HDL HW3 報告

B023040050 顏義洋

第一部分：撰寫組合語言將 RGB 轉為 YUV

檔案：YUV.v

R*0.299：

```
LDR r0, data_memory[0]
r3 = r0 >> 2
r4 = r0 >> 5
r5 = r0 >> 6
r6 = r0 >> 8
ADD r3,r3,r4
ADD r5,r5,r6
ADD r3,r3,r5
STR r3,data_memory[12]
```

G*0.589：

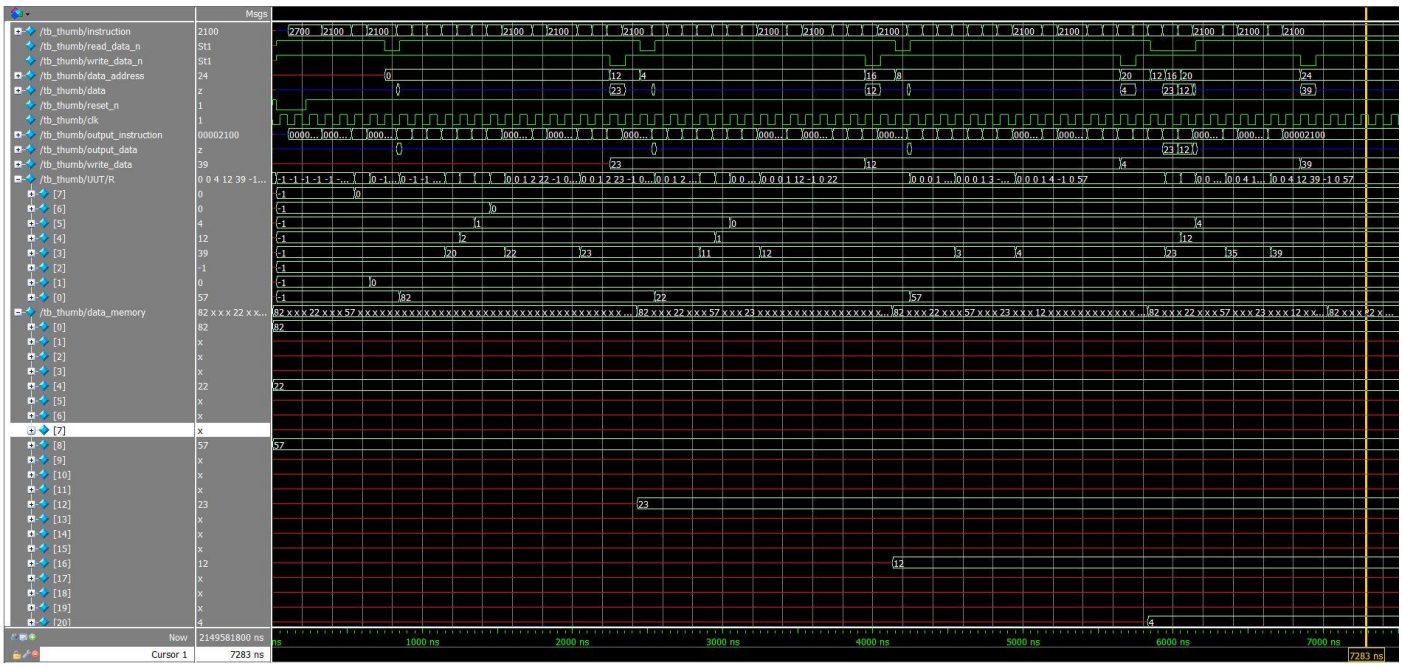
```
LDR r0, data_memory[4]
r3 = r0 >> 1
r4 = r0 >> 4
r5 = r0 >> 6
r6 = r0 >> 7
ADD r3,r3,r4
ADD r5,r5,r6
ADD r3,r3,r5
STR r3,data_memory[16]
```

B*0.114：

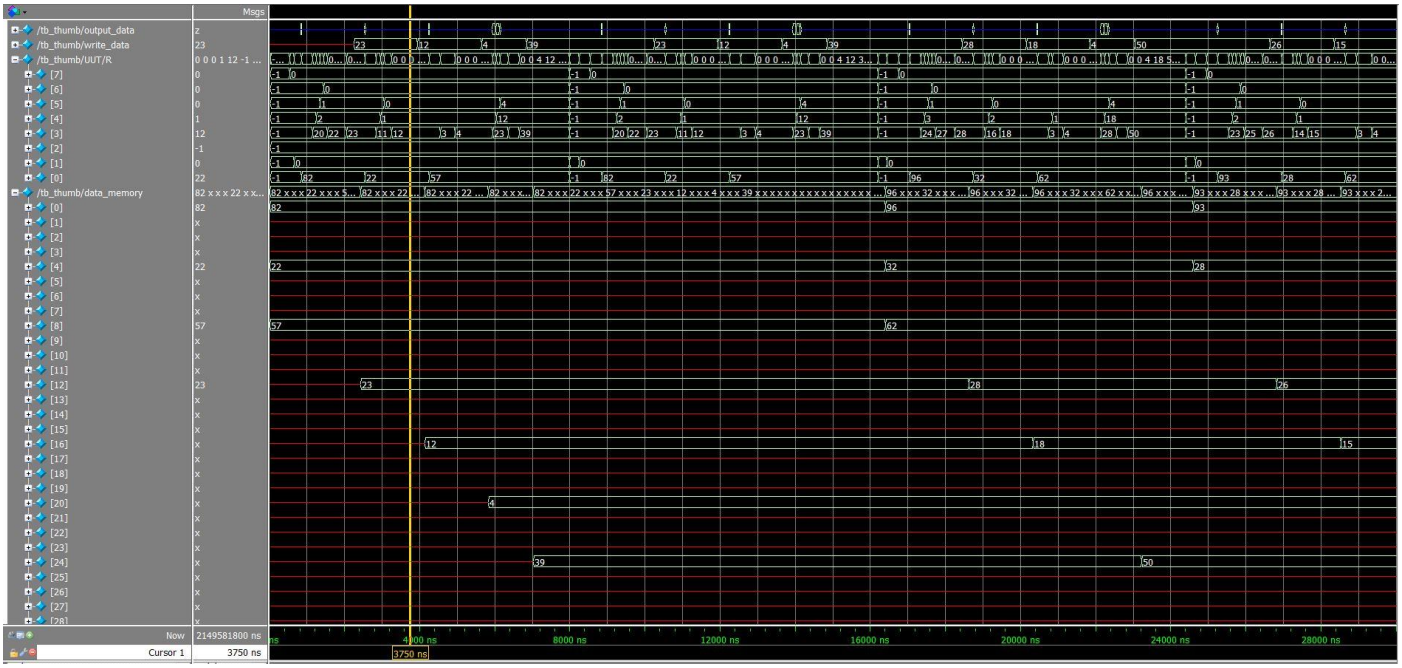
```
LDR r0, data_memory[8]
r3 = r0 >> 4
r4 = r0 >> 5
r5 = r0 >> 6
r6 = r0 >> 7
ADD r3,r3,r4
ADD r5,r5,r6
ADD r3,r3,r5
STR r3,data_memory[20]
```

Y=R+G+B：

LDR r3, data_memory[12]
LDR r4, data_memory[16]
LDR r5, data_memory[20]
ADD r3,r3,r4
ADD r3,r3,r5
STR r3,data_memory[24]



▲圖一



▲圖二

運算流程：

(1)把 R load 到 r0，再對 r0 作數個 shift 分別存到不同 reg，最後把總合存到 r3

(2)把 R*0.299 的近似值 store 到 data_memory[12]

(3)G*0.589、B*0.114 的方法同(1)(2)，結果分別 store 到 data_memory[16]，data_memory[20]

(4)計算 Y = R*0.299 + G*0.589 + B*0.114 存到 data_memory[24]

Testbench 流程：

(1) 開啟 input 的 BMP，讀取 header 取得 bmp 的長跟寬還有檔案大小和 pixel data 的起始點。一次讀 300 個 byte，一次放 3 個 byte：把 R、G、B 分別塞到 data_memory[0]，data_memory[4] data_memory[8]

(2) 等運算完再 reset，此時 Y 值已經在 data_memory[24]

(3) 一次寫 3 個 byte 的 Y 值進圖片檔

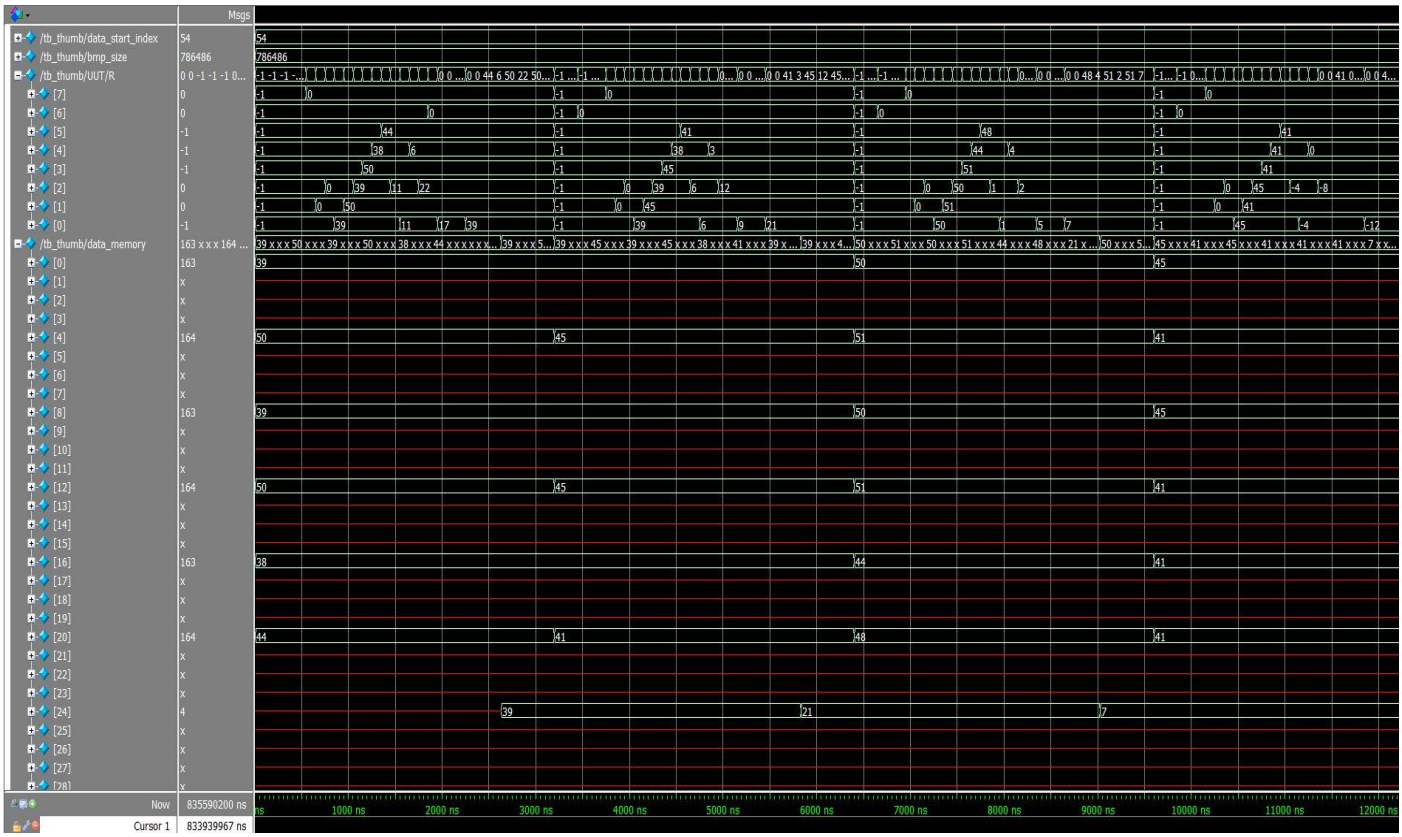
(4) LOOP 以上(2)~(4)的步驟直到讀檔結束

以上流程可以參考圖一和圖二。

第二部分：撰寫組合語言執行 sobel 運算

檔案：sobel.v

LDR r0, data_memory[0]
LDR r1, data_memory[4]
LDR r2, data_memory[8]
LDR r3, data_memory[12]
LDR r4, data_memory[16]
LDR r5, data_memory[20]
SUB r2,r3,r2;
SUB r0,r1,r0;
SUB r4,r5,r4;
LSL r2,#2
ADD r0,r0,r4
ADD r0,r0,r2
STR r0,data_memory[24]



運算流程：

r0	0	r1
r2	0	r3
r4	0	r5

(1) 將 sobel 所需要的 Y 值分別從 data_memory[0,4,8,12,16,20]load 進 r0~r5 內

(2) 利用加減和 shift 運算，算出所需要的梯度值

(3) 把算出來的梯度值放入 data_memory[24]

Testbench 流程：

(1) 開啟 input 的 BMP，讀取 header 取得 bmp 的長跟寬還有檔案大小和 pixel data 的起始點。

(2) 開個很大個陣列把所有的 BMP pixel 的 data 先讀進來

(3) 最外圍的那一圈 pixel 不做 sobel 運算，寫檔時直接寫黑色

(4) 把相對應的 pixel 放進 data_memory

(5) 運算完後在 reset，此時 sobel 算出來的梯度已經在 data_memory[24]

(6) 因為在 thumb 內沒有類似 arm conditional execution 的指令，所以不方便在 thumb 內就直接算出要寫白色還是黑色的 pixel 進圖片

(7) 將出來的梯度取絕對值

(8) 選定 64 為標準，梯度大於 64 寫白色(255,255,255)進圖片 ，反之填入黑色(0,0,0,)

(9) 用兩層 LOOP 執行以上(3)~(8)的步驟直到所有的 pixel 都算完

第三部分：

刪除 thumb.v 裡面在此份作業 sobel 運算用不到的指令，我將 ADD、SUB、LDR、STR、MOV 系列之外的程式碼刪除，減少約 300 行左右

再用同一份 testbench 驗證所得到的結果與原本相同

比較原始與優化的 thumb processor 所合成出來的 timing 和 area

Area 比較：

原始 thumb.v(檔案 thumb_area.txt)

Number of ports:	117
Number of nets:	6154
Number of cells:	5398
Number of combinational cells:	4612
Number of sequential cells:	768
Number of macros/black boxes:	0
Number of buf/inv:	375
Number of references:	62

Combinational area:	93778.330076
Buf/Inv area:	3119.616081
Noncombinational area:	20509.286325
Macro/Black Box area:	0.000000
Net Interconnect area:	556216.691343

Total cell area:	114287.616402
Total area:	670504.307744

刪減後 modified_thumb(檔案 modified_thumb_area.txt)

Number of ports:	117
Number of nets:	3841
Number of cells:	3394
Number of combinational cells:	2665
Number of sequential cells:	719
Number of macros/black boxes:	0
Number of buf/inv:	191
Number of references:	48

Combinational area:	46167.552201
Buf/Inv area:	1184.256031
Noncombinational area:	19264.204739
Macro/Black Box area:	0.000000
Net Interconnect area:	372709.291742

Total cell area:	65431.756940
Total area:	438141.048681

Timing Path 比較：

原始 thumb.v(檔案 thumb_timing.txt)

```
-------------------------------------------
  thumb                  tc8000                 saed90nm_typ

  Point                                                 Incr       Path
  -----------------------------------------------------------
  write_data_n_reg/CLK  (DFFASX1)                       0.00       0.00 r
  write_data_n_reg/QN  (DFFASX1)                        0.24       0.24 f
  data_tri[15]/INOUT2 (BSLEX1)                          0.28       0.52 f
  data[15] (inout)                                      0.00       0.52 f
  data arrival time                                                0.52
  -----------------------------------------------------------
  (Path is unconstrained)
```

刪減後 modified_thumb(檔案 modified_thumb_timing.txt)

```
-------------------------------------------
  thumb                  tc8000                 saed90nm_typ

  Point                                                 Incr       Path
  -----------------------------------------------------------
  DR_reg[31]/CLK (DFFX1)                                0.00       0.00 r
  DR_reg[31]/Q (DFFX1)                                  0.22       0.22 f
  data_tri[31]/INOUT2 (BSLEX1)                          0.78       1.00 f
  data[31] (inout)                                      0.00       1.00 f
  data arrival time                                                1.00
  -----------------------------------------------------------
  (Path is unconstrained)
```

第四部分：

利用 DV 合成之 thumb proceesor 的 gatelevel 檔案用相同 testbench 進行驗證，這部分沒有成功，原因可能是因為 timing delay 給的不夠正確，也有可能是作者原本提供的 testbench 沒有考量到硬體的 delay 層面。

第五部分：

利用 FPGA 驗證，成功合成出 gatelevel 檔案，但是用同一個 testbech 進行驗證也是錯誤。

寫作業遇到的所有問題 or 心得：

(1) 一開始誤會題目的意思，以為所有的步驟包含開檔寫檔都要用組合語言寫，經由詢問助教後才知道只有運算的部分，運算部份的組語對我們並沒有很困難。

(2) 雖然在計算機組織學過 hazard 的觀念，不過在寫組語丟進 thumb 跑的時候並沒有想到要考慮 hazard 的問題，所以有遇到 data hazard 的問題，還有遇到 structural hazard 的問題，不能有連續兩行做 LDR/STR，解決方式都是插入不重要的指令在中間來避免 hazard。

(3) 合成出來的 thumb processor 用相同的 testbench 怎麼改都跑不出結果，通常在第一個指令要 WB 的那一個 clock 正緣，thumb 內部的 regfile 就變成紅線，原因不明。嘗試過將 Clock PERIOD1, READ_DELAY, WRITE_DELAY, STABLE_TIME 的延遲時間都加長到 10 倍，結果仍是錯誤。