Homework #2 (1)

- Write an ARM assembly program that does the following computation and puts the result at register r0. (不考慮overflow)

  – r0 = 2*r1 + 4*r2 + 8*r3

- The initial values of r1, r2, and r3 are assigned by yourself.

Template

```
/* ============================ */
/*         TEXT section         */
/* ============================ */
        .section .text
        .global main
        .type main,%function
main:
        mov   r1, #10          /* r1 = 10 */

        Your codes

        nop
        .end
```

- 一開始指定給r1, r2, r3的數值
- 助教批改作業時, 可能會測試不同的數值
- 因為編碼的緣故，不是每個數都能表示，請直接在GUI上修改register的值

- #num : 表示10進位數字
- #0xnum : 表示16進位數字
- #0bnum: 表示2進位數字
- #0num: 表示8進位數字

Template (1)

```
/* ============================ */
/*        TEXT section          */
/* ============================ */
        .section .text
        .global main
        .type main,%function
main:
        mov   r1,#10          /* r1 = 10 */
        mov   r2,#20          /* r2 = 20 */
        mov   r3,#12          /* r3 = 12 */

        Your codes

        nop
        .end
```

執行到nop時，r0的值為答案。

Pseudo Instruction: NOP

- This pseudo op will always evaluate to a legal ARM instruction that does nothing.
- Currently it will evaluate to **MOV r0, r0**.

Pseudo Instruction: LDR <register>, =<exp>

- **`LDR <register>,=label`**
  - If you plan to reference labels in other sections of code
- **`LDR <register>,=constant`**
  - If expression evaluates to a numeric constant then a MOV or MVN instruction will be used in place of the LDR instruction, if the constant can be generated by either of these instructions. Otherwise the constant will be placed into the nearest literal pool (if it not already there) and a PC relative LDR instruction will be generated.

Template (2)

```
/* ============================ */
/*        TEXT section          */
/* ============================ */
        .section .text
        .global main
        .type main,%function
main:
        ldr   r1,=#10          /* r1 = 10 */
        ldr   r2,=#20          /* r2 = 20 */
        ldr   r3,=#12          /* r3 = 12 */

        Your codes

        nop
        .end
```

執行到nop時，r0的值為答案。

Homework #2 (2)

- How to compile:

```
$ arm-none-eabi-gcc -g -O0 hw2.s -o \
hw2.exe
```

- How to execute
  - arm-none-eabi-insight

Homework #2 (3)

- Program should be assembled and linked by GNU cross toolchain.
- Program can be executed under **GDB ARM simulator**
- 程式中應有適當的說明（註解）
- You should turn in to **ECOURSE2**
  - "**README.txt**" file: 文字檔，描述你程式的內容、如何編譯程式、程式的執行環境、如何執行你的程式
  - "**hw2.s**": Your ARM assembly program
  - "**hw2.exe**": 編譯好的執行檔
  - 請將欲繳交的檔案壓縮成 <**hw2_學號.tar.bz2**>，上傳壓縮檔
- **Deadline: October 25 (Monday), 2021, 24:00**

(此次作業, 不可補交)