

# Calcolatori Elettronici

## Esercitazioni Assembler

M. Sonza Reorda – M. Monetti

M. Rebaudengo – R. Ferrero

L. Sterpone – M. Grosso

[renato.ferrero@polito.it](mailto:renato.ferrero@polito.it)

Politecnico di Torino

Dipartimento di Automatica e Informatica

# Informazioni generali

- LABINF
  - 1° piano, ingresso da C.so Castelfidardo lato ovest
- Esercitazioni assistite: 2 squadre
  - Martedì h. 11.30-13.00 (fino a Latino)
  - Venerdì h. 11.30-13.00 (da Lenzini in poi)
- 1° laboratorio: 7-17 aprile 2020

# Organizzazione delle esercitazioni

- All'inizio della settimana saranno caricati:
  - il pdf con il testo dell'esercitazione
  - un breve video introduttivo all'esercitazione
- Durante la settimana si tengono le virtual classroom
  - anche se lo studente riceve gli inviti per entrambe le virtual classroom, ne deve seguire solo una (la suddivisione è in base al proprio cognome, vedi slide precedente).
- Al termine della settimana sarà caricato:
  - il pdf con la soluzione dell'esercitazione

# Interazione con docente e borsisti (1)

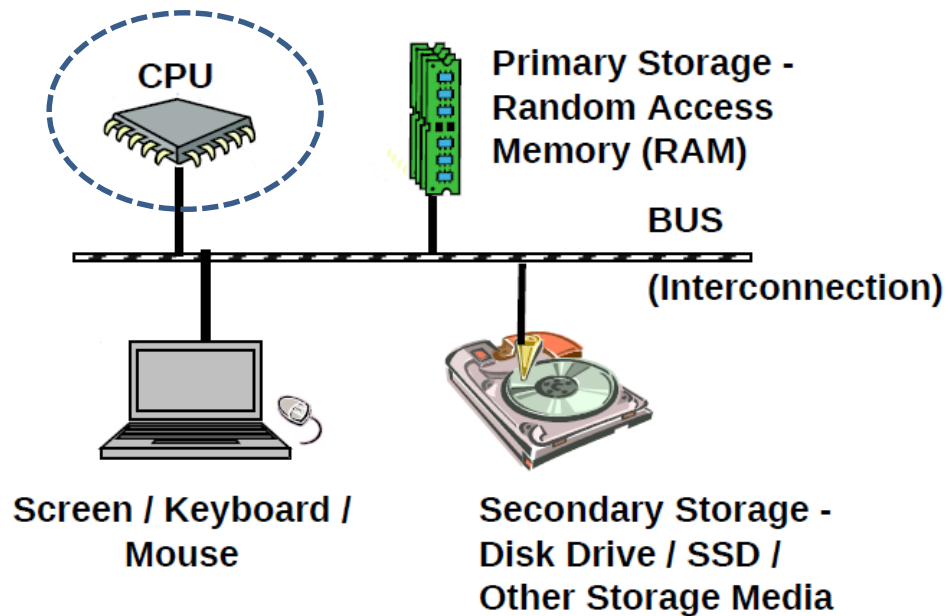
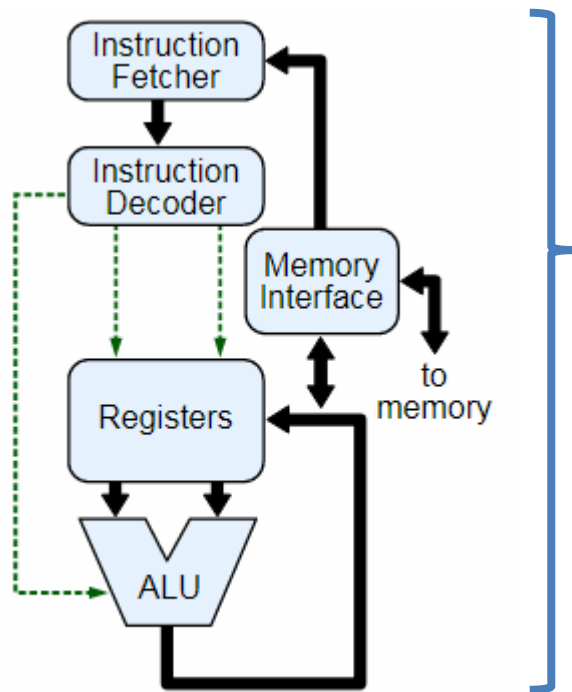
- Domande e risposte in tempo reale tramite la chat della virtual classroom
- Domande offline sul forum del portale, risposte durante la virtual classroom e sul forum
  - dare un titolo significativo al thread
  - prima di fare una domanda, controllare se esiste già un thread simile
  - il forum aumenta la collaborazione fra gli studenti: siete invitati a dare una risposta ad un vostro collega, oppure a continuare il thread con commenti, casi particolari, ecc.
  - le email sono scoraggiate: email di più studenti con la stessa domanda, nessuna interazione fra studenti.

# Interazione con docente e borsisti (2)

- Richiesta ad un borsista per consulenza individuale
  - tramite Zoom: <https://zoom.us/>
  - la prenotazione di uno slot (15 minuti) avviene scrivendo la propria matricola su questo documento:  
[https://docs.google.com/spreadsheets/d/1uymqFg3PH5d7UePoKMmxWwuOkJO\\_mgonq-Zpi-ng7Sk/edit?usp=sharing](https://docs.google.com/spreadsheets/d/1uymqFg3PH5d7UePoKMmxWwuOkJO_mgonq-Zpi-ng7Sk/edit?usp=sharing)
- Altre richieste di chiarimento sul gruppo Telegram.
- E' una proposta, sono possibili modifiche in corso
  - potreste essere contattati via mail nei giorni successivi ad una consulenza individuale per esprimere un giudizio: siete pregati di rispondere al veloce questionario.

# Il calcolatore

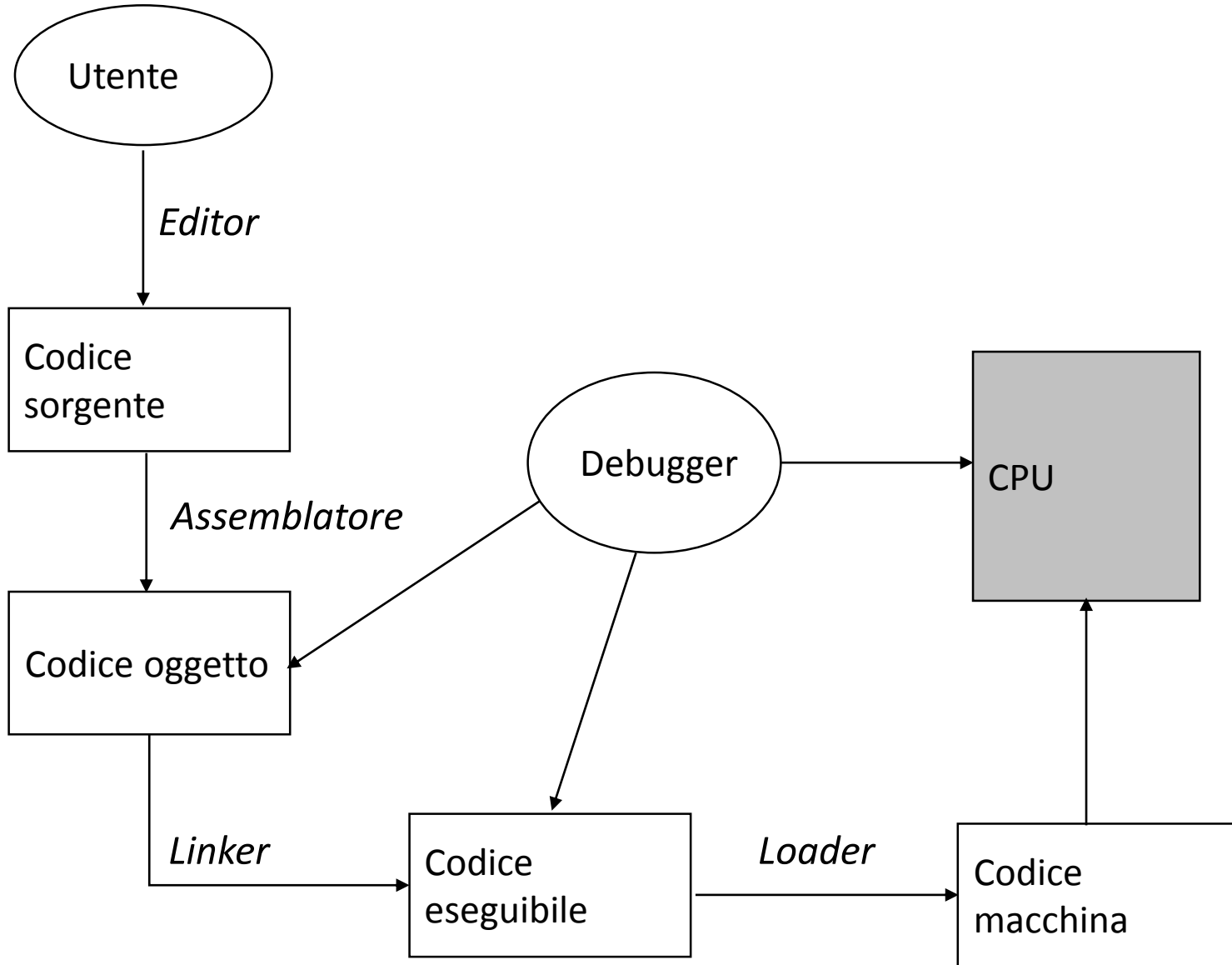
- Schema dal punto di vista del programmatore in linguaggio Assembly



# Architettura MIPS32 - Registri

Nome	Numero	Uso
<b>\$zero</b>	\$0	il valore costante 0
<b>\$at</b>	\$1	<i>riservato per assembler</i>
<b>\$v0-\$v1</b>	\$2-\$3	valori di ritorno della funzione
<b>\$a0-\$a3</b>	\$4-\$7	argomenti della funzione
<b>\$t0-\$t7</b>	\$8-\$15	valori temporanei
<b>\$s0-\$s7</b>	\$16-\$23	variabili salvate
<b>\$t8-\$t9</b>	\$24-\$25	altri valori temporanei
<b>\$k0-\$k1</b>	\$26-\$27	<i>riservato per sistema operativo</i>
<b>\$gp</b>	\$28	<i>riservato per puntatore al segmento di variabili globali</i>
<b>\$sp</b>	\$29	stack pointer
<b>\$fp</b>	\$30	frame pointer
<b>\$ra</b>	\$31	indirizzo di ritorno della funzione

# Ciclo di vita di un programma





# Editor

- Qualunque editor di testo va bene
    - Un editor di documenti (come Microsoft Word) non va bene perché aggiunge dati sulla formattazione non comprensibili da QtSpim.
  - Per Windows si può utilizzare Notepad++
    - è gratuito
    - permette di evidenziare le parole chiave del MIPS
- <https://notepad-plus-plus.org/downloads/>

# Colorazione della sintassi

- Un editor di testo può visualizzare un testo con diversi colori in base a regole sintattiche.
- Il vantaggio è il miglioramento della leggibilità del codice sorgente.
- Notepad++ supporta molti linguaggi.
- L'assembly MIPS non è incluso fra i linguaggi supportati di default, ma è possibile aggiungerlo a Notepad ++.

# Aggiunta di un nuovo linguaggio

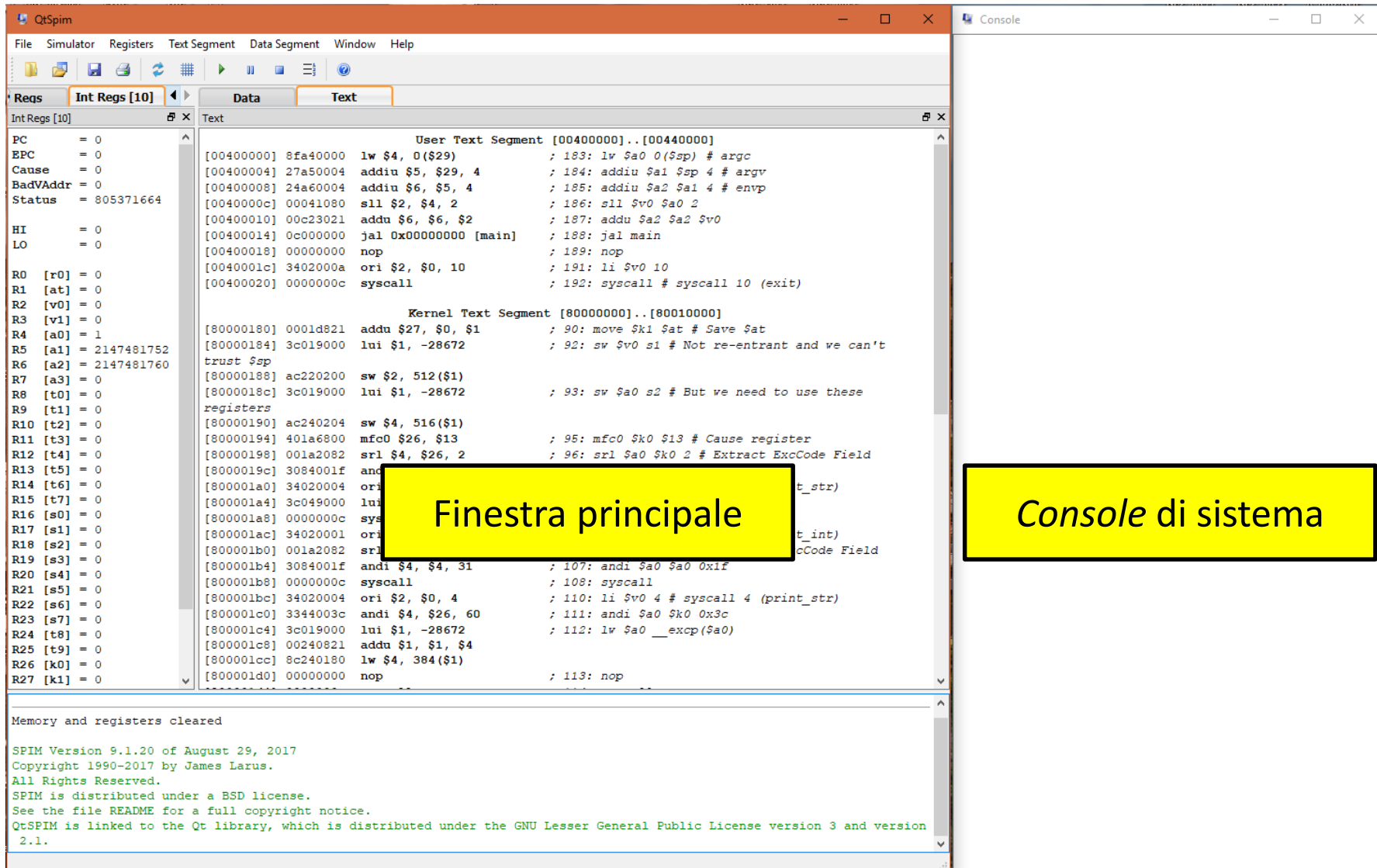
- Questo è un passaggio opzionale per la colorazione della sintassi di un nuovo linguaggio.
- E' necessario avere un file xml che descrive le regole sintattiche del linguaggio, ad esempio:  
<https://github.com/notepad-plus-plus/userDefinedLanguages/blob/master/udl-list.md>
- Per MIPS, scaricare ASM for MIPS R2000
- Copiare il file xml nella directory: Language -> User Defined Language -> Open User Defined Language folder; poi riavviare Notepad ++.



# QtSpim

- Simulatore di programmi per MIPS32
  - Legge ed esegue programmi scritti nel linguaggio assembly di questo processore
  - Include un semplice *debugger* e un insieme minimo di servizi del sistema operativo
  - Non è possibile eseguire programmi compilati (binario)
- È compatibile con (quasi) l'intero *instruction set* MIPS32 (Istruzioni e Pseudolstruzioni)
  - Non include confronti e arrotondamenti *floating point*
- È gratuito e open-source, e sono disponibili versioni per MS-Windows, Mac OS X e Linux  
<http://spimsimulator.sourceforge.net/>
- Informazioni utili: <http://spimsimulator.sourceforge.net/further.html>

# Interfaccia di QtSpim



# Finestra principale

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

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] = 1  
R5 [a1] = 2147481752  
R6 [a2] = 2147481760  
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

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 $1 # Not re-entrant and we can't
trust $sp
[80000188] ac220200 sw $2, 512($1)
[8000018c] 3c019000 lui $1, -28672 ; 93: sw $a0 $2 # 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
```

# tra princ

Visualizzazione dei registri  
dell'unità floating point

Barra dei pulsanti

- Carica / Reinizializza e carica
- Salva log / Stampa
- Azzera registri / Reinizializza
- Run/pause/stop/single-step
- Help

The screenshot shows a debugger window with a menu bar (File, Registers, Text Segment, Data Segment, Window, Help) and a toolbar. The 'Registers' tab is active, displaying a list of registers (PC, EPC, Cause, BadVAddr, Status, HI, LO, R0-R18) and their values. The 'Text' tab is also visible, showing memory segments (User Text Segment, Kernel Text Segment) and their contents (hex addresses, instructions, and comments). A callout points to the 'Int Regs [10]' tab, and another points to the 'Text' tab.

Registers (Int Regs [10]):

Register	Value
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]	= 0
R5 [a1]	= 0
R6 [a2]	= 0
R7 [a3]	= 0
R8 [a4]	= 0
R9 [a5]	= 0
R10 [a6]	= 0
R11 [a7]	= 0
R12 [a8]	= 0
R13 [a9]	= 0
R14 [a10]	= 0
R15 [a11]	= 0
R16 [a12]	= 0
R17 [a13]	= 0
R18 [a14]	= 0

Text Segment (User Text Segment [00400000]..[00440000]):

Address	Instruction	Comment
[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: ori \$v0 10
[00400020]	0000000c syscall	; 192: syscall # syscall 10 (exit)

Text Segment (Kernel Text Segment [80000000]..[80000000]):

Address	Instruction	Comment
[80000000]	00000000 nop	
[80000004]	00000000 nop	
[80000008]	00000000 nop	
[8000000c]	00000000 nop	
[80000010]	00000000 nop	
[80000014]	00000000 nop	
[80000018]	00000000 nop	
[8000001c]	00000000 nop	
[80000020]	00000000 nop	
[80000024]	00000000 nop	
[80000028]	00000000 nop	
[8000002c]	00000000 nop	
[80000030]	00000000 nop	
[80000034]	00000000 nop	
[80000038]	00000000 nop	
[8000003c]	00000000 nop	
[80000040]	00000000 nop	
[80000044]	00000000 nop	
[80000048]	00000000 nop	
[8000004c]	00000000 nop	
[80000050]	00000000 nop	
[80000054]	00000000 nop	
[80000058]	00000000 nop	
[8000005c]	00000000 nop	
[80000060]	00000000 nop	
[80000064]	00000000 nop	
[80000068]	00000000 nop	
[8000006c]	00000000 nop	
[80000070]	00000000 nop	
[80000074]	00000000 nop	
[80000078]	00000000 nop	
[8000007c]	00000000 nop	
[80000080]	00000000 nop	
[80000084]	00000000 nop	
[80000088]	00000000 nop	
[8000008c]	00000000 nop	
[80000090]	00000000 nop	
[80000094]	00000000 nop	
[80000098]	00000000 nop	
[8000009c]	00000000 nop	
[800000a0]	00000000 nop	
[800000a4]	00000000 nop	
[800000a8]	00000000 nop	
[800000ac]	00000000 nop	
[800000b0]	00000000 nop	
[800000b4]	00000000 nop	
[800000b8]	00000000 nop	
[800000bc]	00000000 nop	
[800000c0]	00000000 nop	
[800000c4]	00000000 nop	
[800000c8]	00000000 nop	
[800000cc]	00000000 nop	
[800000d0]	00000000 nop	
[800000d4]	00000000 nop	
[800000d8]	00000000 nop	
[800000dc]	00000000 nop	
[800000e0]	00000000 nop	
[800000e4]	00000000 nop	
[800000e8]	00000000 nop	
[800000ec]	00000000 nop	
[800000f0]	00000000 nop	
[800000f4]	00000000 nop	
[800000f8]	00000000 nop	
[800000fc]	00000000 nop	
[80000100]	00000000 nop	
[80000104]	00000000 nop	
[80000108]	00000000 nop	
[8000010c]	00000000 nop	
[80000110]	00000000 nop	
[80000114]	00000000 nop	
[80000118]	00000000 nop	
[8000011c]	00000000 nop	
[80000120]	00000000 nop	
[80000124]	00000000 nop	
[80000128]	00000000 nop	
[8000012c]	00000000 nop	
[80000130]	00000000 nop	
[80000134]	00000000 nop	
[80000138]	00000000 nop	
[8000013c]	00000000 nop	
[80000140]	00000000 nop	
[80000144]	00000000 nop	
[80000148]	00000000 nop	
[8000014c]	00000000 nop	
[80000150]	00000000 nop	
[80000154]	00000000 nop	
[80000158]	00000000 nop	
[8000015c]	00000000 nop	
[80000160]	00000000 nop	
[80000164]	00000000 nop	
[80000168]	00000000 nop	
[8000016c]	00000000 nop	
[80000170]	00000000 nop	
[80000174]	00000000 nop	
[80000178]	00000000 nop	
[8000017c]	00000000 nop	
[80000180]	00000000 nop	
[80000184]	00000000 nop	
[80000188]	00000000 nop	
[8000018c]	00000000 nop	
[80000190]	00000000 nop	
[80000194]	00000000 nop	
[80000198]	00000000 nop	
[8000019c]	00000000 nop	
[800001a0]	00000000 nop	
[800001a4]	00000000 nop	
[800001a8]	00000000 nop	
[800001ac]	00000000 nop	
[800001b0]	00000000 nop	
[800001b4]	00000000 nop	
[800001b8]	00000000 nop	
[800001bc]	00000000 nop	
[800001c0]	00000000 nop	
[800001c4]	00000000 nop	
[800001c8]	00000000 nop	
[800001cc]	00000000 nop	
[800001d0]	00000000 nop	
[800001d4]	00000000 nop	
[800001d8]	00000000 nop	
[800001dc]	00000000 nop	
[800001e0]	00000000 nop	
[800001e4]	00000000 nop	
[800001e8]	00000000 nop	
[800001ec]	00000000 nop	
[800001f0]	00000000 nop	
[800001f4]	00000000 nop	
[800001f8]	00000000 nop	
[800001fc]	00000000 nop	
[80000200]	00000000 nop	
[80000204]	00000000 nop	
[80000208]	00000000 nop	
[8000020c]	00000000 nop	
[80000210]	00000000 nop	
[80000214]	00000000 nop	
[80000218]	00000000 nop	
[8000021c]	00000000 nop	
[80000220]	00000000 nop	
[80000224]	00000000 nop	
[80000228]	00000000 nop	
[8000022c]	00000000 nop	
[80000230]	00000000 nop	
[80000234]	00000000 nop	
[80000238]	00000000 nop	
[8000023c]	00000000 nop	
[80000240]	00000000 nop	
[80000244]	00000000 nop	
[80000248]	00000000 nop	
[8000024c]	00000000 nop	
[80000250]	00000000 nop	
[80000254]	00000000 nop	
[80000258]	00000000 nop	
[8000025c]	00000000 nop	
[80000260]	00000000 nop	
[80000264]	00000000 nop	
[80000268]	00000000 nop	
[8000026c]	00000000 nop	
[80000270]	00000000 nop	
[80000274]	00000000 nop	
[80000278]	00000000 nop	
[8000027c]	00000000 nop	
[80000280]	00000000 nop	
[80000284]	00000000 nop	
[80000288]	00000000 nop	
[8000028c]	00000000 nop	
[80000290]	00000000 nop	
[80000294]	00000000 nop	
[80000298]	00000000 nop	
[8000029c]	00000000 nop	
[800002a0]	00000000 nop	
[800002a4]	00000000 nop	
[800002a8]	00000000 nop	
[800002ac]	00000000 nop	
[800002b0]	00000000 nop	
[800002b4]	00000000 nop	
[800002b8]	00000000 nop	
[800002bc]	00000000 nop	
[800002c0]	00000000 nop	
[800002c4]	00000000 nop	
[800002c8]	00000000 nop	
[800002cc]	00000000 nop	
[800002d0]	00000000 nop	
[800002d4]	00000000 nop	
[800002d8]	00000000 nop	
[800002dc]	00000000 nop	
[800002e0]	00000000 nop	
[800002e4]	00000000 nop	
[800002e8]	00000000 nop	
[800002ec]	00000000 nop	
[800002f0]	00000000 nop	
[800002f4]	00000000 nop	
[800002f8]	00000000 nop	
[800002fc]	00000000 nop	
[80000300]	00000000 nop	
[80000304]	00000000 nop	
[80000308]	00000000 nop	
[8000030c]	00000000 nop	
[80000310]	00000000 nop	
[80000314]	00000000 nop	
[80000318]	00000000 nop	
[8000031c]	00000000 nop	
[80000320]	00000000 nop	
[80000324]	00000000 nop	
[80000328]	00000000 nop	
[8000032c]	00000000 nop	
[80000330]	00000000 nop	
[80000334]	00000000 nop	
[80000338]	00000000 nop	
[8000033c]	00000000 nop	
[80000340]	00000000 nop	
[80000344]	00000000 nop	
[80000348]	00000000 nop	
[8000034c]	00000000 nop	
[80000350]	00000000 nop	
[80000354]	00000000 nop	
[80000358]	00000000 nop	
[8000035c]	00000000 nop	
[80000360]	00000000 nop	
[80000364]	00000000 nop	
[80000368]	00000000 nop	
[8000036c]	00000000 nop	
[80000370]	00000000 nop	
[80000374]	00000000 nop	
[80000378]	00000000 nop	
[8000037c]	00000000 nop	
[80000380]	00000000 nop	
[80000384]	00000000 nop	
[80000388]	00000000 nop	
[8000038c]	00000000 nop	
[80000390]	00000000 nop	
[80000394]	00000000 nop	
[80000398]	00000000 nop	
[8000039c]	00000000 nop	
[800003a0]	00000000 nop	
[800003a4]	00000000 nop	
[800003a8]	00000000 nop	
[800003ac]	00000000 nop	
[800003b0]	00000000 nop	
[800003b4]	00000000 nop	
[800003b8]	00000000 nop	
[800003bc]	00000000 nop	
[800003c0]	00000000 nop	
[800003c4]	00000000 nop	
[800003c8]	00000000 nop	
[800003cc]	00000000 nop	
[800003d0]	00000000 nop	
[800003d4]	00000000 nop	
[800003d8]	00000000 nop	
[800003dc]	00000000 nop	
[800003e0]	00000000 nop	
[800003e4]	00000000 nop	
[800003e8]	00000000 nop	
[800003ec]	00000000 nop	
[800003f0]	00000000 nop	
[800003f4]	00000000 nop	
[800003f8]	00000000 nop	
[800003fc]	00000000 nop	
[80000400]	00000000 nop	
[80000404]	00000000 nop	
[80000408]	00000000 nop	
[8000040c]	00000000 nop	
[80000410]	00000000 nop	
[80000414]	00000000 nop	
[80000418]	00000000 nop	
[8000041c]	00000000 nop	
[80000420]	00000000 nop	
[80000424]	00000000 nop	
[80000428]	00000000 nop	
[8000042c]	00000000 nop	
[80000430]	00000000 nop	
[80000434]	00000000 nop	
[80000438]	00000000 nop	
[8000043c]	00000000 nop	
[80000440]	00000000 nop	
[80000444]	00000000 nop	
[80000448]	00000000 nop	
[8000044c]	00000000 nop	
[80000450]	00000000 nop	
[80000454]	00000000 nop	
[80000458]	00000000 nop	
[8000045c]	00000000 nop	
[80000460]	00000000 nop	
[80000464]	00000000 nop	
[80000468]	00000000 nop	
[8000046c]	00000000 nop	
[80000470]	00000000 nop	
[80000474]	00000000 nop	
[80000478]	00000000 nop	
[8000047c]	00000000 nop	
[80000480]	00000000 nop	
[80000484]	00000000 nop	
[80000488]	00000000 nop	
[8000048c]	00000000 nop	
[80000490]	00000000 nop	
[80000494]	00000000 nop	
[80000498]	00000000 nop	
[8000049c]	00000000 nop	
[800004a0]	00000000 nop	
[800004a4]	00000000 nop	
[800004a8]	00000000 nop	
[800004ac]	00000000 nop	
[800004b0]	00000000 nop	
[800004b4]	00000000 nop	
[800004b8]	00000000 nop	
[800004bc]	00000000 nop	
[800004c0]	00000000 nop	
[800004c4]	00000000 nop	
[800004c8]	00000000 nop	
[800004cc]	00000000 nop	
[800004d0]	00000000 nop	</

# Finestra principale

```
--- ---- -
R4 [a0] = 1
R5 [a1] = 2147481752
R6 [a2] = 2147481760
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

[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, 0($1)
[800001d0] 00000000 nop

Memory and registers cleared

SPIM Version 9.1.20 of August 29, 2017
Copyright 1990-2017 by James Larus.
All Rights Reserved.
SPIM is distributed under a BSD license.
See the file README for a full copyright notice.
QtSPIM is linked to the Qt library, which is distributed under the GNU Lesser General Public License version 3 and version 2.1.
```

Console informativa

- Messaggi di informazione e di errore



# Finestra principale

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

Reqs 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] = 1  
R5 [a1] = 2147481752  
R6 [a2] = 2147481760  
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

Data

User data segment [10000000]..[10040000]  
[10000000]..[1000ffff] 00000000  
[10010000] 00000203 00000000 00000000 00000000 . . . . .  
[10010010]..[1003ffff] 00000000

User Stack [7ffff894]..[80000000]  
[7ffff894] 00000001 7ffff946 00000000 . . . . F . . . . .  
[7ffff8a0] 7fffffe1 7fffffb3 7fffff7c 7fffff40 . . . . . | . . . @ . . .  
[7ffff8b0] 7fffff0f 7ffffef2 7ffffece 7ffffe9c . . . . .  
[7ffff8c0] 7ffffe6b 7fffff36 7ffffe36 7ffffe19 k . . . C . . . 6 . . . .  
[7ffff8d0] 7ffffde8 7ffffdb3 7ffffdb3 7ffffdb3 . . . . .  
[7ffff8e0] 7ffffd7d 7ffffc1d 7ffffc1d . . . v . . . 8 . . . .  
[7ffff8f0] 7ffffc00 7ffffb8e 7ffffb8e . . . . .  
[7ffff900] 7ffffb73 7ffffb73 . . . . .  
[7ffff910] 7ffffb73 7ffffb73 . . . . .  
[7ffff920] 7ffffb73 7ffffb73 . . . . .  
[7ffff930] 7ffffb73 7ffffb73 . . . . .  
[7ffff940] 00000000 7ffffb73 7ffffb73 . . . . .  
[7ffff950] 677ffffb73 7ffffb73 7ffffb73 . . . . .  
[7ffff960] 6f7ffffb73 7ffffb73 7ffffb73 . . . . .  
[7ffff970] 3a7ffffb73 7ffffb73 7ffffb73 . . . . .  
[7ffff980] 4f7ffffb73 7ffffb73 7ffffb73 . . . . .  
[7ffff990] 697ffffb73 7ffffb73 7ffffb73 . . . . .  
[7ffff9a0] 4e7ffffb73 7ffffb73 7ffffb73 . . . . .  
[7ffff9b0] 6f7ffffb73 7ffffb73 7ffffb73 . . . . .  
[7ffff9c0] 4d414f52 50474e49 49464f52 443d454c R O A M I N G P R O F I L E = D  
[7ffff9d0] 544b5345 412d504f 4f4e4b4e 55003548 E S K T O P - A N K N O H S . U

Sezione di dati in memoria (utente, stack e kernel)

- Indirizzo (word) o intervallo di indirizzi hex
- Contenuto memoria in esadecimale (little- o big- endian dipende dal processore. MS-Windows usa little-endian)
- Contenuto memoria in ASCII

# Codice di esempio

- Il codice può essere introdotto con un qualsiasi editor di testo, e salvato in un file con estensione `.a`, `.s` o `.asm`

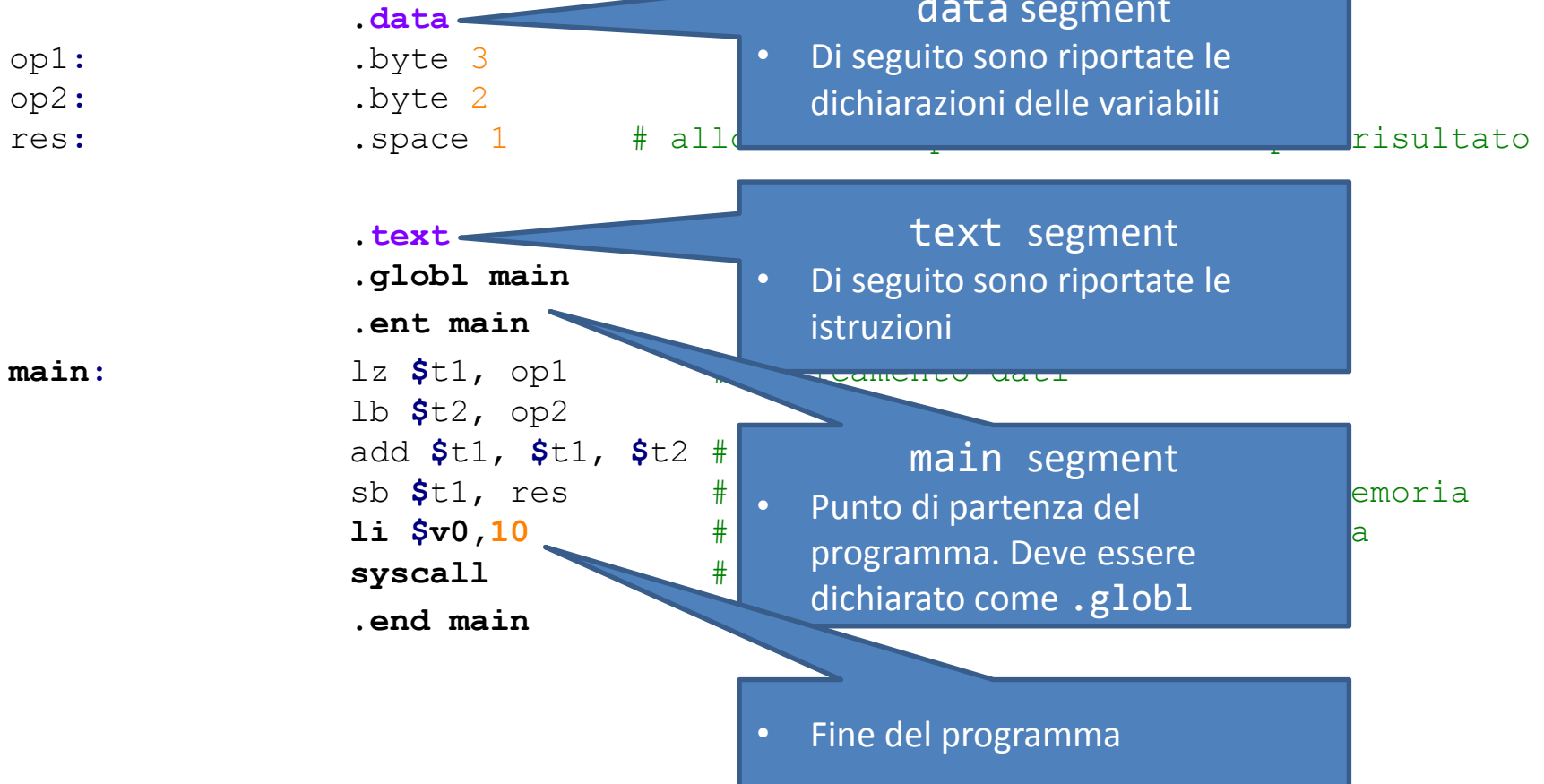
```

                                .data          # dichiarazione dati
op1:                            .byte 3
op2:                            .byte 2
res:                            .space 1      # allocazione spazio in memoria per risultato

                                .text
                                .globl main
                                .ent main
main:                            lz $t1, op1      # caricamento dati
                                lb $t2, op2
                                add $t1, $t1, $t2 # esecuzione somma
                                sb $t1, res        # salvataggio del risultato in memoria
                                li $v0, 10         # codice per uscita dal programma
                                syscall            # fine
                                .end main
```

# Codice di esempio

- Il codice può essere introdotto con un qualsiasi editor di testo, e salvato in un file con estensione `.asm`



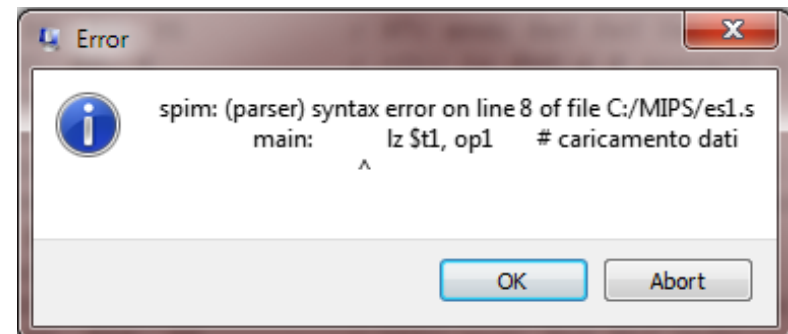
# Caricamento del codice

- In QtSpim, dal menu *File* selezionare “*Reinitialize and Load File*”, quindi selezionare il codice salvato precedentemente

– In alternativa, premere il pulsante



- Eventuali errori di sintassi sono segnalati e richiedono la correzione del codice



- Quando il codice è correttamente caricato, è possibile agire sugli opportuni pulsanti per eseguirlo



# Debug

- L'esecuzione passo-passo è fondamentale per il *debug*
  - Osservare il valore di memoria e registri al termine di ogni istruzione
- È possibile inserire un *breakpoint* facendo click con il pulsante destro sull'istruzione desiderata nella sezione *Text* della finestra principale, e selezionando "*Set Breakpoint*"
- Ogni volta che il codice viene modificato, è necessario ripartire da "*Reinitialize and Load File*"

# Debug [cont.]

- Esempio: esecuzione dell'istruzione di memorizzazione

```
[00400034] 012a4820  add $9, $9, $10          ; 10: add $t1, $t1, $t2 # esecuzione somma
[00400038] 3c011001  lui $1, 4097             ; 11: sb $t1, res # salvataggio del risultato in memoria
[0040003c] a0290002  sb $9, 2($1)             ; 
[00400040] 3402000a  ori $2, $0, 10           ; 12: li $v0,10 # codice per uscita dal programma
[00400044] 0000000c  syscall                  ; 13: syscall # fine
```

- Variabili prima del salvataggio:

Data	Text
Data	
User data segment [10000000]..[10040000]	
[10000000]..[1000ffff]	00000000
[10010000]	00000203 00000000 00000000 . . . . .
[10010010]..[1003ffff]	00000000

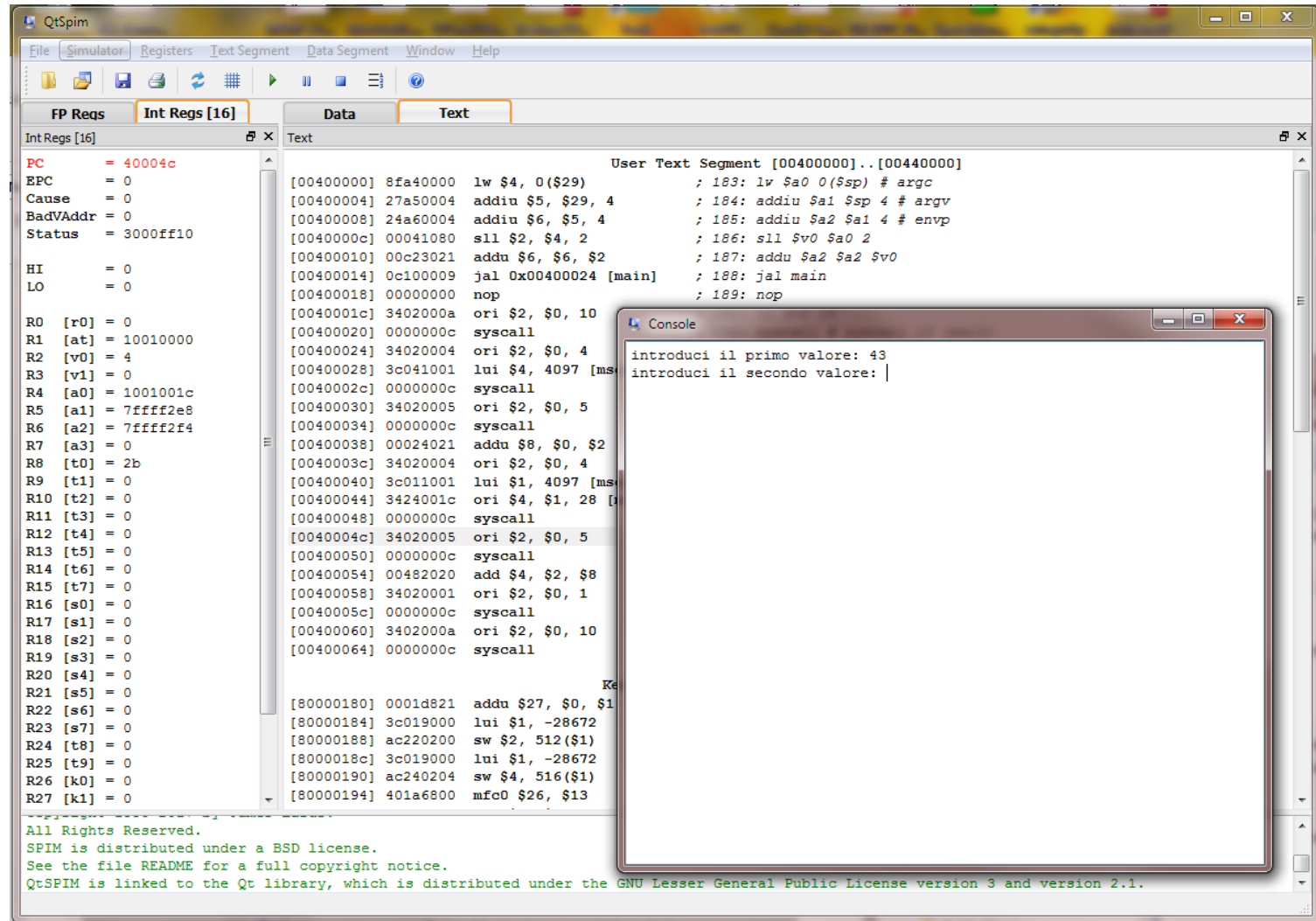
- Variabili dopo il salvataggio:

Data	Text
Data	
User data segment [10000000]..[10040000]	
[10000000]..[1000ffff]	00000000
[10010000]	00050203 00000000 00000000 00000000 . . . . .
[10010010]..[1003ffff]	00000000

# Input/Output da *console*

```
.data
msg1: .ascii "introduci il primo valore: "
msg2: .ascii "introduci il secondo valore: "
.text
.globl main
.ent main
main:  li $v0, 4          # syscall 4 (print_str)
      la $a0, msg1      # argomento: stringa
      syscall           # stampa la stringa
      li $v0, 5          # syscall 5 (read_int)
      syscall
      move $t0, $v0      # primo operando
      li $v0, 4
      la $a0, msg2
      syscall
      li $v0, 5
      syscall
      add $a0, $v0, $t0  # somma degli operandi
      li $v0, 1          # syscall 1 (print_int)
      syscall
      li $v0, 10         # codice per uscita dal programma
      syscall            # fine
      .end main
```

# Input/Output da *console* [cont.]





# Scrittura di un valore in una cella di memoria

```
variabile:  .data
            .space 4      # int variabile
            .text
            .globl main
            .ent main

main:

            li  $t0, 19591501      # variabile = 19591501 (012A F14D hex)
            sw  $t0, variabile

            li  $v0, 10
            syscall

            .end main
```

# Ricerca del carattere minimo

```
.data
wVet:      .space 5
wRes:      .space 1
message_in: .ascii "Inserire caratteri\n"
message_out: .ascii "\nValore Minimo : "

.text
.globl main
.ent main

main:
    la $t0, wVet          # puntatore a inizio del vettore
    li $t1, 0             # contatore

    la $a0, message_in    # indirizzo della stringa
    li $v0, 4             # system call per stampare stringa
    syscall
```

# Ricerca del carattere minimo [cont.]

```
ciclo1: li  $v0, 12          # legge 1 char
        syscall            # system call (risultato in $v0)
        sb  $v0, ($t0)
        add $t1, $t1, 1
        add $t0, $t0, 1
        bne $t1, 5, ciclo1  # itera 5 volte

        la  $t0, wVet
        li  $t1, 0          # contatore
        lb  $t2, ($t0)      # in $t2 memorizzo MIN iniziale

ciclo2: lb  $t3, ($t0)
        bgt $t3, $t2, salta  # salta se NON deve aggiornare MIN
        lb  $t2, ($t0)      # aggiorna MIN
salta:  add $t1, $t1, 1
        add $t0, $t0, 1
        bne $t1, 5, ciclo2
```

# Ricerca del carattere minimo [cont.]

```
la $a0, message_out  
li $v0, 4  
syscall
```

```
li    $v0, 11           # stampa 1 char  
move  $a0, $t2  
syscall
```

```
li $v0, 10  
syscall  
.end main
```

# Laboratorio:

- Si prendano gli esempi di codice presentati in precedenza e quelli nell' "Introduzione al linguaggio MIPS" (ASSEM\_00.pdf).
- Si richiede di inserire tali esempi di codice in QtSpim, compilarli, eseguirli e analizzarne il comportamento in modalità di debug.