**Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ID: \_\_\_\_\_\_\_\_\_\_\_\_**

**Department of Computer Science and Engineering**

**MIDTERM EXAMINATION**

**Fall 2013**

**CSE340: Computer Architecture**

**Total Marks:** **40**  **Time Allowed: 1 hour**

* Answer **ANY FOUR** questions.
* Return the question with your answer script

Question 1

1. What do you mean by High level language, Compiler and Assembler? **4.5**
2. Define ISA and ABI. **2**
3. There are different types of instructions available in MIPS. Draw various instruction formats and also mention their types. **3.5**

Question 2

1. Encode the following MIPS instructions. For each instruction, you should identify the format type (R, I, or J format): **4.5**
   1. sll $t2, $s0, 4 #$t2 is register 10 and $s0 is register 16, op=0,func=0
   2. slt $t2, $s1,100 # $s1=17,op=42
   3. add $t2, $s0, $s1 # $ op=0,func=32

1. The following problems deal with translating from C to MIPS. Assume that the variables f and g are assigned to registers $s0 and $s1, respectively. Assume that the base address of the arrays A and B are registers $s6 and $s7, respectively.
   1. **f=g-A[B[4]]**

For the C statements above, what is the corresponding MIPS assembly code? **3.5**

1. What type instruction is **JAL**? How would you represent **JAL 201** in the register? What would be the value in **$ra**/**$31? 2**

Question 3

1. Explain various addressing modes in MIPS. **4**
2. Multiply (-5x3) using Booth’s algorithm. **3**
3. What do you mean by overflow, underflow and double precision for floating point values? **3**

Question 4

1. Convert **-71.6875** into **IEEE-754** single point floating point representation. Also show the hex equivalent of the representation. **5**
2. 1 00000111 11000000000000000000000 convert this to decimal value using **IEEE754 32-bit** floating point representation. **5**

Question 5

1. **X**=0100 0110 1100 0000 0000 0000 0000 00002 **Y**=1011 1110 1010 0000 0000 0000 0000 00002 representing single precision **IEEE 754** floating-point numbers, Find **X+Y**. **5**
2. Multiply 0.510 and -0.437510 using floating point multiplication method. **5**