ΤΕΧΝΟΛΟΓΙΚΟ
ΕΚΠΑΙΔΕΥΤΙΚΟ
ΙΔΡΥΜΑ ΠΕΙΡΑΙΑ

ΤΜΗΜΑ ΑΥΤΟΜΑΤΙΣΜΟΥ

ΜΕΛΕΤΗ ΣΥΧΡΟΝΩΝ ΣΥΣΤΗΜΑΤΩΝ ΑΥΤΟΜΑΤΙΣΜΟΥ ΣΤΗ ΒΙΟΜΗΧΑΝΙΑ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΛΑΔΑΚΑΚΟΥ ΘΑΛΗ

Α.Μ. 30388

ΕΠΙΒΛΕΠΟΝΤΕΣ ΚΑΘΗΓΗΤΕΣ :

Κ. ΑΛΑΦΟΔΗΜΟΣ ΚΩΝΣΤΑΝΤΙΝΟΣ – Κ. ΝΙΚΟΛΑΟΥ ΓΡΗΓΟΡΙΟΣ

2012

Εισαγωγή

Στην παρούσα πτυχιακή εργασία θα αναπτυχθεί το θέμα ¨ Μελέτη Σύγχρονων Συστημάτων Αυτοματισμού στη Βιομηχανία¨. Στο πρώτο κεφάλαιο γίνεται μια περιληπτική παρουσίαση μερικών βασικών εννοιών . Στο δεύτερο κεφάλαιο παρουσιάζονται τα συστήματα SCADA. Στο τρίτο κεφάλαιο γίνεται λόγος για τους Προγραμματισμένους Λογικούς Ελεγκτές PLC. Έπειτα στο τέταρτο κεφάλαιο αναπτύσσεται το θέμα του «Πρωτοκόλλου CANBUS». Στην συνεχεία, στο τελευταίο κεφάλαιο αναπτύσσεται και περιγράφεται μια εφαρμογή καθώς και ο βασικός εξοπλισμός που απαιτείται για την υλοποίηση της .

ΠΕΡΙΕΧΟΜΕΝΑ

4°Κεφαλαιο: Πρωτόκολλο CANBUS

5°Κεφαλαιο: Εφαρμογές

1ο ΚΕΦΑΛΑΙΟ
ΒΑΣΙΚΕΣ ΕΝΝΟΙΕΣ

1.1 **Βιομηχανία**

Ως βιομηχανία ορίζεται η κατασκευή ενός αγαθού ή υπηρεσίας μέσα σε μία οικονομία. Παρόλο που η βιομηχανία αποτελεί μία ευρεία έννοια για κάθε είδος οικονομικής παραγωγής, sta οικονομικά και τον αστικό σχεδιασμό η βιομηχανία είναι συνώνυμη της δευτερογενούς παραγωγής. Λέγοντας δευτερογενή παραγωγή εννοούμε το είδος της οικονομικής δραστηριότητας που ασχολείται με την κατασκευή αγαθών και προϊόντων .

Υπάρχουν τέσσερις κύριοι βιομηχανικοίοικονομικοί τομείς, ο *πρωτογενής τομέας* που αφορά κυρίως τη βιομηχανία εξαγωγής πρώτων υλών όπως η εξόρυξη και η γεωργία, ο *δευτερογενής τομέας* που αφορά την διύλιση, την κατασκευή και την παραγωγή, ο *τριτογενής τομέας* ασχολείται με υπηρεσίες (όπως η νομική ή η ιατρική) και τη διανομή των αγαθών ,τέλος υπάρχει ο *τεταρτογενής τομέας* ο οποίος είναι σχετικά νέος τύπος της γνωστικής βιομηχανίας που εστιάζει στην τεχνολογική έρευνα, σχεδιασμό και εξέλιξη, όπως ο προγραμματισμός ή η βιοχημεία. Επίσης έχει προταθεί και ένας πεμπτογενής τομέας που θα περιλαμβάνει τις μη κερδοσκοπικές δραστηριότητες. Η οικονομία χωρίζεται επίσης δύο τομείς, δημόσιο και ιδιωτικό, με τη βιομηχανία να κατατάσσεται συνήθως στον ιδιωτικό.

Η βιομηχανία με την έννοια της παρασκευής-κατασκευής έγινε ο κύριος τομέας παραγωγής και εργασίας στις Ευρωπαϊκές και Βορειοαμερικανικές χώρες κατά τη διάρκεια της Βιομηχανικής Επανάστασης, με την ανατροπή των προηγούμενων εμπορικών και φεουδαρχικών οικονομιών, μέσω της ταχείας και επιτυχούς τεχνολογικής ανάπτυξης, όπως επήλθε για παράδειγμα με την εισαγωγή του χάλυβα και του γαιάνθρακα στην παραγωγή. Βοηθήθηκε από τις τεχνολογικές εξελίξεις και συνέχισε να αναπτύσσεται σε νέους τύπους και τομείς μέχρι και σήμερα. Ως επακόλουθο, οι βιομηχανικές χώρες άρχισαν να υιοθετούν καπιταλιστική οικονομική πολιτική. Σιδηρόδρομοι και ατμόπλοια άρχισαν γρήγορα να συνδέουν παγκόσμιες αγορές οι οποίες ήταν προηγούμενα απρόσιτες, επιτρέποντας σε ιδιωτικές εταιρίες να εξελιχθούν σε ανήκουστα για την τότε εποχή μεγέθη και πλούτο. Από την Βιομηχανική επανάσταση και έπειτα , το ένα τρίτο περίπου της παγκόσμιας οικονομικής παραγωγής προέρχεται από τη βιομηχανική παραγωγή, ποσοστό μεγαλύτερο από το αντίστοιχο ποσοστό της αγροτικής παραγωγής.

Πολλές ανεπτυγμένες χώρες, όπως το Ηνωμένο Βασίλειο, οι ΗΠΑ και ο Καναδάς, και αρκετές αναπτυσσόμενες χώρες, όπως η Κίνα και η Ινδία, εξαρτώνται σημαντικά από τη βιομηχανία. Οι βιομηχανίες, οι χώρες στις οποίες βρίσκονται και δραστηριοποιούνται και οι οικονομίες των χωρών αυτών διασυνδέονται με ένα πολύπλοκο δίκτυο αλληλεξάρτησης.

1.2 **Αυτοματισμός**

Ο **αυτοματισμός** (ή *αυτοματοποίηση*) αφορά δύο έννοιες σχετιζόμενες μεταξύ τους. Αρχικά, σημαίνει την τυποποίηση μίας διαδικασίας μέσω της εύρεσης καλώς ορισμένων βημάτων τα οποία πρέπει να ακολουθηθούν για να παραχθεί κάποιο επιθυμητό αποτέλεσμα. Έτσι ο αυτοματισμός δεν είναι τίποτα άλλο παρά η εύρεση ενός αλγορίθμου για την επίλυση ενός προβλήματος, ή η κατασκευή ενός αυτόνομου μηχανισμού που εκτελεί αυτόν τον αλγόριθμο για κάποια είσοδο χωρίς ανθρώπινη παρέμβαση.

Ο αυτοματισμός επανοηματοδοτήθηκε μέσα από τη μηχανολογία και την ηλεκτρολογία κατά τον 20ο αιώνα, ως ένα πεδίο της επιστήμης μηχανικού ασχολούμενο με τον *έλεγχο διεργασιών* και τη διατήρησή τους σε καθορισμένη κατάσταση. Παραδείγματος χάριν ο αυτοματισμός στοχεύει στη διατήρηση σε σταθερά επίπεδα της θερμοκρασίας ενός θερμοστάτη, της πορείας ενός αεροπλάνου, της ταχύτητας ενός αυτοκινήτου κλπ.

Ο αυτοματισμός στηρίζεται εννοιολογικά στη θεωρία ελέγχου και στους μηχανισμούς ανάδρασης, επομένως αποτέλεσε μία κύρια αφετηρία της κυβερνητικής. Σε αντίθεση με την τελευταία όμως, ο αυτοματισμός έχει έναν αυστηρά εφαρμοσμένο χαρακτήρα και στην πράξη αξιοποιεί ποικιλία εξειδικευμένων προϊόντων ηλεκτρονικής και τεχνολογίας πληροφοριών (π.χ. μικροελεγκτές, συστήματα πραγματικού χρόνου, τεχνολογίες CAx). Η σημασία του αυτοματισμού είναι μεγάλη στη βιομηχανία, όπου μειώνει σημαντικά την ανάγκη για ανθρώπινη παρέμβαση (π.χ. σε τηλεμετρίες, αυτόματο έλεγχο γραμμών παραγωγής κλπ).

Εξειδικευμένοι υπολογιστές υψηλής αντοχής, οι προγραμματιζόμενοι λογικοί ελεγκτές (PLC), χρησιμοποιούνται για να συγχρονίσουν τη ροή εισόδων από φυσικούς αισθητήρες με τη ροή εντολών προς συσκευές εξόδου (π.χ. βραχίονες). Η αναδραστική και ντετερμινιστική λειτουργία του συστήματος οδηγεί σε αυστηρά ελεγχόμενες διεργασίες, κατάλληλες για χρήση σε βιομηχανικές μονάδες. Η εργασία του αυτοματιστή στο προηγούμενο παράδειγμα συνίσταται στον κατάλληλο προγραμματισμό του PLC (συνήθως σε συμβολική γλώσσα).

Τι κάνει ένας αυτοματισμός

Ο αυτοματισμός ερευνά τη συμπεριφορά δυναμικών συστημάτων μοντελοποιώντας τα μέσω των μεθοδολογικών και μαθηματικών εργαλείων της επεξεργασίας σήματος. Έτσι μεταχειρίζεται τα συστήματα ως μαύρα κουτιά με είσοδο και έξοδο. Ως είσοδος θεωρείται ένα σήμα, αναλογικό ή ψηφιακό, συλλεγόμενο από κάποιο σημείο του συστήματος. Τα ενδιάμεσα κουτιά αναπαριστούν τις διάφορες διαταράξεις που επηρεάζουν το σήμα, όπως τριβές στους ενεργοποιητές, αποτέλεσμα των στοιχείων του ελέγχου που παρεμβάλλονται, τους ελεγκτές.

Αυτά τα αποτελέσματα συνήθως αναπαρίστανται με μαθηματικές συναρτήσεις, τις *συναρτήσεις μεταφοράς*. Μία **συνάρτηση μεταφοράς** προσδιορίζει ένα σύστημα και τον τρόπο που μεταβάλλει κάθε σήμα εισόδου. Η έξοδος του συστήματος ονομάζεται *αναφορά* και ανταποκρίνεται στην τιμή του σήματος κατόπιν ενεργοποίησης των προηγούμενων συναρτήσεων μεταφορών σε αυτήν. Όταν μια ή περισσότερες μεταβλητές εξόδου ενός συστήματος πρέπει να ακολουθήσουν την τιμή κάποιας αναφοράς που μεταβάλλεται με τον χρόνο, χρειάζεται να προστεθεί ένας ελεγκτής που να χειρίζεται τις τιμές των σημάτων εισόδου έως ότου επιτευχθεί η επιθυμητή έξοδος.

2ο ΚΕΦΑΛΑΙΟ

SCADA

2.1 Γενικά

Ξεκινώντας την περιγραφή των συστημάτων SCADA είναι σκόπιμο να αναφερθεί τι είναι το SCADA. Η λέξη SCADA αποτελεί τα αρχικά των λέξεων Supervisory Control And Data Acquisition System, δηλαδή σύστημα εποπτείας, ελέγχου και συλλογής πληροφοριών. Είναι συνεπώς συστήματα τηλεμετρίας και τηλεχειρισμού, τα οποία συλλέγουν πληροφορίες από διάφορες διεργασίες και χρησιμοποιούνται για τον εποπτικό τους έλεγχο. Αρχικά υλοποιήθηκαν στο λειτουργικό σύστημα DOS(VMS) και το Unix αλλά τα τελευταία χρόνια
όλοι οι προμηθευτές SCADA έχουν κινηθεί προς τα Windows και μερικά επίσης προς το λειτουργικό Linux.
Τα SCADA βρίσκουν τεράστιες εφαρμογές, τόσο σε βιομηχανικές μονάδες όσο και σε συστήματα μεταφορές και διανομής ηλεκτρικής ενέργειας. Αυτό γιατί επιτρέπουν την διαχείριση και την εποπτεία ενός συστήματος που μπορεί να βρίσκεται αρκετές εκατοντάδες χιλιόμετρα μακριά από τον χώρο ελέγχου, όπως συνήθως συμβαίνει με τα συστήματα ηλεκτρικής ενέργειας, αλλά και τις μεγάλες βιομηχανικές μονάδες. Το μέγεθος τέτοιας σειράς εγκαταστάσεων εκτείνεται από 1000 μέχρι 10 χιλιάδες κανάλια εισόδου-εξόδου (I/O).
Παρά όλα αυτά δεν συναντάμε συστήματα SCADA σε μικρομεσαίες βιομηχανικές μονάδες, λόγω του μεγάλου κόστους της υλικοτεχνικής υποδομής (hardware) αλλά και του λογισμικού (software). Ένα σύστημα SCADA είναι υπεύθυνο για την διαχείριση και τον έλεγχο διαφόρων διεργασιών, δηλαδή είναι υπεύθυνο για την παρακολούθηση, την καταγραφή και τον έλεγχο ενός πλήθους βασικών μεταβλητών και παραμέτρων του συστήματος.
Αυτά τα συστήματα καλύπτουν τη μεταφορά των δεδομένων μεταξύ ενός κεντρικού οικοδεσπότη υπολογιστή SCADA (central host PC) και διάφορες μονάδες απομακρυσμένων τερματικών (RTUs) και προγραμματιζόμενων λογικών ελεγκτών (PLCs), και τα τερματικά χειριστών. Παραδοσιακά, τα συστήματα SCADA έχουν χρησιμοποιήσει στο δημόσιο δίκτυο μεταβίβασης (PSN) για λόγους ελέγχου. Σήμερα πολλά συστήματα ελέγχονται χρησιμοποιώντας την υποδομή του εταιρικού δικτύου LocalAreaNetwork (LAN)/WideAreaNetwork (WAN). Οι ασύρματες τεχνολογίες τώρα πλέον ευρέως επεκτείνονται και χρησιμοποιούνται για λόγους του ελέγχου.

Οι στόχοι του SCADA αναφέρονται ευθύς αμέσως :

- Η διασφάλιση της ποιότητας του παραγόμενου προϊόντος.
- Η μεγιστοποίηση της παραγωγής με χρήση των ελάχιστων δυνατών (ενεργειακών) πόρων.
- Η βέλτιστη διαχείριση του εξοπλισμού, των υλικών και της ενέργειας της εγκατάστασης.
- Η ασφάλεια του εξοπλισμού και του προσωπικού παρακολούθησης της διεργασίας.

Είναι σημαντικό όλοι οι παραπάνω στόχοι να επιτυγχάνονται παράλληλα. Δεν είναι δυνατό να επιδιώκουμε μεγιστοποίηση της παραγωγής χωρίς πρώτα να έχουμε εξασφαλίσει την βελτιστοποίηση διαχείρισης των διαθέσιμων πόρων και υλικών. Επιπλέον ο έλεγχος των διεργασιών θα πρέπει να είναι εξαιρετικά γρήγορος, ώστε να έχουμε επίγνωση της κατάστασηςτων επιτηρούμενων μεγεθών σε πραγματικά χρόνο (real time). Αυτό γίνεται αμέσως εμφανές αν αναφερθούμε στο θέμα της ασφάλειας. Η επέμβαση στις διεργασίες σε περίπτωση κινδύνου θα πρέπει να είναι άμεση και αποτελεσματική ώστε να αποφεύγουμε μερική ή ολικήκαταστροφή του εξοπλισμού, ακόμα και ανθρώπινες απώλειες. Έκτος των παραπάνω, ένα σύστημα SCADA μπορεί να επιτηρεί και να χειρίζεται ένα πλήθος μεταβλητών του συστήματος αυτομάτου ελέγχου, αλλά και να διαχειρίζεται και οικονομικά μεγέθη (παραγγελίες - παραδόσεις προϊόντων) σε συνεργασία με οικονομικά πακέτα, προκειμένου να παρέχει στον χειρίστη **του** συνολική εποπτεία της παραγωγικής μονάδας.

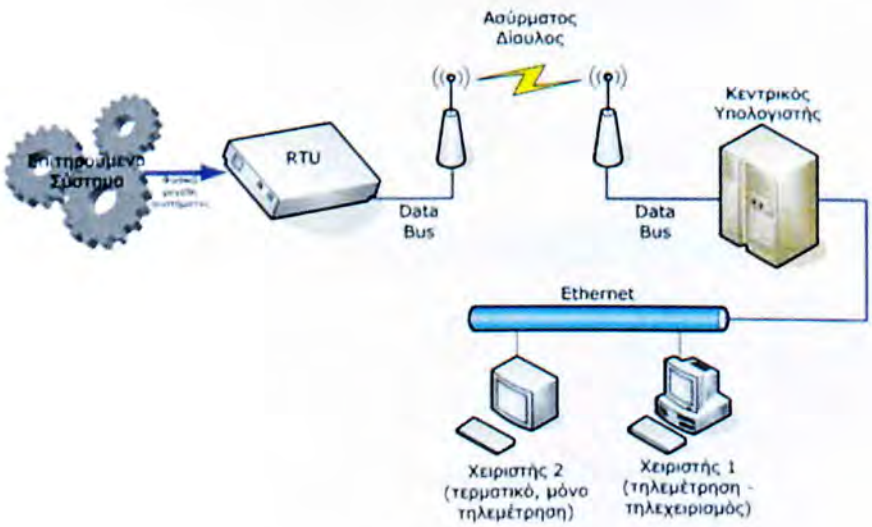
Πιο συγκεκριμένα, ένα σύστημα SCADA προσφέρει :

- Άμεση πληροφόρηση της κατάστασης της διεργασίας
- Αντιστάθμιση των μεταβλητών ελέγχου της διεργασίας με στόχο τη διατήρηση των δεδομένων ονομαστικών τους τιμών (setpoints) καθώς και τη διατήρηση των απαιτούμενων επιπέδων παραγωγής
- Έγκαιρη σήμανση των βλαβών και της κακής λειτουργίας του εξοπλισμού στις διάφορες διεργασίες, ώστε να παρέχεται η μέγιστη ασφάλεια του εξοπλισμού και των εργαζομένων
- Πρόγνωση και διάγνωση των βλαβών του εξοπλισμού και έγκαιρο εντοπισμό τους για την μεγιστοποίηση της διαθεσιμότητας του
- Καταγραφή και αποθήκευση πληροφοριών σχετικά με την παραγωγή και τη διαχείριση της .
- Καλή λειτουργία του εξοπλισμού με στόχο τη βελτιστοποίηση της χρήσης και επομένως της παραγωγικότητάς του.

2.2 Αρχή Λειτουργιάς του SCADA

Όπως έχουμε ήδη αναφέρει ένα σύστημα SCADA αποτελεί ένα πλήρες σύστημα τηλεμετρίας και τηλεχειρισμού. Έτσι χρησιμοποιείται σε περιπτώσεις όπου το προς διαχείριση σύστημα απέχει αρκετά από το χώρο διαχείρισης. Προκειμένου λοιπόν να υλοποιήσουμε ένα SCADA απαιτούνται, εκτός των άλλων, ένα σύστημα τηλεμετρίας. Το σύστημα αυτό υλοποιείται με τη βοήθεια σταθμών RTU (Remote Telemetry Units), οι οποίοι είναι συνδεδεμένοι με τη παραγωγική διαδικασία, «διαβάζουν» τις τιμές διαφόρων φυσικών μεγεθών που μας ενδιαφέρουν, π.χ. πίεση, θερμοκρασία, συχνότητα, τα μετατρέπουν σε ηλεκτρικά σήματα και τα μεταδίδουν μέσω ενός ενσύρματου ή ασύρματου διαύλου (ανάλογα με τις ανάγκες της εφαρμογής) στον υπολογιστή που φέρει το λογισμικά SCADA, σε τακτά χρονικά διαστήματα. Το χρονικό διάστημα που μεσολαβεί ανάμεσα στις διαδοχικές μεταδόσεις εξαρτάται αφενός από την ταχύτητα εξέλιξης της επιτηρούμενης διεργασίας και αφετέρου από την ακρίβεια που επιθυμούμε για το σύστημα μας.

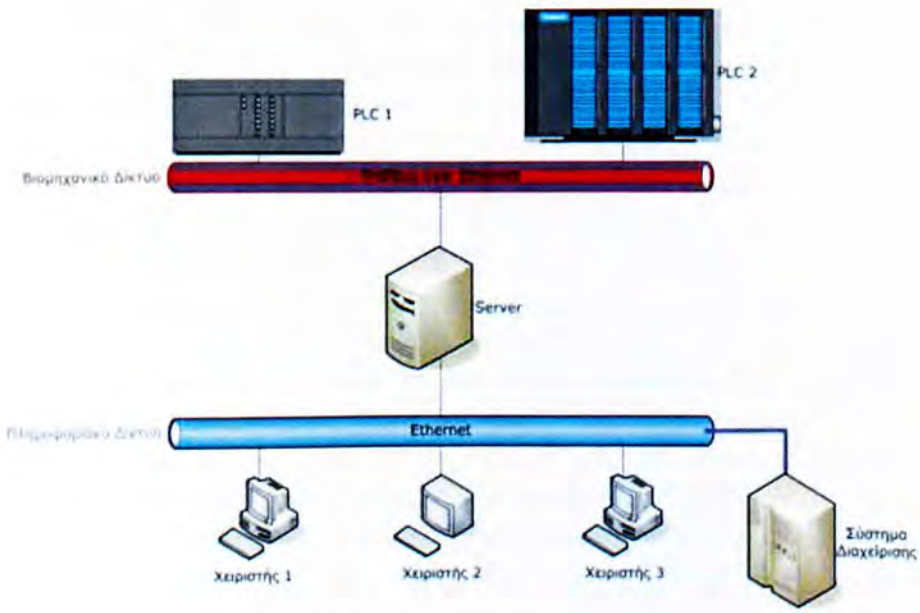
Εκτός των RTU, για την υλοποίηση του SCADA, απαιτούνται και ένας κεντρικός υπολογιστής, αρκετά μεγάλης υπολογιστικής ισχύος, που θα φέρει το λογισμικό SCADA και στον οποίο θα καταλήγουν οι μετρήσεις από όλους τους σταθμούς RTU καθώς και οι απαιτούμενες τηλεπικοινωνιακές ζεύξεις ανάμεσα στους σταθμούς RTU και τον κεντρικό υπολογιστή. Στο ακόλουθο σχήμα φαίνεται μια συνηθισμένη τοπολογία ενός συστήματος SCADA.



Εικόνα 2.1

2.3 Σύγχρονα κατανεμημένα συστημένα SCADA

Στις μέρες μας, όπου οι ηλεκτρονικοί υπολογιστές έχουν κατακλύσει την αγορά, οι κατασκευαστές λογισμικού SCADA καλούνται να αναπτύξουν περιβάλλοντα SCADA το οποία θα μπορούν να «τρέχουν» σε προσωπικούς ηλεκτρονικούς υπολογιστές (PC), θα είναι σχεδιασμένα με βάση την 32bit αρχιτεκτονική των σύγχρονων λειτουργικών συστημάτων καθώς και θα υποστηρίζουν τα πιο διαδεδομένα πρωτόκολλα διασύνδεσης υπολογιστών (Ethernet, TokenRing) αλλά και PLC (Profibus, IndustrialEthernet, TCP/IP). Επίσης, από τα σύγχρονα SCADA απαιτείται να είναι εξαιρετικά φιλικά ως προς τον χρήστη αλλά και να μην απαιτούν ιδιαίτερες γνώσεις προγραμματισμού. Τέλος, πρέπει να είναι πλήρως αντικειμενοστραφή (object-oriented) σε επίπεδο δεδομένων (data) και λειτουργιών (functions), έτσι ώστε να περιέχουν όλους τους απαραίτητους οδηγούς (drivers) για επικοινωνία με τις εισόδους, τις εξόδους και τις μνήμες του PLC. Τα σύγχρονο συστήματα SCADA υλοποιούνται με βάση την κατανεμημένη αρχιτεκτονική της οποία η τοπολογία φαίνεται στο παρακάτω σχήμα .



Εικόνα 2.2 : Τοπολογία ενός συστήματος

Όπως φαίνεται από το σχήμα , η κατανεμημένη αρχιτεκτονική αποτελείται από δύο ανεξάρτητα δίκτυα. Το πρώτο είναι το βιομηχανικό δίκτυο και περιέχει τα PLC. Στην κορυφή του δικτύου αυτού βρίσκεται ένας εξυπηρετητής (server), ο οποίος συνήθως είναι ένας σταθμός εργασίας (π.χ. Alpha, Vaxstation, RISC, Sun SPARK, IBM RS6000 κλπ.) με λειτουργικά σύστημα ικανότητας επεξεργασίας πολλαπλών έργων (multitasking) (όπως π.χ.UNIX, Sun Solaris, WINDOWS-NT κλπ.) ή ένας μικροϋπολογιστής μεγάλης υπολογιστικής ισχύος, όπως ο Pentium 4 3.2GHz με 1 GB RAM, επίσης με λειτουργικό ικανότητας διαχείρισης πολλαπλών έργων (όπως UNIX,WINDOWS-NT, Linux).

Ο εξυπηρετητής αφενός διαχειρίζεται το βιομηχανικό δίκτυο, δηλαδή ελέγχει και καθορίζει της εκπομπές πακέτων από και προς τα PLC, και αφετέρου φέρει δύο βάσεις δεδομένων. Η πρώτη είναι μια βάση δεδομένων πραγματικού χρόνου (Real Time Data Base, RTDB) στην οποία αποθηκεύονται όλες οι πληροφορίες που συλλέγονται από το PLC που βρίσκονται προδεδομένα στην επιτηρούμενη παραγωγική διαδικασία, Η δεύτερη αποτελεί μια σχεσιακή βάση δεδομένων (Relational Data Base-RDB), στην οποία βρίσκονται αποθηκευμένες όλες οι απαραίτητες πληροφορίες για την λειτουργία του συστήματος.
Ένας ακόμη ρόλος του server είναι να εξυπηρετεί τα διάφορα μικροϋπολογιστικά συστήματα που είναι συνδεδεμένα στο δεύτερο ανεξάρτητο δίκτυο του συστήματος SCADA. Πρόκειται για το πληροφοριακό δίκτυο (information network), υψηλής ταχύτητας που συνήθως είναιτύπου Thin ή ThichEthernet (π.χ. Novell). Η αρχιτεκτονική που αναφέραμε παραπάνω είναι ηκλασσική "πελάτη-εξυπηρετητή" (client-server), που συνηθίζεται στα σύγχρονα κατανεμημένα πληροφορικά συστήματα. Κάθε μικροϋπολογιστής στο πληροφοριακό δίκτυο έχει το κατάλληλο λογισμικό εφαρμογής και εκτελεί τις δικές του εφαρμογές ανεξάρτητα των άλλων και ανάλογα με τις απαιτήσεις του συγκεκριμένου χρήστη. Για παράδειγμα ένας χρήστης (χειριστής 2) μπορεί να είναι ο χειριστής της διαδικασίας, η οποία έχει προσπέλαση σε ορισμένα παραγωγικά στοιχεία της υπό έλεγχο διαδικασίας (αφού χρησιμοποιεί απλό τερματικό), ενώ ένας δεύτερος χρήστης (χειριστής 1) μπορεί να είναι ο μηχανικός παραγωγής που έχει προσπέλαση σε όλα τα τμήματα (σημεία) της παραγωγής, αφού χρησιμοποιεί υπολογιστικό σύστημα, Έτσι κάθε χρήστης του συστήματος SCADA έχει το δικό του ηλεκτρονικό υπολογιστή, ή τερματικό, ανάλογα με τις ανάγκες, μέσω του οποίου μπορεί να ανακαλέσει πληροφορίες (information retrieval) που είναι αποθηκευμένες στον εξυπηρετητή, να τις επεξεργαστεί και να τις απεικονίσει κατάλληλα. Ο server συνεπώς έχει διττό ρόλο, αφού αφενός είναι ο συνδετικός κρίκος των δύο ανεξάρτητων δικτύων και αφετέρου αποτελεί τον πυρήνα του συστήματος Συλλογής Πληροφοριών και Εποπτικού Ελέγχου. Πρέπει να γίνει σαφές ότι ένας χρήστης μέσω του υπολογιστή του δεν μπορεί να επέμβει άμεσα στο βιομηχανικό δίκτυο με τα PLC. Αυτό συμβαίνει διότι ο τελικός χρήστης αυτό που

βλέπει και χειρίζεται στην οθόνη του υπολογιστή του είναι μια εικόνα (image) του συστήματος αυτοματισμού, όπως αυτή βρίσκεται στην βάση δεδομένων του server.

Η αρχιτεκτονική αυτή δεν επιβαρύνει το βιομηχανικό δίκτυο από τους χρήστες, αφού τα PLC προσπελαύνονται μόνο από το server και επιπλέον προσφέρει πρόσθετη ασφάλεια των πληροφοριών.
Τέλος, σημαντικό ρόλο στη λειτουργία του SCADA διαδραματίζει η επίδοση του δικτύου, η οποία προφανώς δεν είναι σταθερή σε όλα τα σημεία του δικτύου, αφού τα χρησιμοποιούμενα πρωτόκολλα του βιομηχανικού δικτύου διαφέρουν συνήθως από αυτά του πληροφοριακού. Τα πιο συνηθισμένα πρωτόκολλα για το βιομηχανικό δίκτυο είναι το Profibus, το TCP/IP και το Industrial Ethernet, ενώ για το πληροφοριακό δίκτυο το Ethernet και το Token Ring.

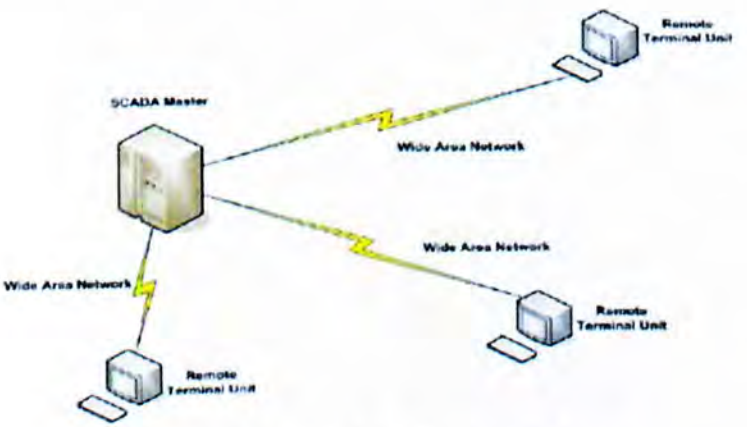
2.4 Αρχιτεκτονική SCADA

Τα συστήματα SCADA έχουν εξελιχθεί παράλληλα με την αύξηση και την βελτίωσης της σύγχρονης τεχνολογίας υπολογιστών. Παρακάτω παρουσιάζεται μια περιγραφή των ακόλουθων τριών γενεών των συστημάτων SCADA:

- Πρώτη γενεά - μονολιθική (Monolithic)
- Δεύτερη γενεά – διανεμημένη (Distributed)
- Τρίτη γενεά - δικτυωμένη (Networked)

2.4.1 Μονολιθικά συστήματα SCADA

Τα συστήματα SCADA ήταν αυτόνομα συστήματα με ουσιαστικά καμία συνδεσιμότητα σε άλλα συστήματα. Τα δίκτυα ευρείας περιοχής (WANs) που εφαρμόστηκαν για να επικοινωνήσουν με τις μονάδες απομακρυσμένων τερματικών (RTUs) σχεδιάστηκαν με έναν ενιαίο και μοναδικό σκοπό, αυτόν της επικοινωνίας με RTUs στον τομέα και τίποτα άλλο. Επιπλέον, τα πρωτόκολλα WAN, σε ευρεία λειτουργία σήμερα, ήταν κατά ένα μεγάλο μέρος άγνωστα τότε.



Εικόνα 2.3 : Μονολιθικά συστήματα Scada

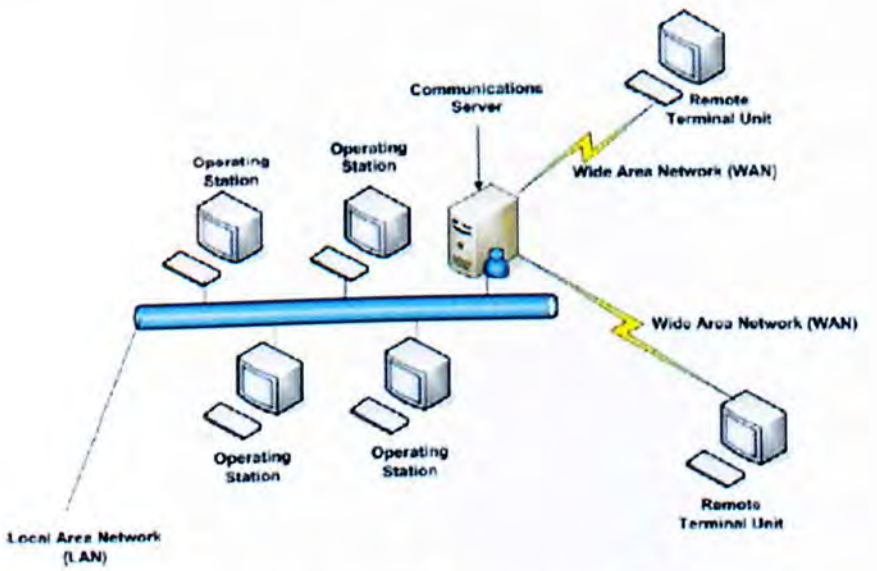
Τα πρωτόκολλα επικοινωνίας σε λειτουργία σχετικά με τα δίκτυα SCADA αναπτύχθηκαν από τους προμηθευτές του εξοπλισμού RTU και ήταν συχνά ιδιόκτητα. Επιπλέον, αυτά τα πρωτόκολλα ήταν γενικά πολύ «αδύνατα», μη ενισχύοντας ουσιαστικά καμία λειτουργία πέρα από την απαραίτητη, την ανίχνευση και τον έλεγχο των σημείων μέσα στη μακρινή συσκευή. Επίσης, δεν ήταν γενικά εφικτό να αναμιχθούν άλλοι τύποι κυκλοφορίας δεδομένων με τις ανακοινώσεις RTU σχετικά με το δίκτυο. Η συνδεσιμότητα στον κύριο σταθμό ήταν πολύ περιορισμένη από τον προμηθευτή συστημάτων. Οι συνδέσεις στον κύριο υπολογιστή γινόταν με

την χρήση ενός διαύλου (bus) η ενός προσαρμογέα δικτύου που συνδεόταν απευθείας με την ΚΜΕ (CPU) του υπολογιστή.

Ο λειτουργικότητα σε αυτά τα συστήματα πρώτης γενεάς ολοκληρώθηκε με την χρήση δύο όμοια εξοπλισμένων συστημάτων κεντρικών υπολογιστών, ένας αρχικός και ένα στήριγμα, που συνδέθηκε στον κοινό δίαυλο (bus). Η αρχική λειτουργία του εφεδρικού συστήματος ήταν να ελεγχθεί ο αρχικός και να αναλάβει σε περίπτωση ανιχνευμένης αποτυχίας. Αυτός ο τύπος εφεδρικής λειτουργίας σήμανε ότι ελάχιστη ή καμία επεξεργασία δεν γινόταν στο εφεδρικό σύστημα.

2.4.2 Διανεμημένα ΣυστήματαSCADA

Η επόμενη γενεά των συστημάτων SCADA εκμεταλλεύτηκε τις εξελίξεις και τη βελτίωση στη μικρογράφηση συστημάτων και την τοπική τεχνολογία δικτύωσης περιοχής (τοπικό LAN) για να διανείμει την επεξεργασία στα πολλαπλάσια συστήματα. Οι πολλαπλάσιοι σταθμοί, κάθε ένας με μια συγκεκριμένη λειτουργία, συνδέθηκαν με το τοπικό LAN και μοιράστηκαν τις πληροφορίες ο ένας με τον άλλον στον πραγματικό χρόνο. Αυτοί οι σταθμοί είχαν χαρακτηριστικά της κατηγορίας μίνι υπολογιστών, μικρότεροι και λιγότερο ακριβοί από τους επεξεργαστές πρώτης γενεάς. Μερικοί από αυτούς τους διανεμημένους σταθμούς χρησιμοποιήθηκαν ως επεξεργαστές επικοινωνιών, με πρώτιστο έργο τους να μπορούν να επικοινωνούν με τις συσκευές τομέων όπως RTUs. Μερικοί χρησιμοποιήθηκαν ως ο χείριστες διασύνδεσης, παρέχοντας την διασύνδεση ανθρώπου - μηχανής (HMI) για τους χειριστές συστημάτων. Ακόμα, άλλοι που χρησιμοποιήθηκαν ως επεξεργαστές υπολογισμού ή ως κεντρικοί υπολογιστές βάσεων δεδομένων. Η διανομή των μεμονωμένων λειτουργιών των συστημάτων SCADA στα πολλαπλάσια συστήματα παρείχε περισσότερη εξουσία επεξεργασίας για το σύστημα συνολικά από την χρήση ενός ενιαίου και μοναδικού επεξεργαστή. Τα δίκτυα που σύνδεσαν αυτά τα μεμονωμένα συστήματα βασίστηκαν γενικά στα πρωτόκολλα του τοπικού LAN και δεν ήταν σε θέση να αναπτύξουν πέρα από τα όρια του τοπικού περιβάλλοντος.



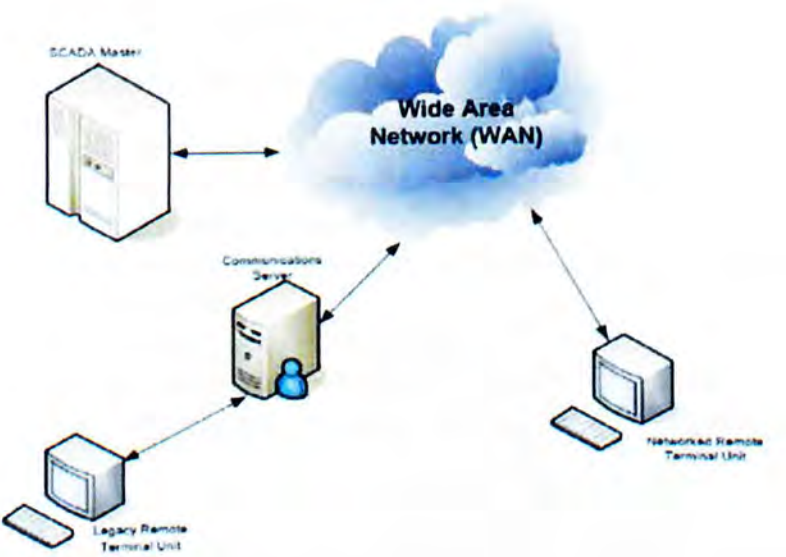
Εικόνα 2.4 : Διανεμημένα συστήματα SCADA

Μερικά από τα πρωτόκολλα του τοπικού LAN που χρησιμοποιήθηκαν ήταν ιδιόκτητης φύσης,όπου ο προμηθευτής δημιούργησε το πρωτόκολλο δικτύων ή την έκδοσή του. Αυτό επέτρεψε σε έναν προμηθευτή να βελτιστοποιήσει το πρωτόκολλο του τοπικού LAN του για την πραγματικό χρόνο κυκλοφορία, αλλά περιόρισε τη σύνδεση του δικτύου από άλλους προμηθευτές στο τοπικό LAN SCADA.

Η διανομή της λειτουργίας συστημάτων στα συνδεμένα στο δίκτυο συστήματα χρησίμευσε όχι μόνο για την αύξηση της δύναμης επεξεργασίας, αλλά και για να βελτιώσει τον πλεονασμό και την αξιοπιστία του συστήματος συνολικά. Παρά το απλό αρχικό/εφεδρικό σχέδιο που χρησιμοποιήθηκε σε πολλά συστήματα πρώτης γενεάς, η διανεμημένη αρχιτεκτονική κράτησε συχνά όλους τους σταθμούς στο τοπικό LAN σε απευθείας σύνδεση πραγματικού χρόνου. Παραδείγματος χάριν, εάν ένας σταθμός HMI επρόκειτο να αποτύχει, ένας άλλος σταθμός HMI θα μπορούσε να χρησιμοποιηθεί για να ενεργοποιήσει το σύστημα, χωρίς αναμονή από το αρχικό σύστημα στο δευτεροβάθμιο. Το δίκτυο (LAN) που χρησιμοποιήθηκε για να επικοινωνήσει με τις συσκευές στον τομέα ήταν κατά ένα μεγάλο μέρος αμετάβλητο από την ανάπτυξη της συνδεσιμότητας του τοπικού LAN μεταξύ των τοπικών σταθμών στον κύριο SCADA. Αυτά τα δίκτυα εξωτερικών επικοινωνιών περιορίστηκαν στα πρωτόκολλα RTU και δεν ήταν διαθέσιμα για άλλους τύπους κυκλοφορίας στα δίκτυα. Όπως συνέβη με την πρώτη γενεά των συστημάτων, η δεύτερη γενεά των συστημάτων SCADA περιορίστηκε επίσης στο υλικό, το λογισμικό, και τις περιφερειακές συσκευές που ήταν αναγκαίες.

2.4.3 Δικτυωμένα Συστήματα SCADA

Η χρήση της αρχιτεκτονικής κύριων σταθμών SCADA είναι στενά συνδεδεμένη με αυτόν της δεύτερης γενεάς, με την αρχική διαφορά ότι είναι μια αρχιτεκτονική ανοικτών συστημάτων παρά μια αρχιτεκτονική ενός προμηθευτής και ενός ελεγχόμενου, ιδιόκτητου περιβάλλοντος . Υπάρχουν ακόμα πολλά δικτυωμένα συστήματα, που διαμοιράζουν τις λειτουργίες κύριων σταθμών. Υπάρχει ακόμα RTUs που χρησιμοποιούν πρωτόκολλα που είναι ιδιόκτητα. Η σημαντικότερη βελτίωση στην τρίτη γενεά είναι αυτή του ανοίγματος της αρχιτεκτονικής συστημάτων, της χρησιμοποίησης των ανοικτών προτύπων και των πρωτοκόλλων και την δυνατότητα να διανείμει τη λειτουργία SCADA σε ένα από έναν μεγάλο WAN και όχι μόνο σε ένα το τοπικό LAN. Τα ανοικτά πρότυπα αποβάλλουν διάφορους περιορισμούς των προηγούμενων γενεών των συστημάτων SCADA. Η χρησιμοποίηση των off-the-shelf συστημάτων διευκολύνει το χρήστη να συνδεθούν περιφερειακές συσκευές τρίτων (όπως τα όργανα ελέγχου, οι εκτυπωτές, οι κινήσεις δίσκων, οι κινήσεις ταινιών, κ.λπ.) με το σύστημα ή και το δίκτυο. Δεδομένου ότι έχουν κινηθεί για «να ανοίξουν» τα συστήματα, οι προμηθευτές SCADA έχουν πάρει βαθμιαία την επιχείρηση ανάπτυξης υλικού. Αυτοί οι προμηθευτές έχουν κοιτάξει στους προμηθευτές συστημάτων όπως Compaq, η Hewlett Packard και Microsoft για την πείρα τους στην ανάπτυξη των βασικών πλατφορμών υπολογιστών και του λογισμικού λειτουργικών συστημάτων. Αυτό επιτρέπει στους προμηθευτές SCADA να συγκεντρώσουν την ανάπτυξή τους σε μια περιοχή όπου μπορούν να προσθέσουν τη συγκεκριμένη αξία σύστημα στο σύστημα - δηλαδή το λογισμικό κύριων σταθμών SCADA. Η σημαντικότερη βελτίωση στα συστήματα τρίτης γενεάς SCADA προέρχεται από τη χρήση των WAN πρωτοκόλλων όπως το πρωτόκολλο Διαδικτύου (IP Internet) για την επικοινωνία μεταξύ του κύριου σταθμού και του εξοπλισμού επικοινωνιών. Αυτό επιτρέπει το τμήμα του κύριου σταθμού που είναι αρμόδιος για τις επικοινωνίες με τις συσκευές τομέων που χωρίζονται από τον κύριο σταθμό να είναι ανεξάρτητος από το WAN. Οι προμηθευτές παράγουν τώρα RTUs που να μπορούν να επικοινωνήσουν με τον κύριο σταθμό χρησιμοποιώντας μια σύνδεση Ethernet.



Εικόνα 2.5 : Δικτυωμένα Συστήματα SCADA

Ένα άλλο πλεονέκτημα που επέρχεται από τη διανομή της λειτουργίας SCADA πέρα από έναν WAN είναι αυτό της ικανότητας επιβίωσης σε περίπτωση καταστροφής. Η διανομή της επεξεργασίας SCADA πέρα από το τοπικό LAN στα δεύτερης γενεάς συστήματα βελτιώνει την αξιοπιστία, αλλά σε περίπτωση συνολικής απώλειας της δυνατότητας του κύριου σταθμού SCADA, ολόκληρο το σύστημα θα μπορούσε να χαθεί επίσης. Με τη διανομή της επεξεργασίας στις φυσικά χωριστές θέσεις, είναι δυνατό να χτιστεί ένα σύστημα SCADA που μπορεί να επιζήσει μιας συνολικής απώλειας οποιασδήποτε θέσης, ακόμη και του κεντρικού σταθμού. Για μερικούς οργανισμούς η βιομηχανίες που βλέπουνε το σύστημα SCADA ως εξαιρετικά κρίσιμη λειτουργία, αυτό είναι ένα πραγματικό όφελος.

2.5 Τα οφέλη των συστημάτωνSCADA

Μερικά από τα πλεονεκτήματα των συστημάτων SCADA :

- Βελτιωμένη λειτουργία των εγκαταστάσεων ή της διαδικασίας με συνέπεια την αποταμίευση λόγω της βελτιστοποίησης του συστήματος
- Αυξανόμενη παραγωγικότητα του προσωπικού
- Βελτιωμένη ασφάλεια του συστήματος λόγω των καλύτερων πληροφοριών και του βελτιωμένου ελέγχου
- Προστασία του εξοπλισμού εγκαταστάσεων
- Προστασία του περιβάλλοντος από μια αποτυχία του συστήματος
- Βελτιωμένη αποταμίευση ενέργειας λόγω της βελτιστοποίησης των εγκαταστάσεων

Το λογισμικό SCADA μπορεί να διαιρεθεί σε δύο τύπους, ιδιόκτητου ή ανοικτού κώδικα . Οι επιχειρήσεις αναπτύσσουν το ιδιόκτητο λογισμικό για να επικοινωνήσουν με το υλικό τους. Αυτά τα συστήματα πωλούνται ως βασικές "λύσεις" συστημάτων SCADA. Το κύριο πρόβλημα με αυτά τα σύστημα είναι η συντριπτική εμπιστοσύνη στον προμηθευτή του συστήματος. Τα ανοικτά συστήματα λογισμικού έχουν κερδίσει τη δημοτικότητα λόγω της διαλειτουργικότητας που φέρνουν στο σύστημα. Η διαλειτουργικότητα είναι η δυνατότητα να αναμιχθεί ο εξοπλισμός των διαφορετικών κατασκευαστών στο ίδιο σύστημα.

Το Citect και WonderWare είναι ακριβώς δύο από τα ανοικτά πακέτα λογισμικού διαθέσιμα στην αγορά για τα συστήματα SCADA. Μερικές συσκευασίες περιλαμβάνουν τώρα τη διαχείριση ενεργητικών στοιχείων που ενσωματώνονται μέσα στο σύστημα SCADA. Τα χαρακτηριστικά συστατικά ενός συστήματος SCADA είναι υποδειγμένα στο επόμενο διάγραμμα.

Τα κύρια χαρακτηριστικά του λογισμικού SCADA είναι:

- Διεπιφάνεια χρήστη
- Γραφικό περιβάλλον
- Συναγερμοί
- Διεπαφή του PLC εξελιξιμότητα
- Πρόσβαση στα δεδομένα
- Βάση δεδομένων
- Δικτύωση
- Ανοχή ελαττωμάτων και πλεονασμός
- Client/server διανεμημένη επεξεργασία

Υπάρχουν χαρακτηριστικά πέντε στόχοι σε οποιοδήποτε σύστημα SCADA. Κάθε μια από αυτές τις διεργασίες εκτελεί και χωριστή επεξεργασία.

- ***Διεργασίες εισόδων - εξόδων***

Αυτό το πρόγραμμα είναι η διεπαφή μεταξύ του συστήματος ελέγχου και παρακολούθησης και των εγκαταστάσεων.

- ***Διεργασίες Συναγερμών***

Διαχειρίζεται όλους τους συναγερμούς με την ανίχνευση των ψηφιακών σημείων συναγερμών και τη σύγκριση των τιμών των αναλογικών σημείων συναγερμών.

- ***Διεργασίες ενεργειών***

Η διεργασία ενεργειών συλλέγει τα στοιχεία που ελέγχονται με την πάροδο του χρόνου.

- ***Διεργασίες αναφορών***

Οι αναφορές συντάσσονται από τα στοιχεία των εγκαταστάσεων. Αυτές οι αναφορές είναι περιοδικές, γεγονός που προκαλείται ή που ενεργοποιείται από το χειριστή.

- ***Ενδείξεις διεργασιών***

Αυτό διαχειρίζεται όλα τα στοιχεία που ελέγχονται από το χειριστή και όλες οι ενέργειες ελέγχου από το χειριστή.

2.6 Σχεδιασμός Γραφικού Περιβάλλοντος Κεντρικού Υπολογιστή

Ο στόχος αυτής της αναφοράς είναι να παρασχεθεί ένα χρήσιμο σύνολο οδηγιών για το σχέδιο ενός αποτελεσματικού γραφικού περιβάλλοντος χειρισμού. Η προσέγγιση πρέπει να είναι τέτοια ώστε να εξασφαλιστεί ότι οι γραφικές παρουσιάσεις είναι όσο το δυνατόν πιο εύκολο να διαβαστούν και να κατανοηθούν. Αυτό μειώνει τη διαδικασία αποκωδικοποίησης στον ανθρώπινο εγκέφαλο σε ένα ελάχιστο επίπεδο και μεγιστοποιεί τις διαδικασίες λήψης αποφάσεων του εγκεφάλου. Αυτό εξασφαλίζει ότι ο χειριστής μπορεί να αντιδράσει γρήγορα και αποτελεσματικά χωρίς να πρέπει να προβληματιστεί που είναι το πρόβλημα. Το χαρακτηριστικό υλικό που παρέχεται είναι:

- Μια ή περισσότερες οθόνες χειρισμού (που μπορούν να είναι και τύπου αφής)
- Βιομηχανικά πληκτρολόγια που πρέπει ευδιάκριτος ή και αφής Πάνελ χειρισμού που αποτελούνται από τα τονισμένα κουμπιά που θα φέρουν επάνω τις προκαθορισμένες γραφικές ενδείξεις
- Εκτυπωτές (ένας για τους συναγερμούς και ένας για τις αναφορές)
- Σειρήνες συναγερμών (ή εξωτερικές σειρήνες)

Οι γραφικές ενδείξεις πρέπει να εμφανιστούν μέσα σε ένα δευτερόλεπτο από την στιγμή που ο χειριστής πιέσει το κατάλληλο πλήκτρο. Η οργάνωση του γραφικού περιβάλλοντος πρέπει να γίνει με έναν σαφή και λογικό τρόπο ώστε να επιτραπεί ο χειριστής γρήγορα και αποτελεσματικά να προσδιορίσει τις πληροφορίες ενδιαφέροντος. Η αρχιτεκτονική των γραφικών χειρισμού είναι να υπάρξει μια προοδευτική μείωση στο πεδίο των γραφικών και μιας προοδευτικής αύξησης λεπτομερών δεδομένου ότι ο χειριστής ψάχνει κάποιες συγκεκριμένες πληροφορίες (και προχωρά κάτω από την ιεραρχία των γραφικών). Οι γραφικές επιδείξεις πρέπει να οργανωθούν σε τρία βασικά στρώματα:

- Το αρχικό επίπεδο που είναι ένα επίπεδο επισκόπησης του συνόλου του συστήματος και που πρέπει να είναι εύκολα προσβάσιμο άμεσα από τα κουμπιά λειτουργίας στο πληκτρολόγιο.
- Το δεύτερο επίπεδο, το οποίο αποτελείται από διάφορες επιδείξεις, που συνδέονται με αυτήν του αρχικού επιπέδου. Αυτές πρέπει να είναι σε θέση να προσεγγιστούν άμεσα από το αρχικό επίπεδο.
- Το τρίτο επίπεδο, το οποίο δίνει περισσότερες λεπτομέρειες στον τομέα των ενδείξεων του δευτέρου επιπέδου ενδείξεων.

Με περισσότερα από τρία επίπεδα, η επίδειξη γίνεται αδικαιολόγητα σύνθετη και αυτό πρέπει να αποφευχθεί εκτός αν είναι απολύτως απαραίτητο. Υπάρχουν διάφορες σχετικές επιδείξεις όπως οι επιδείξεις ενεργειών και οι επιδείξεις βοήθειας. Τα εργαλεία ζουμ είναι χρήσιμα χαρακτηριστικά γνωρίσματα, τα οποία παρέχουν περισσότερη λεπτομέρεια σε μια συγκεκριμένη περιοχή της σχηματικής αναπαράστασης. Οι διάφορες γραφικές οθόνες που είναι διαθέσιμες είναι:

- ***Επιδείξεις ελευθέρου γραφικού περιβάλλοντος***

Εδώ είναι όπου το γραφικό περιβάλλον της οθόνης μπορεί να δημιουργηθεί από τους χρήστες, χρησιμοποιώντας οτιδήποτε σχεδιάγραμμα και σύμβολα μπορούν να δημιουργήσουν. Αυτά κατασκευάζονται καλύτερα από τους χειριστές (με τη βοήθεια από το μηχανικό). Προσφέρουν στο σχεδιαστή την πλήρη ευελιξία στο σχεδιάγραμμα των πληροφοριών.

- ***Επιδείξεις ομάδας λειτουργιών***

Εδώ ένα τυποποιημένο σύνολο συμβόλων χρησιμοποιείται για να δημιουργήσει τις επιδείξεις όπως απαιτείται. Εδώ παρέχονται τα στοιχεία με ένα τυποποιημένο σχήμα παρουσίασης.

- ***Επιδείξεις ενεργειών***

Αυτές οι επιδείξεις καταλαμβάνουν ένα μέρος ή την ολόκληρη οθόνη ανάλογα με τη διαμόρφωση. Παρέχουν τις ενέργειες για τις διαφορές ενδείξεις από τα στοιχεία του συστήματος

- **Επιδείξεις συναγερμών**

Εδώ καταγράφονται οι τρέχοντες συναγερμοί του συστήματος.Έχει αποδειχθεί ότι οι χειριστές συμβουλεύονται και χρησιμοποιούν τις σχηματικές αναπαραστάσεις τύπων επισκόπησης τουλάχιστον δέκα φορές συχνότερα από τις δευτεροβάθμιες και τριτογενείς σχηματικές αναπαραστάσεις. Είναι έτσι επιτακτικό ότι όσο το δυνατόν περισσότερη προσπάθεια να πηγαίνει στο σωστό σχέδιο τέτοιων επιδείξεων. Οι χειριστές πρέπει να ερωτηθούν όσο το δυνατόν περισσότερο στο σχέδιο αυτών των οθονών για να τις καταστήσουν όσο το δυνατόν πιο χρήσιμες. Οι επιδείξεις επισκόπησης πρέπει να καλύψουν ένα μεγάλο ποσό του συστήματος και είναι έτσι σημαντικό να αποβληθεί οποιοδήποτε μέρος της επίδειξης, η οποία δεν μεταβιβάζει εύκολα τις πληροφορίες στο χειριστή. Αυτό θα σήμαινε ότι οι περιλήψεις εξοπλισμού και οι γραμμές ροής δεν τίθενται στις σχηματικές αναπαραστάσεις επισκόπησης. Οι δευτεροβάθμιες και τριτογενείς επιδείξεις επιθυμούν περισσότερες πληροφορίες από τις επιδείξεις της πρώτης βαθμίδας. Οι περιλήψεις των μηνυμάτων εξοπλισμού και κειμένων πρέπει να υπογραμμιστούν με τη χρησιμοποίηση των χαμηλών χρωμάτων έντασης. Μια εικόνα πρέπει να σχεδιαστεί για να μπορεί να δείξει σαφώς την περιοχή που συνδέεται με μια δεδομένη σχηματική αναπαράσταση ή μιαλειτουργία. Αυτό επιτρέπει στο χειριστή για να επιλύσει γρήγορα ποια περιοχή ητρέχουσα επίδειξη αναφέρει πρόβλημα.

2.7 Αναφορές και Σφάλματα

Η επεξεργασία συναγερμών είναι ένα σημαντικό μέρος του σταθμού χειριστών. Οι κώδικες λάθους που προσδιορίζουν τα ελαττώματα συμπεριλαμβάνονται κανονικά με την περιγραφή της αποτυχημένης συσκευής. Κανένα άλλο μέρος της επίδειξης χειριστών δεν ασκεί τόση επίδραση στην υγεία του συστήματος (και αυτή του χειριστή). Η λειτουργία συναγερμών πρέπει να αντιμετωπισθεί, δεδομένου ότι αποτελεί μια άλλη προσέγγιση σε αντιδιαστολή με την καθαρή λίστα οθόνης. Η επίδειξη των συναγερμών είναι τοποθετημένη σε μια σχετική επιτροπή δίπλα στην επίδειξη χειριστών με τα φωτισμένα button. Κάθε button πρέπει ναδείχνει την περιοχή από την οποία οι συναγερμοί προέρχονται και να ενημερώνει πλήρως τον χειριστή μέσο της οθόνης γραφικών. Μόνο τέσσερις προτεραιότητες συναγερμών πρέπει να εφαρμοστούν και να εμφανίζονται στην οθόνη. Αυτοί είναι:

1. Υψηλή Προτεραιότητα
Συναγερμοί που προειδοποιούν για επικίνδυνους όρους που θα μπορούσαν να προκαλέσουν ένα κλείσιμο μιας σημαντικής δραστηριότητας.

2. Μέση προτεραιότητα
Συναγερμοί που πρέπει να ληφθούν υπόψη όσο το δυνατόν γρηγορότερα αλλά δεν μπορούν να προκαλέσουν κλείσιμο μιας σημαντικής διεργασίας.

3. Χαμηλή προτεραιότητα
Συναγερμοί που πρέπει να εξεταστούν όταν υπάρχει χρονικό περιθώριο.

4. Γεγονότα μόνο
Στατιστικές ή τεχνικές πληροφορίες. Κανένας ήχος συναγερμού γι αυτούς.

Ο περιορισμός του αριθμού των τύπων συναγερμών πρόκειται να κρατήσει το σύστημα απλό και με εύκολη ερμηνεία των συναγερμών. Οι συναγερμοί υψηλής προτεραιότητας πρέπει να είναι δυνατότεροι ως προς τον τρόπο αναγγελίας τους. Οι συναγερμοί είναι ταξινομημένοι και πρέπει να αναγνωρίζονται, λάμποντας στην οθόνη και προκαλώντας ηχητικά, ώστε ο χειριστής να τους διαχειρίζεται μέσω του πληκτρολογίου. Μια αδυναμία σε πολλά συστήματα συναγερμών είναι η παρουσία των τετριμμένων συναγερμών, οι οποίοι ενοχλούν και συγχέουν το χειριστή. Είναι σημαντικό να ελέγχεται συνεχώς, να διατηρείται, και να βελτιώνεται το σύστημα συναγερμών μέσω της ανάλυσης και την αναθεώρησης του με τους χειριστές ώστε να έχουμε την βέλτιστη απόδοση του συστήματος.

Για κάθε συναγερμό τα εξής πρέπει να τεκμηριωθούντα εξής:

- Τύπος συναγερμού
- Ετικέτα συναγερμού
- Περιγραφή της ετικέτας
- Σχέση στους σχετικούς συναγερμούς (επακόλουθες συνέπειες)
- Περιγραφή της λογικής στην παραγωγή του συναγερμού
- Πιθανές αιτίες του συναγερμού
- Μέτρα δράσης που μπορούν να ληφθούν για να διορθωθεί η κατάσταση συναγερμού
- Οι συναγερμοί πρέπει να είναι σε θέση να τεθούν εκτός λειτουργίας υπό τον όρο ότιο χειριστής έχει το σχετικό κλειδί ελέγχου του συστήματος.

2.8 Το μέλλον των συστημάτων SCADA

Οι προμηθευτές SCADA κυκλοφορούν μια σημαντική έκδοση και μια έως δύο πρόσθετες δευτερεύουσες εκδόσεις το χρόνο. Αυτά τα προϊόντα εξελίσσονται πολύ γρήγορα ώστε να εκμεταλλευτούν τις νέες ευκαιρίες αγοράς, για να καλύψουν τις νέες απαιτήσεις των πελατών τους και τις νέες τεχνολογίες. Περισσότερα από τα προϊόντα SCADA που αξιολογήθηκαν συνθέτουν τη διαδικασία σε «ατομικές» παραμέτρους στις οποίες συνδέεται ένα ετικέτα-όνομα (Tag-name). Αυτό είναι μη πρακτικό στην περίπτωση πολύ μεγάλων διαδικασιών όταν πρέπει να διαμορφωθούν πολύ μεγάλα σύνολα ετικετών. Δεδομένου ότι οι βιομηχανικές εφαρμογές αυξάνονται στο μέγεθος, οι νέες εκδόσεις SCADA έχουν ως σκοπό τώρα να χειριστούν τις συσκευές όπως και ολόκληρα συστήματα ως πλήρεις οντότητες (classes) που τοποθετούν όλες τις συγκεκριμένες ιδιότητες και τη λειτουργία τους σε αυτά. Επιπλέον, θα υποστηρίζουν την ανάπτυξη πολύ-ομάδων (multi-team development).

Οι μελλοντικές τάσεις είναι τόσο το λογισμικό των PLC όσο και το λογισμικό HMI/SCADA να ενοποιηθούν και να θεωρούνται έως ένα πλήρες και ενοποιημένο πακέτο. Στη μέση της δεκαετίας του '90, ο κατασκευαστές των I/O προσέφεραν τα ιδιόκτητα πρωτόκολλα επικοινωνιών και απλούς τρόπους σύνδεσης των συσκευών. Προς την δεκαετία του '90, η μετατόπιση προς τις ανοικτές επικοινωνίες συνεχίστηκε με τους I/O κατασκευαστές να προσφέρουν την υποστήριξη των ανοικτών δομών μηνυμάτων και ανοιχτών πρωτοκόλλων με σύνδεση μέσω της θύρας Ethernet και του πρωτοκόλλου TCP/IP. Τα αρχικά εμπόδια της εισόδου Ethernet TCP/IP στη βιομηχανική αυτοματοποίηση (αιτιοκρατία, συγχρονισμός, επιλογή πρωτοκόλλου, καταλληλότητα περιβάλλοντος) είναι ακόμα μια ανησυχία σε μερικές εξαιρετικά εξειδικευμένες εφαρμογές, αλλά για τη μεγάλη πλειοψηφία των αγορών HMI/SCADA αυτά τα εμπόδια έχουν σπάσει. Πρόσφατα, εντούτοις, η ίδια η ύπαρξη των συστημάτων SCADA και τις χρήσης του διαδικτύου (TCP/IP) έχει προκαλέσει προβλήματα και ερωτήματα ως αναφορά την ασφάλεια τους καθώς παρατηρούνται όλο και περισσότερες επιθέσεις και τα συστήματα παρουσιάζονται εξαιρετικά τρωτά στις επιθέσεις μέσω το διαδικτύου . Λαμβάνοντας υπόψη την αποστολή και την κρίσιμη φύση ενός μεγάλου αριθμού συστημάτων SCADA, τέτοιες επιθέσεις θα μπορούσαν, στην χειρότερη περίπτωση, να προκαλέσουν τις ογκώδεις οικονομικές απώλειες μέσω της απώλειας δεδομένων ή πραγματικής φυσικής καταστροφής, κακής χρήσης ή κλοπής, ακόμη και απώλειας ζωής, είτε άμεσα είτε έμμεσα. Γίνονται λοιπόν μεγάλες προσπάθειες, κυρίως σε επίπεδο λογισμικού αλλά και υλικού, ώστε στο μέλλον να έχουμε συστήματα SCADA ασφαλέστερα και λιγότερο εκτεθειμένα σε οποιοσδήποτε μορφής επιθέσεων.

Φυσικά με απώτερο και επιθυμητό στόχο όσο το δυνατόν ασφαλέστερα αλλά και γρηγορότερα, σε απόκριση και μεταφορά δεδομένων, συστήματα SCADA.Εάν τέτοιες ανησυχίες θα προκαλέσουν μια απομάκρυνση από τη χρήση των συστημάτων SCADA για τις κρίσιμες εφαρμογές προς τις ασφαλέστερες αρχιτεκτονικές και τις διαμορφώσεις είναι ένα τεράστιο ζήτημα, δεδομένου ότι τουλάχιστον μερικοί

επιδρώντες άνθρωποι στους εταιρικούς και κυβερνητικούς κύκλους θεωρούν ότι τα κέρδη και οι χαμηλότερες αρχικές δαπάνες των βασισμένων σε SCADA συστημάτων αντισταθμίζουν ακόμα τις πιθανούς απώλειες και τους κινδύνους.

3^{Ο} ΚΕΦΑΛΑΙΟ

ΠΡΟΓΡΑΜΜΑΤΙΖΟΜΕΝΟΙ ΛΟΓΙΚΟΙ ΕΛΕΓΚΤΕΣ (PLC)

3.1 Εισαγωγή

Οι Προγραμματιζόμενοι Λογικοί Ελεγκτές PLC (Programmable Logic Controller), είναι μία ειδική κατηγορία συσκευών που η λειτουργία τους βασίζεται στην χρήση μικροεπεξεργαστών, ανάλογων με αυτούς που χρησιμοποιούνται στους ηλεκτρονικούς υπολογιστές. Συμφώνα με τον κανονισμό DIN19226 , το PLC είναι ένα σύστημα έλεγχου το όποιο επεξεργάζεται σήματα εισόδου και με συγκεκριμένοι απαίτηση (πρόγραμμα) , ενεργοποιεί στοιχειά εξόδου . Ο αρχικός προσδιορισμός του ήταν να αντικαταστήσει τα κυκλώματα εντολοδότησης του κλασσικού αυτοματισμού (ρελέ) με τις πολύπλοκες συνδέσεις του . Στην συνεχεία όμως , η εξέλιξη της τεχνολογίας το διαφοροποιήθηκε και το προσάρμοσε στις απαιτήσεις των συγχρόνων εφαρμογών. Έτσι το PLC σήμερα εκτός του να μετρά, να συγκρίνει , να υπολογίζει και να ελέγχει αναλογικά και ψηφιακά σήματα, είναι σε θέση να παρακολουθεί ,να αλλάζει παραμέτρους μέσω οθόνης HMI (HumanMachineInterface), να διαθέτει εντολές έλεγχου θέσεις σε βηματικούς κινητήρες, να έχει ενσωματωμένη δυνατότητα έλεγχου PID κ.α. Το σύστημα PLC δεν είναι σχεδιασμένο για μια συγκεκριμένη εφαρμογή αλλά αντιπροσωπεύει μια γενική μορφή έλεγχου, η οποία προσαρμόζεται ανάλογα με τις απαιτήσεις της κάθε εφαρμογής.

3.2 Προγραμματιζόμενοι Λογικοί Ελεγκτές PLC

Ένας *προγραμματιζόμενος λογικός ελεγκτής* (PLC), είναι μια συσκευή στερεάς κατάστασης σχεδιασμένη να εκτελεί λογικές λειτουργίες, που μέχρι τώρα επιτυγχάνονταν με ηλεκτρομηχανικούς ηλεκτρονόμους.

Η σχεδίαση των περισσοτέρων PLC είναι παρόμοια με αυτή ενός Η/Υ. Βασικά ο PLC είναι μια ομάδα λογικών ψηφιακών στοιχείων στερεάς κατάστασης σχεδιασμένη να παίρνει λογικές αποφάσεις και να παρέχει εξόδους. Οι PLC χρησιμοποιούνται για τον έλεγχο και την λειτουργία βιομηχανικής παραγωγής συσκευών και μηχανημάτων.

Οι προγραμματιζόμενοι λογικοί ελεγκτές είναι ένα σχέδιο υπολογιστή για χρήση στον έλεγχο των μηχανών. Σε αντίθεση με τους Η/Υ έχει σχεδιαστεί να λειτουργεί σε βιομηχανικό περιβάλλον και είναι εφοδιασμένος με ειδικές διασυνδέσεις εισόδου/εξόδου και με μια γλώσσα ελέγχου προγραμματισμού. Η βασική συντομία που χρησιμοποιείται στη βιομηχανίαγια αυτές τις μονάδες είναι το PC, αν και μπορεί να υπάρχει "σύγχυση" γιατί χρησιμοποιείται σαν συντομία ο όρος αυτός και για τους ηλεκτρονικούς υπολογιστές. Γι' αυτό πολλοί αναφέρονται στους προγραμματιζόμενους ελεγκτές με τον όρο PLC, ο οποίος αντιπροσωπεύει το "προγραμματιζόμενοι λογικοί ελεγκτές". Παρακάτω παραθέτονται διάφοροι προγραμματιζόμενοι λογικοί ελεγκτές από εταιρείες που δραστηριοποιούνται στο χώρο των αυτοματισμών.

Εικόνα 3.1 :PLCSimatic της Siemens

Εικόνα 3.2: PLC της εταιρείας Telemecanique

Εικόνα 3.3 : PLC της εταιρείας Mistubishi

Αρχικά ο PLC χρησιμοποιήθηκε για να αντικαταστήσει την λογική των ηλεκτρονόμων, αλλά από τότε αύξησε την κλίμακα των λειτουργιών του, με την έννοια ότι βρέθηκε σε περισσότερες και πιο πολύπλοκες μορφές.

Επειδή η δομή ενός PLC βασίζεται στις ίδιες αρχές με αυτές που εφαρμόζονται στην αρχιτεκτονική υπολογιστών, είναι ικανό όχι μόνο στην εκτέλεση των εργασιών ενός ηλεκτρονόμου αλλά και στην εκτέλεση άλλων εφαρμογών όπως μετρήσεις, υπολογισμού σύγκρισης και επεξεργασία λογικών σημάτων. Οι προγραμματιζόμενοι ελεγκτές παρέχουν αρκετά **πλεονεκτήματα** σε σχέση με τον έλεγχο με συμβατικό τύπο ηλεκτρονόμων. Οι ηλεκτρονόμοι πρέπει να καλωδιωθούν για να εκτελέσουν μια συγκεκριμένη λειτουργία. Όταν οι απαιτήσεις του συστήματος αλλάζουν, η καλωδίωση των ηλεκτρονόμων πρέπει να αλλάξει ή να αντικατασταθούν. Σε εξαιρετικές περιπτώσεις, όπως είναι η βιομηχανία αυτοκινήτων, ολόκληροι πίνακες ελέγχου έπρεπε να αντικατασταθούν, την στιγμή που ήταν οικονομικά αδύνατο να τροποποιηθούν οι παλιοί πίνακες για κάθε αλλαγή μοντέλου. Ο προγραμματιζόμενος ελεγκτής έχει σχεδόν εξαλείψει την καλωδίωση με το χέρι, που συναντάται στα συμβατικά κυκλώματα ελέγχου επεξεργασίας, που βασίζεται σε ηλεκτρονόμους. Οι προγραμματιζόμενοι ελεγκτές παρέχουν επίσης αξιοπιστία στερεάς κατάστασης, χαμηλότερη κατανάλωση ισχύος, και ευκολία επέκτασης. Εάν μια εφαρμογή έχει πάνω από 6 ηλεκτρονόμους, η αντικατάσταση της με PLC πιθανόν να είναι μεγαλύτερου κόστους. Η εξομοίωση 100 ηλεκτρονόμων, χρονικών και μετρητών δεν είναι πρόβλημα ούτε καν στους μικρούς PLC. Οι προγραμματιζόμενοι λογικοί ελεγκτές είναι εύκολο να προγραμματιστούν και να εγκατασταθούν. Η πρόσβαση στους PLC μπορεί να είναι απαγορευτική από τα χαρακτηριστικά του hardware (διακόπτες με συνδυασμούς) και από τα χαρακτηριστικά του software (με κωδικούς).

Η επίλυση προβλημάτων με PLC είναι ένα σημαντικό πλεονέκτημα έναντι των συστημάτων ελέγχου με ηλεκτρονόμους. Οι PLC μπορούν να σχεδιαστούν με ικανότητες επικοινωνίας που τους επιτρέπουν να συνδέονται με άλλα συστήματα υπολογιστών ή να παρέχουν διασύνδεση με τους ανθρώπους. Ο προγραμματιζόμενος ελεγκτής είναι μια συσκευή καθοδηγούμενη από το γεγονός, το οποίο σημαίνει ότι ένα γεγονός που συμβαίνει στο πεδίο θα έχει σαν αποτέλεσμα να συμβεί μια λειτουργία ή να δοθεί μια έξοδο.

3.3 Πλεονεκτήματα των προγραμματιζόμενων λογικών ελεγκτών

Τα πρώτα μεγάλα πλεονεκτήματα των PLCαφορούν τους κατασκευαστές εξοπλισμού αυτοματισμών και πινάκων αυτοματισμού και ήδη τα έχουμε αναφέρει:

1. Το κόστος κατασκευής ενός PLCείναι σημαντικά μικρότερο από το κόστος παραγωγής ενός μεγάλου αριθμού βοηθητικών ηλεκτρονόμων, χρονικών και απαριθμητών.

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

Υπάρχουν όμως πολλά πλεονεκτήματα που έχουν σχέση με τον τελικό χρήστη, τις βιομηχανίες δηλαδή που εφαρμόζουν τους αυτοματισμούς, και είναι αυτά που μας ενδιαφέρουν περισσότερο. Κατά σειρά σπουδαιότητας αναφέρουμε:

- ✓ Τα PLC ελαχιστοποιούν το κόστος συντήρησης του πίνακα αυτοματισμού. Το κόστος αυτό αναλύεται ως εξής: Συχνότητα βλαβών, χρόνος εντοπισμού μιας βλάβης και αποκατάστασής της. Δηλαδή, όταν υπάρχει μια βλάβη στον πίνακα μιας εγκατάστασης κλασικού αυτοματισμού, υπάρχει καθυστέρηση στην παραγωγή μέχρι να εντοπιστεί η βλάβη. Αφού εντοπιστεί, πρέπει να έχουμε διαθέσιμο το κατάλληλο ανταλλακτικό στην αποθήκη, γιατί διαφορετικά θα υπάρξει σημαντική καθυστέρηση, αφού θα χρειαστεί να γίνει η σχετική παραγγελία και η προμήθεια. Στον αυτοματισμό με PLCδεν υπάρχει ουσιαστικό θέμα βλάβης εσωτερικά στον πίνακα της εγκατάστασης. Θα πείτε, δεν "χαλά" το PLC; Αυτό συμβαίνει σπάνια και οι εγγυήσεις είναι πάρα πολύ μεγάλες.

- ✓ Τα PLCείναι ευέλικτα στην τροποποίηση της λειτουργίας του αυτοματισμού. Δηλαδή, αν υποθέσουμε ότι θέλουμε να κάνουμε μια αλλαγή στον αυτοματισμό, αυτή μπορεί να γίνει μέσα σε λίγα λεπτά, αρκεί μόνο να αλλάξουμε το πρόγραμμα. Σε ένα πίνακα κλασικού αυτοματισμού, τέτοιου είδους αλλαγές είναι πράγμα πολύ δύσκολο και χρονοβόρο.

- ✓ Ο αυτοματισμός με PLC επεκτείνεται πολύ εύκολα. Αυτό γίνεται είτε απλά αλλάζοντας το πρόγραμμα, είτε με την τοποθέτηση νέων μονάδων εισόδων και εξόδων. Κάθε επέκταση στον κλασικό αυτοματισμό είναι πολύ δύσκολη.

- ✓ Ο αυτοματισμός με PLCμας παρέχει καταπληκτικές δυνατότητες. Μπορούμε να δημιουργούμε πολύ εύκολα πολύπλοκες και έξυπνες επεξεργασίες, οι οποίες στον κλασικό αυτοματισμό είναι εξαιρετικά δύσκολο να υλοποιηθούν.

- ✓ Σε μια εγκατάσταση, που χρησιμοποιεί αυτοματισμούς με PLC, σήμερα παρέχονται δυνατότητες σύνδεσης με κεντρικό ηλεκτρονικό υπολογιστή, σύνδεσης με το σύστημα αποθήκης, λογιστηρίου κ.λπ.

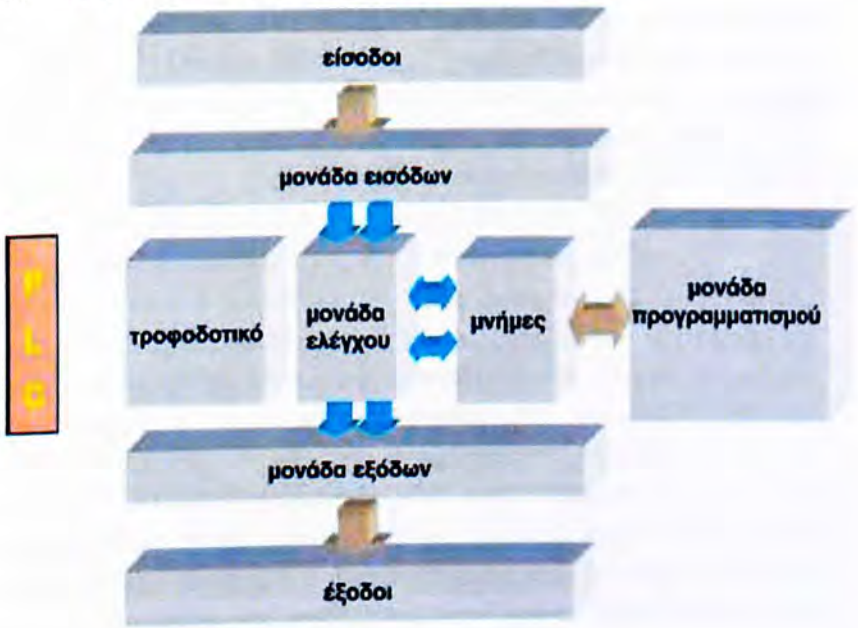
- ✓ Το PLCκαταλαμβάνει ελάχιστο χώρο σε σχέση με τον αντίστοιχο πίνακα κλασικού αυτοματισμού.

Βλέπουμε ότι από τη χρήση των PLC προκύπτουν μόνο πλεονεκτήματα.

3.4 Η δομή ενός προγραμματιζόμενου λογικού ελεγκτή.

Στην αγορά υπάρχουν σήμερα εκατοντάδες μοντέλα PLC κατασκευασμένα από πλήθος διαφορετικών εταιρειών. Γενικά, σε ένα PLC μπορούμε να διακρίνουμε τα παρακάτω μέρη:

- Την κεντρική μονάδα επεξεργασίας (CentralProcessingUnit, CPU), που αποτελεί και την καρδιά ή μάλλον τον εγκέφαλο του PLC.
- Τη μονάδα τροφοδοσίας.
- Τις μονάδες εισόδων - εξόδων (I/Omodules).



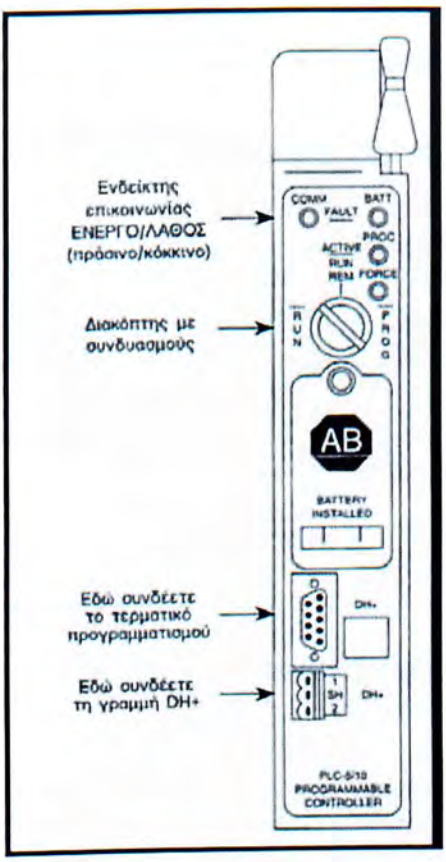
Εικόνα 3.4: Δομή PLC

Η κεντρική μονάδα, η μονάδα τροφοδοσίας και οι μονάδες εισόδων - εξόδων αποτελούν την κύρια μονάδα αυτοματισμού, δηλαδή το κύριο μέρος του PLC. Υπάρχουν δύο τρόποι με τους οποίους το τμήμα I/O ενσωματώνεται στους PLC: σταθερά και ρυθμιζόμενα. Το σταθερό I/O είναι τυπικό στους μικρούς PLC, τα οποία έρχονται με ενσωματωμένα σταθερά κομμάτια. Ο επεξεργαστής και η μονάδα I/O τοποθετούνται μαζί και τα τερματικά I/O είναι διαθέσιμα αλλά δεν μπορούν να μεταβληθούν. Το βασικό πλεονέκτημα αυτού του τύπου τοποθέτησης είναι το μικρότερο κόστος. Ο αριθμός των διαθέσιμων I/O σημείων ποικίλει, και συχνά μπορεί να διευρυνθεί αγοράζοντας επιπρόσθετες μονάδες σταθερών I/O. Ένα μειονέκτημα των σταθερών I/O είναι η έλλειψη ελαστικότητας: είσαι περιορισμένος σχετικά με το τι μπορείς να πάρεις, στην ποσότητα και στους τύπους που υπαγορεύονται από την τοποθέτηση. Επιπλέον σε πολλά μοντέλα, εάν ένα από τα τμήματα καταστραφεί, το όλο σύστημα θα πρέπει να αλλάξει.

Το ρυθμιζόμενο τμήμα I/O διαιρείται σε τμήματα μέσα στα οποία μπορούν να συνδεθούν ξεχωριστές μονάδες. Το χαρακτηριστικό αυτό αυξάνει υπερβολικά τις επιλογές και την ελαστικότητα της μονάδας. Μπορεί να επιλεγούν από τις μονάδες που είναι διαθέσιμες από τους κατασκευαστές και να συνδυαστούν με οποιονδήποτε επιθυμητό τρόπο. Η βασική μονάδα ελέγχου αποτελείται από μια βάση (rack), τη μονάδα επεξεργασίας ,τον χειριστή διασύνδεσης για τον προγραμματισμό και την απεικόνιση. Οι μονάδες συνδέονται στη βάση.

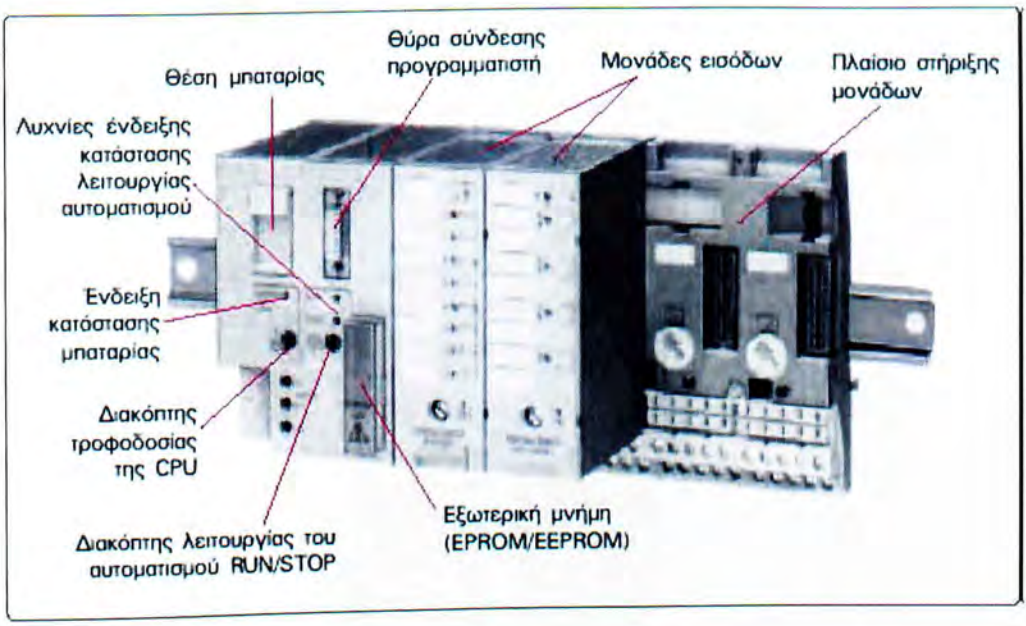
Όταν μια μονάδα ολισθαίνει στη βάση, δημιουργεί μια ηλεκτρική σύνδεση με μια σειρά επαφών, πάνω σε μια πλακέτα τυπωμένου κυκλώματος (backplane) και η οποία συνδέεται στο τέλος της βάσης (μονάδα συρταρωτής σύνδεσης). Ο επεξεργαστής του PLC ενώνεται επίσης στην πλακέτα του κυκλώματος και μπορεί να επικοινωνεί με όλες τις μονάδες στη βάση. Η μονάδα τροφοδοσίας, τροφοδοτεί με dc ισχύ τα άλλα κομμάτια τα οποία συνδέονται στη βάση. Για μεγάλα συστήματα PLC, η μονάδα τροφοδοσίας δεν τροφοδοτεί με φυσιολογική τάση τις συσκευές πεδίου. Σε μεγαλύτερα συστήματα, η ισχύς στις συσκευές πεδίου παρέχεται από εξωτερική πηγή εναλλασσομένου ή συνεχούς ρεύματος. Για μικρά και πολύ μικρά PLC συστήματα, η μονάδα τροφοδοσίας χρησιμοποιείται για να τροφοδοτεί συσκευές πεδίου.

Ο επεξεργαστής (CPU) είναι ο εγκέφαλος του PLC. Ένας τυπικός επεξεργαστής συνήθως αποτελείται από ένα μικροεπεξεργαστή για να εκτελεί την λογική και να ελέγχει τις επικοινωνίες ανάμεσα στα κομμάτια. Ο επεξεργαστής απαιτεί μνήμη για να αποθηκεύει τα αποτελέσματα των λογικών λειτουργιών που εκτελούνται από τον μικροεπεξεργαστή. Μνήμη απαιτείται επίσης για το πρόγραμμα EPROM ή EEPROM καθώς και για τη RAM. Η CPU είναι σχεδιασμένη έτσι ώστε ο χρήστης να μπορεί να εισάγει το απαιτούμενο κύκλωμα σε λογική κλίμακα ηλεκτρονόμων (relay ladder logic). Ο επεξεργαστής δέχεται (διαβάζει) δεδομένα εισόδου από διάφορες συσκευές ανίχνευσης, εκτελεί το αποθηκευμένο πρόγραμμα του χρήστη από μνήμης και στέλνει τις κατάλληλες εντολές στις συσκευές ελέγχου. Μια πηγή ισχύος συνεχούς ρεύματος απαιτείται για να παράγει την τάση χαμηλής στάθμης που χρησιμοποιείται στον επεξεργαστή. Το τροφοδοτικό αυτό μπορεί να τοποθετηθεί μέσα στην μονάδα της CPU ή μπορεί να είναι μια ξεχωριστή μονάδα, ανάλογα με τον κατασκευαστή του συστήματος PLC.



Εικόνα 3.5 : Τυπική μονάδα επεξεργασίας

Το τμήμα I/O αποτελείται από τις μονάδες εισόδου και εξόδου .Το σύστημα I/O αποτελεί την μονάδα διασύνδεσης (interface), με την οποία έρχονται σε επαφή οι συσκευέςπεδίου με τον ελεγκτή. Ο σκοπός αυτής της διασύνδεσης είναι να προσαρμόσει τα διάφορασήματα που λαμβάνονται ή στέλνονται στις εξωτερικές συσκευές πεδίου.



Εικόνα 3.6: Ένα PLC οπού διακρίνονται όλα τα στοιχειά του

Συσκευές εισόδου, όπως διακόπτες αυτόματης επαναφοράς (push buttons), διακόπτες ορίων (limit switches), ανιχνευτές (sensors), επιλεκτικοί διακόπτες (selector switches), και διακόπτες που χειρίζονται με τα δάκτυλα, καλωδιώνονται στις τερματικές μονάδες εισόδου των συσκευών. Συσκευές εξόδου, όπως μικροί κινητήρες, εκκινητές κινητήρων, σωληνοειδή (solenoid valves) και ενδεικτικές λυχνίες (indicator lights), καλωδιώνονται στις τερματικές μονάδες εξόδου των συσκευών. Αυτές οι συσκευές επίσης αναφέρονται σαν πεδίου (field) ή πραγματικού κόσμου (real world) είσοδοι — έξοδοι. Οι όροι "πεδίου" ή "πραγματικού κόσμου" χρησιμοποιούνται για να διαχωρίσουν τις πραγματικές εξωτερικές συσκευές οι οποίες υπάρχουν και πρέπει φυσικά να συνδεθούν με το εσωτερικό πρόγραμμα του χρήστη το οποίο αντιγράφει την λειτουργία των ηλεκτρονόμων, χρονικών και μετρητών.Η συσκευή προγραμματισμού ή τερματική συσκευή, χρησιμοποιείται για να εισάγει το α παιτούμενο πρόγραμμα στη μνήμη του επεξεργαστή.

Το πρόγραμμα αυτό εισάγεται χρησιμοποιώντας λογική κλίμακα ηλεκτρονόμων (relay ladder logic). Το πρόγραμμα καθορίζειτην ακολουθία της λειτουργίας και τον απόλυτο έλεγχο της συσκευής ή του μηχανήματος. Η συσκευή προγραμματισμού πρέπει να συνδέεται στον ελεγκτή μόνο όταν εισάγουμε ή παρακολουθούμε το πρόγραμμα. Σχεδιάζοντας τον ελεγκτή έτσι ώστε να είναι "φιλικός στον τεχνικό", ο PLC μπορεί να προγραμματιστεί από ανθρώπους χωρίς εκτεταμένη εμπειρία προγραμματισμού Η/Υ Ο πραγματικός προγραμματισμός συνήθως επιτυγχάνεται πατώντας πλήκτρα σ' ένα πληκτρολόγιο.

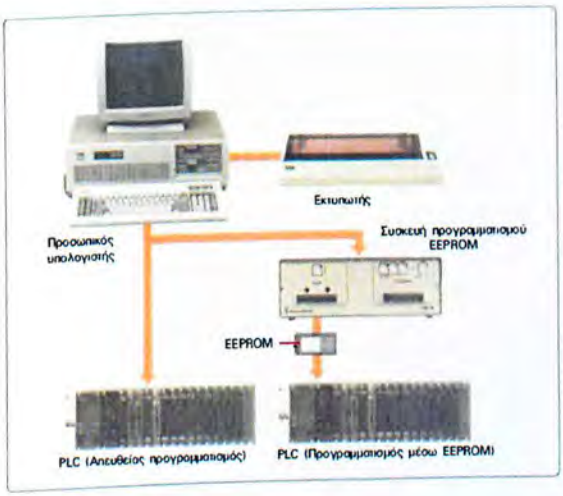
Η συσκευή προγραμματισμού μπορεί να είναι μια συσκευή χειρός με απεικόνιση διόδων εκπεμπόμενου φωτός (LED) (Εικόνα 3.7) ή μία μονάδαβιομηχανικού τερματικού με οθόνη (Εικόνα 3.8). Σε μερικές μικρές προγραμματιζόμενεςσυσκευές χειρός, το πρόγραμμα εισάγεται χρησιμοποιώντας σύμβολα αριθμητικών πράξεων της άλγεβρας Boole (AND, OR και NOT) ανεξάρτητα ή σε συνεργασία μεταξύ τους,σχηματίζοντας λογικές προτάσεις. Μεγαλύτερα συστήματα PLC χρησιμοποιούν, είτε ένα Η/Υ,είτε προγραμματιζόμενα τερματικά μαζί με την λογική κλίμακα ηλεκτρονόμων. Ο σκοπός είναι να χρησιμοποιούμε Η/Υ για τον προγραμματισμό.

Εικόνα 3.7 : Μονάδα Χειρός με οθόνη LED (Συσκευή προγραμματισμού)

Εικόνα 3.8 :Εργαστηριακή τερματική μονάδα εικόνας (Συσκευή προγραμματισμού)

Εικόνα 3.9 : Προσωπικός Η/Υ με κατάλληλο software (Συσκευή προγραμματισμού)

Η εργαστηριακή οθόνη, όπως αυτή που φαίνεται στο παραπάνω σχήμα, προσφέρει το πλεονέκτημα να δείχνει μεγάλη ποσότητα λογικού στην οθόνη και έτσι να απλοποιεί την ερμηνεία του προγράμματος. Η μονάδα προγραμματισμού επικοινωνεί με τον επεξεργαστή μέσω μια παράλληλης ή σειριακής σύνδεσης επικοινωνίας δεδομένων. Εάν μια μονάδα προγραμματισμού δεν χρησιμοποιείται, μπορεί να αποσυνδεθεί και να απομακρυνθεί. Απομακρύνοντας την μονάδα προγραμματισμού, δεν θα επηρεάσουμε την λειτουργία χρήσης του προγράμματος. Ένας προσωπικός υπολογιστής με το κατάλληλο λογισμικό (software) μπορεί επίσης να δράσει σαν τερματική συσκευή προγράμματος καθιστώντας δυνατή την απομάκρυνση του προγραμματιστή από τη φυσική θέση του προγραμματιζόμενου λογικού ελεγκτή. Όταν το πρόγραμμα ολοκληρωθεί, αποθηκεύεται σε μια μορφή μαζικής αποθήκευσης και επαναφορτίζεται στον προγραμματιζόμενο λογικό ελεγκτή, όταν απαιτείται.

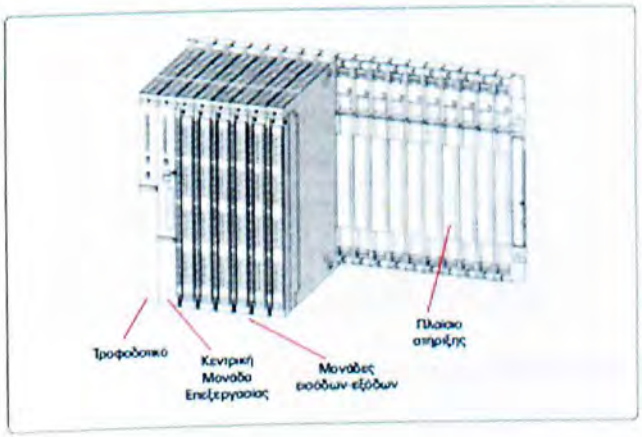


Εικόνα 3.10 :Προγραμματισμός PLC με την βοήθεια προσωπικού Η/Υ

3.4.1 Πλαίσιο τοποθέτησης μονάδων.

Οι μονάδες ενός μεγάλου PLC τοποθετούνται σε ένα κεντρικό πλαίσιο. Στο πλαίσιο αυτό είναι ενσωματωμένο ένα σύστημα αγωγών (σύστημα ζυγών) μέσω των οποίων επικοινωνούν οι διάφορες μονάδες με την Κεντρική Μονάδα Επεξεργασίας.

Αν οι θέσεις του κεντρικού πλαισίου, που διατίθεται, δεν επαρκούν για να τοποθετηθούν οι μονάδες εισόδων και εξόδων που απαιτούνται σε μια συγκεκριμένη εφαρμογή, τότε χρησιμοποιούνται ένα ή περισσότερα πλαίσια επέκτασης για την τοποθέτηση των πρόσθετων μονάδων. Κάθε πλαίσιο επέκτασης διασυνδέεται με το κεντρικό πλαίσιο ή με τα άλλα πλαίσια επέκτασης μέσω ειδικής μονάδας διασύνδεσης και καλωδίου. Κάθε εταιρεία και το κάθε μοντέλο έχει το δικό του σύστημα πλαισίου. Στην εικόνα 3.11 δείχνεται ένα πλαίσιο στήριξης PLC.



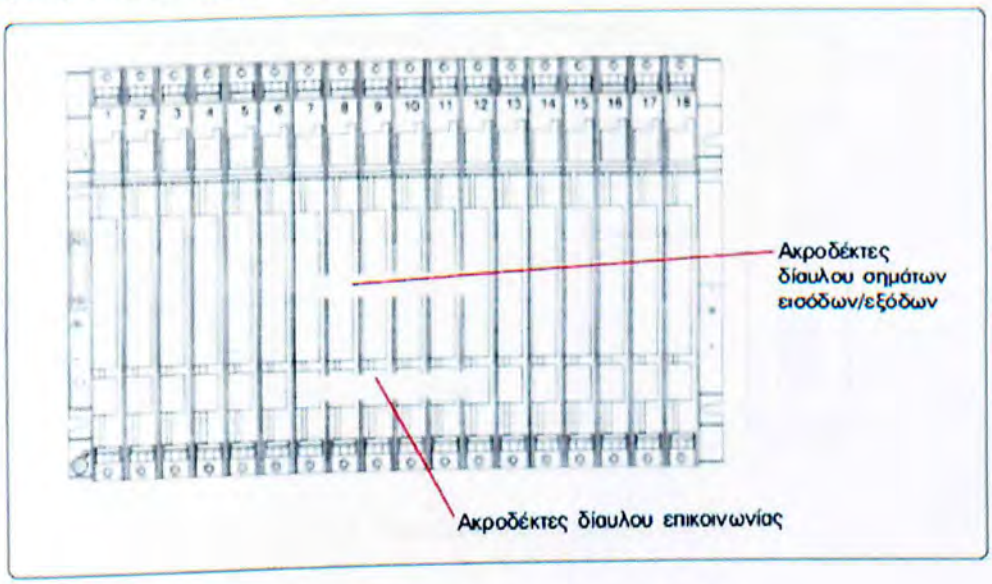
Εικόνα 3.11 :Δομή ενός PLC

3.4.2 Μονάδα τροφοδοσίας.

Η μονάδα τροφοδοσίας ενός PLCέχει σκοπό να δημιουργήσει από την τάση του δικτύου τροφοδοσίας τις απαραίτητες εσωτερικές τάσεις, που απαιτούνται για την τροφοδοσία των ηλεκτρονικών στοιχείων (τρανζίστορ, ολοκληρωμένα κυκλώματα κ.λπ.) του PLC. Οι τυπικές εσωτερικές τάσεις των PLCείναι συνήθως: DC5 V, DC9 V, DC24 V.

Σε ορισμένα μοντέλα PLC, όταν το PLCδεν τροφοδοτείται από το δίκτυο, η μονάδα τροφοδοσίας διατηρεί το περιεχόμενο της μνήμης του PLCμε την βοήθεια μιας μπαταρίας (συνήθως λιθίου), που διαθέτει. Σε άλλα μοντέλα PLCη παραπάνω μπαταρία βρίσκεται στην Κεντρική Μονάδα Επεξεργασίας.

Χρειάζεται προσοχή ώστε να μην προκαλέσουμε υπερφόρτιση της μονάδας τροφοδοσίας. Για τον σκοπό αυτό συμβουλευόμαστε τα τεχνικά φυλλάδια της εταιρείας κατασκευής του PLC.

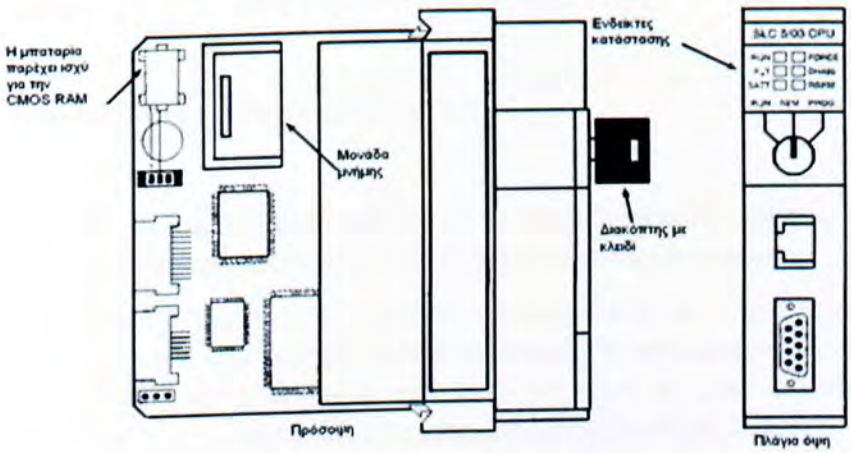


Εικόνα 3.12 : Πλαίσιο στήριξης PLC

3.4.3Κεντρική Μονάδα Επεξεργασίας.

Είναι η βασική μονάδα του PLC, η οποία είναι υπεύθυνη για τη λειτουργία του αυτοματισμού. Η κεντρική μονάδα επεξεργασίας είναι στην ουσία ένας μικροϋπολογιστής και διακρίνουμε σ' αυτήν όλα τα κύρια μέρη ενός μικροϋπολογιστή, δηλαδή τον μικροεπεξεργαστή και τη μνήμη.

Ο μικροεπεξεργαστής είναι ένα ολοκληρωμένο κύκλωμα, το οποίο αποτελεί τον "εγκέφαλο" κάθε μικροϋπολογιστή. Οι μικροεπεξεργαστές εξελίσσονται με ταχύτατους ρυθμούς το όνομά τους δίνει συνήθως και το όνομα στο μοντέλο του μικροϋπολογιστή (π.χ. λέμε PC386, 486, Pentiumκ.λπ.). Στα PLC πολύ λίγο μας ενδιαφέρει να ξέρουμε ποιον μικροεπεξεργαστή χρησιμοποιεί η κεντρική μονάδα, αν και πολλές φορές μπορούμε να το διαβάσουμε στα χαρακτηριστικά που δίνουν οι εταιρείες. Ο μικροεπεξεργαστής για το PLC είναι ο κύριος υπεύθυνος για όλες τις λειτουργίες του (τη λειτουργία του PLC περιγράφουμε στην επόμενη ενότητα).



Εικόνα 3.13 :Τυπική μονάδα επεξεργασίας

Η μνήμη της κεντρικής μονάδας διακρίνεται σε μνήμη RAM, ROMκαι EEPROM.

Μνήμη τυχαίας προσπέλασης (RAM ή R/W)

Ημνήμη RAM (RandomAccessMemory, μνήμη τυχαίας προσπέλασης) είναι εκείνη στην οποία μπορούμε να γράφουμε και να σβήνουμε, και η οποία σβήνει μόλις λείψει η ηλεκτρική τροφοδοσία της. Στη μνήμη RAM η κεντρική μονάδα αποθηκεύει μια σειρά από πληροφορίες σε ξεχωριστές περιοχές εργασίας. Μπορούμε να διακρίνουμε τις εξής περιοχές:

- Περιοχή μνήμης όπου αποθηκεύονται οι καταστάσεις των εισόδων και των εξόδων. Η περιοχή αυτή ονομάζεται για τις εισόδους "εικόνα εισόδων" και για τις εξόδους "εικόνα εξόδων".

- Περιοχή μνήμης όπου αποθηκεύονται οι ενδιάμεσες πληροφορίες, που αφορούν τη λειτουργία του αυτοματισμού.

- Περιοχή μνήμης των χρονικών.

- Περιοχή μνήμης των απαριθμητών.

- Περιοχή μνήμης όπου αποθηκεύονται τα προγράμματα του χρήστη, δηλαδή τα προγράμματα που λειτουργούν ένα συγκεκριμένο αυτοματισμό.

Η μνήμη RAM σβήνει μόλις σταματήσει η τροφοδοσία της. Όμως το πρόγραμμα που λειτουργεί τον αυτοματισμό πρέπει να παραμένει αναλλοίωτο στη μνήμη και αφού κλείσουμε την τροφοδοσία του PLC. Γι' αυτό το λόγο η μνήμη RAM παραμένει πάντα σε τροφοδοσία μέσω μιας μπαταρίας (συνήθως λιθίου).

Μνήμη EEPROM.

Τα διάφορα PLC δεν χρησιμοποιούν μόνο τον παραπάνω τρόπο, της "πάντα τροφοδοτούμενης μνήμης RAM", για να διατηρήσουν το πρόγραμμα στη μνήμη. Ένας πιο ασφαλής τρόπος είναι η χρήση της μνήμης EEPROM (ElectricallyErasableProgrammableReadOnlyMemory), μνήμης η οποία προγραμματίζεται και σβήνει ηλεκτρικά. Πρόκειται για μνήμη που δε σβήνει, όταν μείνει χωρίς τροφοδοσία, στην οποία μπορούμε να γράφουμε, να σβήνουμε και να ξαναγράφουμε μέσω ειδικού μηχανήματος. Σε πολλά PLC η EEPROM χρησιμοποιείται σαν "κασέτα" για την εύκολη αλλαγή του τρόπου λειτουργίας του αυτοματισμού από ένα απλό χειριστή. Δηλαδή έχουμε "γραμμένο" το εναλλακτικό πρόγραμμα σε ένα "τσιπάκι" EEPROMκαι απλά αλλάζουμε την ηλεκτρονική πλακέτα του PLC, όταν θέλουμε να αλλάξουμε το πρόγραμμα λειτουργίας του αυτοματισμού.

Εικόνα 3.14 : EEPROM ολοκληρωμένο

Μνήμη μόνο ανάγνωσης (ROM)

Στη μνήμη ROMο κατασκευαστής του PLC αποθηκεύει το λειτουργικό σύστημα του PLC, δηλαδή τις οδηγίες (το πρόγραμμα) για όλες τις βασικές λειτουργίες που είναι απαραίτητες για να δουλέψει το PLC.

Εξωτερικά σε μια κεντρική μονάδα επεξεργασίας συνήθως υπάρχουν (σχήμα 3):

- Θέση σύνδεσης (ειδικός κονέκτορας) της συσκευής προγραμματισμού.
- Θέση σύνδεσης επεκτάσεων.
- Διακόπτης δύο θέσεων (συνήθως) ο οποίος θέτει το PLC se κατάσταση RUN ή STOP, δηλαδή σε κατάσταση λειτουργίας του αυτοματισμού (RUN) ή όχι (STOP).

Λυχνίες ένδειξης, όπως: λυχνία ένδειξης ότι το PLC είναι σε τροφοδοσία, λυχνία ένδειξης ότι το PLC είναι σε κατάσταση RUN (λειτουργεί ο αυτοματισμός), λυχνία ένδειξης ότι το PLC είναι σε κατάσταση STOP(δεν λειτουργεί ο αυτοματισμός) και λυχνία που δείχνει, εάν έχει πρόβλημα η μπαταρία του PLC.

3.4.4 Μονάδες εισόδων - εξόδων.

Ο μονάδες των εισόδων και των εξόδων αποτελούν τις μονάδες επικοινωνίας της κεντρικής μονάδας με τον έξω κόσμο, δηλαδή με τους αισθητήρες, τους διακόπτες και τα button, που δίνουν τις πληροφορίες (εντολές), καθώς και με τους ηλεκτρονόμους ισχύος των κινητήρων, ηλεκτρομαγνητικές βαλβίδες, ενδεικτικές λυχνίες και γενικά τους αποδέκτες που εκτελούν τις εντολές του αυτοματισμού.

Η κεντρική μονάδα μπορεί να δεχτεί ψηφιακά σήματα εισόδου και εξόδου χαμηλής τάσης και πολύ μικρού ρεύματος. Η τάση που δέχεται η κεντρική μονάδα είναι συνήθως 0 Volt για το λογικό "0" και 5Volt για το λογικό "1". Το ρεύμα εισόδου καθώς και το ρεύμα εξόδου δεν μπορεί να περάσει τα μερικά mA. Οι μονάδες εισόδων και εξόδων αναλαμβάνουν να προσαρμόσουν τα σήματα εισόδου και εξόδου, που έχουμε στον αυτοματισμό, με τα σήματα που μπορεί να δεχτεί η κεντρική μονάδα, τόσο από άποψη τάσεων όσο και από άποψη ρευμάτων.

Η προσαρμογή αυτή γίνεται με τη χρήση ηλεκτρονικών στοιχείων ισχύος, δηλαδή τρανζίστορ ισχύος, θυρίστορ και triac, είτε ακόμη με τη χρήση κατάλληλων μικροηλεκτρονόμων.

Κάθε σύστημα PLC καταλήγει πάντα σε ακροδέκτες (κλέμες). Οι ακροδέκτες αυτοί ανήκουν στις μονάδες εισόδων και εξόδων του PLC. Στους ακροδέκτες εισόδων καταλήγουν οι αγωγοί που έρχονται από αισθητήρες (τερματικούς διακόπτες, πιεζοστάτες, κ.λπ.), διακόπτες, button κ.λπ. Στους ακροδέκτες εξόδων καταλήγουν οι αγωγοί που τροφοδοτούν πηνία ηλεκτρονόμων ισχύος, ηλεκτρομαγνητικές βαλβίδες, λυχνίες ένδειξης και λοιπούς αποδέκτες.

Στους διαφόρους τύπους των PLC οι μονάδες εισόδων και εξόδων αντιμετωπίζονται με διαφορετικό τρόπο. Γενικά ισχύουν τα παρακάτω:

- Μία μονάδα εισόδων ή εξόδων μπορεί να λειτουργεί με συνεχή τάση ή με εναλλασσόμενη τάση. Τυπικές τάσεις που συναντούμε στα PLCείναι: DC 24V, 48V, 60Vκαι AC 24V, 48V, 115V, 230V, με συνηθέστερες τις DC 24Vκαι AC 115Vκαι 230V.

- Η τάση αυτή δεν παρέχεται συνήθως από τη μονάδα τροφοδοσίας του PLC. Πρέπει να τη δημιουργήσουμε εμείς με άλλη τροφοδοτική μονάδα.

- Τα κυκλώματα και οι τάσεις των εισόδων είναι τελείως ανεξάρτητα από τα κυκλώματα και τις τάσεις των εξόδων. Επομένως η τάση για τις εισόδους μπορεί να είναι διαφορετική από την τάση για τις εξόδους. Αν η τάση εξόδων είναι η ίδια με την τάση των εισόδων μπορεί να χρησιμοποιηθεί το ίδιο τροφοδοτικό (για τάσεις DC), ή μετασχηματιστής χειρισμού (για τάσεις AC) για τις εισόδους και τις εξόδους.
- Η τάση εισόδων (δηλαδή η τάση που θα φθάσει σε μια είσοδο, όταν ενεργοποιηθεί ο αντίστοιχος αισθητήρας) διαχωρίζεται συνήθως γαλβανικά από το υπόλοιπο εσωτερικό κύκλωμα του PLC. Τα ίδια ισχύουν και για τις εξόδους. Αν σε κάποιες μονάδες εισόδων ή εξόδων δεν έχουμε γαλβανική απομόνωση πρέπει να προσέξουμε ιδιαίτερα το θέμα των γειώσεων.

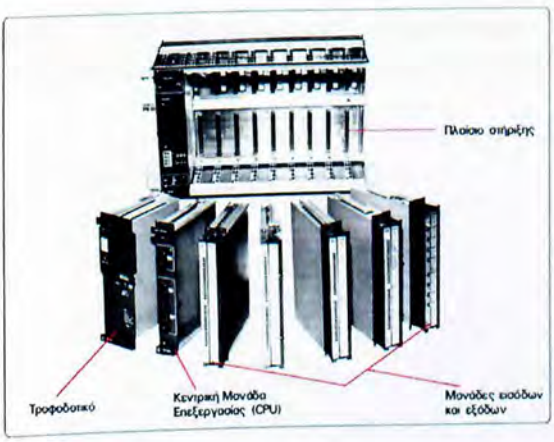
3.5 Οι προγραμματιζόμενοι λογικοί ελεγκτές της αγοράς.

Η κατάσταση που έχει διαμορφωθεί σήμερα στην αγορά από τις εταιρείες, που κατασκευάζουν PLC, είναι η εξής:

Περίπτωση 1 - t a ***ModularPLC****(συνήθως τα μεγάλα PLC)*

Σ' αυτή την περίπτωση το PLC ρωλείται σε modular μορφή, δηλαδή κομμάτι-κομμάτι. Τα βασικά κομμάτια ενός τέτοιου PLCείναι:

- Η μονάδα τροφοδοσίας.
- Η κεντρική μονάδα επεξεργασίας, η οποία έχει τη δυνατότητα να οδηγήσει ένα ανώτατο αριθμό εισόδων και εξόδων. Π.χ. το PLCSIMATICS7-300 (CPU316) της SIEMENS μπορεί να οδηγήσει μέχρι 1024 ψηφιακές εισόδους και εξόδους (συνολικά).
- Οι μονάδες εισόδων και εξόδων. Sta modular PLC πωλούνται και αυτές σε κομμάτια. Κάθε μονάδα εισόδων (ή εξόδων) μπορεί να έχει 4, 8, 16 ή 32 εισόδους (ή εξόδους). Μ' αυτό τον τρόπο μπορούμε να επιλέγουμε μία μονάδα εισόδων ή εξόδων η οποία να έχει τα τεχνικά χαρακτηριστικά που επιθυμούμε. Γίνεται κατανοητό ότι σε ένα modular PLC μπορούμε να έχουμε μονάδες εισόδων ή εξόδων που να λειτουργούν σε διαφορετικές τάσεις τροφοδοσίας

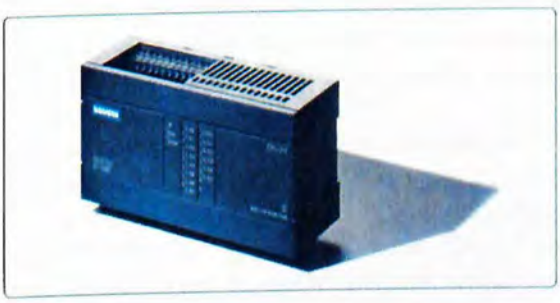


Εικόνα 3.15:ModularPLC (αποτελούνται συνήθως από ανεξάρτητες μονάδες οι όποιες προσαρμόζονται στο πλαίσιο στήριξης)

Περίπτωση 2 - ***Συμπαγή PLC****(μικρά συνήθως)*

- Όλες οι εταιρείες διαθέτουν και μικρά PLC, στα οποία όλες οι μονάδες τους (τροφοδοσίας, κεντρική μονάδα και μονάδες εισόδων - εξόδων) είναι ενσωματωμένες σε μια συσκευή. Σ' αυτού του είδους τα PLC είσοδοι και έξοδοι είναι συνήθως μέχρι 20 και όλες οι είσοδοι (ή έξοδοι) έχουν τα ίδια τεχνικά χαρακτηριστικά.
- Το σημείο που πρέπει κάποιος να προσέξει σχετικά με τις εισόδους και εξόδους είναι ότι κάθε είσοδος ή έξοδος είναι για το PLCακριβώς

καθορισμένη, δηλαδή έχει καθορισμένο όνομα με το οποίο αναφέρεται και στο πρόγραμμα. Στα συμπαγή PLC σε κάθε ακροδέκτη αναγράφεται το όνομα της εισόδου ή της εξόδου. Στα modularPLC υπάρχει σαφές σύστημα με το οποίο αναγνωρίζουμε το όνομα της εισόδου (ή εξόδου) σε κάθε ακροδέκτη μιας μονάδας εισόδων (ή εξόδων)



Εικόνα 3.16 : Συμπαγές PLC (Περιλαμβάνει τροφοδοτικό , κεντρική μονάδα επεξεργασίας, εισόδους και εξόδους , όλα ενσωματωμένα σε μια ενιαία συσκευή)

3.6 Αρχή λειτουργίας ενός προγραμματιζόμενου λογικού ελεγκτή

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

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

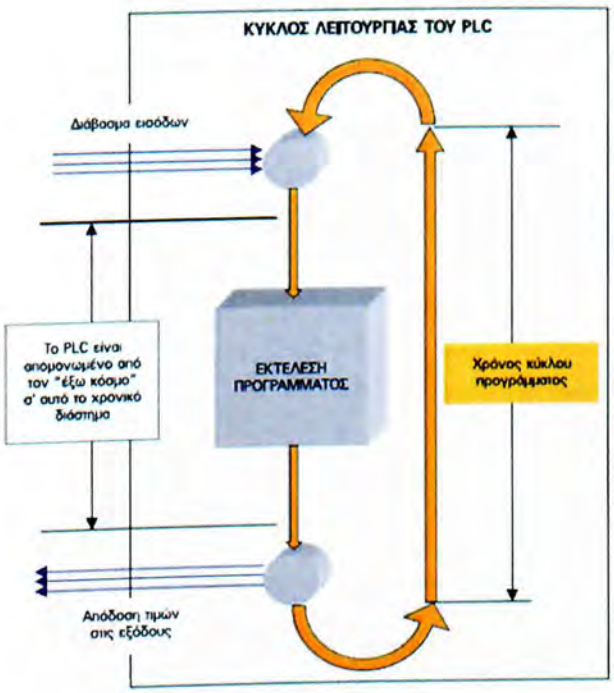
Βήμα 2ο. Στη συνέχεια ο μικροεπεξεργαστής χρησιμοποιώντας σαν δεδομένα τις τιμές των εισόδων, που διάβασε, εκτελεί τις εντολές του προγράμματος, το οποίο λειτουργεί τον αυτοματισμό. Το πρόγραμμα αυτό στην ουσία περιέχει μια σειρά από λογικές πράξεις.

Η εκτέλεση του προγράμματος θα δώσει αποτελέσματα για τις εξόδους. Τα αποτελέσματα αυτά αποθηκεύονται στην ειδική περιοχή της μνήμης που ονομάζεται **εικόνα εξόδων.** Όπως η εικόνα εισόδων, η εικόνα εξόδων περιέχει την τιμή ("0" ή "1") για κάθε έξοδο, π.χ. Q1="1", Q2="1", Q3="0" κ.ο.κ. Σημειώνουμε ότι οι τιμές αυτές προκύπτουν από την εκτέλεση των λογικών πράξεων του προγράμματος.

Βήμα 3ο. Στη συνέχεια ο μικροεπεξεργαστής αποδίδει τις τιμές της εικόνας εξόδων στις εξόδους. Αυτό σημαίνει ότι θα δοθεί "υψηλή" τάση σε όποια έξοδο έχει "1" και θα δοθεί "χαμηλή" τάση σε όποια έξοδο έχει "0".

Με τη συμπλήρωση του 3^{ου} βήματος συμπληρώνεται ένας πλήρης κύκλος λειτουργίας και η διαδικασία ξαναρχίζει από την αρχή. Ο κύκλος λειτουργίας εκτελείται συνεχώς όσο το PLC βρίσκετε σε κατάσταση RUN. Δηλαδή ένα PLCεκτελεί συνεχώς τα βήματα του κύκλου λειτουργίας.

Ο χρόνος που χρειάζεται για να εκτελέσει το PLCένα πλήρη κύκλο λειτουργίας ονομάζεται ***χρόνος κύκλου*** και εξαρτάται από τη "ταχύτητα" του μικροεπεξεργαστή του PLC, αλλά και από τον αριθμό και το είδος των εντολών του προγράμματος. Δηλαδή στο ίδιο PLC για ένα μεγαλύτερο πρόγραμμα έχουμε μεγαλύτερο χρόνο κύκλου. Ο χρόνος κύκλου αποτελεί και ένα μέτρο σύγκρισης μεταξύ των PLC. Για να μπορούν να συγκριθούν τα PLC ως προς τη ταχύτητα εκτέλεσης ενός προγράμματος, ορίζουμε το ***μέσο χρόνο κύκλου,*** σαν το χρόνο κύκλου ενός προγράμματος που περιλαμβάνει 1 Kbyte δυαδικές εντολές. Πάντως στη χειρότερη περίπτωση και σε ένα αργό PLC, ο χρόνος κύκλου δεν ξεπερνά μερικές εκατοντάδες χιλιοστά του δευτερολέπτου. Θα θέλαμε στο σημείο αυτό να τονίσουμε την ουσιαστική διαφορά στη λειτουργία ενός αυτοματισμού με PLC από ένα κλασικό αυτοματισμό με ηλεκτρονόμους.



Εικόνα 3.17 : Κύκλος λειτουργιάς PLC

Στην περίπτωση του κλασικού αυτοματισμού, όταν έχουμε αλλαγή της κατάστασης ενός διακόπτη εισόδου, η αλλαγή αυτή προκαλεί εκείνη τη στιγμή διαδοχικές αλλαγές στα στοιχεία του κυκλώματος που τροφοδοτούνται από το συγκεκριμένο διακόπτη. Έχουμε δηλαδή διαδικασία που συμβαίνει σε πραγματικό χρόνο.

Αν μελετήσουμε τον κύκλο λειτουργία του PLC, θα δούμε ότι το PLC "δεν βλέπει" συνεχώς τον "έξω κόσμο" (την εξωτερική εγκατάσταση), παρά μόνο κατά τα χρονικά διαστήματα που διαβάζει τις εισόδους και αποδίδει τιμές στις εξόδους. Στον υπόλοιπο χρόνο του κύκλου, το PLCείναι ένας υπολογιστής ο οποίος εκτελεί πράξεις (λογικές βέβαια ή και αριθμητικές) απομονωμένο από τον έξω κόσμο. Για να γίνει αυτό κατανοητό υποθέστε ότι αλλάζει η κατάσταση μιας εισόδου, κατά την διάρκεια του χρόνου κατά τον οποίον εκτελούνται οι εντολές προγράμματος. Στην περίπτωση αυτή στο τέλος του κύκλου το PLC θα δώσει αποτελέσματα στις εξόδους, στα οποία δεν θα έχει ληφθεί υπ' όψη η αλλαγή στην κατάσταση της συγκεκριμένης εισόδου. Αυτό γιατί το PLC θα ενημερώσει την εικόνα των εισόδων για την αλλαγή στην κατάσταση της συγκεκριμένης εισόδου στην αρχή του επόμενου κύκλου επεξεργασίας του προγράμματος. Το PLC θα δώσει αποτελέσματα στις εξόδους, στα οποία θα έχει ληφθεί υπ' όψη η αλλαγή στην κατάσταση της παραπάνω εισόδου στο τέλος του επόμενου κύκλου επεξεργασίας του προγράμματος. Λαμβάνοντας τα παραπάνω υπ' όψη θα έλεγε κάποιος ότι τελικά το PLC ανταποκρίνεται πολύ καθυστερημένα στις αλλαγές μιας αυτοματοποιημένης διαδικασίας.

Όμως, αυτό δεν είναι η πραγματικότητα, αφού ο χρόνος πραγματοποίησης ενός κύκλου προγράμματος από ένα PLCείναι πάρα πολύ μικρός, το πολύ 300ms σε πολύπλοκες εγκαταστάσεις αυτοματισμού.

3.7.1 Γλώσσες προγραμματισμού των προγραμματιζόμενων λογικών ελεγκτών.

Τρεις είναι σήμερα οι κυριότερες κατηγορίες γλωσσών προγραμματισμού για PLC, τις οποίες συναντούμε με μικρές διαφορές στα PLCόλων των εταιρειών:

- **Γλώσσα LADDERή γλώσσα ηλεκτρολογικών γραφικών**

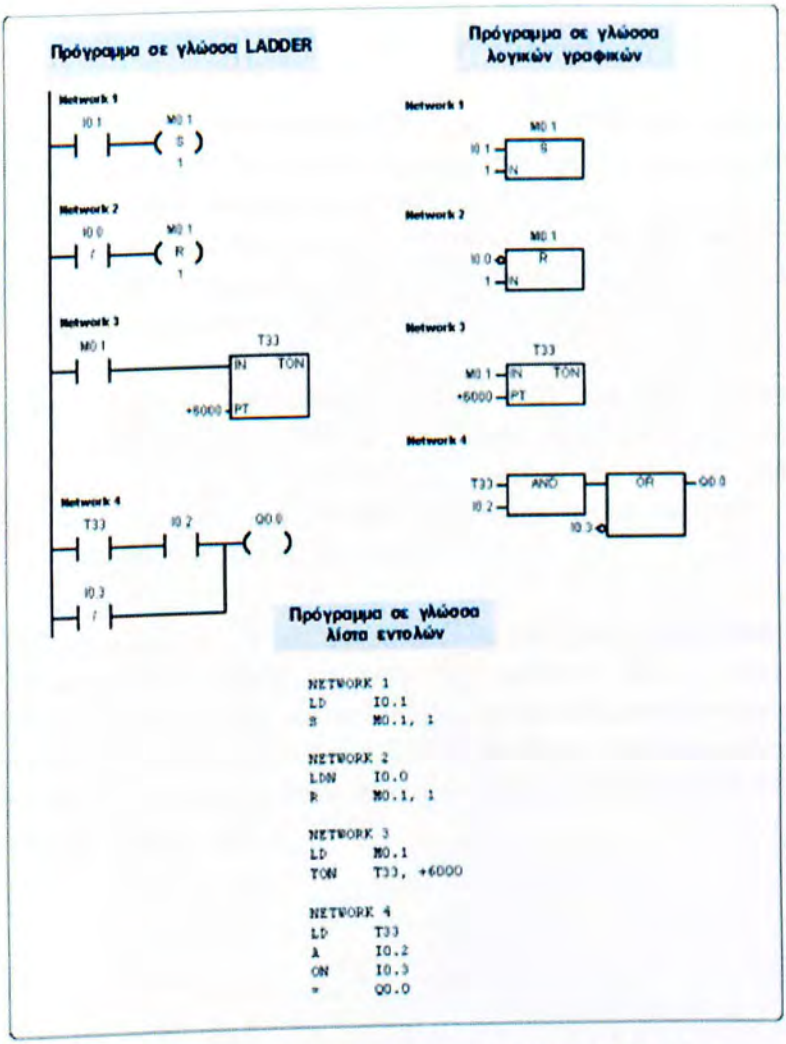
Είναι η πρώτη γλώσσα που αναπτύχθηκε ιστορικά. Ηγλώσσα Ladderστην ουσία επιτρέπει τη μεταφορά του ηλεκτρολογικού σχεδίου, μέσω της συσκευής προγραμματισμού στο PLC.Με τη γλώσσα αυτή η εκπαίδευση των τεχνικών, που ήταν συνηθισμένοι στον κλασικό αυτοματισμό, γινόταν εύκολα και γρήγορα, αφού δεν άλλαζε ουσιαστικά την εργασία σχεδιασμού του αυτοματισμού. Η γλώσσα LADDER χρησιμοποιεί όχι την Ευρωπαϊκή προτυποποίηση στο σχεδιασμό των ηλεκτρικών επαφών, αλλά την Αμερικάνικη. Αυτό ίσως οφείλεται στο γεγονός ότι τα πρώτα PLC αναπτύχθηκαν στην Αμερική. Όμως στη συνέχεια ο τρόπος αυτός σχεδιασμού "βόλεψε" και έτσι διατηρήθηκε και από τις Ευρωπαϊκές εταιρείες, με αποτέλεσμα σήμερα να είναι πλέον καθιερωμένος.

- **Γλώσσα λίστα εντολών (StatementList, STL) ή γλώσσα λογικών εντολών**

Η γλώσσα αυτή αναπτύχθηκε σχεδόν ταυτόχρονα με τη γλώσσα LADDER, αν και οι εταιρείες έδειξαν στην αρχή δισταγμό στο να την "προωθήσουν", φοβούμενες μην τρομάξουν το τεχνικό κατεστημένο της βιομηχανίας. Η γλώσσα αυτή δημιουργεί λίστα προγράμματος με εντολές, οι οποίες αντιστοιχούν στις λογικές πύλες (AND, OR, NOTκ.λπ.). Στην αρχή η γλώσσα λίστα εντολών ήταν πολύ φτωχή και περιοριζόταν μόνο στις βασικές λογικές εντολές, οι οποίες αντιστοιχούσαν αμέσως στις γραφικές εντολές της γλώσσας LADDER. Σήμερα οι γλώσσες αυτές έχουν εξελιχθεί πάρα πολύ και συναντά κανείς σε αυτές στοιχεία από τις γλώσσες των υπολογιστών και κυρίως των γλωσσών Assembly. Οπρογραμματισμός σε λίστα εντολών απαιτεί από τον ηλεκτρολόγο να έχει έστω στοιχειώδεις γνώσεις προγραμματισμού.

- **Γλώσσα λογικών γραφικών ή λογικού διαγράμματος .**

Η γλώσσα αυτή είναι επίσης γραφική, αλλά αντί του ηλεκτρολογικού σχεδίου του αυτοματισμού, χρησιμοποιεί το αντίστοιχο λογικό κύκλωμα. Η γλώσσα αυτή είναι νεότερη και δεν χρησιμοποιείται από όλες τις εταιρείες.



Εικόνα 3.18 : Γλώσσες προγραμματισμού PLC.

4^{Ο} ΚΕΦΑΛΑΙΟ
ΠΡΩΤΟΚΟΛΛΟCANBUS

4.1 Γενικά

Το CANBus (Controller Area Network) είναι ένας τυπικός δίαυλος σχεδιασμένος για να επιτρέπει σε μικρό-ελεγκτές και συσκευές να επικοινωνούν μεταξύ τους εντός του διαύλου χωρίς την χρήση ενός κεντρικού υπολογιστή.
Είναι ένα πρωτόκολλο το οποίο βασίζεται σε μηνύματα, που είχε σχεδιαστεί αρχικά για εφαρμογές αυτοκινητοβιομηχανία, αλλά πλέον χρησιμοποιείται και σε άλλους τομείς όπως βιομηχανικής αυτοματοποίησης και ιατρικού εξοπλισμού.

Ανάπτυξη του CANBus ξεκίνησε αρχικά το 1983 από την Bosch. Το πρωτόκολλο είχε κυκλοφορήσει επίσημα το 1986, στο Συνέδριο Societyof Automotive Engineers (SAE) στο Ντιτρόιτ. Το πρώτο ολοκληρωμένο CANBus κατασκευάστηκε από την Intel και Philips, ήρθε στην αγορά το 1987. Bosch δημοσίευσε την προδιαγραφή CAN 2.0 το 1991.

Το CANBus είναι ένα από τα πέντε πρωτόκολλα που χρησιμοποιούνται για τα διαγνωστικά αυτοκινήτων OBD-II πρότυπο. Το πρότυπο OBD-II έχει γίνει υποχρεωτικό για όλα τα αυτοκίνητα και τα ελαφρά φορτηγά οχήματα που πωλούνται στις Ηνωμένες Πολιτείες από το 1996, και το EOBD πρότυπο είναι υποχρεωτικό για όλα τα οχήματα βενζίνης που πωλούνται στην Ευρωπαϊκή Ένωση από το 2001 και όλα τα ντιζελοκίνητα οχήματα από το 2004.

4.2 Εφαρμογές

Αυτοκινητοβιομηχανία

Ένα σύγχρονο αυτοκίνητο μπορεί να έχει όσο 70 μονάδες ηλεκτρονικού ελέγχου (ECU) για διάφορα υποσυστήματα. Συνήθως μεγαλύτερη επεξεργαστή είναι η μονάδα ελέγχου του κινητήρα ECU ή λειτουργική μονάδα ελέγχου κινητήρα ECM) άλλα χρησιμοποιούνται για την μετάδοση, τους αερόσακους,τ ην antilock πέδηση/ABS, το cruise control, τα ηλεκτρικά συστήματα διεύθυνσης EPS, ta συστήματα ήχου, τα παράθυρα, τις πόρτες κ.λπ. Ορισμένες μονάδες από αυτές αποτελούν ανεξάρτητα υποσυστήματα, αλλά ή επικοινωνία μεταξύ άλλων είναι απαραίτητη. Ένα υποσύστημα μπορεί να χρειαστεί να χειριστεί κάποιους ενεργοποιήτες ελέγχου ή λήψη δεδομένων από αισθητήρες. Το CANBus επινοήθηκε για να καλύψει αυτή την ανάγκη.

Το CANBus μπορεί να χρησιμοποιηθεί σε οχήματα και να συνδέσει τη μονάδα ελέγχου κινητήρα και τη μετάδοση, ή (σε διαφορετικό διαύλου) για να συνδέσει τις κλειδαριές για πόρτες, τονελέγχου κλιματισμού, κλπ. Σήμερα το CANBus χρησιμοποιείται επίσης ως ένα fieldbus (Δίαυλος πεδίου) γενικά σε αυτοματοποίησης περιβάλλοντα, κυρίως λόγω του χαμηλού κόστους.

4.3 Τεχνολογία

Το CANBus είναι ένας multi-master σειριακός διαύλους για τη σύνδεση των μονάδων ηλεκτρονικού ελέγχου (Ecu).
Κάθε κόμβος είναι σε θέση να στείλει και λάβει μηνύματα, αλλά όχι ταυτόχρονα. Ένα μήνυμα που αποτελείται κυρίως από ένα ID (αναγνωριστικό), που αντιπροσωπεύει την προτεραιότητα του μηνύματος, και έως και οκτώ δεδομένων byte. Μεταδίδεται σειριακά σε το δίαυλο. Αυτό το μοτίβο σήμα κωδικοποιείται σε μη-επιστροφή σε μηδενικό (NRZ) και είναι ανιχνεύεται από όλους τους κόμβους.

Οι συσκευές που συνδέονται από ένα CANBus δικτύου είναι συνήθως αισθητήρων, των ενεργοποιητών και άλλες συσκευές ελέγχου. Οι συσκευές αυτές δεν συνδέονται άμεσα με το δίαυλο, αλλά μέσω ενός κεντρικού επεξεργαστή και ενός ελεγκτή CANBus.

Εάν ο δίαυλος είναι ελεύθερος, κάθε κόμβος μπορεί να αρχίσει να μεταδίδει. Αν δύο ή περισσότεροι κόμβοι αρχίσουν αποστολή μηνυμάτων, την ίδια στιγμή, το μήνυμα με το πιο κυρίαρχο Αναγνωριστικό ID (με πιο κυρίαρχη bit, δηλαδή τα μηδενικά) θα αντικαταστήσει λιγότερο κυρίαρχη ID από άλλους κόμβους, ώστε τελικά (μετά από αυτό να είναι το μόνο το κυρίαρχο μήνυμα λαμβάνεται από όλους τους κόμβους. Ο μηχανισμός αυτός αναφέρεται ως προτεραιότητα με βάση διαύλου. Τα μηνύματα με αριθμητικά μικρότερη τιμές αναγνωριστικών ID έχουν υψηλότερη προτεραιότητα και μεταδίδονται πρώτα.

Κάθε κόμβος απαιτεί :

1) Κεντρικός Επεξεργαστής

- Ο κεντρικός επεξεργαστής αποφασίζει τι σημαίνουν τα παραληφθέντα μηνύματα και ποιά μηνύματα θέλει να μεταδώσει ο ίδιος.
- Αισθητήρων, των ενεργοποιητών και των συσκευών ελέγχου που μπορούν να συνδέονται με τον κεντρικό επεξεργαστή.

2) CAN ελεγκτή

- Λήψη: ο ελεγκτής μπορεί να αποθηκεύει τα ληφθέντα bits σειριακά από το δίαυλο μέχρι ένα ολόκληρο το μήνυμα είναι διαθέσιμο, που στη συνέχεια είναι δυνατή η λήψη από το κεντρικού υπολογιστή (συνήθως μετά ο ελεγκτής μπορεί να προκαλέσει μια διακοπή).
- Αποστολή: ο κεντρικός επεξεργαστής αποθηκεύει τα μηνύματα που έλαβε μέσω του ελεγκτή CAN και μεταδίδει τα bit σειριακά στο δίαυλο.

3) Πομποδέκτη

- Λήψη: προσαρμόζει το σήμα από το δίαυλο σε επίπεδα πουο υπεύθυνος της επεξεργασίας CAN ελεκτή μπορεί να αναμένει.
- Αποστολή: μετατρέπει το σήμα μετάδοσης bit που δέχεται από τον ελεγκτή CAN σε ένα μήνυμα που αποστέλλεται στο δίαυλο.

Με ρυθμούς μετάδοσης bit μέχρι 1 Mbit/s είναι δυνατά μήκη δικτύων κάτω από 40 m., μειώνοντας το ρυθμό μετάδοσης bit επιτρέπει μεγαλύτερες αποστάσεις δικτύου (π.χ., 500 m σε 125 kbit/s).

4.4 Μετάδοση δεδομένων

Το CANBus έχει την δυνατότητα αυτόματης μετάδοσης δεδομένων.Ένα Can μήνυμα που μεταδίδεται με υψηλότερη προτεραιότητα ID θα προωθηθεί και ο κόμβος μετάδοσης της χαμηλότερης προτεραιότητας μηνύματος υποχωρήσει και θα περιμένει.

Αυτό επιτυγχάνεται με διαβίβαση δεδομένων μέσω ενός δυαδικού μοντέλου των bits "δεσπόζουσα θέση" και "φυλοσύνδετης" bits όπου η δεσπόζουσα θέση είναι ένα λογικό 0 και φυλοσύνδετης είναι ένα λογικό 1. Αυτό σημαίνει ανοικτού συλλέκτη ή ενσύρματοι ή φυσική υλοποίηση του διαύλου (αλλά από δεσπόζουσα θέση είναι 0 αυτό αναφέρεται μερικές φορές ως ενσύρματου AND). Εάν ένας κόμβος μεταδίδει μια κυρίαρχη bit και έναν άλλο κόμβο μεταδίδει ένα φυλοσύνδετης bit στη συνέχεια το κυρίαρχο bit "κερδίζει" (μια λογική και μεταξύ των δύο).

Έτσι, εάν ένα bit φυλοσύνδετης μεταδίδεται ενώ αποστέλλεται μια κυρίαρχη bit, εμφανίζεται το κυρίαρχο bit, και έχουμε αποδεικτικά στοιχεία της σύγκρουσης. (Δεν είναι ορατές όλες οι άλλες συγκρούσεις.) Διαβεβαιώνεται λίγο δεσπόζουσα θέση δημιουργώντας μια τάση σε ολόκληρη την συρμάτων, ενώ λίγο φυλοσύνδετης δεν έχει απλώς εδραιωθεί στο δίαυλο. Εάν κάθε κόμβος ορίζει μια διαφορά τάσης, όλοι οι κόμβοι θα την ανιχνεύσουν. Επομένως δεν υπάρχει καμία καθυστέρηση στην υψηλότερη προτεραιότητα των μηνυμάτων και τον κόμβο που μεταδίδουν χαμηλότερη προτεραιότητα μηνύματος αυτόματα προσπαθεί να ξαναστείλει έξι κύκλους bit μετά το τέλος του μηνύματος που κατέχει δεσπόζουσα θέση.

Όταν χρησιμοποιείται με μια διαφορετικό δίκτυο, μια έννοια του μεταφορέα πολλαπλές πρόσβασης/bitwise διαιτησίας (CSMA/BA) σύστημα υλοποιείται συχνά: αν δύο ή περισσότερες συσκευές αρχίσουν να διαβιβάζουν την ίδια στιγμή, υπάρχει ένα καθεστώς με βάση διαιτησίας προτεραιότητας να αποφασίσει εκείνη που θα παραχωρηθούν δικαιώματα για να συνεχίσετε τη μετάδοση. Αυτή η λύση μπορεί να είναι προτεραιότητα διαιτησίας (και για την ελεύθερη καθυστέρηση δεσπόζουσα μήνυμα), καθιστώντας το CAN κατάλληλο για πραγματικό χρόνο σε συστήματα επικοινωνιών.

Κατά τη διάρκεια της διαιτησίας, κάθε διαβίβασης κόμβο παρακολουθεί την κατάσταση του διαύλου και συγκρίνει το ληφθέν bit με το bit που μεταδίδονται. Εάν μια κυρίαρχη bit λαμβάνεται όταν λίγο πριν ένα φυλοσύνδετη bit διαβιβαστεί στη συνέχεια στον κόμβο σταματά η μετάδοσης (δηλαδή, αυτό που έχασε διαιτησίας). Διαιτησία πραγματοποιείται κατά τη διάρκεια της μεταφοράς από το πεδίο "Αναγνωριστικό". Σε κάθε κόμβο που αρχίζουν να διαβιβάζουν την ίδια στιγμή στέλνει ένα Αναγνωριστικό με δεσπόζουσα θέση ως δυαδικό 0, ξεκινώντας από το υψηλό bit. Μόλις τους ID είναι ένα μεγαλύτερο αριθμό (χαμηλότερη προτεραιότητα) θα είναι αποστολή 1 (recessive) και θα δείτε 0 (δεσπόζουσα), ώστε να τους

υποχώρησης. Στο τέλος της μετάδοσης ID, όλων των κόμβων αλλά μία κρατήσατε και λαμβάνει την υψηλότερη προτεραιότητα μηνύματος απρόσκοπτη.

Για παράδειγμα, σκεφτείτε μια 11-bit CAN Αναγνωριστικό δικτύου, με δύο κόμβων με αναγνωριστικά των 15 (δυαδική αναπαράσταση, 00000001111) και 16 (δυαδική αναπαράσταση, 00000010000). Εάν αυτοί οι δύο κόμβοι διαβιβάσουν ταυτόχρονα, κάθε ένα θα μεταδίδει τα πρώτα έξι μηδενικά τους αναγνωριστικό με καμία απόφαση διαιτησίας. Όταν το έβδομο bit έχει διαβιβαστεί, του κόμβου με το Αναγνωριστικό των 16 μεταδίδει ένα 1 (φυλοσύνδετης) για το Αναγνωριστικό και τον κόμβο με το Αναγνωριστικό 15 μεταδίδει 0 (δεσπόζουσα) για το αναγνωριστικό. Όταν συμβαίνει αυτό, τον κόμβο με το Αναγνωριστικό των 16 θα συνειδητοποιήσουν ότι έχασε τη διαιτησία, και θα επιτρέψει τον κόμβο με Αναγνωριστικό 15, για να συνεχίσετε τη μετάδοση. Αυτό εξασφαλίζει ότι ο κόμβος με τη χαμηλότερη τιμή bit πάντα θα κερδίσει η διαιτησία. Το Αναγνωριστικό με το μικρότερο αριθμό θα κερδίσει το δικαίωμα χρήσης.

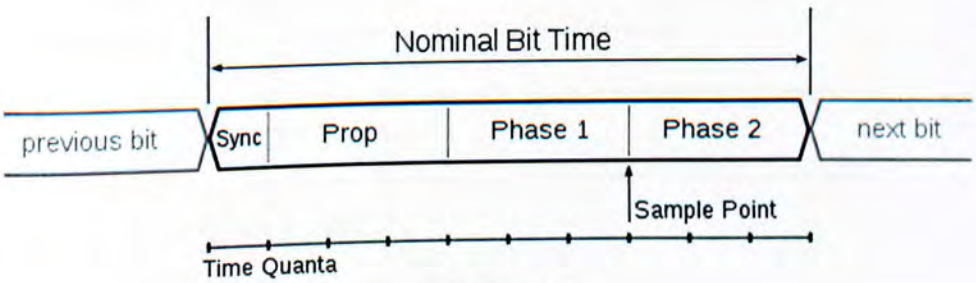
4.5 Αναγνωριστικό ID

Τα αναγνωριστικά ID πρέπει να είναι μοναδικό σε ένα δίαυλο, διαφορετικά δύο κόμβοι με κοινόID θα συνεχίσουν τη μετάδοση πέρα από το τέλος του πεδίου διαιτησίας (ID) και θα προκληθεί σφάλμα.

Στις αρχές της δεκαετίας του 1990, έγινε η επιλογή των αναγνωριστικών για μηνύματα απλώς με βάση τονπροσδιορισμό του τύπου δεδομένων και τον κόμβο αποστολής, Ωστόσο, όπως το Αναγνωριστικό χρησιμοποιείται ως την προτεραιότητα του μηνύματος, αυτό οδήγησε σε κακή επίδοση σε πραγματικό χρόνο. Σε αυτές τις περιπτώσεις, το CANBUS ένα χαμηλής χρήσης, περίπου 30% ήταν συνήθως απαιτούνται για να διασφαλιστεί ότι όλα τα μηνύματα του θα πληρούν τις προθεσμίες. Ωστόσο, εάν τα αναγνωριστικά αντίθετα καθορίζονται με βάση την προθεσμία του μηνύματος, το χαμηλότερο αριθμητικό Αναγνωριστικό και ως εκ τούτου υψηλότερη προτεραιότητα μηνύματος, στη συνέχεια διαύλου το 70 έως 80% μπορεί συνήθως να επιτευχθεί πριν να χαθεί από κάθε μήνυμα η προθεσμία μετάδοσης του .

4.6 Το bit χρονισμού

Κάθε κόμβος δικτύου CANBUS έχει το δικό του ρολόι χρονισμού , και το ρολόι δεν αποστέλλεται κατά τη μετάδοση δεδομένων. Γίνεται συγχρονισμός διαιρώντας κάθε bit του πλαισίου σε αριθμό τμημάτων: συγχρονισμού, διάδοσης, η φάση 1 και η φάση 2. Το μήκος του κάθε τμήματος μπορεί να προσαρμόζεται με βάση το χρονισμό του δικτύου και του κόμβου. Το σημείο δειγματοληψίας βρίσκεται μεταξύ φάση Buffer τμήμα 1 και 2 , το οποίο διευκολύνει τη συνεχή συγχρονισμό. Συνεχή συγχρονισμό επιτρέπει, με τη σειρά του, ο δέκτης να είναι σε θέση να διαβάσει σωστά τα μηνύματα.



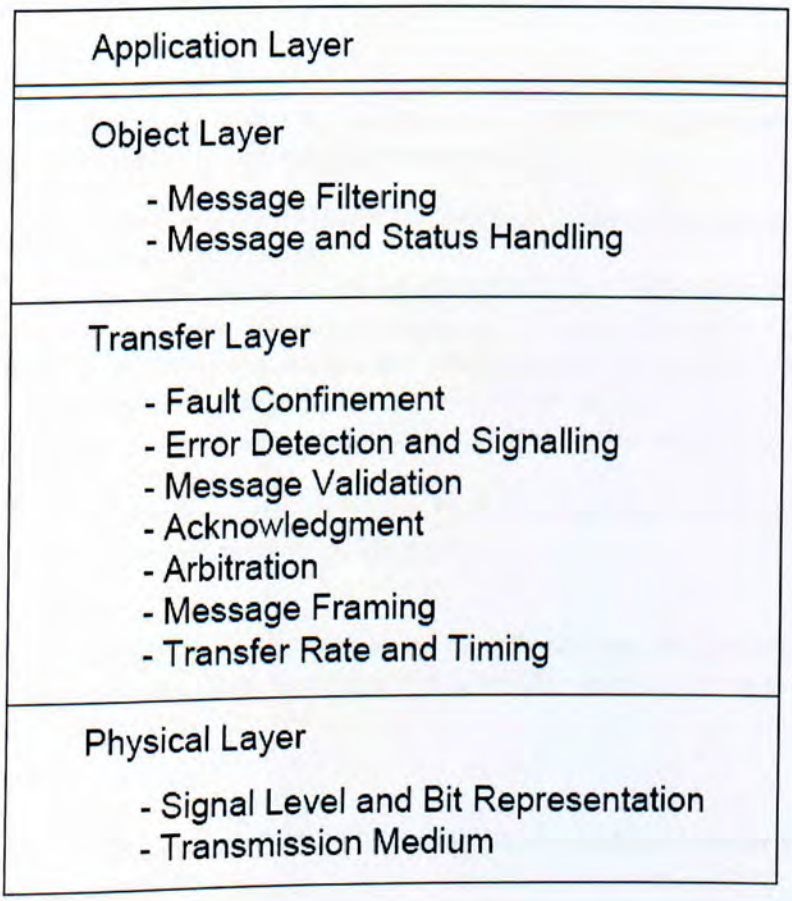
Εικόνα 4.1

4.7 Κύρια Χαρακτηριστικά

Τα κύρια χαρακτηριστικά του πρωτοκόλλου CAN είναι:

- Προτεραιότητα στα μηνύματα
- Εξασφαλισμένοι χρόνοι καθυστέρησης
- Ευελιξία στην παραμετροποίηση
- Δυνατότητα πολλαπλών παραληπτών μηνυμάτων με δυνατότητα συγχρονισμού
- Multimaster
- Ανίχνευση λαθών
- Αυτόματη επαναποστολή κατεστραμμένων μηνυμάτων μόλις ο διάδρομος ελευθερωθεί
- Διάκριση μεταξύ προσωρινών λαθών και μόνιμων αποτυχιών κόμβων και αυτόματη απομάκρυνση αυτών

Τα επίπεδα του πρωτοκόλλου CAN παρουσιάζονται παρακάτω:



Εικόνα 4.2 : CanLayers

Το φυσικό επίπεδο (PhysicalLayer) καθορίζει πως μεταδίδονται τα σήματα στην πραγματικότητα.

Το επίπεδο Μεταφοράς (TransferLayer) αποτελεί τον πυρήνα του πρωτοκόλλου CAN. Παρουσιάζει τα μηνύματα που λαμβάνονται από το επίπεδο αντικειμένου (ObjectLayer) και δέχεται να μεταδοθούν μηνύματα από αυτό. Το επίπεδο Μεταφοράς είναι υπεύθυνο για χρονισμό των bit, το διαχωρισμό των μηνυμάτων, τη διαιτησία του διαδρόμου, την επαλήθευση των μηνυμάτων και τον εντοπισμό λαθών.

Το επίπεδο αντικειμένου ασχολείται με το φιλτράρισμα των μηνυμάτων καθώς και τη διαχείρισή τους.

Να δούμε τώρα μερικά σημαντικά σημεία στα οποία πλεονεκτεί το πρωτόκολλο CAN.

ΜΗΝΥΜΑΤΑ: Μπορεί να στείλει μηνύματα μεγέθους μέχρι 8 byte. Μπορεί να αποστείλει και πακέτο με μηδενικό μήκος δεδομένων (όταν παραδείγματος χάριν ζητάει κάποιος κόμβος πληροφορίες από έναν άλλο).

ΔΡΟΜΟΛΟΓΗΣΗ: Κάθε κόμβος δεν ασχολείται με το πως είναι σχεδιασμένο το σύστημα (τρόπο σύνδεσης κόμβων). Αυτό έχει ως συνέπεια τα εξής:

- *Ευελιξία του συστήματος*: μπορούν να εισαχθούν κόμβοι σε ένα δίκτυο CAN χωρίς αλλαγή υλικού ή λογισμικού.
- *Δρομολόγηση μηνυμάτων*: το κάθε μήνυμα αναγνωρίζεται από το πεδίο IDENTIFIER. Αυτό το πεδίο δεν δείχνει τον προορισμό του μηνύματος αλλά περιγράφει τη σημασία των δεδομένων που έχει μέσα του έτσι ώστε όλοι οι κόμβοι να είναι σε θέση να αποφασίσουν αν θα κρατήσουν ή όχι το μήνυμα (MESSAGEFILTERING).
- *Multicasting*: αποτέλεσμα του παραπάνω πολλοί κόμβοι μπορούν ταυτόχρονα να λάβουν το ίδιο μήνυμα.
- *Συνέπεια δεδομένων*: μέσα στο δίκτυο CAN είναι εγγυημένο ότι μπορεί να ληφθεί και από όλους τους κόμβους και από κανένα.

ΤΑΧΥΤΗΤΑ ΔΙΑΥΛΟΥ: Είναι δεδομένη για το κάθε σύστημα και μέγιστη το 1Mbps. Συνήθως στα σύγχρονα μέσα μεταφοράς υπάρχουν δύο δίκτυα. Ένα υψηλής και ένα χαμηλότερης ταχύτητας.

ΠΡΟΤΕΡΑΙΟΤΗΤΕΣ: Το πεδίο IDENTIFIER καθορίζει στατική προτεραιότητα στο δίαυλο.

MULTIMASTER: Όταν ο δίαυλος είναι ελεύθερος, οποιοσδήποτε κόμβος μπορεί να ξεκινήσει να αποστέλλει μηνύματα. Αυτό με τη μεγαλύτερη προτεραιότητα, θα αποσταλεί τελικά.

ΔΙΑΙΤΗΣΙΑ: Όταν ο δίαυλος είναι ελεύθερος, οποιοσδήποτε κόμβος μπορεί να ξεκινήσει να αποστέλλει μηνύματα. Αν δύο ή παραπάνω κόμβοι ξεκινήσουν να αποστέλλουν ταυτόχρονα μηνύματα, έχουμε σύγκρουση στο δίαυλο η οποία επιλύεται με διαιτησία bit προς bit χρησιμοποιώντας το πεδίο IDENTIFIER. Έτσι, ούτε χρόνος αλλά ούτε και μηνύματα χάνονται.

ΑΣΦΑΛΕΙΑ: Το δίκτυο CAN για τη διασφάλιση των μηνυμάτων έχει συστήματα εντοπισμού σφαλμάτων τα οποία μπορούν να ανιχνεύσουν:

- Όλα τα συνολικά λάθη
- Όλα τα τοπικά λάθη εντοπίζονται στους αποστολείς
- Μέχρι 5 τυχαία λάθη σε ένα μήνυμα εντοπίζονται
- Λάθη υπερχείλισης με μήκος λιγότερο από 15 σε ένα μήνυμα
- Λάθη σε μονό πλήθος σε ένα μήνυμα

ΣΥΝΔΕΣΕΙΣ: Το CAN είναι σειριακό πρωτόκολλο επικοινωνίας όπου ένας αριθμός κόμβων μπορεί να συνδεθεί. Αυτός ο αριθμός είναι θεωρητικά απεριόριστος, πρακτικά όμως περιορίζεται από τις καθυστερήσεις και/ή την επιφόρτιση της γραμμής.

ΜΟΝΟΚΑΝΑΛΗ: Ο δίαυλος αποτελείται από ένα κανάλι και μόνο. Έτσι μπορεί να γίνει εύκολα επανασυγχρονισμός των δεδομένων.

4.8 Πλαίσιο (FRAMES)

ΈναCAN δίκτυο μπορεί να ρυθμιστεί να λειτουργεί με δύο διαφορετικές μορφές μηνυμάτων (ή "πλαίσιο"): το πρότυπο ή βάσης πλαισίου μορφή (ή CAN 2.0 A), και το εκτεταμένο πλαίσιο μορφή (ή CAN 2.0 B). Η μόνη διαφορά μεταξύ των δύο μορφών είναι ότι το "CAN βασικό FRAME" υποστηρίζει μήκος 11 bits για το αναγνωριστικό και το "εκτεταμένο πλαίσιο μπορεί να" υποστηρίζει το μήκος των 29 bits για το αναγνωριστικό, που αποτελείται από το αναγνωριστικό 11 bit ("Αναγνωριστικό βάσης") και μια επέκταση 18 bit "(επέκταση Αναγνωριστικό"). Η διάκριση μεταξύ τους γίνεται με τη χρήση του IDE bit, που μεταδίδεται ως κατέχει δεσπόζουσα θέση σε περίπτωση ένα πλαίσιο 11-bit, και διαβιβάζονται ως φυλοσύνδετης στην περίπτωση ενός πλαισίου 29-bit. Οι CAN ελεγκτές, που υποστηρίζουν εκτεταμένης πλαισίου μορφή μηνυμάτων έχουν επίσης τη δυνατότητα να στέλνουν και να λαμβάνουν μηνύματα βασικούπλαισίου. Όλα τα πλαίσια που ξεκινούν με ένα bit (SOFέναρξη πλαισίου) που χαρακτηρίζει την έναρξη της μετάδοσης πλαισίων.

Το πρωτόκολλο CAN υποστηρίζει 4 ειδών διαφορετικά μηνύματα:

- *DATAFRAME*: είναι το κύριο είδος μηνύματος με τα δεδομένα.
- *REMOTEFRAME*: μεταδίδεται από ένα κόμβο για να ζητήσει τα συγκεκριμένα δεδομένα με το ίδιο IDENTIFIER.
- *ERRORFRAME*: μεταδίδεται από ένα κόμβο για να δηλώσει σφάλμα στο δίαυλο
- *OVERLOADFRAME*: μεταδίδεται για επιπλέον καθυστέρηση ανάμεσα στα διαδοχικά DATA και REMOTEFRAMES.

Τα DATA και REMOTEFRAMES διαχωρίζονται από τα επόμενα μηνύματα από ένα ενδομηνυματικό κενό (INTERFRAMESPACE).

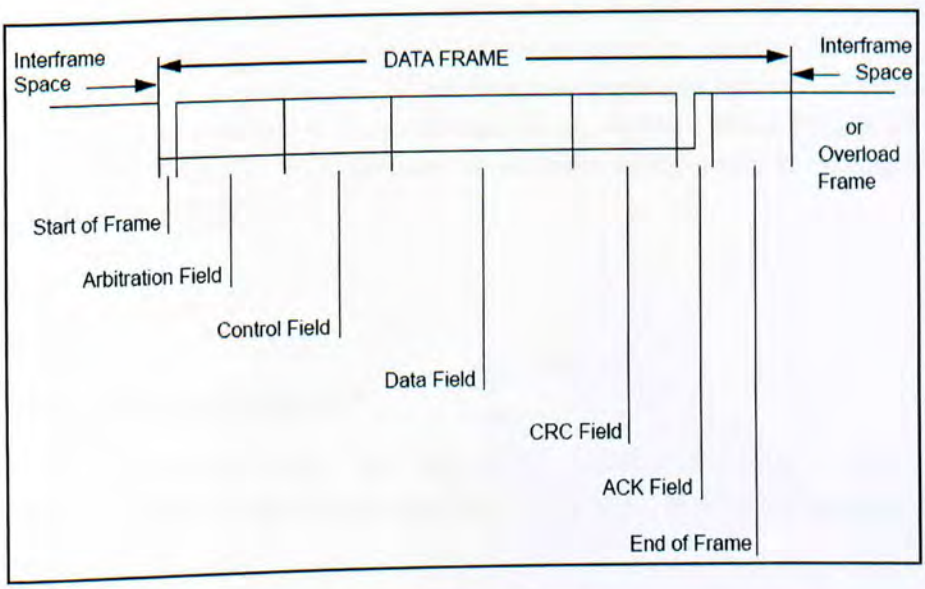
Αναλυτικότερα η δομή κάθε μηνύματος είναι:

4.8.1 DATAFRAME

Το DATAFRAME χωρίζεται σε επτά τμήματα. Αυτά τα τμήματα κατά σειρά είναι:

- Start of Frame
- Arbitration Field
- Control Field
- Data Field
- CRC Field
- ACK Field
- End of Frame

Παρακάτω παρουσιάζεται σχηματικά η μορφή του DATAFRAME



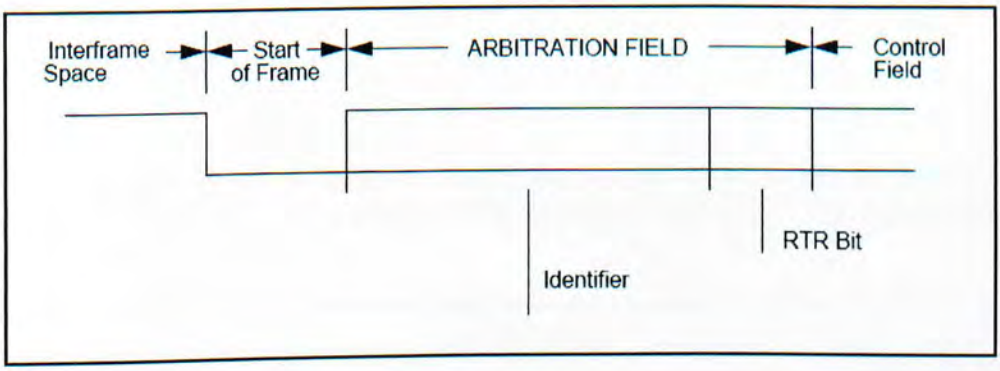
Εικόνα 4.3 : DATA FRAME

START OF FRAME

Σηματοδοτεί την έναρξη του DATA ή REMOTEFRAME. Αποτελείται από ένα και μόνο 'dominant' bit.

ARBITRATIONFIELD

Αποτελείται από το IDENTIFIER και το RTR-BIT όπως φαίνεται παρακάτω σχηματικά:



Εικόνα 4.4 :Arbitration Field

IDENTIFIER

Το IDENTIFIER έχειμήκος 11 bit γιατοτιςπροδιαγραφέςστο CAN 1.2 (standard frame format) και 29 bit για πλέον με τις προδιαγραφές CAN 2.0 (extended frame format). Εμείς στην προσομοίωση χρησιμοποιήσαμε το standard frame format μιας και δεν είχαμε το μεγάλο πλήθος κόμβων που έρχεται να καλύψει το extended frame format. Για αυτό και θα ασχοληθούμε με το standard frame format και θα κάνουμε απλώς ορισμένες παρατηρήσεις για τις τυχόν αλλαγές που έχουν προστεθεί με την έλευση του CAN 2.0.

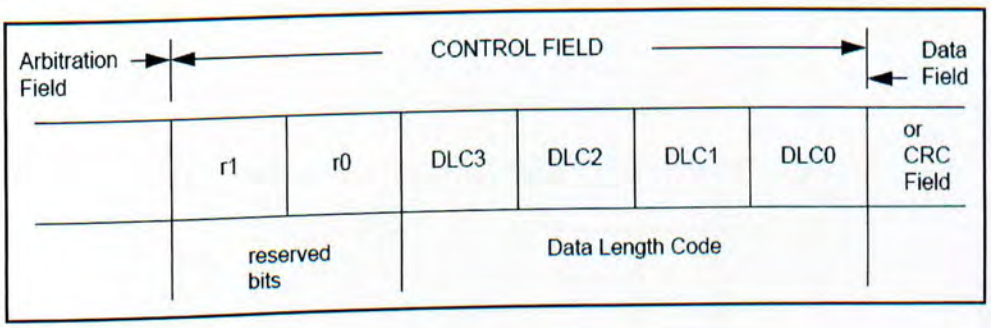
RTR-BIT

Remote Transmission Request BIT

Sta DATAFRAMES το bit αυτό είναι 'dominant' ενώ στα REMOTEFRAMES πρέπει να είναι 'recessive' Είναι το bit με το οποίο ξεχωρίζουν οι δύο μορφές frame.

CONTROL FIELD

Έχειμήκοςέξι bit. Τα δύο πρώτα είναι δεσμευμένα για μελλοντική χρήση και τα άλλα τέσσερα αποτελούν το DATALENGTHCODE (μήκος μηνύματος). Τα δεσμευμένα είναι πάντα 'dominant'.



Εικόνα 4.5: Control Field

DATALENGTHCODE

Δείχνουν το μήκος του μηνύματος όπως φαίνεται στον παρακάτω πίνακα:

Number of Data Bytes	Data Length Code			
	DLC3	DLC2	DLC1	DLC0
0	d	d	d	d
1	d	d	d	r
2	d	d	r	d
3	d	d	r	r
4	d	r	d	d
5	d	r	d	r
6	d	r	r	d
7	d	r	r	r
8	r	d	d	d

Εικόνα 4.6 : Data length

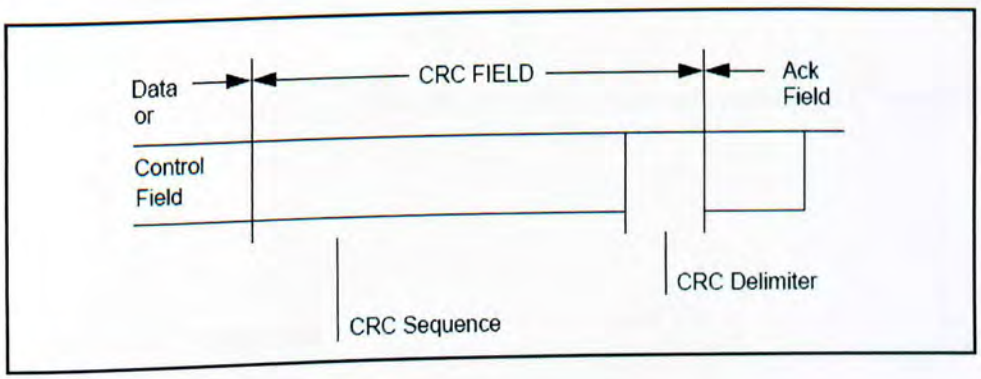
όπου d για 'dominant' και r για 'recessive'. Όλοι οι άλλοι συνδυασμοί δεν είναι δεκτοί

DATAFIELD

Περιέχει τα προς αποστολή δεδομένα. Μπορεί να περιέχει από 0 έως 8 bytes δεδομένων.

CRCFIELD

Περιέχει το CRCSEQUENCE και το CRCDELIMITER όπως παρουσιάζεται στο παρακάτω σχήμα:



Εικόνα4.7 : CRC Field

CRCSEQUENCE

Αποτελεί το τμήμα ελέγχου της ορθότητας του μηνύματος. Προκειμένου να γίνει ο CRC υπολογισμός έχουμε τη διαίρεση των προηγούμενων τμημάτων του μηνύματος με το πολυώνυμο:

$$X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1$$

Το υπόλοιπο της διαίρεσης μπαίνει στο CRCSEQUENCE.

CRCDELIMITER

Αποτελείται από ένα και μόνο 'recessive' bit.

ACKFIELD

Το ACKFIELD αποτελείται από δύο bit, το ACKSLOT και το ACKDELIMETER. Ο αποστολέας αποστέλλει δύο 'recessive' bits στο ACKFIELD. Ο παραλήπτης στο ACKSLOT αποστέλλει 'dominant' bit.

END OF FRAME

Κάθε DATA FRAME και REMOTE FRAME τελειώνειμεεπτάδιαδοχικά 'recessive' bits.

4.8.2 REMOTEFRAME

Το REMOTEFRAME είναι ακριβώς ίδιο με το DATAFRAME. Έχει τα ίδια πεδία, τα οποία συμπληρώνονται και με τις ίδιες τιμές. Το μόνο που αλλάζει είναι το RTR-BIT του ARBITRATIONFIELD που είναι 'recessive' ενώ στο DATAFRAME είναι 'dominant'.

4.8.3 ERRORFRAME

Υπάρχουν τριών ειδών καταστάσεις στις οποίες μπορεί να βρεθεί ένας κόμβος όταν εντοπιστεί κάποιο σφάλμα. Οι καταστάσεις αυτές είναι:

Error-Active

Ένας Error-Active κόμβος μπορεί να πάρει ενεργά μέρος στην επικοινωνία διαύλου, εμπεριέχοντας σε αυτή τη λειτουργία ένα ενεργό error flag, που αποτελείται από έξι διαδοχικά dominantbits. Το error flag ενεργά παραβιάζει τον κανόνα για το bit stuffing και αναγκάζει όλους τους άλλους κόμβους να στείλουν ένα ErrorFlag, επονομαζόμενο the Error Echo Flag, προς απάντηση. Ένα ενεργό Active Error Flag και το επακόλουθο Error Echo Flag μπορούν να προκαλέσουν τουλάχιστον δώδεκα διαδοχικά dominantbits στον δίαυλο: έξι από το Active Error Flag και από μηδέν μέχρι έξι από το Error Echo Flag, γεγονός που εξαρτάται από το λάθος που ανιχνεύεται η όχι στον δίαυλο από τον κόμβο. Ένας κόμβος είναι Error-Active όταν και το Transmit Error Counter (TEC) και το Receive Error Counter (REC) είναι κάτω από 128. Error-Active είναι ένας τρόπος λειτουργίας που επιτρέπει στον κόμβο είτε να λαμβάνει, είτε να εκπέμπει, χωρίς περιορισμούς.

Error-Passive

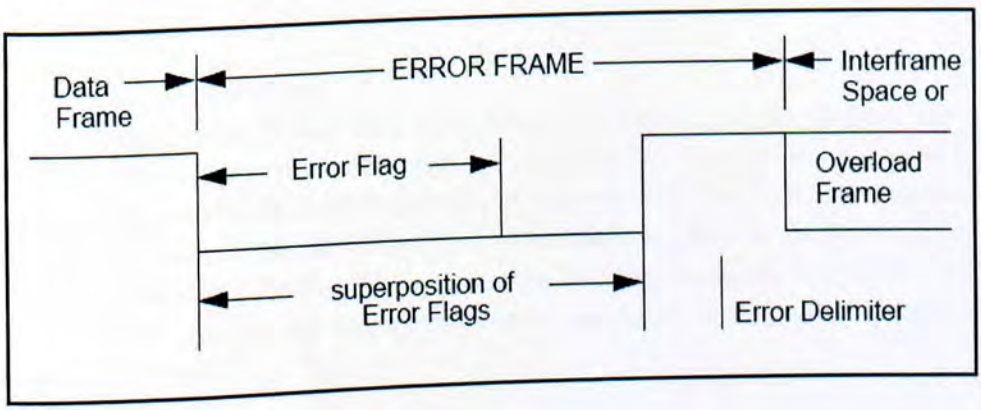
Ένας κόμβος γίνεται Error-Passive όταν, είτε ο Transmit Error Counter (μετρητής), είτε ο Receive Error Counter υπερβεί το 127.Οι Error-Passive κόμβοι δεν επιτρέπεται να μεταδώσουν Active Error Flags σε δίαυλο, αλλά αντίθετα μεταδίδουν Passive Error Flags που αποτελούνται από έξι διαδοχικά bits. Αν ο Error-Passive κόμβος είναι ο τρέχων αποστολέας στον δίαυλο τότε το (παθητικό) passive error flag θα παραβιάζει τον κανόνα του bit stuffing (γέμισμα bit) και οι κόμβοι-παραλήπτες θα απαντήσουν με δικά τους Error Flags (είτε ενεργητικά είτε παθητικά, τα οποία εξαρτώνται από την κατάσταση error (λάθους) στην οποία αυτοί βρίσκονται). Αν ο Error-Passive κόμβος στην περίπτωσή μας δεν είναι ο μόνος αποστολέας ή ο παραλήπτης, τότε το Passive Error Flag δεν θα έχει καμία επίδραση στο δίαυλο εξαιτίας της διαδοχικής φύσης του error flag. Όταν ένας Error-Passive κόμβος μετ αδίδει ένα Passive Error Flag και εντοπίζει ένα dominant bit, πρέπει να θεωρεί τον δίαυλο σαν ανενεργό για οχτώ επιπλέον bit-times (χρόνους), μετά από μία διακοπή και πριν αναγνωρίσει τον δίαυλο σαν διαθέσιμο. Μετά από εκείνη την στιγμή θα

προσπαθήσει να μεταδώσει ξανά.

Bus-Off

Ένας κόμβος εισέρχεται σε Bus-Off κατάσταση όταν ο Transmit Error Counter (μετρητής) είναι μεγαλύτερος από 255 (λάθη λήψης δεν μπορούν να κάνουν ένα κόμβο να εισέλθει σε Bus-Off).Σε αυτή την κατάσταση, ο κόμβος δεν μπορεί ούτε να λάβει , ούτε να στείλει μηνύματα, acknowledge μηνύματα (επιβεβαίωσης), ή να μεταδώσει οποιουδήποτε είδους Error Frames. Έτσι δηλαδή, επιτυγχάνεται ο Fault Confinement (λανθασμένος περιορισμός). Υπάρχει μία διαδικασία επαναλειτουργίας του διαύλου που ορίζεται από το CAN πρωτόκολλο και επιτρέπει στον κόμβο που βρίσκεται σε Bus-Off να επανέλθει, να επιστρέψει δηλαδή σε Error-Active και να αρχίσει να μεταδίδει ξανά εφόσον ο λανθασμένος όρος απομακρυνθεί.

Το ERRORFRAME έχει δύο πεδία. Το πρώτο είναι το ERRORFLAG και το δεύτερο το ERRORDELIMITER όπως παρουσιάζεται στο παρακάτω σχήμα:



Εικόνα 4.8 : Error Frame

Error Flag

Υπάρχουν δύο ειδών ERROR FLAG.

1. ActiveERRORFLAG που αποτελείται από έξι διαδοχικά 'dominant' bits
2. PassiveERRORFLAG που αποτελείται από έξι διαδοχικά 'recessive' bits

Ένας ενεργός σταθμός αποστέλλει το πρώτο ενώ ένας παθητικός το δεύτερο. Αποστέλλοντας έξι διαδοχικά 'dominant' bits παραβιάζεται ο νόμος του bitstuffing

(κάθε πέντε διαδοχικά 'dominant' bits τοποθετείται ένα 'recessive') με αποτέλεσμα να γίνεται αντιληπτό το σφάλμα από όλους τους σταθμούς.

ERRORDELIMITER

Αποτελείται από οχτώ διαδοχικά 'recessive' bits.

Τα σφάλματα που μπορούν να εντοπιστούν μέσα σε ένα μήνυμα του δικτύου CAN είναι τα παρακάτω:

- **CRC Error**

Η τιμή ενός 15-bit Cyclic Redundancy Check (CRC) υπολογίζεται από τον κόμβο που εκπέμπει και αυτή μεταδίδεται στο CRC πεδίο. Όλοι οι κόμβοι του δικτύου που λαμβάνουν αυτό το μήνυμα, υπολογίζουν ένα CRC και επιβεβαιώνουν ότι η τιμή του CRC ταιριάζει. Αν οι τιμές δεν ταιριάζουν, ένα CRCerror συμβαίνει και ένα ERRORFRAME παράγεται. Εάν και εφόσον ένας κόμβος τουλάχιστον δεν λάβει κανονικά το μήνυμα, αυτό απορρίπτεται μετά από τον κανονικό χρόνο που μεσολαβεί ανάμεσα στη μετάδοση των πακέτων.

- **Acknowledge Error**

Στο Acknowledge FIELD ενός μηνύματος ο transmitting κόμβος ελέγχει εάν το Acknowledge Slot (το οποίο εστάλησαν ένα recessive bit) περιέχει ένα dominant bit. Αυτό το dominant bit θα αναγνώριζε εάν τουλάχιστον ένας κόμβος έλαβε σωστά το μήνυμα. Εάν αυτό το bit είναι recessive, τότε κανείς κόμβος δεν έλαβε το μήνυμα κανονικά. Τότε ένα Acknowledge Error συνέβη.Συνεπώς παράγεται ένα Error Frame και το αρχικό μήνυμα θα επαναληφθεί ,μετά το χρόνο διαλείμματος μετάδοσης μεταξύ των πακέτων.

- **Form Error**

Εάν ένας κόμβος εντοπίσει ένα dominant bit σε κάποιο από τα παρακάτω τέσσερα τμήματα του μηνύματος: End of Frame, Inter frame Space, Acknowledge Delimiter or CRC Delimiter, το CAN πρωτόκολλο ορίζει αυτό σαν ένα τύπο παραβίασης και ένα Form Error παράγεται. Το αρχικό μήνυμα απορρίπτεται μετά από τον κανονικό χρόνο που μεσολαβεί ανάμεσα στην μετάδοση των πακέτων μηνυμάτων.

- **BitError**

Ένα Bit Error συμβαίνει εάν ο αποστολέας στείλει ένα dominantbit και εντοπίσει ένα recessivebit, ή εάν αυτός στείλει ένα recessivebit και εντοπίσει ένα dominantbit όταν παρακολουθεί το πραγματικό buslevel και συγκρίνει αυτό με το bit που εστάλη. Σε περίπτωση που ο αποστολέας στείλει ένα recessivebit και ένα dominantbit εντοπιστεί κατά το Arbitration Field η το Acknowledge Slot κανένα BitError δεν παράγεται, διότι κανονική διαιτησία η επιβεβαίωση συμβαίνει. Εάν ένα BitError εντοπιστεί, παράγεται ένα ErrorFrame και το αρχικό μήνυμα απορρίπτεται μετά από τον κανονικό χρόνο που μεσολαβεί ανάμεσα στην μετάδοση των πακέτων μηνυμάτων.

- **Stuff Error**

Το CAN πρωτόκολλο χρησιμοποιεί μία NonReturntoZero (NRZ) μέθοδο μετάδοσης. Αυτό σημαίνει ότι το bit level μένει στο δίαυλο για ολόκληρο το bit-χρόνο. Το CAN είναι επίσης ασύγχρονο, και το bit stuffing χρησιμοποιείται για να επιτρέψει σε κόμβους- παραλήπτες να συγχρονιστούν παίρνοντας πληροφορίες μέσω ενός ρολογιού από το data stream.

Οι κόμβοι παραλήπτες συγχρονίζονται από recessive s e dominant μεταδόσεις. Εάν υπάρχουν παραπάνω των πέντε bit της ίδιας πολικότητας σε μία σειρά ,το CAN αυτόματα βάζει ένα αντίθετης πολικότητας bit στο datastream.Οι κόμβοι παραλήπτες θα το χρησιμοποιήσουν για συγχρονισμό , αλλά θα αγνοήσουν το bit γεμίσματος για σκοπό δεδομένων (data). Αν ανάμεσα στο Start of Frame και το CRC Delimiter, έξι διαδοχικά bit με την ίδια πολικότητα εντοπιστούν , τότε ο κανόνας για το (γέμισμα των bit) bit- stuffing παραβιάζεται. Τελικά συμβαίνει ένα Stuff Error, ένα ErrorFrame στέλνεται και το μήνυμα επαναλαμβάνεται.

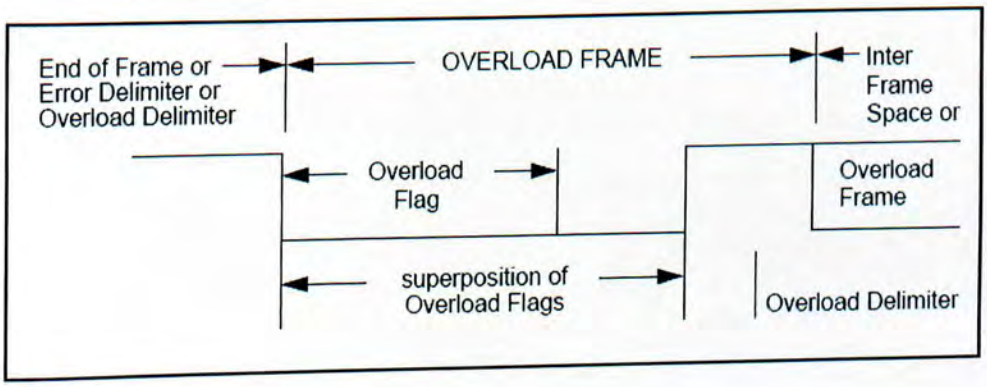
4.8.4 OVERLOATFRAME

Αποτελείται από δύο πεδία. ToOVERLOADFLAGκαιτοOVERLOADDELIMITER.

Υπάρχουν δύο ειδών καταστάσεις OVERLOAD όπου και στις δύο αποστέλλεται OVERLOADFRAME

1. Ο δέκτης απαιτεί μία καθυστέρηση για το επόμενο DATA ή REMOTEFRAME.
2. Ανίχνευση 'dominant' bit κατά τη διάρκεια του INTERMISSION.

Το πολύ μέχρι δύο OVERLOADFRAMES μπορούν να δημιουργηθούν πριν από το επόμενο DATA ή REMOTEFRAME. Παρακάτω παρουσιάζεται η μορφή του:



Εικόνα 4.9 : Overload Frame

OVERLOAD FLAG

Αποτελείταιαπόέξι 'dominant' bits όπωςτο Active ERROR FLAG.

OVERLOAD DELIMITER

Αποτελείταιαπόοχτώ 'recessive' bits όπωςκαιο ERROR DELIMITER.

INTERMISSION

Αποτελείταιαπότρία 'recessive' bits. Κατά τη διάρκεια του INTERMISSION κανένας κόμβος δεν μπορεί να ξεκινήσει την αποστολή DATA ή REMOTEFRAME. Μόνο OVERLOADFRAMES.

BUSIDLE

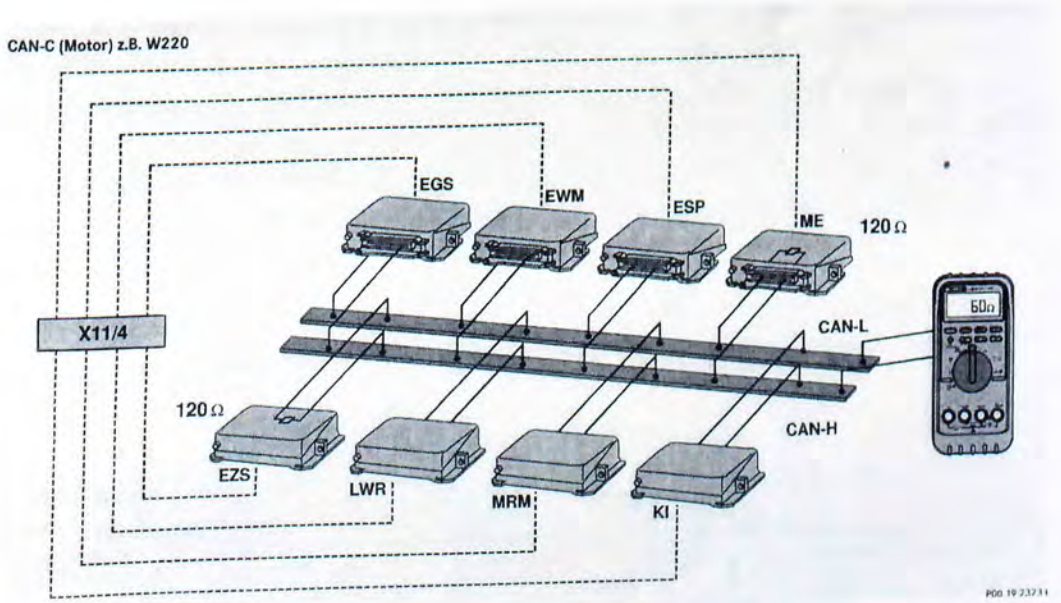
Κατά τη διάρκεια αυτή ο εντοπισμός ενός 'dominant' bit σηματοδοτεί την εκκίνηση αποστολής νέου FRAME (START OF FRAME).

SUSPENDTRANSMISSION

Οι κόμβοι που αναφέραμε αποστέλλουν οχτώ διαδοχικά 'recessive' bits. Αν ωστόσο αρχίσει αποστολή από άλλο κόμβο, τότε ο κόμβος αυτός γίνεται δέκτης.

4.9 Φυσικό επίπεδο

Το CAN-BUS όπως προαναφέραμε είναι μονοκάναλο, δηλαδή υπάρχει ένα μόνο σύρμα πάνω στο οποίο συνδέονται όλοι οι κόμβοι. Για την ακρίβεια κάθε κόμβος εκπέμπει το σήμα που θέλει και το κατοπτρικό του σήματός του (CAN-high, CAN-low), όπως φαίνεται στο παρακάτω σχήμα.



Εικόνα 4.12 : CanBus σχηματικά

Έτσι υπάρχει μία μορφή ελέγχου των σημάτων, δηλαδή όταν υπάρχει απόκλιση μεταξύ του CAN-high και του CAN-low τότε έχουμε παρεμβολές στο σήμα. Έπειτα τα σήματα αυτά τα αναλαμβάνουν microcontrollers για να τα μετατρέψουν σε σήματα txd και rxd, που είναι κατανοητά.

5^{O} ΚΕΦΑΛΑΙΟ

ΕΦΑΡΜΟΓΗ

5.1 Εισαγωγή

Το παράδειγμα που ακολουθεί περιλαμβάνει την εποπτεία της θερμοκρασίας στους κυλίνδρους μιας μηχανής εσωτερικής καύσης καθώς και των υπερτροφοδοτών της, μέσω ενός συστήματος SCADA. Se αυτό το σύστημα σαν κύριος ελεγκτής χρησιμοποιείται ένα PLC το οποίο έχει την δυνατότητα απεικόνισης γραφημάτων και επιτρέπει στον άνθρωπο – χειρίστη της μηχανής να έχει άμεση πληροφόρηση για τη κατάσταση της λειτουργιάς της, ώστε se περίπτωση κάποιου σφάλματος, να ειδοποιηθεί έγκαιρα με κάποιο συναγερμό και να προβεί στις απαραίτητες ενεργές για την διασφάλιση της ορθής λειτουργιάς ή παύσης της μηχανής, που σε αντίθετη περίπτωση θα είχε σοβαρά επακόλουθα όπως καταστροφή της ίδιας της μηχανής ή μερών αυτής ή ακόμα χειρότερα επακόλουθα στον περιβάλλοντα χώρο.

Οι ενδείξεις της θερμοκρασίας του κάθε κυλίνδρου ξεχωριστά λαμβάνονται με θερμοζεύγη τύπου K. Η έξοδος του κάθε αισθητήρα είναι αναλογική και ρυθμισμένη να μας δίνει από 4 έως 20 mA. Την μετατροπή της αναλογικής εξόδου του αισθητήρα se ψηφιακή αναλαμβάνει να την κάνει μια κάρτα αναλογική se ψηφιακή TMPCan-2 (AnaloguetoDigital) 8 καναλιών ενώ τους συναγερμούς (A) του συστήματος προς το περιβάλλον αναλαμβάνει να τους στείλει μια κάρτα με ψηφιακές εξόδους DIOCan-1 (DigitalInputOutput). Ο δίαυλοςεπικοινωνίας μεταξύ των καρτών και του κυρίως ελεκτή είναι μέσω CanBus και η όλη ανάπτυξη της εφαρμογής γίνεται μέσω του προγράμματος CodeSys.

5.2 Ξεκινώντας

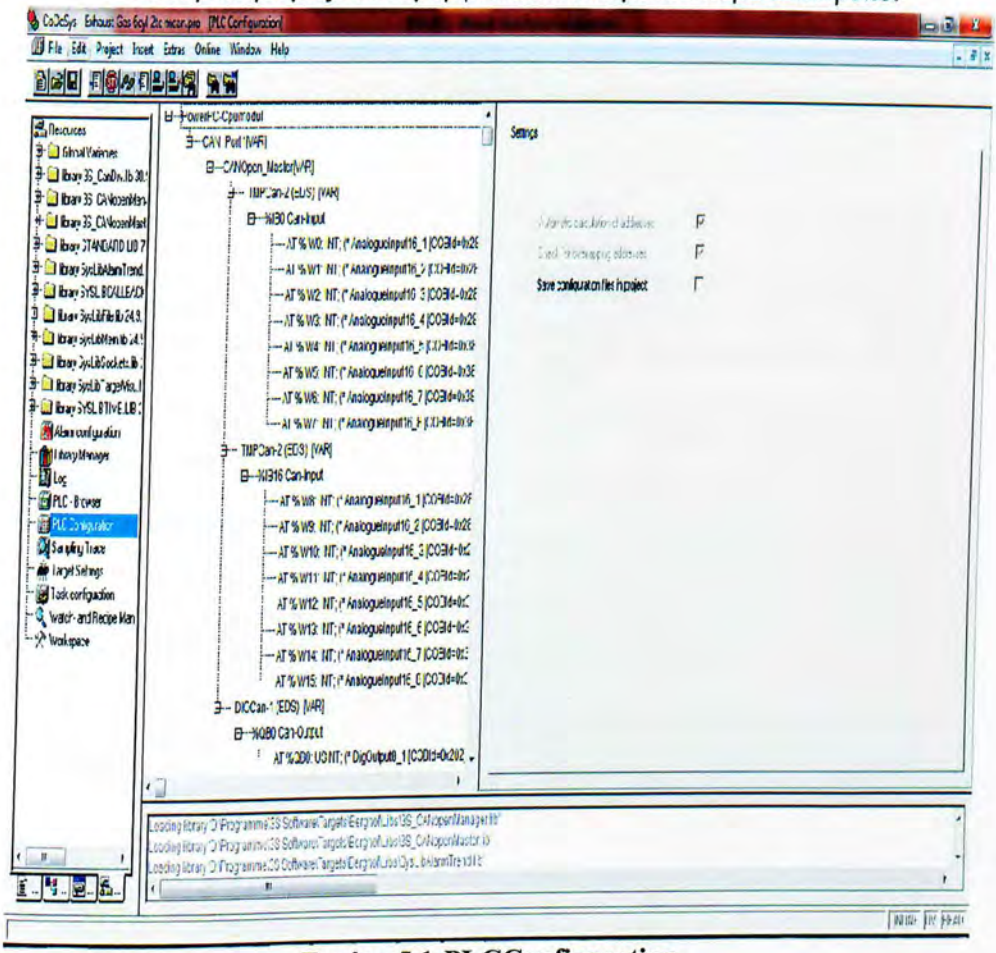
Πριν ξεκινήσουμε την διαδικασία προγραμματισμού στο CodeSys πρέπει να κάνουμε κάποιες ρυθμίσεις στον εξοπλισμό που πρόκυτε να χρησιμοποιήσουμε καθώς είναι επιτακτικό για την ορθή λειτουργία του προγράμματος.

5.2.1 Ορισμός των συσκευών

Στο παράθυρο Resourses επιλεγούμε την επιλογή PLCconfiguration. Se αυτό το σημείο δηλώνουμε:

- Το δίαυλο επικοινωνίας μας CANPORT1
- Τον κύριοελεγκτή μας CanOpen_Master
- Και τις κάρτες στις όποιες θα συνδεθούν οι αισθητήρες. Δυο αναλογικές κάρτες εισόδου θερμοκρασίας TMPCan-2, και η μια κάρτα Ψηφιακής Εξόδου-εισόδου DIOCan-1στην οποία θα συνδεθούν οι ενεργοποιητές (για παράδειγμα μια ηχητική ένδειξη).

Το παράθυρο μας θα διαμορφωθεί λοιπόν με τον παρακάτωτρόπο.

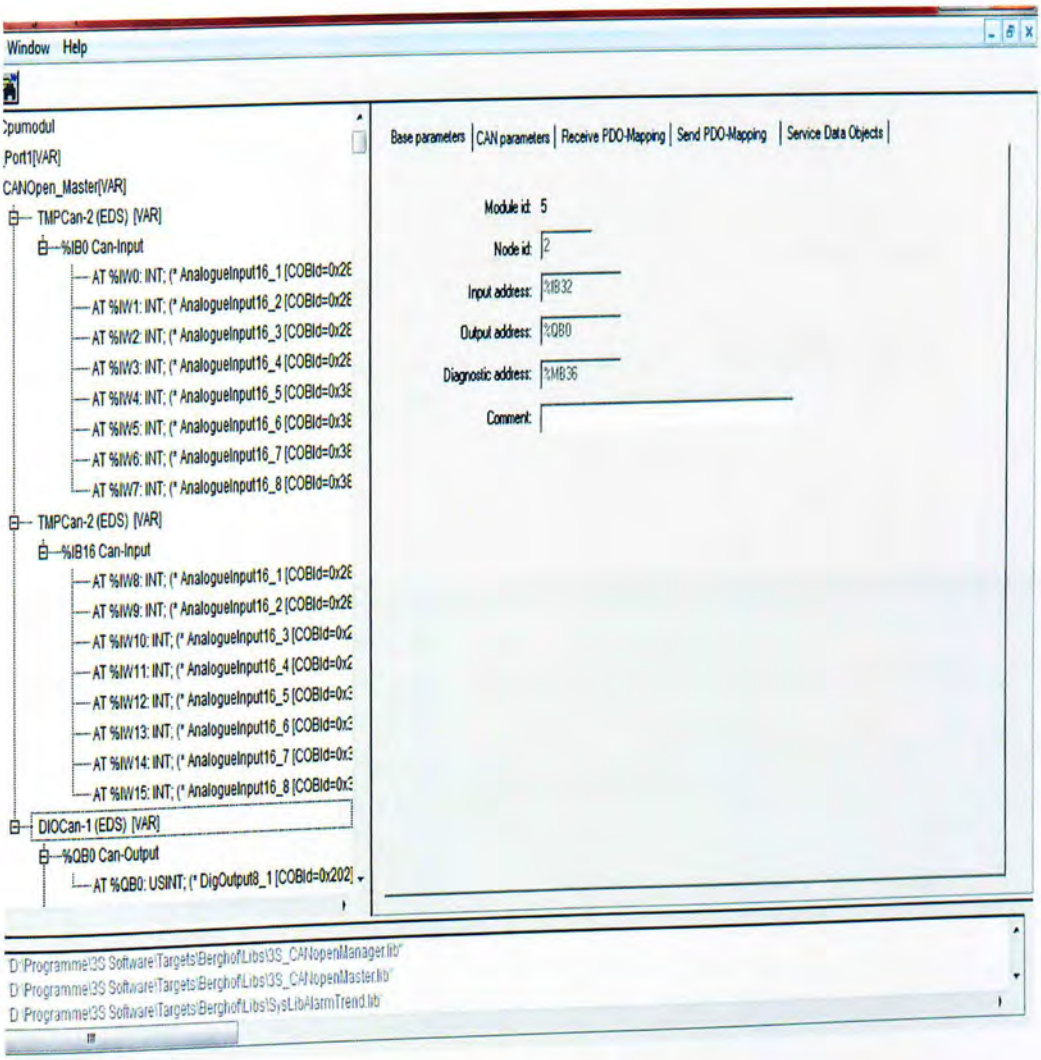


Εικόνα 5.1:PLCConfiguration

Στη συνεχεία πρέπει να γίνει επιμέρους ρύθμιση της κάθε συσκευής για τον τρόπο επικοινωνίας της μέσω του CanBus . Μερικές παραμέτρους από αυτές που πρέπει να ρυθμίσουμε είναι:

- NodeId. Το όποιοείναι η διεύθυνση κάθε συσκευής στο δίαυλο του CanBus, μοναδική για κάθε συσκευή.
- Baudrate. Ορυθμός μετάδοσης δεδομένων.
- Καθώς και ο τρόπος αποστολής και λήψης δεδομένων (Send – ReceivePDOMapping)

Όλες τις παραπάνω ρυθμίσεις τις κάνουμε επιλέγοντας κάθε συσκευή και η προσαρμογή τους γίνεται στο δεξιό τμήμα επιλέγοντας το κατάλληλο Table.



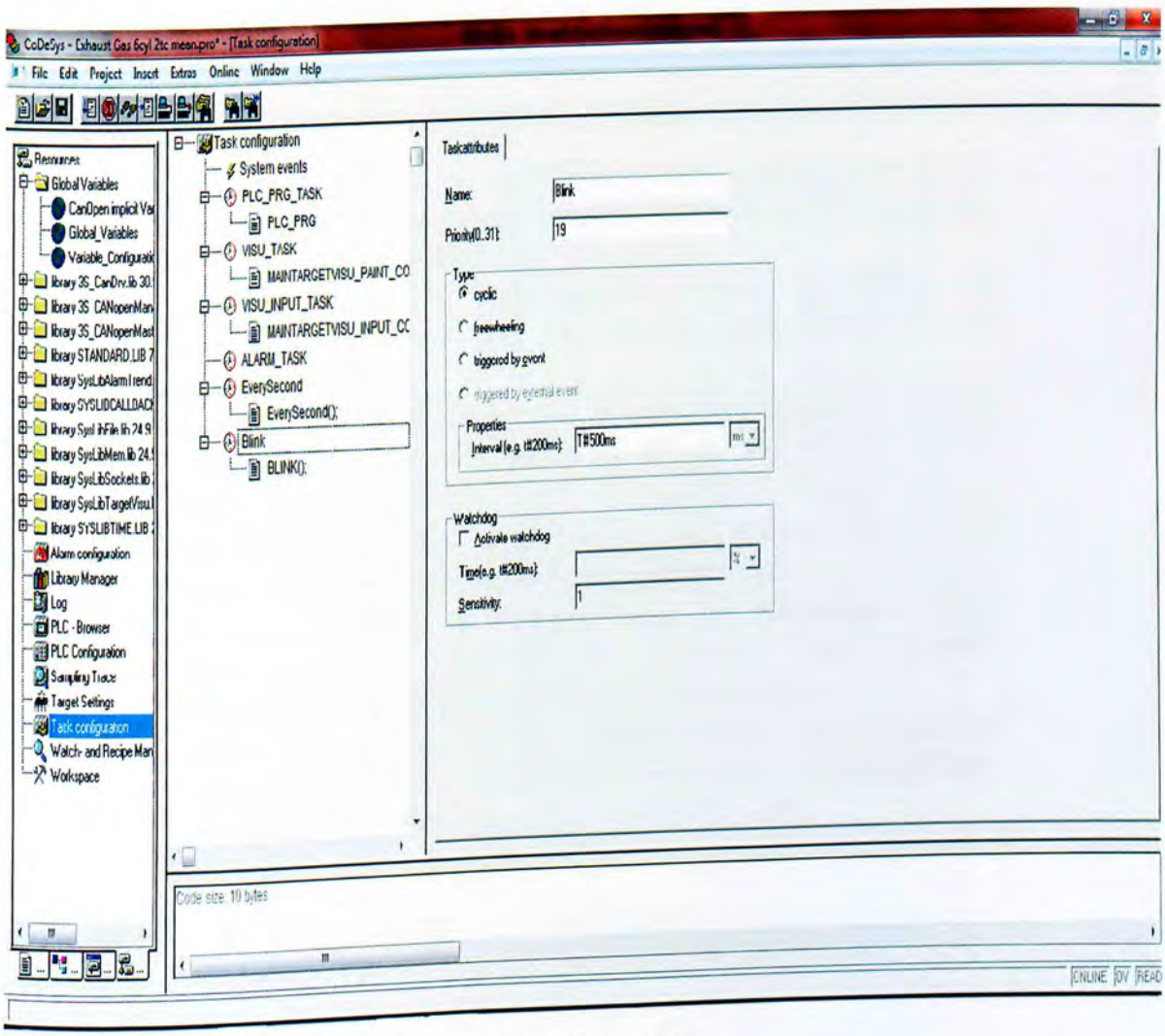
Εικόνα 5.2 : PLCConfigurationTABLES

Αφού τελειώσουμε με τις ρυθμίσεις του εξοπλισμού που θα χρησιμοποιήσουμε είμαστε έτοιμοι να περάσουμε στην ρύθμιση των γεγονότων Task της εφαρμογής καθώς και τους τύπους των συναγερμώνAlarms.

5.2.2 Ορισμός των Συνθηκών (TASKCONFIGOURATION)

Το πρόγραμμα μας για την λειτουργία του αποτελείτε από κάποιες συνθήκες που είτε τρέχουν καθ' όλη την διάρκεια του (cyclic) ή καλούνται από άλλες όταν έχουμε ένα συμβάν (triggeredbyevent). Μέσω του taskconfigurator το όποιο μας παρέχει το CodeSys επιλεγούμε και ρυθμίζουμε την προτεραιότητα της συνθήκης μας στο πρόγραμμα τον τρόπο που καλείται καθώς και τον χρόνο που θα ολοκληρώνει έναν κύκλο πριν ανανεώσει τα δεδομένα της.

Στο παρακάτωπαράθυρο θα δούμε ποιο αναλυτικά τις ρυθμίσεις που μπορούμε να κάνουμε sta Events του προγράμματος μας.



Εικόνα 5.3 :TaskConfiguration

Πρώτα ονομάζουμε το Task στο πεδίο Name και στην συνεχεία του δίνουμε έναν αριθμό προτεραιότητας . Αυτό το κάνουμε ώστε τα δευτερεύοντα Events να μην επισκιάζουν τα κυρία σε περίπτωση που ένα κύριο και ένα δευτερεύον event κληθούν ταυτόχρονα. Έπειτα δηλώνουμε τον τρόπο που θα καλείται το κάθε eventμέσα στο πρόγραμμα μας.

- Κυκλικό (τρέχει συνεχώς στο πρόγραμμα μας)
- TriggeredByEvent (Εκτελείται μόνο όταν το καλέσει μια άλλη συνθήκη)
- Freewheeling (εκτελείται τυχαία κατά διαστήματα.)

Οι συνθήκες χωρίζονται σε επιμέρους κατηγορίες ανάλογα με τον σκοπό που εξυπηρετούν. Ο διαχωρισμός τους εναπόκειται σε συνθήκες καθαρού κώδικα , σε συνθήκες οπτικής απεικόνισης (Visualizations) και σε συνθήκες συναγερμών (alarms).Οι δυο πρώτες κατηγορίες είναι το πρόγραμμα μας και δημιουργούνται από εμάς για να επιτελέσουμε το στόχο μας.

Στην εφαρμογή μας έχουμε τις παρακάτω συνθήκες

- PLC_PRGTask.Είναι το βασικόπρόγραμμα της εφαρμογής μας. Τρέχει συνεχώς έχει μέγιστη προτεραιότητα για αυτό και της δίνουμε priority #1 και ανανεώνετε κάθε 10ms
- VISU_TASK.Είναι το βασικό μιμικό μας που εποπτεύουμε την εφαρμογή. Τρέχει συνεχώς ανανεώνεται κάθε 200ms και έχει προτεραιότητα #15.
- VISU_INPUT_TASK.Δευτερεύον μιμικό στην εφαρμογή που μας δίνει την κατάσταση των κόμβων NODE του δίαυλου επικοινωνίας. Τρέχει συνεχώς ανανεώνεται κάθε 50ms και έχει προτεραιότητα #14.
- ALARM_TASK. Είναι το μπουκέτο των συναγερμών.Τρέχει συνεχώς ανανεώνεται κάθε 50ms και έχει προτεραιότητα #13.
- EverySecond Είναι το βασικόστοιχειό στο πρόγραμμα για να διασφαλίσει την επικοινωνία του δίαυλου. Τρέχει συνεχώς ανανεώνεται κάθε #1s0ms και έχει προτεραιότητα #19.
- Blink. Είναι το κομμάτι του κώδικα το οποίο όταν έχουμε ένα συναγερμό μας δίνει ενδείξεις στο μιμικό μας. Τρέχει συνεχώς ανανεώνεται κάθε #500ms και έχει προτεραιότητα #19.

Οι τύποι των alarms που παίρνουμε από το πρόγραμμα μας πρέπει να τους ορίσουμε σε ένα άλλο παράθυρο του CodeSys ALARM CONFIGURATION.

5.2.3 Ορισμός των Συναγερμών (ALARMCONFIGOURATION)

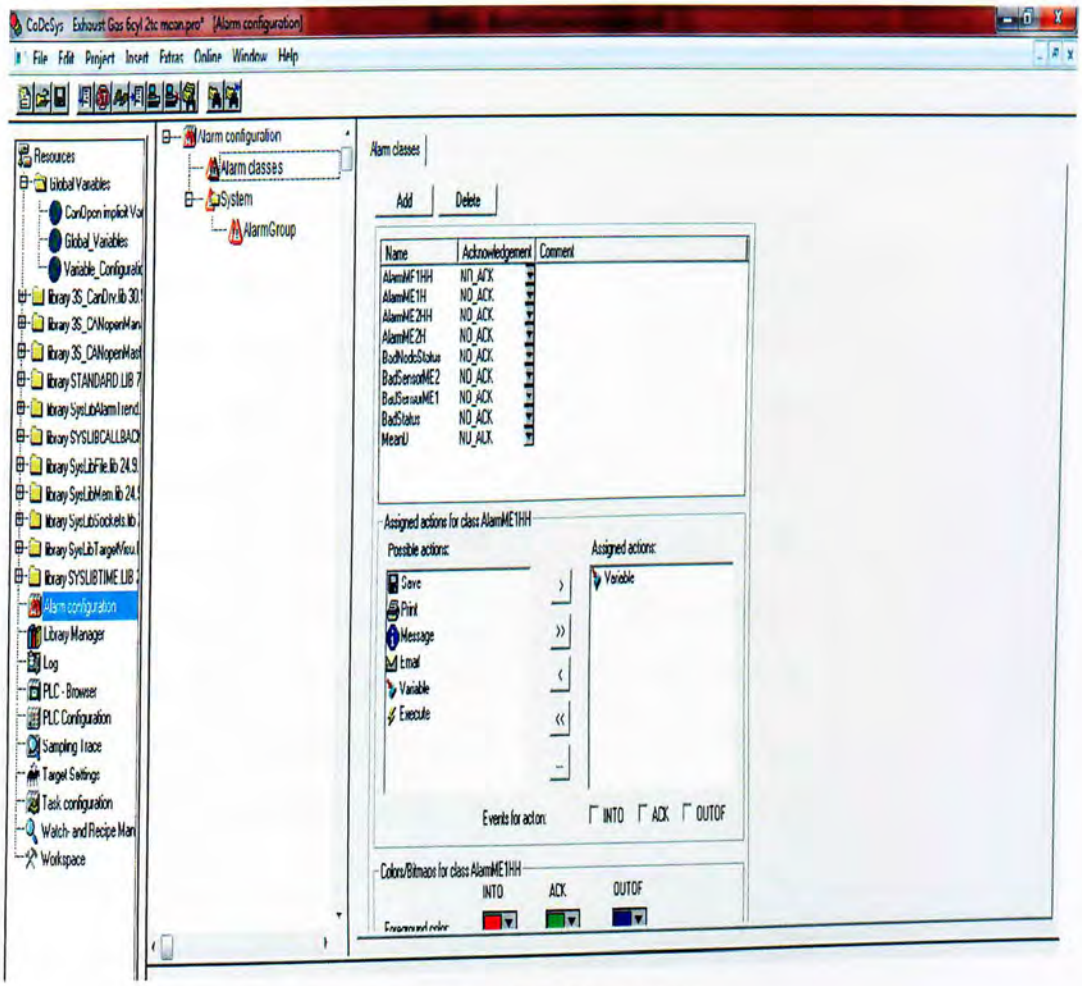
Όταν μια από τις μεταβλητές του προγράμματος μας λάβει μια τιμή που δεν είναι επιθυμητή τότε πρέπει ο χρηστής- χειρίστης του συστήματος να ενημερωθεί για την κατάσταση του συστήματος. Στο παράδειγμα μας παρουσιάζουμε την εποπτεία της θερμοκρασίας των κυλίνδρων μια μηχανής εσωτερικής καύσης. Ο χειρίστης πρέπει να γνωρίζει ανά πάσα στιγμή την θερμοκρασία κάθε κυλίνδρου χωριστά και σαν μέσω όρο. Διότι κακή λειτουργία έχουμε όταν ένας η περισσότεροι κύλινδροι υπερθερμανθούν, ένας η περισσότεροι κύλινδροι κρυώσουν, άρα έχουν σταματήσει είτε ένας ή περισσότεροι κύλινδροι υπολειτουργούν όποτε η θερμοκρασία τους έχει μεγάλη απόκλιση από τον μέσω όρο της θερμοκρασίας των υπολοίπων κυλίνδρων.

Όμως και το σύστημα μας πρέπει να ελέγχεται για σφάλματα γιατί οι λάθος ενδείξεις για την θερμοκρασία της μηχανής που μπορεί να λαμβάνουμε μπορεί να οφείλονται σε σφάλμα του εξοπλισμού μας. Όποτε πρέπει να εποπτεύουμε και τους αισθητήρες μας επιμέρους και τις αναλογικές σε ψηφιακές κάρτες αλλά και τον Master ελεκτή μας που είναι το PLC.

Στο παράθυρο που ακόλουθη, ορίζουμε λοιπόν τις κλάσεις των συναγερμών ανάλογα με τις συνθήκες που αναφέραμε παραπάνω.

- AlarmME1HH-Ο κύλινδρος έχει υπερθερμανθεί
- AlarmME1H- Ο κύλινδρος είναι κρύος
- AlarmME2HH-Ο κύλινδρος έχει υπερθερμανθεί (Για δεύτερη μηχανή αν υπάρχει)
- AlarmME2H- Ο κύλινδρος είναι κρύος (Για δεύτερη μηχανή αν υπάρχει)
- BadNodeStatus- Ένας από τους κόμβους του δικτύου το CAnBUS έχει σφάλμα
- BadSensor1- Υπάρχε ισφάλμα στον αισθητήρα της μηχανής
- BadSensor2- Υπάρχει σφάλμα στον αισθητήρα της δεύτερη ςμηχανής
- BadStatus – Υπάρχει σφάλμα στο σύστημα γενικότερα.
- MeanD- Η μέση θερμοκρασία παρουσιάζει μεγάλες αποκλίσεις

Το παράθυρο μας λοιπόν τροποποιείται και είναι το παρακάτω



Εικόνα5.4 : AlarmConfiguration-AlarmClass

Αφού λοιπόν έχουμε δημιουργήσει ta Classes των συναγερμών του συστήματος πρέπει να αντιστοιχίσουμε τις μεταβλητές του προγράμματος μας με αυτούς, καθώς και σε ποιες τιμές των μεταβλητών μας θα καλούνται. Στο παράθυρο μας διαλέγουμε το AlarmGroup και ξεκινάμε να εισάγουμε τις μεταβλητές μας όπως τις έχουμε δηλώσει στο κυρίως πρόγραμμα μας και τις ανάγουμε στο κατάλληλο alarmclass που έχουμε δημιουργήσει. Συνεχίζουμε λοιπόν παρακάτω επιλέγοντας System ->Alarmgroup και με το κουμπί ADD προσθέτουμε τις μεταβλητές μας και τους ορίζουμε τον τύπο του συναγερμού που θα ενεργοποιούν.

- Στην στήλη expression γράφουμε τις μεταβλητές.
- Στην στήλη Type επιλεγούμε τον τρόπο που θα ενεργοποιείται ο συναγερμός π.χ. αλλάζοντας τιμή από true s e falseή το αντίθετο
- Στην τρίτη στήλη επιλεγούμε την κλάση του συναγερμού που έχουμε δημιουργήσει
- Στην τετάρτη επιλέγουμε την σημαντικότητα του έναντι των άλλων συναγερμών

Αναλυτικότερα θα έχουμε τις ακόλουθες μεταβλητές με τις εξής κλάσεις

Τιμή Μεταβλητής	Κλάση Συναγερμού
PLC_PRG.temp11>PLC_PRG.HHAlarm11	AlarmME1HH
PLC_PRG.temp11<PLC_PRG.HAlarm11	AlarmME1H
PLC_PRG.temp12>PLC_PRG.HHAlarm11	AlarmME1HH
PLC_PRG.temp12<PLC_PRG.HAlarm11	AlarmME1H
PLC_PRG.temp13>PLC_PRG.HHAlarm11	AlarmME1HH
PLC_PRG.temp13<PLC_PRG.HAlarm11	AlarmME1H
PLC_PRG.temp14>PLC_PRG.HHAlarm11	AlarmME1HH
PLC_PRG.temp14<PLC_PRG.HAlarm11	AlarmME1H
PLC_PRG.temp15>PLC_PRG.HHAlarm11	AlarmME1HH
PLC_PRG.temp15<PLC_PRG.HAlarm11	AlarmME1H
PLC_PRG.temp16>PLC_PRG.HHAlarm11	AlarmME1HH
PLC_PRG.temp16<PLC_PRG.HAlarm11	AlarmME1H

Στις παραπάνω μεταβλητές βλέπουμε τους τύπους του συναγερμού που έχει αποδοθεί σε κάθε κύλινδρο του κινητήρα όταν υπερβεί ή πέσει κάτω από μια θερμοκρασία. Παρακάτω εργαζόμαστε το ίδιο και για τις θερμοκρασίες του υπερτροφοδότη και την μέση θερμοκρασία των κυλίνδρων

PLC_PRG.temp1TC>PLC_PRG.HHAlarm1TC	AlarmME1HH
PLC_PRG.temp1TC<PLC_PRG.HAlarm1TC	AlarmME1H
PLC_PRG.temp1Mean>PLC_PRG.HHAlarm1Mean	MeanD
PLC_PRG.temp1Mean<PLC_PRG.HAlarm1Mean	MeanD

Στην συνέχεια αποδίδουμε στις μεταβλητές που προκύπτουν μέσα από το τον δίαυλο του Canbus την κλάση badNodeStatus όταν η τιμή τους γίνει 5.

PLC_PRG.NodeStatus[0]=5BadNodeStatus

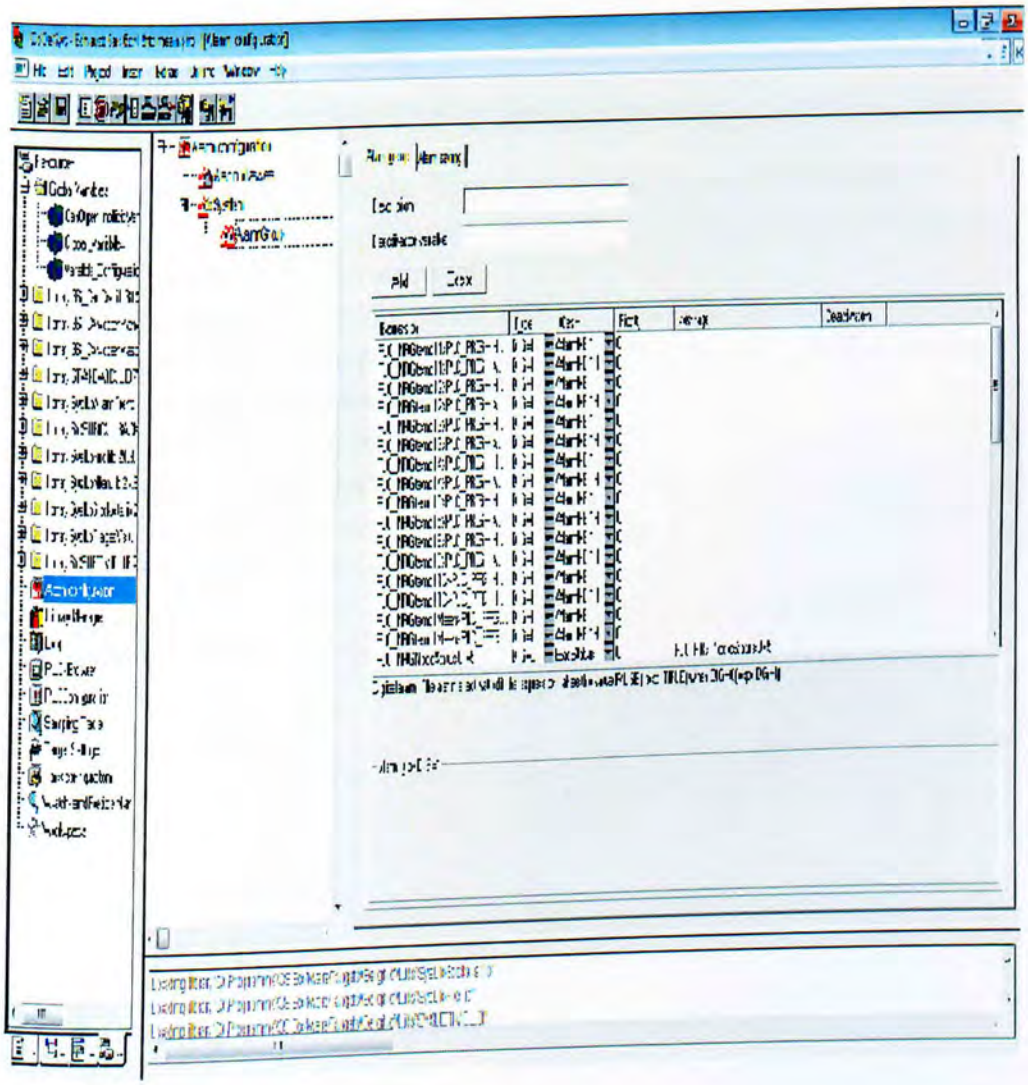
PLC_PRG.NodeStatus[1]=5BadNodeStatus

PLC_PRG.NodeStatus[2]=5BadNodeStatus

Τέλος ελέγχουμε εάν υπάρχει κάποιος προβληματικός αισθητήρας στην εφαρμογή μας, όταν διαβάσουμε μια αφύσικη τιμή από έναν αισθητήρα. Εδώ επιλέγουμε μια

τιμή αρνητική που δεν θα μπορούσαμε να την πάρουμε ποτέ σε κατάσταση καλής λειτουργίας.

Τιμή Μεταβλητής	Κλάση Συναγερμού
PLC_PRG.temps11=-5472	BadSensor1
PLC_PRG.temps12=-5472	BadSensor1
PLC_PRG.temps13=-5472	BadSensor1
PLC_PRG.temps14=-5472	BadSensor1
PLC_PRG.temps15=-5472	BadSensor1
PLC_PRG.temps16=-5472	BadSensor1
PLC_PRG.temps1TC=-5472	BadSensor1



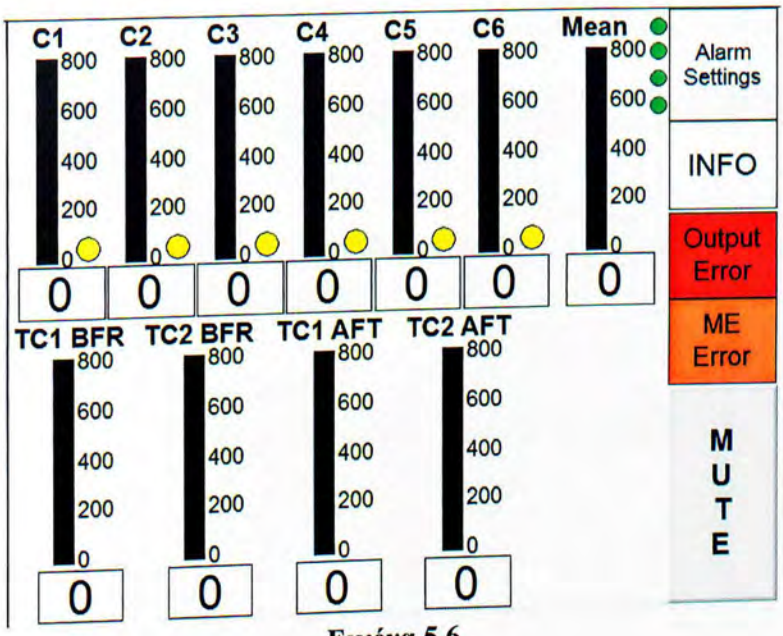
Εικόνα 5.5

5.4 ΟπτικήΑπεικόνιση (Visualizations)

Όπως έχουμε ήδη αναφέρει, το masterPLC που χρησιμοποιούμε για τον έλεγχο της διαδικασίας έχει την δυνατότητα απεικόνισης μιμικού πάνελ αφού διαθέτει μια οθόνη αφής 7΄΄.Το σύστημα μας έχει την ικανότητα να αντιλαμβάνεται σφάλματα που προξενεύονται στην διαδικασία καθώς και σφάλματα που μπορούν να προκληθούν στον εξοπλισμό που χρησιμοποιούμε. Έτσι έχουμε δύο παράθυρα που εποπτεύει ο χειριστής κατά την διάρκεια της διαδικασίας. Ένα που αφόρα την ίδια την διαδικασία και ένα δεύτερο που αφόρα τα κομμάτια του εξοπλισμού μας και πιο συγκριμένα τις αναλογικές μας κάρτες που μετατρέπουν τα σήματα των αισθητήρων σε μορφή μηνύματος CANBUS

.Στο πρώτο παράθυρο βλέπουμε τα έξης:

- Μπάρες οι οποίες δείχνουν την θερμοκρασία του κάθε κυλίνδρου της μηχανής (C1, C2,C3 …..)
- Μπάρες οι οποίες δείχνουν την θερμοκρασία του κάθε υπερτροφοδότη (TC1 BRF, TC2 BRF…..)
- Μια μπάρα που δείχνει την μέση θερμοκρασία της μηχανής (MEAN)
- Ένα πλήκτρο για να ρυθμίζουμε τα επίπεδα που θέλουμε να ενεργοποιείται ένας συναγερμός (Alarmsetting)
- Δυο ενδεικτικά ME Error και OUTPUTerror τα οποία μας ενημερώνουν για το αν ο συναγερμός έχει σημάνει λόγο προβλήματος που προέκυψε από την ίδια την μηχανή ή τον εξοπλισμό μας
- Ένα πλήκτρο MUTE το όποιο σιγεί τον συναγερμό μας .
- Ενα πλήκτρο ACK (acknowledged) το οποίο εμφανίζεται όταν έχουμε ένα συναγερμό και το οποίο το πιέζουμε για να κάνουμε reset και ότι έχουμε δει το είδος του συναγερμού.
- Ένα πλήκτρο το οποίο μας περνά στην επόμενη σελίδα μας που είναι το μιμικό για την εποπτεία των καρτών μας.

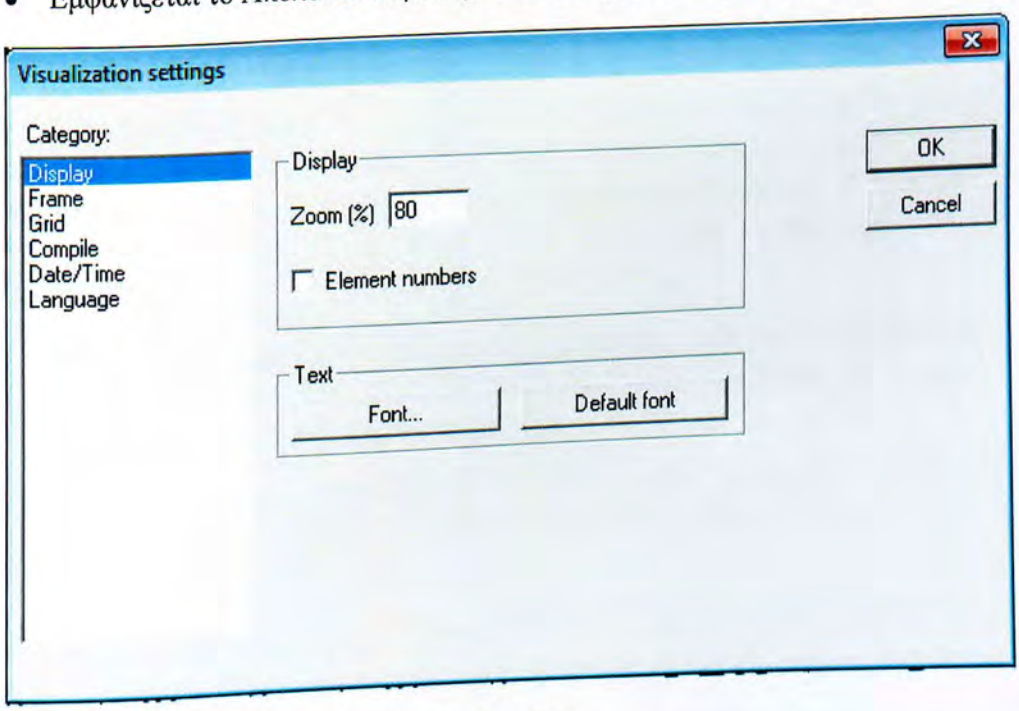


Εικόνα 5.6

Σχεδιασμός Button

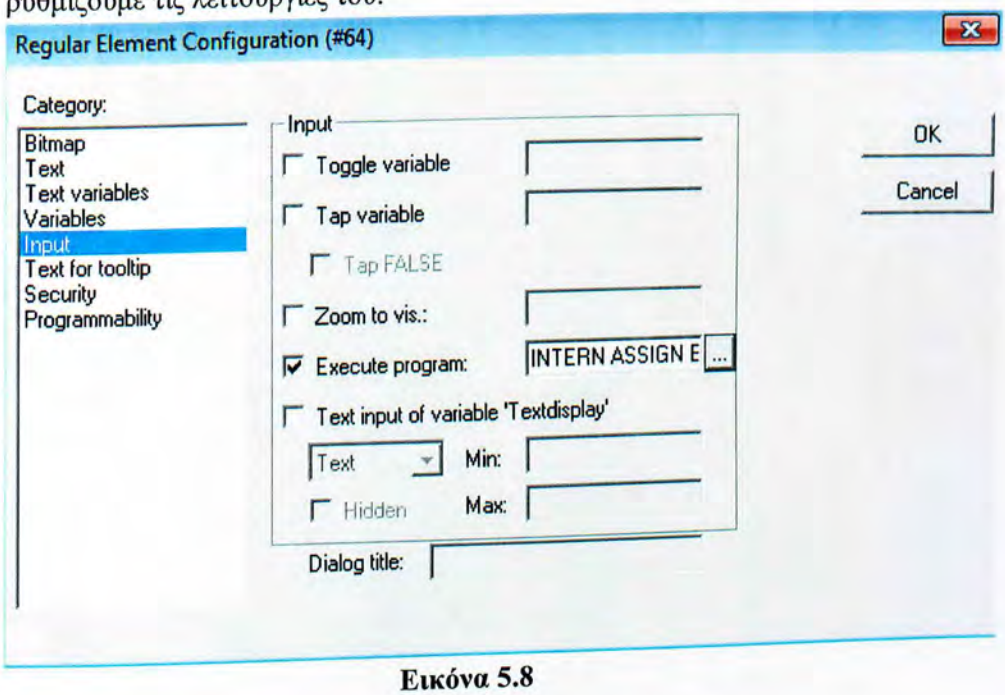
Για να σχεδιάσουμε ένα button εντολής στο σύστημα μας ακολουθούμε τα εξής βήματα:

- Σχεδιάζουμε το Buttonκαι αφού κάνουμε δεξί κλικ επιλέγουμε settings
- Εμφανίζεται το Ακόλουθο παράθυροοπού ρυθμίζουμε την εμφάνιση του.



Εικόνα 5.7

- Έπειτα κάνοντας διπλό κλικ εμφανίζεται το παρακάτω παράθυρο οπού του ρυθμίζουμε τις λειτουργίες του.



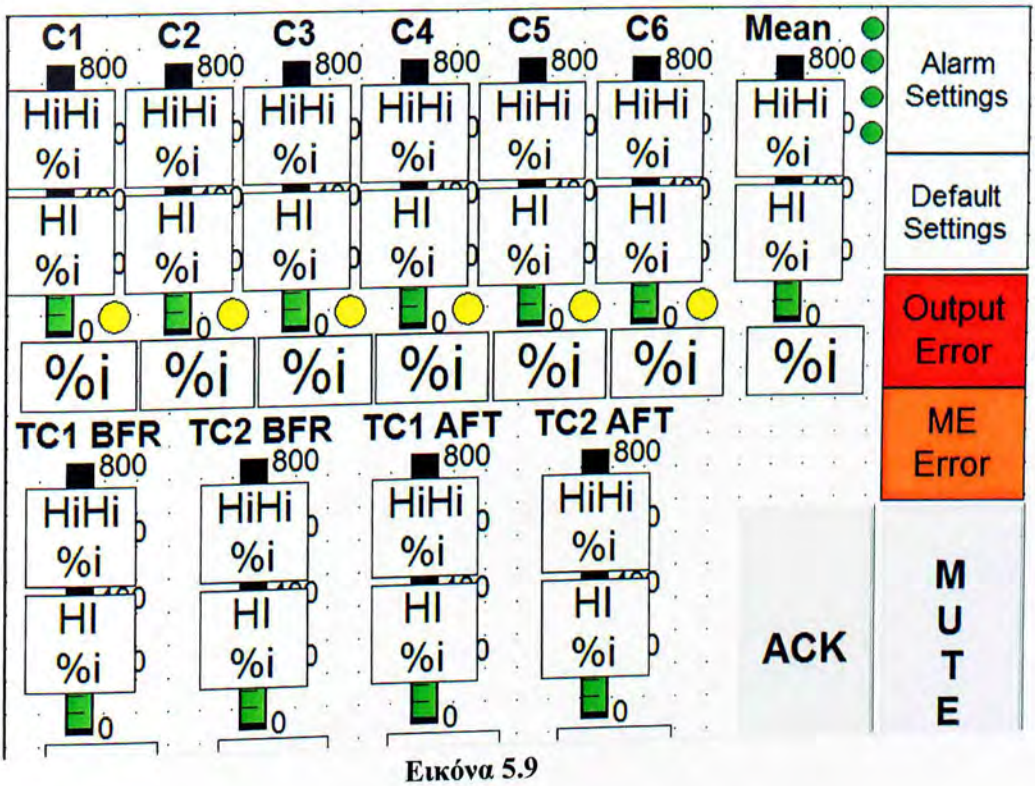
Εικόνα 5.8

Από τις ρυθμίσεις του παραπάνω πίνακα οι πιο βασικές είναι η Input και η Variables.

Με την Input ρυθμίζουμε ποιο πρόγραμμα θα τρέξει όταν ενεργοποιηθεί το button ενώ με την Variable από ποια μεταβλητή ενεργοποιείται.

Παραδείγματος χάρη, σε περίπτωση που προκύψει έναςσυναγερμός και ενεργοποιηθεί η μεταβλητή INTERN ASSIGN BLINK.blinkreset:=(TRUE). Τότε εμφανίζεται στο μιμικό μας το button ACK. Όταν πιέσουμε εμείς το Button τότε δίνουμε στο πρόγραμμα μας εντολή να κάνει reset μέσω του NOT BLINK.resetenable.

Για να ρυθμίσουμε τις ανώτερες και τις κατώτερες τιμές που αν ξεπεραστούν τότε Τα έχουμε έναν συναγερμό στο σύστημά μας, Πιέζουμε το Κουμπί AlarmSettings και εμφανίζεται η παρακάτω εικόνα.



Εικόνα 5.9

Στην συνέχεια πατώντας πάνω στο HiHi ή Hi της αντίστοιχης θερμοκρασίας που επιθυμούμε, αυξομειώνουμε το επίπεδο της Θερμοκρασίας.

Το μιμικό για την εποπτεία των καρτών μας περιλαμβάνει τις έξης παραμέτρους:

- Μια στήλη που μας δείχνει το id του κόμβου στο CANBUS για τον οποίο αναφερόμαστε
- Μια στήλη που μας δείχνει την κατάσταση του
- Μια στήλη που μας δείχνει την τιμή του πακέτου του μηνύματος που στέλνει στον δίαυλο του CANBUS
- Τέλος έχει ένα πλήκτρο για να μας γυρίσει στην πλοηγούμένη σελίδα.

Για την ρύθμιση των παραμέτρων του παρακάτω πίνακα κάνουμε διπλό κλικ για να εμφανιστεί το παράθυρο ρυθμίσεων και ενεργούμε όπως έχουμε υποδείξει σε προηγούμενη παράγραφο.

Index	Status	ValueStr
0	0	
0	0	
0	0	

MainPage

Εικόνα 5.10

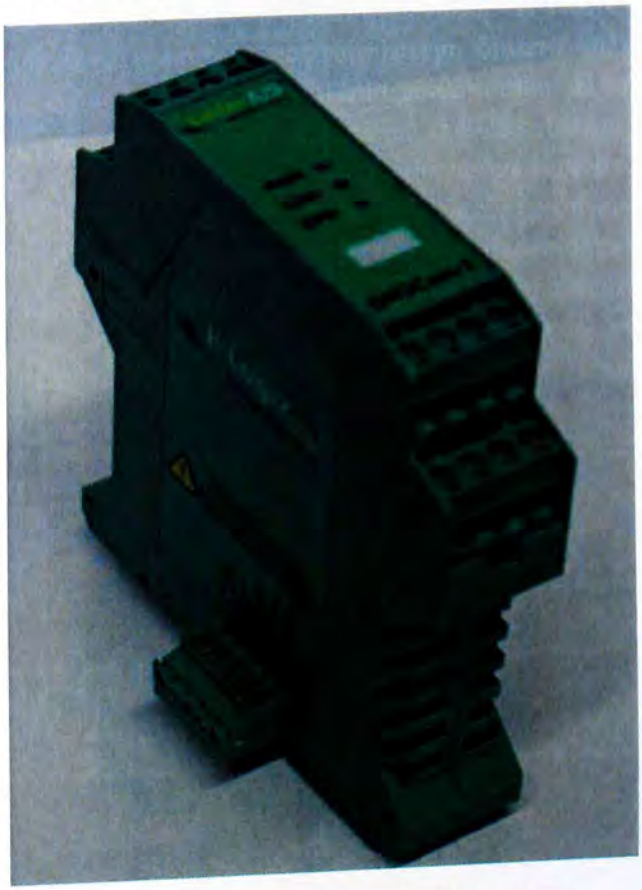
5.5 Συσκευές Εφαρμογής

5.5.1 Γενικά

Η TMPCan-2 είναι μια αναλογική κάρτα εισόδου θερμοκρασίας ,αποτελείται από οκτώ κανάλια εισόδου θερμοζεύγους τύπου K σε δίαυλο CanOpen .Μέσω της TMPCan -2 τα θερμοζεύγοι μπορούν εύκολα να συνδεθούν με τα δίκτυα CanOpen . Τα δεδομένα της εισόδου του κόμβου μπορούν να μεταδοθούν συγχρονισμένα ή ασυγχρόνιστα . Η TMPCan-2 είναι σχεδιασμένη με τέτοιο τρόπο ώστε να απαιτεί την χαμηλότερη δυνατή καλωδίωση. Δεν χρειάζεται καλωδίωση για να συνδεθεί μια TMPCan-2 με μια άλλη ,καθώς οι συσκευές συνδέονται μεταξύ τους με πέντε ακτίδων TBUS. Τελικά η επικοινωνία Canbus μας δίνει την δυνατότητα να ελαχιστοποιούμε το κόστος της καλωδίωσης με το να μπορούμε να συνδέσουμε έως 127 TMPCan-2 (ή συνδυασμός πολλαπλών από την οικογένεια προϊόντων E-Logix 2K) κόμβου διανεμημένη στο πεδίο χρησιμοποιώντας μόνο ένα καλώδιο για την διασύνδεση των απομακρυσμένων κόμβων .Η συγκεκριμένη συσκευή είναι σχεδιασμένη για δυσμενή περιβάλλοντα. Η τροφοδοσία της είναι προστατευμένη σε ανάστροφη πόλωση , υπερτάσεις και υπερεντάσεις.

Εικόνα 5.11:TMPCan-2 CanOpenNode

Η DIOCan1 είναι μια οκτακάναλη πολυλειτουργική ψηφιακή κάρτα εισόδου – εξόδου CanOpen κόμβου . Διαθέτει απομόνωση μεταξύ των καναλιών της .Επιπλέον τα κανάλια επτά και οκτώ μπορούν να χρησιμοποιηθούν ως ρυθμιζόμενοι είσοδοι και να μετρούν στροφές το δευτερόλεπτο σε δυο διαφορετικούς τύπους λειτουργιάς . Στον «ανεξάρτητο» τύπο λειτουργίας η κάθε είσοδο πραγματοποιεί μια διαφορετική μέτρηση στροφών . Ενώ στον δεύτερο «συνδυαστικό» τύπο , οι δυο είσοδοι συνδυάζονται και η συσκευή μεταδίδει στροφές ανά λεπτό και διεύθυνση .



Εικόνα 5.12 :DIOCan1 CanOpenNode

Τα προϊόντα της εταιρίας E-Logix 2K , μπορούν να υποστηρίξουν λειτουργιά επαναλαμβανόμενης μετάδοσης δεδομένων (Heartbeat) . Με αυτό τον ελεγκτικό μηχανισμό του κόμβου , ο ίδιος κόμβος μπορεί να μεταδίδει την ικανότητα του να επικοινωνεί ,εκπέμποντας κυκλικά ένα "heartbeat" μήνυμα . Με αυτόν τον τρόπο ένας κόμβος με μια σοβαρή βλάβη εισέρχεται αυτόνομα σε μια προκαθορισμένη κατάσταση με σκοπό να μην επηρεάσει την λειτουργιά άλλων συσκευών μέσα στα δίκτυα CanOpen .

5.5.2 Χαρακτηρίστηκα

5.5.2.1 Συνδεσμολογία της αναλογικής κάρτας TMPCan-2.

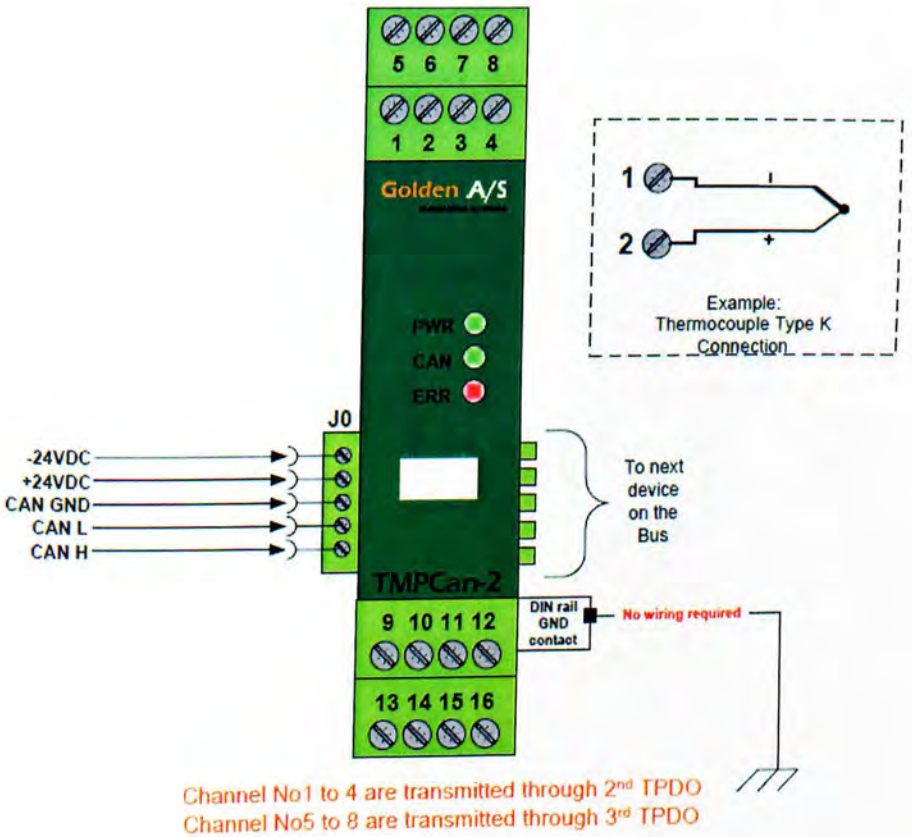
Υπάρχουν δυο διαφορετικοί τύποι συνδεσμολογίας της συσκευής όπως βλέπουμε παρακάτω (Εικόνα 6.3).

Εικόνα 5.13

ή J0: είναι η είσοδος για το καλώδιο του CanBus και της τροφοδοσίας.

J1-J4 : είναι οι κλέμες για τις εισόδους 1-8 θερμοζεύγους τύπου K.

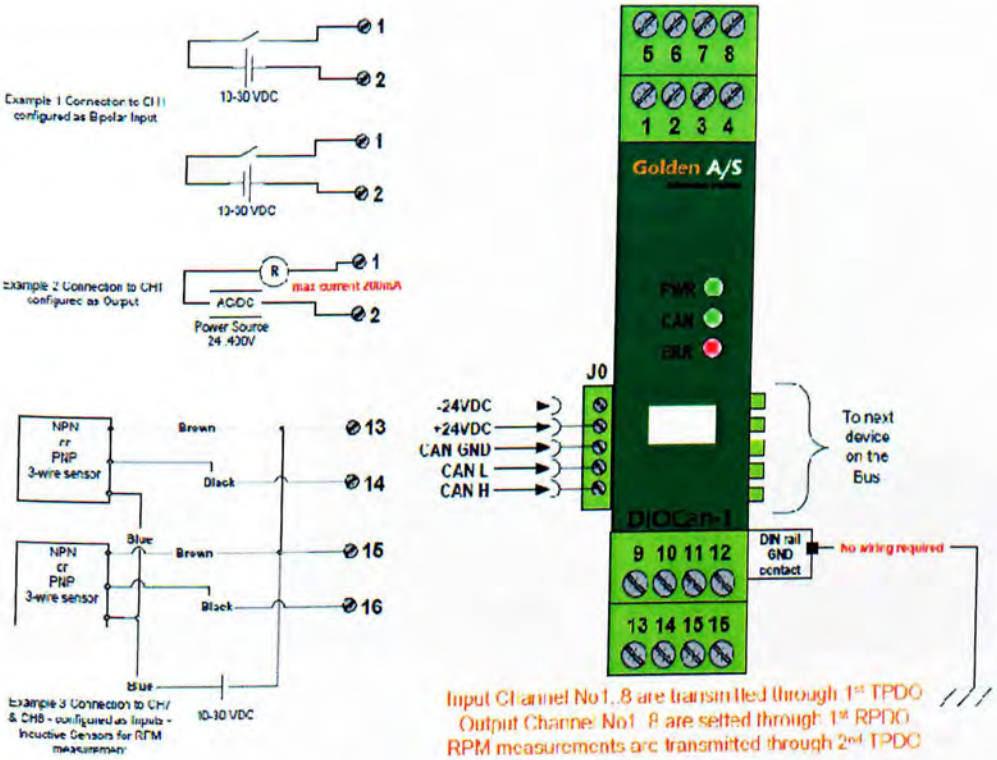
Σε κάθε συνδυασμό έχουμε αποδώσει μια μοναδική θέση , η όποια μπορεί να τοποθετηθεί στην συσκευασία έτσι ώστε να αποφευχθούν λάθη , όταν προσπαθήσουμε να αλλάξουμε την κάρτα στο χώρο εργασίας . Όλοι οι σύνδεσμοι μπορούν να αποσυνδεθούν από την συσκευή χωρίς να χρειαστεί να αποσυνδεθούν τα καλώδια .Όλα τα καλώδια είναι συνδεδεμένα σε ένα αρσενικό σύνδεσμο ο όποιος βρίσκεται μόνιμα πάνω στην πλακέτα PCM .



Εικόνα5.14 : Συνδεσμολογία TMPCan-2

Channel No	Terminal	Type	Description
CH1	1	IN(-)	Thermocouple Input
	2	IN(+)	
CH2	3	IN(-)	Thermocouple Input
	4	IN(+)	
CH3	5	IN(-)	Thermocouple Input
	6	IN(+)	
CH4	7	IN(-)	Thermocouple Input
	8	IN(+)	
CH5	9	IN(-)	Thermocouple Input
	10	IN(+)	
CH6	11	IN(-)	Thermocouple Input
	12	IN(+)	
CH7	13	IN(-)	Thermocouple Input
	14	IN(+)	
CH8	15	IN(-)	Thermocouple Input
	16	IN(+)	

Εικόνα 5.15: Διάγραμμα Συνδεσμολογίας



Εικόνα5.16: Συνδεσμολογία DIOCan-1

Channel No	Terminal	Type	Description
CH1	1	IN(+)/IN(-)/OUT1	Digital Input / Relay SPST output
	2	IN(-)/IN(+)/OUT1	
CH2	3	IN(+)/IN(-)/OUT2	Digital Input / Relay SPST output
	4	IN(-)/IN(+)/OUT2	
CH3	5	IN(+)/IN(-)/OUT3	Digital Input / Relay SPST output
	6	IN(-)/IN(+)/OUT3	
CH4	7	IN(+)/IN(-)/OUT4	Digital Input / Relay SPST output
	8	IN(-)/IN(+)/OUT4	
CH5	9	IN(+)/IN(-)/OUT5	Digital Input / Relay SPST output
	10	IN(-)/IN(+)/OUT5	
CH6	11	IN(+)/IN(-)/OUT6	Digital Input / Relay SPST output
	12	IN(-)/IN(+)/OUT6	
CH7	13	IN(+)/IN(-)/OUT7	Digital Input (RPM CH1) / Relay SPST output
	14	IN(-)/IN(+)/OUT7	
CH8	15	IN(+)/IN(-)/OUT8	Digital Input (RPM CH2) / Relay SPST output
	16	IN(-)/IN(+)/OUT8	

Εικόνα 5.17: Διάγραμμα Συνδεσμολογίας

5.5.2.2 Προκαθορισμός αριθμού κόμβου

Κάθε συσκευή σε ένα CANBUS δίκτυο πρέπει να έχει μια μοναδική ταυτότητα (διεύθυνση)στον κόμβο (NODEID) .Οι διαθέσιμες διευθύνσεις εκτίνονται συμφώνα με τις προδιαγραφές του δικτυού από το 1-127 . Η TMPCan-2 υποστηρίζει μόνο μηχανικό τρόπο εισαγωγής διεύθυνσης δικτυού (NODEID) .Η ρύθμιση της διεύθυνσης δικτύου μπορεί να γίνει μέσω μιας σειράς διακόπτων που βρίσκονται μέσα στο PCB . Για να πραγματοποιηθεί αυτό ακολουθούντα τα παρακάτω βήματα .

Εικόνα 5.18: Αριστερά Αφαίρεση καλύμματος- Δεξιά διακόπτες επιλογής διεύθυνσης

Χρησιμοποιώντας ένα μικρό κατσαβίδι ,πιέζουμε τις δαγκάνες του πάνω καλύμματος , τραβώντας το ταυτόχρονος προς τα πάνω (και από τις δυο μεριές κάνουμε το ίδιο). Στην συνεχεία αφού απαγκιστρωθεί το τραβάμε για ανά εφαρμοστεί το PCB οι διακόπτες από το 1-7 χρησιμοποιούνται για να δώσουμε διεύθυνση στο κόμβο από 1-127 . Κάθε διακόπτης μπορεί να είναι σε θέση ON-OFF. Ο πινάκας στο παράρτημα Χ μας δείχνει τις πιθανές θέσεις των διακόπτων ώστε να έχουμε τιμές από 1-127 . Ο πινάκας αυτός είναι μια απλή μετατροπή του δεκαδικού σε δυαδικού συστήματος , οπού ο διακόπτης 1 αντίστοιχοι στο χαμηλότερης ισχύς bit (LSB) και ο διακόπτης 7 αντιστοιχεί στο υψηλότερης ισχύς bit (MSB) .

5.5.2.3 Ρυθμός μετάδοσης δεδομένων

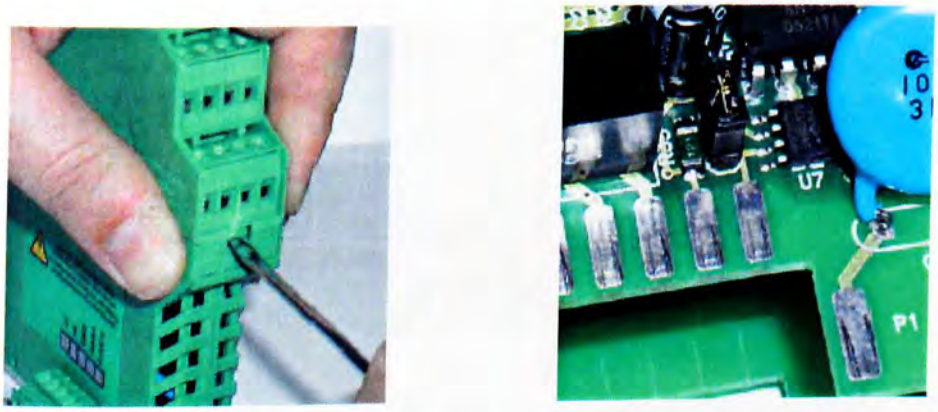
Υπάρχουν δυο διαθέσιμοι ρυθμοί μετάδοσης δεδομένων , οι οποίοι επιλέγονται μηχανικά μέσω του ογδόου διακόπτη του PCB . Οι δυο διαθεσίμου ρυθμοί μετάδοσης είναι 125 kbit/s που αντιστοιχούν στην θέση OFF (κατάσταση 0) του διακόπτη οκτώ και 250Kbit/s που αντιστοιχεί στην θέσηON (κατάσταση 1) του διακόπτης οκτώ. Επιπλέον υπάρχουν ρυθμοί μετάδοσης , οι οποίοι υποστηρίζονται από τοTMPCan-2 και επιλέγονται από λογισμικό μέσω πρωτοκόλλου LSS.

Bit Rate (kBit/s)	Maximum Line Length (m)
1000	25
800	50
500	100
250	250
125	500
50	1000
20	2500
10	5000

Εικόνα 5.19 Προτεινόμενοι Ρυθμοί μετάδοσης δεδομένων σύμφωνα με το μήκος του καλωδίου του κόμβου.

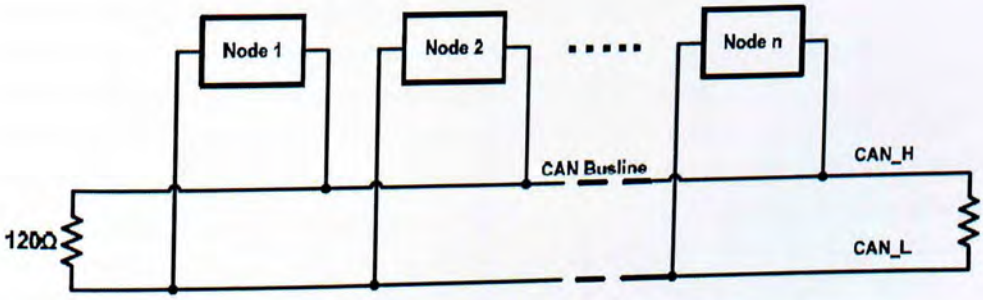
5.5.2.4 Τερματισμός του δίαυλου.

Όλα τα προϊόντα της σεις α E-Logix 2K έχουν ενσωματωμένη τερματική αντίσταση ,η όποια ενεργοποιείτε με την χρήση μια γέφυρας (jumper). Για να την εντοπίσουμε και να την τοποθετήσουμε εξ' αρχής κάνουμε τα εξής :



Εικόνα 5.20ΑριστεράΆνοιγμακαλύμματος – Δεξιά εντοπισμός jumper

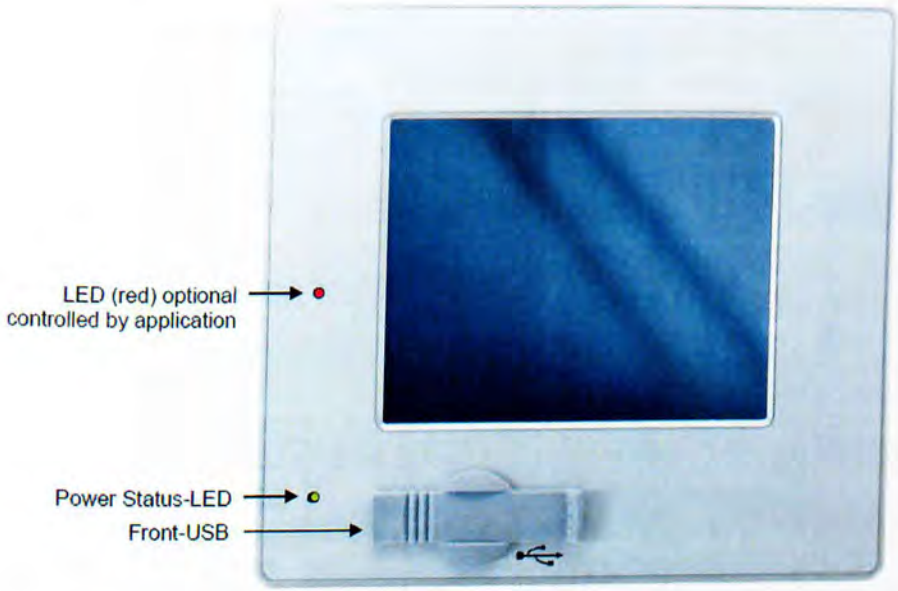
Η γέφυρα τερματισμού (jumper) πρέπει να τοποθετηθεί μόνο στην τελευταία συσκευή του δίαυλου. Για το ορθό τερματισμό διαύλου συμφώνα με την τοπολογία του ISO11898-2.



Εικόνα 5.21 Παράδειγμα εγκατάστασης και τερματισμού κόμβου.

5.5.3ΚεντρικόςΕλεγκτής

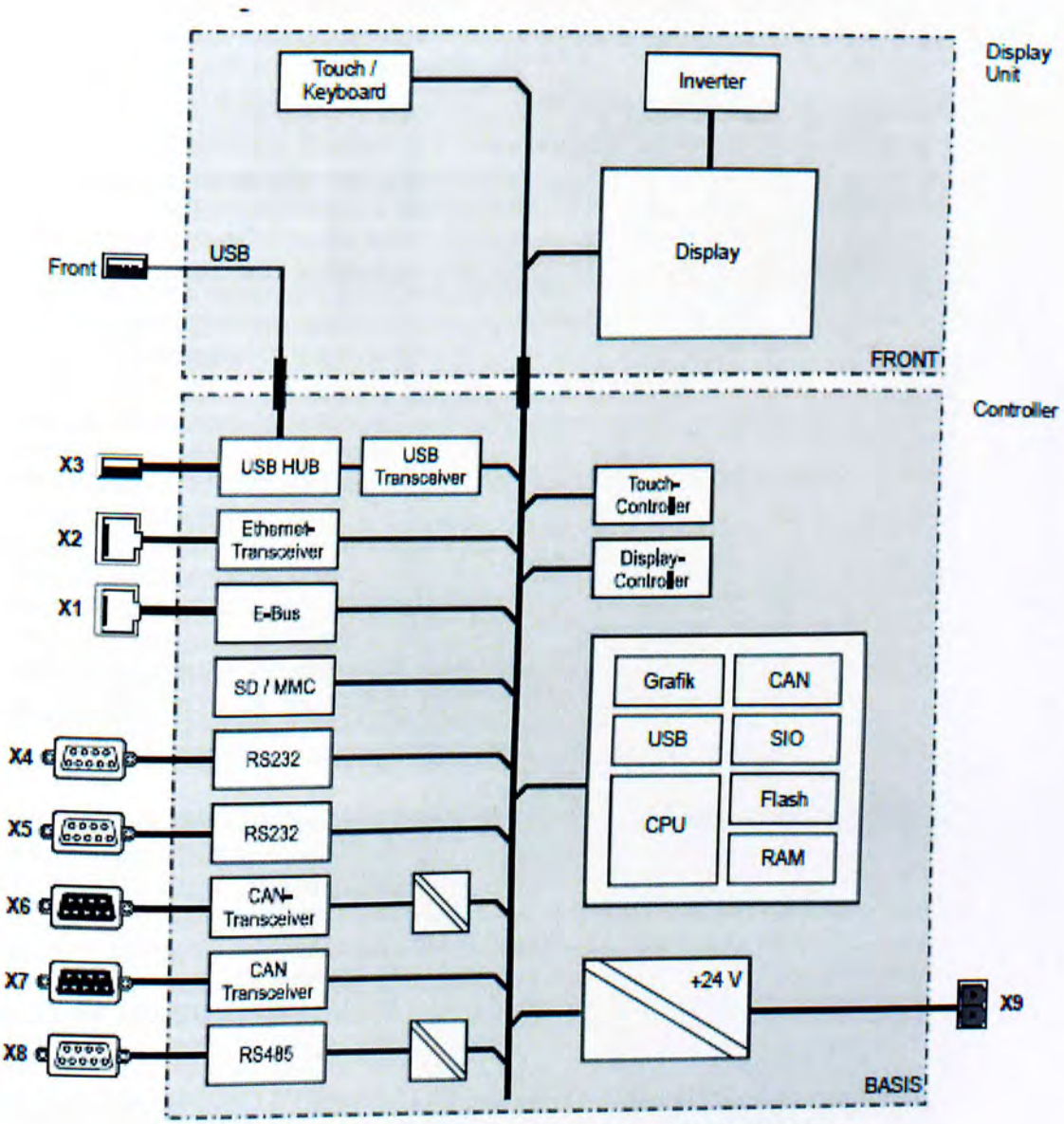
5.5.3.1 Γενικά χαρακτηριστικά.



Ο ελεγκτής μας είναι μια συσκευή Έλεγχου πραγματικού χρόνου με δυνατότητα απεικόνισης δεδομένων και μιμικών διαγραμμάτων. Ο προγραμματισμός του γίνεται μέσω της γλώσσας Ceite με το αναπτυξιακό πρόγραμμα CoDESYS. Τοποθετείται στην μπροστινή όψη ενός ηλεκτρολογικού πάνελ και έχειαντοχές στις δυσμενής συνθήκες των βιομηχανικών περιοχών. Η συντήρηση που απαιτείται είναι ελάχιστη καθώς δεν χρειάζεται ιδιαίτερο εξαερισμό.

Είναι εφοδιασμένος με δυο τύπους επεξεργαστών στα 266Mhz και 400MHz. Διαθέτει μια οθόνη αφής των 5.7 ιντσών έγχρωμη και υδατοστεγή IP65.Έχει θύρα επικοινωνίας ETHERNET 10/100 Mbit/s με πλήρη υποστήριξη των πρωτοκόλλων επικοινωνίας TCP/IP που του προσφέρουν μεγάλη ευελιξία στην σύνδεση του με κεντρικούς υπολογιστές και την ανταλλαγή δεδομένων. Είναι εφοδιασμένος με δύο θύρες USBμε τις όποιεςμπορεί να συνδεθεί με διαφορά περιφερικά όπως στίκ μνήμης USB για την ενημέρωση του προγράμματα του ή την εξαγωγή δεδομένων. Η ΘύραεπικοινωνίαςCANBUS που διαθέτει βρίσκεται στην πίσω όψη της συσκευής μας και έχει ρυθμό μετάδοσης το 1Mbit/s.Τέλος έχει τρείς σειριακές πόρτες, δύο RS232 και μια RS485.

5.5.3.2 BLOCK ΔΙΑΓΡΑΜΜΑ.



5.5.3.3 ΤεχνικάΧαρακτηριστικά

Dialog-Controller		
Display	**1/4 VGA**	**VGA**
Diagonal measurement	5.7"	
Resolution	320 x 240 pixels (1/4 VGA)	640 x 480 pixel
Colors	Monochrome (STN): 4 Color (CSTN): 256 (8 bits / pixel) Color TFT 256 (8 bits / pixel)	Color TFT: 65536 (16 bits / pixel)
CPU, application memory		
CPU	Freescale PowerPC 266 MHz	400 MHz
Program memory (Flash)	16 MB onboard / 8 MB for application	32 MB onboard / 24 MB for application
Program memory and data memory (RAM)	64 MB onboard / 32 MB for application	128 MB onboard / 96 MB for application
Retain memory	16 KB	
Development environment	CP1131 (CoDeSys 2.3)	
Dimensions and weight		
Dimensions (WxHxD [mm])	194 x 172 x 52 (+6 mm front panel)	
Weight	approx. 1.5 kg	
Operating conditions		
Ambient temperature range	0 °C to 55 °C (if installation rule is adhered to)	
Relative humidity	max. 85 %, non-condensing (monochrome: max. 75 %)	
Transport, storage		
Ambient temperature range	-20 °C to +70 °C	
Relative humidity	max. 85 %, non-condensing (monochrome: max. 75 %)	
Shock resistance		
Vibration	Sinusoidal (EN 60068-2-6) test: Fc 10 ... 150 Hz, 1 G (operation mode)	
Shock resistance	15 G (approx. 150 m/s²); duration: 10 ms; semi-sinusoidal (EN 60068-2-27) test: Ea	

EMC, protection type	
Interference emission	EN 61000-6-4, industrial environments
Interference resistance	EN 61000-6-2, industrial environments
Protection class	III
Insulating resistance	EN 61131-2; DC 500 V test voltage
Protection type	IP20 (front: IP65)
Power supply (24 V power pack)	
Supply voltage	+24 VDC (-15% / +20%) SELV max. alternating component: 5%
Current consumption	Typically 1.0 A; max. 2.0 A at +24 VDC Fusing depending on the I/O load, max. 12A
Polarity reversal protection	Yes
Potential isolation	Yes, between CAN bus and I/Os

Ethernet port	
Number / interface type	1x 10/100 Base T
Connection method	RJ45
USB ports	
Number / interface types	1 x host USB, Rev. 1.1 (on rear) 1 x host USB, Rev. 1.1 (on front, on Prime Interface version)
Plug/unplug cycles	max. 1000
CAN bus ports	
Number / interface types	2x standard CAN ISO 11898 1x standard CAN ISO 11898 on Basic Interface
Potential isolation	CAN0 (X6) potential isolated (Prime Interface only)
Transmission rate	max. 1 MBit/s
Terminating resistor	Switchable
Serial port	
Number / interface types	2x RS232 (1x RS232 on Basic Interface) 1x RS485
Potential isolation	RS485 (X8) potential isolated (Prime Interface only)
Terminating resistor	Switchable with the RS485
E-bus port	
Interface types	I/O expansion bus for up to 7 E-bus subscribers
Expansion slots	
Number / interface types	3 slots for 3 I/O modules or 2 I/O modules and one AnyBus® module (Prime Interface only)
Further features	
Real-time clock	Yes, battery backed
SD-Card	1 SD-Card slot or RS232 (X5) (Prime Interface only)

5.6 Το Πρόγραμμα

Το πρόγραμμα μας αποτελείται από τρία τμήματα. Το κυρίως πρόγραμμα PLC_PRG , τη συνθήκη που διαβάζει τις πληροφορίες που του στέλνονται μέσω του CANBUS κάθε δευτερόλεπτο EverySecond καθώς και μια τρίτη συνθήκη που θα χρησιμοποιηθεί στο μιμικό πάνελ του ελεκτή μας Blink.

Κάθε τμήμα χωρίζεται σε δύο επιμέρους. Το τμήμα που δηλώνουμε τις μεταβλητές καθώς και τον τύπο τους βρίσκεται στο άνω μέρος του παράθυρου μας και στο τμήμα που γραφούμε τον κύριο κορμό του προγράμματος. Εάν μια μεταβλητή μας δεν έχει οριστεί τότε το πρόγραμμα όταν κάνουμε Debugging θα μας βγάλει σφάλμα.

5.6.1PLC_PRG

Στο πρώτο σκέλος όπως έχουμε αναφέρει παραπάνω αναγράφονται οι μεταβλητές του συστήματος μας καθώς και οι τύποι τους.

```
PROGRAM PLC_PRG
VAR CONSTANT
    MaxD:INT:=50;
    MaxDEnb:INT:=200;
END_VAR
VAR
    SettHideShow: BOOL:=TRUE;
    blink1:BOOL;
    visib1:BOOL;
    ACK:BOOL;
```

Εδώ ορίζουμε από ποιές εισόδους των καρτών θα διαβάζουμε την επιθυμητή θερμοκρασία καθώς και τον τύπο της μεταβλητής.

```
    temps11 AT %IW0 :INT;
    temps12 AT %IW1 :INT;
    temps13 AT %IW2 :INT;
    temps14 AT %IW3 :INT;
    temps15 AT %IW4 :INT;
    temps16 AT %IW5 :INT;
    temps1TC AT %IW6 :INT;
    temps1TC2 AT %IW7 :INT;
    temps1TC3 AT %IW8 :INT;
    temps1TC4 AT %IW9 :INT;
    temp1Mean :INT;
```

```
(*      temps21 AT %IW8 :INT;
        temps22 AT %IW9 :INT;
        temps23 AT %IW10 :INT;
        temps24 AT %IW11 :INT;
        temps25 AT %IW12 :INT;
        temps26 AT %IW13 :INT;
        temps2TC AT %IW14 :INT;
        temp2Mean :INT;*)
```

Εδώ ορίζουμε από ποιά έξοδο των καρτών θα διαβάζουμε την επιθυμητή θερμοκρασία καθώς και τον τύπο της μεταβλητής

```
        OUT AT %QB0:BYTE;
        temp11,temp12,temp13,temp14,temp15,temp16,temp1TC,temp1TC2,temp1TC3,temp1TC4:INT;
        temp11DM,temp12DM,temp13DM,temp14DM,temp15DM,temp16DM,temp1TCDM,temp1TC2DM,temp1TC3DM,temp1TC4DM:INT;
        MeanD1,MeanD2,MeanD3,MeanD4,MeanD5,MeanD6:BOOL;
(*      temp21,temp22,temp23,temp24,temp25,temp26,temp2TC:INT;*)
        AlarmME1HH,AlarmME1H,AlarmME2HH,AlarmME2H,MeanD :BOOL;
```

Εδώ ορίζουμε κάποιους χρωματισμούς που θα βλέπουμε στο μιμικό ανάλογα με την κατάσταση του συναγερμού alarm που λαμβάνουμε.

```
        ValueBadSensor:INT :=-1;
        ColorHAlarm:DWORD :=16#0000FFFF;
        ColorHHAlarm:DWORD :=16#000000FF;
        ColorNormal:DWORD :=16#00FFFFFF;
        ColorBadSensor:DWORD :=16#00C0C0C0;
        ColorBadNode:DWORD :=16#00404040;
        Color11,Color12,Color13,Color14,Color15,Color16,Color1TC,Color1TC2,Color1TC3,Color1TC4,Color1Mean:DWORD :=16#00FFFFFF;
```

```
        Color11B,Color12B,Color13B,Color14B,Color15B,Color16B,Color1TCB,Col
or1TC2B,Color1TC3B,Color1TC4B,Color1MeanB:DWORD :=16#00FFFFFF;

        (*Color21,Color22,Color23,Color24,Color25,Color26,Color2TC,Color2Mean
:DWORD :=16#00FFFFFF;*)

        NodeStatus:ARRAY[0..MAX_NODEINDEX] OF DWORD;

        istatus: BYTE;

        BadNodesStatus:BOOL;

        hide, hidensw:BOOL;

        MeanD1B,MeanD2B,MeanD3B,MeanD4B,MeanD5B,MeanD6B:
BOOL:=FALSE;

        MeanD1Blnk,MeanD2Blnk,MeanD3Blnk,MeanD4Blnk,MeanD5Blnk,MeanD
6Blnk: BOOL:=FALSE;

END_VAR

VAR RETAIN


        HHAlarm11,HHAlarm12,HHAlarm13,HHAlarm14,HHAlarm15,HHAlarm16,
HHAlarm1TC,HHAlarm1TC2,HHAlarm1TC3,HHAlarm1TC4,HHAlarm1Mean:INT;

        (*HHAlarm21,HHAlarm22,HHAlarm23,HHAlarm24,HHAlarm25,HHAlarm2
6,HHAlarm2TC,HHAlarm2Mean: INT;*)

        HAlarm11,HAlarm12,HAlarm13,HAlarm14,HAlarm15,HAlarm16,HAlarm1T
C,HAlarm1TC2,HAlarm1TC3,HAlarm1TC4,HAlarm1Mean: INT;

        (*HAlarm21,HAlarm22,HAlarm23,HAlarm24,HAlarm25,HAlarm26,HAlarm
2TC,HAlarm2Mean: INT;*)

DefaultSettings,BadStatus:BOOL;


END_VAR
```

Το κυρίως σκέλος του προγράμματος

```
(*IF blink1 THEN visib1:=NOT visib1; END_IF;

IF ACK THEN visib1:=TRUE;

                    blink1:=FALSE;

END_IF;*)
```

Εδώ αρχικά ανιχνεύουμε την κατάσταση λειτουργιάς του αισθητήρα s e περίπτωση που μας στέλνει τιμές που είναι εντελώς παραλογές με εκείνες που περιμένουμε.

```
IF temps11=-5472 THEN temp11:=ValueBadSensor; ELSE
temp11:=temps11;END_IF

IF temps12=-5472 THEN temp12:=ValueBadSensor; ELSE
temp12:=temps12;END_IF

IF temps13=-5472 THEN temp13:=ValueBadSensor; ELSE
temp13:=temps13;END_IF

IF temps14=-5472 THEN temp14:=ValueBadSensor; ELSE
temp14:=temps14;END_IF

IF temps15=-5472 THEN temp15:=ValueBadSensor; ELSE
temp15:=temps15;END_IF

IF temps16=-5472 THEN temp16:=ValueBadSensor; ELSE
temp16:=temps16;END_IF

IF temps1TC=-5472 THEN temp1TC:=ValueBadSensor; ELSE
temp1TC:=temps1TC;END_IF

IF temps1TC2=-5472 THEN temp1TC2:=ValueBadSensor; ELSE
temp1TC2:=temps1TC2;END_IF

IF temps1TC3=-5472 THEN temp1TC3:=ValueBadSensor; ELSE
temp1TC3:=temps1TC3;END_IF

IF temps1TC4=-5472 THEN temp1TC4:=ValueBadSensor; ELSE
temp1TC4:=temps1TC4;END_IF

(*

IF temps21=-5472 THEN temp21:=ValueBadSensor; ELSE
temp21:=temps21;END_IF
```

```
IF temps22=-5472 THEN temp22:=ValueBadSensor; ELSE
temp22:=temps22;END_IF

IF temps23=-5472 THEN temp23:=ValueBadSensor; ELSE
temp23:=temps23;END_IF

IF temps24=-5472 THEN temp24:=ValueBadSensor; ELSE
temp24:=temps24;END_IF

IF temps25=-5472 THEN temp25:=ValueBadSensor; ELSE
temp25:=temps25;END_IF

IF temps26=-5472 THEN temp26:=ValueBadSensor; ELSE
temp26:=temps26;END_IF

IF temps2TC=-5472 THEN temp2TC:=ValueBadSensor; ELSE
temp2TC:=temps2TC;END_IF

*)
```

Εδώ θα ανιχνεύσουμε την περίπτωση που ένας κύλινδρος του κινητήρα μας υπολειτουργεί ή υπολειτουργεί. Έτσι συγκρίνουμε την μέση θερμοκρασία και των 6 κυλίνδρων με τον καθένα ξεχωριστά και στην συνεχεία ανάλογα με το ποσοστό που έχουμε ορίσει κρίνουμε και τα αποτελέσματα.

```
temp1Mean:=(temp11+temp12+temp13+temp14+temp15+temp16)/6;

IF temp11>MaxDEnb THEN temp11DM:=ABS(temp1Mean-temp11); ELSE
temp11DM:=0; END_IF;

MeanD1:=temp11DM>MaxD;

IF MeanD1 AND NOT MeanD1B THEN blinkenablem1:=TRUE; END_IF;

MeanD1B:=MeanD1;

MeanD1Blnk:=NOT ((PLC_PRG.MeanD1 AND NOT BlinkEnableM1) OR
(BlinkEnableM1 AND BLINK.blinkm1));

IF temp12>MaxDEnb THEN temp12DM:=ABS(temp1Mean-temp12); ELSE
temp12DM:=0; END_IF;

MeanD2:=temp12DM>MaxD;

IF MeanD2 AND NOT MeanD2B THEN blinkenablem2:=TRUE; END_IF;

MeanD2B:=MeanD2;

MeanD2Blnk:=NOT ((PLC_PRG.MeanD2 AND NOT BlinkEnableM2) OR
(BlinkEnableM2 AND BLINK.blinkm2));
```

```
IF temp13>MaxDEnb THEN temp13DM:=ABS(temp1Mean-temp13); ELSE
temp13DM:=0; END_IF;

MeanD3:=temp13DM>MaxD;

IF MeanD3 AND NOT MeanD3B THEN blinkenablem3:=TRUE; END_IF;

MeanD3B:=MeanD3;

MeanD3Blnk:=NOT ((PLC_PRG.MeanD3 AND NOT BlinkEnableM3) OR
(BlinkEnableM3 AND BLINK.blinkm3));

IF temp14>MaxDEnb THEN temp14DM:=ABS(temp1Mean-temp14); ELSE
temp14DM:=0; END_IF;

MeanD4:=temp14DM>MaxD;

IF MeanD4 AND NOT MeanD4B THEN blinkenablem4:=TRUE; END_IF;

MeanD4B:=MeanD4;

MeanD4Blnk:=NOT ((PLC_PRG.MeanD4 AND NOT BlinkEnableM4) OR
(BlinkEnableM4 AND BLINK.blinkm4));


IF temp15>MaxDEnb THEN temp15DM:=ABS(temp1Mean-temp15); ELSE
temp15DM:=0; END_IF;

MeanD5:=temp15DM>MaxD;

IF MeanD5 AND NOT MeanD5B THEN blinkenablem5:=TRUE; END_IF;

MeanD5B:=MeanD5;

MeanD5Blnk:=NOT ((PLC_PRG.MeanD5 AND NOT BlinkEnableM5) OR
(BlinkEnableM5 AND BLINK.blinkm5));


IF temp16>MaxDEnb THEN temp16DM:=ABS(temp1Mean-temp16); ELSE
temp16DM:=0; END_IF;

MeanD6:=temp16DM>MaxD;

IF MeanD6 AND NOT MeanD6B THEN blinkenablem6:=TRUE; END_IF;

MeanD6B:=MeanD6;

MeanD6Blnk:=NOT ((PLC_PRG.MeanD6 AND NOT BlinkEnableM6) OR
(BlinkEnableM6 AND BLINK.blinkm6));
```

Εδώ ορίζουμε πως όταν όλα λειτουργούν κανονικά το σύστημα μας θα βρίσκεται σε ηρεμία.

```
(*temp2Mean:=(temp21+temp22+temp23+temp24+temp25+temp26)/6;*)

IF AlarmME1HH THEN OUT.0:=TRUE;  AlarmME1HH:=0;END_IF;

IF AlarmME1H THEN OUT.1:=TRUE; AlarmME1H:=0;END_IF;

(*IF AlarmME2HH THEN OUT.2:=TRUE; AlarmME2HH:=0;END_IF;

IF AlarmME2H THEN OUT.3:=TRUE; AlarmME2H:=0;END_IF;*)

IF BadStatus THEN OUT.4:=TRUE; BadStatus:=0;END_IF;

IF MeanD THEN OUT.5:=TRUE;MeanD:=FALSE; END_IF;
```

Από εδώ ξεκινά η δήλωση των συναγερμών που έχουμε ορίσει παραπάνω κάθε ποτέ θα εμφανίζονται και τι χρωματισμό θα πάρει στο μιμικό μας πρόγραμμα ανάλογα με το είδος του συναγερμού. Επαναλαμβάνουμε για κάθε είσοδο αντίστοιχα.

```
IFPLC_PRG.temps11=-5472 THENColor11:=ColorBadSensor;
        ELSE
                IF PLC_PRG.temp11>PLC_PRG.HHAlarm11 THEN
Color11:=ColorHHAlarm;
                        ELSE
                                IF PLC_PRG.temp11>PLC_PRG.HAlarm11 THEN
Color11:=ColorHAlarm;
                                        ELSE Color11:=ColorNormal;
                                END_IF;
                END_IF;
END_IF;
IF Color11<>ColorNormal AND Color11B=ColorNormal THEN
blinkenable11:=TRUE; END_IF;
Color11B:=Color11;
```

```
IF PLC_PRG.temps12=-5472 THEN Color12:=ColorBadSensor;
        ELSE
                IF PLC_PRG.temp12>PLC_PRG.HHAlarm12 THEN
Color12:=ColorHHAlarm;
                        ELSE
IF PLC_PRG.temp12>PLC_PRG.HAlarm12 THEN Color12:=ColorHAlarm;
                                        ELSE Color12:=ColorNormal;
                                END_IF;
                END_IF;
END_IF;
IF Color12<>ColorNormal AND Color12B=ColorNormal THEN
blinkenable12:=TRUE; END_IF;
Color12B:=Color12;
IF PLC_PRG.temps13=-5472 THEN Color13:=ColorBadSensor;
        ELSE
                IF PLC_PRG.temp13>PLC_PRG.HHAlarm13 THEN
Color13:=ColorHHAlarm;
                        ELSE
                                IF PLC_PRG.temp13>PLC_PRG.HAlarm13 THEN
Color13:=ColorHAlarm;
                                        ELSE Color13:=ColorNormal;
                                END_IF;
                END_IF;
END_IF;
IF Color13<>ColorNormal AND Color13B=ColorNormal THEN
blinkenable13:=TRUE; END_IF;
Color13B:=Color13;

IF PLC_PRG.temps14=-5472 THEN Color14:=ColorBadSensor;
```

```
        ELSE
                IF PLC_PRG.temp14>PLC_PRG.HHAlarm14 THEN
Color14:=ColorHHAlarm;
                    ELSE
IF PLC_PRG.temp14>PLC_PRG.HAlarm14 THEN Color14:=ColorHAlarm;
                                ELSE Color14:=ColorNormal;
                            END_IF;
                END_IF;
END_IF;
IF Color14<>ColorNormal AND Color14B=ColorNormal THEN
blinkenable14:=TRUE; END_IF;
Color14B:=Color14;
IF PLC_PRG.temps15=-5472 THEN Color15:=ColorBadSensor;
        ELSE
                IF PLC_PRG.temp15>PLC_PRG.HHAlarm15 THEN
Color15:=ColorHHAlarm;
                    ELSE
                            IF PLC_PRG.temp15>PLC_PRG.HAlarm15 THEN
Color15:=ColorHAlarm;
                                ELSE Color15:=ColorNormal;
                            END_IF;
                END_IF;
END_IF;
IF Color15<>ColorNormal AND Color15B=ColorNormal THEN
blinkenable15:=TRUE; END_IF;
Color15B:=Color15;
IF PLC_PRG.temps16=-5472 THEN Color16:=ColorBadSensor;
        ELSE
```

```
            IF PLC_PRG.temp16>PLC_PRG.HHAlarm16 THEN
Color16:=ColorHHAlarm;
                ELSE
                    IF PLC_PRG.temp16>PLC_PRG.HAlarm16 THEN
Color16:=ColorHAlarm;
                        ELSE Color16:=ColorNormal;
                    END_IF;
            END_IF;
END_IF;
IF Color16<>ColorNormal AND Color16B=ColorNormal THEN
blinkenable16:=TRUE; END_IF;
Color16B:=Color16;
IF PLC_PRG.temps1TC=-5472 THEN Color1TC:=ColorBadSensor;
    ELSE
        IF PLC_PRG.temp1TC>PLC_PRG.HHAlarm1TC THEN
Color1TC:=ColorHHAlarm;
            ELSE
                IF PLC_PRG.temp1TC>PLC_PRG.HAlarm1TC THEN
Color1TC:=ColorHAlarm;
                    ELSE Color1TC:=ColorNormal;
                END_IF;
        END_IF;
END_IF;
IF Color1TC<>ColorNormal AND Color1TCB=ColorNormal THEN
blinkenable1TC:=TRUE; END_IF;
Color1TCB:=Color1TC;
IF PLC_PRG.temps1TC2=-5472 THEN Color1TC2:=ColorBadSensor;
    ELSE
        IF PLC_PRG.temp1TC2>PLC_PRG.HHAlarm1TC2 THEN
Color1TC2:=ColorHHAlarm;
```

```
                    ELSE

                        IF PLC_PRG.temp1TC2>PLC_PRG.HAlarm1TC2
THEN Color1TC2:=ColorHAlarm;

                            ELSE Color1TC2:=ColorNormal;

                        END_IF;

                END_IF;

END_IF;

IF Color1TC2<>ColorNormal AND Color1TC2B=ColorNormal THEN
blinkenable1TC2:=TRUE; END_IF;

Color1TC2B:=Color1TC2;

IF PLC_PRG.temps1TC3=-5472 THEN Color1TC3:=ColorBadSensor;

    ELSE

        IF PLC_PRG.temp1TC3>PLC_PRG.HHAlarm1TC3 THEN
Color1TC3:=ColorHHAlarm;

                ELSE

                    IF PLC_PRG.temp1TC3>PLC_PRG.HAlarm1TC3
THEN Color1TC3:=ColorHAlarm;

                        ELSE Color1TC3:=ColorNormal;

                    END_IF;

            END_IF;

END_IF;

IF Color1TC3<>ColorNormal AND Color1TC3B=ColorNormal THEN
blinkenable1TC3:=TRUE; END_IF;

Color1TC3B:=Color1TC3;


IF PLC_PRG.temps1TC4=-5472 THEN Color1TC4:=ColorBadSensor;

    ELSE

        IF PLC_PRG.temp1TC4>PLC_PRG.HHAlarm1TC4 THEN
Color1TC4:=ColorHHAlarm;
```

```
                    ELSE
                        IF PLC_PRG.temp1TC4>PLC_PRG.HAlarm1TC4
THEN Color1TC4:=ColorHAlarm;
                            ELSE Color1TC4:=ColorNormal;
                        END_IF;
                END_IF;
END_IF;
IF Color1TC4<>ColorNormal AND Color1TC4B=ColorNormal THEN
blinkenable1TC4:=TRUE; END_IF;
Color1TC4B:=Color1TC4;

IF PLC_PRG.temp1Mean>PLC_PRG.HHAlarm1Mean THEN
Color1Mean:=ColorHHAlarm;
    ELSE
        IF PLC_PRG.temp1Mean>PLC_PRG.HAlarm1Mean THEN
Color1Mean:=ColorHAlarm;
            ELSE Color1Mean:=ColorNormal;
        END_IF;
END_IF;
IF Color1Mean<>ColorNormal AND Color1MeanB=ColorNormal THEN
blinkenable1Mean:=TRUE; END_IF;
Color1MeanB:=Color1Mean;

(*
IF PLC_PRG.temps21=-5472 THEN Color21:=ColorBadSensor;
    ELSE
        IF PLC_PRG.temp21>PLC_PRG.HHAlarm21 THEN
Color21:=ColorHHAlarm;
            ELSE
```

```
                    IF PLC_PRG.temp21>PLC_PRG.HAlarm21 THEN
Color21:=ColorHAlarm;
                        ELSE Color21:=ColorNormal;
                    END_IF;
                END_IF;
            END_IF;
            IF PLC_PRG.temps22=-5472 THEN Color22:=ColorBadSensor;
                ELSE
                    IF PLC_PRG.temp22>PLC_PRG.HHAlarm22 THEN
Color22:=ColorHHAlarm;
                        ELSE
                            IF PLC_PRG.temp22>PLC_PRG.HAlarm22 THEN
Color22:=ColorHAlarm;
                                ELSE Color22:=ColorNormal;
                            END_IF;
                    END_IF;
            END_IF;
            IF PLC_PRG.temps23=-5472 THEN Color23:=ColorBadSensor;
                ELSE
                    IF PLC_PRG.temp23>PLC_PRG.HHAlarm23 THEN
Color23:=ColorHHAlarm;
                        ELSE
                            IF PLC_PRG.temp23>PLC_PRG.HAlarm23 THEN
Color23:=ColorHAlarm;
                                ELSE Color23:=ColorNormal;
                            END_IF;
                    END_IF;
            END_IF;
            IF PLC_PRG.temps24=-5472 THEN Color24:=ColorBadSensor;
```

```
        ELSE
                IF PLC_PRG.temp24>PLC_PRG.HHAlarm24 THEN
Color24:=ColorHHAlarm;
                    ELSE
                        IF PLC_PRG.temp24>PLC_PRG.HAlarm24 THEN
Color24:=ColorHAlarm;
                            ELSE Color24:=ColorNormal;
                        END_IF;
                END_IF;
END_IF;
IF PLC_PRG.temps25=-5472 THEN Color25:=ColorBadSensor;
        ELSE
                IF PLC_PRG.temp25>PLC_PRG.HHAlarm25 THEN
Color25:=ColorHHAlarm;
                    ELSE
                        IF PLC_PRG.temp25>PLC_PRG.HAlarm25 THEN
Color25:=ColorHAlarm;
                            ELSE Color25:=ColorNormal;
                        END_IF;
                END_IF;
END_IF;
IF PLC_PRG.temps26=-5472 THEN Color26:=ColorBadSensor;
        ELSE
                IF PLC_PRG.temp26>PLC_PRG.HHAlarm26 THEN
Color26:=ColorHHAlarm;
                    ELSE
                        IF PLC_PRG.temp26>PLC_PRG.HAlarm26 THEN
Color26:=ColorHAlarm;
                            ELSE Color26:=ColorNormal;
```

```
                    END_IF;
            END_IF;
END_IF;
IF PLC_PRG.temps2TC=-5472 THEN Color2TC:=ColorBadSensor;
    ELSE
        IF PLC_PRG.temp2TC>PLC_PRG.HHAlarm2TC THEN
Color2TC:=ColorHHAlarm;
            ELSE
                IF PLC_PRG.temp2TC>PLC_PRG.HAlarm2TC THEN
Color2TC:=ColorHAlarm;
                    ELSE Color2TC:=ColorNormal;
                END_IF;
        END_IF;
END_IF;
    IF PLC_PRG.temp2Mean>PLC_PRG.HHAlarm2Mean THEN
Color2Mean:=ColorHHAlarm;
        ELSE
            IF PLC_PRG.temp2Mean>PLC_PRG.HAlarm2Mean THEN
Color2Mean:=ColorHAlarm;
                ELSE Color2Mean:=ColorNormal;
            END_IF;
    END_IF;*)
FOR istatus:=0 TO 2 DO
    NodeStatus[istatus]:=pCanOpenNode[istatus].nStatus;
(*  NodeNumber[istatus]:=pCanOpenNode[istatus].ucNodeNr;*)
END_FOR;
```

Ορίζουμε ποτέ θα πάρουμε τον κάθε alarm όταν ξεπεράσει την θερμοκρασία που του ορίζουμε

.

```
IF PLC_PRG.DefaultSettings THEN
        PLC_PRG.DefaultSettings:=0;
        PLC_PRG.HHAlarm11:=430;
        PLC_PRG.HHAlarm12:=430;
        PLC_PRG.HHAlarm13:=430;
        PLC_PRG.HHAlarm14:=430;
        PLC_PRG.HHAlarm15:=430;
        PLC_PRG.HHAlarm16:=430;
        PLC_PRG.HHAlarm1TC:=430;
        PLC_PRG.HHAlarm1TC2:=430;
        PLC_PRG.HHAlarm1TC3:=350;
        PLC_PRG.HHAlarm1TC4:=350;
        PLC_PRG.HHAlarm1Mean:=430;
        (*PLC_PRG.HHAlarm21:=450;
        PLC_PRG.HHAlarm22:=450;
        PLC_PRG.HHAlarm23:=450;
        PLC_PRG.HHAlarm24:=450;
        PLC_PRG.HHAlarm25:=450;
        PLC_PRG.HHAlarm26:=450;
        PLC_PRG.HHAlarm2TC:=450;
        PLC_PRG.HHAlarm2Mean:=450;*)
        PLC_PRG.HAlarm11:=400;
        PLC_PRG.HAlarm12:=400;
        PLC_PRG.HAlarm13:=400;
```

```
        PLC_PRG.HAlarm14:=400;
        PLC_PRG.HAlarm15:=400;
        PLC_PRG.HAlarm16:=400;
        PLC_PRG.HAlarm1TC:=410;
        PLC_PRG.HAlarm1TC2:=410;
        PLC_PRG.HAlarm1TC3:=325;
        PLC_PRG.HAlarm1TC4:=325;
        PLC_PRG.HAlarm1Mean:=400;
(*      PLC_PRG.HAlarm21:=425;
        PLC_PRG.HAlarm22:=425;
        PLC_PRG.HAlarm23:=425;
        PLC_PRG.HAlarm24:=425;
        PLC_PRG.HAlarm25:=425;
        PLC_PRG.HAlarm26:=425;
        PLC_PRG.HAlarm2TC:=425;
        PLC_PRG.HAlarm2Mean:=425;*)
END_IF;
```

5.6.2 EverySecond

Ο κάθε κόμβος όπως έχουμε αναφέρει στο δίαυλο του CANBUS έχει ένα αναγνωριστικό ID που το κάνει να ξεχωρίσει. Στο πρόγραμμα μας τα ID τα έχουμε ορίσει όταν συνθέταμε τα στοιχεία που αποτελούν το υλικό της εφαρμογής μας. Κάθε δευτερόλεπτο παίρνουμε πληροφορίες από τις κάρτες μας για την κατάσταση των αισθητήρων. Όμως κι εμείς οι ίδιοι ζητάμε πληροφορίες για την κατάσταση του των ίδιων των καρτών. Με το παρακάτω υποπρόγραμμα .

PROGRAMEverySecond

Έχουμε ήδη κατασκευάσει μια μεταβλητή την TableStruct η οποία είναι ένας πινάκας που βλέπουμε τα αποτελέσματα της κατάστασης των κόμβων μας. Στην μεταβλητή αύτη έχουμε οπτική πρόσβαση στο μιμικό μας πάνελ.

```
VAR
        i: INT;
        Table:ARRAY [0..MAX_NODEINDEX] OF TableStruct;
END_VAR


Το κύριο σκέλος.


IF CurrentVisu='INFOPAGE' THEN


        FOR i:=0 TO MAX_NODEINDEX DO
                Table[i].NodeIndex:=pCanOpenNode[i].ucNodeNr;
                Table[i].Status:=pCanOpenNode[i].nStatus;

        Table[i].ValueStr:=BYTE_TO_STRING(pCanOpenPDO_Rx[pCanOpenNode
[i].wFirstRxPDOIndex].Buffer.pData[0]);
        END_FOR;


END_IF;
```

5.6.3 Blink

Στο μιμικό πάνελ μας όταν προκύπτει ένας συναγερμός για να μπορέσουμε να τον αντιληφθούμε έχουμε δημιουργήσει εικονικές λυχνίες οι οποίες εμφανίζονται σε αυτό. Γι' αυτό το σκοπό χρησημοποιούμε την παρακάτω συνθήκη.

```
PROGRAMBLINK
VAR
    blink11,blink12,blink13,blink14,blink15,blink16,blink1Mean: BOOL;
    blink1TC,blink1TC2,blink1TC3,blink1TC4:BOOL;

    resetenable: BOOL;
    blinkreset: BOOL;
    blinkm1,blinkm2,blinkm3,blinkm4,blinkm5,blinkm6: BOOL:=FALSE;
END_VAR
```

Το Κύριοσκέλος.

```
IF blinkreset THEN
    resetenable:=FALSE;
    blinkreset:=FALSE;
    blinkenable11:=FALSE;
    blinkenable12:=FALSE;
    blinkenable13:=FALSE;
    blinkenable14:=FALSE;
    blinkenable15:=FALSE;
    blinkenable16:=FALSE;
    blinkenable1Mean:=FALSE;
    blinkenable1TC:=FALSE;
    blinkenable1TC2:=FALSE;
    blinkenable1TC3:=FALSE;
```

```
        blinkenable1TC4:=FALSE;
        blinkenablem1:=FALSE;
        blinkenablem2:=FALSE;
        blinkenablem3:=FALSE;
        blinkenablem4:=FALSE;
        blinkenablem5:=FALSE;
        blinkenablem6:=FALSE;
END_IF;

IF blinkenable11 THEN blink11:=NOT blink11; resetenable:=TRUE; ELSE
blink11:=FALSE; END_IF;

IF blinkenable12 THEN blink12:=NOT blink12; resetenable:=TRUE; ELSE
blink12:=FALSE; END_IF;

IF blinkenable13 THEN blink13:=NOT blink13; resetenable:=TRUE; ELSE
blink13:=FALSE; END_IF;

IF blinkenable14 THEN blink14:=NOT blink14; resetenable:=TRUE; ELSE
blink14:=FALSE; END_IF;

IF blinkenable15 THEN blink15:=NOT blink15; resetenable:=TRUE; ELSE
blink15:=FALSE; END_IF;

IF blinkenable16 THEN blink16:=NOT blink16; resetenable:=TRUE; ELSE
blink16:=FALSE; END_IF;

IF blinkenable1Mean THEN blink1Mean:=NOT blink1Mean; resetenable:=TRUE;
ELSE blink1Mean:=FALSE; END_IF;

IF blinkenable1TC THEN blink1TC:=NOT blink1TC; resetenable:=TRUE; ELSE
blink1TC:=FALSE; END_IF;

IF blinkenable1TC2 THEN blink1TC2:=NOT blink1TC2; resetenable:=TRUE; ELSE
blink1TC2:=FALSE; END_IF;

IF blinkenable1TC3 THEN blink1TC3:=NOT blink1TC3; resetenable:=TRUE; ELSE
blink1TC3:=FALSE; END_IF;
```

```
IF blinkenable1TC4 THEN blink1TC4:=NOT blink1TC4; resetenable:=TRUE; ELSE
blink1TC4:=FALSE; END_IF;


IF blinkenablem1 THEN blinkm1:=NOT blinkm1; resetenable:=TRUE; ELSE
blinkm1:=FALSE; END_IF;

IF blinkenablem2 THEN blinkm2:=NOT blinkm2; resetenable:=TRUE; ELSE
blinkm2:=FALSE; END_IF;

IF blinkenablem3 THEN blinkm3:=NOT blinkm3; resetenable:=TRUE; ELSE
blinkm3:=FALSE; END_IF;

IF blinkenablem4 THEN blinkm4:=NOT blinkm4; resetenable:=TRUE; ELSE
blinkm4:=FALSE; END_IF;

IF blinkenablem5 THEN blinkm5:=NOT blinkm5; resetenable:=TRUE; ELSE
blinkm5:=FALSE; END_IF;

IF blinkenablem6 THEN blinkm6:=NOT blinkm6; resetenable:=TRUE; ELSE
blinkm6:=FALSE; END_IF;
```

Τεχνικά χαρακτηριστικά ΚαρτώνοικογένειαςE-Logix 2K

General:

Number of Channels:	8 Type K
ADC resolution:	12bits
Filters:	50/60Hz rejection
Data Scaling Option:	Temperature (°C, °F)
Temperature Range:	-20...+850°C
Accuracy:	±1.0°C
Operating Temperature:	-40...+85°C
Storage Temperature:	-65...+120°C
Relative Humidity:	10 to 90%, noncondensing
Power Supply:	24VDC ±5%
Isolation Voltage:	1000VDC min.
Isolation Capacitance:	30pF min. / 150pF max.
Isolation Resistance:	1GΩ min.

Certifications:

CE compliance

Marine Approval (IN PROCESS)

CANopen Compatibility:

NMT (Network management):	
Slave	Yes
Master	No
Error control:	
Node guarding	No
Heartbeat	Yes
Node ID assignment:	
HW-switch (local interface)	Yes
Software switch	No
LSS	IN PROCESS
LMT	No
Node claiming	No
Proprietary	No
Number of PDOs	
RPDOs (Receive PDOs)	*0*
TPDOs (Transmit PDOs)	*2 (TPDO2, TPDO3)*
Supported PDO modes:	
Event-triggered	Yes
Time-triggered	Yes
Remotely-requested	No
Sync (cyclic)	Yes
Sync (acyclic)	Yes
PDO linking (change of CAN-ID is possible)	
supported	Yes
PDO mapping:	
Static	Yes
Variable (only in pre-operational state)	No
Dynamic	No
Number of SDOs	
Server SDO	*1*
Client SDO	*0*
Emergency message	
supported	Yes
CANopen application layer versions supported	
CiA 301 V 3.0	Yes
CiA 301 V 4.02	Yes
Device and application profiles supported	
CiA 401 V 2.1: CANopen device profile for generic I/O modules	Yes

General:

Number of Channels:	6 I/O
Isolation Between Channels:	YES

Input Characteristics:

Input Voltage (ON State):	12..40VDC
Input Voltage (OFF State):	0..4VDC
Input Current max.	5.7mA
Resistance of input channel:	6.4..7.2kOhm
Input Stage:	bipolar

Output Characteristics:

Output Type:	OptoMOS
Load Voltage:	400V
Load Current:	200mA
Isolation:	3750Vrms

Operating Temperature:	-40...+85°C
Storage Temperature:	-55...+120°C
Relative Humidity:	10 to 90%, noncondensing
Power Supply:	24VDC +-5%
Isolation Voltage:	1000VDC min.
Isolation Capacitance:	3pF min. / 15pF max.
Isolation Resistance:	1GΩ min.

Certifications:

CE compliance

Marine Approval (IN PROCESS)

CANopen Compatibility:

NMT (Network management):	
Slave	Yes
Master	No
Error control:	
Node guarding	No
Heartbeat	Yes
Node ID assignment:	
HW-switch (local interface)	Yes
Software switch	No
LSS	IN PROCESS
LMT	No
Node claiming	No
Proprietary	No
Number of PDOs	
RPDOs (Receive PDOs)	1
TPDOs (Transmit PDOs)	3
Supported PDO modes:	
Event-triggered	Yes
Time-triggered	Yes
Remotely-requested	No
Sync (cyclic)	Yes
Sync (acyclic)	Yes
PDO linking (change of CAN-ID is possible)	
supported	Yes
PDO mapping:	
Static	Yes
Variable (only in pre-operational state)	No
Dynamic	No
Number of SDOs	
Server SDO	1
Client SDO	0
Emergency message	
supported	Yes
CANopen application layer versions supported	
CiA 301 V 3.0	Yes
CiA 301 V 4.02	Yes
Device and application profiles supported	
CiA 401 V 2.1: CANopen device profile for generic I/O modules	Yes

ΕΠΙΛΟΓΟΣ

Η πτυχιακή εργασία αποτελεί το τελικό στάδιο των σπουδών μου στο τμήμα του Αυτοματισμού. Η ολοκλήρωση της πτυχιακής μου εργασίας ήταν ο τελευταίος στόχος για να ολοκληρώσω τις σπουδές μουκαι αποτελεί μια απόδειξη ότι η φοίτηση μου στη σχολή αυτή και οι γνώσεις που αποκόμισα, θα αποτελούν σημαντικά εφόδια για το μέλλον. Η ενασχόληση μου με την πτυχιακή εργασία μου, καθώς και μια μικρή επαγγελματική ενασχόληση στην τομέα της βιομηχανίας μου έδωσε τη δυνατότητα να ασχοληθώ με ένα θέμα που με ενδιαφέρει να αποκομίσω γνώσεις και εμπειρία που σίγουρα θα μου είναι χρήσιμες στις μελλοντικές μου αποφάσεις και στη μελλοντική μου σταδιοδρομία ως αυτοματιστής. Η επιλογή του θέματος ήταν μια επιθυμία μου να ασχοληθώ εκτενέστερα με την εφαρμογή των σύγχρονων αυτοματισμών στην βιομηχανία.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. Bosch CAN SPECIFICATION Version 2.0
2. SIEMENS Controller Area Network Version 2.0
3. DenisCollins-EamonnLane , «Προγραμματισμένοι Ελεγκτές» , εκδόσεις Τζιόλα .
4. Wikepedia
   - Λήμμα : Scada , http://en.wikipedia.org/wiki/SCADA
   - Λήμμα : Programmable logic controller , http://en.wikipedia.org/wiki/Programmable_logic_controller
   - Λυμμα : Can Bus , http://en.wikipedia.org/wiki/CAN_bus

5. Κρανάς Γ., Δασκαλόπουλος Ε., *«Βιομηχανικοί αυτοματισμοί & Προγραμματιζόμενοι Λογικοί Ελεγκτές PLC»*, 5η έκδοση, Εκδόσεις «ΙΩΝ», 1995.
6. Πανταζής Α. Νικόλαος, *«Προγραμματιζόμενοι λογικοί ελεγκτές»* Εκδόσεις Ίων, 2001.
7. Σμυρλής Ν. ,Ζαρογιάννης Λ. , «Συστήματα Αυτόματου Έλεγχου», Εκδόσεις Τζιολα,2005.
8. National Communications System , "Supervisory Control and Data Acquisition (SCADA) Systems", 2004 http://www.ncs.gov/library/tech_bulletins/2004/tib_04-1.pdf
9. TMP CAN2 manual http://www.gma.gr/xmsAssets/File/Demo/PRODUCTS-R/TMPCan-2/TMPCan-2-UM-V10.pdf
10. DIO CAN1 manual http://www.gma.gr/xmsAssets/File/Demo/PRODUCTS-R/DIOCan-1/DIOCan-1-UM-V10.pdf
11. CodeSys General Info http://www.3s-software.com/index.shtml?homepage
12. BERGHOFDC1000 (Βασικος ελεκτης εφαρμογής)http://www.berghof.com/Automation/en/Products/Controls/Display+controllers/DC1000+Basic+Interface.html http://www.berghof.com/multimedia/Downloads/BAT/Dokumente/CANtrol_dialog/DialogController/DC1000/DC1000_HB_en.pdf