**PRÁCTICA 2.3  
Programación en ensamblador MIPS. Bucles. Modos de direccionamiento**

**Objetivos:**

* Conocer el formato de las instrucciones del MIPS
* Modos de direccionamiento
* Manejo vectores

**Duración: 1,5** horas presenciales + trabajo personal en casa

**Desarrollo / Comentario:**

**Parte 1)** Análisis de un Programa Ejemplo.

Estudia el programa ejemplo que se muestra a continuación para comprender su funcionamiento.

#segmento de datos

.data

vectorx: .word 0x1ab25,20,5,0x12345678,0xff0000ff,30,26,0x10

vectory: .space 32

const: .word 0x310000

tam: .word 8

# segmento de texto

.text

.globl main

main:

la $t0, vectorx # $t0 tenemos la dirección de inicio de vectorx

la $t1, vectory # $t1 tenemos la dirección de inicio de vectory

la $t2, const

lw $t3, 0($t2) # $t3 tenemos el contenido de const

la $t2, tam

lb $t4, 0($t2) # $t4 tenemos nº de elementos del vector

bucle:

beq $t4,$0,fin #si nº de elementos a sumar es 0 voy a fin

lw $t5, 0($t0) #leo elemento de vectorx

add $t5, $t5, $t3 #le sumo la cte

sw $t5, 0($t1) #escribo la suma en vectory

addi $t0, $t0, 4 #apunto al siguiente elemento del vectorx

addi $t1, $t1, 4 #apunto al siguiente elemento del vectory

addi $t4, $t4, -1 #decremento el número de elementos a sumar

j bucle

fin:

addi $v0, $0, 10 # llamada para salir del programa

syscall

Responde a las siguientes cuestiones:

1. Indica qué hace el programa.

Suma cada elemento de vectorx con la constante 0x310000 y guarda el resultado en la posición correspondiente de vectory.

1. Recordemos que las etiquetas representan a direcciones. A qué dirección (en hexadecimal) representan las siguientes etiquetas:

|  |  |
| --- | --- |
| vectorx | 0x10010000 |
| vectory | 0x10010020 |
| const | 0x10010040 |
| main | 0x00400000 |
| bucle | 0x00400028 |
| fin | 0x00400048 |

1. Una vez ejecutado el programa, indica el valor que toman los bytes almacenados en las siguientes direcciones:

|  |  |
| --- | --- |
| vectory +0 | 25 |
| vectory+1 | AB |
| vectory+2 | 32 |
| vectory+3 | 00 |
| 0x10010000 | 25 |

1. Obtener el código máquina de las siguientes instrucciones del programa

|  |
| --- |
| beq $t4,$0,fin  000100 01100 00000 0000000000000111  0x11800007 |
| j bucle  0000 1000 0001 0000 0000 0000 0000 1010  0x0810000A |

1. Indica los modos de direccionamiento de las siguientes instrucciones: (Recordemos que hay un modo de direccionamiento por cada operando)

|  |  |
| --- | --- |
| lw $t5, 0($t1) | $t5: directo a registro, 0($t1): relativo a registro base |
| addi $t1, $t1, 4 | $t1: directo a registro, 4: inmediato |
| beq $t4,$0,fin | $t4: directo a registro, $0: directo a registro, fin: relativo a PC |
| j bucle | bucle: pseudodirecto |

**Parte 2)** Realizar un programa que:

* Contenga las directivas para:
  + Almacenar dos vectores de enteros en memoria inicializados con 8 valores.
  + Reservar memoria para almacenar un dato de tipo *word*. La dirección de este dato se corresponderá con la etiqueta num\_ele e indicará el número de elementos del vector con los que queremos trabajar.
* Contenga las instrucciones para:
  + Intercambiar el contenido de los dos vectores almacenados en memoria, sabiendo que el número de elementos a intercambiar viene dado por la variable num\_ele.
  + Es obligatorio utilizar bucles