COURSE: CSC 230 # OF PAGES: 10

DATE: DEC 1996 (X20¢/PG=) $ +GST: $2.14

PROFESSOR: Miller & van. Emden ADDL. INFO: includes ANSWERS

1. (8 marks)
Write the 8-bit 2's complement representation of each value below as a binary number and its hexadecimal equivalent:

| | binary | hex | | binary | hex |
|---|---|---|---|---|---|
| a) 0 | 0000 0000 | 00 | b) 35 | 0010 0011 | 23 |
| c) -12 | 1111 0100 | F4 | d) -128 | 1000 0000 | 80 |

2. (4 marks)
Adding two 2's complement numbers having the same sign with the result having a different sign is a condition known as OVERFLOW. (P. 476)

This can never happen when subtracting two 2's complement numbers. Is this statement true or false? FALSE

3. (10 marks)
Consider the following program:

```
        org   $8000
count   rmb   2

        org   $9000
        lds   #$dfff
        ldd   #0
        std   count
loop    jsr   update
        ldd   count
        cpd   #77
        bls   loop
        jmp   $e000
```

6 marks

(a) Write the subroutine "update" in as few as possible instructions so that variable "count" is incremented by one each time "update" is executed.

too long
4 marks or fewer depending on how much too long

Does not have to be this

```
update  ldd   count
        addd  #1
        std   count
        rts
```

```
inc $8001
rts
```

```
update  ldd count  { ldaa count
                   { ldab count+1
        addb #1
        adca #0
        std count  { staa count
                   { stab count+1
        rts
```

4 marks

(b) How many times is subroutine "update" executed? 78

(For suppose "77" were "1". Then "update" is executed twice.

```
update  ldx #count
        inc 0,X
        rts
```

```
update  xgdx
        inx
        xgdx
        rts
```

4. (16 marks)
Consider the following program:

```
;The sum of the first 3 elements of array arr is placed in location
;labelled "total".

        org     $8000
arr     fcb     121,144,169,75,38,205
total   rmb     2
        org     $9000
        lds     #$dfff
        ldx     #arr
        ldy     #3
        jsr     sum
        std     total
        jmp     $e000

;Subroutine sum receives the number of elements to totalled in IY and
;the array address in IX. It leaves the result in ACCD.
sum     ldd     #0
loop    cpy     #0
        beq     end
        addb    0,x
        adca    #0
        inx
        dey
        bra     loop
end     rts
```

3 marks

(a) What is the content of accumulator A on exit from the call to subroutine "sum"?

$01

3 marks

(b) What is the content of index register X on exit from the call to subroutine "sum"?

$8003

(question continues on the next page)

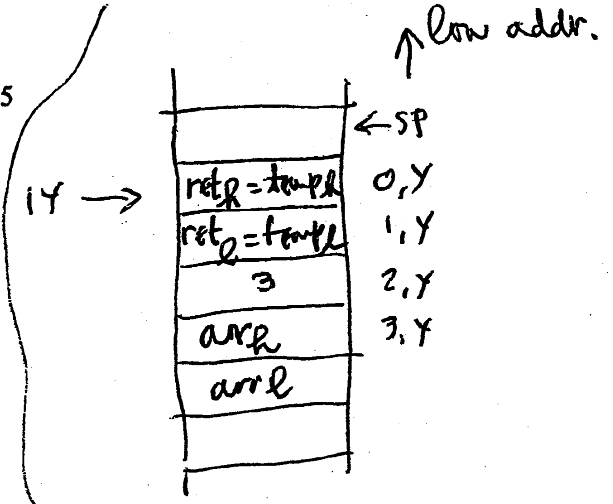
(4.c) The above program passes its parameters, both input and output, via registers. Below it is modified so that the input parameters are passed via the stack and so that the result is passed via program memory.

In the program below, parts have been omitted, as shown by underlines. Complete the program by writing the omitted parts at the underlines. Do not modify any of the given code.

→ not required for solution

```
;The sum of the first 3 elements of array arr is placed in location
;labelled "total".

        org     $8000
arr     fcb     121,144,169,75,38,205
total   rmb     2
        org     $9000
        lds     #$dfff
        ldx     #arr
        pshx
        ldaa    #3
        psha
        jsr     sum
temp    rmb     2
        ldd     temp
        std     total
        jmp     $e000
```



```
;Subroutine sum receives the number of elements to totalled in IY and
;the array address in IX. It leaves the result in ACCD.
sum     ldd     #0
        tsy
        ldx     3,y
loop    tst     2,y
        beq     end
        addb    0,x
        adca    #0
        inx
        dec     2,y
or jmp  bra     loop
end     ldx     0,y      (IX) = temp = ret
        std     0,x        (temp) = sum
        ldd     0,y      (ACCD) = ret = temp
        addd    #2
        std     0,y      <(SP-1), (SP-2)> = temp+2
        rts
```

5. (10 marks)
Below you will find the same program that was the starting point one of the other questions, except that the first 2 instead of the first 3 elements of the array are added.

$\frac{1}{2}$ mark off for each wrong entry

```
;The sum of the first 2 elements of array arr is placed in location
;labelled "total".
        org     $8000
arr     fcb     121,144,169,75,38,205
total   rmb     2
        org     $9000
        lds     #$dfff
        ldx     #arr
        ldy     #2
        jsr     sum
        std     total
        jmp     $e000

;Subroutine sum receives the number of elements to totalled in IY and
;the array address in IX. It leaves the result in ACCD.
sum     ldd     #0
loop    cpy     #0
        beq     end
        addb    0,x
        adca    #0
        inx
        dey
        bra     loop
end     rts
```

(question continues on the next page)

5. (continued)
Below is an interaction with the Buffalo monitor. The program that is loaded is the one shown above. Below is a trace, which is incomplete. Please complete it.

```
BUFFALO 3.4
>load t
done
>br 9000
9000 0000 0000 0000
>g 9000
P-9000 Y-0000 X-27FF A-00 B-09 C-D0 S-0041
>t 20
LDS   #$DFFF        P-9003 Y-0000 X-27FF A-00 B-09 C-98 S-DFFF
LDX   #$8000        P-9006 Y-0000 X-8000 A-00 B-09 C-98 S-DFFF
LDY   #$0002        P-900A Y-0002 X-8000 A-00 B-09 C-90 S-DFFF
JSR   $9013         P-9013 Y-0002 X-8000 A-00 B-09 C-90 S-DFFD
LDD   #$0000        P-9016 Y-0002 X-8000 A-00 B-00 C-94 S-DFFD
CPY   #$0000        P-901A Y-0002 X-8000 A-00 B-00 C-90 S-DFFD
BEQ   $9025         P-901C Y-0002 X-8000 A-00 B-00 C-90 S-DFFD
ADDB  $00,X         P-901E Y-0002 X-8000 A-00 B-79 C-90 S-DFFD
ADCA  #$00          P-9020 Y-0002 X-8000 A-00 B-79 C-94 S-DFFD
INX                 P-9021 Y-0002 X-8001 A-00 B-79 C-90 S-DFFD
DEY                 P-9023 Y-0001 X-8001 A-00 B-79 C-90 S-DFFD
BRA   $9016         P-9016 Y-0001 X-8001 A-00 B-79 C-90 S-DFFD
CPY   #$0000        P-901A Y-0001 X-8001 A-00 B-79 C-90 S-DFFD
BEQ   $9025         P-901C Y-0001 X-8001 A-00 B-79 C-90 S-DFFD
ADDB  $00,X         P-901E Y-0001 X-8001 A-00 B-09 C-91 S-DFFD
ADCA  #$00          P-9020 Y-0001 X-8001 A-01 B-09 C-90 S-DFFD
INX                 P-9021 Y-0001 X-8002 A-01 B-09 C-90 S-DFFD
DEY                 P-9023 Y-0000 X-8002 A-01 B-09 C-94 S-DFFD
BRA   $9016         P-9016 Y-0000 X-8002 A-01 B-09 C-94 S-DFFD
CPY   #$0000        P-901A Y-0000 X-8002 A-01 B-09 C-94 S-DFFD
BEQ   $9025         P-9025 Y-0000 X-8002 A-01 B-09 C-94 S-DFFD
RTS                 P-900D Y-0000 X-8002 A-01 B-09 C-94 S-DFFF
STD   $8006         P-9010 Y-0000 X-8002 A-01 B-09 C-90 S-DFFF
JMP   $E000         P-E000 Y-0000 X-8002 A-01 B-09 C-90 S-DFFF
LDAA  #$93          P-E002 Y-0000 X-8002 A-93 B-09 C-98 S-DFFF
STAA  $1039         P-E005 Y-0000 X-8002 A-93 B-09 C-98 S-DFFF
LDAA  #$00          P-E007 Y-0000 X-8002 A-00 B-09 C-94 S-DFFF
STAA  $1024         P-E00A Y-0000 X-8002 A-00 B-09 C-94 S-DFFF
LDAA  #$00          P-E00C Y-0000 X-8002 A-00 B-09 C-94 S-DFFF
STAA  $1035         P-E00F Y-0000 X-8002 A-00 B-09 C-94 S-DFFF
LDX   #$2600        P-E012 Y-0000 X-2600 A-00 B-09 C-90 S-DFFF
STX   $98           P-E014 Y-0000 X-2600 A-00 B-09 C-90 S-DFFF
>
```

6. (~~20~~ 21 marks) => some students may get 101 for the whole exam - no problem

In the program below, parts have been omitted, as shown by underlines. Complete the program by writing the omitted parts at the underlines. Do not modify any of the given code.

It may be the case that some entire lines of the program, not indicated by underlines, have been omitted. If so, add such lines, and add as few as possible.

```
;To run the program below, the BLT11 board has the left button connected
;to the PA0 (IC3) pin. The buzzer is connected to the PA5 (OC3) pin.
;
;The BLT11 board has the same interrupt vector jump table as the EVB
;board.
;
;The program uses the OC3 interrupt to activate the buzzer.
;Releasing the left button generates the IC3 interrupt.
;
;The resulting behaviour is as follows: releasing the left botton
;causes the button to emit a 1024 Hz tone if it was silent, and to
;silence it if it was sounding the tone.
;
;OC2F clear implies that time is still in the debouncing period of a
;button press.
```

mark each

```
DEBNCE    equ   50000  ;debouncing period is 50000 cycles
HALFP     equ   1024   ;half period for 1024 Hz is 1024 cycles
REGBAS    equ   $1000  ;base for register indexing
TCNT      equ   $0e    ;offset for TCNT register
TOC2      equ   $18    ;offset for TOC2 register
TOC3      equ   $1a    ;offset for TOC3 register
TCTL1     equ   $20    ;offset for TCTL1 register
TCTL2     equ   $21    ;offset for TCTL2 register
TMSK1     equ   $22    ;offset for TMSK1 register
TFLG1     equ   $23    ;offset for TFLG1 register
OC3I      equ   $20    ;select OC3I bit 5 in TMSK1
OC3F      equ   $20    ;select OC3F bit 5 in TFLG1
OC2F      equ   $40    ;select OC2F bit 6 in TFLG1
IC3I      equ   $01    ;select IC3I bit 0 in TMSK1
IC3F      equ   $01    ;select IC3F bit 0 in TFLG1
IC3TGL    equ   $01    ;rising edge IC3 (bits 1, 0 in TCTL2)
OC3TGL    equ   $10    ;toggle mode OC3 (bit 4 in TCTL1)
```

(question continues on the next page)

```
org $ 00d9     }  in either order
jmp   oc3_isr  }  anywhere
 org $ 00e2 FFEA }
 jmp ic3_isr     }
```

2 marks

2 marks

6. (continued)

2 marks for each correct entry

```
        org     $9000
        sei
        lds     #$dfff
        ldx     #REGBAS
        ldaa    #IC3TGL                 ;rising edge...
        staa    TCTL2,x                 ;... for IC3
        ldaa    #OC3TGL                 ;toggle mode ...
        staa    TCTL1,x                 ;... for OC3
        bclr    TMSK1,x OC3I            ;buzzer initially off
        bset    TMSK1,x IC3I            ;enable IC3 interrupt
        cli
loop    bra     loop                    ;do nothing but listen to interrupts


;ISR for IC3 and left button
ic3_isr ldx     #REGBAS
        ldaa    #IC3F                           ;clear ...
        staa    TFLG1,x                         ;... IC3F
        brclr   TFLG1,x OC2F endic3             ;exit if OC2F clear
        brclr   TMSK1,x OC3I sk1ic3             ;branch if OC3 int. disabled
        bclr    TMSK1,x OC3I                    ;disable OC3 interrupt
        bra     sk2ic3
sk1ic3  bset    TMSK1,x OC3I                    ;enable OC3 interrupt
sk2ic3  ldd     TCNT,x
        addd #DEBNCE
        std TOC2,x            ;set up for OC2F after DEBNCE cycles
        ldaa #OC2F            ;clear ...
        staa TFLG1,x          ;... OC2F
endic3  rti

;ISR for OC3 and buzzer
oc3_isrldx      #REGBAS
        ldaa    #OC3F         ;clear ...
        staa    TFLG1,x       ;... OC3F
        ldd     TCNT,x
        addd    #HALFP
        std     TOC3,x
        rti
```

Note: alternative ISR installation:

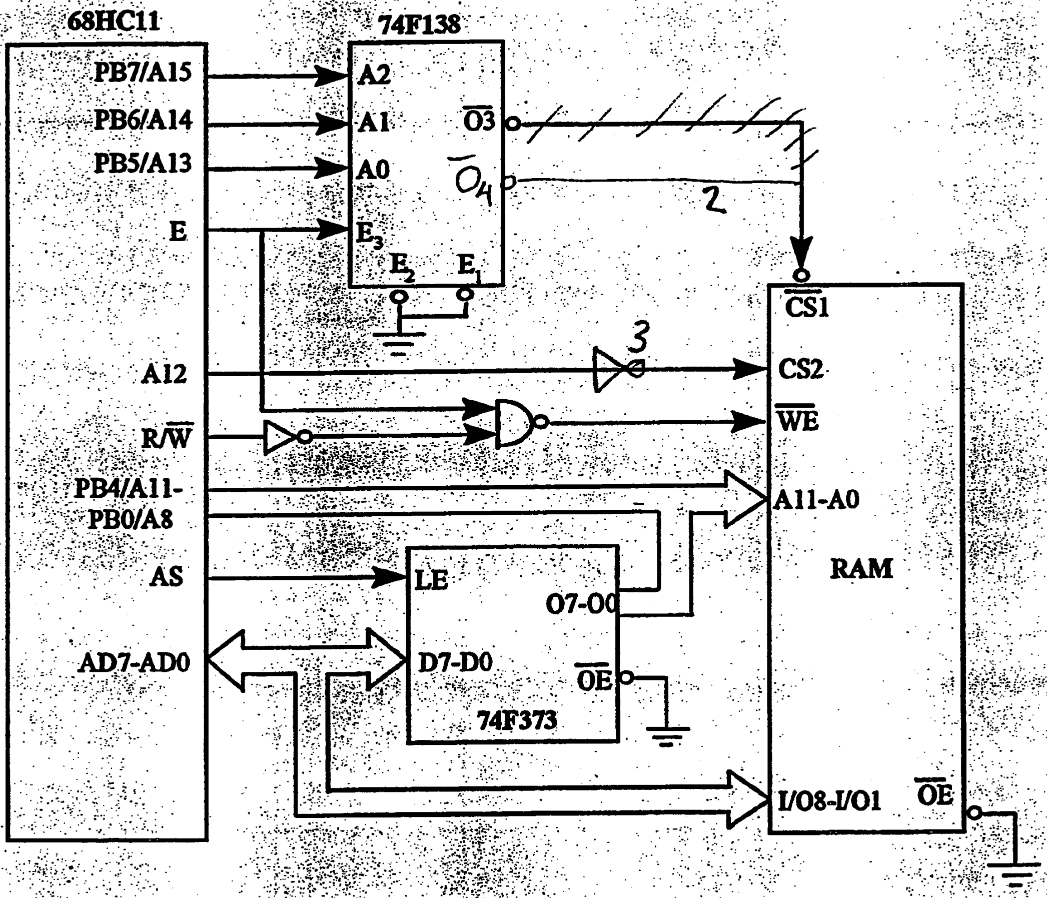
```
ldaa $7E      % opcode for jmp          { ldaa $7E
staa $ 00d9                             { staa $ 00E2
ldx #oc3_isr  }                         { ldx # ic3_isr
stx   $ 00da  } or use reg Y or reg D   { stx  $ 00E3
```

7. (10 marks)
Consider the diagram below (note it is **NOT** identical to the one in the text).

(a) How many bytes of memory does the RAM chip have? 4K (or $2^{12}$) 2

(b) What range of addresses does the RAM chip occupy? $7000 - 7FFF
1 2

(c) Modify the diagram so that the RAM chip occupies the block of memory beginning at location $8000.



8. (6 marks)
Indicate whether each of the following statements is true or false.

2 marks each

A processor using memory mapped I/O requires special I/O instructions. false

A processor which has separate data and address busses does not need an address strobe signal. true

Multiplexed addressing is used to reduce the pin count of the processor. true

9. (3 marks)
Indicate whether each of the following statements is true or false.

RISC designs are so efficient that CISC designs will soon fail to exist. false

Effective RISC design requires the processor have at least one cache. true

RISC designs are optimized as a target architecture for compilers rather than as a target architecture for assembly language programmers. true

10. (7 marks)

1 mark

(a) A PowerPC processor can achieve an execution rate of more than one instruction per clock cycle. It is thus an example of a superscalar design.

1 mark

(b) State briefly one major feature that helps achieve this.

multiple pipelines

(c) A typical RISC design has a five level memory hierarchy. Identify the five levels from (i) fastest to (v) slowest.

5 marks

(i) CPU registers

(ii) level-1 cache

(iii) level-2 cache

(iv) main memory (also: DRAM)

(v) disk

11. (2 marks)
What type of parallel processing does the acronym SIMD describe.

single instruction stream, multiple data stream

12. (4 marks)
Circle the letter identifying the correct answer for each of the following:

A set of workstations on a LAN working on the same problem is an example of a

a) tightly coupled shared memory multiprocessor.
1 mark [✓ b)] loosely coupled private memory multiprocessor.

In all multiprocessor systems,

a) the processors are identical.
1 mark [✓ b)] the processors are not necessarily identical.

In a multiprocessor system,

a) the processor to memory connections must be static.
1 mark [✓ b)] the processor to memory connections may be static or dynamic depending on the design.

Transputers are best used in a

1 mark [✓ a)] mesh interconnection scheme.
b) hypercube interconnection scheme.

****** END OF EXAMINATION ******