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 151 (Fall 2015-2016)
Major Exam 1
Saturday Oct. 10, 2015

Time: 120 minutes, Total Pages: 9

Name:_____ 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

Question	Max Points	Score
Q1	35	
Q2	25	
Total	60	

Dr. Aiman El-Maleh
Dr. Mayez Al-Muhammad

[35 Points]

(Q1) Fill in the blank in each of the following questions:

(1) Assuming 12-bit unsigned representation, the binary number 1111 0000 1111 is equal to the decimal number _____.

(2) Assuming 12-bit signed 2`s complement representation, the hexadecimal number FC0 is equal to the decimal number _____.

(2) Accessibility to hardware resources is an advantage of programming in _____ language.

(3) Code portability is an advantage of programming in _____ language.

(4) With a 36-bit address bus and 64-bit data bus, the maximum memory size (assuming byte addressable memory) that can be accessed by a processor is _____ and the maximum number of bytes that can be read or written in a single cycle is _____.

(5) The bandwidth mismatch between the speed of processor and the speed of main-memory is alleviated by using _____.

(6) The advantage of dynamic RAM over static RAM is that it is _____ and _____ but the disadvantage is _____.

(7) The instruction set architecture of a processor consists of _____
_____.

(8) Assuming that the CPU has just read a 32-bit MIPS instruction from the address 0x00400008. Then, the address of the next instruction that this CPU is going to read is _____.

(9) 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 20 consecutive sectors is _____ ms.

(10) The pseudo instruction *neg $s2, $s1 (*$s2 is computed as the negative value of $s1) is implemented by the following minimum MIPS instructions:

_____
_____
_____

(11) The pseudo instruction *ble $s2, $s1, Next* is implemented by the following minimum MIPS instructions:

_____
_____
_____

(12) The pseudo instruction *rol $s0, $s0, 8 (*$s0 is rotated to the left by 8 bits and stored in $s0) is implemented by the following minimum MIPS instructions:

_____
_____
_____

(13) Assuming that $a0 contains an Alphabetic character, the instruction _____ will guarantee that the character in $a0 is an upper case character. 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 0x00400020 in the text segment, and the label NEXT is at address 0x00400010. Then, the address stored in the assembled instruction for the label NEXT is _____.

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

Array: .byte 1, 2, -3, 4

After executing the following sequence of instructions, the content of the three registers is $t1=_____, $t2=_____, and $t3=_____.

```
la   $t0, Array
lb   $t1, 2($t0)
lh   $t2, 2($t0)
lw  $t3, 0($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 _____ and _____.

```
.data
X:      .byte  1
Y:      .half   2, 3
Z:      .word 4
```

(17) To multiply the **signed** content of register $t0 by 127.75 without using multiplications and division instructions, we use the following MIPS instructions:

_____
_____
_____

(18) The condition for which the data stored in $t0 must satisfy in order for the following MIPS fragment to branch to L1 is:

_____

```
ori $t1, $0, 0x111
and $t0, $t0, $t1
beq $t0, $t1, L1
```

(19) The content of register **$t0** after executing the following code is _____:

```
        li $s1, 0x4321
        xor $t0, $t0, $t0
Next:
        andi $t1, $s1, 0xf
        add $t0, $t0, $t1
        srl  $s1, $s1, 4
        bne $s1, $0, Next
```

[25 Points]

(Q2) Write separate MIPS assembly code fragments with **minimum** instructions to implement each of the given requirements.

(i) [6 points] Given two arrays of words A and B with their base addresses stored in registers $s0 and $s1, array size N is stored in $s2, and index i is stored $s3, write the smallest MIPS assembly fragment for the following computation:

for (i=0; i<n; i++) if ((A[i]-B[i])*5 >=0) then A[i]= (A[i]-B[i])*5;

(ii) [6 points] Given the following MIPS assembly fragment:

```
        bne $s1, $s2, exit
        bge $s2, $s3, exit
        addi $s4, $s4, 5
Exit:
```

Assume that variables a, b, c, and d are stored in registers $s1, $s2, $s3, and $s4, respectively.

Fill in the Boolean expression in the following IF statement:

If (_____) then d=d +5;

Repeat the above question for the following MIPS assembly fragment:

```
        beq $s1, $s2, process
        bgt $s2, $s3, exit
        ble $s3, $s4, exit
process: add $s4, $s4, $s1
Exit:
```

Fill in the Boolean expression in the following IF statement:

If (_____) then d=d +a;

(iii) [3 points] Write a MIPS assembly fragment for the following IF statement:

if ([(a == b) || (c== d)] && (a < c)) then b = d ;

Assume that variables a, b, c, and d are stored into registers $s0, $s1, $s2, and $s3, respectively.

(iv) [5 points] Write a MIPS assembly fragment to count the number of occurrence of alphabetic characters (can be lowercase or uppercase) in a null terminated string, where the base address of the string is in register $s0 and the count is to be in $s1**.**

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
2	space	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	`	a	b	c	d	e	f	G	h	i	j	k	l	m	n	o
7	p	q	r	s	t	u	v	W	x	y	z	{	|	}	~	DEL

(v) [5 points] Write the most optimized MIPS assembly fragment for the following WHILE statement:

i = 0;
WHILE ((A[i] >= B[i]*2) && (i<N)) { A[i] = A[i]- B[i]; i = i+1; }

Where A and B are arrays of Bytes. The base addresses of arrays A and B are stored into registers $s0 and $s1, respectively. The index i and count N are stored into registers $s2 and $s3.

MIPS Instructions:

Instruction	Meaning	R-Type Format					
add $s1, $s2, $s3	$s1 = $s2 + $s3	op = 0	rs = $s2	rt = $s3	rd = $s1	sa = 0	f = 0x20
addu $s1, $s2, $s3	$s1 = $s2 + $s3	op = 0	rs = $s2	rt = $s3	rd = $s1	sa = 0	f = 0x21
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 & $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 = 0x25
xor $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 = 0x27

Instruction	Meaning	R-Type Format					
sll $s1,$s2,10	$s1 = $s2 << 10	op = 0	rs = 0	rt = $s2	rd = $s1	sa = 10	f = 0
srl $s1,$s2,10	$s1 = $s2>>>10	op = 0	rs = 0	rt = $s2	rd = $s1	sa = 10	f = 2
sra $s1, $s2, 10	$s1 = $s2 >> 10	op = 0	rs = 0	rt = $s2	rd = $s1	sa = 10	f = 3
sllv $s1,$s2,$s3	$s1 = $s2 << $s3	op = 0	rs = $s3	rt = $s2	rd = $s1	sa = 0	f = 4
srlv $s1,$s2,$s3	$s1 = $s2>>>$s3	op = 0	rs = $s3	rt = $s2	rd = $s1	sa = 0	f = 6
srav $s1,$s2,$s3	$s1 = $s2 >> $s3	op = 0	rs = $s3	rt = $s2	rd = $s1	sa = 0	f = 7

Instruction	Meaning	I-Type Format			
addi $s1, $s2, 10	$s1 = $s2 + 10	op = 0x8	rs = $s2	rt = $s1	imm^{16} = 10
addiu $s1, $s2, 10	$s1 = $s2 + 10	op = 0x9	rs = $s2	rt = $s1	imm^{16} = 10
andi $s1, $s2, 10	$s1 = $s2 & 10	op = 0xc	rs = $s2	rt = $s1	imm^{16} = 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	op = 0xf	0	rt = $s1	imm^{16} = 10

Instruction	Meaning	Format			
j label	jump to label	op^6 = 2	imm^{26}		
beq rs, rt, label	branch if (rs == rt)	op^6 = 4	rs^5	rt^5	imm^{16}
bne rs, rt, label	branch if (rs != rt)	op^6 = 5	rs^5	rt^5	imm^{16}
blez rs, label	branch if (rs<=0)	op^6 = 6	rs^5	0	imm^{16}
bgtz rs, label	branch if (rs > 0)	op^6 = 7	rs^5	0	imm^{16}
bltz rs, label	branch if (rs < 0)	op^6 = 1	rs^5	0	imm^{16}
bgez rs, label	branch if (rs>=0)	op^6 = 1	rs^5	1	imm^{16}

Instruction	Meaning	Format					
slt 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}		
sltiu rt, rs, imm^{16}	rt=(rs<imm?1:0)	0xb	rs^5	rt^5	imm^{16}		

Instruction	Meaning	I-Type Format			
lb rt, imm^{16}(rs)	rt = MEM[rs+imm^{16}]	0x20	rs^5	rt^5	imm^{16}
lh rt, imm^{16}(rs)	rt = MEM[rs+imm^{16}]	0x21	rs^5	rt^5	imm^{16}
lw rt, imm^{16}(rs)	rt = MEM[rs+imm^{16}]	0x23	rs^5	rt^5	imm^{16}
lbu rt, imm^{16}(rs)	rt = MEM[rs+imm^{16}]	0x24	rs^5	rt^5	imm^{16}
lhu rt, imm^{16}(rs)	rt = MEM[rs+imm^{16}]	0x25	rs^5	rt^5	imm^{16}
sb rt, imm^{16}(rs)	MEM[rs+imm^{16}] = rt	0x28	rs^5	rt^5	imm^{16}
sh rt, imm^{16}(rs)	MEM[rs+imm^{16}] = rt	0x29	rs^5	rt^5	imm^{16}
sw rt, imm^{16}(rs)	MEM[rs+imm^{16}] = rt	0x2b	rs^5	rt^5	imm^{16}