Lab0 Environment setup and Debugger Operation

實驗零 實驗環境建立與 Debugger 操作

1. Lab objectives 實驗目的

Familiar with the toolkits.

Familiar with the development environment.

熟悉實驗工具

熟悉開發環境

2. Steps 實驗步驟

2.1. Development board basic 開發板基礎知識

In the crash course, we learned the basic concepts of the development board. But there is still a lot of important information that we did not cover. Please find the following in the manuals or find them on google.

在速成課程中，我們學習了開發板的基本概念。但是仍然有很多重要的信息我們沒有涵蓋。請參考使用手冊或在google上查找以下內容。

Question 1-1: How many general purpose registers does the arm cortex m4 processor have?

Arm cortex m4 處理器有多少一般目的暫存器?

Question 1-2: What's the flash memory and SRAM size of NUCLEO-L476RG?

NUCLEO-L476RG 的快閃記憶體與靜態隨機存取記憶體大小為多少？

Question 1-3: What's the beginning address of SRAM? Find the memory map.

靜態隨機存取記憶體的起始位置為多少？找出記憶體映射表。

2.2. Project creation and program compilation 專案建立與程式編譯

Please refer to the sample code below, and create a file named "main.s" under the project "Lab0", build the project and observe how the program runs through the debugger tool.

請參考下面的範例程式，並在專案 "Lab0" 底下創建一個名為“ main.s”的檔案，建置該專案並透過 Debugger 工具觀察程式如何運行。

```
    .syntax unified
    .cpu cortex-m4
    .thumb

.text
      .equ X, 0x55
      .equ Y, 0x01234567

.global main
main:
   movs r0, #X
   ldr  r1, =Y
   adds r2, r0, r1

L: B L
```

Require 2-1: Observe the value of registers in the register monitor window.

在暫存器監測視窗中觀察暫存器的變化。

Require 2-2: Observe the compiled .elf file by external tools.

使用外部工具觀察編譯後的 .elf 檔。

Question 2-1: Is there any difference between the code disassembled by external tools and our source code.

透過反組譯工具所得到的程式碼與我們的原始碼內容上有和不同？

2.3. Variable declaration and Memory observation 變數宣告與記憶體觀察

Please follow the sample code below to modify "main.s" and observe the change of the memory value through the memory browser.

請按照以下面範例程式修改“ main.s”，並通過記憶體瀏覽器觀察記憶體內存的變化。

```
    .syntax unified
    .cpu cortex-m4
    .thumb

.data
   X: .word 100
   str: .asciz "Hello World!"
.text
   .global main
   .equ AA, 0x55

main:
   ldr  r1, =X
   ldr  r0, [r1]
   movs r2, #AA
   adds r2, r2, r0
   str   r2, [r1]

   ldr   r1, =str
   ldr   r2, [r1]

L: B L
```

Require 3-1: Show the memory content of variable "X" and "str" by memory browser.

00000064 ↘　↙ Hello World! 的 ascii code 編碼

透過記憶體瀏覽器顯示變數 X 和 str 的記憶體內容。

Question 3-1: When did the memory content of variable "X" and "str" be initialized? Is it initialized during execution?

變數 X 和 str 的記憶體內容是何時被初始化的？是在程式執行過程中被初始化嘛的嘛？

↳ 按 debug 之後立刻看就被初始化好了

Question 3-2: What effect will the execution result have, if the variable X is declared in the text section? (Note: You can use external tools to verify your guess. ex: objdump, readelf, nm ...etc.)

如果改將變量X宣告在 text section，對執行結果會產生什麼影響？(註：你可以使用外部工具驗證你的猜想。例如：objdump，readelf，nm等。

↓
X在記憶體的位置從原本的 0X20000000 → 0X80001C8
便得 str 的位置從 0X20000004 → 0X20000000

Question 3-3: What is the difference between the content of "r2" and the first 4 bytes of "str" in the memory after the program is executed?

程式執行完畢後"r2"的內容與 字串"str"在 memory 內的前4個 byte 內容有何差異？ little endian：相同（0x6c6c6548）
big endian：相反（0x48656c6c）

Question 3-4: Here we use the reserved word ".asciz" for string declaration. Is there any other way to declare the variable str, "Hello World!". If so, please explain one of them.

這裡我們使用保留字 .asciz 進行字串宣告。還有什麼其他方法可以聲明變量str, “ Hello World！”。如果有，請解釋其中之一。