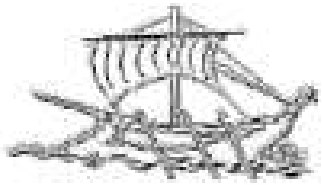


ΑΕΙ ΠΕΙΡΑΙΑ Τ.Τ.
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ Τ.Τ.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΠΡΟΣΟΜΟΙΩΤΕΣ PLC

Κωνσταντίνος Καλόγερος
Α.Μ 38360

Εισηγητής: κ. Βελώνη Αναστασία

ΑΘΗΝΑ
ΙΑΝΟΥΑΡΙΟΣ 2017

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΠΡΟΣΟΜΟΙΩΤΕΣ PLC

Κωνσταντίνος Καλόγερος
Α.Μ 38360

Εισηγητής:

Βελώνη Αναστασία

Εξεταστική Επιτροπή:

Ημερομηνία εξέτασης:

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή εργασία ασχολείται με την παρουσίαση μερικών προσομοιωτών PLC όπως HYDRAN – PSIM – LADSIM – LOGIXPRO.

ABSTRACT

The present thesis concerns with some PLC SIMULATORS such as HYDRAN – PSIM – LADSIM – LOGIXPRO.

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ : Βιομηχανικός έλεγχος – Αυτοματισμός – Ρομποτική.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ : PLC – SIMULATION

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία αποτελεί το τελευταίο μέρος της ολοκλήρωσης των σπουδών μου στο τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστικών Συστημάτων του ΑΕΙ Πειραιά Τ.Τ.

Θα ήθελα να ευχαριστήσω την επιβλέπουσα καθηγήτρια κ. Βελώνη για την πολύτιμη βοήθεια και συμπαράστασή της καθ' όλη την διάρκεια μελέτης και συγγραφής της παρούσας εργασίας, όπως και τους φίλους και την οικογένειά μου για την αμέριστη βοήθειά τους.

ΠΕΡΙΕΧΟΜΕΝΑ

1. ΕΙΣΑΓΩΓΗ –ΓΕΝΙΚΕΣ ΓΝΩΣΕΙΣ

ΚΕΦΑΛΑΙΟ 1

ΕΙΣΑΓΩΓΗ – ΓΕΝΙΚΕΣ ΓΝΩΣΕΙΣ

1.1 Ιστορική Αναδρομή

Ήδη από τη δεκαετία του ’60 στην Ευρώπη άρχισε η μετάβαση στα συστήματα με ψηφιακά ηλεκτρονικά. Αυτό δεν άλλαξε μόνο τον τρόπο σκέψης των κατασκευαστών αλλά και τη δομή και τον τρόπο λειτουργίας εγκαταστάσεων και μηχανών. Υπήρξαν όμως και αρνητικά σημεία αφού απαιτήθηκε η γνώση υψηλής ηλεκτρονικής για τη σωστότερη εγκατάσταση και συντήρησή τους. Οι πρώτοι προγραμματιζόμενοι λογικοί ελεγκτές (PLC – Programmable Logic Controllers) στην αρχή της δεκαετίας του ’70 χρησιμοποιήθηκαν κυρίως για την αντικατάσταση των ρελέ.

Η μεγάλη απαίτηση για μείωση του κύκλου παραγωγής άρχισε στην αρχή της δεκαετίας του ’80. Η τεχνολογία γινόταν γρηγορότερη και αναπτυσσόταν συνεχώς, παράλληλα με τις απαιτήσεις του χρήστη. Όπως σε όλους τους τομείς έτσι και εδώ, η επικοινωνία και η πληροφορία έγιναν η σημαντικότερη βάση για αποδοτική παραγωγή. Οι νέες συσκευές επεξεργάζονται πλέον δεδομένα και ανταλλάσουν πληροφορίες μεταξύ τους ή με υπερκείμενους υπολογιστές. Οι διαδικασίες παραγωγής γίνονται πιο σύνθετες, οι νεκροί χρόνοι στη παραγωγή μειώνονται συνεχώς, οι απαιτήσεις για αυξημένη ποιότητα αυξάνονται. Αλλάζει και ο ρόλος του ανθρώπου στην παραγωγική διαδικασία, τώρα σχεδιάζει, κατασκευάζει, προγραμματίζει, επιτηρεί και επισκευάζει.

Κι ενώ η τεχνολογία προχωρά, 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 (Windows) για να μειώσουν

τους χρόνους στον προγραμματισμό των PLC(σχόλια προγράμματος, αντιγραφή τμημάτων προγράμματος από ένα πρόγραμμα σ’ ένα άλλο κ.λ.π).

1.2 Τι είναι ο προγραμματιζόμενος λογικός ελεγκτής (PLC)

Τα PLCs είναι οι ηλεκτρονικοί υπολογιστές της βιομηχανίας. Ο όρος προγραμματιζόμενος λογικός ελεγκτής, προκύπτει από τον αγγλικό όρο programmable logic controller και ο ορισμός του είναι ο ακόλουθος:

Ο προγραμματιζόμενος λογικός ελεγκτής είναι ένα ψηφιακό σύστημα, σχεδιασμένο για χρήση σε βιομηχανικό περιβάλλον, το οποίο χρησιμοποιεί μια προγραμματιζόμενη μνήμη για την αποθήκευση εντολών, ώστε να επιτελούνται διάφορες λειτουργίες, όπως λογικές, χρονικές, μετρητικές και αριθμητικές πράξεις και να ελέγχονται μέσω αναλογικών – ψηφιακών μονάδων, διάφορες μηχανές ή διαδικασίες.

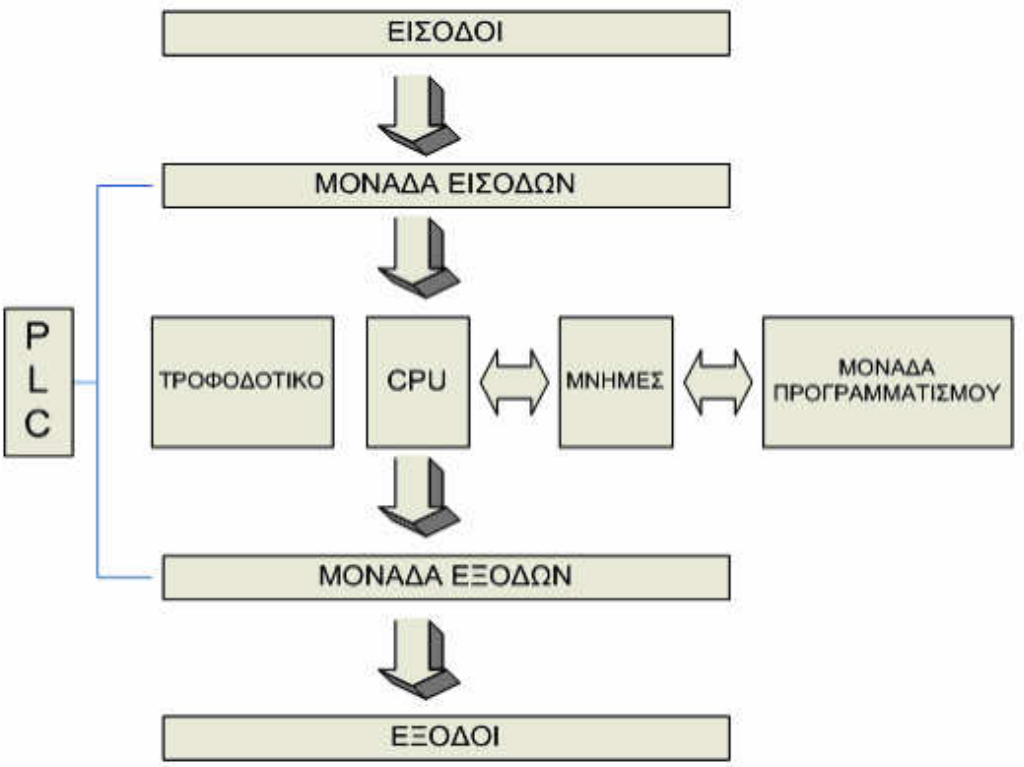
Σχήμα 1.1:Προγραμματιζόμενοι λογικοί ελεγκτές

1.3 Δομή ενός προγραμματιζόμενου λογικού ελεγκτή

Στην αγορά υπάρχουν πολλά μοντέλα PLCκατασκευασμένα από πολλές εταιρείες. Η επιλογή ενός προγραμματιζόμενου ελεγκτή (τύπος, μέγεθος, κόστος) εξαρτάται από το πλήθος των στοιχείων που δίνουν εντολή σ’ αυτόν (είσοδοι) και το πλήθος των στοιχείων που δέχονται εντολή απ’ αυτόν (έξοδοι), καθώς και από το πλήθος των λειτουργιών που

απαιτείται να κάνει ο αυτοματισμός (μέγεθος προγράμματος, δηλαδή απαιτούμενη μνήμη και δυνατότητες της κεντρικής μονάδας). Ανεξάρτητα από τον τύπο και το μέγεθος, ένας προγραμματιζόμενος ελεγκτής, συνιστάται από τα εξής απαραίτητα στοιχεία:

- Πλαίσιο τοποθέτησης των μονάδων
- Μονάδα τροφοδοσίας
- Κεντρική μονάδα επεξεργασίας (CPU) που αποτελεί τον εγκέφαλο του PLC
- Μονάδες εισόδων – εξόδων και
- Συσκευή προγραμματισμού



Σχήμα 1.2:Δομή PLC

Πλαίσιο τοποθέτησης μονάδων

Όλες οι μονάδες, από τις οποίες αποτελείται ένας προγραμματιζόμενος ελεγκτής, πρέπει να τοποθετηθούν σε κάποιο πλαίσιο. Σ' αυτό είναι ενσωματωμένο το σύστημα αγωγών (BUS), μέσω των οποίων επικοινωνούν οι διάφορες μονάδες μεταξύ τους για την ανταλλαγή πληροφοριών και για την τροφοδοσίας τους.

Αν οι θέσεις του κεντρικού πλαισίου που προσφέρεται, δεν επαρκούν για να τοποθετηθούν οι μονάδες εισόδων και εξόδων που απαιτούνται σε μια συγκεκριμένη

εφαρμογή, τότε χρησιμοποιούνται περισσότερα πλαίσια επέκτασης για την τοποθέτηση των επιπλέον μονάδων. Κάθε πλαίσιο επέκτασης, συνδέεται με το κεντρικό πλαίσιο ή με τα άλλα πλαίσια μέσω ειδικής μονάδας διασύνδεσης και καλωδίου.

Μονάδα τροφοδοσίας

Η μονάδα τροφοδοσίας χρησιμεύει για να δημιουργήσει από την τάση του δικτύου τις απαραίτητες εσωτερικές τάσεις για την τροφοδοσία αποκλειστικά των ηλεκτρονικών εξαρτημάτων, που υπάρχουν μέσα στον προγραμματιζόμενο ελεγκτή (τρανζίστορ, ολοκληρωμένα κ.λ.π). Οι τυπικές εσωτερικές τάσεις των ελεγκτών είναι: DC 5V, DC 9V, DC 24V.

Κεντρική μονάδα επεξεργασίας (CPU)

Είναι η βασική μονάδα του ελεγκτή, η οποία είναι υπεύθυνη για τη λειτουργία του αυτοματισμού. Η κεντρική μονάδα επεξεργασίας είναι στην ουσία ένας μικρουπολογιστής και διακρίνουμε σ' αυτήν όλα τα κύρια μέρη ενός μικρουπολογιστή, δηλαδή τον μικροεπεξεργαστή και τη μνήμη. Ο μικροεπεξεργαστής είναι αυτός που εκτελεί όλες τις λειτουργίες του προγραμματιζόμενου ελεγκτή.

Μονάδες εισόδων – εξόδων

Οι μονάδες εισόδων – εξόδων, αποτελούν τις μονάδες επικοινωνίας της κεντρικής μονάδας με τον έξω κόσμο, δηλαδή με τους αισθητήρες, τους διακόπτες κ.α., που δίνουν τις πληροφορίες (εντολές) στην κεντρική μονάδα, καθώς και με τα ρελέ ισχύος των κινητήρων, ηλεκτρομαγνητικές βαλβίδες, ενδεικτικές λυχνίες και γενικά τους αποδέκτες που εκτελούν τις εντολές της κεντρικής μονάδας.

Η κεντρική μονάδα, μπορεί να δεχτεί ψηφιακά σήματα εισόδου και εξόδου χαμηλής τάσης και πολύ μικρού ρεύματος. Η τάση που δέχεται είναι συνήθως 0 Voltγια το λογικό «0» και 5 Voltγια το λογικό «1». Το ρεύμα εισόδου καθώς και το ρεύμα εξόδου δεν μπορεί να ξεπεράσει τα λίγα mA. Οι μονάδες εισόδων και εξόδων αναλαμβάνουν να προσαρμόσουν τα σήματα εισόδου και εξόδου, που έχουμε στον αυτοματισμό, σε σήματα που μπορεί να δεχτεί η κεντρική μονάδα. Η προσαρμογή αυτή, γίνεται με χρήση ηλεκτρονικών στοιχείων ισχύος, είτε με τη χρήση κατάλληλων μικρο-ρελέ.

Κάθε σύστημα PLC καταλήγει πάντα σε ακροδέκτες. Οι ακροδέκτες αυτοί, ανήκουν στις μονάδες εισόδων και εξόδων του. Στους ακροδέκτες εισόδων καταλήγουν οι αγωγοί

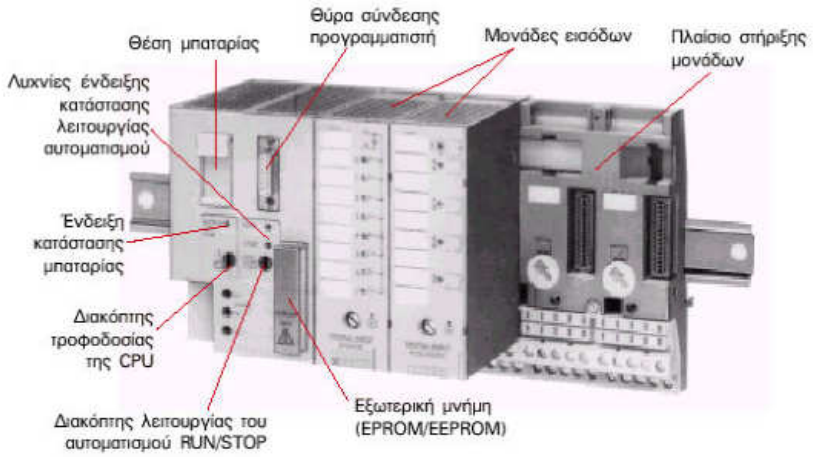
που έρχονται από αισθητήρες ή τερματικούς διακόπτες, διακόπρες μπουτόν κ.τ.λ. Στους ακροδέκτες εξόδων, καταλήγουν οι αγωγοί που τροφοδοτούν πηνία ρελέ ισχύος, ηλεκτρομαγνητικές βαλβίδες, λυχνίες ένδειξης και λοιπούν αποδέκτες.

Στους διάφορους τύπους PLC που υπάρχουν, οι μονάδες εισόδων και εξόδων αντιμετωπίζονται με διαφορετικό τρόπο. Γενικά όμως ισχύουν τα παρακάτω:

- Μια μονάδα εισόδων ή εξόδων, μπορεί να λειτουργεί με συνεχή τάση ή με εναλλασσόμενη. Τυπικές τάσεις λειτουργίας είναι: DC 24V,48V, 60Vκαι AC 24V, 48V, 115V, 230Vμε συνηθέστερες τις DC 24V, AC 115Vκαι AC 230V.
- Τα κυκλώματα και οι τάσεις των εισόδων είναι τελείως ανεξάρτητα από τα αντίστοιχα κυκλώματα των εξόδων. Επομένως η τάση για τις εισόδους μπορεί να είναι διαφορετική από την τάση για τις εξόδους. Στην περίπτωση, που αυτές οι τάσεις είναι ίδιες μπορεί να χρησιμοποιηθεί το ίδιο τροφοδοτικό (για συνεχείς τάσεις), ή μετασχηματιστής χειρισμού (για ACτάσεις) για τις εισόδους και για τις εξόδους.
- Η τάση εισόδων (δηλ. η τάση που φτάνει σε μια είσοδο, όταν ενεργοποιηθεί ο αντίστοιχος αισθητήρας) συνήθως διαχωρίζεται γαλβανικά από το υπόλοιπο εσωτερικό κύκλωμα του PLC. Τα ίδια ισχύουν και για τις εξόδους. Αν σε κάποιες μονάδες εξόδων δεν έχουμε γαλβανική απομόνωση πρέπει να προσέξουμε ιδιαίτερα το θέμα των γειώσεων.

Συσκευή προγραμματισμού

Η συσκευή προγραμματισμού είναι μια τελείως ξεχωριστή συσκευή από τη μονάδα αυτοματισμού. Χρησιμοποιείται για την εισαγωγή του προγράμματος στο PLC και την παρακολούθηση της εξέλιξης του αυτοματισμού μέσα από την οθόνη που διαθέτει. Με έναν μόνο προγραμματιστή μπορεί να γίνει ο χειρισμός όλων των μονάδων της ίδιας εταιρίας PLC σε μία αυτοματοποιημένη εγκατάσταση.



Σχήμα 1.3:Στοιχεία PLC

1.4 Η μνήμη της κεντρικής μονάδας

Η μνήμη της κεντρικής μονάδας επεξεργασίας (CPU) διακρίνεται σε μνήμη RAM, ROMκαι EEPROM.

- **Μνήμη RAM** : Η μνήμη RAM (Random Access Memory, μνήμη τυχαίας προσπέλασης) είναι εκείνη στην οποία μπορούμε να γράφουμε και να σβήνουμε και η οποία χάνει τα περιεχόμενα της μόλις πέσει η τροφοδοσία της. Στη μνήμη RAMη κεντρική μονάδα αποθηκεύει μία σειρά από πληροφορίες σε ξεχωριστές περιοχές εργασίας. Μπορούμε να διακρίνουμε τις εξής περιοχές:

    - Περιοχή μνήμης όπου αποθηκεύονται οι καταστάσεις των εισόδων και των εξόδων.
    - Περιοχή μνήμης όπου αποθηκεύονται οι ενδιάμεσες πληροφορίες που αφορούν τη λειτουργία του αυτοματισμού.
    - Περιοχή μνήμης των χρονικών.
    - Περιοχή μνήμης των απαριθμητών.
    - Περιοχή μνήμης όπου αποθηκεύονται τα προγράμματα του χρήστη, δηλαδή τα προγράμματα που λειτουργούν με ένα συγκεκριμένο αυτοματισμό.

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 για να δουλέψει το PLC.

Μνήμη EEPROM : Η μνήμη αυτή προγραμματίζεται και σβήνει ηλεκτρικά και την χρησιμοποιούν τα PLCδιότι τα δεδομένα δεν χάνονται με την απώλεια της τροφοδοσίας.

1.5 Αρχή λειτουργίας ενός προγραμματιζόμενου λογικού ελεγκτή

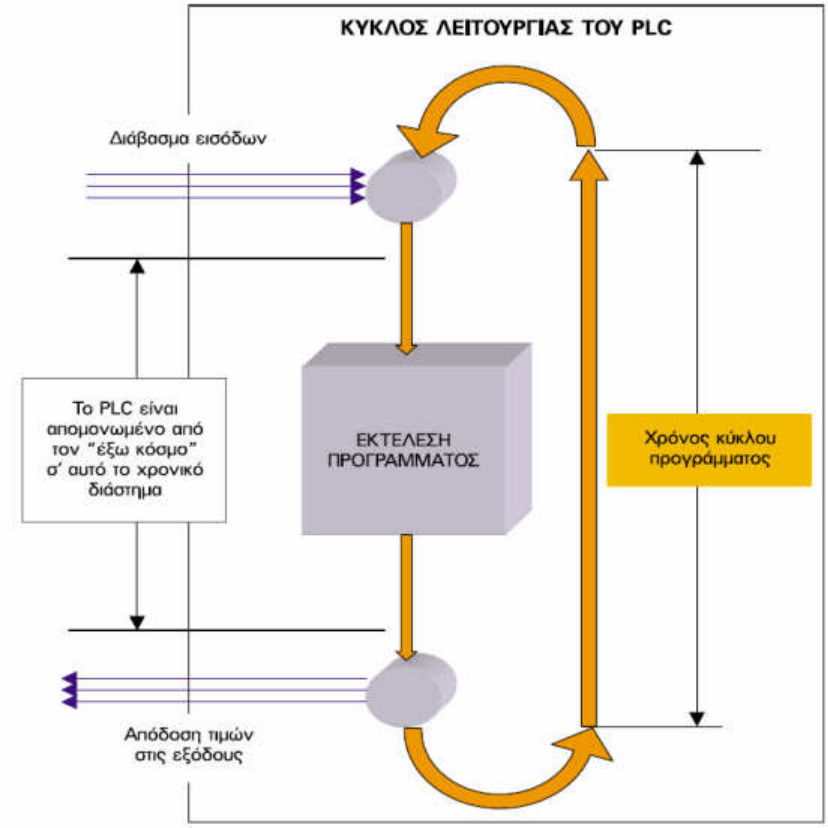
Έστω ότι ένα PLCβρίσκεται σε κατάσταση εκτέλεσης του αυτοματισμού (RUN). Τα βήματα που ακολουθεί κατά τη λειτουργία του είναι τα εξής:

1. Αρχικά ο μικροεπεξεργαστής, διαβάζει τις εισόδους. Αυτό σημαίνει ότι για κάθε είσοδο ελέγχει αν έχει υψηλή τάση (λογικό «1») ή χαμηλή τάση (λογικό «0»). Η τιμή 0 ή 1 για κάθε είσοδο αποθηκεύεται σε μια ειδική περιοχή της μνήμης η οποία ονομάζεται εικόνα εισόδων (input image). Την εικόνα εισόδων μπορείτε να την φανταστείτε σαν έναν πίνακα, όπου ο μικροεπεξεργαστής σημειώνει τις τιμές που διάβασε. Π.χ είσοδος I1 = 1, I2 = 0, I3 = 0 κ.τ.λ.
2. Εν συνεχεία, ο μικροεπεξεργαστής χρησιμοποιώντας σαν δεδομένα τις τιμές των εισόδων, που διάβασε, εκτελεί τις εντολές του προγράμματος. Το πρόγραμμα αυτό στην ουσία περιέχει μία σειρά από λογικές πράξεις. Η εκτέλεση του προγράμματος θα δώσει αποτελέσματα για τις εξόδους. Τα αποτελέσματα αυτά αποθηκεύονται στην ειδική περιοχή της μνήμης που ονομάζεται εικόνα εξόδων (output image). Όπως η εικόνα εισόδων, έτσι και η εικόνα εξόδων περιέχει την τιμή (0 ή 1) για κάθε έξοδο. Οι τιμές αυτές, προκύπτουν από την εκτέλεση των λογικών πράξεων του προγράμματος.
3. Έπειτα, ο μικροεπεξεργαστής θέτει τις τιμές της εικόνας εξόδων στις εξόδους. Αυτό σημαίνει ότι θα δοθεί υψηλή τάση σε όποια έξοδο έχει 1 και χαμηλή τάση σε όποια έξοδο έχει 0. Με την ολοκλήρωση των τριών αυτών βημάτων, συμπληρώνεται ένας πλήρης κύκλος λειτουργίας και η διαδικασία αρχίζει από την αρχή. Ο κύκλος λειτουργίας εκτελεί συνεχώς τα βήματα του κύκλου λειτουργίας. Στο Σχήμα 1.4 που ακολουθεί φαίνεται ένας κύκλος λειτουργίας PLC.

Ο χρόνος που χρειάζεται για να εκτελέσει το PLCένα πλήρη κύκλο λειτουργίας ονομάζεται χρόνος κύκλου και εξαρτάται από την ταχύτητα του επεξεργαστή του

PLCκαθώς επίσης και από τον αριθμό και το είδος των εντολών του προγράμματος. Αυτό σημαίνει πως στο ίδιο PLC, για ένα μεγαλύτερο πρόγραμμα έχουμε χρόνο κύκλου.

Ο χρόνος κύκλου αποτελεί ένα μέτρο σύγκρισης μεταξύ των PLC. Για είναι δυνατή η σύγκριση των PLCως προς την ταχύτητα εκτέλεσης ενός προγράμματος, ορίζουμε τον μέσο χρόνο κύκλου, σαν τον χρόνο κύκλου ενός προγράμματος που περιλαμβάνει 1 Kbytes δυαδικές εντολές. Πάντως στη χειρότερη περίπτωση και σε ένα αργό PLC, ο χρόνος κύκλου δεν υπερβαίνει τις μερικές εκατοντάδες millisecond.



Σχήμα 1.4:Κύκλος λειτουργίας PLC

1.6 Κύριες λειτουργίες PLC

Τα PLCσήμερα έχουν και επιπλέον λειτουργίες που βοηθούν στην δημιουργία του αυτοματισμού. Οι σημαντικότερς από αυτές είναι οι παρακάτω:

- **Λειτουργία απαριθμητών**. Οι απαριθμητές αποτελούν ακόμα ένα πολύ σημαντικό στοιχείο των PLC. Οι απαριθμητές μπορούν να απαριθμούν εξωτερικούς ή εσωτερικούς παλμούς. Η απαρίθμηση μπορεί να είναι προς τα πάνω (countup) ή προς τα κάτω (countdown). Η λειτουργία των απαριθμητών δεν είναι ίδια σε όλα τα PLC.
- **Δυνατότητα πραγματικού ρολογιού**. Με τη βοήθεια του μπορούμε να προγραμματίσουμε κάποιες εξόδους σε πραγματικό χρόνο, ημερομηνία και ώρα.
- **Αριθμητικές επεξεργασίες**. Τα σύγχρονα PLC έχουν προσεγγίσει πάρα πολύ τις δυνατότητες των ηλεκτρονικών υπολογιστών. Σχεδόν όλα τα PLC έχουν σήμερα τη δυνατότητα να επεξεργάζονται αριθμητικές πράξεις.
- **Αναλογικές είσοδοι – έξοδοι**. Τα PLC ενώ αρχικά ήρθαν για να αντικαταστήσουν τους αυτοματισμούς καλωδιωμένης λογικής (αυτοματισμούς με ρελέ), οι δυνατότητες τους έχουν εξαπλωθεί με προοπτική να καλύψουν πλήρως και τα συστήματα αυτομάτου ελέγχου, όπως είναι αναλογικοί έλεγχοι θερμοκρασίας, πίεσης, στάθμης, στροφών κινητήρων κ.τ.λ. Αυτό γίνεται δυνατό με την δυνατότητα των PLC να δέχονται και να επεξεργάζονται αναλογικές εισόδους, όπως και να παρέχουν αναλογικές εξόδους. Ο προγραμματιζόμενος ελεγκτής, μετατρέπει τις αναλογικές τιμές των εισόδων σε ψηφιακές και στη συνέχεια επεξεργάζεται τις τιμές αυτές αξιοποιώντας τις δυνατότητες για επεξεργασία ψηφιακών αριθμών.
- **Δικτύωση PLC**. Η εξέλιξη των PLC σήμερα αλλάζει τη μορφή της βιομηχανίας. Τα PLC μπορούν να συνδέονται μεταξύ τους ανταλλάσσοντας πληροφορίες, όπως και να συνεργάζονται με ηλεκτρονικούς υπολογιστές , οι οποίοι ασχολούνται με τον έλεγχο της αποθήκης και του λογιστηρίου του εργοστασίου. Όλα αυτά μαζί αποτελούν ένα βασικό δίκτυο αυτοματισμού (Computer Automati cNetwork).

1.7 Βασικά χαρακτηριστικά PLC

Τα βασικά τεχνικά χαρακτηριστικά ενός PLCείναι:

- Ο αριθμός των εισόδων
- Ο αριθμός των εξόδων
- Η τάση εισόδου (συνήθως 24VDC)
- Ο τύπος των εξόδων (Relay Transistor)

- Ο αριθμός των αναλογικών εισόδων και εξόδων (αν υπάρχουν) και
- Η τάση τροφοδοσίας (κυμαίνεται από 100 μέχρι 240VAC)

1.8 Μέγεθος των PLC

Υπάρχει μεγάλη ποικιλία στο μέγεθος των PLCs. Τυπικά οι ΠΛΕ κατατάσσονται σε τρεις μεγάλες κατηγορίες ανάλογα με το μέγεθός τους: μικρά, μεσαία και μεγάλα, το καθένα με συγκεκριμένα χαρακτηριστικά λειτουργίας. Η μικρή κατηγορία καλύπτει μονάδες μέχρι 128 I/Os και μνήμη μέχρι 2 Kbytes. Αυτά τα PLCs είναι ικανά να παρέχουν απλό έως και σύνθετο επίπεδο ελέγχου συσκευών. Ειδικές μονάδες I/Οκάνουν αυτά τα PLCs κατάλληλα για ρύθμιση θερμοκρασίας, πίεσης, ροής, βάρους, θέσης, ή οποιαδήποτε τύπο αναλογικής λειτουργίας, που συνήθως συναντάται σε εφαρμογές ελέγχου παραγωγής ή επεξεργασίας. Στην μεγάλη κατηγορία των PLCs, φυσικά, έχουμε τις πιο πολυσύνθετες μονάδες της οικογένειας των PLCs.

Αυτές έχουν μέχρι 8192 I/Oskαι μνήμες μέχρι 750Kbytes, PLCsαυτού του μεγέθους έχουν πρακτικά απεριόριστες εφαρμογές. Τα μεγάλα PLCs μπορούν να ελέγξουν μεμονωμένες διαδικασίες παραγωγής, ή ολόκληρη εργοστασιακή μονάδα. Ο παράγων κλειδί στην επιλογή ενός PLC, είναι να καθορίσουμε ακριβώς την απαιτούμενη λειτουργία της μονάδας. Γενικά, δεν συνιστάται να χρησιμοποιηθεί ένα σύστημα PLC, μεγαλύτερο από ότι οι ανάγκες το απαιτούν. Όμως οι μελλοντικές απαιτήσεις πρέπει να προβλεφθούν, ώστε το σύστημα να έχει το κατάλληλο μέγεθος για να ανταποκριθεί στις πιθανές μελλοντικές απαιτήσεις της εφαρμογής.

Από την δημιουργία τους τα PLCs, έχουν ικανοποιητικά εφαρμοστεί ουσιαστικά σε κάθε τομέα της βιομηχανίας. Ο κατάλογος περιλαμβάνει χαλυβουργία, μονάδες χαρτιού και πολτού, μονάδες παραγωγής ισχύος κ.α. Τα PLCs εκτελούν μία μεγάλη ποικιλία ελέγχων, από επαναλαμβανόμενο έλεγχο ON/OFFαπλών συσκευών, έως και πολυσύνθετους ελέγχους παραγωγής και επεξεργασίας.

1.9 Πλεονεκτήματα

- Το κόστος των PLCsεπιτρέπει σήμερα και στο πιο απλό μηχάνημα, που έχει κάποιον ηλεκτρικό έλεγχο να το χρησιμοποιεί.

- Είναι συσκευές γενικής χρήσης, δεν είναι κατασκευασμένοι για ένα συγκεκριμένο είδος παραγωγής.
- Με τη χρήση τους δεν μας ενδιαφέρει ο αριθμός των επαφών, χρονικών, απαριθμητών κλπ. που θα χρησιμοποιηθούν μιας και αποτελούν στοιχεία μνήμης της CPU και όχι φυσικές οντότητες.
- Το PLC διαθέτει τέτοιες δυνατότητες, που μπορεί να χρησιμοποιηθεί για οποιαδήποτε μηχάνημα.
- Ο προγραμματισμός του είναι σχετικά εύκολος. Μπορούμε να αλλάξουμε τη διαδικασία στη λειτουργία ενός μηχανήματος, που ελέγχεται από PLCδιαφοροποιώντας το πρόγραμμα του και δεν είναι ανάγκη να αλλάξουμε ηλεκτρικά μέσα και καλωδίωση.
- Η χρήση του PLC στο κύκλωμα ελέγχου φέρνει και πιο χαμηλή κατανάλωση ηλεκτρικής ενέργειας.
- Είναι εύκολη η διασύνδεση μεταξύ τους για ανταλλαγή πληροφοριών, ο τηλεχειρισμός και η τηλε-εποπτεία, ο εξ' αποστάσεως προγραμματισμός τους και η σύνδεση τους στο internet.
- Γενικά η παραγωγικότητα των μηχανημάτων με τη χρήση του PLCαυξάνεται, οι βλάβες και τα σταματήματα μειώνονται, η αυτοματοποίηση γίνεται εύκολη.

1.10 Είδη PLC

Στην αγορά, υπάρχουν δύο τύποι προγραμματιζόμενων λογικών ελεγκτών:

- **Τα Compact PLC:**Σ' αυτήν την κατηγορία, ανήκουν τα PLCπου όλα τα επιμέρους στοιχεία που απαρτίζουν ένα PLC, είναι ενσωματωμένα σε μια συσκευή. Είναι περιορισμένων δυνατοτήτων καθώς έχουν 48 το πολύ εισόδους και εξόδους, όλες με τα ίδια χαρακτηριστικά, καθώς και μικρό αριθμό χρονικών και απαριθμητών. Το πλεονέκτημά τους είναι το χαμηλό κόστος τους.
- **Τα Modular PLC:** Σ' αυτήν την κατηγορία, κάθε μονάδα (module) του PLCείναι ξεχωριστή και συνδέονται όλες μαζί πάνω στο πλαίσιο τοποθέτησης των μονάδων. Είναι επεκτάσιμα και χρησιμοποιούνται συνήθως όταν έχουμε μεγάλο αριθμό εισόδων και εξόδων. Με αυτό τον τρόπο, μπορούμε να επιλέξουμε την

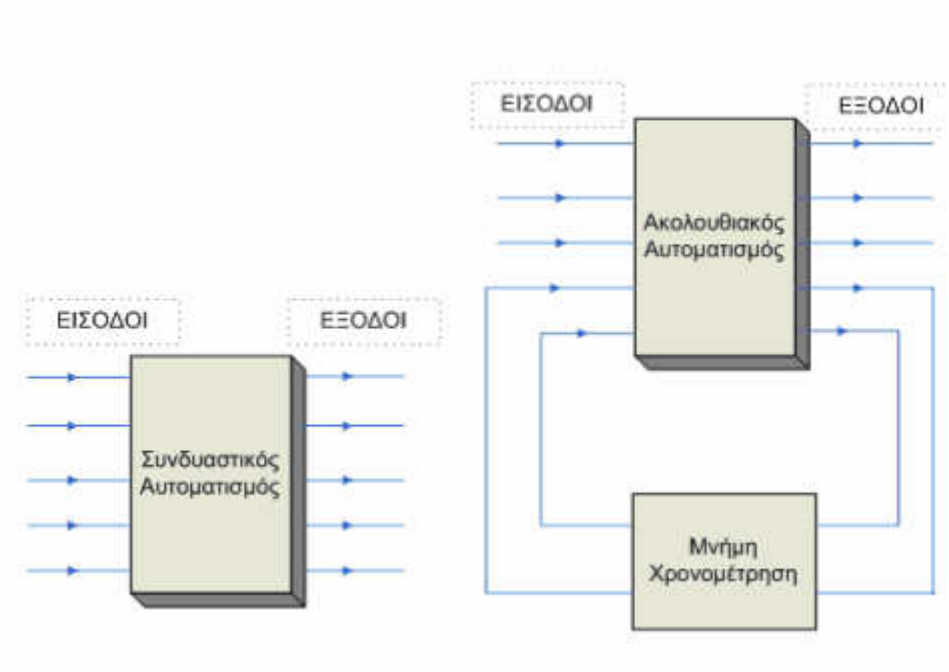
κεντρική μονάδα και τις μονάδες εισόδων – εξόδων με τα χαρακτηριστικά που επιθυμούμε.

1.11 Ανάπτυξη προγράμματος σε προγραμματιζόμενο λογικό ελεγκτή

Ο προγραμματισμός σ' ένα PLC μπορεί να γίνει είτε με συνδυαστικούς είτε με ακολουθιακούς αυτοματισμούς. Αυτό γίνεται, γιατί οι βασικές διαφορές στον προγραμματισμό των PLC εμφανίζονται όταν υπάρχει χρήση χρονικών, απαριθμητών και των λοιπών ειδικών συναρτήσεων των ακολουθιακών αυτοματισμών.

Συνδυαστικός αυτοματισμός: Είναι ο αυτοματισμός στον οποίο οι έξοδοι εξαρτώνται μόνο από τις εισόδους. Αυτό σημαίνει ότι οι κινητήρες, βαλβίδες και οι υπόλοιποι αποδέκτες του αυτοματισμού λαμβάνουν εντολές μόνο από τους αισθητήρες και τους διακόπτες εισόδου και δεν εξαρτώνται από το χρόνο ή από προηγούμενες καταστάσεις των εξόδων.

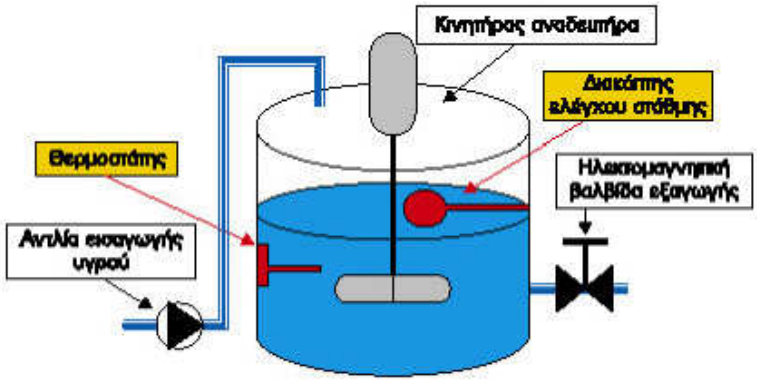
Ακολουθιακός αυτοματισμός: είναι ο αυτοματισμός στον οποίο οι έξοδοι εξαρτώνται όχι μόνο από τις εισόδους, αλλά και από το χρόνο ή από προηγούμενες καταστάσεις των εξόδων. Σχηματικά οι δύο κατηγορίες των αυτοματισμών φαίνονται στην παρακάτω εικόνα.



Σχήμα 1.5:Κατηγορίες αυτοματισμού

Παράδειγμα συνδυαστικού και ακολουθιακού αυτοματισμού

Ας υποθέσουμε ότι έχουμε μια δεξαμενή, η οποία γεμίζει με κάποιο υγρό μέσω μιας αντλίας και που αδειάζει ανοίγοντας μία βαλβίδα εξαγωγής. Επίσης υπάρχει ένας αναδευτήρας και ένας φλοτεροδιακόπτης. Ο αυτοματισμός αυτός, είναι συνδυαστικός γιατί οι έξοδοι (αντλία, βαλβίδα, αναδευτήρας) εξαρτώνται μόνο από τις καταστάσεις των εισόδων (θερμοστάτης, φλοτεροδιακόπτης).



Σχήμα 1.6:Παράδειγμα συνδυαστικού αυτοματισμού

1.12 Προγραμματιστικά χαρακτηριστικά και ονοματολογία στοιχείων ενός PLC

Όταν ξεκινάει η μελέτη πως θα προγραμματιστεί ένα PLC, πρέπει να γνωρίζεται ο αριθμός και η περιγραφή των :

1.12.1 Εισόδων

Οι είσοδοι ενός PLCσυμβολίζονται με το γράμμα I (Input). Στα μικρά συμπαγή PLCτο γράμμα Iακολουθεί ένας απλός αύξοντας αριθμός, ξεκινώντας από το 1 (ή το 0) και φθάνοντας στο πλήθος των εισόδων π.χ I1, I2, I3,κ.λ.π. Στα modular PLC, όπου οι είσοδοι βρίσκονται σε μονάδες εισόδων , το γράμμα I ακολουθούν δύο αριθμοί που χωρίζονται με τελεία. Ο πρώτος αριθμός χαρακτηρίζει συνήθως τη θέση της μονάδας που βρίσκεται η είσοδος και ο δεύτερος αριθμός χαρακτηρίζει την είσοδο πάνω στην μονάδα.

Για παράδειγμα, έχουμε εισόδους I0.0, I0.1, …,I1.1, I1.2 κλπ. Μονοσήμαντα μια είσοδος χαρακτηρίζεται από δύο στοιχεία:

- σε ποια οκτάδα (byte) ανήκει
- σε ποια επιμέρους θέση στα όρια αυτής της οκτάδας (bit)

Χαρακτηρισμός

Ix.y

x: Διεύθυνση byte (0…n, ανάλογα με τη χρησιμοποιούμενη CPU)

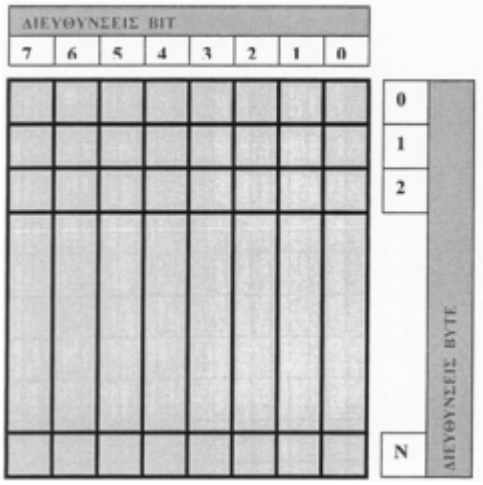
y :Διεύθυνσηbit (0…7)

Έχουμε τη δυνατότητα να παρουσιάσουμε ή να ζητήσουμε:

Byte εισόδων: π.χ IB 3, με αυτήν την ονοματολογία δηλώνουμε τις εισόδους I 3.0… I 3.7

Word εισόδων: π.χ IW 2, με αυτήν την ονοματολογία δηλώνουμε τις εισόδους I2.0…. I7.0, I 3.0… I 3.7

Double Word εισόδων :π.χ ID4, με αυτήν την ονοματολογία δηλώνουμε τις εισόδους I4.0….I4.7, I5.0…..I5.7, I6.0…..I6.7, 17.0……17.7.



Σχήμα 1.7:Περιοχή απεικόνισης της μνήμης εισόδων PII

1.12.2 Εξόδων

Τα ίδια, που ισχύουν για τις εισόδους, ισχύουν και για τις εξόδους. Οι έξοδοι ενός PLC συμβολίζονται με το γράμμα Q ή το O (Output). Μονοσήμαντα μια έξοδος χαρακτηρίζεται από δύο στοιχεία:

- σε ποια οκτάδα (byte) ανήκει
- σε ποια επιμέρους θέση στα όρια αυτής της οκτάδας (bit)

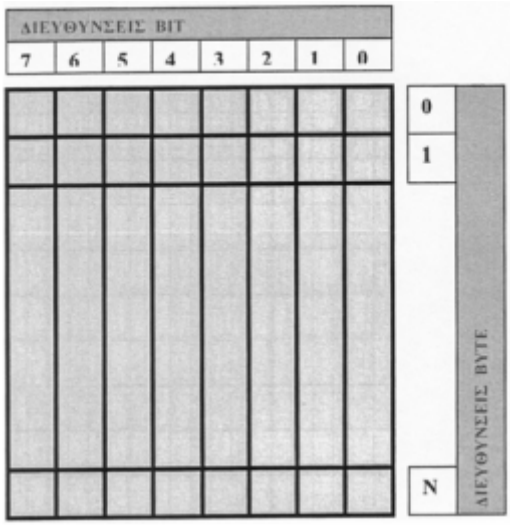
Χαρακτηρισμός

Η ονοματολογία της έχει τη μορφή :

Qx.y όπου x: Διεύθυνση byte

y :Διεύθυνση bit (0…7)

Όπως στις ψηφιακές εισόδους έτσι για τις ψηφιακές εξόδους έχουμε byte εξόδων, Word εξόδων, double word εξόδων.



Σχήμα 1.8:Περιοχή απεικόνισης της μνήμης εξόδων PIQ

1.12.3 Βοηθητικών μνημών

Προκειμένου να γραφτεί ο επαναλαμβανόμενος κώδικας τόσες φορές όσες χρειάζονται, πράγμα που κοστίζει σε χρόνο και σε μνήμη προγράμματος, είναι προτιμότερη η χρήση βοηθητικών διευθύνσεων. Καταγράφεται μια φορά η λογική,

αποθηκεύεται σ’ μια βοηθητική διεύθυνση και αυτή χρησιμοποιείται όσες φορές και σε όποιο σημείο του προγράμματος είναι αναγκαίο.

Οι βοηθητικές μνήμες παίζουν το ρόλο των βοηθητικών ρελέ στον κλασσικό αυτοματισμό. Χρησιμοποιούνται στο πρόγρμμα για να αποθηκευτούν ορισμένες καταστάσεις. Τα bitεδώ παρομοιάζονται όσον αφορά τη λειτουργία τους με τις εξόδους, με τη διαφορά ότι αυτά δεν απεικονίζονται σε LED. Επίσης, στα διάφορα PLC, θα τις συναντήσουμε με το όνομα Markers.

Οι βοηθητικές μνήμες ουσιαστικά είναι θέσεις μνήμης, ατις οποίες αποθηκεύονται ενδιάμεσες λογικές καταστάσεις και πληροφορίες και χαρακτηρίζονται με ένα γράμμα ακολουθούμενο από έναν αριθό ή δύο αριθμούς που χωρίζονται με τελεία.

Χαρακτηρισμός

Mx.y

x: Διεύθυνση byte (0…n, ανάλογα με τη χρησιμοποιούμενη CPU)

y :Διεύθυνσηbit (0…7)

1.12.4 Τις ειδικές συναρτήσεις του PLC

Οι ειδικές συναρτήσεις κατά σειρά σπουδαιότητας είναι:

- χρονικά
- απαριθμητές
- συγκριτές
- γεννήτριες παλμοσειρών
- μετρητής πραγματικού χρόνου

1.13 Γνωρίστε τα plc με την μορφή ερωταπαντήσεων

1) Τι σημαίνει PLC

Plc (Programmable Logic Controllers) λέγονται οι προγραμματιζόμενοι λογικοί ελεγκτές και ανήκουν στην κατηγορία των ψηφιακών ηλεκτρονικών

2) Τι είναι τα PLC

Το PLC είναι μία ηλεκτρονική διάταξη η οποία από την άποψη της λειτουργίας θα μπορούσε να προσομοιωθεί με έναν πίνακα αυτοματισμού. Έχει δηλαδή εισόδους και εξόδους που συνδέονται με τα στοιχεία μιας εγκατάστασης και βέβαια έναν αλγόριθμο που καθορίζει ότι κάποιος συνδυασμός εισόδων παράγει ένα αποτέλεσμα στις εξόδους. η λογική της λειτουργίας πού ενσωματώνεται στο PLC μέσω του προγραμματισμού του είναι μεταβαλλόμενη.

3) Από τι αποτελούνται τα PLC

σε ότι αφορά το υλικό όλα τα PLC αποτελούνται 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) πού έχουμε αποθηκεύσει στην μνήμη του.

4) Πως λειτουργεί ένα σύστημα PLC

Αρχικά η CPU διαβάζει τις εισόδους, δηλαδή παρατηρεί την κάθε είσοδο, και αν σε αυτή έχει εμφανισθεί τάση (πού σημαίνει ότι έχει κλείσει ο διακόπτης) καταχωρεί ένα λογικό 1 σε μία περιοχή της μνήμης του που είναι ειδική γι’ αυτό τον σκοπό (Input Image). Η περιοχή αυτή περιέχει σε κάθε στιγμή την κατάσταση των εισόδων και λειτουργεί σαν ενδιάμεσος σταθμός ανάμεσα στον "έξω κόσμο" και την CPU. Στην συνέχεια εκτελείται το πρόγραμμα δηλαδή εξετάζεται η τιμή των εισόδων και αποφασίζεται η τιμή της εξόδου η οποία και καταχωρείται σε μία αντίστοιχη περιοχή μνήμης εξόδου (Output Image). Τέλος η περιοχή της μνήμης εξόδου μεταφέρεται στην κάρτα εξόδου και διεγείρει με την σειρά της το ρελέ.

5) Ποια η διαφορά τους εν σχέση με άλλα συστήματα αυτοματισμού

Διατηρώντας τις ίδιες ακριβώς καλωδιώσεις και αλλάζοντας μόνο το πρόγραμμα, το σύστημα μπορεί να συμπεριφέρεται εντελώς διαφορετικά. Αυτή είναι βέβαια και η μεγάλη διαφορά του PLC από οποιοδήποτε άλλο σύστημα αυτοματισμού που καθορίζει και το όνομα του δηλαδή προγραμματιζόμενος λογικός ελεγκτής.

6) Ποια τα πλεονεκτήματα των PLC

Συγκριτικά με τον κλασσικό αυτοματισμό τα πλεονεκτήματα του προγραμματισμού με PLC είναι πάρα πολλά. Ενδεικτικά μπορεί να γίνει αναφορά σε ότι: • Είναι συσκευές γενικής χρήσεως - δεν είναι κατασκευασμένοι για ένα συγκεκριμένο είδος

παραγωγής. • Δεν ενδιαφέρει ο συνολικός αριθμός των επαφών, χρονικών, απαριθμητών κλπ. που Θα χρησιμοποιηθούν μιας και αποτελούν στοιχεία μνήμης της CPU και όχι φυσικές οντότητες. • Η λειτουργία του αυτοματισμού μπορεί ν' αλλάξει σε οποιοδήποτε στάδιο θελήσουμε (μελέτη, κατασκευή, Θέση σε λειτουργία ή αργότερα) χωρίς επέμβαση στο υλικό. • Εύκολος οπτικός εντοπισμός με μία ματιά, της λειτουργίας ή μη στοιχείων της εγκατάστασης με τη βοήθεια των LED που υπάρχουν σε όλες τις κάρτες εισόδου / εξόδου. Με τη βοήθεια συσκευής προγραμματισμού μπορεί να παρακολουθηθεί και η ροή εκτέλεσης του προγράμματος. • Η κατασκευή του πίνακα που θα τοποθετηθεί το PLC γίνεται παράλληλα με τον προγραμματισμό του, πράγμα το οποίο οδηγεί στη συντομότερη παράδοση του αυτοματισμού. • Πολύ συχνό είναι το φαινόμενο ο τεχνικός να κληθεί να επισκευάσει μια βλάβη και να δει έκπληκτος ότι άλλα υπάρχουν στα σχέδια και άλλα βλέπει αυτός στην εγκατάσταση. Το πρόβλημα αυτό δεν υπάρχει στα PLC αφού πάντα υπάρχει μόνο ένα "σχέδιο" αποθηκευμένο - το τελευταίο πρόγραμμα που του έχουμε περάσει. Εάν απαιτούνται περισσότερα προγράμματα, αυτό είναι δυνατό με τη χρήση δισκετών. • Τα PLC ως ηλεκτρονικές συσκευές καταλαμβάνουν πολύ μικρότερο χώρο στο πίνακα σε σχέση με τα υλικά του κλασσικού αυτοματισμού και καταναλώνουν πολύ λιγότερη ενέργεια από αυτά. • Τοποθετούνται άφοβα και σε πεδία ισχύος - ο κατασκευαστής δίνει οδηγίες γι’ αυτές τις περιπτώσεις οι οποίες πρέπει να τηρούνται (αποστάσεις, γειώσεις κλπ.). • Η γλώσσες προγραμματισμού καλύπτουν όλο το φάσμα των ανθρώπων που καλούνται να ασχοληθούν με την τεχνολογία αυτή - υπάρχει γλώσσα προγραμματισμού γι' ανθρώπους με γνώση στο συμβατικό αυτοματισμό (Ladder), γλώσσες για όσους έχουν υπόβαθρο σε υπολογιστές (Statement List, SCL, FBD, C++) καθώς και γλώσσες εξειδικευμένες για διάφορες τεχνολογίες (GRAPH 7, HIGRAPH, CSF).

7) Ποια η δομή ενός συστήματος PLC

Στην αγορά υπάρχουν σήμερα πάρα πολλά μοντέλα PLC κατασκευασμένα από πολλές εταιρίες. Η επιλογή ενός προγραμματιζόμενου ελεγκτή (τύπος, μέγεθος, κόστος) εξαρτάται από το πλήθος των στοιχείων που δίνουν εντολή σ' αυτόν (είσοδοι) και το πλήθος των στοιχείων που δέχονται εντολή απ' αυτόν (έξοδοι), καθώς και από το πλήθος των λειτουργιών που απαιτείται να κάνει ο αυτοματισμός (μέγεθος προγράμματος, δηλ. απαιτούμενη μνήμη και δυνατότητες της κεντρικής μονάδας). Ανεξάρτητα όμως από τον τύπο και το μέγεθος, ένας προγραμματιζόμενος ελεγκτής, συνίσταται από τα εξής απαραίτητα στοιχεία:

Α. Πλαίσιο τοποθέτησης των μονάδων.

Β. Μονάδα τροφοδοσίας.

Γ. Κεντρική μονάδα επεξεργασίας (CPU) που αποτελεί τον εγκέφαλο του PLC.

Δ. Μονάδες εισόδων / εξόδων.

Ε. Συσκευή προγραμματισμού.

8) Ποια είναι η αρχή λειτουργίας ενός PLC

Έστω ότι ένα PLC βρίσκεται σε κατάσταση εκτέλεσης του αυτοματισμού (RUN). Τα βήματα που ακολουθεί κατά τη λειτουργία του είναι τα εξής:

Βήμα 1°: Στην αρχή ο μικροεπεξεργαστής "διαβάζει" της εισόδους. Αυτό σημαίνει ότι για κάθε είσοδο ελέγχει αν έχει "υψηλή" τάση (λογικό "1") ή "χαμηλή" τάση (λογικό "0"). Η τιμή "0" ή "1" για κάθε είσοδο αποθηκεύεται σε μια ειδική περιοχή της μνήμης η οποία ονομάζεται εικόνα εισόδων (input image). Την εικόνα εισόδων μπορείτε να την φανταστείτε σαν έναν πίνακα, όπου ο μικροεπεξεργαστής σημειώνει τις τιμές που διάβασε. Π.χ. είσοδος I1="1". I2="0", I3="0" κ.ο.κ.

Βήμα 2ο: Στη συνέχεια ο μικροεπεξεργαστής χρησιμοποιώντας σαν δεδομένα τις τιμές των εισόδων, που διάβασε, εκτελεί τις εντολές του προγράμματος. Το πρόγραμμα αυτό στην ουσία περιέχει μια σειρά από λογικές πράξεις. Η εκτέλεση του προγράμματος θα δώσει αποτελέσματα για τις εξόδους. Τα αποτελέσματα αυτά αποθηκεύονται στην ειδική περιοχή της μνήμης που ονομάζεται εικόνα εξόδων (output image). Όπως η εικόνα εισόδων, έτσι και η εικόνα εξόδων περιέχει την τιμή ("0" ή "1") για κάθε έξοδο. Σημειώνουμε ότι οι τιμές αυτές προκύπτουν από την εκτέλεση των λογικών πράξεων του προγράμματος.

Βήμα 3°: Στη συνέχει ο μικροεπεξεργαστής θέτει τις τιμές της εικόνας εξόδων στις εξόδους. Αυτό σημαίνει ότι θα δοθεί "υψηλή" τάση σε όποια έξοδο έχει "1" και χαμηλή τάση σε όποια έξοδο έχει "0". Με τη συμπλήρωση του 3ου βήματος συμπληρώνεται ένας πλήρης κύκλος λειτουργίας και η διαδικασία αρχίζει από την αρχή. Ο κύκλος λειτουργίας εκτελείται συνεχώς όσο το PLC βρίσκεται σε κατάσταση RUN. Δηλαδή ένα PLC εκτελεί συνεχώς τα βήματα του κύκλου λειτουργίας.

9) Ποιες είναι οι κύριες λειτουργίες των PLC

Τα PLC σήμερα έχουν και επιπλέον λειτουργίες που βοηθούν στην δημιουργία του αυτοματισμού. Οι λειτουργίες αυτές αυξάνουν συνεχώς καθώς τα PLC εξελίσσονται με ταχύτατους ρυθμούς. Αναφέρονται ενδεικτικά οι σημαντικότερες από αυτές.

• Λειτουργία απαριθμητών. Οι απαριθμητές αποτελούν ακόμα ένα πολύ σημαντικό στοιχείο των PLC. Οι απαριθμητές μπορούν να απαριθμούν εξωτερικούς ή εσωτερικούς παλμούς. Η απαρίθμηση μπορεί να είναι προς τα πάνω (count up) ή προς τα κάτω (count down). Η λειτουργία των απαριθμητών δεν είναι ίδια σε όλα τα PLC.

• Δυνατότητα πραγματικού ρολογιού, μέσω του οποίου μπορούμε να προγραμματίσουμε κάποιες εξόδους σε πραγματικό χρόνο, ημερομηνία και ώρα.

• Αριθμητικές επεξεργασίες. Τα σύγχρονα PLC έχουν προσεγγίσει πάρα πολύ τις δυνατότητες των ηλεκτρονικών υπολογιστών. Σχεδόν όλα τα PLC έχουν σήμερα τη δυνατότητα να επεξεργάζονται αριθμητικές πράξεις.

• Αναλογικές είσοδοι-έξοδοι. Τα PLC ενώ αρχικά ήρθαν για να αντικαταστήσουν τους αυτοματισμούς καλωδιωμένης λογικής (αυτοματισμούς με ρελέ), οι δυνατότητές τους έχουν εξαπλωθεί με προοπτική να καλύψουν πλήρως και τα συστήματα αυτομάτου ελέγχου, όπως είναι αναλογικοί έλεγχοι θερμοκρασίας, πίεσης, στάθμης, στροφών κινητήρων κλπ. Αυτό γίνεται δυνατό με την δυνατότητα των PLC να δέχονται και να επεξεργάζονται αναλογικές εισόδους, όπως και να παρέχουν αναλογικές εξόδους. Το PLC μετατρέπει τις αναλογικές τιμές των εισόδων σε ψηφιακές τιμές και στη συνέχεια επεξεργάζεται τις τιμές αυτές αξιοποιώντας τις δυνατότητες για επεξεργασία ψηφιακών αριθμών όπως ήδη προαναφέρθηκε. Η δυνατότητα επεξεργασίας αναλογικών σημάτων έχει δώσει άλλη δυναμική στην εξέλιξη στα PLC.

10) Πόσα κύρια είδη προγραμματισμού των plc υπάρχουν

Υπάρχουν δυο συνδυαστικός αυτοματισμός και ακολουθιακός αυτοματισμός

11) Τι είναι ο συνδυαστικός αυτοματισμός των PLC

Είναι ο αυτοματισμός στον οποίο οι έξοδοι εξαρτώνται μόνο από τις εισόδους. Αυτό σημαίνει ότι οι κινητήρες, βαλβίδες και οι υπόλοιποι αποδέκτες του αυτοματισμού λαμβάνουν εντολές μόνο από τους αισθητήρες και τους διακόπτες εισόδου και δεν εξαρτώνται από το χρόνο ή από προηγούμενες καταστάσεις των εξόδων.

12) Τι είναι ο Ακολουθιακός αυτοματισμός των PLC

Είναι ο αυτοματισμός στον οποίο οι έξοδοι εξαρτώνται όχι μόνο από τις εισόδους, αλλά και από το χρόνο ή και από προηγούμενες καταστάσεις των εξόδων

13) Ποια τα προγραμματιστικά χαρακτηριστικά των στοιχείων του PLC

Είναι η είσοδος η έξοδος , οι βοηθητικές μνήμες ,και οι ειδικές συναρτήσεις του plc

14) Ποιες οι είσοδοι των PLC

Οι είσοδοι ενός PLC συμβολίζονται με το γράμμα I (Input). Μονοσήμαντα μια είσοδος χαρακτηρίζεται από δύο στοιχεία: i. σε ποια οκτάδα ανήκει (byte) και ii. σε ποια επιμέρους θέση στα όρια αυτής της οκτάδας (bit).

Χαρακτηρισμός

I x.y x - Διεύθυνση byte (0 ... n, ανάλογα με τη χρησιμοποιούμενη CPU)

y - Διεύθυνση bit (0 ... 7)

Byte εισόδων: π.χ. IB 5, περιλαμβάνει τα bit I 5.0 ... I 5.7

Word εισόδων: π.χ. IW 8, περιλαμβάνει τα byte I B8 και I B9

Double Word εισόδων: π.χ. I D4, περιλαμβάνει τις word IW4 και IW6

15) Ποιες οι έξοδοι των PLC

Τα ίδια, που ισχύουν για τις εισόδους, ισχύουν και για τις εξόδους. Οι έξοδοι ενός PLC συμβολίζονται με το γράμμα Q (Output). Μονοσήμαντα μια έξοδος χαρακτηρίζεται από δύο στοιχεία: i. σε ποια οκτάδα ανήκει (byte) και ii. σε ποια επιμέρους θέση στα όρια αυτής της οκτάδας (bit).

Χαρακτηρισμός Q χ.y x - Διεύθυνση byte (0.. n, ανάλογα με τη χρησιμοποιούμενη CPU) y - Διεύθυνση bit (0 ... 7)

Παράδειγμα Q5.0, Q 12.7, Q2.1

Byte εξόδων: π.χ. QB 5, περιλαμβάνει τα bit Q 5.0 ... Q 5.7

Word εξόδων: π.χ. QW 8, περιλαμβάνει τα byte QB8 και QB9

Double Word εξόδων: π.χ. QD4, περιλαμβάνει τις word QW4 και QW6

16) Τι είναι οι βοηθητικές μνήμες των plc

Οι βοηθητικές μνήμες παίζουν το ρόλο των βοηθητικών ρελέ στο κλασσικό αυτοματισμό. Χρησιμοποιούνται στο πρόγραμμά για να αποθηκευτούν ορισμένες καταστάσεις. Τα bit εδώ παρομοιάζονται όσον αφορά τη λειτουργία τους με τις εξόδους, με τη διαφορά ότι αυτά δεν απεικονίζονται σε LED (δεν πηγαίνουν απ' ευθείας στην εγκατάσταση και φαίνεται η κατάστασή τους μόνο με τη βοήθεια συσκευής προγραμματισμού).

Χαρακτηρισμός M x.y x - Διεύθυνση byte (0 ... n, ανάλογα με τη χρησιμοποιούμενη CPU) y - Διεύθυνση bit (0 ... 7)

Παράδειγμα M 15.0, M 102.7, M 42.1

Byte Βοηθητικών : π.χ. MB 7, περιλαμβάνει τα bit M 7.0 ... M 7.7

Word βοηθητικών: π.χ. MW 6, περιλαμβάνει τα byte MB6 και MB7

Double Word Βοηθητικών : π.χ. MD4, περιλαμβάνει τις word MW4 και MW6

17) Ποιες οι ειδικές συναρτήσεις

Οι ειδικές συναρτήσεις είναι:

ο χρονικά ο απαριθμητές ο συγκριτές ο γεννήτριες παλμοσειρών ο μετρητής πραγματικού χρόνου

18) Που διαφέρει το plc από τον Η/Υ

Α) Μπορεί να εγκατασταθεί, χρησιμοποιηθεί και συντηρηθεί εύκολα από τεχνικούς βιομηχανικών εγκαταστάσεων

Β) Εκτελεί μόνο ένα συγκεκριμένο πρόγραμμα κάθε φορά, αντίθετα με τη δυνατότητα του multitasking του multitasking των Η/Υ

Γ) Η ένδειξη βλαβών του γίνεται μέσω ενδεικτικών λυχνιών

Δ) Λειτουργεί σταθερά σε βιομηχανικό περιβάλλον

19) Που μοιάζει στον Η/Υ

Μοιάζει στον Η/Υ με την έννοια ότι μπορεί να εκτελεί το πρόγραμμα που είναι αποθηκευμένο στην μνήμη του

20) Τι εννοούμε πρόγραμμα των plc

Λέγοντας πρόγραμμα εννοούμε μια ακολουθιακή λειτουργία ελέγχου. Στην περίπτωση του plc η εκτέλεση του προγράμματος γίνεται σε πραγματικό χρόνο και σε βιομηχανικό περιβάλλον

21) Ποιος είναι ο σκοπός του προγράμματος

Ο σκοπός κάθε προγράμματος είναι να εξηγεί στο PLC πώς να αλληλοεπιδρά με το περιβάλλον του, δηλ. την εγκατάσταση,

Το πρόγραμμα πρέπει να εκτιμά τις πληροφορίες που του δίνονται και με βάση κάποια σοφή λογική/αριθμητική διαδικασία να αποφασίζει την επίδραση που επιθυμούμε να έχει, το plc , στην ελεγχόμενη εγκατάσταση

22) Που χρησιμοποιείται το PLC

- Στη μεταλλουργία: Επεξεργασία κοκ, τροφοδοσία κλιβάνων, αυτοματισμός χυτηρίων, ανάλυση φωταερίου, έλεγχος
- Στις βιομηχανίες κατασκευών και αυτοκινητοβιομηχανίες: Γραμμές παραγωγής και συναρμολόγησης, εργαλειομηχανές.
- Χημικές βιομηχανίες: Έλεγχος μονάδων παραγωγής, μμέτρηση ή ανάμειξη, επεξεργασία πλαστικών, σύνδεση με μηχανές ελαστικών.
- Πετροχημικές βιομηχανίες: Σταθμοί άντλησης, έλεγχος και επίβλεψη αγωγών, διανομή αερίων και υγρών καυσίμων.
- Βιομηχανίες αγροτικών προϊόντων & τροφίμων: Γραμμή παραγωγής, διαδικασία ξήρανσης, κλιματισμός στην αποθήκευση.

- Μεταφορά και διαχείριση υλικών: Συσκευασία κιβωτίων, παλετών, έλεγχο μεταφορικών και ανυψωτικών μηχανημάτων.

23) Πώς προέκυψε το plc

Τα προβλήματα βιομηχανικού ελέγχου πριν από τα PLCs αντιμετωπίζονταν κυρίως με τη χρήση ηλεκτρομηχανικών μεταγωγέων (relays – ρελέ)

Τα PLCs έχουν προγραμματιζόμενη λογική για να την αντιπαραβάλλουμε με την καλωδιωμένη λογική των προκατόχων

Τέτοιοι συνδυασμοί διακοπτών – αισθητήρων ενωμένοι σε ομάδες, ανάλογα με την επιθυμητή λειτουργία διατάσσονται σε επίπεδα και δίνουν το τελικό ηλεκτρικό κύκλωμα. Έτσι στην καλωδιωμένη λογική η λειτουργία του κάθε στοιχείου που ανήκει στο ίδιο επίπεδο γίνεται ταυτόχρονα (παράλληλα), ενώ η λειτουργία των επιπέδων γίνεται ακολουθιακά (σειριακά).

Με τα PLCs αντικαταστάθηκαν τα διάφορα επίπεδα με ένα και μοναδικό επίπεδο ακολουθιακής λογικής, το οποίο ονομάζουμε επεξεργαστή. Η λειτουργία του βασίζεται στη διαδοχική εκτέλεση εντολών οι οποίες βρίσκονται στη μνήμη του και τον οδηγούν στα ίδια συμπεράσματα με το αντίστοιχο ηλεκτρικό κύκλωμα.

24) Τι είναι και τι κάνει η CPU του PLC

Η κεντρική μονάδα επεξεργασίας ή CPU (Central Processing Unit) είναι το τμήμα του PLC που είναι υπεύθυνο για όλη την εφαρμογή της λογικής, την αποθήκευση και εκτέλεση του προγράμματος. Επίσης οργανώνει και τη ροή των πληροφοριών μέσα στο PLC. Δηλαδή:

- Διαβάζει τα δεδομένα από τις συσκευές ανίχνευσης (συνδεδεμένες στις εισόδους).
- Εκτελεί το αποθηκευμένο πρόγραμμα για να εκτιμήσει τα παραπάνω δεδομένα και να αποφασίσει για τη δράση.
- Στέλνει τις «αποφάσεις» (εντολές δράσης) του προγράμματος στις συσκευές ελέγχου (στις εξόδους)

Η CPU περιέχει τον μικροεπεξεργαστή, πιθανώς κάποια επιπλέον μνήμη και κυκλώματα επικοινωνίας με τα υπόλοιπα μέρη του PLC.

25) Τι είναι ο μικροεπεξεργαστής της CPU

Ο μικροεπεξεργαστής είναι είδος ημιαγωγού που περιέχει ενσωματωμένες τις λειτουργίες ενός υπολογιστή. Τις λειτουργίες τις πραγματοποιεί εκτελώντας ένα πρόγραμμα που ονομάζεται λειτουργικό σύστημα.

Οι βασικές λειτουργίες του μικροεπεξεργαστή είναι:

- Λειτουργίες εισόδων / εξόδων: Επικοινωνία του μικροεπεξεργαστή με τα υπόλοιπα μέρη της CPU.
- Αριθμητικές και λογικές πράξεις.
- Διαχείριση των περιεχομένων της μνήμης (δεδομένα και εντολές).

Η CPU περιέχει τον μικροεπεξεργαστή, όπως ήδη αναφέραμε, αλλά μπορεί να μην περιέχει μόνο έναν μικροεπεξεργαστή. Εφόσον υπάρχουν ιδιαίτερες απαιτήσεις στη χρήση του PLC μπορεί να χρησιμοποιηθεί κάποια συσκευή που ενσωματώνει δύο ή τρεις μικροεπεξεργαστές οι οποίοι επικοινωνούν και μεταξύ τους. Όταν υπάρχουν περισσότεροι από ένας μικροεπεξεργαστές ο κάθε ένας αναλαμβάνει να διαδραματίσει έναν διαφορετικό ρόλο.

26) Τι είναι η κεντρική μνήμη

Η κεντρική μνήμη χρησιμοποιείται για να διατηρεί όλες τις πληροφορίες που αφορούν το PLC. Έτσι στη μνήμη αυτή αποθηκεύονται: το λειτουργικό σύστημα, το πρόγραμμα, τις καταστάσεις των εισόδων και των εξόδων του PLC, πληροφορίες και ενδιάμεσα αποτελέσματα του προγράμματος, κ.α.

Η μνήμη είναι οργανωμένη σε bytes, που είναι η ελάχιστη πληροφορία που χρησιμοποιεί ο μικροεπεξεργαστής σε κάθε εντολή του. Το ίδιο το byte όμως αναλύεται σε 8 bits, τα οποία αποτελούν τα πιο βασικά στοιχεία πληροφορίας. Το κάθε bit μπορεί να έχει την τιμή 0 ή την τιμή 1. Υπάρχουν δύο μεγάλες κατηγορίες στις οποίες μπορεί να ανήκει κάποια μνήμη

Η πρώτη κατηγορία είναι αυτή των πτητικών (volatile) μνημών, οι οποίες διατηρούν τα περιεχόμενα τους όσο διατηρείται η τροφοδοσία τους με ρεύμα. Η δεύτερη κατηγορία είναι αυτή των μη πτητικών (non–volatile) μνημών που μπορούν και διατηρούν τα περιεχόμενα τους μόνιμα

27) Ποσα είδη κεντρικής μνήμης υπάρχουν

- Μνήμη τυχαίας προσπέλασης RAM (πτητική): Στη μνήμη αυτή διατηρούνται πληροφορίες του προγράμματος, ενίοτε και το ίδιο το πρόγραμμα.
- Μνήμη μόνο ανάγνωσης ROM (μη πτητική):Τα περιεχόμενα της δεν είναι δυνατόν να μεταβληθούν. Μόνο η ανάγνωση επιτρέπεται από τη μνήμη αυτή. Μπορεί να περιέχει το λειτουργικό σύστημα, αλλά όχι και το πρόγραμμα, αφού αυτό μπορεί να αλλάξει κάποια στιγμή.
- Προγραμματιζόμενη μνήμη μόνο ανάγνωσης (μη πτητική): Η μνήμη αυτή λειτουργεί όπως η μνήμη ROM, αλλά είναι δυνατόν να μεταβληθούν τα περιεχόμενα της, εάν τα διαγράψουμε πρώτα, φωτίζοντας τη μνήμη σε ειδικό σημείο με υπεριώδες φως (EPROM) ή επιβάλλοντας προκαθορισμένη τάση σε ειδικό ακροδέκτη της (EEPROM). Η δεύτερη κατηγορία προγραμματιζόμενης μνήμης είναι πιο ευέλικτη (απαιτεί πολύ μικρότερο χρόνο διαγραφής).

28) Τι είναι το τμήμα εισόδων/εξόδων

Αποτελείται από τη μονάδα διασύνδεσης εισόδων, η οποία επιτρέπει τη μετατροπή των πραγματικών σημάτων που λαμβάνει από τους αισθητήρες (που είναι συνδεδεμένοι στις εισόδους) σε σήματα χαμηλής ισχύος, κατάλληλα για τη CPU. Ταυτόχρονα προστατεύειτη CPU από το εξωτερικό περιβάλλον (δηλαδή από τις τάσεις και τα ρεύματα που έρχονται στις εισόδους) χρησιμοποιώντας την τεχνική της οπτικής απομόνωσης.

Το δεύτερο τμήμα είναι η μονάδα διασύνδεσης εξόδων, η οποία περιλαμβάνει ένα τμήμα αναλογικών και ένα τμήμα ψηφιακών εξόδων. Η πρώτη μας επιτρέπει να στέλνουμε τάσεις με αρκετά μεγάλη δυνατότητα διαβάθμισης της τιμής τους, ενώ οι έξοδοι της δεύτερης (ψηφιακής) μονάδας λειτουργούν περισσότερο σαν διακόπτες. Και στις εξόδους εφαρμόζεται η τεχνική της οπτικής απομόνωσης. Η επικοινωνία του τμήματος εισόδων / εξόδων με την CPU δεν είναι συνεχόμενη στον χρόνο.

Αντίθετα πραγματοποιείται κατά τακτά χρονικά διαστήματα Το χρονικό αυτό διάστημα ονομάζεται κύκλος σάρωσης.

Οι ψηφιακές είσοδοι και έξοδοι του PLC μπορούν να βρεθούν μόνο σε μία από δύο καταστάσεις. Μπορούν να είναι μόνο ενεργές (ON) ή ανενεργές (OFF), δηλαδή είναι δυνατόν να περιγραφούν από ένα μόνο bit. Το bit αυτό ονομάζεται bit κατάστασης. Οι έξοδοι αυτές μπορεί να οδηγούν δυσανάλογα μεγάλα φορτία και αυτό συμβαίνει αρκετά συχνά. Το PLC όμως δεν μπορεί να δώσει πολύ μεγάλα ρεύματα στις εξόδους του, έτσι χρησιμοποιούμε βοηθητικές τροφοδοσίες. Οι τιμές τάσης που μπορούμε συνήθως να συναντήσουμε στις ψηφιακές I/O είναι 5,12, 24, 48, 120, 220 βολτ συνεχούς ή εναλλασσόμενου ρεύματος. Οι αναλογικές είσοδοι και έξοδοι του PLC μπορούν να πάρουν τιμές από ένα εύρος διαθέσιμης τάσης. Το εύρος αυτό μπορεί να είναι 0–5 Vdc, 0–10 Vdc ή -10 – 10Vdc. Το πλήθος των διαφορετικών αναλογικών τάσεων που μπορεί να χειριστεί η κάθε μονάδα ταυτόχρονα (πλήθος καναλιών), διαφέρει από PLC σε PLC. Πάντως, κατά κανόνα, τα κανάλια των εισόδων είναι περισσότερα από αυτά των εξόδων.

29) Τι είναι το τερματικό

Τερματικό ονομάζεται η συσκευή που επιτρέπει στον χρήστη να αλληλεπιδρά με το PLC σχετικά με την κατάσταση του και το πρόγραμμα που περιέχει. Η σύνδεση γίνεται σχεδόν κατ’ αποκλειστικότητα με σειριακό καλώδιο (RS-232)

30) Πόσα είδη τερματικών υπάρχουν

- Απλό τερματικό (dumb terminal) – Χαμηλό κόστος

Δεν διαθέτει δική του CPU

Στέλνει χαρακτήρες απ’το πληκτρολόγιο.

Λαμβάνει χαρακτήρες που εμφανίζει στην οθόνη.

- Ειδικό βιομηχανικό τερματικό (Dedicated Industrial Terminal) – Δυσανάλογα υψηλό κόστος Λειτουργεί με PLC μιας εταιρείας (εξ’ού και ειδικό)

Παρέχει δυνατότητες προγραμματισμού μέσω οθόνης και πληκτρολογίου, με δικό του λογισμικό.

Λειτουργεί συνδεδεμένο (on-line) ή και ανεξάρτητα (off-line programming) από το PLC, ή και μέσω LAN.

Επιτρέπει την παρακολούθηση εκτέλεσης του προγράμματος και ανίχνευσης των λαθών του

- Προγραμματιστής χειρός (Handheld programmer)

Μικρά σε μέγεθος.

Λετουργούν ως απλά τερματικά.

Δυνατότητα αποθήκευσης προγράμματος.

Δυνατότητα αποθήκευσης προγράμματος.

Δυνατότητα ανίχνευσης σφαλμάτων.

- Η/Υ με λογισμικό για PLC:

Οικονομικότατη λύση, αφού οι Η/Υ υπάρχουν και για άλλους λόγους στον βιομηχανικό χώρο.

Παρέχουν ότι και οι προηγούμενες λύσεις.

Επιπλέον επιτρέπουν:

Λογισμικό προγραμματισμού.

Λογισμικό τεκμηρίωσης.

Λογισμικό συλλογής και ανάλυσης δεδομένων.

Λογισμικό διασύνδεσης λειτουργίας πραγματικού χρόνου.

Λογισμικό προσομοίωσης.

31) Τι είναι οι περιφερειακές συσκευές του plc

Οι περιφερειακές συσκευές είναι συσκευές που συνεργάζονται με ένα PLC αλλά δεν είναι απαραίτητες για τη λειτουργία του

32) Σε ποσα είδη διακρίνονται οι περιφερειακές συσκευές

- Φορτωτές προγράμματος.

- Μαζικές μνήμες (για αποθήκευση και ανάκτηση data).
- Μονάδες δισκετών.
- Μονάδες σκληρών δίσκων.
- □ Εκτυπωτές.
- □ Modems, αποτελούν ενδιάμεσους για άλλα περιφερειακά.

33) Τι είναι η διαδικασια προγραμματισμού του plc

Η διαδικασία του προγραμματισμού, αφορά τη συγγραφή οδηγιών με συγκεκριμένη σειρά (πρόγραμμα), σε κάποια γλώσσα που είναι κατανοητή στο PLC (γλώσσα προγραμματισμού) και τη μεταφορά του στο PLC.

34) Ποια τα απαιτούμενα δεδομένα για τον σχεδιασμό της λογικής ενός προγράμματος plc

Τα απαιτούμενα δεδομένα για το σχεδιασμό της λογικής ενός προγράμματος είναι:

- Η προς έλεγχο εγκατάσταση.

  Οι αισθητήρες που μας δίνουν τις πληροφορίες.

  Οι επενεργητέςπου επιτρέπουν την επιβολή αλλαγών.

- Το PLC που θα χρησιμοποιηθεί.

  Τις προδιαγραφές της συμπεριφοράς του συνδυασμένου συστήματος.

35) Τι είναι η δομή προγράμματος του plc

Κάθε πρόγραμμα γράφεται σε μία γλώσσα προγραμματισμού. Για τα PLCs υπάρχουν δύο βασικές γλώσσες: Η γλώσσα διαγραμ- μάτων κλίμακας (ladder diagram) και η γλώσσα λίστας εντολών (instruction list/STL).

36) Από τι αποτελείται η δομή του προγράμματος

Η δομή του προγράμματος αποτελείται από τρία βασικά τμήματα:

- Το κυρίως πρόγραμμα (υποχρεωτικό): Περιλαμβάνει την κυρίως λογική του προγράμματος. Από αυτό αρχίζει και η εκτέλεση του προγράμματος.
- Το τμήμα των υπορουτινών (προαιρετικό): Περιλαμβάνει αυτοτελή τμήματα κώδικα που υλοποιούν κάποια συγκεκριμένηλειτουργία Κάθε ένα από αυτά εκτελείται συγκεκριμένη λειτουργία. Κάθε ένα από αυτά εκτελείται παρενθετικά στη ροή του προγράμματος, κατ'εντολή του προγράμματος, προκειμένου να εκπληρώσει την εν λόγω λειτουργία.
- Το τμήμα των διακοπών (προαιρετικό): Όμοιο με το τμήμα υπορουτινών. Η εκτέλεση όμως, δεν γίνεται κατ'εντολή του προγράμματος, αλλά όταν συμβαίνει διακοπή (interrupt).

1.14 Simatic S7-300

1.14.1 Η σειρά PLC S7-300

Η σειρά SIMATIC S7300, περιλαμβάνει PLCπου χρησιμοποιούνται ευρέως σε μεγάλο φάσμα εφαρμογών. Διαθέτουν συμπαγή κατασκευή, αρθρωτή δομή και απευθύνονται σε εφαρμογές μεσαίας και μικρής κλίμακας. Η εύκολη και απλή υλοποίηση κατανεμημένων δομών, η ευπροσάρμοστη δικτύωση, η έλλειψη ανάγκη για ανεμιστήρα και ο φιλικός προς το χρήστη χειρισμός τα καθιστούν ως την πιο βέλτιστη και εύχρηστη λύση σε πολλά προβλήματα αυτοματισμού. Στην εικόνα φαίνεται ένα PLCτης σειράς S7-300.

Σχήμα 1.9: PLCτης σειράς S7-300

1.14.2 Χαρακτηριστικά S7-300

Μερικά από τα κυριότερα χαρακτηριστικά του S7-300 είναι:

- Αρθρωτή και συμπαγής μορφή για χρήση σε περιορισμένους χώρους
- Μεγάλη ποικιλία από CPUγια τη βέλτιστη επιλογή ανάλογα με την επιθυμητή απόδοση. Υπάρχουν 6 compact, 7 standard, 2 technologyκαι 5 fail-safe ενώ 9 μοντέλα διατίθενται για λειτουργία σε ακραίες θερμοκρασίες περιβάλλοντος (-25 ^{o}Cέως +60).
- Δικτυώνεται με όλα τα πρότυπα δίκτυα (Profibus, IndustrialEthernet)
- Μεγάλο εύρος εισόδων – εξόδων και καρτών επικοινωνίας (επέκταση έως 32 κάρτες και δικτύωση με όλα τα πρότυπα δίκτυα).
- Πλήρες 32 bitσετ εντολών και ευρεία συλλογή μπλοκ συναρτήσεων, έτοιμα για χρήση (ακόμα και για ημίτονο, συνημίτονο, λογάριθμο και τετραγωνική ρίζα).
- Δεν έχει περιορισμό για τη θέση των επιμέρους μονάδων.
- Δεν υπάρχουν μικροδιακόπτες (Dipswitches) για την παραμετροποίηση. Όλα γίνονται μέσω λογισμικού.
- Ενσωματωμένη δυνατότητα δικτύωσης MPI (Multi Pointinterface)
- Ενσωματωμένες δυνατότητες διασύνδεσης με HMI
- Μνήμη διαγνωστικών για αυτόματη αποθήκευση με χρόνο και ημερομηνία όλων των συμβάντων στο PLC.
- Μια μονάδα για τα αναλογικά. Η επιλογή γίνεται μέσω του λογισμικού

1.14.3 Δομή και μονάδες του S7-300

Οι μονάδες από τις οποίες μπορεί να αποτελείται το S7-300 είναι οι παρακάτω:

- **Τροφοδοτικό PS (PowerSupply) :** Μετατρέπει την τάση του δικτύου τροφοδοσίας στην κατάλληλη τάση λειτουργίας του PLC.
- **Κεντρική μονάδα επεξεργασίας (CPU):** εκτελεί λειτουργικό πρόγραμμα του PLC και το πρόγραμμα του χρήστη. Ελέγχει τις επικοινωνίες σε ένα MPIδίκτυο.
- **Μονάδες εισόδων – εξόδων (ψηφιακές ή αναλογικές) :** Προσαρμόζουν τα ηλεκτρικά σήματα από το εξωτερικό περιβάλλον προς την CPUκαι αντιστρόφως.

- **Μονάδες ειδικών λειτουργιών:**Μία ειδική μονάδα όπως λέει και το όνομά της, αναλαμβάνει να ελέγξει μία συγκεκριμένη λειτουργία αυτοματισμού. Έτσι μία λειτουργία που είναι επαναλαμβανόμενη και συνηθισμένη μπορεί να υλοποιηθεί (τόσο σε hardwareόσο και σε software) από μία ειδική μονάδα.
- **Επεξεργαστής επικοινωνίας CP** (Communication Processor)
- **Κάρτα διασύνδεσης** τωνRack Interface Module IM.
- **Καλώδιο Profibus δικτύου με τους Bus Connector :** Συνδέει μεταξύ τους κόμβους ενός MPI ή Profibus δικτύου.
- **Το πλαίσιο στήριξης (Rack) :** Ο ρόλος του είναι απλά να στηρίζει τις διάφορεςς κάρτες που θα συνθέσουν το σύστημα αυτοματισμού.

1.14.4 Μνήμη του S7-300

Η μνήμη, είναι μία από τις βασικότερες συνιστώσες σ' ένα σύστημα αυτοματισμού. Η μνήμη μίας CPU χωρίζεται σε πέντε κατηγορίες:

i. **Μνήμη φόρτωσης**(Load Memory)

Στη μνήμη αυτή αποθηκεύεται το πρόγραμμα του χρήστη, η περιγραφή του συστήματος καθώς και οι παράμετροι των μονάδων. Στα εν εξελίξει μοντέλα σε αυτήν την περιοχή μνήμης θα αποθηκεύονται και τα χρησιμοποιούμενα σύμβολα και ο σχολιασμός του προγράμματος, κάτι που μέχρι τώρα γίνεται στη συσκευή προγραμματισμού. Η μνήμη φόρτωσης προϋπάρχει σε κάθε CPUκαι μπορεί να επεκταθεί με εξωτερικές μνήμες RAMή FlashEPROM.

ii. **Μνήμη εργασίας** (Work Memory)

Η μνήμη εργασίας είναι ενσωματωμένη στη CPUκαι δε γίνεται να επεκταθεί. Είναι RAMυψηλής απόδοσης (ταχύτητας) και διατηρεί το περιεχόμενό της μόνο με τη βοήθεια μπαταρίας. Αυτή επιτρέπει την επεξεργασία του κώδικα και τα μπλοκ δεδομένων του χρήστη. Η μεταφορά των απαραίτητων στοιχείων από την εξωτερική μνήμη στη μνήμη εργασίας γίνεται από το λειτουργικό σύστημα της CPU.

Στην περίπτωση, που το πρόγραμμα μεταφέρεται από συσκευή προγραμματισμού, τότε αυτό καταχωρείται στη μνήμη φόρτωσης και εργασίας ταυτόχρονα. Η μνήμη αυτή

προϋπάρχει σε κάθε CPUστη μέγιστη τιμή της, πράγμα που σημαίνει ότι δεν επεκτείνεται με εξωτερικές μνήμες.

iii. **Μνήμη συστήματος** (System Memory)

Η μνήμη του συστήματος είναι ενσωματωμένη στη κεντρική μονάδα και δεν μπορεί να επεκταθεί. Σε αυτήν περιέχονται ομαδοποιημένες οι μεταβλητές που χρησιμοποιούμε στο πρόγραμμα μας. Η κάθε ομάδα των μεταβλητών αυτών καταλαμβάνει ένα ορισμένο χώρο το μέγεθος του οποίου εξαρτάται από τη CPUπου χρησιμοποιούμε. Οι περιοχές (ομάδες) που χωρίζεται η μνήμη συστήματος είναι:

- **Μνήμη απεικόνισης εισόδων PII**

Σε αυτή την περιοχή, αποθηκεύονται οι τιμές των εισόδων που διαβάζει η CPU από τις κάρτες εισόδου στην αρχή κάθε κύκλου λειτουργίας.

- **Μνήμη απεικόνισης εξόδων PIQ**

Σε αυτή την περιοχή, αποθηκεύεται η τιμή κάθε μιας από τις χρησιμοποιούμενες εξόδους κατά τη χρονική περίοδο του κύκλου λειτουργίας κατά την οποία εκτελείται το πρόγραμμα του χρήστη. Αυτή η περιοχή μνήμης στο τέλος του κύκλου στέλνεται για να ενημερώσει τις κάρτες εξόδου.

- **Βοηθητικά M**

Σε αυτήν την περιοχή της μνήμης αποθηκεύονται ενδιάμεσα αποτελέσματα τα οποία έχουν υπολογιστεί κατά την εκτέλεση του προγράμματος. Τα αποτελέσματα αυτά μπορεί να είναι μεγέθους ενός bitπ.χ M0.0 ή μεγέθους wordπ.χ MW4 ή doubleword. Επειδή ο χώρος της μνήμης είναι κοινός για όλα αυτά θα πρέπει πάντα η διευθυνσιοδότητση να γίνεται με προσοχή ώστε να μην έχουν επικάλυψη θέσεων μνήμης.

- **Χρονικά T**(Timers)

Είναι η περιοχή της μνήμης του συστήματος όπου αποθηκεύονται οι χρόνοι των χρονικών που χρησιμοποιούμε

- **Απαριθμητές C** (Counters)

Είναι η περιοχή της μνήμης του συστήματος όπου αποθηκεύονται τα περιεχόμενα των απαριθμητών

- **Τοπικά βοηθητικά L** (LocalData)

Είναι η περιοχή της μνήμης του συστήματος, όπου αποθηκεύονται προσωρινά δεδομένα ενός μπλοκ που περιέχει κώδικα (πχ ενός OB, FB, FC). Τα τοπικά βοηθητικά έχουν ισχύ για όσο τρέχει το συγκεκριμένο μπλοκ το οποίο τα περιέχει.

- **Διαγνωστικά**

Καταχωρούνται διάφορες ενέργειες που έχουν γίνει στο σύστημα με ώρα και ημερομηνίες όπως CPUσε RUN/STOP, βραχυκυκλωμένη κάρτα αναλογικών, κλπ.

iv. Διατηρούμενη μνήμη

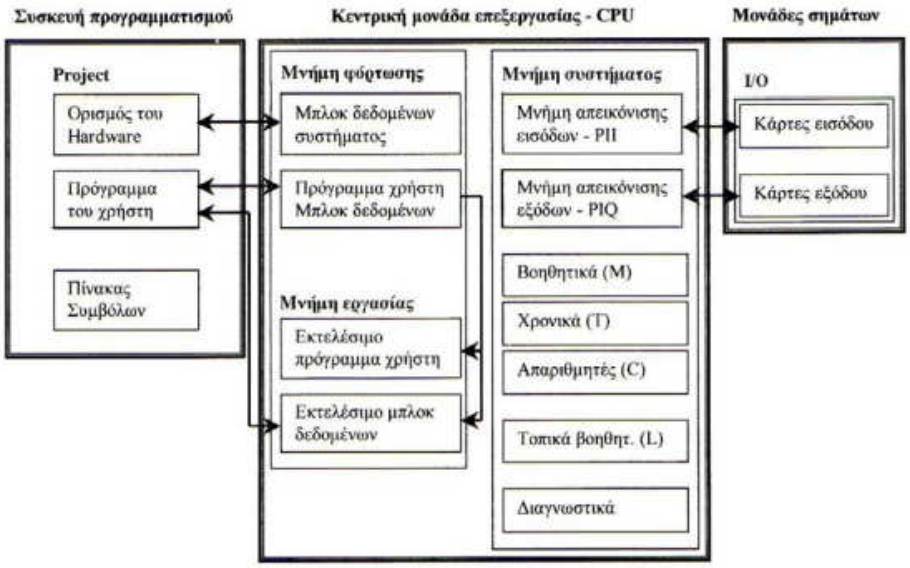
Διατηρούμενη μνήμη, είναι η μνήμη η οποία είτε η τροφοδοσία κλείσει είτε μετάβαση από STOP σε RUN γίνει αυτή θα διατηρηθεί. Τέτοιες περιοχές είναι αυτές των βοηθητικών, των χρονικών, των απαριθμητών και των περιοχών με δεδομένα. Η δήλωση των περιοχών που θέλουμε να διατηρηθούν γίνεται με τη βοήθεια του προγράμματος παραμετροποίησης και προγραμματισμού STEP 7. Οι περιοχές που δηλώνουμε, αποθηκεύονται στη μνήμη φόρτωσης.

v. Μνήμη ρολογιού

Πολλές διαδικασίες στο PLC, χρειάζονται περιοδικό σήμα. Αυτό μπορούμε να το παράγουμε με πολλούς τρόπους (π.χ χρονικά) αλλά ο πιο ενδεδειγμένος κι ευκολότερος είναι η χρήση της μνήμης του ρολογιού. Η μνήμη αυτή έχει μήκος 8 bitπου το καθένα αναβοσβήνει περιοδικά με μια συγκεκριμένη συχνότητα.

7	6	5	4	3	2	1	0
0.5 Hz	0.625 Hz	1 Hz	1.25 Hz	2 Hz	2.5 Hz	10 Hz	10 Hz

Αυτό που κάνουμε εμείς όταν ορίζουμε το υλικό, στις παραμέτρους της CPU, να δηλώνουμε ένα Byteβοηθητικών το οποίο ενημερώνεται από το λειτουργικό σύστημα της CPU. Αν για παράδειγμα έχουμε δηλώσει ένα byte βοηθητικών MB 100, τότε με συχνότητα 2 Hz θα πάλλεται το Bit M 100.3.



Σχήμα 1.10: Κύριες μνήμες στη CPUτου PLC

1.14.5 Προγραμματισμός του S7-300

Ο προγραμματισμός γίνεται με διάφορες συσκευές προγραμματισμού που έχει κατασκευάσει η SIEMENS γι' αυτό το σκοπό ή με τη χρήση ηλεκτρονικού υπολογιστή.

Συσκευές προγραμματισμού Simatic S7-300

Η συσκευή PG 702 είναι φορητή και μπορεί να προγραμματίσει το PLCμόνο σε γλώσσα λίστα εντολών με Boolean εντολές και γι' αυτό χρησιμοποιεί τη γλώσσα StatementList. Τα μοντέλα PG 720, PG 740 είναι σαν φορητοί προσωπικοί υπολογιστές. Έχουν προεγκατεστημένο το πρόγραμμα STEP 7 και μπορούν να προγραμματίσουν το PLCκαι στις τρεις γλώσσες που αυτό δέχεται.

Προγραμματισμός με προσωπικό υπολογιστή

Ο προγραμματισμός με προσωπικό υπολογιστή είναι ο πιο εύκολος. Αν χρησιμοποιηθεί προσωπικός υπολογιστής για τον προγραμματισμό του PLC, θα πρέπει ο χρήστης να εγκαταστήσει το πρόγραμμα STEP 7 που συνοδεύει το PLC. Ο προγραμματισμός μπορεί να γίνει και στις τρεις γλώσσες με τις οποίες προγραμματίζεται το PLCS7 300.

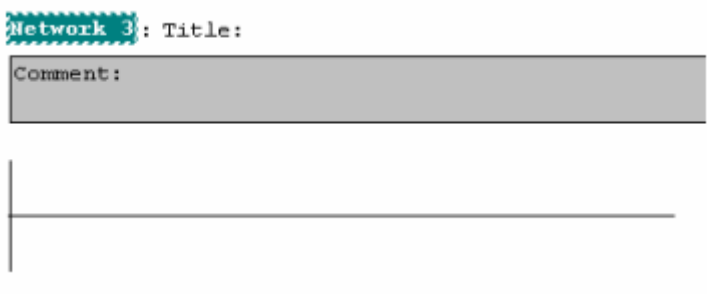
Επίσης ο υπολογιστής μπορεί να συνδεθεί και με τη συσκευή προγραμματισμού EEPROMστην περίπτωση που θέλετε το πρόγραμμά σας να το φορτώσετε σε μονάδα μνήμης EEPROM. Επιπλέον ο υπολογιστής παρέχει τη δυνατότητα αρχειοθέτησης των προγραμμάτων του χρήστη καθώς και την εκτύπωση αυτών.

1.15 Γλώσσες προγραμματισμού PLC

Υπάρχουν τρεις τυποποιημένες μορφές προγραμματισμού που έχουν επικρατήσει:

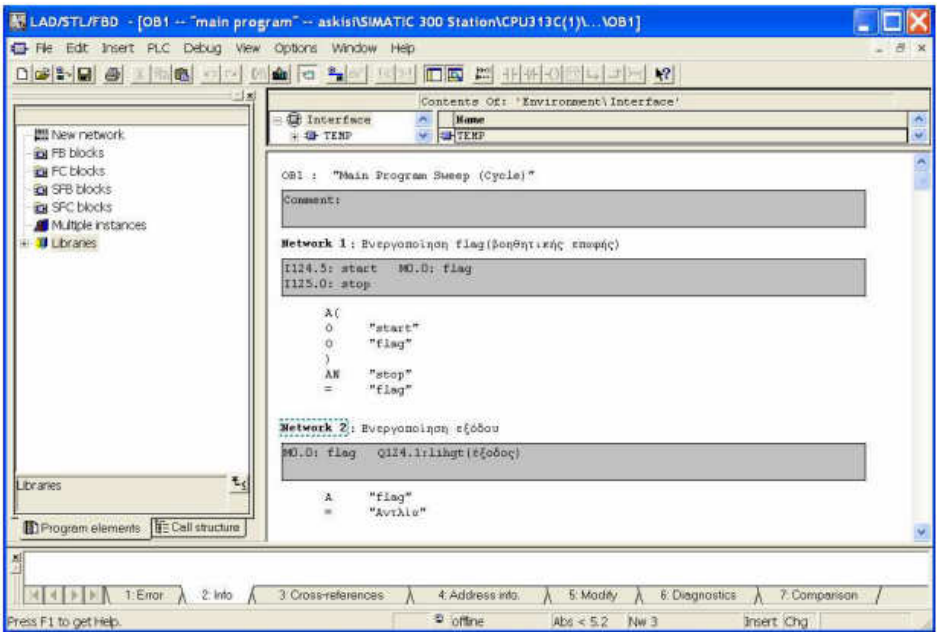
- Σχέδιο επαφών (LAD – Ladder Diagram)
- Λίστα εντολών (STL – Statement List) και
- Διάγραμμα λογικών πυλών (FBD – Function Block Diagram).

Η **LAD**, είναι η πρώτη γλώσσα προγραμματισμού, όπου η σύνταξη των εντολών μοιάζει με το διάγραμμα κυκλώματος κλασικού αυτοματισμού και επιτρέπει να παρακολουθείται εύκολα η ροή του σήματος από τις επαφές και τα πηνία. Τα στοιχεία αυτά επιλέγονται και τοποθετούνται στον LAD/STL/FBDEditor από το ειδικό παράθυρο επιλογής στοιχείων. Στην Εικόνα φαίνεται το περιβάλλον της γλώσσας Ladder.



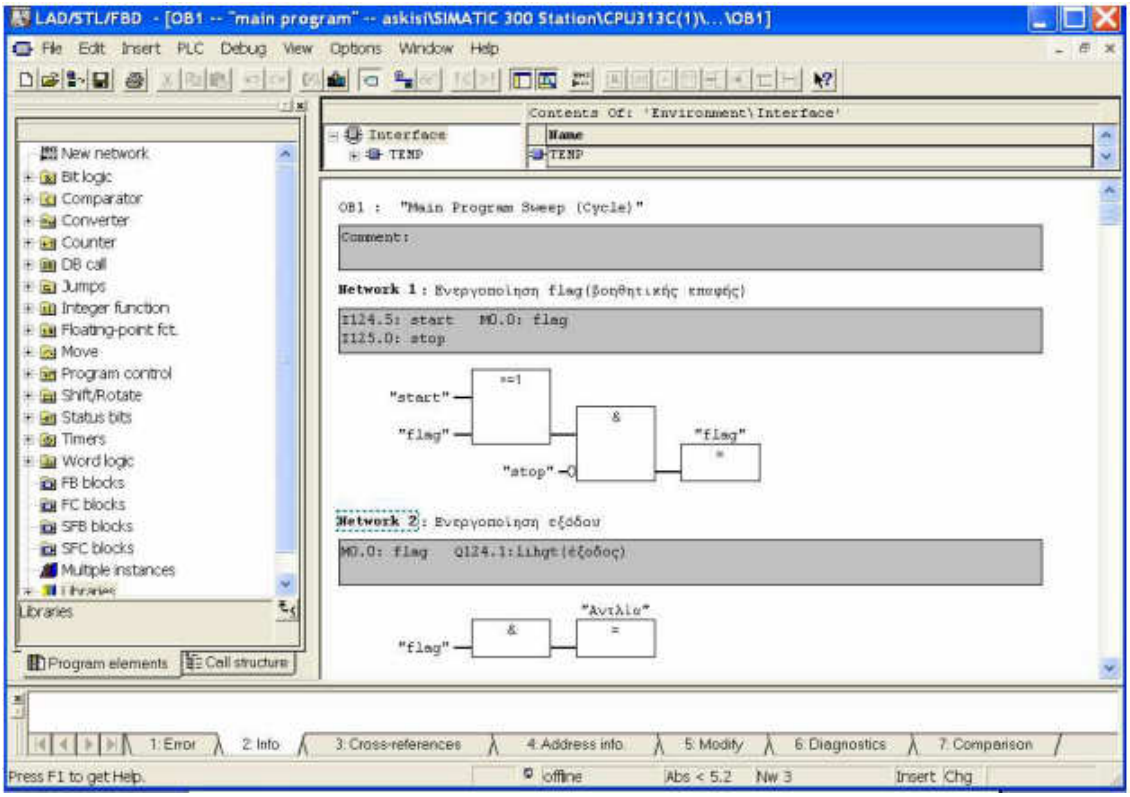
Σχήμα 1.11: Αρχικό παράθυρο για προγραμματισμό σε Ladder

Η **STL** είναι η δεύτερη γλώσσα προγραμματισμού, η οποία αναπτύχθηκε σχεδόν ταυτόχρονα με τη Ladder. Η γλώσσα αυτή, δημιουργεί λίστα προγράμματος με εντολές, που αντιστοιχούν στις λογικές πύλες (AND, OR, NOT κ.τ.λ.). Στην αρχή η γλώσσα αυτή ήταν πολύ φτωχή και περιοριζόταν μόνο στις βασικές Booleanεντολές. Στη συνέχεια, οι γλώσσες αυτές αναπτύχθηκαν πολύ και συναντά κανείς σε αυτές στοιχεία από τις γλώσσες των υπολογιστών και κυρίως των γλωσσών Assembly. Ο προγραμματισμός σε αυτή τη γλώσσα, απαιτεί από το χρήστη να έχει στοιχειώδεις γνώσεις προγραμματισμού. Στην ακόλουθη Εικόνα φαίνεται το περιβάλλον της γλώσσας για το ίδιο πρόγραμμα.



Σχήμα 1.12: Προγραμματισμός σε STL

Η **FBD**, είναι η τρίτη γλώσσα προγραμματισμού με γραφικά. Οι εντολές αναπαρίστανται με λογικά «κουτιά», παρόμοια με αυτά που συναντάμε στην άλγεβρα Bool και επιτρέπεται να παρακολουθούμε τη ροή του σήματος ανάμεσα στα κουτιά. Τα στοιχεία αυτά επιλέγονται και τοποθετούνται στον LAD/STL/FBD Editor από το ειδικό παράθυρο επιλογής στοιχείων. Στο παρακάτω Σχήμα 1.13 φαίνεται το περιβάλλον της Function Block Diagram.



Σχήμα 1.13: Προγραμματισμός σε FBD

Και οι τρεις αυτές μορφές υπάρχουν ενσωματωμένες στο πακέτο προγραμματισμού STEP 7. Η επιλογή τους είναι ελεύθερη και μπορεί να γίνει οποιοσδήποτε συνδυασμός στα όρια ενός project. Ακόμα, υπάρχει η δυνατότητα μετατροπής ενός μπλοκ από μια μορφή απεικόνισης σε μια άλλη. Αυτό είναι πάντα δυνατό από LAD ή FBD σε STLενώ δεν ισχύει πάντοτε το αντίθετο, αφού στη λίστα εντολών μπορούν να προγραμματιστούν πράγματα που είναι αδύνατο να απεικονιστούν σε γραφική μορφή.

1.15.1 Προγραμματισμός σε γλώσσα Ladder (LAD)

Η LADείναι μία γλώσσα προγραμματισμού που χρησιμοποιείται στο βιομηχανικό έλεγχο. Η γλώσσα αυτή χρησιμοποιεί γραφικά σύμβολα παρόμοια με τα σχηματικά διαγράμματα κυκλωμάτων ρελέ.

1.15.2 Βασικά στοιχεία

Τα βασικά στοιχεία που χρησιμοποιούνται για να αναπαραστήσουν τη λογική ενός προγράμματος είναι τα ακόλουθα:

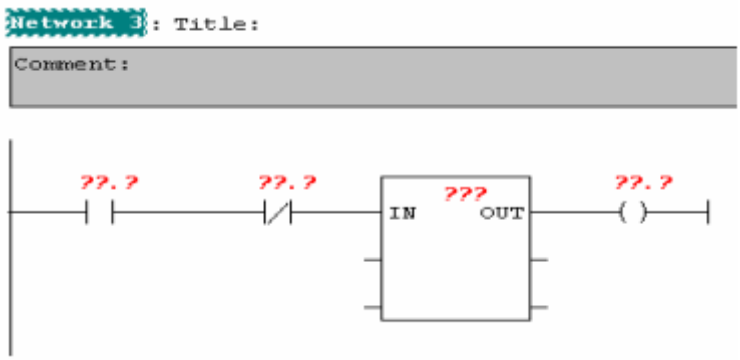
 εισαγωγή network.

 Οι επαφές αναπαριστούν επαφές διακοπτών μέσα από τις οποίες περνάει ρεύμα. Οι επαφές μπορεί να είναι είτε κανονικά ανοικτές είτε κανονικά κλειστές.

 Τα πηνία αναπαριστούν ρελέ τα οποία οπλίζονται όταν τροφοδοτηθούν με τάση.

 Τα κουτιά αναπαραστούν συνήθως χρονικά (timers), απαριθμητές (counter) και μαθηματικές πράξεις. Τα κουτιά εκτελούν τις λειτουργίες τους όταν τροφοδοτηθούν με τάση.

Παράδειγμα



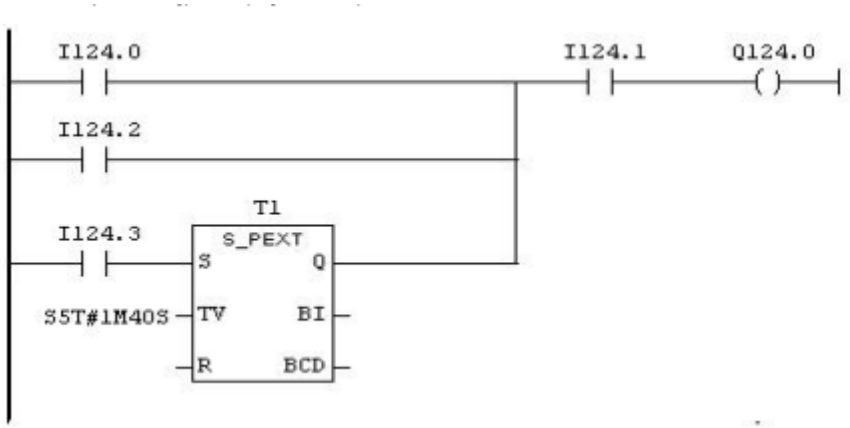
Σχήμα 1.14: Παράδειγμα

Τι είναι ένα δικτύωμα (Network) στη γλώσσα Ladder

Δικτύωμα στη γλώσσα Laddereίναι ένα σύνολο συνδεδεμένων μεταξύ τους στοιχείων που συνιστούν ένα ολοκληρωμένο κύκλωμα μεταξύ της γραμμής τροφοδοσίας που βρίσκεται αριστερά και του στοιχείου εξόδου που βρίσκεται δεξιά.

- Η γραμμή τροφοδοσίας που βρίσκεται στο αριστερό μέρος αναπαριστά τον αγωγό φάσης.

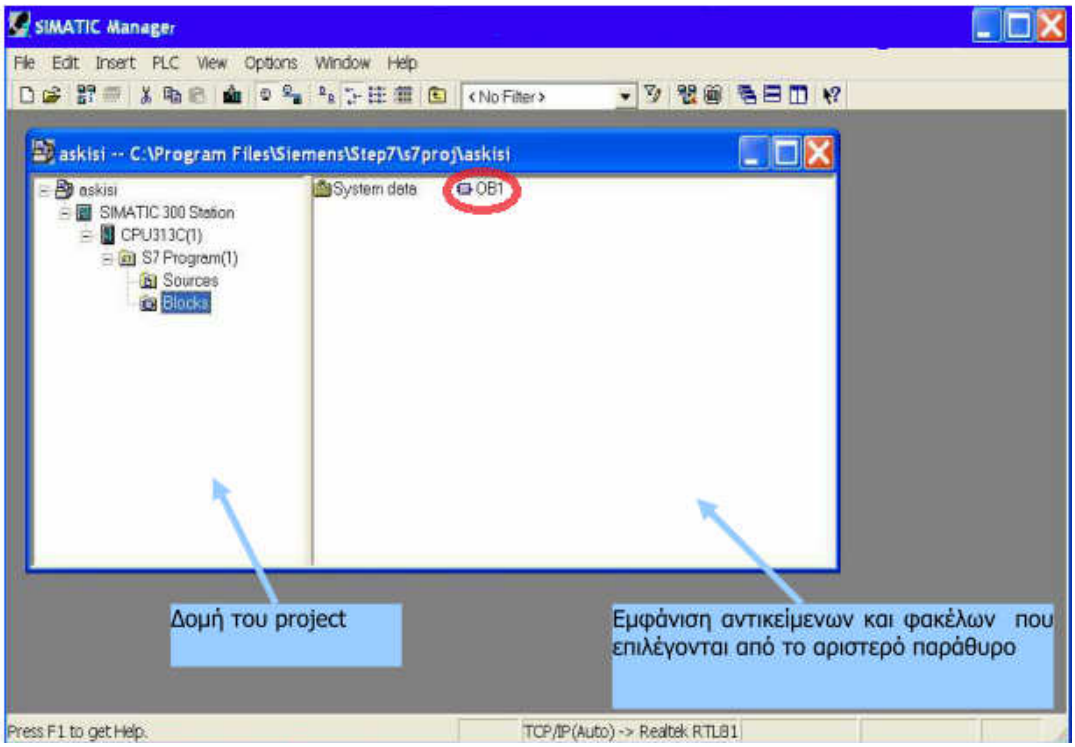
- Το στοιχείο εξόδου που βρίσκεται δεξιά είναι συνδεδεμένο με τον ουδέτερο ο οποίος δεν φαίνεται.
- Στο δικτύωμα, υποτίθεται ότι έχουμε ροή ρεύματος από αριστερά προς τα δεξιά μέσα από τις ενεργοποιημένες (κλειστές) επαφές στα πηνία και στα κουτιά.



Σχήμα 1.15: Παράδειγμα δικτυώματος

1.15.3 Περιβάλλον γλώσσας Ladder

Η γλώσσα που έχει επιλεχθεί σαν γλώσσα προγραμματισμού του PLC είναιη Ladder. Στη συνέχεια λοιπόν, θα αναφέρουμε πως μπορούμε να εισέλθουμε στο περιβάλλον της γλώσσας αυτής, με σκοπό να προγραμματίσουμε όλα τα blocks που μπορεί να χρειαστούμε. Ο πρώτος τρόπος είναι να κάνουμε διπλό κλικ στο εικονίδιο OB1 που βρίσκεται στο Object Blocksστο παράθυρο του Simatic Manager. Ο δεύτερος τρόπος, είναι επιλέγοντας το εικονίδιο που βρίσκεται κάτω από το μενού Έναρξη >Simatic>Step7 >LAD,STL,FBD-ProgrammingS7 Blocks.



Σχήμα 1.16: Εισαγωγή σε περιβάλλον Ladder

Κάνοντας λοιπόν διπλό κλικ, στο εικονίδιο OB1 του παραπάνω σχήματος, μας εμφανίζεται το παράθυρο της Ladder.

1.15.4 Βασικές εντολές Ladder

Οι βασικότερες εντολές της γλώσσας αυτής είναι οι παρακάτω:

- Εντολές λογικής Bit
- Σύγκρισης
- Μετατροπής
- Απαριθμητών
- Διακλάδωσης
- Ολίσθησης και περιστροφής
- Κατάστασης
- Χρονιστών
- Λογικών πράξεων μεταξύ λέξεων
- Μετακίνησης

Εντολές λογικής Bit

Αρχικά θα παρουσιάσουμε τις βασικές εντολές που βασίζονται στη Booleanλογική.

 Ανοιχτή επαφή κλείνει όταν η τιμή που αποθηκεύεται στη συγκεκριμένη διεύθυνση της είναι «1».

 Κλειστή επαφή κλείνει όταν η τιμή που αποθηκεύεται στη συγκεκριμένη διεύθυνση της είναι «0».

 Πηνίο εξόδου. Θα δώσει αποτέλεσμα «1» (τροφοδότηση με τάση), όταν η συνθήκη που προηγείται του πηνίου είναι αληθής και «0», όταν είναι ψευδής.

---|NOT|--- Αναστροφέας. Η εντολή αυτή αντιστρέφει το σήμα που φτάνει σ' αυτήν. Έτσι αν έρθει «1» θα το κάνει «0» και αν έρθει «0» θα το κάνει «1».

Εντολές σύγκρισης

- CMPxx1 Compare Integer (>, <, ==, <>, <=, >=): Με την εντολή αυτή γίνεται οποιαδήποτε σύγκριση μεταξύ δύο ακέραιων αριθμών (IN1 και IN2) που έχουν αποθηκευτεί σε δύο θέσεις μνήμης. Η έξοδος της εντολής αυτής θα είναι «1», αν η σύγκριση είναι αληθής. Στη θέση xx1, εισάγουμε το είδος της σύγκρισης που θέλουμε να γίνει. Οι δύο ακέραιοι μπορούν να είναι οποιοσδήποτε 16 bit αριθμός.

Εντολές μετατροπής

Σε αυτή την ενότητα, θα παρουσιαστούν οι σημαντικότερες εντολές μετατροπής από έναν αριθμό σε κάποιον άλλο.

- **BCD_IBCD to INTEGER:**μετατρέπει έναν BCD αριθμό σε 16 bit ακέραιο.
- **I_DINT Integer to Double Integer:** μετατρέπει ένα 16 bit ακέραιο σε 32 bit ακέραιο.
- **TRUNC Truncate Double Integer Part:**μετατρέπει ένα πραγματικό αριθμό σε ακέραιο με στρογγυλοποίηση προς τα κάτω.
- **ROUND Round to Double Ineteger:** μετατρέπει ένα πραγματικό αριθμό σε ακέραιο με κανονική στρογγυλοποίηση.
- **CEIL Ceiling:** μετατρέπει ένα πραγματικό αριθμό, σε ακέραιο με προς τα πάνω στρογγυλοποίηση.

- **INV_I Ones Complement Integer:** εκτελώντας μία πράξη XORμε το FFFFHex, μετατρέπει έναν 16 bit ακέραιο στο συμπλήρωμα ως προς 1.
- **NEG_IT wos Complement Integer:** μετατρέπει έναν 16 bit ακέραιο στο συμπλήρωμα ως προς 2.
- **NEG_R Negate Floating Point Number:** μετατρέπει έναν πραγματικό αριθμό στον αντίθετό του.

Εντολές απαριθμητών

Ένα βασικό σετ εντολών, είναι οι εντολές μετρητών – απαριθμητών που μπορούν να μετρήσουν κάποιο γεγονός που συνέβηκε από 0 ως και 999 φορές. Υπάρχουν 5 είδη μετρητών και παρουσιάζονται παρακάτω :

- **S_CUD UP DOWN Counter:** μόλις φτάσει 1 στην ακίδαS (Set), ο μετρητής φορτώνει την τιμή που έχει τεθεί από πριν στην ακίδα PVκαι θα αυξηθεί κατά 1 αν η κατάσταση στην ακίδα CU (Counter Up) αλλάξει από 0 σε 1. Ομοίως θα μειωθεί κατά 1, αν αλλάξει η κατάσταση στην ακίδα CD (Counter Down). Ο μετρητής θα δώσει 1 στην έξοδο Qόταν η τρέχουσα τιμή του είναι μεγαλύτερη από 0.
- **S_CU UP Counter:**είναι παρόμοιος με τον προηγούμενο, με την διαφορά ότι μετράει μόνο προς τα πάνω, δηλαδή μόνο αυξάνει όταν παρουσιαστεί αλλαγή από 0 σε 1 στην ακίδα CU.
- **S_CD DOWN Counter:** ομοίως και αυτός λειτουργεί ανάλογα προς τα κάτω. Μειώνεται μόνο όταν παρουσιαστεί αλλαγή από 0 σε 1 στην ακίδα CD.
- **SCSet Counter Value:** όταν φτάσει 1, τότε η τιμή που έχουμε ορίσει μεταφέρεται στον μετρητή που αναφέρεται ως Cno. π.χ C5.
- **CUUP Counter Coil:** ο μετρητής αυτός θα αυξάνεται κατά 1, κάθε φορά που το αποτέλεσμα της λογικής πράξης που έγινε πριν τον μετρητή είναι 1. Αυτό σημαίνει ότι ο μετρητής θα αρχίσει να αυξάνεται από την τιμή που έχει οριστεί και θα φτάνει μόνο μέχρι το 999.

Εντολές διακλάδωσης

- **Jump:**έχει δύο εφαρμογές, το άλμα υπό συνθήκη και το άλμα χωρίς συνθήκη. Στην πρώτη, το άλμα θα γίνει αν το αποτέλεσμα της λογικής πράξης πριν την εντολή άλματος είναι 1. Στην δεύτερη, δεν πρέπει να παρεμβάλλεται τίποτα μεταξύ της

εντολής και της κάθετης γραμμής που είναι η γραμμή ρεύματος. Ως αποτέλεσμα, αυτή η εντολή θα εκτελεστεί ότι και να γίνει.

- **Jump if not:** εκτελείται το άλμα αν το αποτέλεσμα της λογικής πράξης πριν την εντολή είναι 0.
- **Label:** με την εντολή αυτή, δηλώνεται το όνομα της διεύθυνσης στην οποία θα πηδήξει το πρόγραμμα μετά από μία εντολή άλματος.

Εντολές ολίσθησης και περιστροφής

- **SHR_I Shift Right Integer**: η εντολή αυτή, κάνει ολίσθηση δεξιά έναν 16 bitακέραιο. Κατά την εκτέλεση της εντολής, δημιουργούνται κενές θέσεις οι οποίες γεμίζουν με το bit του πρόσημου, δηλαδή του 15 bit. Όσα bit ολισθήσουν χάνονται.
  - SHR_W Shift Right Word: για δεξιά ολίσθηση 16 bit λέξης.
  - SHR_DW Shift Right Double Word: για δεξιά ολίσθηση 32 bit λέξης.
  - SHL_W Shift Left Word: για αριστερή ολίσθηση 16 bitλέξης.
- **ROR_DW Rotate Right Double Word:** κάνει περιστροφή δεξιά 32 bitλέξης, που βρίσκεται στον ακροδέκτη INτόσες φορές όσες λέει ο ακροδέκτης N. Το αποτέλεσμα αποθηκεύεται στον ακροδέκτη OUT.
  - ROL_DW Rotate Left Double Word: κάνει αριστερή περιστροφή μιας 32 bit λέξης.

Εντολές κατάστασης

- **Result bit equal 0:** η εντολή αυτή χρησιμοποιείται για να ελέγξουμε αν το αποτέλεσμα μίας μαθηματικής πράξης είναι ίσο με το 0.
- **Result bit no tequal 0:** χρησιμοποιείται για να ελέγξουμε αν το αποτέλεσμα μιας διαφορετικής πράξης είναι διαφορετικό του 0.
- **Result bit greater than 0:** χρησιμοποιείται για να ελέγξουμε αν το αποτέλεσμα μίας μαθηματικής πράξης είναι μεγαλύτερο του 0.
- **Result bit less than 0:** χρησιμοποιείται για να ελέγξουμε αν το αποτέλεσμα μίας μαθηματικής πράξης είναι μικρότερο του 0.
- **Result bit greater equal 0:** χρησιμοποιείται για να ελέγξουμε αν το αποτέλεσμα μίας μαθηματικής πράξης είναι μεγαλύτερο ή ίσο του 0.

Εντολές χρονιστών

Οι χρονιστές μας δίνουν την δυνατότητα, να εισάγουμε μία συγκεκριμένη καθυστέρηση πριν την εκτέλεση κάποιας εντολής ή ακόμα μπορούμε να μετρήσουμε το χρόνο που διήρκεσε ένα γεγονός. Τα χρονικά μπορούν να μετρήσουν χρόνους μέχρι και 9990 δευτερόλεπτα. Ο μικρότερος χρόνος που μπορεί να μετρηθεί είναι 10 msec. Οι εντολές που χρησιμοποιούμε σε αυτή την περίπτωση είναι οι παρακάτω:

- **S_Pulse:** το χρονικό τρέχει όσο το σήμα στην ακίδα Sείναι 1. Αν γίνει 0, τότε το χρονικό θα σταματήσει και θα βγάλει στην έξοδο Q 0.
- **S_Pext:** η έξοδος του χρονικού παραμένει 1 για όλο τον χρόνο που έχει προγραμματιστεί, ακόμα κι αν το σήμα στην ακίδα Sέχει γίνει 0.
- **S_ODT:** η έξοδος θα γίνει 1, μόνο όταν περάσει ο χρόνος που έχουμε θέσει στο χρονικό και το σήμα στην είσοδο είναι ακόμα 1.
- **S_ODTS:** η έξοδος θα γίνει 1, μόνο όταν περάσει ο χρόνος που έχουμε θέσει, ακόμα και αν το σήμα στην είσοδο έχει γίνει 0.
- **S_OFFDT:** αυτό το χρονικό φορτώνει τον προκαθορισμένο χρόνο όταν έρθει 1 στην ακίδα S και θέλει αρνητικό παλμό για να ξεκινήσει. Η έξοδος του θα μείνει άσσος για όσο χρόνο θα μετράει το χρονικό και αν ξαναέρθει 1 στην είσοδο θα γίνει Reset.

Εντολές λογικών πράξεων μεταξύ λέξεων

- **WAND_W (Word) and Word:** με την εντολή αυτή γίνεται η λογική πράξη ANDμεταξύ των λέξεων που βρίσκονται στους ακροδέκτες IN1 και IN2. Το αποτέλεσμα αποθηκεύεται στον ακροδέκτη OUT. Αν η πράξη εκτελεστεί ο ακροδέκτης ENOθα δώσει 1.
- **WOR_W (Word) or Word:** είναι ίδια με την παραπάνω, με τη διαφορά ότι εδώ η λογική πράξη είναι η ORκαι όχι η AND.
- **WXOR_W (Word) Exclusive OR Word:** η εντολή αυτή εκτελεί τη λογική πράξη XORμεταξύ δύο αριθμών.

ΚΕΦΑΛΑΙΟ 2

ΠΡΟΣΟΜΟΙΩΤΗΣ PLC - HYDRAN

2.1 Περιγραφή του προγράμματος προσομοίωσης HYDRA

Ανοίγουμε τον υπολογιστή και περιμένουμε να φορτωθεί το DOS. Στη συνέχεια πληκτρολογούμε cd hydran και πατάμε ENTER. Στην οθόνη εμφανίζεται : C:\HYDRAN> και πληκτρολογούμε hydran και ENTER οπότε φορτώνεται το πρόγραμμα.

Στο κάτω μέρος της οθόνης εμφανίζεται το βασικό menu το οποίο μας δίνει τις εξής δυνατότητες :

Editor Compiler Filer Simulator Exit

Η επιλογή γίνεται με τα βελάκια ← → και ENTER.

EDITOR

Με την επιλογή της λειτουργίας Edit από το κυρίως menu, πηγαίνουμε στον επεξεργαστή κειμένου, αφού πρώτα μας ζητηθεί το όνομα του αρχείου της λίστας εντολών του προγράμματος εφαρμογής, στο οποίο θέλουμε να δουλέψουμε. Αν υπάρχει, τότε φορτώνεται από τον σκληρό δίσκο και η γραφίδα τοποθετείται στο τέλος του προγράμματος εφαρμογής, διαφορετικά αν είναι ένα νέο πρόγραμμα, η γραφίδα τοποθετείται στην αρχή της πρώτης κενής σειράς.

Ο επεξεργαστής κειμένου φροντίζει για την αυτόματη διασπορά διαστημάτων ανάμεσα στα επιμέρους τμήματα κάθε οδηγίας, όταν πατηθεί το πλήκτρο διαστήματος (space bar).

COMPILER

Ο Compiler μεταφράζει το κείμενο του προγράμματος εφαρμογής στη γλώσσα μηχανής του PS-14N.

Ταυτόχρονα ψάχνει και για συντακτικά λάθη στο πρόγραμμα εφαρμογής. Σε περίπτωση που διαπιστώθηκε σφάλμα, αναγράφεται στη γραμμή εισαγωγής ο αριθμός της σειράς

όπου βρίσκεται το λάθος και με το πάτημα οποιουδήποτε πλήκτρου πηγαίνουμε στον επεξεργαστή κειμένου, με την γραφίδα στο σημείο που είναι το λάθος.

FILER

Με την επιλογή της λειτουργίας Filer από το κυρίως menu πηγαίνουμε στον διαχειριστή αρχείων.

Σε μια δισκέτα ή στον σκληρό δίσκο μπορούν να υπάρχουν τρία είδη αρχείων.Ο διαχωρισμός τους γίνεται με την βοήθεια των τριών χαρακτήρων που ακολουθούν το όνομα του αρχείου μετά την τελεία (extension).Αυτά είναι :

XXXXXXXX.TXT : Αρχεία που περιέχουν προγράμματα εφαρμογής.

XXXXXXXX.BIN : Αρχεία που περιέχουν κώδικα σε δυαδική μορφή.

XXXXXXXX.HEX : Αρχεία που περιέχουν κώδικα σε δεκαεξαδική μορφή.

Οι επεκτάσεις .XXX είναι απαραίτητο να πληκτρολογηθούν μόνο στις υπολειτουργίες COPY (Αντιγραφή) και KILL (Διαγραφή) του FILER.Σε όλες τις άλλες περιπτώσεις που ζητείται το όνομα ενός αρχείου, το τελευταίο πρέπει να πληκτρολογηθεί χωρίς αυτές.

SIMULATOR

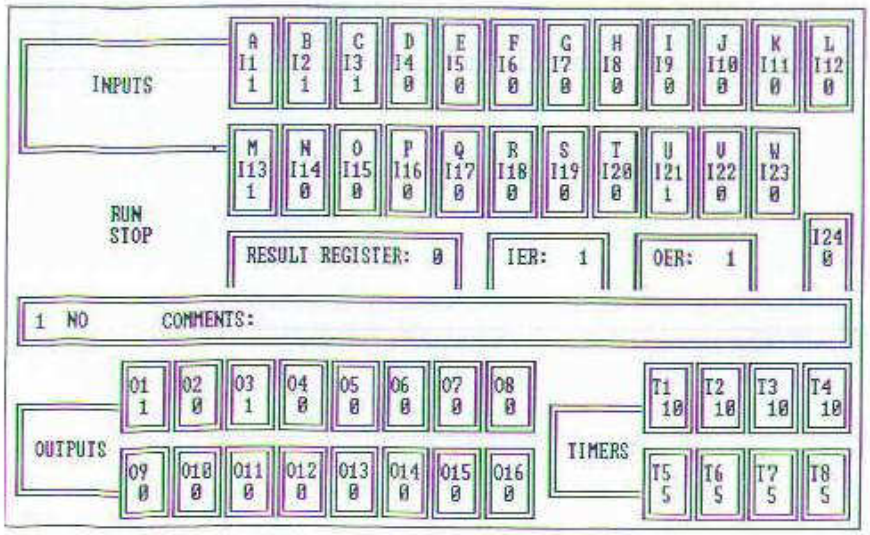
Ο Simulator προσομοιώνει την κεντρική μονάδα του PS-14N.Το πρόγραμμα εφαρμογής εκτελείται κυκλικά και τα τρέχοντα αποτελέσματα μεταφέρονται στην οθόνη.

Στο Σχήμα 2.1 έχουμε την οθόνη του Simulator, όπου όπως φαίνεται έχουμε μια πλήρη εποπτεία της κατάστασης όλων των εισόδων, εξόδων και χρονικών του PS-14N.

Στο μέσον της οθόνης έχουμε την δυνατότητα να βλέπουμε το πρόγραμμα μας (STL) γραμμή ανά γραμμή με την βοήθεια των cursor keys. Στο άνω μέρος της οθόνης έχουμε τις ενδείξεις των 23 εισόδων του PS-14N ενώ στο κάτω μέρος έχουμε αριστερά τις 16 εξόδους και δεξιά τα 8 διαθέσιμα χρονικά.

Μέσω του πληκτρολογίου μπορούμε να θέτουμε στην επιθυμητή κατάσταση (1 ή 0) τις εισόδους. Σε κάθε είσοδο αντιστοιχεί ένα πλήκτρο που αναγράφεται πάνω στην οθόνη.

Η οθόνη του Simulator



Σχήμα 2.1: Εισαγωγή σε περιβάλλον Ladder

Μέσω των function keys (F1 … F8) μπορούμε να μεταβάλουμε τον χρόνο καθυστέρησης των χρονικών, π.χ. με F1 (shift F1στην επέκταση) αυξάνουμε την χρονοκαθυστέρηση του χρονικού T1 σε βήματα του ενός δευτερολέπτου, ενώ με ALT-F1 (ctrl F1 στην επέκταση) την μειώνουμε. Αυτοί οι χρόνοι δεν χάνονται αν επιστρέψουμε κάποια στιγμή στον Editor.

Η αλλαγή της κατάστασης των χρονικών γίνεται ορατή με την αλλαγή του χρώματος του πλαισίου.

Στο παράθυρο κειμένου έχουμε την δυνατότητα να βλέπουμε μία μία τις οδηγίες του προγράμματος εφαρμογής που προσομοιώνουμε, πηγαίνοντας προς τα πάνω ή προς τα κάτω μία οδηγία, με την χρήση των πλήκτρων κίνησης της γραφίδας. Το αποτέλεσμα της εκτέλεσης της οδηγίας που βρίσκεται στο παράθυρο περιέχεται στον καταχωρητή (RR, I24).Στα παράθυρα IER και OER περιέχονται οι καταστάσεις των ομώνυμων καταχωρητών κατά την εκτέλεση της οδηγίας που επιλέξαμε στο παράθυρο κειμένου.

Εντολές:

F1…8 : Αύξηση του χρόνου των χρονικών T1…8

ALT F1…8 : Μείωση του χρόνου των χρονικών T1…8

F9 : GOTO LINE NR.

Επιλογή σειράς του προγράμματος

F10 : START/STOP.

Έναρξη ή διακοπή της κυκλικής επεξεργασίας.Με τη διακοπή της κυκλικής επεξεργασίας μηδενίζονται οι έξοδοι και τα χρονικά και ο μετρητής προγράμματος (program counter) του προσομοιωτή δείχνει την πρώτη εντολή του προγράμματος.

ESC : Επιστροφή στο κυρίως menu του προγράμματος.

EXIT

Με την επιλογή της λειτουργίας Exit επιστρέφουμε πίσω στο λειτουργικό σύστημα (DOS).

2.2 Παραδείγματα-Εφαρμογές

α) Με την βοήθεια του Editor θα περάσουμε στην μνήμη το ακόλουθο πρόγραμμα.Μετά την επιλογή του Editor, ονομάζουμε το πρόγραμμα π.χ. Sample1 και ENTER.

Στον κειμενογράφο πληκτρολογούμε τις ακόλουθες εντολές (STL πρόγραμμα) :

L I1

O I2

= O1

PE

Πατώντας στην συνέχεια ESC και μετά Save το πρόγραμμα αποθηκεύεται αυτόματα στον σκληρό δίσκο.

Με ESC και Exit επιστρέφουμε στο κεντρικό menu.Με τον Compiler ελέγχουμε το πρόγραμμα για τυχόν λάθη. Εάν υπάρχουν τα διορθώνουμε μέσω του Editor αλλιώς πηγαίνουμε στον Simulator, όπου μπορούμε να δούμε την λειτουργία του προγράμματος. Επιλέγουμε τον Simulator από το βασικό menu και μας ζητείται το όνομα του αρχείου που θα χρησιμοποιήσουμε. Πληκτρολογούμε Sample1 και ENTER και στην οθόνη εμφανίζεται η οθόνη του Simulator με τις διάφορες ενδείξεις.

Θέτουμε τις εισόδους I1=1, I2=1 με την χρήση των πλήκτρων A και B αντίστοιχα και κάνουμε RUN στο πρόγραμμα πατώντας το F10α από τα function keys.

Στην συνέχεια θέτουμε τις εισόδους I1 και I2 με τους τρεις υπόλοιπους δυνατούς συνδυασμούς και καταγράφουμε την έξοδο O1 για κάθε περίπτωση στον παρακάτω πίνακα :

INPUTS		OUTPUT
I1	I2	O1
1	1	
0	1	
1	0	
0	0	

Αφού τελειώσουμε, πατώντας ESC επιστρέφουμε στο κεντρικό menu.

β) Επιστρέφουμε στο κεντρικό menu. Με τη βοήθεια του Editor θα περάσουμε στην μνήμη το πρόγραμμα Sample2.

Στον κειμενογράφο πληκτρολογούμε τις ακόλουθες εντολές (STL πρόγραμμα) :

```
L     I1
A     I2
=     O1
PE
```

Πατώντας στην συνέχεια ESC και μετά Save το πρόγραμμα αποθηκεύεται αυτόματα στον σκληρό δίσκο.

Με ESC και Exit επιστρέφουμε στο κεντρικό menu.Με τον Compiler ελέγχουμε το πρόγραμμα για τυχόν λάθη. Εάν υπάρχουν τα διορθώνουμε μέσω του Editor αλλιώς πηγαίνουμε στον Simulator, όπου μπορούμε να δούμε την λειτουργία του προγράμματος. Ακολουθούμε την ίδια διαδικασία όπως και πριν και θέτουμε τους 4 δυνατούς συνδυασμούς των εισόδων I1 και I2 καταγράφοντας την έξοδο O1 κάθε φορά στον ακόλουθο πίνακα :

INPUTS		OUTPUT
I1	I2	O1
1	1	
0	1	
1	0	
0	0	

γ) Επιστρέφουμε στο κεντρικό menu.Με την βοήθεια του Editor θα περάσουμε στην μνήμη το πρόγραμμα Sample3.

Στον κειμενογράφο πληκτρολογούμε τις ακόλουθες εντολές (STL πρόγραμμα) :

```
L    I1
S    O1
L    O1
=    T1
L    T1
R    O1
PE
```

Με τον Compiler ελέγχουμε το πρόγραμμα για τυχόν λάθη. Πηγαίνουμε στον Simulator, όπου μπορούμε να δούμε την λειτουργία του προγράμματος.

Θέτουμε τα εξής δεδομένα : I1=1 και με το πλήκτρο F1, T1=15 sec.Κάνουμε RUN με F10 και παρατηρούμε την έξοδο O1.

δ) Επιστρέφουμε στο κεντρικό menu.Με την βοήθεια του Editor θα περάσουμε στην μνήμη το ακόλουθο πρόγραμμα. Μετά την επιλογή του Editor, ονομάζουμε το πρόγραμμα με τρία γράμματα της αρεσκείας μας, π.χ. LGM και ENTER.

Στον κειμενογράφο πληκτρολογούμε τις ακόλουθες εντολές (STL πρόγραμμα) :

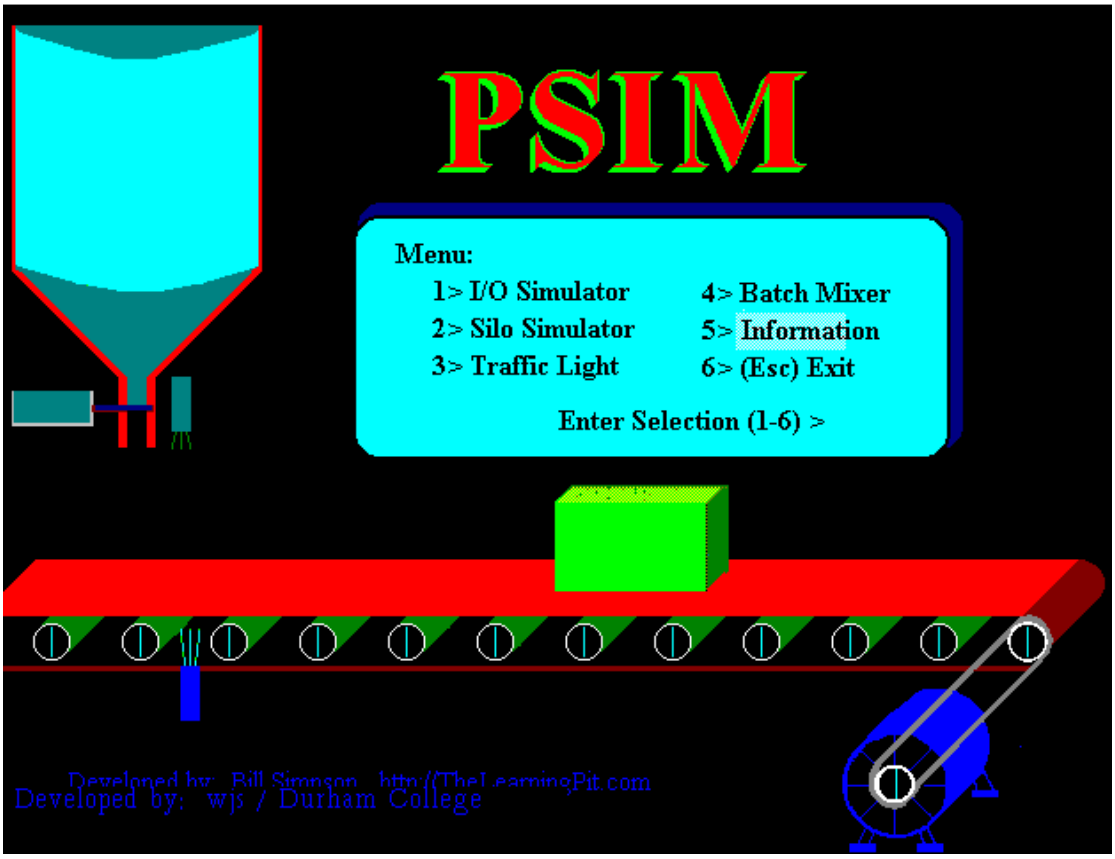
```
L    I1
O    I2
=    T1
L    T1
=    O1
PE
```

ΚΕΦΑΛΑΙΟ 3

ΠΡΟΣΟΜΟΙΩΤΗΣ PLC - PSIM

3.1 Περιγραφή του προγράμματος προσομοίωσης PLC –PSIM

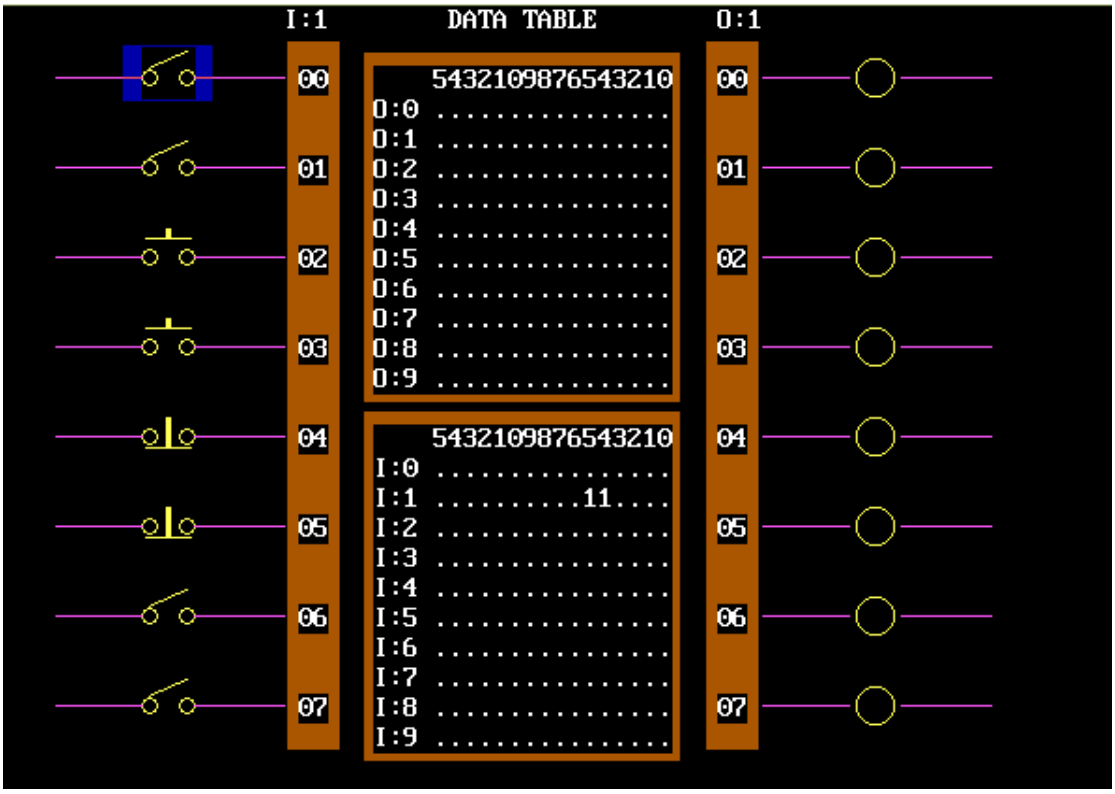
Το PSIM είναι ένα πρόγραμμα προσομοίωσης PLC (Programmable Logic Controller). Το PSIM δίνει στον χρήστη την δυνατότητα να προγραμματίσει την συμπεριφορά προσομοιωμένων εξόδων συνδεδεμένων στο PLC χρησιμοποιώντας εισόδους και προγραμματίζοντας το PLC με χρήση κώδικα Ladder



Σχήμα 3.1: PSIM κεντρική οθόνη

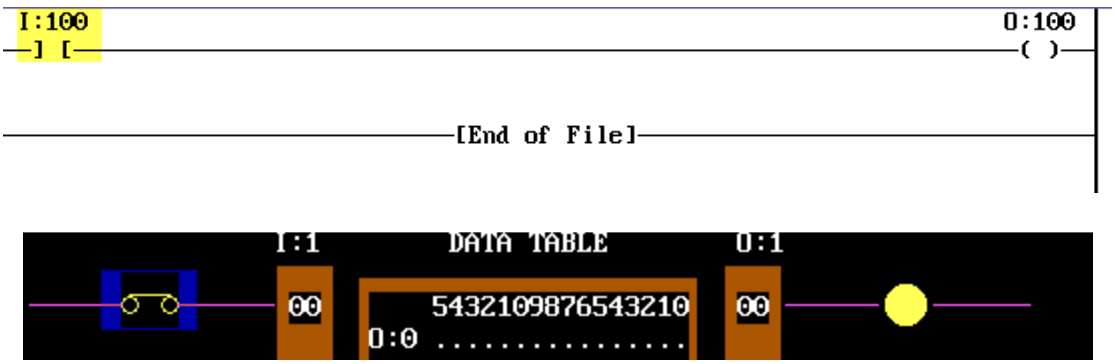
Το PSIM έχει 4 προσομοιώσεις PLC της οποίες μπορεί ο χρήστης να διαχειριστεί, και η κάθε μια από αυτές τις προσομοιώσεις συμπεριλαμβάνει πίνακες με δεδομένα για τις καταστάσεις των εισόδων, εξόδων, μεταβλητών Timer ή Counter και άλλων χρήσιμων πληροφοριών

3.1.1 I/O Simulator



Σχήμα 3.2: I/O Simulator

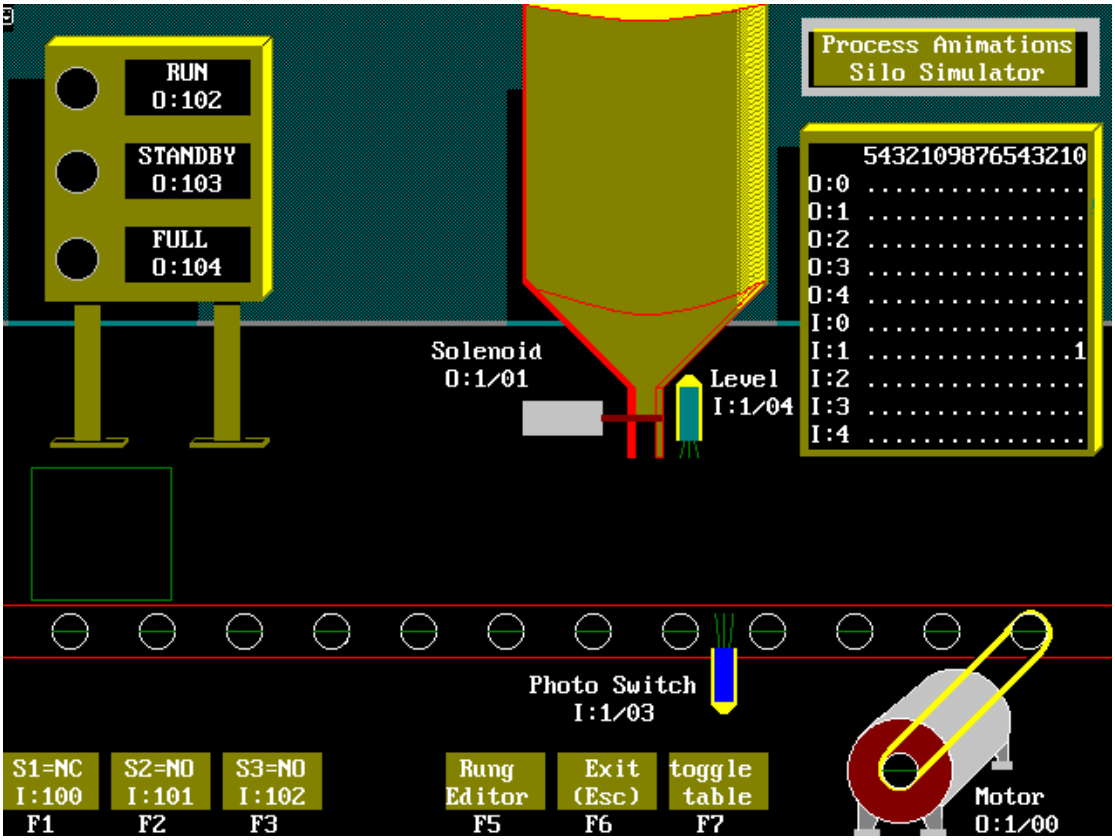
Η προσομοίωση I/O Simulator δίνει στον χρήστη την δυνατότητα να διαχειριστεί την συμπεριφορά 8 εισόδων και 8 εξόδων χρησιμοποιώντας την γλώσσα Ladder στον Rung Editor.



Με την εντολή Examine if Closed (XIC) όταν ο διακόπτης 1:00 είναι κλειστός, η έξοδος 1:00 θα ενεργοποιείται.

Αυτή η προσομοίωση βοηθάει στην κατανόηση των εντολών της γλώσσας Ladder

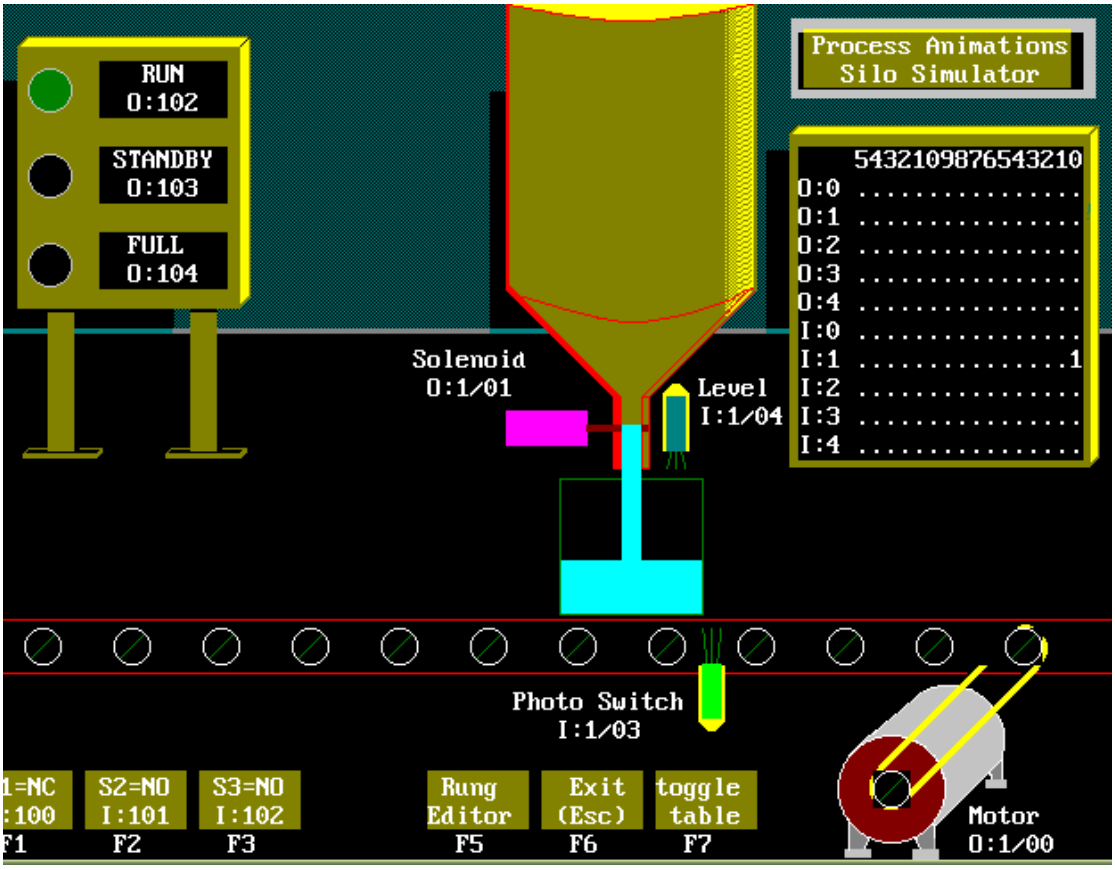
3.1.2 Silo Simulator

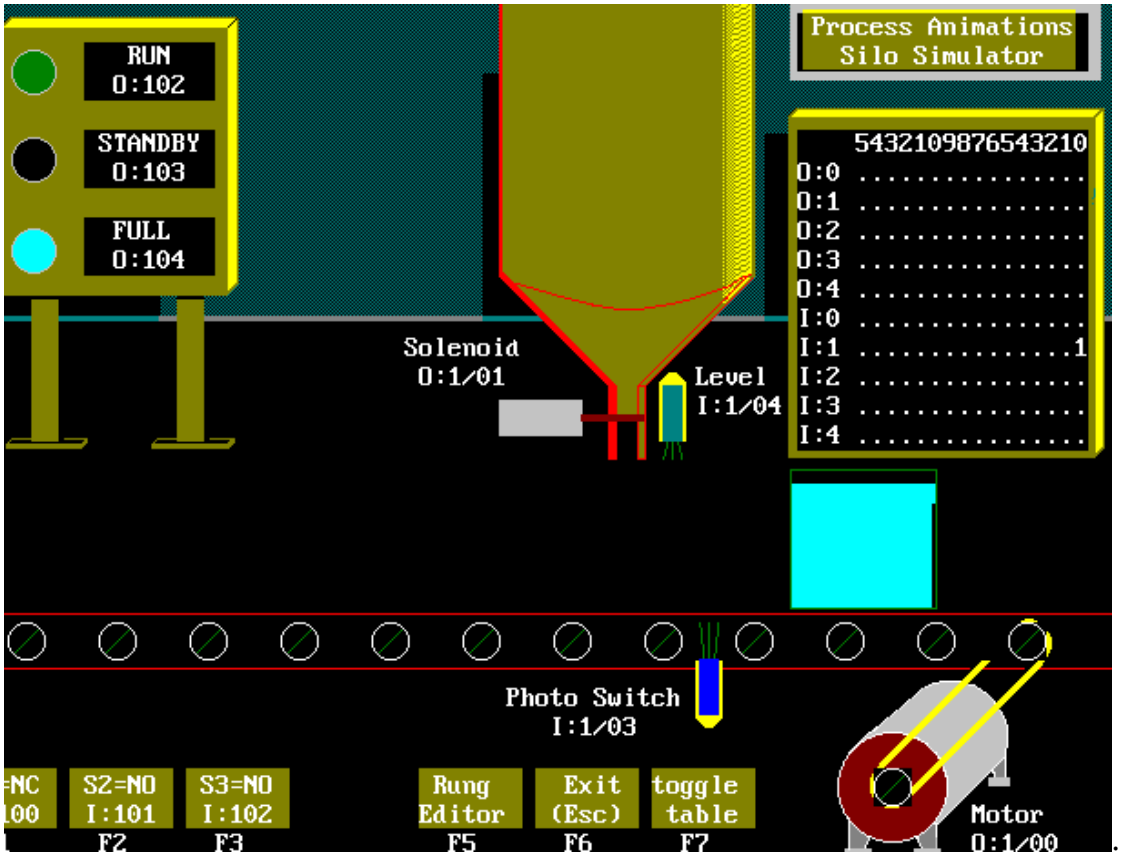


Σχήμα 3.3: Silo Simulator

Σε αυτή την προσομοίωση σκοπός του χρήστη είναι μέσω της γλώσσας Ladder να φτιάξει την λογική ενός εργοστασίου. Υπάρχουν αισθητήρες και κινητήρες που μπορεί ο χρήστης να προγραμματίσει για να κινήσει τον ιμάντα και να γεμίσει το κουτί. Σε αυτή την προσομοίωση υπάρχει κατάσταση αποτυχίας αν ο χρήστης χύσει το υλικό έξω από το κουτί.

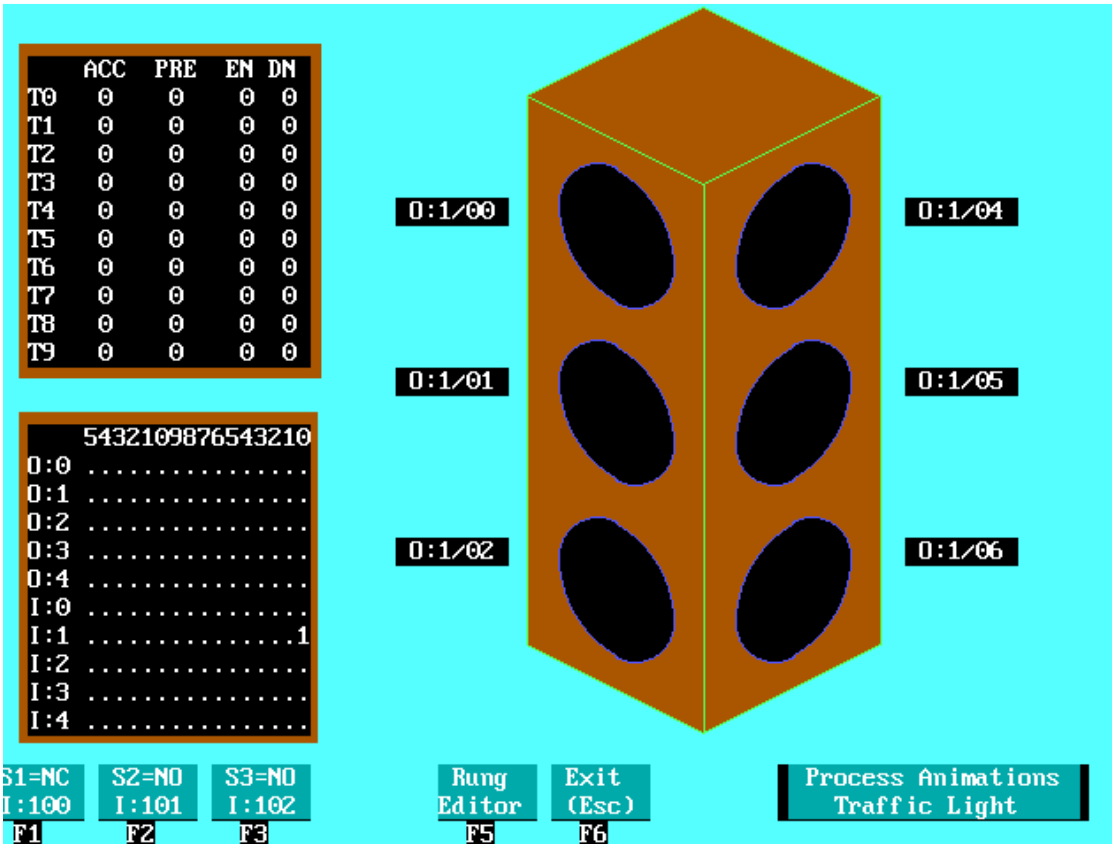
Αυτή η προσομοίωση έχει σκοπό ο χρήστης να καταφέρει το επιθυμητό αποτέλεσμα χρησιμοποιώντας τις εντολές της γλώσσας Ladder που έχει κατανοήσει χωρίς να κάνει κάποια “ζημιά” στην προσομοίωση


RUN
O:102
STANDBY
O:103
FULL
O:104
Process Animations
Silo Simulator
5432109876543210
O:0 ....
O:1 ....
O:2 ....
O:3 ....
O:4 ....
I:0 ....
I:1 1
I:2 ....
I:3 ....
I:4 ....
Solenoid
O:1/01
Level
I:1/04
Photo Switch
I:1/03
1=NC
:100
F1
S2=NO
I:101
F2
S3=NO
I:102
F3
Rung
Editor
F5
Exit
(Esc)
F6
toggle
table
F7
Motor
O:1/00


RUN
O:102
STANDBY
O:103
FULL
O:104
Process Animations
Silo Simulator
5432109876543210
O:0 ....
O:1 ....
O:2 ....
O:3 ....
O:4 ....
I:0 ....
I:1 1
I:2 ....
I:3 ....
I:4 ....
Solenoid
O:1/01
Level
I:1/04
Photo Switch
I:1/03
S2=NO
I:101
F2
S3=NO
I:102
F3
Rung
Editor
F5
Exit
(Esc)
F6
toggle
table
F7
Motor
O:1/00

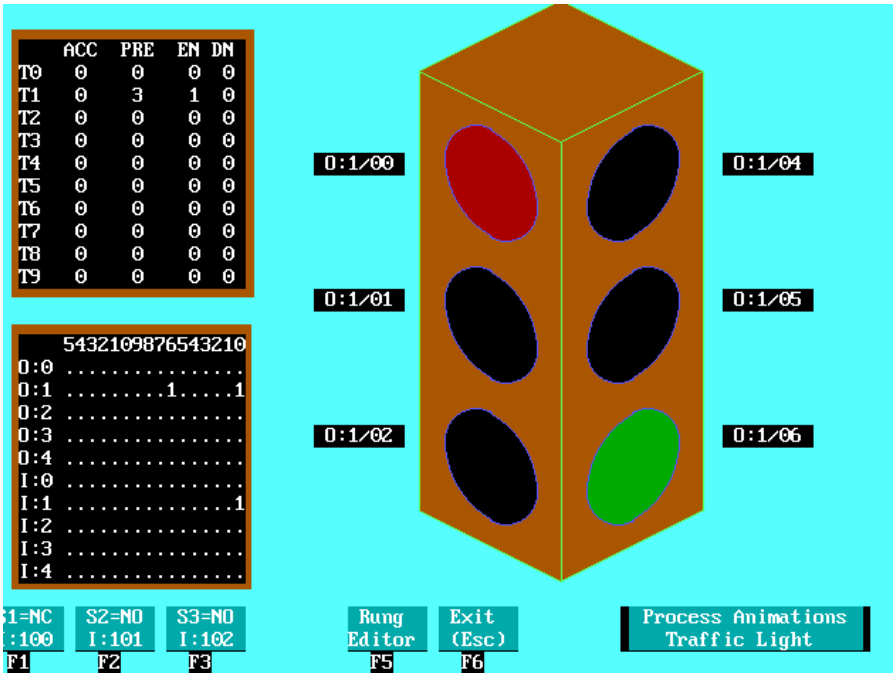
Το Silo Simulator έχει animation για να βλέπουμε τα αποτελέσματα του προγραμματισμού μας

3.1.3 Traffic Light

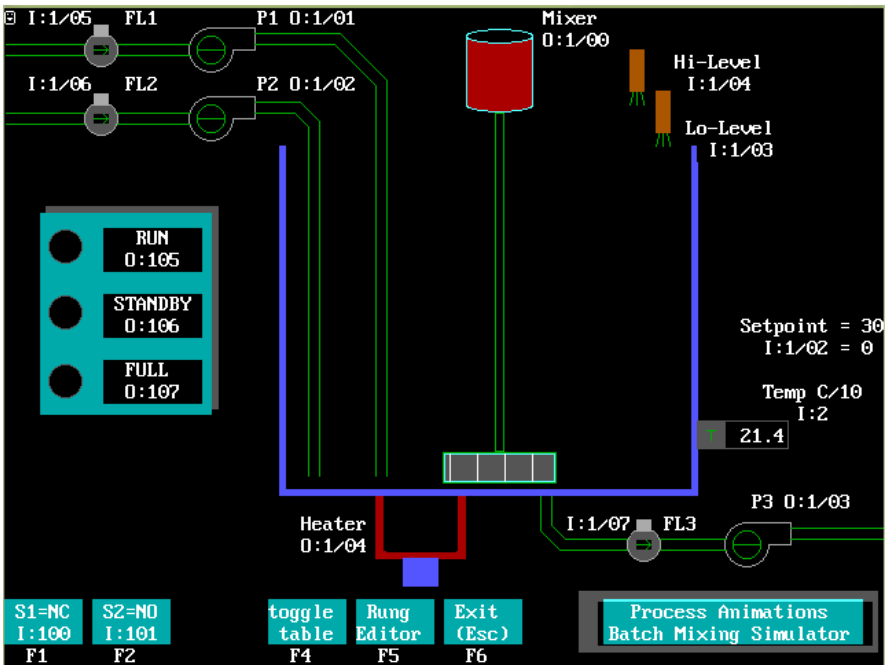


Σχήμα 3.4: Traffic Light

Αυτή η προσομοίωση δίνει στον χρήστη την δυνατότητα να προγραμματίσει ένα φανάρι. Το κάθε φως του φαναριού είναι μια έξοδος. Σκοπός της προσομοίωσης είναι να διδάξει στον χρήστη πώς να συγχρονίζει εξόδους, καθώς όταν η μια πλευρά είναι πράσινη, ή κίτρινη, η άλλη πλευρά του φαναριού θα πρέπει να είναι κόκκινη

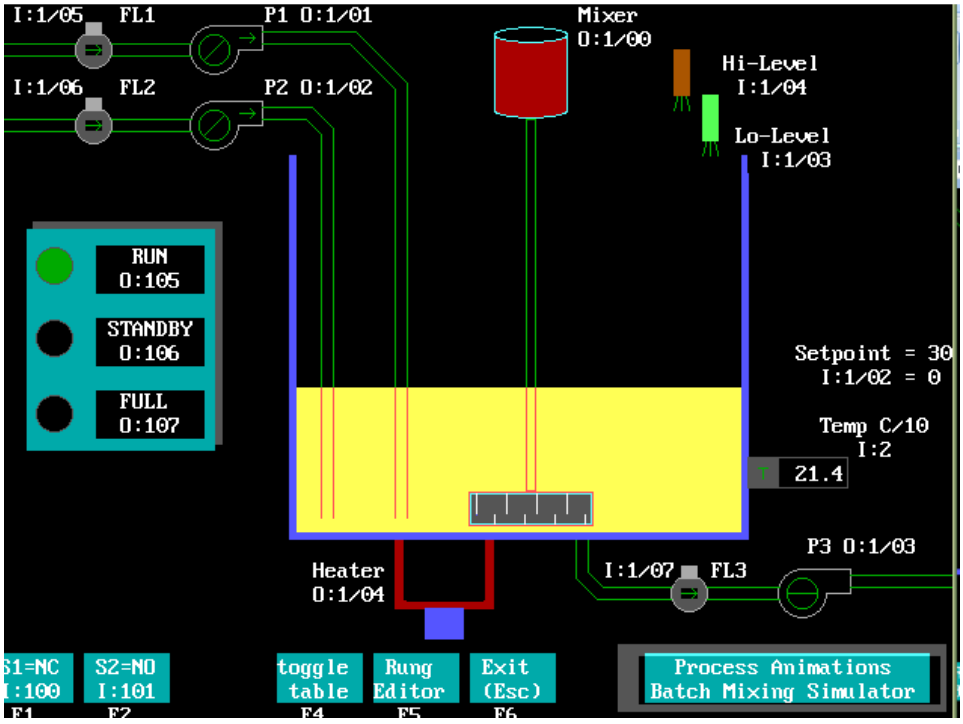


3.1.4 Batch Mixer



Σχήμα 3.5: Batch Mixer

Αυτή είναι η πιο πολύπλοκη προσομοίωση του PSIM. Δίνει στον χρήστη τον έλεγχο διάφορων αισθητήρων ,κινητήρων κ.α. για την μίξη και θέρμανση ενός μίγματος. Αυτή η προσομοίωση κατάσταση αποτυχίας αν η θερμοκρασία ανέβει υπερβολικά. Σκοπός της προσομοίωσης είναι ο χρήστης να χρησιμοποιήσει την γνώση του από τις προηγούμενες προσομοίωσης για να καταφέρει τον προγραμματισμό ενός ρεαλιστικού σεναρίου χρήσης PLC.



3.1.5 Προγραμματισμός σε Ladder

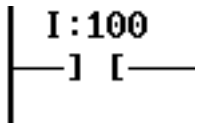
Ο τρόπος με τον οποίο προγραμματίζουμε την συμπεριφορά εξόδων μέσα από ένα PLC είναι με την γλώσσα Ladder. Η γλώσσα Ladder μας αφήνει να γράψουμε γραμμές με εντολές (instructions) οι οποίες αν είναι αληθείς, θα ενεργοποιούν την έξοδο στο τέλος της γραμμής.

Δίπλα από κάθε είσοδο και έξοδο της κάθε προσομοίωσης του PSIM υπάρχει μια ετικέτα (Label) που το περιγράφει, π.χ. ένας διακόπτης μπορεί να έχει δίπλα του την ετικέτα I:1/00 . Αυτός είναι ο τρόπος ονομασίας υλικού στην γλώσσα Ladder, και χρησιμοποιώντας τες, ο χρήστης μπορεί να ορίσει σε ποια έξοδο ή είσοδο αναφέρετε η κάθε εντολή και έτσι να φτιάξει τις σχέσεις εισόδων-εξόδων με την γλώσσα Ladder. Με

το γράμμα “I” ορίζουμε τις εισόδους της προσομοίωσης και με το γράμμα “O” τις εξόδους.

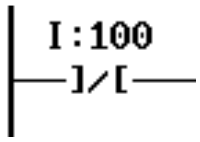
Εντολές της γλώσσας Ladder

- Examine if Closed (XIC)



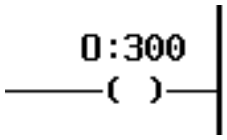
Όταν η είσοδος στην οποία αντιστοιχεί η εντολή XIC είναι κλειστή (ολοκληρώνει το κύκλωμα), τότε η εντολή XIC γίνετε αληθής. Όταν η είσοδος είναι ανοιχτή (δεν ολοκληρώνει το κύκλωμα) τότε η εντολή XIC γίνεται ψευδής.

- Examine if Open (XIO)



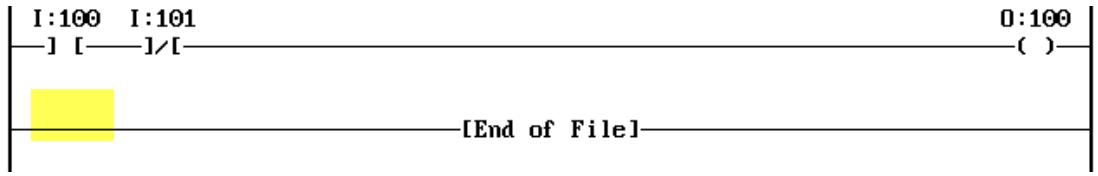
Η αντίθετη εντολή της XIC. Όταν η αντίστοιχη είσοδος της XIO είναι κλειστή, τότε η εντολή XIO γίνεται ψευδής. Αντιθέτως όταν η είσοδος είναι ανοιχτή, τότε η εντολή XIO γίνεται αληθής.

- Out Energize (OTE)

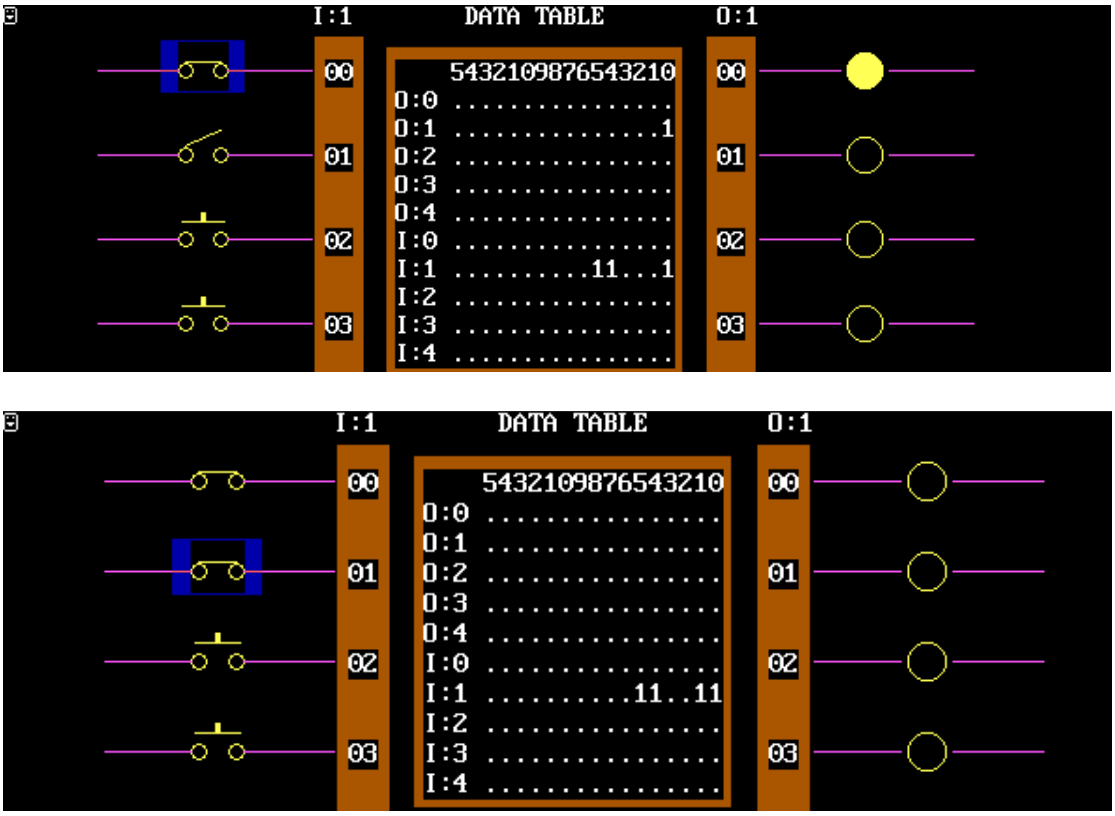


Η OTE ενεργοποιείται αν η γραμμή στην οποία ανήκει είναι αληθής. Αν η γραμμή στην οποία ανήκει είναι ψευδής, τότε δεν ενεργοποιείται. Όταν η OTE ενεργοποιείται, τότε ενεργοποιείται και η έξοδος με την αντίστοιχη ετικέτα.

Παράδειγμα XIC , XIO και OTE

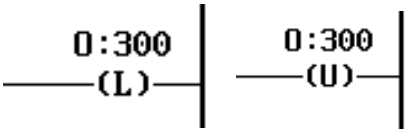


Σχήμα 3.6: Παράδειγμα XIC , XIO και OTE



Στο παραπάνω Σχήμα 3.6 όταν η είσοδος I:100 είναι κλειστή, ΚΑΙ η είσοδος I:101 είναι ανοιχτή, η έξοδος o:100 θα ενεργοποιείται. Βάζοντας πολλαπλές εντολές σε μια γραμμή, φτιάχνουμε το λογικό ‘AND’ στην γλώσσα Ladder.

- Output Latch (OTL) Και Output Unlatch OTU)

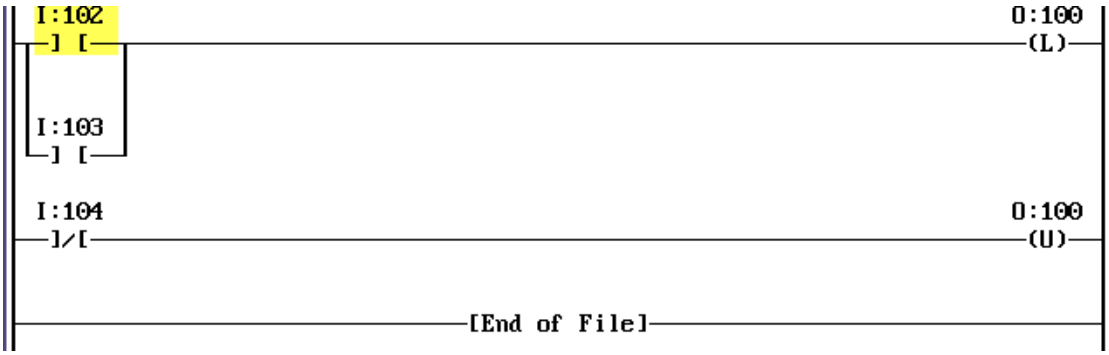


Η εντολή OTL ενεργοποιείται με τον ίδιο τρόπο που ενεργοποιείτε η OTE. Αν η εντολή OTL ενεργοποιηθεί μια φορά, ενεργοποιεί την έξοδο με την αντίστοιχη ετικέτα, και την κρατάει ενεργοποιημένη, ακόμα και αν η γραμμή στην οποία βρίσκετε γίνει ψευδής, μέχρι να ενεργοποιηθεί μια εντολή OTU για την ίδια έξοδο. Η εντολή OTU ενεργοποιείται και αυτή όπως η OTE.

- Branch (Παρακλάδι)

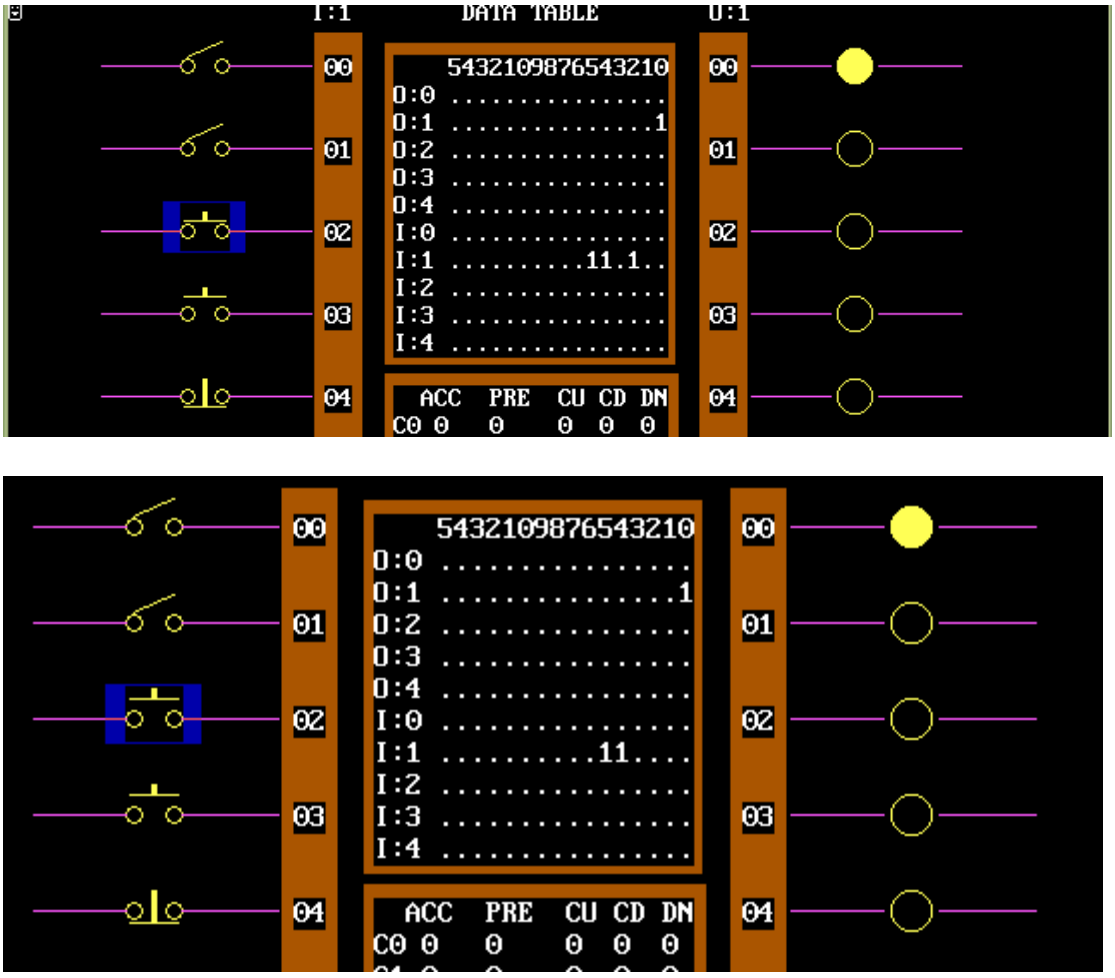
Χρησιμοποιούμε παρακλάδια στην γλώσσα Ladder, όταν θέλουμε να χρησιμοποιήσουμε το λογικό OR σε μια γραμμή. Αν οποιοδήποτε από τα παρακλάδια γίνει αληθές, τότε η γραμμή μέχρι το τέλος του παρακλαδιού θα είναι αληθής. Στην παραπάνω εικόνα, αν η είσοδος I:300 είναι κλειστή, ή αν η είσοδος I:301 είναι κλειστή ΚΑΙ η είσοδος I:302 είναι ανοιχτή, τότε το παρακλάδι θα είναι αληθές.

Παράδειγμα Output Latch και Branch

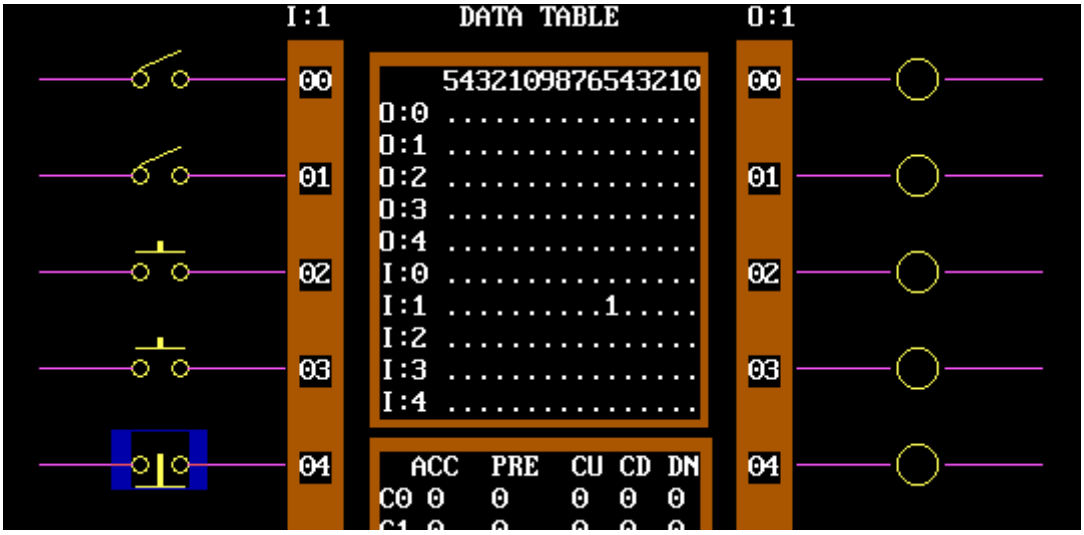


Σχήμα 3.7: Παράδειγμα Output Latch και Branch

Στο παραπάνω Σχήμα 3.7 όταν το κουμπί I:102 ή I:103 πατηθούν, η έξοδος O:100 θα ενεργοποιηθεί, και θα μείνει ενεργοποιημένη μέχρι να σταματήσει να είναι πατημένο το κουμπί I:104

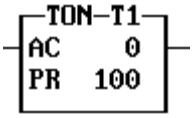


Πατώντας το κουμπί I:102 το λαμπάκι O:100 ανάβει και μένει αναμμένο (Latched)



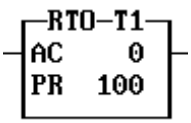
Όταν πατάμε το κουμπί I:104 το λαμπάκι O:100 γίνετε unlatch και σβήνει.

- Timer On-delay (TON)



Η εντολή TON αρχίζει να μετράει όταν η γραμμή στην οποία βρίσκετε είναι αληθής. Όσο η γραμμή είναι αληθής, η μεταβλητή ACC (Accumulated Value – Αυξανόμενη Μεταβλητή) αυξάνεται σταδιακά, μέχρι να φτάσει την τιμή της μεταβλητής PRE (Preset Value) την οποία έχει ορίσει ο προγραμματιστής. Η μεταβλητή AC μηδενίζεται αν η γραμμή σταματήσει να είναι αληθής. Όταν η μεταβλητή AC φτάσει την μεταβλητή PRE, η τιμή DN (Done Bit) του χρονομετρητή γίνεται ‘1’. Η τιμή EN (Enable Bit) του χρονομετρητή γίνεται ‘1’ όσο ο χρονομετρητής είναι ενεργοποιημένος Οι τιμές DN και EN μηδενίζονται όταν ο χρονομετρητής σταματάει να είναι ενεργός.

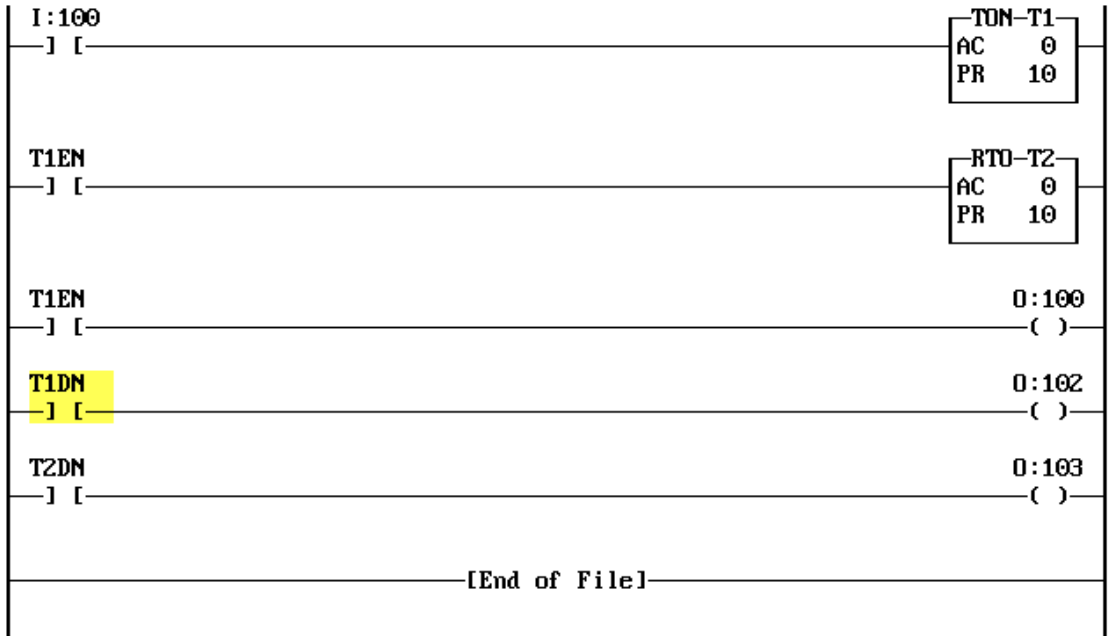
- Retentive Timer (RTO)



Ο χρονομετρητής RTO έχει παρόμοια λειτουργία με τον χρονομετρητή TON, με την διαφορά ότι όταν σταματάει να ενεργοποιείται ο χρονομετρητής RTO, οι μεταβλητές

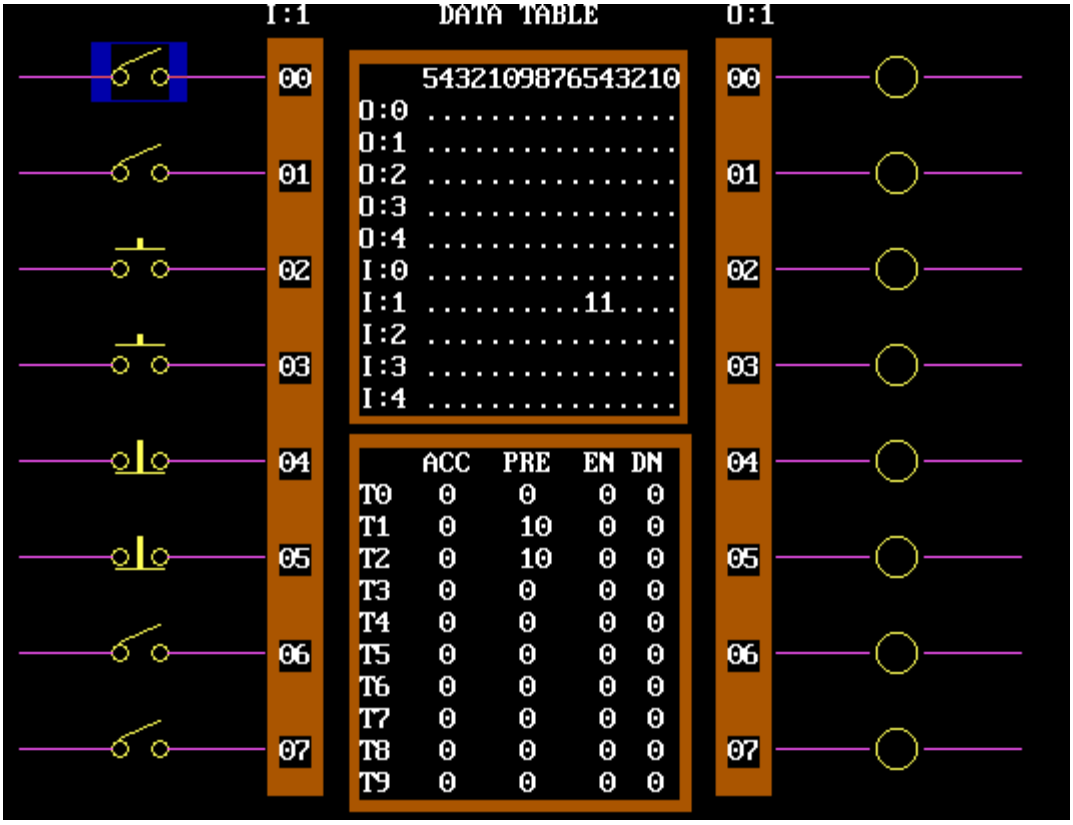
ACC και DN του δεν μηδενίζονται. Αντιθέτως μηδενίζονται όταν ενεργοποιείται μια εντολή RES με την ίδια ετικέτα του χρονομετρητή.

Παράδειγμα Timer TON και RTO

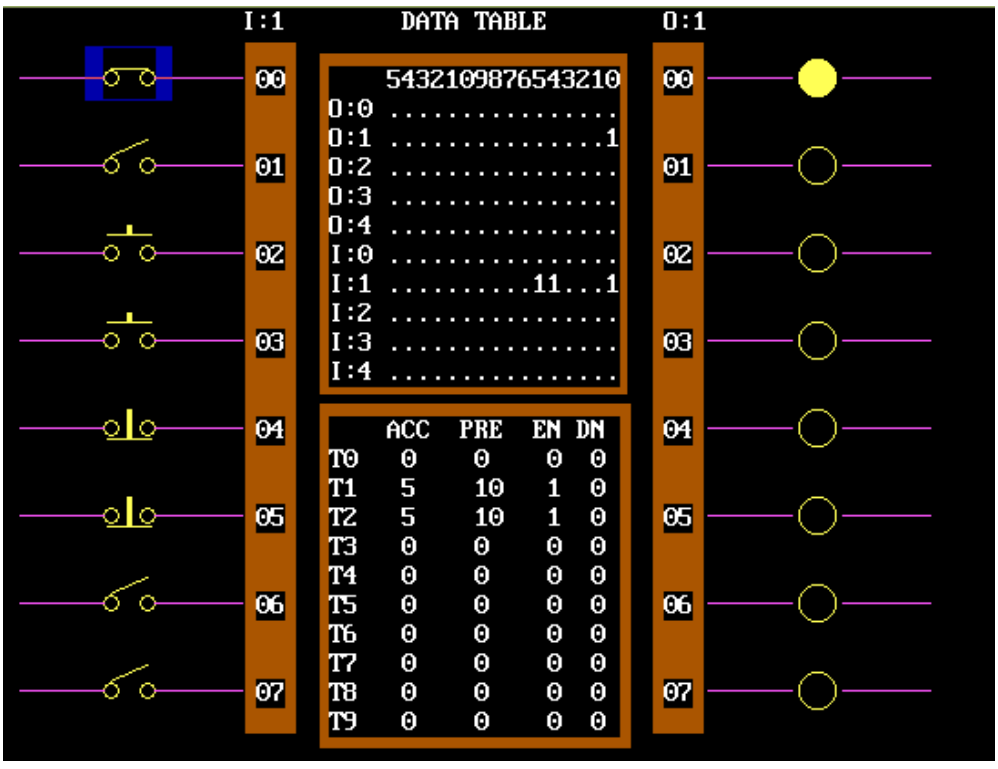


Σχήμα 3.8: Παράδειγμα Timer TON και RTO

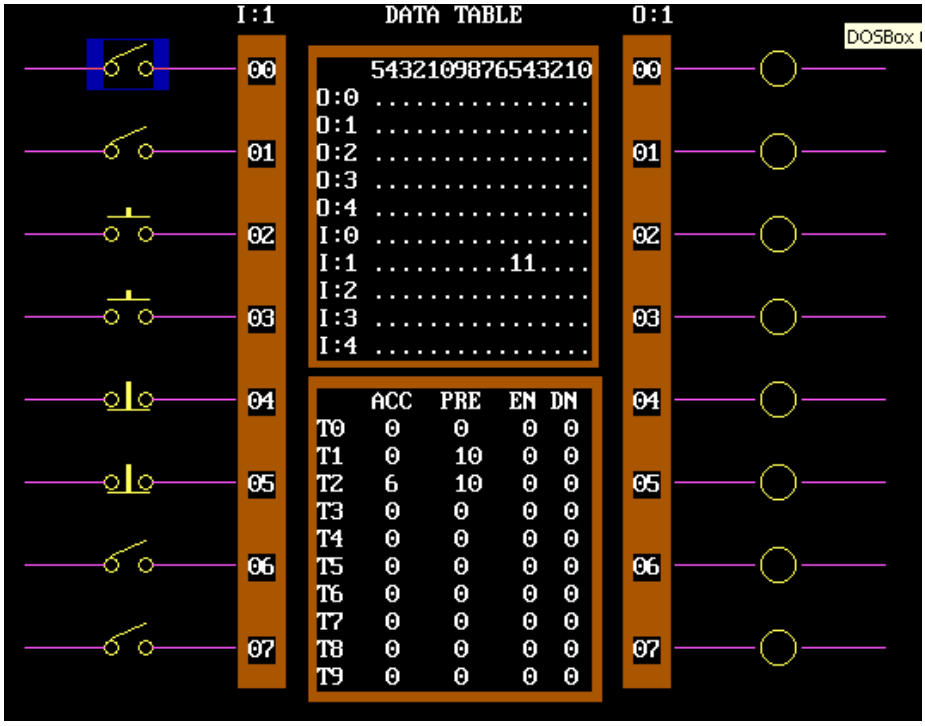
Στο Σχήμα 3.8 όταν πατάμε το κουμπί I:100 ο TON Timer T1 ξεκινάει να λειτουργεί. Όσο το T1:EN είναι αληθές, δηλαδή όσο ο T1 είναι ενεργοποιημένος, θα είναι και ο RTO Timer T2 ενεργοποιημένος, και η έξοδος O:100 θα είναι αναμμένη. Όταν ο μετρητής T1 μετρήσει μέχρι το 10 (δηλαδή η τιμή T1:AC = T1:PR) θα ανάψει το λαμπάκι O:102. Όταν ο μετρητής T2 μετρήσει μέχρι το 10, θα ανάψει το λαμπάκι O:102. Αν η είσοδος I:100 γίνει ψευδής, ο μετρητής T1 θα μηδενιστεί η τιμή T1:AC, αφού είναι μετρητής TON, ενώ ο T2 θα κρατήσει την τιμή του T2:AC καθώς είναι μετρητής RTO.



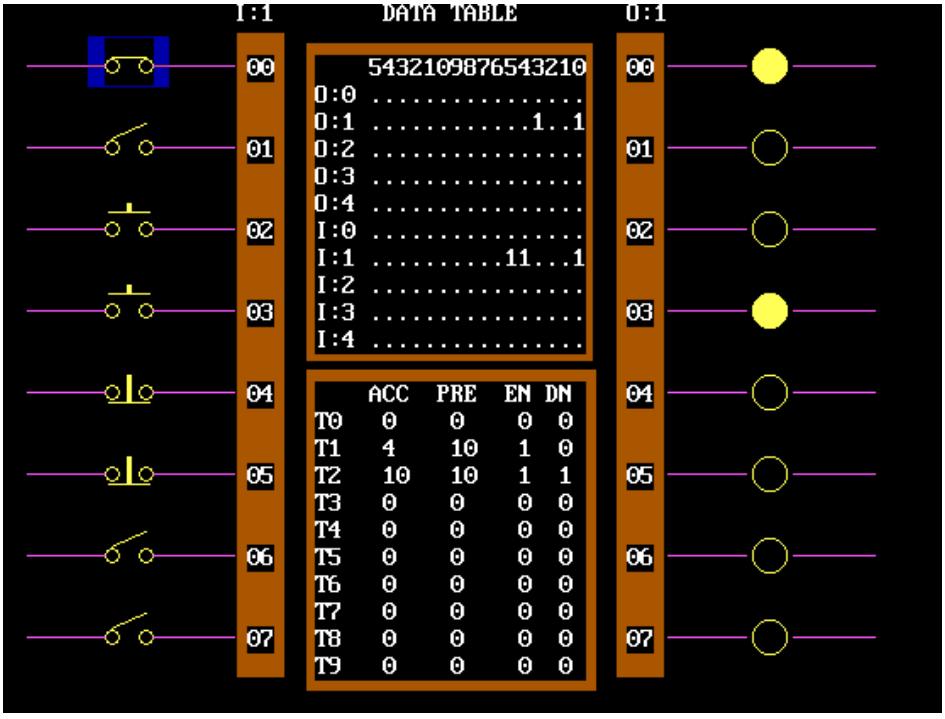
Όλοι οι διακόπτες είναι ανοιχτοί και στον πίνακα βλέπουμε ότι οι 2 μετρητές T1 και T2 δεν είναι ενεργοποιημένοι.



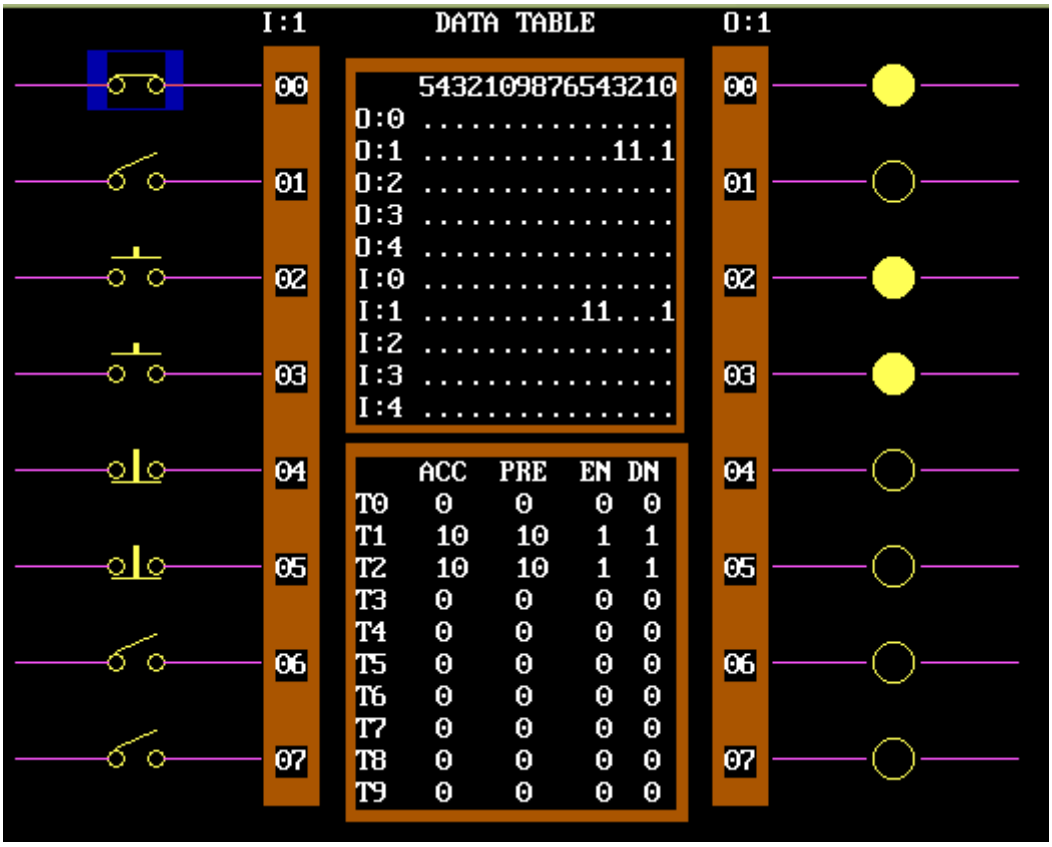
Όταν πατάμε το I:1/00 (πρώτος διακόπτης) το λαμπάκι O: 1/00 και οι Timer ενεργοποιούνται και όπως βλέπουμε στον πίνακα αρχίζουν να μετράνε μέχρι το ACC πάρει την τιμή του PRE.



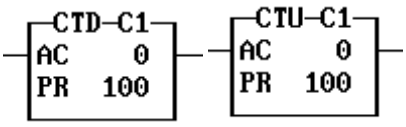
Όταν ανοίξουμε τον διακόπτη ο T1 (TON) μηδενίζεται, ενώ ο T2(RTO) κρατάει την τελευταία τιμή που είχε.



Ξανά πατάμε τον διακόπτη I:1/00 και οι μετρητές ξανά ξεκινάνε. Ο T2 τελειώνει πρώτος καθώς είχε κρατήσει την τιμή του ACC . Όταν τελειώνει ο T2, ανάβει η έξοδος O:1/03. Όταν τελειώνει ο T1, ανάβει η έξοδος O:1/02

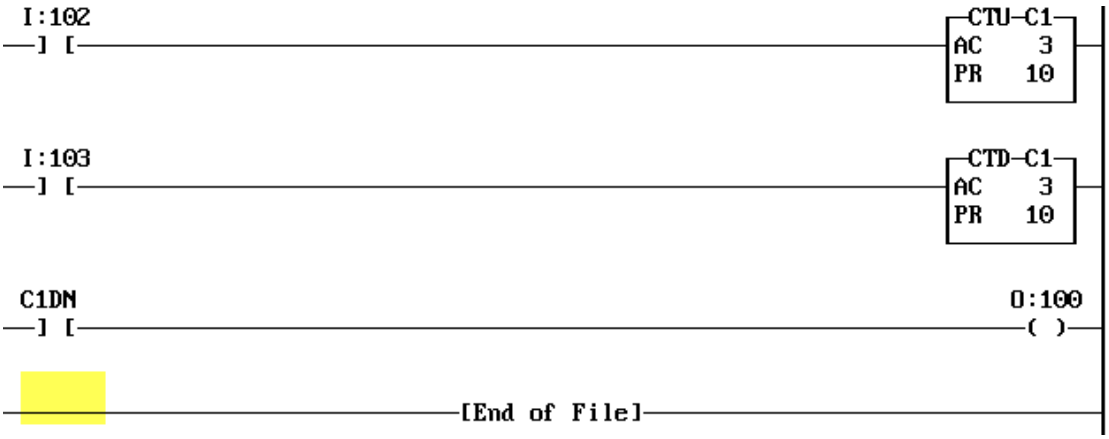


- Μετρητές Count Up (CTU) και Count Down (CTD)



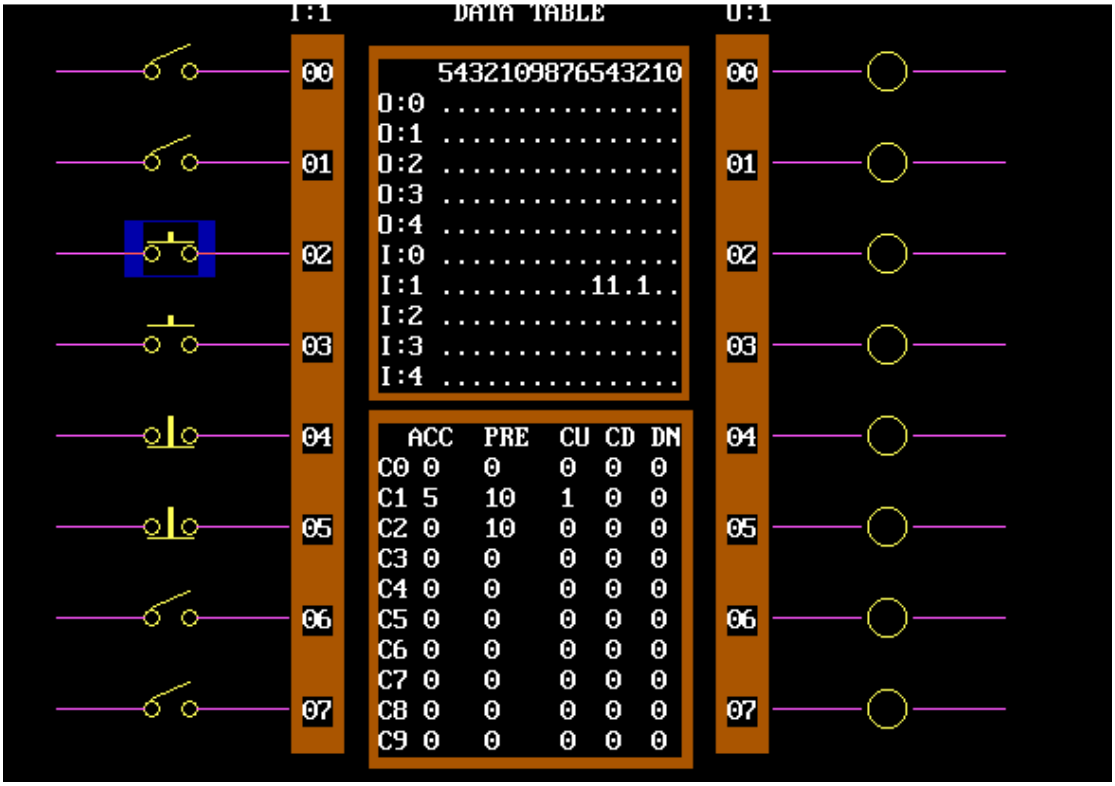
Οι μετρητές CTU και CTD έχουν και αυτοί μεταβλητές ACC PRE και DN όπως οι χρονομετρητές. Έχουν επίσης τις μεταβλητές CU , CD και CZ. Κάθε φορά που η γραμμή στην οποία βρίσκεται ο μετρητής ενεργοποιείται, η μεταβλητή CU ή CD (ανάλογα αν έχουμε CTU ή CTD) γίνεται ‘1’. Όταν η τιμή CU γίνει ‘1’, η μεταβλητή ACC αυξάνεται κατά +1. Όταν η τιμή CD γίνεται ‘1’ η μεταβλητή ACC μειώνεται κατά -1. Όταν η μεταβλητή ACC περάσει την μεταβλητή PRE, η τιμή DN γίνεται ‘1’. Οι τιμές ACC και DN των μετρητών παραμένουν μέχρι να ενεργοποιηθεί μια εντολή RES με την ίδια ετικέτα.

Παράδειγμα CTU, CTD και RES

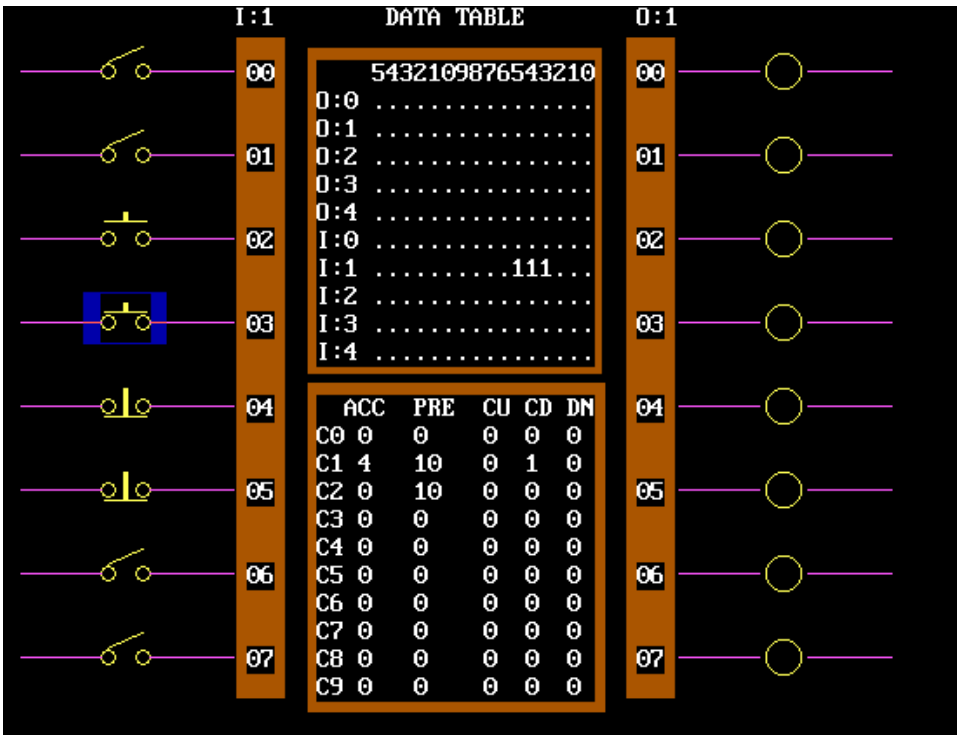


Σχήμα 3.9: Παράδειγμα CTU, CTD και RES

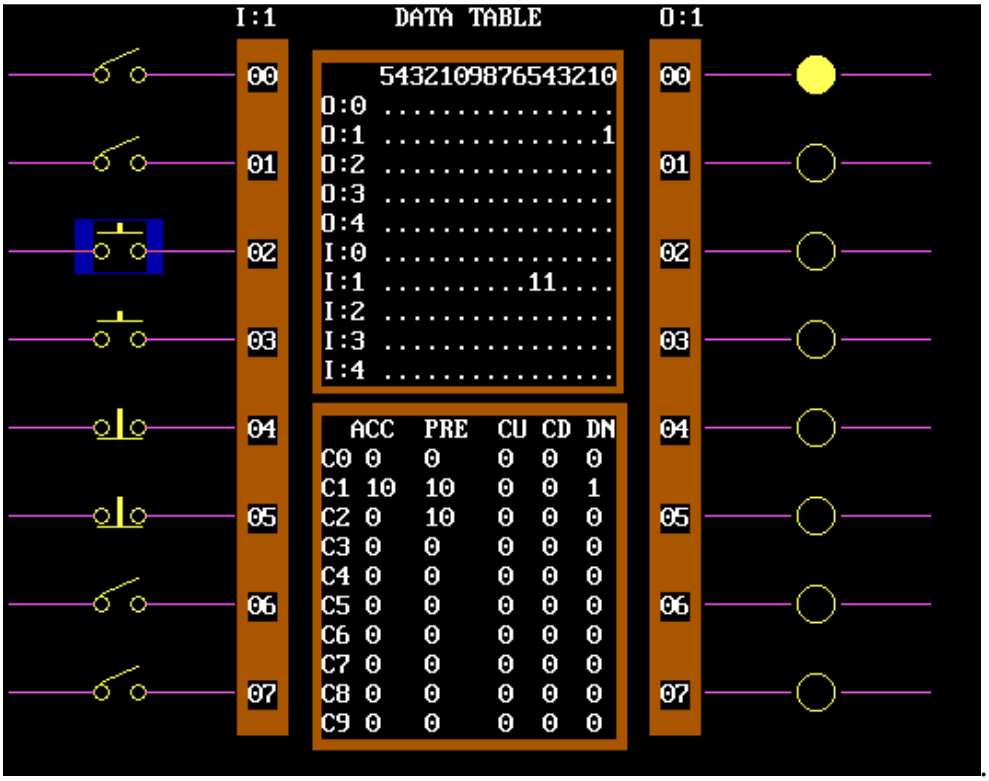
Κάθε φορά που πατάμε τον διακόπτη I:102 , ο μετρητής C1:AC θα αυξάνεται κατά 1 αλλά όταν πατάμε τον διακόπτη I:103 ο μετρητής C1:AC θα μειώνεται κατά 1. Όταν η τιμή C1:AC γίνει 10, θα ανάψει η έξοδος O:100 θα ανάψει.



Κάθε φορά που πατάμε τον διακόπτη I:1/02 η τιμή ACC του C1 αυξάνεται κατά 1.

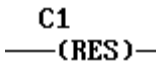


Κάθε φορά που πατάμε τον διακόπτη I:1/02 η τιμή ACC του C1 μειώνεται κατά 1.



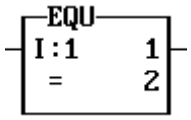
Όταν η τιμή C1:AC γίνει ίδια με την τιμή C1:PR τότε το λαμπάκι O:100 ανάβει.

- Reset (RES)



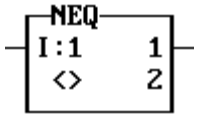
Όταν ενεργοποιείται μια εντολή RES, μηδενίζει τον χρονομετρητή ή μετρητή με την ίδια ετικέτα.

- Equal (EQU)



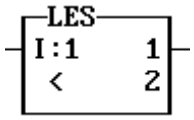
Η εντολή EQU συγκρίνει μια λογική διεύθυνση μεγέθους Word (16bit) με έναν αριθμό. Αν οι δυο τιμές είναι ίσες, τότε η EQU παίρνει την τιμή αληθής, αλλιώς παίρνει την τιμή ψευδής. Στην παραπάνω εικόνα η EQU συγκρίνει την το Word στην λογική διεύθυνση I:1 (που έχει τιμή 1 στο δεκαδικό) με την δεκαδική τιμή 2. Επειδή οι τιμές δεν είναι ίσες, η EQU είναι ψευδής

- Not Equal (NEQ)



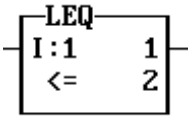
Η εντολή NEQ είναι το ακριβώς αντίθετο της EQU. Όταν οι δυο τιμές δεν είναι ίσες η NEQ γίνεται αληθής, ενώ όταν είναι ίσες οι NEQ γίνεται ψευδής.

- Less Than (LES)



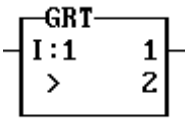
Η εντολή LES συγκρίνει μια λογική διεύθυνση με έναν αριθμό. Αν η τιμή της λογική διεύθυνσης είναι μικρότερη από τον αριθμό που έδωσε ο προγραμματιστής, τότε η LES είναι αληθής, αλλιώς είναι ψευδής

- Less Than or Equal (LEQ)



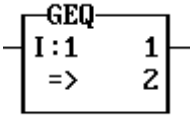
Όμοια εντολή με την LES, με την διαφορά ότι αν οι αριθμοί είναι ίσοι, η LEQ είναι αληθής.

- Greater Than (GRT)



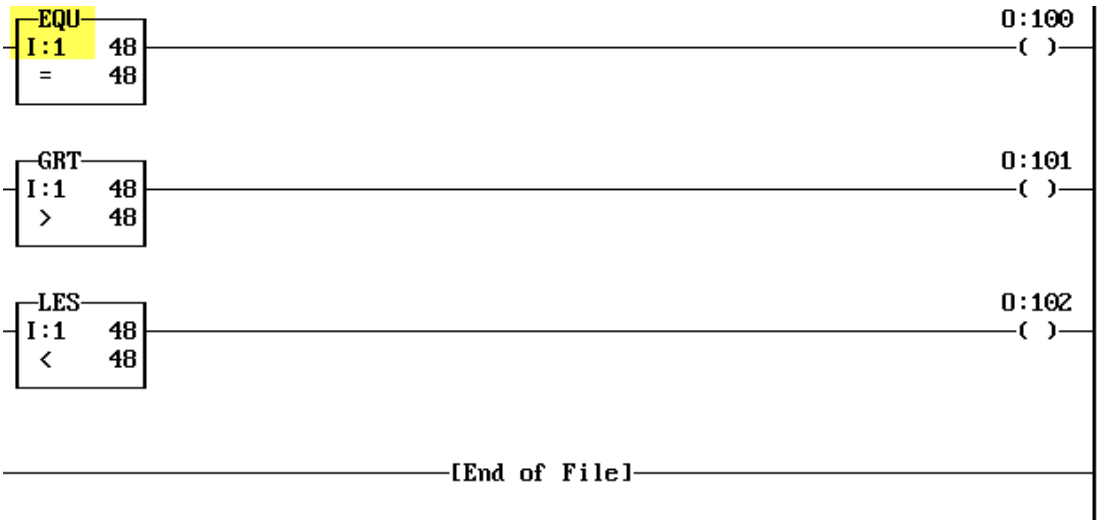
Η εντολή GRT συγκρίνει μια λογική διεύθυνση με έναν αριθμό. Αν η τιμή της λογική διεύθυνσης είναι μεγαλύτερη από τον αριθμό που έδωσε ο προγραμματιστής, τότε η GRT είναι αληθής, αλλιώς είναι ψευδής

- Greater than or Equal (GEQ)



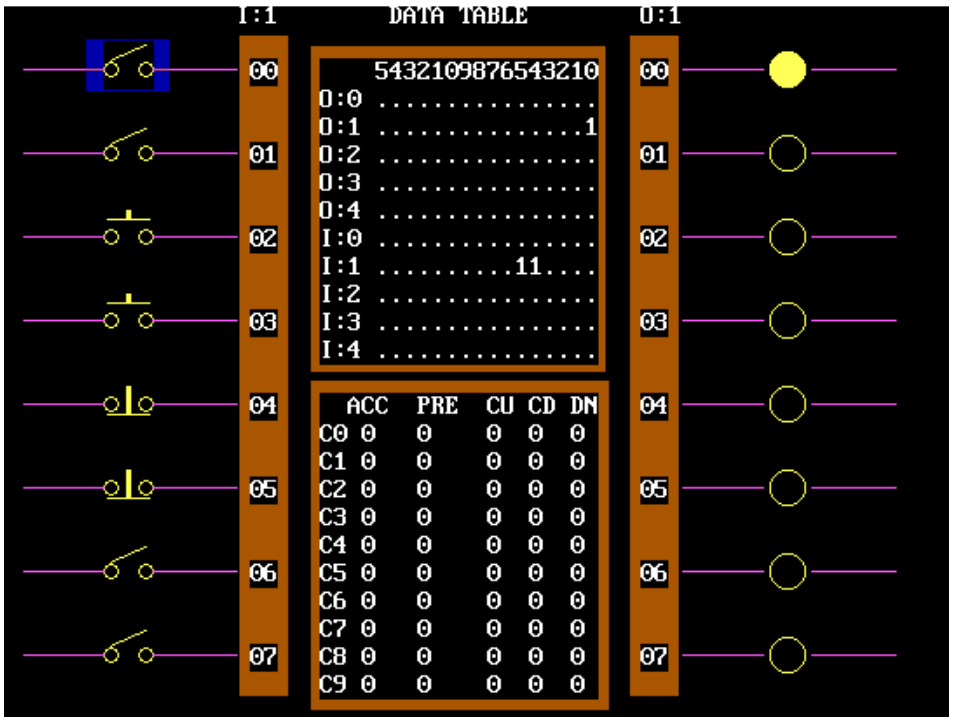
Όμοια εντολή με την GRT, με την διαφορά ότι αν οι αριθμοί είναι ίσοι, η GEQ είναι αληθής.

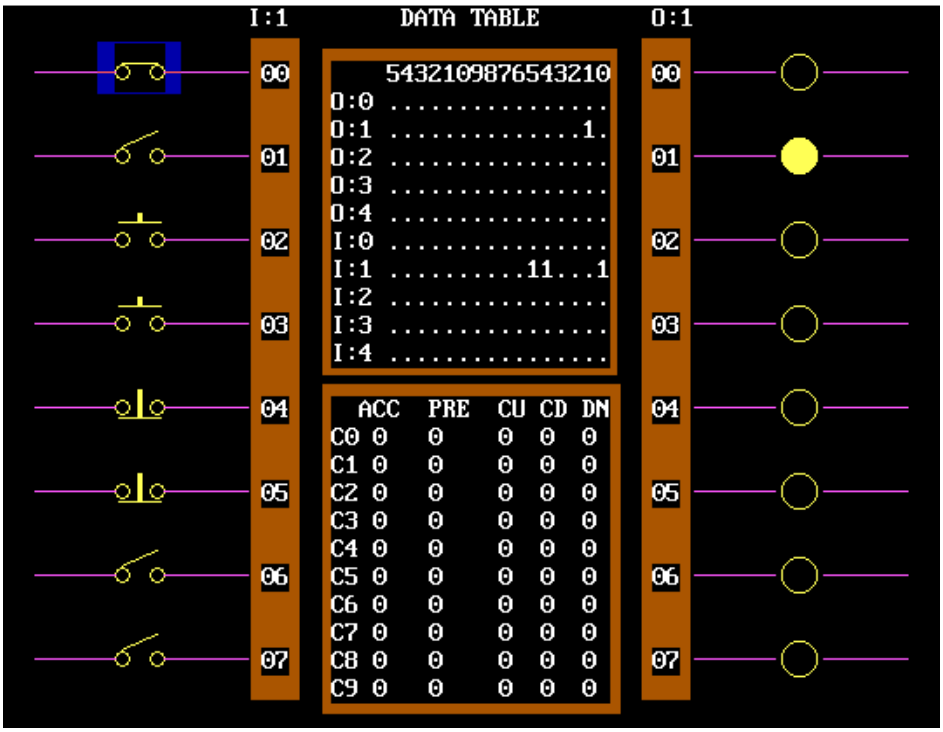
Παράδειγμα λογικών πράξεων



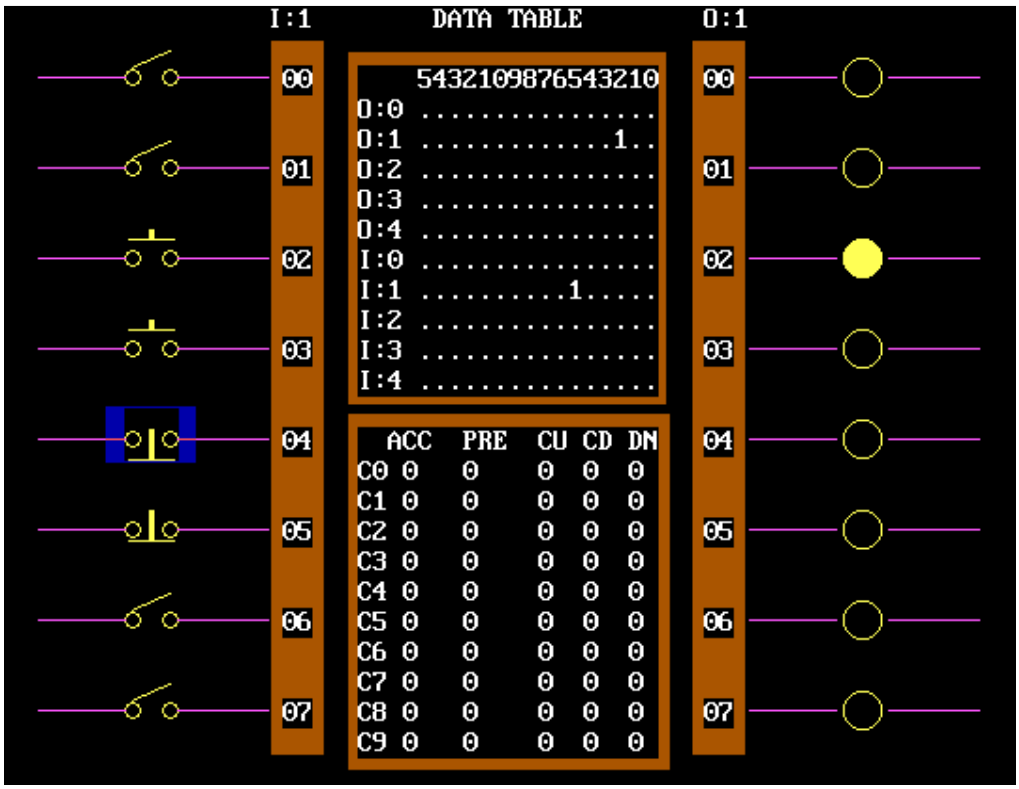
Σχήμα 3.10: Παράδειγμα λογικών πράξεων

Στο Σχήμα 3.10, αν δεν πειράξουμε κάποια είσοδο, η τιμή του I:1 είναι 48(0000000000110000) . Όσο η τιμή I:1 είναι 48, το λαμπάκι O:100 θα είναι αναμμένο. Όσο η τιμή του I:1 είναι μικρότερη του 48, το λαμπάκι O:102 θα είναι αναμμένο. Όσο η τιμή I:1 είναι μεγαλύτερη του 48, το λαμπάκι O:101 θα είναι αναμμένο.





Πατώντας τον διακόπτη I:1/00 η τιμή του I:1 από 48(0000000000110000) γίνεται 49(0000000000110001). Αφού 49>48 ανάβει το λαμπάκι O:1/01.



Ανοίγοντας τον διακόπτη I:1/04 η τιμή του I:1 από 48(0000000000110000) γίνεται 32(0000000000100000). Αφού 32<48 ανάβει το λαμπάκι O:1/02.

3.2 Βασικός Χειρισμός

Το πρόγραμμα PSIM έχει σχεδιαστεί με τέτοιο τρόπο ώστε να ζητάει όσο δυνατόν λιγότερη πληκτρολόγηση από τον χρήστη, για αυτό τον λόγο οι περισσότερες εντολές του χρησιμοποιούνται απλά πατώντας κάποιο από τα πλήκτρα F1 μέχρι F10. Για να επιταχθεί αυτό, σε κάθε οθόνη το PSIM αλλάζει τον ορισμό των πλήκτρων Fn, για παράδειγμα πατώντας το πλήκτρο F1 στην αρχική οθόνη του “I/O Simulation” ο χρήστης ανοιγοκλείνει διακόπτες, ενώ πατώντας το ίδιο πλήκτρο στην οθόνη προγραμματισμού “Rung Editor” ο χρήστης προσθέτει μια γραμμή στο πρόγραμμα. Ο ορισμός του κάθε κουμπιού Fn βρίσκετε στο κάτω μέρος της οθόνης.

3.3 Γενικοί Ορισμοί

Στις περισσότερες προσομοιώσεις του PSIM, με λίγες εξαιρέσεις, οι ορισμοί των πλήκτρων Fn κατά την διάρκεια της προσομοίωσης είναι οι εξής:

Τα πλήκτρα F1,F2 και F3 αντιστοιχούν σε διακόπτες NO (Normally Open – Κανονικά ανοιχτοί) και NC (Normally Closed – Κανονικά Κλειστοί) τους οποίους ο χρήστης μπορεί χειριστεί, και να τους συσχετίσει στην οθόνη προγραμματισμού Rung Editor για να ελέγχει την συμπεριφορά εξόδων.

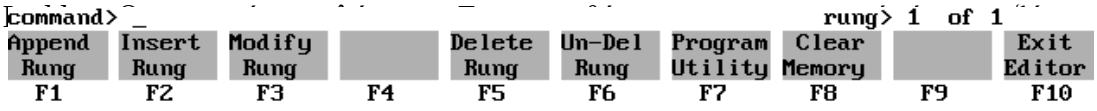
Με το πλήκτρο F5 ο χρήστης αλλάζει από την οθόνη της προσομοίωσης, στην οθόνη προγραμματισμού, στην οποία μπορεί να φτιάξει τους συσχετισμούς εισόδων και εξόδων στη γλώσσα Ladder.

Το πλήκτρο F6 γυρνάει την χρήστη στο βασικό μενού.

Το πλήκτρο F7 αλλάζει το κομμάτι του πίνακα δεδομένων του PLC που βλέπει ο χρήστης δίπλα από την προσομοίωση.

3.4 Rung Editor

Στην οθόνη προγραμματισμού Rung Editor ο χρήστης προγραμματίζει τους συσχετισμούς εισόδου και εξόδου της προσομοίωση χρησιμοποιώντας την γλώσσα προγραμματισμού



Με τα πλήκτρα F1 και F2 ο χρήστης μπορεί να προσθέσει μια γραμμή για προγραμματισμό, μετά την θέση του κέρσορα, ή πριν την θέση του κέρσορα αντίστοιχα.

Με το πλήκτρο F3 ο χρήστης μπορεί να επεξεργαστεί την γραμμή του προγράμματος στην οποία είναι πάνω ο κέρσορας.

Με το πλήκτρο F5 ο χρήστης μπορεί να διαγράψει την γραμμή του προγράμματος στην οποία είναι πάνω ο κέρσορας.

Με το πλήκτρο F6 ο χρήστης μπορεί να επιστρέψει μια γραμμή του προγράμματος που είχε διαγράψει με το πλήκτρο F5. Σε συνδυασμό με το πλήκτρο F5 ο χρήστης μπορεί να μετακινεί γραμμές κώδικα, διαγράφοντας τες και επαναφέροντας τες στο σημείο που θέλει.

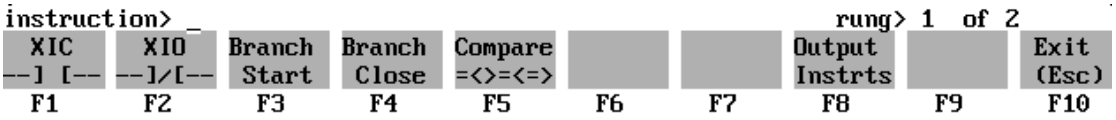
Το πλήκτρο F7 εμφανίζει το menu αποθήκευσης και επαναφοράς του προγράμματος.

Το πλήκτρο F8 διαγράφει τις τιμές που έχουν αποθηκευτεί στον πίνακα δεδομένων του προγράμματος .

Με το F10 ο χρήστης επιστρέφει στην οθόνη της προσομοίωσης.

Μενού Εντολών Εισόδου

Όταν ο χρήστης προσθέτει μια γραμμή στο πρόγραμμα, εμφανίζετε το μενού εντολών εισόδου τα πλήκτρα Fn παίρνουν τους εξής ορισμούς.



Με τα πλήκτρα F1 και F2 ο χρήστης μπορεί να προσθέσει στην γραμμή τις εντολές XIC (Examine if closed) και XIO (Examine if Open) της γλώσσας Ladder.

Με τα πλήκτρα F3 και F4 ο χρήστης μπορεί να φτιάξει παρακλάδια στην γραμμή, για να δώσει παράλληλες εντολές στο πρόγραμμα. Με τα παρακλάδια σχηματίζετε το λογικό OR στην γλώσσα Ladder.

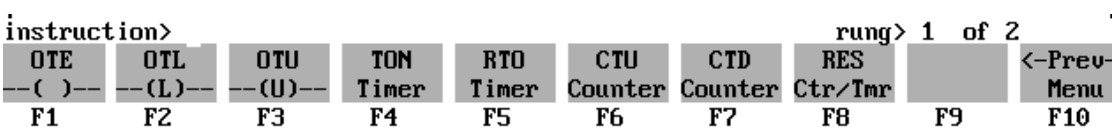
Με το πλήκτρο F5 ο χρήστης μπορεί να προσθέσει εντολές σύγκρισης (εμφανίζετε το μενού σύγκρισης)

Με το πλήκτρο F8 ο χρήστης μπορεί να προσθέσει εντολές εξόδου στην γραμμή. (εμφανίζετε το μενού εντολών εξόδου)

Με το πλήκτρο F10 ο χρήστης επιστρέφει στο βασικό μενού του Rung Editor

Μενου Εντολών Εξόδου

Όταν ο χρήστης επιλέξει να προσθέσει μια εντολή εξόδου στην γραμμή, το εμφανίζετε το μενού εντολών εξόδου και τα πλήκτρα Fn παίρνουν τους εξής ορισμούς.



Με το πλήκτρο F1 προστίθεται η εντολή εξόδου OTE (Output Energize) στην γραμμή, η οποία ενεργοποιεί την έξοδο αν η γραμμή είναι αληθής.

Με τα πλήκτρα F2 και F3 προστίθεται η εντολή εξόδου OTL (Output Latch) ή η εντολή OTU (Output Unlatch) αντίστοιχα

Με το πλήκτρο F4 προστίθεται η εντολή εξόδου TON (Timer On-delay)

Με το πλήκτρο F5 προστίθεται μια εντολή εξόδου RTO(Retentive Timer).

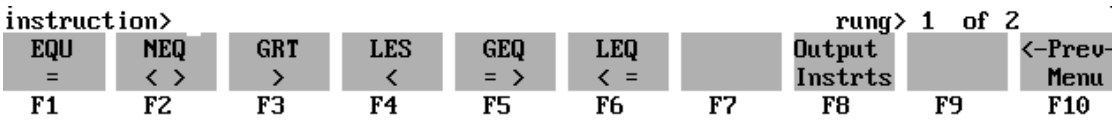
Με τα πλήκτρα F6 και F7 προστίθενται οι εντολές εξόδου CTU (Count Up Counter) και CTD (Count Down Counter) αντίστοιχα.

Με το πλήκτρο F8 προστίθεται μια εντολή εξόδου RES (Reset).

Με το πλήκτρο F10 γυρνάει στο μενού εντολών εισόδου.

Μενού Σύγκρισης

Όταν ο χρήστης επιλέξει να χρησιμοποιήσει μια εντολή σύγκρισης, τότε εμφανίζετε το μενού σύγκρισης, και τα πλήκτρα Fn παίρνουν τους παρακάτω ορισμούς:



Με τα πλήκτρο F1 και F2 προστεθέντα οι εντολές σύγκρισης EQU, η οποία ελέγχει αν μια τιμή Word είναι ίση με κάποια τιμή που διαλέγει ο προγραμματιστής, και NEQ η οποία είναι η αντίθετη της EQU.

Με τα πλήκτρο F3 και F5 προστεθέντα οι εντολές σύγκρισης GRT, η οποία ελέγχει αν μια τιμή Word είναι μεγαλύτερη από κάποια τιμή που διαλέγει ο προγραμματιστής, και GEQ η οποία ελέγχει αν η τιμή είναι μεγαλύτερη ή ίση.

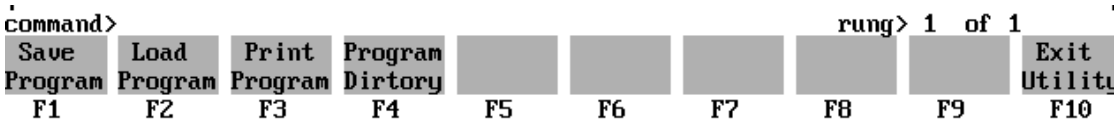
Με τα πλήκτρο F4 και F6 προστεθέντα οι εντολές σύγκρισης LES, η οποία ελέγχει αν μια τιμή Word είναι μικρότερη από κάποια τιμή που διαλέγει ο προγραμματιστής, και LEQ η οποία ελέγχει αν η τιμή είναι μικρότερη ή ίση.

Με το πλήκτρο F8 ο χρήστης μπορεί να προσθέσει εντολές εξόδου στην γραμμή. (εμφανίζετε το μενού εντολών εξόδου)

Με το πλήκτρο F10 γυρνάει στο μενού εντολών εισόδου.

Μενού Αποθήκευσης (Utility Menu)

Το μενού αποθήκευσης έχει λειτουργίες για να μπορεί ο χρήστης να αποθηκεύει και να επαναφέρει το πρόγραμμα του. Στο μενού αποθήκευσης τα πλήκτρα Fn έχουν τους ορισμούς:



Με το πλήκτρο F1 γίνεται η αποθήκευση του προγράμματος. Το PSIM δίνει μόνο του ονόματα στα αρχεία που αποθηκεύονται. Τα αρχεία αποθηκεύονται σε κάποιο προεπιλεγμένο φάκελο

Με το πλήκτρο F2 γίνετε η επαναφορά ενός αποθηκευμένου προγράμματος

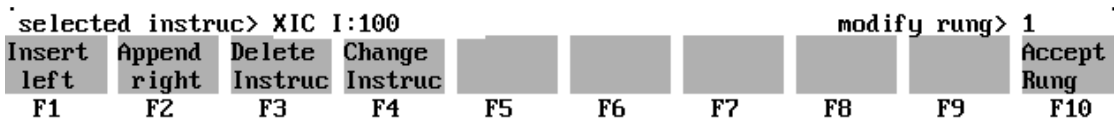
Το πλήκτρο F3 εκτυπώνει το πρόγραμμα στη συσκευή PRN του DOS

Το πλήκτρο F4 δείχνει τον φάκελο με τα αποθηκευμένα προγράμματα.

Με το πλήκτρο F10 ο χρήστης επιστρέφει στο βασικό μενού του Rung Editor

3.5 Μενού Μορφοποίησης Γραμμής

Όταν ο χρήστης αρχίσει την επεξεργασία μιας γραμμής του προγράμματος, εμφανίζετε το μενού μορφοποίησης γραμμής. Με το πληκτρολόγιο ο χρήστης μπορεί να κουνήσει τον κέρσορα πάνω από την εντολή της γραμμής που θέλει να αλλάξει. Τα πλήκτρα Fn παίρνουν τους παρακάτω ορισμούς



Με το πλήκτρο F1 και F2 ο χρήστης μπορεί να προσθέσει μια εντολή στην γραμμή, δεξιά και αριστερά του κέρσορα αντίστοιχα.

Με το πλήκτρο F3 ο χρήστης διαγράφει την εντολή στην οποία βρίσκετε ο κέρσορας.

Με το πλήκτρο F4 ο χρήστης αλλάζει την εντολή στην οποία βρίσκετε ο κέρσορας με κάποια άλλη.

Με το πλήκτρο F10 τελειώνει η μορφοποίηση της γραμμής, και ο χρήστης γυρνάει στο βασικό μενού του Rung Editor.

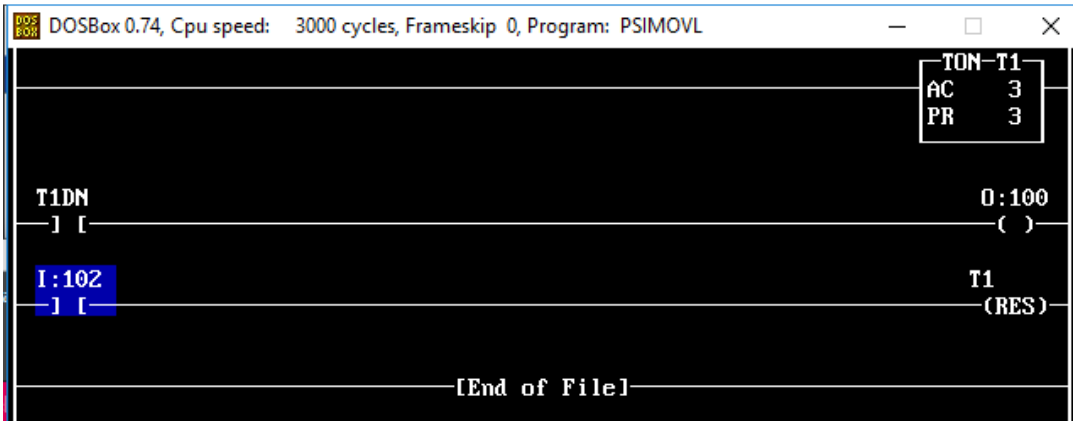
3.6 Παραδείγματα

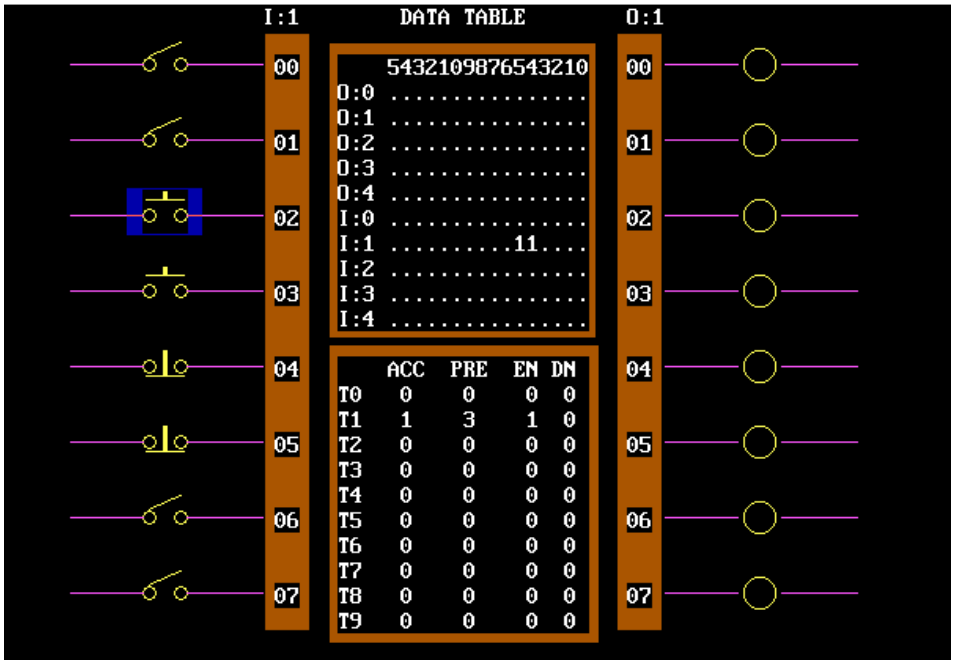
1. I/O SIMULATOR

- **TIMERS**

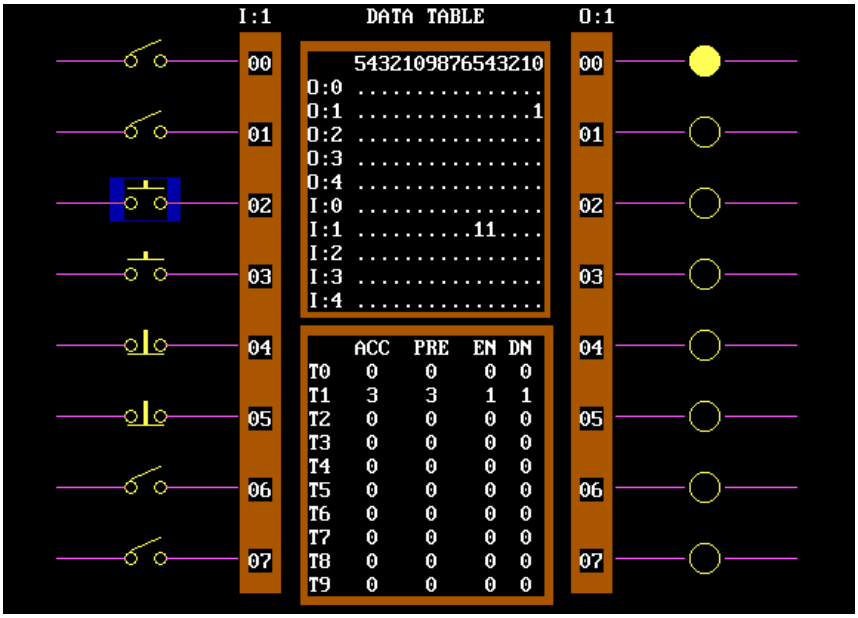
Θα προγραμματίσουμε το PLC έτσι ώστε μετά από 3 δευτερόλεπτα να ανάβει ένα λαμπάκι, και με το πάτημα ενός κουμπιού να γίνετε Reset.

Θα χρησιμοποιήσουμε ένα TON timer T1. Κάθε φορά που το T1 μετράει μέχρι το 3, το T1:DN θα γίνετε 1. Όταν το T1:DN είναι 1 θα ανάβει το λαμπάκι O:1/00. Όταν πατήσουμε το κουμπί I:1/02 ο T1 θα ξεκινάει από την αρχή





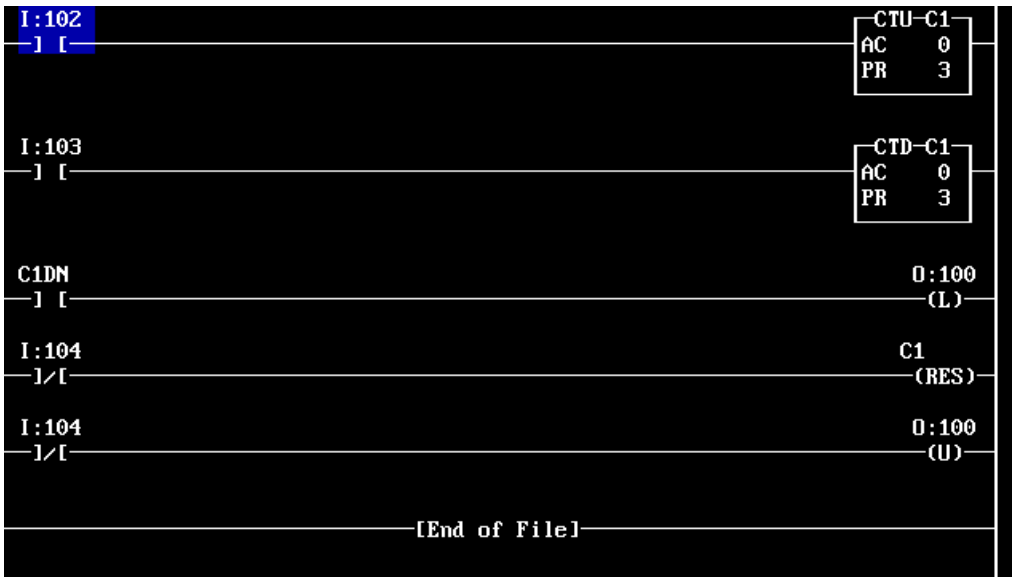
Το T1 μετράει μέχρι να φτάσει στην τιμή T1:PR

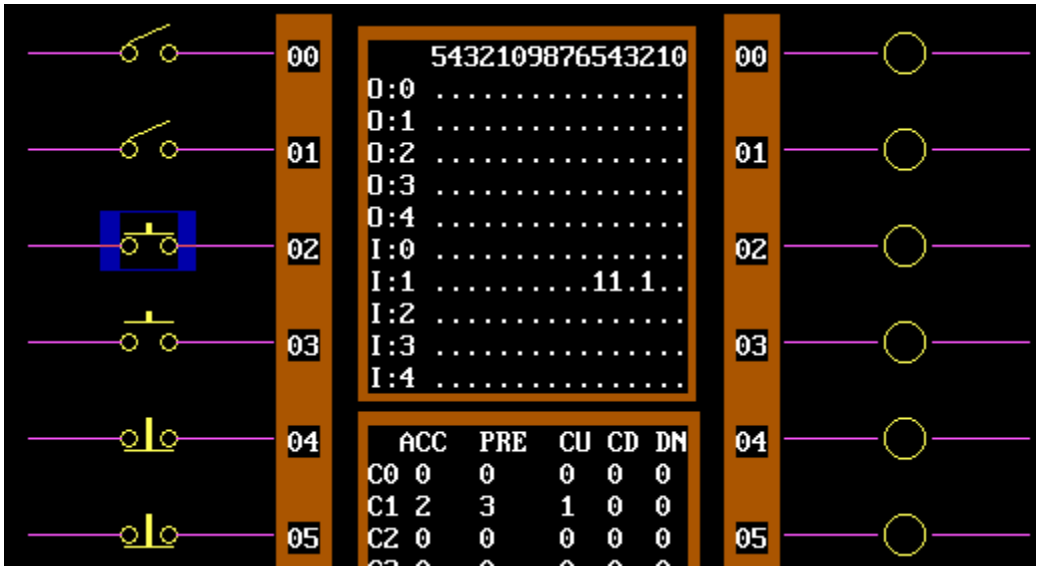


Μόλις φτάσει την τιμή T1:PR, το T1:DN γίνετε 1 για να μας δείξει οτι ο Timer έφτασε τον στόχο του. Το λαμπάκι ανάβει όταν το T1:DN γίνει 1 όπως το προγραμματίσαμε. Αν πατήσουμε το κουμπί I:1/02 ο μετρητής θα αρχίσει απο την αρχή

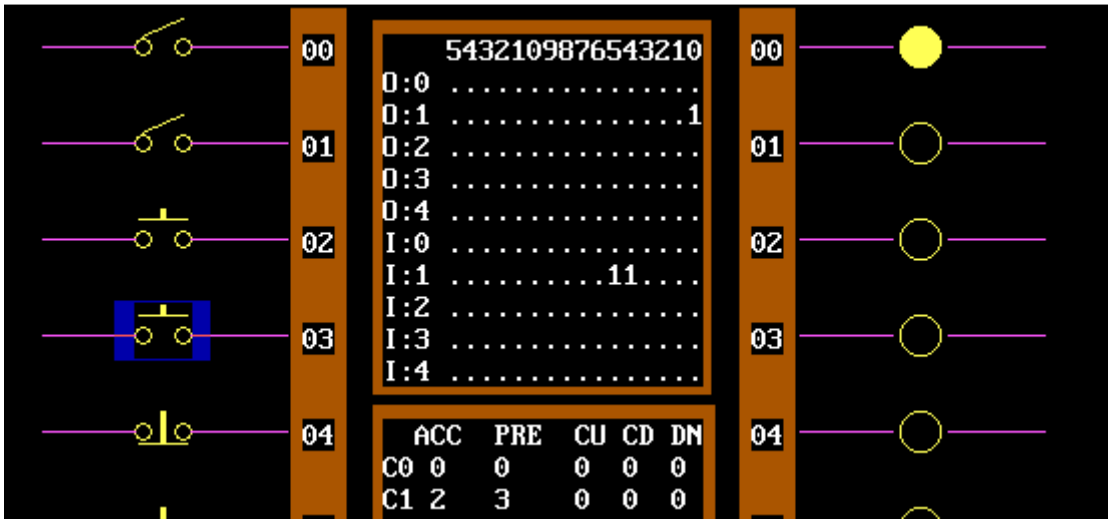
- **Counters και Latches**

Θα φτιάξουμε ένα counter, που αν πατήσουμε ένα κουμπί θα αυξάνετε κατά 1, με ένα άλλο κουμπί θα μειώνεται κατά 1, και αν φτάσει στον αριθμό 3, θα κάνει Latch μια έξοδο. Αν πατήσουμε ένα τρίτο κουμπί η έξοδος θα γίνεται unlatch και το counter θα γίνεται reset.

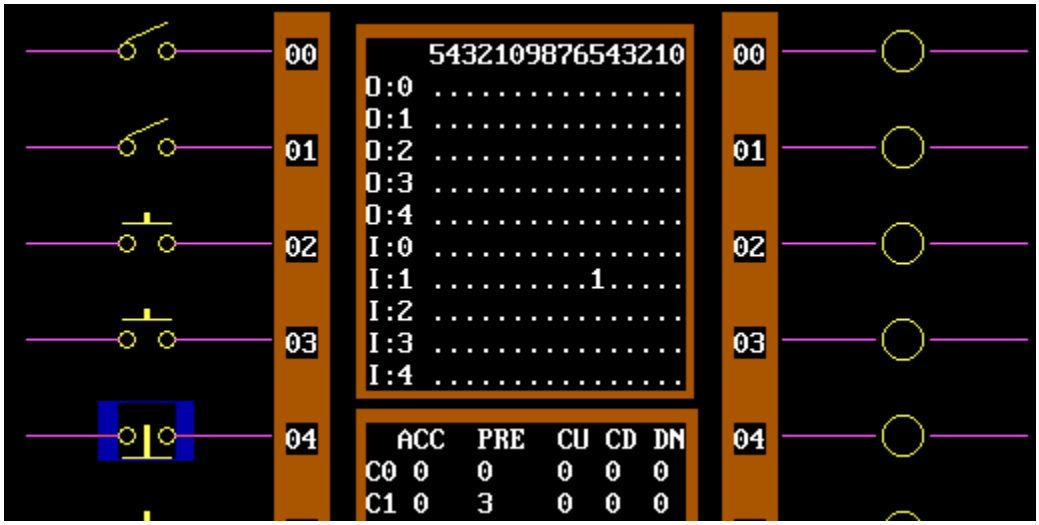




Κάθε φορά που πατάμε το I:1/02 ο C1 αυξάνεται κατά 1.

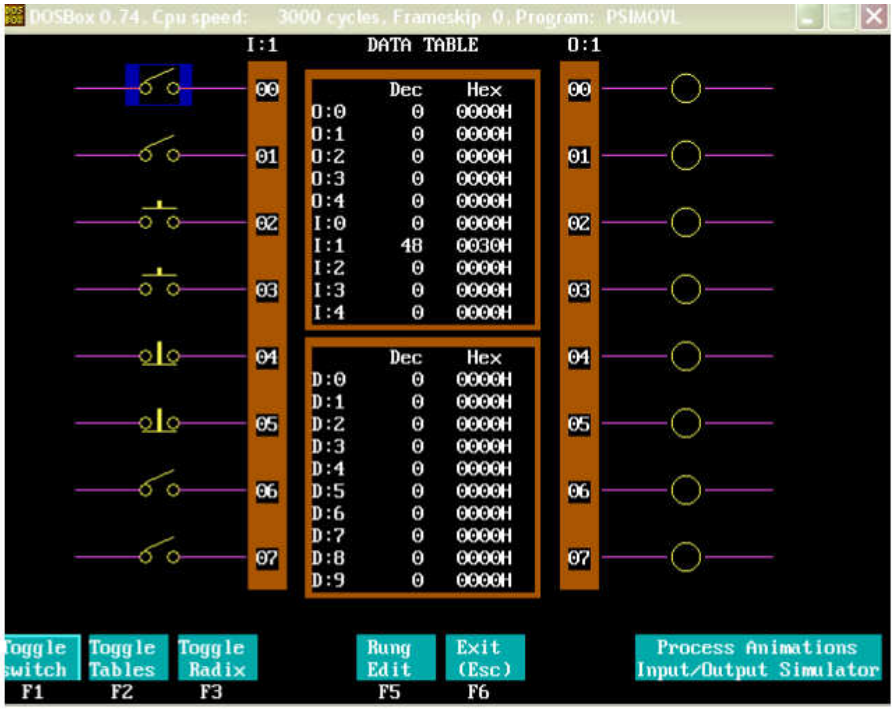


Αφού ο C1 φτάσει το 3, και το C1:DN γίνει 1, το λαμπάκι O:1/00 θα ανάψει και δεν θα σβήσει ακόμα και αν κάνουμε το C1:DN πάλι 0



Ο μόνος τρόπος για να σβήσει το O:1/00 είναι να πατήσουμε το κουμπί I:1/04 για να το κάνει Unlatch.

- Θα φτιάξουμε ένα πρόγραμμα με το οποίο, αν ο διακόπτης I:1/00 είναι κλειστός για 10 χρονικές περιόδους, ή αν το κουμπί I:1/02 πατηθεί 10 φόρες, το λαμπάκι O:1/07 θα ανάψει και θα μείνει ανοιχτό, μέχρι να πατηθεί το κουμπί I1:/03 που θα σβήσει το λαμπάκι και θα κάνει reset τους μετρητές



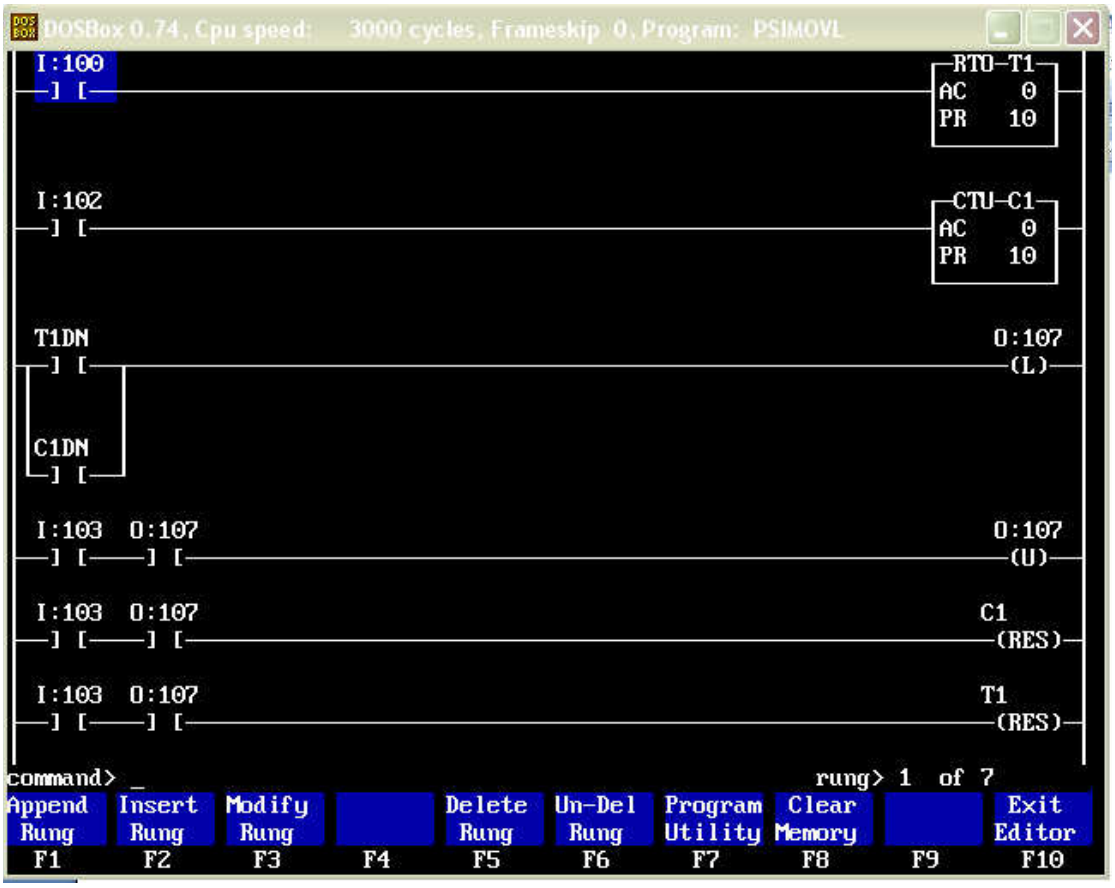
Θέλουμε όταν πατάμε το I:1/00 να ξεκινάει μια χρονομέτρηση. Όταν έχει μείνει πατημένο για 10 δευτερόλεπτα θα ανάβει το O:1/07. Για αυτό θα χρησιμοποιήσουμε ένα Retentive

Timer (δεν χρησιμοποιούμε TON επειδή θέλουμε όταν ανοίγει ο διακόπτης, να θυμάται ο χρονομετρητής για πόσα δευτερόλεπτα ήταν κλειστό το I:1/00)

Επίσης θέλουμε όταν πατήσουμε το I:1/02 10 φορές, να ανάβει το O:1/07. Για αυτό θα χρησιμοποιήσουμε ένα μετρητή Count Up Counter.

Θέλουμε το O:1/07 αφού να ανάψει να μείνει αναμμένο, άρα θα χρησιμοποιήσουμε ένα Output Latch, και τέλος όταν πατάμε το κουμπί I:1/03 θέλουμε να σβήνει το λαμπάκι και να γίνονται reset οι μετρητές, άρα θα βάλουμε ένα Output Unlatch και ένα reset για τον κάθε μετρητή.

Πηγαίνουμε στην οθόνη προγραμματισμού (F5) για να φτιάξουμε το πρόγραμμα

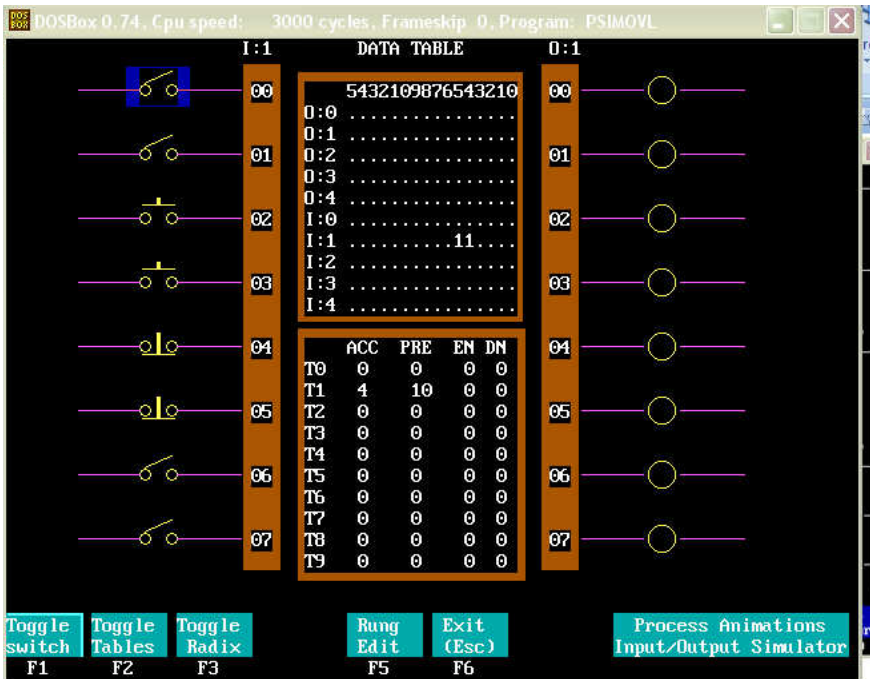


Αν το I:1/00 μείνει πατημένο για 10 δευτερόλεπτα, το ACC του T1 θα γίνει 10, άρα το T1DN θα γίνει αληθές και το O:1/07 θα ανάψει.

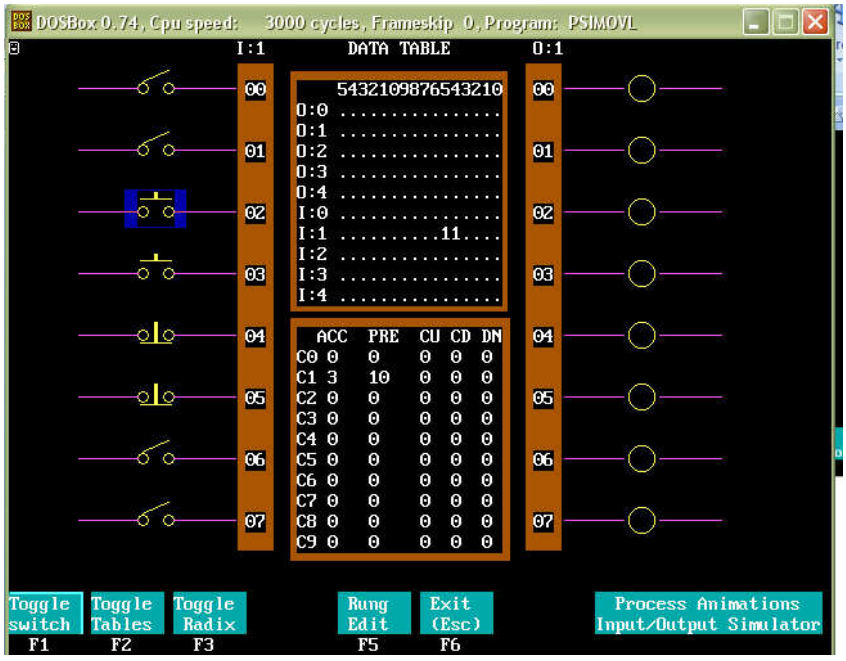
Αν το I:1/02 πατηθεί 10 φορές, το ACC του C1 θα γίνει 10, άρα το C1DN θα γίνει αληθές και το O:1/07 θα ανάψει.

Αν πατηθεί το I:1/03 και το O:1/07 είναι ανοιχτό, τότε το O:1/07 θα σβήσει και τα T1 και C1 θα γίνουν Reset

Γυρνάμε στην προσομοίωση. Πατώντας F2 βλέπουμε τους πίνακες δεδομένων για τα Timer και τα Counter



Κλείνουμε τον διακόπτη I:1/00 για 4 δευτερόλεπτα και τον ανοίγουμε. Βλέπουμε οτι το T1:AC έχει πάρει την τιμή 4 και την κρατάει, έτσι ξέρουμε ότι ο διακόπτης I:1/00 έχει μείνει κλειστός συνολικά για 4 δευτερόλεπτα



Βλέπουμε οτι κάθε φορά που πατάμε τον διακόπτη I:1/02 η τιμή C1:AC αυξάνετε κατα 1

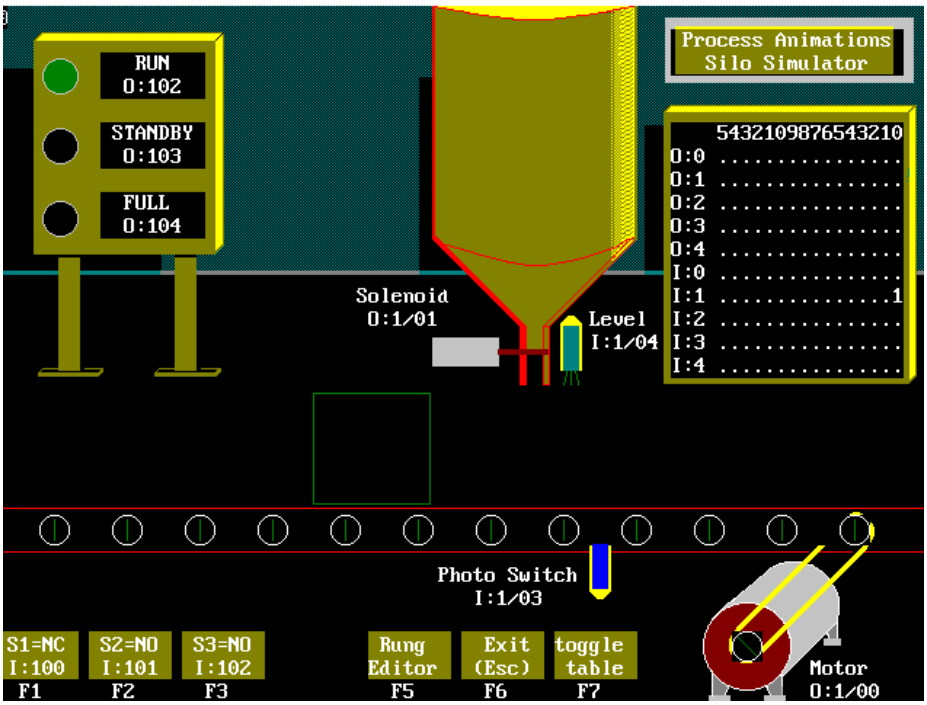
Όταν κάποιο από τα T1 ή C1 φτάσει το 10, το O:1/07 ανάβει και μένει αναμμένο. Αν το O:1/07 είναι αναμμένο και πατήσουμε το I:1/03 , το O:1/07 σβήνει (Unlatch) και τα T1 και C1 μηδενίζονται (Reset).

2. SILO SIMULATOR

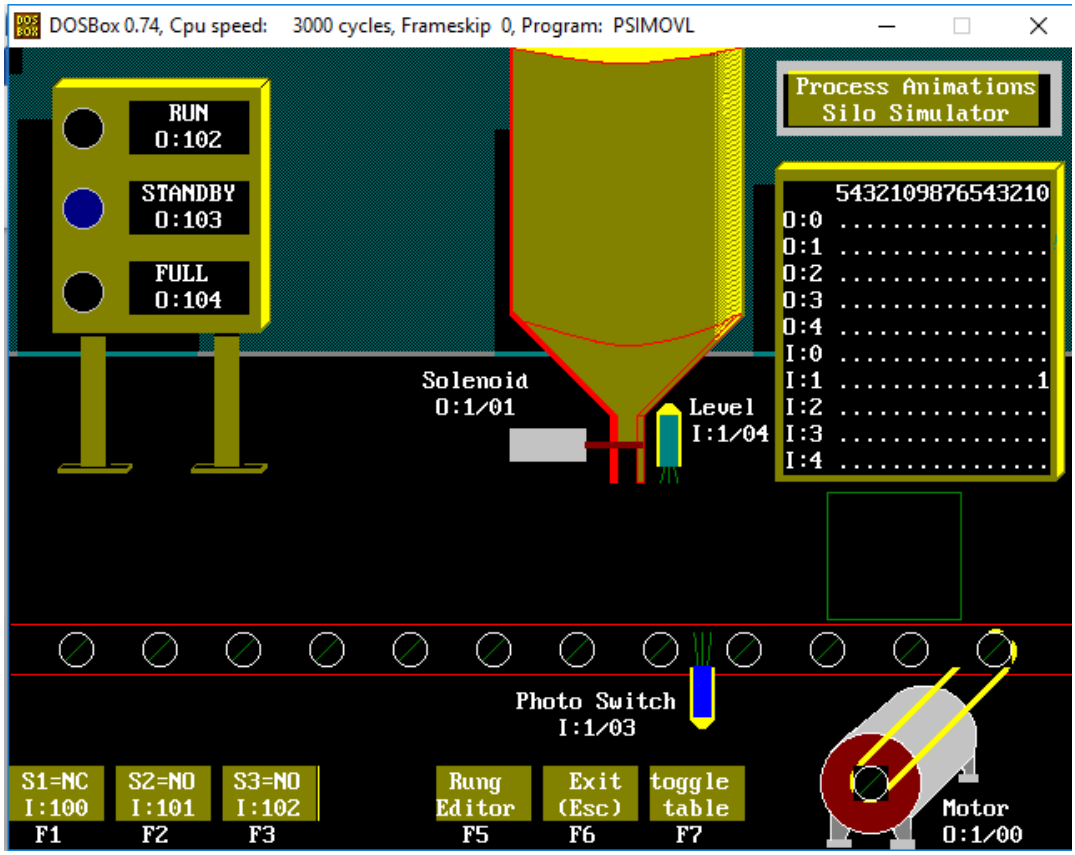
- **Κινητήρας**

Θέλουμε όταν πατάμε ένα κουμπί να ξεκινάει να δουλεύει ο κινητήρας του silo, και πατώντας ένα άλλο κουμπί να σταματάει. Τα λαμπάκια θα ανάβουν ανάλογα με την κατάσταση του κινητήρα

Ο κινητήρας είναι μια απλή έξοδος. Όταν πατάμε ένα κουμπί θα τον κάνουμε latch, και όταν πατάμε ένα άλλο κουμπί θα τον κάνουμε unlatch



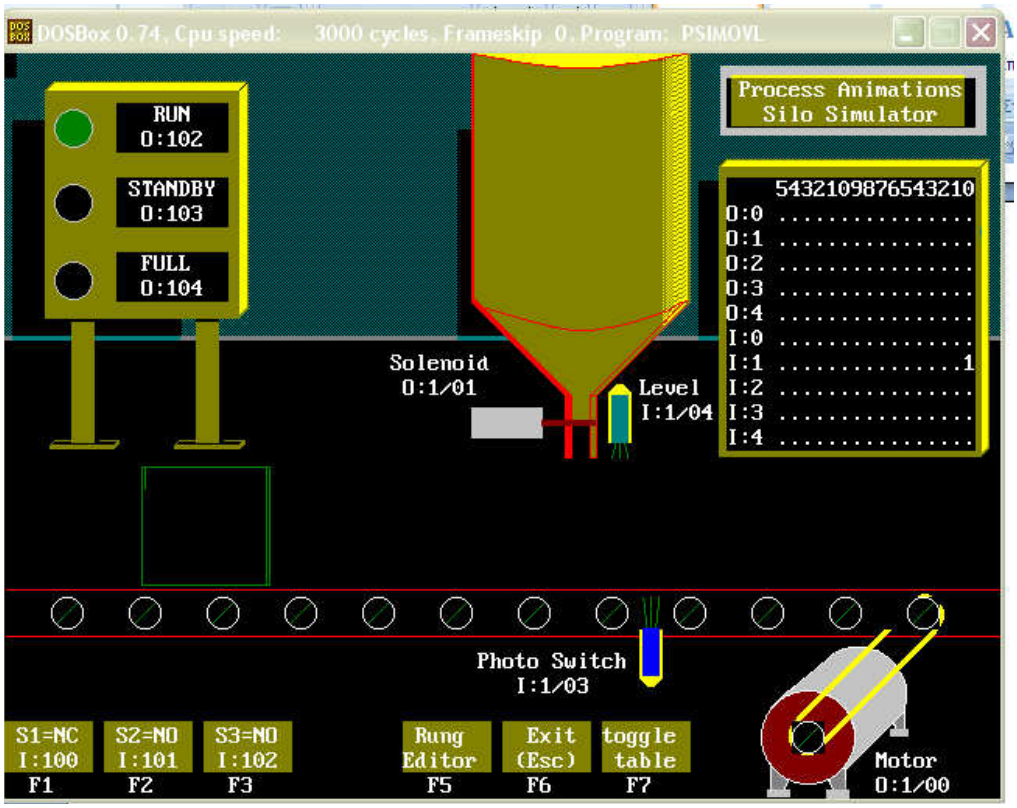
Όταν πατάμε το I:1/01 ο κινητήρας ξεκινάει να κινείται και το λαμπάκι Run ανάβει.



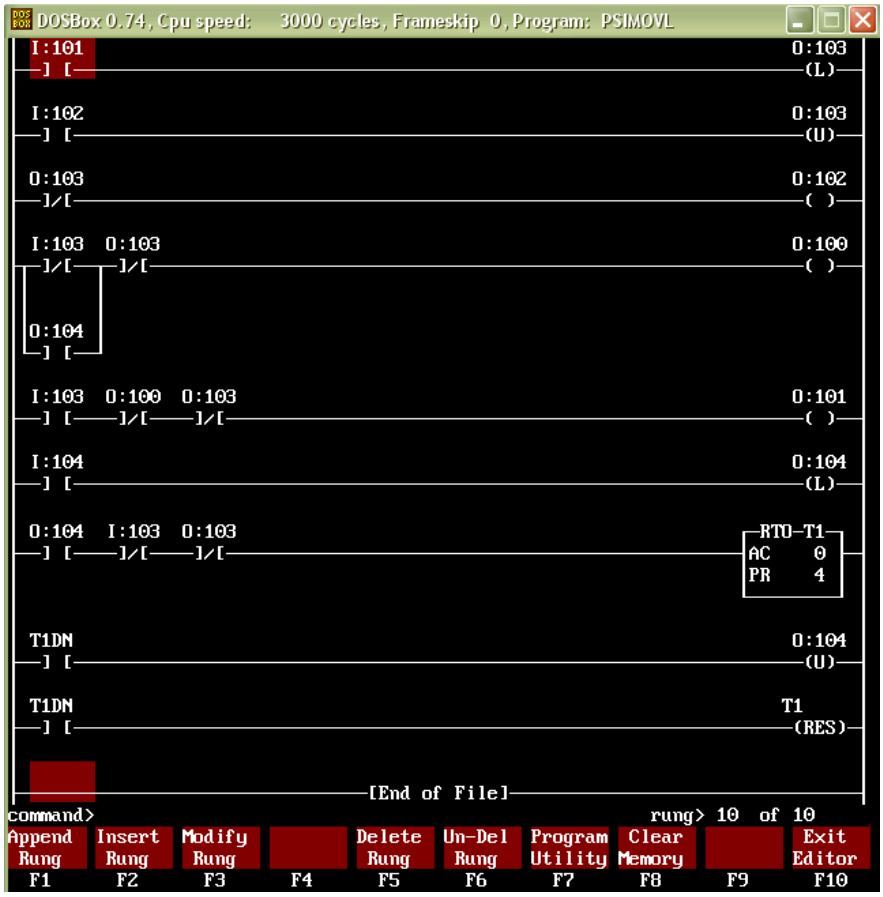
Όταν πατάμε το I:1/02 ο κινητήρας σταματάει να κινείται και το λαμπάκι Standby ανάβει.

Θέλουμε να φτιάξουμε ένα εργοστάσιο που θα γεμίζει κουτιά. Το κουτί θα πηγαίνει μέσω μοτέρ στην βαλβίδα, η οποία θα το γεμίζει. Όταν το κουτί γεμίζει η βαλβίδα κλείνει ,το κουτί φεύγει από το μοτέρ και έρχεται καινούργιο. Με έναν διακόπτη θα κάνουν παύση όλες οι διαδικασίες, και με έναν άλλο διακόπτη θα συνεχίζουν. Τα αντίστοιχα λαμπάκια θα είναι αναμμένα σε κάθε κατάσταση.

Θέλουμε ο κινητήρας O:1/00 να λειτουργεί μέχρι το κουτί να συναντήσει τον αισθητήρα I:1/03. Όταν φτάσει εκεί θα σταματήσει, η βαλβίδα O:1/01 θα ανοίξει και θα αρχίσει να γεμίζει το κουτί. Όταν γεμίσει, η βαλβίδα O:1/03 θα κλείσει, το λαμπάκι O:1/04 θα ανάψει και ο κινητήρας O:1/00 θα αρχίσει να λειτουργεί. Όταν το κουτί φύγει , το λαμπάκι O:1/04 θα σβήσει και θα έρθει ένα καινούργιο κουτί. Επίσης όσο όλα λειτουργούν το λαμπάκι O:1/02 θα είναι αναμμένο. Όταν πατηθεί το κουμπί I:1/01 θα σταματήσουν όλες οι λειτουργίες και θα ανάψει το λαμπάκι O:1/02. Όταν πατηθεί το κουμπί I:1/02 θα συνεχίσουν όλες οι λειτουργίες .



Πηγαίνουμε στην οθόνη προγραμματισμού (F5) για να φτιάξουμε το πρόγραμμα.



Όταν πατήσουμε το κουμπί I:1/01, το λαμπάκι O:1/03 Stand by γίνεται Latch.

Όταν πατάμε το κουμπί I:1/02 το λαμπάκι O:1/03 Stand by γίνεται Unlatch.

Όταν το λαμπάκι O:1/03 Standby είναι κλειστό, το λαμπάκι O:1/02 Run, είναι αναμμένο.

Όσο ο αισθητήρας I:1/03 δεν έχει δει το κουτί, ή αν ο αισθητήρας I:1/04 λέει ότι το κουτί είναι γεμάτο, και εφόσον το λαμπάκι O:1/03 Standby είναι κλειστό, λειτουργεί ο κινητήρας O:1/00

Όσο ο αισθητήρας I:1/03 βλέπει το κουτί, ο κινητήρας O:1/00 δεν λειτουργεί και το λαμπάκι O:1/03 Standby δεν είναι αναμμένο, η βαλβίδα O:1/01 ανοίγει και το κουτί γεμίζει.

Όταν ο αισθητήρας I:1/04 δει ότι το κουτί γέμισε, το λαμπάκι O:1/04 full γίνεται Latch.

Όσο το λαμπάκι O:1/04 Full είναι αναμμένο, και ο αισθητήρας I:1/03 δεν βλέπει το κουτί, και το λαμπάκι O:1/03 Standby είναι κλειστό, αρχίζει μια χρονομέτρηση μέχρι το 4. Αυτό το κάνουμε επειδή από όταν φύγει το κουτί από τον αισθητήρα I:1/03 μέχρι να έρθει το επόμενο κουτί, περνάνε 4 δευτερόλεπτα, στα οποία το λαμπάκι Full θα ήταν κλειστό χωρίς αυτό τον Timer. Επίσης χρησιμοποιούμε RTO για την περίπτωση που ο χρήστης πατήσει το Standby πριν τελειώσει η χρονομέτρηση.

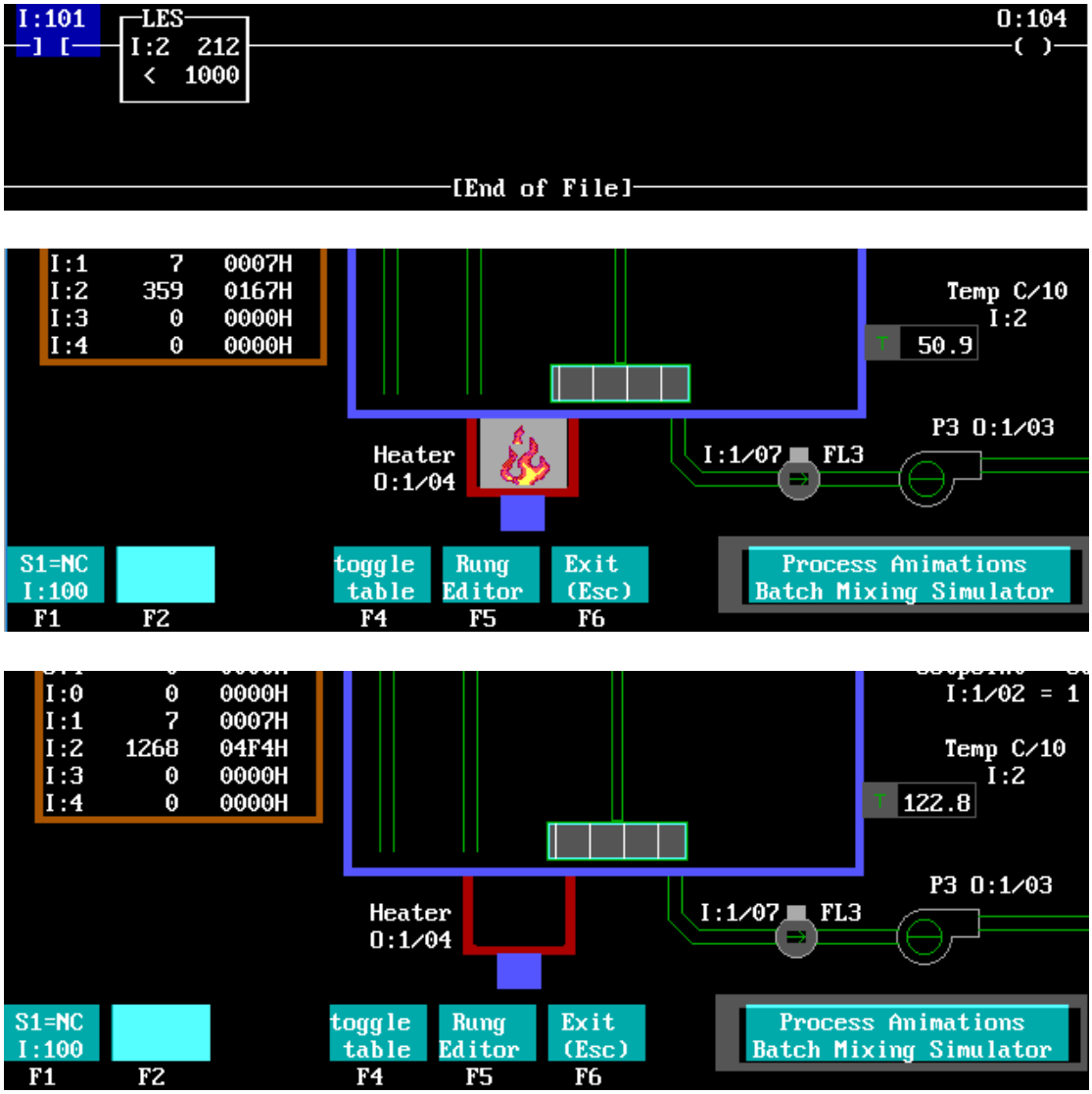
Όταν ο Timer φτάσει τα 4 δευτερόλεπτα, το λαμπάκι O:1/04 Full σβήνει και ο Timer γίνεται reset.

3. Batch Mixer

- **Θερμόμετρο**

Θέλουμε όταν πατάμε ενα κουμπί να ανάβει η φωτιά, αλλά αν η θερμοκρασία ξεπεράσει τους 100 βαθμούς, να μην την σβήνει το PLC.

Στη μεταβλητή I:2 είναι αποθηκευμένη η τιμή της θερμοκρασίας *10. Άρα θέλουμε να σταματάει η φωτιά όταν η τιμή της I:2 φτάσει τα 1000. Αυτό θα το κάνουμε με εντολές σύγκρισης.



Όταν η τιμή της I:2 υπερβεί τα 1000, παρόλο που πατάμε το κουμπί I:1/01 , η φωτιά δεν ανάβει.

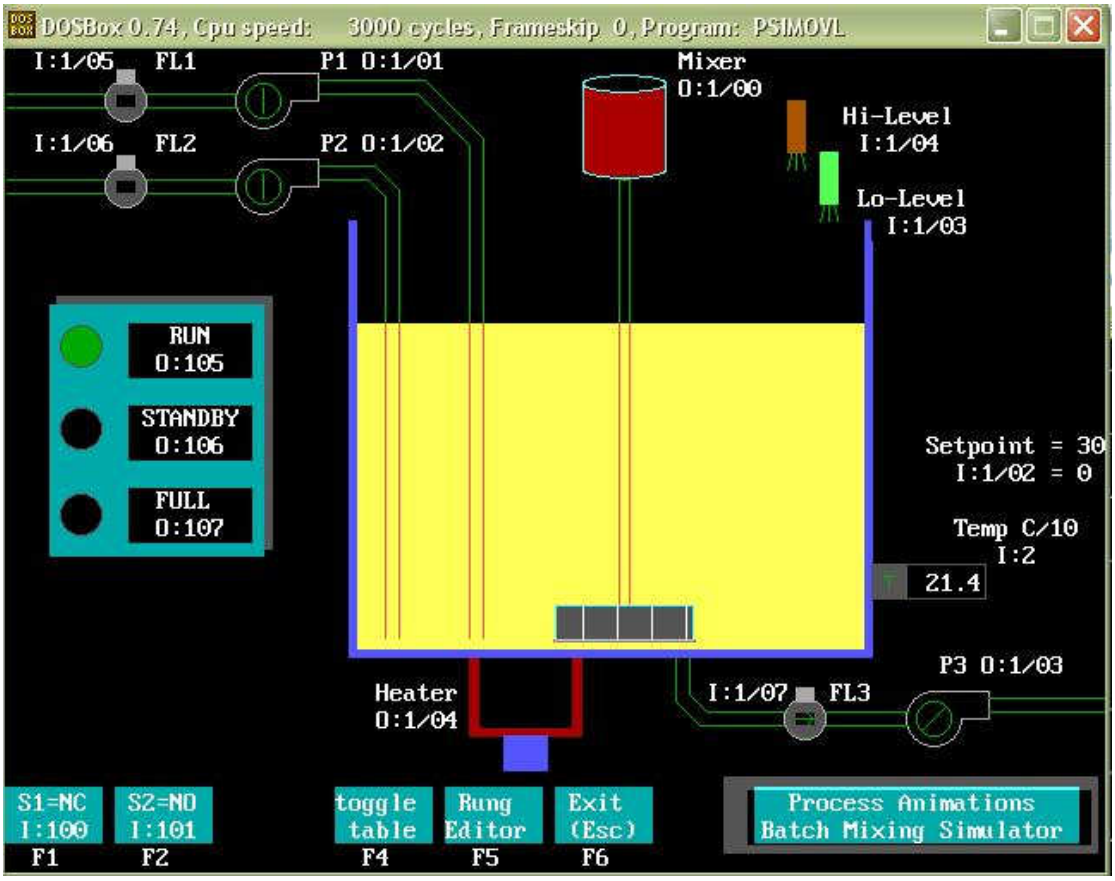
Θα φτιάξουμε ένα mixer το οποίο θα αναμιγνύει , θα θερμάνει το μίγμα μέχρι τους 40 βαθμούς κελσίου και θα το διώχνει. Η μίξη θα γίνετε από όταν ξεκινήσουν να μπαίνουν τα υλικά στην χύτρα, μέχρι το μίγμα να φτάσει τους 30 βαθμούς. Θα συνεχίσει να θερμάνετε μέχρι τους 40 και τότε θα σταματήσει και θα αδειάσει η χύτρα. Με έναν διακόπτη θα κάνουν παύση όλες οι διαδικασίες, και με έναν άλλο διακόπτη θα συνεχίζουν. Τα αντίστοιχα λαμπάκια θα είναι αναμμένα σε κάθε κατάσταση.

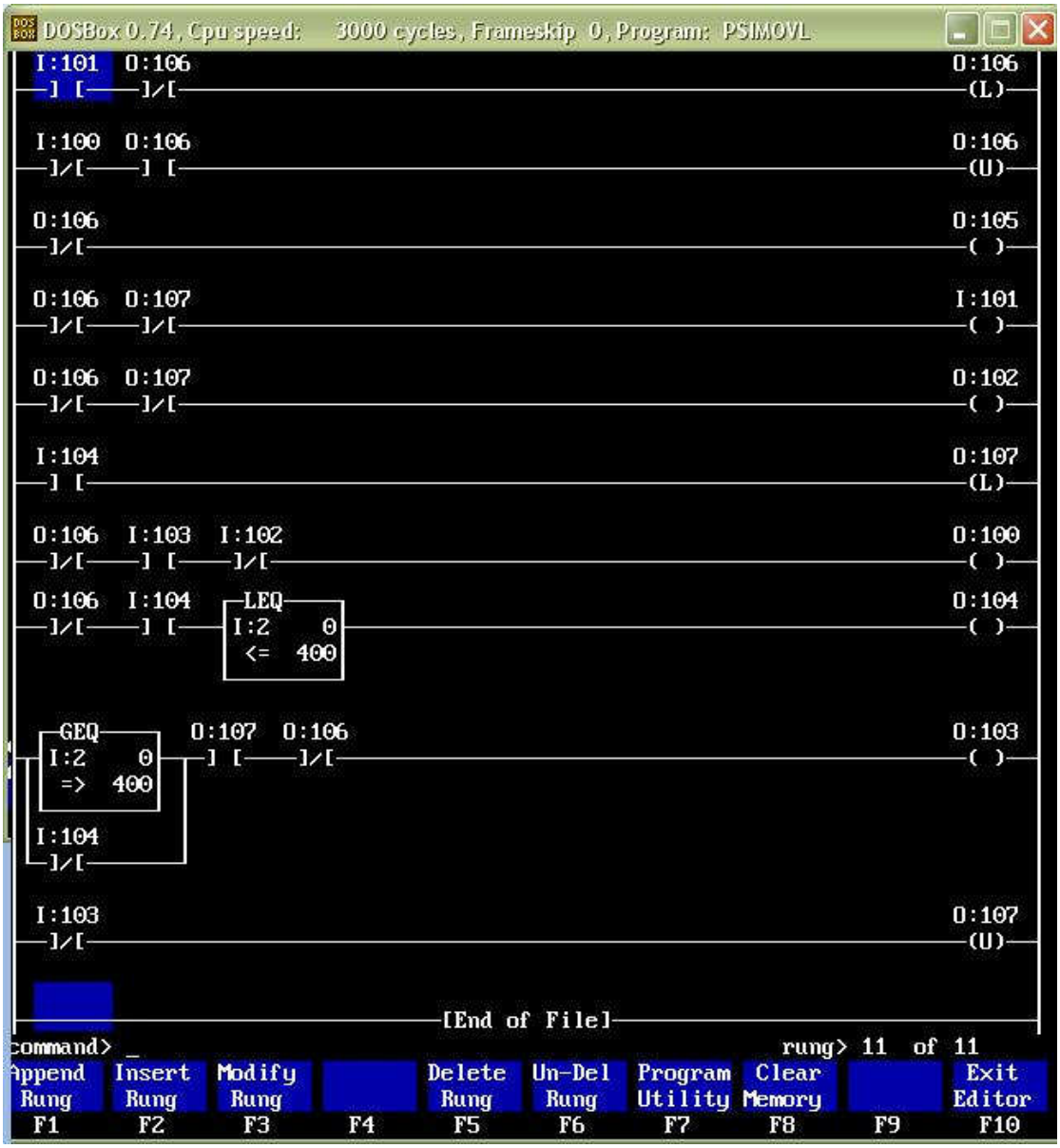
Θέλουμε οι κινητήρες O:1/01 και O:1/02 να γεμίζουν με την χύτρα μέχρι ο αισθητήρας I:1/04 να καταλάβει ότι η χύτρα γέμισε και να ανάψει το λαμπάκι O:1/07. Όταν ο αισθητήρας I:1/03 ανάψει, το μοτέρ O:1/00 θα αρχίσει να αναμιγνύει μέχρι ο αισθητήρας

I:1/02 καταλάβει ότι η χύτρα έφτασε στους 30 βαθμούς κελσίου, και τότε το μοτέρ θα σταματήσει.

Ο Θερμαντήρας O:1/04 θα αρχίσει να θερμάνει το μίγμα όταν γεμίσει η χύτρα, μέχρι η λογική διεύθυνση I:2 να πάρει την τιμή 400 (40 βαθμοί κελσίου). Όταν η χύτρα φτάσει στους 40 βαθμούς, ο κινητήρας O:1/03 θα αρχίσει να αδειάζει το μίγμα. Όταν ο αισθητήρας I:1/03 σταματήσει να είναι ενεργός, η χύτρα θα έχει αδειάσει και η διαδικασία θα αρχίσει ξανά από την αρχή.

Όταν πατηθεί το κουμπί I:1/01 σταματάνε όλες οι διαδικασίες και το κουμπί O:1/06 (standby) ανάβει. Όταν πατηθεί το κουμπί I:1/00 συνεχίζουν οι διαδικασίες και ανάβει το λαμπάκι O:1/05





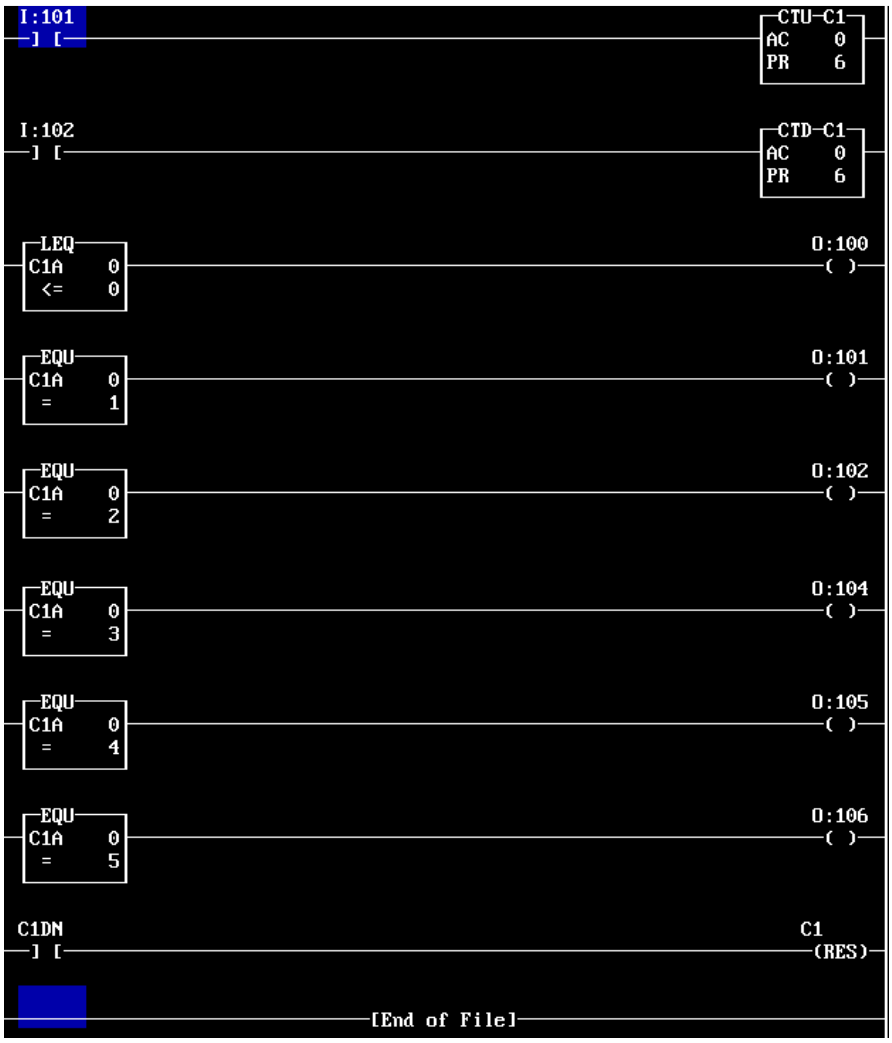
4. Traffic Lights

- **Counter με συγκριτικές εντολές**

Θέλουμε κάθε φορά που πατάμε ένα κουμπί, να ανάβει το επόμενο φως στο φανάρι, και όταν πατάμε ένα άλλο κουμπί να ανάβει το προηγούμενο φως στο φανάρι. Πάντα θα υπάρχει μόνο ένα φως αναμμένο.

Θα χρησιμοποιήσουμε ένα Counter, και θα συγκρίνουμε την τιμή του C1:A. Αν είναι 0 θα είναι αναμμένο το πρώτο φανάρι, αν είναι 1 θα είναι αναμμένο το δεύτερο, κ.ο.κ. Με το

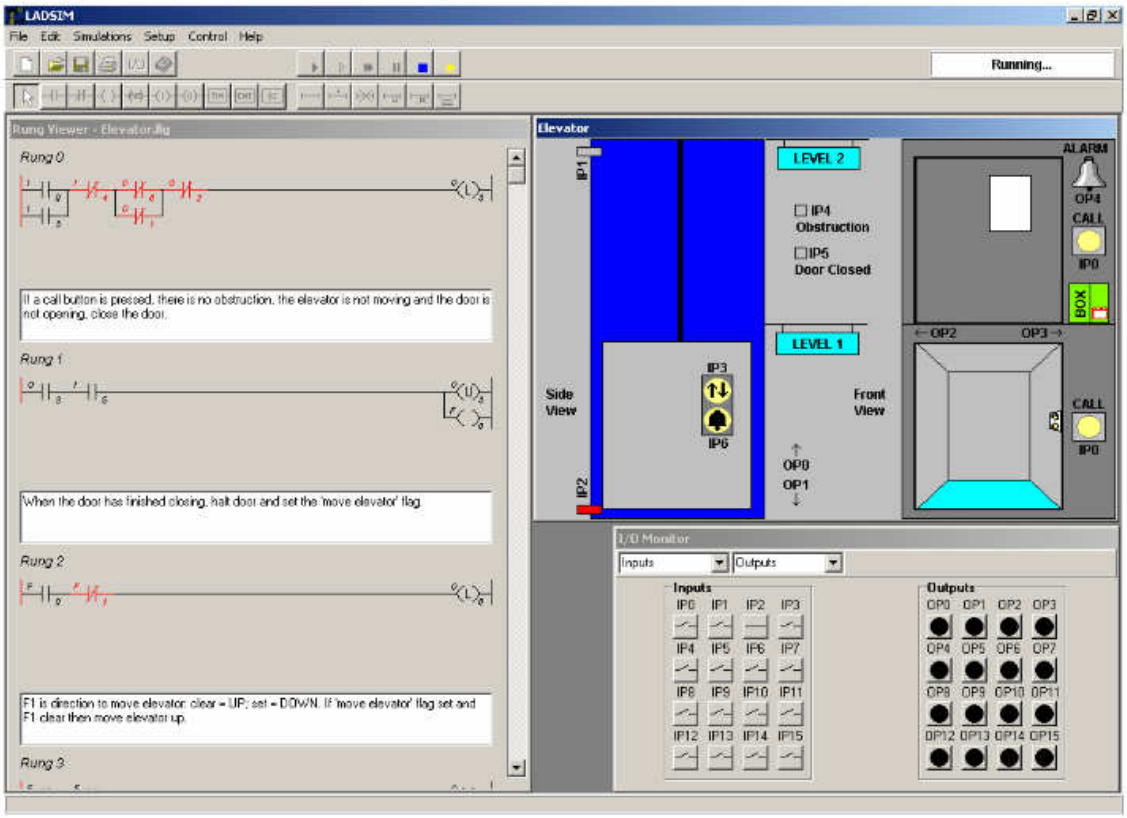
ένα κουμπί θα κάνουμε count up και με το άλλο count down. Όταν ο counter περάσει την τιμή 5, θα γίνεται reset.



ΚΕΦΑΛΑΙΟ 4

ΠΡΟΣΟΜΟΙΩΤΗΣ PLC - LADSIM

4.1 Περιγραφή του προγράμματος προσομοίωσης PLC –LADSIM



Το LADSIM (Ladder Logic Editor and Programmable Logic Controller Simulator) είναι ένα πρόγραμμα προσομοίωσης που επιτρέπει την λογική σχεδίαση PLC στον ηλεκτρονικό υπολογιστή. Μόλις τελειώσει ο σχεδιασμός μπορεί να μεταφερθεί από τον υπολογιστή σε μία πραγματική συσκευή PLC. Υπάρχουν πολλοί κατασκευαστές PLC όπως για παράδειγμα η Allen Bradley, η Mitsubishi και η Siemens.

Πριν από την επιλογή ενός PLC πρέπει να ληφθούν υπόψη δύο πράγματα : πρώτον το μέγεθος του PLC και δεύτερον η λειτουργία του (το είδος και ο αριθμός μονάδων εισόδου και εξόδου).

Το LADSIM έχει τα παρακάτω τρία κύρια χαρακτηριστικά:

- Ο χρήστης μέσω ενός υπολογιστή έχει την δυνατότητα να προσομοιώσει ένα PLC έτσι ώστε το πρόγραμμα Ladder Logic PLC να μπορεί να υλοποιηθεί με ένα απλό ‘drag and drop’.
- Τα προγράμματα PLC μπορούν να χρησιμοποιηθούν για να ελέγχουν με κάποια από τις πολλές ‘οθόνες‘ προσομοίωσης τα προβλήματα ελέγχου.
- Επιτρέπει στον χρήστη να ελέγχει την προσομοίωση και να εξοικειωθεί με τις τεχνικές προγραμματισμού PLC. Ο χρήστης μπορεί στη συνέχεια να προχωρήσει στον έλεγχο των πραγματικών εξωτερικών στοιχείων του εξοπλισμού μέσω του Bytronic Interface, εάν αυτό απαιτείται.

Η Bytronic κατασκευάζει μια σειρά προϊόντων σχεδιασμένων για αυτό το σκοπό. Ανάμεσα σε αυτά καταλέγονται συστήματα μεταφοράς, μονάδες ελέγχου κυκλοφορίας, μονάδες μεταφοράς κυκλικής κίνησης και άλλα.

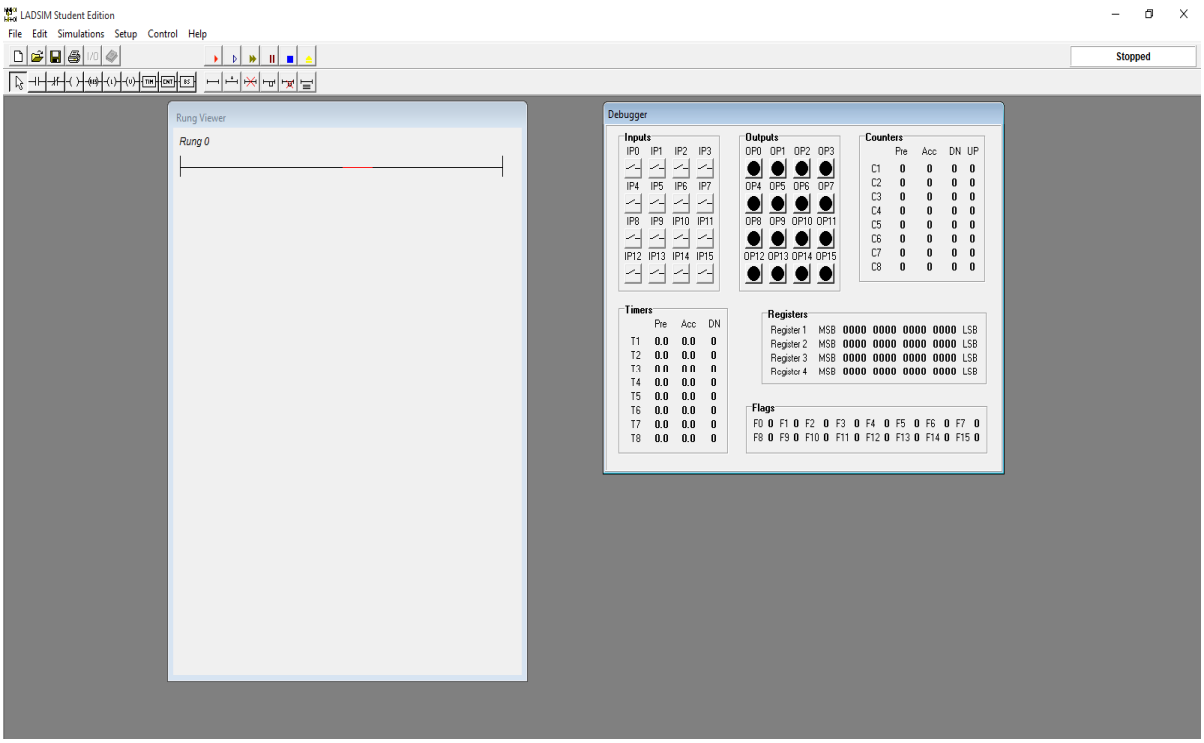
Στο LADSIM τα ακόλουθα στοιχεία είναι διαθέσιμα για τα προγράμματά σας:

- 16 είσοδοι που ξεκινούν από το IP0 μέχρι το IP15.
- 16 έξοδοι που ξεκινούν από το OP0 μέχρι το OP15.
- 16 flags που ξεκινούν από το F0 μέχρι το F15 (βοηθητικά ρελέ).
- 8 timers που ξεκινούν από το T1 μέχρι το T8.
- 8 counters: C1 έως C8 (που μπορούν να χρησιμοποιηθούν για ανιούσα ή κατιούσα μέτρηση).
- 80 λειτουργίες επαναφοράς για τα χρονόμετρα, τους μετρητές και τους καταχωρητές ολίσθησης.
- 16-shift-bit registers.

4.2 Βασική λογική

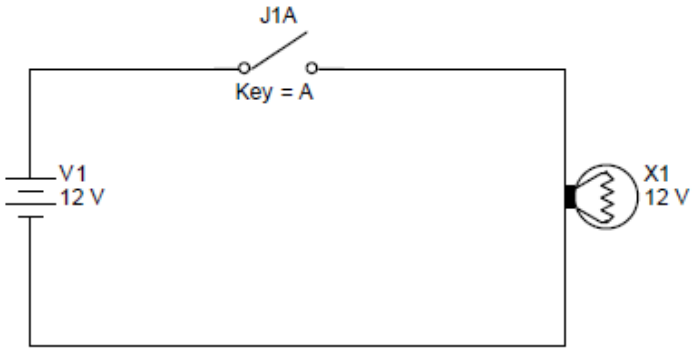
Ανοίγουμε το LADSIM από την Μενού Έναρξη > Προγράμματα > Bytronic > LADSIM3 > LADSIM3 menu.

Ένα παράθυρο με τίτλο LADSIM θα εμφανιστεί όπως φαίνεται στο Σχήμα 1.



Σχήμα 4.1: Παράθυρο LADSIM

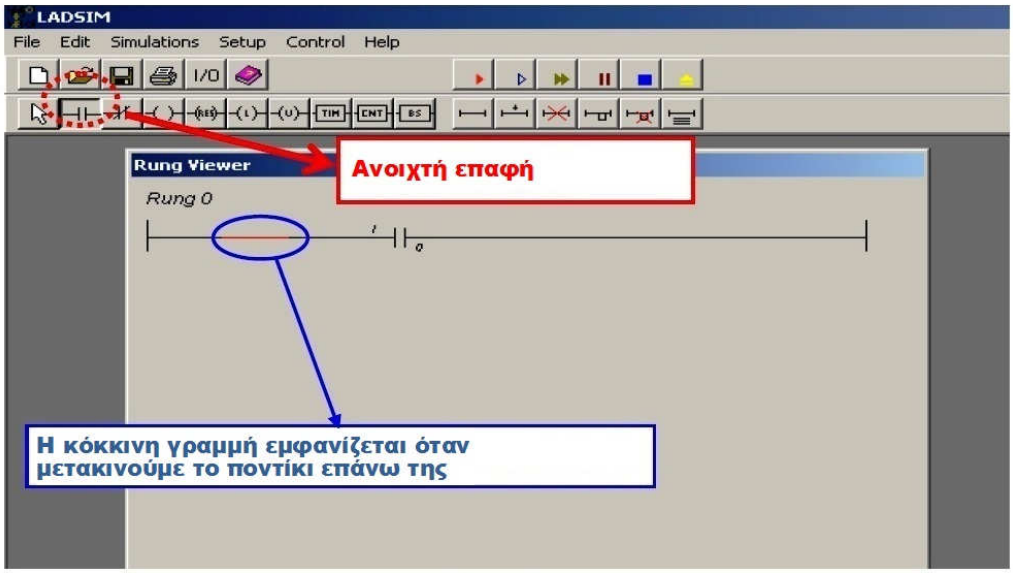
Θεωρούμε το ηλεκτρικό κύκλωμα που φαίνεται στο Σχήμα 4.2. Αποτελείται από ένα διακόπτη, μια λάμπα και ένα τροφοδοτικό. Κλείνοντας το διακόπτη, επιτρέπεται η ροή ρεύματος στο κύκλωμα και η λάμπα θα ανάψει.



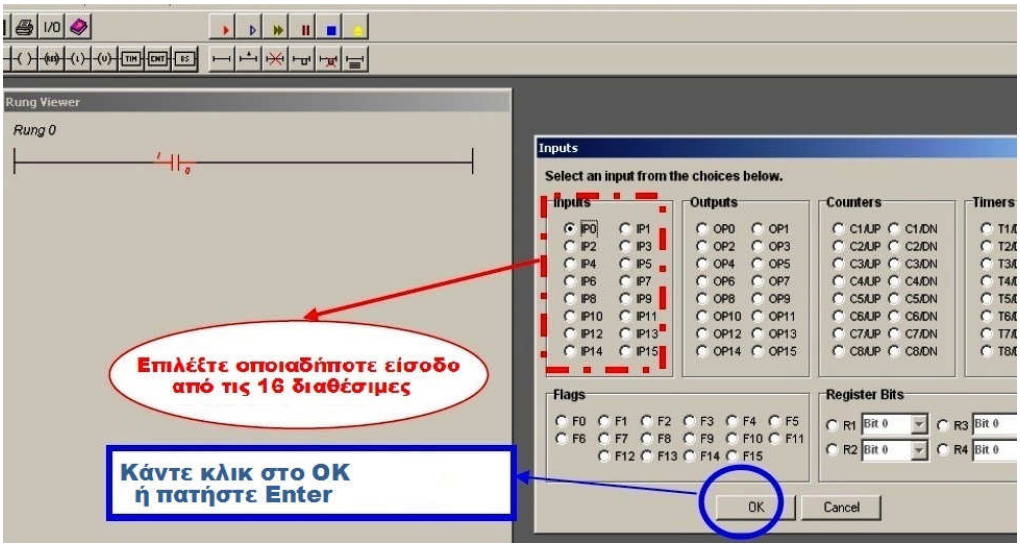
Σχήμα 4.2: Ηλεκτρικό κύκλωμα

Θα χρησιμοποιήσουμε το LADSIM για να δημιουργήσουμε μία προσομοίωση του κυκλώματος.

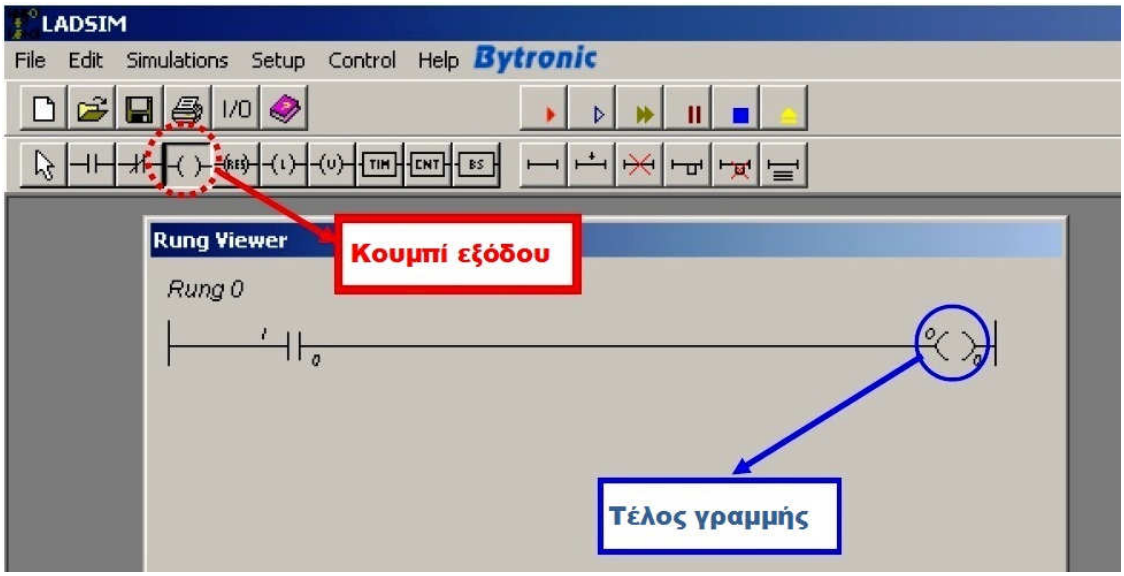
Βήμα 1: Κάνουμε κλικ στο εικονίδιο normally open contact, το μετακινούμε και το τοποθετούμε όπου φαίνεται η κόκκινη γραμμή, στην συνέχεια κάνουμε κλικ σε αυτό, όπως φαίνεται παρακάτω.



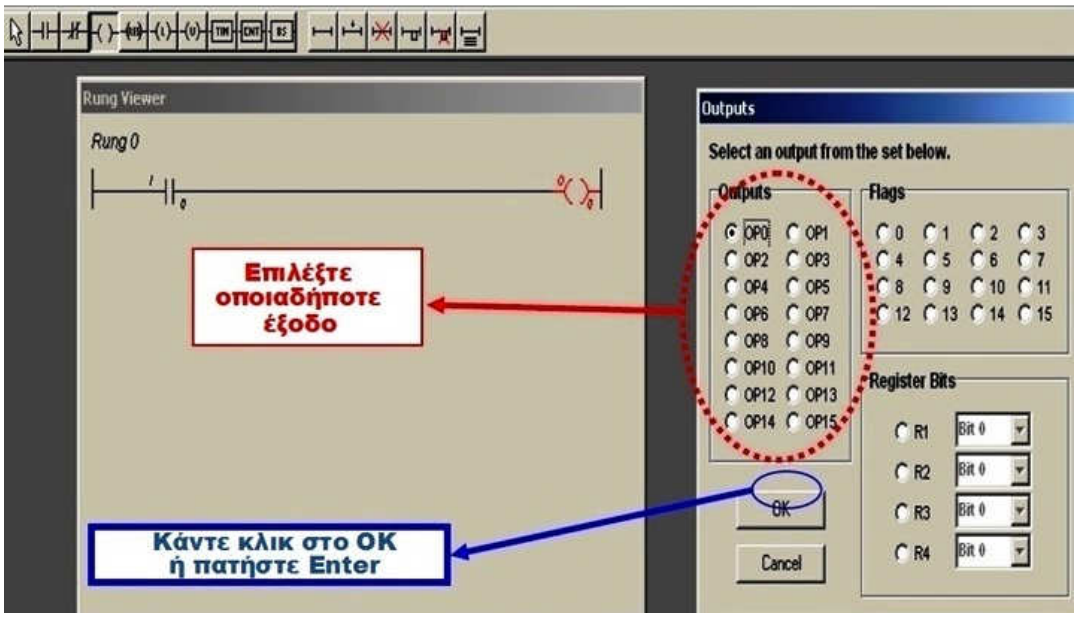
Βήμα 2: Όταν τοποθετηθεί η κατάλληλη ανοιχτή επαφή στην κόκκινη γραμμή και κάνουμε κλικ πάνω της, θα εμφανιστεί το παράθυρο επιλογής εισόδου. Στο παράθυρο 'Inputs' θα εμφανιστεί μια ομάδα από 16 εισόδους. Επιλέγουμε οποιαδήποτε είσοδο από τις 16 διαθέσιμες κάνοντας κλικ μέσα στο λευκό κύκλο στα αριστερά της. Στη συνέχεια κάνουμε κλικ στο <OK> ή πατάμε Enter.



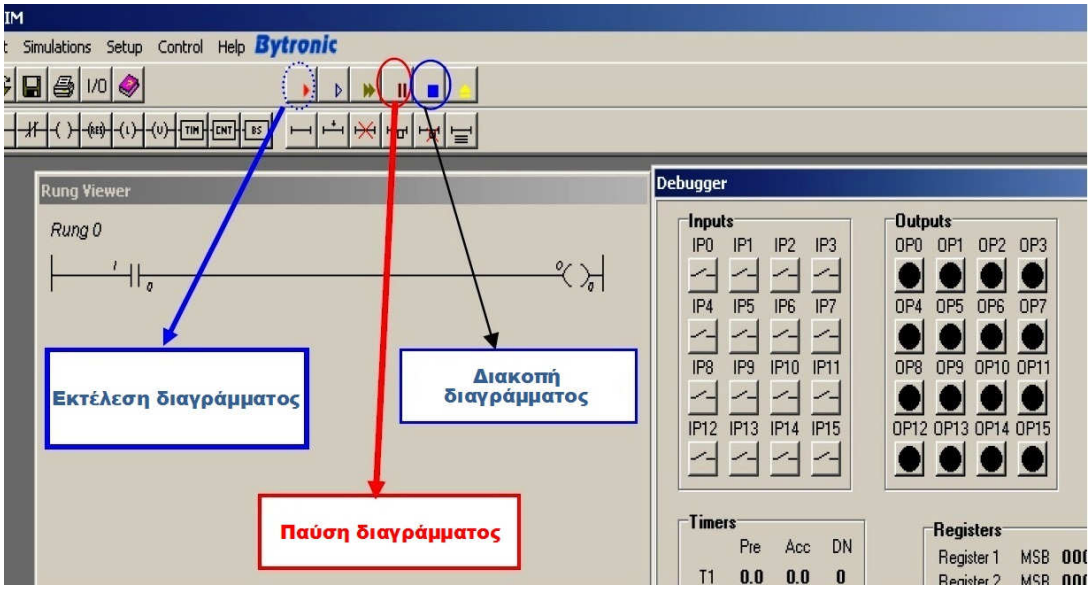
Βήμα 3: Κάνουμε κλικ στο κατάλληλο εικονίδιο κουμπιού εξόδου, το μετακινούμε και το τοποθετούμε το όπου φαίνεται η κόκκινη γραμμή στο Rung0. Στη συνέχεια κάνουμε κλικ πάνω της, όπως φαίνεται παρακάτω.



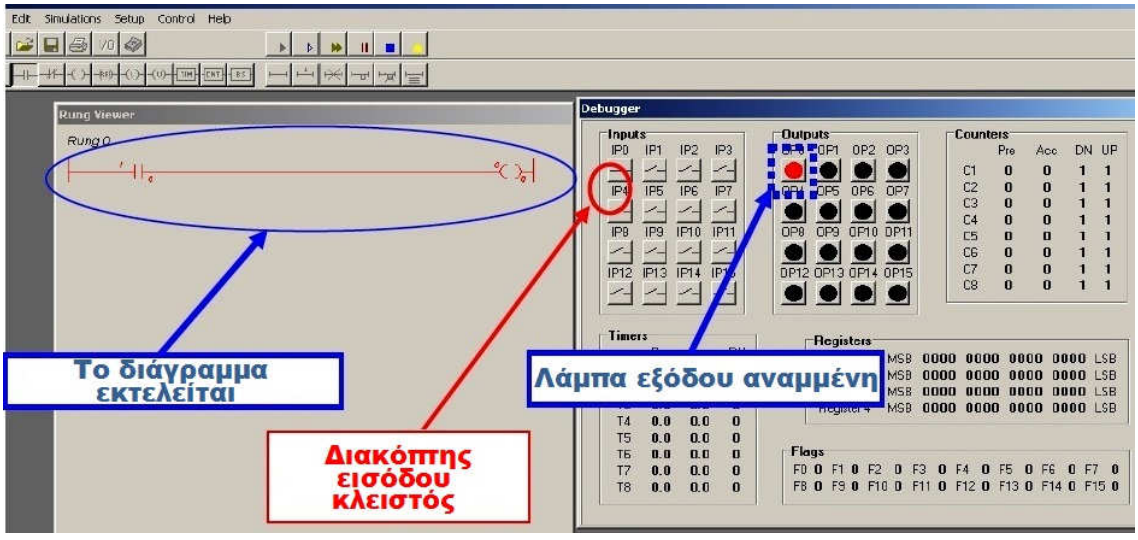
Βήμα 4ο: Όταν τοποθετήσουμε το αντίστοιχο εικονίδιο εξόδου στην κόκκινη γραμμή και κάνουμε κλικ, θα εμφανιστεί το πλαίσιο επιλογής. Στο παράθυρο 'Outputs' θα εμφανιστεί μια ομάδα από 16 εξόδους. Επιλέγουμε οποιαδήποτε από τις 16 διαθέσιμες εξόδους, κάνοντας κλικ μέσα στο λευκό κύκλο στα αριστερά της. Στη συνέχεια κάνουμε κλικ στο <OK> ή πατάμε Enter.



Βήμα 5ο: Κάνουμε κλικ στο εικονίδιο RUN για να εκτελεστεί το διάγραμμα Ladder. Μπορούμε επίσης να χρησιμοποιήσουμε το RUN για να συνεχίσουμε το διάγραμμα όταν βρίσκεται σε παύση. Χρησιμοποιούμε το εικονίδιο PAUSE για παύση της εκτέλεσης του διαγράμματος. Όταν κάνουμε κλικ στο εικονίδιο STOP θα σταματήσει η εκτέλεση του διαγράμματος.

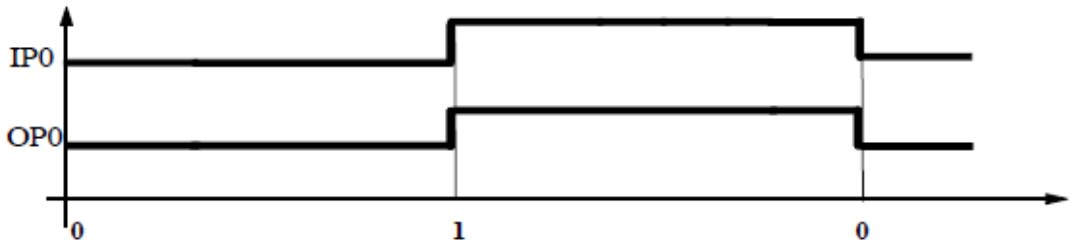


Βήμα 6ο: Όταν εκτελείται το διάγραμμα, η λάμπα δεν θα ανάψει μέχρι να κάνουμε κλικ στο IP0. Στη συνέχεια, το ρεύμα θα περάσει από την έξοδο. Στο παράθυρο 'Outputs' το IP0 θα γίνει κόκκινο που σημαίνει ότι το κύκλωμα είναι ενεργό, όπως φαίνεται παρακάτω.



Αποτέλεσμα

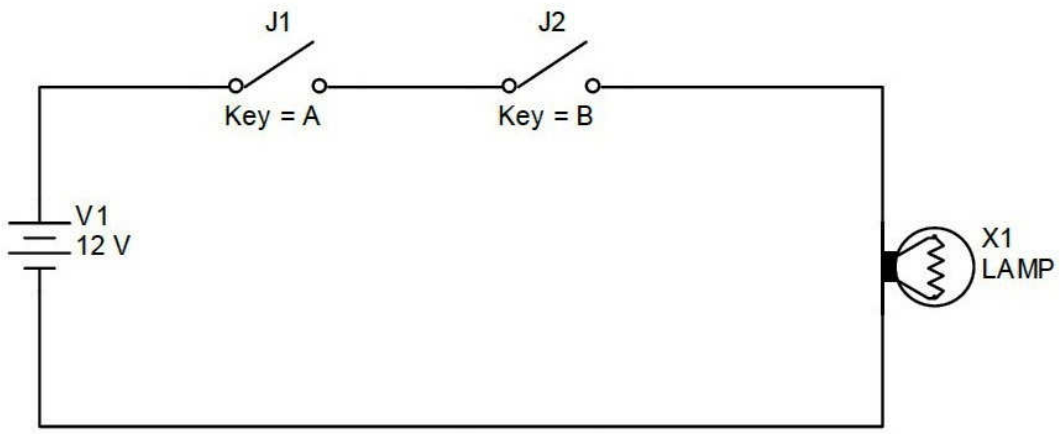
Η χρονική αλληλουχία εμφανίζεται σε δύο σήματα 1 και 0. Όταν το ρεύμα περνά από τα IP0 και OP0, η λάμπα ανάβει (σήμα 1). Όταν η είσοδος IP0 είναι απενεργοποιημένη, δεν διέρχεται ρεύμα στο κύκλωμα (σήμα 0). Όταν πατήσουμε να ενεργοποιηθεί το IP0 θα ενεργοποιηθεί και το κύκλωμα και το OP0. Διαφορετικά, αν απενεργοποιήσουμε το IP0, θα απενεργοποιηθεί το κύκλωμα όπως και το OP0. Το ακόλουθο Σχήμα δείχνει τα δύο στάδια (on=1 και off=0).



4.2.1 Παραδείγματα βασικής λογικής

Παράδειγμα 1 (Άλυτο)

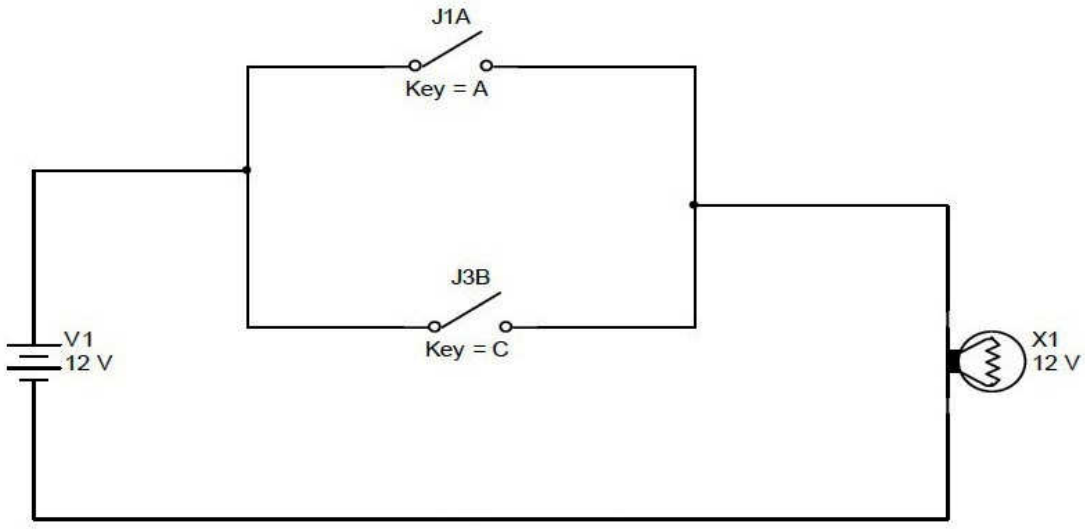
Θεωρούμε το ηλεκτρικό κύκλωμα που φαίνεται παρακάτω στο Σχήμα 4.3. Αποτελείται από δύο διακόπτες σε σειρά, μια λάμπα και ένα τροφοδοτικό. Όταν ο διακόπτης Α είναι κλειστός ενώ ο διακόπτης Β ανοιχτός η λάμπα δεν θα ανάψει. Η λάμπα δεν θα ανάψει ούτε και αν ο διακόπτης Β είναι κλειστός ενώ ο διακόπτης Α είναι ανοιχτός. Μόνο όταν διακόπτες είναι και οι δύο κλειστοί ταυτόχρονα η λάμπα θα ανάψει. Θα δημιουργήσουμε την προσομοίωση του κυκλώματος.



Σχήμα 4.3: Ηλεκτρικό κύκλωμα

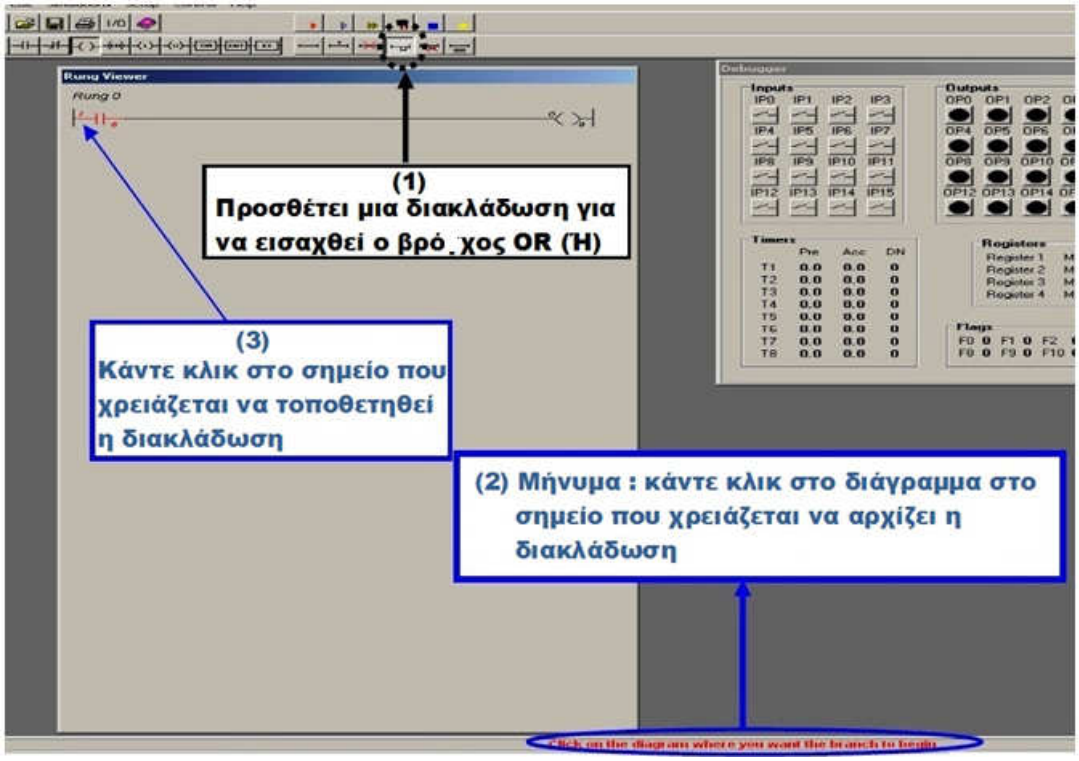
Παράδειγμα 2 (Λυμένο)

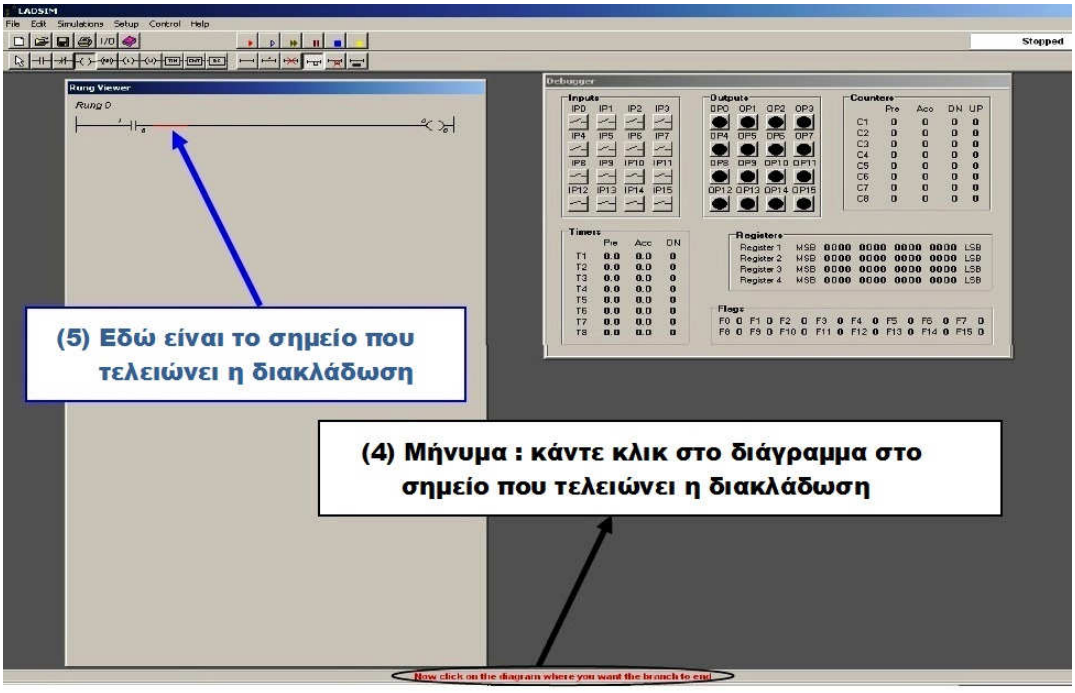
Θεωρούμε το ηλεκτρικό κύκλωμα που φαίνεται παρακάτω στο Σχήμα 4.4. Αποτελείται από δύο διακόπτες παράλληλα, μια λάμπα και ένα τροφοδοτικό. Όταν κλείνουμε το διακόπτη Α ή το διακόπτη Β, το κύκλωμα διαρρέεται από ρεύμα και η λάμπα ανάβει. Θα δημιουργήσουμε την προσομοίωση του κυκλώματος.

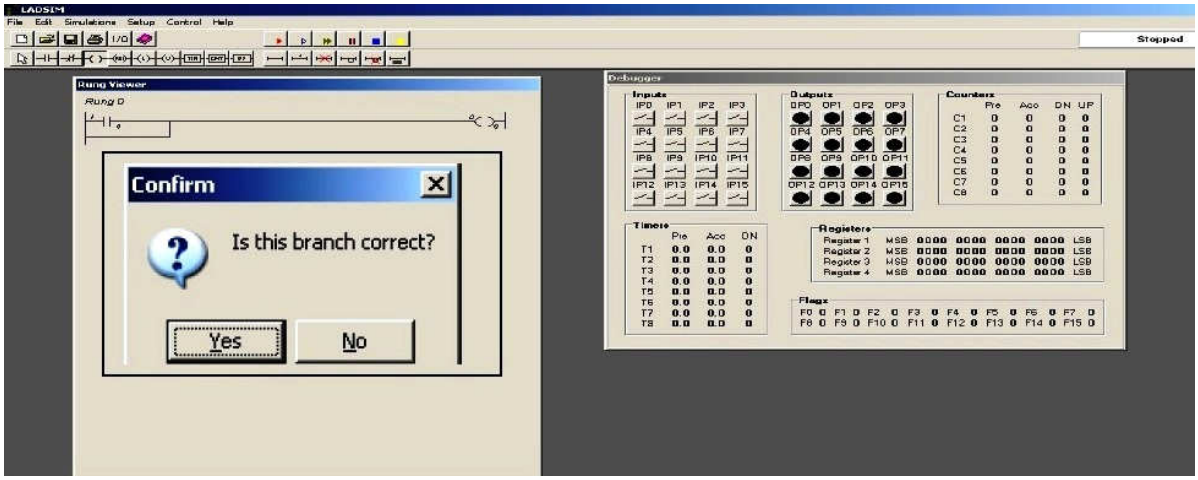


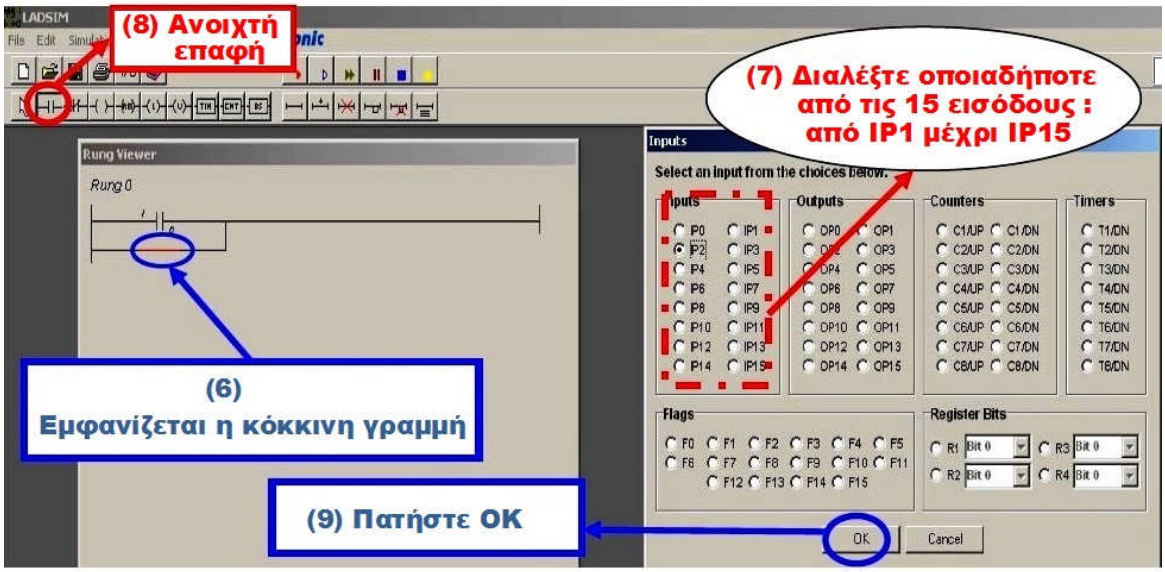
Σχήμα 4.4: Ηλεκτρικό κύκλωμα

Για να μπορέσουμε να εφαρμόσουμε την δήλωση Ή, θα χρειαστεί να χρησιμοποιήσουμε μια διακλάδωση.









4.3 Η λειτουργία του χρονοδιακόπτη

Υπάρχουν δύο τύποι χρονοδιακοπτών στο LADSIM, που έχουν και οι δύο παρόμοιες ιδιότητες, αλλά λειτουργούν με ελαφρώς διαφορετικούς τρόπους.

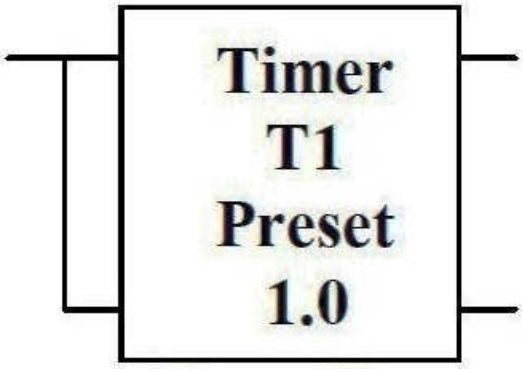
Ο χρονοδιακόπτης On-delay (Σχήμα 4.5) έχει μια καθυστέρηση και θα αρχίσει να μετράει όταν ενεργοποιηθεί και όταν απενεργοποιηθεί θα κάνει reset.

Ο χρονοδιακόπτης Latched On-Delay timer (Σχήμα 18) θα συνεχίσει να μετράει όταν ενεργοποιηθεί, άσχετα με το αν έχει απενεργοποιηθεί πριν φτάσει το χρόνο καθυστέρησής του.

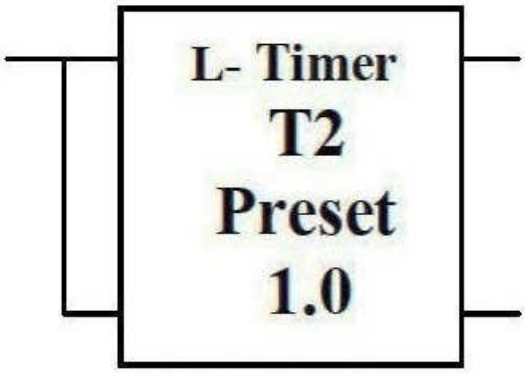
Το LADSIM περιορίζει το συνολικό αριθμό των χρονοδιακοπτών (on-delay και latched on-delay) σε οκτώ.

Για όλους τους χρονοδιακόπτες LADSIM ισχύουν τα ακόλουθα:

- Η προσαύξηση που μπορεί να παρατηρηθεί σε κατάσταση προσομοίωσης στο χρονοδιακόπτη Accumulator είναι της τάξης των δεκάτων των δευτερολέπτων.
- Αν ο χρονοδιακόπτης απενεργοποιηθεί, η μέτρηση θα συνεχίσει μέχρι να φτάσει τον προκαθορισμένη χρόνο (σε δευτερόλεπτα) μόνο αν ο χρονοδιακόπτης είναι Latched On-Delay.
- Το Done Bit (DN) ενεργοποιείται όταν ο Accumulator φτάσει στον προκαθορισμένη χρόνο που έχει ρυθμιστεί.
- Όταν το DN έχει οριστεί, μπορεί να αλλάξει μόνο με τη χρήση της λειτουργίας επαναφοράς (RES).
- Η λειτουργία επαναφοράς είναι μία εξωτερική λειτουργία και μπορεί να ενεργοποιηθεί από μία επαφή εισόδου και έχει προτεραιότητα σε σχέση με την αύξηση του Accumulator του χρονοδιακόπτη.
- Όταν το χρονόμετρο ολοκληρώσει την μέτρησή του θα εμφανιστεί ένα X στο παράθυρο του DN στο πλαίσιο του Simulator.



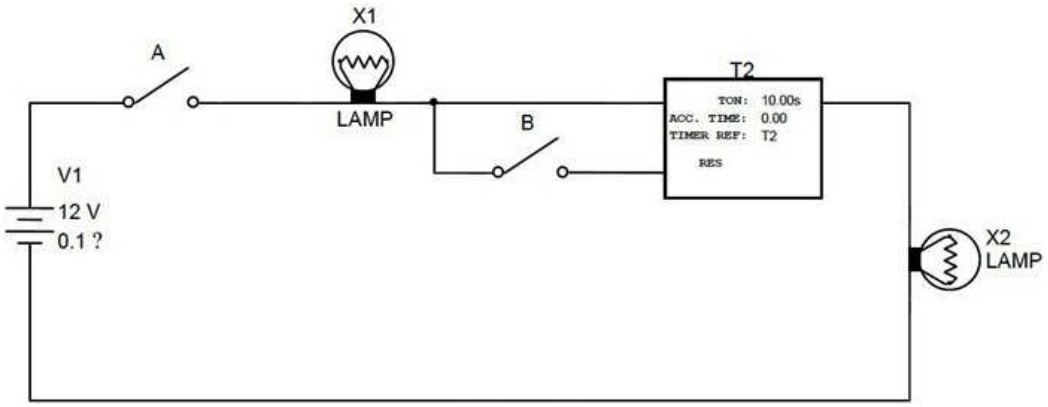
Σχήμα 4.5: Χρονοδιακόπτης On-delay



Σχήμα 4.6: Χρονοδιακόπτης Latched On-Delay

4.3.1 Παράδειγμα λειτουργίας του χρονοδιακόπτη

Θεωρούμε το ηλεκτρικό κύκλωμα που φαίνεται παρακάτω στο Σχήμα 4.7. Αποτελείται από ένα διακόπτη, δύο λάμπες, ένα χρονόμετρο (Latched On-Delay) και ένα τροφοδοτικό. Η πρώτη λάμπα X1 θα ανάψει εάν ο διακόπτης A είναι κλειστός (on) και μετά από 10 δευτερόλεπτα η δεύτερη λάμπα X2 θα ανάψει. Εναλλακτικά αν ο διακόπτης B είναι κλειστός (on) τότε η λάμπα X2 θα σβήσει. Όταν ο διακόπτης A είναι ανοιχτός (off) οι λάμπες θα πρέπει να απενεργοποιούνται. Θα δημιουργήσουμε την προσομοίωση του κυκλώματος.



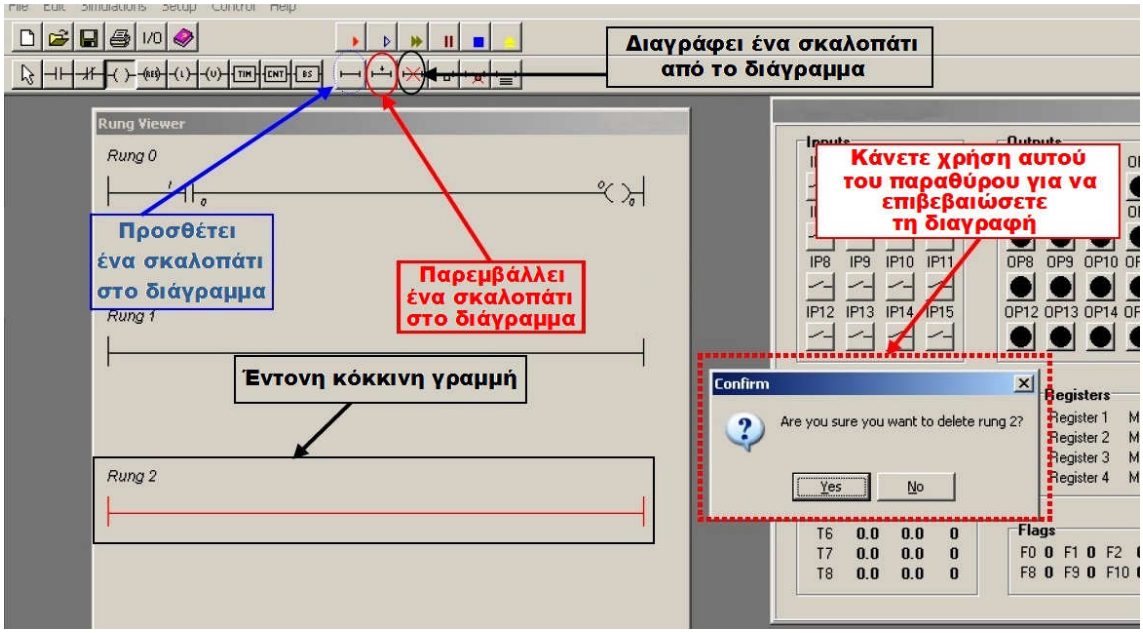
Σχήμα 4.7: Ηλεκτρικό κύκλωμα

Βήμα 1ο

Ξεκινάμε ένα νέο project και δημιουργούμε μία κανονική ανοιχτή επαφή, NOC (IP0) και την έξοδο (OP0).

Βήμα 2ο

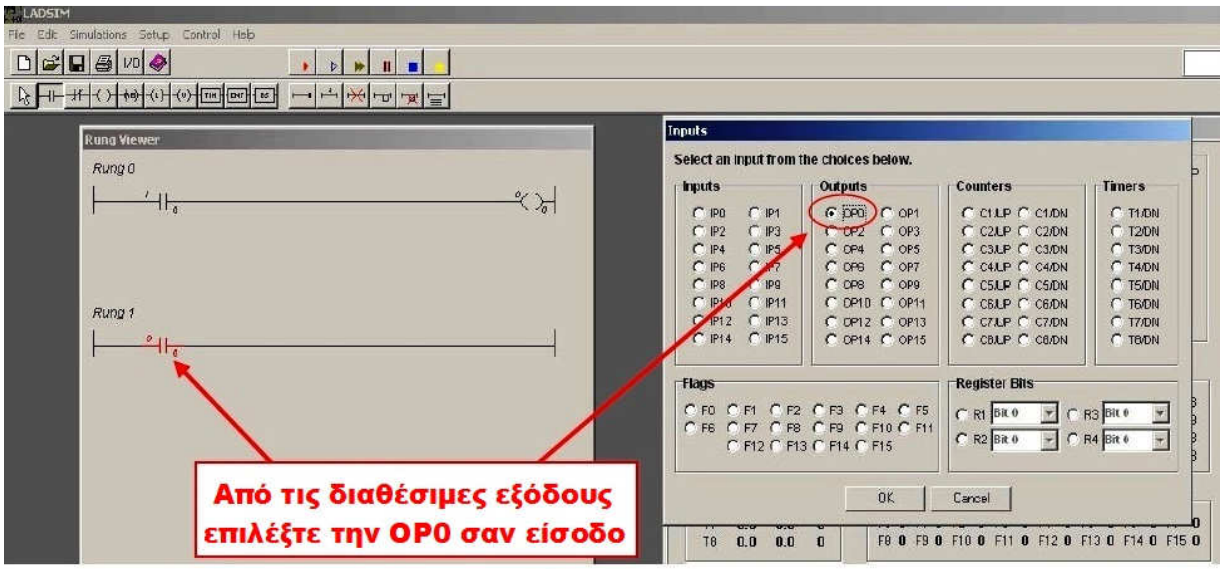
Κάνουμε κλικ στο εικονίδιο add rung για να προσθέσουμε ένα σκαλοπάτι στο τρέχον διάγραμμα. Το νέο σκαλοπάτι θα προστεθεί στο τέλος του διαγράμματος. Μπορούμε επίσης να χρησιμοποιήσουμε το εικονίδιο delete rung για να διαγράψουμε ένα σκαλοπάτι από το διάγραμμά σας. Κάνοντας κλικ θα εμφανιστεί το μήνυμα 'κάντε κλικ στο σκαλοπάτι που θέλετε να διαγραφεί'. Κάνουμε κλικ σε οποιοδήποτε σημείο του σκαλιού που θέλουμε να διαγράψουμε, στη συγκεκριμένη περίπτωση το σκαλοπάτι 2. Το σκαλοπάτι 2 θα γίνει κόκκινο και θα εμφανιστεί ένα μήνυμα που θα ζητάει να επιβεβαιώσουμε τη διαγραφή, όπως φαίνεται παρακάτω.



Βήμα 3ο

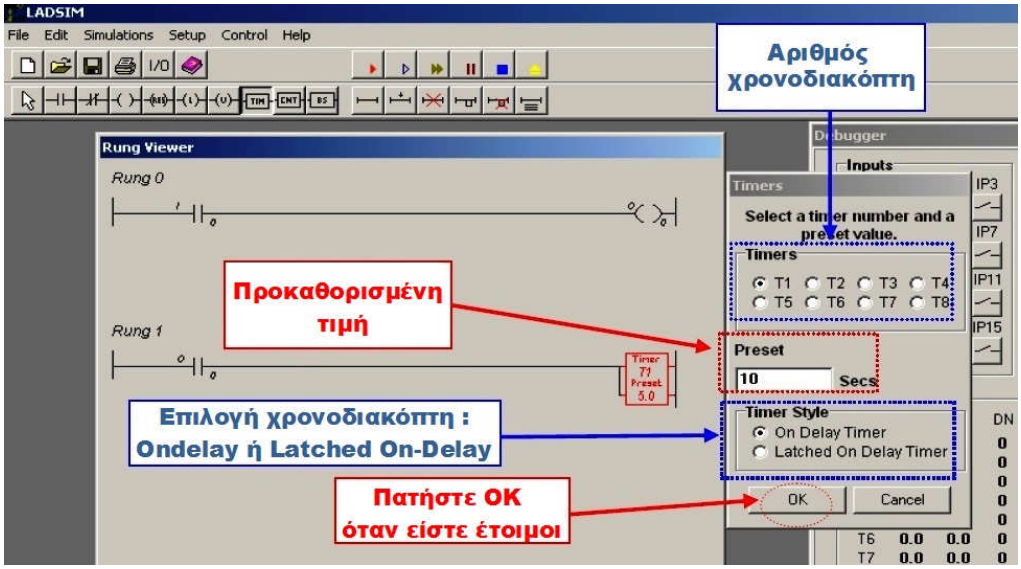
Δημιουργούμε μία άλλη ανοιχτή επαφή στο σκαλοπάτι 1 και στη συνέχεια επιλέγουμε την OP0 από τις εξόδους, η οποία θα είναι η είσοδος στο συγκεκριμένο διάγραμμα, κάνοντας

κλικ μέσα στο λευκό κύκλο στα αριστερά της. Στη συνέχεια κάνουμε κλικ στο <OK> ή πατάμε Enter όπως φαίνεται παρακάτω.



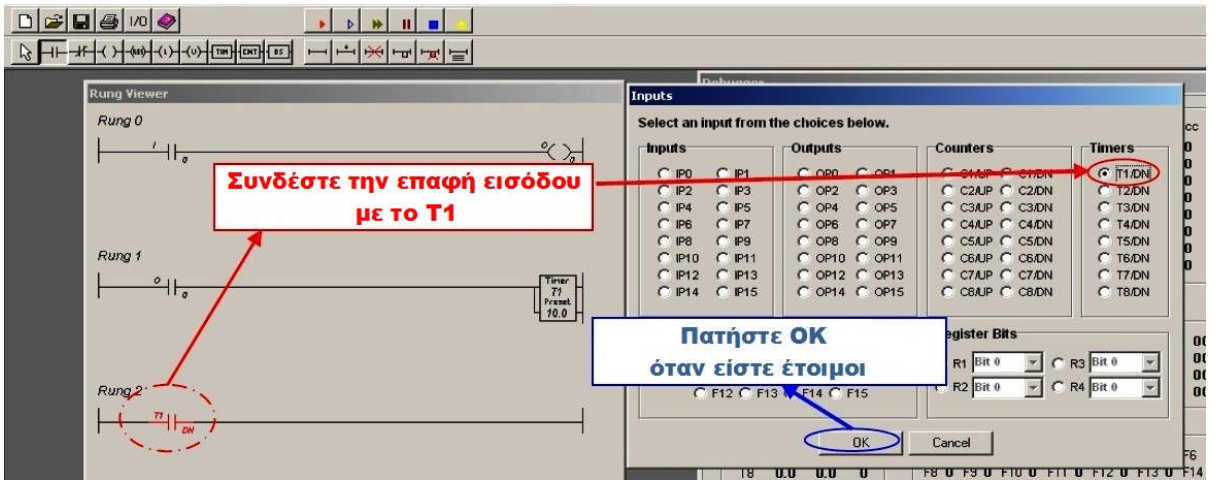
Βήμα 4ο

Κάνουμε κλικ στο εικονίδιο του χρονοδιακόπτη και σύρουμε ένα χρονόμετρο στο σκαλοπάτι 1 το οποίο αφήνουμε στα δεξιά του. Όταν το χρονόμετρο τοποθετηθεί πάνω στο σκαλοπάτι, θα εμφανιστεί το παράθυρο επιλογής για το χρονόμετρο. Από τη λίστα επιλογής χρονομέτρων επιλέγουμε το T1. Σαν προκαθορισμένη τιμή επιλέγουμε τα 10 δευτερόλεπτα κάνοντας κλικ στο αντίστοιχο κουμπί. Στη συνέχεια επιλέγουμε τον τύπο του χρονοδιακόπτη (On-Delay). Τέλος κάνουμε κλικ στο <OK> για να αποθηκευτούν οι ρυθμίσεις του χρονοδιακόπτη.



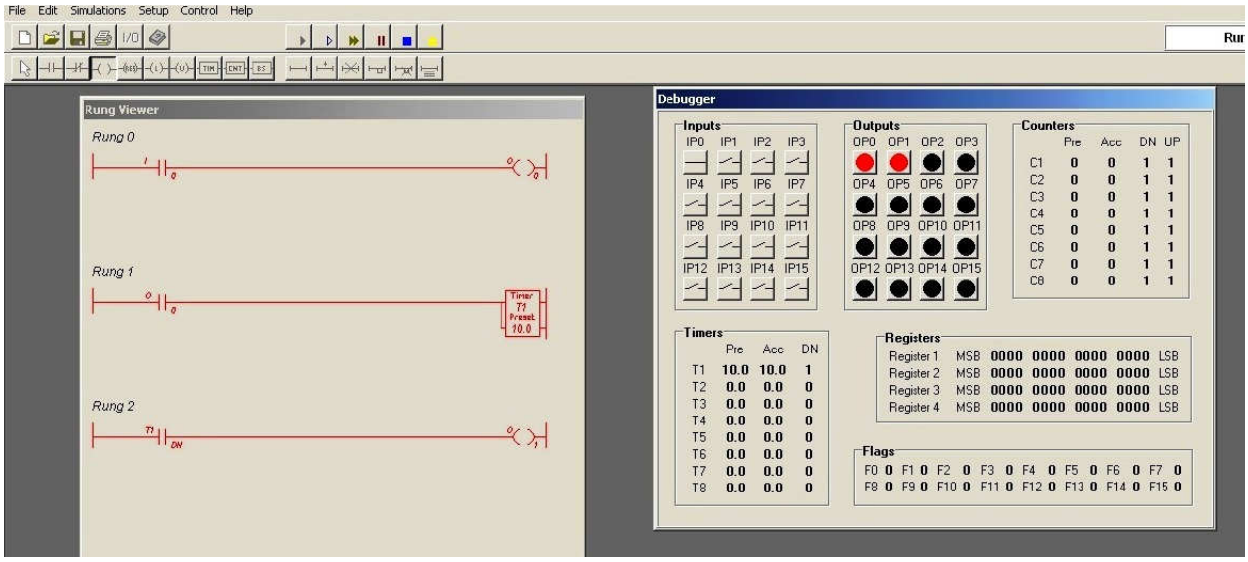
Βήμα 5ο

Κάνουμε κλικ στο εικονίδιο add rung για να προσθέσουμε ένα σκαλοπάτι και προσθέτουμε μια ανοιχτή επαφή εισόδου σε αυτό. Επιλέγουμε την επαφή εισόδου που θα λειτουργήσει σαν χρονόμετρο T1 (T1/DN), επιλέγοντάς το από τις επιλογές χρονομέτρου, όπως φαίνεται παρακάτω.



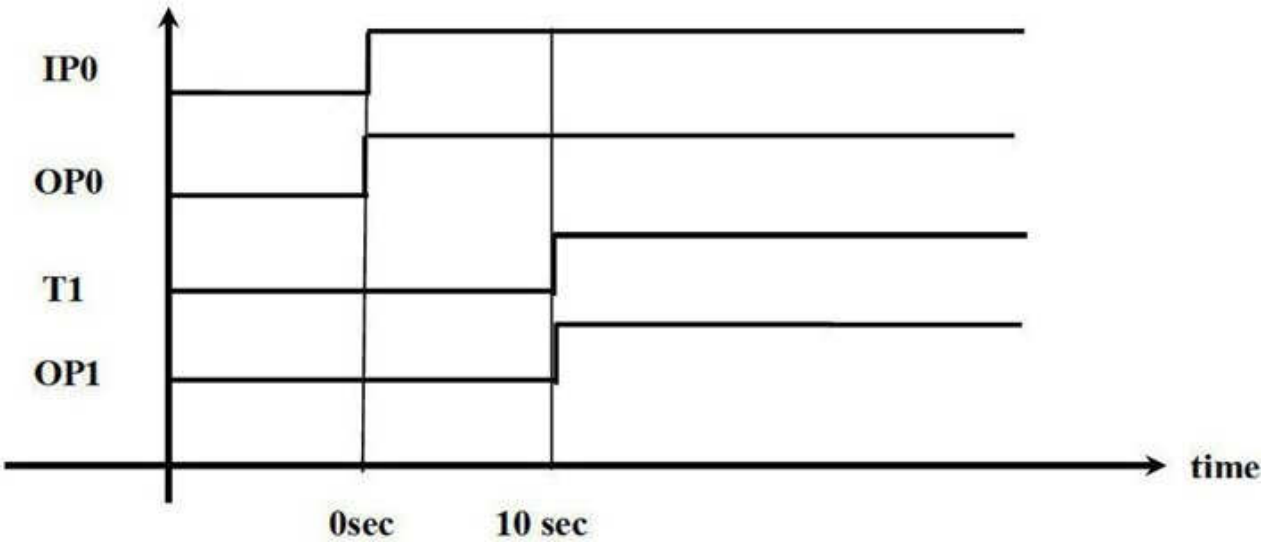
Βήμα 6ο

Κάνουμε κλικ στο OP1 για να γίνει και το σκαλοπάτι 2 έξοδος, στη συνέχεια εκτελούμε το διάγραμμά μας.



Αποτέλεσμα

Όταν η IP0 είναι απενεργοποιημένη, το κύκλωμα δεν διαρρέεται από ρεύμα. Κάνοντας κλικ στην IP0, θα γυρίσει ο διακόπτης. Η έξοδος OP0 αλλά και ο χρονοδιακόπτης θα ενεργοποιηθούν. Μετά από 10 δευτερόλεπτα το χρονόμετρο θα επιτρέψει στην OP1 να ενεργοποιηθεί και αυτή, όπως φαίνεται παρακάτω.

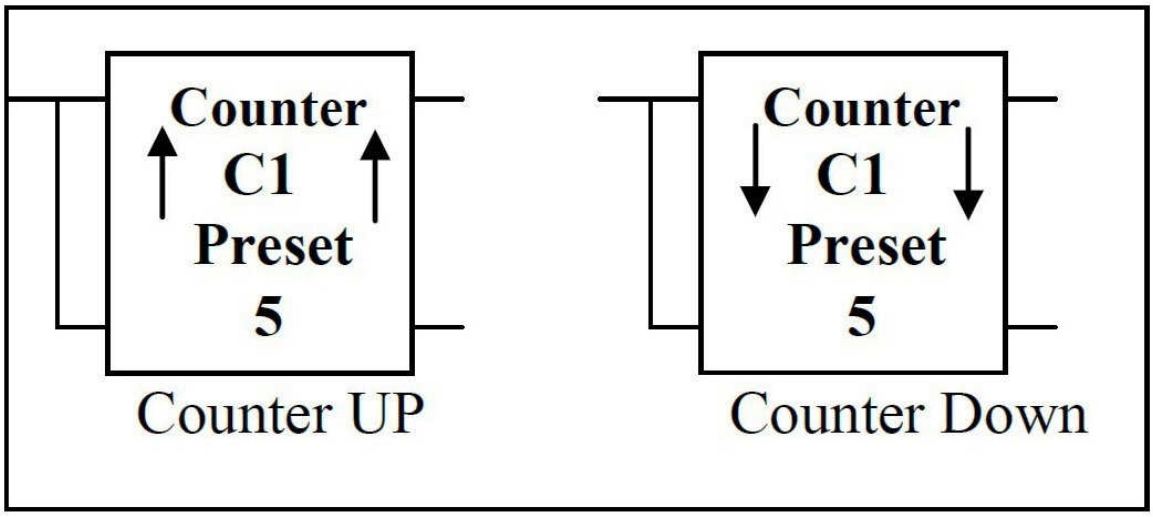


4.4 Η λειτουργία του μετρητή

Η λειτουργία των μετρητών στο LADSIM εκτελείται με δύο βασικές μορφές. Και οι δύο μορφές χρησιμοποιούν την θετική μετάβαση (OFF σε ON) για την για την αλλαγή των εισόδων ως έναυσμα για να αλλάξουν τιμές.

Ο μετρητής UP προσαυξάνει τη μέτρησή του μέχρι ενός ορίου, ενώ ο μετρητής DOWN μειώνει την μέτρηση του από ένα όριο μέχρι το μηδέν. Η τιμή και των δύο μετρητών αποθηκεύεται σε ένα καταχωρητή, ο μετρητής UP αυξάνει τον καταχωρητή μέχρι το προκαθορισμένο όριο, ενώ ο μετρητής DOWN μειώνει τον καταχωρητή από το προκαθορισμένο όριο μέχρι το μηδέν. Το Bit που υπάρχει μετά την ολοκλήρωση της μέτρησης, διαφέρει στους δύο μετρητές. Ο μετρητής UP ενεργοποιεί το UP bit (που ονομάζεται Cn/UP, όπου ‘n’ είναι ο αριθμός του μετρητή) όταν η τιμή του φτάσει στο προκαθορισμένο όριο. Αντίστοιχα ο μετρητής DOWN ενεργοποιεί το DN bit (που

ονομάζεται Cn/DN, όπου ‘n’ είναι ο αριθμός του μετρητή) όταν η τιμή του φτάσει στο μηδέν. Το σύμβολο για τους μετρητές Up και Down φαίνεται στο Σχήμα 26.



Σχήμα 4.8: Μετρητές Up και Down

Σημείωση:

Η καταμέτρηση μπορεί να παρατηρηθεί στη λειτουργία προσομοίωσης του μετρητή.

Ο ίδιος μετρητής μπορεί να χρησιμοποιηθεί για μέτρηση και προς τα πάνω και προς τα κάτω.

Όπως και στα χρονόμετρα, αφού έχει οριστεί το Bit μπορεί να μηδενιστεί μόνο από τη λειτουργία επαναφοράς (RES) που έχει προτεραιότητα σε σχέση με την καταμέτρηση. Μόλις ο μετρητής τελειώσει την μέτρησή του επιστρέφει στην αρχική του κατάσταση (δηλαδή μηδέν για μετρητή UP και το προκαθορισμένο όριο για μετρητή DOWN).

4.4.1. Παράδειγμα Λειτουργίας Μετρητή

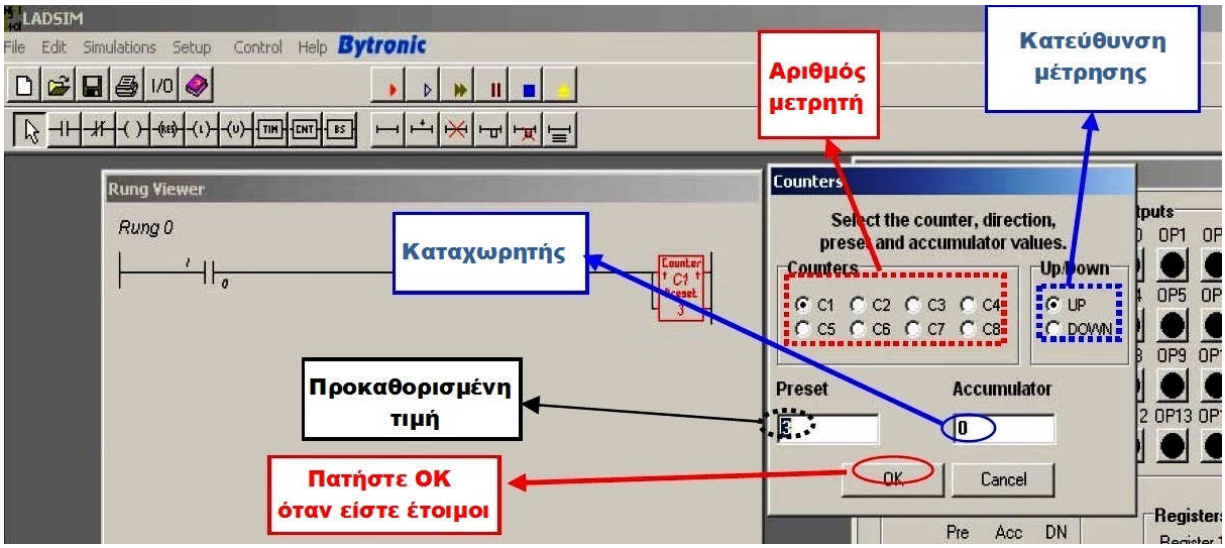
Θα φτιάξουμε ένα μοντέλο προσομοίωσης με δύο λάμπες, όπου μετά την μέτρηση μέχρι τα τρία δευτερόλεπτα οι λάμπες θα ανάβουν. Εάν ο διακόπτης Α είναι κλειστός μετά από τρία δευτερόλεπτα θα ανάβουν. Όταν ο διακόπτης Β είναι κλειστός (ON) οι λάμπες θα πρέπει να σβήνουν και ο μετρητής να μηδενίζεται.

Βήμα 1ο

Όπως και στα προηγούμενα, θα δημιουργήσουμε μία ανοιχτή επαφή (IP0).

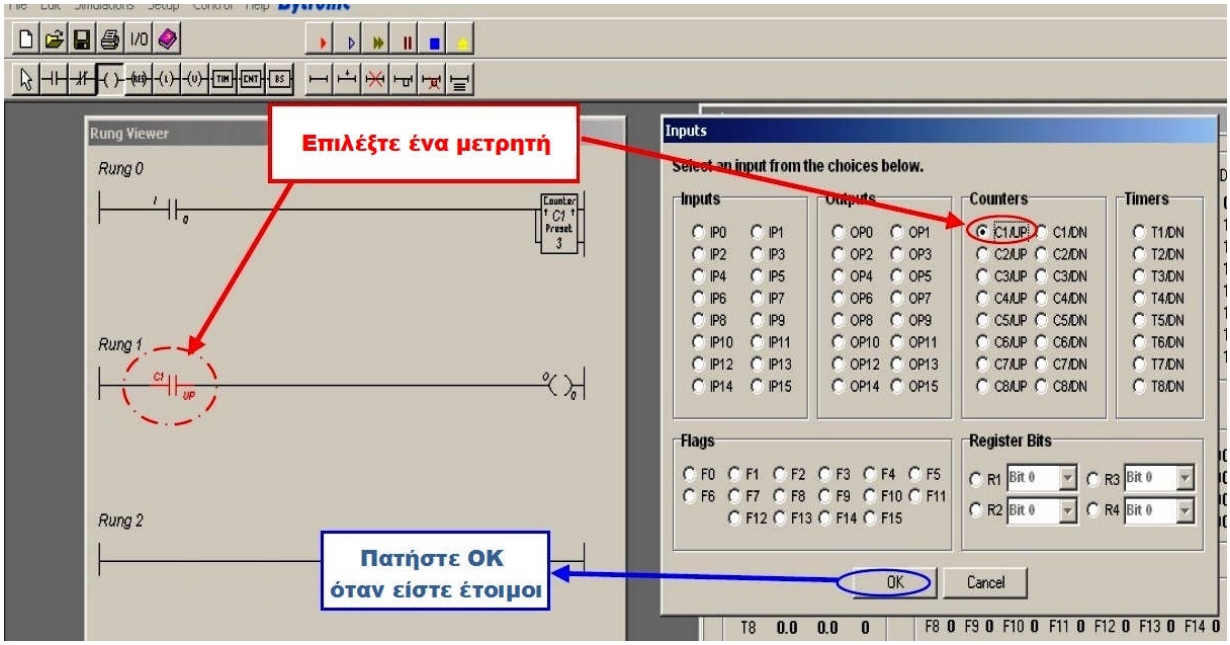
Βήμα 2ο

Κάνουμε κλικ στο εικονίδιο του μετρητή και σύρουμε ένα μετρητή πάνω στο σκαλοπάτι 0. Όταν ο μετρητής έχει τοποθετηθεί στο σκαλοπάτι, θα ανοίξει το παράθυρο επιλογής των μετρητών. Επιλέγουμε τον μετρητή C1 και το Up από την επιλογή Up / Down. Για να ορίσουμε σαν τιμή τα 3 δευτερόλεπτα, πληκτρολογούμε 3 στο κουτί επιλογής προκαθορισμένης τιμής. Στον καταχωρητή πληκτρολογούμε 0. Κάνουμε κλικ στο <OK> για να αποθηκεύσουμε την ρύθμιση, όπως φαίνεται παρακάτω.



Βήμα 3ο

Κάνουμε κλικ για να προσθέσουμε ένα σκαλοπάτι και τοποθετούμε σε αυτό μία ανοιχτή επαφή εισόδου. Στη συνέχεια τοποθετούμε μία επαφή εξόδου στα δεξιά του σκαλοπατιού και την εκχωρούμε σαν OP0.



Βήμα 4ο

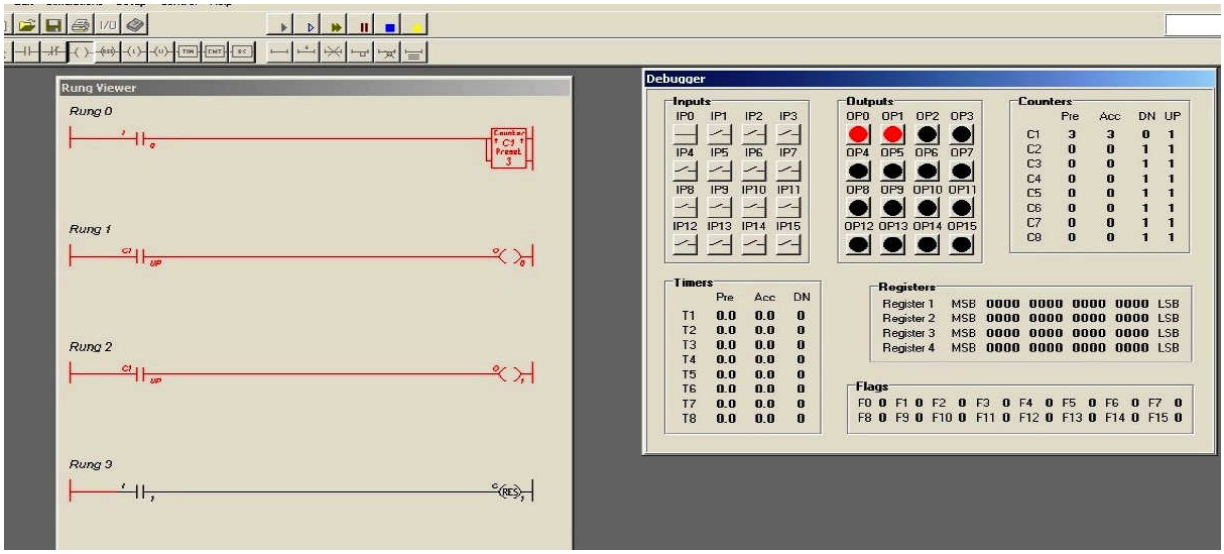
Κάνουμε κλικ για να προσθέσουμε ένα νέο σκαλοπάτι. Προσθέτουμε μία ανοιχτή επαφή πάνω του και επιλέγουμε C1 / UP από τις επιλογές. Στη συνέχεια προσθέτουμε μία επαφή εξόδου στα δεξιά του σκαλοπατιού και ορίζουμε την κατάλληλη έξοδο (π.χ. OP1).

Βήμα 5ο

Δημιουργούμε ένα τελικό σκαλοπάτι και προσθέτουμε μία επαφή IP1. Τοποθετούμε ένα εικονίδιο επαναφοράς και το ορίζουμε σαν C1.

Βήμα 6ο

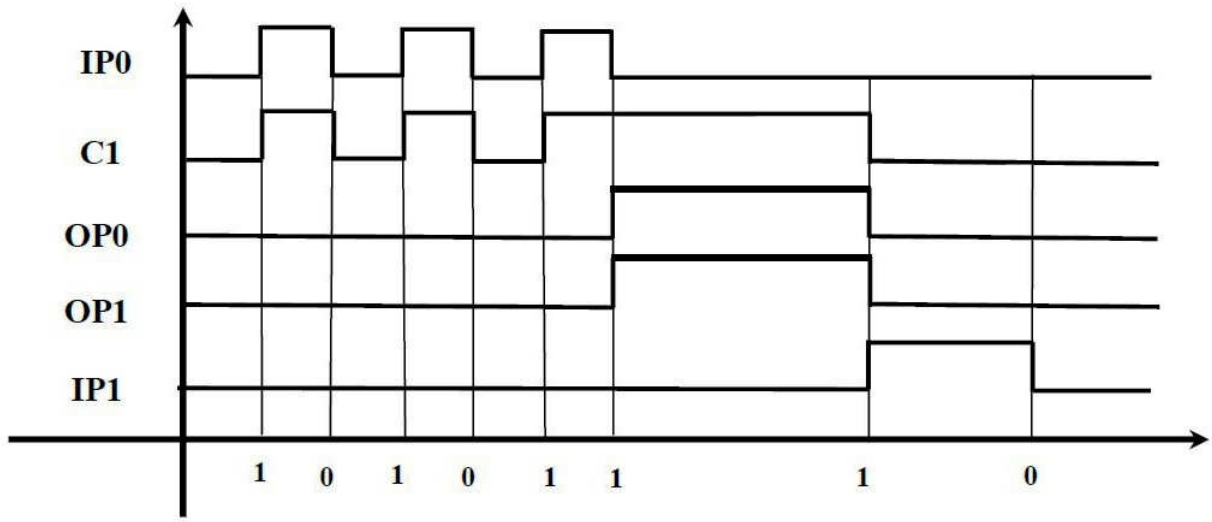
Κάνουμε κλικ στο RUN για να δοκιμάσουμε την λειτουργία του μετρητή κάνοντας την IP0 on και την IP1 off, παρακολουθώντας τις τιμές του μετρητή.



Παρατήρηση: Τρέχουμε το πρόγραμμα μας και ελέγχουμε τη λειτουργία του counter αλλάζοντας την IP0 (on) και IP1 (off) ενώ παράλληλα θα παρακολουθούμε τον προκαθορισμένο counter και τον accumulator. Πειραματιζόμαστε με τη Reset της C1 κάνοντας κλικ πάνω στην IP1 και πραγματοποιώντας τη λειτουργία Reset και παρατηρούμε πως επηρεάζει τον προκαθορισμένο counter και τον accumulator..

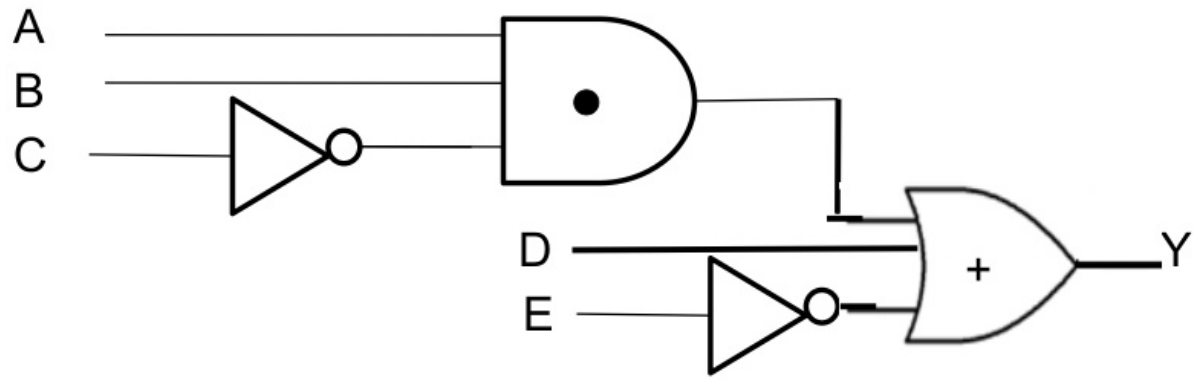
Αποτέλεσμα

Όταν αρχίζει να εκτελείται η προσομοίωση, η IP0 στην αρχή είναι απενεργοποιημένη και το διάγραμμα δεν διαρρέεται από ρεύμα. Αν πατηθεί η IP0 για να κλείσει ο διακόπτης, ο μετρητής C1 θα ξεκινήσει να μετρά προς τα πάνω. Ο μετρητής C1 επιτρέπει στην OP1 και στην OP0 να ενεργοποιηθούν. Όταν λειτουργήσει η IP1, ο μετρητής θα απενεργοποιηθεί (reset), ώστε να απενεργοποιηθούν η OP1 και η OP2, όπως φαίνεται παρακάτω.



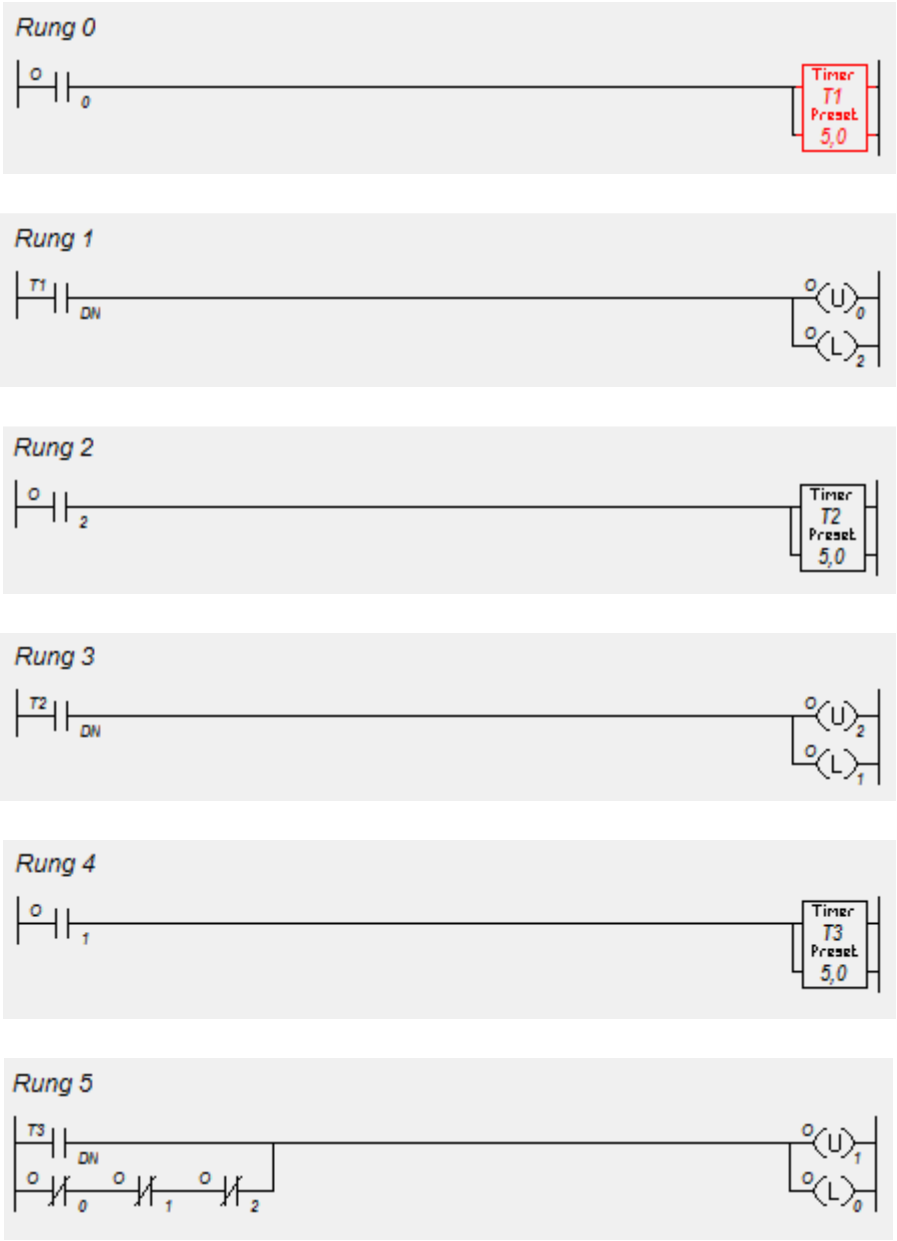
4.5 Εργασίες προς εξάσκηση

1. Υλοποιήστε ένα Ladder διάγραμμα που είναι ισοδύναμο με το ακόλουθο λογικό διάγραμμα



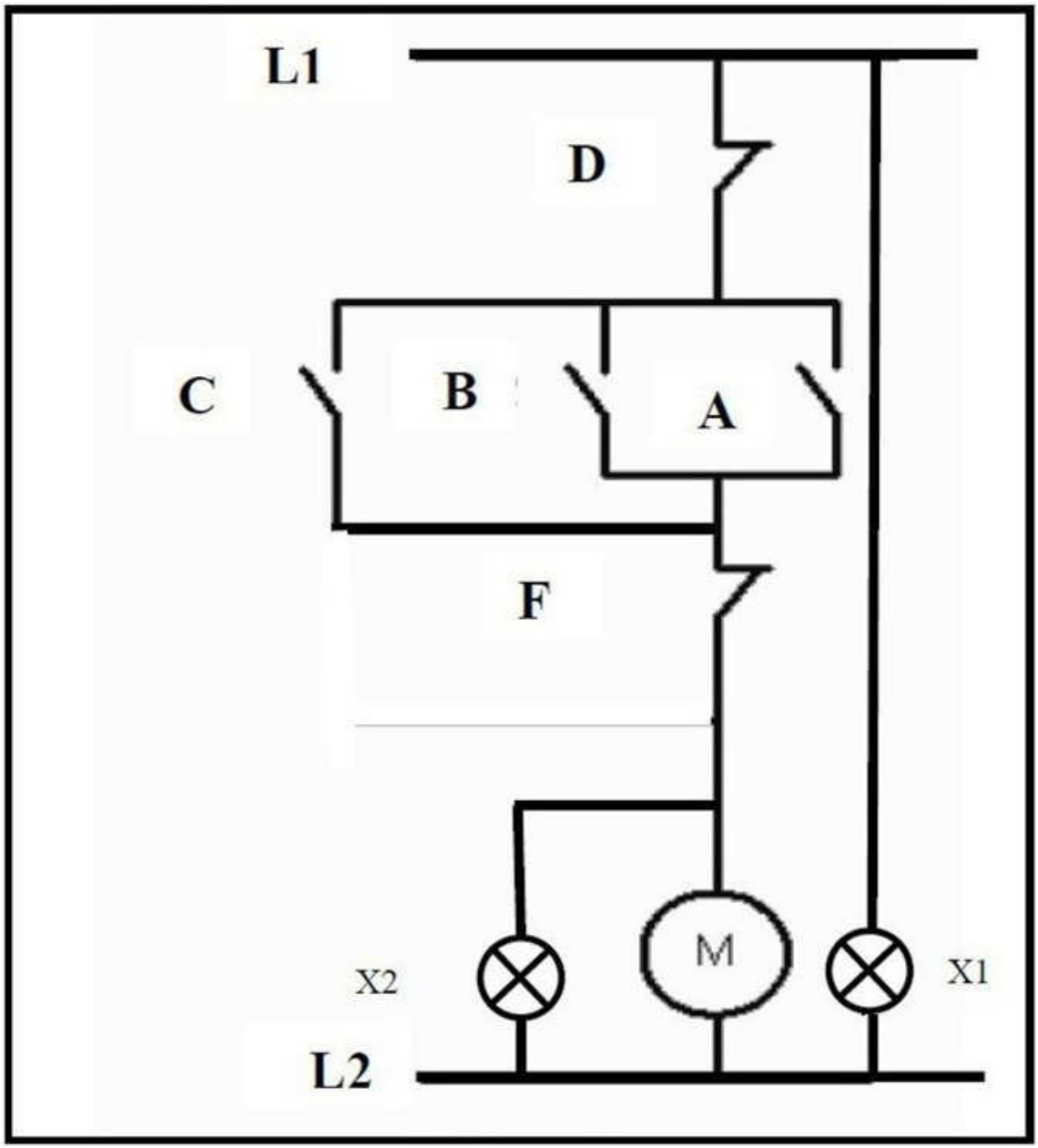
2. Έλεγχος φωτεινής σηματοδότησης

Δίνεται το διάγραμμα Ladder ελέγχου Traffic Light. Υλοποιήστε το στο LADSIM και εξηγήστε αναλυτικά τη λειτουργία του.



3. Χρονοδιακόπτες και Μετρητές

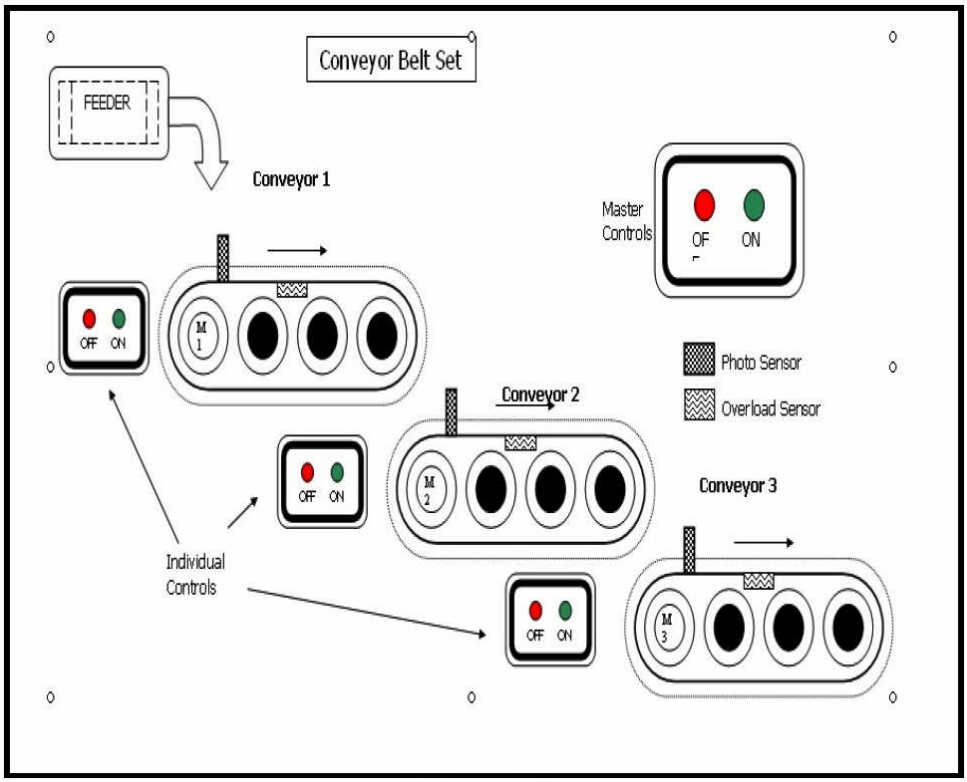
Θεωρούμε το ηλεκτρικό κύκλωμα που φαίνεται παρακάτω στο Σχήμα 33. Έχετε έναν κινητήρα που ελέγχεται από πέντε διακόπτες. Οι διακόπτες A, B και C θα ξεκινάνε τον κινητήρα ενώ οι διακόπτες D και F θα τον σταματάνε. Όταν ο κινητήρας είναι ενεργός η λάμπα X2 θα ανάβει. Επίσης όταν ο κινητήρας δεν θα δουλεύει η λάμπα X1 θα ανάβει. Δημιουργήστε την προσομοίωση του κυκλώματος.



4. Μία σήραγγα αυτοκινήτων είναι εξοπλισμένη με φανάρια στην είσοδο και αισθητήρες και στην είσοδο και στην έξοδο. Επιπλέον, υπάρχουν τρεις ανεμιστήρες διαχειριζόμενοι από κινητήρες (M1, M2 και M3). Υπάρχουν δύο φανάρια (κόκκινο και πράσινο). Το πράσινο φανάρι θα ανάβει όσο τα οχήματα στη σήραγγα είναι λιγότερα από 3 και λειτουργεί ο πρώτος ανεμιστήρας (M1). Εάν ο αριθμός των οχημάτων στη σήραγγα είναι μεγαλύτερος από 3 και μικρότερος από 5, θα λειτουργούν ο πρώτος και ο δεύτερος ανεμιστήρας (M1, M2). Όταν ο αριθμός των οχημάτων υπερβεί τα 5, τότε θα λειτουργήσουν και οι 3 ανεμιστήρες (M1, M2 και M3). Αν ο αριθμός των οχημάτων φτάσει στα 10, θα ανάψει το

κόκκινο φανάρι, για να αποτραπεί η είσοδος άλλου οχήματος. Δημιουργήστε την προσομοίωση του προγράμματος.

5.



Θεωρούμε το ηλεκτρικό κύκλωμα που φαίνεται στο παραπάνω. Ένας πελάτης, ιδιοκτήτης τυπογραφείου, θέλει να μεταφέρεται η τυπωμένη εφημερίδα από την αίθουσα εκτύπωσης στο σημείο διανομής με ένα συνδυασμό ιμάντων μεταφοράς. Ο συνδυασμός των ιμάντων μεταφοράς αποτελείται από μία σειρά τριών μικρών ιμάντων που λειτουργούν με κινητήρες. Ο ιμάντας λειτουργεί ως εξής:

Το σύστημα έχει ένα μόνο κύριο κουμπί έναρξης (MSTB) και ένα κύριο κουμπί διακοπής (MSPB), τα οποία ελέγχουν όλους τους ιμάντες.

Κάθε ιμάντας μεταφοράς είναι εξοπλισμένος με αισθητήρες οι οποίοι ανιχνεύουν την υπερφόρτωση. Η υπερφόρτωση μπορεί να προκαλέσει υπερθέρμανση και κατά συνέπεια σοβαρές βλάβες στο σύστημα μεταφοράς.

Επίσης, εάν η λειτουργία του συστήματος μεταφοράς σταματήσει για οποιοδήποτε λόγο, τότε είναι σημαντικό να σταματήσει ένας ιμάντας η και όλοι οι ιμάντες για να μην καταστραφεί το φορτίο. Επομένως:

Εάν υπάρχει υπερφόρτωση στον ιμάντα 1 (OVL1) θα πρέπει να σταλεί σήμα και να σταματήσει αμέσως, ενώ οι άλλοι 2 ιμάντες να συνεχίσουν να λειτουργούν.

Εάν υπάρχει υπερφόρτωση στον ιμάντα 2 (OVL2) θα πρέπει να σταλεί σήμα και να σταματήσουν αμέσως οι ιμάντες 1 και 2, ενώ ο ιμάντας 3 να συνεχίσει να λειτουργεί.

Εάν υπάρχει υπερφόρτωση στον ιμάντα 3 (OVL3) θα πρέπει να σταλεί σήμα και να σταματήσουν αμέσως όλοι οι ιμάντες (1, 2 και 3).

Κατάλογος εισόδων και εξόδων :

Ο κατάλογος εισόδων και εξόδων παρέχει μια σαφή κατανόηση του συστήματος, ώστε να μπορέσει να βοηθήσει στην κατασκευή του διαγράμματος.

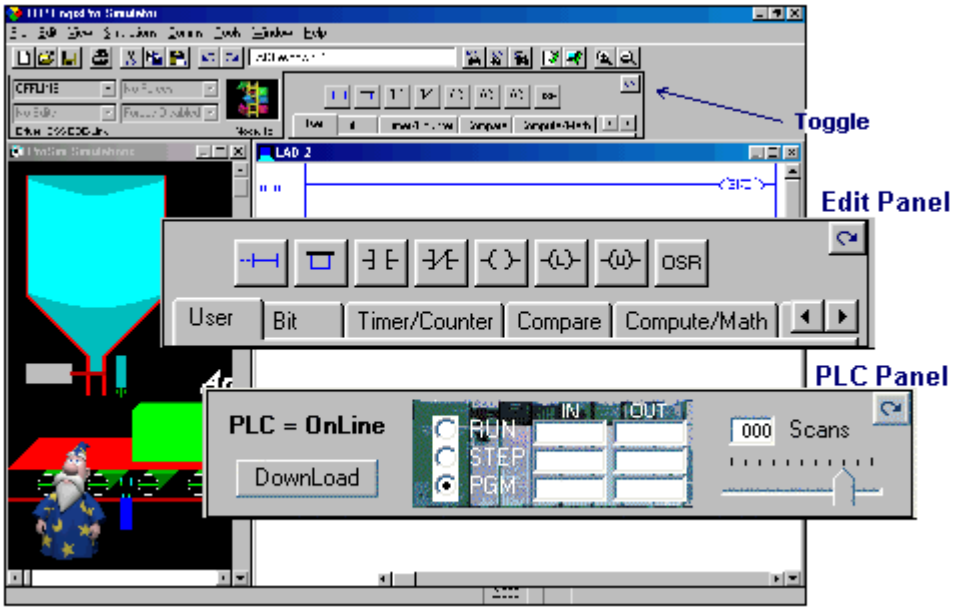
Είσοδος	**Αντιστοιχία**
Κύριο Κουμπί Παύσης	IP0
Κύριο Κουμπί Εκκίνησης	IP1
Ανιχνευτής υπερφόρτωσης 1	IP2
Ανιχνευτής υπερφόρτωσης 2	IP3
Ανιχνευτής υπερφόρτωσης 3	IP4
Αισθητήρας 1	IP5
Αισθητήρας 2	IP6
Αισθητήρας 3	1P7
Κουμπί ιμάντα 1 Εκκίνησης	IP8
Κουμπί ιμάντα 1 Παύσης	1P8
Κουμπί ιμάντα 2 Εκκίνησης	IP9
Κουμπί ιμάντα 2 Παύσης	IP9
Κουμπί ιμάντα 3 Εκκίνησης	1P10
Κουμπί ιμάντα 3 Παύσης	1P10

Είσοδος	**Αντιστοιχία**
Feeder	OP0
Κινητήρας 1	OP1
Κινητήρας 2	OP2
Κινητήρας 3	OP3
Κινητήρας 1 & Λάμπα 1	OP4
Κινητήρας 2 & Λάμπα 2	OP5
Κινητήρας 3 & Λάμπα 3	OP6
Συναγερμός	OP10

ΚΕΦΑΛΑΙΟ 5

ΠΡΟΣΟΜΟΙΩΤΗΣ PLC - LOGIXPRO

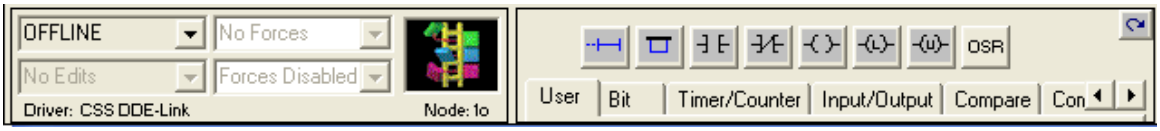
5.1 Περιγραφή του προγράμματος προσομοίωσης PLC –LOGIXPRO



Σχήμα 5.1: LOGIXPRO SIMULATOR

Το LogixPro είναι ένα εκπαιδευτικό εργαλείο που χρησιμοποιείται για τη διδασκαλία του προγραμματισμού των PLC . Προσφέρει οπτικά βοηθήματα για να βοηθήσει τους προγραμματιστές PLC να δουν τα προγράμματά τους σε δράση , όπως την κατασκευή τους , επιτρέποντάς τους αλλαγές και βελτιώσεις

Στο πάνω μέρος της οθόνης και κάτω από τη βασική γραμμή εργαλείων υπάρχουν τα χειριστήρια του προγράμματος σε δύο εναλλασσόμενες οθόνες.



Η εναλλαγή μεταξύ των οθονών γίνεται με το πλήκτρο  .

Όταν το σύστημα είναι Offline, μπορείτε να κάνετε αλλαγές στο Ladder διάγραμμα του PLC, τοποθετώντας νέα στοιχεία, με τη διαδικασία drag &drop, ή να διαγράψετε τα παλιά.

Για να προσθέσετε ένα στοιχείο στο διάγραμμα, κάντε τα εξής:

Αναζητήστε το στοιχείο που θέλετε κάνοντας κλικ στην κατάλληλη καρτέλα της οθόνης Offline. Η καρτέλα που θα χρησιμοποιήσετε γι αυτή την άσκηση είναι η καρτέλα User.

Κάντε κλικ στο στοιχείο που θέλετε και κρατήστε το αριστερό πλήκτρο του ποντικιού σας πατημένο.

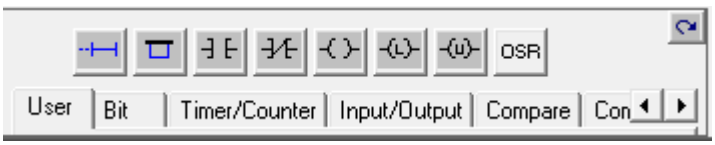
Σύρετε το στοιχείο μέχρι το διάγραμμα.

Μετακινήστε το στοιχείο σε ένα από τα κόκκινα τετράγωνα που έχουν εμφανιστεί και δηλώνουν τις επιτρεπτές θέσεις στις οποίες μπορεί να τοποθετηθεί το στοιχείο.

Αφήστε το αριστερό πλήκτρο του ποντικιού σας όταν το τετράγωνο γίνει πράσινο, για να τοποθετηθεί το στοιχείο. Πληκτρολογήστε το όνομα του στοιχείου και πατήστε Enter για να καταχωρηθεί.

5.2 Relay Logic

Για το RELAY LOGIC χρησιμοποιήσαμε την ακόλουθη εργαλειοθήκη:

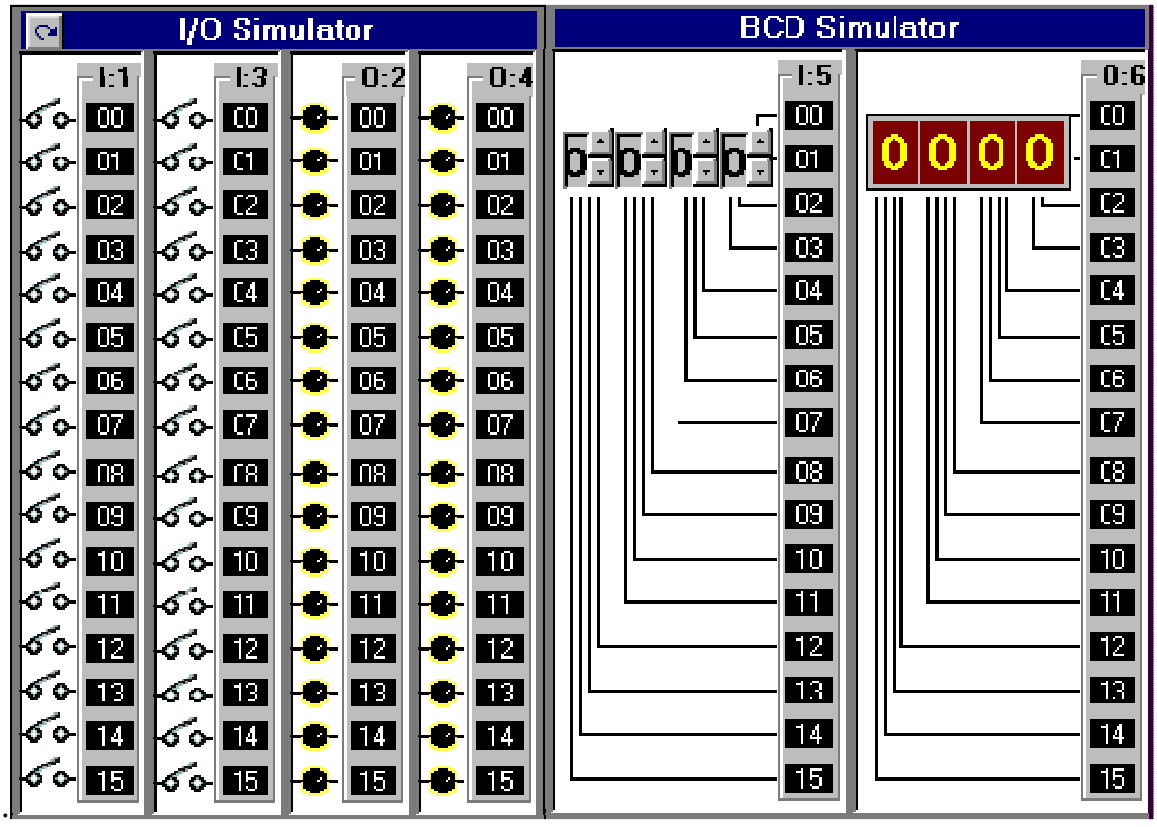


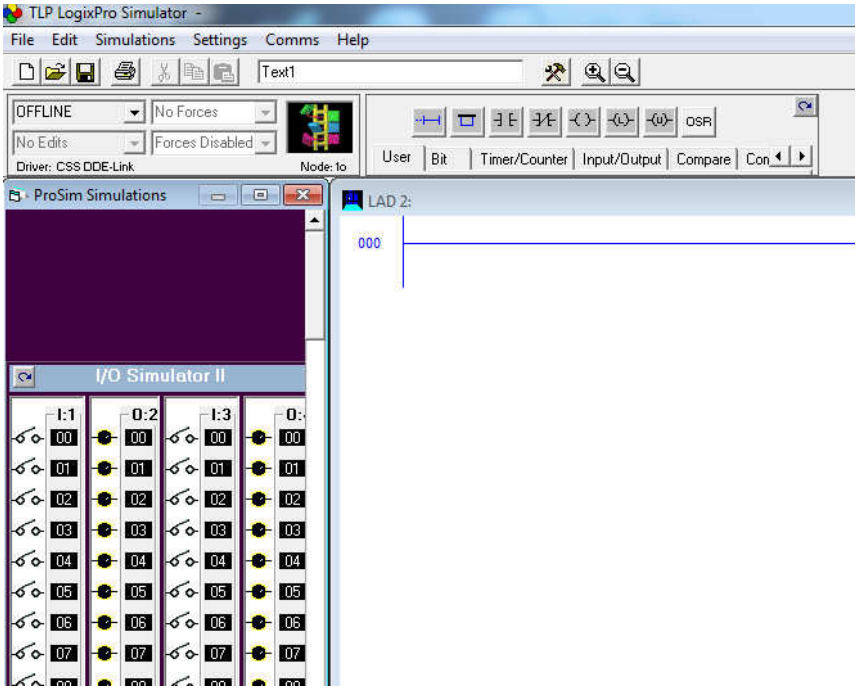
Σχήμα 5.1: Relay Logic

Μπορούμε να χρησιμοποιήσουμε το εργαλείο examine if close(αν ο διακόπτης είναι ενεργός) ή examine if open(αν ο διακόπτης είναι κλειστός). Στη συνέχεια τοποθετούμε εξόδους με το εξής σύμβολο. Πατώντας στο Simulations επιλέγουμε το I/O Simulator.

ΒΗΜΑ 1:

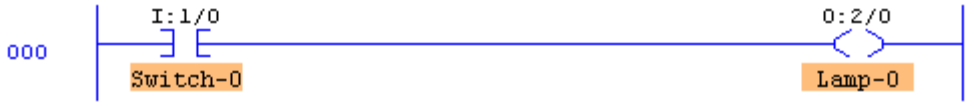
Από το "Simulations Menu" , επιλέγουμε το "I/O Simulation" και πατάμε τη μεγιστοποίηση ώστε να φανεί στην οθόνη μας ο I/O Simulator. Παρατηρούμε ότι περιέχει 32 κουμπιά και φωτάκια. Πατώντας τους διακόπτες, αλλάζει το χρώμα του terminal στα δεξιά. Στη συνέχεια πατάμε πάλι το κουμπί "Maximize" για να δούμε καθαρά τον I/O Simulator, αλλά και το κανονικό πρόγραμμα.





ΒΗΜΑ 2:

Δημιουργούμε το ακόλουθο πρόγραμμα:

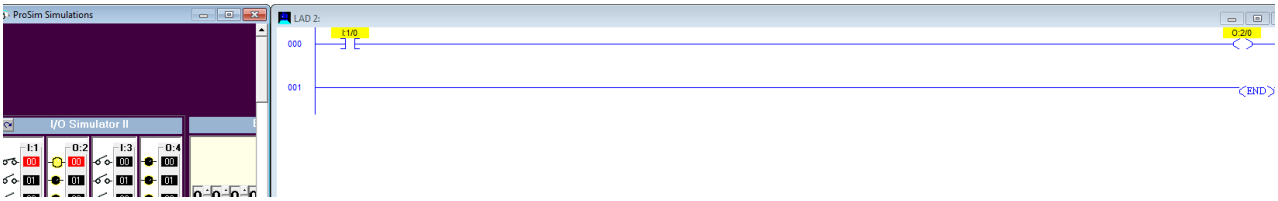


ΒΗΜΑ 3:

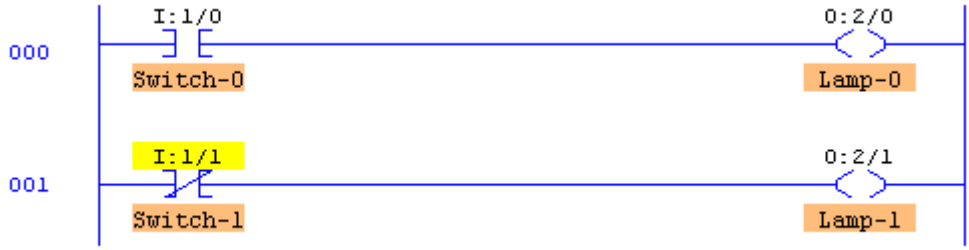
Κάνουμε Download το πρόγραμμα στο PLC. Κλικάρουμε το κουμπί στη δεξιά γωνία του Edit Panel που θα φέρει το PLC Panel σε θέα. Πατάμε το Download για να αρχίσει το κατέβασμα στο PLC. Όταν τελειώσει, πατάμε το Run.

ΒΗΜΑ 4:

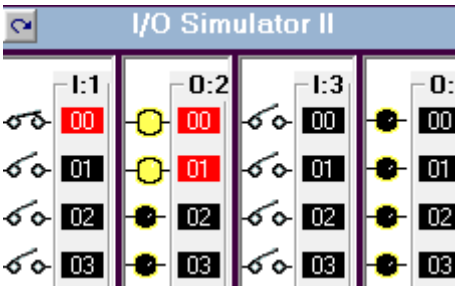
Αν πατήσουμε το διακόπτη I:1/0 στον simulator, η λάμπα O:2/00 θα ανάψει.



ΒΗΜΑ 5: Επαναφέρατε το PLC Offline και προσθέστε ένα νέο rung στο πρόγραμμα σας όπως φαίνεται παρακάτω.



ΒΗΜΑ 6: Τρέξτε το πρόγραμμα και παρατηρήστε ότι για να ανάψουν οι λάμπες, ο διακόπτης I:1/0 πρέπει να είναι κλειστός ενώ ο διακόπτης I:1/1 ανοικτός.

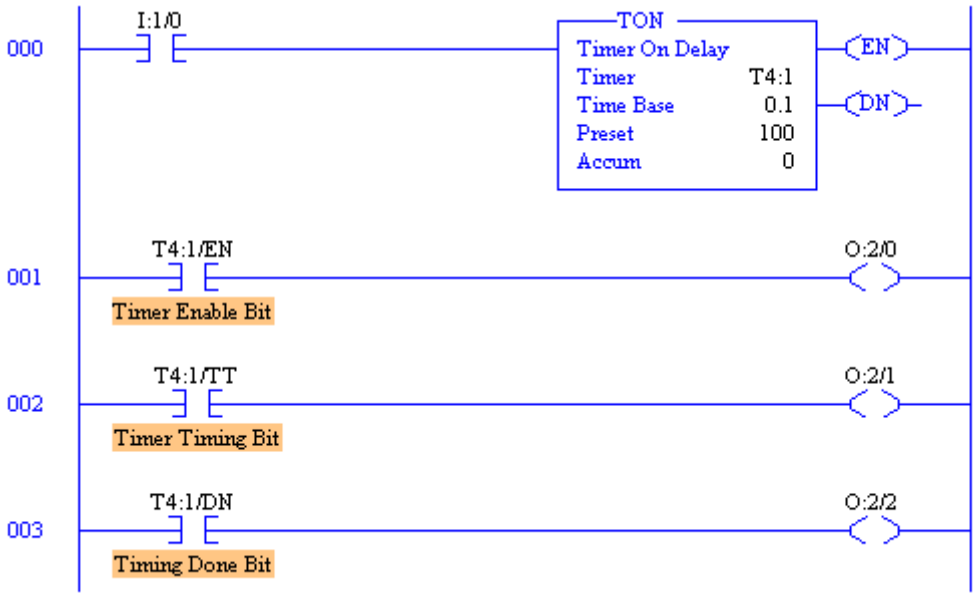


5.3 RSLogix Timers

Από το LogixPro Simulations Menu, επιλέξτε I/O Simulation.

Καθαρίστε οποιοδήποτε πρόγραμμα επιλέγοντας "New" στο File menu, και στη συνέχεια "Clear Data Table" στο Simulations menu.

Υλοποιήστε το ακόλουθο πρόγραμμα:



Εξηγήστε τι αντιπροσωπεύει η τιμή preset value =100 as the timer's preset value.

Κάντε download το πρόγραμμα στο PLC.

Σιγουρευτείτε ότι ο διακόπτης I:1/0 είναι Open, και θέστε το PLC στο Run mode.

Με δεξί κλικ στον Timer επιλέξτε "GoTo DataTable" από το drop-down menu.

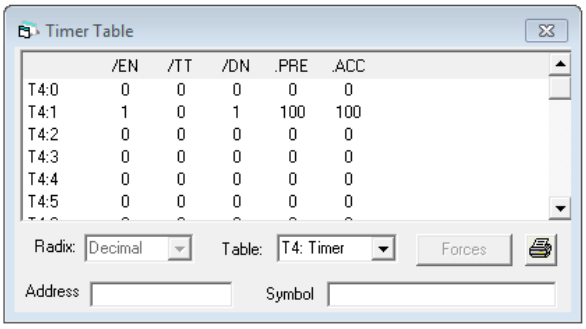
Σημειώστε τις παρακάτω αρχικές τιμές:

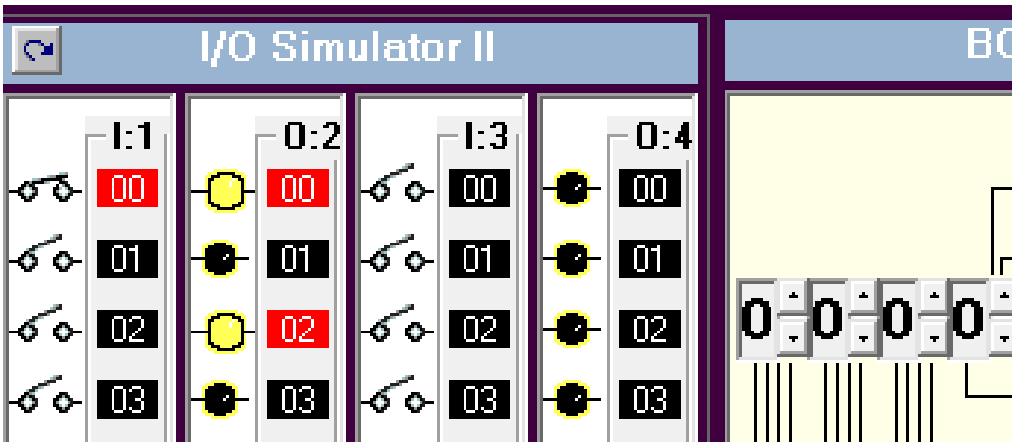
T4:1.ACC = _____ T4:1.PRE = _____ T4:1/EN = ____ T4:1/TT = ____ T4:1/DN = ____

Κλείστε τον διακόπτη I:1/0 και σημειώστε τις ακόλουθες τελικές τιμές:

T4:1.ACC = _____ T4:1.PRE = _____ T4:1/EN = ____ T4:1/TT = ____ T4:1/DN = ____

.





Επαναλάβατε την προηγούμενη διαδικασία και παρατηρήστε το πρόγραμμα σας με ένα Timer Off Delay (TOF).

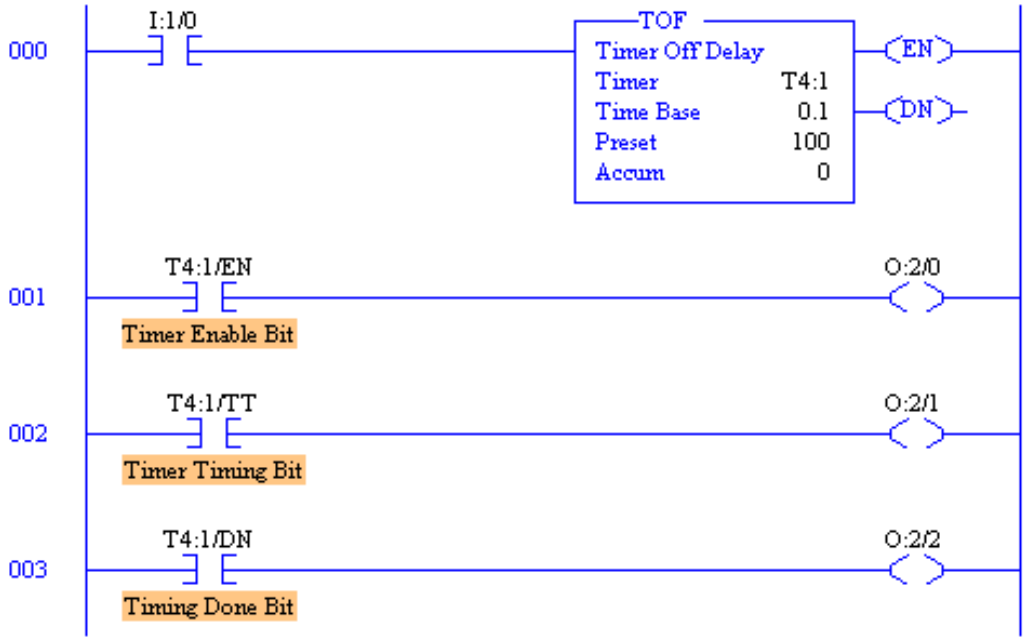
Συμπληρώστε τις ακόλουθες τιμές:

Initial State (Switch I:1/0=Closed):

T4:1.ACC = _____ T4:1.PRE = _____ T4:1/EN = ____ T4:1/TT = ____ T4:1/DN = ____

Final State (Switch I:1/0=Open):

T4:1.ACC = _____ T4:1.PRE = _____ T4:1/EN = ____ T4:1/TT = ____ T4:1/DN = ____



5.4 Παραδείγματα

5.4.1 Αυτόματη Εκφόρτωση Σιλό (Silo Simulation)

Η λογική της εφαρμογής έχει ως εξής: Υπάρχει μια βιομηχανία που γεμίζει με υγρό μερικά κουτιά. Έχουμε 3 καταστάσεις: Η Α κατάσταση γεμίζει ολόκληρο το κουτί, η Β κατάσταση κάνει παράκαμψη γεμίσματος και η Γ κατάσταση γεμίζει ένα μέρος του κουτιού. Με το Start ξεκινάμε το σύστημα να τρέχει και με το Stop το σταματάμε. Όταν το κουτί φτάσει στον Prox sensor, αυτός ενεργοποιείται και ανοίγει η Solenoid valve για να γεμίσει το κουτί. Όσο γεμίζει το κουτί, είναι ανοιχτό το Fill light και είναι απενεργοποιημένο το Motor. Όταν γεμίσει το κουτί, ενεργοποιείται ο Level sensor, απενεργοποιείται ο Solenoid valve, ενεργοποιείται το Motor και ανάβει το Full light. Το Run είναι ενεργοποιημένο όσο τρέχει το σύστημα.

Από το menu Simulations, επιλέξτε Silo Simulation.

Από το menu File επιλέξτε Open και silo.rsl.

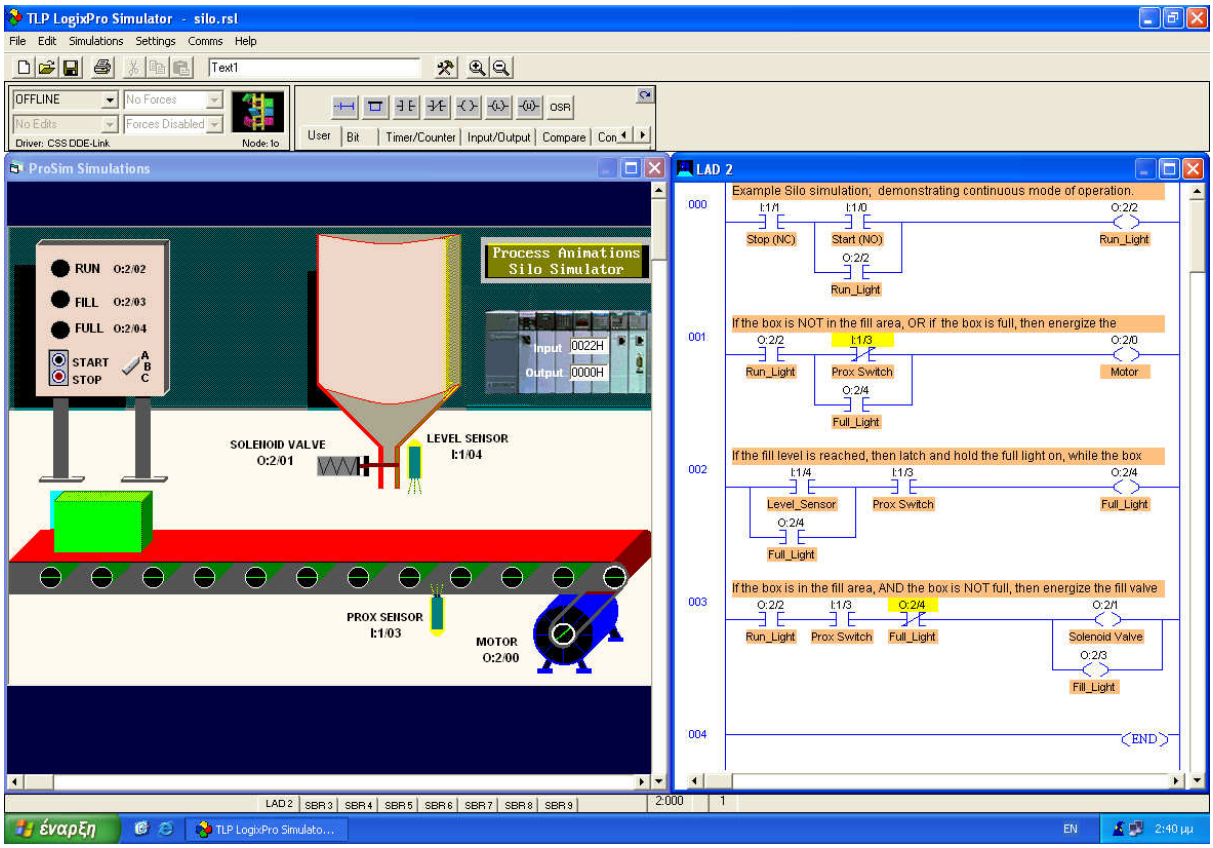
Το περιβάλλον του προγράμματος είναι παραθυρικό. Επομένως μετά από μια μικρή τακτοποίηση παραθύρων, η οθόνη σας θα πρέπει να δείχνει όπως στην επόμενη εικόνα.

Στο αριστερό παράθυρο της εφαρμογής εμφανίζεται η προσομοίωση του συστήματος, ενώ στο δεξί παράθυρο υπάρχει το πρόγραμμα του PLC σε γλώσσα Ladder.

Πατήστε το πλήκτρο  για να μεταφερθείτε στην οθόνη Online. Από την κατάσταση PGM (προγραμματισμού στην οποία βρίσκεστε, πατήστε το πλήκτρο DownLoad. Με τον τρόπο αυτό το πρόγραμμα από τη δεξιά οθόνη «φορτώνεται» στο PLC.

Κάντε κλικ στην κατάσταση RUN. Τώρα μπορείτε να ελέγξετε τη λειτουργία του προγράμματος PLC, από την οθόνη προσομοίωσης.

Πατήστε το μπουτόν START στην οθόνη προσομοίωσης με το διακόπτη επιλογής να είναι στη θέση "A" και παρατηρήστε ότι το σύστημα εργάζεται σε συνεχόμενη λειτουργία.



Παρατηρήστε ότι το χρώμα του πλαισίου ονόματος των στοιχείων που διαρρέονται από ρεύμα, αριστερά, στο ladder διάγραμμα, γίνεται κίτρινο. Εξηγήστε αναλυτικά τι συμβαίνει κατά τη διάρκεια της προσομοίωσης σε κάθε κλάδο. Στην εξήγησή σας, χρησιμοποιείστε τις λογικές εκφράσεις ΚΑΙ και Η.

ΑΠΑΝΤΗΣΗ:

Κλαδος 0:

Με αυτόν τον κλάδο επιτυγχάνουμε τη συνεχή λειτουργία του συστήματος μας. Το stop στην αρχή (είναι normally closed άρα αν δεν ενεργοποιηθεί δεν διακόπτει τη λειτουργία του κλάδου). Μέτα έχουμε παράλληλα το start(που είναι normally open άρα αν δεν ενεργοποιηθεί διακόπτει τη λειτουργία του κλάδου) και το run_light. Τέλος σαν έξοδο έχουμε το run_light. Όταν ο χρήστης πατήσει το start τότε θα ανάψει το run_light όμως το start είναι button και όχι κάποιος διακόπτης άρα θα πρέπει να το πιέζουμε συνεχώς γι αυτό έχουμε βάλει σε παραλληλία(λογικό 'Η) με στο start το run_light όποτε τροφοδοτείται από τον ίδιο τον εαυτό πιέζοντας το μόνο μια φορά.

Κλάδος 1:

Με αυτόν τον κλάδο επιτυγχάνουμε τον έλεγχο για το πότε θα πρέπει να ενεργοποιείται ο κινητήρας μεταφοράς του συστήματος μας. Στην αρχή έχουμε το run_light οποί είναι η έξοδος του κλάδου 0 έτσι αν δεν έχουμε ενεργοποιηθεί το κλάδο 0 δε μπορεί να έρθει σε λειτουργία ο κλάδος 1. Μέτα έχουμε το prox switch που είναι συνδεδεμένο σε παραλληλία (λογικό 'Η) με το full_ligh όπου εδώ γίνεται ο έλεγχος για το πότε θα είναι σε λειτουργία ο κινητήρας μεταφοράς εδω έχουμε ακόμα σε σειρά το A το οποίο άπλα καθορίζει τη λειτουργιά των διακοπτών(από τη στιγμή που ο διακόπτης είναι στη θέση A δεν επηρεάζεται η λειτουργιά του). Αν το prox switch (που μας δείχνει αν το κουτί είναι σε θέση γεμίσματος) είναι ενεργοποιημένο και το full light(που μας δείχνει αν το κουτί είναι γεμάτο) δεν είναι ενεργοποιημένο τότε ο κινητήρας μεταφοράς είναι σταματημένος. Αντίθετα αν το full light είναι ενεργοποιημένο και το prox switch όχι τότε έχουμε κίνηση από το κινητήρα μεταφοράς.

Κλάδος 2:

Με αυτόν τον κλάδο επιτυγχάνουμε τον έλεγχο για το πότε είναι σε λειτουργία γεμίσματος το σύστημά μας και πότε γέμισε το κουτί μας ούτως ώστε να επιστρέψουμε στο κλάδο 1 για να επαναληφθεί η διαδικασία. Στην αρχή έχουμε σε παραλληλία(λογικό 'Η) το level_sensor(που είναι υπεύθυνο για το πότε το κουτί είναι γεμάτο) με το full_light (που μας δείχνει αν το κουτί είναι γεμάτο). Έπειτα έχουμε σε σειρά το prox swith (που μας δείχνει αν το κουτί είναι σε θέση γεμίσματος). Εδώ για να πάρουμε την έξοδο μας που θα μας δείξει πως το κουτί γέμισε θα πρέπει το level_sensor να είναι ενεργοποιημένο και

το prox switch να είναι και αυτό αλλιώς δε θα ενεργοποιηθεί πότε το full_light. Αυτό σημαίνει πως το κουτί θα είναι στη θέση γεμίσματος και εκείνη τι στιγμή θα δεχθεί το σήμα πως γέμισε για να ενεργοποιηθεί το full_light και να επιστρέψουμε μέσω αυτού στον κλάδο 1.

Κλάδος 3:

Με αυτόν τον κλάδο επιτυγχάνουμε τον έλεγχο για το αν το σύστημα μας είναι στη περιοχή γεμίσματος, τότε ανοίγει η βαλβίδα για να γεμίσει το κουτί. Τέλος σε αυτόν τον κλάδο έχουμε σε σειρά τα run_light ,prox switch και full_light. Εδω αν το run_light και το prox switch είναι ενεργοποιημένα και το full_light δεν είναι τότε είμαστε σε κατάσταση γεμίσματος και ενεργοποιείται η βαλβίδα ούτως ώστε να γεμίσει το κουτί μας. Ταυτόχρονα ενεργοποιείται και το fill_ligh που μας δείχνει πως είμαστε σε κατάσταση γεμίσματος.Πατήστε το STOP στην οθόνη προσομοίωσης και στη συνέχεια μεταβείτε στην οθόνη Offline.

Τροποποιήστε ή ξαναγράψτε το πρόγραμμα σας έτσι ώστε να τοποθετεί και να φορτώνει τους κάδους που κινούνται πάνω στη ταινία μεταφοράς. Σιγουρέψτε ότι εκπληρώνονται οι πιο κάτω απαιτήσεις:

Η διαδικασία πρέπει να μπορεί να σταματάει και να συνεχίζει οποιαδήποτε στιγμή, χρησιμοποιώντας τους διακόπτες "Stop" και "Start" του πίνακα ελέγχου.

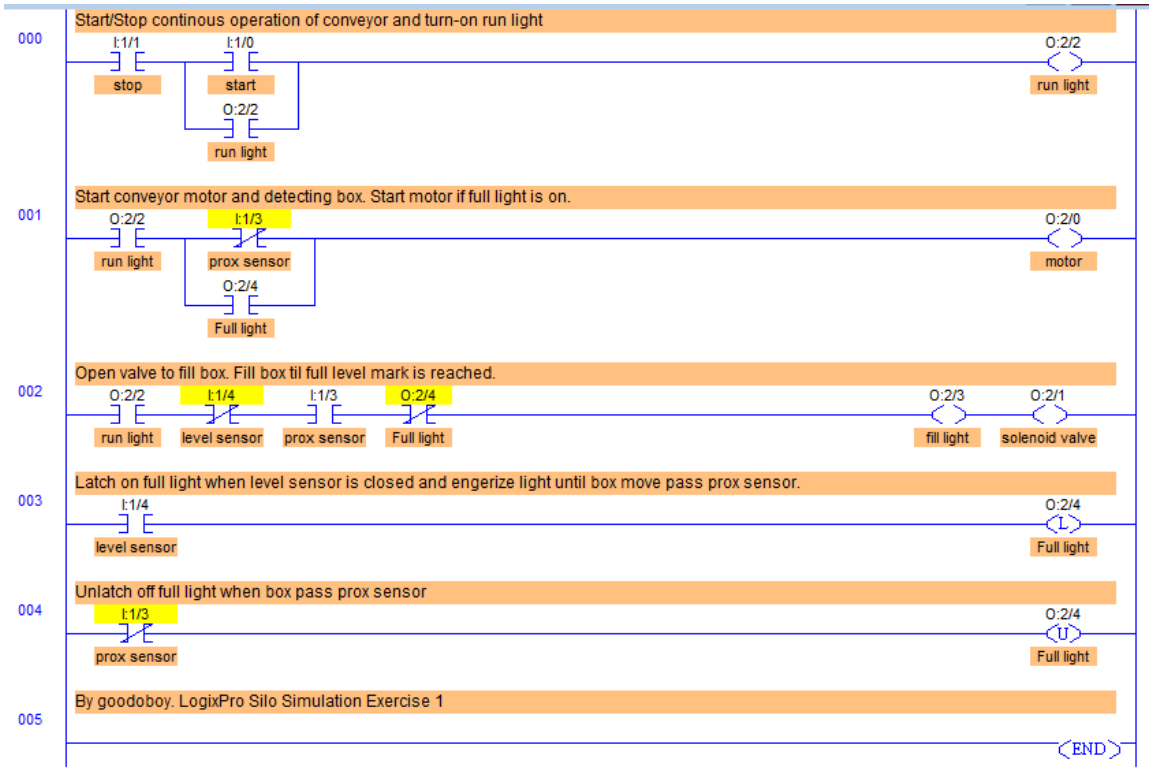
Η λυχνία "RUN" να παραμένει αναμμένη καθ'όλη τη διάρκεια της αυτόματης λειτουργίας.

Η λυχνία "RUN" , το μοτέρ κίνησης της ταινίας μεταφοράς και η βαλβίδα "Solenoid" θα πρέπει να απενεργοποιούνται οποτεδήποτε το σύστημα σταματάει με τη χρήση του διακόπτη "STOP".

Η λυχνία "FILL" πρέπει να είναι αναμμένη, καθ'όλη τη διάρκεια που ο κάδος γεμίζει.

Η λυχνία "FULL" να ενεργοποιείται όταν ο κάδος είναι πλήρως γεμάτος και να παραμένει αναμμένη εως ότου ο κάδος περάσει εντελώς το διακόπτη - "prox-sensor".

ΑΠΑΝΤΗΣΗ:



Τροποποιήστε ή ξαναγράψτε το πρόγραμμα σας έτσι ώστε όταν ο διακόπτης επιλογής είναι στη θέση "C", το σύστημα θα εργάζεται σε Γέμισμα κάδου με χειροκίνητη επανάληψη . Σε αυτό το τρόπο λειτουργίας, θα πρέπει να εκπληρώνονται οι πιο κάτω απαιτήσεις:

Η ταινία μεταφοράς να σταματάει όταν η δεξιά άκρη του κάδου ενεργοποιήσει το διακόπτη "prox-sensor".

Με τον κάδο στη πιο πάνω θέση και την ταινία μεταφοράς σταματημένη, ανοίγει η βαλβίδα "solenoid valve" για τη φόρτωση του κάδου. Η φόρτωση πρέπει να σταματήσει όταν ο αισθητήρας "Level sensor" γίνει λογικό "1".

Η λυχνία "FILL" πρέπει να είναι αναμμένη μόνο κατά τη διάρκεια φόρτωσης του κάδου.

Η λυχνία "FULL" πρέπει να ανάψει όταν ο κάδος είναι πλήρως φορτωμένος και να σβήσει όταν ο κάδος ξεπεράσει εντελώς τον διακόπτη "prox-sensor".

Όταν ο κάδος είναι πλήρως γεμάτος, στιγμιαίο πάτημα του διακόπτη "Start" θα πρέπει να εκκινεί την ταινία μεταφοράς μετακινώντας τον κάδο δεξιά. Στη συνέχεια, και στο αριστερό άκρο της ταινίας θα εμφανίζεται νέος κάδος για να επαναληφθεί η όλη διαδικασία.
Ακόμα κι αν κρατήσουμε συνεχώς πατημένο το διακόπτη "Start", ο νέος κάδος δε θα ξεπεράσει το σημείο φόρτωσης αλλά θα σταματήσει κανονικά στη θέση φόρτωσης.

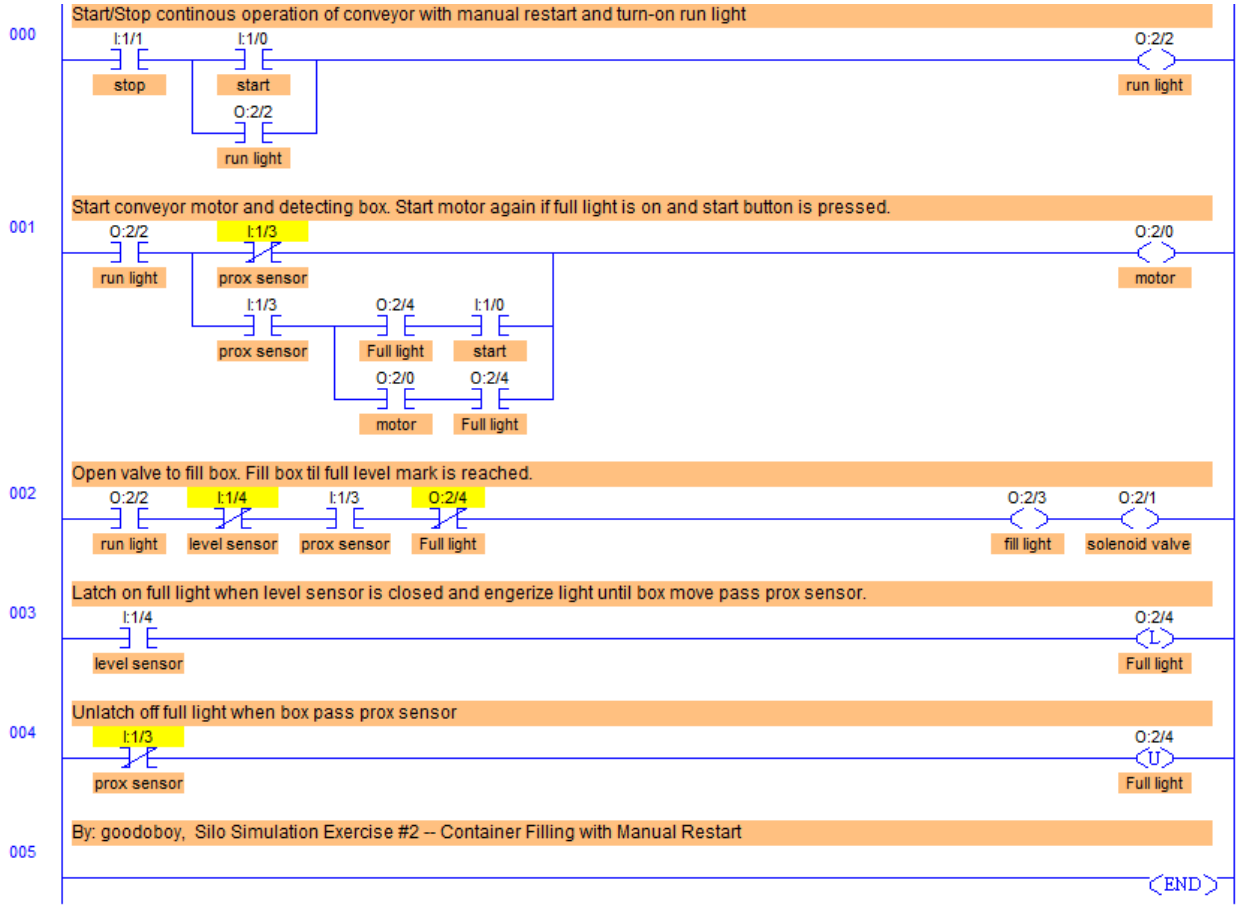
ΑΠΑΝΤΗΣΗ:

Πρέπει να τροποποιήσουμε το σύστημα μας ώστε όταν είναι ο διακόπτης στη θέση C ο κάδος να γεμίζει κανονικά άλλα να μην ξεκινάει αν εμείς δεν του δώσουμε το έναυσμα.

Η αλλαγή που πρέπει να κάνουμε αφορά τον κλάδο 1 εκεί όπου γίνεται η κίνηση από τον κινητήρα μεταφοράς.

Στο σημείο που έχουμε την παράλληλη σύνδεση θα προσθέσουμε ακόμα μια γραμμή παράλληλα με πρώτο στοιχείο της τον διακόπτη C για να λειτουργεί μόνο αν είναι σε λειτουργία ο διακόπτης. Μέτα τον διακόπτη μας θα συνδέσουμε παράλληλα το start με το motor (βλέπε εικόνα παρακάτω) ούτως ώστε αφού γεμίσει ο κάδος μας να προχωρήσει αν πατήσουμε εμείς το start.

Όμως επειδή το start είναι button βάζουμε παράλληλα το motor όπως το κάναμε παραπάνω με παρόμοιο τρόπο ώστε να τροφοδοτήσουμε ξανά την έξοδο χωρίς να πιέζουμε συνεχεία το start.

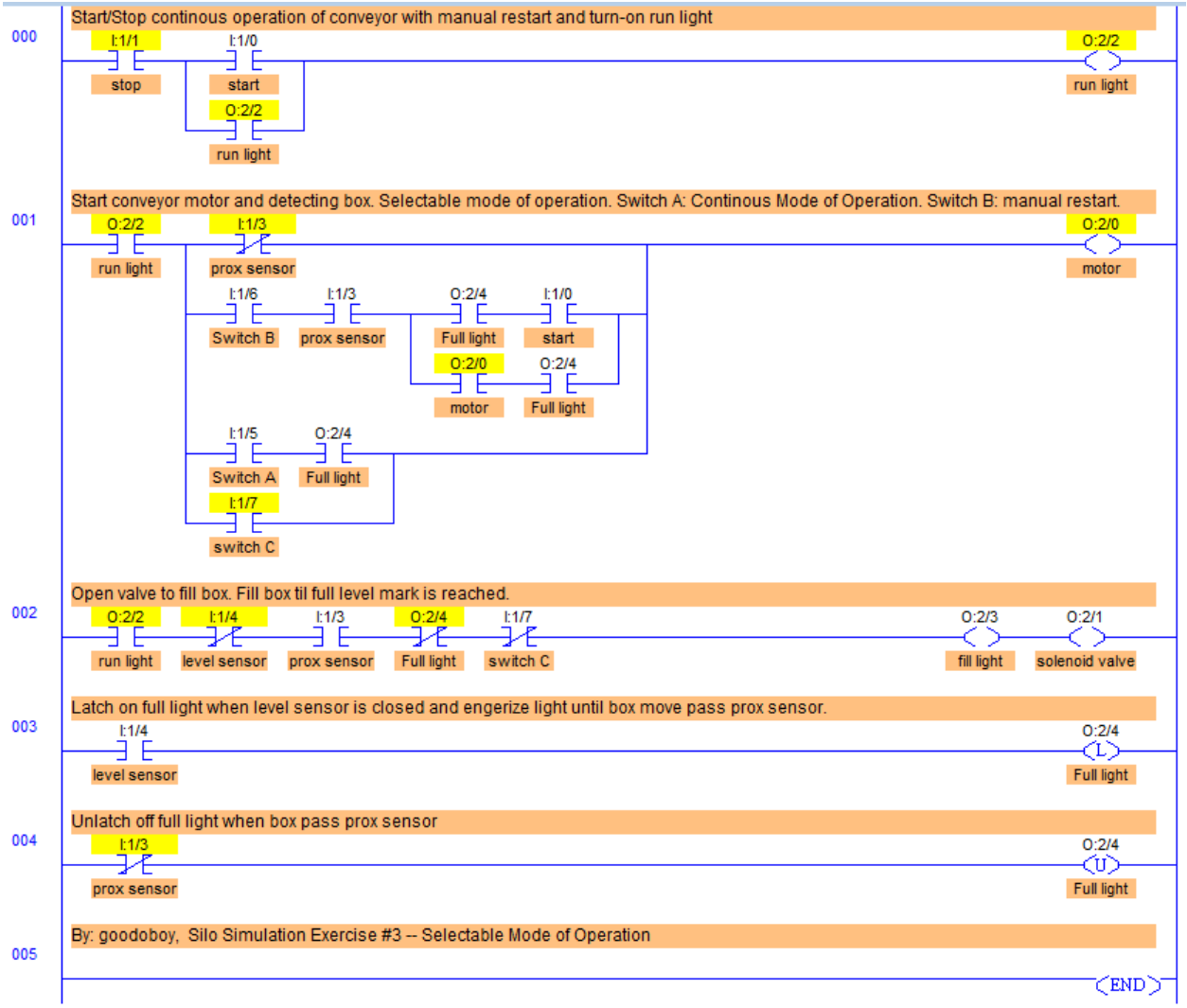


Τροποποιήστε ή ξαναγράψτε το πρόγραμμα σας ετσι ώστε να εκπληρώνει τις πιό κάτω απαιτήσεις:

Όταν ο διακόπτης επιλογής είναι στη θέση "A", το σύστημα θα εργάζεται σε συνεχόμενη λειτουργία . Είναι ο τρόπος λειτουργίας της Περίπτωσης 10.

Όταν ο διακόπτης επιλογής είναι στη θέση "B", το σύστημα θα εργάζεται σε Γέμισμα κάδου με χειροκίνητη επανάληψη . Είναι ο τρόπος λειτουργίας της Περίπτωσης 11.

Όταν ο διακόπτης επιλογής είναι στη θέση "C", το σύστημα θα εργάζεται σε Παράκαμψη γεμίσματος . Σε αυτό το τρόπο λειτουργίας, παρακάμπτεται η λειτουργία φόρτωσης και οι κάδοι θα περνούν χωρίς να φορτώνονται. Οπως και στους άλλους δυο τρόπους λειτουργίας, οι πιεστικοί διακόπτες "Start" και "Stop" θα ελέγχουν τη κίνηση της ταινίας, καθως και η λυχνία "Run" θα λειτουργεί κατά τα αναμενόμενα.



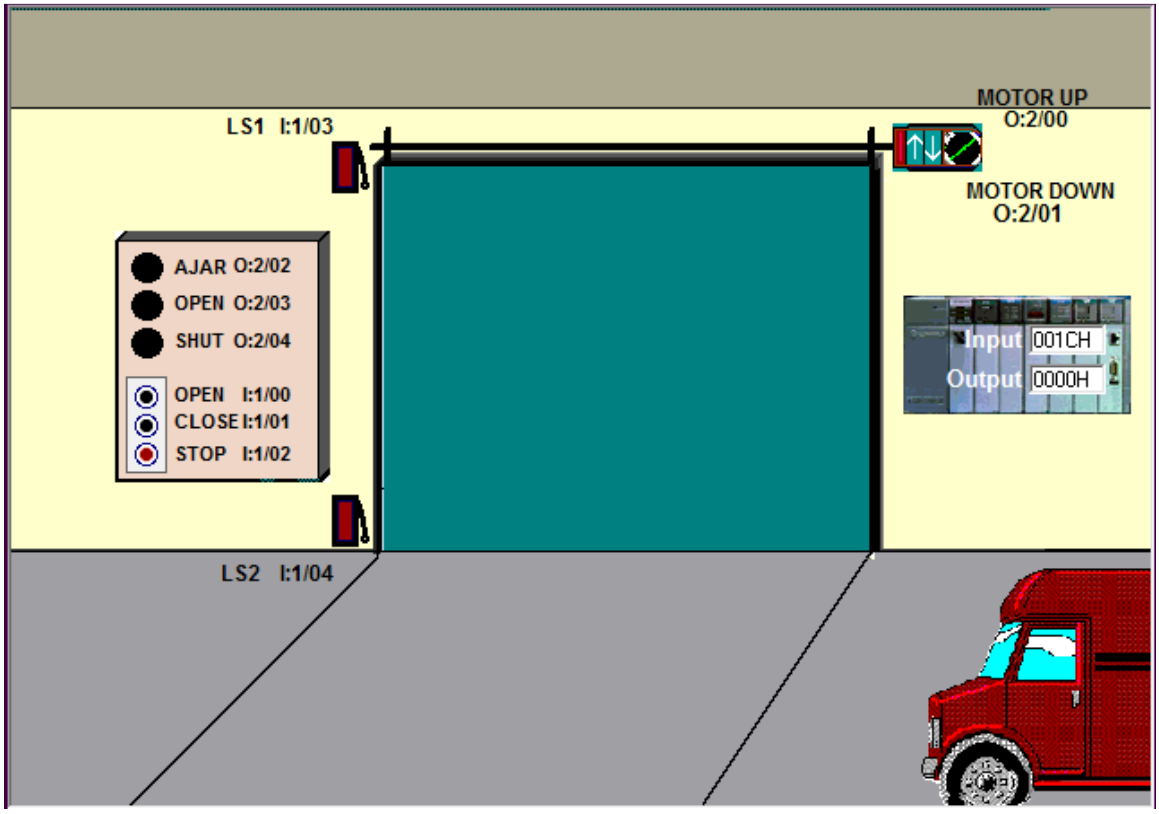
5.4.2 **Λειτουργία μιας αυτόματης πόρτας γκαράζ** (Door Simulation)

Η λογική της εφαρμογής έχει ως εξής: Τα κουμπιά Open και Close θα πιέζονται μόνο μια φορά (όχι παρατεταμένα) είναι εκείνα που θα ρυθμίζουν την κίνηση της πόρτας πάνω αν δεν έχει ανοίξει ολόκληρη και κάτω αν δεν έχει κλείσει ολόκληρη αντίστοιχα. Το κουμπί Stop, όποτε πιέζεται, θα σταματάει την λειτουργία της πόρτας και το σύστημα θα περιμένει να πατήσουμε ξανά ένα εκ των δύο αρχικών κουμπιών για να λειτουργήσει ξανά.

Όσο η πόρτα είναι έστω και ελάχιστα ανοιχτή η ένδειξη του LS2 θα είναι πράσινη και η LS1 κόκκινη, ενώ αν έχει ανοίξει όλη, τότε θα είναι πράσινες και οι δύο ενδείξεις (LS1 και LS2). Όσο η πόρτα είναι τελείως ανοιχτή, το κουμπί Open δεν θα προξενεί κάτι στο σύστημα. Αναλόγως και για το κουμπί Close. Πρέπει να αποφύγουμε την κατάσταση να

πατιούνται και τα δύο κουμπιά μαζί. Το Open lamp θα πρέπει να είναι ανοιχτό μόνο όταν η πόρτα είναι τελείως ανοιχτή. Αναλόγως για το Shut lamp. Όσο η πόρτα ανεβαίνει ή κατεβαίνει το Ajar lamp πρέπει να είναι ανοιχτό. Σε περίπτωση που φτάνει στο μέγιστο ύψος ή κλείσει τελείως η πόρτα, σβήνει.

Από το menu Simulations, επιλέξτε Door Simulation.

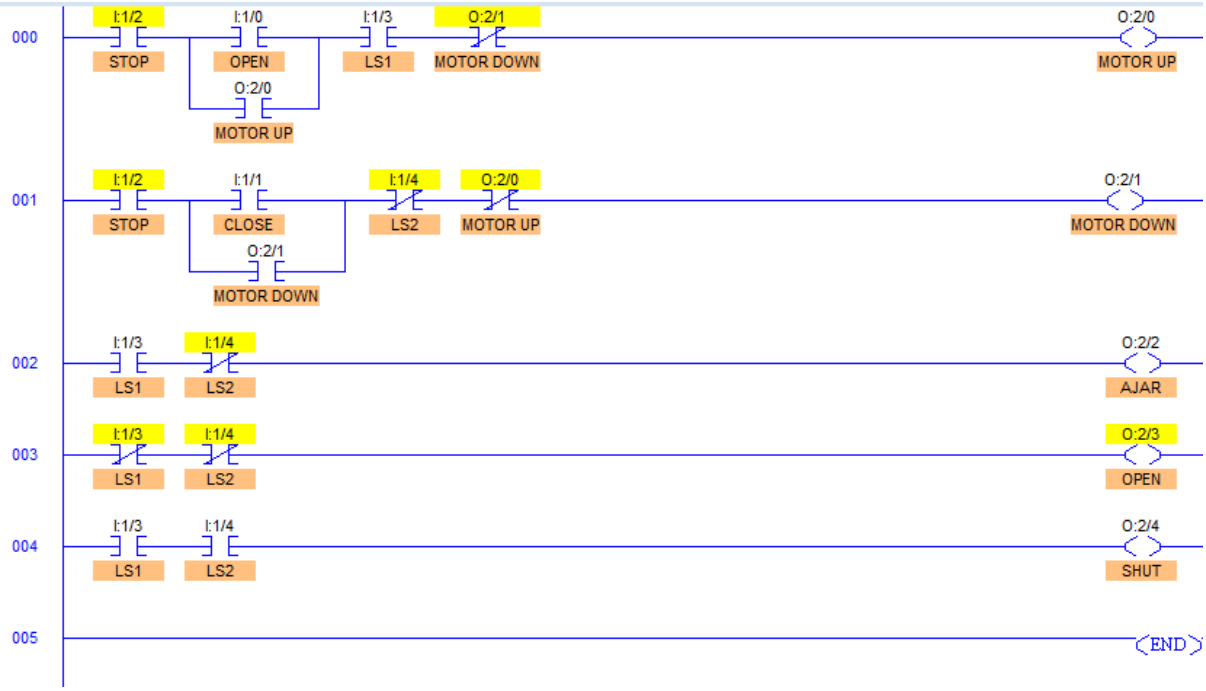


Υλοποιήστε το ακόλουθο διάγραμμα LAD. Εξηγήστε αναλυτικά τη λειτουργία του. Τρέξτε το πρόγραμμα και απεικονίστε τις λειτουργίες της πόρτας σε διάφορες φάσεις:

Τελείως κλειστή γκαραζόπορτα

Μισάνοιχτη γκαραζόπορτα

Τελείως ανοιχτή γκαραζόπορτα



ΑΠΑΝΤΗΣΗ

Κλάδος 000:

Εδώ εξετάζουμε υπό ποιες προϋποθέσεις θα ανοίξει η πόρτα (Motor Up). Αρχικά, θα πρέπει να μην είναι πατημένο το Stop button (το οποίο είναι Normally Closed-κοινή αρχή και για την άνοδο και για την κάθοδο της πόρτας) ΚΑΙ να είναι πατημένο το Open button (το οποίο είναι Normally Open) Ή να κατεβαίνει η πόρτα προς τα κάτω (εδώ έγκειται το ότι δεν έχουμε πατημένο παρατεταμένα το STOP button, αλλά το πατάμε μόνο μία φορά) ΚΑΙ το Open LS να είναι κλειστό (κόκκινο) ΚΑΙ να μην δουλεύει το Motor Down (να μην κατεβαίνει η πόρτα).

Κλάδος 001:

Εδώ εξετάζουμε υπό ποιες προϋποθέσεις θα κλείσει η πόρτα (Motor Down). Αρχικά, θα πρέπει να μην είναι πατημένο το Stop button (ομοίως με πριν) ΚΑΙ να είναι πατημένο το Close button (που είναι Normally Open) Ή η πόρτα να έχει ξεκινήσει να κλείνει (πάλι για το παρατεταμένο πάτημα του Stop) ΚΑΙ να είναι ανοιχτό το Close LS (πράσινο) ΚΑΙ να μην δουλεύει το Motor Up.

Κλάδος 002:

Εδώ εξετάζουμε υπό ποιες προϋποθέσεις θα ανοίξει το Ajar lamp το οποίο είναι το λαμπάκι που δείχνει αν γίνεται κάποια ενέργεια με την πόρτα (είτε ανοίγει είτε κλείνει

χωρίς να είναι τελείως ανοιχτή ή τελείως κλειστή). Για να ανάψει το λαμπάκι, αρκεί να δουλεύει το Open LS ΚΑΙ να μην δουλεύει το Close LS (δηλαδή να είναι κόκκινο το λαμπάκι του διακόπτη Open LS και πράσινο το λαμπάκι του διακόπτη Close LS).

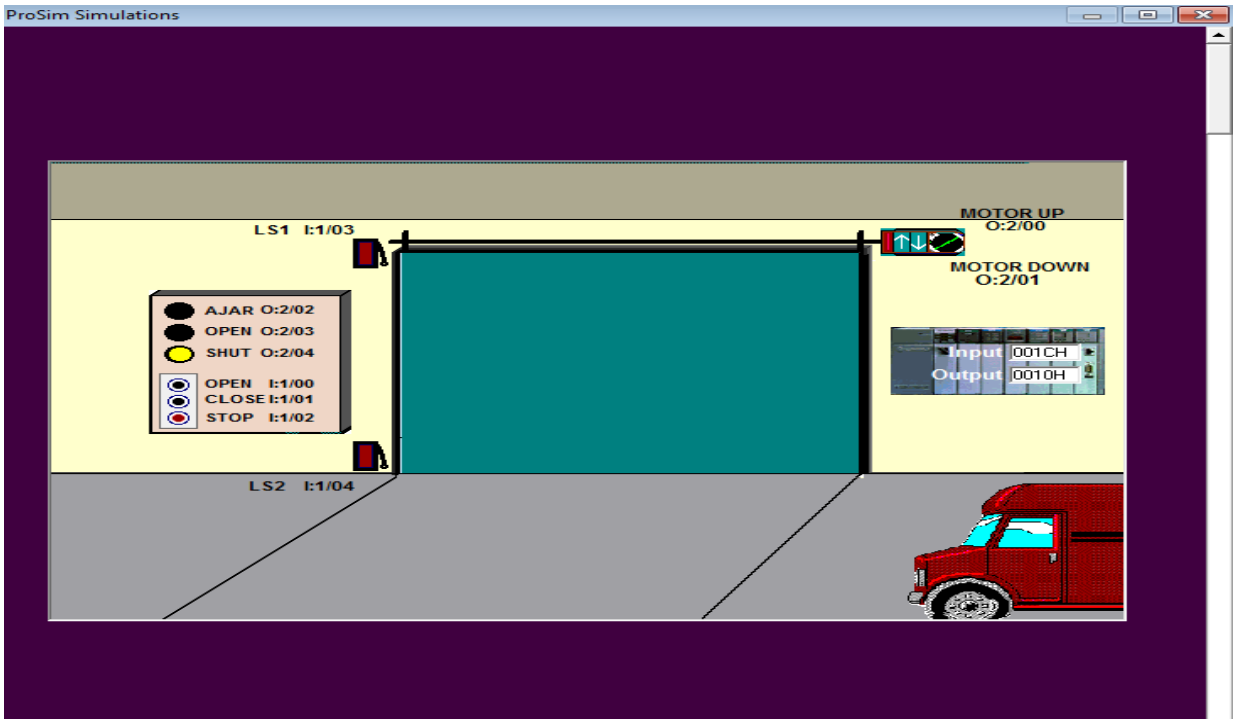
Κλάδος 003:

Εδώ εξετάζουμε υπό ποιες προϋποθέσεις θα ανοίξει το Open lamp. Για να γίνει αυτό, αρκεί να είναι δουλεύει το Open LS (δηλαδή το πάνω λαμπάκι του πάνω διακόπτη της γκαραζόπορτας να είναι πράσινο).

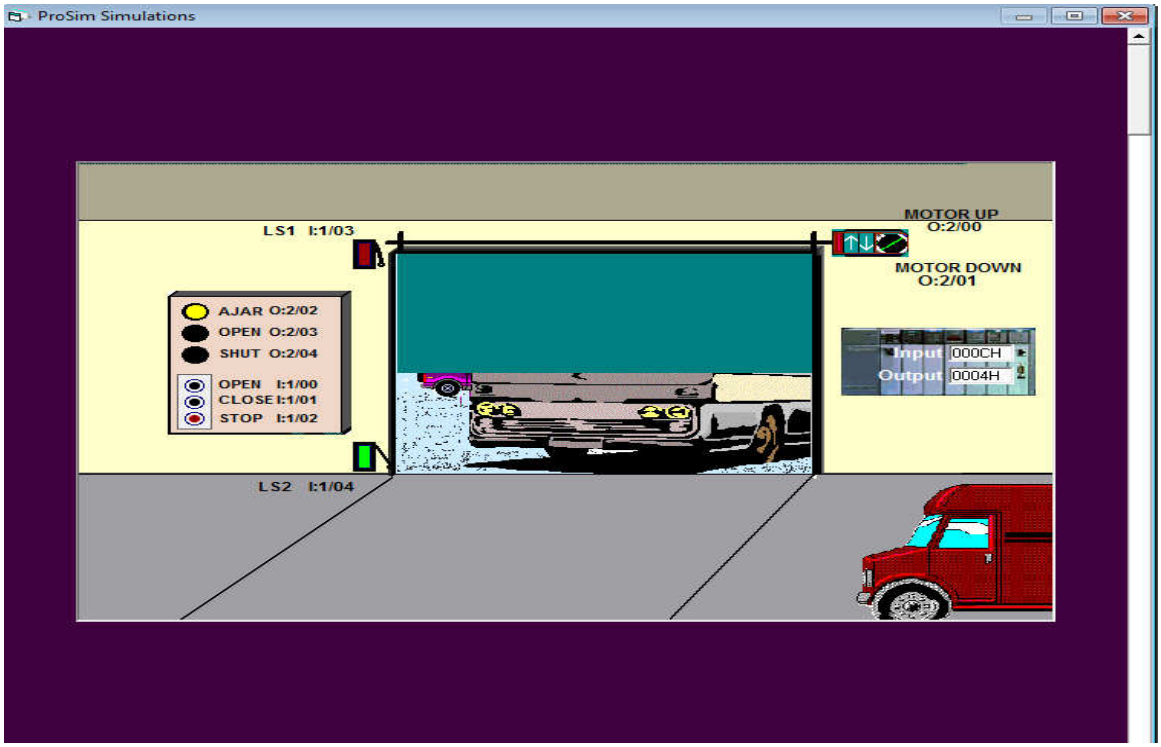
Κλάδος 004:

Εδώ εξετάζουμε υπό ποιες προϋποθέσεις θα ανοίξει το Shut lamp. Για να γίνει αυτό, αρκεί να δουλεύει το Close LS (δηλαδή το κάτω λαμπάκι του κάτω διακόπτη της γκαραζόπορτας να είναι κόκκινο).

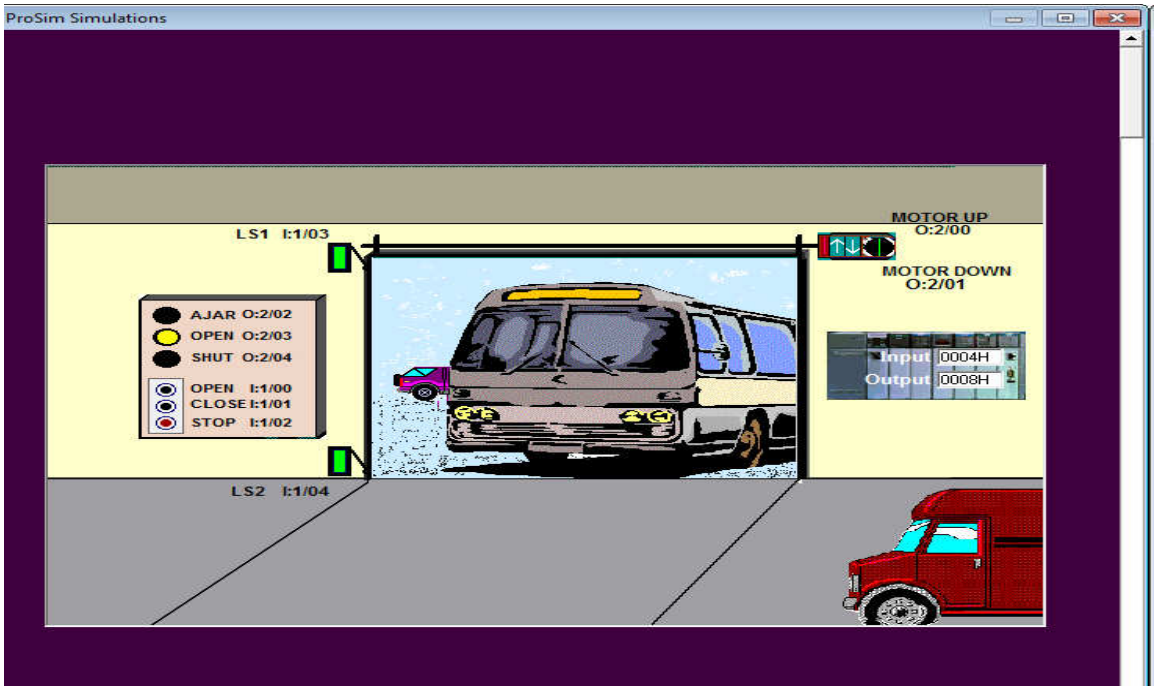
Τελείως κλειστή γκαραζόπορτα:



Μισάνοιχτη γκαραζόπορτα:



Τελείως ανοιχτή γκαραζόπορτα:

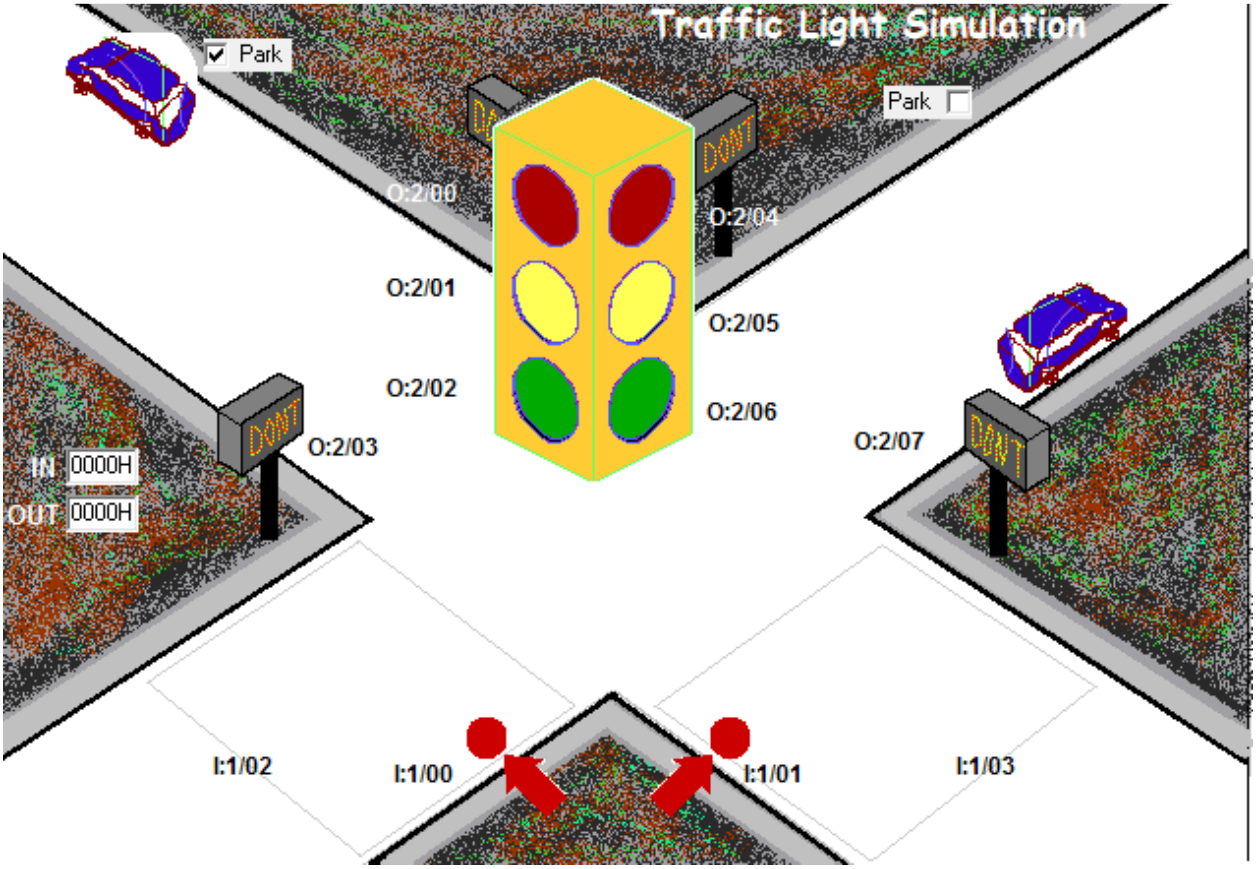


5.4.3 Φωτεινοί σηματοδότες

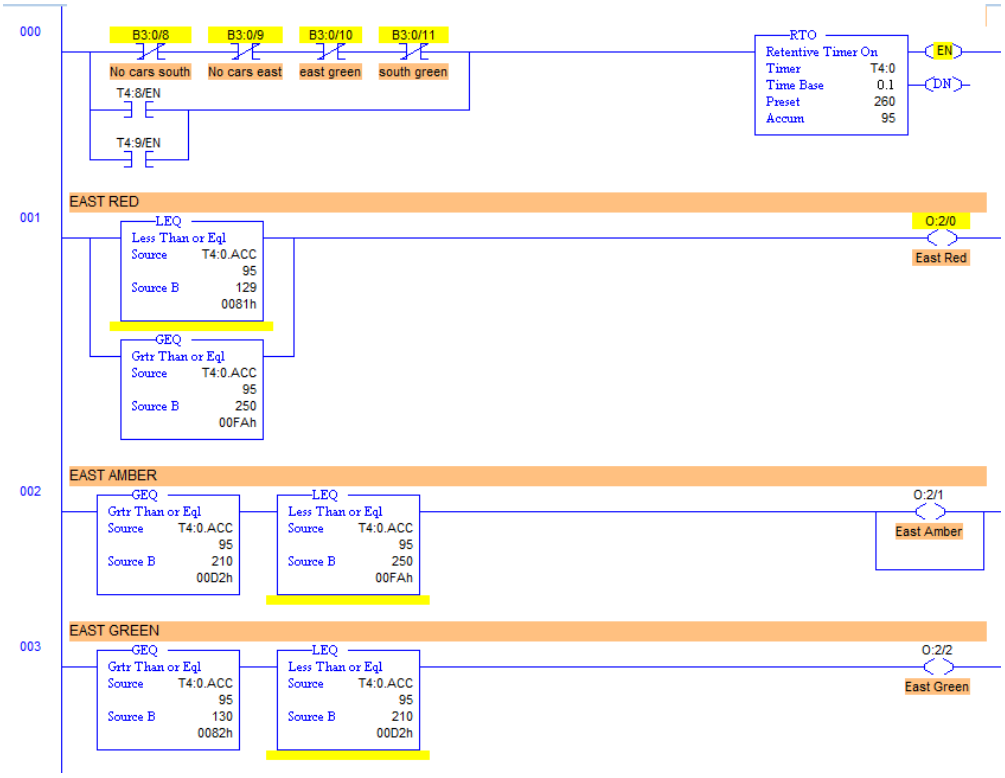
Ανοίγοντας το πρόγραμμα αναζητούμε από το menu την καρτέλα Simulations κ' έπειτα Traffic Simulator.

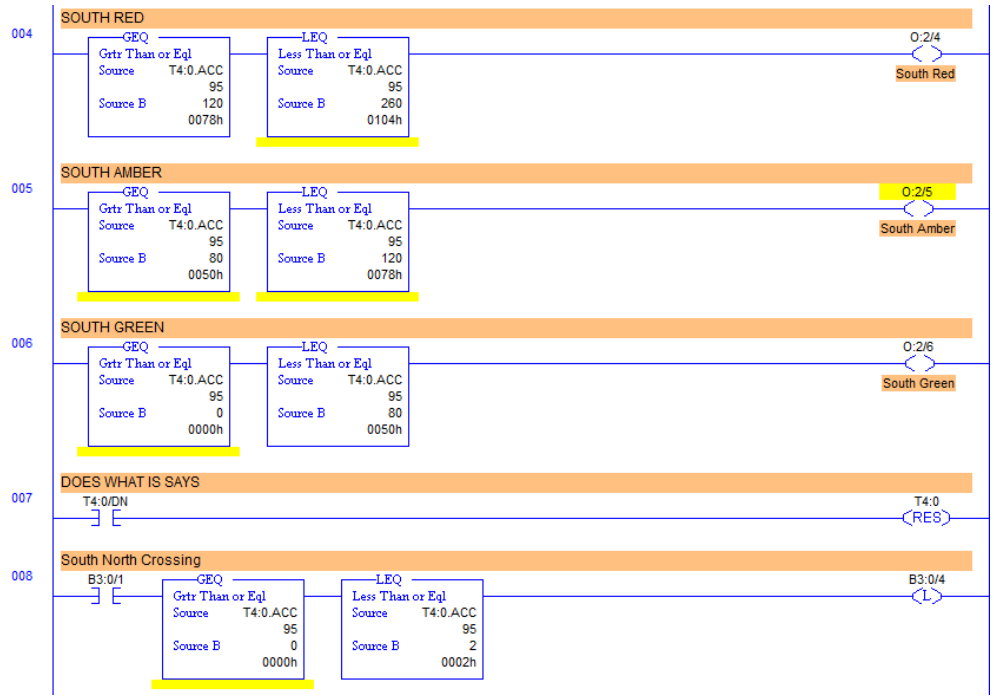
Είναι ένα αυτοματοποιημένο σύστημα που μπορεί να παρουσιαστεί με γλώσσα ladder. Η λογική του είναι σχετικά απλή, σε μερικά σημεία όμως θέλει προσοχή.

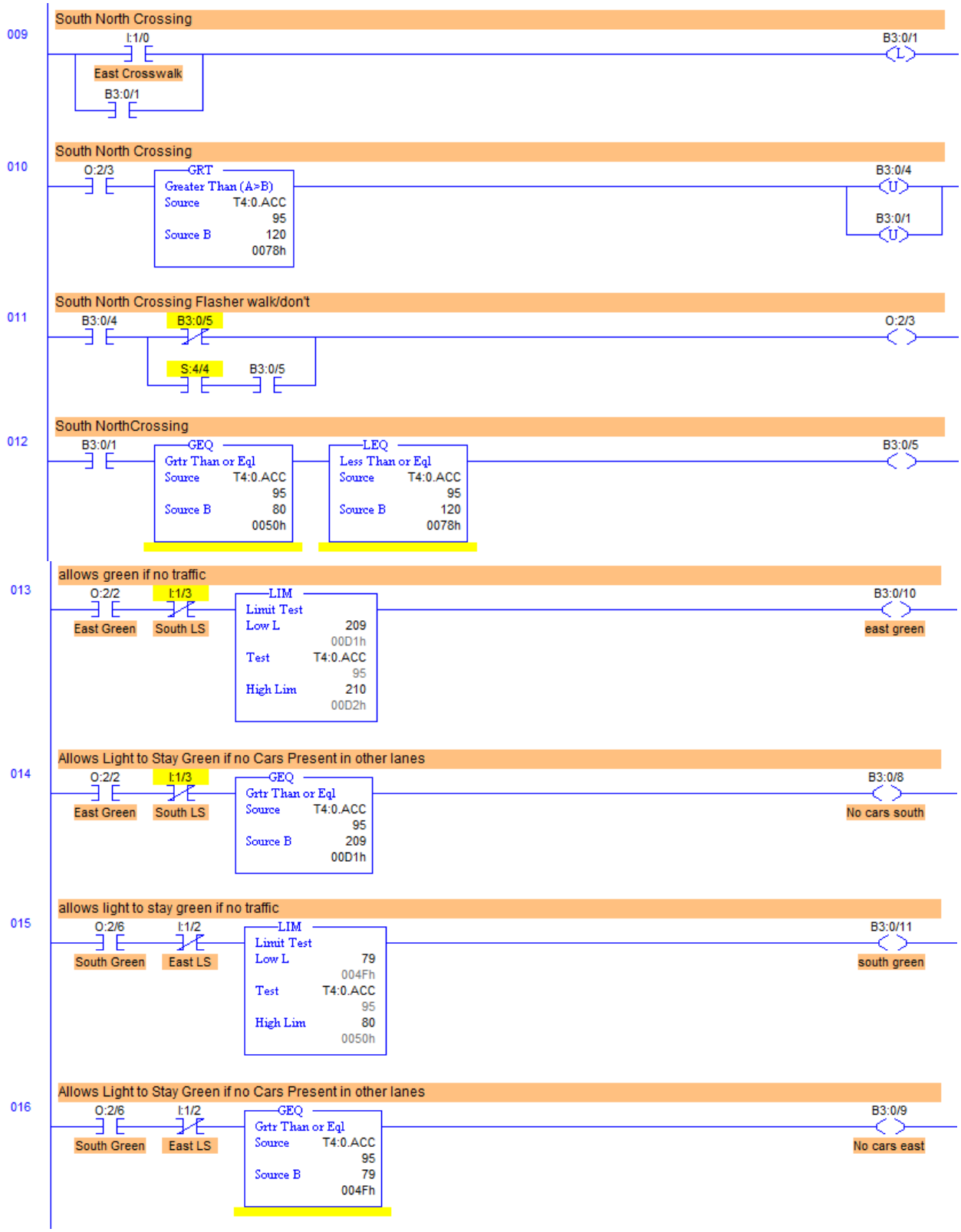
Η λογική της εργασίας έχει ως εξής: Θέλοντας να εμπλουτίσουμε τα περιεχόμενά μας, επιλέγουμε ένα σύστημα που λειτουργεί κατ' εξοχήν με counters: Έχουμε μια διασταύρωση στην οποία θέλουμε να προγραμματίσουμε τα δύο φανάρια να ρυθμίζουν την κυκλοφορία σε συγκεκριμένο χρονικό διάστημα (το οποίο το δίνουμε εμείς για οποιοδήποτε φανάρι).

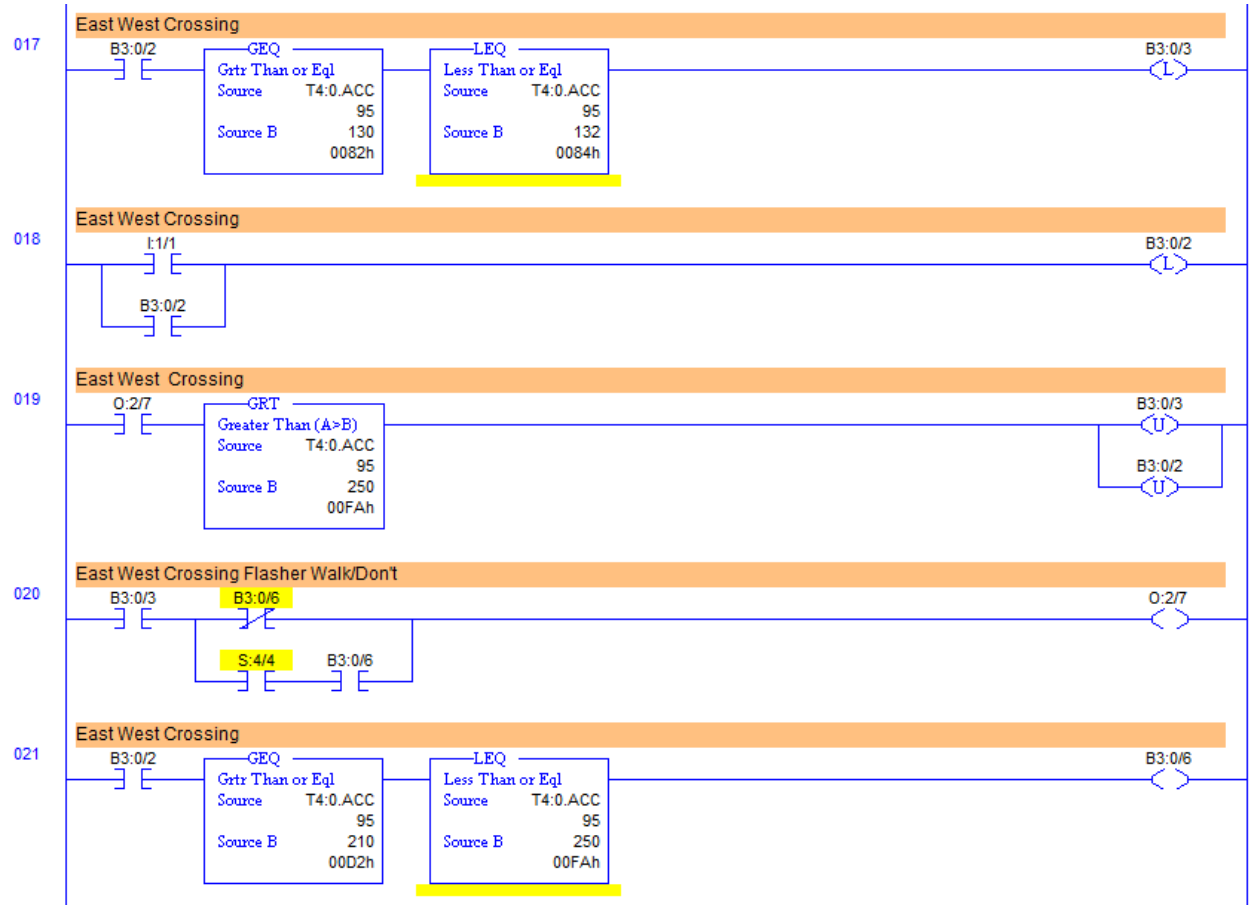


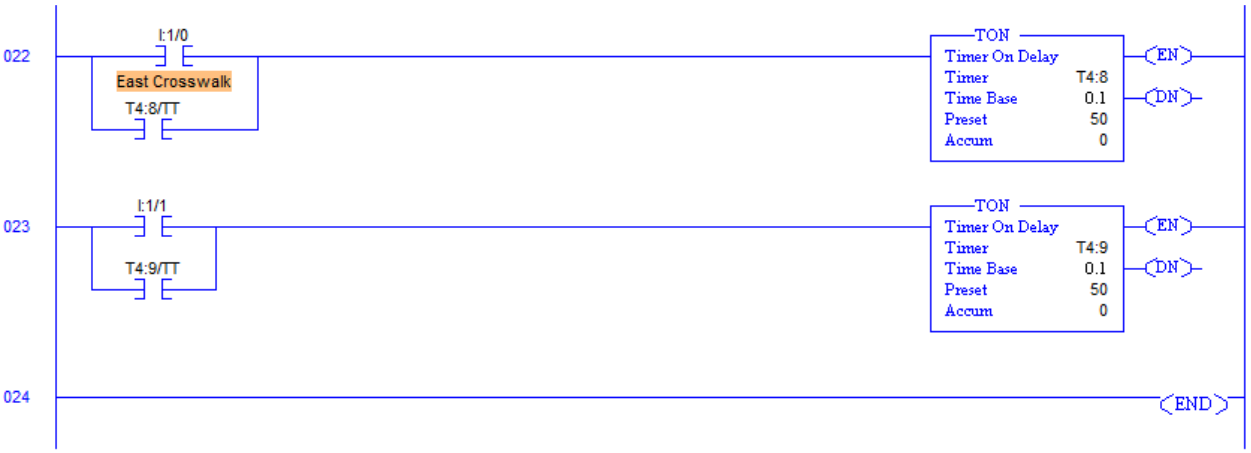
Μία ενδεικτική λύση του προβλήματος απεικονίζεται παρακάτω. Ζητείται να υλοποιήσετε το διάγραμμα LAD στον προσομοιωτή LOGIXPRO και αφού το τρέξετε να εξηγήσετε τη λειτουργία του.


000
B3:0/8
No cars south
B3:0/9
No cars east
B3:0/10
east green
B3:0/11
south green
T4:8/EN
T4:9/EN
RTO
Retentive Timer On
Timer T4:0
Time Base 0.1
Preset 260
Accum 95
EN
DN
001
EAST RED
LEQ
Less Than or Eql
Source T4:0.ACC 95
Source B 129 0081h
GEQ
Grtr Than or Eql
Source T4:0.ACC 95
Source B 250 00FAh
O:2/0
East Red
002
EAST AMBER
GEQ
Grtr Than or Eql
Source T4:0.ACC 95
Source B 210 00D2h
LEQ
Less Than or Eql
Source T4:0.ACC 95
Source B 250 00FAh
O:2/1
East Amber
003
EAST GREEN
GEQ
Grtr Than or Eql
Source T4:0.ACC 95
Source B 130 0082h
LEQ
Less Than or Eql
Source T4:0.ACC 95
Source B 210 00D2h
O:2/2
East Green


004
SOUTH RED
GEQ
Grtr Than or Eql
Source T4:0.ACC 95
Source B 120 0078h
LEQ
Less Than or Eql
Source T4:0.ACC 95
Source B 260 0104h
O:2/4
South Red
005
SOUTH AMBER
GEQ
Grtr Than or Eql
Source T4:0.ACC 95
Source B 80 0050h
LEQ
Less Than or Eql
Source T4:0.ACC 95
Source B 120 0078h
O:2/5
South Amber
006
SOUTH GREEN
GEQ
Grtr Than or Eql
Source T4:0.ACC 95
Source B 0 0000h
LEQ
Less Than or Eql
Source T4:0.ACC 95
Source B 80 0050h
O:2/6
South Green
007
DOES WHAT IS SAYS
T4:0/DN
T4:0
RES
008
South North Crossing
B3:0/1
GEQ
Grtr Than or Eql
Source T4:0.ACC 95
Source B 0 0000h
LEQ
Less Than or Eql
Source T4:0.ACC 95
Source B 2 0002h
B3:0/4
L


009
South North Crossing
I:1/0
East Crosswalk
B3:0/1
B3:0/1
010
South North Crossing
O:2/3
GRT
Greater Than (A>B)
Source T4:0.ACC
95
Source B 120
0078h
B3:0/4
B3:0/1
011
South North Crossing Flasher walk/don't
B3:0/4
B3:0/5
S:4/4
B3:0/5
O:2/3
012
South NorthCrossing
B3:0/1
GEQ
Grtr Than or Eql
Source T4:0.ACC
95
Source B 80
0050h
LEQ
Less Than or Eql
Source T4:0.ACC
95
Source B 120
0078h
B3:0/5
013
allows green if no traffic
O:2/2
East Green
I:1/3
South LS
LIM
Limit Test
Low L 209
00D1h
Test T4:0.ACC
95
High Lim 210
00D2h
B3:0/10
east green
014
Allows Light to Stay Green if no Cars Present in other lanes
O:2/2
East Green
I:1/3
South LS
GEQ
Grtr Than or Eql
Source T4:0.ACC
95
Source B 209
00D1h
B3:0/8
No cars south
015
allows light to stay green if no traffic
O:2/6
South Green
I:1/2
East LS
LIM
Limit Test
Low L 79
004Fh
Test T4:0.ACC
95
High Lim 80
0050h
B3:0/11
south green
016
Allows Light to Stay Green if no Cars Present in other lanes
O:2/6
South Green
I:1/2
East LS
GEQ
Grtr Than or Eql
Source T4:0.ACC
95
Source B 79
004Fh
B3:0/9
No cars east


017
East West Crossing
B3:0/2
GEQ
Grtr Than or Eql
Source T4:0.ACC
95
Source B 130
0082h
LEQ
Less Than or Eql
Source T4:0.ACC
95
Source B 132
0084h
B3:0/3
018
East West Crossing
I:1/1
B3:0/2
B3:0/2
019
East West Crossing
O:2/7
GRT
Greater Than (A>B)
Source T4:0.ACC
95
Source B 250
00FAh
B3:0/3
B3:0/2
020
East West Crossing Flasher Walk/Don't
B3:0/3
B3:0/6
S:4/4
B3:0/6
O:2/7
021
East West Crossing
B3:0/2
GEQ
Grtr Than or Eql
Source T4:0.ACC
95
Source B 210
00D2h
LEQ
Less Than or Eql
Source T4:0.ACC
95
Source B 250
00FAh
B3:0/6


022
I:1/0
East Crosswalk
T4:8/TT
TON
Timer On Delay
Timer T4:8
Time Base 0.1
Preset 50
Accum 0
EN
DN
023
I:1/1
T4:9/TT
TON
Timer On Delay
Timer T4:9
Time Base 0.1
Preset 50
Accum 0
EN
DN
024
END

Βιβλιογραφία

1. http://en.wikipedia.org/wiki/Programmable_logic_controller
2. http://eureka.lib.teithe.gr:8080/bitstream/handle/10184/1231/gkor_papa_main.pdf?sequence=3
3. http://thelearningpit.com/lp/doc/start.html
4. https://www.youtube.com/watch?v=cinbawOwzaQ
5. https://www.youtube.com/watch?v=9dqpmg7mz44
6. Beginner's Guide To PLC Programming by Neal Babcock.
7. Programmable Logic Controllers by William Bolton.
8. Simatic S7-1200 Document 04-2012.
9. Programmable Logic Controllers, Fifth Edition by W.Bolton.
10. Plc Programming for Industrial Automation by Kevin Collins.
11. Introduction to PLC controllers by Nebojsa Matic.
12. PLC Programming using RSLogix 500: Basic Concepts of Ladder Logic Programming by Gary D. Anderson
13. Introduction to PLCs: A beginner's guide to Programmable Logic Controllers by Elvin Pérez Adrover