Hardware System Design, Lab 8

2013-11392 김지현

개요

이번 과제는 1초마다 숫자가 1씩 증가하는 counter를 ZedBoard의 LED로 구현하는것이다. 지난 실습들에 비하면 구현할 로직 자체는 몹시 간단하지만, 이번 실습에서 처음으로 시뮬레이터가 아닌 실제 FPGA인 ZedBoard를 사용하게 된다.

구현체 설명

```
module one_sec_checker(
    /* 100MHz clock */ input GCLK,
    /* Centor Button */ input BTNC,
    /* LEDs */ output [7:0] LD
);
    reg [7:0] seconds; initial seconds = 0;
    reg [27:0] ten_nanoseconds; initial ten_nanoseconds = 0;
    always @(posedge GCLK) begin
        if (BTNC) begin
            seconds = 0; ten_nanoseconds = 0;
        end else begin
            ten_nanoseconds = ten_nanoseconds + 1;
            if (ten_nanoseconds == 100_000_000) begin
                ten_nanoseconds = 0; seconds = seconds + 1;
            end
        end
    end
    assign LD = seconds;
endmodule
```

Figure 1. Simplified source code of one_sec_checker.v

Verilog 구현 자체는 몹시 간단하였다. Figure 1과 같이, 100MHz의 진동수를 갖는 GCLK가 1억번 돌았는지 여부를 체크하는 ten_nanoseconds 레지스터와, 클락이 1억번 돌아 1초가 지날때마다 1씩 증가하는 seconds 레지스터 두개를 사용해 쉽게 구현할 수 있었다. 중앙 버튼인 BTNC가 입력될 때 레지스터의 값들이 모두 0으로 초기화되도록 하면 reset 기능도 쉽게 구현할 수 있었다.

Figure 2. I/O Port names of ZedBoard LEDs and buttons

GCLK, BTNC, LD에 맞는 I/O Port를 찾는 과정이 살짝 까다로웠다. LED와 버튼의 경우, LED와 버튼 옆에 어떤 이름의 I/O 포트를 사용하는지가 적혀있었기 때문에 쉽게 찾을 수 있었다. GCLK의 경우 ZedBoard Schematic 문서 9페이지에서 Figure 3과 같이 찾을 수 있었다.

각 I/O Port가 몇번 IO Bank에 해당하고 각 IO Bank는 전압을 얼마로 걸어야하는지 알아낼 필요가 있었는데, 이 정보는 과제에서 주어진 zedboard_master_XDC_RevC_D_v3.xdc 파일을 참고해 알아낸 결과 Table 1과 같았다. 34번 IO Bank의 경우 전압을 고를 수가 있었는데, 이 과제에선 조교님께서 예시로 사용한것과 동일하게 2.5V를 채택하였고, 이를 그대로 constraints 파일로 작성하여 성공적으로 직접 작성한 Verilog 코드가 ZedBoard의 주변장치를 사용할 수 있도록 설정하였다.

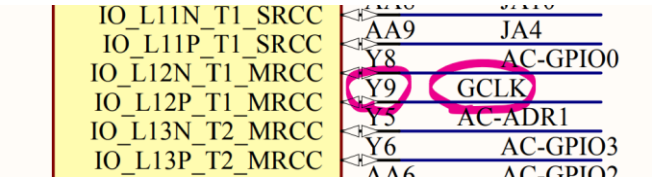

Figure 3. I/O Port name of GCLK

유형	I/O PORT	IO BANK	전압
GCLK	Y9	13	3.3V
BTNC	P16	34	1.8V, 2.5V, 3.3V
LD0	T22	33	3.3V
LD1	T21	33	3.3V
LD2	U22	33	3.3V
LD3	U21	33	3.3V
LD4	V22	33	3.3V
LD5	W22	33	3.3V
LD6	U19	33	3.3V
LD7	U14	33	3.3V

Table 1. I/O Ports constraints

실험 결과 분석

Figure 4. Testing one_sec_checker.v with ZedBoard

성공적으로 synthesize하여 ZedBoard에 프로그래밍할 수 있었고, Figure 4와 같이 직접 테스트해볼 수 있었다. 정상적으로 동작하였다.

결론

이번 실습을 통해 처음으로 시뮬레이터가 아닌 실제 FPGA를 써볼 수 있었다. ZedBoard 문서를 보고 직접 I/O Port constraint를 작성하는 방법 또한 익힐 수 있었다. 시뮬레이터가 아닌 실제 FPGA를 드디어 써볼 수 있게 되어서 너무 재밌었고, 지금까지 구현했던 PE를 FPGA에 올릴 앞으로의 실습들이 기대된다.