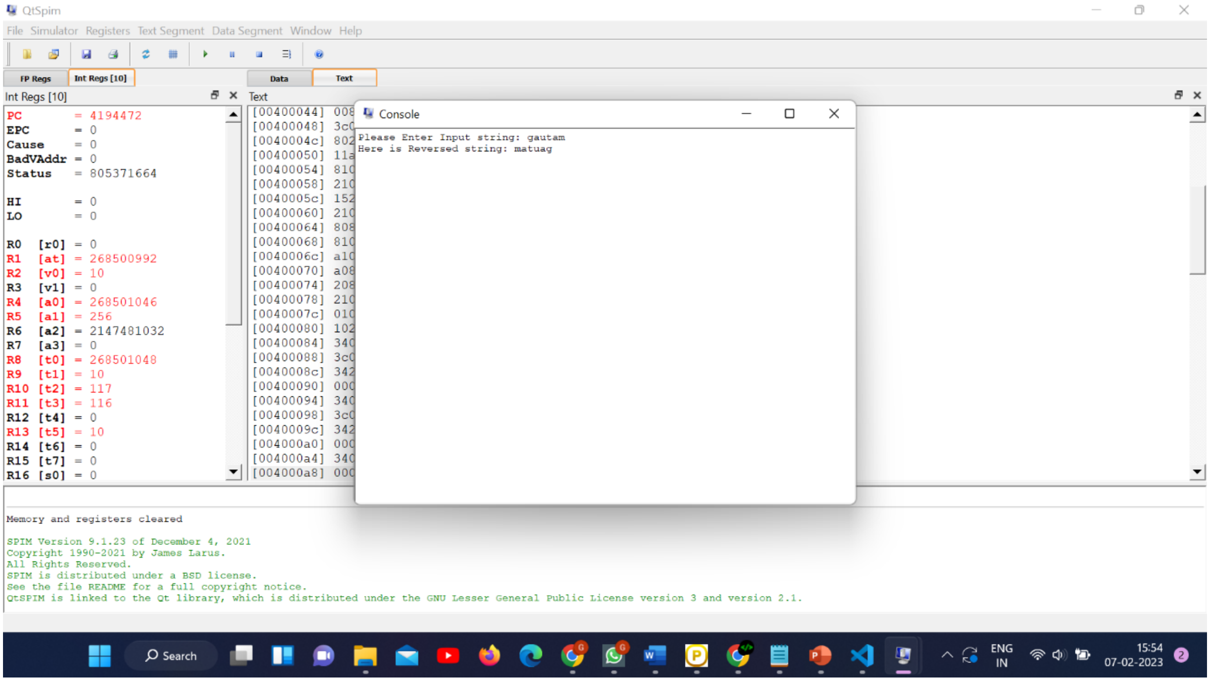
CS 211
LAB 3: Exceptions in MIPS
Name – Gautam Kumar Mahar
Roll No. – 2103114
Branch – Computer Science Engineering

PART B

In this part, you will write MIPS assembly language programs to do the following exercises.

1. **Reverse a string entered by a user. (Hint: Ask the user to enter a string. After reading the string in a buffer, copy it in reversed order to a second buffer. Write out the reversed string.)**

In Decimal –



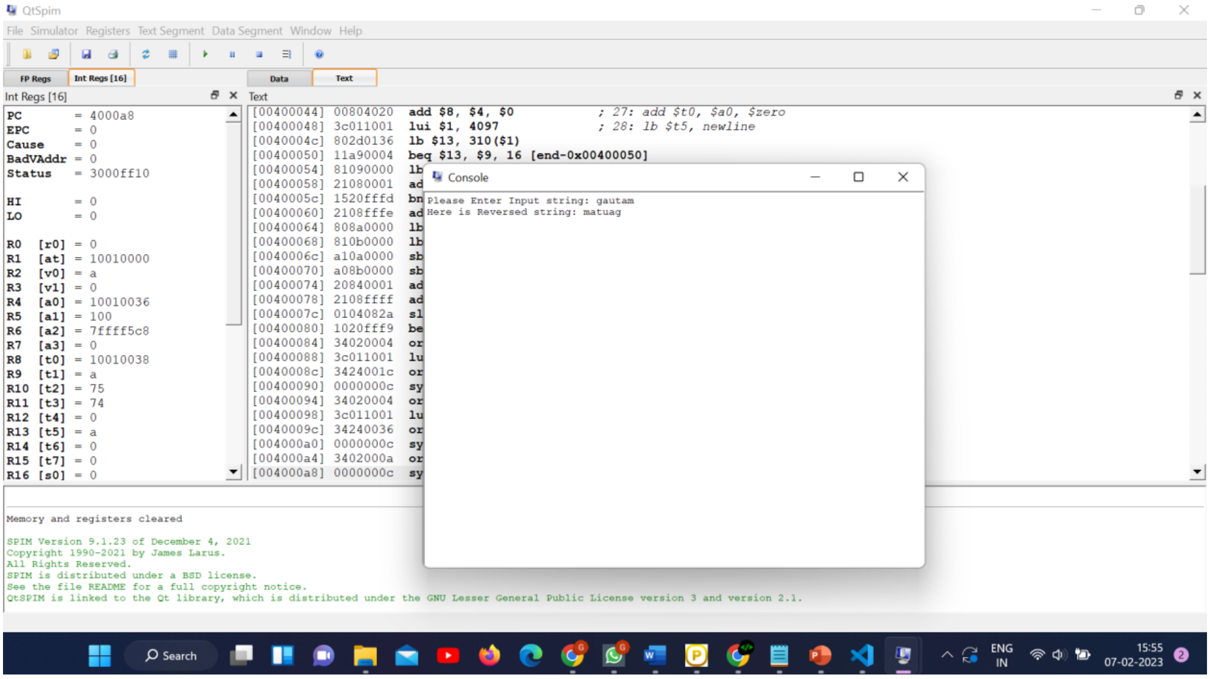
```
PC        = 4194472
EPC       = 0
Cause     = 0
BadVAddr  = 0
Status    = 805371664

HI        = 0
LO        = 0

R0  [r0] = 0
R1  [at] = 268500992
R2  [v0] = 10
R3  [v1] = 0
R4  [a0] = 268501046
R5  [a1] = 256
R6  [a2] = 2147481032
R7  [a3] = 0
R8  [t0] = 268501048
R9  [t1] = 10
R10 [t2] = 117
R11 [t3] = 116
R12 [t4] = 0
R13 [t5] = 10
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
```

```
[00400044] 008
[00400048] 3c0
[0040004c] 802
[00400050] 11a
[00400054] 810
[00400058] 210
[0040005c] 152
[00400060] 210
[00400064] 808
[00400068] 810
[0040006c] a10
[00400070] a08
[00400074] 208
[00400078] 210
[0040007c] 010
[00400080] 102
[00400084] 340
[00400088] 3c0
[0040008c] 342
[00400090] 000
[00400094] 340
[00400098] 3c0
[0040009c] 342
[004000a0] 000
[004000a4] 340
[004000a8] 000
```

```
Please Enter Input string: gautam
Here is Reversed string: matuag
```

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
```

In Hexa Decimal –



```
PC        = 4000a8
EPC       = 0
Cause     = 0
BadVAddr  = 0
Status    = 3000ff10

HI        = 0
LO        = 0

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 10010036
R5  [a1] = 100
R6  [a2] = 7ffff5c8
R7  [a3] = 0
R8  [t0] = 10010038
R9  [t1] = a
R10 [t2] = 75
R11 [t3] = 74
R12 [t4] = 0
R13 [t5] = a
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0
```

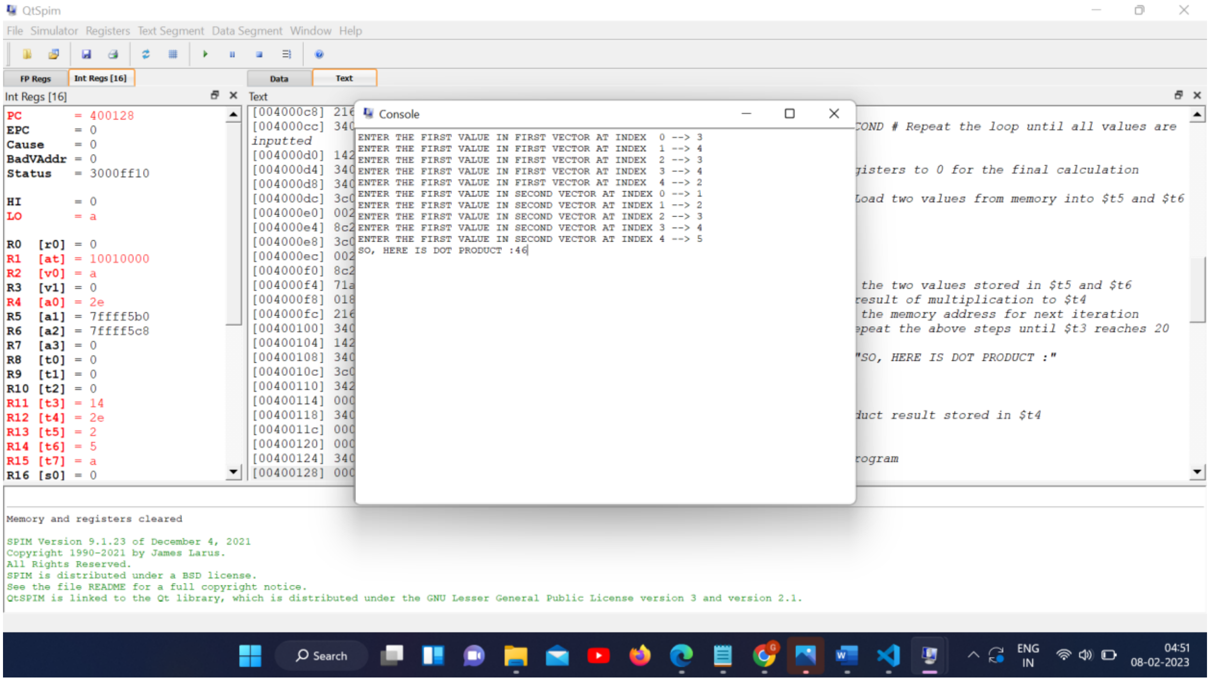
```
[00400044] 00804020  add $8, $4, $0              ; 27: add $t0, $a0, $zero
[00400048] 3c011001  lui $1, 4097                ; 28: lb $t5, newline
[0040004c] 802d0136  lb $13, 310($1)
[00400050] 11a90004  beq $13, $9, 16 [end-0x00400050]
[00400054] 81090000  lb
[00400058] 21080001  ad
[0040005c] 1520fffd  bn
[00400060] 2108fffe  ad
[00400064] 808a0000  lb
[00400068] 810b0000  lb
[0040006c] a10a0000  sb
[00400070] a08b0000  sb
[00400074] 20840001  ad
[00400078] 2108ffff  ad
[0040007c] 0104082a  sl
[00400080] 1020fff9  be
[00400084] 34020004  or
[00400088] 3c011001  lu
[0040008c] 3424001c  or
[00400090] 0000000c  sy
[00400094] 34020004  or
[00400098] 3c011001  lu
[0040009c] 34240036  or
[004000a0] 0000000c  sy
[004000a4] 3402000a  or
[004000a8] 0000000c  sy
```

```
Please Enter Input string: gautam
Here is Reversed string: matuag
```

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
```

2. **Compute the dot product of two vectors each of length 5. Ask the user to enter the value of each element of the two vectors. Display the dot product. (Hint: The dot product of two vectors is sum of the product of the corresponding elements. For example, (1,2,3) dot (4,5,6) is 1*4+2*5+3*6 = 32)**

In Hexa Decimal –



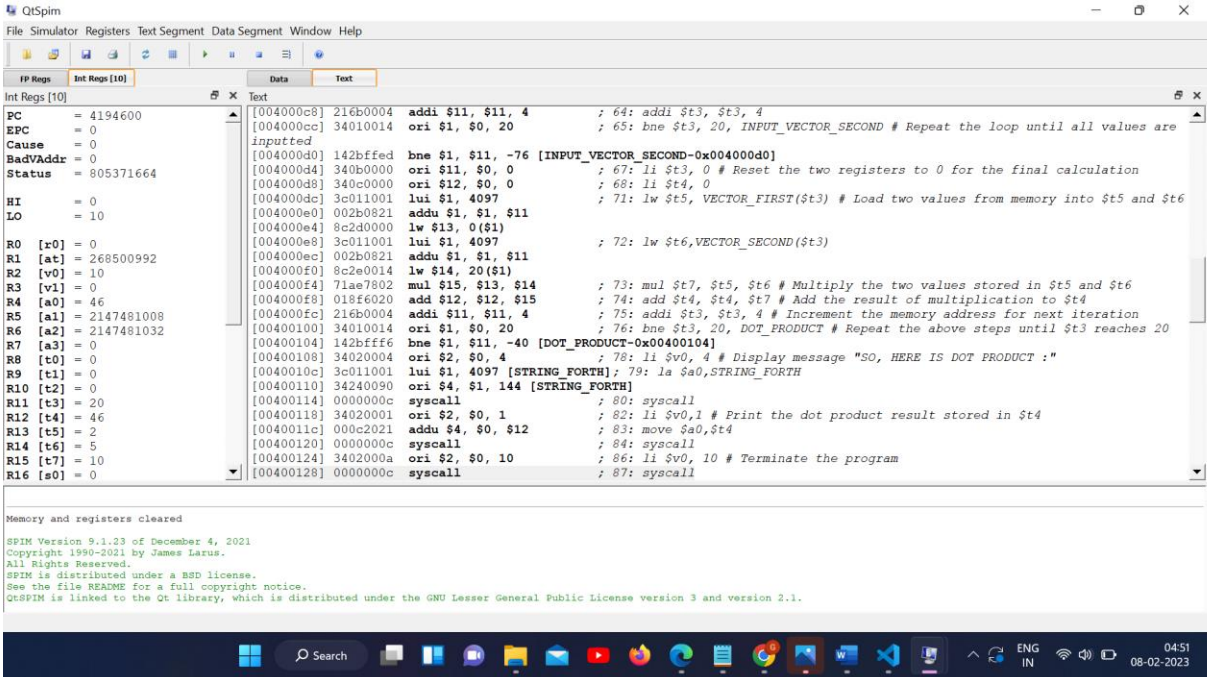
```
PC        = 400128
EPC       = 0
Cause     = 0
BadVAddr  = 0
Status    = 3000ff10

HI        = 0
LO        = a

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 2e
R5  [a1] = 7ffff5b0
R6  [a2] = 7ffff5c8
R7  [a3] = 0
R8  [t0] = 0
R9  [t1] = 0
R10 [t2] = 0
R11 [t3] = 14
R12 [t4] = 2e
R13 [t5] = 2
R14 [t6] = 5
R15 [t7] = a
R16 [s0] = 0
```

```
[004000c8] 216
[004000cc] 340        COND # Repeat the loop until all values are
inputted
[004000d0] 142
[004000d4] 340        gisters to 0 for the final calculation
[004000d8] 340
[004000dc] 3c0        Load two values from memory into $t5 and $t6
[004000e0] 002
[004000e4] 8c2
[004000e8] 3c0
[004000ec] 002
[004000f0] 8c2
[004000f4] 71a         the two values stored in $t5 and $t6
[004000f8] 018        result of multiplication to $t4
[004000fc] 216         the memory address for next iteration
[00400100] 340        epeat the above steps until $t3 reaches 20
[00400104] 142
[00400108] 340        "SO, HERE IS DOT PRODUCT :"
[0040010c] 3c0
[00400110] 342
[00400114] 000
[00400118] 340        duct result stored in $t4
[0040011c] 000
[00400120] 000
[00400124] 340        rogram
[00400128] 000
```

```
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  0 --> 3
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  1 --> 4
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  2 --> 3
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  3 --> 4
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  4 --> 2
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 0 --> 1
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 1 --> 2
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 2 --> 3
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 3 --> 4
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 4 --> 5
SO, HERE IS DOT PRODUCT :46
```

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
```

In Decimal –

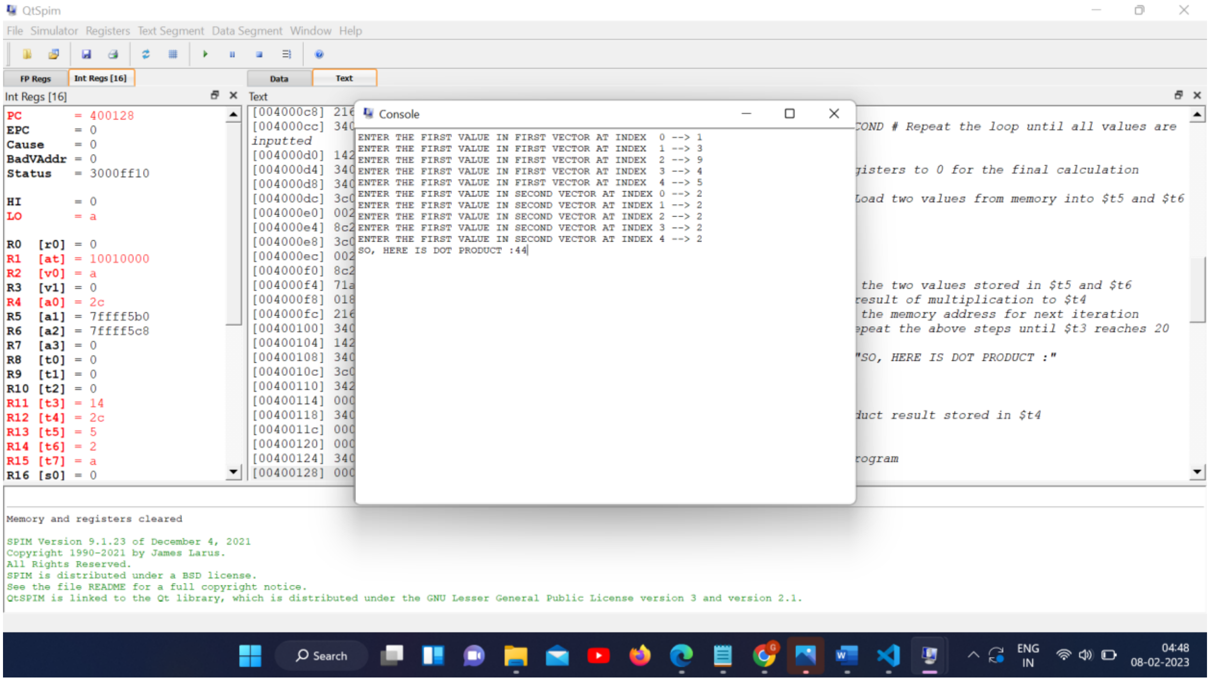


```
PC       = 4194600
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 805371664

HI       = 0
LO       = 10

R0  [r0] = 0
R1  [at] = 268500992
R2  [v0] = 10
R3  [v1] = 0
R4  [a0] = 46
R5  [a1] = 2147481008
R6  [a2] = 2147481032
R7  [a3] = 0
R8  [t0] = 0
R9  [t1] = 0
R10 [t2] = 0
R11 [t3] = 20
R12 [t4] = 46
R13 [t5] = 2
R14 [t6] = 5
R15 [t7] = 10
R16 [s0] = 0
```

```
[004000c8] 216b0004  addi $11, $11, 4            ; 64: addi $t3, $t3, 4
[004000cc] 34010014  ori $1, $0, 20              ; 65: bne $t3, 20, INPUT_VECTOR_SECOND # Repeat the loop until all values are
inputted
[004000d0] 142bffed  bne $1, $11, -76 [INPUT_VECTOR_SECOND-0x004000d0]
[004000d4] 340b0000  ori $11, $0, 0              ; 67: li $t3, 0 # Reset the two registers to 0 for the final calculation
[004000d8] 340c0000  ori $12, $0, 0              ; 68: li $t4, 0
[004000dc] 3c011001  lui $1, 4097                ; 71: lw $t5, VECTOR_FIRST($t3) # Load two values from memory into $t5 and $t6
[004000e0] 002b0821  addu $1, $1, $11
[004000e4] 8c2d0000  lw $13, 0($1)
[004000e8] 3c011001  lui $1, 4097                ; 72: lw $t6,VECTOR_SECOND($t3)
[004000ec] 002b0821  addu $1, $1, $11
[004000f0] 8c2e0014  lw $14, 20($1)
[004000f4] 71ae7802  mul $15, $13, $14           ; 73: mul $t7, $t5, $t6 # Multiply the two values stored in $t5 and $t6
[004000f8] 018f6020  add $12, $12, $15           ; 74: add $t4, $t4, $t7 # Add the result of multiplication to $t4
[004000fc] 216b0004  addi $11, $11, 4            ; 75: addi $t3, $t3, 4 # Increment the memory address for next iteration
[00400100] 34010014  ori $1, $0, 20              ; 76: bne $t3, 20, DOT_PRODUCT # Repeat the above steps until $t3 reaches 20
[00400104] 142bfff6  bne $1, $11, -40 [DOT_PRODUCT-0x00400104]
[00400108] 34020004  ori $2, $0, 4               ; 78: li $v0, 4 # Display message "SO, HERE IS DOT PRODUCT :"
[0040010c] 3c011001  lui $1, 4097 [STRING_FORTH]; 79: la $a0,STRING_FORTH
[00400110] 34240090  ori $4, $1, 144 [STRING_FORTH]
[00400114] 0000000c  syscall                     ; 80: syscall
[00400118] 34020001  ori $2, $0, 1               ; 82: li $v0,1 # Print the dot product result stored in $t4
[0040011c] 000c2021  addu $4, $0, $12            ; 83: move $a0,$t4
[00400120] 0000000c  syscall                     ; 84: syscall
[00400124] 3402000a  ori $2, $0, 10              ; 86: li $v0, 10 # Terminate the program
[00400128] 0000000c  syscall                     ; 87: syscall
```

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
```

Another Test Case –



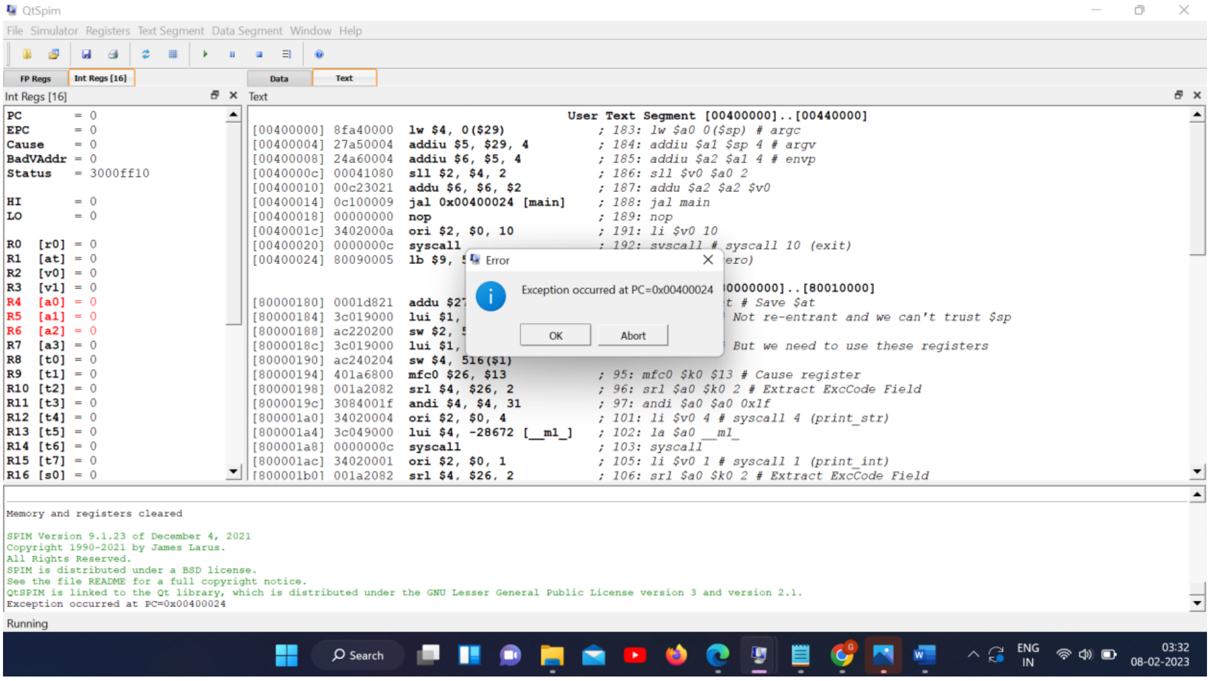
```
PC       = 400128
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = a

R0  [r0] = 0
R1  [at] = 10010000
R2  [v0] = a
R3  [v1] = 0
R4  [a0] = 2c
R5  [a1] = 7ffff5b0
R6  [a2] = 7ffff5c8
R7  [a3] = 0
R8  [t0] = 0
R9  [t1] = 0
R10 [t2] = 0
R11 [t3] = 14
R12 [t4] = 2c
R13 [t5] = 5
R14 [t6] = 2
R15 [t7] = a
R16 [s0] = 0
```

```
[004000c8] 216
[004000cc] 340                    COND # Repeat the loop until all values are
inputted
[004000d0] 142
[004000d4] 340                    gisters to 0 for the final calculation
[004000d8] 340
[004000dc] 3c0                    Load two values from memory into $t5 and $t6
[004000e0] 002
[004000e4] 8c2
[004000e8] 3c0
[004000ec] 002
[004000f0] 8c2
[004000f4] 71a                     the two values stored in $t5 and $t6
[004000f8] 018                    esult of multiplication to $t4
[004000fc] 216                     the memory address for next iteration
[00400100] 340                    epeat the above steps until $t3 reaches 20
[00400104] 142
[00400108] 340                    "SO, HERE IS DOT PRODUCT :"
[0040010c] 3c0
[00400110] 342
[00400114] 000
[00400118] 340                    duct result stored in $t4
[0040011c] 000
[00400120] 000
[00400124] 340                    rogram
[00400128] 000
```

```
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  0 --> 1
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  1 --> 3
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  2 --> 9
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  3 --> 4
ENTER THE FIRST VALUE IN FIRST VECTOR AT INDEX  4 --> 5
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 0 --> 2
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 1 --> 2
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 2 --> 2
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 3 --> 2
ENTER THE FIRST VALUE IN SECOND VECTOR AT INDEX 4 --> 2
SO, HERE IS DOT PRODUCT :44
```

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
```

3) **Use lb $t1, 5($zero) to cause an exception when attempting to load a byte from address 5. What is the address of the lb instruction in your program? What is the value of the cause register, the exception code, the vaddr, and the epc when the exception occurs?**

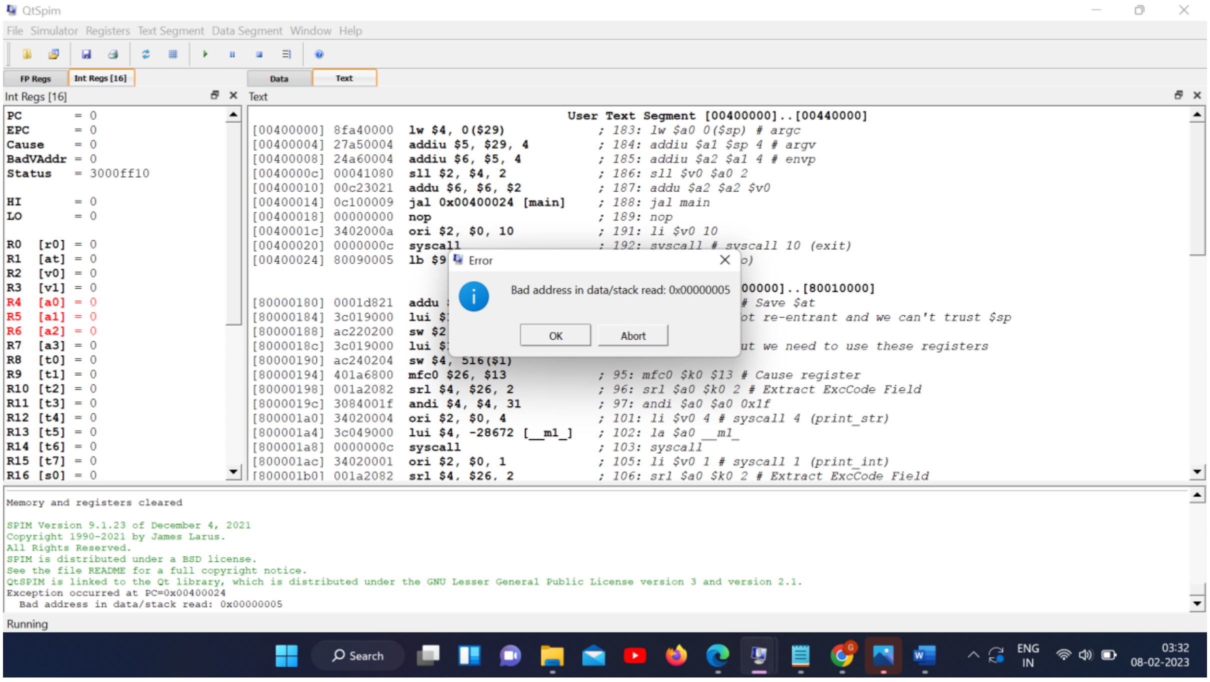


```
PC       = 0
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

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

```
                                  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] 80090005  lb $9, 5                                ero)

                                                0000000]..[80010000]
[80000180] 0001d821  addu $27                               t # Save $at
[80000184] 3c019000  lui $1,                                 Not re-entrant and we can't trust $sp
[80000188] ac220200  sw $2, 5
[8000018c] 3c019000  lui $1,                                 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
```

Exception occurred at PC=0x00400024

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
Exception occurred at PC=0x00400024
```

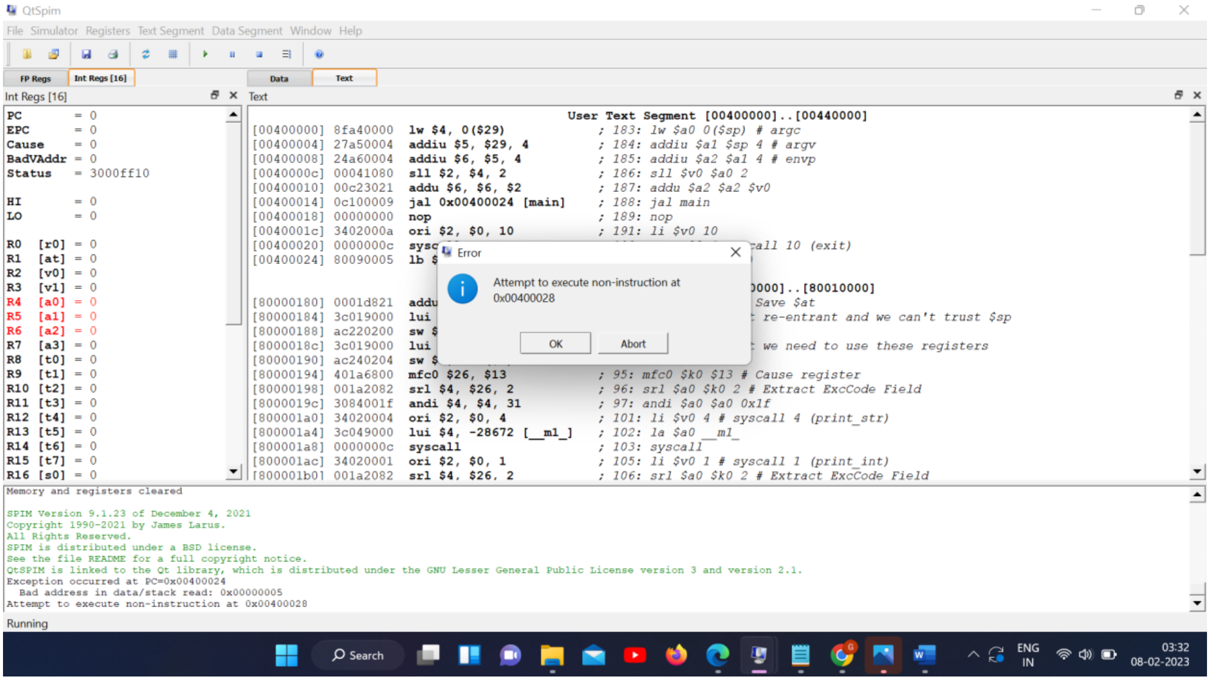


```
PC       = 0
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

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

```
                                  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] 80090005  lb $9                                     o)

                                                  00000]..[80010000]
[80000180] 0001d821  addu                                     # Save $at
[80000184] 3c019000  lui $                                  ot re-entrant and we can't trust $sp
[80000188] ac220200  sw $2
[8000018c] 3c019000  lui $                                  ut 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
```

Bad address in data/stack read: 0x00000005

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
Exception occurred at PC=0x00400024
  Bad address in data/stack read: 0x00000005
```



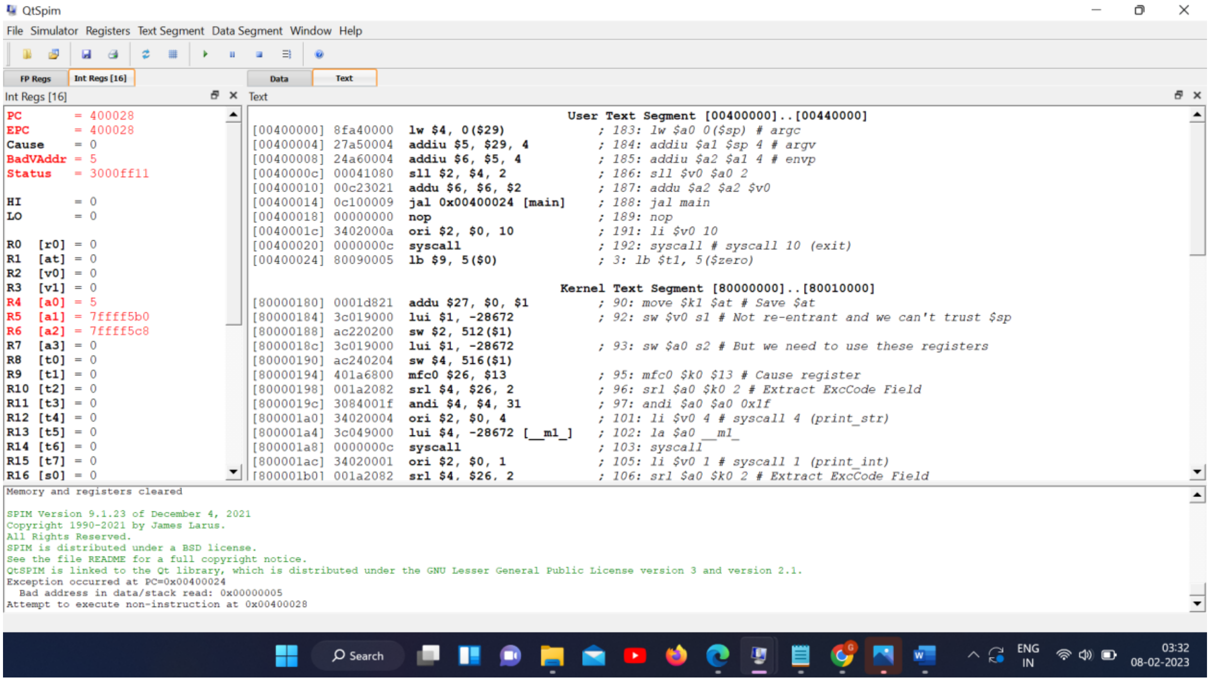
```
PC       = 0
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 3000ff10

HI       = 0
LO       = 0

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

```
                              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  sysc                               call 10 (exit)
[00400024] 80090005  lb $

                                                   0000]..[80010000]
[80000180] 0001d821  addu                          Save $at
[80000184] 3c019000  lui                           t re-entrant and we can't trust $sp
[80000188] ac220200  sw $
[8000018c] 3c019000  lui                           t we need to use these registers
[80000190] ac240204  sw $
[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
```

Attempt to execute non-instruction at 0x00400028

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
Exception occurred at PC=0x00400024
  Bad address in data/stack read: 0x00000005
Attempt to execute non-instruction at 0x00400028
```

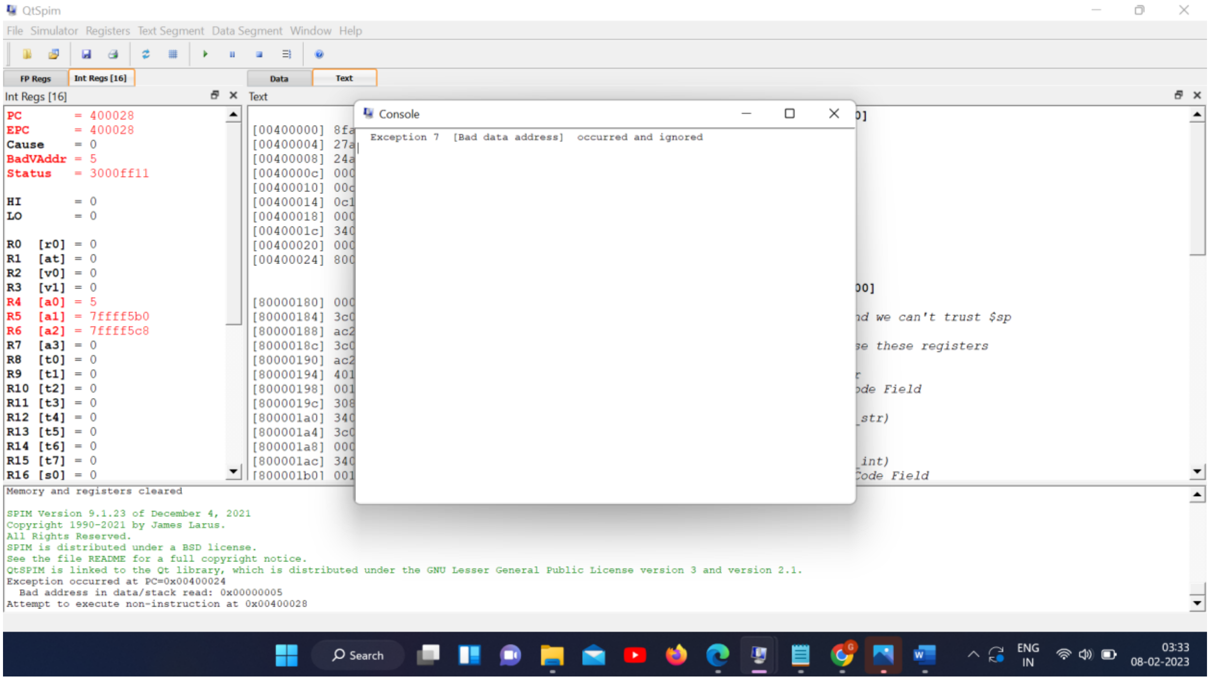
Running



```
PC       = 400028
EPC      = 400028
Cause    = 0
BadVAddr = 5
Status   = 3000ff11

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 5
R5  [a1] = 7ffff5b0
R6  [a2] = 7ffff5c8
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
```

```
                              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] 80090005  lb $9, 5($0)            ; 3: lb $t1, 5($zero)

                            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
```

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
Exception occurred at PC=0x00400024
  Bad address in data/stack read: 0x00000005
Attempt to execute non-instruction at 0x00400028
```



```
PC       = 400028
EPC      = 400028
Cause    = 0
BadVAddr = 5
Status   = 3000ff11

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 0
R3  [v1] = 0
R4  [a0] = 5
R5  [a1] = 7ffff5b0
R6  [a2] = 7ffff5c8
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
```

```
[00400000] 8fa
[00400004] 27a
[00400008] 24a
[0040000c] 000
[00400010] 00c
[00400014] 0c1
[00400018] 000
[0040001c] 340
[00400020] 000
[00400024] 800


[80000180] 000
[80000184] 3c0
[80000188] ac2
[8000018c] 3c0
[80000190] ac2
[80000194] 401
[80000198] 001
[8000019c] 308
[800001a0] 340
[800001a4] 3c0
[800001a8] 000
[800001ac] 340
[800001b0] 001
```

```
0]

00]
nd we can't trust $sp
se these registers
r
ode Field
_str)
_int)
Code Field
```

```
Exception 7  [Bad data address]  occurred and ignored
```

```
Memory and registers cleared

SPIM Version 9.1.23 of December 4, 2021
Copyright 1990-2021 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.
Exception occurred at PC=0x00400024
  Bad address in data/stack read: 0x00000005
Attempt to execute non-instruction at 0x00400028
```

The memory location is not allowed to be accessible. Permission Denied!

Thank You.