Arquitecturas y Organización de Computadoras I
2° Cuatrimestre

TP N° 10 – Programación en lenguaje ensamblador MIPS con MIPSX

Facultad de Informática

Objetivo: Comprender la estructura de un programa en lenguaje ensamblador MIPS, convención de llamada a procedimientos y funciones.

Recursos y Bibliografía:

Arq. MIPS Vol I, II and III.
Programa mipsx desarrollado por la cátedra.
Apunte MIPS.

Utilice la siguiente captura de pantalla del desarrollo de un programa en lenguaje ensamblador MIPS para responder las preguntas. El sistema MIPS en cuestión es **Big Endian**. Las celdas de memoria no inicializadas contienen el valor cero.

Editor del programa

```
.data

memoria:
    .word -15, -1
    .byte 0x01
    .half -11
total:
    .word 0
origen:
    .asciiz "BIG ENDIAN"
pi:
    .float 3.14159

.text
.globl main
.globl __start

__start:
main:
    li $t1, 0
loop:
    lb $t2, origen($t1)
    beq $t2, $zero, fin
    addi $t1, $t1, 1
    j loop
fin:
    sb $t1, total($zero)
#Retornar al SO
add $a0, $zero, $zero
addi $v0, $zero, 4001
syscall
nop
```

Registros

```
          zero       at       v0       v1       a0       a1       a2       a3
 R0   00000000 00000000 00000000 00000000 00000000 00000000 00000000 00000000
           t0       t1       t2       t3       t4       t5       t6       t7
 R8   00000000 00000003 00000020 00000000 00000000 00000000 00000000 00000000
           s0       s1       s2       s3       s4       s5       s6       s7
 R16  00000000 0043b0b0 0043b0a0 004306b4 0043b0a0 0050b6b0 00000000 00000000
           t8       t9       k0       k1       gp       sp       s8       ra
 R24  00000000 00000000 00000000 00000000 00000000 7fa65850 7fc36618 00000000
        status       lo       hi badvaddr    cause       pc
      0000a413 00001b41 000002e7 2ab023ba 10800024 004000c4
          fcsr      fir  restart
      00000000 00739300 00000000
```

Programa Binario Decodificado

```
   0x004000b0 <+0>:   li      t1,0
   0x004000b4 <+4>:   lui     t2,0x41
   0x004000b8 <+8>:   addu    t2,t2,t1
   0x004000bc <+12>:  lb      t2,256(t2)
   0x004000c0 <+16>:  nop
   0x004000c4 <+20>:  beqz    t2,0x4000d8 <fin>
   0x004000c8 <+24>:  nop
   0x004000cc <+28>:  addi    t1,t1,1
   0x004000d0 <+32>:  j       0x4000b4 <loop>
   0x004000d4 <+36>:  nop
   0x004000d8 <+40>:  lui     at,0x41
   0x004000dc <+44>:  sb      t1,252(at)
   0x004000e0 <+48>:  add     a0,zero,zero
   0x004000e4 <+52>:  addi    v0,zero,4001
   0x004000e8 <+56>:  syscall
   0x004000ec <+60>:  nop
```

Segmento de texto

```
0x4000b0 <main>:      0x24090000     0x3c0a0041     0x01495021     0x814a0100
0x4000c0 <loop+12>:   0x00000000     0x11400004     0x00000000     0x21290001
0x4000d0 <loop+28>:   0x0810002d     0x00000000     0x3c010041     0xa02900fc
0x4000e0 <fin+8>:     0x00002020     0x20020fa1     0x0000000c     0x00000000
```

El segmento de datos del programa se carga en memoria en la dirección 0x4100F0
El segmento de código del programa se carga en memoria en la dirección 0x4000B0

1) ¿Cuales son las instrucciones máquina que se corresponden a las instrucciones ensamblador `beq $t2, $zero, fin`,`j loop` y `sb $t1, total($zero)`? ¿Cuál es su codificación hexadecimal?
2) ¿Cuál será el contenido de la dirección de memoria `total` al finalizar el programa? ¿Cuál será el contenido de los registros `t1`, `t2` y `s1`?
3) ¿Cuales son las direcciones de memoria de las etiquetas `origen` y `pi`?
4) Dado el siguiente vuelco de memoria:

```
0x4100d0:    0x0000002a    0xffff686f    0x6c61206d    0x756e646f
0x4100e0:    0x00000000    0x4048f5c3    0x00000000    0x00000000
```

   ¿Qué cadena de texto ASCII está almacenada a partir de la dirección `0x4100d6`? ¿Cuál será el resultado de ejecutar la instrucción `lh $t0, 0x4100d6`?
5) Dadas las siguientes instrucciones:

```
li $t0, 0x4100F0
lb $t1, 10($t0)
lw $t2, memoria+6
```

   ¿Qué valores tendrán los registros al finalizar la ejecución?
   ¿Cuáles son las direcciones efectivas de los accesos?

6) Diseñe circuito lógico que compute la siguiente tabla de verdad:

A	B	C	resultado
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1