# Computer Systems Organization
# CSCI-UA.0201 Spring 2021
# SAMPLE Final Exam

**Note: This is a little longer than the actual final exam will be.**

1. True/False. Please circle the correct response.

   a. T F In the C and assembly calling convention that we used for this class, all arguments to a function call are passed in registers (no matter how many).

   b. T F In C, for the expression `(x | THE_MASK)`, where `THE_MASK` has at least one bit that is not zero, the result will be zero if all the bits of `x` are zero.

   c. T F In x86-64 assembly, the instruction `movq 16(%rcx),%rax` will add 16 to the value contained in the %rcx register in order to compute a memory address.

   d. T F In x86-64 assembly, the instruction `popq %rbx` adds 8 to the value of the %rsp register.

   e. T F When processor with a direct-mapped cache issues a request to load data from memory, if the tag in the specified address matches the tag in the corresponding cache entry, then a cache hit results – no matter what the status of any other bit in the cache entry is.

   f. T F Any circuit constructed using only AND, OR, or NOT gates can also be constructed using only NAND gates (which perform an AND and then a NOT).

   g. T F The use of a write-through cache is slower than a write-back cache because, using a write-through cache, the CPU must wait until the data is written to main memory before proceeding with the next instruction.

   h. T F A multiplexer uses N select lines to select from $2^N$ input lines.

   i. T F A 32-bit adder can be used for subtraction if each bit of the second operand is first sent through a NOT gate and the carry-in to the adder is set to 1.

   j. T F In an unclocked latch, setting both the S and the R inputs to 0 will cause the output Q to retain its current value.

2.

   a. Write the number AB31 hex in binary: ______________________________.

   b. Write the number 73 decimal in hex: ______________ and in binary: _____________.

   c. log 16G = ______________ .

   d. In order to access all bytes in a 64GB memory, an address must have at least _____ bits.

   e. If an integer variable is represented by 25 bits, how many different numbers could that variable take on? ____________

3. Write a C function, `int foo(int x)`, that returns the index of the most significant bit of `x` whose value is 1. For example, if bit 15 of `x` is the most significant bit of `x` whose value is 1, then `foo` should return 15. If no bits are 1, then `foo` should return -1.

4. Given the following definition of a `CELL` type used for the elements of linked lists,

```
typedef struct cell {
   long x;
   struct cell *next;
} CELL;
```

a. Write a C function, `long sumList(CELL *head)`, that returns the sum of the `x` fields in the cells of a linked list whose first element is pointed to by `head`.

b. Translate your C function into x86-64 assembly code for either Unix (e.g. MacOS or Linux) or Cygwin/Windows. You can assume that the two fields of a `CELL` are next to each other, with `x` coming first. The calling conventions for Unix and Windows are provided at the end of this exam.

5.

a. On modern processors, a C program whose main loop reads the elements of a huge array in random order will run much slower than an otherwise identical C program that reads the array elements in consecutive order. Why is that?

b. Suppose that, on a computer with only one cache, the following holds:

- the cache has a 90% hit rate for both data and instructions,
- if an instruction does not cause a cache miss, executing the instruction takes one cycle,
- the cache miss penalty is 200 cycles

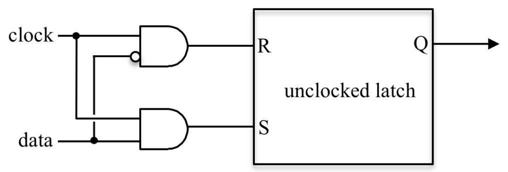
What is the overall execution time (in cycles) of a program that executes N instructions, where 25% of the instructions access data in memory?

c. On a machine with 32-bit words, suppose there is an 8MB, 2-way set associative cache with 16 words per cache line. Further, suppose that an instruction issues a memory address for loading a single word from memory into a register. Indicate how the various bits of the address are used by the cache hardware to determine whether a cache hit has occurred and, if so, to return the requested word.

6.

a. Build, from AND, OR, and NOT gates, a circuit that represents the two-bit "less-than" function. That is, it has two two-bit inputs, A and B, and a single one-bit output, R, such that R is true when A< B. [Hint: enumerate the possible inputs for which the output is true.]

b. As you saw in class, a clocked latch is built from an unclocked latch as shown below. Why are flip-flops used for storing bits in a CPU rather than clocked latches?



c. Build from gates, multiplexers, decoders, and/or adders (including 32-bit versions of each) a circuit that takes two 32-bit inputs, X and Y, and outputs the value of the larger of X and Y. For example, if the value of X is 20 and the value of Y is 15, then the output of your circuit should be 20.

d. Draw the datapath for an X64 instruction, such as `addq`, that has two registers as operands, where the second operand is also the destination register (where the result is put). Include the portions of the processor that use the PC to fetch the instruction and update the PC to point to the next instruction.

e. Suppose that, instead of having two register operands, the `addq` instruction specified three registers, where the third operand is the destination. That is, for example, suppose the `addq` instruction is written:

```
addq  %rax,%rbx,%rcx   # sets %rcx = %rax + %rbx
```

In this case, what would be different about the datapath than what you drew for the previous part?

# X86-64 Calling Conventions

## Unix (e.g. macOS and Linux)

Passing Parameters:

- First six integer/pointer parameters (in order): %rdi, %rsi, %rdx, %rcx, %r8, %r9
- Additional parameters are passed on the stack, pushed in right-to-left (reverse) order.

Return Value:

- %rax (for 64-bit result), %eax (for 32-bit result)

Caller-Saved Registers (may be overwritten by called function):

- %rax , %rcx , %rdx , %rdi , %rsi , %rsp , %r8, %r9, %r10, %r11

Callee-Saved Registers (must be preserved – or saved and restored – by called function):

- %rbx, %rbp, %r12, %r13, %r14, %r15

## Microsoft (e.g. for Windows)

Passing Parameters:

- First four integer/pointer parameters (in order): %rcx, %rdx, %r8, %r9
- Additional parameters are passed on the stack, pushed in right-to-left (reverse) order.
- Important: The caller must allocate 32 bytes on stack (by subtracting 32 from %rsp) right before calling the function. Don’t forget to restore the %rsp (by adding 32) after the call.

Return Value:

- %rax (for 64-bit result), %eax (for 32-bit result)

Caller-Saved Registers (may be overwritten by called function):

- %rax, %rcx, %rdx, %r8, %r9, %r10, %r11

Callee-Saved Registers (must be preserved – or saved and restored – by called function):

- %rbx, %rbp, %rdi, %rsi, %rsp, %r12, %r13, %r14, %r15