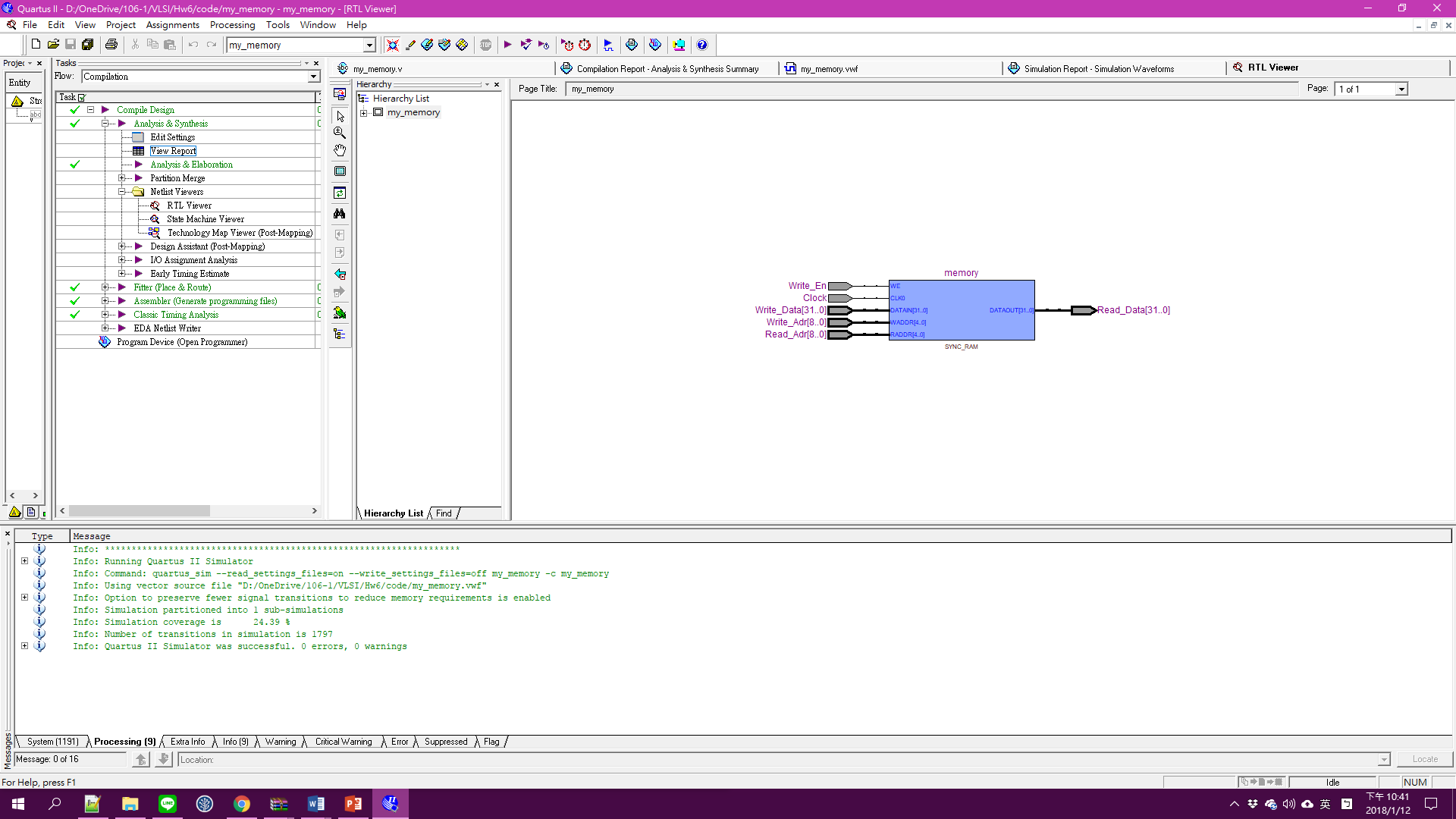
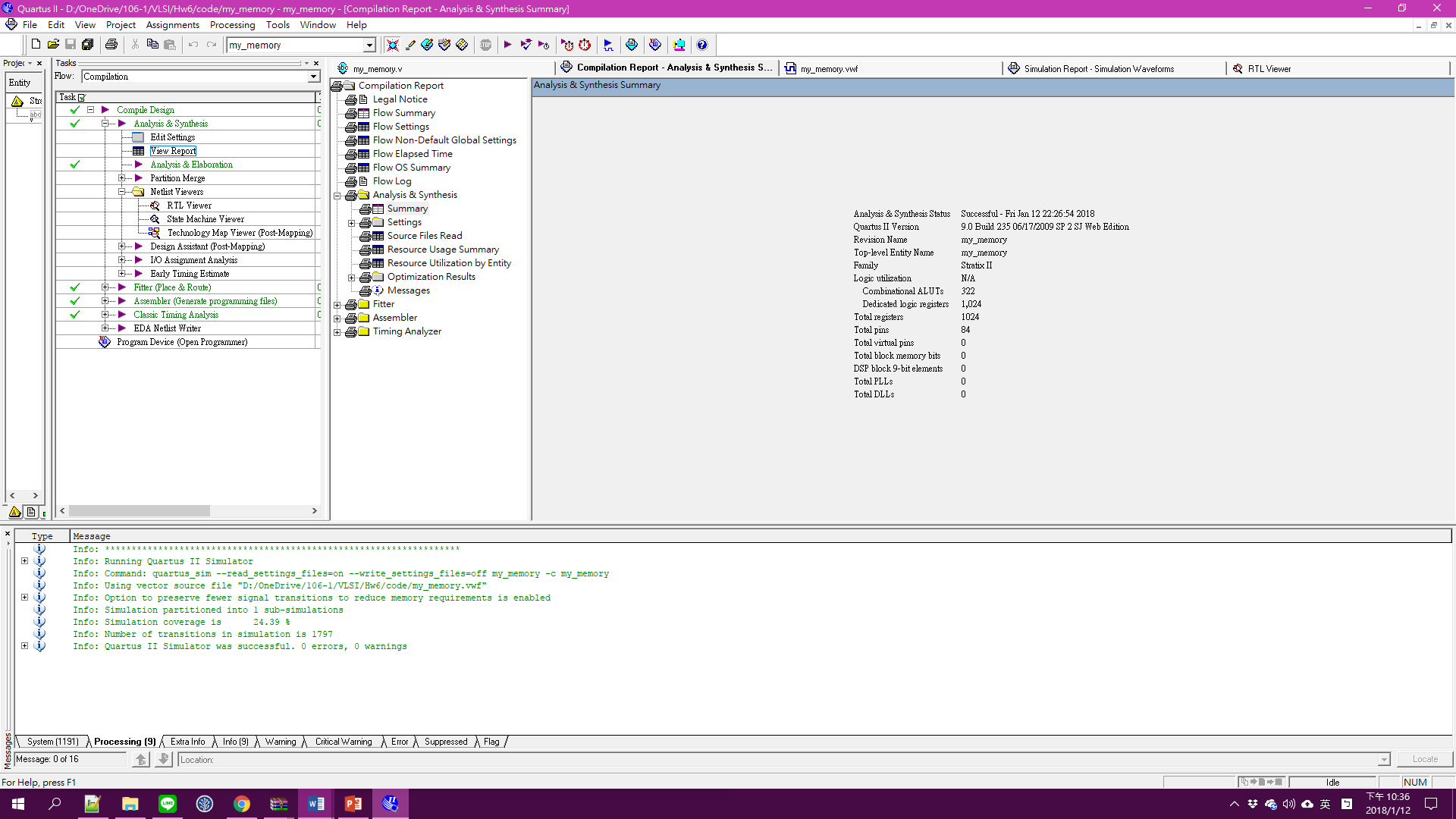
Hw 6 40243118S 王擎天

1. 設計電路說明(20%):
2. RAM using only LEs (or ALMs) by Verilog:

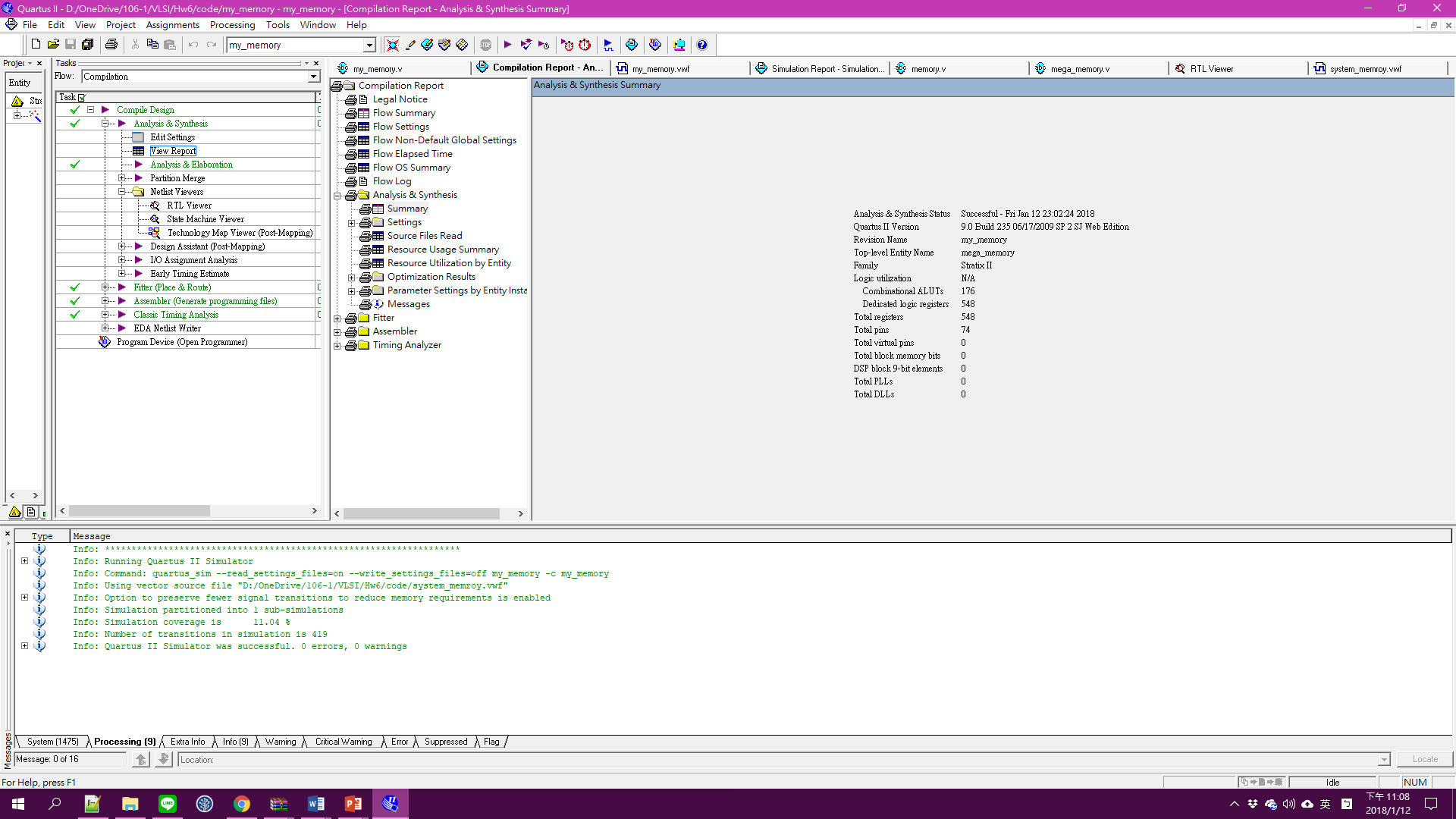


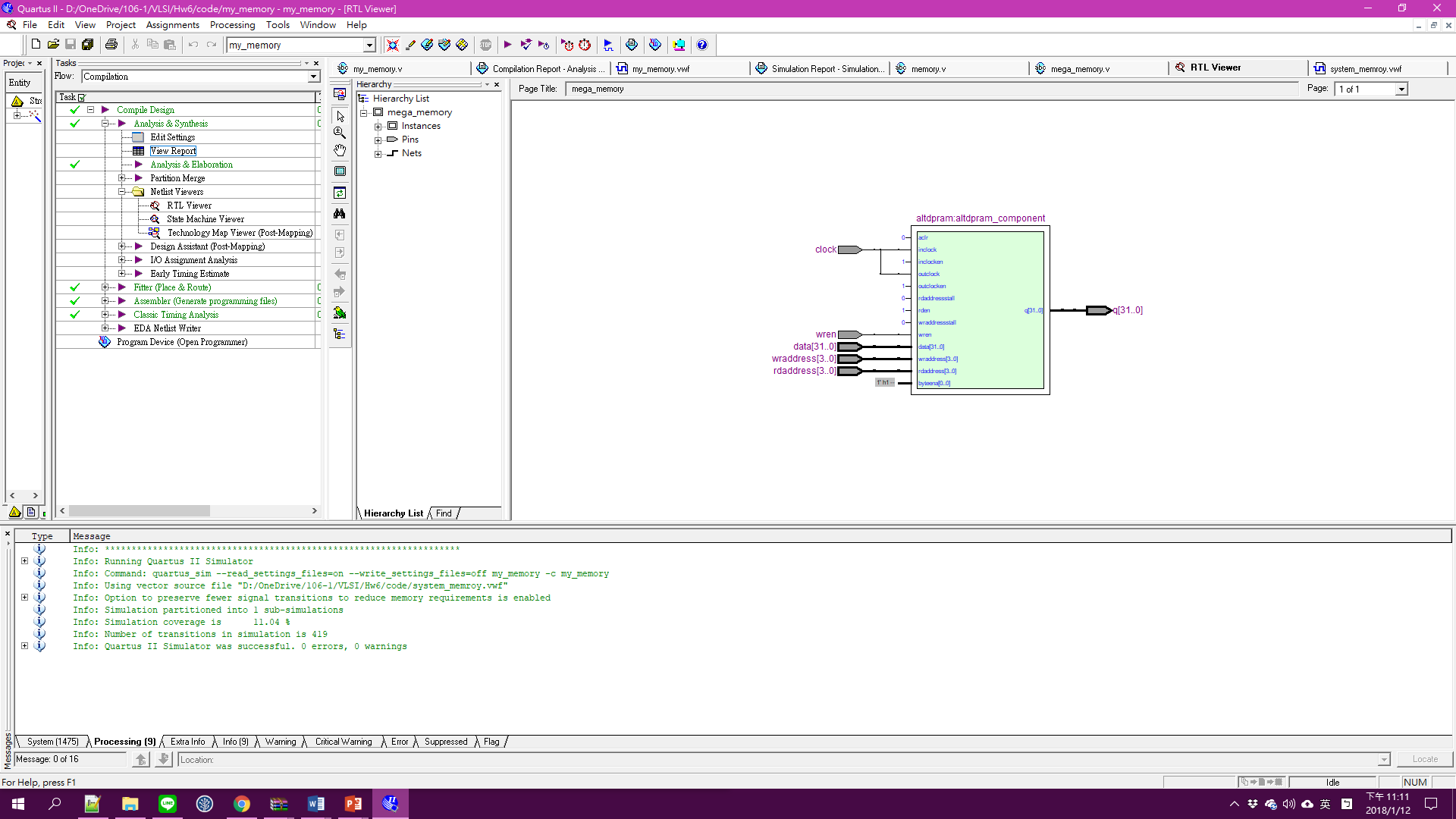
宣告一個memroy為512大小且每個位置均有32 bit,並利用Write\_En決定是否寫入,Write\_Data為寫入的資料,Write\_Adr為要寫入的位置, Read\_Adr為要讀入的位置.



**b.** Design the RAM using only embedded memory bits using MegaWizard Plug-In Manager

透過MegaWizard Plug-In Manager產生





1. 程式碼說明(30%):

(a) RAM using only LEs (or ALMs) by Verilog:

此程式先設計一個有512大小位置且每個位置均有32 bit的memory設計.以Write\_En當作可否寫入的訊號.若為一,則將memory中第Write\_Adr個資料以Write\_Data覆寫. Read\_Data則顯示memory中第Read\_Adr個資料.

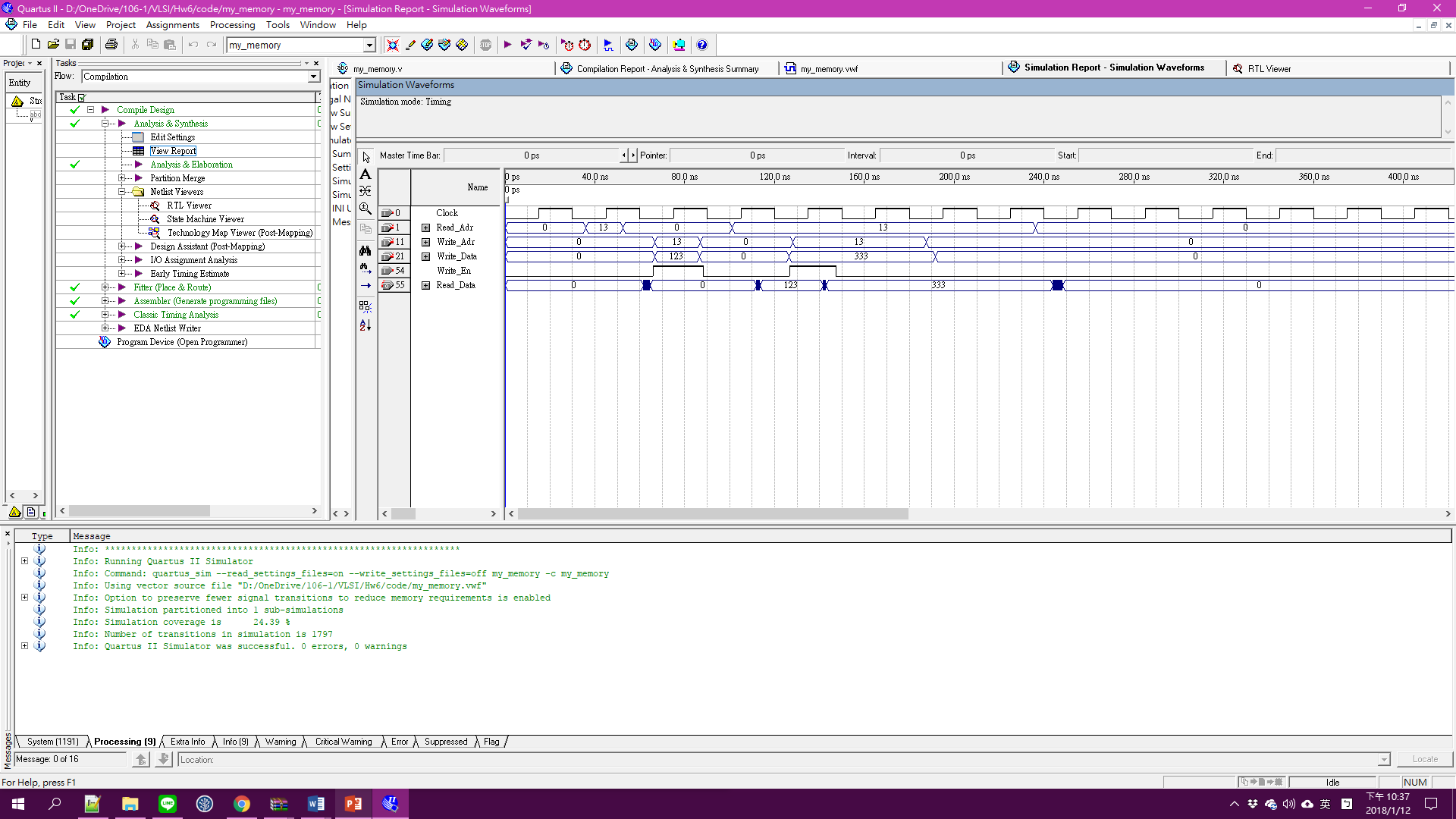
|  |
| --- |
| module my\_memory(Read\_Data,Read\_Adr,Write\_Adr,Write\_Data,Write\_En,Clock);  input [8:0] Read\_Adr,Write\_Adr;  input [31:0] Write\_Data;  input Write\_En,Clock;  output [31:0]Read\_Data;  reg [511:0] memory[31:0];  always @(posedge Clock)  begin  if(Write\_En)  begin  memory[Write\_Adr]<=Write\_Data;  end  end  assign Read\_Data=memory[Read\_Adr];  endmodule |

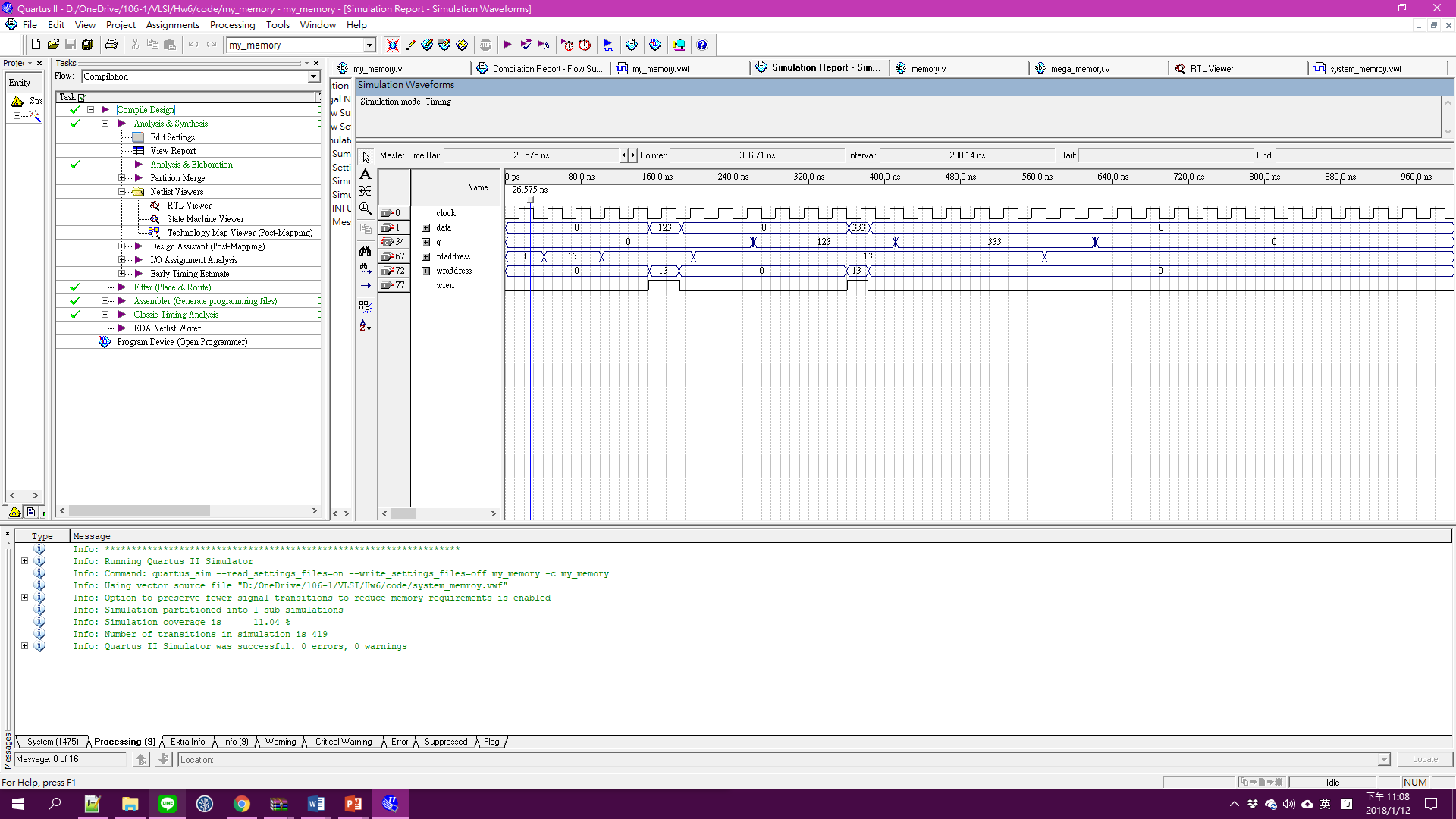
**(b)** Design the RAM using only embedded memory bits using MegaWizard Plug-In Manager

透過MegaWizard Plug-In Manager產生

1. 模擬結果說明(30%):

從此圖得知我一開始從記憶體13得出的資料為零,將123寫入記憶體13後,再將之讀出,可以看出輸出結果為123.後來我將333寫入記憶體13後,再將之讀出,可以看出輸出結果為333,代表記憶體中的數值成功被改變.

(a) RAM using only LEs (or ALMs) by Verilog:

**(b).** Design the RAM using only embedded memory bits using MegaWizard Plug-In Manager

1. 結論及心得(20%):

這次的練習分別使用了自己寫的記憶體和系統產生的記憶體.由模擬圖可得知結果皆為正確. 藉由MegaWizard Plug-In Manager所產生的記憶體程式可以省下自己寫記憶體的時間,也可以觀看系統產生的程式碼來學新知識.