# Computer Organization (0306-550) - Winter 2012
# Homework Assignment #1 - Due December 13

1. What factors influence the choice of computing elements in a system?

2. What is the classification of computer/processor generations based on? Specify one common characteristic from the first generation to the current fourth generation.

3. What are the similarities and differences between General Purpose Processors (GPPs) and Embedded Processors?

4. Differentiate between independent RTN and dependant RTN.

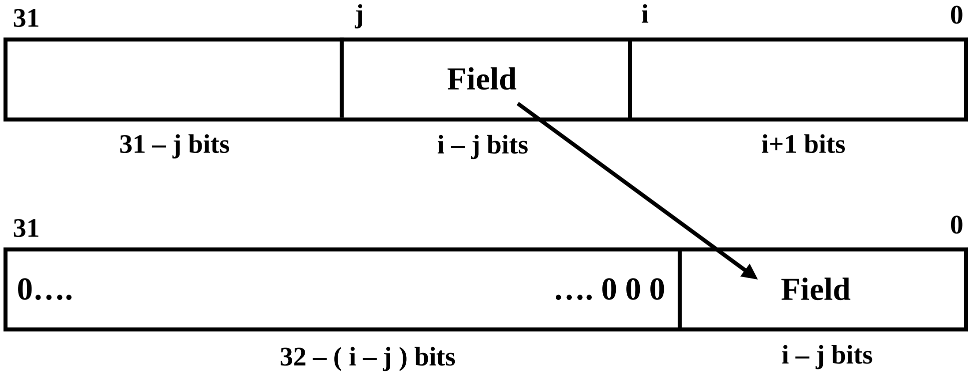
5. First, give a concise definition of Instruction Set Architecture (ISA) including its main purpose and then list five issues that must be fully specified by an ISA.

6. What types of information must be specified either explicitly or implicitly by every machine (CPU) instruction?

7. Differentiate between load-store GPR ISAs and register-memory GPR ISAs. What are the advantages of load-store GPR ISAs over register-memory GPR ISAs? What are the disadvantages, if any?

8. Contrast between CISC and RISC ISAs in terms motivation/emphasis and ISA characteristics.

9. Some processor ISAs have explicit instructions to extract an arbitrary field from a 32-bit register and to place it in the least significant bits of another register. The figure below shows the desired operation:



Find the shortest sequence of MIPS instructions that extracts a field for the constant values i = 5 and j = 22 from register $t3 and places it in register $t0.
(Hint: It can be done in two instructions)

**10.** **The following MIPS code fragment processes two arrays and produces an important value in register $v0. Assume that each array consists of 2500 words indexed 0 through 2499, that the base addresses of the arrays are stored in $a0 and $a1 respectively, and their sizes (2500) are stored in $a2 and $a3, respectively. Add comments to the code and describe in one sentence what this code does. Specifically, what will be returned in $v0?**

```
                  sll    $a2,  $a2, 2
                  sll    $a3,  $a3, 2
                  add    $v0,  $zero, $zero
                  add    $t0,  $zero, $zero
       outer:     add    $t4,  $a0, $t0
                  lw     $t4,  0($t4)
                  add    $t1,  $zero, $zero
       inner:     add    $t3,  $a1, $t1
                  lw     $t3,  0($t3)
                  bne    $t3,  $t4, skip
                  addi   $v0,  $v0, 1
       skip:      addi   $t1,  $t1, 4
                  bne    $t1,  $a3, inner
                  addi   $t0,  $t0, 4
                  bne    $t0,  $a2, outer
```

**11.** **The following MIPS program tries to copy words from the address in register $a0 to the address in register $a1, counting the number of words copied in register $v0. The program stops copying when it finds a word equal to 0. The last terminating word should be copied but not counted.**

```
           addi   $v0,  $zero, 0  # Initialize count
 loop:     lw     $v1,  0($a0)    # Read next word from source
           sw     $v1,  0($a1)    # Write to destination
           addi   $a0,  $a0, 4    # Advance pointer to next source
           addi   $a1,  $a1, 4    # Advance pointer to next destination
           beq    $v1,  $zero, loop   # Loop if word copied != zero
```

**There are multiple bugs in this MIPS program; identify the bugs, fix them, and turn in a bug-free version.**

**12.** **Given your understanding of MIPS PC-relative addressing, explain why an assembler might have problems directly implementing the branch instruction in the following code sequence:**

```
    here:     bne    $s0, $s2, there

        .  .    .

    there:    add    $s0, $s0, $s0
```