**Q1. Convert the hexadecimal number 973D4 to base 11. You must show your work. (2 points)**

|  |  |
| --- | --- |
| 973D4 = 619476 |  |
| 619476 |  |
| 56316 | 0 |
| 5119 | 7 |
| 465 | 4 |
| 42 | 3 |
| 3 | 9 |
| 0 | 3 |

973D416 = 39347011

**Q2. Error Finding (4 points)**

*Each of the following 68K assembly language instructions will cause an assembler error. Examine each instruction and explain why the assembler would flag it as an error*

1) MOVE.B $A000, A3 \* Destination is address direct, MOVE destination

2) ADD.B #$1000, D2 \* $1000 hex exceeds the size Byte, .B can only accept $00-$FF, 0-256

3) MOVEA.W $1234, D0 \* Destination is Data , MOVEA requires address direct

4) ANDI.B #23, #$100 \*Invalid Addressing Mode, can’t move result into the hexadecimal 100

**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? (3 Points)**

The Word value in address $4000 is 4515. Found this by using 68k memory viewer as well as tracing the program logic (F7 – Trace Into) to find this. To check that 4515 is correct I step by stepped the program seeing AND.W D3, (A1) was comparing $4F17 and $5555 which when I wrote it out was 0100 0101 0001 0101 or 4515.

**Q4. Two’s complement (6 points)**

1. **Convert the decimal numbers -102 and -87 into hexadecimal number**

-102 is FF9A. -87 is FFA9.

102 = 0110 0110 -> Two’s complement = 1001 1010 || FF9A (8 leading 1’s for 16bit)

87 = 0101 0111 -> Two’s complement = 1010 1001 || FFA9

1. **Add two numbers of the previous question as hexadecimal, and state,**

1111 1111 1001 1010

1111 1111 1010 1001

1 1111 1111 0100 0011 -> FF43

* 1. **Whether the sign bit of the result is 1?** Yes it is
  2. **Whether an overflow occurred.** Yes, it did

1. Source Code

|  |
| --- |
| data1 EQU $FF9A \*-102  data2 EQU $FFA9 \*-87  addr1 EQU $5000  ORG $1000  START: ; first instruction of program  MOVE.W #data1, D1 \*Load data1(FF9A) into D1  MOVE.W #data2, D2 \*Load data2(FFA9) into D2  ADD.W D2, D1 \*ADD D1 and D2, put result in D2  BCS OVERFLOWERR \*If carry flag is set, branch to OVERFLOWERR  DISPLAY  MOVEA.W #addr1, A1 \*Load addr1(5000) into A1  MOVE.W D1, (A1) \*Move result into A0 ($5000)  MOVE.B #3, D0 \*Task 3 for Trap 15 (displays as decimal)  TRAP #15 \*Display whatever is in A1    SIMHALT ; halt simulator  OVERFLOWERR \*OVER FLOW ERROR FOUND  LEA overflmsg, A1 \*Give Address A1 the Error msg  MOVE.B #14, D0 \*Task 14 for TRAP 15  TRAP #15 \*Display Error Msg  JMP DISPLAY \*Jump back up and display result  CR EQU $0D ;ASCII code for Carriage Return  LF EQU $0A ;ASCII code for Line Feed  overflmsg DC.B 'Overflow Occured',CR,LF,0  \* Put variables and constants here  END START ; last line of source |
|  |
|  |