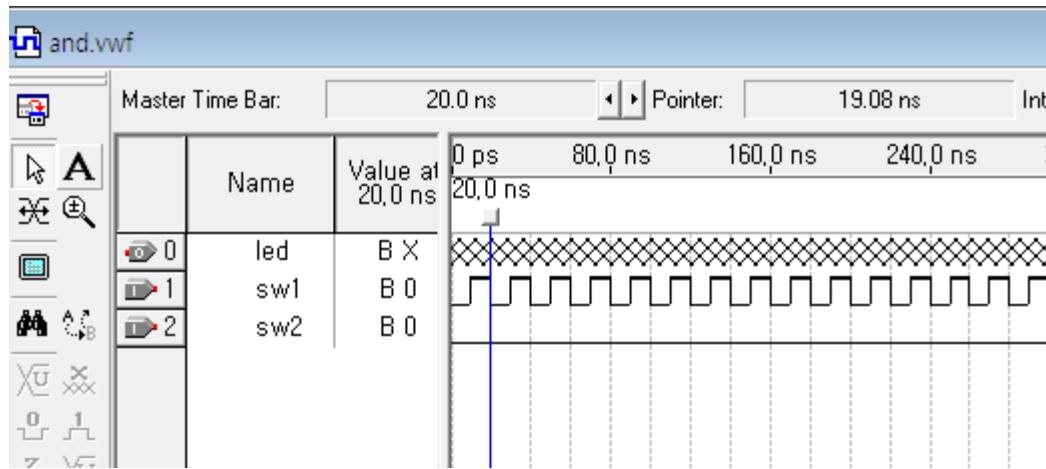
6. 시뮬레이션 확인

7. 컴파일

8. 프로그램 업로드

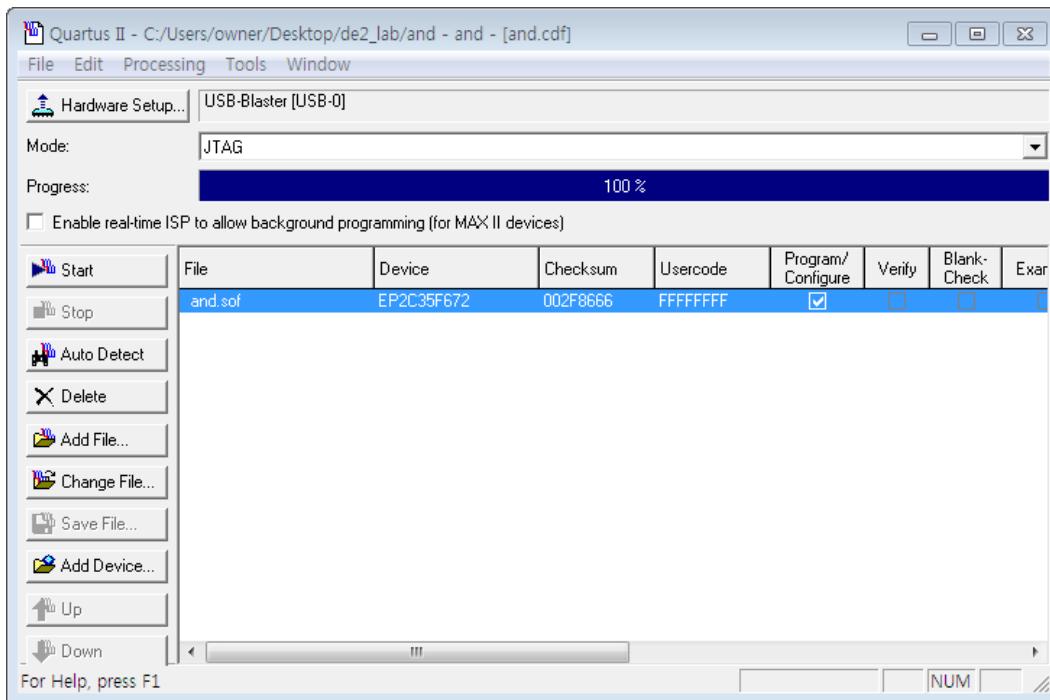
- 첫번째 실험: AND 게이트를 생성하고 led를 우선 켜봤습니다.

시뮬레이션 동작 확인하기 .



- 첫번째 실험: AND 게이트를 생성하고 led를 우선 켜봤습니다.

프로그램 넣기



1. JTAG 프로세스 설정

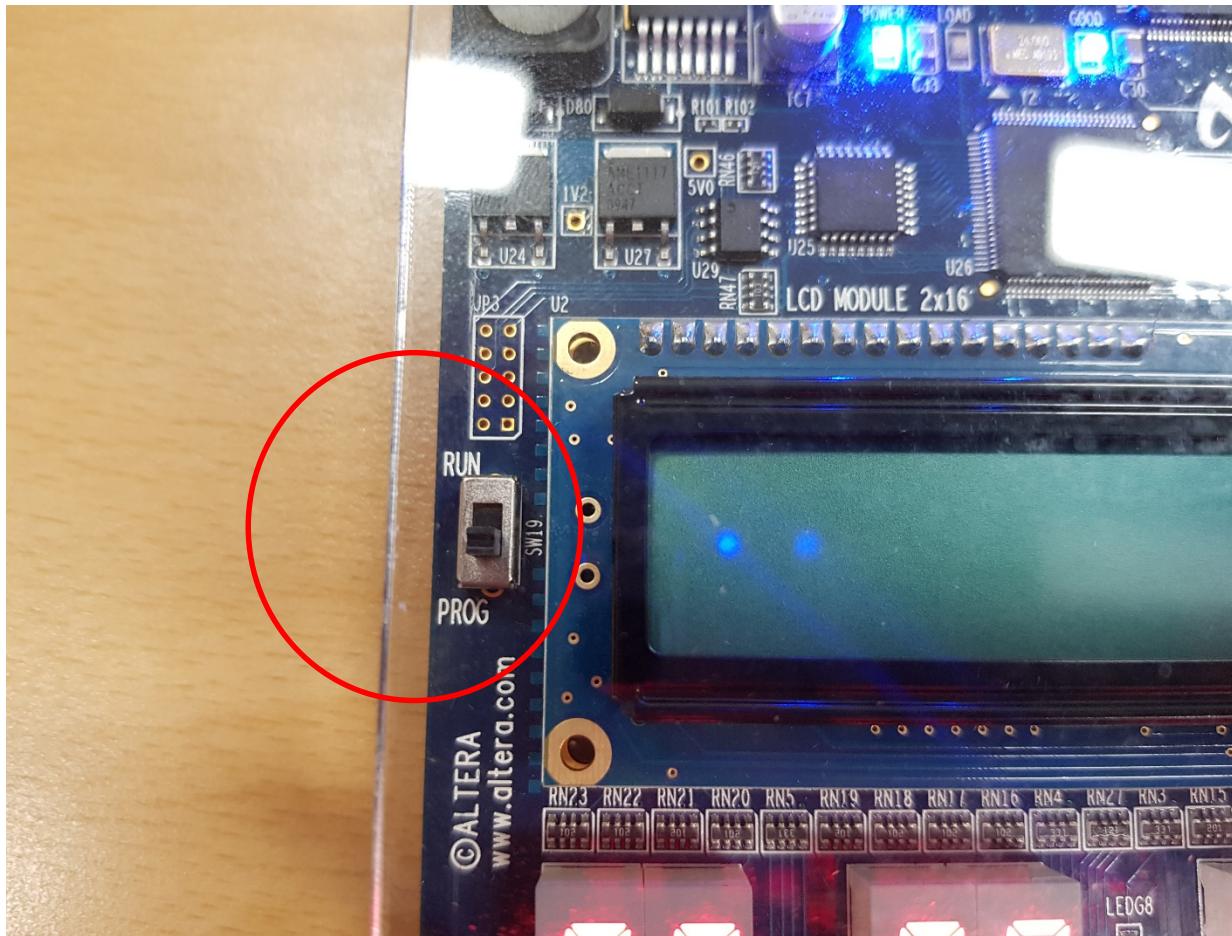
2. 하드웨어 설정에서 장치 관리자
에서 등록한 USB 버스터 등록

3. 왼쪽 상단의 스타트 버튼 클릭해
보드에 업로드

- 첫번째 실험: AND 게이트를 생성하고 led를 우선 켜봤습니다.

업로드 시 주의 사항

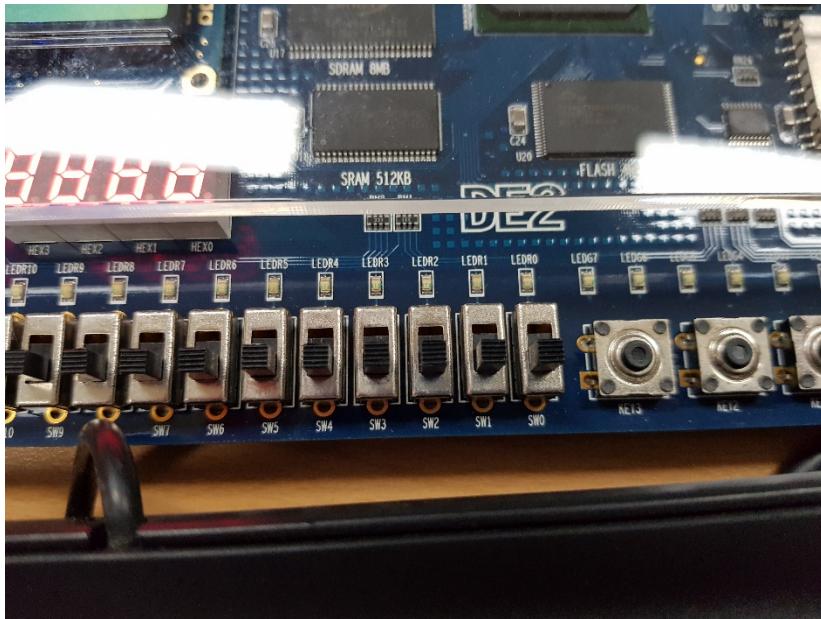
보드에 업로드 할 때 주의해야 하는 것은 CLCD 옆에 스위치가 하나 있습니다. 이 걸 프로그램 모드로 변경해야 합니다.



- 첫번째 실험: AND 게이트를 생성하고 led를 우선 켜봤습니다.

실험 결과

스위치 1, 스위치2, led 1 포팅은 성공적이었고 포팅하는 방법을 익혔다.
And gate를 설계한 만큼 스위치 1, 스위치 2를 올리면 led1 이 켜진다.



- 두 번째 실험: 스위치로 LED 켜기 .

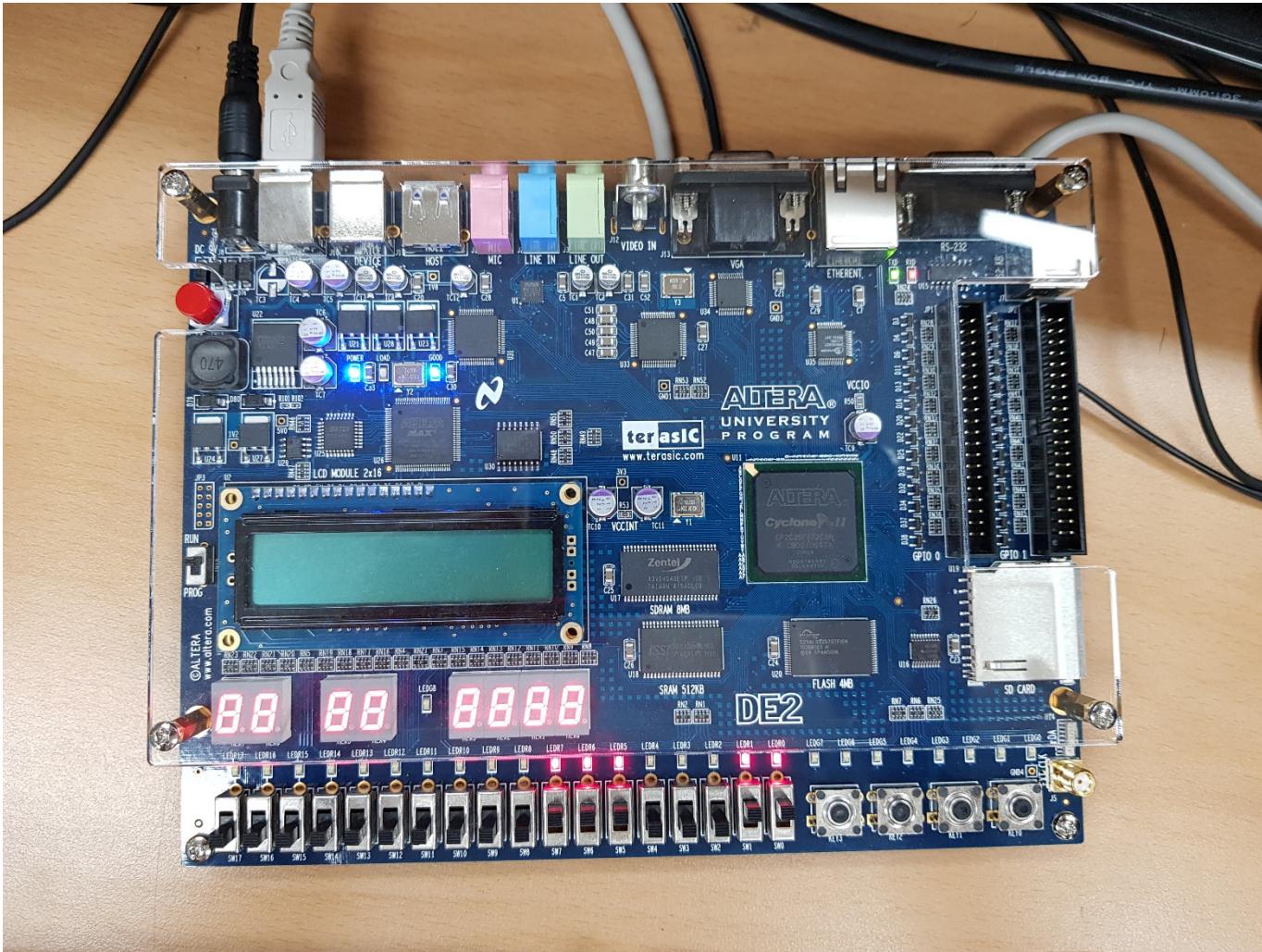
벡터를 사용한 모든스위치 제어를 통한 LED 개별 점등

```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY part1 IS
PORT (    SW : IN STD_LOGIC_VECTOR(17 DOWNTO 0);
           LEDR : OUT STD_LOGIC_VECTOR(17 DOWNTO 0)); --red LEDs
END part1;
ARCHITECTURE Behavior OF part1 IS
BEGIN
    LEDR <= SW;
END Behavior;
```

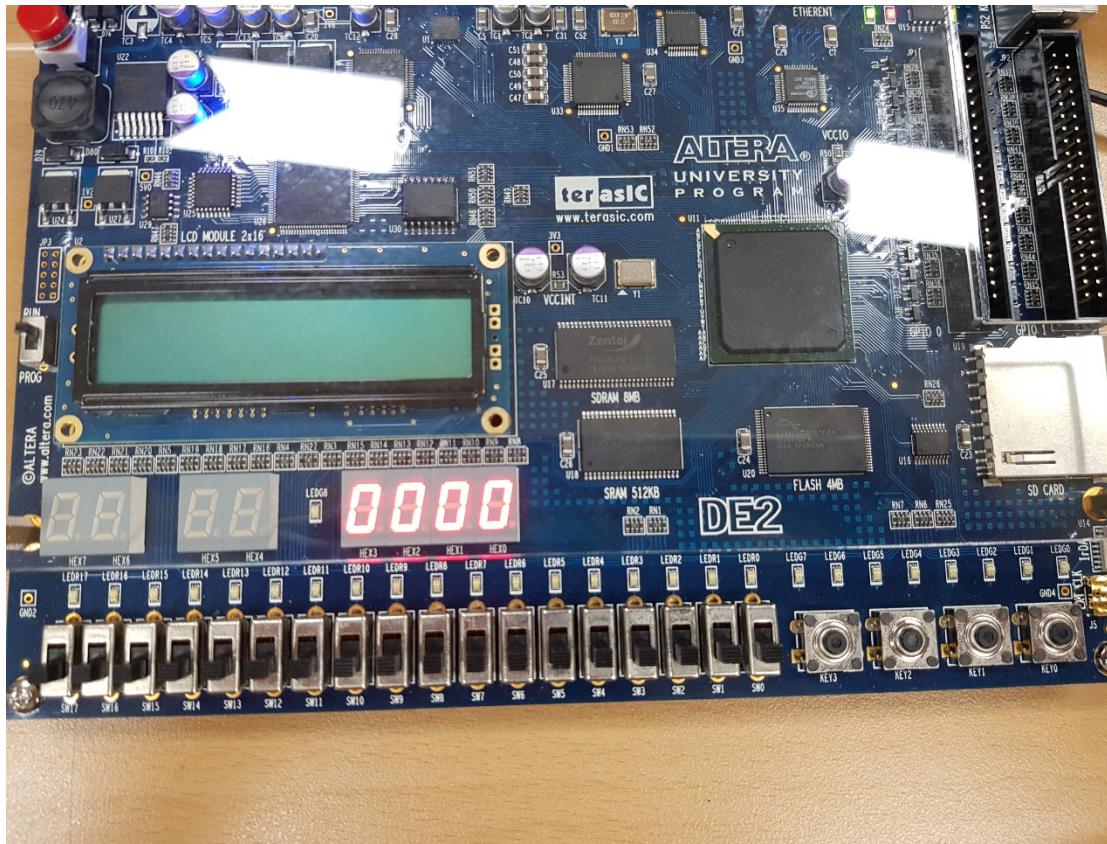
- 두 번째 실험: 스위치로 LED 켜기 .

실험 결과



- 세 번째 실험: 스위치와 푸시 스위치를 통해 세그먼트에 비트 표기하기 .

17개의 스위치를 비트라고 보고 0011의 신호를 세팅하고 스위치 3을 누르면 세그먼트에 표기하는 방식



- 세 번째 실험: 스위치와 푸시 스위치를 통해 세그먼트에 비트 표기하기 .

➤ 베릴로그 세그먼트 코드

```
module hex_7seg(hex_digit,seg);
input [3:0] hex_digit;
output [6:0] seg;
reg [6:0] seg;
// seg = {g,f,e,d,c,b,a};
// 0 is on and 1 is off

always @ (hex_digit)
case (hex_digit)
    4'h0: seg = ~7'h3F;      // 00111111 g,f,e,d,c,b,a
    4'h1: seg = ~7'h06;      // ---a-----
    4'h2: seg = ~7'h5B;      // |       |
    4'h3: seg = ~7'h4F;      // f       b
    4'h4: seg = ~7'h66;      // |       |
    4'h5: seg = ~7'h6D;      // ---g-----
    4'h6: seg = ~7'h7D;      // |       |
    4'h7: seg = ~7'h07;      // e       c
    4'h8: seg = ~7'h7F;      // |       |
    4'h9: seg = ~7'h67;      // ---d-----
    4'ha: seg = ~7'h77;
    4'hb: seg = ~7'h7C;
    4'hc: seg = ~7'h39;
    4'hd: seg = ~7'h5E;
    4'he: seg = ~7'h79;
    4'hf: seg = ~7'h71;

endcase

endmodule
```

■ 세 번째 실험: 스위치와 푸시 스위치를 통해 세그먼트에 비트 표기하기 .

▶ 베릴로그 상위레벨 코드

```

module segmentlab1(                                     // map to 7-segment displays
    // Clock Input (50 MHz)
    input CLOCK_50,
    // Push Buttons
    input [3:0] KEY,
    // DPDT Switches
    input [17:0] SW,
    // 7-SEG Displays
    output [6:0] HEX0, HEX1, HEX2, HEX3, HEX4, HEX5, HEX6,
    HEX7,
    // LEDs
    output [8:0] LEDG, // LED Green[8:0]
    output [17:0] LEDR, // LED Red[17:0]
    // GPIO Connections
    inout [35:0] GPIO_0, GPIO_1
);

// set all inout ports to tri-state
assign GPIO_0      = 36'hzzzzzzzz;
assign GPIO_1      = 36'hzzzzzzzz;

// Connect dip switches to red LEDs
assign LEDR[17:0] = SW[17:0];

// turn off green LEDs
assign LEDG[8:0] = 0;

reg [15:0] A;

hex_7seg dsp0(A[3:0],HEX0);
hex_7seg dsp1(A[7:4],HEX1);
hex_7seg dsp2(A[11:8],HEX2);
hex_7seg dsp3(A[15:12],HEX3);

wire [6:0] blank = ~7'h00;

// blank remaining digits
assign HEX4 = blank;
assign HEX5 = blank;
assign HEX6 = blank;
assign HEX7 = blank;

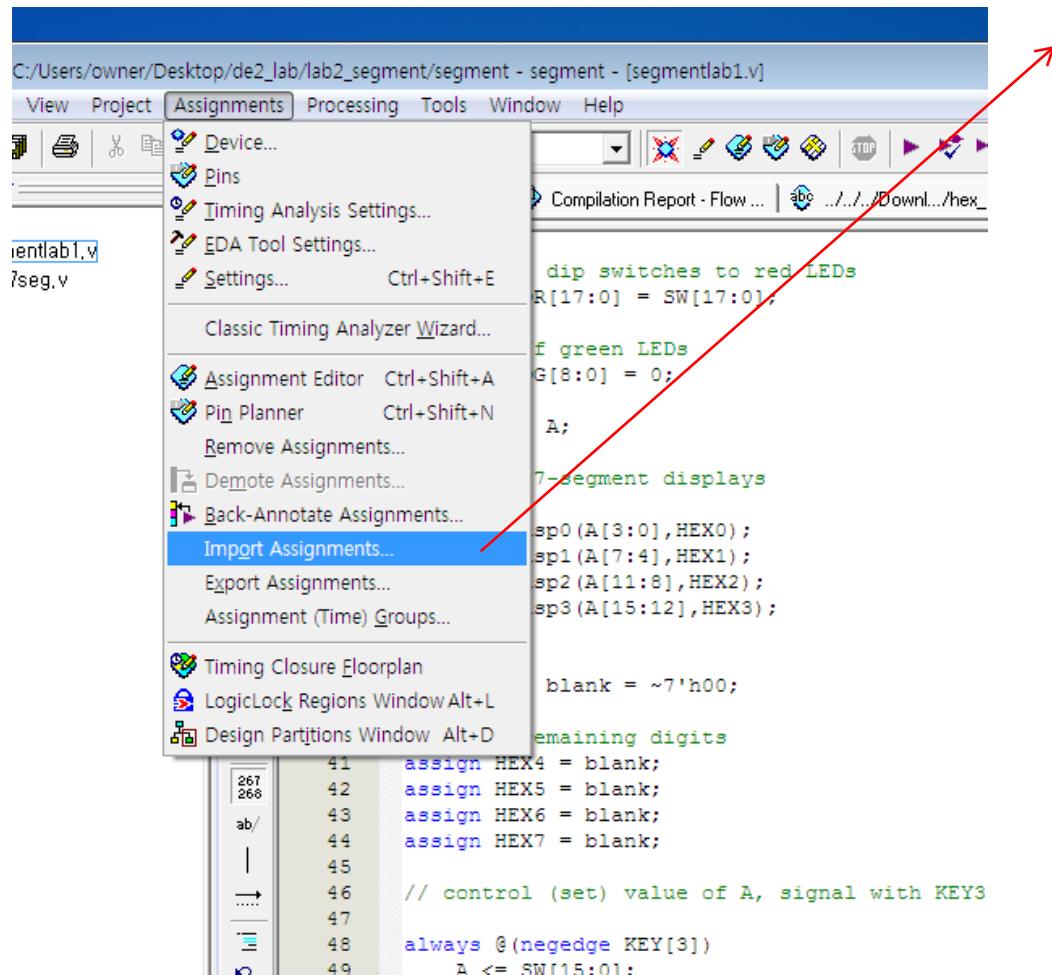
// control (set) value of A, signal with
KEY3

always @(negedge KEY[3])
    A <= SW[15:0];

endmodule

```

➤ 핀 포팅을 쉽게 하는 방법.



The screenshot shows the Quartus II software interface. The menu bar is visible at the top, and the 'Assignments' menu is currently selected. A red arrow points from the text 'Import Assignments...' to the menu item in the screenshot. The main window displays Verilog code for a 7-segment display. The code includes assignments for pins R[17:0] and G[8:0], and logic for segments sp0 through sp3. It also includes assignments for HEX4 through HEX7 and a control section for KEY3.

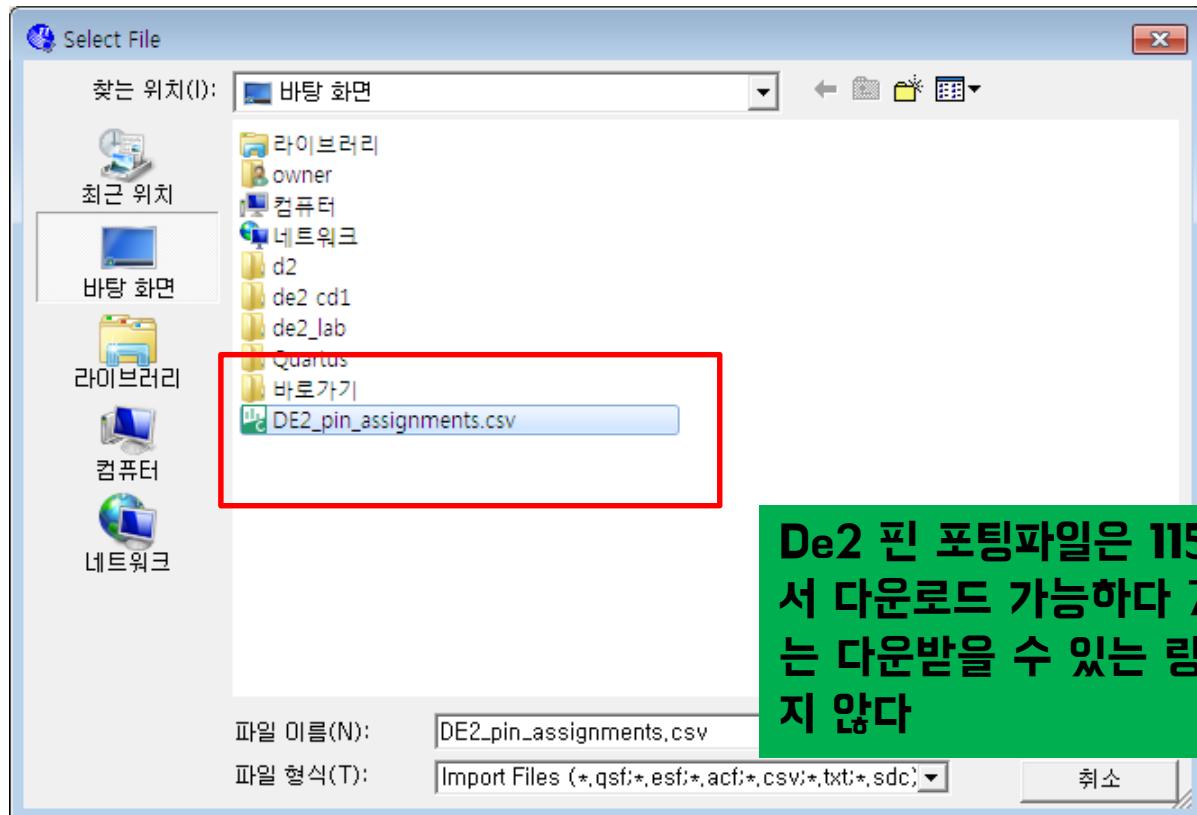
```
dip switches to red LEDs  
R[17:0] = SW[17:0];  
  
f green LEDs  
G[8:0] = 0;  
  
A;  
  
7-segment displays  
sp0(A[3:0],HEX0);  
sp1(A[7:4],HEX1);  
sp2(A[11:8],HEX2);  
sp3(A[15:12],HEX3);  
  
blank = ~7'h00;  
  
remaining digits  
41 assign HEX4 = blank;  
42 assign HEX5 = blank;  
43 assign HEX6 = blank;  
44 assign HEX7 = blank;  
45  
46 // control (set) value of A, signal with KEY3  
47  
48 always @ (negedge KEY[3])  
49 A <= SW[15:0];
```

DE2 보드만의 핀을 Assign하는 방법은 이미 mapping된 csv 파일을 import 하는 방법이 있다.

이 방법은 실습을 해가면서 찾은 방법이다.

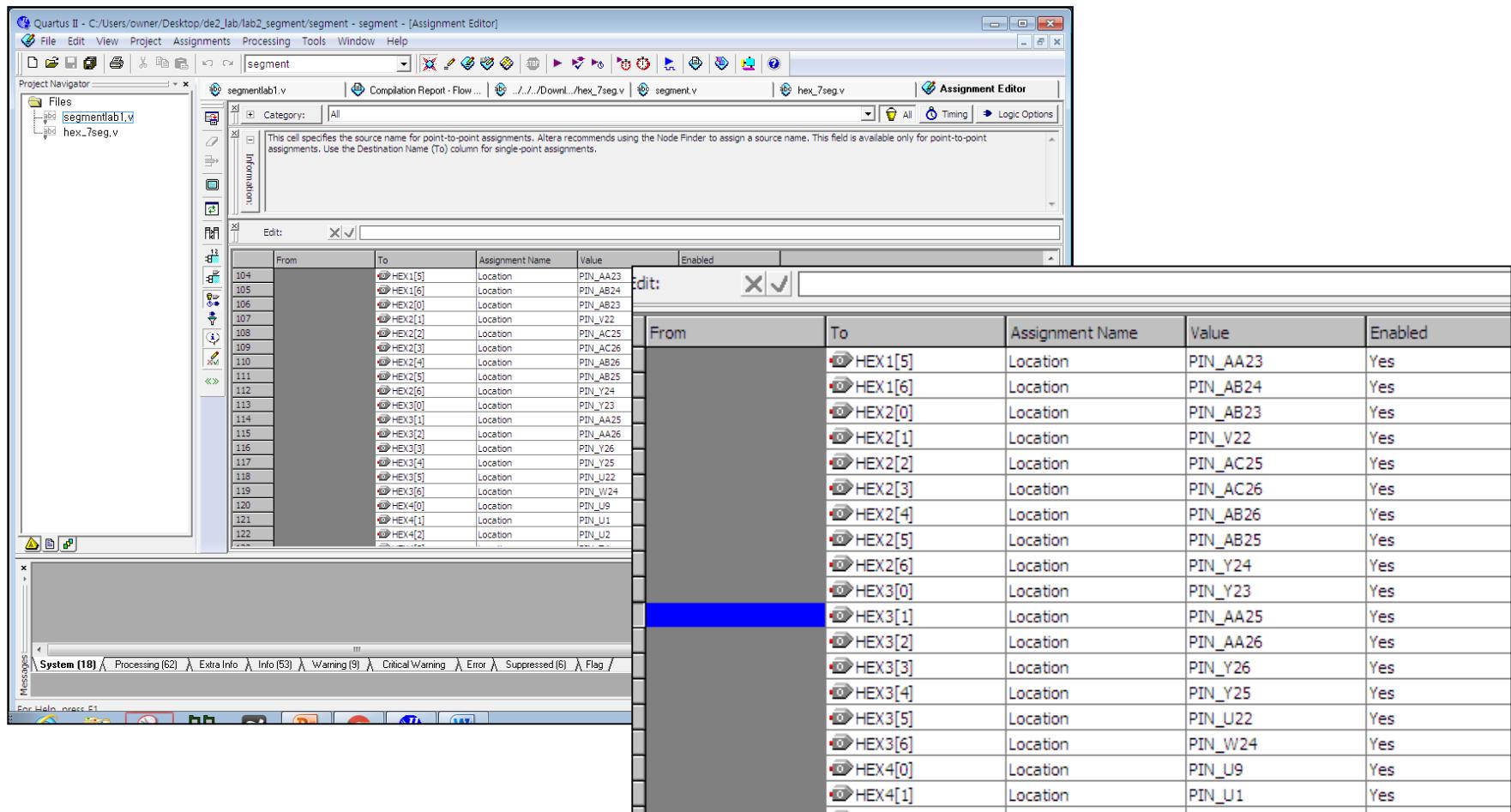
단 코드에서 변수명을 동일하게 설정해 줘야한다.

➤ 핀 포팅을 쉽게 하는 방법.



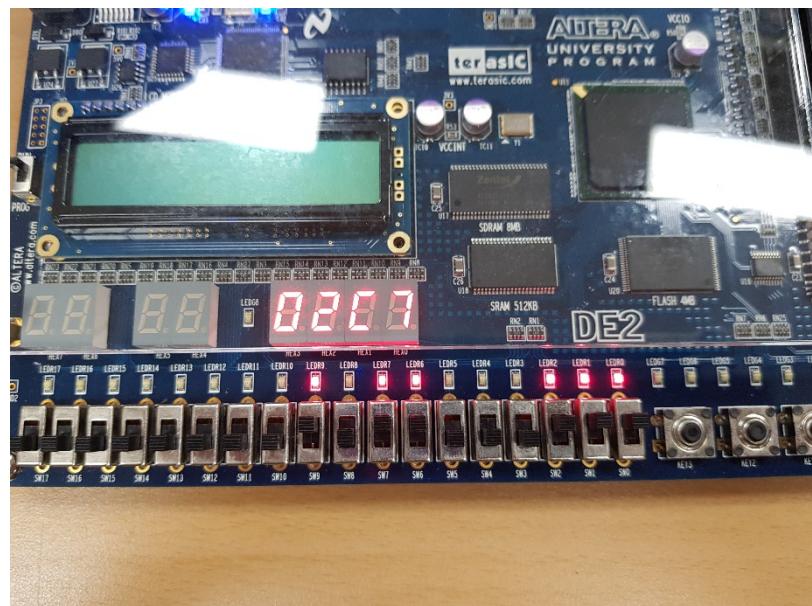
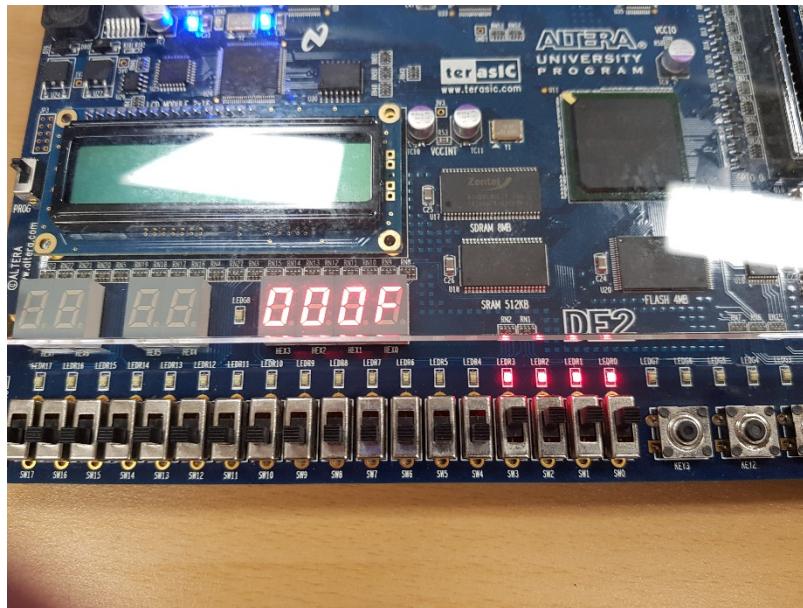
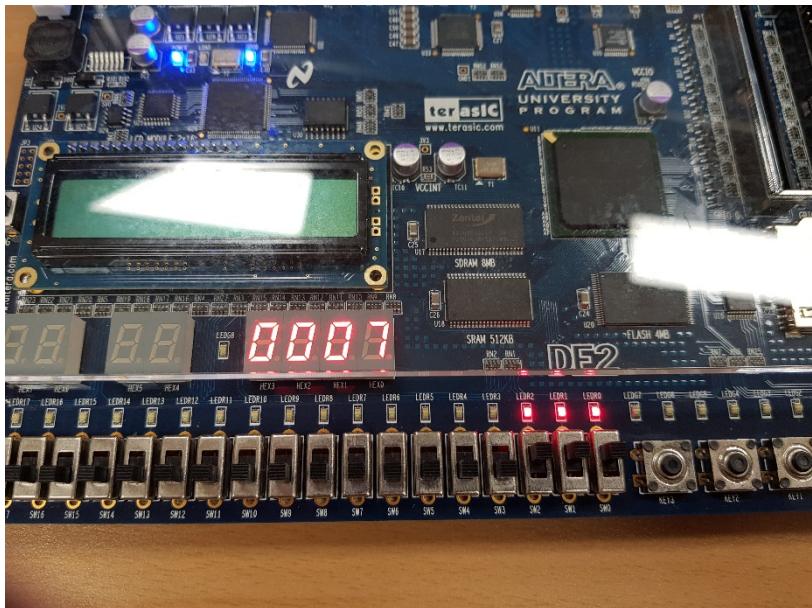
**De2 핀 포팅파일은 115버전까지는 인텔에
서 다운로드 가능하다 70버전의 핀 포트
는 다운받을 수 있는 링크를 제공하고 있
지 않다**

➤ 핀 포팅을 쉽게 하는 방법.



Import가 성공적으로 진행된다면 다음과 같이 맵핑된
핀 결과를 확인할 수 있다.

■ 세 번째 실험: 스위치와 푸시 스위치를 통해 세그먼트에 비트 표기하기 .

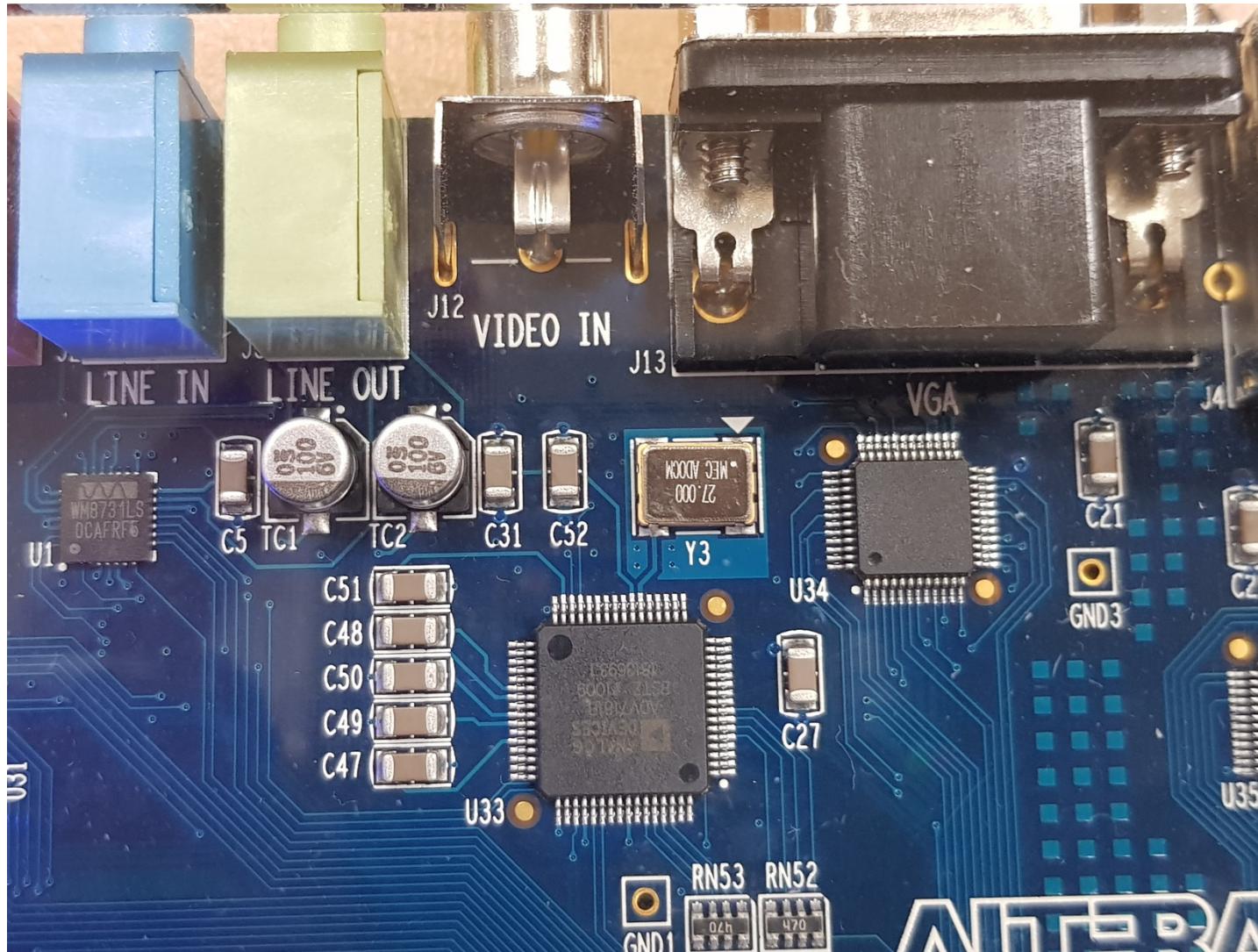


- 네 번째 실험: 교통신호 제어를 위한 기본 타이머

LEDG[5]	Location	PIN_U1/	Yes
LEDG[6]	Location	PIN_AA20	Yes
LEDG[7]	Location	PIN_Y18	Yes
LEDG[8]	Location	PIN_Y12	Yes
CLOCK_27	Location	PIN_D13	Yes
CLOCK_50	Location	PIN_N2	Yes
EXT_CLOCK	Location	PIN_P26	Yes
PS2_CLK	Location	PIN_D26	Yes
PS2_DAT	Location	PIN_C24	Yes
UART_RXD	Location	PIN_C25	Yes

DE2 보드는 27M, 50MHz 내부 발진 크리스탈을 통해 클럭 생성을 하고 있으며 이를 분주하여 적당한 시스템 클럭으로 분주하는 것이 목표이다.

- 네 번째 실험: 교통신호 제어를 위한 기본 타이머



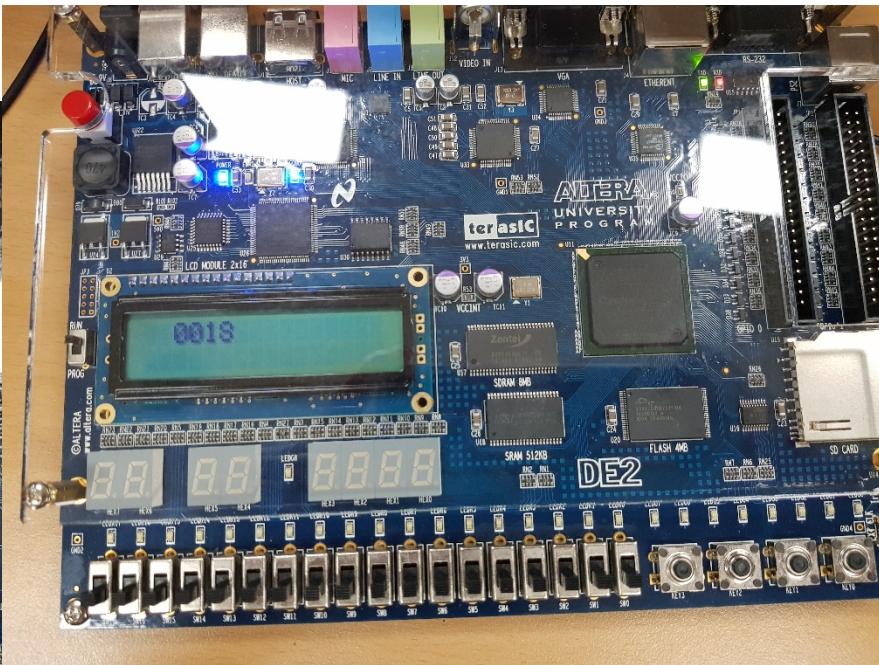
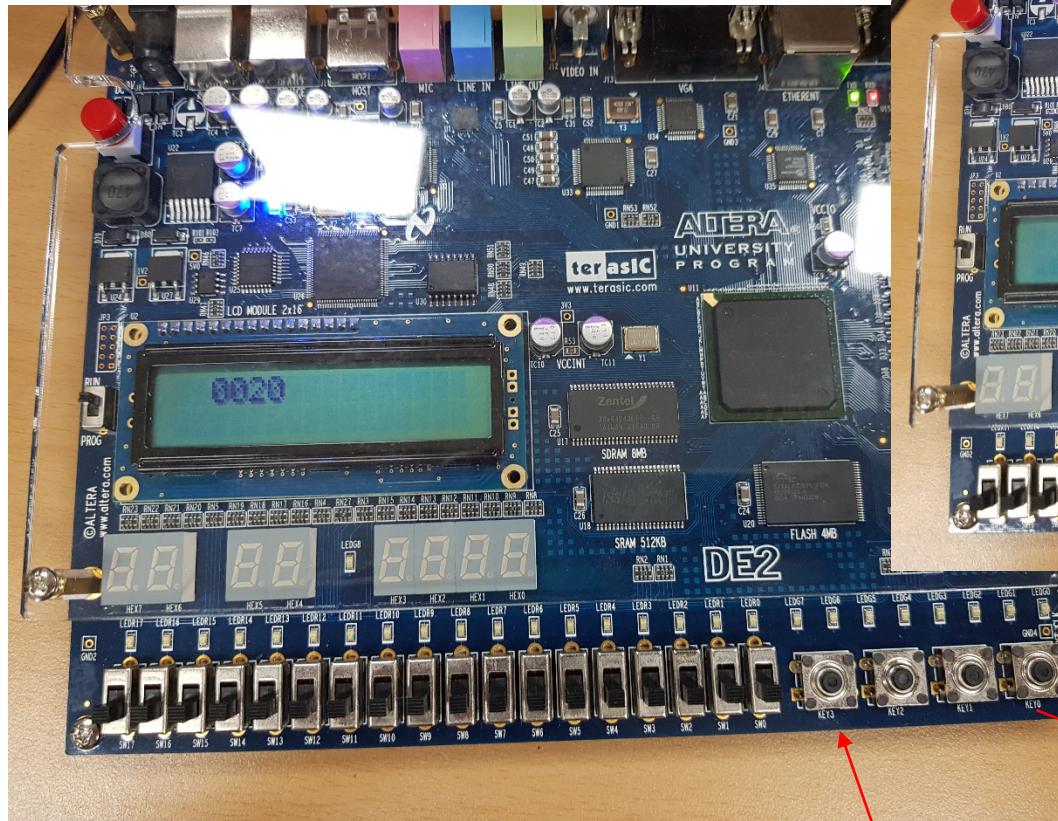
27MHz 실장 확인

- 네 번째 실험: 교통신호 제어를 위한 기본 타이머



50MHz 실장 확인

- 네 번째 실험: 교통신호 제어를 위한 기본 타이머



Sw3를 누르면 타이머 시작

리셋