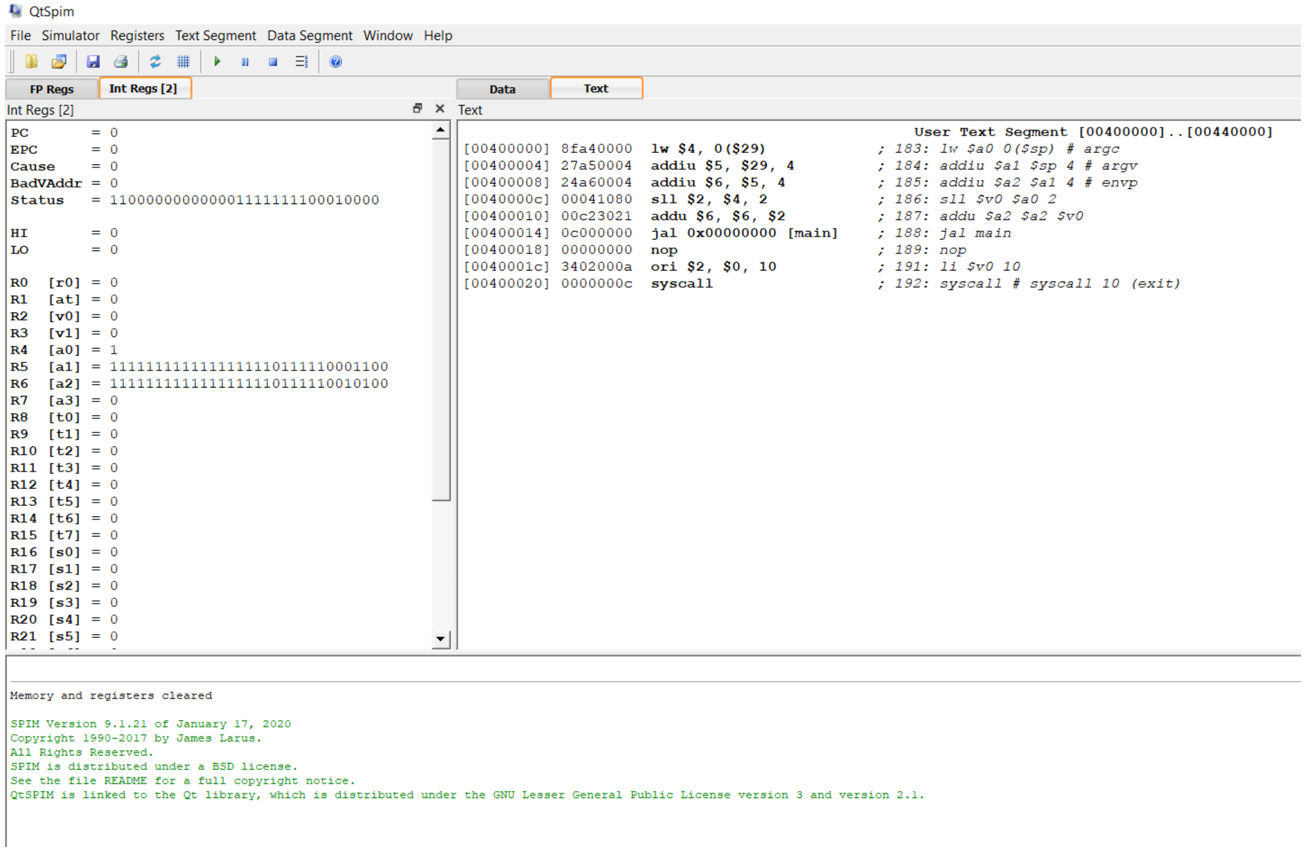
QtSpim Guide

ECE4801

SPIM

- When you open QtSpim, a window will open as shown on the right.
- The window is divided into different sections.


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [2]
Int Regs [2]
Data
Text
Text

```
PC       = 0
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 110000000000001111111100010000

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 1111111111111111110111110001100
R6  [a2] = 1111111111111111110111110010100
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
```

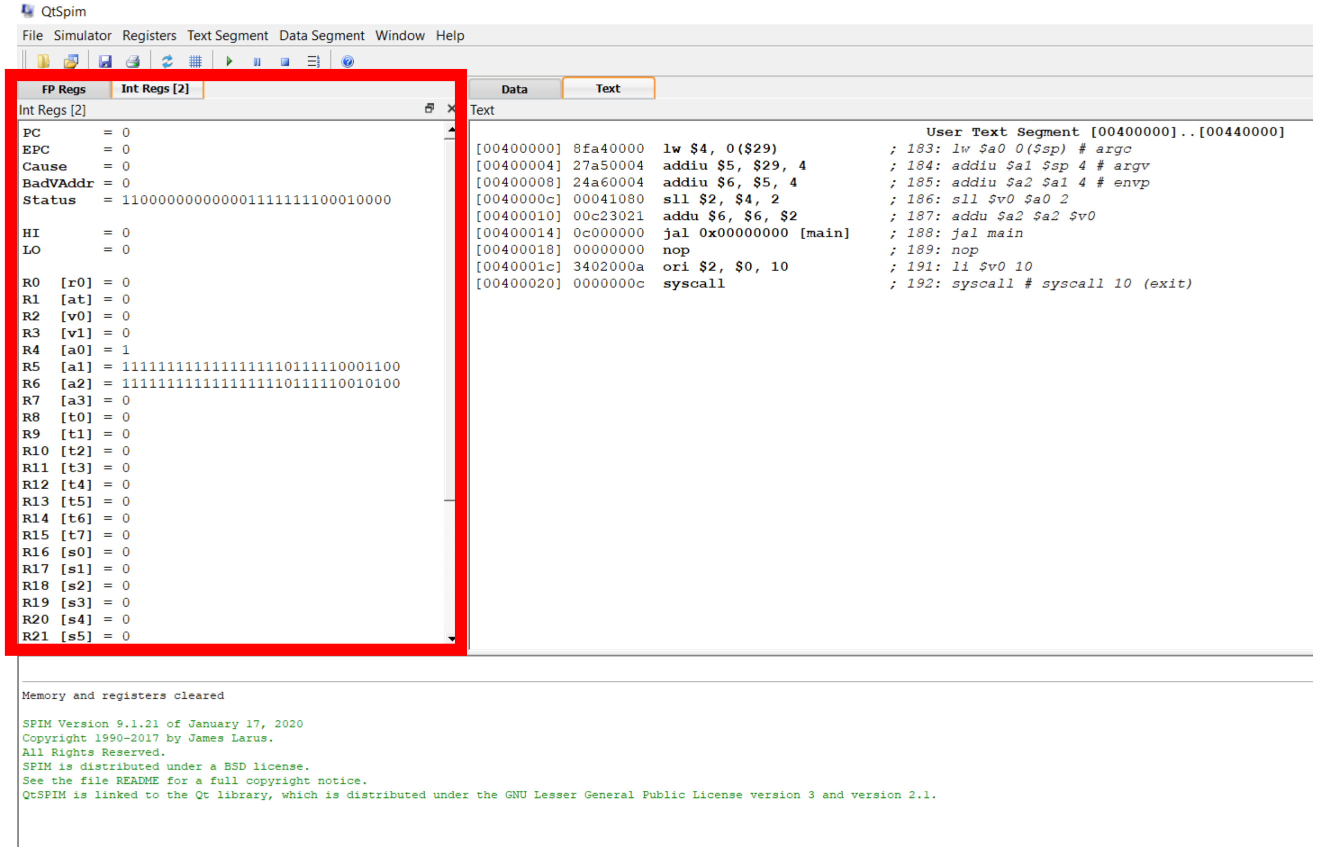
```
                                                 User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)              ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4           ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4            ; 185: 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)
```

```
Memory and registers cleared

SPIM Version 9.1.21 of January 17, 2020
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.
```

SPIM

- 1. The *Register* tabs display the content of all registers.



QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [2]
Int Regs [2]
Data
Text
Text

```
PC        = 0
EPC       = 0
Cause     = 0
BadVAddr  = 0
Status    = 11000000000000011111111100010000

HI        = 0
LO        = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 11111111111111111101111110001100
R6  [a2] = 11111111111111111101111110010100
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
```

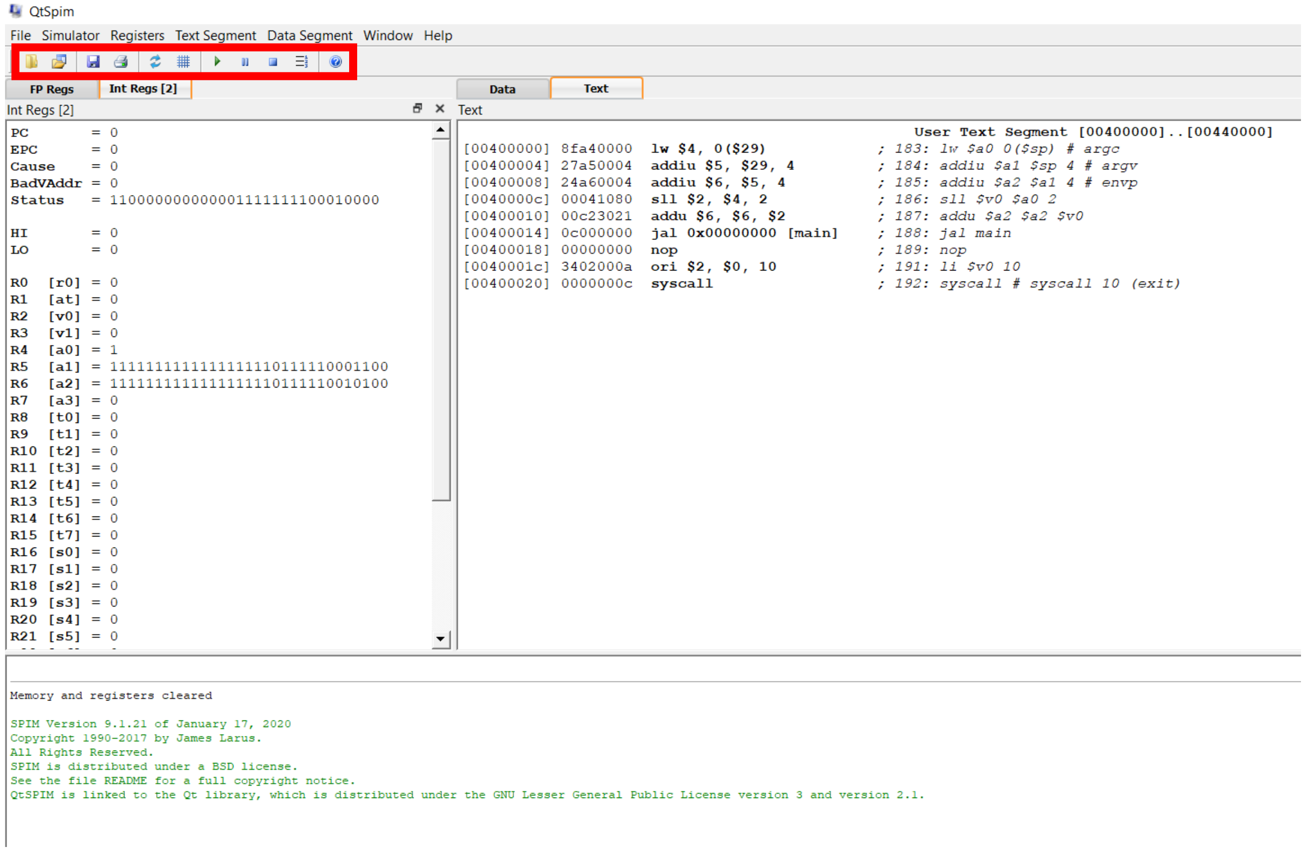
```
                                                        User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)             ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4          ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4           ; 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)
```

```
Memory and registers cleared

SPIM Version 9.1.21 of January 17, 2020
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.
```

SPIM

- 1. The Register tabs display the content of all registers.
- 2. Buttons across the top are used to load and run a simulation. You can also use the single-step button for debugging.



QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [2]
Data
Text
Int Regs [2]
Text

```
PC       = 0
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 11000000000000111111100010000

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 11111111111111111011111110001100
R6  [a2] = 11111111111111111011111110010100
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
```

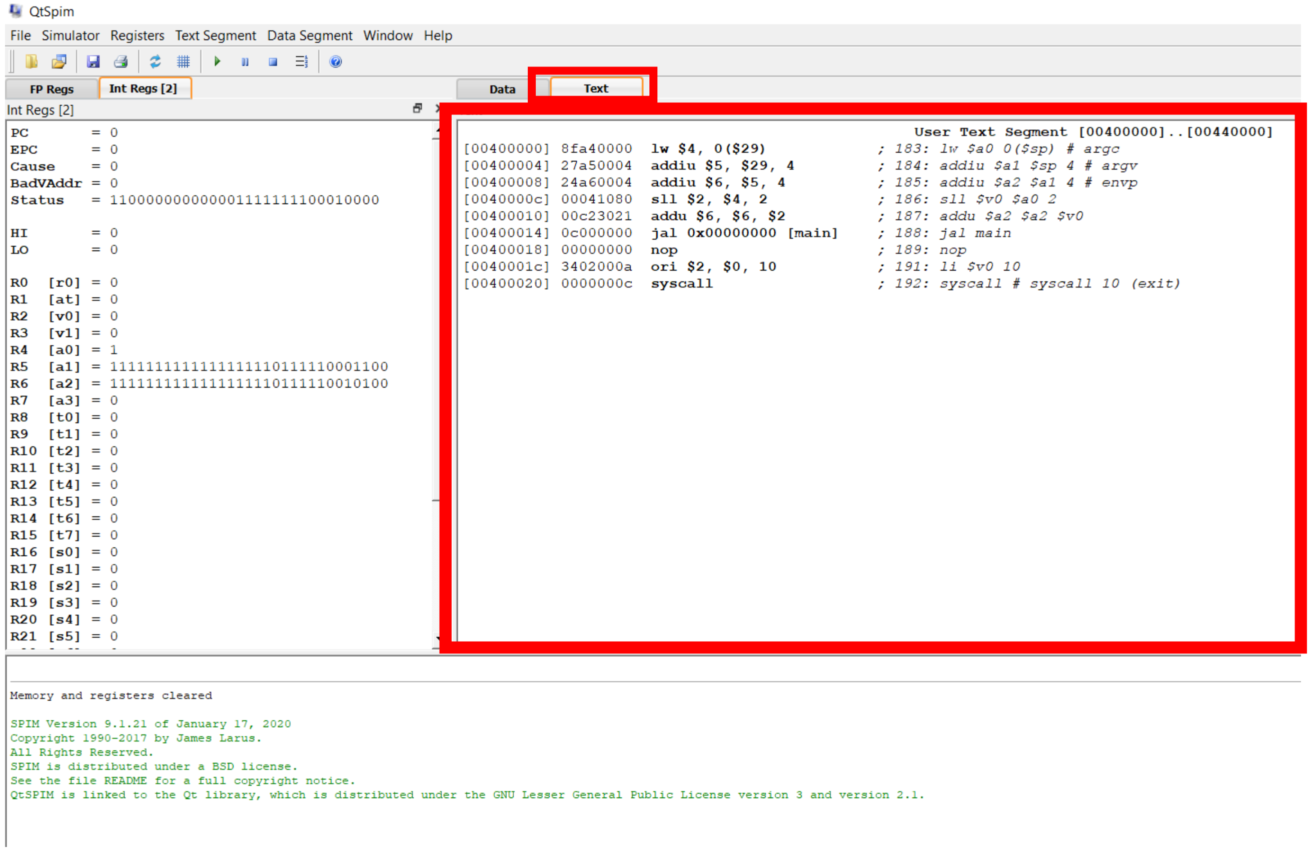
```
                                                       User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)              ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4           ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4            ; 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)
```

```
Memory and registers cleared

SPIM Version 9.1.21 of January 17, 2020
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.
```

SPIM

- 1. The Register tabs display the content of all registers.
- 2. Buttons across the top are used to load and run a simulation.
- 3. The Text tab displays the MIPS instructions loaded into memory to be executed.



```
PC       = 0
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 1100000000000011111111100010000

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 11111111111111111011111110001100
R6  [a2] = 11111111111111111011111110010100
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
```

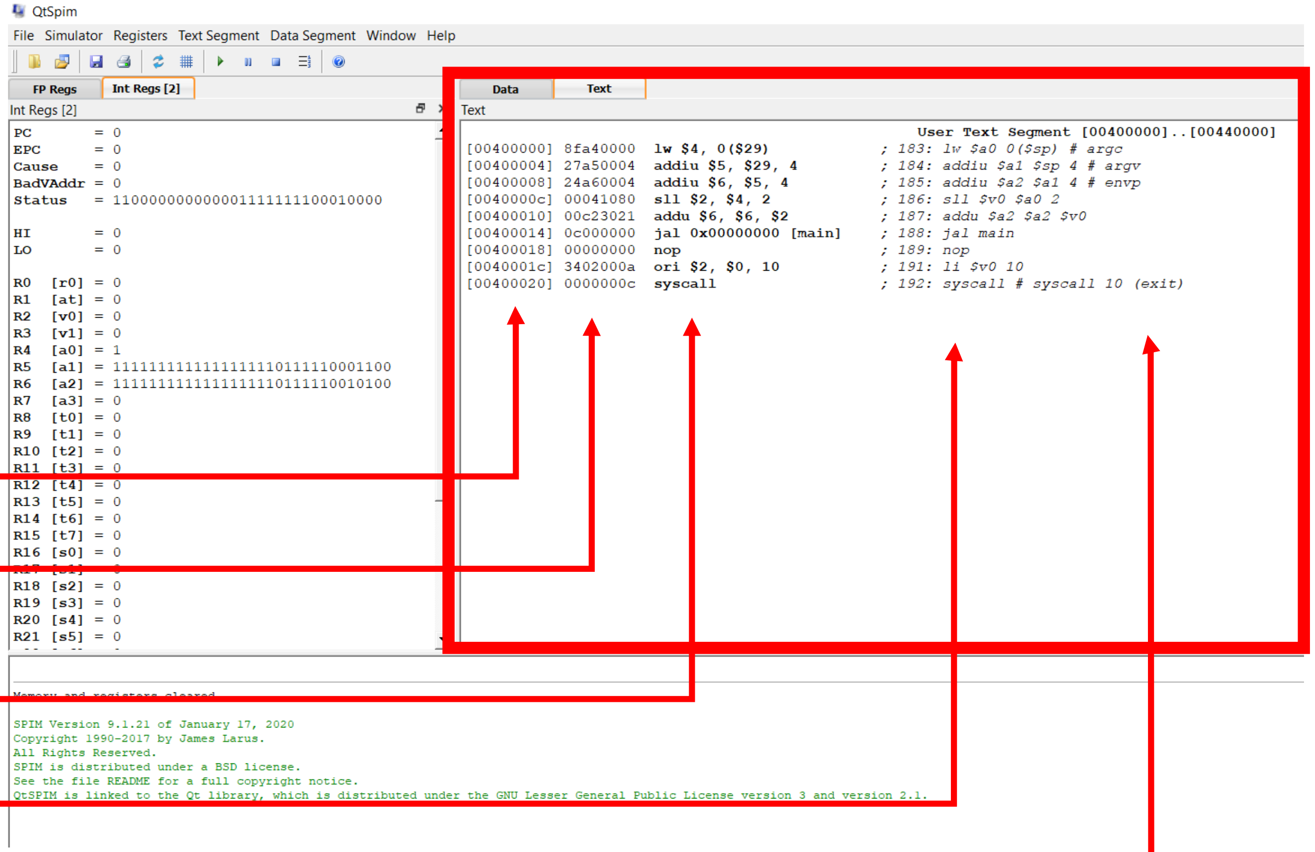
```
                                                    User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)           ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4        ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4         ; 185: addiu $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)
```

```
Memory and registers cleared

SPIM Version 9.1.21 of January 17, 2020
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.
```

SPIM

- 1. The Register tabs display the content of all registers.
- 2. Buttons across the top are used to load and run a simulation.
- 3. The Text tab displays the MIPS instructions loaded into memory to be executed. From left-to-right,
  - the memory address of an instruction,
  - The content of the memory (machine code of the instruction in hex)
  - the actual MIPS instructions where register numbers are used,
  - the MIPS assembly code that you wrote,
  - and any comments you made in your code are displayed.



```
PC       = 0
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 11000000000000111111100010000

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 111111111111111110111110001100
R6  [a2] = 111111111111111110111110010100
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
```

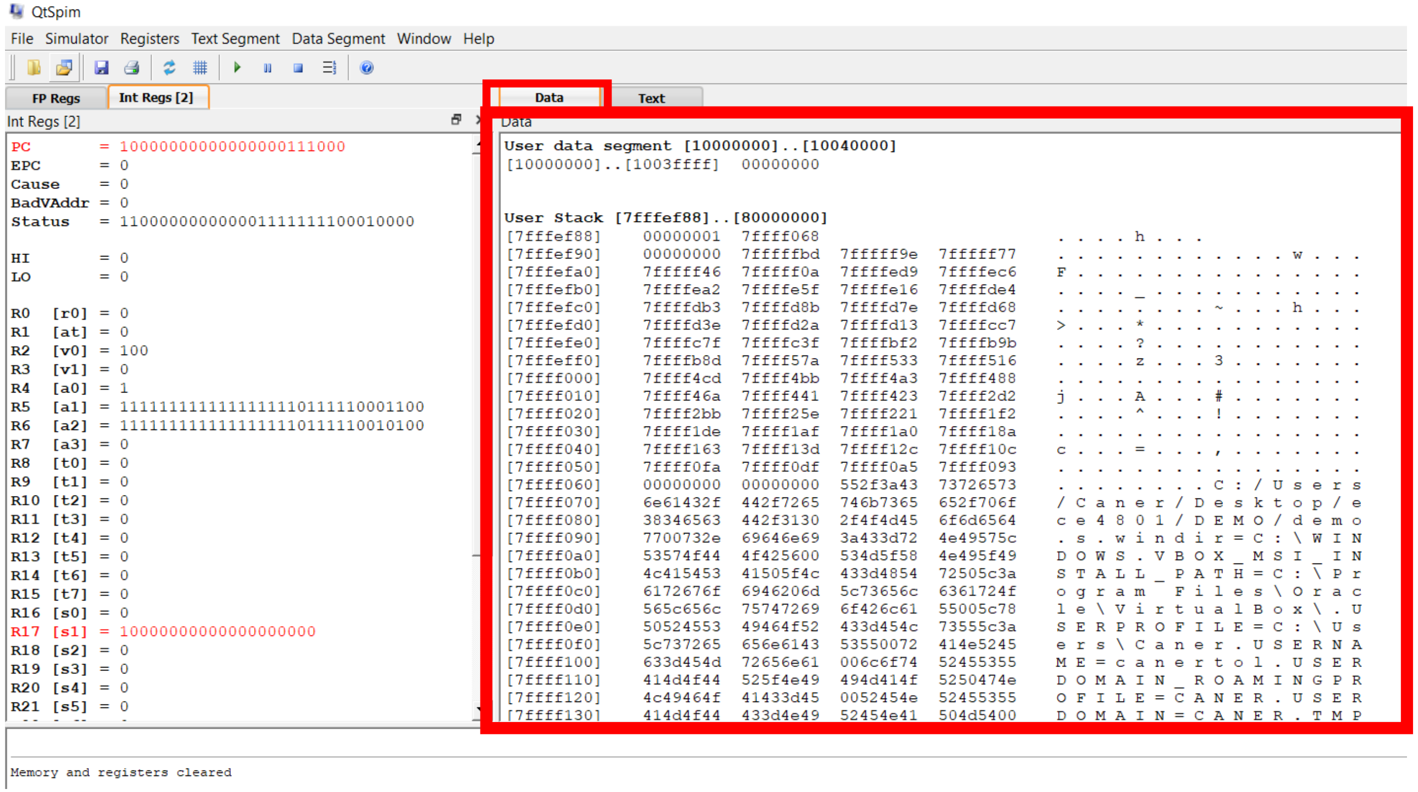
```
                                                      User Text Segment [00400000]..[00440000]
[00400000] 8fa40000  lw $4, 0($29)               ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004  addiu $5, $29, 4            ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004  addiu $6, $5, 4             ; 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)
```

```
Memory and registers cleared

SPIM Version 9.1.21 of January 17, 2020
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.
```

SPIM

- 4. The Data tab displays memory addresses and their values in the data and stack segments of the memory.


QtSpim
File Simulator Registers Text Segment Data Segment Window Help
FP Regs
Int Regs [2]
Data
Text
Int Regs [2]
Data

```
PC       = 1000000000000000000111000
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 110000000000000111111111100010000

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 100
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 11111111111111111101111110001100
R6  [a2] = 11111111111111111101111110010100
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] = 10000000000000000000
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
```

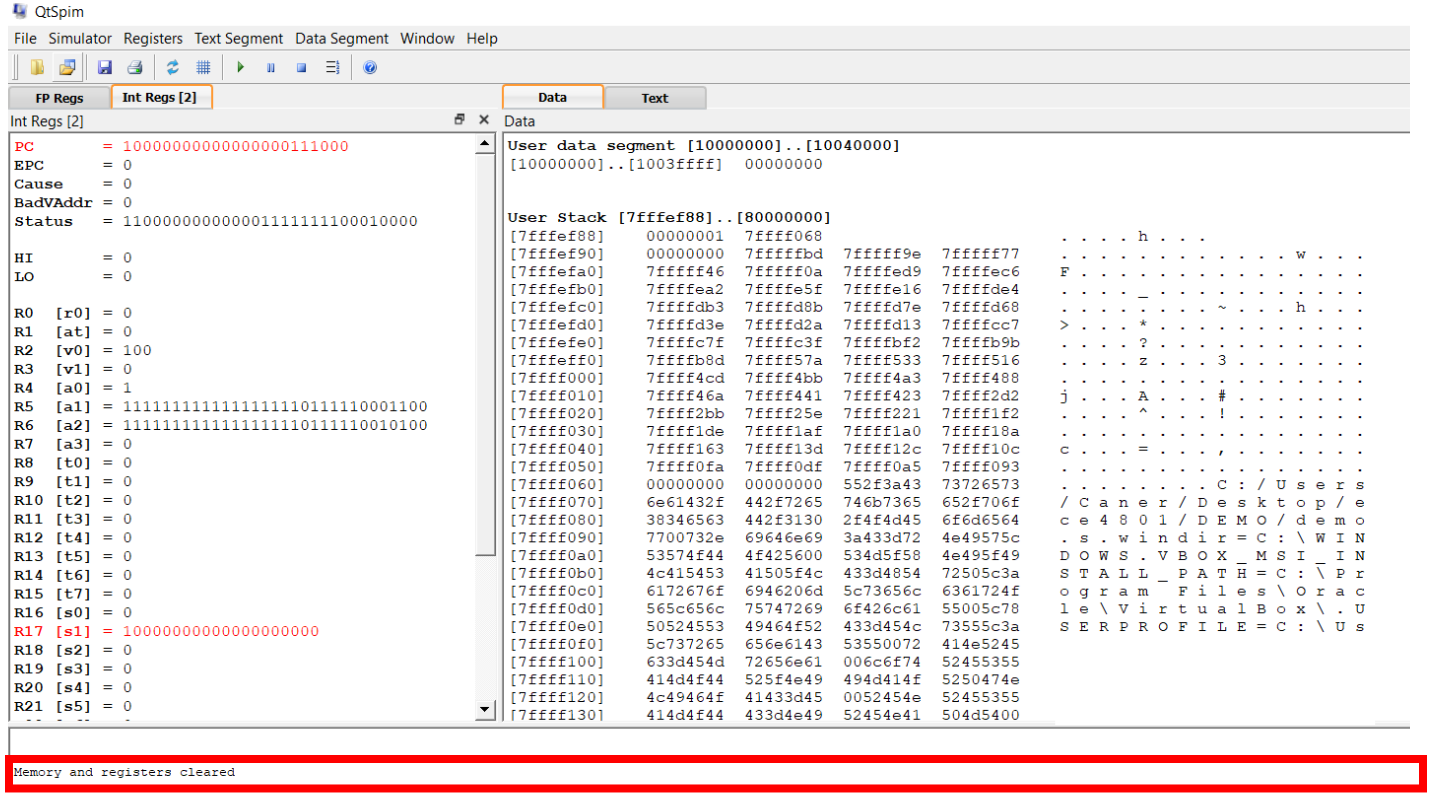
```
User data segment [10000000]..[10040000]
[10000000]..[1003ffff]  00000000


User Stack [7fffef88]..[80000000]
[7fffef88]    00000001  7ffff068                          . . . . h . . .
[7fffef90]    00000000  7ffffbd   7fffff9e  7fffff77      . . . . . . . . . . . . w . . .
[7fffefa0]    7fffff46  7fffff0a  7ffffed9  7ffffec6      F . . . . . . . . . . . . . . .
[7fffefb0]    7ffffea2  7ffffe5f  7ffffe16  7ffffde4      . . . . _ . . . . . . . . . . .
[7fffefc0]    7ffffdb3  7ffffd8b  7ffffd7e  7ffffd68      . . . . . . . . ~ . . . h . . .
[7fffefd0]    7ffffd3e  7ffffd2a  7ffffd13  7ffffcc7      > . . . * . . . . . . . . . . .
[7fffefe0]    7ffffc7f  7ffffc3f  7ffffbf2  7ffffb9b      . . . . ? . . . . . . . . . . .
[7fffeff0]    7ffffb8d  7ffff57a  7ffff533  7ffff516      . . . . z . . . 3 . . . . . . .
[7ffff000]    7ffff4cd  7ffff4bb  7ffff4a3  7ffff488      . . . . . . . . . . . . . . . .
[7ffff010]    7ffff46a  7ffff441  7ffff423  7ffff2d2      j . . . A . . . # . . . . . . .
[7ffff020]    7ffff2bb  7ffff25e  7ffff221  7ffff1f2      . . . . ^ . . . ! . . . . . . .
[7ffff030]    7ffff1de  7ffff1af  7ffff1a0  7ffff18a      . . . . . . . . . . . . . . . .
[7ffff040]    7ffff163  7ffff13d  7ffff12c  7ffff10c      c . . . = . . . , . . . . . . .
[7ffff050]    7ffff0fa  7ffff0df  7ffff0a5  7ffff093      . . . . . . . . . . . . . . . .
[7ffff060]    00000000  00000000  552f3a43  73726573      . . . . . . . . C : / U s e r s
[7ffff070]    6e61432f  442f7265  746b7365  652f706f      / C a n e r / D e s k t o p / e
[7ffff080]    38346563  442f3130  2f4f4d45  6f6d6564      c e 4 8 0 1 / D E M O / d e m o
[7ffff090]    7700732e  69646e69  3a433d72  4e49575c      . s . w i n d i r = C : \ W I N
[7ffff0a0]    53574f44  4f425600  534d5f58  4e495f49      D O W S . V B O X _ M S I _ I N
[7ffff0b0]    4c415453  41505f4c  433d4854  72505c3a      S T A L L _ P A T H = C : \ P r
[7ffff0c0]    6172676f  6946206d  5c73656c  6361724f      o g r a m   F i l e s \ O r a c
[7ffff0d0]    565c656c  75747269  6f426c61  55005c78      l e \ V i r t u a l B o x \ . U
[7ffff0e0]    50524553  49464f52  433d454c  73555c3a      S E R P R O F I L E = C : \ U s
[7ffff0f0]    5c737265  656e6143  53550072  414e5245      e r s \ C a n e r . U S E R N A
[7ffff100]    633d454d  72656e61  006c6f74  52455355      M E = c a n e r t o l . U S E R
[7ffff110]    414d4f44  525f4e49  494d414f  5250474e      D O M A I N _ R O A M I N G P R
[7ffff120]    4c49464f  41433d45  0052454e  52455355      O F I L E = C A N E R . U S E R
[7ffff130]    414d4f44  433d4e49  52454e41  504d5400      D O M A I N = C A N E R . T M P
```

```
Memory and registers cleared
```

SPIM

- 4. The Data tab displays memory addresses and their values in the data and stack segments of the memory.
- 5. The Information Console lists the actions performed by the simulator.



```
PC       = 1000000000000000000111000
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 11000000000000001111111100010000

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 100
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 11111111111111111110111110001100
R6  [a2] = 11111111111111111110111110010100
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] = 100000000000000000000
R18 [s2] = 0
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
```

```
User data segment [10000000]..[10040000]
[10000000]..[1003ffff]  00000000


User Stack [7fffef88]..[80000000]
[7fffef88]    00000001  7ffff068                        . . . . h . . .
[7fffef90]    00000000  7ffffbd  7fffff9e  7fffff77     . . . . . . . . . . . . w . . .
[7fffefa0]    7fffff46  7fffff0a  7ffffed9  7ffffec6    F . . . . . . . . . . . . . . .
[7fffefb0]    7ffffea2  7ffffe5f  7ffffe16  7ffffde4    . . . . _ . . . . . . . . . . .
[7fffefc0]    7ffffdb3  7ffffd8b  7ffffd7e  7ffffd68    . . . . . . . . ~ . . . h . . .
[7fffefd0]    7ffffd3e  7ffffd2a  7ffffd13  7ffffcc7    > . . . * . . . . . . . . . . .
[7fffefe0]    7ffffc7f  7ffffc3f  7ffffbf2  7ffffb9b    . . . . ? . . . . . . . . . . .
[7fffeff0]    7ffffb8d  7ffff57a  7ffff533  7ffff516    . . . . z . . . 3 . . . . . . .
[7ffff000]    7ffff4cd  7ffff4bb  7ffff4a3  7ffff488    . . . . . . . . . . . . . . . .
[7ffff010]    7ffff46a  7ffff441  7ffff423  7ffff2d2    j . . . A . . . # . . . . . . .
[7ffff020]    7ffff2bb  7ffff25e  7ffff221  7ffff1f2    . . . . ^ . . . ! . . . . . . .
[7ffff030]    7ffff1de  7ffff1af  7ffff1a0  7ffff18a    . . . . . . . . . . . . . . . .
[7ffff040]    7ffff163  7ffff13d  7ffff12c  7ffff10c    c . . . = . . . , . . . . . . .
[7ffff050]    7ffff0fa  7ffff0df  7ffff0a5  7ffff093    . . . . . . . . . . . . . . . .
[7ffff060]    00000000  00000000  552f3a43  73726573    . . . . . . . . C : / U s e r s
[7ffff070]    6e61432f  442f7265  746b7365  652f706f    / C a n e r / D e s k t o p / e
[7ffff080]    38346563  442f3130  2f4f4d45  6f6d6564    c e 4 8 0 1 / D E M O / d e m o
[7ffff090]    7700732e  69646e69  3a433d72  4e49575c    . s . w i n d i r = C : \ W I N
[7ffff0a0]    53574f44  4f425600  534d5f58  4e495f49    D O W S . V B O X _ M S I _ I N
[7ffff0b0]    4c415453  41505f4c  433d4854  72505c3a    S T A L L _ P A T H = C : \ P r
[7ffff0c0]    6172676f  6946206d  5c73656c  6361724f    o g r a m   F i l e s \ O r a c
[7ffff0d0]    565c656c  75747269  6f426c61  55005c78    l e \ V i r t u a l B o x \ . U
[7ffff0e0]    50524553  49464f52  433d454c  73555c3a    S E R P R O F I L E = C : \ U s
[7ffff0f0]    5c737265  656e6143  53550072  414e5245
[7ffff100]    633d454d  72656e61  006c6f74  52455355
[7ffff110]    414d4f44  525f4e49  494d414f  5250474e
[7ffff120]    4c49464f  41433d45  0052454e  52455355
[7ffff130]    414d4f44  433d4e49  52454e41  504d5400
```

```
Memory and registers cleared
```

Running a program in QtSpim

- 1. Use a text editor to create your program yyyyyy.s
- 2. Click on the “load” button and open yyyyyy.s
- 3. You can then run the program by simply pressing the “run” (play) button – all instructions will be executed, and the final contents of the memory and the register file will be reflected in the QtSpim window.

Debugging 1/2

- Suppose your program does not do what you expect. What can you do?
- QtSpim has two features that help debug your program.
- The first, and perhaps the most useful, is single-stepping, which allows you to run your program an instruction at a time. The single stepping icon can be found in the toolbar. Every time you do single stepping, QtSpim will execute one instruction and update its display, so that you can see what the instruction changed in the registers or memory.
- What do you do if your program runs for a long time before the bug arises?
- You could single-step until you get to the bug, but that can take a long time.

Debugging 2/2

- A better alternative is to use a breakpoint , which tells QtSpim to stop your program immediately before it executes a particular instruction. When QtSpim is about to execute the instruction where there is a breakpoint, it asks for continue, single stepping or abort.
- Single-stepping and setting breakpoints will probably help you find a bug in your program quickly.
- How do you fix it?
- Go back to the editor that you used to create your program and change it.
- Click on the Reinitialize simulator tab in the toolsbar and load the source file again.

Generally Useful Information

- You can access all of the commands via the “File” and “Simulator” menus as well.
- When examining register or memory data, you can view the data in binary, hex, or decimal format. Just use the “Register” pull down menu to select.
- Kernel Text and Kernel Data may not be necessary to be viewed all the times, you can unselect them by unselecting “Kernel Text” in the “Text Segment” pull down menu and unselecting
- “Kernel Data” in the “Data Segment” pull down menu.
- You can set breakpoints in your code simply by right clicking on an instruction in the Text tab.
- To view memory data, simply click on the Data tab.
- By right clicking on a register file value or memory address value, you can change its contents dynamically.
- To open the console window or any other window, click the “Window” tab and click one of them.