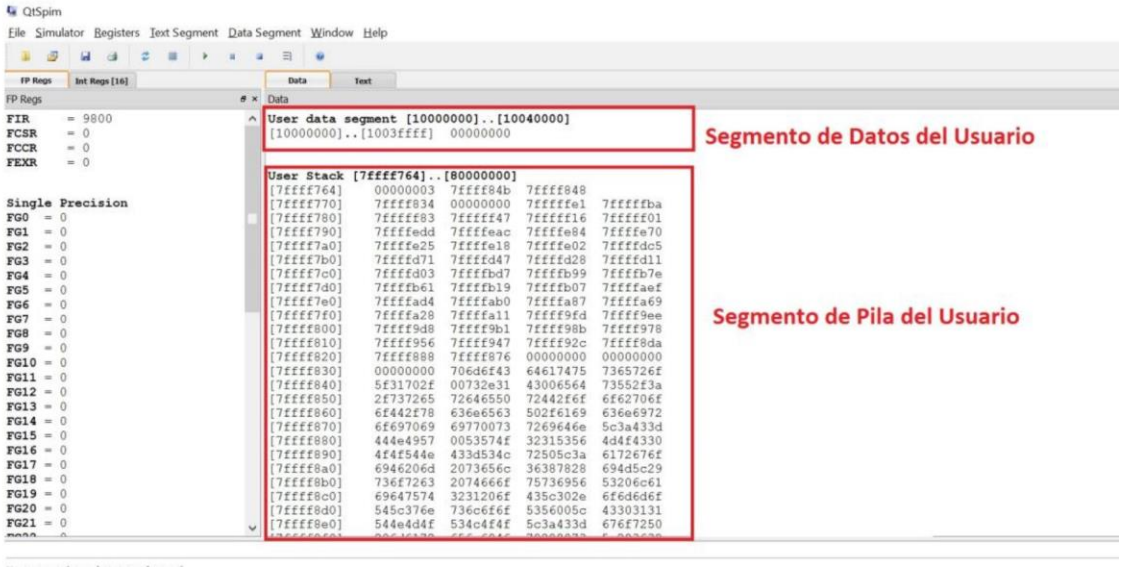
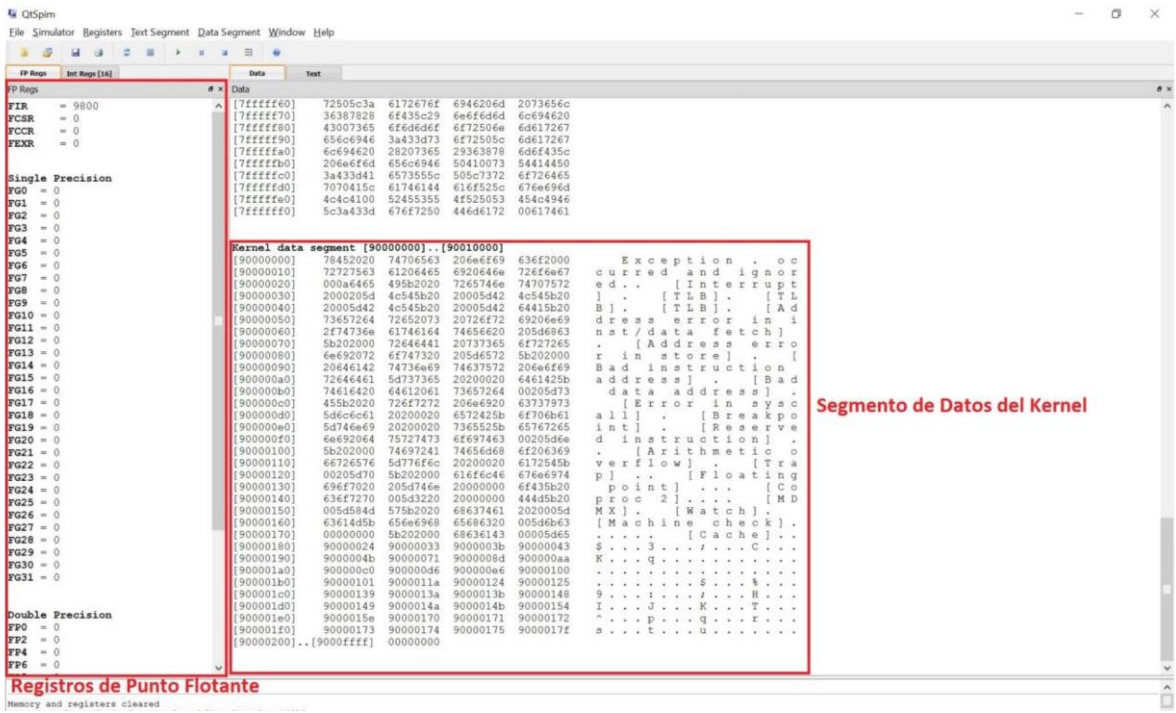
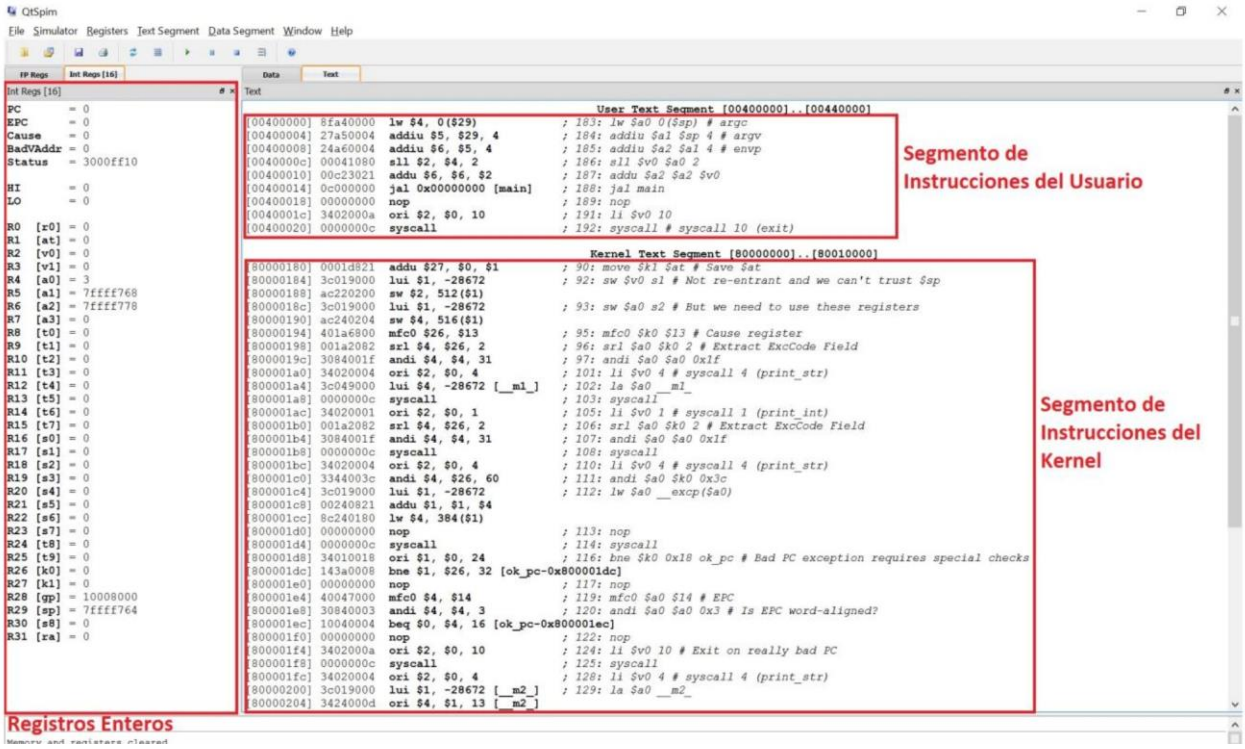
EJEMPLO DE RESOLUCIÓN DE INFORME.

a) Identificar en el emulador los siguientes elementos
b) I. El segmento de Datos (Usuario, Kernel y Pila)
c) II. El segmento de Instrucciones (Usuario y Kernel)
d) III. El contenido de los Registros Enteros
e) IV. El contenido de los Registros en Punto Flotante.
f) V. La consola del sistema





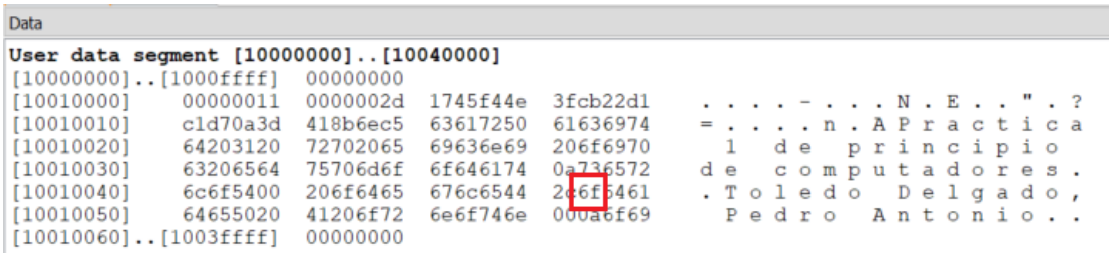

QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
PC = 0
EPC = 0
Cause = 0
BadVAddr = 0
Status = 3000ff10
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 0
R3 [v1] = 0
R4 [a0] = 3
R5 [a1] = 7ffff768
R6 [a2] = 7ffff778
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 0
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
R17 [s1] = 0
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
R28 [gp] = 10008000
R29 [sp] = 7ffff764
R30 [s8] = 0
R31 [ra] = 0
Text
User Text Segment [00400000]..[00440000]
[00400000] 8fa40000 lw $4, 0($29) ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004 addiu $5, $29, 4 ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004 addiu $6, $5, 4 ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080 sll $2, $4, 2 ; 186: sll $v0 $a0 2
[00400010] 00c23021 addu $6, $6, $2 ; 187: addu $a2 $a2 $v0
[00400014] 0c000000 jal 0x00000000 [main] ; 188: jal main
[00400018] 00000000 nop ; 189: nop
[0040001c] 3402000a ori $2, $0, 10 ; 191: li $v0 10
[00400020] 0000000c syscall ; 192: syscall # syscall 10 (exit)
Segmento de Instrucciones del Usuario
Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821 addu $27, $0, $1 ; 90: move $k1 $at # Save $at
[80000184] 3c019000 lui $1, -28672 ; 92: sw $v0 s1 # Not re-entrant and we can't trust $sp
[80000188] ac220200 sw $2, 512($1)
[8000018c] 3c019000 lui $1, -28672 ; 93: sw $a0 s2 # But we need to use these registers
[80000190] ac240204 sw $4, 516($1)
[80000194] 401a6800 mfc0 $26, $13 ; 95: mfc0 $k0 $13 # Cause register
[80000198] 001a2082 srl $4, $26, 2 ; 96: srl $a0 $k0 2 # Extract ExcCode Field
[8000019c] 3084001f andi $4, $4, 31 ; 97: andi $a0 $a0 0x1f
[800001a0] 34020004 ori $2, $0, 4 ; 101: li $v0 4 # syscall 4 (print_str)
[800001a4] 3c049000 lui $4, -28672 [__m1_] ; 102: la $a0 __m1_
[800001a8] 0000000c syscall ; 103: syscall
[800001ac] 34020001 ori $2, $0, 1 ; 105: li $v0 1 # syscall 1 (print_int)
[800001b0] 001a2082 srl $4, $26, 2 ; 106: srl $a0 $k0 2 # Extract ExcCode Field
[800001b4] 3084001f andi $4, $4, 31 ; 107: andi $a0 $a0 0x1f
[800001b8] 0000000c syscall ; 108: syscall
[800001bc] 34020004 ori $2, $0, 4 ; 110: li $v0 4 # syscall 4 (print_str)
[800001c0] 3344003c andi $4, $26, 60 ; 111: andi $a0 $k0 0x3c
[800001c4] 3c019000 lui $1, -28672 ; 112: lw $a0 __excp($a0)
[800001c8] 00240821 addu $1, $1, $4
[800001cc] 8c240180 lw $4, 384($1)
[800001d0] 00000000 nop ; 113: nop
[800001d4] 0000000c syscall ; 114: syscall
[800001d8] 34010018 ori $1, $0, 24 ; 116: bne $k0 0x18 ok_pc # Bad PC exception requires special checks
[800001dc] 143a0008 bne $1, $26, 32 [ok_pc-0x800001dc]
[800001e0] 00000000 nop ; 117: nop
[800001e4] 40047000 mfc0 $4, $14 ; 119: mfc0 $a0 $14 # EPC
[800001e8] 30840003 andi $4, $4, 3 ; 120: andi $a0 $a0 0x3 # Is EPC word-aligned?
[800001ec] 10040004 beq $0, $4, 16 [ok_pc-0x800001ec]
[800001f0] 00000000 nop ; 122: nop
[800001f4] 3402000a ori $2, $0, 10 ; 124: li $v0 10 # Exit on really bad PC
[800001f8] 0000000c syscall ; 125: syscall
[800001fc] 34020004 ori $2, $0, 4 ; 128: li $v0 4 # syscall 4 (print_str)
[80000200] 3c019000 lui $1, -28672 [__m2_] ; 129: la $a0 __m2_
[80000204] 3424000d ori $4, $1, 13 [__m2_]
Segmento de Instrucciones del Kernel
Registros Enteros

b) Edita con un editor de textos plano (vi, vim, gedit, kate o el que prefieras) el fichero practica1.s y sustituye la cadena “"nombre apellido1 apellido2\n" con tu nombre y tus apellidos y graba el fichero. A continuación carga el programa en QtSpim y ejecuta el programa de una sola vez. Saca un pantallazo de la consola y marca mediante un cuadro rojo la impresión de tu nombre.

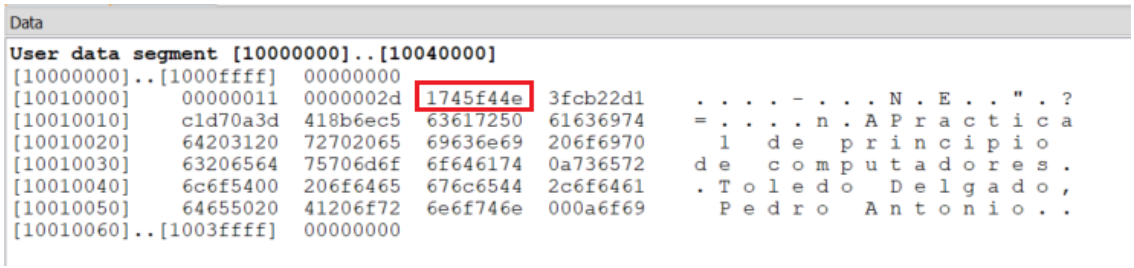
c)
I. ¿Qué dirección de memoria (expresa la dirección en hexadecimal) ocupa el último carácter de tu nombre (p.ej: en “Simpson, Homer” el carácter en cuestión es “n”)?

En este caso estoy buscando la dirección del carácter ‘o’ que es [1001004E] Que corresponde con el carácter hexadecimal 0x6F que en la tabla ascii es ‘o’

II ¿Qué carácter es y qué representación tiene en hexadecimal? Saca un pantallazo de User data Segment y marca con un cuadro rojo el byte correspondiente a ese carácter.

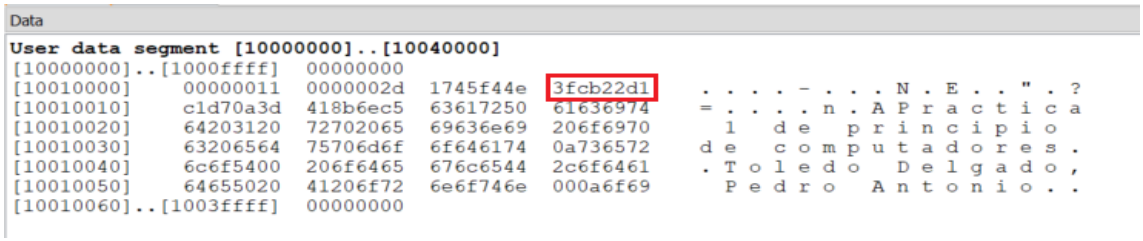


III. Recuerda que estás en hexadecimal. Busca en el segmento de datos de qtspim el número que se encuentra en la dirección etiquetada como num3. Saca un pantallazo y marca con un recuadro en rojo la palabra correspondiente.



IV. Convierte el número 1.578 a formato IEE-754 para 32 bits (usa los apuntes del profesor o utiliza una calculadora online). Busca ahora este número en el segmento de datos, saca un pantallazo y márcalo con un cuadro en rojo.

1.578 en IEEE754 32 bits es 0x3FCB22D1

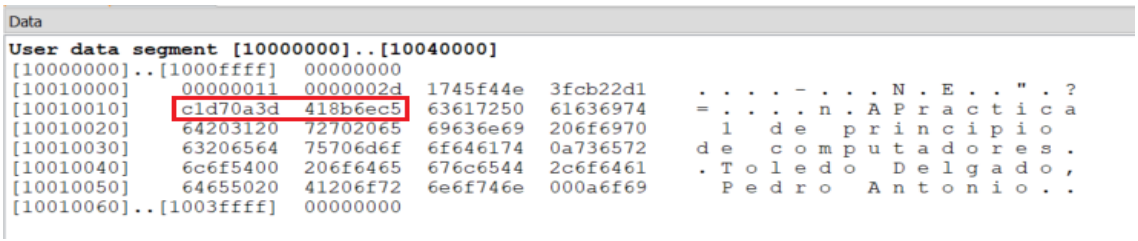


V. ¿En qué dirección empieza el número 1.578? expresa la dirección en hexadecimal.

[1001000C] (empieza en la dirección 1001000C y su último byte está en la 1001000F)

VI. Convierte el número 57530552.23 a formato IEE-754 para 64 bits (usa los apuntes del profesor o utiliza una calculadora online). Busca ahora este número en el segmento de datos, saca un pantallazo y márcalo con un cuadro en rojo.

57530552.23 en IEEE754 64bits es 0x418B6EC5C1D70A3D



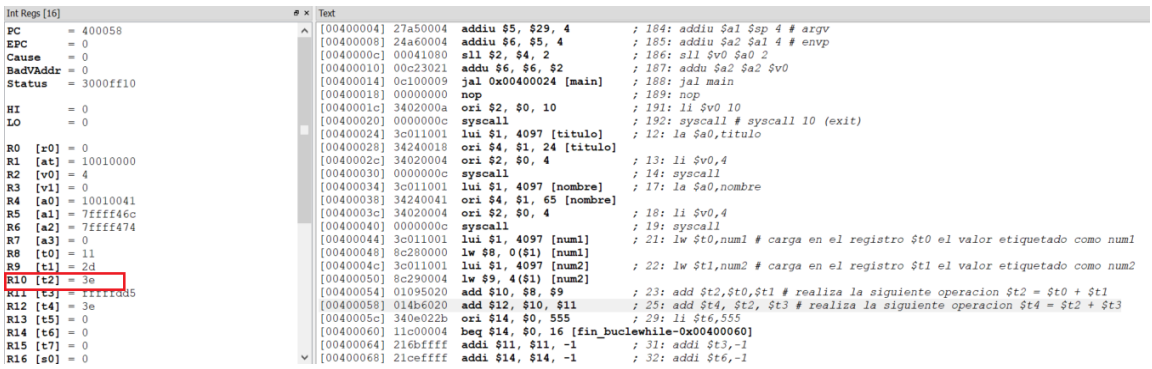
VII. ¿En qué dirección empieza el número 53125322.45? expresa la dirección en hexadecimal.

[10010010]] (empieza en la dirección 10010010 y su último byte está en la 10010017)

d) Ejecución del programa:

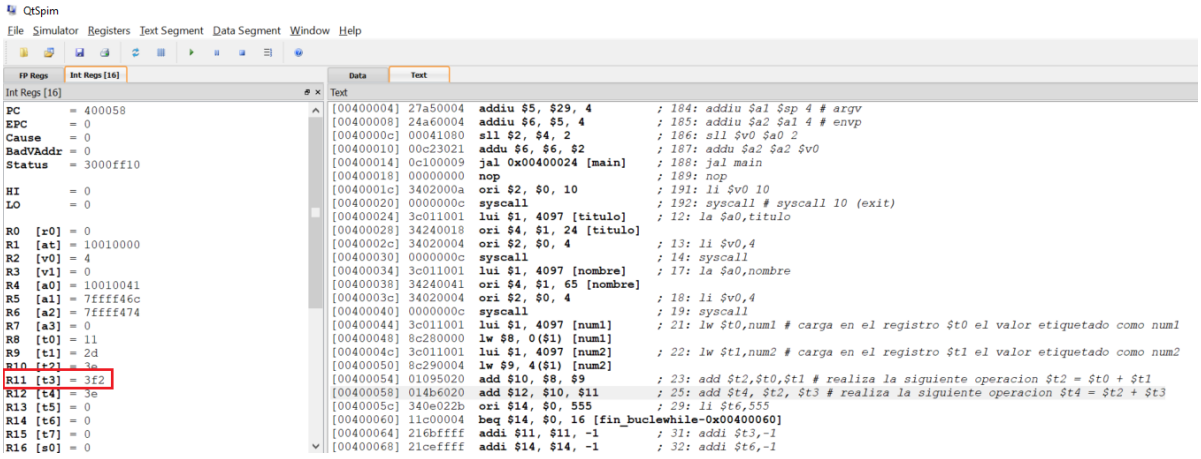
I. Ejecuta paso a paso el programa hasta que hayas encontrado la instrucción add $t2,$t0,$t1 Una vez se haya ejecutado saca un pantallazo del banco de registros enteros y pon un cuadro rojo sobre el registro $t2. ¿Qué valor contiene? ¿sabrías expresarlo en decimal?

Contiene el valor 0x3E que es 16x3+ 14 = **62**



II. Cuando hayas terminado de ejecutar esta instrucción, modifica a mano el valor del registro $t3 (pulsa con el botón derecho del ratón sobre el registro correspondiente en el banco de registro y selecciona “Change Register Contents”, allí puedes seleccionar el formato y el valor). Deberás introducir un valor 1010 en formato decimal. Una vez lo hayas hecho saca un pantallazo y marca con un cuadro en rojo el registro correspondiente.

Se queda con el valor 0x3f2 que es 1010 en decimal



III. A continuación sigue ejecutando paso a paso hasta terminar de ejecutar la instrucción add $t4,$t2,$t3. ¿Qué valor tiene el registro $t4 en hexadecimal? ¿y en decimal?

En hexadecimal tiene el valor 0x430

En decimal tiene el valor 1072
(se puede comprobar cambiando el modo de visualización en el menú Registers)

IV. A continuación establece un punto de ruptura “breakpoint” sobre la instrucción move $a0,$t3 (sobre la instrucción correspondiente, pulsa en el botón derecho del ratón y selecciona “Set Breakpoint”. Después ejecuta todo el código (no paso a paso) y observarás que la ejecución se para en esta instrucción saltándose el bucle que hemos puesto. En este punto. ¿Qué valor tiene $t3 (expresado en hexadecimal y también en decimal? ¿y qué valor tiene $t6?
$t3 en hexadecimal tiene un 0x1c7 y en decimal 455
$t6 tiene un cero (que es igual en los dos formatos)