第06次組語實習課

學生：林培瑋

2023 Advanced Mixed-Operation System (AMOS) Lab.

Tamkang University
Department of Electrical and Computer Engineering
No.151, Yingzhuan Rd., Tamsui Dist., New Taipei City 25137, Taiwan (R.O.C.)

p.90 EXAMPLE 5.4

TABLE 4.2
Frequently Used Directives

Keil Directive	CCS Directive	Uses
AREA	.sect	Defines a block of code or data
RN	.asg	Can be used to associate a register with a name
EQU	.equ	Equates a symbol to a numeric constant
ENTRY		Declares an entry point to your program
DCB, DCW, DCD	.byte, .half, .word	Allocates memory and specifies initial runtime contents
ALIGN	.align	Aligns data or code to a particular memory boundary
SPACE	.space	Reserves a zeroed block of memory of a particular size
LTORG		Assigns the starting point of a literal pool
END	.end	Designates the end of a source file

EXAMPLE 5.4

Consider a simple ARM7TDMI program that moves (copies) a string of characters from one memory location to another.

(C語言寫法)
#define SRAM_BASE 0x40000000

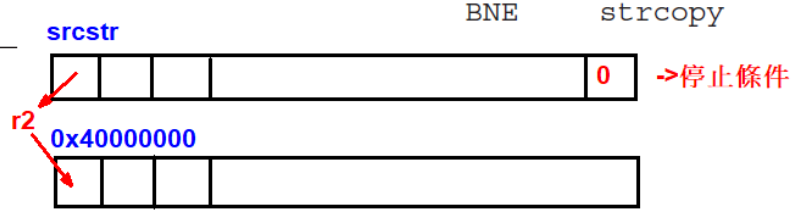
Equal

0x40000000 ->從這個位置以後能讓使用者存取。

(psuedo instruction)
LDR r1, =srcstr

Address into Register

```
SRAM_BASE   EQU     [redacted]      ; start of SRAM for STR910FM32
            AREA    StrCopy, CODE
            ENTRY                   ; mark the first instruction
Main        ADR     r1, srcstr      ; pointer to the first string
            LDR     r0, =SRAM_BASE  ; pointer to the second string
strcopy
            LDRB    r2, [r1], #1    ; load byte, update address
            STRB    r2, [r0], #1    ; store byte, update address
            CMP     r2, #0          ; check for zero terminator
            BNE     strcopy         ; keep going if not
```




```
stop        B       stop            ; terminate the program
srcstr      DCB     "This is my (source) string", 0
            END
```

p.90 EXAMPLE 5.4-實作

Registers

Register	Value
Current	
R0	0x4000001A
R1	0x00000037
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x00000018
CPSR	0x600000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x00000018
Mode	Supervisor
States	284
Sec	0.00002367

Disassembly

```
    10:                 STRBNE  r2, [r0], #1
0x00000010  14C02001  STRNEB    R2,[R0],#0x0001
    11:                 BNE     strcopy
0x00000014  1AFFFFFB  BNE       0x00000008
    12: stop    B   stop
0x00000018  EAFFFFFE  B         0x00000018
```

ex5.4.s

```
1  SRAM_BASE        EQU     0x40000000
2          AREA     StrCopy, CODE
3          ENTRY
4  Main
5          ADR      r1, srcstr
6          LDR      r0, =SRAM_BASE
7  strcopy
8          LDRB     r2, [r1], #1
9          CMP      r2, #0
10         STRBNE   r2, [r0], #1
11         BNE      strcopy
12 stop    B        stop
13 srcstr  DCB      "This is my (source) string", 0
14         END
```

Memory 1

Address: 0x40000000

```
0x40000000: 54 68 69 73 20 69 73 20 6D 79 20 28 73 6F 75 72 63 65 29 20 73 74 72 69 6E 67 00 00 00 00 00 00 00 00
0x40000022: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000044: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000066: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Call Stack + Locals | Memory 1

p.90 EXAMPLE 5.4-補充題型1

- Reverse string

Registers

Register	Value
Current	
R0	0x3FFFFFFF
R1	0x0000003B
R2	0x00000000
R3	0x00000000
R4	0x00000000
R5	0x00000000
R6	0x00000000
R7	0x00000000
R8	0x00000000
R9	0x00000000
R10	0x00000000
R11	0x00000000
R12	0x00000000
R13 (SP)	0x00000000
R14 (LR)	0x00000000
R15 (PC)	0x0000001C
CPSR	0x600000D3
SPSR	0x00000000
User/System	
Fast Interrupt	
Interrupt	
Supervisor	
Abort	
Undefined	
Internal	
PC $	0x0000001C
Mode	Supervisor
States	279
Sec	0.00002325

Disassembly

```
    10:                 CMP     r2, #0
0x00000010  E3520000  CMP     R2,#0x00000000
    11:                 STRBNE  r2, [r0], #-1
0x00000014  14402001  STRNEB  R2,[R0],#-0x0001
    12:                 BNE     strcopy
0x00000018  1AFFFFFB  BNE     0x0000000C
```

ex5.4r.s

```
SRAM_BASE       EQU     0x40000000
        AREA    StrCopy, CODE
        ENTRY
Main
        ADR     r1, srcstr
        LDR     r0, =SRAM_BASE
        ADD     r0, #25
strcopy
        LDRB    r2, [r1], #1
        CMP     r2, #0
        STRBNE  r2, [r0], #-1
        BNE     strcopy
stop    B       stop
srcstr  DCB     "This is my (source) string", 0
        END
```

→字串總長度為26 bytes
→先讓初始位置r0變為r0+25，
從字串總長度最後一個位置開始往前存。
(存完1個byte→r0 = r0 -1)

Memory 1

Address: 0x40000000

```
0x40000000: 67 6E 69 72 74 73 20 29 65 63 72 75 6F 73 28 20 79 6D 20 73 69 20 73 69 68 54 00 00 00 00 00
0x4000001F: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x4000003E: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Memory 1

Address: 0x40000000

```
0x40000000: 54 68 69 73 20 69 73 20 6D 79 20 28 73 6F 75 72 63 65 29 20 73 74 72 69 6E 67 00 00 00 00 00 00 00
0x40000022: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000044: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0x40000066: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

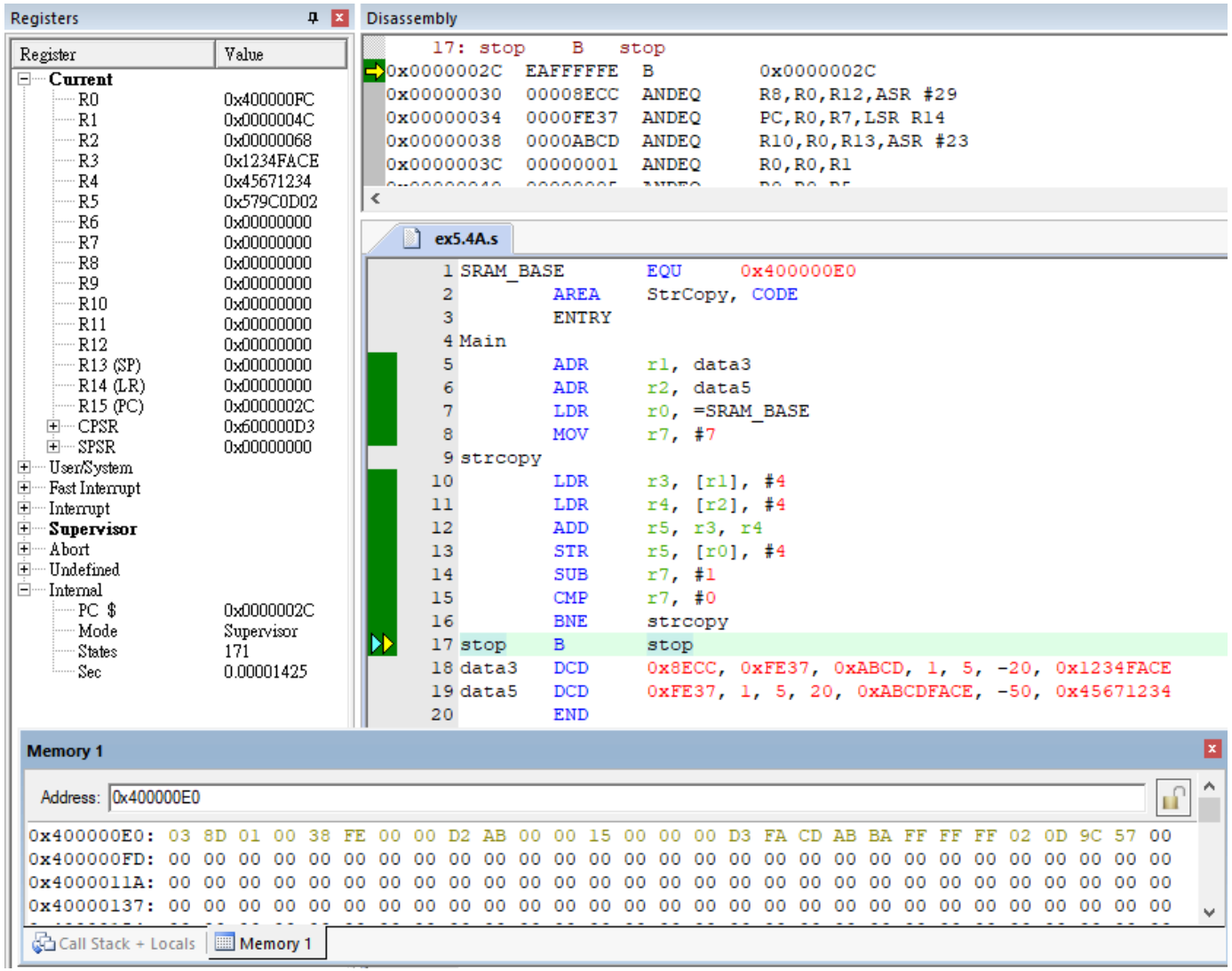
Call Stack + Locals | Memory 1

→原本的

p.90 EXAMPLE 5.4-補充題型2

- Word array(DCD) #4 #-4
- Halfword array(DCW) #2 #-2
- Add the corresponding words between data3 and data5 and put the sums one by one in memory started from address 0x400000E0.

```
data3       DCD 0x8ECC, 0xFE37, 0xABCD, 1, 5, -20, 0x1234FACE
data4       DCB 0xCF, 23, 39, 0x54, 250, 0xFF, 0xAD, -20, -30
data5       DCD 0xFE37, 1, 5, 20, 0xABCDFACE, -50, 0x45671234
```



p.124 TST指令

Q：read bits(讀取某個bits)

A1：AND(具破壞性，使暫存器數值發生變化)

A2：TST (undestructive，暫存器數值不發生改變)

補充： CMP r0, r1 CPU實際會做r0-r1

= 0 -> EQ
!= 0 -> NE

Ex：

```
        TST   r0, #0b0100   ;read bit2 of r0
        BNE   X1

        ; bit2 = 0

        B     NEXT
X1

        ; bit2 = 1

NEXT
```

bit2 = 0 -> EQ
bit2 = 1 -> NE

用這個隔開

Read multiple bits

EX：

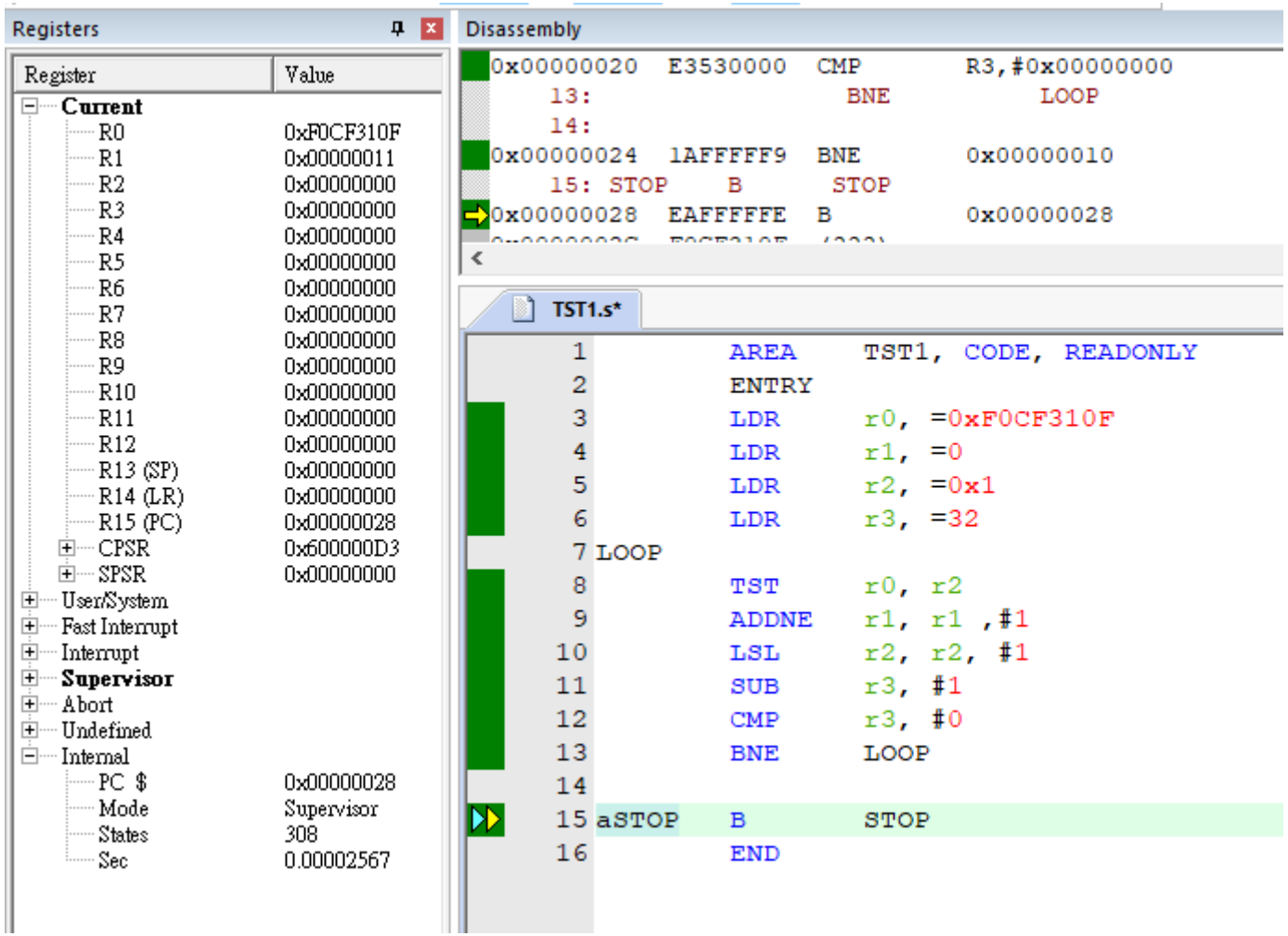
```
TST   r0, #0b1111
TST   r1, #0x0000000F
```

All bits read = 0 -> EQ

otherwise -> NE

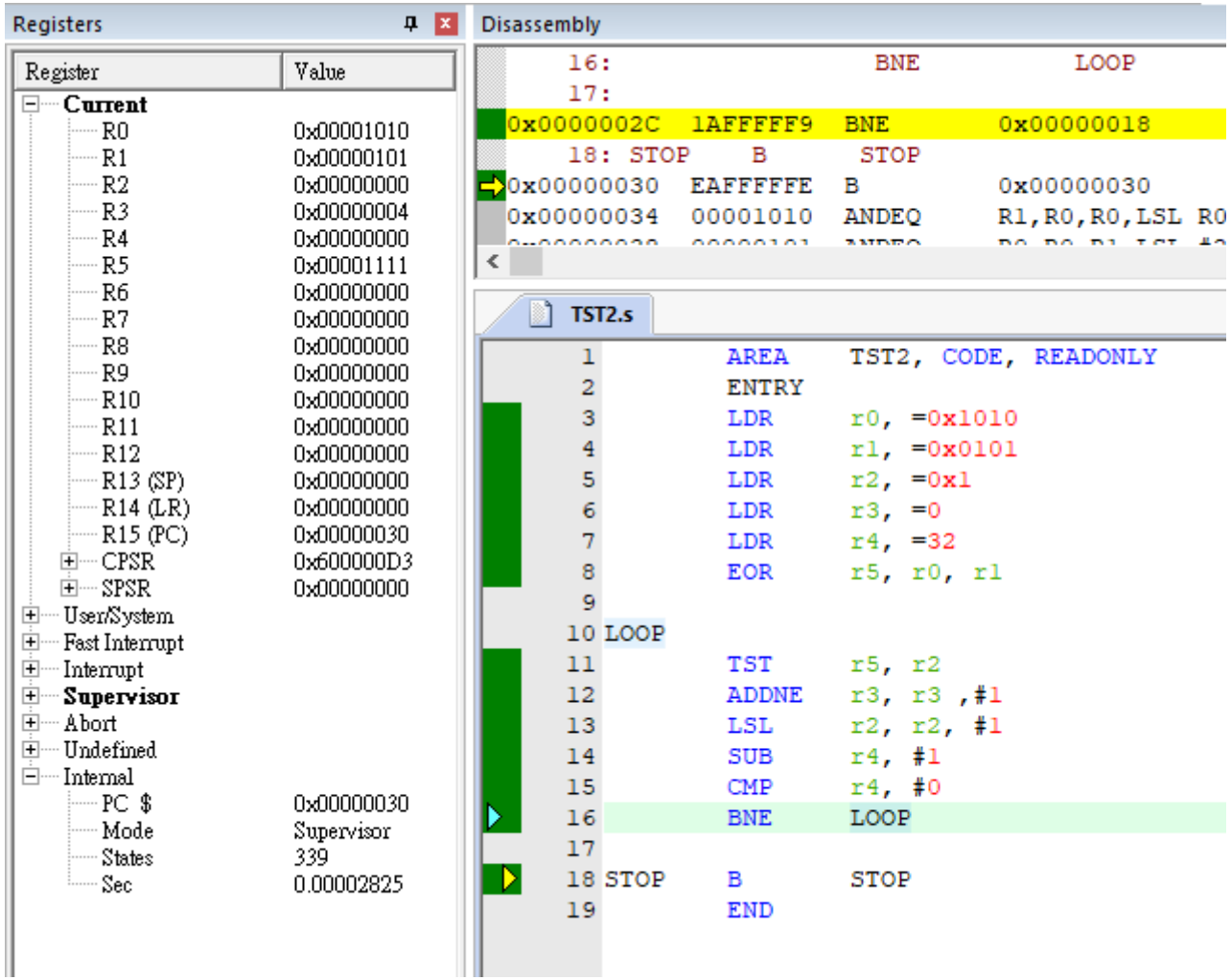
p.124 TST指令-補充題型1

- Put the number of 1's of r0 into r1.(r0 = 0b11110000110011110011000100001111)　ANS：r1 = 17



p.124 TST指令-補充題型2

- Put the number of different bits between r0 and r1 into r3.(r0 = 0x1010, r1 = 0101)　　ANS：4



作業2 注意事項(1/4)

第一部分

- 將前面兩大題相關程式如上課時所示範的一樣，各打入一個program，並單步執行（F11），且截圖展示出題目所要求的結果，手算驗證部分用註解呈現在program中

作業2 注意事項(2/4)

第二部分

- 如上課時所示範的一樣，將前一頁之宣稱寫入程式
- 在記憶體視窗中紅色標出每一個值儲存之位置並註明data1~data10每一變數的位址

作業2 注意事項(3/4)

→第三部分1(1)、2.可以寫在同一個程式碼，每執行完一行程式碼截一張圖。

第三部分

1. (1) Assume r3 = 0x40000000 and r4= 0x60. What would r3 contain after executing each of the following instructions?
   - (a) STR r6, [r3, #8]
   - (b) STRB r7, [r3], #12
   - (c) LDRH r5, [r3], #12
   - (d) LDR r12, [r3, #4]!
   - (e) LDR r6, [r3, r4, ROR #28]!
   - (f) LDR r0, [r3, r4, LSL #2]

   (2) What would register r4 contain after executing the following instructions? Register r6 holds the value 0xDEADBEEF and register r3 holds 0x40000000.

   STR r6, [r3]
   LDRB r4, [r3]

第三部分

2. If r2 = 0x12345678, r3 = 0x87654321, and r4 = 0x00000012 before execution, give the contents of r2 after each of the following instructions is executed.
   - (1) BIC r2, r2, #0xFF000000
   - (2) LSL r2, r3, #4
   - (3) LSL r2, r2, r4
   - (4) ROR r2, r2, #12 ;
   - (5) AND r2, r2, r3
   - (6) ORR r2, r2, r4
   - (7) EOR r2, r2, r4
   - (8) BIC r2, r2, r4
   - (9) EOR r2, r2, r3, ROR #7

作業2 注意事項(4/4)

→第三部分3.的每一小題獨立寫，避免因為前幾題結果錯，而導致後面也全錯。

第三部分

- 3. Assume r6 = 0xABCD8765. Write a sequence of instructions to
- (1) calculate the 2's complement of r6 and put the result in r7.
- (2) set bits 1, 5, and 13 in register r6 and leave the remaining bits unchanged.
- (3) clear bits 0, 4, and 12 in register r6 and leave the remaining bits unchanged.
- (4) change bits 4, 8, and 11 of r6.
- (5) insert the value 0x5555 into the lower half of register r0 so that the final value is 0xBEEF5555, assuming register r0 contains the value 0xBEEFABCD.

Q&A

Thanks for your attention !!