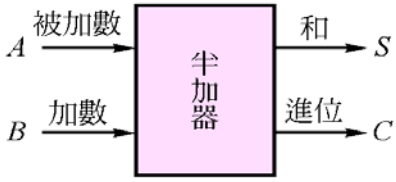


(a)方塊圖

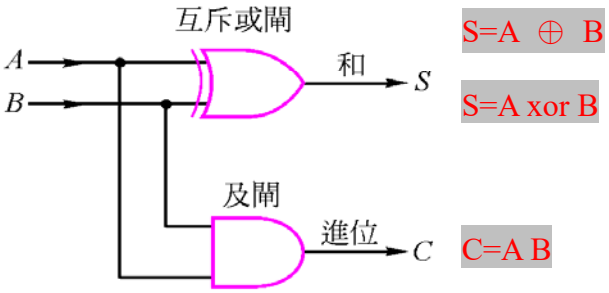
$$\begin{array}{r} A \\ +\quad B \\ \hline C\ \ S \end{array}$$

(b)功能

註: S = sum = 和
C = carry = 進位

輸入		輸出	
A	B	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

(c)眞值表



(d)電路圖

圖 4-3　一位元之二進位半加器

半加器程式(1)

half_add.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Half_add is
    port (   A,B : in    std_logic;
             S,C : out   std_logic   ) ;
end Half_add ;
--******************************
architecture A_Boolean of Half_add is
begin
   S <= A xor B ;
   C <= A and B ;
end A_Boolean   ;
```

半加器程式(2)

add_1.vhd

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
--*************************
entity add_1 is
port(
      a,b        :     in std_logic_vector(0 to 0);
      s,c        :     out std_logic   ) ;
end add_1;
--**************************
architecture arch of add_1 is
signal temp : std_logic_vector(1 downto 0);
begin
      temp<= “00”+a + b;
      s     <= temp(0);
      c     <= temp(1);
end arch;
```

須為 vector

◆建立自己的元件庫

●先建立存放自建元件之資料夾

c:\maxplus2\max2lib\mylib

●新增元件庫

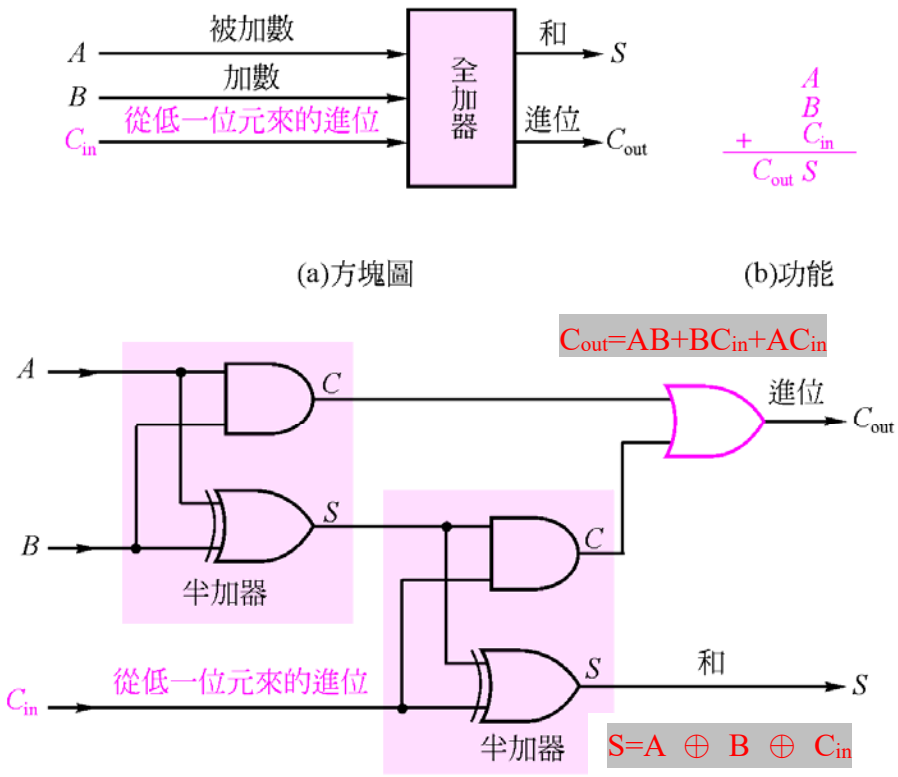
Option→User Libraries

●創造元件符號

File→Creat Default Symbol

◆ 匯流排的使用

■



輸 入			輸 出	
A	B	C_{in}	C_{out}	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

(c)眞值表

(d)電路圖

圖 4-4　一位元之二進位全加器

全加器程式(1)-布林式

full_add.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--*******************************
entity Full_add is
    port (   Ai,Bi,Ci      : in     std_logic;
             So,Co         : out    std_logic    ) ;
end Full_add ;
--*******************************
architecture A_Boolean of Full_add is
begin
   So <= Ai xor Bi xor Ci ;
   Co <= (Ai and Bi) or (Ai and Ci) or (Bi and Ci) ;
end A_Boolean    ;
```

全加器程式(2)-運算式

add_1.vhd

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_unsigned.all;
use ieee.std_logic_arith.all;
entity add_1 is
port(
      a,b,ci       :in std_logic_vector(0 to 0);
      s,c          :out std_logic) ;
end add_1;
architecture arch of add_1 is
signal temp : std_logic_vector(1 downto 0);
--內部訊號，不會出現在實體外部接線上
begin
      temp<= "00"+a + ci + b;
      s      <= temp(0);--temp(0)為 temp 低位元
      c      <= temp(1); --temp(1)為 temp 高位元
end arch;
```

須為 vector

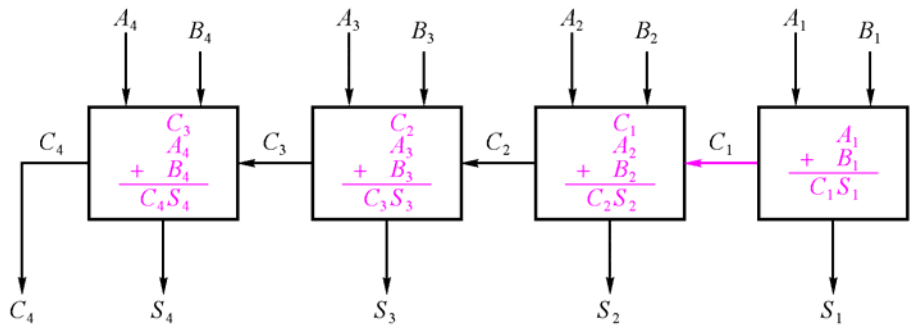
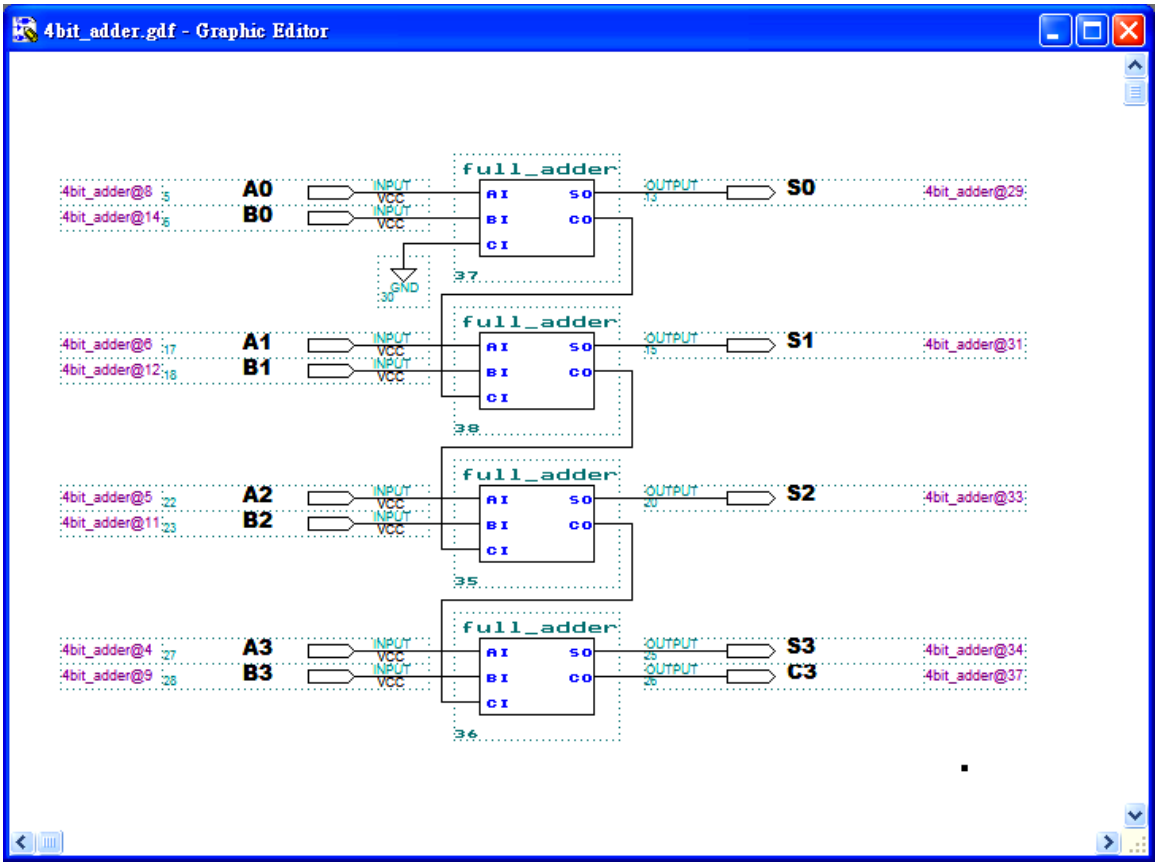


圖 4-5　4 位元加法器之概念圖



用 component 與 port map 的敘述方法描述電路

語法格式

component 元件(程式)名稱 --此一元件(程式)儲存位置須與本程式在同一目錄下

port(

　　訊號 1(電路腳位名稱):I/O 屬性　資料型態;

　　訊號 2(電路腳位名稱):I/O 屬性　資料型態;

　　:

　　:

　　);

end component;

四位元加法器-依電路圖方式

add_4bit.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Add_4bit is
  port ( A,B    : in std_logic_vector(3 downto 0) ;
         S      : out std_logic_vector(3 downto 0) ;
         Cout : out std_logic ) ;
end Add_4bit ;
--******************************
architecture A_port_map of Add_4bit is
  signal C1,C2,C3 : std_logic ;
component Half_add --公用元件名稱
    port ( A,B :   in     std_logic ;
           S,C :   out    std_logic) ;
end component ;
component Full_add--公用元件名稱
    port ( Ai,Bi,Ci : in    std_logic ;
           So,Co       : out std_logic    );
end component;
  begin
   U1 : Half_add    port map (A(0),B(0),S(0),C1);
   U2 : Full_add    port map (A(1),B(1),C1,S(1),C2);
   U3 : Full_add    port map (A(2),B(2),C2,S(2),C3);
   U4 : Full_add    port map (A(3),B(3),C3,S(3),Cout);
end A_port_map    ;
--位置對應方式為不使用"=>"符號，所以一定要依照公用元件中 port 內宣告的 I/O 順序為對應之順序
--使用名稱對應"=>"
--U1:Half_add    port map
            (A=>A(0),B=>B(0),S=>S(0),C=>C1);
--U2:Full_add    port map
    (Ai=>A(1),Bi=>B(1),Ci=>C1,So=>S(1),Co=>C2);
```

半加器程式(1)

half_add.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Half_add is
    port (    A,B : in    std_logic;
              S,C : out    std_logic    ) ;
end Half_add ;
--******************************
architecture A_Boolean of Half_add is
begin
   S <= A xor B ;
   C <= A and B ;
end A_Boolean    ;
```

全加器程式(1)-布林式

full_add.vhd

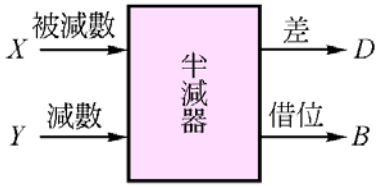
```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Full_add is
    port (    Ai,Bi,Ci       : in      std_logic;
              So,Co       : out    std_logic    ) ;
end Full_add ;
--******************************
architecture A_Boolean of Full_add is
begin
   So <= Ai xor Bi xor Ci ;
   Co <= (Ai and Bi) or (Ai and Ci) or (Bi and Ci) ;
end A_Boolean    ;
```

四位元加法器 2-使用算術運算

add_4bit_1.vhd

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

entity Add_4bit_1 is
  port ( A,B    : in std_logic_vector(3 downto 0) ;
             S        : out std_logic_vector(3 downto 0) ;
             Cout : out std_logic ) ;
end Add_4bit_1;
--******************************
architecture A_arith of Add_4bit_1 is
  signal Temp : std_logic_vector(4 downto 0)    ;
--signal 內部訊號宣告,宣告為標準邏輯向量,位於 architecture 與 begin 之間
  begin
    Temp <= ( '0' & A ) + B ;    --temp<=a+b;
    S <= Temp(3 downto 0);
    Cout <= Temp(4);
end A_arith    ;
```



(a)方塊圖

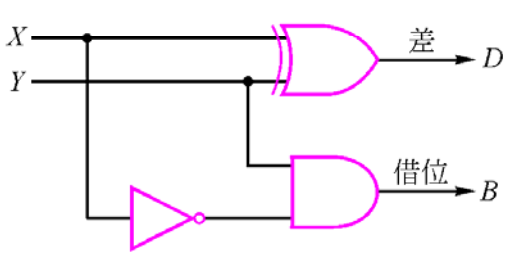
$$\begin{array}{r} X \\ -\quad Y \\ \hline B\ D \end{array}$$

(b)功能

註: D = difference = 差
B = borrow = 借位

輸入		輸出	
X	Y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

(c)眞值表



D=X ⊕ Y
=X xor Y

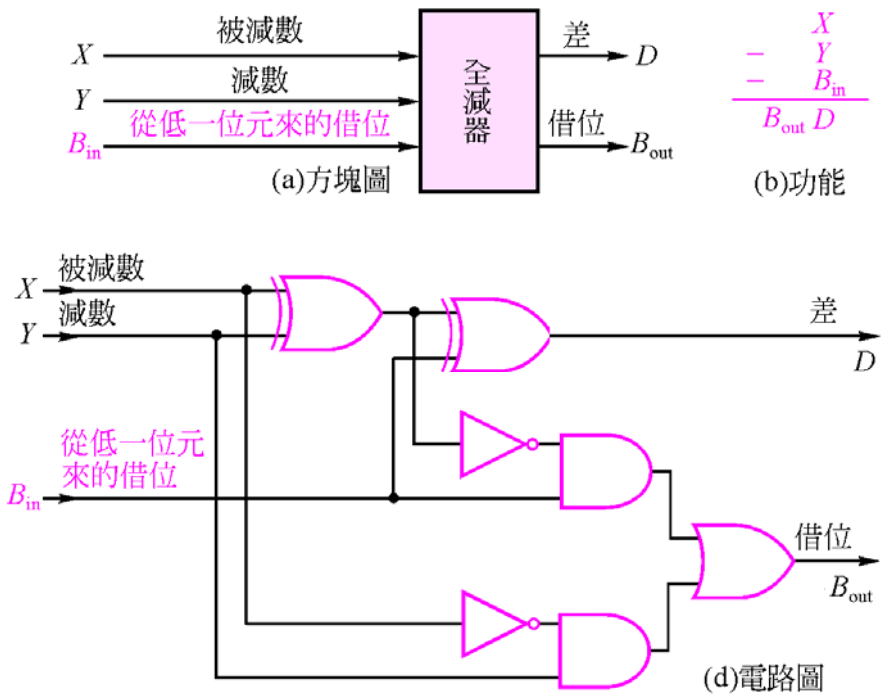
B=X'Y

(d)電路圖

圖 5-1　一位元的二進位半減器

Half_sub.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Half_sub is
    port ( A,B     : in    std_logic ;
           Do,Bo : out std_logic ) ;
end Half_sub ;
--******************************
architecture A_boolean of Half_sub is
begin
    Do <= A xor B;
    Bo <= (not A) and B ;
end A_boolean ;
```



輸入			輸出	
X	Y	B_{in}	B_{out}	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

(c)眞值表

圖 5-2　一位元之二進位全減器

差 D(a,b,bi)= Σ(1,2,4,7)=a'b'bi+a'bbi'+ab'bi'+abbi

借位 B(a,b,bi)= Σ(1,2,3,7)=a'b'bi+a'bbi'+a'bbi+abbi

(1) Full_sub.vhd—布林式

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Full_sub is
    port ( A,B,Bi    : in    std_logic;
                Do,Bo      : out std_logic ) ;
end Full_sub ;
--******************************
architecture A_table of Full_sub is
begin
   Do <= ((not A) and (not B) and Bi)    or
            ((not A) and    B and (not Bi)) or
            (A and (not B) and (not Bi))    or
            (A and B and Bi) ;
   Bo <= ((not A) and (not B) and Bi)    or
            ((not A) and    B and (not Bi)) or
            ((not A) and    B and    Bi)          or
            (A and B and Bi) ;
end A_table    ;
```

使用 when ….else 描述敘述，其語法格式為

[輸出]訊號	<=	輸出邏輯狀態 1	when	判斷條件 1	else
		[輸出邏輯狀態 2	when	判斷條件 2	else]
		[……]			
		輸出邏輯狀態 N;			

(2) Full_sub_1.vhd -- 使用 when …else

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Full_sub_1 is
   port ( A,B,Bi    : in    std_logic;
                Do,Bo      : out std_logic ) ;
end Full_sub_1 ;
--******************************
architecture A_when_else of Full_sub_1 is
begin
   Do <= '1' when (A='0' and B='0' and Bi='1') else
            '1' when (A='0' and B='1' and Bi='0') else
            '1' when (A='1' and B='0' and Bi='0') else
            '1' when (A='1' and B='1' and Bi='1') else
            '0';
   Bo <= '1' when (A='0' and B='0' and Bi='1') else
            '1' when (A='0' and B='1' and Bi='0') else
            '1' when (A='0' and B='1' and Bi='1') else
            '1' when (A='1' and B='1' and Bi='1') else
            '0';
end A_when_else    ;
```

Full_sub_2.vhd-- 使用 when …else

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Full_sub_2 is
   port ( A,B,Bi : in    std_logic;
                Do,Bo    : out std_logic ) ;
end Full_sub_2 ;
--******************************
architecture A_when_else of Full_sub_2 is
  signal Temp : std_logic_vector( 2 downto 0 ) ;
begin
  Temp <= A & B & Bi ;
   Do <= '1' when Temp="001" else
            '1' when Temp="010" else
            '1' when Temp="100" else
            '1' when Temp="111" else
            '0';
   Bo <= '1' when Temp="001" else
            '1' when Temp="010" else
            '1' when Temp="011" else
            '1' when Temp="111" else
            '0';
end A_when_else    ;
```

Full_sub_3.vhd-- 使用 when …else

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Full_sub_3 is
   port ( A,B,Bi    : in    std_logic;
                Do,Bo      : out std_logic ) ;
end Full_sub_3 ;
--******************************
architecture A_when_else of Full_sub_3 is
  signal Temp : std_logic_vector( 2 downto 0 ) ;
  signal O_temp : std_logic_vector( 1 downto 0 ) ;
begin
   Temp <= A & B & Bi ;
   O_temp <= "11" when Temp="001" else
                  "11" when Temp="010" else
                  "10" when Temp="011" else
                  "01" when Temp="100" else
                  "11" when Temp="111" else
                  "00" ;
   Do <= O_temp(0) ;
   Bo <= O_temp(1) ;
  end A_when_else    ;
```

使用 with…..select……when 語法格式如下

[輸出]訊號 <= 輸出的邏輯狀態 1 when 判斷訊號的邏輯狀態 1,
輸出的邏輯狀態 2 when 判斷訊號的邏輯狀態 2,
[……],
輸出的邏輯狀態 N when others;

Full_sub_4.vhd--使用 with.....select…..when

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

--******************************
entity Full_sub_4 is
   port ( A,B,Bi    : in    std_logic;
                Do,Bo      : out std_logic ) ;
end Full_sub_4 ;
--******************************
architecture A_with_select_when of Full_sub_4 is
  signal Temp : std_logic_vector( 2 downto 0 ) ;
begin
  Temp <= A & B & Bi;
  with Temp select
       Do <= '1' when "001" ,
                '1' when "010" ,
                '1' when "100" ,
                '1' when "111" ,
                '0' when others;
  with Temp select
       Bo <= '1' when "001" ,
                '1' when "010" ,
                '1' when "011" ,
                '1' when "111" ,
                '0' when others;
end A_with_select_when    ;
```

補數

$x - y = x + (-y)$

將一正數的二進位取 2 的補數，就是其負數。

步驟：　(1)先寫出該數之 2 進位數。

(2)將此 2 進位數的每個位元反相，此時的值稱為 1 的補數。

(3)再把 1 的補數加 1，即為 2 的補數。

例：10 進位 -3_{10} 的 2 的補數

因為 3_{10}=0011

所以　0011

↓ 反相

1100

↓ 加 1

1101

所以得知-3_{10}用 2 的補數表示為 1101

例：11-4=7

先求-4 的 2 的補數為

0100

↓ 反相

1011

↓ 加 1

1100

則　1011 ← 11

+　1100 ← -4 的 2 的補數

10111

最高位元 0 表正數

溢位，忽略

所以答案為 0111=7

例：9-21=-12

-21 的 2 的補數為

10101

↓ 反相

01010

↓ 加 1

01011

則　01001 ← 9

+　01011 ← -12 的 2 的補數

10100

沒有進位最高位元 1 表負數

答案如為負數，需再取其 2 的補數

10100

↓ 反相

01011

↓ 加 1

$01100 = 12_{10}$

所以答案為 -12

運用 process 順序性敘述 與 if..then..else 敘述

Add_sub_4bit.vhd

```
library ieee ;
use ieee.std_logic_1164.all ;
use ieee.std_logic_unsigned.all ;
use ieee.std_logic_arith.all ;
--******************************
entity Add_sub_4bit is
  port (
          SUB : in    std_logic;
          A,B : in    std_logic_vector(3 downto 0);
          S     : out std_logic_vector(3 downto 0);
          C3    : out std_logic
        ) ;
end Add_sub_4bit ;
--******************************
architecture ARCH of Add_sub_4bit is
   signal Temp : std_logic_vector(4 downto 0);
begin
  process(SUB,A,B)
  begin
    if SUB = '0' then
          Temp <= ( '0' & A ) + B;  ← 加法
          S <= Temp(3 downto 0);
          C3<= Temp(4);
    else
          Temp <= ('0' & A) + ( not(B) + 1 );  ←減法
          S <= Temp(3 downto 0);
          C3 <= Temp(4);
    end if;
  end process;
end ARCH ;
```

```
[標名:] process(感測訊號 1,2,…)
      [訊號、變數宣告]
      begin
            [順序性敘述]或[指定敘述]
      ……————————
      end process[標名];
```

```
if  判斷條件 1 then
            第一組敘述;
      [elsif  判斷條件 2 then
            第二組敘述;
                  :
      ————————————————]
      else
            第 N 組敘述;
      end if ;
```

BCD 碼對共陰 7 段顯示器之解碼器

```
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    ;
```

```
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    ;
```

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 ;
```