King Fahd University of Petroleum and Minerals College of Computer Science and Engineering Computer Engineering Department

### COE 301 COMPUTER ORGANIZATION ICS 233: COMPUTER ARCHITECTURE & ASSEMBLY LANGUAGE Term 171 (Fall 2017-2018) Major Exam 1 Saturday Oct. 21, 2017

### Time: 120 minutes, Total Pages: 10

KEY	ID:	Section:

#### Notes:

- Do not open the exam book until instructed
- Answer all questions
- All steps must be shown
- Any assumptions made must be clearly stated
- No calculators are allowed to be used in the exam

Question	Max Points	Score
Q1	28	
Q2	11	
Q3	17	
Total	56	

Dr. Aiman El-Maleh Dr. Marwan Abu-Amara

- (Q1) Fill in the blank in each of the following questions:
  - (1) Assuming 12-bit unsigned number representation, the binary number 1111 1111 0000 is equal to the decimal number <u>4080</u>.
  - (2) Assuming 16-bit signed 2's complement representation, the hexadecimal number FEA0 is equal to the decimal number -352.
  - (3) Two advantages of programming in assembly language are <u>space and time</u> <u>efficiency</u> and <u>accessibility to system hardware</u>.
  - (4) Two advantages of programming in high-level language are <u>programs are portable</u> and <u>program development and maintenance are faster</u>.
  - (5) The instruction set architecture of a processor consists of the instruction set, memory and programmer accessible registers.
  - (6) With a 24-bit address bus and 32-bit data bus, the maximum memory size (assuming byte addressable memory) that can be accessed by a processor is  $2^{24}=16$  MB and the maximum number of bytes that can be read or written in a single cycle is 32/8=4.
  - (7) The advantage of static RAM over dynamic RAM is that it is <u>faster</u> but the disadvantage is that <u>it is less dense and more expensive</u>.
  - (8) Given a magnetic disk with the following properties:
    - Time of one rotation is 8 ms
    - Average seek = 8 ms, Sector = 512 bytes, Track = 200 sectors

The average time to access a block of 100 consecutive sectors is 8 ms + 0.5\*8 ms + 100/200\*8 ms = 16 ms.

(9) Assuming variable Array is defined as shown below:

Array: .word 10 .half 11, 12 .byte 13, 14, 15, 16

The content of register 1 (in hexadecimal) after executing the following sequence of instructions is <u>0x000c000b</u>.

la \$t0, Array lw \$t1, 4(\$t0)

(10) The pseudo instruction *bgt* \$*s*2, *10*, *Next* is implemented by the following minimum MIPS instructions:

<u>slti \$at, \$s2, 11</u> bne \$at, \$0, Next

(11) The pseudo instruction *li \$t0, 0x12345678* is implemented by the following <u>minimum</u> MIPS instructions:

<u>lui \$t0, 0x1234</u> ori \$t0, \$t0, 0x5678

(12) The pseudo instruction *rol* \$*s*0, \$*s*0, 4 (\$*s*0 is rotated to the left by 4 bits and stored in \$*s*0) is implemented by the following <u>minimum</u> MIPS instructions:

<u>srl \$at, \$s0, 28</u> <u>sll \$s0, \$s0, 4</u> <u>or \$s0, \$s0, \$at</u>

- (13) Assuming that \$a0 contains an Alphabetic character, the instruction <u>andi \$a0, \$a0, 0xDF</u> will make the character stored in \$a0 always upper case. Note that the ASCII code of character 'A' is 0x41 while that of character 'a' is 0x61.
- (14) Assume that the instruction *bne* \$*t0*, \$*t1*, *NEXT* is at address 0x00400040 in the text segment, and the label NEXT is at address 0x00400028. Then, the value stored in the assembled instruction for the label NEXT is (0x00400028-0x00400044)/4=FFF9.

(15) Assuming that variable Array is defined as shown below:

Array2: .half -2,-3, 4, 5

After executing the following sequence of instructions, the content of the two registers (in hexadecimal) is t1=000000FF and t2=FFFFFFFD.

la \$t0, Array2 lbu \$t1, 1(\$t0) lh \$t2, 2(\$t0)

(16) Assuming the following data segment, and assuming that the first variable X is given the address 0x10010000, then the addresses for variables Y and Z will be 0x10010006 and 0x10010010.

.data

- X: .byte 10, 11, 12, 13, 14
- Y: .half 15, 16, 17, 18
- Z: .word 19, 20
- (17) To multiply the signed content of register \$t0 by 112 without using multiplication instructions, we use the following minimum MIPS instructions (HINT: 112=16\*7):

<u>sll \$t1, \$t0, 4</u> <u>sll \$t0, \$t1, 3</u> sub \$t0, \$t0, \$t1

#### (Q2) Answer each of the following questions. Show how you obtained your answer:

(i) Given that TABLE is defined as: TABLE: .asciiz "Aiman El-Maleh"

Determine the content of register **\$t0** after executing the following code:

xor \$t0, \$t0, \$t0 la \$t1, TABLE li \$t2, 'a' Next: lbu \$t3, (\$t1) beq \$t3, \$zero, ENL ori \$t3, \$t3, 0x20 addi \$t1, \$t1, 1 bne \$t2, \$t3, Next addi \$t0, \$t0, 1 j Next ENL:

The content of register  $t_0=3$  as the program counts the number of characters equal to 'A' or 'a' in TABLE.

(ii) Determine the content of register \$t1 after executing the following code:

AGAIN:	li \$t0, 0x1234 xor \$t1, \$t1, \$t1 andi \$t2, \$t0, 0xf add \$t1, \$t1, \$t2 srl \$t0, \$t0, 4
	bne \$t0, \$zero, AGAIN

The content of register  $t_{0,x}$  as the program computes the sum of the hexadecimal digits in register  $t_{0,x}$ .

## (iii) Given that TABLE is defined as: TABLE: .word 90, 70, 80, 60, 100

Determine the content of register **\$v0** after executing the following code:

	la	\$a0, TABLE
	addi	\$a1, \$a0, 16
	lw	\$v0, 0(\$a0)
loop:	addi	\$a0, \$a0, 4
	lw	\$t1, 0(\$a0)
	bge	\$t1, \$v0, skip
	move	\$v0, \$t1
skip:	bne	\$a0, \$a1, loop

The content of register  $v_0=0x_3C=60$  as the program computes the minimum of the numbers stored in TABLE.

(Q3) Write <u>separate MIPS</u> assembly code fragments with <u>minimum</u> instructions to implement each of the given requirements. You can use pseudo instructions in your solution.

(i) [10 points] Write a MIPS code fragment that returns the <u>maximum</u> integer value found in a user-specified row number of a 32 × 32 matrix A of 32-bit signed integers. The program should read the desired row number from the user and check that it is in the range between 0 and 31. If not, the program should display the error message "Row number is out of range." and terminate. Otherwise, the program should display the message "Maximum integer in the row is " and the value of the maximum integer found in the specified row, and then terminate. Assume that matrix A is already stored in memory.

```
.data
prompt:
             .asciiz
                         "Please enter a row number between 0 and 31: "
outofrange:
            .asciiz
                         "Row number is out of range.\n"
                         "Maximum integer in the row is "
             .asciiz
outmsg:
.text
.globl main
main:
            $a0,prompt
                         # display prompt string
      la
      li
            $v0,4
      syscall
      li
            $v0,5
                        # read row number into $t0
      syscall
      move $t0,$v0
      bltz
           $t0,error
                         # check row boundary
      addiu $t1,$t0,-31 # If $t0 > 31, then result of ($t0-31) > 0
      bgtz $t1,error
      la
            $t1,A
                         # compute starting location of 1st element in desired row
      s11
            $t2,$t0,5
                         # $t2 = i*32
                                         (ixCOL+0)
      s11
                         # $t2 = i*32*4 (ixCOL+0)x(int size)
            $t2,$t2,2
      addu $t2,$t1,$t2 # $t2 = address of 1st element in desired row
      li
            $t3,31
                         \# \max j = 31
      lw
            $t4,0($t2)
                         # read 1st element of desired row & set as maximum
loop:
      addiu $t2,$t2,4
                         # increment index to point to next row element
                         # read next element of desired row
      lw
            $t5,0($t2)
      ble
            $t5,$t4,next # next element ($t5) <= current max ($t4)?</pre>
      move $t4,$t5
                         # No -> set max ($t4) = next element ($t5)
next:
      addiu $t3,$t3,-1
                         # prepare for next row element
      bgtz $t3,loop
            $a0,outmsg
      la
                         # display prompt string
      1i
            $v0,4
      syscall
      move $a0,$t4
                        # output $t4 = maximum in desired row
      1i
            $v0,1
      syscall
      j
            exit
error:
            $a0,outofrange
      la
      1i
            $v0,4
      syscall
exit:
            $v0,10
                       # exit
      li
      syscall
```

(ii) [7 points] Given two arrays A and B, write the smallest MIPS assembly fragment for the following computation. Assume that register \$s0 will be used to store cnt and assume that the following registers have the mentioned values: register \$s1 = number of elements, N, in each array, register \$s2 = base address of the array A, and register \$s3 = base address of the array B. Each array element is a 32-bit signed integer. Assume that N > 0. Insert comments to clarify the meaning of instructions and the use of registers.

```
int cnt = 0;
      for (i=0; i != N; i++) {
         if (((A[i] - B[i]) > 5) || ((B[i] - A[i]) > 5)) cnt = cnt + 1;
      }
     li
           $s0,0
                          # $s0 = cnt = 0
loop:
     lw
           $t0,0($s2)
                          # $t0 = A[i]
           $t1,0($s3)
     lw
                          # $t1 = B[i]
      addiu $t2,$t0,5
                          # $t2 = A[i]+5
     addiu $t3,$t1,5
                          # $t3 = B[i]+5
                          # Check if (A[i]-B[i]>5)
     bgt
           $t0,$t3,incr
                          # Check if (B[i]-A[i]>5)
     ble
           $t1,$t2,done
incr:
     addiu $s0,$s0,1
                          # cnt++
done:
     addiu $s2,$s2,4
                          # point to A[i+1]
      addiu $s3,$s3,4
                          # point to B[i+1]
                          # decrement loop index
      addiu $s1,$s1,-1
      bne $s1,$0,loop
```

# **MIPS Instructions:**

$\begin{array}{ c c c c c c c c c c c c c c c c c c c$	Inet	ruction	Meaning				P_T	vne Er	rmat		
$\begin{array}{c c c c c c c c c c c c c c c c c c c $				00 -	0 10	- ¢c					f = 0x20
sub         \$s1, \$s2, \$s3         \$s1 = \$s2 - \$s3         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x22           subu         \$s1, \$s2, \$s3         \$s1 = \$s2 - \$s3         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x23           Instruction         Meaning         R-Type         Format           and \$s1, \$s2, \$s3         \$s1 = \$s2 \$s2 \$s3<					_		_				
subu \$s1, \$s2, \$s3         \$s1 = \$s2 - \$s3         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x23           Instruction         Meaning         R-Type Format           and \$s1, \$s2, \$s3         \$s1 = \$s2 & \$s3         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x23           or \$s1, \$s2, \$s3         \$s1 = \$s2 ^ \$s3         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x27           Instruction         Meaning         R-Type Format           sil \$s1, \$s2, \$s3         \$s1 = \$s2 < < \$s3					_		_				
Instruction         Meaning         R-Type Format           and \$s1, \$s2, \$s3         \$s1 = \$s2 & \$s3         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x24           or         \$s1, \$s2, \$s3         \$s1 = \$s2 & \$s3         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x26           xr \$s1, \$s2, \$s3         \$s1 = \$s2 ^ \$s3         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x26           nor \$s1, \$s2, \$s3         \$s1 = ~(\$s2]\$s3)         op = 0         rs = \$s2         rt = \$s3         rd = \$s1         sa = 0         f = 0x26           nstruction         Meaning         R-Type Format         sa = 0         f = 0x27           Instruction         Meaning         R-Type Format         sa = 10         f = 0           srl \$s1, \$s2, 10         \$s1 = \$s2 <<10							_				
$\begin{array}{c c c c c c c c c c c c c c c c c c c $				op -	0 13	- <b>J</b> S	2  IL -	φοσητα	- 431	5a – U	1-0723
or       \$s1, \$s2, \$s3       \$s1 = \$s2   \$s3       op = 0       rs = \$s2       rt = \$s3       rd = \$s1       sa = 0       f = 0x25         xor       \$s1, \$s2, \$s3       \$s1 = \$s2 ^ \$s3       op = 0       rs = \$s2       rt = \$s3       rd = \$s1       sa = 0       f = 0x25         Instruction       Meaning       R-Type       Format       sa = 10       f = 0x27         Instruction       Meaning       R-Type       Format       sa = 10       f = 0x25         s1, \$s2, \$s3       \$s1 = \$s2 <<10       op = 0       rs = 0       rt = \$s2       rd = \$s1       sa = 0       f = 0x25         s1, \$s2, \$s3       \$s1 = \$s2 <<<10       op = 0       rs = 0       rt = \$s2       rd = \$s1       sa = 0       f = 0         sr \$s1, \$s2, \$s3       \$s1 = \$s2 <<<53       op = 0       rs = 0       rt = \$s2       rd = \$s1       sa = 0       f = 3         s1/s \$s1, \$s2, \$s3       \$s1 = \$s2 <<\$s3       op = 0       rs = \$s3       rt = \$s2       rd = \$s1       sa = 0       f = 6         s1/s \$s1, \$s2, 10       \$s1 = \$s2 < 10       op = 0x8       rs = \$s2       rd = \$s1       sa = 0       f = 6         s1/s \$s2, 10       \$s1 = \$s2 + 10       op = 0x8       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10 <td></td>											
$\begin{array}{c c c c c c c c c c c c c c c c c c c $				<u> </u>			_				
nor       \$s1, \$s2, \$s3       \$s1 = -(\$s2 \$s3)       op = 0       rs = \$s2       rt = \$s3       rd = \$s1       sa = 0       f = 0x27         Instruction       Meaning       R-Type Format         s1       \$s1, \$s2, 10       \$s1 = \$s2 < < 10       op = 0       rs = 0       rt = \$s2       rd = \$s1       sa = 10       f = 0         sr1       \$s1, \$s2, 10       \$s1 = \$s2 >> 10       op = 0       rs = 0       rt = \$s2       rd = \$s1       sa = 10       f = 0         sr1       \$s1, \$s2, 10       \$s1 = \$s2 >> 10       op = 0       rs = 0       rt = \$s2       rd = \$s1       sa = 10       f = 0         satt       \$s1, \$s2, 10       \$s1 = \$s2 >> 10       op = 0       rs = 0       rt = \$s2       rd = \$s1       sa = 10       f = 3         sllv       \$s1, \$s2, 10       \$s1 = \$s2 <<       \$s3       op = 0       rs = \$s3       rt = \$s1       sa = 0       f = 4         srtv       \$s1, \$s2, 10       \$s1 = \$s2 + 10       op = 0x8       rs = \$s2       rt = \$s1       sa = 0       f = 6         srav       \$s1, \$s2, 10       \$s1 = \$s2 + 10       op = 0x8       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         addi       \$s1, \$s2, 10       \$s1 = \$s2 + 10       op = 0x6       rs = \$				· ·							
$\begin{array}{ c c c c c } \hline \textbf{Meaning} & \textbf{R-Type Format} \\ \hline \textbf{s1} & \textbf{s1}, \textbf{s2}, 10 & \textbf{s1} = \textbf{s2} 2 << 10 & op = 0 & rs = 0 & rt = \textbf{s2} rd = \textbf{s1} & \textbf{sa} = 10 & f = 0 \\ sri & \textbf{s1}, \textbf{s2}, 10 & \textbf{s1} = \textbf{s2} 2 >> 10 & op = 0 & rs = 0 & rt = \textbf{s2} rd = \textbf{s1} & \textbf{sa} = 10 & f = 2 \\ sra & \textbf{s1}, \textbf{s2}, 10 & \textbf{s1} = \textbf{s2} 2 >> 10 & op = 0 & rs = 0 & rt = \textbf{s2} rd = \textbf{s1} & \textbf{sa} = 10 & f = 3 \\ sllv & \textbf{s1}, \textbf{s2}, \textbf{s3} & \textbf{s1} = \textbf{s2} 2 << \textbf{s3} & op = 0 & rs = 0 & rs = \textbf{s3} & rt = \textbf{s2} rd = \textbf{s1} & \textbf{sa} = 10 & f = 3 \\ sllv & \textbf{s1}, \textbf{s2}, \textbf{s3} & \textbf{s1} = \textbf{s2} 2 << \textbf{s3} & op = 0 & rs = \textbf{s3} & rt = \textbf{s2} rd = \textbf{s1} & \textbf{sa} = 0 & f = 4 \\ srlv & \textbf{s1}, \textbf{s2}, \textbf{s3} & \textbf{s1} = \textbf{s2} 2 >> \textbf{s3} & op = 0 & rs = \textbf{s3} & rt = \textbf{s2} rd = \textbf{s1} & \textbf{sa} = 0 & f = 6 \\ srav & \textbf{s1}, \textbf{s2}, \textbf{s3} & \textbf{s1} = \textbf{s2} 2 >> \textbf{s3} & op = 0 & rs = \textbf{s3} & rt = \textbf{s3} rt = \textbf{s3} & rt = $				<u> </u>	_		_			-	
$\begin{array}{c c c c c c c c c c c c c c c c c c c $	nor	\$s1, \$s2, \$s3	\$s1 = ~(\$s2 \$s3)	op =	: 0  rs	= \$s	2   rt =	\$s3   rd	= \$s1	sa = 0	f = 0x27
$\begin{array}{c c c c c c c c c c c c c c c c c c c $	Inst	ruction	Meaning				R-T	ype Fo	orma	t	
$\begin{array}{c c c c c c c c c c c c c c c c c c c $	sll	\$s1,\$s2,10	\$s1 = \$s2 << 10	op :	= 0   rs	s = 0	rt =	\$s2 rd	= \$s1	sa = 10	f = 0
$ \frac{\text{slv} \$s1,\$s2,\$s3}{\text{sr} = \$s2,<<\$s3} \text{ op = 0} rs = \$s3}{rt = \$s2} rt = \$s2} rt = \$s1} \frac{\text{sa = 0}}{\text{sa = 0}} f = 4 \\ \frac{\text{srv} \$s1,\$s2,\$s3}{\text{ss} = \$s1,\$s2,\$s3} \frac{\$s1 = \$s2,>>\$s3}{\text{ss} = \$s2,>>\$s3} rt = \$s2} rt = \$s2} rt = \$s1}{rt = \$s1} \frac{\text{sa = 0}}{\text{sa = 0}} f = 6 \\ \frac{\text{srav} \$s1,\$s2,\$s3}{\text{ss} = \$s2,\$s3} \frac{\$s1 = \$s2,>>\$s3}{\text{ss} = \$s2,>>\$s3} rt = \$s2} rt = \$s1} \frac{\text{sa = 0}}{\text{sa = 0}} f = 7 \\ \frac{\text{Instruction}}{\text{addi} \$s1,\$s2,\$s3} \frac{\$s1 = \$s2,>>\$s3} rs = \$s2, rt = \$s1}{rt = \$s1} rt = 10 \\ \frac{\text{addi} \$s1,\$s2,10}{\text{ss} = \$s2,10} \frac{\$s2 + 10}{\text{op = 0x8}} rs = \$s2} rt = \$s1} rt = 10 \\ \frac{\text{addi} \$s1,\$s2,10}{\text{ss} = \$s2,10} \frac{\$s2 + 10}{\text{op = 0x9}} rs = \$s2} rt = \$s1} rt = 10 \\ \frac{\text{addi} \$s1,\$s2,10}{\text{ss} = \$s2,10} rs = \$s2,10 \\ \text{op = 0xc}}{ss2,11} rs = 10 \\ \frac{\text{addi} \$s1,\$s2,10}{\text{ss} = \$s2,10} rs = \$s2,11} rt = 10 \\ \frac{\text{andi} \$s1,\$s2,10}{\text{ss} = \$s2,10} rs = \$s2,10 \\ rs = \$s2} rt = \$s1} rt = 10 \\ rt = 10 \\ \frac{\text{andi} \$s1,\$s2,10}{\text{ss} = \$s2,10} rs = 10 \\ rs = \$s2,11} rt = 10 \\ rt = 10 \\ \frac{\text{andi} \$s1,\$s2,10}{\text{ss} = \$s2,10} rs = 10 \\ rt = \$s1,10 \\ rt = 10 \\ rt $	srl	\$s1,\$s2,10		<u> </u>							
$\begin{array}{c c c c c c c c c c c c c c c c c c c $	sra	\$s1, \$s2, 10	\$s1 = \$s2 >> 10	<u> </u>			_				f = 3
$\begin{array}{ c c c c c c c c c c c c c c c c c c c$	sllv	\$s1,\$s2,\$s3	\$s1 = \$s2 << \$s3	<u> </u>			_				f = 4
Instruction         Meaning         I-Type Format           addi         \$\$1, \$\$2, 10         \$\$1 = \$\$2 + 10         op = 0x8         rs = \$\$2         rt = \$\$1         imm <sup>16</sup> = 10           addiu         \$\$1, \$\$2, 10         \$\$1 = \$\$2 + 10         op = 0x9         rs = \$\$2         rt = \$\$1         imm <sup>16</sup> = 10           andi         \$\$1, \$\$2, 10         \$\$1 = \$\$2 & 10         op = 0xe         rs = \$\$2         rt = \$\$1         imm <sup>16</sup> = 10           ori         \$\$1, \$\$2, 10         \$\$1 = \$\$2 & 10         op = 0xc         rs = \$\$2         rt = \$\$1         imm <sup>16</sup> = 10           ori         \$\$1, \$\$2, 10         \$\$1 = \$\$2 & 10         op = 0xe         rs = \$\$2         rt = \$\$1         imm <sup>16</sup> = 10           xori         \$\$1, \$\$2, 10         \$\$1 = \$\$2 ^ 10         op = 0xe         rs = \$\$2         rt = \$\$1         imm <sup>16</sup> = 10           lui         \$\$1, 10         \$\$1 = 10 <<	srlv			<u> </u>			_				
addi       \$s1, \$s2, 10       \$s1 = \$s2 + 10       op = 0x8       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         addiu       \$s1, \$s2, 10       \$s1 = \$s2 + 10       op = 0x9       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         andi       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0x0       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         andi       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0xc       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         ori       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0xc       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         xori       \$s1, \$s2, 10       \$s1 = \$s2 ^ 10       op = 0xe       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         lui       \$s1, 10       \$s1 = \$s2 ^ 10       op = 0xe       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         lui       \$s1, 10       \$s1 = 10 << 16	srav	\$s1,\$s2,\$s3	\$s1 = \$s2 >> \$s3	op :	= 0   rs	5 = \$	s3 rt =	\$s2 rd	= \$s1	sa = 0	f = 7
addi       \$s1, \$s2, 10       \$s1 = \$s2 + 10       op = 0x8       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         addiu       \$s1, \$s2, 10       \$s1 = \$s2 + 10       op = 0x9       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         andi       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0x0       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         andi       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0xc       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         ori       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0xc       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         xori       \$s1, \$s2, 10       \$s1 = \$s2 ^ 10       op = 0xe       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         lui       \$s1, 10       \$s1 = \$s2 ^ 10       op = 0xe       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         lui       \$s1, 10       \$s1 = 10 << 16	Inst	ruction	Meaning				I-T	vpe Fo	ormai	t	
andi       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0xc       rs = \$s2       rt = \$s1       imm16 = 10         ori       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0xd       rs = \$s2       rt = \$s1       imm16 = 10         xori       \$s1, \$s2, 10       \$s1 = \$s2 & 10       op = 0xd       rs = \$s2       rt = \$s1       imm16 = 10         ui       \$s1, \$s2, 10       \$s1 = \$s2 ^ 10       op = 0xf       0       rt = \$s1       imm16 = 10         lui       \$s1, 10       \$s1 = \$10 <<<16       op = 0xf       0       rt = \$s1       imm16 = 10         lui       \$s1, 10       \$s1 = \$10 <<<16       op = 0xf       0       rt = \$s1       imm16 = 10         lui       \$s1, 10       \$s1 = \$10 <<<16       op = 0xf       0       rt = \$s1       imm16 = 10         lui       \$s1, 10       \$s1 = \$10 <<<16       op = 0xf       0       rt = \$s1       imm16 = 10         lui       \$s1, \$s2, 10       \$s1 = \$10 <<<16       op 6 = 2       imm16       imm16 = 10         lui       pinne trift(rs = rt)       op 6 = 2       rs 5       rt 5       imm16       imm16         blez       rs, label       branch if (rs < 0)       op 6 = 1       rs 5       0       imm16				ор	= 0x8	rs :					= 10
ori       \$s1, \$s2, 10       \$s1 = \$s2   10       op = 0xd $rs = $s2$ $rt = $s1$ $imm^{16} = 10$ xori       \$s1, \$s2, 10       \$s1 = \$s2 ^ 10       op = 0xe $rs = $s2$ $rt = $s1$ $imm^{16} = 10$ lui       \$s1, 10       \$s1 = 10 << 16	addii					= \$s2			imm <sup>16</sup> = 10		
xori       \$s1, \$s2, 10       \$s1 = \$s2 ^ 10       op = 0xe       rs = \$s2       rt = \$s1       imm <sup>16</sup> = 10         lui       \$s1, 10       \$s1 = 10 << 16       op = 0xf       0       rt = \$s1       imm <sup>16</sup> = 10         Instruction       Meaning       Format       Format       imm <sup>16</sup> = 10         j       label       jump to label       op <sup>6</sup> = 2       Format         beq       rs, rt, label       branch if (rs == rt)       op <sup>6</sup> = 4       rs <sup>5</sup> rt <sup>5</sup> imm <sup>16</sup> bne       rs, rt, label       branch if (rs == rt)       op <sup>6</sup> = 5       rs <sup>5</sup> rt <sup>5</sup> imm <sup>16</sup> blez       rs, label       branch if (rs <=0)       op <sup>6</sup> = 7       rs <sup>5</sup> 0       imm <sup>16</sup> bltz       rs, label       branch if (rs <0)       op <sup>6</sup> = 1       rs <sup>5</sup> 0       imm <sup>16</sup> blgz       rs, label       branch if (rs <0)       op <sup>6</sup> = 1       rs <sup>5</sup> 0       imm <sup>16</sup> blgz       rs, label       branch if (rs <=0)       op <sup>6</sup> = 1       rs <sup>5</sup> 0       imm <sup>16</sup> blgz       rs, label       branch if (rs <=0)       op <sup>6</sup> = 1       rs <sup>5</sup> 0       ox         blgz       rs, label       branch if (rs <	andi	\$s1, \$s2, 10	\$s1 = \$s2 & 10								
Iui $\$s1, 10$ $\$s1 = 10 << 16$ $op = 0xf$ $0$ $rt = \$s1$ $imm^{16} = 10$ InstructionMeaning $op^6 = 2$ Formatjlabeljump to label $op^6 = 2$ $imm^{16}$ beqrs, rt, labelbranch if (rs == rt) $op^6 = 4$ $rs^5$ $rt^5$ $imm^{16}$ bners, rt, labelbranch if (rs != rt) $op^6 = 5$ $rs^5$ $rt^5$ $imm^{16}$ blezrs, labelbranch if (rs <= 0)	ori				_ ·						
$\begin{array}{ c c c c c } \hline Instruction & Meaning & Format \\ \hline j & label & jump to label & op^6 = 2 & imm^{26} \\ \hline beq & rs, rt, label & branch if (rs == rt) & op^6 = 4 & rs^5 & rt^5 & imm^{16} \\ \hline bne & rs, rt, label & branch if (rs != rt) & op^6 = 5 & rs^5 & rt^5 & imm^{16} \\ \hline blez & rs, label & branch if (rs <= 0) & op^6 = 6 & rs^5 & 0 & imm^{16} \\ \hline blez & rs, label & branch if (rs > 0) & op^6 = 7 & rs^5 & 0 & imm^{16} \\ \hline bltz & rs, label & branch if (rs <= 0) & op^6 = 1 & rs^5 & 0 & imm^{16} \\ \hline bltz & rs, label & branch if (rs > 0) & op^6 = 1 & rs^5 & 0 & imm^{16} \\ \hline bltz & rs, label & branch if (rs >= 0) & op^6 = 1 & rs^5 & 1 & imm^{16} \\ \hline bltz & rs, label & branch if (rs >= 0) & op^6 = 1 & rs^5 & 1 & imm^{16} \\ \hline bltz & rs, label & branch if (rs >= 0) & op^6 = 1 & rs^5 & 1 & imm^{16} \\ \hline sltz & rd, rs, rt & rd=(rs < rt?1:0) & op^6 = 0 & rs^5 & rt^5 & rd^5 & 0 & 0x2a \\ \hline sltu & rd, rs, rt & rd=(rs < rt?1:0) & op^6 = 0 & rs^5 & rt^5 & rd^5 & 0 & 0x2b \\ \hline slti & rt, rs, imm^{16} & rt=(rs < imm?1:0) & 0xa & rs^5 & rt^5 & imm^{16} \\ \hline \end{array}$		· · ·									
$\begin{array}{c c c c c c c c c c } j & label & jump to label & op^6 = 2 & imm^{26} \\ \hline beq rs, rt, label & branch if (rs == rt) & op^6 = 4 & rs^5 & rt^5 & imm^{16} \\ \hline bne rs, rt, label & branch if (rs != rt) & op^6 = 5 & rs^5 & rt^5 & imm^{16} \\ \hline blez rs, label & branch if (rs <= 0) & op^6 = 6 & rs^5 & 0 & imm^{16} \\ \hline bgtz rs, label & branch if (rs > 0) & op^6 = 7 & rs^5 & 0 & imm^{16} \\ \hline bltz rs, label & branch if (rs < 0) & op^6 = 1 & rs^5 & 0 & imm^{16} \\ \hline bgez rs, label & branch if (rs >= 0) & op^6 = 1 & rs^5 & 1 & imm^{16} \\ \hline bgez rs, label & branch if (rs >= 0) & op^6 = 1 & rs^5 & 1 & imm^{16} \\ \hline \hline bgez rs, label & branch if (rs >= 0) & op^6 = 1 & rs^5 & 1 & imm^{16} \\ \hline sltz rd, rs, rt & rd=(rs < rt?1:0) & op^6 = 0 & rs^5 & rt^5 & rd^5 & 0 & 0x2a \\ sltu rd, rs, rt & rd=(rs < rt?1:0) & op^6 = 0 & rs^5 & rt^5 & rd^5 & 0 & 0x2b \\ slti rt, rs, imm^{16} & rt=(rs < imm?1:0) & 0xa & rs^5 & rt^5 & imm^{16} \\ \hline \end{array}$	lui	\$s1, 10	\$s1 = 10 << 16	op	op = 0xf 0 rt =		rt = \$s	1	Imm <sup>16</sup>	= 10	
beqrs, rt, labelbranch if (rs == rt) $op^6 = 4$ $rs^5$ $rt^5$ $imm^{16}$ bners, rt, labelbranch if (rs != rt) $op^6 = 5$ $rs^5$ $rt^5$ $imm^{16}$ blezrs, labelbranch if (rs<=0) $op^6 = 6$ $rs^5$ 0 $imm^{16}$ bgtzrs, labelbranch if (rs > 0) $op^6 = 7$ $rs^5$ 0 $imm^{16}$ blezrs, labelbranch if (rs < 0) $op^6 = 7$ $rs^5$ 0 $imm^{16}$ bltzrs, labelbranch if (rs < 0) $op^6 = 1$ $rs^5$ 0 $imm^{16}$ blezrs, labelbranch if (rs >=0) $op^6 = 1$ $rs^5$ 1 $imm^{16}$ bltzrs, labelbranch if (rs>=0) $op^6 = 1$ $rs^5$ 1 $imm^{16}$ blezrs, labelbranch if (rs>=0) $op^6 = 1$ $rs^5$ 1 $imm^{16}$ blezrs, labelbranch if (rs>=0) $op^6 = 1$ $rs^5$ 1 $imm^{16}$ blezrs, labelbranch if (rs>=0) $op^6 = 1$ $rs^5$ 1 $imm^{16}$ blezrs, rk $rd=(rsop^6 = 0rs^5rt^5rd^500x2aslturd, rs, rtrd=(rsop^6 = 0rs^5rt^5rd^500x2bsltirt, rs, imm^{16}rt=(rs0xars^5rt^5imm^{16}$	Inst	ruction	Meaning	Format							
bne       rs, rt, label       branch if (rs != rt)       op6 = 5       rs5       rt5       imm16         blez       rs, label       branch if (rs<=0)	j	label	jump to label		op <sup>6</sup> = 2			imm <sup>26</sup>			
blezrs, labelbranch if (rs<=0)op6 = 6rs50imm16bgtzrs, labelbranch if (rs > 0)op6 = 7rs50imm16bltzrs, labelbranch if (rs < 0)	beq	rs, rt, label	branch if (rs ==	rt)	op6 :	= 4	rs <sup>5</sup>	rt <sup>5</sup>		imm <sup>16</sup>	5
bgtzrs, labelbranch if (rs > 0)op6 = 7rs50imm16bltzrs, labelbranch if (rs < 0)	bne	rs, rt, label	branch if (rs !=	rt)	op6 :	= 5	rs <sup>5</sup>	rt <sup>5</sup>		imm <sup>16</sup>	5
bltzrs, labelbranch if (rs < 0)op6 = 1rs50imm16bgezrs, labelbranch if (rs>=0)op6 = 1rs51imm16InstructionMeaningFormatsltrd, rs, rtrd=(rs <rt?1:0)< td="">op6 = 0rs5rt5rd500x2aslturd, rs, rtrd=(rs<rt?1:0)< td="">op6 = 0rs5rt5rd500x2bsltirt, rs, imm16rt=(rs<imm?1:0)< td="">0xars5rt5imm16</imm?1:0)<></rt?1:0)<></rt?1:0)<>	blez	rs, label	branch if (rs<=0	))	op6 :	= 6	rs <sup>5</sup>	0		imm <sup>16</sup>	5
bgez rs, labelbranch if (rs>=0)op6 = 1rs51imm16InstructionMeaningFormatslt rd, rs, rtrd=(rs <rt?1:0)< td="">op6 = 0rs5rt5rd500x2asltu rd, rs, rtrd=(rs<rt?1:0)< td="">op6 = 0rs5rt5rd500x2bsltu rd, rs, rtrd=(rs<rt?1:0)< td="">0xars5rt5rd500x2b</rt?1:0)<></rt?1:0)<></rt?1:0)<>	bgtz	rs, label	. , ,		op6 :	= 7	rs <sup>5</sup>	0		imm <sup>16</sup>	6
$\begin{array}{c c c c c c c c c c c c c c c c c c c $	bltz	rs, label			op <sup>6</sup> = 1		rs <sup>5</sup>	0	imm <sup>16</sup>		6
slt       rd, rs, rt       rd=(rs <rt?1:0)< th="">       op<sup>6</sup> = 0       rs<sup>5</sup>       rt<sup>5</sup>       rd<sup>5</sup>       0       0x2a         sltu       rd, rs, rt       rd=(rs<rt?1:0)< td="">       op<sup>6</sup> = 0       rs<sup>5</sup>       rt<sup>5</sup>       rd<sup>5</sup>       0       0x2b         slti       rt, rs, imm<sup>16</sup>       rt=(rs<imm?1:0)< td="">       0xa       rs<sup>5</sup>       rt<sup>5</sup>       imm<sup>16</sup></imm?1:0)<></rt?1:0)<></rt?1:0)<>	bgez	z rs, label	branch if (rs>=0)		op <sup>6</sup> = 1 rs <sup>5</sup>		rs <sup>5</sup>	1	imm <sup>16</sup>		3
slt       rd, rs, rt       rd=(rs <rt?1:0)< th="">       op<sup>6</sup> = 0       rs<sup>5</sup>       rt<sup>5</sup>       rd<sup>5</sup>       0       0x2a         sltu       rd, rs, rt       rd=(rs<rt?1:0)< td="">       op<sup>6</sup> = 0       rs<sup>5</sup>       rt<sup>5</sup>       rd<sup>5</sup>       0       0x2b         slti       rt, rs, imm<sup>16</sup>       rt=(rs<imm?1:0)< td="">       0xa       rs<sup>5</sup>       rt<sup>5</sup>       imm<sup>16</sup></imm?1:0)<></rt?1:0)<></rt?1:0)<>	Inst	ruction	Meaning					Forr	nat		
sltu         rd, rs, rt         rd=(rs <rt?1:0)< th="">         op<sup>6</sup> = 0         rs<sup>5</sup>         rt<sup>5</sup>         rd<sup>5</sup>         0         0x2b           slti         rt, rs, imm<sup>16</sup>         rt=(rs<imm?1:0)< td="">         0xa         rs<sup>5</sup>         rt<sup>5</sup>         imm<sup>16</sup></imm?1:0)<></rt?1:0)<>			_		op <sup>6</sup> = 0		rs <sup>5</sup>			0	0x2a
slti rt, rs, imm <sup>16</sup> rt=(rs <imm?1:0) 0xa="" rs<sup="">5 rt<sup>5</sup> imm<sup>16</sup></imm?1:0)>					· ·					+ +	
	L				•		rs <sup>5</sup>	rt⁵			
	sltiu					_	rs <sup>5</sup>	rt <sup>5</sup>			

Page 10 of 10

Instruction		Meaning	I-Type Format				
lb	rt, imm <sup>16</sup> (rs)	rt = MEM[rs+imm <sup>16</sup> ]	0x20	rs <sup>5</sup>	rt⁵	imm <sup>16</sup>	
lh	rt, imm <sup>16</sup> (rs)	rt = MEM[rs+imm <sup>16</sup> ]	0x21	rs <sup>5</sup>	rt⁵	imm <sup>16</sup>	
W	rt, imm <sup>16</sup> (rs)	rt = MEM[rs+imm <sup>16</sup> ]	0x23	rs <sup>5</sup>	rt⁵	imm <sup>16</sup>	
lbu	rt, imm <sup>16</sup> (rs)	rt = MEM[rs+imm <sup>16</sup> ]	0x24	rs <sup>5</sup>	rt⁵	imm <sup>16</sup>	
lhu	rt, imm <sup>16</sup> (rs)	rt = MEM[rs+imm <sup>16</sup> ]	0x25	rs <sup>5</sup>	rt⁵	imm <sup>16</sup>	
sb	rt, imm <sup>16</sup> (rs)	MEM[rs+imm <sup>16</sup> ] = rt	0x28	rs <sup>5</sup>	rt⁵	imm <sup>16</sup>	
sh	rt, imm <sup>16</sup> (rs)	MEM[rs+imm <sup>16</sup> ] = rt	0x29	rs⁵	rt⁵	imm <sup>16</sup>	
SW	rt, imm <sup>16</sup> (rs)	MEM[rs+imm <sup>16</sup> ] = rt	0x2b	rs <sup>5</sup>	rt⁵	imm <sup>16</sup>	

# **Syscall Services:**

Service	\$v0	Arguments / Result
Print Integer	1	<pre>\$a0 = integer value to print</pre>
Print Float	2	<pre>\$f12 = float value to print</pre>
Print Double	3	<pre>\$f12 = double value to print</pre>
Print String	4	<pre>\$a0 = address of null-terminated string</pre>
Read Integer	5	Return integer value in <mark>\$v0</mark>
Read Float	6	Return float value in <mark>\$f0</mark>
Read Double	7	Return double value in <mark>\$f0</mark>
Read String	8	<pre>\$a0 = address of input buffer \$a1 = maximum number of characters to read</pre>
Exit Program	10	
Print Char	11	<pre>\$a0 = character to print</pre>
Read Char	12	Return character read in \$v0