463-1 Fall 2017 REVIEW SHEET FOR EXAM #2

The format will be similar to the last exam, i.e., conceptual questions and problems to work. Bring a pencil because you will need something to write with. The output formats will include charts, numbers, etc. similar to the HW.

Conceptual questions will be based on the following slides: 4a1, 4a2, 4bcefg, 2fghi. (Slides 2de are also available if you need to review binary/decimal conversion.) A list of the potential conceptual questions is below; however, the format may be different.

Problems to work will be based on the following assignments: HW4, HW5, HW6.

Make sure you have memorized the prefixes listed on 4a2-9bc and 4a2-12c as "memorize these". Also make sure you know which chart (9b or 12c) applies to clock speed, memory and disk.

I will give you the following reference material: a table of powers of 2, an RTL chart for the MARIE instructions (i.e., a sheet like the one that came with the homework), a picture of the MARIE datapath showing the bus addresses, and a list of microcode instructions for MARIE.

Simple calculators are permitted but not required. Cell phones, laptops, tablets, desktop computers (i.e., the one in the front of the classroom) and other electronic devices are not permitted. Devices with a text memory are not permitted. You may not share a calculator with another student. If you have any doubts about your calculator, the detailed calculator policy is on the last page of this review sheet.

Part I. Core architecture

1. What is the datapath? control unit? CPU? ALU? What is the relationship between them?

2. What are the three types of lines on a bus? What does each of them do?

3. How many address lines are required for a byte-addressable machine with 2^n bytes of memory?

4. What is the difference between a point-to-point bus and a multipoint bus?

5. What is bus arbitration? Name several types of bus arbitration and the differences between them.

6. What is a system clock? What is clock cycle time? What is frequency? What is the relationship between them?

7. What are four types of I/O and their main characteristics? Which one is obsolete (but we mention it anyway to explain the motivation for interrupt-driven I/O, and in case anyone wants to understand MARIE I/O)? Which one is used primarily in mainframes?

Part II. The instruction cycle

1. What do the AC, MAR, MBR, PC and IR do in MARIE? How many bits does each have? Why does each register have the number of bits that it has? (Hints: How many instructions does MARIE have? Remember that MARIE has 4K 16-bit words of word-addressable memory.)

2. What is the fetch-decode-execute cycle? What happens in each step? How is the fetch-decode-execute cycle different depending on whether the instruction contains an address operand or not?

3. Which MARIE instructions need a fetch operand step? What is the maximum number of fetch operand steps a MARIE instruction can have? Why?

4. What is RTL? What is a microoperation? Which steps in the execution cycle (fetch, decode, fetch operand, execute) use the same RTL for every instruction? What does the RTL for fetch do? the RTL for decode? the RTL for fetch operand? the RTL for execute?

5. Under what circumstances can two RTL instructions be written on the same line?

6. What data format does MARIE use internally (decimal, unsigned binary, or 2's complement)? What do the DEC and BIN assembler directives do?

7. What is an interrupt? What types of things cause interrupts? Under what conditions would one want to disable interrupts? What is a non-maskable interrupt? Why can some interrupts not be disabled? How do interrupts interact with the fetch-decode-execute cycle?

8. The 'execute' section of the RTL for LOAD on the reference chart in HW4 contains 3 instructions:

```
MAR   <- X
MBR   <- M[MAR]
AC    <- MBR
```

The 'execute' section of the RTL for LOAD on slide 4c-39 only includes one instruction:

```
AC    <- MBR
```

The shorter version must work or it wouldn't be in the slides. Why does it work, i.e., why don't you need the longer version?

(So why is the longer version on the reference chart and in the textbook? So that you know the context in which that last line has to be executed. If the other lines weren't there, it would look like loading any old value of MBR into AC would work.)

9. What is indirect addressing? Which MARIE instruction(s) use indirect addressing?

10. How do you call a subroutine in MARIE? How do you return from a subroutine? Why can't MARIE handle recursive calls (instances where a function calls itself)?

Part III. Instruction decoding

1. What is hardwired control? What is microprogramming? What is the difference between them?

2. What is a signal line? What is a datapath address? How many signal lines (input and output) does a machine need if there are 2^n addresses on the datapath? What is a timing signal? How many timing signals are needed to handle all the instructions, including the indirect addressing instructions and the other advanced ones in section 4e (i.e., all the instructions on the chart in HW4)? Why do some instructions need more timing signals than others? What does the counter reset signal do? When is it triggered? What would happen if we didn't have a counter reset signal? Do these terms refer to a hardwired machine or a microprogrammed machine?

3. What do the following symbols represent: P_0-P_5, T_0-T_7, A_0-A_3, C_r?

4. What is firmware? How often are microcode instructions retrieved from the firmware? Why do MARIE microinstructions have space for two microoperations? What goes in the second one if it's not used in a given microinstruction? What are the Jump and Dest fields and how are they used?

5. Which steps of the fetch-decode-execute cycle are located before the jump table in MARIE's microcode? Which lines of the microcode implement fetch? decode? fetch operand? execute? What does the jump table contain? What is at the addresses jumped to by the jump table? Why does the last line of each of the jumped-to sections have the jump flag set? Where does each of these lines jump to?

Part IV. Conceptual questions on number representation

1. When you look at a number in two's complement form, how can you tell if it is positive or negative?

2. What does overflow mean? When you add two numbers in unsigned binary, how can you tell if there is overflow? When you add two numbers in two's complement notation, how can you tell if there is overflow?

3. How many two's complement numbers can be represented with n bits? How many are positive? How many are negative? How many have the value of zero? What is the lowest positive number? What is the highest positive number? What is the lowest negative number? What is the highest positive number?

4. How many unsigned numbers can be represented with n bits?

5. Why are signed numbers useful? Why are unsigned numbers useful?

6. Why are two's complement numbers useful? How do you do subtraction with two's complement numbers? (Note: One's complement is only important as a step on the way to two's complement and will not specifically appear on the exam.)

7. How do you convert an 8-bit unsigned binary number to a larger field, e.g., a 12-bit binary number? I.e., where do you put the zeroes?

8. How do you convert a binary fraction with 4 bits after the binary point to a larger field, e.g., to a binary number with 8 bits after the binary point? I.e., where do you put the zeroes?

9. How do you convert an 8-bit two's complement binary number to a larger field, e.g., a 12-bit binary number? I.e., what bit do you propagate? Why does this look different for positive and negative numbers? (Hint: same as for one's complement).

10. What is Booth's algorithm used for? (You do not need to know how to do it.)

Calculator policy:

You do not need a calculator for the exams. All of the problems can be done by hand in the time available. However, most students do prefer to use them. You may use any simple four-function or scientific calculator. If you are planning to bring a calculator, I highly recommend a scientific one (i.e., one that can use exponents). You may not share a calculator with another student.

This calculator policy is a result of student feedback in previous semesters. Your feedback on all aspects of the course is always welcome.

The calculator policy is essentially the same as the one used by the ACT and the College Board. The following types of calculators are prohibited:

- Portable or handheld computers, laptops, tablets, desktop computers, electronic writing pads, and pocket organizers
- Devices with wireless or Bluetooth capability, cell phone capability, or any way of accessing the Internet
- Devices with audio/video recording capability, a digital audio/video player, a camera or scanning capability
- Calculators with built-in computer algebra systems
- Electronic writing pads, pen-input devices or any device with stylus or touch-screen capability
- Calculators with a typewriter keypad or text memory
- Calculators with paper tape, unless you remove the paper tape
- Calculators that make noise, unless you turn off the sound
- Calculators with an infrared data port, unless the infrared data port is completely covered with duct tape or electrical tape
- Calculators that use a power cord, unless you run it from a battery
- Devices to which hardware peripherals have been added