

## CS 245 Fall 2017

### Assignment 1A – Ands and Ors

### Assignment 1B – Working with Sprites

For all assignments, be sure to:

- Include file comments at the top, listing your name, the name of the program, and briefly describe what the program does.
- Line up instruction parts in columns: 0=labels; 1tab=instruction mnemonic; 2tabs=operands; 3-4tabs=comments.
- Include pseudo-code comments to the right of your assembly code. **Avoid** comments that tell me what the assembly does, such as: move 25 to register \$s3 – I already know this. Keep your comments to the logic.

#### *Program 1A: AND and OR*

For the first program, simply perform the following calculation:

\$T1 = 0x3f AND 0xaa OR 0x03

Print your result: “0x3f & 0xaa | 0x03 = 0x” <N>

Hint: Modify lab1.s to start. You will print your result as a string and an integer. See notes on Assembly logic for print details.

To print a hexadecimal number, set \$v0 to 34 and \$a0 to the integer to print. For more information of what syscall can do, see

<http://courses.missouristate.edu/KenVollmar/MARS/Help/SyscallHelp.html>.

Turn this program in as hwk1.asm via paper and electronic copy:

\$ submit 245 hwk1.asm

#### *Program 1B: Moving the sprite back and forth in memory*

Write a MIPS assembly program that when run in MARS at 25 instructions/second will display in the **Memory .data segment** a “sprite” that moves across one word.

Assuming that you chose a “background” of the hex digit e and a “sprite” of the hex digit 0, a “time lapse” series of snapshots of the Memory segment display would look like the data below:

Time 1: 0x0e000000  
Time 2: 0xe0000000  
Time 3: 0xee000000  
Time 4: 0xeee00000  
Time 5: 0xeeee0000  
Time 6: 0xeeeeee00  
Time 7: 0xeeeeeee0  
Time 8: 0xeeeeeeee0  
Time 8: 0xeeeeeee0e  
Time 9: 0xeeeeee0ee  
Time 10: 0xeeee0eee  
Time 11: 0xee000000  
Time 12: 0x0e000000

0 is the  
sprite  
character

...  
Time 15:0x0e000000

You may choose the hex digits used for the “background” and the “sprite” – select two hex digits that have as much contrast as possible for best visibility. For instance, the digits **e** and **c** are poor choices because they look so similar.

The learning goals of this assignment is the use of bit manipulation and loops. Please use a loop (or two) as part of this assignment.

Hint: There is an assembly language shift/rotate instruction that makes this program a lot easier. Look for it! First get the sprite to move over the first word. Once you have that working, consider how you can get the sprite to move over the second word.

***Important: set the MARS slider bar to execute at a slower speed. Omitting this step will run at full speed and no output will be apparent.***

Turn this program in as hwk1a.asm and hwk1b.asm via paper and electronic copy:

\$ submit 245 hwk1a.asm

\$ submit 245 hwk1b.asm

### *Grading*

Each homework assignment is worth 10 points. Be careful to include all comments and format correctly, as directed in the beginning of this file.