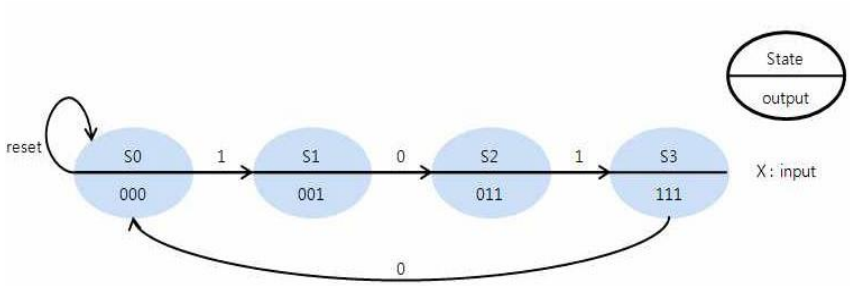
Schematic을 이용한 State machine의 2가지 설계

1) Schematic State machine design

아래의 Moore Machine의 상태도를 D-FF을 사용하여 설계하라.



[위 회로 설계 제출 합니다./VHDL설계 내용은 참조 학습]

```
library ieee;
use ieee.std_logic_1164.all;

entity moore_2p is
port( clk, x, reset : in  std_logic;
        y : out std_logic_vector(2 downto 0));
end moore_2p;

architecture sample of moore_2p  is

type states is (s0, s1, s2, s3);
signal state : states;

begin

p1: process(reset, clk)
begin
  if reset='1' then
```

```
   state <= s0;
  elsif clk'event and clk='1' then
   case state is
    when  s0 =>
                  if x='1' then
                                 state  <= s1;
                  end if;
    when s1 =>
                  if x='0' then
                                 state  <= s2;
                  end if;
    when s2 =>
                  if x='1' then
                                 state  <= s3;
                  end if;
    when s3 =>
                  if x='0' then
                                 state  <= s0;
                  end if;
   end case;
  end if;
end process;

p2: process(state)
begin
  case state is
   when s0 =>
                y <= "000";
   when s1 =>
                y <= "001";
   when s2 =>
                y <= "011";
   when s3 =>
                y <= "111";
```

```
 end case;
end process;

end sample;
```

[변수 정의]

입력: X, RESET
출력: Y2, Y1, Y0
D-FF A의 입력: D_A
D-FF B의 입력: D_B
D-FF A의 입력: A
D-FF B의 입력: B

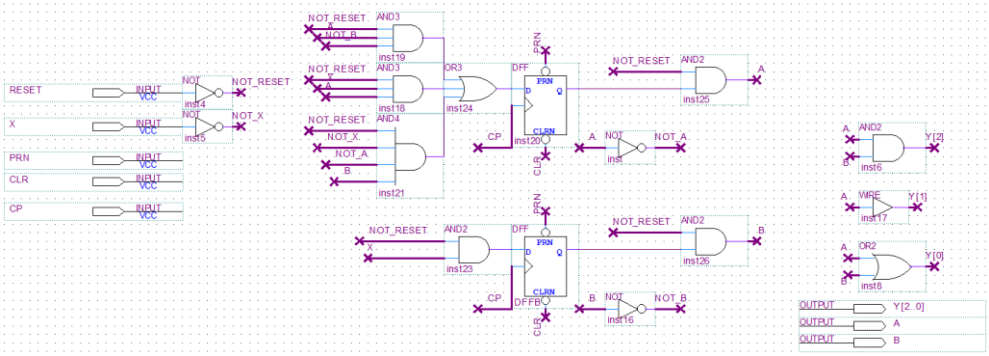
입력		현재 상태			차기 상태			출력		
RESET	X	A	B	상태	A	B	상태	Y2	Y1	Y0
0	0	0	0	S0	0	0	S0	0	0	0
0	0	0	1	S1	1	0	S2	0	0	1
0	0	1	0	S2	1	0	S2	0	1	1
0	0	1	1	S3	0	0	S0	1	1	1
0	1	0	0	S0	0	1	S1	0	0	0
0	1	0	1	S1	0	1	S1	0	0	1
0	1	1	0	S2	1	1	S3	0	1	1
0	1	1	1	S3	1	1	S3	1	1	1
1	0	0	0	S0	0	0	S0	0	0	0
1	0	0	1	Invalid Status (Force S0)	0	0	S0	0	0	0
1	0	1	0		0	0	S0	0	0	0
1	0	1	1		0	0	S0	0	0	0
1	1	0	0		0	0	S0	0	0	0
1	1	0	1		0	0	S0	0	0	0
1	1	1	0		0	0	S0	0	0	0
1	1	1	1		0	0	S0	0	0	0

<상태표>

상태표를 참조하여 부울식으로 A(t+ 1), B(t+ 1)를 나타내면 다음과 같다.
A(t+ 1)=(!RESET∧!X∧!A∧B)+ (!RESET∧!X∧A∧!B)+ (!RESET∧X∧A∧!B)+ (!RESET∧X∧A∧B)
B(t+ 1)=(!RESET∧X∧!A∧!B)+ (!RESET∧X∧!A∧B)+ (!RESET∧X∧A∧!B)+ (!RESET∧X∧A∧B)
D-FF의 특성 방정식은 Q(t+ 1)=D로 나타낼 수 있으므로 D_A, D_B를 4변수 카르노 맵 또는 부울대수 원리를 이용하여 각 식을 최적화하여 나타내면 다음과 같이 나타낼 수 있다.
D_A=(!RESET∧A∧!B)+ (!RESET∧X∧A)+ (!RESET∧!X∧!A∧B)
D_B=!RESET∧X

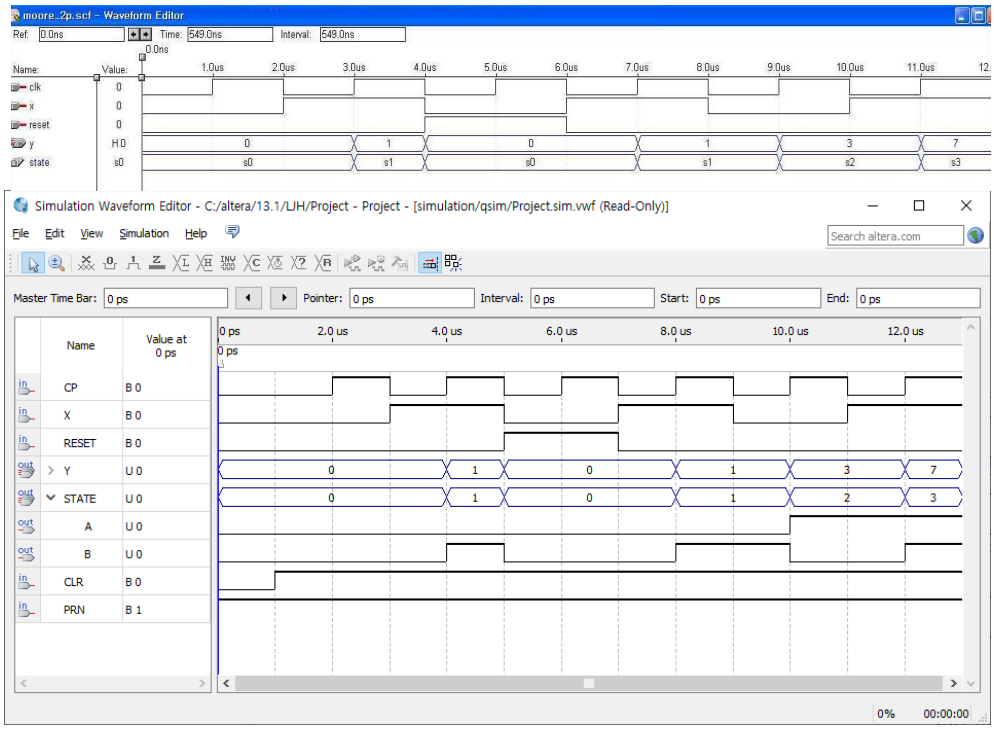
또한 출력 Y2, Y1, Y0는 상태표를 참조하여 다음과 같이 나타낼 수 있다.
Y2=A∧B
Y1=Y2+(A∧!B) =A
Y0=Y1+(!A∧B) =A+B

그런데 RESET=1일 때 D-FF A와 D-FF B의 출력은 항상 0이 되어야 하므로 A와 B를 재정의하여 나타내면
A=!RESET∧RAW_A
B=!RESET∧RAW_B (단, RAW_A, RAW_B는 D-FF A, D-FF B의 출력)

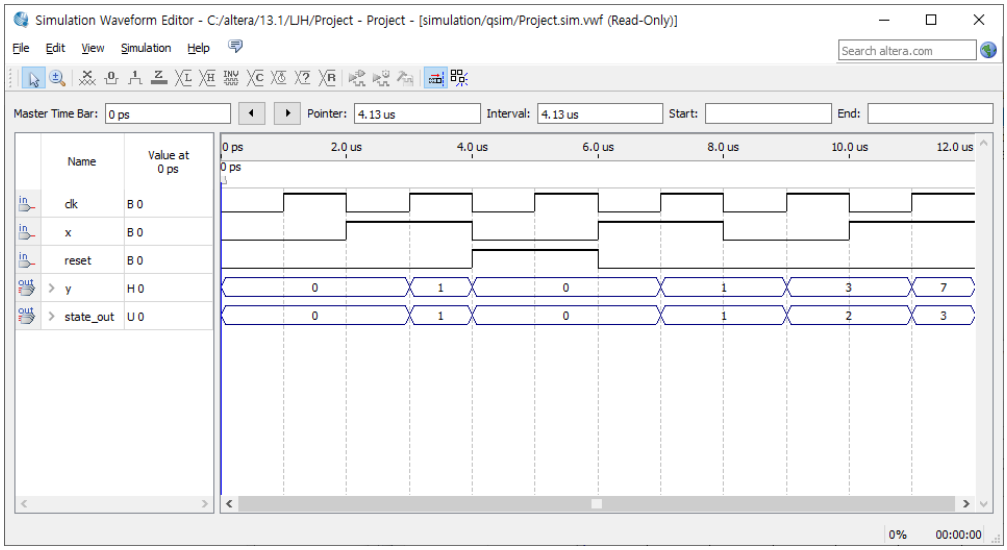


<최종 구현 회로도>

[시뮬레이션]



<설계 회로(Schematic) 시뮬레이션>



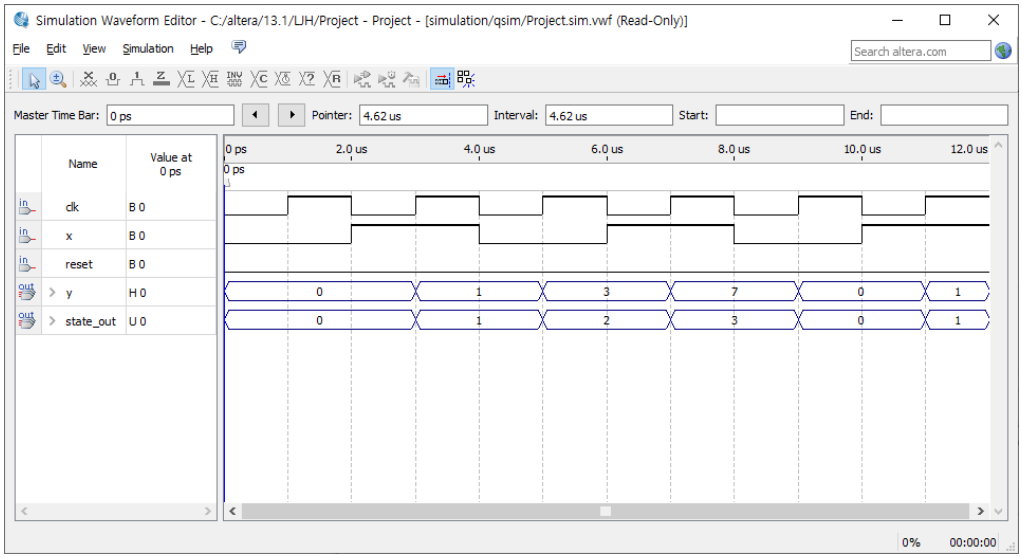
<VHDL 시뮬레이션>

[검토] 위의 moore machine으로 표현한 VHDL 구문을 설계(입력/컴파일)후, 그 결과를 Schematic 결과와 분석 비교하라. ☞ **실행 (File /New/ VHDL File)**

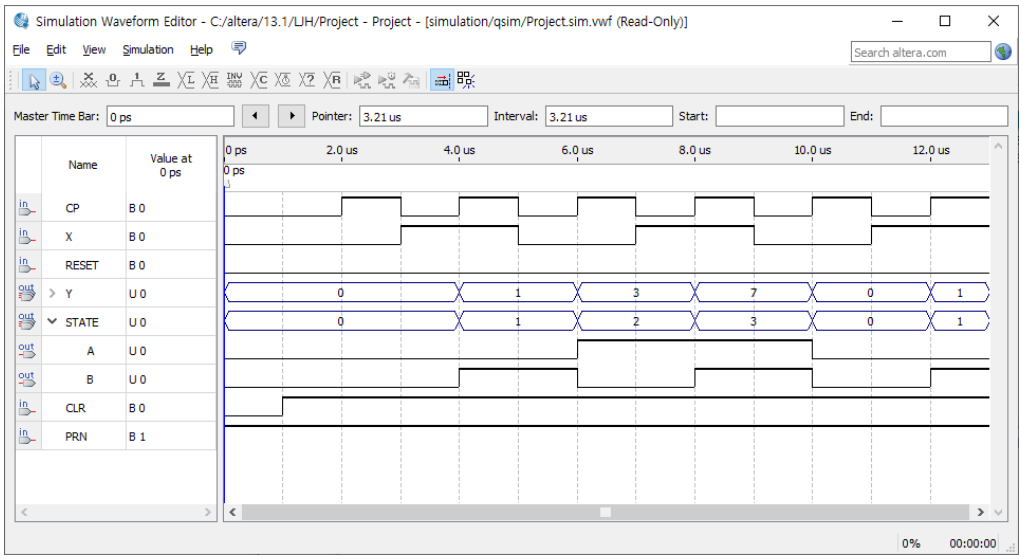
STATE ENUM이 정상적으로 표시되지 않아 숫자로 표시하기 위해 VHDL 코드의 일부를 다음과 같이 수정하였다.
관련 이슈: Quartus Gives Undefined Signal For the State of a Finite State Machine. Supposed to Be Showing Enum of the State_type (StackExchange)

```
port( clk, x, reset : in std_logic;
y : out std_logic_vector(2 downto 0);
state_out : out std_logic_vector(1 downto 0));
--Port 재정의

case state is
        when s0 =>
        state_out <= "00";
        when s1 =>
        state_out <= "01";
        when s2 =>
        state_out <= "10";
        when s3 =>
        state_out <= "11";
end case;
--Process p2 내부에 state_out 처리용 case 추가
```



<설계 회로(Schematic) 시뮬레이션>

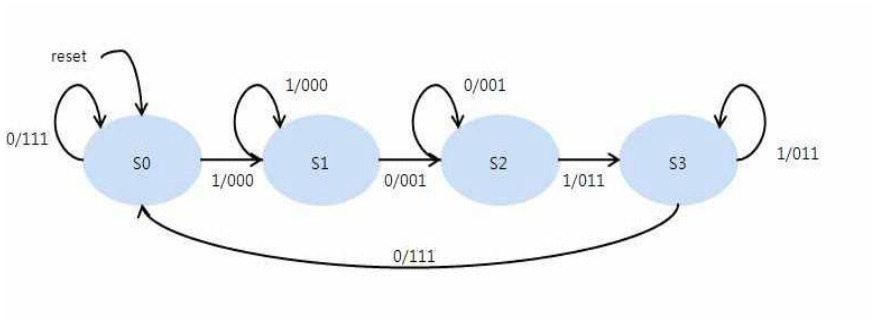


<VHDL 시뮬레이션>

Schematic 설계 회로와 VHDL 두 시뮬레이션 모두 같은 결과를 확인할 수 있었다.

2) Mealy Machine Schematic design

Moore Machine은 출력이 현재상태에만 의존되어 있지만, Mealy Machine의 경우 출력은 현재상태와 현재입력에 의존한다. 아래의 Mealy Machine의 상태도를 이용하여 Schematic 결과와 비교 해 보자.



[위 회로 설계 제출 합니다./VHDL설계 내용은 참조 학습]

```
library ieee;
use ieee.std_logic_1664.all;

entity mealy_2p is
port( clk, x, reset : in  std_logic;
        y : out std_logic_vector(2 downto 0));
end mealy_2p;

architecture sample of mealy_2p  is

type states is (s0, s1, s2, s3);
signal state : states;

begin

p1: process(reset, clk)
begin
 if reset='1' then
   state <= s0;
 elsif clk'event and clk='1'  then
```

```
    case state is
     when s0 =>
                  if x='1' then
                                 state  <= s1;
                  end if;
     when s1 =>
                  if x='0' then
                                 state  <= s2;
                  end if;
     when s2 =>
                  if x='1' then
                                 state  <= s3;
                  end if;
     when s3 =>
                  if x='0' then
                                 state  <= s0;
                  end if;
     end case;
    end if;
end process;

p2: process(state, x)
begin
  case state is
   when s0 =>
                if x='1' then
                               y  <= "000";
                else
                               y  <= "111";
                end if;
   when s1 =>
                if x='1' then
                               y  <= "000";
                else
```

```
                        y  <= "001";
            end if;
  when s2 =>
            if x='1' then
                        y  <= "011";
            else
                        y  <= "001";
            end if;
  when s3 =>
            if x='1' then
                        y  <= "011";
           else
                        y  <= "111";
            end if;
 end case;
end  process;

end  sample;
```

[변수 정의]

입력: X, RESET
출력: Y2, Y1, Y0
D-FF A의 입력: D_A
D-FF B의 입력: D_B
D-FF A의 입력: A
D-FF B의 입력: B

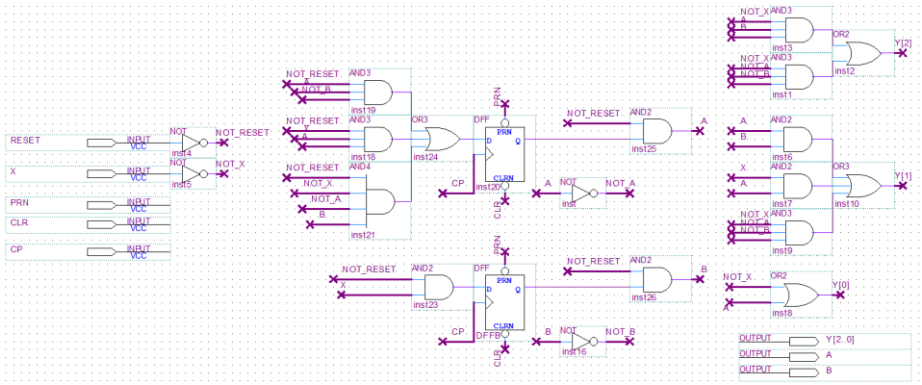
입력		현재 상태			차기 상태			출력		
RESET	X	A	B	상태	A	B	상태	Y2	Y1	Y0
0	0	0	0	S0	0	0	S0	1	1	1
0	0	0	1	S1	1	0	S2	0	0	1
0	0	1	0	S2	1	0	S2	0	0	1
0	0	1	1	S3	0	0	S0	1	1	1
0	1	0	0	S0	0	1	S1	0	0	0
0	1	0	1	S1	0	1	S1	0	0	0
0	1	1	0	S2	1	1	S3	0	1	1
0	1	1	1	S3	1	1	S3	0	1	1
1	0	0	0	S0	0	0	S0	0	0	0
1	0	0	1	Invalid Status (Force S0)	0	0	S0	0	0	0
1	0	1	0		0	0	S0	0	0	0
1	0	1	1		0	0	S0	0	0	0
1	1	0	0		0	0	S0	0	0	0

1	1	0	1		0	0	S0	0	0	0
1	1	1	0		0	0	S0	0	0	0
1	1	1	1		0	0	S0	0	0	0

상태표를 참조하여 다음과 같이 D_A, D_B를 나타낼 수 있다. (1번의 Moore Machine과 동일)
D_A=(!RESET∧A∧!B)+ (!RESET∧X∧A)+ (!RESET∧!X∧!A∧B)
D_B=!RESET∧X

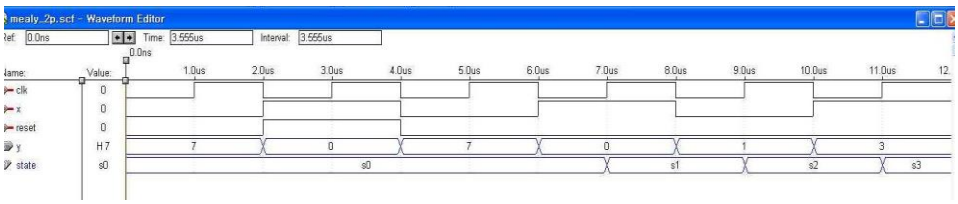
출력 Y2, Y1, Y0는 상태표를 참조하여 다음과 같이 나타낼 수 있다.
Y2=(!X∧!A∧!B)+ (!X∧A∧B)=(A∧B∧!X)+ (!A∧!B∧!X)
Y1=Y2+ (X∧A∧!B)+ (X∧A∧B)=(A∧B)+ (A∧X)+ (!A∧!B∧!X)
Y0=Y1+ (!X∧!A∧B)+ (!X∧A∧!B)=A+ !X

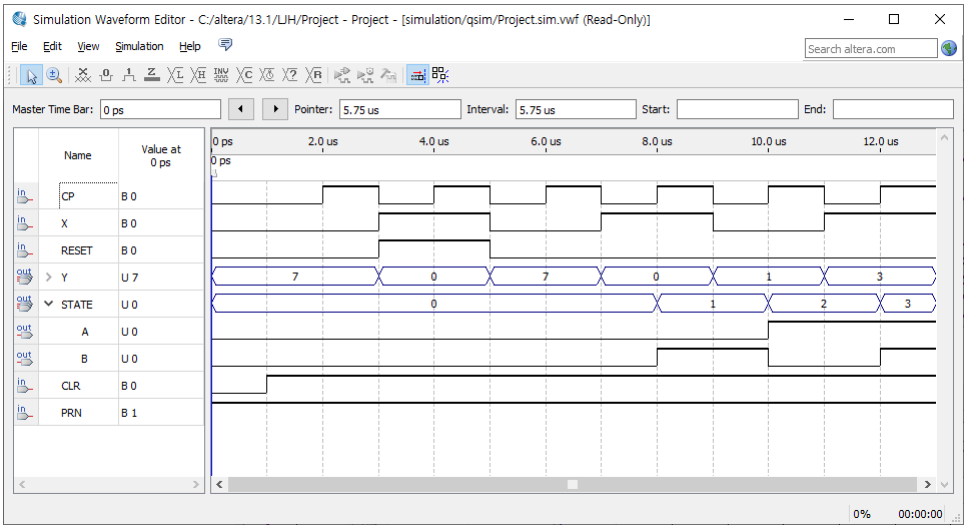
RESET=1일 때 D-FF A와 D-FF B의 출력은 항상 0이 되어야 하므로 A와 B를 재정의하여 나타내면
A=!RESET∧RAW_A
B=!RESET∧RAW_B (단, RAW_A, RAW_B는 D-FF A, D-FF B의 출력)



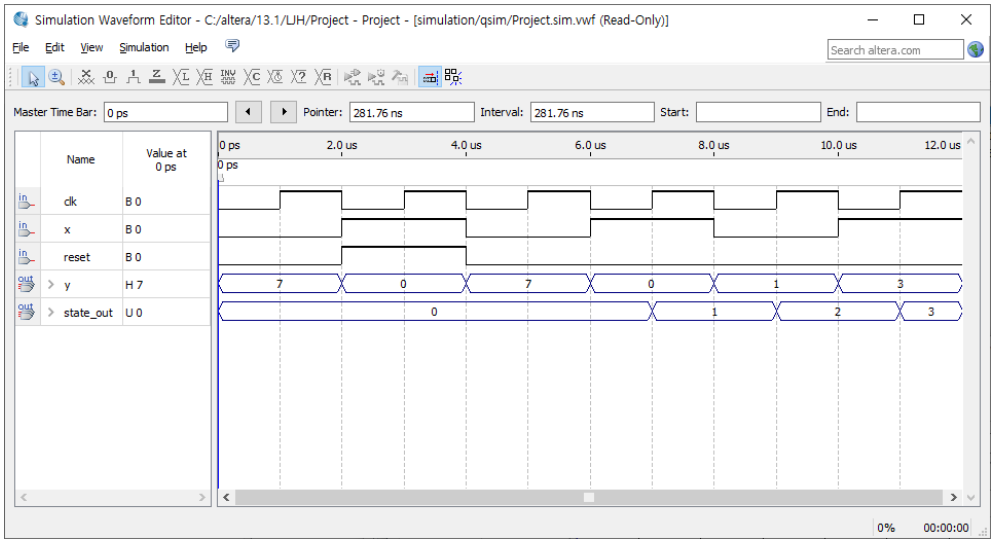
<최종 구현 회로도>

[시뮬레이션]





<설계 회로(Schematic) 시뮬레이션>

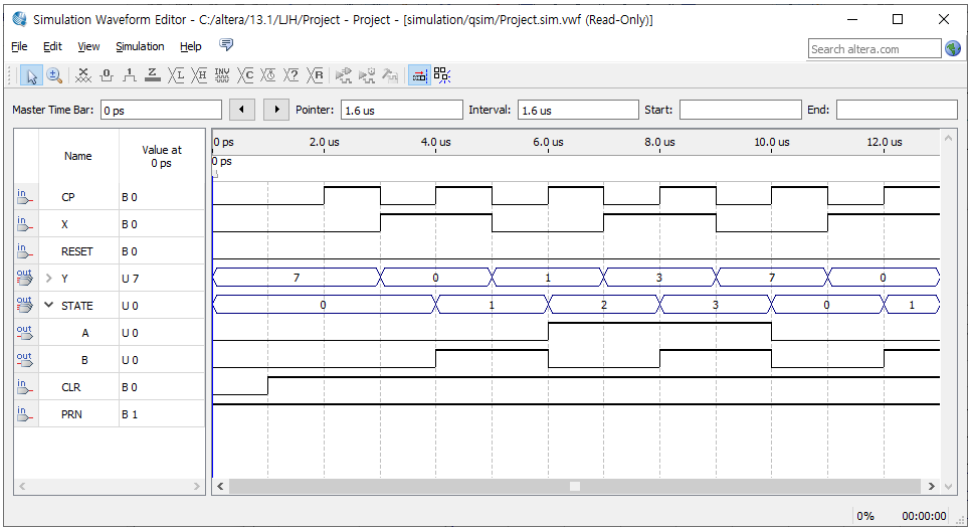


<VHDL 시뮬레이션>

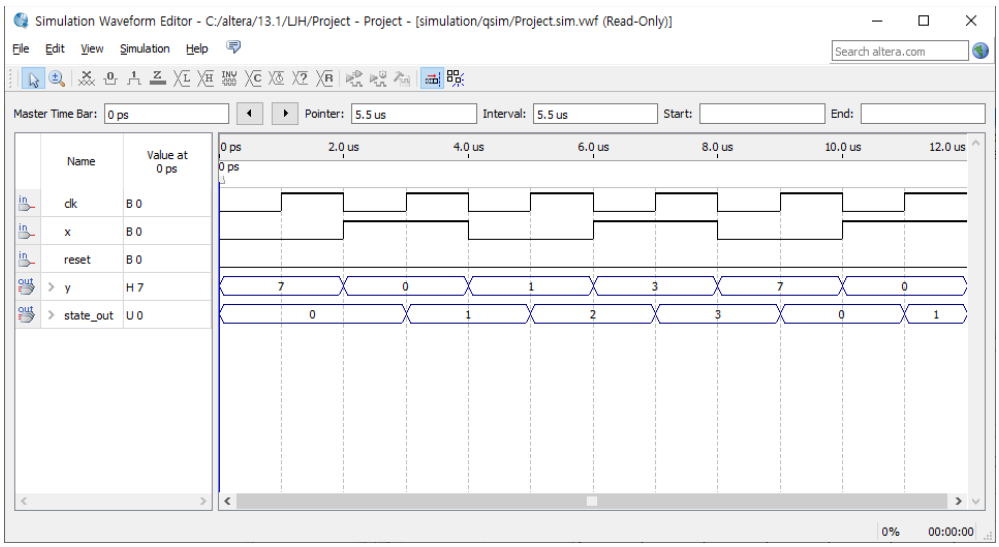
[검토]

앞서 설명한 mealy machine으로 표현한 VHDL 구문을 설계(입력/컴파일)후, 그 결과를 Schematic 모의 실험결과와 분석 비교하라.

☞ **실행 (File /New/ VHDL File)**



<설계 회로(Schematic) 시뮬레이션>



<VHDL 시뮬레이션>

Schematic 설계 회로와 VHDL 두 시뮬레이션 모두 같은 결과를 확인할 수 있었다.

[느낀 점]

플립플롭과 같은 저장 장치를 이용하여 회로를 직접 설계 & 구성해보니 문득 컴퓨터도 이런 원리에서 확장하여 만들어 진 것이구나 생각이 들었고, 이런 순차논리회로를 이용하면 무궁무진한 디지털 회로들을 만들 수 있을 것 같다는 생각이 들었습니다.