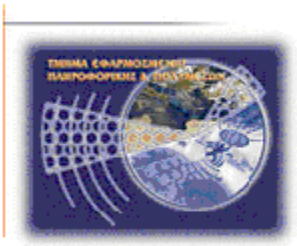
Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων



Πτυχιακή Εργασία

Ανάπτυξη εφαρμογής εικονικού κόμβου ασύρματης τηλεμετρίας

ΑΝΔΡΕΑΔΑΚΗΣ ΓΕΩΡΓΙΟΣ (ΑΜ:558)

Επιβλέπων καθηγητής : Μιαουδάκης Ανδρέας

Επιτροπή Αξιολόγησης :

Ημερομηνία Παρουσίασης :

Ευχαριστίες

Θα ήθελα να ευχαριστήσω όλους τους καθηγητές για τις γνώσεις που μου προσέφεραν σε όλα τα έτη της φοίτησης μου στην σχολή και ιδιαίτερα τον κ. Μιαουδάκη Ανδρέα για την καθοδήγηση του στην πτυχιακή μου εργασία. Επίσης ένα μεγάλο ευχαριστώ στην οικογένεια μου που με ενθάρρυνε να συνεχίσω και να τελειώσω τις σπουδές μου όταν παρουσιάστηκαν δυσκολίες για την φοίτηση μου.

Abstract

The purpose of this study is to develop application that will emulate the device "Netmaster RS485" made by "Elsist" Company, which is a Programmable Logic Controller (PLC) and functions as a remote telemetry node. This application will be used during the design and implementation of applications that control remotely this device or other similar devices with some upgrades for the original application.

Σύνοψη

Σκοπός της εργασίας είναι η ανάπτυξη εφαρμογής που θα εξομοιώνει την λειτουργία της συσκευής “Netmaster RS485” της εταιρείας “Elsist”, η οποία είναι ένας Προγραμματιζόμενος Λογικός Ελεγκτής (Programmable Logic Controller – PLC) και λειτουργεί ως κόμβος ασύρματης τηλεμετρίας. Η εφαρμογή αυτή θα χρησιμοποιηθεί κατά την διάρκεια της σχεδίασης και υλοποίησης εφαρμογών ασύρματου χειρισμού της παραπάνω συσκευής και άλλων παρόμοιων συσκευών, μετά όμως από ορισμένες αναβαθμίσεις για την αρχική εφαρμογή.

Πίνακας Περιεχομένων

Πίνακας Εικόνων

Λίστα Πινάκων

1 Εισαγωγή

Η πτυχιακή εργασία (ΠΕ) είναι ένα από τα ποιο σημαντικά μέρη του πτυχίου. Αποτελεί εξομοίωση του πραγματικού εργασιακού περιβάλλοντος και προετοιμάζει τον φοιτητή για την είσοδο του στην αγορά εργασίας. Με τη σημαντική συμμέτοχη και καθοδήγηση του καθηγητή ο φοιτητής εφοδιάζεται με εμπειρία και εξειδίκευση στο αντικείμενο που έχει επιλέξει να πραγματεύεται η πτυχιακή του εργασία.

1.1 Περίληψη

Σημαντικά κεφάλαια γενικότερα στην Πληροφορική και ειδικότερα στο Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων είναι οι Γλώσσες Προγραμματισμού και η επικοινωνία μεταξύ εφαρμογών με τη χρήση πακέτων δεδομένων. Στην παρούσα πτυχιακή εργασία χρησιμοποιήθηκε η τεχνολογία Visual C με την βοήθεια της εφαρμογής Microsoft Visual Basic 6.0 και έγινε εξομοίωση επικοινωνίας μέσω σειριακών θυρών υπολογιστή με την βοήθεια της δωρεάν εφαρμογής Virtual Serial Ports Emulator.

Αρχικά θα γίνει αναφορά στα Συστήματα Συλλογής Πληροφοριών και εποπτικού ελέγχου (Supervisory Control And Data Acquisition - SCADA) και παρουσίαση του Προγραμματιζόμενου Λογικού Ελεγκτή (Programmable Logic Controller - PLC) “NETMASTER RS485” και των λειτουργιών του. Στη συνέχεια θα παρουσιαστούν οι εφαρμογές που αναφέρονται στην προηγούμενη παράγραφο δίνοντας έμφαση στα εργαλεία που χρησιμοποιήθηκαν και θα αναλυθεί θεωρητικά το πρόβλημα. Επίσης θα παρουσιαστεί η υλοποίηση σε Visual C και θα αναλυθούν οι ρουτίνες και η δομή του κώδικα. Τέλος θα αναφερθούν τα συστήματα στα οποία μπορεί να εφαρμοστεί το πρόγραμμα και οι δυνατότητες επέκτασης του.

1.2 Κίνητρο για την Διεξαγωγή της Εργασίας

Κίνητρο μου υπήρξε η δυνατότητα μέσω της εφαρμογής αυτής να εμβαθύνω τις γνώσεις μου στην γλωσσά προγραμματισμού Visual C και τα εργαλεία της καθώς και στην κωδικοποίηση και αποκωδικοποίηση πακέτων δεδομένων. Επίσης ενδιαφέρων ήταν και η μελέτη των συστημάτων ασύρματης τηλεμετρίας και των εφαρμογών τους σε βιομηχανικές μονάδες και όχι μόνο, γεγονός που θα αποτελέσει σημαντικό εφόδιο στην σταδιοδρομία μου μετά την απόκτηση του πτυχίου της σχολής.

1.3 Σκοπός και Στόχοι Εργασίας

Η δημιουργία ενός δικτύου από μονάδες ασύρματης τηλεμετρίας, σε έναν εσωτερικό χώρο ερευνάς, ώστε να γίνει η μελέτη της λειτουργίας τους και να καταγραφούν οι απαιτήσεις είναι αρκετά χρονοβόρα διαδικασία και απαιτεί εξειδικευμένο εξοπλισμό. Σκοπός λοιπόν αυτής της πτυχιακής εργασίας είναι να αναπτυχτεί μια εφαρμογή η οποία θα αντικαθιστά τις μονάδες αυτές κατά την διάρκεια της σχεδίασης και υλοποίησης εφαρμογών απομακρυσμένου χειρισμού συγκεκριμένων μονάδων ασύρματης τηλεμετρίας.

Κατ’ επέκταση οι στόχοι της πτυχιακής αυτής εργασίας είναι η μελέτη, σχεδίαση και υλοποίηση εφαρμογής εικονικού κόμβου ασύρματης τηλεμετρίας. Μιας εφαρμογής δηλαδή που θα εξομοιώνει την λειτουργία ενός κόμβου από ένα σύστημα SCADA, ο οποίος είναι μια συσκευή που πραγματοποιεί μετρήσεις τοποθετημένη σε μια βιομηχανική μονάδα ή σε ένα απομακρυσμένο σημείο και στέλνει αυτές τις μετρήσεις στο κέντρο του συστήματος. Η εφαρμογή πρέπει να είναι πιστό λειτουργικό αντίγραφο του ελεγκτή “NETMASTER RS485” τον οποίο εξομοιώνει και να υπάρχουν όλες οι λειτουργιές και οι δυνατότητες που έχει και η συγκεκριμένη συσκευή.

1.4 Δομή Εργασίας

Κεφάλαιο 2: Συστήματα SCADA
Περιγράφονται τα Συστήματα SCADA και η δομή τους. Αναλύονται, οι αρχιτεκτονικές, τα πρωτόκολλα και τα μέρη από τα οποία αποτελούνται.
Κεφάλαιο 3: Συσκευή "Netmaster RS485" της "Elsist"
Γίνεται περιγραφή των Συσκευών PLC και RTU. Παρουσιάζονται τα χαρακτηριστικά και το πρωτόκολλο επικοινωνίας που χρησιμοποιεί η συσκευή "Netmaster RS485".
Κεφάλαιο 4: Visual Basic 6 και VSPE της "Eterlogic"
Περιγράφεται η γλώσσα προγραμματισμού Visual Basic 6 και ορισμένα εργαλεία της. Επίσης παρουσιάζεται η εφαρμογή "Visual Serial Port Emulator" και οι λειτουργίες της.
Κεφάλαιο 5: Εφαρμογή εικονικού κόμβου ασύρματης τηλεμετρίας
Παρουσιάζονται η διεπαφή, η ανάλυση, ο τρόπος ανάπτυξης, ο κώδικας και το διάγραμμα ροής της εφαρμογής "Netmaster RS485 Emulator".
Κεφάλαιο 6: Αποτελέσματα
Αναφέρονται οι εντυπώσεις και οι παρατηρήσεις από την εγγραφή της πτυχιακής εργασίας. Επίσης γίνεται λόγος για την χρησιμότητα της εφαρμογής και πιθανές επεκτάσεις της.

2 Συστήματα SCADA

2.1 Τηλεμετρία

Η Τηλεμετρία είναι σύνθετη λέξη και τα δύο συνθετικά της είναι οι ελληνικές λέξεις “τηλέ” και “μέτρον”, των οποίων οι έννοια είναι απομακρυσμένος και μέτρο αντίστοιχα. Η Τηλεμετρία είναι μια αυτοματοποιημένη διαδικασία επικοινωνίας με την οποία γίνονται μετρήσεις και συλλέγονται δεδομένα από απομακρυσμένα ή δυσπρόσιτα σημεία και διαβιβάζονται σε έναν δέκτη ώστε να γίνεται η παρακολούθηση τους. Επίσης η τηλεμετρία χρησιμοποιείται και σε συστήματα που βρίσκονται σε απομακρυσμένα σημεία και χρειάζονται εντολές για να λειτουργήσουν, δηλαδή σε συστήματα που απαιτούν τηλεχειρισμό.

Παρά το γεγονός ότι ο όρος αναφέρεται συνήθως σε μηχανισμούς ασύρματης μεταφοράς δεδομένων, όπως η χρήση των υπερηχητικών ή των συστημάτων υπερύθρων, περιλαμβάνει επίσης δεδομένα που μεταφέρονται μέσω άλλων μέσων, όπως το τηλέφωνο, το δίκτυο υπολογιστών και των οπτικών ή άλλων ενσύρματων επικοινωνιών. Επίσης πολλά σύγχρονα συστήματα τηλεμετρίας μπορούν να επωφεληθούν από το χαμηλό κόστος και την πανταχού παρουσία των δικτύων GSM με τη χρήση SMS για να λαμβάνουν και να διαβιβάζουν τα δεδομένα τηλεμετρίας.

Η Τηλεμετρία ξεκίνησε να χρησιμοποιείται τον $19^{ο}$ αιώνα από ορισμένες εταιρείες και οργανισμούς και μέχρι σήμερα έχει βρει εφαρμογή σε πάρα πολλούς τομείς της βιομηχανίας, των υπηρεσιών και των επιστημών μερικοί από αυτούς είναι :

- Μετεωρολογία
- Βιομηχανία Πετρελαίου και Φυσικών Αερίων
- Εξερεύνηση του Διαστήματος
- Γεωργία
- Διαχείριση Νερού
- Πτητικές δοκιμές
- Στρατιωτική νοημοσύνη
- Παρακολούθηση και παροχής ενέργειας
- Ιατρική
- Έρευνα και διαχείριση της αλιείας και της άγριας ζωής
- Επιβολή του νόμου
- Επικοινωνίες

Εικόνα 2.1 Γενική Ιδέα Τηλεμετρίας

2.2 Περιγραφή Συστημάτων SCADA

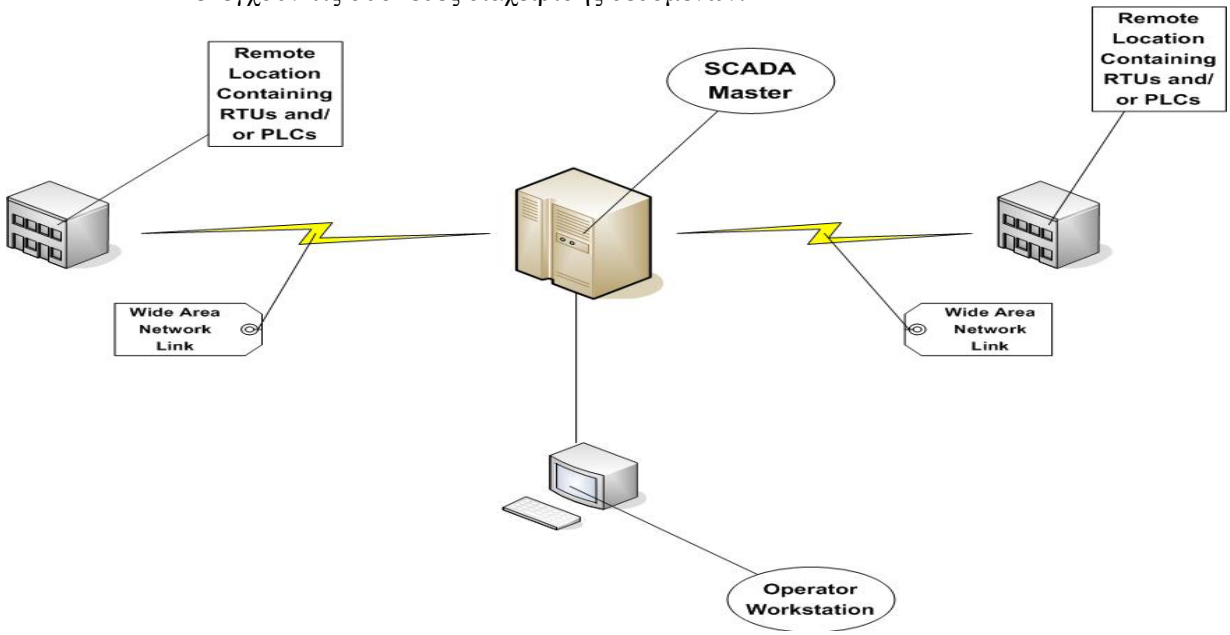
2.2.1 Περίληψη

Η συντομογραφία S.C.A.D.A. προέρχεται από τα αρχικά των λέξεων Supervisory Control And Data Acquisition (Εποπτικός Έλεγχος και Συλλογή Πληροφοριών). Τα συστήματα SCADA εκτελούν τηλεμετρία και χρησιμοποιούνται για την παρακολούθηση και τον έλεγχο μιας εγκατάστασης ή εξοπλισμού σε βιομηχανίες όπως τηλεπικοινωνίες, έλεγχο νερού, αποβλήτων και ενέργειας, διύλιση και μεταφορά πετρελαίου και φυσικού αερίου. Τα συστήματα αυτά περιλαμβάνουν την μεταφορά δεδομένων μεταξύ ενός κεντρικού διακομιστή SCADA και ενός αριθμού από Απομακρυσμένες Τερματικές Μονάδες (Remote Terminal Units – RTUs) επίσης μεταξύ του κεντρικού διακομιστή και των τερματικών των χρηστών.

Ένα σύστημα SCADA συγκεντρώνει πληροφορίες, όπως που υπάρχει διαρροή σε μια σωλήνα, μεταφέρει τις πληροφορίες στην RTU και ειδοποιεί το κεντρικό σύστημα για την διαρροή, συμπεριλαμβάνοντας στο μήνυμα την απαραίτητη ανάλυση για την κρισιμότητα της διαρροής και παρουσιάζοντας την σε οργανωμένη μορφή. Τα συστήματα αυτά είναι είτε απλά, όπως κάποιο που αναλαμβάνει την παρουσίαση συνθηκών περιβάλλοντος σε ένα κτήριο, είτε αρκετά πολύπλοκα, όπως κάποιο το οποίο συλλέγει και παρουσιάζει την δραστηριότητα σε ένα μητροπολιτικό δίκτυο νερού. Η μεταφορά των πληροφοριών γίνεται είτε μέσω τοπικών δικτύων (Local Area Networks – LANs) και ευρείας περιοχής δικτύων (Wide Area Networks – WANs), είτε μέσω ασύρματων δικτύων (Wireless Local Area Networks – WLANs).

Τα συστήματα SCADA, όπως φαίνεται και στην Εικόνα 2.2, αποτελούνται από :

- Μία ή περισσότερες συσκευές συλλογής και διαχείρισης πληροφοριών (RTUs), οι οποίες επικοινωνούν με συσκευές μέτρησης, τοπικού ελέγχου διακοπτών και ενεργοποίησης βαλβίδων.
- Ένα σύστημα επικοινωνίας, το οποίο είναι υπεύθυνο για την μεταφορά των δεδομένων είτε μέσω ασύρματης ζεύξης, μέσω τηλεφώνου ή μέσω δορυφόρου είτε συνδυασμό των προαναφερθέντων.
- Έναν κεντρικό υπολογιστή διακομιστή, ο οποίος ονομάζεται συνήθως SCADA Center ή Κεντρική Μονάδα Επεξεργασίας (Master Terminal Unit – MTU).
- Έναν αριθμό από συστήματα λογισμικού (Man Machine Interfaces – MMIs), τα οποία παρέχουν την λειτουργία της MTU και των σταθμών εργασίας των διαχειριστών. Επίσης υποστηρίζουν το σύστημα επικοινωνίας, παρουσιάζουν τις πληροφορίες και ελέγχουν τις συσκευές διαχείρισης δεδομένων.



Εικόνα 2.2 Τυπικό Σύστημα SCADA

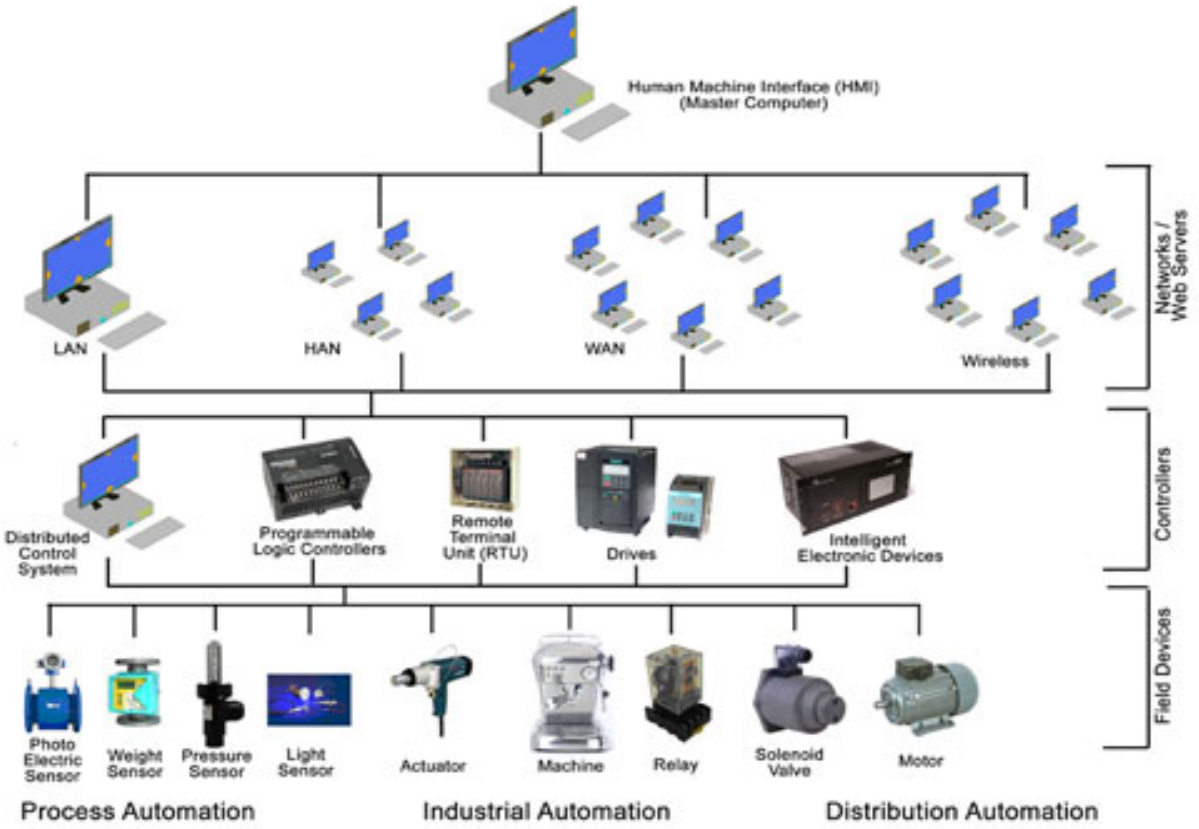
2.2.2 Συσκευές Συλλογής και Διαχείρισης Πληροφοριών

Οι συσκευές Συλλογής και Διαχείρισης Πληροφοριών είναι "τα μάτια και τα αυτιά" ενός συστήματος SCADA καθώς παρέχουν πληροφορίες που όμως μόνο ένας έμπειρος διαχειριστής γνωρίζει να επεξεργαστεί. Τέτοιες συσκευές μπορεί να είναι μετρητές στάθμης δεξαμενών, ροής νερού και επιπέδου ενέργειας όπως και πομποί θερμοκρασίας και κατάστασης βαλβίδων. Επίσης οι συσκευές αυτές είναι "τα χέρια" των συστημάτων SCADA δίνοντας την δυνατότητα στον διαχειριστή να ελέγχει εξοπλισμό όπως οι μηχανισμοί κίνησης ηλεκτρικών βαλβίδων, διακόπτες ελέγχου κινητήρων και ηλεκτρονικές εγκαταστάσεις δοσολογίας χημικών, αυτοματοποιώντας έτσι για παράδειγμα την διαδικασία διανομής νερού.

Οι πληροφορίες που λαμβάνονται ή στέλνονται στις συσκευές αυτές μετατρέπονται σε συμβατή μορφή με τα συστήματα SCADA προτού παρουσιαστούν σε κάποιο τερματικό ή ελέγξουν κάποια συσκευή. Τον ρόλο αυτό αναλαμβάνουν οι Απομακρυσμένες Τερματικές Μονάδες (RTUs), που μετατρέπουν τα ηλεκτρονικά μηνύματα, που είτε προέρχονται από τους αισθητήρες είτε προορίζονται για τους ελεγκτές, σε μια γλώσσα επικοινωνίας ή αλλιώς πρωτόκολλο επικοινωνίας, το οποίο χρησιμοποιείται για την μεταφορά των δεδομένων μέσω ενός καναλιού επικοινωνίας.

Οι εντολές για τους αυτοματισμούς σε ένα σύστημα SCADA συνήθως αποθηκεύονται τοπικά στους Προγραμματιζόμενους Λογικούς Ελεγκτές (Programmable Logic Controllers - PLCs), οι οποίοι είναι συσκευές αυτόματης προβολής και ελέγχου βιομηχανικών εγκαταστάσεων. Μπορούν να χρησιμοποιηθούν είτε αυτόνομα είτε σε συνδυασμό με άλλες συσκευές SCADA ή και διαφορετικά συστήματα και πολλές φορές αντικαθιστούν συστήματα ηλεκτρονόμων (relay logic). Ενσωματώνουν προγραμματισμένη νοημοσύνη με τη μορφή λογικών διαδικασιών και έτσι τα σήματα που λαμβάνουν από τις εισόδους τα επεξεργάζεται η κεντρική μονάδα τους σύμφωνα με τις αποθηκευμένες εντολές και εκτελώντας λογικές και αριθμητικές πράξεις μεταβιβάζει τα αποτελέσματα στις εξόδους τους.

Η διάθρωση ενός συστήματος SCADA παρουσιάζεται στην Εικόνα 2.3. Επίσης μια τυπική συσκευή RTU και μια PLC, των οποίων θα αναλυθεί η δομή και οι λειτουργίες στο επόμενο κεφάλαιο, φαίνονται στην Εικόνα 2.4 και στην Εικόνα 2.5 αντίστοιχα.



Εικόνα 2.3 Διάθρωση συστήματος SCADA

Εικόνα 2.4 Τυπική Ασύρματη Συσκευή RTU

Εικόνα 2.5 Τυπικός Ελεγκτής PLC

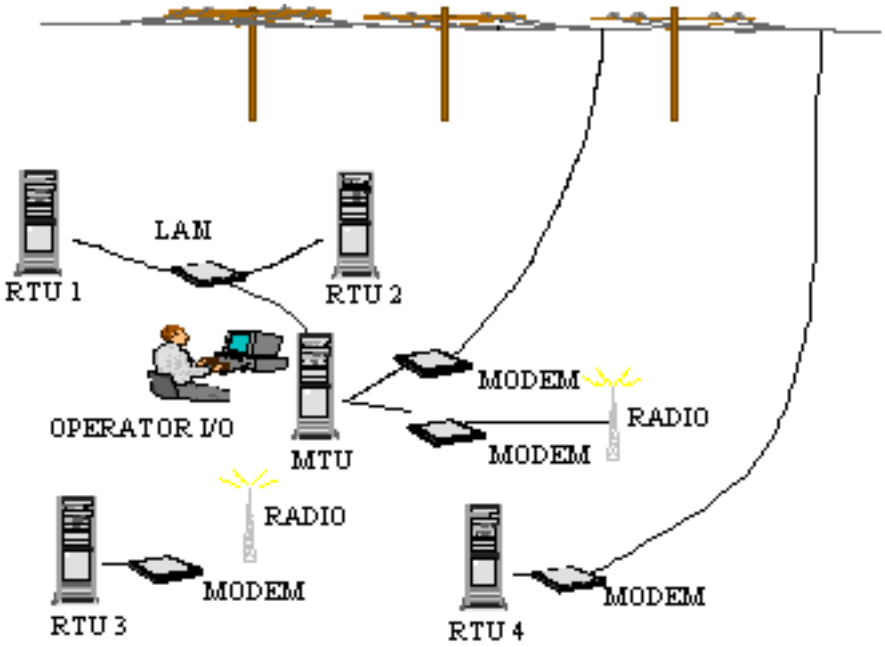
2.2.3 Δίκτυο Επικοινωνιών

Για να λειτουργήσει το σύστημα SCADA απαιτείται επικοινωνία μεταξύ του κεντρικού διακομιστή (MTU), των επιμέρους τερματικών MMI και των μονάδων RTUs. Τα μέσα για την επικοινωνία αυτή τα παρέχει το Δίκτυο Επικοινωνιών, το οποίο αναφέρεται στον εξοπλισμό που χρειάζεται για να μεταφερθούν δεδομένα από και προς διαφορετικές τοποθεσίες. Η σχέση μεταξύ MTU και RTUs είναι ανάλογη με τη σχέση master – slave και στην Εικόνα 2.6 παρατίθεται η τοπολογία ενός τέτοιου συστήματος. Η επικοινωνία μπορεί να είναι ενσύρματη (σειριακή επικοινωνία RS232, RS485, δίκτυα Profibus, Ethernet, τηλεφωνική σύνδεση, Internet) ή και ασύρματη (ραδιοκύματα, δορυφορική σύνδεση, μικροκύματα).

Ενσύρματα δίκτυα όπως Σειριακές Θύρες και Ethernet χρησιμοποιούνται σε εργοστάσια και μικρής έκτασης δίκτυα διότι είναι μεγάλο το κόστος των καλωδίων και της εγκατάστασης τους. Μια ποιο οικονομική λύση για δίκτυα που καλύπτουν μεγάλη γεωγραφική περιοχή είναι το τηλεφωνικό καλώδιο όπου με μόνιμη ή dial-up σύνδεση Internet γίνεται η ένωση των κεντρικών συστημάτων με τους απομακρυσμένους σταθμούς. Επειδή όμως με την μόνιμη σύνδεση χρειάζεται μια σύνδεση ανά σταθμό, χρησιμοποιούνται dial-up συνδέσεις σε συστήματα που κάνουν ανανέωση δεδομένων ανά τακτά χρονικά διαστήματα. Σε αυτές τις συνδέσεις ο διακομιστής καλεί τον αριθμό που αντιστοιχεί σε μια απομακρυσμένη συσκευή για να ζητήσει τις ενδείξεις των αισθητήρων και να στείλει εντολές προς διάφορους μηχανισμούς.

Οι τηλεφωνικές γραμμές δεν φτάνουν σε πολύ απομακρυσμένα σημεία όπου και χρησιμοποιείτε η ασύρματη τεχνολογία. Μέσω είτε τοπικών ασύρματων δικτύων (Wi-Fi), είτε δορυφορικής λήψης είτε του δικτύου κινητής τηλεφωνίας (GPRS ή 3G) ενώνεται ο MTU με τις μονάδες RTU και όταν η ασύρματη δικτύωση δεν είναι εφικτή χρησιμοποιούνται αναμεταδότες για να επιτευχτεί η σύνδεση. Η γενική τοπολογία των συστημάτων SCADA παρουσιάζεται στην Εικόνα 2.6.

Τα Δίκτυα SCADA στην αρχή ήταν αποκλειστικά για την εξυπηρέτηση των συσκευών τους, στην πορεία όμως και καθώς η τεχνολογία δικτύων αναπτύχθηκε, τα τοπικά δίκτυα (LAN) άρχισαν να χρησιμοποιούνται ευρέως σε γραφεία, επιχειρήσεις και βιομηχανικές εγκαταστάσεις. Έτσι υπήρξε η δυνατότητα ενοποίησης των δύο αυτών δικτύων ώστε να μην χρειάζεται η δημιουργία ενός ξεχωριστού δικτύου για τα τερματικά SCADA. Επίσης βρέθηκε τρόπος να ενοποιηθούν τα SCADA δεδομένα με της υπάρχουσες εφαρμογές γραφείου, όπως προγράμματα λογιστικών φύλλων, συστήματα διαχείρισης λειτουργίας, προγράμματα βάσεων δεδομένων και συστήματα μοντελοποίησης διανομής νερού.



Εικόνα 2.6 Γενική τοπολογία συστήματος SCADA

2.2.4 Κεντρική Μονάδα Επεξεργασίας

Η Κεντρική Μονάδα Επεξεργασίας (Master Terminal Unit – MTU) είναι είτε ένας υπολογιστής, είτε μια ομάδα από εξυπηρετητές (servers) και λειτουργεί σαν κεντρικός διακομιστής (central computer host). Αναλαμβάνει δηλαδή την διασύνδεση των χρηστών με το σύστημα SCADA και επεξεργάζεται την πληροφορία που λαμβάνεται ή στέλνεται στις διάφορες μονάδες RTU. Την πληροφορία αυτή την παρουσιάζει το σύστημα SCADA στους χρήστες σε τέτοια μορφή ώστε να μπορούν με την σειρά τους να την επεξεργαστούν. Τα τερματικά των χρηστών είναι συνδεμένα με την MTU μέσω τοπικού δικτύου (LAN) ή μέσω δικτύου ευρείας περιοχής (WAN).

Το λογισμικό της MTU είναι ένα εξαιρετικά πολύπλοκο πρόγραμμα. Χρειάζεται να διαχειρίζεται σωστά έναν όγκο πληροφοριών πραγματικού χρόνου και να τις δρομολογεί σωστά με τέτοιο τρόπο ώστε ο κάθε χρήστης του συστήματος και η κάθε μηχανή, που συνδέεται σε αυτό, να έχει την πληροφορία που χρειάζεται, τη στιγμή που τη χρειάζεται και στη μορφή που τη χρειάζεται. Επιπλέον, καθώς οι διαθέσιμες τεχνολογικές λύσεις δεν παραμένουν σταθερές σε όλη τη διάρκεια της ζωής του συστήματος αλλά μεταβάλλονται με την πάροδο του χρόνου, το λογισμικό αυτό πρέπει να είναι αρκετά ευέλικτο και επεκτάσιμο ώστε να μπορεί να προσαρμόζεται εύκολα και ομαλά στις καινούργιες κάθε φορά συνθήκες.

Με την αυξημένη χρήση προσωπικών υπολογιστών, η δικτύωση στο χώρο ενός γραφείου ή μιας επιχείρησης έχει γίνει αναγκαία και κατ' επέκταση τα συστήματα SCADA έπρεπε να είναι διαθέσιμα ώστε να συνεργάζονται με τους προσωπικούς υπολογιστές των γραφείων. Αποτέλεσμα ήταν τα συστήματα SCADA να εγκαθίστανται σε εξυπηρετητές (servers) ίδιους με αυτούς που εξυπηρετούν εφαρμογές γραφείου. Το γεγονός αυτό δημιούργησε μεγάλο εύρος δυνατοτήτων συνεργασίας των συστημάτων SCADA με εφαρμογές γραφείου όπως συστήματα GIS, προγράμματα μοντελοποίησης υδραυλικών συστημάτων, πληροφοριακές βάσεις δεδομένων και συστήματα σχεδιασμού διαχείρισης και ελέγχου.

2.2.5 Τερματικά διαχειριστών και λογισμικό διασύνδεσης

Τα τερματικά των χειριστών του συστήματος ή αλλιώς οι σταθμοί εργασίας των χρηστών είναι διασυνδεμένα με την MTU, η οποία λειτουργεί σαν εξυπηρετητής για την εφαρμογή SCADA. Ενώ τα τερματικά, που "τρέχουν" το λογισμικό διασύνδεσης, λειτουργούν σαν πελάτες ζητώντας και λαμβάνοντας πληροφορίες από την κεντρική μονάδα επεξεργασίας βασισμένες στις αιτήσεις και εντολές των διαχειριστών.

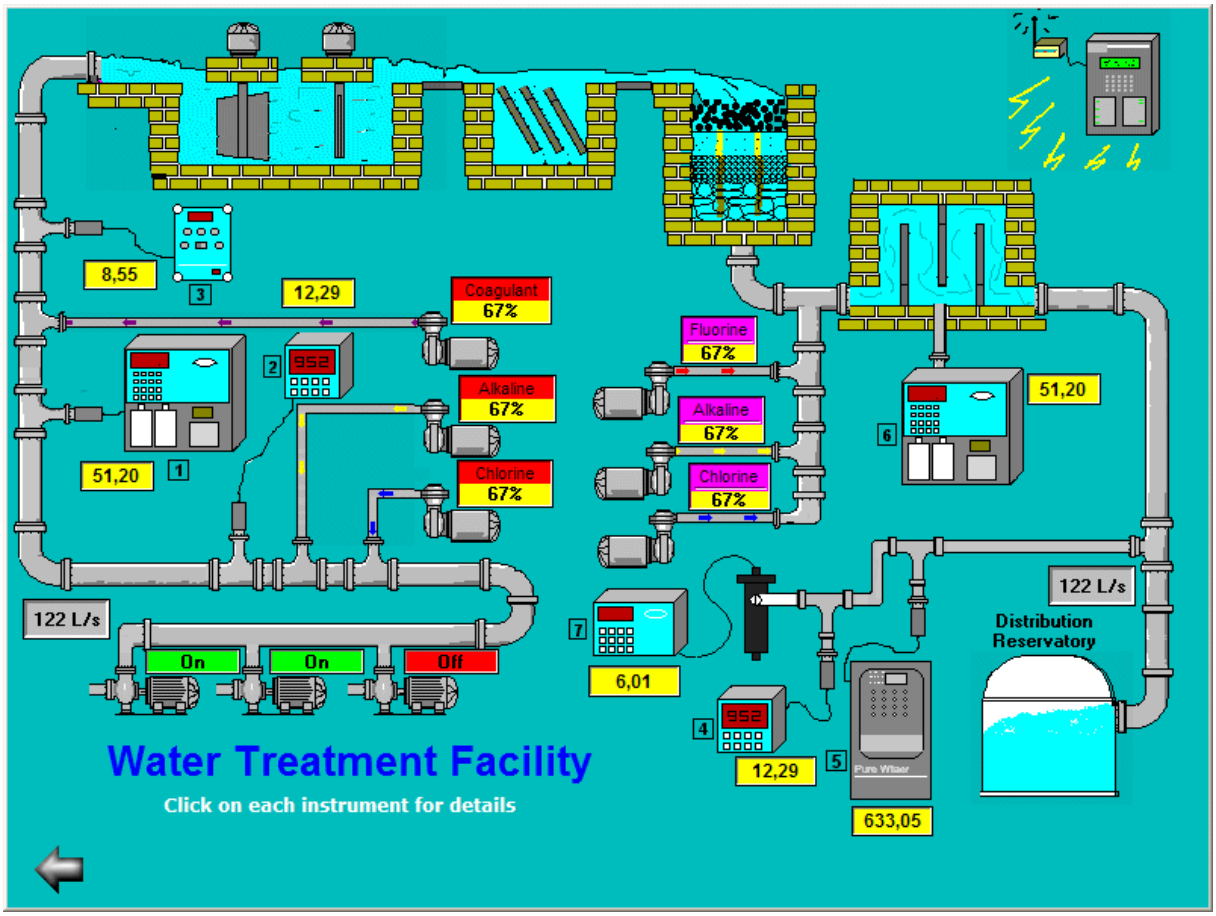
Το λογισμικό διασύνδεσης ή αλλιώς Man Machine Interface/Human Machine Interface (MMI/HMI) αποτελεί το μέρος της λειτουργίας των SCADA που αλληλεπιδρά με τον τελικό χρήστη. Συνήθως αποτελούνται από μια οπτική απεικόνιση της διεργασίας, πάνω στην οποία εμφανίζονται τιμές μεταβλητών, καταστάσεις ή και διαγράμματα. Ακόμη τα συστήματα αυτά επιτρέπουν την κατ' απαίτηση εμφάνιση δεδομένων όπως ιστορικών των μεταβλητών και ειδικών διαγραμμάτων. Ένα στιγμιότυπο από ένα τέτοιο λογισμικό παρουσιάζεται στην Εικόνα 2.7.

Οι πληροφορίες αντλούνται από τη βάση δεδομένων του συστήματος SCADA, γι' αυτό όπως είναι κατανοητό τα συστήματα SCADA και HMI είναι αλληλένδετα συνδεδεμένα μεταξύ τους και για το λόγο αυτό συχνά δεν διαχωρίζονται. Στην πραγματικότητα βέβαια οι περισσότεροι κατασκευαστές συστημάτων SCADA ενσωματώνουν την δυνατότητα ανάπτυξης HMI εφαρμογών στις υπηρεσίες ή τα πακέτα SCADA τους. Η ουσία είναι όμως ότι μια εφαρμογή HMI μπορεί να κατασκευαστεί ανεξάρτητα από των πυρήνα καταγραφής ενός SCADA, έτσι ώστε να αντλεί πληροφορίες από αυτό και να το χρησιμοποιεί για να διεξάγει τον απαραίτητο έλεγχο. Επίσης έχει αρχιτεκτονική on-line ώστε να μπορούμε να προσθέσουμε, να διαγράψουμε και να τροποποιήσουμε συνδέσεις εισόδου και εξόδου, γραφικά και λογικές εποπτικού ελέγχου, χωρίς διακοπή της διαδικασίας.

Ορισμένα πακέτα λογισμικού που χρησιμοποιούνται συνήθως σε ένα σύστημα SCADA είναι τα εξής:

- Το Λειτουργικό Σύστημα του Κεντρικού Υπολογιστή Διακομιστή, το οποίο ελέγχει τον κεντρικό διακομιστή του συστήματος SCADA.

- Το Λειτουργικό Σύστημα των Τερματικών των Χρηστών, που είναι συνήθως το ίδιο με αυτό του κεντρικού διακομιστή και αναλαμβάνει την διασύνδεση του διακομιστή με τα τερματικά των χρηστών.
- Την Εφαρμογή της MTU, λογισμικό που διαχειρίζεται την μετάδοση και λήψη δεδομένων από και προς τις RTU και την MTU. Παρέχει επίσης το γραφικό περιβάλλον που περιέχει τις λειτουργίες ελέγχου και τις σελίδες των συναγερμών και της γραμμής παραγωγής.
- Την Εφαρμογή των Τερματικών των Χρηστών, η οποία δίνει την δυνατότητα στους χρήστες να προσπελαύνουν πληροφορίες διαθέσιμες στην εφαρμογή της MTU.
- Τα Προγράμματα Οδήγησης Πρωτοκόλλων Επικοινωνίας, τα οποία εγκαθίστανται στην MTU και στις RTU και ελέγχουν την μετάφραση των δεδομένων μεταξύ των διασυνδέσεων επικοινωνίας στο σύστημα SCADA.
- Το Λογισμικό Διαχείρισης του Δικτύου Επικοινωνιών, το οποίο ελέγχει το δίκτυο και του δίνει την δυνατότητα να ρυθμίζει την λειτουργία του για καλύτερη απόδοση και τυχόν λάθη.
- Το Λογισμικό Αυτοματισμού των RTU, όπου χρησιμοποιώντας το λογισμικό αυτό οι μηχανικοί ρυθμίζουν και συντηρούν την εφαρμογή που είναι εγκατεστημένη στις μονάδες RTU. Επίσης συμπεριλαμβάνει την τοπική εφαρμογή αυτοματισμού και δεδομένα διαδικασιών που εκτελούνται στις RTUs.



Εικόνα 2.7 Εφαρμογή HMI της Ellipse Software για την εποπτεία υδραγωγείου.

2.3 Αρχιτεκτονικές SCADA

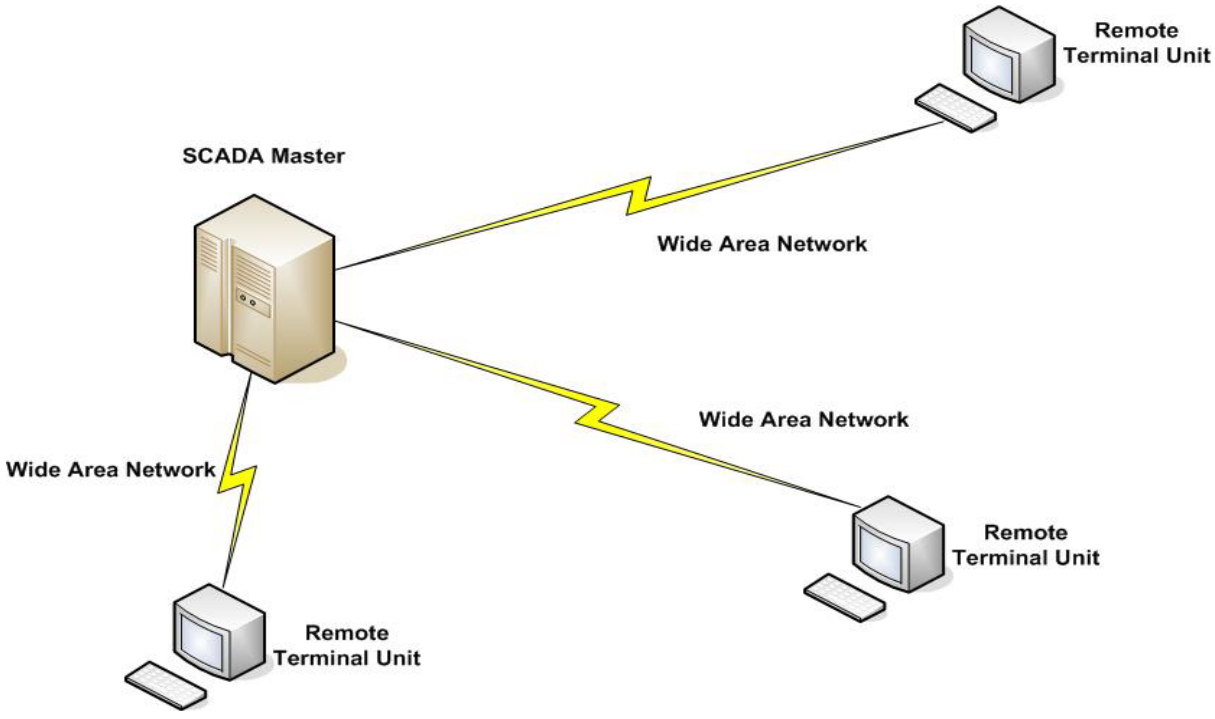
Τα συστήματα SCADA αναπτύχθηκαν παράλληλα με την τεχνολογία των υπολογιστών και διακρίνονται τρείς αρχιτεκτονικές τους. Η πρώτη αρχιτεκτονική που χρησιμοποιήθηκε ήταν τα συγκεντρωτικά συστήματα, στη συνέχεια η δεύτερη ήταν τα κατανεμημένα και η τρίτη τα δικτυωμένα συστήματα SCADA. Στο κεφάλαιο αυτό γίνεται ανάλυση των αρχιτεκτονικών αυτών και της λειτουργίας τους.

2.3.1 Συγκεντρωτικά συστήματα SCADA

Όταν εμφανίστηκαν τα συστήματα SCADA η επιστήμη της πληροφορικής στηρίζονταν σε κεντρικούς υπολογιστές με μεγάλη υπολογιστική ισχύ, οι οποίοι εξυπηρετούσαν όλες τις περιφερικές μονάδες. Λόγω του ότι δεν υπήρχαν τα δίκτυα όπως είναι σήμερα τα συστήματα SCADA ήταν συγκεντρωτικά χωρίς σχεδόν καμία σύνδεση με άλλα συστήματα. Στη συνέχεια σχεδιάστηκαν τα Ευρείας Περιοχής Δίκτυα (WAN) και ενσωματώθηκαν ώστε να επικοινωνούν οι κεντρικοί υπολογιστές με τις RTUs σε μια περιοχή.

Τα πρωτόκολλα επικοινωνίας που χρησιμοποιούνταν στα δίκτυα SCADA αναπτύχτηκαν από αντιπροσώπους και πωλητές εξοπλισμού RTU και κατά συνέπεια δεν ήταν αρκετά αποδοτικά. Τα πρωτόκολλα αυτά δεν υποστήριζαν σχεδόν καμία λειτουργία πέραν των αναγκαίων, όπως η σάρωση και ο έλεγχος των μονάδων που βρίσκονταν στις απομακρυσμένες συσκευές. Επίσης ήταν γενικά μη εφικτό να ενσωματωθούν άλλοι τύποι δεδομένων στις επικοινωνίες των RTU μέσα στο δίκτυο. Η συνδεσιμότητα με τον κεντρικό υπολογιστή ήταν περιορισμένη και η σύνδεση γινόταν στο επίπεδο δίαυλων μέσω ενός πνευματικά κατοχυρωμένου προσαρμογέα ή ελεγκτή, τοποθετημένου στο πίσω μέρος της κεντρικής μονάδας επεξεργασίας.

Περιττό σε αυτήν την αρχιτεκτονική ήταν το γεγονός ύπαρξης δύο πανομοιότυπων από θέμα εξοπλισμού κεντρικών συστημάτων, ενός κύριου και ενός εφεδρικού, συνδεμένα στο επίπεδο δίαυλων. Η βασική λειτουργία του εφεδρικού ήταν να ελέγχει το κύριο σύστημα και αναλαμβάνει την επεξεργασία σε περίπτωση ανίχνευσης λάθους. Αυτό ο τύπος εφεδρικής λειτουργίας είχε σαν αποτέλεσμα την ελάχιστη ή καθόλου επεξεργασία και διαχείριση των λειτουργιών από το εφεδρικό σύστημα. Στην Εικόνα 2.8 φαίνεται μια τυπική αρχιτεκτονική πρώτης γενιάς.



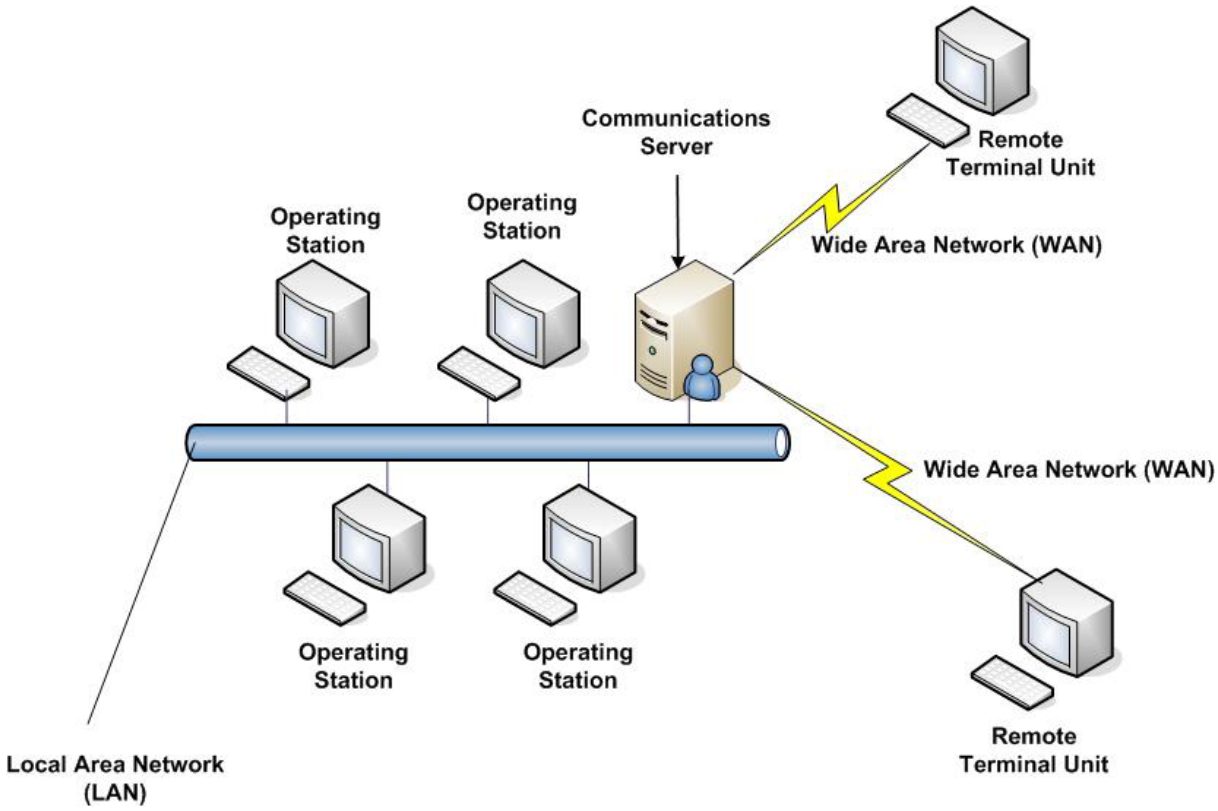
Εικόνα 2.8 Αρχιτεκτονική SCADA πρώτης γενιάς

2.3.2 Κατανεμημένα συστήματα SCADA

Η νέα γενιά των συστημάτων SCADA εκμεταλλεύτηκε τις εξελίξεις και την βελτίωση στην τεχνολογία των τοπικών δικτύων (LAN) έτσι ώστε να διανέμει την επεξεργασία μέσω πολλαπλών συστημάτων. Διάφοροι σταθμοί εργασίας ήταν συνδεμένοι σε ένα τοπικό δίκτυο ώστε να μοιράζονται μεταξύ τους πληροφορίες σε πραγματικό χρόνο και να έχει ο καθένας συγκεκριμένη λειτουργία. Κάποιοι από τους σταθμούς αυτούς εξυπηρετούσαν την επικοινωνία των διαφόρων συσκευών όπως τις RTUs, άλλοι φιλοξενούσαν την διεπαφή των χρηστών παρέχοντας το λογισμικό διασύνδεσης μεταξύ χειριστών και υπολογιστή για τους διαχειριστές του συστήματος. Ακόμη ορισμένοι σταθμοί λειτουργούσαν ως επεξεργαστές υπολογισμών ή εξυπηρετητές βάσεων δεδομένων.

Η διανομή των λειτουργιών των συστημάτων SCADA σε διάφορους σταθμούς εργασίας έδωσε την δυνατότητα για εκμετάλλευση μεγαλύτερης υπολογιστικής ισχύς συνολικά, σε σχέση με την χρησιμοποίηση ενός επεξεργαστή. Τα δίκτυα που ένωναν τα συστήματα αυτά βασίζονταν κυρίως σε πρωτόκολλα τοπικών δικτύων και δεν ήταν ικανά να χρησιμοποιηθούν πέρα των ορίων του τοπικού περιβάλλοντος. Η κατανομή των λειτουργιών του συστήματος μέσω του δικτύου βοηθούσε επίσης στο να βελτιωθεί η αξιοπιστία του, σε αντίθεση με τα συστήματα πρώτης γενιάς που υπήρχαν πολλές περιπτώσεις αναμονής του συστήματος σε τυχόν εμφάνιση σφάλματος. Η κατανεμημένη αρχιτεκτονική, κρατούσε μια κατάσταση από όλους τους συνδεμένους στο δίκτυο σταθμούς και σε περιπτώσεις σφάλματος μιας διεπαφής μετέφερε την λειτουργία του συστήματος σε άλλη διεπαφή.

Συνήθως κάποια πρωτόκολλα ήταν ιδιοκτησία κάποιου προμηθευτή, ο οποίος το δημιουργούσε μόνος του από την αρχή ή κάποια έκδοση του. Έτσι μπορούσε ο προμηθευτής να βελτιστοποιεί το πρωτόκολλο για κυκλοφορία πραγματικού χρόνου, με αποτέλεσμα όμως να περιορίζει την συνδεσιμότητα στο δίκτυο SCADA για δίκτυα άλλων προμηθευτών. Ένα τυπικό σύστημα SCADA δεύτερης γενιάς παρουσιάζεται στην Εικόνα 2.9.



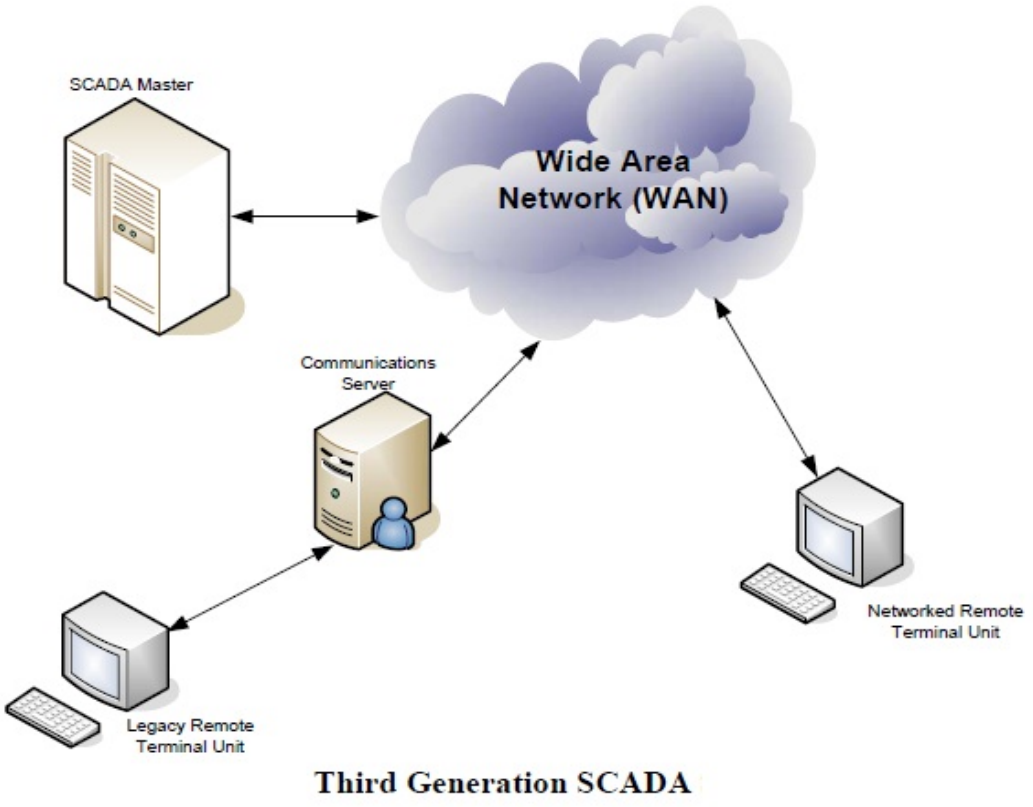
Εικόνα 2.9 Σύστημα SCADA Δεύτερης Γενιάς

2.3.3 Δικτυωμένα Συστήματα SCADA

Η τελευταία και τωρινή γενιά των συστημάτων SCADA μοιάζει με την δεύτερη γενιά, με σημαντική διαφορά ότι δεν ελέγχονται πια τα πρωτόκολλα από τους προμηθευτές, αλλά είναι μια αρχιτεκτονική ανοιχτού συστήματος χωρίς κατοχυρωμένα δικαιώματα. Και εδώ υπάρχουν πολλά δικτυωμένα συστήματα που μοιράζονται λειτουργίες της κεντρικής μονάδας επεξεργασίας. Μια άλλη σημαντική βελτίωση, που δημιουργήθηκε λόγω της χρήσης ανοιχτών προτύπων και πρωτοκόλλων, είναι ότι έγινε δυνατή η διανομή των λειτουργιών των συστημάτων SCADA μέσω του δικτύου ευρείας περιοχής (WAN) και όχι μόνο μέσω τοπικού δικτύου (LAN).

Τα ανοιχτά πρότυπα βοήθησαν να εξαλειφθούν πολλοί περιορισμοί που υπήρχαν στις προηγούμενες αρχιτεκτονικές των συστημάτων SCADA. Έγινε ευκολότερη για τον χρήστη η σύνδεση στο σύστημα συσκευών από τρίτους κατασκευαστές, όπως οθόνες, εκτυπωτές και σκληροί δίσκοι. Όμως το μεγαλύτερο κέρδος που προέκυψε από την βελτίωση αυτή είναι η χρησιμοποίηση πρωτοκόλλων δικτύων ευρείας περιοχής, όπως το πρωτόκολλο διαδικτύου (Internet Protocol – IP), για την επικοινωνία μεταξύ του κεντρικού σταθμού και του εξοπλισμού επικοινωνίας. Έτσι υπάρχει η δυνατότητα το κομμάτι του κεντρικού σταθμού που είναι υπεύθυνο για την επικοινωνία με τις διάφορες συσκευές, να διαχωρίζεται από το κεντρικό σύστημα μέσω ενός δικτύου WAN. Οι προμηθευτές παράγουν τώρα RTUs που επικοινωνούν με τον κεντρικό σταθμό χρησιμοποιώντας σύνδεση Ethernet. Η Εικόνα 2.10 παρουσιάζει ένα δικτυωμένο σύστημα SCADA.

Επίσης σημαντικό πλεονέκτημα της δικτύωσης των συστημάτων SCADA είναι η διάσωση του συστήματος σε περίπτωση καταστροφής. Στα δεύτερης γενιάς συστήματα αναβαθμίστηκε η αξιοπιστία, αλλά σε μια εξ ολοκλήρου καταστροφή των εγκαταστάσεων που είναι τοποθετημένος ο κεντρικός επεξεργαστής θα χανόταν ολόκληρο το σύστημα. Η διανομή της επεξεργασίας σε διαφορετικά φυσικά σημεία δίνει την δυνατότητα να δημιουργηθεί ένα σύστημα SCADA ικανό να επιβιώσει σε τυχόν απώλεια μιας εγκατάστασης.



Εικόνα 2.10 Συστήματα SCADA Τρίτης Γενιάς

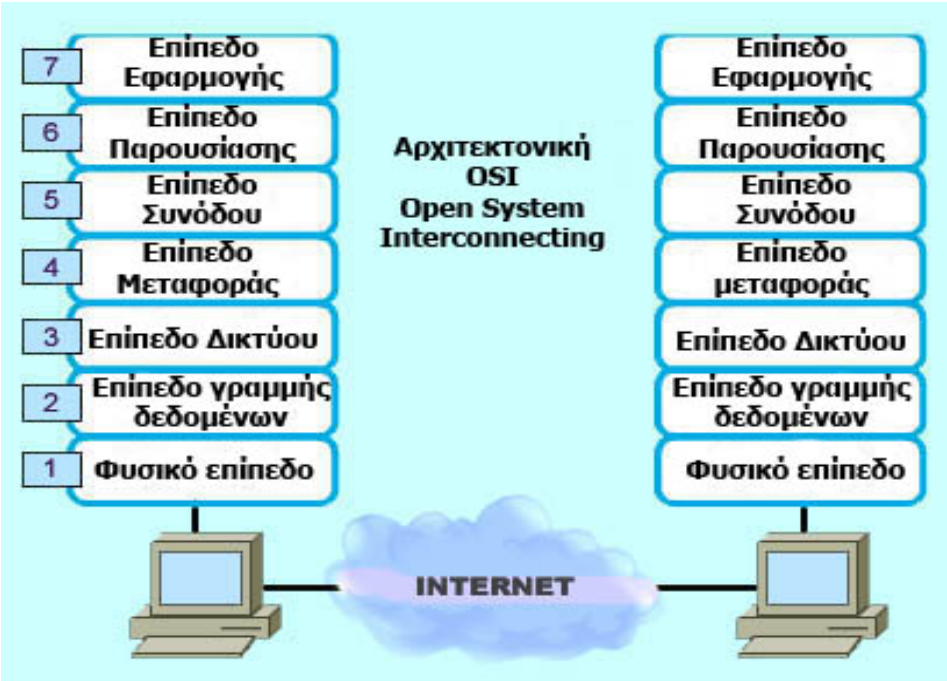
2.4 Πρωτόκολλα SCADA

Σε ένα σύστημα SCADA, η συσκευή RTU δέχεται εντολές για την λειτουργία των σημείων ελέγχου και ανταποκρίνεται σε αιτήματα. Επίσης παρέχει στην MTU του συστήματος SCADA, την κατάσταση των συσκευών και αναλογικά δεδομένα. Τα δεδομένα αυτά αποστέλλονται με μοναδική διευθυνσιοδότηση, η οποία συσχετίζεται με την βάση δεδομένων της MTU. Με την μοναδική διευθυνσιοδότηση κάθε συσκευή έχει την δική της μοναδική διεύθυνση στο τοπικό δίκτυο, όμως η RTU δεν έχει καμιά γνώση για τους μοναδικούς παραμέτρους που παρουσιάζονται στον πραγματικό κόσμο. Ο κεντρικός σταθμός SCADA (MTU) είναι το κομμάτι του συστήματος που πρέπει να γνωρίζει ότι το πρώτο μέρος του μηνύματος από κάποια RTU, με συγκεκριμένο αριθμό, είναι οι πληροφορίες για την μοναδική διαδρομή που ακολούθησε από τον συγκεκριμένο υποσταθμό. Αυτό αντιπροσωπεύει την κύρια χρησιμότητα των συστημάτων SCADA και των πρωτοκόλλων στην βιομηχανία και όχι μόνο.

Τα πρωτόκολλα είναι σχεδιασμένα να μεταφέρουν αναφορές που περιέχουν την κατάσταση με όλες τις συσκευές εισόδου και εξόδου στο δίκτυο. Κάθε πρωτόκολλο αποτελείτε από δυο σύνολα ή ζεύγη μηνυμάτων. Το ένα ζεύγος περιέχει τις απαραίτητες πληροφορίες για την εισαγωγή και απόκριση της MTU στο δίκτυο και το άλλο ζεύγος, το οποίο είναι το πρωτόκολλο της RTU, περιέχει τις πληροφορίες με της οποίες μια RTU μπορεί να συνδεθεί και να αποκριθεί. Στις περισσότερες περιπτώσεις τα ζεύγη αυτά μπορούν να θεωρηθούν μια αίτηση για πληροφορίες και μια απόκριση επιβεβαίωσης.

Όλες οι αρχιτεκτονικές δικτύων βασίζονται στο μοντέλο επτά επιπέδων OSI (Open Systems Interconnection – Διασύνδεση Ανοικτών Συστημάτων), το οποίο είναι πρότυπο του Διεθνή Οργανισμού Προτύπων (International Standards Organization –ISO). Ο σκοπός του μοντέλου OSI είναι να δημιουργεί ένα πλαίσιο που θα επιτρέπει σε κάθε σύστημα ή δίκτυο να συνδέεται και να ανταλλάσει σήματα, πακέτα μηνυμάτων και διευθύνσεις. Το μοντέλο OSI φαίνεται στην Εικόνα 2.11.

Σε γενικές γραμμές, τα κάτω τέσσερα επίπεδα καλύπτουν τη φυσική καλωδίωση, τα πρωτόκολλα δικτύου και επικοινωνίας των τοπικών δικτύων και των δικτύων ευρείας περιοχής, όπως το Ethernet. Τα επίπεδα Συνόδου και Παρουσίασης συνήθως ασχολούνται με την ίδρυση και στη συνέχεια την περάτωση της συνόδου μεταξύ των δύο άκρων. Από το επίπεδο εφαρμογής και πάνω, είναι εκεί όπου ένα τυπικό RTU πρωτόκολλο παρέχει τα δεδομένα σε ένα σταθμό εργασίας ή εξυπηρετητή SCADA από τις συσκευές RTU και τα τοπικά PLC συστήματα.



Εικόνα 2.11 Αρχιτεκτονική 7 επιπέδων OSI

Ένα μεγάλο μέρος από το σχεδιασμό και του ποιο πολύπλοκου συστήματος SCADA περιλαμβάνει την εναρμόνιση των πρωτοκόλλων και των παραμέτρων επικοινωνίας μεταξύ των συνδεμένων συσκευών. Υπάρχουν περίπου 200 πρωτόκολλα επιπέδου εφαρμογής και χρήστη, με κάποια να είναι πνευματικά κατοχυρωμένα και άλλα όχι. Η βιομηχανία απομακρύνεται στις μέρες μας από τα παλιά πρωτόκολλα και από όσα είναι με κατοχυρωμένα δικαιώματα. Τα παρακάτω πρωτόκολλα RTU επικρατούν σαν κύρια πρότυπα για τα νέα συστήματα SCADA.

2.4.1 MODBUS

Το Modbus είναι πρωτόκολλο επιπέδου εφαρμογής και αναπτύχθηκε αρχικά από την Modicon για την μεταφορά δεδομένων από τους ελεγκτές της. Η μεταφορά των δεδομένων οργανωνόταν σε καταχωρητές των 16bit (μορφή ακεραίου) ή ως πληροφορίες κατάστασης σε bytes δεδομένων, γι' αυτό το λόγο δεν μπορούσε να μεταχειριστεί μεγάλους θετικούς και αρνητικούς αριθμούς. Με τα χρόνια το πρωτόκολλο επεκτάθηκε και έχει υιοθετηθεί και από άλλους κατασκευαστές. Επίσης νέοι τύποι δεδομένων προστέθηκαν, ειδικά για να επιτευχθεί υψηλότερη ανάλυση για τις τιμές που μεταδίδονται.

Το Modbus είναι πρωτόκολλο που χρησιμοποιεί έναν μόνο κεντρικό υπολογιστή σε κάθε σύστημα. Ο κεντρικός υπολογιστής ελέγχει πλήρως την μετάδοση και την παρακολουθεί όταν συμβεί λήξη χρονικού όριο δηλαδή δεν υπάρχει απάντηση από την συσκευή που απευθύνεται. Οι συνδεδεμένες συσκευές είναι τύπου "slave" και έχουν τη δυνατότητα να στέλνουν μηνύματα μόνο κατόπιν αιτήματος της MTU. Κατά την διάρκεια της επικοινωνίας σε ένα δίκτυο MODBUS το πρωτόκολλο καθορίζει το πως κάθε ελεγκτής θα γνωρίζει την διεύθυνση μιας συσκευής, θα αναγνωρίσει ένα μήνυμα που προορίζεται γι' αυτόν, θα αποφασίζει την ενέργεια που πρέπει να εκτελεστεί και θα εξάγει τις πληροφορίες που σχετίζονται μαζί του.

2.4.2 MODBUS X

Το Modbus X, το οποίο είναι μια επέκταση του Modbus, υιοθετήθηκε από ορισμένες εταιρείες, εφαρμογές και προμηθευτές συστημάτων SCADA. Το πρωτόκολλο αυτό διόρθωσε τα ελαττώματα του Modbus, το έκανε αναγνώσιμο από τους χρήστες και ικανό να μεταχειρίζεται αρνητικούς και θετικούς αριθμούς μέχρι και 9 ψηφία με τάξη εκθέτη -99 έως +99. Σχεδιάστηκε για να διαβάζει τις μονάδες εισόδου και να γράφει στις μονάδες εξόδου μιας συσκευής PLC και επιτρέπει να χειρίζονται μεγάλες μεταβλητές στο επίπεδο ASCII. Με το καθολικό αναβαθμισμένο πρωτόκολλο Modbus X δεν είναι πια αναγκαίο να γίνονται πειραματισμοί με διάφορες επεκτάσεις του πρωτοκόλλου Modbus.

2.4.3 DNP

Το πρωτόκολλο DNP (Distributed Network Protocol – Πρωτόκολλο Κατανεμημένου Δικτύου) είναι ένα σύνολο από πρωτόκολλα επιπέδου εφαρμογής και ζεύξης δεδομένων, που χρησιμοποιούνται μεταξύ των εξαρτημάτων σε συστήματα αυτοματοποίησης της διαδικασίας. Η κύρια χρήση του είναι σε επιχειρήσεις κοινής ωφελείας, όπως εταιρείες παροχής ηλεκτρικού ρεύματος και νερού.

Το πρωτόκολλο αυτό αναπτύχθηκε για την επικοινωνία μεταξύ διαφόρων τύπων εξοπλισμού ελέγχου και λήψης δεδομένων και έχει περάσει μέχρι σήμερα από διάφορες αναβαθμίσεις. Κατά κύριο λόγο χρησιμοποιείται για την επικοινωνία μεταξύ ενός κεντρικού σταθμού και των RTUs. Το πρωτόκολλο DNP έχει σημαντικά χαρακτηριστικά που το καθιστούν πιο ισχυρό και αποτελεσματικό από τα παλαιότερα πρωτόκολλα όπως το Modbus, με κόστος όμως την κάπως μεγαλύτερη πολυπλοκότητα.

2.4.4 ASCII

Το κύριο πρωτόκολλο των υπολογιστών είναι ο Αμερικάνικος Κώδικας Προτύπων για την Ανταλλαγή Πληροφοριών (American Standard Code for Information Interchange – ASCII), το οποίο

αναπτύχθηκε από τους τηλεγραφικούς κωδικούς και η πρώτη του εμπορική χρήση ήταν ως ένας επταψήφιος κώδικας τηλετυπικής. Οι περισσότεροι υπολογιστές, εκτυπωτές, μόντεμ και πολλοί αισθητήρες, ενεργοποιητές και υπολογιστές ροής επικοινωνούν τώρα σε ASCII.

Ο κώδικας ASCII βασίζεται στον πίνακα κωδικοποιημένων χαρακτήρων που φαίνεται στην Εικόνα 2.12 και περιλαμβάνει ορισμούς για 128 χαρακτήρες. Από αυτούς οι 33 είναι μη εκτυπώσιμοι χαρακτήρες ελέγχου που επηρεάζουν τον τρόπο με τον οποίο υποβάλλεται σε επεξεργασία το κείμενο και οι υπόλοιποι 95 είναι εκτυπώσιμοι χαρακτήρες, συμπεριλαμβανομένου και του κενού το οποίο θεωρείται ένα αόρατο γραφικό.

Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040		Space	64	40	100	@	@	96	60	140	`	`
1	1	001	SOH (start of heading)	33	21	041	!	!	65	41	101	A	A	97	61	141	a	a
2	2	002	STX (start of text)	34	22	042	"	"	66	42	102	B	B	98	62	142	b	b
3	3	003	ETX (end of text)	35	23	043	#	#	67	43	103	C	C	99	63	143	c	c
4	4	004	EOT (end of transmission)	36	24	044	$	$	68	44	104	D	D	100	64	144	d	d
5	5	005	ENQ (enquiry)	37	25	045	%	%	69	45	105	E	E	101	65	145	e	e
6	6	006	ACK (acknowledge)	38	26	046	&	&	70	46	106	F	F	102	66	146	f	f
7	7	007	BEL (bell)	39	27	047	'	'	71	47	107	G	G	103	67	147	g	g
8	8	010	BS (backspace)	40	28	050	(	(	72	48	110	H	H	104	68	150	h	h
9	9	011	TAB (horizontal tab)	41	29	051	)	)	73	49	111	I	I	105	69	151	i	i
10	A	012	LF (NL line feed, new line)	42	2A	052	*	*	74	4A	112	J	J	106	6A	152	j	j
11	B	013	VT (vertical tab)	43	2B	053	+	+	75	4B	113	K	K	107	6B	153	k	k
12	C	014	FF (NP form feed, new page)	44	2C	054	,	,	76	4C	114	L	L	108	6C	154	l	l
13	D	015	CR (carriage return)	45	2D	055	-	-	77	4D	115	M	M	109	6D	155	m	m
14	E	016	SO (shift out)	46	2E	056	.	.	78	4E	116	N	N	110	6E	156	n	n
15	F	017	SI (shift in)	47	2F	057	/	/	79	4F	117	O	O	111	6F	157	o	o
16	10	020	DLE (data link escape)	48	30	060	0	0	80	50	120	P	P	112	70	160	p	p
17	11	021	DC1 (device control 1)	49	31	061	1	1	81	51	121	Q	Q	113	71	161	q	q
18	12	022	DC2 (device control 2)	50	32	062	2	2	82	52	122	R	R	114	72	162	r	r
19	13	023	DC3 (device control 3)	51	33	063	3	3	83	53	123	S	S	115	73	163	s	s
20	14	024	DC4 (device control 4)	52	34	064	4	4	84	54	124	T	T	116	74	164	t	t
21	15	025	NAK (negative acknowledge)	53	35	065	5	5	85	55	125	U	U	117	75	165	u	u
22	16	026	SYN (synchronous idle)	54	36	066	6	6	86	56	126	V	V	118	76	166	v	v
23	17	027	ETB (end of trans. block)	55	37	067	7	7	87	57	127	W	W	119	77	167	w	w
24	18	030	CAN (cancel)	56	38	070	8	8	88	58	130	X	X	120	78	170	x	x
25	19	031	EM (end of medium)	57	39	071	9	9	89	59	131	Y	Y	121	79	171	y	y
26	1A	032	SUB (substitute)	58	3A	072	:	:	90	5A	132	Z	Z	122	7A	172	z	z
27	1B	033	ESC (escape)	59	3B	073	;	;	91	5B	133	[	[	123	7B	173	{	{
28	1C	034	FS (file separator)	60	3C	074	<	<	92	5C	134	\	\	124	7C	174			|
29	1D	035	GS (group separator)	61	3D	075	=	=	93	5D	135	]	]	125	7D	175	}	}
30	1E	036	RS (record separator)	62	3E	076	>	>	94	5E	136	^	^	126	7E	176	~	~
31	1F	037	US (unit separator)	63	3F	077	?	?	95	5F	137	_	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com

Εικόνα 2.12 Πίνακας ASCII

2.4.5 IEC 60870

Αυτό το πρωτόκολλο χρησιμοποιείται κυρίως στα συστήματα μετάδοσης και διανομής ηλεκτρικής ενέργειας. Το IEC 60870-5-101 είναι ένα διεθνές πρότυπο πρωτόκολλου επικοινωνιών για τον Τηλεχειρισμό των συστημάτων μετάδοσης ηλεκτρικής ενέργειας, ο οποίος έχει ευρέως υιοθετηθεί σε πολλές χώρες σε όλο τον κόσμο.

2.4.6 Πρωτόκολλα για Τοπικές Συσκευές

Τα τοπικά δίκτυα και τα πρωτόκολλα τοπικών δικτύων που χρησιμοποιούνται για την μετάδοση δεδομένων από αισθητήρες και άλλες συσκευές στα PLC ή RTU και από τις RTU στα συστήματα SCADA είναι τα :

- Δίκτυα Αισθητήρων: Αυτά είναι απλές βασικές on / off συσκευές που συνδέουν τα δίκτυα.
- Fieldbus Δίκτυα: Αυτά χρησιμοποιούνται για τη σύνδεση αναλογικών και έξυπνων συσκευών πεδίου όπως ενεργοποιητές βαλβίδων, αντλίες και άλλων συστημάτων ελέγχου πεδίου.
- Δίκτυα Ελέγχου: Αυτά χρησιμοποιούνται για peer to peer συνδέσεις μεταξύ συστημάτων ελέγχου, όπως SCADA, DCS, αναλυτές και ασφαλή συστήματα PLC.

3 Συσκευή “Netmaster RS485” της “Elsist”

3.1 Εισαγωγή στις Συσκευές RTU και PLC

Όπως αναφέρεται και στο προηγούμενο κεφάλαιο τα Συστήματα SCADA αποτελούνται από διακομιστές, απομακρυσμένες τερματικές μονάδες και συσκευές παρακολούθησης και έλεγχου του εξοπλισμού και της διαδικασίας. Οι Απομακρυσμένες Τερματικές Μονάδες (RTU) και οι Προγραμματιζόμενοι Λογικοί Ελεγκτές (PLCs) συνήθως συγχέονται μεταξύ τους ενώ παίζουν σημαντικό ρόλο στην σωστή λειτουργία του συστήματος SCADA και χρησιμοποιούνται σε συνδυασμό ή/και μεμονωμένα.

3.1.1 Απομακρυσμένη Τερματική Μονάδα (RTU)

Μια Απομακρυσμένη Τερματική Μονάδα (Remote Telemetry Unit - RTU) είναι μία ηλεκτρονική συσκευή ελεγχόμενη από μικροεπεξεργαστή, η οποία διασυνδέει αντικείμενα από το φυσικό κόσμο με ένα κατανεμημένο σύστημα ελέγχου ή ένα σύστημα εποπτικού έλεγχου και συγκέντρωσης δεδομένων (SCADA). Αυτό το καταφέρνει διαβιβάζοντας δεδομένα τηλεμετρίας σε ένα κεντρικό σύστημα και χρησιμοποιώντας μηνύματα που προέρχονται από το κεντρικό εποπτικό σύστημα για τον έλεγχο των συνδεδεμένων συσκευών. Ένας άλλος όρος που μπορεί να χρησιμοποιηθεί για τις RTU είναι Απομακρυσμένη Μονάδα Τηλεμετρίας (Remote Telemetry Unit), ο όρος χρήσης ποικίλλει ανάλογα με την περιοχή εφαρμογής.

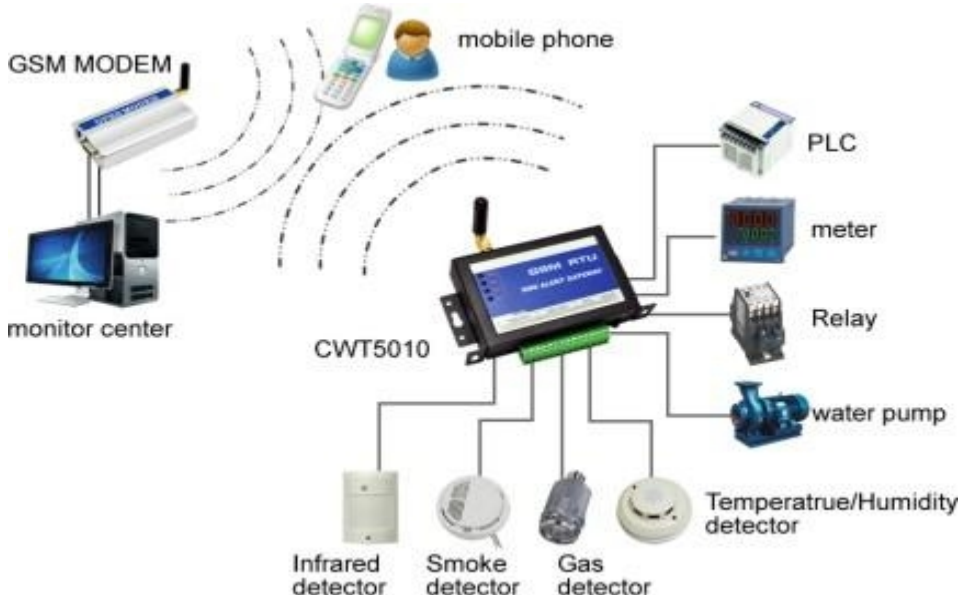
Η RTU όπως, φαίνεται στην Εικόνα 3.1, μετατρέπει εισερχόμενα σήματα από τον πραγματικό κόσμο, όπως πίεση, ροές, τάσεις, ρεύματα, επαφές και παλμούς σε σήματα τα οποία μπορούν να αποσταλούν ενσύρματα ή ασύρματα. Επίσης μετατρέπει εισερχόμενα σήματα από άλλο RTU ή έναν κεντρικό υπολογιστή σε σήματα εξόδου, ώστε να ανοίξουν ή να κλείσουν ηλεκτρονόμοι (relays), βαλβίδες και να ξεκινήσουν ή σταματήσουν κινητήρες.

Μια τυπική RTU για να λειτουργήσει χρειάζεται μια μονάδα τροφοδοσίας, η οποία μετατρέπει το εναλλασσόμενο ρεύμα σε συνεχές και συνήθως περιέχει και μια μπαταρία για να συνεχιστεί η λειτουργία σε περίπτωση διακοπής της παροχής του ρεύματος. Επίσης η RTU περιλαμβάνει έναν κεντρικό επεξεργαστή, την μνήμη και τις μονάδες εισόδου και εξόδου μέσω των οποίων αλληλεπιδρά με τις διάφορες συσκευές. Η RTU μπορεί να αποτελείτε από μια πολύπλοκη κάρτα με διάφορα τμήματα που περιέχουν τα παραπάνω ή μπορεί να αποτελείτε από πολλές κάρτες. Η δομή μιας RTU φαίνεται στην Εικόνα 3.2.

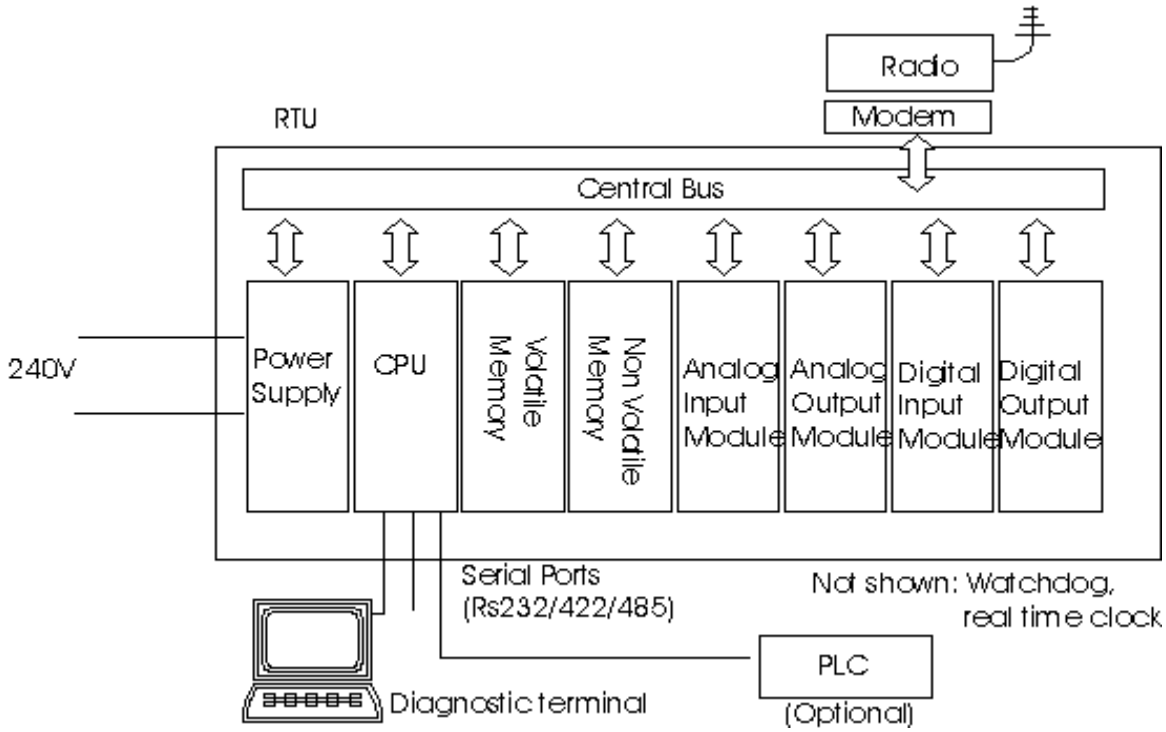
Οι κάρτες διασύνδεσης ή αλλιώς τα κυκλώματα εισόδου και εξόδου μιας RTU μπορεί να είναι ένα ή περισσότερα και σε συνδυασμό από τα παρακάτω:

- Ψηφιακή Είσοδος: Χρησιμοποιείται για την λήψη ψηφιακής πληροφορίας από τον πραγματικό κόσμο ή οποία ονομάζεται ψηφιακή διότι έχει δύο καταστάσεις: ON ή OFF. Όπως για παράδειγμα αν ένας διακόπτης είναι ανοιχτός ή κλειστός και αν μια αντλία δουλεύει ή όχι.
- Αναλογική Είσοδος: Μια RTU μπορεί να λαμβάνει πληροφορίες από αισθητήρες σχετικά με ταχύτητες ροής υγρών, τάσεις, ηλεκτρικά ρεύματα, πιέσεις και στάθμες υγρών. Αυτός ο τύπος πληροφορίας ονομάζεται αναλογικός καθώς τα ηλεκτρικά σήματα που λαμβάνει η RTU είναι ανάλογα των μετρούμενων φυσικών μεγεθών και έρχονται από μετατροπείς στα 4-20 mA.
- Παλμική Είσοδος: Η RTU λαμβάνει επίσης παλμούς, οι οποίοι αποστέλλονται από μετρήσεις για παράδειγμα ηλεκτρικής ενέργειας ή ροής υγρών. Κάθε παλμός αντιστοιχεί σε μια μονάδα μέτρησης του μεγέθους, η RTU αθροίζει τους παλμούς αυτούς και στη συνέχεια απεικονίζει το συνολικό άθροισμα που προκύπτει (accumulated), όπου είναι επίπεδων τάσεως +5V.
- Ψηφιακή Έξοδος: Μέσω των ψηφιακών εξόδων της η RTU στέλνει εντολές ελέγχου όπως άνοιγμα ή κλείσιμο βαλβίδων, εκκίνηση ή σταμάτημα κινητήρων και άνοιγμα ή κλείσιμο διακοπτών ισχύος.

- Αναλογική Έξοδος: Σήμα εντολής θέσης (set point) μιας βαλβίδας, ένα αναλογικό σήμα θέσης σε ένα PLC ή ένα σήμα 4-20 mA σε ένα καταγραφικό, απαιτούν η RTU να δώσει στην έξοδο της ένα μεταβλητό ρεύμα (4-20 mA) ανάλογο της εντολής από τον κεντρικό Η/Υ ή της αναλογικής εισόδου από κάποιο άλλο RTU.
- Παλμική Έξοδος: Η RTU με κάποιο πρόγραμμα επιτρέπει στα ψηφιακά κυκλώματα εξόδου να δώσουν παλμούς ακριβώς αντίστοιχους με τους παλμούς εισόδου του RTU σε κάποιο άλλο άκρο.



Εικόνα 3.1 Πιθανές διασυνδέσεις μιας RTU



Εικόνα 3.2 Δομή της RTU

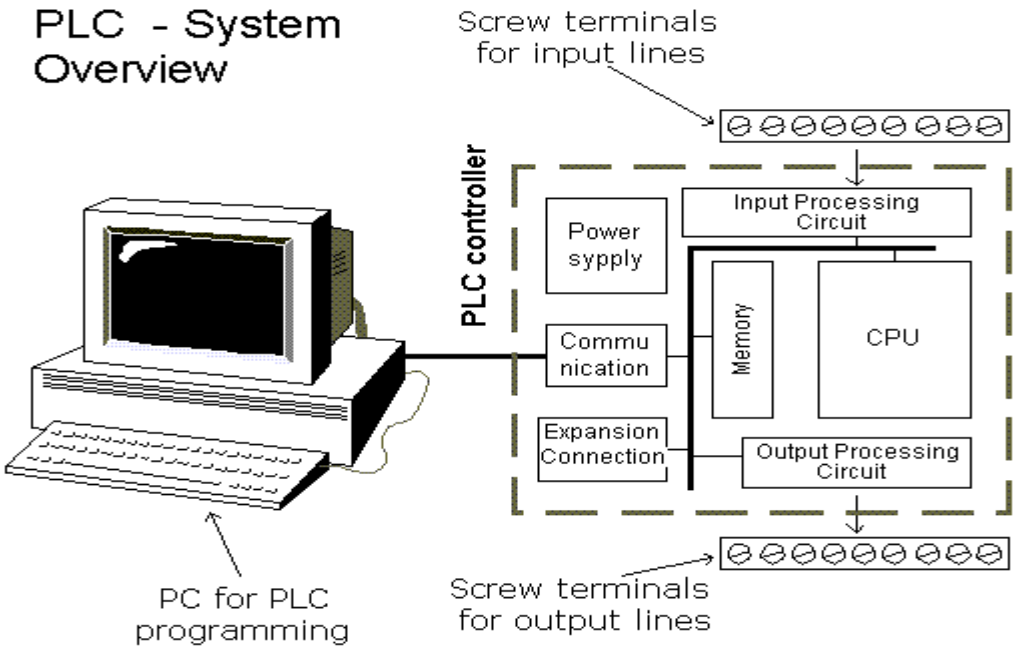
3.1.2 Προγραμματιζόμενος Λογικός Ελεγκτής (PLC)

Ένας Προγραμματιζόμενος Λογικός Ελεγκτής (Programmable Logic Controller - PLC) είναι ένα ψηφιακό ηλεκτρονικό σύστημα που χρησιμοποιείται για την αυτοματοποίηση των ηλεκτρομηχανολογικών διεργασιών. Ο PLC χρησιμοποιεί μια προγραμματιζόμενη μνήμη για την αποθήκευση εντολών ώστε να επιτελούνται διάφορες λειτουργίες, όπως χρονικές ή μετρητικές πράξεις και για να ελέγχονται, μέσω αναλογικών ή ψηφιακών μονάδων, διάφορες μηχανές ή διαδικασίες. Σε αντίθεση με τους υπολογιστές γενικής χρήσης, ο PLC είναι σχεδιασμένος για πολλαπλές ρυθμίσεις εισόδου και εξόδου, εκτεταμένο πεδίο τιμών θερμοκρασίας, ανθεκτικότητα στον ηλεκτρικό θόρυβο και αντοχή στους κραδασμούς.

Ένας PLC είναι ένα σύστημα πραγματικού χρόνου, καθώς τα αποτελέσματα εξόδου πρέπει να παράγονται σε απόκριση προς τις συνθήκες εισόδου μέσα σε ένα περιορισμένο χρονικό διάστημα, άλλως θα προκύψει εσφαλμένη λειτουργία. Γι’ αυτό το λόγο ο PLC είναι συνδεμένος σε διάφορα σημεία της παραγωγής και μέσω αισθητήρων λαμβάνει σήματα, τα οποία επεξεργάζεται η κεντρική μονάδα σύμφωνα με τις αποθηκευμένες εντολές στην μνήμη. Η CPU εκτελεί λογικές και αριθμητικές πράξεις και τα αποτελέσματα μεταβιβάζονται στις εξόδους του ελεγκτή και έτσι γίνεται συνεχής έλεγχος και υπάρχει η δυνατότητα πρόβλεψης για την αντιμετώπιση βλαβών και σφαλμάτων.

Κάθε προγραμματισμένος ελεγκτής, όπως φαίνεται και στην Εικόνα 3.3, αποτελείτε από τα εξής κύρια μέρη :

- Το πλαίσιο τοποθέτησης μονάδων, όπου είναι τοποθετημένες οι μονάδες ώστε να τους παρέχεται τροφοδοσία και επικοινωνία μεταξύ τους και με το κεντρικό δίκτυο.
- Την μονάδα τροφοδοσίας που παρέχει το επίπεδο τροφοδοσίας το οποίο χρειάζονται οι ελεγκτές για να λειτουργήσουν. Μετατρέπει το εναλλασσόμενο ρεύμα 110V ή 240V στο συνεχές που απαιτείται από την κεντρική μονάδα επεξεργασίας, την μνήμη και τις μονάδες εισόδου/εξόδου.
- Την Κεντρική Μονάδα Επεξεργασίας (CPU), η οποία λαμβάνει πληροφορίες από τις μονάδες εισόδου, τις επεξεργάζεται σύμφωνα με το αποθηκευμένο πρόγραμμα και ανανεώνει την πληροφορία στις εξόδους.
- Την Μνήμη αποτελείτε από τον επεξεργαστή και την Μνήμη Τυχαίας Προσπέλασης (RAM).
- Τις συσκευές εισόδου που λαμβάνουν σήματα από αισθητήρες, κουμπιά και διακόπτες και τα μετατρέπουν σε τέτοια μορφή ώστε να μπορεί να τα επεξεργαστεί η CPU.
- Τις συσκευές εξόδου, οι οποίες δέχονται τα σήματα από την CPU και τα μεταφράζουν σε μορφή κατάλληλη ώστε να εκτελεστούν λειτουργίες έλεγχου από εξωτερικές συσκευές.



Εικόνα 3.3 Η Δομή ενός PLC

3.1.3 Σύγκριση RTUs και PLCs

Οι RTUs και οι PLCs έχουν και οι δύο την ίδια βασική λειτουργία, την απομακρυσμένη παρακολούθηση και τον έλεγχο απομακρυσμένου εξοπλισμού. Όμως ποικίλλει σημαντικά ο τρόπος που το καταφέρνουν αυτό και γι’ αυτό το λόγο οι RTUs είναι πιο κατάλληλες για ευρεία γεωγραφική τηλεμετρία χρησιμοποιώντας ασύρματες επικοινωνίες. Ενώ οι PLCs είναι πιο κατάλληλοι για τον τοπικό έλεγχο χώρου, όπου το σύστημα χρησιμοποιεί φυσικά μέσα για τον έλεγχο. Αυτός είναι ένας τομέας όπου υστερούν οι RTUs διότι δεν υποστηρίζουν αλγορίθμους ή βρόχους ελέγχου.

Οι Απομακρυσμένες Συσκευές Τηλεμετρίας (RTUs) έχουν αναπτυχθεί περισσότερο σε βιομηχανικά δίκτυα τηλεπικοινωνιών τα οποία χρησιμοποιούνται συνήθως από εταιρείες τηλεπικοινωνιών, εταιρείες μεταφορών όπως οι σιδηρόδρομοι και από κυβερνητικούς οργανισμούς όπως η αστυνομία και η πυροσβεστική. Τα δίκτυα αυτά εκτίνονται σε μεγάλες γεωγραφικές περιοχές και η χρήση των PLC είναι περισσότερο δαπανηρή σε σχέση με τις RTUs, διότι χρειάζονται περισσότερα από δέκα PLC για να χειριστούν τις λειτουργίες που ελέγχει μια μόνο RTU.

Οι τοποθεσίες εγκατάστασης των RTUs μπορεί να είναι εξαιρετικά απομακρυσμένες και με αντίξοες περιβαλλοντικές συνθήκες, όπως μια χιονισμένη κορυφή ενός βουνού ή μια απλή εγκατάσταση στην έρημο. Οι RTUs έχουν ένα πλεονέκτημα σε σχέση με τους PLCs καθώς ο εξοπλισμός σε ένα τέτοιο περιβάλλον θα πρέπει να είναι σε θέση να χειριστεί ένα ευρύ φάσμα θερμοκρασιών. Επίσης σε αυτοτροφοδοτούμενα συστήματα ο εξοπλισμός πρέπει να διαρκέσει περισσότερο πριν τεθεί εκτός λειτουργίας ώστε αποφθεχθούν μεγαλύτερες συνέπειες.

Σε ένα περιβάλλον παραγωγής όπως ένα εργοστάσιο, που χρησιμοποιούνται PLCs, η αδυναμία παρακολούθησης και ελέγχου του συστήματος δεν είναι ένα τεράστιο πρόβλημα, καθώς είναι δυνατή η γρήγορη μετάβαση στο σημείο του προβλήματος και η ελαχιστοποίηση της αποτυχίας της διεργασίας. Στον τομέα όμως των τηλεπικοινωνιών, δεν είναι τόσο εύκολο, καθώς σε μερικά σημεία απαιτείται ακόμη και ελικόπτερο για να γίνει προσέγγιση του εξοπλισμού. Οι RTUs έχουν σχεδιαστεί για να παρέχουν υψηλή αξιοπιστία ώστε να αποφευχθούν οι δαπανηρές και χρονοβόρες μετακινήσεις για την επισκευή τους.

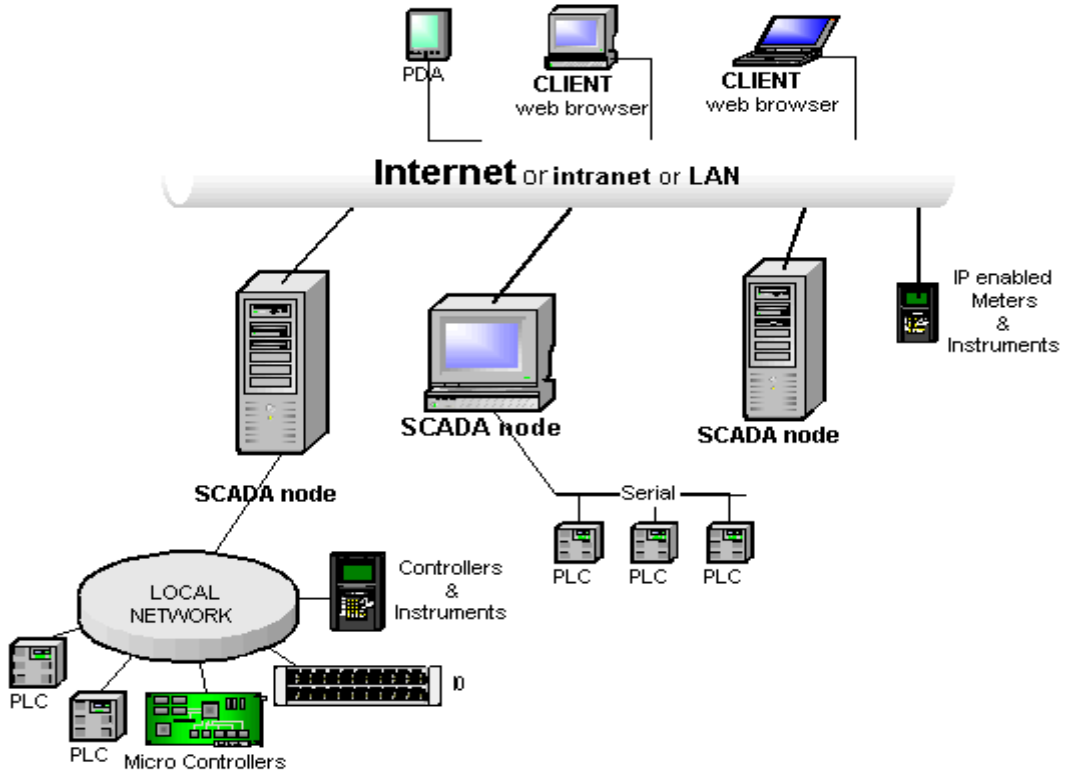
Σε αντίθεση οι PLCs είναι περισσότερο δημοφιλής στη Βιομηχανία. Αν και κάθε RTU θα μπορούσε να χρησιμοποιηθεί σε μια μικρή περιοχή, ένας PLC έχει το πλεονέκτημα εδώ επειδή είναι άνευ σημασίας οι πολλαπλές είσοδοι και έξοδοι που διαθέτει μια RTU. Σε ένα περιορισμένο χώρο, μπορεί να συνδεθεί κάθε μεμονωμένος PLC στο δίκτυο ξεχωριστά, χωρίς να χάνουμε πολλούς πόρους. Το γεγονός αυτό καθιστά την χαμηλή τιμή πλεονέκτημα των PLCs.

Επίσης στους PLCs μπορεί να τροποποιηθεί εύκολα το λογισμικό ή το υλικό τους για να αντιμετωπιστούν τροποποιημένες απαιτήσεις και λόγω της έμφασης των PLCs στο λογισμικό έχει γίνει ευκολότερος ο σχεδιασμός και η εγκατάσταση των συστημάτων SCADA. Οι PLCs επιτρέπουν πολύ πιο εξελιγμένο έλεγχο από τις RTUs, κυρίως λόγω της ικανότητας του λογισμικού που επιτρέπει σαφής αναφορά των προβλημάτων. Για το λόγο αυτό κάνει πιο εύκολη και ταχεία την διάγνωση προβλημάτων υλικού και λογισμικού στο σύστημα, καθώς και τον εντοπισμό των προβλημάτων με τη διαδικασία και το σύστημα αυτοματισμού.

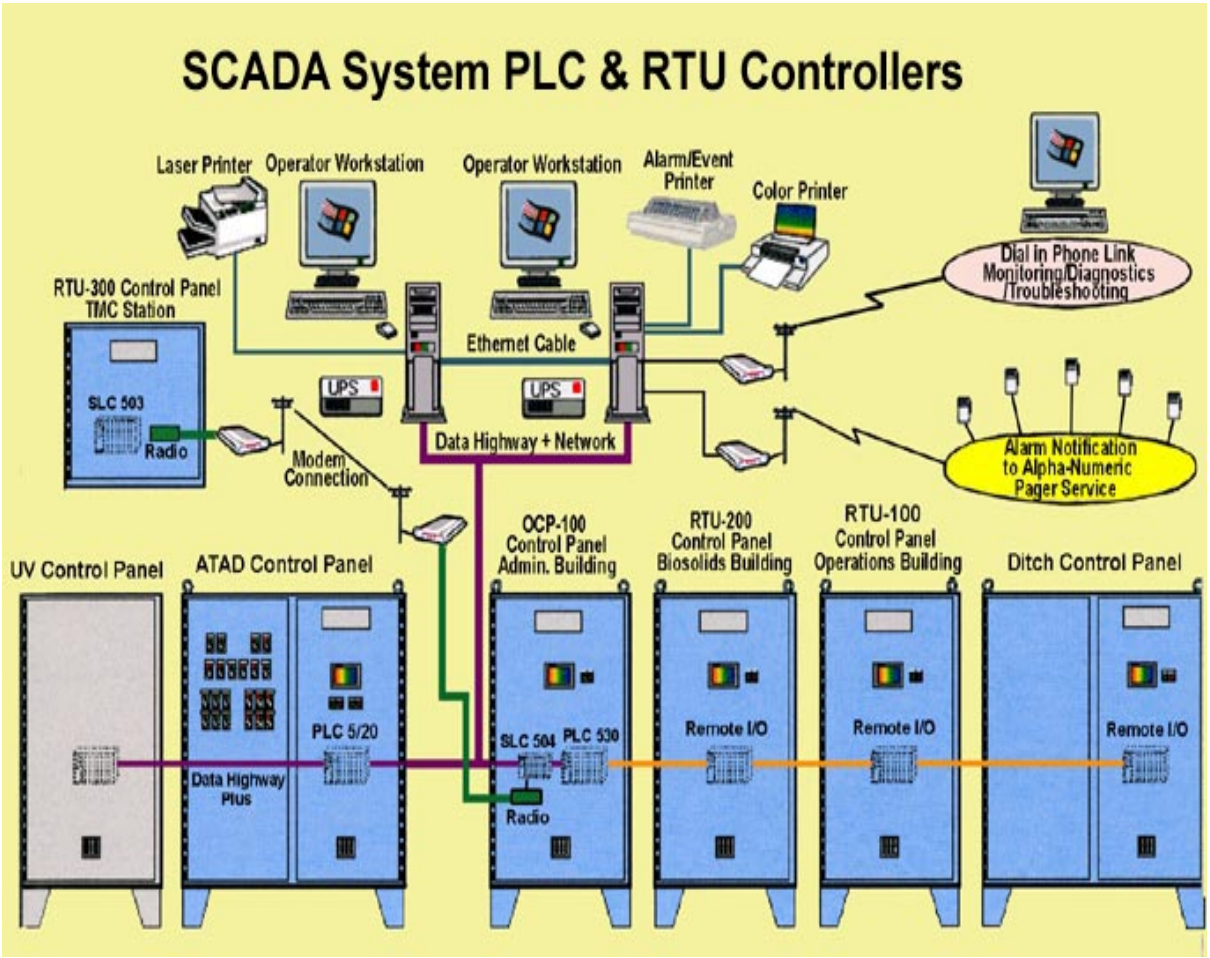
3.1.4 Συνεργασία RTUs και PLCs

Πολλές απομακρυσμένες εγκαταστάσεις χρησιμοποιούν RTUs λόγω των παραπάνω πλεονεκτημάτων τους και επιπλέον απαιτούν αυτόματο έλεγχο σε ένα σημείο. Σε αυτές τις περιπτώσεις προσθέτετε ένας PLC στην RTU, από την οποία παίρνει εντολές που έχουν ληφθεί από τον κεντρικό σταθμό και τροποποιεί τις λειτουργίες του δηλαδή συχνά αναθέτουμε στο PLC να λαμβάνει ψηφιακές ή αναλογικές επιθυμητές τιμές από την RTU.

Με τις αυξανόμενες απαιτήσεις στην βιομηχανία και όχι μόνο η συνεργασία των RTUs και PLCs έχει γίνει αναγκαία και συμφέρουσα λύση. Έτσι χρησιμοποιείτε συνήθως το RTU για εκθέσεις δεδομένων που έχει συλλέξει από τους αισθητήρες, για αναφορές συγκεντρωτικών μετρήσεων και αποτελεσμάτων και για απομακρυσμένες εντολές προς τις διάφορες συσκευές ελέγχου του συστήματος. Ενώ οι PLCs χρησιμοποιούνται περισσότερο για τον τοπικό αυτοματοποιημένο έλεγχο συσκευών όπως μηχανισμούς ελέγχου βαλβίδων, κινητήρων και ροής υγρών. Συστήματα SCADA με την συνεργασία τόσο RTUs όσο και PLCs φαίνονται στην Εικόνα 3.4 και στην Εικόνα 3.5.



Εικόνα 3.4 Συνεργασία RTUs και PLCs



Εικόνα 3.5 Σύστημα SCADA με RTUs και PLCs

3.2 Χαρακτηριστικά του Netmaster RS485 (2nd Series)

Η συσκευή Netmaster RS485 της Ιταλικής εταιρείας Elsist είναι ένας προγραμματιζόμενος ελεγκτής που βασίζεται στην μικροσκοπική μονάδα διασύνδεσης διαδικτύου (Tiny Internet Interface - TINI) με ονομασία “Dallas Semiconductor”, προϊόν της Αμερικάνικης εταιρείας “Maxim Integrated”. Το περίβλημα του είναι το πρότυπο κέλυφος “DIN 43880” και χρησιμοποιεί μία δομή διπύρηνου επεξεργαστή. Η συνδεσιμότητα του με το δίκτυο επιτρέπει την εύκολη ενσωμάτωση του με άλλα συστήματα και προσφέρει μια ευέλικτη και φθηνή λύση για απομακρυσμένο έλεγχο και απόκτηση δεδομένων μέσω διαφόρων δικτύων. Η δεύτερη γενιά είναι εξοπλισμένη με πιο ισχυρό επεξεργαστή, καλύτερη διεπαφή για τις εισόδους και τις εξόδους και ένα πραγματικό PLC, το οποίο “τρέχει” σε ξεχωριστό επεξεργαστή ώστε να υπάρχει καλύτερη υποστήριξη των εφαρμογών πραγματικού χρόνου.

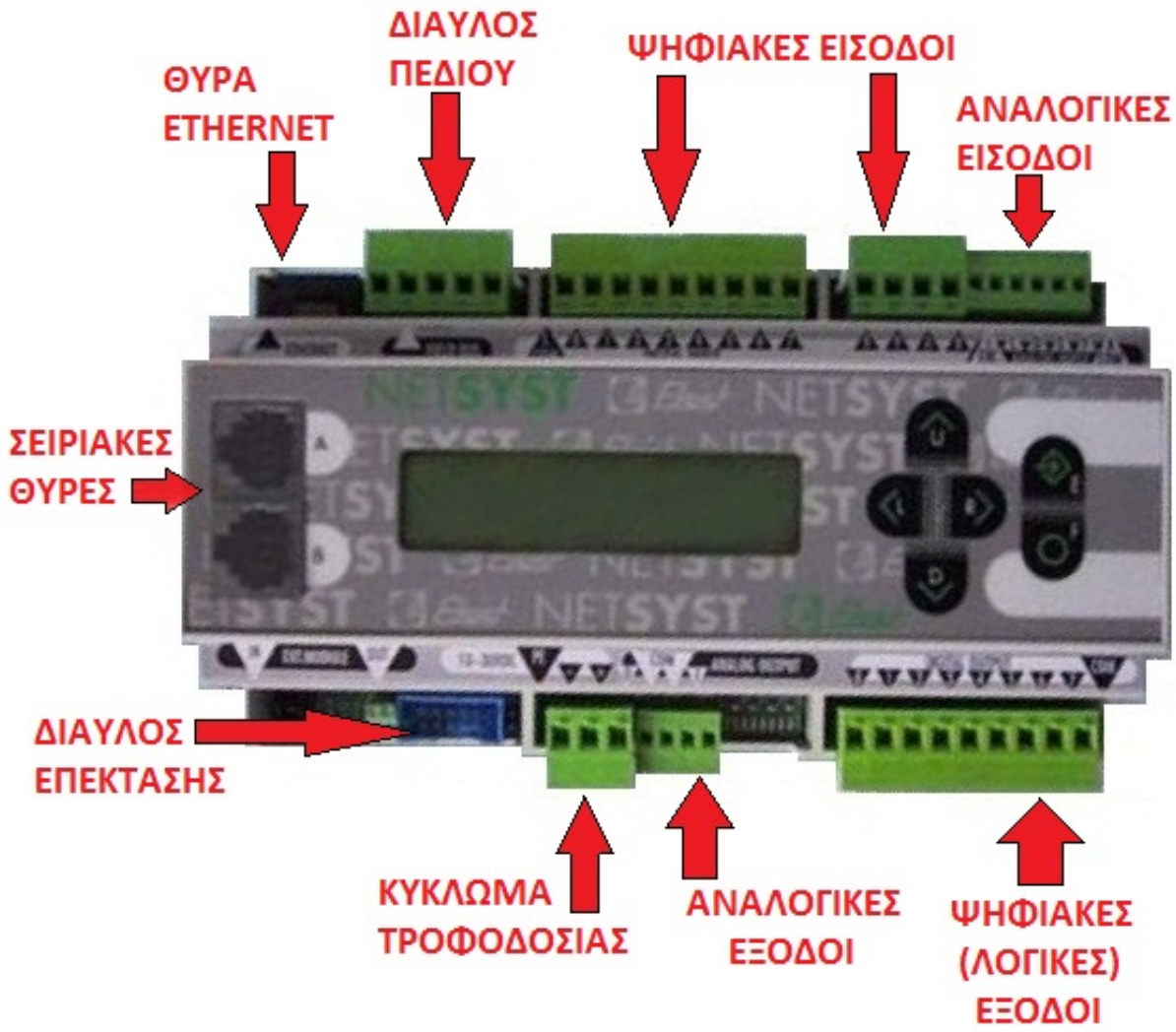
Ο Netmaster τροφοδοτείται με μία πηγή συνεχούς ρεύματος (Direct Current - DC) σε μια ευρεία κλίμακα 10 Volt έως 30 Volt και με χαμηλή απορρόφηση, για το λόγο αυτό η χρησιμοποίηση του απλοποιεί τη δημιουργία συστημάτων όπου είναι απαραίτητη η παροχή ρεύματος από μπαταρία. Επίσης είναι εξοπλισμένος με έξι πλήκτρα και μια αλφαριθμητική οθόνη υγρών κρυστάλλων (Liquid Crystal Display – LCD) 2 γραμμών, με μέγιστη δυνατότητα 16 χαρακτήρες ανά γραμμή. Το γεγονός αυτό, σε συνδυασμό με τη δυνατότητα να αναπτυχτούν προγράμματα σε γλώσσα προγραμματισμού C, επιτρέπει τη δημιουργία προσαρμοσμένων εφαρμογών με χαμηλό κόστος, οι οποίες μπορούν να προσφέρουν λειτουργίες που παρέχονται συνήθως μόνο σε πιο ακριβά προϊόντα.

Ο Netmaster είναι εξοπλισμένος με ψηφιακές και αναλογικές εισόδους και εξόδους , με συνδέσεις μέσω δίαυλου, με βύσματα μετατόπισης μόνωσης (Insulation Displacement Connectors – IDCs) για την σύνδεση μονάδων επέκτασης και με RJ45 υποδοχές για σύνδεση RS232 και Ethernet. Η μονάδα Netmaster και οι συνδέσεις της παρουσιάζονται στην Εικόνα 3.6, τα χαρακτηριστικά των συνδέσεων αυτών είναι τα εξής:

- Ψηφιακές Είσοδοι : Η συσκευή εφοδιάζεται με 12 απομονωμένες ψηφιακές εισόδους που ενεργοποιούνται με σήματα κλίμακας 10 έως 30 Volt DC. Οι είσοδοι μπορεί να είναι PNP ή NPN και η κατάσταση του καθενός εμφανίζεται με μια δίοδο εκπομπής φωτός (Light Emitting Diode – LED). Η τάση που χρειάζεται ώστε να ενεργοποιηθούν οι είσοδοι πρέπει να είναι συνεχής (DC), φιλτραρισμένη και κάτω από το ανώτατο επιτρεπόμενο όριο.
- Αναλογικές Είσοδοι : Οι διαθέσιμες αναλογικές είσοδοι είναι 4, με κοινό τρόπο σύνδεσης ή 2 με διαφοροποιημένη σύνδεση. Η τάση που εφαρμόζεται είναι μέσα στο εύρος 0 έως 10 V και η ανάλυση του μετατροπέα είναι 16 bit.
- Ψηφιακές Έξοδοι : Η μονάδα είναι εξοπλισμένη με 8 ρελέ ή στατικούς εξόδους, για τις οποίες παρέχεται ένα κοινό βύσμα. Η κατάσταση της κάθε εξόδου εμφανίζεται με LED. Σε κάθε εκκίνηση του Netmaster όλες οι έξοδοι επαναφέρονται στην αρχική κατάσταση που είναι ίση με μηδέν δηλαδή το LED είναι σβησμένο.
- Αναλογικές Έξοδοι : Η μονάδα είναι εξοπλισμένη με 2 αναλογικές εξόδους, όπου η αναλογική έξοδος 0 ρυθμίζεται μέσω ενός 12bit DAC και η έξοδος 1 ρυθμίζεται μέσω ενός 16bit PWM. Το εύρος τάσης εξόδου είναι 0 έως 10Volt DC.
- Δίαυλοι Επέκτασης : Ο δίαυλος επικοινωνίας με τις μονάδες επέκτασης χρησιμοποιεί την διασύνδεση I2C και είναι διαθέσιμο σε βύσμα 10 IDC. Η διευθυνσιοδότηση των μονάδων επέκτασης είναι αυτόματη καθώς το λογισμικό αναγνωρίσει αν μια μονάδα επέκτασης είναι συνδεδεμένη ή όχι και της απονέμει μια διεύθυνση ίση με την διεύθυνση της προηγούμενης μονάδας συν ένα. Ο δίαυλος επέκτασης έχει την δυνατότητα διασύνδεσης 1-Wire δηλαδή επιτρέπει να συνδέονται όλες τις συσκευές που είναι συμβατές με αυτή την διασύνδεση.
- Σειριακές Θύρες : Η μονάδα είναι εξοπλισμένη με 2 σειριακές θύρες DTE (Data Terminal Equipment – Εξοπλισμός Τερματικών Δεδομένων). Η σύνδεση μεταξύ DTEs και προσωπικών υπολογιστών ή τερματικά χειρισμού γίνεται μέσω ενός καλωδίου τύπου nullmodem με μέγιστο μήκος 15 μέτρα.
- Δίαυλος Πεδίου : Η μονάδα είναι εξοπλισμένη με μια διασύνδεση διαύλου πεδίου, η οποία ανάλογα με την εντολή κώδικα, μπορεί να είναι τύπου CAN ή τύπου RS485.
- Θύρα Ethernet : Η συσκευή Netmaster εφοδιάζεται με μια 10/100-BaseT θύρα Ethernet. Οι συνδέσεις είναι συμβατές με το πρότυπο IEEE 802.3 Ethernet 100 BaseT και για να συνδεθεί

η μονάδα Netmaster σε ένα τοπικό δίκτυο Ethernet χρειάζεται ένα καλώδιο UDP Cat.5 και ένα HUB. Ενώ για την πραγματοποίηση μιας σύνδεσης από σημείο σε σημείο είναι αρκετό ένα καλώδιο cross RJ45 χωρίς HUB.

Η διασύνδεση Ethernet επιτρέπει να εγκαταστήσετε τον Netmaster στο δίκτυο μιας εταιρείας και εγγυάται την αλληλεπίδραση του προϊόντος με όλους τους πόρους του δικτύου. Η υποστήριξη του πρωτοκόλλου TCP-IP σε συνδυασμό με έναν εξυπηρετητή διαδικτύου κάνει εφικτή τη σύνδεση της συσκευής Netmaster με το διαδίκτυο. Με τον τρόπο αυτό είναι δυνατόν, από οποιοδήποτε υπολογιστή συνδεδεμένο στο δίκτυο, να λαμβάνετε πληροφορίες και να στέλνεται εντολές στην μονάδα Netmaster από ένα πρόγραμμα περιήγησης χωρίς τη χρήση ειδικών προγραμμάτων.



Εικόνα 3.6 Η Συσκευή Netmaster RS485 και οι Συνδέσεις της

3.3 Πρωτόκολλο Επικοινωνίας του Netmaster RS485 (2nd Series)

Ο προγραμματιζόμενος ελεγκτής Netmaster RS485, όπως και οι περισσότερες συσκευές σε ένα σύστημα SCADA, χρησιμοποιεί το πρωτόκολλο ASCII για την επικοινωνία του με τον κεντρική τερματική μονάδα επεξεργασίας (MTU) και τις RTU. Κατά την επικοινωνία με την MTU, ο Netmaster στέλνει και λαμβάνει μηνύματα από ένα λογισμικό διασύνδεσης (HMI), σε μορφή πακέτων δεδομένων. Τα πακέτα δεδομένων αυτά πριν αποσταλούν, κωδικοποιούνται βάση του πίνακα κωδικοποιημένων χαρακτήρων του κώδικα ASCII και συνθέτονται βάση μιας συγκεκριμένης δομής για να είναι εφικτή η σωστή επικοινωνία. Η δομή αυτή των πακέτων που λαμβάνει και στέλνει ο Netmaster διαιρείται σε ψηφιολέξεις (Bytes) και αναλύεται παρακάτω.

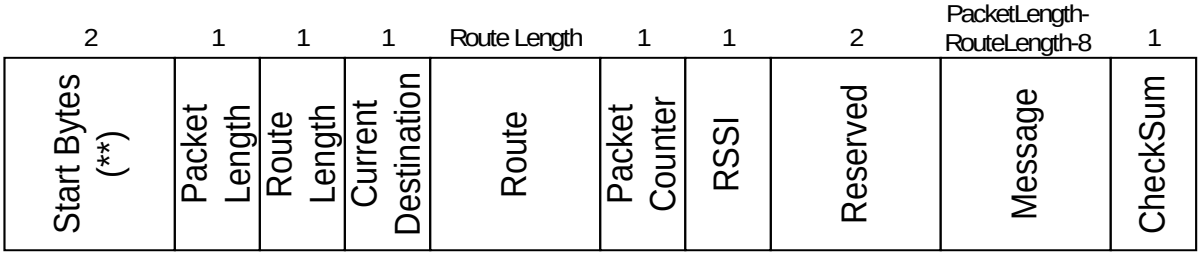
3.3.1 Δομή Πακέτων Δεδομένων

Τα πακέτα δεδομένων που προορίζονται για έναν ελεγκτή Netmaster RS485, έχουν στα δύο πρώτα Bytes την αλφαριθμητική τιμή “αστεράκι” (*). Με αυτό τον τρόπο το λογισμικό του ελεγκτή αναγνωρίζει την αρχή ενός έγκυρου πακέτου και αρχίζει να αποθηκεύει τα εισερχόμενα bytes στην μνήμη τυχαίας προσπέλασης (RAM). Το λογισμικό γνωρίζει το τέλος του πακέτου, ώστε να σταματήσει την αποθήκευση του στη μνήμη, από την τιμή του τρίτου byte, η οποία είναι το συνολικό μήκος του πακέτου σε bytes (Packet Length**).**

Το πακέτο για να φτάσει στον προορισμό του ακολουθεί μια διαδρομή (Route) από κόμβους, η διαδρομή αυτή είναι αποθηκευμένη από το έκτο byte και μετά. Δηλαδή είναι αποθηκευμένες οι διευθύνσεις των κόμβων με τη σειρά που πρέπει να περάσει το πακέτο, όπου η πρώτη είναι του πομπού, η τελευταία του προορισμού και οι ενδιάμεσες των επαναληπτών. Κάθε διεύθυνση αποθηκεύεται σε δύο byte και το συνολικό μήκος των byte του πεδίου της διαδρομής (Route Length**)** είναι η τιμή του τέταρτου byte. Η τιμή του πέμπτου byte είναι ένας δείκτης στον επόμενο έγκυρο προορισμό του πακέτου (Current Destination), δηλαδή σε ποια θέση στο πεδίο της διαδρομής βρίσκεται η διεύθυνση του επόμενου προορισμού. Κάθε δέκτης αυξάνει τον δείκτη αυτό μέχρι να φτάσει στην τελευταία διεύθυνση, όπου θα είναι ίσος με το μήκος σε byte του πεδίου της διαδρομής.

Κάθε φορά που ένας κόμβος στέλνει ένα πακέτο αυξάνει κατά μια μονάδα την τιμή του καταμετρητή πακέτου (Packet Counter), ο οποίος είναι αποθηκευμένος στο byte που βρίσκεται αμέσως μετά το τέλος του πεδίου της διαδρομής. Στο ακριβώς επόμενο byte βρίσκεται η ενδεικτική τιμή της ισχύς του σήματος κατά την λήψη του πακέτου (RSSI), ενώ τα δύο επόμενα byte είναι εφεδρικά για μελλοντική χρήση (Reserved). Στο τελευταίο byte του πακέτου είναι αποθηκευμένο το άθροισμα ελέγχου (Check Sum), το οποίο προκύπτει από το άθροισμα όλων των byte με την πράξη XOR και χρησιμοποιείτε από το λογισμικό για να ελέγξει αν το μήνυμα έφτασε χωρίς λάθη στον δέκτη.

Η δομή που περιγράφεται παραπάνω φαίνεται στην Εικόνα 3.7 και είναι κοινή και για τα πακέτα που λαμβάνει ο Netmaster και γι’ αυτά που στέλνει πίσω ως απάντηση. Το κύριο μήνυμα (Message), του οποίου η δομή θα αναλυθεί παρακάτω, είναι αυτό που διαφέρει στα ληφθέντα και στα αποστέλλοντα πακέτα και περιέχει τις εντολές και τις πληροφορίες αντίστοιχα. Το μήκος του μηνύματος αυτού είναι ίσο με PacketLength-RouteLength-8 και βρίσκεται αποθηκευμένο από το byte με αριθμό 9+Route Length και μετά.



Εικόνα 3.7 Δομή Πακέτου Δεδομένων Netmaster RS485

3.3.2 Δομή Εισερχόμενου Μηνύματος

Το μήνυμα που περιέχεται στα πακέτα που λαμβάνει ο Netmaster είναι μια εντολή ελέγχου, η οποία έχει σταλεί από το λογισμικό διασύνδεσης (HMI) που “τρέχει” στην Κεντρική Μονάδα Επεξεργασίας (MTU). Το πρώτο byte, του μηνύματος αυτού, είναι ένας αριθμός εντολής (Command), ο οποίος καθορίζει την λειτουργία (Operation) που ζητάει το HMI να εκτελέσει ο προγραμματιζόμενος ελεγκτής Netmaster. Σε αυτήν την έκδοση του Netmaster οι πιθανές εντολές είναι εννέα και έτσι ο αριθμός παίρνει τιμές από μηδέν έως οκτώ. Από το δεύτερο byte και μετά είναι αποθηκευμένοι οι όροι της εντολής (Operands), δηλαδή οι λεπτομέρειες για της εκτέλεση της λειτουργίας. Η δομή των εισερχόμενων πακέτων φαίνεται στον Πίνακα 3.1.

Index	Value
0	*
1	*
2	Packet Length
3	Route Length
4	Current Destination
5	Route0H
6	Route0L
7	Route1H
8	Route1L
5+RouteLength	Packet Counter
6+ RouteLength	Rssi
7+ RouteLength	Reserved
8+ RouteLength	Reserved
9+ RouteLength	Command
10+ RouteLength	Operands
	…
PacketLength-1	Check Sum

Πίνακας 3.1 Δομή Εισερχόμενων Πακέτων

Οι εντολές με τους αντίστοιχους αριθμούς τους και τις Operands αναλύονται παρακάτω και παρουσιάζονται στον Πίνακας 3.2.

- Με Αριθμός Εντολής (Command) ίσο με μηδέν (0), εκτελείται από τον Netmaster η λειτουργία καθαρισμού (Clear) της ψηφιακής εξόδου που αναφέρεται στο πεδίο Operands, και σβήνει το αντίστοιχο LED.
- Με τιμή ένα (1) στο πεδίο Command ανάβει το LED της εξόδου που αναφέρει το πεδίο Operands.
- Με την τιμή δύο (2) δεν εκτελείται καμία λειτουργία από τον Netmaster.
- Η τιμή τρία (3) στο Command δίνει την εντολή στον Netmaster να αλλάξει το αποθηκευμένο κάτω επιτρεπτό όριο της στάθμης του υγρού στη δεξαμενή που παρακολουθεί ο ελεγκτής. Η νέα τιμή είναι αποθηκευμένη στο πεδίο Operands.
- Ενώ με την τιμή τέσσερα (4) αλλάζει παρόμοια το πάνω επιτρεπτό όριο.
- Με τιμή πέντε (5) στο πεδίο Command ρυθμίζεται το ρολόι του ελεγκτή Netmaster με την ώρα και ημερομηνία που του στέλνεται στο πεδίο Operands.
- Με την τιμή έξι (6) αλλάζει το η διεύθυνση (Identification Number – ID) του συγκεκριμένου ελεγκτή με αυτή στο πεδίο Operands.
- Όταν το πεδίο Command έχει την τιμή επτά (7) ο ελεγκτής αναγνωρίζει ότι οι πληροφορίες, που είναι αποθηκευμένες, στο πεδίο Operands είναι δεδομένα που προορίζονται για το modem (Modulation-Demodulation) που χρησιμοποιεί ο Netmaster.
- Τέλος με την τιμή οκτώ (8) αλλάζει ο αποθηκευμένος αριθμός του Τοπικού Αυτοματισμού (Local Automation) παίρνοντας τιμές ένα (1) και (2).

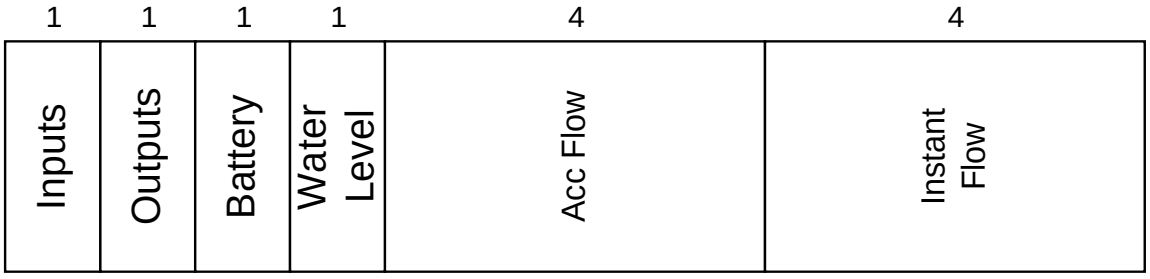
Command	Operation	Operands (Bytes)
0	Clear	Data Mask
1	Set	Data Mask
2	Don Change	-
3	Set Low Limit	Low Limit High - Low Limit Low
4	Set High Limit	High Limit High - High Limit Low
5	SetRTC	Hour,Minute,Second, Day,Month,Year
6	Change ID	ID High – ID Low
7	Send to modem	Data
8	Local Automation	0 or 1

Πίνακας 3.2 Εντολές Ελέγχου του Netmaster RS485 (2nd Series)

3.3.3 Δομή Εξερχόμενου Μηνύματος Απάντησης

Όταν ο ελεγκτής Netmaster λάβει ένα πακέτο ελέγχει αν είναι έγκυρο και στη συνέχεια αν προορίζεται γι’ αυτόν αλλιώς το προωθεί στον επόμενο, κατά το πεδίο διαδρομής, κόμβο. Στην περίπτωση που το πακέτο προορίζεται για τον συγκεκριμένο ελεγκτή το αποκωδικοποιεί και εκτελεί αν περιέχετε, σε αυτό, κάποια εντολή. Στη συνέχεια συντάσσει ένα μήνυμα απάντησης και με αυτό το μήνυμα δημιουργεί ένα πακέτο, το οποίο έχει τη δομή που αναφέρεται στην αρχή του κεφαλαίου, και το στέλνει στον αποστολέα του αρχικού μηνύματος.

Το μήνυμα απάντησης περιέχει την κατάσταση των εισόδων και εξόδων του Netmaster και τις πληροφορίες που έχει συλλέξει αυτός από τους αισθητήρες. Στο πρώτο και στο δεύτερο byte του μηνύματος είναι αποθηκευμένοι δυαδικοί αριθμοί που έχουν προκύψει από τις καταστάσεις των εισόδων και εξόδων, αντίστοιχα. Στο τρίτο byte βρίσκεται το επίπεδο ενέργειας της μπαταρίας και στο τέταρτο, για την συγκεκριμένη περίπτωση, η στάθμη του νερού στην δεξαμενή. Τέλος στα επόμενα τέσσερα byte είναι αποθηκευμένη η συνολική μέτρηση της ροής (Συσσωρευμένη Ροή – Accumulate Flow) που παρακολουθείτε από τον ελεγκτή και στα τελευταία τέσσερα η ροή την στιγμή που έγινε η μέτρηση (Στιγμιαία Ροή – Instant Flow). Η δομή του μηνύματος απάντησης φαίνεται στην Εικόνα 3.8 και η συνολική δομή του πακέτου απάντησης στον Πίνακας 3.3.



Εικόνα 3.8 Δομή Μηνύματος Απάντησης

Index	Value
0	*
1	*
2	PacketLength
3	Route Length
4	Current Destination
5	Route0H
6	Route0L
7	Route1H
8	Route1L
5+RouteLength	Packet Counter
6+ RouteLength	Rssi
7+ RouteLength	Reserved
8+ RouteLength	Reserved
9+ RouteLength	Inputs
10+ RouteLength	Outputs
11+ RouteLength	Battery
12+ RouteLength	Water Level
13+ RouteLength	Acc Flow3
14+ RouteLength	Acc Flow2
15+ RouteLength	Acc Flow1
16+ RouteLength	Acc Flow0
17+ RouteLength	Instant Flow3
18+ RouteLength	Instant Flow2
19+ RouteLength	Instant Flow1
20+ RouteLength	Instant Flow0
PacketLength-1	Check Sum

Πίνακας 3.3 Δομή Πακέτου Απάντησης

4 Visual Basic 6 και VSPE της “Eterlogic”

Η υλοποίηση της εφαρμογής εικονικού κόμβου ασύρματης τηλεμετρίας πραγματοποιήθηκε στην γλώσσα προγραμματισμού Visual C με την βοήθεια της εφαρμογής Microsoft Visual Basic 6.0 και ορισμένων εργαλείων της. Επίσης χρησιμοποιήθηκε ο Εξομοιωτής Εικονικών Σειριακών Θυρών (Virtual Serial Ports Emulator – VSPE) για γίνει εξομοίωση της επικοινωνίας του εικονικού κόμβου με ένα λογισμικό διασύνδεσης SCADA, δηλαδή μια εφαρμογή ελέγχου απομακρυσμένων συσκευών τηλεμετρίας.

4.1 Γλώσσες Προγραμματισμού

Κάθε υπολογιστής μπορεί να κατανοήσει και να εκτελέσει εντολές οι οποίες είναι γραμμένες σε γλώσσα μηχανής και αποτελούνται από μια ακολουθία 0 και 1 σταθερού ή μεταβλητού μήκους. Κάθε τέτοια ακολουθία είναι μια εντολή προς την κεντρική μονάδα επεξεργασίας να εκτελέσει μια στοιχειώδη λειτουργία, όπως για παράδειγμα πρόσθεση, αφαίρεση και αποθήκευση στην κεντρική μνήμη. Ο προγραμματισμός σε γλώσσα μηχανής, ο οποίος είναι ο πρώτος τρόπος προγραμματισμού των ηλεκτρονικών υπολογιστών, ήταν επίπονος για τον προγραμματιστή της εποχής εκείνης. Για τον λόγο αυτό δημιουργήθηκαν οι γλώσσες προγραμματισμού, οι οποίες ουσιαστικά χρησιμοποιούνται για την επίλυση εξειδικευμένων προβλημάτων.

Οι γλώσσες προγραμματισμού είναι προγράμματα, τα οποία έχουν ένα ιδιαίτερο χαρακτηριστικό, το πλαίσιο εγγραφής του κώδικα, ένα χώρο δηλαδή όπου ο χρήστης συντάσσει τις εντολές που του παρέχει η γλώσσα προγραμματισμού και κατασκευάζει τον κώδικά του. Κατόπιν, η γλώσσα προγραμματισμού μετατρέπει τον κώδικα σε μορφή ώστε να είναι κατανοητή στο λειτουργικό σύστημα του υπολογιστή και κατασκευάζει το εκτελέσιμο αρχείο της εφαρμογής.

Στη συνέχεια ένα μεταφραστικό πρόγραμμα μεταφράζει το πρόγραμμα του χρήστη, στο αντίστοιχο πρόγραμμα σε γλώσσα μηχανής. Το προς μετάφραση πρόγραμμα ονομάζεται πηγαίο (source) πρόγραμμα, και το μεταφρασμένο ονομάζεται αντικείμενο (object) πρόγραμμα. Τα μεταφραστικά προγράμματα διακρίνονται σε τρείς κατηγορίες. Η πρώτη κατηγορία είναι οι μεταφραστές γλωσσών χαμηλού επιπέδου ή συμβολομεταφραστές (Assemblers), η δεύτερη είναι οι μεταφραστές γλωσσών υψηλού επιπέδου (Compilers) και η τρίτη οι διερμηνείς γλωσσών υψηλού επιπέδου με ταυτόχρονη εκτέλεση του προγράμματος (Interpreters).

Οι Compilers χρησιμοποιούνται περισσότερο από τους τρεις παραπάνω μεταφραστές και είναι ενσωματωμένα ή εξωτερικά προγράμματα που παρέχουν οι γλώσσες προγραμματισμού. Μέσω αυτών γίνεται η μετάφραση του κώδικα από την μορφή που τον συντάσσει ο χρήστης προς την μορφή που τον κατανοεί το λειτουργικό σύστημα. Ο Compiler χρησιμεύει σε τρία πράγματα. Όταν γίνει εκκίνηση του Compiler, αυτός αρχικά ελέγχει τον κώδικα για πιθανά συντακτικά λάθη, στη συνέχεια τον μεταφράζει σε γλώσσα μηχανής και τέλος κατασκευάζει το εκτελέσιμο αρχείο της εφαρμογής.

Από τις πρώτες γλώσσες προγραμματισμού που χρησιμοποιήθηκαν είναι η GWbasic, η Qbasic και η Pascal. Οι γλώσσες προγραμματισμού αυτές είχαν περιορισμένες δυνατότητες καθώς οι προγραμματιστές μπορούσαν να αναπτύξουν πολύ απλές εφαρμογές με λειτουργίες όπως η δημιουργία και διαχείριση αρχείων, η εκτύπωση και η πραγματοποίηση αριθμητικών πράξεων. Οι εφαρμογές που δημιουργούνταν “έτρεχαν” σε λογισμικό MS-DOS και δεν ήταν τόσο φιλικές προς το χρήστη.

Μετά την κυκλοφορία των Windows δημιουργήθηκαν νέες γλώσσες προγραμματισμού με τις οποίες υπήρχε η δυνατότητα να κατασκευαστούν εφαρμογές με γραφικό περιβάλλον εργασίας. Από τις ποιο δημοφιλής νέες γλώσσες είναι η Java, η Visual C και η Visual Basic, οι οποίες σε αντίθεση με τις παλαιότερες είναι πολύ φιλικές προς τον προγραμματιστή και έχουν πλήθος δυνατοτήτων. Επίσης έχουν εξελιγμένους Compilers για εύκολη διάγνωση σφαλμάτων στην δομή του κώδικα και συνοδευτικά υποπρογράμματα για την κατασκευή και διανομή των προγραμμάτων που κατασκευάζουν. Μπορούν να διαχειριστούν εύκολα μεγάλα αρχεία και να επεξεργαστούν μεγάλο όγκο δεδομένων απαιτώντας από τον προγραμματιστή λίγες γραμμές κώδικα.

4.2 Visual Basic

Η Microsoft έχει δημιουργήσει μια πολύ εύχρηστη γλώσσα προγραμματισμού με την οποία αναπτύσσονται εφαρμογές με γραφικό περιβάλλον εργασίας οι οποίες "τρέχουν" στο περιβάλλον των Windows. Η γλώσσα αυτή είναι η Visual Basic και στην πτυχιακή αυτή χρησιμοποιήθηκε η έκδοση 6.0. Η Visual Basic είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης στο οποίο ο χρήστης μπορεί να αναπτύξει, να εκτελέσει, να δοκιμάσει και να διορθώσει τις εφαρμογές του. Συνήθως οι εφαρμογές που φτιάχνονται με την Visual Basic είναι τυπικές εκτελέσιμες εφαρμογές (Standard Executable - EXE). Βεβαίως με την Visual Basic μπορούν να κατασκευαστούν και προχωρημένοι τύποι έργων όπως ActiveX Exe, DHTML ή IIS εφαρμογές.

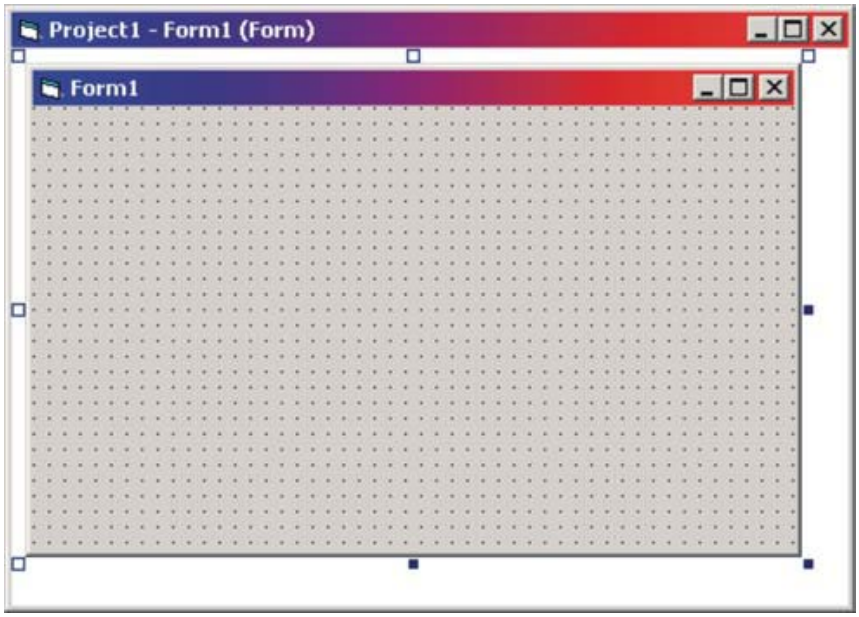
4.2.1 Περιβάλλον εργασίας της Visual Basic

Όπως και σε πολλές εφαρμογές, έτσι και στην Visual Basic στο πάνω μέρος υπάρχει η γραμμή μενού και η γραμμή εργαλείων. Στην αριστερή πλευρά στην εφαρμογή αυτή υπάρχει η εργαλειοθήκη. Στην δεξιά πλευρά και στο πάνω μέρος υπάρχει η εξερεύνηση των σχεδίων (Project Explorer) που δείχνει δηλαδή τις φόρμες του προγράμματος που κατασκευάζεται, ενώ στο κάτω μέρος φαίνεται η διάταξη της φόρμας και το παράθυρο ιδιοτήτων.

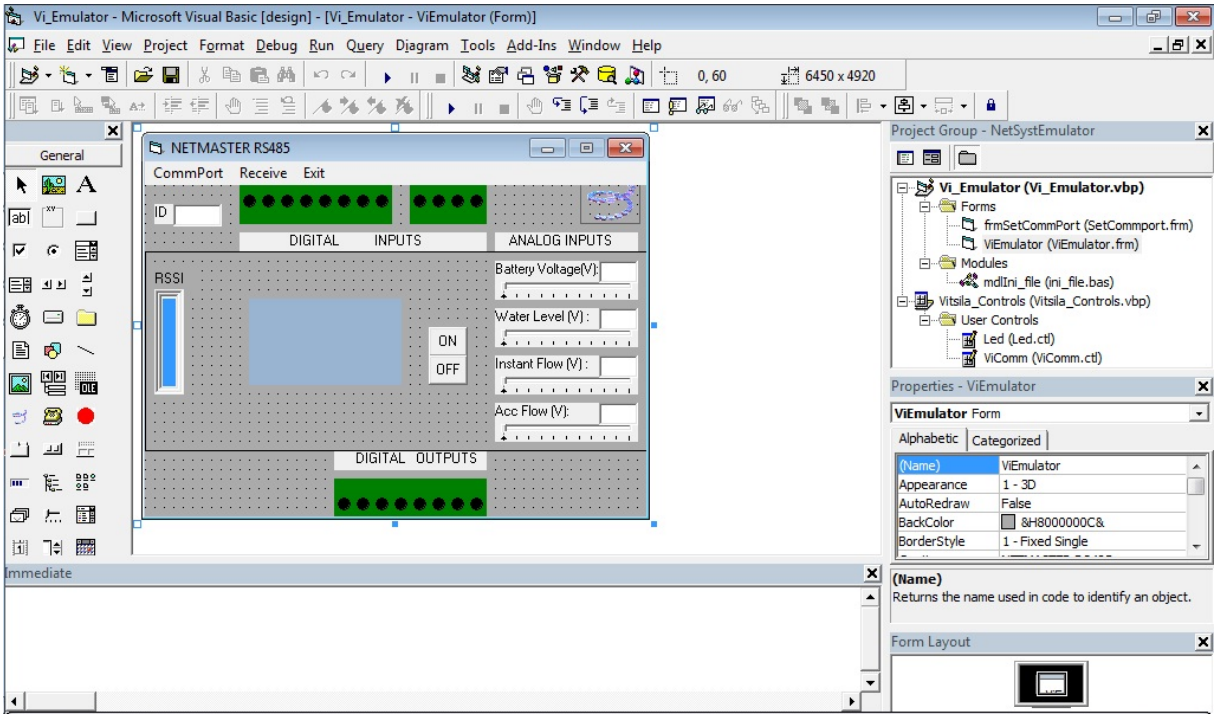
Η εργαλειοθήκη περιέχει τα εικονίδια των στοιχείων που μπορούν να τοποθετηθούν σε μια φόρμα για να δημιουργηθεί η διεπαφή χρήστη της εφαρμογής. Προεπιλεγμένα, η εργαλειοθήκη περιέχει τα βασικά στοιχεία και ονομάζεται "General". Εκτός από αυτή την εργαλειοθήκη, ο χρήστης μπορεί να δημιουργήσει δικές του προσαρμοσμένες εργαλειοθήκες με στοιχεία της επιλογής του.

Το παράθυρο των ιδιοτήτων περιέχει τις ρυθμίσεις ιδιοτήτων για το επιλεγμένο αντικείμενο. Οι ιδιότητες είναι τα χαρακτηριστικά ενός αντικειμένου όπως το μέγεθος, η λεζάντα και το χρώμα. Ο χρήστης μπορεί να ρυθμίσει την εμφάνιση των στοιχείων στη φόρμα με λειτουργίες κατάδειξης και επιλογές. Για παράδειγμα μπορεί να θέσει μια ακολουθία χαρακτήρων που εμφανίζεται σε ένα κουμπί εντολής θέτοντας την ιδιότητα "Caption" στο παράθυρο ιδιοτήτων και πληκτρολογώντας μια νέα τιμή. Για να αλλάξει το χρώμα μιας φόρμας μπορεί να δώσει μια νέα τιμή στην ιδιότητα "Back Color".

Στη Εικόνα 4.1 φαίνεται ο σχεδιαστής φορμών, ο οποίος είναι το κύριο παράθυρο και βρίσκεται στο μέσον της οθόνης. Σε αυτό ο χρήστης σχεδιάζει και επεξεργάζεται τη διεπαφή χρήστη της εφαρμογής. Στο ίδιο παράθυρο όπως φαίνεται και στην Εικόνα 4.2, εμφανίζεται και ένας επεξεργαστής κειμένου στον οποίο γίνεται η εισαγωγή και επεξεργασία του κώδικα της εφαρμογής.



Εικόνα 4.1 Σχεδιαστής Φορμών της Visual Basic



Εικόνα 4.2 Το περιβάλλον εργασίας και ανάπτυξης εφαρμογών της Visual Basic

Ένας από τους λόγους που έχει γίνει δημοφιλής η γλώσσα προγραμματισμού Visual Basic είναι το παράθυρο Immediate, το οποίο φαίνεται στην Εικόνα 4.2. Το παράθυρο αυτό είναι ένα βοήθημα διόρθωσης σφαλμάτων και η προεπιλεγμένη θέση του στην εφαρμογή είναι στο κάτω μέρος. Ο χρήστης ενώ εκτελείται μια εφαρμογή μπορεί να σταματήσει και να χρησιμοποιήσει το παράθυρο Immediate. Μέσω αυτού, μπορεί να εξετάσει ή να αλλάξει τις τιμές των μεταβλητών της εφαρμογής και να εκτελέσει εντολές της Visual Basic σε κατάσταση λειτουργίας Immediate.

Επίσης επιτρέπει στο χρήστη να βηματίζει μέσα στον κώδικα της εφαρμογής ενώ εκείνος εκτελείται. Έτσι του δίνει την δυνατότητα να αλλάζει τις τρέχουσες τιμές των μεταβλητών ή ακόμη και να εισάγει προτάσεις στον κώδικα και μετά να επιλέγει την συνέχεια εκτέλεσης της εφαρμογής. Το παράθυρο μπορεί να χρησιμοποιηθεί επίσης για να εκτελεσθούν άμεσα προτάσεις σαν να ήταν το παράθυρο μιας προχωρημένης αριθμομηχανής. Για παράδειγμα, αν σε κατάσταση Immediate ο χρήστης εισάγει την πρόταση “Print 15/4” και πατήσει Enter, στο παράθυρο Immediate θα εμφανιστεί το αποτέλεσμα της διαίρεσης, δηλαδή “3.75”.

4.2.2 Στοιχεία διεπαφής χρήστη

Όταν εκτελείται μια εφαρμογή, εμφανίζεται στο παράθυρο της η διεπαφή του υλοποιημένου προγράμματος. Μια διεπαφή μπορεί να αποτελείται από διάφορα στοιχεία με τα οποία ο χρήστης μπορεί να αλληλεπιδράσει και να ελέγξει την εφαρμογή. Το πρώτο στοιχείο της διεπαφής χρήστη είναι η φόρμα. Αυτό είναι το παράθυρο που εμφανίζεται κατά τον χρόνο εκτέλεσης και δρα ως υποδοχέας για όλα τα άλλα στοιχεία της διεπαφής. Τα στοιχεία στην διεπαφή χρήστη είναι κοινά σε όλες τις εφαρμογές των Windows και εμφανίζονται ως εικονίδια στην εργαλειοθήκη.

Εικόνα 4.3 Η βασική εργαλειοθήκη αντικειμένων της Visual Basic

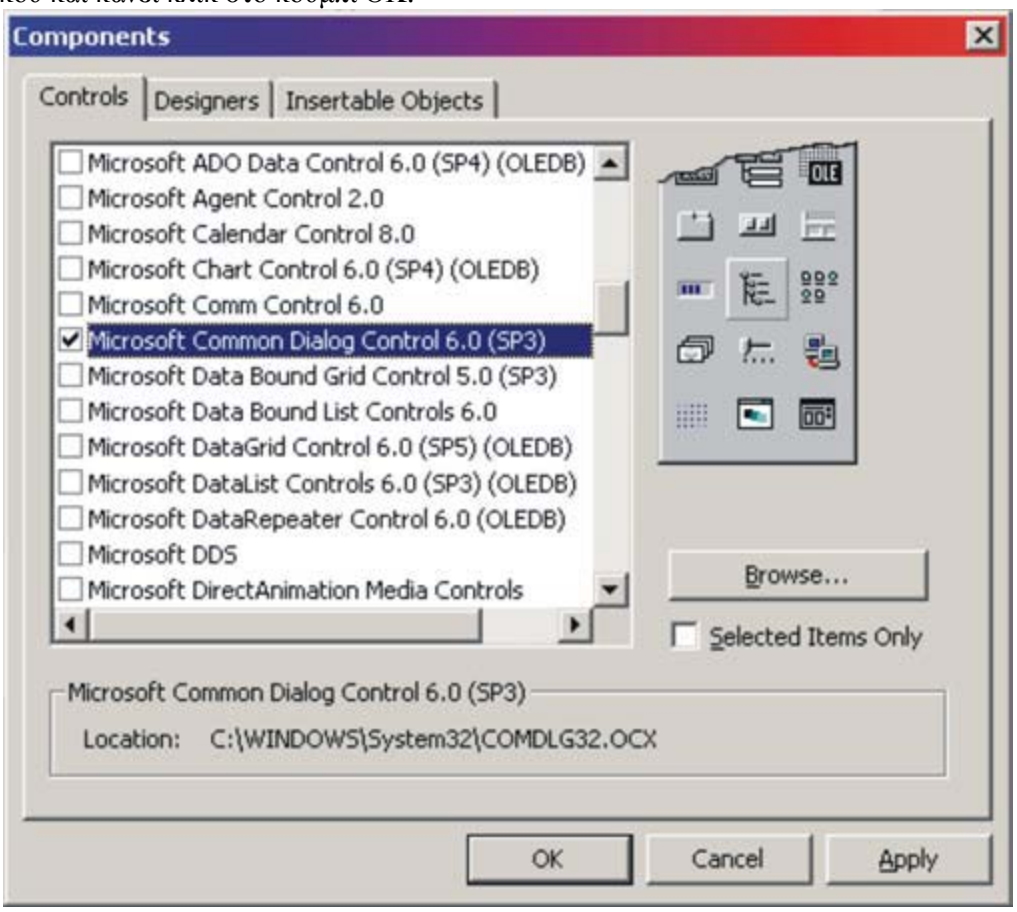
Τα στοιχεία της διεπαφής χρήστη καλούνται αντικείμενα, επειδή μπορεί να χειρίζονται σαν αντικείμενα. Τα βασικά αντικείμενα που περιλαμβάνει η προεπιλεγμένη εργαλειοθήκη της Visual Basic φαίνονται στην Εικόνα 4.3 και είναι τα εξής:

- Πλαίσιο Εικόνας (PictureBox): Το αντικείμενο αυτό χρησιμοποιείται για την εμφάνιση εικόνων και υποστηρίζει αρκετές μεθόδους για παραγωγή σχεδίων.
- Ετικέτα (Label): Το αντικείμενο αυτό εμφανίζει κείμενο σε μια φόρμα, την οποία ο χρήστης δεν μπορεί να επεξεργαστεί. Οι ετικέτες συνήθως δηλώνουν άλλα στοιχεία και μπορούν να είναι διαφανείς έτσι ώστε το κείμενο να εμφανίζεται σαν να είναι τοποθετημένο κατευθείαν επάνω στη φόρμα. Το κείμενο της ετικέτας του αντικείμενου αντιστοιχεί με την ιδιότητα Caption.
- Πλαίσιο Κειμένου (TextBox): Εμφανίζει κείμενο το οποίο μπορεί να επεξεργαστεί ο χρήστης. Είναι δηλαδή ένας μικροεπεξεργαστής κειμένου και η σημαντικότερη ιδιότητά του είναι η "Text", η οποία μπορεί να θέσει το κείμενο στο αντικείμενο ή μπορεί να διαβάσει το κείμενο που εισάγει ο χρήστης.
- Πλαίσιο (Frame): Το αντικείμενο αυτό χρησιμοποιείται για σχεδίαση πλαισίων στην φόρμα και για ομαδοποίηση άλλων στοιχείων.
- Κουμπί Εντολής (Command Button): Είναι το συνηθέστερο αντικείμενο διασύνδεσης των Windows. Ένα αντικείμενο "Command Button" παριστά μια ενέργεια που γίνεται όταν ο χρήστης πατήσει το κουμπί.
- Πλαίσιο Ελέγχου (Check Box): Παρουσιάζει μια ή περισσότερες επιλογές από τις οποίες μπορεί να επιλέξει ο χρήστης. Η κύρια ιδιότητά του είναι η Value, η οποία έχει τιμές 1 ή 0 αν αντίστοιχα το πλαίσιο ελέγχου είναι επιλεγμένο ή όχι. Το CheckBox είναι σαν διακόπτης και κάθε φορά που ο χρήστης κάνει κλικ επάνω του, αλλάζει κατάσταση από επιλεγμένο σε λευκό και πάλι πίσω.
- Κουμπί Επιλογής (Option Button): Τα κουμπιά επιλογής εμφανίζονται σε ομάδες, και ο χρήστης μπορεί να επιλέξει μόνο ένα από αυτά. Η κύρια ιδιότητά του είναι η Checked, και είναι True αν το αντικείμενο είναι επιλεγμένο και False αν δεν είναι. Το Option button είναι σαν διακόπτης και κάθε φορά που ο χρήστης κάνει κλικ επάνω του, αλλάζει κατάσταση από επιλεγμένο σε λευκό και πάλι πίσω.
- Σύνθετο Πλαίσιο (ComboBox): Το αντικείμενο αυτό είναι παρόμοιο με το αντικείμενο ListBox αλλά περιέχει ένα πεδίο επεξεργασίας κειμένου. Ο χρήστης μπορεί είτε να επιλέξει ένα στοιχείο από την λίστα ή να εισάγει μια νέα ακολουθία χαρακτήρων στο πεδίο επεξεργασίας. Το στοιχείο που επιλέγεται από την λίστα ή εισάγεται στο πεδίο επεξεργασίας, δίνεται από την ιδιότητα Text του αντικειμένου.
- Πλαίσιο Λίστας (ListBox): Το αντικείμενο αυτό περιέχει μια λίστα επιλογών από τις οποίες ο χρήστης μπορεί να επιλέξει μια ή περισσότερες. Σε αντίθεση με μια ομάδα πλαισίων ελέγχου ή κουμπιών επιλογής, το αντικείμενο ListBox μπορεί να περιέχει πολλές γραμμές και ο χρήστης μπορεί να κυλήσει στη λίστα για να εντοπίσει ένα στοιχείο. Το επιλεγμένο στοιχείο σε ένα αντικείμενο ListBox δίνεται από την ιδιότητα Text.
- Οριζόντια και Κάθετη Γραμμή Κύλισης (Horizontal και Vertical ScrollBar): Οι οριζόντιες και κατακόρυφες γραμμές κύλισης επιτρέπουν στο χρήστη να καθορίσει ένα μέγεθος, όπως για παράδειγμα την τρέχουσα θέση σε ένα μεγάλο κομμάτι κειμένου, κυλίοντας το κουμπί του αντικειμένου ανάμεσα στην ελάχιστη και στη μέγιστη τιμή.
- Χρονοδιακόπτης (Timer): Ο χρήστης μπορεί να το χρησιμοποιήσει για να κάνει εργασίες σε τακτά χρονικά διαστήματα. Η κύρια ιδιότητα του αντικειμένου αυτού είναι η IntervaI, που καθορίζει πόσο συχνά θα ειδοποιείται η εφαρμογή. Αν η ιδιότητα Interval τεθεί σε 10000, το αντικείμενο Timer εκδίδει ένα συμβάν Timer κάθε 10 δευτερόλεπτα.
- DriveListBox: Εμφανίζει τις μονάδες δίσκων του συστήματος σε μια αναπτυσσόμενη λίστα από την οποία ο χρήστης μπορεί να επιλέξει.
- DirectoryListBox: Εμφανίζει μια λίστα όλων των φακέλων στην τρέχουσα μονάδα δίσκου και επιτρέπει στο χρήστη να κινηθεί προς τα επάνω ή προς τα κάτω στην ιεραρχία των φακέλων.
- FileListBox: Εμφανίζει μια λίστα όλων των αρχείων στον τρέχοντα φάκελο.

- Σχήμα (Shape): Το αντικείμενο Shape χρησιμοποιείται για να σχεδιάσει γραφικά στοιχεία, όπως πλαίσια και κύκλους, στην επιφάνεια μιας φόρμας.
- Γραμμή (Line): Παρόμοιο με το αντικείμενο Shape, χρησιμοποιείται για σχεδίαση γραμμών σε μια φόρμα.
- Εικόνα (Image): Το αντικείμενο αυτό είναι παρόμοιο με το Picturebox στο ότι μπορεί να εμφανίσει εικόνες, αλλά υποστηρίζει μόνο μερικά χαρακτηριστικά του αντικείμενου Picturebox.
- Δεδομένα (Data): Το αντικείμενο αυτό παρέχει προσπέλαση με κατάδειξη και κλικ σε δεδομένα αποθηκευμένα σε βάσεις δεδομένων.
- OLE: Το αντικείμενο αυτό είναι ένα παράθυρο που μπορεί να τοποθετηθεί σε μια φόρμα για να δέχεται έγγραφα από άλλες εφαρμογές, όπως από το Microsoft Word ή το Microsoft Excel. Μέσω αυτού του αντικειμένου ο χρήστης μπορεί να προσπελάσει τις λειτουργίες άλλων εφαρμογών, αρκεί να υποστηρίζουν OLE.

Οι εφαρμογές των Windows χρησιμοποιούν και άλλα αντικείμενα που δεν εμφανίζονται προεπιλεγμένα στην εργαλειοθήκη της Visual Basic. Δύο αντικείμενα που χρησιμοποιούνται συχνά είναι το είναι τα "TreeView" και "Rich-Textbox". Το πρώτο εμφανίζει ιεραρχικές λίστες στοιχείων όπως αυτήν που χρησιμοποιεί ο Windows Explorer για να παρουσιάζει την δεντρική δομή ενός σκληρού δίσκου. Το δεύτερο είναι παρόμοιο με αυτό που χρησιμοποιεί το Microsoft Word για να επεξεργάζεται κείμενα.

Για να μπορέσει ο προγραμματιστής να χρησιμοποιήσει άλλα αντικείμενα, εκτός από αυτά που έχει η General εργαλειοθήκη της Visual Basic, θα πρέπει να τα προσθέσει στην εργαλειοθήκη. Για να γίνει αυτό, ο προγραμματιστής θα πρέπει να κάνει δεξί κλικ πάνω στην εργαλειοθήκη και από το μενού συντόμευσης να επιλέξει την επιλογή "Components". Μετά από αυτό, στην οθόνη θα εμφανιστεί το παράθυρο "Components" το οποίο φαίνεται στην Εικόνα 4.4. Για να γίνει εισαγωγή κάποιου νέου στοιχείου στο Project της Visual Basic, ο προγραμματιστής απλά επιλέγει το όνομα του συστατικού και κάνει κλικ στο κουμπί OK.



Εικόνα 4.4 Διαχειριστής Components

4.2.3 Ιδιότητες, Μέθοδοι και Συμβάντα Αντικειμένων

Τα αντικείμενα, όπως κάθε φυσικό αντικείμενο, έχουν ιδιότητες. Ένα αυτοκίνητο για παράδειγμα είναι ένα φυσικό αντικείμενο και μια από τις ιδιότητές του είναι το χρώμα. Κανονικά, οι ιδιότητες τίθενται όταν δημιουργείται ένα αντικείμενο, το χρώμα όμως του αυτοκινήτου μπορεί να αλλαχτεί. Το ίδιο συμβαίνει και με τις περισσότερες ιδιότητες των αντικειμένων, οι οποίες τίθενται όταν δημιουργείται το αντικείμενο, όταν δηλαδή τοποθετείται στη φόρμα. Υπάρχει όμως και η δυνατότητα αλλαγής των ιδιοτήτων της αργότερα, αντιστοιχίζοντας μια νέα τιμή σε αυτή την ιδιότητα. Ο χρήστης μπορεί να αλλάξει μια ιδιότητα κατά το χρόνο σχεδίασης μέσω του παράθυρου Ιδιοτήτων ή κατά το χρόνο εκτέλεσης μέσω του κώδικα.

Η Visual Basic αντιστοιχίζει προεπιλεγμένες ιδιότητες σε κάθε νέο αντικείμενο που τοποθετείται σε μια φόρμα. Η προεπιλεγμένη ιδιότητα Name, για παράδειγμα, είναι το όνομα του αντικειμένου ακολουθούμενο από ένα αριθμό, όπως για παράδειγμα "Command1". Ο χρήστης μπορεί να εξετάζει τις τιμές των ιδιοτήτων ενός νέου αντικειμένου στο παράθυρο Ιδιοτήτων. Ορισμένες ιδιότητες διατίθενται μόνο κατά το χρόνο σχεδίασης και μερικές άλλες διατίθενται μόνο κατά το χρόνο εκτέλεσης. Για παράδειγμα, δεν είναι δυνατό να καθοριστεί ένα στοιχείο σε ένα αντικείμενο Listbox κατά το χρόνο σχεδίασης, επειδή το αντικείμενο είναι κενό, αλλά γεμίζει με προτάσεις της Visual Basic όταν γίνει εκκίνηση του προγράμματος. Είναι επίσης δυνατό να γεμίσει το αντικείμενο κατά τον χρόνο σχεδίασης αντιστοιχίζοντας τιμές στην ιδιότητα List.

Ο πραγματικός χειρισμός όμως των στοιχείων του αντικειμένου γίνεται κατά το χρόνο εκτέλεσης. Έτσι η ιδιότητα Text του αντικειμένου Listbox δεν έχει σημασία κατά το χρόνο σχεδίασης αλλά κατά το χρόνο εκτέλεσης όπου και είναι η σημαντικότερη ιδιότητά του. Η ιδιότητα Multiline του αντικειμένου Textbox, καθορίζει αν το πλαίσιο κειμένου περιέχει πολλαπλές γραμμές κειμένου. Ο χρήστης μπορεί να θέσει αυτή την ιδιότητα μόνο κατά το χρόνο σχεδίασης και δεν μπορεί να αλλάξει κατά το χρόνο εκτέλεσης. Υπάρχουν ιδιότητες που ισχύουν για πολλά αντικείμενα. Οι παρακάτω ιδιότητες είναι ορισμένες από αυτές που ισχύουν για τα περισσότερα αντικείμενα της Visual Basic.

- Name: Η ιδιότητα αυτή θέτει το όνομα ενός αντικειμένου, μέσω του οποίου ο χρήστης μπορεί να προσπελάσει τις ιδιότητες και τις μεθόδους του αντικειμένου.
- BackColor: Αυτή η ιδιότητα θέτει το χρώμα φόντου επί του οποίου εμφανίζονται το κείμενο ή τα γραφικά.
- ForeColor: Η ιδιότητα αυτή θέτει το χρώμα προσκήνιου, χρώμα πένας ή χρώμα κειμένου.
- Captíon: Η ιδιότητα αυτή θέτει το κείμενο που εμφανίζεται σε πολλά αντικείμενα που δε δέχονται είσοδο. Για παράδειγμα το κείμενο σε ένα αντικείμενο Label, τη λεζάντα ενός Command Button, και τις ακολουθίες χαρακτήρων που εμφανίζονται δίπλα σε αντικείμενα CheckBox και OptionButton.
- Width και Height: Οι ιδιότητες αυτές θέτουν τις διαστάσεις του αντικειμένου. Συνήθως, οι διαστάσεις ενός αντικειμένου καθορίζονται με τα οπτικά εργαλεία σχεδιασμού της Visual Basic. Ο χρήστης όμως μπορεί να διαβάσει τις διαστάσεις του αντικειμένου ή να τις θέσει μέσα στον κώδικα του με αυτές τις ιδιότητες.
- Left, Top: Οι ιδιότητες αυτές θέτουν τις συντεταγμένες της επάνω αριστερής γωνίας του αντικειμένου, εκφραζόμενες συνήθως στις μονάδες του υποδοχέα, μιας φόρμας. Η τοποθέτηση ενός αντικειμένου στη φόρμα μπορεί να καθοριστεί με το παράθυρο Διάταξης Φόρμας, αλλά μπορεί να αλλαχτεί και μέσω του κώδικα με αυτές τις δύο ιδιότητες.

Τα αντικείμενα έχουν επίσης μεθόδους, που είναι οι ενέργειες που μπορούν να πραγματοποιήσουν. Για παράδειγμα, οι μέθοδοι σε ένα βίντεο είναι Play, Fast Forward, Rewind, Pause και Record. Όταν πατηθεί ένα από αυτά τα κουμπιά, το βίντεο μπορεί να λειτουργήσει χωρίς καμία άλλη βοήθεια από το χρήστη. Ένα άλλο παράδειγμα είναι ένα αντικείμενο Form όπου ξέρει πώς να καθαριστεί και ο χρήστης αρκεί να καλέσει την μέθοδο Cls για να καθαρίσει μια φόρμα. Μια φόρμα ξέρει επίσης πώς να κρυφτεί καλώντας ο χρήστης την μέθοδος Hide μέσα από τον κώδικά. Δύο από τις πιο συχνά χρησιμοποιούμενες κοινές μεθόδους περιγράφονται παρακάτω:

- Clear: Ορισμένες μέθοδοι είναι απλά ρήματα που λένε στο αντικείμενο ποια ενέργεια να κάνει. Η μέθοδος Clear λέει στο αντικείμενο να απορρίψει τα περιεχόμενά του. Αν

το αντικείμενο είναι ένα ListBox, η μέθοδος Clear καταργεί όλα στοιχεία του. Η μέθοδος Clear μπορεί επίσης να εφαρμοστεί στο αντικείμενο Clipboard, για να καθαρίσει τα περιεχόμενά του.

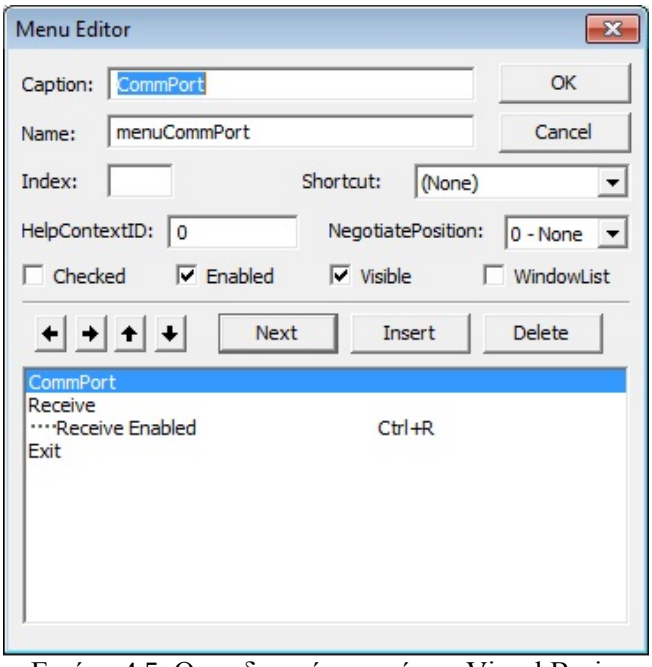
- Move: Όλα τα αντικείμενα που είναι ορατά κατά το χρόνο εκτέλεσης, παρέχουν μια μέθοδο Move που επιτρέπει στο χρήστη να τα μετακινήσει και να τους αλλάξει μέγεθος μέσα από τον κώδικα της εφαρμογής.

Τέλος τα συμβάντα καθορίζουν τις αντιδράσεις του αντικειμένου σε εξωτερικές συνθήκες. Τα συμβάντα αναγνωρίζονται από τα διάφορα αντικείμενα, αλλά ο χειρισμός τους γίνεται μέσα από την εφαρμογή. Ένα κουμπί εντολής θα αναγνωρίσει, ότι έχει γίνει κλικ επάνω του, αλλά δε θα αντιδράσει στο συμβάν παρά μόνο αν ο χρήστης έχει δώσει κάποιο κώδικα. Με άλλα λόγια, ο προγραμματιστής πρέπει να πει στην Visual Basic τι να κάνει, όταν ο χρήστης κάνει κλικ στο συγκεκριμένο κουμπί εντολής. Αφού καθοριστεί μια υπορουτίνα για το συμβάν Click του αντικειμένου, αυτή η υπορουτίνα εκτελείται κάθε φορά που γίνεται κλικ στο αντικείμενο αυτό. Η υπορουτίνα που καθορίζει πώς αντιδρά ένα αντικείμενο σε ένα συμβάν, καλείται χειριστής συμβάντος.

4.2.4 Επεξεργαστής μενού

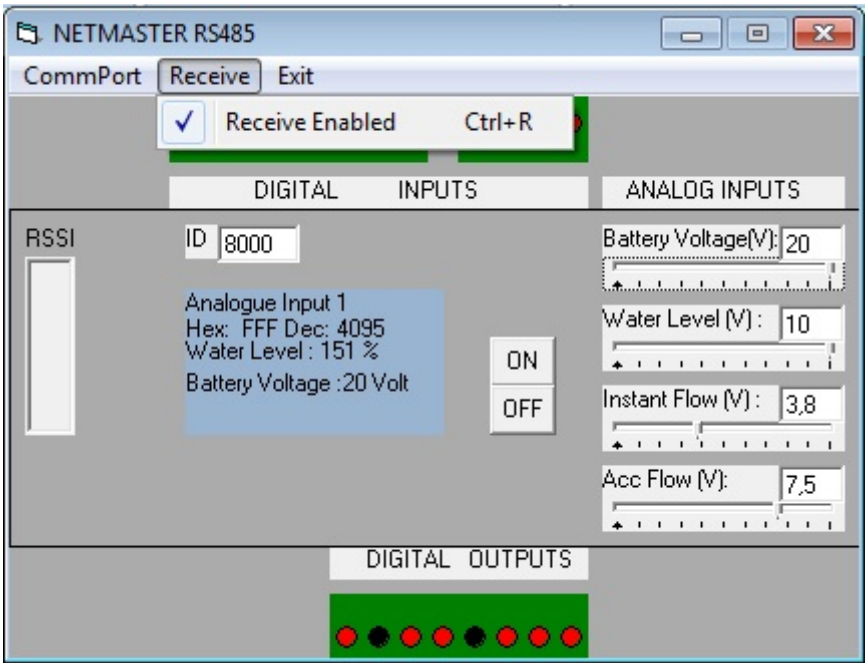
Το μενού είναι ένα από τα σημαντικότερα και χαρακτηριστικότερα στοιχεία της διεπαφής χρήστη των Windows. Ακόμη και στα αρχικά συστήματα που βασίζονταν στη χρήση χαρακτήρων, τα μενού χρησιμοποιούταν για να εμφανίζουν μεθοδικά οργανωμένες επιλογές και να καθοδηγούν το χρήστη μέσα σε μια εφαρμογή. Παρά τις οπτικά πλούσιες διασυνδέσεις των εφαρμογών των Windows και των πολλών εναλλακτικών λύσεων που παρέχουν, τα μενού συνεχίζουν να είναι ο δημοφιλέστερος τρόπος οργάνωσης ενός μεγάλου αριθμού επιλογών. Το μενού είναι ένα βασικό στοιχείο μιας φόρμας και πολλές εφαρμογές επαναλαμβάνουν ορισμένα ή όλα τα μενού τους με τη μορφή εικονιδίων σε μια γραμμή εργαλείων.

Στην Εικόνα 4.5 φαίνεται ο σχεδιαστής της Visual Basic και με τη βοήθεια του, ο χρήστης μπορεί να εισάγει όσα μενού, υπομενού και επιλογές θέλει σε κάθε φόρμα της εφαρμογής του. Κάθε επιλογή έχει δύο βασικές ιδιότητες, την Ιδιότητα Caption και την ιδιότητα Name. Στην ιδιότητα Caption, ο προγραμματιστής συμπληρώνει το όνομα που θέλει να είναι ορατό στο μενού της εφαρμογής του, ενώ η ιδιότητα Name είναι το όνομα της εντολής μενού. Η ιδιότητα αυτή δεν εμφανίζεται στη γραμμή μενού της εφαρμογής στην οθόνη αλλά την χρησιμοποιεί ο προγραμματιστής για να προγραμματίσει την εντολή μενού.



Εικόνα 4.5 Ο σχεδιαστής μενού της Visual Basic

Τα μενού μοιάζουν με δενδρικές δομές, δηλαδή το στοιχείο που βρίσκεται πιο αριστερά στη λίστα είναι το μενού που εμφανίζεται πάνω στη φόρμα. Τα στοιχεία που είναι πιο δεξιά από αυτό είναι οι επιλογές του, ενώ αν η ίδια η επιλογή έχει και άλλα στοιχεία δεξιότερά της, τότε πλέον γίνεται υπομενού και οι επιλογές του είναι τα στοιχεία που έχει δεξιότερά του. Στην Εικόνα 4.6 φαίνεται κατασκευασμένο πάνω στην φόρμα όπως είναι την στιγμή που η εφαρμογή δουλεύει.



Εικόνα 4.6 Παράδειγμα μενού εφαρμογής

4.2.5 Παγίδευση σφαλμάτων

Ένα πρόγραμμα δε φτάνει να είναι ορθό μόνο όσον αφορά τη σύνταξη του κώδικα. Αυτό είναι το εύκολο κομμάτι γιατί τα συντακτικά λάθη τα παγιδεύει ο μεταφραστής και ενημερώνει ανάλογα τον προγραμματιστή. Ο προγραμματιστής όμως από την πλευρά του θα πρέπει να σιγουρευτεί ότι ο κώδικάς του δε θα προσπαθήσει να εκτελέσει κάποια λειτουργία η οποία δεν είναι δυνατή. Για παράδειγμα, δεν είναι δυνατό να γίνει η διαίρεση 100 / 0. Όταν η εφαρμογή φτάσει λοιπόν στο σημείο όπου θα πρέπει να κάνει τη συγκεκριμένη διαίρεση, θα καταρρεύσει.

Μια λύση θα ήταν, οι προγραμματιστές να μη χρησιμοποιούν ποτέ στον κώδικά τους διαιρέσεις με μηδενικό διαιρετέο. Άλλωστε δεν έχει και νόημα αφού η διαίρεση είναι αδύνατη. Αυτό φυσικά δε δίνει καμία λύση στο πρόβλημα γιατί ενώ στο παραπάνω παράδειγμα παρουσιάζεται μια διαίρεση με συγκεκριμένους αριθμούς, πολλές φορές μέσα σε έναν κώδικα παρουσιάζονται διαιρέσεις με μεταβλητές. Οι μεταβλητές μπορεί να προέρχονται από εισαγωγή στοιχείων που κάνει ο χρήστης κατά την εκτέλεση του προγράμματος ή από αποτελέσματα προηγούμενης επεξεργασίας.

Λύσεις στα προβλήματα όπως αυτό που περιγράφτηκε παραπάνω, δίνουν οι εντολές ελέγχου συνθήκης. Η πιο συνηθισμένη από αυτές τις εντολές είναι η "IF". Η εντολή αυτή ελέγχει μια ή περισσότερες συνθήκες και ανάλογα αν είναι αληθείς ή ψευδείς εκτελεί κάποιες εντολές. Για το παράδειγμα που αναφέραμε παραπάνω, ο προγραμματιστής έχει τη δυνατότητα να ελέγξει το διαιρετέο. Αν είναι μηδέν, ο χρήστης θα ενημερωθεί ότι η διαίρεση είναι αδύνατη ενώ αν ο διαιρετέος είναι οποιοσδήποτε άλλος αριθμός, θα γίνει η διαίρεση.

Η Visual Basic έχει ένα δικό της χειριστή για την παγίδευση σφαλμάτων. Η "On Error Goto ErrorLabel" είναι μια πρόταση παγίδευσης σφαλμάτων. Η πρόταση αυτή λέει στην Visual Basic ότι σε περίπτωση που παρουσιαστεί κάποιο σφάλμα, να αλλάξει η ροή εκτέλεσης του προγράμματος και να μεταφερθεί στην υπορουτίνα "ErrorLabel". Σφάλματα όπως το παραπάνω, ονομάζονται "overflow". Είναι το σφάλμα της υπερχείλισης και είναι ένα από τα χειρότερα γιατί δε διορθώνεται

εύκολα. Η υπερχείλιση είναι πάντα το αποτέλεσμα μια σειράς αριθμητικών πράξεων που δεν μπορεί να αναιρεθεί. Δεν μπορεί να γίνει πρόβλεψη του αποτελέσματος της πράξης αν δεν γίνει η πράξη.

Οπότε με το χειριστή σφαλμάτων της Visual Basic, η πράξη θα γίνει. Αντί όμως να "κρεμάσει" η εφαρμογή και να σταματήσει η λειτουργία της, ενεργοποιείται ο χειριστής σφάλματος και εκτελεί τις εντολές που έχει γράψει ο χρήστης για την περίπτωση του σφάλματος.

Η δομή μια υπορουτίνας με παγίδα σφάλματος είναι ως εξής :

```
Sub MySubroutine ()
        On Error Goto ErrorHandler
                K = a / b
                Exit Sub

ErrorHandler:
        Msgbox "Η πράξη δεν είναι δυνατό να γίνει"
End Sub
```

Η πρώτη πρόταση δεν είναι εκτελέσιμη. Δεν προκαλεί καμία ενέργεια στον κώδικα του προγράμματος και απλώς λέει στην Visual Basic ότι αν συμβεί σφάλμα, θα πρέπει να εκτελέσει τις γραμμές που ακολουθούν στην ετικέτα ErrorHandler. Η ErrorHandler είναι μια ακολουθία χαρακτήρων που δηλώνει την αρχή του κώδικα χειρισμού σφαλμάτων.

4.2.6 Οι Μεταβλητές στην Visual Basic

Στην Visual Basic, όπως και σε κάθε άλλη γλώσσα προγραμματισμού, οι μεταβλητές αποθηκεύουν τιμές κατά τη διάρκεια της εκτέλεσης ενός προγράμματος. Οι μεταβλητές είναι κρατήσεις θέσεων στη μνήμη στις οποίες το πρόγραμμα καταχωρεί τιμές ή τις ανακαλεί ανάλογα με τον κώδικά του. Για να δηλωθεί μια μεταβλητή στη Visual Basic χρησιμοποιείται η πρόταση Dim ακολουθούμενη από το όνομα της μεταβλητής και τον τύπο της ως εξής :

```
Dim meters as integer
Dim greetings as string
```

Η πρώτη πρόταση δηλώνει μια αριθμητική μεταβλητή με όνομα meters ακεραίου τύπου. Η δεύτερη πρόταση δηλώνει μια μεταβλητή κειμένου με όνομα greetings. Η Visual Basic έχει ενσωματωμένες συναρτήσεις για να κάνει ελέγχους σε μεταβλητές και να τις συγκρίνει μεταξύ τους. Η συνάρτηση "IsNumeric(meters)" ελέγχει αν στη μεταβλητή meters έχει καταχωρηθεί κάποιος αριθμός και επίσης μπορεί να μετατρέψει τον τύπο μιας μεταβλητής μέσω του κώδικα κατά την διάρκεια εκτέλεσης του προγράμματος. Επίσης έχει συναρτήσεις με τις οποίες απλουστεύονται όλες οι γνωστές αριθμητικές πράξεις. Για παράδειγμα, ο χρήστης για να στρογγυλοποιήσει την τιμή μιας μεταβλητής δε χρειάζεται να κάνει αριθμητικές πράξεις αλλά μπορεί να χρησιμοποιήσει την συνάρτηση Abs().

Οι μεταβλητές στη Visual Basic χαρακτηρίζονται από την εμβέλειά τους και τη διάρκεια της ζωής τους. Η εμβέλειά μιας μεταβλητής είναι η ενότητα της εφαρμογής που μπορεί να τη δει και να τη χειριστεί. Η διάρκεια ζωής χαρακτηρίζει την περίοδο κατά την οποία η μεταβλητή διατηρεί την τιμή της. Όσο αφορά την εμβέλεια μιας μεταβλητής, μπορεί να περιορίζεται σε μια μόνο διαδικασία. Δηλαδή μπορεί να δηλωθεί στο συμβάν Click ενός Command Button. Έτσι, η μεταβλητή είναι ορατή μόνο σε αυτό το σημείο του κώδικα. Αν σε άλλο συμβάν δηλωθεί μεταβλητή με το ίδιο όνομα, τότε αυτές οι δύο δεν έχουν κανένα κοινό στοιχείο και είναι δύο διαφορετικές μεταβλητές.

Όσον αφορά τη διάρκεια ζωής των μεταβλητών, χωρίζονται σε δημόσιες και τοπικές. Μια τοπική μεταβλητή κρατάει την τιμή της όσο η ροή του προγράμματος είναι στη συγκεκριμένη φόρμα όπου έχει δηλωθεί η μεταβλητή. Αν ο χρήστης αλλάξει φόρμα τότε η μεταβλητή παύει να υπάρχει και η δεσμευμένη μνήμη επιστρέφεται στο σύστημα. Σε αντίθεση με τις τοπικές, οι δημόσιες μεταβλητές κρατάνε τις τιμές τους καθ' όλη την εκτέλεση του προγράμματος.

4.2.7 Συστατικά ActiveX και λειτουργικές μονάδες κλάσεων

Αν υπάρχει ένα χαρακτηριστικό της Visual Basic που ελκύει όλων των ειδών τους προγραμματιστές, είναι τα αντικείμενα ActiveX. Η Visual Basic είναι μια απλή, εύκολη στην εκμάθηση και διασκεδαστική γλώσσα για τη δημιουργία ενός αντικείμενου ActiveX. Τα αντικείμενα ActiveX ονομαζόταν προηγουμένως αντικείμενα OLE και είναι κάτι σα μια επέκταση στη γλώσσα Visual Basic. Είναι τα αντικείμενα που παρίστανται στην εργαλειοθήκη της Visual Basic με ένα μικρό εικονίδιο και μπορούν να περιληφθούν σε κάθε φόρμα για να προσθέσουν λειτουργικότητα στις εφαρμογές που τα χρησιμοποιούν.

Ένα συστατικό ActiveX είναι ένας γενικός όρος που περιέχει τρεις τύπους έργων. ActiveX DLL, ActiveX EXE και αντικείμενα ActiveX. Τα συστατικά ActiveX DLL και ActiveX EXE είναι εφαρμογές διακομιστή που εκθέτουν τη λειτουργικότητά τους μέσω μιας διασύνδεσης, που αποτελείται από ιδιότητες μεθόδους και συμβάντα. Οι κύριες διαφορές ανάμεσα στα αντικείμενα ActiveX και στα συστατικά κώδικα βρίσκονται στις διασυνδέσεις τους και στην ενσωμάτωσή τους στη Visual Basic. Τα αντικείμενα ActiveX ενσωματώνονται στο IDE της Visual Basic και έχουν μια ορατή διασύνδεση.

Τα συστατικά κώδικα παρέχουν μια λειτουργικότητα παρόμοια με αυτή των αντικείμενων ActiveX, δεν είναι τόσο ενσωματωμένα με το περιβάλλον ανάπτυξης (π.χ. ένα συστατικό κώδικα δεν μπορεί να τοποθετηθεί σε μια φόρμα όπως μπορεί ένα αντικείμενο) και δεν έχουν ορατή διασύνδεση. Αντί αυτού, τα συστατικά κώδικα είναι κλάσεις που πρέπει να προσπελαύνονται μέσω μιας ιδιότητας που δηλώνεται σαν μεταβλητή αντικειμένου. Μια άλλη κατηγορία συστατικών ActiveX είναι το έγγραφο ActiveX. Τα έγραφα αυτά είναι εφαρμογές που μπορούν να περιέχονται σε υποδοχείς, σαν τον Internet Explorer και το Office Binder.

Η ανάπτυξη αντικείμενων ActiveX μέσω της Visual Basic έχει υλοποιηθεί τόσο καλά, που είναι πολύ οικεία προς το μέσο προγραμματιστή. Όμως πολλοί προγραμματιστές τα καταφέρνουν και χωρίς αυτά. Άλλωστε η Visual Basic διαθέτει στον προγραμματιστή πλήθος αντικειμένων τα οποία καλύπτουν τις περισσότερες εφαρμογές που μπορεί να χρειαστεί να αναπτύξει. Αν όμως χρησιμοποιεί μια συγκεκριμένη εργασία σε πολλές εφαρμογές που αναπτύσσει τότε είναι ένας σημαντικός λόγος για να αναπτύξει προσαρμοσμένα αντικείμενα ActiveX.

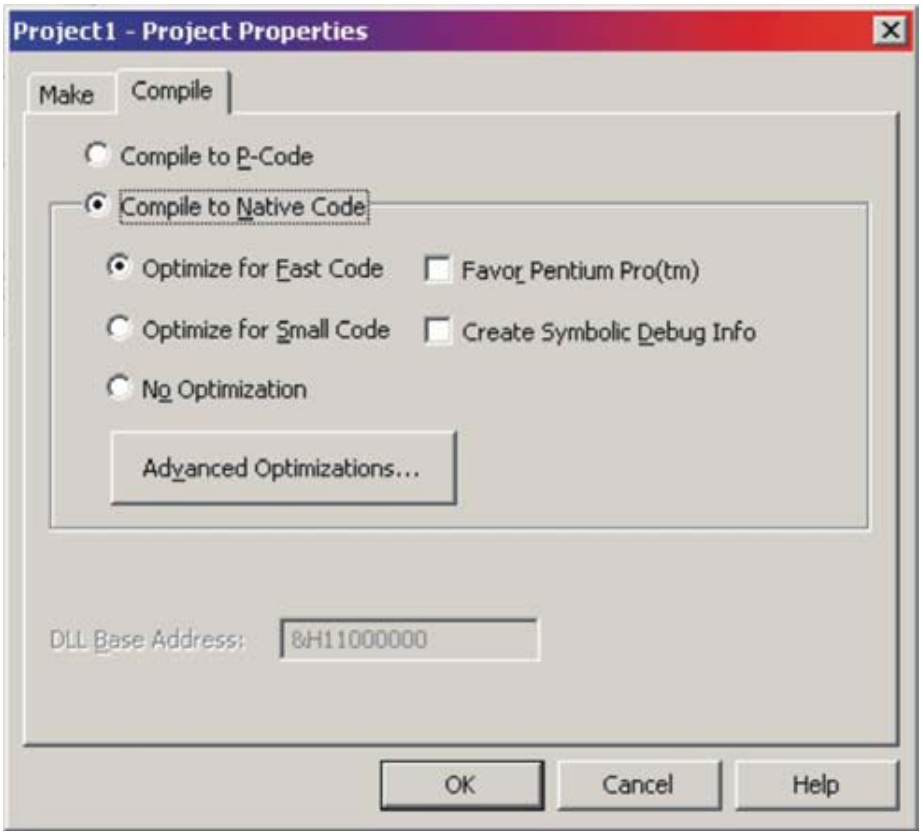
Για παράδειγμα κάποιος προγραμματιστής μπορεί με τη Visual Basic να εργάζεται πάνω σε βάσεις δεδομένων και να αναπτύσσει εφαρμογές διαχείρισης βάσεων δεδομένων. Συχνά θα χρειάζεται μέσα από τον κώδικα της εφαρμογής του να κάνει ταξινομήσεις, ή προσαρμοσμένες εκτυπώσεις. Φυσικά η ταξινόμηση είναι κάτι που εύκολα λύνεται με την χρήση της SQL. Στην εκτύπωση όμως, η Visual Basic έχει σημαντικό πρόβλημα. Πέρα από το αντικείμενο Common Dialogs δεν υπάρχει κάποιο άλλο που να βοηθάει σημαντικά σε προσαρμοσμένες εκτυπώσεις.

Με την έκδοση 6 λοιπόν της Visual Basic, ο προγραμματιστής είναι σε θέση να αναπτύξει ένα αντικείμενο ActiveX, να το προγραμματίσει όπως εκείνος χρειάζεται και κατόπιν να το χρησιμοποιεί στις εφαρμογές του. Το αντικείμενο ActiveX πλέον θα ξέρει πώς να χειριστεί τη συγκεκριμένη εργασία και ο προγραμματιστής γλιτώνει κάθε φορά πολλές γραμμές κώδικα οι οποίες δεν κάνουν τίποτα άλλο παρά δυσανάγνωστο τον κώδικα της εφαρμογής του.

4.2.8 Κατασκευή και διανομή εκτελέσιμων αρχείων

Η Visual Basic είναι σε θέση να εκτελεί τις εφαρμογές οι οποίες αναπτύσσονται μέσω αυτής. Φυσικά το ζητούμενο είναι ένα πρόγραμμα να μπορεί να σταθεί μόνο του μέσα στο περιβάλλον των Windows. Για να γίνει αυτό πρέπει να κατασκευαστεί το εκτελέσιμο αρχείο του προγράμματος. Η εκκίνηση του Compiler της Visual Basic γίνεται από την επιλογή "Make project.exe" από το μενού "File". Ο Compiler της Visual Basic ο οποίος φαίνεται στην Εικόνα 4.7 δίνει τη δυνατότητα στον προγραμματιστή να καθορίσει τον τύπο του εκτελέσιμου αρχείου.

Η μεταγλώττιση του αρχείου μπορεί να γίνει σε P-Code ή σε Native Code. Αν γίνει σε P-Code, τότε η εφαρμογή μεταγλωττίζεται σε ψευδοκώδικα τον οποίο η κεντρική μονάδα επεξεργασίας δεν μπορεί να εκτελέσει αμέσως. Προγράμματα γραμμένα σε μια γλώσσα Interpreter όπως αυτή που χρησιμοποιούσε η BASIC, δε μεταφράζονται σε γλώσσα μηχανής πριν εκτελεστούν αλλά κάθε γραμμή κώδικα μεταφράζεται σε γλώσσα μηχανής όταν χρειάζεται και κατόπιν εκτελείται.



Εικόνα 4.7 Ο Compiler της Visual Basic

Η μεταγλώττιση που χρησιμοποιείται συνήθως είναι σε Native Code. Με αυτή την επιλογή, η Visual Basic μεταγλωττίζει μια εφαρμογή χρησιμοποιώντας εγγενή κώδικα, ο οποίος είναι η γλώσσα μηχανής, δηλαδή αυτό που καταλαβαίνει και εκτελεί η κεντρική μονάδα επεξεργασίας. Το παραγόμενο εκτελέσιμο είναι ταχύτερο από το ισοδύναμο εκτελέσιμο P-Code κατά περίπου είκοσι φορές. Όταν γίνει επιλογή για μεταγλώττιση σε εγγενή κώδικα, ο Compiler της Visual Basic δίνει στο χρήστη τις παρακάτω πέντε επιλογές :

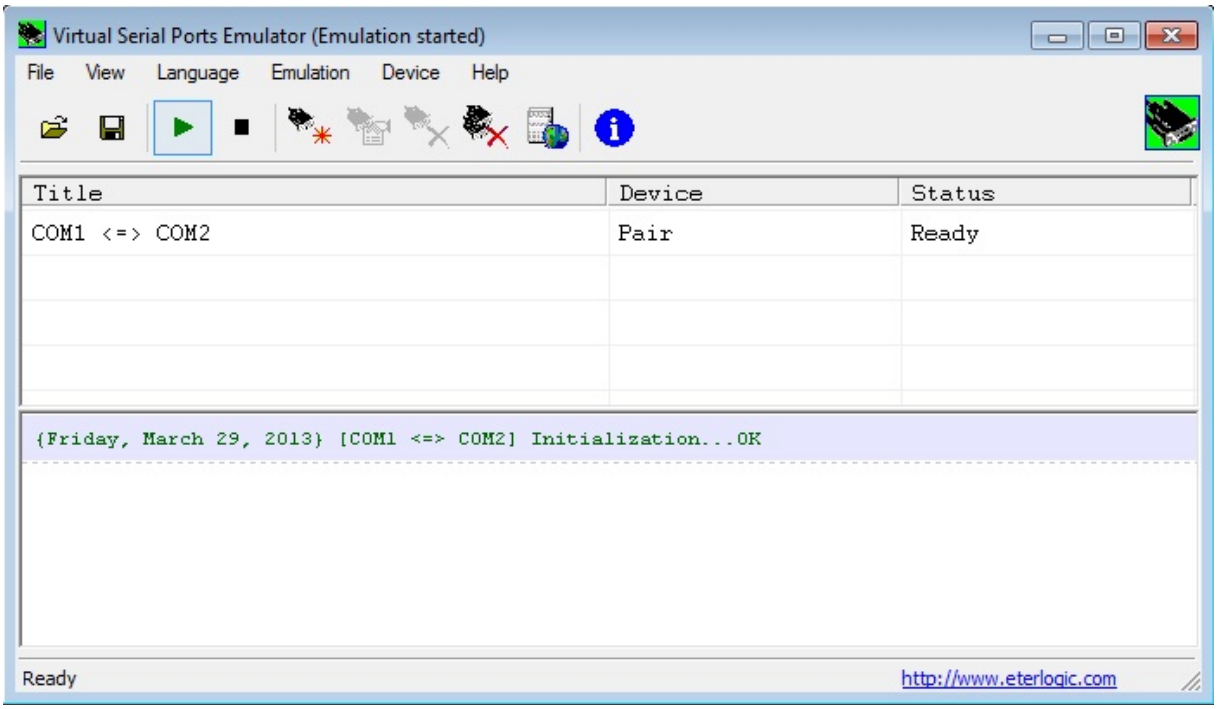
- Optimize for Fast Code: μεγιστοποιεί την ταχύτητα του εκτελέσιμου αρχείου λέγοντας στο μεταγλωττιστή να προτιμήσει την ταχύτητα από το μέγεθος.
- Optimize for Small code: μεγιστοποιεί το μέγεθος του εκτελέσιμου αρχείου λέγοντας στο μεταγλωττιστή να προτιμήσει το μέγεθος από την ταχύτητα.
- No Optimization: μεταγλωττίζει χωρίς βελτιστοποιήσεις.
- Favor Pentium Pro: βελτιστοποιεί τον κώδικα για επεξεργαστή Pentium Pro. Η επιλογή αυτή χρησιμοποιείται για προγράμματα που είναι μόνο για Pentium Pro. Ο κώδικας που παράγεται με αυτή την επιλογή εκτελείται και σε άλλους επεξεργαστές, αλλά δεν έχει τόσο καλή απόδοση, όση θα είχε αν μεταγλωττιζόταν με άλλες επιλογές.
- Create Symbolic Debug Info: παράγει πληροφορίες συμβολικής διόρθωσης σφαλμάτων στο εκτελέσιμο αρχείο. Ένα εκτελέσιμο αρχείο που δημιουργείται με αυτήν την επιλογή μπορεί να διορθωθεί με την Visual C++. Η επιλογή αυτή συνήθως χρησιμοποιείται από προγραμματιστές της Visual C++ που χρησιμοποιούν επίσης τη Visual Basic.

Πέρα από την κατασκευή του εκτελέσιμου αρχείου, η Visual Basic έχει ένα εργαλείο συσκευασίας και διανομής των εφαρμογών, το οποίο ονομάζεται "Package & Deployment Wizard". Ο "Package & Deployment Wizard" είναι ένα εύχρηστο εργαλείο για την κατασκευή προγραμμάτων εγκατάστασης λογισμικού, φροντίζει δηλαδή να δημιουργήσει μια νέα εφαρμογή της οποίας μοναδικός σκοπός είναι να εγκαταστήσει την εφαρμογή σε έναν υπολογιστή. Ο "Package & Deployment Wizard", μαζί με το εκτελέσιμο αρχείο, συλλέγει και όλα τα αρχεία υποστήριξης (OCX και DLL) που χρειάζεται η εφαρμογή και έπειτα κατασκευάζει ένα πρόγραμμα εγκατάστασης για την εφαρμογή.

4.3 Virtual Serial Ports Emulator της “Eterlogic”

Για τις ανάγκες της πτυχιακής χρησιμοποιήθηκε μια πολύ χρήσιμη και απλή εφαρμογή για την εξομοίωση συνδέσεων μέσω σειριακών θυρών. Η εφαρμογή αυτή είναι ένας εξομοιωτής εικονικών σειριακών θυρών, ο VSPE (Virtual Serial Ports Emulator) της εταιρείας “Eterlogic” και οι πληροφορίες του φαίνονται στην Εικόνα 4.8. Η διεπαφή του εξομοιωτή φαίνεται στην Εικόνα 4.9 όπου διακρίνεται το μενού, οι βασικές λειτουργίες του και μια δημιουργημένη σύνδεση μεταξύ εικόνων σειριακών θυρών.

Εικόνα 4.8 Πληροφορίες VSPE

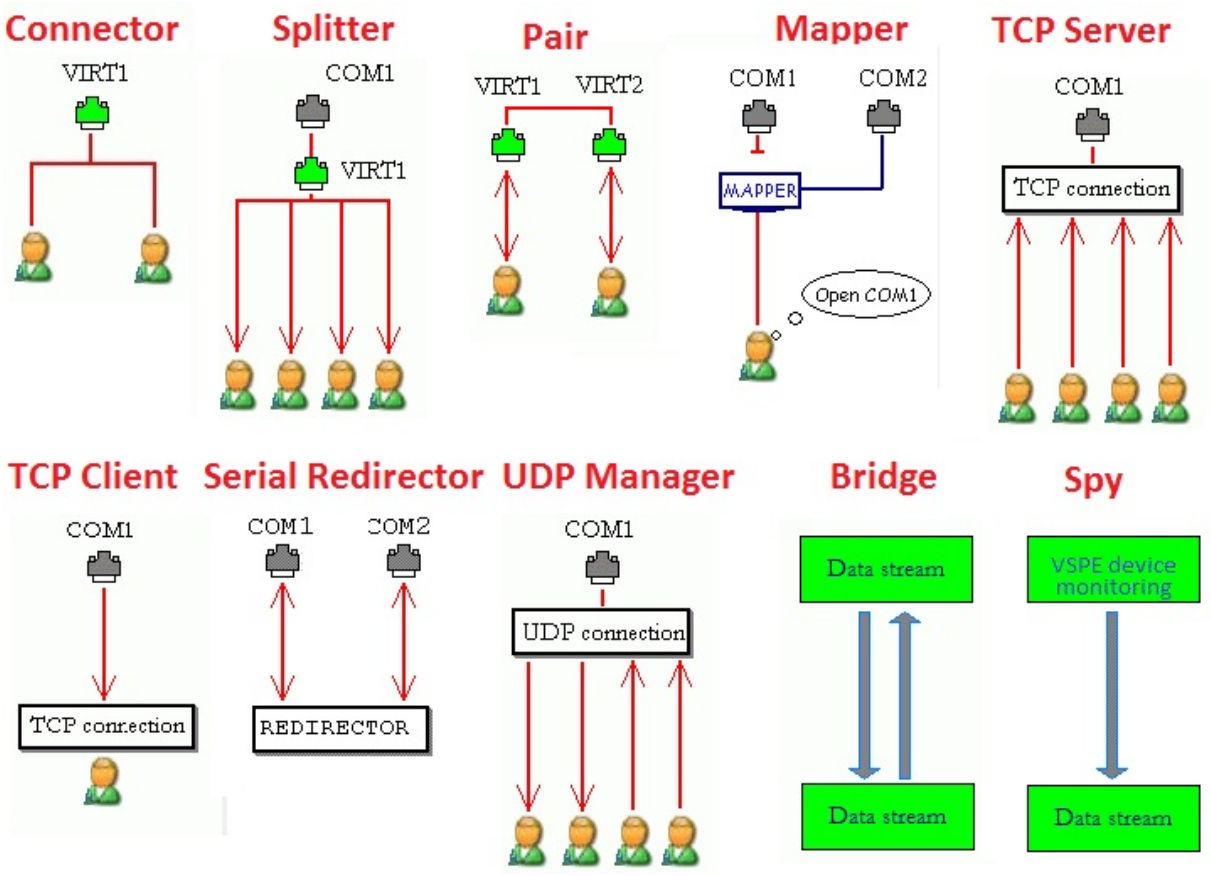


Εικόνα 4.9 Διεπαφή VSPE

Με τον VSPE είναι δυνατόν να δημιουργηθούν δέκα διαφορετικές εικονικές συνδέσεις και να “τρέχουν” ταυτόχρονα ή ξεχωριστά κάθε φορά, πατώντας το κουμπί “Play” που φαίνεται στη παραπάνω εικόνα. Οι συνδέσεις αυτές παρουσιάζονται στην και αναλύονται παρακάτω:

- Connector: Ο connector είναι μια εικονική συσκευή που μπορεί να εκκινήσει δύο φορές και όταν γίνει αυτό δημιουργεί ένα αγωγό δεδομένων ανάμεσα στους πελάτες της. Αυτή η λειτουργία επιτρέπει σε ξεχωριστές εφαρμογές να χρησιμοποιούν την ίδια σειριακή θύρα για την ανταλλαγή δεδομένων.

- Splitter: Ο διαμοιραστής δεδομένων δημιουργεί μια εικονική συσκευή, η οποία αντιπροσωπεύει μια υπάρχουσα σειριακή θύρα και την διαμοιράζει ανάμεσα σε διαφορετικές εφαρμογές.
- Pair: Η συσκευή αυτή χρησιμοποιήθηκε στην παρούσα πτυχιακή και η λειτουργία της είναι να δημιουργεί και να ενώνει, μεταξύ τους, δύο εικονικές σειριακές θύρες.
- Mapper: Ο Mapper μπορεί να ανακατευθύνει της αιτήσεις από μια σειριακή θύρα σε μια άλλη. Είναι πολύ σημαντική η χρησιμοποίηση του όταν χρειάζεται να εναλλάσσουμε σειριακές θύρες για εφαρμογές όπου δεν μπορούμε να αλλάζουμε τις ιδιότητες, όπως σε εφαρμογές τρίτων κατασκευαστών. Η συσκευή αυτή όπως και οι επόμενες που αναλύονται παρακάτω δεν δημιουργούν νέες εικονικές σειριακές θύρες.
- TCPServer: Ο TCPServer ανοίγει μια θύρα TCP για την μεταφορά δεδομένων μιας υπάρχουσας σειριακής θύρας στους πελάτες της. Δεν υπάρχει όριο στον αριθμό των πελατών που συνδέονται και η συσκευή αυτή.
- TCPClient: Η συσκευή αυτή εγκαθιδρύει μια σύνδεση TCP/IP με έναν απομακρυσμένο υπολογιστή και μεταφέρει δεδομένα από μια υπάρχουσα σειριακή θύρα στον υπολογιστή αυτό. Αν η σύνδεση αυτή χαθεί για κάποιο λόγο, γίνεται αυτομάτως επανασύνδεση.
- Serial Redirector: Ο Serial Redirector ανακατευθύνει δεδομένα μεταξύ δυο σειριακών θυρών.
- UDP Manager: Η συσκευή αυτή μπορεί να στείλει και να λάβει δεδομένα μιας σειριακής θύρας μέσω του πρωτοκόλλου UDP. Επίσης μπορεί να ρυθμιστεί ώστε να ανταλλάσει δεδομένα μεταξύ πολλών πελατών.
- Bridge: Η συσκευή Bridge συνδέει ροές δεδομένων.
- Spy: Αυτή η συσκευή επιτρέπει την παρακολούθηση μιας VSPE συσκευής, η οποία υποστηρίζει την λειτουργία αυτή.



Εικόνα 4.10 Λειτουργίες VSPE

5 Εφαρμογή εικονικού κόμβου ασύρματης τηλεμετρίας

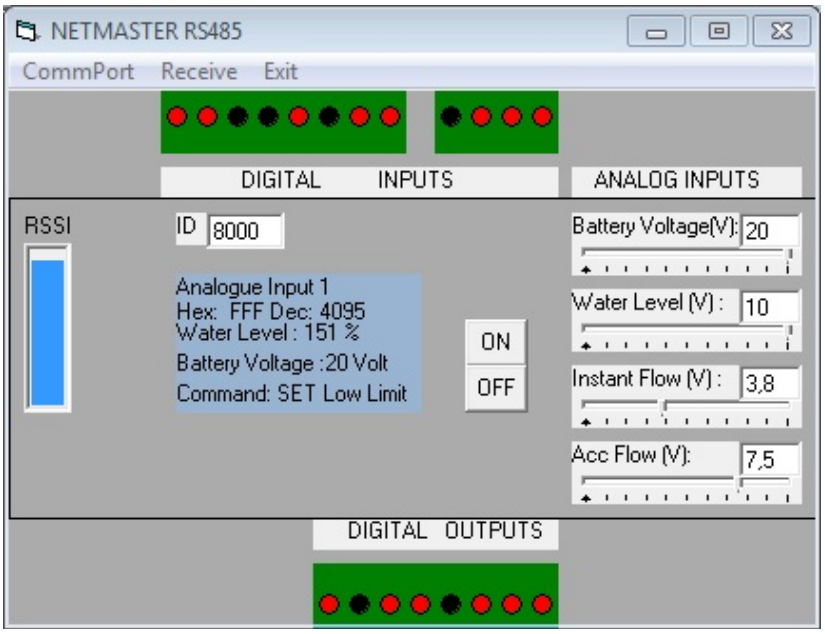
Μια εφαρμογή που αντικαθιστά κόμβους ασύρματης τηλεμετρίας είναι ένα χρήσιμο "εργαλείο" για τους προγραμματιστές εφαρμογών ελέγχου απομακρυσμένων τερματικών συσκευών SCADA. Μια τέτοια συσκευή είναι και ο Προγραμματιζόμενος Λογικός Ελεγκτής (PLC) Netmaster RS485 της "Elsist" που παρουσιάστηκε στο Κεφάλαιο 3 και η εφαρμογή που αναπτύχθηκε ώστε να εξομοιώνει αυτήν την συσκευή ονομάζεται "Netmaster RS485 Emulator" (Εξομοιωτής του Netmaster RS485).

Η ανάλυση, ο τρόπος ανάπτυξης και ο κώδικας, σε Visual Basic 6, της εν λόγω εφαρμογής θα παρουσιαστούν στο παρόν κεφάλαιο. Η ανάπτυξη της εφαρμογής χωρίζεται σε τρεις ενότητες ή αλλιώς στάδια όπου στο πρώτο στάδιο αναφέρεται η δημιουργία της Διεπαφής της εφαρμογής. Το δεύτερο στάδιο αναφέρεται στην ανάπτυξη του κώδικα του αντικειμένου ViComm το οποίο ασχολείται με την επικοινωνία και στο τρίτο αναλύεται ο κώδικας που αναπτύχθηκε για τις αρχικοποιήσεις και τις λοιπές λειτουργίες της εφαρμογής. Επίσης για καλύτερη κατανόηση του τρόπου εκτέλεσης του κώδικα, τοποθετήθηκε στο τέλος του κεφαλαίου το Διάγραμμα Ροής (flowchart) της εφαρμογής.

5.1 Διεπαφή Εφαρμογής "Netmaster RS485 Emulator"

Στο πρώτο στάδιο της ανάπτυξης της εφαρμογής μελετήθηκε η διεπαφή του προγραμματιζόμενου ελεγκτή Netmaster RS485 και επισημάνθηκαν τα στοιχεία, τα οποία είναι χρήσιμα για την μελέτη της λειτουργίας του. Στη συνέχεια έγινε η σχεδίαση της διεπαφής του εξομοιωτή με κριτήριο τον ελεγκτή και την λειτουργικότητα της εφαρμογής. Επίσης προστέθηκαν στοιχεία στην εφαρμογή, τα οποία δεν υπάρχουν στην συσκευή, αλλά ήταν αναγκαία για την προσημείωση ορισμένων λειτουργιών και μετρήσεων που γίνονται αυτόματα από τον Netmaster RS485.

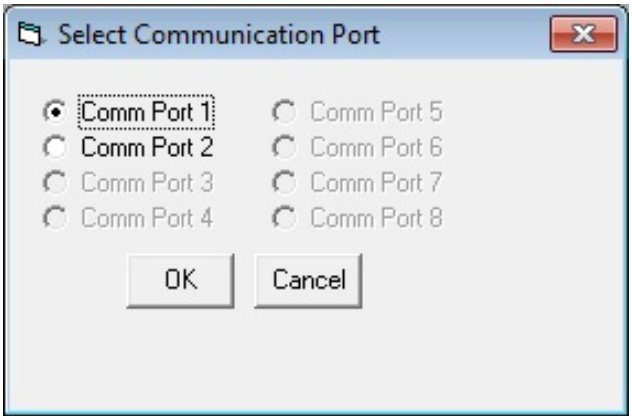
Η διεπαφή του εξομοιωτή φαίνεται στην Εικόνα 5.1 και η δομή της χωρίζεται σε τρία βασικά μέρη. Το πρώτο μέρος είναι το μενού επιλογών της εφαρμογής, το οποίο προστέθηκε για να εκτελεί ορισμένες βασικές λειτουργίες. Το δεύτερο μέρος είναι οι ψηφιακές είσοδοι και έξοδοι που βρίσκονται στο πάνω και κάτω μέρος της διεπαφής αντίστοιχα και το τρίτο μέρος, το οποίο βρίσκεται στο κέντρο, περιέχει της αναλογικές εισόδους, την οθόνη LCD και ορισμένες ακόμα λειτουργίες.



Εικόνα 5.1 Διεπαφή Netmaster RS485 Emulator

5.1.1 Μενού Επιλογών

Η βασική λειτουργία του εξομοιωτή είναι να επικοινωνεί με κάποια εφαρμογή ελέγχου, συσκευών SCADA και για να γίνει αυτό χρησιμοποιούνται εικονικές σειριακές θύρες. Στην πρώτη επιλογή του μενού επιλογών, υπάρχει η δυνατότητα να διαλέγεται, ποια από τις διαθέσιμες σειριακές θύρες θα αντιστοιχεί στην εφαρμογή του εξομοιωτή. Όταν γίνει επιλογή (κλικ) στο στοιχείο "CommPort" εμφανίζεται στην οθόνη μας το παράθυρο "Select Communication Port" το οποίο φαίνεται στην Εικόνα 5.2.



Εικόνα 5.2 Παράθυρο Επιλογής Σειριακής Θύρας

Στο παράθυρο "Select Communication Port" υπάρχει η δυνατότητα χρησιμοποίησης 8 σειριακών θυρών αλλά μπορούν να γίνουν και 256, δηλαδή ο μέγιστος αριθμός, με αναβάθμιση της εφαρμογής. Από αυτές τις θύρες, όπως φαίνεται στην παραπάνω εικόνα, είναι με γκρι χρώμα οι μη διαθέσιμες προς χρησιμοποίηση, ενώ με μαύρο οι διαθέσιμες. Επίσης κάθε θύρα έχει δίπλα της ένα κουμπί επιλογής (Option Button) και είναι επιλεγμένο εκείνο το οποίο αντιστοιχεί στην θύρα που χρησιμοποιείτε την συγκεκριμένη στιγμή από την εφαρμογή.

Μόλις ο χρήστης επιλέξει την θύρα επικοινωνίας που επιθυμεί και πατώντας το κουμπί εντολής (Command Button) "OK" ενεργοποιείτε το κομμάτι του κώδικα που αλλάζει την αντιστοιχία της θύρας. Ενώ πατώντας το "Cancel" επιστρέφει στην αρχική διεπαφή της εφαρμογής και το ίδιο συμβαίνει και όταν πατήσει το κουμπί εξόδου που βρίσκεται στην πάνω δεξιά γωνία του παραθύρου "Select Communication Port".

Η επόμενη επιλογή στο μενού είναι το "Receive" το οποίο όταν επιλεχτεί, δηλαδή γίνει "κλικ" σε αυτό, εμφανίζεται το παράθυρο που φαίνεται στην Εικόνα 5.3. Αν η ιδιότητα "Receive Enabled" είναι επιλεγμένη, τότε ο εξομοιωτής δέχεται εισερχόμενα πακέτα και η σειριακή θύρα που αντιστοιχεί σε αυτόν είναι ανοιχτή. Διαφορετικά η θύρα είναι κλειστή και δεν λαμβάνονται πακέτα δεδομένων. Η εναλλαγή της επιλογής αυτής γίνεται επίσης και με το ταυτόχρονο πάτημα των πλήκτρων "Ctrl" και "R", από το πληκτρολόγιο.

Εικόνα 5.3 Επιλογή Μενού "Receive"

Η έξοδος από την εφαρμογή μπορεί να γίνει με δύο τρόπους. Ο πρώτος τρόπος είναι το πάτημα της τρίτης επιλογής στο μενού επιλογών, η οποία είναι η "Exit". Ενώ ο δεύτερος είναι το πάτημα στο κουμπί εξόδου, το οποίο βρίσκεται στην πάνω δεξιά γωνία του παραθύρου της εφαρμογής και συναντάται σχεδόν σε όλες τις εφαρμογές που έχουν δημιουργηθεί με Visual Basic. Επίσης στο σημείο εκείνο της εφαρμογή υπάρχει και το κουμπί ελαχιστοποίησης και το κουμπί μεγιστοποίησης, το οποίο είναι ανενεργό στην συγκεκριμένη εφαρμογή.

5.1.2 Ψηφιακές Είσοδοι και Έξοδοι

Ο Προγραμματιζόμενος Ελεγκτής Netmaster RS485, όπως αναφέρεται στην Παράγραφο 3.2, είναι εξοπλισμένος με ψηφιακές εισόδους και εξόδους, μέσω των οποίων ενημερώνει και παρουσιάζει την κατάσταση συγκεκριμένων διακοπτών. Οι ψηφιακές είσοδοι και έξοδοι φαίνονται στην Εικόνα 3.6 και σχεδιάστηκαν στον εξομοιωτή ως ActiveX αντικείμενα, όπως παρουσιάζεται στην Εικόνα 5.1, με την ονομασία "LED".

Τα αντικείμενα αυτά έχουν δύο καταστάσεις που αντιστοιχούν σε διαφορετικά χρώματα της ιδιότητας "FillColor". Τα χρώματα που εναλλάσσονται στα αντικείμενα είναι μαύρο, το οποίο αντιστοιχεί σε σβηστό LED και κόκκινο, όπου αντιστοιχεί σε αναμμένο LED. Η κατάσταση των εξόδων μπορεί να αλλάξει από τον χρήστη με το πάτημα (κλικ) πάνω στο αντίστοιχο LED. Ενώ οι είσοδοι αλλάζουν κατάσταση με συγκεκριμένη εντολή, μέσω μηνύματος, του προγράμματος ελέγχου του Netmaster.

5.1.3 Βασικό Μέρος Διεπαφής

Στο κέντρο του παράθυρου της εφαρμογής βρίσκονται οι βασικές λειτουργίες της διεπαφής του εξομοιωτή Netmaster RS485. Καθώς η εφαρμογή λαμβάνει ένα μήνυμα, περιέχεται σε αυτό η τιμή της ισχύς του σήματος σε εκείνη τη στιγμή. Αυτή η τιμή εμφανίζεται σε μια μπάρα που βρίσκεται στο αριστερό μέρος της εφαρμογής και έχει εύρος τιμών από 0 έως 255. Δεξιότερα βρίσκεται ένα πλαίσιο κειμένου (TextBox) στο οποίο εμφανίζεται η αναγνωριστική διεύθυνση του εξομοιωτή και προσφέρει την δυνατότητα να γίνει τροποποίηση της διεύθυνσης αυτής.

Στο κέντρο της εφαρμογής έχει σχεδιαστεί, όπως υπάρχει και στον "Netmaster RS485", μια οθόνη LCD με περισσότερες όμως γραμμές και χαρακτήρες από ότι στον ελεγκτή. Στον εξομοιωτή οι γραμμές στην οθόνη είναι πέντε ώστε να εμφανίζονται εκεί πληροφορίες και μετρήσεις, οι οποίες είναι χρήσιμες για την παρακολούθηση των λειτουργιών της εφαρμογής. Στην πρώτη γραμμή αναγράφεται συνεχώς "Analogue Input 1", δηλώνοντας έτσι ότι στη δεύτερη γραμμή εμφανίζεται η τιμή της δεύτερης αναλογική εισόδου δηλαδή εκείνης με αριθμό 1, καθώς αριθμούνται με αρχή το μηδέν.

Όπως φαίνεται και στην Εικόνα 5.1, η τιμή της δεύτερης αναλογικής εισόδου εμφανίζεται σε δύο μορφές, δεκαεξαδική και δεκαδική. Επίσης στην αμέσως επόμενη γραμμή, δηλαδή στην τρίτη από την αρχή, αναγράφεται το επίπεδο του νερού της δεξαμενής που παρακολουθείται σε ποσοστό επί της εκατό. Αυτό το ποσοστό προκύπτει μετά από πράξεις με την δεύτερη αναλογική είσοδο, το ανώτατο και το κατώτατο όριο που έχει οριστεί από τον χρήστη για την δεξαμενή αυτή. Ενώ στην τέταρτη γραμμή εμφανίζεται το επίπεδο ενέργειας της μπαταρίας του Netmaster RS485 Emulator και στην τελευταία γραμμή εμφανίζεται η εντολή, αν υπάρχει, που περιέχεται στο μήνυμα το οποίο λαμβάνει ο εξομοιωτής κάθε φορά. Στο στιγμιότυπο της εφαρμογής που φαίνεται στην Εικόνα 5.1 η εντολή που έχει λάβει η εφαρμογή είναι να αλλάξει το κατώτατο όριο του νερού για την δεξαμενή.

Στα δεξιά της οθόνης LCD υπάρχουν δύο κουμπιά εντολής (Command Buttons), το πρώτο που έχει ως τίτλο "ON" όταν πατηθεί ανάβει ένα συγκεκριμένο LED ενώ αυτό με τίτλο "OFF" το σβήνει. Αμέσως δεξιότερα και στο δεξιότερο μέρος της εφαρμογής βρίσκονται τέσσερεις ολισθητές επιλογής τιμής και πάνω από τον καθένα είναι η ετικέτα του και το πλαίσιο κειμένου, στο οποίο εμφανίζεται η επιλεγμένη τιμή.

Οι ολισθητές αυτοί σχεδιάστηκαν στην διεπαφή αν και δεν υπάρχουν στον ελεγκτή αντιπροσωπεύοντας ο καθένας μια αναλογική είσοδο. Οι τέσσερεις αναλογικές είσοδοι υπάρχουν στον Netmaster RS485 για να εισέρχονται σε αυτόν πληροφορίες από αισθητήρες όμως στην εφαρμογή γίνεται προσημείωση των λειτουργιών του ελεγκτή και για το λόγω αυτόν οι ολισθητές δίνουν τις τιμές αυτές με την παρέμβαση του χρήστη. Ο πρώτος ολισθητής είναι για το επίπεδο ενέργειας της μπαταρίας και μετριέται σε Volt όπως και οι επόμενοι ολισθητές. Ο δεύτερος δίνει το επίπεδο νερού της υποτιθέμενης δεξαμενής ενώ ο τρίτος δίνει την τιμή του "Instant Flow", δηλαδή της στιγμιαίας ροής και ο τέταρτος του "Accumulate Flow", δηλαδή της συνολικής ροής.

5.2 ActiveX αντικείμενο “ViComm”

Η ραγδαία ανάπτυξη των υπολογιστών οφείλεται κατά πολύ στην δημιουργία δικτύων και συνεπώς την επικοινωνία διαφόρων συστημάτων μεταξύ τους. Η επίτευξη της επικοινωνίας, ιδιαίτερα μιας απομακρυσμένης, μεταξύ εφαρμογών είναι ένα σημαντικό κομμάτι του προγραμματισμού, στο οποίο χρησιμοποιούνται διάφορα πρωτόκολλα δικτύων. Για την επικοινωνία του εξομοιωτή Netmaster με το πρόγραμμα ελέγχου χρησιμοποιήθηκε το πρωτόκολλο ASCII και με βάση το αντικείμενο “MSComm”, ενός από τα στοιχεία (Components) της Visual Basic 6, αναπτύχθηκε το αντικείμενο ActiveX με ονομασία “ViComm” (Virtual Communication).

Το αντικείμενο “ViComm” είναι εκείνο το κομμάτι του κώδικα της εφαρμογής, το οποίο αναγνωρίζει και αποκωδικοποιεί τα πακέτα δεδομένων. Όπως φαίνεται στον κώδικα της συνάρτησης “BuffHandler”, όταν η εικονική σειριακή θύρα, που έχουμε αντιστοιχίσει με το πρόγραμμα “VSPE” στην εφαρμογή μας, λαμβάνει δεδομένα τότε το “ViComm” τα διαβάζει περιμένοντας ένα έγκυρο μήνυμα. Για να αναγνωρίσει το “ViComm” ένα πακέτο ως έγκυρο μήνυμα πρέπει αρχικά τα δύο πρώτα bytes, μετά την αποκωδικοποίηση βάση του πίνακα ASCII, να είναι το σύμβολο αστερίσκος (*).

Μόλις το “ViComm” αναγνωρίσει την αρχή ενός πιθανού έγκυρου μηνύματος αρχίζει να αποθηκεύει τα bytes σε μια συμβολοσειρά, της οποίας το μήκος θα είναι ίσο με το μήκος του πακέτου. Η τιμή που προσδιορίζει το μήκος του πακέτου, όπως αναφέρεται και στην Παράγραφο 3.3, υπάρχει αποθηκευμένη στο τρίτο byte του πακέτου . Όταν αποθηκευτεί ο αριθμός των bytes που αναμένεται, η εφαρμογή ελέγχει εάν το πακέτο προορίζεται για τον συγκεκριμένο εξομοιωτή. Ο έλεγχος γίνεται χρησιμοποιώντας την αποθηκευμένη, στο πακέτο, διαδρομή από διευθύνσεις (Route) και την διεύθυνση (ID) του εξομοιωτή. Εάν δεν προορίζεται γι’ αυτόν, το “ViComm”προωθεί το πακέτο στη συσκευή με διεύθυνση την επόμενη, κατά την αποθηκευμένη διαδρομή, από αυτήν που έχει η συσκευή στην οποία «τρέχει».

Στη συνέχεια και αν το πακέτο προορίζεται γι’ αυτήν την συσκευή, ελέγχει αν έχει ληφθεί σωστά, δηλαδή χωρίς λάθη. Τον έλεγχο αυτόν τον πραγματοποιεί με την βοήθεια του αθροίσματος ελέγχου, το οποίο είναι αποθηκευμένο στο τελευταίο byte του πακέτου. Το ελεγμένο για την εγκυρότητα του μήνυμα, το “ViComm” το αποκωδικοποιεί, δηλαδή αποθηκεύει τις πληροφορίες που χρειάζεται στις αντίστοιχες μεταβλητές και εξάγει το μήνυμα που προορίζεται για την κεντρική εφαρμογή. Τέλος ενημερώνει την εφαρμογή ότι έχει ληφθεί ένα έγκυρο μήνυμα προωθώντας το σε αυτήν ώστε να διαχειριστεί τα δεδομένα.

Το “ViComm” έχει επίσης το καθήκον να στέλνει πακέτα, όπως στην προαναφερθείσα περίπτωση όπου προωθεί το πακέτο που δεν προορίζεται γι’ αυτόν. Επίσης το μήνυμα απάντησης, το οποίο δημιουργεί η κεντρική εφαρμογή για να το στείλει πίσω στην εφαρμογή ελέγχου, το κωδικοποιεί το “ViComm”. Με το μήνυμα αυτό συντάσσει ένα πακέτο που περιέχει και τις αναγκαίες πληροφορίες για την μετάδοση του και το στέλνει πίσω στην εφαρμογή ελέγχου.

Η συνάρτηση του “ViComm” που αναλαμβάνει να δημιουργήσει και να στείλει το πακέτο απάντησης είναι η “SendReply”. Σε αυτήν την συνάρτηση εκχωρείτε η τιμή στα byte, του πακέτου απάντησης, ένα προς ένα και πάντα βάση της δομής που έχει περιγραφεί στην Παράγραφο 3.3.3. Η συνάρτηση “SendReply” συλλέγει και αποθηκεύει τις πληροφορίες από τα αντικείμενα που βρίσκονται στην κεντρική εφαρμογή και έχουν το ρόλο των εικονικών αισθητήρων. Επίσης αντιστρέφει την διαδρομή διευθύνσεων που έχει εξαχτεί από το πακέτο λήψης και αυξάνει κατά ένα του αύξοντα αριθμό πακέτου. Στη συνέχεια περιλαμβάνοντας και το άθροισμα ελέγχου στο πακέτο, το προωθεί στη εικονική σειριακή θύρα ώστε να μεταφερθεί στην εφαρμογή ελέγχου.

Τον υπολογισμό του αθροίσματος ελέγχου και στις δύο περιπτώσεις που είναι αναγκαίο, όπως περιγράφηκε παραπάνω, τον αναλαμβάνει η συνάρτηση “CheckSum”. Το άθροισμα αυτό δημιουργείτε με την αλφαριθμητική πράξη “XOR” και ξεκινώντας από τα δύο πρώτα bytes. Στο αποτέλεσμα των δύο πρώτων bytes γίνεται η πράξη “XOR” με το τρίτο byte. Αυτή η διαδικασία συνεχίζεται μέχρι το προτελευταίο byte του πακέτου, καθώς το τελευταίο byte θα είναι το αποτέλεσμα όλων αυτών των πράξεων. Τέλος στο αντικείμενο “ViComm” υπάρχουν και οι συναρτήσεις εισαγωγής και εξαγωγής μεταβλητών, οι οποίες δηλώνονται ως Ιδιότητα (Property) και βοηθούν το αντικείμενο αυτό να επικοινωνεί με την κεντρική εφαρμογή.

5.3 Βασικό Μέρος Εφαρμογής και Αντικείμενα

Η εφαρμογή Netmaster RS485 Emulator αναλύθηκε και σχεδιάστηκε σε ξεχωριστά κομμάτια κώδικα για κάθε βασική λειτουργία. Αυτός ο τρόπος προγραμματισμού ακολουθείτε από τους περισσότερους προγραμματιστές, ώστε να είναι δυνατή η εύκολη αναβάθμιση του κώδικα και η αντιμετώπιση λαθών από τους ίδιους αλλά και μελλοντικούς προγραμματιστές. Η ανάπτυξη σε αυτήν την εφαρμογή ξεκίνησε από το κομμάτι της επικοινωνίας, το οποίο περιγράφηκε στην προηγούμενη παράγραφο, καθώς είναι το σημαντικότερο και μπορεί ο ίδιος κώδικας να χρησιμοποιηθεί και σε άλλες παρόμοιες εφαρμογές. Και στην συνέχεια δημιουργήθηκε η κεντρική φόρμα και τα υπόλοιπα αντικείμενα που ήταν αναγκαία.

Ο κώδικας του εξομοιωτή του προγραμματιζόμενου ελεγκτή Netmaster RS485 αποτελείτε από πέντε αντικείμενα, τα οποία είναι τα εξής:

- ViComm: Αντικείμενο ActiveX, υπεύθυνο για την επικοινωνία του εξομοιωτή με άλλες εφαρμογές.
- LED: Αντικείμενο ActiveX, το οποίο λειτουργεί ως φωτάκι για την ένδειξη της καταστάσεις των ψηφιακών εισόδων και εξόδων.
- FrmSetCommPort: Φόρμα, η οποία εμφανίζεται από το μενού επιλογών και μας δίνει την δυνατότητα να επιλέξουμε σειριακή θύρα για την εφαρμογή.
- MdIni_file: Module, υπεύθυνο για την αποθήκευση και ανάκτηση παραμέτρων από το αρχείο παραμέτρων ώστε να μην χάνονται οι καταστάσεις, οι μετρήσεις και οι αρχικοποιήσεις κατά το κλείσιμο της εφαρμογής.
- ViEmulator: Κεντρική Φόρμα, στην οποία υπάρχει ο κώδικας δημιουργίας και παρουσίασης των εικονικών μετρήσεων και καταστάσεων του εξομοιωτή. Επίσης περιέχει τον κώδικα διαχείρισης των εντολών που δέχεται ο εξομοιωτής με τα πακέτα λήψης.

5.3.1 “LED”: ActiveX Αντικείμενο

Ο προγραμματιζόμενος ελεγκτής Netmaster RS485, όπως περιγράφηκε στην παράγραφο 3.2, είναι εφοδιασμένος με ψηφιακές εισόδους και εξόδους και η κατάσταση τους παρουσιάζεται σε LED. Για να γίνει η εξομοίωση της λειτουργίας τους, δημιουργήθηκαν αντικείμενα ActiveX με ονομασία “LED”. Τα αντικείμενα αυτά έχουν σαν βασική λειτουργία να ανταποκρίνονται στο δεξιό πάτημα (κλικ) του ποντικιού ώστε να “ανάβουν” ή να “σβήνουν” ανάλογα την κατάσταση τους εκείνη τη στιγμή. Το αναμμένο “LED” διακρίνεται με το κόκκινο χρώμα και το σβηστό με το μαύρο.

Τα αντικείμενα “LED” ανταποκρίνονται και σε εντολές, για την αλλαγή της κατάστασης τους, που προέρχονται από την κεντρική εφαρμογή. Επίσης με την κεντρική εφαρμογή επικοινωνούν μέσω της ιδιότητας “LedOn” και να την ενημερώνουν για την κατάσταση τους, ώστε να μπορεί με την σειρά της να αποθηκεύει αυτές τις καταστάσεις σε αριθμούς ή να τις στέλνει στην εφαρμογή ελέγχου μέσω του πακέτου απάντησης.

5.3.2 “frmSetCommPort”: Φόρμα Επιλογής Σειριακής Θύρας

Η εφαρμογή Netmaster RS485 Emulator μας δίνει την δυνατότητα να επιλέξουμε ποια σειριακή θύρα θα αντιστοιχεί σε αυτήν. Την σειριακή θύρα την επιλέγουμε, όπως περιγράφηκε και στην Παράγραφο 5.1.1, μέσω του Παράθυρου Επιλογής Σειριακής Θύρας, το οποίο εμφανίζεται όταν κάνουμε “κλικ” στην επιλογή “CommPort” στο μενού. Μόλις εμφανιστεί το παράθυρο εκτελείτε η συνάρτηση “GetAvailableCommports”, η οποία βρίσκει ποιες σειριακές θύρες είναι διαθέσιμες στον υπολογιστή που τρέχει η εφαρμογή.

Στην συνέχεια βρίσκει ο κώδικας της φόρμας αυτής, ποιά σειριακή θύρα είναι ήδη κατειλημμένη από την εφαρμογή και έτσι εμφανίζονται στο παράθυρο, που φαίνεται στην Εικόνα 5.2, οι σειριακές θύρες με την κατάσταση τους. Τέλος όταν πατηθεί το κουμπί εντολής “OK” “τρέχει” το κομμάτι του κώδικα, το οποίο αντιστοιχεί στην εφαρμογή την σειριακή θύρα που έχει επιλέξει ο χρήστης από το παράθυρο επιλογής σειριακής θύρας.

5.3.3 "mdIni_file": Module για την αποθήκευση παραμέτρων

Το module "mdIni_file" είναι ένα αντικείμενο που έχει αναπτυχθεί στην Visual Basic 6, με σκοπό να χρησιμοποιείτε από οποιαδήποτε εφαρμογή δημιουργούμε ώστε να αποθηκεύει παραμέτρους σε ένα αρχείο αρχικοποιήσεων (Initialization file – INI). Το module αυτό περιέχει δύο συναρτήσεις, όπου η "AddToINI" αποθηκεύει δεδομένα και αρχικοποιήσεις στο επιλεγμένο αρχείο και αν δεν υπάρχει κάποια εγγραφή για τα δεδομένα αυτά, την δημιουργεί μια καινούρια. Η συνάρτηση "GetFromINI" ανασύρει τις αποθηκευμένες παραμέτρους από το επιλεγμένο αρχείο και αν δεν υπάρχουν επιστρέφει και αποθηκεύει τις αρχικοποιήσεις που δίνει ο χρήστης.

5.3.4 "ViEmulator": Κεντρική Φόρμα Εφαρμογής

Κάθε διεπαφή περιέχει μια κεντρική φόρμα, η οποία είναι αυτή που εμφανίζεται πρώτη όταν εκκινήσει η εφαρμογή. Δηλαδή είναι το αρχικό παράθυρο το οποίο περιέχει το μενού επιλογών και των κώδικα με τον οποίο εμφανίζονται τα αντικείμενα και καλούνται οι διάφορες συναρτήσεις. Η κεντρική φόρμα στην εφαρμογή Netmaster RS485 Emulator είναι η "ViEmulator" και όταν εμφανιστεί η πρώτη ενέργεια της είναι να ανασύρει, με την βοήθεια της συνάρτησης "GetFromINI", τις αρχικοποιήσεις που είναι αποθηκευμένες στο αρχείο "INI.

Στη συνέχεια, έχοντας αποθηκεύσει σε μεταβλητές τις τιμές που αντιστοιχούν στα αντικείμενα τα οποία εμφανίζονται στην διεπαφή, τους αλλάζει την κατάσταση βάση των μεταβλητών αυτών. Επίσης αντιστοιχίζει στην εφαρμογή την τελευταία χρησιμοποιούμενη ή, αν δεν υπάρχει ,την αρχική σειριακή θύρα και την "ανοίγει" ώστε να δέχεται πακέτα. Αντίθετα όταν κλείνει η εφαρμογή αποθηκεύει τις μεταβλητές και τον αριθμό της θύρας στο αρχείο παραμέτρων και κλείνει την θύρα.

Η θύρα μπορεί επίσης να κλείσει ή να ανοίξει καλώντας την συνάρτηση "menuReceiveEnabled_Click". Αυτό γίνεται με τον συνδυασμό των πλήκτρων "Ctrl+R" ή κάνοντας "κλικ" στο "Received Enabled" που είναι δεύτερο κατά σειρά στο μενού επιλογών. Οι άλλες δύο επιλογές στο μενού είναι η "CommPort" και η "Exit", όπου η πρώτη εμφανίζει το παράθυρο επιλογής σειριακής θύρας που περιγράφηκε στην Παράγραφο 5.3.2 και η δεύτερη καλεί την συνάρτηση που κλείνει την εφαρμογή.

Όσο μένει "ανοιχτή" η σειριακή θύρα, η εφαρμογή μπορεί να δεχτεί τα πακέτα τα οποία θα φτάσουν στη θύρα αυτή. Όταν λάβει ένα πακέτο η εφαρμογή τότε ο κώδικας του "ViComm", μετά που θα ελέγξει την εγκυρότητα του, το αποκωδικοποιήσει και εγείρεται το συμβάν "OnVMsg". Κατά συνέπεια καλείτε η συνάρτηση "ViComm_OnVMsg", η οποία αναγνωρίζει, από τον κωδικό της, την εντολή που υπάρχει μέσα στο μήνυμα και την εκτελεί.

Επίσης η συνάρτηση αυτή ενημερώνει την μπάρα που εμφανίζει την τιμή της ισχύς του σήματος κατά την λήψη του πακέτου (RSSI). Στη συνέχεια καλεί την συνάρτηση "refreshMe", η οποία ανανεώνει τις τιμές των αντικειμένων που περιέχονται στην διεπαφή και τέλος καλεί την συνάρτηση "SendReply" του "ViComm", η οποία περιγράφηκε στην παράγραφο 5.2 και στέλνει το πακέτο απάντησης.

Στο κεντρικό παράθυρο της εφαρμογής είναι σχεδιασμένα τα "LED" των ψηφιακών εισόδων και εξόδων. Στον κώδικα του παράθυρου αυτού υπάρχουν οι τρείς συναρτήσεις "LedToNum", "NumToLed" και "DecToBin", οι οποίες μετατρέπουν την κατάσταση των LED σε αριθμό και αντίστροφα. Πιο αναλυτικά, η συνάρτηση "DecToBin" μετατρέπει έναν δεκαδικό αριθμό στον ισοδύναμο δυαδικό με τόσα ψηφία όσα του ζητηθεί και τον αποθηκεύει σε μια συμβολοσειρά. Η συνάρτηση "LedToNum" συλλέγει τις καταστάσεις (0 ή 1) των LED και τις αποθηκεύει σε έναν δεκαδικό αριθμό. Αυτή η αποθήκευση γίνεται με τρόπο τέτοιο ώστε ο ισοδύναμος δυαδικός αριθμός να έχει ως ψηφία τις καταστάσεις των LED σε συγκεκριμένη σειρά.

Αντίθετα η συνάρτηση "NumToLed" μετατρέπει έναν δεκαδικό σε καταστάσεις 0 ή 1. Στην αρχή μετατρέπει τον δεκαδικό αριθμό σε δυαδικό με την βοήθεια της "DecToBin". Στην συνέχεια αντιστοιχίζει τα ψηφία του δυαδικού στα LED και ανάλογα την τιμή του ψηφίου αλλάζει και η κατάσταση των LED. Επομένως αν το ψηφίο που αντιστοιχεί στο πρώτο LED έχει τιμή 1 τότε το LED "ανάβει", δηλαδή το χρώμα του γίνεται κόκκινο ενώ αν είναι η τιμή 0 "σβήνει" και το χρώμα γίνεται μαύρο. Τέλος όταν αλλάζουμε την κατάσταση συνεπώς και την τιμή κάποιου αντικειμένου καλείται αυτόματα η συνάρτηση "refreshMe" και ανανεώνει τις τιμές των αντικειμένων της διεπαφής.

5.4 Ανάλυση του Διαγράμματος Ροής του Κώδικα (Flowchart)

Στις προηγούμενες παραγράφους του κεφαλαίου έγινε ανάλυση της διεπαφής της εφαρμογής και του τρόπου εκτέλεσης του κώδικα των αντικειμένων του εξομοιωτή. Για την καλύτερη κατανόηση του τρόπου λειτουργίας στο σύνολο του κώδικα και της αλληλεπίδρασης μεταξύ των αντικειμένων του, σχεδιάστηκε το διάγραμμα ροής του κώδικα της εφαρμογής (flowchart) και παρουσιάζεται στην Εικόνα 5.4.

Όπως φαίνεται και στο διάγραμμα με την εκκίνηση της εφαρμογής γίνεται ανάκτηση των αρχικοποιήσεων και των δεδομένων που υπήρχαν στην εφαρμογή πριν κλίσει την προηγούμενη φορά. Η ανάκτηση γίνεται από το αρχείο αρχικοποιήσεων (Initialization File) με την βοήθεια της συνάρτησης "GetFromINI" που υπάρχει στο module "mdIni_file". Στη συνέχεια η εφαρμογή τίθεται σε κατάσταση αναμονής έως ότου δοθεί κάποια εντολή από το χρήστη ή ληφθούν δεδομένα στην σειριακή θύρα που έχει αντιστοιχηθεί στον εξομοιωτή.

Οι εντολές που έχει την δυνατότητα να δώσει στην εφαρμογή ο χρήστης, μέσω των αντικειμένων τις, είναι οι εξής:

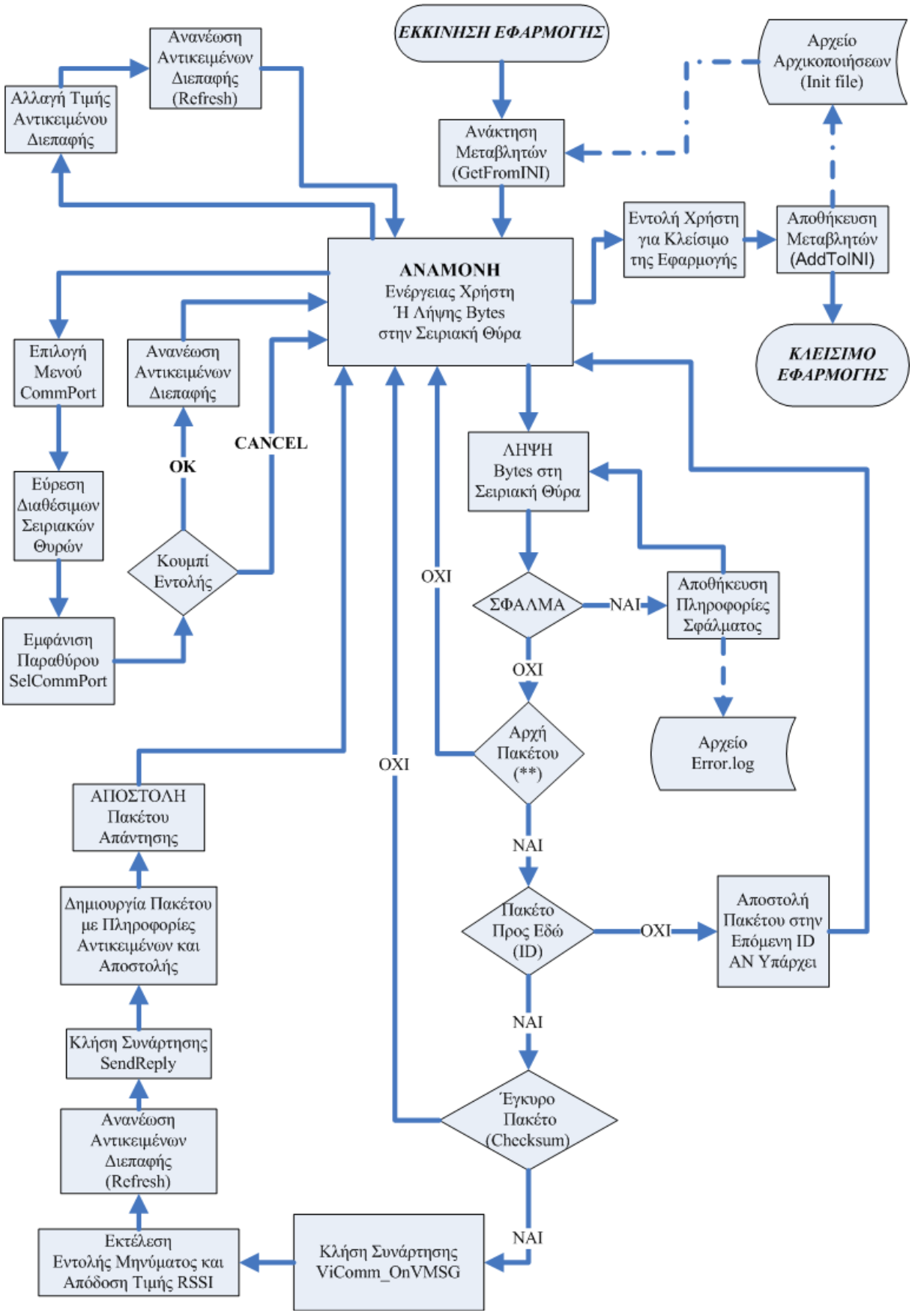
1. Το Κλείσιμο της εφαρμογής, όπου μπορεί να γίνει μέσω της επιλογής "Exit" από το μενού ή πατώντας το κουμπί που υπάρχει στην πάνω δεξιά στην διεπαφή. Όταν δοθεί η εντολή για το κλείσιμο, η εφαρμογή καταρχήν αποθηκεύει στο "Init File", μέσω της συνάρτησης "AddToINI", τα δεδομένα που υπάρχουν εκείνη την στιγμή στα αντικείμενα της διεπαφής και στις μεταβλητές του κώδικα και στη συνέχεια τερματίζει την εφαρμογή.
2. Την Αλλαγή Τιμής των αντικειμένων της διεπαφής, την οποία επιτυγχάνει ο χρήστης με τον κύλιση του κέρσορα του ποντικιού, την πληκτρολόγηση της επιθυμητής τιμής ή κάνοντας "κλικ" στο αντικείμενο, όπως στην περίπτωση της επιλογής "Receive" του μενού. Με την αλλαγή μιας τιμής η εφαρμογή ανανεώνει όλα τα αντικείμενα και τις μεταβλητές ώστε να υπάρχει σωστή πληροφόρηση γι' αυτά.
3. Την Αλλαγή της Θύρας Επικοινωνίας, όπου γίνεται μέσω του παραθύρου "SelCommPort" το οποίο εμφανίζεται όταν γίνει "κλικ" στην επιλογή "CommPort" του μενού και έχει δύο επιλογές, την επιβεβαίωση της επιλογής και την ακύρωση της.

Στην περίπτωση που γίνεται λήψη bytes από την σειριακή θύρα, με την οποία επικοινωνεί η εφαρμογή, τότε το αντικείμενο "ViComm" αρχίζει να διαβάζει τα bytes μέχρι να αναγνωρίσει την αρχή ενός πιθανού πακέτου αποθηκεύοντας, ταυτόχρονα, τα σφάλματα στο αρχείο "Error.log". Την αρχή πακέτου δεδομένων την αναγνωρίζει το "ViComm" όταν λάβει στη σειρά δύο χαρακτήρες "αστεράκι" και τότε λαμβάνει ολόκληρο το πακέτο αναγνωρίζοντας το τέλος του από την τιμή του επόμενο byte μετά τα "αστεράκια", όπως αναλύσαμε και στην Παράγραφο 3.3.1.

Όταν λάβει ολόκληρο το πακέτο, ελέγχει αν το πακέτο προορίζεται για τον παρόν κόμβο μέσω της μεταβλητής "ID" που είναι αποθηκευμένη στην εφαρμογή. Αν δεν προορίζεται για την εφαρμογή αυτή, προωθεί το μήνυμα στην επόμενη διεύθυνση, αν υπάρχει, που είναι αποθηκευμένη στη διαδρομή του μηνύματος, διαφορετικά γυρίζει σε κατάσταση αναμονής. Ενώ στην περίπτωση που το πακέτο είναι στον σωστό προορισμό, ο κώδικας προχωράει στον έλεγχο ορθότητας του πακέτου χρησιμοποιώντας την μεταβλητή Checksum με τον τρόπο που περιγράφεται στην Παράγραφο 5.2.

Μόλις έχει ληφθεί ένα ορθό πακέτο καλείται η συνάρτηση "ViComm_OnVMsg", η οποία εκτελεί μια πιθανή εντολή και δίνει την τιμή σε συγκεκριμένες μεταβλητές, όπως το "RSSI". Στη συνέχεια η παραπάνω συνάρτηση καλεί με την σειρά της την συνάρτηση "SendReply", η οποία δημιουργεί το πακέτο απάντησης χρησιμοποιώντας τα δεδομένα από την διεπαφή της εφαρμογής και από το πακέτο λήψης, τα οποία είναι αναγκαία για την αποστολή του πακέτου.

Όταν γίνουν οι παραπάνω ενέργειες, στέλνεται το πακέτο απάντησης μέσω της σωστής διαδρομής, στην διεύθυνση από όπου είχε έρθει το πακέτο λήψης και τότε η εφαρμογή μεταβαίνει εκ νέου σε κατάσταση αναμονής. Στην κατάσταση αναμονής η εφαρμογή δεν εκτελεί κάποια ενέργεια αν δεν δώσει ο χρήστης κάποια εντολή ή αν δεν λάβει ένα πακέτο. Για το λόγο αυτό η εφαρμογή χρησιμοποιεί ελάχιστους πόρους από το σύστημα μας και σε συνδυασμό με την απλότητα των εντολών της, την καθιστούν μια αρκετά "ελαφριά" εφαρμογή.



Εικόνα 5.4 Διάγραμμα Ροής του Κώδικα της Εφαρμογής (Flowchart)

6 Αποτελέσματα

Οι στόχοι οι οποίοι είχαν τεθεί, από εμένα και τον επιβλέποντα καθηγητή, με την επιλογή του θέματος της πτυχιακής ήταν η ανάπτυξη μιας εφαρμογής, η οποία θα ήταν ικανή να εκτελεί εικονικά της λειτουργίες του προγραμματιζόμενου ελεγκτή “Netmaster RS485”. Επίσης ήταν αναγκαίο να εξομοιώνει την διαδικασία επικοινωνίας του ελεγκτή αυτού με τις εφαρμογές ελέγχου συσκευών ασύρματης τηλεμετρίας. Η εφαρμογή “Netmaster RS485 Emulator” θα βοηθήσει στην ανάπτυξη και βελτιστοποίηση τέτοιων εφαρμογών και στην μελέτη της λειτουργίας τους μέσω στατιστικών μετρήσεων.

Γι’ αυτό το λόγο κατά την περίοδο εγγραφής του κώδικα έγιναν πολλές δοκιμές επικοινωνίας με μια εφαρμογή ελέγχου συσκευών SCADA, η οποία έχει αναπτυχθεί από τον επιβλέποντα καθηγητή και χρησιμοποιείτε ήδη για τον έλεγχο συσκευής “Netmaster RS485”. Μέσω αυτής της εξομοίωσης ανακαλύφθηκαν σφάλματα (bugs) στον κώδικα των δύο εφαρμογών και έγιναν οι απαραίτητες διορθώσεις βελτιστοποίησης της λειτουργίας τους.

Μέσω της πτυχιακής εργασίας μου είχα την ευκαιρία να χρησιμοποιήσω της γνώσεις που απέκτησα, όσον αφορά τον προγραμματισμό στην γλώσσα Visual C, στα μαθήματα του κύκλου σπουδών της σχολής αλλά και να αποκτήσω ακόμα περισσότερες χρησιμοποιώντας αρκετά εργαλεία και αντικείμενα της εν λόγω γλώσσας. Επίσης απέκτησα αρκετές γνώσεις όσον αφορά την κωδικοποίηση και αποκωδικοποίηση πακέτων δεδομένων βάση του πρωτοκόλλου ASCII και την αποστολή τους μέσω σειριακής θύρας.

Τέλος ασχολήθηκα πρώτη φορά με το γνωστικό αντικείμενο των συστημάτων SCADA και τους τρόπους χρησιμοποίησης τους στην βιομηχανία και όχι μόνο. Οι παραπάνω γνώσεις που μου προσέφερε η μελέτη και εγγραφή της πτυχιακής εργασίας, θα είναι σημαντικό εφόδιο για την μετέπειτα σταδιοδρομία μου και ειδικά με πιθανή ενασχόληση μου με τα συστήματα SCADA.

6.1 Μελλοντική εργασία και Επεκτάσεις

Η εφαρμογή που αναπτύχθηκε για τις ανάγκες αυτής της πτυχιακής εργασίας μπορεί να χρησιμοποιηθεί, μετά από κάποιες αναγκαίες αναβαθμίσεις, για την μελέτη μεγάλου εύρος εφαρμογών ελέγχου συσκευών ασύρματης τηλεμετρίας. Κυρίως για εφαρμογές που χρησιμοποιούνται σε :

- Βιομηχανικές εγκαταστάσεις, όπως για την παρακολούθηση επιπέδων ενέργειας και αποθεμάτων πρώτων υλών.
- Συστήματα Ύδρευσης και Άρδευσης, όπως για την παρακολούθηση και έλεγχο επιπέδου δεξαμενών.
- Νοσοκομεία, όπως για τον έλεγχο του επιπέδου οξυγόνου και κενού στις φιάλες που χρησιμοποιούνται στα χειρουργεία.
- Απομακρυσμένες εγκαταστάσεις, για τον έλεγχο αποθεμάτων ενέργειας και αντιμετώπιση δυσλειτουργιών.

Η εφαρμογή “Netmaster RS485 Emulator” διευκολύνει προγραμματιστές που ασχολούνται με την ανάπτυξη εφαρμογών ελέγχου συσκευών SCADA, βοηθώντας τους να μελετάνε τις λειτουργίες και να παρατηρούν σφάλματα. Επίσης υπάρχει η δυνατότητα χρησιμοποίησης της από φοιτητές, οι οποίοι θα κληθούν να μελετήσουν ένα σύστημα SCADA ή να ασχοληθούν στα πλαίσια της πτυχιακής τους εργασίας με την ανάπτυξη μιας εφαρμογής ελέγχου συσκευών ασύρματης τηλεμετρίας.

Στην εφαρμογή αυτή μπορούν να γίνουν αναβαθμίσεις για πιθανών περισσότερα ή άλλου τύπου αντικείμενα, από αυτά που ήδη έχει, για την παρουσίαση επιπλέων μετρήσεων και καταστάσεων. Επίσης με αναβάθμιση μπορεί να αλλάξει η εντολή που εκτελείται βάση του κωδικού που λαμβάνεται με το πακέτο λήψης και έτσι να εμπλουτιστεί με διαφορετικές και περισσότερες λειτουργίες ο εξομοιωτής. Τέλος καθώς το κύριο μέρος είναι η επικοινωνία και η κωδικοποίηση/αποκωδικοποίηση των πακέτων, θα μπορούσε το αντικείμενο ActiveX “ViComm” να χρησιμοποιηθεί ξεχωριστά στην ανάπτυξη άλλων εφαρμογών που επικοινωνούν μέσω σειριακής θύρας.

Βιβλιογραφία

[1] King, Robert - Eric : Πληροφορικός έλεγχος. Εποπτικός έλεγχος και συλλογή πληροφοριών βιομηχανικών διεργασιών (SCADA), Α. Παπασωτηρίου & ΣΙΑ Ο.Ε., Αθήνα 1994

[2] Boyer, Stuart A: SCADA: Supervisory Control and Data Acquisition
Instrument Society of America, Research Triangle, N.C. 1993

[3] http://en.wikipedia.org/wiki/Telemetry

[4] http://en.wikipedia.org/wiki/SCADA

[5] http://en.wikipedia.org/wiki/Modbus

[6] http://en.wikipedia.org/wiki/Ascii

[7] http://en.wikipedia.org/wiki/Programmable_logic_controller

[8] http://en.wikipedia.org/wiki/Remote_Terminal_Unit

[9] http://www.elsist.it/WebSite/Html/English/Products/Hardware/PLC/Netsyst/EnNetmasterII.php

[10] Ευάγγελος Πετρούτσος: Πλήρες Εγχειρίδιο της Visual Basic 6
Εκδόσεις Μ. Γκιούρδας

[11] Halvorson M: MS Visual Basic 6.0 Professional
Κλειδάριθμος , 1998

[12] http://www.eterlogic.com

Παραρτήματα

Παράρτημα Α: Κώδικας

Παράρτημα Α1: Αντικείμενο ActiveX “ViComm”

```
Option Explicit
   Private Packet, ProceedPacket, CurDestAddr As String
   Private i, PackLen, RtLen, PackCntr, CurDest, ChkSum As Integer
   Public Route, Msg, ID As String
   Public RSSI, InputsLow, InputsHigh, Outputs, LowLimit, HighLimit, MsgLen As Integer
   Public Water, Battery, InstFlow, AccFlow, TempFlow As Double
   Public Event OnVMsg ()

Private Sub UserControl_Initialize ()
   i = 0
End Sub

Private Sub UserControl_Resize ()
   Size 770, 620
End Sub

Private Sub MSComm_OnComm ()
   On Error GoTo ErrHandler              'Σε περίπτωση λάθους η ροή του προγράμματος
                                          μεταβαίνει στον ErrorHandler
   While MSComm.InBufferCount <> 0     Όσο η σειριακή θύρα δεν είναι άδεια
      Dim ReadChar As String
       ReadChar = MSComm.Input       'Τίνεται αποθήκευση των δεδομένων από την σειριακή
                                          θύρα στην μεταβλητή ReadChar τύπου string ΚΑΤΑ
                                          1 χαρακτήρα κάθε φορά όπως έχει οριστεί στις
                                          ιδιότητες του MSComm (InputLen=1)
       Call BuffHandler (ReadChar)    'Καλείτε η συνάρτηση BuffHandler με όρισμα τον
                                          χαρακτήρα που διαβάστηκε από τη σειριακή
   Wend
   Exit Sub
ErrHandler:
   If Err.Number <> 0 Then             'Εάν ο κωδικός του λάθους δεν είναι 0(μηδέν)
      Dim FileNum As Integer
      Debug.Print "MSComm_OnComm" & Err.Number & Err.Description & FileNum; ""
                                          'ορίζουμε μια μεταβλητή τύπου ακεραίου αριθμού
      FileNum = FreeFile                  'η οποία θα πάρει τιμή από την συνάρτηση FreeFile,
                                          η οποία δίνει τον επόμενο διαθέσιμο αριθμό αρχείου
     Open App.Path & "\Errors.log" For Append As FileNum 'ανοίγουμε το αρχείο Errors.log
                                                          στον φάκελο της εφαρμογής
'και αποθηκεύουμε μέσα τον κωδικό και την περιγραφή του λάθους επίσης την συνάρτηση και την
ώρα-ημερομηνία που δημιουργήθηκε
      Write #FileNum, Err.Number, Err.Description, "MSComm_OnComm", Now ()
      Close FileNum                              'κλείνει το αρχείο
      Err.Clear                                  'μηδενίζει τον κωδικό του λάθους
       Resume Next              'μεταβαίνει την ροή του προγράμματος στην επόμενη εντολή
   End If
End Sub
```

Συνάρτηση Διαχείρισης Αποθηκευμένου Πακέτου από την Σειριακή Θύρα

```
Private Sub BuffHandler (ByVal Buff As String)

    Debug.Print "[" & Hex (Asc (Buff)) & "]" & "i=" & i   'Τυπώνει στο "Immediate Window"
                                                            την Δεκαεξαδική τιμή του χαρακτήρα

    Select Case i                'Ο χαρακτήρας διαχειρίζεται ανάλογα την τιμή της μεταβλητής "i"
      Case 0                       'Στην περίπτωση που το i είναι 0
        If Buff = "*" Then   'και ο χαρακτήρας είναι το σύμβολο "αστεράκι",
                                    έχουμε πιθανό μήνυμα
          Packet = Buff       'τότε αποθηκεύουμε τον χαρακτήρα στη συμβολοσειρά
                                    Packet ως πρώτο (έτσι μηδενίζεται ταυτόχρονα η
                                    συμβολοσειρά)
          i = i + 1             'επίσης αυξάνουμε το i κατά 1.
        End If

      Case 1                       'Αν το i=1
        If Buff = "*" Then   'και ο χαρακτήρα είναι "αστεράκι" τότε έχουμε
                                    την αρχή μηνύματος
          Packet = Packet & Buff      'αποθηκεύουμε τον χαρακτήρα
          i = i + 1               'και αυξάνουμε το i κατά 1.
          Else                'Αν το i=1 αλλά δεν είναι ο χαρακτήρας αστεράκι τότε
                                    δεν έχουμε έγκυρο μήνυμα,
            i = 0              'κάνουμε το i πάλι 0
            Packet = ""            'και "αδειάζουμε" την συμβολοσειρά Packet ώστε
                                        να περιμένουμε πάλι
        End If                    'χαρακτήρα "αστεράκι" για πιθανό μήνυμα.

      Case 2                       'Εφόσον το i είναι ίσο με 2, έχουμε ήδη τσεκάρει ότι
                                        έχουμε την αρχή μηνύματος,
        Packet = Packet & Buff'αποθηκεύουμε τον χαρακτήρα στο Packet ο οποίος είναι ο 3ος

        PackLen = Asc (Buff)          'άρα η αριθμητική τιμή του είναι το μήκος του
                                    μηνύματος και την αποθηκεύουμε στην ανάλογη μεταβλητή
        i = i + 1                         'και αυξάνουμε το i κατά 1.

      Case Is > 2                        'Για i μεγαλύτερο της τιμής 2
        Packet = Packet & Buff         'αποθηκεύουμε τον χαρακτήρα στο packet
        i = i + 1                         'και αυξάνουμε το i κατά 1,

        If i = PackLen Then               'όταν φτάσουμε σε αριθμό χαρακτήρων ίσο
                                            με το μήκος του πακέτου
          i = 0    'μηδενίζουμε το i και ΑΠΟΘΗΚΕΥΟΥΜΕ ΤΗΣ ΑΡΙΘΜΗΤΙΚΕΣ ΤΙΜΕΣ
                ΤΩΝ ΧΑΡΑΚΤΗΡΩΝ ΣΤΙΣ ΚΑΤΑΛΛΗΛΕΣ ΜΕΤΑΒΛΗΤΕΣ.
          RtLen = Asc (Mid (Packet, 4, 1))         'Στην 4η θέση είναι το μήκος της διαδρομής,
          Route = Mid (Packet, 6, RtLen)       'Αποθήκευση διαδρομής από το πακέτο
          CurDest = Asc (Mid (Packet, 5, 1))     'στην 5η θέση είναι ο αριθμός που δείχνει τον
                                                    τωρινό προορισμό του πακέτου μέσα από την
                                                    διαδρομή.
          PackCntr = Asc (Mid (Packet, RtLen + 6, 1))    'Ο αριθμός στην 6 συν το μήκος της
                                                            διαδρομής θέση είναι ο αύξων
                                                            αριθμός του πακέτου(δείχνει πόσα
                                                            πακέτα έχουμε μεταφερθεί)
```

```
        RSSI = Asc (Mid (Packet, RtLen + 7, 1)) ‘Ο αριθμός στην 7 συν το μήκος της
        Διαδρομής θέση είναι ο αριθμός RSSI, δηλαδή η ισχύς του σήματος κατά την λήψη
        του πακέτου (Received signal strength indication).

        MsgLen = PackLen - RtLen - 10   'Το μήκος του μηνύματος είναι ίσο με το μήκος
                                        του πακέτου μείον το μήκος της διαδρομής μείον 10

        Msg = Mid (Packet, RtLen + 10, MsgLen)       'Αποθήκευση του μηνύματος
                                                      από το πακέτο

      'Έχοντας σαν δεδομένα την διαδρομή και τον αριθμό του τωρινού προορισμού
      'βρίσκουμε την διεύθυνση του τωρινού προορισμού που είναι αποθηκευμένη σε δύο
      'χαρακτήρες και την αποθηκεύουμε στην συμβολοσειρά "CurDestAddr"
    CurDestAddr = Format (Hex (Asc (Mid (Route, 2 * CurDest - 1, 1))), "00")
    CurDestAddr = CurDestAddr & Format (Hex (Asc (Mid (Route, 2 * CurDest, 1))), "00")

        If CurDestAddr = ID Then    'Ελέγχουμε εάν η διεύθυνση του τωρινού προορισμού
                                     είναι η ίδια με την διεύθυνση της εφαρμογής μας
                                  'εάν NAI
          If CurDest = RtLen / 2 Then 'Ελέγχουμε εάν είναι η τελευταία διεύθυνση
                                       στην διαδρομή
            Call CheckSum 'εάν Ναι καλούμε την συνάρτηση "CheckSum" για να
                          υπολογίσουμε τον αριθμό ελέγχου ορθότητας του μηνύματος
              If ChkSum = Asc (Mid (Packet, PackLen, 1)) Then   'και τον συγκρίνουμε με
                                     τον αποθηκευμένο αριθμό ελέγχου, έτσι βλέπουμε αν
                          το μήνυμα είναι σωστό και δεν είχαμε λάθη λόγω διάδοσης.
                RaiseEvent OnVMsg 'Εάν είναι σωστό το μήνυμα εγείρεται το
                                                       συμβάν "OnVMsg"
              End If

          Else                      'Εάν δεν είναι η τελευταία διεύθυνση στην διαδρομή τότε
            ProceedPacket = ""          'Αρχικοποιούμε την συμβολοσειρά που θα
                                        χρησιμοποιήσουμε για την προώθηση του μηνύματος
            Do Until i = PackLen            'Μέχρι το i να γίνει ίσο με το μήκος του πακέτου
                                  'αντιγράφουμε το αρχικό πακέτο στο πακέτο για προώθηση
              If i = 5 Then                 'εκτός του αριθμού τωρινού προορισμού που
                                             βρίσκεται στη θέση 5 του πακέτου
                ProceedPacket = ProceedPacket & Chr (CurDest + 1)    'τον αριθμό αυτό
                          τον αυξάνουμε κατά 1 για να σταλεί στον επόμενο προορισμό
                Else
                ProceedPacket = ProceedPacket & Mid (Packet, i, 1)
              End If
              i = i + 1                  ' αυξάνουμε το i κατά 1
            Loop
            MSComm.Output = ProceedPacket 'Αποστολή του Πακέτου
                                           Προώθησης στον Επόμενο Προορισμό
          End If

        End If
      End If
  End Select
End Sub
```

'Συνάρτηση Δημιουργίας Και Αποστολής Πακέτου Απάντησης

```
Public Sub SendReply ()
  Dim AccFlowPin (4), InstFlowPin (4) As Double

  Packet = ""
  MsgLen = 14
  PackLen = 9 + RtLen + MsgLen
  RSSI = Int ((255 * Rnd) + 0)        'Τυχαία επιλογή της τιμής του "RSSI"
                          ' ΘΕΣΗ ΣΥΜΒΟΛΟΣΕΙΡΑΣ
  Packet = Chr (42)                   'Packet 1
  Packet = Packet & Chr (42)          'Packet 2
  Packet = Packet & Chr (PackLen)     'Packet 3
  Packet = Packet & Chr (RtLen)       'Packet 4
  Packet = Packet & Chr (1)           'Packet 5

'Αποθηκεύουμε την διαδρομή στην συμβολοσειρά αποστολής ανεστραμμένη
  i = RtLen
  Do Until i = 0
    Packet = Packet & Mid (Route, i - 1, 1)     'Κάθε διεύθυνση αποτελείτε από 2
                                                 χαρακτήρες
    Packet = Packet & Mid (Route, i, 1)
    i = i - 2
  Loop

  'ΑΠΟΘΗΚΕΥΟΥΜΕ ΤΙΣ ΜΕΤΑΒΛΗΤΕΣ ΜΕ ΤΙΣ ΠΛΗΡΟΦΟΡΙΕΣ ΤΗΣ ΕΦΑΡΜΟΓΗΣ
      ΣΤΗΝ ΚΑΤΑΛΛΗΛΗ ΘΕΣΗ ΤΗΣ ΣΥΜΒΟΛΟΣΕΙΡΑΣ
                                        ‘ΘΕΣΗ ΣΥΜΒΟΛΟΣΕΙΡΑΣ
  Packet = Packet & Chr (PackCntr + 1)    'Packet 6+RtLen
  Packet = Packet & Chr (RSSI)            'Packet 7+RtLen
  Packet = Packet & Chr (255)             'Packet 8+RtLen
                          ‘ΘΕΣΗ ΜΗΝΥΜΑΤΟΣ /ΘΕΣΗ ΣΥΜΒΟΛΟΣΕΙΡΑΣ
  Packet = Packet & Chr (255)             ‘Msg 1   /Packet 9+Rtlen
  Packet = Packet & Chr (InputsLow)       ‘Msg 2   /Packet 10+Rtlen
  Packet = Packet & Chr (Outputs)         ‘Msg 3   /Packet 11+Rtlen
  Packet = Packet & Chr (Battery)         ‘Msg 4   /Packet 12+Rtlen
  Packet = Packet & Chr (Water)           ‘Msg 5   /Packet 13+Rtlen

  'Χωρίζουμε την τιμή του Accumulate Flow σε πίνακα 4 μεταβλητών
                                  ΘΕΣΗ ΜΗΝΥΜΑΤΟΣ/ΘΕΣΗ ΣΥΜΒΟΛΟΣΕΙΡΑΣ
  AccFlowPin (3) = 70 Mod 256             'Msg9   / Packet 17+Rtlen
  TempFlow = (AccFlow - AccFlowPin (3)) / 256
  AccFlowPin (2) = TempFlow Mod 256       'Msg8   / Packet 16+Rtlen
  TempFlow = ((TempFlow - AccFlowPin (2)) / 256)
  AccFlowPin (1) = TempFlow Mod 256       'Msg7   / Packet 15+Rtlen
  TempFlow = ((TempFlow - AccFlowPin (1)) / 256)
  AccFlowPin (0) = TempFlow Mod 256       'Msg6   / Packet 14+Rtlen

  i = 0
  Do Until i = 4
  Packet = Packet & Chr (AccFlowPin (i))   'Αποθηκεύουμε τις 4 μεταβλητές του
                                  Accomulate Flow στις κατάλληλες θέσης της συμβολοσειράς
    i = i + 1
  Loop
```

```
'Χωρίζουμε την τιμή του Instant Flow σε πίνακα 4 μεταβλητών//
                ΘΕΣΗ ΜΗΝΥΜΑΤΟΣ/ΘΕΣΗ ΣΥΜΒΟΛΟΣΕΙΡΑΣ
InstFlowPin (3) = InstFlow Mod 256                    '   Msg13   / Packet 21+Rtlen
TempFlow = (InstFlow - InstFlowPin (3)) / 256
InstFlowPin (2) = TempFlow Mod 256                    '   Msg12   / Packet 20+Rtlen
TempFlow = ((TempFlow - InstFlowPin (2)) / 256)
InstFlowPin (1) = TempFlow Mod 256                    '   Msg11   / Packet 19+Rtlen
TempFlow = ((TempFlow - InstFlowPin (1)) / 256)
InstFlowPin (0) = TempFlow Mod 256                    '   Msg10    / Packet 18+Rtlen

i = 0
Do Until i = 4
Packet = Packet & Chr (InstFlowPin (i)) 'Αποθηκεύουμε τις 4 μεταβλητές του Instant Flow
                                          στις κατάλληλες θέσεις της συμβολοσειράς
  i = i + 1
Loop

Packet = Packet & Chr (InputsHigh)        'Αποθηκεύουμε την μεταβλητή "Inputs High"
                                          (που είναι η 14η θέση του μηνύματος) στην
                                          22+Rtlen θέση της συμβολοσειράς

Call CheckSum                  'καλούμε την συνάρτηση "CheckSum" για να υπολογίσουμε
                               τον αριθμό ελέγχου ορθότητας του μηνύματος
Packet = Packet & Chr (ChkSum)            'και το αποθηκεύουμε στην τελευταία θέση της
                                          συμβολοσειράς του πακέτου

MSComm.Output = Packet        'στέλνουμε την συμβολοσειρά, όπου είναι αποθηκευμένο
                              το πακέτο απάντησης, από την σειριακή θύρα
i = 0                         'και τέλος μηδενίζουμε το i.
End Sub
```

'Συνάρτηση Υπολογισμού Αθροίσματος Ελέγχου

```
Private Sub CheckSum ()
  ChkSum = 0              'Μηδενίζουμε το άθροισμα ελέγχου
  i = 1                   'Ξεκινάμε από τον πρώτο χαρακτήρα του πακέτου

  Do Until i = PackLen    'και μέχρι το i να γίνει ίσο με το μήκος του πακέτου
    ChkSum = ChkSum Xor Asc (Mid (Packet, i, 1))   'προσθέτουμε με XOR όλες τις
                                                    αριθμητικές τιμές των χαρακτήρων του πακέτου
    i = i + 1                                       'αυξάνουμε το i κατά 1
  Loop
  i = 0                                             'μηδενίζουμε ξανά το i για αποφυγή σφαλμάτων
End Sub
```

'Ιδιότητες

```
Public Property Let CommPort (ByVal Val As Integer)
  MSComm.CommPort = Val
  PropertyChanged "CommPort"
End Property

Public Property Get CommPort () As Integer
  CommPort = MSComm.CommPort
End Property
```

```
Public Property Let RThreshold (ByVal Val As Integer)
   MSComm.RThreshold = Val
   PropertyChanged "RThreshold"
End Property

Public Property Let SThreshold (ByVal Val As Integer)
   MSComm.SThreshold = Val
   PropertyChanged "SThreshold"
End Property

Public Property Let Handshaking (ByVal Val As Integer)
   MSComm.Handshaking = Val
   PropertyChanged "Handshaking"
End Property

Public Property Let PortOpen(ByVal Val As Boolean)
   On Error GoTo ErrHandler
   MSComm.PortOpen = Val
   PropertyChanged "PortOpen"
   Exit Property

ErrHandler:
   Msgbox ("NO VALID COMM PORT" & vbCrLf & "CHOOSE COMM PORT")
   If Err.Number <> 0 Then
      Dim FileNum As Integer
      FileNum = FreeFile
      Open App.Path & "\Errors.log" For Append As FileNum
      Write #FileNum, Err.Number, Err.Description, "MSComm_OnComm", Now()
      Close FileNum
      Err. Clear
   End If
End Property

Public Property Get PortOpen () As Boolean
  PortOpen = MSComm.PortOpen
End Property

Public Property Let DTREnable (ByVal Val As Boolean)
   MSComm.DTREnable = Val
   PropertyChanged "DTREnable"
End Property

Public Property Let EOFEnable (ByVal Val As Boolean)
   MSComm.EOFEnable = Val
   PropertyChanged "EOFEnable"
End Property

Public Property Let RTSEnable (ByVal Val As Boolean)
   MSComm.RTSEnable = Val
   PropertyChanged "RTSEnable"
End Property
```

Παράρτημα Α2: Κεντρική Φόρμα “ViEmulator”

```
Option Explicit
Private i, LocalAuto As Integer
Private strTmp As String
Public IniFName As String
Private modemData As Double

Private Sub Form_Load ()

 'Παίρνουμε τις τιμές των αποθηκευμένων μεταβλητών από το αρχείο παραμέτρων
   IniFName = App.Path & "\NetmasterRS485Emulator.ini"

   ViComm.InputsLow = Val (GetFromINI ("INPUTS/OUTPUTS", "Digital Inputs 0 to 7", "0", IniFName))

   ViComm.InputsHigh = Val (GetFromINI ("INPUTS/OUTPUTS", "Digital Inputs 8 to 11", "0", IniFName))

   ViComm.Outputs = Val (GetFromINI ("INPUTS/OUTPUTS", "Digital Outputs", "0", IniFName))

   ViComm.Water = Val (GetFromINI ("Water Level", "Water Level", "0", IniFName))
   ViComm.LowLimit = Val (GetFromINI ("Water Level", "Low Limit", "0", IniFName))
   ViComm.HighLimit = Val (GetFromINI ("Water Level", "High Limit", "128", IniFName))
   ViComm.Battery = Val (GetFromINI ("Sliders", "Battery Voltage", "0", IniFName))
   ViComm.AccFlow = Val (GetFromINI ("Sliders", "Accomulate Flow", "0", IniFName))
   ViComm.InstFlow = Val (GetFromINI ("Sliders", "Instant Flow", "0", IniFName))
   sliderWater.Value = ViComm.Water
   sliderBattery.Value = ViComm.Battery
   sliderAccFlow.Value = ViComm.AccFlow
   sliderInstFlow.Value = ViComm.InstFlow

   NumToLed 'Καλούμε την συνάρτηση η οποία ανάβει τα LED σύμφωνα με τις τιμές των
                 αντίστοιχων μεταβλητών

   With ViComm
      .CommPort = Val (GetFromINI ("General", "Port", "2", IniFName))
      .RThreshold = 1
      .SThreshold = 1
       .Handshaking = 0 '0=comNone, 1=comXOnXOff, 2=comRTS, 3=comRTSXOnXOff
      .ID = GetFromINI ("General", "ID", "BBBB", IniFName)
   End With

   txtID.Text = ViComm.ID ‘Εμφανίζουμε το ID και αυτόματα γίνεται REFRESH
End Sub

Private Sub Form_Unload (Cancel As Integer)
   If ViComm.PortOpen = True Then
      ViComm.PortOpen = False
   End If

  'Αποθηκεύουμε τις μεταβλητές της εφαρμογής στο αρχείο παραμέτρων
   Call LedToNum
```

```
        AddToINI "General", "ID", ViComm.ID, IniFName
        AddToINI "General", "Port", ViComm.CommPort, IniFName
        AddToINI "INPUTS/OUTPUTS", "Digital Inputs 0 to 7", Str (ViComm.InputsLow), IniFName
        AddToINI "INPUTS/OUTPUTS", "Digital Inputs 8 to 11", Str (ViComm.InputsHigh), IniFName
        AddToINI "INPUTS/OUTPUTS", "Digital Outputs", Str (ViComm.Outputs), IniFName
        AddToINI "Water Level", "Low Limit", Str (ViComm.LowLimit), IniFName
        AddToINI "Water Level", "High Limit", Str (ViComm.HighLimit), IniFName
        AddToINI "Water Level", "Water Level", Str (ViComm.Water), IniFName
        AddToINI "Sliders", "Battery Voltage", Str (ViComm.Battery), IniFName
        AddToINI "Sliders", "Accomulate Flow", Str (ViComm.AccFlow), IniFName
        AddToINI "Sliders", "Instant Flow", Str (ViComm.InstFlow), IniFName
    End Sub

    Private Sub menuCommPort_Click ()
        frmSetCommPort.Show vbModal, Me
    End Sub

    Private Sub menuExit_Click ()
        Unload Me
    End Sub
```

Συνάρτηση "ViComm_OnVMsg"

```
    'Εκτελείτε όταν εγερθεί το γεγονός "OnVMsg", δηλαδή όταν έχει ληφθεί έγκυρο μήνυμα.
    Private Sub ViComm_OnVMsg ()
        Dim i As Integer
        Dim j As Double
        BarRSSI.Value = ViComm.RSSI
        BarRSSI.ToolTipText = "RSSI:" & Format (BarRSSI.Value, "00.0") & "(" & Format (100 * BarRSSI.Value / 255, "00") & "%)"

        'Εκτέλεση της εντολής του εισερχόμενου μηνύματος βάση τον κωδικό της (0 έως 8)
        Select Case Asc (Mid (ViComm.Msg, 1, 1))
            Case 0                      'Κωδικός 0: Σβήσιμο Συγκεκριμένου LED
                LedOut (Asc (Mid (ViComm.Msg, 2, 1)) - 1).TurnOff
                lcdStatus.Caption = "Command: LED OFF"
            Case 1                      'Κωδικός 1: Άνοιγμα Συγκεκριμένου LED
                LedOut (Asc (Mid (ViComm.Msg, 2, 1)) - 1).TurnOn
                lcdStatus.Caption = "Command: LED ON"
            Case 2                      ' Κωδικός 2: Δεν κάνει τίποτα, έτσι επιστρέφεται ένα
                                            μήνυμα με την κατάσταση του NetSystem
                lcdStatus.Caption = "Command: STATUS"
            Case 3                      'Κωδικός 3: Αλλάζει το LOW LIMIT
                ViComm.LowLimit = 256 * Asc (Mid (ViComm.Msg, 2, 1)) + Asc (Mid (ViComm.Msg, 3, 1))
                lcdStatus.Caption = "Command: SET Low Limit"
            Case 4                      "Κωδικός 4: Αλλάζει το HIGH LIMIT
                ViComm.HighLimit = 256 * Asc (Mid (ViComm.Msg, 2, 1)) + Asc (Mid (ViComm.Msg, 3, 1))
                lcdStatus.Caption = "Command: SET High Limit"
            Case 5                      "Κωδικός 5: Αλλάζει την Ημερομηνία και την Ώρα
                lcdStatus.Caption = "Command: SET RTC"
            Case 6                      'Κωδικός 6: Αλλάζει το ID
```

```
            lcdStatus.Caption = "Command: CHANGE ID"
            If Hex (Asc (Mid (ViComm.Msg, 2, 1))) = 0 Then
               ViComm.ID = "00"
               Else
                  ViComm.ID = Hex (Asc (Mid (ViComm.Msg, 2, 1)))
            End If

            If Hex (Asc (Mid (ViComm.Msg, 3, 1))) = 0 Then
               ViComm.ID = ViComm.ID & "00"
               Else
                  ViComm.ID = ViComm.ID & Hex (Asc (Mid (ViComm.Msg, 3, 1)))
            End If
            txtID = ViComm.ID
         Case 7                          'Κωδικός 7: Στέλνει δεδομένα στο MODEM
            i = 2
            modemData = 0
            Do Until i = ViComm.MsgLen + 1
               modemData = modemData * 100 + Hex (Asc (Mid (ViComm.Msg, i, 1)))
               i = i + 1
            Loop
            lcdStatus.Caption = "Send to modem:" & modemData
         Case 8                          'Κωδικός 8: Ορίζει το LOCAL AUTOMATION
            LocalAuto = Asc (Mid (ViComm.Msg, 2, 1))
            lcdStatus.Caption = "Local Automation:" & LocalAuto
      End Select

      Call refreshMe
      Call LedToNum           'Αποθηκεύουμε τις καταστάσεις των LED μέσω της
                                 συνάρτησης LedToNum
      Call ViComm.SendReply   'Προχωράμε στην δημιουργία και αποστολή του μηνύματος
                                 Απάντησης μέσω του ViComm.
   End Sub
```

'Συνάρτηση Ενεργοποίησης / Απενεργοποίησης παραλαβής μηνυμάτων ανοίγοντας ή κλείνοντας την πόρτα αντίστοιχα

```
   Private Sub menuReceiveEnabled_Click ()
      menuReceiveEnabled.Checked = Not menuReceiveEnabled.Checked
      Select Case menuReceiveEnabled.Checked
         Case False
            ViComm.PortOpen = False
         Case True
            ViComm.PortOpen = True
      End Select
   End Sub
```

'Συνάρτηση ανανέωσης διεπαφής

```
   Private Sub refreshMe ()
      'Ανανέωση μεταβλητής, μπάρας, ετικέτας και οθόνης επιπέδου νερού
      ViComm.Water = sliderWater.Value
      sliderWater.ToolTipText = "Water Level: " & ViComm.Water & «deciVolt"
      txtWL = ViComm.Water / 10
      ViComm.Water = ViComm.Water * 40.95
      lcdAn1.Caption = "Analogue Input 1" & vbCrLf & "Hex:  " & Hex (ViComm.Water) & "
Dec: " & ViComm.Water
```

```
        ViComm.Water = CByte ((100 * (ViComm.Water - ViComm.LowLimit) /
(ViComm.HighLimit - ViComm.LowLimit)) Mod 255)
        lcdWL.Caption = "Water Level: " & ViComm.Water & «%"

        'Ανανέωση μεταβλητής, μπάρας, ετικέτας και οθόνης βολτ μπαταρίας
        ViComm.Battery = sliderBattery.Value * 2
        sliderBattery.ToolTipText = "Battery Level: " & ViComm.Battery & «deciVolt"
        txtBV = ViComm.Battery / 10
        lcdBat.Caption = "Battery Voltage:" & ViComm.Battery / 10 & “Volt”

        'Ανανέωση μεταβλητής, μπάρας και οθόνης accomulate flow
        ViComm.AccFlow = sliderAccFlow.Value
        sliderAccFlow.ToolTipText = "Accomulate Flow: " & ViComm.AccFlow & "deciVolt"
        txtAF = ViComm.AccFlow / 10

        'Ανανέωση μεταβλητής, μπάρας και οθόνης instant flow
        ViComm.InstFlow = sliderInstFlow.Value
        sliderInstFlow.ToolTipText = "Instant Flow: " & ViComm.InstFlow & "deciVolt"
        txtIF = ViComm.InstFlow / 10

        'Ανανέωση του πλαισίου κειμένου του ID
        ViComm.ID = txtID.Text
    End Sub
```

'Συναρτήσεις Μετατροπής Κατάστασης LED σε Αριθμό και Αντίστροφα

```
    'Με τις καταστάσεις (0 ή 1) του κάθε LED, δημιουργούμε έναν ακέραιο δεκαδικό αριθμό
    'για κάθε ομάδα LED με βάση το δυαδικό σύστημα, δηλαδή κάθε αριθμός 0 ή 1 αντιστοιχεί
    'σε ψηφίο ενός υποτιθέμενου δυαδικού αριθμού που τον μετατρέπουμε σε δεκαδικό
    Private Sub LedToNum ()
      ViComm.InputsLow = 0
      For i = 0 To 7
        ViComm.InputsLow = ViComm.InputsLow + (LedIn (i).ledon * -1) * (2 ^ i)
      Next i

      ViComm.InputsHigh = 0
      For i = 8 To 11
        ViComm.InputsHigh = ViComm.InputsHigh + (LedIn (i).ledon * -1) * (2 ^ (i - 8))
      Next i

      ViComm.Outputs = 0
      For i = 0 To 7
        ViComm.Outputs = ViComm.Outputs + (LedOut (i).ledon * -1) * (2 ^ i)
      Next i

    End Sub

    'Μετατρέπουμε τον ακέραιο δεκαδικό αριθμό, που αντιστοιχεί σε κάθε ομάδα από LED,
    'σε δυαδικό με την βοήθεια της συνάρτησης "DecToBin" και με τα ψηφία του
    'να είναι η κατάσταση των LED (0 ή 1), έτσι στη συνέχεια ανάβουν τα LED
    'που τους αντιστοιχεί τιμή ίση με 1.
    Private Sub NumToLed ()
      strTmp = DecToBin (Int (ViComm.InputsLow), 8)
      For i = 0 To 7
        If Val (Mid (strTmp, 8 - i, 1)) = 1 Then
```

```
        LedIn (i).TurnOn
      End If
    Next i

    strTmp = DecToBin (Int (ViComm.InputsHigh), 4)
    For i = 0 To 3
      If Val (Mid (strTmp, 4 - i, 1)) = 1 Then
        LedIn (i + 8).TurnOn
      End If
    Next i

    strTmp = DecToBin (Int (ViComm.Outputs), 8)
    For i = 0 To 7
      If Val (Mid (strTmp, 8 - i, 1)) = 1 Then
        LedOut (i).TurnOn
      End If
    Next i
End Sub

'Μετατρέπει έναν ακέραιο δεκαδικό αριθμό στον ισοδύναμο δυαδικό με αριθμό ψηφίων ίσο
'με τον αριθμό "Bits" και τα αποθηκεύει σε συμβολοσειρά
Private Function DecToBin (ByVal Dec As Integer, ByVal Bits As Integer) As String
    Dim i As Integer
    DecToBin = vbNullString         'Μηδενισμός της συμβολοσειράς DecToBin
    For i = 0 To (Bits - 1)     'Εκτελούμε τόσες φορές όσα και τα bit που θέλουμε να παράγουμε
        DecToBin = CStr ((Dec And 2 ^ i) / 2 ^ i) & DecToBin     'Προσθέτει κάθε φορά στη
                στη συμβολοσειρά τον χαρακτήρα 0 ή 1 ανάλογα το αποτέλεσμα της πράξης
    Next i
End Function
```

'Συναρτήσεις οι οποίες όταν προβούμε σε μια ενέργεια στην διεπαφή καλούν την συνάρτηση που πρέπει να εκτελεστεί

```
Private Sub cmdOff_Click ()
    LedOut (0).TurnOff
End Sub
Private Sub cmdOn_Click ()
    LedOut (0).TurnOn
End Sub
Private Sub txtID_Change ()
    refreshMe
End Sub
Private Sub sliderBattery_Scroll ()
    refreshMe
End Sub
Private Sub sliderWater_Scroll ()
    refreshMe
End Sub
Private Sub sliderAccFlow_Scroll ()
    refreshMe
End Sub
Private Sub sliderInstFlow_Scroll ()
    refreshMe
End Sub
```

Παράρτημα Α3: Φόρμα Παράθυρου Επιλογής Σειριακής Θύρας "frmSetCommPort"

```
Option Explicit
Dim i As Byte
Dim portOpen As Boolean

Private Sub Form_Load ()
   GetAvailableCommports   'Καλούμε την συνάρτηση που βρίσκει τις διαθέσιμες πόρτες

   For i = 1 To 8          'Επίσης βρίσκουμε ποιά πόρτα χρησιμοποιείτε από την εφαρμογή
         If i = ViEmulator.ViComm.CommPort And Me.optUsedComPort(i).Enabled = True
         Then
         Me.optUsedComPort(i).Value = True 'και "τσεκάρουμε" το αντίστοιχο Option Button
      End If
   Next i
End Sub

'Συνάρτηση Εύρεσης Διαθέσιμων Σειριακών Πορτών
Sub GetAvailableCommports ()
   On Error GoTo errorHandler      'Όταν προκύψει λάθος μεταφερόμαστε στις εντολές
                                          του "ErrorHandler".

   portOpen = False
   If ViEmulator.ViComm.portOpen = True Then 'Αν η Πόρτα Επικοινωνίας είναι ανοιχτή
      portOpen = ViEmulator.ViComm.portOpen 'αποθηκεύουμε την κατάσταση της σε μια
μεταβλητή και
       ViEmulator.ViComm.portOpen = False  'κλείνουμε την Πόρτα Επικοινωνίας της
       εφαρμογής για να "δουλέψει" σωστά η παρακάτω ρουτίνα.
   End If

   For i = 1 To 8                  'Για όλες από τις 8 σειριακές πόρτες επικοινωνίας,
      Me.mscTestAvPort.CommPort = i       'δοκιμάζουμε μέσω του mscTestAvPort
      Me.mscTestAvPort.PortOpen = True    'να τις τροποποιήσουμε και αν προκύψει σφάλμα
      Me.mscTestAvPort.PortOpen = False   'μεταφερόμαστε στις παρακάτω εντολές του
   Next i

   Exit Sub

errorHandler:                          'errorHandler ο οποίος για την πόρτα που προέκυψε το
                                          λάθος και άρα δεν είναι διαθέσιμη,
   Me.optUsedComPort (i).Enabled = False    'αφαιρεί την δυνατότητα επιλογής
                                                του αντίστοιχου Option Button
   Resume Next                     'και συνεχίζει την ρουτίνα.
End Sub

Private Sub cmdOK_Click ()
   For i = 1 To 8      'Βρίσκουμε ποιά πόρτα επιλέχτηκε από το OptionButton και
                           επιλέγουμε αυτήν την πόρτα για την εφαρμογή
      If Me.optUsedComPort (i).Value = True Then ViEmulator.ViComm.CommPort = i
   Next i
   Unload Me
End Sub
```

```
Private Sub cmdCancel_Click()
   Unload Me
End Sub

Private Sub Form_Unload(Cancel As Integer)

'Αν η Πόρτα Επικοινωνίας ήταν ανοιχτή την ανοίγουμε ξανά
   If portOpen = True Then
      ViEmulator.ViComm.portOpen = True
   End If

'Ελέγχουμε αν η θύρα που είναι επιλεγμένη είναι διαθέσιμη ώστε να μην έχουμε λάθος
   Select Case Me.optUsedComPort (ViEmulator.ViComm.CommPort).Enabled

      Case True ‘Αν ναι ενεργοποιούμε το κουμπί με το οποίο ανοίγουμε την θύρα
         ViEmulator.menuReceiveEnabled.Enabled = True

      Case False ‘Αν όχι απενεργοποιούμε και ξετσεκάρουμε το κουμπί
         ViEmulator.menuReceiveEnabled.Enabled = False
         ViEmulator.menuReceiveEnabled.Checked = False
   End Select

End Sub
```

Παράρτημα Α4: Αντικείμενο ActiveX “Led”

```
Option Explicit

Private Sub UserControl_Click ()
   If LedOn = False Then   ‘αν το LED είναι σβηστό
      TurnOn            'το ανάβει
    Else                'αν είναι αναμμένο
      TurnOff           'το σβήνει
   End If
End Sub

'Ιδιότητα για εντολή ελέγχου για το LED
Property Let LedOn (ByVal newVal As Boolean)
   If newVal = True Then      'Εάν η εντολή είναι να ανάψει το LED
      TurnOn               'τότε το ανάβει
   Else                    'αλλιώς αν η εντολή είναι να σβήσει το LED
      TurnOff              'το σβήνει
   End If
End Property

'Ιδιότητα για τον έλεγχο σε τι κατάσταση βρίσκεται το LED
Property Get LedOn () As Boolean
   If shpLed.FillColor = &HFF& Then       'Αν το χρώμα είναι κόκκινο τότε
      LedOn = True                  'το LED είναι αναμμένο
   Else                             'αλλιώς
      LedOn = False                 'είναι σβηστό.
   End If
End Property
```

```
'Συνάρτηση η οποία ανάβει το LED
Public Sub TurnOn ()
      shpLed.FillColor = &HFF&    'Κόκκινο χρώμα = αναμμένο
End Sub

'Συνάρτηση η οποία σβήνει το LED
Public Sub TurnOff ()
      shpLed.FillColor = &H0&     'Μαύρο χρώμα = σβηστό
End Sub

Private Sub UserControl_Resize ()
   Size 255, 165
End Sub
```

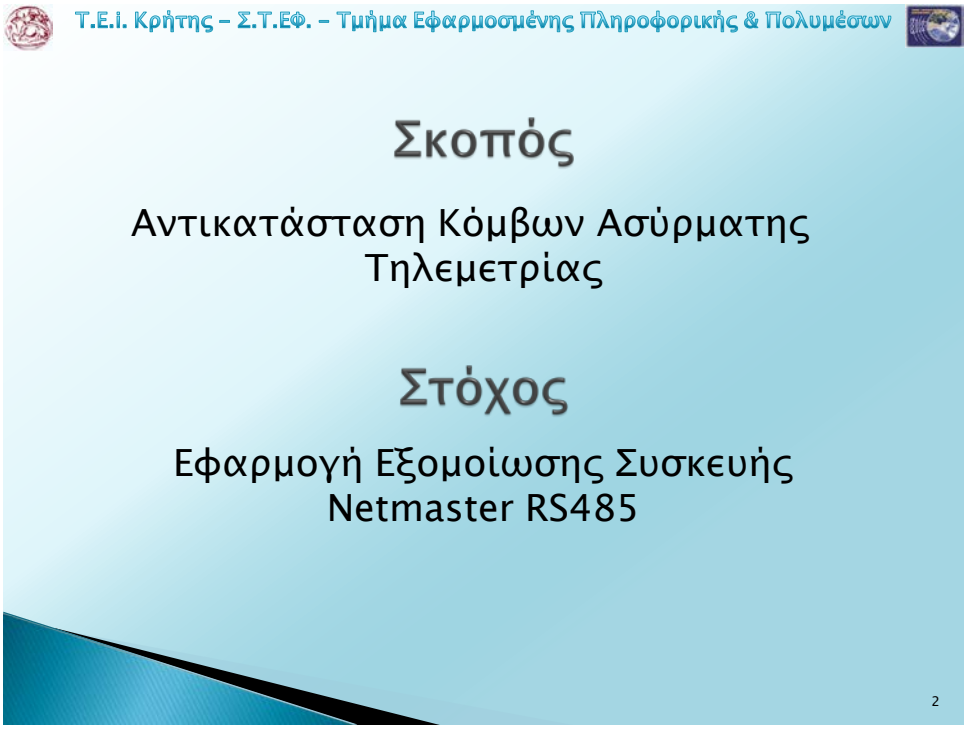
Παράρτημα Α5: Module Αποθήκευσης Αρχικοποιήσεων “mdIni_file”

```
Private Declare Function GetPrivateProfileString Lib "kernel32" Alias "GetPrivateProfileStringA" (ByVal lpApplicationName As String, ByVal lpKeyName As Any, ByVal lpDefault As String, ByVal lpReturnedString As String, ByVal nSize As Long, ByVal lpFileName As String) As Long
Private Declare Function WritePrivateProfileString Lib "kernel32" Alias "WritePrivateProfileStringA" (ByVal lpApplicationName As String, ByVal lpKeyName As Any, ByVal lpString As Any, ByVal lpFileName As String) As Long

'Συναρτήσεις
Function GetFromINI (sSection As String, sKey As String, sDefault As String, sIniFile As String)
   Dim sBuffer As String, lRet As Long
   sBuffer = String$(255, 0)                  ' Γεμίζει το String με 255 κενά
   lRet = GetPrivateProfileString (sSection, sKey, "", sBuffer, Len (sBuffer), sIniFile) ' Καλεί το DLL
   If lRet = 0 Then
      If sDefault <> "" Then AddToINI sSection, sKey, sDefault, sIniFile
      GetFromINI = sDefault                   ' DLL αποτυχής, σώζει τα default
   Else
     GetFromINI = Left (sBuffer, InStr (sBuffer, Chr (0)) - 1) 'DLL επιτυχής επιστρέφει string
   End If
End Function

'Επιστρέφει True εάν είναι επιτυχής και αν δεν υπάρχει η εγγραφή, την δημιουργεί.
Public Function AddToINI(sSection As String, sKey As String, sValue As String, sIniFile As String) As Boolean
   Dim lRet As Long
   lRet = WritePrivateProfileString (sSection, sKey, sValue, sIniFile)    ' Καλεί το DLL
   AddToINI = (lRet)
End Function
```

Παράρτημα Β: Διαφάνειες Παρουσίασης



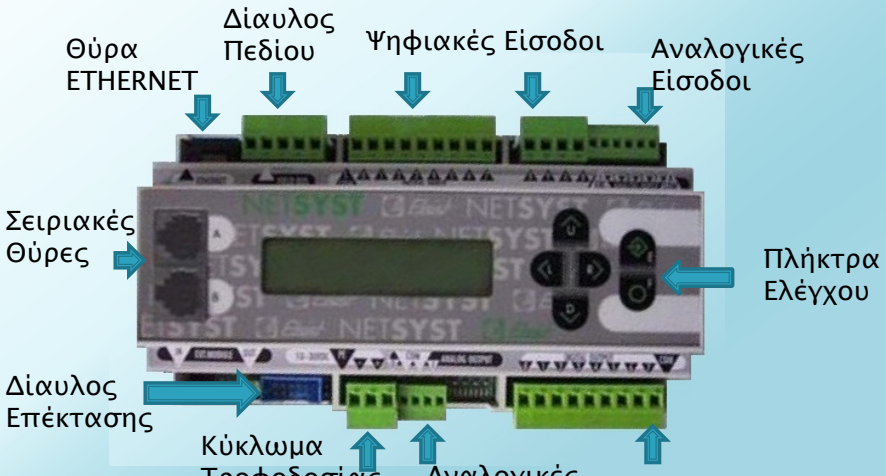

Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Μελέτη
Τηλεμετρία
Συστήματα SCADA
Netmaster RS485
Υλοποίηση Εφαρμογής
3


Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Τηλεμετρία
Τηλέ – Μέτρον
Παρακολούθηση & Τηλεχειρισμός
Ασύρματα & Ενσύρματα
4


Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Συστήματα SCADA
Ορισμός
Supervisory Control And Data Acquisition
Συστατικά
MTU – Master Terminal Unit
MMI – Man Machine Interfaces
Δίκτυο
RTU – PLC
5


Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Netmaster RS485
Τι είναι ;
Συστατικά
Πλαίσιο
Τροφοδοσία
CPU
RAM
LCD Οθόνη
Κουμπιά Ελέγχου
Inputs/Outputs
6

Netmaster RS485 – Inputs/Outputs



7

Netmaster RS485 – Πρωτόκολλο

- Δομή Εξερχόμενων Πακέτων
- Δομή Εισερχόμενων Πακέτων

Index	Value
0	*
1	*
2	Packet Length
3	Route Length
4	Current Destination
5	Route0H
6	Route0L
7	Route1H
8	Route1L
5+RouteLength	Packet Counter
6+ RouteLength	Rssi
7+ RouteLength	Reserved
8+ RouteLength	Reserved
9+ RouteLength	Command
10+ RouteLength	Operands
	……
PacketLength-1	Check Sum

Index	Value
0	*
1	*
2	PacketLength
3	Route Length
4	Current Destination
5	Route0H
6	Route0L
7	Route1H
8	Route1L
5+RouteLength	Packet Counter
6+ RouteLength	Rssi
7+ RouteLength	Reserved
8+ RouteLength	Reserved
9+ RouteLength	Inputs
10+ RouteLength	Outputs
11+ RouteLength	Battery
12+ RouteLength	Water Level
13+ RouteLength	Acc Flow3
14+ RouteLength	Acc Flow2
15+ RouteLength	Acc Flow1
16+ RouteLength	Acc Flow0
17+ RouteLength	Instant Flow3
18+ RouteLength	Instant Flow2
19+ RouteLength	Instant Flow1
20+ RouteLength	Instant Flow0
PacketLength-1	Check Sum

8

Υλοποίηση

- Τίτλος Εφαρμογής

Netmaster RS485 Emulator

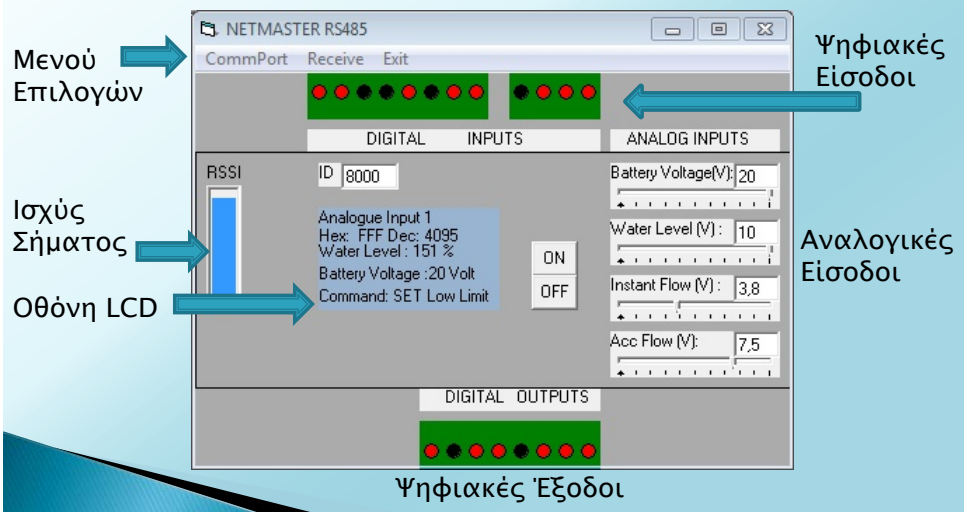
- Τι Κάνει;

- Ανάπτυξη Εφαρμογής
  - Διεπαφή
  - Κώδικας
  - Διάγραμμα Ροής

9

Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

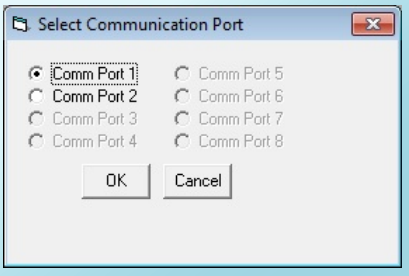
Υλοποίηση – Διεπαφή



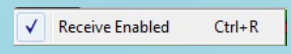
10

Υλοποίηση – Διεπαφή (2)

- Παράθυρο Επιλογής Σειριακής Θύρας



- Παράθυρο Κατάστασης Σειριακής Θύρας



11

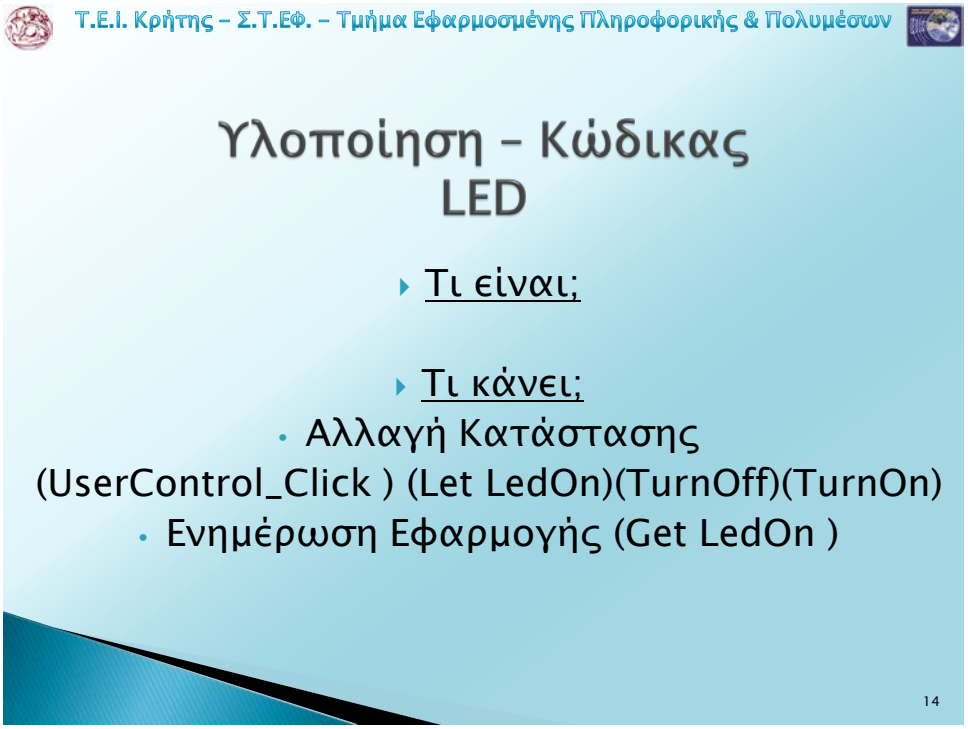
Υλοποίηση – Κώδικας

- Μέρη Κώδικα Εφαρμογής

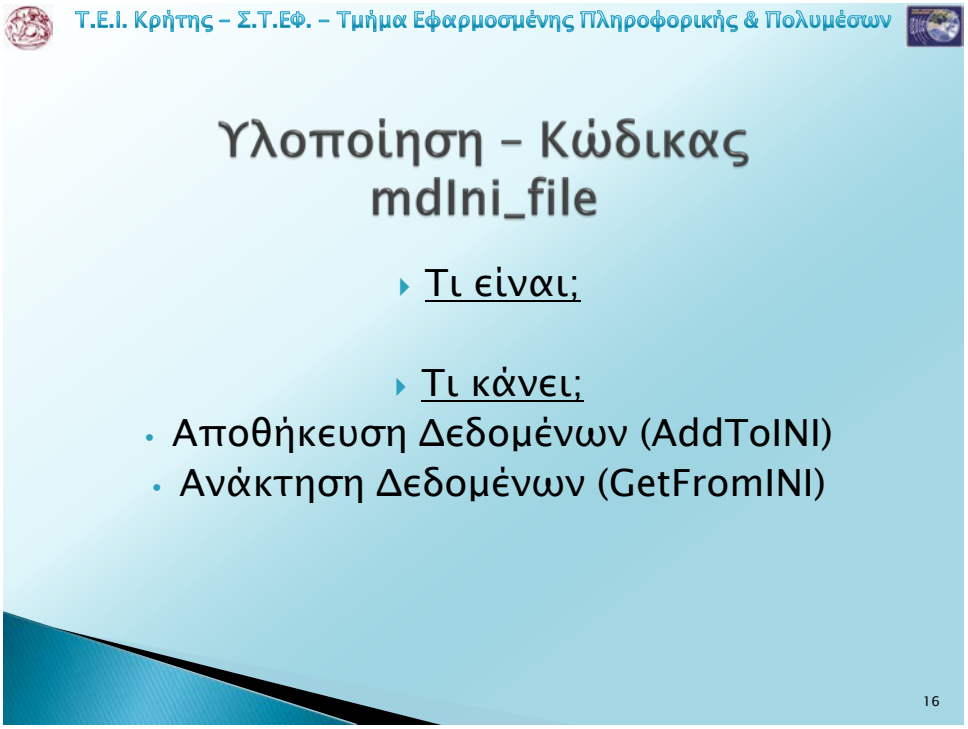
- ViComm
- LED
- frmSetCommPort
- mdIni_file
- ViEmulator

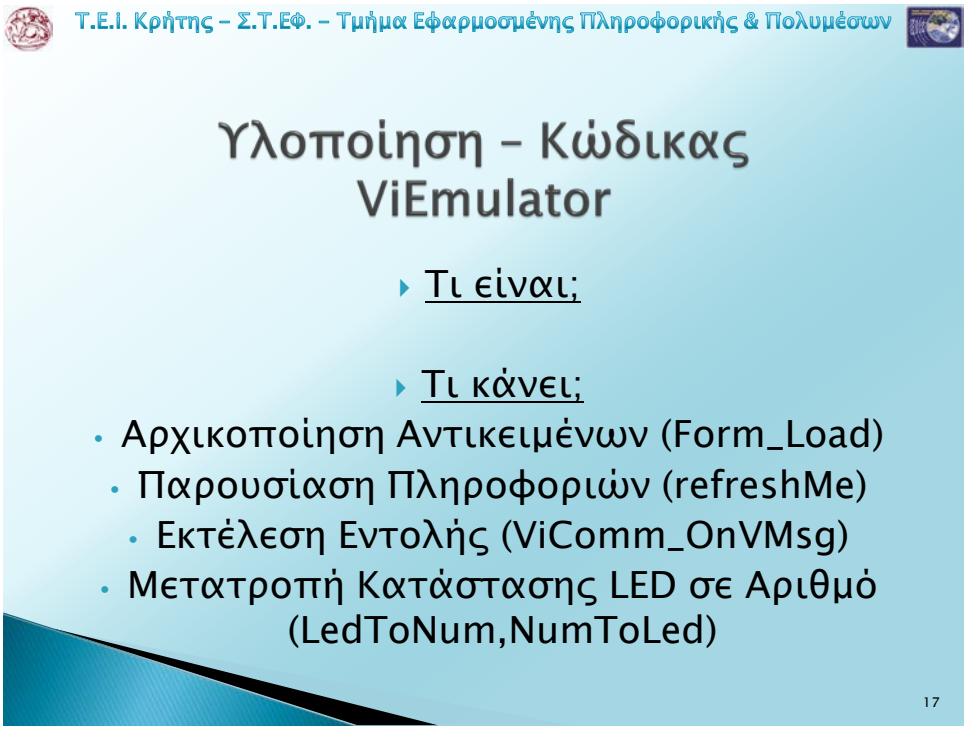
12


Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Υλοποίηση – Κώδικας ViComm
Τι είναι;
Τι κάνει;
Λήψη (MSComm_OnComm)
Αποκωδικοποίηση (BuffHandler)
Κωδικοποίηση
Αποστολή (SendReply)
13

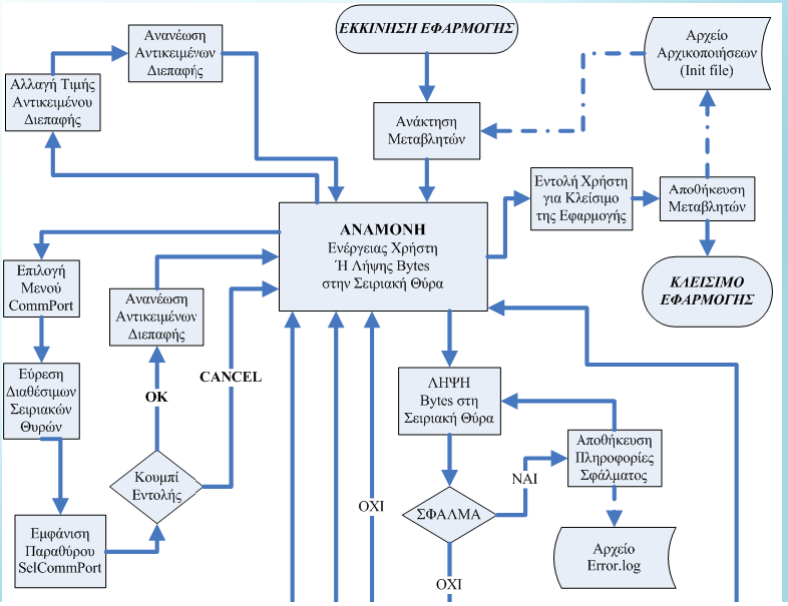

Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Υλοποίηση – Κώδικας LED
Τι είναι;
Τι κάνει;
Αλλαγή Κατάστασης (UserControl_Click) (Let LedOn)(TurnOff)(TurnOn)
Ενημέρωση Εφαρμογής (Get LedOn)
14


Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Υλοποίηση – Κώδικας frmSetCommPort
Τι είναι;
Τι κάνει;
Εύρεση Διαθέσιμων Σειριακών Θυρών (GetAvailableCommports)
Επιλογή Σειριακή Θύρας (cmdOK_Click)
15

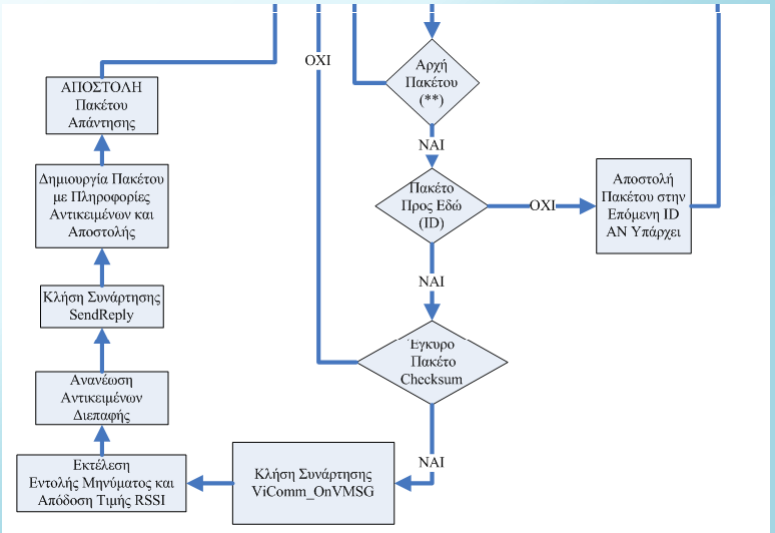

Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Υλοποίηση – Κώδικας mdIni_file
Τι είναι;
Τι κάνει;
Αποθήκευση Δεδομένων (AddToINI)
Ανάκτηση Δεδομένων (GetFromINI)
16

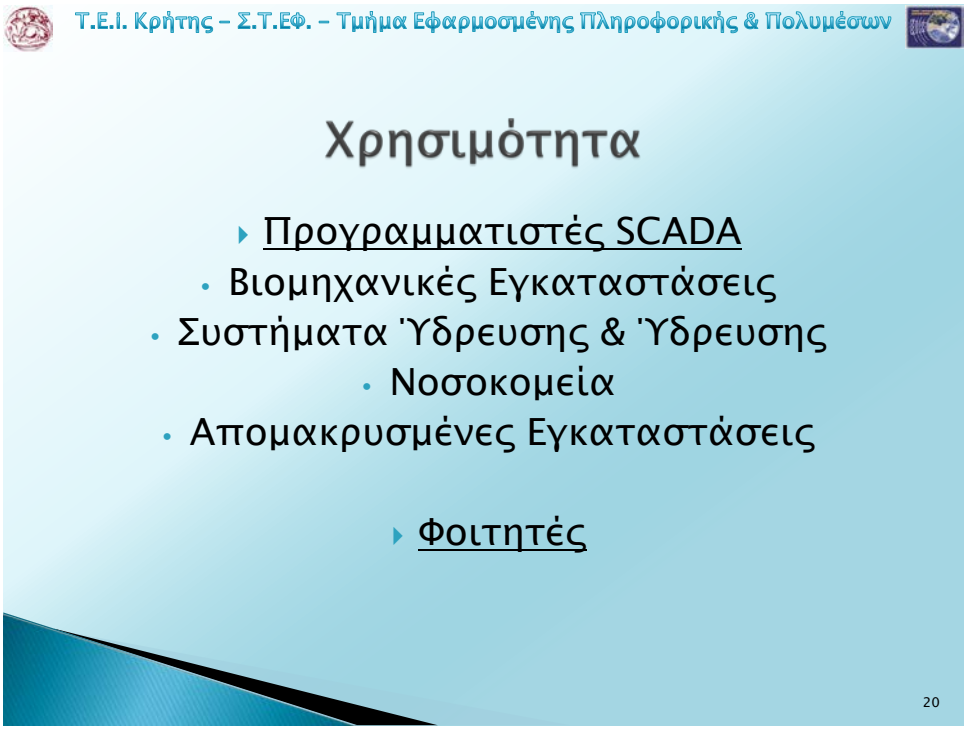

Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Υλοποίηση – Κώδικας ViEmulator
Τι είναι;
Τι κάνει;
Αρχικοποίηση Αντικειμένων (Form_Load)
Παρουσίαση Πληροφοριών (refreshMe)
Εκτέλεση Εντολής (ViComm_OnVMsg)
Μετατροπή Κατάστασης LED σε Αριθμό (LedToNum,NumToLed)
17

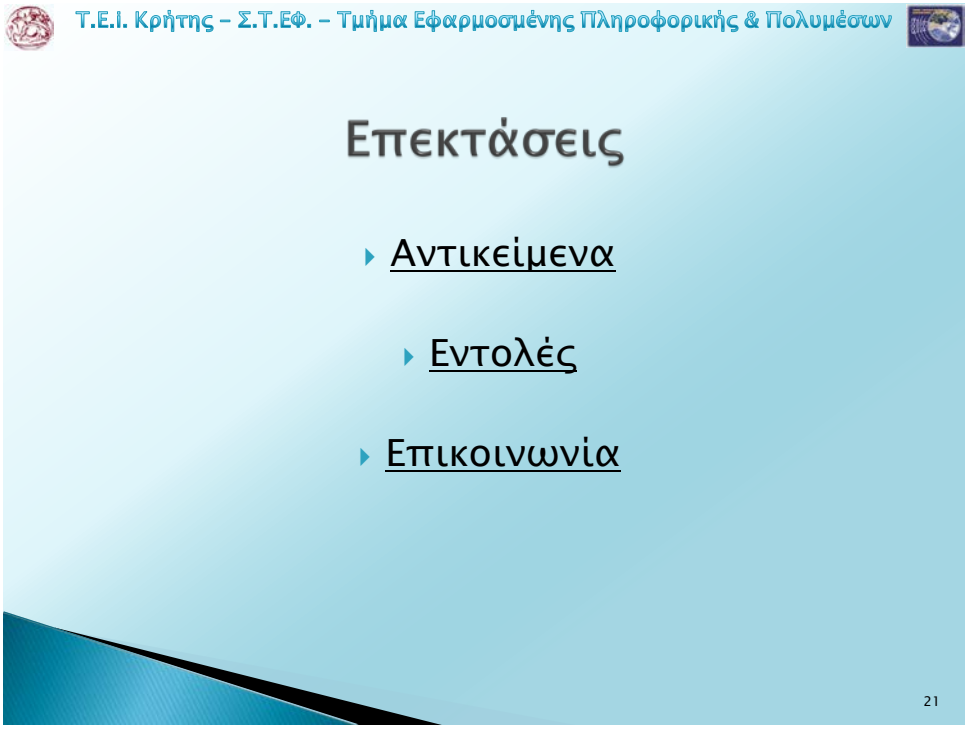
Υλοποίηση – Διάγραμμα Ροής



Υλοποίηση – Διάγραμμα Ροής (2)




Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Χρησιμότητα
Προγραμματιστές SCADA
Βιομηχανικές Εγκαταστάσεις
Συστήματα Ύδρευσης & Ύδρευσης
Νοσοκομεία
Απομακρυσμένες Εγκαταστάσεις
Φοιτητές
20


Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Επεκτάσεις
Αντικείμενα
Εντολές
Επικοινωνία
21


Τ.Ε.Ι. Κρήτης – Σ.Τ.ΕΦ. – Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Βιβλιογραφία
[1] King, Robert – Eric : Πληροφορικός έλεγχος. Εποπτικός έλεγχος και συλλογή πληροφοριών βιομηχανικών διεργασιών (SCADA), Α. Παπασωτηρίου & ΣΙΑ Ο.Ε., Αθήνα 1994
[2] Boyer, Stuart A: SCADA: Supervisory Control and Data Acquisition Instrument Society of America, Research Triangle, N.C. 1993
[3] http://en.wikipedia.org/wiki/Telemetry
[4] http://en.wikipedia.org/wiki/SCADA
[5] http://en.wikipedia.org/wiki/Modbus
[6] http://en.wikipedia.org/wiki/Ascii
[7] http://en.wikipedia.org/wiki/Programmable_logic_controller
[8] http://en.wikipedia.org/wiki/Remote_Terminal_Unit
[9] http://www.elsist.it/WebSite/Html/English/Products/Hardware/PLC/Netsyst/EnNetmasterII.php
[10] Ευάγγελος Πετρούτσος: Πλήρες Εγχειρίδιο της Visual Basic 6 Εκδόσεις Μ. Γκιούρδας
[11] Halvorson M: MS Visual Basic 6.0 Professional Κλειδάριθμος , 1998
[12] http://www.eterlogic.com
22


Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης
Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων
Σας Ευχαριστώ Πολύ
Για Την Προσοχή Σας !
Ερωτήσεις;
23

Παράρτημα Γ: Περίληψη Πτυχιακής σε στυλ Δημοσίευσης

Vitrual Remote Telemetry Node Application

George Emm. Andreadakis
Dept. of Applied Informatics and Multimedia
Technological Educational Institute of Crete
Heraklion, Greece
{epp558@epp.teicrete.gr}

Andreas Miaoudakis
Dept. of Applied Informatics and Multimedia
Technological Educational Institute of Crete
Heraklion, Greece
{miaoudak@epp.teicrete.gr}

The purpose of this paper is to present the development of an application that emulates the device "Netmaster RS485" made by "Elsist" Company. This device is a Programmable Logic Controller (PLC) and functions as a remote telemetry node. The application will be used during the design and implementation of applications that control remotely this device or, with some upgrades, other similar devices.

Keywords: SCADA, Remote Telemetry, Netmaster RS485, Visual Basic Application

I. INTRODUCTION

Nowadays, the use of systems SCADA in the industry is essential. For this reason the devices for this system developed rapidly. But the creation of a network of remote telemetry units, in an interior research place, in order to make the study of their function and documented requirements are quite time consuming and require specialized equipment. Therefore, the aim of this work is to develop an application which will replace the real wireless telemetry nodes during the design and implementation of applications that control wireless such nodes.

II. SCADA SYSTEMS

A. Telemetry

Telemetry is the highly automated communications process by which measurements are made and other data collected at remote or inaccessible points and transmitted to receiving equipment for monitoring. Systems that need external instructions and data to operate require the counterpart of telemetry, telecommand.

Although the term commonly refers to wireless data transfer mechanisms (e.g. using radio, hypersonic or infrared systems), it also encompasses data transferred over other media such as a telephone or computer network, optical link or other wired communications like phase line carriers.

B. Description of SCADA

The abbreviation S.C.A.D.A. comes from the acronym for Supervisory Control and Data Acquisition. The SCADA systems are used to monitor and control a plant or equipment in industries such as telecommunications, control water, waste and energy, refining and transportation of oil and gas. These systems include the transfer of data between a SCADA host server and a number of Remote Terminal Units (RTUs) also between the central server and user terminals. Systems SCADA, consisting of:

- One or more RTU devices to collect and manage information, which communicate with devices for measuring and local control of switches and triggering valves devices.
- A communication system, which is responsible for the transfer of data via a wireless link, a telephone, a satellite, or a combination of them.
- A central server computer, which is commonly called SCADA Center or Master Terminal Unit (MTU)
- A number of Man Machine Interfaces (MMIs), which provide the function of the MTU and administrators workstations. They also support the communication system, present information and control devices for data management.

C. SCADA Architectures

SCADA systems have developed in parallel with the computer technology and divided into three architectures. The first architecture was centralized system based on servers with high computing power, which served all the peripheral units. Then the second was the distributed systems that distributed the process through multiple systems and the third was networked SCADA systems which is open system architecture.

D. SCADA Protocols

The protocols are designed to carry references that contain the list of all input and output devices on the network. Each protocol consists of two sets or pairs of messages. One pair contains the necessary information on admissions and response MTU of the network and the other pair, which is the protocol of RTU, contains the information of which an RTU can connect and respond. In most cases these pairs can be considered a request for information and a confirmation response. Some important protocols used in SCADA systems are Modbus, Modbus X, DNP, ASCII and IEC 60870.

III. "NETMASTER RS485" OF "ELSIST"

A. PLC and RTU Devices

The Remote Terminal Units (RTU) and Programmable Logic Controllers (PLCs) have an important role in the proper

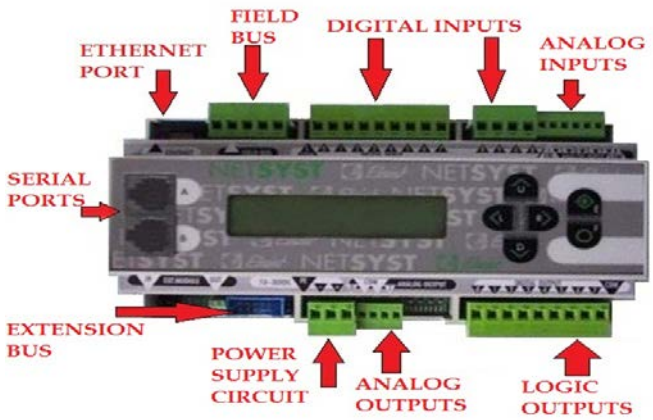
operation of the SCADA system and their usage depends on the needs of the facility and can be in combination or/and individually. The RTUs and PLCs are usually confused because both have the same basic function, remote monitoring and control of remote equipment. Nevertheless, the way they manage it is different, for this reason the RTUs are more appropriate for widespread geographical telemetry by using wireless communications. While PLCs are more suitable for local control space, where the system uses natural ways means to control the devices. This is an area where RTUs are lagging behind because they do not support algorithms or control loops.

1) A Remote Terminal Unit (RTU) is an electronic device controlled by a microprocessor, which interconnects objects from the natural world to a distributed control system or a system of supervisory control and data acquisition (SCADA). The RTU converts incoming signals from the real world, such as pressure, flow, voltages, currents, contacts and pulses into signals that can be sent wired or wirelessly. It also converts incoming signals from another RTU or a central computer to output signals to open or close relays, valves and to start or stop engines.

2) A Programmable Logic Controller (PLC) is a digital electronic system used for automation of electromechanical processes. The PLC uses a programmable memory to store instructions that perform different functions, such as time or measuring operations and to be controlled, via analog or digital units, various machines or processes. Unlike general purpose computers, the PLC is designed for multiple entry and exit arrangements, extended temperature range and resistance to electrical noise and vibrations.

B. Characteristics of Netmaster RS485 (2nd Series)

The device Netmaster RS485 of the Italian company "Elsist" is a programmable controller. As shown at Picture III.1 it is equipped with digital and analog inputs and outputs, with links via bus, with Insulation Displacement Connector (IDC) for connecting expansion modules and with RJ45 connectors for RS232 connection and Ethernet. The Netmaster's connectivity of the network allows easy integration with other systems and offers a flexible and inexpensive solution for remote control and data acquisition through various networks.



Picture1 Connections of Netmaster RS485

C. Communication Protocol of Netmaster RS485

The programmable controller Netmaster RS485, like most devices in a system SCADA, is using ASCII protocol for communications with the master terminal unit (MTU) and the RTUs. When contacting the MTU, the Netmaster sends and receives messages from a software interface (HMI), in the form of data packets. Data packets before they were sent, they were encoded based on the ASCII control code chart and they were synthesized based on a specific structure so that there is a proper communication.

The structure of the packets that the Netmaster receives and sends is common and is divided into Byte. The main Message is what differs between the receive packets and the send packets and there are saved the commands and the information. The message which is stored in the packets that Netmaster receives is a control command and it has sent by the software interface (HMI) that operates the Master Terminal Unit (MTU). On the other hand, the response message contains the status of the inputs and outputs of Netmaster and the information it has gathered from the sensors. At Table 1 is shown the structure of the incoming packets and at Table 3 is shown the response packet structure.

Index	Value
0	*
1	*
2	Packet Length
3	Route Length
4	Current Destination
5	Route0H
6	Route0L
7	Route1H
8	Route1L
5+RouteLength	Packet Counter
6+ RouteLength	Rssi
7+ RouteLength	Reserved
8+ RouteLength	Reserved
9+ RouteLength	Command
10+ RouteLength	Operands
	……..
PacketLength-1	Check Sum

Table 1 Structure of incoming packets

Index	Value
0	*
1	*
2	PacketLength
3	Route Length
4	Current Destination
5	Route0H
6	Route0L
7	Route1H
8	Route1L
5+RouteLength	Packet Counter
6+ RouteLength	Rssi
7+ RouteLength	Reserved
8+ RouteLength	Reserved
9+ RouteLength	Inputs
10+ RouteLength	Outputs
11+ RouteLength	Battery
12+ RouteLength	Water Level
13+ RouteLength	Acc Flow3
14+ RouteLength	Acc Flow2
15+ RouteLength	Acc Flow1
16+ RouteLength	Acc Flow0
17+ RouteLength	Instant Flow3
18+ RouteLength	Instant Flow2
19+ RouteLength	Instant Flow1
20+ RouteLength	Instant Flow0
PacketLength-1	Check Sum

Table 3 Structure of outgoing packets

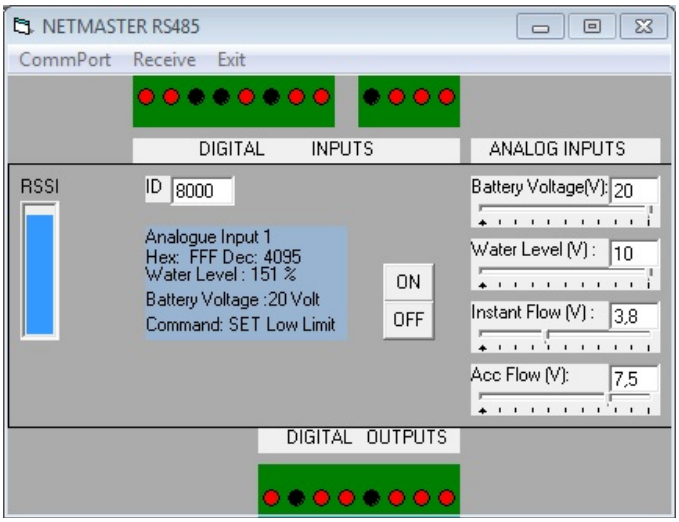
IV. VIRTUAL REMOTE TELEMETRY NODE'S APPLICATION

An application that replaces nodes of wireless telemetry is a useful tool for remote SCADA terminal control applications developers. One such node is the Programmable Logic Controller (PLC) Netmaster RS485 of the "Elsist" that presented in Chapter III and the application developed to emulate this node called "Netmaster RS485 Emulator".

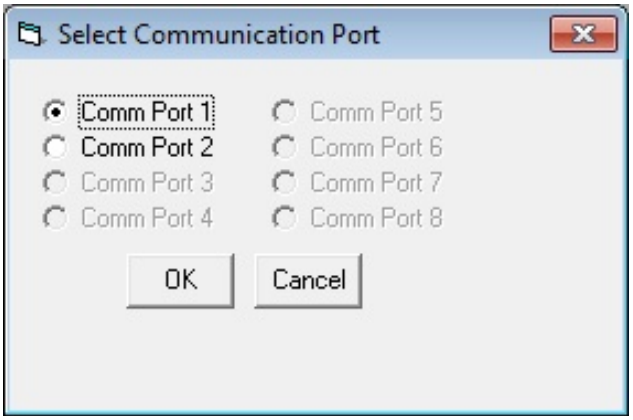
A. Interface Of Application "Netmaster RS485 Emulator"

The interface of the simulator is shown in Picture 2 and its structure is divided into three main parts. The first part is the menu of the application, which was added to perform some basic functions. The second part is the digital inputs and outputs which are located on the top and the bottom of the interface and the third part, which is in the center, contains the analog inputs, the LCD and the other functions.

1) At the menu, the first option reveal the window that is shown at the Picture 3 and where is possible to choose which of the available serial ports will be correspond to the application of the simulator.

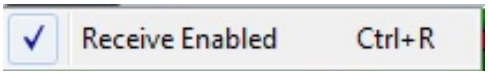


Picture 2 Interface of Netmaster RS485 Emulator



Picture 3 Window of Serial Port Selection

The next item on the menu is the "Receive" and when it is selected the window shown in Picture 4 is revealed. If the property "Receive Enabled" is selected, then the simulator receives incoming packets because the serial port that corresponds to him is open. Otherwise, the door is closed and no packets can be received.



Picture 4 Menu Option "Receive"

The third option on the menu is the EXIT which exits the application. Outputs can also be made by pressing pressing the exit button, which is located in the upper right corner of the application window.

2) The LEDs of the digital inputs and outputs, show the state of specific switches. These objects have two states corresponding to different colors. The colors of the objects are rotated to black, which means that the LED is off and red, which means that the LED is on.

3) In the center of the application window are the essential functions of the Netmaster RS485 simulator's interface. The bar in the left part of the application displays the value of signal strength when the application receives a message. At the right side is a text box that displays the address identification of the simulator and offers the possibility to change this address.

In the center is designed a LCD screen that displays information and measurements, which are useful for monitoring the functions of the application. Beside the LCD screen there are two command buttons, when we press the button that is titled "ON", the LED light on and when we press the button "OFF" the LED turn off. At the right part of the application are four sliders and over everyone is the label and the text box that displays the selected value. The sliders are designed at the interface to represent the value of virtual analog inputs.

B. ActiveX Object"ViComm"

The object "ViComm" is the part of the application's code, which recognizes and decodes packets. When the virtual serial port, which we have assigned to our application receives data then the "ViComm" reads the bytes waiting for a valid message. The "ViComm" recognize a packet as a valid message, when the first two bytes have the value that according to the table ASCII is the symbol asterisk (*).

Once the "ViComm" identify the start of a possible valid message, it begins to store the bytes in a string, whose length is equal to the length of the packet. When the number of bytes that expected is received, the application checks whether the packet is intended for this simulator. The verification is done by using the stored route of addresses and the ID of the simulator. If the packet is not for this emulator, the "ViComm" forwards the packet to the device with the next address in the saved route.

The "ViComm" also has the function of sending packets, as in the above case which forwards the packet not intended for him. The reply message, which is created by the main application, is encoded by the "ViComm". With this message the "ViComm" prepares a package that contains all the necessary information for the transmission and sends it back to the control application.

C. Main Part Of The Application And The Objects

The application Netmaster RS485 Emulator analyzed and designed in separate parts of code for each basic function. This way of programming is followed from the most developers, because makes easy the upgrading and the error handling from the first or future developers.

The code of the Netmaster RS485 Emulator consists of five parts, which are:

- ViComm: It is an ActiveX Object, responsible for the communication of emulator with other applications.
- LED: It is an ActiveX Object which functions as a light to indicate the status of the digital inputs and outputs.
- FrmSetCommPort: It is a Form, which allows us to choose from the available serial ports which one we want to use with the application.
- mdIni_file: It is a Module, responsible for storing and retrieving parameters from the initialization file. The states, measurements and initializes aren't lost after the application closes.
- ViEmulator: It is the Main Form of the application and includes the code that creates and displays the virtual measurements and statements of the emulator. It also contains the code which manages the commands that accepts to receive packets.

D. Conclusion

The application "Netmaster RS485 Emulator" will be useful to programmers who are dealing with the development of applications that control SCADA devices. They can use the emulator to study the functions and to find errors. There is also the possibility to be used by students, who will study a SCADA system or develop, for their final work, an application that controls wireless telemetry devices.

In this application is possible to add more of the items or different ones by an upgrade in order to show more measurements and statements. Also with an upgrade can change the command, that is executed with a specific code which is in the received package and thus the emulator be enriched with different and more functions. Finally, since the main part is the communication and encoding or decoding packets, ActiveX object "ViComm" could be used separately to develop other applications that communicate via a serial port.

REFERENCES

[1] King, Robert - Eric : Informational control. Supervisory control and Industrial Processes Information Acquisition (SCADA) A. Papasotiriou & SIA O.E., Athens 1994

[2] Boyer, Stuart A: SCADA: Supervisory Control and Data Acquisition Instrument Society of America, Research Triangle, NC. 1993

[3] http://en.wikipedia.org/wiki/Telemetry

[4] http://en.wikipedia.org/wiki/SCADA

[5] http://en.wikipedia.org/wiki/Modbus

[6] http://en.wikipedia.org/wiki/Ascii

[7] http://en.wikipedia.org/wiki/Programmable_logic_controller

[8] http://en.wikipedia.org/wiki/Remote_Terminal_Unit

[9] http://www.elsist.it/WebSite/Html/English/Products/Hardware/PLC/Netsyst/EnNetmasterII.php

[10] http://www.eterlogic.com