Sesión 3: El procesador segmentado

Procesador segmentado.

El objetivo de esta parte es comprender el funcionamiento del procesador segmentado y practicar los conceptos vistos en teoría. Carga y ejecuta (usando la configuración segmentada y salto fijo) el siguiente código:

```
# programa segmentado.s
#
# segmento de texto

.text
.globl main

main:
  la $t0, array
  la $t1, count
  lw $t1, 0($t1)
  addi $t2, $0, 1
  addi $t0, $t0, 20

Loop:
  lw $t3, 0($t0)
  add $t3, $t3, $t2
  sw $t3, 0($t0)
  addi $t1, $t1, -1
  addi $t0, $t0, -4
  bne $t1, $0, Loop


fin:
  addi $v0,$0,10      # la llamada para salir del programa
  syscall

# segmento de datos
.data
array: .word 3, 4, 9, 8, 5, 1
count: .word 6

# fin
```

Analiza qué está haciendo el código y contesta a la primera batería de preguntas del cuestionario habilitado en la página de la asignatura en el campus virtual de la UDC (`http://campusvirtual.udc.gal`).

Procesador segmentado e instrucciones en punto flotante

Carga y ejecuta (usando la configuración segmentada, salto fijo con una unidad de suma en punto flotante segmentada con latencia 2 y una de multiplicación segmentada con latencia 5) el siguiente código:

```
.data

Num: .word 4
vector: .float 5, 4, 3, 4, 6
datos: .float 1, 2

.text
.globl main

main:

          la $s4, Num
          la $s5, vector
          la $t0, datos
          lwc1 $f2, 0($t0)
          lwc1 $f4, 4($t0)
          lw $s0, 0($s4)
          addi $s1, $0, 0

Loop:
          slt $t1, $s1, $s0
          beq $t1, $zero, fin
          mul.s $f6, $f2, $f4
          lwc1 $f0, 0($s5)
          add.s $f2, $f0, $f2
          swc1 $f2, 0($s5)

          addi $s1, $s1, 1
          addi $s5, $s5, 4
          j Loop

fin:
       swc1 $f6, 0($t0)
       addi $v0, $0, 10
       syscall
```

Analiza qué está haciendo el código y contesta a la segunda batería de preguntas del cuestionario habilitado en la página de la asignatura en el campus virtual de la UDC (`http://campusvirtual.udc.gal`)).

Técnicas de procesamiento de salto

El objetivo de esta parte es comparar las dos técnicas de procesamiento de salto que se estudian en teoría: salto fijo no efectivo y salto retardado. También se pretende comprobar en qué consiste la técnica de salto retardado y cómo se puede mejorar el código con técnicas de optimización.

Cargar y ejecutar en la configuración segmentada el siguiente código:

```
.data

N: .word 10
v1: .float 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
v2: .float 1, 2, 3, 1, 2, 3, 1, 2, 3, 1
r: .float 0

.text
.globl main

main:

          la $t0, N
          lw $t0, 0($t0)
          la $t1, v1
          la $t2, v2
          la $t3, r
          lwc1 $f12, 0($t3)

Loop:
          lwc1 $f2, 0($t1)
          lwc1 $f4, 0($t2)
          add.s $f12, $f12, $f4
          add.s $f2, $f2, $f4
          swc1 $f2, 0($t2)
          addi $t1, $t1, 4
          addi $t2, $t2, 4
          addi $t0, $t0, -1
          bne $t0, $0, Loop

fin:
          swc1 $f12, 0($t3)
          addi $v0, $0, 2
          syscall

          addi $v0, $0, 10
          syscall
```

Analiza qué está haciendo el código y contesta a la batería de preguntas del cuestionario habilitado en la página de la asignatura en el campus virtual de la UDC (`https://campusvirtual.udc.gal`).