表 3-4-6　電路圖輸出入腳位、CPLD 腳位及實驗器模組腳位對應表

腳位對應關係	輸入				輸出						
名　稱	D	C	B	A	Y_A	Y_B	Y_C	Y_D	Y_E	Y_F	Y_G
CPLD 晶片腳位	PIN 24	PIN 22	PIN 21	PIN 20	PIN 30	PIN 31	PIN 33	PIN 34	PIN 35	PIN 36	PIN 37
實驗器模組對應腳位	DIP A_4	DIP A_3	DIP A_2	DIP A_1	A_0	B_0	C_0	D_0	E_0	F_0	G_0

實驗器使用第 A 個指撥開關的 $DIPA_1$~$DIPA_4$ 接腳當輸入端，第 0 個七段顯示器 A_0~G_0 接腳當輸出端。

3-5　多工器與解多工器

資料傳輸時，若是將由許多位置的資料點傳輸至許多位置的資料點時，則連接的線路將錯縱複雜。如果傳輸距離遠時，線路連接及維護更是一大問題。這種多端點資料傳送情形，一般都用多工器(Multiplexer)與解多工器(Demultiplexer)來解決。在資料發送端安裝多線變一線的多工器。多工器上有選擇線路，用來決定該送出哪一線路上的資料。多工器用於發送資料端資料選擇之用，又稱為資料選擇器(Data selector)。當資料接收端有多點位置可接收資料時，則需安裝一線變多線的解多工器，解多工器上亦有選擇線路，用來決定該由哪一條線路接收資料。解多工器用於接收資料端分配資料之用，又稱資料分配器(Data distributor)。如圖 3-5-1 所示，甲地資料欲傳送到乙地，甲地資料傳輸點數量有 2^n 個，則需有 n 條選擇線，用以決定送出哪一個端點之資料，乙地接收點數量有 2^m 個，需有 m 條選擇線用以決定將資料分配給哪一點。

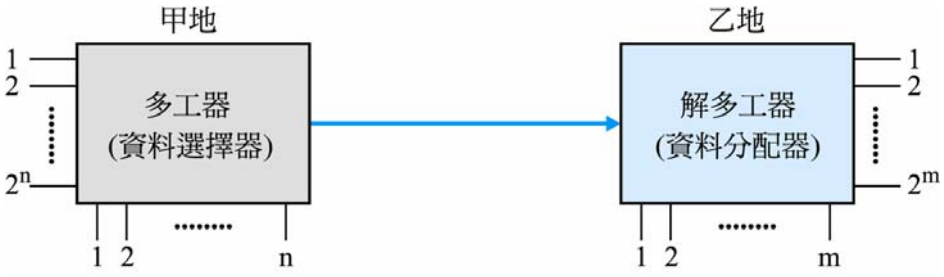

圖 3-5-1　多工器與解多工器示意圖

3-5-1 多工器

多工器是能將多個輸入訊號中選擇其中一個傳送到輸出端的電路。若輸入有 M 條，稱為 M 對 1 線多工器(M to 1 Multiplexer)。在此介紹 4 對 1 線多工器作法。如圖 3-5-2 所示為 4 對 1 線多工器方塊圖。

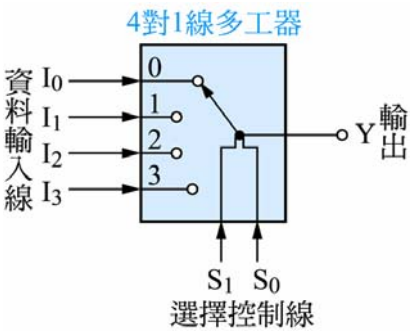

圖 3-5-2 多工器方塊圖

真值表

選擇線		輸出信號
S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

布林函數：

$$Y = \overline{S}_1\overline{S}_0I_0 + \overline{S}_1S_0I_1 + S_1\overline{S}_0I_2 + S_1S_0I_3$$

設計方法如下：

1. 開啟新圖形編輯檔，依照真值表繪製邏輯電路圖，如圖 3-5-3 所示。

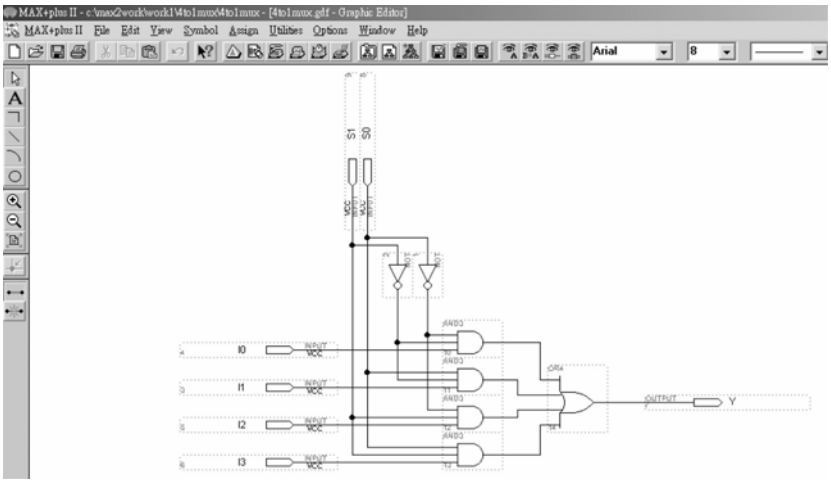

圖 3-5-3 四對一多工器電路圖

2. 如圖 3-5-1 所示，經由步驟：畫電路圖→存檔→設為工作專案→指定 CPLD 晶片→編譯→軟體模擬→規劃腳位→編譯→下載燒錄(若是 Atmel 的晶片下載前需轉檔) 後，即完成電路設計製作，此時可配合外部電路(或實驗器模組，需配合腳位規劃)來驗證。

執行步驟如下：

1. 開啟一個新圖形編輯檔，繪製一個四對一多工器的電路圖。
2. 存檔，取檔名為 4 to 1 mux.gdf。
3. 設為指定工作專案(File→Project→Set Project to Current File)，指定 CPLD 晶片(Assign→Device)，並編譯(MAX+PLUS Ⅱ→Compiler)。
   如圖所示，編譯後沒有錯誤與警告，表示邏輯正確，可進行軟體模擬。

圖 3-5-4　編譯訊息視窗

4. 開啟新的波形編輯檔案，設定功能模擬結束時間(File→End Time)設定 64 μs，設定格線間距(Options→Grid Size)設定 1 μs，顯示在視窗中適當大小格線(View→Fit in Window)。
   (因為在此輸入端共有六支腳，每支腳有兩種狀態(0 與 1)，總共 2^6 =64 種變化。

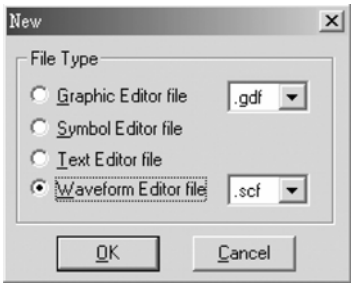

圖 3-5-5　開啟新檔視窗

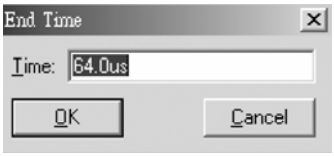

圖 3-5-6　模擬結束時間設定視窗

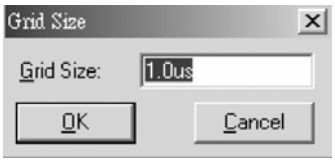

圖 3-5-7　模擬單位時間設定視窗

5. 儲存檔案(Save As)，檔名 4 to1mux.scf，輸入節點(Node → Enter Nodes from SNF，按 List 及 =>，OK)，編輯輸入信號，輸入端有六支腳，總共有六十四種組合，剛好 64 μs 可模擬完，可將輸入端六支腳群組化，用計數時脈編輯 000000~111111 即可完成輸入信號編輯。

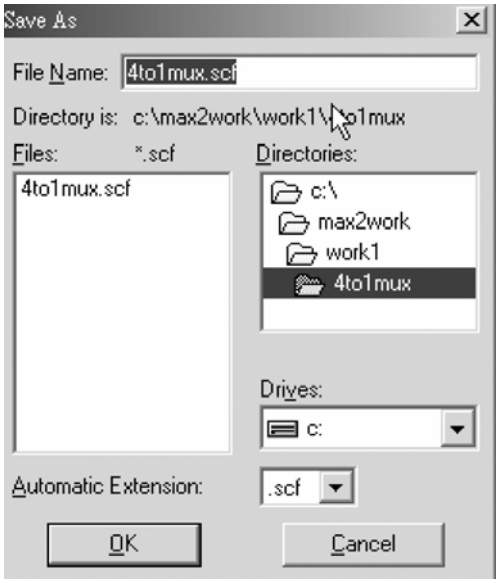

圖 3-5-8　儲存檔案視窗

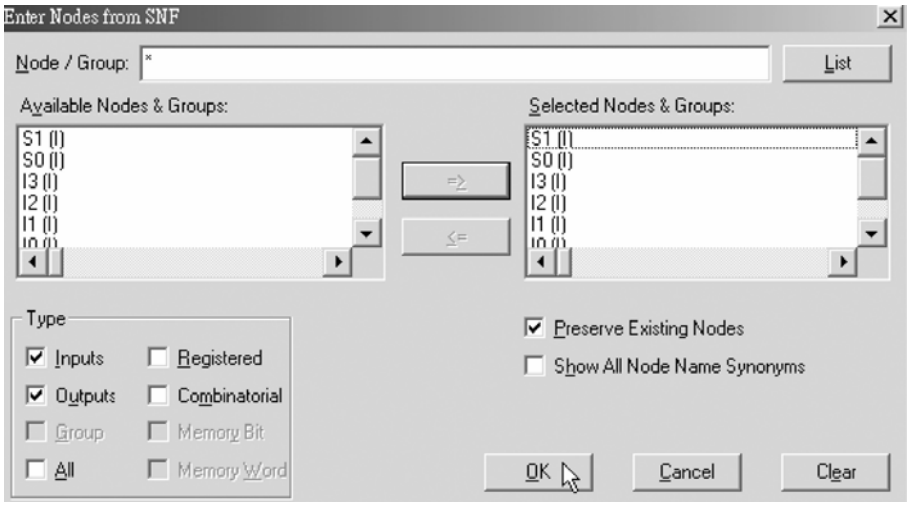

圖 3-5-9　輸出入節點選擇視窗

6. 執行模擬(MAX+PLUS II → Simulator，Start)如圖 3-5-10 所示，所得波形模擬結果符合 4 to1 線多工器，代表我們製作的電路是正確可用的。

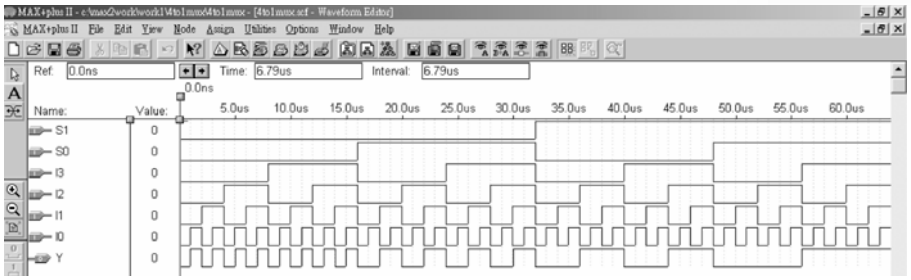

圖 3-5-10　模擬結果

模擬成功之後，可下載(燒錄)到實驗板做實際電路測試，以下為接到尼德公司實驗板的接腳表格，讀者可以按表中接腳設定以完成硬體電路測試。

表 3-5-1　電路圖輸出入腳位、CPLD 腳位及實驗器模組腳位對應表

腳位對應關係	輸入						輸出
名稱	I_0	I_1	I_2	I_3	S_0	S_1	Y
CPLD 晶片腳位	PIN 20	PIN 21	PIN 22	PIN 24	PIN 25	PIN 27	PIN 81
實驗器模組對應腳位	DIP A_1	DIP A_2	DIP A_3	DIP A_4	DIP A_5	DIP A_6	DG_7

實驗器使用第 A 個指撥開關的 $DIPA_1$~$DIPA_6$ 接腳當輸入端，第 8 個綠色 LED 接腳當輸出端。

3-5-2　解多工器

解多工器是能將一個輸入訊號選擇由多個輸出端中的一個傳送出去的電路。若輸出有 M 條，稱為 1 對 M 線解多工器(M to 1 Multiplexer)。在此介紹 1 對 4 線解多工器作法。如圖 3-5-11 所示為 1 對 4 線解多工器方塊圖。

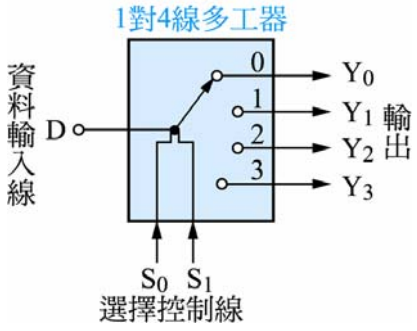

圖 3-5-11　1 對 4 線解多工器方塊圖

真值表

選擇線		輸出信號			
S_1	S_0	Y_0	Y_1	Y_2	Y_3
0	0	D	0	0	0
0	1	0	D	0	0
1	0	0	0	D	0
1	1	0	0	0	D

布林函數：

$Y_0 = \overline{S_1}\,\overline{S_0}D$

$Y_1 = \overline{S_1}S_0D$

$Y_2 = S_1\overline{S_0}D$

$Y_3 = S_1S_0D$

設計方法如下：

1. 開啟新圖形編輯檔，依照真值表繪製邏輯電路圖，如圖 3-5-12 所示。

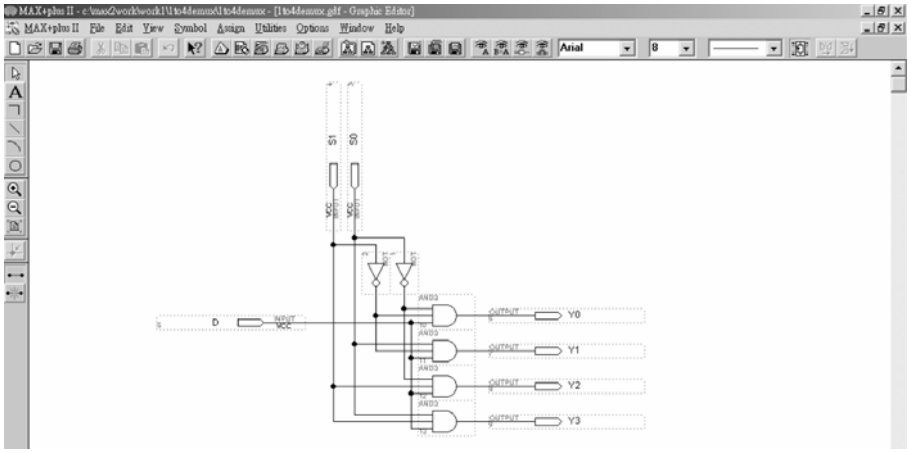

圖 3-5-12　1 對 4 線解多工器電路圖

2. 如圖 3-5-12 所示，經由步驟：畫電路圖 →存檔→設為工作專案→指定 CPLD 晶片→編譯→軟體模擬→規劃腳位→編譯→下載燒錄(若是 Atmel 的晶片下載前需轉檔)後，即完成電路設計製作，此時可配合外部電路(或實驗器模組，需配合腳位規劃)來驗證。

執行步驟如下：

1. 開啟一個新圖形編輯檔，繪製一個共陽極七段顯示解碼器的電路圖。

2. 存檔，取檔名為 1 to 4 demux.gdf。

3. 設為指定工作專案(File→Project→Set Project to Current File)，指定 CPLD 晶片(Assign→Device)，並編譯(MAX+PLUS Ⅱ→ Compiler)。

如圖 3-5-13 所示，編譯後沒有錯誤與警告，表示邏輯正確，可進行軟體模擬。

圖 3-5-13　編譯訊息視窗

4. 開啟新的波形編輯檔案，設定功能模擬結束時間(File→End Time)設定 8 μs，設定格線間距(Options→Grid Size)設定 1 μs，顯示在視窗中適當大小格線(View→Fit in Window)。

(因為在此輸入端共有 3 支腳，每支腳有兩種狀態(0 與 1)，總共有 8 種變化。)

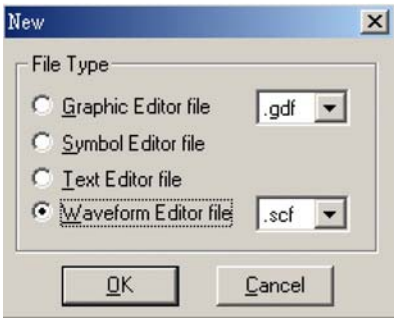

圖 3-5-14　開啟新檔視窗

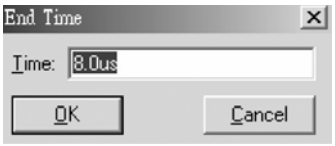

圖 3-5-15　模擬結束時間設定視窗

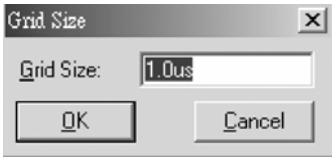

圖 3-5-16　模擬單位時間設定視窗

5. 儲存檔案(Save As)，檔名 1 to 4demux.scf，輸入節點(Node → Enter Nodes from SNF，按 List 及 =>，OK)，編輯輸入信號，輸入端有 3 支腳，總共有 8 種組合，剛好 8 μs 可模擬完，可將輸入端 3 支腳群組化，用計數時脈編輯 000~111 即可完成輸入信號編輯。

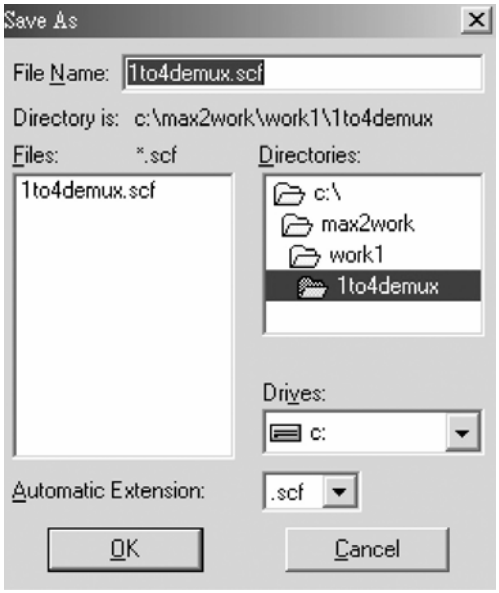

圖 3-5-17　儲存檔案視窗

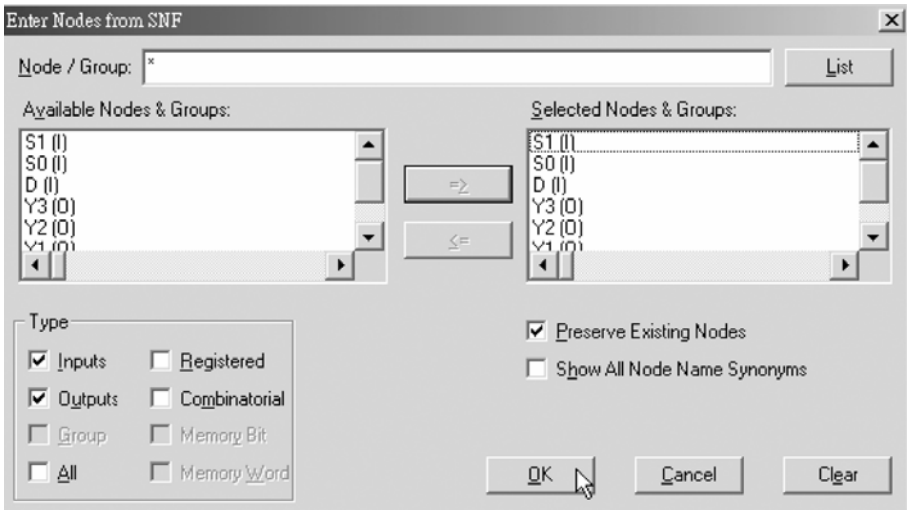

圖 3-5-18　輸出入節點選擇視窗

6. 執行模擬(MAX+PLUS Ⅱ → Simulator，Start)如圖 3-5-19 所示，所得波形模擬結果符合 1 to 4 線解多工器，代表我們製作的電路是正確可用的。

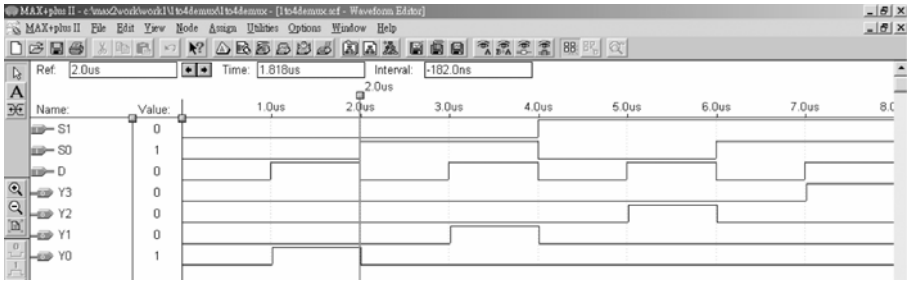

圖 3-5-19　模擬結果

模擬成功之後，可下載(燒錄)到實驗板做實際電路測試，以下為接到尼德公司實驗板的接腳表格，讀者可以按表中接腳設定以完成硬體電路測試。

表 3-5-2　電路圖輸出入腳位、CPLD 腳位及實驗器模組腳位對應表

腳位對應關係	輸入			輸出			
名稱	D	S_0	S_1	Y_0	Y_1	Y_2	Y_3
CPLD 晶片腳位	PIN 20	PIN 21	PIN 22	PIN 77	PIN 79	PIN 80	PIN 81
實驗器模組對應腳位	DIP A_1	DIP A_2	DIP A_3	DG_4	DG_5	DG_6	DG_7

實驗器使用第 A 個指撥開關的 $DIPA_1$~$DIPA_3$ 接腳當輸入端，第 5,6,7,8 個綠色 LED 接腳當輸出端。

3-6　二進制轉 BCD 碼數碼轉換器

BCD 碼(Binary Coded Decimal Code)表示是二進碼的十進數，以四位元的二進碼數值來表示一個位元的十進制數值，可表示範圍為 0~9。因此，四個位元的 BCD 碼只能有十個數值，即 0000~1001。以下為 BCD 碼所表示的十進數數值表。

十進數值	BCD 碼			
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1

3-6-1 四位元二進制轉 BCD 碼

四位元的 BCD 碼只能代表 0~9 的數值，當所表示的十進制數值超過 9 時，則需進位表示。四位元的二進制可表示的範圍為 0~15，因此二進制與 BCD 碼的轉換需特別注意，當二進制數值超過 9 時，BCD 碼即需以進位來表示數值。因為四位元二進制可表示的範圍的數值有 16 種(即 2^4=16，代表 0~15)，而四位元的 BCD 碼只能有 10 種(代表 0~9)，相差 6 種，所以有一種簡單的轉換方式是以加 6 的方法，當四位元二進制的數值超過 10(包含 10)時，將其加 6，則其數值就恰好為 BCD 碼，由以下真值表觀察可知。

真值表

數值	二進制				BCD 碼				
0	0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	0	1
2	0	0	1	0	0	0	0	1	0
3	0	0	1	1	0	0	0	1	1
4	0	1	0	0	0	0	1	0	0
5	0	1	0	1	0	0	1	0	1
6	0	1	1	0	0	0	1	1	0
7	0	1	1	1	0	0	1	1	1
8	1	0	0	0	0	1	0	0	0
9	1	0	0	1	0	1	0	0	1
10	1	0	1	0	1	0	0	0	0
11	1	0	1	1	1	0	0	0	1
12	1	1	0	0	1	0	0	1	0
13	1	1	0	1	1	0	0	1	1
14	1	1	1	0	1	0	1	0	0
15	1	1	1	1	1	0	1	0	1

藉由真值表可用前述所介紹組合邏輯設計方式來完成製作。

3-6-2 多位元二進制轉 BCD 碼

轉換二進制成 BCD 碼時，當所轉換的二進位有 5 個位元時，則數字表示為 0~31 的 32 種組合，若有 6 位元，則為 0~63 的 64 種組合。由此可知當位元數愈多，則組合數愈多，若要以列真值表、組合邏輯化簡的方式來製作的話，相當不容易。因此，在此要介紹一種利用自行製作元件的方法來製作多位元二進制轉 BCD 碼電路，並以 8 位元二進碼轉 BCD 為例來介紹。

如表 3-6-1 所示為超過 5 則加 3 的二進碼轉 BCD 碼的轉換真值表，在此的主要觀念在於二進碼轉換成 BCD 碼時，當二進制數值超過 10(包含 10)就必須加 6 才會得到正確的 BCD 碼，亦即當四位元的二進碼數值超過 1010 時，就必須加 6。因此我們可先檢查高位元的三個數值 Bin_3 Bin_2 Bin_1，當其值若是超過 101 時，我們可以預測此四位元的二進碼必然超過或等於 1010，此時將此數碼加 3，則在檢測第 4 碼 Bin0 之後，前三個位元左移一位元，即將原數值 Bin_3 Bin_2 Bin_1 乘 2。亦即當四位元二進制的高位元 Bin_3 Bin_2 Bin_1 的數值超過 5 時，則將其加 3 再左移一位元(乘 2)後，就變成超過 10 加 6 的二進碼轉 BCD 之轉換，以此觀念可依超過 5 則加 3 真值表製作 4 位元二進碼轉 BCD 碼的自製元件，利用重覆使用這元件將多位元二進碼由高位元輸入，其輸出端只要超過 4 個位元就需經過這超過 5 則加 3 的轉換，如此便可將多位元的二進制轉成 BCD 碼。

表 3-6-1　超過 5 則加 3 的二進碼轉 BCD 碼的轉換真值表

數值	二進制				BCD 碼超過 5 則加 3			
	Bin_3	Bin_2	Bin_1	Bin_0	BCD_3	BCD_2	BCD_1	BCD_0
	H	G	F	E	N	M	L	K
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	0
3	0	0	1	1	0	0	1	1
4	0	1	0	0	0	1	0	0
5	0	1	0	1	1	0	0	0
6	0	1	1	0	1	0	0	1
7	0	1	1	1	1	0	1	0
8	1	0	0	0	1	0	1	1
9	1	0	0	1	1	1	0	0

超過 5 則加 3

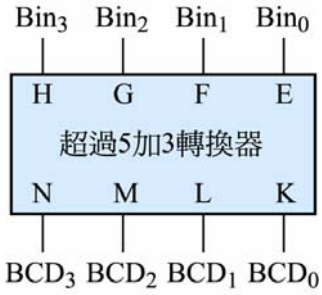

圖 3-6-1　四位元二進碼超過 5 則加 3 轉換示意圖

布林函數：

$$N=\overline{H}G\overline{F}E+\overline{H}GF\overline{E}+\overline{H}GFE+H\overline{G}\,\overline{F}\,\overline{E}+H\overline{G}\,\overline{F}E$$

$$M=\overline{H}G\overline{F}\,\overline{E}+H\overline{G}\,\overline{F}E$$

$$L=\overline{H}\,\overline{G}F\overline{E}+\overline{H}\,\overline{G}FE+\overline{H}GFE+H\overline{G}\,\overline{F}\,\overline{E}$$

$$K=\overline{H}\,\overline{G}\,\overline{F}E+\overline{H}\,\overline{G}FE+\overline{H}GF\overline{E}+H\overline{G}\,\overline{F}\,\overline{E}$$

利用真值表依組合邏輯電路化簡方式，繪製如下電路圖：

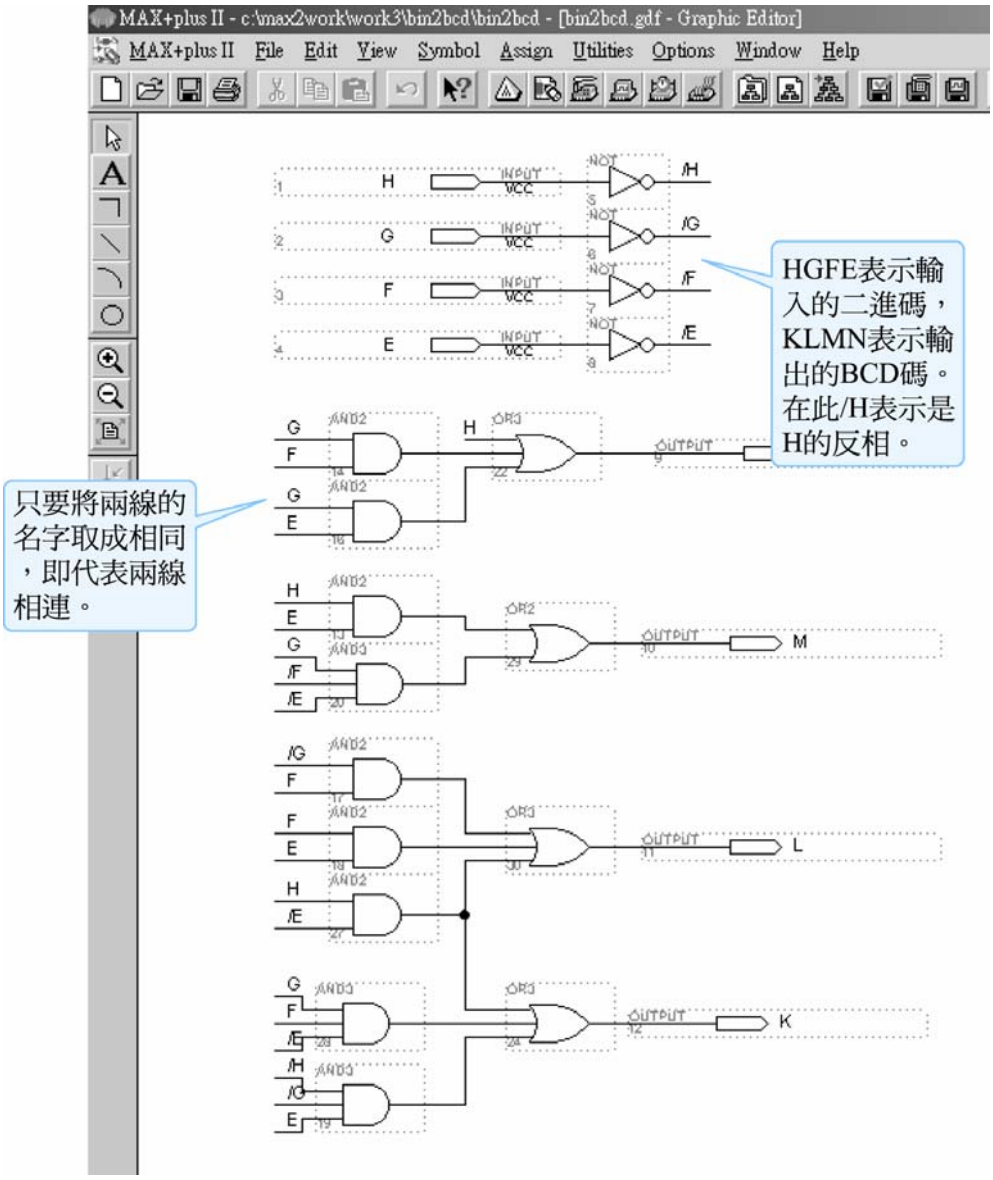

圖 3-6-2　四位元二進碼超過 5 則加 3 轉換電路圖

如圖 3-6-2 所示為依超過 5 則加 3 的真值表所完成的二進碼轉 BCD 碼轉換器。圖中輸入端接腳與其它接腳看似沒有連接，其實是利用 MAX+PLUS Ⅱ的功

能將接線取成相同名字即可代表線條相連。

設計超過 5 則加 3 二進碼轉 BCD 碼轉換器步驟如下：

1. 開啟一個新圖形編輯檔，繪製一個二進碼轉 BCD 碼轉換器的電路圖。
2. 存檔，取檔名為 bin2bcd.gdf。
3. 設為指定工作專案(File→Project→Set Project to Current File)，指定 CPLD 晶片(Assign→Device)，並編譯(MAX+PLUS Ⅱ → Compiler)。
   如圖 3-6-3 所示，編譯後沒有錯誤與警告，表示邏輯正確，可進行軟體模擬。

圖 3-6-3　編譯訊息視窗

4. 開啟新的波形編輯檔案，設定功能模擬結束時間(File→End Time)設定 10 μs，設定格線間距(Options → Grid Size)設定 1 μs，顯示在視窗中適當大小格線(View→Fit in Window)。
   (因為在此輸入端計數 0~9，因此可以 10 種狀態模擬完成。)

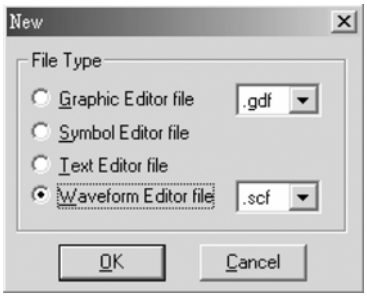

圖 3-6-4　開啟新檔視窗

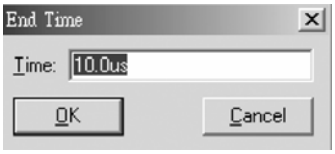

圖 3-6-5　模擬結束時間設定視窗

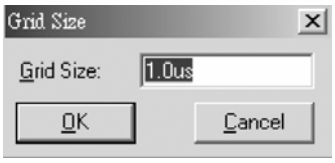

圖 3-6-6　模擬單位時間設定視窗

5. 儲存檔案(Save As)，檔名 bin2bcd.scf，輸入節點(Node→Enter Nodes from SNF，按 List 及 =≥ ，OK)，編輯輸入信號，計數 0~9。

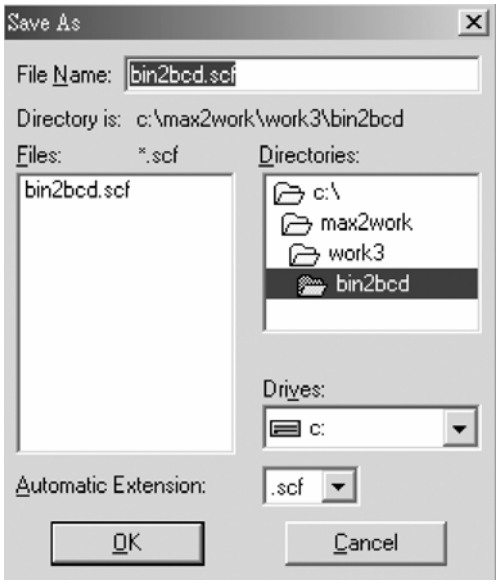

圖 3-6-7　儲存檔案視窗

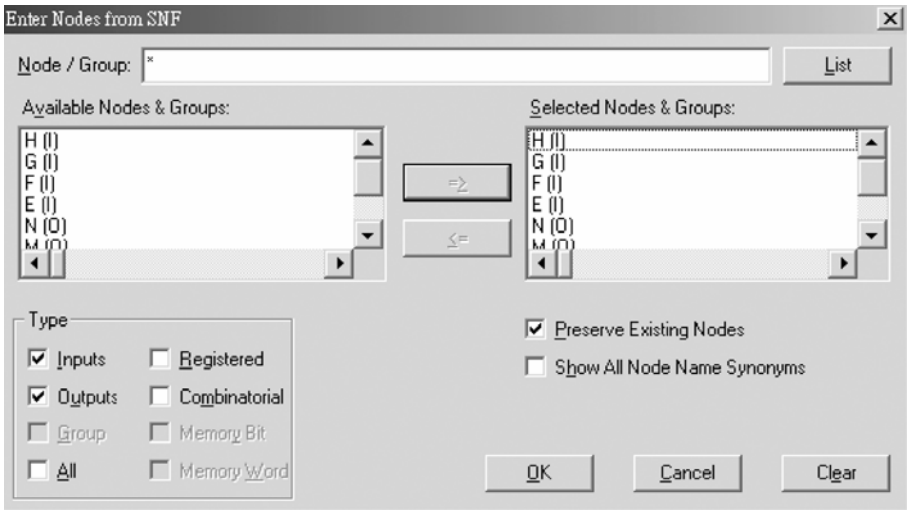

圖 3-6-8　輸出入節點選擇視窗

6. 執行模擬(MAX+PLUS Ⅱ → Simulator，Start)如圖 3-6-9 所示，所得波形模擬結果符合超過 5 則加 3 二進碼轉 BCD 碼轉換器，代表我們製作的電路是正確可用的。

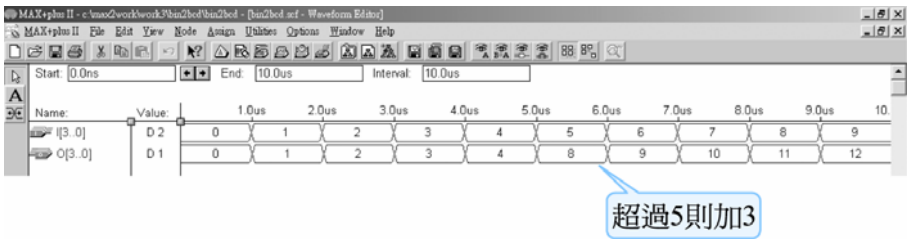

圖 3-6-9　模擬結果

製作超過 5 則加 3 二進碼轉 BCD 碼轉換器元件：

1. 開啟一個新圖形編輯檔，繪製一個二進碼轉 BCD 碼轉換器的電路圖。

2. 存檔，取檔名為 bin2bcd.　gdf。

3. 編譯。

4. 產生符號檔，點選 File→Create Default Symbol，即可產生新的元件(名稱 bin2bcd.sym)。

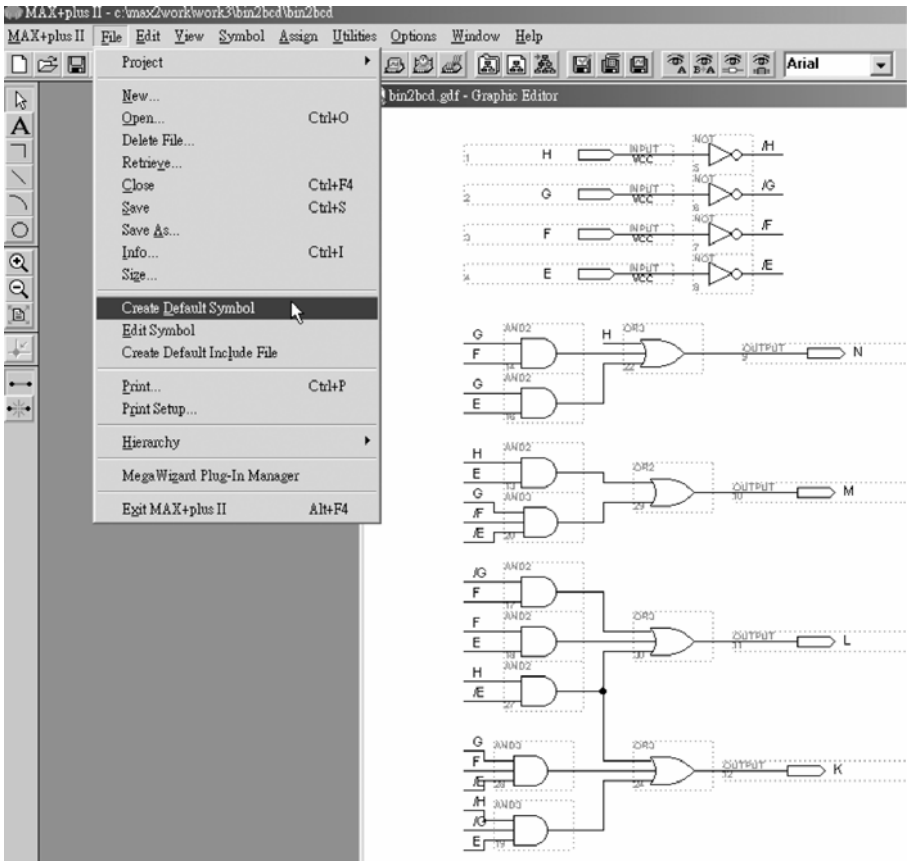

圖 3-6-10　製作自製元件視窗

5. 開啟新的圖形編輯檔，此時即可取用所製作的元件(bin2bcd.sym)，點選 Symbol→Enter Symbol(或在編輯視窗按兩下 Double Click)，即出現元件取用視窗。

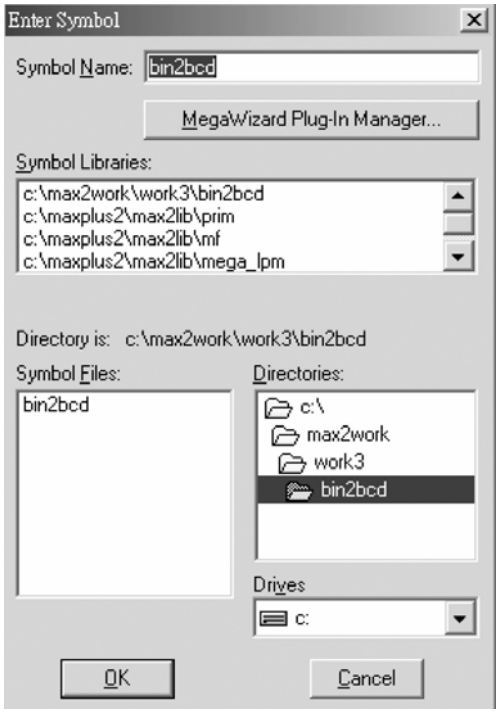

圖 3-6-11　元件取用視窗

6. 取用二進碼轉 BCD 元件(symbol)：進入剛才存放二進碼轉 BCD 元件的資料夾，點選符號元件檔(Symbol File) bin2bcd，按 OK，在圖形編輯視窗中，即可看到該元件(包裝後的二進碼轉 BCD 元件)。可在二進碼轉 BCD 元件上按兩下，即可開啟其內部電路視窗。

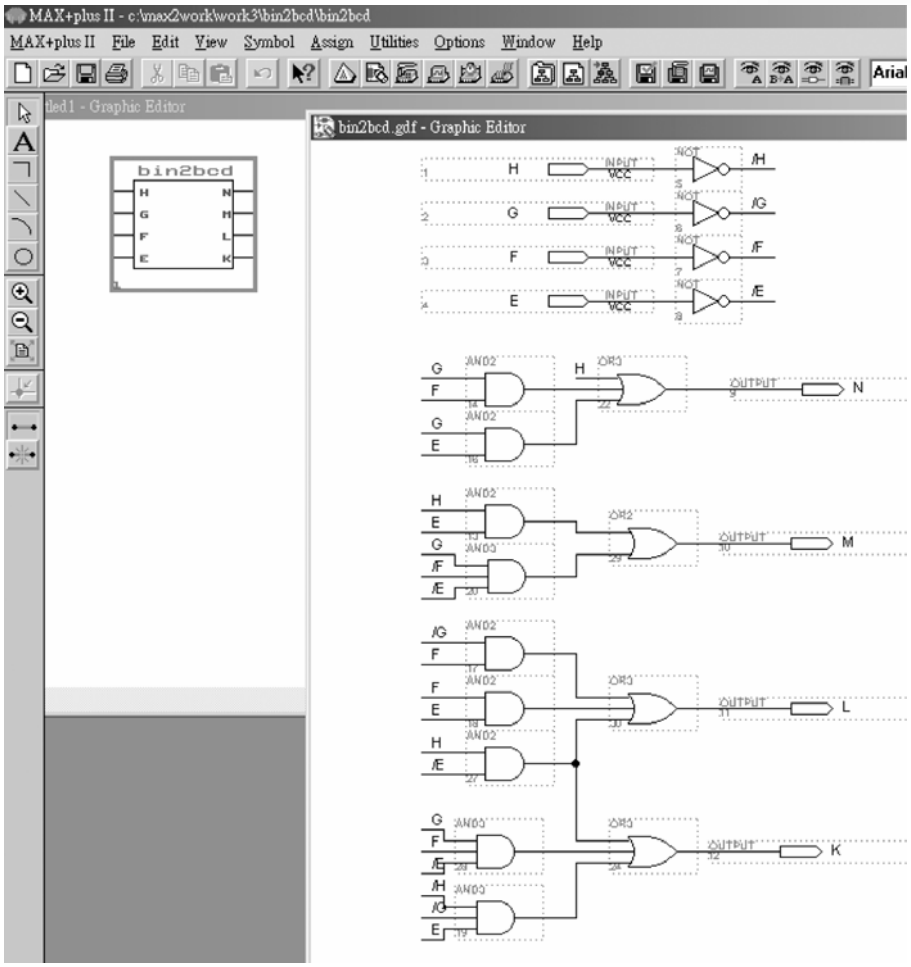

圖 3-6-12　自製半加器元件及內部電路

製作完成超過 5 則加 3 二進碼轉 BCD 碼轉換器元件，則可利用此元件設計 8 位元二進碼轉 BCD 碼轉換器，如圖 3-6-13 所示，在新的圖形檔中叫出此元件，由高位元依序輸入，高三個位元先輸入第一個 bin2bcd 的 GFE，其 H 腳則接地，因為前三個位元僅有 8 種組合，用不到 H 腳，所以最高位元 H 接地，之後的二進制位元依序輸入第二個 bin2bcd、第三個 bin2bcd、第四個 bin2bcd、第五個 bin2bcd 的 E 腳，其由 bin2bcd 前面的 GFE 輸入，而輸出接到下一個 bin2bcd 的 HGF，則相當於左移一個位元，即超過 5 則加 3 後再左移，完成二進碼轉 BCD 碼轉換，在此要注意其每一位元進位後一樣有可能超過 10，所以仍需有二進碼轉 BCD 碼的轉換，因此在其進位後超過三個位元之後仍需再經由超過 5 則加 3 後再左移的轉換。

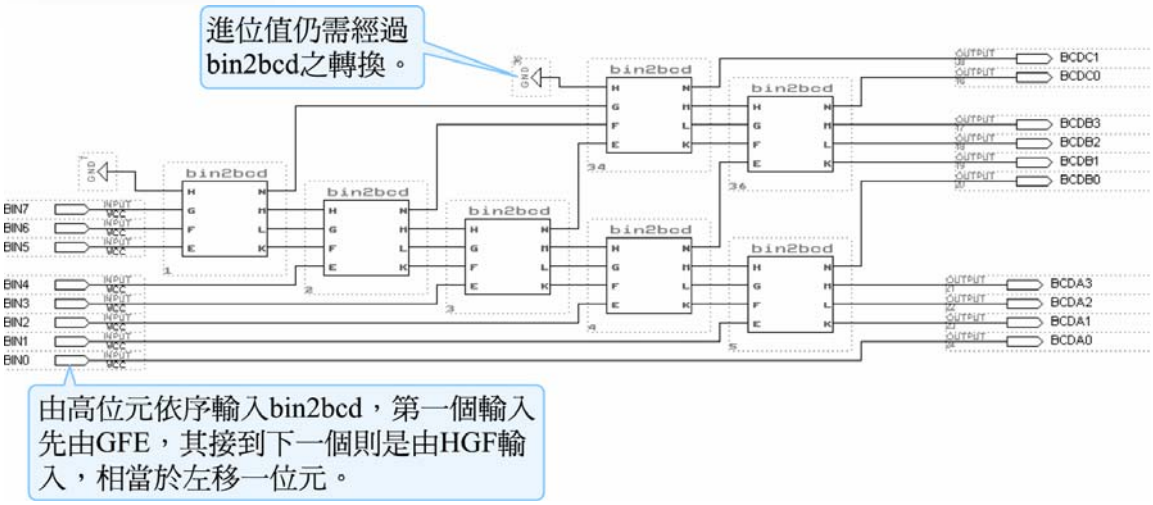

圖 3-6-13　八位元二進碼轉 BCD 碼電路圖

要以自製的超過 5 則加 3 二進碼轉 BCD 碼轉換器元件(symbol)做出 8 位元二進碼轉 BCD 碼電路時，步驟如下：

1. 建立一個新資料夾，取名 8 bin2bcd ，複製事先製妥 bin2bcd 元件的圖形檔 bin2bcd.gdf)以及元件檔(bin2bcd.sym)至該資料夾。
   執行 MAX+PLUS Ⅱ 軟體，開啟新的圖形編輯檔，叫出儲存 8 bin2bcd 資料夾的 bin2bcd 元件檔 bin2bcd.sym，利用它完成 8 位元二進碼轉 BCD 碼電路繪圖，存檔於 8 bin2bcd 資料夾內。如圖 3-6-13 所示(因為此圖形很大，若讀者覺得印刷之圖看不清楚，可於本書所附光碟中的 8 bin2bcd 資料夾找到此圖之檔案)，在此取名檔案名稱為 8bin2bcd。
2. File→Project→Set Project to Current File。
3. 編譯。

設計 8 位元二進碼轉 BCD 碼電路完成後，要知道其是否可正常執行，可執行**模擬功能測試。模擬步驟如下：**

1. 開啟 8bin2bcd.gdf 檔案，設為指定工作專案(File→Project→Set Project to Current File)，指定 CPLD 晶片(Assign→Device)，並編譯(MAX+PLUS Ⅱ→Compiler)。
2. 開啟新的波形編輯檔案，設定功能模擬結束時間(File→End Time)設定 2.56ms，設定格線間距(Options → Grid Size)設定 10 μs，顯示在視窗中適當大小格線(View→Fit in Window)。
   (因為在此輸入端共有 8 支腳，可計數 0~255，總共有 256 種變化，故需 2560 μs，即 2.56ms。)

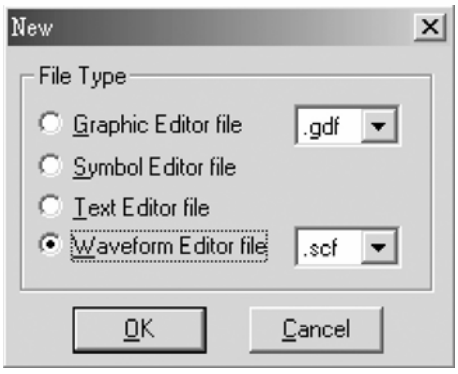

圖 3-6-14　開啟新檔視窗

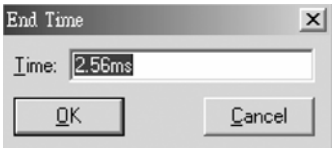

圖 3-6-15　模擬結束時間設定視窗

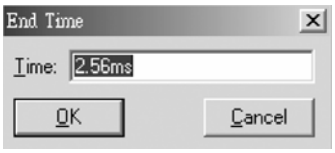

圖 3-6-16　模擬單位時間設定視窗

3. 儲存檔案(Save As)，檔名 8bin2bcd.scf，輸入節點(Node→Enter Nodes from SNF，按 List 及 =≥ ，OK)，編輯輸入信號，輸入端有 8 支腳，總共有 256 種組合，用計數時脈編輯 0~255 即可完成輸入信號編輯。

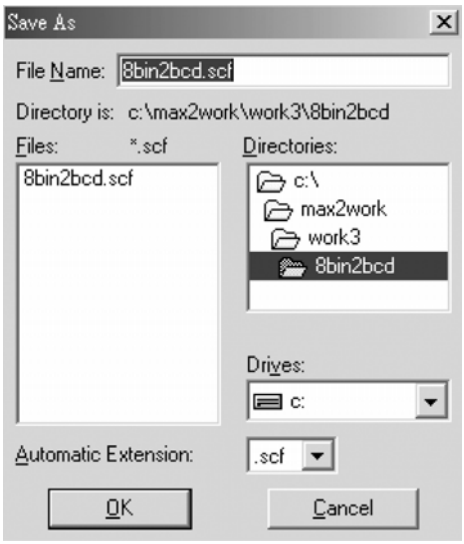

圖 3-6-17　儲存檔案視窗

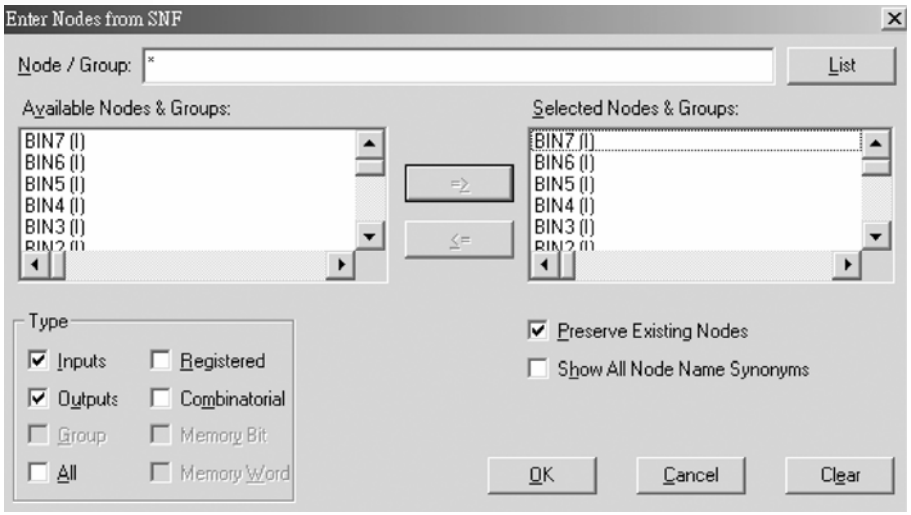

圖 3-6-18　輸出入節點選擇視窗

4. 執行模擬(MAX+PLUS Ⅱ → Simulator，Start)如圖 3-6-19 所示，所得波形模擬結果符合 8 位元二進碼轉 BCD 碼轉換器，代表我們製作的電路是正確可用的。

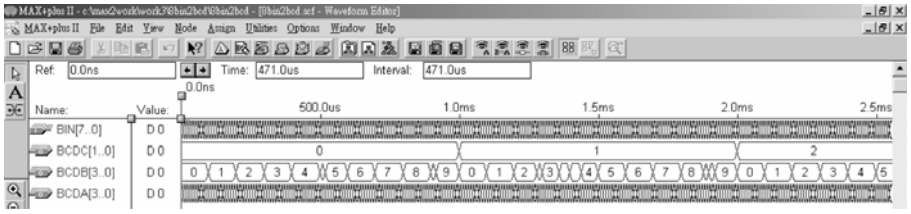

圖 3-6-19　模擬結果

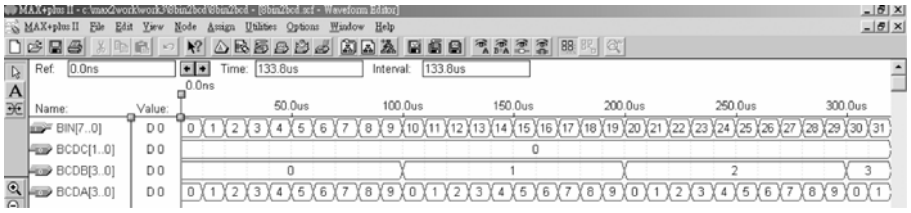

圖 3-6-20　模擬結果

3-7 動動腦 組合邏輯練習

1. 請自行製作半加器元件，完成後並用其繪製全加器圖形編輯檔，並以模擬功能且下載測試其是否正常。
2. 請依照全減器之真值表用邏輯閘電路繪出全減器之電路圖，並模擬且下載測試。
3. 請以全減器元件製作全減器電路，並模擬且下載其功能是否正常。
4. 請依照四位元減法原則，用四個全減器元件製作四位元全減器。
5. 請以八個全減器元件製作八位元全減器。
6. 請製作共陰極七段顯示器解碼器並模擬。
7. 請以八位元二進碼轉 BCD 碼配合三個七段顯示解碼器製作解碼顯示電路。