PRÁCTICA 4
LENGUAJE MAQUINA PROCESADOR RISC -V. VECTORES

Objetivos:

- Familiarizarse con la arquitectura del RISC-V
- Manejo de vectores:

Desarrollo:

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 0x5f01ab25,-3,2,0xffffffff,-1
vectory:    .word 0,0,0,0,0
const:      .word 0x3
tam:        .word 4

# 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

        lw t3, const      #t3 tenemos una constante
        lw t4, tam        #t4 tenemos nº de elementos del vector

bucle:
        beq t4,zero,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 vector
        addi t1, t1, 4
        addi t4, t4, -1    #decremento el número de elementos a sumar
        j bucle
fin:
        addi a7, zero, 10  #llamada para salir del programa
        ecall
```

Responde a las siguientes cuestiones:

a. Indica qué hace el programa.

El programa recorre un vector (vectorX) de tamaño 4 (5 en realidad porque es 0,1,2,3,4), en él va sumando 3 a cada valor una sola vez y lo almacena en el siguiente vector (vectorY). Hasta que el número de elementos no sea 0 no saldrá del bucle y seguirá ejecutando esta acción.

b. Recordemos que las etiquetas representan a direcciones. A qué dirección (en hexadecimal) representan las siguientes etiquetas:
Suponiendo que los datos están cargados a partir de la dir. de memoria 0x10000000
Suponiendo que el programa se ejecuta a partir de la dir. de memoria 0x00000000

vectorx	0x10000000
vectory	0x10000014
const	0x10000028
main	0x00000000
bucle	0x00000020
fin	0x00000040

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

vectory +0	0x28
vectory+1	0xab
vectory+2	0x01
vectory+3	0x5f

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

Código sin acabar, muy complejo y falta de registros

```
#segmento de datos
        .data
vectorx:   .word 0,1,2,3,4,5,6,7
vectory:   .word 7,6,5,4,3,2,1,0
vectorz:   .word 0,0,0,0,0,0,0,0 #lo necesitaremos para copiar un vector
num_ele:   .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, vectorz                  #t2 tenemos la dirección de inicio de vectorz

        lw t3, num_ele     #t3 tenemos una constante

buclecopia:
        beq t3,zero,buclecambio #si nº elementos = 0; salto a buclecambio
        lw t5, 0(t0)          #leo elemento de vectorx
        sw t5, 0(t2)          #escribo elemento en vectorz
        addi t0, t0, 4        #apunto al siguiente elemento del vector
        addi t2, t2, 4
        addi t3, t3, -1       #decremento el número de elementos a copiar
        j buclecopia

buclecambio:
        beq t3,zero,copia2    #si nº elementos = 0; salto a fin
        lw t5, 0(t1)        #leo elemento de vectory
        sw t5, 0(t0)        #escribo elemento en vectorx
        addi t1, t1, 4     #apunto al siguiente elemento del vector
        addi t0, t0, 4
        addi t3, t3, -1    #decremento el número de elementos a escribir
        j buclecopia

copia2:
        beq t3,zero,fin    #si nº elementos = 0; salto a fin
        lw t5, 0(t2)        #leo elemento de vectorz
        sw t5, 0(t1)        #escribo elemento en vectory
        addi t2, t2, 4     #apunto al siguiente elemento del vector
        addi t1, t1, 4
        addi t3, t3, -1    #decremento el número de elementos a escribir
        j copia2


fin:
        addi a7, zero, 10  #llamada para salir del programa
        ecall
```

```
#segmento de datos
            .data
vectorx:    .word 0,1,2,3,4,5,6,7
vectory:    .word 7,6,5,4,3,2,1,0
num_ele:    .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

        lw t3, num_ele    #t3 tenemos el tamaño de arrays


bucle:
        beq t3,zero,fin    #si nº de elementos a sumar es 0 voy a fin
        lw t4, 0(t0)       #leo elemento de vectorx
        lw t5, 0(t1)       #leo elemento de vectory

        sw t5, 0(t0)       #escribo el vectorx en el lugar del vectory
        sw t4, 0(t1)       #escribo el vectory en el lugar del vectorx
        addi t0, t0, 4     #apunto al siguiente elemento del vector
        addi t1, t1, 4
        addi t3, t3, -1    #decremento el número de elementos a cambiar
        j bucle
fin:
        addi a7, zero, 10  #llamada para salir del programa
        ecall
```

Parte 3) Completar el siguiente programa para que obtenga el elemento mayor del vector V1 de 8 elementos y lo almacene en la variable mayor.

```
      #segmento de datos
      .data
V1:    .word 1,2,3,0x400,51, 6,7,8
mayor: .word 0

      # segmento de texto
      .text
      .globl main
main:
      la t0, V1       #t0 tenemos la dirección de inicio de V1
      lw t1, mayor    #t1 tenemos el número mayor inicializado a 0


bucle:
      lw t2, 0(t0)
      bgt t2, t1, cambiomayor #si t2 es mayor que t1, salto a cambiomayor
      addi t0, t0, 4          #paso al siguiente elemento de V1
      j bucle


cambiomayor:
      sw t2, mayor            #guardo en mayor el nuevo mayor
      addi t0, t0, 4          #paso al siguiente elemento de V1
      j bucle

final:
      addi a7, zero, 10 # llamada para salir del programa
      ecall


      Solución profe (imprime el resultado, no lo pide)
      .data
V1:     .word 1,2,3,0x400,51, -6,7,8
mayor: .word 0

      .text
      .globl main
main:
      li t0,8                  #numero elementos del vector
      la t1, V1
      lw a0, 0(t1)            #en a0 llevaré el elemento mayor
      addi t0, t0, -1

bucle:  addi t1, t1, 4
      beq t0, zero, final
      lw t4, 0(t1)
      blt t4,a0, salto
      add a0,zero,t4          #en a0 voy guardando el elemento mayor
```

```
salto:
        addi t0, t0, -1
        j bucle
final:
        la t4, mayor
        sw a0, 0(t4)

        addi a7, zero, 34 # imprimo en consola el mayor
        ecall

        addi a7, zero, 10 # llamada para salir del programa
        ecall
```

Diferencias: yo modifico el dato y ella lo cambia en a0 para poder imprimirlo luego