# PRELIMINARY REPORT

**Lab 02** 

MUHAMMAD ARHAM KHAN
21701848 - CS
SECTION 06
SPRING 2019

**Dated: 02 March, 2019** 

### **INTERACT WITH USER**

```
interactWithUser:
              #print prompt
                     $a0, interactionPrompt
              la
              li
                     $v0,4
              syscall
              #input integer
              li
                    $v0, 5
              syscall
              #menu selection
              beq $v0, 3, returnToMain
              beq
                     $v0, 1, option1
              beq $v0, 2, option2
                    interactWithUser
              j
       option1:
              #initialize registers
                     $s0, $0, $0
              add
                     $s1, $0, $0
              add
                     $s2, $0, $0
              add
                   $s3, $0, $ra
              add
                     $s4, $0, $0
              add
              #print prompt
                     $a0, convertToDecPrompt
              la
                      $v0,4
              li
              syscall
              #input string
                    $a0, inputBuffer
                     $a1, 30
              li
              move
                    $s0,$a0
              li
                     $v0,8
              syscall
              #initialize stack
              addi $sp, $sp, -20
                     $s0, 0($sp)
              SW
                     $s1, 4($sp)
$s2, 8($sp)
              SW
              SW
                    $s3, 12($sp)
              SW
                    $s4, 16($sp)
              SW
              #got to sub
              jal
                    convertToDec
              #reload stack
                   $s0, 0($sp)
              lw
                     $s1, 4($sp)
              lw
                     $s2, 8($sp)
              ٦w
              lw
                     $s3, 12($sp)
                     $s4, 16($sp)
              #print prompt
              move $ra, $s3
                      $a0, convertToDecOutputPrompt
              la
              li
                      $v0,4
              syscall
              #print string outh
              move $a0, $s4
                     $v0,1
              li
              syscall
              #print newline
                     $a0, newline
              la
              li
                     $v0,4
              syscall
                     interactWithUser
       option2:
              #initialize registers
              add $s0, $0, $0
                     $s1, $0, $0
              add
```

```
add $s2, $0, $0
              add
                   $s3, $0, $ra
              #input string
                 $s4, storageBuffer
              la
                     $a0, reverseNumberPrompt
                   $v0,4
              li
              syscall
              #input integer
              li $v0, 5
              syscall
                   $s0, $v0
              move
              #initlizing stack
                   $s0, 0($sp)
              SW
                     $s1, 4($sp)
              SW
              SW
                    $s2, 8($sp)
              SW
                    $s3, 12($sp)
                    $s4, 16($sp)
              SW
              #go to sub
              jal reverseNumber
              #reload stack
                    $s0, 0($sp)
                    $s1, 4($sp)
              lw
                    $s2, 8($sp)
              ٦w
                    $s3, 12($sp)
$s4, 16($sp)
              lw
              lw
              #print prompt
              move $ra, $s3
              la
                     $a0, reverseNumberOutputPrompt2
              1 i
                     $v0,4
              syscall
              #print string out
              move $a0, $s4
              li
                    $v0,4
              syscall
              #print newline
              la
                   $a0, newline
                    $v0,4
              li
              syscall
                  interactWithUser
returnToMain:
              jr $ra
```

### **CONVERT TO DEC**

```
#reload stack // initialize values
      $s0, 0($sp)
lw
       $s1, 4($sp)
lw
      $s2, 8($sp)
lw
```

convertToDec:

add \$s3, \$0, \$0 ٦w \$s4, 16(\$sp) add \$s6, \$0, \$0 findOctalSize:

```
$s6, 0($s0) #current character
       lb
             $s6, $0, endLoop
      beq
       addi $s1, $s1, 1
             $s1, 4($sp)
       SW
      addi
            $s0, $s0, 1
             findOctalSize
       j
endLoop:
```

addi \$s1, \$s1, -1

```
$s1, 4($sp)
       ٦w
             $s0, 0($sp)
              $s2, 8($sp)
       lw
       addi
              $s2, $s2, 1
goToEnd:
              $s2, $s1, atEnd
       beq
              $s2, $s2, 1
       addi
       addi
             $s0, $s0, 1
              goToEnd
       j
atEnd:
       add
              $s2, $0, $0 #iteration counter
              $s3, $0, $0 #head power counter
       add
              $s4, $0, $0 #sum
       add
        loopAddition:
              beq
                    $s2, $s1, partADone
              add
                     $s5, $0, $0
              addi
                     $s6, $0, 1
              loopPower:
                           $s5, $s3, endloopPower
                     beq
                           $s6, $s6, 8
                      mul
                      addi $s5, $s5, 1
                     j loopPower
              endloopPower:
              lb
                     $s5, 0($s0)
                      $s5, $s5, -48
              addi
                     $s5, $s5, $s6
              mul
                     $s4, $s4, $s5
              add
              SW
                     $s4, 16($sp)
              addi $s2, $s2, 1
              addi
                    $s3, $s3, 1
              addi $s0, $s0, -1
j loopAddition
partADone:
              $v0, $s4
       move
       jr
              $ra
```

### **REVERSE NUMBER**

```
reverseNumber:
```

```
#load stack
   $s0, 0($sp)
       $s1, 4($sp)
lw
      $s2, 8($sp)
٦w
       $s3, 12($sp)
lw
       $s4, 16($sp)
lw
#print in hex
la
    $a0, reverseNumberOutputPrompt1
li
       $v0, 4
syscall
#output hex
move
      $a0, $s0
li
       $v0, 34
syscall
la
      $a0, newline
li
       $v0, 4
syscall
add
     $s5, $0, $0
loopToHex:
            $s3, $s0, 4
       sra
```

```
sll $s1, $s3, 4
      xor $s2, $s0, $s1
      move $s0, $s3
      #if value is greater than 9
      bgt $s2, 9, convertToAlphabet
            $s2, $s2, 48
      addi
      #storing value
      sb $s2, 0($s4)
            $s0, $0, endConversion
      ble
      #increment
      addi $s4, $s4, 1
             loopToHex
      j
convertToAlphabet:
      addi $s2, $s2, 55
      sb
            $s2, 0($s4)
      ble
            $s0, $0, endConversion
      addi $s4, $s4, 1
      j
             loopToHex
endConversion:
      addi $s4, $s4, 1
      sb
           $0, 0($s4)
      move $v0, $s4
      jr
            $ra
```

### **WHOLE PROGRAM**

```
.text
      main:
             jal interactWithUser
             #returns from sub
             li $v0, 10
             syscall
       interactWithUser:
             #print prompt
                   $a0, interactionPrompt
             la
             li
                    $v0,4
             syscall
             #input integer
             li
                    $v0, 5
             syscall
             #menu selection
             beq $v0, 3, returnToMain
             beq $v0, 1, option1
             beq $v0, 2, option2
                    interactWithUser
             j
       option1:
             #initialize registers
             add $s0, $0, $0
                    $s1, $0, $0
             add
                    $s2, $0, $0
             add
             add
                    $s3, $0, $ra
                    $s4, $0, $0
             add
              #print prompt
             la
                    $a0, convertToDecPrompt
             li
             syscall
             #input string
                 $a0, inputBuffer
             la
             li
                    $a1, 30
             move $s0,$a0
```

```
li
             $v0,8
       syscall
       #initialize stack
       addi $sp, $sp, -20
              $s0, 0($sp)
              $s1, 4($sp)
       SW
              $s2, 8($sp)
       SW
       SW
             $s3, 12($sp)
             $s4, 16($sp)
       SW
       #got to sub
       jal
             convertToDec
       #reload stack
             $s0, 0($sp)
       lw
              $s1, 4($sp)
       lw
       lw
              $s2, 8($sp)
       ٦w
              $s3, 12($sp)
              $s4, 16($sp)
       lw
       #print prompt
       move $ra, $s3
       la.
              $a0, convertToDecOutputPrompt
              $v0,4
       syscall
       #print string outh
       move $a0, $s4
       li
              $v0,1
       syscall
       #print newline
       la
             $a0, newline
       li
              $v0,4
       syscall
              interactWithUser
       j
option2:
       #initialize registers
            $s0, $0, $0
       add
              $s1, $0, $0
              $s2, $0, $0
       add
       add
              $s3, $0, $ra
       #input string
       la
              $s4, storageBuffer
              $a0, reverseNumberPrompt
       la
       li
              $v0,4
       syscall
       #input integer
       li
              $v0, 5
       syscall
             $s0, $v0
       move
       #initlizing stack
             $s0, 0($sp)
       SW
              $s1, 4($sp)
              $s2, 8($sp)
       SW
       SW
              $s3, 12($sp)
              $s4, 16($sp)
       SW
       #go to sub
       jal reverseNumber
       #reload stack
       lw
              $s0, 0($sp)
              $s1, 4($sp)
       lw
              $s2, 8($sp)
       lw
              $s3, 12($sp)
              $s4, 16($sp)
       lw
       #print prompt
       move $ra, $s3
              $a0, reverseNumberOutputPrompt2
              $v0,4
       1i
       syscall
       #print string out
```

```
move $a0, $s4
       li
              $v0,4
       syscall
       #print newline
              $a0, newline
              $v0,4
       li
       syscall
       j
              interactWithUser
convertToDec:
       #reload stack // initialize values
              $s0, 0($sp)
              $s1, 4($sp)
       lw
              $s2, 8($sp)
       lw
       add
              $s3, $0, $0
       lw
              $s4, 16($sp)
              $s6, $0, $0
       add
       findOctalSize:
              lb
                      $s6, 0($s0) #current character
                      $s6, $0, endLoop
              beq
              addi $s1, $s1, 1
              SW
                     $s1, 4($sp)
                     $s0, $s0, 1
              addi
                     findOctalSize
              j
       endLoop:
                    $s1, $s1, -1
              addi
                     $s1, 4($sp)
              SW
              lw
                     $s0, 0($sp)
              lw
                     $s2, 8($sp)
              addi
                     $s2, $s2, 1
       goToEnd:
                     $s2, $s1, atEnd
$s2, $s2, 1
              beq
              addi
                     $s0, $s0, 1
              addi
                     goToEnd
              j
       atEnd:
                     $s2, $0, $0 #iteration counter
              add
                      $s3, $0, $0 #head power counter
              add
              add
                      $s4, $0, $0 #sum
               loopAddition:
                            $s2, $s1, partADone
                      beq
                      add
                             $s5, $0, $0
                             $s6, $0, 1
                      addi
                      loopPower:
                                  $s5, $s3, endloopPower
                             beq
                                    $s6, $s6, 8
                             mul
                             addi $s5, $s5, 1
                             j
                                    loopPower
                      endloopPower:
                      lb
                             $s5, 0($s0)
                      addi
                             $s5, $s5, -48
                      mul
                             $s5, $s5, $s6
                             $s4, $s4, $s5
                      add
                             $s4, 16($sp)
                      SW
                            $s2, $s2, 1
                      addi
                      addi
                            $s3, $s3, 1
                      addi $s0, $s0, -1
                             loopAddition
                      j
       partADone:
                      $v0, $s4
              move
              jr
                      $ra
reverseNumber:
       #load stack
           $s0, 0($sp)
       lw
              $s1, 4($sp)
$s2, 8($sp)
       ٦w
       lw
              $s3, 12($sp)
       lw
```

```
$s4, 16($sp)
               #print in hex
                   $a0, reverseNumberOutputPrompt1
              la
                      $v0, 4
              li
              syscall
               #output hex
              move $a0, $s0
                     $v0, 34
              syscall
              la
                     $a0, newline
              li
                      $v0, 4
              syscall
                     $s5, $0, $0
              add
              loopToHex:
                           $s3, $s0, 4
                           $s1, $s3, 4
                      sll
                             $s2, $s0, $s1
                      xor
                            $s0, $s3
                      move
                      #if value is greater than 9
                      bgt $s2, 9, convertToAlphabet
                      addi $s2, $s2, 48
                      #storing value
                      sb $s2, 0($s4)
                      ble
                             $s0, $0, endConversion
                      #increment
                      addi $s4, $s4, 1
                      i
                             loopToHex
              convertToAlphabet:
                      addi $s2, $s2, 55
                             $s2, 0($s4)
                      sb
                      ble
                             $s0, $0, endConversion
                      addi $s4, $s4, 1
                            loopToHex
                      j
              endConversion:
                      addi $s4, $s4, 1
                      sb
                            $0, 0($s4)
                            $v0, $s4
                      move
                      jr
       returnToMain:
              jr
                      $ra
.data
inputBuffer: .space 30
storageBuffer: .space 30
interactionPrompt: .asciiz "Please select one of the options:\n1. Octal to decimal\n2.
reverse Hex\n3. Return to main\nYour selection: "
convertToDecPrompt: .asciiz "Enter octal: "
                             .asciiz "The decimal value is: "
convertToDecOutputPrompt:
            .asciiz "\n"
newline:
test: .asciiz "test\n"
reverseNumberPrompt: .asciiz "Please enter the decimal number: "
reverseNumberOutputPrompt2: .asciiz "Reversed hex: "reverseNumberOutputPrompt1: .asciiz "normal hex: "
```

# **PART D: GENERATING MACHINE INSTRUCTION**

### A. beq \$t0, \$t1, next

if equal, go from: 0x10010054 to 0x10010064, so immediate = (0x10010064 - 0x10010054) / 4 = 0100

000100 01000 01001 0000 0000 0000 0100
--

Hex instructions: 0x11090004

# B. bne \$t2, \$t3, again

if not equal, go from: 0x10010058 to 0x10010040, so immediate = (0x10010040 - 0x10010058) / 4 = -0110

000101 01010 01011 1111 1111 1010
-----------------------------------

Hex instructions: 0x154BFFFA

# C. jr \$ra

000000 11111 00000 00000	00000	001000
--------------------------	-------	--------

Hex instructions: 0x03E00008

# D. j again

if equal, go from: 0x10010070 to 0x10010040, so immediate = (0x10010040 - 0x10010068) / 4 = -1010

000010	0000 0000 0001 0000 0000 0100 00
--------	----------------------------------

Hex instructions: 0x08004010