Experiment 1. 記憶體資料掃瞄輸出控制電路

【目的】

製作記憶體的輸出控制電路

i. 了解記憶體基本原理

ii. 利用 Quartus 建立記憶體模組

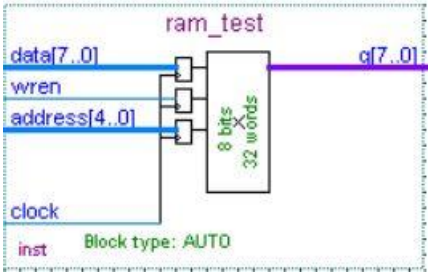
iii. 學習如何將 verilog code 建立成 symbol

【實驗背景】

Verilog 設計經驗

【原理與說明】

1. 記憶體原理



由 quartus megafunction 產生出來記憶體模組訊號有：輸入資料線(data)、寫入使能線(wren)、位址線(address)、clock、輸出資料線(q)。

此模組由兩部分所組成：(1) 記憶體訊號控制電路、(2) 記憶體單元。

記憶體訊號控制電路：

由三個 latch 所組成，利用 clock 正緣將資料、位址、寫入使能拴鎖供記憶體單元使用。

記憶體單元：

記憶體單元由記憶體訊號控制電路產生出來的控制訊號來做出動作

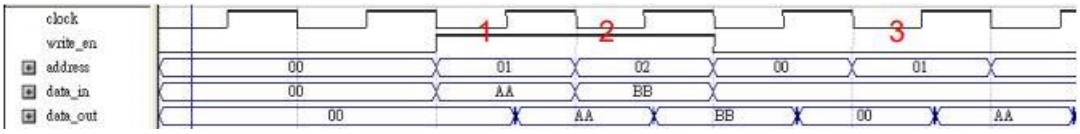
1. 寫入使能的控制線如果拉起，把寫入資料寫入位址線所指的位置。
2. 根據位址線送出位置中的內容值至輸出資料線(q)。

記憶體模組的整體動作：

寫入:當寫入使能被拉起時，等待 clock 正緣訊號，將輸入資料存放至位址線所指到的位置，完成寫入動作。

讀取: clock 正緣訊號來時，位址線所指到的位置中之內容值，傳送到輸出資料線端，完成讀取動作。

下圖為記憶體的波形模擬:



說明: 標示 1 號處，在 clock 正緣偵測到 write_en 是否為 1，於是將此時的資料 (AA)放入所指到的位址(01)上，標示 2 號處，同樣在 clock 正緣偵測到 write_en 是否為 1，於是將此時的資料(BB)放入所指到的位址(02)上，在標示 3 號處於 clock 正緣偵測到位址線傳送(01)，於是將(01)位址上的資料(AA)送出至 data_out。

2. 利用 Quartus 建立記憶體模組

步驟一： 建立專案 開啟 Quartus [File]-> New project wizard …

Key: 選擇路徑不可有中文，

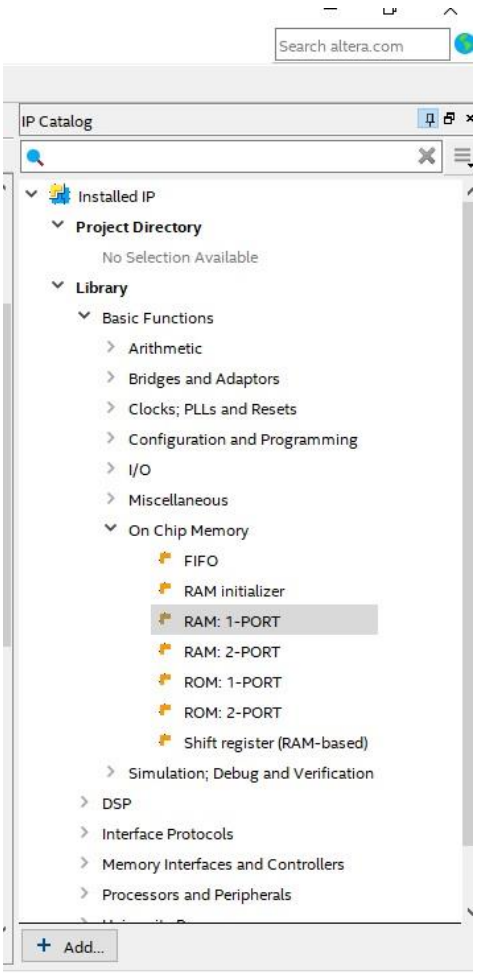
板子型號為 Cyclone V - **5CSEMA5F31C6** 建立完成之後，建立一個新檔案 [File]-> New 選擇 Block Diagram / Schematic File

步驟二： 使用 Mega Function 建立記憶體模組

Quartus 介面右側欄位如右圖

開啟 Library -> Basic Functions -> On Chip Memory

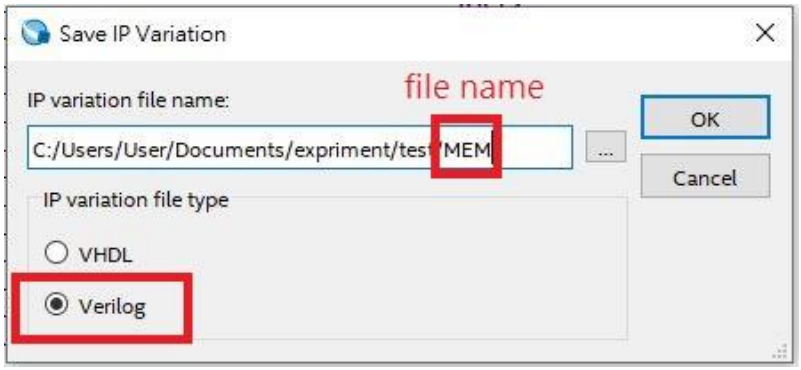
選擇 RAM1-PORT



如果右側的介面關閉了

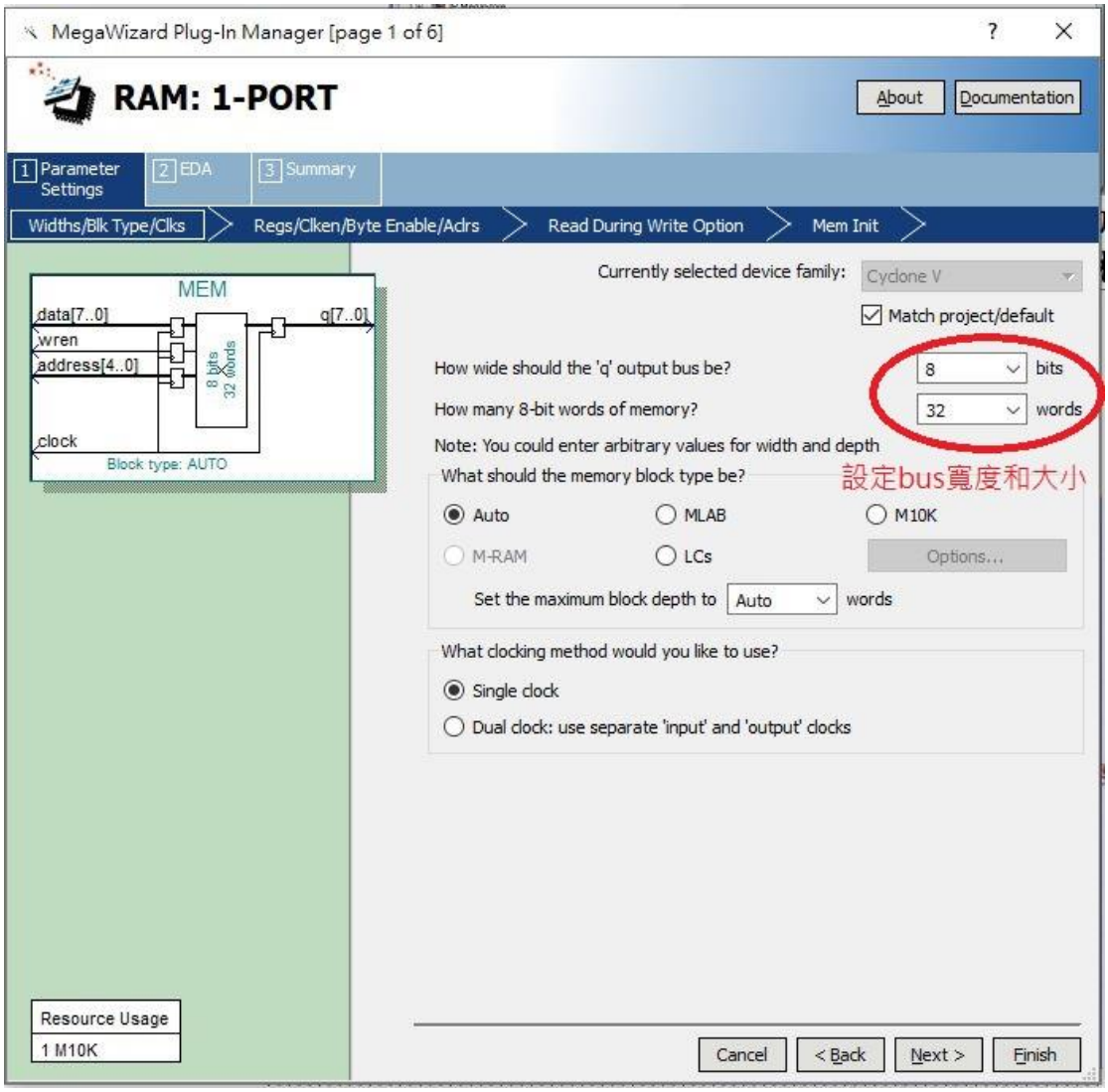
可以在工具列找到它

雙擊 RAM-PORT 後會出現下圖



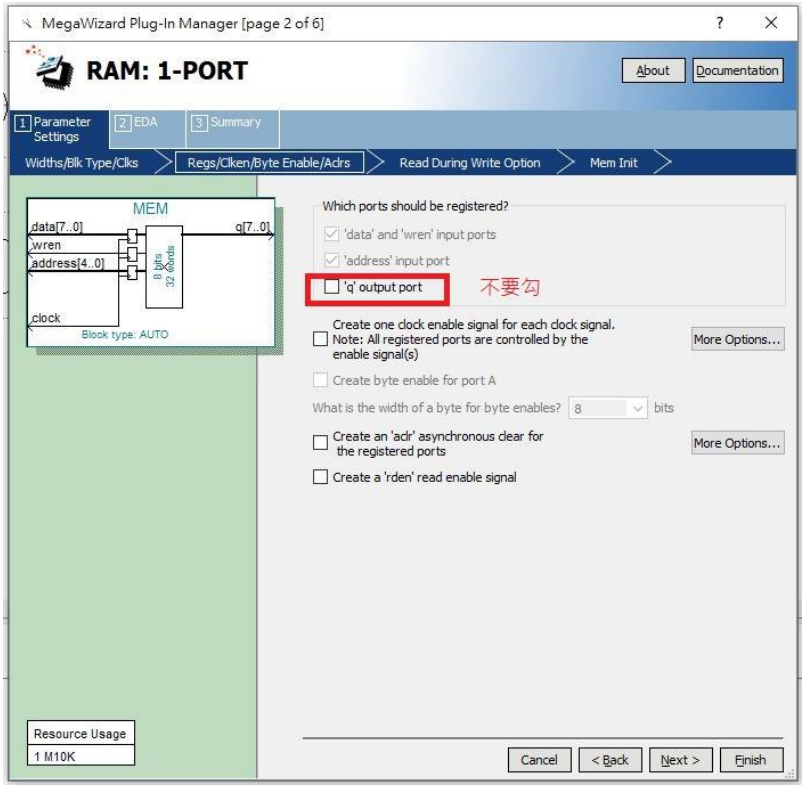
在檔案路徑中輸入檔案名稱，file type 選擇 Verilog，點擊 ok

註:記憶體的名稱不可以與檔名一樣

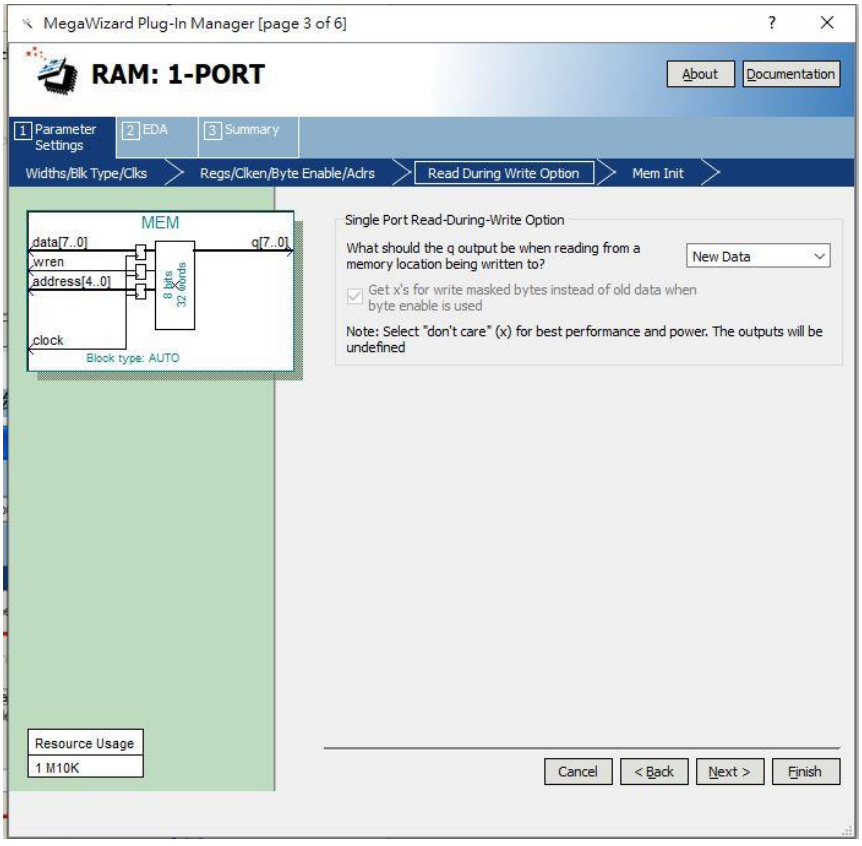


設定 bus 的寬度與記憶體要設多少個 words 在本次實驗 bus 寬度

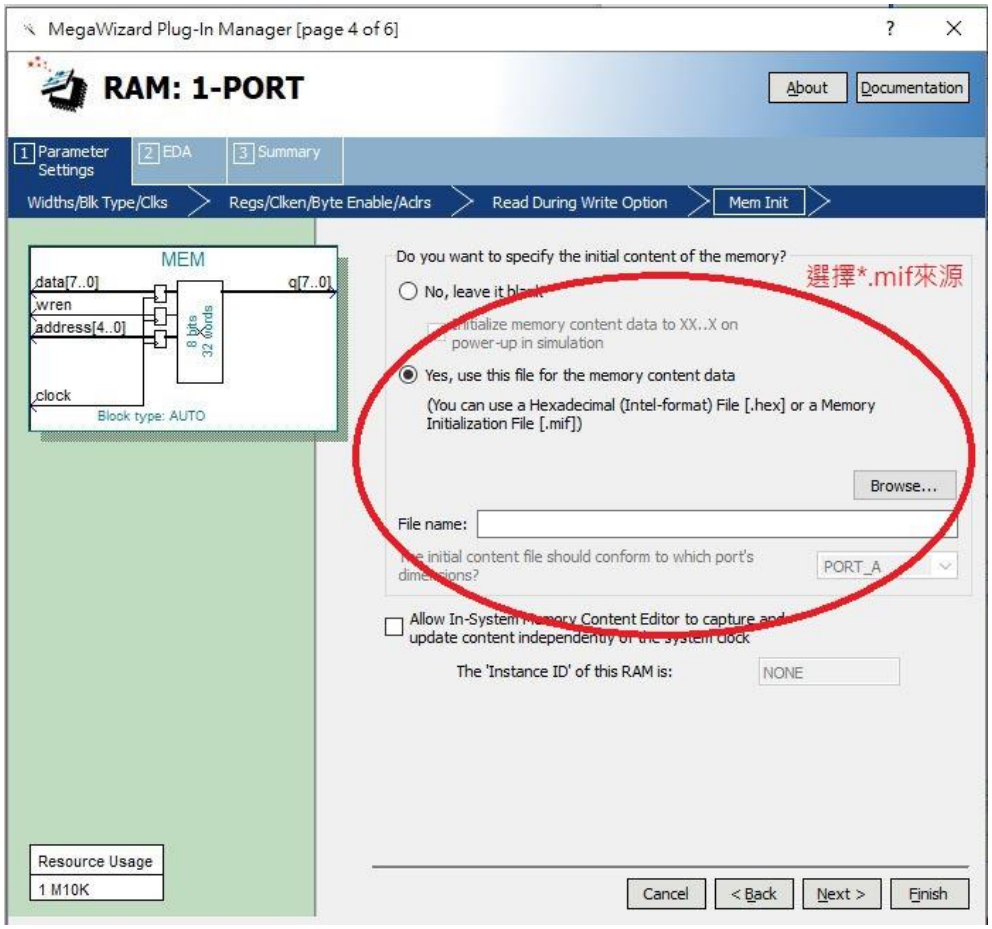
設定為 8 bits，因為要一次顯示 2 個十六進位的值



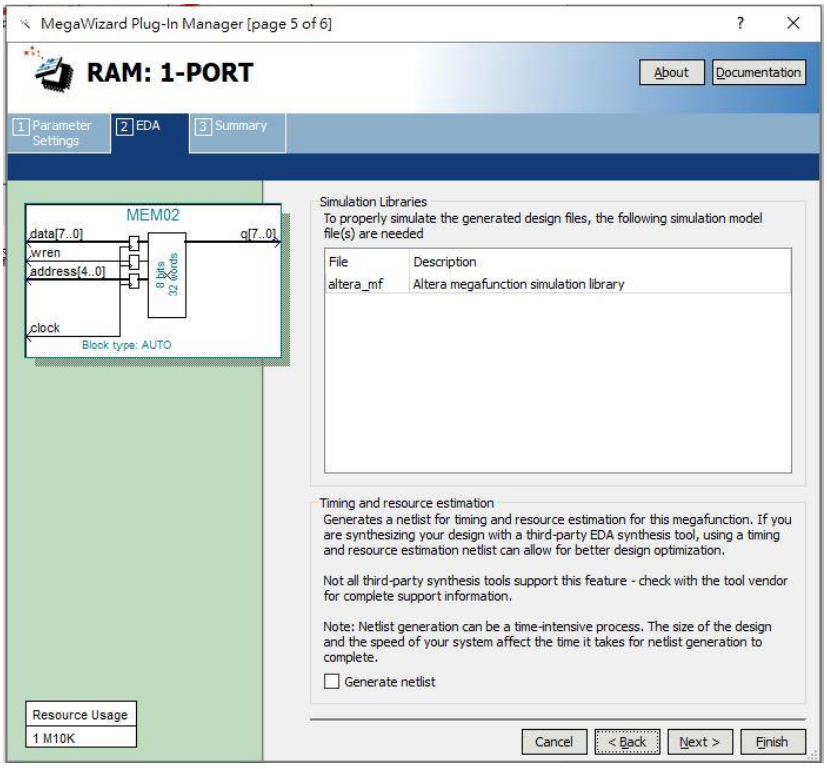
選擇輸出要不要有 latch，在本實驗不需要所以不打勾



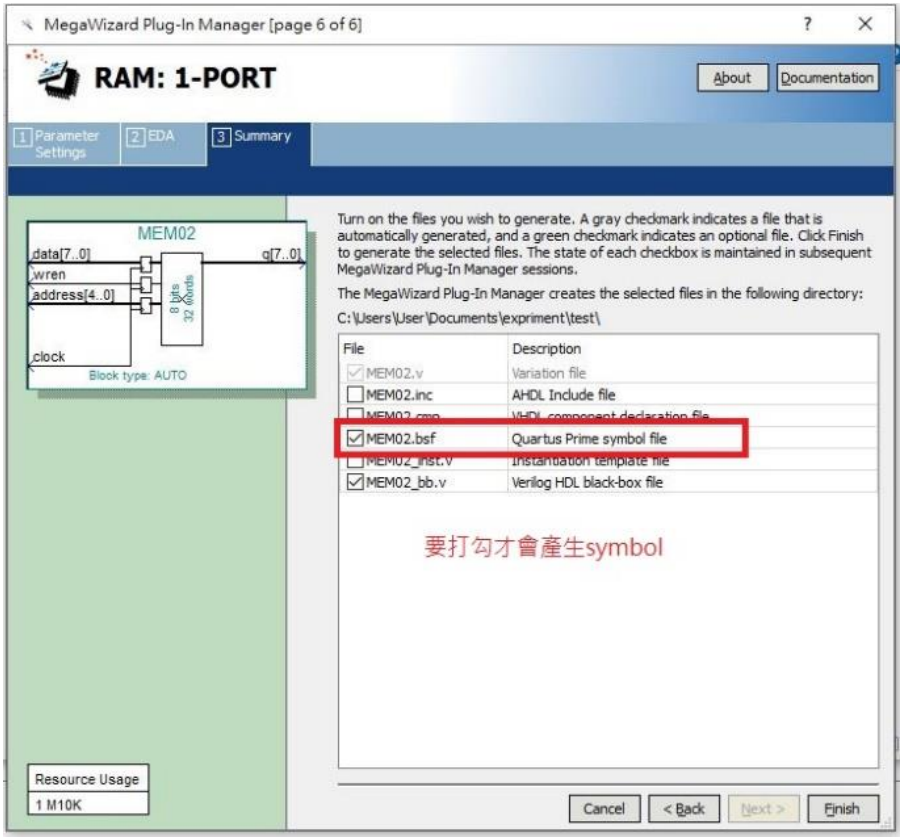
此頁沒差，因為本次實驗沒有要寫入 Memory



助教會提供 mif 檔來初始記憶體的內容值。 選取存放 mif 檔的檔案路徑

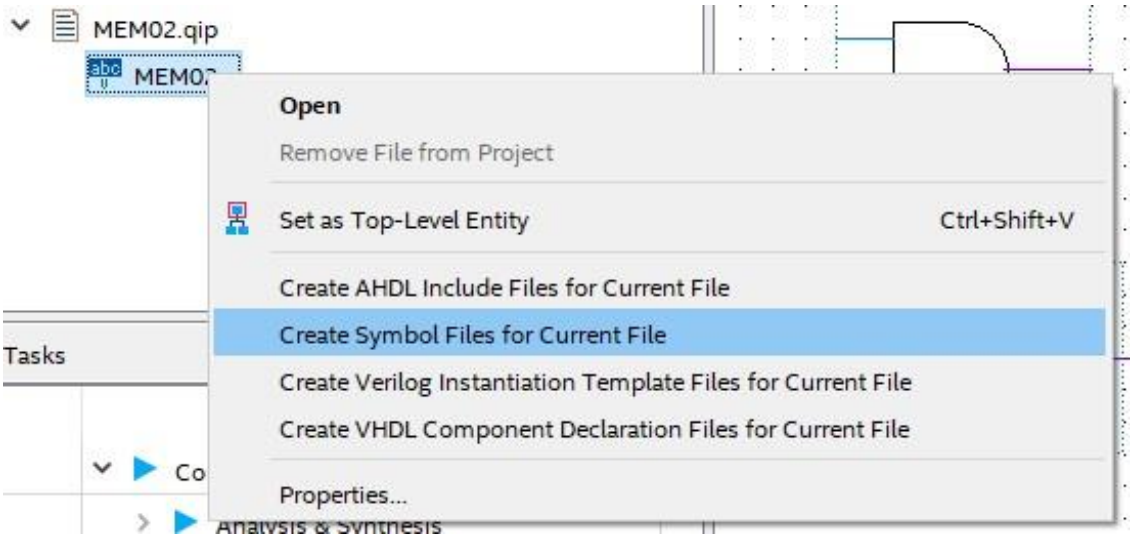


Page5 of 6 不用管他

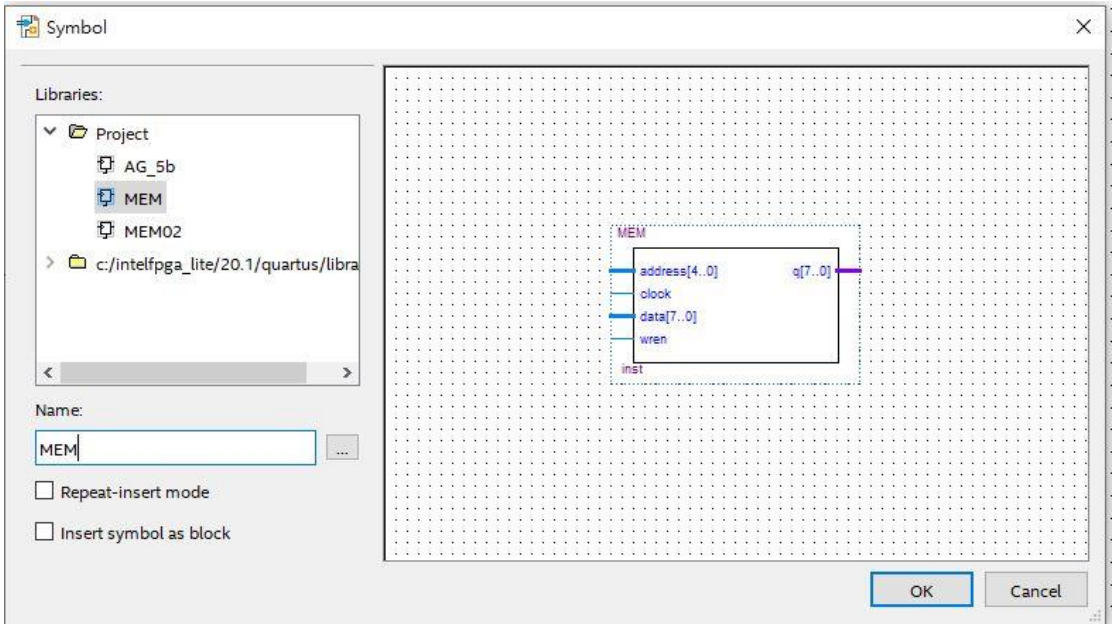


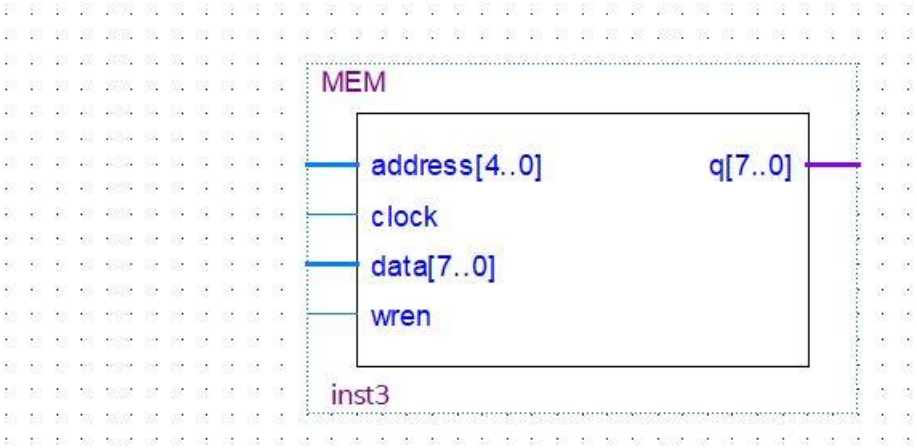
第 3 項選項要打勾才會產生 symbol，Block Diagram 才能引用 symbol。

如果這邊忘記勾選就點下 finish 的話也可以在左邊的 file 區找到記憶體模組的 verilog，一樣可以對它點右鍵，選擇”**Create Symbol Files for Current File**”



步驟三： 把產生出來的記憶體模組放在 bdf 檔中，完成建立。



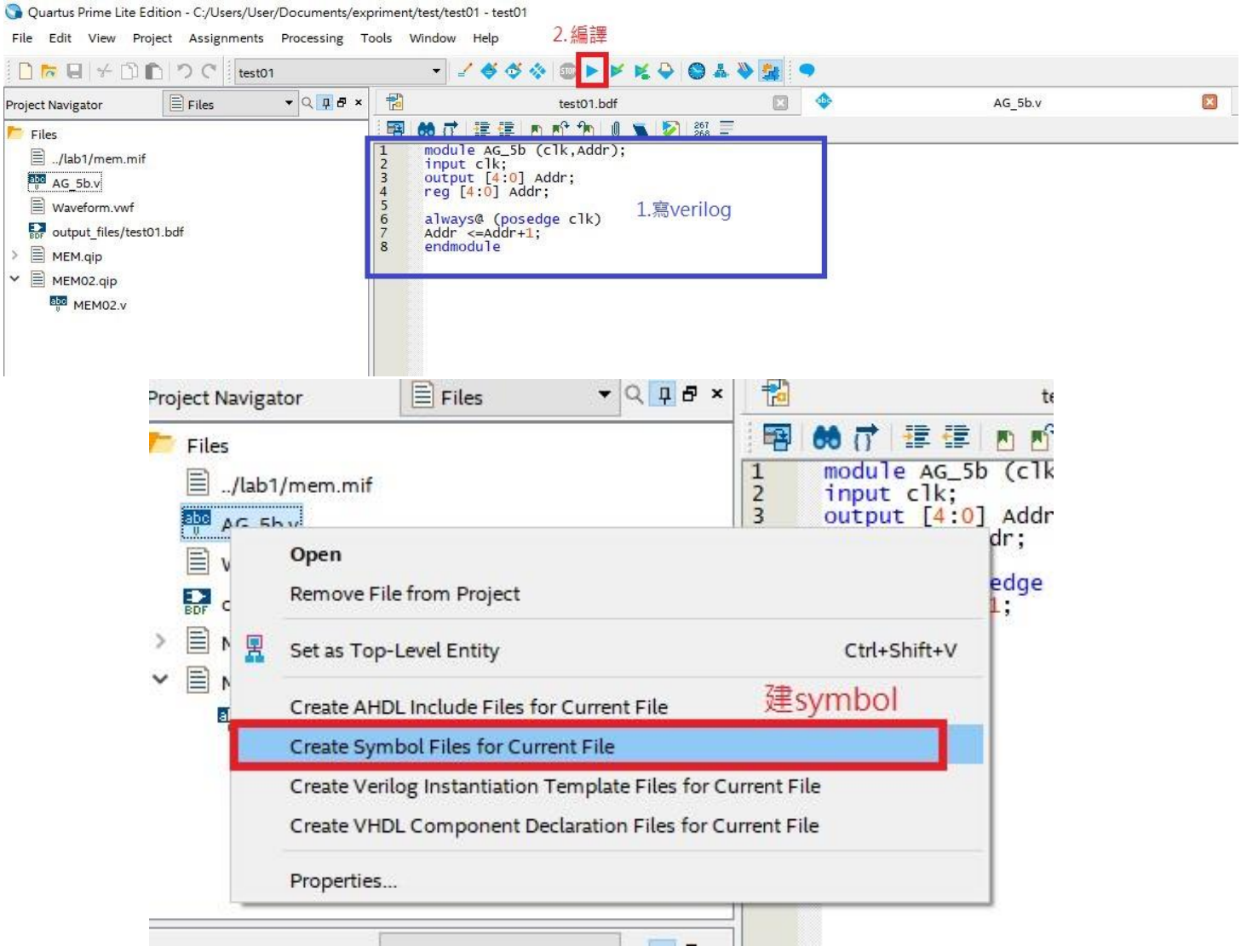


3. 利用 Quartus 將寫好的 Verilog code 建立成 symbol

隨著設計的電路越來越複雜，使用的 Verilog 模組也越來越多，為了方便程 式的設計與維護，會把設計好的 Verilog code 變成模組加入在 Block Diagram / Schematic File 中(如同加入記憶體)，如此一來，程式設計者可以非常容易的知道電路中包含了哪一些模組、模組之間是如何去連接與溝通。這一小結將介紹，如何把 Verilog code 轉變成 symbol file，並如何在 bdf 檔中插入

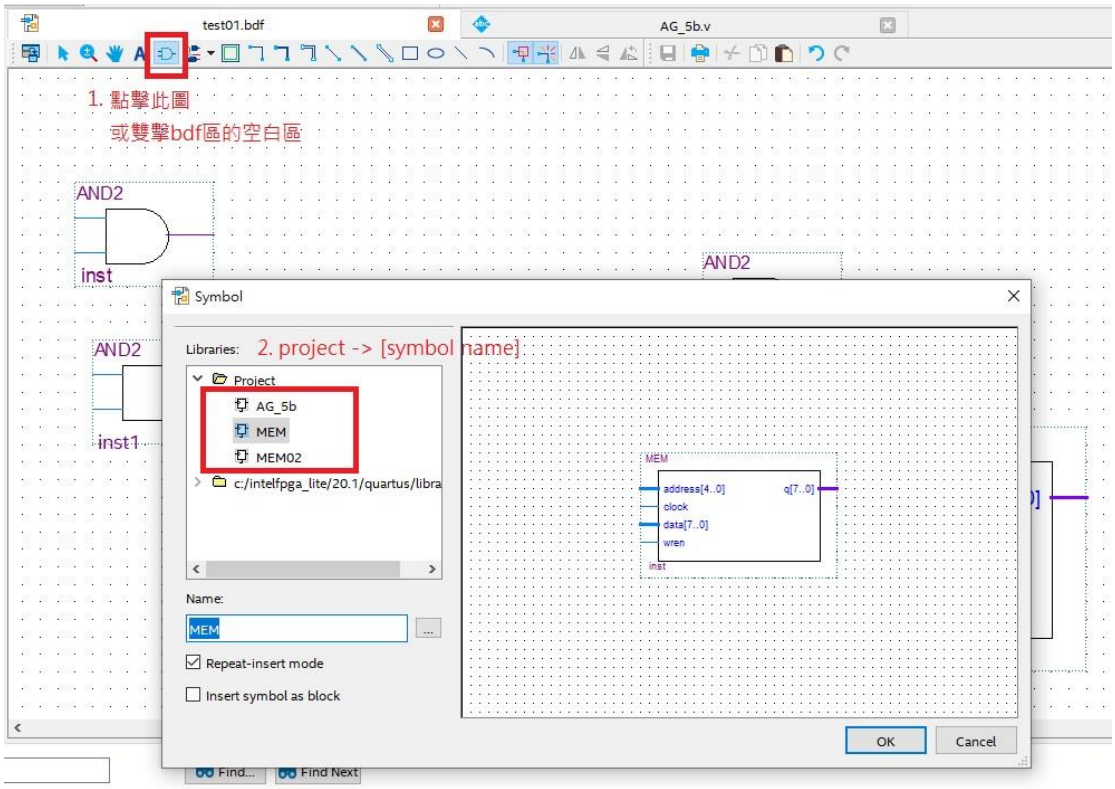
步驟一：

寫好 Verilog code 並編譯，編譯成功後回到此模組 Verilog code 畫面，點選後點右鍵，選擇”**Create Symbol Files for Current File**”



步驟二：

在 bdf file 畫面中，選擇插入 symbol，並在 project 的子項目中點選欲插入 symbol 名稱，即可以在 bdf 中看到插入的 symbol。

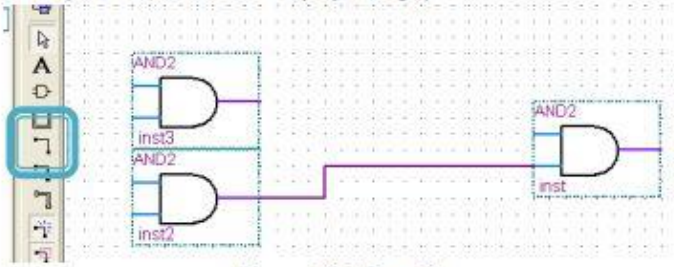


4. 電路拉線與輸入輸出之 Pin 腳

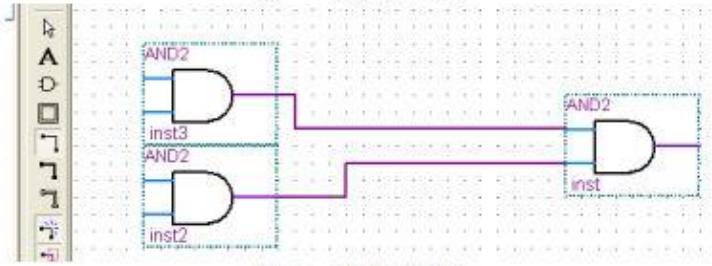
拉線：每個元件與元件之間需要用線路加以連接以傳遞訊號。

步驟：

如下圖，要把左上角 AND 閘的輸出連接到右邊 AND 閘的輸入，選擇下圖中藍色框框內的連接工具，將它們連接起來。

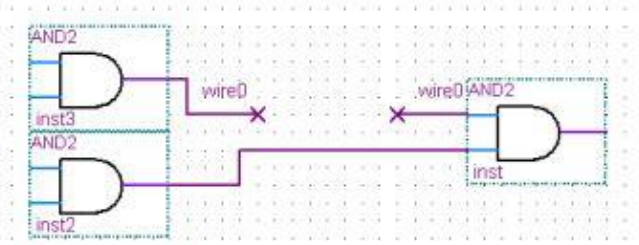


圖一 選擇工具



圖二 連接完成

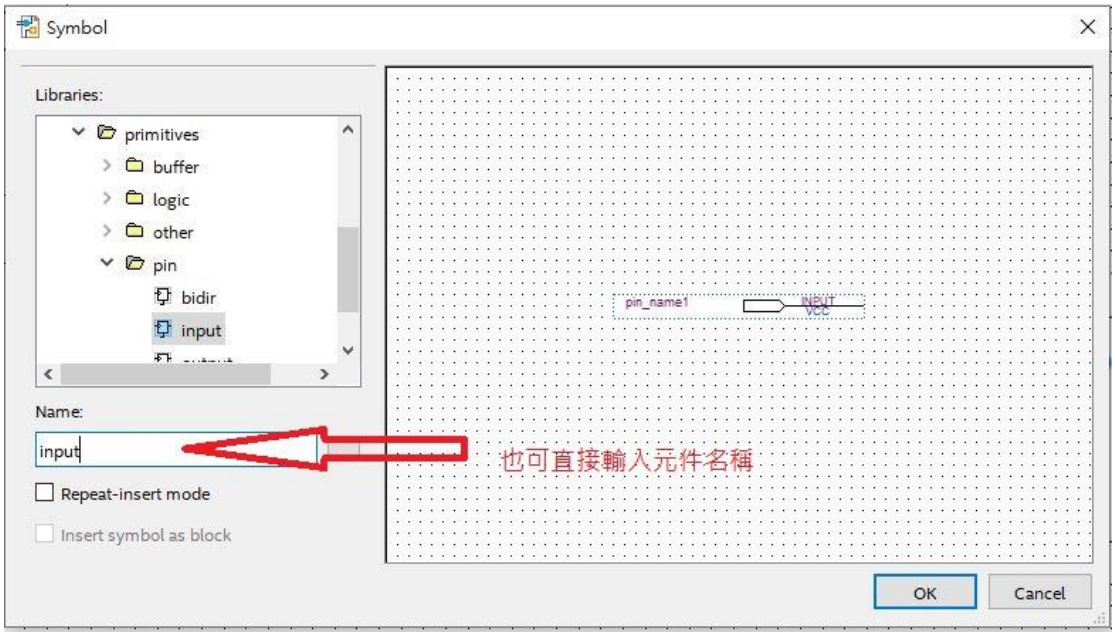
也可以將線命名，在線上點一下並入名字，如下圖，取名字的好處之一就是線在圖上不需要連接，可使畫面簡潔。



輸入輸出之 Pin 腳： 用於電路對外部的連接

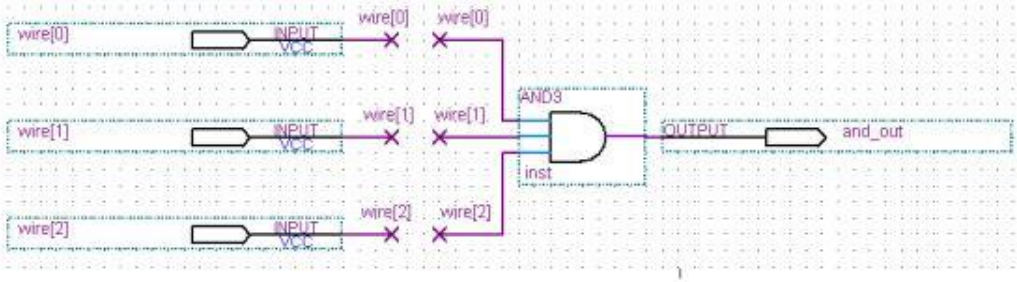
步驟一：

空白處點兩下叫出元件庫，在 primitives/pin/ 下面有三種 Pin 腳可選擇，分別為雙向、輸入與輸出。在使用元件庫時，也可以在 Name 欄中輸入所需的元件名稱，可以直接得到想要的元件。



步驟二：

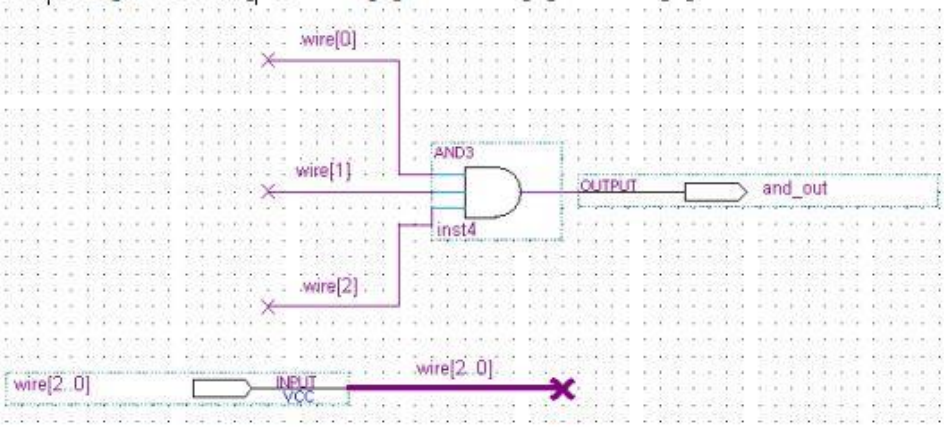
將輸入/輸出 PIN 放到適當位置並與電路連接。下面的圖示中，是一個 3bit AND 閘，AND 閘的輸出接 OUTPUT PORT，AND 閘的輸入連接由 3 個 INPUT PORT 。



用一條粗線來表示**多個訊號通道**我們稱之為 Bus，其命名格式為：

名稱[最高位元的編號.. 最低位元的編號]

如下圖所示，粗線就是一條 bus 為 wire[2..0] ，其連接的 INPUT PORT 是一個多腳位的埠，故命名方式亦同。若要從 Bus 內單獨拉出**一條**訊號，其命名方法為名稱[位元編號] 如 wire[0] 、 wire[1] 、wire[2] 。

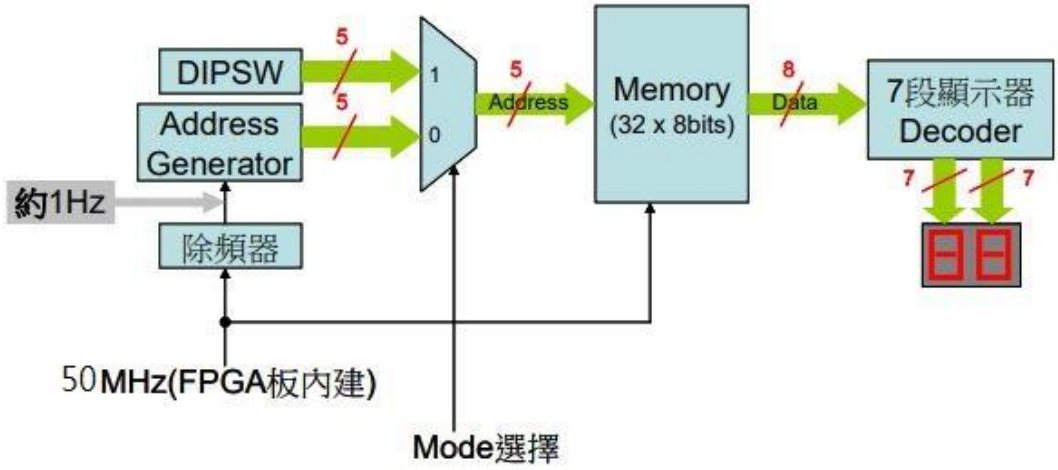


【實作】

實現記憶體輸出的控制電路，改變記憶體輸入的位址，在七段顯示器上顯示出對應位址的內容值。

此實驗分為兩種模式：

1. 由 DIPSW 當作位置的輸入，在七段顯示器看到記憶體對應位址的內容值。
2. 由系統內部自動產生位址，在七段顯示器看到記憶體對應位址的內容值。



此架構如上圖所示，包含：

除頻器：把板子上提供的 50M Hz 轉換成 1 Hz。

位址產生器：自動產生出記憶體位址。

Memory：由 megafunction 產生出來的記憶體模組。

7 segment decoder：把記憶體的內容值解碼顯示上 7 段顯示器上。

2 對 1 多工器：選擇記憶體位址的來源。模式 0 時，記憶體位址由位址產生器決定；模式 1 時，記憶體位址由 Dip Switch(DIPSW)決定。

課堂上實作內容：

1. 完成上述架構圖之所有模組，各模組請用 Verilog 語法來撰寫
2. 位址產生器設計為簡單的每個 clk 正緣位置+1 功能，並於數到 31 時歸 0
3. 顯示方面由 7 段顯示器顯示出 16 進位值

補充：

本實驗為記憶體輸出的控制電路，所以記憶體模組的寫入使能(wren/write enable)以及資料輸入(data)皆為接地即可。

額外 bonus：

做出任何有創意或額外的功能。例如：使用 10 進位值表示輸出、使用 LED 燈、LCD 顯示以及其他創意之設計。

實驗報告：報告內容包含：整體架構圖、各模組波形模擬 (解釋如何驗證功能正確)、各模組的 Verilog code 及註解(記憶體模組除外)、創意介紹 (有實作創意者)以及實驗心得