

Lab Report

ECPE 170 – Computer Systems and Networks – Fall 2012

Name: Erich Viebrock

Lab Topic: MIPS Assembly Programming (Basic) (Lab #: 8)

Lab

Question #1:

Take two screenshots of the MIPS register panel: before your program runs, and after your program finishes. Put the register panel in Decimal mode (right-click) so it is easy to see register values.

Answer:

Before-execution:

The screenshot shows the QtSpim MIPS simulator interface. The top menu bar includes File, Simulator, Registers, Text Segment, Data Segment, Window, and Help. Below the menu bar is a toolbar with various icons. The main window is divided into two panes. The left pane, titled 'Int Regs [10]', shows the MIPS registers in decimal mode. The right pane, titled 'Text', shows the assembly code for the 'User Text Segment' and 'Kernel Text Segment'.

Int Regs [10] (Decimal mode):

| 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] | 1 |
| R5 [a1] | 2147482284 |
| R6 [a2] | 2147482292 |
| 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 |

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] 3409000a ori $9, $0, 10 ; 24: ori $t1, $0, 10
[00400028] 340a000f ori $10, $0, 15 ; 26: ori $t2, $0, 15
[0040002c] 340b0005 ori $11, $0, 5 ; 28: ori $t3, $0, 5
[00400030] 340c0002 ori $12, $0, 2 ; 30: ori $t4, $0, 2
[00400034] 340d0007 ori $13, $0, 7 ; 32: ori $t5, $0, 7
[00400038] 3c01ffff lui $1, -1 ; 34: li $t6, -3
[0040003c] 342efffd ori $14, $1, -3
[00400040] 340f0000 ori $15, $0, 0 ; 36: ori $t7, $0, 0
[00400044] 012a8020 add $16, $9, $10 ; 50: add $s0, $t1, $t2
[00400048] 716c8802 mul $17, $11, $12 ; 52: mul $s1, $t3, $t4
[0040004c] 01ae9022 sub $18, $13, $14 ; 54: sub $s2, $t5, $t6
[00400050] 15600002 bne $11, $0, 8 ; 56: div $s3, $t1, $t3
[00400054] 0000000d break $0
[00400058] 012b001a div $9, $11
[0040005c] 00009812 mflo $19
[00400060] 0211a022 sub $20, $16, $17 ; 58: sub $s4, $s0, $s1
[00400064] 0292a820 add $21, $20, $18 ; 60: add $s5, $s4, $s2
[00400068] 02b3b022 sub $22, $21, $19 ; 62: sub $s6, $s5, $s3
[0040006c] 3c011001 lui $1, 4097 [Z] ; 64: sw $s6, Z
[00400070] ac360018 sw $22, 24($1) [Z]
[00400074] 3402000a ori $2, $0, 10 ; 69: li $v0, 10 # Sets $v0 to "10" to select exit
[00400078] 0000000c syscall ; 70: syscall # 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
```

After-execution:

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

FP Regs Int Regs [10] Data Text

Int Regs [10]

PC = 4194424
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664

HI = 0
LO = 2

R0 [r0] = 0
R1 [at] = 268500992
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 1
R5 [a1] = 2147482284
R6 [a2] = 2147482292
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 10
R10 [t2] = 15
R11 [t3] = 5
R12 [t4] = 2
R13 [t5] = 7
R14 [t6] = -3
R15 [t7] = 0
R16 [s0] = 25
R17 [s1] = 10
R18 [s2] = 10
R19 [s3] = 2
R20 [s4] = 15
R21 [s5] = 25
R22 [s6] = 23
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)
[00400024] 3409000a ori $9, $0, 10 ; 24: ori $t1, $0, 10
[00400028] 340a000f ori $10, $0, 15 ; 26: ori $t2, $0, 15
[0040002c] 340b0005 ori $11, $0, 5 ; 28: ori $t3, $0, 5
[00400030] 340c0002 ori $12, $0, 2 ; 30: ori $t4, $0, 2
[00400034] 340d0007 ori $13, $0, 7 ; 32: ori $t5, $0, 7
[00400038] 3c01ffff lui $1, -1 ; 34: li $t6, -3
[0040003c] 342efffd ori $14, $1, -3
[00400040] 340f0000 ori $15, $0, 0 ; 36: ori $t7, $0, 0
[00400044] 012a8020 add $16, $9, $10 ; 50: add $s0, $t1, $t2
[00400048] 716c8802 mul $17, $11, $12 ; 52: mul $s1, $t3, $t4
[0040004c] 01ae9022 sub $18, $13, $14 ; 54: sub $s2, $t5, $t6
[00400050] 15600002 bne $11, $0, 8 ; 56: div $s3, $t1, $t3
[00400054] 0000000d break $0
[00400058] 012b001a div $9, $11
[0040005c] 00009812 mflo $19
[00400060] 0211a022 sub $20, $16, $17 ; 58: sub $s4, $s0, $s1
[00400064] 0292a820 add $21, $20, $18 ; 60: add $s5, $s4, $s2
[00400068] 02b3b022 sub $22, $21, $19 ; 62: sub $s6, $s5, $s3
[0040006c] 3c011001 lui $1, 4097 [Z] ; 64: sw $s6, Z
[00400070] ac360018 sw $22, 24($1) [Z]
[00400074] 3402000a ori $2, $0, 10 ; 69: li $v0, 10 # Sets $v0 to "10" to select exit
[00400078] 0000000c syscall ; 70: syscall # 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
```

Question #2:

Take two screenshots of the MIPS memory panel (data tab): before your program runs, and after your program finishes. Put the memory panel in Decimal mode (right-click), so it is easy to see memory values. In the after-execution capture, **circle the memory element that contains the final calculated value of Z**.

Answer:

Before-execution:

The screenshot shows the QtSpim MIPS simulator interface. The top menu bar includes File, Simulator, Registers, Text Segment, Data Segment, Window, and Help. The main window is divided into several panes. On the left, the 'Int Regs [10]' pane shows the initial values of the MIPS registers: PC=0, EPC=0, Cause=0, BadVAddr=0, Status=805371664, HI=0, LO=0, R0-R27 are all 0. The right pane is titled 'Data' and shows the 'User data segment [10000000]..[10040000]' and 'User Stack [7ffffaa8]..[80000000]'. The data segment contains several lines of memory addresses and their corresponding values in hexadecimal. The stack segment contains a large block of memory addresses and their corresponding values in hexadecimal. The status bar at the bottom indicates the date and time: Thursday, November 01, 2012, 9:16:17 am, and the user name: erich.

| 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] | 1 |
| R5 [a1] | 2147482284 |
| R6 [a2] | 2147482292 |
| 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 |

User data segment [10000000]..[10040000]

| Address | Value |
|------------|---|
| [10000000] | 00000000 |
| [10010000] | 0000000010 0000000015 0000000005 0000000002 |
| [10010010] | 0000000007 00000000-3 0000000000 0000000000 |
| [10010020] | 00000000 |

User Stack [7ffffaa8]..[80000000]

| Address | Value |
|------------|---|
| [7ffffaa8] | 0000000001 2147482424 |
| [7ffffab0] | 0000000000 2147483625 2147483614 2147483596 |
| [7ffffac0] | 2147483577 2147483560 2147483536 2147483456 |
| [7ffffad0] | 2147483404 2147483306 2147483286 2147483244 |
| [7ffffae0] | 2147483189 2147483175 2147483123 2147483021 |
| [7ffffaf0] | 2147483001 2147482943 2147482921 2147482904 |
| [7ffffb00] | 2147482879 2147482844 2147482797 2147482776 |
| [7ffffb10] | 2147482760 2147482741 2147482725 2147482670 |
| [7ffffb20] | 2147482596 2147482573 2147482493 2147482480 |
| [7ffffb30] | 0000000000 0000000000 1836017711 1919233893 |
| [7ffffb40] | 0795370345 1651796322 1701536629 0808595316 |
| [7ffffb50] | 1717514801 1600941153 1701864293 0791689009 |
| [7ffffb60] | 0811753836 1634742072 0774993010 0007172961 |
| [7ffffb70] | 1347635524 1029259596 0808333370 1397052160 |
| [7ffffb80] | 1313818963 1312902495 1380271937 1668246589 |
| [7ffffb90] | 1697606753 1751345522 1949253690 0774860909 |
| [7ffffba0] | 0759513929 2020175477 0943010095 1853172787 |
| [7ffffbb0] | 1697609833 1751345522 1836330810 1227763568 |
| [7ffffbc0] | 1965901123 0796420462 0859321649 1112099840 |
| [7ffffbd0] | 1145130055 1330470725 1162630468 1413566559 |
| [7ffffbe0] | 0003816776 1598506072 1096040772 1380533343 |
| [7ffffbf0] | 1966030163 1932489331 1701994856 1651865647 |
| [7ffffc00] | 1970564725 1937059642 1869361010 0795631971 |
| [7ffffc10] | 1918986355 0792342373 0796029813 1918986355 |
| [7ffffc20] | 0792342373 0796029813 1918986355 1146617957 |
| [7ffffc30] | 1329815367 1195984462 1380533343 1697594707 |
| [7ffffc40] | 2016371572 2016372580 2016241508 1853186677 |
| [7ffffc50] | 0792360308 0795047013 0979854456 1668572463 |
| [7ffffc60] | 1734637615 1146572800 1869098813 1697604973 |

After-execution:

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

FP Regs Int Regs [10] Data Text

Int Regs [10]

PC = 4194424
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664

HI = 0
LO = 2

R0 [r0] = 0
R1 [at] = 268500992
R2 [v0] = 10
R3 [v1] = 0
R4 [a0] = 1
R5 [a1] = 2147482284
R6 [a2] = 2147482292
R7 [a3] = 0
R8 [t0] = 0
R9 [t1] = 10
R10 [t2] = 15
R11 [t3] = 5
R12 [t4] = 2
R13 [t5] = 7
R14 [t6] = -3
R15 [t7] = 0
R16 [s0] = 25
R17 [s1] = 10
R18 [s2] = 10
R19 [s3] = 2
R20 [s4] = 15
R21 [s5] = 25
R22 [s6] = 23
R23 [s7] = 0
R24 [t8] = 0
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0

User data segment [10000000]..[10040000]

| | | | | | |
|------------------------|------------|------------|------------|------------|-----|
| [10000000]..[1000ffff] | 00000000 | | | | |
| [10010000] | 0000000010 | 0000000015 | 0000000005 | 0000000002 | ... |
| [10010010] | 0000000007 | 0000000003 | 0000000023 | 0000000000 | ... |
| [10010020]..[1003ffff] | 00000000 | | | | |

User Stack [7ffffaa8]..[80000000]

| | | | | | |
|------------|------------|------------|------------|------------|--------------------------|
| [7ffffaa8] | 0000000001 | 2147482424 | | | ... 8 ... |
| [7ffffab0] | 0000000000 | 2147483625 | 2147483614 | 2147483596 | ... |
| [7ffffac0] | 2147483577 | 2147483560 | 2147483536 | 2147483456 | ... @ ... |
| [7ffffad0] | 2147483404 | 2147483306 | 2147483286 | 2147483244 | ... l ... |
| [7ffffae0] | 2147483189 | 2147483175 | 2147483123 | 2147483021 | 5 ... ' ... |
| [7ffffaf0] | 2147483001 | 2147482943 | 2147482921 | 2147482904 | y ... ? ...) ... |
| [7ffffb00] | 2147482879 | 2147482844 | 2147482797 | 2147482776 | ... |
| [7ffffb10] | 2147482760 | 2147482741 | 2147482725 | 2147482670 | ... u ... e ... |
| [7ffffb20] | 2147482596 | 2147482573 | 2147482493 | 2147482480 | ... } ... p ... |
| [7ffffb30] | 0000000000 | 0000000000 | 1836017711 | 1919233893 | ... / home / er |
| [7ffffb40] | 0795370345 | 1651796322 | 1701536629 | 0808595316 | ich / bit bucket / 20 |
| [7ffffb50] | 1717514801 | 1600941153 | 1701864293 | 0791689009 | 12 _ fall _ ecpe170 / |
| [7ffffb60] | 0811753836 | 1634742072 | 0774993010 | 0007172961 | lab08 / part1.asm. |
| [7ffffb70] | 1347635524 | 1029259596 | 0808333370 | 1397052160 | DISPLAY = : 0 . 0 . SES |
| [7ffffb80] | 1313818963 | 1312902495 | 1380271937 | 1668246589 | SION _ MANAGER = loc |
| [7ffffb90] | 1697606753 | 1751345522 | 1949253690 | 0774860909 | al / erich : @ / tmp / . |
| [7ffffba0] | 0759513929 | 2020175477 | 0943010095 | 1853172787 | ICE - unix / 1583 , un |
| [7ffffbb0] | 1697609833 | 1751345522 | 1836330810 | 1227763568 | ix / erich : / tmp / . I |
| [7ffffbc0] | 1965901123 | 0796420462 | 0859321649 | 1112099840 | CE - unix / 1583 . LIB |
| [7ffffbd0] | 1145130055 | 1330470725 | 1162630468 | 1413566559 | GLADE _ MODULE _ PAT |
| [7ffffbe0] | 0003816776 | 1598506072 | 1096040772 | 1380533343 | H = : . XDG _ DATA _ DIR |
| [7ffffbf0] | 1966030163 | 1932489331 | 1701994856 | 1651865647 | S = / usr / share / xub |
| [7ffffc00] | 1970564725 | 1937059642 | 1869361010 | 0795631971 | untu : / usr / local / |
| [7ffffc10] | 1918986355 | 0792342373 | 0796029813 | 1918986355 | share / : / usr / shar |
| [7ffffc20] | 0792342373 | 0796029813 | 1918986355 | 1146617957 | e / : / usr / share . XD |
| [7ffffc30] | 1329815367 | 1195984462 | 1380533343 | 1697594707 | G _ CONFIG _ DIRS = / e |
| [7ffffc40] | 2016371572 | 2016372580 | 2016241508 | 1853186677 | tc / xdg / xdg - xubun |
| [7ffffc50] | 0792360308 | 0795047013 | 0979854456 | 1668572463 | tu : / etc / xdg : / etc |
| [7ffffc60] | 1734637615 | 1146572800 | 1869098813 | 1697604973 | / xdg . PWD = / home / e |

Question #3:

Take two screenshots of the MIPS register panel: before your program runs, and after your program finishes. Put the register panel in Decimal mode (right-click) so it is easy to see register values.

Answer:

Before-execution:

The screenshot shows the QtSpim MIPS simulator interface. The top menu bar includes File, Simulator, Registers, Text Segment, Data Segment, Window, and Help. The top toolbar contains icons for file operations, simulation control, and window management. The status bar at the top right shows the date and time: Tuesday, November 06, 2012, 10:49:06 pm, and the username: erich.

The main window is divided into two panes. The left pane, titled "Int Regs [10]", displays the MIPS registers in decimal mode. The right pane, titled "Text", displays the assembly code for the "User Text Segment [00400000]..[00440000]".

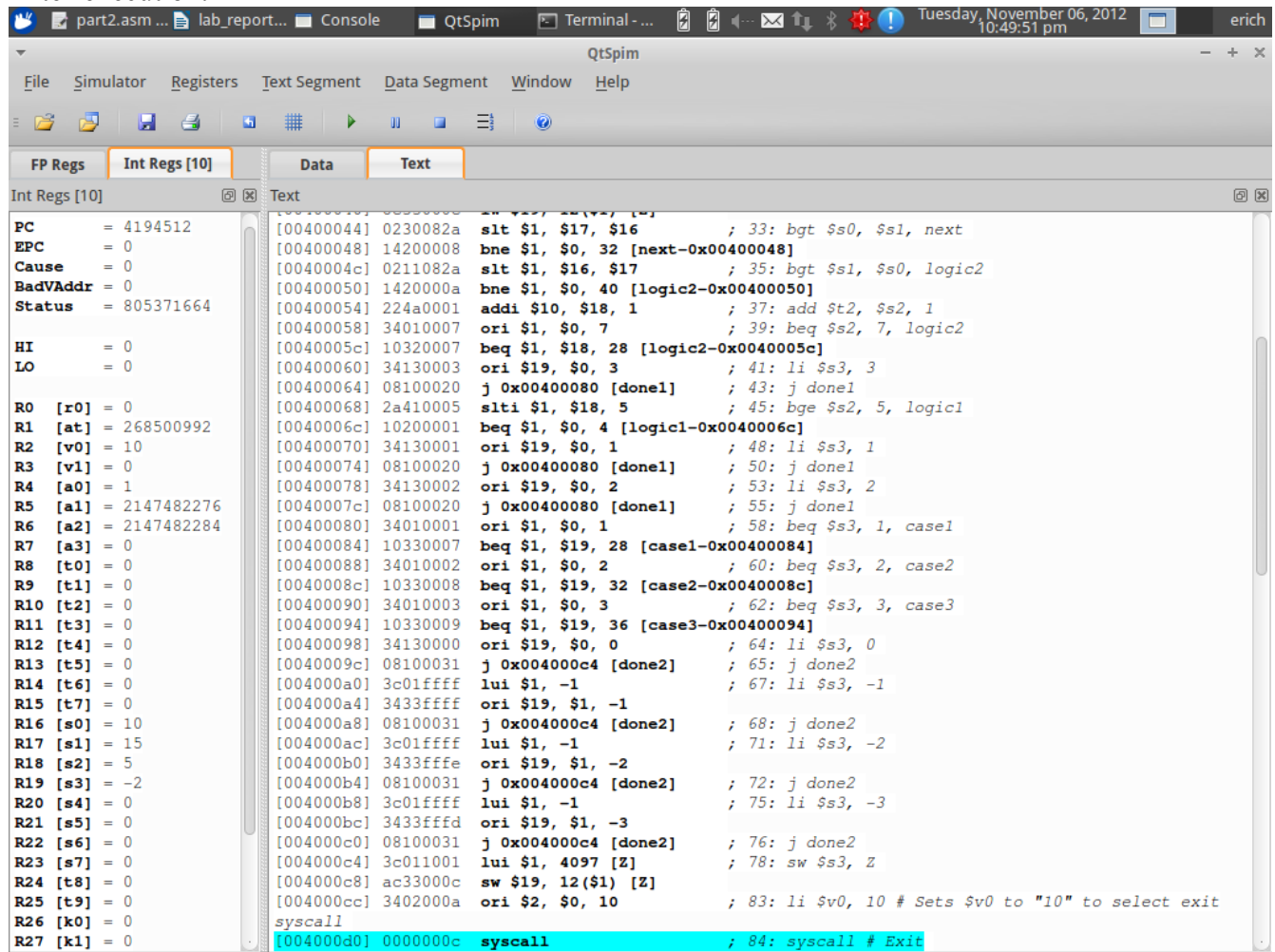
Registers (Int Regs [10]):

| 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] | = 1 |
| R5 [a1] | = 2147482276 |
| R6 [a2] | = 2147482284 |
| 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 |

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] 3c011001 lui $1, 4097 [A] ; 22: lw $s0, A
[00400028] 8c300000 lw $16, 0($1) [A]
[0040002c] 3c011001 lui $1, 4097 [B] ; 24: lw $s1, B
[00400030] 8c310004 lw $17, 4($1) [B]
[00400034] 3c011001 lui $1, 4097 [C] ; 26: lw $s2, C
[00400038] 8c320008 lw $18, 8($1) [C]
[0040003c] 3c011001 lui $1, 4097 [Z] ; 28: lw $s3, Z
[00400040] 8c33000c lw $19, 12($1) [Z]
[00400044] 0230082a slt $1, $17, $16 ; 33: bgt $s0, $s1, next
[00400048] 14200008 bne $1, $0, 32 [next-0x00400048]
[0040004c] 0211082a slt $1, $16, $17 ; 35: bgt $s1, $s0, logic2
[00400050] 1420000a bne $1, $0, 40 [logic2-0x00400050]
[00400054] 224a0001 addi $10, $18, 1 ; 37: add $t2, $s2, 1
[00400058] 34010007 ori $1, $0, 7 ; 39: beq $s2, 7, logic2
[0040005c] 10320007 beq $1, $18, 28 [logic2-0x0040005c]
[00400060] 34130003 ori $19, $0, 3 ; 41: li $s3, 3
[00400064] 08100020 j 0x00400080 [done1] ; 43: j done1
[00400068] 2a410005 slti $1, $18, 5 ; 45: bge $s2, 5, logic1
[0040006c] 10200001 beq $1, $0, 4 [logic1-0x0040006c]
[00400070] 34130001 ori $19, $0, 1 ; 48: li $s3, 1
[00400074] 08100020 j 0x00400080 [done1] ; 50: j done1
[00400078] 34130002 ori $19, $0, 2 ; 53: li $s3, 2
[0040007c] 08100020 j 0x00400080 [done1] ; 55: j done1
[00400080] 34010001 ori $1, $0, 1 ; 58: beq $s3, 1, case1
[00400084] 10330007 beq $1, $19, 28 [case1-0x00400084]
[00400088] 34010002 ori $1, $0, 2 ; 60: beq $s3, 2, case2
[0040008c] 10330008 beq $1, $19, 32 [case2-0x0040008c]
```


After-execution:



The screenshot shows the QtSpim MIPS simulator interface. The top menu bar includes File, Simulator, Registers, Text Segment, Data Segment, Window, and Help. Below the menu is a toolbar with icons for file operations, simulation, and viewing. The main window is divided into two panes. The left pane, titled 'Int Regs [10]', displays the state of MIPS registers: PC (4194512), 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] (1), R5 [a1] (2147482276), R6 [a2] (2147482284), 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] (10), R17 [s1] (15), R18 [s2] (5), R19 [s3] (-2), R20 [s4] (0), R21 [s5] (0), R22 [s6] (0), R23 [s7] (0), R24 [t8] (0), R25 [k0] (0), R26 [k1] (0), and R27 [k1] (0). The right pane, titled 'Text', displays assembly code with addresses and comments. The code includes instructions like `slt $1, $17, $16`, `bne $1, $0, 32 [next-0x00400048]`, `addi $10, $18, 1`, `ori $1, $0, 7`, `beq $1, $18, 28 [logic2-0x0040005c]`, `ori $19, $0, 3`, `j 0x00400080 [done1]`, `slli $1, $18, 5`, `beq $1, $0, 4 [logic1-0x0040006c]`, `ori $19, $0, 1`, `j 0x00400080 [done1]`, `ori $19, $0, 2`, `j 0x00400080 [done1]`, `ori $1, $0, 1`, `beq $1, $19, 28 [case1-0x00400084]`, `ori $1, $0, 2`, `beq $1, $19, 32 [case2-0x0040008c]`, `ori $1, $0, 3`, `beq $1, $19, 36 [case3-0x00400094]`, `ori $19, $0, 0`, `j 0x004000c4 [done2]`, `lui $1, -1`, `ori $19, $1, -1`, `j 0x004000c4 [done2]`, `lui $1, -1`, `ori $19, $1, -2`, `j 0x004000c4 [done2]`, `lui $1, -1`, `ori $19, $1, -3`, `j 0x004000c4 [done2]`, `lui $1, 4097 [Z]`, `sw $19, 12($1) [Z]`, `ori $2, $0, 10`, `syscall`, and `syscall`. The last instruction, `syscall`, is highlighted in blue. The status bar at the bottom shows the address 0000000c and the instruction `syscall`.

Question #4:

Take two screenshots of the MIPS memory panel (data tab): before your program runs, and after your program finishes. Put the memory panel in Decimal mode (right-click), so it is easy to see memory values. In the after-execution capture, **circle the memory element that contains the final calculated value of Z**.

Answer:

Before-execution:

The screenshot shows the QtSpim MIPS simulator interface. The top menu bar includes File, Simulator, Registers, Text Segment, Data Segment, Window, and Help. The top toolbar contains icons for file operations, simulation, and viewing. The 'Int Regs [10]' panel on the left shows the following register values:

| 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] | = 1 |
| R5 [a1] | = 2147482276 |
| R6 [a2] | = 2147482284 |
| 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 |

The 'Data' panel on the right shows memory contents in decimal mode. The 'User data segment [10000000]..[10040000]' is visible, containing the following values:

| Address | Value |
|------------------------|---|
| [10000000]..[1000ffff] | 00000000 |
| [10010000]..[1001ffff] | 000000000010 000000000015 000000000005 000000000000 |
| [10010010]..[1003ffff] | 00000000 |

The 'User Stack [7ffffaa0]..[80000000]' is also visible, containing the following values:

| Address | Value |
|------------|---|
| [7ffffaa0] | 00000000001 2147482417 00000000000 2147483625 |
| [7ffffab0] | 2147483614 2147483596 2147483577 2147483560 |
| [7ffffac0] | 2147483536 2147483455 2147483403 2147483305 |
| [7ffffad0] | 2147483285 2147483243 2147483188 2147483174 |
| [7ffffae0] | 2147483122 2147483020 2147483000 2147482942 |
| [7ffffaf0] | 2147482920 2147482903 2147482878 2147482843 |
| [7ffffb00] | 2147482796 2147482775 2147482759 2147482740 |
| [7ffffb10] | 2147482724 2147482669 2147482595 2147482572 |
| [7ffffb20] | 2147482492 2147482479 0000000000 0000000000 |
| [7ffffb30] | 1869098752 1697604973 1751345522 1953063471 |
| [7ffffb40] | 1801680226 0841970789 1597124912 1819042150 |
| [7ffffb50] | 1885562207 0808923493 1650551855 1882142768 |
| [7ffffb60] | 0846492257 1918988335 1630417524 1140878707 |
| [7ffffb70] | 1280332617 0977099073 0003157552 1397966163 |
| [7ffffb80] | 1598967625 1095647565 1028801863 1633906540 |
| [7ffffb90] | 1919233900 0979919721 1836330816 1227763568 |
| [7ffffba0] | 1965901123 0796420462 0825570353 1768846636 |
| [7ffffbb0] | 1919233912 0979919721 1886221359 1128869423 |
| [7ffffbc0] | 1853173061 0825194601 0003224884 1195526476 |
| [7ffffbd0] | 1162101068 1146047839 1598377045 1213481296 |
| [7ffffbe0] | 1476409917 1147094852 1598116929 1397901636 |
| [7ffffbf0] | 1937059645 1752379250 0795177569 1969386872 |
| [7ffffc00] | 0980776046 1920169263 1668246575 1932487777 |
| [7ffffc10] | 1701994856 1966029359 1932489331 1701994856 |
| [7ffffc20] | 1966029359 1932489331 1701994856 1195661312 |
| [7ffffc30] | 1313817439 1598507334 1397901636 1952788285 |
| [7ffffc40] | 1685598051 1685598055 1970810215 1953396066 |
| [7ffffc50] | 1697593973 2016371572 0792356708 0795047013 |
| [7ffffc60] | 0006775928 1027888976 1836017711 1919233893 |
| [7ffffc70] | 0006841193 1397572679 1230197573 2017283663 |

After-execution:

part2.asm ... lab_report... Console QtSpim Terminal - ... Tuesday, November 06, 2012 10:50:37 pm erich

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

FP Regs Int Regs [10] Data Text

Int Regs [10]

PC = 4194512
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] = 1
R5 [a1] = 2147482276
R6 [a2] = 2147482284
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] = 10
R17 [s1] = 15
R18 [s2] = 5
R19 [s3] = -2
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 data segment [10000000]..[10040000]

[10000000]..[1000ffff] 00000000
[10010000] 0000000010 0000000015 0000000005 00000000-2
[10010010]..[1003ffff] 00000000

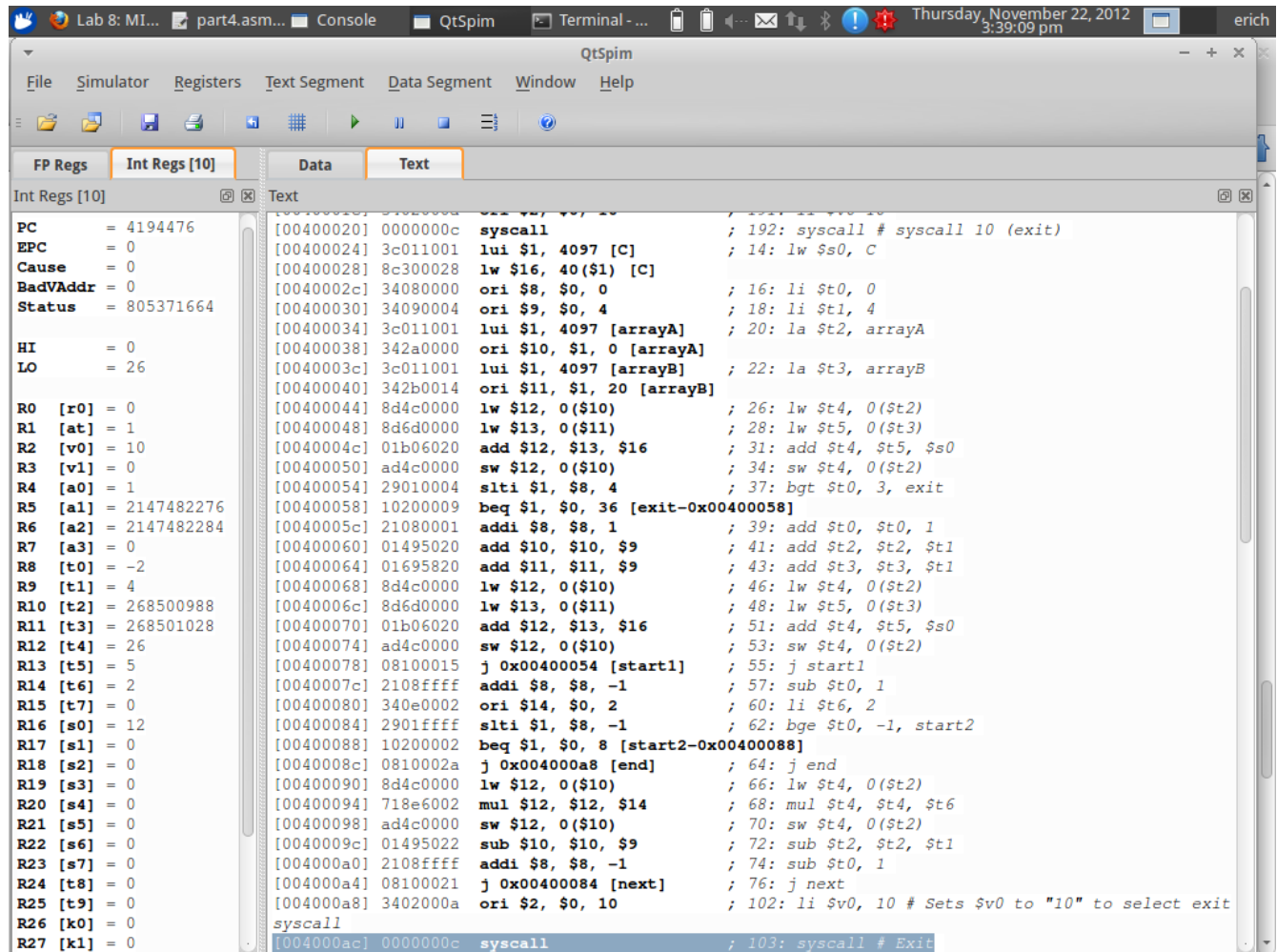
User Stack [7ffffaa0]..[80000000]

[7ffffaa0] 0000000001 2147482417 0000000000 2147483625 1
[7ffffab0] 2147483614 2147483596 2147483577 2147483560
[7ffffac0] 2147483536 2147483455 2147483403 2147483305 ?
[7ffffad0] 2147483285 2147483243 2147483188 2147483174 k . . . 4 . . & . . .
[7ffffae0] 2147483122 2147483020 2147483000 2147482942 x . . . > . . .
[7ffffaf0] 2147482920 2147482903 2147482878 2147482843 (.
[7ffffb00] 2147482796 2147482775 2147482759 2147482740 t . . .
[7ffffb10] 2147482724 2147482669 2147482595 2147482572 d . . . -
[7ffffb20] 2147482492 2147482479 0000000000 0000000000 l . . . o
[7ffffb30] 1869098752 1697604973 1751345522 1953063471 . / h o m e / e r i c h / b i t
[7ffffb40] 1801680226 0841970789 1597124912 1819042150 b u c k e t / 2 0 1 2 _ f a i l l
[7ffffb50] 1885562207 0808923493 1650551855 1882142768 _ e c p e 1 7 0 / l a b 0 8 / p
[7ffffb60] 0846492257 1918988335 1630417524 1140878707 a r t 2 / p a r t 2 . a s m . D
[7ffffb70] 1280332617 0977099073 0003157552 1397966163 I S P L A Y = : 0 . 0 . S E S S
[7ffffb80] 1598967625 1095647565 1028801863 1633906540 I O N _ M A N A G E R = l o c a
[7ffffb90] 1919233900 0979919721 1836330816 1227763568 l / e r i c h : @ / t m p / . I
[7ffffba0] 1965901123 0796420462 0825570353 1768846636 C E - u n i x / 1 4 5 1 , u n i
[7ffffbb0] 1919233912 0979919721 1886221359 1128869423 x / e r i c h : / t m p / . I C
[7ffffbc0] 1853173061 0825194601 0003224884 1195526476 E - u n i x / 1 4 5 1 . L I B G
[7ffffbd0] 1162101068 1146047839 1598377045 1213481296 L A D E _ M O D U L E _ P A T H
[7ffffbe0] 1476409917 1147094852 1598116929 1397901636 = : . X D G _ D A T A _ D I R S
[7ffffbf0] 1937059645 1752379250 0795177569 1969386872 = / u s r / s h a r e / x u b u
[7ffffc00] 0980776046 1920169263 1668246575 1932487777 n t u : / u s r / l o c a l / s
[7ffffc10] 1701994856 1966029359 1932489331 1701994856 h a r e / : / u s r / s h a r e
[7ffffc20] 1966029359 1932489331 1701994856 1195661312 / : / u s r / s h a r e . X D G
[7ffffc30] 1313817439 1598507334 1397901636 1952788285 _ C O N F I G _ D I R S = / e t
[7ffffc40] 1685598051 1685598055 1970810215 1953396066 c / x d g / x d g - x u b u n t
[7ffffc50] 1697593973 2016371572 0792356708 0795047013 u : / e t c / x d g : / e t c /
[7ffffc60] 0006775928 1027888976 1836017711 1919233893 x d g . P W D = / h o m e / e r
[7ffffc70] 0006841193 1397572679 1230197573 2017283663 i c h . G D M S E S S I O N = x

Question #5:

Take a screenshot of the MIPS register panel after your program finishes. Put the register panel in Decimal mode (right-click) so it is easy to see register values.

Answer:



The screenshot shows the QtSpim MIPS simulator interface. The top menu bar includes File, Simulator, Registers, Text Segment, Data Segment, Window, and Help. Below the menu bar is a toolbar with various icons. The main window is divided into two panes. The left pane, titled 'Int Regs [10]', displays the MIPS register panel in decimal mode. The right pane, titled 'Text', displays the assembly code. The assembly code is a MIPS program that calculates the sum of two arrays, arrayA and arrayB, and stores the result in register \$s0. The program uses a loop to iterate through the arrays and calculate the sum. The final result is stored in register \$s0, which is then printed to the console using the syscall instruction.

| Register | Value |
|----------|------------|
| PC | 4194476 |
| EPC | 0 |
| Cause | 0 |
| BadVAddr | 0 |
| Status | 805371664 |
| HI | 0 |
| LO | 26 |
| R0 [r0] | 0 |
| R1 [at] | 1 |
| R2 [v0] | 10 |
| R3 [v1] | 0 |
| R4 [a0] | 1 |
| R5 [a1] | 2147482276 |
| R6 [a2] | 2147482284 |
| R7 [a3] | 0 |
| R8 [t0] | -2 |
| R9 [t1] | 4 |
| R10 [t2] | 268500988 |
| R11 [t3] | 268501028 |
| R12 [t4] | 26 |
| R13 [t5] | 5 |
| R14 [t6] | 2 |
| R15 [t7] | 0 |
| R16 [s0] | 12 |
| 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 |

```
[00400020] 0000000c syscall ; 192: syscall # syscall 10 (exit)
[00400024] 3c011001 lui $1, 4097 [C] ; 14: lw $s0, C
[00400028] 8c300028 lw $16, 40($1) [C]
[0040002c] 34080000 ori $8, $0, 0 ; 16: li $t0, 0
[00400030] 34090004 ori $9, $0, 4 ; 18: li $t1, 4
[00400034] 3c011001 lui $1, 4097 [arrayA] ; 20: la $t2, arrayA
[00400038] 342a0000 ori $10, $1, 0 [arrayA]
[0040003c] 3c011001 lui $1, 4097 [arrayB] ; 22: la $t3, arrayB
[00400040] 342b0014 ori $11, $1, 20 [arrayB]
[00400044] 8d4c0000 lw $12, 0($10) ; 26: lw $t4, 0($t2)
[00400048] 8d6d0000 lw $13, 0($11) ; 28: lw $t5, 0($t3)
[0040004c] 01b06020 add $12, $13, $16 ; 31: add $t4, $t5, $s0
[00400050] ad4c0000 sw $12, 0($10) ; 34: sw $t4, 0($t2)
[00400054] 29010004 slti $1, $8, 4 ; 37: bgt $t0, 3, exit
[00400058] 10200009 beq $1, $0, 36 [exit-0x00400058]
[0040005c] 21080001 addi $8, $8, 1 ; 39: add $t0, $t0, 1
[00400060] 01495020 add $10, $10, $9 ; 41: add $t2, $t2, $t1
[00400064] 01695820 add $11, $11, $9 ; 43: add $t3, $t3, $t1
[00400068] 8d4c0000 lw $12, 0($10) ; 46: lw $t4, 0($t2)
[0040006c] 8d6d0000 lw $13, 0($11) ; 48: lw $t5, 0($t3)
[00400070] 01b06020 add $12, $13, $16 ; 51: add $t4, $t5, $s0
[00400074] ad4c0000 sw $12, 0($10) ; 53: sw $t4, 0($t2)
[00400078] 08100015 j 0x00400054 [start1] ; 55: j start1
[0040007c] 2108ffff addi $8, $8, -1 ; 57: sub $t0, 1
[00400080] 340e0002 ori $14, $0, 2 ; 60: li $t6, 2
[00400084] 2901ffff slti $1, $8, -1 ; 62: bge $t0, -1, start2
[00400088] 10200002 beq $1, $0, 8 [start2-0x00400088]
[0040008c] 0810002a j 0x004000a8 [end] ; 64: j end
[00400090] 8d4c0000 lw $12, 0($10) ; 66: lw $t4, 0($t2)
[00400094] 718e6002 mul $12, $12, $14 ; 68: mul $t4, $t4, $t6
[00400098] ad4c0000 sw $12, 0($10) ; 70: sw $t4, 0($t2)
[0040009c] 01495022 sub $10, $10, $9 ; 72: sub $t2, $t2, $t1
[004000a0] 2108ffff addi $8, $8, -1 ; 74: sub $t0, 1
[004000a4] 08100021 j 0x00400084 [next] ; 76: j next
[004000a8] 3402000a ori $2, $0, 10 ; 102: li $v0, 10 # Sets $v0 to "10" to select exit
[004000ac] 0000000c syscall ; 103: syscall # Exit
```

Question #6:

Take a screenshot of the MIPS memory panel (data tab) after your program finishes. Put the memory panel in Decimal mode (right-click), so it is easy to see memory values. **Circle the final values of array A.**

Answer:

Lab 8: MI...

part4.asm...

Console

QtSpim

Thursday, November 22, 2012 3:40:21 pm

File

Simulator

Registers

Text Segment

Data Segment

Window

Help

FP Regs

Int Regs [10]

Data

Text

Int Regs [10]

Data

PC = 4194476

EPC = 0

Cause = 0

BadVAddr = 0

Status = 805371664

HI = 0

LO = 26

User data segment [10000000]..[10040000]

[10000000]..[10005555] 00000000

[10010000] 0000000026 0000000028 0000000030 0000000032

[10010010] 0000000034 0000000001 0000000002 0000000003

[10010020] 0000000004 0000000005 0000000012 0000000000

[10010030]..[1003ffff] 00000000

User Stack [7ffffaa0]..[80000000]

[7ffffaa0] 0000000001 2147482418 0000000000 2147483625

[7fffffab0] 2147483614 2147483596 2147483577 2147483560

[7fffffac0] 2147483536 2147483456 2147483404 2147483306

[7fffffad0] 2147483286 2147483244 2147483189 2147483175

[7fffffae0] 2147483123 2147483021 2147483001 2147482943

[7fffffaf0] 2147482921 2147482904 2147482879 2147482844

[7fffffb00] 2147482797 2147482776 2147482760 2147482741

[7fffffb10] 2147482725 2147482670 2147482596 2147482573

[7fffffb20] 2147482493 2147482480 0000000000 0000000000

[7fffffb30] 1747910656 0795176303 1667854949 1768042344

[7fffffb40] 1668637300 0796157291 0842084402 1818322527

[7fffffb50] 1667587948 0925984112 1634479920 0792211554

[7fffffb60] 1953653104 1634742068 0775189618 0007172961

[7fffffb70] 1347635524 1029259596 0808333370 1397052160

[7fffffb80] 1313818963 1312902495 1380271937 1668246589

[7fffffb90] 1697606753 1751345522 1949253690 0774860909

[7fffffba0] 0759513929 2020175477 0842281263 1853172788

[7fffffbb0] 1697609833 1751345522 1836330810 1227763568

[7fffffbc0] 1965901123 0796420462 0875705393 1112099840

[7fffffbd0] 1145130055 1330470725 1162630468 1413566559

[7fffffbe0] 0003816776 1598506072 1096040772 1380533343

[7fffffbf0] 1966030163 1932489331 1701994856 1651865647

[7fffffc00] 1970564725 1937059642 1869361010 0795631971

[7fffffc10] 1918986355 0792342373 0796029813 1918986355

[7fffffc20] 0792342373 0796029813 1918986355 1146617957

[7fffffc30] 1329815367 1195984462 1380533343 1697594707

[7fffffc40] 2016371572 2016372580 2016241508 1853186677

[7fffffc50] 0792360308 0795047013 0979854456 1668572463

R0 [r0] = 0

R1 [at] = 1

R2 [v0] = 10

R3 [v1] = 0

R4 [a0] = 1

R5 [a1] = 2147482276

R6 [a2] = 2147482284

R7 [a3] = 0

R8 [t0] = -2

R9 [t1] = 4

R10 [t2] = 268500988

R11 [t3] = 268501028

R12 [t4] = 26

R13 [t5] = 5

R14 [t6] = 2

R15 [t7] = 0

R16 [s0] = 12

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

Question #7:

Take a screenshot of the MIPS memory panel (data tab) after your program finishes. Put the memory panel in Hex mode (right-click), since Decimal mode will not allow us to distinguish between bytes.

Circle two things: the final value of the pointer 'result' in memory, and the corresponding location that result points to. Does that location in memory contain the ASCII code for the character 't'? (If not, you had better check your work!)

Answer:

The screenshot shows the QtSpim MIPS simulator interface. The 'Int Regs [10]' tab is selected, and the 'Data' segment is visible. The memory address 00000074 is circled, and the corresponding value 00000000 is also circled. The memory address 00000000 is also circled. The memory address 00000000 is also circled.

| PC | EPC | Cause | BadVAddr | Status |
|---------|-----|-------|----------|-----------|
| 4194468 | 0 | 0 | 0 | 805371664 |

| HI | LO |
|----|----|
| 0 | 0 |

| R0 [r0] | R1 [at] | R2 [v0] | R3 [v1] | R4 [a0] | R5 [a1] | R6 [a2] | R7 [a3] | R8 [t0] | R9 [t1] | R10 [t2] | R11 [t3] | R12 [t4] | R13 [t5] | R14 [t6] | R15 [t7] | R16 [s0] | R17 [s1] | R18 [s2] | R19 [s3] | R20 [s4] | R21 [s5] | R22 [s6] | R23 [s7] | R24 [t8] | R25 [t9] | R26 [k0] | R27 [k1] |
|---------|-----------|---------|---------|---------|---------|------------|---------|---------|-----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|----------|
| 0 | 268500992 | 10 | 0 | 116 | 256 | 2147482284 | 0 | 116 | 268500992 | 116 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | |

| User data segment [10000000]..[10040000] | | | | | |
|--|----------|----------|----------|----------|-------------------------------|
| [10000000]..[10000000] | 00000000 | 00000000 | 00000000 | 00000000 | t |
| [10010000]..[10010000] | 00000000 | 00000000 | 00000000 | 00000000 | |
| [10010010]..[10010010] | 65746e45 | 20612072 | 64726f77 | 4d00203a | E n t e r a w o r d : . M |
| [10010100]..[10010100] | 68637461 | 20676e69 | 72616863 | 65746361 | a t c h i n g c h a r a c t e |
| [10010110]..[10010110] | 73692072 | 4e00203a | 616d206f | 20686374 | r i s : . N o m a t c h |
| [10010120]..[10010120] | 6e756f66 | 000a2e64 | 00000000 | 00000000 | f o u n d |
| [10010130]..[10010130] | 00000000 | 00000000 | 00000000 | 00000000 | |
| [10010140]..[1003ffff] | 00000000 | 00000000 | 00000000 | 00000000 | |

| User Stack [7ffffaa0]..[80000000] | | | | | |
|-----------------------------------|----------|----------|----------|----------|-----------------------------------|
| [7ffffaa0]..[7ffffaa0] | 00000001 | 7ffffb32 | 00000000 | 7fffffe9 | 2 |
| [7ffffab0]..[7ffffab0] | 7fffffde | 7fffffcc | 7fffffb9 | 7fffffa8 | |
| [7ffffac0]..[7ffffac0] | 7fffff90 | 7fffff40 | 7fffff0c | 7ffffeaa | @ |
| [7ffffad0]..[7ffffad0] | 7ffffe96 | 7ffffe6c | 7ffffe35 | 7ffffe27 | l 5 |
| [7ffffae0]..[7ffffae0] | 7ffffdf3 | 7ffffd8d | 7ffffd79 | 7ffffd3f | y ? |
| [7ffffaf0]..[7ffffaf0] | 7ffffd29 | 7ffffd18 | 7ffffcf7 | 7ffffcdc |) |
| [7ffffb00]..[7ffffb00] | 7ffffcad | 7ffffc98 | 7ffffc88 | 7ffffc75 | u |
| [7ffffb10]..[7ffffb10] | 7ffffc65 | 7ffffc2e | 7ffffbe4 | 7ffffbcd | e |
| [7ffffb20]..[7ffffb20] | 7ffffb7d | 7ffffb70 | 00000000 | 00000000 | } p |
| [7ffffb30]..[7ffffb30] | 682f0000 | 2f656d6f | 63697265 | 69622f68 | . . / h o m e / e r i c h / b i |
| [7ffffb40]..[7ffffb40] | 63756274 | 2f74656b | 32313032 | 6c61665f | t b u c k e t / 2 0 1 2 _ f a l |
| [7ffffb50]..[7ffffb50] | 63655f6c | 37316570 | 616c2f30 | 2f383062 | l _ e c p e 1 7 0 / l a b 0 8 / |
| [7ffffb60]..[7ffffb60] | 74726170 | 61702f35 | 2e357472 | 006d7361 | p a r t 5 / p a r t 5 . a s m . |
| [7ffffb70]..[7ffffb70] | 50534944 | 3d59414c | 302e303a | 53455300 | D I S P L A Y = : 0 . 0 . S E S |
| [7ffffb80]..[7ffffb80] | 4e4f4953 | 4e414d5f | 52454741 | 636f6c3d | S I O N _ M A N A G E R = l o c |
| [7ffffb90]..[7ffffb90] | 652f6c61 | 68636972 | 742f403a | 2e2f706d | a l / e r i c h : @ / t m p / . |
| [7ffffba0]..[7ffffba0] | 2d454349 | 78696e75 | 3434312f | 6e752c33 | I C E - u n i x / 1 4 4 3 , u n |
| [7ffffbb0]..[7ffffbb0] | 652f7869 | 68636972 | 6d742f3a | 492e2f70 | i x / e r i c h : / t m p / . I |
| [7ffffbc0]..[7ffffbc0] | 752d4543 | 2f78696e | 33343431 | 42494c00 | C E - u n i x / 1 4 4 3 . L I B |
| [7ffffbd0]..[7ffffbd0] | 44414c47 | 4f4d5f45 | 454c5544 | 5441505f | G L A D E _ M O D U L E _ P A T |
| [7ffffbe0]..[7ffffbe0] | 003a3d48 | 5f474458 | 41544144 | 5249445f | H = : . X D G _ D A T A _ D I R |
| [7ffffbf0]..[7ffffbf0] | 752f3d53 | 732f7273 | 65726168 | 6275782f | S = / u s r / s h a r e / x u b |
| [7ffffc00]..[7ffffc00] | 75746e75 | 73752f3a | 6f6c2f72 | 2f6c6163 | u n t u : / u s r / l o c a l / |
| [7ffffc10]..[7ffffc10] | 72616873 | 2f3a2f65 | 2f727375 | 72616873 | s h a r e / : / u s r / s h a r |
| [7ffffc20]..[7ffffc20] | 2f3a2f65 | 2f727375 | 72616873 | 44580065 | e / : / u s r / s h a r e . X D |

Post-Lab

Question #1:

What was the best aspect of this lab?

Answer:

Practicing an assembly language gives the programmer a closer look at how programs work behind the scenes.

Question #2:

What was the worst aspect of this lab?

Answer:

The appendixes were difficult to decipher from. Went online to find most of the help.

Question #3:

How would you suggest improving this lab in future semesters?

Answer:

More example programs similar to the assembly code asked for.