2η ΣΕΙΡΑ ΑΣΚΗΣΕΩΝ ΣΥΣΤΗΜΑΤΑ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΩΝ

ΣΗΜΜΥ 12-05-2020

Αϊβαλής Θεόδωρος 03117099

Σαλιαράκης Παύλος 03117135

ΑΣΚΗΣΗ 1η:

Θέλουμε να γραφεί σε assembly που να επιτελεί τις παρακάτω λειτουργίες:

(α) Να αποθηκευθούν οι αριθμοί 0-255 με φθίνουσα σειρά στις διαδοχικές θέσεις της μνήμης με αρχή τη διεύθυνση 0900 Η.

(β) Να υπολογίσουμε τον αριθμό των μηδενικών των παραπάνω δεδομένων. Το αποτέλεσμα να αποθηκευτεί στον διπλό καταχωρητή DE.

(γ) Να υπολογίσουμε το πλήθος από τους παραπάνω αριθμούς (0-255) που είναι μεταξύ των αριθμών 20Η και 70Η περιλαμβανομένων και να φυλάξουμε το αποτέλεσμα στον καταχωρητή C.

```
START:
ERWTHMA_A:
IN 10H          ;ΑΡΣΗ ΠΡΟΣΤΑΣΙΑΣ ΜΝΗΜΗΣ
LXI H,0900H     ;ΦΟΡΤΩΣΗ ΔΙΕΥΘΥΝΣΗΣ 0900 ΣΤΟΝ H-L
MVI B,00H       ;ΑΡΧΙΚΟΠΟΙΗΣΗ ΚΑΤΑΧΩΡΗΤΩΝ
MVI A,00H

MEM_NUMS:
MOV M,B         ;ΦΟΡΤΩΣΗ ΣΤΗ ΜΝΗΜΗ ΤΟΥ ΚΑΤΑΛΛΗΛΟΥ ΑΡΙΘΜΟΥ
INR A           ;ΑΥΞΑΝΩ ΤΟ ΜΕΤΡΗΤΗ Α
CPI 00H         ;ΕΛΕΓΧΟΣ ΑΝ ΕΦΤΑΣΑ ΣΤΟ 255
JZ ERWTHMA_B      ;ΑΝ ΝΑΙ ΠΑΩ ΣΤΟ ΕΡΩΤΗΜΑ
INX H           ;ΑΥΞΑΝΩ ΤΟ ΔΙΠΛΟ ΚΑΤΑΧΩΡΗΤΗ H-L ΓΙΑ ΝΑ ΠΑΡΩ ΤΗΝ ΕΠΟΜΕΝΗ ΘΕΣΗ
ΜΝΗΜΗΣ
INR B           ;ΑΥΞΑΝΩ ΤΟΝ ΑΡΙΘΜΟ ΜΟΥ
JMP MEM_NUMS

ERWTHMA_B:
LXI H,0900H     ;ΠΗΓΑΙΝΩ ΠΑΛΙ ΣΤΗ ΘΕΣΗ ΜΝΗΜΗΣ ΠΟΥ ΕΧΕΙ ΤΟΝ ΠΡΩΤΟ ΑΡΙΘΜΟ
MVI B,08H       ;ΚΑΤΑΧΩΡΩ ΣΤΟΝ Β ΤΟ 8 (ΔΕΚΑΔΙΚΟ) ΓΙΑΤΙ ΚΑΘΕ ΑΡΙΘΜΟΣ
ΑΝΑΠΑΡΙΣΤΑΤΑΙ ΜΕ 8 BIT
MVI D,00H       ;ΜΗΔΕΝΙΖΩ ΤΟΥΣ ΚΑΤΑΧΩΡΗΤΕΣ
MVI E,00H

COUNT_ZEROS:
MOV A,M         ;ΚΑΤΑΧΩΡΩ ΣΤΟΝ Α ΤΟΝ ΑΡΙΘΜΟ ΑΠΟ ΤΗ ΜΝΗΜΗ

BIT_CHECK:
MOV C,A         ;ΠΡΟΣΩΡΙΝΟ ΣΩΣΙΜΟ ΤΟΥ Α
MOV A,B         ;ΜΕΤΑΦΕΡΩ ΣΤΟΝ Α ΤΟΝ ΜΕΤΡΗΤΗ
CPI 00H         ;ΕΛΕΓΧΩ ΑΝ ΤΕΛΕΙΩΣΑΝ ΤΑ ΨΗΦΙΑ ΤΟΥ ΕΚΑΣΤΟΤΕ ΑΡΙΘΜΟΥ
JZ NEXT_CHECK     ;ΑΝ ΤΕΛΕΙΩΣΑΝ ΠΑΩ ΣΤΟΝ ΕΠΟΜΕΝΟ
MOV A,C
RAL             ;ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΤΟΥ ΑΡΙΘΜΟΥ
DCR B           ;ΜΕΙΩΣΗ ΤΟΥ ΜΕΤΡΗΤΗ
JC BIT_CHECK    ;ΕΛΕΓΧΟΣ ΜΕΣΩ ΤΗΣ ΣΗΜΑΙΑΣ ΚΡΑΤΟΥΜΝΟΥ ΓΙΑ 0 Ή 1
INX D           ;ΑΥΞΗΣΗ ΤΟΥ ΖΕΥΓΟΥΣ ΚΑΤΑΧΩΡΗΤΩΝ D-E
```

```
JMP BIT_CHECK


NEXT_CHECK:
INX H           ;ΑΥΞΗΣΗ ΤΟΥ ΖΕΥΓΟΥΣ ΚΑΤΑΧΩΡΗΤΩΝ H-L
MOV A,H
CPI 0AH           ;ΕΛΕΓΧΟΣ ΓΙΑ ΤΟ ΑΝ ΕΧΕΙ ΞΕΠΕΡΑΣΕΙ ΤΗ ΤΕΛΕΥΤΑΙΑ ΘΕΣΗ ΜΝΗΜΗΣ
ΠΟΥ ΕΧΟΥΝ ΑΠΟΘΗΚΕΥΤΕΙ ΟΙ ΑΡΙΘΜΟΙ (09FF)
JZ ERWTHMA_C      ;ΑΝ ΝΑΙ ΠΗΓΑΙΝΩ ΣΤΟ ΕΠΟΜΕΝΟ ΕΡΩΤΗΜΑ
MVI B,08H         ;ΕΠΑΝΑΡΧΙΚΟΠΟΙΩ ΤΟΝ ΜΕΤΡΗΤΗ ΤΩΝ BITS
JMP COUNT_ZEROS

ERWTHMA_C:

LXI H,0900H     ;ΠΗΓΑΙΝΩ ΠΑΛΙ ΣΤΗ ΘΕΣΗ ΜΝΗΜΗΣ ΠΟΥ ΕΧΕΙ ΤΟΝ ΠΡΩΤΟ ΑΡΙΘΜΟ
MVI C,00H       ;ΑΡΧΙΚΟΠΟΙΗΣΗ ΤΟΥ ΜΕΤΡΗΤΗ

ENTOS_ORIWN:
MOV A,M         ;ΦΟΡΤΩΣΗ ΑΠΟ ΤΗ ΜΝΗΜΗ ΤΟΥ ΑΡΙΘΜΟΥ
CPI 20H
JC NEXT_NUM     ;ΑΝ Ο ΑΡΙΘΜΟΣ ΠΟΥ ΕΧΩ ΕΙΝΑΙ ΜΙΚΡΟΤΕΡΟΣ ΑΠΟ ΤΟ 20Η ΠΗΓΑΙΝΩ
ΣΤΟΝ ΕΠΟΜΕΝΟ
CPI 71H
JNC END_PROGRAM   ;ΑΝ Ο ΑΡΙΘΜΟΣ ΕΙΝΑΙ ΜΕΓΑΛΥΤΕΡΟΣ ΤΟΥ 70Η ΤΟΤΕ ΤΕΛΕΙΩΝΕΙ ΤΟ
ΠΡΟΓΡΑΜΜΑ
INR C           ;ΑΛΛΙΩΣ ΑΥΞΑΝΩ ΤΟΝ ΜΕΤΡΗΤΗ

NEXT_NUM:
INX H           ;ΑΥΞΗΣΗ ΤΟΥ ΖΕΥΓΟΥΣ ΚΑΤΑΧΩΡΗΤΩΝ H-L
MOV A,H
CPI 0AH         ;ΕΛΕΓΧΟΣ ΓΙΑ ΤΟ ΑΝ ΕΧΕΙ ΞΕΠΕΡΑΣΕΙ ΤΗ ΤΕΛΕΥΤΑΙΑ ΘΕΣΗ ΜΝΗΜΗΣ
ΠΟΥ ΕΧΟΥΝ ΑΠΟΘΗΚΕΥΤΕΙ ΟΙ ΑΡΙΘΜΟΙ (09FF)
JZ END_PROGRAM     ;ΑΝ ΝΑΙ ΤΕΛΕΙΩΝΕΙ ΤΟ ΠΡΟΓΡΑΜΜΑ
JMP ENTOS_ORIWN   ;ΑΛΛΙΩΣ ΠΗΓΑΙΝΩ ΣΤΟΝ ΕΠΟΜΕΝΟ ΑΡΙΘΜΟ

END_PROGRAM:
RST 1
END
```

ΑΣΚΗΣΗ 2η:

Θέλουμε να γραφεί πρόγραμμα σε Assembly, που όταν το LSB της θύρας εισόδου dip switch (θέση μνήμης 2000 Hex) από OFF γίνει ON και ξανά OFF τότε να αναβοσβήνουν (σε ορατή συχνότητα) όλα τα LED της πόρτας εξόδου για περίπου 15 sec και μετά να σβήνουν. Αν ενδιάμεσα ενεργοποιηθεί πάλι το push-button (OFF - ON - OFF το LSB των dip switch) να ανανεώνεται ο χρόνος των 15 sec.

Ο κώδικας φαίνεται παρακάτω:

```
START:
MVI D,FFH
MOV A,D
STA 3000H           ; ΑΡΧΙΚΑ ΟΛΑ ΣΒΗΣΤΑ
MVI E,4BH           ; E = 75 (ΜΕΤΡΗΤΗΣ ΓΙΑ 15sec)
LXI B,00C8H         ; BC = 200ms
MAIN:
CALL GET_LSB         ; ΚΑΛΩ ΤΗ ΡΟΥΤΙΝΑ ΓΙΑ ΝΑ ΠΑΡΩ ΤΟ LSB
CPI 00H              ; ΠΕΡΙΜΕΝΩ ΤΟ ΠΡΩΤΟ OFF
JZ FIRST_OFF         ; ΑΝ ΕΡΘΕΙ ΠΑΩ ΣΤΗΝ FIRST_OFF (OFF)
CALL DELB            ; ΑΛΛΙΩΣ ΕΦΑΡΜΟΖΩ ΤΗΝ ΚΑΘΥΣΤΕΡΗΣΗ
JMP MAIN             ; ΚΑΙ ΠΕΡΙΜΕΝΩ ΜΕΧΡΙ ΤΟ 1ο OFF

FIRST_OFF:
CALL GET_LSB         ; ΚΑΛΩ ΤΗ ΡΟΥΤΙΝΑ ΓΙΑ ΝΑ ΠΑΡΩ ΤΟ LSB
CPI 01H              ; ΠΕΡΙΜΕΝΩ ΤΟ ΟΝ ΜΕΤΑ ΑΠΟ ΤΟ OFF
JZ FIRST_ON          ; ΑΝ ΕΡΘΕΙ ΠΑΩ ΣΤΗΝ FIRST_ON (OFF-ON)
CALL DELB            ; ΑΛΛΙΩΣ ΕΦΑΡΜΟΖΩ ΤΗΝ ΚΑΘΥΣΤΕΡΗΣΗ
JMP FIRST_OFF        ; ΚΑΙ ΠΕΡΙΜΕΝΩ ΜΕΧΡΙ ΤΟ 1ο ON

FIRST_ON:
CALL GET_LSB         ; ΚΑΛΩ ΤΗ ΡΟΥΤΙΝΑ ΓΙΑ ΝΑ ΠΑΡΩ ΤΟ LSB
CPI 00H              ; ΠΕΡΙΜΕΝΩ ΤΟ 2ο OFF
JZ NEXT_OFF          ; ΑΝ ΕΡΘΕΙ ΠΑΩ ΣΤΗΝ NEXT_OFF (OFF-ON-OFF)
CALL DELB            ; ΑΛΛΙΩΣ ΕΦΑΡΜΟΖΩ ΤΗΝ ΚΑΘΥΣΤΕΡΗΣΗ
JMP FIRST_ON         ; ΚΑΙ ΠΕΡΙΜΕΝΩ ΤΟ 2ο ON

NEXT_OFF:
CALL GET_LSB         ; ΚΑΛΩ ΤΗ ΡΟΥΤΙΝΑ ΓΙΑ ΝΑ ΠΑΡΩ ΤΟ LSB
CPI 01H              ; ΠΕΡΙΜΕΝΩ ΤΟ ΟΝ ΜΕΤΑ ΤΟ OFF
JZ NEXT_ON           ; ΑΝ ΕΡΘΕΙ ΠΑΩ ΣΤΗΝ NEXT_ON (OFF-ON-OFF-ON)
CALL LABEL           ; ΑΛΛΙΩΣ ΚΑΛΩ ΤΗΝ LABEL
CALL DELB            ; ΕΦΑΡΜΟΖΩ ΤΗΝ ΚΑΘΥΣΤΕΡΗΣΗ
DCR E                ; ΚΑΙ ΜΕΙΩΝΩ ΤΟ ΧΡΟΝΟΜΕΤΡΟ ΚΑΤΑ 200ms
MOV A,E
CPI 00H              ; ΑΝ ΔΕΝ ΕΧΟΥΝ ΠΕΡΑΣΕΙ ΤΑ 15 secs
JNZ NEXT_OFF         ; ΕΠΑΝΑΛΑΜΒΑΝΩ ΚΑΙ ΠΕΡΙΜΕΝΩ ΝΑ ΠΕΡΑΣΟΥΝ
CALL TURN_OFF        ; ΑΛΛΙΩΣ ΚΑΛΩ ΤΗ ΡΟΥΤΙΝΑ ΠΟΥ ΣΒΗΝΕΙ ΤΑ ΛΑΜΠΑΚΙΑ
JMP MAIN             ; ΚΑΙ ΕΠΙΣΤΡΕΦΩ ΣΤΗΝ MAIN

NEXT_ON:
CALL GET_LSB         ; ΚΑΛΩ ΤΗ ΡΟΥΤΙΝΑ ΓΙΑ ΝΑ ΠΑΡΩ ΤΟ LSB
CPI 00H              ; ΠΕΡΙΜΕΝΩ ΤΟ 3ο OFF
JZ ANOTHER_OFF       ; ΑΝ ΕΡΘΕΙ ΠΑΩ ΣΤΗΝ ANOTHER_OFF
CALL LABEL           ; ΑΛΛΙΩΣ ΚΑΛΩ ΤΗΝ LABEL
CALL DELB            ; ΕΦΑΡΜΟΖΩ ΤΗΝ ΚΑΘΥΣΤΕΡΗΣΗ
DCR E                ; ΚΑΙ ΜΕΙΩΝΩ ΤΟ ΧΡΟΝΟΜΕΤΡΟ ΚΑΤΑ 200ms
MOV A,E
CPI FFH
JNZ NEXT_ON          ; ΕΠΑΝΑΛΑΜΒΑΝΩ ΚΑΙ ΠΕΡΙΜΕΝΩ ΝΑ ΠΕΡΑΣΟΥΝ
CALL TURN_OFF        ; ΑΛΛΙΩΣ ΚΑΛΩ ΤΗ ΡΟΥΤΙΝΑ ΠΟΥ ΣΒΗΝΕΙ ΤΑ ΛΑΜΠΑΚΙΑ
JMP MAIN

ANOTHER_OFF:
MVI E,4BH            ; ΕΠΑΝΑΘΕΤΩ ΤΟ ΧΡΟΝΟΜΕΤΡΟ ΣΤΑ 15sec
```

```
JMP NEXT_OFF

LABEL:
MOV A,D
CMA
MOV D,A
STA 3000H
RET

TURN_OFF:           ; ΡΟΥΤΙΝΑ ΓΙΑ ΝΑ ΣΒΗΣΩ ΟΛΑ ΤΑ ΛΑΜΠΑΚΙΑ
MVI A,FFH
STA 3000H
RET

GET_LSB:            ; ΡΟΥΤΙΝΑ ΓΙΑ ΝΑ ΠΑΡΩ ΤΟ LSB
LDA 2000H
ANI 01H
RET

END
```

Αρχικά για να αναβοσβήνουν τα λαμπάκια θα πρέπει να δώσουμε OFF-ON-OFF και μετά κάθε φορά που δέχεται το σύστημα ON-OFF ανάβουν πάλι μηδενίζοντας το χρονόμετρο.

ΑΣΚΗΣΗ 3η:

Θέλουμε να γραφούν σε assembly 8085 και να εκτελεστούν στο μLAB τα 3 παρακάτω προγράμματα:

(i) Διαβάζουμε την πόρτα εισόδου των dip switches και με βάση το 1ο αριστερότερο ON, ανάβει το αντίστοιχης τάξης led και όλα τα υψηλότερης τάξης led μετά από αυτό.

```
START:

LDA 2000H       ;ΦΟΡΤΩΝΩ ΤΗΝ ΕΙΣΟΔΟ
RAL             ;ΚΑΝΩ ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΚΑΙ ΕΛΕΓΧΩ ΒΙΤ 7
JC LIGHT_1      ;ΑΝ BIT 7=1 ΑΝΑΒΕΙ ΤΟ(10000000)
RAL             ;ΚΑΝΩ ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΚΑΙ ΕΛΕΓΧΩ ΒΙΤ 6
JC LIGHT_2      ;ΑΝ BIT 6=1 ΑΝΑΒΕΙ ΤΟ(11000000)
RAL             ;ΚΑΝΩ ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΚΑΙ ΕΛΕΓΧΩ ΒΙΤ 5
JC LIGHT_3      ;ΑΝ BIT 5=1 ΑΝΑΒΕΙ ΤΟ(11100000)
RAL             ;ΚΑΝΩ ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΚΑΙ ΕΛΕΓΧΩ ΒΙΤ 4
JC LIGHT_4      ;ΑΝ BIT 4=1 ΑΝΑΒΕΙ ΤΟ(11110000)
RAL             ;ΚΑΝΩ ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΚΑΙ ΕΛΕΓΧΩ ΒΙΤ 3
JC LIGHT_5      ;ΑΝ BIT 3=1 ΑΝΑΒΕΙ ΤΟ(11111000)
RAL             ;ΚΑΝΩ ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΚΑΙ ΕΛΕΓΧΩ ΒΙΤ 2
JC LIGHT_6      ;ΑΝ BIT 2=1 ΑΝΑΒΕΙ ΤΟ(11111100)
RAL             ;ΚΑΝΩ ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΚΑΙ ΕΛΕΓΧΩ ΒΙΤ 1
JC LIGHT_7      ;ΑΝ BIT 1=1 ΑΝΑΒΕΙ ΤΟ(11111110)
RAL             ;ΚΑΝΩ ΑΡΙΣΤΕΡΗ ΟΛΙΣΘΗΣΗ ΚΑΙ ΕΛΕΓΧΩ ΒΙΤ 0
JC LIGHT_8      ;ΑΝ BIT 0=1 ΑΝΑΒΕΙ ΤΟ(11111111)
MVI A,FFH
STA 3000H
JMP START       ;ΑΝ ΚΑΝΕΝΑ ΔΕΝ ΕΙΝΑΙ ΑΣΣΟΣ ΚΑΝΕΙ ΤΙΣ 3 ΑΥΤΕΣ ΕΝΤΟΛΕΣ


LIGHT_1:        ; ΑΝΑΒΕΙ ΤΟ (10000000)
MVI A,7FH
STA 3000H
JMP START

LIGHT_2:        ; ΑΝΑΒΕΙ ΤΟ (11000000)
MVI A,3FH
STA 3000H
JMP START

LIGHT_3:        ; ΑΝΑΒΕΙ ΤΟ (11100000)
MVI A,1FH
STA 3000H
JMP START

LIGHT_4:        ; ΑΝΑΒΕΙ ΤΟ (11110000)
MVI A,0FH
STA 3000H
JMP START

LIGHT_5:        ; ΑΝΑΒΕΙ ΤΟ (11111000)
MVI A,07H
STA 3000H
JMP START

LIGHT_6:        ; ΑΝΑΒΕΙ ΤΟ (11111100)
MVI A,03H
STA 3000H
JMP START

LIGHT_7:        ; ΑΝΑΒΕΙ ΤΟ (11111110)
MVI A,01H
STA 3000H
```

```
JMP START

LIGHT_8:        ; ΑΝΑΒΕΙ ΤΟ (11111111)
MVI A,00H
STA 3000H
JMP START


END
```

(ii) Να αναμένει το πάτημα του δεκαεξαδικού πληκτρολογίου και μόνο των αριθμών 1 έως 8. Για αριθμούς από 1- 4 να αναβοσβήνουν 4 φορές και τα 4 LSB led ενώ για αριθμούς από 5- 8 να αναβοσβήνουν επίσης 4 φορές και τα 4 MSB led. Γίνεται χρήση της ρουτίνας KIND.

```
MVI D,0FH
START:
MVI E,04H           ; ΒΑΖΩ ΤΟ 4 ΣΤΟΝ Ε ΓΙΑ ΤΟΝ ΜΕΤΡΗΤΗ
LXI B,0064H         ; ΓΙΑ ΤΗΝ ΚΑΘΥΣΤΕΡΗΣΗ
CALL KIND           ; ΔΙΑΒΑΖΩ ΑΠΟ ΤΟ ΠΛΗΚΤΡΟΛΟΓΙΟ
CPI 01H             ; 01
JZ LAMP_1           ; ΑΝ ΕΙΝΑΙ 01 ΠΑΩ ΝΑ ΑΝΑΨΩ ΤΑ 01-04
CPI 02H             ; 02
JZ LAMP_1           ; ΑΝ ΕΙΝΑΙ 02 ΠΑΩ ΝΑ ΑΝΑΨΩ ΤΑ 01-04
CPI 03H             ; 03
JZ LAMP_1           ; ΑΝ ΕΙΝΑΙ 03 ΠΑΩ ΝΑ ΑΝΑΨΩ ΤΑ 01-04
CPI 04H             ; 04
JZ LAMP_1           ; ΑΝ ΕΙΝΑΙ 04 ΠΑΩ ΝΑ ΑΝΑΨΩ ΤΑ 01-04
CPI 05H             ; 05
JZ LAMP_2           ; ΑΝ ΕΙΝΑΙ 05 ΠΑΩ ΝΑ ΑΝΑΨΩ ΤΑ 05-08
CPI 06H             ; 06
JZ LAMP_2           ; ΑΝ ΕΙΝΑΙ 06 ΠΑΩ ΝΑ ΑΝΑΨΩ ΤΑ 05-08
CPI 07H             ; 07
JZ LAMP_2           ; ΑΝ ΕΙΝΑΙ 07 ΠΑΩ ΝΑ ΑΝΑΨΩ ΤΑ 05-08
CPI 08H             ; 08
JZ LAMP_2           ; ΑΝ ΕΙΝΑΙ 08 ΠΑΩ ΝΑ ΑΝΑΨΩ ΤΑ 05-08

MVI A,FFH           ; ΑΝ ΔΕΝ ΕΧΩ ΠΑΤΗΣΕΙ ΤΙΠΟΤΑ ΤΑ ΑΦΗΝΩ ΟΛΑ ΣΒΗΣΤΑ
STA 3000H
JMP START



LAMP_1:             ; ΑΝΑΒΩ ΤΑ ΛΑΜΠΑΚΙΑ 01-04 (00001111)=(F)
MOV A,D
CMA
CALL DELB
CALL DELB
CALL DELB
CALL DELB
CALL DELB
STA 3000H           ; ΕΜΦΑΝΙΖΩ ΤΟ ΑΠΟΤΕΛΕΣΜΑ
JMP COUNTER_1       ; ΠΑΩ ΣΤΗΝ ΡΟΥΤΙΝΑ ΠΟΥΕΛΕΓΧΕΙ ΟΤΙ ΑΝΑΒΟΣΒΗΝΕΙ 4 ΦΟΡΕΣ



LAMP_2:             ; ΑΝΑΒΩ ΤΑ ΛΑΜΠΑΚΙΑ 05-08 (11110000)=(F0)
MOV A,D
CALL DELB           ; ΕΦΑΡΜΟΖΩ ΤΗΝ ΚΑΘΥΣΤΕΡΗΣΗ
CALL DELB
CALL DELB
CALL DELB
CALL DELB
STA 3000H           ;΄ΕΜΦΑΝΙΖΩ ΤΟ ΑΠΟΤΕΛΕΣΜΑ
JMP COUNTER_2       ; ΠΑΩ ΣΤΗΝ ΡΟΥΤΙΝΑ ΠΟΥ ΕΛΕΓΧΕΙ ΟΤΙ ΑΝΑΒΟΣΒΗΝΕΙ 4 ΦΟΡΕΣ
```

```
COUNTER_1:
MVI A,00H           ; ΣΒΗΝΩ ΟΛΑ ΤΑ ΛΑΜΠΑΚΙΑ
CMA
CALL DELB           ; ΕΦΑΡΜΟΖΩ ΤΗ ΚΑΘΥΣΤΕΡΗΣΗ
CALL DELB
CALL DELB
CALL DELB
CALL DELB
STA 3000H           ; ΕΜΦΑΝΙΖΩ ΤΗΝ ΕΞΟΔΟ
DCR E               ; ΜΕΙΩΝΩ ΤΟΝ ΜΕΤΡΗΤΗ
JZ START            ; ΑΝ ΕΙΝΑΙ ΜΗΔΕΝ ΕΠΙΣΤΡΕΦΩ ΣΤΗΝ ΑΡΧΗ
JMP LAMP_1          ; ΑΛΛΙΙΩΣ ΠΑΩ ΝΑ ΤΑ ΞΑΝΑΑΝΑΨΩ

COUNTER_2:
MVI A,00H           ; ΣΒΗΝΩ ΟΛΑ ΤΑ ΛΑΜΠΑΚΙΑ
CMA
CALL DELB           ; ΕΦΑΡΜΟΖΩ ΤΗΝ ΚΑΘΥΣΤΕΡΗΣΗ
CALL DELB
CALL DELB
CALL DELB
CALL DELB
STA 3000H           ; ΕΜΦΑΝΙΖΩ ΤΗΝ ΕΞΟΔΟ
DCR E               ; ΜΕΙΩΝΩ ΤΟΝ ΜΕΤΡΗΤΗ
JZ START            ; ΑΝ ΕΙΝΑΙ ΜΗΔΕΝ ΕΠΙΣΤΡΕΦΩ ΣΤΗΝ ΑΡΧΗ
JMP LAMP_2          ; ΑΛΛΙΩΣ ΠΑΩ ΚΑΙ ΤΑ ΞΑΝΑΑΝΑΒΩ
END
```

(iii)Να γίνει απευθείας ανάγνωση του πληκτρολογίου χωρίς τη χρήση της ρουτίνας KIND. Το αποτέλεσμα του κωδικού να εμφανίζεται στα 2 αριστερότερα 7-segment display με βάση τις ρουτίνες DCD και STDM.

```
IN 10H              ; DISABLE MEMORY PROTECTION
LXI H,0A02H         ; ΔΕΙΧΝΩ ΣΤΗΝ ΘΕΣΗ 0A02
MVI M,10H
INX H               ; ΔΕΙΧΝΩ ΣΤΟ ΕΠΟΜΕΝΟ ΚΕΛΙ ΤΗΣ ΜΝΗΜΗΣ
MVI M,10H
INX H               ; ΔΕΙΧΝΩ ΣΤΟ ΕΠΟΜΕΝΟ ΚΕΛΙ ΤΗΣ ΜΝΗΜΗΣ
MVI M,10H
INX H               ; ΔΕΙΧΝΩ ΣΤΟ ΕΠΟΜΕΝΟ ΚΕΛΙ ΤΗΣ ΜΝΗΜΗΣ
MVI M,10H



START:

LINE_0:             ; ΕΧΟΥΜΕ INSTR STEP, FETCH PC, HDWR STEP

MVI A,FEH
STA 2800H           ; ΔΙΑΒΑΖΩ ΓΡΑΜΜΗ
LDA 1800H           ; ΔΙΑΒΑΖΩ ΣΤΗΛΗ
ANI 07H             ; ΚΑΝΩ ΤΑ 5 MSB ΜΗΔΕΝ
MVI C,86H
CPI 06H             ; INSTR STEP
JZ DISPLAY
MVI C,85H
CPI 05H             ; FETCH PC
JZ DISPLAY
MVI C,F7H
CPI 03H             ; HDWR STEP
JZ DISPLAY


LINE_1:             ; ΕΧΟΥΜΕ RUN, FETCH REG, FETCH ADRS

MVI A,FDH
STA 2800H           ; ΔΙΑΒΑΖΩ ΓΡΑΜΜΗ
```

```
LDA 1800H          ; ΔΙΑΒΑΖΩ ΣΤΗΛΗ
ANI 07H            ; ΚΑΝΩ ΤΑ 5 MSB ΜΗΔΕΝ
MVI C,84H
CPI 06H            ; RUN
JZ DISPLAY
MVI C,80H
CPI 05H            ; FETCH REG
JZ DISPLAY
MVI C,82H
CPI 03H            ; FETCH ADRS
JZ DISPLAY


LINE_2:            ; ΕΞΟΥΜΕ 0, STORE/INCR, INCR

MVI A,FBH
STA 2800H          ; ΔΙΑΒΑΖΩ ΓΡΑΜΜΗ
LDA 1800H          ; ΔΙΑΒΑΖΩ ΣΤΗΛΗ
ANI 07H            ; ΚΑΝΩ ΤΑ 5 MSB ΜΗΔΕΝ
MVI C,00H
CPI 06H            ; 0
JZ DISPLAY
MVI C,83H
CPI 05H            ; STORE/INCR
JZ DISPLAY
MVI C,81H
CPI 03H            ;INCR
JZ DISPLAY


LINE_3:            ; ΕΞΟΥΜΕ 1, 2, 3

MVI A,F7H
STA 2800H          ; ΔΙΑΒΑΖΩ ΓΡΑΜΜΗ
LDA 1800H          ; ΔΙΑΒΑΖΩ ΣΤΗΛΗ
ANI 07H            ; ΚΑΝΩ ΤΑ 5 MSB ΜΗΔΕΝ
MVI C,01H
CPI 06H            ; 1
JZ DISPLAY
MVI C,02H
CPI 05H            ; 2
JZ DISPLAY
MVI C,03H
CPI 03H            ; 3
JZ DISPLAY


LINE_4:            ; ΕΞΟΥΜΕ 4, 5, 6

MVI A,EFH
STA 2800H          ; ΔΙΑΒΑΖΩ ΓΡΑΜΜΗ
LDA 1800H          ; ΔΙΑΒΑΖΩ ΣΤΗΛΗ
ANI 07H            ; ΚΑΝΩ ΤΑ 5 MSB ΜΗΔΕΝ
MVI C,04H
CPI 06H            ; 4
JZ DISPLAY
MVI C,05H
CPI 05H            ; 5
JZ DISPLAY
MVI C,06H
CPI 03H            ; 6
JZ DISPLAY


LINE_5:            ; ΕΞΟΥΜΕ 7, 8, 9

MVI A,DFH
STA 2800H          ; ΔΙΑΒΑΖΩ ΓΡΑΜΜΗ
```

```
LDA 1800H          ; ΔΙΑΒΑΖΩ ΣΤΗΛΗ
ANI 07H            ; ΚΑΝΩ ΤΑ 5 MSB ΜΗΔΕΝ
MVI C,07H
CPI 06H            ; 7
JZ DISPLAY
MVI C,08H
CPI 05H            ; 8
JZ DISPLAY
MVI C,09H
CPI 03H            ; 9
JZ DISPLAY


LINE_6:            ; ΕΧΟΥΜΕ A, B, C

MVI A,BFH
STA 2800H          ; ΔΙΑΒΑΖΩ ΓΡΑΜΜΗ
LDA 1800H          ; ΔΙΑΒΑΖΩ ΣΤΗΛΗ
ANI 07H            ; ΚΑΝΩ ΤΑ 5 MSB ΜΗΔΕΝ
MVI C,0AH
CPI 06H            ; A
JZ DISPLAY
MVI C,0BH
CPI 05H            ; B
JZ DISPLAY
MVI C,0CH
CPI 03H            ; C
JZ DISPLAY


LINE_7:            ; ΕΧΟΥΜΕ D, E, F

MVI A,7FH
STA 2800H          ; ΔΙΑΒΑΖΩ ΓΡΑΜΜΗ
LDA 1800H          ; ΔΙΑΒΑΖΩ ΣΤΗΛΗ
ANI 07H
MVI C,0DH
CPI 06H            ; D
JZ DISPLAY
MVI C,0EH
CPI 05H            ; E
JZ DISPLAY
MVI C,0FH
CPI 03H            ; F
JZ DISPLAY
JMP START          ; ΑΝ ΔΕΝ ΕΧΕΙ ΠΑΤΗΘΕΙ ΚΑΠΟΙΟ ΚΟΥΜΠΙ ΠΑΕΙ ΣΤΗΝ ΑΡΧΗ ΚΑΙ
ΞΑΝΑΕΛΕΓΧΕΙ


DISPLAY:

LXI H,0A06H        ;
MOV A,C            ; Ο C ΕΧΕΙ ΤΟ ΚΩΔΙΚΑ ΤΟΥ ΚΟΥΜΠΙΟΥ ΚΑΙ ΤΟΝ ΜΕΤΑΦΕΡΩ ΣΤΟ A
ANI 0FH            ; ΑΠΟΜΟΝΩΝΩ ΤΑ 4 LSB
MOV M,A            ; ΑΠΟΘΗΚΕΥΩ ΣΤΗΝ ΜΝΗΜΗ ΤΟΝ A
INX H              ; ΔΕΙΧΝΩ ΣΤΟ ΕΠΟΜΕΝΟ ΚΕΛΙ ΜΝΗΜΗΣ
MOV A,C            ; Ο C ΕΧΕΙ ΤΟ ΚΩΔΙΚΑ ΤΟΥ ΚΟΥΜΠΙΟΥ ΚΑΙ ΤΟΝ ΜΕΤΑΦΕΡΩ ΣΤΟ A
ANI F0H            ; ΑΠΟΜΟΝΩΝΩ ΤΑ 4 MSB
RRC
RRC                ; ΚΑΝΩ ΟΛΙΣΘΗΣΗ ΣΤΑ 4 LSB
RRC
RRC
MOV M,A            ; ΑΠΟΘΗΚΕΥΩ ΣΤΗΝ ΜΝΗΜΗ ΤΟΝ A
LXI D,0A02H        ; Η ΔΙΕΥΘΥΝΣΗ ΜΝΗΜΗΣ ΠΟΥ ΕΙΝΑΙ ΤΟ message
CALL STDM
CALL DCD
JMP START
END
```

ΑΣΚΗΣΗ 4η:

Να εξομοιωθεί η λειτουργία ενός υποθετικού I.C. που περιλαμβάνει 5 πύλες όπως φαίνεται στο σχήμα της εκφώνησης. Τα bits εισόδου πρέπει να αντιστοιχούν ακριβώς όπως φαίνονται στο σχήμα με τα dip switches της πόρτας εισόδου 2000 Hex, και οι έξοδοι με τα LEDs που πρέπει να είναι τα τέσσερα LSB της πόρτας εξόδου 3000 Hex. Οι πύλες, όπως φαίνεται στο σχήμα, είναι 2 AND, 2 OR και 1 XOR.

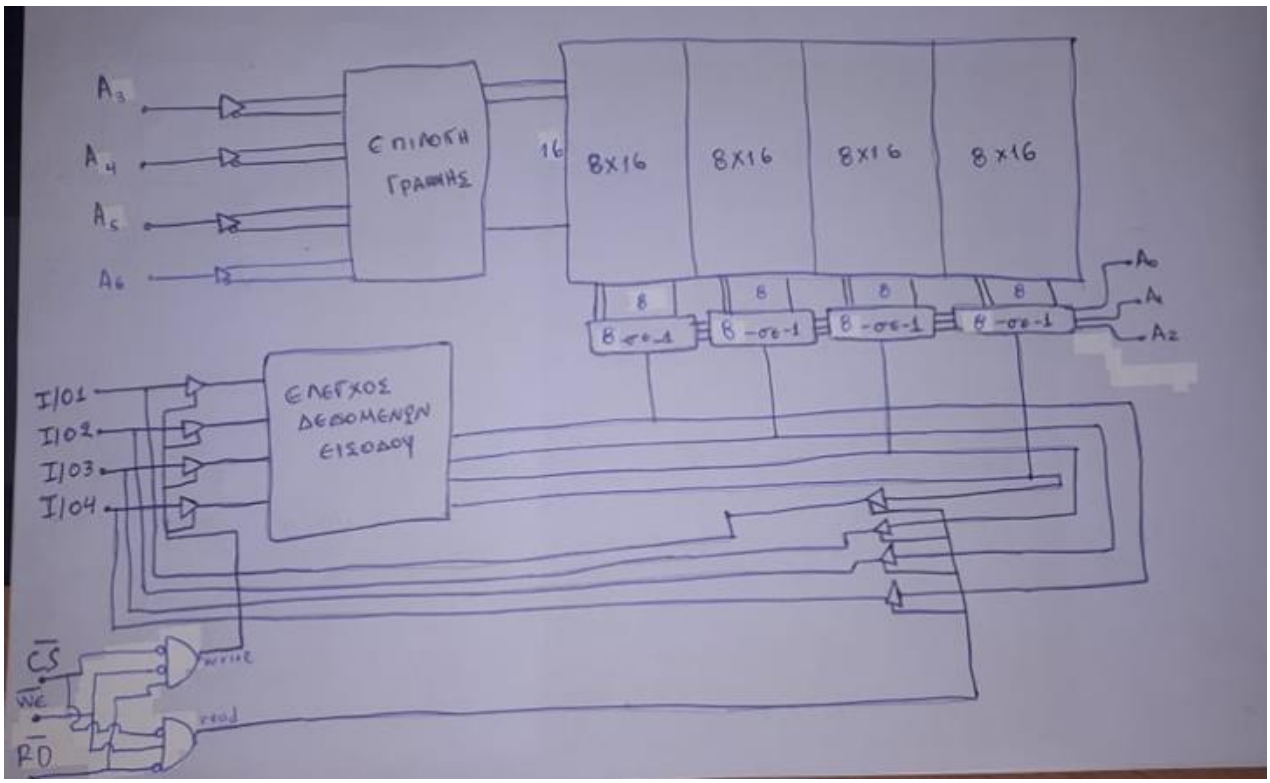
```
START:
MVI E,00H
LDA 2000H         ; ΔΙΑΒΑΖΩ ΑΠΟ ΕΙΣΟΔΟ
MOV C,A
MVI B,00H         ;
ANI 03H           ; ΑΠΟΜΟΝΩΝΩ ΤΑ 2 ΤΕΛΕΥΤΑΙΑ ΨΗΦΙΑ
CALL SUBR_0       ; ΕΔΩ ΘΑ ΑΝΑΒΟΥΜΕ ΤΟ LSB
MOV A,C
ANI 0CH           ; ΑΠΟΜΟΝΩΝΩ ΤΑ (A1 B1)
CALL SUBR_1       ; ΚΑΙ ΠΗΓΑΙΝΩ ΣΤΗΝ ΥΠΟΡΟΥΤΙΝΑ_1
MOV A,C
ANI 30H           ; ΑΠΟΜΟΝΩΝΩ ΤΑ (A2 B2)
CALL SUBR_2       ; ΚΑΙ ΠΗΓΑΙΝΩ ΣΤΗΝ ΥΠΟΡΟΥΤΙΝΑ_2
MOV A,C
ANI 80H           ; ΑΠΟΜΟΝΩΝΩ ΤΑ (A3 B3)
MOV D,A
MOV A,C
ANI 40H
RLC
ANA D             ; ΥΛΟΠΟΙΩ ΤΗΝ ΠΥΛΗ AND
XRA E             ; ΥΛΟΠΟΙΩ ΤΗΝ ΠΥΛΗ XOR
CALL SUBR_3       ; ΚΑΙ ΠΗΓΑΙΝΩ ΣΤΗΝ ΥΠΟΡΟΥΤΙΝΑ_3
MOV A,B
CMA
STA 3000H         ; ΚΑΙ ΕΜΦΑΝΙΖΩ ΤΟ ΑΠΟΤΕΛΕΣΜΑ

JMP START         ; ΓΥΡΙΖΩ ΣΤΗΝ ΑΡΧΗ ΓΙΑ ΝΑ ΕΠΙΤΥΧΩ ΤΗΝ ΣΥΝΕΧΗ ΛΕΙΤΟΥΡΓΙΑ


SUBR_0:
CPI 01H           ; ΕΛΕΓΧΩ ΑΝ ΤΟ B0 ΕΙΝΑΙ ΟΝ
JZ LABEL_0        ; ΑΝ ΝΑΙ ΠΗΓΑΙΝΩ ΚΑΙ ΑΝΑΒΩ ΤΟ 1ο ΛΑΜΠΑΚΙ
CPI 02H           ; ΕΛΕΓΧΩ ΑΝ ΤΟ A0 ΕΙΝΑΙ ΟΝ
JZ LABEL_0        ; ΑΝ ΝΑΙ ΠΗΓΑΙΝΩ ΚΑΙ ΑΝΑΒΩ ΤΟ 1ο ΛΑΜΠΑΚΙ
CPI 03H           ; ΕΛΕΓΧΩ ΑΝ ΚΑΙ ΤΟ B0 ΚΑΙ ΤΟ A0 ΕΙΝΑΙ ΟΝ
JZ LABEL_0        ; ΑΝ ΝΑΙ ΠΗΓΑΙΝΩ ΚΑΙ ΑΝΑΒΩ ΤΟ 1ο ΛΑΜΠΑΚΙ
MVI B,00H         ; ΑΛΛΙΩΣ ΕΜΦΑΝΙΖΩ ΟΤΙ ΕΙΧΑ ΠΡΙΝ (ΤΟ ΜΗΔΕΝ ΕΔΩ)
JMP RETURN_0      ; ΚΑΙ ΕΠΙΣΤΡΕΦΩ ΑΠΟ ΤΗ ΡΟΥΤΙΝΑ
LABEL_0:
MVI B,01H         ; ΑΝΑΒΩ ΤΟ ΛΑΜΠΑΚΙ ΠΟΥ ΠΡΕΠΕΙ
RETURN_0:         ; ΚΑΙ ΕΠΙΣΤΡΕΦΩ ΑΠΟ ΤΗ ΡΟΥΤΙΝΑ
RET




SUBR_1:
CPI 0CH           ; ΕΛΕΓΧΩ ΑΝ ΤΑ A1 ΚΑΙ B1 ΕΙΝΑΙ ΟΝ
JZ LABEL_1        ; ΑΝ ΝΑΙ ΠΑΩ ΚΑΙ ΑΝΑΒΩ ΤΟ 2ο ΛΑΜΠΑΚΙ
MVI A,00H         ; ΑΛΛΙΩΣ ΑΦΗΝΩ ΟΤΙ ΗΤΑΝ ΠΡΙΝ ΑΝΑΜΕΝΟ ΜΟΝΟ
ORA B
MOV B,A
JMP RETURN_1      ; ΚΑΙ ΕΠΙΣΤΡΕΦΩ ΑΠΟ ΤΗ ΡΟΥΤΙΝΑ
LABEL_1:
MVI A,02H         ; ΑΝΑΒΩ ΤΟ 2ο ΛΑΜΠΑΚΙ ΑΙ ΟΤΙ ΑΛΛΟ ΗΤΑΝ ΠΡΙΝ
ORA B
MOV B,A
RETURN_1:         ; ΕΠΙΣΤΡΕΦΩ ΑΠΟ ΤΗ ΡΟΥΤΙΝΑ
```

```
RET




SUBR_2:
CPI 10H          ; ΕΛΕΓΧΩ ΑΝ ΤΟ Β2 ΕΙΝΑΙ ΟΝ
JZ LABEL_2       ; ΑΝ ΝΑΙ ΠΑΩ ΚΑΙ ΑΝΑΒΩ ΤΟ 3ο ΛΑΜΠΑΚΙ
CPI 20H          ; ΕΛΕΓΧΩ ΑΝ ΤΟ Α2 ΕΙΝΑΙ ΟΝ
JZ LABEL_2       ; ΑΝ ΝΑΙ ΠΑΩ ΚΑΙ ΑΝΑΒΩ ΤΟ 3ο ΛΑΜΠΑΚΙ
CPI 30H          ; ΕΛΕΓΧΩ ΑΝ ΤΑ Α2 ΚΑΙ Β2 ΕΙΝΑΙ ΟΝ
JZ LABEL_2       ; ΑΝ ΝΑΙ ΠΑΩ ΚΑΙ ΑΝΑΒΩ ΤΟ 3ο ΛΑΜΠΑΚΙ
MVI A,00H        ; ΑΛΛΙΩΣ ΑΦΗΝΩ ΟΤΙ ΕΙΧΑ ΠΡΙΝ ΑΝΑΜΕΝΟ ΜΟΝΟ
ORA B
MOV B,A
JMP RETURN_2     ; ΚΑΙ ΕΠΙΣΤΡΕΦΩ ΑΠΟ ΤΗ ΡΟΥΤΙΝΑ
LABEL_2:
MVI A,04H        ; ΑΝΑΒΩ ΤΟ 3Ο ΛΑΜΠΑΚΙ ΚΑΙ ΟΤΙ ΑΛΛΟ ΗΤΑΝ ΠΡΙΝ ΑΝΑΜΕΝΟ
ORA B
MOV B,A
MVI A,80H
MOV E,A
RETURN_2:        ; ΕΠΙΣΤΡΕΦΩ ΑΠΟ ΤΗ ΡΟΥΤΙΝΑ
RET




SUBR_3:
CPI 80H          ; ΕΛΕΓΧΩ ΑΝ ΠΡΕΠΕΙ ΝΑ ΑΝΑΨΕΙ ΤΟ 4ο ΛΑΜΠΑΚΙ
JZ LABEL_3       ; ΑΝ ΝΑΙ ΠΑΩ ΚΑΙ ΑΝΑΒΩ ΤΟ 4ο ΛΑΜΠΑΚΙ
MVI A,00H        ; ΑΛΛΙΩΣ ΑΦΗΝΩ ΟΤΙ ΕΙΧΑ ΠΡΙΝ ΑΝΑΜΕΝΟ ΜΟΝΟ
ORA B
MOV B,A
JMP RETURN_3     ; ΚΑΙ ΕΠΙΣΤΡΕΦΩ ΑΠΟ ΤΗ ΡΟΥΤΙΝΑ
LABEL_3:
MVI A,08H        ; ΑΝΑΒΩ ΤΟ 4ο ΛΑΜΠΑΚΙ ΚΑΙ ΟΤΙ ΑΛΛΟ ΕΙΧΑ ΠΡΙΝ
ORA B
MOV B,A
RETURN_3:        ; ΕΠΙΣΤΡΕΦΩ ΑΠΟ ΤΗ ΡΟΥΤΙΝΑ
RET



END
```

ΑΣΚΗΣΗ 5η:

Να δοθεί η εσωτερική οργάνωση μια μνήμη SRAM 128×4 bit. Θέλουμε επίσης να εξηγηθεί μέσω ενός παραδείγματος με ποιο τρόπο γίνεται η ανάγνωση και εγγραφή στη μνήμη αυτή.

Η μνήμη είναι 128 x 4 bits. Ο δεύτερος αριθμός , το 4, δηλώνει σε πόσα ίσα μέρη θα χωριστεί η μνήμη και ο πρώτος αριθμός , το 128, δηλώνει τη χωρητικότητα του κάθε μέρους. Άρα η μνήμη θα χωρίζεται σε 4 ίσα μέρη των 128 bits. Χρησιμοποιούμε 3 bits (A0-A2) για τη κωδικοποίηση των 8 στηλών κάθε μέρους και άλλα 4 bits (A3-A6) για τις αντίστοιχες γραμμές.



Η τετραγωνική διάταξη του παραπάνω σχήματος αποτελείται 32x16 στοιχεία μνήμης. Από τον πίνακα της μνήμης επιλέγεται με βάση τις γραμμές διεύθυνσης A3-A6 μια από τις 16 γραμμές. Η κάθε γραμμή του πίνακα που επιλέγεται από ένα κύκλωμα αποκωδικοποίησης. Αυτά είναι οργανωμένα σε 4 8άδες. Οι πολυπλέκτες υλοποιούνται με διακόπτες και επιτρέπουν τη διέλευση δεδομένων και προς τις 2 πλευρές (είτε για ανάγνωση είτε για εγγραφή).

ΑΣΚΗΣΗ 6η :

Να σχεδιασθεί ένα σύστημα μνήμης που να περιλαμβάνει χώρο μνήμης 6KBytes ROM ακολουθούμενη χωρίς κενό διευθύνσεων από 10KBytes RAM. Η ROM ξεκινά από τη διεύθυνση 0000H και υλοποιείται χρησιμοποιώντας 1 ολοκληρωμένα μνήμης: ένα των 2K×8 bit και ένα των 4K×8 bit (2 ICs). Η RAM να υλοποιηθεί με χρήση μιας μνήμης 2K×8 και μιας 8K×8 SRAMs (2 ICs).

Έχουμε 2 μνήμες RAM και 2 μνήμες ROM.

ROM:

Η 2Kx8 χρειάζεται 11 bits για να παρασταθεί (A0-A10)

Η 4Kx8 χρειάζεται 12 bits για να παρασταθεί (A0-A11)

RAM:

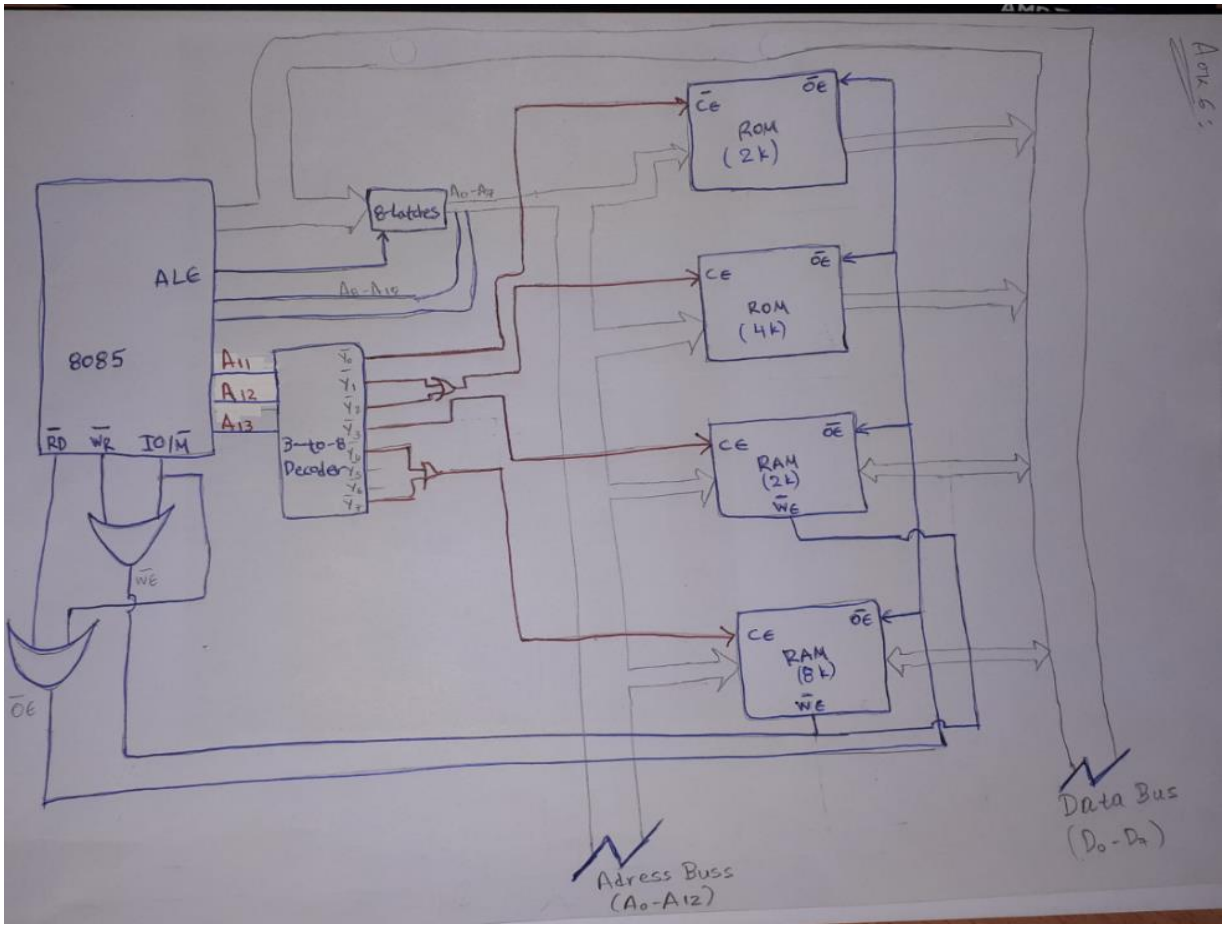
Η 2Kx8 χρειάζεται 11 bits για να παρασταθεί (A0-A10)

Η 8Kx8 χρειάζεται 13 bits για να παρασταθεί (A0-A12)

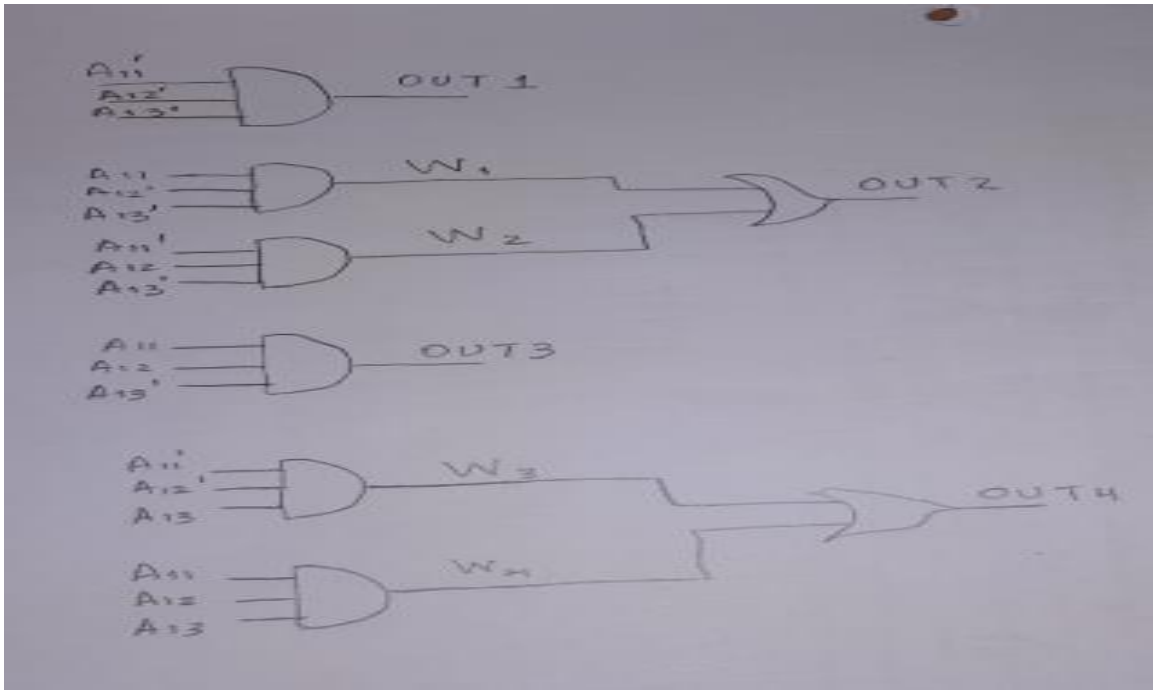
ΜΝΗΜΗ	ΘΕΣΕΙΣ	ΑΡΧΗ	ΤΕΛΟΣ
ROM1	2048	0000	7FF
ROM2	4096	800	17FF
RAM1	4096	1800	1FFF
RAM2	8192	2000	3FFF

Η κάθε μνήμη ξεκινά στην επόμενη θέση που τελειώνει η προηγούμενή της.

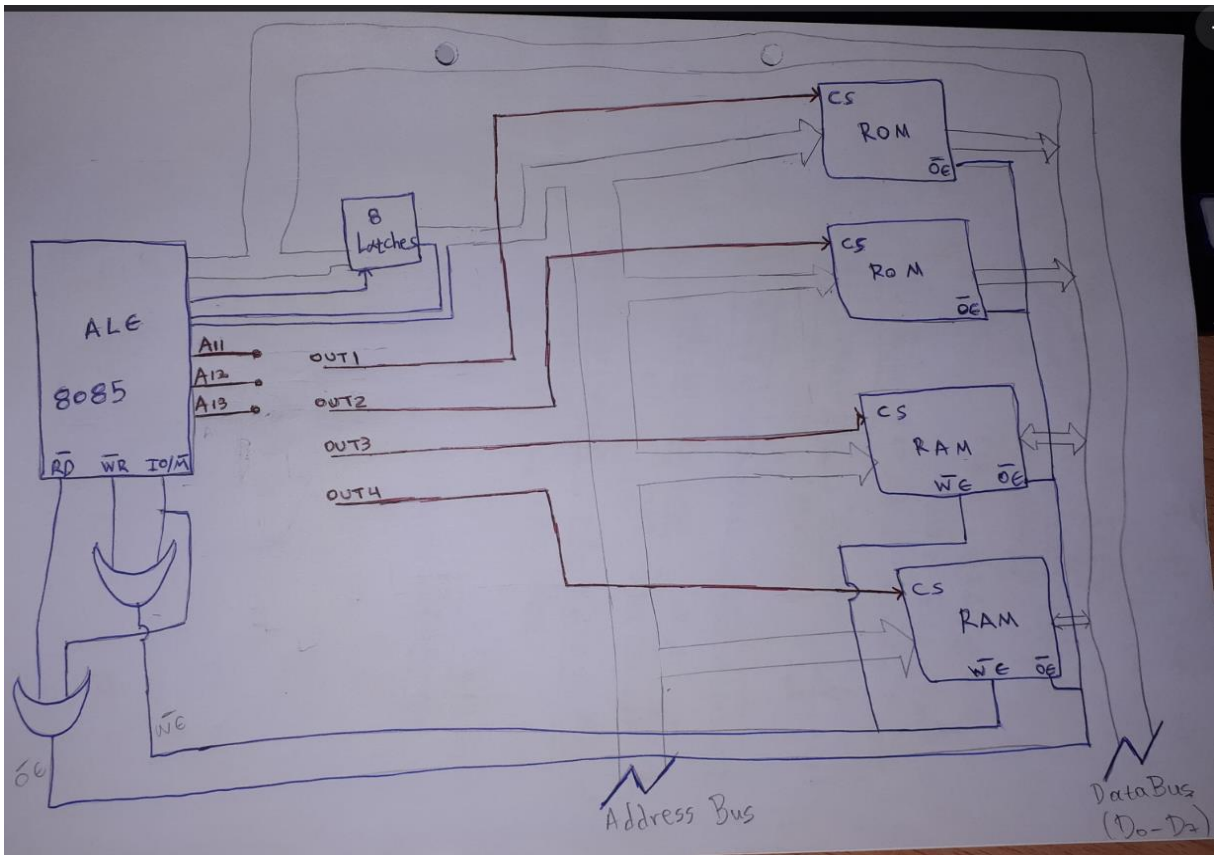
(α) Παρατηρούμε ότι τα bit που διαχωρίζουν τις 4 μνήμες είναι τα A11,A12,A13 και άρα αυτά θα χρησιμοποιήσουμε για εισόδους του πολυπλέκτη.



(β) Εδώ αντί για πολυπλέκτη χρησιμοποιούμε μόνο λογικές πύλες με είσοδο τα A11,A12 και A13 όπως πριν.



Κάθε έξοδος από τις πύλες πάει σε μια μνήμη όπως φαίνεται παρακάτω στο σχήμα ώστε να ξεχωρίζουμε ποια μνήμη θα χρησιμοποιηθεί.



ΑΣΚΗΣΗ 7η:

Έχουμε από την εκφώνηση τα εξής στοιχεία :

0000-0FFF Hex : ROM (4Kbytes)

1000-3FFF Hex : RAM (12Kbytes)

4000-6FFF Hex : ROM (12Kbytes)

7000 Hex : θύρα εισόδου (Memory map I/O)

70 Hex : θύρα εξόδου (Standard I/O)

Μας παρέχονται τα εξής 3 ολοκληρωμένα κυκλώματα :

ROM των 16Kbytes

RAM των 4Kbytes και 8Kbytes

ROM(4Kbytes)				
ΑΡΧΗ	0	0	0	0
	0000	0000	0000	0000
ΤΕΛΟΣ	0	F	F	F
	0000	1111	1111	1111
RAM(12Kbytes)				
ΑΡΧΗ	1	0	0	0
	0001	0000	0000	0000
ΤΕΛΟΣ	3	F	F	F
	0011	1111	1111	1111
ROM(12Kbytes)				
ΑΡΧΗ	4	0	0	0
	0100	0000	0000	0000
ΤΕΛΟΣ	6	F	F	F
	0110	1111	1111	1111

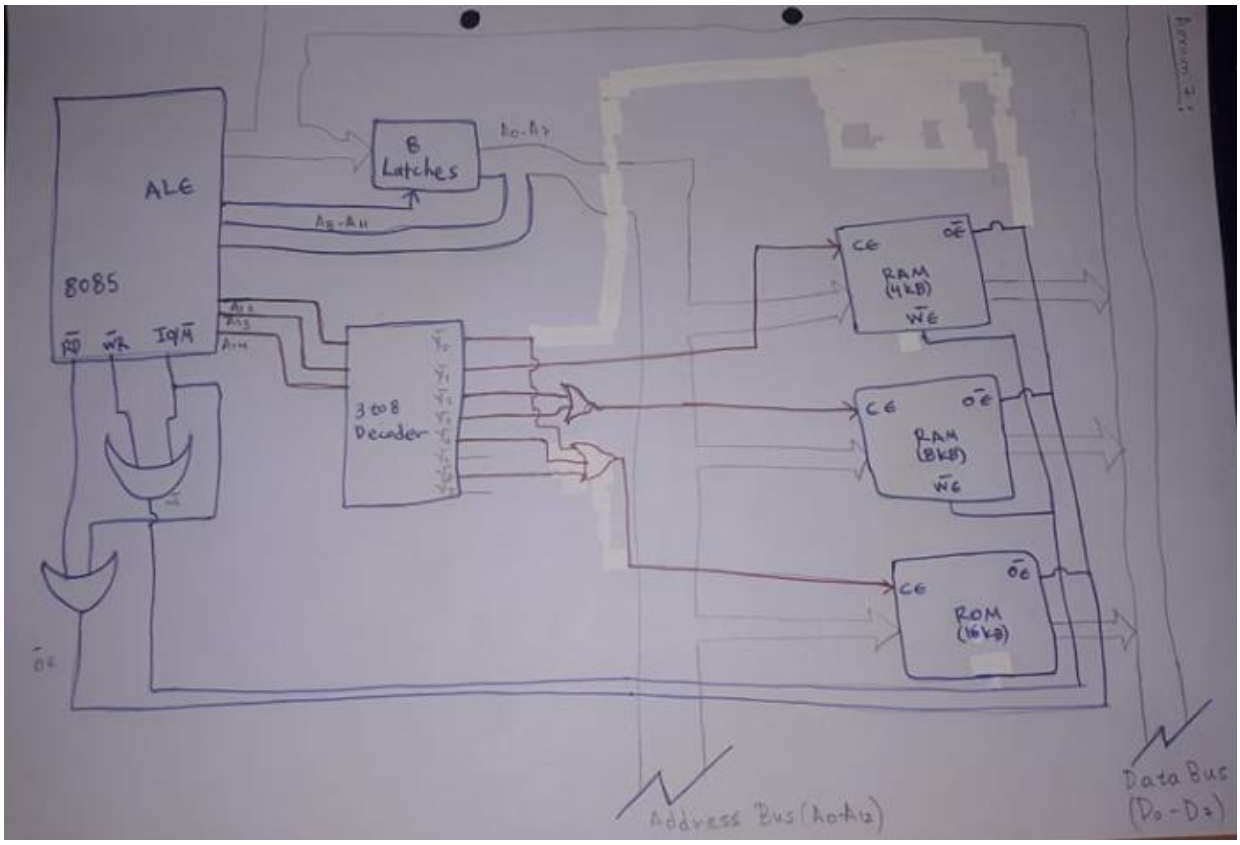
Για την 1η ROM θέλουμε 12bits

Για τη RAM θέλουμε 14bits

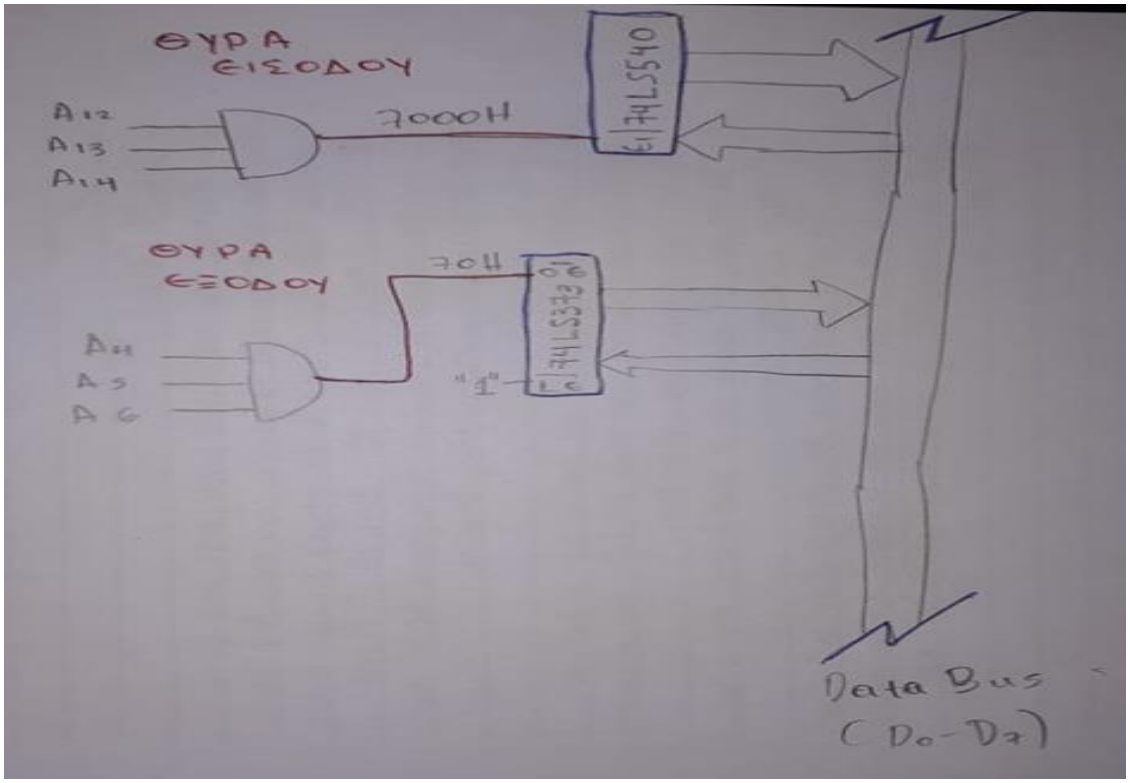
Για τη 2η ROM θέλουμε 14bits

Για τις ROM θα χρησιμοποιήσουμε αυτή των 16Kbytes ενώ για τη RAM αυτές των 4Kbytes και 8 Kbytes.

Βλέπουμε ότι για να ξεχωρίσουμε εδώ τις διάφορες μνήμες χρειαζόμαστε τα bit A12,A13 και A14. Επιλέγουμε τις εξόδους του πολυπλέκτη όπως φαίνεται στο σχήμα με τρόπο ώστε κάθε φορά να επιλέγουμε την κατάλληλη μνήμη.



Τέλος για τις θύρες εισόδου και εξόδου έχουμε τα παρακάτω:


ΘΥΡΑ
ΕΙΣΟΔΟΥ
A12
A13
A14
7000H
74LS540
ΘΥΡΑ
ΕΞΟΔΟΥ
70H
A4
A5
A6
"1"
74LS138
Data Bus
(D0-D7)