

Digital System Assignment 3

- FSM (Tail Light) -

Tail Light 제어기 설계

■ 목표

- FSM(finite state machine)을 사용해 자동차 후미등을 제어한다.
- 최소 개의 state을 사용한다.

■ 입력

- CLK
- Reset
- Left, Right, Break

■ 출력

- 8-bit LED

Tail Light 제어기 설계

■ Tail Light 제어기의 동작

- Reset을 제외한 각각의 동작은 CLK의 Rising Edge에 맞춰(synchronous) 동작한다.
- Reset : 0이 입력되면 활성화(Active low)
- Break, Left, Right : 1이 입력되면 활성화(Active High)

■ Reset

- Reset은 CLK에 관계없이(asynchronous) 동작한다
- Reset이 입력되면 Left, Right, Break에 관계없이 Reset 상태에 머무른다
- Reset이 입력되지 않으면 Ready 상태가 된다.

■ Ready

- Break가 입력되면 Left, Right에 관계없이 Break 상태가 된다
- Break, Right가 입력되지 않은 상태로 Left가 입력되면 Left 상태가 된다
- Break, Left가 입력되지 않은 상태로 Right가 입력되면 Right 상태가 된다
- Break가 입력되지 않은 상태로 Right, Left가 모두 입력되면 Ready 상태에 머무른다

■ Break

- Break가 입력되면 Left, Right에 관계없이 Break 상태에 머무른다
- Break가 입력되지 않으면 Ready 상태가 된다

Tail Light 제어기 설계

■ Left

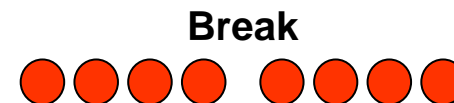
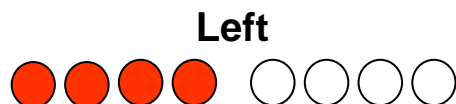
- Break가 입력되면 Break 상태로 간다
- Break가 입력되지 않고, Left가 입력되면 Left 상태에 머무른다
- Break가 입력되지 않고, Left가 입력되지 않으면 Ready 상태로 간다

■ Right

- Break가 입력되면 Break 상태로 간다
- Break가 입력되지 않고, Right가 입력되면 Right 상태에 머무른다
- Break가 입력되지 않고, Right가 입력되지 않으면 Ready 상태로 간다

■ LED 동작은 아래의 그림과 같다

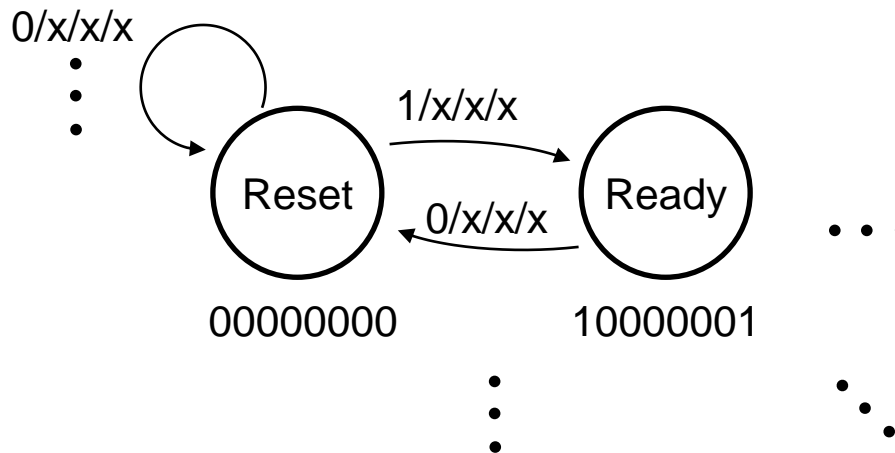
- 꺼진게 0, 켜진게 1



State Diagram

■ 상태기계

- 아래의 예시를 참고해 무어 상태기계를 각각 완성하고 상태도와 상태표를 보고서에 추가
 - ❖ 상태의 개수가 최소가 되도록 설계
 - ❖ 상태의 이름을 적절히 정하고, 필요하다면 설명
- Moore 기계 상태도 예시
 - ❖ 입력은 Reset, Break, Left, Right 순서
 - ❖ 출력은 8비트의 LED



State Transition Table

- Moore 기계 상태표 예시
 - Reset, Break, Left, Right

Current State	입력	Next State	Output (8 bit LED)
Reset	0/x/x/x	Reset	0000 0000
	1/x/x/x	Ready	
⋮			⋮
	⋮	⋮	
Left	0/x/x/x	Reset	1111 0000
	1/0/1/0	Left	
	⋮	⋮	
⋮			⋮

Assignment 4

■ Entity

- 아래의 Entity를 그대로 사용하라


```
library IEEE;
use IEEE.std_logic_1164.ALL;
use IEEE.std_logic_ARITH.ALL;
use IEEE.std_logic_UNSIGNED.ALL;


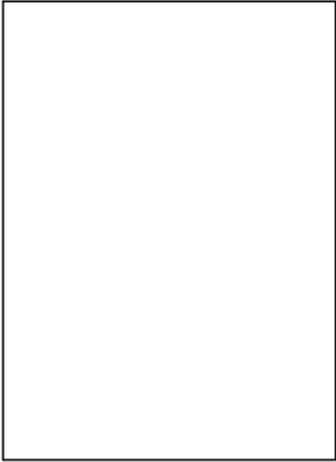
entity TAIL_LIGHT is
  port(
    CLK : in std_logic;
    RESET : in std_logic;
    BREAK : in std_logic;
    LEFT : in std_logic;
    RIGHT : in std_logic;
    LED : out std_logic_vector(7 downto 0)
  );
end TAIL_LIGHT;
```

Assignment 4

- Moore Machine Source Code 예시
 - 되도록 Enumeration Types을 사용

```
architecture Behavioral of TAIL_LIGHT is
    signal state : std_logic_vector(2 downto 0);
    signal next_state : std_logic_vector(2 downto 0);
begin
    -- state transition
    process(CLK, RESET)
    begin
        if(RESET = '0') then
            state <= "000";
        elsif(CLK = '1' and CLK'event) then
            state <= next_state;
        end if;
    end process;
```

```
-- led out
process(state)
begin
    if (state = "000") then
        LED <= "00000000";
    elsif (state = "001") then
        
    end if;
end process;
```

```
-- next state
process(RESET, BREAK, LEFT, RIGHT, state)
begin
    -- reset state
    if (state = "000") then
        
    -- ready state
    elsif (state = "001") then
        
    end if;
end process;
```


Assignment 4

■ Testbench

- 아래의 Testbench를 그대로 사용하고, Waveform 제출

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity TL_tb is
end TL_tb;

architecture Behavioral of TL_tb is
    component TAIL_LIGHT is
        port(
            CLK : in std_logic;
            RESET : in std_logic;
            BREAK : in std_logic;
            LEFT : in std_logic;
            RIGHT : in std_logic;
            LED : out std_logic_vector(7 downto 0)
        );
    end component;

    signal CLK : std_logic;
    signal RESET : std_logic;
    signal BREAK : std_logic;
    signal LEFT : std_logic;
    signal RIGHT : std_logic;
    signal LED : std_logic_vector(7 downto 0);

```

Assignment 4

■ Testbench

- 아래의 Testbench를 그대로 사용하고, Waveform 제출

```
begin
  t_l : TAIL_LIGHT port map (CLK => CLK, RESET => RESET, BREAK => BREAK, LEFT => LEFT, RIGHT => RIGHT, LED => LED);

  rst_t : process
  begin
    wait for 10 ns;
    RESET <= '0';
    wait for 20 ns;
    RESET <= '1';
    wait;
  end process;

  clk_t : process
  begin
    wait for 5 ns;
    CLK <= '1';
    wait for 5 ns;
    CLK <= '0';
  end process;
```

Assignment 4

■ Testbench

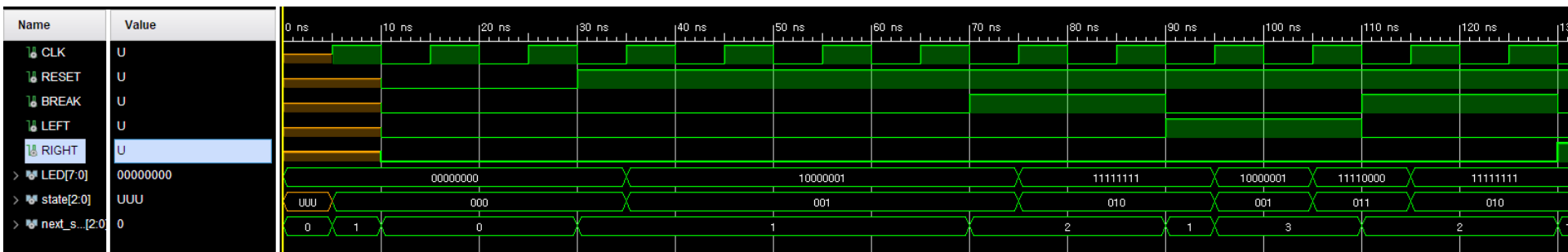
- 아래의 Testbench를 그대로 사용하고, Waveform 제출

```
tl_t : process
begin
    wait for 10ns;
    BREAK <= '0';
    LEFT <= '0';
    RIGHT <= '0';
    wait for 60 ns;
    BREAK <= '1';
    wait for 20 ns;
    BREAK <= '0';
    LEFT <= '1';
    wait for 20 ns;
    BREAK <= '1';
    LEFT <= '0';
    wait for 20 ns;
    BREAK <= '0';
    RIGHT <= '1';
    wait for 20 ns;
    BREAK <= '1';
    RIGHT <= '0';
    wait;

end process;
end Behavioral;
```

Tail Light 제어기 설계

■ Simulation Waveform (예시)



Homework

■ 제출

- 제출 기한 : 5/28 (화) 23:00
- i-campus에 업로드
- Late Penalty
 - ❖ 1s ~ 1 day : -10%
 - ❖ 1day ~ 2day : -30%
 - ❖ 2day ~ 3day : -50%
 - ❖ 3day ~ : Cannot submit

■ 과제 관련 문의 : 고준성 조교(myun243@skku.edu)

Assignment

■ 제출

- 워드 프로그램에 VHDL 코드와 waveform을 첨부한 파일
 - ❖ 빈칸을 완성한 VHDL 코드와 설명
 - 직접 작성한 부분에서 핵심이 되는 내용만 주석으로 설명
 - 필요하다면 코드를 추가
 - VHDL 코드 설계를 위해 사용한 이론, 과정 등의 Effort 설명
 - ❖ 시뮬레이션 코드, 결과 waveform과 설명
 - 시뮬레이션 코드는 위의 testbench code를 사용
 - waveform에 관련된 설명
 - ❖ 파일명 : 학번_이름.xxx
 - ❖ 한컴오피스 또는 MS Word에서 열 수 없는 확장자, 파일은 미제출 처리
- 직접 작성한 VHDL 파일 제출
 - ❖ Source code 파일명 : 학번_이름.vhd
 - ❖ Testbench code 파일명 : 학번_이름_tb.vhd
- 하나의 파일로 압축해 제출