

組合語言與嵌入式系統 Final Project

108 學年度第一學期

老師：朱守禮 老師

組別：9

學生：10727138 資訊二甲 游子諭

學生：10727122 資訊二甲 廖奕銘

一、背景

透過這次的 Final Project 來更深入地了解 sp 與 fp 堆疊的關係與應用方法，以及得知在使用二維陣列時堆疊是以 row-major 的方式使用。同時也了解到以 C 呼喚 assembly 時如何傳送及回傳變數。

本組希望研究到的問題是記憶體空間拿來做使用時的變化情形，並以實作驗證瞭解上課時不甚清楚的問題。

二、方法

利用課本後面的說明來下載 QEMU 模擬器，並用裡面的 CodeBlocks 來寫，同時三項作業必須放在同一個 Project 才能讓 MAIN 成功呼叫到另外兩個函數。

設計重點說明：

Name：只要用 .asciz 將名字存在記憶體空間，之後再用 ldr 把位置給 r0 最後用 blprintf 印出。

ID：利用 .asciz “%d”來讀取學號，.word 來儲存學號，先用 ldr 將帶有 .asciz 的 label 的位置給 r0，接著用帶有 .word 的 label 的位置給 r1，最後用 bl scanf 讀取學號。

drawJuliaSet：這個函數主要是用來計算並決定 Frame 二維陣列裡每個元素的值，並以此來決定該元素投影至畫面(Frame Buffer)上的 Pixel 顏色。範例程式 drawJuliaSet.c 裡提供相關計算方式。需與所提供之範例程式功能相同。在與 main 函數整合後，可以正確繪製畫面。

MAIN：在 main 中呼叫上述三個函數，以 Name 及 ID 回傳 pointer 並印出所指向之記憶體位址的值。

三、結果

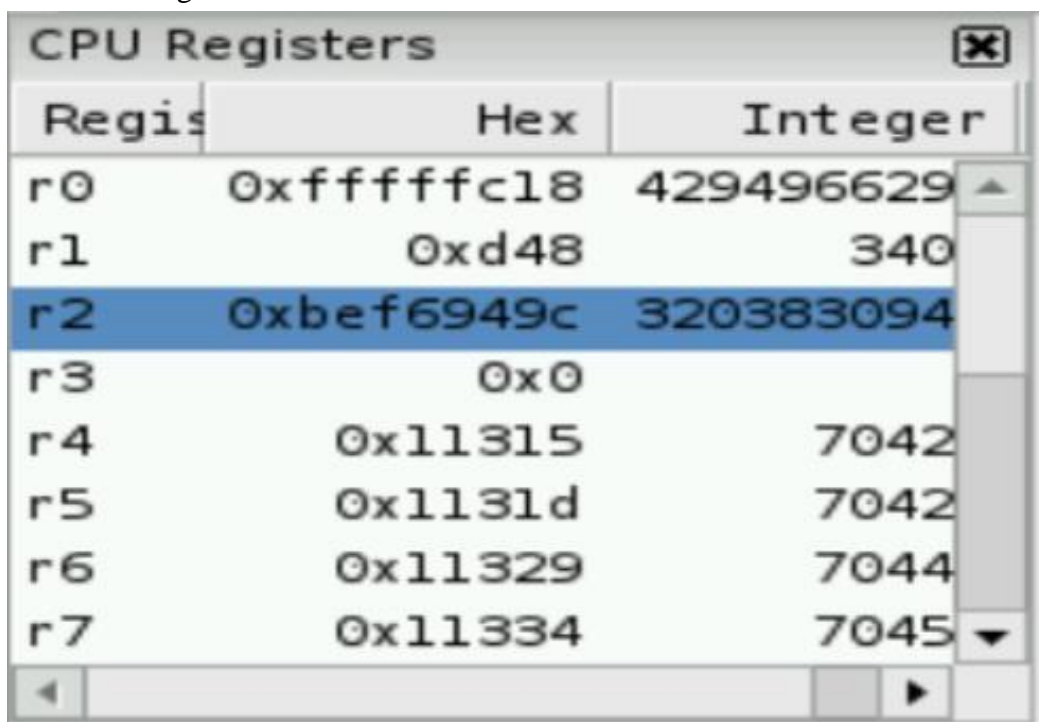
JuliaSet 中代表 frame 的堆疊區域

Assembly language



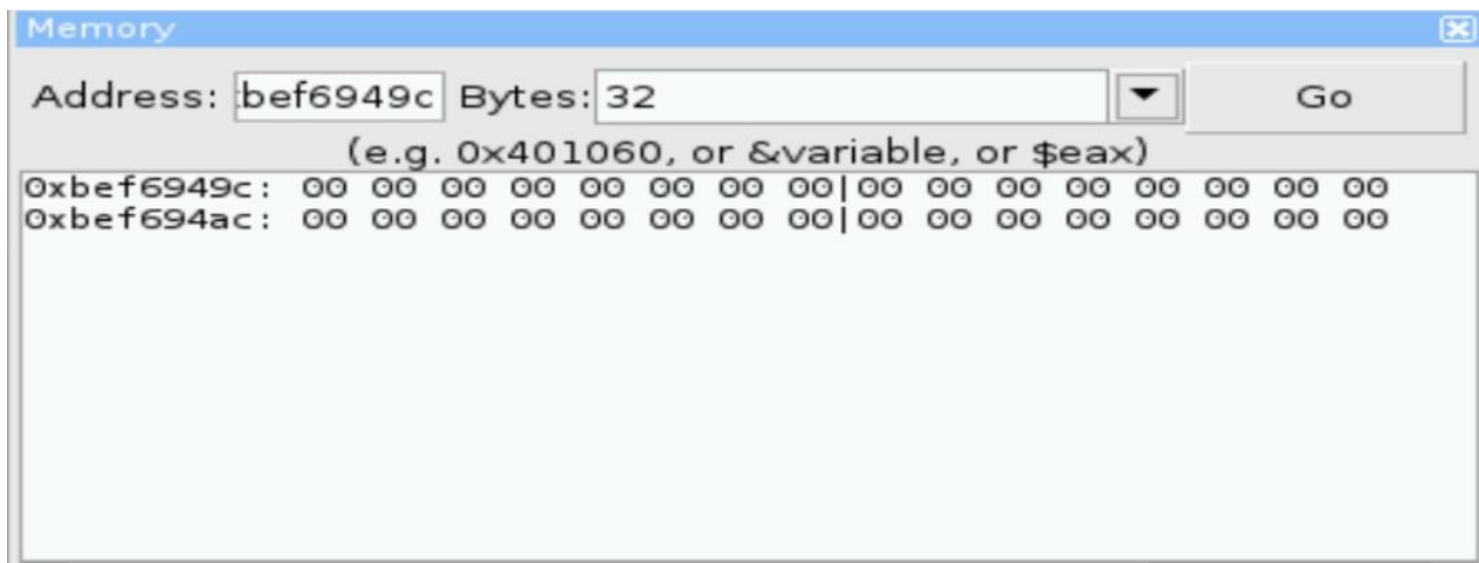
```
id.s  main.c  name.s  Untitled1.c  Untitled1.s [X]
136      mov r3, r3, asl #2
137      add r3, r3, r2      @Opperand2
138      mov r3, r3, asl #8
139      ldr r2, [fp, #4]
140      add r2, r2, r3
141      ldr r3, [fp, #-20]
142      mov r3, r3, asl #1      @Opperand2
143      add r3, r2, r3
144      ldrh r2, [fp, #-34]      @ movhi
145      strh r2, [r3, #0]      @ movhi
146      ldr r3, [fp, #-24]
147      add r3, r3, #1
148      str r3, [fp, #-24]
149      .L3:
150      ldr r2, [fp, #-24]
151      ldr r3, [fp, #-52]
```

CPU Registers 的位置



Regis	Hex	Integer
r0	0xfffffc18	429496629
r1	0xd48	340
r2	0xbef6949c	320383094
r3	0x0	
r4	0x11315	7042
r5	0x1131d	7042
r6	0x11329	7044
r7	0x11334	7045

Memory dump 的顯示



0xbef6949c 是 fp 跟 sp 所借的用來存 frame 之堆疊空間的頭，其堆疊的順序是依照 row major 的邏輯去執行。

程式說明：

NAME：

(1) 功能： 印出組別與組員名單。

(2) 程式設計：

(a) 規劃 4 個記憶體區塊，分別存放組別與組員的英文姓名。

(b)組別與組員名單已事先填入所屬之記憶體區塊。

(c) 撰寫程式，印出組別與組員姓名

NAME 的所在位置以及回傳位置

lr 是所在位置，pc 是回傳位置

CPU Registers		
Regis	Hex	Integer
r10	0x40026000	107389747
r11	0xbefff4c4	320444538
r12	0x0	
sp	0xbef69440	320383084
lr	0x4020b8b4	107588626
pc	0x8a84	3546
cpsr	0x60000010	161061275

```
id.s  main.c  name.s  Untitled1.c  Untitled1.s
45      char *Name[20] ;
46      int *Num = malloc(sizeof(int)) ;
47      int address ;
48
49      printf( "Function1: Name\n" );
50
51
52      //Dummy Function. Please refer to the specification of Project 1.
53      address = name();
54      *Name = address ;
55      I
56
57      printf( "Function2: ID\n" );
58
59      //Dummy Function. Please refer to the specification of Project 1.
60
```

ID :

(1) 功能： 輸入組員的學號，並印出組員學號與學號總和。

(2) 程式設計：

(a) 規劃 4 個記憶體位址，作為之後輸入組員學號與總和的緩衝區

(b) 請以輸入的方式，輸入三個組員的學號，並記錄於先前規劃的記憶體位址。

(c) 學號輸入完後，將 3 組輸入學號加總成一個數值，記錄於第 4 個記憶體區塊。

(d) 輸入完 3 組學號後，按下 p 鍵，即分行印出完整的組員學號與學號總和

ID 的所在位置以及回傳位置

lr 是所在位置，pc 是回傳位置

CPU Registers		
Regis	Hex	Integer
r10	0x40026000	107389747
r11	0xbefff4c4	320444538
r12	0x0	
sp	0xbef69440	320383084
lr	0x4020b8b4	107588626
pc	0x8aa8	3549
cpsr	0x60000010	161061275

```
id.s  main.c  name.s  Untitled1.c  Untitled1.s
54      *Name = address ;
55
56
57      printf( "Function2: ID\n" );
58
59      //Dummy Function. Please refer to the specification of Project 1.
60
61      |  address = id();
62      |  Num = address ;
63
64      printf( "\n\nMain Function:\n" );
65      printf( "*****Printf All*****\n" );
66
67      printf( "%s", *Name ) ;
68
69      *Name = *Name + 8 ;
```

Juliaset:

(1) 功能：這個函數主要是用來計算並決定 Frame 二維陣列裡每個元素的值，並以此來決定該元素投影至畫面(Frame Buffer)上的 Pixel 顏色。範例程式 drawJuliaSet.c 裡提供相關計算方式。

(2) 程式設計：(a) 請參考範例程式，以 ARM 組合語言重新設計 drawJuliaSet 這個函數，並儲存至 drawJuliaSet.s 檔案中。

Juliaset 的所在位置以及回傳位置

lr 是所在位置，pc 是回傳位置

CPU Registers		
Regis	Hex	Integer
r10	0x112f3	7038
r11	0xbeffff4c4	320444538
r12	0x11334	7045
sp	0xbef69440	320383084
lr	0x8c50	3592
pc	0x8c7c	3596
cpsr	0x20000010	53687092

```
id.s  main.c  name.s  Untitled1.c  Untitled1.s
107      else
108      {
109          for( cY=400 ; cY>=min_cY; cY = cY + cY_step ) {
110              // 5A08 7B37 7EEB E240 CX,cY 53C3 C978 4E08 7EB4 Julia set 755
111              drawJuliaSet( cX, cY, FRAME_WIDTH, FRAME_HEIGHT, frame );
112              // 300F 304E 4F4E 5EBE I/O 54C0 4F5C 547C 53EB Frame Buffer 75
113              // ( 5C07 75C8 97E2 5CC7 C9D9 5EB8 51C9 Frame Buffer)
114              write( fd, frame, sizeof(int16_t)*FRAME_HEIGHT*FRAME_WIDTH);
115              // 73FB 52D3 EA34 C848 C4C0 4F5C 4F4D 7FEE 51F3 C700 524D 7A
116              lseek( fd, 0, SEEK_SET );
117          }
118      }
119
120
121
122
```

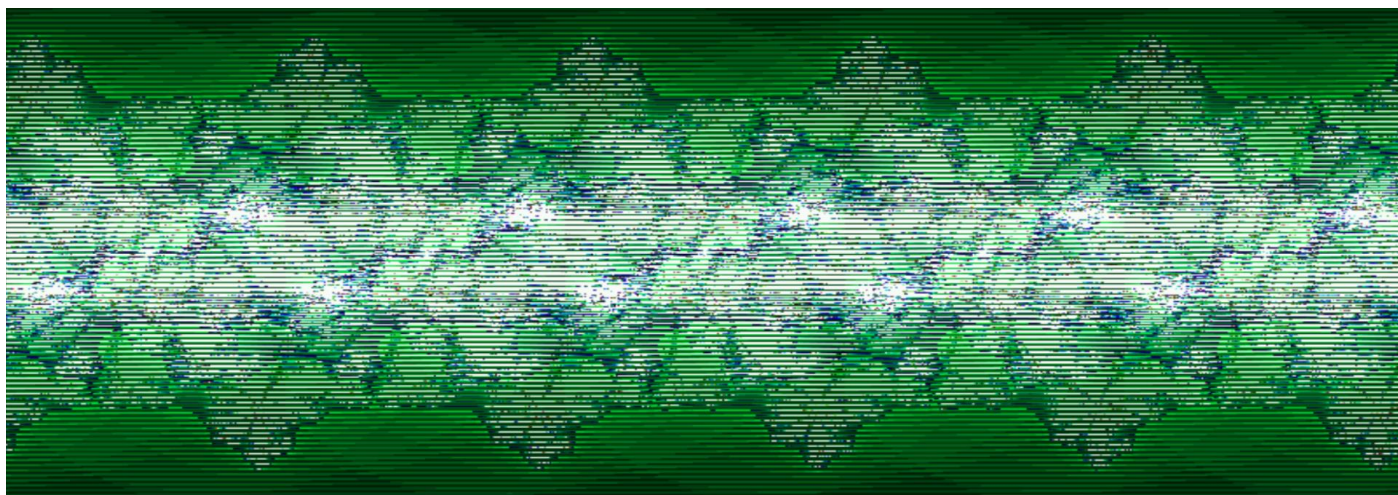

四、討論

1.本次作業遇到其中一個很大的問題是:究竟二維陣列 Memory 中的排列是如何?為了瞭解這個問題，我們將反組譯後的 drawJuliaSet 研究了一下，並上網查詢相關的資料。之後知道了 C 的二維陣列是 Row-Major，也就是排序是以列為優先。對應到 Assembly 的內容，發現程式會先將列(X 軸)填滿後，才開始填下一行(Y 軸)。

2.另一個很大的問題是要如何在 C 中傳送與呼叫 Assembly 的變數。藉由不斷測試後我們發現 Assembly 會將 r0 的值回傳，於是我們利用這點讓 Assembly 回傳一個位址，這個位址指向我們已經分配好的空間，裡面存著會在 main 中被用到的值，以此解決這個問題。

五、結論

做完這次 Final Project 後，我對於 sp、fp 與堆疊空間等 Memory 有了更加深入的了解。同時，透過這次的作業我也更加了解 Assembly 語言與 C 語言所組合的程式是如何互相合作使用，並藉由 drawJuliaSet 來實做了解。雖然一開始聽到老師講解時，完全沒有頭緒...但當我們實際開始作業後，才發現沒有想像中那麼難，只是需要花一點時間思考整理，再配合老師上課的內容，最後得以完成這份 Final Project。



六、未來展望

希望之後能應用所學無論是在使用 C、C++、C#或甚至 python、R 等語言時藉由本堂課所學到的知識，更了解程式背後實際的運作，進而幫助寫程式時 debug 或增進效率能有頭緒不混亂。

七、分工

廖奕銘:50%

游子諭:50%