BCD 碼(0~9)(又稱 8421 碼)，以 4 個 bit 表示一個 10 進位的值。

BCD 碼加法器 BCD_add_1d.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity BCD_add_1d is
  port ( A,B : in    std_logic_vector(3 downto 0) ;
             S      : out std_logic_vector(3 downto 0) ;
             Co     : out std_logic );
end BCD_add_1d ;
--******************************
architecture A_arith of BCD_add_1d is
  signal Temp : std_logic_vector(4 downto 0) ;
begin
process(A,B)
   begin
      Temp <= ('0'&A)+B ;
      if (Temp(3 downto 0)>9) OR (Temp(4)='1') then
-- if (Temp(4 downto 0)>9) then
            S <= Temp(3 downto 0)+6 ;
            Co <= '1';
      else
            S <= Temp(3 downto 0);
            Co <= '0' ;
      end if;
end process ;
end A_arith ;
```

當出現合超過 20 的數時會發生錯誤

例：9+12=21

21→10101 則 0101+6

\+ 0110

11011 →1011=11

所以 Co=1

$S=1011_2=11_{10}$

具有輸入偵誤功能之一位數 BCD 碼加法器

BCD_add_1d_1.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity BCD_add_1d_1 is
  port ( A,B      : in    std_logic_vector(3 downto 0) ;
             S          : out std_logic_vector(3 downto 0) ;
             Co,E     : out std_logic );
end BCD_add_1d_1 ;
--******************************
architecture A_arith of BCD_add_1d_1 is
  signal Temp : std_logic_vector(4 downto 0) ;
begin
process(A,B)
   begin
      Temp <= ('0'&A)+B ;
      if (A(3 downto 0)>9) OR (B(3 downto 0)>9) then
          S <= "ZZZZ" ;      --大寫'Z'表示高阻抗
          Co <= 'Z';
          E <= '1' ;    --錯誤輸出端設定為 1
                              --以下程式與上一程式相同
      elsif (Temp(3 downto 0)>9) OR (Temp(4)='1') then
             S <= Temp(3 downto 0)+6 ;
             Co <= '1';
             E <= '0' ;
      else
             S <= Temp(3 downto 0);
             Co <= '0' ;
             E <= '0' ;
      end if;
end process ;
end A_arith ;
```

```
if  判斷條件 1 then
            敘述區塊 1;
elsif  判斷條件 2 then
            敘述區塊 2;
elsif  判斷條件 3 then
            敘述區塊 3;
            :
else
            敘述區塊 N;
end if ;
```

一位數 BCD 碼減法器

BCD_sub_1d.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity BCD_sub_1d is
  port ( A,B : in    std_logic_vector(3 downto 0) ;
             S      : out std_logic_vector(3 downto 0) ;
             Co     : out std_logic );
end BCD_sub_1d ;
--******************************
architecture A_arith of BCD_sub_1d is
  signal Temp : std_logic_vector(4 downto 0) ;
begin
process(A,B)
   begin
      Temp <= ('0'&A)+(10-B) ;     --減數取其 10 的補數
      if (A(3 downto 0)>9) OR (B(3 downto 0)>9) then    --偵錯 A 或 B 大於 9 時
          S <= "ZZZZ" ;          --高阻抗
          Co <= 'Z';
      elsif (Temp(3 downto 0)>9) OR (Temp(4)='1') then
            S <= Temp(3 downto 0)+6 ;
            Co <= '1';        --表示 A-B 的差為正數
      else
            S <= 10-Temp(3 downto 0);   --負數再取其 10 的補數
            Co <= '0' ;       --表示 A-B 的差為負數
      end if;
end process ;
end A_arith ;
```

3*3 位元乘法器

mul_3x3.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity mul_3x3 is
  port (    A,B : in    std_logic_vector(2 downto 0);
                M     : out std_logic_vector(5 downto 0) );
end mul_3x3 ;
--******************************
architecture A_operator of mul_3x3 is
begin
       M <= A*B ;
end A_operator ;
```

when…else 敘述可針對眾多訊號做判斷比對。

With…select….when 敘述只針對某一特定訊號做判斷比對。

兩者皆是並行敘述。

BCD 碼對共陰 7 段顯示器之解碼器(1)

利用 case ….is ….when—順序性敘述(需 process)

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity BCD_to_7seg_c is
  port ( B0,B1,B2,B3      : in std_logic    ;
              Y          : out std_logic_vector(0 to 6)) ;
end BCD_to_7seg_c ;
--******************************
architecture A_case_when of BCD_to_7seg_c is
  signal Temp : std_logic_vector(3 downto 0) ;
begin
     Temp <= B3 & B2 & B1 & B0 ;
   process(Temp)
   begin
       case Temp is
          when "0000"    =>       Y<= "1111110" ;
          when "0001"    =>       Y<= "0110000";
          when "0010"    =>       Y<= "1101101" ;
          when "0011"    =>       Y<= "1111001" ;
          when "0100"    =>       Y<= "0110011" ;
          when "0101"    =>       Y<= "1011011" ;
          when "0110"    =>       Y<= "0011111" ;
          when "0111"    =>       Y<= "1110000" ;
          when "1000"    =>       Y<= "1111111" ;
          when "1001"    =>       Y<= "1110011" ;
          when others    =>       Y<= "1001111" ;
       end case;
   end process;
end A_case_when    ;
```

BCD 碼對共陰 7 段顯示器之解碼器(2)

利用 with ….select ……when---並行敘述

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity BCD_to_7seg_c_2 is
  port ( BCD    :    in integer range 0 to 15 ;
              Y      : out std_logic_vector(0 to 6)) ;
end BCD_to_7seg_c_2 ;
--******************************
architecture A_with_select_when of BCD_to_7seg_c_2 is
begin
     with BCD select
     Y <= "1111110"    when 0 ,
          "0110000"    when 1 ,
          "1101101"    when 2 ,
          "1111001"    when 3 ,
          "0110011"    when 4 ,
          "1011011"    when 5 ,
          "0011111"    when 6 ,
          "1110000"    when 7 ,
          "1111111"    when 8 ,
          "1110011"    when 9 ,
          "1001111"    when 10 to 15 ;
        -- "1001111"    when others;

   end A_with_select_when    ;
```

解碼器(Decoder)

解碼器：N 個輸入端，M 個輸出端，則 $M \leq 2^N$。如 2 對 4 線解碼器 或 3 對 8 線解碼器。

輸入		輸出			
B	A	Y_0	Y_1	Y2	Y_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

Y_0=B'A'
Y_1=B'A
Y_2=BA'
Y_3=BA

二對四線解碼器 VHDL(1)
利用布林代數

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Decoder_2x4 is
  port ( A,B : in std_logic ;
              Y     : out std_logic_vector(3 downto 0)) ;
end Decoder_2x4;
--******************************
architecture A_table of Decoder_2x4 is
begin
   Y(0) <= (not A) and (not B) ;
   Y(1) <= A and (not B) ;
   Y(2) <= (not A) and    B ;
   Y(3) <= A and B ;
end A_table    ;
```

二對四線解碼器 VHDL(2)
利用典型之並行敘述 when……else

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

--******************************
entity Decoder_2x4_1 is
  port ( A,B : in std_logic ;
              Y     : out std_logic_vector(3 downto 0)) ;
end Decoder_2x4_1;
--******************************
architecture A_when_else of Decoder_2x4_1 is
signal Temp : std_logic_vector(1 downto 0) ;
begin
   Temp <= B & A ;
   Y(0) <= '1' when Temp="00" else '0' ;
   Y(1) <= '1' when Temp="01" else '0' ;
   Y(2) <= '1' when Temp="10" else '0' ;
   Y(3) <= '1' when Temp="11" else '0' ;
end A_when_else    ;
```

三對八線解碼器

真值表

輸入			輸出							
C	B	A	Y_0	Y_1	Y_2	Y_3	Y_4	Y_5	Y_6	Y_7
0	0	0	0	1	1	1	1	1	1	1
0	0	1	1	0	1	1	1	1	1	1
0	1	0	1	1	0	1	1	1	1	1
0	1	1	1	1	1	0	1	1	1	1
1	0	0	1	1	1	1	0	1	1	1
1	0	1	1	1	1	1	1	0	1	1
1	1	0	1	1	1	1	1	1	0	1
1	1	1	1	1	1	1	1	1	1	0

加入致能輸入端 EN 之 3 對 8 解碼器 VHDL(1)

利用 when …. Else---並行敘述

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Decoder3x8 is
  port ( A,B,C,EN : in    std_logic ;
                Y              : out std_logic_vector(7 downto 0));
end Decoder3x8 ;
--******************************
architecture A_when_else of Decoder3x8 is
signal Temp : std_logic_vector(2 downto 0) ;
begin
   Temp <= C&B&A ;
   Y<= "11111110" when (Temp="000" and EN='1') else
         "11111101" when (Temp="001" and EN='1') else
         "11111011" when (Temp="010" and EN='1') else
         "11110111" when (Temp="011" and EN='1') else
         "11101111" when (Temp="100" and EN='1') else
         "11011111" when (Temp="101" and EN='1') else
         "10111111" when (Temp="110" and EN='1') else
         "01111111" when (Temp="111" and EN='1') else
         "11111111" ;
end A_when_else    ;
```

加入致能輸入端 EN 之 3 對 8 解碼器 VHDL(2)

利用 with ….select ……when---並行敘述

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Decoder3x8_1 is
  port ( A,B,C,EN : in std_logic ;
                Y              : out std_logic_vector(7 downto 0)) ;
end Decoder3x8_1 ;
--******************************
architecture A_with_select_when of Decoder3x8_1 is
signal Temp : std_logic_vector(3 downto 0) ;
begin
   Temp <= EN & C & B & A ;
   with Temp select
      Y<= "11111110" when "0000" ,
            "11111101" when "0001" ,
            "11111011" when "0010" ,
            "11110111" when "0011" ,
            "11101111" when "0100" ,
            "11011111" when "0101" ,
            "10111111" when "0110" ,
            "01111111" when "0111" ,
            "11111111" when others ;
end A_with_select_when    ;
```

3 對 8 解碼器 VHDL(3)

利用 case ….is ….when—順序性敘述(需 process)

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Decoder3x8_2 is
  port ( A,B,C,EN : in    std_logic ;
              Y              : out std_logic_vector(7 downto 0)) ;
end Decoder3x8_2 ;
--******************************
architecture A_case_when of Decoder3x8_2 is
signal Temp : std_logic_vector(3 downto 0) ;
begin
   Temp <= EN & C & B & A ;
   process(Temp)--順序性敘述
   begin
      case Temp is
         when "0000"    =>    Y<= "11111110" ;
         when "0001"    =>    Y<= "11111101" ;
         when "0010"    =>    Y<= "11111011" ;
         when "0011"    =>    Y<= "11110111" ;
         when "0100"    =>    Y<= "11101111" ;
         when "0101"    =>    Y<= "11011111" ;
         when "0110"    =>    Y<= "10111111" ;
         when "0111"    =>    Y<= "01111111" ;
         when others    =>    Y<= "11111111" ;
      end case;
   end process;
end A_case_when    ;
```

3 對 8 解碼器 VHDL(4)

利用迴圈 for…in…to…generate---並行敘述

(另一種 for…in…to….loop---順序性敘述)

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Decoder3x8_3 is
  port ( A : in    integer range 0 to 7 ;
            Y : out std_logic_vector(7 downto 0)) ;
end Decoder3x8_3 ;
--******************************
architecture A_for_generate of Decoder3x8_3 is
begin
   Lop : for N in 7 downto 0 generate --標名 Lop 不可省略
            Y(N) <= '0' when A=N else '1' ;
         end generate Lop; --可省略標名
end A_for_generate    ;
```

標名

BCD 碼對共陰 7 段顯示器之解碼器(3)

利用 with ….select ……when---並行敘述

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity BCD_to_7seg_c_2 is
  port ( BCD    : in   integer range 0 to 15 ;
             Y     : out std_logic_vector(0 to 6)) ;
end BCD_to_7seg_c_2 ;
--******************************
architecture A_with_select_when of BCD_to_7seg_c_2 is
begin
     with BCD select
     Y <= "1111110"    when 0 ,
             "0110000"    when 1 ,
             "1101101"    when 2 ,
             "1111001"    when 3 ,
             "0110011"    when 4 ,
             "1011011"    when 5 ,
             "0011111"    when 6 ,
             "1110000"    when 7 ,
             "1111111"    when 8 ,
             "1110011"    when 9 ,
             "1001111"    when 10 to 15 ;
   end A_with_select_when    ;
```

編碼器(Encoder)

編碼器：與解碼器相反，一般為 2^N 輸入，N 個輸出

8 對 3 編碼器真值表

輸入								輸出		
I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	Y_2	Y_1	Y_0
0	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	0	1	0	0	0	1
0	0	0	0	0	1	0	0	0	1	0
0	0	0	0	1	0	0	0	0	1	1
0	0	0	1	0	0	0	0	1	0	0
0	0	1	0	0	0	0	0	1	0	1
0	1	0	0	0	0	0	0	1	1	0
1	0	0	0	0	0	0	0	1	1	1

$Y_0=I_1+I_3+I_5+I_7$

$Y_1=I_2+I_3+I_6+I_7$

$Y_2=I_4+I_5+I_6+I_7$

8 對 3 編碼器加致能輸入控制 VHDL(1)---布林式

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Encoder_8x3 is
  port ( EN : in    std_logic ;
         I   : in    std_logic_vector(7 downto 0) ;
         Y   : out std_logic_vector(2 downto 0)) ;
end Encoder_8x3;
--******************************
architecture A_Boolean of Encoder_8x3 is
begin
   Y(2) <= (I(7) or I(6) or I(5) or I(4)) and EN ;
   Y(1) <= (I(7) or I(6) or I(3) or I(2)) and EN ;
   Y(0) <= (I(7) or I(5) or I(3) or I(1)) and EN ;
end A_Boolean    ;
```

8 對 3 編碼器具偵錯功能 VHDL(2)

利用 when ….else

```
--******************************
entity Encoder_8x3_1 is
  port ( I    : in    std_logic_vector(7 downto 0) ;
         Y    : out std_logic_vector(2 downto 0)) ;
end Encoder_8x3_1;
--******************************
architecture A_when_else of Encoder_8x3_1 is
begin
   Y<= "000"    when    I="00000001" else
       "001"    when    I="00000010" else
       "010"    when    I="00000100" else
       "011"    when    I="00001000" else
       "100"    when    I="00010000" else
       "101"    when    I="00100000" else
       "110"    when    I="01000000" else
       "111"    when    I="10000000" else
       "ZZZ"    ;
end A_when_else    ;
```

優先編碼器:之前電路如同時有 2 個以上的輸入端被激發時會造成輸出的錯誤動作,可用具有優先次序的優先編碼器，當訊號同時被觸發時，它將會以優先權較高的輸入端為編碼的對象。

輸入								輸出		
I_7	I_6	I_5	I_4	I_3	I_2	I_1	I_0	Y_2	Y_1	Y_0
1	X	X	X	X	X	X	X	1	1	1
0	1	X	X	X	X	X	X	1	1	0
0	0	1	X	X	X	X	X	1	0	1
0	0	0	1	X	X	X	X	1	0	0
0	0	0	0	1	X	X	X	0	1	1
0	0	0	0	0	1	X	X	0	1	0
0	0	0	0	0	0	1	X	0	0	1
0	0	0	0	0	0	0	1	0	0	0

8 對 3 優先編碼器 VHDL(3)

利用 when ….else

```
--******************************
entity Encoder_8x3_p1 is
  port ( I    : in    std_logic_vector(7 downto 0);
         Y    : out std_logic_vector(2 downto 0));
end Encoder_8x3_p1 ;
--******************************
architecture A_when_else of Encoder_8x3_p1 is
begin
        Y<= "111"    when    I(7)='1' else
            "110"    when    I(6)='1' else
            "101"    when    I(5)='1' else
            "100"    when    I(4)='1' else
            "011"    when    I(3)='1' else
            "010"    when    I(2)='1' else
            "001"    when    I(1)='1' else
            "000" ;
end A_when_else ;
```

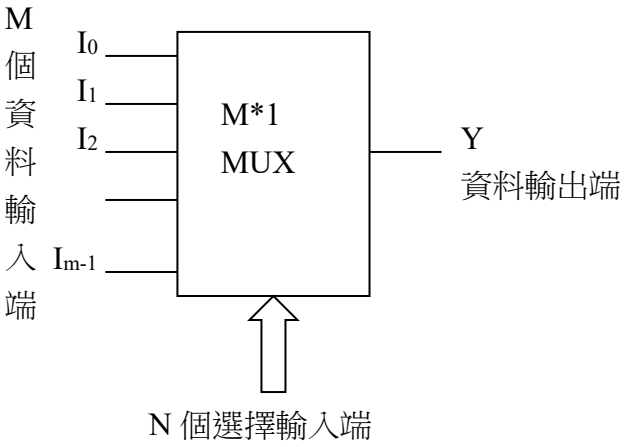
8 對 3 優先編碼器 VHDL(4)

利用 if ….then….elsif….then…. …..else

```
--******************************
entity Encoder_8x3_p2 is
  port ( I    : in    std_logic_vector(7 downto 0) ;
         Y    : out std_logic_vector(2 downto 0)) ;
end Encoder_8x3_p2 ;
--******************************
architecture A_if_then_else of Encoder_8x3_p2 is
begin
   process (I)
   begin
          if I(7)='1' then    Y<= "111" ;
          elsif I(6)='1' then    Y<= "110" ;
          elsif I(5)='1' then    Y<= "101" ;
          elsif I(4)='1' then    Y<= "100" ;
          elsif I(3)='1' then    Y<= "011" ;
          elsif I(2)='1' then    Y<= "010" ;
          elsif I(1)='1' then    Y<= "001" ;
          else    Y<= "000" ;
          end if ;
   end process;
end A_if_then_else    ;
```

多工器(Multiplexer)

簡稱 MUX，又稱資料選擇器。



四線對 1 線多工器

真值表

選擇輸入		輸出
S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

四線對 1 線多工器 VHDL(1)

```
--*****************************
entity Mux_4x1 is
  port ( I   : in   std_logic_vector(3 downto 0) ;
         S   : in   std_logic_vector(1 downto 0) ;
         Y   : out std_logic ) ;
end Mux_4x1 ;
--*****************************
architecture A_when_else of Mux_4x1 is
begin
     Y <= I(0) when S="00"   else
          I(1) when S="01"   else
          I(2) when S="10"   else
          I(3) ;
end A_when_else ;
```

四線對 1 線多工器 VHDL(2)

```
--*****************************
entity Mux_4x1_1 is
port ( I   : in   std_logic_vector(3 downto 0) ;
       S   : in   std_logic_vector(1 downto 0) ;
       Y   : out std_logic ) ;
end Mux_4x1_1;
--*****************************
architecture A_with_select of Mux_4x1_1 is
begin
   with S select
    Y <= I(0) when "00"   ,
         I(1) when "01"   ,
         I(2) when "10"   ,
         I(3) when others ;
end A_with_select ;
```

四線對 1 線多工器 VHDL(3)

```
--*****************************
entity Mux_4x1_2 is
  port ( I : in   std_logic_vector(3 downto 0) ;
         S : in   std_logic_vector(1 downto 0) ;
         Y : out std_logic ) ;
end Mux_4x1_2 ;
--*****************************
architecture A_case_is of Mux_4x1_2 is
begin
   process (S,I)
   begin
      case S is
         when "00"    =>   Y <= I(0) ;
         when "01"    =>   Y <= I(1) ;
         when "10"    =>   Y <= I(2) ;
         when others =>   Y <= I(3) ;
      end case ;
   end process ;
end A_case_is ;
```

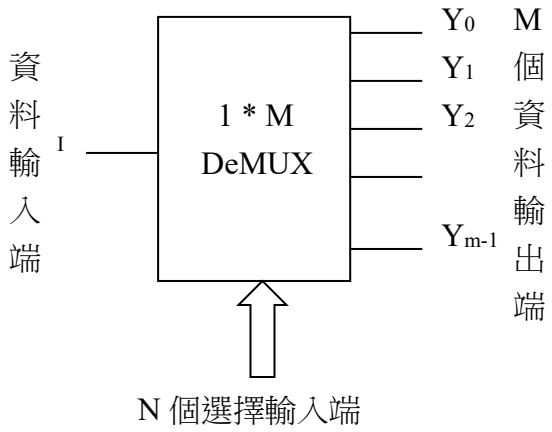
四線對 1 線多工器 VHDL(4)

```
--******************************
entity Mux_4x1_3 is
  port ( I : in    std_logic_vector(3 downto 0) ;
             S : in    std_logic_vector(1 downto 0) ;
             Y : out std_logic ) ;
end Mux_4x1_3 ;
--******************************
architecture A_if_then_else of Mux_4x1_3 is
begin
   process (S,I)
   begin
      if       S="00"    then    Y <= I(0) ;
      elsif S="01"    then    Y <= I(1) ;
      elsif S="10"    then    Y <= I(2) ;
      else    Y <= I(3)    ;
      end if;
   end process;
end A_if_then_else ;
```

四線對 1 線多工器加入閃控(致能)輸入 VHDL(5)

```
--******************************
entity Mux_4x1_4 is
  port ( I : in    std_logic_vector(3 downto 0) ;
           S : in    std_logic_vector(1 downto 0) ;
           G : in    std_logic ;
           Y : out std_logic ) ;
end Mux_4x1_4 ;
--******************************
architecture A_if_then_else of Mux_4x1_4 is
begin
   process (G,S,I)
   begin
      if       G='1'     then    Y <= 'Z' ;
      elsif S="00"    then    Y <= I(0) ;
      elsif S="01"    then    Y <= I(1) ;
      elsif S="10"    then    Y <= I(2) ;
      elsif S="11"    then    Y <= I(3) ;
      end if ;
   end process ;
end A_if_then_else ;
```

解多工器 Demux



選擇輸入		輸出			
S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	I	0	0	0
0	1	0	I	0	0
1	0	0	0	I	0
1	1	0	0	0	I

1 對 4 線解多工器 VHDL(1)

```
--*****************************
entity Demux_1x4 is
  port ( I              : in    std_logic ;
         S1,S0          : in    std_logic ;
         Y3,Y2,Y1,Y0 : out std_logic ) ;
end Demux_1x4;
--*****************************
architecture A_Boolean of Demux_1x4 is
   begin
      Y0 <= I and (not S1) and (not S0) ;
      Y1 <= I and (not S1) and S0 ;
      Y2 <= I and S1 and (not S0) ;
      Y3 <= I and S1 and S0 ;
   end A_Boolean ;
```

1 對 4 線解多工器 VHDL(2)

```
--*****************************
entity Demux_1x4_1 is
  port ( I : in    std_logic ;
         S : in    std_logic_vector(1 downto 0) ;
            Y : out std_logic_vector(3 downto 0)) ;
end Demux_1x4_1 ;
--*****************************
architecture A_case_when of Demux_1x4_1 is
begin
   process(S,I)
   begin
      case S is
         when "00"     =>     Y(0)<=I ;
         when "01"     =>     Y(1)<=I ;
         when "10"     =>     Y(2)<=I ;
         when others =>     Y(3)<=I ;
      end case ;
   end process ;
end A_case_when ;
```

1 對 4 線解多工器(起始值預設) VHDL(3)

```
--*****************************
entity Demux_1x4_2 is
  port ( I : in    std_logic ;
         S : in    std_logic_vector(1 downto 0) ;
         Y : out std_logic_vector(3 downto 0)) ;
end Demux_1x4_2 ;
--*****************************
architecture A_case_when of Demux_1x4_2 is
begin
process(S,I)
variable Y_temp : std_logic_vector(3 downto 0);
   begin
      for i in 0 to 3 loop      --Y <= "0000" ;
            Y_temp(i) := '0' ;--起始值設定為 0
      end loop ;
      Y <= Y_temp ;
      case S IS
         when "00"     =>     Y(0)<=I ;
         when "01"     =>     Y(1)<=I ;
         when "10"     =>     Y(2)<=I ;
         when others =>     Y(3)<=I ;
      end case ;
   end process ;
end A_case_when ;
```

比較器

輸入		輸出		
A	B	L(A<B)	E(A=B)	G(A>B)
0	0	0	1	0
0	1	1	0	0
1	0	0	0	1
1	1	0	1	0

L=A' B

E=A' B'+AB=A⊙B

G=A B'

8 位元比較器 VHDL(1)

```
--*****************************
entity Compare_8bit is
  port ( A,B      : in    std_logic_vector(7 downto 0) ;
              G,E,L : out std_logic ) ;
end Compare_8bit;
--*****************************
architecture ARCH of Compare_8bit is
begin
   process(A,B)
   begin
      if A>B then      G <= '1' ;
      else    G <= '0' ;
      end if ;
      if A=B then      E <= '1' ;
      else    E <= '0' ;
      end if ;
      if A<B then      L <= '1' ;
      else    L <= '0' ;
      end if ;
   end process;
end ARCH    ;
```

3 個 if 敘述並行

8 位元比較器 VHDL(2)

```
--*****************************
entity Compare_8bit_1 is
  port ( A,B      : in    std_logic_vector(7 downto 0) ;
              G,E,L : out std_logic ) ;
end Compare_8bit_1;
--*****************************
architecture A_process of Compare_8bit_1 is
begin
   process(A,B)
   begin
      if A>B then    G <= '1' ;
      else    G <= '0' ;
      end if ;
   end process;
   process(A,B)
   begin
      if A=B then    E <= '1' ;
      else    E <= '0' ;
      end if ;
   end process;
   process(A,B)
   begin
      if A<B then    L <= '1' ;
      else    L <= '0' ;
      end if ;
   end process;
end A_process    ;
```

3 個 process 敘述並行

8 位元比較器 VHDL(3)

```
--*****************************
entity Compare_8bit_2 is
  port ( A,B : in    std_logic_vector(7 downto 0) ;
              GEL : out std_logic_vector(2 downto 0)) ;
end Compare_8bit_2;
--*****************************
architecture ARCH of Compare_8bit_2 is
begin
     GEL <= "100" when A>B else
                "010" when A=B else
                "001" when A<B ;
end ARCH    ;
```

8 位元 2 進碼轉 BCD 碼 VHDL(1)

```
--******************************
entity Bin_to_BCD_8bit is
  port ( B                    : in    unsigned(7 downto 0) ;
            BCD2,BCD1,BCD0 : out std_logic_vector(3
downto 0)) ;
end Bin_to_BCD_8bit ;
--******************************
architecture A_arith of Bin_to_BCD_8bit is
signal Temp : unsigned(6 downto 0) ;
begin
   process(B)
   begin
      if       B >199    then
               BCD2 <= "0010" ;
               Temp <= B-200 ;
      elsif B >99    then
               BCD2 <= "0001" ;
               Temp <= B-100 ;
      else
               BCD2 <= "0000" ;
               Temp <= B-0 ;
      end if;
      if       Temp >89    then
               BCD1 <= "1001" ;
               BCD0 <= Temp-90 ;
      elsif Temp >79 then
               BCD1 <= "1000" ;
               BCD0 <= Temp-80 ;
      elsif Temp >69 then
               BCD1 <= "0111" ;
               BCD0 <= Temp-70 ;
      elsif Temp >59 then
               BCD1 <= "0110" ;
               BCD0 <= Temp-60 ;
      elsif Temp >49 then
               BCD1 <= "0101" ;
               BCD0 <= Temp-50 ;
      elsif Temp >39 then
               BCD1 <= "0100" ;
               BCD0 <= Temp-40 ;
      elsif Temp >29 then
               BCD1 <= "0011" ;
               BCD0 <= Temp-30 ;
      elsif Temp >19 then
               BCD1 <= "0010" ;
               BCD0 <= Temp-20 ;
      elsif Temp >9 then
               BCD1 <= "0001" ;
               BCD0 <= Temp-10 ;
      else
               BCD1 <= "0000" ;
               BCD0 <= Temp-0 ;
      end if ;
   end process;
end A_arith ;
```