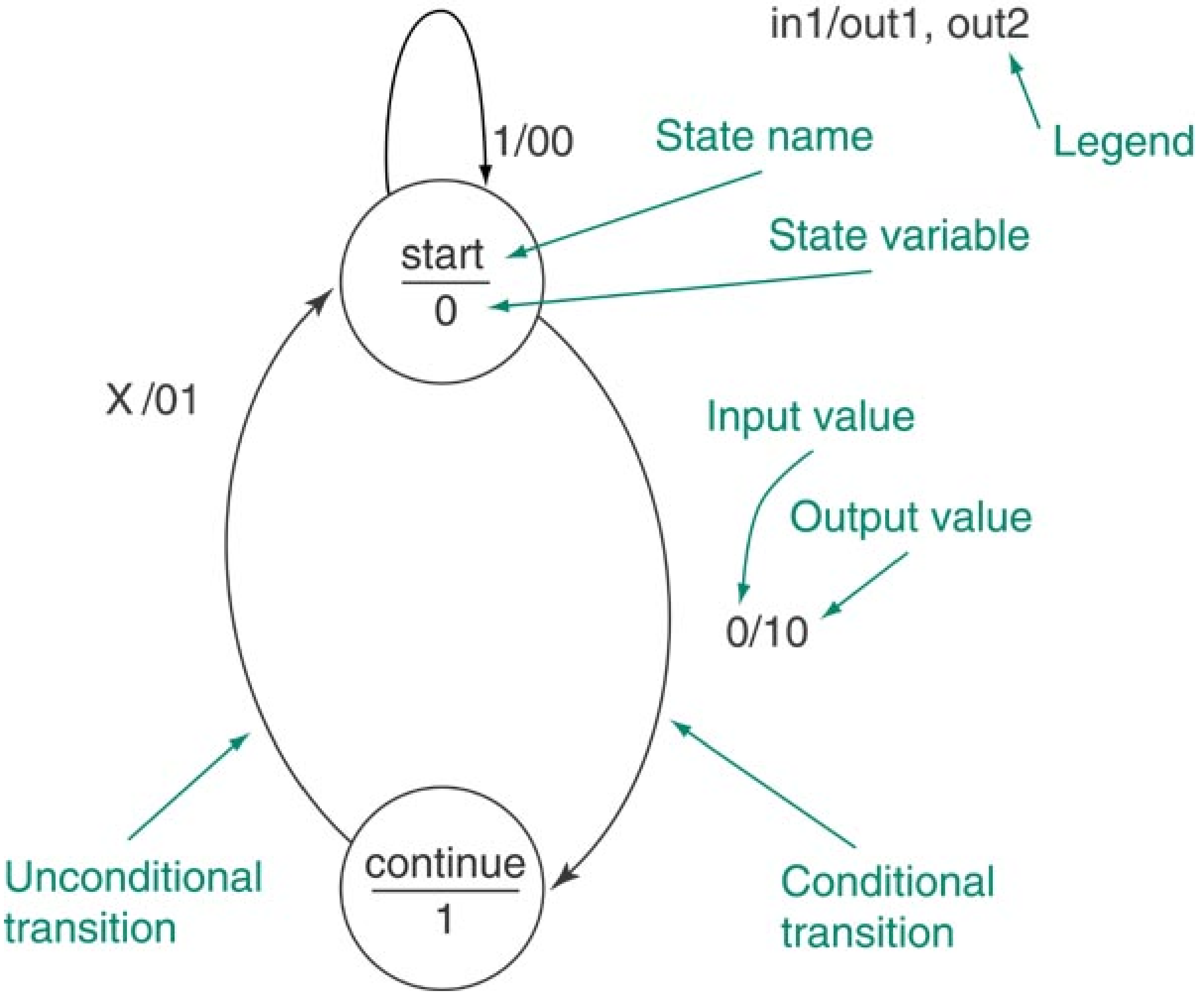
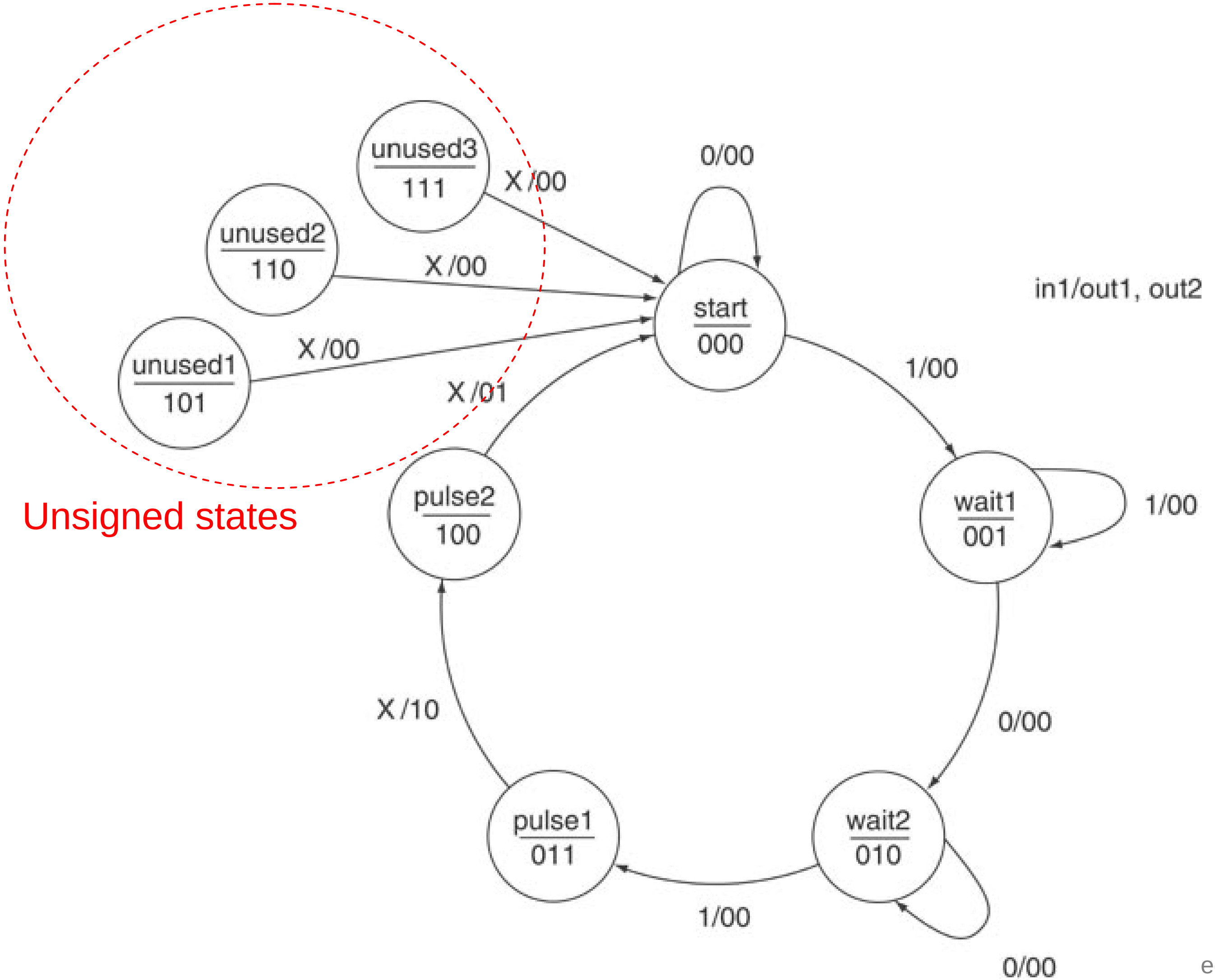
Class 12
State Machine

State Machine



Unused States in State Machines



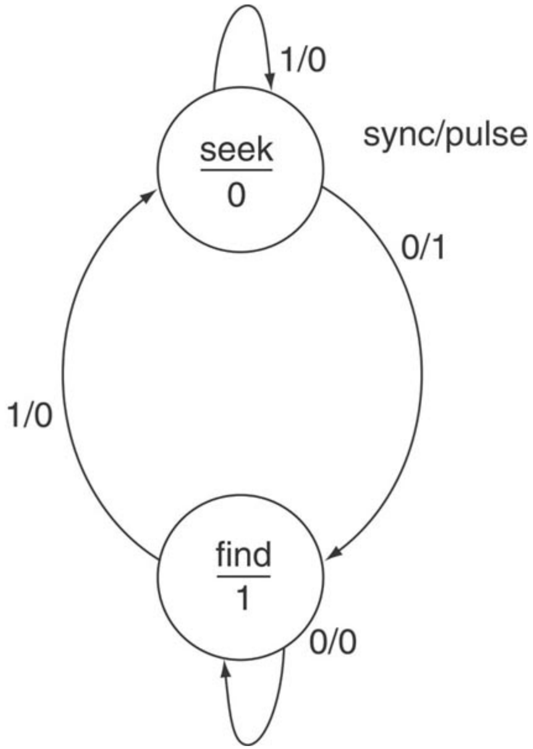
Unsigned states

State Machine (Cont.)

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY sngl_pls IS
 PORT(
  clk, sync: IN STD_LOGIC;
  pulse: OUT STD_LOGIC);
END sngl_pls;

ARCHITECTURE pulser OF sngl_pls IS
 TYPE PULSE_STATE IS (seek, find);
 SIGNAL status: PULSE_STATE;
BEGIN
 PROCESS (clk)
 BEGIN
  IF (clk'EVENT and clk = '1') THEN
   CASE status IS
    WHEN seek =>
      IF (sync = '1') THEN
       status <= seek;
       pulse <= '0';
      ELSE
       status <= find;
       pulse <= '1';
      END IF;
    WHEN find =>
      IF (sync = '1') THEN
        status <= seek;
        pulse <= '0';
      ELSE
        status <= find;
        pulse<= '0';
      END IF;
   END CASE;
  END IF;
 END PROCESS;
END pulser;
```

Type
enumeration



Push Button Debouncer

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;

ENTITY pbdebouncer IS
  PORT(
    clk, pb2: IN STD_LOGIC;
    Hex0: OUT STD_LOGIC_VECTOR(0 to 7));
END pbdebouncer;

ARCHITECTURE a  OF pbdebouncer IS
  CONSTANT TicksPerMilliSecond: INTEGER := 50000;
  CONSTANT DebounceTime: INTEGER := TicksPerMilliSecond*5;
  SIGNAL PressedTime: NATURAL := 0;
  SIGNAL cnt: NATURAL RANGE 0 to 9;
BEGIN
  PROCESS (clk)
  BEGIN
    -- Change the shifting rate
    IF(clk'EVENT and clk = '1') THEN
      IF( pb2='0‘ ) THEN -- Push button is pressed
        IF(PressedTime < DebounceTime) THEN -- Wait for debounce time
          PressedTime <= PressedTime + 1;
        ELSIF(PressedTime = DebounceTime) THEN -- Debounce time is reached
          PressedTime <= DebounceTime + 1; -- stop counting
          cnt <= cnt + 1;
          IF(cnt >=9) THEN
            cnt <= 0;
          END IF;
        END IF;
      ELSE
        PressedTime <= 0;
      END IF;
    END IF;
```

Define constants to debounce 5ms

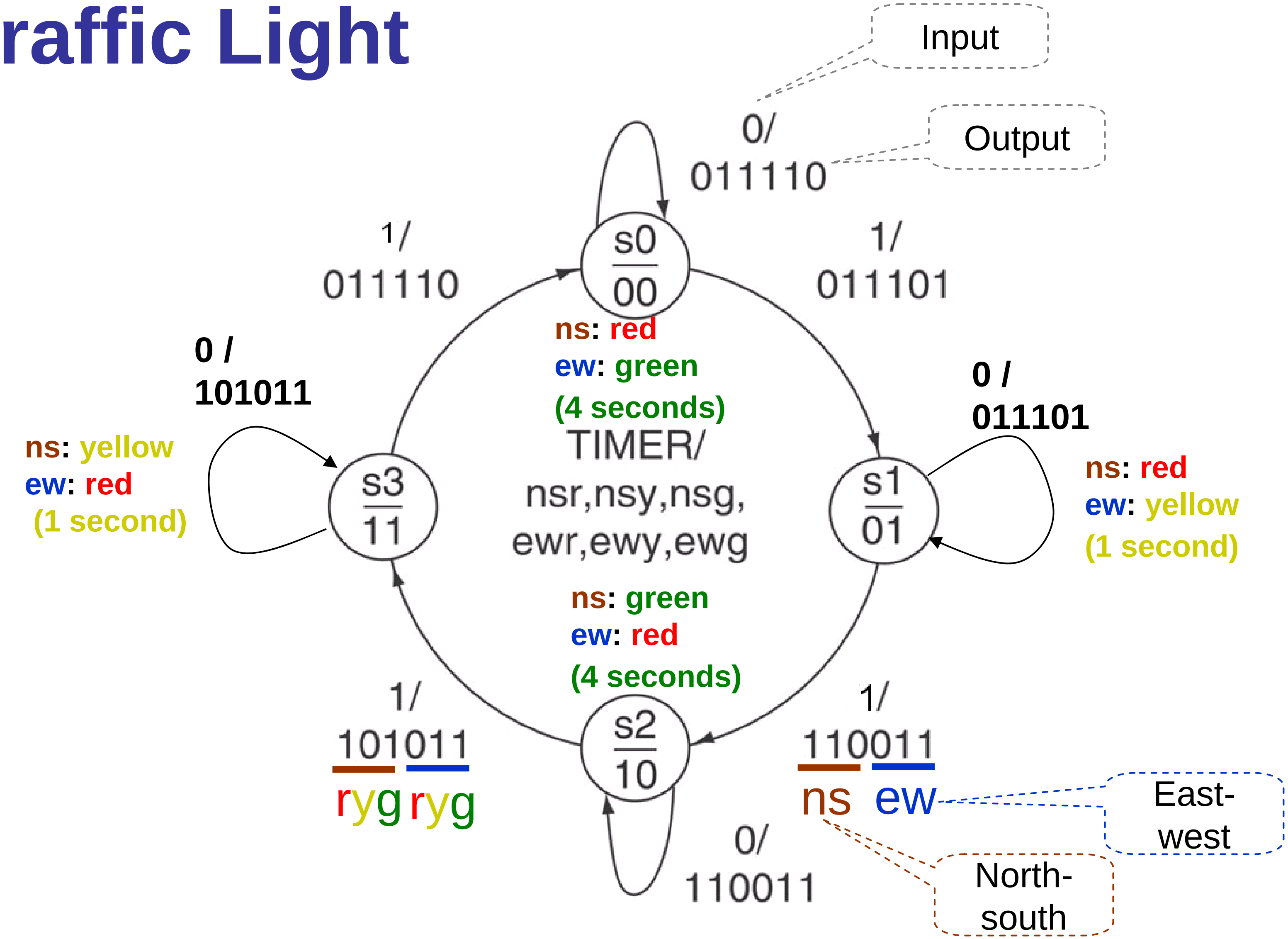
Equals to:
1. cnt <= (cnt + 1) mod 10;
2. cnt - (cnt / 10) * 10;

Push button is not pressed

```
      -- Display the counter
      CASE cnt IS
        WHEN 0 => Hex0 <= x"03"; -- 0
        WHEN 1 => Hex0 <= x"9F"; -- 1
        WHEN 2 => Hex0 <= x"25"; -- 2
        WHEN 3 => Hex0 <= x"0D"; -- 3
        WHEN 4 => Hex0 <= x"99"; -- 4
        WHEN 5 => Hex0 <= x"49"; -- 5
        WHEN 6 => Hex0 <= x"C1"; -- 6
        WHEN 7 => Hex0 <= x"1F"; -- 7
        WHEN 8 => Hex0 <= x"01"; -- 8
        WHEN 9 => Hex0 <= x"19"; -- 9
        WHEN others => Hex0 <= x"FF"; -- blank
      END CASE;
    END PROCESS;
END a;
```

hexadecimal

Traffic Light

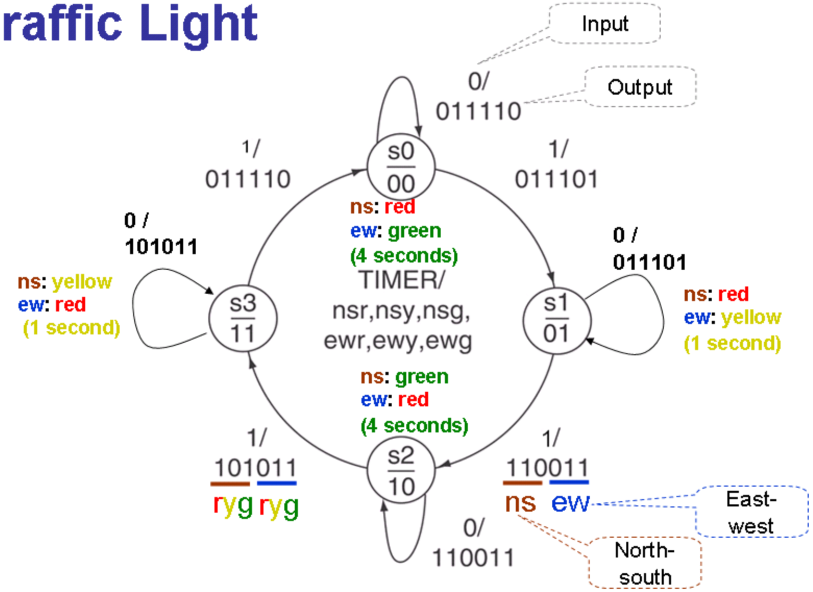


Lab 12

- Design a counter with a push button debouncer
  - Implement a two-digit counter that counts from 0 to 99.
    - Hex1 shows the digit of 10s, and hex0 shows the digit of 1s.
  - When PushButton2 is pressed the 2-digit counter is advanced by 1.
- Design a traffic light
  - Red light time is 5s, green light time is 4s, and yellow light time is 1s.
  - Hex3(/Hex2) is on to show the remaining time of the red light of the north-south (/west-east) direction; otherwise, Hex3 (/Hex2) is off.
  - Initial state: **s0**

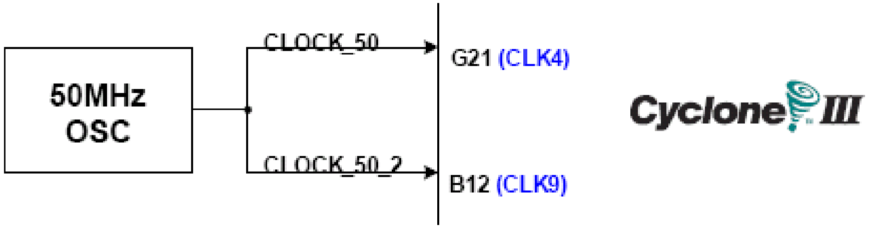
LED_9	LED_8	LED_7	LED_2	LED_1	LED_0
ns Red	ns Yellow	ns Green	ew Red	ew Yellow	ew Green
Hex3			Hex2		

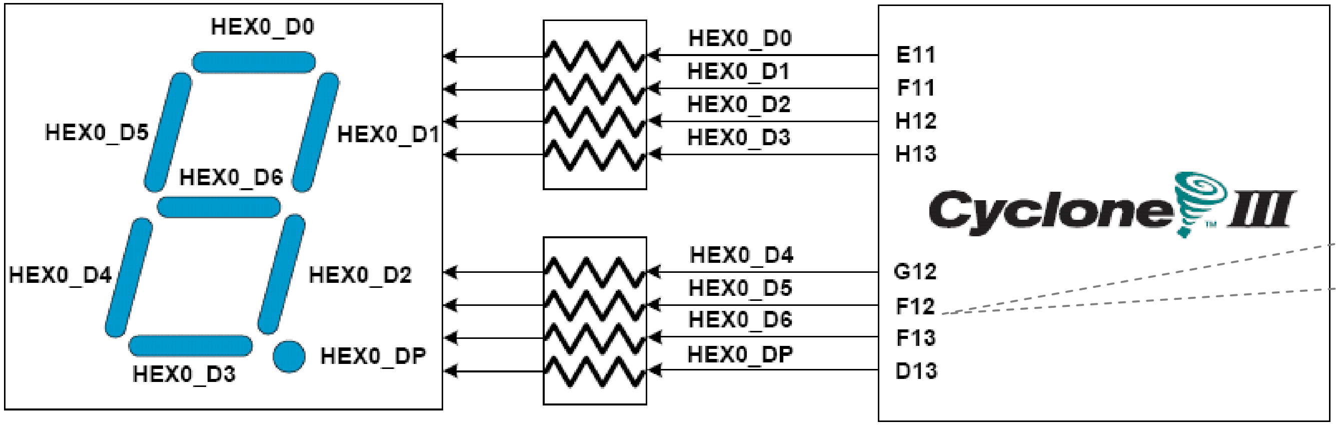
Traffic Light



- Report:
  - Write down what you have learned from this lab. (實驗心得)

7-Segment Displays & DE0 – External Clock





Pin number (active-low)

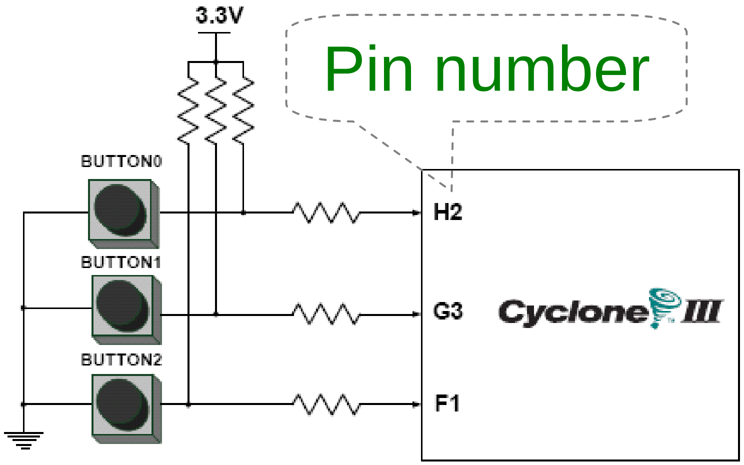
Signal Name	FPGA Pin No.
HEX0_D[0]	PIN_E11
HEX0_D[1]	PIN_F11
HEX0_D[2]	PIN_H12
HEX0_D[3]	PIN_H13
HEX0_D[4]	PIN_G12
HEX0_D[5]	PIN_F12
HEX0_D[6]	PIN_F13
HEX0_DP	PIN_D13

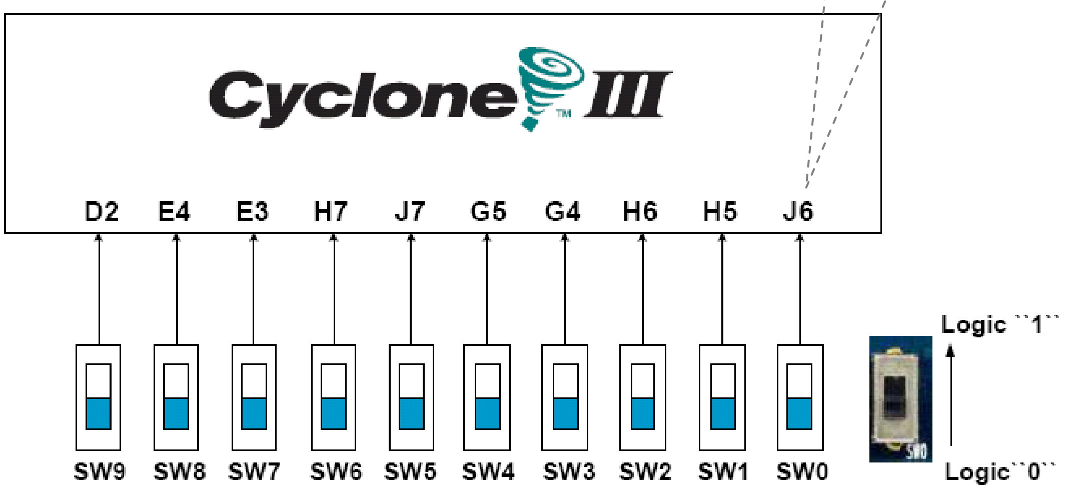
HEX1_D[0]	PIN_A13
HEX1_D[1]	PIN_B13
HEX1_D[2]	PIN_C13
HEX1_D[3]	PIN_A14
HEX1_D[4]	PIN_B14
HEX1_D[5]	PIN_E14
HEX1_D[6]	PIN_A15
HEX1_DP	PIN_B15

HEX2_D[0]	PIN_D15
HEX2_D[1]	PIN_A16
HEX2_D[2]	PIN_B16
HEX2_D[3]	PIN_E15
HEX2_D[4]	PIN_A17
HEX2_D[5]	PIN_B17
HEX2_D[6]	PIN_F14
HEX2_DP	PIN_A18

HEX3_D[0]	PIN_B18
HEX3_D[1]	PIN_F15
HEX3_D[2]	PIN_A19
HEX3_D[3]	PIN_B19
HEX3_D[4]	PIN_C19
HEX3_D[5]	PIN_D19
HEX3_D[6]	PIN_G15
HEX3_DP	PIN_G16

Pushbutton and Slide Switches





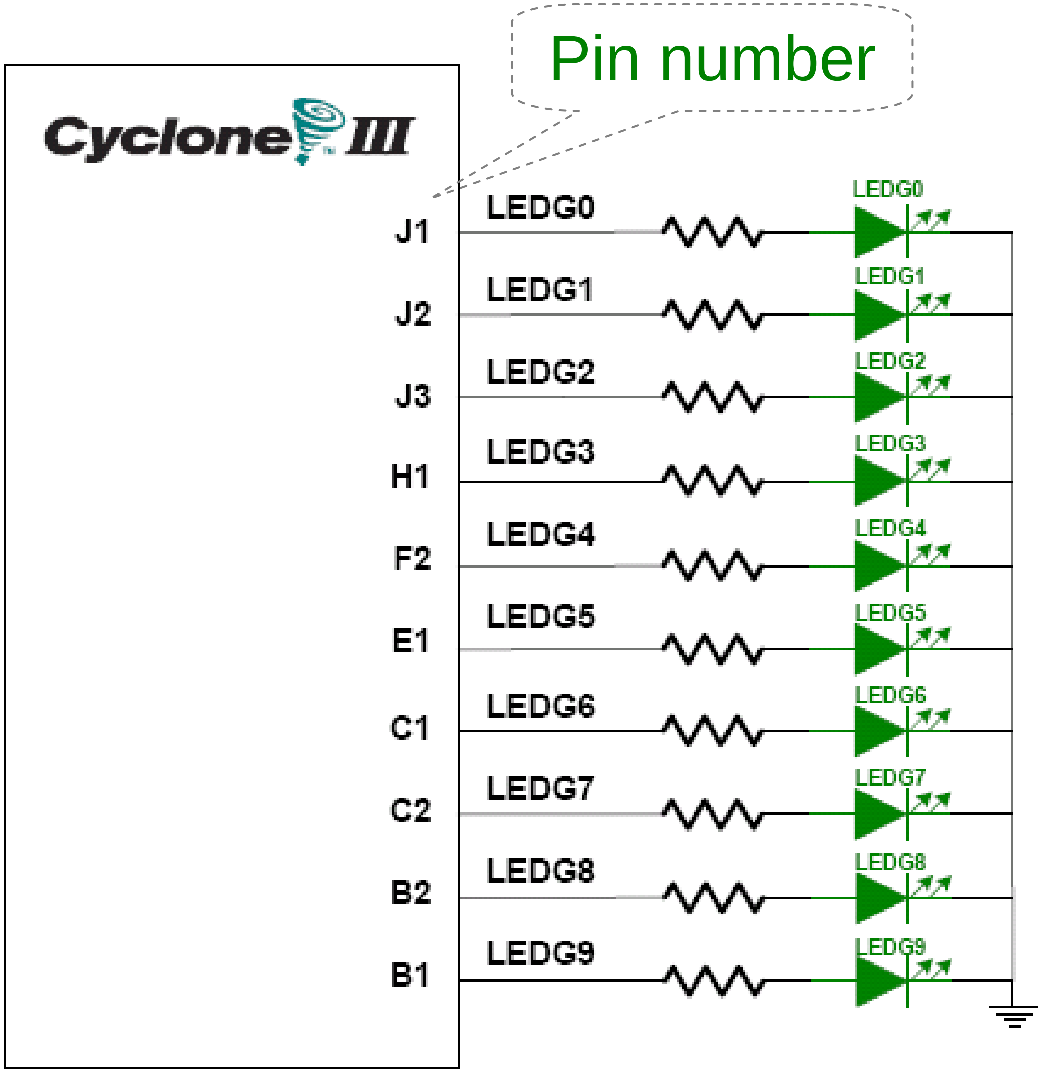
3 Pushbutton switches:
Not pressed → Logic High
Pressed → Logic Low

Signal Name	FPGA Pin No.
BUTTON [0]	PIN_ H2
BUTTON [1]	PIN_ G3
BUTTON [2]	PIN_ F1

10 Slide switches (Sliders):
Up → Logic High
Down → Logic

SW[0]	PIN_J6	SW[5]	PIN_J7
SW[1]	PIN_H5	SW[6]	PIN_H7
SW[2]	PIN_H6	SW[7]	PIN_E3
SW[3]	PIN_G4	SW[8]	PIN_E4
SW[4]	PIN_G5	SW[9]	PIN_D2

LEDs


Pin number

10 LEDs
Opuput high → LED on
Output low → LED off

Signal Name	FPGA Pin No.
LEDG[0]	PIN_J1
LEDG[1]	PIN_J2
LEDG[2]	PIN_J3
LEDG[3]	PIN_H1
LEDG[4]	PIN_F2
LEDG[5]	PIN_E1
LEDG[6]	PIN_C1
LEDG[7]	PIN_C2
LEDG[8]	PIN_B2
LEDG[9]	PIN_B1