Name: (as it would appear on official course roster)
UCSB email address: **@ucsb.edu**
Lab Section Time:
Optional: name you wish to be called if different from above
Optional: name of "homework buddy" (leaving this blank signifies "I worked alone")

Assignment 01: MIPS Assembly & Logic Refresher

Assigned: *Friday, January 10th, 2020*
Due: *Wednesday, January 15th, 2020*
Points: *60 (normalized to 100 in gradebook)*

- You may collaborate on this homework with AT MOST one person, an optional "homework buddy".
- MAY ONLY BE TURNED IN ON **GRADESCOPE as a PDF file** (see instructions in online lab01 description).
- There is NO MAKEUP for missed assignments.
- We are strict about enforcing the LATE POLICY for all assignments (see syllabus).

Only use the space provided for answers. Use clear and clean handwriting (or typing).

1. (1 pt) Go to http://spimsimulator.sourceforge.net and re-familiarize yourself with SPIM (as used in CS 64). If you do not already have it, download SPIM on your personal computer (this is completely optional). PLEASE NOTE – you do NOT need to download it on the CSIL computers as it is already installed on there. Also, go to our main class website and under "Documentation", find the file called "spim.pdf" and use it as a reference. If you have notes from CS 64, this may be a good time to put them aside, should you need them.

2. (9 pts) Log into your CSIL account, write/copy this assembly program below into a text file, save it as **simple.asm**, and execute it on SPIM. Then answer the questions below:

```
.text
main:
      li $a0, 1
      li $a1, 5              #
      addi $a0, $a0, 5       #
      add $a1, $a1, $a0      #
      li $v0, 1
      move $a0, $a1
      syscall                #
      li $v0, 10
      syscall                #
```

   a. (2 pts) What is the Linux command you typed to execute this program?

Name:

(as it would appear on official course roster)

b. (2 pts) What did you see on your terminal window (exact replica please)?

c. (5 pts) There are five hashtags in the program above (comments in MIPS assembly). What comments would you write after each of them? Show each comment here and be brief, that is, only describe what the relevant lines are doing.

Comment 1:_____

Comment 2:_____

Comment 3:_____

Comment 4:_____

Comment 5:_____

3. (20 pts) Consider the following MIPS assembly program fragment which swaps values of two locations in memory. The program assumes two parameters **v** and **k** which are in registers **$a0** and **$a1**, respectively. Note that the line numbers shown here are NOT part of the program, but for your reference only.

```
1: swap:
2:    sll $t1, $a1, 2
3:    add $t1, $a0, $t1
4:    lw $t0, 0($t1)
5:    lw $t2, 4($t1)
6:    sw $t2, 0($t1)
7:    sw $t0, 4($t1)
```

Name:

(as it would appear on official course roster)

a. (6 pts) Write commentary to lines 2 thru 7 of the MIPS assembly program to describe what is happening:

Comment line 2:_____

Comment line 3:_____

Comment line 4:_____

Comment line 5:_____

Comment line 6:_____

Comment line 7:_____

b. (2 pts) What additional instructions to the beginning of the program will place the integer array **v = [13, 17, 19, 23, 29]** in MIPS memory?

c. (2 pts) What instruction would place the starting address of **v** (i.e. **&v[0]**) into the register **$a0**?

d. (2 pts) What instruction would place the value of **k = 3** into the register **$a1**?

e. (2 pts) What are the final values inside registers **$a0** and **$a1** when this is executed?

f. (2 pts) Write the lines of code instructions in MIPS assembly that would print out the values inside registers **$a0** and **$a1**.

Name:

(as it would appear on official course roster)

g. (4 pts) Write/type-up the **entire** program up with the additions made thus far, **add** program exit instructions. Execute the program to make sure it runs without errors. Write/print/paste the entire program here **AND** tell me what did you see on your terminal window when you executed this program (exact replica please)?

Name:

(as it would appear on official course roster)

4. (20 pts) Refresher on digital logic design:
   a. (3 pts) What is a 1-bit 2:1 mux? Describe all its I/O. Draw a representative *symbol* (not the entire logic, just the symbol) illustrating this.

   b. (3 pts) What is a 3-bit 5:1 mux? Describe all its I/O. Draw a representative *symbol* illustrating this.

   c. (2 pts) Is there such a thing as a 2-bit 2:2 mux? Why or why not?

   d. (8 pts) Consider a simplified 1-bit ALU that has 2 inputs: A and B, and 2 outputs: R and C_{OUT}, that can compute one of 3 functions:
   R = A & B, **R = A XNOR B**, and **R, C_{OUT} = A + B** (addition with carry-out)
      i. (3 pts) Draw a *symbol* describing this, showing all its I/O.

Name:

(as it would appear on official course roster)

ii. (5 pts) Design and draw the digital logic inside this simplified ALU. You may use any *combinatorial* digital logic to do this (example: AND/OR/NOT/NOR gates, Muxes)

e. (2 pts) What type of *sequential* digital logic would a *CPU register* be designed with? Why?

f. (2 pts) What type of *sequential* digital logic would a *finite state machine (FSM*) that might be found inside the control unit of a CPU be designed with? Why?

Name:

(as it would appear on official course roster)

5. (10 pts) Assume a 15 cm diameter wafer has a cost of $12, contains 84 dies, and has 0.020 defects/cm^2. Assume a 20 cm diameter wafer that has a cost of $15, contains 100 dies, and has 0.031 defects/cm^2.
    a. (5 pts) Find the yield for both wafers. Show your calculations.

    b. (5 pts) Find the cost per die for both wafers. Show your calculations.