EEAP 282

EXAM #2
SOLUTIONS

October 21, 1997

NAME: _____ CWRUnet ID:_____

IMPORTANT INFORMATION:
1. All questions are worth TEN (10) points apiece.
2. Exam is closed book, closed notes. Only the M68000 Programmer's Reference Manual and/or Programming Reference Card are allowed to be used.

Problem	Score
1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
TOTAL SCORE	

Notation used: $ indicates hex, % indicates binary
NOTE: If you come to an address calculation that is outside the data given for any problem, BE SURE TO SHOW HOW YOU DID YOUR CALCULATIONS.

Note: 38+84+34=156 took exam in class + 4 makeups

BIT MANIPULATION INSTRUCTIONS:

1. For this problem you are given that

```
($25EFE)=$0B          ($2FFFE)=$60
($25EFF)=$EA          ($2FFFF)=$A0
($25F00)=$FF          ($30000)=$0E
($25F01)=$40          ($30001)=$F0
($25F02)=$AF          ($30002)=$EE
($25F03)=$8F          ($30003)=$34
($25F04)=$10          ($30004)=$78
```

and

```
SR=$0500
```

If (D1) = $00000005 and (D2) = $0002FFFE, what is the result of executing the instruction `BCHG D1,D2`? Be sure to indicate any changes to the Status Register.

Answer: The memory is irrelevant.
(D2)=$0002 FFDE
(6 pts) The instruction looks at the 5th bit of $0002FFDE = %0000 0000 0000 0010 1111 1111 1111 1110 which is 1 and changes it to 0 to give (D2)= %0000 0000 0000 0010 1111 1111 1101 1110.

(4 points) The SR does not change. The status register does not change since the bit tested was 1 originally.

BRANCHING INSTRUCTIONS:

2. For the following program segment:

```
        CLR.L       D1
        MOVE.L      #10,D0
LOOP2:  ADD.L       D0,D1
        SUBQ        #1,D0
        BPL         LOOP2
        <next instruction>
```

`SUBQ` gets executed _____ times Answer: 11 ($B) 5 points
(D1) = _____ after the program stops. Answer: 55 ($37) points

3. What are the values of the N, Z, V and C flags after each of the following instructions is executed **independently**. Assume the flags are all zero and the register contents are as shown immediately prior to executing each instruction

register contents:

```
(D0) = $FFFFFFFF
(D1) = $00001000
(A0) = $00010010
(A1) = $00010020
(A3) = $00010030
```

memory contents:

```
($10000.W) = $1234
($10010.W) = $31D0
($10020.W) = $0D0A
($10030.W) = $1234
```

Instruction:	X	N	C	V	Z
before each instruction	0	0	0	0	0
`CMP.B $10000,D0`	0	1	0	0	0
`CMPA.W D1,A0`	0	0	0	0	0
`CMPI.W #$1234,(A0)`	0	0	0	0	0
`CMPM.B (A1)+,(A3)+`	0	0	0	0	0

ANSWER:

Many people had a problem with CMPA.W D1,A0. According to the Programmers Reference Manual the CMPA instruction does not change the X bit and the N,Z,V and C bits are set according to the result. This is correct, however, many students interpreted the instruction as computing the sign extended word-length contents of both A0 and D1, i.e.,

```
   $0000 0010
-  $0000 1000
```

which gives

$FFFF F010

and sets both the N and C bits. However, this is incorrect. The confusion comes from interpreting the .W works. The correct interpretation is that the instruction uses the entire contents of A0 and the .W only applies to the source operand. In this case the actual operation computed is:

```
   $0001 0010
-  $0000 1000
```

which gives

$0000 F010 with no C or N flag set. This was a subtle problem.

Take off 0.5 points for each flag that is wrong.

ADDRESSING:

4. Given that

```
($4500) = $12345678,
($4508) = $82344234,
(A0) = $00004500
(D0) = $FFFFFF88
```

what will be the contents of A0, D0 and the status bits after the execution of

```
ADD.B     (A0)+,D0
```

(A0)=_____ $00004501 (D0)=_____ $FFFFFF9A

N=____1 Z=____0 V=____0 C=____0

Score 4 points each for A0 and D0, 1/2 point for each flag.

Answer: (A0)=$00004501,
(D0)=$FFFFFF9A and
XNZVC=01000

5. For this problem assume that

```
(A1) = $ 00005500,
(A2) = $ 00005502,
(D1) = $ FFFF0000,<---should be $0000FFFF
(D2) = $ 00000000
```

and

```
($5500) = $ 10000A04 Note all long words
($5504) = $ 00008002 <---should be $00005502
($5508) = $ 10023013
($550C) = $ B0214A31
($5510) = $ C1D11122 <---should be $02CD3256
                          ($5514) = $C1D11122
                          ($5518) = $ABCD3256
```

Indicate what is in D1, D2, A1 and A2 after the following program fragment is executed:

```
MOVEA.L     4(A1),A1
MOVE.W      #$9000,D1<---should have been #$E
MOVE.B      0(A1,D1.L),D1
MOVE.W      6(A2,D1.W),D2
```

(A1) = _____ ANSWER $0000 5502

(A2) = _____ ANSWER $0000 5502

(D1) = _____ ANSWER $0000 0002

(D2) = _____ ANSWER $0000 3013

HINT: Look carefully at how addresses are calculated in this problem.

ANSWERS (For the #$E):
After the first instruction (A1) = 0000 5502
After the second instruction, (D1) = 0000 000E
The third instruction adds D1 ($0000 000E) to A1 ($0000 5502) to produce the address $0000 5510. The byte $02 is fetched and put in the least significant byte of D1 making (D1) = $0000 0002.
The address in A1 $0000 5502 is added to the word length contents of D1 $0002 plus an addition $06 and all added together to give $

ANSWERS (For the #$9000):
(3 points) After the first instruction (A1) = $0000 5502
(3 points) The value of (A2) never changes, so (A2)=$00005502
(4 points) The third instruction adds D1 ($0000 9000) to A1 ($0000 5502) to produce the address $0000 E502. We don't know what this byte is so just look for this address. Full credit was given if the address $0000E502 was anywhere.
(don't count) The address in A1 $0000 5502 is added to the word length contents of D1 which we don't know plus an addition $06 and all added together to give $????

6. Assume that

```
($53EFE)=$ 0A          ($53F05)=$ 00 -> $82
($53EFF)=$ EE          ($53F06)=$ 00 -> $0A
($53F00)=$ 03          ($53F07)=$ 00 -> $EE
($53F01)=$ 82          ($53F08)=$ 00
($53F02)=$ 0A          ($53F09)=$ 00
($53F03)=$ EE          ($53F0A)=$ 00
($53F04)=$ 30          -> $03   ($60004)=$ 00
```

and

```
(A0)=$ 0005 3F00
```

What is in A0 and what memory, if any, is changed when the instruction

```
MOVE.L  (A0)+,(A0)
```

is executed?

Answer:
(A3)= $0005 3F04 (5 points)
($53F04.L)=$03820AEE (5 points)
The instruction fetches $03 82 0A EE from memory,
increments A0 by one long word (4 bytes) to $0005 3F04, and
puts $03 82 0A EE into memory beginning at the new location. Partial credit was given if you got the length of the move or the address wrong.

ADDRESS REGISTER INSTRUCTIONS AND LABELS:

7. What is in A0, A1 and D2 after the following program fragment is executed? Initially, (A0) = $10000000, (A1)=$FFFF0000, and (D2)=$FFFFFFFA.

```
        ORG     $7200
TABLE1  DC.L    $7000
        DC.L    $7008 AB12
        DC.W    $0100,$0090,$02EE,$AB02
        DC.B    $82
        ORG     $7400
TABLE2  DC.W    $0200,$0290,$01EE,$AC0D
        DC.L    $1000

        ORG     $7000
main7   MOVEA.W #TABLE1,A0
        MOVEA.W 4(A0),A1
        ADDA    A0,A1
```

(A0) = _____ ANSWER: $0000 7200

(A1) = _____ ANSWER: $0000 7200

(D2) = _____ ANSWER: $FFFF FFFA

Scoring: -4 points each for A0 and A1. Take off 2 points for D2. Take off -2 point each for any sign extension or an incorrect upper word. Trickly problem. It loads $00007200 into A0. The word at $4+$00007200=$00007204 is $0000. This $0000 gets put into A1 and sign extended to $00000000. We then add $00007200 to it to get $00007200.

STATUS REGISTER:

8. What are the values of the N, Z, V and C flags as each of the following instructions is executed in sequential order. Assume the flags are all initially zero.

Instruction:	X	N	C	V	Z
initial value of SR	0	0	0	0	0
MOVE.W #$FEC2,DO	0	1	0	0	0
MOVE.W #$AF5D,D1	0	1	0	0	0
ADD.W D1,D0	1	1	1	0	0

Scoring: Take off 0.6 point for each flag that is wrong.

DEBUGGER AND LAB RELATED QUESTIONS:

9. Answer the following questions about the debugger screen shown below:

```
======================Monitor====================12===========Stack=======14==
|1                                                ||    00000010=4E724E72  |
|2                                                      0000000C=4E724E72  |
|3                                                      00000008=4E724E72  |
|4                                                      00000004=4E724E72  |
|5                                                    SP->00000000=4E724E72  |
================================================================================
                        Code                        11  =======Registers=====13==
  00009000 41F9 0000 9500    LEA     BUF,A0            |PC=00009010 pi=0000900E|
1 00009006 4E71              NOP                        D0=00000000 A0=00009500|
  00009008 323C 0007         MOVE.W  #$7,D1             D1=00000007 A1=00000000|
  0000900C 7400              MOVEQ   #$0,D2             D2=0000FFFE A2=00000000|
 LOOP:                                                 D3=0000004E A3=00000000|
  0000900E 1618              MOVE.B  (A0)+,D3           D4=00000000 A4=00000000|
  00009010 5642              ADDQ.W  #$3,D2             D5=00000000 A5=00000000|
  00009012 0C03 0046         CMPI.B  #$46,D3            D6=00000000 A6=00000000|
  00009016 57C9 FFF6         DBEQ    D1,LOOP            D7=00000000 A7=00000000|
  0000901A 33C2 0000 950C    MOVE.W  D2,DAT             SR=0010011100000000    |
  00009020 4E71              NOP                           T S  III   XNZVC    |
                                                       ========================
STATUS: Command         68000   MODULE: exam2_f97        BREAK #: 0   HELP=F5
> Program Step
  Breakpt  Debugger  Expression  File  Memory  Program  Symbol  Window
  Assign  Block_Operation  Display  Map  Inport  Outport  Unload_BBA  Register
```

(a) According to the above debugger screen what will be the NEXT instruction to be executed? Describe what parts of the screen will change AFTER this instruction is executed. Be as specific as you can.

ANSWER: PC=$00009010 which means
ADDQ.W #$3,D2 .will be executed next (2 points)
This instruction adds 3 to the contents of D2, i.e. $00000003 + $0000FFFE. Note that this will be a word addition giving (D2)=$00000001. (1 points)
The SR will change to XNZVC=10001. (1 points)
PC and PI will also change. No points off for this.

(b) What debugger command do you use to make MOVE.W #$7,D1 instruction the NEXT instruction to be executed?

ANSWER:
To execute this instruction you have to set the Program Counter (PC) to this address. This is done by the command
Memory Register @PC=00009008h. Both
Memory Register @PC=00009008h and
Program Step From $00009008h
were acceptable answers. The following was not accepted:
Program Run From $9008

(c) Suppose that the debugger screen does NOT show the correct values. What simple command will redraw the debugger screen?
ANSWER: ^L

10. Define the following terms:

(a) memory mapped i/o:
The i/o operations are "mapped" to physical addresses in memory. Uses a memory address for read and write operations.

(b) polling
Continuous monitoring of a variable. Typically used for input/output and usually implemented as an infinite loop.

LEA AND ADDRESS REGISTER INSTRUCTIONS:

11. Give the value of A0 after executing each of the following instructions. Assume that (A0)=$00FF6000, (D0)=$4371FDEA and ($6000.L)=$42B01152 **before** each instruction is executed.

instructions	(A0)
`ADDA.W #$C000,A0`	`$00FF2000`
`SUBA.L #$14,A0`	`$00FF5FEC`
`MOVEA.W (A0),A0`	`$42B01152`
`LEA 14(A0,D0),A0`	`$00FF5DF8`

```
ANSWERS:
(a) Sign extends $C000 to $FFFFC000 and adds this to
$00FF6000 to give $00FF2000. No memory changes.
(b) Subtracts $14 from $00FF6000 to give $00FF5FEC.  No
memory changes.
(c) Bad problem.  Moves the word at memory location
$00FF6000 to A0 and sign extends it.
(d) Computes the address $00FF6000
                         $FFFFFDEA
                       + $0000000E
                       ------------
                         $00FF5DF8
Scoring -2.5 points each problem.  -1 for upper word.
```

12. Assume the following data is available for each of the indicated instructions. Give the resulting 32-bit contents of registers D2 and A3 and the contents of memory after executing the instructions independently.

Address	$012218	$01221A	$01221C	$01221E	$012220
Contents	$00F7	$AB06	$FFFD	$01A3	$000E
(a)	$00F7	**$0000**	**$00A3**	$01A3	$000E
(b)	$00F7	$AB06	$FFFD	$01A3	$000E
(c)	$00F7	$AB06	$FFFD	$01A3	$000E
(d)	$00F7	$AB**AB**	$FFFD	$01A3	$000E

	D2	A3
initially	$FFFF000B	$0001221A
(a)	$FFFF000B	$000122**1E**
(b)	$FFFF000B	$0001221A
(c)	$FFFF**0102**	$00012218

(d)	$FFFF000B	$0001221**C**

```
(a) MOVE.L  #$A3,(A3)+
(b) LEA     (A3),A3
(c) ADD.W   -(A3),D2
(d) MOVE.B  (A3)+,(A3)+
```

Answers:
(a) Moves $A3 to the memory location pointed to by the contents of A3, i.e. ($0000101A.L) becomes $000000A3. The address register is then incremented by 4 to $0000101E.

(b) Moves the contents of A3 into A3. Nothing changes.

(c) Decrements (A3) by 2 to $00012218. Gets $00F7 and adds this to $000B giving $FFFF0102 in D2.

(d) Gets the byte at $0001221A, $AB, adds 1 to $0001221A to get $0001221B. and moves $AB to the byte in memory $0001221B. It then increments the address by one byte to get $0001221C.

Scoring. Take off 1 point for each register wrong. Take off 1/2 point for each memory location wrong.