

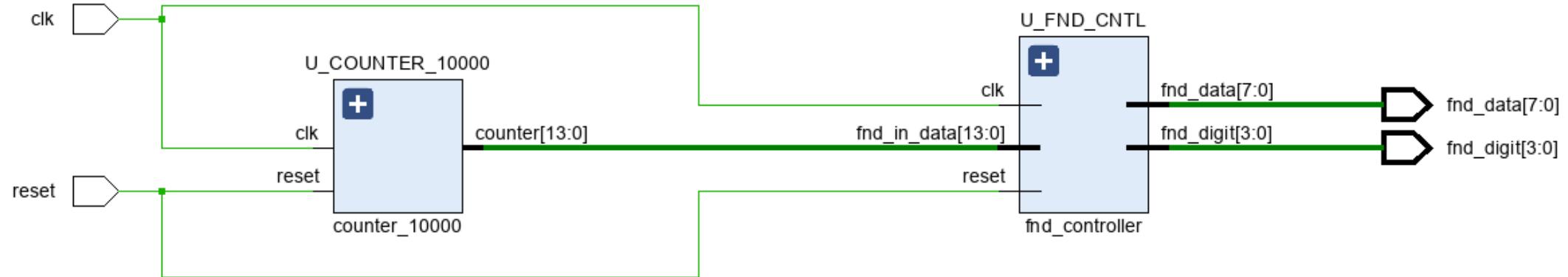
# Counter\_10000

## RTL schematic & Simulation

### 분석

김민기

# RTL schematic

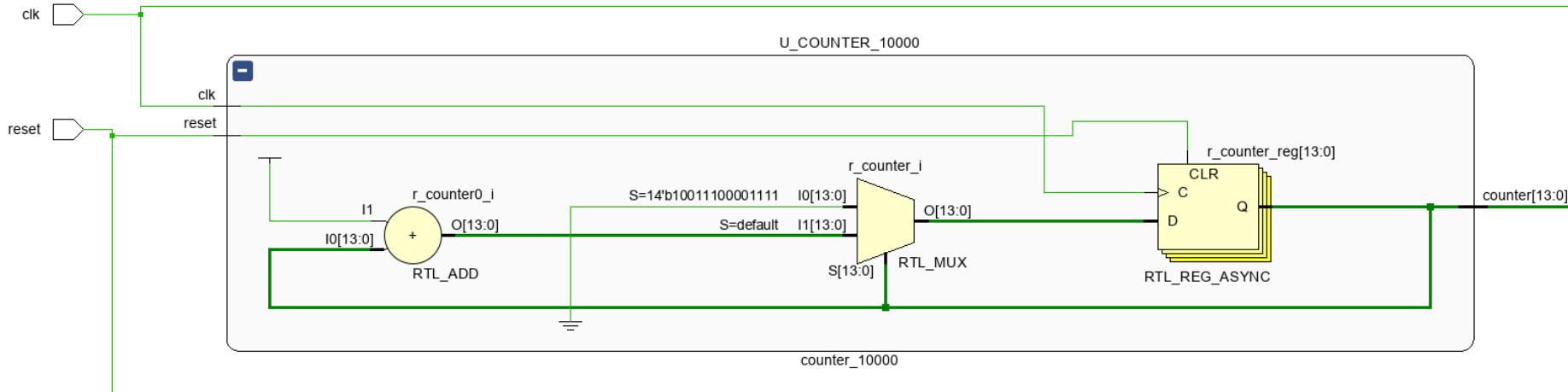


counter\_10000과 rnd\_controller의 전체 schematic

Input: clk, reset

Output: rnd\_data[7:0], rnd\_digit[3:0]

# RTL schematic



counter\_10000의 schematic

Input: clk, reset

Output: counter[13:0]

clk가 한 번씩 발생할 때마다 r\_counter의 값을 1씩 증가시키도록 설계하였다.

추가로, 옆의 코드는 r\_counter의 값이 9999가 되면 다시 0으로 초기화하는 코드이다.

```
module counter_10000 (
    input      clk,
    input      reset,
    output [13:0] counter
);

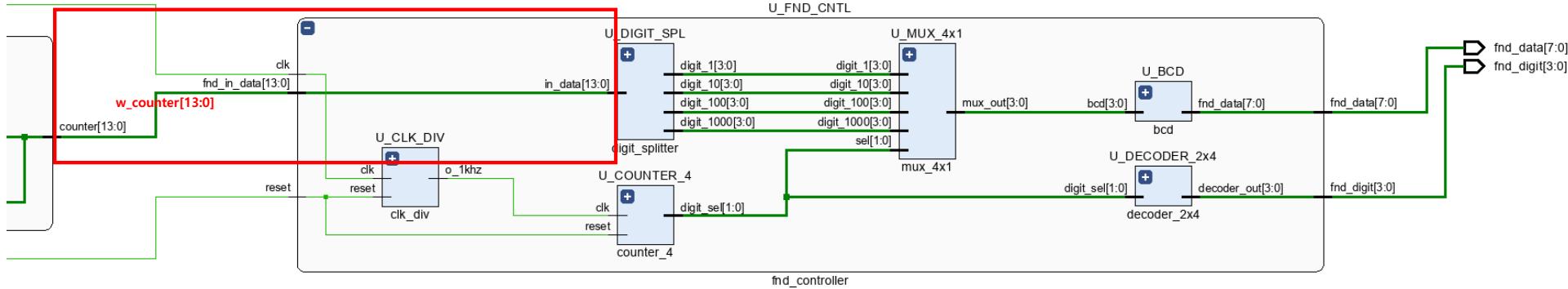
reg [13:0] r_counter;

assign counter = r_counter;

always @(posedge clk, posedge reset) begin
    if (reset) begin
        r_counter <= 14'd0;
    end else begin
        r_counter <= r_counter + 1;
        if (r_counter == 9999) begin
            r_counter <= 14'd0;
        end
    end
end

endmodule
```

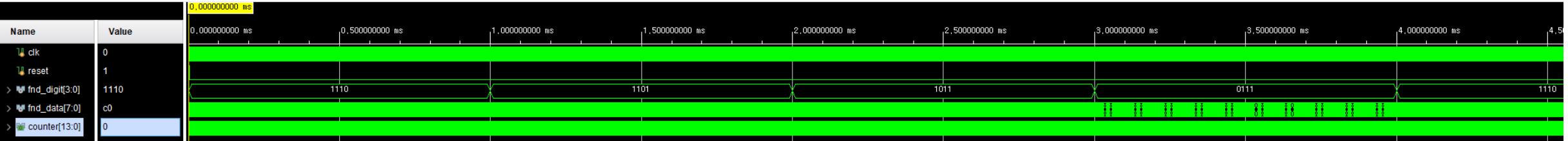
# RTL schematic



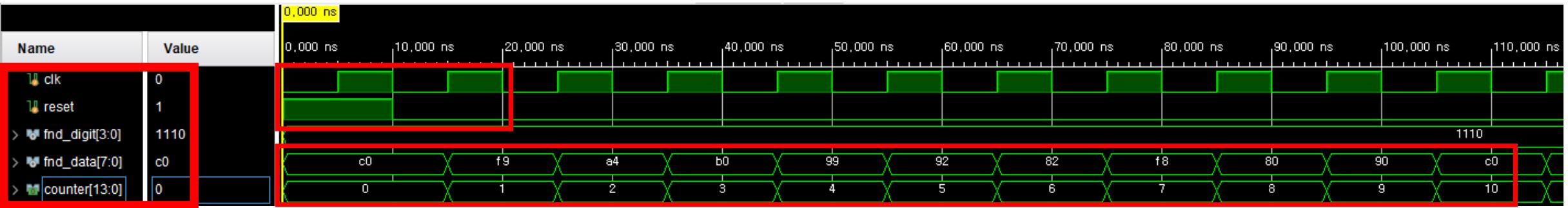
위의 사진은 `counter_10000`과 `fnd_controller` module 서로 잘 연결되었는지 확인하기 위한 사진이다.

강조된 부분을 보면 `counter_10000`에서 나온 14bit counter 출력이 `w_counter[13:0]` wire를 통해 `fnd_controller`의 입력 `in_data[13:0]`와 연결되도록 설계하였다.

# Simulation (Waveform)



위의 사진은 전체 simulation의 사진이다.



위의 사진은 simulation 분석을 위해 파형을 0ns부터 110ns까지 확대한 사진이다.

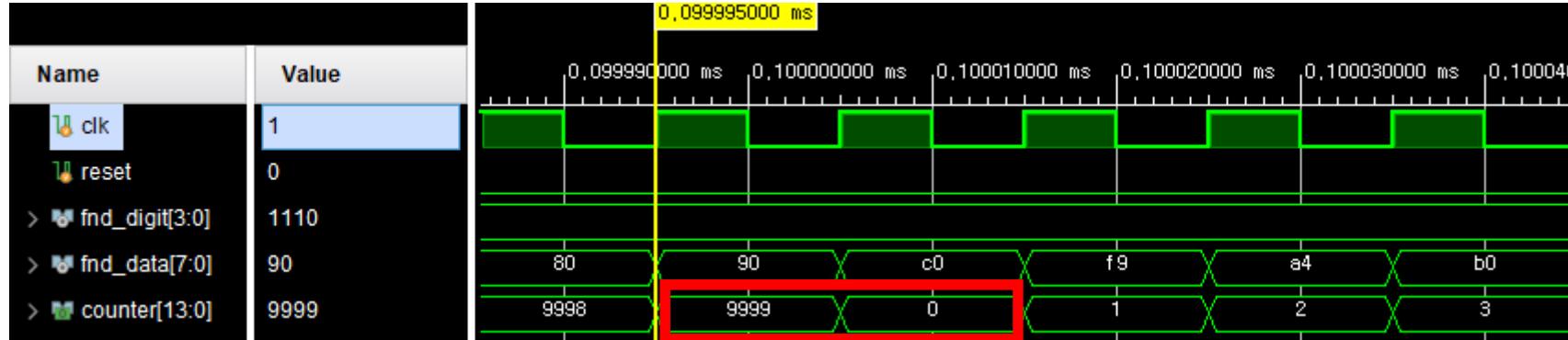
사진 설명: 0ns에서 clk는 0에서 시작해 5ns마다 자기 자신의 상태를 반전시킨다.  
reset은 0ns에서 1로 유지되다가 10ns delay 이후에는 계속 0의 상태를 가진다.

counter는 clk이 발생할 때마다 값이 1이 증가하는 모습으로 의도한 대로 파형이 나오고 있다.  
fnd\_data 또한, counter 숫자에 맞춰 그에 해당하는 fnd\_data 값을 가지는 모습을 볼 수 있다.

```
always #5 clk = ~clk;  
  
initial begin  
    #0;  
    clk    = 0;  
    reset  = 1;  
    #10;  
    reset  = 0;
```

참고 코드

# Simulation (Waveform)



counter값이 계속 증가하여 9999까지 오고 그 이후 clk에서는 다시 0으로 초기화 된다.

```
module counter_10000 (
    input      clk,
    input      reset,
    output [13:0] counter
);

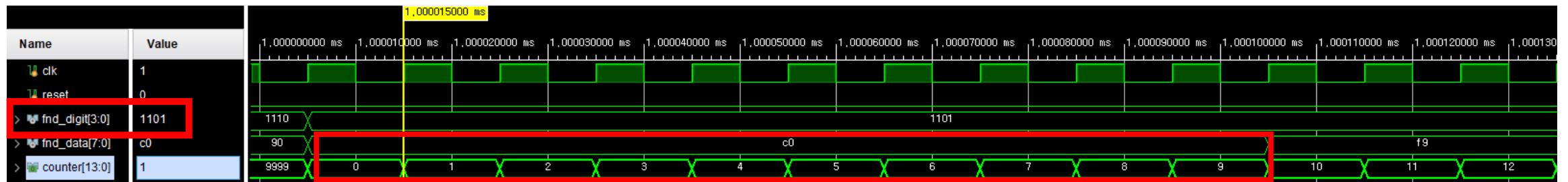
reg [13:0] r_counter;

assign counter = r_counter;

always @(posedge clk, posedge reset) begin
    if (reset) begin
        r_counter <= 14'd0;
    end
    else begin
        r_counter <= r_counter + 1;
        if (r_counter == 9999) begin
            r_counter <= 14'd0;
        end
    end
end
endmodule
```



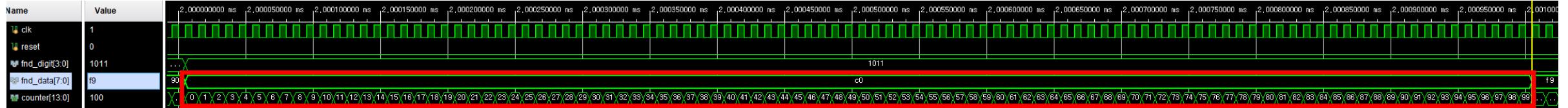
fnd\_data가 counter와 함께 값이 바뀌는 이유는 fnd\_digit의 값이 1110이고, data의 값은 1의 자리만 해당하기 때문이다.



fnd\_digit이 1101인 경우를 보자.

이 경우에는 fnd\_data가 10의 자리에만 해당하는 값을 가지기 때문에 counter가 9씩 증가할 때마다 fnd\_data의 값이 바뀌는 모습을 보인다.

# Simulation (Waveform)

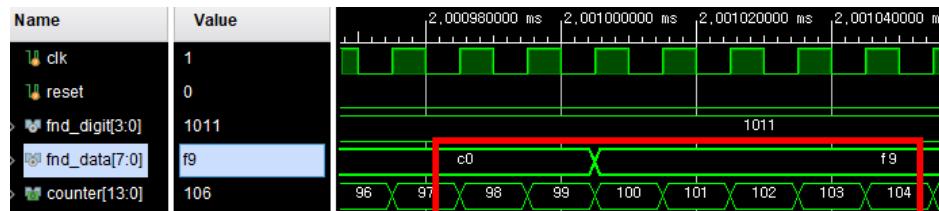


이와 마찬가지로 fnd\_digit이 1011인 경우를 보면

fnd\_data가 100의 자리에만 해당하는 값을 가지기 때문에 counter가 99씩 증가할 때마다 fnd\_data의 값이 바뀌는 모습을 보인다.

counter의 값이 0부터 99까지는 fnd\_data의 값이 c0이다. (100의 자리는 모두 0이기 때문)

counter의 값이 100이 넘어가는 순간 fnd\_data의 값이 숫자 1을 표현하는 f9로 바뀐다. (100의 자리가 1이기 때문)



counter의 값이 100을 넘어가는 순간 fnd\_data의 값이 바뀐다.

확대 사진