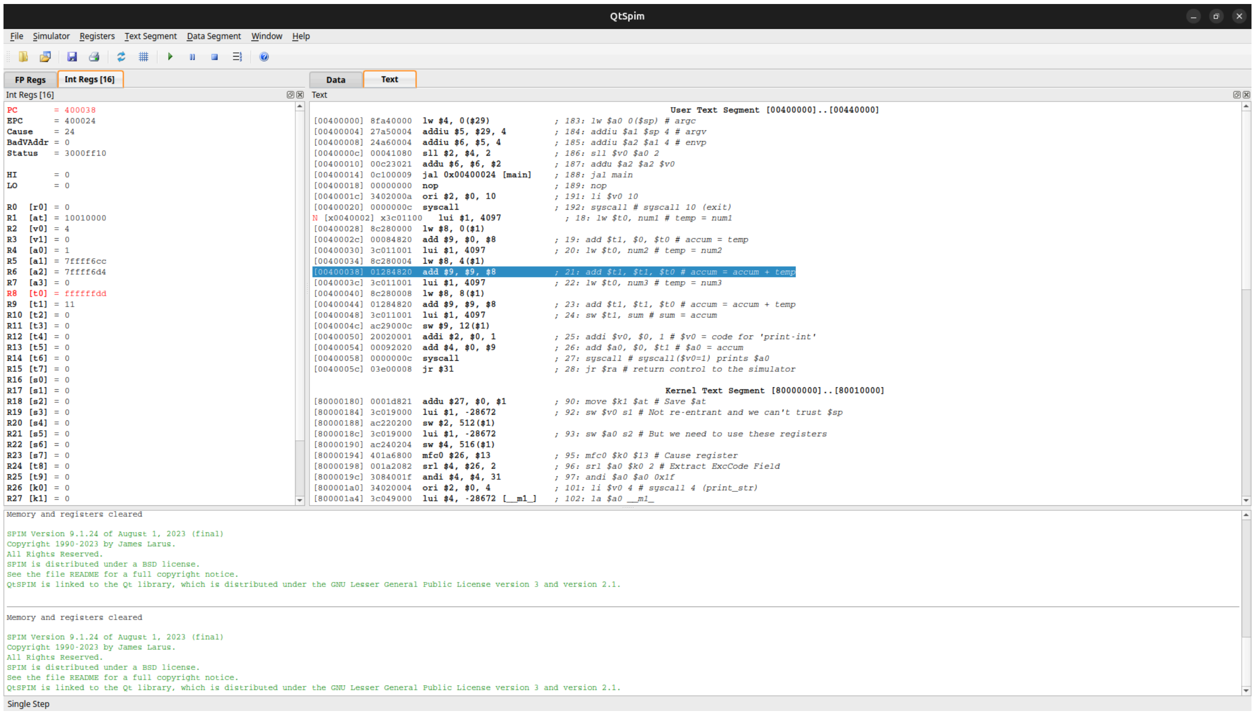
270 Lab 5

Screenshots #7

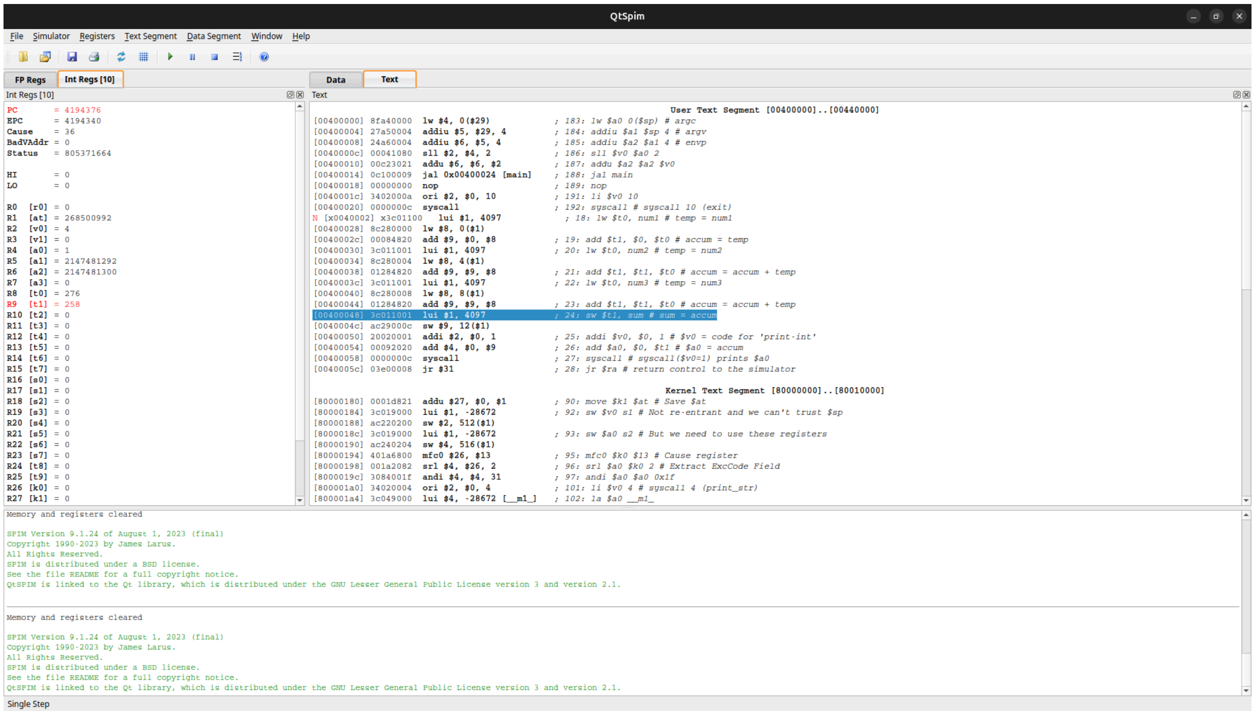


```
PC       = 400038
EPC      = 400024
Cause    = 24
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

R0   [r0] = 0
R1   [at] = 10010000
R2   [v0] = 4
R3   [v1] = 0
R4   [a0] = 1
R5   [a1] = 7ffff6cc
R6   [a2] = 7ffff6d4
R7   [a3] = 0
R8   [t0] = ffffffdd
R9   [t1] = 11
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
```

```
                                                  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)
N [x0040002] x3c01100   lui $1, 4097            ; 18: lw $t0, num1 # temp = num1
[00400028] 8c280000  lw $8, 0($1)
[0040002c] 00084820  add $9, $0, $8           ; 19: add $t1, $0, $t0 # accum = temp
[00400030] 3c011001  lui $1, 4097             ; 20: lw $t0, num2 # temp = num2
[00400034] 8c280004  lw $8, 4($1)
[00400038] 01284820  add $9, $9, $8           ; 21: add $t1, $t1, $t0 # accum = accum + temp
[0040003c] 3c011001  lui $1, 4097             ; 22: lw $t0, num3 # temp = num3
[00400040] 8c280008  lw $8, 8($1)
[00400044] 01284820  add $9, $9, $8           ; 23: add $t1, $t1, $t0 # accum = accum + temp
[00400048] 3c011001  lui $1, 4097             ; 24: sw $t1, sum # sum = accum
[0040004c] ac29000c  sw $9, 12($1)
[00400050] 20020001  addi $2, $0, 1           ; 25: addi $v0, $0, 1 # $v0 = code for 'print-int'
[00400054] 00092020  add $4, $0, $9           ; 26: add $a0, $0, $t1 # $a0 = accum
[00400058] 0000000c  syscall                  ; 27: syscall # syscall($v0=1) prints $a0
[0040005c] 03e00008  jr $31                   ; 28: jr $ra # return control to the simulator

                                                  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_
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.

Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.
```

Registers $t0 and $t1 after line 19 is sum.asm. (First number added)



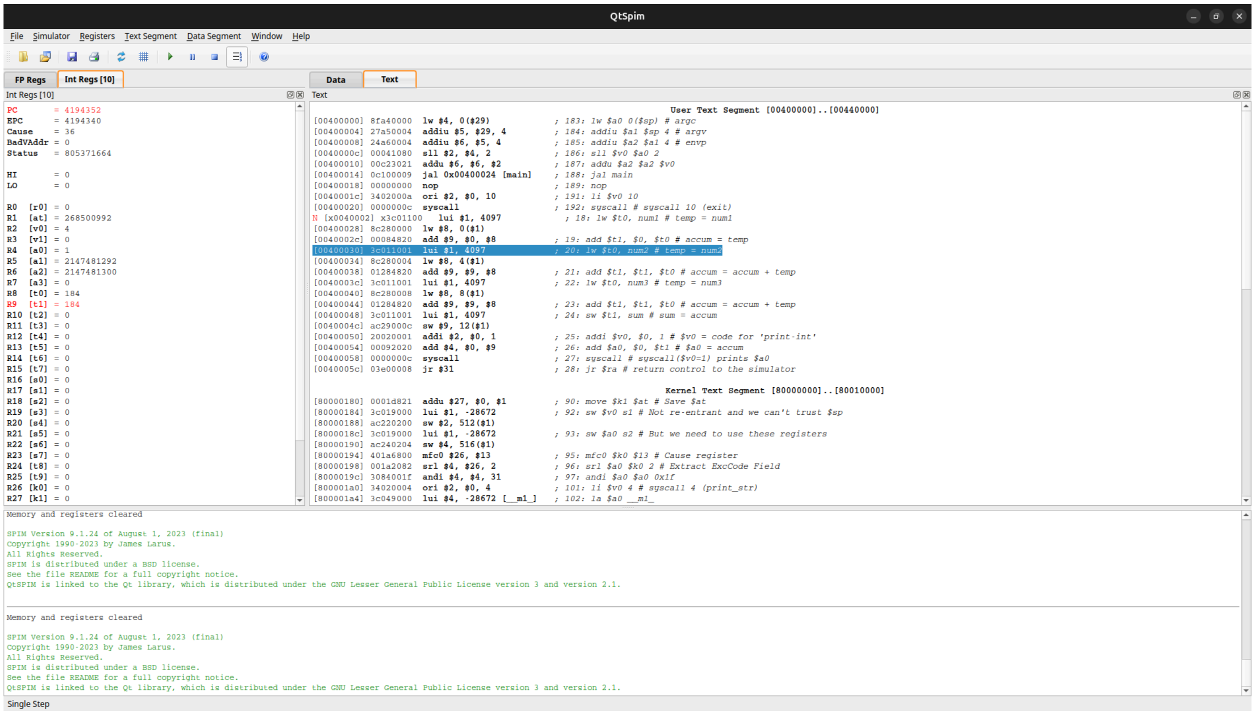
```
PC       = 4194376
EPC      = 4194340
Cause    = 36
BadVAddr = 0
Status   = 805371664

HI       = 0
LO       = 0

R0   [r0] = 0
R1   [at] = 268500992
R2   [v0] = 4
R3   [v1] = 0
R4   [a0] = 1
R5   [a1] = 2147481292
R6   [a2] = 2147481300
R7   [a3] = 0
R8   [t0] = 276
R9   [t1] = 258
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
```

```
                                                  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)
N [x0040002] x3c01100   lui $1, 4097            ; 18: lw $t0, num1 # temp = num1
[00400028] 8c280000  lw $8, 0($1)
[0040002c] 00084820  add $9, $0, $8           ; 19: add $t1, $0, $t0 # accum = temp
[00400030] 3c011001  lui $1, 4097             ; 20: lw $t0, num2 # temp = num2
[00400034] 8c280004  lw $8, 4($1)
[00400038] 01284820  add $9, $9, $8           ; 21: add $t1, $t1, $t0 # accum = accum + temp
[0040003c] 3c011001  lui $1, 4097             ; 22: lw $t0, num3 # temp = num3
[00400040] 8c280008  lw $8, 8($1)
[00400044] 01284820  add $9, $9, $8           ; 23: add $t1, $t1, $t0 # accum = accum + temp
[00400048] 3c011001  lui $1, 4097             ; 24: sw $t1, sum # sum = accum
[0040004c] ac29000c  sw $9, 12($1)
[00400050] 20020001  addi $2, $0, 1           ; 25: addi $v0, $0, 1 # $v0 = code for 'print-int'
[00400054] 00092020  add $4, $0, $9           ; 26: add $a0, $0, $t1 # $a0 = accum
[00400058] 0000000c  syscall                  ; 27: syscall # syscall($v0=1) prints $a0
[0040005c] 03e00008  jr $31                   ; 28: jr $ra # return control to the simulator

                                                  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_
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.

Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.
```

Registers $t0 and $t1 after line 24 is sum.asm. (Second and third number added)

Screenshots #8

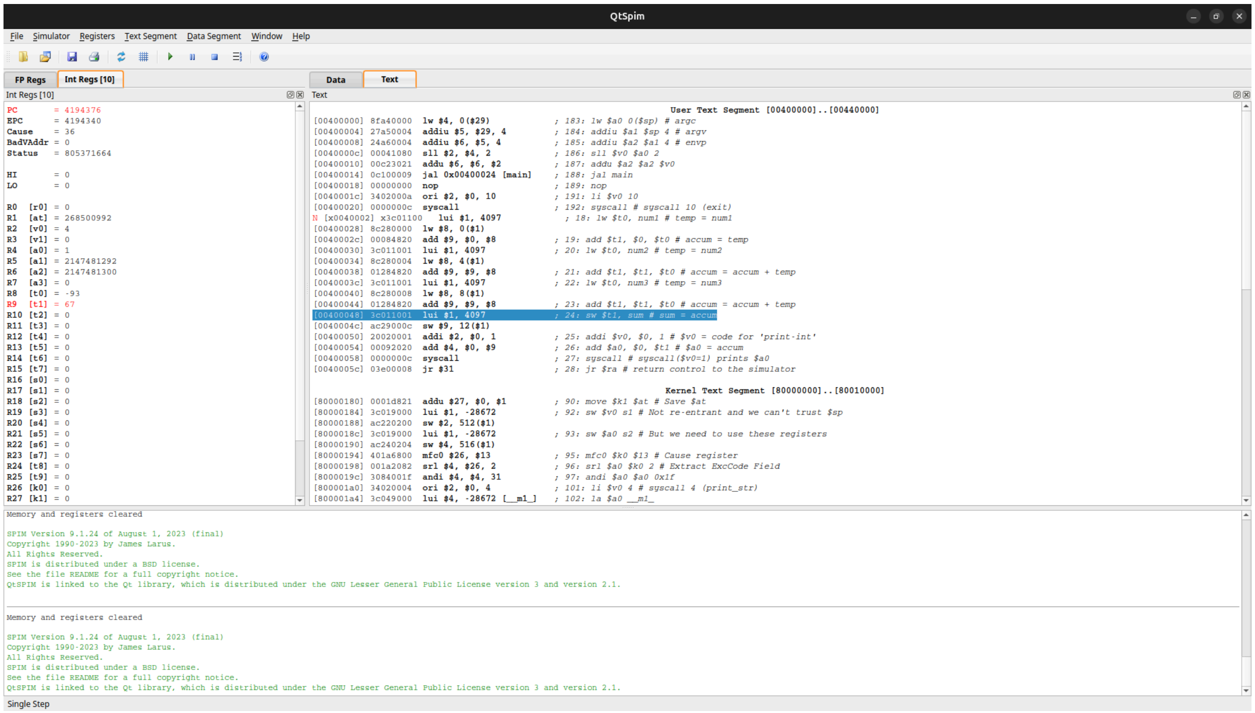


```
PC       = 4194352
EPC      = 4194340
Cause    = 36
BadVAddr = 0
Status   = 805371664

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 268500992
R2  [v0] = 4
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 2147481292
R6  [a2] = 2147481300
R7  [a3] = 0
R8  [t0] = 184
R9  [t1] = 184
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
```

```
                                              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)
N [x0040002] x3c01100   lui $1, 4097            ; 18: lw $t0, num1 # temp = num1
[00400028] 8c280000  lw $8, 0($1)
[0040002c] 00084820  add $9, $0, $8           ; 19: add $t1, $0, $t0 # accum = temp
[00400030] 3c011001  lui $1, 4097             ; 20: lw $t0, num2 # temp = num2
[00400034] 8c280004  lw $8, 4($1)
[00400038] 01284820  add $9, $9, $8           ; 21: add $t1, $t1, $t0 # accum = accum + temp
[0040003c] 3c011001  lui $1, 4097             ; 22: lw $t0, num3 # temp = num3
[00400040] 8c280008  lw $8, 8($1)
[00400044] 01284820  add $9, $9, $8           ; 23: add $t1, $t1, $t0 # accum = accum + temp
[00400048] 3c011001  lui $1, 4097             ; 24: sw $t1, sum # sum = accum
[0040004c] ac29000c  sw $9, 12($1)
[00400050] 20020001  addi $2, $0, 1           ; 25: addi $v0, $0, 1 # $v0 = code for 'print-int'
[00400054] 00092020  add $4, $0, $9           ; 26: add $a0, $0, $t1 # $a0 = accum
[00400058] 0000000c  syscall                  ; 27: syscall # syscall($v0=1) prints $a0
[0040005c] 03e00008  jr $31                   ; 28: jr $ra # return control to the simulator

                                              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_
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.

Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.
```

Registers $t0 and $t1 after line 19 is sum2.asm. (First number added)



```
PC       = 4194376
EPC      = 4194340
Cause    = 36
BadVAddr = 0
Status   = 805371664

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 268500992
R2  [v0] = 4
R3  [v1] = 0
R4  [a0] = 1
R5  [a1] = 2147481292
R6  [a2] = 2147481300
R7  [a3] = 0
R8  [t0] = -93
R9  [t1] = 67
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
```

```
                                              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)
N [x0040002] x3c01100   lui $1, 4097            ; 18: lw $t0, num1 # temp = num1
[00400028] 8c280000  lw $8, 0($1)
[0040002c] 00084820  add $9, $0, $8           ; 19: add $t1, $0, $t0 # accum = temp
[00400030] 3c011001  lui $1, 4097             ; 20: lw $t0, num2 # temp = num2
[00400034] 8c280004  lw $8, 4($1)
[00400038] 01284820  add $9, $9, $8           ; 21: add $t1, $t1, $t0 # accum = accum + temp
[0040003c] 3c011001  lui $1, 4097             ; 22: lw $t0, num3 # temp = num3
[00400040] 8c280008  lw $8, 8($1)
[00400044] 01284820  add $9, $9, $8           ; 23: add $t1, $t1, $t0 # accum = accum + temp
[00400048] 3c011001  lui $1, 4097             ; 24: sw $t1, sum # sum = accum
[0040004c] ac29000c  sw $9, 12($1)
[00400050] 20020001  addi $2, $0, 1           ; 25: addi $v0, $0, 1 # $v0 = code for 'print-int'
[00400054] 00092020  add $4, $0, $9           ; 26: add $a0, $0, $t1 # $a0 = accum
[00400058] 0000000c  syscall                  ; 27: syscall # syscall($v0=1) prints $a0
[0040005c] 03e00008  jr $31                   ; 28: jr $ra # return control to the simulator

                                              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_
```

```
Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.

Memory and registers cleared

SPIM Version 9.1.24 of August 1, 2023 (final)
Copyright 1990-2023 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.
```

Registers $t0 and $t1 after line 23 is sum2.asm. (Second and third number added)

Questions:

1. The immediate 0x22220000 will be loaded into $t6.
2. The instruction lui means to load upper immediate which shifts the immediate value to the left and add zeros to the right to make the value 32-bits long.
3. Since immediate values only have 16-bits in the instruction word, we need to initialize the most significant 16-bits to be added to the least significant 16-bits.
4. The last jump in the assembly program returns control to the simulator. The jump in QtSpim jumps to the function main address because this is the start of the program.