CMPUT 229 – FINAL EXAM B1

April 19, 2000

Instructor: M. Polak

The time for this test is 120 minutes. The exam consists of 12 questions, so allocate your time carefully.
Only MC68000 Reference Cards are allowed – no books, notes, or calculators.

Name: _____

I.D. #: _____

Question #1 (4 points)

Below is an incomplete RTL (register transfer language) instruction sequence that implements the following instruction:

ADD.W (A1), D1

This sequence includes both instruction fetch and execution. Besides the PC and address/data registers, assume the machine incorporates MBR, MAR, and IR registers in its architecture. Complete the missing RTL steps (1, 3, 4, 7) in this sequence.

Instruction Fetch:

1.

2. MBR ← M[MAR]

3.

4.

5. DECODE

Operand Fetch :

6. MAR ← A1

7.

Instruction Execution:

8. D1 ← D1 + MBR

Question #2 (2 points)

What is the difference between JMP $1000 and BRA $1000 in terms of execution.

Question #3 (2 points)

Give two addressing modes that cannot be used to specify a destination operand.

Question #4 (6 points)

CCR = 08
D1 = 00008C4F D2 = AFBF4C4F D4 = 00000000
A1 = 008003D4 A2 = 00002002

>MD 2000
002000 00 01 00 02 00 03 00 00 00 00 00 00 00 00 00 00

Assume the above initial conditions before executing the following sequence of 3 instructions.

```
1000   ASL.W  #1, D1
1002   LEA  -1(A2,D4), A1
1006   MOVE.B  2(A2), D2
```

For each instruction in the sequence, show the contents of all affected registers and memory locations after executing the instruction. Remember to include both the PC and CCR in your list of affected registers.
Note: In ASL instruction, the V-bit is set if the most-significant bit is changed at any time during the shift operation and cleared otherwise.

Question #5 (2 points)

If the CCR bits are XNZVC = 01010, specify whether or not the following two instructions will branch or not:

a) BLT NEXT

b) BGT NEXT

Question #6 (10 points total. a=4 points, b=3 points, c=3 points)

Consider the following sections of a C subroutine:

```
void bubble( short array[], short len)
{
 short tmp;
        if (len < 2) return;
        :
        :
        tmp = array[0];
        array[0] = array[1];
        array[1] = tmp;
        :
        :
        bubble( &array[1], len-1 );
        :
        :
}
```

All arguments are passed on the stack and all reference to *tmp* are to be made with respect to a local stack frame.

The following assembly code segment is used to implement the beginning of this subroutine:

```
BUBBLE    LINK A6, #-2
          MOVEM.L D0-D4/A0-A1, -(SP)
                :
                :
```

The following code segment is used to call this subroutine in the main routine:

```
          MOVE.W  #5, -(SP)
          PEA.L  ARRAY
          BSR BUBBLE
          ADDA.L  #6, SP
```

<<< Question #6 continues on the next page >>>

a) Assuming that the variable *len* is not modified anywhere in the function, what will be the maximum number of bytes taken up by the stack when the specified call from the main routine is made to this assembly subroutine. For partial marks, show your work.

b) Implement in assembly the following instructions as they appear in the *bubble()* function, assuming the subroutine call and initialization given on the last page:

```
tmp = array[0];
array[0] = array[1];
array[1] = tmp;
```

c) Implement in assembly language the following instruction as it appears in the *bubble()* function:

```
bubble( &array[1], len-1 );
```

Note: Remember to properly push and pop the needed parameters to and from the stack.

Question #7 (3 points)

Express the number -10.5_{10} in 32-bit IEEE floating point format.

Question #8 (4 points total. a=1 point, b=2 points, c=1 point)

Consider the following loop that will copy a certain number of characters of one string into another.

```
        MOVE.W  #N, D0
NEXT    MOVE.B  (A0)+, (A1)+
        DBF  D0, NEXT
```

a) How many bytes of storage will the above code require (not including the strings to be copied)?

b) How many clock cycles will execution of the above code take? Write the answer in terms of N.

c) Assuming that the CPU is clocked at 8 MHz, what amount of time (in seconds) will the execution of this code take if N = 100? (Leave your answer in rational form – don't need to calculate out the result).

Question #9 (6 points total. a=1, b=1, c=2, d=2 points)

You are writing an exception handling routine to manage address error exceptions. Address error exception occurs when the processor attempts to read a 16-bit word or a 32-bit long word at an odd address. Attempting to read a word at an odd address would require two accesses to memory – one to access the odd byte of an operand and the other to access the even byte at the next address. The new address error exception handling routine is to be located at address $6000.
This is a group 0 exception, and the saved PC will depend on the type of instruction being executed. Also, for simplicity, you may assume that nothing has been pushed onto the stack since the exception occurred.

a) Write the MC68000 instructions that will install the handler in the exception vector table.

b) As part of the handler, write the instruction that will extract the address being accessed when the exception occurred (this is the address that caused the exception) and put the address into A0.

c) Assume that the offending address is in A0 and word size read access was being made when the exception occurred. Write the instructions that will correctly read the contents at A0 and load the results into D0 without causing any exceptions.

d) Assume it was determined that the instruction which caused the exception was a single-effective-address instruction. Write MC68000 instructions that, as part of the handler, will determine the mode and register number of the effective address access. Store the effective addressing mode in D1 and the register number in D2.

Question #10 (5 points total. a=1 point, b=1 point, c=3 points)

Write your own TRAP#12 handler to print a string of characters to the screen. The handler is to assume that A5 points to the beginning of the string and A6 points one byte past the end of the string. Printing is to be done using polling on the DUART, where the output of channel A is connected to the screen.

a) Calculate the exception vector address for your handler.

b) Write the instruction to install your handler.

c) Write the TRAP#12 handler, assuming the following constants have been declared:

```
SRA   EQU   $FF0003     Status Register A
TBA   EQU   $FF0007     Transmit Buffer A
```

Note: Bit #2 of SRA is TxRDYA.

Question #11 (3 points)

Describe the purpose of the DUART registers ISR (Interrupt Status Register), IMR (Interrupt Mask Register), and IVR (Interrupt Vector Register).
Note: Only 1 point is given per register description, so be brief and to the point.

Question #12 (3 points)

D0 = FFFFFFFF	D1 = 00000001	D2 = 00000000
A0 = 008003D4	A1 = 00002002	A2 = 00000000

Given the above initial conditions, assume the following binary image (machine code) is loaded at address $1000 and executed (GO $1000). What is the content of D0 right before the instruction at $1006 is to be executed.

Address	Address Content
$1000:	$2001
$1002:	$0600
$1004:	$000A
$1006:	