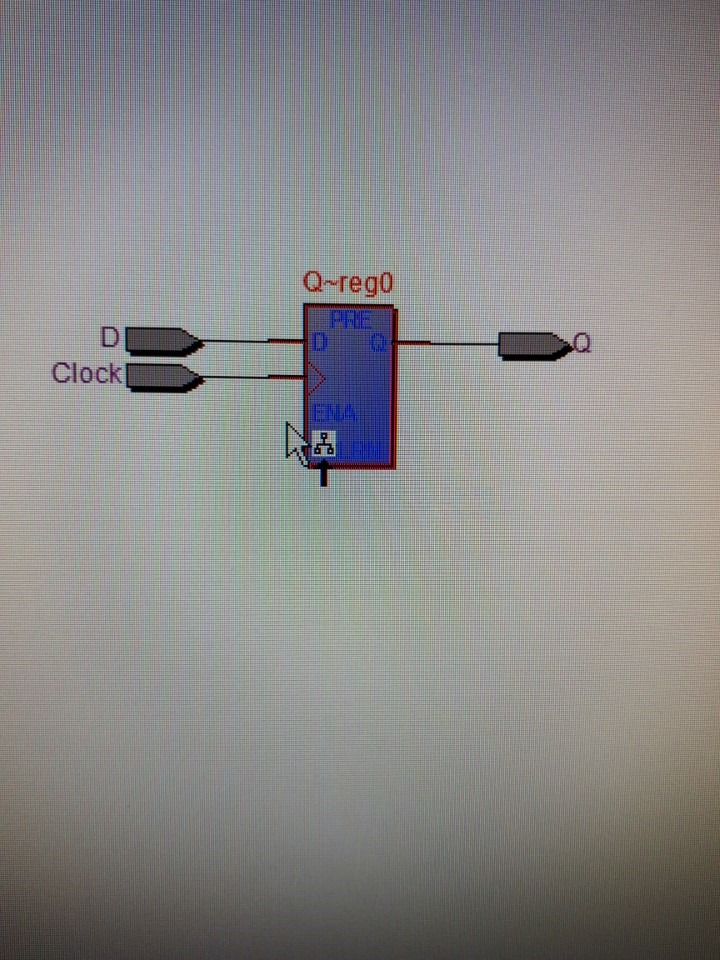
Lab8 Final Report

20160410 윤석훈

1. RTL Diagram Comparing



D Flip-Flop RTL Diagram with Behavioral Code.

D Flip-Flop의 심볼이 나타나 있다. 아주 간략화된 모습이다.

D Flip-Flop RTL Diagram with logical expression의 이미지를 저장하지 못했습니다. 이것의 RTL Diagram은 4개의 XOR Gate와 1개의 Not Gate로 이루어져 있어서 D Flip-Flop의 심볼로 간략화되지는 않을 것이라고 예측합니다.

2.(A) Binary Counter Source Code

library ieee;  
use ieee.std\_logic\_1164.all;

entity T\_FF is  
  port(clk,reset,t: in std\_logic;  
        Q: out std\_logic);  
end T\_FF;

architecture behave of T\_FF is  
  signal q\_reg: std\_logic;  
  signal q\_next: std\_logic;  
begin  
 process(reset, clk)  
 begin  
       if (reset = '1') then  
          q\_reg <= '0';  
       elsif (clk'event and clk = '1') then  
          q\_reg <= q\_next;  
       end if;  
 end process;

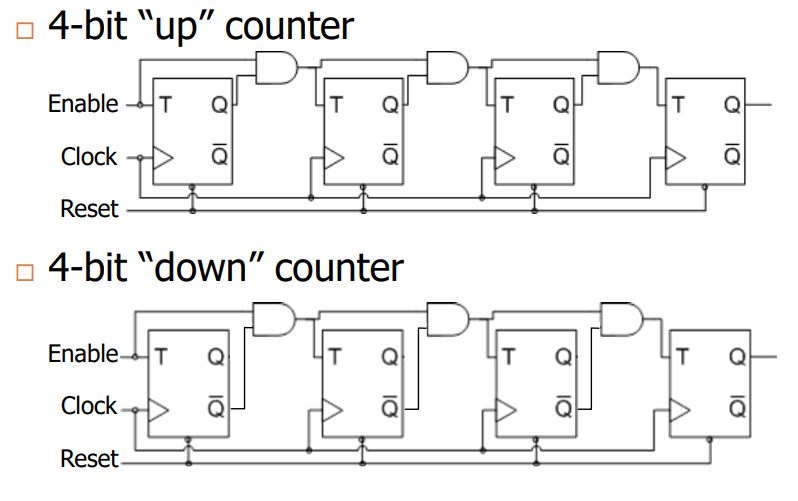
       q\_next <= q\_reg when t = '0' else  
      not(q\_reg);

       Q <= q\_reg;  
end behave;

Library ieee;  
USE ieee.std\_logic\_1164.all;

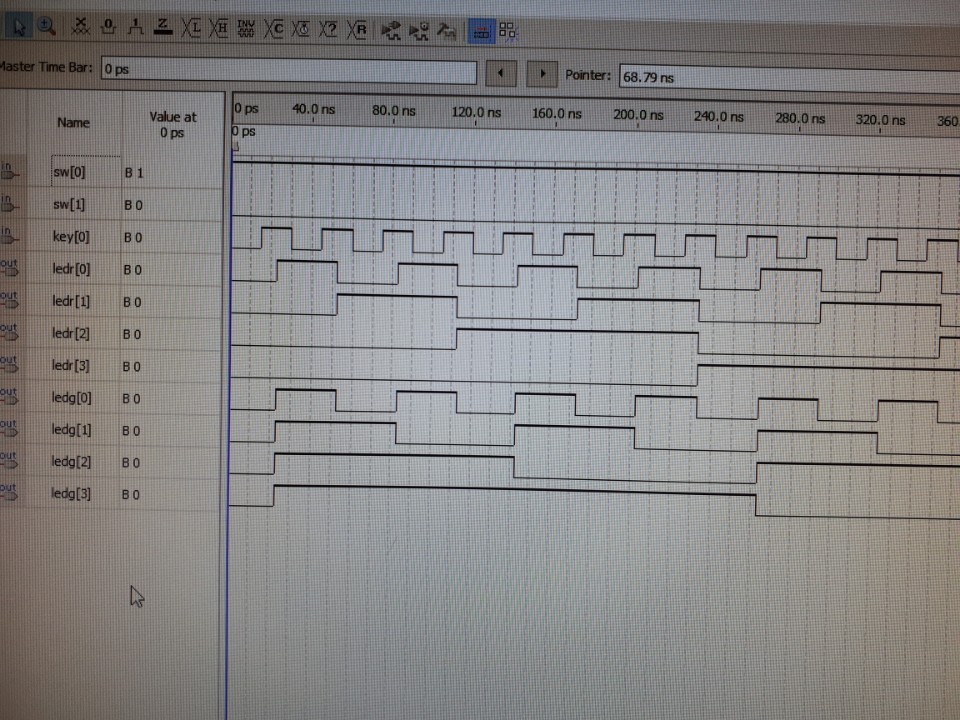
Entity LAB8 is  
 port(sw: in std\_logic\_vector(1 downto 0);  
    key: in std\_logic\_vector(0 downto 0);  
  ledr: out std\_logic\_vector(3 downto 0);  
  ledg: out std\_logic\_vector(3 downto 0));  
end entity;  
architecture LB8 of LAB8 is  
Component T\_FF  
port( t, reset, clk: in std\_logic;  
Q: out std\_logic);  
end component;  
signal Q0, Q1, Q2, Q3, Q4, Q5, Q6, Q7: std\_logic;  
begin  
 U0: T\_FF PORT MAP(t=> sw(0), reset=>sw(1), clk=>key(0), Q=>Q0);  
 U1: T\_FF PORT MAP(t=> (sw(0) and Q0), reset=>sw(1), clk=>key(0), Q=>Q1);  
 U2: T\_FF PORT MAP(t=> ((sw(0) and Q0) and Q1), reset=>sw(1), clk=>key(0), Q=>Q2);  
 U3: T\_FF PORT MAP(t=> (((sw(0) and Q0) and Q1) and Q2), reset=>sw(1), clk=>key(0), Q=>Q3);  
   
 U4: T\_FF PORT MAP(t=> sw(0), reset=>sw(1), clk=>key(0), Q=>Q4);  
 U5: T\_FF PORT MAP(t=> (sw(0) and (not Q4)), reset=>sw(1), clk=>key(0), Q=>Q5);  
 U6: T\_FF PORT MAP(t=> ((sw(0) and (not Q4)) and (not Q5)), reset=>sw(1), clk=>key(0), Q=>Q6);  
 U7: T\_FF PORT MAP(t=> (((sw(0) and (not Q4)) and (not Q5)) and (not Q6)), reset=>sw(1), clk=>key(0), Q=>Q7);  
   
 ledr(0)<=Q0;  
 ledr(1)<=Q1;  
 ledr(2)<=Q2;  
 ledr(3)<=Q3;  
 ledg(0)<=Q4;  
 ledg(1)<=Q5;  
 ledg(2)<=Q6;  
 ledg(3)<=Q7;  
end LB8;

\*Explanation: Binary Counter는 T Flip-flop으로 구현되었다. 여기의 T Flip-Flop에는 Reset의 기능이 필요하므로( Up Counter와 Down Counter를 한번에 구현해야 하므로), Reset, Clk, t를 입력으로 하고 출력 Q를 가지는 컴포넌트로 구현했다. Reset=1이면 레지스터가 0으로 초기화되게 하고, 그렇지 않으면 Clock이 0에서 1로 변할 때 T=1이면 출력값이 반전되도록 한다.



Up과 Down의 경우 4개의 T FF가 있으므로 총 8번의 컴포넌트 매핑이 필요하다. 특히 플립플롭이 뒤로 갈수록 그것의 입력값 T가 그 이전의 플립플롭에서의 출력값과 입력값들의 AND 조합들로 이루어진다. 모든 매핑에서 Clk과 Q는 key(0)과 고유한 시그널 하나씩만 대응시키면 된다.

2.(B) Timing Diagram



Sw(0) (Enable)은 1로, Sw(1) (Reset)은 0으로 고정시킨 후, Key(0) (Clock)를 반복해서 누르는 형태로 설정하고 ledr(3~0)과 ledg(3~0)의 Timing Wave를 관찰한다. Ledr(3~0)은 0부터 15까지 증가하는 것을 알 수 있는데, ledr과 ledg의 index는 2^index값을 나타내기 때문이다. 예를 들어, Ledr(3~0)=”1101”이면 2^3+2^2+2^0=13을 나타낸다. Ledr(3~0)은 Clock을 반복할 때마다 0000, 0001,0010,0011,0100,0101… 순서로, Ledg(3~0)은 1111,1110,1101,1100… 의 순서로 진행하므로 Ledr은 0부터 15까지의 Up counter, Ledg는 15부터 0까지의 Down Counter로 잘 구현되었음을 확인할 수 있다.