hroac2HW2

by Hannah Roach

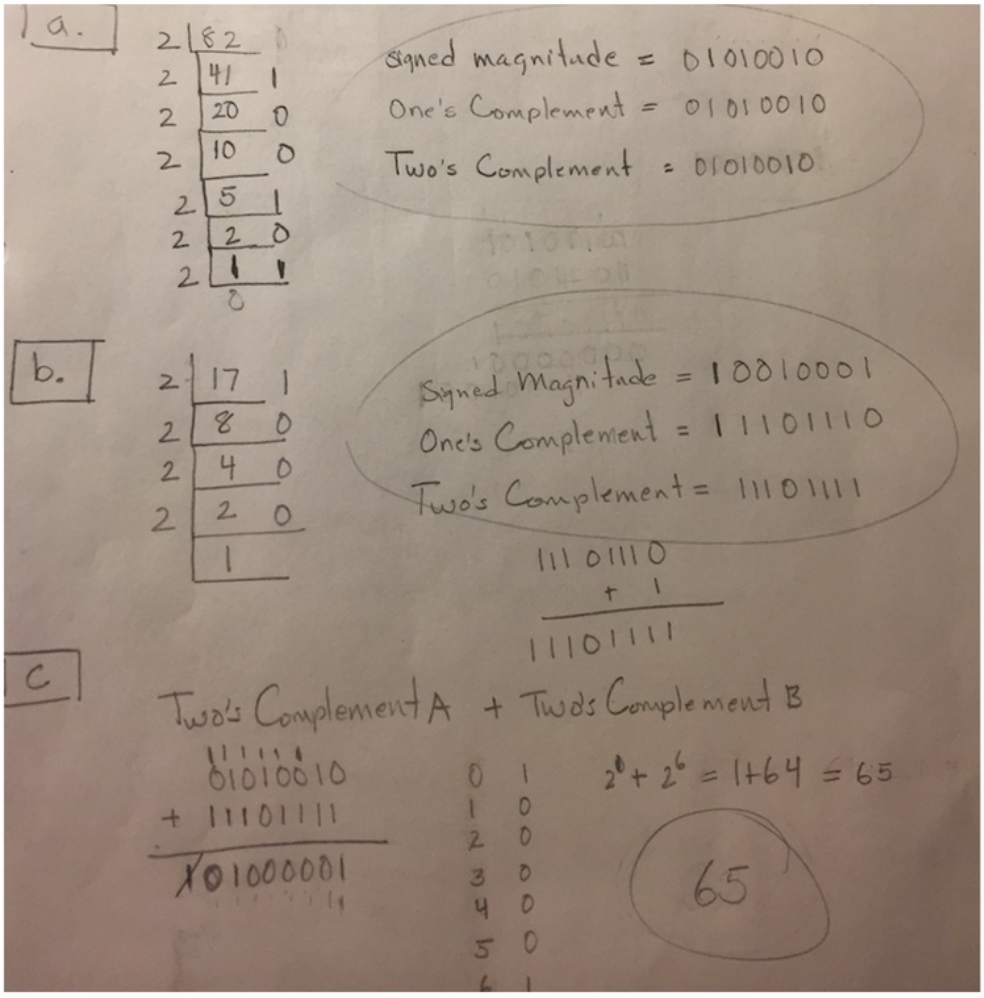
Submission date: 21-Sep-2018 11:49PM (UTC-0500)
Submission ID: 1006349188
File name: hroac2HW2.pdf (2.74M)
Word count: 582
Character count: 2506

CSC 376 Computer Organization
Hannah Roach

Problem 1: (8 points) Show work Represent each of the following decimal numbers in each of the following binary representations using 8-bits:

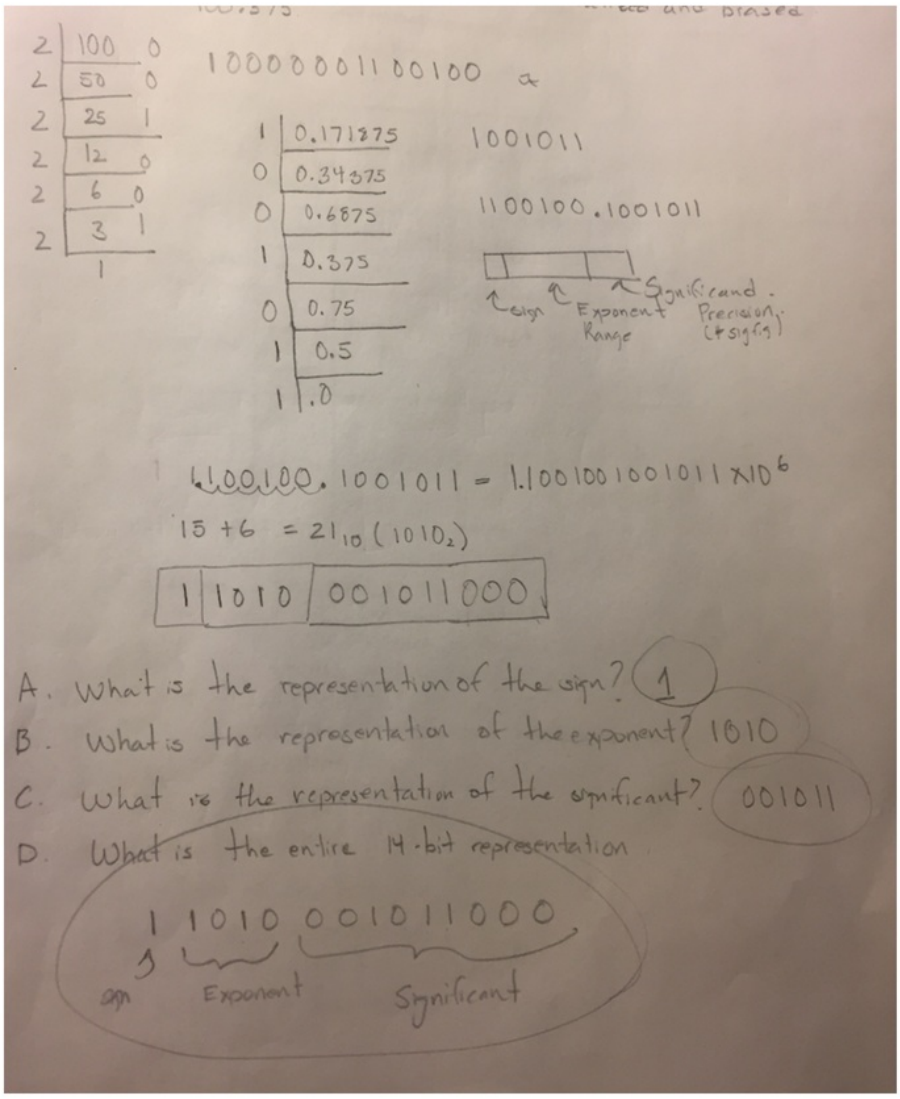
1. Signed magnitude
2. One's complement
3. Two's complement

A. 82 B. -17
C. Add the two's complement of A and B above and then convert the result to decimal.



Problem 2: (7 points)
Show work Using the 14-bit floating point model, what is the Signed Magnitude Representation -normalized & biased for: -100.375

A. What is the representation of the Sign?
B. What is the representation of the Exponent?
C. What is the representation of the Significant?
D. What is the entire 14-bit representation?



Problem 3: (8 points)

Show work Complete the chart below for the following questions for the Pep/8 machine language instructions given. Each Instruction starts from the Original Content (i.e. each instruction is independent and is not based on the previous instruction). You will need to jump ahead to Chapter 6 to determine (D) for one of the instructions.

Original Content Instruction	Do	Register	Mode	A	X	Mem[0A3c]	Mem[2A42]
				10B6	FE25	0A41	0A3F
792A42	Add to register r	Index Register X	Direct		0A25		0A3F
E12A42	Store register r to memory	Accumulator A	Direct	0A3F			0A3F
A90A3C	Bitwise OR to register r	Index Register X	Direct		FE65		
C22A42	Load register r from memory	Accumulator A	Indirect	0A3F			0A3F

Problem 4: (7 points)

(Show your work) Determine the output of the following machine-language program by running it in Pep/8. The left column in each part is the memory address of the first byte on the line:

```
Address    Machine Language (hex)
-------    ----------------------
0000        C1000C   ;Load into A from address
0003        18       ;LDA,d - Load into AC from memory address
0004        51000B   ;NOTA - Invert every bit (1's complement)
0007        51000B   ;STBYTE,d - Store every byte in Pep8
000A        00       ;CHARO,d - Character out from memory
000B        00       ;STOP
000C        FFDA     ;%
```

Problem 5: (10 points)

Write a machine-language program to output your first name on the output device. Write it in a format suitable for the loader and execute it on the PEP/8 simulator. You may only use instructions in Fig 4.6. You shall submit

A. A written algorithm

B.
```
#include <iostream>
using namespace std;

int main(void)
{
    printf("Hannah");
```

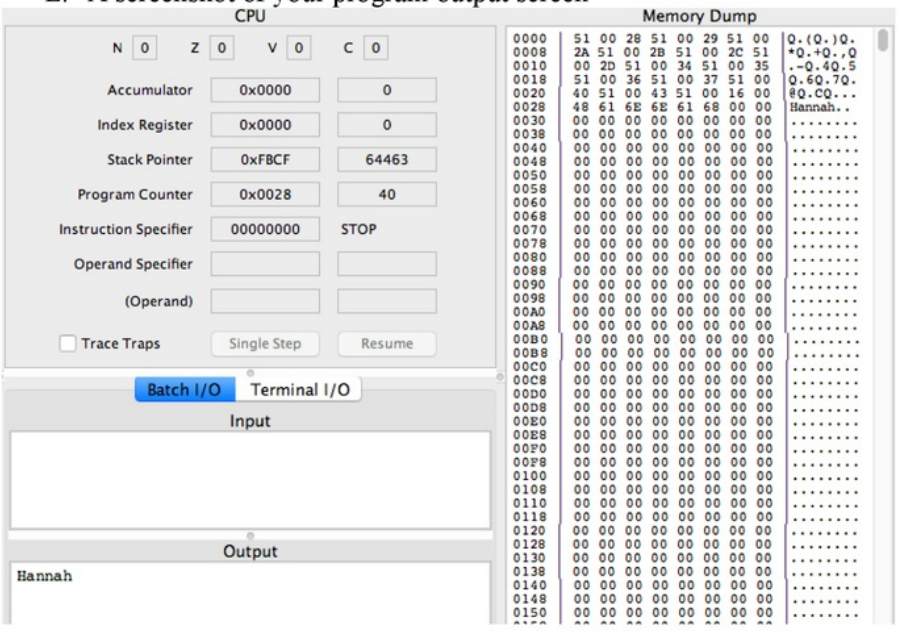
C. B. Your commented program similar to the Machine Language (hex) of Fig 4.35

```
 Address   Machine Language (hex)
-------    ----------------------
  0000     510028 ;Character output
  0003     510029 ;Character output
  0006     51002a ;Character output
  0009     51002b ;Character output
  000C     51002c ;Character output
  000F     51002d ;Character output
  0010     510034 ;Character output
  0011     510035 ;Character output
  0012     510036 ;Character output
  0013     510037 ;Character output
  0014     510040 ;Character output
  0015     510043 ;Character output
  0016     510016 ;Character output
  0017     00     ;Stop
  0018     48     ;ASCII H character
  0019     61     ;ASCII a character
  0020     6e     ;ASCII n character
  0021     6e     ;ASCII n character
  0022     61     ;ASCII a character
  0023     68     ;ASCII h character
```

D. The hexadecimal program suitable for the Pep/8 loader. (Cut & paste the text into your document so I can check your code)

51 0 28 51 00 29 51 00 2a 51 00 2b 51 00 2c 51 00 2d 51 00 34 51 00 35 51 00 36 51 00 37 51 00 40 51 00 43 51 00 16 00 48 61 6e 6e 61 68 zz

E. A screenshot of your program output screen



Problem 6: (10 points)

Write a machine-language program to add the three numbers 6, -7 and 4 and output the sum on the output device. Note-the inputs can be hard coded in data storage at the end of your code. The accumulator holds 16 bits so use 16 bit Two's Complement for the -7 and do not use the subtract instruction. Write it in a format suitable for the loader and execute it on the PEP/8 simulator. You may only use instructions in Fig 4.6. (Use the Fig 4.6 revised found with the HW2 assignment) You shall submit:

A. A written algorithm

```
#include <iostream>
using namespace std;

int main(void)
{
    int sum = 6+4+(-7);
    cout<<sum<<endl;
}
```

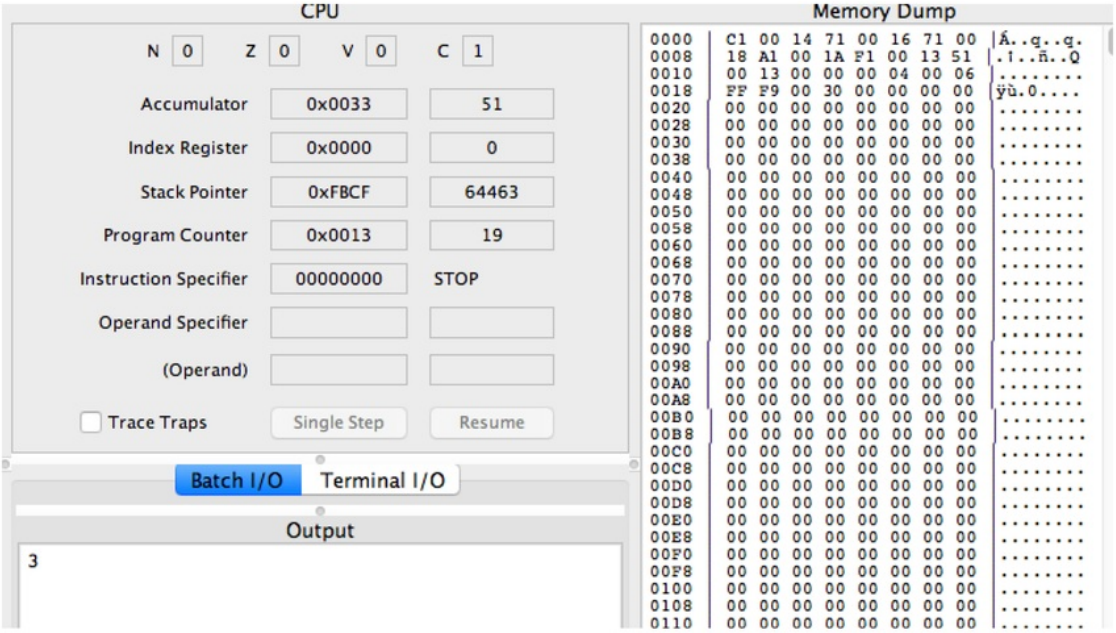
B. Your commented program similar to the Machine Language (hex) of Fig 4.35

```
 Address   Machine Language (hex)
-------    ----------------------
  0000     C10014   ;A := first number
  0003     710016   ;Add the two numbers
  0006     710018   ;Add the two numbers
  0009     A1001a   ;Convert sum to character
  000C     F10013   ;Store the character
  000F     510013   ;Output the character
  0012     00       ;Stop
  0013     00       ;Character to output
  0014     0004     ;Decimal 4
  0016     0006     ;Decimal 6
  0018     FFF9     ;Decimal -7
  0020     0030     ;Mask for ACSCII char
```

C. The hexadecimal program suitable for the Pep/8 loader. (Cut & paste the text into your document so I can check your code)

C1 00 14 71 00 16 71 00 18 A1 00 1a F1 00 13 51 00 13 00 00 00 04 00 06 FF F9 00 30 zz

D. A screenshot of your program output screen



E. Explain the range of numbers that this will work correctly

This code will work when the sum is between 0 and 10.

2

hroac2HW2

GRADEMARK REPORT

FINAL GRADE

49/50

GENERAL COMMENTS

Instructor

PAGE 1

PAGE 2

PAGE 3

Comment 1

0864

PAGE 4

PAGE 5

PAGE 6

Comment 2

0-9

RUBRIC: SP2015-HW2-POINTS 49 / 50

1A 3 / 3

FULL CREDIT
(3)

MINUS 1
(2)

MINUS 2
(1)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

1B 3 / 3

FULL CREDIT
(3)

MINUS 1
(2)

MINUS 2
(1)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

1C 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

2A 1 / 1

FULL CREDIT
(1)

MINUS 1
(0)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

2B 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

2C 2 / 2

FULL CREDIT

(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

2D 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

3A 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

3B 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

3C 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

3D 1 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2

(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

4A 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

4B 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

4C 1 / 1

FULL CREDIT
(1)

MINUS 1
(0)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

4D 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

5A 1 / 1

FULL CREDIT
(1)

MINUS 1
(0)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

5B 5 / 5

FULL CREDIT
(5)

MINUS 1
(4)

MINUS 2
(3)

MINUS 3
(2)

MINUS 4
(1)

NO CREDIT
(0)

5C 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

5D 2 / 2

FULL CREDIT
(2)

MINUS 1

(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

6A 1 / 1

FULL CREDIT
(1)

MINUS 1
(0)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

6B 4 / 4

FULL CREDIT
(4)

MINUS 1
(3)

MINUS 2
(2)

MINUS 3
(1)

MINUS 4
(0)

NO CREDIT
(0)

6C 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

6D 2 / 2

FULL CREDIT
(2)

MINUS 1
(1)

MINUS 2
(0)

MINUS 3
(0)

MINUS 4
(0)

NO CREDIT
(0)

6E 1 / 1

FULL CREDIT
(1)

MINUS 1
(0)

MINUS 2
(0)

MINUS 3

(0)

MINUS 4
(0)

NO CREDIT
(0)