**컴퓨터공학실험2**

11주차 결과보고서

컴퓨터공학과

20191621 이민영

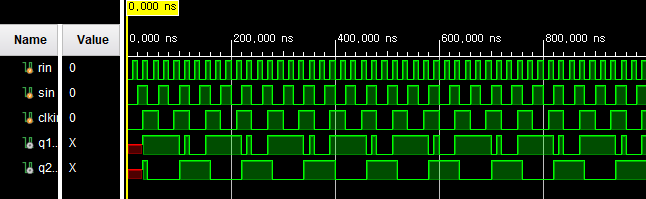
1. RS Flip-Flop의 결과 및 Simulation 과정에 대해서 설명하시오.

(Verilog source, 출력 예시, 과정 상세히 적을 것)

|  |
| --- |
| RS.v |
| `timescale 1ns / 1ps  module RS(  input inr, ins, inclk,  output outq1,outq2  );  assign outq1 = ~(~(inr&inclk)&outq2);  assign outq2 = ~(~(ins&inclk)&outq1);  endmodule |

|  |
| --- |
| RS\_tb.v |
| `timescale 1ns / 1ps  module RS\_tb;  reg rin,sin,clkin;  wire q1out,q2out;  RS  u\_RS(  .inr(rin),  .ins(sin),  .inclk(clkin),  .outq1(q1out),  .outq2(q2out)  );  initial  begin  rin=1'b0;  sin=1'b0;  clkin=1'b0;  end  always@(rin or sin or clkin)begin  rin<=#10 ~rin;  sin<=#20 ~sin;  clkin<=#30 ~clkin;  end  endmodule |

|  |
| --- |
| RS\_c.xdc |
| set\_property IOSTANDARD LVCMOS18 [get\_ports inr]  set\_property IOSTANDARD LVCMOS18 [get\_ports ins]  set\_property IOSTANDARD LVCMOS18 [get\_ports inclk]  set\_property IOSTANDARD LVCMOS18 [get\_ports outq1]  set\_property IOSTANDARD LVCMOS18 [get\_ports outq2]  set\_property PACKAGE\_PIN J4 [get\_ports inr]  set\_property PACKAGE\_PIN L3 [get\_ports ins]  set\_property PACKAGE\_PIN K3 [get\_ports inclk]  set\_property PACKAGE\_PIN M2 [get\_ports outq1]  set\_property PACKAGE\_PIN K6 [get\_ports outq2] |



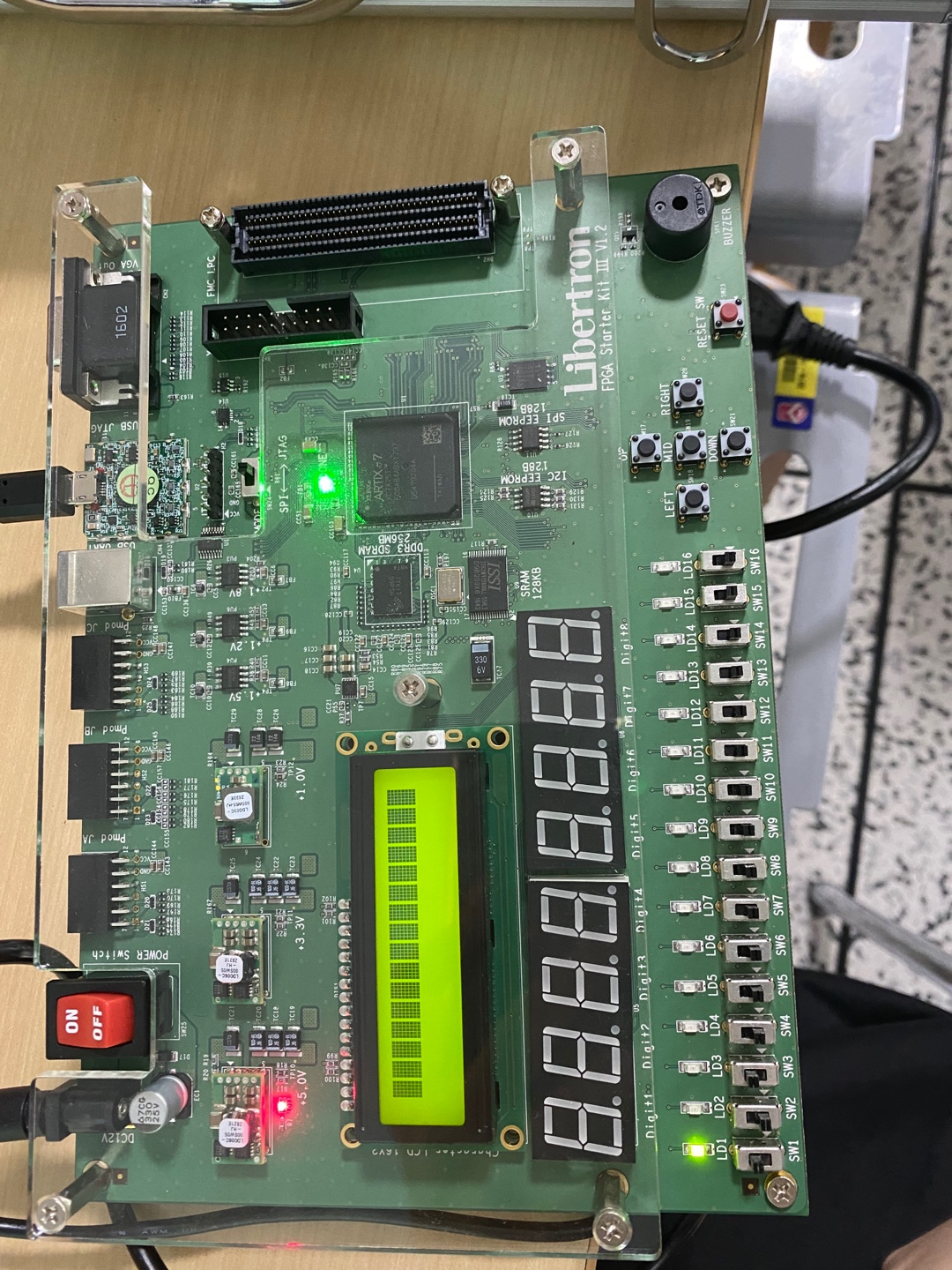
RS 플립플롭은 2개의 입력 R, S가 들어오고, 출력으로 Q, Q’이 나온다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| q SR | 00 | 01 | 11 | 10 |
| 0 | 0 | 0 | X | 1 |
| 1 | 1 | 0 | X | 1 |

<K-map>

K맵을 통해서=S+R’ 임을 알 수 있고 이를 이용해서 Verilog 코드를 작성해볼 수 있다.CLK가 1이고 S=1, R=0일 때 Q=1, Q’=0이 되어 Set인 상태가 되고, CLK가 1이고 S=0, R=1일 때 Q=0, Q’=1인 Reset 상태가 된다. S와 R 모두 0인 경우에 Q는 이전 값을 그대로 가지게 되고 S가 0, R이 1일 때는 Q가 0으로 설정되고 S가 1, R이 0일 때는 Q가 1로 설정된다. RS 플립플롭에서 S와 R 모두 1인 입력은 가능하지 않다. 이는 뒤에 나오는 JK플립플롭을 통해서 보완할 수 있다. Q = ~(~(R&CLK)&Q’) 이고 Q’ = ~(~(S&CLK)&Q)이다. 이 때 출력값인 Q와 Q’이 들어가있기 때문에 첫번째 출력값은 출력되지 않는 것을 알 수 있다.

Tb 코드에서는 입력값에 변화를 주었고, xdc 코드에서는 R, S, CLK 차례로 입력값을 조절할 수 있도록 FPGA의 J4, L3, K3, M2, K6에 연결하였고, 첫번째 두번째 불빛을 통해서 Q와 Q’의 값을 확인할 수 있도록 하였다.



<왼쪽부터 차례로 S, R, CLK일 때, S가 1, R이 0, CLR가 1 이므로 Set상태로 Q에 불이 들어와있다.>



상태도를 통해서도 동일한 결과를 얻는다는 것을 알 수 있었다.

1. D Flip-Flop의 결과 및 Simulation 과정에 대해서 설명하시오.

(Verilog source, 출력 예시, 과정 상세히 적을 것)

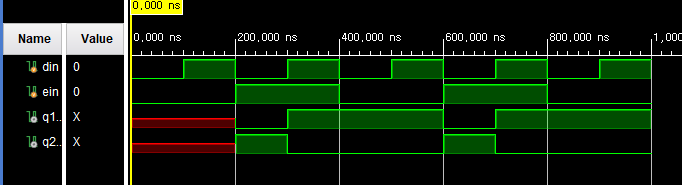
|  |
| --- |
| D.v |
| `timescale 1ns / 1ps  module D(  input ind, ine,  output outq1,outq2  );  assign outq1 = ~((~ind&ine)|outq2);  assign outq2 = ~((ind&ine)|outq1);  endmodule |

|  |
| --- |
| D\_tb.v |
| `timescale 1ns / 1ps  module D\_tb;  reg din, ein;  wire q1out,q2out;  D  u\_D(  .ind(din),  .ine(ein),  .outq1(q1out),  .outq2(q2out)  );  initial  begin  din = 1'b0;  ein = 1'b0;  end  always@(din or ein)begin  din<=#100 ~din;  ein<=#200 ~ein;  end  endmodule |

|  |
| --- |
| D\_c.xdc |
| set\_property IOSTANDARD LVCMOS18 [get\_ports {ind}]  set\_property IOSTANDARD LVCMOS18 [get\_ports {ine}]  set\_property IOSTANDARD LVCMOS18 [get\_ports {outq1}]  set\_property IOSTANDARD LVCMOS18 [get\_ports {outq2}]  set\_property PACKAGE\_PIN J4 [get\_ports {ind}]  set\_property PACKAGE\_PIN L3 [get\_ports {ine}]  set\_property PACKAGE\_PIN F15 [get\_ports {outq1}]  set\_property PACKAGE\_PIN F13 [get\_ports {outq2}]  set\_property ALLOW\_COMBINATORIAL\_LOOPS TRUE [get\_nets {outq1}]  set\_property ALLOW\_COMBINATORIAL\_LOOPS TRUE [get\_nets {outq1\_OBUF}]  set\_property ALLOW\_COMBINATORIAL\_LOOPS TRUE [get\_nets {outq2}]  set\_property ALLOW\_COMBINATORIAL\_LOOPS TRUE [get\_nets {outq2\_OBUF}] |

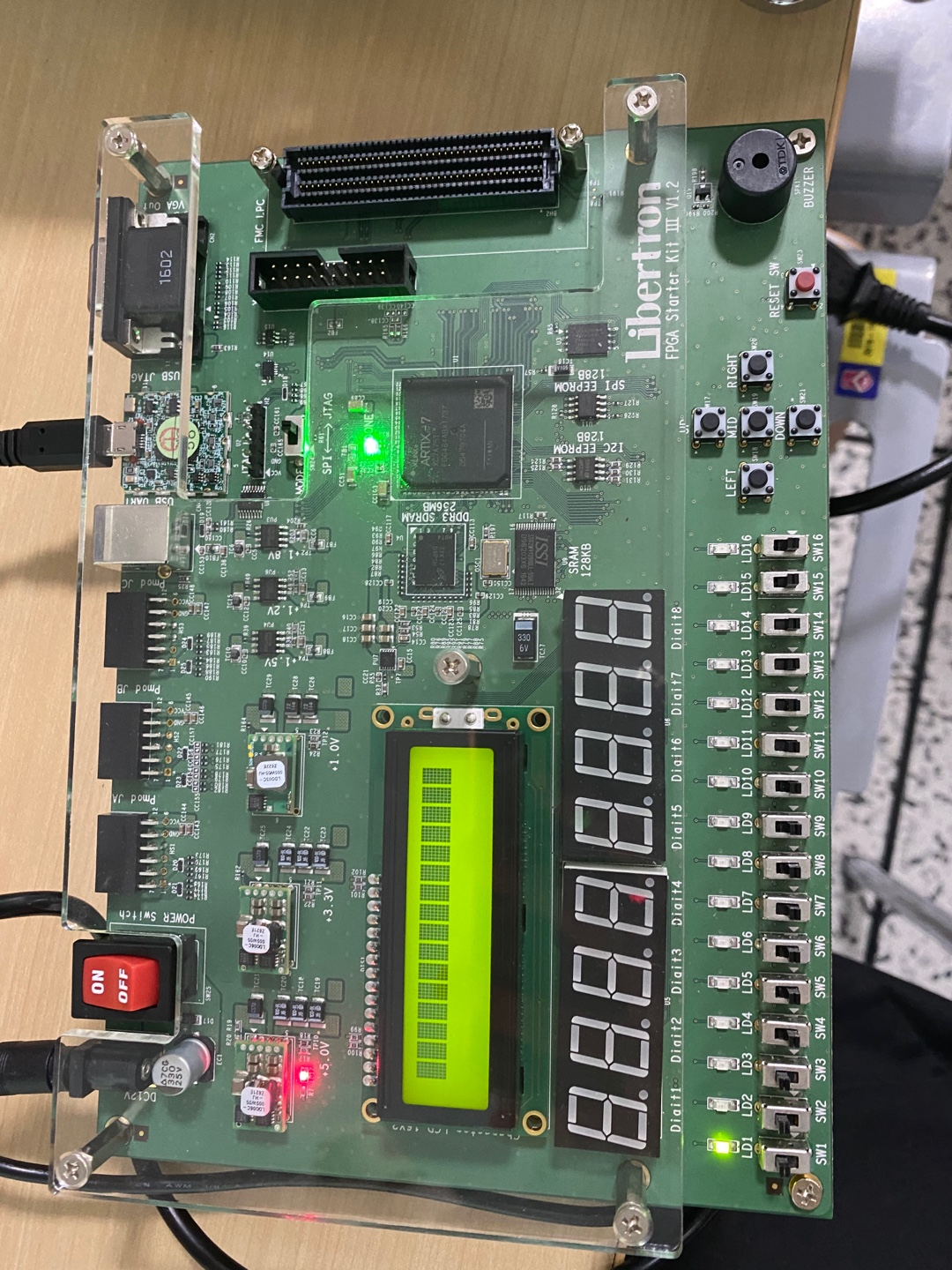
D 플립플롭은 입력값으로 D와 CLK가 들어온다. Q = ~((~D&CLK)|Q’)이고, Q’ = ~((D&CLK)|Q) 이다. 이를 D.v 파일에 저장해두었고, tb 파일을 통해서 입력값에 변화를 주었다. 첫 출력값의 경우 Q와 Q’이 정해지지 않았기 때문에 결과값이 나오지 않는 것을 볼 수 있다.

xdc파일을 통해서 D는 FPGA의 첫번째 J4 입력단자에, CLK는 두번째 L3 입력 단자에 연결해두었고, 출력값인 Q와 Q’는 첫번째 두번째 불빛으로 결과값을 알아볼 수 있도록 설정해두었다.

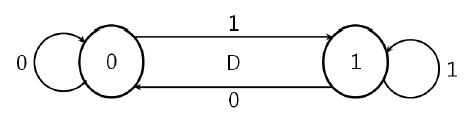


D Flip-Flop은 입력값인 D가 0이고 CLK가 0일 때 Q와 Q’에는 변화가 없고, D가 0이고 CLK가 1일 때는 Q는 0, Q’는 1, D가 1이고 CLK가 1일 때는 Q와 Q’에 변화가 없고 D가 1이고 CLK가 1일 때 Q는 1, Q’은 0이 된다.

이는 시뮬레이션을 통해서도 알 수 있으며, FPGA를 통해서도 확인할 수 있었다.



<FPGA에서 D와 CLK 모두 1일 때 Q는 1, Q’은 0이 된다는 것을 알 수 있다.>



상태도를 통해서도 D 플립플롭의 결과를 확인해볼 수 있었다.

1. JK Flip-Flop

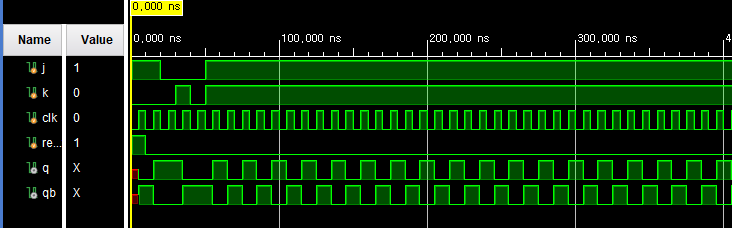
의 결과 및 Simulation 과정에 대해서 설명하시오.

(Verilog source, 출력 예시, 과정 상세히 적을 것)

|  |
| --- |
| JK.v |
| `timescale 1ns / 1ps  module JK(q,qb,j,k,clk,reset);  output q,qb;  input j,k,clk,reset;  reg q;  assign qb=~q;  always @(posedge clk) begin      if(reset)          q<=0;      else if (k==0 && j==0)          q<=q;      else if (k==1 && j==0)          q<=0;      else if (k==0 && j==1)          q<=1;      else          q<=~q;  end  endmodule |

|  |
| --- |
| JK\_tb.v |
| ‘timescale 1ns / 1ps  module jkfffD\_tb();  reg j,k,clk,reset;  wire q,qb;  jkffd con(q, qb, j, k, clk, reset);  initial begin  clk = 0; reset = 1; j = 1; k = 0;  reset = #10 0;  j = #10 0;  k = #10 1;  k = #10 0;  j = #10 1; k = 1;  end;  always clk = #5 ~clk;  endmodule |

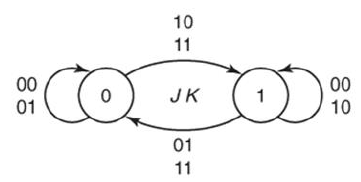
|  |
| --- |
| JK\_c.xdc |
| set\_property IOSTANDARD LVCMOS18 [get\_ports {j}]  set\_property IOSTANDARD LVCMOS18 [get\_ports {k}]  set\_property IOSTANDARD LVCMOS18 [get\_ports {clk}]  set\_property IOSTANDARD LVCMOS18 [get\_ports {reset}]  set\_property IOSTANDARD LVCMOS18 [get\_ports {q}]  set\_property IOSTANDARD LVCMOS18 [get\_ports {qb}]  set\_property PACKAGE\_PIN J4 [get\_ports {j}]  set\_property PACKAGE\_PIN L3 [get\_ports {k}]  set\_property PACKAGE\_PIN K3 [get\_ports {clk}]  set\_property PACKAGE\_PIN M2 [get\_ports {reset}]  set\_property PACKAGE\_PIN F15 [get\_ports {q}]  set\_property PACKAGE\_PIN F13 [get\_ports {qb}]  set\_property CLOCK\_DEDICATED\_ROUTE FALSE [get\_nets {clk\_IBUF}] |



JK Flip-Flop은 RS 플립-플롭에서 Set에 1, Reset 에 1이 들어왔을 때의 문제점을 보완하는 플립플롭이다. J가 0, K가 0 일때는 변화가 없고 J가 1, K가 1일 때는 Output이 이전상태의 역으로 변화된다. J가 0, K가 1일때는 Q는 0, Q’은 1이 되며, J가 1, K가 0일때는 Q는 1, Q’은 0이 된다.

이를 JK.v 코드에서는 if-else구문을 이용해서 표현할 수 있었다. J,K,CLK,RESET 순서대로 J4, L3, K3,M2 입력단자에 연결하여서 스위치를 통해서 입력을 제어할 수 있었다. 또한 Q와 Q’은 불빛을 통해서 값을 판단할 수 있도록 연결해주었으며, 이는 xdc 코드로 살펴볼 수 있다.

Simulation을 통해서도 J가 0, K가 0이거나 J가 1, K가 1일때는 값의 변화가 존재하지 않는 것을 알 수 있고, J가 0, K가 1일 때는 Q가 0, Q’이 1이 된다는 것을 알 수 있었다. J가 1, K가 0일 때는 Q가 1, Q’이 0이 된다는 것을 알 수 있다. 이를 FPGA 스위치 조절을 통해서도 확인해볼 수 있었다.



JK의 상태도를 통해서도 결과를 도출해낼 수 있었다. SR 플립플롭과 비교했을 때 J,K가 모두 1인 경우가 보완된 것을 알 수 있다.