CMSC 311: Computer Organization

Fall 2002

Exam 1

Given: Monday, October 21, 2002

Instructions: This is a 75 minute exam. This exam is closed book and closed notes. No calculators are permitted on the exam. All computations should be done on paper.

This is a 140 point exam. The exam consists of 4 parts:

- Quiz 1: Data Representation (40 points) pass: 28
- Quiz 2: Combinational Logic (40 points) pass: 28
- Quiz 3: Low level C functions/CPU (20 points) pass: 14
- Challenge ("Drool Kitty") Section (40 points)

SEE PAGE 2 for USEFUL FACTS and CONVENTIONS.

Name	
SID (Last 6 digits)	
Email	

HONOR STATEMENT

The following is an honor statement. This is NOT the same as the "honor pledge". It only states your *awareness* of the following statement. It is the equivalent of saying "I know it is against the law to drive faster than the speed limit" instead of saying "I pledge not to drive faster than the speed limit".

> *I am aware that giving or receiving unauthorized assistance or using unauthorized aids on this exam is considered a violation of academic integrity.*

Please sign below to indicate that you READ and UNDERSTAND what the statement says. Your exam will not be returned to you, unless this is signed.

Signature	

Conventions:

- Circle data/control inputs from outside world. SQUARE outputs to outside world.
- Select (control) lines are written using c. For example, a 3 bit control lines are written $c_2c_1c_0$. c_2 is the most significant bit and c_0 is the least.
- Data input lines are written using x. For example, a 3 bit input lines are written $x_2x_1x_0$. x_2 is the most significant bit and x_0 is the least. (Assuming the input is a binary number).
- Data output lines are written using z. For example, a 3 bit output lines are written $z_2z_1z_0$. z_2 is the most significant bit and z_0 is the least. (Assuming the output is binary).
- Full adders will use A, B, C_i, C_o, and S inside the chip to represent the two bits being added, the carry in, the carry out, and the sum, respectively.
- Inside the chip, use capital letters. Outside the chip use lowercase letters to represent inputs coming from outside world, and outputs going to outside world.

Useful facts:

- $1000\ 0000_2 = 128$ and $1111\ 1111_2 = 255$
- The following are gates: AND, OR, NOT, XOR, XNOR, NAND, and NOR. Unless otherwise specified, you may assume that each gate (except NOT) can have infinite fan-in.
- The following are parts: any gates, multiplexer (MUX), demultiplexer (DEMUX), encoder, decoder, half adder (HA), full adder (FA), k-bit ripple carry adder, priority encoder.
  - HA uses inputs: A, B and outputs: C (carry out), S (sum).
  - FA uses inputs: A, B, C_i (carry in) and outputs: C_o (carry out), S (sum).

Grading

Part	Points
Quiz 1	
Quiz 2	
Quiz 3	
Challenge	
Total	

CMSC 311 Computer Organization
Fall 2002
RTfP

Name	
Student ID	

Quiz 1 (Data Representation): ".. and the bases are loaded" (40 points)

1.1 (2 pts) Here's a declaration in a C program:

```
int x ;
```

What's the internal data representation for `x`?

1.2 (2 pts) Here's a declaration in a C program:

```
float y ;
```

What's the internal data representation for `y`? (Hint: there's a `double` type that stores a longer version of a `float`)

1.3 (2 pts) Here's a declaration in a C program:

```
char ch ;
```

What's the internal data representation for `ch` on most computers (e.g., DC class cluster)?

1.4 (3 pts) Given a k-bit bitstring (where $k \geq 1$), how many different bit strings are there? Write a formula for this (hint: use k).

1.5 (3 pts) Suppose you are given a bitstring with *exactly* k 1's, and you're told the number is an unsigned integer. Write a formula for the base 10 (decimal) value of this bitstring. You can assume $k \geq 2$. (Don't use summations, unless you can't think of another way).

1.6 (3 pts) Suppose you are given a bitstring with *exactly* k 1's, and you're told the number is a signed integer. Write a formula for the base 10 (decimal) value of this bitstring. You can assume $k \geq 2$. State ANY ASSUMPTIONS needed for the formula to be true.

1.7 (4 pts) Write $+5_{10}$ in the following representations using 4 bits. If it is not possible to write it in the representation, write N/A.

Signed Magnitude	
1's Complement	
2's Complement	
Excess 8	
Excess -3	

1.8 (4 pts) Write -8_{10} in the following representations using 4 bits. If it is not possible to write it in the representation, write N/A.

Signed Magnitude	
1's Complement	
2's Complement	
Excess 8	
Excess -3	

1.9 (3 pts) Define overflow. That is, explain what overflow MEANS, not how to detect it.

1.10 (4 pts) Someone provides you the following function which does arithmetic with `float` values (32 bits). This function performs add, subtract, and multiply operations in some sequence with the arguments, though you don't know in which order. It returns a `float` value.

```
float doStuff( float x, float y ) ;
```

We want to use this function with `int` values, and to do so by casting from `int` to `float`. For example,

```
int j = 3, k = 4 ;
int result
  = static_cast<int>( doStuff( static_cast<float>( j ),
                               static_cast<float>( k ) ) ) ;
```

Does casting to float and back to int cause any problems for any arbitrary, but valid, values of j and k (answer YES, if it does, and NO if it does not). Assume a working compiler that does casting correctly. EXPLAIN for full credit.

1.11 (4 pts) Which CATEGORY (hint: there are 5 of them) do the following 32 bit floating point values belong?

(a) 0 1111 1111 0000 0000 0000 0000 0000 000

(b) 0 1111 1111 0011 0000 0000 0000 0000 000

(c) 0 0000 0001 0011 0000 0000 0000 0000 000

(d) 0 1000 0000 0000 0000 0000 0000 0000 000

(e) 0 0000 0000 1111 0000 0000 0000 0000 000

1.12 (3 pts) Convert -17.5 to IEEE single precision. Label bits clearly (sign bit, etc).

1.13 (3 pts) Assume the following representation for floating point:

- No sign bit (numbers are non-negative)
- 3 bits used to represent the mantissa/significand
- No hidden 1
- 3 bits used to represent the exponent (assume representation is unsigned).

You want to multiply $1.01_2 \times 2^{2_{10}}$ by $1.11_2 \times 2^{3_{10}}$. Both numbers are written in the desired representation (i.e., use 3 bits for number, etc). What is the answer in scientific notation using the above constraints? (Note: the values shown earlier are in scientific notation). Assume truncation is used.

1.14 (2 pts) Other than ASCII, name another (common) representation for characters. Describe any facts you know about this representation (one fact is fine).

Quiz 2 (Combinational Logic): "Conjunction Junction" (40 points)

Name	
Student ID	

2.1 (3 pts) Combinational logic circuits implement what mathematical abstraction?

2.2 (5 pts) A NAND_3-gate implements the Boolean function $f(w,x,y) = (wxy)'$

Implement this circuit using as FEW NAND_2 (2-inputs) gates as needed. Label inputs as w, x, y (draw circle around each input variables) and label output as z (draw a box around the z).

2.3 (5 pts) Implement a $NAND_2$-gate (2-input) using as few NOR_2 gates as necessary.

2.4 (5 pts) Implement a circuit which is equivalent to the carry out of a full adder using any part except a full adder or half adder. (See back of page 1 for definition of parts).

2.5 (5 pts) You wish to implement an AND_k (i.e., a k-input AND gate), but all you have are AND_2 gates (as many as you want). Assume the time delay of a gate is 1 unit. Write a formula that expresses the MINIMUM delay for an implementation of AND_k using AND_2 gates. EXPLAIN how you came up with this formula.

2.6 (10 pts) Fill out the following table, indicating how many control, data inputs, outputs each has (in general).

Use ”k” in your expression, as needed.

	Control Inputs	Data inputs	Outputs
MUX			
DeMUX			
Encoder			
Decoder			

2.7 (6 pts) Suppose you have a priority encoder with four inputs: x_3, x_2, x_1, x_0. In lecture, we said that higher indexes represent higher priorities. Suppose lower indexes represent higher priorities (index 0 has highest priority).

Write the Boolean expression for the outputs of z_1 and z_0, where $z_1 z_0$ represent the 2 bit subscript of the input with the highest priority.

2.8 (4 pts) Answer the following questions about encoders.

(a) What assumptions, if any, are made about the inputs of a regular encoder?

(b) What assumptions, if any, are made about the inputs of a priority encoder?

Quiz 3 (Low-level C functions/Odds and Ends): “10 Little Endians” (20 points)

Name	
Student ID	

3.1 (4 pts) The following are the 5 steps to execute an instruction. Explain each step.

IF

ID

ALU

MEM

WB

3.2 (5 pts) Here's a prototype for a C++ function:

`bool checkRange( unsigned int x, int low, int high ) ;`

Using bitwise operations and shifting, determine if AT LEAST ONE bit between index `low` and index `high`, inclusive, has value 1. Assume low $\leq$ high and that the indexes lie between 0 and 31, inclusive. Assume index 0 is the LSB and index 31 is the MSB.

3.3 (3 pts) Given the following 32 bit value: $ABCDEF01_{16}$

Write these values in a byte-addressable memory starting at address 1000 in LITTLE-ENDIAN order. Make sure you write ALL addresses properly.

3.4 (5 pts) Here's a prototype for a C++ function:

```
void clearBits( unsigned int & x, int low, int high ) ;
```

Clear bits (i.e, set it to 0) from index `low` to index `high`, inclusive, but leave remaining bits unchanged. Use same assumptions as `checkRange`.

3.5 (3 pts) Without using a temp variable (i.e., using logical bit operators), write a swap function

```
void swap( unsigned int & x, unsigned int & y ) ;
```

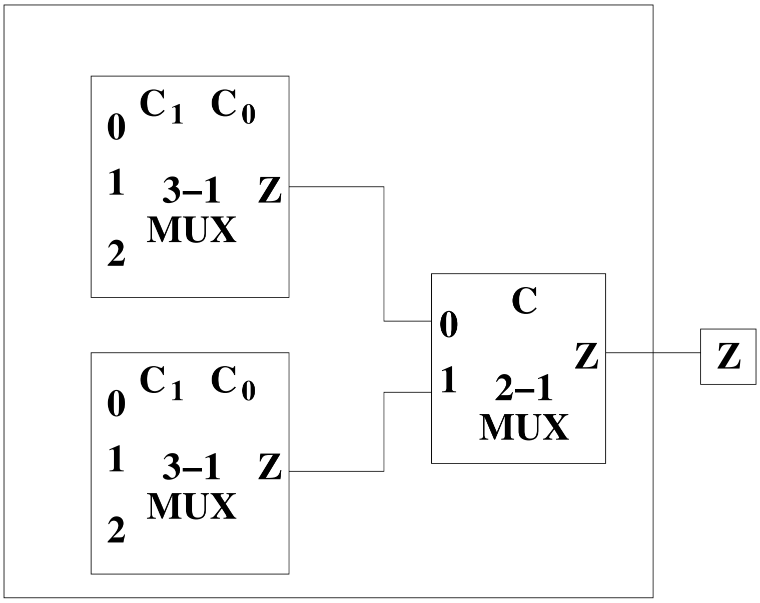
Challenge ("Drool Kitty") Section: "Meow!" (40 points)

C1 (10 pts) Consider the following 6-1 MUX built from 3-1 MUXes and 2-1 MUXes. The 3-1 MUXes have the following operation

c_1	c_0	z
0	0	x_0
0	1	x_1
1	0	x_2
1	1	random (don't care)

(a) Write a table for a 6-1 MUX and state assumptions about its operations.

(b) Clearly label inputs and outputs. Use conventions for data inputs using x with subscripts and control inputs using c with subscripts.



C2 (10 pts) Design a circuit which implements the following. Elegant answers (i.e. simple ones) receive more credit. However, any valid answers will get some credit.

x_3	x_2	x_1	x_0	z_3	z_2	z_1	z_0
0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0
0	0	1	0	0	1	0	0
0	0	1	1	1	1	0	0
0	1	0	0	0	0	0	1
0	1	0	1	1	0	0	1
0	1	1	0	0	1	0	1
0	1	1	1	1	1	0	1
1	0	0	0	0	0	1	0
1	0	0	1	1	0	1	0
1	0	1	0	0	1	1	0
1	0	1	1	1	1	1	0
1	1	0	0	0	0	1	1
1	1	0	1	1	0	1	1
1	1	1	0	0	1	1	1
1	1	1	1	1	1	1	1

Clearly label inputs (by circling them) and outputs (by boxing). You can use any part/gate you wish.

C3 (10 pts) For a carry lookahead adder, what are the general formulas for:

(a) g_i and p_i

(b) Briefly explain what g_i and p_i refer to (not just the names, but the purpose).

(c) Assume you are adding two 4-bit numbers

$$a_3a_2a_1a_0 + b_3b_2b_1b_0$$

and c_0 is the carry-in to the least significant bit (normally, a ripple-carry adder has no carry-in to the LSB, but pretend you have a full adder adding the LSB).

Write a formula for c_2 for a carry-lookahead adder, using only the following variables: $c_0, a_3, a_2, a_1, a_0, b_3, b_2, b_1, b_0$. Not all variables need to appear in the formula.

C4 (10 pts) Drool Kitty is a kitty that drools so much, that she needs a towel to sop it up. She also loves circuit problems. She argues that another way of performing two's complement operation is to subtract 1, and flip bits (the usual way is to flip bits and add 1) where subtract 1 may wrap around (thus 0000 - 1 results in 1111, for 4 bits). Is Drool Kitty correct?

If so, provide a mathematical argument. To do so, use facts about function composition and function inverses, and also check boundary cases (what happens near all 0's or all 1's).

If not, derive a counterexample where this fails. You should assume that the operation applies to k bit numbers where $k \geq 2$.