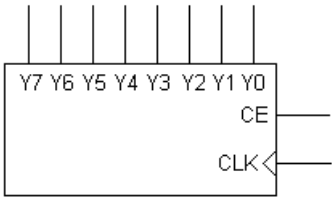
การทดลองที่ 6 วงจรนับ

แบบที่ 1 วงจรนับขึ้นที่มีสัญญาณควบคุมการนับ



Y7..0 เป็นเอาท์พุท

CE ควบคุมการนับ ถ้าเท่ากับ 1 จะนับขึ้น แต่ถ้าเท่ากับ 0 จะหยุดนับค้างค่าเดิม

CLK เป็นสัญญาณนาฬิกา

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

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;
```

ให้ออกแบบวงจรนับที่มีสัญญาณควบคุมต่อไปนี้

CE สัญญาณควบคุมการนับและหยุดนับ

DIR ควบคุมทิศทางการนับจะให้นับขึ้นหรือนับลง

CLR สัญญาณรีเซ็ท ใช้สำหรับค่าการนับเป็น 0

แบบที่ 2 ใช้การแปลงประเภทสัญญาณด้วย VHDL Module

VHDL Module สำหรับการแปลงสัญญาณจาก Integer เป็น Standard logic vector

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;
use IEEE.std_logic_arith.all;

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;
```

โมดูลหลัก

```
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;
```

แบบที่ 3 ใช้การแปลงประเภทสัญญาณด้วย การเรียกใช้ฟังก์ชั่นที่สร้างอยู่ใน Package

Package ที่สร้างขึ้นใหม่ ภายในมีเพียง 1 ฟังก์ชั่น

```
-- i2bv : Integer to Bit_vector.
-- In :  Integer, Value and width.
-- Return : std_logic_vector, with left bit is the most significant bit.
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
package my_pack is
     function i2bv (val : integer) return std_logic_vector;
end my_pack;
package body my_pack is
     function i2bv (val : integer) return std_logic_vector is
     variable result : std_logic_vector(7 downto 0) := (others => '0');
     begin
           for i  in 0 to 7 loop
                 if ( (val/(2**i) ) mod 2 = 1) then
                       result(i) := '1';
                 end if;
           end loop;
           return (result);
     end i2bv;
end my_pack;
```

ส่งคืนค่าผลลัพธ์

โมดูลหลัก

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use work.my_pack.all;
entity coup3 is
   port(dout : inout std_logic_vector(7 downto 0);
           clk, ce  : in STD_LOGIC);
end coup3;
architecture coup3_beh of coup3 is
begin
   counter: process(clk, ce)
   variable cnum : integer range 255 downto 0;
      begin
      if clk='1' and clk'event then
            if ce = '1' then
            cnum:= cnum+1;
            end if;
      end if;
      dout <= i2bv(cnum);
      end process;
end coup3_beh;
```

เรียกใช้ฟังก์ชั่น