BLG212E - SAMPLE MIDTERM EXAM
(1.5 hours)

QUESTIONS

QUESTION 1) [15 points]

- Write the X and Y decimal numbers as **Binary** (8 bits), for each question below.
- Write the result of **X+Y** as **Binary**, for each question.
- Write which **Status Flags** (Overflow, Zero, Negative, Half Carry, Carry) are set to 1, for each question.

a) X = 120, Y = 75 (Both are unsigned.)

b) X = –30, Y = +45 (Both are signed. Negative number is represented as Two's complement.)

QUESTION 2) [50 points]
A memory subsystem will be designed, by using the following memory chip modules. (Data Bus is 8 bits).

- M1 : 16K×8 bit ROM
- M2 : 32K×8 bit RAM
- M3 : 32K×4 bit RAM
- M4 : 32K×4 bit RAM
- Note that the M3 and M4 are 4-bit chips, and they should be combined to obtain one 8-bit chip.
- Memory modules should be in consecutive address map.

a) [20 points]

- Determine the minimum number of lines required in the **Address Bus**.
- Calculate the total address capacity and the total used amount of memory (in Kilo bytes).
- For each of the four memory chips, write the **memory map ranges** (smallest and biggest addresses) with hexadecimal notation.

b) [30 points]

- Draw the memory design with all necessary connections.
  (Address bus, Data bus, Chip select signals).
- Use an address decoder.
- Assume memory chip select signals are active high (1).

QUESTION 3) [35 points] Write an Assembly program (use the EDU-CPU instruction set) to find the **largest** and the **second largest** elements in an array.

- Define a constant symbol named SIZE, which is equal to 5.
- Define a variable named ARRAY, with the defined SIZE. (Each element is 1 byte.)
- Initialize the ARRAY with the following decimal data : 1, 5, 2, 4, 3
- Define two variables named LARGEST and SECOND_LARGEST (1 byte each).
- Program should perform **looping** thru the array elements.
- The array should NOT be sorted.

ANSWERS

ANSWER 1) [15 points]

a) [8 points]

X = 0111 1000 (120)
Y = 0100 1011 (75)
+----------
1100 0011

Result with 8 bits = 195 (unsigned)

FLAGS:
Overflow=1 , Negative=1 , Half Carry=1

b) [7 points]

X = 1110 0010 (–30) (Two's complement)
Y = 0010 1101 (+45)
+----------
1 0000 1111

Result with 8 bits = +15 (Carry bit is discarded)

FLAGS:
Carry=1

ANSWER 2) [50 points]

a) [20 points]
The chips with 32K locations (2^{15}) requires 15 address lines for location selection.
The address decoder requires 2 address lines, since there are 4 chips for selection.
Total minimum number of lines in the Address Bus is = 15 + 2 = **17**
Total address capacity = $2^{17} = 2^7 * 2^{10}$ = **128** KB
Total used amount of memory = 16 + 32 + 16 + 16 = **80** KB

Smallest Adresses :

Chip	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	HEXADECIMAL
M1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$0 0000
M2	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$0 8000
M3	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$1 0000
M4	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	$1 0000

Biggest Adresses :

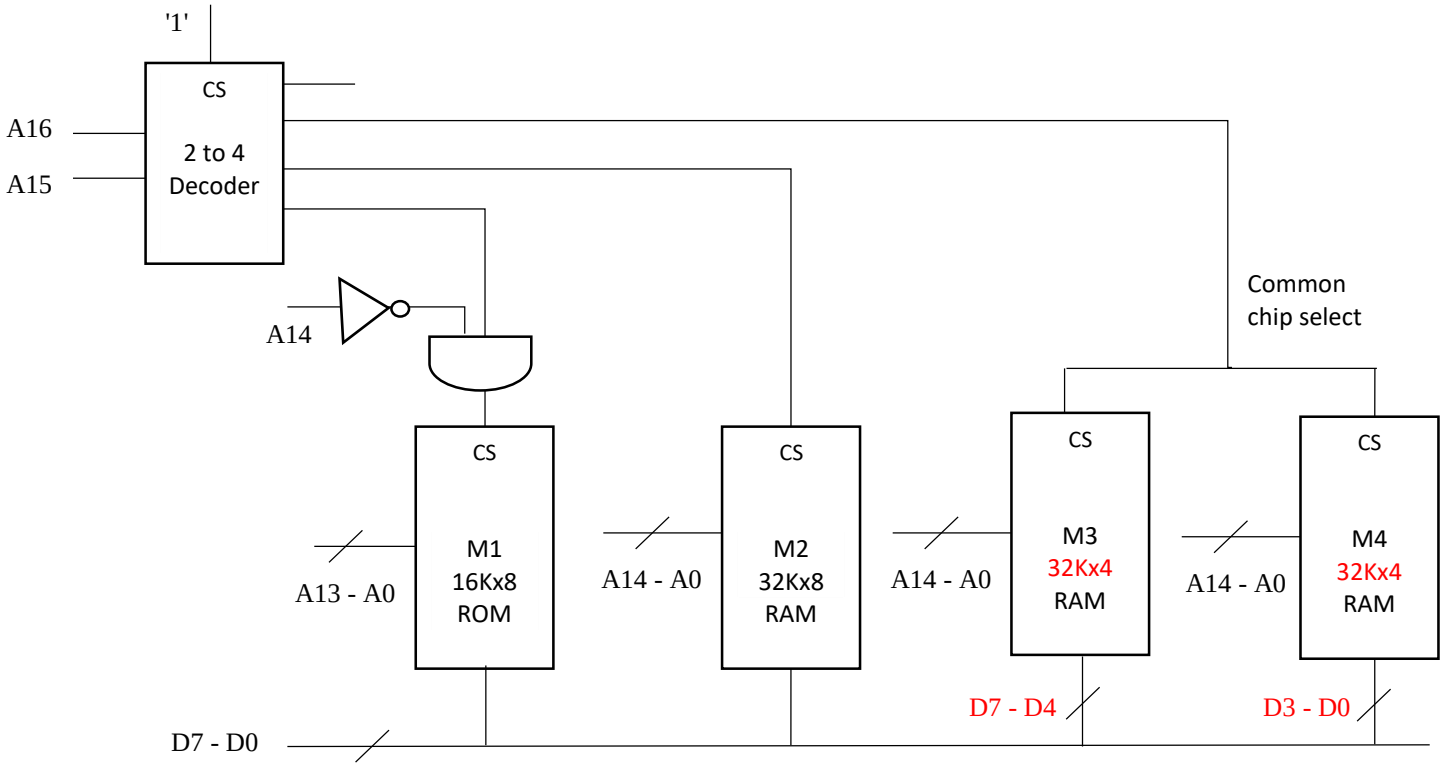
Chip	A16	A15	A14	A13	A12	A11	A10	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0	HEXADECIMAL
M1	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$0 3FFF
M2	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$0 FFFF
M3	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$1 7FFF
M4	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	$1 7FFF

Memory Map :

Module Name	Memory Chip Type	Chip Capacity (KB)	Smallest Address	Biggest Address
M1	16 K x 8 bit ROM	16 KB	0 0000	0 3FFF
EMPTY	None	16 KB	0 4000	0 7FFF
M2	32 K x 8 bit RAM	32 KB	0 8000	0 FFFF
M3	32 K x 4 bit RAM	16 KB	1 0000	1 7FFF
M4	32 K x 4 bit RAM	16 KB	1 0000	1 7FFF

(Note that the M3 and M4 chips share the same address range.)

b) [30 points]



ANSWER 3) [35 points]

```
*Program finds largest and second largest elements in an array.
*CD register is used as loop counter and as index on array.

SIZE EQU 5
DIZI   RMB  SIZE
    ORG  DIZI
    DAT  1,5,2,4,3   ;Array

LARGEST RMB 1
SECOND_LARGEST RMB 1

START
*Perform initializations
    STA 0, LARGEST
    STA 0, SECOND_LARGEST
    LDA  SK, DIZI   ;Beginning address of array
    LDA  CD, 0      ;Loop counter  initialized to 0

DONGU
    LDA A, <SK+CD+0>       ;Get next element of array to A
    CMP A, <LARGEST>       ;Compare
    BLT DEVAM1

*Update LARGEST and SECOND_LARGEST both
    LDA B, <LARGEST>
    STA B, SECOND_LARGEST
    STA A, LARGEST
    BRA DEVAM2
```

```
DEVAM1
    CMP A, <SECOND_LARGEST>     ;Compare
    BLT DEVAM2

*Update SECOND_LARGEST only
    STA A, SECOND_LARGEST

DEVAM2
    INC  CD                     ;Increment loop counter
    CMP CD, SIZE   ;Compare to loop limit
    BLT  DONGU     ;If CD is less than SIZE goto loop
    INT
```