Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή: Ηλεκτρολόγων Μηχανικών & Μηχανικών Υπολογιστών

Συστήματα Μικροϋπολογιστών (6ο εξάμηνο)
2η Ομάδα Ασκήσεων

Δημήτριος Καλαθάς - el18016
Δημήτριος Καλέμης - el18152

Ασκήσεις Προσομοίωσης

Άσκηση 1

Πρόγραμμα 1 σε Assembly

```
        IN 10H
        MVI A,00H               ;numbers from 0 to 255
        MVI D,00H               ;counter for numbers from 10H to 60H
        LXI H,0900H             ;addresses
        LXI B,0000H             ;counter for '1'
START:
        MOV M,A                 ;store in memory
        MOV E,A                 ;store A for later
        JMP ROTATE
ROTATE:
        STC                     ;set CY = 1
        CMC                     ;now CY = 0
        RAR                     ;rotate A right
        JNC FOUND_ZERO          ;If MSB of A is 0 go to FOUND_ZERO
        INX B                   ;else increase B by one
FOUND_ZERO:
        CPI 00H                 ;If A is 00H after rotation
        JZ RETRIEVE_A           ;get initial A
        JMP ROTATE              ;else rotate again
RETRIEVE_A:
        MOV A,E                 ;E has initial value of A
        JMP CHECK
CHECK:
        CPI 10H                 ;If number less than 10H
        JC NEXT                 ;go to NEXT
        CPI 60H             ;else check if number less than 60H
        JC BETWEEN              ;if yes go to BETWEEN
        JNZ NEXT                ;else go to NEXT
BETWEEN:
        INR D                   ;found number between in [10H,60H]
        JMP NEXT
NEXT:
        INX H                   ;next address
        INR A                   ;next number
        CPI 00H                 ;if A gets bigger than 255
        JZ FINISH               ;end
        JMP START               ;else go to START
FINISH:
        END
```

Άσκηση 2

Πρόγραμμα 2 σε Assembly

```
        MVI A,FFH           ;Turn off leds
        STA 3000H
        MVI D,64H           ;Set clock counter D=100
        LXI B,00C8H         ;Set delay 0.2sec
START:
        LDA 2000H           ;Load A the input
        ANI 80H             ;Check MSB of A
        CPI 00H             ;check if MSB=0
        JZ OFF1             ;MSB = 0 go to OFF1
        JMP START           ;MSB = 1 go to START
OFF1:
        LDA 2000H
        ANI 80H             ;Check MSB of A
        CPI 80H             ;Check if MSB = 1
        JZ ON1              ;MSB = 1 go to ON1
        JMP OFF1            ;MSB = 0 go to OFF1
ON1:
        LDA 2000H           ;same as above
        ANI 80H
        CPI 00H
        JZ OFF2
        JMP ON1
OFF2:
        LDA 2000H
        ANI 80H
        CPI 80H
        JZ ON2
        MVI A,00H
        STA 3000H           ;Open all leds
        CALL DELB           ;Delay for 0.2sec
        DCR D               ;Decrease Counter
        MOV A,D             ;
        CPI 00H             ;If MSB switch doesn't change do this D times
        JNZ OFF2            ;Else go to OFF2
        MVI A,FFH           ;turn off leds
        STA 3000H
        MVI D,64H           ;Reset D
        JMP OFF1            ;Go to OFF1 (Start again)
ON2:
        LDA 2000H
        ANI 80H             ;If MSB switch changes again
        CPI 00H             ;
        JZ STARTAGAIN       ;Go to STARTAGAIN to reset counter and wait  2 secs again
        MVI A,00H           ;Else continue as before
        STA 3000H
        CALL DELB
        DCR D
        MOV A,D
        CPI 00H
        JNZ ON2             ;Loop until D = 0 or MSB switch change
        MVI A,FFH           ;Turn off leds
        STA 3000H
```

```
        MVI D,64H               ;Reset D
        JMP OFF1                ;Go again from the beginning

STARTAGAIN:
        MVI D,64H               ;MSB switch Changes so we reset D
        JMP OFF2

        END
```

Άσκηση 3

Ερώτημα Α

Πρόγραμμα 3i σε Assembly

```
START:
        LDA 2000H               ;load in A the input
        MVI D,01H               ;this is a helpful counter
SCAN:
        CPI 00H                 ;if A = 00H
        JZ ALLZERO              ;go to ALLZERO
        RRC                     ;else rotate right
        JC FOUND                ;If LSB of A = 1 go to FOUND
        MOV B,A
        MOV A,D
        RLC                     ;rotate D left
        MOV D,A
        MOV A,B
        JMP SCAN
ALLZERO:
        MVI A,FFH               ;all leds off
        STA 3000H
        JMP START
FOUND:
        MOV A,D                 ;found correct led
        CMA
        STA 3000H               ;store number and show correct led
        JMP START               ;go to read new input

        END
```

Ερώτημα Β

Πρόγραμμα 3ii σε Assembly

```
START:
        CALL KIND               ;input from keyboard
        CPI 00H                 ;for input 0 error
        JZ ERROR
        CPI 09H                 ;for input >= 9 error
        JNC ERROR
        MVI D,FFH
SCAN:
        DCR A
```

```
        MOV B,A                 ;B counts the rotations of D
        MOV A,D
        STC                     ;CY = 1
        CMC                     ;CY = 0
        RAL                     ;rotate left
        MOV D,A
        MOV A,B
        JZ FOUND                ;if A = 0 go to FOUND
        JMP SCAN                ;else go to SCAN
ERROR:
        MVI A,FFH               ;error
        STA 3000H               ;turn off all leds
        JMP START
FOUND:
        MOV A,D
        STC
        RAR
        CMA
        STA 3000H               ;store the correct value
        JMP START               ;go from start to read new input

        END
```

Ερώτημα Γ

Πρόγραμμα 3iii σε Assembly

```
        IN 10H
        LXI H,08FAH             ;addresses for the messages 08FAH-08FFH
        MVI M,10H               ;clear the screen
        LXI H,08FBH
        MVI M,10H
        LXI H,08FCH
        MVI M,10H
        LXI H,08FDH
        MVI M,10H
        LXI H,08FEH
        MVI M,10H
        LXI H,08FFH
        MVI M,10H

SCAN3:
        MVI A,F7H               ;line3 - keys 1,2,3
        STA 2800H               ;scan
        LDA 1800H               ;read
        MVI B,07H
        ANA B

        CPI 06H                 ;column 0
        JZ KEY1                 ;check for key 1
        CPI 05H                 ;column 1
        JZ KEY2                 ;check for key 2
        CPI 03H                 ;column 2
        JZ KEY3                 ;check for key 3
```

```
        CALL SHOW
        JMP SCAN1
KEY3:                           ;show key 3
        LXI H,08FFH
        MVI M,00H               ;0
        LXI H,08FEH
        MVI M,03H               ;3
        CALL SHOW
        JMP SCAN1
KEY2:                           ;show key 2
        LXI H,08FFH
        MVI M,00H               ;0
        LXI H,08FEH
        MVI M,02H               ;2
        CALL SHOW
        JMP SCAN1
KEY1:                           ;show key 1
        LXI H,08FFH
        MVI M,00H               ;0
        LXI H,08FEH
        MVI M,01H               ;1
        CALL SHOW

SCAN1:
        MVI A,FDH               ;line1 - keys FETCH ADRS, FETCH REG, RUN
        STA 2800H
        LDA 1800H
        MVI B,07H
        ANA B

        CPI 03H                 ;column 2
        JZ KEYFETCHADRS         ;check for key FETCH ADRS
        CPI 05H                 ;column 1
        JZ KEYFETCHREG          ;check for key FETCH REG
        CPI 06H                 ;column 0
        JZ KEYRUN               ;check for key RUN

        CALL SHOW
        JMP SCAN6
KEYFETCHADRS:                   ;show FETCH ADRS
        LXI H,08FFH
        MVI M,08H               ;8
        LXI H,08FEH
        MVI M,02H               ;2
        CALL SHOW
        JMP SCAN6
KEYFETCHREG:                    ;show FETCH REG
        LXI H,08FFH
        MVI M,08H               ;8
        LXI H,08FEH
        MVI M,00H               ;0
        CALL SHOW
        JMP SCAN6
KEYRUN:                         ;show RUN
        LXI H,08FFH
```

```
        MVI M,08H               ;8
        LXI H,08FEH
        MVI M,04H               ;4
        CALL SHOW

SCAN6:
        MVI A,BFH               ;line6 - keys A, B, C
        STA 2800H
        LDA 1800H
        MVI B,07H
        ANA B

        CPI 03H
        JZ KEYC                 ;check for key C
        CPI 05H
        JZ KEYB                 ;check for key B
        CPI 06H
        JZ KEYA                 ;check for key A

        CALL SHOW
        JMP SCAN3
KEYC:                           ;show key C
        LXI H,08FFH
        MVI M,00H               ;0
        LXI H,08FEH
        MVI M,0CH               ;C
        CALL SHOW
        JMP SCAN3
KEYB:                           ;show key B
        LXI H,08FFH
        MVI M,00H               ;0
        LXI H,08FEH
        MVI M,0BH               ;B
        CALL SHOW
        JMP SCAN3
KEYA:                           ;show key A
        LXI H,08FFH
        MVI M,00H               ;0
        LXI H,08FEH
        MVI M,0AH               ;A
        CALL SHOW
        JMP SCAN3

SHOW:                           ;show on screen
        LXI D,08FAH
        CALL STDM
        CALL DCD
        RET

        END
```

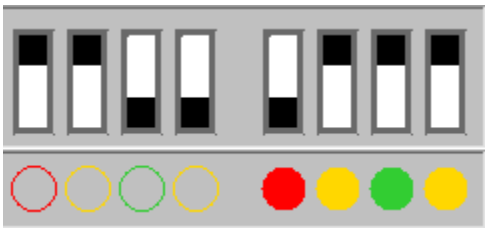
Άσκηση 4

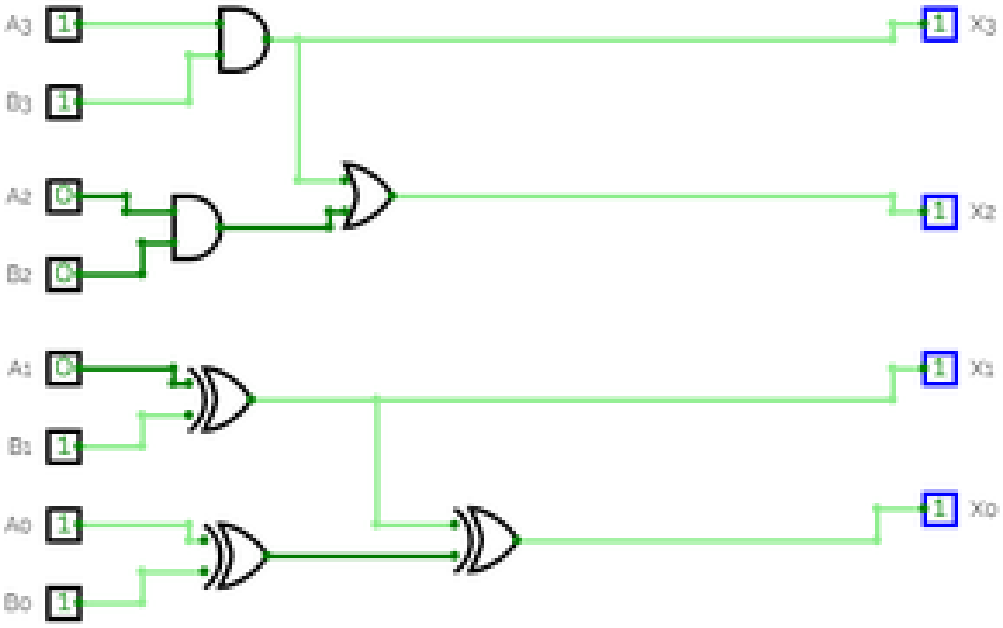
Πρόγραμμα 4 σε Assembly

```
START:
        MVI D,00H               ;LEDS

                                ;X3
        LDA 2000H               ;A3
        ANI 80H                 ;10000000
        RRC
        MOV B,A
        LDA 2000H               ;B3
        ANI 40H                 ;01000000
        ANA B                   ;A3 KAND B3
        MOV C,A
        RRC
        RRC
        RRC
        MOV D,A                 ;SAVE X3

                                ;X2
        LDA 2000H               ;A2
        ANI 20H                 ;00100000
        RRC
        MOV B,A
        LDA 2000H               ;B2
        ANI 10H                 ;00010000
        ANA B                   ;A2 AND B2
        RLC
        RLC                     ;01000000
        ORA C                   ;(A3 AND B3) OR (A2 AND B2)
        RRC
        RRC
        RRC
        RRC
        ORA D                   ;SUM
        MOV D,A                 ;SAVE X3,X2
                                ;X1
        LDA 2000H               ;A1
        ANI 08H                 ;00001000
        RRC
        MOV B,A
        LDA 2000H               ;B1
        ANI 04H                 ;00000100
        XRA B                   ;A1 XOR B1
        MOV C,A
        RRC
        ORA D                   ;SUM
        MOV D,A                 ;SAVE X3,X2,X1
                                ;X0
        LDA 2000H               ;A0
        ANI 02H                 ;00000010
        RRC
        MOV B,A
        LDA 2000H               ;B0
```

```
ANI 01H                 ;00000001
XRA B                   ;A0 X0R B0
RLC
RLC
XRA C                   ;(A1 XOR B1) XOR (A0 XOR B0)
RRC
RRC
ORA D                   ;SAVE X3,X2,X1,X0

CMA
STA 3000H
JMP START

END
```

Παράδειγμα για είσοδο 11000111:





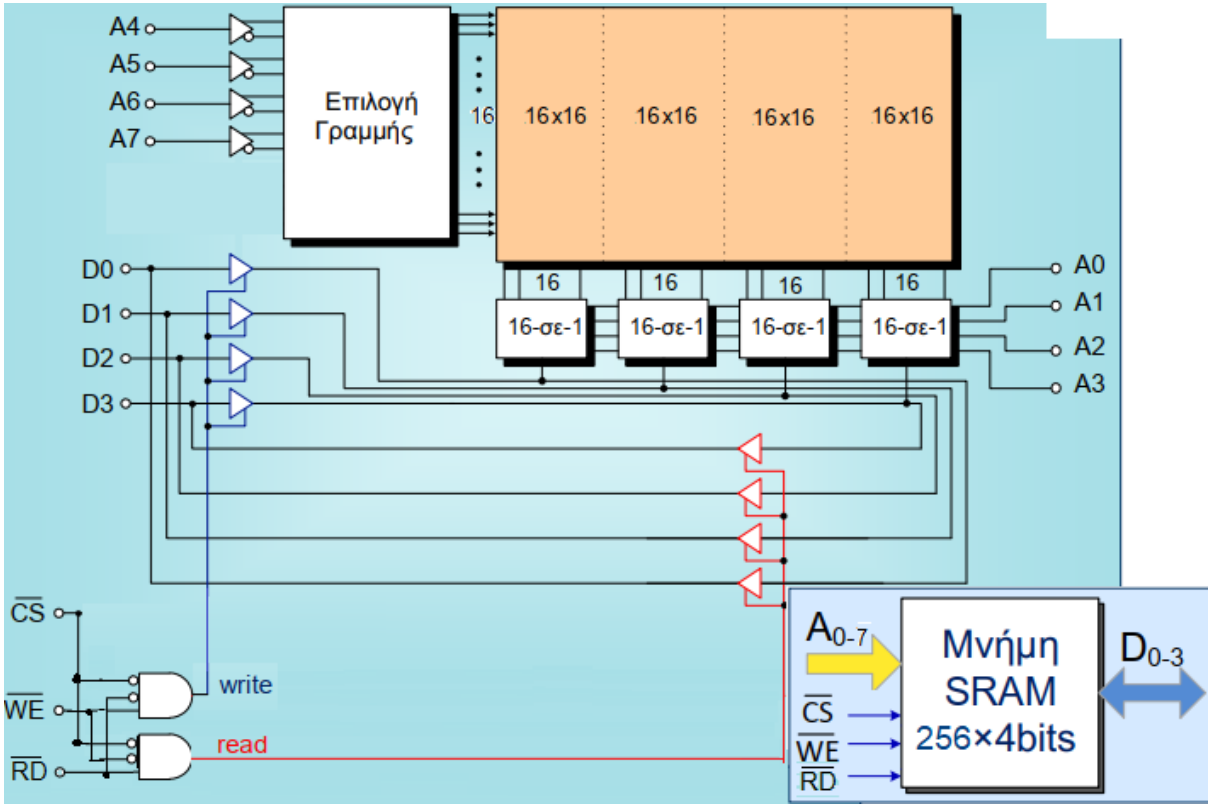
Άσκηση 5

Παρακάτω παρουσιάζεται η εσωτερική οργάνωση μιας μνήμη SRAM 256×4 bit, η οποία προέκυψε από τροποποίηση του σχήματος 3.2 του βιβλίου θεωρίας (Συστήματα Μικροϋπολογιστών, *Κ. Πεκμεστζή*, Εκδόσεις Συμμετρία, 1995).

Αρχικά, τα σήματα CS, WE και RD είναι αντίστροφης λογικής, οπότε πρέπει να πάρουν την λογική τιμή «0» προκειμένου να εκτελεστεί η λειτουργία που θέλουμε. Το CS πρακτικά ενεργοποιεί τη λειτουργία της μνήμης, το WE μας επιτρέπει να γράψουμε στην μνήμη, ενώ το RD μας επιτρέπει την ανάγνωση από αυτή. Η έξοδος read και write από την υλοποίηση που φαίνεται στο σχήμα δεν μπορούν να είναι ταυτόχρονα «1».

Τα A0-A7 μας δίνουν τη διεύθυνση της μνήμης που θα διαβάσουμε ή θα γράψουμε αντίστοιχα. Πιο συγκεκριμένα τα τέσσερα LSB bits A0-A3 επιλέγουν την επιθυμητή στήλη από κάθε έναν από τους 4 «πίνακες» (16 πιθανές τετράδες), ενώ τα 4 MSB (A4-A7) επιλέγουν την γραμμή των πινάκων και σε συνδυασμό με τα LSB μας δίνουν 4 μοναδικές θέσεις (μια για κάθε πίνακα).

Στη περίπτωση που κάνει read ενεργοποιούνται οι μπλε πύλες και φορτώνονται στα D0-D3 τα δεδομένα που δείχνει η διεύθυνση A0-A7. Από την άλλη στην περίπτωση που κάνει write ενεργοποιούνται οι κόκκινες πύλες και αποθηκεύονται τα δεδομένα D0-D3 στη διεύθυνση που ορίζουν τα A0-A7.



Άσκηση 6

Από την εκφώνηση μας ζητείται να σχεδιάσουμε ένα σύστημα μνήμης που να περιλαμβάνει τα εξής:

- 8Kbytes ROM (2 ολοκληρωμένα μνήμης των 2Kx8 bit (ROM) και 1 ολοκληρωμένο μνήμης 4Kx8 bit (ROM)
- 4Kbytes RAM (2 ολοκληρωμένα μνήμης των 2Kx8 SRAMs)

Διευθύνσεις στη μνήμη

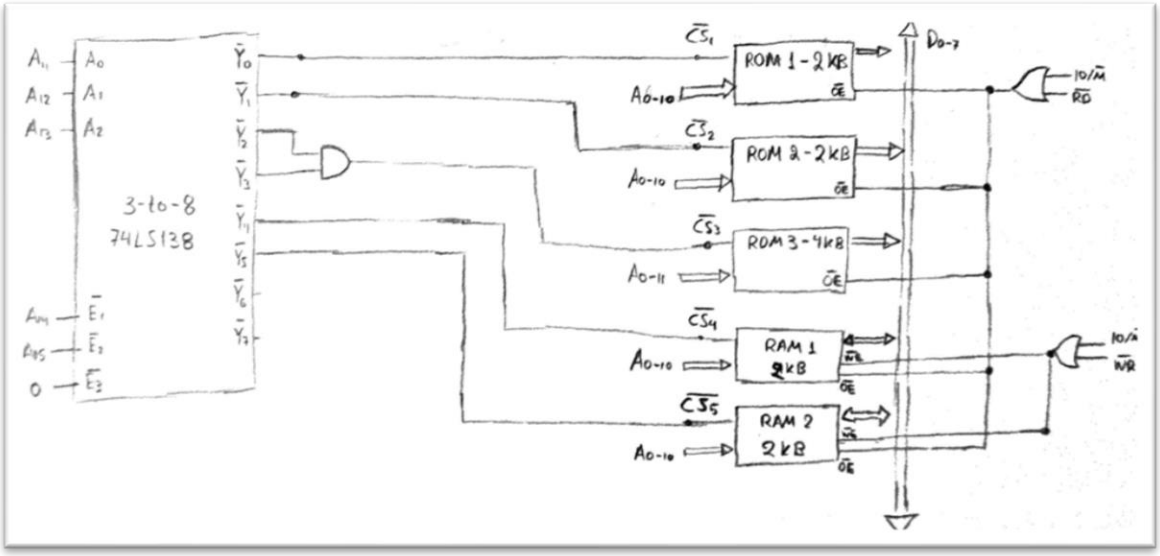
ROM 1	**0000 – 07FF**
ROM 2	**0800 – 0FFF**
ROM 3	**1000 – 1FFF**
RAM 1	**2000 – 27FF**
RAM 2	**2800 – 2FFF**

Χάρτης Μνήμης

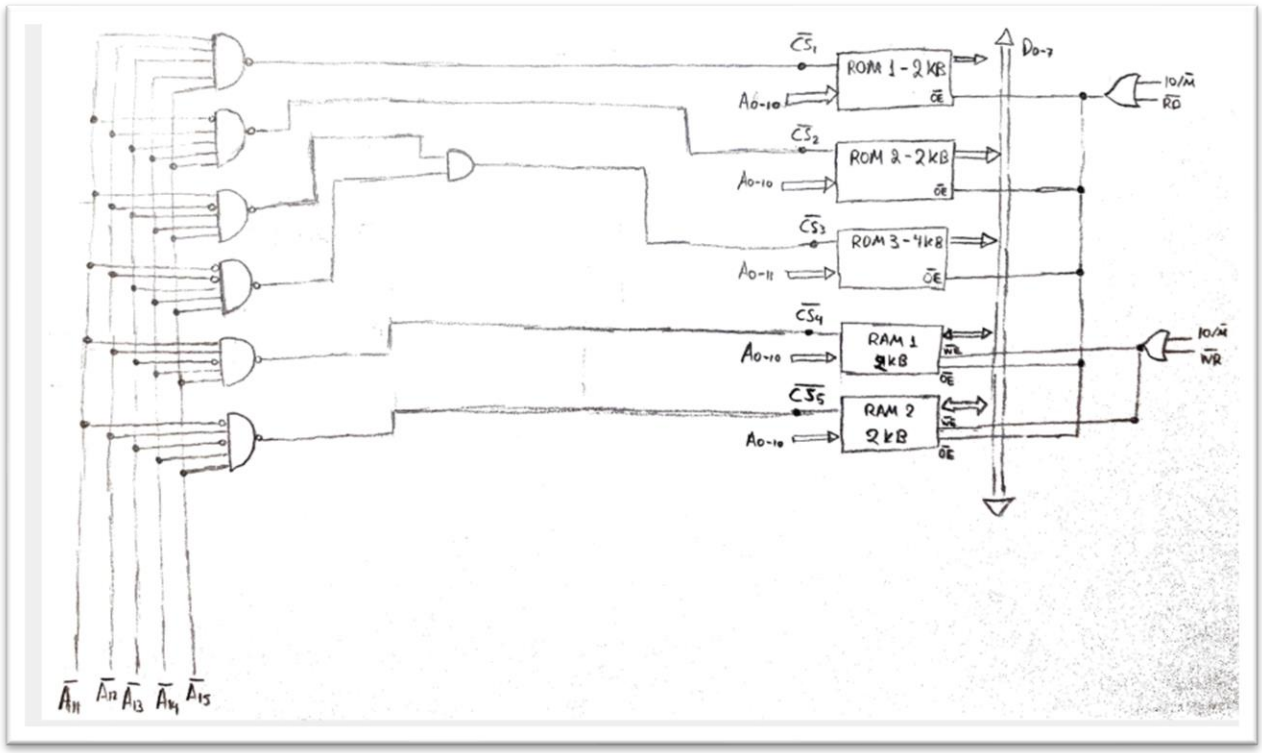
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Memory
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000	ROM 1 – 2K
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	07FF	
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0800	ROM 2 – 2K
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	0FFF	
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1000	ROM 3 – 4K
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1FFF	
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	2000	RAM 1 – 2K
0	0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	27FF	
0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	2800	RAM 2 – 2K
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	2FFF	

Λογικό Διάγραμμα

a) Αποκωδικοποιητής 3:8 (74LS138) και λογικές πύλες



b) Μόνο λογικές πύλες



Άσκηση 7

Από την εκφώνηση μας ζητείται να σχεδιάσουμε ένα μΥ-Σ 8085 που να έχει τον εξής χάρτη μνήμης:

0000-2FFF Hex : ROM (12Kbytes)
3000-5FFF Hex : RAM (12Kbytes)
6000-6FFF Hex : ROM (4Kbytes)
7000 Hex : θύρα εξόδου (Memory map I/O)
70 Hex : θύρα εισόδου (Standard I/O)

Χάρτης Μνήμης

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	Address	Memory
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0000	ROM – 16K
0	0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	2FFF	
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	6000	
0	1	1	0	1	1	1	1	1	1	1	1	1	1	1	1	6FFF	
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	3000	RAM 1 – 4K
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	3FFF	
0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	4000	RAM 2 – 4K
0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	4FFF	
0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	5000	RAM 3 – 4K
0	1	0	1	1	1	1	1	1	1	1	1	1	1	1	1	5FFF	

Λογικό Διάγραμμα