

# Programação Em Nível de Instruções com SPIM

- Antes de prosseguir, vejamos um simulador (SPIM) para a programação em linguagem de máquina baseado no MIPS:

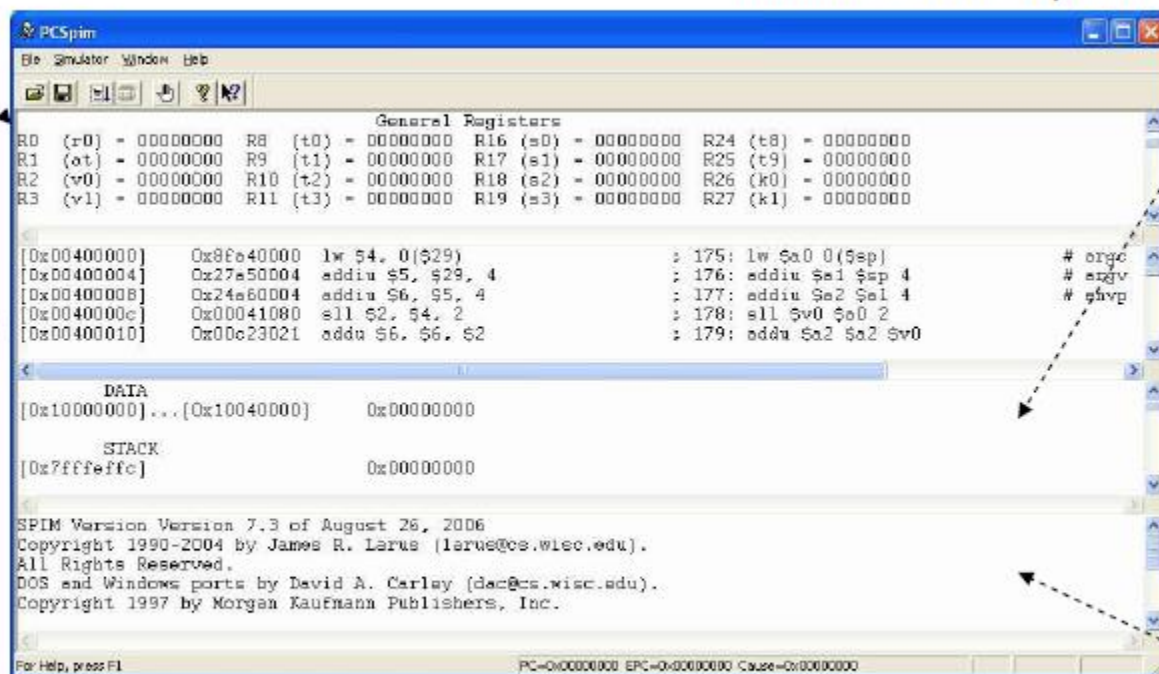
The screenshot shows the PCSpim MIPS simulator interface. The title bar is 'PCSpim'. The menu bar includes 'File', 'Simulator', 'Window', and 'Help'. Below the menu bar is a toolbar with icons for file operations and simulation control. The main window is divided into several sections:

- General Registers:** A table showing the state of 32 MIPS registers. Registers R0-R3 are labeled with names like \$r0, \$a0, etc. Registers R4-R31 are labeled with temporary (\$t0-\$t7), saved (\$s0-\$s7), and kernel (\$k0-\$k1) names. All registers currently show a value of 00000000.
- Instructions:** A list of MIPS instructions with their hexadecimal addresses and assembly code. For example:
  - 0x00400000: lw \$4, 0(\$29)
  - 0x00400004: addiu \$5, \$29, 4
  - 0x00400008: addiu \$6, \$5, 4
  - 0x0040000c: sll \$2, \$4, 2
  - 0x00400010: addu \$6, \$6, \$2
- DATA:** A section showing memory data at address 0x10000000, with a value of 0x00000000.
- STACK:** A section showing the stack pointer at address 0x7fffffc, with a value of 0x00000000.
- Footer:** Copyright information for SPIM Version 7.3, dated August 26, 2006, by James R. Larus and David A. Carley.

The status bar at the bottom indicates 'PC=0x00000000 EPC=0x00000000 Cause=0x00000000'.

Todos os Registradores  
da CPU MIPS

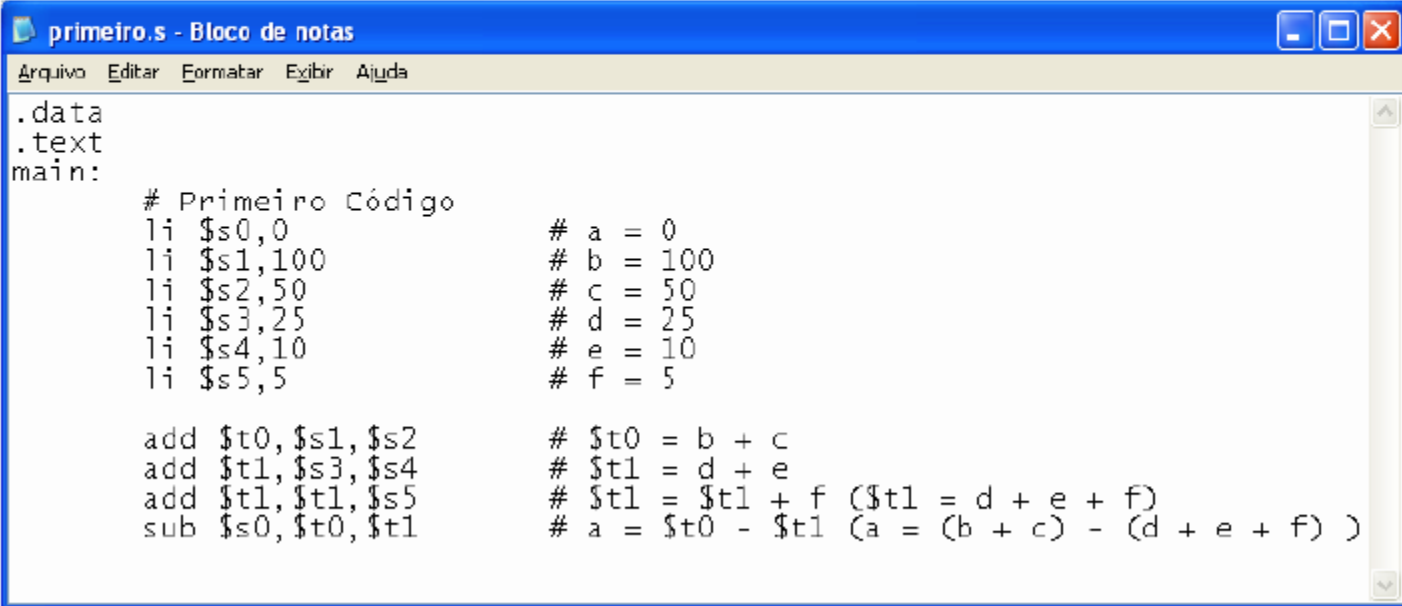
Display com os dados carregados  
na memória do programa e os  
dados sobre a pilha de programa



Display com as instruções tanto do  
programa quanto do sistema carregado  
automaticamente quando o SPIM roda.

Por fim, a área reservada para escrita  
de mensagens de erro, por exemplo

- Carregando o primeiro código assembly no SPIM:



```
primeiro.s - Bloco de notas
Arquivo  Editar  Formatar  Exibir  Ajuda

.data
.text
main:
    # Primeiro Código
    li $s0,0           # a = 0
    li $s1,100         # b = 100
    li $s2,50          # c = 50
    li $s3,25          # d = 25
    li $s4,10          # e = 10
    li $s5,5           # f = 5

    add $t0,$s1,$s2     # $t0 = b + c
    add $t1,$s3,$s4     # $t1 = d + e
    add $t1,$t1,$s5     # $t1 = $t1 + f ($t1 = d + e + f)
    sub $s0,$t0,$t1     # a = $t0 - $t1 (a = (b + c) - (d + e + f) )
```

# Primeiro Exemplo

algoritmo "Primeiro Assembly"

var

// e:inteiro  
a, k:inteiro  
b, f:inteiro  
c, g:inteiro  
d, h:inteiro

inicio

a <- 0  
b <- 100  
c <- 50  
d <- 25  
k <- 10  
f <- 5  
g <- b + c  
h <- d + k  
h <- h + f  
a <- g - h  
escreval (a)

fimalgoritmo

primeiroexemplo.s - Bloco de notas

Arquivo Editar Formatar Exibir Ajuda

.data  
.text  
main:

#Primeiro Exemplo

li \$s0, 0	#int a = 0
li \$s1, 100	#int b = 100
li \$s2, 50	#int c = 50
li \$s3, 25	#int d = 25
li \$s4, 10	#int e = 10
li \$s5, 5	#int f = 5

add \$t0, \$s1, \$s2	#t0 = b + c
add \$t1, \$s3, \$s4	#t1 = d + e
add \$t1, \$t1, \$s5	#t1 = t1 + f
sub \$s0, \$t0, \$t1	#a = t0 - t1

# Segundo Exemplo

```
algoritmo "Segundo Assembly"
var
    // e:inteiro
    a, b:inteiro

inicio
    escreval("1º Valor: ")
    leia(a)
    escreval("2º Valor: ")
    leia(b)
    escreval("O Maior é : ")
    se a > b entao
        escreval(a)
    senao
        escreval(b)
    fimse
fimalgoritmo
```

```
.data
msg1: .asciiz "\nDigite um valor inteiro: "
msg2: .asciiz "\nO Maior é : "

.text
main:
    #Segundo Exemplo - Verifica o maior entre 2 números inteiros

    #Lê o primeiro valor:
    li $v0, 4                # Código SysCall p/ escrever strings
    la $a0, msg1             # Parametro (string a ser escrita)
    syscall
    li $v0, 5                # Código SysCall p/ ler inteiros
    syscall
    add $t0, $v0, $zero      # Inteiro lido vai ficar em $t0

    #Lê o segundo valor:
    li $v0, 4
    la $a0, msg1
    syscall
    li $v0, 5
    syscall
    add $t1, $v0, $zero

    #Estrutura de Decisão
    bgt $t0, $t1, Se         #Se t0 > t1
    add $s0, $t1, $zero
    j Exit

Se:
    add $s0, $t0, $zero

Exit:
    li $v0, 4
    la $a0, msg2
    syscall
    li $v0, 1                # Código SysCall p/ escrever inteiros
    add $a0, $zero, $s0      # Parametro (inteiro a ser escrito)
    syscall

    #Pausa
    li $v0, 5
    syscall
```

# Terceiro Exemplo

```
algoritmo "Primeiro Assembly"
var
    // e:inteiro
    a, b:inteiro
    soma, media:inteiro
    d:inteiro
inicio
    soma <- 0
    escreval("Digite Valor: ")
    leia(a)
    enquanto a <> 0 faca
        soma <- soma + a
        escreval("Digite Valor: ")
        leia(a)
    fimenquanto

    escreval("Soma : ")
    escreval(soma)
finalgoritmo
```

```
.data
    msg1: .asciiz "\nDigite um valor inteiro: "
    msg2: .asciiz "\nA soma dos valores digitados é : "

.text
main:
    #Terceiro Exemplo - Soma os inteiro inseridos pelo usuário
    #até que se entre com o valor zero

    #Inicializando a posição de memória
    li $t0, 0

    #Estrutura de Laço
    Enquanto:
        li $v0, 4
        la $a0, msg1
        syscall
        li $v0, 5
        syscall
        add $t1, $v0, $zero
        add $t0, $t0, $t1
        beq $t1, 0, Fimenquanto
    j Enquanto

    #Escreve a soma
    Fimenquanto:
        add $s0, $t0, $zero
        li $v0, 4
        la $a0, msg2
        syscall
        li $v0, 1
        add $a0, $zero, $s0
        syscall

    #Pausa
    Exit:
        li $v0, 5
        syscall
```

# Quarto Exemplo

algoritmo "Quinto Assembly"

var

// e:inteiro

a, b, c, d:real

inicio

a <- 2.5

b <- 4.3

c <- 6.7

d <- a + b

d <- d - c

fimalgoritmo

.data

.text

main:

#Primeiro Exemplo Ponto Flutuante

li.s \$f1, 2.5        #int a = 2.5

li.s \$f2, 4.3        #int b = 4.3

li.s \$f3, 6.7        #int c = 6.7

add.s \$f4,\$f1,\$f2        #f0 = a + b

sub.s \$f4,\$f4,\$f3        #f0 = f0 - c



# Quinto Exemplo

```
algoritmo "Sexto Assembly"
var
    // e:inteiro
    a, b, c:real

inicio
    escreval("1º Valor: ")
    leia(a)
    escreval("2º Valor: ")
    leia(b)
    c <- a + b
    escreval("Soma = ",c)
fimalgoritmo
```

```
.data
    msg1: .asciiz "\nDigite um valor real: "
    msg2: .asciiz "\nA soma é : "
.text
main:
    #Segundo Exemplo Ponto Flutuante - Operação de soma entre dois números

    #Lê o primeiro valor:
    li $v0, 4                # Código SysCall p/ escrever strings
    la $a0, msg1             # Parametro (string a ser escrita)
    syscall
    li $v0, 6                # Código SysCall p/ ler reais para $f0
    syscall
    mov.s $f1, $f0           # Real lido vai ser movido para $f1

    #Lê o segundo valor:
    li $v0, 4
    la $a0, msg1
    syscall
    li $v0, 6
    syscall
    mov.s $f2, $f0           # Real lido vai ser movido para $f1

    #Faz soma:
    add.s $f3, $f1, $f2

    #Escrita:
    li $v0, 4
    la $a0, msg2
    syscall
    li $v0, 2                # Código SysCall p/ escrever números
    mov.s $f12, $f3          # Parametro (real a ser escrito)
    syscall
```

# Exercícios

- Fazer, em SPIM, um algoritmo que leia 15 números, a partir de uma estrutura de laço e diga qual o menor
- Fazer, em SPIM, um algoritmo que leia um número e diga se é par ou ímpar
- Fazer, em SPIM, um algoritmo que calcule a média da disciplina de AOC.