實驗三 **ARM Assembly II**

1. 實驗目的

熟悉基本ARMv7組合語言語法使用。

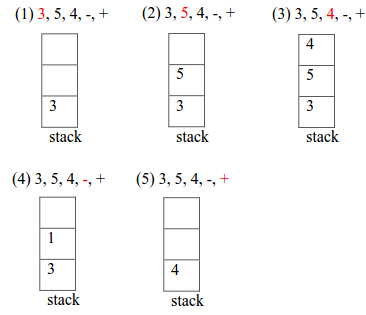
1. 實驗原理

請參考上課Assembly部分講義。

1. 實驗步驟
   1. **Postfix arithmetic**

操作stack來完成postfix的加減法運算

* + 1. *Example: 3, 5, 4, -, +*



* + 1. 實作要求

完成以下的程式碼，必須要利用PUSH,POP操作stack來完成postfix expression的運算，並將結果存進expr\_result這個變數裡。

|  |
| --- |
| .syntax unified  .cpu cortex-m4  .thumb  .data  user\_stack .zero 128  expr\_result .word 0    .text  .global main  postfix\_expr .asciz “-100 10 20 + - 10 +”  main:  LDR R0, =postfix\_expr    //TODO: Setup stack pointer to end of user\_stack and calculate the expression using PUSH, POP operators, and store the result into expr\_result  program\_end:  B program\_end    atoi:  //TODO: implement a “convert string to integer” function  BX LR |

* 1. 求最大公因數並計算最多用了多少**stack size**

在程式碼中宣告2個變數m與n ，並撰寫Stein版本的最大公因數，將結果存入變數result裡，請使用recursion的寫法，並使用stack傳遞function的parameters，禁止單純用register來傳。

計算在recursion過程中，記錄最多用了多少stack size，並將它存進max\_size這個變數中。

|  |
| --- |
| .data  result: .word 0  max\_size: .word 0  .text  m: .word 0x5E  n: .word 0x60  GCD:  //TODO: Implement your GCD function  BX LR |

4.實驗結果與分析

此次實驗花了我不少時間，又比lab2再更難了一些。其中，ldr指令使我卡了很久，還一度以為是板子壞掉。我試著寫ldr r2 , [r4]，r2、r4皆非與記憶體連結的reg，因此當我想把r4的值load進r2時，出現的卻是亂碼（或許是r4的位址）。此問題build時不會報錯，debug時也不會在這裡壞掉，而是停在一個與此無關的地方，然後按”step into”的按鈕就再也沒有反應了，讓我花了好一段時間才發現。