*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 161 (Fall 2016-2017)**
**Major Exam 2**
**Saturday Dec. 10, 2016**

**Time: 150 minutes, Total Pages: 13**

**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 | 20 | |
| Q2 | 10 | |
| Q3 | 17 | |
| Q4 | 23 | |
| Total | 70 | |

Dr. Aiman El-Maleh
Dr. Marwan Abu Amara

**[20 Points]**

**(Q1)** Write MIPS programs with minimal used instructions. Use MIPS programming convention in saving and restoring registers in procedures.

**(i)** **[4 points]** Write a procedure **GetAscii** that receives a single hexadecimal digit in register $a0 and returns the ASCII code of that digit in register $v0. For example, if $a0=0x9 the procedure will return 0x39 in $v0 and if $a0=0xA, the procedure will return 0x41 in $a0. Assume the use of capital letters for the digits A to F.

**(ii)** **[11 points]** Write a procedure **DispHex** that receives a number in register $a0 and displays the hexadecimal representation of that number. Only significant hexadecimal digits need to be displayed. For example, if $a0=0x1E, the procedure will display 1E. Your DisHex procedure should utilize the GetAscii procedure.

**(iii)** **[5 points]** Write a MIPS program that asks the user to enter a decimal number and displays its hexadecimal content using the **DispHex** procedure. Two sample runs of the program are given below:

Enter a decimal number: 260
Your number in hexadecimal is: 0x104

Enter a decimal number: 0
Your number in hexadecimal is: 0x0

**(Q2)**

**(i)** **[4 Points]** Given that **Multiplicand=0111** and **Multiplier=1011** are signed 2's complement numbers, show the **signed** multiplication of **Multiplicand** by **Multiplier**. The result of the multiplication should be an 8 bit **signed** number in HI and LO registers. Show the steps of your work.

| **Iteration** | | **Multiplicand** | **Sign** | **Product =HI,LO** |
|---|---|---|---|---|
| 0 | Initialize | | | |
| 1 | | | | |
| | | | | |
| 2 | | | | |
| | | | | |
| 3 | | | | |
| | | | | |
| 4 | | | | |
| | | | | |

**(ii)** **[6 Points]** Given that **Dividend=0111** and **Divisor=1011** are signed 2's complement numbers, show the **signed** division of **Dividend** by **Divisor**. The result of division should be stored in the Remainder and Quotient registers. Show the steps of your work, and show the final result.

| **Iteration** | | **Remainder (HI)** | **Quotient (LO)** | **Divisor** | **Difference** |
|---|---|---|---|---|---|
| 0 | Initialize | | | | |
| 1 | | | | | |
| | | | | | |
| 2 | | | | | |
| | | | | | |
| 3 | | | | | |
| | | | | | |
| 4 | | | | | |
| | | | | | |
| Final Result | | | | | |

**(Q3)**

1. **[2 Points]** Find the **decimal value** of the following single precision float:

   [0, 1000 1000, 0000 0100 1100 0000 0000 000]

2. **[2 Points]** Find the **decimal value** of the following single precision float:

   [1, 0000 0000, 0110 0000 0000 0000 0000 000]

3. **[3 Points]** Find the normalized single precision representation of –59.625.

4. **[4 Points]** Round the given single precision float with the given GRS bits using the following rounding modes showing the resulting normalized number:

   **GRS**

   **+1.111 1111 1111 1111 1111 1111 100 x** $2^{-127}$

   Zero: [ ]

   +infinity: [ ]

   -infinity: [ ]

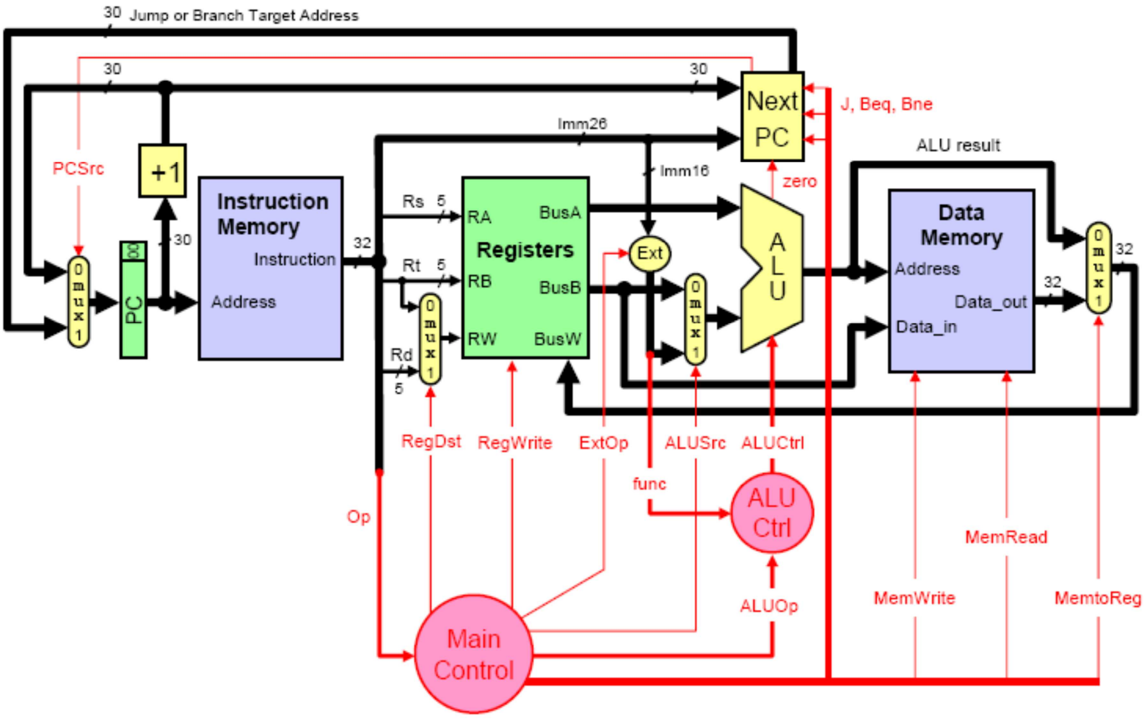
   Nearest Even: [ ]

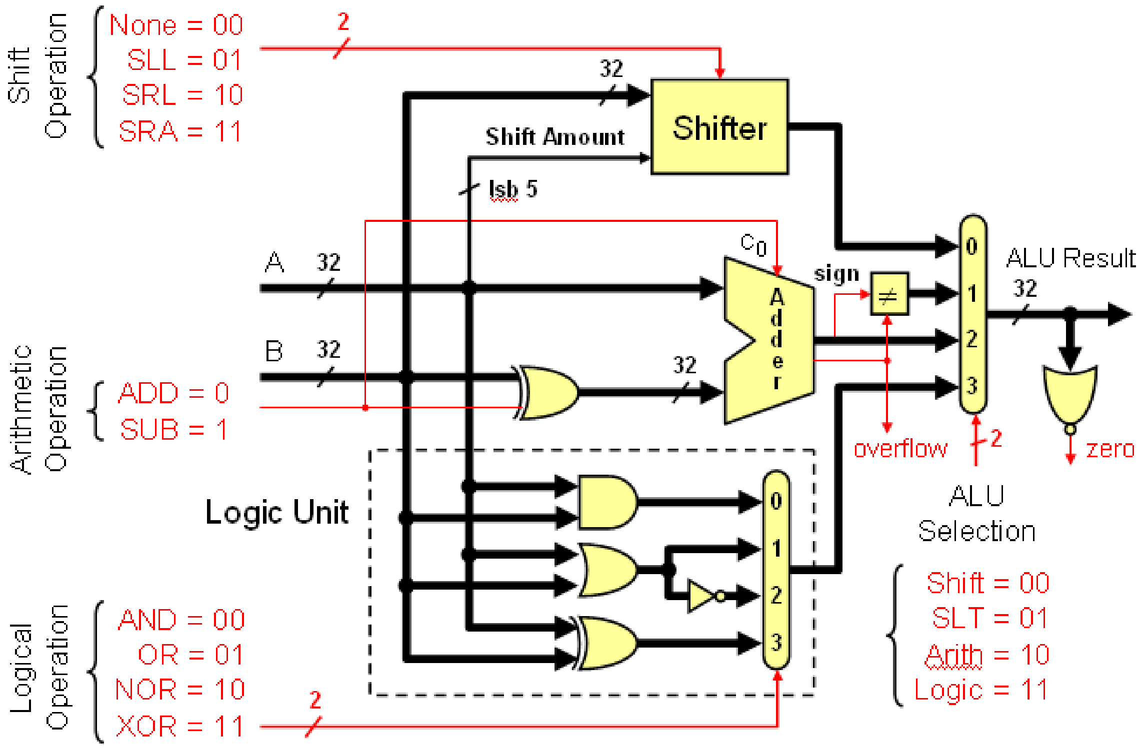
5. **[6 Points]** Find the normalized **<u>difference</u>** between **A** and **B** (i.e., A-B) by using rounding to **<u>+infinity</u>**. Perform the operation using **guard**, **round** and **sticky** bits.

**A = +1.000 0101 1100 1010 1000 0001 × $2^4$**
**B = +1.011 1001 0101 0000 0010 1000 × $2^{-1}$**

**[23 Points]**

**(Q4)** Consider the single-cycle datapath and control given below along with ALU design for the MIPS processor implementing a subset of the instruction set:



**(i)** Show the control signals generated for the execution of the following instructions by filling the table given below: **(5 points)**

| Op | RegDst | RegWrite | ExtOp | ALUSrc | ALUOp | Beq | Bne | J | MemRead | MemWrite | MemtoReg |
|---|---|---|---|---|---|---|---|---|---|---|---|
| R-type | | | | | | | | | | | |
| slti | | | | | | | | | | | |
| sw | | | | | | | | | | | |
| beq | | | | | | | | | | | |
| j | | | | | | | | | | | |

**(ii)** Excluding the ALUOp, Beq, Bne and J signals, show the design of the control unit for the control signals given in the table above based on the given instructions. Assume that the opcode of these instructions is a 6-bit opcode such that the opcode for R-type instructions is 0, the opcode for slti is 1, the opcode for sw is 2, and so on for the rest of the instructions. **(5 points)**

**(iii)** Show the design of the Next PC block. **(4 points)**

**(iv)** We wish to add the following instructions to the MIPS single-cycle datapath. Add any necessary datapath modifications and control signals needed for the implementation of these instructions. Show only the **<u>modified</u>** and **<u>added</u>** components to the datapath.

**a.** sra **(3 points)**

| Instruction | Meaning | Format | | | | | |
|---|---|---|---|---|---|---|---|
| sra rd, rt, $imm^5$ | rd= rt>>$imm^{16}$ | $Op^6 = 0$ | 0 | $rt^5$ | $rd^5$ | $Imm^5$ | $f^5$=3 |

**b.** jr **(3 points)**

| Instruction | Meaning | Format | | | | | |
|---|---|---|---|---|---|---|---|
| jr rs | PC=rs | $op^6 = 0$ | $rs^5$ | 0 | 0 | 0 | 8 |

**(v)** Assume that the propagation delays for the major components used in the datapath are as follows:

- Instruction and data memories: 120 ps
- ALU and adders: 30 ps
- Register file access (read or write): 14 ps
- Main control: 8 ps
- ALU control: 7 ps

Ignore the delays in the multiplexers, PC access, extension logic, and wires. What is the cycle time for the single-cycle datapath given above? **(3 points)**

# Syscall Services:

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

# 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 | $rs^5$ | $rt^5$ | $imm^{16}$ |

| Instruction | Meaning | Format | | | | | |
|---|---|---|---|---|---|---|---|
| jal label | \$31=PC+4, jump | $op^6 = 3$ | $imm^{26}$ | | | | |
| jr Rs | PC = Rs | $op^6 = 0$ | $rs^5$ | 0 | 0 | 0 | 8 |
| jalr Rd, Rs | Rd=PC+4, PC=Rs | $op^6 = 0$ | $rs^5$ | 0 | $rd^5$ | 0 | 9 |

| Instruction | Meaning | Format | | | | | |
|---|---|---|---|---|---|---|---|
| mult Rs, Rt | Hi, Lo = Rs × Rt | $op^6 = 0$ | $Rs^5$ | $Rt^5$ | 0 | 0 | 0x18 |
| multu Rs, Rt | Hi, Lo = Rs × Rt | $op^6 = 0$ | $Rs^5$ | $Rt^5$ | 0 | 0 | 0x19 |
| mul Rd, Rs, Rt | Rd = Rs × Rt | 0x1c | $Rs^5$ | $Rt^5$ | $Rd^5$ | 0 | 0x02 |
| div Rs, Rt | Hi, Lo = Rs / Rt | $op^6 = 0$ | $Rs^5$ | $Rt^5$ | 0 | 0 | 0x1a |
| divu Rs, Rt | Hi, Lo = Rs / Rt | $op^6 = 0$ | $Rs^5$ | $Rt^5$ | 0 | 0 | 0x1b |
| mfhi Rd | Rd = Hi | $op^6 = 0$ | 0 | 0 | $Rd^5$ | 0 | 0x10 |
| mflo Rd | Rd = Lo | $op^6 = 0$ | 0 | 0 | $Rd^5$ | 0 | 0x12 |