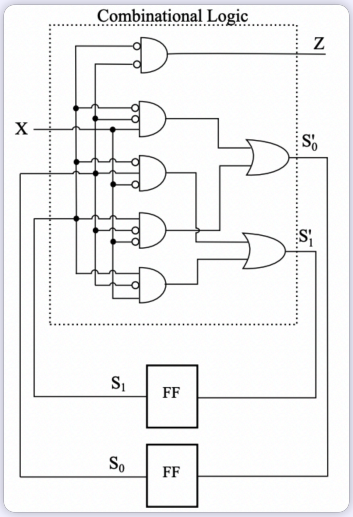
ICS hw3

howlingggggg

October 29, 2022

T1. The logic diagram shown below is a finite-state machine.



(a). Construct the truth table for the combinational logic:

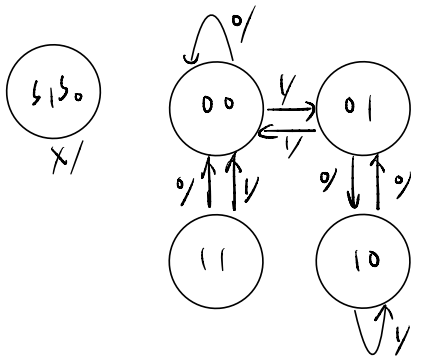
S_1	S_0	X	Z	S_1'	S_0'
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

(b). Draw the state machine.

answer.

(a). $Z = \bar{S}_1\bar{S}_0$, $S_1' = (\bar{S}_1 S_0 \bar{X}) + (S_1\bar{S}_0 X)$, $S_0' = (\bar{S}_1\bar{S}_0 X) + (S_1\bar{S}_0\bar{X})$.

S_1	S_0	X	Z	S_1'	S_0'
0	0	0	1	0	0
0	0	1	1	0	1
0	1	0	0	1	0
0	1	1	0	0	0
1	0	0	0	0	1
1	0	1	0	1	0
1	1	0	0	0	0
1	1	1	0	0	0



(b).

T2. After these two instructions execute:

```
1  x3030  0001  000  001  0  00  010
2  x3031  0000  011  000000111
```

The next instruction to execute will be the instruction at x3039 if what?
Hint: refer to page 133 Example 4.5.

answer. If (R0 ADD R2) is positive or zero.

T3. An LC-3 instruction is made up of two parts, `Bits[15:12]` and `Bits[11:0]` .

(a). What do `Bits[15:12]` specify?

(b). What do `Bits[11:0]` specify?

answer.

(a). opcode

(b). operands

T4. Say it takes 100 cycles to read from or write to memory and only one cycle to read from or write to a register.
Calculate the number of cycles it takes for each phase of the instruction cycle for the LC-3 instruction

```
1 ADD R6, R2, R6
```

Assume each phase (if required) takes one cycle unless memory access is required.
Hint: refer to page 135 Figure 4.4

answer.

- fetch: $1 + 100 + 1 = 102$
- decode: 1
- execute: 1

T5. Suppose a 32-bit instruction takes the following format:

OPCODE	SR	DR	IMM

If there are 56 opcodes and 64 registers, what is the range of values that can be represented by the immediate (IMM)? Assume IMM is a 2's complement value.

answer. OPCODE: $\lceil \log_2 56 \rceil = 6$ bits, SR/DR: $\lceil \log_2 64 \rceil = 6$ bits. $\Rightarrow$ IMM: $32 - 6 - 6 \times 2 = 14$ bits
range: 00 0000 0000 0000 11 1111 1111 1111

T6. Suppose we changed the LC-3 to have **only four registers** instead of 8. Fewer registers are in general a bad idea since it means loading from memory and storing to memory more often.

(a). If we keep the basic format of all instructions as they currently are (and keep each instruction 16 bits), is there any benefit that could be had for operating (0001, 0101, 1001) instructions, if we reduce the number of registers to 4?

(b). Is there any benefit that could be had for load (0010) and store (0011) instructions if we reduce the number of registers to 4?

(c). Is there any benefit that could be had for conditional branch (0000) instructions if we reduce the number of registers to 4?

answer.

(a). Increasing the imm range from $-16 \sim 15$ to $-64 \sim 63$.

(b). PCoffset9 $\rightarrow$ PCoffset10.

(c). no.

T7. Consider these three instructions: ADD, STR, and JMP.

The PC, IR, MAR, and MDR are written in various phases of the instruction cycle, depending on the opcode of the particular instruction. In each location in the following table, enter the opcodes that write to the corresponding register (row) during the corresponding phase (column) of the instruction cycle.

	fetch	decode	evaluate address	fetch data	execute	store result
PC						
IR						
MAR						
MDR						

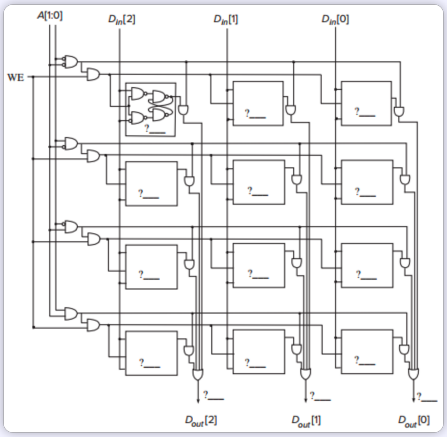
answer. 1. ADD, 2. STR, 3. JMP

	fetch	decode	evaluate address	fetch data	execute	store result
PC	1, 2, 3			3		
IR	1, 2, 3					
MAR	1, 2, 3			1, 2		
MDR	1, 2, 3				1, 2	

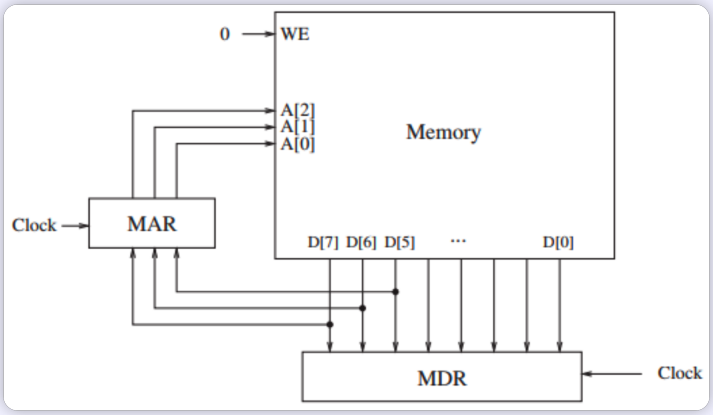
T8. The 2^2-*by*-3 bit memory discussed in class is accessed during five consecutive clock cycles. The table below shows the values of the two-bit address, one-bit write enables, and three-bit data-in signals during each access.

	A[1:0]	WE	D_{in}[2:0]
cycle 1	0 1	1	1 0 1
cycle 2	1 1	0	1 1 0
cycle 3	1 0	1	0 1 0
cycle 4	0 1	1	0 1 1
cycle 5	1 1	0	1 0 1
cycle 6	0 0	1	1 1 1
cycle 7	1 1	1	1 1 1
cycle 8	1 1	0	0 1 0

Your job: Fill in the value stored in each memory cell and the three data-out lines just before the end of the eighth cycle. Assume initially that all 12 memory cells store the value 1. In the figure below, each question mark (?) indicates a value that you need to fill in.



T9. Shown below is a byte-addressible memory consisting of eight locations, and its associated MAR and MDR. Both MAR and MDR consist of flip-flops that are latched at the start of each clock cycle based on the values on their corresponding input lines. A memory read is initiated every cycle, and the data is available by the end of that cycle.



Just before the start of cycle 1, MAR contains 000, MDR contains 00011001, and the contents of each memory location is as shown.

Memory Location	Value
x0	00110000
x1	11110001
x2	10000011
x3	00010101
x4	11000110
x5	10101011
x6	00111001
x7	01100010

(a). What do MAR and MDR contain just **before** the end of cycle 1?

(b). What does MDR contain just before the end of cycle 4?

answer.

(a.) MAR:000 MDR:00011001

(b.) 01100010

T10. In this problem, we perform five successive accesses to memory. The following table shows for each access whether it is a read (load) or write (store), and the contents of the MAR and MDR at the completion of the access. Some entries are not shown. Note that we have shortened the addressability to 5 bits, rather than the 16 bits that we are used to in the LC-3, in order to decrease the excess writing you would have to do.

Operations on Memory

	R/W	MAR	MDR				
Operation 1	W		1	1	1	1	0
Operation 2							
Operation 3	W		1	0			
Operation 4							
Operation 5							

The following three tables show the contents of memory locations x4000 to x4004 before the first access, after the third access, and after the fifth access. Again, not all entries are shown. We have added an unusual constraint to this problem in order to get one correct answer. The MDR can **ONLY** be loaded from memory as a result of a load (read) access.

Memory before Access 1

x4000	0	1	1	0	1
x4001	1	1	0	1	0
x4002		1			
x4003	1	0	1	1	0
x4004	1	1	1	1	0

Memory after Access 3

x4000					0
x4001		0			0
x4002					
x4003					
x4004	1	1	1	1	0

Memory after Access 5

x4000					
x4001					
x4002					
x4003	0	1	1	0	1
x4004	1	1	1	1	0

Your job: Fill in the missing entries **in all four tables**.
Hint: As you know, writes to memory require MAR to be loaded with the memory address and MDR to loaded with the data to be written (stored). The data in the MDR must come from a previous read (load).

T11. Suppose a 32-bit instruction takes the following format:

OPCODE	DR	SR1	SR2	UNUSED

If there are 225 opcodes and 120 registers,

(a). What is the minimum number of bits required to represent the OPCODE?

(b). What is the minimum number of bits required to represent the destination register (DR)?

(c). What is the maximum number of UNUSED bits in the instruction encoding?

answer.

(a). $\lceil \log_2 225 \rceil = 8$ bits.

(b). $\lceil \log_2 120 \rceil = 7$ bits.

(c). $32 - 8 - 7 \times 3 = 3$ bits.

T12.

(a). If a machine cycle is 2 nanoseconds (i.e., 2×10^{-9} seconds), how many machine cycles occur each second?

(b). If the computer requires on the average eight cycles to process each instruction, and the computer processes instructions one at a time from beginning to end, how many instructions can the computer process in 1 second?

(c). Preview of future courses: In today's microprocessors, many features are added to increase the number of instructions processed each second. One such feature is the computer's equivalent of an assembly line. Each phase of the instruction cycle is implemented as one or more separate pieces of logic. Each step in the processing of an instruction picks up where the previous step left off in the previous machine cycle. Using this feature, an instruction can be fetched from memory every machine cycle and handed off at the end of the machine cycle to the decoder, which performs the decoding function during the next machine cycle while the next instruction is being fetched. Ergo, the assembly line. Assuming instructions are located at sequential addresses in memory, and nothing breaks the sequential flow, how many instructions can the microprocessor execute each second if the assembly line is present? (The assembly line is called a pipeline, which you will encounter in your advanced courses. There are many reasons why the assembly line cannot operate at its maximum rate, a topic you will consider at length in some of these courses.)

answer.

(a). $(2 \times 10^{-9})^{-1} = 5 \times 10^8$ cycles.

(b). $5 \times 10^8 / 8 = 6.25 \times 10^7$ instructions.

(c). $5 \times 10^8 - 5$ instructions

T13. State the phases of the instruction cycle, and briefly describe what operations occur in each phase.

answer.

1. Fetch: MAR ← PC; MDR ← next instruction; IR ← MDR; incrementing the PC.
2. Decode: examine the instruction.
3. Evaluate Address: computing the address of memory.
4. Fetch Operand: obtain the operand.
5. Execute: execute the instruction.
6. Store Result: write the result to the designated destination.