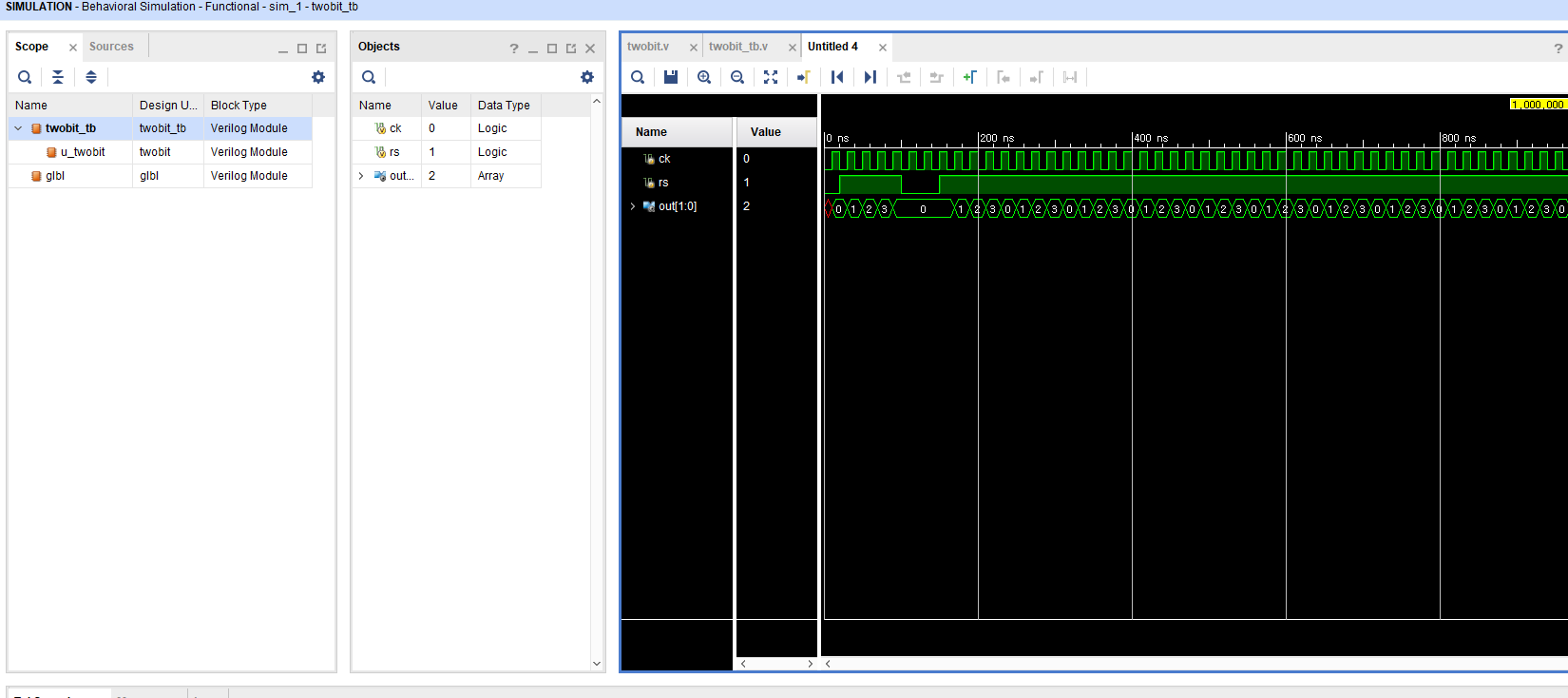
12주차 결과보고서

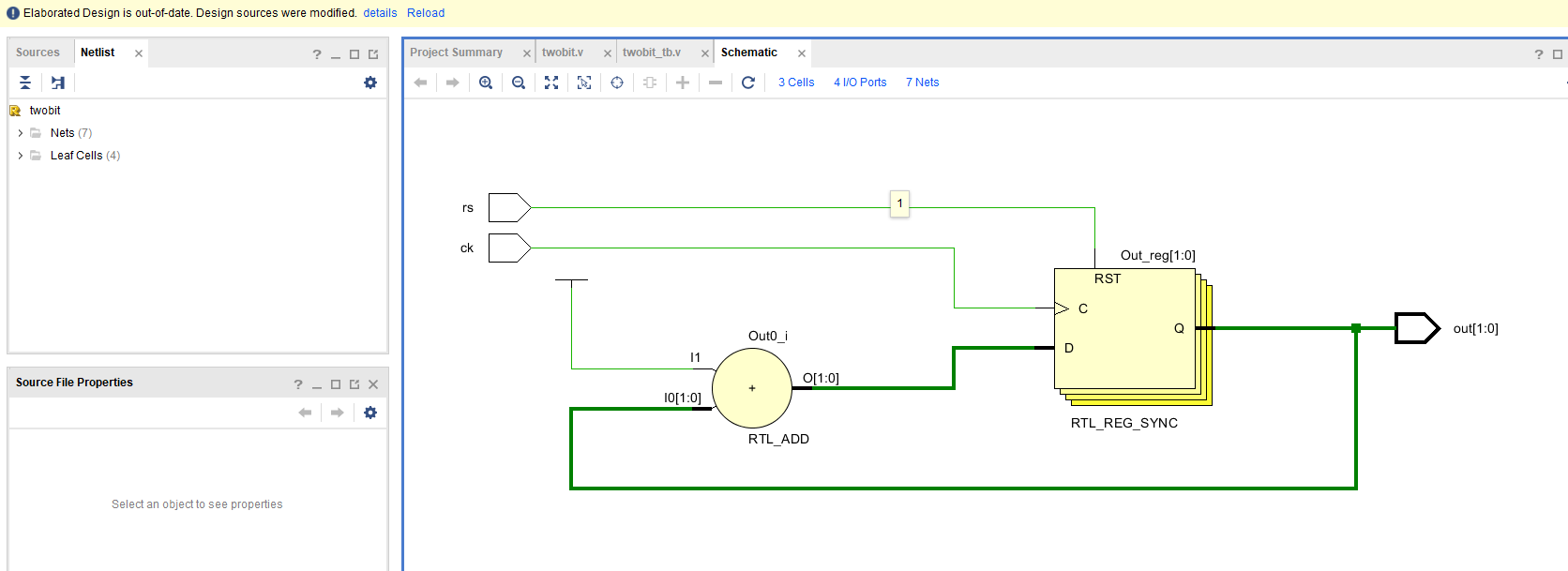
전공: 컴퓨터공학과 학년: 2학년 학번: 20221549 이름: 김효림

**1.**

.................



2-bit counter의 simulation 결과



2-bit counter의 schematic 결과

|  |  |
| --- | --- |
| Source code | Simulation code |
| module twobit(  ck, rs, out  );  output [1:0] out;  reg [1:0] out;  input ck;  input rs;  always @ (posedge(ck)) begin  if(!rs)  out <= 0;  else  out <= out +1;  end  endmodule | module twobit\_tb;  reg ck, rs;  wire [1:0] out;  twobit u\_twobit(  .ck(ck),  .rs(rs),  .out(out)  );  always #10 ck = ~ck;  initial begin  ck <= 1'b0;  rs <= 1'b0;  #20 rs <= 1'b1;  #80 rs <= 1'b0;  #50 rs <= 1'b1;  end  endmodule |

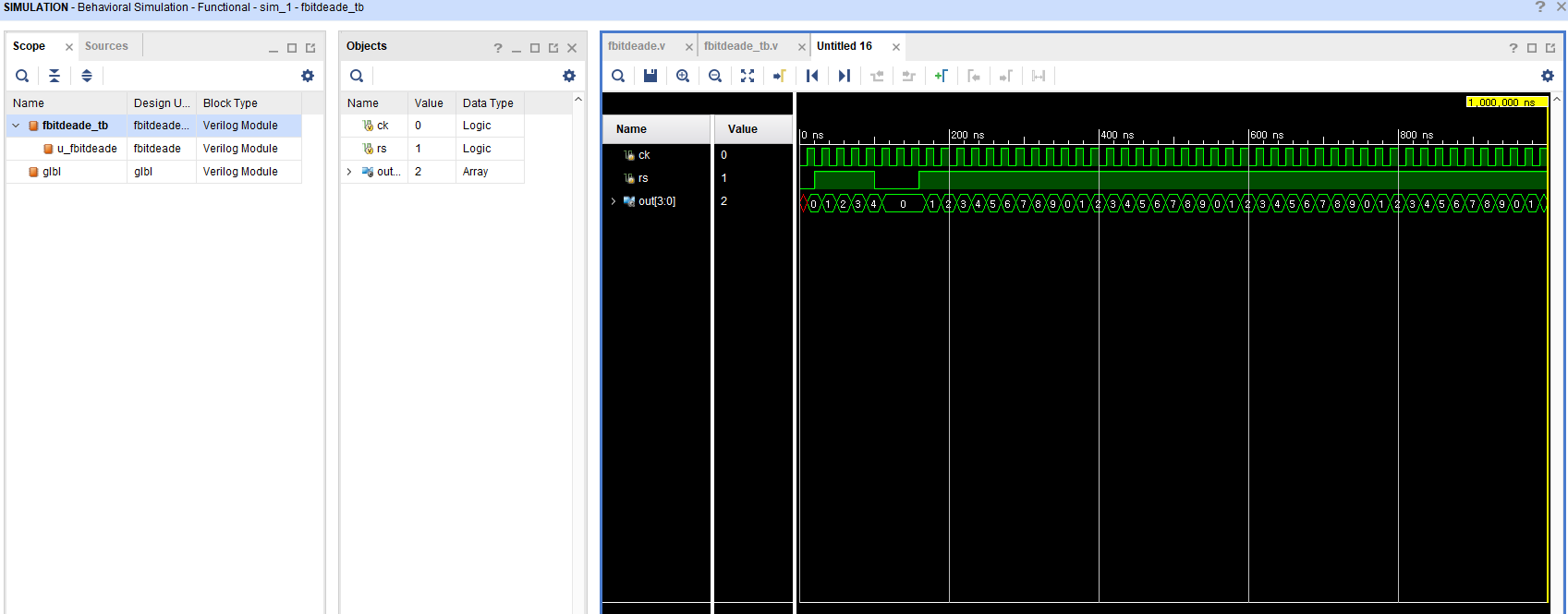
2-bit counter의 Verilog source code

2비트 계수기는 0부터 3까지의 수를 셀 수 있는 계수기로, 시뮬레이션 결과에서 확인할 수 있듯이 0부터 3까지의 값이 순차적으로 나타난다. Reset 버튼이 눌린 상태라면 out의 값을 1씩 증가시키고, reset 버튼을 눌렀다 뗐을 때(reset 신호가 활성화되었을 때) out 값을 0으로 만들어 준다.

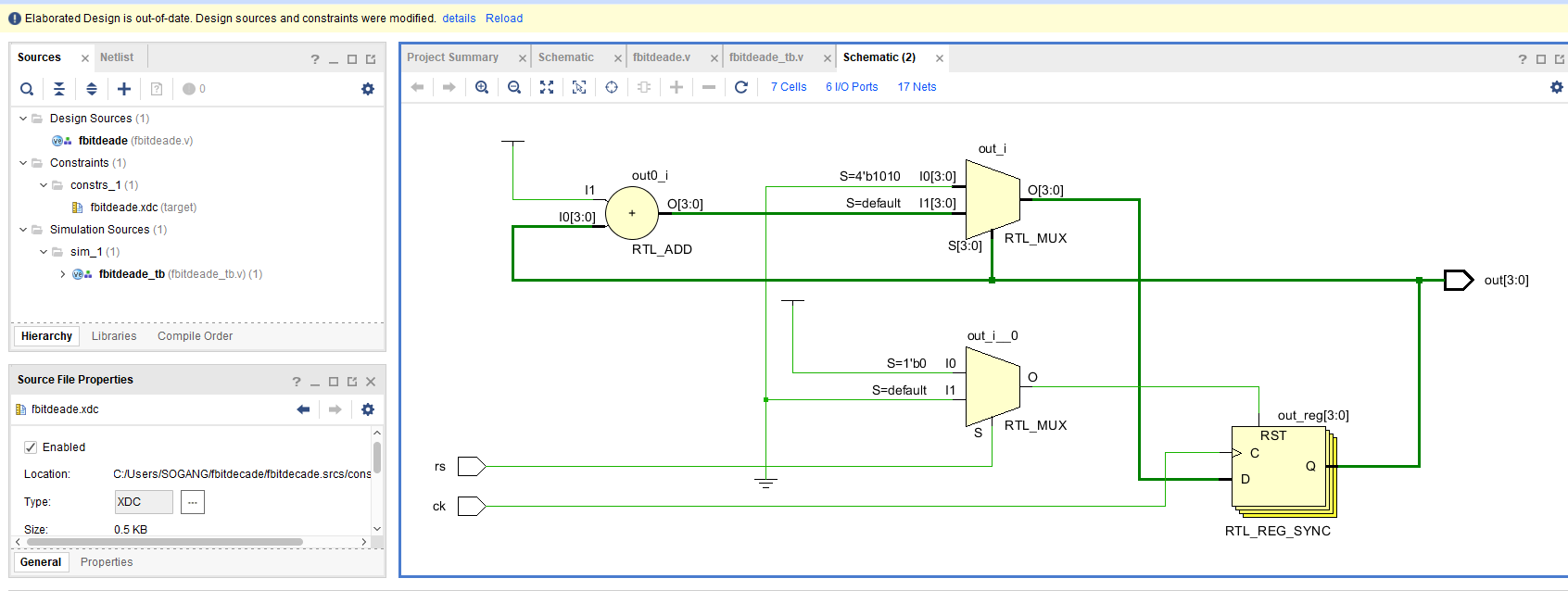
................

**2.**

.......................



4-bit decade counter의 simulation 결과



4-bit decade counter의 schematic 결과

|  |  |
| --- | --- |
| Source code | Simulation code |
| module fbitdeade(  ck,rs,out  );    output [3:0] out;  reg [3:0] out;    input ck;  input rs;      always @( posedge ck) begin  if(!rs)  out <= 0;  else if (out == 4'b1001) begin  out <=0;  end  else  out <= out+1;  end      endmodule | module fbitdeade\_tb;  reg ck, rs;  wire [3:0] out;  fbitdeade u\_fbitdeade(  .ck(ck),  .rs(rs),  .out(out)  );  always #10 ck = ~ck;  initial begin  ck <= 1'b0;  rs <= 1'b0;  #20 rs <= 1'b1;  #80 rs <= 1'b0;  #60 rs <= 1'b1;  end  endmodule |

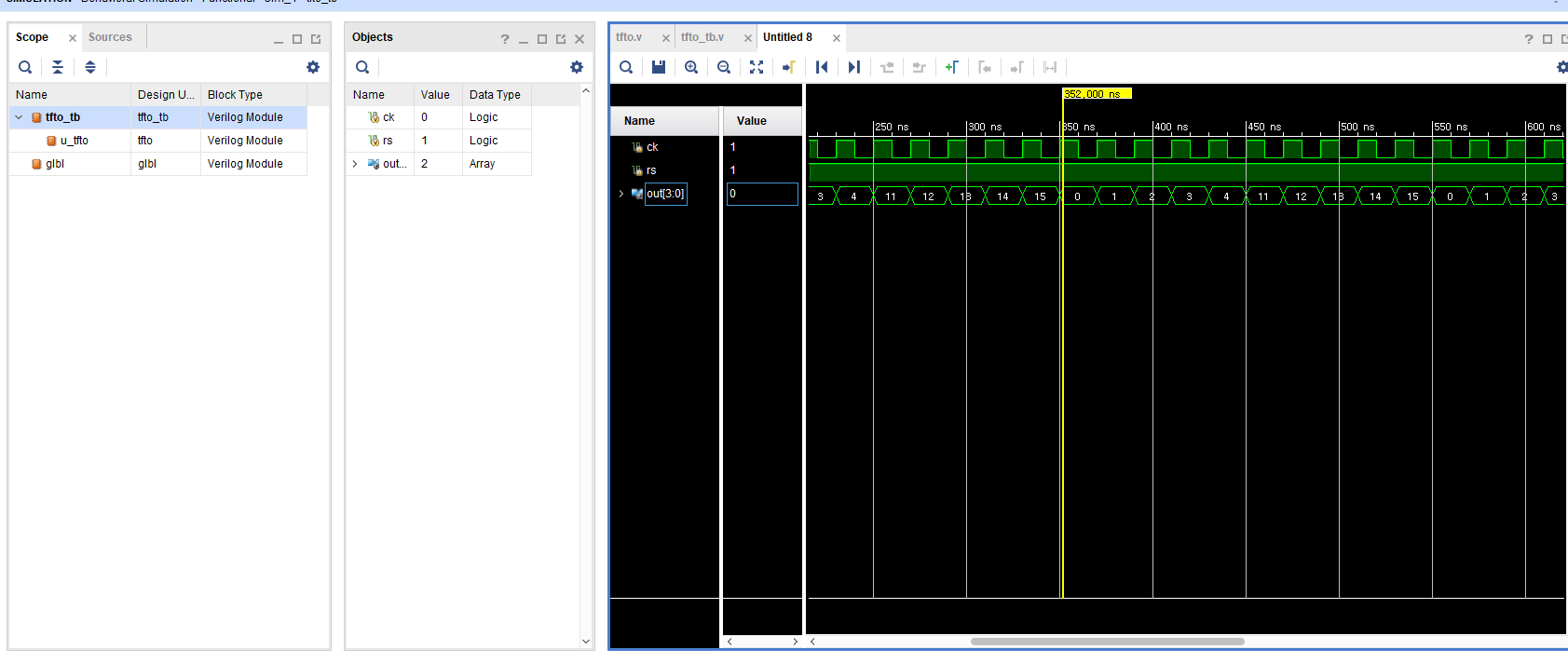
4 bit decade counter의 Verilog source code

4 비트 십진 계수기는 BCD 숫자를 세주는 계수기로, 0부터 9까지의 숫자를 셀 수 있다. 시뮬레이션 결과를 통해 그것을 확인할 수 있다. 소스 코드를 보면 알 수 있듯이, reset 버튼이 활성화되지 않고 1001(10진수로는 9)가 됐을 때 계수기를 0000으로 초기화해 주고, 그 전까지는 다음 숫자를 셀 수 있도록 설계되어 있다. 따라서 클럭 버튼과 리셋 버튼이 함께 눌려 있을 때는 수를 세는 동작을 수행하고, 리셋 버튼이 떼어지는 순간(활성화되는 순간) 클락 버튼을 떼면 초기화된다.

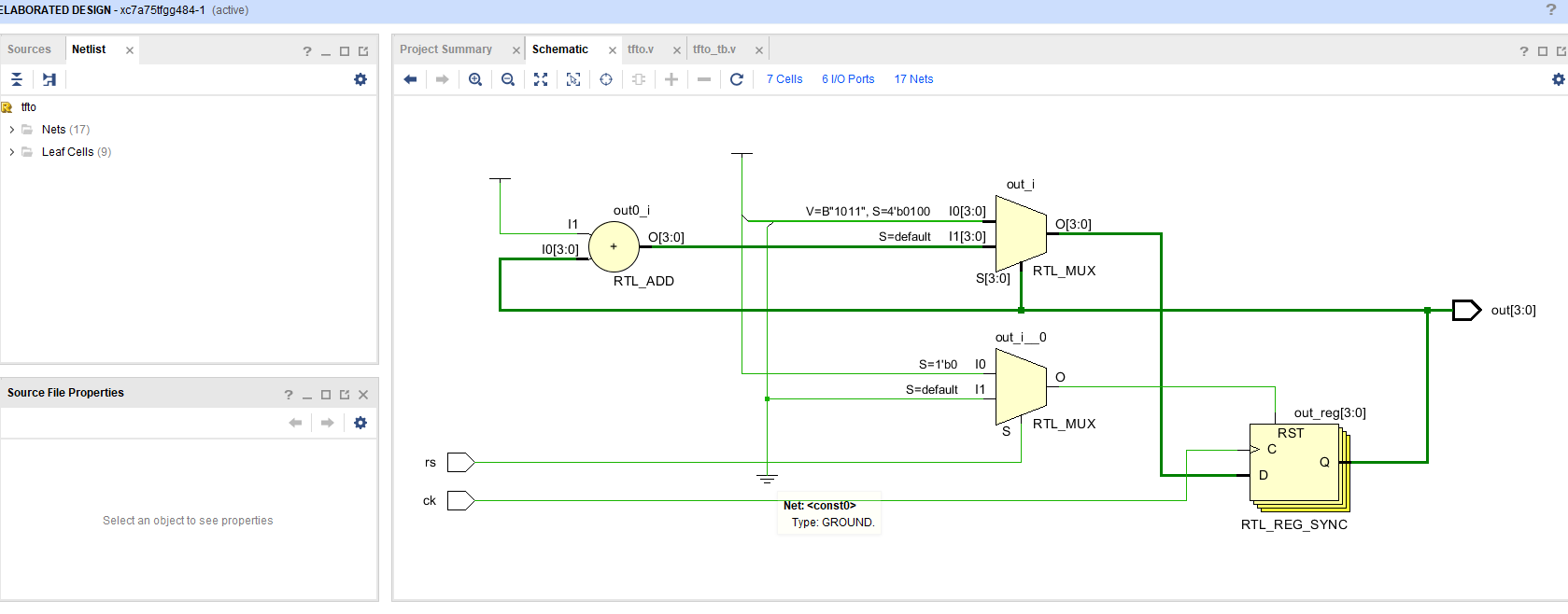
.........................

**3.**

.......................



4-bit 2421 decade counter의 simulation 결과



4-bit 2421 decade counter의 schematic 결과

|  |  |
| --- | --- |
| Source code | Simulation code |
| module tfto(  ck,rs,out  );    output [3:0] out;  reg [3:0] out;    input ck;  input rs;    always @ (posedge(ck)) begin  if(!rs)  out <= 0;  else if (out == 4'b0100)  out <= 4'b1011;  else  out <= out +1;  end    endmodule | module tfto\_tb;  reg ck, rs;  wire [3:0] out;  tfto u\_tfto(  .ck(ck),  .rs(rs),  .out(out)  );  always #10 ck = ~ck;  initial begin  ck <= 1'b0;  rs <= 1'b0;  #20 rs <= 1'b1;  #80 rs <= 1'b0;  #50 rs <= 1'b1;  end  endmodule |

4-bit 2421 decade counter의 Verilog source code

4 비트 2421 BCD 코드는 4비트 수의 최상위 비트를 8이 아닌 2로 해석하는 10진수 표현법이다. 따라서 source code를 보면 확인할 수 있듯이, 0부터 4까지는 다음 수를 세다가 out 값에 4(0100)가 저장되면 각각의 비트가 반전된 것(1011)이 out 값에 저장된다. 그 후로는 또 reset 버튼이 활성화되기 전까지 숫자를 센다.

.........................

**4.**

.......................

각각의 기능에 따라 Verilog 코드를 구현하여, 각 counter의 기능이 제대로 작동하는 것을 simulation 결과를 통해 확인할 수 있었다. 처음으로 if 문을 사용하여 코드를 구현하는 법을 알 수 있었고, 그 덕분에 간단하게 구현할 수 있었다.

.........................

**5.**

.......................

실습 시간에 다루었던 카운터 외에도, modulo N counter라는 것도 존재한다. 이것은 외부에서 신호가 전달될 때마다 숫자의 크기를 하나씩 증가시키는 계수기로, 표현되는 수가 n-1에 도달하면 다음의 외부 신호로 인하여 계수기의 값은 0이 된다. 예를 들어, mod-6 counter는 상태의 수가 6개인 카운터로, 0부터 5까지의 수를 카운트하여 나타내는 상태들을 순차적으로 발생시키고 다시 그 상태 시퀀스를 반복하게 된다. 이는 디지털 시계에서 분을 가리키는 두 자릿수 중에서 앞의 숫자들을 결정하는 데 사용된다. 동기식 mod-6 counter는 입력 펄스가 들어올 때마다 상태 값이 000부터 101까지 순서대로 증가하며, 그 다음 펄스에서 다시 000으로 되돌아가서 이를 반복하게 된다.

.........................