BACHELOR OF INFORMATION TECHNOLOGY

ALGONQUIN COLLEGE **CARLETON UNIVERSITY**

ALGONQUIN COLLEGE - SCHOOL OF ADVANCED TECHNOLOGY

COMPUTER STUDIES DEPARTMENT

MIDTERM EXAMINATION I

COURSE: NET1004 - Assembly & Machine Language

PROFESSOR: R. Dyer

DATE: Thursday, 17 February 2009, 8:00am

PLACE: WT327

****

INSTRUCTIONS

1. Permitted Resources: "Assembly Language Programming Course Notes" as published by Algonquin College. (Text may **not** be shared among two or more students during this exam), NO CALCULATORS OR LOSE PAPERS other than above notes.

2. Write your name in the space provided above. Do not write your name on any other page.

3. The time allotted for this examination is **1.75 hours**.

4. The marks for each question are shown below . The total is 65 marks.

5. **Write your answers on this examination paper**. Show your work and intermediate results on the examination paper.

6. This exam has 8 questions.

****

— 70 5

Permitted Resources: "Assembly Language Programming Course Notes" as published by Algonquin College (Text may **not** be shared among two or more students during this test).
Answer all questions on these pages.

Portions of Assembly Listing (to be used for questions 1 and 2):

```
  42            /************ VARIABLE DATA ***********************************/
  43 0111 00                          .DATA
  44 0000 0010      wv_first:         .WORD       16
  45 0002 0020      wv_second:        .WORD       32
  46 0004 0030      wv_third:         .WORD       48
```

```
  80 0014 6500 0030   chkErro:    BCS    inDataErr
  81 0018 33C0 0000               MOVE.W   D0, wv_first
  81      0112
  82
  83
  84 001e 41FA 008A   ask2:       LEA    ask2, A0          | Prompt, get number in hex
  85 0022 4EB9 0000               JSR    PrintMsg          |      and  save it
  85      0000
  86 0028 123C 0008               MOVE.B   #8, D1          | Max # of hex digits to get
  87 002c 4EB9 0000               JSR    InHex
  87      0000
  88 0032 6500 0012               BCS    inDataErr
  89 0036 33C0 0000               MOVE.W D0, wv_second
  89      0114
  90 003c 2239 0000               MOVE.L lv_value, D1
  90      0118
```

1. Based on the above partial Assembly listing, [3 marks]

   a. How long is the machine code translation of the instruction: **`MOVE.W d0,wv_second`**

   6

   b. Suppose it were necessary to change statement: **`MOVE.W D0, wv_second`** to **`MOVE.L D0, lv_value`**
   what line number would you go to in your text editor to make this change?

   89

   c. What is the assembly address (within its subsection) for the instruction:
   `MOVE.L lv_value, d1`

   003c

2. Given that the linker relocates the TEXT section to address 1000 (hex) and the DATA section to address 2000 (hex), what address will be used at execution time for: [6 marks]

a. `wv_second`

2002

b. `ask2`

1018

c. `lv_value` (note that you will need to use the machine code version of `MOVE.L lv_value, D1` to determine this)

2006

3. Given that the following pattern is the machine instruction for either an **ADD**, a **SUB**, or a **MOVE** instruction; use the "Programmer's Reference Manual" section of your "Assembly Language Programming Course Notes" to fully decode this instruction: [4 marks]

207C

Move.L #, A0

0010

-4

20.5

4. Code an M68000 Assembler program (omit all comments) which would **prompt for and get two single 4 digit hex numbers, add them together and then display the ASCII codes for that number** (as 6 hexadecimal digits). [10 marks]

List
Test.o

Data

ASCII

print

pseudo-code:

print msg: " enter 2 4 digit hex numbers.

Num1, D1

Num2, D2

add.l D1, D2

JSR

Out hex. ASCII

more pseudo code:

int Num1, Num2

Cout << enter 2 4 digit hex numbers << endl;

Cin >> Num1.

cin >> Num2 hex

Num1 + Num2 = Num3

Cout << Num3 << endl;

6.5

-8

5. Given the BDB emulator dump of the next instruction: [7 marks]

```
D 00000000 00000007 0000[illegible] 00000000  00000000 00000000 00000000 00000000
A 000010A0 00000000 00000000 00000000  00000000 00000000 00000000
USP=00007700  SSP=00007F00        SR=2700 Tr=0 S=1 Int=7 X=0 N=0 Z=0 V=0 C=0
00001030  0441 0009   subi.w      #$0009,d1
```

Assuming this next instruction were executed:

a. What would be the address in the PC (Program Counter)?

0000 1034

b. Which Data register would change and what would be the value (as a longword) in this register?

D1 : FFFE

c. Which of the flag(s) would change?

N, C

6. Fill in the blanks. Show complete contents of data registers. **(all 32 bits)** Use Hexadecimal numbers. [15 marks]

a)
```
move.l #0x0000F111,d0
addi.w #0x10EF,d0
```

d0 0000 0200.

N 0 V 0 C 1.

```
 1 1
F111
10EF
0200
```

b)
```
move.w      #0x800A,0x4000
move.l #0x001180A6,d3
add.w       x4000,d3
```

d3 0000 200B0.

N 0 V 0 C 1.

```
  1
 800A
180A6
200B0
```

c)
```
movea.l     #0x3000,a6
move.w      #0x1010,(a6)
addi.w #0xFA,(a6)+
```

a6 120A.

N 0 V 0 C 1.

```
  1
1010
  FA
120A
```

7. Consider the following instruction Sequence loaded and executed from within BDB: (15 Points)

```
                .DATA
Array:          .SPACE    5       | 5 bytes

                .TEXT 0
                .GLOBAL   MIDIW11
MIDIW11:
                CLR       D0
                CLR       D1
                LEA       Array, A0
                MOVE.L    #0x04010302, (A0)
                MOVE.W    Array, D1
                MOVE.B    D1, Array+2
                ADDA.L    #3, A0
                SUBQ.B    #2, D1
                MOVE.B    d1, (A0)
                MOVE.L    #0xA, D2
                MOVE.B    #0xAA, (-11, A0, D2.l)
                ADD.B     D1, Array+4
                SUB.W     D1, Array+2
                RTS
```

Answer the following questions assuming each instruction has bee executed up to the RTS statement.

A) What **Registers** have changed and what are their **contents in hex (as longwords)**?

D0 02 -7
D1
A0

B) Suppose the memory allocated to **Array** in line 2 starts at location **0x2000**. Complete the diagram below showing what is in each memory location in the chart **AFTER** all of the instructions have been executed.

12.5

Address	Contents
0x2000	04
0x2001	03 -1 01
0x2002	A -5 A6
0x2003	E -1 00
0x2004	B -1 FF

-10.5

8. Assuming Num1 is stored in d5 and Num2 is stored in d6, write assembly language code corresponding to the following pseudo-code. Num1 and Num2 are 32 bit signed integers. [5 Marks]

```
DO
  Num1 <- Num1 - 10
  Num2 <- Num2 + Num1
WHILE (Num1 Not= 0)
```

```
Sub.Q #10, Q5
Cdvq.L d5, d6
Cmd #d5
CmP -
BNE. h
```

-2