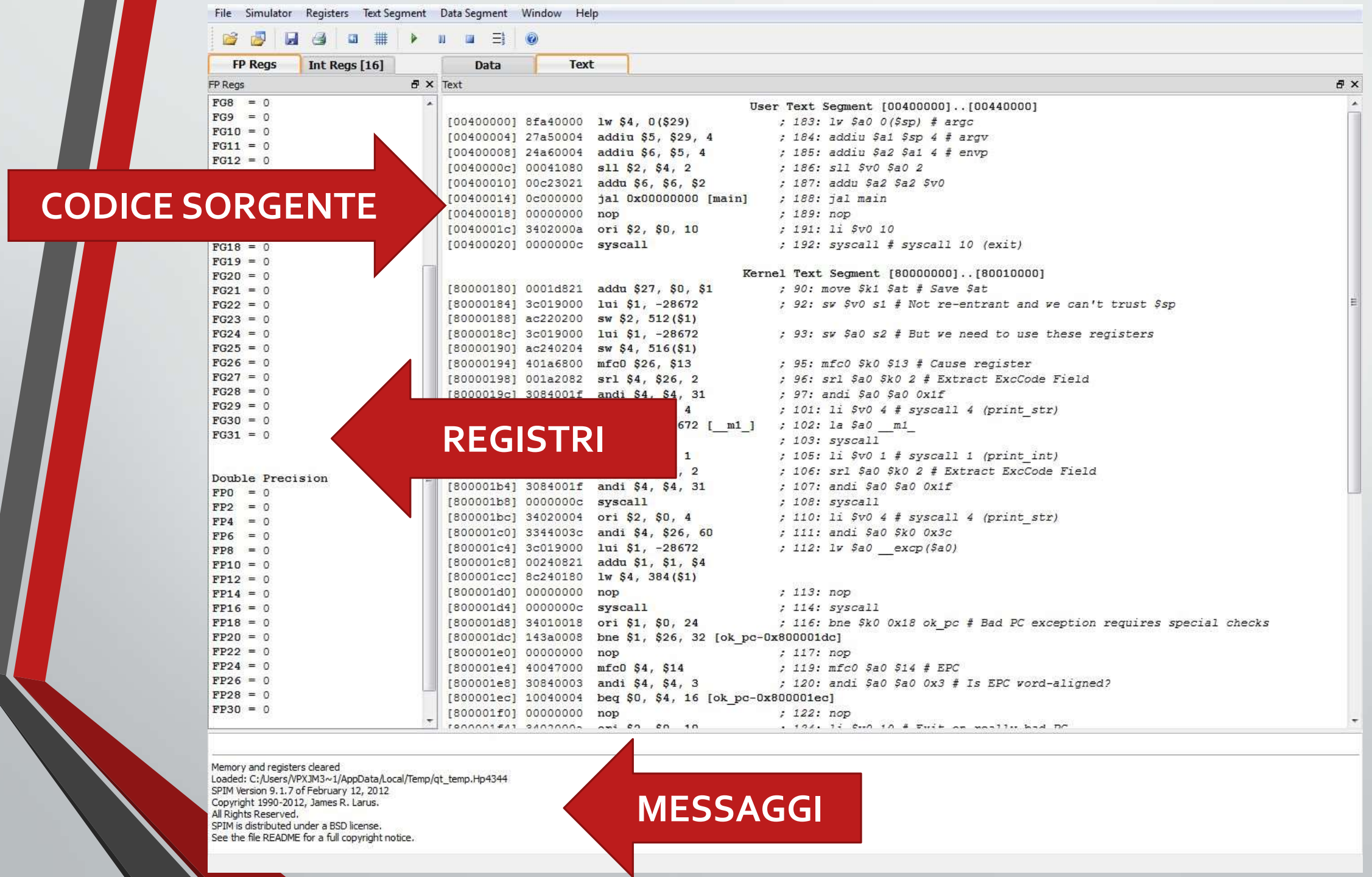
QtSpim

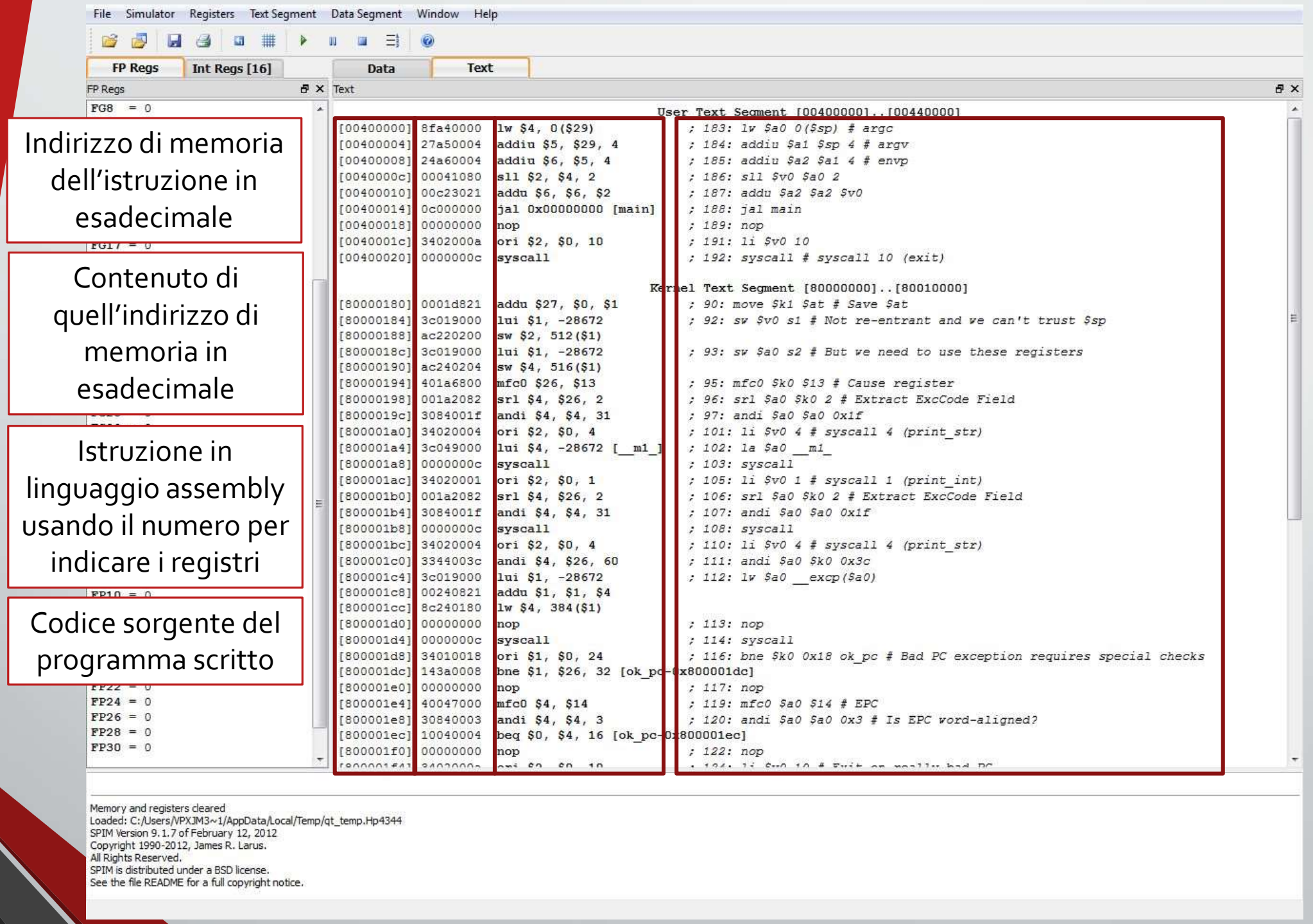
Andrea Gasparetto

Materiale del tutorato

Slide e sorgenti

http://www.dsi.unive.it/~gasparetto/teaching.htm


File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
FP Regs
FG8 = 0
FG9 = 0
FG10 = 0
FG11 = 0
FG12 = 0
FG18 = 0
FG19 = 0
FG20 = 0
FG21 = 0
FG22 = 0
FG23 = 0
FG24 = 0
FG25 = 0
FG26 = 0
FG27 = 0
FG28 = 0
FG29 = 0
FG30 = 0
FG31 = 0
Double Precision
FP0 = 0
FP2 = 0
FP4 = 0
FP6 = 0
FP8 = 0
FP10 = 0
FP12 = 0
FP14 = 0
FP16 = 0
FP18 = 0
FP20 = 0
FP22 = 0
FP24 = 0
FP26 = 0
FP28 = 0
FP30 = 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)
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
4 ; 101: li $v0 4 # syscall 4 (print_str)
672 [__m1_] ; 102: la $a0 __m1_
; 103: syscall
1 ; 105: li $v0 1 # syscall 1 (print_int)
, 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
Memory and registers cleared
Loaded: C:/Users/VPXJM3~1/AppData/Local/Temp/qt_temp.Hp4344
SPIM Version 9.1.7 of February 12, 2012
Copyright 1990-2012, James R. Larus.
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.
CODICE SORGENTE
REGISTRI
MESSAGGI

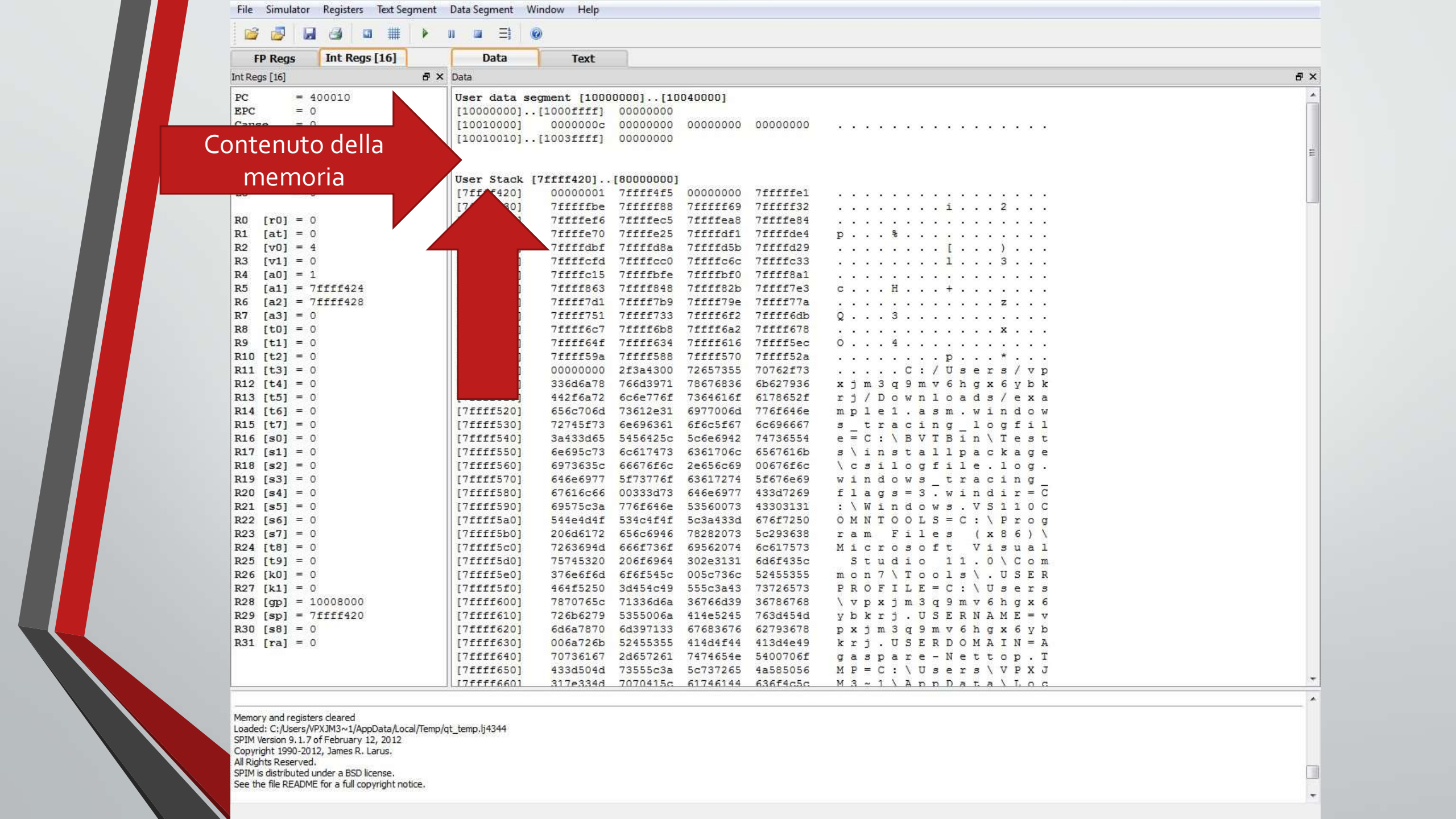


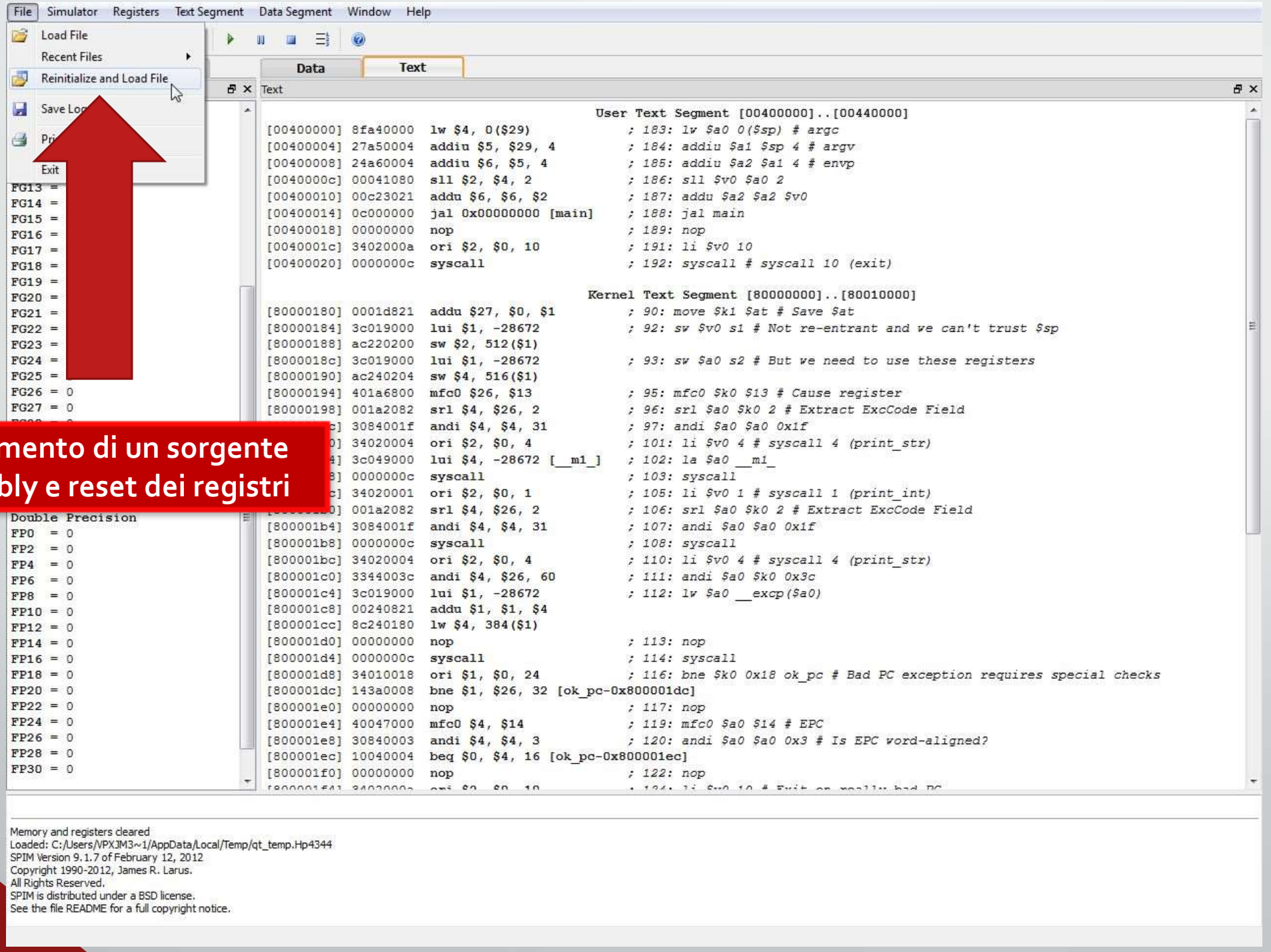
Indirizzo di memoria dell'istruzione in esadecimale

Contenuto di quell'indirizzo di memoria in esadecimale

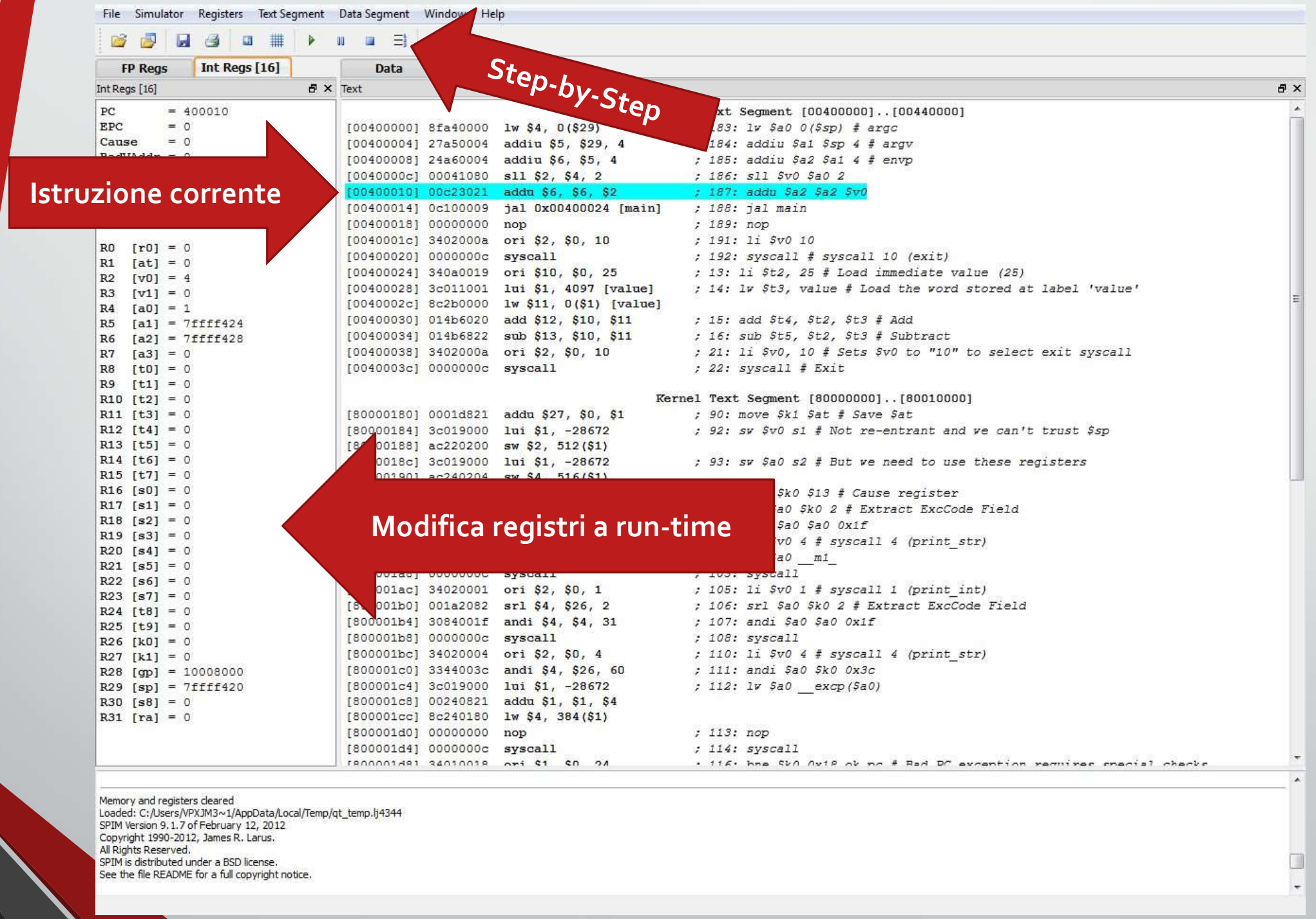
Istruzione in linguaggio assembly usando il numero per indicare i registri

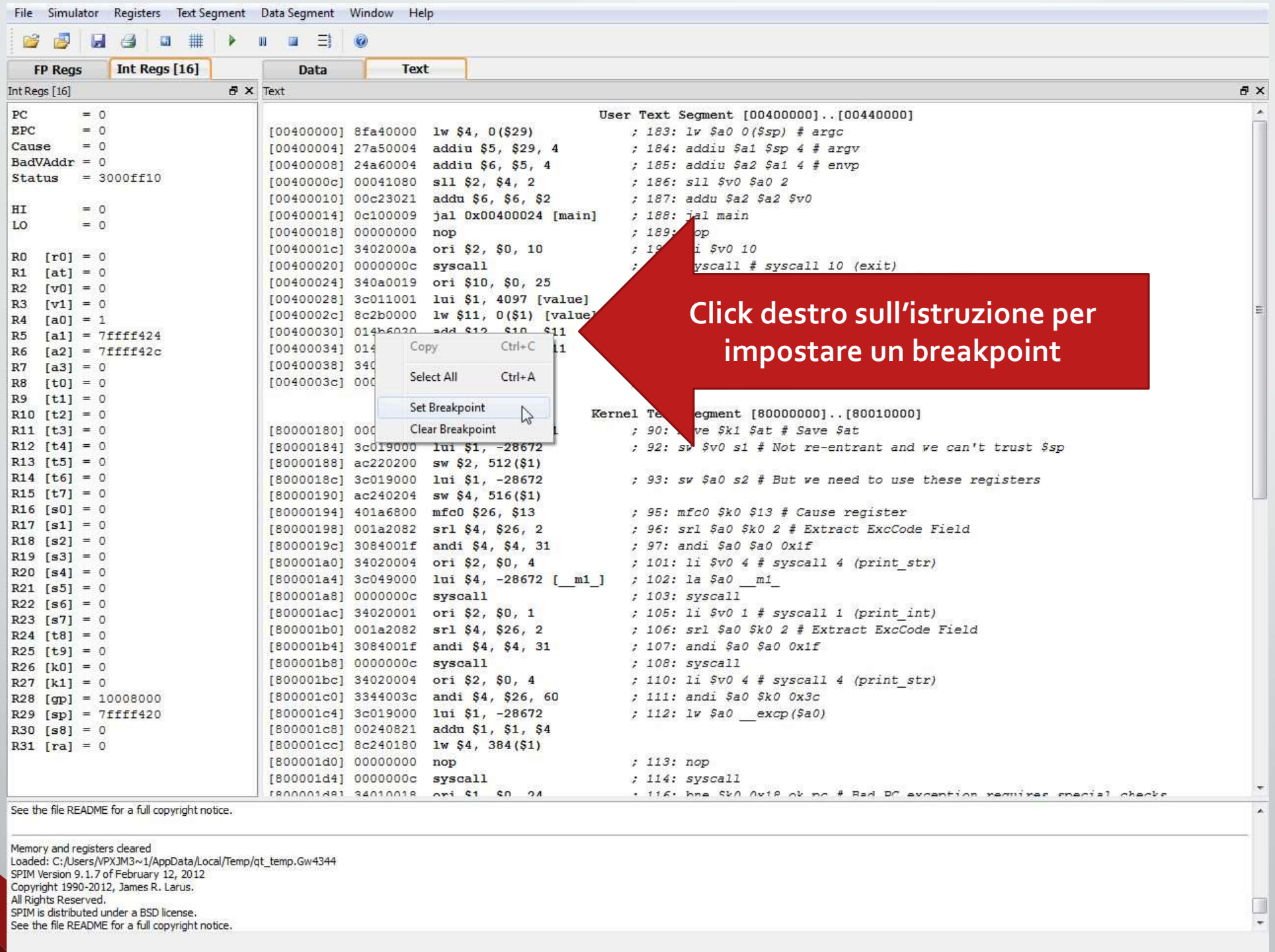
Codice sorgente del programma scritto


File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Text
Int Regs [16]
PC = 400010
EPC = 0
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 4
R3 [v1] = 0
R4 [a0] = 1
R5 [a1] = 7ffff424
R6 [a2] = 7ffff428
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] = 7ffff420
R30 [s8] = 0
R31 [ra] = 0
Data
User data segment [10000000]..[10040000]
[10000000]..[1000ffff] 00000000
[10010000] 0000000c 00000000 00000000 00000000
[10010010]..[1003ffff] 00000000
User Stack [7ffff420]..[80000000]
[7ffff420] 00000001 7ffff4f5 00000000 7fffffe1
[7ffff430] 7fffffbe 7fffff88 7fffff69 7fffff32
7ffffef6 7ffffec5 7ffffea8 7ffffe84
7ffffe70 7ffffe25 7ffffdf1 7ffffde4
7ffffdbf 7ffffd8a 7ffffd5b 7ffffd29
7ffffcfd 7ffffcc0 7ffffc6c 7ffffc33
7ffffc15 7ffffbfe 7ffffbf0 7ffff8a1
7ffff863 7ffff848 7ffff82b 7ffff7e3
7ffff7d1 7ffff7b9 7ffff79e 7ffff77a
7ffff751 7ffff733 7ffff6f2 7ffff6db
7ffff6c7 7ffff6b8 7ffff6a2 7ffff678
7ffff64f 7ffff634 7ffff616 7ffff5ec
7ffff59a 7ffff588 7ffff570 7ffff52a
00000000 2f3a4300 72657355 70762f73
336d6a78 766d3971 78676836 6b627936
442f6a72 6c6e776f 7364616f 6178652f
[7ffff520] 656c706d 73612e31 6977006d 776f646e
[7ffff530] 72745f73 6e696361 6f6c5f67 6c696667
[7ffff540] 3a433d65 5456425c 5c6e6942 74736554
[7ffff550] 6e695c73 6c617473 6361706c 6567616b
[7ffff560] 6973635c 66676f6c 2e656c69 00676f6c
[7ffff570] 646e6977 5f73776f 63617274 5f676e69
[7ffff580] 67616c66 00333d73 646e6977 433d7269
[7ffff590] 69575c3a 776f646e 53560073 43303131
[7ffff5a0] 544e4d4f 534c4f4f 5c3a433d 676f7250
[7ffff5b0] 206d6172 656c6946 78282073 5c293638
[7ffff5c0] 7263694d 666f736f 69562074 6c617573
[7ffff5d0] 75745320 206f6964 302e3131 6d6f435c
[7ffff5e0] 376e6f6d 6f6f545c 005c736c 52455355
[7ffff5f0] 464f5250 3d454c49 555c3a43 73726573
[7ffff600] 7870765c 71336d6a 36766d39 36786768
[7ffff610] 726b6279 5355006a 414e5245 763d454d
[7ffff620] 6d6a7870 6d397133 67683676 62793678
[7ffff630] 006a726b 52455355 414d4f44 413d4e49
[7ffff640] 70736167 2d657261 7474654e 5400706f
[7ffff650] 433d504d 73555c3a 5c737265 4a585056
[7ffff660] 317e334d 7070415c 61746144 636f4c5c
Contenuto della memoria
Memory and registers cleared
Loaded: C:/Users/VPXJM3~1/AppData/Local/Temp/qt_temp.lj4344
SPIM Version 9.1.7 of February 12, 2012
Copyright 1990-2012, James R. Larus.
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.



Caricamento di un sorgente assembly e reset dei registri


File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [16]
Data
Step-by-Step
Istruzione corrente
Modifica registri a run-time
Int Regs [16]
PC = 400010
EPC = 0
Cause = 0
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 4
R3 [v1] = 0
R4 [a0] = 1
R5 [a1] = 7ffff424
R6 [a2] = 7ffff428
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] = 7ffff420
R30 [s8] = 0
R31 [ra] = 0
Text
[00400000] 8fa40000 lw $4, 0($29)
[00400004] 27a50004 addiu $5, $29, 4
[00400008] 24a60004 addiu $6, $5, 4
[0040000c] 00041080 sll $2, $4, 2
[00400010] 00c23021 addu $6, $6, $2
[00400014] 0c100009 jal 0x00400024 [main]
[00400018] 00000000 nop
[0040001c] 3402000a ori $2, $0, 10
[00400020] 0000000c syscall
[00400024] 340a0019 ori $10, $0, 25
[00400028] 3c011001 lui $1, 4097 [value]
[0040002c] 8c2b0000 lw $11, 0($1) [value]
[00400030] 014b6020 add $12, $10, $11
[00400034] 014b6822 sub $13, $10, $11
[00400038] 3402000a ori $2, $0, 10
[0040003c] 0000000c syscall
Kernel Text Segment [80000000]..[80010000]
[80000180] 0001d821 addu $27, $0, $1
[80000184] 3c019000 lui $1, -28672
[80000188] ac220200 sw $2, 512($1)
[80000190] 34020001 ori $2, $0, 1
[80000194] 001a2082 srl $4, $26, 2
[80000198] 3084001f andi $4, $4, 31
[8000019c] 0000000c syscall
[800001a0] 34020004 ori $2, $0, 4
[800001a4] 3344003c andi $4, $26, 60
[800001a8] 3c019000 lui $1, -28672
[800001ac] 00240821 addu $1, $1, $4
[800001b0] 8c240180 lw $4, 384($1)
[800001b4] 00000000 nop
[800001b8] 0000000c syscall
Memory and registers cleared
Loaded: C:/Users/VPXJM3~1/AppData/Local/Temp/qt_temp.lj4344
SPIM Version 9.1.7 of February 12, 2012
Copyright 1990-2012, James R. Larus.
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.


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] = 1
R5 [a1] = 7ffff424
R6 [a2] = 7ffff42c
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] = 7ffff420
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] 0c100009 jal 0x00400024 [main] ; 188: jal main
[00400018] 00000000 nop ; 189: nop
[0040001c] 3402000a ori $2, $0, 10 ; 19 i $v0 10
[00400020] 0000000c syscall ; yscall # syscall 10 (exit)
[00400024] 340a0019 ori $10, $0, 25
[00400028] 3c011001 lui $1, 4097 [value]
[0040002c] 8c2b0000 lw $11, 0($1) [value]
[00400030] 014b6020 add $12, $10, $11
[00400034] 014 11
[00400038] 340
[0040003c] 000
Kernel Te egment [80000000]..[80010000]
[80000180] 000 ; 90: ve $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
Copy Ctrl+C
Select All Ctrl+A
Set Breakpoint
Clear Breakpoint
Click destro sull'istruzione per impostare un breakpoint
See the file README for a full copyright notice.
Memory and registers cleared
Loaded: C:/Users/VPXJM3~1/AppData/Local/Temp/qt_temp.Gw4344
SPIM Version 9.1.7 of February 12, 2012
Copyright 1990-2012, James R. Larus.
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.

Primo esempio

```
.globl main
.text
main:
   li $t2, 25      # Load immediate value (25)
   lw $t3, value   # Load the word stored at label 'value'
   add $t4, $t2, $t3    # Add
   sub $t5, $t2, $t3    # Subtract
   li $v0, 10           # Exit the program
   syscall              # by placing its code in $v0
   .data
value:  .word 12
```

Esercitazione

http://www.dsi.unive.it/~architet/lezioni/mod2/esercit_asm_1.html