

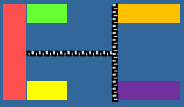
PRÁCTICA 3.2

INTRODUCCIÓN AL LENGUAJE

ENSAMBLADOR - MARS

dtic





Implementación de estructuras de salto condicional

Salto condicional

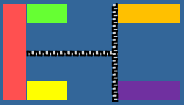
1. Implementación de instrucciones de salto condicional

Las dos instrucciones utilizadas son *beq* y *bne* cuyos formatos son, respectivamente:

`beq $t0, $t1, etiqueta`

`bne $t0, $t1, etiqueta`

Ambas instrucciones comparan el contenido de los registros \$t0 y \$t1 y, dependiendo del resultado de esta comparación, saltan a la etiqueta indicada. La comparación puede ser cierta (*beq*) o no (*bne*).



Implementación de estructuras de comparación con 0

Comparación con 0

En algunos casos resulta muy útil comparar el contenido de un registro con 0, en cuyo caso se utilizan las pseudoinstrucciones:

`bgez $t0, etiqueta` (cierta cuando $\$t0 \geq 0$)

`bgtz $t0, etiqueta` (cierta cuando $\$t0 > 0$)

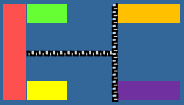
`blez $t0, etiqueta` (cierta cuando $\$t0 \leq 0$)

`bltz $t0, etiqueta` (cierta cuando $\$t0 < 0$)

Estas cuatro instrucciones pueden utilizarse en forma de pseudoinstrucciones, para comparar el contenido de dos registros ($\$t0$ y $\$t1$), siendo su formato genérico el siguiente:

`bXX $t0, $t1, etiqueta`

donde XX puede ser *ge* (cierta cuando $\$t0 \geq \$t1$), *gt* (cierta cuando $\$t0 > \$t1$), *le* (cierta cuando $\$t0 \leq \$t1$) o *lt* (cierta cuando $\$t0 < \$t1$).



Implementación de estructuras de comparación

Instrucciones de comparación

2. Implementación de instrucciones de comparación.

Para comparar el contenido de dos registros, MIPS dispone de la instrucción *slt* cuyo formato es el siguiente:

`slt $t2, $t0, $t1`

El resultado de la comparación es el siguiente: si el contenido de \$t0 es menor que el de \$t1, \$t2 se carga con 1; de lo contrario se carga con un 0. También se pueden utilizar las pseudoinstrucciones siguientes, que almacenan en \$t2 un 1 o un 0 según se cumpla o no la condición de comparación:

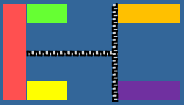
`sge $t2, $t0, $t1` (cierta si $\$t0 \geq \$t1$)

`sgt $t2, $t0, $t1` (cierta si $\$t0 > \$t1$)

`sle $t2, $t0, $t1` (cierta si $\$t0 \leq \$t1$)

`sne $t2, $t0, $t1` (cierta si $\$t0 \neq \$t1$)

`sqe $t2, $t0, $t1` (cierta si $\$t0 = \$t1$)



Implementación de estructuras de comparación

Comparación con constante

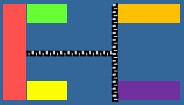
También es posible comparar el contenido de un registro con una constante:

`slti $t1,$t0, 84` (compara el contenido de \$t0 con la constante 84; si $\$t0 < 84$, $\$t1=1$, si no $\$t1=0$)

Para comparar números enteros se utilizan las instrucciones:

`sltu $t2,$t0,$t1` (cierta si $\$t0 < \$t1$)

`sltiu $t1,$t0, 84` (cierta si $\$t0 < 84$)



Implementación de estructuras de pseudosalto

Pseudosaltos

La mayoría de los pseudosaltos son implementados usando `slt`. Por ejemplo, un salto si es menor que (branch-if-less-than) `blt $a0, $a1, Label` equivale a lo siguiente:

```
SlT $at,$a0,$a1 // $at = 1 si $a0 < $a1
```

```
bne $at,$0,Label // Salto si $at != 0
```

Esto permite saltos con inmediatos, los cuales son también pseudoinstrucciones. Por ejemplo, `blti $a0, 5, Label` equivale a las dos instrucciones siguientes:

```
slti $at,$a0,5 // $at = 1 si $a0 < 5
```

```
bne $at,$0,Label // Branch if $a0 < 5
```

Todos los pseudosaltos necesitan un registro para guardar los resultados de `slt`, aunque no se necesite después.

El ensamblador de MIPS usa el registro `$1`, o `$at`, como registro temporal.

Se debe ser cuidadoso cuando se usa `$at` en un programa, ya que puede ser reescrito por el código generado por el ensamblador.



Implementación de estructuras de salto incondicional

Salto
incondicional

3. Implementación de instrucciones de salto incondicional

j etiqueta

Salta a la etiqueta. Por ejemplo: j menu

j nº instrucción

Salta a la instrucción. Por ejemplo: j 10000

j registro

Salta a la instrucción contenida en el registro

jal nº instrucción

Utilizada en llamadas a procedimientos.



Ejercicio propuesto

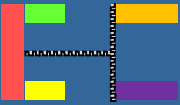
Práctica 3.2

Introducción a la lectura e impresión de enteros y cadenas de caracteres y a las instrucciones de salto.

Objetivos:

- 1.- Programación de menús con opciones de enteros.
- 2.- Impresión por pantalla de cadenas y enteros.
- 3.- Introducción al manejo de las instrucciones de salto.

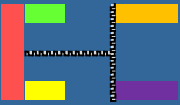
Carga el siguiente código y grábalo con el nombre `menu_basico.asm`. Ejecútalo y comprueba su funcionamiento.



Ejercicio propuesto

Práctica 3.2

```
#ESTRUCTURA DE COMPUTADORES
#CURSO 2015-2016
#EJEMPLO DE MENÚ GENÉRICO CON BIFURCACIONES
.data
cadena1: .ascii "\n\n MENU DE OPERACIONES\n"
cadena2: .ascii "1.- Sumar números\n"
cadena3: .ascii "2.- Restar números\n"
cadena4: .ascii "3.- Multiplicar números\n"
cadena5: .ascii "4.- Salir\n\t"
cadena6: .ascii "Introduce opcion: "
cadena61: .ascii "Opción no válida."
cadena7: .ascii "Introduce el primer número: "
cadena8: .ascii "Introduce el segundo número: "
cadena9: .ascii "La suma es "
cadena10: .ascii "La resta es "
cadena11: .ascii "El producto es "
cadena12: .ascii "\n\nFin del programa. Adios ..."
.text
main:
menu:
li $v0, 4           #LLamada a imprimir cadena
la $a0, cadena1      #Muestra la cadena por pantalla
syscall
li $v0, 4
la $a0, cadena2
syscall
li $v0, 4
la $a0, cadena3
syscall
li $v0, 4
la $a0, cadena4
syscall
```



Ejercicio propuesto

Práctica 3.2

```
li $v0, 4
la $a0, cadena5
syscall
valmaxmin:
li $t8, 1      #Almacena el valor mínimo posible de la opción
li $t9, 4      #Almacena el valor máximo posible de la opción
opcion:
la $a0, cadena6
li $v0, 4
syscall
li $v0, 5      #Lee el valor introducido por teclado
syscall
move $t1, $v0  #Almacena el valor leído
blt $t1, $t8, volver    #Compara el valor leído con el valor mínimo.
bgt $t1, $t9, volver    #Compara el valor leído con el valor máximo.
seleccion:
move $s0, $v0
beq $s0, 1, sumar      #Si el valor introducido es 1 salta a sumar:
beq $s0, 2, restar      #Idem si es 2
beq $s0, 3, multiplicar #Idem si es 3
beq $s0, 4, fin         #Finaliza la ejecución si es 4

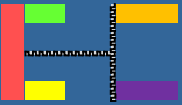
sumar:
li $v0, 4
la $a0, cadena7
syscall
li $v0, 5
syscall
move $t0, $v0
li $v0, 4
la $a0, cadena8
```



Ejercicio propuesto

Práctica 3.2

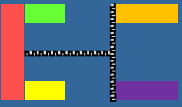
```
syscall
li $v0, 5
syscall
move $t1, $v0
li $v0, 4
la $a0, cadena9
syscall
add $a0, $t0, $t1      #Efectúa la suma
li $v0, 1              #Muestra por pantalla el mensaje correspondiente
syscall
j menu                 #Vuelve a mostrar el menú de operaciones
restar:
li $v0, 4
la $a0, cadena7
syscall
li $v0, 5
syscall
move $t0, $v0
li $v0, 4
la $a0, cadena8
syscall
li $v0, 5
syscall
move $t1, $v0
li $v0, 4
la $a0, cadena10
syscall
sub $a0, $t0, $t1
li $v0, 1
syscall
```



Ejercicio propuesto

Práctica 3.2

```
j menu
multiplicar:
li $v0, 4
la $a0, cadena7
syscall
li $v0, 5
syscall
move $t0, $v0
li $v0, 4
la $a0, cadena8
syscall
li $v0, 5
syscall
move $t1, $v0
li $v0, 4
la $a0, cadena11
syscall
mul $a0, $t0, $t1
li $v0, 1
syscall
fin:
la $a0, cadena12
li $v0, 4
syscall
li $v0, 10
syscall
volver:
la $a0, cadena61
li $v0, 4
syscall          #muestra por pantalla el mensaje Opción no válida
j opcion          #Muestra el menú de operaciones
```



Ejercicio propuesto

Práctica 3.2

Cuestiones

- 1.- ¿Qué instrucciones utiliza para detectar las entradas por teclado erróneas?
Explica su funcionamiento.
- 2.- Comenta el propósito de las instrucciones de salto que se han utilizado.
- 3.- Mejora el programa añadiendo la opción Dividir.
- 4.- Mejora el programa eliminando la opción Salir de forma que, cuando finaliza la ejecución de una de las operaciones del menú aparezca por pantalla el mensaje “¿Deseas realizar otra operación S/N?” Si la respuesta es S presenta de nuevo el menú y si es N finaliza la ejecución del programa con un mensaje de despedida.



Ejercicio propuesto

Práctica 3.2

- ④ La realización de la práctica consiste en la contestación de cada una de las seis cuestiones del ejercicio propuesto.
- ④ Se debe entregar una memoria con la respuesta a cada una de las cuestiones planteadas. Cuando proceda, añadir volcados de pantalla de la ejecución para verificar su correcto funcionamiento. El documento de la memoria tendrá el siguiente formato:
 - Página 1: Nombre de la asignatura, título de la práctica, número de la práctica, nombre alumno, email alumno, D.N.I. del alumno, grupo teoría, fecha.
 - Página 2: Índice de la práctica.
 - Página 3: listado de los archivos que entrega.
 - Página 4 y sucesivas el resto de la práctica (descripción de la práctica, qué hace, cómo lo hace, problemas surgidos, volcados de pantalla de la simulación, ...).
 - Las páginas deben tener número de página.
- ④ Se debe entregar también el código en ensamblador del programa con las modificaciones solicitadas.