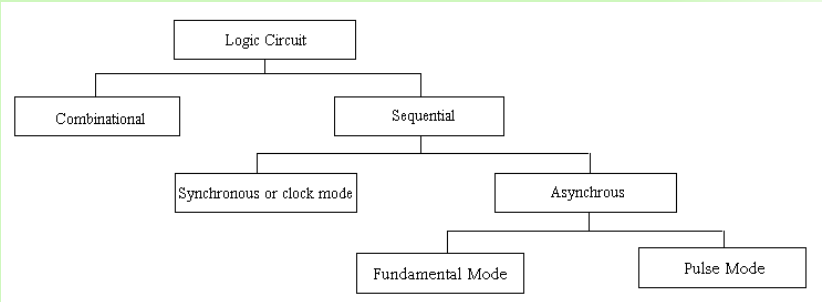
การออกแบบวงจรซีเควนเชียลด้วย VHDL

- วงจรซีเควนเชียล (Sequential Logic Circuit)
- สเตทแมชชีน (State Machine)

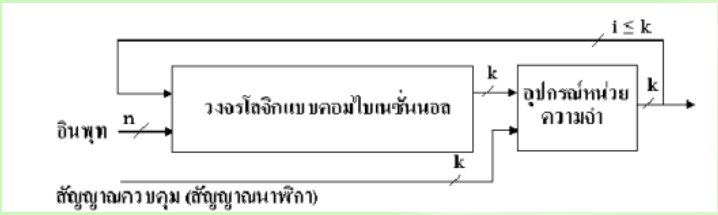
1

บทนำ



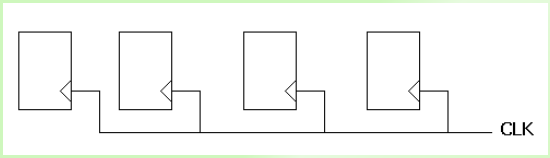
2

วงจรซิงโครนัส



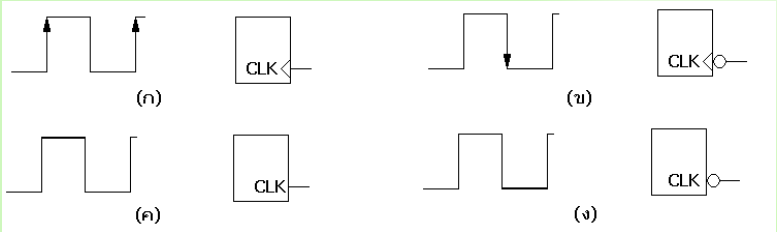
3

การต่อสัญญาณนาฬิกาของวงจรซีเควนเชียลแบบซิงโครนัส



4

ลักษณะของสัญญาณนาฬิกา

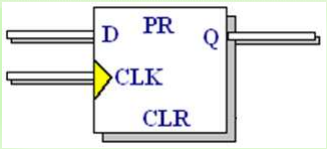


(ก) ขอบบวก (ข) ขอบลบ

(ค) โลจิก 1 และ (ง) โลจิก 0

5

D ฟลิบฟลอบทำงานด้วยสัญญาณนาฬิกาขอบบวก

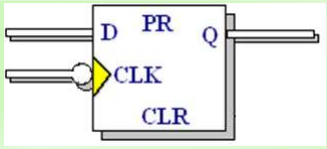



```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
 entity D_Flipflop is
   Port (D : in std_logic;
         CLK : in std_logic;
           Q : Buffer std_logic);
end D_Flipflop;

architecture Behavioral of D_Flipflop is
begin
          PROCESS (clk)
          BEGIN
                    IF(CLK'EVENT and CLK = '1') THEN
                              Q <= D;
                    ELSE
                              Q <= Q;
                    END IF;
          END PROCESS;
end Behavioral;
```

6

D ฟลิบฟลอบทำงานด้วยสัญญาณนาฬิกาขอบลบ

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity D_Flipflop is
   Port (D : in std_logic;
         CLK : in std_logic;
          Q : Buffer std_logic);
end D_Flipflop;

architecture Behavioral of D_Flipflop is
begin
        PROCESS (clk)
        BEGIN
                IF(CLK'EVENT and CLK = '0') THEN
                        Q <= D;
                ELSE
                        Q <= Q;
                END IF;
        END PROCESS;
end Behavioral;
```

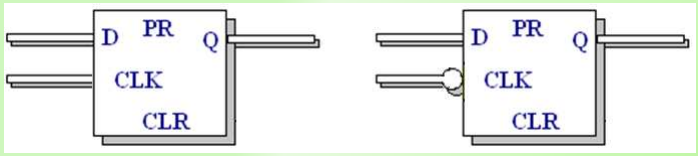


7

D ฟลิบฟลอบทำงานด้วยสัญญาณนาฬิกาโลจิก 1และ 0

```
PROCESS (clk, D)
BEGIN
        IF(CLK = '1') THEN
                Q <= D;
        ELSE
                Q <= Q;
        END IF;
END PROCESS;
```

```
PROCESS (clk, D)
BEGIN
        IF(CLK = '0') THEN
                Q <= D;
        ELSE
                Q <= Q;
        END IF;
END PROCESS;
```



8

การออกแบบวงจรซีเควนเชียลด้วย

ขั้นที่ 1. กำหนดพฤติกรรม หรือการทำงานของระบบ
ขั้นที่ 2. เขียนขอบเขตของงานออกมาในรูปของตาราง State diagram
ขั้นที่ 3. กำหนดจำนวนฟลิปฟลอปและตัวแปรสถานะ (ฟลิปฟลอปหนึ่งตัวใช้ตัวแปรหนึ่งตัว) ใช้รหัสหนึ่งต่อหนึ่งสถานะ
ขั้นที่ 4. เลือกชนิดของฟลิปฟลอป กำหนดอินพุท และสมการเอาท์พุทแบบมัวร์และ/หรือเมียลี
ขั้นที่ 5. เขียนวงจรโลจิก

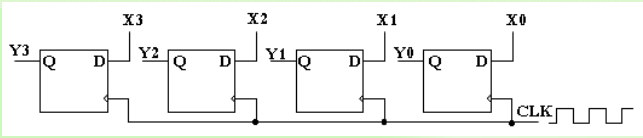
9

การออกแบบวงจรซีเควนเชียลด้วย VHDL

ขั้นที่ 1. กำหนดพฤติกรรม หรือการทำงานของระบบ
ขั้นที่ 2. เขียนขอบเขตของงานออกมาในรูปของตาราง State diagram
ขั้นที่ 3. กำหนดจำนวนฟลิปฟลอปและตัวแปรสถานะ (ฟลิปฟลอปหนึ่งตัวใช้ตัวแปรหนึ่งตัว) ใช้รหัสหนึ่งต่อหนึ่งสถานะ
ขั้นที่ 4. เลือกชนิดของฟลิปฟลอป กำหนดอินพุท และสมการเอาท์พุทแบบมัวร์และ/หรือเมียลี
ขั้นที่ 5. เขียนวงจรโลจิก

10

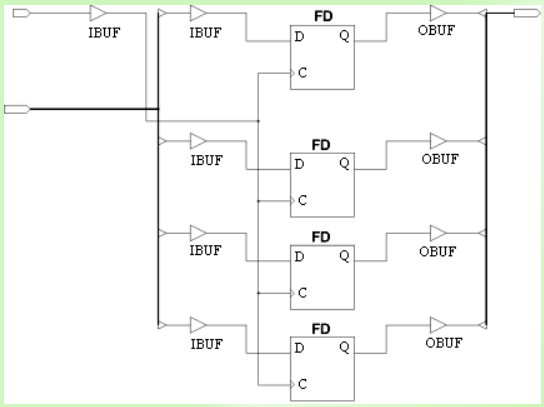
รีจิสเตอร์ (Register)




```
entity buf_reg is
   Port ( clk : in  STD_LOGIC;
          x : in STD_LOGIC_vector(3 downto 0);
          y : out STD_LOGIC_vector(3 downto 0));
end buf_reg;
architecture Behavioral of buf_reg is
begin
        process (clk)
        begin
                if CLK='1' and CLK'event then
                        y <= x;
                end if;
        end process;
end Behavioral;
```

11

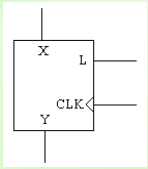
โลจิกไดอะแกรมวงจรบัฟเฟอร์รีจิสเตอร์ที่ได้จากการสังเคราะห์



12

บัฟเฟอร์รีจิสเตอร์ที่มีการควบคุม

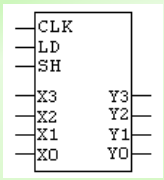
```
entity cont_reg is
   Port ( clk : in  STD_LOGIC;
                l : in  STD_LOGIC;
               x : in STD_LOGIC_vector(3 downto 0);
               y : inout STD_LOGIC_vector(3 downto 0));
end cont_reg;
architecture Behavioral of cont_reg is
begin
            process (clk)
            begin
                        if CLK='1' and CLK'event then
                                    if l='1' then
                                                y <= x;
                                    else
                                                y <= y;
                                    end if;
                        end if;
            end process;
end Behavioral;
```



13

ชิฟท์รีจิสเตอร์ (Shift Register)

```
entity shift_reg is
   Port ( clk : in  STD_LOGIC;
             ld : in  STD_LOGIC;
            sh : in  STD_LOGIC;
             x : in STD_LOGIC_vector(3 downto 0);
             y : inout STD_LOGIC_vector(3 downto 0));
end shift_reg;
```



14

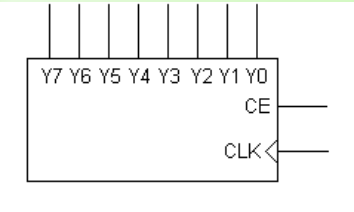
```
architecture Behavioral of shift_reg is

begin
        process (clk)
        begin
                if CLK='1' and CLK'event then
                        if ld='1' then
                                y <= x;
                        else
                                if sh='1' then
                                        y(3 downto 1)<=y(2 downto 0);
                                        y(0) <= x(0);
                                else
                                        y <= y;
                                end if;
                        end if;
                end if;
        end process;
end Behavioral;
```

15

วงจรนับ

เขียนจาก พฤติกรรม หรือการทำงาน

```
entity coup is
    port(CLK: in STD_LOGIC;
         CE: in STD_LOGIC;
         y: inout INTEGER range 255 downto 0);
end coup;
architecture Behavioral of coup is
begin
process (CLK)
begin
  if CLK='1' and CLK'event then
    if CE = '1' then
      y <= y + 1;
    else
      y <= y;
    end if;
  end if;
end process;
end Behavioral;
```



16

โมดูล int2bit8

กรณีที่ใช้การแปลงข้อมูล

```
entity int2bit8 is
   port (x : in integer range 255 downto 0;
         z : out std_logic_vector(7 downto 0));
end int2bit8;
architecture beh of int2bit8 is
begin
  process(x)
  variable i : integer range 0 to 7;
  begin
    for i in 0 to 7 loop
      if ((x/(2**i) ) mod 2 = 1) then
         z(i) <= '1';
      else
         z(i) <= '0';
                end if;
    end loop;
  end process;
end beh;
```

17

กรณีที่ใช้การแปลงข้อมูล

```
library IEEE;
use IEEE.std_logic_1164.all;
entity coup2 is
    port (pclk, pce : in std_logic;
                py : out  std_logic_vector (7 downto 0));
end coup2;
 architecture coup2_arch of coup2 is
   component int2bit8
       port (x : in integer range 255 downto 0;
       z : out std_logic_vector(7 downto 0));
   end component;
   component coup
      port(CLK, CE: in STD_LOGIC;
           y: inout integer range 255 downto 0);
   end component;
signal bus1 : INTEGER range 255 downto 0;
begin
  c1: coup port map(pclk, pce, bus1);
  c2: int2bit8 port map(bus1, py);
end coup2_arch;
```

18

วงจรนับเลขฐานสิบขนาด 1 หลัก


```
entity bcd1 is
   Port ( clk : in  STD_LOGIC;
          reset : in  STD_LOGIC;
          y : out  STD_LOGIC_VECTOR (3 downto 0));

end bcd1;
architecture Behavioral of bcd1 is
signal ya : INTEGER range 0 to 15;
component int2std_logic is
  port (bin : in integer range 0 to 15 ;
                y : out std_logic_vector(3 downto 0));
end component;
```

19

```
begin
        process (clk, reset)            begin
                if Reset='0' then
                        ya <= 0;
                else
                        if CLK='1' and CLK'event then
                                if ya >= 9 then
                                        ya <= 0;
                                else
                                        ya <= ya + 1;
                                end if;
                        else
                                ya <= ya;
                        end if;
                end if;
        end process;
        c1: int2std_logic port map(ya, y);
end Behavioral;
```

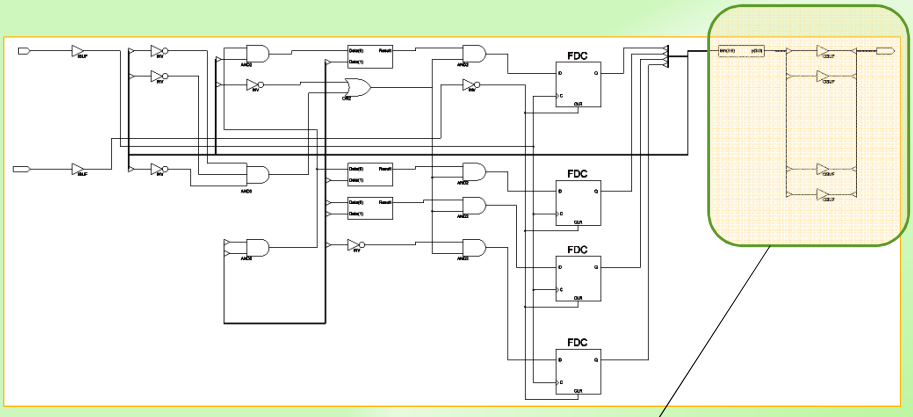
20

โมดูลแปลงสัญญาณ

```
entity int2std_logic is
  port (bin : in integer range 0 to 15 ;
                y : out std_logic_vector(3 downto 0));
end int2std_logic;
architecture Behavioral of int2std_logic is
        begin
                int2bv:process(bin)
                begin
                        case bin is
                                when 0 => y <= "0000";
                                when 1 => y <= "0001";
                                when 2 => y <= "0010";
                                when 3 => y <= "0011";
                                when 4 => y <= "0100";
                                when 5 => y <= "0101";
                                when 6 => y <= "0110";
                                when 7 => y <= "0111";
                                when 8 => y <= "1000";
                                when 9 => y <= "1001";
                                when 10 => y <= "1010";
                                when 11 => y <= "1011";
                                when 12 => y <= "1100";
                                when 13 => y <= "1101";
                                when 14 => y <= "1110";
                                when others => y <= "1111";
                        end case;
                end process;
end Behavioral;
```

21

โลจิกไดอะแกรมวงจรนับเลขฐานสิบที่ได้จากการสังเคราะห์



แปลงประเภทสัญญาณ

22

ไม่ใช้การแปลงสัญญาณ

```
entity bcd1 is
   Port ( clk : in  STD_LOGIC;
          reset : in  STD_LOGIC;
           ya : inout  integer range 0 to 15 );
end bcd1;
architecture Behavioral of bcd1 is
begin
          process (clk, reset)
          begin
                    if Reset='0' then
                              ya <= 0;
                    else
                              if CLK='1' and CLK'event then
                                        if ya >= 9 then
                                                  ya <= 0;
                                        else
                                                  ya <= ya + 1;
                                        end if;
                              else
                                        ya <= ya;
                              end if;
                    end if;
          end process;
end Behaviora
```

23