CS3330 Exam 1 – Spring 2015

Name: ______________________

Directions: Put the letter of your selection or the short answer requested in the box. **Write clearly**: if we are unsure what you wrote you will get a zero on that problem.

There are several variants of this exam being given at the same time. Copying from your neighbor is not only cheating, it is also foolish.

Test proctors will *not* provide clarification during the exam. If you find something **ambiguous** or unclear, explain that clearly on your exam and add a $*$ to the top right corner of your answer box so we know to look for your note when grading.

Unless otherwise specified, all questions assume a **little-endian** computer.

If you do not sign the pledge on the last page you will get a zero on the entire exam.

. . . .

Question 1: A register has which of the following?

A a clock input
B more data outputs than data inputs
C a clock output
D more data inputs than data outputs

Answer: *A*

Question 2: Let f_{32} be the number of distinct numbers that can be represented in 4-byte IEEE-style floating point and u_{32} be the number of distinct numbers that can be represented as unsigned 4-byte integers. Which of the following is true?

A $f_{32} = u_{32} + 1$
B $f_{32} = u_{32}$
C $f_{32} > u_{32} + 1$
D $f_{32} < u_{32} - 1$
E $f_{32} = u_{32} - 1$

Answer: *D*

Question 3: Assume that eax is register number 0, that register eax contains the number 0x7, and that edx is register number 2. The assembly `2(%eax)` means:

A 2 + the value stored in memory at address `0x7`
B the value stored in memory at address `0x9`
C the same thing as `%edx`
D call function 2 with argument 0x7
E `0x9` (i.e., $2 + 7$)
F `0xE` (i.e., 2×7)
G the value stored in memory at address `0xE`
H `%edx` plus `%eax`
I call function 2 with argument 0
J `%edx` times `%eax`

Answer: *B*

Question 4: Let s_8 be the number of distinct numbers that can be represented as signed (two's-complement) 1-byte integers and u_8 be the number of distinct numbers that can be represented as unsigned 1-byte integers. Which of the following is true?

A $s_8 = u_8$
B $s_8 = u_8 - 1$
C $s_8 < u_8 - 1$
D $s_8 > u_8 + 1$
E $s_8 = u_8 + 1$

Answer: *A*

Question 5: Which of the five phases determines how many bytes long an instruction is?

A Fetch
B Decode
C Writeback
D Execute
E Memory
F Which of the above depends on which instruction is being run

Answer: *A*

Question 6: `a & b` is the bitwise version of `a && b`; `a | b` is the bitwise version of `a || b`; which of the following is the bitwise version of `a == b`?

A `~(a ^ b)`
B `~(a = b)`
C `a = b`
D `a ^ b`
E `~a ^ ~b`
F `~a = ~b`

Answer: *A*

Question 7: Four-bit binary `1011 - 0110` is: (answers in binary)

A `0100`
B `0010`
C `1101`
D `1111`
E `0001`
F `0101`
G `0011`

Answer: *F*

Question 8: Which of the five phases writes a value when running `rmmovl %eax, (%ebx)`?

A Memory
B Execute
C Decode
D Writeback
E Fetch
F More than one of the above
G None of the above

Answer: *A*

Question 9: Y86 does not have an instruction for jumping to an address stored in a register location; instead the conceptual action `jmp %eax` can be implemented by

A `rmmovl %eax, $1234 ; jmp $1234`
B `pushl %eax ; ret`
C `rrmovl %eax, %PC`
D All of the above work
E None of the above work

Answer: *B*

Question 10:

What is the result of binary 1100 times decimal 10? Write your answer in hexidecimal (no leading 0x or leading 0s, just the hex digits)

Answer: *78*

Question 11:
Consider an 3-bit IEEE-style floating point number with 1 exponent bit (bias 0). How many distinct finite numbers can it represent? Answer as a number written in decimal (e.g., if the answer is eleven write 11, not 0xB or 0b1011).

Answer: *3*

Question 12:
How many bits are used in the smallest IEEE-style floating-point number format that is able to represent all numbers between -7 and 7 (inclusive)? Answer as a number written in decimal (e.g., if the answer is eleven write 11, not 0xB or 0b1011).

$-7 = 1\ \ 101\ \ 11$

Answer: *6*

Question 13:
A hardware multiplexer has 1 data output and 3 selection inputs; how many data inputs does it have? Answer as a number written in decimal (e.g., if the answer is eleven write 11, not 0xB or 0b1011).

Answer: *8*

Question 14: Register `%esi` is one of the "caller-save" registers. This means that if procedure `baz` calls procedure `xyxxy`

A `xyxxy` may communicate with `baz` through register `%esi`
B `baz` must assume `xyxxy` could have changed `%esi`
C `baz` may assume `xyxxy` did not change `%esi`
D `baz` may communicate with `xyxxy` through register `%esi`

Answer: *B*

Question 15: A `call` instruction is conceptually

A a `popl`
B a `popl` and a `jmp`
C a `jmp`
D a `pushl`
E a `pushl` and a `jmp`

Answer: *E*

Question 16: Register `%edx` is one of the "callee-save" registers. This means that if procedure `baz` calls procedure `xyxxy`

A `xyxxy` may communicate with `baz` through register `%edx`
B `baz` must assume `xyxxy` could have changed `%edx`
C `baz` may communicate with `xyxxy` through register `%edx`
D `baz` may assume `xyxxy` did not change `%edx`

Answer: *D*

Question 17: Which of the five phases has no work to do when executing a `call` instruction?

A Memory
B Decode
C Writeback
D Fetch
E Execute
F All of the above have work to do for a `call` instruction

Answer: *F*

Question 18: If we did not have an assembly instruction for `pushl`, we could achieve the same functionality in x86 by (pick the minimal correct answer):

A one `movl` instruction
B one `addl` (or `subl`) instruction and two `movl` instructions
C two `movl` instructions
D two `addl` (or `subl`) instructions and one `movl` instruction
E one `addl` (or `subl`) instruction and one `movl` instruction
F two `addl` (or `subl`) instructions and two `movl` instructions
G None of the above, though `pushl` can be replaced by other operations
H None of the above because `pushl` cannot be replaced by other operations

Answer: *E*

Question 19: Initially byte i of memory is $i + \text{0x20} \pmod{256}$, `%eax` contains the number 3, and `%ebx` contains the number 5. What is in `%ebx` after running x86 `movl (%eax), %ebx` (or Y86 `mrmovl (%eax), %ebx`)?

A 3
B 5
C 0x25
D 0x23
E None of the above
F Insufficient information given to determine which of the above

Answer: *E*

Question 20: Which of the five phases determines what the next PC is?

A Memory
B Fetch
C Decode
D Execute
E Writeback
F Which of the above depends on which instruction is being run

Answer: *B*

Question 21: Initially byte i of memory is $i + 0x20 \pmod{256}$, `%eax` contains the number 0x01020304, and `%ebx` contains the number 5. What is in byte 0x06 of memory after running x86 `movl %eax, (%ebx)` (or Y86 `rmmovl %eax, (%ebx)`)?

- **A** 0x02
- **B** 0x01
- **C** 0x03
- **D** 0x26
- **E** 0x04
- **F** None of the above
- **G** Insufficient information given to determine which of the above

Answer: C

Question 22: Which of the following is more typically true of CISC than RISC instruction sets?

- **A** register operands have the same location within every instruction
- **B** all instructions are the same length
- **C** many program registers
- **D** a clean and simple set of opcodes
- **E** most opcodes can accept operands in several addressing mode

Answer: E

Question 23: Edsger Dijkstra wrote "Go To Statement Considered Harmful" in 1968 in which he argued (translated into current language) that code that uses `goto` becomes needlessly hard to read and maintain compared to only using `if` and `while`. In assembly, we use the `goto`-like jump instructions extensively; the main reason for this is

- **A** Dijkstra was wrong
- **B** no one is reading assembly anyway
- **C** there is no other alternative in assembly
- **D** `goto` is only bad when mixed with higher-level constructs

Answer: C

Question 24: Which of the five phases determines the destination register for instructions that write to registers?

- **A** Writeback
- **B** Fetch
- **C** Memory
- **D** Decode
- **E** Execute
- **F** Which of the above depends on which instruction is being run

Answer: B

Question 25: Consider a 5-bit IEEE-style floating point number with 3 exponent bits (bias 3). Which of the following *can* be expressed in this format? Options are written in binary

- **A** 1.01
- **B** -0.0001
- **C** 11000
- **D** -111
- **E** All of the above can be represented in this format
- **F** None of the above can be represented in this format
- **G** None of the above because you can't have a 5-bit IEEE-style floating point number with 3 exponent bits

Answer: F

Pledge:

On my honor as a student, I have neither given nor received aid on this exam.

Your signature here