Computer Architecture
CS 429

Short In-Class Exam

Date: September 24, 2014
Unique Number: 52915, 52920, 52935, 52940, 52945, 52960, 52965
Instructor: Warren A. Hunt, Jr. & Bill Young
Teaching Assistants: Cuong Chau, Ji Hong, Keshav Kini, Wei-Ju Chen, Ben Selfridge

Time for Exam: 50 minutes

You should attempt to do all of the problems. Partial credit will be awarded on a problem by problem basis, so show your work. Be sure to state your assumptions carefully, and outline your solution as you work. Budget your time -- each problem can be done in 5 to 10 minutes. Please write your solutions directly on the exam. Each problem has the same value. Good luck!

Problems 1 - 2: Just as you are doing for your data manipulation laboratory, write straight-line C code to implement the code fragments below. Each code fragment is one problem. To remind you, here are the rules to follow when creating your solution.

Each "Expr" is an expression using ONLY the following:
1. Integer constants 0 through 255 (0xFF), inclusive. You are not allowed to use big constants such as 0xffffffff.
2. Function arguments and local variables (no global variables).
3. Unary integer operations ! ~
4. Binary integer operations & ^ | + << >>

Some of the problems restrict the set of allowed operators even further. Each "Expr" may consist of multiple operators. You are not restricted to one operator per line.

You are expressly forbidden to:
1. Use any control constructs such as if, do, while, for, switch, etc.
2. Define or use any macros.
3. Define any additional functions in this file.
4. Call any functions.
5. Use any other operations, such as ==, !=, <, >, =>, <=, &&, ||, -, or ?:
6. Use any form of casting.

You may assume that your machine:
1. Uses 2s complement, 32-bit representations of integers.
2. Performs right shifts arithmetically.
3. Has unpredictable behavior when shifting an integer by more than the word size.

Computer Architecture
CS 429

Short In-Class Exam

Date: September 24, 2014
Unique Number: 52915, 52920, 52935, 52940, 52945, 52960, 52965
Instructor: Warren A. Hunt, Jr. & Bill Young
Teaching Assistants: Cuong Chau, Ji Hong, Keshav Kini, Wei-Ju Chen, Ben Selfridge

NAME:

CS USERID:

UT EID:

Each problem worth 20 pts.

1: Write straight-line C code to implement the code fragment.

```
/*
 * copyLSB - set all bits of result to least significant bit of x
 *   Example: copyLSB(5) = 0xFFFFFFFF, copyLSB(6) = 0x00000000
 *   Legal ops: ! ~ & ^ | + << >>
 *   Max ops: 5
 *   Rating: 2
 */
int copyLSB(int x) {

        return (x<<31)>>31;

                  -OR-

        return ~((x&1) + ~0)

        (or anything else that worked)

}
```

2: Write straight-line C code to implement the code fragment.

```
/*
 * isPositive - return 1 if x > 0, return 0 otherwise
 *   Example: isPositive(-1) = 0.
 *   Legal ops: ! ~ & ^ | + << >>
 *   Max ops: 8
 *   Rating: 3
 */
int isPositive(int x) {

    return !(x>>31) & !!x;

}
```

3: Using the index values given, perform the following operation.
Values given in class.

	Young	Hunt
unsigned int shift =	3	3
unsigned int left =	6	6
unsigned int right =	5	11
int num =	−42	0x76543210

```
int  ans =  ( ( num >> shift ) << left ) >> right;
```

What is the value of variable ans?

Young

num = −42 = 11..1010110

num >> 3 = 11..1111010

<< 6 = 11..1010000000

>> 5 = 11..10100

ans = 0xfffffff4 = −12

Hunt

num = [0111 0110 0101] 43210 ([] means binary; otherwise, hex)

num >> 3 = [0000 1110 1100 101] 4321 [0]

<< 6 = [1011 0010 1] 4321 [0000000]

>> 11 = [1111 1111 1111 0110 0101] 432

= 0xfff65432

= −633806

4 pts each

4. Given the floating-point format on page 106 of the class textbook ("Computer Systems, A Programmer's Perspective"), represent the following numbers. If you need to round the number so it can be represented, then use round to even. Represent your answer as: S EEEE MMM where S is a sign bit (1 or 0), EEEE is the four-bit exponent, and MMM is the three-bit significand.

Example. 224: 0 1110 110

Young

a. $-3\frac{1}{4}$

-1.101×2^1

$e = 1 + bias = 8$

| 1 | 1000 | 101 |

Hunt

$-\frac{13}{2}$

-1.101×2^2

$e = 1 + bias = 9$

| 1 | 1001 | 101 |

b. $1\frac{15}{16}$

1.1111×2^0

Round to even: $2.0 \times 2^0 = 1.0 \times 2^1$

$e = 1 + bias = 8$

| 0 | 1000 | 000 |

$2\frac{7}{8}$

1.0111×2^1

Round to even: 1.100×2^1

$e = 1 + bias = 8$

| 0 | 1000 | 100 |

c. $\frac{5}{512}$

1.01×2^{-7} ← too small! denormal

$= 0.101 \times 2^{-6}$

| 0 | 0000 | 101 |

$\frac{-3}{512}$

-1.1×2^{-8} ← too small! denormal

$= -0.011 \times 2^{-6}$

| 1 | 0000 | 011 |

Perform the following operations as floating-point operations and state your answer using the above described format. Calculate the exact result and then use round-to-even rounding.

Young

d. $6\frac{1}{2} \cdot -3\frac{1}{8}$

$6\frac{1}{2}$: | 0 | 1001 | 101 | $= 6\frac{1}{2}$, 1.101×2^2

$-3\frac{1}{8}$: | 1 | 1000 | 100 | $= 3$, 1.100×2^2

$6\frac{1}{2} \times -3 = -19\frac{1}{2} = -1.00111 \times 2^4$ ← round-to-even: .010

| 1 | 1011 | 010 |

Hunt

d. $84 + 17$

84: | 0 | 1101 | 010 | $= 80$, 1.01×2^6

17: | 0 | 1011 | 000 | $= 16$, 1.0×2^4

$80 + 16 = 96 = 1.1 \times 2^6$

| 0 | 1101 | 100 |

e. $22 + 2\frac{3}{8}$

22: | 0 | 1011 | 011 | $= 22$, 1.011×2^4

$2\frac{3}{8}$: | 0 | 1000 | 010 | $= 2\frac{1}{2}$, 1.010×2^1

$22 + 2\frac{3}{8} = 24\frac{3}{8} = 1.1000011 \times 2^4$ ← rounds down to .100

| 0 | 1011 | 100 |

e. $-10 * \frac{3}{4}$

-10: | 1 | 1010 | 010 | $= -10$, -1.010×2^3

$\frac{3}{4}$: | 0 | 0110 | 100 | $= \frac{3}{4}$, 1.1×2^{-1}

$-10 * \frac{3}{4} = -7\frac{1}{2} = -111.1 = -1.111 \times 2^2$

| 1 | 1001 | 111 |

5. Perform the following multiplication operations. Given four-bit binary numbers calculate both the signed and unsigned products. This problem is just like Practice Problem 2.34 (on page 90).

Young!

Mode	x		y		x * y	Truncated x * y
Unsigned	8	[1000]	7	[0111]	56	8
Two's comp	-8	[1000]	7	[0111]	-56	-8
Unsigned	13	[1101]	5	[0101]	65	1
Two's comp	-3	[1101]	5	[0101]	-15	1

-1 pt for each blank, but -5pts if a whole line was blank.

Page left intentionally blank.

5. Perform the following multiplication operations. Given four-bit binary numbers calculate both the signed and unsigned products. This problem is just like Practice Problem 2.34 (on page 90).

Hunt:

Mode	x		y		x * y	Truncated x * y
Unsigned	7	[0111]	8	[1000]	56	8
Two's comp	7	[0111]	-8	[1000]	-56	-8
Unsigned	14	[1110]	13	[1101]	182	6
Two's comp	-2	[1110]	-3	[1101]	6	6

Page left intentionally blank.