

## 4.10 Εξομοίωση Κυκλωμάτων μέσω Προγράμματος

Στην παράγραφο αυτή παρουσιάζεται η τεχνική υλοποίησης κυκλωμάτων με τη χρήση μΥ-Σ και με βάση κάποιο πρόγραμμα. Αν η ταχύτητα δεν είναι κρίσιμο μέγεθος τότε η τεχνική αυτή μπορεί να μας δώσει απλούστερα κυκλώματα, το κυριώτερο όμως πλεονέκτημα είναι η ευελιξία. Δηλαδή αλλάζοντας το πρόγραμμα μπορούμε να αλλάξουμε τη συμπεριφορά του εξομοιούμενου κυκλώματος χωρίς να χρειάζεται η ανακατασκευή του. Στη συνέχεια δίνονται τα παραδείγματα της εξομοίωσης πυλών SSI, πολλαπλασιαστών και ελέγχου διακίνησης κιβωτίων.

### Παράδειγμα 1ο: Εξομοίωση 7400

Δίνεται ένα μΥ σύστημα που διαθέτει μια πόρτα εισόδου (διεύθυνση 10) και μια πόρτα εξόδου (διεύθυνση 20). Με το κύκλωμα αυτό θα υλοποιηθεί η λειτουργία του IC 7400. Οι ακίδες 1, 2, 4, 5, 9, 10, 12 και 13 αντιστοιχούνται στα ψηφία 0-7 της πόρτας εισόδου και οι ακίδες 3, 6, 8 και 11 στα ψηφία 0-3 της πόρτας εξόδου.

Το IC 7400 περιέχει 4 πύλες NAND κάθε μια από τις οποίες έχει σαν εισόδους τις ακίδες 1,2, 4,5, 9,10, 12,13 αντίστοιχα και εξόδους τις ακίδες 3, 6, 8, 11 αντίστοιχα.

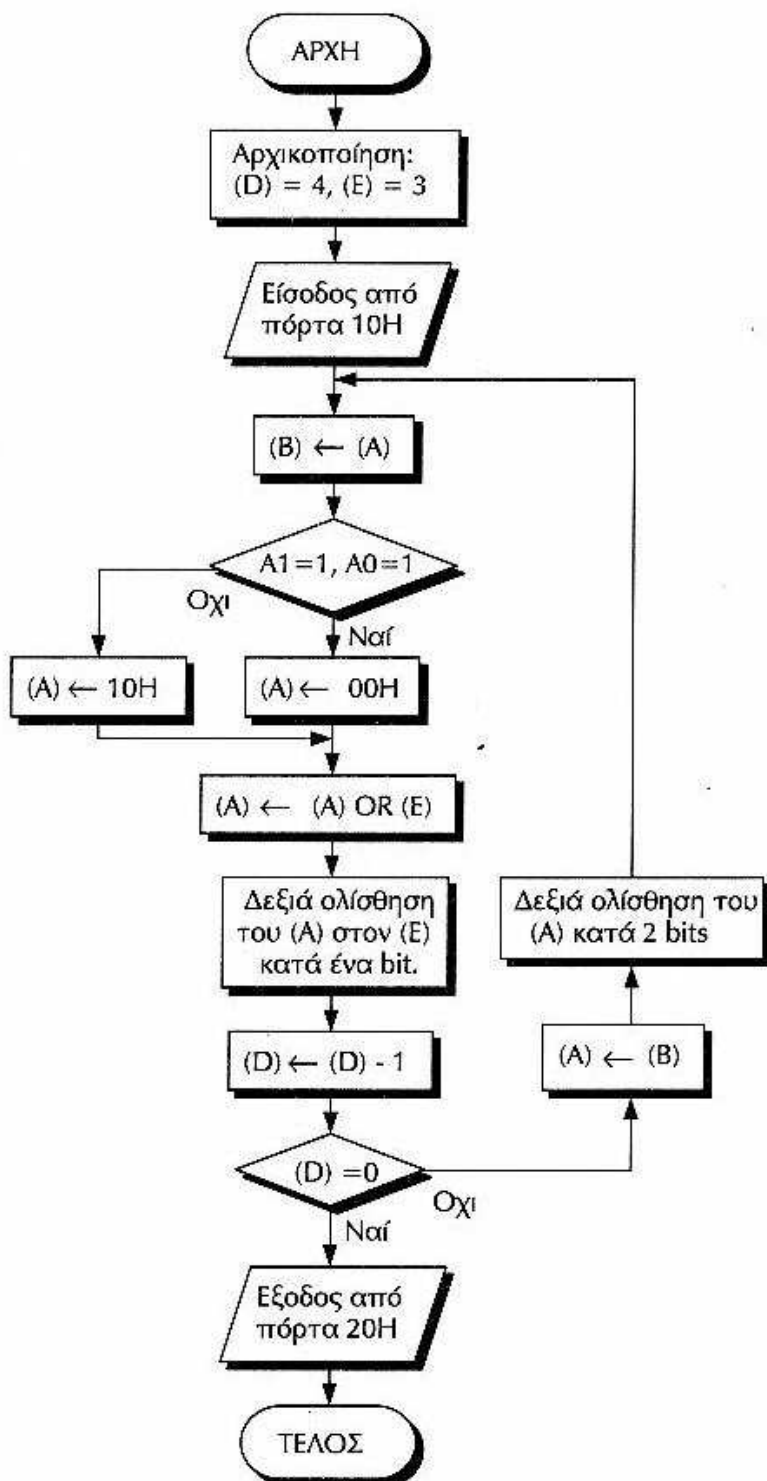
Το διάγραμμα ροής του προγράμματος βρίσκεται στο σχήμα 4.25. Το αντίστοιχο πρόγραμμα είναι:

MVI	E, 00	;Αρχικοποίηση του E
MVI	C, 03	;Αρχικοποίηση της μάσκας
MVI	D, 04	;Αρχικοποίηση του μετρητή για τα 4 ζεύγη εισόδων
IN	10	;Είσοδος από την πόρτα 10
CONTROL:		
MOV	B, A	;(B) ← (A)
ANA	C	;Αν $A_1A_0 \neq 11$ τότε η πύλη
CMP	C	;NAND έχει έξοδο 1 και κάνουμε
JNZ	FORM1	;άλμα στην διεύθυνση FORM1
MVI	A, 00	;Διαφορετικά η πύλη NAND έχει
JMP	FORM0	;έξοδο 0 και κάνουμε άλμα στην διεύθυνση FORM0
FORM1:		
MVI	A, 10	;A <sub>4</sub> ← 1
FORM0:		
ORA	E	;Τοποθετούμε στον (A) και τους άλλους άσσους αν
		;υπάρχουν από προηγούμενα ζεύγη
RRC		;Τοποθετούμε στον E το νέο μερικό
MOV	E, A	;αποτέλεσμα
DCR	D	;Εάν έχουν ελεγχθεί όλα τα ζεύγη

JZ	OUTPUT	;εισόδων τότε άλμα στην διεύθ. OUTPUT
MOV	A, B	;Διαφορετικά τοποθετούμε στις θέσεις
RRC		;A1A0 το επόμενο ζεύγος εισόδων και
RRC		;το ελέγχουμε με τον ίδιο τρόπο
JMP	CONTROL	

OUTPUT:

OUT	20	;Έξοδος από την πόρτα 20
HLT		;Τέλος



Σχήμα 4.25 Διάγραμμα ροής εξομοιωτή 7400.

**Παράδειγμα 2ο: Υλοποίηση παράλληλου πολλαπλασιαστή**

Με το κύκλωμα του προηγούμενου παραδείγματος (δηλαδή ένα μΥ σύστημα που διαθέτει μια θύρα εισόδου με διεύθυνση 10H και εξόδου με διεύθυνση 20) θα υλοποιηθεί ένας πολλαπλασιαστής 4 bits x 4 bits (το γινόμενο σε 8 bits), για αριθμούς  $x$ ,  $y$ . Οι αριθμοί  $y$  και  $x$  παρέχονται από το πιο σημαντικό και το λιγότερο σημαντικό δεκαεξαδικό ψηφίο της θύρας εισόδου αντίστοιχα. Το αποτέλεσμα (8 bit) παρέχεται συνεχώς για κάθε νέο ζεύγος δεδομένων στη θύρα εξόδου.

Ο πολλαπλασιαστής  $y$  είναι στον καταχωρητή A και ο πολλαπλασιαστέος  $x$  στον καταχωρητή C. Με την αριστερή ολίσθηση του C εκταμιεύεται το ψηφίο μέγιστης αξίας του πολλαπλασιασμού ( $2^3 y_3$ ). Αν είναι 1 τότε προκαλεί την πρόσθεση του όρου  $x$  που μέχρι το τέλος του προγράμματος θα ολισθήσει τρεις φορές αριστερά. Έτσι θα προστεθεί τελικά ο όρος ο όρος  $2^3 y_3 x$ . Στο επόμενο βήμα προωθείται ο όρος  $2^2 y_2 x$  (2 αριστερά ολισθήσεις) κ.ο.κ. Το διάγραμμα ροής των αλγορίθμων φαίνεται στο σχήμα 4.26. Το πρόγραμμα που υλοποιεί τον παραπάνω αλγόριθμο είναι το ακόλουθο:

ARXH:

```
IN    10H      ; Είσοδος παραγόντων
MOV   B, A     ;
ANI   0FH      ; Διαχωρισμός x
MOV   C, A     ; C ← x
MOV   A, B     ; Επανάκτηση της εισόδου
ANI   F0H      ; Διαχωρισμός y
MVI   B, 4     ;
```

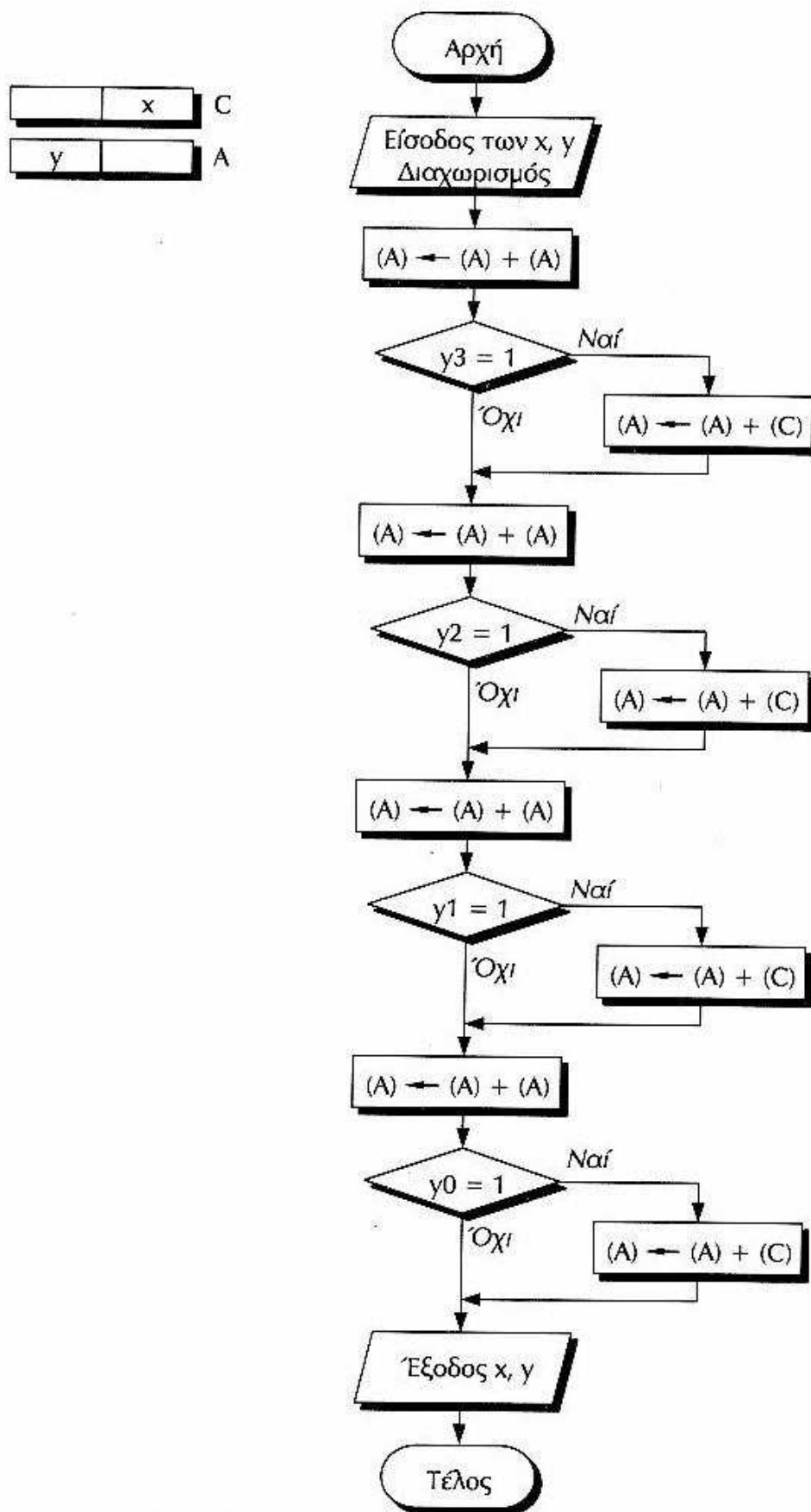
ADDR2:

```
ADD   A        ; Διπλασιασμός και εκταμίευση των  $y_i$ 
JNC   ADDR1    ; Αν  $y_i = 1$  προσθέτω το  $x$ 
ADD   C        ; Αλλιώς παρακάμπτεται
```

ADDR1:

```
DCR   B        ; Επαναλαμβάνεται 4 φορές
JNZ   ADDR2    ;
OUT   20H      ; Το γινόμενο 8 bits στην έξοδο
JMP   ARXH     ; Συνεχίζεται η διαδικασία για νέα δεδομένα
END
```

Ένας δεύτερος τρόπος υλοποίησης για μπορεί να επιτευχθεί με την χρήση ενός πίνακα γινομένων 256 θέσεων (π.χ στις διευθύνσεις 0100 - 01FF). Εδώ ο αριθμός  $xy$  όπως λαμβάνεται από τη θύρα εισόδου θα αποτελεί τη σχετική απόσταση από την αρχή του πίνακα. Η ταχύτητα υλοποίησης του προγράμματος στην περίπτωση αυτή θα είναι μεγαλύτερη.

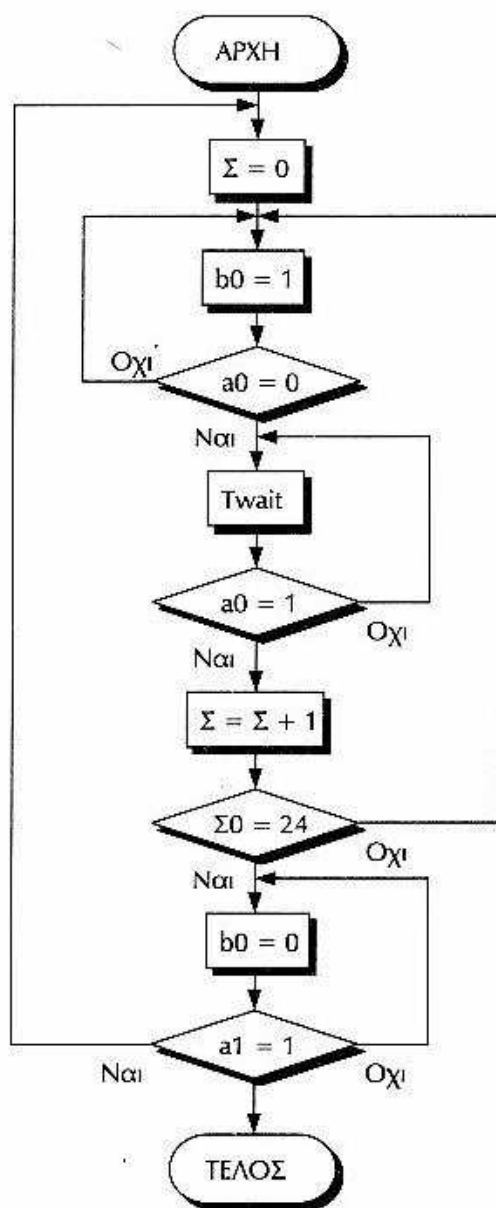


Σχήμα 4.26 Διάγραμμα ροής πολλαπλασιαστή

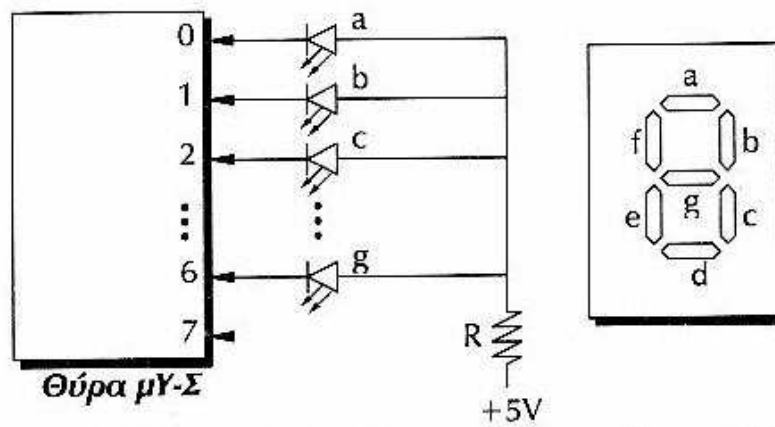
**Παράδειγμα 3ο**

Να γραφεί πρόγραμμα που υλοποιεί το παρακάτω διάγραμμα ροής

που παριστά τον αλγόριθμο ελέγχου διακίνησης κιβωτίων με ιμάντα. Τα σήματα που το ελέγχουν παρέχονται από την πόρτα εισόδου Α, που βρίσκεται στην διεύθυνση 2000H, ως εξής: το bit  $a_0$  οδηγείται από ένα φωτοκύτταρο που απενεργοποιείται όταν διέρχεται κιβώτιο. Το σύστημα αφού μετρήσει 24 κιβώτια σταματάει και αναμένει το bit  $a_1$  να γίνει 1 οπότε και επαναλαμβάνει την διαδικασία. Η κίνηση του ιμάντα ελέγχεται από το bit  $b_0$  της πόρτας εξόδου Β που βρίσκεται στην διεύθυνση 3000H ως εξής:  $b_0=1$  κίνηση,  $b_0 = 0$  στάση. Ο αριθμός των κιβωτίων να απεικονίζεται συνέχεια σε δεκαδική μορφή. Το διάγραμμα ροής της διαδικασίας φαίνεται στο σχήμα 4.27. Οι δύο ενδείκτες 7 τμημάτων (7-segments) να οδηγηθούν από τις πόρτες 20H και 30H. Τα τμήματα a, b, c, d, e, f, g να αντιστοιχούν στα bits 0-6. Ο τρόπος σύνδεσης ενός ενδείκτη 7 τμημάτων με πόρτα εισόδου μΥ-Σ δίνεται στο σχήμα 4.28.



Σχήμα 4.27 Διάγραμμα ροής ελέγχου διακίνησης κιβωτίων με ιμάντα

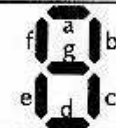
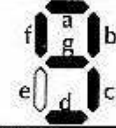


Σχήμα 4.28 Σύνδεση ενδείκτη 7 τμημάτων σε θύρα εισόδου μΥ-Σ

Όπως φαίνεται από τη συνδεσμολογία του σχήματος 4.28 για να ανάψει ένα Led (τμήμα) πρέπει η αντίστοιχη έξοδος της θύρας του μΥ-Σ να είναι 0. Έτσι οι αντίστοιχοι κωδικοί πρέπει να είναι αυτοί που φαίνονται στον πίνακα 4.5.

Πίνακας 4.5 Αντιστοιχία ενεργοποίησης τμημάτων ενδείκτη και δεκαδικών ψηφίων

		g	f	e	d	c	b	a	
0		1	0	0	0	0	0	0	=40H
1		1	1	1	1	0	0	1	=79H
2		0	1	0	0	1	0	1	=25H
3		0	1	1	0	0	0	0	=30H
4		0	0	1	1	0	0	1	=19H
5		0	0	1	0	0	1	0	=12H
6		0	0	0	0	0	1	0	=02H
7		1	1	1	1	0	0	0	=78H

8		0 0 0 0 0 0 0 =00H
9		0 0 1 0 0 0 0 =10H

Στη συνέχεια παραθέτουμε το πρόγραμμα υλοποίησης σε ένα μΥ-Σ που βασίζεται στο μΕ 8085. Στις θέσεις 0800 έως 0809 (διεύθυνση CODES στο πρόγραμμα) βρίσκονται αποθηκευμένοι οι κώδικες που αντιστοιχούν στην παράσταση των αριθμών 0 έως 9 σε μορφή 7-segments (a, b, c, d, e, f, g). Στην πόρτα 20H απεικονίζουμε το MSD του διψήφιου δεκαδικού αριθμού.

A1:	MVI B, 00H	;Ο καταχωρητής B είναι ο μετρητής κιβωτίων
	LXI H, 0800H	;Ο καταχωρητής H-L δείχνει τη θέση CODES
	MOV A, M	
	OUT 20H	;Απεικόνιση αρχικής τιμής στα 7-segments
	OUT 30H	
A2:	MVI A, 01H	;b0 = 1
	STA 3000H	;Κίνηση ιμάντα
	LDA 2000H	;αν α0 = 1 συνέχισε
	RRC	
	JC A2	
A3:	LDA 2000H	
	RRC	
	JNC A3	;Αναμονή μέχρι να γίνει α0=1, αλλιώς για α0=0
	INR B	; διέρχεται κιβώτιο και αυξάνουμε το μετρητή
	MOV A, B	;Μεταφορά στον καταχωρητή A για
	DAA	;δεκαδική ρύθμιση
	MOV B, A	;Φύλαξη μετρητή
	RRC	;Ολίσθηση 4ων θέσεων
	RRC	
	RRC	
	RRC	
	ANI 0FH	;Απομόνωση του πιο σημαντικού δεκαδικού ψηφίου
	MOV L, A	
	MOV A, M	
	OUT 20H	;Απεικόνιση του πιο σημαντικού ψηφίου στα 7-segments
	MOV A, B	
	ANI 0FH	;Απομόνωση του λιγότερου σημαντικού ψηφίου
	MOV L, A	
	MOV A, M	
	OUT 30H	;Απεικόνιση του λιγότερου σημαντικού δεκαδικού ψηφίου
	MVI A, 24H	;Ελεγχος αν πέρασαν 24 κιβώτια (Προσοχή: 24 Hex !)

```

CMP    B
JNZ    A2
A4:    MVI    A, 00H
        STA    3000H           ;b0=0
        LDA    2000H
        RRC
        RRC
        JC     A1              ;αν α1 = 1 πήγαινε στο A1
        JMP    A4
        HLT

```

CODES:

40H, 79H, 25H, 30H, 19H, 12H, 02H, 78H, 00H, 10H

Ο τυπικός τρόπος εισαγωγής τέτοιων δεδομένων με την οδηγία DB δίνεται στο επόμενο κεφάλαιο.

## Ασκήσεις προς λύση

**4.1** Γράψτε ένα πρόγραμμα που συνεχόμενα διαβάζει δεδομένα από δύο πόρτες PORT1 και PORT2. Όταν τα 4 λιγότερο σημαντικά bits της πόρτας PORT1 ταυτίζονται με αυτά της πόρτας PORT2, το bit 0 της πόρτας εξόδου PORT\_3 να γίνεται 1, διαφορετικά να μένει 0. Γράψτε ένα ελαχιστο πρόγραμμα με σχόλια για την κάθε λειτουργία.

**4.2** Για τα σήματα  $S_0$ ,  $S_1$ ,  $S_2$  και  $S_3$  που είναι συνδεδεμένα με τα bits 0, 1, 2 και 3 της πόρτας εισόδου 05H το bit 2 (C2) της πόρτας εξόδου 34H να καθορίζεται σαν το αποτέλεσμα της λογικής συνάρτησης:

$$C2 = S_2 \bar{S}_2 S_0 + \bar{S}_2 S_1 \bar{S}_0 + S_2 S_1 S_0$$

Γράψτε ένα πρόγραμμα που ελέγχει το C2. Το πρόγραμμα να ελέγχει συνεχώς τα σήματα εισόδου και να καθορίζει αντιστοίχα το C2 αφήνοντας το υπόλοιπα bits της πόρτας 34H ανεπηρέαστα.

**4.3** Σχεδιάστε την συνδεσμολογία ανάμεσα σε επτά συσκευές εισόδου και έναν  $\mu\text{Y}-\Sigma$  8085A το οποίο μεταφέρει τα δεδομένα κάνοντας χρήση προγραμματιζόμενης I/O. Οι συσκευές είναι αριθμημένες από 1 έως 7, και η συσκευή 7 έχει την μεγαλύτερη προτεραιότητα. Χρησιμοποιήστε έναν 74LS148 κωδικοποιητή προτεραιότητας για να πετύχετε την ρύθμιση της προτεραιότητας. Το 74LS148 έχει 8-ενεργές εισόδους αρνητικής λογικής και 3 εξόδους. Η είσοδος 7 αυτού του κωδικοποιητή έχει την μεγαλύτερη προτεραιότητα. Η έξοδος του κωδικοποιητή είναι το συμπλήρωμα του δυαδικού ισοδύναμου της ενεργού εισόδου με την μεγαλύτερη προτεραιότητα.

α) Σχεδιάστε ένα λογικό διάγραμμα της συνδεσμολογίας των flip-flop δεδομένων με τον κωδικοποιητή προτεραιότητας και μιάς απλής πόρτας εισόδου που παρέχει το  $\mu\text{Y-}\Sigma$  για το διάβασμα της κατάστασης των 7 συσκευών εισόδου.

β) Γράψτε ένα πρόγραμμα που ελέγχει την κατάσταση και κάνει άλμα στην κατάλληλη περιοχή προγράμματος για να εξυπηρετήσει τη συσκευή αυτή.

**4.4** Καθορίστε τις αναγκαίες λέξεις ελέγχου για να τεθεί ένα 8255 στο mode 0 με την ακόλουθη λειτουργία.

α) Θύρα A, είσοδος; Θύρες B and C έξοδος

β) Θύρες A and C lower, έξοδος; Θύρες B and C upper, είσοδος

γ) Θύρες A and C upper, είσοδος; Θύρες B and C lower, έξοδος

δ) Θύρες A and B, είσοδος; Θύρα C, έξοδος

**4.5** Ένα 8155 χρησιμοποιείται με την πόρτα A σαν έξοδο, την πόρτα B σαν είσοδο και την πόρτα C για χειραψία στις θύρες A και B. Το 8155 ενεργοποιείται όταν  $A_{15}-A_8=00110XXX$ . Γράψτε την ακολουθία εντολών που προγραμματίζουν το 8155 για τις παραπάνω λειτουργίες.

**4.6** Σχεδιάστε ένα  $\mu\text{Y-}\Sigma$  8085 που να έχει τον εξής χάρτη μνήμης:

0000-1FFF Hex	: EPROM
2000-3FFF Hex	: RAM (Memory map I/O)
4000 Hex	: θύρα εξόδου
8000-9FFF Hex	: EPROM (Standard I/O)
0040 Hex	: θύρα εισόδου

Παρέχονται τα εξής ολοκληρωμένα κυκλώματα: EPROMs 8KBytes, RAMs 2KBytes,  $\mu\text{E}$  8085, καταχωρητές και απομονωτές των 8 bits, κωδικοποιητές 3 σε 8 και πύλες.

**4.7** Να γραφεί πρόγραμμα που να υλοποιεί τον παρακάτω αλγόριθμο:

$$Y_n = X_n - \frac{Y_{n-1}}{4}$$

Υποθέτουμε ότι τα δεδομένα  $X_n$  παρέχονται από την θύρα εισόδου A και τα αποτελέσματα  $Y_n$  στην θύρα εξόδου B στις δεκαεξαδικές διευθύνσεις 1000 και 2000 αντίστοιχα. Θεωρούμε επίσης τους αριθμούς  $X_n$ ,  $Y_n$  8ψήφιους σε μορφή συμπληρώματος ως προς 2 και ότι δεν παρατηρείται υπερχείλιση. Το πρόγραμμα να γραφεί σε συμβολική γλώσσα (assembly) του  $\mu\text{E}$  8085.

Κάθε εντολή που θα χρησιμοποιηθεί πρέπει να επεξηγηθεί στο χώρο των σχολίων. Θεωρούμε ότι κάθε νέο δεδομένο  $X_n$  εμφανίζεται αυτομάτως στη θύρα A μετά την έξοδο του  $Y_{n-1}$ . Η αρχική τιμή του  $Y_0$  για την εκκίνηση του αλγόριθμου θεωρείται μηδενική.

**4.8** Να σχεδιαστεί ένα πλήρες  $\mu Y$ -Σ (8085) που να προσθέτει 16 bits τα οποία είναι του ίδιου βάρους και παρέχονται παράλληλα. Είναι δυνατόν να αποφευχθεί η χρήση μνήμης RAM; Απαντήστε ειδικά και γενικά.

**4.9** Ένα σύστημα που βασίζεται στον  $\mu E$ -8085 παρακολουθεί και απεικονίζει θερμοκρασίες από  $0^\circ C$  ως  $99^\circ C$ . Το  $\mu Y$ -Σ διαθέτει μια θύρα εισόδου (PORT1 - Διεύθυνση 30H) από την οποία λαμβάνει σε δυαδική μορφή των 7 bits τη θερμοκρασία. Το 8ο bit (MSB) γίνεται "1" για να δείξει ότι τα δεδομένα είναι έγκυρα και για κάθε νέο δεδομένο πρέπει πρώτα να γίνει "0" και στη συνέχεια "1". Το  $\mu Y$ -Σ διαθέτει επίσης και μια θύρα εξόδου (PORT2 - Διεύθυνση 10H) για την απεικόνιση των δύο ακεραίων δεκαδικών ψηφίων του μέσου όρου 128 συνεχόμενων μετρήσεων της θερμοκρασίας.

- α) Να δοθεί το πλήρες σύστημα (αναλυτική σχεδίαση του κυκλώματος και το πρόγραμμα εφαρμογής). Να χρησιμοποιηθούν μνήμες 1K.
- β) Μπορεί να αποφευχθεί η χρήση μνήμης RAM;
- γ) Ποια πρέπει να είναι η ταχύτητα εισόδου και εξόδου των δεδομένων; Να ληφθεί για χρόνο κατάστασης του  $\mu E$  200ns. Τέλος αν εμφανιστούν τιμές θερμοκρασίας από  $100^\circ C$  ως  $127^\circ C$  στο μέσο όρο να ληφθούν ίσες με  $49^\circ C$ .

**4.10 α)** Σχεδιάστε το κύκλωμα διασύνδεσης του  $\mu Y$  8085 με μια πόρτα εισόδου (διεύθυνση 60H) και μια πόρτα εξόδου (διεύθυνση 80H). Σημειώστε ότι τα βασικά σήματα ελέγχου του 8085 είναι  $\overline{RD}$ ,  $\overline{WR}$ ,  $IO/\overline{M}$  και ALE.

β) Στο παραπάνω  $\mu Y$ -Σ, γράψτε πρόγραμμα που να διαβάζει την δυαδική ένδειξη μιας συσκευής και να την συγκρίνει με δύο κατώφλια K1 και K2 με  $K1 < K2$ , που οι τιμές τους βρίσκονται στους καταχωρητές B και C αντίστοιχα. Στην συνέχεια να ανάβει ένα από τα τρία LED εξόδου που αντιστοιχούν στις περιοχές τιμών  $[0..K1]$ ,  $(K1..K2]$  και  $(K2..K3]$ .

**4.11** Δίνεται ένα  $\mu Y$ -Σ, που διαθέτει μια πόρτα εισόδου και μια πόρτα εξόδου 8 ψηφίων με συμβολικές διευθύνσεις PORTB και PORTA αντίστοιχα. Ακολουθία δεδομένων  $X_n$  παρέχεται στην πόρτα εισόδου. Ζητείται να γίνει η υλοποίηση μέσω προγράμματος ενός καταχωρητή ολίσθησης 128 θέσεων, που αμέσως μετά την είσοδο του  $X_n$  να δίνει το δεδομένο  $X_{n-128}$  στην πόρτα εξόδου συνεχώς. Κατά την εκκίνηση του αλγορίθμου,

υποθέτουμε ότι ο καταχωρητής έχει μηδενικές αρχικές τιμές. Αν σε οποιαδήποτε στιγμή, το  $X_n$  πάρει την τιμή FFH, τότε η διαδικασία σταματάει, στην έξοδο παρέχεται ο μέσος όρος των 128 τελευταίων τιμών και οι τιμές του καταχωρητή ολίσθησης μηδενίζονται. Δώστε την ταχύτερη δυνατή υλοποίηση.