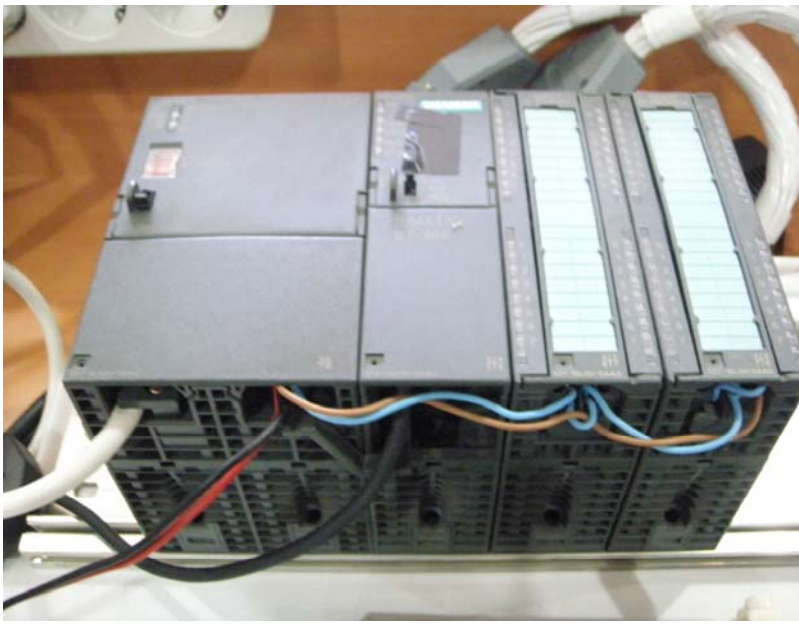
ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΣΕΡΡΩΝ
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΟΛΟΓΙΑΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΘΕΜΑ «Εκπόνηση προγράμματος PLC για την αυτοματοποίηση διάταξης ταινιομεταφορέα πνευματικού ρομποτικού βραχίονα για την παλετοποίηση αντικειμένων.»



ΚΑΨΙΩΧΑΣ ΠΑΝΑΓΙΩΤΗΣ
ΠΑΡΔΑΛΗΣ ΑΠΟΣΤΟΛΟΣ

ΕΙΣΗΓΗΤΗΣ: ΚΩΝΣΤΑΝΤΙΝΟΣ ΔΑΥΙΔ

ΣΕΡΡΕΣ 19-5-2011

ΠΕΡΙΕΧΟΜΕΝΑ

1. Εισαγωγή Πτυχιακής Εργασίας

Στην παρούσα πτυχιακή εργασία παρουσιάζεται τμήμα ενός αυτοματισμού μιας βιομηχανικής διεργασίας. Πιο συγκεκριμένα, αφορά τον έλεγχο και την εποπτεία ενός αυτοματοποιημένου συστήματος παλετοποίησης αντικειμένων μέσω πνευματικού ρομποτικού βραχίονα. Για τον έλεγχο του αυτοματισμού χρησιμοποιείται ένας προγραμματιζόμενος λογικός ελεγκτής (P.L.C) της σειράς SIMATIC S7 300 της Siemens, ενώ η εποπτεία θα γίνεται μέσω Ηλεκτρονικού Υπολογιστή.

Έχουμε έναν ταινιομεταφορέα, κινούμενο από ηλεκτροκινητήρα, πάνω στον οποίο τοποθετούνται τα διάφορα αντικείμενα. Όταν φτάσουν σε μία συγκεκριμένη θέση πάνω σ ’αυτόν όπου βρίσκεται ο ρομποτικός βραχίονας, σταματάει να κινείται παίρνοντας εντολή από φωτοκύτταρο και ξεκινάει με αυτόν τον τρόπο η διαδικασία της παλετοποίησης .

Στο πρώτο μέρος της γραπτής εργασίας, γίνεται μία εισαγωγή στα PLC όσων αφορά την εγκατάσταση, διαμόρφωση, δομή, τον προγραμματισμό και τη λειτουργία τους. Οι βασικές αρχές που αναφέρονται ισχύουν σε γενικές γραμμές για όλα τα PLC, αλλά οι αναφορές είναι στοχευμένες στη σειρά S7-300 της Siemens μια και αυτός είναι ο τύπος του PLC με το οποίο έχει υλοποιηθεί το πρακτικό μέρος της πτυχιακής εργασίας .

Στο δεύτερο μέρος, αναπτύσσονται τα μέρη της διάταξης (μηχανικά, ηλεκτρικά, πνευματικά, ηλεκτρονικά) με φωτογραφίες του καθένα ξεχωριστά και επεξήγηση της λειτουργίας τους, καθώς επίσης γίνεται η περιγραφή λειτουργίας της αυτόματης μονάδας με χρήση του PLC.

Στο τρίτο μέρος έχουμε την αυτοματοποίηση της διάταξης με το λογικό διάγραμμα αυτόματου ελέγχου.

Στο τέλος της πτυχιακής μας εργασίας υπάρχει εγχειρίδιο που αναλύει τον προγραμματισμό και τη λειτουργία του Simatic S7-300.

2.1 Εισαγωγή στους Προγραμματιζόμενους Λογικούς Ελεγκτές (PLC)

Εισαγωγή:

Στον χώρο του βιομηχανικού αυτοματισμού ο προγραμματιζόμενος λογικός ελεγκτής ο οποίος συμβολίζεται και σαν **P.L.C.** (Programmable Logic Controller) παρουσίασε με την εμφάνιση του, την δεκαετία του ΄70, μια σημαντική εξέλιξη έναντι των παραδοσιακών ηλεκτρομηχανικών και ηλεκτρονικών κυκλωμάτων. Η σημαντικότερη διαφορά είναι ότι στην περίπτωση των PLC τα κυκλώματα αυτοματισμού δεν πραγματοποιούνται με την λεγόμενη «Συρματωμένη λογική» αλλά με πρόγραμμα ή όπως αλλιώς λέγεται με την «προγραμματιζόμενη λογική».

Ορισμός

Ο Προγραμματιζόμενος Λογικός Ελεγκτής (PLC) είναι μια ειδική συσκευή, η οποία έρχεται να αντικαταστήσει στον πίνακα του κλασικού αυτοματισμού όλους τους βοηθητικούς ηλεκτρονόμους, τα χρονικά και τους απαριθμητές. Αντί για την κατασκευή ενός πίνακα με πολύπλοκες συνδεσμολογίες ,που έχουμε στον κλασικό αυτοματισμό, με την χρήση του (PLC) η λειτουργία του αυτοματισμού ΄΄ προγραμματίζεται ΄΄ μέσω μια ειδικής συσκευής (προγραμματιστή) ή μέσω ενός ηλεκτρονικού υπολογιστή με τη βοήθεια ειδικού λογισμικού, και είναι σε θέση:

I. Να δέχεται διάφορα ηλεκτρικά σήματα ρεύματος ή τάσεις (Inputs)

II. Να τα επεξεργάζεται και

III. Να παράγει τα κατάλληλα σήματα εξόδου(Outputs),τα οποία θα ενεργοποιήσουν τις υπό έλεγχο διατάξεις .

Τα στάδια εργασίας για το σχεδιασμό και την κατασκευή ενός αυτοματισμού στην προγραμματιζόμενη λογική είναι τα εξής:

1. Περιγραφή του αυτοματισμού

2.Ανάπτυξη του σχεδίου εφαρμογής του πίνακα(σχέδιο καλωδίωσης).

3.Κατασκευή του πίνακα της εγκατάστασης.

4.Ανάπτυξη του προγράμματος λειτουργίας του αυτοματισμού και εισαγωγή του προγράμματος στο PLC μέσω του προγραμματιστή.

5.Εγκατάσταση και σύνδεση στους ακροδέκτες (κλέμες) του πίνακα των αισθητήρων που δίνουν τις πληροφορίες (είσοδοι) και των συσκευών (αποδεκτών) που εκτελούν τις εργασίες (έξοδοι).

6.Δοκιμή λειτουργίας της εγκατάστασης.

7.Πλήρης λειτουργία του αυτοματισμού.

Ιστορική ανασκόπηση

Στις αρχές της δεκαετίας του ΄80 οι εταιρείες παραγωγής ηλεκτρολογικού υλικού εμφανίζουν στους μηχανικούς και τεχνικούς της βιομηχανίας ένα νέο προϊόν αυτοματισμού, το οποίο ονόμασαν PLC. Η πλήρης ονομασία αυτής της νέας συσκευής είναι Programmable Logic Controller (Προγραμματιζόμενος Λογικός Ελεγκτής). Οι εταιρίες δεν χρησιμοποίησαν αρχικά στην αγορά την πλήρη ονομασία, μιλώντας απλά για PLC, πράγμα που ίσως έγινε έντεχνα για να μην τρομάξουν το τεχνικό κατεστημένο της βιομηχανίας.

Το PLC εφευρέθηκε στην ανταπόκριση στις ανάγκες της αμερικανικής αυτοκινητοβιομηχανίας. Προγραμματιζόμενοι λογικοί ελεγκτές είχαν αρχικά εγκριθεί από την αυτοκινητοβιομηχανία, όπου το λογισμικό αντικατέστησε την εκ νέου καλωδίωση των συρματωμένων πινάκων ελέγχου, όταν τα μοντέλα παραγωγής άλλαξαν.

Πριν από το PLC, τον έλεγχο, την αλληλουχία, και την λογική ασφάλιση για τα αυτοκίνητα παραγωγής επιτεύχθηκε με εκατοντάδες ή χιλιάδες ρελέ , χρονοδιακόπτες cam , και ενορχήστρωσης και ειδικά κλειστού βρόχου ελεγκτές. Η διαδικασία για την πιστοποίηση αυτών των εγκαταστάσεων για το ετήσιο πρότυπο μετάβασης ήταν πολύ χρονοβόρα και δαπανηρή, όπως ηλεκτρολόγοι που απαιτούνται για την επανακαλωδίωση ατομικά κάθε ρελέ.

Το 1968 η GM Hydramatic (η αυτόματη κατανομή μετάδοση της General Motors) εξέδωσε ένα αίτημα για πρόταση για τη δημιουργία ηλεκτρονικού αντικαταστάτη για συρματωμένα συστήματα αναμετάδοσης. Η νικήτρια πρόταση προήλθε από τη Bedford Associates του Μπέντφορντ, της Μασαχουσέτης . Το πρώτο PLC, το οποίο ορίζεται το 084 επειδή ήταν ογδοηκοστό τέταρτο έργο της Bedford Associates », ήταν το αποτέλεσμα. Η Bedford Associates άρχισε μια νέα εταιρεία αφιερωμένη στην ανάπτυξη, κατασκευή, πώληση και συντήρηση αυτού του νέου προϊόντος: Modicon, το οποίο ανήλθε για τον αρθρωτό ψηφιακό ελεγκτή. Ένας από τους ανθρώπους που εργάστηκαν για το έργο ήταν **ο Dick Morley** , ο οποίος θεωρείται ο "πατέρας" του PLC. Η μάρκα Modicon πουλήθηκε το 1977 στην Gould Electronics , και αργότερα αποκτήθηκε από Γερμανική Εταιρεία AEG και στη συνέχεια από τη Γαλλική Schneider Electric (ο σημερινός ιδιοκτήτης). Ένα από τα πρώτα 084 μοντέλα σήμερα εκτίθενται στην έδρα Modicon στη Βόρεια Andover, Μασαχουσέτη.

Ο σκοπός PLCs στη ζωή :

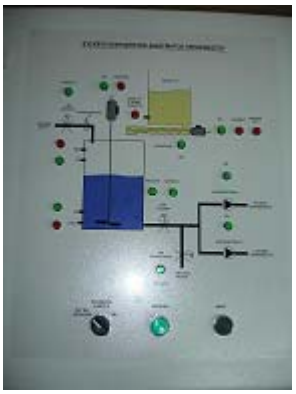
Το PLC κατά κύριο λόγο χρησιμοποιείται για τον έλεγχο μηχανών. Ένα πρόγραμμα είναι γραμμένο για το PLC που ενεργοποιεί και απενεργοποιεί εξόδους με βάση τις συνθήκες εισόδου και το εσωτερικό πρόγραμμα. Από την άποψη αυτή, είναι παρόμοιο με έναν υπολογιστή. Ωστόσο, ένα PLC μπορεί να έχει σχεδιαστεί για να προγραμματιστεί μία φορά, και να τρέξει κατ' επανάληψη, όπως απαιτείται. Στην πραγματικότητα, ένας προγραμματιστής μπορεί να χρησιμοποιήσει ένα PLC για τον έλεγχο όχι μόνο απλών

συσκευών, όπως μια πόρτα γκαράζ, αλλά και ολόκληρο το σπίτι, π.χ. τα φώτα να σβήνουν σε ορισμένες χρονικές περιόδους, η παρακολούθηση με προσαρμοσμένο ενσωματωμένο σύστημα ασφαλείας, κλπ. Πιο συχνά, ένα PLC βρίσκεται εντός της μηχανής σε ένα βιομηχανικό περιβάλλον. Ένα PLC μπορεί να τρέξει μια αυτόματη μηχανή για χρόνια με ελάχιστη ανθρώπινη παρέμβαση και τέλος έχει σχεδιαστεί να αντέχει στις πιο σκληρές συνθήκες.

Παραδείγματα κλασικού βιομηχανικού αυτοματισμού με χρήση PLC.

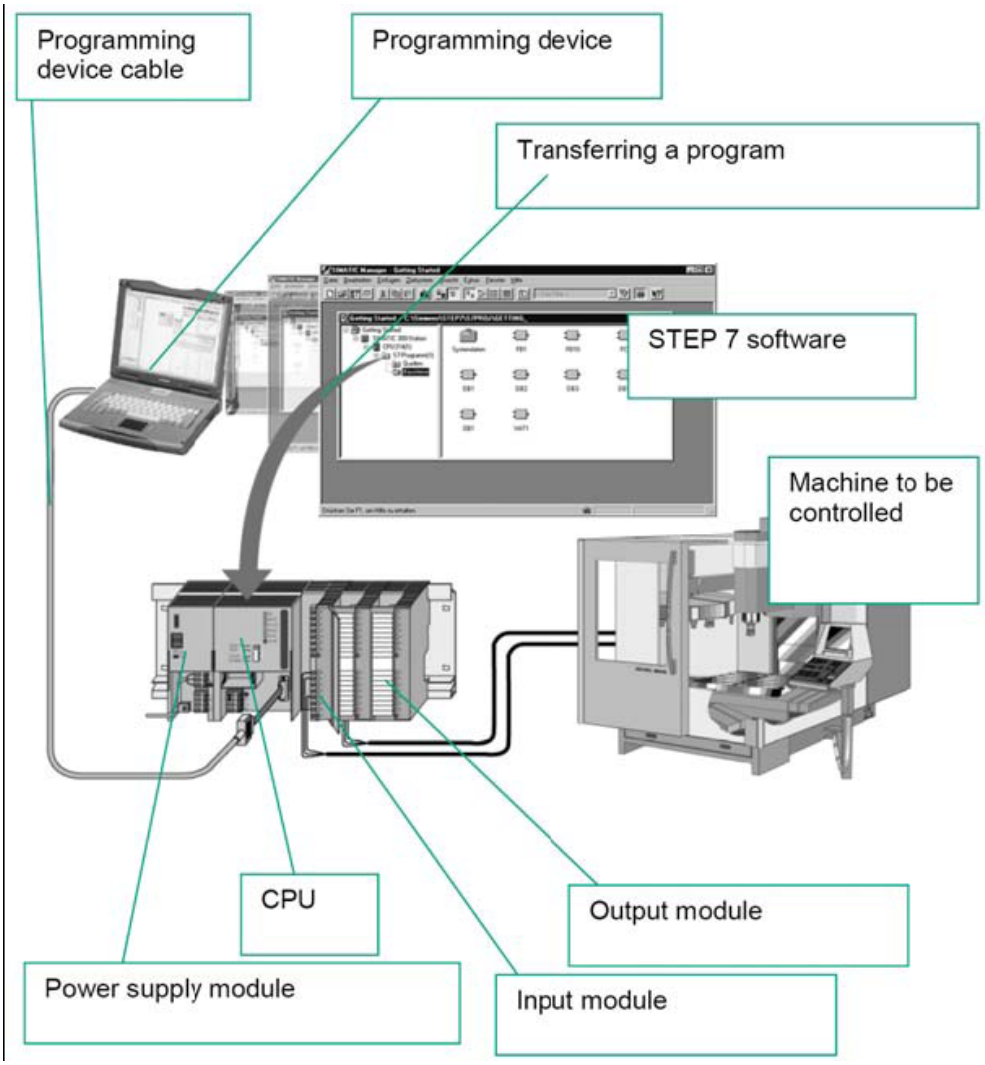
1) Ένα καλό παράδειγμα βιομηχανικού αυτοματισμού με PLC που μπορούμε να αναφερθούμε, είναι ένας αυτοματισμός **περιστροφικού κλιβάνου**. (Ο κλίβανος είναι ένας ειδικός θερμοθάλαμος που κατασκευάζεται με πυρίμαχα υλικά που θερμαίνονται με πολλούς τρόπους και χρησιμοποιούν την θερμότητα αυτή για διάφορους σκοπούς.) Η χρήση των PLC και η εφαρμογή της τεχνολογίας αυτών καθώς και των χειριστηρίων αφής (touch panel) βελτιστοποίησε τη λειτουργία του μηχανήματος και απλούστευσε τον χειρισμό του. Πάνω σ' αυτόν εμφανίζονται διάφοροι συναγερμοί που διαχειρίζονται με το PLC και μείωσε τους χρόνους στάσης, και είναι βέβαιο ότι θα οδηγήσει σε αύξηση του χρόνου ζωής του.

2) Επίσης σαν ένα άλλο παράδειγμα που μπορούμε να αναφερθούμε είναι μία **συσκευή παρασκευής διαλύματος υδρασβεστίου** που ελέγχεται από έναν ηλεκτρικό πίνακα. Με ένα σχέδιο στην πρόσοψη του πίνακα το οποίο θυμίζει SCADA, δίνεται η δυνατότητα στον χειριστή του μηχανήματος να ελέγχει την όλη διαδικασία με απλές κινήσεις και ταυτόχρονα να αναγνωρίζει οποιοδήποτε σφάλμα προκύπτει. Όλα αυτά μέσα από απλές φωτεινές ενδείξεις στην πρόσοψη του πίνακα οι οποίες ελέγχονται από ένα PLC το οποίο ευθύνεται και για την απροβλημάτιστη λειτουργία του μηχανήματος.



Υλικά Για Τον Έλεγχο Μιας Εγκατάστασης Μέσω P.L.C.

Στο επόμενο σχήμα παρουσιάζεται η δομή την οποία πρέπει να έχουμε σε μια εφαρμογή ελέγχου μέσω PLC. Αυτή αποτελείται:



- **ΠΡΟΓΡΑΜΜΑΤΙΣΤΡΙΑ**: Είναι το μέσο με το οποίο ο άνθρωπος επικοινωνεί με το PLC.

- **ΠΑΚΕΤΟ SOFTWARE**: Είναι το πρόγραμμα (γλώσσα) με το οποίο ο άνθρωπος επικοινωνεί με την προγραμματίστρια.

- **ΤΡΟΦΟΔΟΤΙΚΟ**: Ο ρόλος του είναι να δημιουργεί τις αναγκαίες τάσεις που χρειάζεται το PLC για την τροφοδοσία του.

- **CPU**: Είναι ο εγκέφαλος του συστήματος, εδώ περιέχονται και εκτελούνται τόσο το λειτουργικό πρόγραμμα του PLC όσο και το πρόγραμμα του χρήστη.

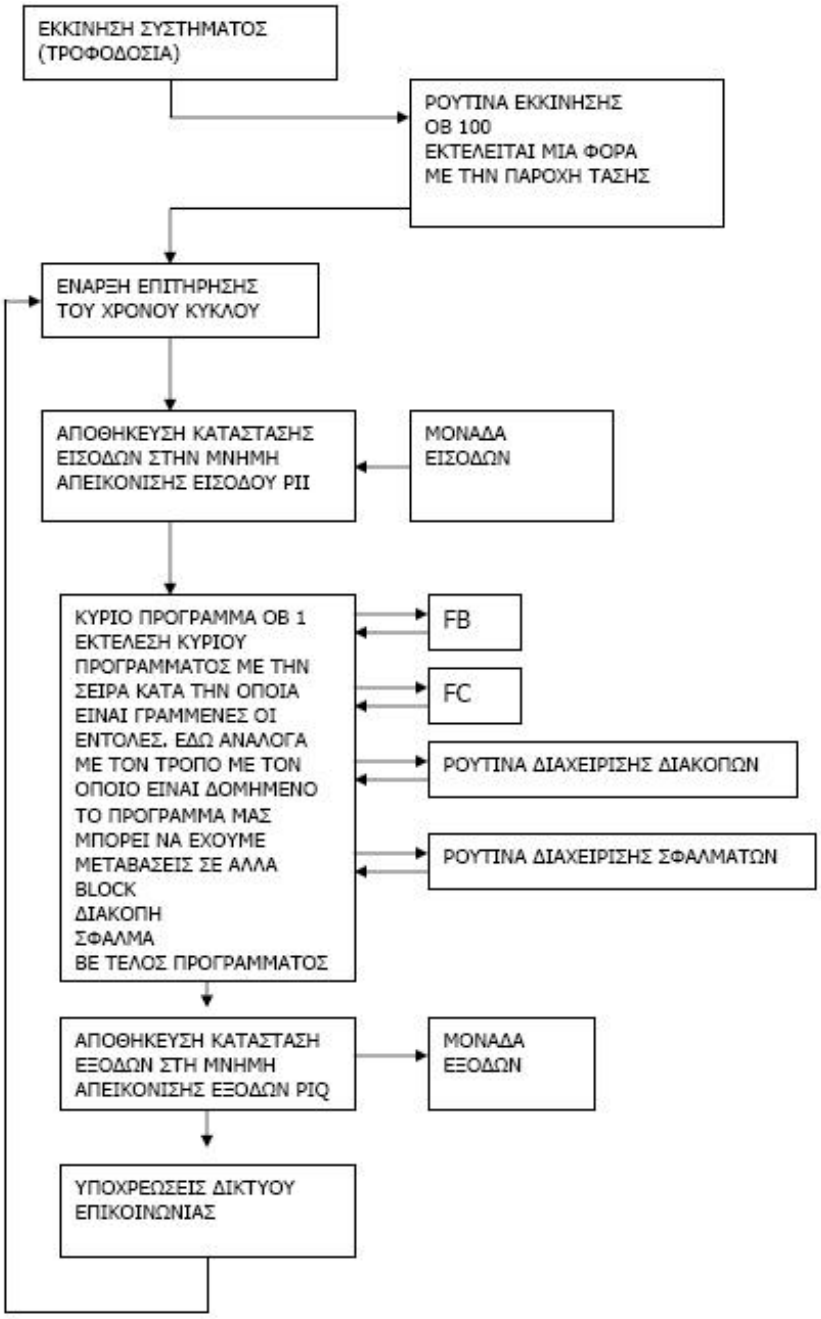
- **ΚΑΡΤΕΣ ΕΙΣΟΔΟΥ**: Είτε ψηφιακές, είτε αναλογικές, αυτές έχουν τον ρόλο να μετατρέπουν τα σήματα της εγκατάστασης σε σήματα τα οποία μπορεί να επεξεργαστεί η CPU.

- **ΚΑΡΤΕΣ ΕΞΟΔΟΥ**: Είτε ψηφιακές, είτε αναλογικές, αυτές έχουν τον ρόλο να μετατρέπουν τα σήματα που έχει ήδη επεξεργαστεί η CPU σε κατάλληλες τάσεις τις οποίες στέλνουμε προς την εγκατάσταση.

Σε εφαρμογές με χρήση των PLC η παρουσία της καλωδίωσης περιορίζεται μόνο στα περιφερειακά εξαρτήματα (αισθητήρια, διακόπτες , λυχνίες, …).

Κύκλος λειτουργίας της CPU

Ένα από τα πιο απλά αλλά πολύ σημαντικό σημείο που πρέπει να κατανοήσουμε είναι ο λεγόμενος κύκλος λειτουργίας μιας CPU. Ο κύκλος αυτός (scan cycle) παρουσιάζεται στο επόμενο σχήμα.



Παρατηρήσεις:

Α) Η πληροφορία για την κατάσταση της εισόδου αποκτάται μόνο στην αρχή του κύκλου και η κατάσταση της κατά τον χρόνο εκτέλεσης του προγράμματος θεωρείται σταθερή. Φυσικά για ιδιαίτερα κρίσιμες εισόδους υπάρχουν τεχνικές που επιτρέπουν την ακριαία πληροφόρηση και δράση της CPU.
Β) Η εκτέλεση μιας εντολής και η ενημέρωση της αντίστοιχης θέσης μνήμης γίνεται με την σειρά με την οποία είναι γραμμένη η εντολή στο πρόγραμμα.

ΠΛΕΟΝΕΚΤΗΜΑΤΑ PLC ΣΥΓΚΡΙΤΙΚΑ ΜΕ ΤΟΝ ΚΛΑΣΙΚΟ ΑΥΤΟΜΑΤΙΣΜΟ

- Είναι συσκευές γενικές χρήσης (δεν είναι κατασκευασμένα για ένα συγκεκριμένο είδος εφαρμογής).
- Δεν ενδιαφέρει ο συνολικός αριθμός των επαφών, χρονικών, απαριθμητών (δεν είναι φυσικά στοιχεία, αλλά στοιχεία μνήμης).
- Η λειτουργία του αυτοματισμού μπορεί να αλλάξει σε οποιοδήποτε στάδιο θελήσουμε.
- Εύκολος οπτικός έλεγχος της λειτουργίας ή μη στοιχείων της εγκατάστασης με την βοήθεια των LED που υπάρχουν σε όλες τις κάρτες.
- Με την βοήθεια της προγραμματίστριας μπορούμε να παρακολουθήσουμε την ροή της εκτέλεσης του προγράμματος και μέσω διαγνωστικών να εντοπίσουμε τυχόν βλάβες.
- Κάθε αλλαγή στο πρόγραμμα του χρήστη αποθηκεύεται στην μνήμη του PLC, έτσι ο τεχνικός δεν βρίσκεται προ απρόοπτου να διαβάζει ένα σχέδιο και άλλο να βρίσκεται πραγματικά στην εγκατάσταση.
- Τα PLC καταλαμβάνουν πολύ μικρό χώρο απ' ότι ένα αντίστοιχος πίνακας αυτοματισμού.
- Μπορούν να τοποθετηθούν και μέσα σε πεδίο ισχύος χωρίς πρόβλημα εφ' όσον τηρήσουμε τις οδηγίες του κατασκευαστή.
- Το PLC κρατά πάντοτε στη μνήμη του το τελευταίο πρόγραμμα το οποίο μπορεί να διαβαστεί από μια προγραμματίστρια ή να εκτυπωθεί σε ένα χαρτί από κάποιον εκτυπωτή.
- Έχουμε την δυνατότητα να συνδέσουμε επάνω τους οθόνες, εκτυπωτές, πληκτρολόγια και HMI συστήματα.
- Οι γλώσσες προγραμματισμού καλύπτουν όλο το φάσμα των ανθρώπων που καλούνται να ασχοληθούν με την τεχνολογία αυτή.
- Είναι επεκτάσιμα, αξιόπιστα, έχουν μεγάλη ταχύτητα και πολλή μεγάλη διάρκεια ζωής.
- Έχουν μεγάλες δυνατότητες δικτύωσης με πρότυπα βιομηχανικά δίκτυα.
- Μας δίνουν δυνατότητα αντιγραφής εφαρμογών.
- Απαιτούν ελάχιστη συντήρηση.
- Το κόστος κατασκευής ενός PLC είναι σημαντικά μικρότερο από το κόστος παραγωγής ενός μεγάλου αριθμού βοηθητικών ηλεκτρονόμων, χρονικών και απαριθμητές.
- Ο χρόνος κατασκευής του αυτοματισμού είναι μηδαμινός σε σχέση με την κατασκευή ενός κλασικού πίνακα αυτοματισμού.

ΜΕΙΟΝΕΚΤΗΜΑ: Θα μπορούσε να θεωρηθεί η έλλειψη επαρκούς ενημέρωσης των τεχνικών όλων των βαθμίδων, ειδικά στην Ελλάδα, πράγμα το οποίο δυσκολεύει και δημιουργεί προβλήματα στην εφαρμογή των PLC.

Συμπερασματικά μπορούμε να πούμε ότι σήμερα η λύση των προβλημάτων αυτοματισμού δίνεται από τις διάφορες προγραμματιζόμενες συσκευές. Με τη λέξη “διάφορες” εννοούμε ότι υπάρχει μια τεράστια ποικιλία PLCs τα οποία διαφέρουν μεταξύ τους ανάλογα με την κατασκευάστρια εταιρία, το μέγεθος, τις δυνατότητες και το κόστος τους.

Η συγκεκριμένη πτυχιακή εργασία υλοποιείται με PLC της σειράς ***S7 – 300***, ελεγκτές της Siemens που προορίζονται για μεσαίας κλίμακας βιομηχανικές εφαρμογές.

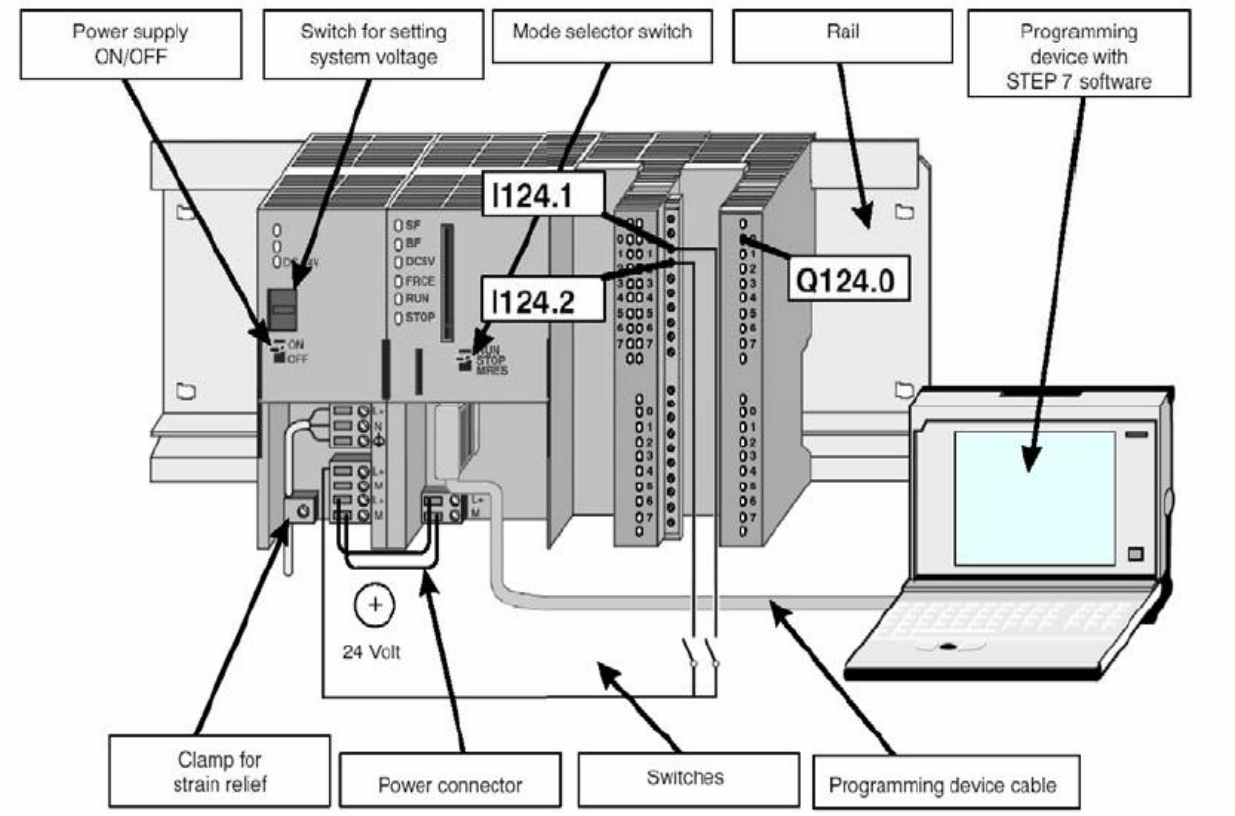
Τα κυριότερα χαρακτηριστικά τους είναι:

- Μνήμη προγράμματος μέχρι 85K.
- Μέχρι 1024 εισόδους και εξόδους.
- Ενσωματωμένη πολυκομβική διασύνδεση (MPI) για δημιουργία μικρών δικτύων και για σύνδεση με την προγραμματίστρια μονάδα.
- Μεγάλη ταχύτητα. Μία CPU μπορεί να εκτελέσει 1024 δυαδικές πράξεις σε 0.1-0.3 sec.
- MODULAR μορφή.
- Δυνατότητα επέκτασης έως και 32 κάρτες.
- Ενσωματωμένες ειδικές λειτουργίες: counters, positioners, έλεγχος κλειστού βρόχου με τις CPU 3xx IFM.
- Ενσωματωμένη διασύνδεση PROFIBUS-DP στη σειρά S7-300 2DP. Χρήση της CPU ως master ή slave.
- Ενσωματωμένες λειτουργίες για HMI.
- Εύκολη και γρήγορη διαμόρφωση και προγραμματισμός μέσω λογισμικού STEP7.
- Εκτεταμένες διαγνωστικές λειτουργίες μέσω του STEP7. Μηνύματα σφαλμάτων που αποθηκεύονται στον διαγνωστικό buffer με αναγραφή ημερομηνίας και ώρας.
- Μεγάλη ποικιλία από CPU για καλύτερη επιλογή αναλόγως εφαρμογής.
- Μεγάλες δυνατότητες δικτύωσης (MPi, PROFIBUS, Industrial Ethernet).
- Μία μόνο κάρτα για όλους τους τύπους αναλογικών σημάτων.
- 32-bit σετ εντολών για μαθηματικές συναρτήσεις.
- Ελεύθερη διευθυνσιοδότηση των καρτών.

2.2 Η ΔΟΜΗ ΤΩΝ PLC

ΒΑΣΙΚΗ ΔΟΜΗ ΤΩΝ PLC

Κάθε PLC μπορεί να δομηθεί από επιμέρους μονάδες ανάλογα με την εφαρμογή για την οποία θα χρησιμοποιηθεί. Στο παρακάτω σχήμα φαίνονται τα βασικά στοιχεία μιας απλής εφαρμογής.

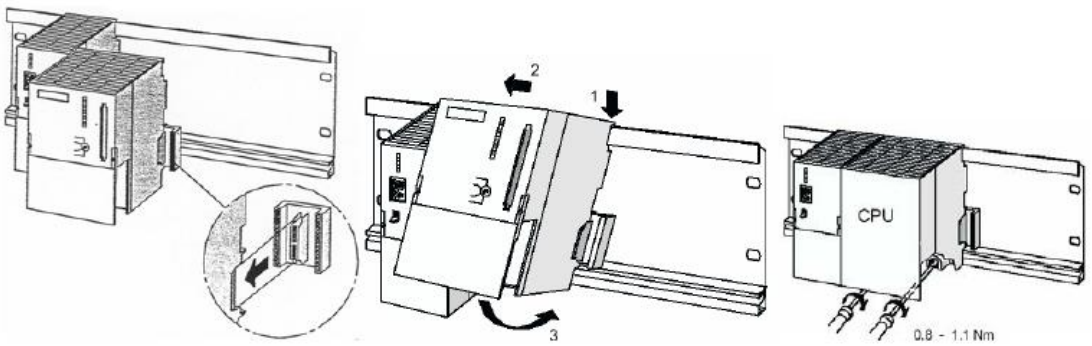


Τα σημαντικότερα στοιχεία μιας εφαρμογής με PLC της σειράς S7-300 δίνονται παρακάτω.

ΠΛΑΙΣΙΟ ΣΤΗΡΙΞΗΣ (RACK)

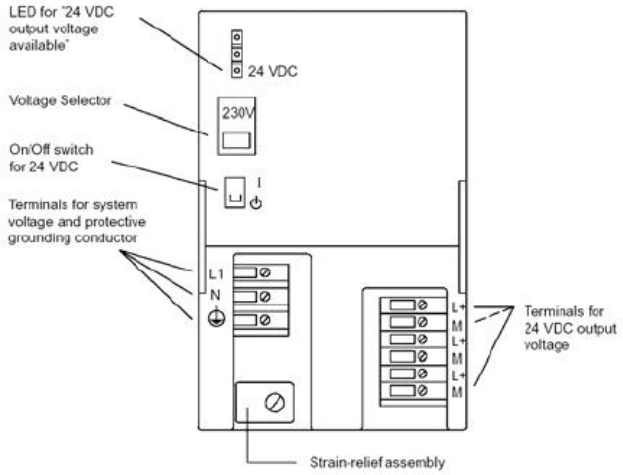
Ο ρόλος του είναι απλά να στηρίζει τις διάφορες κάρτες που θα συνθέσουν το σύστημα αυτοματισμού. Σ' ένα σύστημα με υλικό της σειράς S7 – 300 μπορούμε συνολικά να έχουμε έως τέσσερα πλαίσια στήριξης (rack).

Στήριξη Καρτών στο Rack:



ΤΡΟΦΟΔΟΤΙΚΟ PS (Power Supply)

Ο ρόλος του είναι να μετατρέπει την τάση του δικτύου τροφοδοσίας στην κατάλληλη τάση λειτουργίας του PLC.
Στην κατασκευή μας, χρησιμοποιούμε ένα τροφοδοτικό PS 307: 5A. Μορφολογικά αυτό παρουσιάζεται στην επόμενη εικόνα

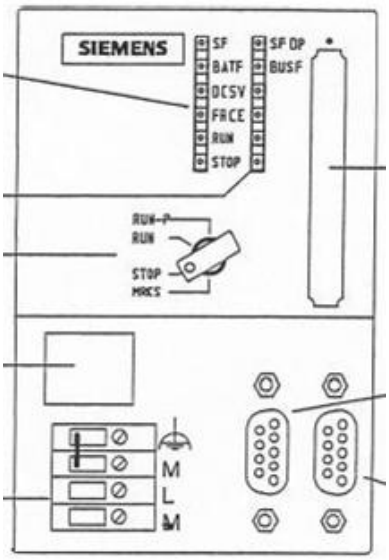


Διαθέτει:

- Κλέμες για τάση τροφοδοσίας (L1, N) και γείωση.
- Κλέμες για τάση εξόδου 24 V (L+, M)
- Διακόπτης ON – OFF
- Επιλογικό διακόπτη τάσης τροφοδοσίας (230 VAC ή 120 VAC)
- Ενδεικτικά LED ύπαρξης τάσεως εξόδου 24 VDC.

ΚΕΝΤΡΙΚΗ ΜΟΝΑΔΑ ΕΠΕΞΕΡΓΑΣΙΑΣ - CPU

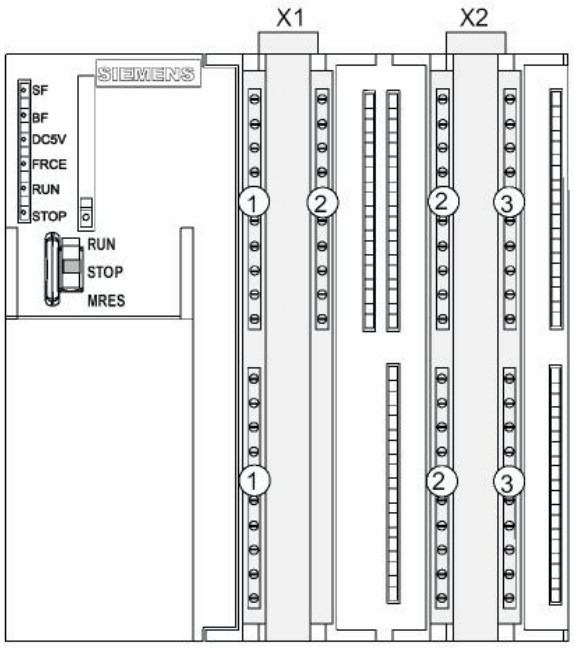
Εκτελεί λειτουργικό πρόγραμμα του χρήστη. Ελέγχει τις επικοινωνίες σε ένα MPI δίκτυο.



ΚΑΡΤΕΣ ΕΙΣΟΔΩΝ ΚΑΙ ΕΞΟΔΩΝ

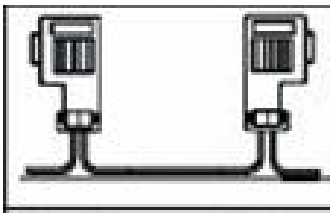
Ψηφιακές –Αναλογικές (digital-analog SM)

Προσαρμόζουν τα ηλεκτρικά σήματα από το εξωτερικό περιβάλλον προς την CPU και αντιστρόφως.



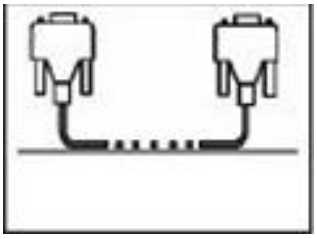
ΚΑΛΩΔΙΟ Profibus ΔΙΚΤΥΟΥ ΜΕ ΤΟΥΣ bus connector

Συνδέει μεταξύ τους κόμβους ενός MPI ή Profibus δικτύου.



ΚΑΛΩΔΙΟ ΣΥΝΔΕΣΗΣ ΠΡΟΓΡΑΜΜΑΤΙΣΤΗ (PG cable)

Συνδέει τη CPU με την συσκευή προγραμματισμού PG (μπορεί ως προγραμματιστής να χρησιμοποιηθεί ένας Η/Υ με adaptor cable).



2.3 ΔΟΜΗ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

ΔΟΜΗ PROJECT

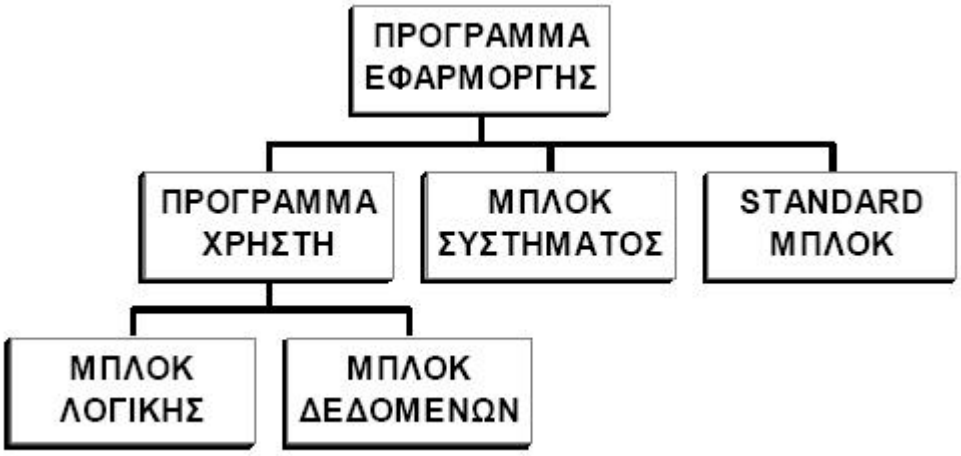
Κατά τη φάση του σχεδιασμού του project μας ένα από τα πρώτα πράγματα που πρέπει να κάνουμε είναι στο να αποφασίσουμε με ποιόν τρόπο θα δομήσουμε το πρόγραμμα μας δηλαδή στο τι μπλοκ θα περιέχει και πως θα συνδέονται μεταξύ τους αυτά τα μπλοκ. Ας δούμε όμως πρώτα πως είναι οργανωμένο ένα πρόγραμμα στην CPU. Κάθε CPU περιλαμβάνει δύο προγράμματα ανεξάρτητα το ένα από το άλλο:

α) ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ

Το λειτουργικό σύστημα είναι το σύνολο των ορισμών και εντολών που ελέγχουν τους πόρους του συστήματος. Είναι αυτό που ενημερώνει το ρολόι του πραγματικού χρόνου στη CPU, που ελέγχει την κατάσταση του διακόπτη της CPU, (RUN, STOP, …), ελέγχει να ανάψει τα LED στη CPU, να ρυθμίσει τις επικοινωνίες μέσα απ το MPI interface, … Στο λειτουργικό σύστημα δεν μπορούμε να κάνουμε μεταβολές, μπορούμε όμως να διαβάσουμε ή να χρησιμοποιήσουμε ορισμένα αποτελέσματα αυτού (π.χ. το ρολόι πραγματικού χρόνου).

β) ΠΡΟΓΡΑΜΜΑ ΕΦΑΡΜΟΓΗΣ

Το πρόγραμμα εφαρμογής είναι το σύνολο των εντολών και ορισμών που χρειάζεται το PLC για τον έλεγχο της εγκατάστασης. Η δομή ενός προγράμματος εφαρμογής δίνεται στην κάτω εικόνα.



ΤΥΠΟΙ ΤΩΝ ΔΙΑΘΕΣΙΜΩΝ ΜΠΛΟΚ

Για το χτίσιμο της εφαρμογής μας έχουμε στην διάθεση μας διαφορετικά είδη μπλοκ προγραμματισμού. Το τι θα χρησιμοποιήσουμε και πως θα τα διασύνδεουμε είναι τις περισσότερες φορές υποκειμενική υπόθεση και εξαρτάται από την εφαρμογή που έχουμε να προγραμματίσουμε.

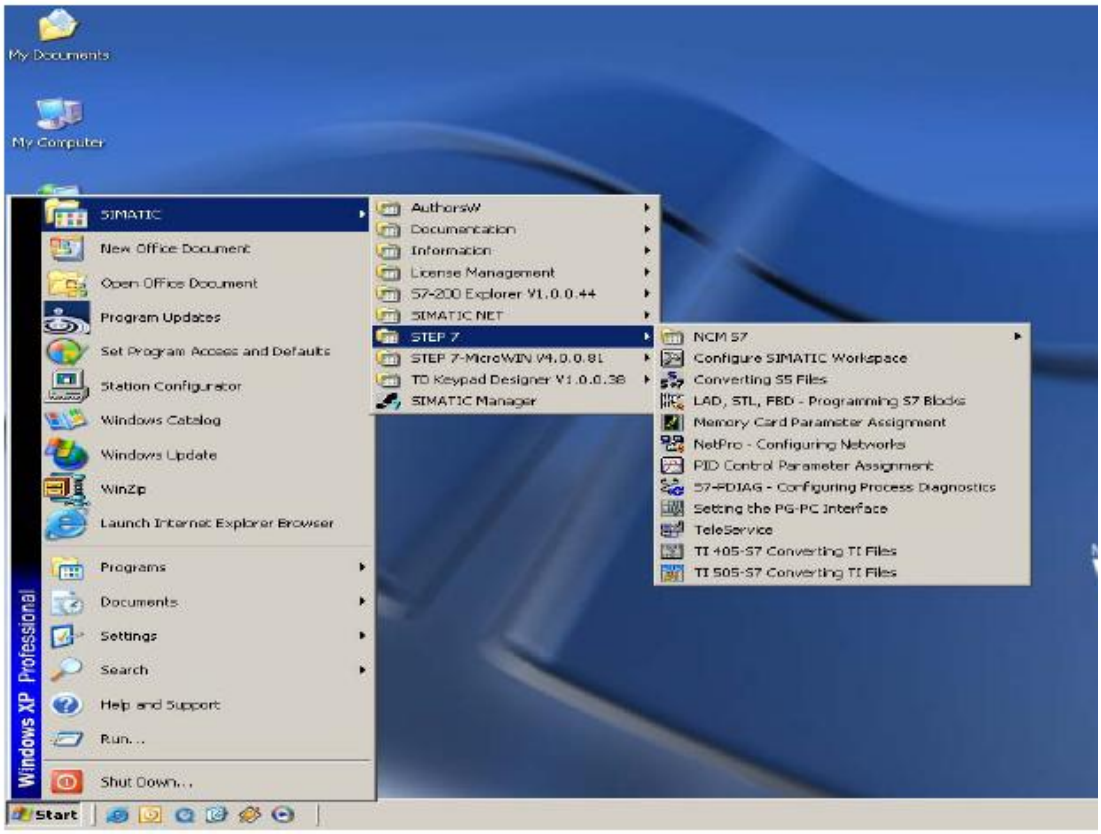
Οι διάφοροι τύποι των διαθέσιμων μπλοκ είναι:

- **ΜΠΛΟΚ ΟΡΓΑΝΩΣΗΣ OB (Organization Blocks).**
- **ΣΥΝΑΡΤΗΣΕΙΣ FC (Functions)**
- **ΜΠΛΟΚ ΣΥΝΑΡΤΗΣΕΩΝ FB (Function Block)**
- **ΜΠΛΟΚ ΔΕΔΟΜΕΝΩΝ DB (Data Blocks)**

2.4 SIMATIC MANAGER

Ο Simatic Manager είναι το κύριο εργαλείο στο Step 7 αφού είναι αυτός που διαχειρίζεται τα project. Το χαρακτηριστικό του εικονίδιο υπάρχει στο Desktop των Windows και πατώντας διπλό κλικ του εκκινεί το πρόγραμμα.

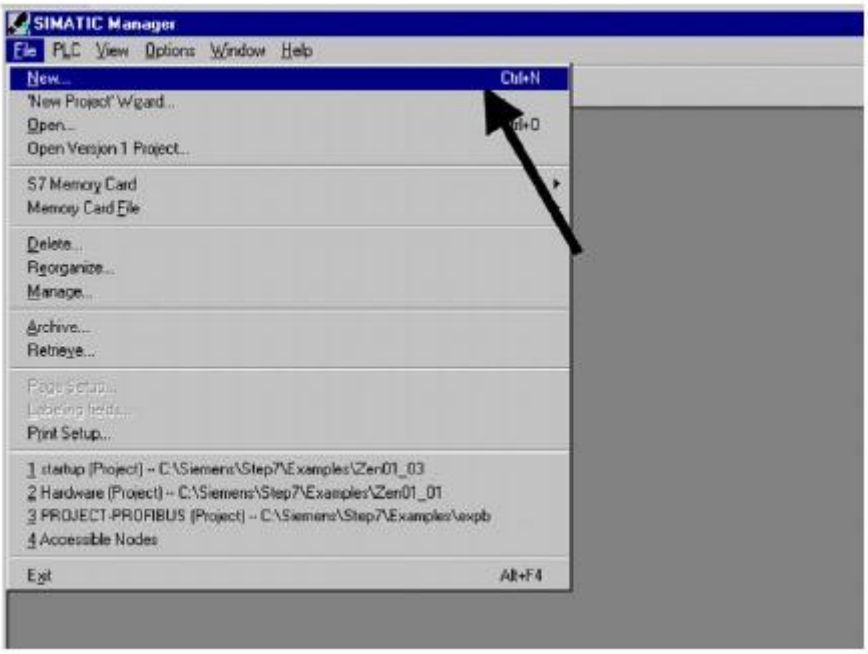
Ένας άλλος τρόπος εκκίνησης είναι και από την επιλογή Start> Simatic >Step7 ,όπως φαίνεται στο σχήμα που ακολουθεί:



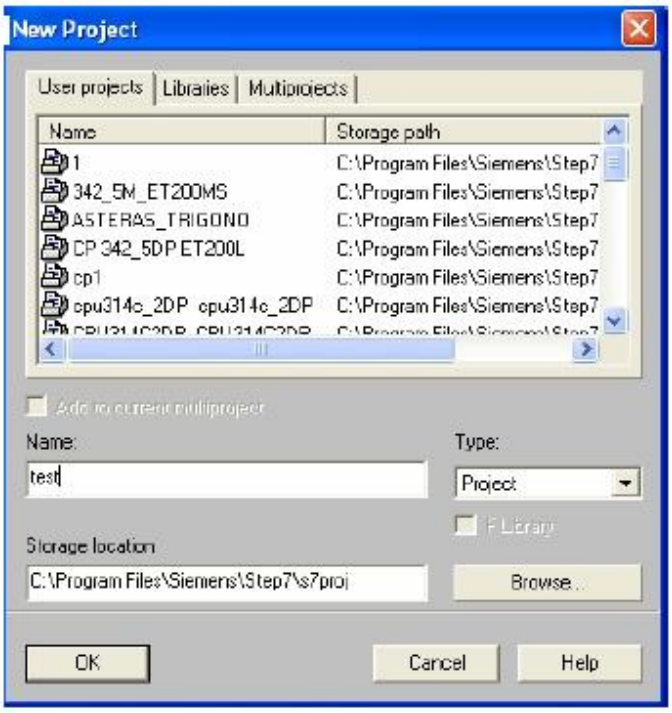
Δημιουργία Νέου Project

Ας ξεκινήσουμε τώρα με την δημιουργία του πρώτου μας project. Η σειρά εργασιών για τη δημιουργία ενός νέου project είναι:

Για να δημιουργήσουμε ένα νέο έργο (project), επιλέγουμε **File → New.**



Στο παράθυρο που ανοίγει δίνουμε το όνομα που θέλουμε να έχει το project μας.

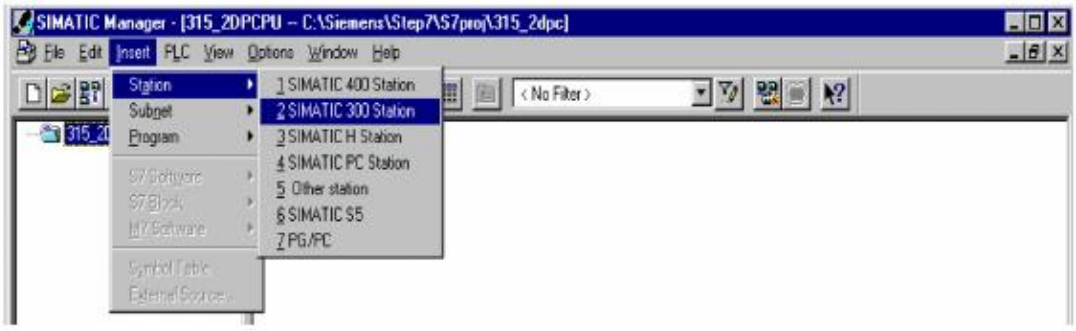


Το **«Storage location (path)»** δείχνει την διαδρομή αποθήκευσης των file που προκαθορίσαμε στο Simatic Manager (από τα Options → Customize). Επικυρώνουμε τις επιλογές μας με το OK. Έτσι δημιουργούμε τον φάκελο έργου. Στην οθόνη του υπολογιστή ανοίγει ένα παράθυρο στο οποίο μας γράφει τον τίτλο των φακέλων, την διαδρομή αρχειοθέτησης του και την δομή του.

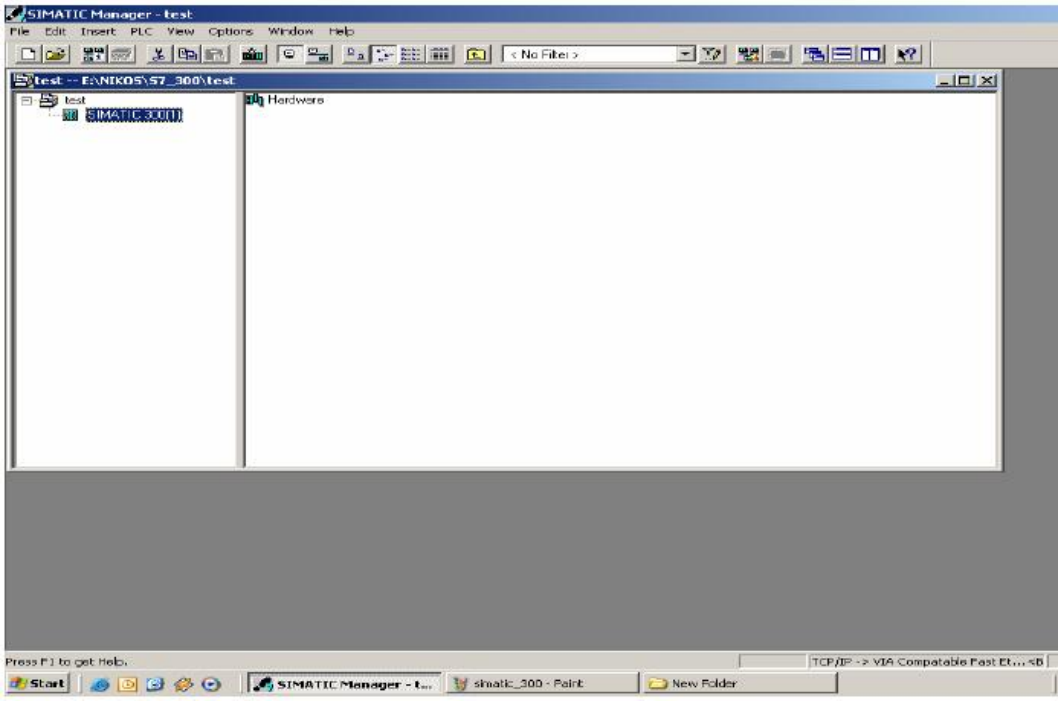
Αρχικά, κάνουμε διαμόρφωση του Hardware

ΔΙΑΜΟΡΦΩΣΗ ΣΤΑΘΜΟΥ- HARDWARE CONFIGURATION

Για να εισάγουμε ένα σταθμό αυτοματισμού στο project μας επιλέγουμε **Insert → Station →SIMATIC-300 Station.**

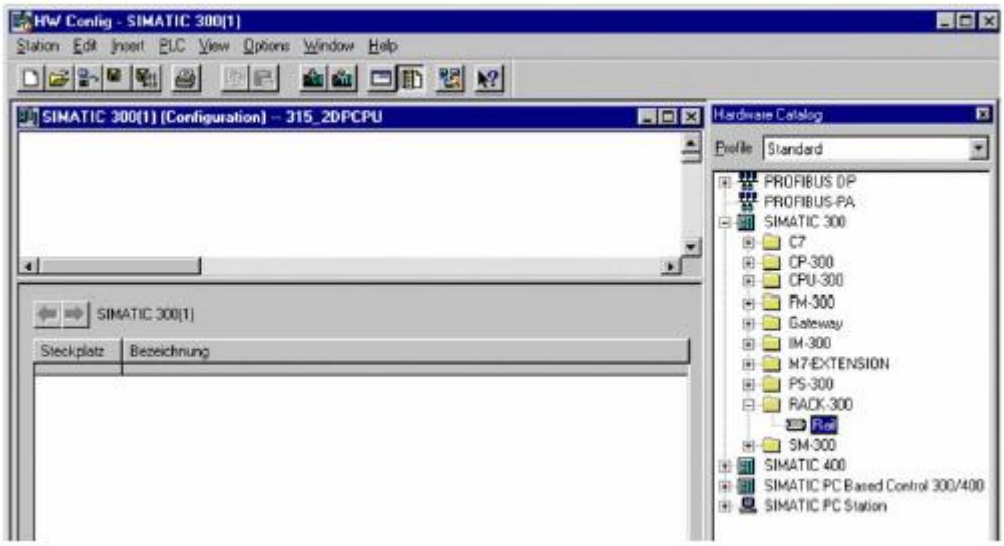


Στο έργο μας εμφανίζεται ο σταθμός και στα δεξιά η καρτέλα Hardware. Κάνουμε double click στο Hardware.



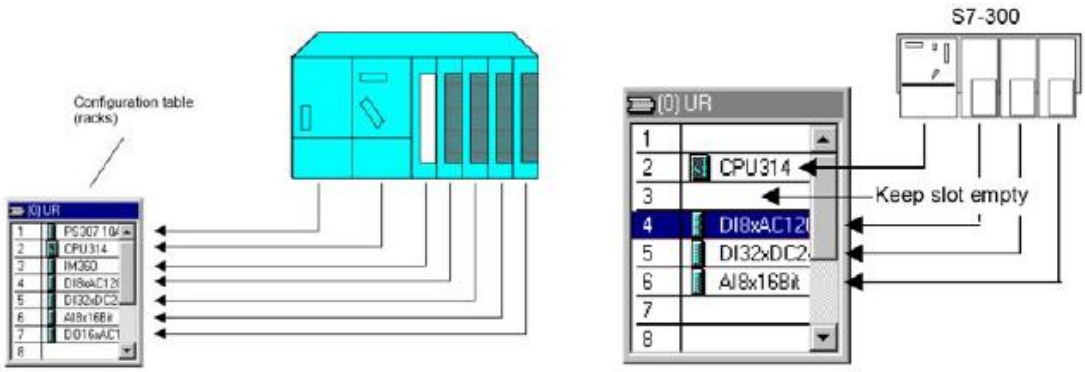
Εμφανίζεται η οθόνη διαμόρφωσης του σταθμού, και για να επιλέξουμε τα υλικά του αυτοματισμού μας ανοίγουμε την βιβλιοθήκη των υλικών πατώντας στο εικονίδιο :

ή επιλέγοντας ***view→catalog.***

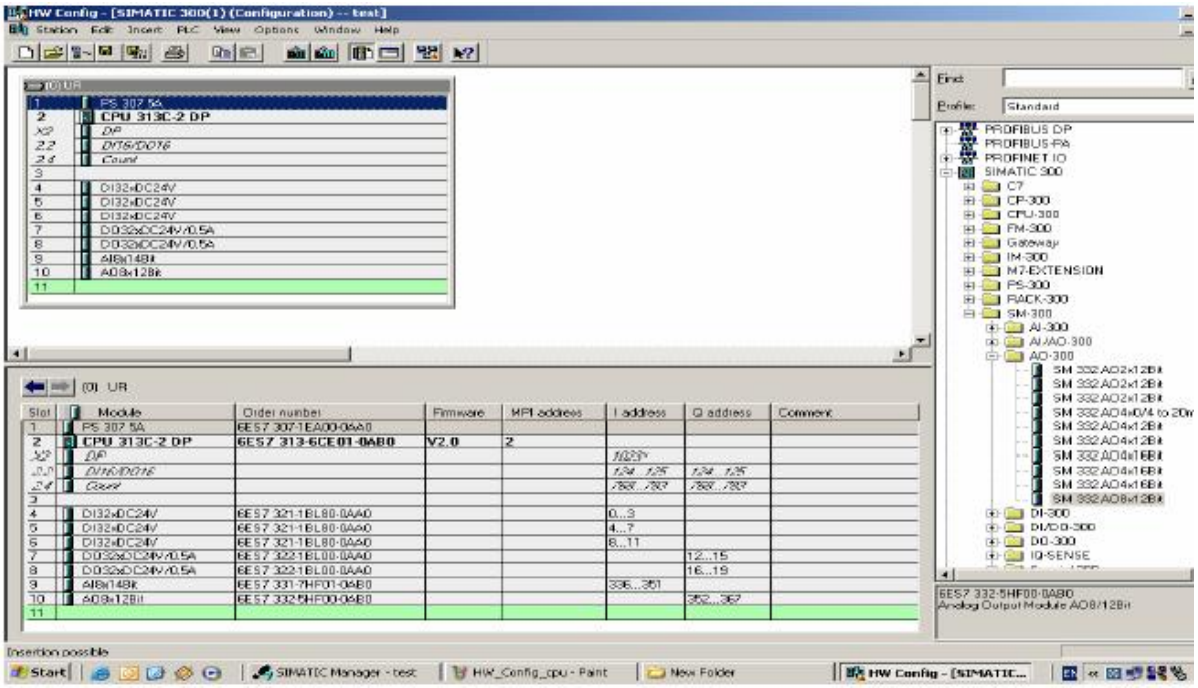


Αρχικά εισάγουμε τη ράγα στήριξης των υλικών (SIMATIC 300→RACK-300→Rail).

- Στην πρώτη θέση του Rail τοποθετούμε το τροφοδοτικό.
- Στη δεύτερη τη CPU
- Η τρίτη θέση είναι δεσμευμένη για IM(σε περίπτωση που η διαμόρφωση επεκτείνεται και σε άλλα rack).



- Στις υπόλοιπες θέσεις μπορούν να συμπληρωθούν οι κάρτες σημάτων (SM), όπως είναι και στην φυσική εγκατάσταση.



Κάνοντας διπλό κλικ πάνω σε κάθε υλικό που έχουμε τοποθετήσει στο rail μπορούμε να μπούμε στην καρτέλα των ιδιοτήτων του και να κάνουμε ρυθμίσεις (π.χ. αλλαγή διευθύνσεων, αλλαγή MPI address κ.α.)

Πως μπορούμε να δούμε τις διαθέσιμες διευθύνσεις του συστήματος μας;

Έχοντας ανοιγμένο το εργαλείο Hardware Configuration από το μενού επιλέγουμε **VIEW → ADDRESS OVERVIEW** εμφανίζεται ένα παράθυρο το οποίο περιέχει όλες τις διευθύνσεις των βαθμίδων που χρησιμοποιούνται από την επιλεγμένη CPU.

Φόρτωση Προγράμματος στο PLC

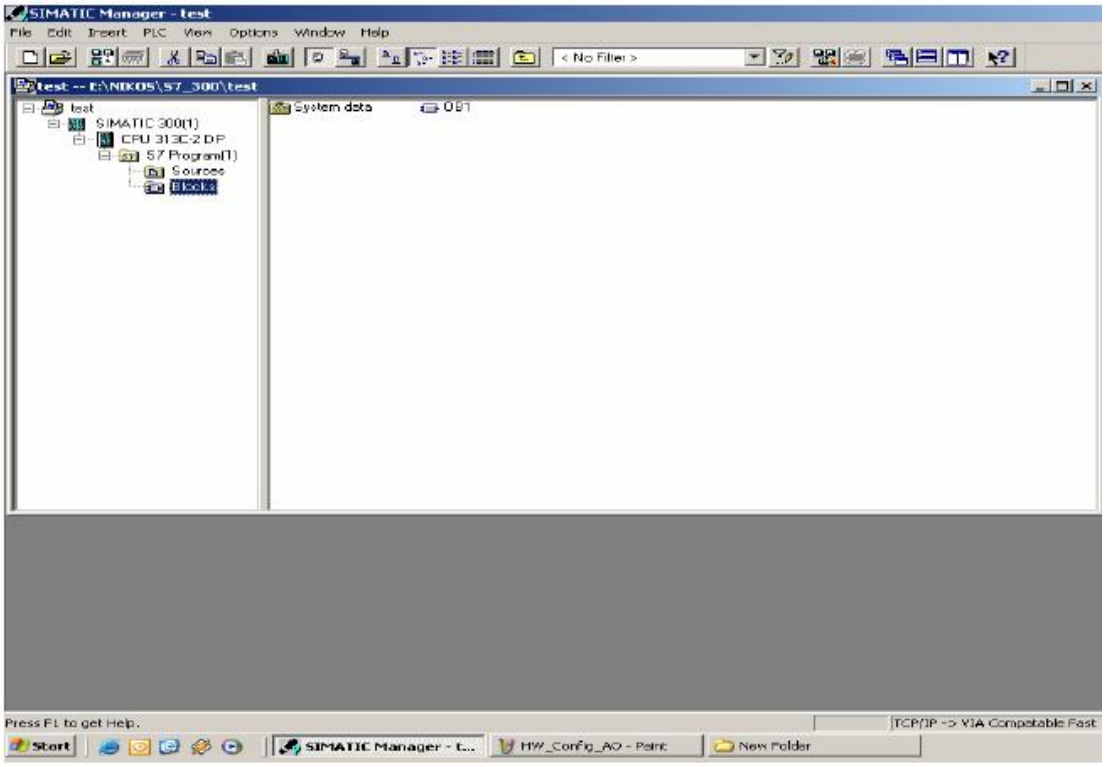
Έχοντας λοιπόν τελειώσει με τη διαμόρφωση και τις ρυθμίσεις σώζουμε το πρόγραμμα  , και το φορτώνουμε στο PLC (download)  .
Προτείνεται κατά το download η CPU να είναι σε κατάσταση STOP.

- Πριν κάνουμε download το πρόγραμμα καλό είναι να χρησιμοποιήσουμε την επιλογή Station ..Check consistency για να σιγουρευτούμε ότι δεν υπάρχουν σφάλματα στη διαμόρφωση του σταθμού μας.

- Έχοντας τελειώσει με την Hardware Configuration επανερχόμαστε στο εργαλείο Simatic Manager και στο project που δημιουργήσαμε.

Έχοντας διαμορφώσει το Hardware, ο Simatic Manager δημιούργει τον φάκελο System data και το μπλοκ οργάνωσης OB1.

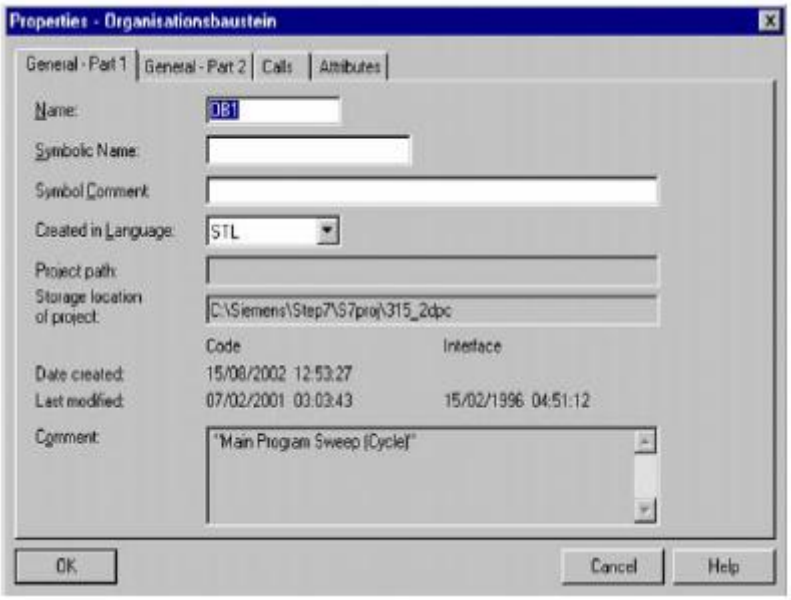
Στα αριστερά βρίσκεται η δομή του ανοικτού αντικειμένου (ιεραρχία αντικειμένου) object hierarchy, στα δεξιά βρίσκονται τα περιεχόμενα του επιλεγμένου αντικειμένου. Κάνοντας αριστερό κλικ στο σύμβολο «+», στο αριστερό μέρος του παραθύρου εμφανίζονται τα επιμέρους επίπεδα της δομής του project. Κάνοντας αριστερό κλικ στο σύμβολο «-» κρύβονται τα επιμέρους επίπεδα της δομής.



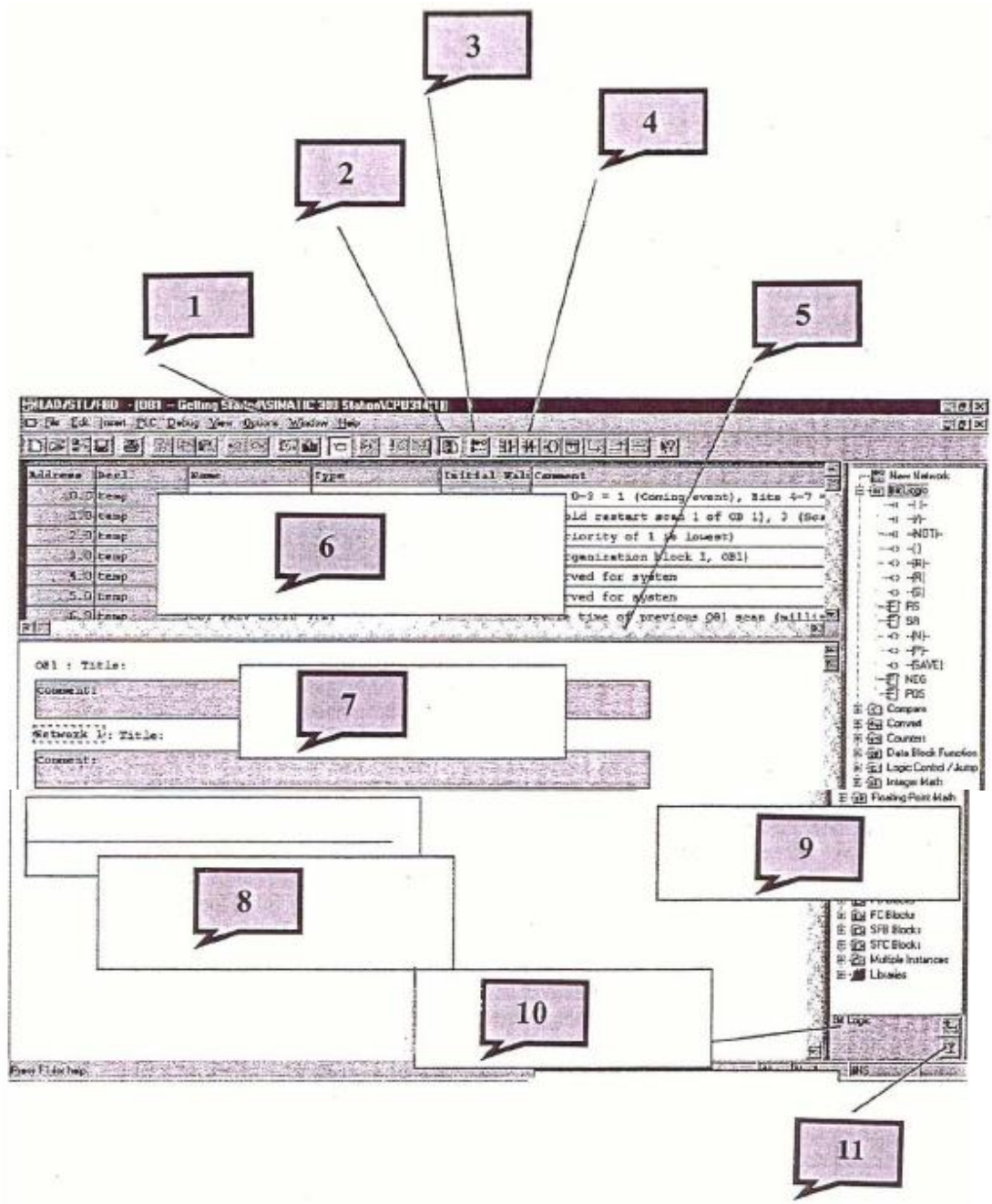
ΤΟ ΠΡΩΤΟ ΜΑΣ ΠΡΟΓΡΑΜΜΑ ΣΤΟ OB1

Είμαστε στο Simatic Manager με ανοιγμένο τα project που δημιουργήσαμε. Στο αριστερό τμήμα του παραθύρου έχουμε την δομή του project, με το ποντίκι ανοίγουμε την δομή έως ότου μου εμφανιστεί το αντικείμενο Blocks. Επιλέγοντας το αντικείμενο Blocks στο δεξί τμήμα του παραθύρου μου εμφανίζονται τα αντικείμενα System Data και OB1. Να θυμηθούμε ότι το OB1 (μπλοκ οργάνωσης) είναι το μπλοκ εκείνο το οποίο η CPU δημιουργεί από μόνη της και στο οποίο πρέπει να γράψουμε το κυρίως πρόγραμμα.

Επιλέγουμε το αντικείμενο OB1 με διπλό αριστερό κλικ από το ποντίκι, ανοίγει ένα παράθυρο διαλόγου του οποίου προς το παρών θα εξετάσουμε την πρώτη καρτέλα (General – Part 1). Σ' αυτήν την καρτέλα δίνεται η ονομασία των block (OB1) και μπορούμε να επιλέξουμε την γλώσσα προγραμματισμού (Created in Language).



Αφού καθαρίσουμε τις επιλογές μας πατάμε το μπουτόν OK. Τώρα ανοίγει το OB1 όπου μπορούμε να γράψουμε το πρώτο μας πρόγραμμα. Ανοίγοντας το OB1 στην οθόνη του υπολογιστή μας παρουσιάζεται η παρακάτω εικόνα :



όπου:

1. Αλλαγές στον τρόπο εμφάνισης της γλώσσας προγραμματισμού (γραμματοσειρά, μέγεθος γραμμάτων, χρώματα). Για την ρύθμιση από OPTIONS→ CUSTOMIZE βγάζει παράθυρο διαλόγου.

2. Ενεργοποίηση – Απενεργοποίηση του καταλόγου των στοιχείου του προγράμματος.
3. Δημιουργία νέου network.
4. Τα κυριότερα στοιχεία προγραμματισμού όταν χρησιμοποιούμε τις γλώσσες προγραμματισμού LAD ή FBD.
5. Με τον ποντίκι και αριστερό κλικ μεταβάλλουμε τα όρια του πίνακα.
6. Πίνακας δήλωσης των μεταβλητών, περιέχει τις παραμέτρους και τις τοπικές μεταβλητής του μπλοκ.
7. Περιοχή όπου δίνουμε τον τίτλο του μπλοκ ή του network και τα τυχόν σχόλια που θέλουμε να γράψουμε.
8. Περιοχή όπου γράφουμε το πρόγραμμα μας.
9. Κατάλογος των στοιχείων προγραμματισμού (διαφορετική μορφή ανάλογα την γλώσσα προγραμματισμού που επιλέξαμε).
10. Πληροφορίες σχετικά με τα στοιχεία που επιλέξαμε από τον κατάλογο προγραμματισμού.
11. Βοήθεια σχετικά με το στοιχείο που επιλέξαμε από τον κατάλογο προγραμματισμού.

ΠΩΣ ΑΛΛΑΖΟΥΜΕ ΓΛΩΣΣΑ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Έχοντας ανοιγμένο το μπλοκ του κώδικα από το μενού επιλέγω VIEW και ανοίγει παράθυρο όπου μεταξύ του άλλου υπάρχουν οι επιλογές LAD, STL, FBD. Ανάλογα την επιλογή που θα κάνουμε το ήδη υπάρχον πρόγραμμα μεταφράζεται στην γλώσσα που επιλέξαμε.

Στο πέρασμα από LAD σε STL, η μετατροπή γίνεται κανονικά, όμως στο πέρασμα από STL σε LAD ή FBD θα πρέπει να προσέξουμε εάν οι γραμμένες εντολές υπάρχουν και στην LAD ή την FBD καθώς και διάφορες άλλες συμβατότητες (π.χ. στην LAD σε κάθε NETWORK μπορεί να υπάρξει μόνο μια ανεξάρτητη έξοδος).

Αφού ολοκληρώσουμε τον προγραμματισμό του block OB1, το σώζουμε  και το φορτώνουμε  στη CPU. Στη συνέχεια μπορούμε να το δοκιμάσουμε πηγαίνοντας τη CPU σε κατάσταση Run και χρησιμοποιώντας τη λειτουργία Debug,αι το φορτώνουμε στη CPU. Στη συνέχεια μπορούμε να το δοκιμάσουμε κάνοντας κλικ στο εικονίδιο  .

3. Σκοπός της εργασίας

Σκοπός της παρούσας πτυχιακής εργασίας είναι καταρχήν η διασύνδεση κινητήριων μερών και των αισθητήρων αυτόματης μηχανολογικής διάταξης ταινιομεταφορέα και πνευματικού ρομποτικού βραχίονα δυο βαθμών ελευθερίας σε προγραμματιζόμενο λογικό ελεγκτή PLC. Στη συνέχεια θα αναπτυχθεί κατάλληλος κώδικας σε γλώσσα προγραμματισμού του συγκεκριμένου PLC της εταιρίας Siemens (Simatic S7), προκειμένου να τεθεί σε λειτουργία. Στο τέλος γίνεται επίδειξη της αυτόματης λειτουργίας της διάταξης παλετοποίησης .

4. ΕΙΣΑΓΩΓΗ ΣΤΑ ΚΥΡΙΑ ΜΕΡΗ ΤΗΣ ΔΙΑΤΑΞΗΣ ΚΑΙ ΣΤΗΝ ΛΕΙΤΟΥΡΓΙΑ ΑΥΤΗΣ

Στο μέρος αυτό της εργασίας μας αναπτύσσονται τα κύρια μέρη της διάταξης ταινιομεταφορέα και ρομποτικού βραχίονα. Για να πετύχει χρειάζεται συστηματική διαδικασία και σωστό προγραμματισμό. Πρόκειται για μια μηχανή μεταφοράς και εναπόθεσης πλαστικών δίσκων με τη χρήση προγραμματιζόμενου λογικού ελεγκτή PLC.Περιέχει τρείς απλούς αυτοματισμούς και παρουσιάζεται η σημαντικότητα αυτών στην παραγωγική διαδικασία.

Αρχικά παρουσιάζονται τα μέρη που είναι μηχανικά, ηλεκτρολογικά, ηλεκτρονικά και πνευματικά της διάταξης καθώς και οι φωτογραφίες με την απεικόνιση αυτών. Στη συνέχεια θα αναπτυχθεί η περιγραφή λειτουργίας της αυτόματης μονάδας παλετοποίησης για την πλήρη κατανόηση αυτής.

4.1 ΜΕΡΗ ΤΗΣ ΔΙΑΤΑΞΗΣ ΚΑΙ ΕΠΕΞΗΓΗΣΗ ΑΥΤΩΝ

Ανάπτυξη ηλεκτρολογικού μέρους

Από την παροχή της πρίζας (τριφασικό 400V) το ρεύμα εισέρχεται στο γενικό διακόπτη **R.S.T** μαζί με τον ουδέτερο **N** και τη γείωση.

Έπειτα περνάει στο γενικό τριφασικό ασφαλοδιακόπτη **FU1**, διαδοχικά στο ρελέ ισχύος, στο θερμικό και κατόπιν στο κινητήρα. Με το ρελέ ελέγχουμε το ξεκίνημα και τη στάση του κινητήρα, ενώ με το θερμικό την προστασία του από την απώλεια κάποιας φάσης.

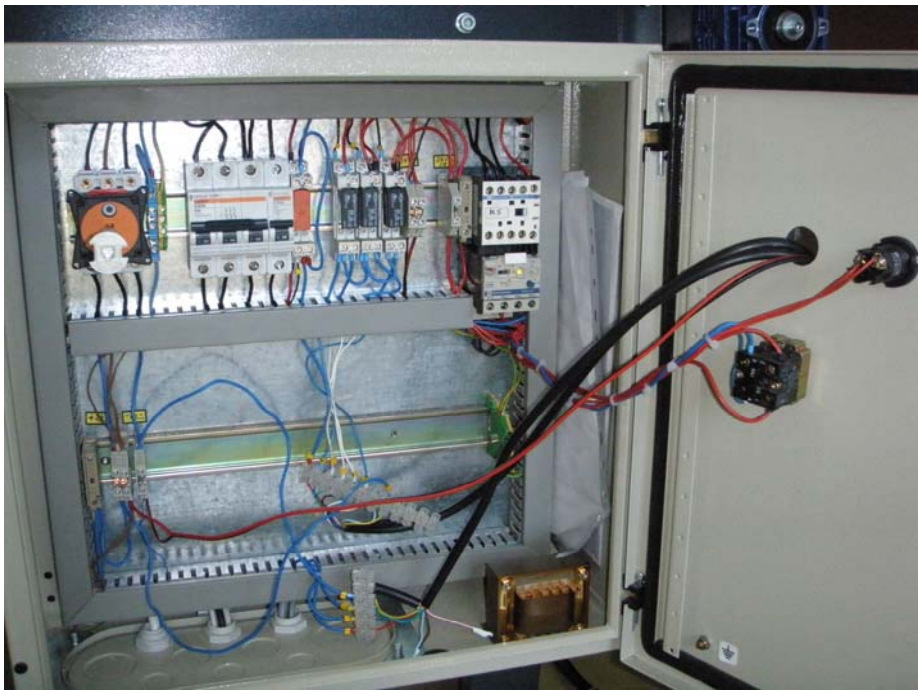
Από μια φάση και τον ουδέτερο Ν τροφοδοτούμε το πρωτεύων πηνίο του μετασχηματιστή με τον ασφαλοδιακόπτη **FU2.** Το δευτερεύων πηνίο του μετασχηματιστή βγάζει **24V AC,** για την τροφοδοσία της λάμπας, του ρελέ ισχύος **K5,** των βαλβίδων **Y1, Y2, Y3,** μέσω των επαφών του ρελέ **K1, K2, K3**.

Με δυο γραμμές από τον ασφαλοδιακόπτη **FU2** και από τον ουδέτερο **N** τροφοδοτούμε στις θέσεις **L-N** το PLC και έτσι τελειώνουν οι γραμμές με υψηλή τάση. Οι οποίες χαρακτηρίζονται με πιο έντονο χρώμα στο σχέδιο.

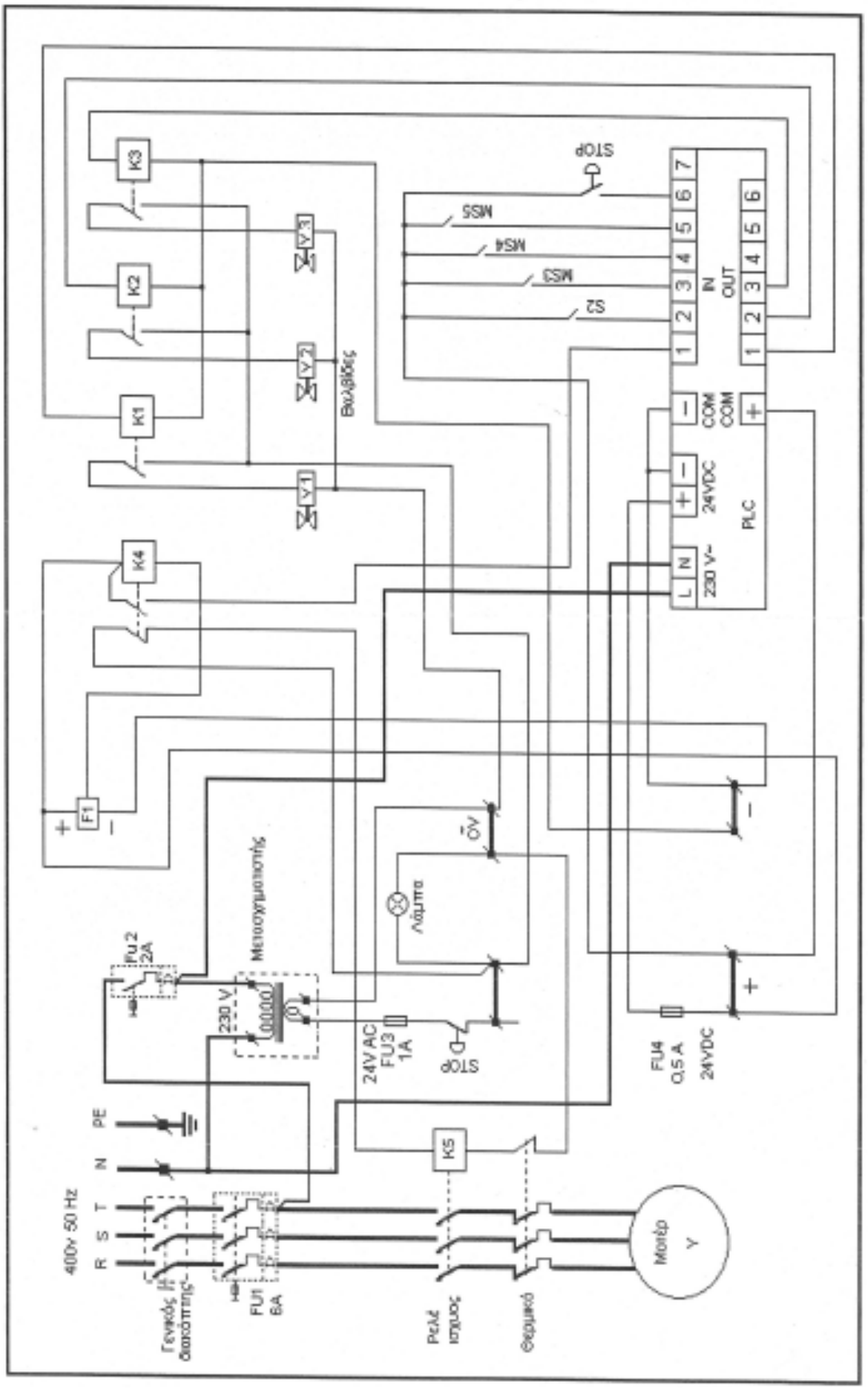
Πάνω στο PLC έχουμε μία πηγή **24V DC** με την οποία τροφοδοτούμε, μέσω μιας ασφάλειας **FU4,** το φωτοκύτταρο και τα τερματικά αισθητήρια των εμβόλων. Καθώς και τα ρελέ **K1,K2, K3, K4,** μέσω των εξόδων **OUT** του PLC.

Παρακάτω απεικονίζεται ο ηλεκτρολογικός πίνακας, καθώς και στη συνέχεια το σχεδιάγραμμα του ηλεκτρολογικού μέρους.

Ηλεκτρολογικός πίνακας



Ηλεκτρολογικό σχεδιάγραμμα



Διακόπτης START

Ο διακόπτης αυτός ξεκινάει τον ηλεκτροκινητήρα και δίνει ρεύμα στην όλη διάταξη.

Διακόπτης STOP

Ο διακόπτης αυτός σταματάει ακαριαία κάθε λειτουργία της διάταξης.

Μειωτήρας

Μειώνει τις στροφές του ηλεκτροκινητήρα και έτσι δίνει κίνηση με αυξημένη δύναμη στον ταινιόδρομο.

Ηλεκτροκινητήρας

Είναι ένας ηλεκτρικός κινητήρας που έχει σκοπό την κίνηση του ταινιόδρομου.

Είναι ο ταινιόδρομος όπου πάνω σε αυτόν μεταφέρονται οι δίσκοι για την παλετοποίηση τους.

Τεντωτήρας

Είναι ένα εξάρτημα που κρατάει τον ταινιόδρομο τεντωμένο και δεν τον αφήνει να χαλαρώσει.

Σύστημα ευθυγράμμισης δίσκων

Χρησιμοποιείται για να τοποθετεί τους δίσκους σε μια συγκεκριμένη θέση που πρέπει να βρίσκονται για να παλετοποιηθούν.

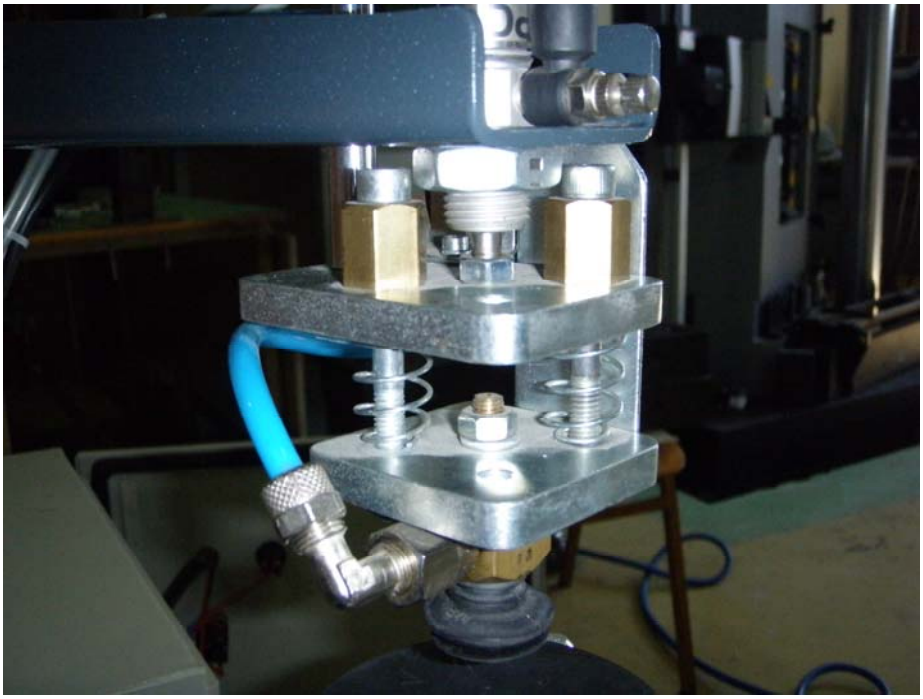
Φωτοκύτταρο

Όταν φτάσει ο δίσκος μπροστά από το φωτοκύτταρο δίνει εντολή να σταματήσει να κινείται ο ταινιόδρομος.

Έμβολο C1 κάθετης κίνησης

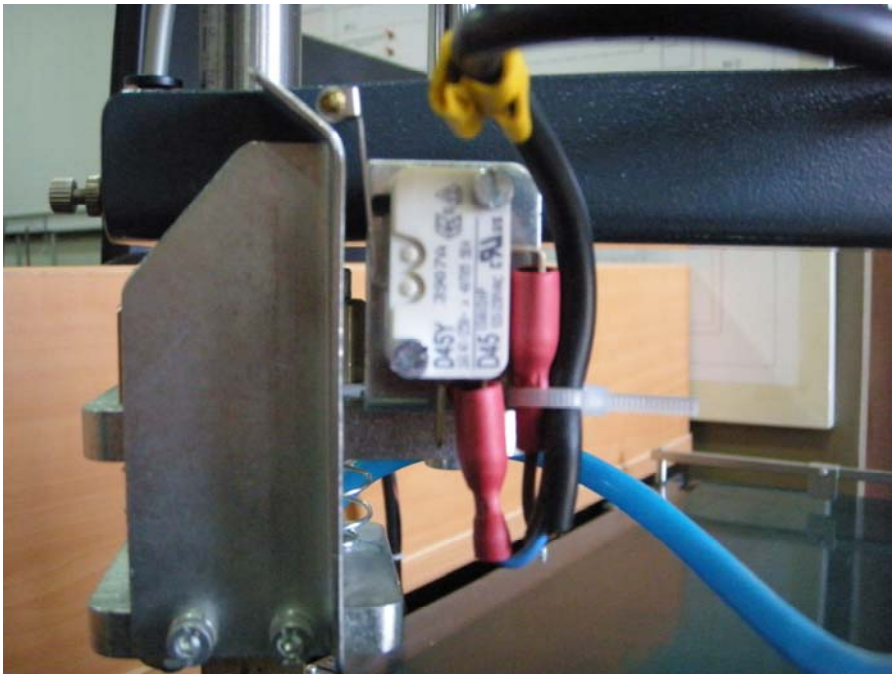
Το έμβολο στηρίζεται πάνω στον βραχίονα και λειτουργία του είναι να ανεβαίνει και να κατεβαίνει, όταν βρίσκεται στα σημεία από όπου παίρνει και αφήνει τους δίσκους.

Κεφαλή συστήματος ανύψωσης



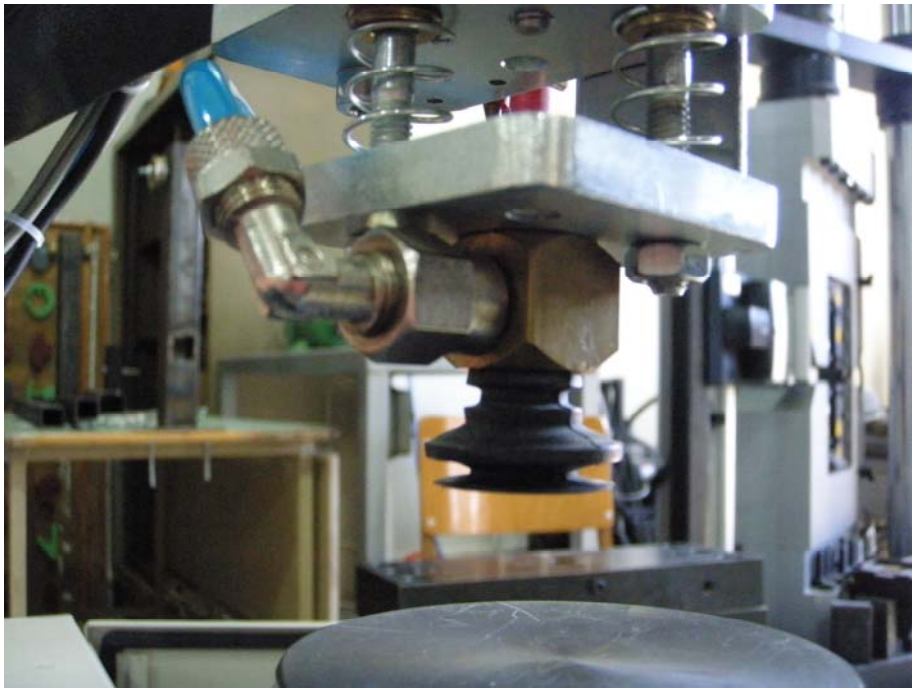
Η κεφαλή αυτή συνδέεται με το κάθετο έμβολο και φέρει πάνω της το σύστημα αναρρόφησης και το διακόπτη S2 του συστήματος ανύψωσης.

Διακόπτης S2 συστήματος ανύψωσης



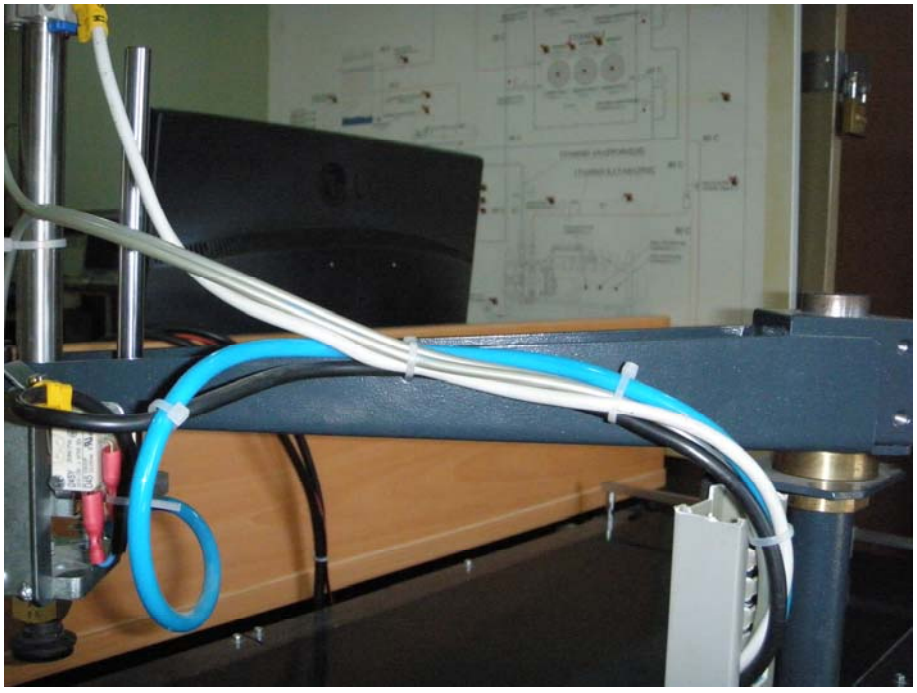
Όταν ενεργοποιείται αυτός ο διακόπτης δίνει εντολή να κάνει βάκουμ και στη συνέχεια να ανέβει το έμβολο προς τα πάνω.

Σύστημα αναρρόφησης



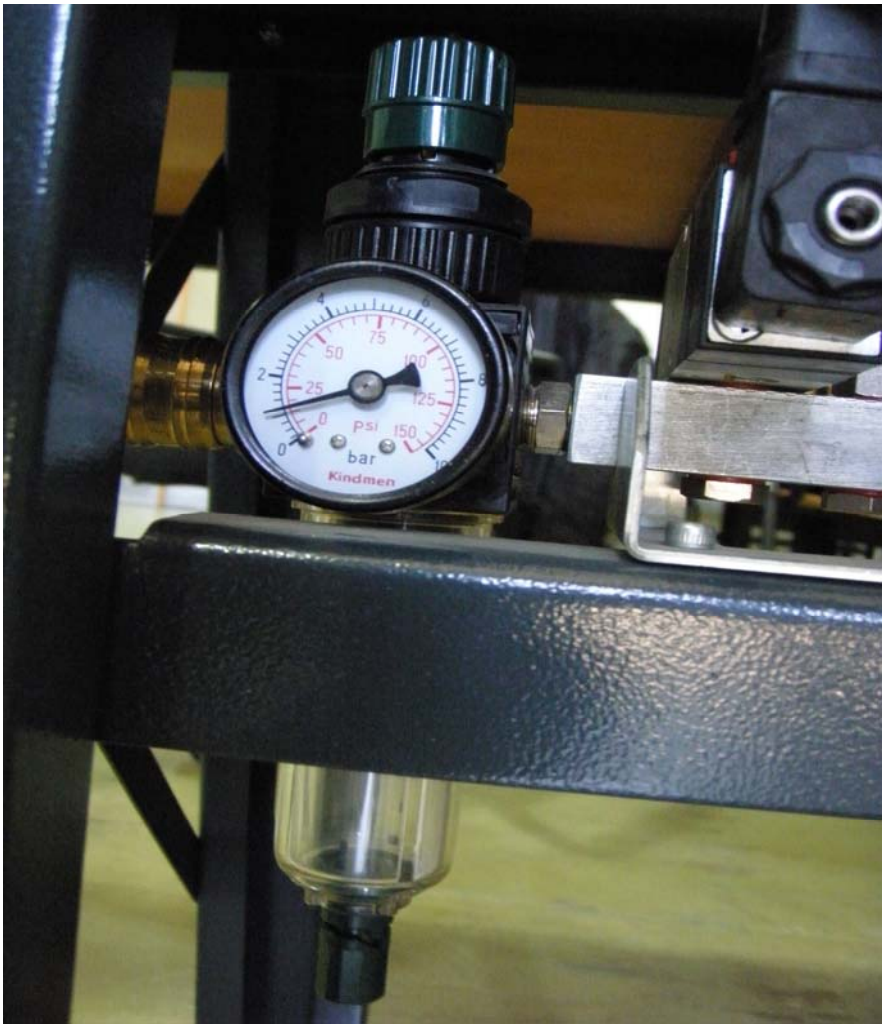
Βρίσκεται στο κάτω μέρος της κεφαλής και μόλις ενεργοποιηθεί ο διακόπτης S2 ξεκινάει την αναρρόφηση για την μεταφορά των δίσκων.

Βραχίονας οριζόντιας κίνησης



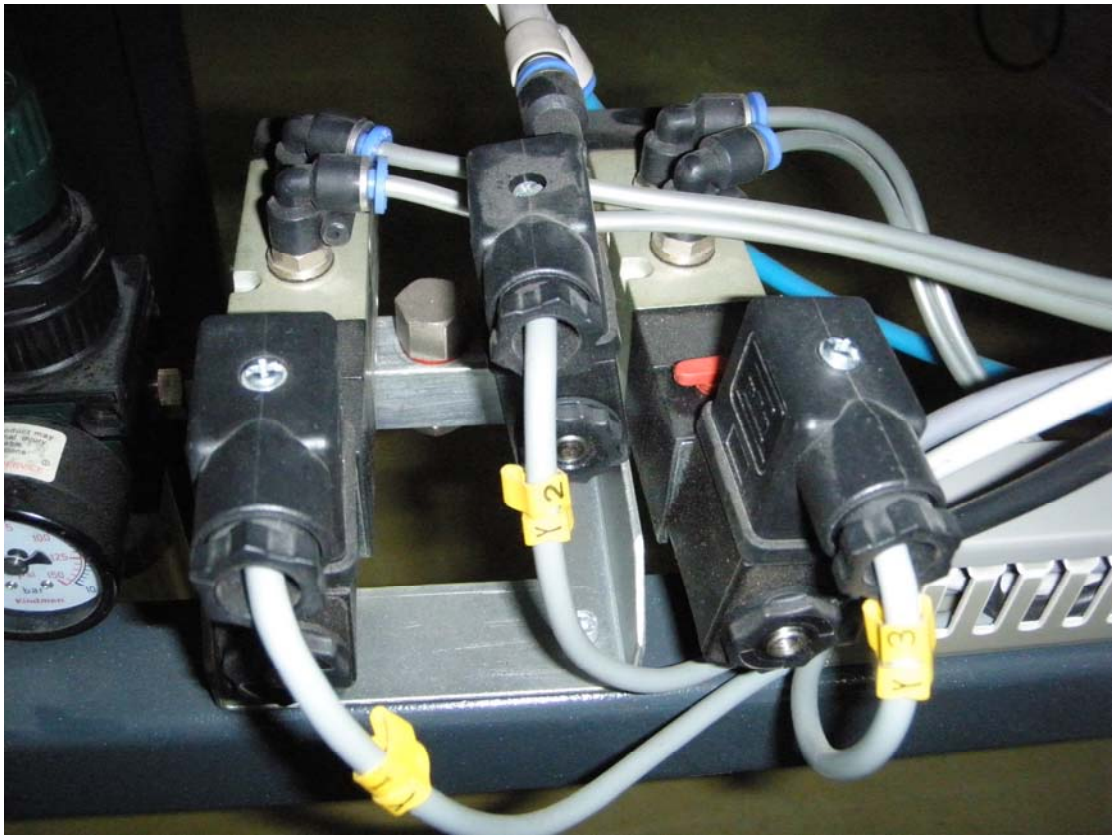
Ο βραχίονας αυτός κινείται οριζόντια από το σημείο του ταινιόδρομου από όπου παίρνει τους δίσκους ,μέχρι το σημείο της παλετοποίησής τους.

Μανόμετρο



Είναι το όργανο που ελέγχει την πίεση του αέρα σε συγκεκριμένο όριο για τη σωστή λειτουργία όλων των απαραίτητων πνευματικών κινήσεων του συστήματος και για την απαραίτητη δύναμη αναρρόφησης για την ανύψωση των δίσκων.

Βαλβίδες Y1,Y2,Y3



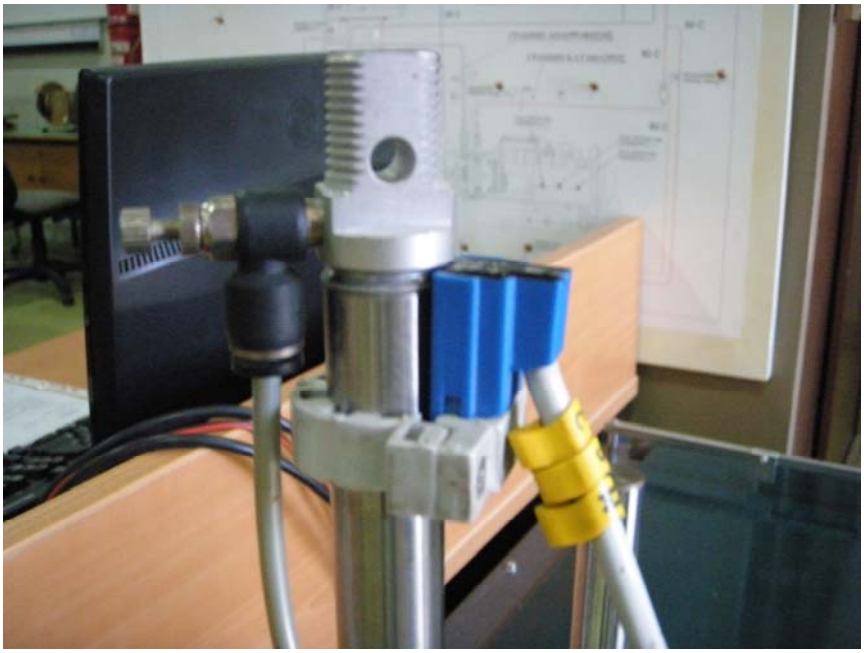
Βαλβίδα Y1 (αριστερά): όταν ανοίγει η Y1 το κάθετο έμβολο κατεβαίνει, και όταν κλείνει τότε το έμβολο ανεβαίνει.

Βαλβίδα Y2 (κέντρο): όταν ανοίγει η Y2 τίθεται σε λειτουργία το σύστημα αναρρόφησης.

Βαλβίδα Y3 (δεξιά): όταν ανοίγει η Y3 ενεργοποιεί τον τερματικό διακόπτη MS4.

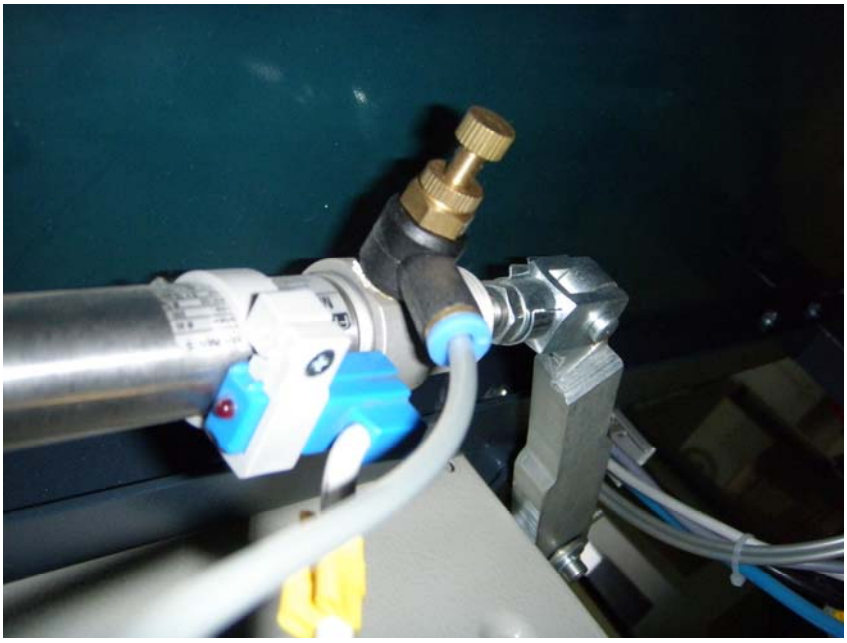
Τερματικοί διακόπτες

Διακόπτης MS3



Όταν ενεργοποιείται ο MS3 το κάθετο έμβολο βρίσκεται στην άνω τερματική του θέση.

Διακόπτης MS4

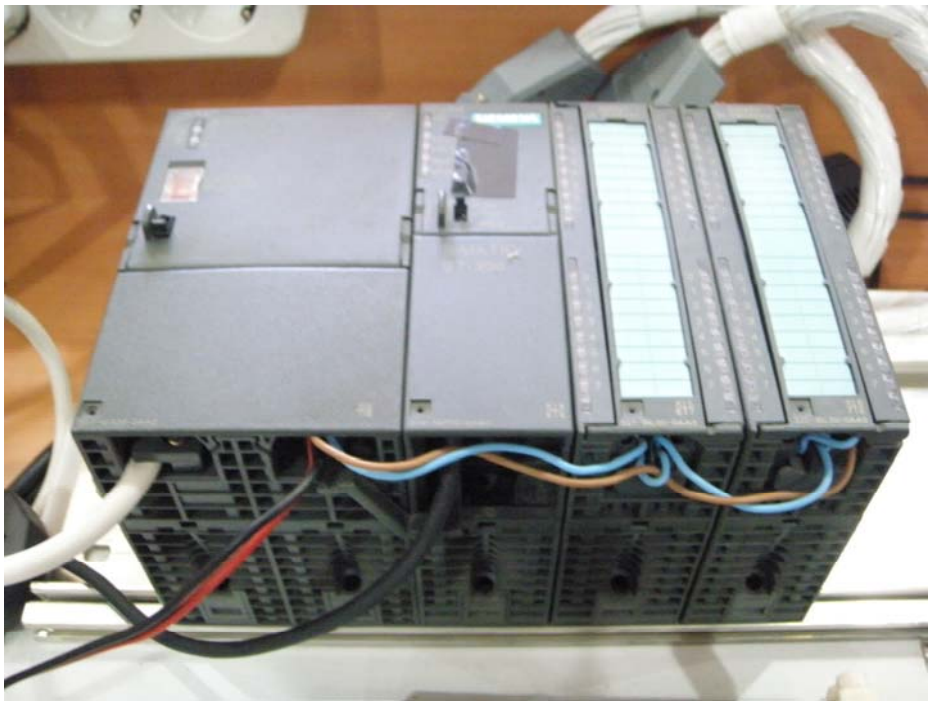


Όταν ενεργοποιείται ο MS4 ο βραχίονας βρίσκεται στην θέση πάνω από τον ταινιόδρομο.

Διακόπτης MS5

Όταν ενεργοποιείται ο MS5 ο βραχίονας βρίσκεται στην θέση της παλετοποίησης.

Προγραμματιζόμενος λογικός ελεγκτής PLC



Siemens Simatic S7-300

Καλώδια σύνδεσης εισόδων και εξόδων του PLC με το μηχάνημα

Αντάπτορας σύνδεσης του PLC με τον υπολογιστή για τον προγραμματισμό του (pc Adaptor)

4.2 ΠΕΡΙΓΡΑΦΗ ΛΕΙΤΟΥΡΓΙΑΣ ΠΡΟΤΥΠΗΣ ΔΙΑΤΑΞΗΣ

Με το γύρισμα του διακόπτη **START** στη θέση **1(ON) ,** ο κινητήρας μπαίνει σε λειτουργία, και μέσω του μειωτήρα ο οποίος συνδέεται μ 'αυτόν, θέτει σε κίνηση με μια σταθερή ταχύτητα του ιμάντα μεταφοράς των δίσκων.

Τοποθετούμε ένα πλαστικό δίσκο στην είσοδο του ιμάντα και με τη βοήθεια ευθυγράμμισης των δίσκων κινείται πάνω του σε μια συγκεκριμένη θέση. Μόλις ο δίσκος φτάσει στο ύψος της δέσμης του φωτοκύτταρου δίνει εντολή να σταματήσει ο κινητήρας και κατά συνέπεια να σταματήσει να κινείται και ο ιμάντας.

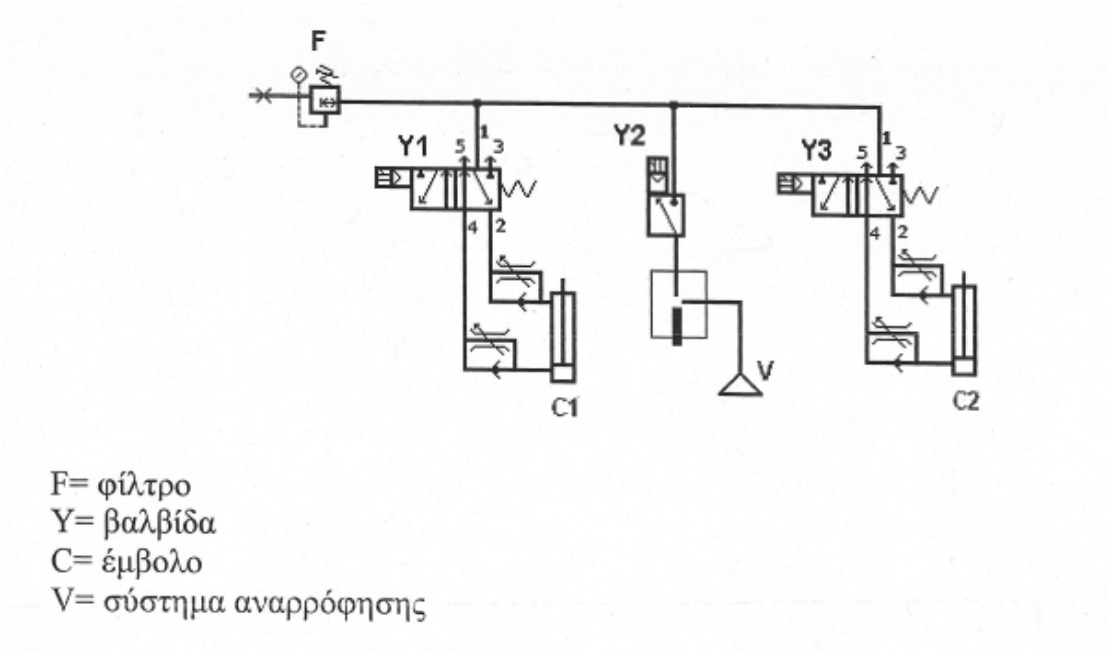
Ανοίγει η βαλβίδα **Y1** και το κάθετο έμβολο **C1** του συστήματος ανύψωσης κατεβαίνει. Με τη συμπίεση της κεφαλής του συστήματος ανύψωσης ενεργοποιείται ο διακόπτης **S2**. Ανοίγει η βαλβίδα **Y2** και η βεντούζα του συστήματος αναρρόφησης που βρίσκεται στο κάτω μέρος της κεφαλής ξεκινάει να αναρροφά το δίσκο, με την κατάλληλη πίεση που ρυθμίζουμε από το μανόμετρο για να έχει την απαραίτητη δύναμη αναρρόφησης για την ανύψωση αυτού. Η βαλβίδα **Y1** κλείνει και το έμβολο **C1** ξεκινάει να ανεβαίνει μαζί με το δίσκο ο οποίος συνεχίζει να αναρροφάται.

Μόλις το έμβολο φτάσει στην πάνω θέση του, ανοίγει η βαλβίδα **Y3**, ενεργοποιεί τον τερματικό διακόπτη **MS3** και επειδή είναι ενεργοποιημένος συγχρόνως και ο τερματικός διακόπτης , **MS4** το σύστημα ανύψωσης μαζί με το δίσκο μέσω του βραχίονα οριζόντιας κίνησης πηγαίνει στη θέση εναπόθεσης των δίσκων.

Μόλις φτάσει στη θέση παλετοποίησης η βαλβίδα **Y1** ανοίγει πάλι και το κάθετο έμβολο **C1** κατεβαίνει μέχρι ο διακόπτης **S2** που βρίσκεται πάνω στην κεφαλή να πατηθεί. Αφού γίνει αυτό οι βαλβίδες **Y1** και **Y2** κλείνουν. Με το κλείσιμο αυτών σταματάει η αναρρόφησης του δίσκου και το κάθετο έμβολο **C1** αρχίζει να ανεβαίνει.

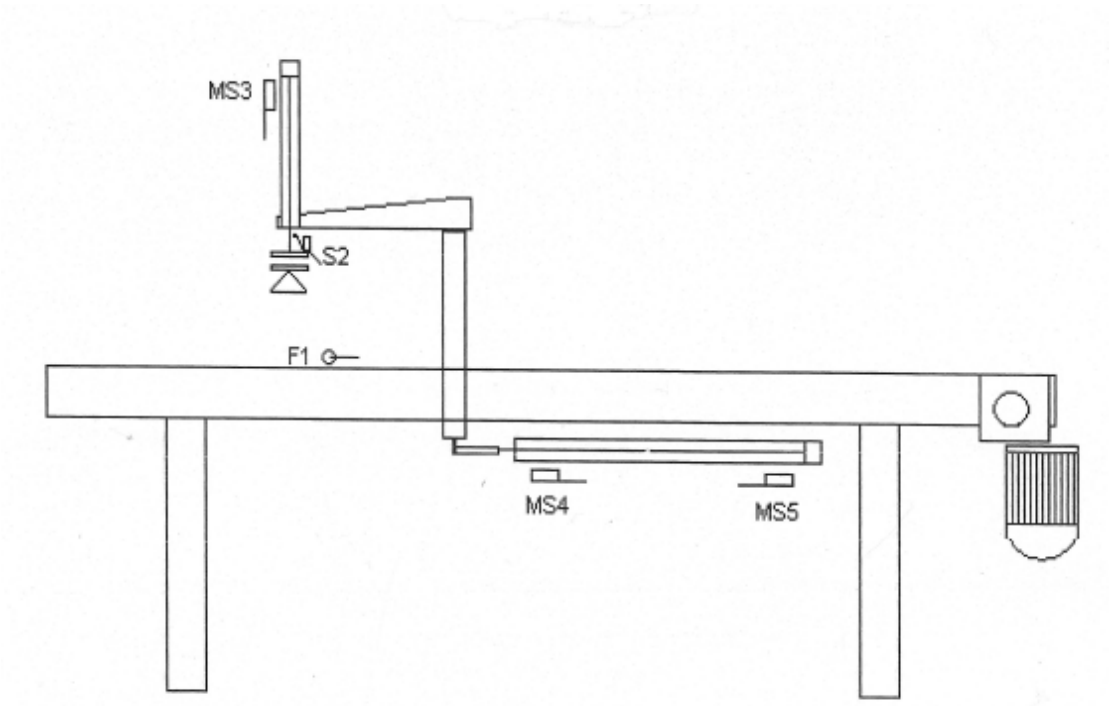
Όταν το έμβολο φτάσει στην άνω θέση του ενεργοποιεί το διακόπτη **MS3** και επειδή είναι ενεργοποιημένος συγχρόνως αυτή τη φορά ο τερματικός διακόπτης **MS5**, η **Y3** βαλβίδα κλείνει και το σύστημα ανύψωσης, τώρα χωρίς το δίσκο, πάλι με τη βοήθεια του βραχίονα γυρίζει στην αρχική του θέση δηλαδή πάνω από τον ιμάντα, για να ακολουθήσει πάλι η ίδια διαδικασία με τον επόμενο δίσκο.

Πνευματικό πλάνο



F= φίλτρο
Y= βαλβίδα
C= έμβολο
V= σύστημα αναρρόφησης

Διακόπτες



5.ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΤΗΣ ΔΙΑΤΑΞΗΣ

Στον προγραμματισμό της διάταξης αυτόματης μονάδας παλετοποίησης χρησιμοποιούμε PLC της Siemens σειράς S7-300. Το συγκεκριμένο που εργαστήκαμε εμείς αποτελείται από τα παρακάτω κομμάτια:

- Από τροφοδοτικό του τύπου: PS 307-5A
- Από την CPU τύπου: 314 SF
- Από την κάρτα εισόδων: SM 321, DI32 X DC24V, η οποία έχει 32 εισόδους και τις τροφοδοτεί με 24 volt.
- Και από την κάρτα εξόδων: SM 322, DO32 x DC24v / 0,5A, η οποία και αυτή έχει 32 εξόδους και μπορεί να δώσει εξόδους μέχρι 24volt και 0,5Ampere.

5.1 Γλώσσα προγραμματισμού LADDER, οι ιδιαιτερότητες και η δομής της.

Η γλώσσα LADDER (LAD) είναι γλώσσα που χρησιμοποιεί τα ηλεκτρολογικά γραφικά. Το πρόγραμμα σε γλώσσα LADDER μοιάζει με το ηλεκτρολογικό σχέδιο του αυτοματισμού.

Οι ιδιαιτερότητες που έχουμε να αντιμετωπίσουμε στη γλώσσα αυτή είναι:

- Χρησιμοποιούνται σύμβολα από την Αμερικάνικη τυποποίηση και όχι από την Ευρωπαϊκή με την οποία είμαστε εξοικειωμένοι.
- Το σχέδιο – πρόγραμμα είναι τυποποιημένο, δεν έχουμε δηλαδή την ελευθερία που έχουμε κατά τη σχεδίαση. Για παράδειγμα σε κάθε κλάδο μπορούμε να έχουμε περιορισμένο αριθμό στοιχείων προγράμματος (διακόπτες και επαφές). Επίσης, δεν μπορούμε να κάνουμε οποιασδήποτε μορφής διακλάδωση.

Η δουλειά που έχουμε να κάνουμε δηλαδή στη γλώσσα LADDER είναι να προσαρμόσουμε το σχέδιο του αυτοματισμού, στα δεδομένα που απαιτεί η γλώσσα.

Όσον αφορά τη δομή της LADDER, το διάγραμμα επαφών της σχεδιάζεται όχι “κατακόρυφα” αλλά “οριζόντια”. Δηλαδή σε ένα πρόγραμμα LADDER έχουμε δύο παράλληλες κατακόρυφες γραμμές (μπάρες), η αριστερή γραμμή παριστάνει τη μπάρα τροφοδοσίας με το υψηλό δυναμικό (+) και η δεξιά γραμμή τη μπάρα τροφοδοσίας με το χαμηλό δυναμικό (-). Μεταξύ των δύο γραμμών σχεδιάζουμε οριζόντια τους κλάδους του κυκλώματος. Κάθε κλάδος του διαγράμματος LADDER, που ξεκινά από την αριστερή μπάρα και καταλήγει στη δεξιά μπάρα, αποτελεί μια γραμμή προγράμματος, η οποία αντιστοιχεί στην μάδα εντολών της γλώσσας λίστα εντολών.

5.2 Πίνακας συμβόλων

ΕΙΣΟΔΟΙ:

(I0.0) : **Φωτοκύτταρο.** Ενεργοποιείται μόλις φτάσει το αντικείμενο μπροστά από αυτό.

S2 → (IO.1) : **Έλεγχος πιέσεως (βάκουμ).** Ενεργοποιείται με το πάτημα του διακόπτη S2

MS3→(I0.2) : **Κάθετο τερματικό.** Ενεργοποιείται όταν το κάθετο έμβολο βρίσκεται στην άνω νεκρή του θέση.

MS4→(I0.3) : **Οριζόντιο έμβολο (ταινιόδρομος).** Ενεργοποιείται όταν το οριζόντιο έμβολο βρίσκεται στην θέση πάνω από τον ταινιόδρομο.

MS5→(I0.4) : **Οριζόντιο έμβολο (μπαλέτα).** Ενεργοποιείται όταν το οριζόντιο έμβολο βρίσκεται στην θέση πάνω από το σημείο παλετοποίησης .

(I0.5) : **Γενικό STOP.** Σταματάει κάθε λειτουργία της διάταξης.

ΕΞΟΔΟΙ:

(Q4.0) : **Κάθετο έμβολο.** Όταν ενεργοποιείται το κάθετο έμβολο μετακινείται από την άνω στην κάτω νεκρή του θέση.

(Q4.1) : **Βάκουμ.** Όταν ενεργοποιείται δημιουργείται κενό στην βεντούζα για την αναρρόφηση των αντικειμένων.

(Q4.2) : **Οριζόντιο έμβολο.** Όταν ενεργοποιείται το οριζόντιο έμβολο μετακινείται από την θέση του πάνω από τον ταινιόδρομο στην θέση της παλετοποιήσης .

Χρονικό

Στον προγραμματισμό επίσης χρησιμοποιούμε τον τύπο χρονικού : S_OFFDT που μας βοηθάει να καθορίσουμε τον χρόνο που εκτελεί μια συγκεκριμένη κίνηση.

Στις παρακάτω 2 σελίδες ακολουθούν τα διαγράμματα προγραμματισμού.

5.3 Διαγράμματα του προγραμματισμού της διάταξης

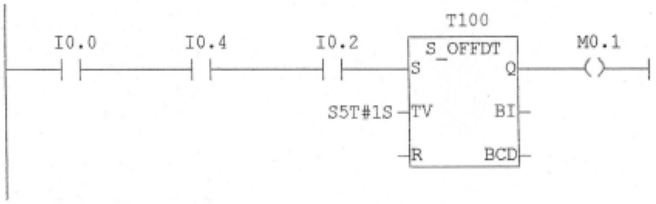
OB1 - <offline>

Name: Family:
Author: Version: 0.1
Block version: 2
Time stamp Code: 12/5/2011 3:06:26 μμμμ
Interface: 15/2/1996 4:51:12 μμμμ
Lengths (block/logic/data): 00226 00100 00020

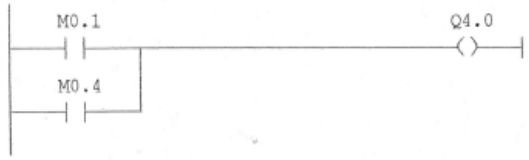
Name	Data Type	Address	Comment
TEMP		0.0	
OB1_EV_CLASS	Byte	0.0	Bits 0-3 = 1 (Coming event), Bits 4-7 = 1 (Event class 1)
OB1_SCAN_1	Byte	1.0	1 (Cold restart scan 1 of OB 1), 3 (Scan 2-n of OB 1)
OB1_PRIORITY	Byte	2.0	Priority of OB Execution
OB1_OB_NUMBR	Byte	3.0	1 (Organization block 1, OB1)
OB1_RESERVED_1	Byte	4.0	Reserved for system
OB1_RESERVED_2	Byte	5.0	Reserved for system
OB1_PREV_CYCLE	Int	6.0	Cycle time of previous OB1 scan (milliseconds)
OB1_MIN_CYCLE	Int	8.0	Minimum cycle time of OB1 (milliseconds)
OB1_MAX_CYCLE	Int	10.0	Maximum cycle time of OB1 (milliseconds)
OB1_DATE_TIME	Date_And_Time	12.0	Date and time OB1 started

Block: OB1 "Main Program Sweep (Cycle)"

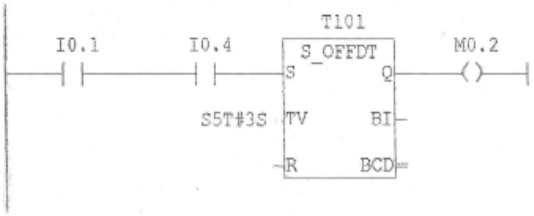
Network: 1 Βοηθητικό ρελέ κάθετου εμβόλου M0.1



Network: 2 Ενεργοποίηση κάθετου εμβόλου Q4.0

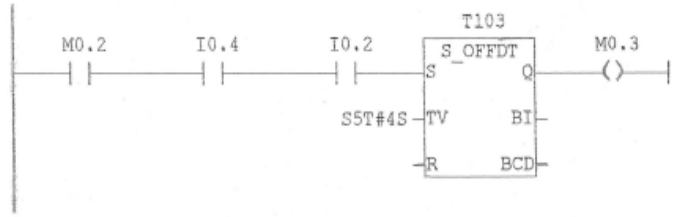


Network: 3 Βοηθητικό ρελέ βάκουμ M0.2



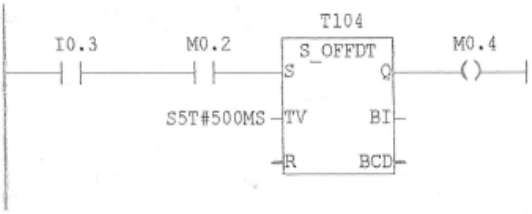
Network: 4 Ενεργοποίηση βάκουμ Q4.1

Network: 5 Βοηθητικό ρελέ οριζόντιου βραχίονα M0.3



Network: 6 Ενεργοποίηση οριζόντιου βραχίονα Q4.2

Network: 7 Βοηθητικό ρελέ απενεργοποίησης βάκουμ M0.4



6.ΣΥΝΟΨΗ-ΣΥΜΠΕΡΑΣΜΑΤΑ

Μετά και την ολοκλήρωση και κατά συνέπεια εκπόνηση του προγράμματος με την κατάλληλη γλώσσα (σωστά διατυπωμένη), που επιλέξαμε να δουλέψουμε (LAD) την οποία και κατανοήσαμε καλύτερα από τις άλλες δύο (STL,FBD) που έχουμε σαν επιλογή, καταφέραμε μετά από πολύ προσπάθεια να βάλουμε σε λειτουργία την όλη διάταξη που προαναφέραμε με όλες τις απαραίτητες κινήσεις και στην σωστή σειρά που πρέπει να εκτελούνται για την ολοκλήρωση ενός πλήρες κύκλου λειτουργίας του μηχανήματος. Μέσα από τη χρήση Προγραμματιζόμενου Λογικού Ελεγκτή μάθαμε καταρχήν τη λογική του προγράμματος αυτού για το πώς και που μπορεί να χρησιμοποιηθεί, καθώς και το ρόλο που παίζει σε μία βιομηχανική μονάδα για το πώς μπορεί να βοηθήσει στην απλοποίηση πολλών εφαρμογών και ενεργειών, που πιθανόν να χρειαζόταν χωρίς αυτό, αλλά και στην ταχύτητα παραγωγής πολλές φορές η οποία παίζει πολύ σημαντικό ρόλο στην βιομηχανία.

7. ΕΓΧΕΙΡΙΔΙΟ

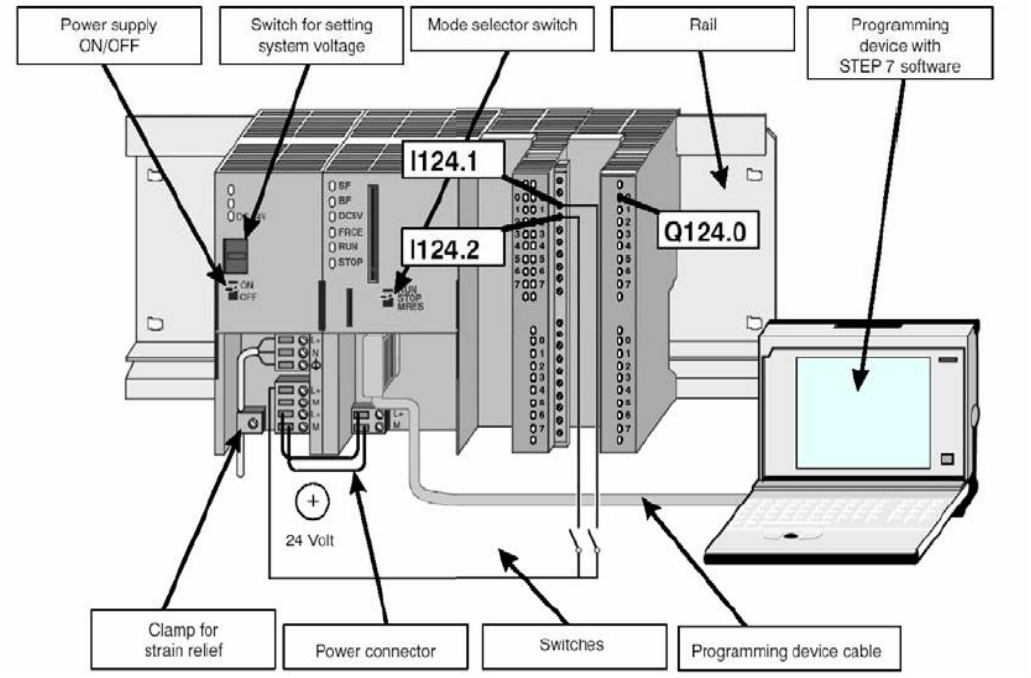
Εγχειρίδιο προγραμματισμού και λειτουργίας PLC

Σε αυτό το μέρος του εγχειριδίου αυτού αναφέρεται αναλυτικότερα η λειτουργία και ο προγραμματισμός των PLC.

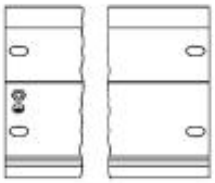
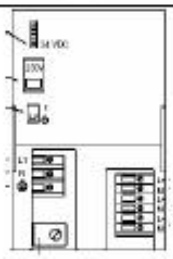
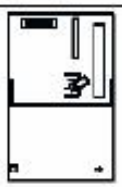
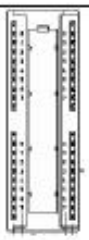
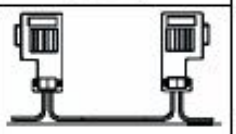
Το simatic manager είναι το κυριότερο εργαλείο στο step 7 αφού είναι αυτό που διαχειρίζεται τα project. Αφού το ανοίξουμε δημιουργούμε ένα νέο project, και εφόσον τελειώσουμε την διαδικασία προσαρμογής συνεχίζουμε κάνοντας διαμόρφωση του hardware. Εκεί επιλέγουμε τα υλικά του αυτοματισμού και τα εισάγουμε στην ράγα στήριξης Rail. Έχοντας τελειώσει με την hardware configuration επανερχόμαστε στο Simatic manager και στο project που δημιουργήσαμε. Αφού διαμορφώσαμε το hardware το Simatic Manager δημιουργεί τον φάκελο system data και στην συνέχεια δημιουργούμε το πρώτο μας μπλοκ οργάνωσης OB1. καθώς το επεξεργαζόμαστε μας δίνεται η δυνατότητα να επιλέξουμε γλώσσα προγραμματισμού επιλέγοντας μία από τις τρείς παρακάτω:LAD, STL, FBD.

Βασική δομή των PLC

Κάθε PLC μπορεί να δομηθεί από επιμέρους μονάδες ανάλογα με την εφαρμογή για την οποία θα χρησιμοποιηθεί. Στο παρακάτω σχήμα φαίνονται τα βασικά στοιχεία μιας απλής εφαρμογής.



Τα σημαντικότερα στοιχεία μιας εφαρμογής με PLC της σειράς S7-300 δίνονται στον παρακάτω πίνακα.

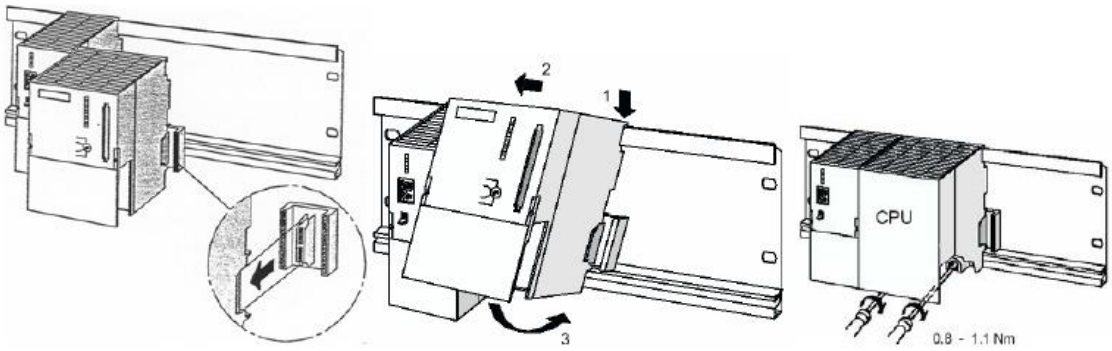
ΟΝΟΜΑΣΙΑ	ΛΕΙΤΟΥΡΓΙΑ	ΜΟΡΦΟΛΟΓΙΑ
Πλαίσιο στήριξης (Rack)	Ο ρόλος του είναι απλά να στηρίζει τις διάφορες κάρτες που θα συνθέσουν το σύστημα αυτοματισμού.	
Τροφοδοτικό PS (Power Supply)	Μετατρέπει την τάση του δικτύου τροφοδοσίας στην κατάλληλη τάση λειτουργίας του PLC	
Κεντρική μονάδα επεξεργασίας (Central Processing Unit)	Εκτελεί λειτουργικό πρόγραμμα του PLC και το πρόγραμμα του χρήστη. Ελέγχει τις επικοινωνίες σε ένα MPI δίκτυο.	
Κάρτες Εισόδων / Εξόδων Ψηφιακές - αναλογικές (Analog- Digital SM)	Προσαρμόζουν τα ηλεκτρικά σήματα από το εξωτερικό περιβάλλον προς την CPU και αντιστρόφως.	
Καλώδιο Profibus δικτύου με τους bus connector	Συνδέει μεταξύ τους κόμβους ενός MPI ή Profibus δικτύου.	
Καλώδιο σύνδεσης προγραμματιστή (PG cable)	Συνδέει τη CPU με την συσκευή προγραμματισμού PG (μπορεί ως προγραμματιστής να χρησιμοποιηθεί ένας H/Y με adaptor cable).	

Ας δούμε όμως αναλυτικά τα βασικά στοιχεία μιας S7-300 δομής.

ΠΛΑΙΣΙΟ ΣΤΗΡΙΞΗΣ (RACK)

Ο ρόλος του είναι να στηρίζει απλά τις διάφορες κάρτες που θα συνδέσουν το σύστημα αυτοματισμού. Πάνω σε κάθε rack πρέπει να τηρήσουμε μια ορισμένη σειρά στην σύνθεση του συστήματος μας. Στην πρώτη θέση του rack πρέπει να κουμπώσουμε την κάρτα του τροφοδοτικού, στην δεύτερη θέση πρέπει να τοποθετήσουμε την CPU, την τρίτη θέση είτε χρησιμοποιούμε είτε όχι κάρτα διασύνδεσης των rack (IM) πρέπει να την διαθέσουμε για αυτήν, από την τέταρτη θέση και πέρα πάνω στο rack συνδέω τα υπόλοιπα στοιχεία. Αυτά ισχύουν για το αρχικό rack (rack 0), Στα rack επέκτασης ξεκινάμε από την θέση 3 η οποία είναι αφιερωμένη για την κάρτα διασύνδεσης και πέρα. Κάθε rack εκτός από τα σταθερά που έχει (τροφοδοτικό, CPU, κάρτα διασύνδεσης) μπορεί να πάρει άλλες οκτώ κάρτες. Σ' ένα σύστημα με υλικό της σειράς S7 – 300 μπορούμε συνολικά να έχουμε έως τέσσερα πλαίσια στήριξης (rack).

Στήριξη Καρτών στο Rack:



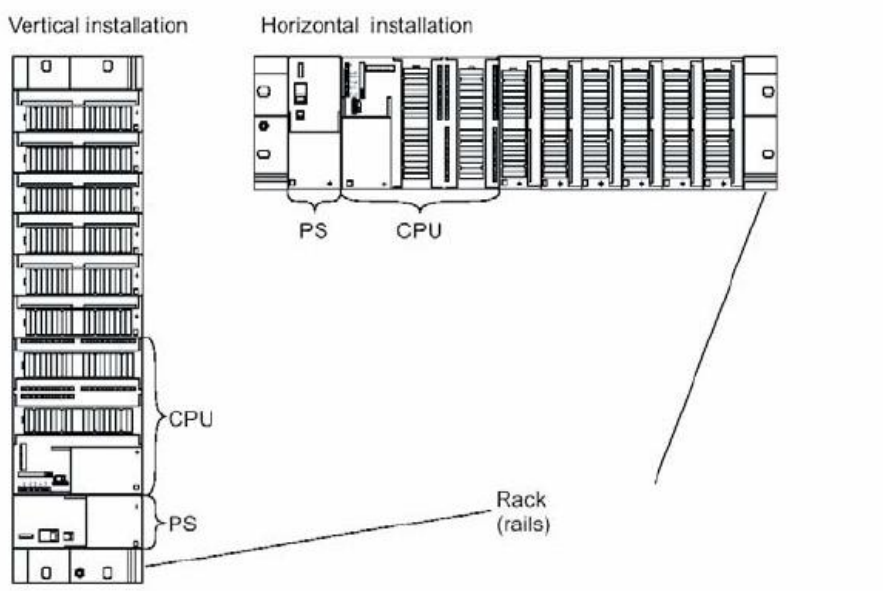
Στην σειρά S7 – 300 το rack χρησιμεύει μόνο για την στήριξη των υλικών που συνθέτουν το σύστημα. Η επικοινωνία μεταξύ καρτών και CPU γίνεται με έναν συνδετήρα σχήματος «Π» στο πίσω μέρος των καρτών. Μέσω αυτού υλοποιούνται δύο δίαυλοι εσωτερικής επικοινωνίας:

P – Bus (Peripheral Bus):Αυτό έχει σαν κύριο στόχο να μεταφέρει πληροφορίες που αφορούν την «περιφέρεια» (επικοινωνία με κάρτες εισόδου ή εξόδου)με ταχύτητα 1,5 Mbps

K – Bus (Communication Bus):Αφορά την επικοινωνία με τις λεγόμενες «ειδικές» κάρτες (κάρτες απαρίθμησης, PID,FM, CP …). Και στο K – Bus η πληροφορία μεταφέρεται σειριακή με ταχύτητα 187,5 Kbps.

Εγκατάσταση

Ένα σύστημα της σειράς S7-300 μπορεί να τοποθετηθεί οριζόντια ή κάθετα όπως δείχνει η επόμενη εικόνα. Σε κάθε περίπτωση το τροφοδοτικό και η CPU ή θα βρίσκονται αριστερά του συστήματος ή κάτω.

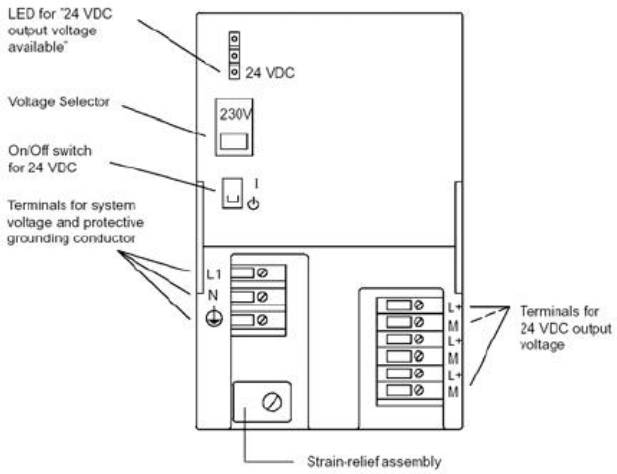


Στην επόμενη εικόνα παρουσιάζεται η μέγιστη δυνατή σύνθεση ενός συστήματος S7_300.

ΤΡΟΦΟΔΟΤΙΚΟ PS (Power Supply)

Ο ρόλος του είναι να δημιουργήσει τις αναγκαίες τάσεις που χρειάζεται το PLC για την τροφοδοσία του. Το ονομαστικό ρεύμα εξόδου του τροφοδοτικού πρέπει να είναι πάντα μεγαλύτερο από το ρεύμα που απορροφούν όλες οι κάρτες που είναι τοποθετημένες στο rack. Γα την σειρά S7 – 300 έχουμε τις εξής επιλογές:

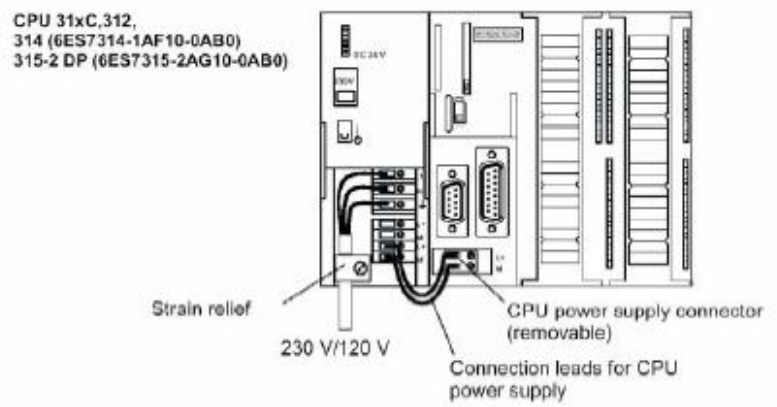
Στην κατασκευή μας, χρησιμοποιούμε ένα τροφοδοτικό PS 307: 5A. Μορφολογικά αυτό παρουσιάζεται στην επόμενη εικόνα



Διαθέτει:

- Κλέμες για τάση τροφοδοσίας (L1, N) και γείωση.
- Κλέμες για τάση εξόδου 24 V (L+, M)
- Διακόπτης ON – OFF
- Επιλογικό διακόπτη τάσης τροφοδοσίας (230 VAC ή 120 VAC)
- Ενδεικτικά LED ύπαρξης τάσεως εξόδου 24 VDC.

Στην επόμενη εικόνα παρουσιάζεται ο τρόπος καλωδίωσης μεταξύ τροφοδοτικού και CPU.



ΚΕΝΤΡΙΚΗ ΜΟΝΑΔΑ ΕΠΕΞΕΡΓΑΣΙΑΣ - CPU

Η κεντρική μονάδα επεξεργασίας η οποία συνηθίζεται να συμβολίζεται με CPU (Central Processing Unit) είναι ταυτόχρονα ο εγκέφαλος και η κινητήριος δύναμη ενός PLC. Η κεντρική μονάδα επεξεργασίας πραγματοποιεί πολλαπλές βασικές λειτουργίες:

- Διάβασμα, ερμηνεία και εκτέλεση, με τη σωστή διαδοχή, των οδηγιών, που περιέχονται στην μνήμη.
- Έλεγχο του πρωτοκόλλου επικοινωνίας που έχουμε καθορίσει στο σύστημά μας.
- Αποθήκευση των πληροφοριών
- Εκτέλεση αριθμητικών πράξεων

Κατά μια άποψη εάν συγκρίνουμε την CPU με την καλωδιωμένη λογική, τότε η CPU είναι το στοιχείο εκείνο το οποίο πραγματοποιεί τις καλωδιώσεις οι οποίες ζητούνται από τον κύκλο εργασίας της μηχανής ή της εγκατάστασης. Σε αντίθεση όμως από την καλωδιωμένη λογική της οποίας η λειτουργία είναι «παράλληλη», **το PLC εκτελεί τις λειτουργίες του με «σειριακό» τρόπο, για τον λόγο αυτό στα PLC είναι χαρακτηριστική η ταχύτητα λειτουργίας των κυκλωμάτων.**

Εσωτερικά για CPU περιέχει:

α) ΤΟΝ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΗ

Αυτός εκτελεί τις εντολές των προγραμμάτων που έχει αποθηκευμένες η μνήμη, καθορίζει την σειρά εκτέλεσης των λειτουργιών του συστήματος και ελέγχει για τυχόν σφάλματα.

β) Η ΜΝΗΜΗ

Η μνήμη μιας CPU χωρίζεται σε τρεις κατηγορίες.

1. **Μνήμη φόρτωσης (Load Memory)**
2. **Μνήμη εργασίας (Work memory)**
3. **Μνήμη συστήματος (System memory)**

Οι περιοχές (ομάδες) που χωρίζεται η μνήμη συστήματος είναι:

- **Μνήμη απεικόνισης εισόδων PII**

Σ΄ αυτήν την περιοχή αποθηκεύονται οι τιμές των εισόδων που διαβάζει η CPU από τις κάρτες εισόδου στην αρχή κάθε κύκλου λειτουργίας.

- **Μνήμη απεικόνισης εξόδων PIQ**

Σ' αυτήν την περιοχή αποθηκεύεται η τιμή κάθε μια από τις χρησιμοποιούμενες εξόδους κατά την 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 (Memory)**

Σ' αυτήν την περιοχή της μνήμης αποθηκεύονται ενδιάμεσα αποτέλεσμα τα οποία έχουν υπολογιστεί κατά την εκτέλεση του προγράμματος.

- **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 (Local Data)**

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

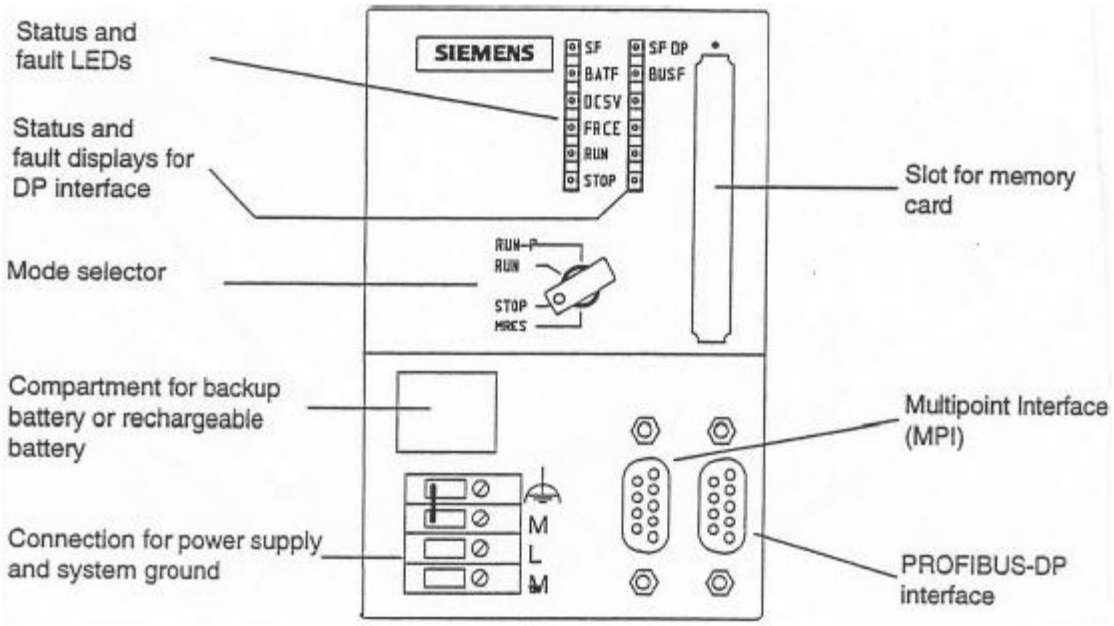
- **Διαγνωστικά (Diagnostics)**

Καταχωρούνται διάφορες ενέργειες που έχουν γίνει στο σύστημα με ώρα και ημερομηνία όπως CPU σε RUN/STOP, βραχυκυκλωμένη κάρτα αναλογικών,…

Εξωτερικά μια CPU παρουσιάζει:

1. Ακροδέκτες τροφοδοσίας
2. Θέση για μπαταρία (οι CPU που χρησιμοποιούν CF cards δεν έχουν).
3. Διακόπτη με κλειδί RUN – P/RUN/STOP/MRES
4. Ενδεικτικά LED για την κατάσταση της CPU

5. Θέση σύνδεσης PROFIBUS δικτύου.
6. Θέση για τοποθέτηση εξωτερικής μνήμης
7. Θέση σύνδεσης συσκευής προγραμματισμού ή MPI δικτύου
8. Ενδεικτικά LED για την κατάσταση του PROFIBUS δικτύου

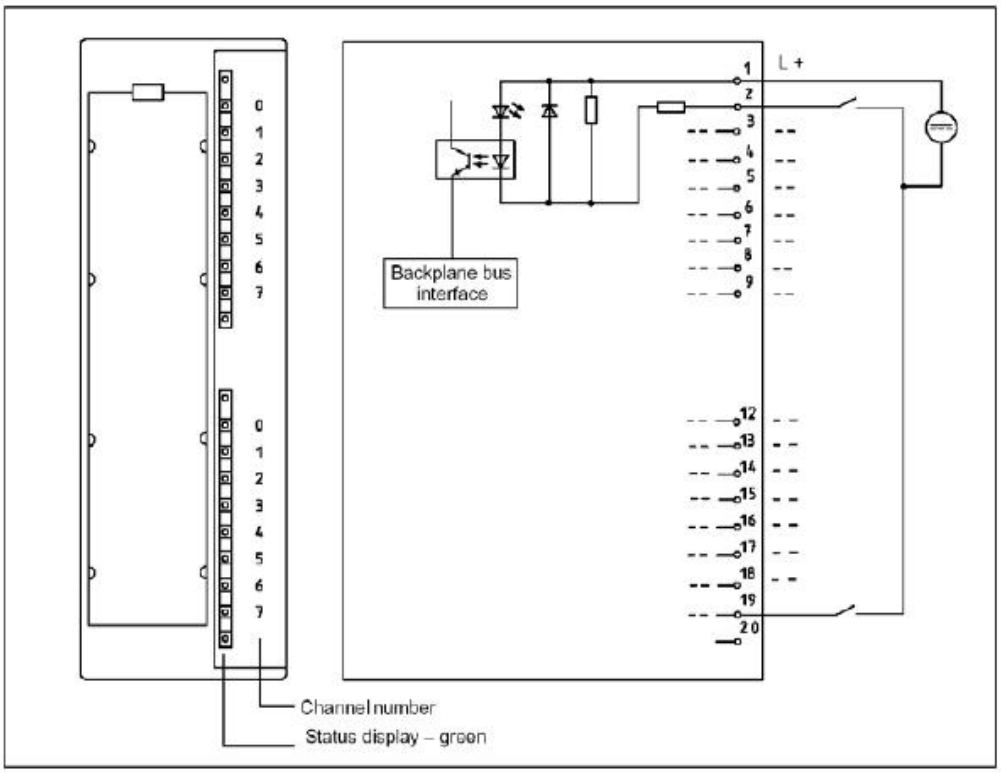


Στην οικογένεια S7 – 300 υπάρχει μια μεγάλη γκάμα από διαφορετικές CPU στην διάθεση του χρήστη. Διαφέρουν κυρίως ως προς το:

- Εάν έχουν ή όχι ενσωματωμένες εισόδους / εξόδους.
- Εάν έχουν ή όχι ενσωματωμένο profibus DP intarface.
- Μέγεθος της ενσωματωμένης μνήμης RAM.
- Πλήθος των εισόδων / εξόδων που υποστηρίζουν.

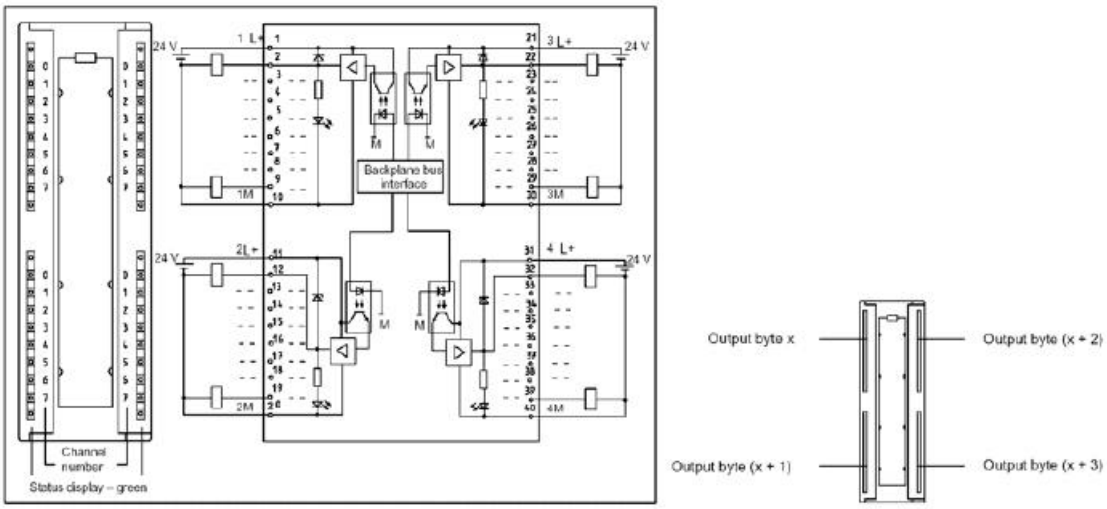
ΨΗΦΙΑΚΕΣ ΜΟΝΑΔΕΣ ΕΙΣΟΔΩΝ DI (Digital Input)

Η χρήση των μονάδων ψηφιακών εισόδων έχει τον σκοπό να μεταφέρει στην CPU τις καταστάσεις των διαφόρων αισθητηρίων ή διακοπτών ελέγχου που χρησιμοποιούμε στην εγκατάσταση.
Μια μονάδα εισόδων έχει 8,16 ή 32 εισόδους ανάλογα με τον τύπο και τάση που χρησιμοποιεί. Οι περισσότερο συνηθισμένες τάσεις για τα σήματα εισόδου είναι 24 VDC ή 230 VAC.
Στα όρια μιας κάρτας πρέπει να χρησιμοποιείται η ίδια τάση, στα όρια όμως όλου του συστήματος μπορούμε να χρησιμοποιήσουμε μονάδες ψηφιακών εισόδων με διαφορετικές τάσεις.
Μια κάρτα ψηφιακών εισόδων των 24 VDC αναγνωρίζει σαν σήμα «+1» τα +24 VDC και σαν σήμα «0» τα 0 V. Στις περιπτώσεις εκείνες που υπάρχει διακύμανση στην τάση (μη σταθεροποιημένο τροφοδοτικό) οι ψηφιακές κάρτες εισόδων έχουν ανοχές. Έτσι σαν σήμα «+1» καταλαβαίνει τις τάσεις από +13 ÷ +30 VDC και σαν σήμα «0» τις τάσεις από –3 ÷ + 5 VDC. Για τις ενδιάμεσες τιμές u964 τάσεων δηλαδή από +6 ÷ +12 VDC δεν είναι δυνατόν να προκαθοριστεί για το πώς θα τις κατανοήσει το PLC. Στην κάτω εικόνα παρουσιάζεται η μορφολογία και η αρχή λειτουργίας μιας ψηφιακής κάρτας εισόδων.



ΨΗΦΙΑΚΕΣ ΜΟΝΑΔΕΣ ΕΞΟΔΩΝ DO (DIGITAL OUTPUT)

Ο ρόλος τους είναι να μετατρέπουν τις αποφάσεις που παίρνει η CPU σε εντολές προς την εγκατάσταση.
Οι αποφάσεις αυτές βρίσκονται καταχωρημένες στην μνήμη απεικόνισης των εξόδων στην CPU και μετατρέπονται σε ηλεκτρικά σήματα από τις κάρτες εξόδων. Οι κάρτες εξόδων λειτουργούν σαν διακόπτες, στους οποίους δίνουμε εμείς την τάση (εξωτερικά) και όταν κλείσει ο διακόπτης η τάση περνάει και πηγαίνει προς το υπόλοιπο κύκλωμα.
Σε αντιστοιχία με τις κάρτες εισόδου το πρώτο χαρακτηριστικό που πρέπει να λάβουμε υπ' όψη μας είναι η τάση και το ρεύμα εξόδου της κάρτας, αυτά θα πρέπει να συμφωνούν με τα αντίστοιχα του φορτίου (π.χ. ρελέ) που θα συνδέσουμε σε κάθε ψηφιακή έξοδο. Μια κάρτα ψηφιακών εξόδων έχει 8, 16, ή 32 εξόδους ανάλογα με τον τύπο και την τάση που έχουν. Στα όρια μιας κάρτας χρησιμοποιείται πάντοτε η ίδια τάση. Στην κάτω εικόνα παρουσιάζεται η μορφολογία και η αρχή λειτουργίας μιας ψηφιακής κάρτας εξόδων.

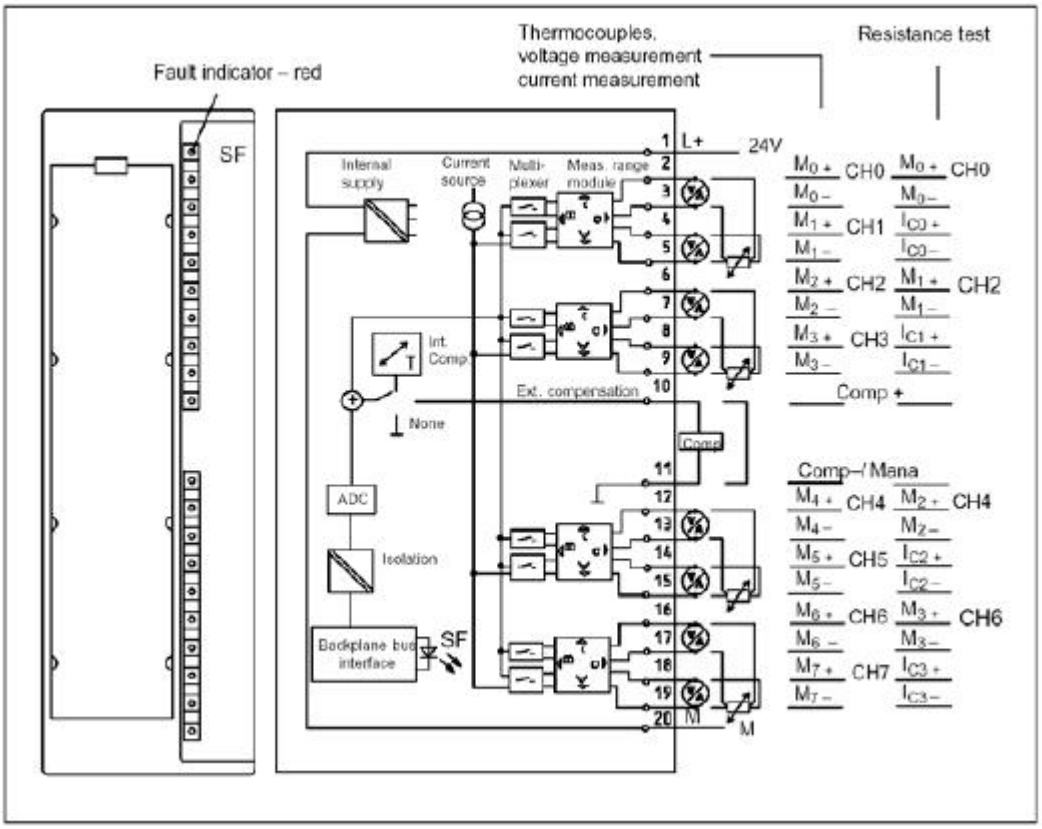


Ένα επί πλέον ιδιαίτερο χαρακτηριστικό των DO είναι το στοιχείο εξόδου (αυτό που παρέχει την ισχύ στο φορτίο). Αυτό συνήθως είναι τρανζίστορ αν πρόκειται για DC κάρτα εξόδων ή triac ή ρελέ εάν πρόκειται για AC κάρτα εξόδου. Όλες οι ψηφιακές έξοδοι είναι γαλβανικά απομονωμένες.

ΜΟΝΑΔΕΣ ΑΝΑΛΟΓΙΚΩΝ ΕΙΣΟΔΩΝ AI (ANALOG INPUT)

Για να επεξεργαστούμε ηλεκτρικά σήματα, με συνεχή μεταβολή της τιμής τους, στο PLC χρειαζόμαστε κάρτες αναλογικών σημάτων. Οι κάρτες αναλογικών εισόδων έχουν τον ρόλο να διαβάζουν ένα ηλεκτρικό μέγεθος και να το μετατρέπουν σε ένα αριθμό (δυαδική αναπαράσταση) το οποίο πλέον μπορεί η CPU να αναγνωρίσει και να επεξεργαστεί. Οι κάρτες αναλογικών εισόδων δέχονται ηλεκτρικά σήματα τάσης ή έντασης. Οι τυποποιημένες τιμές έντασης τις οποίες μπορεί να διαβάσει μια αναλογική κάρτα εισόδων είναι 0 – 20 mA ή 4 – 20 mA για δε τα σήματα τάσης έχουμε 0 ÷ 10 V ή ± 10 V. Ένα άλλο μέγεθος u960 που μας ενδιαφέρει στην επιλογή μιας κάρτας αναλογικών εισόδων είναι η διακριτική τους ικανότητα (ακρίβεια). Κάθε αναλογικό σήμα καταλαμβάνει χώρο 16 bit.

Στην κάτω εικόνα παρουσιάζεται η μορφολογία και η αρχή λειτουργίας μιας αναλογικής κάρτας εισόδων.

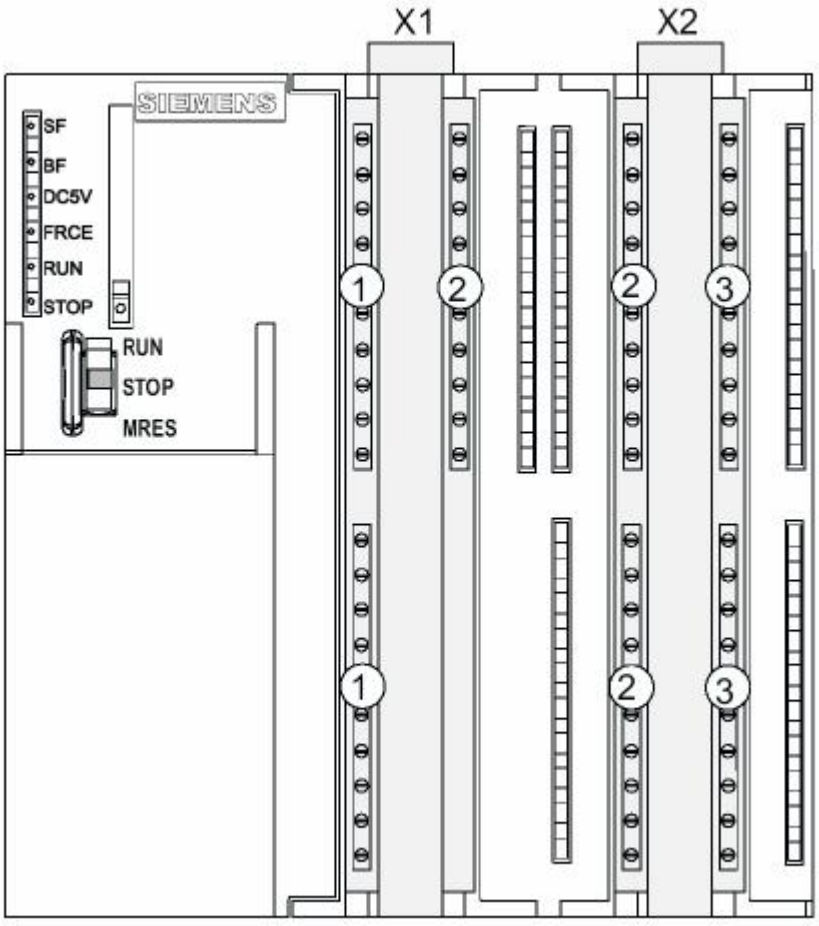


Ένα ακόμα μεγάλο πλεονέκτημα της σειράς S7 είναι ότι μια αναλογική κάρτα εισόδων μπορεί να γίνει τάσης ή έντασης και να μεταβάλουμε την περιοχή μέτρησης της επεμβαίνοντας τόσο εξωτερικά πάνω στην ίδια την κάρτα όσο και στο software.

ΜΟΝΑΔΕΣ ΑΝΑΛΟΓΙΚΩΝ ΕΞΟΔΩΝ Α/Ο (Analog Output)

Οι κάρτες αναλογικών εξόδων έχουν τον ρόλο να μετατρέψουν το αριθμητικό μέγεθος με το οποίο «σκέπτεται» η CPU στην κατάλληλη τιμή έντασης ή τάσης ώστε να μπορεί να οδηγηθεί το ανάλογο εξάρτημα που ελέγχει το φυσικό μέγεθος της εγκατάστασης μας.
Όλα τα χαρακτηριστικά των καρτών είναι σε πλήρη αντιστοιχία με αυτή των αναλογικών εισόδων μια και εκτελούν απλώς την αντίστροφη διαδικασία όποτε δεν απαιτείται κάποια ιδιαίτερη συζήτηση.

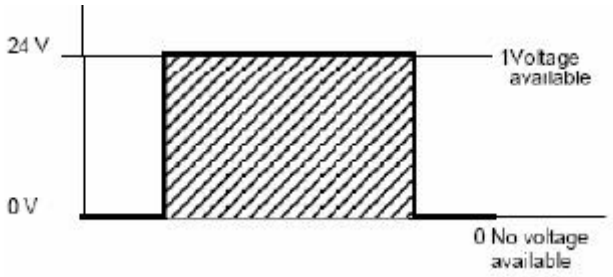
Το σύστημα αυτοματισμού στο οποίο βασίζεται η πτυχιακή εργασία διαθέτει την **CPU314C-2DP** η οποία διαθέτει ενσωματωμένες 24 DI, 16 DO, 4 + 1 AI, 2 AO και υποστηρίζει το πρωτόκολλα διασύνδεσης MPI και ProfiBus-DP(master-slave).



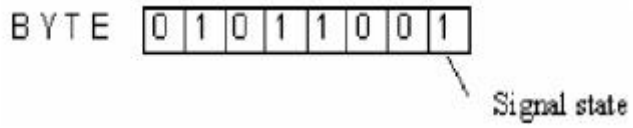
Διευθυνσιδότηση – Ονοματολογία

Έννοιες bit, byte, word, double word

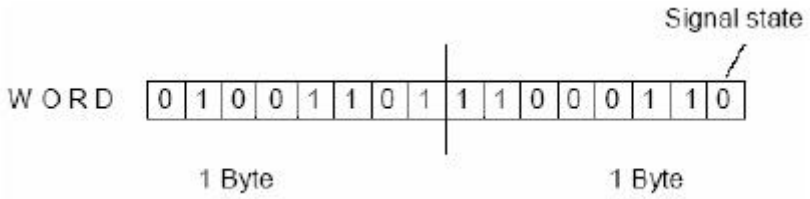
Bit: Το bit είναι η μικρότερη μονάδα αποθήκευσης της κατάστασης ενός ψηφιακού σήματος. Το bit είναι ο χώρος μιας κυψέλης μνήμης και μπορεί να πάρει δύο τιμές την κατάσταση «0» η οποία αντιστοιχεί στην μη ύπαρξη τάσης στο ψηφιακό σήμα και την κατάσταση «1» η οποία αντιστοιχεί στην ύπαρξη τάσης στο ψηφιακό σήμα.



Byte: Μια ομάδα από οκτώ συνεχόμενα bit ορίζει ένα byte.

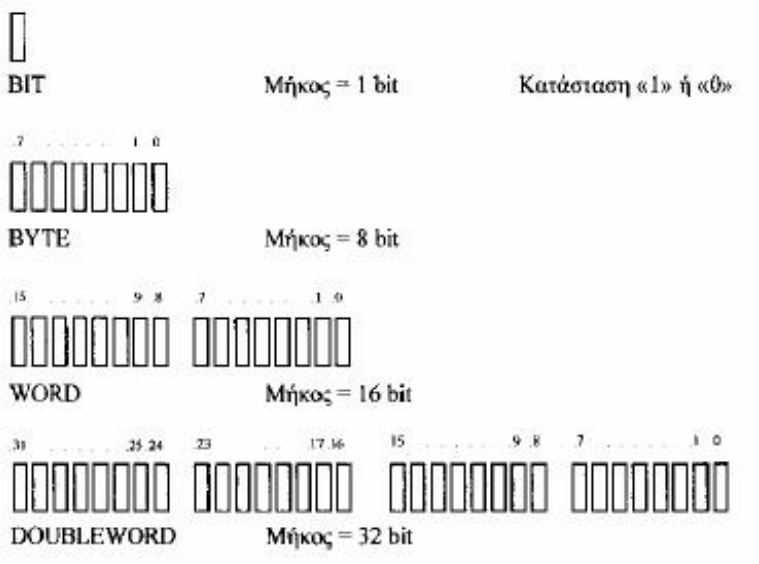


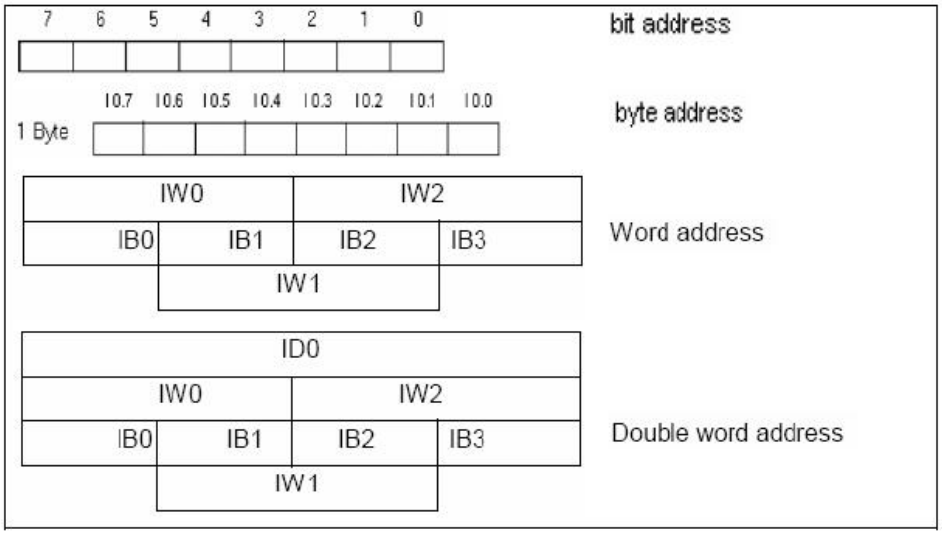
Word: Δύο συνεχόμενα byte ή 16-συνεχόμενα-bit ορίζουν μια word.



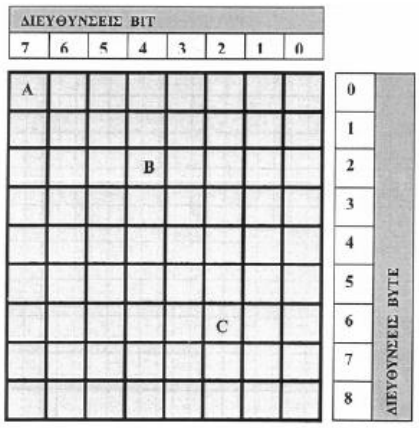
Double word: δύο συνεχόμενες word ή 4-συνεχόμενα-byte ή 32-συνεχόμενα-bit ορίζουν μια double word.

Το κάθε bit έχει μια συγκεκριμένη διεύθυνση (αριθμό). Η αρίθμηση γίνεται πάντοτε από τα δεξιά προς τα αριστερά ξεκινώντας από το bit 0 και φθάνοντας στο 7 (στα
byte) 15 (στις word) ή 31 (στις double word). Στην παρακάτω εικόνα δίνεται η γραφική αναπαράσταση των εννοιών bit, byte, word, double word.



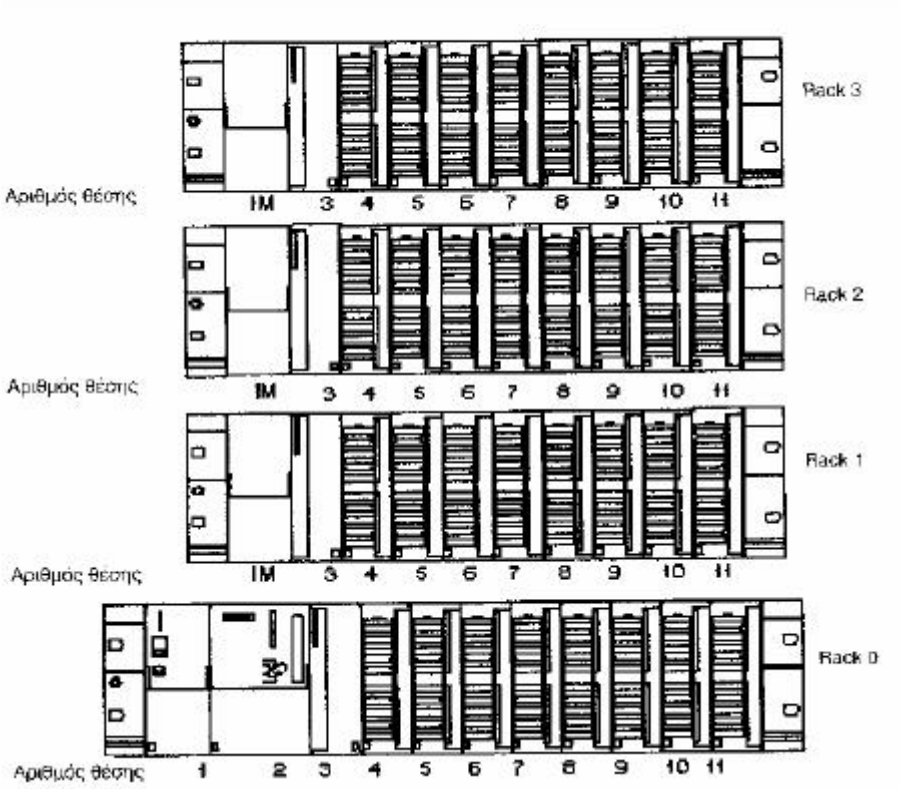


Εκείνο που πρέπει να έχουμε υπ' όψη μας είναι ότι οι μνήμες στα PLC της σειράς S7 είναι οργανωμένες σε byte (κάτω εικόνα).



Έτσι π.χ. η κυψέλη Α έχει διεύθυνση 0.7, η κυψέλη Β έχει διεύθυνση 2.4 και η κυψέλη C έχει διεύθυνση 6.2. Ο πρώτος αριθμός αναφέρεται στην διεύθυνση byte που ανήκει η κυψέλη ενώ ο δεύτερος αριθμός αναφέρεται στην θέση του bit μέσα σ' ένα byte. Όταν αναφερθήκαμε στην οργάνωση της μνήμης ενός PLC είδαμε ότι αυτή είναι χωρισμένη σε διάφορες περιοχές ανεξάρτητες μεταξύ τους π.χ. μνήμη απεικόνισης εισόδων, μνήμη απεικόνισης εξόδων, μνήμη χρονικών … Μεταξύ ανεξαρτήτων περιοχών μνήμης μπορούμε να έχουμε ίδιες διευθύνσεις χωρίς πρόβλημα π.χ. I 1.2. και Q 1.2, όταν όμως βρισκόμαστε στην ίδια περιοχή μνήμης θα πρέπει να είμαστε προσεκτικοί στον ορισμό των διευθύνσεων για να μην έχουμε επικαλύψεις π.χ. η W0 έχει επικάλυψη με τα M0.0 έως M1.7. Όταν έχουμε επικαλύψεις τότε μας δημιουργείται πρόβλημα στο περιεχόμενο μιας κυψέλης.

Διεύθυνση θέσης: κάθε μονάδα η οποία ανήκει σ' ένα σύστημα αυτοματισμού με PLC της σειράς S7 έχει μια διεύθυνση θέσης. Αυτή αποτελείται από τον αριθμό του rack που είναι τοποθετημένη η μονάδα και τον αριθμό της θέσης της. Έτσι σε ένα πλήρες σύστημα (βασικό rack + 3 rack επέκτασης) με PLC της σειράς S7-300 οι διευθύνσεις θέσης που μπορεί να έχουμε για τις διάφορες μονάδες που το απαρτίζουν δίνονται στην επόμενη εικόνα.



Πρέπει να έχουμε υπ' όψιν μας ότι στο αρχικό rack (rack 0) την πρώτη θέση πάντα την έχει το τροφοδοτικό την δεύτερη θέση η CPU, η τρίτη θέση ανήκει στην κάρτα διασύνδεσης του Rack (IM). Εάν έχουμε τέτοια κάρτα χρησιμοποιούμε την τρίτη θέση εάν το σύστημα μας δεν διαθέτει τέτοια κάρτα η θέση 3 παραμένει υποχρεωτικά κενή.

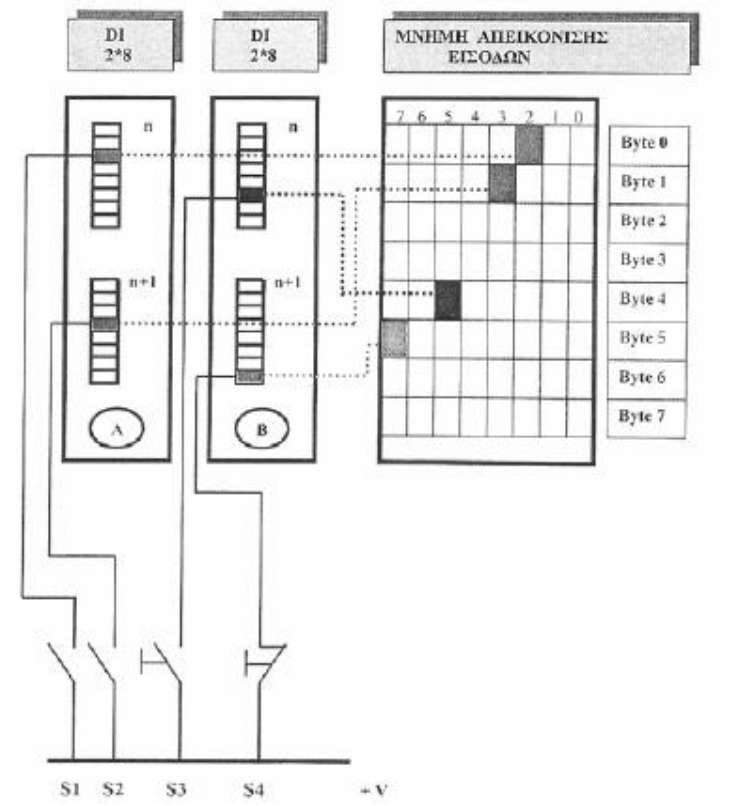
Λογική διεύθυνση: πέρα από την διεύθυνση θέσης, κάθε μονάδα έχει μια αρχική διεύθυνση (διεύθυνση του πρώτου byte που καταλαμβάνει στον χώρο μνήμης που ανήκει) η οποία καθορίζει τη θέση της στο χώρο των λογικών διευθύνσεων. Ο χώρος των λογικών διευθύνσεων ξεκινάει από την διεύθυνση 0 και τελειώνει σε διεύθυνση που εξαρτάται από την χρησιμοποιούμενη CPU. Η λογική διεύθυνση σε ένα σύστημα εξαρτάται από την θέση που βρίσκεται η κάρτα σε σχέση με την CPU και από το εάν είναι ψηφιακή ή αναλογική. Ανάλογα λοιπόν με την θέση και το είδος της κάρτας ισχύει ο παρακάτω πίνακας.

Rack	Αρχική διεύθυνση	Αριθμός θέσης										
	της μονάδας	1	2	3	4	5	6	7	8	9	10	11
0	Ψηφιακά	PS	CPU	IM	0	4	8	12	16	20	24	28
	Αναλογικά				256	272	288	304	320	336	352	368
1[1]	Ψηφιακά	–		IM	32	36	40	44	48	52	56	60
	Αναλογικά	–			384	400	416	432	448	464	480	496
2[1]	Ψηφιακά	–		IM	64	68	72	76	80	84	88	92
	Αναλογικά	–			512	528	544	560	576	592	608	624
3[1]	Ψηφιακά	–		IM	96	100	104	108	112	116	120	124[2]
	Αναλογικά	–			640	656	672	688	704	720	735	752[2]

1 - όχι με τις CPU312IFM/313
2 - όχι με τη CPU314IFM

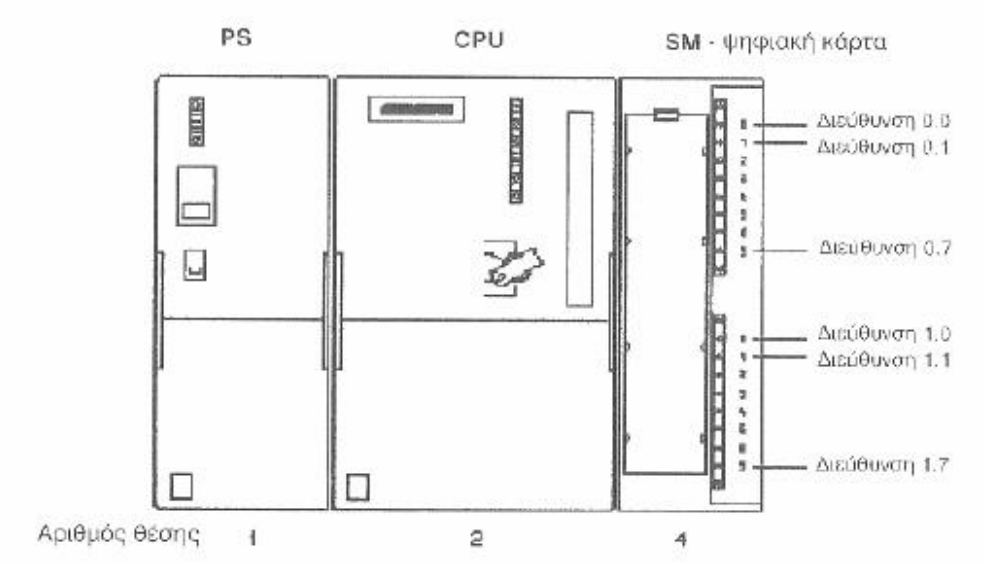
Σαν παράδειγμα διευθυνσιοδότησης ας πάρουμε την περίπτωση δύο ψηφιακών καρτών εισόδων των 16 θέσεων την κάρτα Α και την κάρτα Β. Ας υποθέσουμε u972 ότι η κάρτα Α έχει διεύθυνση θέσης 4 πάνω στο rack και η κάρτα Β διεύθυνση θέσης 5. Οι αρχικές λογικές διευθύνσεις των καρτών έστω ότι έχουν ορισθεί για την κάρτα Α «0» και για την κάρτα Β «4». Στην κάρτα Α έχουμε συνδέσει τους διακόπτες S1 και S2 στην κάρτα Β έχουμε συνδέσει τα μπουτόν S3 και S4.

Έχουμε λοιπόν: **S1 = I0.2** S2=I1.3 **S3=I4.5** και S4=I5.7

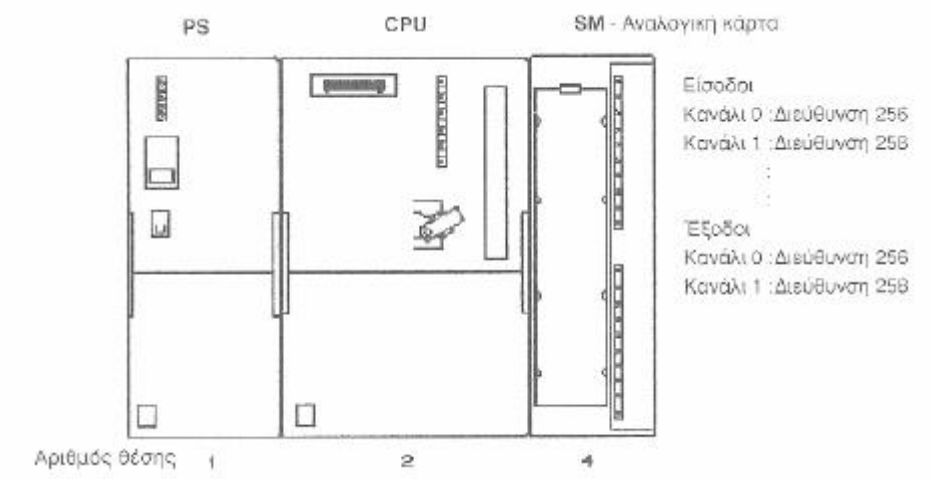


Έχουμε λοιπόν: **S1 = I0.2** S2=I1.3 **S3=I4.5** και S4=I5.7

Στην κάτω εικόνα παρουσιάζεται η διευθυνσιοδότηση μιας ψηφιακής κάρτας εισόδου.



Αντίστοιχα παρουσιάζεται η διευθυνσιοδότηση μιας αναλογικής κάρτας. Παρατηρούμε ότι έχουμε ίδιες διευθύνσεις, χωρίς να παρουσιάζεται πρόβλημα, μεταξύ εισόδων – εξόδων.



ΟΝΟΜΑΤΟΛΟΓΙΑ

Για να ορίσουμε μια παράμετρο σ' ένα σύστημα αυτοματισμού με PLC χρησιμοποιούμε ένα συνδυασμό γραμμάτων και αριθμών. Τα μεν γράμματα είναι τα διευκρινιστικά εκείνα στοιχεία που κατατάσσουν την παράμετρο σε μια ομάδα (π.χ. είσοδοι, έξοδοι, εσωτερικά, βοηθητικά, …) οι δε αριθμοί είναι τα στοιχεία εκείνα τα οποία ορίζουν την διεύθυνση μιας συγκεκριμένης παραμέτρου. Για την σειρά S7 και στην αγγλική γλώσσα χρησιμοποιείται η εξής ονοματολογία.

ΕΙΣΟΔΟΙ I (Input)

Μια ψηφιακή είσοδος συμβολίζεται με το γράμμα I και η ονοματολογία της έχει τη μορφή.

Ix.y όπου x: Διεύθυνση byte (0 …n)
y: Διεύθυνση bit (0 … 7)

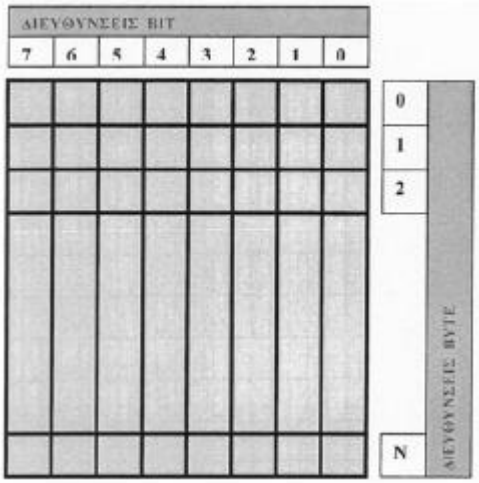
Έχουμε την δυνατότητα να παρουσιάσουμε ή να ζητήσουμε

Byte εισόδων: IBX
Παράδειγμα: IB3 με αυτήν την ονοματολογία δηλώνουμε τις εισόδους I3.0 …I3.7

Word εισόδων: IWX
Παράδειγμα: IW2 με αυτήν την ονοματολογία δηλώνουμε τις εισόδους I2.0 … I2.7, I3.0, I3.1, …I3.7

Double word εισόδων: IDX
Παράδειγμα: ID4 με αυτήν την ονοματολογία δηλώνουμε τις εισόδους I4.0 ….. I4.7, I5.0 … I5.7 , I6.0 … I6.7, I7.0 … I7.7

ΠΕΡΙΟΧΗ ΑΠΕΙΚΟΝΙΣΗΣ ΤΗΣ ΜΝΗΜΗΣ ΕΙΣΟΔΩΝ PII

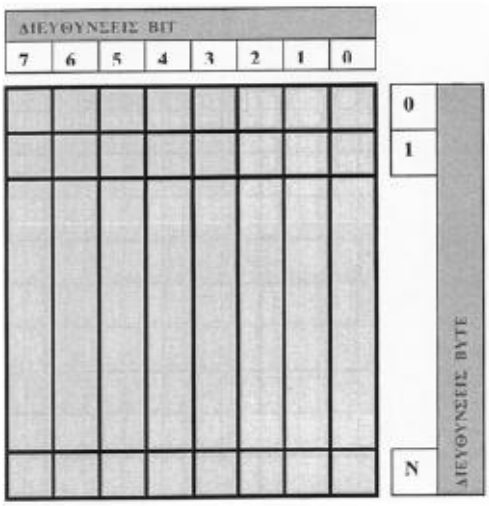


ΈΞΟΔΟΙ Q (Output)
Μια ψηφιακή έξοδος συμβολίζεται με το γράμμα Q και η ονοματολογία της έχει τη μορφή.

Q x.y όπου x: διεύθυνση byte
y: διεύθυνση bit (0…7)

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

ΠΕΡΙΟΧΗ ΑΠΕΙΚΟΝΙΣΗΣ ΤΗΣ ΜΝΗΜΗΣ ΕΞΟΔΩΝ P I Q



ΒΟΗΘΗΤΙΚΑ Μ (Memory bit)

Τα βοηθητικά παίζουν τον ρόλο των βοηθητικών ρελέ στον κλασικό αυτοματισμό, τα χρησιμοποιούμε στο πρόγραμμα για να αποθηκεύσουμε λογικό αποτέλεσμα τμήματος του προγράμματος (ειδικά όταν αυτό είναι επαναλαμβανόμενο). Είναι ρελέ του οποίου το λογικό αποτέλεσμα δεν μπορώ να πάρω απ' ευθείας στην κάρτα εξόδου. Ένα βοηθητικό συμβολίζεται με το γράμμα Μ και η ονοματολογία του έχει τη μορφή

M x.y όπου x: διεύθυνση byte (0…N)
y: διεύθυνση bit (0 … 7) Και εδώ έχουμε **MBX, MWX, MDX**

ΧΡΟΝΙΚΑ Τ (Timers)

Η λειτουργία χρονικών χρησιμοποιείται για να υλοποιήσει αλγορίθμους που έχουν σχέση με χρόνο (επιτήρηση, αναμονή, μέτρηση χρονικών διαστήματος, δημιουργία παλμών). Με τον όρο «χρονικό» εννοούμε μια λέξη (word) σε μια ειδική περιοχή της μνήμης, αυτή των χρονικών. Τα χρονικά συμβολίζονται με το γράμμα Τ και η ονοματολογία του έχει τη μορφή : **Tx** όπου x: αριθμός του χρονικού (0… n)

ΑΠΑΡΙΘΜΗΤΕΣ C (counters)

Οι λειτουργίες απαριθμητή μας δίνουν τη δυνατότητα να εκτελούμε εργασίες απαρίθμησης απ' ευθείας από την CPU. Με τον όρο απαριθμητής εννοούμε μια λέξη (Word) σε μια ειδική περιοχή της μνήμης, αυτή των απαριθμητών. Ο απαριθμητής συμβολίζονται με το γράμμα C και
η ονοματολογία που έχει τη μορφή: **Cx,** όπου x : αριθμός του απαριθμητή (0… n)

ΔΟΜΗ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ

ΔΟΜΗ PROJECT

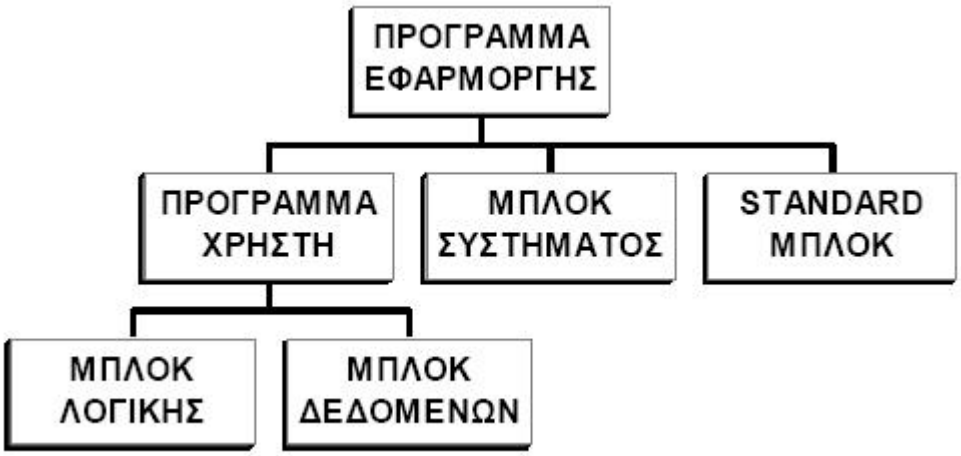
Κατά τη φάση του σχεδιασμού του project μας ένα από τα πρώτα πράγματα που πρέπει να κάνουμε είναι στο να αποφασίσουμε με ποιόν τρόπο θα δομήσουμε το πρόγραμμα μας δηλαδή στο τι μπλοκ θα περιέχει και πως θα συνδέονται μεταξύ τους αυτά τα μπλοκ. Ας δούμε όμως πρώτα πως είναι οργανωμένο ένα πρόγραμμα στην CPU. Κάθε CPU περιλαμβάνει δύο προγράμματα ανεξάρτητα το ένα από το άλλο:

α) ΛΕΙΤΟΥΡΓΙΚΟ ΣΥΣΤΗΜΑ

Το λειτουργικό σύστημα είναι το σύνολο των ορισμών και εντολών που ελέγχουν τους πόρους του συστήματος. Είναι αυτό που ενημερώνει το ρολόι του πραγματικού χρόνου στη CPU, που ελέγχει την κατάσταση του διακόπτη της CPU, (RUN, STOP, …), ελέγχει να ανάψει τα LED στη CPU, να ρυθμίσει τις επικοινωνίες μέσα απ το MPI interface, … Στο λειτουργικό σύστημα δεν μπορούμε u957 να κάνουμε μεταβολές, μπορούμε όμως να διαβάσουμε ή να χρησιμοποιήσουμε ορισμένα αποτελέσματα αυτού (π.χ. το ρολόι πραγματικού χρόνου).

β) ΠΡΟΓΡΑΜΜΑ ΕΦΑΡΜΟΓΗΣ

Το πρόγραμμα εφαρμογής είναι το σύνολο των εντολών και ορισμών που χρειάζεται το PLC για τον έλεγχο της εγκατάστασης. Η δομή ενός προγράμματος εφαρμογής δίνεται στην κάτω εικόνα.



ΠΡΟΓΡΑΜΜΑ ΧΡΗΣΤΗ: Είναι το πρόγραμμα που εμείς γράφουμε για τις λειτουργικές ανάγκες της εγκατάστασης και του αυτοματισμού. Αυτό μπορεί να περιέχει **μπλοκ λογικής** (εντολές) και **μπλοκ δεδομένων** (όπου καταχωρούνται λίστες με αριθμούς).

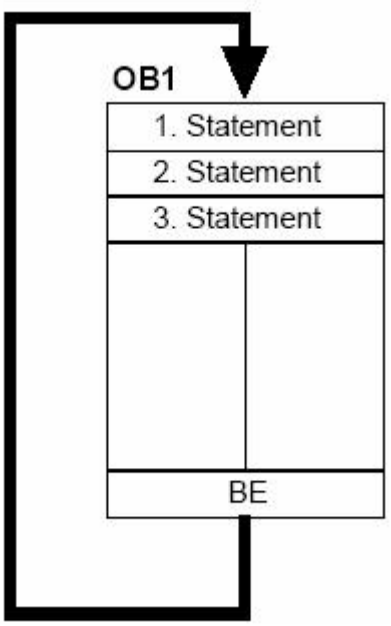
ΜΠΛΟΚ ΣΥΣΤΗΜΑΤΟΣ: Είναι λειτουργίες που είναι από πριν ορισμένες και καταχωρημένες στο λειτουργικό σύστημα του PLC. Στο πρόγραμμα του ο χρήστης καλεί αυτά τα μπλοκ σε οποιοδήποτε σημείο θέλει, τους δίνει κάποιες παραμέτρους και παίρνει μόνο τα αποτελέσματα, χωρίς να ενδιαφέρεται για το πώς έχουν αυτά παραχθεί.

STANDARD ΜΠΛΟΚ: Είναι μπλοκ που μας προσφέρουν έτοιμες λύσεις για τυποποιημένες εργασίες αυτοματισμού που πιθανόν να μας ενδιαφέρουν.

ΕΠΕΞΕΡΓΑΣΙΑ ΠΡΟΓΡΑΜΜΑΤΟΣ

Ανάλογα με τον τρόπο με τον οποίο χτίζουμε ένα πρόγραμμα (πρόγραμμα χρήστη) έχουμε τρία διαφορετικά είδη δόμησης.

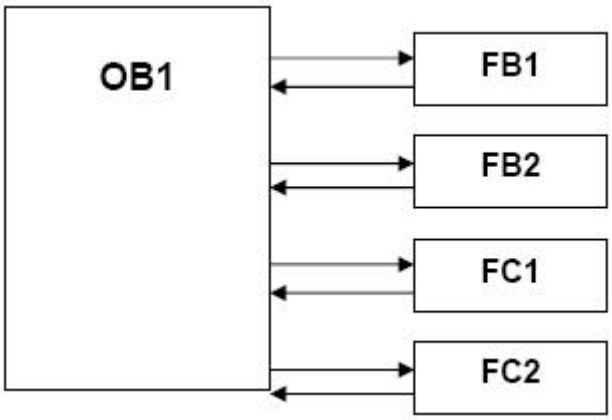
1. ΓΡΑΜΜΙΚΟ ΠΡΟΓΡΑΜΜΑ



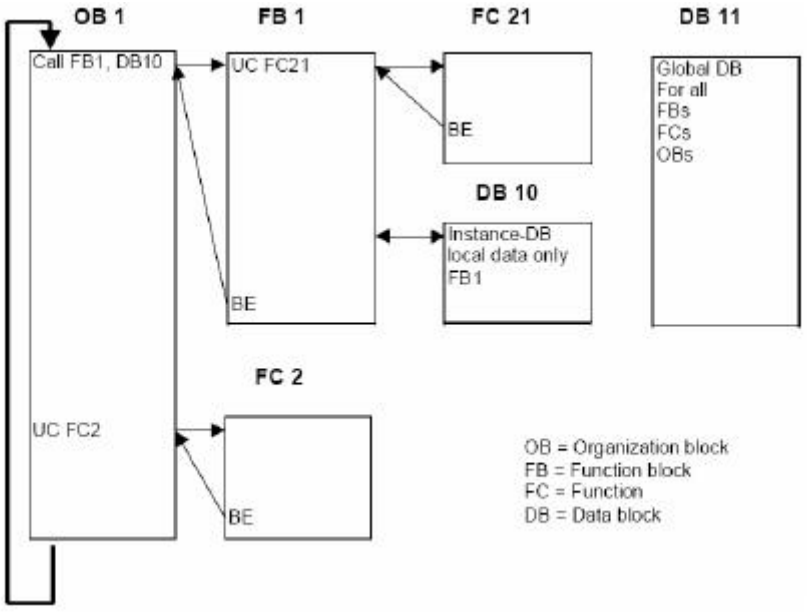
Όλο το πρόγραμμα του χρήστη βρίσκεται σ' ένα συνεχόμενο μπλοκ (OB1 που καλείται αυτόματα σε κάθε κύκλο λειτουργίας). Η CPU επεξεργάζεται τις εντολές την μια μετά την άλλη μέχρι το τέλος του μπλοκ και ξαναρχίζει η ίδια διαδικασία πάλι από την αρχή. Έχει το πλεονέκτημα ότι εύκολα και γρήγορα αρχίζει κάποιος τη φάση του προγραμματισμού. Έχει το μειονέκτημα ότι σε μεγάλα προγράμματα είναι δύσκολο να εντοπίσουμε που γίνεται μια συγκεκριμένη εργασία. Χρησιμοποιείται για μικρές εφαρμογές.

2. ΤΜΗΜΑΤΟΠΟΙΗΜΕΝΟ ΠΡΟΓΡΑΜΜΑ

Το πρόγραμμα χωρίζεται σε μπλοκ όπου κάθε ένα από αυτά υλοποιεί μια συγκεκριμένη εργασία. Για τον τρόπο κλήσης, την σωστή λειτουργία τους καθώς και την σωστή σειρά εκτέλεσης τους φροντίζει ένα ειδικό u956 μπλοκ το οποίο λέγεται μπλοκ οργάνωσης (OB1).



3. ΔΟΜΗΜΕΝΟ ΠΡΟΓΡΑΜΜΑ



Ένα δομημένο πρόγραμμα μπορεί να περιλαμβάνει παραμετροποιημένα μπλοκ. Αυτά τα μπλοκ είναι έτσι σχεδιασμένα ώστε να μπορούν να είναι γενικής χρήσης. Όταν καλείται ένα τέτοιο μπλοκ του δίνουμε τιμές στις παραμέτρους για την διαδικασία που μας ενδιαφέρει (διευθύνσεις εισόδων, εξόδων, χρονικά). Ο δομημένος προγραμματισμός μας προσφέρει πολλά πλεονεκτήματα, όπως :

- Εξοικονόμηση μνήμης (δεν επαναλαμβάνουμε το γράψιμο ίδιων προγραμμάτων)

- Οποιαδήποτε αλλαγή στη λογική του αυτοματισμού την περνάμε μια φορά στο πρόγραμμα και αυτόματα γίνεται η διόρθωση της λειτουργίας όπου χρειάζεται (εξοικονόμηση χρόνου και ελαχιστοποίηση της πιθανότητας σφάλματος από λανθασμένη πληκτρολόγηση).

ΤΥΠΟΙ ΤΩΝ ΔΙΑΘΕΣΙΜΩΝ ΜΠΛΟΚ

Για το χτίσιμο της εφαρμογής μας έχουμε στην διάθεση μας διαφορετικά είδη μπλοκ προγραμματισμού. Το τι θα χρησιμοποιήσουμε και πως θα τα διασύνδεουμε είναι τις περισσότερες φορές υποκειμενική υπόθεση και εξαρτάται από την εφαρμογή που έχουμε να προγραμματίσουμε.

Οι διάφοροι τύποι των διαθέσιμων μπλοκ είναι:

- **ΜΠΛΟΚ ΟΡΓΑΝΩΣΗΣ OB (Organization Blocks).**

Έχουν τον ρόλο του διαμεσολαβητή μεταξύ του λειτουργικού συστήματος και του προγράμματος του χρήστη. Κατά την εκδήλωση κάποιων ειδικών γεγονότων, όπως για παράδειγμα μιας χρονικής διακοπής, μιας διακοπής τροφοδοσίας, …, το λειτουργικό σύστημα της CPU καλεί το αντίστοιχο μπλοκ οργάνωσης. Ένα από τα διάφορα μπλοκ οργάνωσης, σημαντικότερο απ' όλα είναι το OB1. Αυτό είναι ένα μπλοκ το οποίο η CPU καλεί αυτόματα και το εκτελεί συνεχώς κυκλικά. Μέσα σ' αυτό το μπλοκ βρίσκεται το κύριο πρόγραμμα του χρήστη.

Άλλο σημαντικό μπλοκ είναι το OB100 που εκτελείται μία φορά όταν δίνουμε τάση στο σύστημα.

Τα μπλοκ οργάνωσης έχουν τάξεις προτεραιότητας από 0 ως 29. Αν ένα μπλοκ έχει μεγαλύτερη προτεραιότητα από κάποιο άλλο, τότε μπορεί να το διακόψει και να εκτελεστεί το ίδιο. Π.χ. το OB1 που έχει προτεραιότητα 1 μπορεί να διακοπεί από όλα τα άλλα μπλοκ.

- **ΣΥΝΑΡΤΗΣΕΙΣ FC (Functions)**

Οι συναρτήσεις είναι μπλοκ τα οποία προγραμματίζονται από τον χρήστη. Τα FC είναι μπλοκ κώδικα «στερούμενο μνήμης». Οι προσωρινές μεταβλητές (temporary variables) των FC αποθηκεύονται στην περιοχή των τοπικών δεδομένων (local data stack). Μετά την επεξεργασία των FC αυτά τα δεδομένα χάνονται. Για την αποθήκευση των δεδομένων τα FC μπορούν να χρησιμοποιήσουν DB (shared data blocks).

Ένα FC περιέχει ένα πρόγραμμα το οποίο εκτελείται όταν το FC καλείται από ένα άλλο μπλοκ που περιέχει κώδικα.

Τα FC χρησιμοποιούνται για:

- Έλεγχο μιας τεχνολογικής συνάρτησης (π.χ. έλεγχος ανεξάρτητων τμημάτων εγκατάστασης.
- Συχνά επαναλαμβανόμενες λειτουργίες αυτοματισμού
- Υπολογισμό κάποιας συνάρτησης και απόδοσης τιμής στο μπλοκ που το έχει καλέσει (π.χ. υπολογισμός μαθηματικών συναρτήσεων).

Τα FC παραμετροποιούνται και επομένως μπορούν να χρησιμοποιηθούν για περιπτώσεις στις οποίες έχουμε επαναλαμβανόμενη λογική στο πρόγραμμα μας με διαφορετικές παραμέτρους.

- **ΜΠΛΟΚ ΣΥΝΑΡΤΗΣΕΩΝ FB (Function Block)**

Τα μπλοκ συναρτήσεων προγραμματίζονται και αυτά από τον χρήστη και περιέχουν κώδικα**. Ένα μπλοκ συνάρτησης «έχει μνήμη»**, δηλαδή σε αυτό διατίθεται ένα μπλοκ δεδομένων (DB) σαν δικιά του μνήμη. Αυτό το DB λέγεται (instance data block) και είναι μόνιμα δεσμευμένα με το μπλοκ συνάρτησης και για την ακρίβεια με την κλήση (call) του μπλοκ συνάρτησης. Επίσης είναι δυνατόν σε κάθε κλήση μπλοκ συνάρτησης να εκχωρηθεί ένα διαφορετικό μπλοκ δεδομένων (με την ίδια δομή αλλά με διαφορετικές τιμές).

Τα FB παραμετροποιούνται όπως και τα FC επομένως και αυτά χρησιμοποιούνται σε περιπτώσεις που έχουν επαναλαμβανόμενη λογική. Όταν δεν παραμετροποιούνται η λειτουργία τους δεν διαφέρει σε τίποτα από τα FC. Τόσο οι παράμετροι οι οποίες μεταβιβάζονται στα FB όσο και οι στατικές μεταβλητές (static variables) αποθηκεύονται στο instance data block. Οι προσωρινές μεταβλητές (temporary variables) αποθηκεύονται στην περιοχή των τοπικών δεδομένων. Στο τέλος της επεξεργασίας του FB όσα δεδομένα αποθηκεύτηκαν στο instance data block δεν χάνονται ενώ αυτά τα δεδομένα τα οποία αποθηκεύονται στην περιοχή των τοπικών δεδομένων (local data stack) χάνονται. Τα FB περιέχουν πρόγραμμα το οποίο εκτελείται κάθε φορά που τα FB καλείται από άλλο μπλοκ που περιέχει κώδικα. Τα μπλοκ συναρτήσεων (FB) διευκολύνουν τον προγραμματισμό συχνά χρησιμοποιούμενων και σύνθετων συναρτήσεων.

- **ΜΠΛΟΚ ΔΕΔΟΜΕΝΩΝ DB (Data Blocks)**

Τα μπλοκ δεδομένων δεν περιέχουν κώδικα, αλλά περιέχουν δεδομένα του προγράμματος μας. Προγραμματίζοντας τα μπλοκ δεδομένων καθορίζουμε σε ποια μορφή θα αποθηκευτούν τα δεδομένα (σε ποια μπλοκ, με ποια σειρά και με ποιο τύπο δεδομένων). Υπάρχουν δύο βασικοί τρόποι χρησιμοποίησης των μπλοκ δεδομένων:

α) Μπλοκ γενικών δεδομένων (Global data block GD)

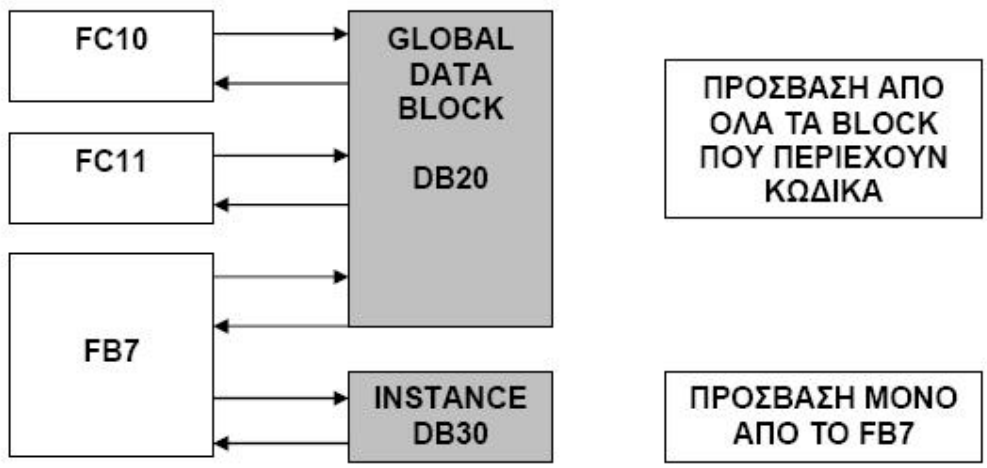
Προγραμματίζονται για κοινή χρήση σε όλο το πρόγραμμα. Ένα μπλοκ γενικών δεδομένων είναι, κατά κάποιο τρόπο, ένα «ελεύθερο», μπλοκ μέσα στο πρόγραμμα του χρήστη και δεν εκχωρείται σε κάποιο μπλοκ «κώδικα».

b) Πρότυπα μπλοκ δεδομένων (instance data block).

Αντίθετα ένα «instance data block» (πρότυπο μπλοκ δεδομένων) εκχωρείται σ' ένα μπλοκ συνάρτησης (FB) και αποθηκεύει ένα μέρος των τοπικών δεδομένων αυτού του μπλοκ συνάρτησης. Το μέγεθος των DB είναι μεταβαλλόμενο, όσον αφορά το μέγιστο μέγεθος αυτού αυτό εξαρτάται από την χρησιμοποιούμενη CPU. Όταν ένα μπλοκ κώδικα (FC, FB, OB) καλείται, αυτό μπορεί ταυτόχρονα να καταλάβει χώρο μνήμης και στην περιοχή των τοπικών δεδομένων (L-Stack) και υπό μορφή ενός DB. Αντίθετα με τα τοπικά δεδομένα, τα δεδομένα οποία περιέχονται σε ένα DB δεν χάνονται όταν κλείσει το DB ή στο τέλος της επεξεργασίας του μπλοκ που περιέχει κώδικα.

Κάθε FB, FC, OB έχει πρόσβαση στο διάβασμα ή γράψιμο ενός DB. Ένα μπλοκ κώδικα έχει την δυνατότητα να ανοίγει ταυτόχρονα ένα global data block και ένα instance data block.

Στην κάτω εικόνα δείχνουμε τους τρόπους πρόσβασης στα DB.

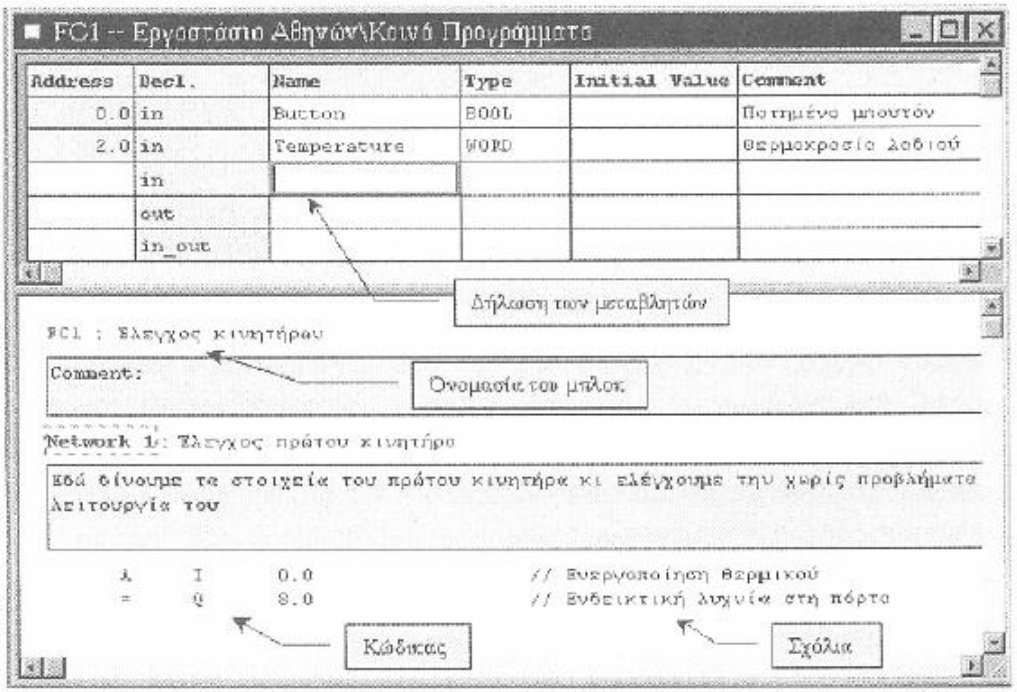


ΔΟΜΗ ΤΩΝ ΜΠΛΟΚ

Σε γενικές γραμμές ένα μπλοκ που περιέχει κώδικα αποτελείται από τα εξής μέρη:

- **Την κεφαλή του μπλοκ** (block header). Αυτό περιλαμβάνει τις ιδιότητες του μπλοκ και το όνομα του.
- **Την περιοχή των δηλώσεων** (declarations) όπου δηλώνονται οι τοπικές μεταβλητές του μπλοκ (Local Variables – L)
- Τέλος την περιοχή η οποία περιλαμβάνει τον **κώδικα** του χρήστη και τα τυχόν σχόλια.

Στην κάτω εικόνα παρουσιάζεται η δομή ενός FC



Τα μπλοκ δεδομένων (DB) είναι και αυτά δομημένα με παρόμοιο τρόπο.

- **Την κεφαλή του μπλοκ** (block header) που περιλαμβάνει τις ιδιότητες του μπλοκ.
- **Την περιοχή των δηλώσεων** declarations όπου δηλώνονται οι τοπικές μεταβλητές του μπλοκ (οι διευθύνσεις των δεδομένων και ο τύπος τους).
- **Το τμήμα με τις αρχικές τιμές**, τις τιμές δηλαδή που θα έχουν κατά την πρώτη εκκίνηση του συστήματος.

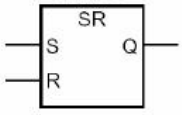
SIMATIC MANAGER

Το Simatic Manager είναι το κυριότερο εργαλείο στο Step7 αφού είναι αυτός που διαχειρίζεται τα project.
Αφού ανοίξουμε αυτό δημιουργούμε ένα νέο project και εφ' όσον τελειώσουμε την διαδικασία προσαρμογής συνεχίζουμε κάνοντας διαμόρφωση του Hardware. Εκεί επιλέγοντας τα υλικά του αυτοματισμού και τα εισάγουμε στη ράγα στήριξης-Rail.
Έχοντας τελειώσει με το Hardware Configuration επανερχόμαστε στο εργαλείο Simatic Manager και στο project που δημιουργήσαμε αφού διαμορφώσουμε το Hardware, το Simatic Manager δημιουργεί το φάκελο System Data και στη συνέχεια δημιουργούμε το πρωτο μας Block οργάνωσης, OB1.
Καθώς το επεξεργαζόμαστε μας δίνεται η δυνατότητα να επιλέξουμε γλώσσα προγραμματισμού οι οποίες είναι οι εξής: LAD, STL και FBD.

Βασικές Εντολές Προγραμματισμού

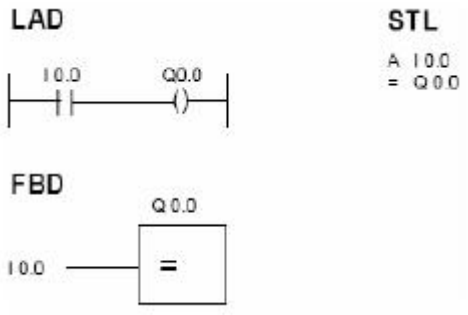
ΔΥΑΔΙΚΕΣ ΛΟΓΙΚΕΣ ΠΡΑΞΕΙΣ – Εντολές bit

Οι εντολές bit λειτουργούν με δύο τιμές 1 και 0. όταν μία επαφή ή ένα πηνίο εξόδου είναι 1 τότε είναι ενεργοποιημένο. Όταν είναι 0 τότε είναι απενεργοποιημένο. Οι εντολές bit συνδυάζουν τα 0 και 1 σήματα των μεταβλητών και αναλόγως των συνδυασμών δίνουν ένα αποτέλεσμα που είναι επίσης 0 ή 1. Αυτό το αποτέλεσμα αναφέρεται και ως **RLO (**Result of Logic Operation).

A I0.0	<address> **---| |---**	Έλεγχος για λογικό "1"
AN I0.0	<address> **---| / |---**	Έλεγχος για λογικό "0"
=Q0.0	<address> **---( )**	Εκχώρηση αποτελέσματος λογικής πράξης σε έξοδο
S Q0.0	<address> **---(S)**	Κάνει την έξοδο "set"
R Q0.0	<address> **---(R)**	Κάνει την έξοδο "reset"
A I0.0 **S Q0.0** **A I0.1** **R Q0.0**	<address> 	Set – Reset Flip flop

ΕΚΧΩΡΗΣΗ ΑΠΟΤΕΛΕΣΜΑΤΟΣ – ASSIGNMENT (=)

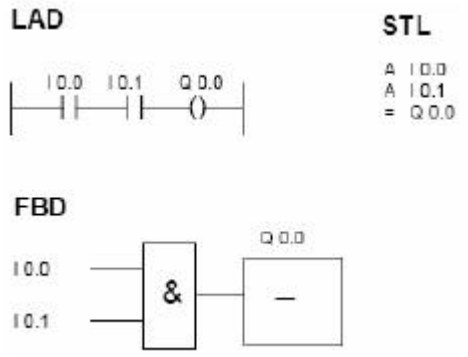
Η εντολή ‘’=’’ αντιγράφει την τιμή του RLO στον τελεστή που ακολουθεί.



Η εντολή της ισότητας κλείνει την λογική αλυσίδα των πράξεων.

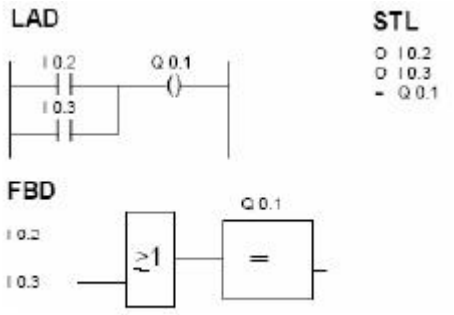
ΛΟΓΙΚΗ ΠΡΑΞΗ AND

Η AND αντιστοιχεί σε μία εν σειρά σύνδεση επαφών του κυκλωματικού διαγράμματος. Αν στο παρακάτω παράδειγμα έστω και μία από τις εισόδους έχει τιμή ‘’0’’, τότε η Q0.0 θα έχει τιμή ‘’0’’. Για να έχει η Q0.0 κατάσταση ‘’1’’ θα πρέπει όλες οι είσοδοι να έχουν κατάσταση ‘’1’’.



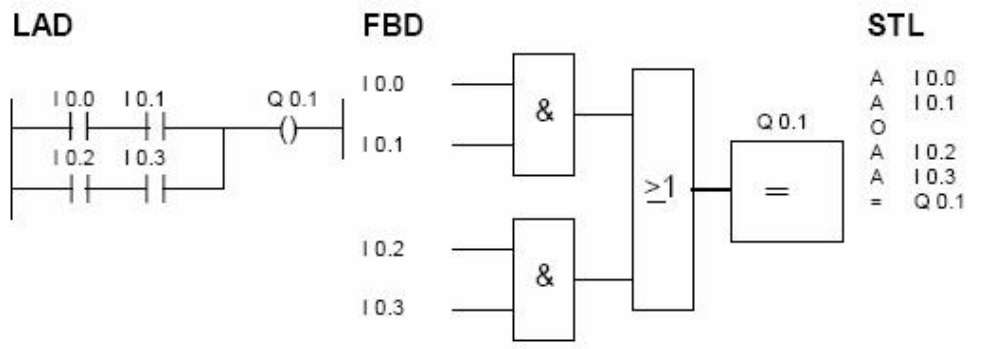
ΛΟΓΙΚΗ ΠΡΑΞΗ OR

Η OR αντιστοιχεί σε μία παράλληλη σύνδεση επαφών του κυκλωματικού διαγράμματος. Αν στο παρακάτω παράδειγμα έστω και μία από τις εισόδους έχει τιμή ‘’1’’, τότε η Q0.1 θα έχει τιμή ‘’1’’. Για να έχει η Q0.0 κατάσταση ‘’0’’ θα πρέπει όλες οι είσοδοι να έχουν κατάσταση ‘’0’’.



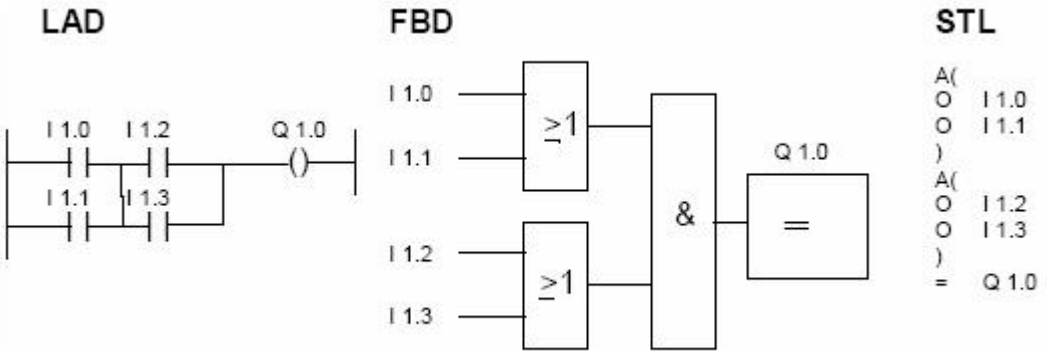
ΛΟΓΙΚΗ ΠΡΑΞΗ ‘’AND ΠΡΙΝ ΤΟ OR’’

Η ‘’AND πριν το OR’’ αντιστοιχεί σε μία παράλληλη σύνδεση ομάδων επαφών που εν σειρά συνδεδεμένες μεταξύ τους. Στο παρακάτω παράδειγμα για να έχει η Q0.1 κατάσταση ‘’1’’ θα πρέπει τουλάχιστον όλες οι επαφές ενός κλάδου να έχουν κατάσταση ‘’1’’. Οι AND πράξη έχει προτεραιότητα της OR και για αυτό δεν χρειάζεται η χρήση των παρενθέσεων στην STL.



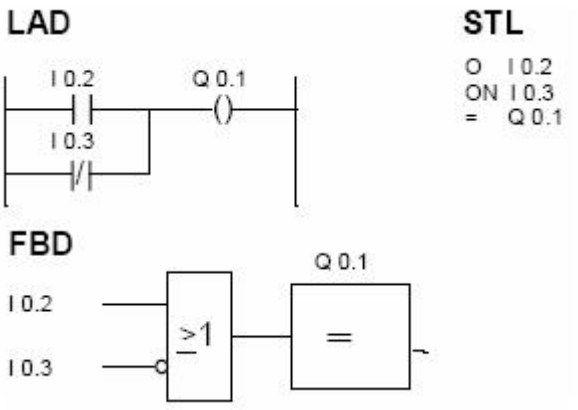
ΛΟΓΙΚΗ ΠΡΑΞΗ ‘’ OR ΠΡΙΝ ΤΟ AND’’

Η ‘’ OR πριν το AND’’ αντιστοιχεί σε μία εν σειρά σύνδεση ομάδων επαφών που συνδεδεμένες μεταξύ τους παράλληλα. Στο παρακάτω παράδειγμα για να έχει η Q1.0 κατάσταση ‘’1’’ θα πρέπει τουλάχιστον μία επαφή του κάθε κλάδου να έχει κατάσταση ‘’1’’. Οι AND πράξη έχει προτεραιότητα της OR και για αυτό απαιτείται η χρήση των παρενθέσεων στην STL.

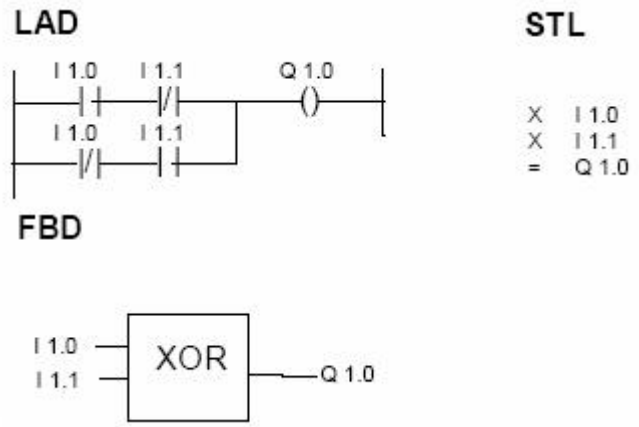


ΕΡΩΤΗΣΗ ΓΙΑ ΚΑΤΑΣΤΑΣΗ ΣΗΜΑΤΟΣ '' 0'' (AN,ON,XN)

Δίνουμε παράδειγμα για την ON:



ΑΠΟΚΛΕΙΣΤΙΚΟ OR (X)

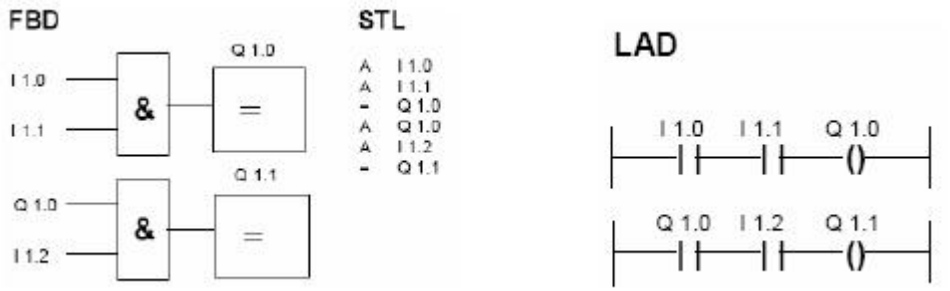


Στο παράδειγμα βλέπουμε τη λειτουργία και τον προγραμματισμό της λογικής πράξης EX-OR. Για να γίνει η Q1.0 ''1'', θα πρέπει μόνο μία επαφή να είναι κλειστή (''1'').

ΕΡΩΤΗΣΗ ΕΞΟΔΩΝ

Μπορούμε να χρησιμοποιήσουμε και τις εξόδους Q (όπως και τις μνήμες M) σαν τελεστές μιας λογικής πράξης (AND,OR…).

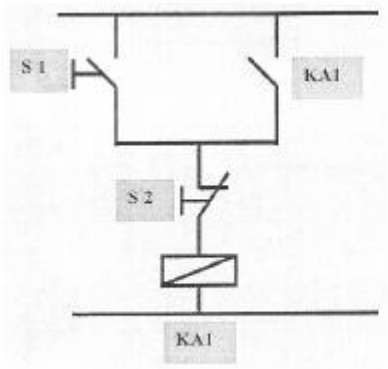
Παράδειγμα:



Σημείωση: Να θυμόμαστε ότι προγραμματίζοντας σε LAD σε κάθε Network μπορούμε να έχουμε μόνο μια ανεξάρτητη έξοδο.

ΕΝΤΟΛΕΣ ΜΝΗΜΗΣ SET-RESET

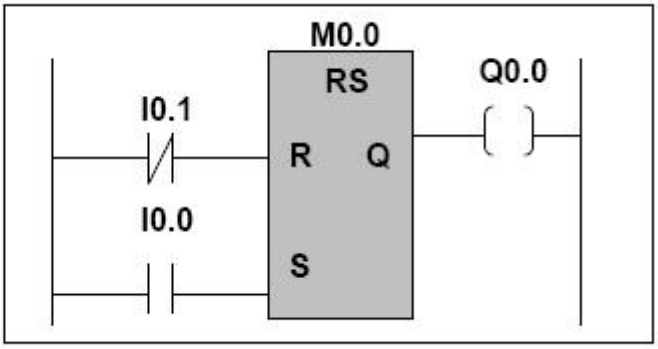
Στον προγραμματισμό των εντολών SET-RESET προτεραιότητα έχει η εντολή που προγραμματίζεται τελευταία.



S1	→	I0.0
S2	→	I0.1
KA1	→	Q0.0

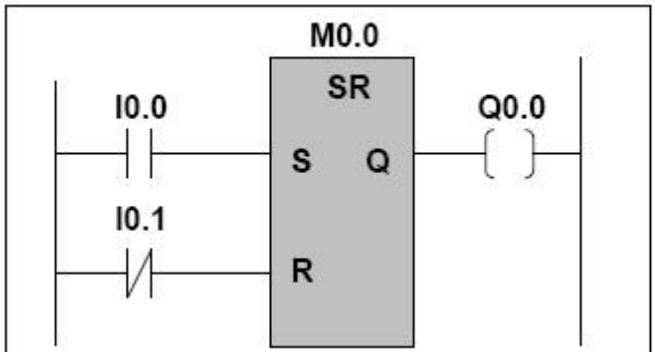
ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΕ STL ΚΑΙ LADDER ΜΕ ΠΡΟΤΕΡΑΙΟΤΗΤΑ ΣΤΟ SET

AN	I0.1
R	Q0.0
A	I0.0
S	Q0.0



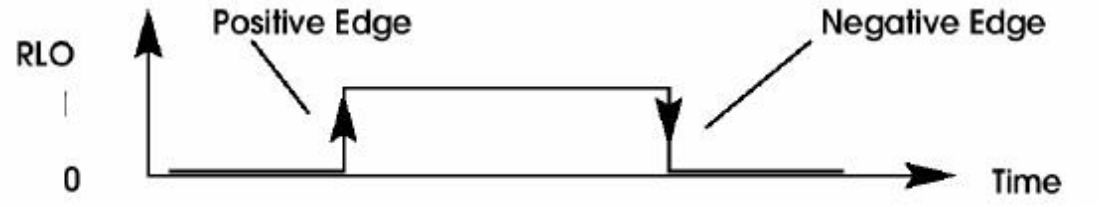
ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΣΕ STL ΚΑΙ LADDER ΜΕ ΠΡΟΤΕΡΑΙΟΤΗΤΑ ΣΤΟ RESET

```
A   I0.0
S   Q0.0
AN  I0.1
R   Q0.0
```



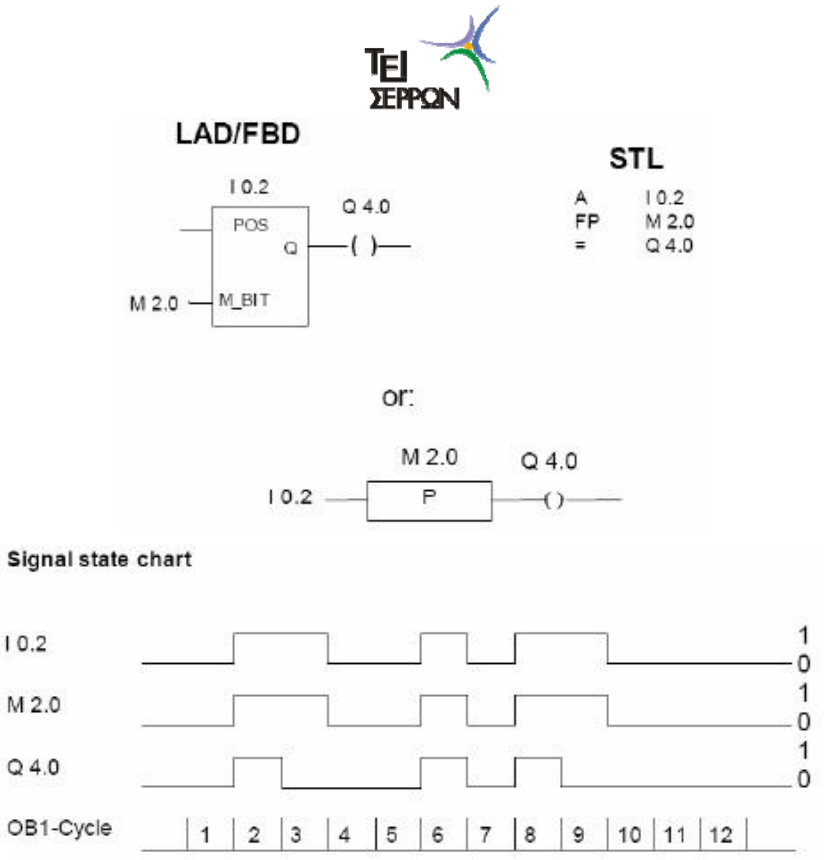
ΕΝΤΟΛΕΣ ΑΝΑΓΝΩΡΙΣΗΣ ΠΑΡΥΦΩΝ – EDGE OPERATIONS (FP- FN)

Υπάρχει η περίπτωση να θέλουμε να αναγνωρίσουμε την μετάβαση από κατάσταση «0» σε «1» ή από «1» σε «0» μιας μεταβλητής ή ενός λογικού αποτελέσματος. Αυτό μπορούμε να το κάνουμε με τις εντολές αναγνώρισης παρυφών.



ΑΝΑΓΝΩΡΙΣΗ ΘΕΤΙΚΩΝ ΠΑΡΥΦΩΝ – FP

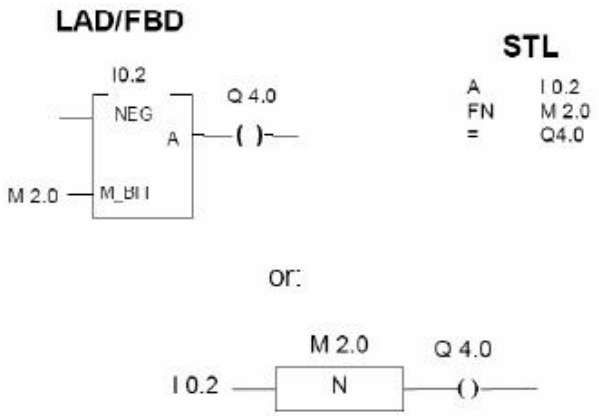
Αν ανιχνευθεί στην I0.2 μία μεταβολή κατάστασης από λογικό ‘’0’’ σε λογικό ‘’1’’ (θετική παρυφή), τότε η Q4.0 θα πάρει την τιμή ‘’1’’ για ένα κύκλο εκτέλεσης προγράμματος.



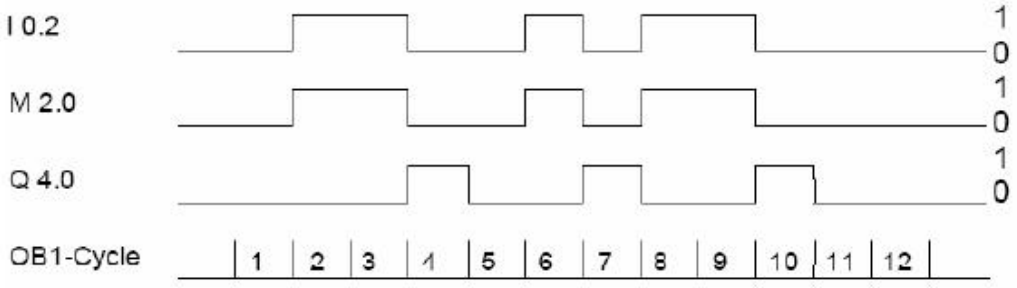
Η θετική παρυφή αναγνωρίζεται καθώς το σύστημα του αυτοματισμού αποθηκεύει το RLO που δίνει η λογική πράξη AND στο bit μνήμης M2.0 και το συγκρίνει με το RLO του τρέχοντος κύκλου.

ΑΝΑΓΝΩΡΙΣΗ ΑΡΝΗΤΙΚΩΝ ΠΑΡΥΦΩΝ

Αν ανιχνευθεί στην I0.2 μία μεταβολή κατάστασης από λογικό ‘’1’’ σε λογικό ‘’0’’ (αρνητική παρυφή), τότε η Q4.0 θα πάρει την τιμή ‘’1’’ για ένα κύκλο εκτέλεσης προγράμματος.



Signalstate chart



Η αρνητική παρυφή αναγνωρίζεται καθώς το σύστημα του αυτοματισμού αποθηκεύει το RLO που δίνει η λογική πράξη AND στο bit μνήμης M2.0 και το συγκρίνει με το RLO του τρέχοντος κύκλου.

Προγραμματίζοντας με Σύμβολα

ΠΙΝΑΚΑΣ ΣΥΜΒΟΛΩΝ – ΣΥΜΒΟΛΙΚΟΣ ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ

Μια από τις δυνατότητες που μας δίνει το STEP 7 είναι να μπορούμε να γράψουμε το πρόγραμμα μας με χρήση συμβόλων και όχι με χρήση των λογικών (απόλυτων) διευθύνσεων. Ο λόγος που χρησιμοποιούμε τον συμβολικό προγραμματισμό είναι κυρίως για να γίνεται ευκολότερη η ανάγνωση του προγράμματος. Φυσικά για να συμβαίνει αυτό πρέπει ως σύμβολα να βάζουμε κάποια ονόματα που να έχουν συμβαίνει αυτό πρέπει ως σύμβολα να βάζουμε κάποια ονόματα που να έχουν σημασία για τον αυτοματισμό μας.
Για να μπορέσουμε να το κάνουμε αυτό θα πρέπει πρώτα να συμπληρώσουμε του πίνακα συμβόλων.
Επιπλέον, η αντίστοιχη συμβολικών ονομασιών στις μεταβλητές μπορεί να μας βοηθήσει να κάνουμε μεταβολές στο πρόγραμμα με μεγαλύτερη ευκολία.

Άνοιγμα Symbol Table: Στο γενικό menu του Simatic Manager ανοίγουμε το project μας μέχρι να φτάσουμε στο S7 program.

Εκεί επιλέγουμε το εικονίδιο Symbols και ανοίγει ο Symbol Editor που περιέχει τον πίνακα συμβόλων. Εάν θέλουμε μπορούμε και εδώ να κάνουμε αλλαγές ή να συμπληρώσουμε κλείνοντας όμως θα πρέπει να θυμηθούμε να κάνουμε save. Επίσης από αυτόν τον δρόμο μπορούμε να συμπληρώσουμε και για πρώτη φορά τον πίνακα συμβόλων απλώς δεν θα έχουμε στην διάθεση μας (πάνω στον πίνακα) τις διαθέσιμες απόλυτες διευθύνσεις.

Σε περίπτωση που έχει σβηστεί ή θέλουμε να δημιουργήσουμε ένα νέο πίνακα συμβόλων, πάμε στην ετικέτα S7 program και με δεξί κλικ στο δεξί μέρος της οθόνης επιλέγουμε : **Insert new object → Symbol Table.**

ΣΤΗΛΗ SYMBOL: Εδώ γράφουμε την συμβολική διεύθυνση (δηλαδή το σύμβολο) που θέλουμε να δώσουμε σε κάθε μια απόλυτη διεύθυνση. Η συμβολική ονομασία δεν πρέπει να είναι μεγαλύτερη από 24 χαρακτήρες. Σε ένα πίνακα συμβόλων μπορούμε να εισάγουμε μέχρι 16380 σύμβολα.

Σημείωση: δεν μπορούμε να δώσουμε συμβολική ονομασία σε δεδομένα από data block. Αυτό γίνεται μόνο στον πίνακα δηλώσεων των DB.

ΣΤΗΛΗ ADDRESS: Εισάγουμε την απόλυτη διεύθυνση της μεταβλητής στην οποία θέλουμε να αντιστοιχίσουμε ένα σύμβολο.

ΣΤΗΛΗ DATA TYPE: Συμπληρώνεται αυτόματα για κάθε απόλυτη διεύθυνση στην οποία δίνουμε συμβολική ονομασία. Μπορούμε να επιλέξουμε από τους διαθέσιμους τύπους Step7, αλλά αν δεν υπάρχει σωστή αντιστοιχία με τον

τύπο της μεταβλητής το πρόγραμμα θα μας βγάλει αυτόματα ένδειξη σφάλματος.

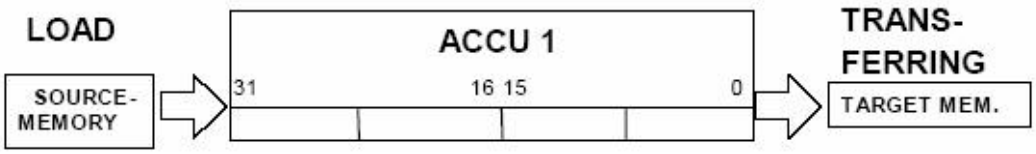
ΣΤΗΛΗ COMMENT: Εδώ μπορούμε να συμπληρώνουμε τυχόν σχόλια που θέλουμε για κάθε σύμβολο. Το κάθε σχόλιο δεν μπορεί να έχει μέγεθος πάνω από 80 χαρακτήρες.

ΕΝΤΟΛΕΣ ΜΑΖΙΚΗΣ ΕΠΕΞΕΡΓΑΣΙΑΣ ΔΕΔΟΜΕΝΩΝ

ΕΝΤΟΛΕΣ LOAD – TRANFER, MOVE

Στον προγραμματισμό της STEP7, με τις εντολές Load και Transfer μπορούμε να ανταλλάξουμε δεδομένα μεταξύ διαφόρων περιοχών μνήμης του συστήματος τα οποία έχουν μέγεθος Byte,Word και Double Word. Η ανταλλαγή αυτών των δεδομένων δεν γίνεται απευθείας αλλά μέσω του συσσωρευτή ACCU1. Ο ACCU1 είναι ένας resister του επεξεργαστή που λειτουργεί ως προσωρινή ενδιάμεση μνήμη (buffer).

Η ροή πληροφορίας από μια περιοχή μνήμης στον ACCU1 λέγεται φόρτωση (loading) ενώ η αντίθετη ροή πληροφορίας λέγεται μεταφορά (transferring) (τα περιεχόμενα του ACCU1 μεταφέρονται στην περιοχή μνήμης).



ΣΥΝΑΡΤΗΣΗ ΦΟΡΤΩΣΗΣ L

Η συνάρτηση φόρτωσης αποτελείται από τον κωδικό λειτουργίας L και από μια σταθερά ή αναγνωρίσιμη διεύθυνση.

Π.χ

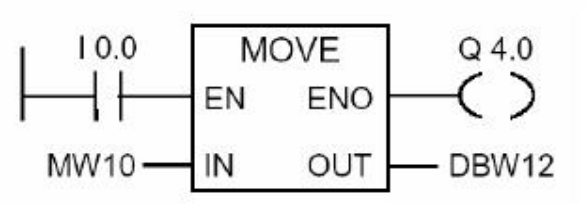
Έστω ότι θέλουμε να φορτώσουμε τον αριθμό +5

L+5

ΣΥΝΑΡΤΗΣΗ ΜΕΤΑΦΟΡΑΣ T

Η συνάρτηση μεταφοράς αποτελείται από τον κωδικό λειτουργίας T και την ψηφιακή διεύθυνση στην οποία θα μεταφερθούν τα περιεχόμενα του συσσωρευτή:
Για παράδειγμα η εντολή T MW120 μεταφέρει το περιεχόμενο του ACCU 1 στην καθορισμένη διεύθυνση MW120

Στην LADDER οι εντολές φόρτωσης και μεταφοράς γίνονται με την εντολή MOVE



όπου η φόρτωση – μεταφορά μπορούν να γίνουν υπό συνθήκη και η τιμή της εξόδου ENO είναι πάντα ίδια με την τιμή της εισόδου EN.

ΣΥΝΑΡΤΗΣΕΙΣ ΧΡΟΝΙΚΩΝ – TIMER FUNCTIONS

Πολύ συχνά για την υλοποίηση εργασιών ελέγχου χρησιμοποιούμε συναρτήσεις διαφόρων χρονικών. Οι συναρτήσεις αυτές των χρονικών είναι ενσωματωμένες στη CPU του συστήματος αυτοματισμού μας αλλά η τοποθέτηση της επιθυμητής τιμής, καθώς και η έναρξη και η παύση του χρονικού πρέπει να γίνει από τον χρήστη μέσω προγραμματισμού. Κάθε CPU s7-300 υποστηρίζει τον προγραμματισμό 256 χρονικών και διαθέτει μια ξεχωριστή περιοχή στη την μνήμη της μέσα στην οποία κάθε χρονικό καταλαμβάνει τον χώρο μιας λέξης (word) δηλαδή 16 bit.

Η ονοματολογία των χρονικών είναι της μορφής **Τχ .**

Εντολές που χρησιμοποιούμε για τα χρονικά

1. Απαραίτητες για την λειτουργία τους.

- Δήλωση της αιτίας εκκίνησης
- Δήλωση της χρονικής καθυστέρησης
- Δήλωση είδους του χρονικού
- Ονομασία Χρονικού

2. Προαιρετικές Εντολές

- Δήλωση της αιτίας μηδενισμού (reset)
- Να φορτώσουμε το περιεχόμενο του χρονικού σε κάποια μνήμη

Να χρησιμοποιήσουμε επαφή του χρονικού. Επαφή οποιουδήποτε χρονικού μπορούμε να ζητήσουμε με τις εντολές ελέγχου **A, O, X, AN, ON, XN.**

ΕΚΚΙΝΗΣΗ ΧΡΟΝΙΚΟΥ

Ένα χρονικό ξεκινάει όταν είσοδο εκκίνησής του ανιχνευθεί μία μεταβολή από λογικό ‘’0’’ σε λογικό ‘’1’’ (θετική παρυφή). Στην STL για να ξεκινήσουμε ένα χρονικό πρέπει να προγραμματίσουμε τρεις εντολές.

Π.χ.

A I0.0 Ερώτηση για την κατάσταση σήματος στην I0.0
L S5T#2S Φόρτωσε την τιμή του χρονικού (2s) στον ACCU1

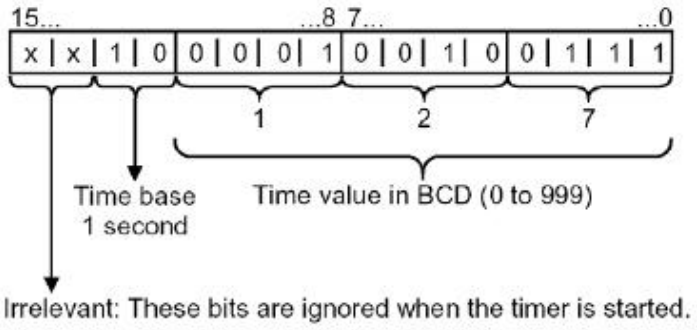
SE T5 Τα παραπάνω ισχύουν για το χρονικό με όνομα T5 που είναι τύπου SE

ΤΙΜΗ ΧΡΟΝΙΚΟΥ

Ένα χρονικό πρέπει πάντα να εκτελείται για συγκεκριμένο χρόνο. Το μέγεθος της τιμής του χρόνου μπορεί να προσδιοριστεί είτε ως μία προκαθορισμένη σταθερά στο πρόγραμμα είτε να δοθεί με τη μορφή δεδομένων σε μία IW, QW, LW, MW ή σε μία word ενός Data Block.

Από τα 16 bit που καταλαμβάνει κάθε χρονικό τα bit 0 έως και 9 περιέχουν την τιμή της χρονικής καθυστέρησης σε δυαδικό κώδικα ενώ τα bit με διεύθυνση 12 και 13 περιέχουν την χρησιμοποιούμενη βάση χρόνου σε δυαδικό κώδικα.

Στην κάτω εικόνα παρουσιάζουμε το περιεχόμενο των 16 bit ενός χρονικού.



RESET ΧΡΟΝΙΚΟΥ (R)

Αν ανιχνευθεί σήμα στην είσοδο reset του χρονικού τότε τερματίζεται η διαδικασία μέτρησης χρόνου (το χρονικό ακυρώνεται). Η τρέχουσα τιμή του διαγράφεται και η έξοδός του (Q) απενεργοποιείται.

ΦΟΡΤΩΣΗ ΧΡΟΝΟΥ (L/LC)

Ο χρόνος φορτώνεται δυαδικά κωδικοποιημένος σε μια word. Η τιμή αυτής της word μπορεί να φορτωθεί ως δυαδικός αριθμός (με την εντολή L) ή ως BCD αριθμός (με την εντολή LC) στον ACCU1 για περαιτέρω επεξεργασία.

ΕΡΩΤΗΣΗ ΚΑΤΑΣΤΑΣΗΣ ΧΡΟΝΙΚΟΥ

Ένα χρονικό μπορεί να ερωτηθεί γα την κατάσταση του (‘’0’’ ή ‘’1’’) και να χρησιμοποιηθεί σε λογικές πράξεις (π.χ. A T1, ON T1…).

Τα είδη των χρονικών που έχουμε είναι:

- Χρονικό παλμού SP
- Χρονικό παλμού με συγκράτηση SE
- Χρονικό καθυστέρησης έναρξης SD
- Χρονικό καθυστέρησης έναρξης με αυτοσυγκράτηση SS
- Χρονικό καθυστέρησης λήξης SF

ΑΠΑΡΙΘΜΗΤΕΣ- COUNTERS

Στα συστήματα ελέγχου οι συναρτήσεις των μετρητών χρειάζονται για την καταμέτρηση αριθμών, τεμαχίων, επαναλήψεων μιας διαδικασίας, καταμέτρηση αποστάσεων κ.α.

Στα Simatic S7 οι μετρητές είναι ήδη ενσωματωμένοι στη CPU και κατέχουν μία δική τους ξεχωριστή περιοχή μνήμης. Αυτή η περιοχή διαθέτει μία 16bit word για κάθε μετρητή. Μέσα σε ένα πρόγραμμα μπορούμε να προγραμματίσουμε μέχρι 256 μετρητές.
Με τους μετρητές έχουμε τη δυνατότητα να μετράμε προς τα επάνω και προς τα κάτω. Το εύρος μέτρησης είναι από 0 έως 999.

Εντολές μετρητή

S ,CV	→	Προτοποθέτηση τιμής περιεχομένου μετρητή
R	→	Μηδενισμός περιεχομένου του μετρητή
CU	→	Αύξηση περιεχομένου κατά 1 (μέχρι 999)
CD	→	Μείωση περιεχομένου κατά 1 (μέχρι 0)
L	→	Φόρτωση του περιεχομένου του μετρητή σε δυαδική μορφή
LC	→	Φόρτωση του περιεχομένου του μετρητή σε BCD κώδικα
A, AN, O, ON	→	Έλεγχος κατάστασης ενός μετρητή

ΤΙΜΗ ΑΠΑΡΙΘΜΗΤΗ (CV)

Όταν ένας counter γίνεται set τότε φορτώνεται σε αυτόν η τιμή προτοποθέτησης που είναι το περιεχόμενο του ACCU1. Η φόρτωση της τιμής του ACCU1 στον μετρητή γίνεται με την χρήση της εντολής Load και μπορεί να είναι σε BCD ή δυαδικό κώδικα.
Η τιμή προτοποθέτησης που θα φορτώσουμε στον μετρητή μπορεί να είναι:

- **Input word** **IW..**
- **Output word** **QW..**
- **Memory bit word** **MW..**
- **Data word** **DBW/DIW ..**
- **Constant** **C#5, 2#...etc.**

ΦΟΡΤΩΣΗ ΠΕΡΙΕΧΟΜΕΝΟΥ ΤΟΥ ΑΠΑΡΙΘΜΗΤΗ (L-LC)

Η τιμή του μετρητή αποθηκεύεται στην περιοχή μνήμης των counters σε μια word σε δυαδικό κώδικα. Η τιμή αυτή μπορεί να φορτωθεί στον ACCU1 για

περαιτέρω επεξεργασία ως δυαδικός αριθμός (με την εντολή L C1) ή ως BCD αριθμός (με την εντολή LC C1).

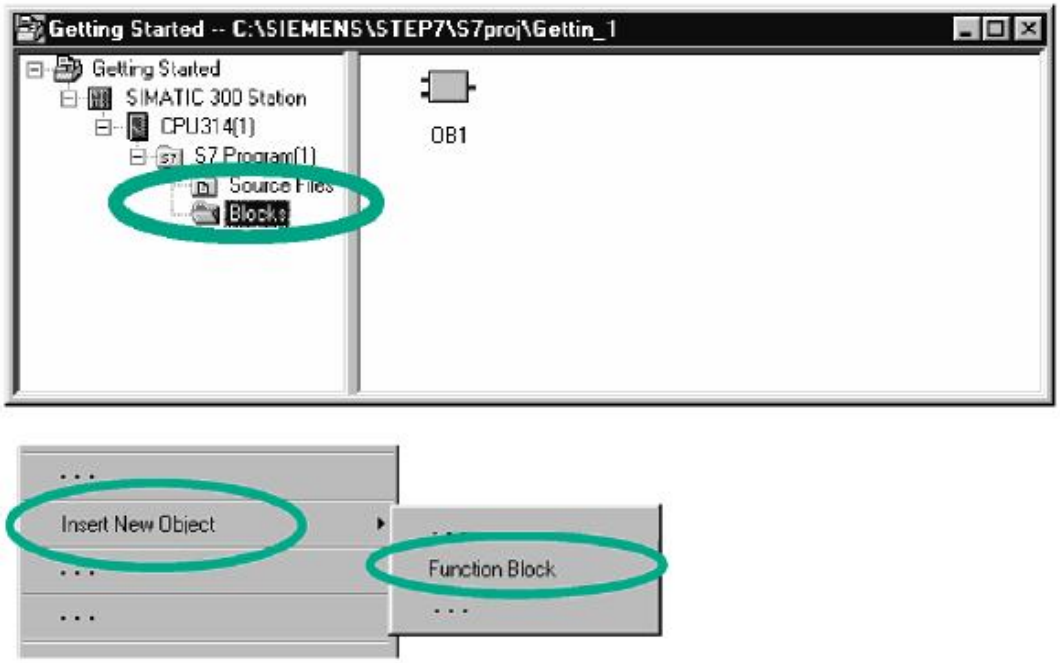
ΕΡΩΤΗΣΗ ΓΙΑ ΚΑΤΑΣΤΑΣΗ ΤΟΥ ΑΠΑΡΙΘΜΗΤΗ (Q)

Η Q του μετρητή μπορεί να πάρει δύο πιθανές καταστάσεις :

- ‘’0’’ .. όταν η τιμή μέτρησης του μετρητή είναι 0.
- ‘’1’’ .. όταν η τιμή μέτρησηδς του μετρητή είναι διάφορη του μηδενός.

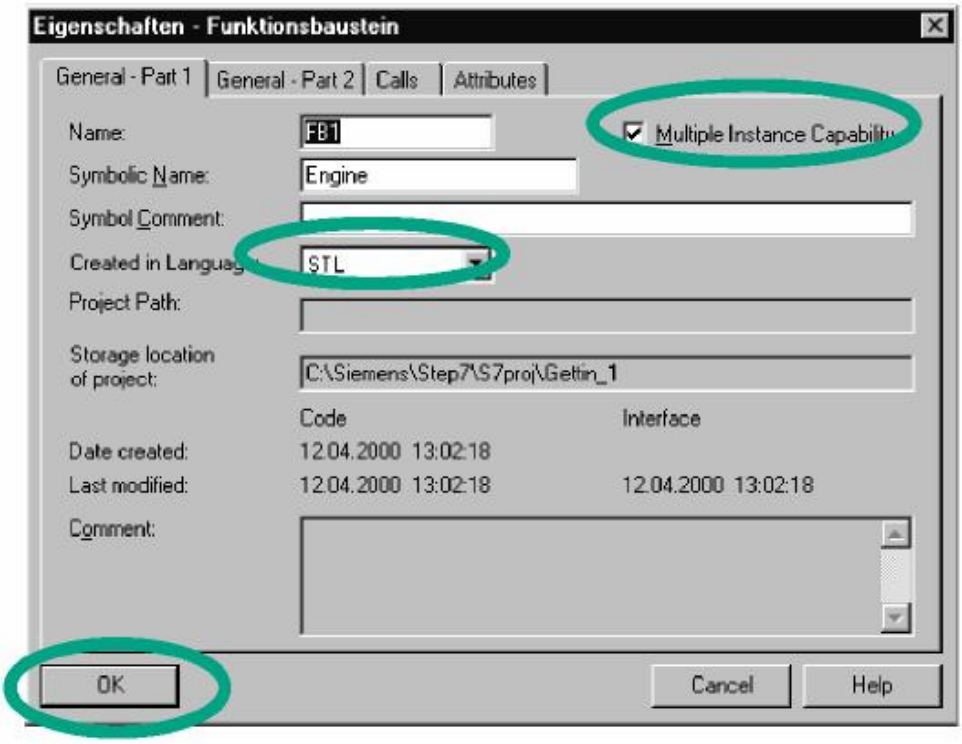
ΔΗΜΙΟΥΡΓΙΑ ΝΕΟΥ ΜΠΛΟΚ

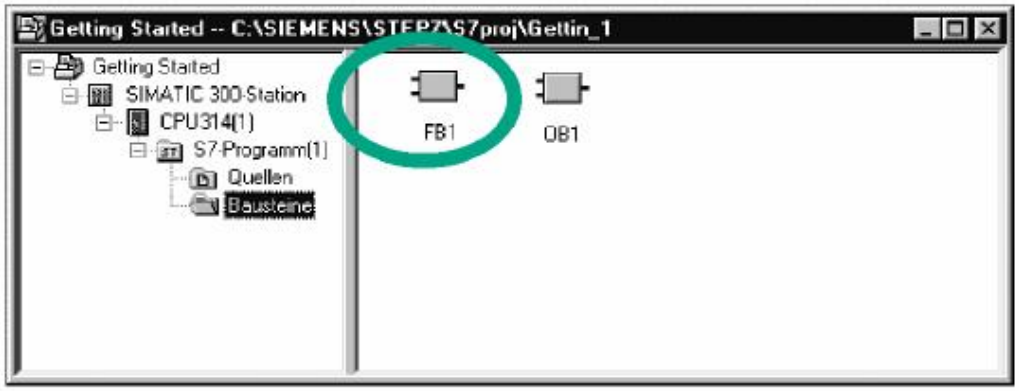
Έχοντας ανοιγμένη την δομή του project μας στο SIMATIC MANAGER επιλέγουμε το αντικείμενο Blocks στο αριστερό τμήμα του παραθύρου (αριστερό κλικ) κατόπιν κάνουμε δεξί κλικ ανοίγει νέος πίνακας διαλόγου επιλέγουμε Insert New Object και το είδος του μπλοκ που θέλαμε να δημιουργήσουμε.



Το ίδιο μπορεί να γίνει και από το μενού επιλέγοντας **Insert → S7 Block →** επιλογή του είδους του μπλοκ που θέλουμε να δημιουργήσουμε.

Και στις δύο περιπτώσεις έχοντας επιλέξει κάποιο μπλοκ ανοίγει νέο παράθυρο διαλόγου όπου κυρίως αναγράφεται η ονομασία του μπλοκ και η γλώσσα προγραμματισμού. Στην παρακάτω εικόνα φαίνεται η μορφή του παραθύρου διαλόγου και η τελική μορφή στην οθόνη του PC όταν επικυρώσουμε επιλέγοντας το μπουτόν OK.





Για να ανοίξουμε το μπλοκ που δημιουργήσαμε κάνουμε διπλό αριστερό κλικ πάνω στο αντικείμενο.

Εκείνο που πρέπει να έχουμε υπ' όψη μας είναι ότι για να τρέξει ένα μπλοκ προγράμματος θα πρέπει να το καλέσουμε από το μπλοκ οργάνωσης OB1.

Μια κλήση μπλοκ αποτελείται την εντολή κλίσης την ονομασία του μπλοκ που καλούμε. Μετά την εκτέλεση της εντολής κλήσης η CPU συνεχίζει με την εκτέλεση του προγράμματος που βρίσκεται μέσα στο καλούμενο μπλοκ. Το μπλοκ αυτό εκτελείται μέχρι να εμφανιστεί μια εντολή τερματισμού του μπλοκ. Τότε η CPU επιστρέφει στο μπλοκ από το οποίο έγινε η κλήση και συνεχίζει την εκτέλεση του υπόλοιπου προγράμματος.

ΕΝΤΟΛΗ ΚΛΗΣΗΣ ΜΠΛΟΚ ΥΠΟ ΣΥΝΘΗΚΗ - CC (Conditional Call)

Με την εντολή κλήσης μπλοκ CC μπορούμε να καλέσουμε FC's, FB's, SFC's και SFB's, αν η λογική πράξη πριν από την εντολή κλήσης δίνει RLO = ''1''. Με αυτή την εντολή δεν μεταφέρονται οι παράμετροι των μπλοκ για αυτό αποτελεί προϋπόθεση η εντολή CC να μην χρησιμοποιείται για παραμετροποιημένα μπλοκ.

Παράδειγμα: Αν η I0.0 είναι ''1'' τότε κάλεσε την FC1.

ΕΝΤΟΛΗ ΚΛΗΣΗΣ ΜΠΛΟΚ ΧΩΡΙΣ ΣΥΝΘΗΚΗ - UC (Unconditional Call)

Η εντολή κλήσης UC είναι μια εντολή χωρίς όρους (χωρίς συνθήκη) το οποίο σημαίνει ότι το καλούμενο μπλοκ εκτελείται χωρίς καμία προϋπόθεση. Με την εντολή κλήσης μπλοκ CC μπορούμε επίσης να καλέσουμε μη παραμετροποιημένα FC's, FB's, SFC's και SFB's.

Παράδειγμα:

ΕΝΤΟΛΗ ΚΛΗΣΗΣ CALL

Με την εντολή CALL καλούμε παραμετροποιημένα FB, FC, SFB και SFC. Η εντολή αυτή είναι μια κλήση που εκτελείται ανεξαρτήτως της τιμής του RLO. Αν χρησιμοποιήσουμε την εντολή CALL για να καλέσουμε ένα μη παραμετροποιημένο μπλοκ, τότε αυτή λειτουργεί όπως ακριβώς η UC. Με την εντολή CALL δεν μπορούμε να καλέσουμε μπλοκ οργάνωσης (OB), αυτά καλούνται μόνο από το λειτουργικό σύστημα ανάλογα με την εμφάνιση συγκεκριμένων γεγονότων (events).

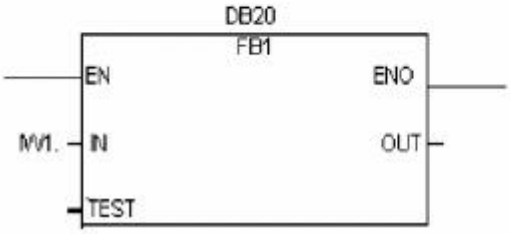
ΚΛΗΣΗ ΜΠΛΟΚ ΣΥΝΑΡΤΗΣΗΣ (FB) ΜΕΣΩ ΤΗΣ ΕΝΤΟΛΗΣ CALL

Παράδειγμα: Κλήση του παραμετροποιημένου FB1 που έχει δεσμευμένο instance DB20.

STL

CALL	FB1, DB20
IN := IW 1	IN (Formal parameter) assigned to IW 1 (Actual parameter).
OUT :=	OUT (Formal parameter) assigned to no parameter.
TEST :=	TEST (Formal parameter) assigned to no parameter.

LAD/FBD



Μπορούμε να καλέσουμε ένα μπλοκ συνάρτησης FB ορίζοντας μετά την εντολή CALL την ονομασία του μπλοκ συνάρτησης (π.χ. FB1) και το πρότυπο μπλοκ δεδομένων που σχετίζεται με την κλήση (π.χ. DB20) χωρισμένα μεταξύ τους με το σύμβολο “ , ”. Η λειτουργία της εντολής CALL ακολουθείται από την λίστα με τις παραμέτρους του μπλοκ. Η χρήση του DB είναι υποχρεωτική.

Στην LAD απλώς κάνουμε Drag and Drop από την λίστα των μπλοκ που εμφανίζεται αριστερά στο παράθυρο των εντολών.

ΚΛΗΣΗ ΣΥΝΑΡΤΗΣΗΣ (FC) ΜΕΣΩ ΕΝΤΟΛΗΣ CALL

Ισχύουν τα ίδια με τα FB με τη διαφορά ότι δεν χρειάζεται να καθορίσουμε το αντίστοιχο instance DB.

ΜΠΛΟΚ ΔΕΔΟΜΕΝΩΝ (DATA BLOCKS)

DATA BLOCKS (DB)

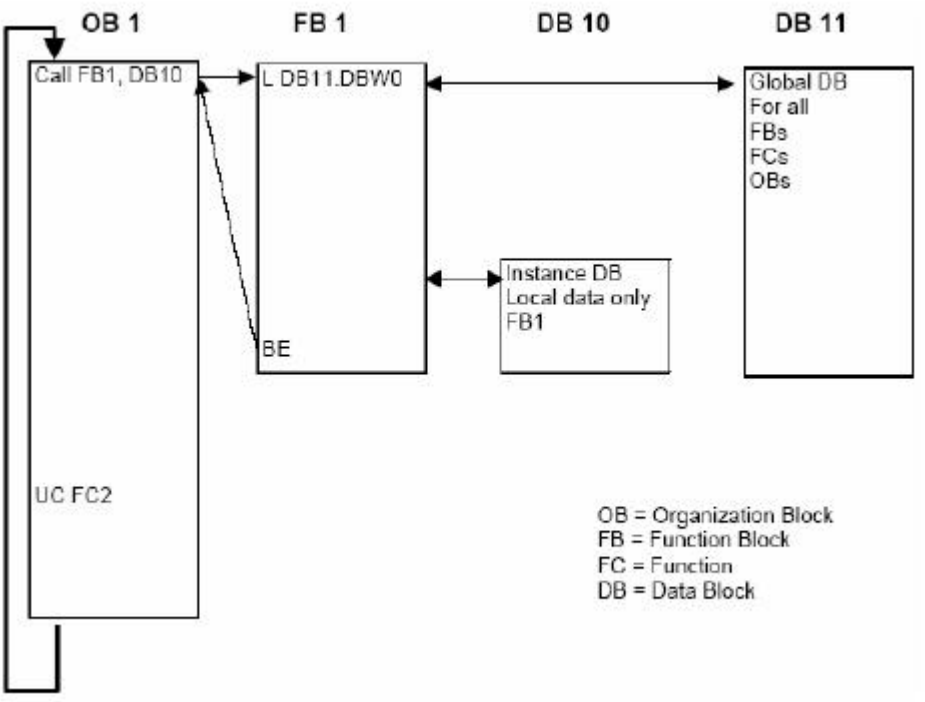
Τα Data Blocks είναι ουσιαστικά χώροι αποθήκευσης δεδομένων του προγράμματος. Η κύρια διαφορά τους από τα υπόλοιπα μπλοκ είναι ότι σε αυτά δεν μπορούμε να γράψουμε πρόγραμμα.

Τα DB μπορούν να κληθούν και να χρησιμοποιηθούν από τα άλλα μπλοκ λογικής για διαχείριση πληροφοριών. Στα DB αποθηκεύουμε δεδομένα, που σε αντίθεση με τα προσωρινά δεδομένα δεν επικαλύπτονται ή χάνονται μετά το κλείσιμο του μπλοκ. Τα DB καταλαμβάνουν χώρο από την περιοχή της μνήμης εργασίας (work memory). Στα DB υπάρχει πρόσβαση σε πληροφορίες μεγέθους bit, byte, word, double word …, ανάλογα με τον τρόπο που έχουμε δομήσει το DB και με τις εντολές που το καλούμε. Στα DB υπάρχει δυνατότητα χρήσης απόλυτης ή συμβολικής διευθυνσιοδότησης.

Οι δύο βασικοί τύποι των DB είναι :

I. **Γενικά data block (Global DB)**

II. **Εξαρτημένα Data Block (Instance DB)**



Γενικά DB (Global DB)

Τα Global DB είναι "ελεύθερα" μπλοκ στο πρόγραμμα χρήστη τα οποία δεν

δεσμεύονται από κάποιο μπλοκ προγραμματισμού και συνεπώς οι πληροφορίες τους είναι προσπελάσιμες από όλα τα μπλοκ κώδικα του προγράμματος (OB's, FC's, FB's). Αυτός είναι και ο τύπος των DB που χρησιμοποιούμε σε αυτή την εργασία.

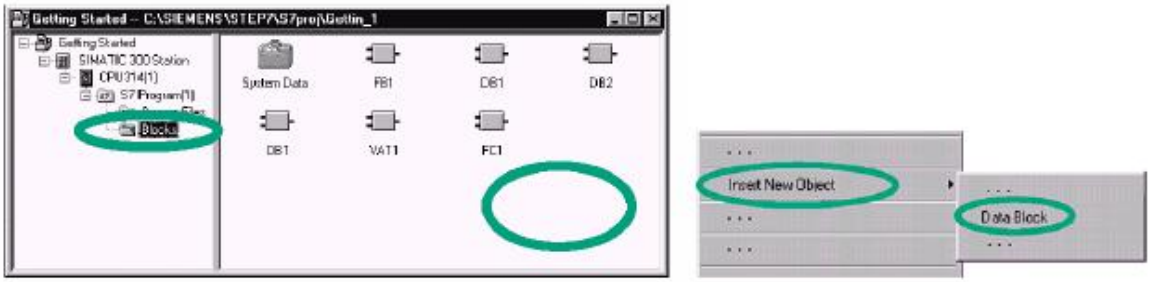
Ακόμα υπάρχουν τα:

- Πρότυπα μπλοκ δεδομένων (Instance DB). Δεσμεύονται από κάποιο FB
- UTD (User Defined Type). Αποτελούν μήτρες δημιουργίας DB.

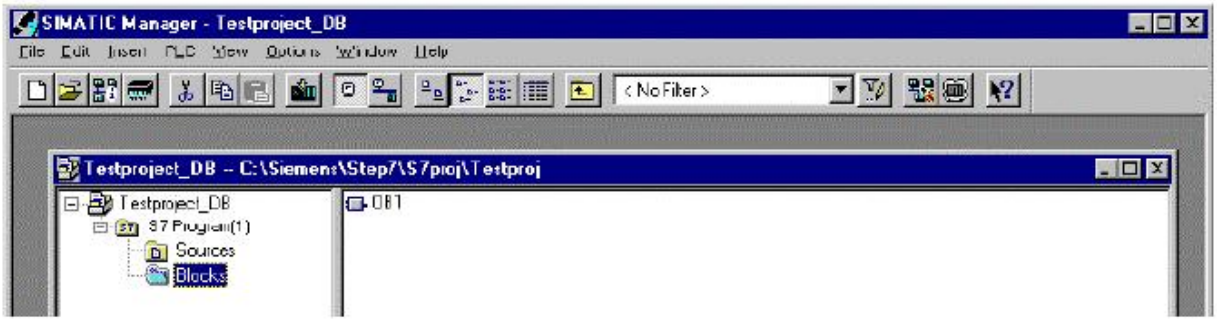
Ο αριθμός των DB εξαρτάται από τη CPU. Ο μέγιστος χώρος ενός DB στη σειρά S7-300 είναι 8Kbyte.

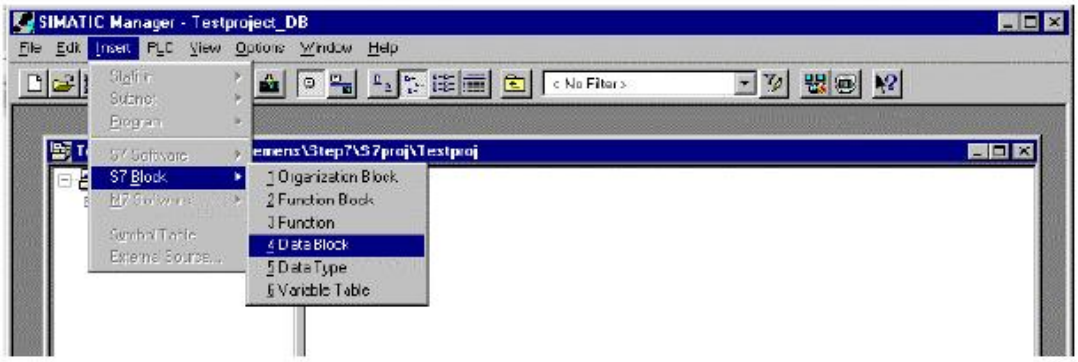
Δημιουργία DB στο Simatic Manager

Ανοίγουμε το project μας μέχρι να φτάσουμε στην ετικέτα Blocks. Στο τμήμα της οθόνης που εμφανίζονται τα blocks κάνουμε **δεξί κλικ → Insert New Object → Data Block** και ανοίγει η καρτέλα του DB.

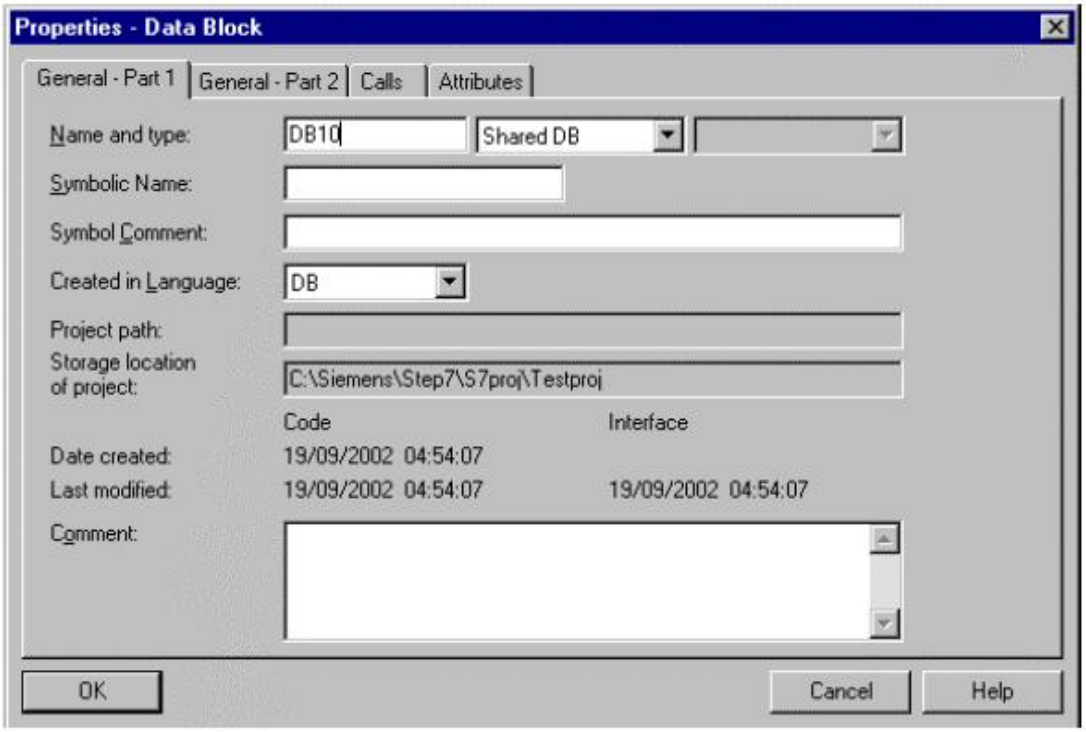


Εναλλακτικά μπορούμε να τσεκάρουμε την ετικέτα Blocks και από το μενού να επιλέξουμε **Insert → S7 Block → Data Block.**





Εκεί ονομάζουμε το DB (π.χ. DB10) και επιλέγουμε τον τύπο του : Shared DB, Instance DB ή DB of Type (UDT).



Επικυρώνουμε με O.K. και το καινούργιο DB έχει δημιουργηθεί.

ΔΟΜΗ DB

Address	Name	Type	Initial Value	Comment
0.0		STRUCT		
+0.0	PE_Actual_Speed	INT	0	Actual speed for petrol engine
+2.0	DE_Actual_Speed	INT	0	Actual speed for diesel engine
+4.0	Preset_Speed_Reached	BOOL	FALSE	Both engines have reached the preset speed
=6.0		END_STRUCT		

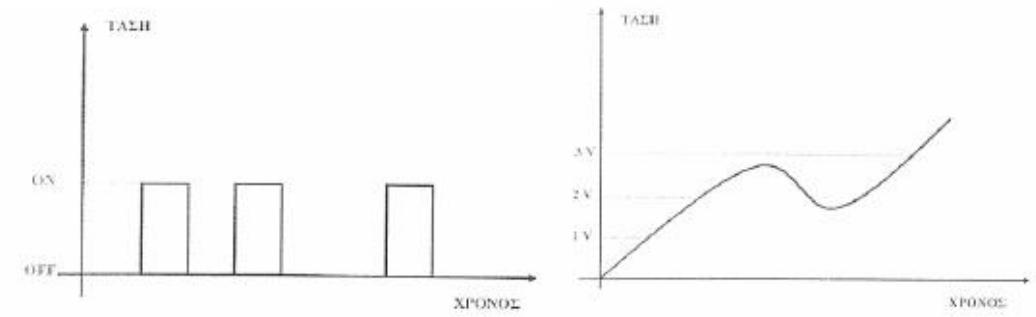
Όταν ανοίγουμε ένα DB στο πάνω μέρος του υπάρχει ο πίνακας δηλώσεων, τον οποίο συμπληρώνουμε. Σ΄αυτόν υπάρχουν οι εξής στήλες :

Address:	Συμπληρώνετε αυτόματα από τον Editor του προγράμματος και δεν μπορεί να τροποποιηθεί. Περιέχει τις απόλυτες διευθύνσεις του χώρου μνήμης ενός DB. Μέσω αυτών των διευθύνσεων αποκτάμε πρόσβαση στις πληροφορίες του DB.
Name:	Εισάγουμε συμβολικό όνομα που αντιστοιχεί στην απόλυτη διεύθυνση του χώρου μνήμης του DB.
Type:	Καθορίζουμε τον τύπο δεδομένων που θα καταχωρηθούν στον αντίστοιχο χώρο μνήμης. Εδώ μπορούμε να έχουμε : *Α)* ΑΠΛΟΥΣ ΤΥΠΟΥΣ ΔΕΔΟΜΕΝΩΝ (Καταλαμβάνουν χώρο μικρότερο από 32bit). *Β)* ΣΥΝΘΕΤΟΥΣ ΤΥΠΟΥΣ ΔΕΔΟΜΕΝΩΝ (> από 32bit).
Initial Value:	Καταχωρούμε τις αρχικές τιμές των δεδομένων του Data Block. Αν δεν συμπληρώσουμε κάποια, ο editor της δίνει την τιμή " 0 ". Το format της τιμής που καταχωρούμε θα πρέπει να είναι σε συμφωνία με τον τύπο (Type) δεδομένων που έχουμε επιλέξει.
COMMENT	Εισάγουμε σχόλια (προαιρετικό).

Ανοίγοντας ένα DB για να διαβάσουμε τις τιμές που καταχωρούνται στους χώρους μνήμης που έχουμε οργανώσει, δίπλα στη στήλη Initial Value δημιουργείται η στήλη Actual Value. Σε αυτήν τη στήλη αναγράφονται οι τρέχουσες τιμές των χώρων μνήμης.

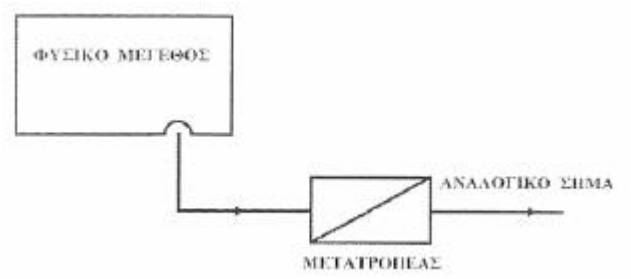
Επεξεργασία Αναλογικών Σημάτων

Σε μια παραγωγική διαδικασία υπάρχουν πολλές περιπτώσεις στις οποίες θα πρέπει να παρακολουθήσουμε την μεταβολή κάποιου φυσικού μεγέθους (π.χ ταχύτητα, πίεση, θερμοκρασία) όχι σαν καταστάσεις ON-OFF αλλά συνεχή μέτρηση τιμών. Αυτά λέγονται αναλογικά μεγέθη. Στην κάτω εικόνα παρουσιάζονται αναλογικά και ψηφιακά σήματα.

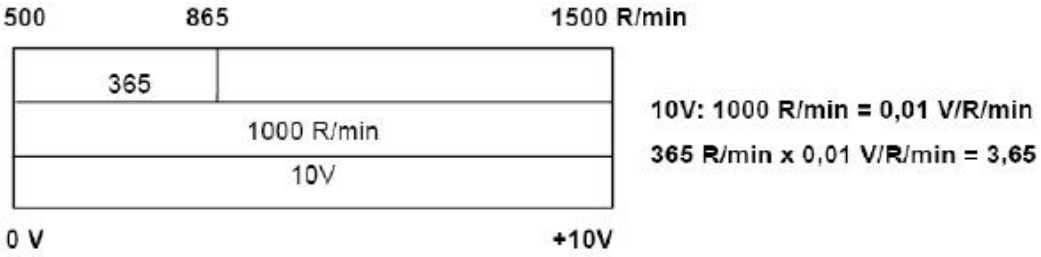


Για να πάρουμε ένα αναλογικό σήμα χρειαζόμαστε

a. **Αισθητήριο μέτρησης:** είναι το όργανο εκείνο το οποίο μετατρέπει τις αλλαγές ενός φυσικού μεγέθους σε μια γραμμική μεταβολή κάποιας ιδιότητας του αισθητήρα (π.χ. μεταβολή ηλεκτρικής επαγωγής ή….)
b. **Μετατροπέα:** Ο μετατροπέας συνδέεται με το αισθητήριο μέτρησης και είναι σε θέση να παρακολουθήσει τις μεταβολές του αισθητήρα και να τις μετατρέψει στο κατάλληλο πρότυπο ηλεκτρικό σήμα (0-10V , 4-20mA….)



Π.χ. σε ένα σύστημα μέτρησης περιστροφών, ο μετατροπέας μπορεί να μετατρέψει ταχύτητες εύρους από 500 ως 1500 Rounds/min σε ηλεκτρικό σήμα τάσης από 0 ως +10V. Δηλαδή όταν μετράει 865R/min θα δίνει ηλεκτρική έξοδο στάθμης +3,65V.



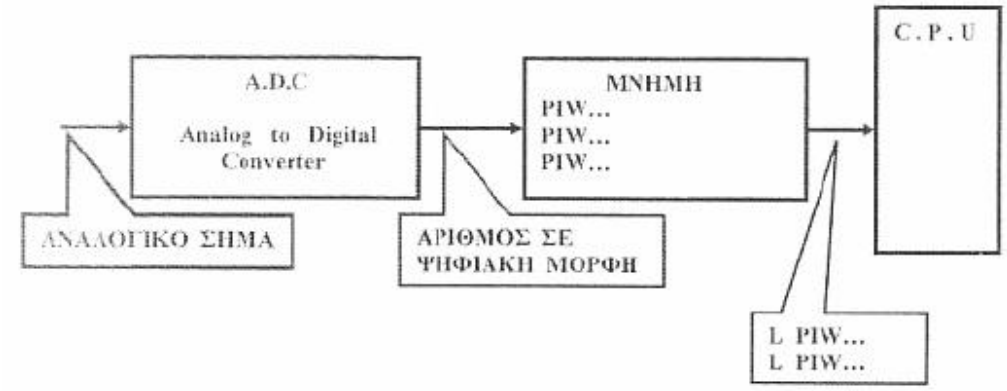
Όταν θέλουμε να επεξεργαστούμε ένα αναλογικό σήμα η τιμή της τάσης, ρεύματος ή αντίστασης της εξόδου του μετατροπέα πρέπει να μετατραπεί σε ψηφιακή πληροφορία. Αυτή η διαδικασία ονομάζεται **A/D conversion** και είναι απαραίτητη καθώς η CPU μπορεί να διαχειριστεί μόνο ψηφιακές πληροφορίες.

Αυτό σημαίνει ότι η τιμή τάσης π.χ. 3,65V θα πρέπει να αποθηκευθεί ως ψηφιακή πληροφορία σε μία σειρά διαδοχικών δυαδικών ψηφίων. Όσο περισσότερα χρησιμοποιούνται για να απεικονίσουν ψηφιακά την τιμή τόσο καλύτερη είναι η διακριτική ικανότητα.

Αυτόν τον ρόλο αναλαμβάνουν να τον κάνουν οι λεγόμενες αναλογικές κάρτες οι οποίες κάνουν συνήθως τη μετατροπή σε 8-bit ή 16-bit . Οι αναλογικές κάρτες εισόδου λοιπόν αναλαμβάνουν τον ρόλο να μετατρέψουν τις διάφορες τιμές των αναλογικών σημάτων σε αριθμούς με ψηφιακή μορφή και αυτούς τους αριθμούς τους αποθηκεύουν σε μια ξεχωριστή περιοχή μνήμης για τα αναλογικά σήματα εισόδου. Η περιοχή αυτή χαρακτηρίζεται με τα γράμματα **PIW** (Peripheral Input Word).

Από αυτήν την περιοχή η CPU είναι σε θέση να διαβάσει την τιμή μιας αναλογικής εισόδου με την εντολή “L” (Load).

Η αναλογική κάρτα εισόδου είναι λοιπόν εξοπλισμένη με :



Οι αναλογικές κάρτες εξόδου αντίθετα αναλαμβάνουν τον ρόλο να μετατρέψουν έναν αριθμό (ψηφιακή μορφή) σε ένα αναλογικό σήμα. Τον αριθμό σε ψηφιακή μορφή τον διαβάζουν από μια ειδική περιοχή μνήμης η οποία χαρακτηρίζεται με τα γράμματα PQW (Peripheral Output Word). Η περιοχή αυτή ενημερώνει από την CPU με την εντολή “T” (Transfer).

Η αναλογική κάρτα εξόδου είναι εξοπλισμένη με :

Ένα μέγεθος που μας ενδιαφέρει στις αναλογικές κάρτες είναι η διακριτική τους ικανότητα με άλλα λόγια το φάσμα των ψηφιακών αριθμών που χρησιμοποιεί για την μετατροπή του αναλογικού σήματος. Όσο μεγαλύτερο είναι αυτό το φάσμα τόσο μεγαλύτερη ακρίβεια έχουμε στην μετατροπή. Στην περίπτωση της CPU314C-2DP με τις ενσωματωμένες αναλογικές εισόδους – εξόδους έχουμε δυνατότητα να επεξεργαστούμε αναλογικά σήματα 0 .10 V, +/- 10 V, 0 .20 mA, 4.20 mA.

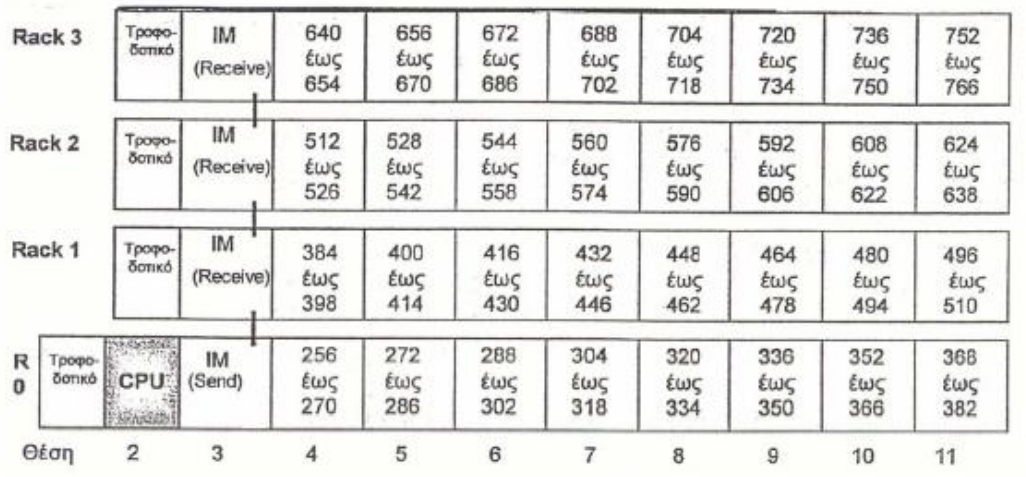
Το φάσμα των ψηφιακών αριθμών που χρησιμοποιείται στην γραμμική περιοχή είναι από – 27648 έως +27648 και ο **χώρος που καταλαμβάνει κάθε αναλογική διεύθυνσή στην μνήμη είναι 16bit ή 1 Word.**

ΔΙΕΥΘΥΝΣΕΙΣ ΤΩΝ ΑΝΑΛΟΓΙΚΩΝ ΣΗΜΑΤΩΝ

Τα PLC της σειράς S7-300 έχουν ξεχωριστή περιοχή διευθύνσεως για τις αναλογικές εισόδους και εξόδους. Οι αναλογικές διευθύνσεις αποδίδονται αυτόματα στις αναλογικές κάρτες ανάλογα από τις θέσεις στις οποίες αυτές είναι τοποθετημένες επάνω στο rack. Η πρώτη αναλογική διεύθυνση που αποδίδεται είναι η 256 (αν η κάρτα είναι τοποθετημένη στην slot 4). Φυσικά αυτές τις τιμές έχουμε την δυνατότητα να τις αλλάξουμε μέσα από το hardware configuration. Εκείνο που πρέπει να θυμόμαστε σχετικά με την διεύθυνση των αναλογικών εισόδων ή εξόδων είναι :

- Κάθε αναλογικό κανάλι καταλαμβάνει χώρο δυο byte

- Κάθε slot θέση πάνω στο rack μιας αναλογικής κάρτας καταλαμβάνει χώρο για οκτώ (8) αναλογικά κανάλια. Βάση αυτών των δυο καναλιών σε ένα πλήρη ανεπτυγμένο σύστημα της σειράς S7-300 η διευθυνσιοδότηση των αναλογικών σημάτων βάση της slot θέσης δίνεται από τον πίνακα που ακολουθεί.

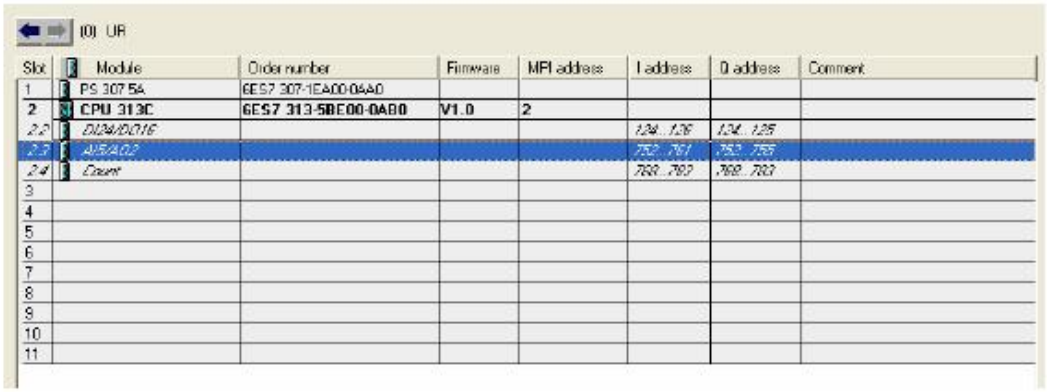


C. P. U 314C-2DP

Στην περίπτωση της εφαρμογής που έχουμε στην διάθεση μας τα χαρακτηριστικά των αναλογικών σημάτων που διαθέτουμε είναι :

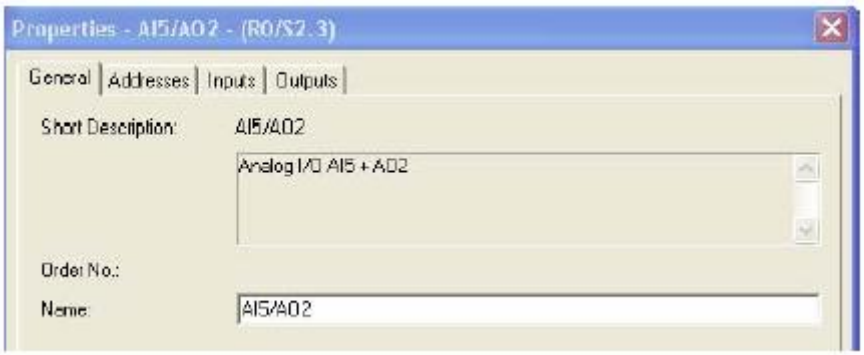
ΠΕΡΙΟΧΕΣ ΜΕΤΡΗΣΗΣ	ΑΝΑΛΟΓΙΚΕΣ ΕΙΣΟΔΟΙ	ΑΝΑΛΟΓΙΚΕΣ ΕΞΟΔΟΙ
ΤΑΣΗ	+/- 10 V ή 0-10 V	+/- 10 V ή 0-10 V
ΡΕΥΜΑ	+/- 20mA ή 0/4-20mA	+/- 20mA ή 0/4-20mA
ΑΝΑΛΥΣΗ	11 bits + sign	11 bits + sign

Έχοντας στήσει ένα νέο project και έχοντας κάνει HARDWARE CONFIGURATION μπορούμε να εντοπίσουμε τις υπάρχουσες αναλογικές εισόδους -εξόδους με τις μητρικές τους διευθύνσεις . Η εικόνα κάτω μας δείχνει τα όσα αναφέραμε προηγουμένως.

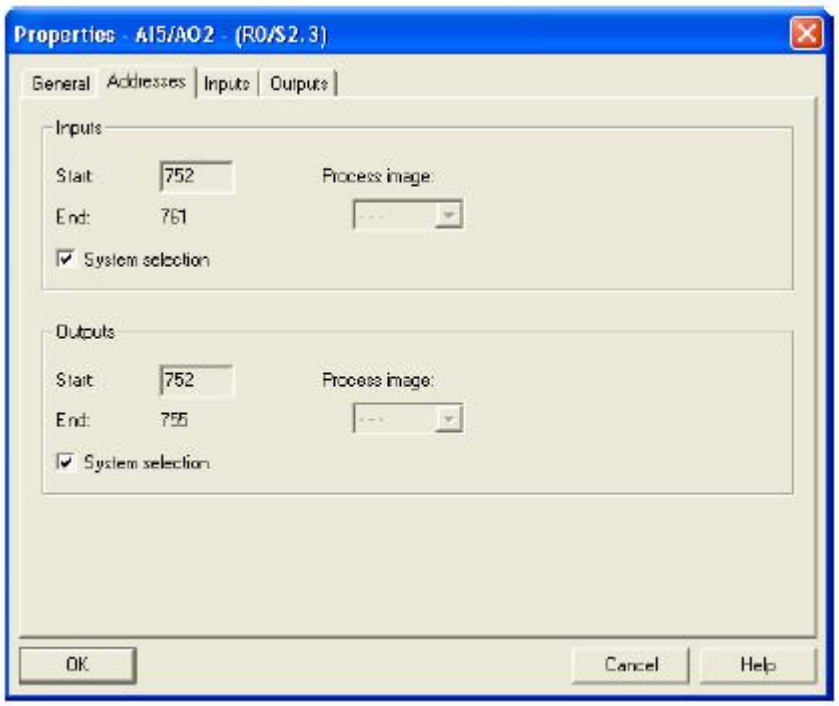


Ρυθμίσεις (από HARDWARE CONFIGURATION)

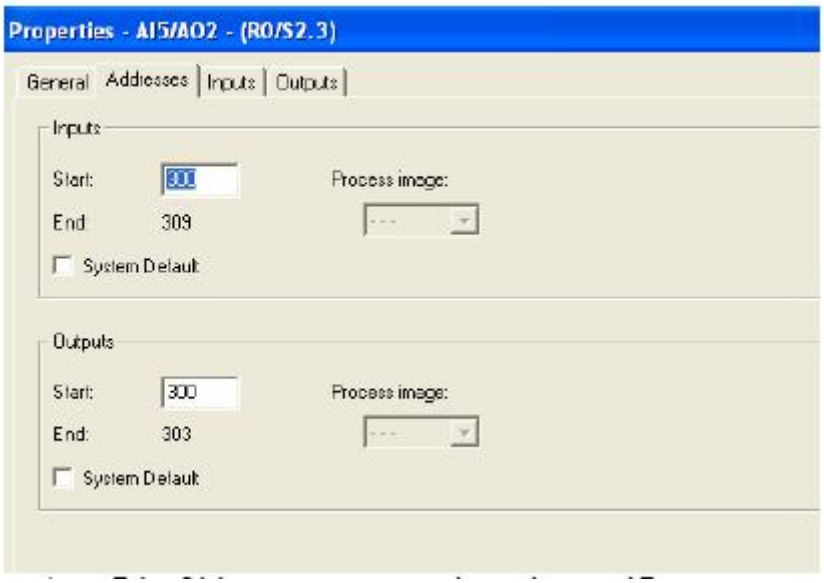
Μπαίνουμε με διπλό κλικ στις ιδιότητες της αναλογικής κάρτας



Εάν επιλέξουμε την καρτέλα Addresses θα έχουμε την παρακάτω εικόνα .



Απενεργοποιώντας την επιλογή system selection μπορούμε να αλλάξουμε τις λογικές διευθύνσεις που έχει δώσει αρχικά το σύστημα, με αυτές της επιλογής μας.
Την τελευταία διαθέσιμη αναλογική διεύθυνση την συμπληρώνει μόνο του το σύστημα.



Καρτέλα Inputs εδώ βλέπουμε τις αναλογικές εισόδους και μπορούμε να επέμβουμε στα εξής σημεία .

a) **Measurement type :** Επιλογή για το εάν πρόκειται για αναλογική είσοδο τάσης η έντασης η εάν θέλουμε να απενεργοποιήσουμε την συγκεκριμένη είσοδο.
b) **Measuring range :** Επιλογή του εύρους της περιοχής μέτρησης του αναλογικού σήματος.
c) **Interference frequency :** Επιλογή συχνότητας φίλτρου

- Ομοίως για τις αναλογικές Εξόδους

ΔΙΑΒΑΣΜΑ ΑΝΑΛΟΓΙΚΗΣ ΤΙΜΗΣ ΕΙΣΟΔΟΥ – ΓΡΑΨΙΜΟ ΑΝΑΛΟΓΙΚΗΣ ΤΙΜΗΣ ΕΞΟΔΟΥ

Οι αναλογικές τιμές εισάγονται στο PLC ως πληροφορίες που καταλαμβάνουν μέγεθος 1word=16bit. Συνεπώς, η πρόσβαση σε αυτή τη word γίνεται με τις εντολές Load και Transfer (ή Move στη Ladder):

L PIWx: Φόρτωσε την αναλογική λέξη εισόδου x

T PQWx : Μετέφερε την αναλογική λέξη εισόδου x

Κάθε αναλογική τιμή (''κανάλι'') αναθέτεται σε μία περιφερειακή λέξη εισόδου ή εξόδου (PIW,PQW). Το format της είναι τύπου integer (INT).

Δηλαδή μία αναλογική κάρτα διαβάζει την αναλογική τιμή ρεύματος ή τάσης (0- 10V, 4-20mA) που προέρχεται από τον μετρητικό μετατροπέα και την ψηφιοποιεί δημιουργώντας ένα INT αριθμό από 0 ως 27648.

Αν αυτή την ψηφιακή τιμή πρέπει να την επεξεργαστούμε περαιτέρω με το PLC τότε πρέπει να την κανονικοποιήσουμε (normalize). Αυτό μπορεί να γίνει στην Step7 με δύο τρόπους :

- Με χρήση μαθηματικών πράξεων
- Με χρήση των έτοιμων συναρτήσεων FC105,FC106

8. Βιβλιογραφία

Πηγές που βοήθησαν στην πτυχιακή μας :

- **Αυτοματισμός με Χρήση PLC**
  Ιωάννης Μπερέτας
  Εκδόσεις ΤΖΙΟΛΑ

- **Προγραμματιζόμενοι Λογικοί Ελεγκτές**
  Νικ. Α. Πανταζής
  Εκδόσεις Ιων

- **Εφαρμογές Αυτοματισμών με PLC's**
  Τζουνίδης Θ. Γεώργιος

- **Συστήματα Αυτοματισμών Β' Τόμος**
  Ο.Ε.Δ.Β. Αθήνα

- **Από το Διαδίκτυο (Internet)**