CS/COE 0447 Fall 2009
Lab 4: Functions
Due Date: October 9, 2009

To get started on this lab, attend recitation on 9/25. Each of you should submit your own solution, according to these instructions: http://www.cs.pitt.edu/~sab104/teaching/cs447/submission.html. You may collaborate with your partner, but each person must turn in their own copy of the lab, with the name of their partner. The lab is due on 10/9 at 11:59pm.

In this lab, we will write four functions that manipulate the memory locations of the LED display to turn on and off some LEDs.

1) Write a function *void setLED(int *address*, *int bitPattern)* that stores the word *bitPattern* in the memory location pointed to by *address*. In the previous definition, an *int* is the size of a word and *int ** is a pointer to a word (address of a word). Use the code below to call your function:

```
        .text
        li $a0, 0xFFFF0000  #LED memory starts at this address
        li $a1, 0x55555555  #LEDs to turn on
        jal setLED          #Jump and link to setLED
        li $v0, 10          #Exit
        syscall
```

2) Write a function *int getLED(int *address)* that returns the bit pattern currently stored in the memory location pointed to by *address*. Use the code below to call your function:

```
        .data
ok:     .asciiz   "The values match!"
not_ok: .asciiz   "The values don't match!"

        .text
        li $a0, 0xFFFF0000  #LED memory starts at this address
        li $a1, 0x55555555  #LEDs to turn on
        jal setLED          #Jump and link to setLED
        jal getLED          #Jump and link to getLED
        bne $a1, $v0, else  #Return values should be in $v0
        la $a0, ok          #Load ok string if equal
        j end
else:   la $a0, not_ok      #Load not_ok string if not equal
```

```
end:    li $v0, 4           #Print the string
        syscall
        li $v0, 10          #Exit
        syscall
```

3) Write a function *void notLED(int *address)* that reads the bit pattern stored in the memory location pointed to by *address*, takes its complement and stores it back to the same location in memory. Your function must use the functions defined in the previous two points. Use the code below to call your function:

```
.text
li $a0, 0xFFFF0000  #LED memory starts at this address
li $a1, 0x55555555  #LEDs to turn on
jal setLED          #Jump and link to setLED
jal notLED          #Jump and link to notLED
li $v0, 10          #Exit
syscall
```

4) Write a function void *setLEDRange(int *address, int bitPattern, int num)* that stores the word *bitPattern* in *num* consecutive memory locations starting at the address pointed to by *address*. Your function must use the functions defined in points 1 and 2. Use the code below to call your function:

```
.text
li $a0, 0xFFFF0000  #LED memory starts at this address
li $a1, 0x55555555  #LEDs to turn on
li $a2, 5           #Number of words to store
jal setLEDRange     #Jump and link to setLEDRange
li $v0, 10          #Exit
syscall
```