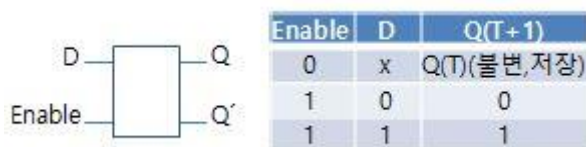


< 실습과제 3 >

2017253041_홍성우

1.

(1)



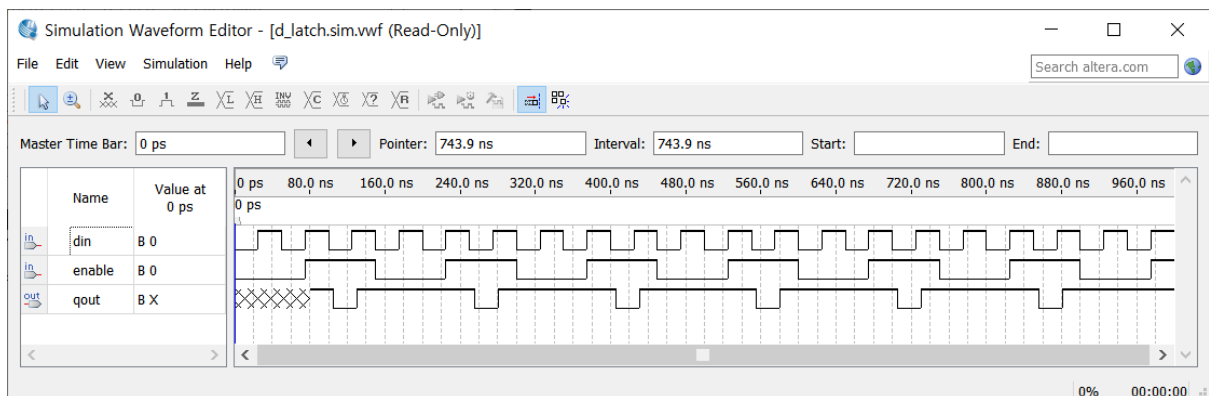
위 그림과 같이 D래치는 SR래치와 다르게 D라는 하나의 입력과 Enable이라는 입력으로 이루어졌다. Enable 값이 0일 경우에는 출력은 변하지 않고 그대로 유지되고 Enable 값이 1일 경우 D값에 따라 0 또는 1로 출력된다.

(2)

```

1 module d_latch(qout,din,enable);
2     output qout;
3     input din,enable;
4     assign qout = enable ? din : qout;
5 endmodule

```



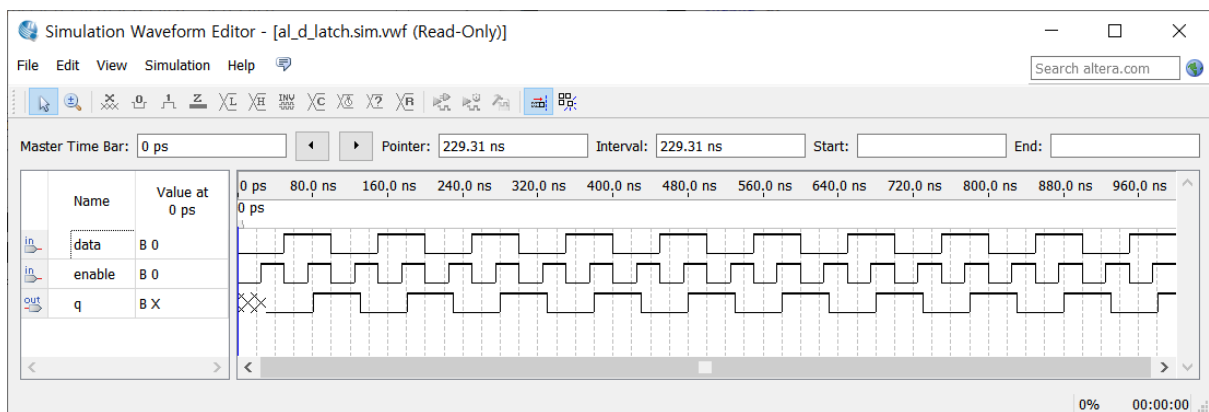
(3)

순차회로는 조합회로와 기억 소자들로 구성되었고 현재 입력과 과거의 입력 또는 출력값들도 같이 고려하여 현재의 출력값을 결정하는 논리회로이다.

2.

(1)

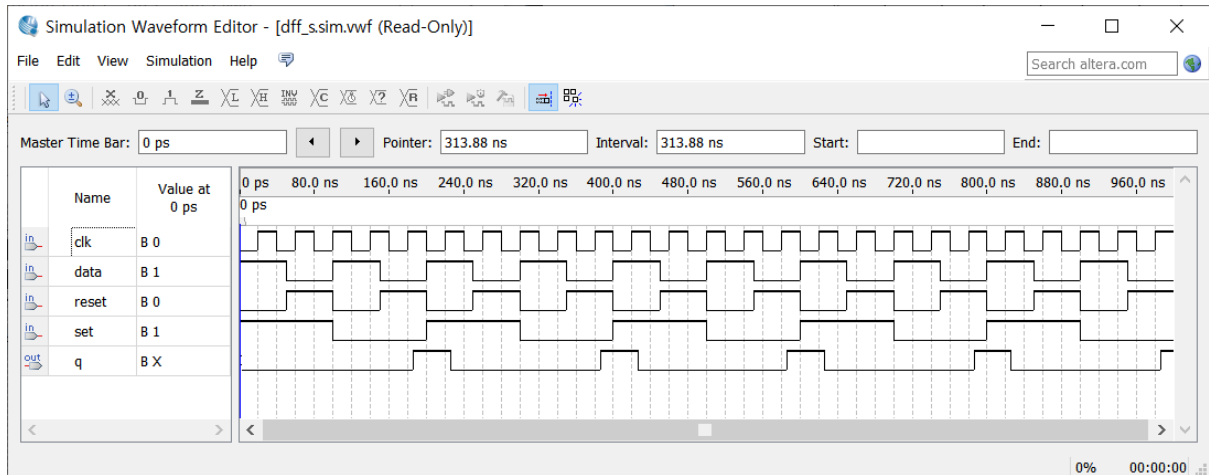
```
al_d_latch.v x Compilation Report - al_d_latch x
1 module al_d_latch(q,enable,data);
2     output q;
3     input enable,data;
4     reg q;
5     always @(enable or data) begin
6         if(enable) q = data;
7     end
8 endmodule
```



3.

(1)

```
dff_s.v x Compilation Report - dff_s x
1 module dff_s(q,data,set,reset,clk);
2     output q;
3     input data,set,reset,clk;
4     reg q;
5
6     always @(posedge clk) begin
7         if(reset==0) q=0;
8         else if(set==0) q=1;
9         else q=data;
10    end
11 endmodule
12
```

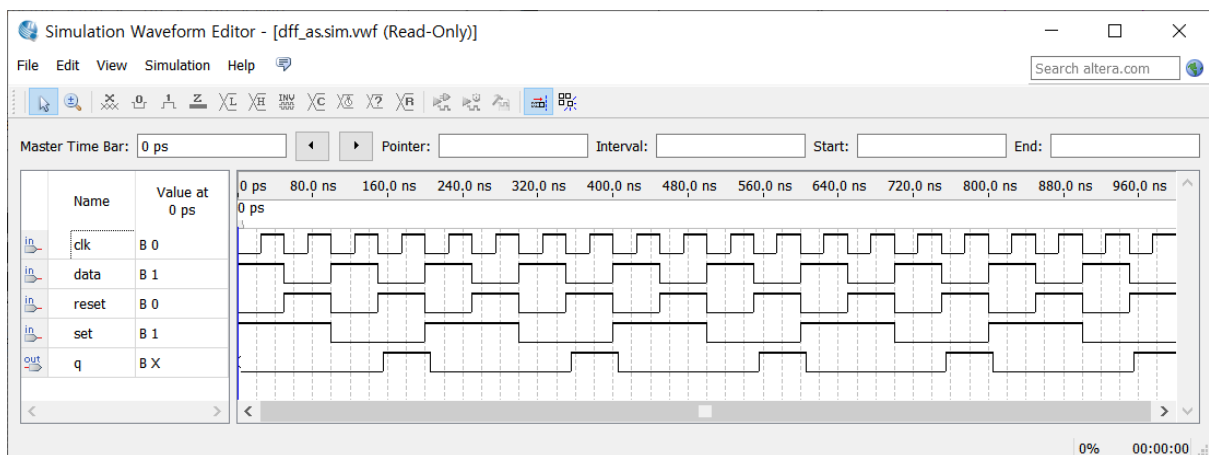


(2)

```

1 module dff_as(q,data,set,reset,clk);
2     output q;
3     input data,set,reset,clk;
4     reg q;
5
6     always @(negedge set or negedge reset or posedge clk)
7     if(reset==0) q=0;
8     else if(set==0) q=1;
9     else q=data;
10    end
11 endmodule
12

```



(3)

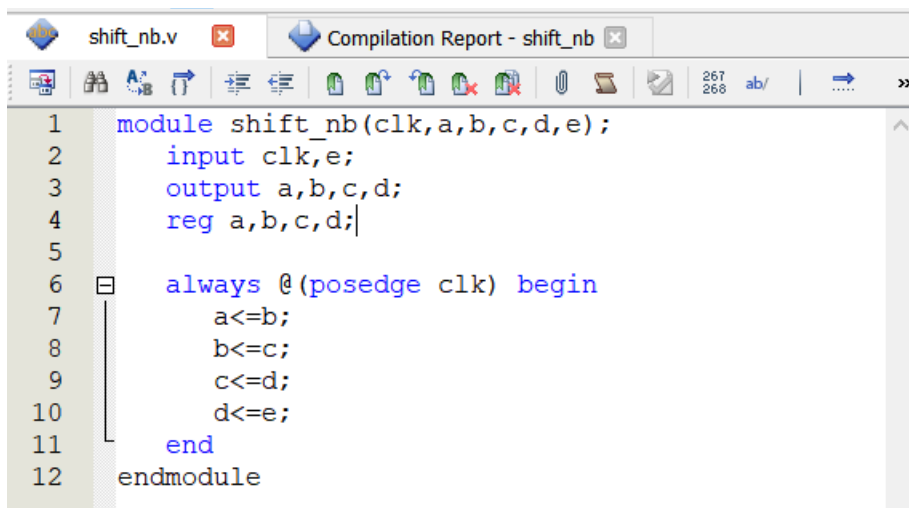
(1)번은 clk이 있을때만 동작하지만 (2)번은 clk이 없을때도 동작한다.

4.

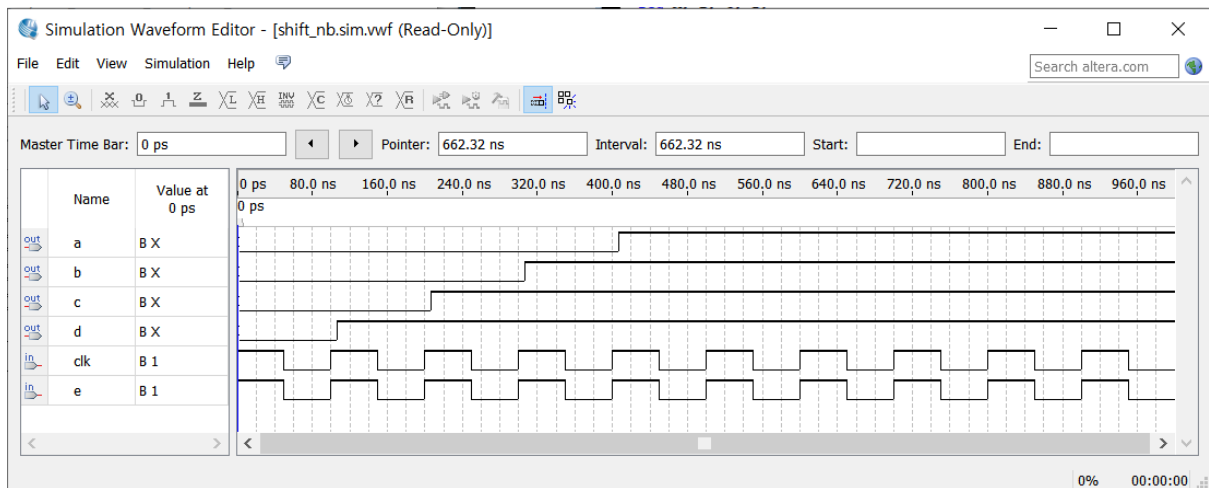
(1)

blocking 할당문은 블록 내의 할당문을 순서대로 수행하여 할당문의 순서에 영향을 받을 수 있지만, nonblocking 할당문은 블록 내의 할당문의 수식들을 먼저 계산한 후에 LHS변수 값을 갱신해서 할당문의 순서에 영향을 받지 않는다.

(2)

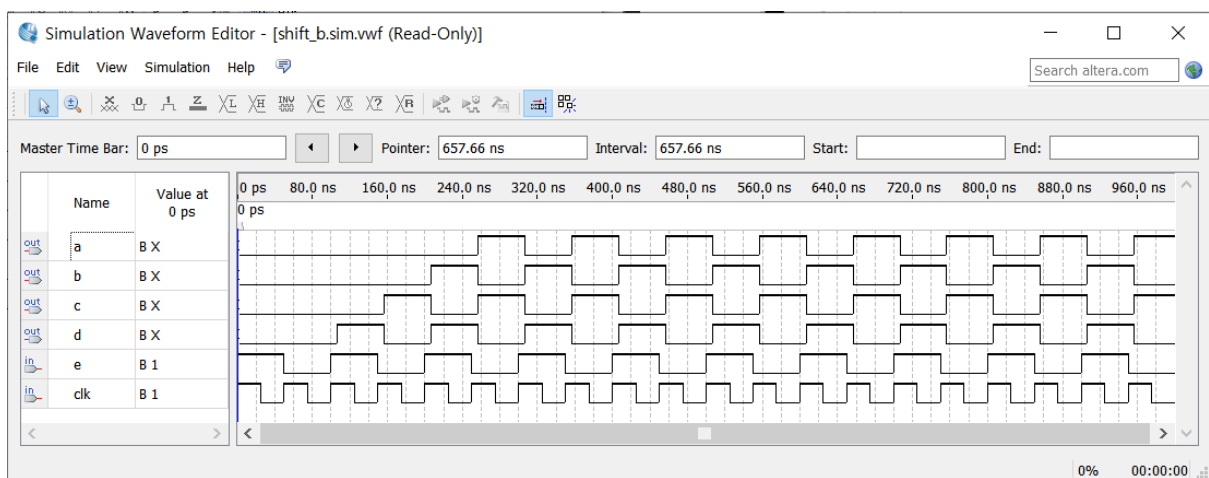


```
1 module shift_nb(clk,a,b,c,d,e);
2     input clk,e;
3     output a,b,c,d;
4     reg a,b,c,d;
5
6     always @(posedge clk) begin
7         a<=b;
8         b<=c;
9         c<=d;
10        d<=e;
11    end
12 endmodule
```



(3)

```
1 module shift_b(clk,a,b,c,d,e);
2     input clk,e;
3     output a,b,c,d;
4     reg a,b,c,d;
5
6     always @(posedge clk) begin
7         a=b;
8         b=c;
9         c=d;
10        d=e;
11    end
12 endmodule
```

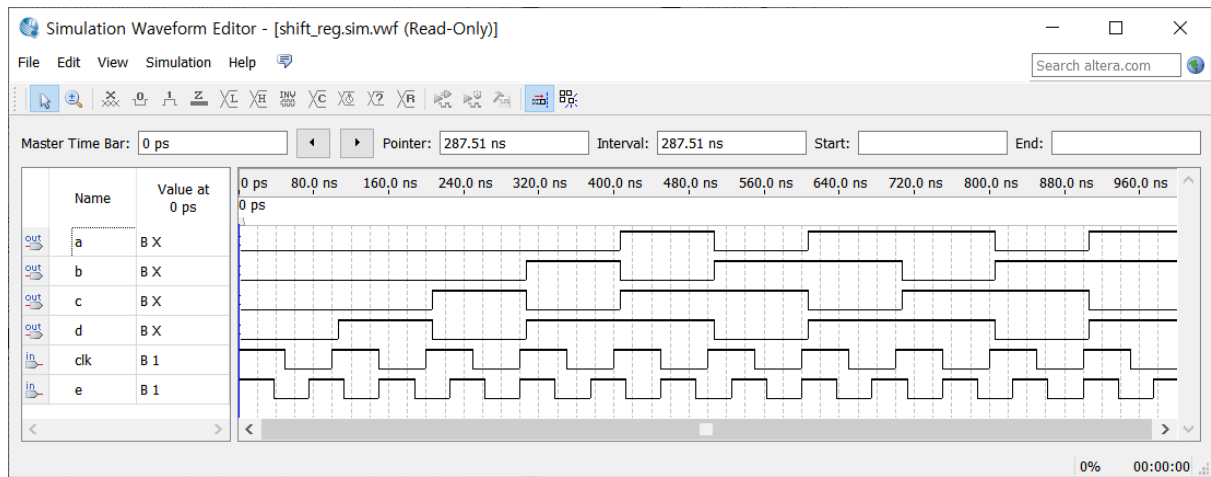


순서에 주의하여 설계해야 한다.

5.

(1)

```
1 module shift_reg(clk,a,b,c,d,e);
2     input clk,e;
3     output a,b,c,d;
4     reg a,b,c,d;
5
6     always @(posedge clk) begin
7         {a,b,c,d}<= {b,c,d,e};
8     end
9 endmodule
```



같다.