

CS 211
LAB 1: Introduction to QtSpim
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:

a) Add numbers 100 and -82 and store the result in register \$10. (Hint: Find 2's complement of 82 and add it to 100)

ans. I attached a file of code, Here is The Output –
In Hexa Decimal –

The screenshot shows the QtSpim MIPS simulator interface. The 'Registers' window on the left displays the current state of registers: PC is 400034, Cause is 0, BadVAddr is 0, Status is 3000ff10, HI is 0, LO is 0, R0-R7 are 0, R8-R15 are 0. The 'Text' window shows the assembly code for the 'User Text Segment' and 'Kernel Text Segment'. The assembly code includes instructions like lw, addiu, ori, sll, addu, jal, nop, syscall, ori, sll, sub, add, and andi. The 'Data' window is empty. The 'Memory' window shows the memory address 0x00000000. The 'Status' window shows the status register value 3000ff10. The 'Registers' window shows the register values: R0-R7 are 0, R8-R15 are 0.

In Decimal -

The screenshot shows the QtSpim MIPS simulator interface. The 'Registers' window on the left displays the current state of registers: PC is 4194356, Cause is 0, BadVAddr is 805371664, Status is 805371664, HI is 0, LO is 0, R0-R7 are 0, R8-R15 are 0. The 'Text' window shows the assembly code for the 'User Text Segment' and 'Kernel Text Segment'. The assembly code includes instructions like lw, addiu, ori, sll, addu, jal, nop, syscall, ori, sll, sub, add, and andi. The 'Data' window is empty. The 'Memory' window shows the memory address 0x00000000. The 'Status' window shows the status register value 805371664. The 'Registers' window shows the register values: R0-R7 are 0, R8-R15 are 0.

b) Store FFFF in \$8. Left shift it 2 times and store the result in memory location 0x10000000.

In Decimal –

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

Int Regs [10]

PC = 4194356
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 0
R0 [x0] = 0
R1 [at] = 268435456
R2 [v0] = 0
R3 [v1] = 0
R4 [a0] = 4
R5 [a1] = 2147481116
R6 [a2] = 2147481136
R7 [a3] = 0
R8 [t0] = 65535
R9 [t1] = 262140
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0

Text

User Text Segment [00400000]..[00440000]

```
[00400000] 8fa40000 lw $4, 0($29) ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004 addiu $5, $29, 4 ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004 addiu $6, $5, 4 ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080 sll $2, $4, 2 ; 186: sll $v0 $a0 2
[00400010] 00c23021 addu $6, $6, $2 ; 187: addu $a2 $a2 $v0
[00400014] 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] 3408ffff ori $8, $0, -1 ; 8: ori $8,$0,0xffff # Here Load the hex value FFFF into register $8
[00400028] 00084880 sll $9, $8, 2 ; 9: sll $9, $8, 2 # the "sll" instruction is used for left shift operation
[0040002c] 3c011000 lui $1, 4096 ; 10: sw $9,0x10000000 # and "sw" instruction is used to store the value of $9
[00400030] ac290000 sw $9, 0($1)
```

Kernel Text Segment [80000000]..[80010000]

```
[80000180] 0001d821 addu $27, $0, $1 ; 90: move $k1 $at # Save $at
[80000184] 3c019000 lui $1, -28672 ; 92: sw $v0 $1 # Not re-entrant and we can't trust $sp
[80000188] ac220200 sw $2, 512($1) ; 93: sw $a0 $2 # But we need to use these registers
[8000018c] 3c019000 lui $1, -28672
[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 ExCoDe 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)
```

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.
Attempt to execute non-instruction at 0x00400034

In Hexa Decimal –

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

Int Regs [16]

PC = 400034
EPC = 0
Cause = 0
BadVAddr = 0
Status = 3000fff10
HI = 0
LO = 0
R0 [x0] = 0
R1 [at] = 10000000
R2 [v0] = 0
R3 [v1] = 0
R4 [a0] = 4
R5 [a1] = 7ffff61c
R6 [a2] = 7ffff630
R7 [a3] = 0
R8 [t0] = ffff
R9 [t1] = 3ffffc
R10 [t2] = 0
R11 [t3] = 0
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 0

Text

User Text Segment [00400000]..[00440000]

```
[00400000] 8fa40000 lw $4, 0($29) ; 183: lw $a0 0($sp) # argc
[00400004] 27a50004 addiu $5, $29, 4 ; 184: addiu $a1 $sp 4 # argv
[00400008] 24a60004 addiu $6, $5, 4 ; 185: addiu $a2 $a1 4 # envp
[0040000c] 00041080 sll $2, $4, 2 ; 186: sll $v0 $a0 2
[00400010] 00c23021 addu $6, $6, $2 ; 187: addu $a2 $a2 $v0
[00400014] 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] 3408ffff ori $8, $0, -1 ; 8: ori $8,$0,0xffff # Here Load the hex value FFFF into register $8
[00400028] 00084880 sll $9, $8, 2 ; 9: sll $9, $8, 2 # the "sll" instruction is used for left shift operation
[0040002c] 3c011000 lui $1, 4096 ; 10: sw $9,0x10000000 # and "sw" instruction is used to store the value of $9
[00400030] ac290000 sw $9, 0($1)
```

Kernel Text Segment [80000000]..[80010000]

```
[80000180] 0001d821 addu $27, $0, $1 ; 90: move $k1 $at # Save $at
[80000184] 3c019000 lui $1, -28672 ; 92: sw $v0 $1 # Not re-entrant and we can't trust $sp
[80000188] ac220200 sw $2, 512($1) ; 93: sw $a0 $2 # But we need to use these registers
[8000018c] 3c019000 lui $1, -28672
[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 ExCoDe 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)
```

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.
Attempt to execute non-instruction at 0x00400034

c) Evaluate the expression $(2x+3)^2$ where x is the content in register \$10 based on exercise (a). Store the result in register \$13.

In Decimal –

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

FP Regs Int Regs [10] Data Text

Int Regs [10]

PC = 4194372
EPC = 0
Cause = 0
BadVAddr = 0
Status = 805371664
HI = 0
LO = 1521
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 0
R3 [v1] = 0
R4 [a0] = 4
R5 [a1] = 2147481116
R6 [a2] = 2147481136
R7 [a3] = 2
R8 [t0] = 18
R9 [t1] = 3
R10 [t2] = 36
R11 [t3] = 39
R12 [t4] = 1521
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [a0] = 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] 34070002 ori $7, $0, 2           ; 8: ori $7,$0,2 # Load the value 2 into register 7 using "ori" instruction
[00400028] 34080012 ori $8, $0, 18          ; 9: ori $8,$0,18 # Here is x==18 and x is in the expression (2x+3)^2
[0040002c] 34090003 ori $9, $0, 3           ; 10: ori $9,$0,3 # Load the value 3 (constant value in the expression)into
[00400030] 00e80018 mult $7, $8             ; 11: mult $7,$8 # Use the "mult" instruction to multiply register 7 and
[00400034] 00005012 mflo $10                ; 12: mflo $10 # Here is use "mflo" instruction to move the above value in
[00400038] 01495820 add $11, $10, $9        ; 13: add $11,$10,$9 # Use the "add" instruction to add value of register 10
[0040003c] 016b0018 mult $11, $11           ; 14: mult $11,$11 # then multiplies the value in $11 by itself
[00400040] 00006012 mflo $12                ; 15: mflo $12 # and store final value in register 12
[80000180] 0001d821 addu $27, $0, $1        ; 90: move $k1 $at # Save $at
[80000184] 3c019000 lui $1, -28672         ; 92: sw $v0 $1 # Not re-entrant and we can't trust $sp

```

Kernel Text Segment [80000000]..[80010000]

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.
Attempt to execute non-instruction at 0x00400044

In Hexa Decimal –

QtSpim

File Simulator Registers Text Segment Data Segment Window Help

FP Regs Int Regs [16] Data Text

Int Regs [16]

PC = 400044
EPC = 0
Cause = 0
BadVAddr = 0
Status = 3000ff10
HI = 0
LO = 5f1
R0 [r0] = 0
R1 [at] = 0
R2 [v0] = 0
R3 [v1] = 0
R4 [a0] = 4
R5 [a1] = 7ffff61c
R6 [a2] = 7ffff630
R7 [a3] = 2
R8 [t0] = 12
R9 [t1] = 3
R10 [t2] = 24
R11 [t3] = 27
R12 [t4] = 5f1
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [a0] = 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] 34070002 ori $7, $0, 2           ; 8: ori $7,$0,2 # Load the value 2 into register 7 using "ori" instruction
[00400028] 34080012 ori $8, $0, 18          ; 9: ori $8,$0,18 # Here is x==18 and x is in the expression (2x+3)^2
[0040002c] 34090003 ori $9, $0, 3           ; 10: ori $9,$0,3 # Load the value 3 (constant value in the expression)into
[00400030] 00e80018 mult $7, $8             ; 11: mult $7,$8 # Use the "mult" instruction to multiply register 7 and
[00400034] 00005012 mflo $10                ; 12: mflo $10 # Here is use "mflo" instruction to move the above value in
[00400038] 01495820 add $11, $10, $9        ; 13: add $11,$10,$9 # Use the "add" instruction to add value of register 10
[0040003c] 016b0018 mult $11, $11           ; 14: mult $11,$11 # then multiplies the value in $11 by itself
[00400040] 00006012 mflo $12                ; 15: mflo $12 # and store final value in register 12
[80000180] 0001d821 addu $27, $0, $1        ; 90: move $k1 $at # Save $at
[80000184] 3c019000 lui $1, -28672         ; 92: sw $v0 $1 # Not re-entrant and we can't trust $sp

```

Kernel Text Segment [80000000]..[80010000]

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.
Attempt to execute non-instruction at 0x00400044