

การใช้งาน SPIM

1. สร้าง MIPS assembly source โดยใช้ text editor ให้มีนามสกุลเป็น .s

```
# Program to read a number and print its square
.data
prompt1:      .asciiz      "Enter a number: "
prompt2:      .asciiz      "\n The square of the number is: "
               .globl      main

.text
main:
    li         $v0,4        #System call code for print string
    la         $a0,prompt1  #Load address of the prompt1 string
    syscall
    li         $v0,5        #System call code for read integer
    syscall
    move       $t1,$v0      #Move the integer into $t1
    mul        $t1,$t1,$t1  #Multiply the content of $t1 with itself
    li         $v0,4        #System call code for print string
    la         $a0,prompt2  #Load address of the prompt2 string
    syscall
    move       $a0,$t1      #Load $a0 with the value in $t1
    li         $v0,1        #System call code for printing integer
    syscall
end:
    li         $v0,10       #System call code to Exit
    syscall                #Call OS to Exit the program
```

2. Source code ประกอบด้วย 2 ส่วนคือส่วนของการประกาศข้อมูลที่จะใช้ในโปรแกรมและส่วนของคำสั่ง
3. ส่วนของการประกาศข้อมูล จะอยู่หลัง directive **.data** สำหรับส่วนของคำสั่งจะอยู่หลัง directive **.text**
4. โปรแกรมจะเริ่มทำงานที่ label **main** เท่านั้น
5. Comment จะอยู่หลังเครื่องหมาย **#** ไปจนถึงบรรทัด
6. Directive **.asciiz** เป็นการประกาศ string และให้ปิดท้ายด้วย null character (**.ascii** จะไม่มี null character)
7. Directive **globl** เป็นการประกาศว่าสามารถจะให้ module อื่นๆ นอก file นี้ สามารถจะอ้างถึง main ได้ (เป็น option)
8. ตารางต่อไปนี้แสดงบางส่วนของ system call และ argument ที่ต้องการ

Service	System Call Code	Arguments	Results
print_int	1	\$a0 = integer	
print_float	2	\$f12 = float	
print_double	3	\$f12 = double	
print_string	4	\$a0 = string	
read_int	5		integer in \$v0
read_float	6		float in \$f0
read_double	7		double in \$f0
read_string	8	\$a0 = buffer, \$a1 = length	
sbrk	9	\$a0 = amount	address in \$v0
exit	10		
print_char	12		char in \$a0

open	13	\$a0 = file name (string), \$a1 = flags, \$a2 = mode	file descriptor in \$a0
read	14	\$a0 = file descriptor, \$a1 = buffer, \$a2 = length	num chars read in \$a0
write	15	\$a0 = file descriptor, \$a1 = buffer, \$a2 = length	num chars written in \$a0
close	16	\$a0 = file descriptor	
exit2	17	\$a0 = result	

9. เข้าสู่โปรแกรม SPIM

The screenshot shows the PCSpim simulator interface. At the top, there's a menu bar (File, Simulator, Window, Help) and a toolbar. Below that, the status bar displays PC, Status, EPC, HI, LO, Cause, and BadVAddr. The General Registers section lists R0 through R25 with their current values. The main window displays assembly code with addresses and comments. Below the code, there are sections for DATA and STACK memory. At the bottom, there's a copyright notice and a status bar with PC, EPC, and Cause values.

```

PC      = 00000000  EPC      = 00000000  Cause   = 00000000  BadVAddr= 00000000
Status  = 3000ff10  HI       = 00000000  LO       = 00000000
General Registers
R0 (r0) = 00000000  R8 (t0) = 00000000  R16 (s0) = 00000000  R24 (t8) = 00000000
R1 (at) = 00000000  R9 (t1) = 00000000  R17 (s1) = 00000000  R25 (t9) = 00000000

[0x00400000] 0x8fa40000 lw $4, 0($29)           ; 175: lw $a0 0($sp)
[0x00400004] 0x27a50004 addiu $5, $29, 4         ; 176: addiu $a1 $sp 4
[0x00400008] 0x24a60004 addiu $6, $5, 4         ; 177: addiu $a2 $a1 4
[0x0040000c] 0x00041080 sll $2, $4, 2           ; 178: sll $v0 $a0 2
[0x00400010] 0x00c23021 addu $6, $6, $2         ; 179: addu $a2 $a2 $v

DATA
[0x10000000]...[0x10040000] 0x00000000

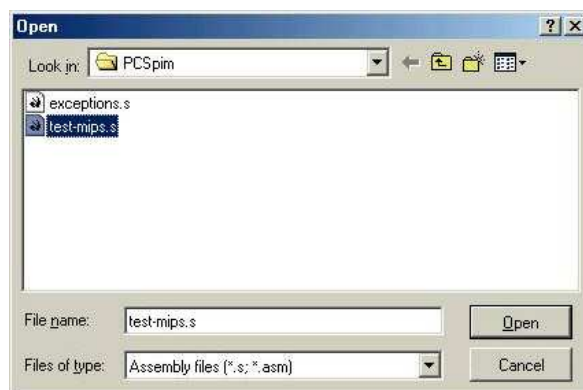
STACK
[0x7ffffeffc] 0x00000000

All Rights Reserved.
DOS and Windows ports by David A. Carley (dac@cs.wisc.edu).
Copyright 1997 by Morgan Kaufmann Publishers, Inc.
See the file README for a full copyright notice.
Loaded: D:\Program Files\PCSpim\exceptions.s

For Help, press F1
PC=0x00000000 EPC=0x00000000 Cause=0x00000000

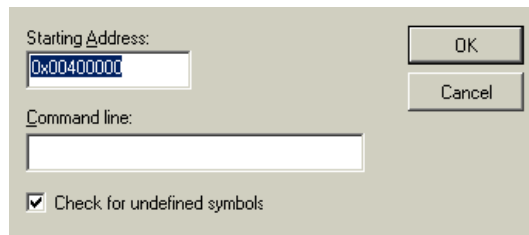
```

10. Load source code



11. Run โปรแกรมโดยกด F5 หรือคลิกที่  หรือไปที่เมนู Simulator -> Go

12. เลือก OK



13. ผลการทำงานจะแสดงใน Console Window



14. ศึกษารายละเอียดเพิ่มเติมใน Appendix A