2016 計算機組織
實驗結報評分標準

2016/9/26 (一)

Advisor: 陳中和教授

T.A.　:黃冠霖、鄭基漢、蘇郁翔、金育涵、曾微中

結報注意事項

- 檔名:
  - LABx_GROUPy (x : 第 x 個實驗、 y : 第 y組)
- 格式:
  - 採用 Microsoft Word 文件格式
- 內容:
  - 參照實驗室網站 : http://caslab.ee.ncku.edu.tw/dokuwiki/_media/course:co:104a:report_example:ex.docx
- 有問題的話…
  - 請寄信給助教 ☺

分數配置

- 實作題:
  - 70 ~ 100 分
- 挑戰題:
  - 加 0 ~ 20 分
- 遲交:
  - 扣 20 分 / 週
- 期末都沒交:
  - 0分！

評分標準

- 分三等級
  - A: 內容較完整 (題目說明、實驗方法、結果圖+說明等...)
  - B: 有結果圖及說明
  - C:只有結果圖
- 注意
  - 只貼 Code 沒說明不算 A 等級唷!!!

A 等級

(1)實作一：call function

1. 題目

利用 call function 的方式，呼叫 sum function，執行一個簡單的加法程式。

原始 c code：

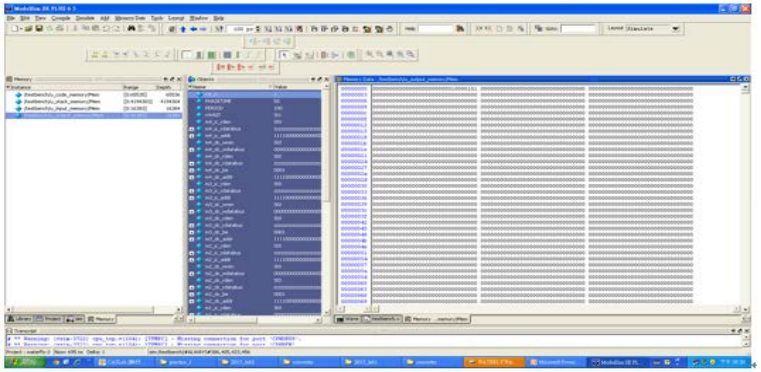
max_return.c

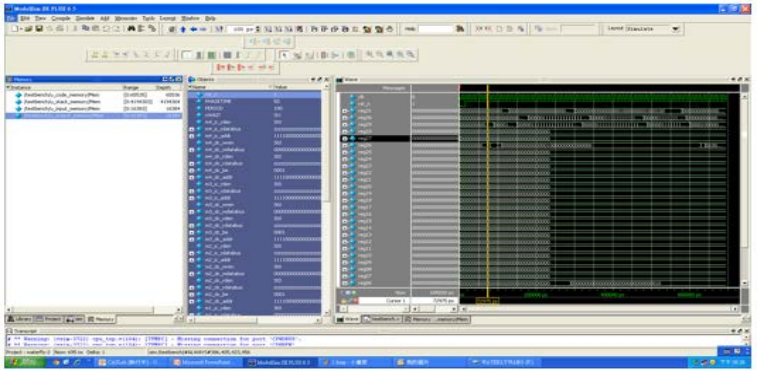
```
int sum(int,int,int);
int main()
{
    int a=44,b=87,c=2;
    volatile int* n = (int*) 0x20000000;
    //int *n;
    *n=sum(a,b,c);
    //printf("%d",*n);
    return 0;
}

int sum(int a,int b,int c)
{
    int n;
    n=a+b+c;
    return n;
}
```

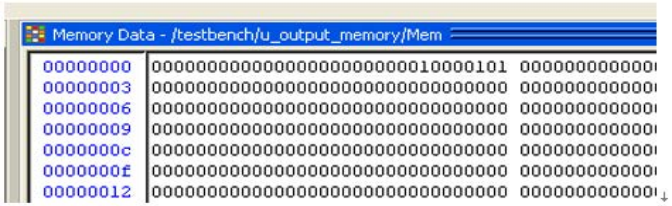
2. 實現方式

用 linux 的 Terminal，先把 sde compiler 位置給 linux，轉我們的 c code 至 assembly code，產生 object 檔，壓縮成 image，再利用 conv2mif 把檔案轉換成 Verilog 記憶體格式 mif 檔，放到 Windows 系統的 RTL code 目錄底下，與 system.mif 系統初值設定 combine 成 code.mif，最後用 Modelsim 看儲存結果方便閱讀。

3. 結果分析







10000101_2 即十進位的 133（44+87+2 結果）

A 等級(Cont.)

實作（三）

1. 實驗討論：

- 從記憶體位置 536870912(0x20000000)開始存放的陣列，依序放入 10、92、55、1、46。由位置 1 開始每次都跟後一位做比較，若是排在前面位置的數字較大則跟後者互換，接著 2,3 位置比，3,4 位置比，以此類推。直到未排序的最後一位-1，即可停止。這麼一來每次迴圈做完的最後一位必定為數列中最大值，經過多次迴圈便可完成排序，這種排序方法是冒泡排序(Bubble Sort)。
- 四次迴圈 Memory 內容變化如下：

	0x20000000	0x20000004	0x20000008	0x2000000c	0x20000010
一	10	92	55	1	46
二	10	55	1	46	92
三	10	1	46	55	92
四	1	10	46	55	92

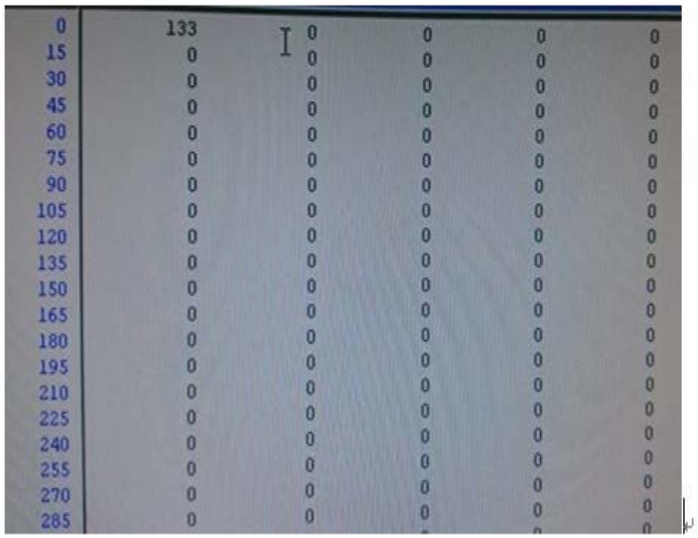
- 程式中用到的 Register 有：

暫存器	用途
$0	預設為 0
$2	用來暫存記憶體抓回來的數字並進行比較大小，也用作 branch 判斷依據
$3	用來暫存記憶體抓回來的數字並進行比較大小
$4	紀錄每次進入迴圈欲抓取的位置
$5	控制每次迴圈應比較的次數
$6	控制每次迴圈應比較的次數
$7	紀錄已排序完成的個數
$8	紀錄陣列初始位置
$9	紀錄陣列大小
$10	紀錄應執行迴圈次數

B 等級

(1)實作一

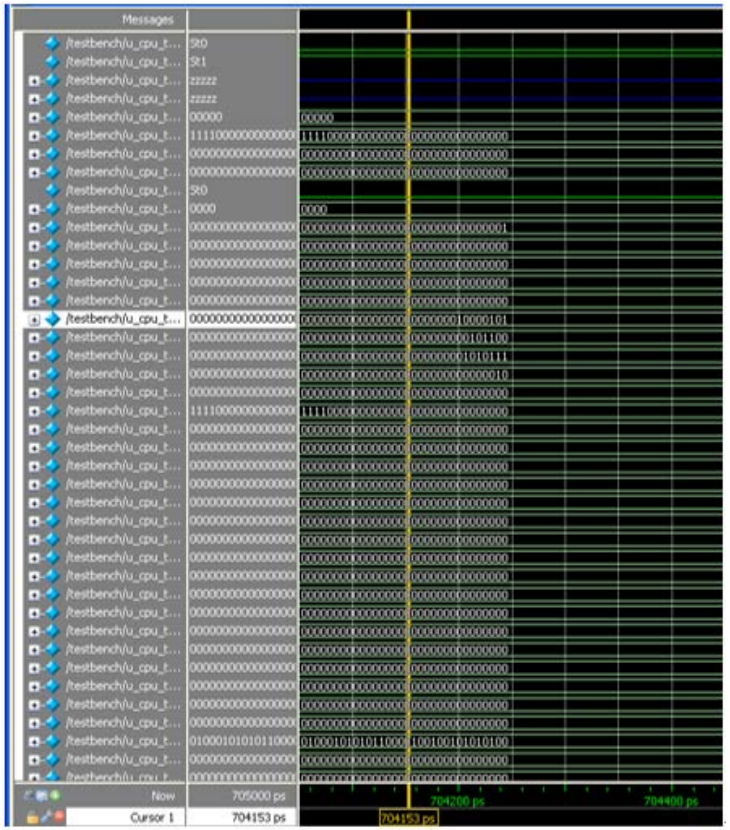
練習題一將數字 44, 87 和 2 加完之後存在*n 之中，可以從照片中很清楚地看到在記憶體 0x20000000 中存著加完的結果 133。



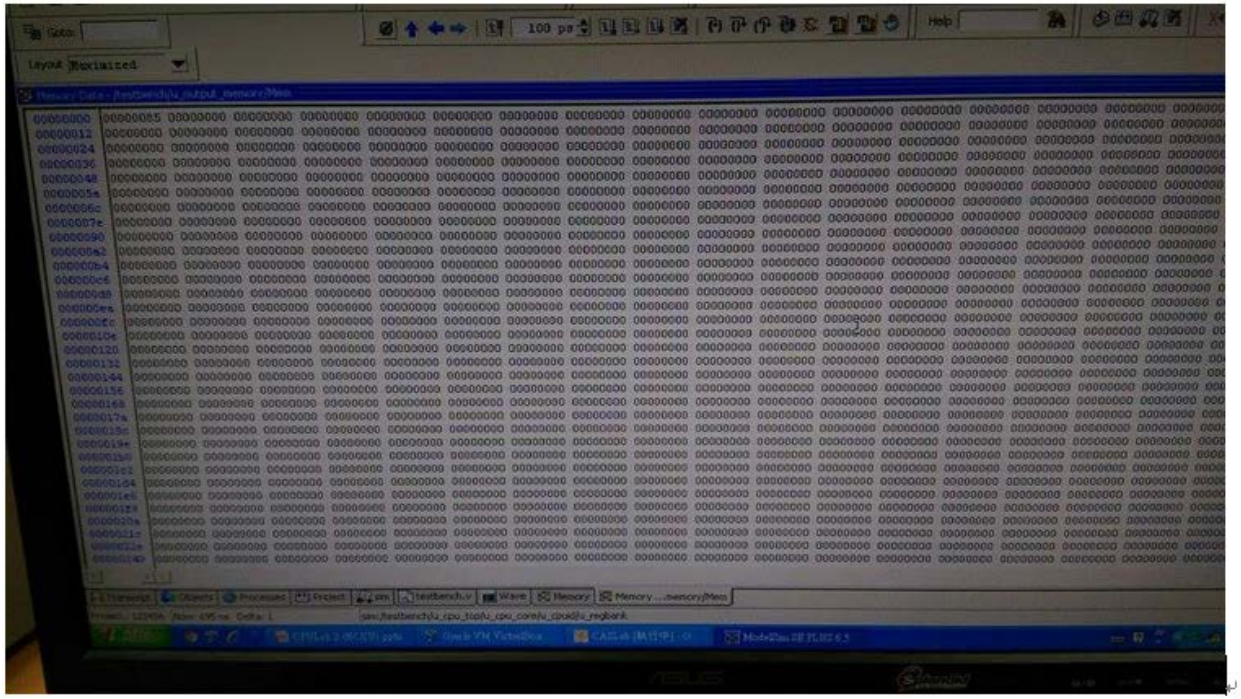
(1) 實作一

反白的是總合 133

往下依序是 a:44、b:87、c:2



C 等級



Memory Data - /testbench/u_output_memory/mem - default

00000000	0000003c	00000046	00000046	00000000	00000000	00000000	00000000	00000000	00000000	00000046	00000032	00000050	00000000	00000000	00000000
0000000e	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
0000001c	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
0000002a	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000038	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000
00000046	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000	00000000

Q&A