CSCI 120 Introduction to Computation
Homework 4
Solution

Saad Mneimneh
Visiting Professor
Hunter College of CUNY

PART 1: Binary System
(a) Find the decimal representation of each of the following bytes:

01010111 01101000 01100001 01110100
00100000 01100100 01101111 01100101
01110011 00100000 01101001 01110100
00100000 01110011 01100001 01111001
00111111

ANSWER: See below.

(b) Assuming the above message is encoded in ASCII, what does it say? (you may search the Internet for ASCII code).

87	104	97	116	32	100	111	101	115	32	105	116	32	115	97	121	63
W	h	a	t		d	o	e	s		i	t		s	a	y	?

(c) Convert each of the following numbers to binary using either the coin algorithm or the repeated division algorithm:

7 11

16 15

33

ANSWERS:
(a)
7/2=3 and r=1.
3/2=1 and r=1.
1/2=**0** and r=1. (stop)
Therefore, 7 is 111.

(b)
11/2=5 and r=1.
5/2=2 and r=1.
2/2=1 and r=0.

1/2=**0** and r=1. (stop)
Therefore, 11 is 1011.

(c)
16/2=8 and r=0.
8/2=4 and r=0.
4/2=2 and r=0.
2/2=1 and r=0.
1/2=**0** and r=1. (stop)
Therefore, 16 is 10000.

(d)
15/2=7 and r=1.
7/2=3 and r=1.
3/2=1 and r=1.
1/2=**0** and r=1. (strop)
Therefore, 15 is 1111.

(e)
33/2=16 and r=1.
16/2=8 and r=0.
8/2=4 and r=0.
4/2=2 and r=0.
2/2=1 and r=0.
1/2=**0** and r=1. (strop)
Therefore, 33 is 100001.

PART 2: Addition and overflow

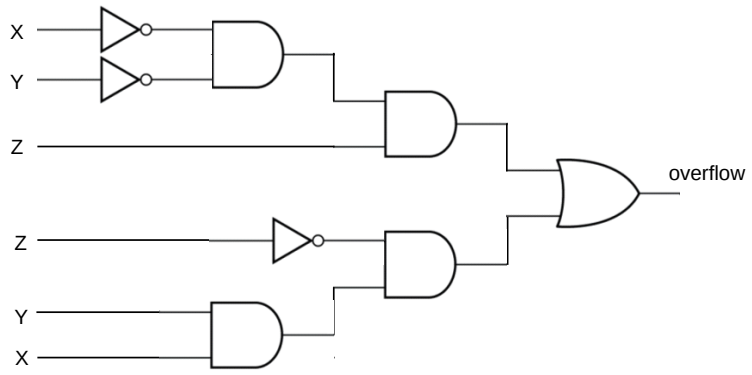
(a) Recall that when adding two numbers, an overflow can occur if the result is outside the range of what can be represented. For example, with a 4 bit two's complement representation, we can only represent the numbers from -8 to 7. Adding 5 and 6 will cause an overflow and, therefore, produce the wrong result (note that overflow does not mean carry). Among the following additions, identify which ones cause an overflow and which ones don't (assume two's complement representation). In both cases, determine what the result is (even if it is wrong).

$$\begin{array}{r} 0010 \\ +\ 0100 \\ \hline 0110 \end{array} \qquad \begin{array}{r} 0100 \\ +\ 0101 \\ \hline 1001 \\ \text{(overflow)} \end{array} \qquad \begin{array}{r} 0011 \\ +\ 1110 \\ \hline 1\ 0001 \end{array} \qquad \begin{array}{r} 0100 \\ +\ 1001 \\ \hline 1101 \end{array}$$

$$\begin{array}{r} 1100 \\ +\ 0110 \\ \hline 1\ 0010 \end{array} \qquad \begin{array}{r} 1010 \\ +\ 0101 \\ \hline 1111 \end{array} \qquad \begin{array}{r} 1100 \\ +\ 1011 \\ \hline 1\ 0111 \\ \text{(overflow)} \end{array} \qquad \begin{array}{r} 1101 \\ +\ 1011 \\ \hline 1\ 1000 \end{array}$$

(b) When adding two numbers as above, let the leftmost bit of the first number be x, the leftmost bit of the second number be y, and the leftmost bit of the result be z. Using x, y, and z design a circuit that outputs 1 when there is an overflow, and 0 otherwise. It will help you first to figure out the truth table for overflow:

x	y	z	$overflow?$
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	1
1	1	1	0



PART 3: Computer Architecture

(a) Draw the architecture of a modern computer showing all the essential components such as: main memory, cache memory, CPU, control unit, ALU, registers, bus, and I/O controllers.

ANSWER: see notes 7 and 8.

(b) For each of the following operations, state which of the above components are involved:

- Instruction fetch: **main memory, cache memory, bus, control unit**
- Arithmetic operation: **control unit, ALU, registers**
- Boolean operation: **control unit, ALU, registers**
- Movement of data between registers: **control unit, registers**

(c) Assume registers R0 and R1 contain the following bit patterns respectively: 00110010 and 00101011. Which memory location is changed by the following instructions? What will that memory location contain (bit pattern) after executing the instructions?

Add R0 R1 R2
Store R2 01101001
Halt

ANSWER: This programs adds the two numbers in registers R0 and R1 and puts the result in register R2. Then it stores the content of R2 into memory address 01101001. Therefore, memory location 105 is going to change after executing the instructions. It will contain the sum of 00110010 and 00101011, which is 01011101.

(d) How does an I/O device communicate with the CPU?

ANSWER: through an I/O controller using the bus and memory mapped IO. There is a set of addresses that the main memory module ignores and only the IO controller responds to.

(e) How do memory and I/O devices resolve conflicts when communicating with the CPU using the same bus?

ANSWER: by responding to a different set of addresses (see above).

(f) What is the part of the instruction that uniquely identifies the instruction to the control unit of the CPU?

ANSWER: the opcode.