CS 2130 Exam 1

Name ______________________________

You MUST clearly write your computing ID on **EACH** page and put your name on the top of this page, too. Please write legibly.

If you are still writing when "pens down" is called, your exam will not be graded – even if you are still signing the honor pledge. So please do that first. Sorry to have to be strict on this!

There are 6 pages to this exam. Once the exam starts, please make sure you have all the pages. Questions are worth different amounts of points, so be sure to look over all the questions and plan your time accordingly.

This exam is CLOSED text book, closed-notes, closed-cell phone, closed-smart watch, closed-computer, closed-neighbor, etc. You may **not** discuss this exam with anyone until after the grades have been released. Please sign the honor pledge below.

On my honor as a student, I have neither given nor received aid on this exam. I will not discuss the content of this exam, even in vague terms, with *anyone* other than current course staff, until *after* grades have been released.

____

Computers are good at following instructions, but not at reading your mind.
–Donald Knuth

Page 2: Scratch Paper

Our Example Toy ISA. This is the ISA described in class and used in Labs 4-5 and Homework 3. Each instruction is one (or two) bytes, defined as:

7	6	5	4	3	2	1	0
`R`	`icode`			`a`		`b`	

byte at `pc`

7	6	5	4	3	2	1	0
immediate							

byte at `pc + 1`

If the reserved bit (bit 7) in our instruction is 0, the following table defines our instruction encoding.

icode	b	behavior
0		`rA = rB`
1		`rA += rB`
2		`rA &= rB`
3		`rA` = read from memory at address `rB`
4		write `rA` to memory at address `rB`
5	0	`rA = ~rA`
	1	`rA = -rA`
	2	`rA = !rA`
	3	`rA = pc`
6	0	`rA` = read from memory at `pc + 1`
	1	`rA` += read from memory at `pc + 1`
	2	`rA` &= read from memory at `pc + 1`
	3	`rA` = read from memory at the address stored at `pc + 1`
		For icode 6, increase `pc` by 2 at end of instruction
7		Compare `rA` as 8-bit two's-complement to `0`
		if `rA <= 0` set `pc = rB`
		else increment `pc` as normal

Nothing on this page will be graded.

Page 3: Programming our Computer

Suppose we extended the ISA simulator you wrote in Lab 4 with the following code:

```
if (reserved == 1 && icode == 2) {
    R[a] = R[a] ^ R[b];
    return oldPC + ___;
}
```

1. [4 points] What is the value that we will need to increment the `oldPC`? (Fill in the blank below.)

```
return oldPC + _____;
```

1

2. [10 points] Using the new instruction above **at least once**, write a program that computes the two's complement negative value of register 1 and stores the result in register 3 (that is, `r3 = -r1`) **without** using icode 5. Your program should only update register 3 and not change the values in registers 0, 1, or 2. Answer in hexadecimal bytes, separated by spaces. *Hint: how can we use xor to calculate bitwise not? You may need to write additional instructions.*

Answer: ______________________________

6C FF AD 6D 01

3. [12 points] Complete the table below listing all the register values as hex digits after the following code executes. Assume that all registers start with value `0x00` and that the first instruction is at address `0x00`.

```
6F 08 53 64 07 11 78 17 2D 00
```

Register	**Value**
0	
1	
2	
3	

0=09, 1=07, 2=00, 3=2D

Page 4: Binary

4. [6 points] Convert `219` into binary.

Answer

11011011

5. [6 points] What is `0b101100110111` in hexadecimal?

Answer

0xB37

6. [11 points] Answer the following questions assuming 8-bit two's-complement numbers.

   A. Compute the following sum, showing your work (such as carry bits, etc).

```
   0 1 1 0 1 0 1 0
 + 1 1 1 0 1 1 1 1
 -----------------
```

`0 1 0 1 1 0 0 1`

   B. Is your result a positive or negative number? (circle one)

Positive **Negative**

Positive

7. [4 points] Which of the following will result in integer overflow, assuming 8-bit two's complement signed integers? *Fill in the circle completely for all that apply.*

- ◯ `127 + 1`
- ◯ `64 + -72`
- ◯ `-25 + -42`

A

Page 5: Floating Point and Bitwise Operations

8. [9 points] Write the following binary number as an 8-bit floating point number assuming a 4-bit exponent value.

`+0.00001110011`

Answer

`0 0010 110`, partial credit for 1011 (-5 unbiased) exponent

9. [16 points] Assume x is an 8-bit two's complement integer. For each of the following pairs of expressions, if the two expressions are equivalent for all x, write "same"; otherwise, write an example x in binary for which the two are different.

A. `(x >> 1) & 0x80` and `x & 0x80`

Answer

same: shifting in the sign bit

B. `!((!x) << 3)` and `!((~x) << 3)`

Answer

different: x = 00111111

C. `~(x - 1)` and `~x + 1`

Answer

same: flip the bits and add one vs subtract one and flip the bits

D. `x ^ x` and `!x`

Answer

different: x = 00000000

Page 6: Circuits

10. [14 points] Using only 2-input `and`, `or`, `xor`, and `not` gates, draw a circuit that has 3 (1-bit) inputs labeled x, y, and s; 1 (1-bit) output labeled z; which computes the following. *Clearly label your inputs and outputs.*

```
z = s ? (x ^ y) : (x | y)
```

11. [4 points] The register that we discussed in class (the positive-edge-triggered D flip-flop) has inputs `D` and `clock` and output `Q`. If `Q` is 1 and the `clock` is transitioning from 1 to 0, which of the following is true? *Fill in the circle completely for all that apply.*

    ◯ `D` must also be 1 (not 0)

    ◯ `D` may be 0 or 1

    ◯ `Q` will transition from 1 to 0 with the clock

    ◯ `Q` will transition to the value of `D` when the clock transitions to 0

    B

12. [4 points] To build a 4-bit counter circuit, we could directly connect the outputs of the circuit back to the inputs without the need of a register.

    ◯ True

    ◯ False

    False