PRINCIPIOS DE COMPUTADORES

Práctica 1. Utilización del emulador QTSpim y representación de la información

Daniel Dóniz García

01/03/2020

a) Identificar en el emulador los siguientes elementos. (Prepara en el informe el número de capturas de pantalla que necesites para señalar mediante un cuadro rojo cada una de ellas):

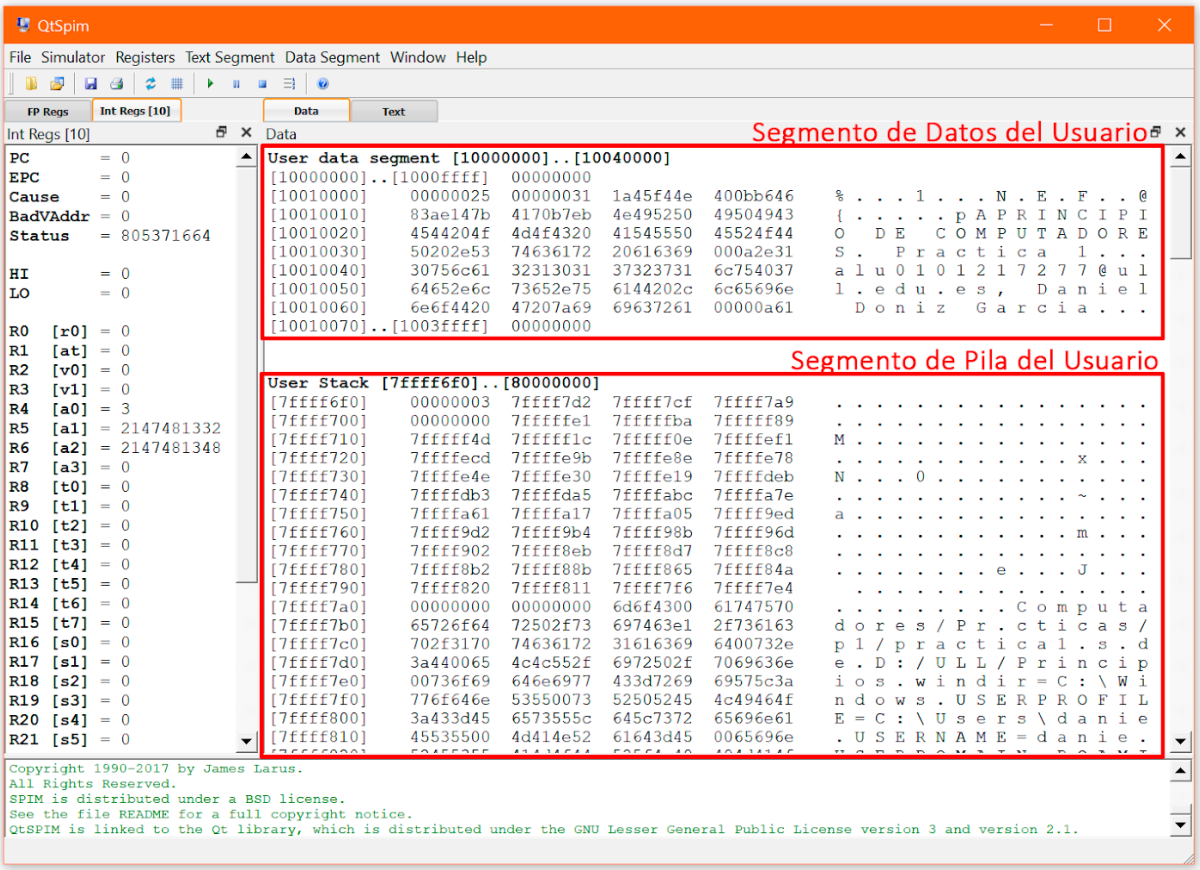
I. El segmento de Datos (Usuario, Kernel y Pila)

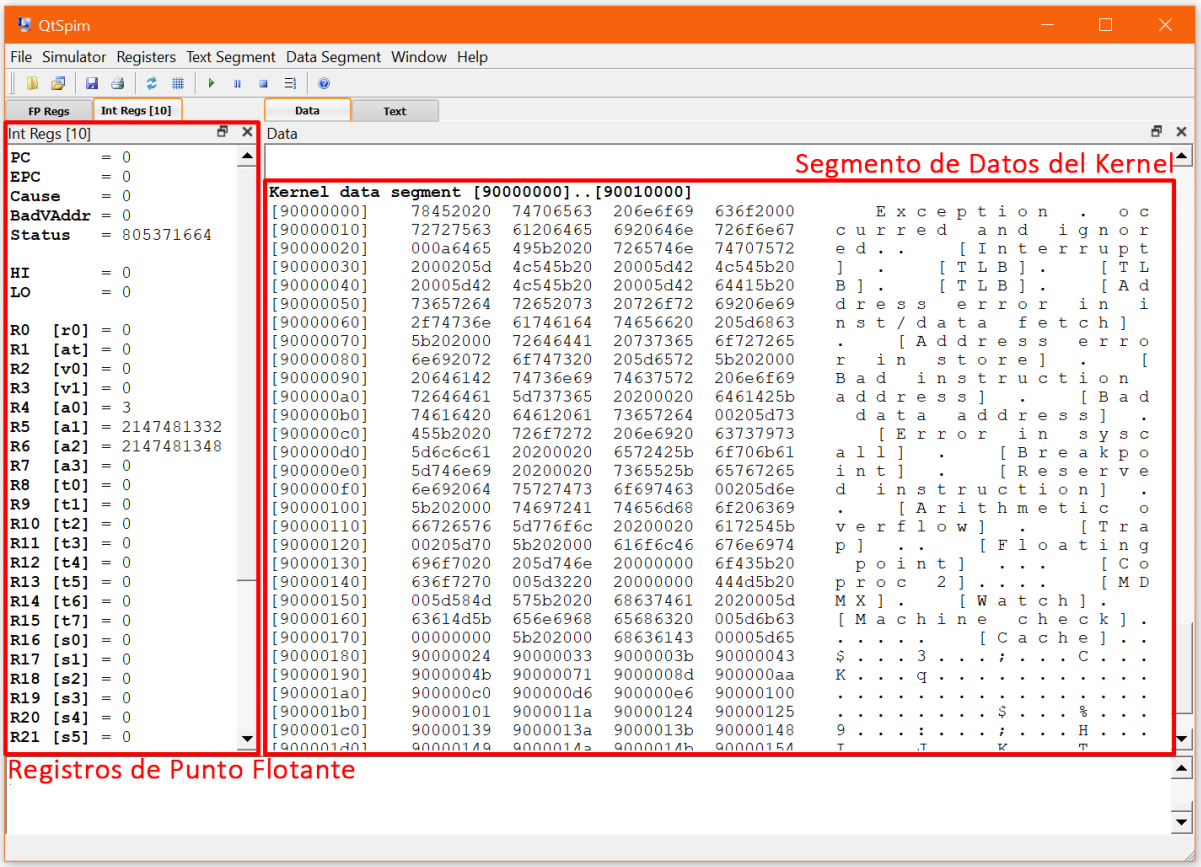
II. El segmento de Instrucciones (Usuario y Kernel)

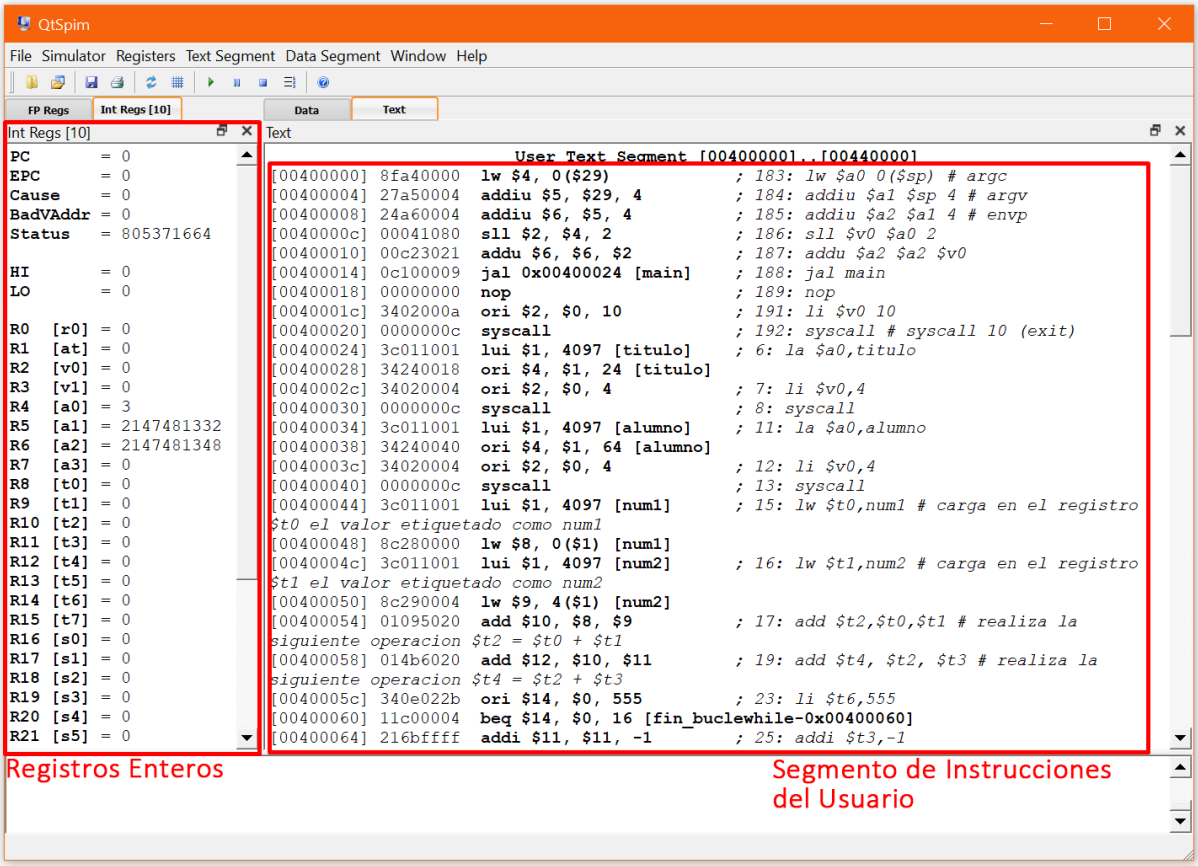
III. El contenido de los Registros Enteros

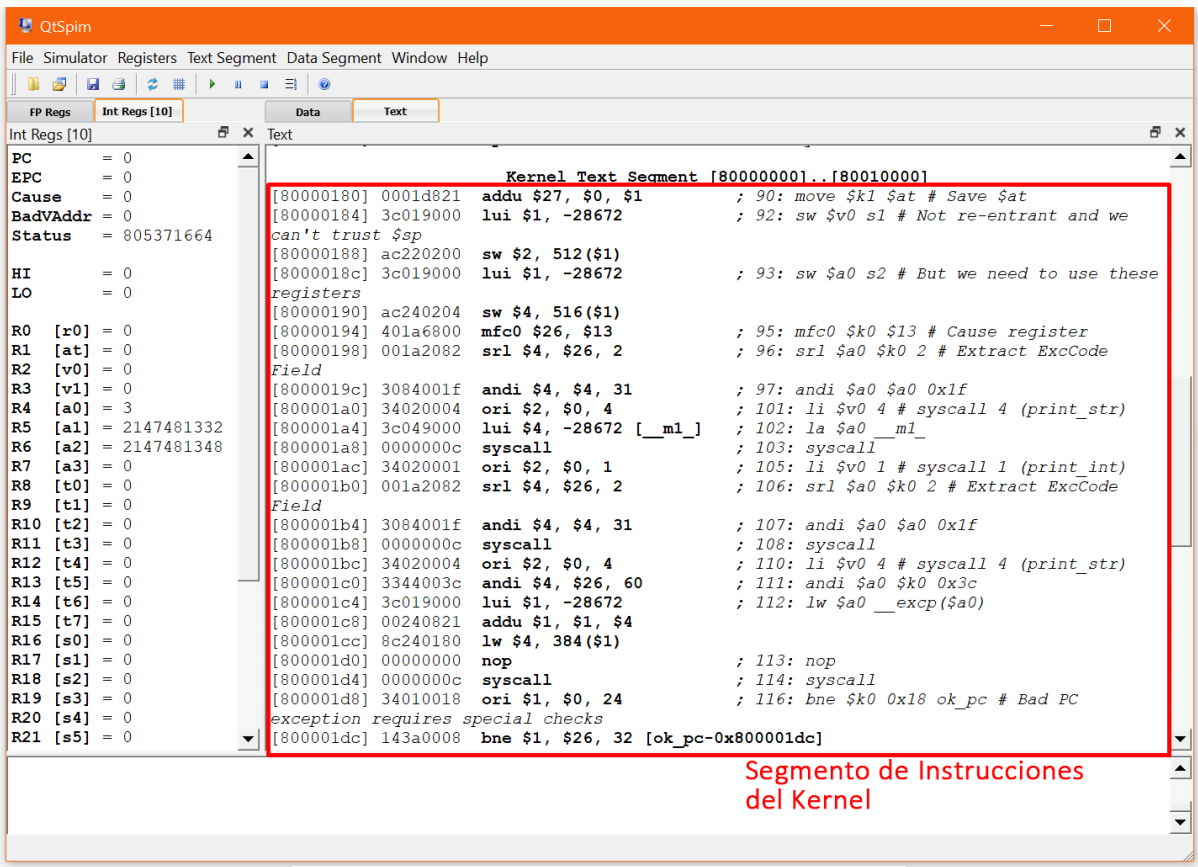
IV. El contenido de los Registros en Punto Flotante.

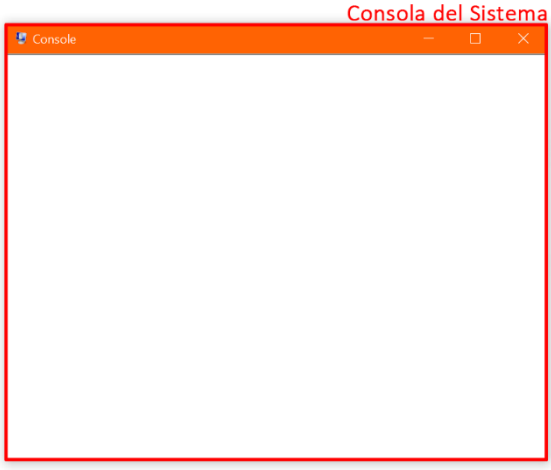
V. La consola del sistema



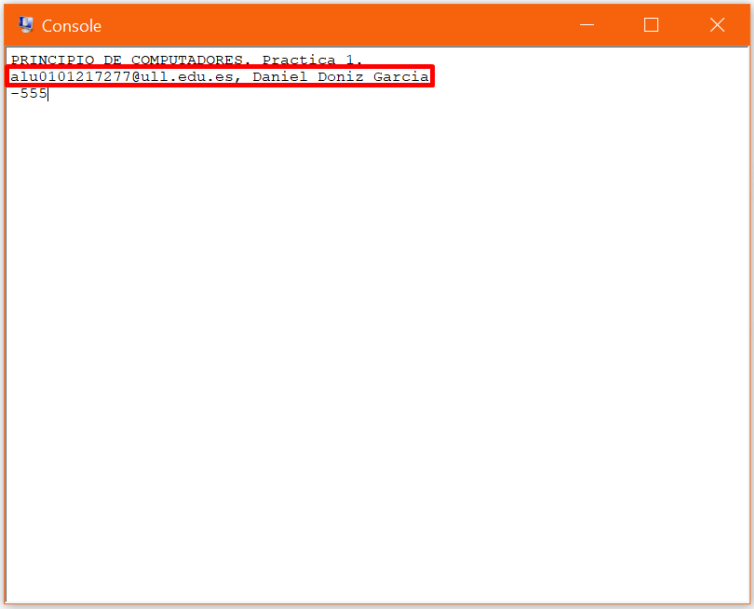



QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [10]
Data
Text
Int Regs [10]
PC = 0
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 0
R3 [v1] = 0
R4 [a0] = 3
R5 [a1] = 2147481332
R6 [a2] = 2147481348
R7 [a3] = 0
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
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] 0c100009 jal 0x00400024 [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)
[00400024] 3c011001 lui $1, 4097 [titulo] ; 6: la $a0,titulo
[00400028] 34240018 ori $4, $1, 24 [titulo]
[0040002c] 34020004 ori $2, $0, 4 ; 7: li $v0,4
[00400030] 0000000c syscall ; 8: syscall
[00400034] 3c011001 lui $1, 4097 [alumno] ; 11: la $a0,alumno
[00400038] 34240040 ori $4, $1, 64 [alumno]
[0040003c] 34020004 ori $2, $0, 4 ; 12: li $v0,4
[00400040] 0000000c syscall ; 13: syscall
[00400044] 3c011001 lui $1, 4097 [num1] ; 15: lw $t0,num1 # carga en el registro $t0 el valor etiquetado como num1
[00400048] 8c280000 lw $8, 0($1) [num1]
[0040004c] 3c011001 lui $1, 4097 [num2] ; 16: lw $t1,num2 # carga en el registro $t1 el valor etiquetado como num2
[00400050] 8c290004 lw $9, 4($1) [num2]
[00400054] 01095020 add $10, $8, $9 ; 17: add $t2,$t0,$t1 # realiza la siguiente operacion $t2 = $t0 + $t1
[00400058] 014b6020 add $12, $10, $11 ; 19: add $t4, $t2, $t3 # realiza la siguiente operacion $t4 = $t2 + $t3
[0040005c] 340e022b ori $14, $0, 555 ; 23: li $t6,555
[00400060] 11c00004 beq $14, $0, 16 [fin_buclewhile-0x00400060]
[00400064] 216bffff addi $11, $11, -1 ; 25: addi $t3,-1
Registros Enteros
Segmento de Instrucciones del Usuario


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [10]
Data
Text
Int Regs [10]
PC = 0
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 0
R3 [v1] = 0
R4 [a0] = 3
R5 [a1] = 2147481332
R6 [a2] = 2147481348
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
Text
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]
Segmento de Instrucciones del Kernel

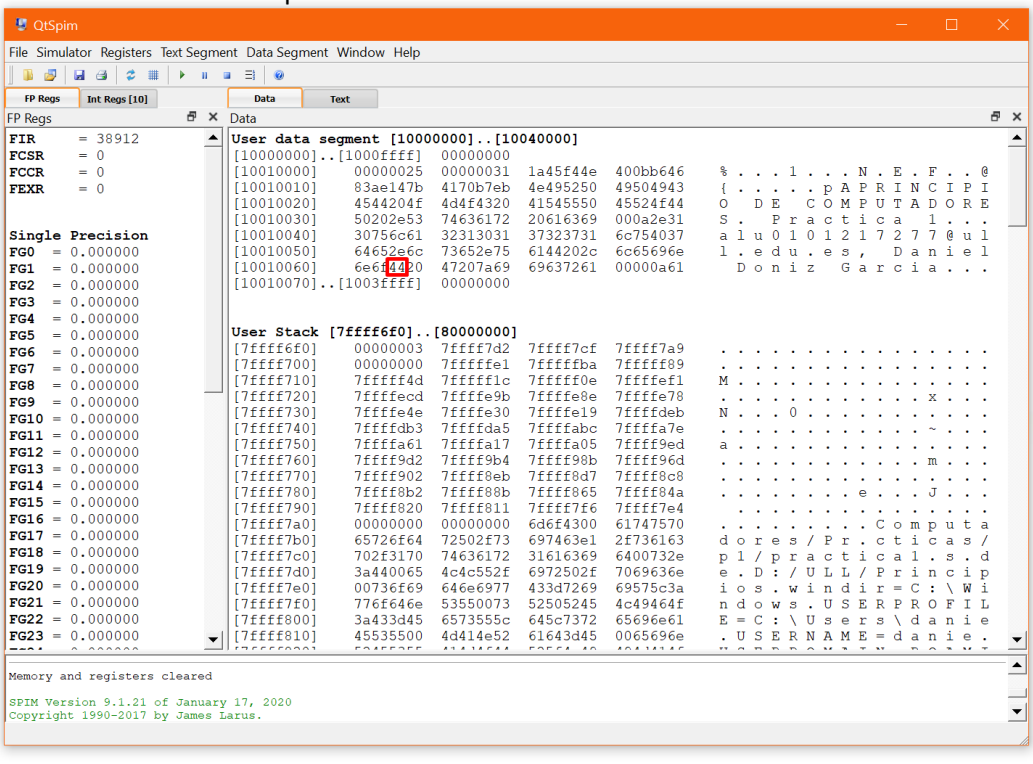

Consola del Sistema
Console

b) Edita con un editor de textos plano (vi, vim, gedit, kate o el que prefieras) el fichero practica1.s y sustituye la cadena "alu9999999999@ull.edu.es, nombre apellido1 apellido2\n" con tu dirección de corrreo, 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 tus datos.

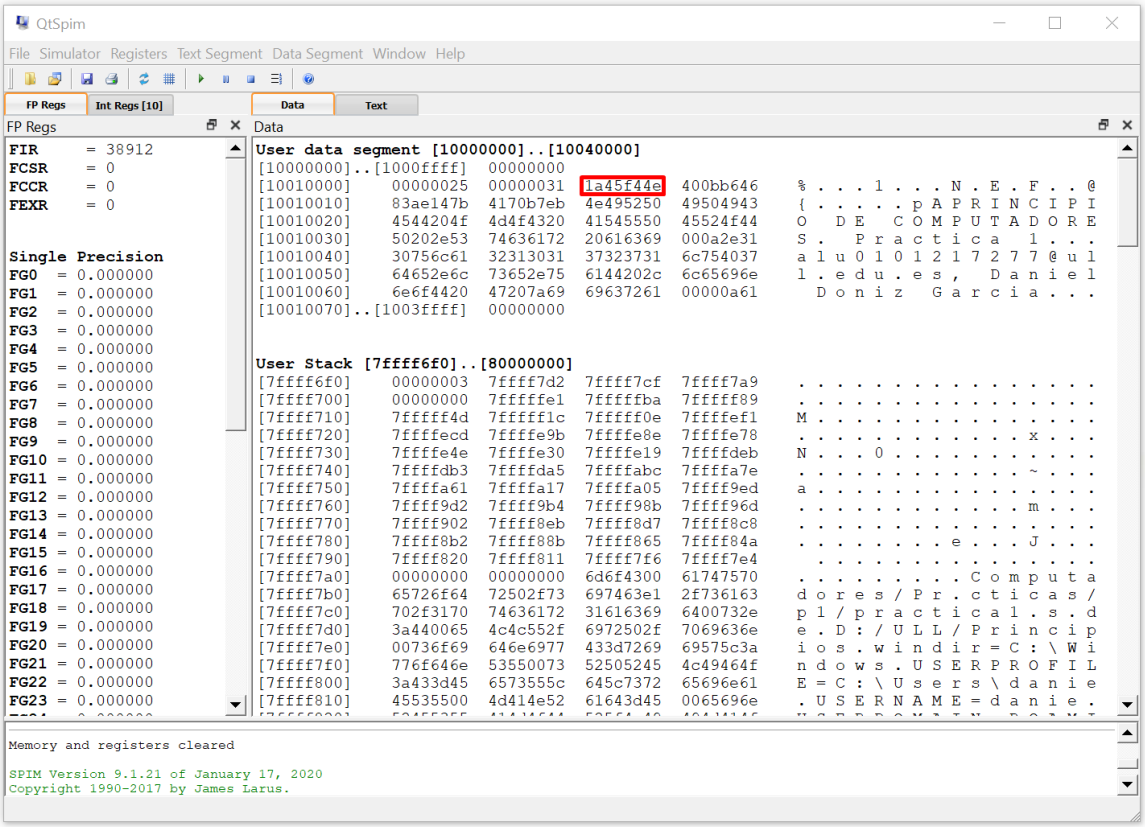


c) Explora el segmento de datos. (NOTA: Al sacar los pantallazos debes incluir las direcciones y los valores del User Data Segment completo, ya que los valores y las direcciones dependerán del nombre de cada alumno y serán necesarios para poder realizar la corrección).
Comprueba que el segmento de datos está representado en Hexadecimal (en el menú Data Segment marca el checkbox correspondiente a Hex). Con las indicaciones que te de tu profesor en la práctica responde a las siguientes preguntas:
I. ¿Qué dirección de memoria (expresa la dirección en hexadecimal) ocupa el primer carácter de tu primer apellido (p.ej: en "alu9999999999@ull.edu.es, Carlos Martin Galan\n" el carácter en cuestión es “M”)?
Está en la dirección: [10010061]

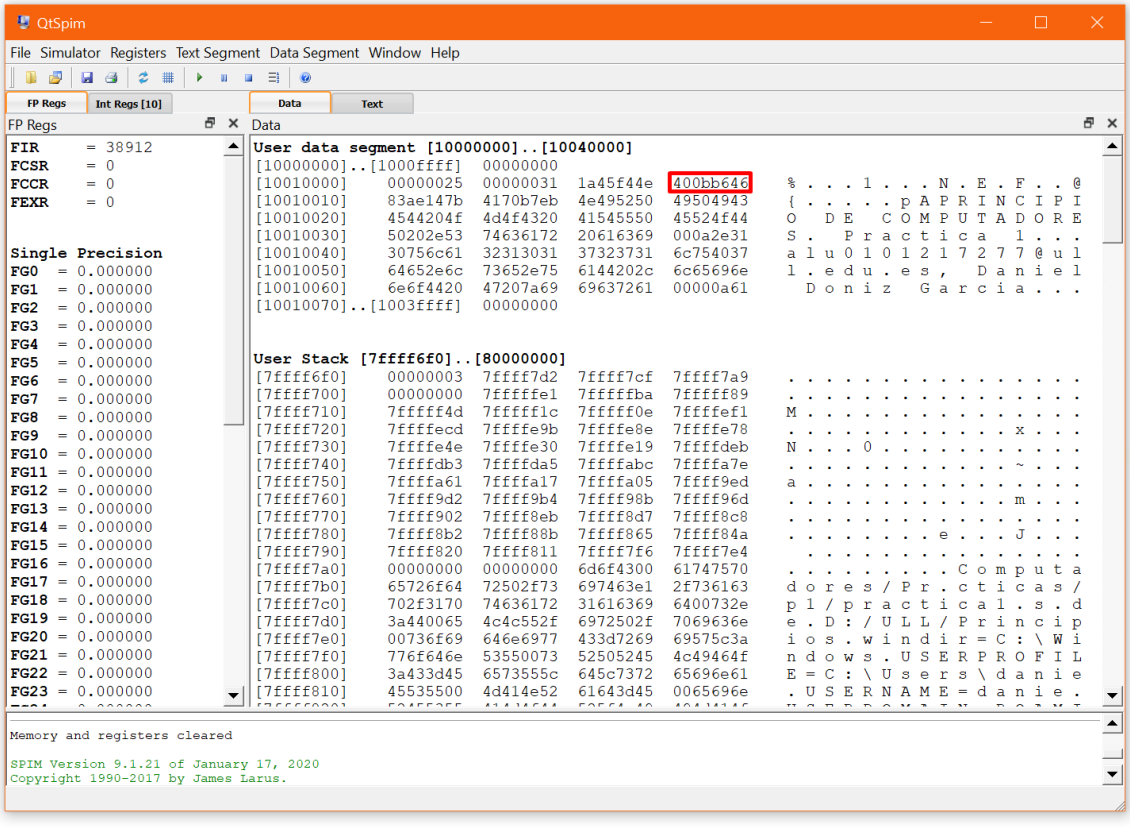
II. ¿Qué carácter es y qué representación tiene en hexadecimal? Saca un pantallazo de User data Segment y marca con un cuadro rojo el byte correspondiente a ese carácter.
Mi letra es “D” la cual corresponde con 0x44 en hexadecimal en la tabla ascii.



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.

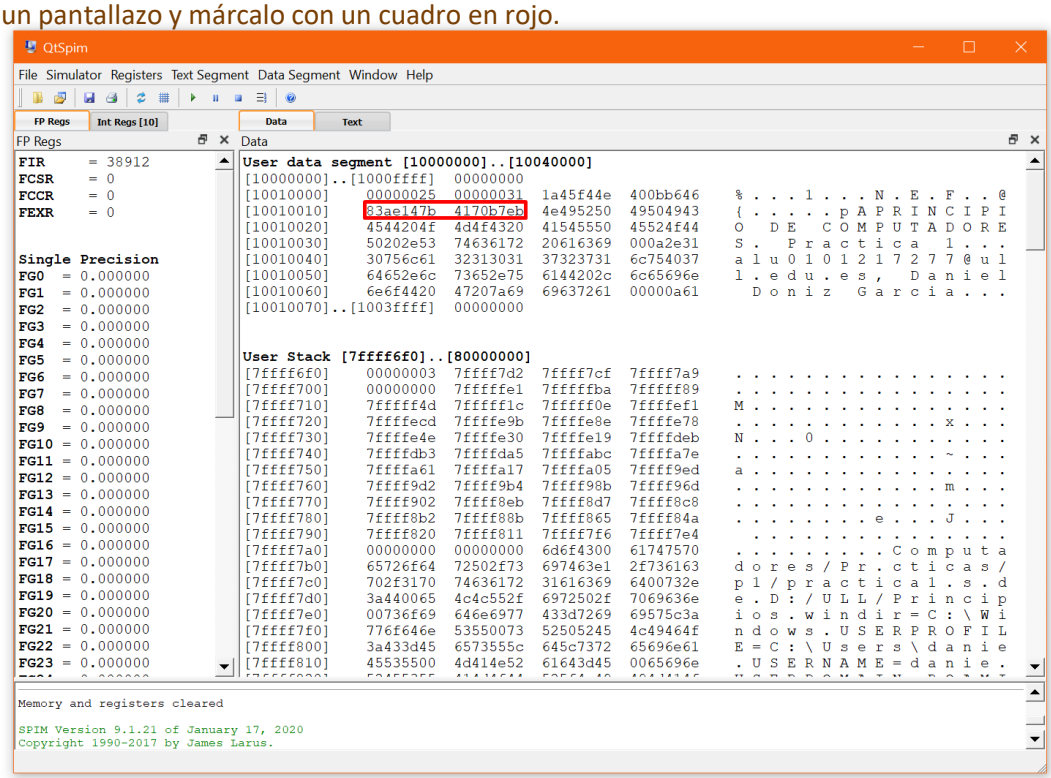


IV. Convierte el número 2,183 a formato IEE-754 para 32 bits (usa los apuntes del profesor o utiliza una calculadora online). Busca ahora este número en el segmento de datos, saca un pantallazo y márcalo con un cuadro en rojo.



V. ¿En qué dirección empieza el número 2,183? expresa la dirección en hexadecimal.
El numero comienza en la dirección [1001000C]

VI. Convierte el número 17530552.23 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.

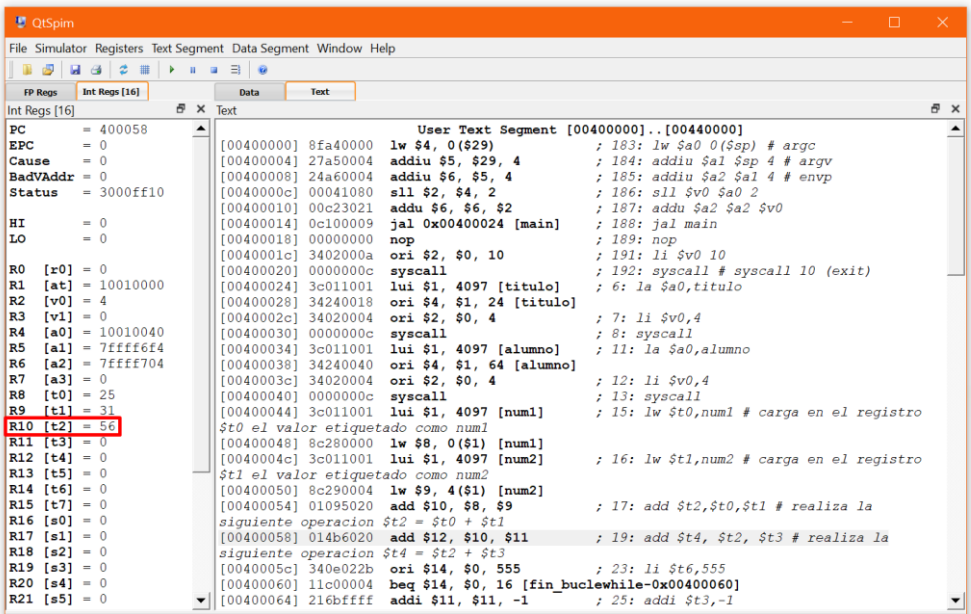


VII. ¿En qué dirección empieza el número 17530552.23? expresa la dirección en hexadecimal.
El numero comienza en la dirección [10010010]

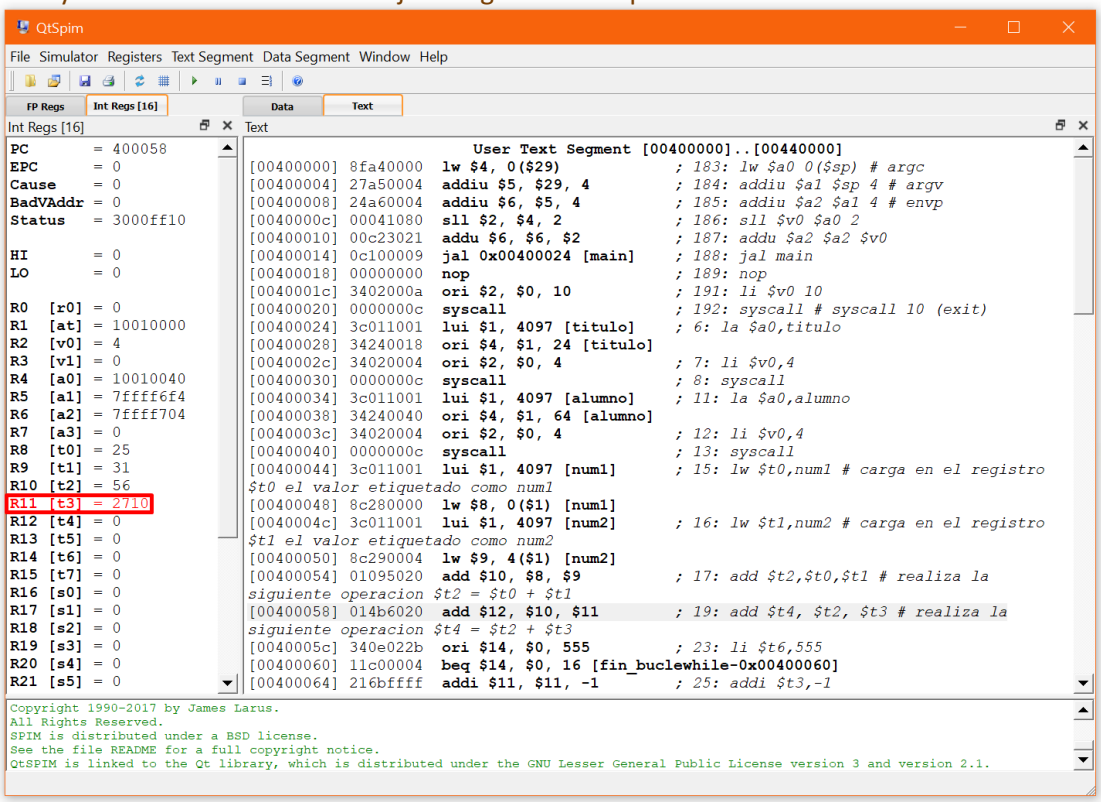
d) Reinicia la máquina y vuelve a cargar el programa en el QtSpim. Visualiza el banco de registros enteros y flotantes en hexadecimal. Menú Registers opción Hex). Recuerda, que las instrucciones de tu programa pueden ser convertidas en una o más instrucciones en QtSpim, por lo que tendrás que buscar la instrucción original de tu programa en los comentarios de la parte derecha.

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 0x56 = 16^1x5 + 16^0x6 = $(86)_{10}$



II. Cuando hayas terminado de ejectuar 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 10000 en formato decimal. Una vez lo hayas hecho saca un pantallazo y marca con un cuadro en rojo el registro correspondiente.



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?
Contiene el valor 0x2766 = $16^3x2 + 16^2x7 + 16^1x6 + 16^0x6 = (10086)_{10}$

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?