ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Προγραμματιζόμενοι Λογικοί Ελεγκτές και Εφαρμογές σε Βιομηχανικούς Αυτοματισμούς

Διπλωματική Εργασία

Ντούλας Μάριος

Επιβλέπων: Σταμούλης Γεώργιος

Ιούλιος 2024

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

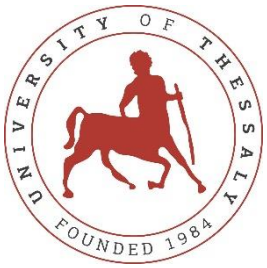
Προγραμματιζόμενοι Λογικοί Ελεγκτές και Εφαρμογές σε Βιομηχανικούς Αυτοματισμούς

Διπλωματική Εργασία

Ντούλας Μάριος

Επιβλέπων: Σταμούλης Γεώργιος

Ιούλιος 2024



UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Programmable Logic Controllers and Applications in Industrial Automation

Diploma Thesis

Ntoulas Marios

Supervisor: Stamoulis Georgios

July 2024

Εγκρίνεται από την Επιτροπή Εξέτασης:

Επιβλέπων	**Σταμούλης Γεώργιος** Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας
Μέλος	**Μπαργιώτας Δημήτριος** Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας
Μέλος	**Πλέσσας Φώτιος** Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΑΚΑΔΗΜΑΪΚΗΣ ΔΕΟΝΤΟΛΟΓΙΑΣ ΚΑΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα διπλωματική εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελούν αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλουν οποιασδήποτε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχουν έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Δηλώνω επίσης ότι τα αποτελέσματα της εργασίας δεν έχουν χρησιμοποιηθεί για την απόκτηση άλλου πτυχίου. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο Δηλών

Ντούλας Μάριος

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism.

The Declarant

Ntoulas Marios

Ευχαριστίες

Στο σημείο αυτό θα ήθελα να ευχαριστήσω τους καθηγητές Σταμούλη Γεώργιο, Μπαργιώτα Δημήτριο και Πλέσσα Φώτιο για την ευκαιρία που μου έδωσαν να υλοποιήσω την εν λόγω διπλωματική εργασία.

Επιπλέον θερμές ευχαριστίες στην οικογένεια και τους φίλους μου για τη σημαντική στήριξη που μου παρείχαν σε όλη τη διάρκεια των σπουδών μου.

Διπλωματική Εργασία

Προγραμματιζόμενοι Λογικοί Ελεγκτές και εφαρμογές σε βιομηχανικούς αυτοματισμού

Ντούλας Μάριος

Περίληψη

Η διπλωματική εργασία αυτή θα εστιάσει στους βιομηχανικούς αυτοματισμούς με χρήση της τεχνολογίας των Προγραμματιζόμενων Λογικών Ελεγκτών. Στόχος μας είναι να παρουσιάσουμε αυτό το θέμα έτσι ώστε, μέχρι το τέλος της παρούσας εργασίας, ο αναγνώστης να αποκτήσει όλες τις απαραίτητες γνώσεις για το θέμα στο βαθμό που πλέον γνωρίζει τη σημασία της τεχνολογίας αυτής και κατανοεί τον τρόπο λειτουργίας και προγραμματισμού. Συγκεκριμένα θα πραγματοποιήσουμε μία εισαγωγή στους αυτοματισμούς εξετάζοντας πως αυτοί εξελίχθηκαν. Εμβαθύνοντας, στη συνέχεια, θα μελετήσουμε τους Προγραμματιζόμενους Λογικούς Ελεγκτές αναλύοντας τη δομή και τον τρόπο λειτουργίας αυτών και αφού εστιάσουμε στα μοντέλα της εταιρείας SIEMENS θα μάθουμε πως μπορούμε να τα προγραμματίζουμε κάνοντας χρήση του λογισμικού της ιδίας εταιρείας με την γλώσσα προγραμματισμού Ladder Logic. Τέλος θα εφαρμόσουμε τις αποκτημένες πλεόν γνώσεις σε πρακτικά παραδείγματα δημιουργόντας προγράμματα και αυτοματισμούς που ανταποκρίνονται σε σύγχρονες βιομηχανικές εφαρμογές.

Λέξεις-κλειδιά:

Βιομηχανικοί Αυτοματισμοί, Προγραμματιζόμενος Λογικός Ελεγκτής, PLC, Ladder Logic, TIA Portal, SIMATIC

Diploma Thesis

Programmable Logic Controllers and applications in industrial automation

Ntoulas Marios

Abstract

This thesis will focus on industrial automation using Programmable Logic Controllers technology. Our goal is to present this topic so that, by the end of this paper, the reader will have all the necessary knowledge on the subject to the extent that he now knows the importance of this technology and understands how it works and can be programmed. In particular, we will make an introduction to automations, examining how they evolved. Going deeper, then we will study the Programmable Logic Controllers analyzing their structure and operation and after focusing on the models of SIEMENS we will learn how we can program them using the same company's software with the use of Ladder Logic programming language. Finally, we will apply the acquired knowledge in practical examples, creating programs and automations that respond to modern industrial applications.

Keywords: Industrial Automation, Programmable Logic Controller, PLC, Ladder Logic, TIA Portal, SIMATIC

Πίνακας περιεχομένων

Κατάλογος εικόνων

Κατάλογος πινάκων

Κεφάλαιο 1 Εισαγωγή

1.1 Αντικείμενο της παρούσας διπλωματικής εργασίας

Στην παρούσα εργασία θα μελετήσουμε τους Προγραμματιζόμενους Λογικούς Ελεγκτές και την συνεισφορά τους στον τομέα των συστημάτων βιομηχανικών αυτοματισμών. Αρχικά θα κάνουμε μία εισαγωγή στους βιομηχανικούς αυτοματισμούς και την εξέλιξη αυτών και έπειτα θα αναλύσουμε τη δομή και τη λειτουργία ενός Προγραμματιζόμενου Λογικού Ελεγκτή ώστε να κατανοήσουμε τη σημασία της τεχνολογίας αυτής. Τέλος, αφού πρώτα εστιάσουμε στα μοντέλα και το περιβάλλον εργασίας της εταιρείας SIEMENS καθώς και στην γλώσσα προγραμματισμού Ladder Logic θα παρουσιάσουμε ενδεικτικές μελέτες και παραδείγματα τα οποία έχουν άμεση πρακτική εφαρμογή στον τομέα του βιομηχανικού αυτοματισμού.

1.2 Οργάνωση του τόμου

Τα επόμενα κεφάλαια της διπλωματικής εργασίας είναι οργανωμένα ως εξής:

Κεφάλαιο 2: Στο συγκεκριμένο κεφαλαίο γίνεται μια εισαγωγική προσέγγιση των βιομηχανικών αυτοματισμών μελετώντας τα στάδια εξέλιξης της βιομηχανίας και τους λόγους αναγκαιότητας των συστημάτων αυτών.

Κεφάλαιο 3: Στο συγκεκριμένο κεφαλαίο θα γίνει μια εισαγωγή στους Προγραμματιζόμενους Λογικούς Ελεγκτές, ώστε να κατανοήσουμε τον τρόπο με τον οποίο αυτοί λειτουργούν.

Κεφάλαιο 4: Στο συγκεκριμένο κεφαλαίο θα εστιάσουμε στη σειρά από Προγραμματιζόμενους Λογικούς Ελεγκτές της εταιρείας SIEMENS και στη συνέχεια θα παρουσιάσουμε το περιβάλλον προγραμματισμού το οποίο παρέχεται από την εταιρεία αυτή για τις συσκευές της.

Κεφάλαιο 5: Στο συγκεκριμένο κεφαλαίο θα γίνει μία εισαγωγή στην γλώσσα προγραμματισμού Ladder Logic εξετάζοντας βασικά της στοιχεία έτσι ώστε να είμαστε πλέον σε θέση να προγραμματίσουμε τους πρώτους αυτοματισμούς μας.

Κεφάλαιο 6: Στο συγκεκριμένο κεφαλαίο θα κάνουμε χρήση των γνώσεων που αποκτήσαμε σε προηγούμενα κεφάλαια εφαρμόζοντάς τες ώστε να υλοποιήσουμε διαδικασίες αυτοματισμών για διάφορες βιομηχανικές λειτουργίες.

Κεφάλαιο 7: Στο συγκεκριμένο κεφαλαίο θα διατυπώσουμε τα συμπεράσματά μας καθώς και ιδέες με τις οποίες θα μπορούσαμε να εξελίξουμε την έρευνα που πραγματοποιήσαμε.

Κεφάλαιο 2 Αυτοματισμοί και βιομηχανία

2.1 Διαδικασία, αυτοματισμός και έλεγχος

Σύμφωνα με τον ορισμό της, μια διαδικασία είναι μια ακολουθία βημάτων που λαμβάνονται για την επίτευξη ενός συγκεκριμένου στόχου. Στον τομέα της βιομηχανίας οι διαδικασίες μετατρέπουν τις πρώτες ύλες σε ολοκληρωμένα προϊόντα. Στη βιομηχανία όμως ο πιο κρίσιμος παράγοντας οποιασδήποτε διαδικασίας είναι η παραγωγικότητα με εγγυημένη ποιότητα. Έτσι στον τομέα αυτό, οι τεχνικές αυτοματισμού χρησιμοποιούνται με αυξανόμενους ρυθμούς σε μια πληθώρα λειτουργιών για τη βελτίωση της απόδοσης διατηρώντας παράλληλα το απαραίτητο επίπεδο ποιότητας. Τι είναι όμως ένα σύστημα αυτοματισμού; Θα μπορούσαμε να ορίσουμε τον αυτοματισμό ως ένα σύνολο τεχνολογιών που επιτρέπουν στα μηχανήματα και τα συστήματα να λειτουργούν με ελάχιστη ανθρώπινη παρέμβαση και να επιτυγχάνουν την επιθυμητή απόδοση η οποία είναι καλύτερη από αυτή της χειροκίνητης λειτουργίας. Τέλος, αυτό που απαιτείται σε κάθε στάδιο μίας βιομηχανικής διαδικασίας έτσι ώστε αυτή να εκτελεστεί κατά τον προβλεπόμενο τρόπο είναι ο έλεγχος. Πρόκειται για μια μια συλλογή από στρατηγικές, όπου παρέχοτας όλες τις απαιτούμενες πληροφορίες βοηθούν στην επίτευξη των επιδιωκόμενων αλλαγών [1].

2.2 Εξέλιξη της Βιομηχανίας

Στις απαρχές του πολιτισμού μας, όλες οι βιομηχανικές διεργασίες έπρεπε να εκτελούνται εξ ολοκλήρου με χρήση χειρωνακτικής εργασίας κάτω από οποιεσδήποτε συνθήκες. Τα συστήματα ελέγχου διεργασιών εφαρμόστηκαν αρχικά σε πνευματικά συστήματα και τοποθετήθηκαν σε κοντινή απόσταση με τα πράγματα που έπρεπε να διαχειριστούν. Με την πάροδο των χρόνων ο βιομηχανικός αυτοματισμός και ο έλεγχος διεργασιών έχουν εξελιχθεί ραγδαία [2].

Η Πρώτη Βιομηχανική Επανάσταση ξεκίνησε τον 18ο αιώνα με τη χρήση της ατμομηχανής και την ανάπτυξη της μηχανοποίησης της παραγωγής. Εφευρέσεις όπως ατμόπλοια και αμαξοστοιχίες έφεραν τεράστιες αλλαγές αφού πλέον μπορούσαμε να πραγματοποιήσουμε μεταφορές σε λιγότερο χρονικό διάστημα. Κατά τον 19ο αιώνα έχουμε τη Δεύτερη Βιομηχανική Επανάσταση με την ανακάλυψη της ηλεκτροδότησης και των γραμμών παραγωγής. Πλέον με τη χρήση των γραμμών παραγωγής ο κάθε εργάτης κάνει ένα τμήμα της συναρμολόγησης, άρα επιτυγχάνουμε γρηγορότερη και με χαμηλότερο κόστος εργασία [1]. Μετά τον Δεύτερο Παγκόσμιο Πόλεμο, ο βιομηχανικός εκσυγχρονισμός οδήγησε στην ανάπτυξη της ηλεκτρονικής, η οποία σηματοδότησε την αρχή της Τρίτης Βιομηχανικής Επανάστασης, επηρεασμένη σε μεγάλο βαθμό από τη ρομποτική, την τεχνολογία πληροφοριών και τις τηλεπικοινωνίες. Η έρευνα σε αυτούς τους τομείς έχει αλλάξει ολόκληρο το σύστημα παραγωγής με στόχο την παραγωγή περισσότερων σε λιγότερο χρόνο [2].

Σήμερα βρισκόμαστε στα πρώιμα στάδια της εποχής Industry 4.0. Ως αποτέλεσμα της ταχύτατης ανάπτυξης του διαδικτύου και των τεχνολογιών επικοινωνίας, η οποία άλλαξε εντελώς τον τρόπο με τον οποίο οι άνθρωποι ανταλλάσσουν πληροφορίες, η τεχνολογία βιομηχανικού αυτοματισμού που χρησιμοποιείται στον κατασκευαστικό τομέα υπέστη μεγάλες αλλαγές. Πλέον οι δραστηριότητες παραγωγής στον πραγματικό και στον εικονικό κόσμο συνδέονται, με την είσοδο “έξυπων” μηχανημάτων και Cyber-Physical συστημάτων, τα οποία επιτρέπουν την πλήρη εικονική βιομηχανική παρακολούθηση και διαχείριση [1].

2.3 Πλεονεκτήματα και στόχοι βιομηχανικών αυτοματισμών

Αξιοποιώντας τις τεχνολογίες αυτοματισμού, οι βιομηχανικές επιχειρήσεις σε όλους τους κλάδους βιώνουν γρήγορη αύξηση στην παραγωγικότητα τους, αφού ο αυτοματισμός έχει εφαρμογές σε όλες τις μορφές παραγωγής. Βασικά πλεονεκτήματα της χρήσης αυτοματισμών είναι τα παρακάτω:

- Αύξηση παραγωγικότητας
- Μείωση του κόστους παραγωγής
- Βελτίωση της ποιότητας των προϊόντων

- Αποτελεσματικός έλεγχος διαδικασιών
- Μεγαλύτερη ασφάλεια καλύτερες συνθήκες εργασίας.

Ορισμένοι από τους κύριους στόχους της εφαρμογής αυτοματισμών στην βιομηχανία είναι οι εξής:

- Ενσωμάτωση διαφορετικών στοιχείων των διαδικασιών παραγωγής με σκοπό την μείωση του χρόνου και κόστους παραγωγής και την αύξηση της ποιότητας παραγωγής.
- Αύξηση παραγωγικότητας μέσω καλύτερου ελέγχου. Τα μηχανήματα χρησιμοποιούνται πιο αποτελεσματικά και η παραγωγή οργανώνεται με τον βέλτιστο τρόπο.
- Μείωση της ανάμιξης ανθρώπινου παράγοντα, κατά συνέπεια και της πιθανότητας ανθρώπινου σφάλματος.
- Μείωση του χώρου παραγωγής μέσω βελτιστοποίησης της διαρρύθμισης τοποθέτησης μηχανημάτων και εξοπλισμού [3].

2.4 Κατηγορίες βιομηχανικών αυτοματισμών

Τα συστήματα βιομηχανικού αυτοματισμού μπορούν να ταξινομηθούν σε τρεις βασικές κατηγορίες, ανάλογα με τον βαθμό ενσωμάτωσης και προσαρμοστικότητάς τους:

Fixed Automation

Στην κατηγορία αυτή οι ακολουθίες λειτουργιών εκτελούνται από την αυτοματοποιημένη διαμόρφωση του εξοπλισμού. Οι διαδικασίες είναι συχνά απλές αλλά το σύστημα μπορεί να καθίσταται πολύπλοκο στην προσπάθεια να συντονίσουμε πολλές από αυτές τις διαδικασίες. Ο τύπος αυτός είναι οικονομικά βιώσιμος εάν υπάρχει σημαντική αύξηση στην ζήτηση των προϊόντων για παρατεταμένη χρονική περίοδο.

Programmable Automation

Αυτή η κατηγορία αυτοματισμού προσφέρει την ελευθερία τροποποίησης της σειράς των βημάτων για να ταιριάζει σε διάφορα σχέδια προϊόντων. Μία σειρά οδηγιών που μπορούν

να αλλάξουν και να οργανωθούν σύμφωνα με τις ανάγκες που διέπουν τη σειρά λειτουργίας. Συνήθως, τα προϊόντα παράγονται σταδιακά. Κάθε φορά που παράγεται μια νέα παρτίδα, το σύστημα πρέπει να επαναπρογραμματίζεται. Έχουμε τελικά δύο περιόδους: μια για το στήσιμο και την άλλη για την παραγωγή του σταδίου.

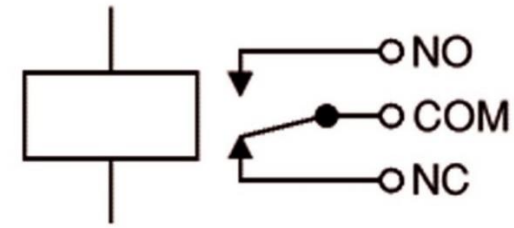
Flexible Automation

Πρόκειται για μία προέκταση του programmable automation με μεγαλύτερο βαθμό ευελιξίας. Με τον επαναπρογραμματισμό της φυσικής διαμόρφωσης δίνεται η δυνατότητα στο σύστημα να δημιουργεί διαφορετικούς συνδυασμούς προϊόντων αντί να χωρίζει σε στάδια χωρίς να χάνει χρόνο παραγωγής. Συνήθως, για την τροποποίηση ενός προγράμματος, πρέπει πρώτα να αναπτυχθεί εκτός σύνδεσης σε υπολογιστή και στη συνέχεια να μεταφερθεί ηλεκτρονικά στη διαδικασία αυτοματισμού. Σε ότι αφορά την φυσική διάταξη του εξοπλισμού αυτή αλλάζει ταυτόχρονα καθώς το ακόλουθο στοιχείο τοποθετείται για επεξεργασία [1].

2.5 Από το Relay στο PLC

Όπως κάθε τεχνολογία περνάει από διάφορα στάδια και συνεχώς εξελίσσεται έτσι και τα συστήματα βιομηχανικών αυτοματισμών δεν θα μπορούσαν να αποτελούν εξαίρεση στον κανόνα αυτόν.

Με την ηλεκτροδότηση των βιομηχανιών τον 20ο αιώνα, έχουμε την είσοδο των Ηλεκτρονόμων (Relays) στα εργοστάσια [4]. Relay (Εικόνα 2.1) είναι ένας διακόπτης που ενεργοποιείται με ηλεκτρική ενέργεια αποτελούμενος από ένα σύνολο τερματικών επαφών και ένα σύνολο ακροδεκτών εισόδου για ένα ή περισσότερα σήματα ελέγχου. Στην κλασική του μορφή χρησιμοποιεί ηλεκτρομαγνήτη για το κλείσιμο ή το άνοιγμα των συνδέσεων. Άλλες ιδέες λειτουργίας έχουν επίσης οδηγήσει στην εφεύρεση άλλων τύπων όπως για παράδειγμα είναι το relay το οποίο βασίζεται σε χαρακτηριστικά ημιαγωγών για έλεγχο αντί για κινούμενα στοιχεία [5]. Τα συστήματα που ελέγχονται με χρήση relays υλοποιούνται με τη μορφή μόνιμων ηλεκτρονικών κυκλωμάτων (hard-wired systems). Όταν εφαρμόζεται ρεύμα ανοίγουν οι Normally Closed Contacts (NC) και κλείνουν οι Normally Open Contacts (NO) παρέχοντας έτσι έλεγχο σε κάποιο σύστημα μέσω της εξόδου του relay Coil (Εικόνα 2.2) [6].



Εικόνα 2.1 Διάγραμμα Ηλεκτρονόμου (Relay Diagram) [7]

Εικόνα 2.2 Συμβολισμοί επαφών (Contacts Symbols)

Το πρόβλημα με την τεχνολογία αυτή ήταν πως σε μια γραμμή παραγωγής οποιεσδήποτε αλλαγές γίνουν σε λογικό κύκλωμα ενσύρματου relay θα ήταν εξαιρετικά δαπανηρές και χρονοβόρες και θα συνεπάγονταν πολλούς περιορισμούς τόσο για τους τεχνικούς όσο και για τους μηχανικούς. Έτσι κατά τη δεκαετία 1960 η εταιρεία General Motors πρότεινε ένα προσχέδιο για έναν εναλλακτικό ελεγκτή αυτοματισμών για να παρακάμψει τα προβλήματα αυτά. Μεταξύ εταιρειών και μηχανικών που εργάστηκαν πάνω σε αυτό ο Richard Morley, ο οποίος θεωρείται και ο “πατέρας” των Προγραμματιζόμενων Λογικών Ελεγκτών συνέβαλε δραστικά στη σχεδίασή του καθώς και στην σχεδίαση της γλώσσας προγραμματισμού Ladder Logic [1].

Κεφάλαιο 3 Προγραμματιζόμενος Λογικός Ελεγκτής

3.1 Τι είναι ο Προγραμματιζόμενος Λογικός Ελεγκτής;

Ένας Προγραμματιζόμενος Λογικός Ελεγκτής(Programmable Logic Controller - PLC) είναι ένα εξειδικευμένο σύστημα βιομηχανικού ελέγχου που παρακολουθεί συνεχώς την κατάσταση συσκευών εισόδου και λαμβάνει αποφάσεις με βάση ένα προσαρμοσμένο πρόγραμμα για τον έλεγχο της κατάστασης των συσκευών εξόδου. Στην ουσία πρόκειται για μία υπολογιστική μονάδα σχεδιασμένη για έλεγχο του μηχανικού εξοπλισμού μιας βιομηχανίας. Βασίζεται σε ψηφιακή λογική, μπορεί να προγραμματιστεί απευθείας στο χώρο που βρίσκεται εγκατεστημένο και είναι κατασκευασμένο για να λειτουργεί και να αλληλεπιδρά με μία πληθώρα συσκευών εισόδου και εξόδου. Τέλος να σημειωθεί πως είναι σχεδιασμένο να ανταποκρίνεται και να αντέχει στις όποιες αντίξοες συνθήκες περιβάλλοντος που μπορεί να παρουσιαστούν σε έναν βιομηχανικό χώρο όπως για παράδειγμα κραδασμοί ή υψηλές θερμοκρασίες [8].

3.2 Η δομή ενός PLC

Όλα τα συστήματα PLC αποτελούνται από τα ιδία βασικά δομικά στοιχεία. Αυτά είναι τα εξής:

Η Βάση Εγκατάστασης (Rack)

Η πλειοψηφία των PLC κατασκευάζεται με συνδυασμό διαφορετικών μονάδων. Η βάση εγκατάστασης συγκρατεί τις μονάδες αυτές. Επίσης χρησιμοποιεί μια πλακέτα τυπωμένου κυκλώματος στο πίσω μέρος προκειμένου να παρέχει ηλεκτρικές συνδέσεις μεταξύ των μονάδων εκτός από τη σταθερή συγκράτηση τους στη θέση τους. Έτσι όταν οι μονάδες είναι συνδεδεμένες στη βάση το προσωπικό συντήρησης μπορεί να ανταλλάξει γρήγορα και με ευκολία τις συσκευές .

Μονάδα Προγραμματισμού

Η CPU προγραμματίζεται μέσω της Μονάδας Προγραμματισμού. Ανάλογα τον κατασκευαστή έχουν διάφορες μορφές όπως για παράδειγμα μικρή συσκευή με πληκτρολόγιο και οθόνη όπου γράφουμε τις εντολές του προγράμματος. Πλέον οι

κατασκευαστές χρησιμοποιούν δικό τους λογισμικό οπότε και ο προγραμματισμός μπορεί να γίνει από υπολογιστή και να εγκατασταθεί στη συνέχεια στο PLC.

Μονάδες Εισόδου / Εξόδου

Όλες οι συσκευές συνδέονται στις Μονάδες Εισόδου / Εξόδου ενός PLC οι οποίες λειτουργούν ως σύνδεσμος μεταξύ των συσκευών αυτών και της CPU. Η μονάδα μπορεί να είναι σταθερή, όπου κάθε μοντέλο έχει προκαθορισμένο αριθμό εισόδων και εξόδων ή μπορεί να είναι μεταβαλλόμενη, όπου τις κάνουμε εγκατάσταση στη βάση rack και η ποσότητα εισόδων και εξόδων μπορεί να αλλάζει ανάλογα τις απαιτήσεις. ΟΙ είσοδοι δέχονται σήματα από τα μηχανήματα και τον εξοπλισμό και τα μετατρέπουν σε σήμα που μπορεί να επεξεργαστεί. Οι έξοδοι μετατρέπουν σήματα και τα στέλνουν σε μηχανήματα και εξοπλισμό ώστε αυτά να μπορούμε να τα ελέγξουμε.

Τα σήματα μπορεί να είναι ψηφιακού τύπου, δηλαδή να είναι της μορφής ON / OFF όπου συνήθως πρόκειται για είσοδο από διακόπτες, μπουτόν κ.τ.λ. και έξοδο σε λυχνίες, relays κ.τ.λ. . ή μπορεί να είναι αναλογικού τύπου όπου χρησιμοποιούνται για μετρήσεις όπως π.χ. θερμοκρασίας, πίεσης, ταχύτητας κ.τ.λ.

Κεντρική Μονάδα Επεξεργασίας(CPU)

Ο εγκέφαλος του PLC είναι η κεντρική μονάδα επεξεργασίας CPU η οποία διαχειρίζεται και συντονίζει το PLC ως σύνολο. Η κύρια εργασία της CPU είναι η ερμηνεία και η εκτέλεση προγραμμάτων που είναι μόνιμα αποθηκευμένα στη μνήμη του επεξεργαστή [9].

Μονάδα Τροφοδοσίας

Η μονάδα τροφοδοσίας παρέχει την σωστή τάση και ένταση ρεύματος στις υπόλοιπες μονάδες του PLC. Ανάλογα την χώρα η παροχή του δικτύου είναι 110 V AC έως και 240 V AC. Η μονάδα τροφοδοσίας παίρνει την τάση αυτή σαν είσοδο και δίνει την σωστή τάση στο PLC η οποία συνήθως είναι στα 24V DC [10].

Μνήμη

Η Μνήμη είναι απαραίτητη για την αποθήκευση δεδομένων, πληροφοριών που σχετίζονται με το σύστημα και οδηγιών του προγράμματος. Οι τύποι μνήμης που συνήθως συναντάμε σε ένα PLC είναι οι ακόλουθοι:

- System Read-Only Memory (ROM): Το λειτουργικό σύστημα και τα σταθερά δεδομένα που χρησιμοποιούνται από την CPU αποθηκεύονται μόνιμα σε αυτή τη μνήμη. Το υλικολογισμικό (firmware) για το PLC και άλλα δεδομένα σε επίπεδο συστήματος αποθηκεύονται συνήθως στη ROM, στην οποία ο χρήστης δεν μπορεί να παρέμβει πραγματοποιώντας αλλαγές.
- Random-Access Memory (RAM) για το πρόγραμμα: Το πρόγραμμα του χρήστη, το οποίο περιλαμβάνει τις οδηγίες που καθορίζουν τον τρόπο με τον οποίο το PLC χειρίζεται τα δεδομένα και διαχειρίζεται το σύστημα, διατηρείται σε αυτή τη μνήμη. Ο χρήστης έχει τη δυνατότητα να αλλάξει το πρόγραμμα που διατηρείται στη μνήμη RAM, γεγονός που του δίνει την ελευθερία να επαναπρογραμματίσει το PLC.
- Random-Access Memory (RAM) για τα δεδομένα: Πληροφορίες σχετικά με την κατάσταση των συσκευών εισόδου και εξόδου καθώς και άλλες εσωτερικές μεταβλητές διατηρούνται σε αυτή τη μνήμη.
- Erasable and Programmable Read-Only Memory (EPROM): Μία επιπλέον μονάδα μνήμης που ονομάζεται EPROM μπορεί να υπάρχει σε ορισμένα συστήματα PLC. Τα προγράμματα μπορούν να αποθηκευτούν μόνιμα με την EPROM, ακόμη και όταν η πηγή ρεύματος είναι απενεργοποιημένη. Χρησιμοποιείται συχνά για την αποθήκευση προγραμμάτων για μεγάλο χρονικό διάστημα [11].

3.3 Περιγραφή του Κύκλου Λειτουργίας ενός PLC

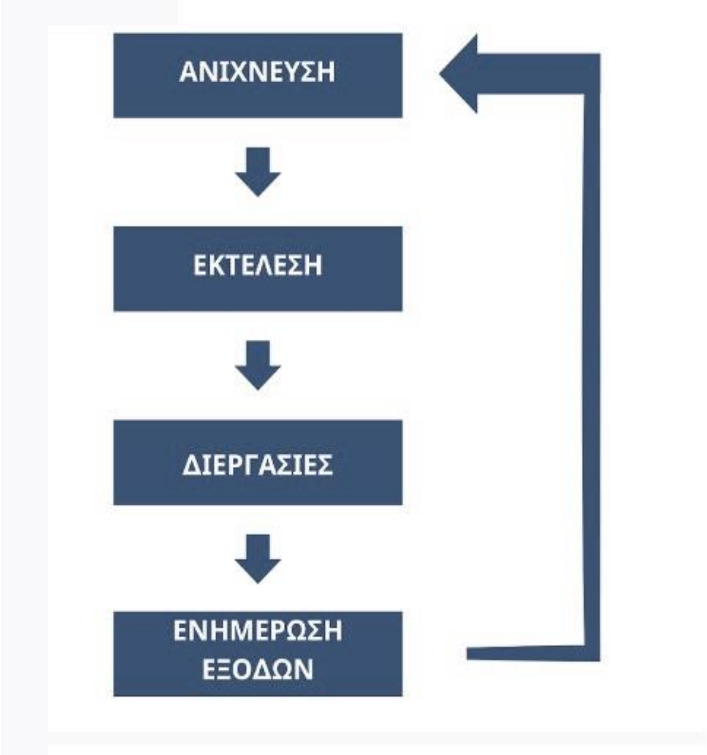
Ένας καλός τρόπος να κατανοήσουμε τον τρόπο με τον οποίο λειτουργεί ένας PLC είναι ο Κύκλος Λειτουργίας του (Scan Cycle) (Εικόνα 3.1).

Ανίχνευση εισόδων: Πραγματοποιείται διάβασμα των τιμών εισόδων και έλεγχος της κατάστασης συνδεδεμένων στοιχείων, όπως διακόπτες ή αισθητήρες.

Εκτέλεση: Πραγματοποιείται εκτέλεση του προγράμματος σύμφωνα με την κατάσταση των εισόδων που διαβάστηκαν.

Διεργασίες επικοινωνίας και διαγνωστικά: Πραγματοποιούνται κάποιες λειτουργίες, όπως επικοινωνία και αποστολή της πληροφορίας με το εκάστοτε δίκτυο επικοινωνίας. Επίσης πραγματοποιούνται διαγνωστικές διαδικασίες ώστε να ελεγχθεί ότι το σύστημα δεν έχει κάποια δυσλειτουργία.

Ενημέρωση εξόδων: Τέλος ενημερώνονται οι έξοδοι πριν ξεκινήσει ο επόμενος κύκλος [10].



Εικόνα 3.1 Διαγραμματική απεικόνιση Κύκλου Λειτουργίας (Scan Cycle Diagram)

3.4 Προγραμματισμός PLC

Στο σημείο αυτό αξίζει να παρουσιάσουμε τους τρόπους σύμφωνα με τους οποίους μπορούμε να πραγματοποιήσουμε τον προγραμματισμό ενός PLC. Σύμφωνα με το International Standard IEC 61131 ορίζονται πέντε βασικοί τρόποι προγραμματισμού PLC, τρεις εκ των οποίων βασίζονται σε γραφική αναπαράσταση και οι άλλοι δύο σε απεικόνιση μέσω εντολών κειμένου. Πρόκειται για τις εξής πέντε:

- Ladder diagram
- Sequential Function Chart
- Function Block Diagram

- Instruction List
- Structured Text [12]

Παρακάτω αναφέρονται λίγα στοιχεία για την κάθε μία από αυτές:

Ladder diagram

Πρόκειται για μια γλώσσα προγραμματισμού γραφικής αναπαράστασης. Μιμείται τη διάταξη των κυκλωμάτων με χρήση relay ή ενός διαγράμματος ηλεκτρικού κυκλώματος. Το όνομα Ladder(σκάλα), προέρχεται από την παρατήρηση ότι έχει δύο κάθετες γραμμές και έναν αριθμό οριζόντιων σκαλοπατιών, τα ονομάζουμε Rungs, σαν μια σκάλα. Αυτά αντιπροσωπεύουν τις γραμμές ισχύος.

Οι επαφές που αναπαριστά έχουν προτεραιότητα από αριστερά προς τα δεξιά και αντιπροσωπεύουν Λογικές τιμές TRUE και FALSE. Όταν η τιμή είναι TRUE τότε η πληροφορία προχωράει προς τα δεξιά [13].

Function Block Diagram

Είναι μια γλώσσα γραφικής αναπαράστασης όπου περιγράφει πως αλληλεπιδρούν οι είσοδοι με τις εξόδους. Ο σχεδιασμός γίνεται σε τμήματα (blocks) τα οποία ενώνονται με γραμμές με τις εισόδους και εξόδους. Η πληροφορία μεταφέρεται από αριστερά προς τα δεξιά και στα άκρα των γραμμών πρέπει να έχουμε τους ίδιους τύπους μεταβλητών [14].

Sequential Function Chart

Μία επίσης γλώσσα γραφικής αναπαράστασης η οποία έχει την δυνατότητα να περιγράψει εργασίες μέσα στο πρόγραμμα με χρονολογική σειρά προτεραιότητας. Πρόκειται στην ουσία για μία σειρά από βήματα που ενώνονται μεταξύ τους με συνδέσεις τις οποίες και ονομάζουμε Μεταβάσεις (transitions) [13].

Instruction List

Είναι μία γλώσσα προγραμματισμού χαμηλού επιπέδου η οποία μοιάζει με την γλώσσα Assembly.

Ο έλεγχος της ροής του προγράμματος εκτελείται από κλήση functions και εντολών jump [15].

Structured Text

Πρόκειται για μία γλώσσα προγραμματισμού υψηλού επιπέδου η οποία συντακτικά μοιάζει αρκετά με τη γλώσσα Pascal. Αποτελείται από σειρές εντολών και όπως σε κάθε γλώσσα υψηλού επιπέδου υπάρχουν λειτουργίες επανάληψης (loops) ή ελέγχου συνθηκών (conditions) [13].

3.5 Πλεονεκτήματα της χρήσης PLC

Τα PLCs είναι πλέον διαδεδομένα στους βιομηχανικούς αυτοματισμούς αφού προσφέρουν αρκετά πλεονεκτήματα σε σχέση με τα παλαιότερα συστήματα ηλεκτρονόμων. Ορισμένα από τα πλεονεκτήματα αυτά είναι τα ακόλουθα:

Ευελιξία

Τα συστήματα PLC μπορούν να διαμορφωθούν για να εκτελούν ένα εύρος λειτουργιών, που κυμαίνονται από τον απλό έλεγχο ενεργοποίησης/απενεργοποίησης έως τον περίπλοκο έλεγχο κίνησης και τον αυτοματισμό διεργασιών. Είναι πολύ ευέλικτα και απλά στον επαναπρογραμματισμό για να ανταποκρίνονται στις όποιες μεταβαλλόμενες ανάγκες [16].

Προγραμματισμός

Τα PLC υποστηρίζονται από φιλικά προς το χρήστη περιβάλλοντα προγραμματισμού που διευκολύνουν την ανάπτυξη των εφαρμογών. Χρησιμοποιώντας εργαλεία προσομοίωσης, οι μηχανικοί μπορούν να μειώσουν την πιθανότητα σφάλματος επιβεβαιώνοντας τη συμπεριφορά του λογισμικού πριν το δοκιμάσουν στο πραγματικό σύστημα [17].

Ευκολία στη συντήρηση

Τα συστήματα αυτά διαθέτουν προσαρμόσιμη αρχιτεκτονική, γεγονός που επιτρέπει την απλή αντικατάσταση εξαρτημάτων που δυσλειτουργούν χωρίς να διακυβεύεται η λειτουργικότητα ολόκληρου του συστήματος. Επιπλέον, περιλαμβάνουν ενσωματωμένες διαγνωστικές δυνατότητες που διευκολύνουν την αντιμετώπιση προβλημάτων [16].

Ικανότητες επικοινωνίας

Τα PLC διευκολύνουν την ομαλή αλληλεπίδραση με άλλα συστήματα αυτοματισμού αφού υποστηρίζουν ένα ευρύ φάσμα πρωτοκόλλων επικοινωνίας. Αυτό καθιστά δυνατό τον κεντρικό έλεγχο, την απομακρυσμένη παρακολούθηση και την ανταλλαγή δεδομένων [17].

Αυξημένη ασφάλεια

Μπορούν να διαμορφωθούν ώστε να εκτελούν διαδικασίες ασφαλείας, όπως η απενεργοποίηση έκτακτης ανάγκης. Αυτό αυξάνει το γενικό επίπεδο ασφάλειας του βιομηχανικού περιβάλλοντος [16].

Κεφάλαιο 4 Αυτοματισμοί με SIMATIC

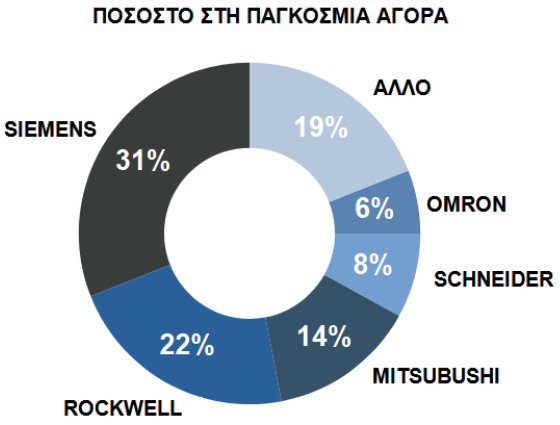
4.1 Κατασκευαστές και παγκόσμια αγορά

Σύμφωνα με έρευνα του IMARC GROUP (International Market Analysis Research and Consulting Group) η παγκόσμια αγορά PLC έφτασε τα $14.6 δισεκατομμύρια κατά το έτος 2022 και εκτιμάται να φτάσει το ποσό των $20.2 δισεκατομμυρίων έως το έτος 2028 με CAGR (ετήσιος συντελεστής αύξησης) 5.38% κατά την περίοδο 2023-2028 [18].

Οι κατασκευαστικές εταιρείες έχουν έναν σημαντικό ρόλο στο να παρέχουν PLC τα οποία ανταποκρίνονται στα διάφορα είδη βιομηχανιών. Κάποιες από τις κυρίαρχες εταιρείες στο χώρο είναι οι παρακάτω:

- SIEMENS
- ROCKWELL AUTOMATION / ALLEN-BRADLEY
- SCHNEIDER ELECTRIC
- MITSUBISHI ELECTRIC
- ABB [19]

Από έρευνα που πραγματοποιήθηκε το 2017 (Εικόνα 4.1) προέκυψε πως στην πρώτη θέση βρίσκεται η SIEMENS αγγίζοντας το ποσοστό του 31% με ακόλουθους τις ROCKWELL στο 22% και MITSUBISHI στο 14% [20].



Εικόνα 4.1 Ποσοστά παγκόσμιας αγοράς για κατασκευαστές (Global market rates for manufactures) [20]

4.2 Η εταιρεία SIEMENS ως κατασκευαστής PLCs

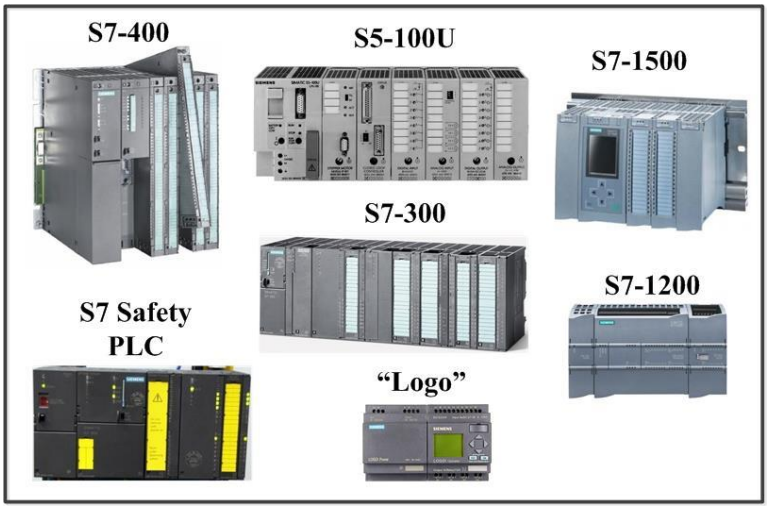
Όπως είδαμε η εταιρεία Siemens είναι ο κυρίαρχος στην παγκόσμια αγορά. Ας δούμε ορισμένα βασικά στοιχεία σχετικά με τα PLCs τα οποία κατασκευάζει.

Η Siemens παράγει μια μεγάλη ποικιλία συσκευών PLC (Εικόνα 4.2). Κάποια από τα πιο γνωστά της μοντέλα είναι τα εξής:

- Siemens Logo!
- SIMATIC ET-200SP
- SIMATIC S7-300
- SIMATIC S7-400
- SIMATIC S7-1200
- SIMATIC S7-1500 [21]

Η ονομασία SIMATIC προέρχεται από τις λέξεις “Siemens” και “Automatic”, είναι κατοχυρωμένος τίτλος της εταιρείας και έχει περάσει τέσσερις γενιές παραγωγής.

- Το 1958 με την έκδοση SIMATIC Version G
- Το 1973 με την έκδοση SIMATIC S3
- Το 1979 με την έκδοση SIMATIC S5
- Το 1995 με την έκδοση SIMATIC S7 [22]



Εικόνα 4.2 Μοντέλα PLCs της SIEMENS (SIEMENS PLCs Models) [23]

Τα PLCs της Siemens χρησιμοποιούνται σε μία πληθώρα διαφορετικών βιομηχανικών εφαρμογών όπως κατασκευαστική, ενέργεια, γεωργία κ.α. Ακόμη ένα πλεονέκτημα ενός Siemens PLC είναι η ικανότητά του να διασυνδέεται με άλλα συστήματα και συσκευές. Έτσι καθίστανται εφικτή η παρακολούθηση και ανάλυση δεδομένων σε πραγματικό χρόνο, ο απομακρυσμένος έλεγχος και διαχείριση συσκευών. Μερικά από τα συστήματα με τα οποία έχουν δυνατότητα επικοινωνίας είναι:

Οθόνες HMI: Οι οθόνες HMI (Human Machine Interface) επιτρέπουν στον χειριστή να παρακολουθεί και να ελέγχει τις λειτουργίες μέσω γραφικής απεικόνισης.

SCADA: Τα συστήματα SCADA (Supervisory Control and Data Acquisition) χρησιμοποιούνται για τον έλεγχο πολλαπλών PLCs καθώς και την συλλογή και ανάλυση δεδομένων.

IIoT systems: Τα PLCs της Siemens προσφέρουν ακόμη την δυνατότητα σύνδεσης με συστήματα IIoT (Industrial Internet of Things) για ανάλυση δεδομένων σε πραγματικό χρόνο, προγνωστική συντήρηση ή απομακρυσμένο έλεγχο.

Για την επικοινωνία των PLC με άλλα συστήματα η Siemens τα έχει σχεδιάσει ώστε να έχουν την δυνατότητα να χρησιμοποιούν διάφορα πρωτόκολλα επικοινωνίας όπως:

PROFIBUS: Πρόκειται για ένα δημοφιλές βιομηχανικό πρωτόκολλο επικοινωνίας το οποίο καθιστά δυνατή τη γρήγορη μεταφορά δεδομένων και τον έλεγχο σε πραγματικό χρόνο. Είναι μια δημοφιλής επιλογή για PLC της Siemens, καθώς είναι εύκολη η εγκατάσταση και υποστηρίζει πολλές συσκευές σε ένα μόνο δίκτυο.

PROFINET: Πρόκειται για ένα πιο σύγχρονο πρωτόκολλο επικοινωνίας που βασίζεται στις δυνατότητες του PROFIBUS και παρέχει υψηλότερη απόδοση και ταχύτερους ρυθμούς μεταφοράς δεδομένων. Λόγω του ειδικού σχεδιασμού του για βιομηχανικό αυτοματισμό και βελτιστοποίηση για PLC της Siemens, αποτελεί μία αρκετά δημοφιλής επιλογή.

ETHERNET/IP: Πρόκειται για ένα βιομηχανικό πρωτόκολλο επικοινωνίας που συνδέει συστήματα και συσκευές χρησιμοποιώντας κοινό hardware και software Ethernet. Παρέχει γρήγορους ρυθμούς μεταφοράς δεδομένων και χρησιμοποιείται ευρέως σε βιομηχανίες όπου η επικοινωνία υψηλής ταχύτητας είναι ζωτικής σημασίας.

Σε ότι αφορά την CPU η εταιρεία παρέχει μία πληθώρα CPUs με διάφορες δυνατότητες ανάλογα τις όποιες απαιτήσεις.

Τέλος οι Μονάδες Εισόδου / Εξόδου που παρέχονται καλύπτουν εισόδους και εξόδους ψηφιακές είτε αναλογικές αλλά υπάρχουν και ειδικές μονάδες για εξειδικευμένες εφαρμογές [24].

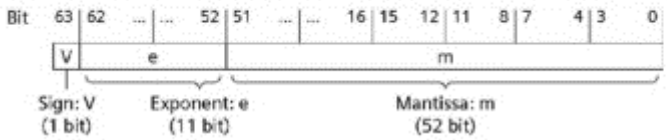
4.3 Τύποι δεδομένων και Περιοχές Μνήμης

Στην Εικόνα 4.3 μπορούμε να εξετάσουμε τους διάφορους τύπους δεδομένων που υποστηρίζονται από τα μοντέλα S7-300, S7-400, S7-1200 και S7-1500 συγκριτικά για καθένα από αυτά, όπως αυτοί παρέχονται από τον κατασκευαστή.

S7-300 S7-400	S7-1200	S7-1500	Data type	Bit length	Value range	Examples, comments
Binary						
✔	✔	✔	BOOL	1	TRUE, FALSE	varBool := (var1 AND var2) BOOL#0, BOOL#1
Binary numbers and character strings						
Decimal, binary, octal or hexadecimal						
✔	✔	✔	BYTE	8	0 ... 255	varByte := 2#0011_1010
✔	✔	✔	WORD	16	0 ... 65 535	varWord := 16#6B0F
✔	✔	✔	DWORD	32	0 ... 4 294 967 295	varDword := 50_000
		✔	LWORD	64	0 ... 18 446 744 073 709 551 615	varLword := 16#F2F6_FA9F_FBFF_FBFF
Integer numbers						
Decimal, binary, octal or hexadecimal When an integer number is not in decimal format, the most significant bit, MSB, determines the sign: 0 = positive, 1 = negative						Bit 7 4 3 0 0 0 1 0 1 1 0 0 Sign Decimal values: 32 8 4 = 44
	✔	✔	SINT	8	-128 ... +127	varSint := -42
S7-300 S7-400	S7-1200	S7-1500	Data type	Bit length	Value range	Examples, comments
✔	✔	✔	INT	16	-32 768 ... +32 767	varInt := 16#0EC9
✔	✔	✔	DINT	32	-2 147 483 648 ... +2 147 483 647	varDint := +125_790
		✔	LINT	64	-9 223 372 036 854 775 808 ... +9 223 372 036 854 775 807	varLint := 16#0000_8C5B_C5F0_F79F
Integer numbers without sign						
Decimal, binary, octal or hexadecimal						
	✔	✔	USINT	8	0 ... 255	varUsint := 2#0100_1110
	✔	✔	UINT	16	0 ... 65 535	varUint := 65_295
	✔	✔	UDINT	32	0 ... 4 294 967 295	varUdint := 8#360_7417_0360
	✔	✔	ULINT	64	0 ... 18 446 744 073 709 551 615	varUlint := 16#0000_8C5B_C5F0_F79F

Floating-point numbers

Floating-point numbers correspond to the standard IEEE 754-1985



S7-300	S7-400	S7-1200	S7-1500	Data type	Bit length	Value range	Examples, comments
✔	✔	✔	✔	REAL	32	-3.402823e+38 ... -1.175 495e-38 ±0 +1.175 495e-38 ... +3.402823e+38	varReal := 1.0e-5 Mantissa: 23 bits, Exponent: 8 bits, Sign 1 bit
	✔	✔	✔	LREAL	64	-1,7976931348623158e+308 ... -2,2250738585072014e-308 ±0 +2.2250738585072014e-308 ... +1.7976931348623158e+308	varLreal := 20.0e-15 Mantissa: 52 bits, Exponent: 11 bits, Sign: 1 bit

Timer

S7-300	S7-400	S7-1200	S7-1500	Data type	Bit length	Value range	Examples, comments
✔		✔		S5TIME	16	0 ms ... 2 h 46 m 30 s 0 ms	varS5time := S5T#10s
✔	✔	✔	✔	TIME	32	-24 d 20 h 31 m 23 s 648 ms ... +24 d 20 h 31 m 23 s 647 ms	varTime := T#10d20h30m20s630ms
			✔	LTIME	64	-106 751 d 23 h 47 m 16 s 854 ms 775 µs 808 ns ... +106 751 d 23 h 47 m 16 s 854 ms 775 µs 807 ns	varLtime := LT#11350d20h25m14s830ms 652us315ns

Date and time

S7-300	S7-400	S7-1200	S7-1500	Data type	Bit length	Value range	Examples, comments
✔	✔	✔	✔	DATE	16	01.01.1990 ... 31.12.2168	varDate := D#2009-12-31
✔	✔	✔	✔	TIME_OF_DAY (TOD)	32	00:00:00.000 ... 23:59:59.999	varTod := TOD#10:20:30.400
			✔	LTOD (LTIME_OF_DAY)	64	00:00:00.000000000 ... 23:59:59.999999999	varLtod := LTOD#10:20:30.400_365_215
✔			✔	DT (DATE_ AND_TIME)	64	01.01.1990--0:0:0 ... 31.12.2089--23:59:59.999	varDt := DT#2008-10-25-8:12:34.567
			✔	LDT	64	01.01.1970--0:0:0.000000000 ... 11.04.2262--23:47:16.854775807	varLdt := LDT#2008-10-25-8:12:34.567
		✔	✔	DTL	96	01.01.1970--00:00:00.0 ... 31.12.1554--23:59:59.999999999	varDtl := DTL#2008-12-16-20:30:20.250

Character string						
An operand of the STRING data type occupies two bytes more than the specified maximum length in the memory. An operand of the WSTRING data type occupies two words (4 bytes) more than the specified maximum length in the memory. You can specify the length of a character string by adding a definition. E.G.: STRING[254]						
✔	✔	✔	CHAR	8	ASCII character set	varChar := 'A'
	✔	✔	WCHAR	16	Unicode character set	varWchar := 'A'
✔	✔	✔	STRING	n+2 (bytes)	0 ... 254 ASCII characters Default length: 254 CHAR + 2 bytes	varString := 'Name'
	✔	✔	WSTRING	n+2 (Word)	0 ... 16382 Unicode characters Default length: 254 WCHAR + 2 words	varWstring := 'Hello World'

Pointer						
✔		✔	POINTER	48	Area-internal pointer, Cross-area pointer, DB pointer, Zero pointer	Symbolic: "MyDB"."MyTag" Absolute: P#20.0, P#DB10.DBX20.0
✔		✔	ANY	80	P#MemoryArea DataAddress Type Number, P#Zero value	Symbolic: "MyDB".StructVariable.Component1" Absolute: P#DB11.DBX20.0 INT 10
S7-300 S7-400	S7-1200	S7-1500	Data type	Bit length	Value range	Examples, comments
	✔	✔	VARIANT	0	*Symbolic* operand, DataBlock.Operant.Component, *Absolut* operand, DataBlockNumber.Operand Type Length, NULL pointer	Symbolic: "DataBlockI".StructVariable.Variable1" Absolute: %MW10, P#DB10.DBX10.0 INT 12

Εικόνα 4.3 Τύποι δεδομένων στον προγραμματισμό SIMATIC (Data types in SIMATIC programming) [25]

Σε ότι αφορά την πρόσβαση στα δεδομένα και την διαχείριση αυτών η Siemens διευκολύνει τον συμβολικό προγραμματισμό. Προκειμένου να χειριζόμαστε τις διευθύνσεις των δεδομένων, ορίζονται συμβολικά ονόματα ή "ετικέτες" (Tags). Αυτά μπορεί να αναφέρονται σε θέσεις μνήμης, σε τοπικές μεταβλητές ή σε εισόδους και εξόδους. Η CPU παρέχει τις εξής επιλογές για την αποθήκευση δεδομένων κατά την εκτέλεση του προγράμματος:

Global memory: παρέχονται οι ειδικές περιοχές μνήμης “M” για τα bit memory, “I” για τις εισόδους και “Q” για τις εξόδους.

Tag table: εδώ εισάγουμε συμβολικά ονόματα για συγκεκριμένες θέσεις μνήμης και έτσι μπορούμε να δημιουργήσουμε προγράμματα χρησιμοποιώντας ονομασίες σχετικές με τις εφαρμογές μας.

Data block (DB): μπορούμε να αποθηκεύσουμε δεδομένα έτσι ώστε να είναι προσβάσιμα και να παραμένουν και μετά την ολοκλήρωση της εκτέλεσης ενός προγράμματος.

Temp memory: συμβολίζεται ως L (local memory) και πρόκειται για προσωρινή μνήμη ή οποία χρησιμοποιείται από την CPU κατά την εκτέλεση του προγράμματος και έπειτα απελευθερώνεται για επόμενη χρήση.

Κάθε διαφορετική θέση μνήμης έχει την δική της διεύθυνση. Η διεύθυνση καθορίζεται από τα ακόλουθα στοιχεία:

- Αναγνωριστικό περιοχής π.χ. I, Q, M
- Μέγεθος στο οποίο θέλουμε να έχουμε πρόσβαση. Δηλαδή "B" για Byte, "W" για Word, "D" για DWord
- Σημείο εκκίνησης διεύθυνσης π.χ. byte 3, word 3

Οι τρόποι με τους οποίους μπορούμε να επικαλεστούμε μία θέση μνήμης ακολουθούν τους παρακάτω τύπους:

Κατηγορίες I, Q, M:
Για bit:
[Αναγνωριστικό περιοχής][διεύθυνση byte].[διεύθυνση bit]
π.χ. I0.3 ή Q1.4 ή M2.7
Για Byte, Word, Dword:
[Αναγνωριστικό περιοχής][τύπος μεγέθους].[Byte εκκίνησης διεύθυνσης]
π.χ. IB20 ή QW10 ή MD40
Κατηγορία DB:
Για bit:
[Αναγνωριστικό περιοχής][αριθμός data block].DBX[διεύθυνση byte]. [διεύθυνση bit]
π.χ. DB2.DBX3.1
Για Byte, Word, Dword:
[Αναγνωριστικό περιοχής][αριθμός data block].DB [τύπος μεγέθους][byte εκκίνησης διεύθυνσης]

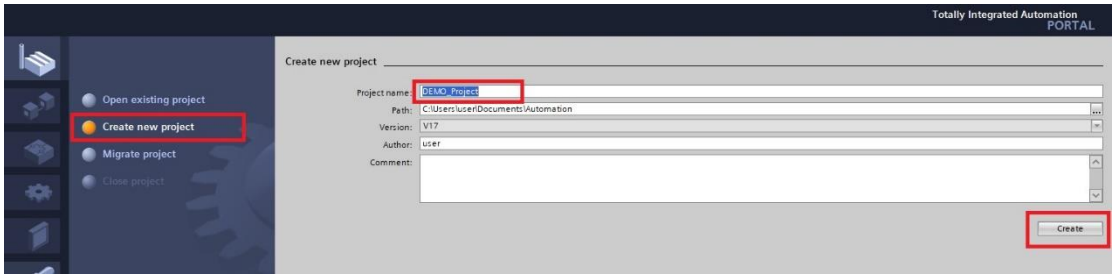
π.χ.DB3.DBW2 [26]

4.4 Το περιβάλλον προγραμματισμού

Στην παράγραφο αυτή θα κάνουμε μία εισαγωγή στο ψηφιακό περιβάλλον εργασίας μας, το λογισμικό TIA Portal (Totally Integrated Automation Portal) της εταιρείας SIEMENS.

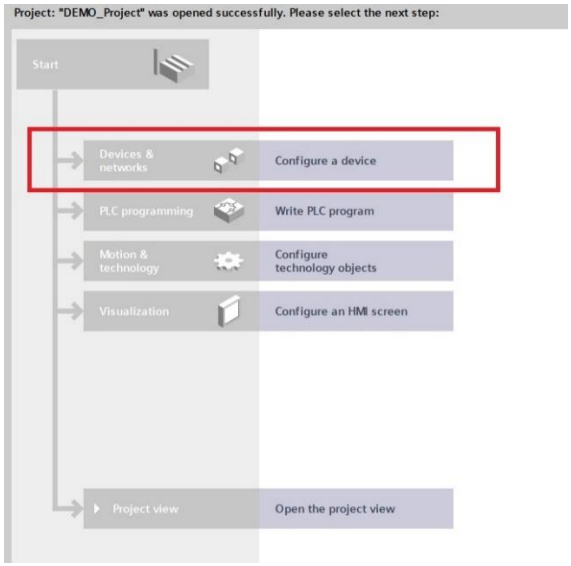
Παρακάτω θα δούμε ορισμένα στιγμιότυπα χρήσης του εν λόγω λογισμικού και θα εξηγήσουμε τις πολύ βασικές και απαραίτητες λειτουργίες που χρειάζεται κάποιος ώστε να ξεκινήσει να το χρησιμοποιεί.

Εκτελώντας εκκίνηση του λογισμικού βλέπουμε πως μπορούμε να δημιουργήσουμε ένα νέο Project (Εικόνα 4.4). Επιλέγουμε “Create new project”, δίνουμε όνομα και επιλέγουμε “Create”.



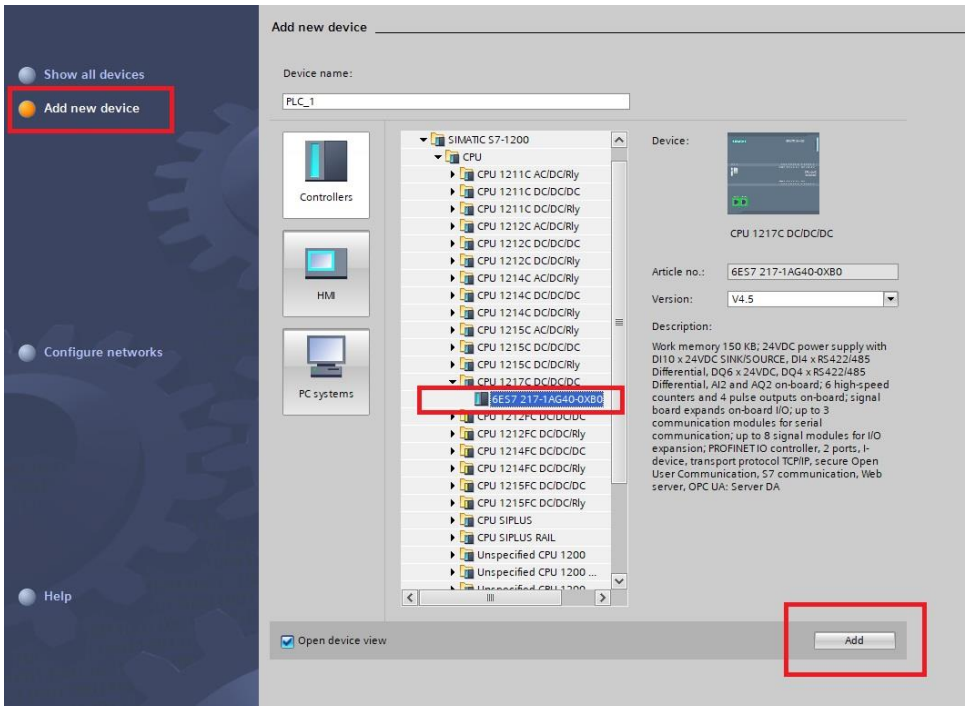
Εικόνα 4.4 Δημιουργία νέου project (New Project Creation)

Στη συνέχεια από το μενού (Εικόνα 4.5) επιλέγουμε την εντολή Configure a device ώστε να επιλέξουμε τον εξοπλισμό μας.



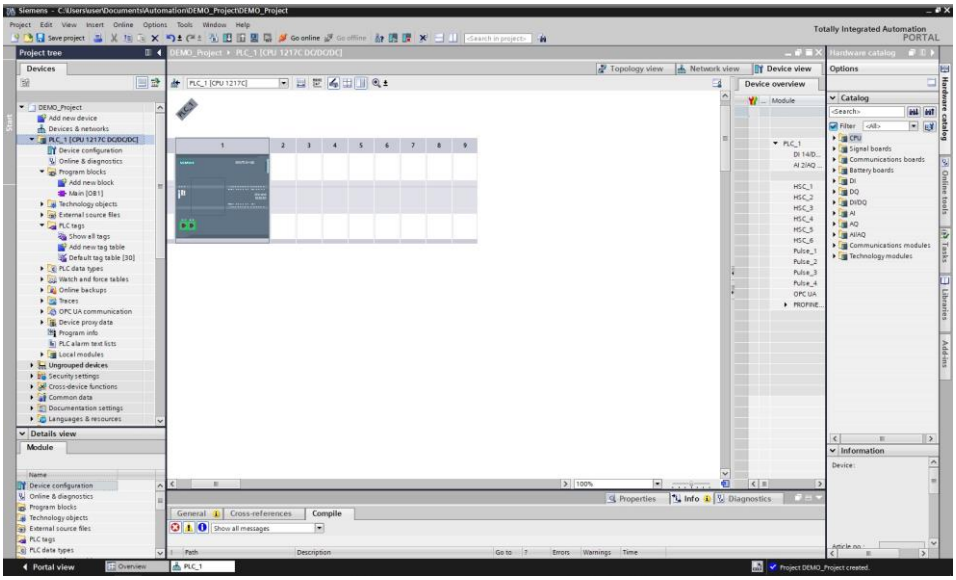
Εικόνα 4.5 Διαμόρφωση συσκευών (Devices configuration)

Με την εντολή “Add new device” μπορούμε να ορίσουμε το hardware το οποίο επιλέγουμε να χρησιμοποιήσουμε. Για παράδειγμα στην προκειμένη περίπτωση επιλέγουμε μία έκδοση του PLC SIMATIC S7-1200 όπου και το λογισμικό μας δείχνει έπειτα κάποια τεχνικά του χαρακτηριστικά (Εικόνα 4.6). Το επιλέγουμε και πατάμε “Add” Εδώ να πούμε πως εκτός του PLC μας δίνεται η δυνατότητα να προσθέσουμε και άλλα στοιχεία hardware όπως οθόνες HMI, PC Systems.



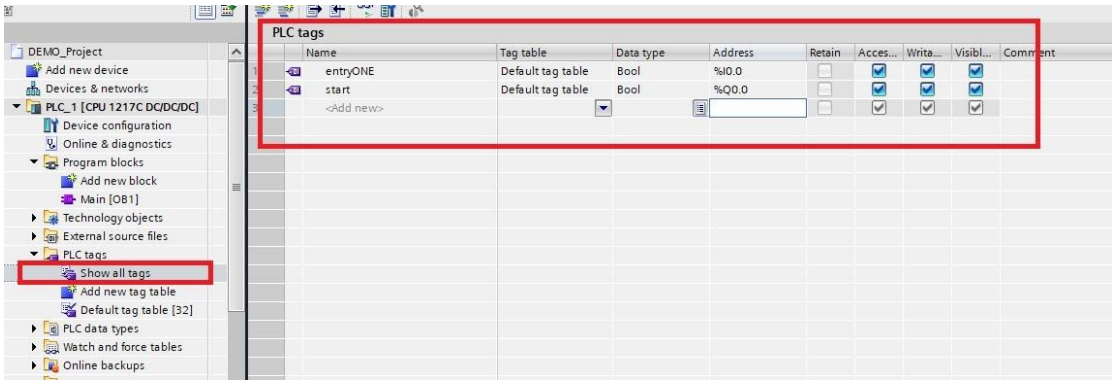
Εικόνα 4.6 Επιλογή εξοπλισμού (Hardware selection)

Σε επόμενο βήμα οδηγούμαστε στο κεντρικό περιβάλλον εργασίας του λογισμικού. Εδώ μας παρουσιάζεται μία πληθώρα επιλογών και πληροφορίας. Για παράδειγμα επιλέγοντας το μενού Device Configuration μπορούμε να ελέγξουμε τις hardware με το οποίο εργαζόμαστε όπως την τοπολογία, τις συνδέσεις κτλ. (Εικόνα 4.7)



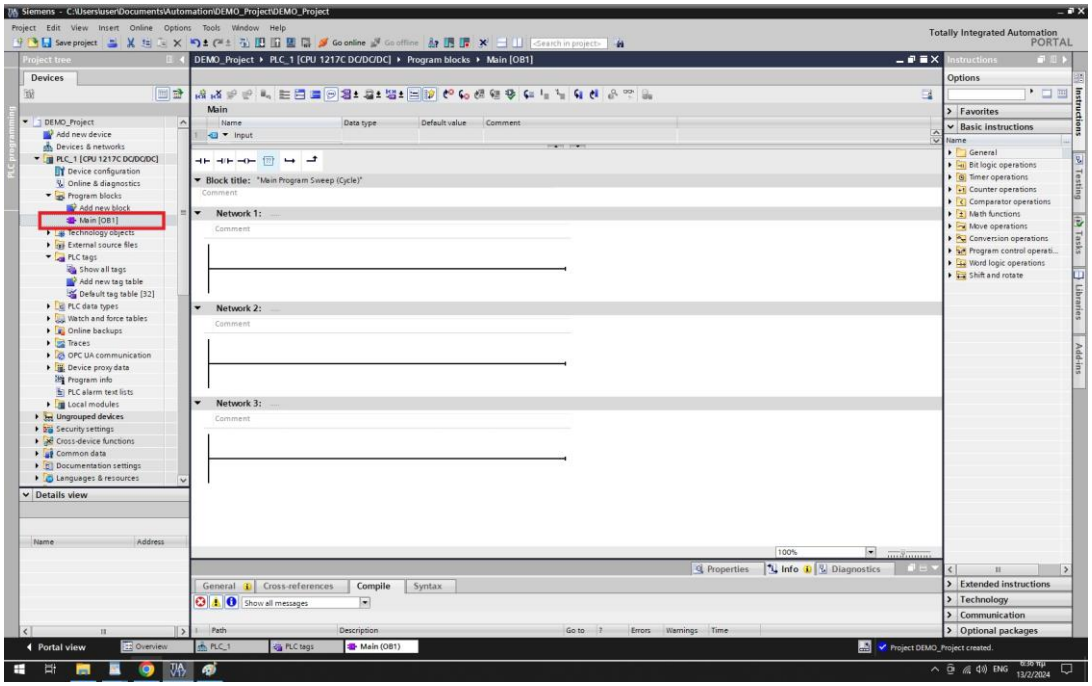
Εικόνα 4.7 Επιλογές hardware (Hardware configuration)

Το επόμενο βήμα μας είναι να ορίσουμε τα tags τα οποία θα χρησιμοποιήσουμε στο πρόγραμμά μας. Από το αριστερό μενού “Show all tags” ανοίγει ο πίνακας για να κάνουμε εισαγωγή. Στο παράδειγμά μας (Εικόνα 4.8) έχουμε ορίσει ένα tag με όνομα entryONE και διεύθυνση I0.0 καθώς πρόκειται για είσοδο και άλλο ένα με όνομα “start” και διεύθυνση Q0.0 ως έξοδο.



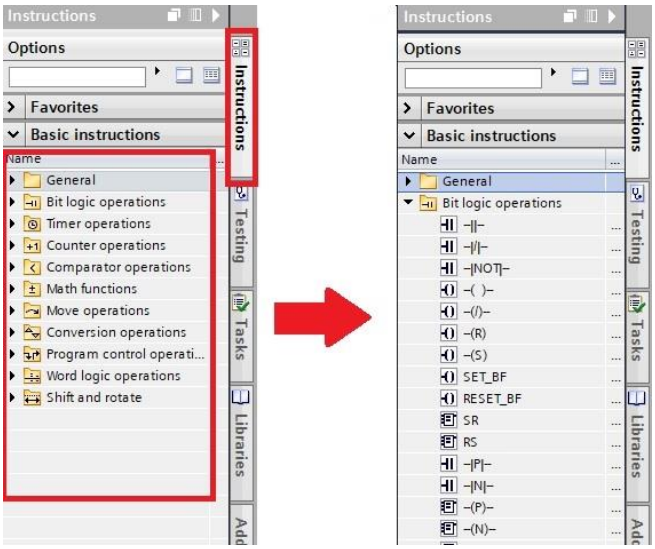
Εικόνα 4.8 Ορισμός Ετικετών (Define Tags)

Κάνοντας μία βασική επισκόπηση του παραθύρου που έχουμε μπροστά μας (Εικόνα 4.9) βλέπουμε πως έχουμε πρόσβαση σε όλες τις απαραίτητες λειτουργίες που χρειαζόμαστε ώστε να δημιουργήσουμε τους αυτοματισμούς μας. Το ίσως βασικότερο όλων είναι η επιλογή Main από τον φάκελο Program Blocks(το πρόγραμμά μας θα εκτελεστεί μέσω της Main). Ανοίγοντας την Main από τον φάκελο Program Blocks εισερχόμαστε στο κεντρικό περιβάλλον όπου δημιουργείται των κώδικά μας(σε μορφή networks). Ο κώδικάς μας θα τρέξει μέσα από την Main, σε αυτή θα τον γράψουμε ή ακόμα και θα καλέσουμε κάποιο άλλο τμήμα κώδικα το οποίο ενδεχομένως επιθυμούμε να έχουμε.



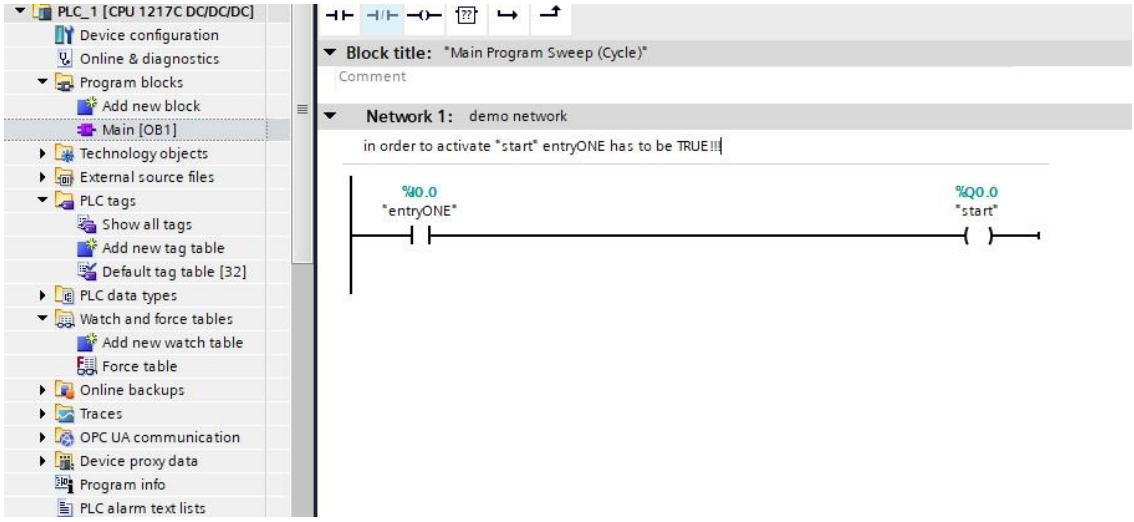
Εικόνα 4.9 Παράθυρο προγραμματισμού (Programming Environment)

Στο πλαίσιο δεξιά μπορούμε να δούμε το μενού εντολών Ladder Logic (Εικόνα 4.10). Από εδώ επιλέγουμε τα στοιχεία που θα χρειαστούμε για τον προγραμματισμό μας.



Εικόνα 4.10 Μενού εντολών (Menu)

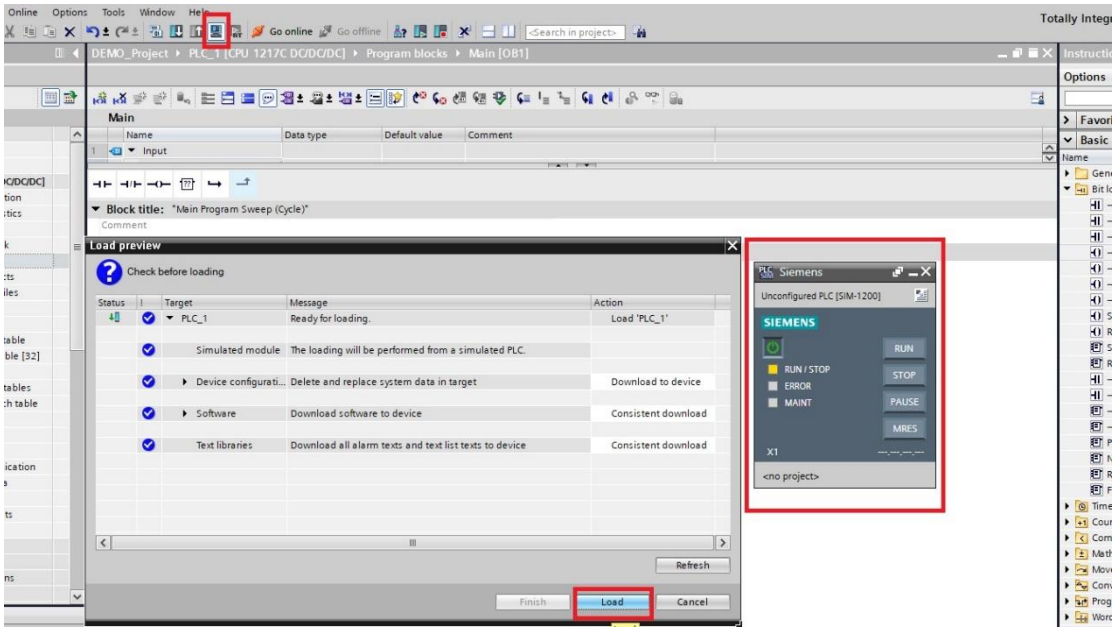
Δημιουργώντας το πρώτο μας network (Εικόνα 4.11) βλέπουμε πως τοποθετήσαμε ένα NO Contact και ένα Coil (περισσότερα για τη Ladder Logic θα αναλύσουμε στο επόμενο κεφάλαιο) με τα αντίστοιχα tag entryONE και start που δημιουργήσαμε νωρίτερα.



Εικόνα 4.11 Δημιουργία network (Network creation)

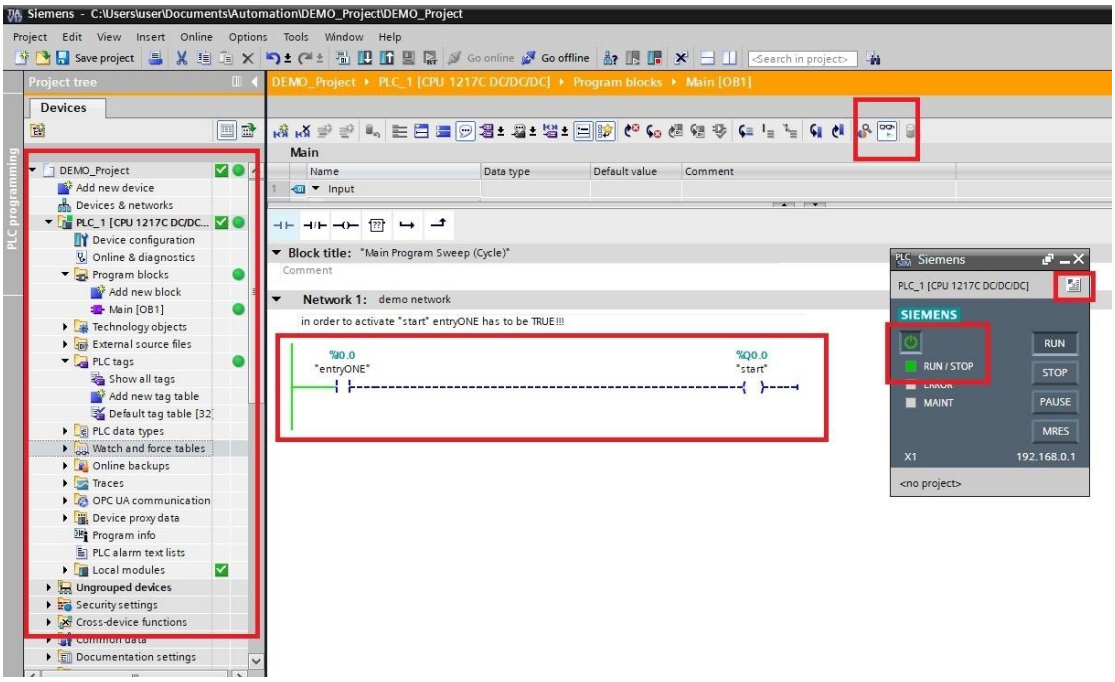
Στην περίπτωση που δεν διαθέτουμε το hardware, μία ακόμη δυνατότητα που μας προσφέρεται είναι η προσομοίωση του προγράμματός μας. Αυτό είναι δυνατόν μέσω συνεργασίας μεταξύ του TIA Portal και του λογισμικού προσομοίωσης PLCSIM που μας παρέχει η Siemens. Έτσι πατώντας στο πάνω μενού την έναρξη της προσομοίωσης ανοίγει

ένα παράθυρο εικονικού ελέγχου (Εικόνα 4.12) Παράλληλα επιλέγουμε την εντολή “Load” ώστε να ανεβάσουμε το πρόγραμμά μας.



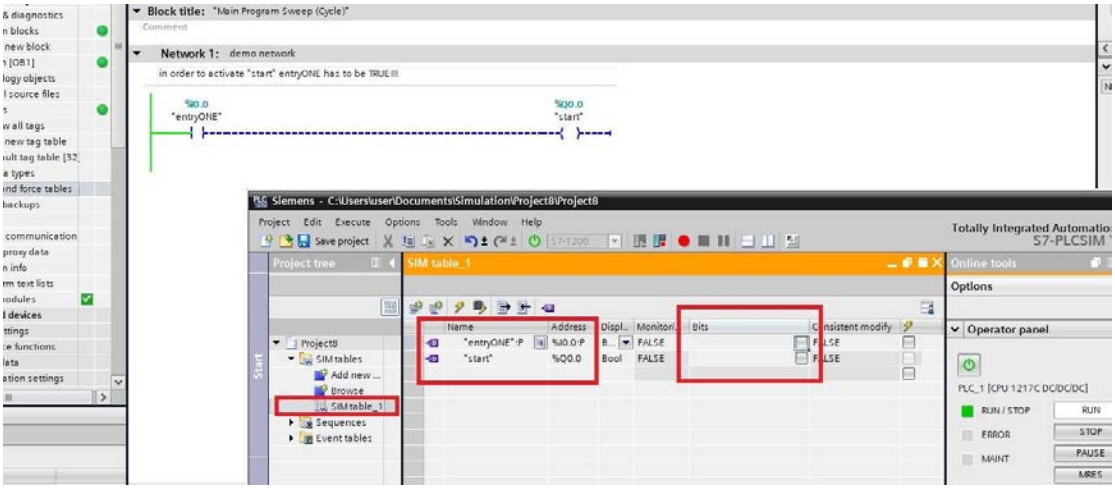
Εικόνα 4.12 Εκκίνηση προσομοίωσης με το PLCSIM (Start Simulation with PLCSIM)

Επιλέγοντας την εντολή “Monitor” από το πάνω μενού βλέπουμε αριστερά ότι δεν υπάρχουν σφάλματα (πράσινη ένδειξη) καθώς και το παράθυρο προσομοίωσης είναι σε κατάσταση “RUN” (Εικόνα 4.13)



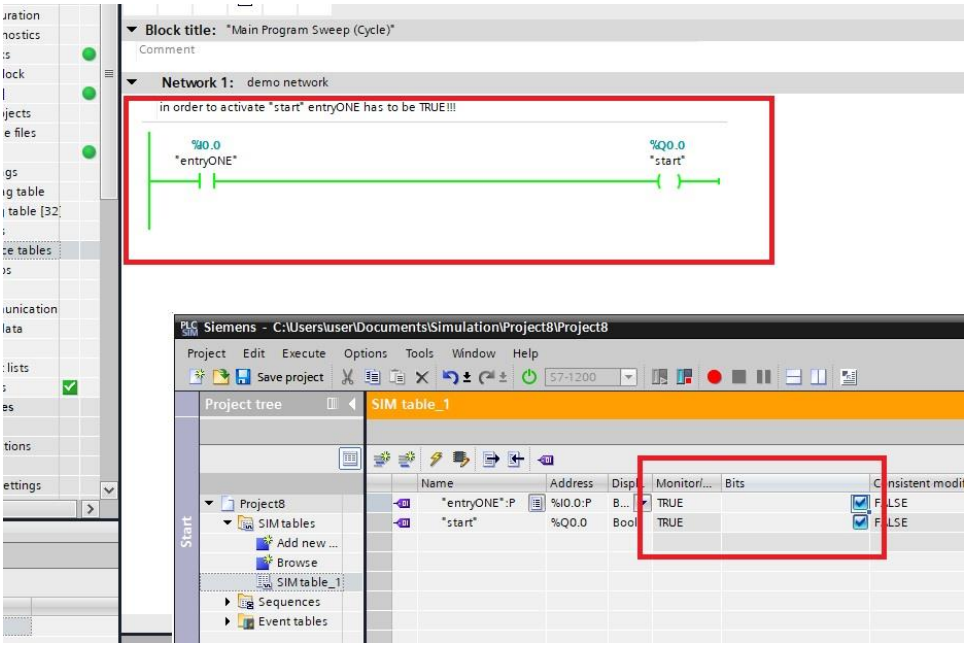
Εικόνα 4.13 Προσομοίωση (Simulation)

Παρατηρούμε στο rung ότι έχει πράσινο χρώμα μέχρι το entryONE. Άρα δεν φτάνει κάποιο σήμα στην έξοδο start. Αυτό συμβαίνει γιατί η NO Contact έχει τιμή False εξ ορισμού. Το επόμενο βήμα μας είναι πατώντας στο παράθυρο προσομοίωσης πάνω δεξιά να ανοίξουμε τους πίνακες προσομοίωσης. Δηλώνουμε και εκεί τις τιμές από τα tags και βλέπουμε πως έχουν τιμή 0 FALSE (Εικόνα 4.14)



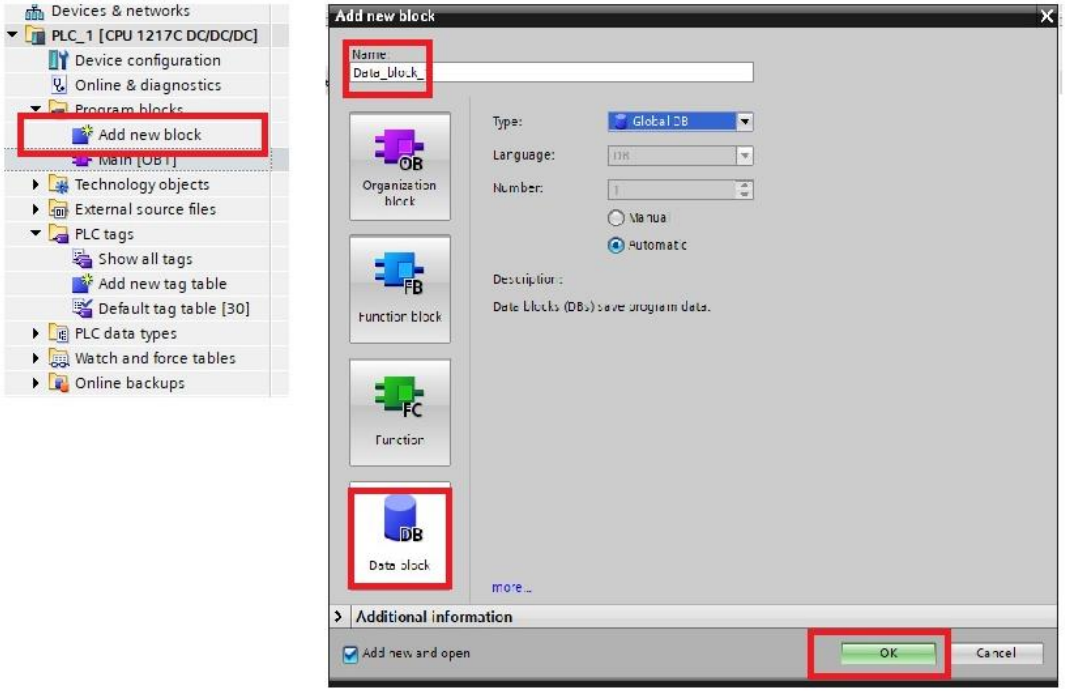
Εικόνα 4.14 Πίνακας Προσομοίωσης (Simulation Table)

Δίνοντας λοιπόν στην είσοδο entryONE την τιμή 1 TRUE αυτή με τη σειρά της θα δώσει σήμα και θα ενεργοποιήσει την έξοδο start. (Εικόνα 4.15) Βλέπουμε έτσι πως το Network πλέον έχει πράσινο χρώμα άρα η λογική μας ενεργοποιεί την έξοδο. Παράλληλα η έξοδος απέκτησε την τιμή TRUE στον πίνακα προσομοίωσης.



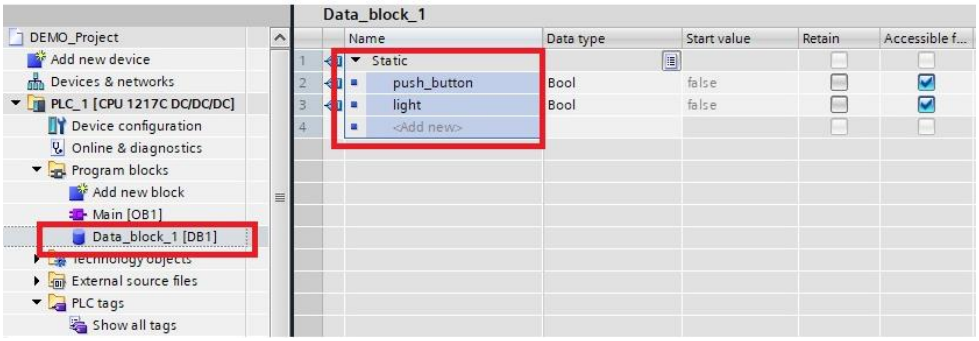
Εικόνα 4.15 Ενεργοποίηση της εξόδου (Output activation)

Στην περίπτωση που θέλουμε να δημιουργήσουμε Data Blocks (Εικόνα 4.16) για τα δεδομένα μπορούμε επιλέγοντας Add new block από το φάκελο Program Blocks. Στη συνέχεια επιλέγουμε “Data Block”, δίνουμε όνομα και πατάμε “OK”.

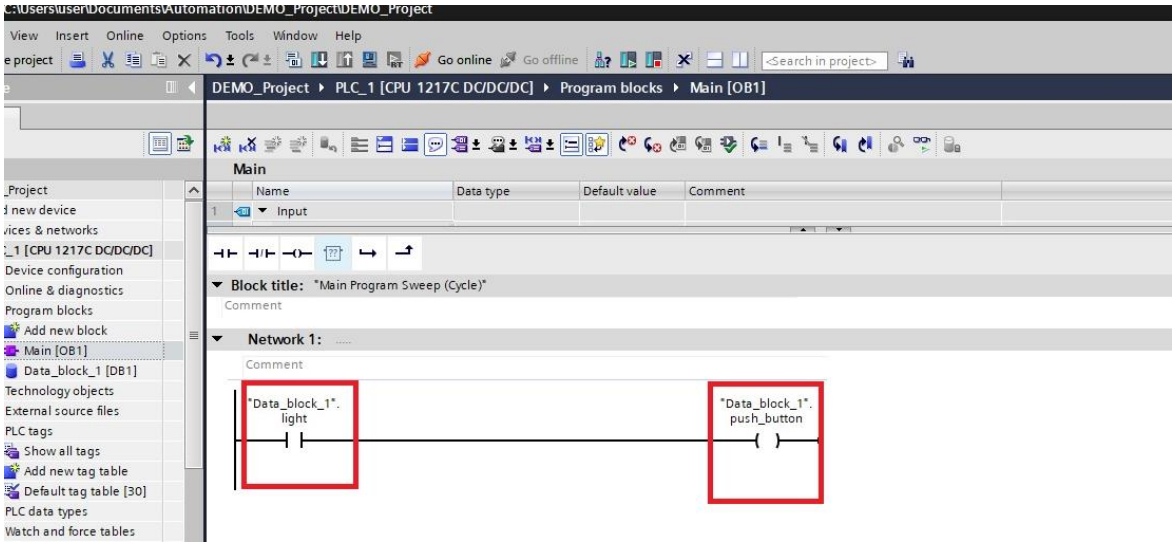


Εικόνα 4.16 Δημιουργία Data Block (Data Block creation)

Στη συνέχεια ανοίγοντας το Data Block που δημιουργήσαμε ορίζουμε τις τιμές μας (Εικόνα 4.17) και πλέον μπορούμε να τις χρησιμοποιήσουμε στα στοιχεία του προγράμματός μας (Εικόνα 4.18) αναφερόμενοι σε αυτά με το όνομα του DB ακολουθούμενου από τελεία και το όνομα της τιμής.

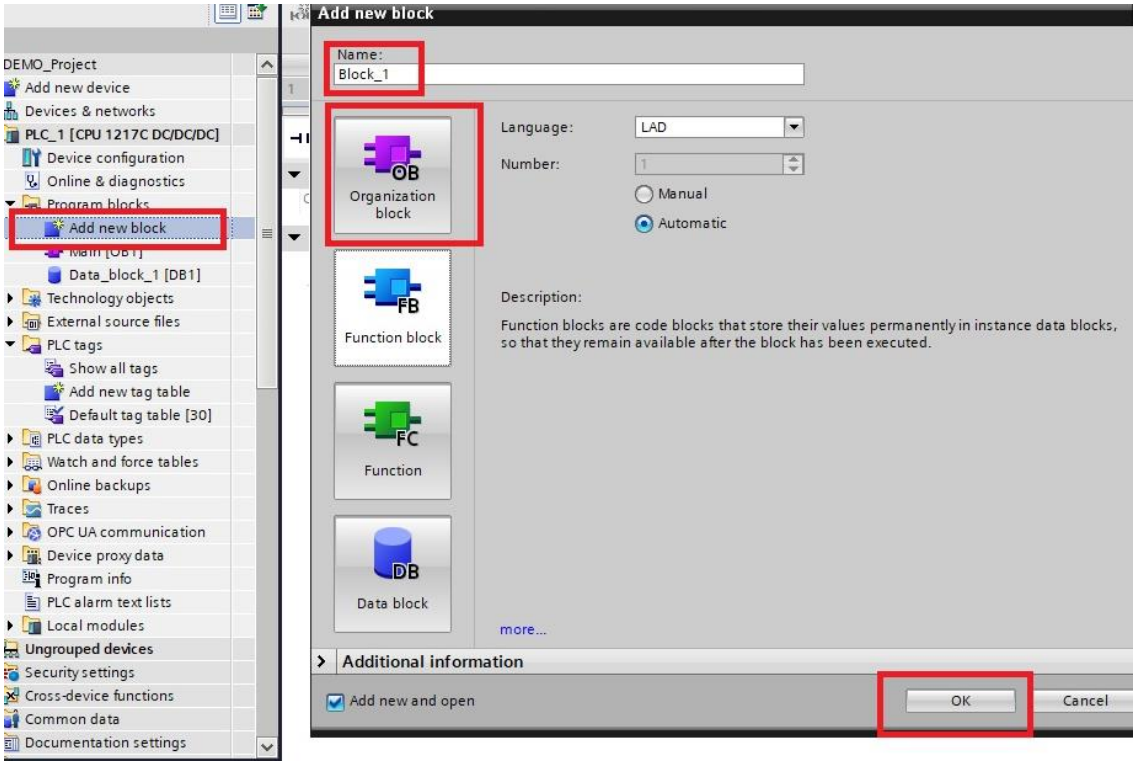


Εικόνα 4.17 Τιμές Data Block (Data Block values)

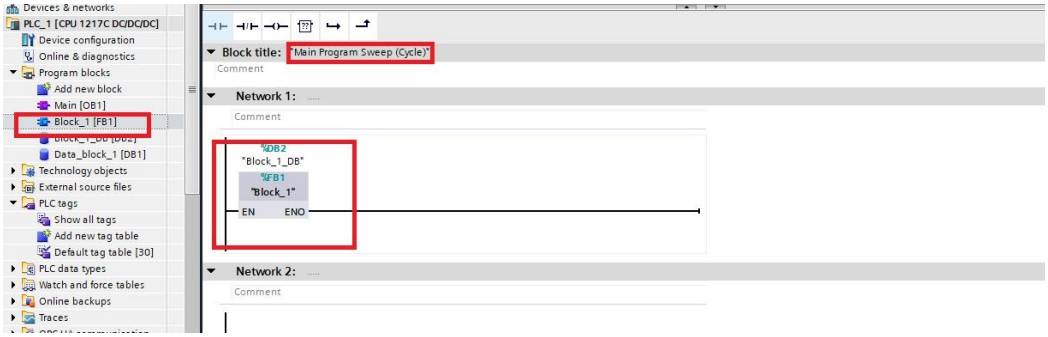


Εικόνα 4.18 Χρήση τιμών Data Block (Use of Data Block values)

Με τον ίδιο τρόπο αν θέλουμε να δημιουργήσουμε τμήματα κώδικα τα οποία θα καλέσουμε αργότερα στη MAIN μπορούμε να δημιουργήσουμε ένα Function Block (Εικόνα 4.19) να δημιουργήσουμε εκεί το τμήμα κώδικα που επιθυμούμε και στη συνέχεια με drag'N'drop θα το τοποθετήσουμε σε κάποιο network της Main (Εικόνα 4.20).



Εικόνα 4.19 Δημιουργία Function Block (Function Block creation)



Εικόνα 4.20 Τοποθέτηση Block (Block placement)

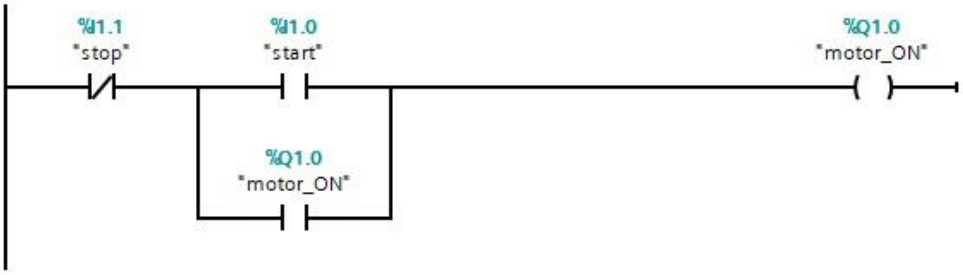
(Όλα τα παραπάνω στιγμιότυπα οθόνης αντλήθηκαν μετά από χρήση των λογισμικών TIA PORTAL και PLCSIM)

Αντιλαμβανόμαστε πως το λογισμικό αυτό έχει πολλές ακόμη δυνατότητες που δεν καλύψαμε όμως με τα παραπάνω απλά παραδείγματα κάναμε μία εισαγωγή στο περιβάλλον εργασίας και κατανοήσαμε την λογική με την οποία εργαζόμαστε για να δημιουργήσουμε τα προγράμματά μας.

Κεφάλαιο 5 Εισαγωγή στην Ladder Logic

5.1 Λογική και σύνταξη

Η Ladder Logic θα μπορούσε να χαρακτηριστεί ως γλώσσα που βασίζεται σε κανόνες. Κάθε κανόνας αντιπροσωπεύεται από ένα Rung (Εικόνα 5.1). Όταν ένα PLC εκτελεί τη γλώσσα οι κανόνες εφαρμόζονται σε σειρά για κάθε κύκλο λειτουργίας. Εκτελώντας τον κύκλο λειτουργίας με μεγάλη ταχύτητα, πολλές φορές ανά δευτερόλεπτο επιτυγχάνουμε το φαινόμενο της ταυτόχρονης εκτέλεσης των εντολών. Θα μπορούσαμε να προσεγγίσουμε την Ladder Logic σαν ένα σύνολο συνδέσεων μεταξύ λογικών ελεγκτών (Contacts) και ενεργοποιητών (Coils). Έτσι εάν υπάρχει μια διαδρομή από την αριστερή πλευρά μέχρι την έξοδο δεξιά , μέσω αληθών ή "κλειστών" επαφών, στο bit του coil στην έξοδο τίθεται η τιμή 1 (TRUE). Αν δεν υπάρχει κλειστή διαδρομή τότε η έξοδος είναι 0 (FALSE) και κατά αναλογία με τα κυκλώματα με χρήση Relays το Coil θεωρείται απενεργοποιημένο. Κάθε Contact μπορεί να αντιπροσωπεύει εισόδους από εξωτερικές συσκευές συνδεδεμένες με το PLC ή την κατάσταση από αποθηκευμένη τιμή που έχουμε δημιουργήσει κάπου αλλού στο πρόγραμμα. Κάθε Coil μπορεί να αντιπροσωπεύει μία έξοδο για συσκευή συνδεδεμένη με το PLC ή κάποια εσωτερική τιμή για χρήση κάπου αλλού στο πρόγραμμα [27].



Εικόνα 5.1 Rung σε γλώσσα Ladder Logic (Ladder Logic Rung)

5.2 Βασικοί συμβολισμοί

Ας αναλύσουμε λοιπόν ορισμένους βασικούς συμβολισμούς που συναντάμε σε ένα πρόγραμμα γραμμένο σε Ladder Logic ώστε να αποκτήσουμε μία καλύτερη άποψη για το πως είναι δομημένη η γλώσσα αυτή.

Αρχικά έχουμε τους Bit Logic Operators (Εικόνα 5.2):

Normally Open (NO)

Κλειστή διαδρομή (ON) όταν η ανατιθέμενη τιμή bit είναι 1.

Δέχεται παράμετρο IN τύπου Bool.

Normally Closed (NC)

Κλειστή (ON) όταν η ανατιθέμενη τιμή bit είναι 0.

Δέχεται παράμετρο IN τύπου Bool.

Τελεστές AND, OR, XOR

Contacts σε σειρά δημιουργούν networks τύπου AND.

Contacts παράλληλα δημιουργούν networks τύπου OR.

NOT

Αντιστρέφει τη λογική κατάσταση της εισόδου σε αυτό άρα και τη "ροή ενέργειας".

Ροή ενέργειας προς το Contact => μη ύπαρξη ροής ενέργειας μετά από αυτό.

Μη ύπαρξη ροής ενέργειας προς το contact => ροή ενέργειας μετά από αυτό.

Output Coil

Αν ενεργοποιηθεί τότε το output bit γίνεται 1.

Αν δεν ενεργοποιηθεί τότε το output bit γίνεται 0.

Inverted Coil

Αν ενεργοποιείται τότε το output bit γίνεται 0.

Αν δεν ενεργοποιείται τότε το output bit γίνεται 1.

Set Coil

Όταν ενεργοποιείται η τιμή OUT γίνεται 1 και παραμένει έτσι.

Reset Coil

Όταν ενεργοποιείται η τιμή OUT γίνεται ξανά 0 και παραμένει έτσι.

SR Flip-Flop

Το Set υπερισχύει.

Αν τα Set και Reset είναι TRUE τότε το OUT είναι 1.

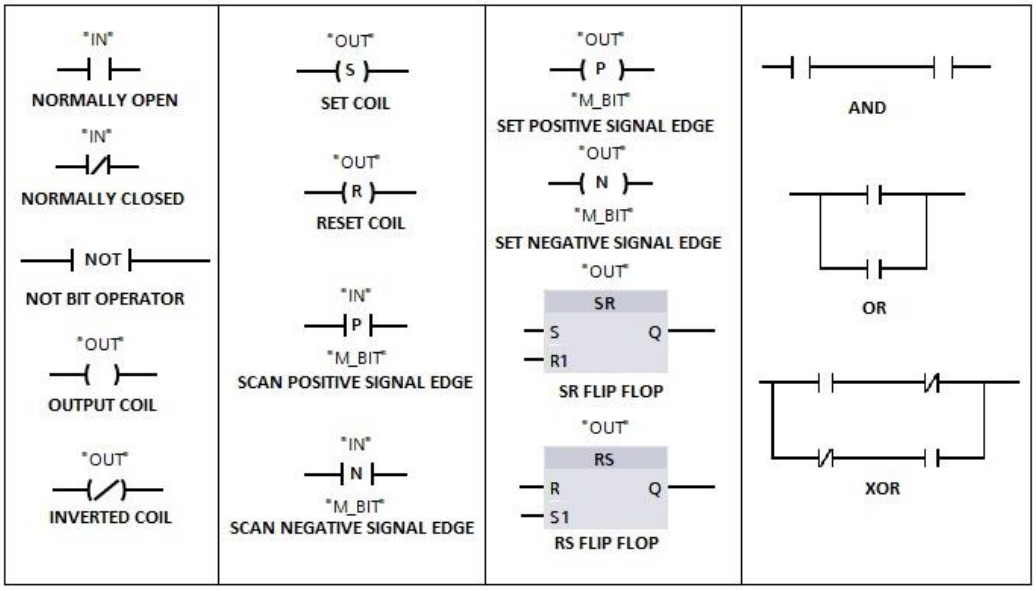
Το Q παίρνει την τιμή του OUT.

RS Flip-Flop

Το Reset υπερισχύει.

Αν τα Set και Reset είναι TRUE τότε το OUT είναι 0.

Το Q παίρνει την τιμή του OUT.



Εικόνα 5.2 Bit Logic Operators

Έπειτα έχουμε τους **Timers** (Εικόνα 5.3):

Η χρήση Timers αποσκοπεί στην δημιουργία καθυστερήσεων στο πρόγραμμα. Ο αριθμός Timers που μπορούμε να χρησιμοποιήσουμε καθορίζεται από το μέγεθος της μνήμης της CPU. Αυτό συμβαίνει για τον λόγο πως κάθε Timer χρησιμοποιεί ένα IEC_Timer Data Block μεγέθους 16 byte για την αποθήκευση των δεδομένων του.

Generate Pulse Timer TP

Παράγει σήμα για ένα προκαθορισμένο χρονικό διάστημα.

On-delay Timer TON

Θέτει το output Q ενεργό (ON) μετά από προκαθορισμένη χρονική καθυστέρηση.

Με το να αλλάξουμε το IN σε FALSE, ενώ ο timer τρέχει, θα έχουμε αποτέλεσμα να σταματήσει και να γίνει reset.

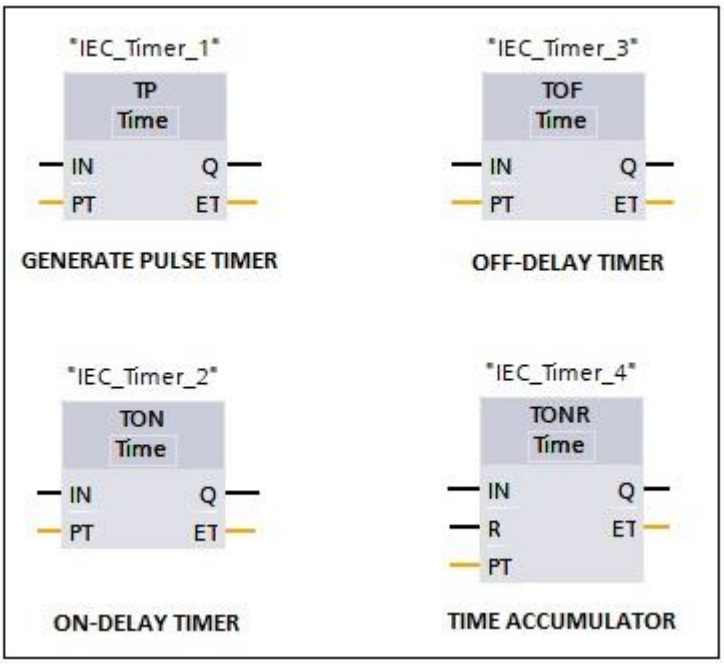
Off-delay Timer TOF

Θέτει το output Q ανενεργό (OFF) μετά από προκαθορισμένη χρονική καθυστέρηση.

Με το να αλλάξουμε το IN σε TRUE, ενώ ο timer τρέχει, θα έχουμε αποτέλεσμα να σταματήσει και να γίνει reset.

Time Accumulator TONR

Θέτει το output Q ενεργό (ON) μετά από προκαθορισμένη χρονική καθυστέρηση. Το Elapsed Time συσσωρεύεται με πολλαπλές χρονικές περιόδους μέχρι το R input χρησιμοποιηθεί προκειμένου να γίνει reset στο ET και στο Q. Με το να αλλάξουμε το IN σε FALSE, ενώ ο timer τρέχει, θα έχουμε αποτέλεσμα να σταματήσει αλλά δεν θα γίνει reset. Αν το επαναφέρουμε σε TRUE θα συνεχίσει από εκεί που σταμάτησε.



Εικόνα 5.3 Timers

Τα PT (preset time) και ET (elapsed time) αποθηκεύονται σε συγκεκριμένο IEC_TIMER Data Block με τύπο signed double integers και αντιπροσωπεύουν milliseconds. Για να καθορίσουμε τον χρόνο τον γράφουμε με τον τύπο T#2s_200ms.

Στη συνέχεια ας δούμε τους **Counters** (Εικόνα 5.4):

Κάθε Counter χρησιμοποιεί ένα Data Block για να κρατάει τα δεδομένα του.

Count Up CTU

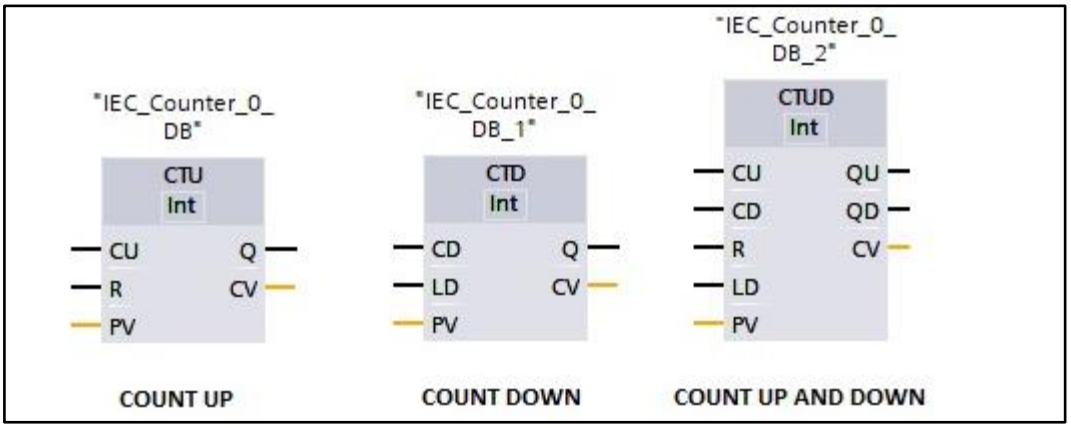
Ο μετρητής αυτός μετράει προς τα πάνω.

Count Down

Ο μετρητής αυτός μετράει προς τα κάτω, δηλαδή αντίστροφη μέτρηση.

Count Up and Down

Ο μετρητής αυτός μετράει και με τους δύο τρόπους.



Εικόνα 5.4 Counters

Στη συνέχεια ας δούμε τους **Comparators** (Εικόνα 5.5):

Μαθηματικές συγκρίσεις

Συγκρίνουν δύο τιμές ίδιου τύπου και αν ισχύει η συνθήκη (TRUE) έχουμε ενεργοποίηση.

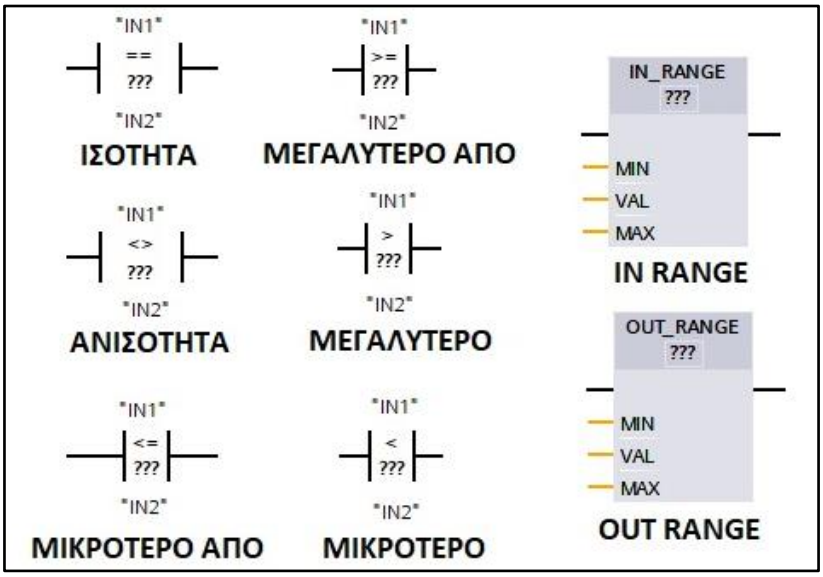
Στο πεδίο ??? επιλέγουμε τον τύπο.

In_range

Ελέγχει αν η τιμή είναι εντός του εύρους τιμών. Αν ισχύει τότε το output είναι TRUE.

Out_range

Ελέγχει αν η τιμή είναι εκτός του εύρους τιμών. Αν ισχύει τότε το output είναι TRUE.



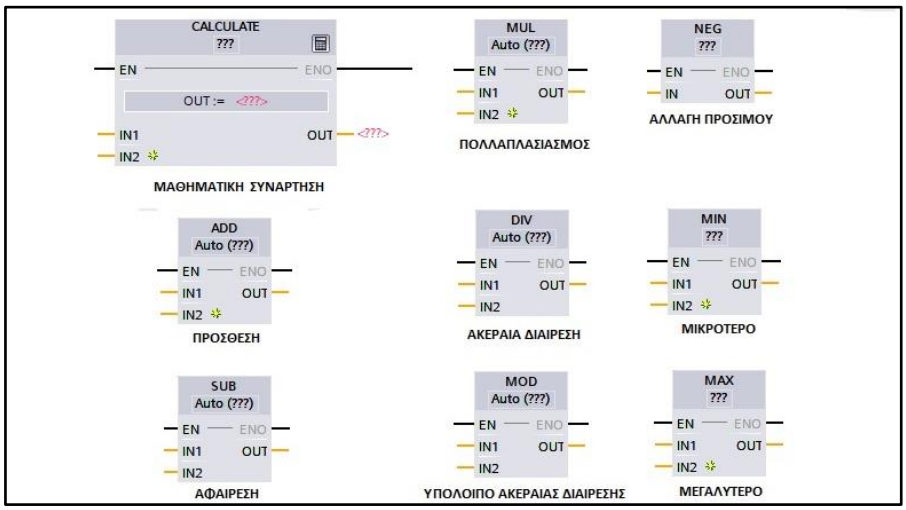
Εικόνα 5.5 Comparators

Στη συνέχεια έχουμε **Math Instructions** (Εικόνα 5.6)

Να αναφέρουμε πως δεν θα εστιάσουμε σε όλα αλλά υπάρχουν λειτουργίες για όλες τις μαθηματικές πράξεις όπως πρόσθεση, αφαίρεση, πολλαπλασιασμός και πολλά άλλα μέχρι τριγωνομετρικές πράξεις ώστε να καλύπτονται όλες οι ανάγκες που μπορεί να προκύψουν.

Calculate

Η πιο σημαντική ίσως λειτουργία είναι αυτή που μας επιτρέπει την δημιουργία συνάρτησης.



Εικόνα 5.6 Math Instructions

Στη συνέχεια έχουμε **Move Instructions** (Εικόνα 5.7)

Move

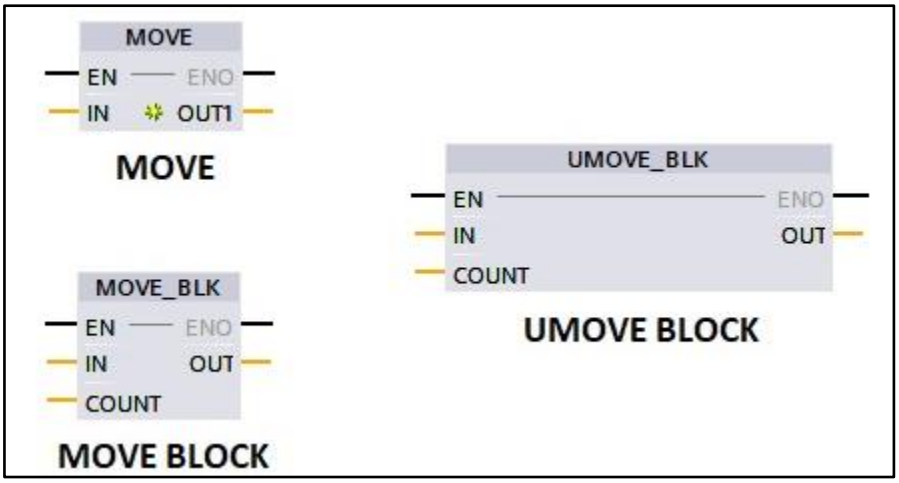
Αντιγράφει τιμή από μία διεύθυνση σε μία άλλη.

Move Block

Αντιγράφει ένα Block από μία διεύθυνση σε μία άλλη.(διακοπτόμενη λειτουργία)

Umove Block

Αντιγράφει ένα Block από μία διεύθυνση σε μία άλλη.(μη διακοπτόμενη λειτουργία)



Εικόνα 5.7 Move Instructions

Στη συνέχεια έχουμε **Conversion Instructions** (Εικόνα 5.8)

Convert

Μετατρέπει ένα στοιχείο από ένα τύπο δεδομένων σε άλλο τύπο δεδομένων.

Οι τύποι δεδομένων επιλέγονται στα πεδία με “???”.

Round

Μετατρέπει μία πραγματική τιμή σε ακέραια εκτελώντας στρογγυλοποίηση.

Ceil

Μετατρέπει μία πραγματική τιμή στον πλησιέστερο μεγαλύτερο από αυτή ή ίσο ακέραιο.

Floor

Μετατρέπει μία πραγματική τιμή στον πλησιέστερο μικρότερο από αυτή ή ίσο ακέραιο.

Truncate

Μετατρέπει μία πραγματική τιμή σε ακέραια. Το τμήμα του αριθμού μετά την υποδιαστολή μετατρέπεται σε μηδέν.

Scale

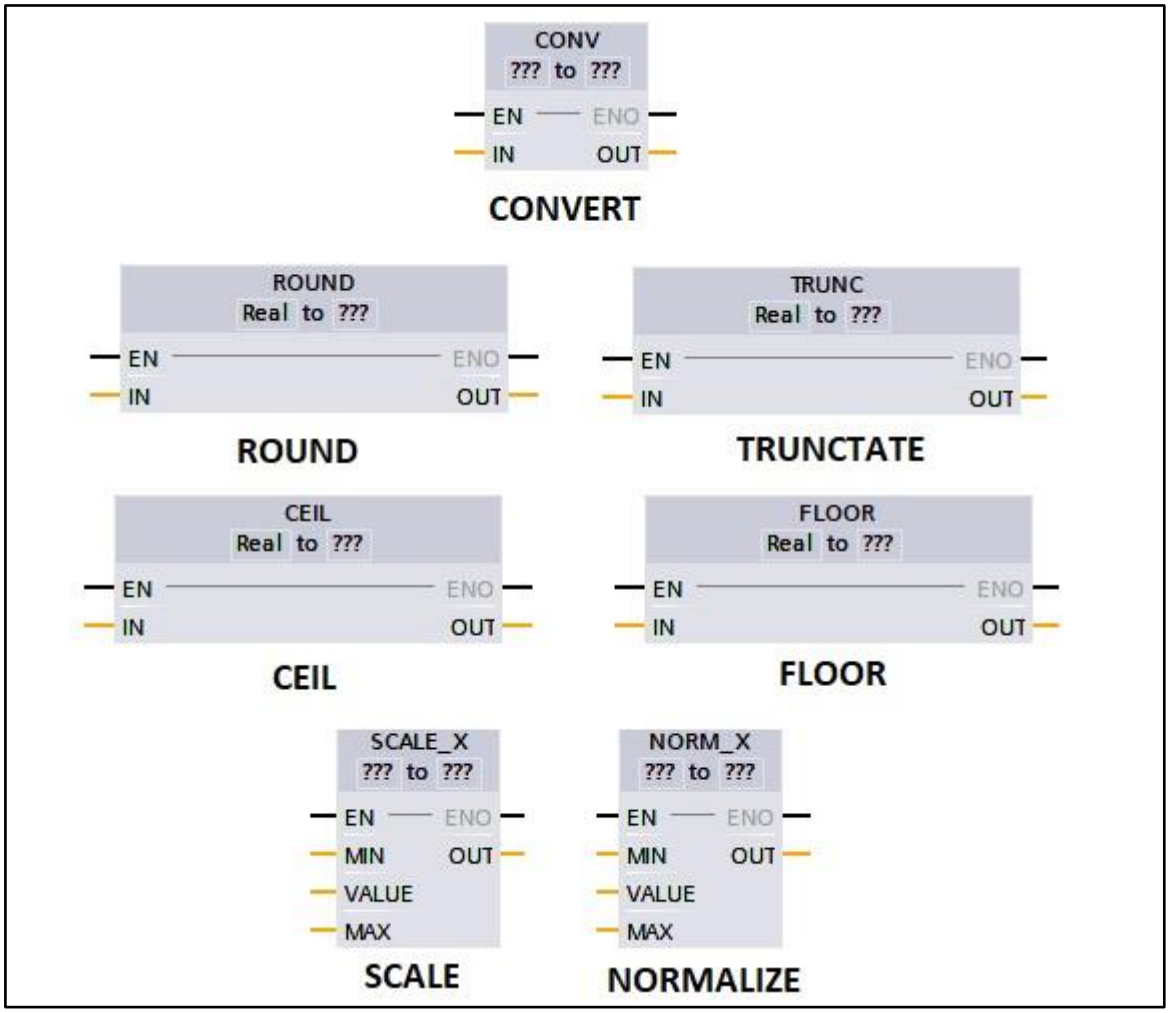
Δέχεται πραγματική τιμή VALUE όπου 0.0 <= VALUE <= 1.0 και την εξάγει σε αναλογία μεταξύ των τιμών MIX και MAX.

Ο τύπος που χρησιμοποιεί είναι OUT = VALUE (MAX - MIN) + MIN.

Normalize

Δέχεται τιμή VALUE όπου MIN<=VALUE<=MAX και την ανάγει σε τιμή OUT, όπου 0.0<=OUT<= 1.0.

Ο τύπος που χρησιμοποιεί είναι OUT = (VALUE - MIN) / (MAX – MIN).



Εικόνα 5.8 Conversion Instructions

Στη συνέχεια έχουμε **Program Contro**l **Instructions** (Εικόνα 5.9)

Label

Προορισμός για τις εντολές JMP or JMPN

Jump

Αν υπάρχει κλειστή διαδρομή προς το Coil τότε η εκτέλεση του προγράμματος συνεχίζεται από από την πρώτη εντολή μετά το συγκεκριμένο Label.

Jump N

Αν δεν υπάρχει κλειστή διαδρομή προς το Coil τότε η εκτέλεση του προγράμματος συνεχίζεται από από την πρώτη εντολή μετά το συγκεκριμένο Label.

Return

Τερματίζει την εκτέλεση του block.

Jump List

Η εντολή JMP_LIST λειτουργεί ως επιλογέας για το ποιό άλμα θα εκτελέσουμε.

Ανάλογα με την τιμή της εισόδου K έχουμε ένα άλμα στην αντίστοιχη ετικέτα προγράμματος.

Η εκτέλεση του προγράμματος συνεχίζεται από από την πρώτη εντολή μετά το συγκεκριμένο Label.

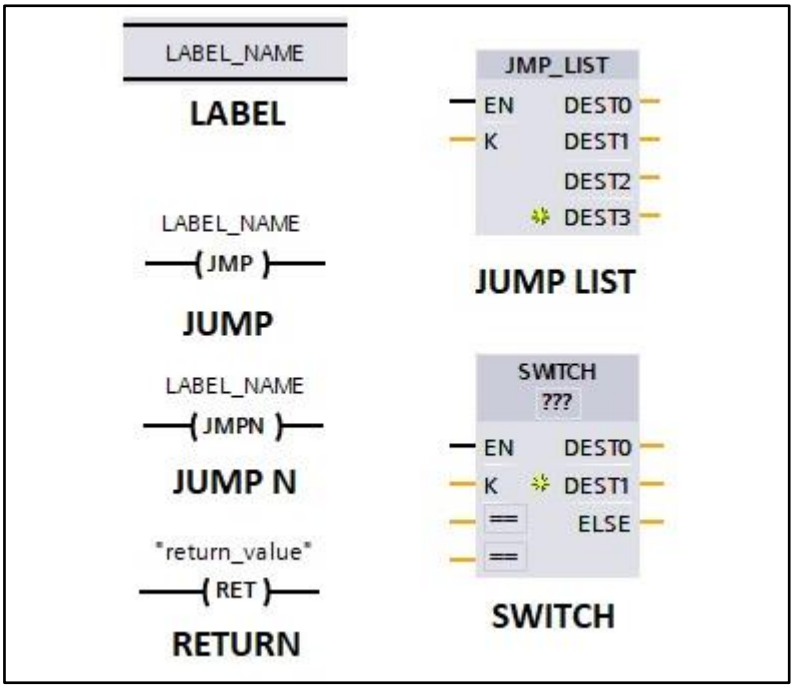
Switch

Η εντολή Switch λειτουργεί ως επιλογέας για το ποιό άλμα θα εκτελέσουμε.

Αναλόγως το αποτέλεσμα σύγκρισης μεταξύ της τιμής K και των τιμών που αναθέτουμε στα inputs θα επιλεγεί το αντίστοιχο άλμα (1ο, 2ο ,3ο κ.τ.λ.).

Αν καμία από τις συγκρίσεις δεν είναι αληθής θα πραγματοποιειθέι το άλμα για το label που αναθέτουμε στο πεδίο ELSE.

Η εκτέλεση του προγράμματος συνεχίζεται από από την πρώτη εντολή μετά το συγκεκτιμένο Label.



Εικόνα 5.9 Program Control Instructions

Στη συνέχεια έχουμε **Word Logic Instructions** (Εικόνα 5.10)

AND, OR, XOR

AND: Λογικό AND

OR: Λογικό OR

XOR: Λογικό exclusive OR

INV

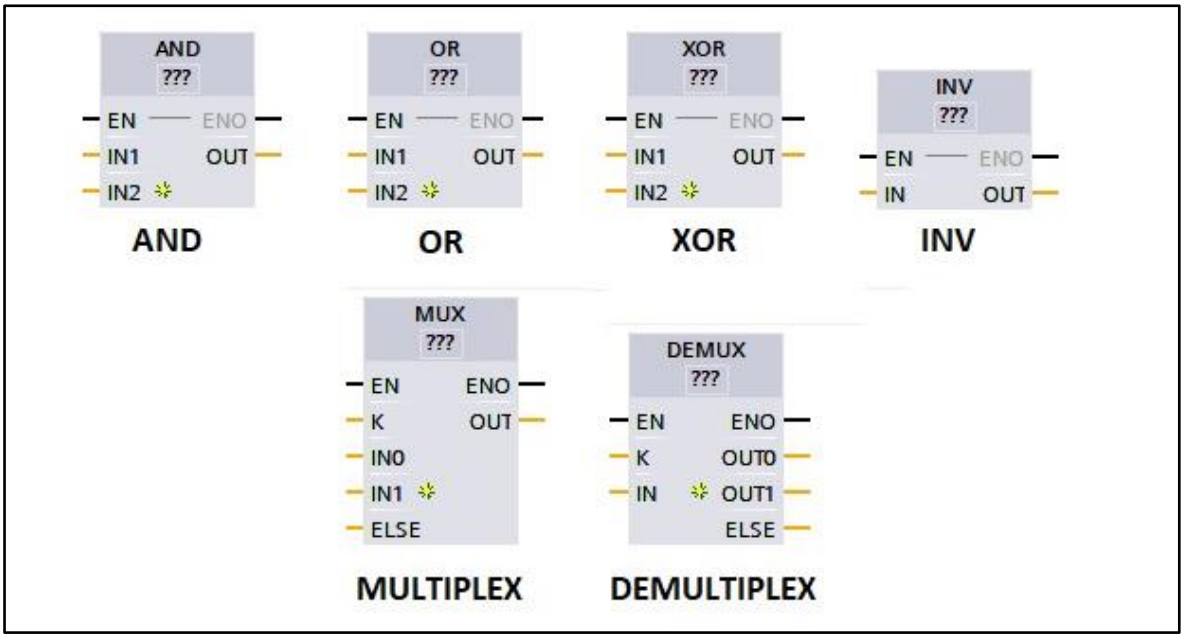
Πραγματοποιεί αντιστροφή στον δυαδικό αριθμό (αλλάζει το 0 σε 1 και το 1 σε 0).

Multiplex

Αντιγράφει την τιμή από κάποια παράμετρο input αναλόγως την τιμή του K.

Demultiplex

Αντιγράφει την τιμή από την παράμετρο input σε κάποια από τις παραμέτρους output αναλόγως την τιμή του K.



Εικόνα 5.10 Word Logic Instructions

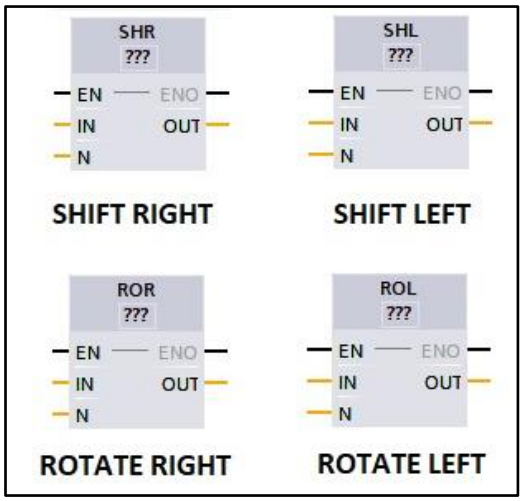
Στη συνέχεια έχουμε **Shift and Rotate Operations** (Εικόνα 5.11)

Shift

Μετατόπιση της αλληλουχίας των bit προς τα δεξιά(SHR) ή αριστερά(SHL). Το αποτέλεσμα αποδίδεται στην παράμετρο OUT. Η παράμετρος N υποδεικνύει τον αριθμό από θέσεις bit οι οποίες μετατοπίστηκαν.

Rotate

Περιστροφή της αλληλουχίας των bit προς τα δεξιά(SHR) ή αριστερά(SHL). Το αποτέλεσμα αποδίδεται στην παράμετρο OUT. Η παράμετρος N υποδεικνύει τον αριθμό από θέσεις bit οι οποίες περιστράφηκαν.



Εικόνα 5.11 Shift and Rotate Operations

Όλες οι πληροφορίες σχετικά με τις εντολές που παρουσιάστηκαν αντλήθηκαν από το εγχειρίδιο χρήσης του S7-1200 (οι εντολές εφαρμόζονται και στα υπόλοιπα μοντέλα PLC) [26].

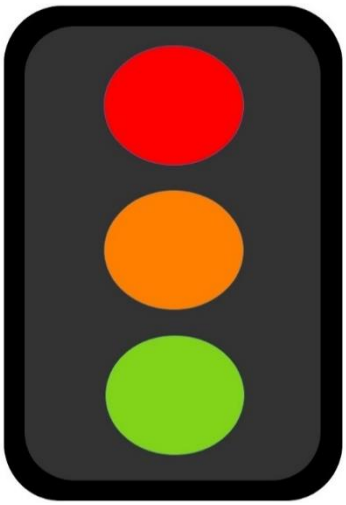
Αυτές ήταν ορισμένες από τις βασικότερες εντολές της Ladder Logic. Προφανώς σαν κάθε γλώσσα προγραμματισμού υπάρχουν πολλές ακόμη επιπρόσθετες λειτουργίες (π.χ. εντολές διαγνωστικών, δικτύων κ.τ.λ.). Η παρούσα εργασία εστιάζει στις παραπάνω οι οποίες είναι οι πλέον πιο συχνά συναντώμενες και αποτελούν βασικό κορμό ώστε να είναι κάποιος σε θέση να κατανοεί ή ακόμη και να δημιουργεί αυτοματισμούς με το λογισμικό TIA PORTAL.

Κεφάλαιο 6 Εφαρμογές βιομηχανικών αυτοματισμών

Σε προηγούμενα κεφάλαια γνωρίσαμε το λογισμικό που θα χρησιμοποιήσουμε και πραγματοποιήθηκε εισαγωγή στη γλώσσα προγραμματισμού Ladder Logic. Στο κεφάλαιο αυτό θα εφαρμόσουμε τις γνώσεις αυτές σε διάφορες μελέτες περιπτώσεων (case studies) εξηγώντας παράλληλα τον τρόπο σκέψης και εργασίας μας.

6.1 Case Study A: Προγραμματισμός Αυτοματισμού Φωτεινού Σηματοδότη

Ως το πρώτο μας case study θα ασχοληθούμε με τον προγραμματισμό ενός φωτεινού σηματοδότη. Αρχικά ας αναλογιστούμε τον τρόπο με τον οποίο λειτουργεί ένας κλασικός φωτεινός σηματοδότης (Εικόνα 6.1) έτσι ώστε να αναλύσουμε την μέθοδο και τον τρόπο σκέψης τον οποίο θα ακολουθήσουμε στον σχεδιασμό των Rung.



Εικόνα 6.1 Case study A: Απεικόνιση συστήματος (Case Study A: System display)

Τα κύρια μέλη που το αποτελούν είναι:

- Κόκκινη λυχνία
- Πορτοκαλί λυχνία
- Πράσινη λυχνία

Η λογική λοιπόν που πρέπει να ακολουθήσουμε είναι τα εξής:

- Η πράσινη λυχνία ανάβει για ορισμένο χρονικό διάστημα.
- Αφού σβήσει η πρώτη, η πορτοκαλί χρώματος προειδοποιητική λυχνία ανάβει για λιγότερο χρόνο.
- Με την πάροδο του χρόνου αυτού σβήνει η πορτοκαλί και ανάβει η κόκκινη λυχνία.
- Τέλος, αφού περάσει και ο ορισμένος χρόνος της κόκκινης λυχνίας η διαδικασία ξεκινάει από την αρχή σε επανάληψη.

Στον Πίνακα 6.1 αναγράφονται οι απαραίτητες είσοδοι και έξοδοι για την υλοποίηση του εν λόγω προγράμματος.

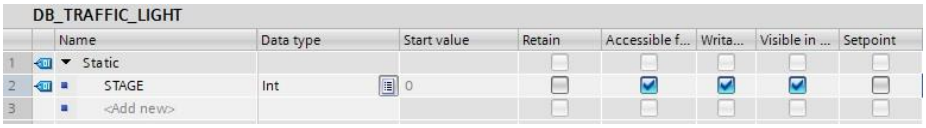
Πίνακας 6.1: Πίνακας εισόδων/εξόδων Case Study A (Case Study A Input / Output table)

ΟΝΟΜΑ	ΕΙΔΟΣ	ΘΕΣΗ
GREEN_LIGHT	ΕΞΟΔΟΣ	Q0.0
ORANGE LIGHT	ΕΞΟΔΟΣ	Q0.1
RED_LIGHT	ΕΞΟΔΟΣ	Q0.2

Στο σχεδιασμό μας ας ορίσουμε πως τα χρονικά διαστήματα θα είναι 15, 5 και 15 δευτερόλεπτα για την πράσινη, πορτοκαλί και κόκκινη λυχνία αντίστοιχα.

Ξεκινώντας θα ορίσουμε τα Data Blocks και Tags που θα χρησιμοποιήσουμε.

Όπως βλέπουμε δημιουργούμε το Data Block DB_TRAFFIC_LIGHT (Εικόνα 6.2) μέσα στο οποίο θέτουμε μία τιμή Integer και αρχική τιμή 0 με όνομα STAGE. Επίσης θέτουμε τρεις τιμές τύπου Boolean στον πίνακα των Tags με ονομασίες GREEN_LIGHT, ORANGE_LIGHT και RED_LIGHT σαν έξοδοι Q0.0, Q0.1, Q0.2 αντίστοιχα για κάθε λυχνία (Εικόνα 6.3).



DB_TRAFFIC_LIGHT

	Name	Data type	Start value	Retain	Accessible f...	Writa...	Visible in ...	Setpoint
1	Static							
2	STAGE	Int	0					
3	<Add new>							

Εικόνα 6.2 Case study A: Data Block

PLC tags

	Name	Tag table	Data type	Address
	GREEN_LIGHT	Default tag table	Bool	%Q0.0
	ORNGE_LIGHT	Default tag table	Bool	%Q0.1
	RED_LIGHT	Default tag table	Bool	%Q0.2
	<Add new>			

Εικόνα 6.3 Case study A: Tags

Στην Εικόνα 6.4 μπορούμε να δούμε τον σχεδιασμό του αυτοματισμού σε γλώσσα Ladder Logic.

Έχουμε σχεδιάσει το πρόγραμμά μας σε τρία Networks:

Network 1: εργαζόμαστε για τον αυτοματισμό του πράσινου σηματοδότη

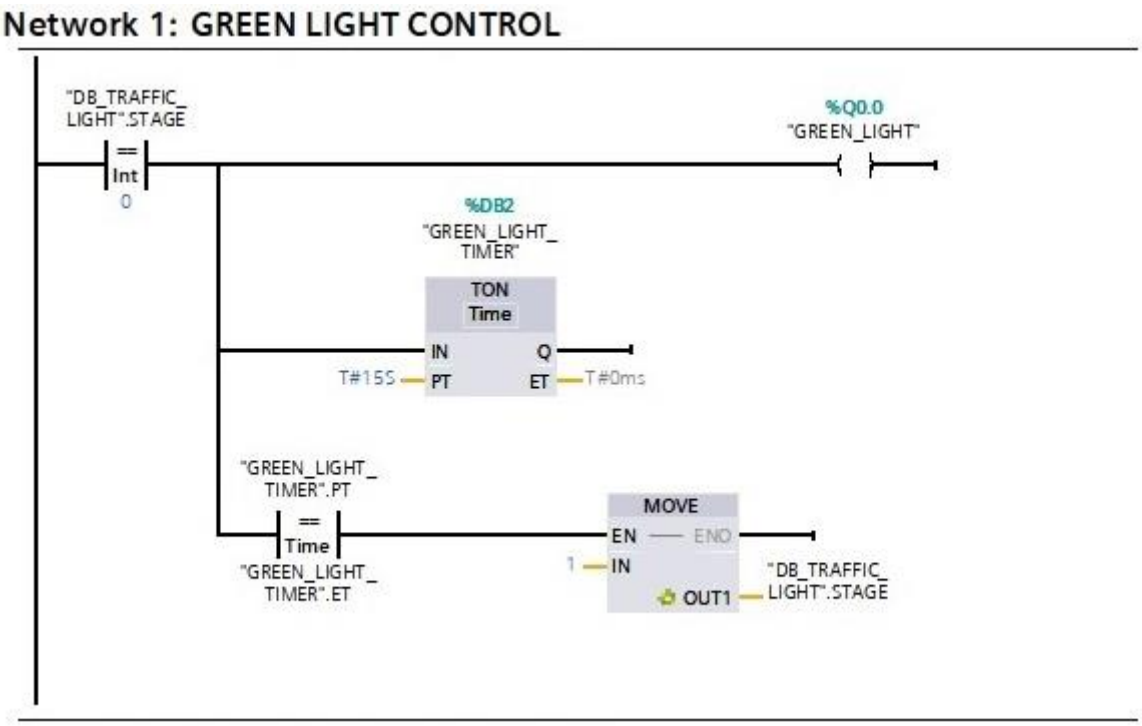
Network 2: εργαζόμαστε για τον αυτοματισμό του πορτοκαλί σηματοδότη

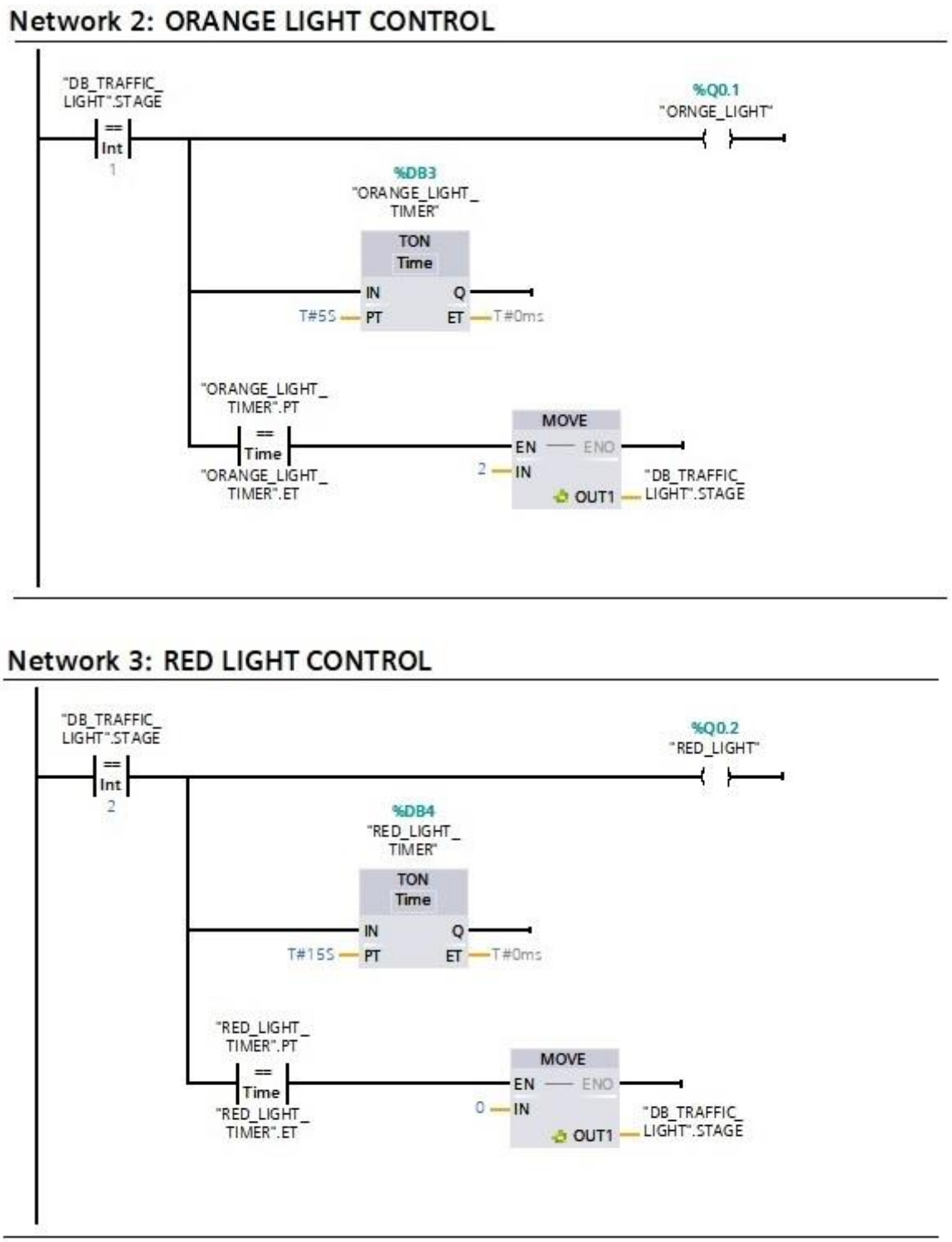
Network 3: εργαζόμαστε για τον αυτοματισμό του κόκκινου σηματοδότη

Η λογική προγραμματισμού έχει ως εξής:

- Χρησιμοποιούμε το STAGE σαν μετρητή για να γνωρίζουμε σε ποιό στάδιο λειτουργίας βρισκόμαστε.
- Με έναν comparator ελέγχου ισότητας τιμών βλέπουμε εάν η τιμή του stage είναι 0. Όσο αυτή η συνθήκη ισχύει ενεργοποιείται αποκλειστικά το rung στο Network1
- Παράλληλα θέτουμε έναν timer τύπου TON (Data Block GREEN_LIGHT_TIMER) ο οποίος μετράει για 15 δευτερόλεπτα.
- Για όσο διαρκεί το χρονικό περιθώριο αυτό δίνεται σήμα στην έξοδο Q0.0 άρα ο πράσινος σηματοδότης παραμένει αναμμένος.
- Ταυτόχρονα, με άλλον έναν comparator ισότητας ελέγχουμε αν η τιμή PT του Data Block GREEN_LIGHT_TIMER ταυτίζεται με την τιμή ET του ίδιου Data Block (αν δηλαδή στον Timer το Preset Time είναι ίσο με το Elapsed Time).
- Τη στιγμή που θα συμβεί αυτό σημαίνει ότι ο χρόνος των 15'' έχει εξαντληθεί. Οπότε η έξοδος Q0.0 θα απενεργοποιηθεί και το STAGE θα λάβει την τιμή 1 μέσω της εντολής Move.

- Πλέον με την τιμή του μετρητή μας STAGE να ισούται με 1 θα ενεργοποιηθεί το Network 2.
- Εδώ να σημειωθεί ότι η λογική που εφαρμόστηκε για τον πράσινο σηματοδότη είναι η ίδια που εφαρμόζεται και για τους επόμενους δύο. Η μόνες ουσιαστικές αλλαγές είναι τα Data Blocks των TON Timers οι έξοδοι Q.
- Τέλος βλέπουμε πως μετά το πέρας του χρόνου και του κόκκινου σηματοδότη στο STAGE θα αποδοθεί η τιμή 0 αυτή τη φορά έτσι ώστε να ενεργοποιηθεί ξανά το Network1 οπότε να ξεκινήσει από την αρχή η όλη διαδικασία.



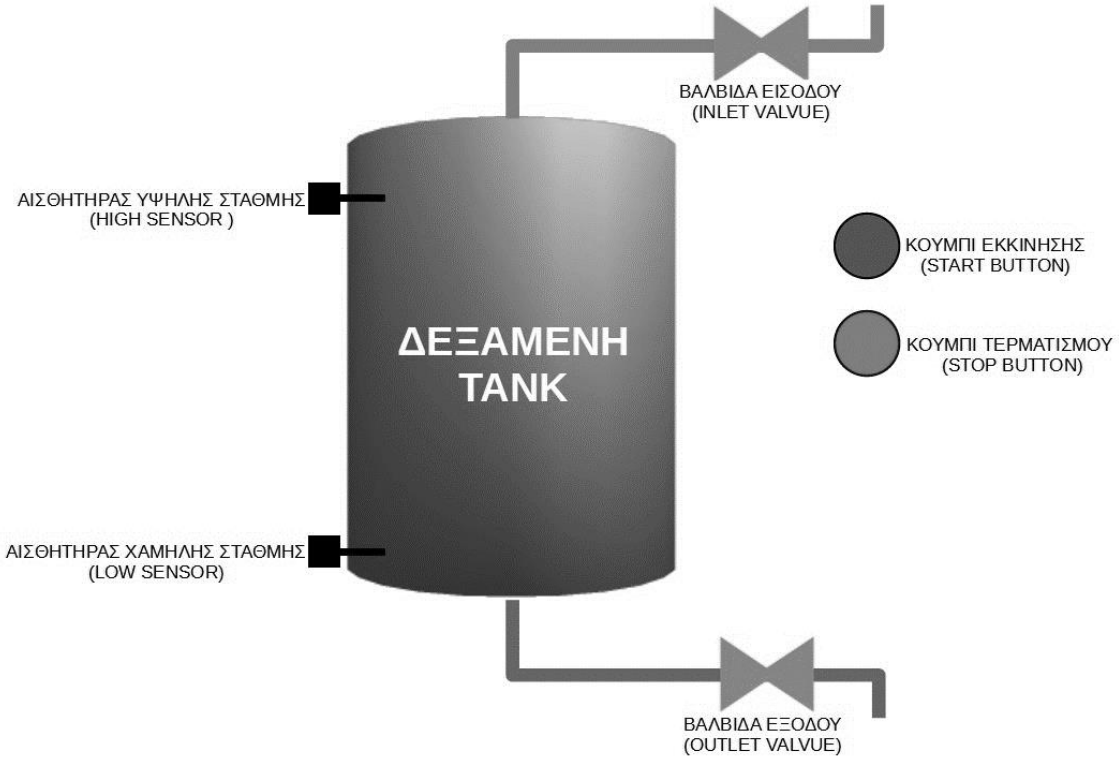


Εικόνα 6.4 Case study A: Ladder Logic

6.2 Case Study B: Προγραμματισμός Αυτοματισμού Ελέγχου Στάθμης Δεξαμενής

Στο δεύτερό case study θα ασχοληθούμε με τον προγραμματισμό ενός PLC το οποίο θα ελέγχει τη στάθμη υγρού σε μία δεξαμενή. Πρόκειται για μία αρκετά χρήσιμη εφαρμογή καθώς όπως μπορούμε να φανταστούμε συναντάται σε μία πληθώρα περιπτώσεων. Μερικά ενδεικτικά παραδείγματα είναι έλεγχος στάθμης νερού, έλεγχος στάθμης σε δεξαμενές καυσίμων, εφαρμογές σε βιομηχανίες επεξεργασίας τροφίμων κ.α.

Για να κατανοήσουμε καλύτερα το πρόβλημα το οποίο καλούμαστε να λύσουμε μπορούμε να δούμε στην Εικόνα 6.5 το σύστημά μας σαν σύνολο.



Εικόνα 6.5 Case study B: Απεικόνιση συστήματος (Case Study B: System display)

Βλέπουμε πως τα κύρια μέλη που το αποτελούν είναι:

- Η δεξαμενή την οποία καλούμαστε να ελέγξουμε
- Τα κουμπιά εκκίνησης και τερματισμού της διαδικασίας
- Οι βαλβίδες εισόδου και εξόδου του υγρού αντίστοιχα
- Οι αισθητήρες στάθμης του υγρού

Η λογική με την οποία εργαζόμαστε:

- Το κουμπί εκκίνησης ξεκινάει την διαδικασία και το κουμπί τερματισμού την σταματάει
- Ο πρώτος αισθητήρας ανιχνεύει τη χαμηλή στάθμη και δίνεται σήμα ενεργοποίησης στην βαλβίδα εισόδου ώστε να γεμίσει η δεξαμενή.
- Όταν ο δεύτερος αισθητήρας ανιχνεύει υψηλή στάθμη δίνεται σήμα στην βαλβίδα εισόδου να σταματήσει και παράλληλα ενεργοποιείται η βαλβίδα εξόδου ώστε να αδειάσει η δεξαμενή.
- Αν η στάθμη πέσει σε σημείο που ανιχνεύσουμε χαμηλή στάθμη η διαδικασία επαναλαμβάνεται.

Στον Πίνακα 6.2 αναγράφονται οι απαραίτητες είσοδοι και έξοδοι για την υλοποίηση του εν λόγω προγράμματος.

Πίνακας 6.2: Πίνακας εισόδων/εξόδων Case Study B (Case Study B Input / Output table)

ΟΝΟΜΑ	**ΕΙΔΟΣ**	**ΘΕΣΗ**
START	ΕΙΣΟΔΟΣ	I0.0
STOP	ΕΙΣΟΔΟΣ	I0.1
LOW	ΕΙΣΟΔΟΣ	I1.0
HIGH	ΕΙΣΟΔΟΣ	I1.1
ON	ΕΞΟΔΟΣ	Q0.0
VALVUE_IN	ΕΞΟΔΟΣ	QO.1
VALVUE_OUT	ΕΞΟΔΟΣ	Q0.2

Στην Εικόνα 6.6 βλέπουμε και τις τιμές μας όπως έχουν δηλωθεί ως Tags στο πρόγραμμά μας.

PLC tags

		Name	Tag table	Data type	Address
1		START	Default tag table	Bool	%I0.0
2		STOP	Default tag table	Bool	%I0.1
3		LOW	Default tag table	Bool	%I1.0
4		HIGH	Default tag table	Bool	%I1.1
5		ON	Default tag table	Bool	%Q0.0
6		VALVUE_IN	Default tag table	Bool	%Q0.1
7		VALVUE_OUT	Default tag table	Bool	%Q0.2
8		<Add new>			

Εικόνα 6.6 Case study B: Tags

Στην Εικόνα 6.7 μπορούμε να δούμε τον σχεδιασμό του αυτοματισμού σε γλώσσα Ladder Logic. Παρατηρούμε ότι σχεδιάσαμε τρία Networks.

Network 1: βλέπουμε τη λειτουργία on/off της διαδικασίας

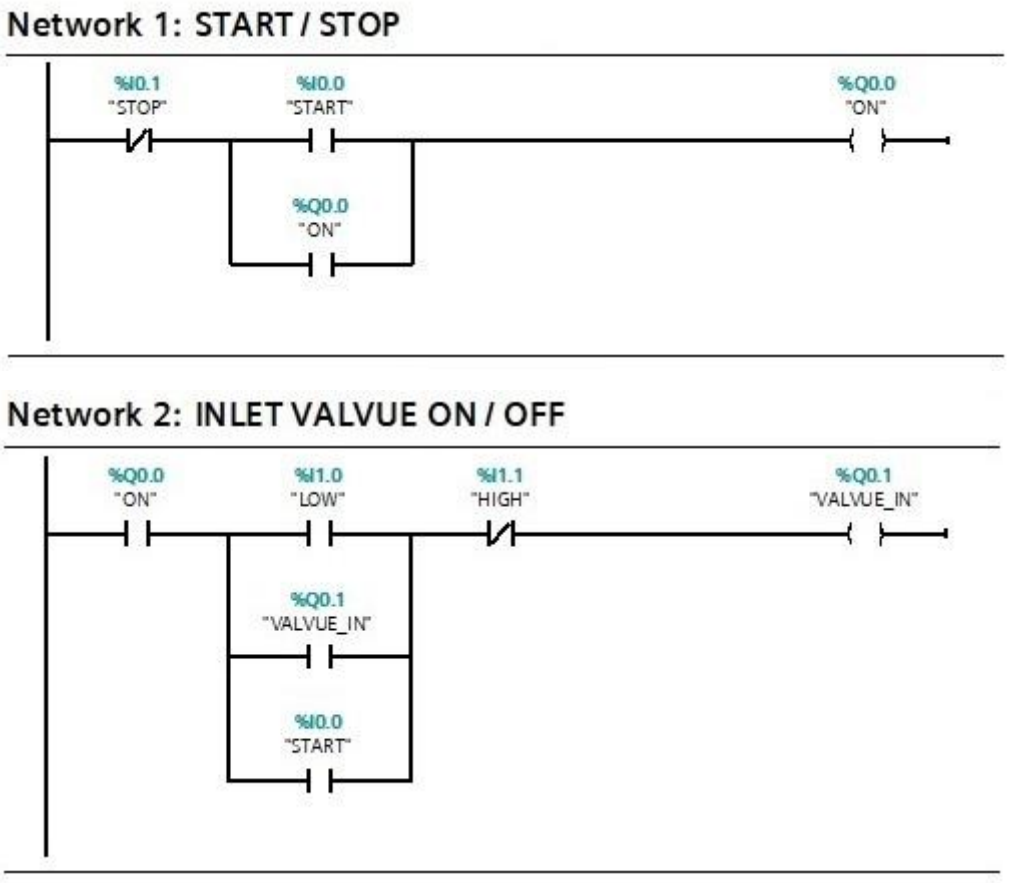
Network 2: ελέγχουμε τη λειτουργία της βαλβίδας εισόδου

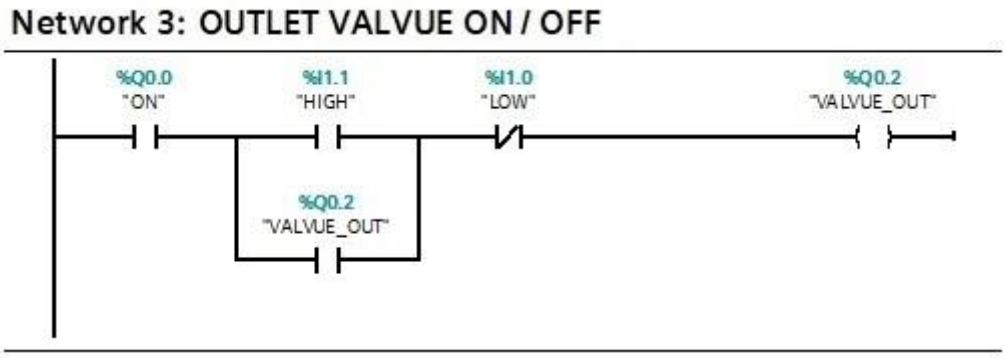
Network 3: ελέγχουμε τη λειτουργία της βαλβίδας εξόδου

Η λογική προγραμματισμού έχει ως εξής:

- Στο Network 1, με το πάτημα του κουμπιού εκκίνησης START θα δοθεί στιγμιαίο σήμα (το NO contact START θα πάρει τιμή TRUE) και θα ενεργοποιηθεί το Coil και η έξοδος ON.
- Με το ON ενεργοποιημένο σημαίνει ότι η διαδικασία μας βρίσκεται σε εξέλιξη. Για τον τερματισμό της μπορούμε να πατήσουμε το κουμπί STOP άρα το NC Contact θα πάρει την τιμή TRUE οπότε και θα διακοπεί η διαδικασία, αφού θα απενεργοποιηθεί το Coil ON.
- Στο Network 2 εφόσον η διαδικασία είναι ενεργή (ON==TRUE) και με τη στιγμιαία ενεργοποίηση του START θα δοθεί σήμα και θα έχουμε και ενεργοποίηση της βαλβίδας εισόδου VALVUE_IN.

- Βλέπουμε πως για την απενεργοποίηση της βαλβίδας εισόδου αρκεί να δοθεί σήμα από τον αισθητήρα υψηλής στάθμης HIGH, οπότε το NC Contact θα πάρει την τιμή TRUE άρα και θα διακόψει την λειτουργία του Coil VALVUE_IN.
- Με τη σειρά του το γεγονός αυτό θα ανοίξει την βαλβίδα εξόδου VALVUE_OUT καθώς το Coil στο Network 3 θα ενεργοποιηθεί εφόσον πλέον θα έχουμε τα ON και HIGH με τιμή TRUE.
- Για το κλείσιμο της βαλβίδας εξόδου αρκεί να ενεργοποιηθεί ο αισθητήρας χαμηλής στάθμης LOW. Έτσι όταν αυτός πάρει την τιμή TRUE το αντίστοιχο NC Contact στο Network 3 θα απενεργοποιήσει το Coil της βαλβίδας εξόδου ενώ ταυτόχρονα στο Network 2 το NO Contact LOW μαζί με το NC Contact HIGH (η στάθμη έχει πέσει άρα έχει ξανά την τιμή FALSE) θα ενεργοποιήσουν το Coil τη βαλβίδας εισόδου VALVUE_IN.
- Έτσι η διαδικασία επαναλαμβάνεται γεμίζοντας και αδειάζοντας την δεξαμενή μέχρις ότου την τερματίσουμε με τη χρήση του STOP.



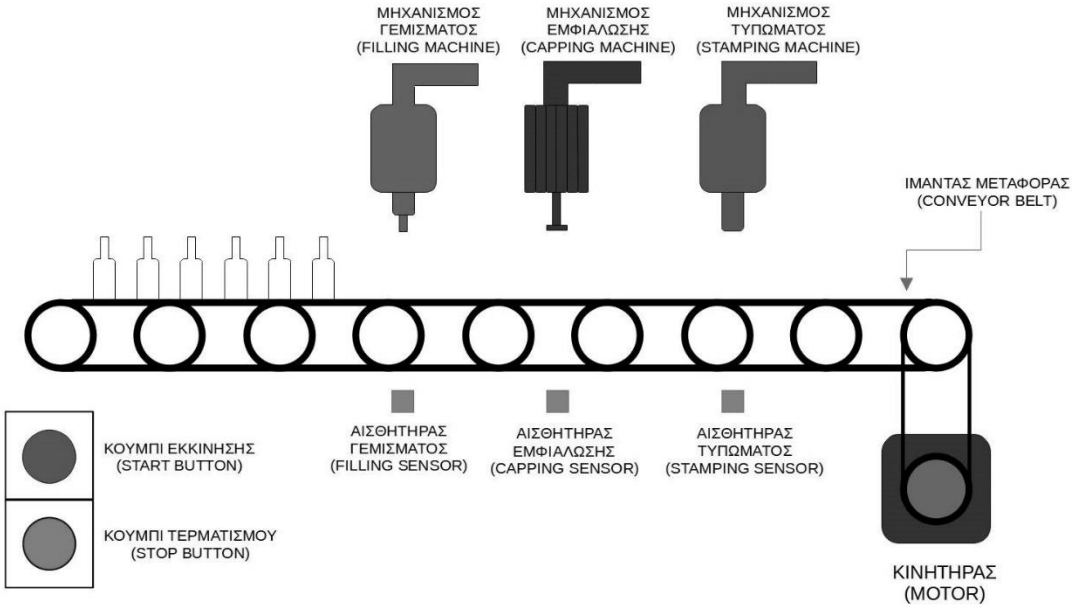


Εικόνα 6.7 Case study B: Ladder Logic

6.3 Case Study C: Προγραμματισμός Αυτοματισμού Εμφιάλωσης

Για επόμενο case study θα ασχοληθούμε με τον προγραμματισμό αυτοματισμού τμήματος μιας διαδικασίας εμφιάλωσης. Όπως καταλαβαίνουμε πρόκειται για μια διαδικασία με αρκετές εφαρμογές όπως στη βιομηχανία τροφίμων, παρασκευή ποτών, βιομηχανία φαρμάκων και πολλά άλλα.

Στην Εικόνα 6.8 μπορούμε να δούμε μια σχηματική απεικόνιση του συστήματος με το οποίο καλούμαστε να εργαστούμε.



Εικόνα 6.8 Case study C: Απεικόνιση συστήματος (Case Study C: System display)

Τα κύρια μέλη που το αποτελούν είναι:

- Ο ιμάντας μεταφοράς
- Ο κινητήρας του ιμάντα
- Τα κουμπιά εκκίνησης και τερματισμού της διαδικασίας
- Τρεις αισθητήρες ελέγχου των σταδίων της διαδικασίας
- Τρία μηχανήματα τα οποία εκτελούν τα βήματα της διαδικασίας (γέμισμα – εμφιάλωση - τοποθέτηση τυπώματος)

Η εφαρμογή της διαδικασίας ακολουθεί τα παρακάτω βήματα:

- Με τα κουμπιά εκκίνησης και τερματισμού ελέγχουμε την εναρξη και το τέλος της διαδικασίας αντίστοιχα.
- Με την έναρξη της διαδικασίας εκκινείται ο κινητήρας ώστε να είναι σε κίνηση ο ιμάντας μεταφοράς.
- Καθώς ένας αισθητήρας αντιλαμβάνεται ένα μπουκάλι η λειτουργία του κινητήρα διακόπτεται και εκκινείται το αντίστοιχο μηχάνημα.
- Ο πρώτος αισθητήρας και το πρώτο μηχάνημα είναι για το στάδιο του γεμίσματος, στη συνέχεια ο δεύτερος και το δεύτερο μηχάνημα για το στάδιο της εμφιάλωσης και τέλος το επόμενο ζεύγος αισθητήρα και μηχανήματος είναι για το στάδιο της τοποθέτησης τυπώματος(ετικέτας) στο μπουκάλι.
- Με το πέρας της διαδικασίας κάθε μηχανήματος διακόπτεται η λειτουργία του και εκκινείται ξανά ο κινητήρας ώστε να προχωρήσουμε στο επόμενο στάδιο.

Στον Πίνακα 6.3 αναγράφονται οι απαραίτητες είσοδοι και έξοδοι για την υλοποίηση του εν λόγω προγράμματος.

Πίνακας 6.3: Πίνακας εισόδων/εξόδων Case Study C (Case Study C Input / Output table)

ΟΝΟΜΑ	**ΕΙΔΟΣ**	**ΘΕΣΗ**
START	ΕΙΣΟΔΟΣ	I0.0
STOP	ΕΙΣΟΔΟΣ	I0.1
MOTOR_ON	ΕΞΟΔΟΣ	Q0.0
FILLING_SENSOR	ΕΙΣΟΔΟΣ	I1.0
FILLING_MACHINE	ΕΞΟΔΟΣ	Q0.1
CAPPING_SENSOR	ΕΙΣΟΔΟΣ	I1.1
CAPPING_MACHINE	ΕΞΟΔΟΣ	Q0.2
STAMPING_SENSOR	ΕΙΣΟΔΟΣ	I1.2
STAMPING_MACHINE	ΕΞΟΔΟΣ	Q0.3

Στην Εικόνα 6.9 βλέπουμε και τις τιμές μας όπως έχουν δηλωθεί ως Tags στο πρόγραμμά μας.

PLC tags

	Name	Tag table	Data type	Address
1	START	Default tag table	Bool	%I0.0
2	STOP	Default tag table	Bool	%I0.1
3	MOTOR_ON	Default tag table	Bool	%Q0.0
4	FILLING_SENSOR	Default tag table	Bool	%I1.0
5	FILLING_MACHINE	Default tag table	Bool	%Q0.1
6	CAPPING_SENSOR	Default tag table	Bool	%I1.1
7	CAPPING_MACHINE	Default tag table	Bool	%Q0.2
8	STAMPING_SENSOR	Default tag table	Bool	%I1.2
9	STAMPING_MACHINE	Default tag table	Bool	%Q0.3
10	<Add new>			

Εικόνα 6.9 Case study C: Tags

Στην Εικόνα 6.10 μπορούμε να δούμε τον σχεδιασμό του αυτοματισμού σε γλώσσα Ladder Logic. Για την υλοποίηση σχεδιάστηκαν τέσσερα Networks.

Network 1: ελέγχουμε τον κινητήρα μας, άρα κατά συνέπεια ελέγχουμε την μετάδοση κίνησης στον ιμάντα μεταφοράς

Network 2: έχουμε τον έλεγχο της διαδικασίας γεμίσματος

Network 3: γίνεται έλεγχος της διαδικασίας εμφιάλωσης

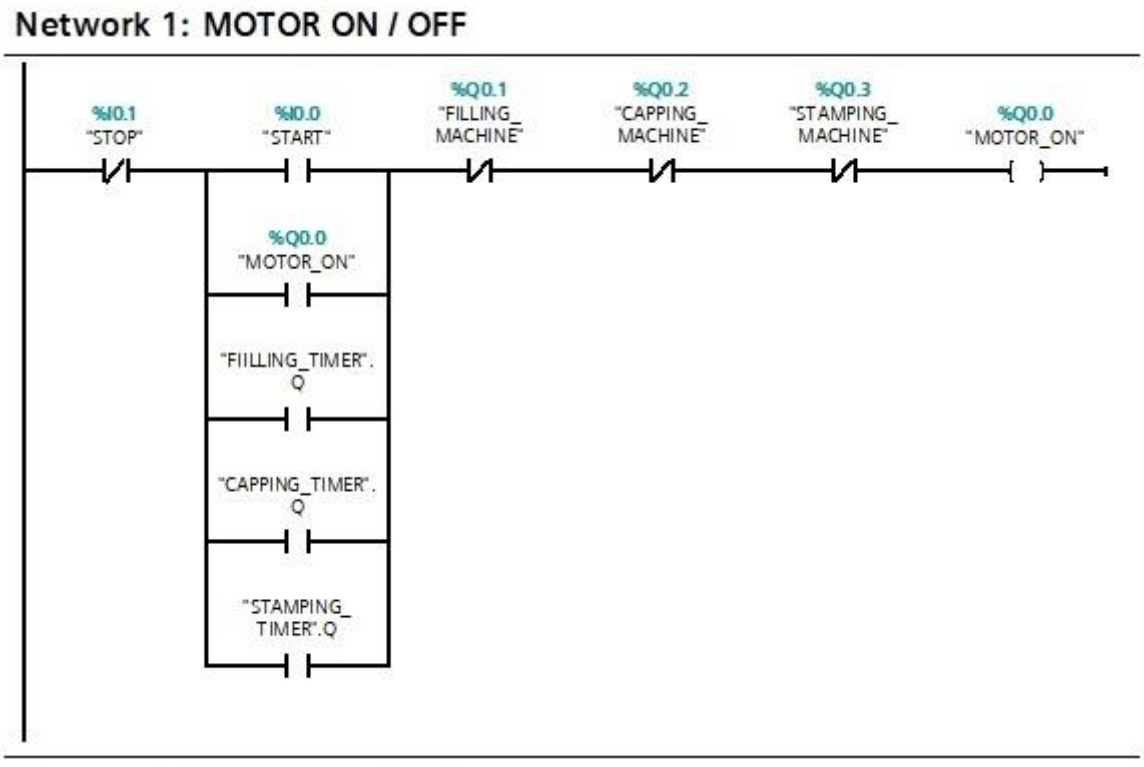
Network 4: ελέγχεται η διαδικασία τοποθέτησης του τυπώματος

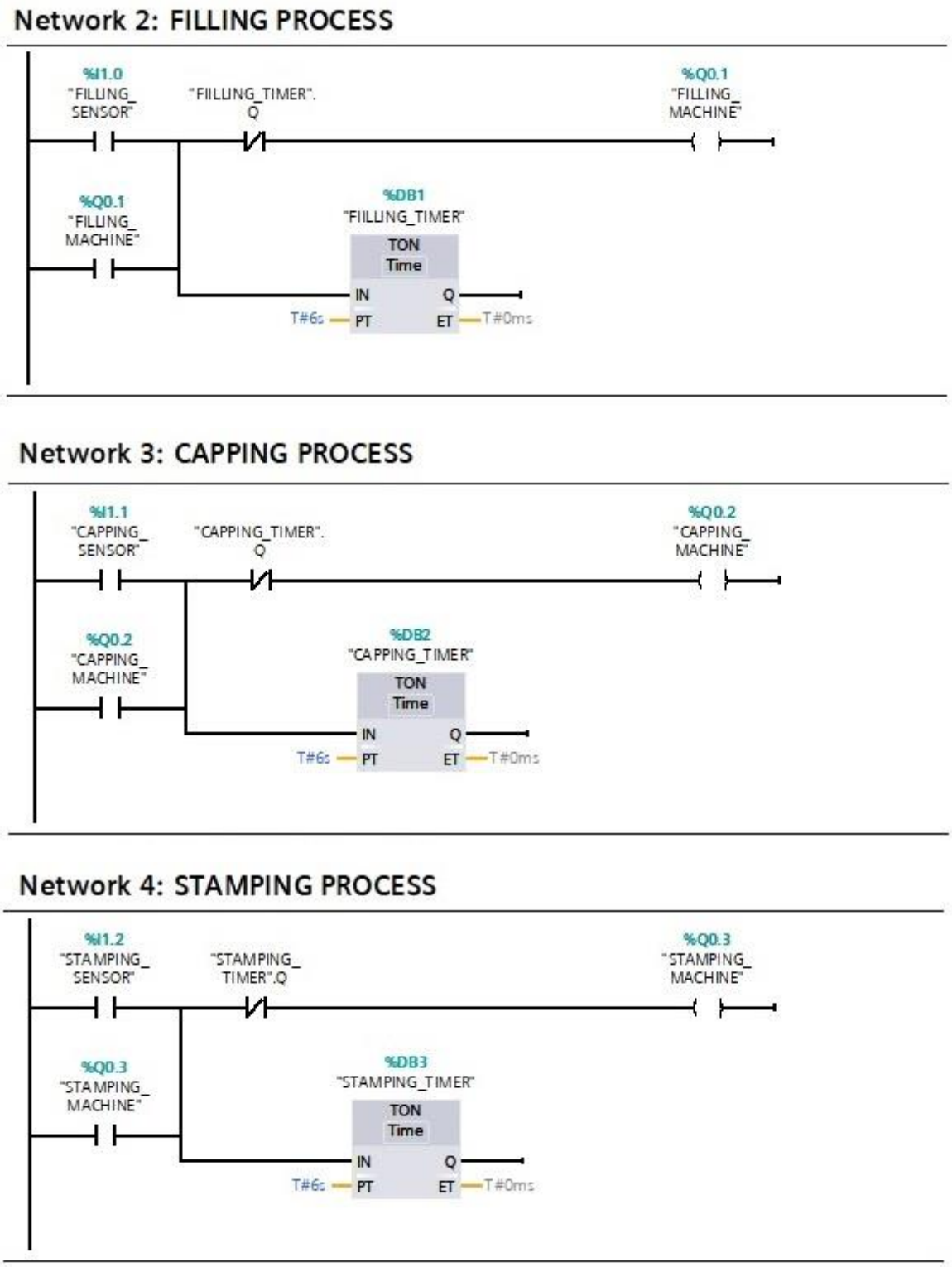
Η λογική προγραμματισμού έχει ως εξής:

- Στο Network 1 η διαδικασία ενεργοποιείται με το πάτημα του κουμπιού εκκίνησης START όπου και αυτό θα πάρει στιγμιαία την τιμή TRUE και θα γίνει εκκίνηση το Coil MOTOR_ON του κινητήρα.
- Ο κινητήρας θα παραμείνει ενεργοποιημένος με την χρήση του NO contact MOTOR_ON.
- Μπορούμε να τερματίσουμε την διαδικασία με το κουμπί STOP όπου το NC contact θα πάρει την τιμή TRUE άρα και θα τερματίσει την λειτουργία.
- Στο Network 2 βλέπουμε πως έχουμε χρησιμοποιήσει έναν timer τύπου TON (δηλώσαμε Data Block DB1 με όνομα FILLING_TIMER) ο οποίος ενεργοποιείται για χρονικό διάστημα 6 δευτερολέπτων (τόσο ορίσαμε την διαδικασία γεμίσματος).

Αυτό θα συμβεί εφόσον ο αισθητήρας γεμίσματος FILLING_SENSOR εντοπίσει μπουκάλι στην θέση γεμίσματος άρα και θα πάρει στιγμιαία την τιμή TRUE.

- Έτσι το Coil FILLING_MACHINE ενεργοποιείται, παραμένει ενεργοποιημένο με την χρήση του NO contact FILLING_MACHINE.
- Ταυτόχρονα στο Network 1 το NC contact FILLING_MACHINE έχει διακόψει την λειτουργία του κινητήρα (άρα και την μετακίνηση του μπουκαλιού το οποίο βρίσκεται στο στάδιο γεμίσματος).
- Η διαδικασία γεμίσματος θα συνεχιστεί για το ορισμένο στον TON Timer χρονικό διάστημα.
- Με το πέρας του χρόνου αυτού βλέπουμε πως στο Network 2 το NC Contact FILLING_TIMER.Q θα πάρει την τιμή TRUE άρα και θα διακοπεί η λειτουργία του FILLING_MACHINE, ενώ παράλληλα στο network 1 το NO Contact FILLING_TIMER.Q έχοντας την τιμή TRUE θα ενεργοποιήσει ξανά το Coil του κινητήρα MOTOR_ON άρα το μπουκάλι θα κατευθυνθεί προς το επόμενο στάδιο της διαδικασίας.
- Εξετάζοντας τα Network 3 και Network 4 παρατηρούμε πως έχουμε εργαστεί με παρόμοιο τρόπο για την εκτέλεση των λειτουργιών εμφιάλωσης και τοποθέτησης ετικέτας.





Εικόνα 6.10 Case study C: Ladder Logic

6.4 Case Study D: Προγραμματισμός Αυτοματισμού Εκκίνησης Κινητήρα

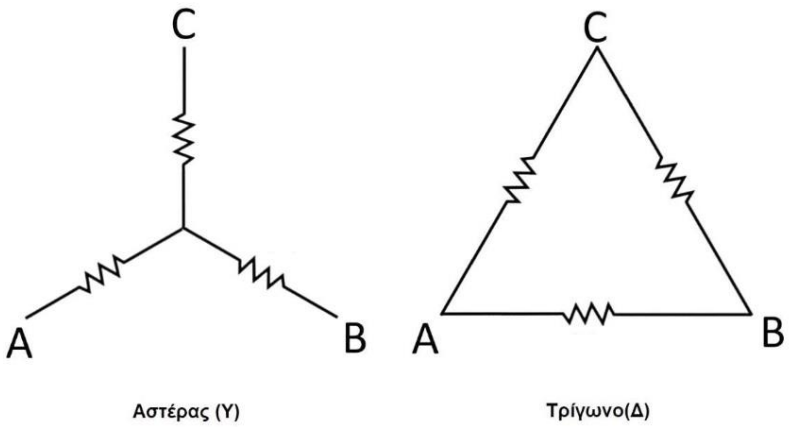
Σε προηγούμενα project που υλοποιήσαμε εστιάσαμε κατά κύριο λόγο στην λογική του προγραμματισμού ώστε να κατανοήσουμε καλύτερα τον τρόπο σκέψης και ανάλυσης ενός προβλήματος χωρίς να ασχοληθούμε ιδιαίτερα με τον εξοπλισμό που απαιτείται για το κάθε πρόβλημα. Όπως έχουμε αναφέρει σε προηγούμενο κεφάλαιο πολλές φορές το σήμα μας θα αποσταλεί σε κάποιο άλλο τμήμα hardware με τη βοήθεια του οποίου θα εκτελεστεί η επιθυμητή ενέργεια. Έτσι αντιλαμβανόμαστε πως ένα βιομηχανικό περιβάλλον είναι κάθε άλλο παρά ιδεατό. Στο παρόν case study θα θέλαμε να πραγματοποιήσουμε μια υλοποίηση λαμβάνοντας υπόψη περισσότερες λεπτομέρειες και εστιάζοντας σε πιο ρεαλιστικούς παράγοντες και περιβάλλον εργασίας.

Στο case study λοιπόν αυτό θα ασχοληθούμε με την διαδικασία εκκίνησης ενός τριφασικού κινητήρα ελέγχοντας όλα τα στάδια της εν λόγω διαδικασίας.

Προκειμένου να κάνουμε την υλοποίησή μας και να γνωρίζουμε τα σήματα που θα λάβει και θα εξάγει το PLC πρέπει να γνωρίζουμε ποια θα είναι η συνδεσμολογία του εξοπλισμού μας. Ας ανατρέξουμε αρχικά σε κάποιες γνώσεις ηλεκτρολογίας.

- Κατά την εκκίνηση ενός τριφασικού κινητήρα αναπτύσσεται στα πηνία του τάση αντίθετη της τάσης τροφοδοσίας, η λεγόμενη και ως αντιηλεκτρεγερτική δύναμη. Ως αποτέλεσμα αυτού η ένταση του ρεύματος εκκίνησης είναι πολλαπλάσια από εκείνη του ρεύματος κανονικής λειτουργίας. Υπάρχει λοιπόν η πιθανότητα να δημιουργηθούν προβλήματα στο δίκτυο μας, θέτοντας έτσι σε κίνδυνο την εγκατάστασή μας και τυχόν ευαίσθητο εξοπλισμό τον οποίο μπορεί να διαθέτουμε.
- Ένας τρόπος επίλυσης του προβλήματος αυτού είναι να εφαρμοστεί στον κινητήρα συνδεσμολογία Αστέρα-Τριγώνου (Υ-Δ). Η διαφορά μεταξύ των δύο συνδέσεων φαίνεται στην Εικόνα 6.11. Έτσι αρχικά κατά την εκκίνηση θα έχουμε

συνδεσμολογία Αστέρα (Υ) και μόλις φτάσουμε την επιθυμητή κατάσταση αλλάζουμε την συνδεσμολογία μας σε Τρίγωνο (Δ) [28].



Εικόνα 6.11 Διάγραμμα Αστέρα και Τριγώνου (Star and Delta diagrams)

Κατά την εκκίνηση ενός τριφασικού κινητήρα σε συνδεσμολογία Αστέρα είναι:

- $\boldsymbol{V_L}$: Τάση γραμμής
- $\boldsymbol{I_{LS}}$: Ρεύμα γραμμής
- $\boldsymbol{I_{PS}}$: Ρεύμα περιέλιξης ανά φάση
- $\boldsymbol{Z}$: Αντίσταση ανά περιέλιξη

και ισχύει ότι:

- το ρεύμα γραμμής ισούται με το ρεύμα περιέλιξης $\boldsymbol{I_{LS} = I_{PS}}$
- η τάση στις επιμέρους φάσεις της περιέλιξης είναι $\frac{V_L}{\sqrt{3}}$ άρα

$$I_{PS} = \frac{V_L}{\sqrt{3}\,Z} \Leftrightarrow I_{LS} = \frac{V_L}{\sqrt{3}\,Z}$$

Κατά την εκκίνηση ενός τριφασικού κινητήρα σε συνδεσμολογία Τριγώνου είναι:

- $\boldsymbol{I_{LD}}$: Ρεύμα γραμμής
- $\boldsymbol{I_{PDL}}$: Ρεύμα περιέλιξης ανά φάση
- $\boldsymbol{Z}$: Αντίσταση ανά περιέλιξη

και ισχύει ότι:

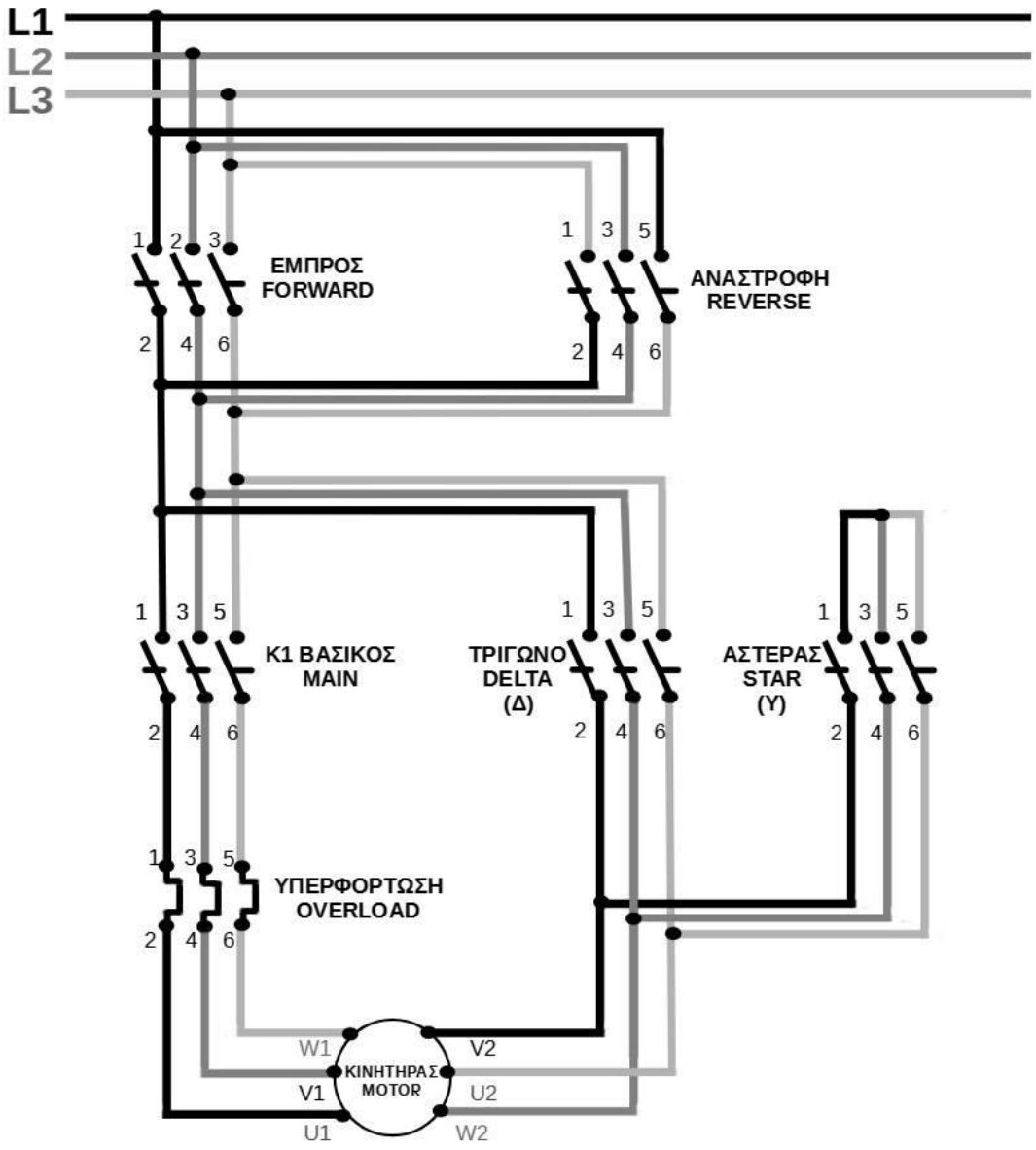
- το ρεύμα γραμμής ισούται με το ρεύμα περιέλιξης $\boldsymbol{I_{LD} = \sqrt{3} I_{PD}}$
- $I_{PD} = \frac{V_L}{Z}$, άρα
- $I_{LD} = \frac{\sqrt{3} V_L}{Z}$

Εξετάζοντας το λόγο των τιμών των εντάσεων του ρεύματος γραμμής τριγώνου και αστέρα είναι:

$$\frac{\boldsymbol{I_{LD}}}{\boldsymbol{I_{LS}}} = \frac{\frac{\boldsymbol{\sqrt{3} V_L}}{\boldsymbol{Z}}}{\frac{\boldsymbol{V_L}}{\boldsymbol{\sqrt{3} Z}}} = \boldsymbol{3} \Leftrightarrow \boldsymbol{I_{LS} = \frac{1}{3} L_D}$$

Κατά συνέπεια το ρεύμα εκκίνησης σε συνδεσμολογία Αστέρα είναι το ~33% του αντίστοιχου σε συνδεσμολογία Τριγώνου [29].

Αφού λοιπόν εξηγήσαμε τον λόγο επιλογής αυτής της συνδεσμολογίας στη συνέχεια θα δούμε μια διαγραμματική απεικόνιση στην Εικόνα 6.12 ώστε να κατανοήσουμε καλύτερα τα στοιχεία με τα οποία εργαζόμαστε.



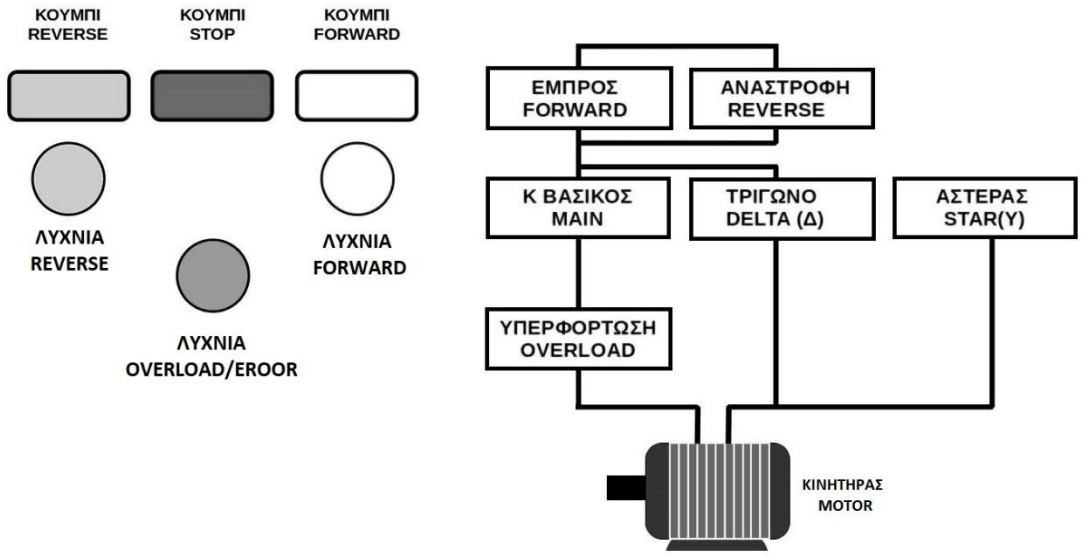
Εικόνα 6.12 Τριφασικός Κινητήρας σε συνδεσμολογία αστέρα-τρίγωνο με δυνατότητα Αντίστροφης κίνησης (Three-phase motor in star delta connection with Reverse ability)

Βλέπουμε λοιπόν πως τα βασικά στοιχεία της συνδεσμολογίας είναι τα εξής:

- Γραμμές L1, L2, L3 του δικτύου
- Ο τριφασικός κινητήρας
- Τρεις contractors (Βασικός Κ1, ένας για την λειτουργία σε Αστέρα, ένας για την λειτουργία σε Τρίγωνο)
  (πρόκειται για ρελέ βαρέως τύπου υψηλής ισχύος που χρησιμοποιείται για την τροφοδοσία του κινητήρα [29])

- Ένα Overload relay όπου στη περίπτωση υπερφόρτωσης του συστήματος θα ανοίξει για να προστατευτεί ο εξοπλισμός μας.
  (Η πλειονότητα των σφαλμάτων προκαλείται από υπερφόρτωση και λειτουργία με μη ισορροπημένη τροφοδοσία τάσης η οποία μπορεί να οδηγήσει σε υπερβολική θέρμανση και υποβάθμιση της μόνωσης του κινητήρα. Για την αποφυγή υπερφόρτωσης και βραχυκυκλώματος στην εσωτερική περιέλιξη απαιτείται προστασία την οποία παρέχει το Overload relay [29])
- Παρατηρούμε δύο επιπλέον contractors Forward και Reverse. Αλλάζοντας την πρώτη με την τρίτη φάση L1 και L3 μπορούμε να αντιστρέψουμε την κίνηση του κινητήρα μας. Έτσι προσθέσαμε και τη λειτουργία αυτή ώστε να έχουμε επιπλέον έλεγχο και δυνατότητες.

Στην Εικόνα 6.13 μπορούμε να δούμε μια σχηματική απεικόνιση του συστήματος με το οποίο καλούμαστε να εργαστούμε.



Εικόνα 6.13 Case study D: Απεικόνιση συστήματος (Case Study D: System display)

‘Εχουμε συνεπώς τα παρακάτω κύρια μέλη της εγκατάστασής μας:

- Κινητήρας
- Πέντε contactors των οποίων τη λειτουργία θα ελέγχουμε
- Το Overload
- Τρία κουμπιά ελέγχου για λειτουργίες Forward, Reverse και Stop
- Τρεις λυχνίες ένδειξης κατάστασης(Forward, Reverse και Error)

Η λογική λειτουργίας του συστήματος:

- Τα κουμπιά Forward και Reverse εκκινούν την αντίστοιχη διαδικασία και το κουμπί τερματισμού την σταματάει.
- Αρχικά ενεργοποιείται ο contactor Main παράλληλα με τον Contactor Star.
- Ο contactor Star θα μείνει ενεργός για ορισμένο χρονικό διάστημα (εξαρτάται από το κινητήρα) και στη συνέχεια απενεργοποιείται και ενεργοποιείται ο Contactor Delta.
- Τέλος όπως αναφέραμε το Overload ανοίγει το κύκλωμα και μας ειδοποιεί στην περίπτωση υπερφόρτωσης.

Στον Πίνακα 6.4 αναγράφονται οι απαραίτητες είσοδοι και έξοδοι για την υλοποίηση του εν λόγω προγράμματος.

Πίνακας 6.4: Πίνακας εισόδων/εξόδων Case Study D (Case Study D Input / Output table)

ΟΝΟΜΑ	ΕΙΔΟΣ	ΘΕΣΗ
FWD_BUTTON	ΕΙΣΟΔΟΣ	I0.0
REV_BUTTON	ΕΙΣΟΔΟΣ	I0.1
STOP_BUTTON	ΕΙΣΟΔΟΣ	I0.2
OL	ΕΙΣΟΔΟΣ	I0.3
FWD_LIGHT	ΕΞΟΔΟΣ	Q0.0
REV_LIGHT	ΕΞΟΔΟΣ	Q0.1
ERROR_LIGHT	ΕΞΟΔΟΣ	Q0.2

ON	ΕΞΟΔΟΣ	Q0.3
K_FWD	ΕΞΟΔΟΣ	Q1.0
K_REV	ΕΞΟΔΟΣ	Q1.1
K_MAIN	ΕΞΟΔΟΣ	Q1.2
K_STAR	ΕΞΟΔΟΣ	Q1.3
K_DELTA	ΕΞΟΔΟΣ	Q1.4

Στην Εικόνα 6.14 βλέπουμε και τις τιμές μας όπως έχουν δηλωθεί ως Tags στο πρόγραμμά μας.

PLC tags

	Name	Tag table	Data type	Address	Retain	Acces...	Writa...	Visibl...
1	FWD_BUTTON	Default tag table	Bool	%I0.0		☑	☑	☑
2	REV_BUTTON	Default tag table	Bool	%I0.1		☑	☑	☑
3	STOP_BUTTON	Default tag table	Bool	%I0.2		☑	☑	☑
4	OL	Default tag table	Bool	%I0.3		☑	☑	☑
5	FWD_LIGHT	Default tag table	Bool	%Q0.0		☑	☑	☑
6	REV_LIGHT	Default tag table	Bool	%Q0.1		☑	☑	☑
7	ERROR_LIGHT	Default tag table	Bool	%Q0.2		☑	☑	☑
8	ON	Default tag table	Bool	%Q0.3		☑	☑	☑
9	K_FWD	Default tag table	Bool	%Q1.0		☑	☑	☑
10	K_REV	Default tag table	Bool	%Q1.1		☑	☑	☑
11	K_MAIN	Default tag table	Bool	%Q1.2		☑	☑	☑
12	K_STAR	Default tag table	Bool	%Q1.3		☑	☑	☑
13	K_DELTA	Default tag table	Bool	%Q1.4		☑	☑	☑
14	<Add new>					☑	☑	☑

Εικόνα 6.14 Case study D: Tags

Στην Εικόνα 6.15 μπορούμε να δούμε τον σχεδιασμό του αυτοματισμού σε γλώσσα Ladder Logic. Για την υλοποίηση σχεδιάστηκαν εννέα Networks.

NETWORK 1: εκτελούμε έλεγχο της λειτουργίας εμπρόσθιας κίνησης (forward)

NETWORK 2: εκτελούμε έλεγχο της λειτουργίας αντίστροφης κίνησης (Reverse)

NETWORK 3: εκκίνηση της διαδικασίας

NETWORK 4: τερματισμός της διαδικασίας

NETWORK 5: έλεγχος για ειδοποίηση σφάλματος

NETWORK 6: έλεγχος λειτουργίας MAIN Contactor

NETWORK 7: έλεγχος λειτουργίας STAR Contactor

NETWORK 8: έλεγχος λειτουργίας χρονοδιακόπτη STAR

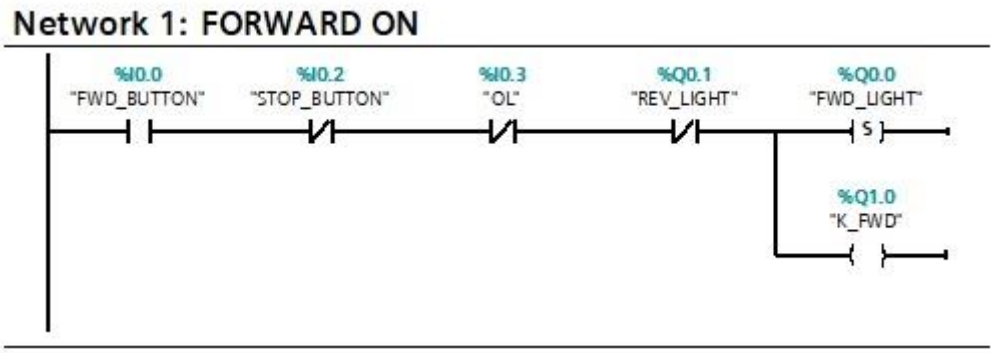
NETWORK 9: έλεγχος λειτουργίας DELTA Contactor

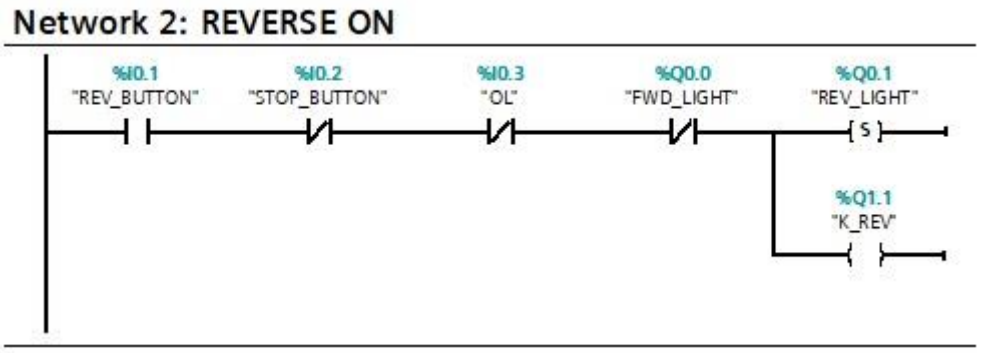
Η λογική προγραμματισμού έχει ως εξής:

- Στο Network 1 πατώντας το κουμπί λειτουργίας Forward ενεργοποιείται το NO Contact FWD_BUTTON και ενεργοποιεί τα Coils των εξόδων FWD_LIGHT (Set Coil για να παραμείνει αναμμένη η λυχνία) και του Contactor K_FWD.
- Τα NC contacts τα οποία έχουν τοποθετηθεί στο Network 1 μας επιβεβαιώνουν ότι δεν είναι ενεργή κάποια άλλη κατάσταση (αντίστροφη κίνηση, σφάλμα υπερφόρτωσης κ.τ.λ.).
- Το Network 2 έχει παρόμοια λειτουργία με το Network1 απλά εκτελεί τις ίδιες διαδικασίες για την εντολή αντίστροφης κίνησης Reverse.
- Στο Network 3 εφόσον δοθεί σήμα και ενεργοποιηθεί κάποιος από τους contactors K_FWD ή K_REV ενεργοποιείται το Set Coil ON με το οποίο θα ελέγξουμε στη συνέχεια την λειτουργία του κινητήρα.
- Στο Network 4 εφόσον δοθεί σήμα διακοπής της διαδικασίας ή υπερφόρτωσης του συστήματος (NO Contact STOP_BUTTON και OL) απενεργοποιούνται τα ON και οι λυχνίες ένδειξης μέσω των αντίστοιχων Reset Coils.
- Το Network 5 με χρήση ενός SR Flip-Flop ενεργοποιεί την λυχνία ERROR_LIGHT αν δοθεί σήμα υπερφόρτωσης. Στην περίπτωση αυτή η λυχνία παραμένει αναμμένη και εφόσον γίνουν όλοι οι απαραίτητοι έλεγχοι στον εξοπλισμό μας και επιλυθούν τα προβλήματα θα απενεργοποιηθεί την επόμενη φορά που θα γίνει εκκίνηση του κινητήρα μέσω του σήματος ON στο Reset.
- Στο Network 6 το SR Flip-Flop K_MAIN ενεργοποιείται με τιμή TRUE στο ON και απενεργοποιείται με διακοπή ή υπερφόρτωση (STOP_BUTTON ή OL).
- Με την ενεργοποίηση του K_MAIN δίνεται σήμα (εφαρμόζεται τάση) στον Βασικό Contactor του κυκλώματος, άρα και το πηνίο οπλίζει και συνδέει τις επαφές του.
- Με την ενεργοποίηση του K_MAIN έχουμε ταυτόχρονη ενεργοποίηση στο Network 7 του K_STAR (Contactor Αστέρα) μέσω του αντίστοιχου Flip-Flop. Η απενεργοποίησή του γίνεται με διακοπή ή Overload ή όταν περάσει ο το χρονικό

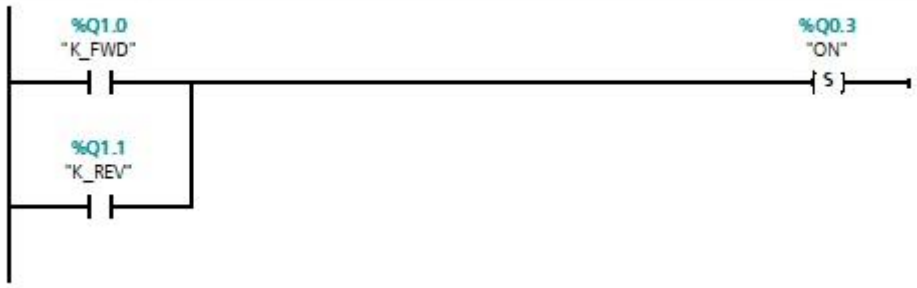
διάστημα που ορίζουμε και ενεργοποιηθεί ο αντίστοιχος Timer (NO Contact STAR_TIMER.Q).

- Τον Timer αυτό ορίζουμε στο network 8 με χρήση ενός TONR Timer ως το Data Block STAR_TIMER. Ενεργοποιείται με input το K_STAR έπειτα από το χρονικό διάστημα που ορίσουμε (στην περίπτωσή μας αυτό θα είναι τα 15 δευτερόλεπτα). Απενεργοποιείται με σήμα διακοπής ή υπερφόρτωσης.
- Τη στιγμή που θα ενεργοποιηθεί η έξοδος Q στον Timer θα γίνει Reset και θα απενεργοποιηθεί ο K_STAR ενώ παράλληλα όπως βλέπουμε στο Network 9 θα ενεργοποιηθεί ο K_DELTA (σήμα στον Contactor Τριγώνου) οπότε και θα ολοκληρωθεί η διαδικασία μας. Και πάλι με σήμα διακοπής ή υπερφόρτωσης έχουμε απενεργοποίηση του K_DELTA(διακόπτεται η τάση στο Τρίγωνο οπότε ανοίγει το κύκλωμα).
- Τέλος για να αποφύγουμε το ενδεχόμενο να ενεργοποιηθούν ταυτόχρονα ο Αστέρας και το Τρίγωνο, γεγονός που θα δημιουργήσει πρόβλημα στον εξοπλισμό μας, φροντίσαμε να τοποθετήσουμε τις NC Contact K_STAR στο Set του Flip-Flop K_DELTA (Network 9) και την αντίστοιχη NC Contact K_DELTA στο set του Flip-Flop K_STAR (Network 7).

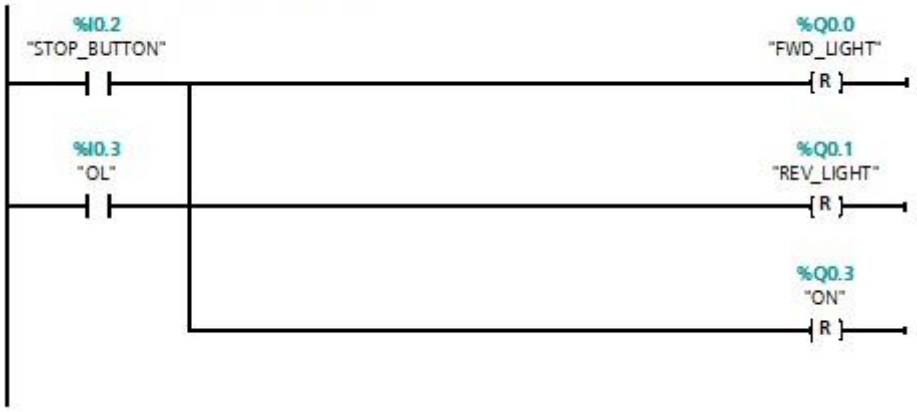




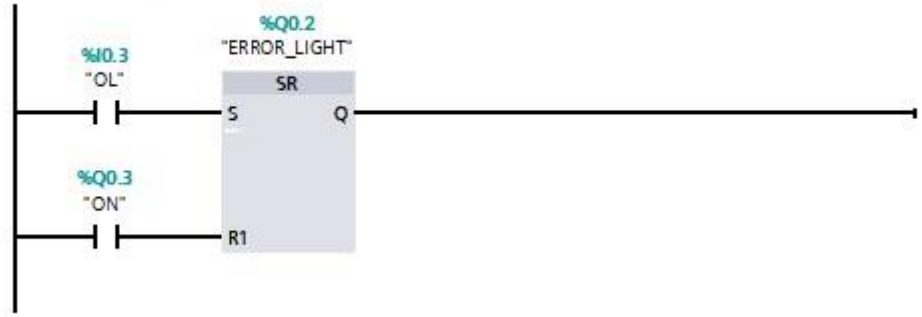
Network 3: START PROCESS



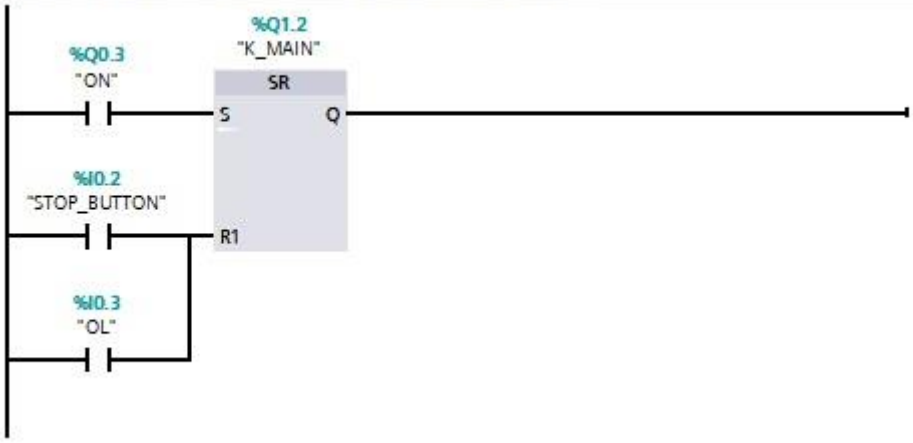
Network 4: STOP PROCESS



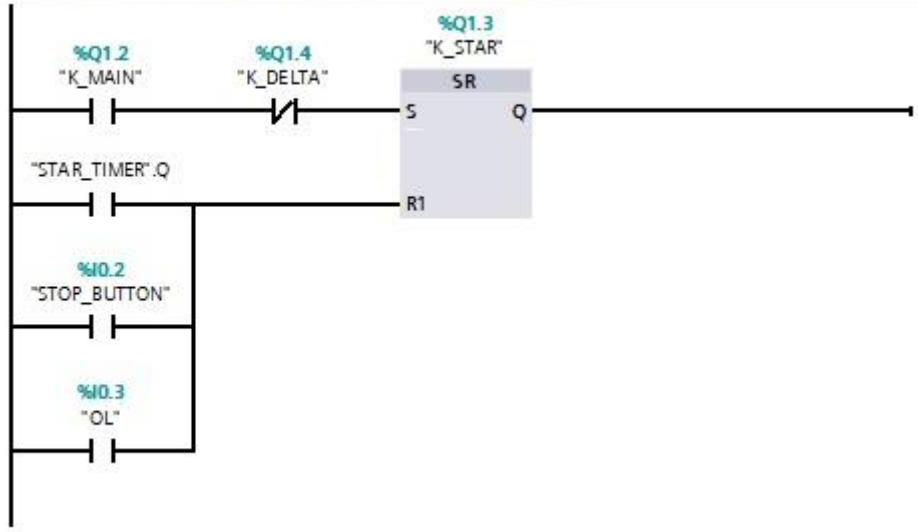
Network 5: ERROR INDICATION



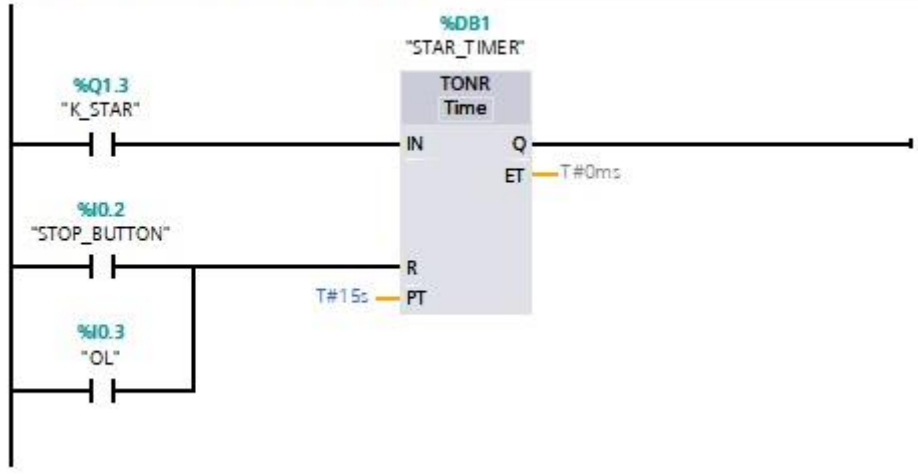
Network 6: MAIN CONTACTOR ON / OFF

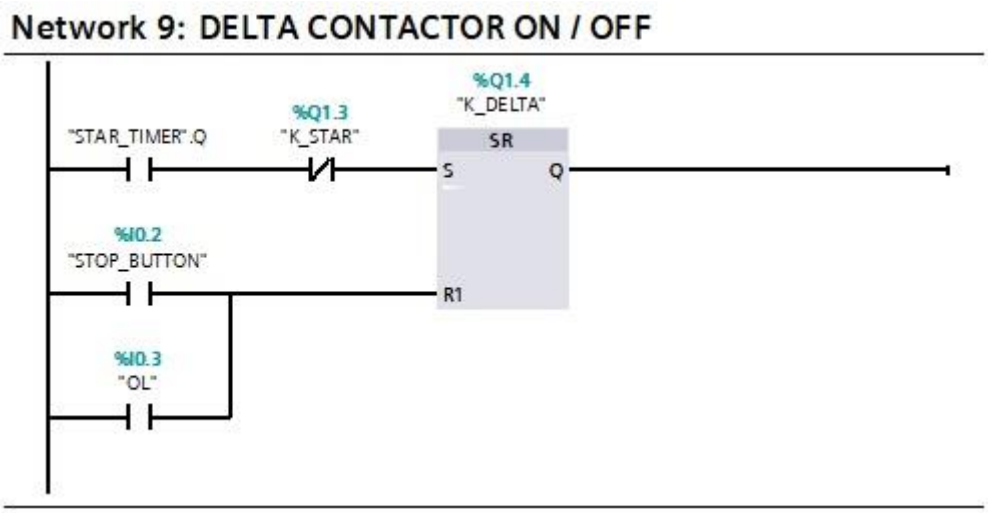


Network 7: STAR CONTACTOR ON / OFF



Network 8: STAR TIMER ON / OFF





Εικόνα 6.15 Case study D: Ladder Logic

Κεφάλαιο 7 Συμπεράσματα

7.1 Κατανόηση βιομηχανικών αυτοματισμών με PLC

Με το πέρας της παρούσας εργασίας είμαστε πλέον σε θέση να αξιολογήσουμε τις γνώσεις που αποκτήθηκαν. Πραγματοποιήσαμε μια εισαγωγή στους αυτοματισμούς και κατανοήσαμε την σημασία αυτών για χρήση στον τομέα της βιομηχανίας. Γνωρίσαμε την τεχνολογία των Προγραμματιζόμενων Λογικών Ελεγκτών και αναλύσαμε την αρχιτεκτονική τους καθώς και τον τρόπο λειτουργίας τους. Έπειτα, εστιάζοντας στη σειρά SIMATIC και τα PLCs της εταιρείας SIEMENS, γνωρίσαμε το λογισμικό ανάπτυξης αυτοματισμών TIA PORTAL καθώς και το λογισμικό προσωμοίωσης PLCSIM και μάθαμε να δημιουργούμε projects από το μηδέν, να επιλέγουμε το hardware και να κάνουμε δοκιμές στα προγράμματά μας. Παράλληλα γνωρίσαμε την γλώσσα προγραμματισμού Ladder Logic με όλες τις απαραίτητες λειτουργίες αυτής. Έτσι αξιοποιώντας και εφαρμόζοντας όλες αυτές τις γνώσεις είδαμε πως είμαστε πλέον σε θέση να προγραμματίζουμε ένα PLC, να δημιουργούμε προγράμματα, να κάνουμε προσομοίωση και να επεξεργαζόμαστε αλληλουχίες αυτοματισμών επιλύοντας έτσι στην πράξη προβλήματα που συναντώνται στη βιομηχανία. Μέσα από τα case studies που παρουσιάσαμε είδαμε βήμα προς βήμα τον προγραματισμό PLC, ενώ ταυτόχρονα μελετήσαμε τμηματικά τα προβλήματα που παρουσιάστηκαν και τον τρόπο σκέψης ώστε να επιλύσουμε αυτά, σχεδιάζοντας όλα τα απαραίτητα networks στα προγράμματά μας και αναλύοντας πλήρως τη λειτουργία καθενός από αυτά.

7.2 Συμπεράσματα της εργασίας

Συμπεραίνοντας, μέσα από την εργασία αυτή μελετήσαμε τους Προγραμματιζόμενους Λογικούς Ελεγκτές, τις χρήσεις τους αλλά και την συμβολή τους στον τομέα της βιομηχανίας. Πλεόν εχουμε πραγματοποιήσει μία ολοκληρωμένη εισαγωγή στον κόσμο των βιομηχανικών αυτοματισμών αφού ξεκινόντας από το μηδέν αναλύσαμε όλα τα

βασικά στοιχεία και φτάσαμε σε σημείο να δημιουργούμε προγράμματα και να δίνουμε άμασες λύσεις σε υπαρκτά σενάρια και προβλήματα.

Προφανώς κατανοούμε πλέον πως η τεχνολογία αυτή αποτελεί το θεμέλιο του σύγχρονου βιομηχανικού αυτοματισμού χάρη στην αξιοπιστία και αποτελεσματικότητα που παρέχει. Έτσι βλέπουμε πως η τεχνολογία των PLC έχει ένα λαμπρό μέλλον μπροστά της αφού θα συνεχίσει να αναπτύσσεται και σε συνεργασία με άλλες τεχνολογίες θα διευρύνει το φάσμα των χρήσεών της καθιστώντας την έτσι αναπόσπαστο τμήμα της βιομηχανίας μας. Καθώς λοιπόν εισερχόμαστε σε μια εποχή ολοένα και πιο περίπλοκων βιομηχανικών αυτοματισμών τα PLC θα συνεχίσουν να παίζουν κομβικής σημασίας ρόλο στην υλοποίηση των διαδικασιών. Προβάλλει έτσι επιτακτική η ανάγκη διάθεσης των πόρων στην περαιτέρω έρευνα και ανάπτυξη της τεχνολογίας αυτής με γνώμονα την καινοτομία και εξερεύνηση περισσότερων δυνατοτήτων που ενδεχομένως μπορεί να μας προσφέρει.

7.3 Ιδεές για μελλοντική έρευνα

Ασφαλώς σε ότι έχει να κάνει με τον τομέα του βιομηχανικού αυτοματισμού υπάρχει μία πληθώρα τεχνολογιών που αξίζει κάποιος να μελετήσει.

Μία πρόταση λοιπόν για διερεύνηση των παραπάνω γνώσεων θα ήταν η μελέτη και η υλοποίηση συστημάτων Human Machine Interface (HMI) καθώς και συστημάτων Supervisory Control And Data Acquisition (SCADA). Έτσι θα μπορούσαμε να υλοποιήσουμε περισσότερο ολοκληρωμένα συστήματα για τους αυτοματισμούς μας, αξιοποιώντας τις τεχνολογίες αυτές με τέτοιο τρόπο ώστε να πραγματοποιείται και να εφαρμόζεται έλεγχος των αυτοματισμών από ανθρώπινο παράγοντα, ακόμη και απομακρυσμένα, με αποτέλεσμα τον καλύτερο συντονισμό των διαδικασιών αυτοματισμού.

Επιπρόσθετα, αξίζει να αναφερθεί πως ένα επόμενο στάδιο θα μπορούσε να είναι και η διαχείριση και εξαγωγή δεδομένων από το PLC σε συνεργασία με την υλοποίηση εφαρμογών με σκοπό την ανάλυση δεδομένων.

Τέλος να αναφέρουμε πως μία καλή κατεύθυνση έρευνας για το μέλλον των Προγραμματιζόμενων Λογικών Ελεγκτών είναι εκείνη που θα συνδυάσει την τεχνολογία αυτή με ραγδαία άνοδο της τεχνητής νοημοσύνης. Έτσι, αρκετά ενδιαφέρον είναι το

γεγονός πως μπορούμε να αναπτύξουμε τρόπους με τους οποίους θα συνδυάσουμε τις τεχνολογίες αυτές ώστε να δημιουργήσουμε πιο έξυπνα και αποδοτικά συστήματα.

Βιβλιογραφία

[1] Chanchal Dey, Sunit Kumar. Industrial Automation Technologies, 1st Edition, 2020

[2] Rodson Henrique Hatahara da Fonseca, Fabiana Rocha Pinto. The Importance of the Programmable Logic Controller “PLC” in the Industry in the Automation Process. International Research Journal of Engineering and Technology (IRJET) Volume: 06 Issue: 11, 2019

[3] A. K. Gupta, S. K. Arora. Industrial Automation and Robotics, 2011

[4] Automation. Available at: https://en.wikipedia.org/wiki/Automation

[5] Relay. Available at: https://en.wikipedia.org/wiki/Relay

[6] William Bolton. Programmable Logic Controllers, 6th Edition, 2015

[7] Σ.Γ. ΜΟΥΡΟΥΤΣΟΣ, Γ. ΜΑΛΙΑΡΗΣ (2016). Τεχνικό Σχέδιο Μηχανολογικό, Ηλεκτρολογικό, Σχέδιο βιομηχανικών αυτοματισμών, 2η Έκδοση, Εκδόσεις Τσότρας, 2016

[8] Dilip Patel. Introduction Practical PLC (Programmable Logic Controller) Programming, 2018

[9] Ephrem Ryan Alphonsus, Mohammad Omar Abdullah. A review on the applications of programmable logic controllers (PLCs).Renewable and Sustainable Energy Reviews, https://doi.org/10.1016/j.rser.2016.01.025

[10] Olushola Akande.Industrial Automation from Scratch: A hands-on guide to using sensors, actuators, PLCs, HMIs, and SCADA to automate industrial processes, Packt Publishing, 2023

[11] PLC Internal Architecture and Diagram Explanation. Available at: https://pcinformation.net/plc-internal-architecture-and-diagram-explanation/

[12] IEC 61131-3. Available at: https://en.wikipedia.org/wiki/IEC_61131-3

[13] Farouk Idris, Michael Blake. Start Programming & Simulating PLC in Your Laptop from Scratch: A No BS, No Fluff, PLC Programming. A Three Parts Series in One Book. The Practical Approach to Coding PLC from Beginning without a Real PLC with Real World Examples, 2020

[14] Function block diagram. Available at: https://en.wikipedia.org/wiki/Function_block_diagram

[15] Instruction list. Available at: https://en.wikipedia.org/wiki/Instruction_list

[16] Wiztech Automation Solutions. 10 Advantages of PLC Systems in Industrial Automation. Available at: https://www.linkedin.com/pulse/10-advantages-plc-systems-industrial-automation-solutions, 2023

[17] Gaurav Aggarwal. Top 10 Advantages of PLC That You Must Know, 2023. Available at: https://trainings.internshala.com/blog/advantages-and-disadvantages-of-plc/

[18] Aadhya Kargeti. Programmable Logic Controller (PLC) Market Share 2023-2028 | Industry Report, 2023. Available at: https://www.linkedin.com/pulse/programmable-logic-controller-plc-market-share-industry-kargeti

[19] PLC MANUFACTURERS. Available at: https://www.automationreadypanels.com/plc-systems/plc-manufacturers/

[20] Statista Research Department. Global PLC market share as of 2017, by manufacturer, 2023. Available at: https://www.statista.com/statistics/897201/global-plc-market-share-by-manufacturer/

[21] Siemens PLC System Overview. Available at: https://automationforum.co/siemens-plc-system-overview/

[22] Simatic. Available at: https://en.wikipedia.org/wiki/Simatic

[23] Siemens PLCs. Available at: https://automationprimer.com/2014/03/16/siemens-plcs/

[24] Siemens PLC. Available at: https://www.plctable.com/siemens-plc/

[25] SEIMENS Comparison list for S7-300, S7-400, S7-1200, S7-1500 Reference Manual. Available at: https://cache.industry.siemens.com/dl/files/648/109797648/att_1067421/v1/s71500_compare_table_en.pdf

[26] SIMENS SIMATIC S7 S7-1200 Programmable controller System Manual. Available at:

https://cache.industry.siemens.com/dl/files/465/36932465/att_106119/v1/s71200_system_manual_en-US_en-US.pdf

[27] Ladder logic. Available at: https://en.wikipedia.org/wiki/Ladder_logic

[28] Εκκινητής αστέρα-τριγώνου Γιατί χρησιμοποιείται ο εκκινητής αστέρα-τριγώνου (Y-D). Available at: https://www.ti-soft.com/el/support/help/electricaldesign/project/distribution/circuits/motor-load/motor-starters/star-delta-starter

[29] STAR – DELTA Starter. Available at: https://forumelectrical.com/star-delta-starter/