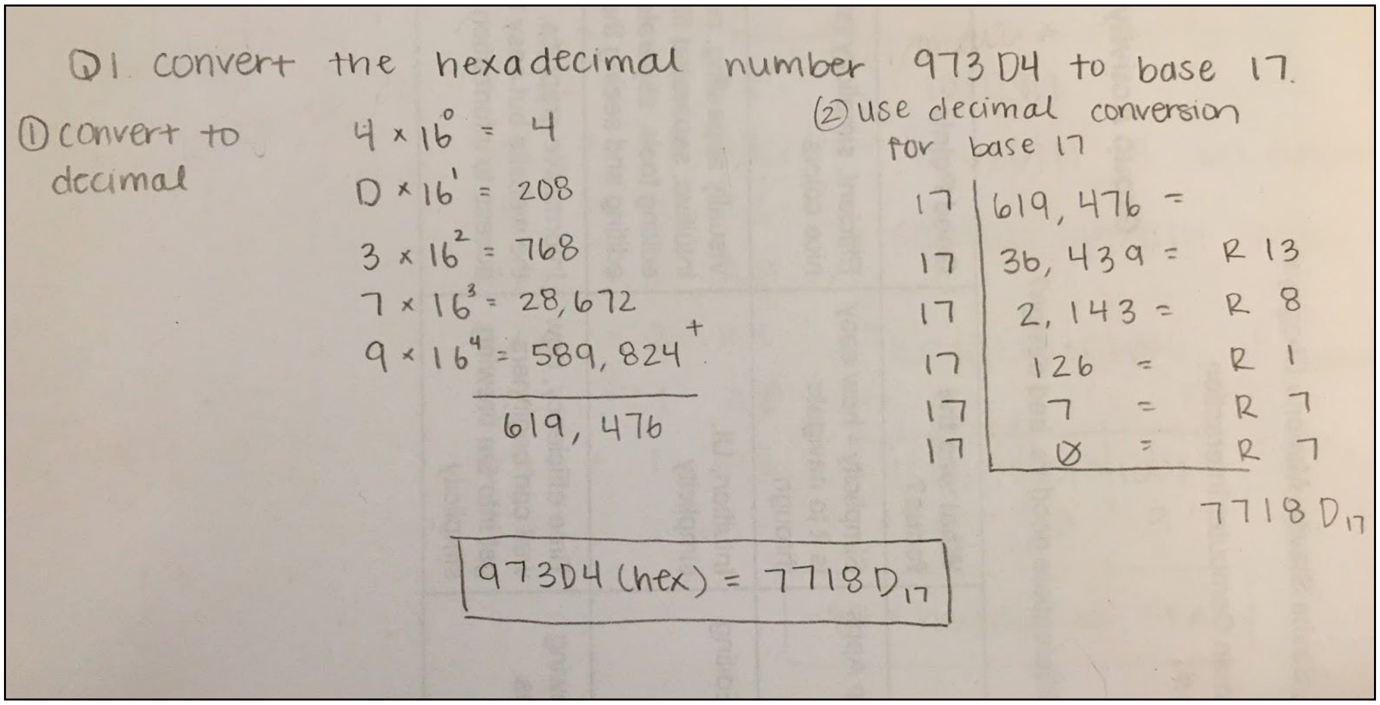
Dwina Solihin
April 3, 2017
CSS 422A
Professor Folsom

Q1.



Q1. convert the hexadecimal number 973D4 to base 17.

① convert to decimal

$4 \times 16^0 = 4$
$D \times 16^1 = 208$
$3 \times 16^2 = 768$
$7 \times 16^3 = 28,672$
$9 \times 16^4 = 589,824$ +

619,476

② use decimal conversion for base 17

17	619,476 =	
17	36,439 =	R 13
17	2,143 =	R 8
17	126 =	R 1
17	7 =	R 7
17	Ø =	R 7

$7718D_{17}$

$\boxed{973D4 \text{ (hex)} = 7718D_{17}}$

Q2.

a. MOVE.B $A000, A3
   i. There is an error because the op-code MOVE.B is used to copy data to a data register. As seen in the instruction above, it is copying data to an address register. To fix this line, you can change A3 to D3 or change the op-code to MOVEA.

b. ADD.B #$1000, D2
   i. There is an error because the op-code says that you want to add a byte to the data in data register D2. However, the hexadecimal value of 1000 exceeds the byte amount it is trying to add. It can be fixed by changing #$1000 to #$10.

c. MOVEA.W $1234,D0
   i. There is an error because the op-code MOVEA is used to copy an address to a new address register destination. In the instruction above, you are copying the data into a data register instead of an address register. To fix this line, you change change D0 to A0.

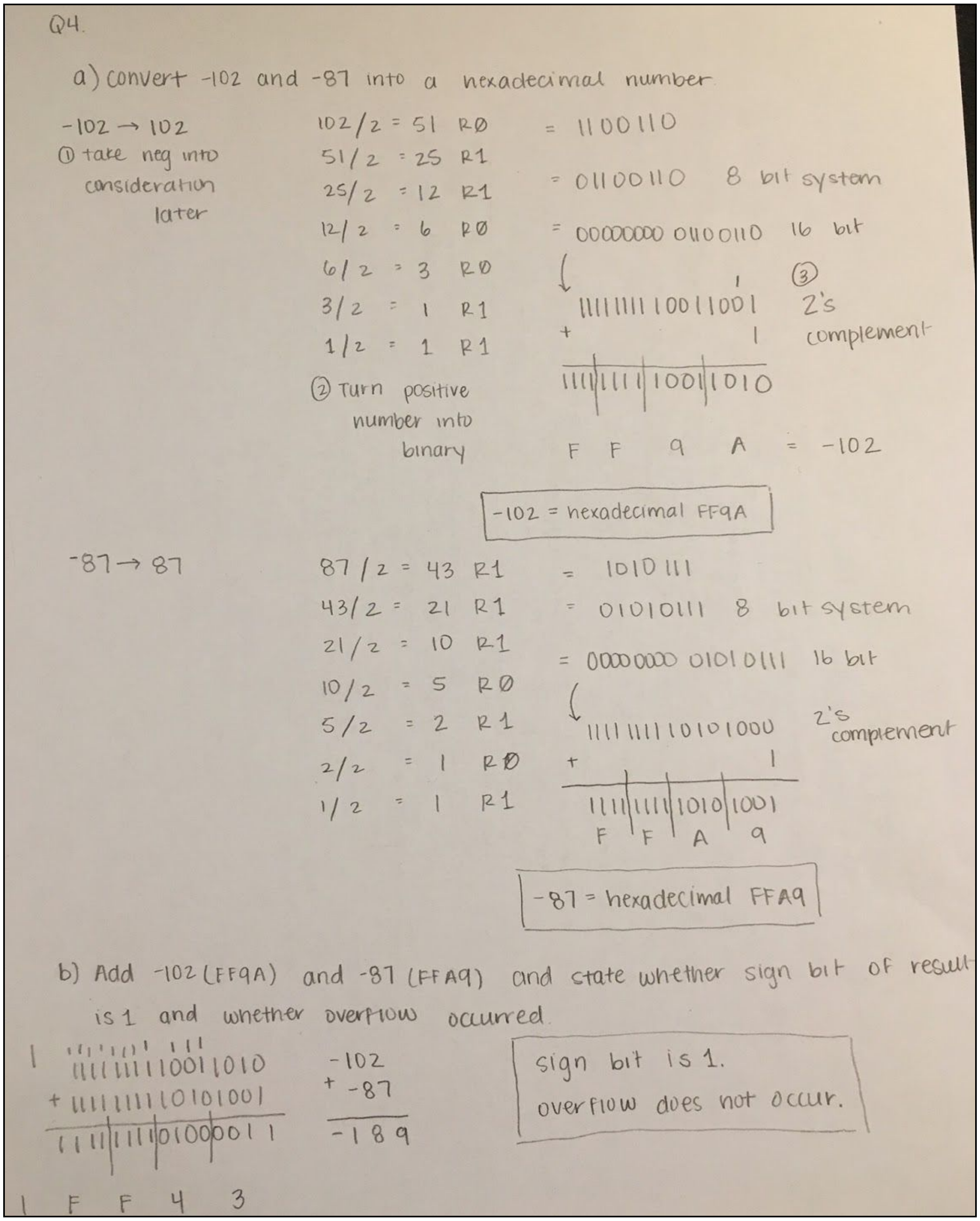
d. ANDI.B #23, #$100
   i. There is an error in the op-code ANDI.B because this op-code is used in order to perform an AND operation of the immediate data (23) with the destination

operand and stores the result in the destination location. In the instruction above, it is not a destination operand that is being called but a hexadecimal value. To fix this line, you would have to change #$100 to a data register.

Q3. What is the **WORD VALUE** (not byte, or longword) of the data in memory location $4000, when the program is just about to loop back to the beginning and start over again?

a. The word value of the data in memory location $4000 when the program is just about to loop back to the beginning and start over again is:
    i. 45 15
b. I was able to reach an answer because I traced the code segment with debugging tools. When going through the the code segment, I stepped through each line one-by-one keeping track of what the data in memory location $4000 for each line. Once I got to the last line before the loop started over from the start, I saw that the data changed to 45 15 right when the loop started over at the beginning.

Q4.



Q4.

a) convert -102 and -87 into a hexadecimal number

-102 → 102

① take neg into consideration later

102/2 = 51 R0	= 1100110			
51/2 = 25 R1				
25/2 = 12 R1	= 01100110	8 bit system		
12/2 = 6 R0	= 00000000 01100110	16 bit		
6/2 = 3 R0	↓	③		
3/2 = 1 R1	11111111 10011001	2's complement		
1/2 = 1 R1	+ 1			

② Turn positive number into binary

1111|1111|1001|1010

F F 9 A = -102

$\boxed{-102 = \text{hexadecimal FF9A}}$

-87 → 87

87/2 = 43 R1	= 1010111	
43/2 = 21 R1	= 01010111	8 bit system
21/2 = 10 R1	= 00000000 01010111	16 bit
10/2 = 5 R0	↓	
5/2 = 2 R1	11111111 10101000	2's complement
2/2 = 1 R0	+ 1	
1/2 = 1 R1	1111|1111|1010|1001	

F F A 9

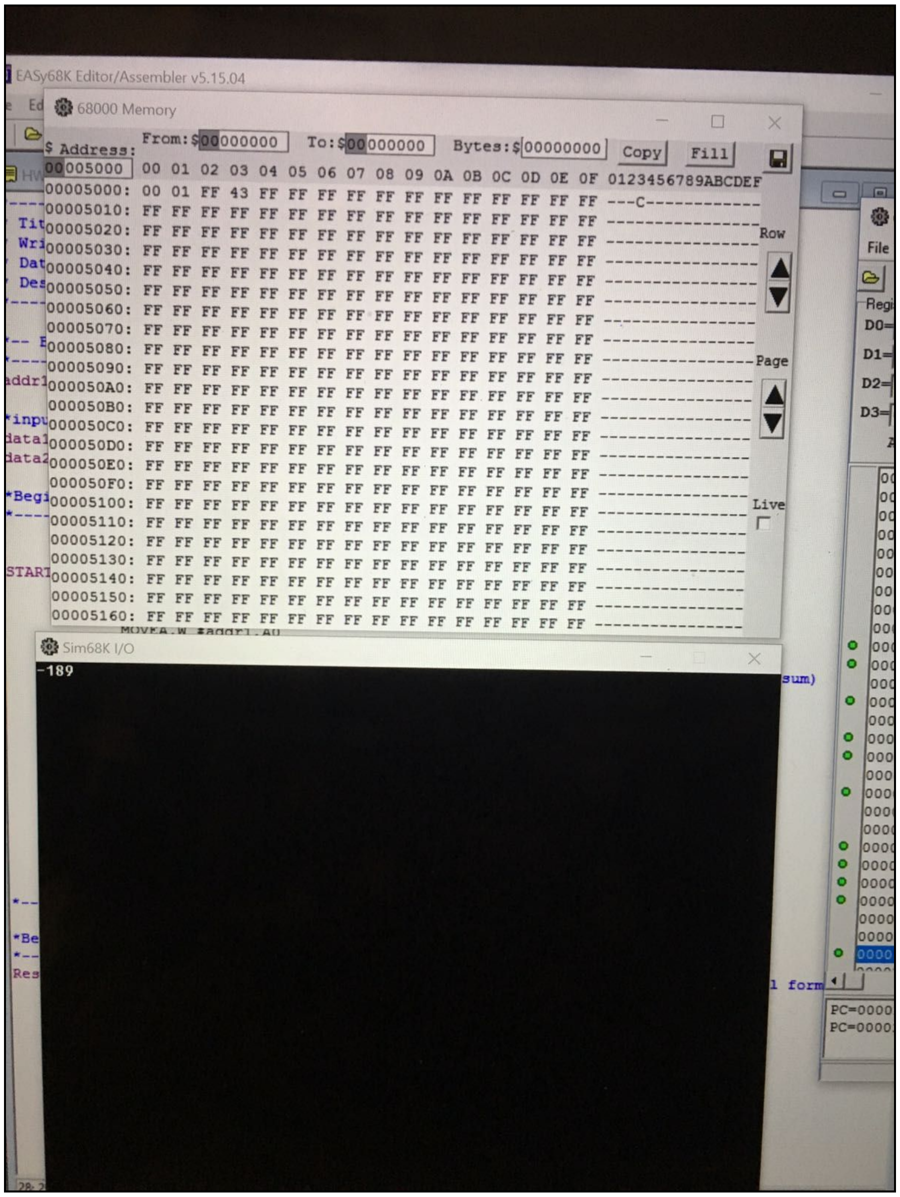
$\boxed{-87 = \text{hexadecimal FFA9}}$

b) Add -102 (FF9A) and -87 (FFA9) and state whether sign bit of result is 1 and whether overflow occurred.

```
1  11111111 10011010      -102
 + 11111111 10101001     + -87
   ----------------      -----
   1111|1111|0100|0011    -189
1   F    F    4    3
```

$\boxed{\text{sign bit is 1. overflow does not occur.}}$

C. Screen shot of memory address at address $5000



```
$ Address:  From:$00000000   To:$00000000   Bytes:$00000000
00005000   00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 0123456789ABCDEF
00005000: 00 01 FF 43 FF FF FF FF FF FF FF FF FF FF FF FF ---C------------
00005010: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005020: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005030: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005040: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005050: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005060: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005070: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005080: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005090: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
000050A0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
000050B0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
000050C0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
000050D0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
000050E0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
000050F0: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005100: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005110: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005120: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005130: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005140: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005150: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00005160: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
```

```
-189
```