# *The* UNIVERSITY *of* NORTH CAROLINA *at* CHAPEL HILL

**Comp 411 Computer Organization**
Spring 2010

**Problem Set #3**
*Issued Wednesday 3 Feb 2010; Due before class on Monday 18 Feb 2010*

**Homework Information**: Some of the problems are probably too long to be done the night before the due date, so plan accordingly. Late homework will not be accepted. Feel free to get help from others, but the work you hand in should be your own.

**Problem 1. "Name that Loop" (20 points)**

(A) Describe in your own words the function performed by the following code fragment when it reaches the label `end`. Consider the following hints. The code fragment operates on the contents of register a0, and the loop will always complete.

```
         addi   $t0,$0,32
loop:    andi  $t1,$a0,1
         sub   $t0,$t0,$t1
         srl   $a0,$a0,1
         bne   $a0,$0,loop
end:     add   $a0,$t0,$0
```

(B) Describe how the operation of the above code fragment changes if the `srl` instruction is replaced with an `sra` instruction with the same arguments.

(C) Describe the function of the following code fragment in your own words:

```
         la    $t1,a
loop:    lw    $t0,4($t1)
         sw    $t0,($t1)
         addi  $t1,$t1,4
         bne   $t0,$0,loop
```

(D) Describe the function of the following code fragment in your own words:

```
         la    $t1,a
loop:    lw    $t0,($t1)
         addi  $t1,$t1,4
         bne   $t0,$0,loop
         la    $t0,a
         sub   $t0,$t1,$t0
         sra   $t0,$t0,2
```

**Problem 2. "A Loop of Your Own" (80 points)**

Download and install MARS, the MIPS instruction set simulator (see class website for download link). Then, write an assembly program to do the following.

You are given an array called Pixels, each element of which is a 32-bit word representing a color value. The lowest significant 8 bits of each color value denote an unsigned integer (from 0 to 255) representing the color's "blue" value, the next 8 bits are the "green" value, the next 8 bits are the "red" value, and the most significant 8 bits are all zeros. The size of the array is not known, but it is known that the last element of the array is a negative number (e.g., "-1"); this negative value should itself not be considered as a valid color value, of course. For example, the following array has three color values:

```
.data
Pixels:  .word   0x010101, 0x5, 0x1111, -1
```

Your program should read the Pixels array, one element at a time, until the end is reached, and for each element read, it should print the following line to the console:

Pixel *n* = *r g b*

where *n* is the number of the array element read (first one is 0th), and *r, g* and *b* are integers (from 0 to 255) representing the red, green and blue values for that pixel. You need not worry too much about formatting and white space etc., but print only one element per line.

So, if you used the Pixels array given above, your output should be:

```
Pixel 0 = 1 1 1
Pixel 1 = 0 0 5
Pixel 2 = 0 17 17
```

Your code fragment will begin execution at the label "main". You will use the syscall instruction (Appendix B. 43-45) to print the above output, and also to terminate your program.

***Submission:*** A starter assembly file ps3.asm is provided for you on the class website. Download the file and edit it to include your name and onyen in the comments at the top and edit your code in at the place indicated.

Submit your code on blackboard as usual.

***Grading:*** Your program will be graded as correct (full-credit), partially correct (half-credit), or incorrect (no credit). Get help early and often.

***Collaboration:*** You are free to take as much help from other students as you need to write and debug your code. You must, however, understand everything you submit, and be able to explain if asked.