**實驗名稱 :**

ARM Assembly I

**實驗目的 :**

熟悉基本ARMv7組合語言語法使用。

* 如何利用條件跳躍指令完成程式迴圈的操作
* 算數與邏輯操作指令使用
* 暫存器(Register)使用與基本函式參數傳遞
* 記憶體與陣列存取
* Random Number Generator 使用 (加分)
* FPU instructions使用 (加分)

**實驗步驟 :**

3.1 Hamming distance: 計算兩個數長度為half-word的漢明距離，並將結果存放至result變數中。

3.2 Fibonacci serial: 宣告一數值N(1<=N<=100)，計算Fib(N)並將回傳值存放至R4暫存器。

3.3 Bubble sort: 利用組合語言完成長度為8byte的8bit泡沫排序法。

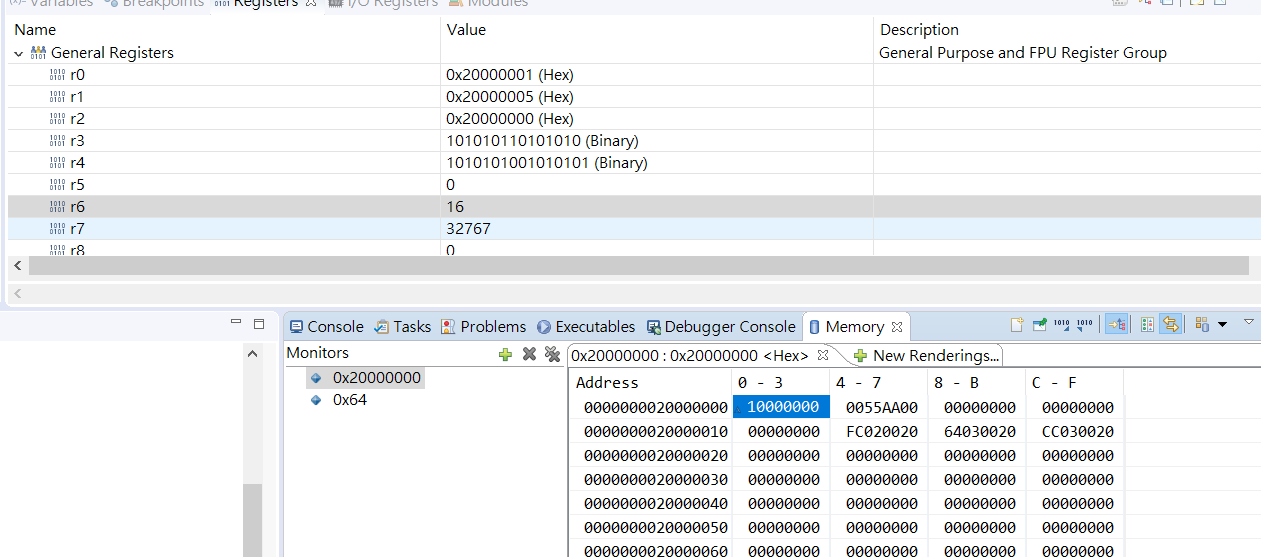
3.4.1 Enabling FPU and Floating Point Manipulation: 開啟FPU計算功能並進行計算。

3.4.2 Random Number Generator: 開啟RNG功能，產生一組(x,y)點在單位平面裡。﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽thodDManipulation d。

3.4.3 Estimation of Pi: 使用Monte Carlo Method 來估算Pi的值。

**實驗結果與分析 :**

3.1:

****

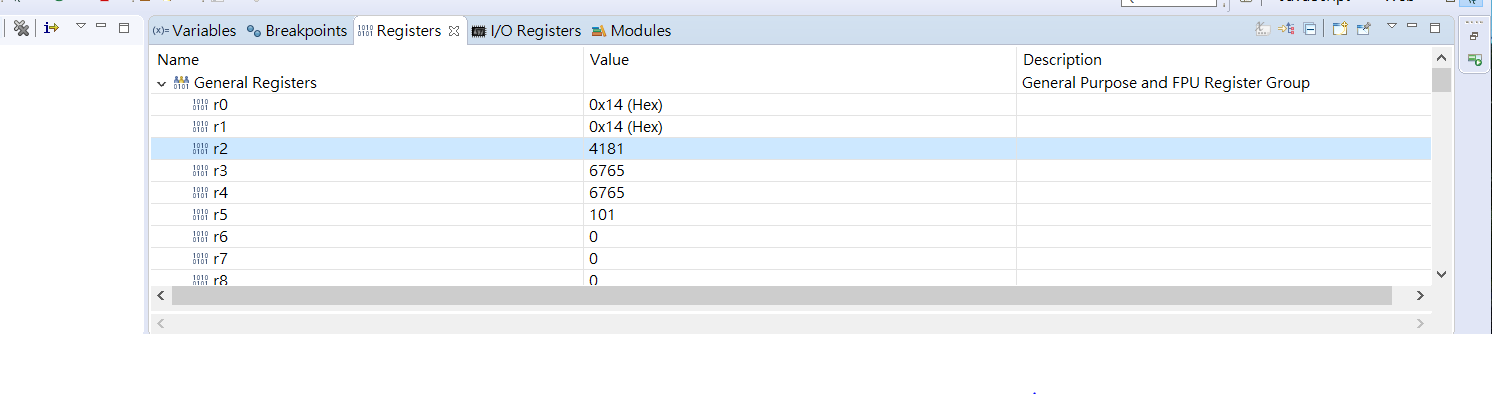
**Q: movs R0, #X will cause assemble error. Why? And How to fix?**

A:因為immediate最多只能load 8bit(0~255)+4bit right rotate的數字，

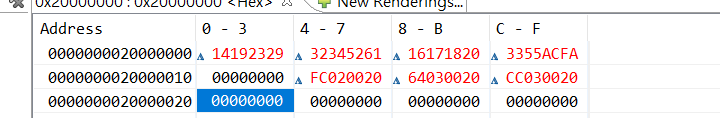
如果不是在這個範圍或是可以right rotate出的數字就沒辦法使用。

我們的解決方案是直接將x,y變成data範圍的宣告再從memory讀出來即可。

3.2:



3.3:



3.4.1:

**Q:如果enable\_fpu留空，程式會停在哪裡？ 為什麼？**

A:會在vldr時停住，因為vldr和s系列的register是屬於浮點數使用的，如果沒有將FPU打開的話，會無法使用浮點數的功能，也就無法使用這些東西。

**Q:為什麼需要將U32轉成F32格式再相加？如果想直接load一個值代表20到s2中不需轉換就能運算，應該將z修改成多少才能得到相同答案呢？**

A:因為floating point並非簡單的用二進位直接存入，而是分成多個部分（指數部分、正負號、有效數字），如果未經過vcvt，那二十就會變成不是二十，而是與20的unsigned表示法相同的浮點數。

至於想要直接load z的話，可以改成 .float 20.0。

3.4.3:

**心得討論與應用聯想 :**

這次的實驗讓我們充分了解如何在組合語言夜﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽﷽e error. Why? And How to fix?中實做迴圈以及熟練使用條件的指令，也能夠讓我們練習如何將一段簡單的c code(bubble sort)轉換成arm的組合語言。

除此之外，開啟FPU的相關使用也讓我們更加了解組合語言當中浮點數的運作模式，同時 RNG也讓我們能夠在組合語言當中使用亂數，最後並運用Monte Carlo的方法算出圓周率。