

計算機組織

基礎組合語言
(MIPS)



指導教授:朱宗賢

教學助理:王士嘉、林琨翔、高偉倫

目錄

Lab 介紹	2
(一)Lab 規則	2
(二)Lab 目的	2
(三)Lab 內容	2
環境介紹及安裝	3
(一)環境所需檔案	3
(二)Java	3
(三)下載 Java 並安裝	4
(四)設定環境變數	4
MARS	6
(一)MARS Simulator 基本介紹	6
(二)MIPS 程式基本架構	8
(三)MIPS 程式範例	8
Example1-Hello world!!	8
Example2-Load/Store	9
Example3-Arithmetic	10
Example4-Input & Output	11
Example5- Jump	12
附錄	13
(一)MIPS 暫存器	13
(二)MIPS System Service Table	13
(三)MIPS 儲存型態	14
(四)MIPS 常用指令介紹- 載入和儲存	14
(五)MIPS 常用指令介紹- 載入和儲存範例	15
(六)MIPS 常用指令介紹- 比較	15
(七)MIPS 常用指令介紹- 無條件跳躍	16
(八)MIPS 常用指令介紹- 條件跳躍	16
(九)MIPS 常用指令介紹- 邏輯運算	17
(十)MIPS 常用指令介紹- 算術運算	17

Lab 介紹

(一) Lab 規則

- (1) 1-3 人一組(專題)(作業若沒特別說明，則該次作業為單人作業)
- (2) 若作業完成，則在 eosplat 平台積分上會進行加分，但不列入學期成績。
- (3) 專題部分則會列入學期成績，詳細部分將在公布專題題目時說明。
- (4) 每次作業請盡力完成，作業皆與專題相關。

(二) Lab 目的

- (1) 透過實際的 MIPS 程式撰寫，使學生能更了解課本所教的組合語言相關內容。
- (2) 配合業界所需人才之組合語言撰寫能力，設計 Lab 內容與回家作業，並提升學生的知識、競爭力及解決問題的能力。
- (3) 藉由作業，使學生可以透過互相討論以解決問題，並整理成報告，讓學生能擁有完整且扎實的學習。

(三) Lab 內容

- (1) 搭配課本上的 MIPS 介紹、指令作為 LAB 內容，讓學生可以透過實際撰寫程式碼，更深入地了解 MIPS 組合語言架構和其語法。
- (2) 透過實際的操作，使學生更能了解組合語言的變化與記憶體運作方式。
- (3) 藉由作業激發學生的創意，加深學生對組合語言的熟悉，並提升學生討論及解決問題的能力。
- (4) 使用 MARS 軟體來模擬組合語言的編譯、執行過程、結果及其應用。

環境介紹及安裝

(一) 環境所需檔案

(1) Java 下載網址

https://www.java.com/zh_TW/

(2) Java SE Development Kit 5 下載網址

32 位元下載網址：

http://ituploads.com/software-downloads/sun/jdk-1_5_0_22-windows-i586-p.exe

64 位元下載網址：

http://ituploads.com/software-downloads/sun/jdk-1_5_0_22-windows-amd64.exe

(3) MARS Simulator V4.5 下載網址：

<http://courses.missouristate.edu/KenVollmar/MARS/download.htm>

(二) Java

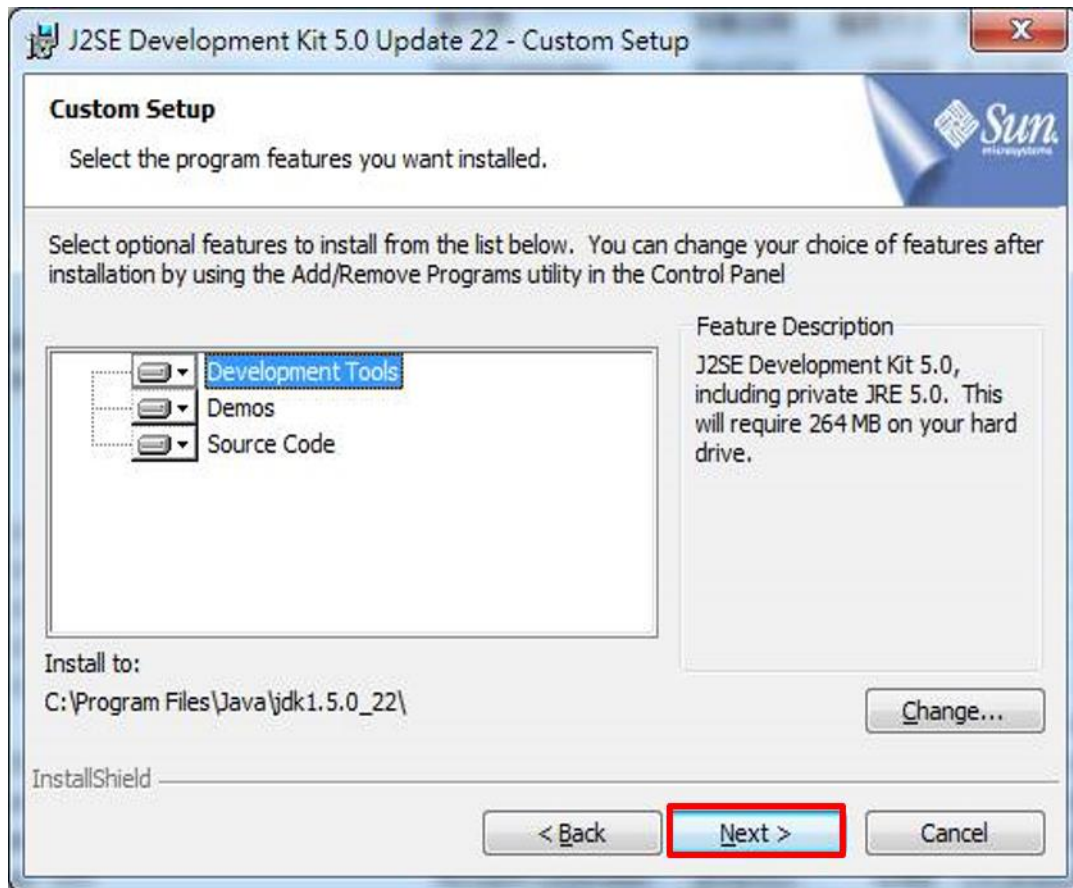
確認電腦是否已有 Java



P.S.若電腦裡已經有 Java 了，即可不用安裝。

若還是無法執行 MIPS，請嘗試移除現有 Java 版本，
並下載/安裝 Java SE Development Kit 5

(三) 下載 Java 並安裝

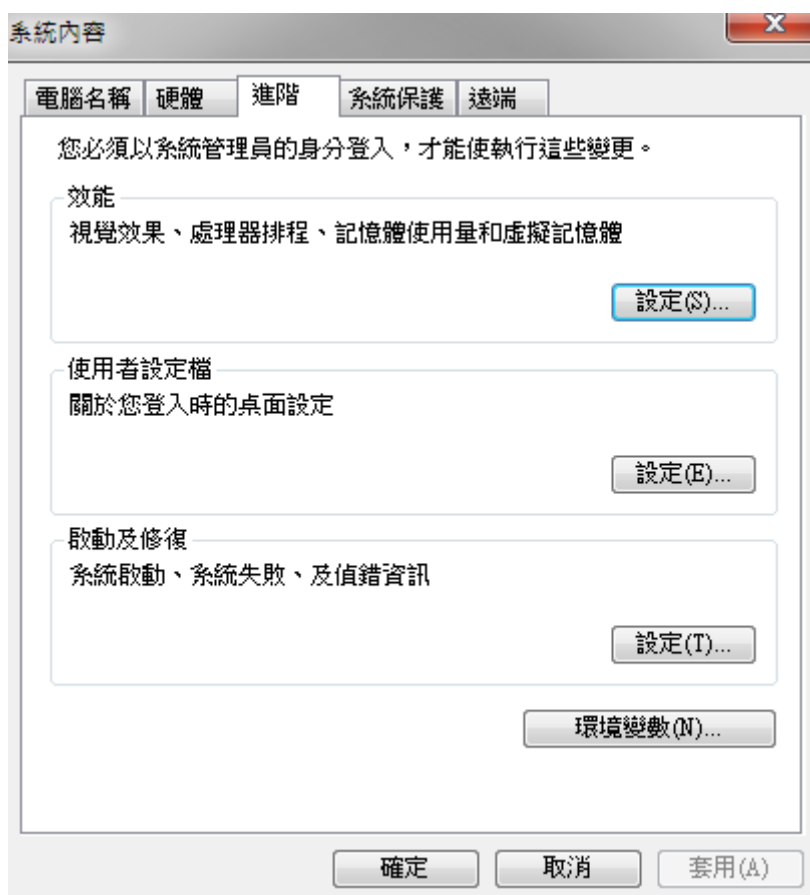


(四) 設定環境變數

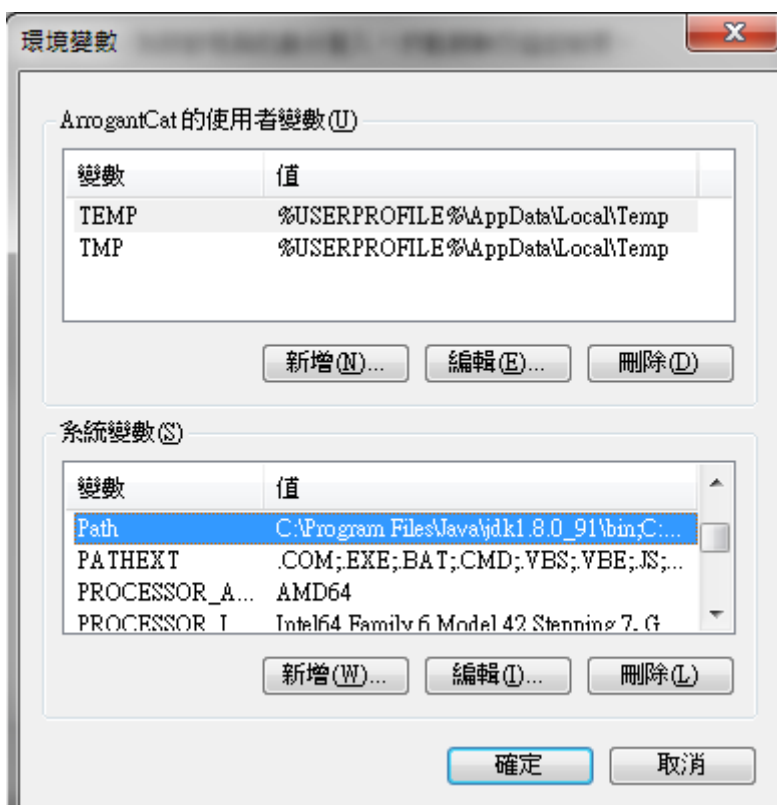
- (1) 環境變數是指：OS 中，用來指定 OS 運行時所需的某些參數。
- (2) Java 安裝完畢之後，需要設定 Path 環境變數。
- (3) Path 環境變數是指：OS 外部「命令」搜尋路徑，當使用者輸入指令時，OS 會至 Path 中搜尋調列其中的路徑，並在資料夾尋找符合其命令需求的檔案，最終以啟動並執行之。

簡而言之，我們需要再 PATH 設定指令的路徑資訊，OS 才可以憑藉找到指令。

STEP1. 電腦右鍵->內容->進階系統設定->進階->環境變數



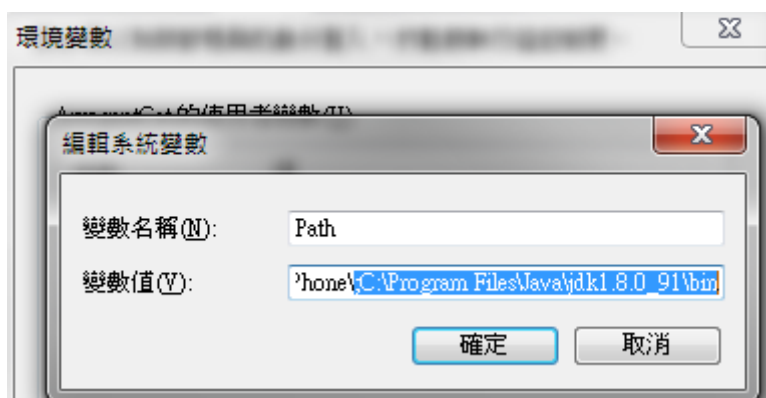
STEP2. 於系統變數(S)中，找到變數「Path」，點選 Path 後點選編輯



STEP3. 搜尋 JDK 的安裝路徑，開啟 bin 資料夾，並將其路徑複製下來。
(預設為 C:\Program Files\Java)

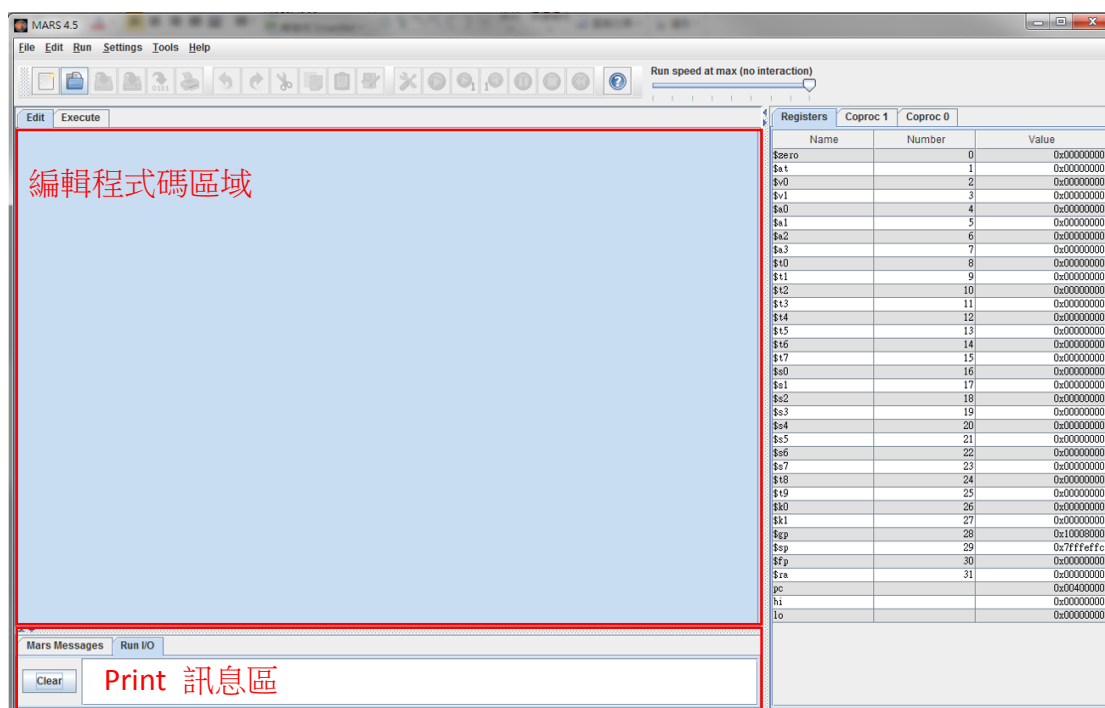


STEP4. 移到「變數值」最後面，加上「分號」，將路徑貼上後，按下確定。(分號用於區別各路徑)



MARS

(一) MARS Simulator 基本介紹



調整執行速度
(最慢速度為一秒一個指令)

- 1.編譯
- 2.執行
- 3.按一下，執行一個指令
- 4.按一下，倒退一個指令
- 5.暫停
- 6.停止
- 7.Reset MIPS 的記憶體和暫存器

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x00000000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000000
\$t1	9	0x00000000
\$t2	10	0x00000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x00000000
\$sp	29	0x7fffffc0
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400000
hi		0x00000000
lo		0x00000000

執行的程式碼及其行數

Text Segment	Address	Code	Basic	Source
6:	0x00400000	lui \$t0, 0x00000001		
7:	0x00400004	lw \$t1, 0x00000001(\$t0)		
8:	0x00400008	lw \$t2, 0x00000004(\$t1)		
9:	0x0040000c	lw \$t3, 0x00000008(\$t2)		
10:	0x00400010	lw \$t4, 0x0000000c(\$t3)		
11:	0x00400014	sw \$t4, 0x00000000(\$t0)		

暫存器及其值

Registers	Coproc 1	Coproc 0
Name	Number	Value
\$zero	0	0x00000000
\$at	1	0x10010000
\$v0	2	0x00000000
\$v1	3	0x00000000
\$a0	4	0x00000000
\$a1	5	0x00000000
\$a2	6	0x00000000
\$a3	7	0x00000000
\$t0	8	0x00000123
\$t1	9	0x00000045
\$t2	10	0x10000000
\$t3	11	0x00000000
\$t4	12	0x00000000
\$t5	13	0x00000000
\$t6	14	0x00000000
\$t7	15	0x00000000
\$s0	16	0x00000000
\$s1	17	0x00000000
\$s2	18	0x00000000
\$s3	19	0x00000000
\$s4	20	0x00000000
\$s5	21	0x00000000
\$s6	22	0x00000000
\$s7	23	0x00000000
\$t8	24	0x00000000
\$t9	25	0x00000000
\$k0	26	0x00000000
\$k1	27	0x00000000
\$gp	28	0x10000000
\$sp	29	0x7fffffc0
\$fp	30	0x00000000
\$ra	31	0x00000000
pc		0x00400020
hi		0x00000000
lo		0x00000000

(二) MIPS 程式基本架構

```
# 程式功能註解說明

.data #宣告變數

#變數名稱:    儲存型態    變數值
var1:         .word       3           #宣告整數
array1:       .byte      'a', 'b'     #宣告陣列
array2:       .space     40           #宣告連續 40byte 的空間
hello:        .asciiz    "Hello word!!\n" #宣告字串

.text        #程式指令區塊

main:
    #程式內容
    #程式內容

    li        $v0, 10    #end and exit
    syscall
```

(三) MIPS 程式範例

Example1-Hello world!!

```
# Example1 - Hello world!!
.data    #宣告變數

#變數名稱:    儲存型態    變數值
out_string:  .asciiz    "Hello world!!\n" #宣告字串變數 out_string

.text    #程式指令區塊
main:
    li      $v0, 4        #列印字串
    la      $a0, out_string #將 out_string 記憶體位置載入$a0 暫存器
    syscall
    li      $v0, 10       #end and exit
    syscall
```

Example2-Load/Store

Example 2 - Load/Store

.data #宣告變數

#變數名稱:	儲存型態	變數值	
array1:	.space	12	#宣告 12byte 空間來儲存 3 個整數
newline:	.ascii	"\n"	#換行

.text #程式指令區塊

main:

```

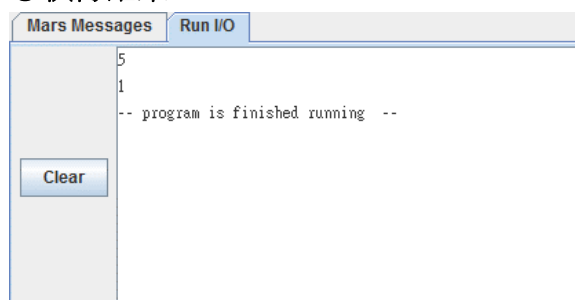
la    $t0, array1    #載入 array1 記憶體位址至$t0#暫存器中
li    $t1, '5'        #將字元 5 儲載入$t1 暫存器
sw    $t1, 0($t0)     #將$t1 的值存至記憶體 #array1[0]中
li    $v0, 4          #輸出字串
la    $a0, 0($t0)     #載入 array[0]記憶體位址至$a0 暫存器
syscall

li    $v0, 4          #輸出字串
la    $a0, newline    #換行
syscall

li    $t1, 1          #將整數 1 載入在$t1 暫存器
sw    $t1, 4($t0)     #將$t1 的值存至記憶體 array[1]中
lw    $t3, 4($t0)     #將 array[1]的值載入$t3 暫存器中
li    $v0, 1          #輸出整數
move  $a0, $t3        #移動$t3 暫存器內容至$a0 暫存器中
syscall

li    $v0, 10         #end and exit
syscall
  
```

◎執行結果:



Example3-Arithmetic

Example 3 – Arithmetic

.text #程式指令區塊

main:

```

li      $t1, 3          #將整數 3 載入$t1 暫存器中
addi    $t2, $zero, 5   #將整數 5 載入$t2 暫存器中

add     $t3, $t2, $t1    # $t3 = t2 + t1$ 
sub     $t3, $t3, $t1    # $t3 = t3 - t1$ 

mult    $t2, $t3         # $Hi, Lo = t2 * t3$ 
mflo    $t4             # $t4 = Lo$ 

div     $t2, $t1         # $Lo = t2 / t1, Hi = t2 \% t1$ 
mflo    $t5             # $t5 = Lo$ 
mfhi    $t6             # $t6 = Hi$ 

li      $v0, 10         #end and exit
syscall
  
```

◎執行結果:

Registers Coproc 1 Coproc 0			
Name	Number	Value	
\$zero	0	0x00000000	
\$at	1	0x00000000	
\$v0	2	0x0000000a	
\$v1	3	0x00000000	
\$a0	4	0x00000000	
\$a1	5	0x00000000	
\$a2	6	0x00000000	
\$a3	7	0x00000000	
\$t0	8	0x00000000	
\$t1	9	0x00000003	
\$t2	10	0x00000005	
\$t3	11	0x00000005	
\$t4	12	0x00000019	
\$t5	13	0x00000001	
\$t6	14	0x00000002	
\$t7	15	0x00000000	
\$s0	16	0x00000000	
\$s1	17	0x00000000	
\$s2	18	0x00000000	
\$s3	19	0x00000000	
\$s4	20	0x00000000	
\$s5	21	0x00000000	
\$s6	22	0x00000000	
\$s7	23	0x00000000	
\$t8	24	0x00000000	
\$t9	25	0x00000000	
\$k0	26	0x00000000	
\$k1	27	0x00000000	
\$gp	28	0x10008000	
\$sp	29	0x7ffffc	
\$fp	30	0x00000000	
\$ra	31	0x00000000	
pc		0x0040002e	
hi		0x00000002	
lo		0x00000001	

Example4-Input & Output

```
# Example 4 – Input & Output

.data    #宣告變數

#變數名稱:    儲存型態    變數值
input:      .space      12      #儲存使用者輸入值
newline:    .ascii      "\n"    #換行

.text    #程式指令區塊
main:
    la      $t0, input    #載入 input 記憶體位址
    li      $v0, 5        #讀取輸入整數
    syscall

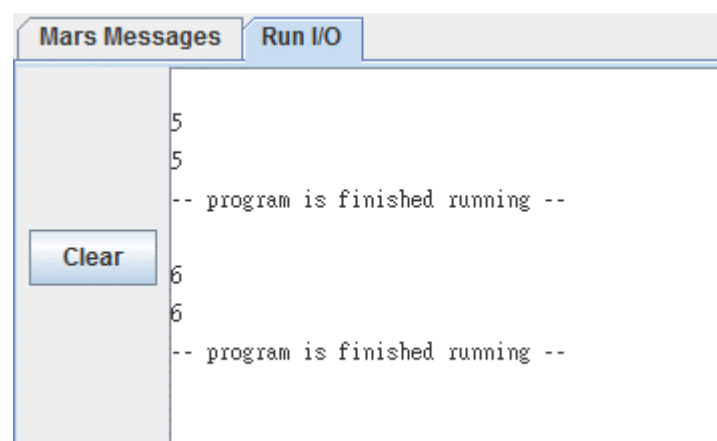
    sw      $v0, ($t0)    #使用者輸入的值儲存在 input

    li      $v0, 4        #列印字串
    la      $a0, newline  #換行
    syscall

    lw      $t3, ($t0)    #讀取 input 值存至$t3 暫存器
    li      $v0, 1        #輸出整數
    move    $a0, $t3      #移動$t3 暫存器內容至$a0 暫#存器中
    syscall

    li      $v0, 10       #end and exit
    syscall
```

◎執行結果:



Mars Messages Run I/O

5
5
-- program is finished running --

Clear

6
6
-- program is finished running --

Example5- Jump

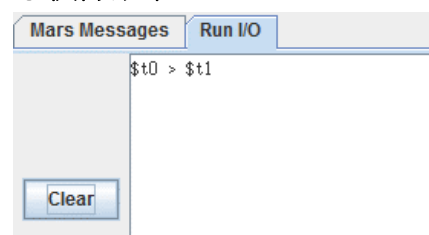
```
# Example 4 – Jump
.data    #宣告變數

#變數名稱:    儲存型態    變數值
string1:    .asciiz    "$t0 < $t1\n" #宣告 12byte 空間來儲存 3 個整數
string2:    .asciiz    "$t0 > $t1\n" #換行

.text    #程式指令區塊

main:
    li      $t0, 2      #$t0=2
    li      $t1, 1      #$t1=1
    slt     $t2,$t0,$t1  #if ($t0<$t1) $t2=1
                        #else $t2=0
    beq     $t2,$zero,L2 #if ($t2==0) goto L2
    bne     $t2,$zero,L1 #if ($t2!=0) goto L1
L1:
    li      $t0, 4      #輸出"$t0 < $t1\n"
    la      $a0,string
    syscall
    j ex     #goto ex
L2:
    li      $t0, 4      #輸出"$t0 < $t1\n"
    la      $a0,string2
    syscall
    j ex     #goto ex
ex:
    li      $v0, 10      #end and exit
    syscall
```

◎執行結果:



附錄

(一)MIPS 暫存器

名稱	暫存器號碼	用途
\$zero	0	常數 0，其值無法修改
\$at	1	保留給組譯器在處理大型常數時使用
\$v0 - \$v1	2 - 3	回傳值暫存器
\$a0 - \$a3	4 - 7	參數暫存器
\$t0 - \$t7	8 - 15	臨時暫存器
\$s0 - \$s7	16 - 23	保留暫存器
\$t8 - \$t9	24 - 25	額外的臨時暫存檔
\$k0	26	保留給 OS 核心使用
\$k1	27	
\$gp	28	全域指標
\$sp	29	堆疊指標
\$fp	30	框指標
\$ra	31	返回位址

(二)MIPS System Service Table

Service	Code in \$v0	Arguments	Result
print_int	1	\$a0 = integer to be printed	
print_float	2	\$f12 = float to be printed	
print_double	3	\$f12 = double to be printed	
print_string	4	\$a0 = address of string in memory	
read_int	5		integer returned in \$v0
read_float	6		float returned in \$v0
read_double	7		double returned in \$v0
read_string	8	\$a0 = memory address of string input buffer \$a1 = length of string buffer (n)	
sbrk	9	\$a = amount	address in \$v0
exit	10		

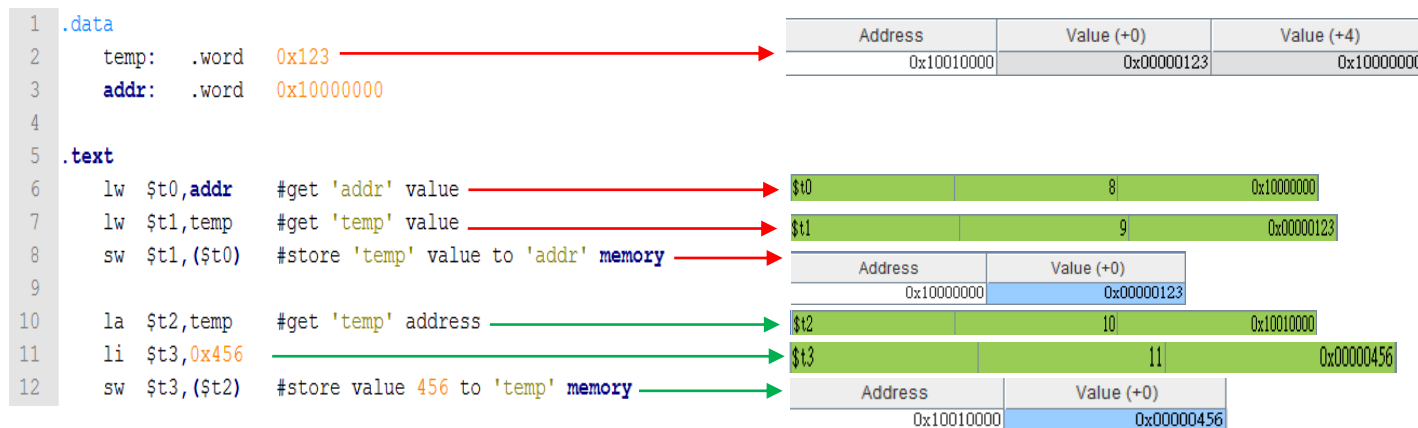
(三)MIPS 儲存型態

儲存型態	範例	說明
.ascii	.ascii str	Store the string str in memory, but do not null-terminate it
.asciiz	.asciiz str	Store the string str in memory and null-terminate it
.byte	.byte b1, b2, b3	Store the n values in successive bytes of memory
.half	.half h1, h2, h3	Store the n 16-bit quantities in successive bytes of memory
.word	.word w1, w2, w3	Store the n 32-bit quantities in successive bytes of memory
.space	.space n	Allocate n bytes of spaces in the current segment

(四)MIPS 常用指令介紹- 載入和儲存

指令名稱	指令用途	指令範例
lui	直接載入高位元	lui \$s1, 4097 [var]
li	直接載入數字	li, \$s1, 1
la	讀取位址	la \$s1, var1
lw	載入字組	lw \$s1, 100(\$2)
lb	載入位元組	lb \$s1, 100(\$2)
lh	載入半字組	lh \$s1, 100(\$2)
sh	儲存半字組	sh \$s1, 100(\$2)
sw	儲存字組	sw \$s1, 100(\$2)
sb	儲存位元組	sb \$s1, 100(\$2)

(五) MIPS 常用指令介紹- 載入和儲存範例



(六) MIPS 常用指令介紹- 比較

指令名稱	指令用途	指令範例	指令說明
slt	小於則設定	slt \$s1, \$s2, \$s3	if (\$s2 < \$s3) \$s1 = 1; else \$s1 = 0
slti	立即小於則定	slti \$s1, \$s2, 100	if (\$s2 < 100) \$s1 = 1; else \$s1 = 0
sltu	無號小於則設定	sltu \$s1, \$s2, \$s3	if (\$s2 < \$s3) \$s1 = 1; else \$s1 = 0
sltiu	無號立即小於則設定	sltiu \$s1, \$s2, 100	if (\$s2 < 100) \$s1 = 1; else \$s1 = 0

(七)MIPS 常用指令介紹- 無條件跳躍

指令名稱	指令用途	指令範例	指令說明
j	跳躍	j L1	goto L1
jr	跳至暫存器	jr \$ra	goto \$ra
jal	跳躍與連結	jal L1	goto L1

1	.data		
2	temp: .word 0x123		
3	addr: .word 0x10000000		
4	.text		
5	jal jump	→	\$ra 31 0x00400004
6	sw \$t0, (\$t1)	→	Address Value (+0) 0x10000000 0x00000123
7	jump:		
8	lw \$t0, temp	→	\$t0 8 0x00000123
9	lw \$t1, addr	→	\$t1 9 0x10000000
10	jr \$ra		

(八)MIPS 常用指令介紹- 條件跳躍

指令名稱	指令用途	指令範例	指令說明
beq	等於則跳躍	beq \$s1, \$s2, L1	if (\$s1 == \$s2) goto L1
bne	不等於則跳躍	bne \$s1, \$s2, L1	if (\$s1 != \$s2) goto L1

1	.data		
2	temp: .word 0x123		
3	temp2: .word 0x456		
4	addr: .word 0x10000000		
5	.text		
6	lw \$t0, temp	→	\$t0 8 0x00000123
7	lw \$t1, temp2	→	\$t1 9 0x00000456
8	bne \$t0, \$t1, jump	→	\$ra 31 0x00000000
9	jump:		
10	lw \$t2, addr	→	\$t2 10 0x10000000
11	sw \$t1, (\$t2)	→	Address Value (+0) 0x10000000 0x00000456

(九)MIPS 常用指令介紹- 邏輯運算

指令名稱	指令用途	指令範例	指令說明
and	及閘	and \$s1, \$s2, \$s3	$\$s1 = \$s2 \& \$s3$
or	或閘	or \$s1, \$s2, \$s3	$\$s1 = \$s2 \$s3$
nor	反或閘	nor \$s1, \$s2, \$s3	$\$s1 = \sim(\$s2 \$s3)$
andi	立即及閘	andi \$s1, \$s2, 100	$\$s1 = \$s2 \& 100$
ori	立即或閘	ori \$s1, \$s2, 100	$\$s1 = \$s2 100$
sll	左移	sll \$s1, \$s2, 10	$\$s1 = \$s2 \ll 10$
srl	右移	srl \$s1, \$s2, 10	$\$s1 = \$s2 \gg 10$

(十)MIPS 常用指令介紹- 算術運算

指令名稱	指令用途	指令範例	指令說明
add	加法	add \$s1, \$s2, \$s3	$\$s1 = \$s2 + \$s3$
sub	減法	sub \$s1, \$s2, \$s3	$\$s1 = \$s2 - \$s3$
addi	立即加法	addi \$s1, \$s2, 100	$\$s1 = \$s2 + 100$
addu	無號加法	addu \$s1, \$s2, \$s3	$\$s1 = \$s2 + \$s3$
subu	無號減法	subu \$s1, \$s2, \$s3	$\$s1 = \$s2 - \$s3$
mult	乘法	mult \$s2, \$s3	Hi, Lo = $\$s2 * \$s3$
multu	無號乘法	multu \$s2, \$s3	Hi, Lo = $\$s2 * \$s3$
div	除法	div \$s2, \$s3	Lo = $\$s2 / \$s3$ Hi = $\$s2 \% \$s3$
divu	無號除法	divu \$s2, \$s3	Lo = $\$s2 / \$s3$ Hi = $\$s2 \% \$s3$
mfhi	搬移 Hi	mfhi \$s1	$\$s1 = \text{Hi}$
mflo	搬移 Lo	mflo \$s1	$\$s1 = \text{Lo}$