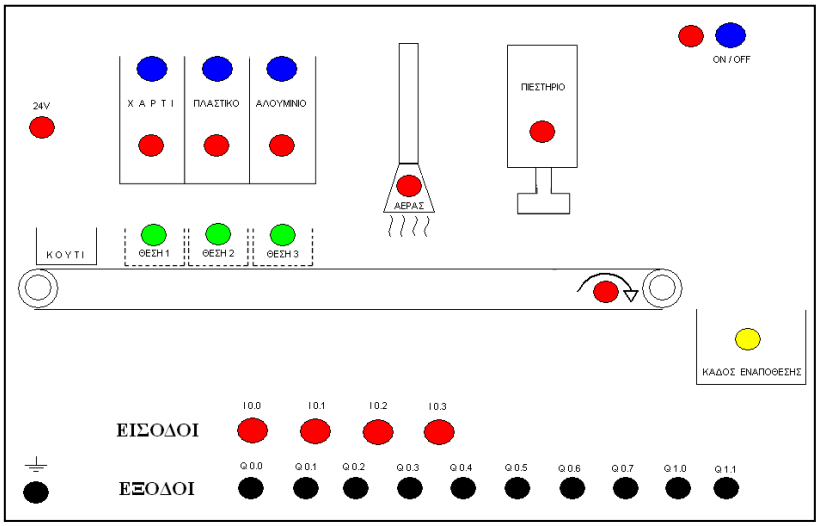
ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ

ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

« ΕΛΕΓΧΟΣ ΣΥΣΤΗΜΑΤΟΣ ΕΠΕΞΕΡΓΑΣΙΑΣ ΠΡΟΪΟΝΤΩΝ ΑΝΑΚΥΚΛΩΣΗΣ ΔΙΑΧΩΡΙΣΜΟΣ ΚΑΙ ΚΑΤΑΝΟΜΗ ΑΥΤΩΝ ΣΕ ΕΙΔΙΚΟΥΣ ΧΩΡΟΥΣ »



ΣΠΟΥΔΑΣΤΗΣ : ΚΩΝΣΤΑΝΤΙΝΟΣ ΜΠΑΛΛΑΣ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΧΡΗΣΤΟΣ Κ. ΜΑΝΑΒΗΣ

ΘΕΣΣΑΛΟΝΙΚΗ 2010

Τίτλος πτυχιακής εργασίας: Έλεγχος συστήματος επεξεργασίας προϊόντων ανακύκλωσης διαχωρισμός και κατανομή αυτών σε ειδικούς χώρους

Σπουδαστής: Κωνσταντίνος Μπαλλάς

Επιβλέπων: Χρήστος Μανάβης

Κωδικός πτυχιακής: 09303ΕΜ

Ημερομηνία ανάληψης: 4-3-2010

Ημερομηνία περάτωσης: 20-5-2010

ΠΕΡΙΛΗΨΗ

Το σύστημα μας αποτελείται από τρία δοχεία με υλικά που πρόκειται να ανακυκλωθούν.

Το πρώτο δοχείο γεμίζει με χαρτί, το δεύτερο με πλαστικό και το τρίτο με αλουμίνιο.

Σε κάθε δοχείο υπάρχει ένας αισθητήρας ώστε να μπορούμε να ξέρουμε πότε γεμίζει το κάθε δοχείο. Όταν γεμίσει το πρώτο δοχείο, με τη βοήθεια ενός ιμάντα ένα κουτί τοποθετείται κάτω από το κάθε δοχείο. Τη στιγμή που το κουτί βρίσκεται κάτω από το δοχείο , ο πάτος στο δοχείο ανοίγει και αδειάζει το υλικό μέσα στο κουτί. Μετά την πάροδο της διαδικασίας του αδειάσματος, το κουτί μεταφέρεται στη θέση του καθαρισμού και στη συνέχεια πηγαίνει στη θέση συμπίεσης όπου και συμπιέζεται, ώσπου στο τέλος το κουτί καταλήγει στον κάδο εναπόθεσης.

SUMMARY

Our system is constituted by three containers with materials that are to be recycled.

The first container fills with paper, second with plastic and third with aluminium.

In each container exists a sensor so that we can know when it fills the each container. When it fills the first container, with the help of belt a box is placed under the each container. The moment where the box is found under the container, floor in the container it opens and throw the material in the box. Afterwards the byway of process of emptying, the box is transported in the place of cleaning and then it goes to the place of compaction where also is compressed, until at the end the box leads to the bucket of deposition.

ΠΡΟΛΟΓΟΣ

Η πτυχιακή αυτή εργασία που έχει τίτλο **"ΕΛΕΓΧΟΣ ΣΥΣΤΗΜΑΤΟΣ ΕΠΕΞΕΡΓΑΣΙΑΣ ΠΡΟΪΝΤΩΝ ΑΝΑΚΥΚΛΩΣΗΣ ΔΙΑΧΩΡΙΣΜΟΣ ΚΑΙ ΚΑΤΑΝΟΜΗ ΑΥΤΩΝ ΣΕ ΕΙΔΙΚΟΥΣ ΧΩΡΟΥΣ "**, δημιουργήθηκε μετά από μια επίπονη μελέτη και μια συλλογή υλικού πάνω στα ψηφιακά συστήματα ελέγχου στο εργαστήριο των Σ.Α.Ε.

Το περιεχόμενο της πτυχιακής αναλύεται σε 4 κεφάλαια.

Στο πρώτο κεφάλαιο αναφέρουμε βασικές έννοιες που αφορούν το PLC (προγραμματιζόμενος λογικός ελεγκτής), βασικά στοιχεία και μέρη του PLC, τις βασικές του λειτουργίες και το λειτουργικό του μέρος.

Στο δεύτερο κεφάλαιο αναφέρουμε τις μορφές και τις γλώσσες προγραμματισμού του PLC. Σ' αυτό το κεφάλαιο συγκρίνουμε τις μορφές προγραμματισμού, αναφέροντας τα πλεονεκτήματα και τα πλεονεκτήματα και καταλήγουμε σε κάποια συμπεράσματα.

Στο τρίτο κεφάλαιο αναφέρουμε αναλυτικά τις εντολές προγραμματισμού για το PLC S7-200 οι οποίες χρησιμοποιήθηκαν προκειμένου να έχουμε την επιτυχή λειτουργία της εργασίας μας.

Στο τέταρτο και τελευταίο κεφάλαιο, αναφερόμαστε σε όλα τα στάδια του προγραμματισμού τα οποία δημιουργήθηκαν ώστε να έχουμε την επιτυχή λειτουργία της κατασκευής μας με το πρόγραμμα Micro Win.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΕΡΙΛΗΨΗ……..….4

ΚΕΦΑΛΑΙΟ 1 : ΕΛΕΓΚΤΕΣ ΠΡΟΓΡΑΜΜΑΤΙΖΟΜΕΝΗΣ ΛΟΓΙΚΗΣ (PLC)……5

1.1 Ιστορική αναδρομή……...5

1.2 Το PLC……..6

1.2.1 Αισθητήρες……...8

1.2.2 Ενεργοποιητές……..……..8

1.2.3 Μονάδα τροφοδοσίας…….……8

1.2.4 Κεντρική μονάδα επεξεργασίας (CPU)……..……9

1.2.5 Μονάδες εισόδων – εξόδων……..…10

1.2.6 Μονάδες ψηφιακών εισόδων……..……..12

1.2.7 Μονάδες ψηφιακών εξόδων…….…12

1.3 Βασικές λειτουργίες των PLC…….……13

1.4 Λειτουργικό σύστημα του PLC……....13

1.4.1 Προσπέλαση προγράμματος……...….14

1.4.2 Δομή προγράμματος……15

1.4.2.1 Τύποι μπλοκ……15

1.4.3 Δομή μπλοκ……..17

ΚΕΦΑΛΑΙΟ 2

2.1 Μορφές προγραμματισμού……..18

2.1.1 Γραμμικός προγραμματισμός……..….18

2.1.2 Μερικός προγραμματισμός…….…..18

2.1.3 Δομημένος προγραμματισμός…….…..…19

2.2 Γλώσσες προγραμματισμού…….……19

2.3 Σύγκριση μορφών προγραμματισμού……..22

2.3.1 Πλεονεκτήματα λίστας εντολών (STL) σε σχέση με τις γραφικές μορφές (LAD, FBD)…….…22

2.3.2 Μειονεκτήματα λίστας εντολών (STL) σε σχέση με τις γραφικές μορφές (LAD, FBD)…….…23

2.4 Συμπεράσματα…….23

ΚΕΦΑΛΑΙΟ 3

3.1 Ανάλυση εντολών προγραμματισμού S7-200…….……25

3.1.1 Εντολές Normally Open – Close…….…25

3.1.2 Η εντολή αντιστροφής NOT……..…….…….27

3.1.3 Εντολές Θετικής - Αρνητικής μετάβασης…….……..28

3.1.4 Οι εντολές Set – Reset……..……30

3.1.5 Ρολόι Πραγματικού Χρόνου(Read, Set, Real-Time Clock).33

3.1.6 Χρονικό με καθυστέρηση στην έναυση (TON)……...34

3.1.7 Χρονικό με καθυστέρηση στην απόζευξη (TOF)……..37

ΚΕΦΑΛΑΙΟ 4

4.1 Στάδια προγραμματισμού……..……39

ΕΠΙΛΟΓΟΣ…….….57
ΒΙΒΛΙΟΓΡΑΦΙΑ…….….58

ΠΕΡΙΛΗΨΗ

Το σύστημα μας αποτελείται από τρία δοχεία με υλικά που πρόκειται να ανακυκλωθούν.

Το πρώτο δοχείο γεμίζει με χαρτί, το δεύτερο με πλαστικό και το τρίτο με αλουμίνιο.

Σε κάθε δοχείο υπάρχει ένας αισθητήρας ώστε να μπορούμε να ξέρουμε πότε γεμίζει το κάθε δοχείο. Όταν γεμίσει το πρώτο δοχείο, με τη βοήθεια ενός ιμάντα ένα κουτί τοποθετείται κάτω από το κάθε δοχείο. Τη στιγμή που το κουτί βρίσκεται κάτω από το δοχείο , ο πάτος στο δοχείο ανοίγει και αδειάζει το υλικό μέσα στο κουτί. Μετά την πάροδο της διαδικασίας του αδειάσματος, το κουτί μεταφέρεται στη θέση του καθαρισμού και στη συνέχεια πηγαίνει στη θέση συμπίεσης όπου και συμπιέζεται, ώσπου στο τέλος το κουτί καταλήγει στον κάδο εναπόθεσης.

ΚΕΦΑΛΑΙΟ 1

ΕΛΕΓΚΤΕΣ ΠΡΟΓΡΑΜΜΑΤΙΖΟΜΕΝΗΣ ΛΟΓΙΚΗΣ (PLC)

1.1 Ιστορική αναδρομή

Οι προγραμματιζόμενοι λογικοί ελεγκτές (PLC – Programmable Logic Controllers) έκαναν την εμφάνιση τους στην αρχή της δεκαετίας του `70 και χρησιμοποιήθηκαν κυρίως για την αντικατάσταση των ρελέ.

Στη δεκαετία του `80 η τεχνολογία γινόταν γρηγορότερη και αναπτυσσόταν συνεχώς, παράλληλα με τις απαιτήσεις του χρήστη. Όπως γίνεται σε όλους τους τομείς έτσι κι εδώ, η επικοινωνία και η πληροφορία έγιναν η σημαντικότερη βάση για να έχουμε μια αποδοτική παραγωγή. Χάρη στις νέες συσκευές μπορούμε να επεξεργαζόμαστε πλέον δεδομένα και να γίνεται ανταλλαγή πληροφοριών μεταξύ τους ή με υπερκείμενους υπολογιστές.

Η τεχνολογία προχωρά. Έτσι φθάνουμε στη δεκαετία του '90 όπου τεχνολογικά έγινε μεγάλο άλμα (συσκευές μικρότερες, φθηνότερες, με σημαντικά αυξημένες δυνατότητες συγκριτικά με αυτές της προηγούμενης δεκαετίας) αλλά παράλληλα αυξήθηκε δυσανάλογα το κόστος εκπόνησης των προγραμμάτων και της θέσης σε λειτουργία των εγκαταστάσεων.

Οι κατασκευαστές ρίχνουν πλέον σημαντικό βάρος στο λογισμικό όπου παρέχονται έτοιμες λύσεις για τομείς του αυτοματισμού με τη βοήθεια βιβλιοθηκών, εκμεταλλεύονται την πρόοδο των ηλεκτρονικών υπολογιστών και χρησιμοποιούν την εξέλιξη στο λειτουργικό τους σύστημα (τεχνολογία Windows) για να μειώσουν τους χρόνους στον προγραμματισμό των PLC.

Έχουμε την εμφάνιση νέων γλωσσών προγραμματισμού για τεχνολόγους σε γραφική μορφή, όπου ο χρήστης μέσω βιβλιοθηκών κι έχοντας γνώση μόνο της παραγωγικής διαδικασίας "συνθέτει" τον αυτοματισμό του. Τα υπόλοιπα γίνονται αυτόματα στο παρασκήνιο για λογαριασμό του. Υποστηρίζεται τέλος και η εξέλιξη στις γλώσσες προγραμματισμού των ηλεκτρονικών υπολογιστών (Pascal, C++) για χρήστες που είναι εξοικειωμένοι σε τέτοια περιβάλλοντα.

Τέλος ιδιαίτερη έμφαση δίνεται πλέον στη δικτύωση - ασύρματη ή ενσύρματο για τον προγραμματισμό / επιτήρηση εξ αποστάσεως μέσω ειδικών συσκευών επικοινωνίας και λογισμικού για ηλεκτρονικό υπολογιστή (SCADA) καθώς και στις επικοινωνίες Internet.

Παρακάτω φαίνονται τα πλεονεκτήματα του PLC

- Ο χρόνος κατασκευής του αυτοματισμού είναι μηδαμινός σε σχέση με την κατασκευή ενός κλασικού πίνακα αυτοματισμού

- Ελαχιστοποίηση του κόστους συντήρησης, εφόσον δεν υπάρχει θέμα βλάβης επειδή τα PLC "χαλάνε" σπάνια.

- Τα PLC είναι "ευέλικτα" στην τροποποίηση της λειτουργίας του αυτοματισμού, εφόσον η αλλαγή στον αυτοματισμό γίνεται σε λίγα λεπτά, αλλάζοντας μόνο το πρόγραμμα.

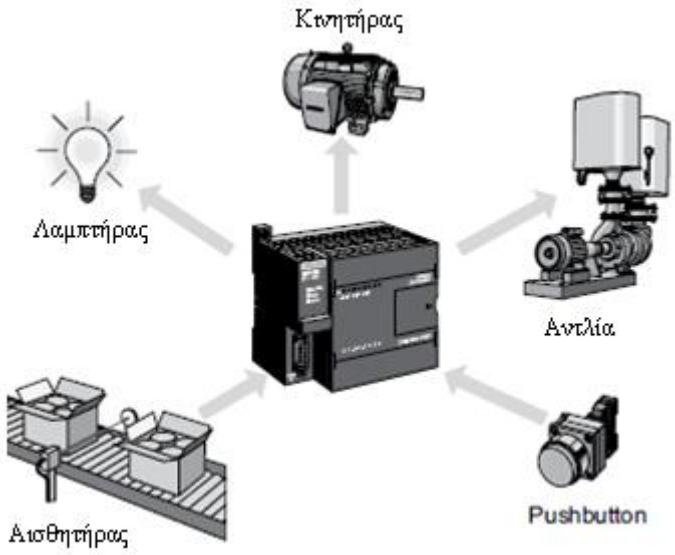
- Αλλάζοντας το πρόγραμμα ή τοποθετώντας νέες μονάδες εισόδων και εξόδων, επεκτείνουμε εύκολα τον αυτοματισμό.

- Μπορούμε να υλοποιούμε πολύπλοκες και έξυπνες επεξεργασίες που στον κλασικό αυτοματισμό είναι εξαιρετικά δύσκολο να γίνουν.

- Η δυνατότητα σύνδεσης του PLC με ηλεκτρονικό υπολογιστή, με το σύστημα αποθήκης, λογιστήριο κλπ είναι επίσης πλεονεκτήματα

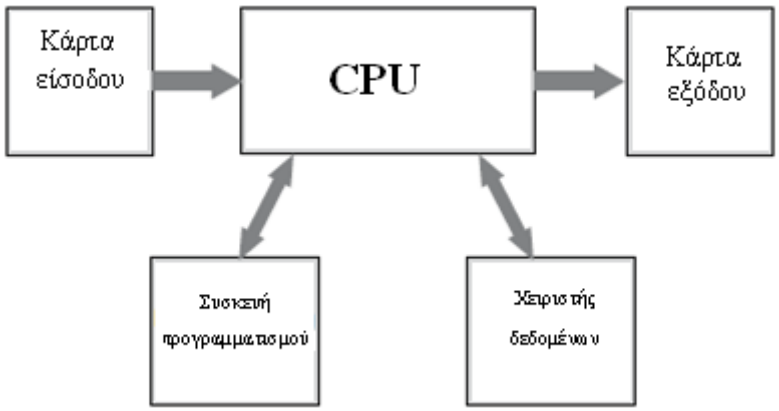
1.2 Το PLC

Το PLC είναι μία ηλεκτρονική διάταξη η οποία έχει εισόδους και εξόδους που συνδέονται με τα στοιχεία μιας εγκατάστασης που ανάλογα με τον αλγόριθμο που εφαρμόζουμε στις εισόδους, έχουμε το επιθυμητό αποτέλεσμα στις εξόδους.



Σχήμα 1.1

Το PLC αποτελείται από την CPU, η οποία περιέχει την λογική του αυτοματισμού και η οποία αφού διαβάσει την κατάσταση των καρτών εισόδου (input modules) ενεργοποιεί τις κάρτες εξόδου (output modules) σύμφωνα με τους κανόνες (πρόγραμμα) που έχουμε αποθηκεύσει στην μνήμη του.



Σχήμα 1.2

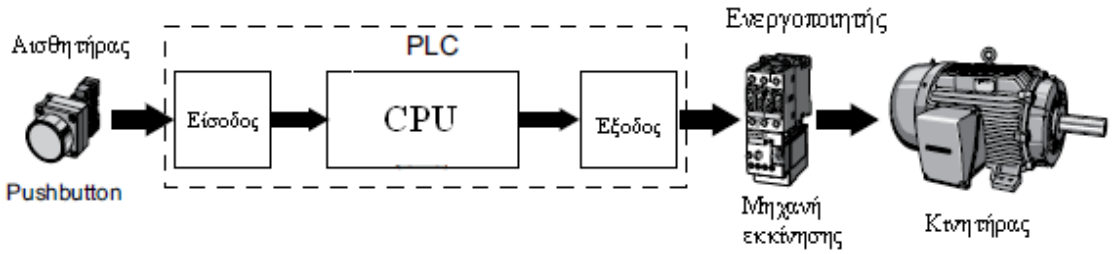
Ωστόσο παρακάτω μπορούμε να δούμε κάποια από τα βασικά στοιχεία του PLC, καθώς και τα βασικά μέρη από τα οποία αποτελείται το PLC.

1.2.1 Αισθητήρες

Οι αισθητήρες είναι συσκευές που μετατρέπουν μία φυσική κατάσταση σε (μια κίνηση ή ένα ερέθισμα) σε ηλεκτρικό σήμα. Οι αισθητήρες συνδέονται στις εισόδους του PLC. Ένα μπουτόν είναι ένα παράδειγμα ενός αισθητήρα που συνήθως συνδέεται με μια είσοδο του PLC. Το ηλεκτρικό σήμα που δείχνει την κατάσταση (ανοικτό ή κλειστό) των επαφών του διακόπτη(push botton), στέλνεται από ένα διακόπτη στο PLC.

1.2.2 Ενεργοποιητές

Οι ενεργοποιητές είναι συσκευές που μετατρέπουν ένα ηλεκτρικό σήμα από ένα PLC, σε μια φυσική κατάσταση. Οι ενεργοποιητές είναι συνδεμένοι στις εξόδους του PLC. Ένας κινητήρας είναι ένα παράδειγμα ενός ενεργοποιητή που συνδέεται σε μια έξοδο του PLC. Ανάλογα με την κατάσταση εξόδου του PLC, η μηχανή εκκίνησης εξασφαλίζει την ενεργεια στον κινητήρα ή αποτρέπει τη ροή ενέργειας στη μηχανή.



Σχήμα 1.3

Τα βασικά μέρη του PLC είναι η μονάδα τροφοδοσίας, η κεντρική μονάδα επεξεργασίας (CPU), και οι μονάδες εισόδων-εξόδων.

1.2.3 Μονάδα τροφοδοσίας

Η μονάδα τροφοδοσίας χρησιμεύει για να υποβιβάσει την τάση του δικτύου σε κάποιες απαραίτητες εσωτερικές τάσεις για την τροφοδοσία αποκλειστικά των ηλεκτρονικών εξαρτημάτων, που υπάρχουν μέσα στον

προγραμματιζόμενο ελεγκτή (τρανζίστορ, ολοκληρωμένα κλπ). Οι τυπικές εσωτερικές τάσεις των ελεγκτών είναι συνήθως: DC 5V, DC 9V, DC 24V.

1.2.4 Κεντρική μονάδα επεξεργασίας (CPU)

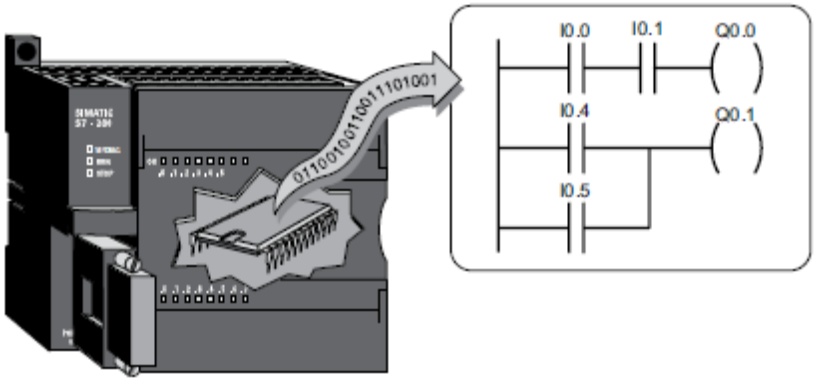
Είναι η βασική μονάδα του PLC, η οποία "ευθύνεται" για τη λειτουργία του αυτοματισμού. Η κεντρική μονάδα επεξεργασίας είναι στην πραγματικότητα ένας μικροϋπολογιστής όπου μπορούμε και διακρίνουμε σ' αυτήν όλα τα κύρια μέρη ενός μικροϋπολογιστή, δηλαδή τον μικροεπεξεργαστή και τη μνήμη. Ο μικροεπεξεργαστής είναι ο αυτός που εκτελεί όλες τις λειτουργίες του PLC.

Η μνήμη της κεντρικής μονάδας επεξεργασίας (CPU) αποτελείται από τη μνήμη RAM, ROM και EEPROM.

Μνήμη RAM: Η μνήμη RAM (Random Access Memory, μνήμη τυχαίας προσπέλασης) είναι εκείνη όπου μπορούν να γραφτούν και να σβηστούν δεδομένα, και η οποία χάνει τα περιεχόμενα όταν κλείσουμε την τροφοδοσία. Στη μνήμη RAM η κεντρική μονάδα αποθηκεύει μια σειρά από πληροφορίες σε ξεχωριστές περιοχές εργασίας.

Μνήμη ROM: Στη μνήμη ROM (Read Only Memory) ο κατασκευαστής του προγραμματιζόμενου ελεγκτή αποθηκεύει το λειτουργικό σύστημα του PLC, δηλαδή το πρόγραμμα για όλες τις βασικές λειτουργίες που είναι απαραίτητες ώστε να μπορέσει να δουλέψει το PLC.

Μνήμη EEPROM: Επειδή η μνήμη RAM με την απώλεια της τροφοδοσίας χάνει τα δεδομένα της (εκτός αν χρησιμοποιείται μπαταρία), τα PLC χρησιμοποιούν έναν άλλο τύπο μνήμης, την EEPROM (Electrically Erasable Programmable Read Only Memory), η οποία προγραμματίζεται και σβήνει ηλεκτρικά. Πρόκειται για μνήμη που με την πτώση της τροφοδοσίας διατηρεί τα δεδομένα της, και η οποία μπορεί να γραφτεί και να σβηστεί μέσω κάποιου ειδικού μηχανήματος.

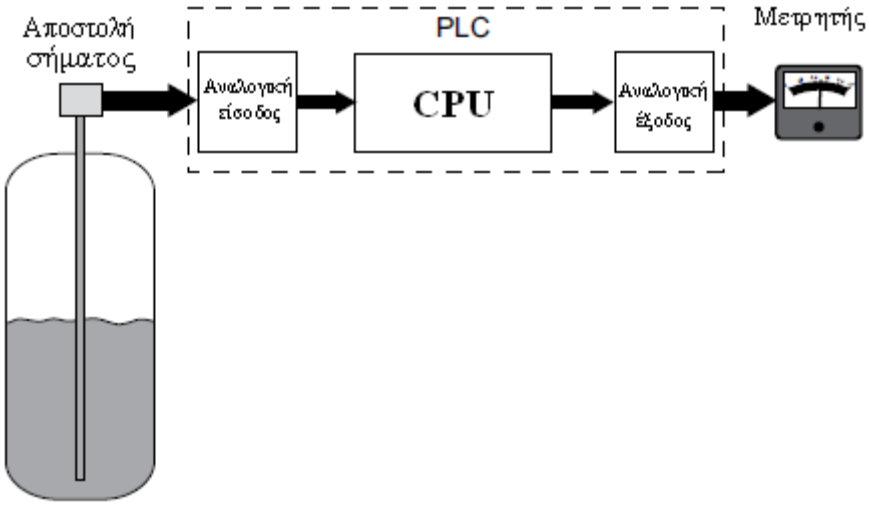


Σχήμα 1.4

1.2.5 Μονάδες εισόδων – εξόδων

Οι μονάδες των εισόδων και των εξόδων αποτελούν τις μονάδες επικοινωνίας μεταξύ της κεντρικής μονάδας και με τους διακόπτες, (μπορεί να είναι αισθητήρες, μπουτόν κ.α.) που δίνουν τις πληροφορίες (εντολές) στη κεντρική μονάδα. Μπορεί να είναι και ρελέ ισχύος των κινητήρων, ηλεκτρομαγνητικές βαλβίδες, ενδεικτικές λυχνίες και γενικά τους αποδέκτες που εκτελούν τις εντολές της κεντρικής μονάδας.

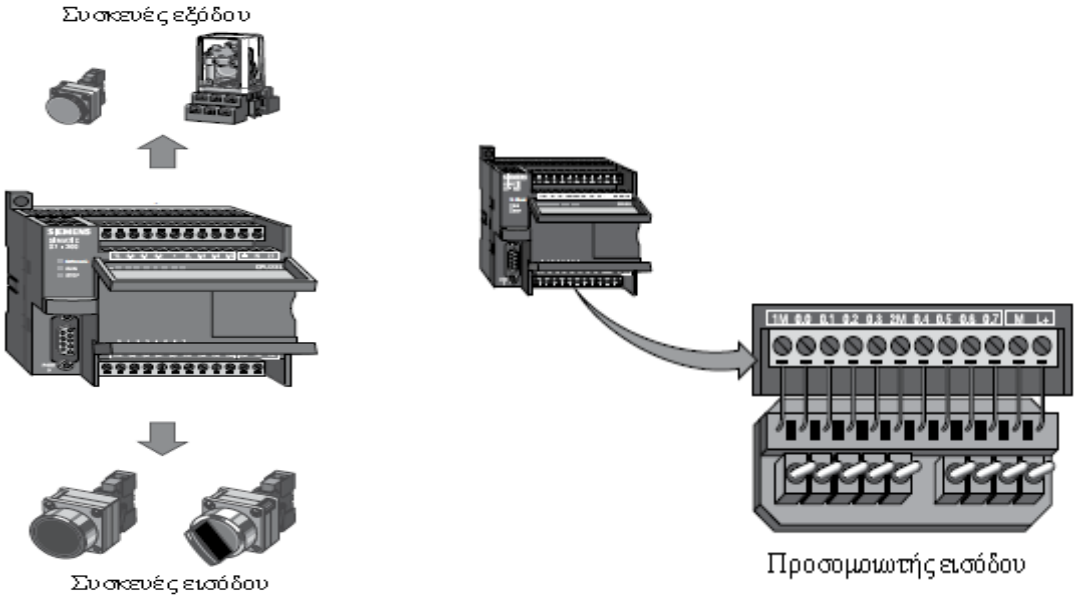
ΠΑΡΑΔΕΙΓΜΑ: Παρακάτω έχουμε ένα παράδειγμα, όπου μια συσκευή ελέγχου που αποστέλλει σήματα, ώστε να ελέγξει το επίπεδο της στάθμης του υγρού σε μια δεξαμενή αποθήκευσης στέλνοντας ένα αναλογικό σήμα σε μια είσοδο του PLC. Μια αναλογική έξοδος από το PLC στέλνει ένα αναλογικό σήμα σε ένα μετρητή για να μας δείξει το επίπεδο υγρού στη δεξαμενή. Δύο άλλες αναλογικές έξοδοι, που δεν φαίνονται εδώ, συνδέονται με τους τρέχων πνευματικούς μετατροπείς που ελέγχουν τις βαλβίδες. Αυτό επιτρέπει στο PLC να ελέγξει αυτόματα τη ροή του υγρού μέσα και έξω από τη δεξαμενή αποθήκευσης.



Σχήμα 1.5

Η κεντρική μονάδα μπορεί να δεχτεί ψηφιακά σήματα εισόδου και εξόδου χαμηλής τάσης και πολύ μικρού ρεύματος. Η τάση που δέχεται είναι συνήθως 0 Volt για το λογικό "0" και 5 Volt για το λογικό "1". Το ρεύμα εισόδου καθώς και το ρεύμα εξόδου είναι συνήθως μερικά mA. Οι μονάδες εισόδων και εξόδων ‘‘αναλαμβάνουν’’ να προσαρμόσουν τα σήματα εισόδου και εξόδου, που έχουμε στον αυτοματισμό, σε σήματα που να μπορεί να δεχτεί η κεντρική μονάδα, τόσο από άποψη τάσεων όσο και από άποψη ρευμάτων. Για να γίνει αυτό, έχουμε τη χρήση ηλεκτρονικών στοιχείων ισχύος, είτε με τη χρήση κατάλληλων μικρό-ρελέ.

Κάθε σύστημα PLC καταλήγει πάντα σε ακροδέκτες. Οι ακροδέκτες αυτοί ανήκουν στις μονάδες εισόδων και εξόδων του. Στους ακροδέκτες εισόδων καταλήγουν οι αγωγοί που έρχονται από αισθητήρες ή κάποιους τερματικούς διακόπτες, διακόπτες μπουτόν, κτλ. Στους ακροδέκτες εξόδων καταλήγουν οι αγωγοί που τροφοδοτούν πηνία ρελέ ισχύος, ηλεκτρομαγνητικές βαλβίδες, λυχνίες ένδειξης κ.ά.



Σχήμα 1.6

1.2.6 Μονάδες ψηφιακών εισόδων

Ένας ελεγκτής αντιλαμβάνεται ότι μια εξωτερική επαφή (π.χ.τερματικός) έκλεισε, όταν στην αντίστοιχη "κλέμα" εισόδου εμφανίζεται τάση. Η τάση αυτή ονομάζεται τάση εισόδων.

Η τάση για την τροφοδοσία των εισόδων δεν δημιουργείται από τη μονάδα τροφοδοσίας του ελεγκτή, αλλά πρέπει να τη δημιουργήσουμε εμείς με κατάλληλο τροφοδοτικό (για DC) ή μετασχηματιστή τάσης χειρισμού (για AC). Εξαίρεση αποτελούν συνήθως οι πολύ μικροί ελεγκτές, στους οποίους ο κατασκευαστής μπορεί να έχει ενσωματώσει ένα μικρό τροφοδοτικό.

1.2.7 Μονάδες ψηφιακών εξόδων

Οι μονάδες ψηφιακών εξόδων χρησιμεύουν για τη διέγερση των εξωτερικών στοιχείων της εγκατάστασης, ρελέ κινητήρων, βαλβίδες, ενδεικτικές λυχνίες κλπ. Όταν από το πρόγραμμα δοθεί εντολή για τη διέγερση ενός π.χ. εξωτερικού ρελέ, τότε κλείνει ο αντίστοιχος "διακόπτης" της εξόδου. Η τάση εμφανίζεται στην "κλέμα" εξόδου και το ρελέ οπλίζει. Η τάση αυτή ονομάζεται τάση

εξόδων. Ο "διακόπτης" εξόδου είναι συνήθως ηλεκτρονικός (τρανζίστορ, triac), αλλά μπορεί να είναι και μηχανική επαφή μικρορελέ.

Η τάση για την τροφοδοσία των μονάδων εξόδων δεν δημιουργείται από τη μονάδα τροφοδοσίας του ελεγκτή, αλλά πρέπει να τη δημιουργήσουμε εμείς με κατάλληλο τροφοδοτικό (για DC) ή μετασχηματιστή τάσης χειρισμού (για AC).

1.3 Βασικές λειτουργίες των PLC

Παρακάτω αναφέρονται κάποιες από τις βασικές λειτουργίες των PLC, οι οποίες βοήθησαν το PLC να εξελιχθεί με αρκετά γρήγορους ρυθμούς.

Λειτουργία απαρίθμησης. Γίνεται με εσωτερικούς ή εξωτερικούς παλμούς και να είναι προς τα πάνω (count up) ή προς τα κάτω (count down).

Δυνατότητα πραγματικού ρολογιού. Μας δίνει τη δυνατότητα να προγραμματίσουμε το PLC σε πραγματικό χρόνο, ώρα και ημερομηνία.

Αριθμητικές ρυθμίσεις. Από τη στιγμή που συνδέονται με ηλεκτρονικό υπολογιστή έχουν τη δυνατότητα να επεξεργάζονται και αριθμητικές πράξεις.

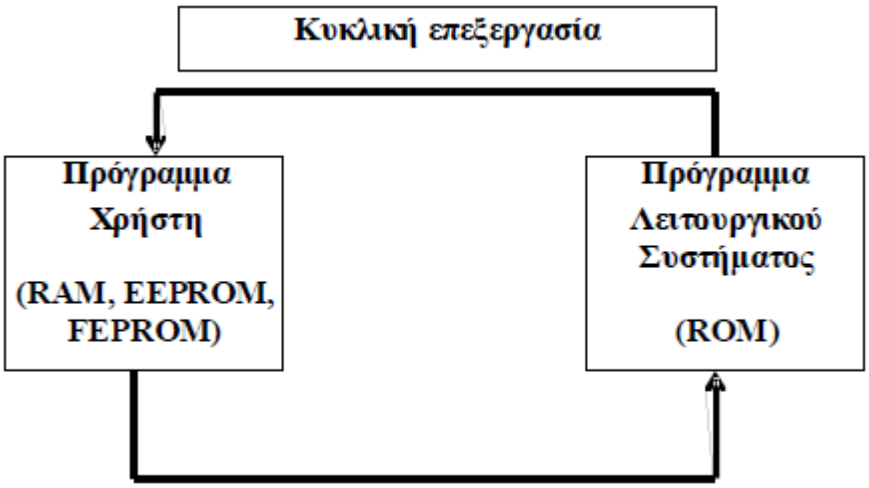
1.4 Λειτουργικό σύστημα του PLC

Ορισμένες ενέργειες του ελεγκτή γίνονται αυτόματα, χωρίς καμία απολύτως εντολή από το χρήστη, όπως π.χ.

- Όταν επανέρχεται η τάση μετά από μία διακοπή του δικτύου γίνεται μηδενισμός των βοηθητικών που ανήκουν στην περιοχή μνήμης χωρίς συγκράτηση.
- Πριν από την έναρξη κάθε κύκλου προγράμματος γίνεται μεταφορά σημάτων από τις κλέμες των μονάδων εισόδων στη μνήμη απεικόνισης καταστάσεων εισόδων (Process - Image Input Register).

- Μετά το τέλος κάθε κύκλου προγράμματος γίνεται μεταφορά της μνήμης απεικόνισης καταστάσεων εξόδων (Process - Image Output Register) στις αντίστοιχες κλέμες των μονάδων εξόδων.

Όλες αυτές οι απαραίτητες ενέργειες, οι οποίες προσδιορίζουν τι παραπάνω πρέπει να κάνει ο ελεγκτής παράλληλα με το κυρίως πρόγραμμα του χρήστη, το οποίο εμείς προγραμματίζουμε, αποτελούν το λειτουργικό πρόγραμμα του ελεγκτή. Το πρόγραμμα αυτό είναι συνήθως αποθηκευμένο σε μια μνήμη ROM μέσα στην κεντρική μονάδα και περιέχει εντολές που δεν μπορούν να διαβαστούν από εμάς, και οι οποίες καθορίζουν τις παραπάνω αντιδράσεις.



Σχήμα 1.4

1.4.1 Προσπέλαση προγράμματος

Το γενικό πρόγραμμα μιας κεντρικής μονάδας αποτελείται από το λειτουργικό σύστημα και το πρόγραμμα του χρήστη.

Το **λειτουργικό σύστημα** όπως προαναφέραμε, αποτελεί το σύνολο που περιέχει όλες τις εντολές και τις δηλώσεις που ελέγχουν τις πηγές του συστήματος, τις διαδικασίες που χρησιμοποιούν αυτές τις πηγές, καθώς και περιοχές λειτουργίας όπως αποθήκευση δεδομένων στην περίπτωση πτώση της τάσης του δικτύου, ενεργοποίηση τάξεων προτεραιότητας, κλπ. Το λειτουργικό σύστημα αποτελεί ένα μέρος της

κεντρικής μονάδας, στο οποίο ο χρήστης δεν έχει πρόσβαση γραφής. Εντούτοις, μπορούμε να φορτώσουμε ξανα το σύστημα αυτό από μια μονάδα μνήμης, π.χ. στην περίπτωση της ενημέρωσης με τις τελευταίες αλλαγές του προγράμματος.

Το **πρόγραμμα του χρήστη** αποτελεί το σύνολο όλων των εντολών και δηλώσεων, στην περίπτωση αυτή τα στοιχεία του προγράμματος, για την οδήγηση των σημάτων, μέσα από την οποία η όλη διαδικασία επηρεάζεται ανάλογα με την προκαθορισμένη εργασία ελέγχου.

1.4.2 Δομή προγράμματος

Μπορούμε να χωρίσουμε το πρόγραμμα σε όσα μέρη θέλουμε με σκοπό να το διαβάζουμε και να το αντιλαμβανόμαστε καλύτερα και ευκολότερα. Κάθε μέρος του προγράμματος πρέπει να έχει τεχνολογική και λειτουργική βάση. Αυτού του είδους τα μέρη ονομάζονται "Μπλοκ". Ένα μπλοκ αποτελεί ένα μέρος του προγράμματος του χρήστη που καθορίζεται από τις λειτουργίες του, τη δομή και τον σκοπό της ύπαρξής του.

1.4.2.1 Τύποι μπλοκ

- Μπλοκ χρήστη.

Τα μπλοκ αυτά περιέχουν το πρόγραμμα και τα δεδομένα του χρήστη.

- Μπλοκ συστήματος

Τα μπλοκ αυτά περιέχουν το πρόγραμμα και τα δεδομένα του συστήματος.

- Στάνταρτ μπλοκ

Τα μπλοκ αυτά αποτελούν το κλειδί λειτουργίας των οδηγών (drivers) των ειδικών καρτών CP και FM.

Μπλοκ χρήστη. Τα μεγάλα και περίπλοκα προγράμματα «δομούνται» (διαχωρίζονται) σε μπλοκ τα οποία εν μέρη είναι απαραίτητα. Μπορούμε να διαλέξουμε μεταξύ των διαφόρων τύπων των μπλοκ, ανάλογα με την εφαρμογή:

- **Μπλοκ οργάνωσης (OB).** Τα προαναφερόμενα μπλοκ συμβάλουν στην επικοινωνία μεταξύ του λειτουργικού συστήματος και του προγράμματος του χρήστη. Οι κεντρικές μονάδες επεξεργασίας καλούν τα μπλοκ οργάνωσης όταν συγκεκριμένα γεγονότα λαμβάνουν χώρα, π.χ. στην περίπτωση διακοπής.

- **Μπλοκ λειτουργίας (FB).** Αποτελούν μέρος του προγράμματος του οποίου οι κλήσεις μπορούν να προγραμματιστούν μέσω παραμέτρων του μπλοκ. Οι μεταβλητές μνήμης που περιέχονται σε ένα μπλοκ δεδομένων το οποίο με την σειρά του περιλαμβάνεται στην **κλήση** του μπλοκ λειτουργίας. Επίσης είναι δυνατόν σε κάθε κλήση να περιέχεται και διαφορετικό μπλοκ δεδομένων (με την ίδια δομή δεδομένων άλλα διαφορετικές τιμές μεταβλητών).

- **Λειτουργία (Fc).** Οι λειτουργίες χρησιμοποιούνται για το προγραμματισμό περίπλοκων αυτόματων λειτουργιών. Μπορούν να παραμετροποιηθούν και να επιστρέψουν μια τιμή στο καλούμενο μπλοκ. Η τιμή της λειτουργίας είναι προαιρετική ενώ οι μπορούν, επίσης, να στέλνουν προς τα έξω διαφορετικές παραμέτρους. Οι λειτουργίες δεν αποθηκεύουν πληροφορίες και δεν περιέχουν μπλοκ δεδομένων.

- **Μπλοκ δεδομένων (DB).** Αυτά τα μπλοκ περιέχουν τα δεδομένα του προγράμματός μας. Προγραμματίζοντάς τα καθορίζουμε σε ποια μορφή θα σωθούν τα δεδομένα (σε ποιο μπλοκ, με ποια σειρά και με τι τύπο δεδομένων). Υπάρχουν δύο τρόποι χρήσης των μπλοκ δεδομένων: ως καθολικά και ως στιγμιαία μπλοκ. Ένα καθολικό μπλοκ δεδομένων είναι ένα "ελεύθερο" μπλοκ μέσα στο πρόγραμμα του χρήστη και δεν περιέχεται σε ένα κωδικοποιημένο μπλοκ. Ένα στιγμιαίο μπλοκ δεδομένων όμως, περιέχεται σε ένα μπλοκ λειτουργίας και αποθηκεύει μέρος των δεδομένων του μπλοκ λειτουργίας. Ο αριθμός των μπλοκ ανά τύπο μπλοκ και το μήκος τους εξαρτάται από την CPU. Οι αριθμοί των μπλοκ οργάνωσης και το πλήθος τους είναι καθορισμένα. Αναθέτονται από το λειτουργικό σύστημα της κεντρικής μονάδας. Μπορούμε να ορίσουμε μόνοι μας τον αριθμό του μπλοκ των άλλων ειδών των μπλοκ, αρκεί αυτός να βρίσκεται μέσα σε καθορισμένα όρια. Επίσης έχουμε την επιλογή να

ονομάσουμε κάθε μπλοκ μέσω του πίνακα συμβόλων και στη συνέχεια να αναφερόμαστε σ' αυτά με το όνομά τους.

- **Μπλοκ συστήματος.** Τα μπλοκ συστήματος αποτελούν μέρος του λειτουργικού συστήματος να περιέχουν προγράμματα (λειτουργίες συστήματος)ή μπλοκ λειτουργιών ή δεδομένα (μπλοκ δεδομένων συστήματος). Τα μπλοκ συστήματος πραγματοποιούν έναν αριθμό από σημαντικές λειτουργίες του συστήματος, προσβάσημες στο χρήστη,όπως είναι ο χειρισμός του εσωτερικού ρολογιού της CPU, ή οι διάφορες λειτουργίες επικοινωνίας. Μπορούμε να καλέσουμε τις λειτουργίες του συστήματος και τα μπλοκ λειτουργιώντου συστήματος, αλλά δεν μπορούμε να τα διαμορφώσουμε ή να τα προγραμματίσουμε. Τα μπλοκ από μόνα τους δεν διατηρούν χώρο στην μνήμη. Μόνο οι κλήσεις των μπλοκ κα τα στιγμιαία μπλοκ δεδομένων των μπλοκ λειτουργιών του συστήματος είναι στην μνήμη.

1.4.3 Δομή μπλοκ

Συνήθως τα μπλοκ αποτελούνται από τρία μέρη:

- Τον αριθμό του μπλοκ που περιέχει τις ιδιότητες του μπλοκ, όπως το όνομα του.
- Το μέρος των δηλώσεων όπου οι τοπικές μεταβλητές του μπλοκ δηλώνονται.
- Το μέρος του προγράμματος όπου περιέχονται οι εντολές του προγράμματος.

ΚΕΦΑΛΑΙΟ 2

2.1 Μορφές προγραμματισμού

Για να αναλύσουμε έναν περίπλοκο αυτοματισμό θα πρέπει να χωρίσουμε την εφαρμογή σε μικρότερα μέρη ανάλογα με την δομή της διαδικασίας που πρέπει να ελεγχθεί. Μετά μπορούμε να διαμορφώσουμε τα επιμέρους κομμάτια καθορίζοντας τις λειτουργίες και διοχετεύοντας τα εσωτερικά σήματα προς την διαδικασία ή άλλα μέρη. Αυτός ο διαχωρισμός μπορεί να εφαρμοστεί και στον προγραμματισμό μας. Μ' αυτόν τον τρόπο η δομή του προγράμματός μας ανταποκρίνεται στον διαχωρισμό της εφαρμογής. Ένα τέτοιο πρόγραμμα μπορεί να διαμορφωθεί πιο εύκολα και να προγραμματιστεί σε μέρη, ακόμα και από διαφορετικά άτομα, στην περίπτωση που το πρόγραμμα είναι πολύ μεγάλο. Τέλος, χωρίζοντας το πρόγραμμα σε μέρη είναι πιο εύκολη η δοκιμή και η αποσφαλμάτωση του. Η δομή του προγράμματος του χρήστη εξαρτάται από το μέγεθος και τις λειτουργίες του. Οι μορφές προγραμματισμού είναι οι εξής:

2.1.1 Γραμμικός προγραμματισμός

Εδώ όλο το κυρίως πρόγραμμα είναι το μπλοκ οργάνωσης OB. Κάθε τρέχον μονοπάτι είναι σε ξεχωριστό **network**. Όταν διορθώνουμε και αποσφαλματώνουμε, μπορούμε να αναφέρουμε το κάθε network απευθείας από τον αριθμό του.

2.1.2 Μερικός προγραμματισμός

Ο μερικός προγραμματισμός βασίζεται στον γραμμικό προγραμματισμό μόνο που το πρόγραμμα χωρίζεται σε μπλοκ. Οι αιτίες για τον διαχωρισμό του προγράμματος σε μικρότερα μέρη είναι είτε το γεγονός ότι το πρόγραμμα είναι πολύ μεγάλο για το OB, είτεεπειδή θέλουμε να διαβάζεται πιο εύκολα. Τα μπλοκ τότε καλούνται με την σειρά. Μπορούμε επίσης να χωρίσουμε το πρόγραμμα ενός μπλοκ σε άλλα μπλοκ όπως κάναμε με το OB. Αυτή η μέθοδος μας επιτρέπει να καλούμε συσχετισμένες λειτουργίες της διαδικασίας μέσα από ένα και το αυτό μπλοκ. Το πλεονέκτημα αυτής της μορφής

προγραμματισμού είναι ότι αν και το πρόγραμμα γραμμικό μπορούμε να το αποσφαλματώσουμε σε μέρη (απλά μόνο καλώντας τα μπλοκ).

2.1.3 Δομημένος προγραμματισμός

Ο δομημένος προγραμματισμός χρησιμοποιείται όταν το επινοημένο σχέδιο είναι εξαιρετικά ακριβό, όταν θέλουμε να δημιουργήσουμε λειτουργίες προγράμματος και όταν πρέπει να λυθούν περίπλοκα προβλήματα. Μ' αυτήν την μέθοδο χωρίζουμε το πρόγραμμα σε κομμάτια (μπλοκ) με ενσωματωμένες λειτουργίες ή σε μπλοκ που εξυπηρετούν έναν συγκεκριμένο σκοπό λειτουργίας και τα οποία ανταλλάσσουν όσο το δυνατόν λιγότερα σήματα με τα άλλα μπλοκ. Αναθέτοντας σε κάθε κομμάτι μια συγκεκριμένη λειτουργία δημιουργούμε ευανάγνωστα μπλοκ με απλούστερη επικοινωνία με τα άλλα μπλοκ. Τέλος, η οργάνωση του προγράμματος καθορίζει την σειρά με την οποία η κεντρική μονάδα επεξεργασίας θα εκτελέσει τα μπλοκ που έχουμε δημιουργήσει. Για να οργανώσουμε το πρόγραμμα μας, προγραμματίζουμε τις κλήσεις των μπλοκ με την σειρά που επιθυμούμε. Η σειρά αυτή θα πρέπει να είναι ανάλογη με την σειρά των επιμέρους λειτουργιών της διαδικασίας που θέλουμε να ελέγξουμε.

2.2 Γλώσσες προγραμματισμού

Τρεις είναι οι μορφές προγραμματισμού που έχουν επικρατήσει διεθνώς:

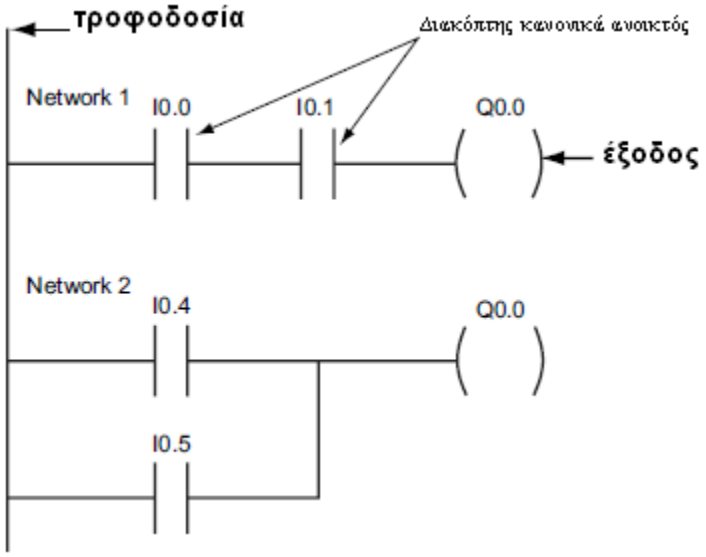
- Λίστα εντολών (STL – Statement List).
- Σχέδιο επαφών (LAD – Ladder Diagram) και
- Διάγραμμα λογικών πυλών (FBD – Function Block Diagram)

1) Η **STL** είναι η γλώσσα προγραμματισμού με τη μορφή κειμένου. Η σύνταξη των εντολών είναι παρόμοια με αυτή του κώδικα μηχανής (Machine Code), όπου οι εντολές και οι λειτουργίες ακολουθούνται από
διευθύνσεις. Η γλώσσα αυτή είναι αυτή που ενδείκνυται για βέλτιστη χρήση της μνήμης και την εκτέλεση του προγράμματος.

```
Network 1
        LD   I0.0
        A    I0.1
        =    Q0.0
Network 2
        LD   I0.4
        O    I0.5
        =    Q0.1
```

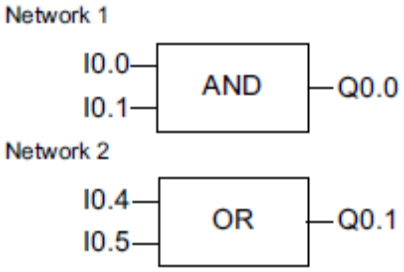
2) Η **LADDER** είναι γλώσσα προγραμματισμού, όπου η σύνταξη των εντολών χρησιμοποιεί τα σύμβολα, αντί των λέξεων, για να μιμηθεί τον πραγματικό έλεγχο λογικής παγκόσμιων ηλεκτρονόμων. Αυτά τα σύμβολα διασυνδέονται από γραμμές για να δείξουν τη ροή του ρεύματος μέσω του ηλεκτρονόμου όπως τις επαφές και τις σπείρες. Κατά τη διάρκεια των χρόνων ο αριθμός συμβόλων έχει αυξηθεί για να παρέχει ένα υψηλό επίπεδο λειτουργίας.

Το ολοκληρωμένο πρόγραμμα μοιάζει με μια "σκάλα"(ladder) αλλά στην πραγματικότητα αντιπροσωπεύει ένα ηλεκτρικό κύκλωμα. Έχουμε τη γραμμή τροφοδοσίας, τους διακόπτες και την έξοδο.



Σχήμα 2.1

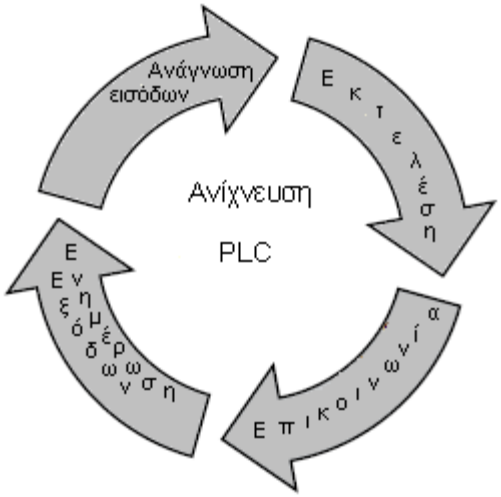
3) Η **FBD** είναι κι αυτή γλώσσα προγραμματισμού με γραφικά. Οι εντολές εδώ αναπαρίστανται με λογικά "κουτιά", παρόμοια με αυτά που συναντάμε στην άλγεβρα Bool.



Σχήμα 2.2

Στο κεφάλαιο 3 θα δούμε αναλυτικά τις εντολές με εικόνες και παραδείγματα.

Ένα πρόγραμμα του PLC εκτελείται ως ένα τμήμα μιας επαναλαμβανόμενης διαδικασίας που αναφέρεται ως **ανίχνευση** . Μια ανίχνευση PLC αρχίζει με την CPU, " διαβάζοντας" την κατάσταση των εισόδων. Έπειτα, το πρόγραμμα εφαρμογής εκτελείται. Η CPU εκτελεί την εσωτερική διάγνωση και τον σκοπό της επικοινωνίας. Τέλος, η CPU ενημερώνει τις εξόδους. Αυτή η διαδικασία επαναλαμβάνεται εφ' όσον η CPU τρέχει το πρόγραμμα(βρίσκεται σε κατάσταση run mode). Ο χρόνος που απαιτείται για να ολοκληρωθεί η ανίχνευση εξαρτάται από μέγεθος του προγράμματος, τον αριθμό των εισόδων και των εξόδων και το απαιτούμενο σύνολο επικοινωνίας.



Σχήμα 2.3

2.3 Σύγκριση μορφών προγραμματισμού

Η "μητρική"γλώσσα κάθε ελεγκτή είναι αναμφίβολα η λίστα εντολών, η οποία έχει και τις μεγαλύτερες δυνατότητες και ευελιξία. Οπωσδήποτε, και οι δύο γραφικές μορφές (σχέδιο επαφών, λογικό διάγραμμα) έχουν το μεγάλο πλεονέκτημα της καλύτερης εποπτείας "με μία "ματιά". Στη συνέχεια θα φαίνονται τα σημαντικότερα πλεονεκτημάτα και μειονεκτημάτα.

2.3.1 Πλεονεκτήματα λίστας εντολών (STL) σε σχέση με τις γραφικές μορφές (LAD, FBD)

- Έχει τις μεγαλύτερες δυνατότητες, γιατί υπάρχουν εντολές, οι οποίες δεν είναι δυνατόν να παρασταθούν γραφικά, αν και στο κοντινό μέλλον αυτό θα διορθωθεί.
- Γνωρίζουμε με απόλυτη ακρίβεια τη σειρά, με την οποία ο μικροεπεξεργαστής επεξεργάζεται το πρόγραμμα (τη μία εντολή ύστερα από την άλλη).
- Καταλαμβάνει μικρότερο χώρο στη μνήμη για την αποθήκευση του προγράμματος.
- Είναι πολύ προσιτή στην χρήση σε όποιον έχει ασχοληθεί ήδη με προγραμματισμό κάθε είδους.
- Μπορούν να χρησιμοποιηθούν μικροί, φτηνοί, φορητοί προγραμματιστές χειρός (ενώ αντίθετα για τη «σχεδίαση» μιας γραφικής μορφής απαιτείται οθόνη, αν θέλουμε να έχουμε εποπτεία).
- Ο χειρισμός κατά την πληκτρολόγηση του προγράμματος είναι πολύ απλούστερος. Αντίθετα, για την πληκτρολόγηση ενός στοιχείου στις γραφικές μορφές, π.χ. μιας επαφής, πρέπει ο δείκτης (cursor) στην οθόνη να βρίσκεται στη σωστή θέση.
- Αν σαν βάση για τον προγραμματισμό χρησιμοποιηθεί ένα κλασσικό συνδεσμολογικό σχέδιο με ρελέ ή ένα λογικό διάγραμμα (flow-chart), τότε η «μετάφραση» τους σε λίστα εντολών είναι το ίδιο εύκολη με την «μετάφραση» τους σε σχέδιο επαφών ή λογικό διάγραμμα αντίστοιχα.

- Πρέπει να τονιστεί, ότι ένα ηλεκτρολογικό συνδεσμολογικό σχέδιο, πολύ σπάνια μπορεί να προγραμματιστεί όπως είναι, χωρίς μετατροπές, σε σχέδιο επαφών.

2.3.2 Μειονεκτήματα λίστας εντολών (STL) σε σχέση με τις γραφικές μορφές (LAD, FBD)

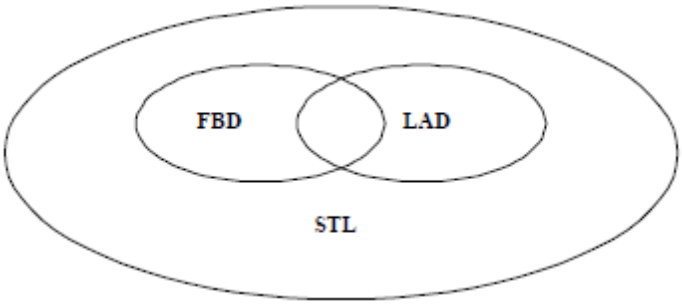
- Ένα πρόγραμμα γραμμένο σε λίστα εντολών δεν έχει την ίδια εποπτεία «με μια ματιά», την οποία έχουν οι γραφικές μορφές. Με τις δυνατότητες όμως σχολιασμού προγράμματος, που παρέχουν οι σύγχρονες συσκευές προγραμματισμού, το μειονέκτημα αυτό παύει να είναι ιδιαίτερα σημαντικό.
- Η παρακολούθηση του αυτοματισμού σε λειτουργία (πάνω σε μια συσκευή προγραμματισμού οθόνης συνδεδεμένη στον ελεγκτή) είναι απλούστερη και πιο εποπτική, αν το πρόγραμμα είναι γραμμένο σε κάποια από τις δύο γραφικές μορφές.

2.4 Συμπεράσματα

Καλό είναι οι ελεγκτές να έχουν τη δυνατότητα να προγραμματιστούν και στις τρεις μορφές που προαναφέρθηκαν και να αφήνεται σε κάποιον που θα φτιάξει το πρόγραμμα η επιλογή της μορφής προγραμματισμού. Θεωρείται αυτονόητο ότι οι τρεις μορφές είναι συμβατές μεταξύ τους, δηλ. σε όποια μορφή κι αν δύο, ζητώντας το από τη συσκευή.

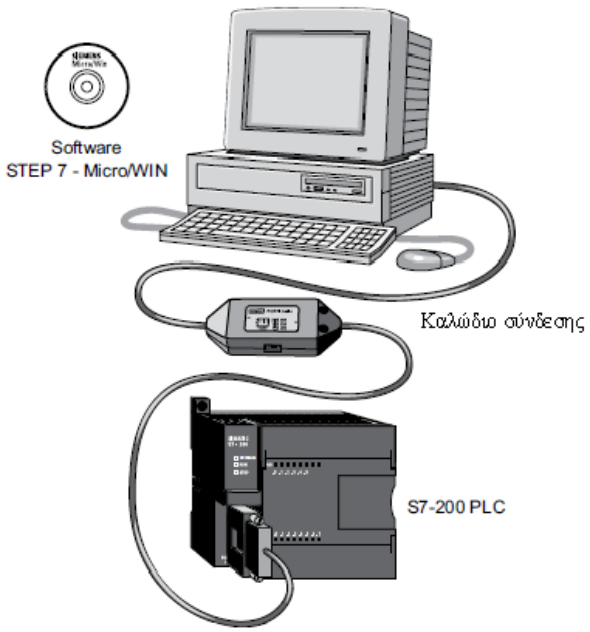
Η χρήση περισσότερων από μία μορφή παράστασης ενός προγράμματος είναι πολλές φορές επιθυμητή και για άλλους λόγους:

Π.χ. σ' ένα μεγάλο εργοστάσιο, αυτός που θα φτιάξει το πρόγραμμα μπορεί να επιλέξει π.χ. τη λίστα εντολών, αλλά η ηλεκτρολογική συντήρηση πιθανόν να προτιμάει στο αρχείο της την παράσταση σχεδίου επαφών, για την ανεύρεση βλαβών, λαμβάνοντας υπ' όψη τα πλεονεκτήματα και τα μειονεκτήματα που προαναφέραμε.



Σχήμα 2.4

Ο προγραμματισμός γίνεται μέσω ενός προγράμματος, του Micro-Win και η συσκευή PLC που θα χρησιμοποιηθεί είναι η **S7-200.**



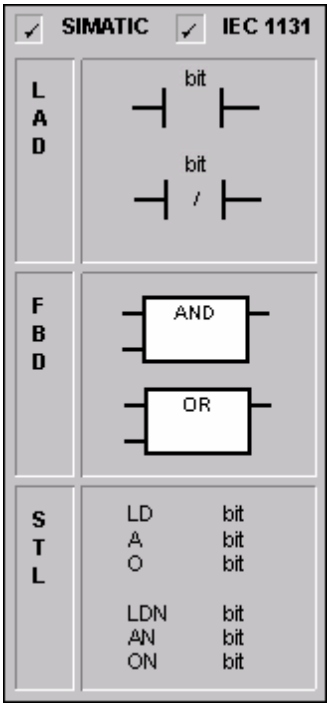
Σχήμα 2.5

ΚΕΦΑΛΑΙΟ 3

3.1 Ανάλυση εντολών προγραμματισμού S7-200

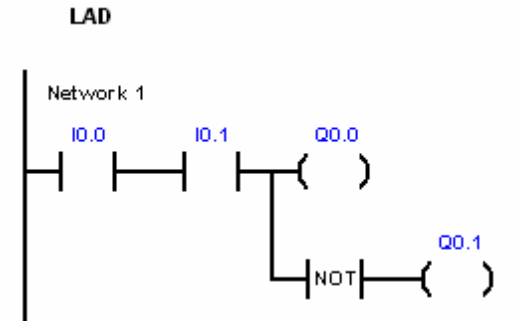
3.1.1 Εντολές Normally Open – Close

Η σύνταξη των εντολών Normally Open – Close στις γλώσσες (LAD/STL/FBD) φαίνεται στην εικόνα 3.1 που ακολουθεί:



Εικόνα 3.1

Σε LAD η Normally Open (LD, A, O) είναι κλειστή όταν το bit της εικόνας 1 είναι ίσο με 1. Η Normally Close (LDN, AN, ON) είναι κλειστή όταντο bit της εικόνας είναι 1 είναι ίσο με 0. Ένα σχηματικό παράδειγμα LAD φαίνεται στο παρακάτω σχήμα 3.1.

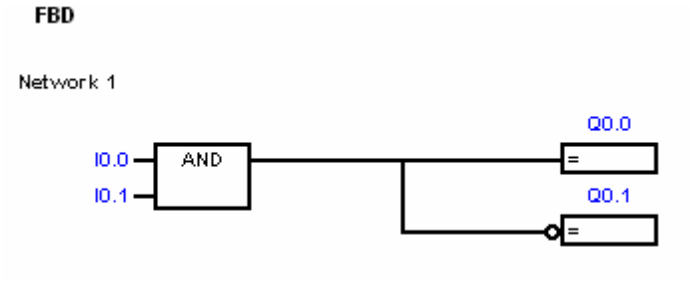


Σχήμα 3.1

Όταν διεγερθεί η I0.0 και η I0.1 τότε δίνει έξοδο στην Q0.0 ενώ δεν δίνει έξοδο στην Q0.1.

Η σύνταξη σε FBD είναι όπως αυτή των λογικών πυλών. Όταν η μια είσοδος της AND είναι 1 **και** η άλλη είναι και αυτή ίση με 1 τότε δίνεται έξοδος σε κάποιο φορτίο. Στην OR σύνταξη, αρκεί έστω μια από τις εισόδους να είναι ίση με 1 για να έχει έξοδο. Στην FBD σύνταξη μπορούν να διαχειριστούν μέχρι 32 είσοδοι οι οποίες θα δίνουν την κατάλληλη έξοδο ανάλογα με την λογική σχέση που τις συνδέει.

Η FBD σύνταξη του παραδείγματος στο σχήμα 3.1 φαίνεται στο σχήμα 3.2 που ακολουθεί:



Σχήμα 3.2

Παρομοίως και εδώ όταν η I0.0 και η I0.1 διεγερθούν τότε παίρνουμε έξοδο στην Q0.0 και όχι στην Q0.1.

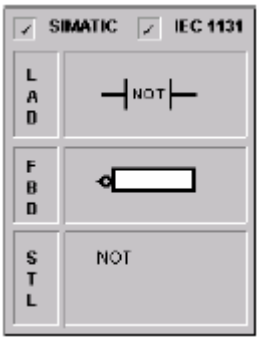
Στην STL η normally open αντιπροσωπεύεται από τις LOAD AND και OR εντολές όπως φαίνεται στην εικόνα 1. Όταν έχει μια είσοδο I0.0 σήμα 1 και μια άλλη είσοδο I0.1 σήμα 1 με σύνδεση AND τότε η έξοδος θα είναι 1. Παρομοίως όταν μια είσοδο I0.0 έχει σήμα 1 και σε μια άλλη είσοδο I0.1 σήμα 0 με σύνδεση OR τότε η έξοδος θα πάλι 1.

Η STL σύνταξη είναι κάπως πιο πολύπλοκη αλλά προσεγγίζει στην γλώσσα μηχανής όπως έχουμε πει σε προηγούμενο κεφάλαιο. Η σύνταξη αυτού του τρόπου φαίνεται παρακάτω.

```
LD    I0.0
A     I0.1
=     Q0.0
  NOT
=     Q0.1
```

3.1.2 Η εντολή αντιστροφής NOT

Η σύνταξη της εντολής NOT στις γλώσσες (LAD/STL/FBD) φαίνεται στην εικόνα 2 που ακολουθεί:



Εικόνα 3.2

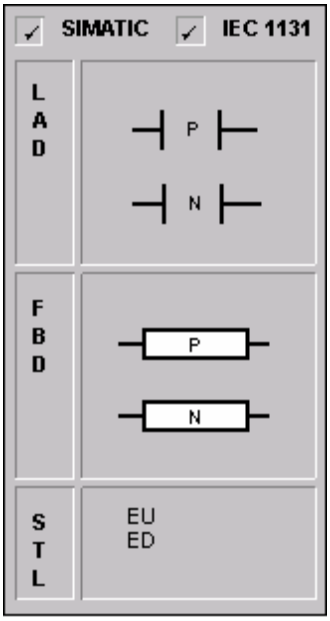
Η χρήση της NOT συνίσταται στο ότι είσοδο πάρει έχει την ιδιότητα να το αντιστρέφει.

Στην LAD σύνταξη της όπως φαίνεται στην εικόνα 1 δίνει αντίστροφη έξοδο στην Q0.1 από ότι πάει στην Q0.0. Το ίδιο συμβαίνει και στην FBD σύνταξη (σχήμα 2) αφού ουσιαστικά ο κύκλος πριν από την Q0.1 αντιστρέφει το σήμα που θα πήγαινε στην Q0.0.

Στην STL σύνταξη αντιστρέφεται ουσιαστικά το σήμα της πράξης πριν από το = με αποτέλεσμα και σε αυτήν την περίπτωση να έχουμε το ίδιο αποτέλεσμα όπως δηλαδή και στα σχήματα 3.1 και 3.2.

3.1.3 Εντολές Θετικής - Αρνητικής μετάβασης

Η σύνταξη των εντολών Positive – Negative transition στις γλώσσες (LAD/STL/FBD) καθώς και η διευθυνσιοδότηση φαίνονται στην εικόνα 3.3.



Εικόνα 3.3 Εντολές Positive – Negative transition

Inputs/Outputs	Data Type	Operands
Bit	BOOL	I, Q, V, M, SM, S, T, C, L, Power Flow
Bit (immediate)	BOOL	I

Εικόνα 3.4 Διευθυνσηοδότηση Positive – Negative transition

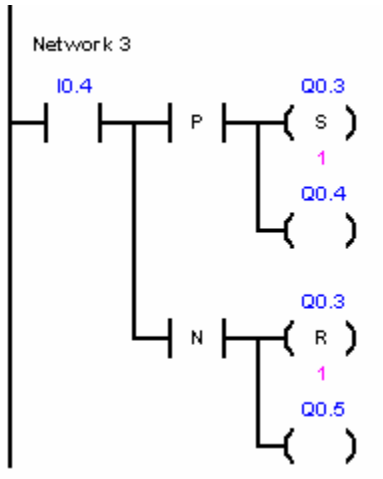
Η Positive transition αφήνει να περάσει σήμα για 1 scan του προγράμματος κατά την μετάβαση από την κατάσταση off σε on.

Η Negative transition αφήνει να περάσει σήμα για 1 scan του προγράμματος κατά την μετάβαση από την κατάσταση on σε off.

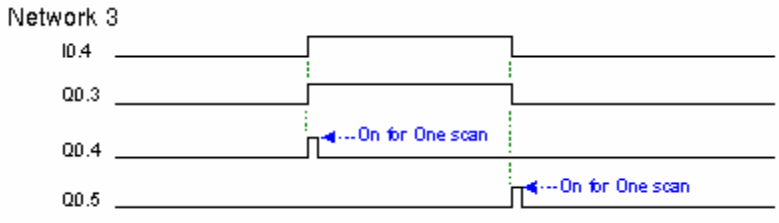
Στην LAD σύνταξη τους οι Positive – Negative transition εκφράζονται με συνδέσεις όπως φαίνεται στην εικόνα 3.3.

Στην FBD σύνταξη τους οι Positive – Negative transition εκφράζονται με κουτιά P και N όπως φαίνεται στην εικόνα 3.3.

Στην STL σύνταξη της η Positive transition εκφράζεται με την εντολή Edge Up (EU). Κατά την μετάβαση από 0 σε 1 η τιμή στην κορυφή της σύνταξης παίρνει 1 αλλιώς παίρνει το 0. Στην STL σύνταξη της η Negative transition εκφράζεται από την Edge Down (ED) εντολή. Κατά την μετάβαση από 1 σε 0 η τιμή στην κορυφή της σύνταξης παίρνει 1 αλλιώς παίρνει το 0. Στο σχήμα 3.3 που ακολουθεί περιγράφεται ένα παράδειγμα χρησιμοποίησης των Positive – Negative transition. Για την καλύτερη κατανόηση του παραδείγματος ακολουθεί και το διάγραμμα 1 που παρουσιάζει τις κυματομορφές. Σε αυτήν την περίπτωση όταν η I0.4 μεταβεί από 0 σε 1 η Q0.4 γίνεται 1 για 1 scan time ενώ παράλληλα η Q0.3 γίνεται set δηλ 1 μέχρις ότου έρθει το reset και γίνει 0. Όταν τώρα η I0.4 μεταβεί από 1 σε 0 η Q0.5 γίνεται 1 για 1 scan time ενώ παράλληλα η Q0.3 παίρνει reset και μεταβαίνει στο 0.

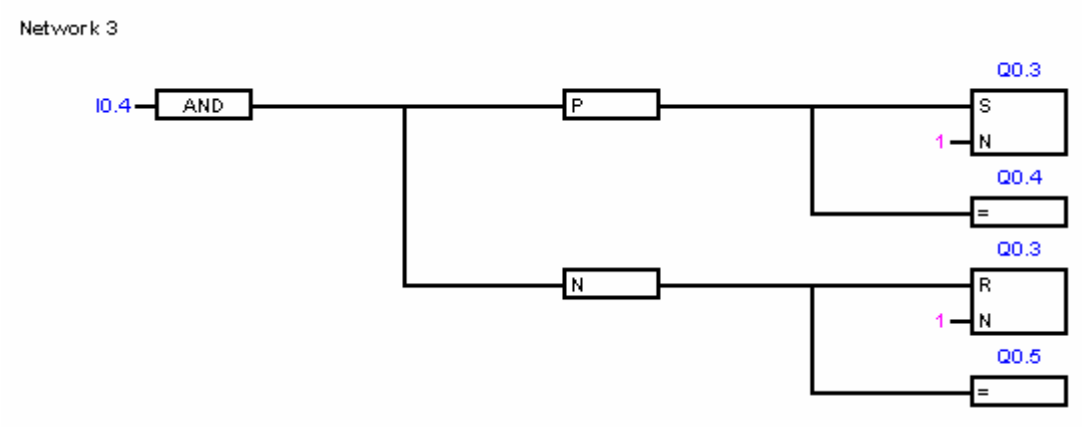


Σχήμα 3.3 Παράδειγμα LADDER εντολών Positive – Negative transition



Διάγραμμα 1 Κυματομορφές σχήματος 3.3

Οι FBD σύνταξη φαίνεται στο σχήμα 3.4 που ακολουθεί (στο οποίο η επεξήγηση της λειτουργίας είναι παρόμοια με αυτή του LAD σχήματος):

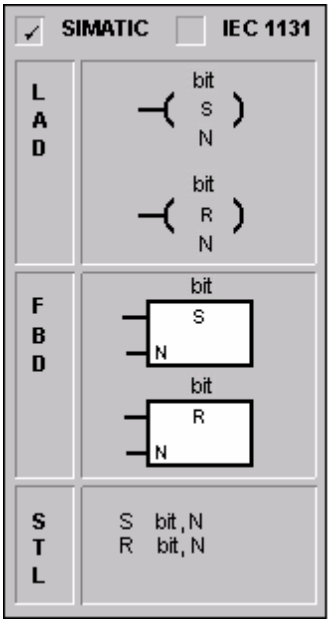


Σχήμα 3.4 Παράδειγμα FBD εντολών Positive – Negative transition

Η STL σύνταξη :
LD I0.4
LPS
EU
S Q0.3, 1
= Q0.4
LPP
ED
R Q0.3, 1
= Q0.5

3.1.4 Οι εντολές Set – Reset

Η σύνταξη των εντολών Set - Reset στις γλώσσες (LAD/STL/FBD) καθώς και η διευθυνσιοδότηση φαίνονται στις εικόνες 3.5 και 3.6.



Εικόνα 3.5 Εντολές Set – Reset

Inputs/Outputs	Data Type	Operands
Bit	BOOL	I, Q, V, M, SM, S, T, C, L
Bit (immediate)	BOOL	Q
N	BYTE	IB, QB, VB, MB, SMB, SB, LB, AC, *VD, *LD, *AC, Constant

Εικόνα 3.6 Διευθυνσιοδότηση εντολών Set – Reset

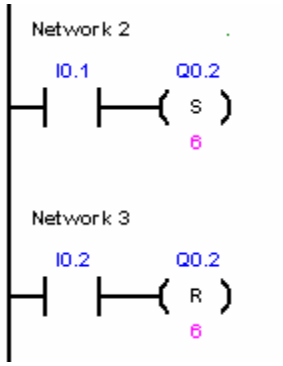
Η λειτουργία των Set - Reset στηρίζεται στο ότι όταν έρθει θετικός παλμός 1 στην είσοδο της Set τότε κάνει το bit εξόδου στο οποίο αναφέρεται ίσο με 1 μέχρις ότου να έρθει το Reset στο ίδιο bit εξόδου και να ξαναγίνει 0. Αν η εντολή Reset αναφέρεται σε bit κάποιου timer – counter η εντολή reset ξεκινά από την αρχή την μέτρηση του timer – counter.

Στην LAD σύνταξη των Set - Reset εκφράζεται με συνδέσεις όπως φαίνεται στην εικόνα 3.5.

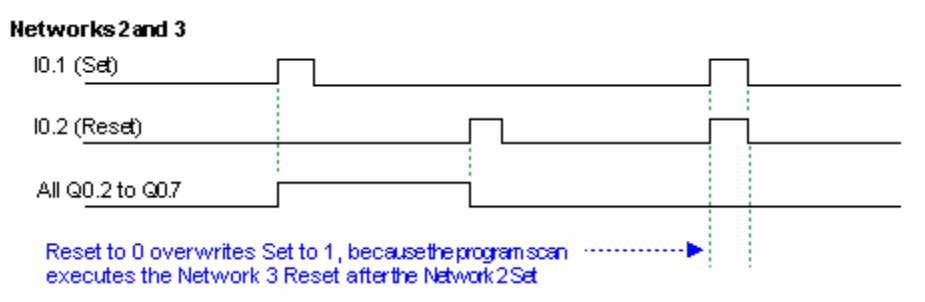
Στην FBD σύνταξη των Set - Reset εκφράζεται με κουτιά S και R όπως φαίνεται στην εικόνα 3.5.

Στην STL σύνταξη τους αυτό που χρειάζεται είναι η εντολή S όταν πρόκειται για την Set και το bit στο οποίο αναφέρεται ενώ αν πρόκειται για την Reset η εντολή R και το bit στο οποίο αναφέρεται. Η λειτουργία

των εντολών Set και Reset εξηγήθηκε κατά την περιγραφή του παραδείγματος στο σχήμα 3.3 και των κυματομορφών του (διάγραμμα 1). Ένα άλλο παράδειγμα που φαίνεται πιο ξεκάθαρα η λειτουργία των εντολών Set – Reset παρατίθεται στο σχήμα 3.5 και διάγραμμα 2.

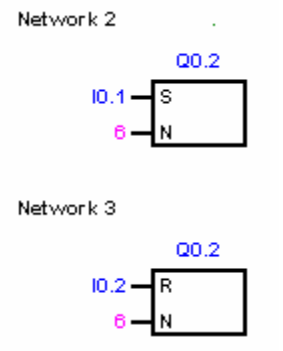


Σχήμα 3.5 Παράδειγμα LADDER εντολών Set – Reset



Διάγραμμα 2 : Κυματομορφές σχήματος 3.5

Η FBD φαίνεται στο σχήμα 3.6 που ακολουθεί:



Σχήμα 3.6 Παράδειγμα FBD εντολών Set – Reset

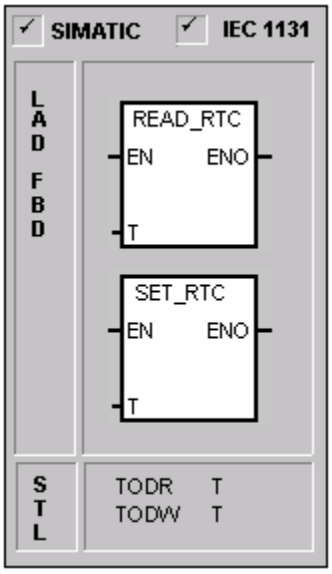
Η STL σύνταξη

```
LD I0.1
S Q0.2, 6
LD I0.2
R Q0.2, 6
```

Εδώ θα πρέπει ν' αναφέρουμε ότι το 6 αφορά τον αριθμό των bit που θα δεχθούν το Set ή το Reset με σημείο αναφοράς το Q0.2.

3.1.5 Ρολόι Πραγματικού Χρόνου(Read, Set, Real-Time Clock)

Η σύνταξη ενός Real-Time Clock στις γλώσσες (LAD/STL/FBD) καθώς και η διευθυνσιοδότηση φαίνονται στις εικόνες 3.7 και 3.8.



Εικόνα 3.7 Εντολές Read, Set, Real-Time Clock

Inputs/Outputs	Data Types	Operands
T	BYTE	IB, QB, VB, MB, SMB, SB, LB, *VD, *LD, *AC

T	T+1	T+2	T+3	T+4	T+5	T+6	T+7
Year: 00 to 99	Month: 01 to 12	Day: 01 to 31	Hours: 00 to 23	Minutes: 00 to 59	Seconds: 00 to 59	0	Day of Week: 0 to 7*

*T+7 1=Sunday, 7=Saturday
0 disables the day of week.

Εικόνα 3.8 Διευθυνσιοδότηση εντολών Read, Set, Real-Time Clock

Η Read Real-Time clock εντολή διαβάζει την τρέχουσα ώρα και

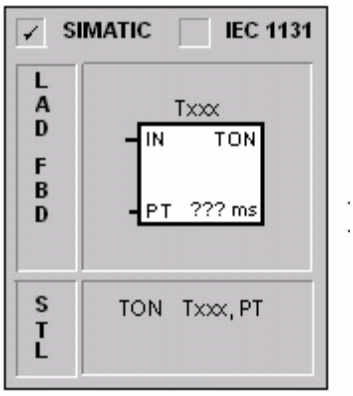
ημερομηνία από το ρολόι και την φορτώνει σε μια 8-byte προσωρινή μνήμη (buffer) ξεκινώντας από την διεύθυνση T.

Η Set Real-Time Clock γράφει την τρέχουσα ώρα και ημερομηνία στην αρχή του ρολογιού σε μια 8-byte προσωρινή μνήμη και σε μια διεύθυνση που ορίζεται από την T.

Σε STL, οι TODR και TODW εντολές αντιπροσωπεύονται ως Ανάγνωση Χρόνου Ημέρας [Time of Day Read (TODR)] και Εγγραφή Χρόνου Ημέρας [Time of Day Write (TODW)], αντίστοιχα.

3.1.6 Χρονικό με καθυστέρηση στην έναυση (TON)

Η σύνταξη ενός χρονικού (TON) στις γλώσσες (LAD/STL/FBD) καθώς και η διευθυνσιοδότηση φαίνονται στις εικόνες 3.9 και 3.10.



Εικόνα 3.9 Εντολή χρονικού (TON)

Inputs/Outputs	Data Types	Operands
Txx	WORD	Constant (T0 to T255)
IN	BOOL	I, Q, V, M, SM, S, T, C, L, Power Flow
PT	INT	IW, QW, VW, MW, SMW, SW, T, C, LW, AC, AIW, *VD, *LD, *AC, Constant

Εικόνα 3.10 Διευθυνσιοδότηση χρονικού

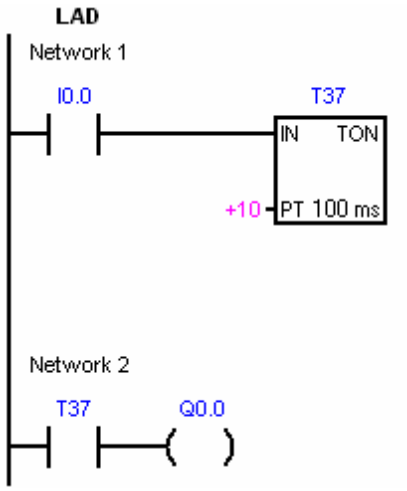
Το χρονικό με καθυστέρηση στην έναυση (TON), μετρά χρόνο όταν η είσοδος του ενεργοποιηθεί. Όταν η τρέχουσα τιμή (current value) γίνει ίση ή μεγαλύτερη από τον προκαθορισμένο χρόνο (preset time),τότε το bit εξόδου του χρονικού ενεργοποιείται. Η τρέχουσα τιμή του

χρονικού μηδενίζει όταν η επαφή εισόδου απενεργοποιηθεί. Το χρονικό σταματάει να μετράει όταν φτάνει την μέγιστη του τιμή (32.767).

Υπάρχουν 3 τύποι (TON) ανάλογα με την ακρίβεια που χρειαζόμαστε:

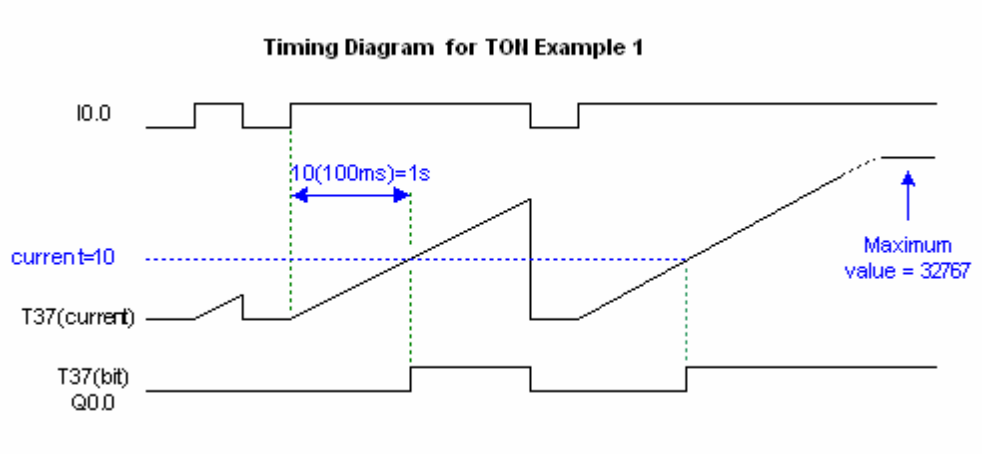
- 1ms
- 10ms
- 100ms

Για τον υπολογισμό του χρόνου αρκεί το γινόμενο της βάσης, δηλαδή την ακρίβεια του χρονικού επί την τρέχουσα τιμή (current value) π.χ.: Έστω χρονικό των 100ms με προκαθορισμένο χρόνο (preset time) 50 μετράει χρόνο 5 sec. Σημείωση: Επειδή υπάρχει μια και μοναδική ονομασία για την διεύθυνση κάθε χρονικού, δεν πρέπει να δίνουμε την ίδια ονομασία σε παραπάνω από 1 timer είτε είναι ON-Delay Timer είτε είναι OFF-Delay Timer. Παράδειγμα ON-Delay Timer φαίνεται στο σχήμα 8 που ακολουθεί:



Σχήμα 3.7 Παράδειγμα LADDER εντολής χρονικού (TON)

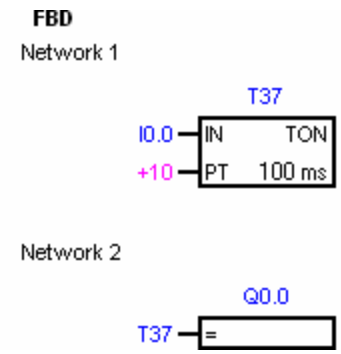
Η επεξήγηση της λειτουργίας του παραδείγματος μπορεί να γίνει ευκολότερα κατανοητή αν ανατρέξουμε στις κυματομορφές του διαγράμματος 3 που ακολουθεί.



Διάγραμμα 3 Κυματομορφές σχήματος 3.7

Όταν η επαφή I0.0 ενεργοποιηθεί και κλείσει, τότε ο Timer θα μετρήσει 1 sec και θα ενεργοποιήσει με την σειρά του την έξοδο του (T37), με αποτέλεσμα να διεγερθεί το πηνίο Q0.0.

Το πρόγραμμα σε FBD φαίνεται στο σχήμα 3.8 που ακολουθεί. Η λειτουργία του είναι ταυτόσημη με αυτή του σχήματος 3.7.

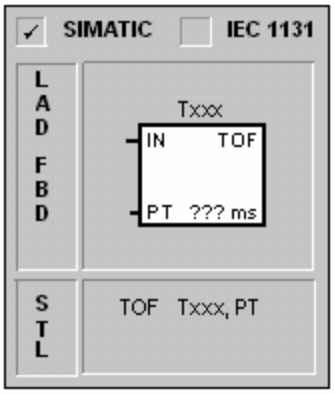


Σχήμα 3.8 Παράδειγμα FBD εντολής χρονικού (TON)

Η STL σύνταξη
NETWORK 1
LD I0.0
TON T37 +10
NETWORK 2
LD T37
= Q0.0

3.1.7 Χρονικό με καθυστέρηση στην απόζευξη (TOF)

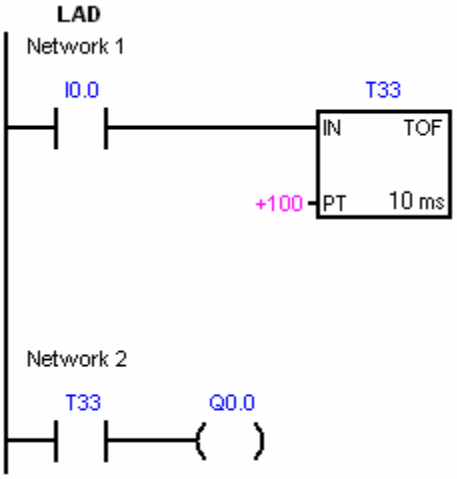
Η σύνταξη ενός χρονικού (TOF) στις γλώσσες (LAD/STL/FBD) φαίνεται στην εικόνα 3.11.



Εικόνα 3.11 Εντολή χρονικού (TOF)

Η διευθυνσιοδότηση είναι ίδια με το (TON).

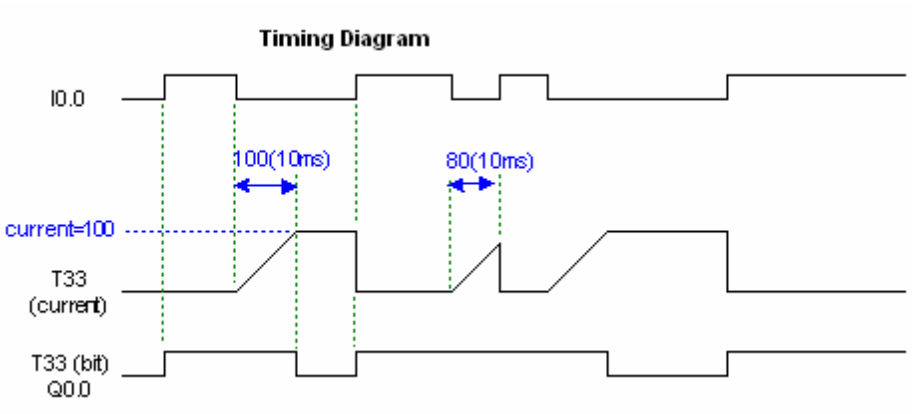
Για την καλύτερη επεξήγηση του (TOF) ακολουθεί το παρακάτω παράδειγμα (Σχήμα 3.8) :



Σχήμα 3.8 Παράδειγμα LADDER εντολής χρονικού (TOF)

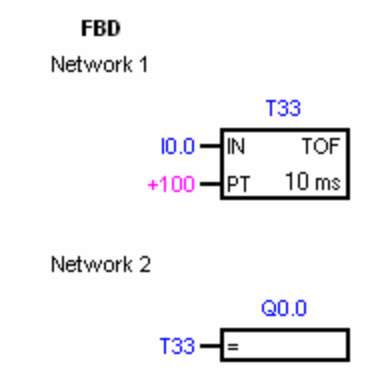
Για όση ώρα ενεργοποιηθεί και κλείσει η επαφή I0.0, το φορτίο Q0.0 διεγείρεται μέσο του T33. Όταν ανοίξει η επαφή I0.0, το φορτίο Q0.0 θα συνεχίσει να διεγείρεται μέχρι να μετρηθεί ο χρόνος που έχουμε προκαθορίσει

στον Timer. Η επεξήγηση της λειτουργίας του παραδείγματος μπορεί να γίνει ευκολότερα κατανοητή αν ανατρέξουμε στις κυματομορφές του διαγράμματος 4 που ακολουθεί.



Διάγραμμα 4 Κυματομορφές σχήματος 3.8

Το πρόγραμμα σε FBD φαίνεται στο σχήμα 3.9 που ακολουθεί. Η λειτουργία του είναι ταυτόσημη με αυτή του σχήματος 3.8.



Σχήμα 3.9 Παράδειγμα FBD εντολής χρονικού (TOF)

Η STL σύνταξη
NETWORK 1
LD I0.0
TOF T33 +10
NETWORK 2
LD T33
= Q0.0

ΚΕΦΑΛΑΙΟ 4

4.1 Στάδια προγραμματισμού

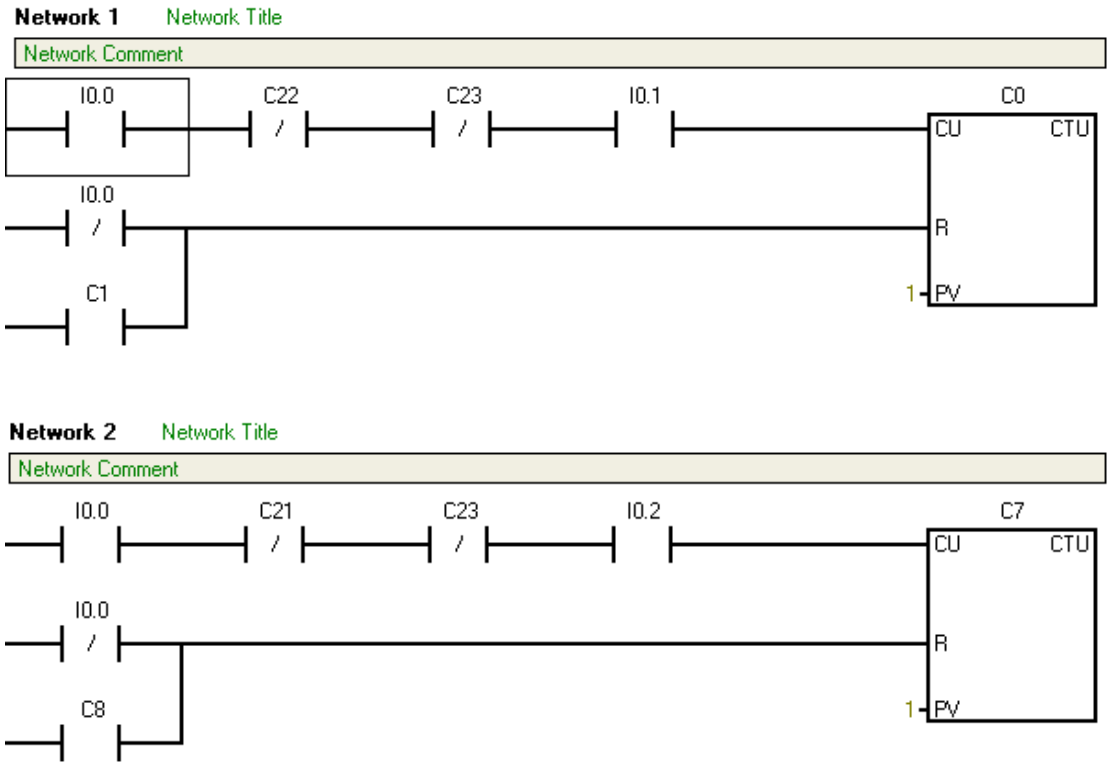
Παρακάτω βλέπουμε όλα τα βήματα σε μορφή ladder και αντίστοιχα υπάρχει η εξήγηση για τις καταστάσεις προκειμένου να έχουμε τον επιτυχή προγραμματισμό και την λειτουργία της εργασίας μας.

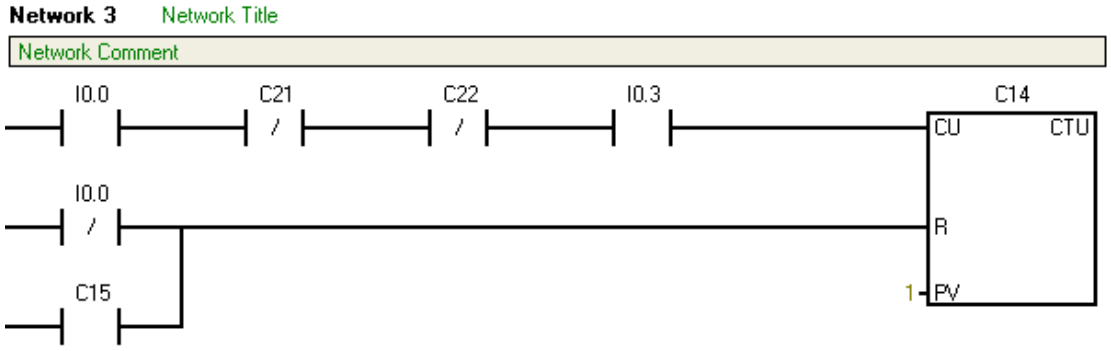
Αντίστοιχα μετά τη μορφή ladder φαίνονται και οι μορφές STL και FBD όπως αυτές προκύπτουν.

Κλείνοντας τους διακόπτες I0.0 και I0.1, ο μετρητής C0 δίνει στην έξοδο του τη λογική κατάσταση 1, και απενεργοποιείται για C1 κλειστό.

Ο μετρητής C7 ενεργοποιείται όταν κλείσουμε τους διακόπτες I0.0 και I0.2.και απενεργοποιείται όταν ο C8 έχει λογική κατάσταση 1.

Ο διακόπτης C14 δίνει λογική κατάσταση 1όταν είναι ο I0.0 και ο I0.3 είναι κλειστοί και λογική κατάσταση 0για C5 κλειστό.





Εικόνα 4.1

Για I0.0 κλειστό και τον C0 κλειστό, η μνήμη M0.0 έχει λογική κατάσταση 1.

Για I0.0 κλειστό και τον C7 κλειστό, η μνήμη M0.1 έχει λογική κατάσταση 1.

Για I0.0 κλειστό και τον C14 κλειστό, η μνήμη M0.2 έχει λογική κατάσταση 1.

Όταν ο I0.0 και η ο M0.0 είναι κλειστοί, ενεργοποιείται η Q0.0 (χαρτί).

Όταν ο I0.0 και η ο M0.0 είναι κλειστοί, ενεργοποιείται η Q0.1 (πλαστικό).

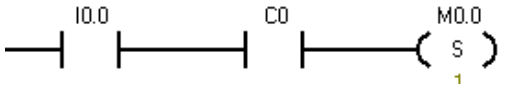
Όταν ο I0.0 και η ο M0.0 είναι κλειστοί, ενεργοποιείται η Q0.2 (αλουμίνιο).

Όταν ο I0.0 και ο C1 είναι κλειστοί, ο M0.0 κάνει reset.

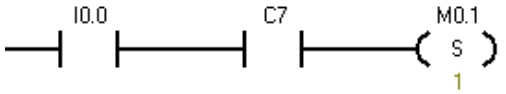
Όταν ο I0.0 και ο C8 είναι κλειστοί, ο M0.1 κάνει reset.

Όταν ο I0.0 και ο C15 είναι κλειστοί, ο M0.2 κάνει reset.

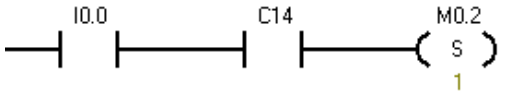
Network 4



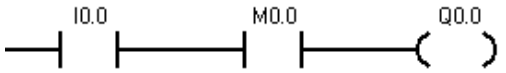
Network 5



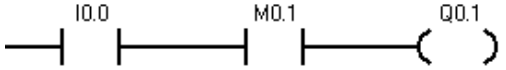
Network 6



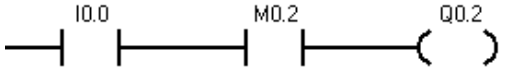
Network 7



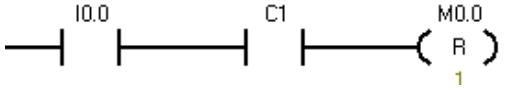
Network 8



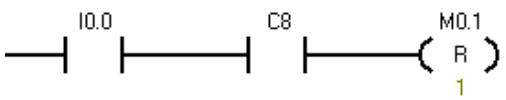
Network 9



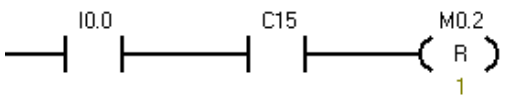
Network 10



Network 11



Network 12

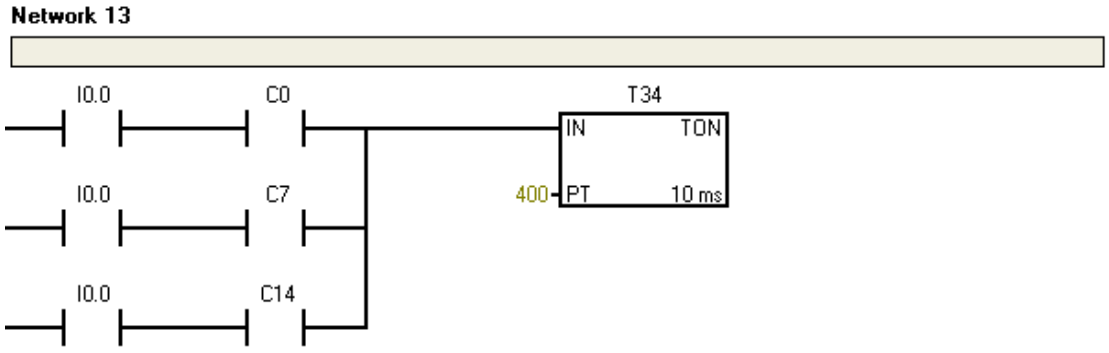


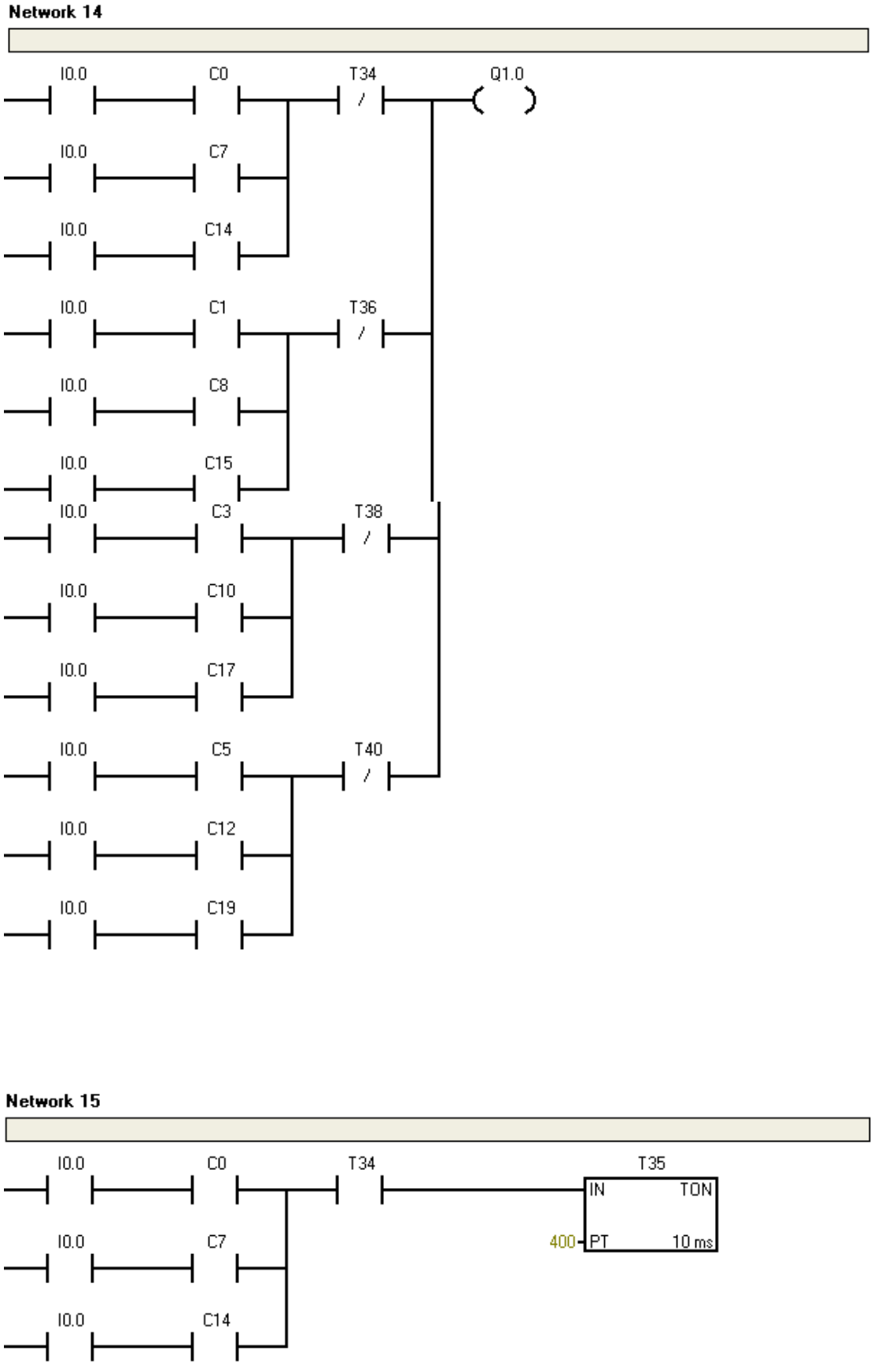
Εικόνα 4.2

Ο μετρητής T34 ενεργοποιείται όταν κλείσει ένας από τους τρεις διακόπτες C0, C7, C14.

Η έξοδος Q1.0 (ιμάντας) ενεργοποιείται ανάλογα με τον συνδιασμό των διακοπτών C0, C7, C14, C1, C8, C15, C3, C10, C17, C5, C12, C19, και έχοντας πάντα τον διακόπτη I0.0 κλειστό ακόμα και για τους χρόνους που προκαθορίζονται από τους χρονοδιακόπτες T36, T38, T40.

Ο μετρητής T35 ενεργοποιείται όταν είναι κλειστός ο I0.0 και κλειστοί ένας από τους διακόπτες C0, C7, C14 μετά από χρόνο που καθορίζεται από τον T34.



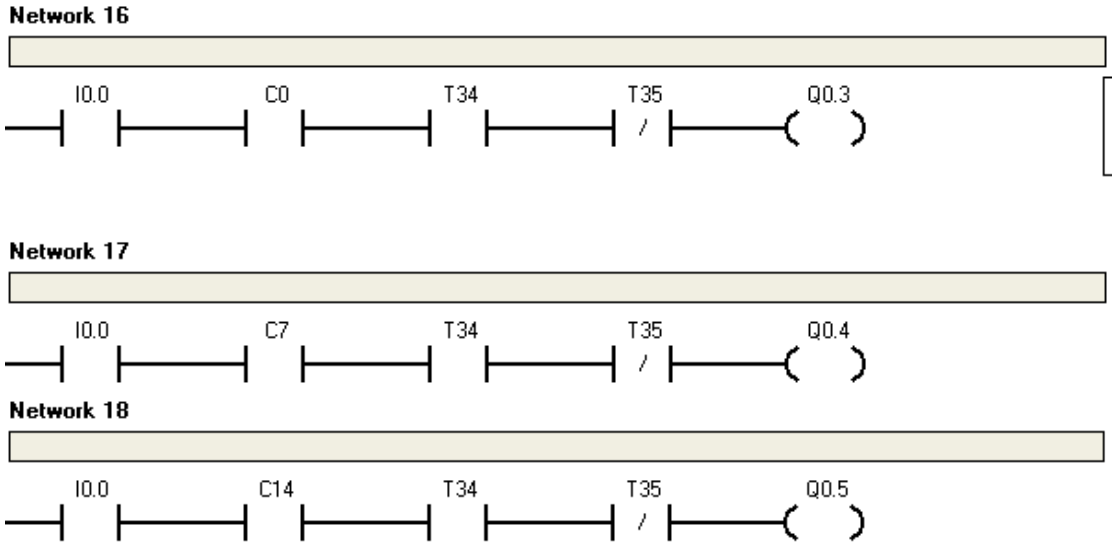

Network 14
I0.0
C0
T34
Q1.0
I0.0
C7
I0.0
C14
I0.0
C1
T36
I0.0
C8
I0.0
C15
I0.0
C3
T38
I0.0
C10
I0.0
C17
I0.0
C5
T40
I0.0
C12
I0.0
C19
Network 15
I0.0
C0
T34
T35
IN
TON
I0.0
C7
400
PT
10 ms
I0.0
C14

Εικόνα 4.3

Η έξοδος Q0.3 (θέση 1) έχει λογική κατάσταση 1 όταν κλείσουν οι διακόπτες I0.0, C0, T34, T35 κλειστό.

Η έξοδος Q0.4 (θέση 2) έχει λογική κατάσταση 1 όταν κλείσουν οι διακόπτες I0.0, C7, T34, T35 κλειστό.

Η έξοδος Q0.5 (θέση 3) έχει λογική κατάσταση 1 όταν κλείσουν οι διακόπτες I0.0, C14, T34, T35 κλειστό.

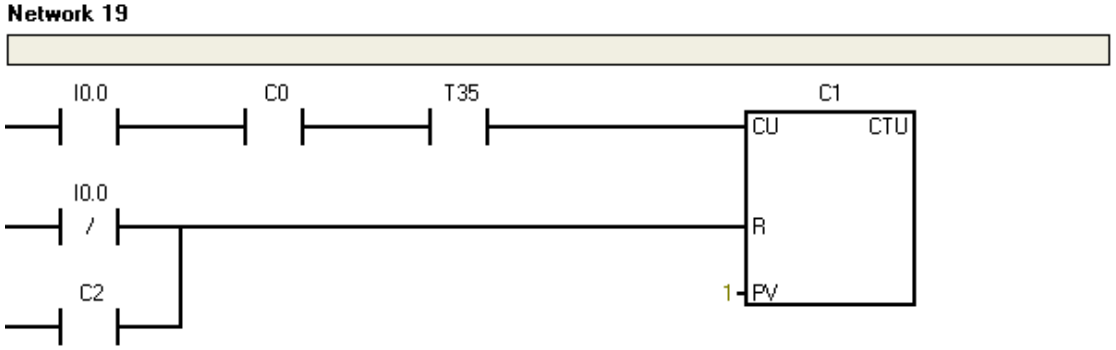


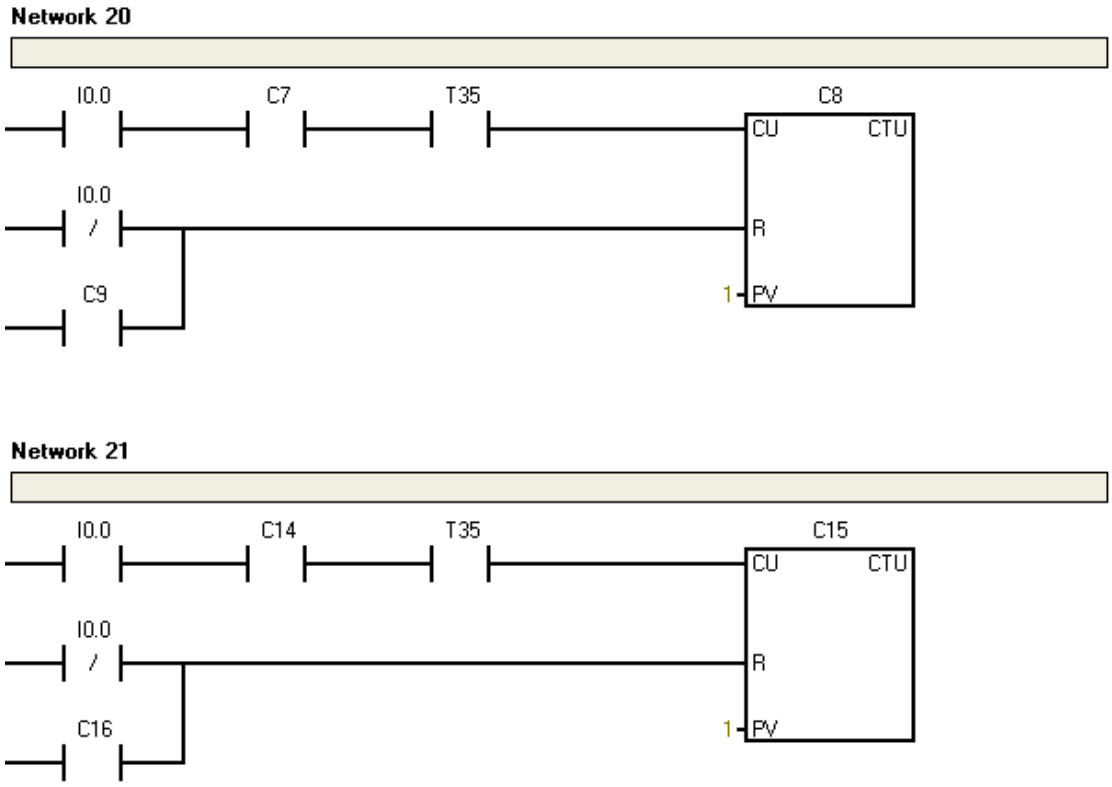
Εικόνα 4.4

Ο μετρητής C1 ενεργοποιείται όταν κλείσουν οι διακόπτες I0.0, C0 και T35 και γίνεται reset όταν ενεργοποιείται ο C2.

Ο μετρητής C8 ενεργοποιείται όταν κλείσουν οι διακόπτες I0.0, C7 και T35 και γίνεται reset όταν ενεργοποιείται ο C9.

Ο μετρητής C15 ενεργοποιείται όταν κλείσουν οι διακόπτες I0.0, C14 και T35 και γίνεται reset όταν ενεργοποιείται ο C16.





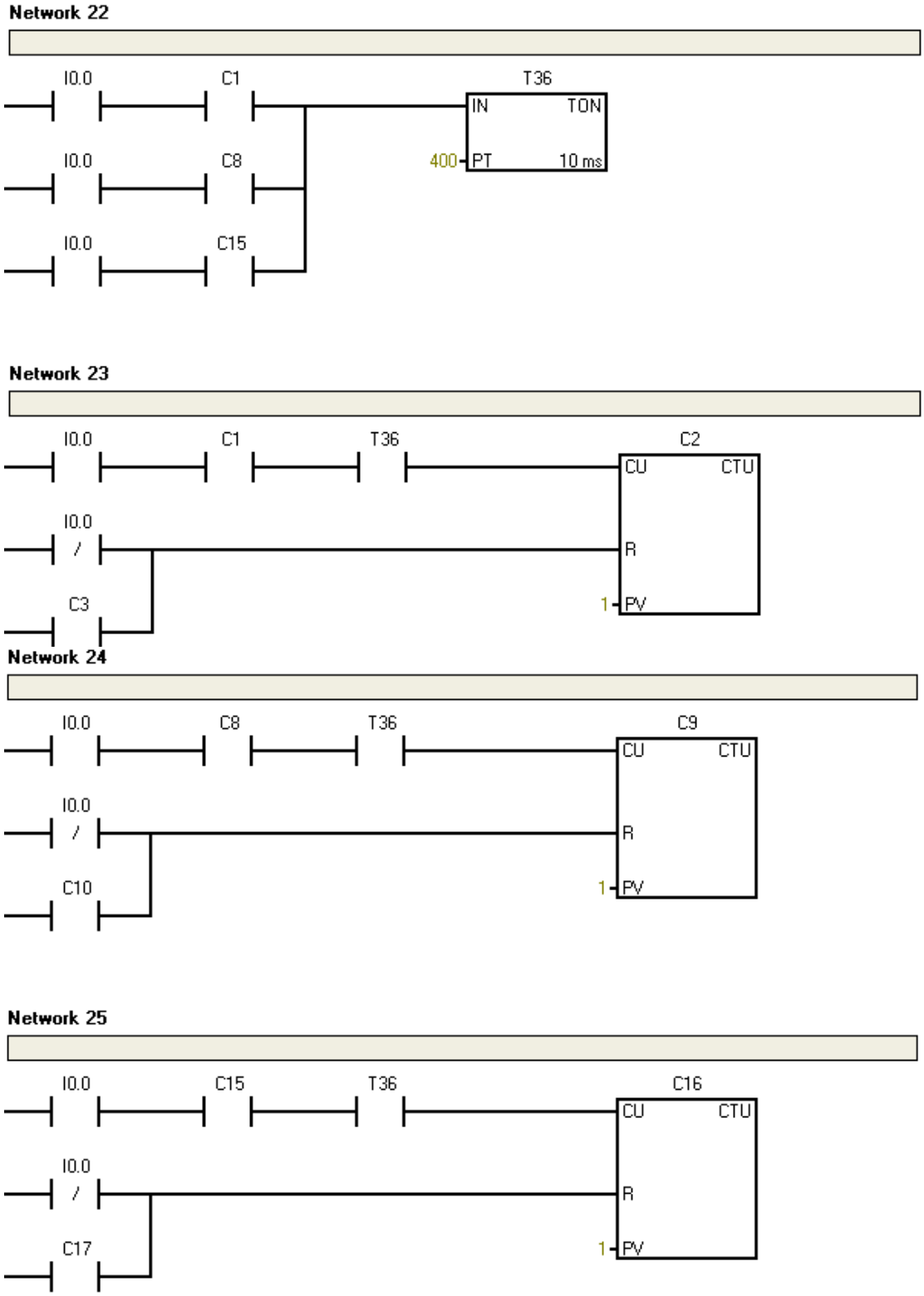
Εικόνα 4.5

Ο χρονοδιακόπτης T36 ενεργοποιείται όταν κλείσει ένας από τους διακόπτες C1, C8, C15 και I0.0 να είναι κλειστός.

Ο μετρητής C2 έχει λογική κατάσταση 1 όταν ο I0.0, C1 και ο T36 είναι κλειστοί και λογική κατάσταση 0 όταν ενεργοποιείται ο C3.

Ο μετρητής C9 έχει λογική κατάσταση 1 όταν ο I0.0, C8 και ο T36 είναι κλειστοί και λογική κατάσταση 0 όταν ενεργοποιείται ο C3.

Ο μετρητής C16 έχει λογική κατάσταση 1 όταν ο I0.0, C15 και ο T36 είναι κλειστοί και λογική κατάσταση 0 όταν ενεργοποιείται ο C17.


Network 22
I0.0
C1
T36
IN
TON
I0.0
C8
400
PT
10 ms
I0.0
C15
Network 23
I0.0
C1
T36
C2
CU
CTU
I0.0
/
R
C3
1
PV
Network 24
I0.0
C8
T36
C9
CU
CTU
I0.0
/
R
C10
1
PV
Network 25
I0.0
C15
T36
C16
CU
CTU
I0.0
/
R
C17
1
PV

Εικόνα 4.6

Ο χρονοδιακόπτης Τ37 ενεργοποιείται όταν κλείσει ένας από τους διακόπτες C2, C9, C16 και I0.0 να είναι κλειστός.

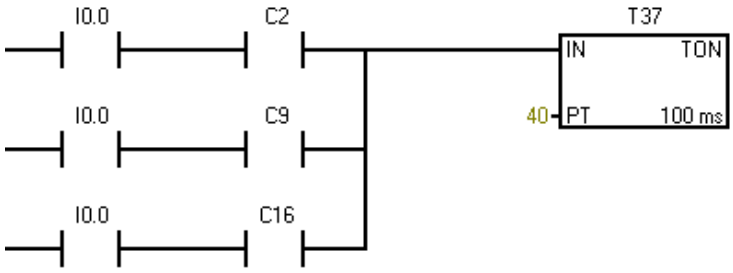
Η έξοδος Q0.6 (αέρας) ενεργοποιείται για χρόνο που καθορίζεται από το χρονοδιακόπτη Τ37. Επίσης όταν κλείσει ένας από τους διακόπτες C2, C9, C16 και ο I0.0 κλειστός.

Ο μετρητής C3 ενεργοποιείται όταν κλείσουν οι διακόπτες I0.0, C2 και Τ37 και απενεργοποιείται όταν ο C4 είναι κλειστός.

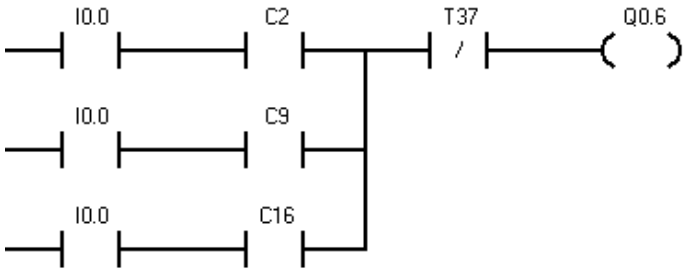
Ο μετρητής C10 ενεργοποιείται όταν κλείσουν οι διακόπτες I0.0, C9 και Τ37 και απενεργοποιείται όταν ο C11 είναι κλειστός.

Ο μετρητής C17 ενεργοποιείται όταν κλείσουν οι διακόπτες I0.0, C16 και Τ37 και απενεργοποιείται όταν ο C18 είναι κλειστός.

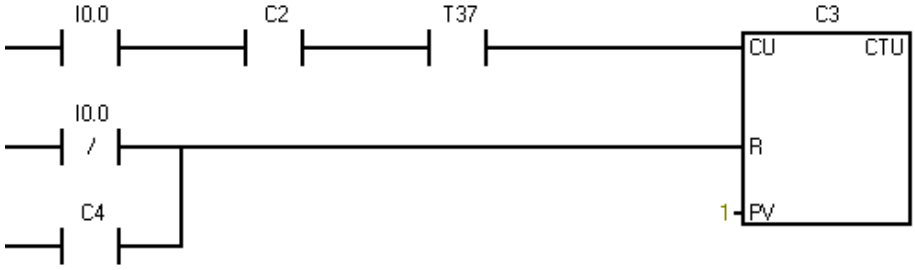
Network 26



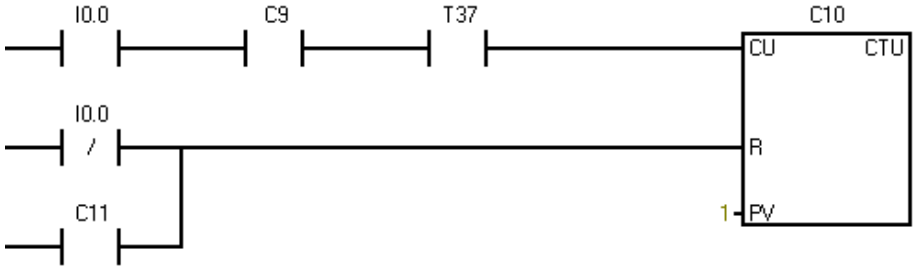
Network 27

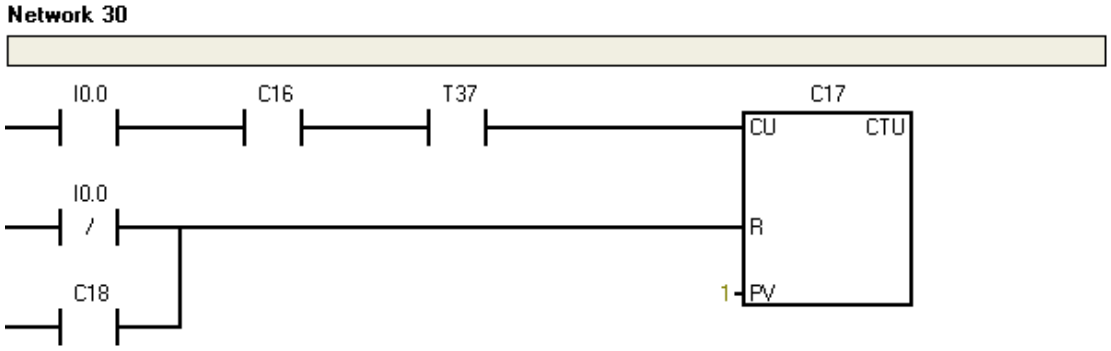


Network 28



Network 29





Εικόνα 4.7

Ο T38 ενεργοποιείται για έναν από τους διακόπτες C3, C10, C17 και ο I0.0 κλειστός.

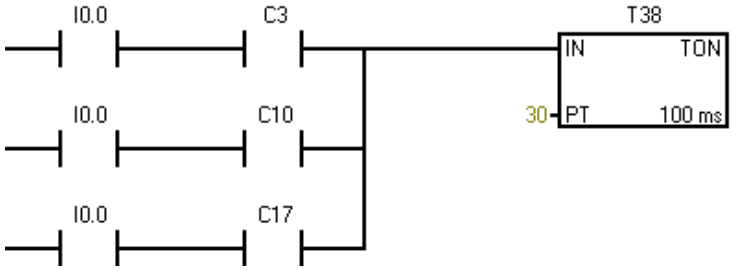
Ο C4 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C3, T38 και γίνεται reset όταν ο C5 είναι κλειστός.

Ο C11 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C10, T38 και γίνεται reset όταν ο C12 είναι κλειστός.

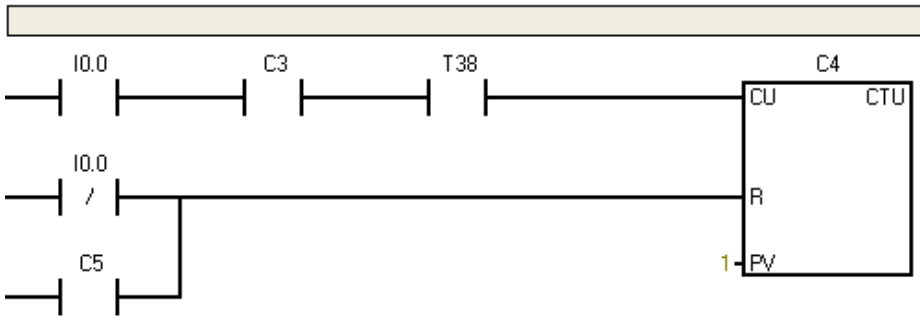
Ο C18 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C17, T38 και γίνεται reset όταν ο C19 είναι κλειστός.

Ο T39 ξεκινάει να μετράει όταν κλείσει ένας από τους διακόπτες C4, C11, C18 και ο I0.0 κλειστός.

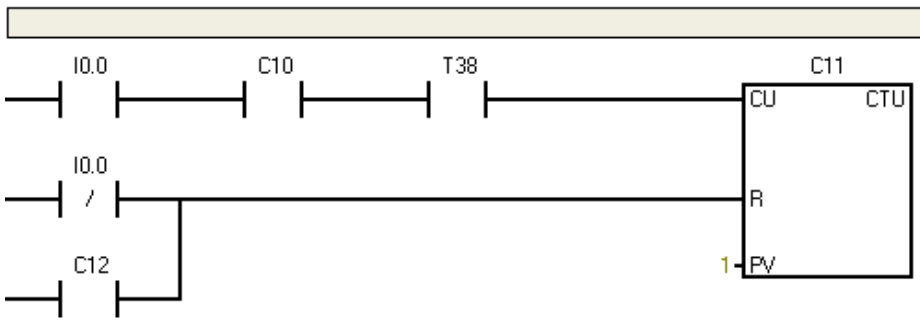
Network 31

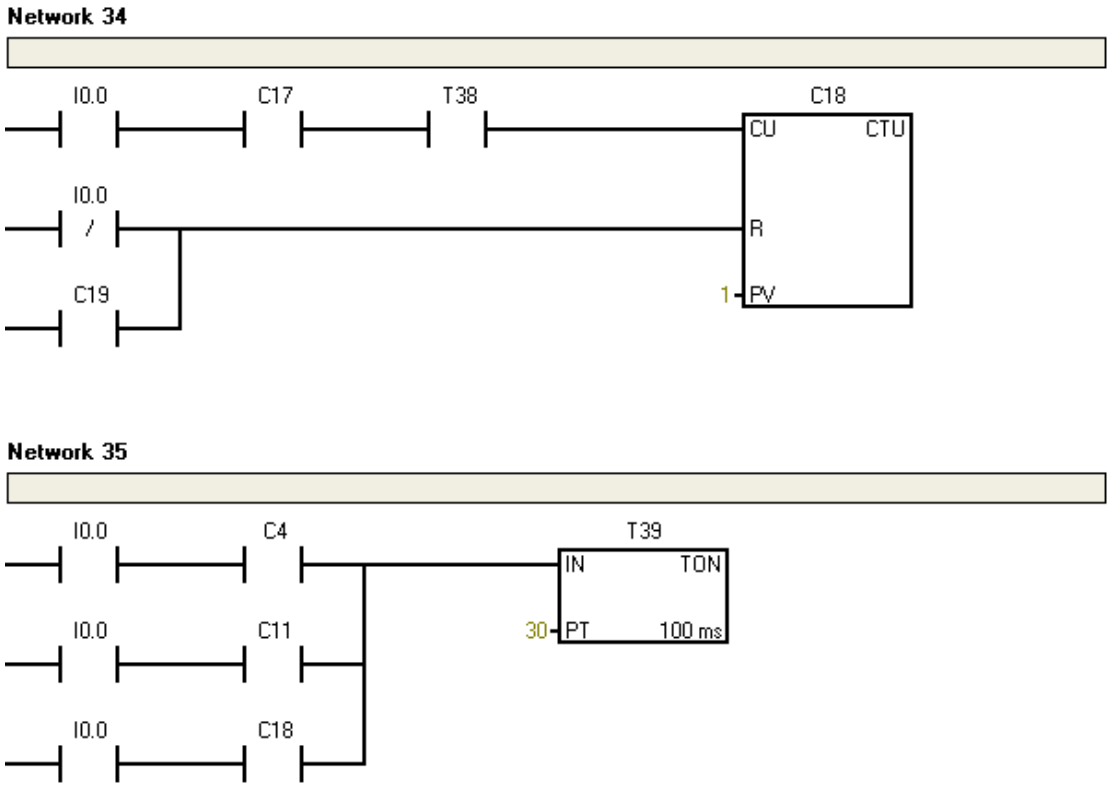


Network 32



Network 33





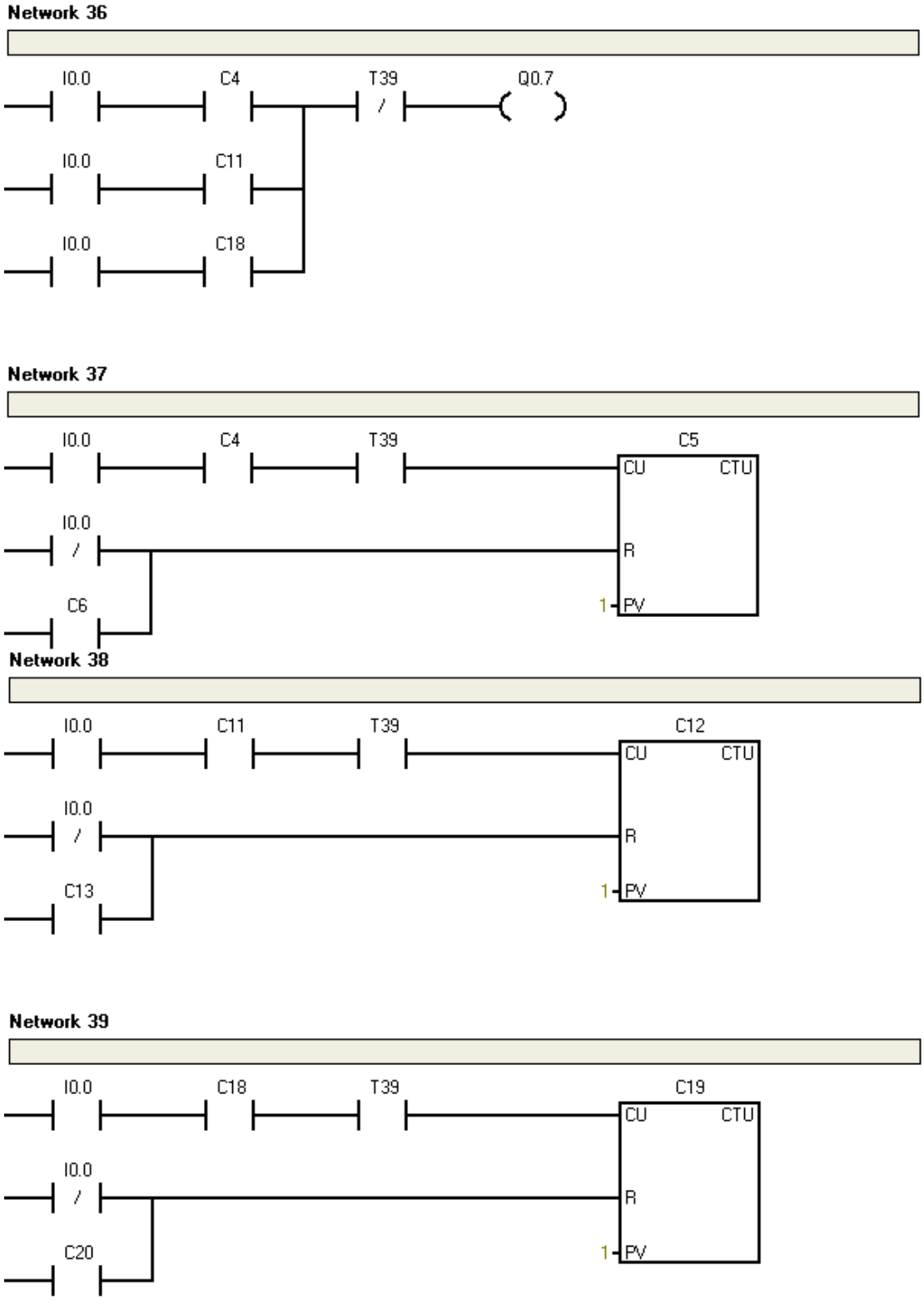
Εικόνα 4.8

Η έξοδος Q0.7 (πιεστήριο) έχει λογική κατάσταση 1για χρόνο που καθορίζεται από τον T39 όταν είναι κλειστός ο I0.0 και ένας από τους τρεις διακόπτες C4, C11, C18 κλειστούς.

Ο C5 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C4, T39 και γίνεται reset όταν ο C6 είναι κλειστός.

Ο C12 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C11, T39 και γίνεται reset όταν ο C13 είναι κλειστός.

Ο C19 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C18, T39 και γίνεται reset όταν ο C20 είναι κλειστός.


Network 36
I0.0
C4
T39
/
Q0.7
I0.0
C11
I0.0
C18
Network 37
I0.0
C4
T39
C5
CU
CTU
I0.0
/
R
C6
1
PV
Network 38
I0.0
C11
T39
C12
CU
CTU
I0.0
/
R
C13
1
PV
Network 39
I0.0
C18
T39
C19
CU
CTU
I0.0
/
R
C20
1
PV

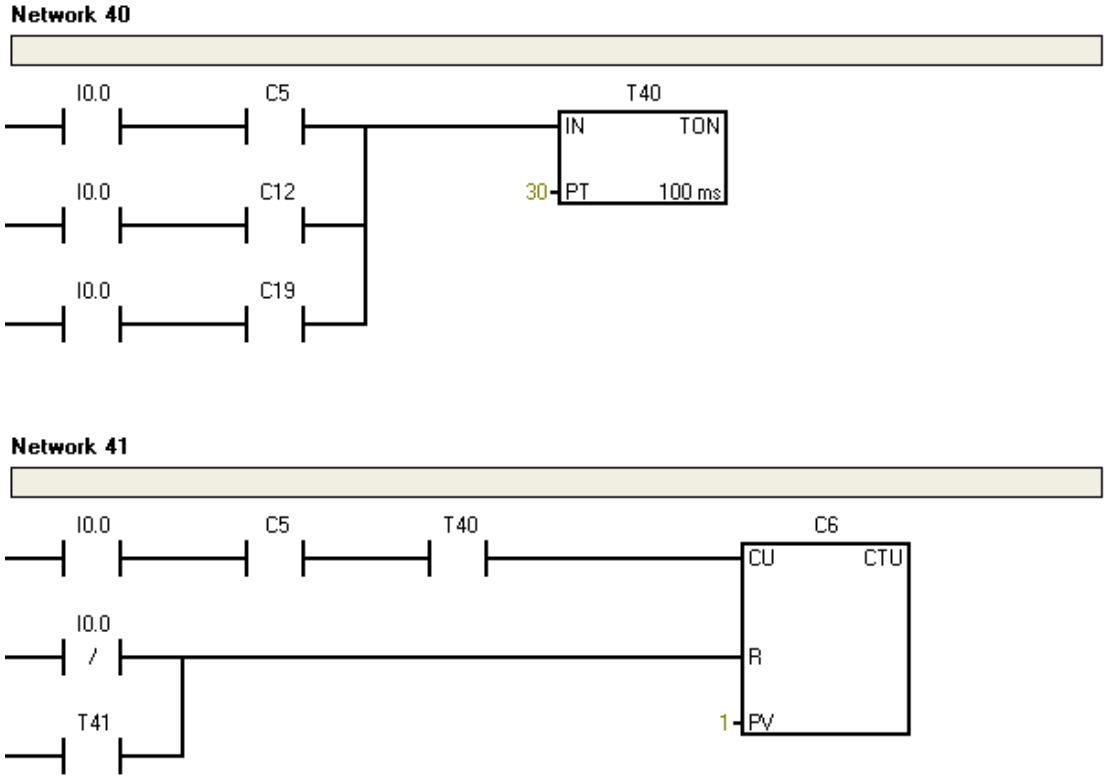
Εικόνα 4.9

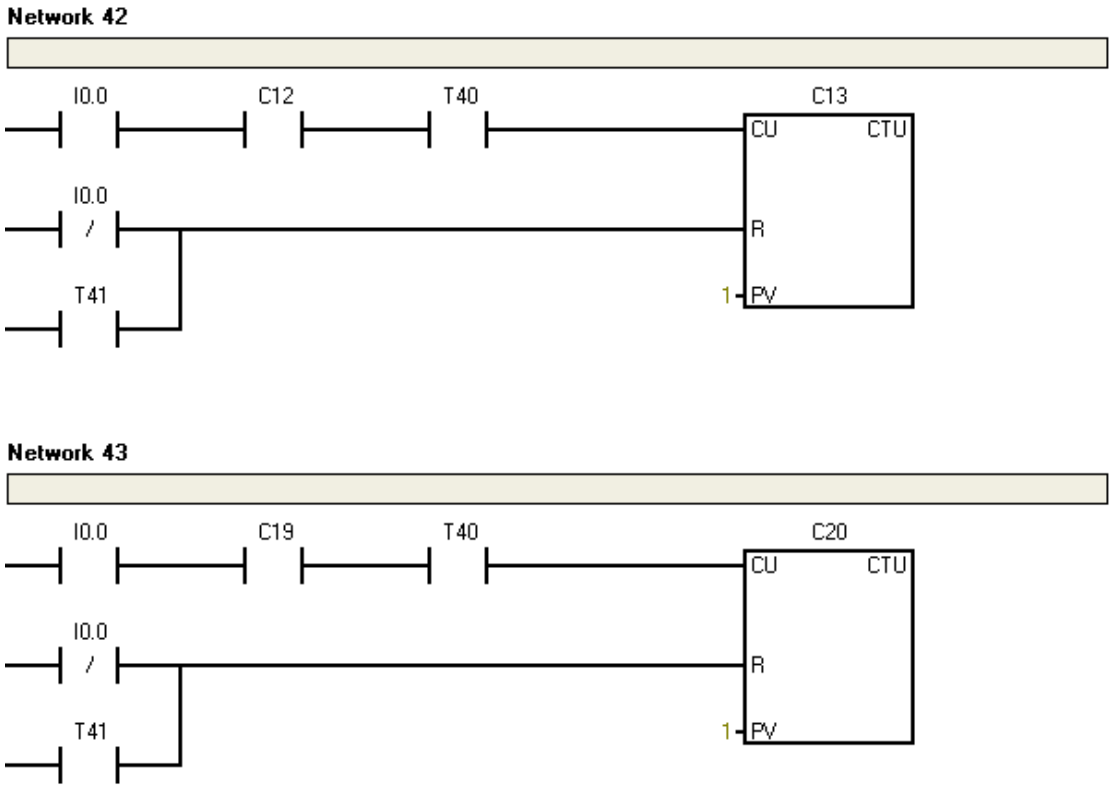
Ο T40 ξεκινάει να μετράει όταν κλείσει ένας από τους διακόπτες C5, C12, C19 και ο I0.0 κλειστός.

C6 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C5, T40 και γίνεται reset όταν ο T41 είναι κλειστός.

C13 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C12, T40 και γίνεται reset όταν ο T41 είναι κλειστός.

C20 δίνει λογική κατάσταση 1 όταν είναι κλειστοί οι I0.0, C19, T40 και γίνεται reset όταν ο T41 είναι κλειστός.





Εικόνα 4.10

Ο T41 ξεκινάει να μετράει όταν κλείσει ένας από τους διακόπτες C6, C13, C20 και ο I0.0 κλειστός.

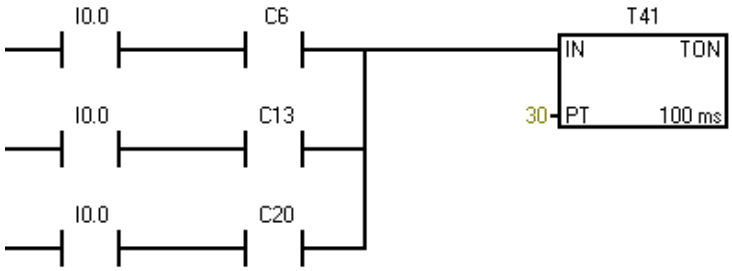
Η έξοδος Q1.1 (κάδος εναπόθεσης) έχει λογική κατάσταση 1 για χρόνο που καθορίζεται από τον T41, όταν I0.0 είναι κλειστός, και έναν από τους διακόπτες C6, C11, C20 κλειστό.

Ο μετρητής C21 δίνει στην έξοδο λογικό 1, όταν κλείσουν οι διακόπτες I0.0 και I0.1 και γίνεται reset μετά από χρόνο που καθορίζεται από τον T41.

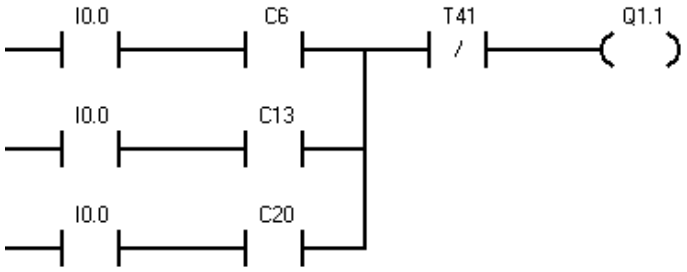
Ο μετρητής C22 δίνει στην έξοδο λογικό 1, όταν κλείσουν οι διακόπτες I0.0 και I0.2 και γίνεται reset μετά από χρόνο που καθορίζεται από τον T41.

Ο μετρητής C23 δίνει στην έξοδο λογικό 1, όταν κλείσουν οι διακόπτες I0.0 και I0.3 και γίνεται reset μετά από χρόνο που καθορίζεται από τον T41.

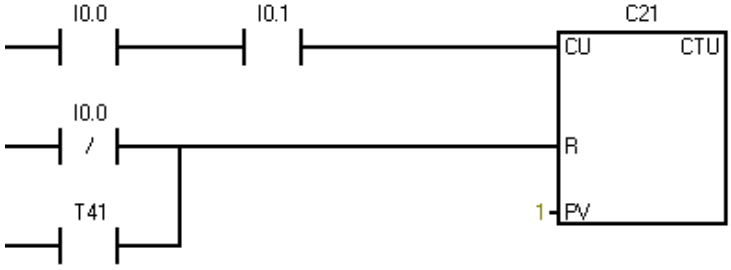
Network 44



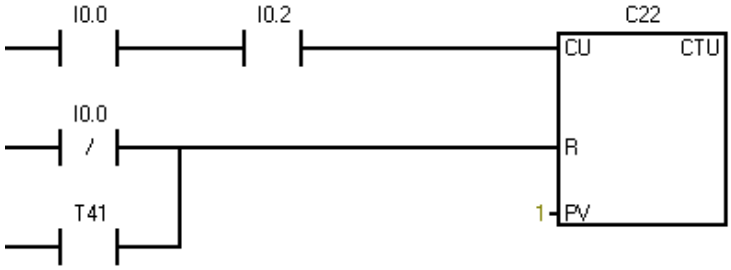
Network 45

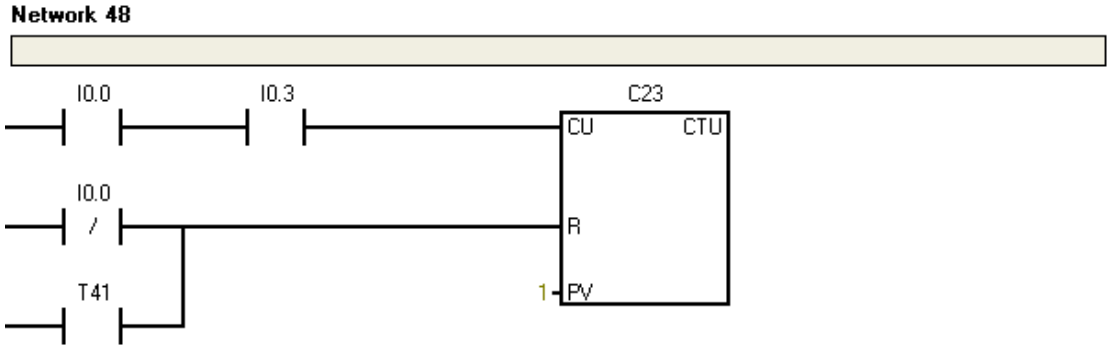


Network 46



Network 47





Εικόνα 4.11

Ακολουθεί η μορφή STL και στη συνέχεια η FBD.

```
PROGRAM COMMENTS
Network 1      Network Title
Network Comment
LD      I0.0
AN      C22
AN      C23
A       I0.1
LDN     I0.0
O       C1
CTU     C0, 1

Network 2      Network Title
Network Comment
LD      I0.0
AN      C21
AN      C23
A       I0.2
LDN     I0.0
O       C8
CTU     C7, 1
```

Network 3 Network Title

Network Comment

```
LD      I0.0
AN      C21
AN      C22
A       I0.3
LDN     I0.0
O       C15
CTU     C14, 1
```

Network 4

```
LD      I0.0
A       C0
S       M0.0, 1
```

Network 5

```
LD      I0.0
A       C7
S       M0.1, 1
```

Network 6

```
LD      I0.0
A       C14
S       M0.2, 1
```

Network 7

```
LD      I0.0
A       M0.0
=       Q0.0
```

Network 8

```
LD      I0.0
A       M0.1
=       Q0.1
```

Network 9

```
LD      I0.0
A       M0.2
=       Q0.2
```

Network 10

```
LD      I0.0
A       C1
R       M0.0, 1
```

Network 11

```
LD      I0.0
A       C8
R       M0.1, 1
```

Network 12

```
LD      I0.0
A       C15
R       M0.2, 1
```

Network 13

```
LD      I0.0
A       C0
LD      I0.0
A       C7
OLD
LD      I0.0
A       C14
OLD
TON     T34, 400
```

Network 14

```
LD      I0.0
A       C0
LD      I0.0
A       C7
OLD
LD      I0.0
A       C14
OLD
AN      T34
LD      I0.0
A       C1
LD      I0.0
A       C8
OLD
LD      I0.0
A       C15
OLD
AN      T36
OLD
LD      I0.0
A       C3
LD      I0.0
A       C10
OLD
LD      I0.0
A       C17
```

```
OLD
AN      T38
OLD
LD      I0.0
A       C5
LD      I0.0
A       C12
OLD
LD      I0.0
A       C19
OLD
AN      T40
OLD
=       Q1.0
```

Network 15

```
LD      I0.0
A       C0
LD      I0.0
A       C7
OLD
LD      I0.0
A       C14
OLD
A       T34
TON     T35, 400
```

Network 16

```
LD      I0.0
A       C0
A       T34
AN      T35
=       Q0.3
```

Network 17

```
LD      I0.0
A       C7
A       T34
AN      T35
=       Q0.4
```

Network 18

```
LD      I0.0
A       C14
A       T34
AN      T35
=       Q0.5
```

Network 19

```
LD      I0.0
A       C0
A       T35
LDN     I0.0
O       C2
CTU     C1, 1
```

Network 20

```
LD      I0.0
A       C7
A       T35
LDN     I0.0
O       C9
CTU     C8, 1
```

Network 21

```
LD      I0.0
A       C14
A       T35
LDN     I0.0
O       C16
CTU     C15, 1
```

Network 22

```
LD      I0.0
A       C1
LD      I0.0
A       C8
OLD
LD      I0.0
A       C15
OLD
TON     T36, 400
```

Network 23

```
LD      I0.0
A       C1
A       T36
LDN     I0.0
O       C3
CTU     C2, 1
```

Network 24

```
LD      I0.0
A       C8
A       T36
LDN     I0.0
O       C10
CTU     C9, 1
```

Network 25

```
LD      I0.0
A       C15
A       T36
LDN     I0.0
O       C17
CTU     C16, 1
```

Network 26

```
LD      I0.0
A       C2
LD      I0.0
A       C9
OLD
LD      I0.0
A       C16
OLD
TON     T37, 40
```

Network 27

```
LD      I0.0
A       C2
LD      I0.0
A       C9
OLD
LD      I0.0
A       C16
OLD
AN      T37
=       Q0.6
```

Network 28

```
LD      I0.0
A       C2
A       T37
LDN     I0.0
O       C4
CTU     C3, 1
```

Network 29

```
LD      I0.0
A       C9
A       T37
LDN     I0.0
O       C11
CTU     C10, 1
```

Network 30

```
LD      I0.0
A       C16
A       T37
LDN     I0.0
O       C18
CTU     C17, 1
```

Network 31

```
LD      I0.0
A       C3
LD      I0.0
A       C10
OLD
LD      I0.0
A       C17
OLD
TON     T38, 30
```

Network 32

```
LD      I0.0
A       C3
A       T38
LDN     I0.0
O       C5
CTU     C4, 1
```

Network 33

```
LD      I0.0
A       C10
A       T38
LDN     I0.0
O       C12
CTU     C11, 1
```

Network 34

```
LD      I0.0
A       C17
A       T38
LDN     I0.0
O       C19
CTU     C18, 1
```

Network 35

```
LD      I0.0
A       C4
LD      I0.0
A       C11
OLD
LD      I0.0
A       C18
OLD
TON     T39, 30
```

Network 36

```
LD      I0.0
A       C4
LD      I0.0
A       C11
OLD
LD      I0.0
A       C18
OLD
AN      T39
=       Q0.7
```

Network 37

```
LD      I0.0
A       C4
A       T39
LDN     I0.0
O       C6
CTU     C5, 1
```

Network 38

```
LD      I0.0
A       C11
A       T39
LDN     I0.0
O       C13
CTU     C12, 1
```

Network 39

```
LD      I0.0
A       C18
A       T39
LDN     I0.0
O       C20
CTU     C19, 1
```

Network 40

```
LD      I0.0
A       C5
LD      I0.0
A       C12
OLD
LD      I0.0
A       C19
OLD
TON     T40, 30
```

Network 41

```
LD      I0.0
A       C5
A       T40
LDN     I0.0
O       T41
CTU     C6, 1
```

Network 42

```
LD      I0.0
A       C12
A       T40
LDN     I0.0
O       T41
CTU     C13, 1
```

Network 43

```
LD      I0.0
A       C19
A       T40
LDN     I0.0
O       T41
CTU     C20, 1
```

Network 44

```
LD      I0.0
A       C6
LD      I0.0
A       C13
OLD
LD      I0.0
A       C20
OLD
TON     T41, 30
```

Network 45

```
LD      I0.0
A       C6
LD      I0.0
A       C13
OLD
LD      I0.0
A       C20
OLD
AN      T41
=       Q1.1
```

Network 46

```
LD      I0.0
A       I0.1
LDN     I0.0
O       T41
CTU     C21, 1
```

Network 47

```
LD      I0.0
A       I0.2
LDN     I0.0
O       T41
CTU     C22, 1
```

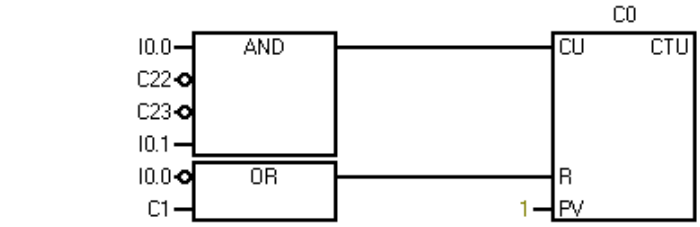
Network 48

```
LD      I0.0
A       I0.3
LDN     I0.0
O       T41
CTU     C23, 1
```

Μορφή προγραμματισμού σε FBD.

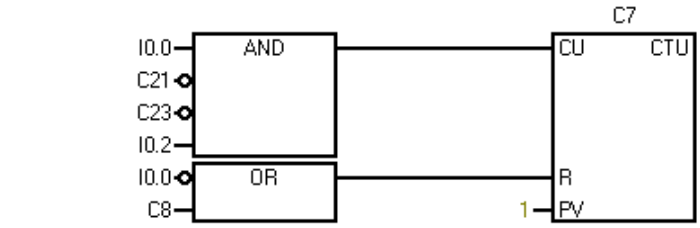
Network 1 Network Title

Network Comment



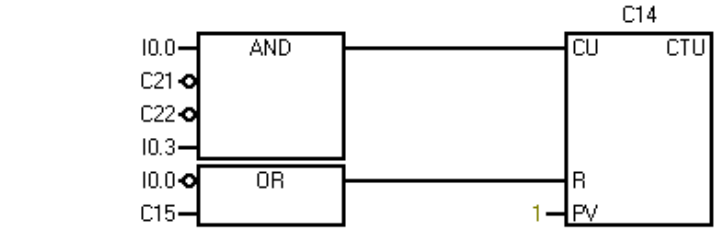
Network 2 Network Title

Network Comment

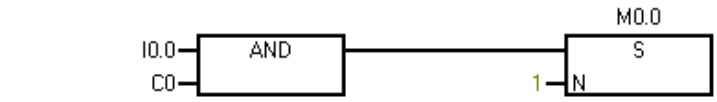


Network 3 Network Title

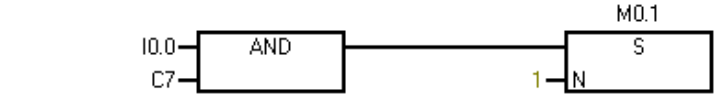
Network Comment



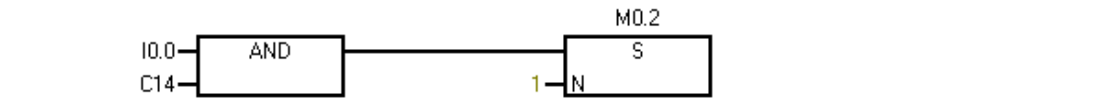
Network 4



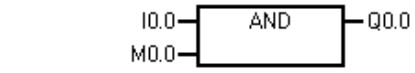
Network 5



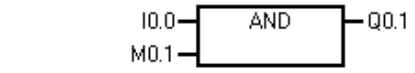
Network 6



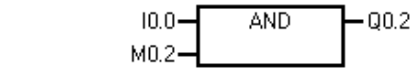
Network 7



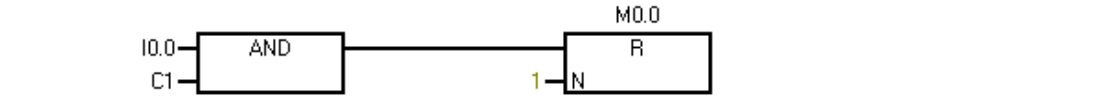
Network 8



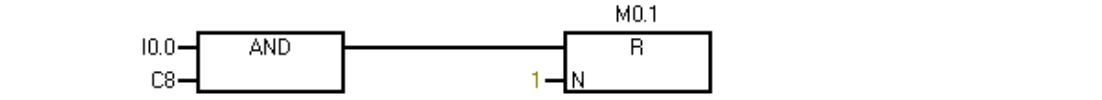
Network 9



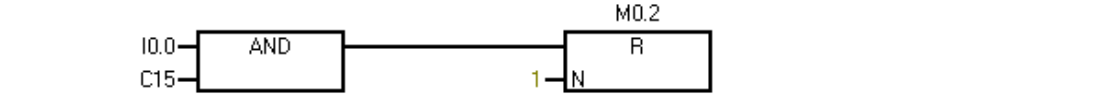
Network 10



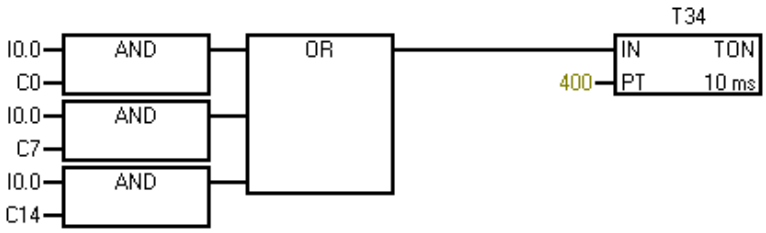
Network 11



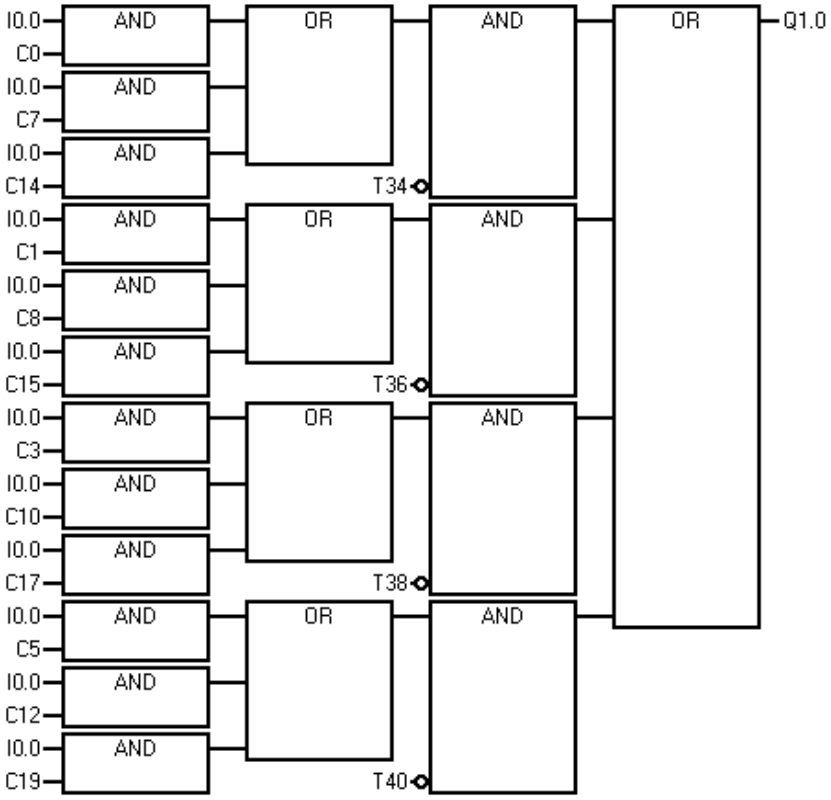
Network 12



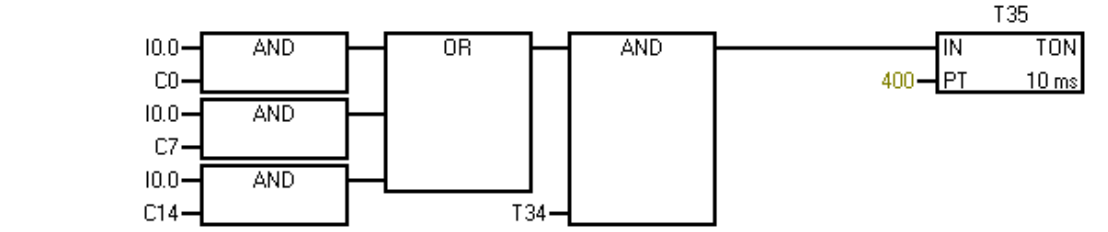
Network 13



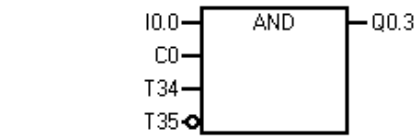
Network 14



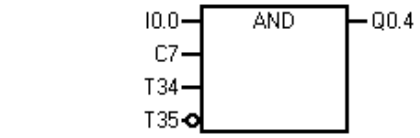
Network 15



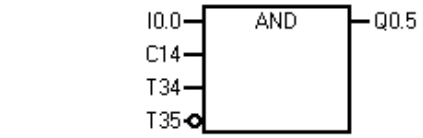
Network 16



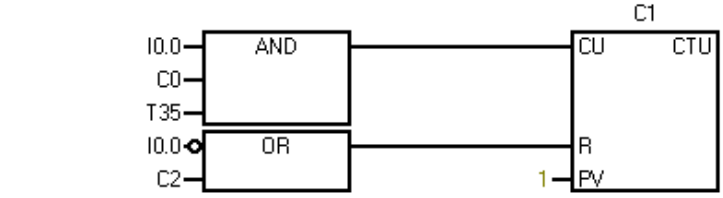
Network 17



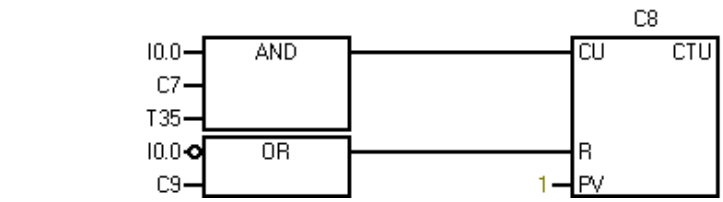
Network 18



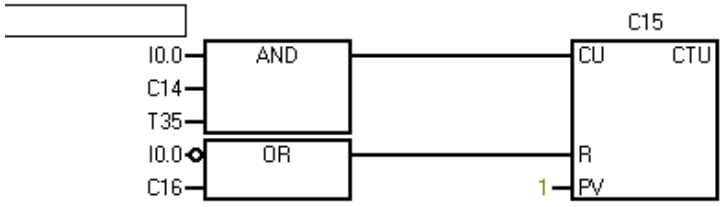
Network 19



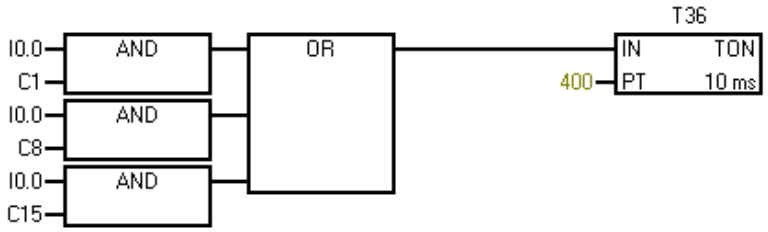
Network 20



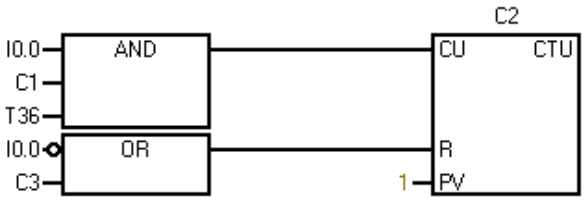
Network 21



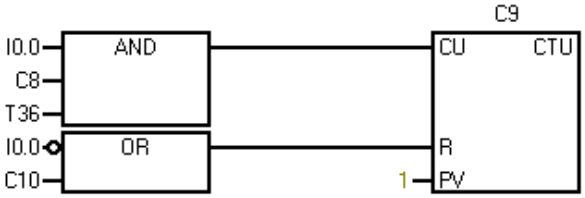
Network 22



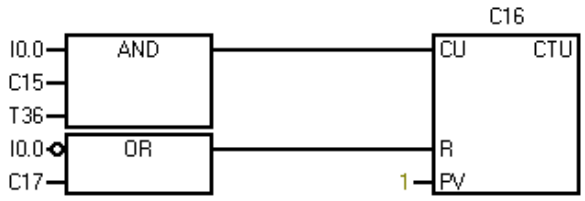
Network 23



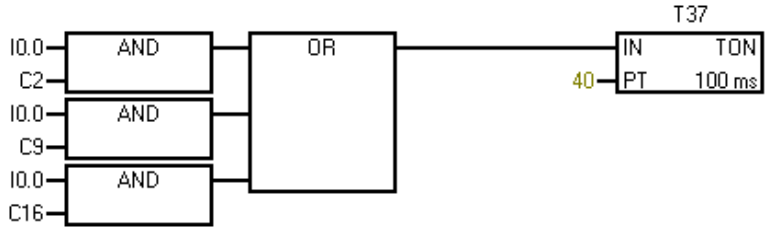
Network 24



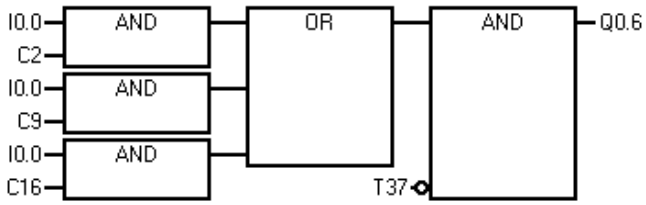
Network 25



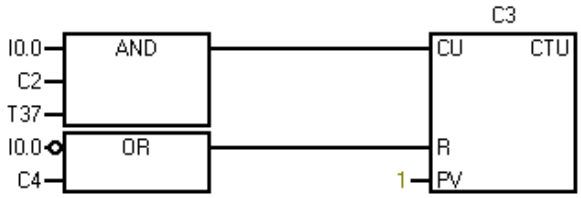
Network 26



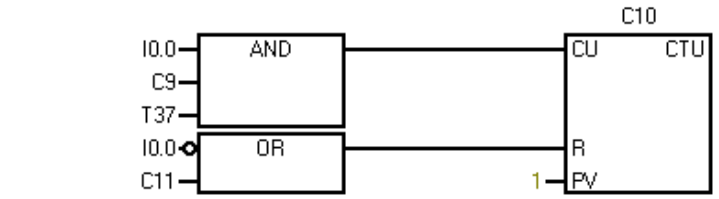
Network 27



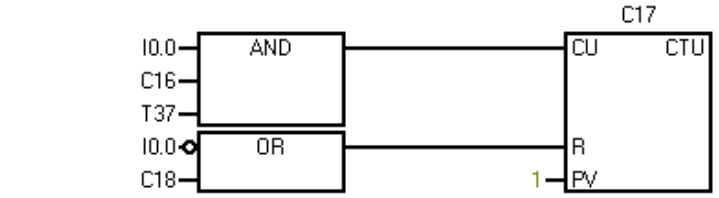
Network 28



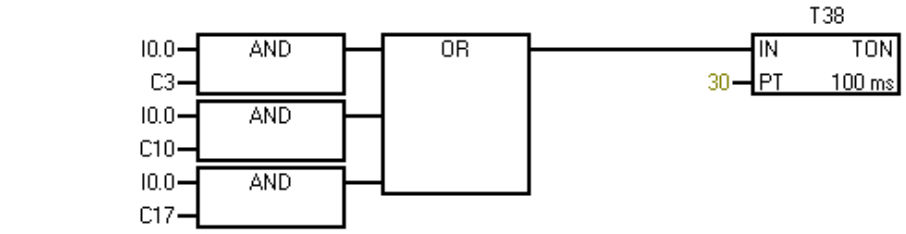
Network 29



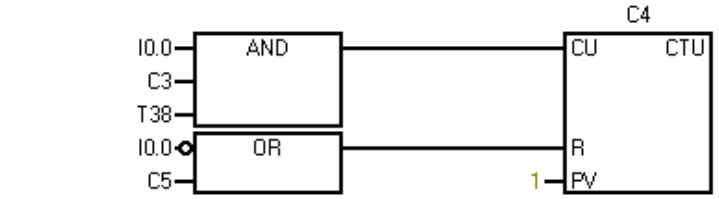
Network 30



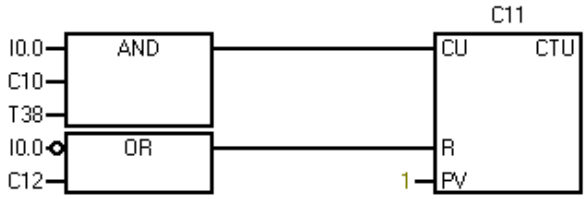
Network 31



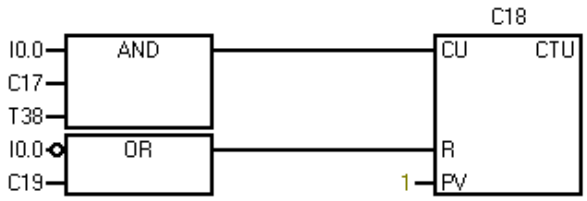
Network 32



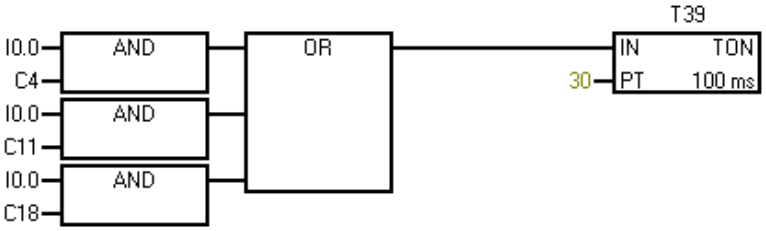
Network 33



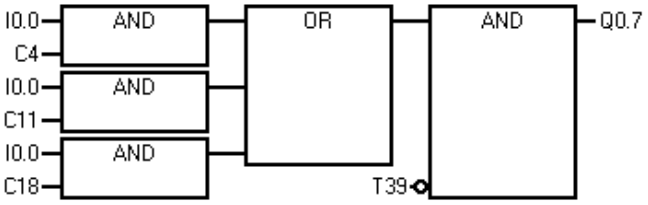
Network 34



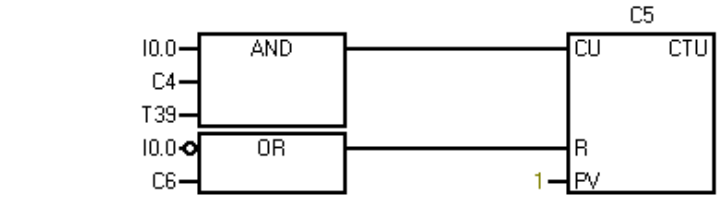
Network 35



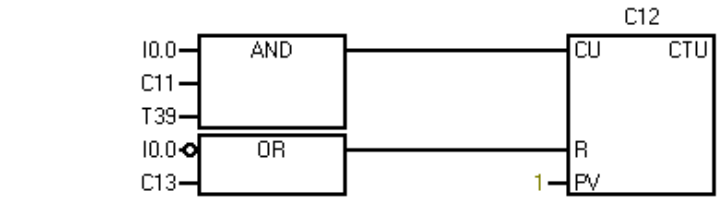
Network 36



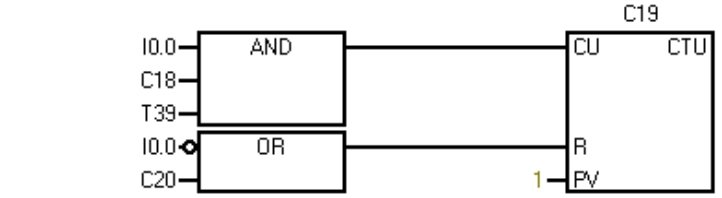
Network 37



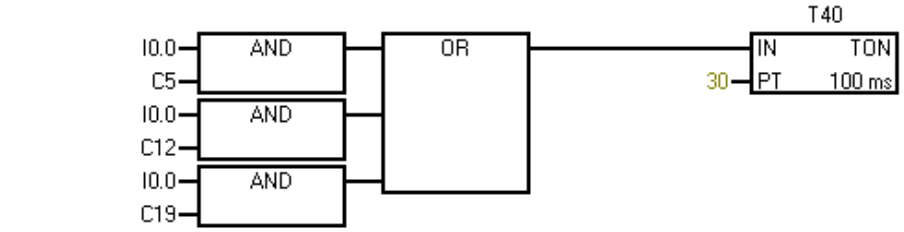
Network 38



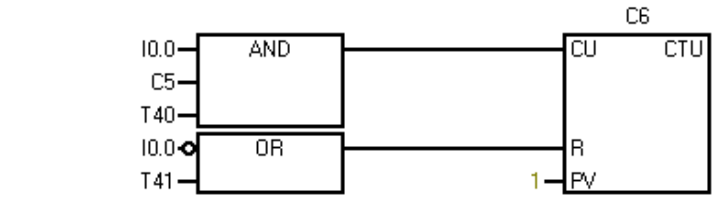
Network 39



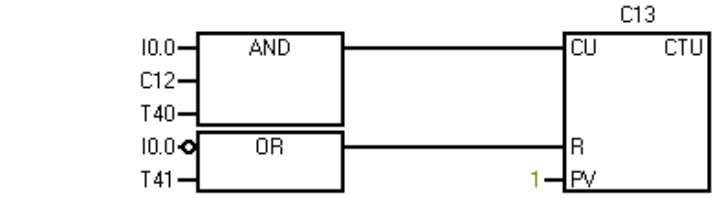
Network 40



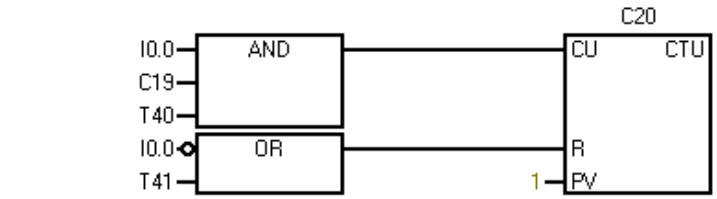
Network 41



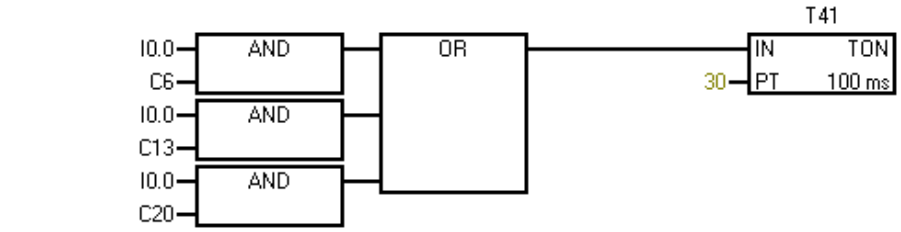
Network 42



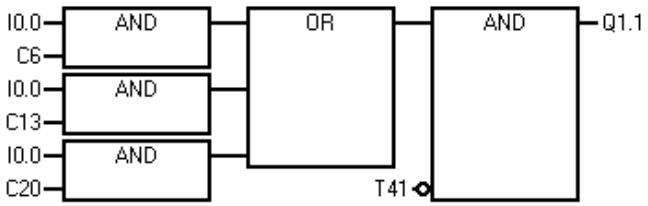
Network 43



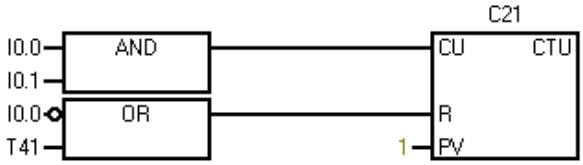
Network 44



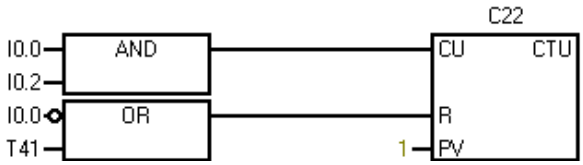
Network 45



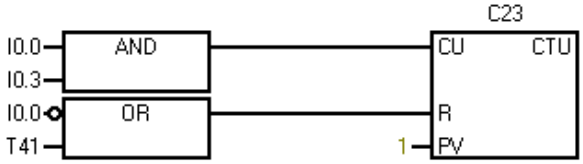
Network 46



Network 47



Network 48



ΕΠΙΛΟΓΟΣ

Με την εργασία αυτή κάνουμε αρχικά μια ιστορική αναδρομή πάνω στα PLC (προγραμματιζόμενος λογικός ελεγκτής) αναφέροντας τα βασικά μέρη του PLC. Στη συνέχεια παρουσιάζοντας τις μορφές προγραμματισμού, προσπάθησα να αναδείξω την ευκολία που έχει ο προγραμματισμός σε γλώσσα ladder, γι' αυτό και χρησιμοποίησα την παραπάνω γλώσσα προγραμματισμού προκειμένου να εκπονήσω την πτυχιακή μου εργασία.

Κάνοντας αυτή την εργασία, έμαθα κάποιες βασικές έννοιες πάνω στα PLC και τον τρόπο με τον οποίο προγραμματίζονται. Έχοντας αυτά τα βασικά εφόδια και συνεχίζοντας την προσπάθεια μου, είναι ένα σημαντικό βοήθημα για μια καλύτερη επαγγελματική αποκατάσταση, γιατί το PLC χρησιμοποιείται όλο και πιο πολύ στη βιομηχανία.

Τέλος θα ήθελα να ευχαριστήσω για τη βοήθεια και στήριξη που είχα, ώστε να μπορέσω να εκπονήσω την πτυχιακή μου εργασία, τόσο τον επιβλέποντα καθηγητή μου, κ. Χρήστο Μανάβη, όσο και τους εργαστηριακούς συνεργάτες Νικόλαο Κυριακούδη, Χάιδω Μιζέλη και Γιώργο Βεκρή του εργαστηρίου "Συστήματα Αυτομάτου Ελέγχου(Σ.Α.Ε), Ψηφιακά Συστήματα Ελέγχου και Βιομηχανικούς Αυτοματισμούς".

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. ΠΡΟΓΡΑΜΜΑΤΙΖΟΜΕΝΟΙ ΛΟΓΙΚΟΙ ΕΛΕΓΚΤΕΣ : Denis Collins – Eamonn Lane

2. Αυτοματισμοί με PLC : ΝΙΚ. Α. ΠΑΝΤΑΖΗΣ, ΕΚΔΟΣΕΙΣ Α. ΣΤΑΜΟΥΛΗ

3. PLC – Δεύτερη έκδοση : ΝΙΚ. Α. ΠΑΝΤΑΖΗΣ

Διευθύνσεις στο διαδίκτυο

4. http://www.festo-didactic.com/ov3/media/customers/1100/093311_web_leseprobe.pdf

5. http://www.control-systems-principles.co.uk/whitepapers/programmable-logic-control.pdf