Homework 4 - CSS422 - Vu Dinh

Question 1

1. MOVE.W D1, $0000A000

MOVE's first 2 bits are 00. Data size = W → 11.
Destination: (xxx).L → Mode: 111, Reg: 001
Source: D1 → Mode: 000, Reg: 001
Followed by additional data words

00	11	001	111	000	001

0000000000000000

1010000000000000

2. MOVE.B $42A3, D1

MOVE's first 2 bits are 00, data size = B → 01
Destination: D1 → Mode: 000, Reg: 001
Source: (xxx).W → Mode: 111, Reg: 000
Followed by one additional data word

00	01	001	000	111	000

0100001010100011

3. ADD.L D7, D0

ADD's first 4 bits are 1101, Opmode is 010 because:
--- data type: L
--- form: <ea> + D0 → D0
Source <ea>: D7 → mode: 000, reg: 111

1101	000	010	000	111

Question 2

1. MOVE.B $A000, A3

Moving data into Address Registers can only be done via MOVEA commands.

2. ADD.B #$1000, D2

#$1000 has 4 hex bits, it does not fit as a byte.

3. MOVEA.W $1234, D0

MOVEA is used for moving data into Address Registers. If normal data is needed to be moved, use MOVE instead

4. ANDI.B #23, #$100

There seem to be multiple things wrong with this operation:

- ~~#$100 has 3 hex bits, it does not fit as a byte.~~ Actually this is not a problem because the specified data size (Byte) only needs to match the first operand, which it does.
- ANDI needs an <ea> as destination operand (so that the result can be stored)

Question 3

1. Represent -99 and -39

99 in binary = %01100011
-99 in binary = %10011101
-99 in hex = $9D

39 in binary = %00100111
-39 in binary = %11011001
-39 in hex = $D9

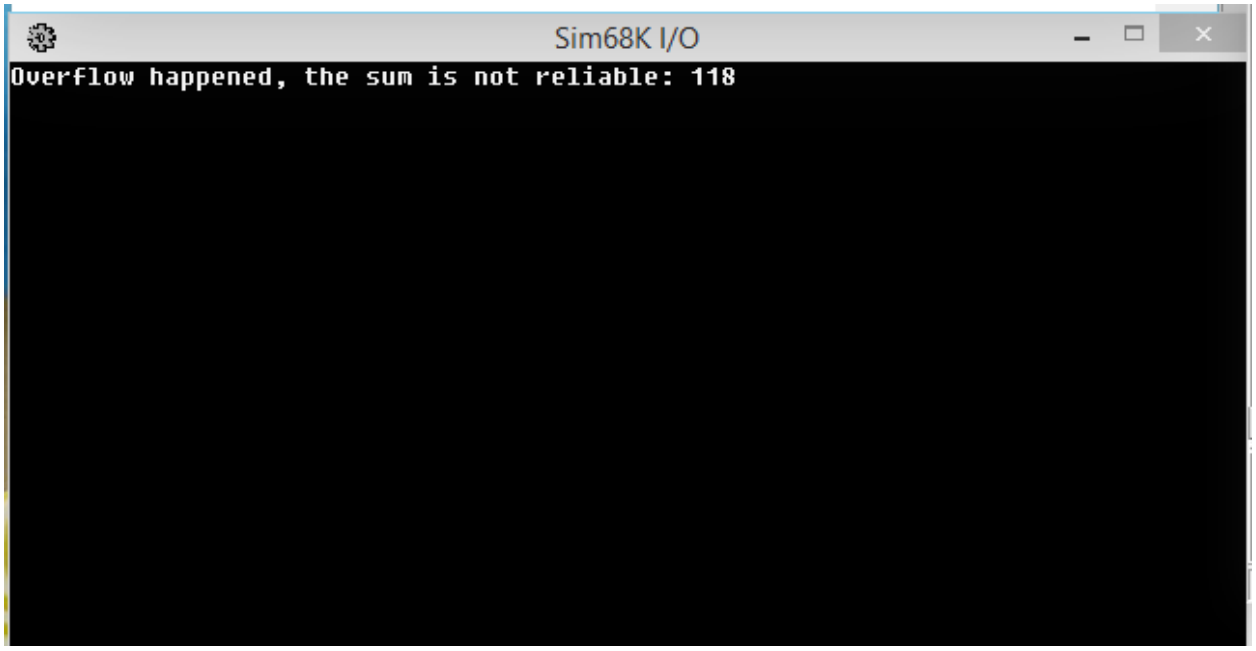
2. Add

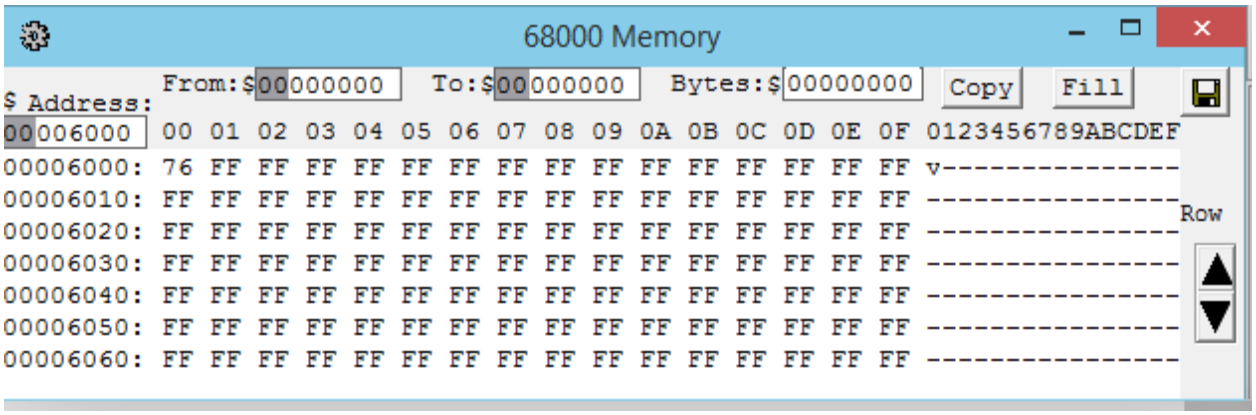
$9D + $D9 = $76 (truncated the carry bit)

a. The sign bit of the sum is 0 (which doesn't make sense when adding 2 negative numbers)
b. An overflow definitely occurred.

3. Assembly program

```
*-----------------------------------------------------------
* Title      : Homework 4
* Written by : Vu Dinh
* Date       : October 31 2014
* Description: I'm late I'm late I'm lateeeee
*-----------------------------------------------------------
    ORG    $1000
START:                  ; first instruction of program

* Put program code here

    MOVE.B #$9D, D2     * test conversion
    MOVE.B #$D9, $6000  * specified address for result
    ADD.B   D2, $6000   * Addition

    BVS     OVERFLOW
    BVC     NOOVERFLOW

OVERFLOW    LEA BADMSG, A1
            MOVE.B  #14, D0
            TRAP    #15
            BRA     PRINTRESULT


NOOVERFLOW  LEA GOODMSG, A1
            MOVE.B  #14, D0
            TRAP    #15
            BRA     PRINTRESULT

PRINTRESULT     MOVE.B  $6000, D1   * prime for output
                MOVE.B  #3, D0      * load trap task #3
                TRAP    #15


    SIMHALT             ; halt simulator

* Put variables and constants here

GOODMSG         DC.B    'There were no overflows, the sum is: ',0
BADMSG          DC.B    'Overflow happened, the sum is not reliable: ',0

    END    START        ; last line of source
```


Sim68K I/O
Overflow happened, the sum is not reliable: 118


68000 Memory
From:$00000000
To:$00000000
Bytes:$00000000
Copy
Fill
$ Address:
00006000 00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F 0123456789ABCDEF
00006000: 76 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF v---------------
00006010: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00006020: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00006030: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00006040: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00006050: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
00006060: FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF ----------------
Row

Question 4

I'm just not going to show the multiplication steps because it's quicker to do it in my head and write down 0 or 1

1. Convert into IEEE single-precision format

   a. -69/32 = -2.15625

-2.15625 in binary is -10.00101 = -1.000101 * 2^1

1	10000000	00010100000000000000000

in hex: C00A0000

   b. 13.625

13.625 in binary is 1101.101 = 1.101101 * 2^3

0	10000010	10110100000000000000000

in hex: 415A0000

2. Convert into decimal

   a. 42E48000

0	10000101	11001001000000000000000

=> 42E48000 = + 1.11001001 * 2^(133-127) = 1.11001001 * 2^6 = 1110010.01 = 114.25

   b. C6F00040

1	10001101	11100000000000001000000

=> C6F00040 = - 1.11100000000000001 * 2^14 = - 111100000000000.001 = -30720.125