zap! OLD EXAM SERVICE
PHONE: 721-8805 FAX: 721-8728

COURSE: CSC 230
OF PAGES: 4
DATE: December 1994
(X20¢/PG=) $ +GST: 0.86
PROFESSOR: D. Wessels
ADDL. INFO:

PART 1: COMPREHENSIVE [90 marks]

Answer questions 1 through 5 based on the following partial listing file for an MC68000 program. The program consists of a MAIN routine with a call to local subroutine FOO.

```
Line Address          Assembly Language Instructions
01  000000            xref       decin_long,decout_long,newline,stop

02  000000  MAIN:     movem.l    d0-d7/a0-a4,-(sp)
03  000004            jsr        decin_long
04  000008            move.l     d0,-(sp)
05  00000A            jsr        FOO
06  00000E            adda.l     #4,sp
07  000012            jsr        decout_long
08  000016            jsr        newline
09  00001A            movem.l    (sp)+,d0-d7/a0-a4
10  00001E            jsr        stop

11  000022  FOO:      link       a6,#0
12  000026            movem.l    d1-d7/a0-a4,-(sp)
13  00002A            move.l     8(a6),d1
14  00002E            move.l     d1,d2
15  000030            move.l     d1,d3
16  000032            clr.l      d4
17  000034            move.l     #15,d5

18  00003A  FOR:      asl.l      #1,d2
19  00003E            asr.l      #1,d3
20  000042            bcc        WASZERO
21  000046            add.l      d2,d4
22  000048  WASZERO:  dbra       d5,FOR

23  00004C            move.l     d4,d0
24  00004E            movem.l    (sp)+,d1-d7/a0-a4
25  000052            unlk       a6
26  000054            rts

27  000056            end
```

Question 1 [15 marks]

[10] (i) Provide meaningful comments for the program (write on the listing above).

[5] (ii) In one or two short sentences, describe the effect of the program.

Question 2 [10 marks]

[5] (i) Draw and clearly label a picture depicting the top of the stack immediately following the execution of the jsr FOO instruction of line 05.

[5] (ii) Draw and clearly label a picture of the current stack frame immediately prior to the execution of the move.l d1,d2 instruction of line 14.

Question 3 [10 marks]

Assuming a load address of $002000 for the MAIN program:

[4] (i) What would be the new contents of address register A0 if we inserted the instruction lea MAIN,A0 after the jsr newline instruction of line 08?

[3] (ii) What is the hex value placed on the address bus to fetch the opcode word for the instruction bcc WASZERO of line 20?

[3] (iii) What is the value of the program counter just prior to fetching the word containing the offset in the instruction move.l 8(a6),d1 of line 13?

Question 4 [15 marks]

Suppose the opcode word for the instruction of line 21 was $2802. Decode the new instruction. What effect would this have on the program? (The relevant decoding information is included in the appendix to this exam).

Question 5 [15 marks]

[9] (i) What is the function of the symbol table, and how is it used in a typical two-pass assembler? Refer to the listing file for examples.

[6] (ii) Describe the addressing modes for the source and destination operands for the instructions on lines numbered 04, 11, and 13.

Question 6 [25 marks]

Suppose we have the following C program which we wish to translate into MC68000 assembly language:

```
extern long neg_total();
void main() {
        long size, sum, array[10];
        size = 10;
        sum = neg_total(array, size);
}
```

MAIN ROUTINE

[8] (i) Provide, using C calling conventions, the MC68000 assembly language code for the main routine.

SUBROUTINE

The neg_total subroutine referred to in the program above takes as input the address of an array of long integers and a long integer, size, which indicates the number of array elements to be processed, The subroutine calculates the absolute value of the sum of the negative elements in the array.

[5] (ii) Provide C-style pseudo-code for the neg_total subroutine.

[12] (iii) Translate your pseudo-code into MC68000 assembly language. Use C-style conventions for parameter passing and stack handling.

Question 7 [25 marks]

[5] (i) An assembler that runs on one machine and produces object code modules for another is called a cross assembler. Under what circumstances might a cross assembler be needed?

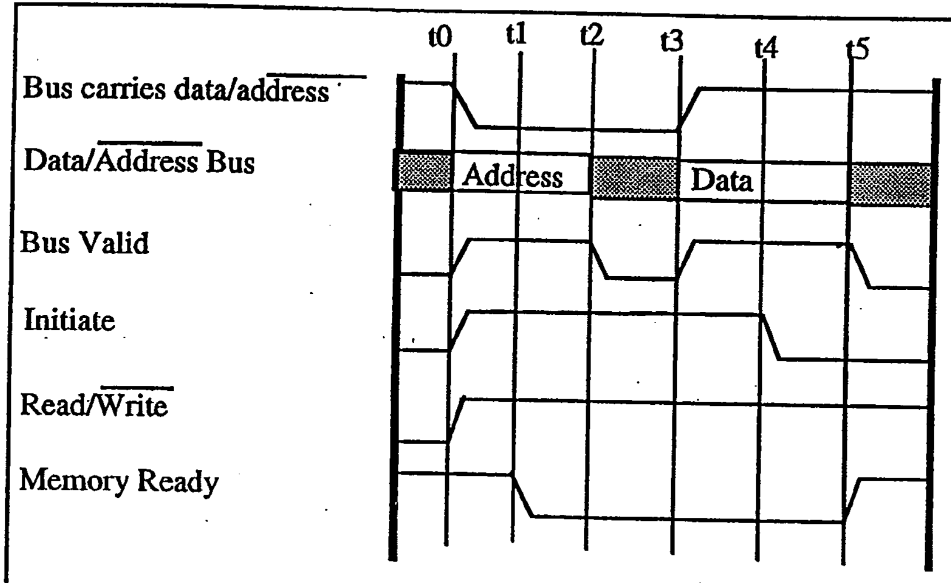
[7] (ii) How does the use of a variety of addressing modes allow us to write assembly language programs which may have shorter instructions than would be possible if the only addressing modes available were absolute long and register direct?

[7] (iii) How might the use of a variety of addressing modes allow us to write assembly language programs which may require fewer instructions than would be possible if the only addressing modes available were absolute and register direct?

[6] (iv) 68000 assembly language does not normally allow the use of PC-relative addressing modes for destination operands. Why would the language designers make such a decision?

Question 8 [15 marks]

Consider a new architecture, the FOOBLAH3, in which seperate data and address buses are not available. Suppose we have the following timing diagram depicting a word read from memory. Describe the protocol with respect to the signals sent and received by the CPU and memory during time steps t_0 - t_3. Time steps t_4 and t_5 have been done for you.


t0
t1
t2
t3
t4
t5
Bus carries data/$\overline{\text{address}}$
Data/$\overline{\text{Address}}$ Bus
Address
Data
Bus Valid
Initiate
Read/$\overline{\text{Write}}$
Memory Ready

The CPU controls the initiate signal, memory controls the memory ready signal, all other signals can be controlled by either device.

t_4: The cpu latches the data off the bus, clears the initiate signal, and waits for the memory ready signal to be set.

t_5: Memory stops refreshing the data on the bus, clears the bus valid signal, sets the memrory ready signal, and waits for the next time an initiate signal is sent by the cpu.

Question 9 [25 marks]

[5] (i) Most interrupts are not processed until the current instruction completes execution. Two exceptions are address errors and bus errors. Why are these allowed to be processed in the midst of the fetch/decode/execute cycle?

[5] (ii) How is the exception vector number and the exception vector address determined for an auto-vectored interrupt?

[10] (iii) If you wished to substitute your own service routine for the divide-by-zero exception, how would you go about doing this on an MC68000 processor?

[5] (iv) What signals are generated for a non-auto-vectored interrupt during the interrupt acknowledgement cycle, and how does the generating device communicate the exception number to the CPU?

Question 10 [25 marks]

[6] (i) What kind of a hardware device is an MMU? Suggest several reasons why one might be added to a computer system.

[6] (ii) Assuming logical addresses of 16 bits, and an MMU with 64 pages in its memory map, how large is a page, and how wide a physical address bus can the system effectively utilize?

[6] (iii) How can an MMU be used to protect users from using and/or corrupting one another's data in a multi-user system?

[7] (ii) Sketch the basic block diagram of a 68000-style architecture to show how the MMU relates to the CPU, main memory, the data bus, control bus, and address bus (for both logical and physical addresses).