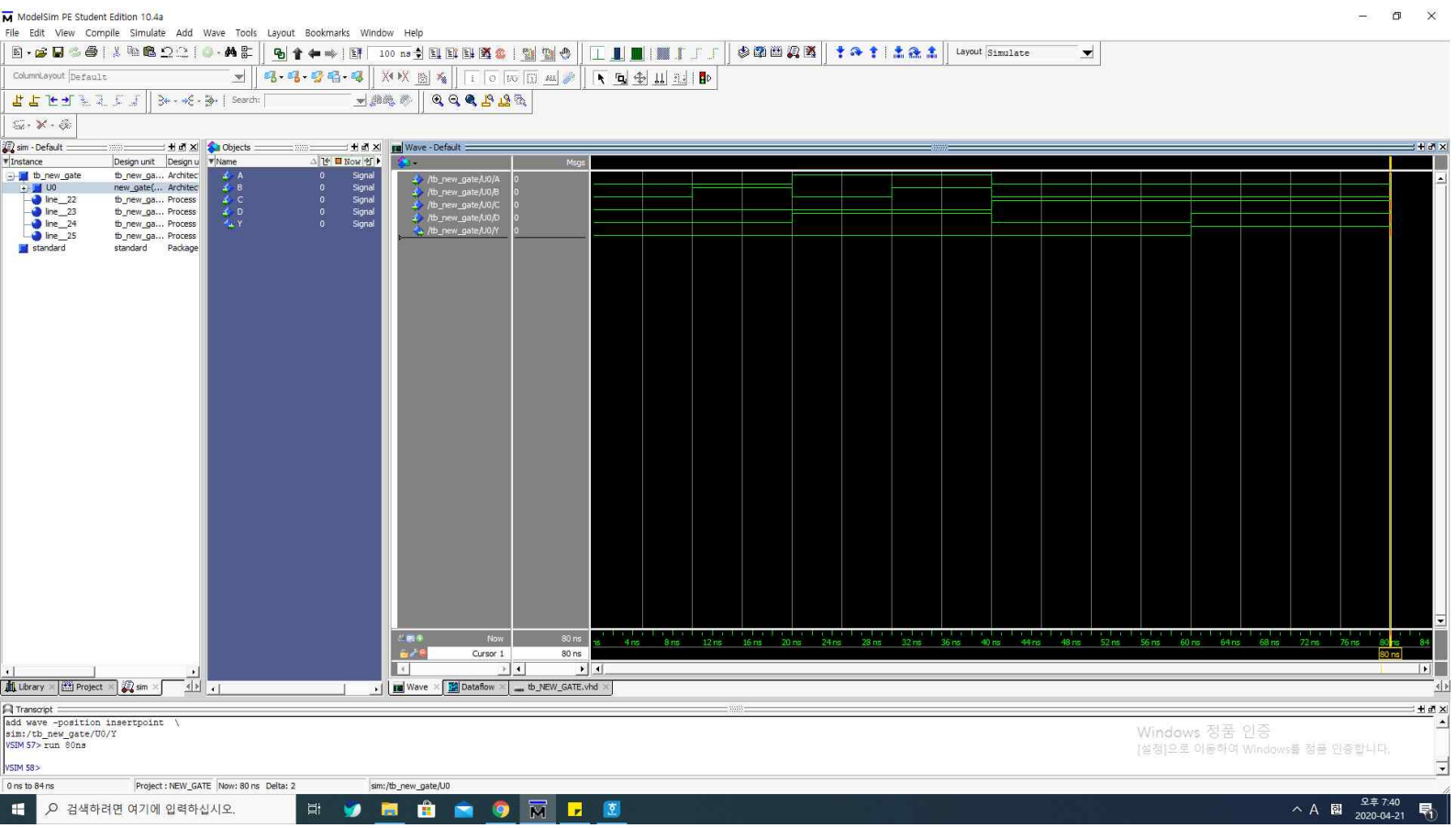
1번.

```
entity NEW_GATE is
port (A, B,C,D : in bit; Y: out bit); --input A,B,C,D output Y
end;

architecture NEW_GATE of NEW_GATE is  --architecture에서 NEW_GATE 내부회로서술
begin
Y<= ((A OR B)NOR(C NAND D)); --output Y의 연산
end;

entity tb_NEW_GATE is --테스트벤치 entity선언
port (Y : out bit) ;
end tb_NEW_GATE ;

architecture simulation of tb_NEW_GATE is --tb achitecture 선언
component NEW_GATE --component 선언
port (A, B,C,D: in bit; Y : out bit);
end component;

signal a,b,c,d : bit; --signal 선언
begin
a <= '0', '1' after 20 ns, '0' after 40 ns; --파형 생성문으로 입력 신호 생성
b <= '0', '1' after 10 ns, '0' after 20 ns, '1' after 30 ns, '0' after 40 ns;
c <= '0', '1' after 40 ns, '0' after 80 ns;
d <= '0', '1' after 20 ns, '0' after 40 ns, '1' after 60 ns, '0' after 80 ns;
U0 : NEW_GATE port map (A=>a, B=>b,C=>c,D=>d,Y=>Y); --component 실체화
end ;
```



출력 Y가 1이 되기 위한 입력 신호의 조건

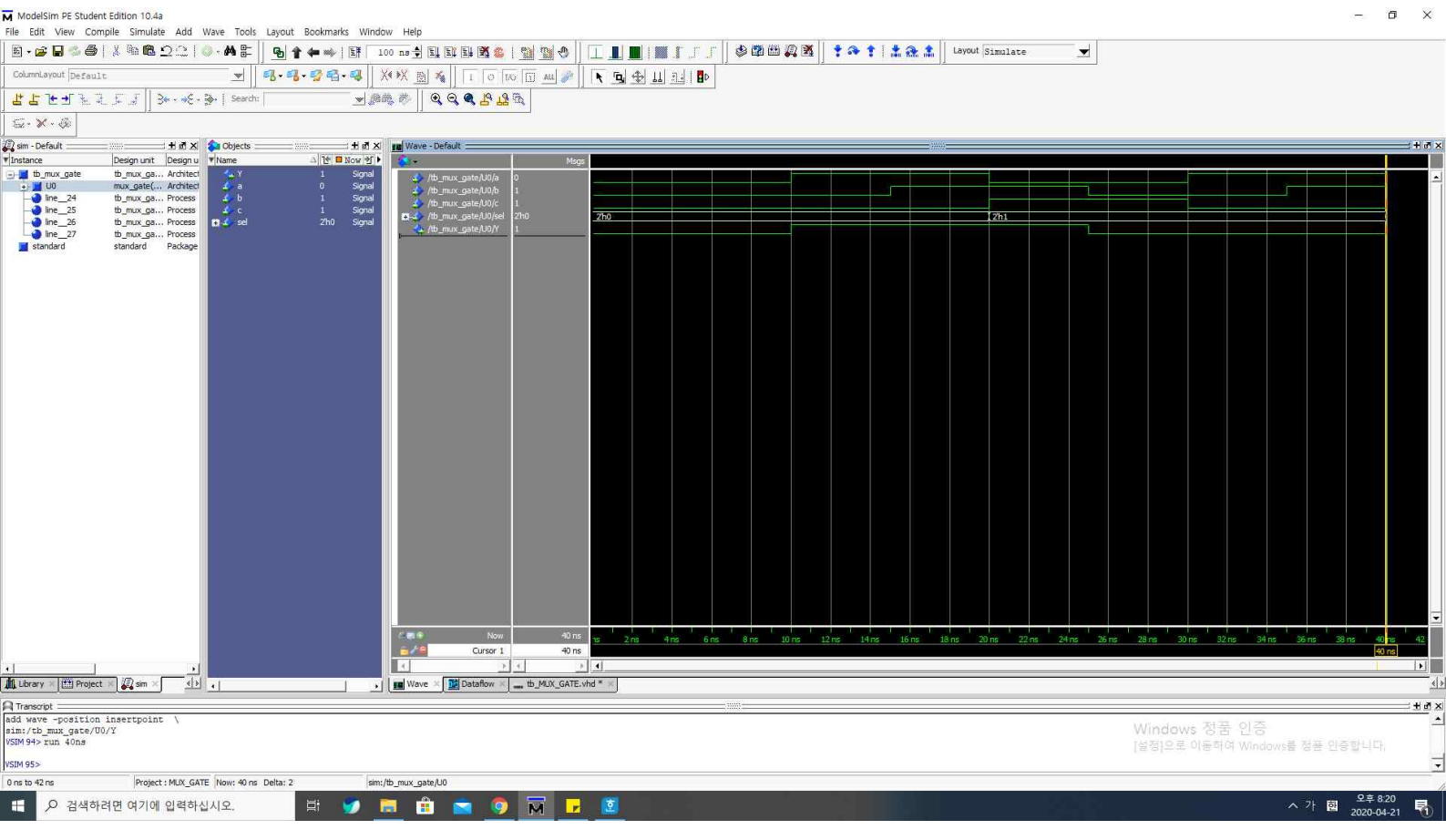
NOR게이트의 출력이 1이 되어야 하기 때문에 NOR게이트의 입력이 0,0이 되어야 한다.

따라서, (A OR B)가 0이면서 (C NAND D)가 0이어야 하기 때문에

위에 그림처럼 A=0, B=0,C=1,D=1인 경우에 출력 Y가 1이 된다.

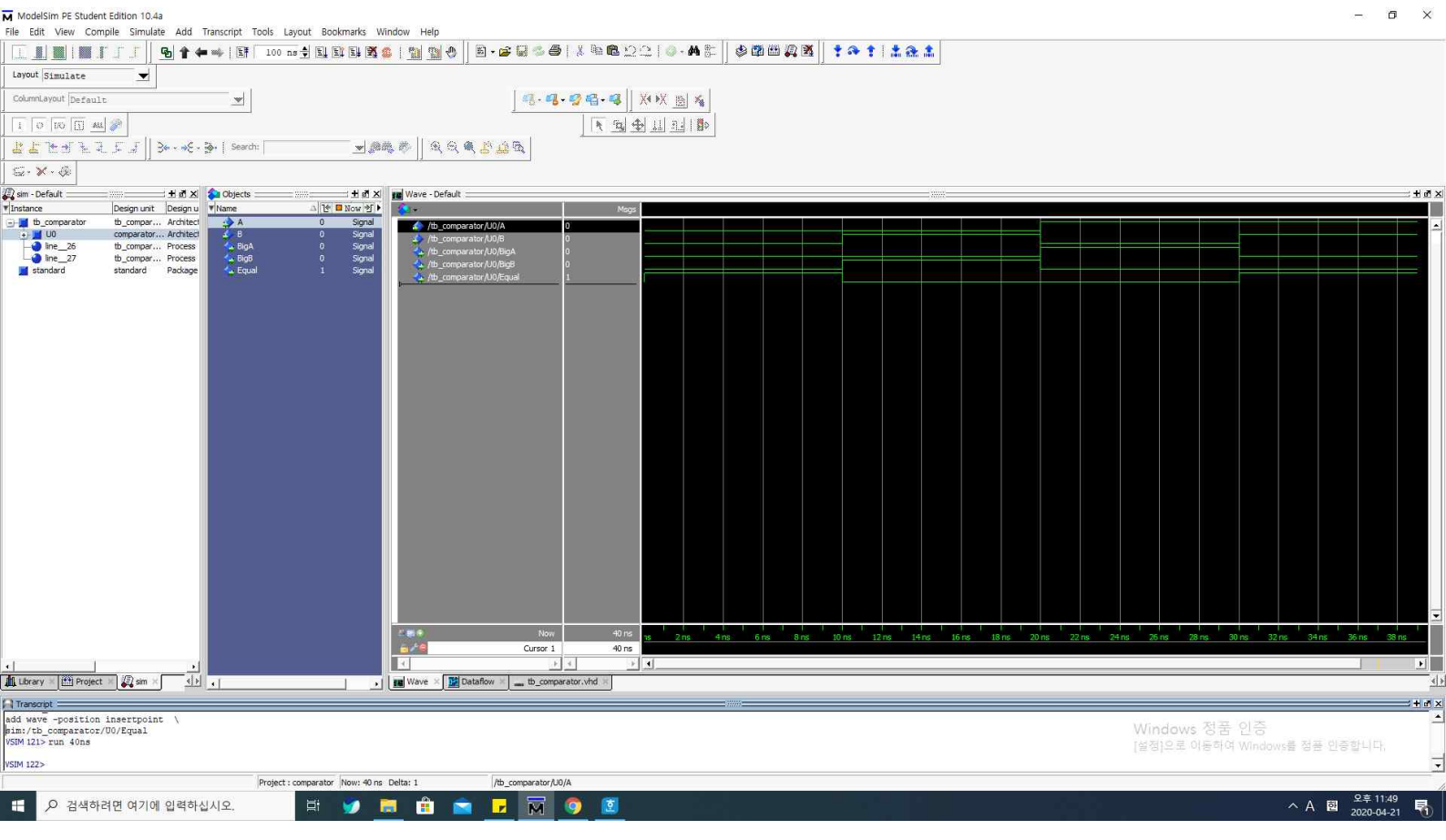
2번

```
entity MUX_GATE is --MUX_GATE entity 선언
port( a, b, c: in bit; sel : in bit_vector(1 downto 0); --입출력 선언
Y : out bit);
end;

architecture with_select of MUX_GATE is --architecture에서 내부회로 서술
begin
with sel select
Y <= (a or b) when "00",--sel 0이면 a or b
        (b and c) when "01", -- sel 1이면 b or c
        a when others;
end;

entity tb_MUX_GATE is --tb entity 선언
end tb_MUX_GATE ;

architecture simulation of tb_MUX_GATE is --tb achitecture 선언
component MUX_GATE --component 선언
port ( a, b, c: in bit; sel : in bit_vector(1 downto 0);
Y : out bit);
end component;

signal a,b,c,d : bit; signal sel : bit_vector(1 downto 0);
begin
A <= '0', '1' after 10 ns, '0' after 20 ns, '1' after 30 ns, '0' after 40 ns; --파형 생성문으로 입력 신호 생성
B <= '0', '1' after 15 ns, '0' after 25 ns, '1' after 35 ns, '0' after 45 ns;
C <= '0', '1' after 20 ns, '0' after 30 ns, '1' after 40 ns, '0' after 50 ns;
SEL <= "00", "01" after 20 ns, "00" after 40 ns, "01" after 60 ns, "00" after 80 ns;
U0 : MUX_GATE port map (A=>A,B=>B,C=>C, SEL=> SEL, Y=>open); --component 실체화
end ;
```



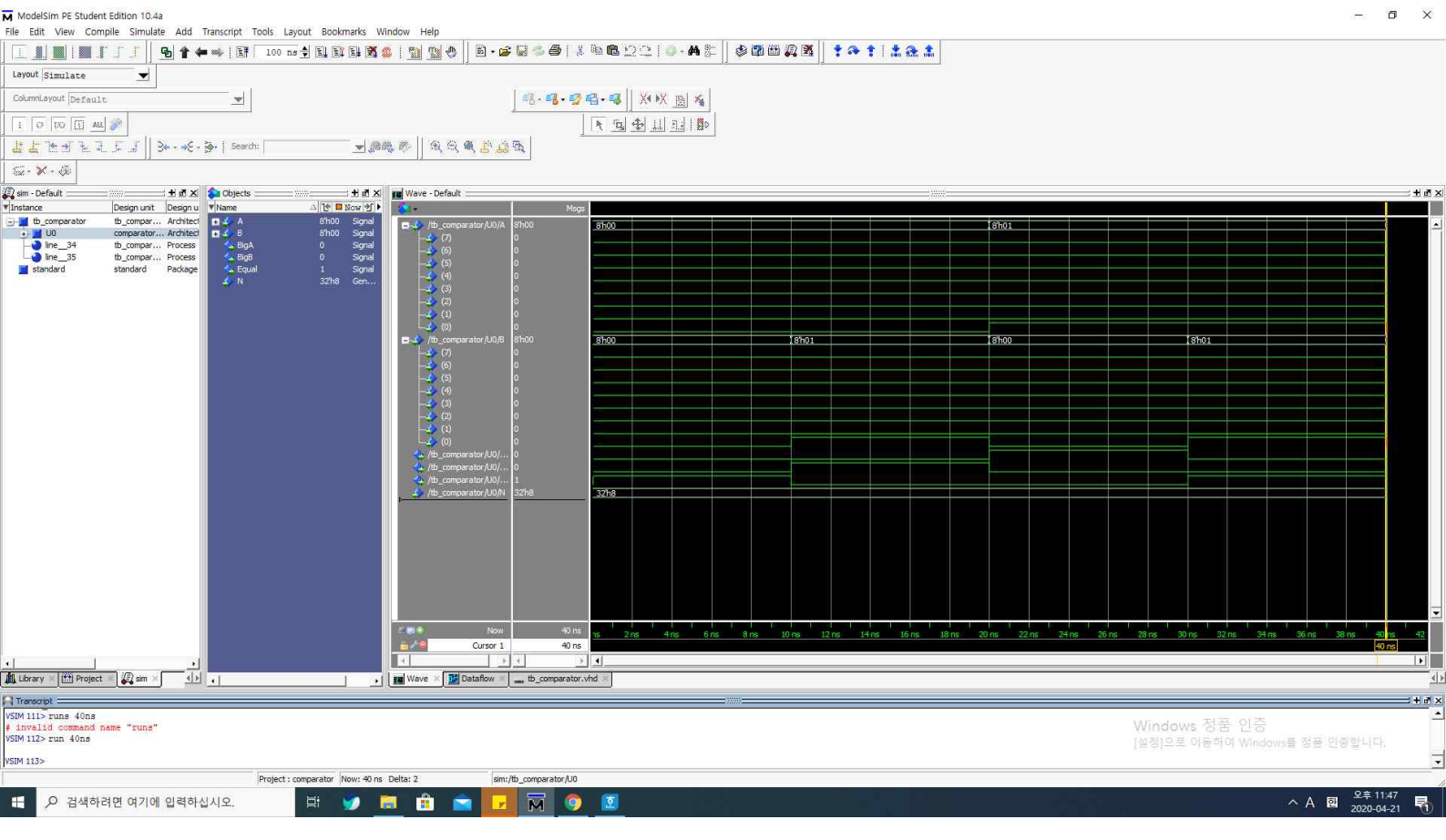
3번

```
entity comparator is --comparator entity 선언
port(A, B: in bit; Equal,BigA,BigB : out bit); --입출력 선언
end comparator;

architecture behavioral of comparator is ㅡ아키텍쳐 선언
begin
process (A, B) --process문(A또는B가 바뀌면 동작)
begin
Equal<='0'; BigA<='0'; BigB<='0'; -- 기본값 0
if (A = B) then Equal<='1';-- A=B이면 Equal=1
elsif (A>B) then BigA<='1';-- A>B이면 BigA=1
else BigB<='1';            -- A<B이면 BigB=1
end if;
end process;
end;

entity tb_comparator is ㅡtb_comparator 선언
end tb_comparator ;

architecture simulation of tb_comparator is --tb_architecture 선언
component comparator --component 선언
port (A, B: in bit; Equal,BigA,BigB : out bit);
end component;

signal signal_a,signal_b : bit; --signal선언
begin
signal_a <= '0', '1' after 20 ns, '0' after 40 ns; --파형 생성문으로 입력 신호 생성
signal_b <= '0', '1' after 10 ns, '0' after 20 ns, '1' after 30 ns, '0' after 40 ns;
U0 : comparator port map (A=>signal_a, B=>signal_b, Equal=>open,BigA=>open,BigB=>open); --component 실체화
end;
```



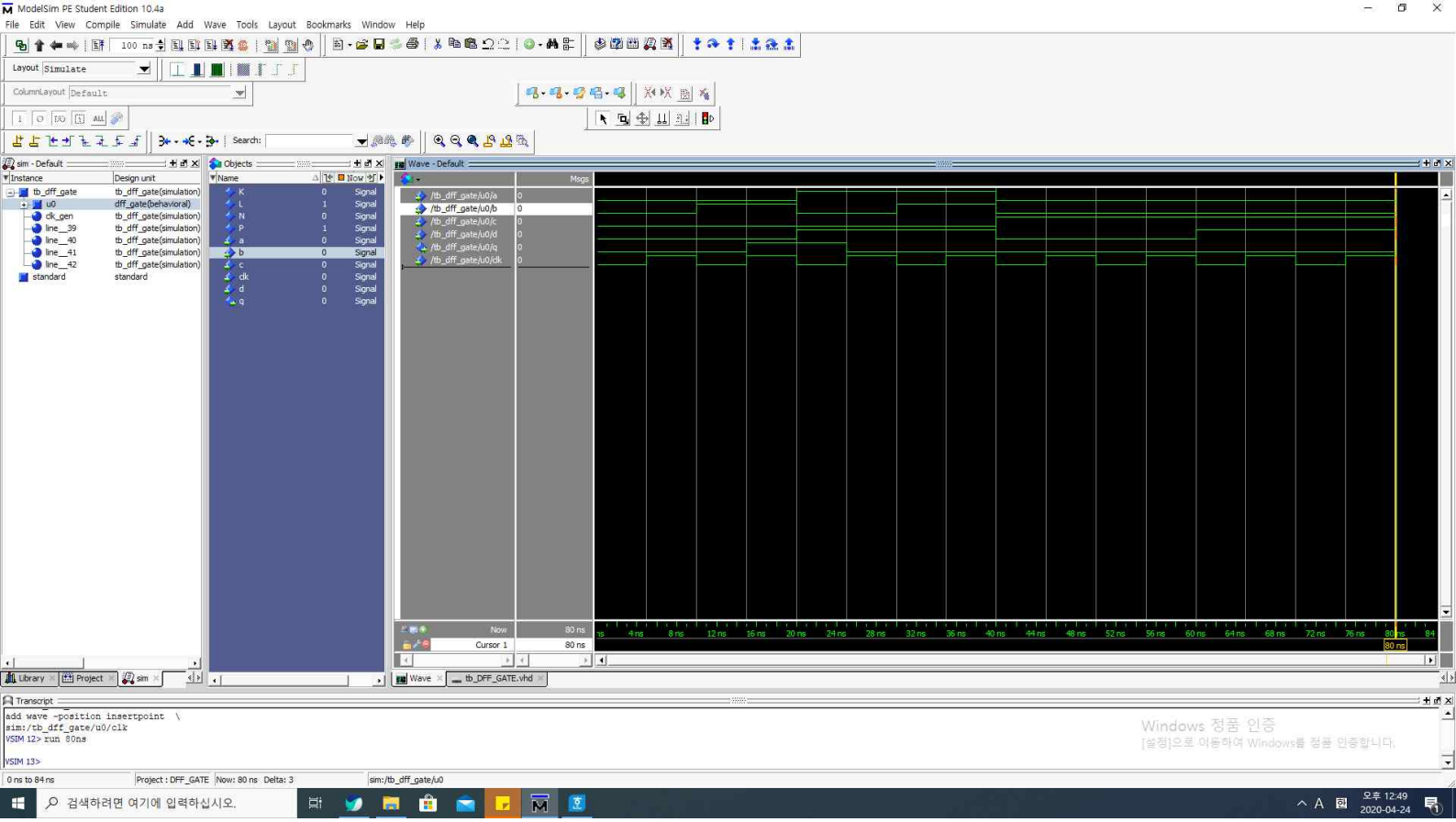
제네릭을 이용하여 입력 벡터 a와b의 비트수를 일반화 하고 8비트로 구현

```
entity comparator is --comparator entity 선언
generic( N : integer :=8); --제네릭 선언 및 초기화
port( A, B : in bit_vector(7 downto 0); --입출력 선언
Equal,BigA,BigB: out bit);
end comparator;
architecture behavioral of comparator is ㅡ아키텍쳐 선언
begin
process (A, B) --process문(A또는B가 바뀌면 동작)
begin
Equal<='0'; BigA<='0'; BigB<='0'; -- 기본값 0
if (A = B) then Equal<='1';-- A=B이면 Equal=1
elsif (A>B) then BigA<='1';-- A>B이면 BigA=1
else BigB<='1';            -- A<B이면 BigB=1
end if;
end process; --process문 종료
end;
entity tb_comparator is -- tb선언
end tb_comparator ;
architecture simulation of tb_comparator is -- tb 아키텍쳐 선언
component comparator --  component 선언
generic( N : integer :=8);
port (A, B : in bit_vector(7 downto 0); Equal,BigA,BigB : out bit);
end component; -- component 종료
signal signal_a,signal_b : bit_vector(7 downto 0); --signal 선언
begin
signal_a <= "00000000", "00000001" after 20 ns, "00000000" after 40 ns; --파형 생성문으로 입력 신호 생성
signal_b <= "00000000", "00000001" after 10 ns, "00000000" after 20 ns,"00000001" after 30 ns, "00000000" after 40 ns;
U0 : comparator generic map(N=>8) -- component 실체화
port map(A=>signal_a,B=>signal_b, Equal=>open,BigA=>open,BigB=>open);
end ;
```



4번

```
entity dff_gate is  -- entity 선언
port ( a,b,c,d : in bit; q : out bit; -- 입출력
clk : in bit);
end dff_gate;
architecture behavioral of dff_gate is -- 아키텍쳐 선언
signal K,L,N,P : bit; -- 신호 선언
begin
process (clk) --  프로세스문 clk이 바뀌면 동작
begin

if (clk'event and clk = '1') then -- 라이징 클럭으로 바뀔 때
K <= (a or b); -- 연산
L <= (c nand d);
N<=L;
P <=(K nor N);
q <= P;
end if; -- if문 종료

end process; -- 프로세스문 종료
end behavioral;  -- 아키텍쳐 종료
entity tb_dff_gate is -- 테스트벤치 선언
end tb_dff_gate ;
architecture simulation of tb_dff_gate is ㅡ 테스트벤치 아키텍쳐
component dff_gate ㅡ컴포넌트 선언
port ( a,b,c,d : in bit; -- 입출력
q : out bit;
clk : in bit);
end component; -- 컴포넌트 종료
constant HALF_PERIOD_100M : time := 5 ns ; --클럭신호 주파수 200Mhz
signal a,b,c,d : bit; -- signal선언
signal clk : bit;
begin
clk_gen : process -- 프로세스문
begin
while (true) loop -- while loof
clk <= '0' ; wait for HALF_PERIOD_100M ;
clk <= '1' ; wait for HALF_PERIOD_100M ;
end loop; --while loof 종료
end process; -- 프로세스문 종료

a <= '0', '1' after 20 ns, '0' after 40 ns; --파형 생성문으로 입력 신호 생성
b <= '0', '1' after 10 ns, '0' after 20 ns, '1' after 30 ns, '0' after 40 ns;
c <= '0', '1' after 40 ns, '0' after 80 ns;
d <= '0', '1' after 20 ns, '0' after 40 ns, '1' after 60 ns, '0' after 80 ns;
u0: dff_gate port map (a=>a,b=>b,c=>c,d=>d, q => open, clk=>clk); -- 컴포넌트 실체화
end simulation; -- 종료
```


ModelSim PE Student Edition 10.4a
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
sim - Default
Instance
Design unit
tb_dff_gate
tb_dff_gate(simulation)
u0
dff_gate(behavioral)
clk_gen
line__39
line__40
line__41
line__42
standard
Objects
Name
Now
K
L
N
P
a
b
c
clk
d
q
Signal
Wave - Default
Msgs
/tb_dff_gate/u0/a
/tb_dff_gate/u0/b
/tb_dff_gate/u0/c
/tb_dff_gate/u0/d
/tb_dff_gate/u0/q
/tb_dff_gate/u0/clk
Now
Cursor 1
80 ns
Library
Project
sim
Wave
tb_DFF_GATE.vhd
Transcript
add wave -position insertpoint \
sim:/tb_dff_gate/u0/clk
VSIM 12> run 80ns
VSIM 13>
0 ns to 84 ns
Project : DFF_GATE
Now: 80 ns Delta: 3
sim:/tb_dff_gate/u0
Windows 정품 인증
[설정]으로 이동하여 Windows를 정품 인증합니다.
검색하려면 여기에 입력하십시오.
오후 12:49
2020-04-24