Homework 3

T1

Your task is to consider the successor to the LC-3. We will add 16 addition opcodes to the ISA and the size of register set remains 8. The memory is byte-addressable, with total address space of 32K bytes. Instructions will remain 16 bits wide, though we may need to change the size of some of the fields.

1. How many bits do we need in the PC to be able to address all of memory?
2. What is the largest immediate value we could represent in an ADD instruction on this new machine?
3. What is the address range that a LD instruction is able to cover on this new machine?

T2

> Please refer to the `3rd` edition textbook rather than the 2nd edition. There may exist some differences.

Instruction cycle consists of six sequential phases: Fetch, Decode, Evaluate Address, Fetch Operands, Execute, Store Result. But not all instructions require all six phases.

1. State the phases of the instruction cycle, and briefly describe what operations occur in each phase.
2. Complete the following table. If a certain opcode requires a certain phase, please check the corresponding box.

	FETCH	DECODE	EVALUATE ADDRESS	FETCH OPERANDS	EXECUTE	STORE RESULT
ADD						
AND						
ST						
STR						
LDI						
LEA						
BR						
JMP						

3. Suppose it takes 3 cycles to fetch an instruction, 1 cycle to decode and 1 cycle to execute. It takes 100 cycles to fetch operands and store result if an instruction needs to read from or write to memory; otherwise it takes only 1 cycle to read from or write to a register. As for evaluate address, it takes 1 cycle for PC-relative mode and

base+offset mode, while 100 cycles for indirect mode. Your task is to calculate the number of cycles it takes for the following LC-3 program.

Address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x30F6	1	1	1	0	0	0	1	1	1	1	1	1	1	1	0	1
x30F7	0	0	0	1	0	1	0	0	0	1	1	0	1	1	1	0
x30F8	0	0	1	1	0	1	0	1	1	1	1	1	1	0	1	1
x30F9	0	1	0	1	0	1	0	0	1	0	1	0	0	0	0	0
x30FA	0	0	0	1	0	1	0	0	1	0	1	0	0	1	0	1
x30FB	0	1	1	1	0	1	0	0	0	1	0	0	1	1	1	0
x30FC	1	0	1	0	0	1	1	1	1	1	1	1	0	1	1	1

4. Consider these three instructions: ADD, STR and JMP. Complete the following table. Enter "R" or/and "W" if the corresponding component may be read or written during the corresponding phase.

	FETCH	DECODE	EVALUATE ADDRESS	FETCH OPERANDS	EXECUTE	STORE RESULT
PC						
IR						
MAR						
MDR						
MEM						
registers						
NZP						

5. 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 themachine cycle to the decoder, which performs the decoding function during the next machine cycle while the next instruction is being fetched. Assuming instructions are located at sequential addresses in memory, nothing breaks the sequential flow, and each instruction takes 6 machine cycle to execute. How many instructions can the microprocessor execute in 1000 machine cycles 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.)

T3

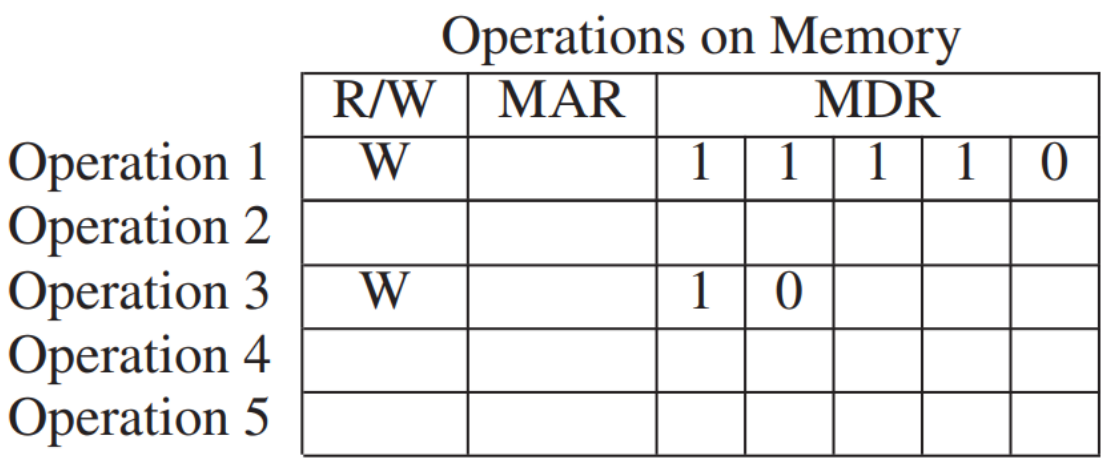
Suppose the following LC-3 program is loaded into memory starting at location x3000:

```
x3000: 0101 000 001 1 00000
x3001: 0001 000 000 1 00001
x3002: 0001 010 001 0 00 000
x3003: 0001 010 001 0 00 010
x3004: 1001 010 010 111111
x3005: 0000 010 000000001
x3006: 1111 0000 00100101
x3007: ......
```

1. Suppose the code at x3007 is executed. What is the initial value of R1?
2. We would like to add XOR operation with the unused opcode 1101. The format of XOR instruction is the same as ADD and AND. Rewrite the above program using the XOR instruction at least once.
3. What changes should we make to the current architecture to implement XOR instruction?

T4

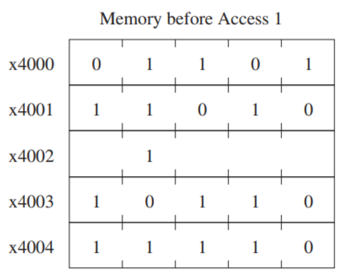
In this problem we perform five successive accesses to memory. The following table shows for each access whether it is a read (load) or write (store), and the contents of the MAR and MDR at the completion of the access. Some entries are not shown. Note that we have shortened the addressability to 5 bits, 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

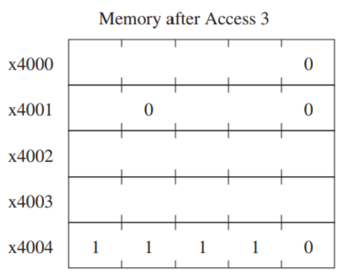
	R/W	MAR	MDR				
Operation 1	W		1	1	1	1	0
Operation 2							
Operation 3	W		1	0			
Operation 4							
Operation 5							

The following five tables show the contents of memory locations x4000 to x4004 before the first access, after the 3nd access and after the 5th 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.



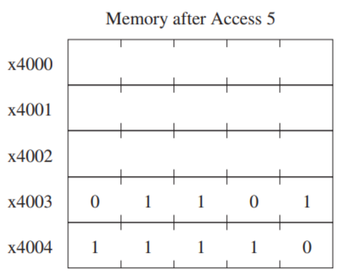
Memory before Access 1

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

T5

An LC-3 program is stored in memory locations `x3000` to `x3005`. Note that the branch instruction in memory location `x3002` has an unspecified PCoffset9, denoted as X.

address	instruction
x3000	0101 000 000 1 00000
x3001	0001 000 000 1 00010
x3002	0000 011 X
x3003	0001 000 000 1 00011
x3004	0001 000 000 1 00001
x3005	1111 0000 0010 0101

The program starts executing with PC = `x3000`.

Your job: In the table below, for each value of X, answer the question: "Does the program halt?"(Yes or No). If your answer is "Yes", answer the question: "What value is stored in R0 immediately after the instruction at `x3004` completes execution?" If your answer is "No", put a dash in the column labeled "Value stored in R0 ".

X	Does the program halt?	Value stored in `R0`
000000010		
000000001		
000000000		
111111111		
111111110		