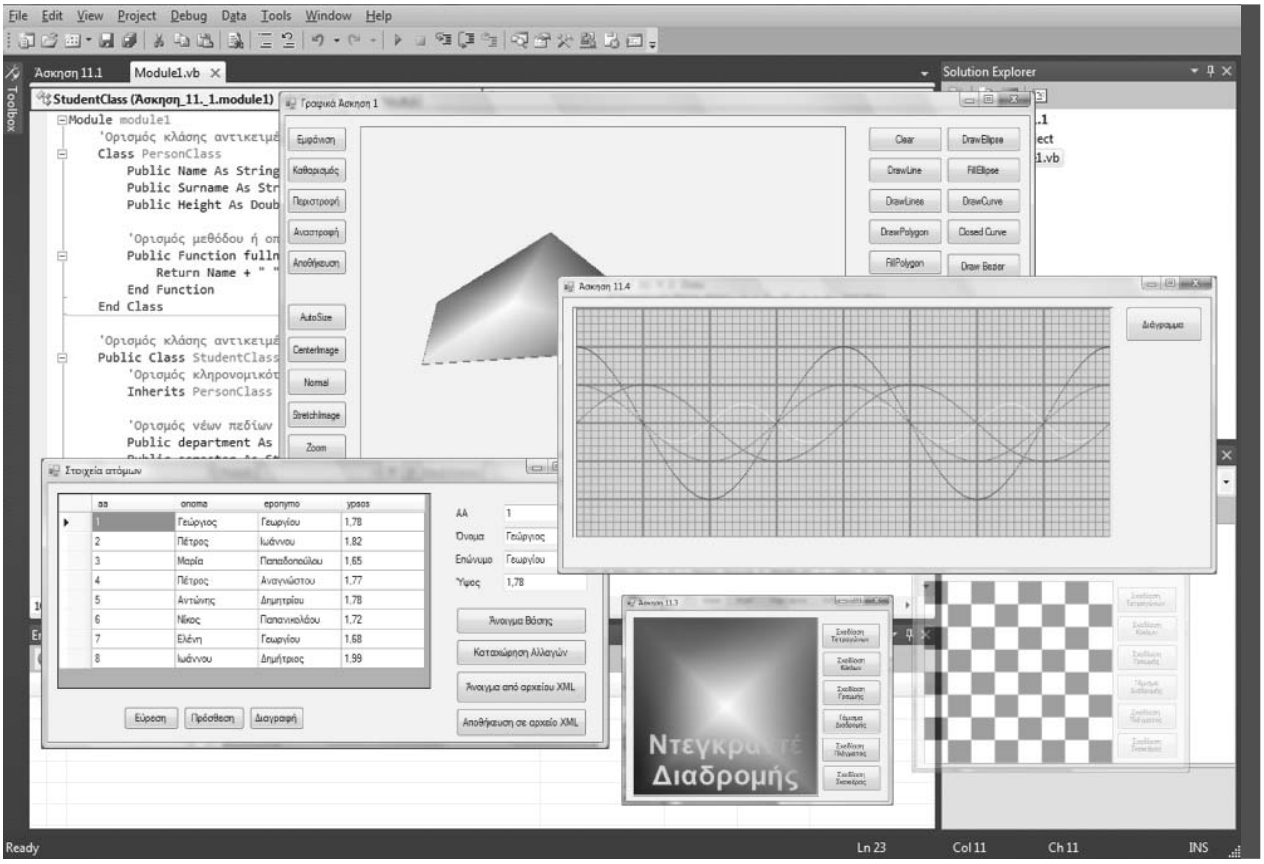
Γρηγόρης Τζιάλλας

Σημειώσεις

Εφαρμογές Προγραμματισμού για Ηλεκτρονικούς



Τμήμα Ηλεκτρονικής
Σ.Τ.ΕΦ.
ΤΕΙ ΛΑΜΙΑΣ

Πίνακας Περιεχομένων

1. Σειριακές επικοινωνίες

1.1 Το RS-232 πρότυπο σειριακών επικοινωνιών

Το RS-232 (Recommended Standard 232) είναι ένα από τα πιο γνωστά πρότυπα για την σειριακή μετάδοση δυαδικών σημάτων δεδομένων μεταξύ συστημάτων.

Το πρότυπο αυτό καθορίζει τα χαρακτηριστικά της επικοινωνίας μεταξύ ενός “Data Terminal Equipment” (DTE, συνήθως ένας υπολογιστής) και ενός “Data Communications Equipment” (DCE, συνήθως μια περιφερειακή μονάδα όπως modem).

Τα κυριότερα χαρακτηριστικά είναι:

- Τα χαρακτηριστικά ηλεκτρικών σημάτων όπως επίπεδα τάσης, χρονισμός, ρυθμός ανόδου των σημάτων και ρυθμός μετάδοσης. Οι λογικές στάθμες είναι η στάθμη ‘1’ (mark) η οποία αντιστοιχεί σε τάση από -3 έως -25V και η στάθμη ‘0’ (space) η οποία αντιστοιχεί σε τάση από 3 έως 25V. Συνήθεις ρυθμοί μετάδοσης αναλόγως της εφαρμογής είναι 200bps, 400bps, 800bps, 1200bps, 2400bps, 4800bps, 9600bps και 19200bps. Αν και σύμφωνα με το πρωτόκολλο ο μέγιστος ρυθμός μετάδοσης είναι 19200bps, σήμερα χρησιμοποιούνται και μεγαλύτερες ταχύτητες όπως 38.4Kbps και 115.2Kbps.
- Η συμπεριφορά σε βραχυκύκλωμα, και η μέγιστη παρασιτική χωρητικότητα. Το πρότυπο δεν ορίζει μέγιστο μήκος του καλωδίου, αλλά αντ ' αυτού χρησιμοποιείται η μέγιστη χωρητικότητα. Το μήκος για συνηθισμένα καλώδια δεν ξεπερνά τα 15 μέτρα, εκτός κι αν γίνει χρήση ειδικών καλωδίων χαμηλής χωρητικότητας, οπότε μπορεί το μήκος του καλωδίου να καλύψει αποστάσεις έως και 300 μέτρα Για μεγαλύτερες αποστάσεις χρησιμοποιούνται άλλα πρότυπα όπως τα RS-422 και RS-485.
- Η λειτουργικότητα των ηλεκτρικών σημάτων. Τα σήματα του RS232 περιλαμβάνουν σήματα για την μεταφορά των δεδομένων και σήματα ελέγχου.
- Τα μηχανικά χαρακτηριστικά της διεπαφής, τους διάφορους τύπους βυσμάτων σύνδεσης και την αντιστοίχιση των ακροδεκτών (pins) στα ηλεκτρικά σήματα .

Τα βασικά σήματα του RS232 τα οποία χρησιμοποιούνται για την μεταφορά δεδομένων είναι το σήμα μετάδοσης TxD , το σήμα λήψης δεδομένων RxD και η κοινή τάσης αναφοράς εδάφους SGND.

Η ελάχιστη RS-232 σύνδεση γίνεται με τα σήματα Txd, RxD και την κοινή τάση αναφοράς του εδάφους SGND. Σε περίπτωση που δεν απαιτείται αμφίδρομη επικοινωνία, όπως για πα-

ράδειγμα όταν ένα μετρητής θερμοκρασίας αποστέλλει περιοδικά την τιμή της θερμοκρασίας, τότε μπορούν να χρησιμοποιηθούν μόνο δύο σήματα.

Όταν χρησιμοποιείται έλεγχος ροής υλικού, τότε είναι απαραίτητη και η χρήση των γραμμών RTS και CTS. Οι συνηθέστεροι σύνδεσμοι RS-232 είναι οι D-9 (με 9 ακροδέκτες) και D-25 (με 25 ακροδέκτες). Στον παρακάτω πίνακα φαίνονται οι αντιστοιχίες των ακροδεκτών τους στα σήματα του RS-232.

Ακροδέκτες σε σύνδεσμο D-25	Ακροδέκτες σε σύνδεσμο D-9	Κωδική ονομασία	Όνομα
2	3	TXD	Transmit Data
3	2	RXD	Receive Data
7	5	SGND	Signal Ground
4	7	RTS	Request To Send
5	8	CTS	Clear To Send
6	6	DSR	Data Set Ready
20	4	DTR	Data Terminal Ready
8	1	CD	Carrier Detect
22	9	RI	Ring Indicator

Πίνακας 1.1 Πίνακας αντιστοιχιών συνδέσμων D-9 και D-25 στα σήματα του RS-232

Η μετάδοση των δεδομένων γίνεται σειριακά με σταθερό ρυθμό αρχίζοντας με το πρώτο λιγότερο σημαντικό bit (LSB). Η λειτουργία της μετάδοσης δεδομένων είναι η ακόλουθη:

- Ο αποστολέας ξεκινά την μετάδοση προσθέτοντας στην αρχή ένα start bit, σκοπός του οποίου είναι ο συγχρονισμός του παραλήπτη.
- Ο παραλήπτης, ο οποίος ελέγχει περιοδικά τη γραμμή, εντοπίζει την κατερχόμενη ακμή του start bit και ξεκινά μετά από χρόνο T/2 (όπου T ισούται με τον ονομαστικό χρόνο κάθε bit) την δειγματοληψία.
- Η λέξη συμπληρώνεται προαιρετικά από ένα parity bit (αναλόγως της επιλεγμένης ισοτιμίας)
- Η μετάδοση ολοκληρώνεται με την αποστολή 0, 1, 1.5 ή 2 stop bits. Τα bits αυτά εξασφαλίζουν ότι η γραμμή θα είναι για κάποιο διάστημα σε υψηλή κατάσταση πριν το επόμενο start bit. Επίσης δίνουν ένα περιθώριο χρόνου στον παραλήπτη (π.χ. για αποθήκευση της λέξης), πριν την έναρξη της επόμενης μεταφοράς.

Τη διαδικασία ασύγχρονης σειριακής αποστολής και λήψης δεδομένων σε ένα υπολογιστικό σύστημα αναλαμβάνει συνήθως τμήμα υλικού, το οποίο ονομάζεται Universal Asynchronous Receiver Transmitter (UART). Το UART μετατρέπει τις λέξεις σε σειριακή ακολουθία bits, προσθέτει start/stop και parity bits, και στη συνέχεια μεταδίδει τα δεδομένα με τον επιλεγμένο ρυθμό μετάδοσης. Το UART με την αντίστροφη διαδικασία λαμβάνει δεδομένα και ειδοποιεί τον μικροεπεξεργαστή για την παραλαβή τους.

Ο αποστολέας και ο παραλήπτης πρέπει να χρησιμοποιούν τις ίδιες παραμέτρους σειριακής επικοινωνίας. Οι παράμετροι αυτοί είναι ο ρυθμός μετάδοσης, ο αριθμός bits ο οποίος μεταδίδεται (5 έως 8 bits με συνηθέστερη τιμή τα 8 bits), ο έλεγχος ισοτιμίας (even, odd, none, mark ή space) και ο αριθμός των stop bits (0, 1, 1.5 ή 2).

1.2 Διαφορική σειριακή μετάδοση.

Μία παραλλαγή του RS-232 αποτελεί το πρότυπο RS-422. Σύμφωνα με το πρότυπο αυτό, η μετάδοση της πληροφορίας επιτυγχάνεται με διαφορική μέθοδο. Για τη λήψη της πληροφορίας χρησιμοποιείται η διαφορά των δύο σημάτων. Το σήμα λόγω του διαφορικού τρόπου μετάδοσης είναι πολύ ανθεκτικό σε κοινό θόρυβο επειδή ο θόρυβος επηρεάζει εξίσου τα δύο σήματα.

Με την χρήση του πρότυπου RS-422 επιτυγχάνεται η αποστολή δεδομένων σε αποστάσεις μεγαλύτερες από 1 χλμ. ενώ ο ρυθμός μετάδοσης των δεδομένων ανέρχεται έως τα 10Mbps.

Συχνά τα σήματα του RS-422 απομονώνονται ηλεκτρικά με οπτικούς απομονωτές (opto-isolators) διότι σε μεγάλες αποστάσεις η διαφορά δυναμικού μπορεί να γίνει στιγμιαία πολύ μεγάλη (από αιχμές ρεύματος ή λόγω φορτίων καιρικών συνθηκών).

1.3 Αναφορές

- "Serial Communication General Concepts", http://zone.ni.com/devzone/cda/tut/p/id/11390
- Μ.Στεφανιδάκης, "Διασύνδεση Μικροϋπολογιστικών Συστημάτων 2004-05", Κεφάλαιο 4. Σειριακή επικοινωνία, http://www.vlsi.ee.upatras.gr/~karagian/dias04n.pdf
- Ι. Καλόμοιρος , "Συστήματα Πληροφορικής και Μετρήσεων", Κεφάλαιο 5, Βασικές τεχνικές εισόδου/εξόδου δεδομένων, http://www.teiser.gr/icd/staff/kalomiros/Syllogi_Metrisewn/Kef5_commun_ser_paral.doc
- "RS-232", http://en.wikipedia.org/wiki/RS-232
- "RS232 Data Interface. A Tutorial on Data Interface and cables", http://www.arcelect.com/rs232.htm

2. Σειριακές επικοινωνίες και Visual Basic

2.1 Σειριακή επικοινωνία με την Visual Basic

Η σειριακή επικοινωνία με την Visual Basic γίνεται με την χρήση της κλάσης SerialPort. Για την χρήση της κλάσης αυτής πρέπει να χρησιμοποιηθεί η δήλωση:

```
Imports System.IO.Ports
```

Οι κυριότερες ιδιότητες, μέθοδοι και συμβάντα της κλάσης αυτής είναι:

Ιδιότητα	Περιγραφή
PortName	Καθορίζει την σειριακή θύρα. π.χ. MySerialPort.PortName="Com1" Το όνομα της σειριακής θύρας πρέπει να αντιστοιχεί σε μια υπαρκτή σειριακή θύρα του Η/Υ
BaudRate	Καθορίζει τον ρυθμό μετάδοσης π.χ. MySerialPort.BaudRate=9600
Parity	Καθορίζει την ισοτιμία (Even, Odd, None, Mark ή Space) π.χ. MySerialPort.Parity = Parity.Even
DataBits	Καθορίζει τον αριθμό των bits τα οποία μεταδίδονται (από 5 έως 8) π.χ. MySerialPort.DataBits = 8
StopBits	Καθορίζει τον αριθμό των stop bits (0, 1, 1.5 ή 2) π.χ. MySerialPort.StopBits = StopBits.One
Encoding	Ιδιότητα η οποία καθορίζει την κωδικοποίηση του κειμένου σε περίπτωση που αποστέλλονται ή λαμβάνονται χαρακτήρες ή κείμενο. πχ. MySerialPort. Encoding = System.Text.Encoding.Unicode

Μέθοδοι	Περιγραφή
New	Η μέθοδος αυτή δημιουργεί ένα νέο αντικείμενο της κλάσης αυτής. πχ. Dim WithEvents MySerialPort As New SerialPort Η δήλωση WithEvents γίνεται ώστε να είναι διαθέσιμα τα συμβάντα του αντικειμένου και συγκεκριμένα το συμβάν DataReceived το οποίο ενεργοποιείται όταν η σειριακή θύρα λαμβάνει δεδομένα.
Open()	Ανοίγει την σειριακή πόρτα για επικοινωνία. π.χ. MySerialPort.Open()
Close()	Κλείνει την σειριακή πόρτα
ReadExisting()	Διαβάζει το κείμενο τα οποίο έχει ληφθεί από την σειριακή θύρα.
Read(buffer() of Byte ή Char, offset as Integer count as Integer)	Διαβάζει και επιστρέφει ένα πίνακα bytes ή χαρακτήρων από την σειριακή θύρα. Η παράμετρος offset καθορίζει τον δείκτη αρχής και η παράμετρος count το πλήθος των bytes ή χαρακτήρων τα οποία θα επιστραφούν από τον πίνακα στοιχείων που έχουν παραληφθεί από την σειριακή πόρτα.
BytesToRead	Επιστρέφει το πλήθος των bytes τα οποία έχουν ληφθεί από την σειριακή θύρα και είναι προσωρινά αποθηκευμένα μέχρι να αναγνωσθούν.
Write(text as String ή Buffer() as Byte ή Buffer as Char())	Αποστέλλει κείμενο ή έναν πίνακα Bytes ή ένα πίνακα χαρακτήρων. Όταν αποστέλλεται πίνακας, τότε πρέπει να δοθούν επιπλέον ως παράμετροι ο δείκτη αρχής του πίνακα (offset) και το πλήθος των στοιχείων (count)
IsOpen	Επιστρέφει την τιμή true όταν η θύρα είναι ανοιχτή και την τιμή false σε αντίθετη περίπτωση.
Συμβάν	Περιγραφή
DataReceived	Το συμβάν αυτό προκαλείται όταν η σειριακή θύρα δεχθεί δεδομένα. Το συμβάν αυτό διασυνδέεται με κάποια μέθοδο ή οποία διαβάζει τα δεδομένα από την σειριακή θύρα με την μέθοδο ReadExisting.

Πίνακας 2.1 Η κλάση αντικειμένων SerialPort

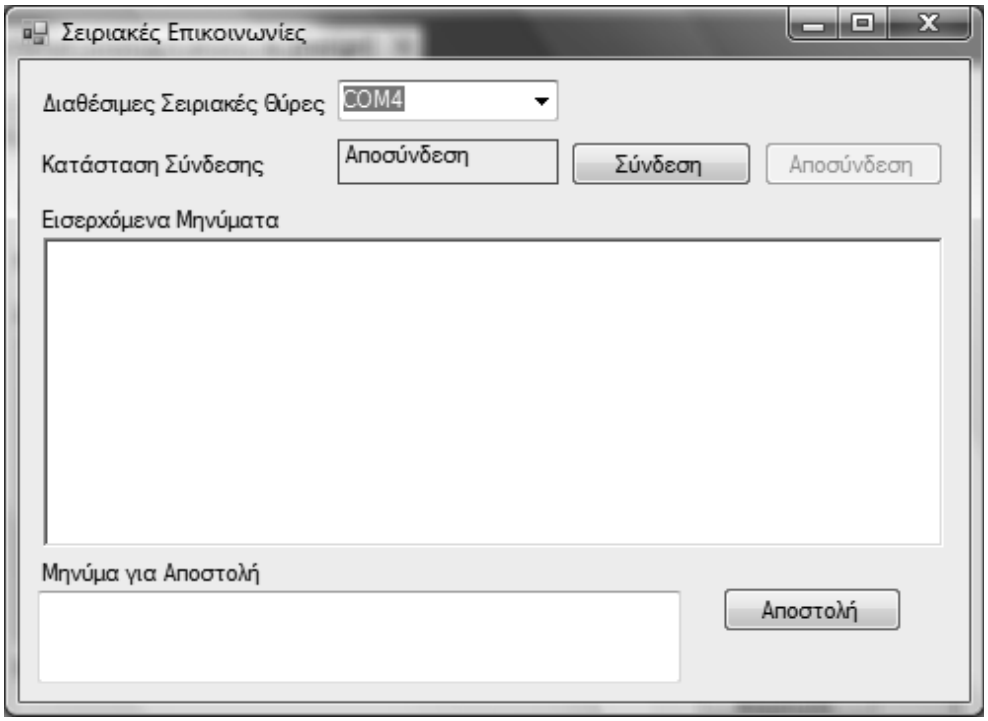
2.2 Πρόγραμμα Visual Basic για σειριακή επικοινωνία

Η παραθυρική εφαρμογή η οποία φαίνεται στο σχήμα 15.1 μπορεί να στείλει και λάβει κείμενο με την χρήση της σειριακής θύρας επικοινωνιών. Η εφαρμογή χρησιμοποιεί ένα παράθυρο με όνομα FormSerialCommunications. Το παράθυρο αυτό έχει τα παρακάτω στοιχεία ελέγχου:

Όνομα	Τύπος	Περιγραφή
ComboBoxCOMPorts	ComboBox	Χρησιμοποιείται για να εμφανίσει όλες τις διαθέσιμες σειριακές θύρες του Η/Υ και να δώσει στον χρήστη την δυνατότητα να επιλέξει την σειριακή θύρα την οποία θα χρησιμοποιήσει

TextBoxDataReceived	TextBox	Εμφανίζει στον χρήστη τα εισερχόμενα μηνύματα
TextBoxDataToSend	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί το μήνυμα το οποίο θέλει να αποστείλει
LabelStatus	Label	Ετικέτα η οποία εμφανίζει την τρέχουσα κατάσταση της σειριακής θύρας (Σύνδεση ή Αποσύνδεση)
ButtonConnect		Πλήκτρο για άνοιγμα της σειριακής θύρας
ButtonDisconnect		Πλήκτρο για το κλείσιμο της σειριακής θύρας
ButtonSend		Πλήκτρο για την αποστολή δεδομένων

Πίνακας 2.2 Τα στοιχεία ελέγχου της εφαρμογής



Εικόνα 2.1 Παραθυρική εφαρμογή για σειριακή επικοινωνία

Η παραθυρική εφαρμογή πρέπει να εκτελείται ταυτόχρονα από δύο χρήστες οι οποίοι έχουν διασυνδέσει τις σειριακές θύρες των υπολογιστών τους με ένα σειριακό καλώδιο.

Στον παρακάτω πίνακα φαίνονται οι αντιστοιχίες των ακροδεκτών σε σύνδεσμο D-9 για σειριακό καλώδιο τριών γραμμών το οποίο συνδέει τις σειριακές θύρες δύο Η/Υ (Η/Υ 1 και Η/Υ 2)

Η/Υ 1 Ακροδέκτες σε σύνδεσμο D-9	Η/Υ 1 Σήμα RS-232	Η/Υ 2 Σήμα RS-232	Η/Υ 2 Ακροδέκτες σε σύνδεσμο D-9
3	TXD	RXD	2
2	RXD	TXD	3
5	SGND	SGND	5

Πίνακας 2.3 Αντιστοιχίες ακροδεκτών για την σειριακή σύνδεση δύο Η/Υ

Όπως φαίνεται από τον πίνακα, τα σήματα RxD και TxD αντιστρέφονται. Το σήματα RxD και TxD του πρώτου Η/Υ συνδέονται με τα σήματα TxD και RxD του δεύτερου Η/Υ αντίστοιχα.

Η αντιστροφή των σημάτων στο σειριακό καλώδιο γίνεται όταν επικοινωνούν δύο Η/Υ μεταξύ τους. Σε περίπτωση όπου η επικοινωνία γίνεται μεταξύ ενός Η/Υ και μιας περιφερειακής συσκευής (όπως εκτυπωτής ή modem), τα σήματα δεν πρέπει να αντιστρέφονται. Το σήματα TxD και RxD του Η/Υ συνδέονται με τα σήματα TxD και RxD της περιφερειακής συσκευής αντίστοιχα.

Οι χρήστες των προγραμμάτων επικοινωνίας στους δύο Η/Υ πρέπει:
- Να επιλέξουν την σειριακή θύρα στην οποία έχει συνδεθεί το καλώδιο επικοινωνίας
- Να πατήσουν το πλήκτρο "Σύνδεση" για να ανοίξουν την σειριακή θύρα και να συνδεθούν μεταξύ τους
- Να πληκτρολογήσουν κάποιο μήνυμα και να πατήσουν στην συνέχεια το πλήκτρο "Αποστολή". Τα μηνύματα τα οποία αποστέλλονται, εμφανίζονται στο πλαίσιο κειμένου με τίτλο "Εισερχόμενα μηνύματα".
- Να τερματίσουν την επικοινωνία τους με το πάτημα του πλήκτρου "Αποσύνδεση"

Για την δοκιμή του προγράμματος σε έναν Η/Υ, ο οποίος μπορεί να μην διαθέτει καν σειριακές θύρες, μπορεί να χρησιμοποιηθεί το πρόγραμμα Com0Com. Το πρόγραμμα αυτό εγκαθιστά 2 εικονικές σειριακές θύρες στον Η/Υ οι οποίες είναι και εικονικά διασυνδεδεμένες μεταξύ τους. Ότι αποστέλλεται από την μία θύρα λαμβάνεται από την άλλη και αντίστροφα. Η δοκιμή του προγράμματος μπορεί να γίνει με την παράλληλη εκτέλεση στον Η/Υ όπου έχει εγκατασταθεί το πρόγραμμα com0com δύο εφαρμογών για την σειριακή επικοινωνία οι οποίες χρησιμοποιούν τις εικονικές σειριακές θύρες του com0com.

Ο κώδικας Visual Basic του παραθύρου της εφαρμογής FormSerialCommunications με τον οποίο υλοποιείται η σειριακή επικοινωνία δίνεται παρακάτω.

```
Imports System.IO.Ports
Public Class FormSerialCommunications

    'Ορισμός αντικειμένου της κλάσης SerialPort για την σειριακή επικοινωνία
    Public WithEvents MySerialPort As New SerialPort

    Private Sub FormSerialComms(ByVal sender As System.Object, _
                    ByVal e As System.EventArgs) Handles MyBase.Load
        'Χρήση της μεθόδου Computer.Ports.SerialPortNames η οποία
        'επιστρέφει μια συλλογή με τις διαθέσιμες θύρες του Η/Υ για
        'το γέμισμα του ComboBoxCOMPorts
        For i As Integer = 0 To My.Computer.Ports.SerialPortNames.Count - 1
            ComboBoxCOMPorts.Items.Add(My.Computer.Ports.SerialPortNames(i))
        Next
        LabelMessage.Text = "Αποσύνδεση"
```

```
        ButtonDisconnect.Enabled = False
    End Sub

    Private Sub ButtonConnect_Click(ByVal sender As System.Object, _
                ByVal e As System.EventArgs) Handles ButtonConnect.Click
        'Έλεγχος αν η επιλεγμένη σειριακή πόρτα είναι ανοικτή
        If MySerialPort.IsOpen Then
            'Κλείσιμο της σειριακής πόρτα εάν είναι ανοικτή
            MySerialPort.Close()
        End If
        Try
            'Καθορισμός παραμέτρων επικοινωνίας
            With MySerialPort
                .PortName = ComboBoxCOMPorts.Text   'Θύρα
                .BaudRate = 96000                   'Ρυθμός μετάδοσης
                .Parity = Parity.None               'Ισοτιμία
                .DataBits = 8                       'Αριθμός Bits δεδομένων
                .StopBits = StopBits.One            'Αριθμός Stop bits
                .Encoding = _
                       System.Text.Encoding.Unicode 'Κωδικοποίηση χαρακτήρων
            End With

            'Άνοιγμα σειρίακής θύρας
            MySerialPort.Open()
            'Εμφάνιση κατάστασης και
            'ενεργοποίηση/απενεργοποίηση πλήκτρων σύνδεσης
            LabelMessage.Text = "Σύνδεση"
            ButtonConnect.Enabled = False
            ButtonDisconnect.Enabled = True
        Catch ex As Exception
            'Εμφάνιση μηνύματος σε περίπτωση σφάλματος
            MsgBox(ex.ToString)
        End Try
    End Sub

    Private Sub ButtonDisconnect_Click(ByVal sender As System.Object, _
          ByVal e As System.EventArgs) Handles ButtonDisconnect.Click
        Try
            'Κλείσιμο σειριακής θύρας
            MySerialPort.Close()
            'Εμφάνιση κατάστασης και
            'ενεργοποίηση/απενεργοποίηση πλήκτρων σύνδεσης
            LabelMessage.Text = "Αποσύνδεση"
            ButtonConnect.Enabled = True
            ButtonDisconnect.Enabled = False
        Catch ex As Exception
            MsgBox(ex.ToString)
        End Try
    End Sub

    'Ορισμός αναφοράς για την κλήση της μεθόδου DisplayReceivedMessage
    Public Delegate Sub myDelegate()

    'Μέθοδος η οποία καλείται από το συμβάν DataReceived. Το συμβάν αυτό
    'ενεργοποιείται κάθε φορά που η σειριακή θύρα δέχεται δεδομένα
    Private Sub DataReceived(ByVal sender As Object, _
                ByVal e As SerialDataReceivedEventArgs) _
                Handles MySerialPort.DataReceived
        'Κλήση της μεθόδου DisplayReceivedMessage για την εμφάνιση του
        'εισερχόμενου μηνύματος. Η κλήση γίνεται από το TextBoxDataReceived
        'ώστε να μη γίνει σφάλμα Cross Thread
        TextBoxDataReceived.Invoke( _
                New myDelegate(AddressOf DisplayReceivedMessage))
    End Sub
```

```
    Private Sub ButtonSend_Click(ByVal sender As System.Object, _
                ByVal e As System.EventArgs) Handles ButtonSend.Click
        Try
            'Αποστολή δεδομένων με την μέθοδο Write της κλάσης SerialPort
            MySerialPort.Write(TextBoxDataToSend.Text & vbCrLf)
            'Καθαρισμός πλασίου κειμένου με το μήνυμα προς αποστολή
            TextBoxDataToSend.Text = ""
        Catch ex As Exception
            MsgBox(ex.ToString)
        End Try
    End Sub

    Public Sub DisplayReceivedMessage()
        Dim ReceivedMessage As String
        'Διάβασμα μηνύματος
        ReceivedMessage = MySerialPort.ReadExisting
        'Πρόσθεση στο τέλος του πλαισίου κειμένου του
        'μηνύματος το οποίο παρελήφθη
        TextBoxDataReceived.AppendText(ReceivedMessage)
        'Κύλιση του πλαισίου κειμένου ώστε να φαίνεται το νέο κείμενο το
        'οποίο προστέθηκε
        TextBoxDataReceived.ScrollToCaret()
    End Sub

End Class
```

Παράδειγμα 2.1 Σειριακή επικοινωνία με την Visual Basic

2.3 Αναφορές

- "Serial Port Using Visual Basic .NET and Windows", http://www.me.umn.edu/courses/me2011/smartprodcourse/technotes/docs/serial-port-vb.pdf

- "Serial COM port communication by means of Visual Basic .NET", http://www.innovatic.dk/knowledg/SerialCOM/SerialCOM.htm

- Vincent Himpe, "Visual Basic for Electronics Engineering Applications. Second Edition", Chapter23 The Serial Port in Detail, σελίδες 422-440

- "Null-modem emulator", http://com0com.sourceforge.net/

- "Delegates", http://www.dotnetzone.gr/cs/forums/thread/7316.aspx

2.4 Ασκήσεις Κεφαλαίου

Άσκηση 2.1

Κατεβάστε από την διαδικτυακή τοποθεσία http://com0com.sourceforge.net/ το πρόγραμμα com0com και εγκαταστήστε το στον υπολογιστή σας. Επιλέξτε ως σειριακές θύρες εγκατάστασης τις θύρες com18 και com19.

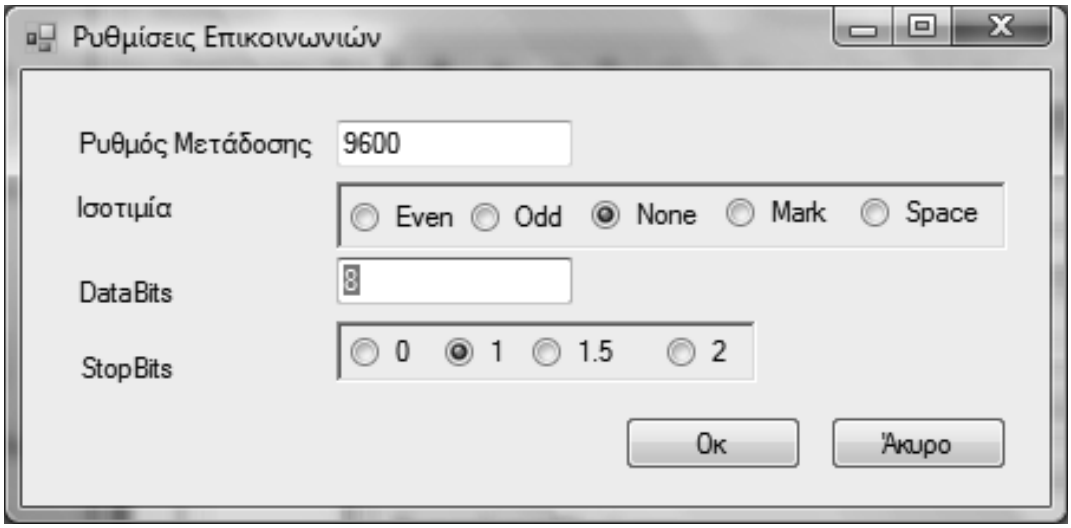
Δημιουργήστε παραθυρική εφαρμογή με όνομα SerialComms, μετονομάστε το παράθυρο Form1 της εφαρμογής σε FormSerialCommunications, προσθέσετε τα στοιχεία ελέγχου του πίνακα 15.1, και αντιγράψτε το κώδικα του παραδείγματος 15.1.

Αποθηκεύστε την εφαρμογή και εκτελέστε τη. Επιλέξτε την θύρα com18 για την επικοινωνία και πατήστε το πλήκτρο "Σύνδεση"

Επιλέξτε το εικονίδιο (η την επιλογή από το μενού εκκίνησης) της Visual Basic για να ξεκινήσετε ένα δεύτερο περιβάλλον ανάπτυξης εφαρμογών της Visual Basic, και ανοίξτε ξανά την εφαρμογή SerialComms. Εκτελέστε την εφαρμογή, επιλέξτε την θύρα επικοινωνίας com19, πατήστε το πλήκτρο σύνδεση και δοκιμάστε να στείλετε δεδομένα μεταξύ των δύο εφαρμογών.

Άσκηση 2.2

Τροποποιήστε την εφαρμογή SerialComms και προσθέστε πλήκτρο με τίτλο "Ρυθμίσεις" και παράθυρο με όνομα FormSettings. Με το πλήκτρο "Ρυθμίσεις" εμφανίζεται το παράθυρο FormSettings το οποίο επιτρέπει στον χρήστη να επιλέξει τον ρυθμό μετάδοσης, την ισοτιμία και τον αριθμό των databits και stopbits.



Το παράθυρο εμφανίζει αρχικά τις τρέχουσες ρυθμίσεις της σειριακής θύρας και με το πάτημα του πλήκτρου "Οκ" ενημερώνει τη σειριακή θύρα με τις νέες ρυθμίσεις που επέλεξε ο χρήστης.

Άσκηση 2.3

Δημιουργήστε δύο εφαρμογές με όνομα SerialTemperatureServer και SerialTemperatureDevice οι οποίες επικοινωνούν μεταξύ τους σειριακά με το πρωτόκολλο RS-232. Η εφαρμογή SerialTemperatureServer στέλνει κάθε 5 δευτερόλεπτα το μήνυμα "Get Temperature" στην εφαρμογή SerialTemperatureDevice. Η εφαρμογή SerialTemperatureDevice όταν λάβει το μήνημα "Get Temperature" (και μόνον τότε) απαντά με το μήνυμα "Temp = xx.xx" όπου xx.xx είναι πραγματικός αριθμός με ακρίβεια δύο δεκαδικών ψηφίων ο οποίος παριστά την θερμοκρασία την οποία μετρά κάποια περιφερειακή συσκευή. Για την δοκιμή της άσκησης χρησιμοποιήστε ως θερμοκρασία την τιμή την οποία εισάγει ο χρήστης σε πλαίσιο κειμένου με όνομα TextBoxTemperture. Η εφαρμογή SerialTemperatureServer διαβάζει την θερμοκρασία και την εμφανίζει σε πλαίσιο κειμένου.

Άσκηση 2.4

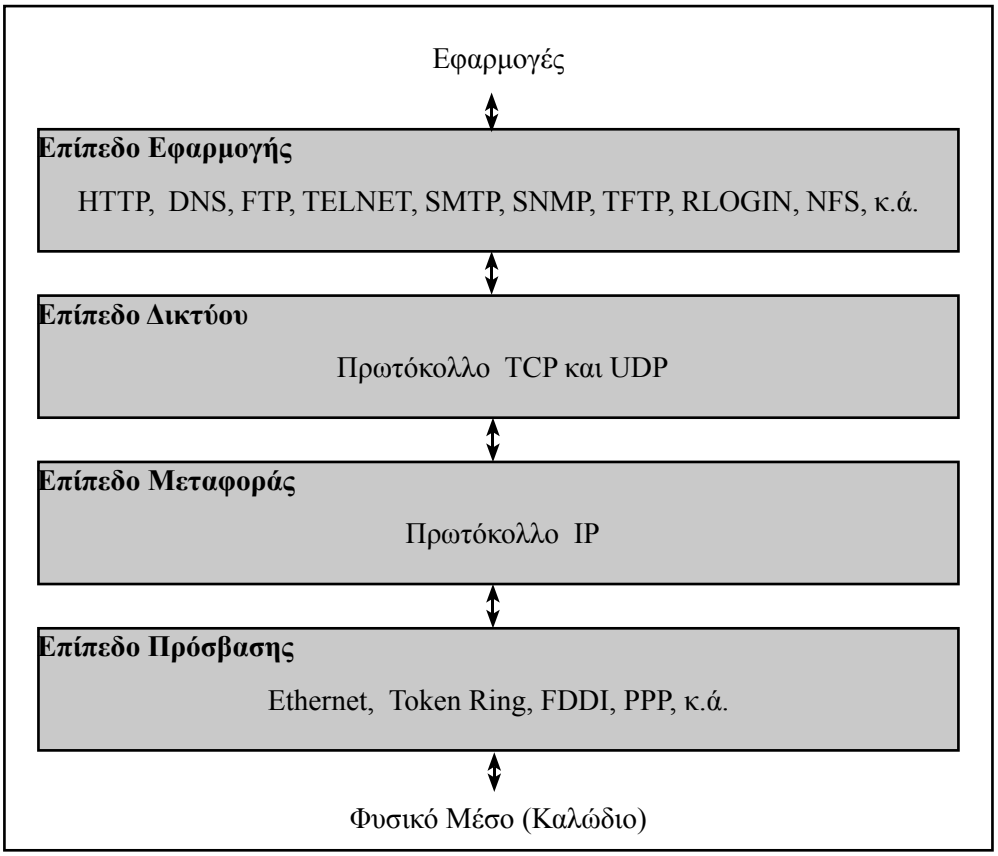
Δημιουργήστε νέα εφαρμογή με όνομα SerialBytesComms η οποία είναι παρόμοια με την εφαρμογή Serial Comms, αλλά διαφέρει στο ότι μεταδίδει ένα πίνακα από bytes αντί για κείμενο. Το πλήκτρο "Αποστολή" αποστέλλει ένα πίνακα από bytes τους οποίους έχει πληκτρολογήσει ο χρήστης στο πλαίσιο κειμένου με τίτλο μήνυμα προς αποστολή χωρισμένους μεταξύ τους με κόμμα (π.χ. 1, 2, 3, 4, 5).

Υπόδειξη για την μετατροπή κειμένου σε πίνακα από bytes: Χρησιμοποιήστε αρχικά την μέθοδο TextBoxDataToSend.Text.Split(",") για να μετατρέψετε το κείμενο TextBoxDataToSend.Text σε πίνακα κειμένου. Η μέθοδος Split της κλάσης String δέχεται ως παράμετρο έναν χαρακτήρα βάση του οποίου γίνεται η κατάτμηση του αρχικού κειμένου σε πίνακα. Στην συνέχεια δημιουργήστε έναν νέο πίνακα από bytes στον οποίο θα μετατρέψετε και αποθηκεύσετε τις τιμές του πίνακα κειμένου που επέστρεψε η μέθοδος Split.

3. Τα πρωτόκολλα UDP και TCP

3.1 Η αρχιτεκτονική του δικτύων TCP/IP

Η αρχιτεκτονική των δικτύων TCP/IP (ή του διαδικτύου) όπως φαίνεται στο σχήμα 16.1 οργανώνεται σε τέσσερα επίπεδα:



Εικόνα 3.1 Η αρχιτεκτονική των δικτύων TCP/IP

- Το επίπεδο πρόσβασης έχει ως κύρια λειτουργία τη μετάδοση πακέτων μεταξύ συγκεκριμένων κόμβων του δικτύου. Το επίπεδο αυτό περιλαμβάνει τα πρωτόκολλα Ethernet, Token Ring, FDDI, PPP κ.ά. και αντιστοιχεί στο φυσικό επίπεδο και το επίπεδο σύνδεσης δεδομένων του μοντέλου αναφοράς OSI.
- Το επίπεδο δικτύου περιλαμβάνει το πρωτόκολλο IP (Internet Protocol). Το πρωτόκολλο IP ελέγχει τη διευθυνσιοδότηση των κόμβων του δικτύου και τη δρομολόγηση των πακέτων. Το πρωτόκολλο IP διασυνδέει δίκτυα με διαφορετικές τεχνολογίες σε ένα ενιαίο λογικό διαδίκτυο.

- Το επίπεδο μεταφοράς περιλαμβάνει τα πρωτόκολλα TCP (Transmission Control Protocol – πρωτόκολλο ελέγχου μετάδοσης) και UDP (User Datagram Protocol – πρωτόκολλο αυτοδύναμων πακέτων χρήστη). Το επίπεδο μεταφοράς ελέγχει την ανταλλαγή των πακέτων και ρυθμίζει την επικοινωνία μεταξύ των τερματικών κόμβων του δικτύου.
- Το επίπεδο εφαρμογής. Το επίπεδο εφαρμογής αντιστοιχεί στα επίπεδα Συνόδου, Παρουσίασης και Εφαρμογής του μοντέλου αναφοράς OSI (τα τρία υψηλότερα επίπεδα). Το επίπεδο αυτό εξασφαλίζει την διαλειτουργικότητα των διαδικτυακών εφαρμογών και περιλαμβάνει μια μεγάλη ποικιλία πρωτοκόλλων εφαρμογής. Τα πρωτόκολλα εφαρμογής χρησιμοποιούν τις υπηρεσίες του TCP ή του UDP για την μετάδοση δεδομένων. Τα πιο γνωστά από τα πρωτόκολλα αυτά είναι τα παρακάτω:
  - HTTP (HyperText Transfer Protocol – πρωτόκολλο μεταφοράς υπερκειμένου),
  - DNS (Domain Name System – σύστημα ονομασίας περιοχών),
  - FTP (File Transfer Protocol – πρωτόκολλο μεταφοράς αρχείου),
  - TELNET (πρωτόκολλο πρόσβασης σε απομακρυσμένο υπολογιστή),
  - SMTP (Simple Mail Transfer Protocol – απλό πρωτόκολλο μεταφοράς ταχυδρομείου)

Στο κεφάλαιο αυτό εξετάζεται η επικοινωνία δεδομένων με τα πρωτόκολλα επικοινωνίας TCP και UDP και δίνονται παραδείγματα εφαρμογών για την επικοινωνία με την χρήση των πρωτοκόλλων αυτών.

3.2 Το πρωτόκολλο UDP

Τα όνομα του πρωτοκόλλου UDP είναι συντομογραφία του User Datagram Protocol. Μία εναλλακτική προέλευση της συντομογραφίας είναι από την ονομασία Universal Datagram Protocol. Με την χρήση του πρωτοκόλλου UDP γίνεται ή αποστολή μηνυμάτων (γνωστών και ως datagrams) σε ένα τοπικό δίκτυο υπολογιστών ή στο διαδίκτυο.

Το πρωτόκολλο αυτό σε σχέση με το πρωτόκολλο TCP είναι απλό, γρήγορο και αποτελεσματικό επειδή δεν διαθέτει μηχανισμούς εγγύησης της αξιοπιστίας των επικοινωνιών, δεν απαιτεί σύνδεση των κόμβων του δικτύου οι οποίοι επικοινωνούν μεταξύ τους και δεν έχει την δυνατότητα τεμαχισμού των δεδομένων.

Το πρωτόκολλο UDP χρησιμοποιείται κυρίως από εφαρμογές ,όπως video και audio streaming, για τις οποίες είναι πολύ σημαντικό τα πακέτα να παραδοθούν άμεσα, χωρίς να υπάρχει διακοπή στην ροή των δεδομένων. Οι εφαρμογές αυτές διαθέτουν δικούς τους μηχανισμούς τεμαχισμού, ελέγχου και διόρθωσης των δεδομένων.

Το πρωτόκολλο UDP υποστηρίζει επίσης την αποστολή ενός πακέτου σε πολλαπλούς υπολογιστές ταυτόχρονα (multicasting) ή σε όλους τους υπολογιστές ενός δικτύου (broadcasting). Με την δυνατότητα αυτή είναι δυνατόν μία ροή ήχου ή εικόνας να μεταδίδεται ταυτόχρονα σε πολλούς συνδρομητές.

Γνωστές εφαρμογές οι οποίες χρησιμοποιούν πακέτα UDP είναι οι παρακάτω:
- Domain Name System (DNS)
- Routing Information Protocol (RIP)
- Simple Network Management Protocol (SNMP)
- Dynamic Host Configuration Protocol (DHCP)
- IPTV

- Voice over IP (VoIP)
- Trivial File Transfer Protocol (TFTP)

Το πρωτόκολλο UDP περιγράφεται αναλυτικά στο πρότυπο IETF RFC 768.

Η Δομή ενός πακέτου UDP

Κάθε πακέτο UDP ξεκινά με μια επικεφαλίδα (header) 64 bits η οποία περιλαμβάνει:
- Την θύρα (source port) από την οποία προήλθε το πακέτο. Η πληροφορία αυτή είναι προαιρετική. Η τιμή 0 δηλώνει ότι δεν έχει καθορισθεί η θύρα αυτή από τον αποστολέα.
- Την θύρα του παραλήπτη (Destination Port) όπου θα παραδοθεί το πακέτο
- Το πλήθος των bytes (Length) τα οποία θα αποσταλούν.
- Το Checksum το οποίο χρησιμοποιείται από τον παραλήπτη για ελέγξει την ορθότητα των δεδομένων που παρέλαβε. Το Checksum είναι προαιρετικό.

Bits 0 - 15	Bits 16 - 31	Bits 32 - 47	Bits 48 - 63	Bits 64 ...
Source Port (2 Bytes)	Destination Port (2 Bytes)	Length (2 Bytes)	Checksum (2 Bytes)	Data

Εικόνα 3.2 Η δομή ενός πακέτου UDP

Μετά την επικεφαλίδα ακολουθούν τα bytes των δεδομένων τα οποία αποστέλλονται.

Τα πρωτόκολλα TCP και UDP χρησιμοποιούν τις TCP θύρες (ports) για αποστολή και παραλαβή δεδομένων. Η αποστολή και παραλαβή δεδομένων χαρακτηρίζονται από την διεύθυνση του αποστολέα/παραλήπτη και την θύρα αποστολής/παραλαβής. Οι θύρες αριθμούνται από 0 έως 65535 και τα δεδομένα τα οποία αποστέλλονται ή παραλαμβάνονται ανήκουν σε κάποιον συγκεκριμένο αριθμό θύρας. Με την χρήση των θυρών επικοινωνίας, οι εφαρμογές ενός Η/Υ μπορούν ταυτόχρονα να αποστέλλουν και παραλαμβάνουν δεδομένα κάνοντας χρήση διαφορετικών θυρών.

3.3 Το πρωτόκολλο TCP

Το πρωτόκολλο TCP (Transmission Control Protocol) έχει σχεδιαστεί για να παράσχει αξιόπιστη αμφίδρομη επικοινωνία μεταξύ των κόμβων ενός δικτύου, υλοποιώντας μεταξύ τους ένα ιδεατό κύκλωμα. Το πρωτόκολλο TCP διενεργεί σύνθετους ελέγχους σφαλμάτων και ροής στα μεταδιδόμενα πακέτα

Το πρωτόκολλο TCP χωρίζει τα δεδομένα προς μετάδοση σε μικρότερα πακέτα τα οποία ονομάζονται τμήματα ή segments. Το τμήματα αποτελούν την μονάδα μεταφοράς στο πρωτόκολλο TCP.

Η Δομή ενός τμήματος TCP

Κάθε τμήμα αποτελείται από την Επικεφαλίδα (Header) και τα προς μετάδοση Δεδομένα (Data). Τα βασικότερα πεδία της επικεφαλίδας είναι:

- Οι Θύρες (ports) TCP αφετηρίας και προορισμού.
- Ο Αριθμός Σειράς (Sequence Number). Το κάθε τμήμα έχει τον δικό του αύξοντα αριθμό σειράς. Ο αριθμός σειράς δηλώνει σε ποια θέση πρέπει να μπει το συγκεκριμένο τμήμα από τον παραλήπτη ώστε να αναπαραχθούν τα δεδομένα με την σωστή σειρά.
- Ο Αριθμός Επιβεβαίωσης (Acknowledgment number). Ο αριθμός επιβεβαίωσης χρησιμοποιείται από τον παραλήπτη για την επιβεβαίωση της παραλαβής των τμημάτων της μετάδοσης. Αν ο αποστολέας δεν λάβει επιβεβαίωση μέσα σε ένα εύλογο χρονικό διάστημα, θα επαναλάβει τη μετάδοση των δεδομένων.
- Το Παράθυρο (Window Size). Το παράθυρο χρησιμοποιείται για τον έλεγχο της ροής δεδομένων. Ο παραλήπτης χρησιμοποιεί το πεδίο αυτό για να δηλώσει στον αποστολέα το πλήθος των δεδομένων τα οποία μπορεί να δεχθεί.

<table>
<tr><th colspan="4">Byte 0</th><th colspan="4">Byte 1</th><th colspan="4">Byte 2</th><th>Byte 3</th></tr>
<tr><td colspan="8">Source Port</td><td colspan="4">Destination Port</td></tr>
<tr><td colspan="12">Sequence Number</td></tr>
<tr><td colspan="12">Acknowledgment number</td></tr>
<tr><td>Data Offset</td><td>Reserved</td><td>NS</td><td>CWR</td><td>ECE</td><td>URG</td><td>ACK</td><td>PSH</td><td>RST</td><td>SYN</td><td>FIN</td><td>Window Size</td></tr>
<tr><td colspan="8">Checksum</td><td colspan="4">Urgent pointer</td></tr>
<tr><td colspan="12">Options</td></tr>
</table>

Εικόνα 3.3 Η δομή μιας επικεφαλίδας τμήματος TCP

Η μετάδοση δεδομένων με το πρωτόκολλο TCP χωρίζεται σε τρεις φάσεις:

- Την σύνδεση των άκρων τα οποία θα μεταδώσουν δεδομένα. Πριν γίνει οποιαδήποτε μεταφορά δεδομένων, πρέπει πρώτα να γίνει μια νοητή σύνδεση μεταξύ του αποστολέα και του παραλήπτη ή των δύο άκρων τα οποία θα ανταλλάξουν.
- Την μεταφορά των δεδομένων.
- Την αποσύνδεση των δύο άκρων που επικοινωνούν και την ελευθέρωση όλων των πόρων (μνήμη, θύρες κλπ) οι οποίοι χρησιμοποιήθηκαν κατά την μεταφορά δεδομένων.

3.4 Αναφορές

- Μανώλης Κιαγιάς, "Δίκτυα Υπολογιστών ΙΙ – Το Ανεπίσημο Βοήθημα", http://www.freebsdworld.gr/diktia/theBookII.pdf
- Γ. Φούσκας, "Δίκτυα Υπολογιστών Ι", Ελληνικό Ανοικτό Πανεπιστήμιο, ISBN: 960-538-468-X.
- RFC 793, TRANSMISSION CONTROL PROTOCOL
- RFC 768, UDP
- UDP, http://el.wikipedia.org/wiki/UDP

- TCP, http://el.wikipedia.org/wiki/TCP
- Κατάλογος των TCP και UDP ports , http://el.wikipedia.org/wiki/Κατάλογος_των_TCP_και_UDP_ports
- TCP and UDP, http://www.cse.wustl.edu/~jain/cis677-96/ftp/e_gtcp2.pdf

4. Επικοινωνίες UDP και TCP με Visual Basic

4.1 Επικοινωνίες UDP με την Visual Basic

Οι επικοινωνίες UDP με την Visual Basic γίνεται με την χρήση της κλάσης UdpClient. Για την χρήση της κλάσης αυτής πρέπει να χρησιμοποιηθεί η δήλωση:

```
Imports System.Net.Sockets
```

Οι κυριότερες ιδιότητες, μέθοδοι και συμβάντα της κλάσης αυτής είναι:

Μέθοδος	Περιγραφή
New ή New(Port as Integer)	Η μέθοδος αυτή δημιουργεί ένα νέο αντικείμενο της κλάσης αυτής. πχ. Dim MyUdpClient As New UdpClient Για την παραλαβή δεδομένων πρέπει να δοθεί ώς παράμετρος και η τοπική θύρα η οποία θα χρησιμοποιηθεί για την λήψη δεδομένων. π.χ. Dim MyUdpClient = New System.Net.Sockets.UdpClient(1000)
Send(buffer() as Byte, length as Integer, hostname as String, port as Integer)	Αποστέλλει έναν πίνακα Bytes στην θύρα και την IP διεύθυνση οι οποίες καθορίζονται από τις παραμέτρους port και hostname. Η παράμετρος length καθορίζει το πλήθος των bytes τα οποία θα αποσταλούν. π.χ. MyUdpClient.Send(buufer, 50, "192.168.1.2", 1000)
Close()	Κλείνει το αντικείμενο UdpClient
Receive(aRemotePoint as IpEndPoint)	Διαβάζει και επιστρέφει έναν πίνακα από bytes από την διεύθυνση και θύρα η οποία καθορίζεται από την παράμετρο aRemotePoint. π.χ. για λήψη δεδομένων τα οποία μπορούν να προέρχονται από οποιαδήποτε διεύθυνση και θύρα μπορούμε να χρησιμοποιήσουμε τις παρακάτω εντολές: Dim aRemotePoint As New IPEndPoint(IPAddress.Any, 0) Dim receiveBytes As Byte() = MyUdpClient.Receive(aRemotePoint)
Close()	Κλείνει το αντικείμενο UdpClient

Οι επικοινωνίες UDP και TCP μεταδίδουν πίνακες από bytes. Για την αποστολή και λήψη κειμένου, πρέπει να γίνει η μετατροπή του κειμένου σε πίνακα bytes και αντίστροφα. Η μετατροπή γίνεται με την χρήση της κλάσης ASCII και τις μεθόδους GetBytes και GetString.

Στο παρακάτω παράδειγμα γίνεται η μετατροπή κειμένου (μεταβλητή aMessage) σε bytes με όνομα buffer.

```
Dim buffer() as Byte = Encoding.ASCII.GetBytes(aMessage)
```

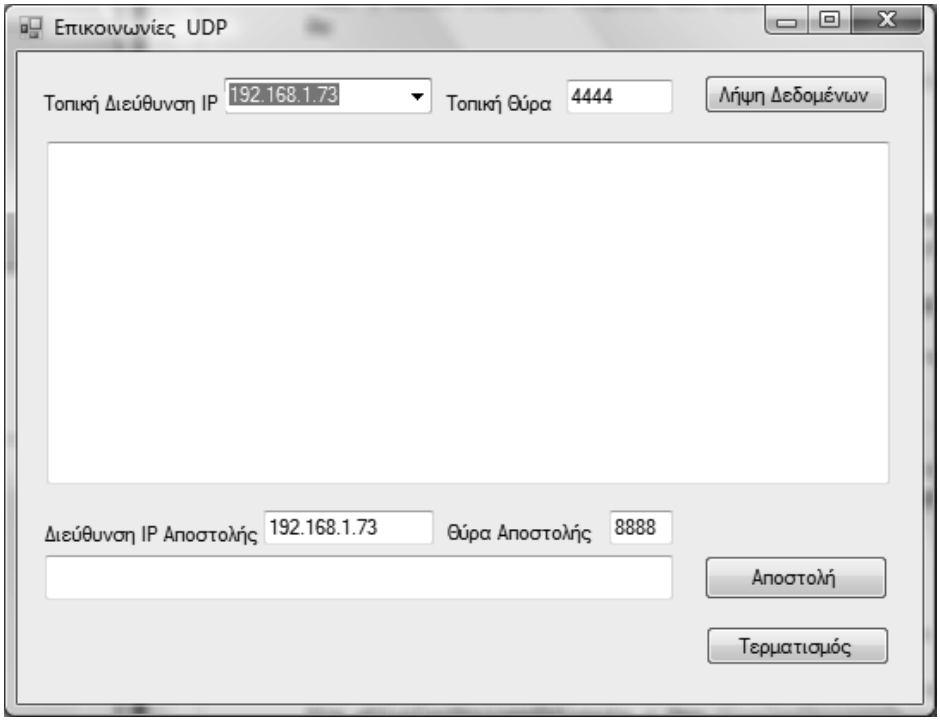
Η αντίστροφη μετατροπή του πίνακα bytes (μεταβλητή buffer) σε κείμενο (μεταβλητή aMessage) γίνεται με το παρακάτω παράδειγμα.

```
Dim aMessage As String = System.Text.Encoding.ASCII.GetString(buffer)
```

Σε περίπτωση που το κείμενο για μετατροπή δεν είναι σε κωδικοποίηση ASCII αλλά Unicode, UTF7, UTF8 ή UTF32, τότε αντί της κλάσης ASCII χρησιμοποιούνται οι κλάσεις Unicode, UTF7, UTF8 και UTF32 αντίστοιχα.

Εφαρμογή για Επικοινωνίες UDP

Η παραθυρική εφαρμογή η οποία φαίνεται στο σχήμα 16.1 μπορεί να αποστείλει και παραλάβει μηνύματα κειμένου με την χρήση επικοινωνιών UDP.



Εικόνα 4.1 Παραθυρική εφαρμογή για επικοινωνίες UDP

Η εφαρμογή χρησιμοποιεί ένα παράθυρο με όνομα FormUdpComms. Το παράθυρο αυτό έχει τα παρακάτω στοιχεία ελέγχου:

Όνομα	Τύπος	Περιγραφή
ComboBoxLocalIP	ComboBox	Χρησιμοποιείται για να εμφανίσει όλες τις διαθέσιμες IP διευθύνσεις του Η/Υ και να δώσει στον χρήστη την δυνατότητα να επιλέξει την IP διεύθυνση την οποία θα χρησιμοποιήσει
TexbBoxLocalPort	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την τοπική θύρα για την παραλαβή δεδομένων
TexbBoxSendIPPort	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την διεύθυνση IP του παραλήπτη
TexbBoxSendPort	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την θύρα του παραλήπτη
TextBoxReceived	TextBox	Εμφανίζει στον χρήστη τα εισερχόμενα μηνύματα
TextBoxSend	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί το μήνυμα το οποίο θέλει να αποστείλει
ButtonStartReceiving		Πλήκτρο για το διάβασμα δεδομένων
ButtonSend		Πλήκτρο για την αποστολή δεδομένων
ButtonClose		Πλήκτρο για τον τερματισμό της εφαρμογής

Πίνακας 4.1 Τα στοιχεία ελέγχου της εφαρμογής

Η παραθυρική εφαρμογή πρέπει να εκτελείται ταυτόχρονα από δύο χρήστες οι οποίοι είναι συνδεδεμένοι στο διαδίκτυο ή σε τοπικό δίκτυο. Θα πρέπει επίσης, εάν χρησιμοποιείται τείχος προστασίας, να προστεθεί στον κάθε Η/Υ εξαίρεση για την εφαρμογή ή τις UDP θύρες οι οποίες χρησιμοποιούνται.

Η εφαρμογή μπορεί να δοκιμασθεί σε έναν και μόνο Η/Υ από τον ίδιο χρήστη με την παράλληλη εκτέλεση δύο η περισσότερων εφαρμογών για τις επικοινωνίες Udp.

Οι χρήστες των προγραμμάτων επικοινωνίας πρέπει:
- Να επιλέξουν τις κατάλληλες διευθύνσεις και θύρες για την επικοινωνία.
- Να πατήσουν το πλήκτρο "Λήψη δεδομένων" για την λήψη και εμφάνιση των δεδομένων τα οποία παραλαμβάνονται από την επιλεγμένη τοπική θύρα.
- Να πληκτρολογήσουν κάποιο μήνυμα και να πατήσουν στην συνέχεια το πλήκτρο "Αποστολή" για να αποστείλουν το μήνυμα στην επιλεγμένη διεύθυνση IP και θύρα του παραλήπτη.
- Να τερματίσουν την επικοινωνία τους και να κλείσουν την εφαρμογή με το πάτημα του πλήκτρου "Τερματισμός"

Ο κώδικας Visual Basic του παραθύρου της εφαρμογής FormUdpComms για τις επικοινωνίες UDP δίνεται παρακάτω.

```
Imports System.Net.Sockets
```

```
Imports System.Net
Imports System.Text

Public Class FormUdpComms

    'Ορισμός αντικειμένου της κλάσης UDPClient για την επικοινωνία
    Dim MyUdpClient As New UdpClient

    'Ορισμός νήματος για την παραλαβή δεδομένων
    Public ThreadReceive As System.Threading.Thread

    Private Sub FormUdpComms_Load(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles MyBase.Load
        'Εμφάνιση όλων των IP διευθύνσεων και εμφάνιση τους
        'στο ComboBoxLocalIP

        'Εύρεση του ονόματος του Η/Υ με την χρήση της μεθόδου Dns.GetHostName
        Dim localHost As String
        localHost = Dns.GetHostName()

        'Εύρεση όλων των τοπικών διευθύνσεων με την χρήση της κλάσης Dns
        Dim IPHEntry = Dns.GetHostEntry(localHost)
        Dim IPAdd = IPHEntry.AddressList
        Me.ComboBoxLocalIp.Items.Clear()

        'Πρόσθεση διευθύνσεων στο ComboBoxLocalIp
        For i As Integer = 0 To IPAdd.GetUpperBound(0)
            Me.ComboBoxLocalIp.Items.Add(IPAdd(i).ToString)
        Next

        'Επιλογή πρώτης διεύθυνσης Ip για αποστολή και λήψη
        If Me.ComboBoxLocalIp.Items.Count > 0 Then
            Me.ComboBoxLocalIp.SelectedIndex = 0
            Me.TextBoxSendIP.Text = IPAdd(0).ToString
        End If
    End Sub

    Private Sub ButtonSend_Click(ByVal sender As System.Object,
                  ByVal e As System.EventArgs) Handles ButtonSend.Click
        'Αποστολή δεδομένων στην επιλεγμένη Ip διεύθυνση και θύρα
        Try
            'Μετατροπή του κειμένου του TextBoxSend σε πίνακα Bytes
            Dim bytesToSend = Encoding.ASCII.GetBytes(Me.TextBoxSend.Text)
            'Αποστολή δεδομένων
            Dim pRet = MyUdpClient.Send(bytesToSend, bytesToSend.Length, _
                        Me.TextBoxSendIP.Text, Me.TextBoxSendPort.Text)
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try

    End Sub

    Private Sub ButtonStartReceiving_Click(ByVal sender As System.Object,
           ByVal e As System.EventArgs) Handles ButtonStartReceiving.Click
        'Δημιουργία νήματος το οποίο δέχεται διαρκώς και εμφανίζει δεδομένα
        Try
            'Συσχέτιση UdpClient με την επιλεγμένη τοπική θύρα
            ' για την λήψη δεδομένων
            MyUdpClient = New System.Net.Sockets.UdpClient(
                                        CInt(Me.TextBoxLocalPort.Text))
            'Δημιουργία νήματος για την λήψη δεδομένων
            ThreadReceive = New System.Threading.Thread(
                                              AddressOf ReceiveMessages)
            'Έναρξη εκτέλεσης του νήματος
```

```
            ThreadReceive.Start()
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
    End Sub

    Public Sub ReceiveMessages()
        'Μέθοδος η οποία καλείται από το νήμα για την λήψη δεδομένων
        'και η οποία διαβάζει διαρκώς και εμφανίζει δεδομένα
        Do

            'Ορισμός απομακρύσμένου σημείου για την ληψη δεδομένων
            Dim aRemotePoint As New System.Net.IPEndPoint(
                                            System.Net.IPAddress.Any, 0)
            'Ανάγνωση δεδομένων τα οποία προέρχονται
            'από το απομακρυσμένο σημείο
            Dim receiveBytes As Byte() = MyUdpClient.Receive(aRemotePoint)
            'Μετατροπή του πίνακα από Bytes ο οποίος παρελήφθη σε κείμενο
            Dim receivedText As String = _
                        System.Text.Encoding.ASCII.GetString(receiveBytes)
            'Εμφάνιση κειμένου
            DisplayMessage(aRemotePoint.ToString + " : " + receivedText)
        Loop
    End Sub

    'Ορισμός αναφοράς για την κλήση της μεθόδου DisplayMessageproc
    Public Delegate Sub DisplayMessageDelegate(ByVal aMessage As String)


    Private Sub DisplayMessage(ByVal aMessage As String)
        'Μέθοδος για την εμφάνιση δεδομένων
        'Η μέθοδος καλείται από το νήμα για την λήψη δεδομένων

        'Δημιουργία αναφοράς στην μέθοδο DisplayMessageProc
        Dim aDisplayMessageDelegate = New DisplayMessageDelegate(
                                            AddressOf DisplayMessageProc)
        'Κλήση της αναφοράς
        Me.Invoke(aDisplayMessageDelegate, aMessage)
    End Sub

    Private Sub DisplayMessageProc(ByVal aMessage As String)
        'Μέθοδος η οποία εμφανίζει το κείμενο

        Me.TextBoxReceived.AppendText(aMessage + vbCrLf)
        Me.TextBoxReceived.ScrollToCaret()
    End Sub

    Private Sub FormUdpComms_FormClosing(ByVal sender As System.Object,
                   ByVal e As System.Windows.Forms.FormClosingEventArgs) _
                                            Handles MyBase.FormClosing
        Try
            'Έλεγχος αν το νήμα για την λήψη δεδομένων εκτελείται
            If Not IsNothing(ThreadReceive) Then
                'Τερματισμός του Νήματος
                ThreadReceive.Abort()
            End If
            'Κλείσιμο του αντικειμένου για την UDP επικοινωνία
            Me.MyUdpClient.Close()
        Catch ex As Exception

        End Try
    End Sub

    Private Sub ButtonClose_Click(ByVal sender As System.Object,
```

```
                    ByVal e As System.EventArgs) Handles ButtonClose.Click
        Me.Close()
    End Sub
End Class
```

4.2 Επικοινωνίες TCP με την Visual Basic

Οι επικοινωνίες TCP γίνονται με την χρήση των κλάσεων TCPClient και TCPListener. Για την χρήση των κλάσεων αυτών, όπως και για τις επικοινωνίες UDP, πρέπει να χρησιμοποιηθεί η δήλωση:

```
Imports System.Net.Sockets
```

Για την επικοινωνία με το πρωτόκολλο TCP, είναι απαραίτητη η δημιουργία συνδέσεων και η χρήση δύο διαφορετικών τύπων εφαρμογών. Η μία εφαρμογή λειτουργεί ώς εξυπηρετητής (TCP Server) και η άλλη ώς πελάτης (TCP Client).

Η εφαρμογή του εξυπηρετητή χρησιμοποιεί την κλάση TCPListener για να μπορεί να δέχεται αιτήματα σύνδεσης από την εφαρμογή πελάτη. Και οι δύο εφαρμογές χρησιμοποιούν την κλάση TCPClient για την αποστολή και λήψη δεδομένων με το πρωτόκολλο TCP.

Σε αντίθεση με τις επικοινωνίες UDP οι οποίες χρησιμοποιούν πίνακες bytes για την αποστολή και λήψη δεδομένων, η κλάση TCPClient χρησιμοποιεί αντικείμενα της κλάσης NetworkStream για την αποστολή και λήψη δεδομένων. Η κλάση NetworkStream χρησιμοποιείται ως ενδιάμεση κλάση για να στείλει η παραλάβει πίνακες από bytes.

Η κλάση TCPListener

Οι κυριότερες ιδιότητες και μέθοδοι της κλάσης αυτής είναι:

Ιδιότητες	Περιγραφή
LocalEndpoint	Επιστρέφει ένα αντικείμενο τύπου EndPoint το οποίο περιγράφει την τοπική διεύθυνση και την τοπική θύρα την οποία χρησιμοποιεί το αντικείμενο TCPListener π.χ. Console.WriteLine(aTcpListener.LocalEndpoint.ToString)
Μέθοδοι	**Περιγραφή**
New(address as IPAdress. Port as Integer)	Η μέθοδος αυτή δημιουργεί ένα νέο αντικείμενο της κλάσης αυτής. Η παράμετρος address καθορίζει την τοπική διεύθυνση IP και η παράμετρος port την τοπική θύρα η οποία χρησιμοποιείται για την επικοινωνία. π.χ. Dim aListener = New TcpListener(IPAddress.Parse("192.168.1.2"), 1000)

Start()	Ενεργοποιεί το αντικείμενο TCPListener και δεσμεύει την τοπική θύρα για τις επικοινωνίες
AcceptTcpClient()	Η μέθοδος αυτή περιμένει αιτήματα σύνδεσης επικοινωνίας TCP, έως ότου κάποιος πελάτης στείλει ένα αίτημα για να συνδεθεί. Όταν παραληφθεί το σχετικό αίτημα, τότε η μέθοδος επιστρέφει ένα αντικείμενο τύπου TCPClient, το οποίο χρησιμοποιείται για την αποστολή και λήψη μηνυμάτων με το πελάτη ο οποίος έκανε την σύνδεση. π.χ. Dim aTcpClient As TcpClient = MyTcpListener.AcceptTcpClient
Stop()	Το αντικείμενο TCPListener σταματά να δέχεται αιτήματα για σύνδεση και η τοπική θύρα αποδεσμεύεται.

Η κλάση TCPClient

Οι κυριότερες ιδιότητες και μέθοδοι της κλάσης αυτής είναι:

Ιδιότητες	Περιγραφή
ReceiveBufferSize	Καθορίζει ή επιστρέφει το μέγεθος του προσωρινού χώρου για την λήψη δεδομένων
SendBufferSize	Καθορίζει ή επιστρέφει το μέγεθος του προσωρινού χώρου για την αποστολή δεδομένων
Client Client.RemoteEndPoint	Η ιδιότητα Client επιστρέφει ένα αντικείμενο τύπου Socket. Μία από τις ιδιότητες αυτού του αντικειμένου είναι η RemoteEndPoint η οποία μας επιτρέφει την διεύθυνση IP και την θύρα την οποία χρησιμοποιεί ο πελάτης TCP. π.χ. Console.WriteLine(aTcpClient.Client.RemoteEndPoint.ToString)
Μέθοδοι	**Περιγραφή**
New	Η μέθοδος αυτή δημιουργεί ένα νέο αντικείμενο TCPClient π.χ. Dim MyTcpClient As New TcpClient
Connect(address as IPAdress, Port as Integer	Η μέθοδος αυτή στέλνει αίτημα σύνδεσης στον εξυπηρετητή TCP του οποίου η διεύθυνση καθορίζεται από την παράμετρο IPAddress και η θύρα του από την παράμετρο port. π.χ. MyTcpClient.Connect(IPAddress.Parse("192.168.1.2"), 1000)
GetStream()	Η μέθοδος αυτή επιστρέφει ένα αντικείμενο τύπου NetworkStream το οποίο χρησιμοποιείται για την αποστολή και λήψη δεδομένων.
Close()	Αποσυνδέει το αντικείμενο TCPClient.

Η κλάση NetworkStream

Οι κυριότερες ιδιότητες και μέθοδοι της κλάσης αυτής είναι:

Μέθοδοι	Περιγραφή
Read(buffer as Byte, offset as Integer, size as Integer)	Η μέθοδος διαβάζει και επιστρέφει έναν πίνακα από bytes. Τα δεδομένα επιστρέφονται στην παράμετρο buffer. Η παράμετρος offset καθορίζει το δείκτη αρχής και η παράμετρος size καθορίζει το πλήθος των bytes τα οποία θα διαβαστούν. π.χ. aReadStream.Read(buffer, 0, MyTcpClient.ReceiveBufferSize)
Write(buffer as Byte, offset as Integer, size as Integer)	Η μέθοδος αποστέλλει έναν πίνακα από bytes. Ο πίνακας bytes ο οποίος θα αποσταλεί καθορίζεται από την παράμετρο buffer, η παράμετρος offset καθορίζει το δείκτη αρχής του πίνακα και η παράμετρος size καθορίζει το πλήθος των στοιχείων του πίνακα τα οποία θα αποσταλούν. π.χ. aWriteStream.Write(buffer, 0, buffer.Length)

4.3 Εφαρμογές εξυπηρετητή και πελάτη TCP

Εφαρμογή Εξυπηρετητή TCP

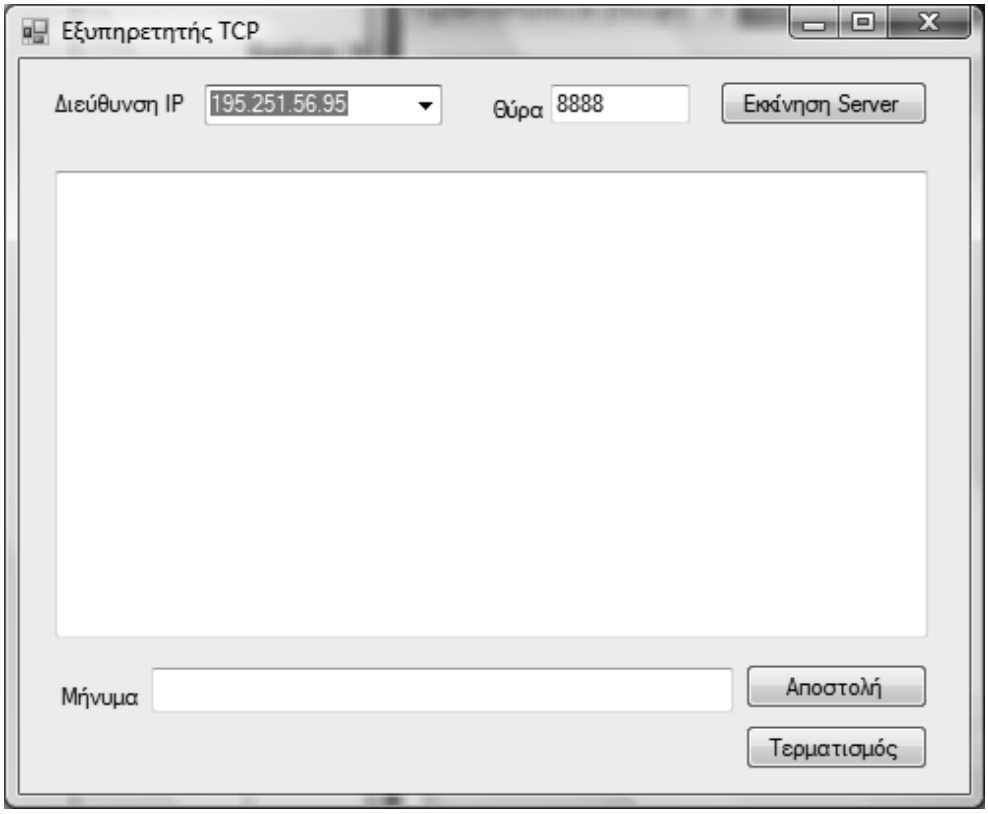
Η παραθυρική εφαρμογή η οποία φαίνεται στο σχήμα 16.5 λειτουργεί ώς εξυπηρετητής TCP. Η εφαρμογή χρησιμοποιεί ένα παράθυρο με όνομα TCPServerForm.

Το παράθυρο αυτό έχει τα παρακάτω στοιχεία ελέγχου:

Όνομα	Τύπος	Περιγραφή
ComboBoxLocalIP	ComboBox	Χρησιμοποιείται για να εμφανίσει όλες τις διαθέσιμες IP διευθύνσεις του Η/Υ και να δώσει στον χρήστη την δυνατότητα να επιλέξει την IP διεύθυνση την οποία θα χρησιμοποιήσει
TextBoxLocalPort	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την τοπική θύρα την οποία χρησιμοποιεί ο εξυπηρετητής για την σύνδεση πελατών και την επικοινωνία
TextBoxReceived	TextBox	Εμφανίζει στον χρήστη τα εισερχόμενα μηνύματα
TextBoxSend	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί το μήνυμα το οποίο θέλει να αποστείλει
ButtonStart		Πλήκτρο για την εκκίνηση του εξυπηρετητή

ButtonSend		Πλήκτρο για την αποστολή δεδομένων
ButtonClose		Πλήκτρο για τον τερματισμό της εφαρμογής

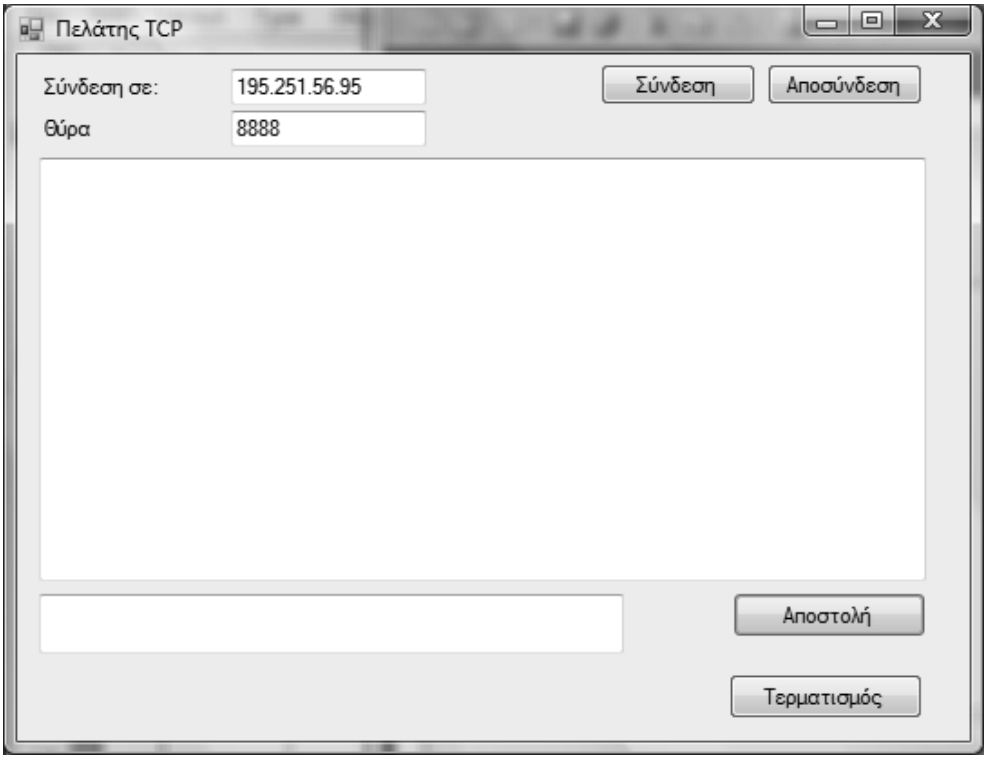
Πίνακας 4.2 Τα στοιχεία ελέγχου του εξυπηρετητή TCP



Εικόνα 4.2 Παραθυρική εφαρμογή εξυπηρετητή TCP

Εφαρμογή Πελάτη TCP

Η παραθυρική εφαρμογή η οποία φαίνεται στην εικόνα 16.6 λειτουργεί ώς πελάτης TCP. Η εφαρμογή χρησιμοποιεί ένα παράθυρο με όνομα TCPClientForm.



Εικόνα 4.3 Παραθυρική εφαρμογή πελάτη TCP

Το παράθυρο αυτό έχει τα παρακάτω στοιχεία ελέγχου:

Όνομα	Τύπος	Περιγραφή
TextBoxIP	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την IP διεύθυνση του εξυπηρετητή στον οποίο θέλει να συνδεθεί.
TextBoxPort	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την θύρα του εξυπηρετητή στον οποίο θέλει να συνδεθεί.
TextBoxReceived	TextBox	Εμφανίζει στον χρήστη τα εισερχόμενα μηνύματα
TextBoxSend	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί το μήνυμα το οποίο θέλει να αποστείλει
ButtonConnect	Button	Πλήκτρο για την σύνδεση με τον εξυπηρετητή
ButtonDisconnect	Button	Πλήκτρο για την αποσύνδεση με τον εξυπηρετητή
ButtonSend	Button	Πλήκτρο για την αποστολή δεδομένων

ButtonClose	Button	Πλήκτρο για τον τερματισμό της εφαρμογής

Πίνακας 4.3 Τα στοιχεία ελέγχου του πελάτη TCP

Οι εφαρμογές του πελάτη και του εξυπηρετητή μπορούν να εκτελούνται στον ίδιο ή διαφορετικούς Η/Υ συνδεδεμένους σε τοπικό δίκτυο ή το διαδίκτυο. Τα βήματα για την σύνδεση και την επικοινωνία με το πρωτόκολλο TCP είναι τα παρακάτω:

- Γίνεται εκκίνηση του εξυπηρετητή TCP και στην συνέχεια με το πάτημα του πλήκτρου "Εκκίνηση Server", ο εξυπηρετητής μπαίνει σε κατάσταση αναμονής νέων συνδέσεων.
- Γίνεται εκκίνηση της εφαρμογής πελάτη TCP, επιλέγεται η διεύθυνση και θύρα του εξυπηρετητή και με το πάτημα του πλήκτρου "Σύνδεση" γίνεται σύνδεση με τον εξυπηρετητή. Αν η σύνδεση είναι επιτυχής, τότε εμφανίζονται μηνύματα σύνδεσης τόσο στον πελάτη όσο και στον εξυπηρετητή.
- Ο χρήστης του πελάτη ή του εξυπηρετητή πληκτρολογεί κάποιο μήνυμα και το αποστέλλει με το πάτημα του πλήκτρο "Αποστολή". Το μήνυμα αποστέλλεται και εμφανίζεται στον εξυπηρετητή ή τον πελάτη TCP.
- Με το πλήκτρο "Αποσύνδεση" ο πελάτης αποσυνδέεται από τον εξυπηρετητή.
- Ο εξυπηρετητής εμφανίζει μήνυμα ότι ο πελάτης αποσυνδέθηκε. Με το πάτημα του πλήκτρου "Εκκίνηση Server", ο εξυπηρετητής μπαίνει να μπει εκ νέου σε κατάσταση αναμονής νέων συνδέσεων ή με τα πλήκτρα "Τερματισμός" μπορεί να γίνει ο τερματισμός των εφαρμογών.

Ο κώδικας Visual Basic του παραθύρου του εξυπηρετητή TCP δίνεται παρακάτω.

```
Imports System.Net.Sockets
Imports System.Net
Imports System.Text

Public Class TcpServerForm
    'Ορισμός αντικειμένου TCP Server για την αποδοχή συνδέσεων TCP
    Dim MyTcpListener As TcpListener
    'Ορισμός αντικειμένου για την επικοινωνία με TCP Client
    Dim MyTcpClient As TcpClient

    Private Sub TcpServerForm_Load(ByVal sender As System.Object,
                   ByVal e As System.EventArgs) Handles MyBase.Load
        'Εμφάνιση όλων των IP διευθύνσεων και εμφάνιση τους
        'στο ComboBoxLocalIP

        'Εύρεση του ονόματος του Η/Υ με την χρήση της μεθόδου Dns.GetHostName
        Dim localHost As String
        localHost = Dns.GetHostName()

        'Εύρεση όλων των τοπικών διευθύνσεων με την χρήση της κλάσης Dns
        Dim anIPHostEntry = Dns.GetHostEntry(localHost)
        Dim anIPAddressArray = anIPHostEntry.AddressList
        Me.ComboBoxLocalIp.Items.Clear()

        'Πρόσθεση διευθύνσεων στο ComboBoxLocalIp
        For i As Integer = 0 To anIPAddressArray.GetUpperBound(0)
            Me.ComboBoxLocalIp.Items.Add(anIPAddressArray(i).ToString)
        Next
        'Επιλογή πρώτης διεύθυνσης IP
        If Me.ComboBoxLocalIp.Items.Count > 0 Then
            Me.ComboBoxLocalIp.SelectedIndex = 0
        End If
```

```
End Sub

Public Sub StartServer()
    If IsNothing(Me.ComboBoxLocalIp.SelectedItem) Then Exit Sub
    Try
        'Εκκίνηση TCP Server για την αποδοχή συνδέσεων
        MyTcpListener = New
            TcpListener(IPAddress.Parse(Me.ComboBoxLocalIp.SelectedItem),
            CInt(TextBoxLocalPort.Text))
        DisplayMessage("Εκκίνηση TCP Server ....")
        MyTcpListener.Start()
        DisplayMessage("Η τοπική διεύθυνση IP και η θύρα είναι:" +
                       MyTcpListener.LocalEndpoint.ToString)
        'Αναμονή σύνδεσης
        DisplayMessage("Αναμονή σύνδεσης.....")
        MyTcpClient = MyTcpListener.AcceptTcpClient
        'Εκτύπωση διεύθυνσης και θύρας IP του πελάτη ο οποίος συνδέθηκε
        DisplayMessage(MyTcpClient.Client.RemoteEndPoint.ToString +
                       ": Σύνδεση ")
        'Εκκίνηση νήματος για την λήψη δεδομένων από τον πελάτη TCP
        Dim aClientTread As Threading.Thread =
               New Threading.Thread(AddressOf ReceiveData)
        aClientTread.Start()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub

Private Sub ReceiveData()
    'Ορίσμός πίνακα Bytes για το διάβασμα δεδομένων
    Dim buffer(10024) As Byte
    Dim aMessage As String
    Do
        Try
            'Ορισμός αντικειμένου NetworkStream για το διάβασμα δεδομένων
            'από το αντικείμενο aTcpClient
            Dim aReadStream As NetworkStream = MyTcpClient.GetStream()
            'Διάβασμα δεδομένων και αποθήκευση τους
            'στο αντικείμενο buffer
            'Στη μεταβλητή bytesRead επιστρέφεται ο αριθμός των bytes
            'τα οποία διαβάσθηκαν
            Dim bytesRead As Integer = aReadStream.Read(buffer, 0,
                                  CInt(MyTcpClient.ReceiveBufferSize))
            'Έλεγχος αν ο Client έχει αποσυνδεθεί
            If bytesRead = 0 Then
                Me.DisplayMessage(
                        MyTcpClient.Client.RemoteEndPoint.ToString +
                        ": Αποσύνδεση")
                'Κλείσιμο TCPListener
                MyTcpListener.Stop()
                'Έξοδος και τερματισμός του νήματος
                Exit Sub
            End If
            'Μετατροπή πίνακα Bytes σε κείμενο ASCII
            aMessage = System.Text.Encoding.ASCII.GetString(buffer)
            'Εμφάνιση μηνύματος
            Me.DisplayMessage(
                        MyTcpClient.Client.RemoteEndPoint.ToString +
                        ": " + aMessage)
        Catch ex As Exception
            'Εμφάνιση μηνύματος λάθους
            DisplayMessage(ex.Message)
            'Κλείσιμο TCPClient
            MyTcpClient.Close()
```

```
                'Έξοδος και τερματισμός του νήματος
                Exit Sub
            End Try
        Loop
    End Sub

    'Ορισμός αναφοράς για την κλήση της μεθόδου DisplayMessageproc
    Public Delegate Sub DisplayMessageDelegate(ByVal aMessage As String)
    Public Sub DisplayMessage(ByVal aMessage As String)
        'Μέθοδος για την εμφάνιση δεδομένων
        'η οποία καλείται από το νήμα για την λήψη δεδομένων

        'Δημιουργία αναφοράς και συσχέτισή με την μέθοδο DisplayMessageProc
        Dim aDisplayMessageDelegate = New
                DisplayMessageDelegate(AddressOf DisplayMessageProc)
        'Κλήση της αναφοράς
        Me.Invoke(aDisplayMessageDelegate, aMessage)
    End Sub

    Private Sub DisplayMessageProc(ByVal aMessage As String)
        'Μέθοδος η οποία εμφανίζει το κείμενο
        Me.TextBoxReceived.AppendText(vbCrLf + aMessage)
        Me.TextBoxReceived.ScrollToCaret()
    End Sub

    Private Sub ButtonStart_Click(ByVal sender As System.Object,
                    ByVal e As System.EventArgs) Handles ButtonStart.Click
        'Εκκίνηση Server
        StartServer()
    End Sub

    Private Sub ButtonClose_Click(ByVal sender As System.Object,
                        ByVal e As System.EventArgs) Handles ButtonClose.Click
        'Τερματισμός
        End
    End Sub

    Private Sub ButtonSend_Click(ByVal sender As System.Object,
                        ByVal e As System.EventArgs) Handles ButtonSend.Click
        'Αποστολή μηνάματος στον συνδεδεμμένο πελάτη TCP
        'Ορισμός αντικειμένου NetworkSystem για την αποστολή δεδομένων
        'στο αντικείμενο aTcpClient
        Dim aWriteStream As NetworkStream = MyTcpClient.GetStream()
        'Μετατροπή κειμένου προς αποστολή σε πίνακα Bytes
        Dim buffer As Byte() = Encoding.ASCII.GetBytes(Me.TextBoxSend.Text)
        'Αποστολή δεδομένων στον TCPClient με το αντικείμενο NetworkStream
        aWriteStream.Write(buffer, 0, buffer.Length)
    End Sub
End Class
```

Ο κώδικας Visual Basic του παραθύρου του πελάτη TCP δίνεται παρακάτω.

```
Imports System.Net.Sockets
Imports System.Text
Imports System.Net

Public Class TcpClientForm

    'Δημιουργία αντικειμένου της κλάσης TCPClient
    Dim MyTcpClient As New TcpClient

    Private Sub TcpClientForm_Load(ByVal sender As System.Object,
                    ByVal e As System.EventArgs) Handles MyBase.Load
```

```
        'Εύρεση όλων των IP διευθύνσεων και εμφάνιση της
        'πρώτης διεύθυνσης

        'Εύρεση του ονόματος του Η/Υ με την χρήση της μεθόδου Dns.GetHostName
        Dim localHost As String
        localHost = Dns.GetHostName()

        'Εύρεση όλων των τοπικών διευθύνσεων με την χρήση της κλάσης Dns
        Dim anIPHostEntry = Dns.GetHostEntry(localHost)
        Dim anIPAddressArray = anIPHostEntry.AddressList

        'Εμφάνιση της πρώτης διεύθυνσης στο πλαίσιο
        'κειμένου με όνομα TextBoxIp
        Try
            Me.TextBoxIP.Text = anIPAddressArray(0).ToString
        Catch ex As Exception

        End Try
    End Sub

    Private Sub ButtonConnect_Click(ByVal sender As System.Object,
                   ByVal e As System.EventArgs) Handles ButtonConnect.Click
        'Μέθοδος για την σύνδεση με TCP Server

        Try
            'Χρήση της μεθόδου connect για την  σύνδεση με TCP Server
            MyTcpClient.Connect(IPAddress.Parse(Me.TextBoxIP.Text),
                                TextBoxPort.Text)
            DisplayMessage("Συνδέθηκε με TcpServer ...")

            'Εκκίνηση νήματος για την λήψη δεδομένων από τον TCP Server
            Dim clientThread As Threading.Thread = New Threading.
                                        Thread(AddressOf ReceiveData)
            clientThread.Start()

        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
    End Sub

    Private Sub ReceiveData()
        'Ορίσμός πίνακα Bytes για το διάβασμα δεδομένων
        Dim buffer(10024) As Byte
        Do
            Try
                'Ορισμός αντικειμένου NetworkStream για το διάβασμα δεδομένων
                'από το αντικείμενο aTcpClient
                Dim aReadStream As NetworkStream = MyTcpClient.GetStream()
                'Διάβασμα δεδομένων και αποθήκευση τους
                'στο αντικείμενο buffer
                aReadStream.Read(buffer, 0, MyTcpClient.ReceiveBufferSize)
                'Μετατροπή πίνακα Bytes σε κείμενο ASCII
                Dim aMessage As String =
                                System.Text.Encoding.ASCII.GetString(buffer)
                'Εμφάνιση μηνύματος
                DisplayMessage(MyTcpClient.Client.RemoteEndPoint.ToString +
                                ": " + aMessage)
            Catch ex As Exception
                'Εμφάνιση μηνύματος λάθους
                DisplayMessage(ex.Message)
                'Κλείσιμο TCPClient
                MyTcpClient.Close()
                'Έξοδος και τερματισμός του νήματος
                Exit Sub
```

```
            End Try
        Loop
    End Sub

    Private Sub ButtonSend_Click(ByVal sender As System.Object, _
        ByVal e As System.EventArgs) Handles ButtonSend.Click
        'Μέθοδος για την αποστολή δεδομένων στον TCP Server

        'Δημιουργία πίνακα Bytes για την αποστολή δεδομένων
        'και μετατροπή του κειμένου προς αποστολή σε bytes
        Dim buffer As Byte() = _
                System.Text.Encoding.ASCII.GetBytes(TextBox2.Text)
        'Ορισμός αντικειμένου της κλάσης NetworkStream  για την
        'αποστολή δεδομένων
        Dim aWriteStream As NetworkStream = MyTcpClient.GetStream()
        'Αποστολή πίνακα bytes στον TCP server
        aWriteStream.Write(buffer, 0, buffer.Length)
    End Sub

    'Ορισμός αναφοράς για την κλήση της μεθόδου DisplayMessageproc
    Public Delegate Sub DisplayMessageDelegate(ByVal aMessage As String)
    Public Sub DisplayMessage(ByVal aMessage As String)
        'Μέθοδος για την εμφάνιση δεδομένων
        'η οποία καλείται από το νήμα για την λήψη δεδομένων

        'Δημιουργία αναφοράς και συσχέτισή με την μέθοδο DisplayMessageProc
        Dim aDisplayMessageDelegate = New
                        DisplayMessageDelegate(AddressOf DisplayMessageProc)
        'Κλήση της αναφοράς
        Me.Invoke(aDisplayMessageDelegate, aMessage)
    End Sub

    Private Sub DisplayMessageProc(ByVal aMessage As String)
        'Μέθοδος η οποία εμφανίζει το κείμενο
        Me.TextBoxReceived.AppendText(vbCrLf + aMessage)
        Me.TextBoxReceived.ScrollToCaret()
    End Sub

    Private Sub ButtonDisconnect_Click(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles ButtonDisconnect.Click
        'Αποσύνδεση πελάτη
        MyTcpClient.Close()
    End Sub

    Private Sub ButtonClose_Click(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles ButtonClose.Click
        'Τερματισμός εφαρμογής
        End
    End Sub
End Class
```

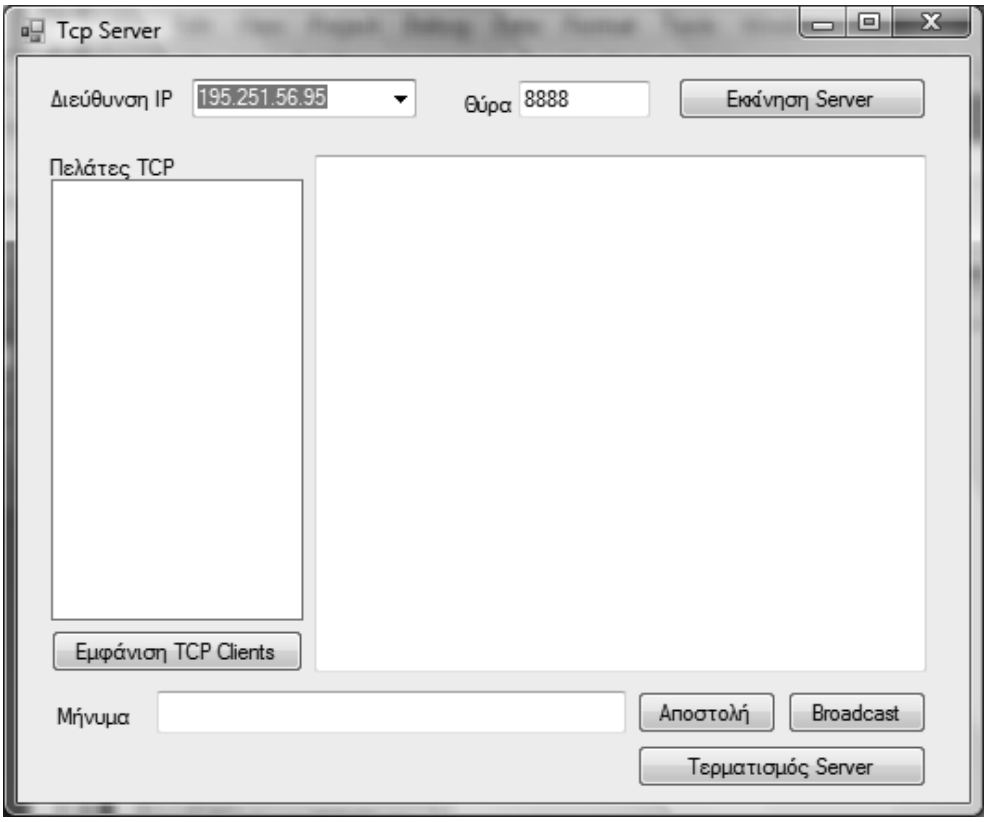
Εξυπηρετητής TCP με δυνατότητα πολλαπλών συνδέσεων

Η εφαρμογή του εξυπηρετητή η οποία παρουσιάσθηκε στην προηγούμενη ενότητα έχει την δυνατότητα να συνδέεται και να επικοινωνεί μόνο με έναν πελάτη κάθε φορά. Σε περίπτωση που απαιτούνται περισσότερες από μια συνδέσεις με πελάτες ταυτόχρονα, τότε πρέπει να γίνουν οι παρακάτω αλλαγές:

- Η αναμονή για συνδέσεις πελατών να εκτελείται διαρκώς με ένα διαφορετικό νήμα το οποίο θα δέχεται αιτήματα για νέες συνδέσεις και να τα εξυπηρετεί.
- Αντί μίας μεταβλητής αντικειμένου TCPClient η οποία χρησιμοποιείται για την επικοινωνία με τους

πελάτες TCP, για την κάθε νέα σύνδεση δημιουργείται κι ένα νέο αντικείμενο της κλάσης TCPClient. Για την κάθε νέα σύνδεση δημιουργείται και ένα νέο νήμα το οποίο διαβάζει τα δεδομένα τα οποία αποστέλλει ο πελάτης.

Επίσης, η τροποποιημένη εφαρμογή του εξυπηρετητή πολλαπλών συνδέσεων (εικόνα 16.7) διαθέτει επιπλέον μία λίστα επιλογής όπου εμφανίζονται τα στοιχεία των συνδεδεμένων πελατών, ένα πλήκτρο για την εμφάνιση των πελατών και ένα πλήκτρο για την αποστολή μηνύματος ταυτόχρονα σε όλους τους συνδεδεμένους πελάτες.



Εικόνα 4.4 Παραθυρική εφαρμογή πελάτη TCP

Παρακάτω δίνεται ο κώδικας για την υλοποίηση του παραθύρου του εξυπηρετητή πολλαπλών συνδέσεων.

```
Imports System.Net.Sockets
Imports System.Net
Imports System.Text

Public Class TcpServerForm

    'Ορισμός αντικειμένου TCP Server για την αποδοχή συνδέσεων TCP
    Dim MyTcpListener As TcpListener
    'Ορισμός νήματος για την αποδοχή συνδέσεων
    Public ThreadListen As System.Threading.Thread
    'Ορισμός συλλογής συνδεδέμμένων TCP Clients
    Public TcpClientsCollection As New Collection

    Private Sub TcpServerForm_Load(ByVal sender As System.Object,
```

```
                ByVal e As System.EventArgs) Handles MyBase.Load
    'Εμφάνιση όλων των IP διευθύνσεων και εμφάνιση τους
    'στο ComboBoxLocalIP

    'Εύρεση του ονόματος του Η/Υ με την χρήση της μεθόδου Dns.GetHostName
    Dim localHost As String
    localHost = Dns.GetHostName()

    'Εύρεση όλων των τοπικών διευθύνσεων με την χρήση της κλάσης Dns
    Dim anIPHostEntry = Dns.GetHostEntry(localHost)
    Dim anIPAddressArray = anIPHostEntry.AddressList
    Me.ComboBoxLocalIp.Items.Clear()

    'Πρόσθεση διευθύνσεων στο ComboBoxLocalIp
    For i As Integer = 0 To anIPAddressArray.GetUpperBound(0)
        Me.ComboBoxLocalIp.Items.Add(anIPAddressArray(i).ToString)
    Next
    'Επιλογή πρώτης διεύθυνσης IP
    If Me.ComboBoxLocalIp.Items.Count > 0 Then
        Me.ComboBoxLocalIp.SelectedIndex = 0
    End If
End Sub
Public Sub StartServer()
    If IsNothing(Me.ComboBoxLocalIp.SelectedItem) Then Exit Sub
    Try
        'Εκκίνηση TCP Server για την αποδοχή συνδέσεων
        MyTcpListener = New TcpListener(
               IPAddress.Parse(Me.ComboBoxLocalIp.SelectedItem),
               CInt(TextBoxPort.Text))
        DisplayMessage("Εκκίνηση TCP Server ....")
        MyTcpListener.Start()
        DisplayMessage("Η τοπική διεύθυνση IP και η θύρα είναι:" +
                       MyTcpListener.LocalEndpoint.ToString)
        'Όρισμός νήματος για την αποδοχή συνδέσεων από πελάτες
        ThreadListen = New
            System.Threading.Thread(AddressOf ListenForConnectionRequests)
        'Εκκινηση νήματος
        ThreadListen.Start()
        DisplayMessage("Αναμονή συνδέσεων.....")
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub

Public Sub ListenForConnectionRequests()
    Do
        'Αναμονή σύνδεσης
        Dim aTcpClient As TcpClient = MyTcpListener.AcceptTcpClient
        'Πρόσθεση νέου πελάτη στην συλλογή TcpClientsCollection
        TcpClientsCollection.Add(aTcpClient,
                       aTcpClient.Client.RemoteEndPoint.ToString)
        'Εκκίνηση νήματος για την λήψη δεδομένων από τον πελάτη TCP
        Dim aClientTread As Threading.Thread =
            New Threading.Thread(AddressOf ReceiveData)
        aClientTread.Start(aTcpClient)
        DisplayMessage(aTcpClient.Client.RemoteEndPoint.ToString +
                       ": Σύνδεση ")
    Loop
End Sub

Private Sub ReceiveData(ByVal aTcpClient As TcpClient)
    'Ορίσμός πίνακα Bytes για το διάβασμα δεδομένων
    Dim buffer(10024) As Byte
    Dim aMessage As String
```

```
        Do
            Try
                'Ορισμός αντικειμένου NetworkStream για το διάβασμα δεδομένων
                'από το αντικείμενο aTcpClient
                Dim aReadStream As NetworkStream = aTcpClient.GetStream()
                'Διάβασμα δεδομένων και αποθήκευση τους
                'στο αντικείμενο buffer
                'Στη μεταβλητή bytesRead επιστρέφεται ο αριθμός
                'των bytes τα οποία διαβάσθηκαν
                Dim bytesRead As Integer = aReadStream.Read(buffer, 0,
                                           CInt(aTcpClient.ReceiveBufferSize))
                'Έλεγχος αν ο Client έχει αποσυνδεθεί
                If bytesRead = 0 Then
                    Me.DisplayMessage(
                          aTcpClient.Client.RemoteEndPoint.ToString +
                          ": Αποσύνδεση")
                    'Διαγραφή TCP Client από το πίνακα των
                    'συνδεδεμμένων Clients
                    ListBoxTcpClients.Items.Remove(
                             aTcpClient.Client.RemoteEndPoint.ToString())
                    'Έξοδος και τερματισμός του νήματος
                    Exit Sub
                End If
                'Μετατροπή πίνακα Bytes σε κείμενο ASCII
                aMessage = System.Text.Encoding.ASCII.GetString(buffer)
                'Εμφάνιση μηνύματος
                Me.DisplayMessage(
                          aTcpClient.Client.RemoteEndPoint.ToString +
                          ": " + aMessage)
            Catch ex As Exception
                'Εμφάνιση μηνύματος λάθους
                DisplayMessage(ex.Message)
                'Κλείσιμο TCPClient
                aTcpClient.Close()
                'Έξοδος και τερματισμός του νήματος
                Exit Sub
            End Try
        Loop
    End Sub

    'Ορισμός αναφοράς για την κλήση της μεθόδου DisplayMessageproc
    Public Delegate Sub DisplayMessageDelegate(ByVal aMessage As String)
    Public Sub DisplayMessage(ByVal aMessage As String)
        'Μέθοδος για την εμφάνιση δεδομένων
        'η οποία καλείται από το νήμα για την λήψη δεδομένων

        'Δημιουργία αναφοράς και συσχέτισή με την μέθοδο DisplayMessageProc
        Dim aDisplayMessageDelegate = New
                      DisplayMessageDelegate(AddressOf DisplayMessageProc)
        'Κλήση της αναφοράς
        Me.Invoke(aDisplayMessageDelegate, aMessage)
    End Sub

    Private Sub DisplayMessageProc(ByVal aMessage As String)
        'Μέθοδος η οποία εμφανίζει το κείμενο
        Me.TextBoxReceived.AppendText(vbCrLf + aMessage)
        Me.TextBoxReceived.ScrollToCaret()
    End Sub

    Private Sub ButtonStart_Click(ByVal sender As System.Object,
                      ByVal e As System.EventArgs) Handles ButtonStart.Click
        'Εκκίνηση Server
        StartServer()
    End Sub
```

```
    Private Sub ButtonClose_Click(ByVal sender As System.Object,
              ByVal e As System.EventArgs) Handles ButtonClose.Click
        'Τερματισμός
        End
    End Sub

    Private Sub ButtonBroadcast_Click(ByVal sender As System.Object,
                   ByVal e As System.EventArgs) Handles ButtonBroadcast.Click
        'Αποστολή μηνύματος για κάθε συνδεδεμμένο TCP Client
        For Each aTcpClient In Me.TcpClientsCollection
            'Ορισμός αντικειμένου NetworkSystem για την αποστολή δεδομένων
            'στο αντικείμενο aTcpClient
            Dim aWriteStream As NetworkStream = aTcpClient.GetStream()
            'Μετατροπή κειμένου προς αποστολή σε πίνακα Bytes
            Dim buffer As Byte() =
                   Encoding.ASCII.GetBytes(Me.TextBoxSend.Text)
            'Αποστολή δεδομένων στον TCPClient με το αντικείμενο NetworkStream
            aWriteStream.Write(buffer, 0, buffer.Length)
        Next
    End Sub


    Private Sub ButtonSend_Click(ByVal sender As System.Object,
                     ByVal e As System.EventArgs) Handles ButtonSend.Click
        If IsNothing(ListBoxTcpClients.SelectedItem) Then Exit Sub
        Dim writeStream As NetworkStream =
             TcpClientsCollection(ListBoxTcpClients.SelectedItem).GetStream()
        Dim sendBytes As Byte() =
                   Encoding.ASCII.GetBytes(Me.TextBoxSend.Text)
        writeStream.Write(sendBytes, 0, sendBytes.Length)
    End Sub

    Private Sub ButtonShowTcpClients_Click(ByVal sender As System.Object,
             ByVal e As System.EventArgs) Handles ButtonShowTcpClients.Click
        Me.ListBoxTcpClients.Items.Clear()
        For Each aTcpClient As TcpClient In TcpClientsCollection
            Me.ListBoxTcpClients.Items.Add(
                              aTcpClient.Client.RemoteEndPoint.ToString)
        Next
    End Sub
End Class
```

4.4 Αναφορές

- Todd Herman, Allen Jones, Matthew MacDonald, Rakesh Rajan, "Visual Basic 2008 Recipes: A Problem-Solution Approach", Chapter 11 Netowrking and Remoting, Apress, ISBN 13: 978-1-59059-970-9

- Joseph Albahari,"Threading in C#",http://www.albahari.info/threading/threading.pdf"

5. MODBUS

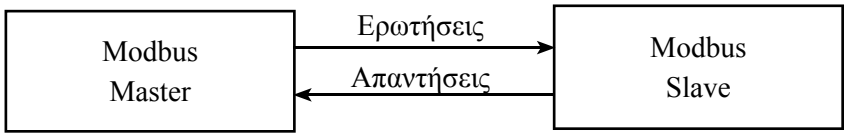
5.1 Το πρωτόκολλο Modbus

Η πρωτόκολλο Modbus είναι ένα ευρέως διαδεδομένο πρωτόκολλο για την αποστολή και λήψη δεδομένων κυρίως σε εφαρμογές αυτοματισμού. Το πρωτόκολλο αυτό ανήκει στο επίπεδο εφαρμογής του μοντέλου αναφοράς OSI και αναπτύχθηκε αρχικά από την Εταιρεία Modicon ως σειριακό πρωτόκολλο για την επικοινωνία με PLC. Πολύ σύντομα καθιερώθηκε ως βιομηχανικό πρότυπο για τις επικοινωνίες με ηλεκτρονικές συσκευές λόγω των παρακάτω πλεονεκτημάτων του:

- Είναι απλό στην χρήση του και σχετικά εύκολο στην υλοποίηση του
- Είναι ανοιχτό πρωτόκολλο και η χρήση του είναι ελεύθερη
- Αναπτύχθηκε ειδικά για βιομηχανικές εφαρμογές

Το πρωτόκολλο Modbus χρησιμοποιείται κυρίως σε συστήματα SCADA για την διασύνδεση των Η/Υ με προγραμματιζόμενους λογικούς ελεγκτές (PLCs), ψηφιακά συστήματα ελέγχου (DCS) και διάφορες άλλες ηλεκτρονικές συσκευές μετρήσεων και ελέγχου.

Το πρωτόκολλο Modbus ορίζει δύο τύπους συσκευών οι οποίες επικοινωνούν: τον Modbus Master και τον Modbus Slave. Ο Modbus Master είναι η συσκευή η οποία στέλνει μηνύματα στον Modbus Slave και ο Modbus Slave απαντά στα μηνύματα τα οποία δέχεται. Ο Modbus Master είναι συνήθως η συσκευή η οποία συλλέγει τα δεδομένα από πολλούς περιφερειακούς σταθμούς μετρήσεων και αυτοματισμού. Ένας Modbus Master μπορεί να επικοινωνεί μέχρι και με 240 Modbus Slaves οι οποίοι είναι συνδεδεμένοι στο ίδιο δίκτυο και έχει ο καθένας την δικιά του μοναδική διεύθυνση επικοινωνίας ή αναγνωριστικό αριθμό σταθμού.



Υπάρχουν τριών ειδών τύποι μηνυμάτων τα οποία ανταλλάσσονται σε επικοινωνίες με την χρήση του πρωτοκόλλου Modbus:

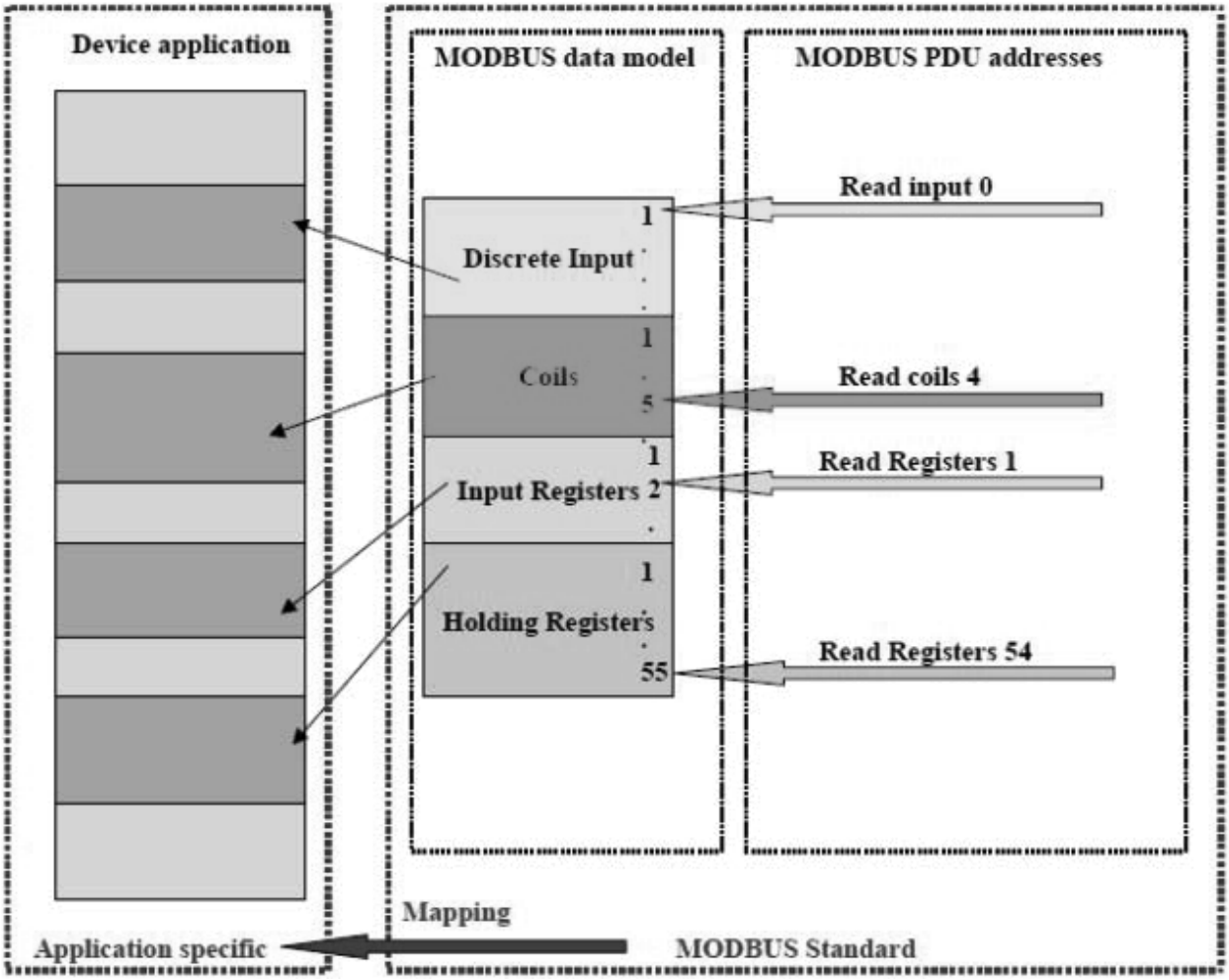
Τύπος	Αποστολέας	Περιγραφή
Ερώτημα	Modbus Master	Ερώτημα προς τον Modbus Slave
Απάντηση	Modbus Slave	Απάντηση προς τον Modbus Master
Μήνυμα Λάθους	Modbus Slave	Μήνυμα λάθους του Modbus Slave

Το πρωτόκολλο Modbus ορίζει τους παρακάτω τύπους δεδομένων :

Ονομασία	Τύπος δεδομένων	Πρόσβαση
Coil	bit	Εγγραφή και ανάγνωση
Input Discrete	bit	Μόνο για ανάγνωση
Holding Register	word (16 bit)	Εγγραφή και ανάγνωση
Input Register	word (16 bit)	Μόνο για ανάγνωση

Πίνακας 5.1 Οι τύποι δεδομένων του πρωτοκόλλου Modbus

Οι τύποι δεδομένων Input Coil και Input Register είναι μόνοι για ανάγνωση ενώ οι τύποι Coil και Holding Register μπορούν να χρησιμοποιηθούν τόσο για ανάγνωση όσο και για εγγραφή τιμών στον Modbus Slave.



Εικόνα 5.1 Τύποι δεδομένων του πρωτόκολου Modbus

Λειτουργίες μηνυμάτων του Modbus

Το κάθε ερώτημα το οποίο αποστέλλεται από τον Modbus Master στον Modbus Slave περιέχει ένα κωδικό

αριθμό λειτουργίας ο οποίος χρησιμοποιείται για να προσδιορίζει την λειτουργία η οποία απαιτείται να γίνει ή το είδος του περιεχομένου του μηνύματος.

Οι λειτουργίες και οι αντίστοιχοι κωδικοί των λειτουργιών δίνονται στον παρακάτω πίνακα.

Τύπος	Τύπος Δεδομένων	Όνομα Λειτουργίας	Κωδικός
Πρόσβαση Δεδομένων	Bit	Read Discrete Inputs	2
		Read Coils	1
		Write Single Coil	5
		Write Multiple Coils	15
	16-bit	Read Input Register	4
		Read Holding Registers	3
		Write Single Register	6
		Write Multiple Registers	16
		Read/Write Multiple Registers	23
		Mask Write Register	22
		Read FIFO Queue	24
	File Record Access	Read File Record	20
		Write File Record	21
Διαγνωστικά Μηνύματα		Read Exception Status	7
		Diagnostic	8
		Get Com Event Counter	11
		Get Com Event Log	12
		Report Slave ID	17
		Read Device Identification	43
Άλλο		Encapsulated Interface Transport	43

Πίνακας 5.2 Οι κωδικοί λειτουργιών του πρωτοκόλλου Modbus

5.2 Τύποι πρωτοκόλλων Modbus

Το πρωτόκολλο Modbus μπορεί να χρησιμοποιηθεί σε σειριακά δίκτυα όπως RS-232 και RS-422/RS-485 και δίκτυα TCP/IP .

Υπάρχουν διάφορες παραλλαγές του πρωτοκόλλου Modbus για την μετάδοση σε διαφορετικά είδη δικτύων, οι κυριότερες από τις οποίες είναι οι παρακάτω:

- Modbus RTU για σειριακή επικοινωνία
- Modbus ASCII για σειριακή επικοινωνία
- Modbus TCP για επικοινωνία με το πρωτόκολλο TCP.

Αναλόγως του τύπου πρωτοκόλλου ορίζονται και διαφορετικές δομές πακέτων για την αποστολή και μετάδοση μηνυμάτων.

Το πρωτόκολλο MODBUS RTU

Το πρωτόκολλο Modbus RTU χρησιμοποιείται για σειριακές επικοινωνίες όπως RS-232 και RS-422/RS-485. Είναι ο περισσότερο διαδεδομένος τύπος από τα είδη πρωτοκόλλων Modbus και χρησιμοποιεί bits για την αποστολή και λήψη δεδομένων και έλεγχο CRC για την ορθότητα της μετάδοσης των δεδομένων.

Όνομα	Μήκος	Περιγραφή
Αρχή	3.5 chars	Κενό τουλάχιστον 3.5 χαρακτήρων
Διεύθυνση Slave	8 bits	Διεύθυνση Σταθμού Slave
Κωδικός λειτουργίας	8 bits	Κωδικός λειτουργίας
Δεδομένα	n * 8 bits	Δεδομένα τα οποία αποστέλλονται ή παραλαμβάνονται
CRC Check	16 bits	Έλεγχος λάθους CRC
Τέλος	3.5c idle	Κενό τουλάχιστον 3.5 χαρακτήρων

Το πρωτόκολλο MODBUS ASCII

Το πρωτόκολλο Modbus ASCII χρησιμοποιείται για σειριακές επικοινωνίες όπως και το πρωτόκολλο Modbus RTU. Το πρωτόκολλο αυτό χρησιμοποιεί ASCII χαρακτήρες για την μετάδοση δεδομένων και χρησιμοποιεί έλεγχο LRC για την ορθότητα της μετάδοσης των δεδομένων. Το δεδομένα τα οποία αποστέλλονται μετατρέπονται σε δεκαεξαδικούς χαρακτήρες ASCII (0..F) πριν την μετάδοση τους.

Όνομα	Μήκος	Περιγραφή
Αρχή	1 char	άνω κάτω τελεία (:) (ASCII 3A hex)
Διεύθυνση Slave	2 chars	Διεύθυνση Σταθμού Slave
Κωδικός λειτουργίας	2 chars	Κωδικός λειτουργίας
Δεδομένα	n chars	Δεδομένα τα οποία αποστέλλονται ή παραλαμβάνονται
LRC Check	2 chars	Έλεγχος λάθους LRC
Τέλος	2 chars	CRLF (δεκαεξαδικές τιμές ASCII 0D και 0A)

Το πρωτόκολλο MODBUS TCP

Το πρωτόκολλο αυτό χρησιμοποιείται για επικοινωνίες TCP/IP. Τα δεδομένα μεταδίδονται σε μορφή bytes και δεν απαιτείται έλεγχος δεδομένων αφού αυτός γίνεται από το πρωτόκολλο TCP.

Όνομα	Μήκος	Περιγραφή
Transaction ID	2 bytes	Κωδικός για τον συγχρονισμό μηνυμάτων
Είδος πρωτοκόλλου	2 bytes	0 για MODBUS/TCP
Πλήθος	2 bytes	Αριθμός Bytes που ακολουθούν
Κωδικός σταθμού	1 byte	Διεύθυνση Σταθμού Slave

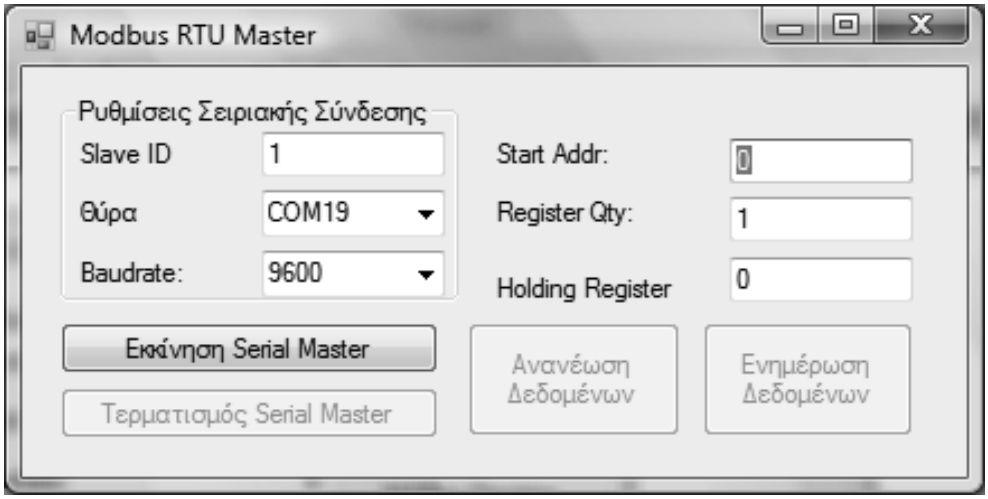
Κωδικός λειτουργίας	1 byte	Κωδικός λειτουργίας
Δεδομένα	n bytes	Δεδομένα

6. Εφαρμογές MODBUS με την Visual Basic

6.1 Εφαρμογή Modbus RTU Master

Περιγραφή της εφαρμογής Modbus RTU Master

Η εφαρμογή η οποία φαίνεται στο σχήμα 17.2 λειτουργεί ώς Modbus RTU Master. Η εφαρμογή χρησιμοποιεί ένα παράθυρο με όνομα FormModbusMaster.



Εικόνα 6.1 Εφαρμογή Modbus RTU Master

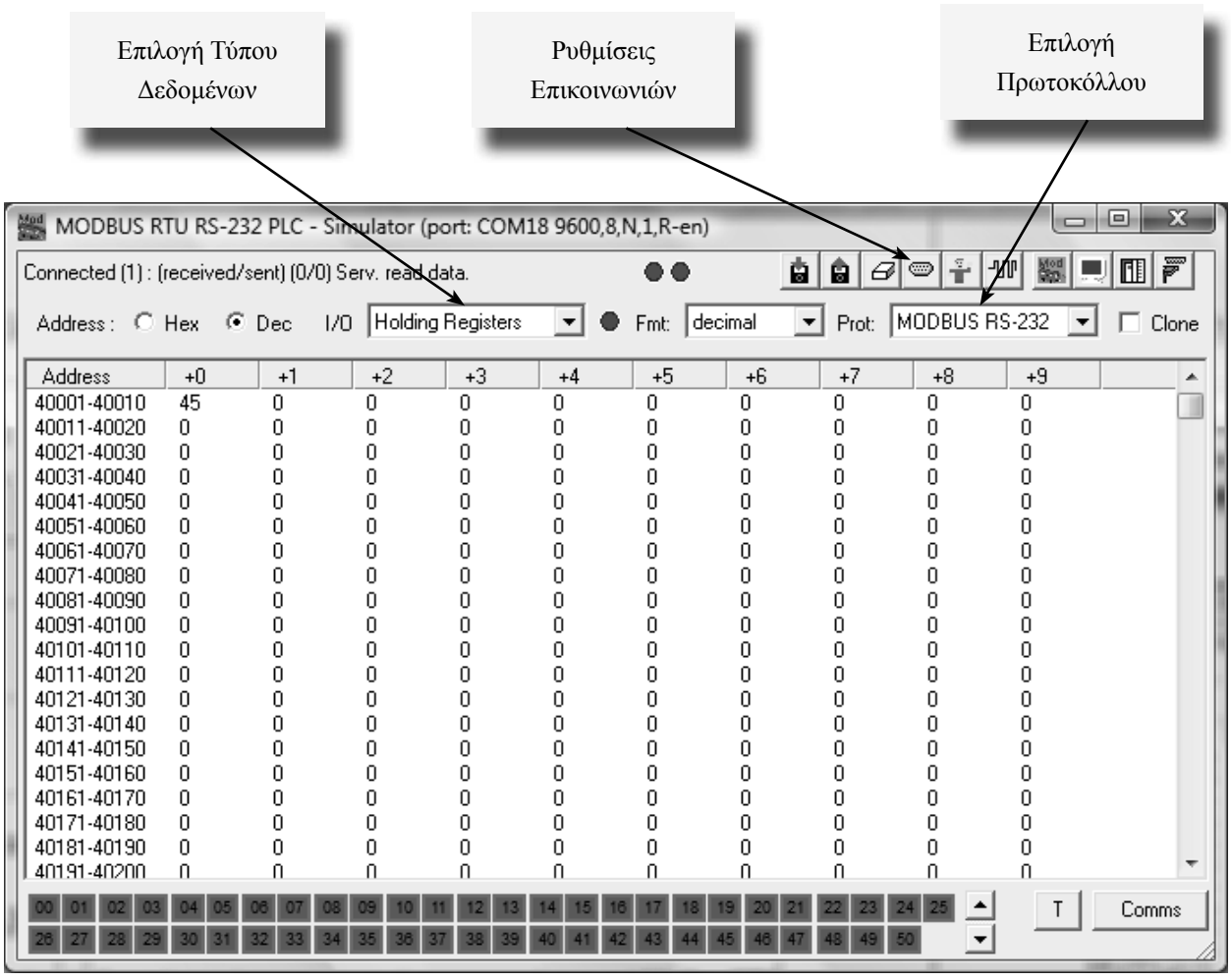
Το παράθυρο FormModbusMaster έχει τα παρακάτω στοιχεία ελέγχου:

Όνομα	Τύπος	Περιγραφή
TextBoxSerialSlaveID	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την διεύθυνση του σταθμού RTU Slave με την οποία θα επικοινωνήσει
ComboBoxComPorts	ComboBox	Χρησιμοποιείται για να εμφανίσει όλες τις διαθέσιμες σειριακές θύρες και την επιλογή από τον χρήστη της θύρας για την σειριακή επικοινωνία
ComboBoxBaudRate	ComboBox	Χρησιμοποιείται για την επιλογή της ταχύτητας της σειριακής επικοινωνίας
TextBoxStartAddress	TextBox	Διεύθυνση RTU Slave από την οποία θα διαβαστούν τα δεδομένα.
TextBoxNumOfRegisters	TextBox	Πλήθος δεδομένων προς ανάγνωση

TextBoxHoldingRegister1	TextBox	Πλαίσιο κειμένου όπου εμφανίζεται η τιμή του πρώτου από τα δεδομένα τα οποία διαβάζονται από τον RTU Slave
ButtonStartMaster		Πλήκτρο για το εκκίνηση του Modbus RTU Master
ButtonStopMaster		Πλήκτρο για τον τερματισμό του Modbus RTU Master
ButtonRefreshData		Πλήκτρο για την ανάγνωση των Holding Registers και εμφάνιση του πρώτου από τα δεδομένα τα οποία επιστρέφονται στο TextBoxHoldingRegister1
ButtonSetData		Πλήκτρο για την τροποποίησης της τιμής του πρώτου Holding Register του ModbusRTUMaster με βάση την τιμή του TextBoxHoldingRegister1

Πίνακας 6.1 Τα στοιχεία ελέγχου του Modbus RTU Master

Δοκιμή της εφαρμογής Modbus RTU Master



Εικόνα 6.2 Εφαρμογή προσομοίωσης Modbus Slave

Για την δοκιμή της εφαρμογής, εάν δεν υπάρχει κάποια πραγματική περιφερειακή συσκευή Modbus RTU Slave η οποία να παρέχει δεδομένα, μπορεί να χρησιμοποιηθεί η εφαρμογή mod_RSsim.exe. Η εφαρμογή αυτή (σχήμα 17.3) δίνει την δυνατότητα εξομοίωσης της λειτουργίας Modbus Slave ώστε να είναι δυνατόν να γίνει η ανάπτυξη και δοκιμή της λειτουργίας εφαρμογών Modbus Master χωρίς να είναι απαραίτητη η χρήση πραγματικών συνθηκών λειτουργίας.

Για τη δοκιμή της επικοινωνίας πρέπει να γίνουν τα παρακάτω βήματα:
- Να γίνει εκκίνηση της εφαρμογής mod_RSsim.exe.
- Να επιλεγεί ως πρωτόκολλο επικοινωνίας το Modbus RS-232
- Να γίνουν οι ρυθμίσεις επικοινωνίας (σειριακή θύρα και ταχύτητα επικοινωνίας)
- Να επιλεγεί ως τύπος δεδομένων τα Holding Registers
- Να δοθεί τιμή στον πρώτο Holding Register (Διεύθυνση 40001)
- Να γίνει εκκίνηση της εφαρμογής Modbus RTU Master
- Να επιλεγεί η σειριακή θύρα και η ταχύτητα επικοινωνίας
- Να πατηθεί το πλήκτρο "Εκκίνηση Modbus Master"
- Να πατηθεί το πλήκτρο "Ανανέωση Δεδομένων". Η τιμή του πρώτου Holding Register (διεύθυνση 4001) της εφαρμογής mod_RSsim.exe μεταφέρεται και εμφανίζεται στο πλαίσιο κειμένου TextBoxHoldingRegister1
- Να τροποποιηθεί η τιμή του πλαισίου κειμένου TextBoxHoldingRegister1 και να πατηθεί το πλήκτρο "Αποστολή δεδομένων". Η τιμή αυτή εμφανίζεται ως η νέα τιμή για του πρώτου Holding Register της εφαρμογής mod_RSsim.exe

Πηγαίος κώδικας της εφαρμογής Modbus RTU Master

Η εφαρμογή Modbus RTU Master αποτελείται από δύο τμήματα: την κλάση αντικειμένων ModbusSerialMaster και το παράθυρο FormModbusMaster.

Η κλάση αντικειμένων ModbusSerialMaster υλοποιεί τμήμα του πρωτοκόλλου Modbus RTU το οποίο αφορά την ανάγνωση και την εγγραφή HoldingRegisters.

```
Imports System.IO.Ports

Public Class ModbusSerialMaster
    'Ορισμός κλάσης η οποία υλοποιεί τμήμα του πρωτοκόλλου Modbus
    'για Serial RTU Master
    Public MySerialPort As SerialPort
    Public ModbusStatus As String

    Public Shared Function CreateRTU(ByVal aSerialPort) As ModbusSerialMaster
        Dim aMaster As ModbusSerialMaster = New ModbusSerialMaster
        'Μέθοδος της κλάσης ModbusSerialMaster για την δημιουργία
        'αντικειμένου της κλάση με παράμετρο σειριακή θύρα
        aMaster.MySerialPort = aSerialPort
        'Καθορισισμός μέγιστων χρόνων αναμονής για την
        'λήψη και αποστολή δεδομένων
        aMaster.MySerialPort.ReadTimeout = 1000
        aMaster.MySerialPort.WriteTimeout = 1000
        Return aMaster
    End Function

    Private Function GetCRC(ByVal message() As Byte) As Byte()
        'Διαδικασία η οποία υπολογίζει και επιστρέφει στην παράμετερο CRC
```

```
        'δύο bytes CRC για τον έλεγχο της ορθότητας του μηνύματος

        'Ορισμός καταχωρητή CRCRegister 16 bit με τιμή 1 σε όλα τα bits
        Dim CRCRegister As UShort = &HFFFF
        'Παράμετρος η οποία ελέγχει το LSB του CRC Register
        Dim CRCLSB As Boolean
        'Έλεγχος όλων των bytes του μηνύματος εκτός των 3 τελευταίων
        For i As Integer = 0 To message.Length - 3
            'XOR όλως των bytes του μηνύματος με τον καταχωρητή CRCRegister
            CRCRegister = CRCRegister Xor message(i)
            For j As Integer = 0 To 7
                'Υπολογισμός του LSB του CRCRegister
                CRCLSB = CRCRegister And &H1
                'Right Shift του καταχωρητή CRCRegister και
                'γέμισμα MSB με 0 (And &H7FFF)
                CRCRegister = (CRCRegister >> 1) And &H7FFF
                'Εαν το LSB = 1 τότε Xor με &HA001
                If CRCLSB Then CRCRegister = CRCRegister Xor &HA001
            Next
        Next
        'Επιστροφή CRC bytes από την τελική τιμή  του καταχωρητή CRCRegister
        Dim CRC(1) As Byte
        CRC(1) = CRCRegister >> 8
        CRC(0) = CRCRegister Mod 256
        Return CRC
    End Function

    Private Sub BuildMessage(ByVal slaveAddress As Byte,
        ByVal messageType As Byte, ByVal startAdress As UShort,
        ByVal numberOfPoints As UShort, ByRef message() As Byte)
        'Η διαδικασία αυτή συμπληρώνει τα απαραίτητα στοιχεία
        'στον πίνακα Bytes messages για την μετάδοση του μηνύματος

        'Byte 0 - Διεύθυνση Modbus Slave
        message(0) = slaveAddress
        'Byte 1 - Τύπος μηνύματος
        message(1) = messageType
        'Bytes 3 και 4 - Διεύθυνση Modbus Slave
        message(3) = startAdress >> 8
        message(4) = startAdress Mod 256
        'Bytes 5 και 6  - Πλήθος Bytes
        message(4) = numberOfPoints >> 8
        message(5) = numberOfPoints Mod 256
        'Πρόσθεση CRC στα δύο τελευταία Bytes
        Dim CRC() As Byte = GetCRC(message)
        message(message.Length - 2) = CRC(0)
        message(message.Length - 1) = CRC(1)
    End Sub

    Private Function CheckResponse(ByVal response() As Byte) As Boolean
        'Έλεγχος αν τα bytes CRC του ληφθέντος μηνύματος είναι
        'ισα με τα bytes του υπολογιζόμενου CRC
        Dim CRC() As Byte = GetCRC(response)
        If (CRC(0) = response(response.Length - 2)) And (CRC(1) =
                                    response(response.Length - 1)) Then
            Return True
        Else
            Return False
        End If
    End Function

    Private Sub GetResponse(ByRef response() As Byte)
        'Διάβασμα εισερχόμενων bytes από την σειριακή θύρα
        For i As Integer = 0 To response.Length - 1
```

```
            response(i) = CType(MySerialPort.ReadByte(), Byte)
        Next
    End Sub

    Public Function ReadHoldingRegisters(ByVal slaveAddress As Byte,
            ByVal startAdress As UShort,
            ByVal numberOfPoints As UShort) As UShort()
        'Αποστολή μηνύματος τύπου 3 για το διάβασμα HoldingRegisters.
        'Το μήνυμα αυτό επιστρέφει ένα πίνακα τιμών για τα Holding Registers

        'Ορισμός πίνακα για τις τιμές των Holding Registers
        Dim values(numberOfPoints) As UShort
        'Ελεγχος αν η σειριακή θύρα είναι ανοικτή
        If (MySerialPort.IsOpen) Then
            'Καθαρισμός εισερχομένης / εξερχόμενης προσωρινής μνήμης
            MySerialPort.DiscardOutBuffer()
            MySerialPort.DiscardInBuffer()
            'Καθορισμός πίνακα 8 Bytes για το προς αποστολή μήνυμα μήκος
            Dim aMessage(7) As Byte
            'Καθορισμός πίνακα bytes για την απάντηση στο μήνυμα τύπου 3
            Dim response(4 + 2 * numberOfPoints) As Byte
            'Συμπλήρωση πεδίων του μηνύματος προς αποστολή
            BuildMessage(slaveAddress, 3, startAdress, numberOfPoints,
                                                        aMessage)
            'Αποστολή μηνύματος στην σειριακή θύρα
            Try
                MySerialPort.Write(aMessage, 0, aMessage.Length)
                'Λήψη απάντησης από την σειριακή θύρα
                GetResponse(response)
            Catch ex As Exception
                ModbusStatus = "Λάθος ανάγνωσης μηνύματος " + ex.Message
            End Try
            'Έλεγχος μηνύματος
            If CheckResponse(response) Then
                'Μετατροπή τιμών και αποθήκευση τους στον πίνακα values
                For i As Integer = 0 To ((response.Length - 5) / 2) - 1
                    values(i) = response(2 * i + 3)
                    values(i) <<= 8
                    values(i) += response(2 * i + 4)
                Next
                ModbusStatus = "Ανάγνωση επιτυχής"
            Else
                ModbusStatus = "Λάθος CRC"
            End If
        Else
            ModbusStatus = "Η σειριακή θύρα δεν είναι ανοικτή"
        End If
        Return values
    End Function

    Public Sub WriteHoldingRegisters(ByVal slaveAddress As Byte,
            ByVal startAdress As UShort,
            ByVal numberOfPoints As UShort, ByVal values() As UShort)
        'Αποστολή μηνύματος τύπου 3 για το διάβασμα HoldingRegisters.
        'Το μήνυμα αυτό επιστρέφει ένα πίνακα τιμών για τα Holding Registers

        'Ελεγχος αν η σειριακή θύρα είναι ανοικτή
        If MySerialPort.IsOpen Then
            'Καθαρισμός εισερχομένης / εξερχόμενης προσωρινής μνήμης
            MySerialPort.DiscardOutBuffer()
            MySerialPort.DiscardInBuffer()
            'Ορισμός πίνακα bytes για το μήνυμα προς αποστολή
            'Το μήνυμα είναι  1 Διεύθυνση + 1 Τύπος + 2 Αρχή + 2 Πλήθος +
            '1 Πλήθος Bytes + 2 * πλήθος δεδο΄μένων + 2 CRC
```

```
            Dim message(9 + 2 * numberOfPoints - 1) As Byte
            'Καθορισμός πίνακα 8 bytes για το μήνυμα απάντησης
            Dim response(7) As Byte
            'Πρόσθεση πλήθους τιμών στο μήνυμα
            message(6) = numberOfPoints * 2
            'Πρόσθεση τιμών στο μήνυμα
            For i = 0 To numberOfPoints - 1
                'High Byte τιμής
                message(7 + 2 * i) = values(i) >> 8
                'Low Byte τιμής
                message(8 + 2 * i) = values(i) Mod 256
            Next
            'Πρόσθεση υπολοίπων πεδίων στο μήμυμα προς αποστολή
            BuildMessage(slaveAddress, 16, startAdress, numberOfPoints,
                                                          message)
            'Αποστολή μηνύματος στην σειριακή θύρα
            Try
                MySerialPort.Write(message, 0, message.Length)
                'Λήψη απάντησης από την σειριακή θύρα
                GetResponse(response)

            Catch ex As Exception
                ModbusStatus = "Λάθος μηνύματος εγγραφής τιμών " + ex.Message
            End Try
            'Evaluate message:
            If CheckResponse(response) Then
                ModbusStatus = "Εγγραγή επιτυχής"
            Else
                ModbusStatus = "Λάθος CRC"
            End If
        Else
            ModbusStatus = "Η σειριακή θύρα δεν είναι ανοικτή"
        End If
    End Sub
End Class
```

Ο κώδικας Visual Basic του παραθύρου FormModbusMaster δίνεται παρακάτω.

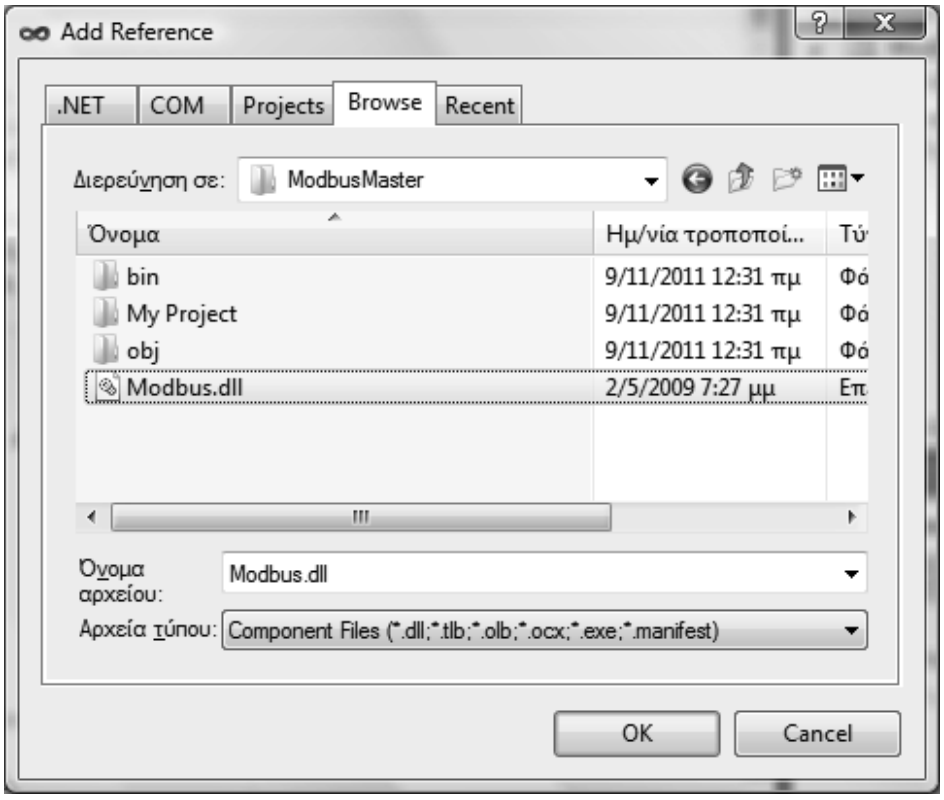
```
Imports System.IO.Ports
Imports System.Timers

Public Class FormModbusMaster

    'Ορισμός σειριακής θύρας επικοινωνιών
    Dim MyMasterPort As New SerialPort
    'Ορισμός αντικειμένου Modbus Master
    Dim MyMaster As ModbusSerialMaster

    Private Sub FormModbusAsciiClient_Load(ByVal sender As System.Object,
                          ByVal e As System.EventArgs) Handles MyBase.Load
        'Χρήση της μεθόδου Computer.Ports.SerialPortNames η οποία
        'επιστρέφει μια συλλογή με τις διαθέσιμες θύρες του Η/Υ για
        'το γέμισμα του ComboBoxCOMPorts
        For i As Integer = 0 To _
           My.Computer.Ports.SerialPortNames.Count - 1
            ComboBoxComPorts.Items.Add(My.Computer.Ports.SerialPortNames(i))
        Next

        'Ορισμός Baudrates και πρόσθεση τους στο ComboBoxBaudRate
        Dim baudrates() As String = {"230400", "115200", "57600", "38400",
                                     "19200", "9600"}
```

```
        For Each aBaudrate In baudrates
            ComboBoxBaudRate.Items.Add(aBaudrate)
        Next
        ComboBoxBaudRate.SelectedIndex = 5

    End Sub

    Private Sub ButtonStartMaster_Click(ByVal sender As System.Object,
              ByVal e As System.EventArgs) Handles ButtonStartMaster.Click
        'Εκκίνηση Modbus Master
        'Καθορισμός παραμέτρων σειριακής πόρτας
        MyMasterPort.PortName = Me.ComboBoxComPorts.SelectedItem
        MyMasterPort.BaudRate = Me.ComboBoxBaudRate.SelectedItem
        MyMasterPort.DataBits = 8
        MyMasterPort.Parity = Parity.None
        MyMasterPort.StopBits = StopBits.One
        MyMasterPort.DataBits = 8
        MyMasterPort.Parity = Parity.None
        MyMasterPort.StopBits = StopBits.One
        'Άνοιγμα σειριακής θύρας
        MyMasterPort.Open()
        'Δημιουργία αντικειμένου Modbus Slave αναλόγως του
        'επιλεγμένου τύπου ASCII ή RTU
        MyMaster = ModbusSerialMaster.CreateRTU(MyMasterPort)
        'Ενεργοποίηση πλήκτρων δεδομένων
        Me.ButtonRefreshData.Enabled = True
        Me.ButtonSetData.Enabled = True
        'Ενεργοποίηση πλήκτρου τερματισμού Slave
        Me.ButtonStopMaster.Enabled = True
        'Απενεργοποίηση πλήκτρου εκκίνησης Modbus Slave
        Me.ButtonStartMaster.Enabled = False
    End Sub

    Private Sub ButtonRefreshData_Click(ByVal sender As System.Object,
              ByVal e As System.EventArgs) Handles ButtonRefreshData.Click
        ReadData()
    End Sub

    Public Sub ReadData()
        'Ανανέωση των στοιχείων ελέγχου του παραθύρου
        'με βάση τις τιμές τις οποίες αναφέριε ο Modbus Slave
        'Καθορισμός αναγνωριστικού σταθμού Modbus Slave
        Dim slaveID As Byte = Me.TextBoxSerialSlaveID.Text
        'Αποστολή μηνυμάτων Modbus για το διάβασμα δεδομένων

        'Διάβασμα και εμφάνιση κατάστασης του πρώτου Holding Register
        Dim HoldingRegisters() As UShort =
                MyMaster.ReadHoldingRegisters(slaveID,
                                Me.TextBoxStartAddress.Text,
                                Me.TextBoxNumOfRegisters.Text)
        Me.TextBoxHoldingRegister1.Text = HoldingRegisters(0)

    End Sub


    Private Sub ButtonSetData_Click(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles ButtonSetData.Click
        'Καθορισμός αναγνωριστικού σταθμού Modbus Slave
        Dim slaveID As Byte = Me.TextBoxSerialSlaveID.Text
        MyMaster.WriteHoldingRegisters(slaveID,
                            Me.TextBoxStartAddress.Text, 1,
                                        {Me.TextBoxHoldingRegister1.Text})
    End Sub
```

```
    Private Sub ButtonStopMaster_Click(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles ButtonStopMaster.Click
        'Τερματισμός του Modbus Master
        'Κλείσιμο και αποδέσμευση σειριακής θύρας
        Me.MyMasterPort.Close()
        'Απενεργοποίηση πλήκτρων δεδομένων
        Me.ButtonRefreshData.Enabled = False
        Me.ButtonSetData.Enabled = False
        'Απενεργοποίηση πλήκτρου τερματισμού Slave
        Me.ButtonStopMaster.Enabled = False
        'Ενεργοποίηση πλήκτρου εκκίνησης Modbus Slave
        Me.ButtonStartMaster.Enabled = True
    End Sub


End Class
```

6.2 Η βιβλιοθήκη NModbus

Η βιβλιοθήκη NModbus είναι μια βιβλιοθήκη ανοικτού κώδικα (open source) η οποία παρέχει τις βασικές κλάσεις αντικειμένων και την απαραίτητη λειτουργικότητα για την ανάπτυξη εφαρμογών Modbus χωρίς να χρειάζεται να υλοποιηθεί το πρωτόκολλο Modbus εξ αρχής όπως έγινε στην προηγούμενη ενότητα 17.4.



Εικόνα 6.3 Προσθήκη αναφοράς στην βιβλιοθήκη NModbus

Για την χρήση της βιβλιοθήκης από μία εφαρμογή πρέπει να αντιγραφεί το αρχείο modbus.dll σε κάποιον

κατάλογο του Η/Υ (κατά προτίμηση στον κατάλογο της εφαρμογής) και να δηλωθεί ότι η εφαρμογή χρησιμοποιεί αυτό το αρχείο. Η προσθήκη αναφοράς στην βιβλιοθήκη NModbus γίνεται την εντολή "Add Reference" από το μενού εντολών "Project" και την επιλογή του αρχείου modbus.dll στο παράθυρο διαλόγου το οποίο θα εμφανισθεί (εικόνα 17.4).

6.3 Ανάπτυξη εφαρμογών RTU και ASCII Modbus

Για την ανάπτυξης εφαρμογών RTU ή ASCII Modbus Master με την χρήση της βιβλιοθήκης NModbus χρησιμοποιείται η κλάση αντικειμένων ModbusSerialMaster. Οι βασικές μέθοδοι της κλάσης αυτής δίνονται από τον παρακάτω πίνακα.

Κλάση Αντικειμένων ModbusSerialMaster

Μέθοδοι	Περιγραφή
CreateAscii (MyMasterPort as SerialPort)	Η μέθοδος αυτή δημιουργεί ένα νέο αντικείμενο Modbus Master για σειριακή επικοινωνία Modbus ACSII. Δέχεται ως παράμετρο ένα αντικείμενο SerialPort στο οποίο προηγουμένως έχουν γίνει οι κατάλληλες ρυθμίσεις και ή θύρα είναι ανοικτή για επικοινωνία.
CreateRTU (MyMasterPort as SerialPort)	Η μέθοδος αυτή δημιουργεί ένα νέο αντικείμενο Modbus Master για σειριακή επικοινωνία Modbus RTU. Δέχεται ως παράμετρο ένα αντικείμενο SerialPort στο οποίο προηγουμένως έχουν γίνει οι κατάλληλες ρυθμίσεις και ή θύρα είναι ανοικτή για επικοινωνία.
ReadCoils(slaveAddress as Byte, startAddress as UShort, numOfOPoints as Ushort) as Boolean ( )	Η μέθοδος αυτή διαβάζει την κατάσταση των Coils του Modbus Slave ο οποίος καθορίζεται από την διεύθυνση SlaveAddress. Η παράμετρος startAddress καθορίζει την διεύθυνση αρχής των Coils και η παράμετρος numOfPoints καθορίζει το πλήθος των δεδομένων (Coils)τα οποία θα διαβαστούν. Η μέθοδος αυτή επιστρέφει ένα πίνακα λογικών τιμών (boolean) οι οποίες αντιστοιχούν στην κατάσταση των Coils του Modbus Slave.
ReadInputs(slaveAddress as Byte, startAddress as UShort, numOfOPoints as Ushort) as Boolean ( )	Η μέθοδος αυτή διαβάζει την κατάσταση των Input Discrete Coils του Modbus Slave. Οι παράμετροι της μεθόδου είναι παρόμοιες με την μέθοδο ReadCoils. Η μέθοδος αυτή επιστρέφει ένα πίνακα λογικών τιμών (boolean) ο οποίος περιέχει τις τιμές των Input Discrete Coils του Modbus Slave.

ReadHoldingRegisters(slaveAddress as Byte, startAddress as UShort, numOfOPoints as Ushort) as UShort ( )	Η μέθοδος αυτή διαβάζει την κατάσταση των Holding Registers του Modbus Slave ο οποίος καθορίζεται από την διεύθυνση SlaveAddress. Η παράμετρος startAddress καθορίζει την διεύθυνση αρχής των Holding Registers και η παράμετρος numOfPoints καθορίζει το πλήθος των δεδομένων (Holding Registers) τα οποία θα διαβαστούν. Η μέθοδος αυτή επιστρέφει ένα πίνακα τιμών τύπου Ushort οι οποίες αντιστοιχούν στην κατάσταση των Holding Registers του Modbus Slave.
ReadCoils(slaveAddress as Byte, startAddress as UShort, numOfOPoints as Ushort) as Boolean ( )	Η μέθοδος αυτή διαβάζει την κατάσταση των Input Registers του Modbus Slave. Οι παράμετροι της μεθόδου είναι παρόμοιες με την μέθοδο ReadHoldingRegisters. Η μέθοδος αυτή επιστρέφει ένα πίνακα τιμών τύπου Ushort οι οποίες αντιστοιχούν στην κατάσταση των Input Registers του Modbus Slave.
WriteSingleCoil(slaveAddress as Byte, coilAddress as UShort, value as Boolean)	Η μέθοδος αυτή καταχωρεί την τιμή η οποία δίνεται από την παράμετρο value στο Coil το οποίο αντιστοιχεί διεύθυνση coilAddress του Modbus Slave με την διεύθυνση slaveAddress.
WriteMultipleCoils(slaveAddress as Byte, startAddress as UShort, data as Boolean())	Η μέθοδος αυτή καταχωρεί τις τιμές οι οποίες δίνονται από τον πίνακα τιμών data στα Coils με αρχική διεύθυνση startAddress του Modbus Slave του οποίου η διεύθυνση δίνεται από την παράμετρο slaveAddress.
WriteSingleRegister(slaveAddress as Byte, registerAddress as UShort, value as UShort)	Η μέθοδος αυτή καταχωρεί την τιμή η οποία δίνεται από την παράμετρο value στον Holding Register το οποίο αντιστοιχεί διεύθυνση registerAddress του Modbus Slave με την διεύθυνση slaveAddress.
WriteMultipleRegisters(slaveAddress as Byte, startAddress as UShort, data as UShort())	Η μέθοδος αυτή καταχωρεί τις τιμές οι οποίες δίνονται από τον πίνακα τιμών data στα Holding Registers με αρχική διεύθυνση startAddress του Modbus Slave του οποίου η διεύθυνση δίνεται από την παράμετρο slaveAddress.

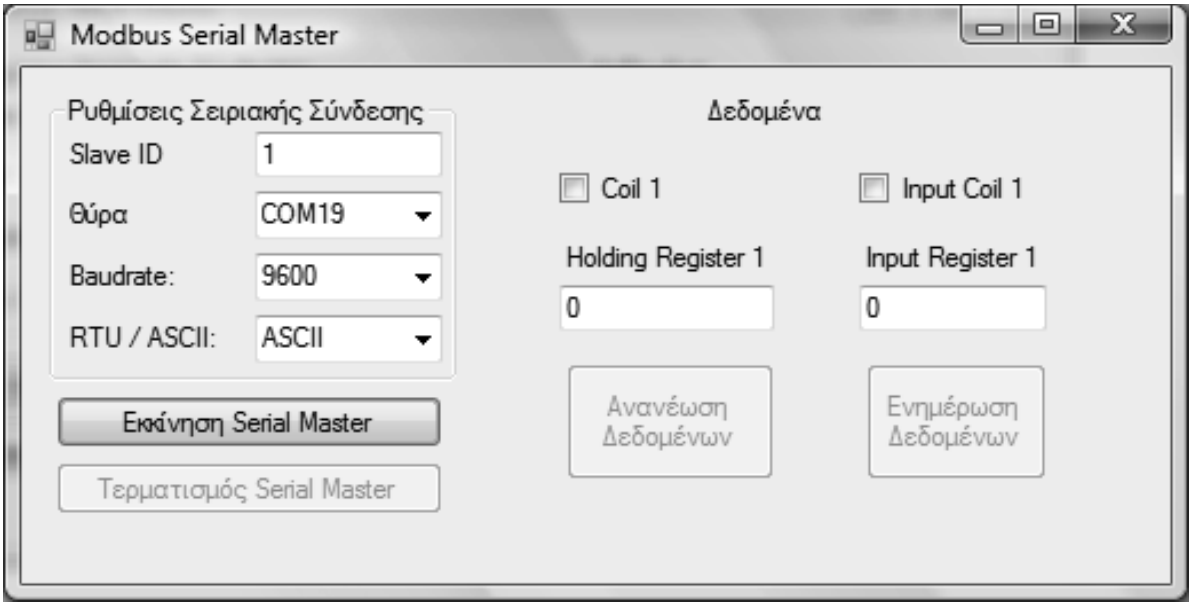
Για την ανάπτυξης εφαρμογών RTU ή ASCII Modbus Slave χρησιμοποιείται η κλάση αντικειμένων ModbusSerialSlave της οποίας οι κυριότεροι μέθοδοι φαίνονται στον παρακάτω πίνακα.

Κλάση Αντικειμένων ModbusSlave

Ιδιοότητες	Περιγραφή
DataStore	Αντικείμενο της κλάσης DataStore όπου αποθηκεύονται οι τιμές των Coils και Registers του Modbus Slave. Για την δημιουργία ενός αντικειμένου και την απόδοση αρχικής τιμής στα δεδομένα χρησιμοποιείται η παρακάτω εντολή: MySlave.DataStore = DataStoreFactory.CreateDefaultDataStore() Η ιδιότητα DataStore έχει τις ιδιότητες CoilDiscretes, InputRegisters, HoldingRegisters και InputRegisters με την χρήση των οποίων γίνεται η πρόσβαση στις τιμές των δεδομένων. Παράδειγμα απόδοσης τιμής στα δεδομένα: MySlave.DataStore.InputDiscretes(1) = true MySlave.DataStore.CoilDiscretes(1)=true MySlave.DataStore.HoldingRegisters(1)=12 MySlave.DataStore.InputRegisters(1)=14
Μέθοδοι	**Περιγραφή**
CreateAscii (unitId as Byte, MySlavePort as SerialPort)	Η μέθοδος αυτή δημιουργεί ένα νέο αντικείμενο ModbusSerialSlave για σειριακή επικοινωνία Modbus ACSII. Δέχεται ως παράμετρο την διεύθυνση του Slave η οποία δίνεται από την παράμετρο unitId και ένα αντικείμενο τύπου SerialPort στο οποίο προηγουμένως έχουν γίνει οι κατάλληλες ρυθμίσεις και ή θύρα είναι ανοικτή για επικοινωνία.
CreateRTU (unitId as Byte, MySlavePort as SerialPort)	Η μέθοδος αυτή δημιουργεί ένα νέο αντικείμενο ModbusSerialSlave για σειριακή επικοινωνία Modbus RTU. Δέχεται ως παράμετρο την διεύθυνση του Slave και ένα αντικείμενο τύπου SerialPort.
Listen	Με την μέθοδο αυτή το αντικείμενο Modbus Slave διαβάζει και απαντά στο μήνυμα το οποίο θα αποσταλεί από τον Modbus Master

Παράδειγμα εφαρμογής Modbus Serial Master

Η εφαρμογή η οποία φαίνεται στην παρακάτω εικόνα λειτουργεί ώς Modbus Serial Master με δυνατότητα επιλογής του πρωτοκόλλου RTU ή ASCII για την επικοινωνία.



Εικόνα 6.4 Εφαρμογή Modbus Serial Master

Το παράθυρο FormModbusMaster έχει τα παρακάτω στοιχεία ελέγχου:

Όνομα	Τύπος	Περιγραφή
TextBoxSerialSlaveID	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την διεύθυνση του σταθμού RTU Slave με την οποία θα επικοινωνήσει
ComboBoxComPorts	ComboBox	Χρησιμοποιείται για να εμφανίσει όλες τις διαθέσιμες σειριακές θύρες και την επιλογή από τον χρήστη της θύρας για την σειριακή επικοινωνία
ComboBoxBaudRate	ComboBox	Χρησιμοποιείται για την επιλογή της ταχύτητας της σειριακής επικοινωνίας
ComboBoxMode	ComboBox	Χρησιμοποιείται για την επιλογή του τύπου επικοινωνίας RTU ή ASCII
CheckBoxCoil1	CheckBox	Χρησιμοποιείται για την εμφάνιση/τροποποίηση της τιμής του Coil με διεύθυνση 1 του Modbus Slave.
CheckBoxInputCoil1	CheckBox	Χρησιμοποιείται για την εμφάνιση της τιμής του Input Descrete Coil με διεύθυνση 1 του Modbus Slave.
TextBoxHoldingRegister1	TextBox	Χρησιμοποιείται για την εμφάνιση/τροποποίηση της τιμής του Holding Register με διεύθυνση 1 του Modbus Slave.

TextBoxInputRegister1	TextBox	Χρησιμοποιείται για την εμφάνιση της τιμής του Input Register με διεύθυνση 1 του Modbus Slave.
ButtonStartMaster	Button	Πλήκτρο για την εκκίνηση του Modbus Serial Master
ButtonStopMaster	Button	Πλήκτρο για τον τερματισμό του Modbus Serial Master
ButtonRefreshData	Button	Πλήκτρο για την ανάγνωση των τιμών του Modbus Slave και εμφάνισης τους στα αντίστοιχα στοιχεία ελέγχου του παραθύρου.
ButtonSetData	Button	Πλήκτρο για την ενημέρωση των τιμών του Coil με διεύθυνση 1 και Holding Register με διεύθυνση 1 του Modbus Slave σύμφωνα με τις τιμές τις οποίες έχει καθορίσει ο χρήστης στα στοιχεία ελέγχου CheckBox1 και TexBoxHoldingRegister1.

Πίνακας 6.2 Τα στοιχεία ελέγχου του Modbus Serial Master

Ο κώδικας για την εφαρμογή ModbusSerialMaster είναι ο παρακάτω:

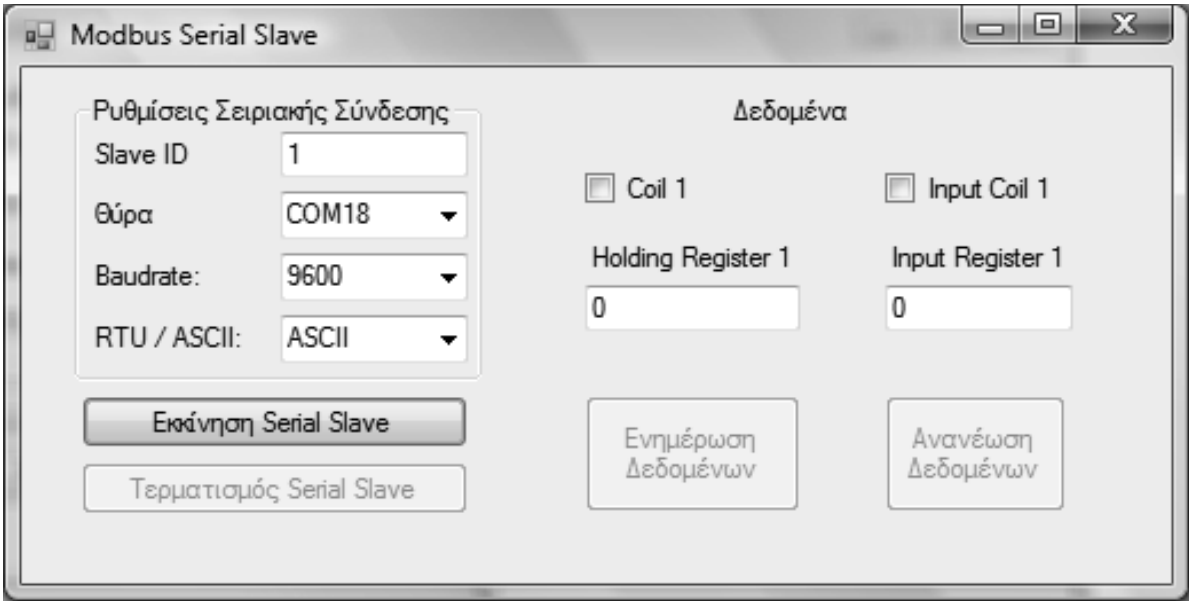
```
Imports Modbus
Imports System.IO.Ports
Imports Modbus.Device
Imports Modbus.Data

Public Class FormModbusMaster
    'Ορισμός σειριακής θύρας επικοινωνιών
    Dim MyMasterPort As New SerialPort
    'Ορισμός αντικειμένου Modbus Master
    Dim MyMaster As ModbusSerialMaster

    Private Sub FormMaster_Load(ByVal sender As System.Object,
                        ByVal e As System.EventArgs) Handles MyBase.Load
        'Χρήση της μεθόδου Computer.Ports.SerialPortNames η οποία
        'επιστρέφει μια συλλογή με τις διαθέσιμες θύρες του Η/Υ για
        'το γέμισμα του ComboBoxCOMPorts
        For i As Integer = 0 To _
           My.Computer.Ports.SerialPortNames.Count - 1
            ComboBoxComPorts.Items.Add(My.Computer.Ports.SerialPortNames(i))
        Next

        'Ορισμός Baudrates και πρόσθεση τους στο ComboBoxBaudRate
        Dim baudrates() As String = {"230400", "115200", "57600", "38400",
                                     "19200", "9600"}
        For Each aBaudrate In baudrates
            ComboBoxBaudrate.Items.Add(aBaudrate)
        Next
        ComboBoxBaudrate.SelectedIndex = 5

        'Ορισμός στοιχείων του ComboBoxMode για την
        'επιλογή του τύπου σειραικής επικοινωνίας
        Me.ComboBoxMode.Items.Add("RTU")
        Me.ComboBoxMode.Items.Add("ASCII")
        Me.ComboBoxMode.SelectedIndex = 1
    End Sub

    Private Sub ButtonStartMaster_Click(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles ButtonStartMaster.Click
```

```
        'Εκκίνηση Modbus Master
        'Καθορισμός παραμέτρων σειριακής πόρτας
        MyMasterPort.PortName = Me.ComboBoxComPorts.SelectedItem
        MyMasterPort.BaudRate = Me.ComboBoxBaudrate.SelectedItem
        MyMasterPort.DataBits = 8
        MyMasterPort.Parity = Parity.None
        MyMasterPort.StopBits = StopBits.One
        MyMasterPort.DataBits = 8
        MyMasterPort.Parity = Parity.None
        MyMasterPort.StopBits = StopBits.One
        'Άνοιγμα σειριακής θύρας
        MyMasterPort.Open()
        'Δημιουργία αντικειμένου Modbus Slave αναλόγως του
        'επιλεγμένου τύπου ASCII ή RTU
        If Me.ComboBoxMode.SelectedItem = "ASCII" Then
            MyMaster = ModbusSerialMaster.CreateAscii(MyMasterPort)
        Else
            MyMaster = ModbusSerialMaster.CreateRtu(MyMasterPort)
        End If
        'Ενεργοποίηση πλήκτρων δεδομένων
        Me.ButtonRefreshData.Enabled = True
        Me.ButtonSetData.Enabled = True
        'Ενεργοποίηση πλήκτρου τερματισμού Slave
        Me.ButtonStopSlave.Enabled = True
        'Απενεργοποίηση πλήκτρου εκκίνησης Modbus Slave
        Me.ButtonStartMaster.Enabled = False
    End Sub

    Private Sub ButtonSetData_Click(ByVal sender As System.Object,
                    ByVal e As System.EventArgs) Handles ButtonSetData.Click
        'Αποστολή νέων τιμών δεδομένων στον Modbus Slave
        Me.WriteData()
    End Sub

    Private Sub ButtonRefreshData_Click(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles ButtonRefreshData.Click
        'Διάβασμα και ανανέωση δεδομένων
        Me.ReadData()
    End Sub

    Public Sub ReadData()
        'Ανανέωση των στοιχείων ελέγχου του παραθύρου
        'με βάση τις τιμές τις οποίες αναφέρει ο Modbus Slave
        'Καθορισμός αναγνωριστικού σταθμού Modbus Slave
        Dim slaveID As Byte = Me.TextBoxSerialSlaveID.Text
        'Απστολή μηνυμάτων Modbus για το διάβασμα δεδομένων

        'Χρήση της μεθόδου ReadCoils για την επιστροφή πίνακα
        'με τις τιμές των Coils με διεύθυνση αρχής 0 και πλήθος
        'τιμών οι οποίες θα διαβασθούν 1
        Dim Coils() As Boolean =
                    MyMaster.ReadCoils(slaveID, 0, 1)
        'Εμφάνιση τιμής του πρώτου coil
        Me.CheckBoxCoil1.Checked = Coils(0)
        'Διάβασμα και εμφάνιση κατάστασης του πρώτου Input Coil
        Dim InputCoils() As Boolean =
                    MyMaster.ReadInputs(slaveID, 0, 1)
        Me.CheckBoxInputCoil1.Checked = InputCoils(0)
        'Διάβασμα και εμφάνιση κατάστασης του πρώτου Holding Register
        Dim HoldingRegisters() As UShort =
                    MyMaster.ReadHoldingRegisters(slaveID, 0, 20)
        Me.TextBoxHoldingRegister1.Text = HoldingRegisters(0)
        'Διάβασμα και εμφάνιση κατάστασης του πρώτου Input Register
        Dim InputRegisters() As UShort =
```

```
                MyMaster.ReadInputRegisters(slaveID, 0, 20)
        Me.TextBoxInputRegister1.Text = InputRegisters(0)

    End Sub

    Public Sub WriteData()
        'Ενημέρωση Slave με νέες τιμές για Coil και Holding Register
        'Καθορισμός αναγνωριστικού σταθμού Modbus Slave
        Dim slaveID As Byte = Me.TextBoxSerialSlaveID.Text
        'Αλλαγή τιμής του Coil με διεύθυνση 0
        MyMaster.WriteSingleCoil(slaveID, 0, Me.CheckBoxCoil1.Checked)
        'Αλλαγή τιμής του Holding Register με διεύθυνση 0
        MyMaster.WriteSingleRegister(slaveID, 0,
                                     Me.TextBoxHoldingRegister1.Text)
     End Sub

    Private Sub ButtonStopSlave_Click(ByVal sender As System.Object,
                   ByVal e As System.EventArgs) Handles ButtonStopSlave.Click
        'Τερματισμός του Modbus Master
        'Κλείσιμο και αποδέσμευση σειριακής θύρας
        Me.MyMasterPort.Close()
        'Απενεργοποίηση πλήκτρων δεδομένων
        Me.ButtonRefreshData.Enabled = False
        Me.ButtonSetData.Enabled = False
        'Απενεργοποίηση πλήκτρου τερματισμού Slave
        Me.ButtonStopSlave.Enabled = False
        'Ενεργοποίηση πλήκτρου εκκίνησης Modbus Slave
        Me.ButtonStartMaster.Enabled = True
    End Sub
End Class
```

Παράδειγμα εφαρμογής Modbus Serial Slave

Η εφαρμογή η οποία φαίνεται στην παρακάτω εικόνα λειτουργεί ώς Modbus Serial Slave με δυνατότητα επιλογής του πρωτοκόλλου RTU ή ASCII για την επικοινωνία.



Εικόνα 6.5 Εφαρμογή Modbus Serial Slave

Το παράθυρο FormModbusSlave έχει τα παρακάτω στοιχεία ελέγχου:

Όνομα	Τύπος	Περιγραφή
TextBoxSerialSlaveID	TextBox	Πλαίσιο κειμένου όπου ο χρήστης πληκτρολογεί την διεύθυνση του σταθμού RTU Slave με την οποία θα επικοινωνήσει
ComboBoxComPorts	ComboBox	Χρησιμοποιείται για να εμφανίσει όλες τις διαθέσιμες σειριακές θύρες και την επιλογή από τον χρήστη της θύρας για την σειριακή επικοινωνία
ComboBoxBaudRate	ComboBox	Χρησιμοποιείται για την επιλογή της ταχύτητας της σειριακής επικοινωνίας
ComboBoxMode	ComboBox	Χρησιμοποιείται για την επιλογή του τύπου επικοινωνίας RTU ή ASCII
CheckBoxCoil1	CheckBox	Χρησιμοποιείται για την εμφάνιση/τροποποίηση της τιμής του Coil με διεύθυνση 1
CheckBoxInputCoil1	CheckBox	Χρησιμοποιείται για την εμφάνιση/τροποποίηση της τιμής του Input Descrete Coil με διεύθυνση 1
TextBoxHoldingRegister1	TextBox	Χρησιμοποιείται για την εμφάνιση/τροποποίηση της τιμής του Holding Register με διεύθυνση 1 του Modbus Slave.
TextBoxInputRegister1	TextBox	Χρησιμοποιείται για την εμφάνιση/τροποποίηση της τιμής του Input Register με διεύθυνση 1.
ButtonStartSlave	Button	Πλήκτρο για την εκκίνηση του Modbus Serial Slave
ButtonStopSlave	Button	Πλήκτρο για τον τερματισμό του Modbus Serial Slave
ButtonRefreshData	Button	Πλήκτρο για την ανάγνωση των τιμών του Modbus Slave και εμφάνισης τους στα αντίστοιχα στοιχεία ελέγχου του παραθύρου.
ButtonSetData	Button	Πλήκτρο για την ενημέρωση των τιμών των στοιχείων ελέγχου του παραθύρου.

Πίνακας 6.3 Τα στοιχεία ελέγχου του Modbus Serial Slave

Ο κώδικας για την εφαρμογή ModbusSerialSlave είναι ο παρακάτω:

```
Imports Modbus
Imports System.IO.Ports
Imports Modbus.Device
Imports Modbus.Data
Imports System.Threading

Public Class FormModbusSlave
    'Ορισμός σειριακής θύρας επικοινωνιών
    Dim MySlavePort As New SerialPort
    'Ορισμός αντικειμένου Modbus Slave
    Dim MySlave As ModbusSerialSlave
    'Ορισμός νήματος για την επικοινωνία με μηνύματα Modbus
```

```
Dim ListenThread As Thread

Private Sub FormModbusSlave_Load(ByVal sender As System.Object,
                        ByVal e As System.EventArgs) Handles MyBase.Load
    'Χρήση της μεθόδου Computer.Ports.SerialPortNames η οποία
    'επιστρέφει μια συλλογή με τις διαθέσιμες θύρες του Η/Υ για
    'το γέμισμα του ComboBoxCOMPorts
    For i As Integer = 0 To _
       My.Computer.Ports.SerialPortNames.Count - 1
        ComboBoxComPorts.Items.Add(My.Computer.Ports.SerialPortNames(i))
    Next

    'Ορισμός Baudrates και πρόσθεση τους στο ComboBoxBaudRate
    Dim baudrates() As String = {"230400", "115200", "57600", "38400",
                                 "19200", "9600"}
    For Each aBaudrate In baudrates
        ComboBoxBaudrate.Items.Add(aBaudrate)
    Next
    ComboBoxBaudrate.SelectedIndex = 5

    'Ορισμός στοιχείων του ComboBoxMode για την
    'επιλογή του τύπου σειραικής επικοινωνίας
    Me.ComboBoxMode.Items.Add("RTU")
    Me.ComboBoxMode.Items.Add("ASCII")
    Me.ComboBoxMode.SelectedIndex = 1
End Sub

Private Sub ButtonStartSlave_Click(ByVal sender As System.Object,
             ByVal e As System.EventArgs) Handles ButtonStartSlave.Click
    'Εκκίνηση Modbus Slave
    Me.CreateSerialSlave()
    Me.SetData()
    Me.StartSlave()
    'Ενεργοποίηση πλήκτρων δεδομένων
    Me.ButtonRefreshData.Enabled = True
    Me.ButtonSetData.Enabled = True
    'Ενεργοποίηση πλήκτρου τερματισμού Slave
    Me.ButtonStopSlave.Enabled = True
    'Απενεργοποίηση πλήκτρου εκκίνησης Modbus Slave
    Me.ButtonStartSlave.Enabled = False
End Sub

Private Sub CreateSerialSlave()
    'Καθορισμός παραμέτρων σειριακής πόρτας
    MySlavePort.PortName = Me.ComboBoxComPorts.SelectedItem
    MySlavePort.BaudRate = Me.ComboBoxBaudrate.SelectedItem
    MySlavePort.DataBits = 8
    MySlavePort.Parity = Parity.None
    MySlavePort.StopBits = StopBits.One
    'Άνοιγμα σειριακής θύρας
    MySlavePort.Open()
    'Καθορισμός αγναγνωριστικού Modbus Slave
    Dim unitID As Byte = Me.TextBoxSerialSlaveID.Text
    'Δημιουργία αντικειμένου Modbus Slave αναλόγως του
    'επιλεγμένου τύπου ASCII ή RTU
    If Me.ComboBoxMode.SelectedItem = "ASCII" Then
        MySlave = ModbusSerialSlave.CreateAscii(unitID, MySlavePort)
    Else
        MySlave = ModbusSerialSlave.CreateRtu(unitID, MySlavePort)
    End If
    'Δημιουργία και αρχικοποίηση δεδομένων του Modbus Slave
    MySlave.DataStore = DataStoreFactory.CreateDefaultDataStore()
End Sub
```

```
    Private Sub StartSlave()
        'Εκκίνηση νήματος για την επικοινωνία με Modbus Master
        'και την αποστολή / λήψη δεδομένων
        Me.ListenThread = New Thread(AddressOf Listen)
        Me.ListenThread.Start()
    End Sub

    Private Sub Listen()
        'Διαδικασία η οποία εκτελείται από το νήμα ListenTread
        Do
            'Στην περίπτωση που η σειριακή θύρα είναι κλειστή
            'τότε έξοδος της διαδικασίας και τερματισμός του νήματος
            If Not MySlavePort.IsOpen Then Exit Sub
            'Αναμονή και απάντηση σε εισερχόμενα μηνύματα Modbus
            MySlave.Listen()
        Loop
    End Sub
    Private Sub ButtonSetData_Click(ByVal sender As System.Object,
                    ByVal e As System.EventArgs) Handles ButtonSetData.Click
        'Ενημέρωση δεδομένων
        Me.SetData()
    End Sub
    Private Sub ButtonRefreshData_Click_1(ByVal sender As System.Object,
                ByVal e As System.EventArgs) Handles ButtonRefreshData.Click
        'Ανανέωση δεδομένων
        Me.RefreshData()
    End Sub

    Public Sub RefreshData()
        'Ανανέωση των στοιχείων ελέγχου του παραθύρου
        'με βάση τις τιμές της βάσης δεδομένων του Modbus Slave
        Me.CheckBoxInputCoil1.Checked = MySlave.DataStore.InputDiscretes(1)
        Me.TextBoxInputRegister1.Text = MySlave.DataStore.InputRegisters(1)
        Me.CheckBoxCoil1.Checked = MySlave.DataStore.CoilDiscretes(1)
        Me.TextBoxHoldingRegister1.Text =
                                    MySlave.DataStore.HoldingRegisters(1)
    End Sub
    Private Sub SetData()
        'Ενημέρωση της βάσης δεδομένων του Modbus Slave με
        'βάση την τιμές των στοιχείων ελέγχου του παραθύρου
        MySlave.DataStore.CoilDiscretes(1) = Me.CheckBoxCoil1.Checked
        MySlave.DataStore.HoldingRegisters(1) =
                                        Me.TextBoxHoldingRegister1.Text
        MySlave.DataStore.InputDiscretes(1) =
                                        Me.CheckBoxInputCoil1.Checked
        MySlave.DataStore.InputRegisters(1) =
                                        Me.TextBoxInputRegister1.Text
    End Sub

    Private Sub ButtonStopSlave_Click(ByVal sender As System.Object,
                  ByVal e As System.EventArgs) Handles ButtonStopSlave.Click
        'Τερματισμός του Modbus Slave
        'Κλείσιμο και αποδέσμευση σειριακής θύρας
        Me.MySlavePort.Close()
        'Απενεργοποίηση πλήκτρων δεδομένων
        Me.ButtonRefreshData.Enabled = False
        Me.ButtonSetData.Enabled = False
        'Απενεργοποίηση πλήκτρου τερματισμού Slave
        Me.ButtonStopSlave.Enabled = False
        'Ενεργοποίηση πλήκτρου εκκίνησης Modbus Slave
        Me.ButtonStartSlave.Enabled = True
    End Sub

End Class
```

6.4 Ανάπτυξη εφαρμογών TCP Modbus

Για την ανάπτυξης εφαρμογών TCP Modbus Master με την χρήση της βιβλιοθήκης NModbus χρησιμοποιείται η κλάση αντικειμένων ModbusIPMaster. Οι βασικές μέθοδοι της κλάσης αυτής για την ανάγνωση και εγγραφή δεδομένων είναι παρόμοιες με την κλάση ModbusSerialMaster,με διαφορά στην μέθοδο για την δημιουργία νέου αντικειμένου όπου χρησιμοποιείται η μέθοδος CreateIpMaster. Η μέθοδος αυτή δέχεται ως παράμετρο ένα αντικείμενο τύπου TCPClient στο οποίο έχουν γίνει οι κατάλληλες ρυθμίσεις για την επικοινωνία με τον Modbus TCP Slave.

Παράδειγμα δημιουργίας νέου αντικειμένου:

```
MyMaster = ModbusIpMaster.CreateIp(MyTcpClient)
```

Για την ανάπτυξης εφαρμογών TCP Modbus Slave χρησιμοποιείται η κλάση αντικειμένων ModbusTCPSlave. Οι μέθοδοι της κλάσης αυτής για την ανάγνωση και εγγραφή δεδομένων είναι παρόμοιες με την κλάση ModbusSerialSlave.Η μέθοδος για την δημιουργία νέων αντικειμένων της κλάσης αυτής είναι η CreateTCP η οποία δέχεται ως παραμέτρους ένα αντικείμενο τύπου TCP Listener με τις κατάλληλες TCP ρυθμίσεις για την σύνδεση με εφαρμογές Modbus TCP Master.

Παράδειγμα δημιουργίας νέου αντικειμένου :

```
MySlave = ModbusTcpSlave.CreateTcp(slaveID, MyTcpListener)
```