ΑΛΕΞΑΝΔΡΕΙΟ ΤΕΧΝΟΛΟΓΙΚΟ
ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΘΕΣΣΑΛΟΝΙΚΗΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΗΣ

Κατασκευή συστήματος ένδειξης και καταγραφής ταχύτητος και κατεύθυνσης ανέμου με αισθητήρες υπερήχων

ΕΛΣΙΑΝΛΗΣ ΘΕΟΔΩΡΟΣ
Κ.Α.Σ. 501017

ΚΩΣΤΑΚΗΣ ΕΜΜΑΝΟΥΗΛ
Κ.Α.Σ. 501038

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: Α. Χατζηγκάιδας

ΘΕΣΣΑΛΟΝΙΚΗ 2008

Ο σκοπός αυτής της εργασίας είναι η υλοποίηση ενός συστήματος το οποίο θα μας παρέχει ένδειξη της ταχύτητας και της κατεύθυνσης του ανέμου χρησιμοποιώντας αισθητήρες υπερήχων. Οι αισθητήρες αυτοί συνδέονται στον Ηλεκτρονικό Υπολογιστή μέσω της θύρας RS-232. Χρησιμοποιήσαμε τη γλώσσα προγραμματισμού Visual Basic 6.0 Enterprise Edition για τη δημιουργία ενός προγράμματος το οποίο μας παρέχει γραφική απεικόνιση των υπό μέτρηση μεγεθών.

Η κατεύθυνση του ανέμου εμφανίζεται σε μοίρες και σε μορφή πυξίδας με τη χρήση ενός αντικειμένου FLASH. Η ένταση του ανέμου αναγράφεται σε 1) Χιλιόμετρα ανά ώρα, 2) Μίλια ανά ώρα, 3) Κόμβους και 4) στην κλίμακα Beaufort. Η γραφική τους απεικόνιση γίνεται σε μορφή Progress Bar. Μέσα από το Software έχουμε τη δυνατότητα να επιλέξουμε τον αριθμό της σειριακής θύρας με την οποία θα επικοινωνεί ο υπολογιστής με τον αισθητήρα καθώς επίσης και να ξεκινάμε και να διακόπτουμε την επικοινωνία χρησιμοποιώντας ένα κουμπί εντολής (Command Button). Τα δεδομένα που παίρνουμε από τον αισθητήρα ελέγχονται ως προς την ορθότητά τους και έπειτα εμφανίζονται στη οθόνη. Αν τα δεδομένα είναι λανθασμένα τότε στην οθόνη εμφανίζεται ένα προειδοποιητικό σήμα λάθους. Για την υλοποίηση του ελέγχου αυτού έχουμε χρησιμοποιήσει στον κώδικα του προγράμματος την τεχνική του Κυκλικού Ελέγχου Πλεονασμού (Cyclic Redundancy Check).

Το πρόγραμμα αυτό θα μπορούσε να χρησιμοποιηθεί σε λιμάνια, σε γέφυρες πλοίων, σε πλωτές εξέδρες, σε φάρους και σε αεροδρόμια γιατί σε κάθε μία από αυτές τις περιπτώσεις είναι αναγκαίο ανά πάσα στιγμή να γνωρίζουμε τις συνθήκες του ανέμου που πνέει στην περιοχή.

The purpose of this project is the materialization of a system which will supply us with indication of the wind's speed and direction by using ultrasonic sensors. These sensors are connected to the PC through an RS-232 port. We used the Visual Basic 6.0 Enterprise Edition programming language for the creation of a software which supplies us with a graphical picture of the under measurement magnitudes.

The wind's direction is shown in degrees and in a compass form by using a FLASH object. The wind's intensity is written down in 1) Kilometers per hour, 2) Miles per hour, 3) Knots and 4) in Beaufort scale. Their graphical picture is shown in progress bar form. Through the Software we have the possibility to choose the number of the serial port with which the PC will communicate with the Sensors as well as starting and interrupting the communication by using a Command Button. The data we receive from the sensors are checked for their accuracy and then are shown on the screen. If these data are corrupted then on the screen appears a mistake warning sign. In order for this check to be carried out we used in our software's source code the CRC technique (Cyclic Redundancy Check).

This software could be used in harbors, ship's bridges, floating platforms, lighthouses and in airports because to each one of these cases is necessary at any time that we are aware of the conditions of a blowing wind in the area.

ΠΕΡΙΕΧΟΜΕΝΑ

ΕΙΣΑΓΩΓΗ

Το θέμα της παρούσας πτυχιακής εργασίας είναι η κατασκευή ενός συστήματος ένδειξης και καταγραφής της ταχύτητας και της κατεύθυνσης του ανέμου χρησιμοποιώντας αισθητήρες υπερήχων. Το σύστημά μας πρέπει να μετράει συνεχώς την ταχύτητα του ανέμου καθώς επίσης και την κατεύθυνσή του και να εμφανίζει στην οθόνη του υπολογιστή την έντασή του σε:

1) χιλιόμετρα ανά ώρα,

2) μίλια ανά ώρα,

3) κόμβους και

4) Μποφόρ.

Η κατεύθυνσή του θα εμφανίζεται γραφικά με μία εφαρμογή σε Flash η οποία θα είναι μία πυξίδα που θα δείχνει προς τα πού φυσάει ο άνεμος. Είναι μία αρκετά ενδιαφέρουσα κατασκευή η οποία μορεί να χρησιμοποιηθεί στα λιμάνια και στις γέφυρες των πλοίων για να γνωρίζουν τα πλοία πόσα Μποφόρ είναι ο άνεμος, καθώς επίσης και στα αεροδρόμια για την κατάσταση του ανέμου στους αεροδιαδρόμους.

1. Έρευνα αγοράς για τον αισθητήρα υπερήχων

Για να υλοποιηθεί αυτή η κατασκευή χρειάζονται φυσικά αισθητήρες υπερήχων. Στην αγορά υπάρχουν έτοιμα συστήματα τα οποία είναι ικανά να μετρούν με μεγάλη ακρίβεια τα ζητούμενα μεγέθη. Για τις ανάγκες της εργασίας θα χρησιμοποιήσουμε μία μετρητική διάταξη που θα μπορεί να μας παρέχει σειριακή έξοδο RS-232 για να μπορούμε να κάνουμε εύκολα τη σύνδεση με τον υπολογιστή.

Παρακάτω διαθέτουμε τα φύλλα δεδομένων μερικών από τους αισθητήρες που βρήκαμε στα μαγαζιά και στο Internet.

Το μοντέλο 200-7000 WindSonic Ultrasonic Anemometer της εταιρίας Novalynx παρέχει έξοδο RS-232 με Baudrate 9600. Μετράει ταχύτητα ανέμου μέχρι 134 μίλια την ώρα, κατεύθυνση ανέμου 0° - 360° , διαθέτει προστασία από την υγρασία του περιβάλλοντος και λειτουργεί σε θερμοκρασία από -31°C έως +70°C. Με αυτά τα χαρακτηριστικά που διαθέτει είναι μία πολύ καλή επιλογή για την κατασκευή μας.

200-7000 WindSonic Ultrasonic Anemometer

A lightweight unit, the **Model 200-7000 WindSonic Ultrasonic Anemometer** is of a robust, high strength construction. Without the need for expensive on-site calibration or maintenance, and with a corrosion free exterior, WindSonic is a true fit and forget unit.

The flexible design enables you easily to configure WindSonic to deliver the information you require. By using the software provided it is possible to select the output rate and choose the units of measurement that suit your application. Ensuring accuracy and reliability, WindSonic automatically transmits an anemometer status code with each output to indicate its operating status. Available in three options, providing a number of different digital and analog outputs, WindSonic is supplied with NMEA and RS232 digital output as standard. Maintenance free, quick and easy to install, WindSonic is designed to be mounted using a standard pole fitting and comes complete with all screw fittings, a mating marine grade connector, and comprehensive user manual.

Ordering Information

200-7000 WindSonic Ultrasonic Anemometer

Outputs

Output 1, 2, or 4 outputs per second Parameters Wind Speed and Direction or UV Units of measure m/s, knots, mph, kph, ft/min

Wind Speed

Range 0-134 mph (0-60 m/s)

Accuracy ± 4%

Resolution 0.02 mph (0.01 m/s)

Wind Direction

Range 0 to 360° – no dead bandAccuracy ± 3°Resolution 1°

Anemometer Status

Message supplied as part of standard output

Power Requirement

Anemometer 9-30 Vdc, 14.5 mA for option 1

Outputs

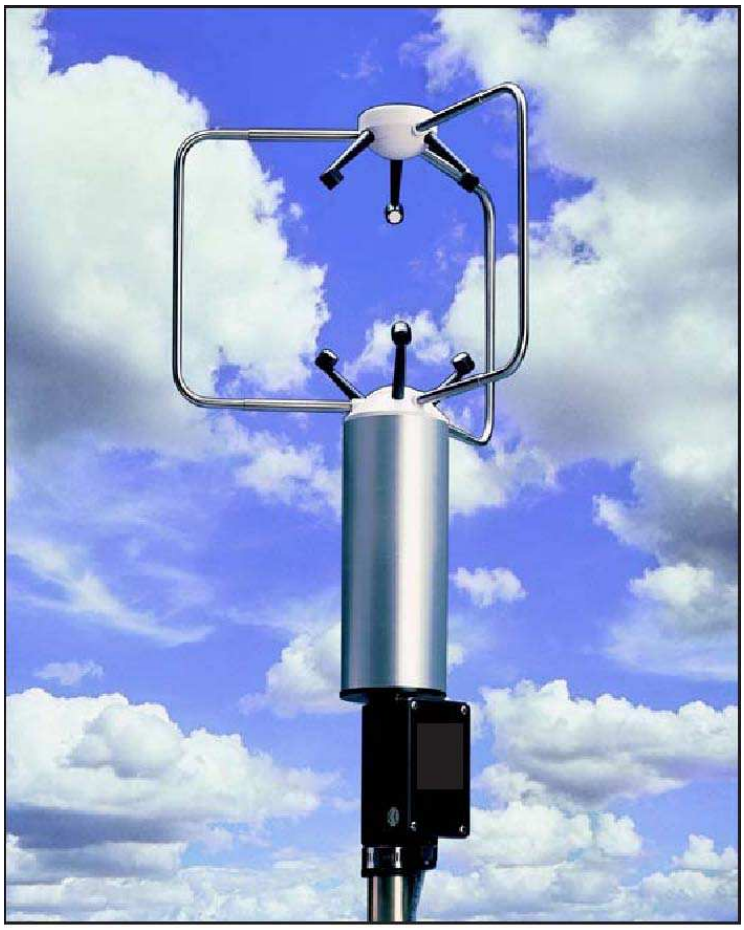
Option: 1 RS232 (9600 baud)

Option 2: RS232 + RS422 + RS485 (9600 baud)

Option 3: RS232 + RS422 + RS485 + 0-5V or 4-

NOVALYNX

Ένα άλλο μοντέλο της ίδιας εταιρίας είναι το **200-81000 Ultrasonic Anemometer.** Διαθέτει και αυτό σειριακή έξοδο, μέγιστη ταχύτητα ανέμου **90** μίλια την ώρα, κατεύθυνση ανέμου 0° - 360° και θερμοκρασία λειτουργίας από -50°C έως +50°C.



200-81000 Ultrasonic Anemometer

The **Model 200-8100 Ultrasonic Anemometer** is a three-dimensional no-moving-parts wind sensor. Two-dimensional anemometers meet the need for economy, but they ignore the important vertical wind component. The 200-81000 measure three dimensional wind velocity based on the transit time of ultrasonic acoustic signals. From speed of sound, sonic temperature is derived. Speed of sound and sonic temperature are corrected for crosswind effects.

Measurement data are available as voltage output signals or serial output using RS232 or RS485 connections. Both voltage and serial output may be configured for various output formats. Operating parameters may be edited using ordinary terminal software on a PC. Simple menus make it easy. All parameters are stored in nonvolatile memory.

The sensor features robust construction with three opposing pairs of ultrasonic transducers supported by stainless steel members. This arrangement provides rigidity to the anemometer while offering a measure of protection to the transducers. The transducers are arranged so that measurement are made through a common volume, thereby improving the validity of the measurement. A fast 160 Hz sampling rate ensures superior measurement resolution.

Superior environmental resistance is achieved by using UV stabilized thermoplastic, stainless steel, and anodized aluminum components. Electrical connections are made via an easily accessible junction box. The unit mounts on standard 1" pipe.

Specifications

Operating Temperature

-50 to +50 °C

Wind speed: 0 to 40 m/s (0 to 90 mph) Resolution: 0.01 m/s Threshold: 0.01 m/s Accuracy: ± 1% rms ± 0.05 m/s (0 to 30 m/s) ± 3% rms (30 to 40 m/s) Wind direction: 0 to 360 degrees Elevation range: ± 60 degrees Resolution: 0.1 degree Accuracy: ± 2 degrees (1 to 30 m/s) ± 5 degrees (30 to 40 m/s) Speed of sound: 300 to 360 m/s Resolution: 0.01 m/s Accuracy: ± 0.1% rms ± 0.05 m/s (0 to 30 m/s) Sonic temperature: -50 to +50 ºC Resolution: 0.01 ºC Accuracy: ± 2 ºC (0 to 30 m/s)

Το μοντέλο ***200-85000 Ultrasonic Anemometer*** της ίδιας εταιρίας μετράει ταχύτητα ανέμου έως 156 mph με ακρίβεια 2% και ανάλυση 0.1 m/s, κατεύθυνση ανέμου 0° - 360° με ακρίβεια 2° και ανάλυση 1°.

NOVALYNX

Wind

200-85000 Ultrasonic Anemometer

The **Model 200-85000 Ultrasonic Anemometer** is a 2-axis, no-moving-parts wind sensor. It is ideal for general meteorological applications requiring accurate, reliable wind measurement.

The sensor features durable, corrosion-resistant construction with opposing pairs of ultrasonic transducers secured in a streamlined molded frame. The 85000 is fully wind-tunnel tested and calibrated to provide accurate wind measurement over a wide operating range.

The standard sensor includes many useful output options. Analog voltage outputs are provided for wind speed and wind direction. A variety of serial output formats are also available on the standard sensor. These include ASCII text, RMYT (compatible with Wind Tracker display), and NMEA formats. Operating parameters may be edited using ordinary terminal software on a PC. Simple menus make it easy. All parameters are stored in nonvolatile memory.

Superior environmental resistance is achieved by using UV stabilized thermoplastic, stainless steel, and anodized aluminum components. The sensor installs on readily available 1" IPS pipe (1.34" o.d.). Wiring connections are made in a convenient weatherproof junction box with screw terminals; special mounting adapters, connectors, and cables are not required.

200-52000 Ultrasonic Anemometer

Specifications

Wind Speed: 0 to 70 m/s (0 to 156 mph)
Resolution: 0.1 m/s
Accuracy: (30 m/s) ± 2% or 0.1 m/s
(70 m/s) ± 3%

Wind Direction: 0 to 360 degrees
Resolution: 1 degree
Accuracy: ± 2 degrees

Serial Output: RS-232 or RS-485

Formats: ASCII Text, RMYT, NMEA

Units: m/s, mph, knots, km/hr

Analog Voltage Outputs:
Wind Speed: 0 to 5 V
Wind Direction: 0 to 5 V

Power Requirement: 9 to 16 VDC, 30 mA typical

Operating Temperature: -50 to +50 °C

Dimensions:
34cm high x 17cm wide
Weight: 0.7 kg (1.5 lb)
Shipping Weight: 1.6 kg (3.5 lb)

Ordering Information

200-85000	Ultrasonic Anemometer
200-85004	Ultrasonic Anemometer, unheated
200-85052	Bird Wire Assembly
330-0524	Cable, per foot

NovaLynx Corporation PO Box 240 Grass Valley CA 95945
Phone: (530) 823-7185 Fax: (530) 823-8997 USA Toll Free: 1-800-321-3577

Το **WS425** της **Vaisala** είναι και αυτό μία πολύ καλή λύση όπως και το **WMT50** της ίδιας εταιρίας. Τα datasheets και των 2 παρατίθενται πιο κάτω.

WS425 **WIND**

Technical Data

Wind speed

Measurement range	
serial output	0...65 m/s (0...144 mph, 0...125 knots)
analog output	0...56 m/s (0...124 mph, 0...107 knots)
Starting threshold	virtually zero
Delay distance	virtually zero
Resolution	0.1 m/s (0.1 mph, 0.1 knots, 0.1 km/h)
Accuracy (range 0...65 m/s)	± 0.135 m/s (±0.3 mph, ±0.26 knots) or 3% of reading, whichever is greater

Wind direction

Measurement range	0...360°
Starting threshold	virtually zero
Delay distance	virtually zero
Resolution	1°
Accuracy (wind speed over 1 m/s)	±2°

Outputs

Digital outputs	
type	RS232, RS422 or RS485, four different message formats
bit rate	adjustable from 1200 to 19200 bit/s
available averages	RS232: 1 to 9 seconds
SDI-12 Standard Data Interface	
type	3 wires for ground, signal and supply
bit rate	fixed 1200 bit/s
available averages	1 to 3600 seconds
Analog outputs	
wind speed	
frequency	5 Hz/mph
voltage	8.0 mV/mph
output impedance	10 kohm
wind direction	
simulated potentiometer	0...V_{ref} represents 0...359°
reference voltage	1.0...4.0 V
output impedance	24 kohm

Response characteristics

maximum reading rate	1 per second
sonic measurement time	0.2 s
signal processing time	0.15 s
response time	0.35 s

General

Operating power supply	10...15 VDC, 12 mA typical (analog)
and for heated model	36 VDC ±10 %, 0.7 A
Operating temperature	
WS425 non-heated	−40...+55 °C (-40...+131 °F)
WS425 heated	−55...+55 °C (-67...+131 °F)
Material	
body	stainless steel (AISI 316)
sensor arms	stainless steel (AISI 316)
transducer heads	silicone rubber
Dimensions	
height	355 mm (14")
width	250 mm (10")
depth	286 mm (12")
Weight	1.7 kg (3.7 lbs)

Complies with EMC standard EN61326-1:1997 + Am1:1998 + Am2:2001; Generic Environment

Accessories

Cable supporting analog outputs, 10 m	ZZ45204
Cable supporting RS-232 outputs, 10 m	ZZ45203
Cable supporting RS-485/422 outputs, 10 m	010411
Cable supporting SDI-12 outputs, 10 m	WS425CABSDI
Adapter for 30-35 mm (1 1/4") diameter vertical tube	WS425FIX30
Adapter for 60 mm (2 1/4") diameter vertical tube	WS425FIX60
Field verifier	WS425VERIFIER

WMT50 Ultrasonic Wind

Sensor

Technical Data, Dimensions

Wind

Wind speed	
Range	0 ... 60 m/s
Response time	0.25 s
Available variables	average, minimum, maximum
Accuracy	
0 ... 35 m/s	±0.3 m/s or ±3 % whichever is greater
35 m/s ... 60 m/s	±5 %
Starting threshold	virtually zero
Output resolution	0.1 m/s (km/h, mph, knots)
Units available	m/s, km/h, mph, knots
Wind direction	
Azimuth	0 ... 360°
Response time	0.25 s
Available variables	average, maximum and minimum
Accuracy	±3°
Starting threshold	virtually zero
Output resolution	1°
Measurement frame	
Averaging time	1 ... 3600 s (=60 min), at one second steps on the basis of samples taken at 4 Hz rate (configurable)
Update interval	1 ... 3600 s (=60 min), at one-second steps

General

Self-diagnostics	separate supervision message, unit/status fields to validate measurement quality
Start-up	automatic, <10 seconds from power on to the first valid output
Serial data interfaces	SDI-12, RS-232, RS-485, RS-422
Communication protocols	SDI-12 v1.3, ASCII automatic & polled, NMEA 0183 v. 3.0 with query option
Port	
Baud rate	1200 ... 115 200
Operating temperature	-52 ... +60 °C (-60 ... +140 °F)
Storage temperature	-60 ... +70 °C (-76 ... +158 °F)
Dimensions	
height	139 mm (5.47")
diameter	127 mm (5.00")
weight	510 g (1.12 lbs)
Housing	IP65

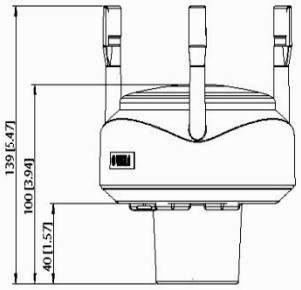
Power supply

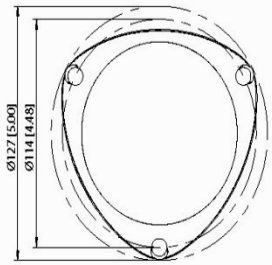
Input voltage	5.3 ... 30 VDC
Power consumption on average	
minimum	0.07 mA at 12 VDC
maximum	13 mA at 30VDC
typical	3 mA at 12 VDC (default measuring intervals)
Heating voltage options	DC, AC, full wave rectified AC

Electromagnetic compatibility

Complies with EMC standard: EN61326-1; 1997 + Am1: 1998 +AM2:2001 Generic Environment

Dimensions





VAISALA
Reliable.

Vaisala Oyj
PO Box 26
FIN-00421 Helsinki,
Finland
Tel: +358 9 894 91
Fax: +358 9 8949 2485

CE

Ref. B210553EN-A 2006-01

- **Ο αισθητήρας της επιλογής μας**

Τελικά χρησιμοποιήσαμε το μοντέλο **WindSonic** του οίκου **Gill** Αγγλίας. Την προσφορά μας την έκανε η εταιρία **ScientAct A.E**. (Κανάρη 16, Τ.Κ. 54644, Θεσσαλονίκη, Τηλ: 2310 946.126):

Μοντέλο Type 2 1405-PK-038

Πρόκειται για αισθητήρα, ο οποίος αντικαθιστά κλασσικά κυπελλοφόρα, ή με προπέλα ανεμόμετρα, καθώς και τους κλασσικούς ανεμοδείκτες. Ο αισθητήρας βασίζεται στην τεχνική των υπερήχων παρέχοντας χαμηλό κόστος αγοράς και μεγάλη ακρίβεια.
Είναι κατάλληλος για χρήση σε οποιοδήποτε περιβάλλον.
Είναι μικρού βάρους, κατασκευασμένος από υλικά με εξαιρετική αντοχή στη διάβρωση.
Δεν έχει κανένα κινητό μέρος
Δεν απαιτεί συντήρηση
Δεν απαιτεί περιοδική βαθμονόμηση
Είναι εξαιρετικά απλός στην εγκατάστασή του και μπορεί να συνδεθεί είτε με Data Logger, είτε απευθείας σε PC, ή άλλες διατάξεις.
Διαθέτει προγραμματιζόμενο ρυθμό εξόδου 1,2, ή 4 μετρήσεις ανά δευτερόλεπτο.
Παρέχει μετρήσεις ταχύτητας / διεύθυνσης ανέμου, ή UV, ή μετρήσεις tunnel.
Μετρά στις παρακάτω μονάδες

- Μέτρα ανά δευτερόλεπτο
- Κόμβους
- Μίλια ανά ώρα
- Χιλιόμετρα ανά ώρα
- Πόδια ανά λεπτό

Περιοχή μέτρησης της ταχύτητας 0 – 60 m/s, με ακρίβεια 2% και ανάλυση 0.01 m/s.

Το Datasheet του αισθητήρα αυτού είναι το παρακάτω

WINDSONIC - ULTRASONIC WIND SENSOR

At last, a real low cost alternative to conventional cup/vane/propeller wind sensors in a single unit - WindSonic from Gill Instruments. Utilising our expertise as the world's leading sonic manufacturer, WindSonic is based on our existing, highly successful, proven ultrasonic technology. Ideal for applications that demand economic wind sensing, WindSonic is suitable for land-based and marine environments.
A lightweight unit, WindSonic is of a robust, high strength construction designed to withstand installation and use with no fear of the damage commonly experienced with more fragile cups, vanes or propellers. Without the need for expensive on-site calibration or maintenance and with a corrosion free exterior, WindSonic is a true fit and forget unit.

The flexible design enables you to easily configure WindSonic to deliver the information you require. By using the software provided it is possible to select the output rate and choose the units of measurement that suit your application. Ensuring accuracy and reliability, WindSonic automatically transmits an anemometer status code with each output to indicate its operating status. Available in four options, providing a number of different digital and analogue outputs.
Maintenance free, quick and easy to install, WindSonic is designed to be mounted using a standard pole fitting and comes complete with all screw fittings, a mating marine grade connector and comprehensive user manual.

The unit is supplied with a 2 year warranty as standard.

CUSTOMER SELECTABLE	
Output	1, 2 or 4 outputs per second
Parameters	Wind Speed & Direction or U and V (vectors)
Units of Measure	m/s, knots, mph, kph, ft/min
WIND SPEED	
Range	0 – 60 m/s (116 knots)
Accuracy	+/- 2%
Resolution	0.01 m/s (0.02 knots)
WIND DIRECTION	
Range	0 to 359° – no dead band
Accuracy	+/- 3°
Resolution	1°
ANEMOMETER STATUS	
Message supplied as part of standard output	
POWER REQUIREMENT	
Anemometer	9-30Vdc @ 14.5mA typical Start up time <1 second
OUTPUTS	
Option 1	RS232
Option 2	RS232 + RS422 + RS485 + NMEA*
Option 3	RS232 + RS422 + RS485 + NMEA* + 0-5V or 4-20mA
Option 4	SDI-12

* NMEA 0183 Version 3

ENVIRONMENTAL	
Ingress Protection	IP65
Operating Temperature	-35°C to +70°C
Storage Temperature	-40°C to +90°C
Operating Humidity	<5% to 100%
EMC	EN 61000-6-2 : 2001 EN 61000-6-3 : 2001
MTBF	
	15 years
MATERIALS	
External Construction	LURAN S KR 2861/1C ASA/PC
DIMENSIONS	
Size	142 x 160 mm
Weight	0.45 kg
WARRANTY	
	2 years
OPTIONAL FACTORY CALIBRATION	
	Traceable to national standards
ACCESSORIES	
Pipe Mounting	44.45 mm (1.75 in) diameter
WindCom - Display & logging software *	
Cables	
Display	

* download WindCom free from www.gill.co.uk

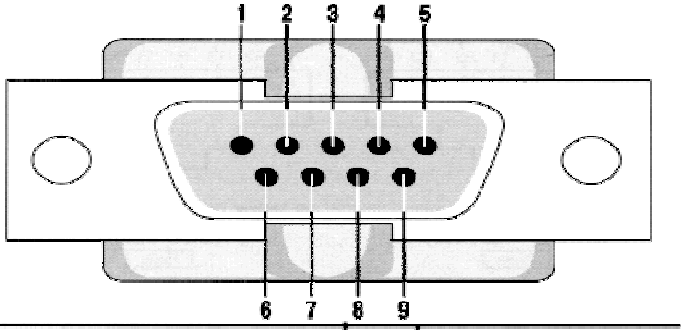
GILL INSTRUMENTS LTD
Saltmarsh Park, 67 Gosport Street,
Lymington, Hampshire, SO41 9EG, UK
Tel: +44 (0) 1590 613500
Fax: +44 (0) 1590 613555
E-mail: anem@gill.co.uk
Website: www.gill.co.uk

The WindSonic is part of the Solent range of ultrasonic anemometers. The range is in continuous development and therefore specifications may be subject to change without prior notice.

D300

2. Θύρα RS-232

Τα δεδομένα από τον αισθητήρα θα τα περάσω στον υπολογιστή μέσω της σειριακής θύρας RS-232. Για να επιτευχθεί αυτή η επικοινωνία πρέπει και ο πομπός (δηλαδή η μετρητική διάταξη που έχω) και ο δέκτης (Η/Υ) να συμφωνούν στο πρωτόκολλο επικοινωνίας καθώς και στα χαρακτηριστικά αυτού. Το πρωτόκολλο που θα χρησιμοποιηθεί είναι το **RS-232**.



Pin	Signal	Pin	Signal
1	Data Carrier Detect	6	Data Set Ready
2	Received Data	7	Request to Send
3	Transmitted Data	8	Clear to Send
4	Data Terminal Ready	9	Ring Indicator
5	Signal Ground		

Χαρακτηριστικά αυτού είναι:

1) η ταχύτητα με την οποία οι συσκευές θα ανταλλάσουν δεδομένα, το λεγόμενο Baudrate.

2) Το μήκος της λέξης σε bit. Μπορεί να είναι 5,6,7,8, ή 9 bits. Πιο συχνά όμως συναντάμε τα 8 bits σαν μήκος λέξης**.**

3) Το bit ισοτιμίας (Parity bit). Αυτό είναι ένας τρόπος να ελέγξω αν η πληροφορία που πήρε ο παραλήπτης είναι η σωστή ή όχι. Βέβαια είναι πολύ απλός τρόπος και υπάρχει μεγάλη πιθανότητα να μην εντοπιστεί το σφάλμα γι αυτό και πολλές φορές δε χρησιμοποιείται. Σε περίπτωση που χρησιμοποιηθεί πρέπει να ξέρουμε ότι η ισοτιμία μπορεί να είναι μονή ή ζυγή. Τι σημαίνει αυτό: Σε κάθε λέξη που στέλνεται μετρούνται πόσα από τα bits είναι 1. Έστω ότι έχω μονή ισοτιμία. Αν η λέξη είναι 10001101 το parity bit θα είναι 1 για να έχω στο σύνολο μονό αριθμό άσσων. Αν είχα επιλέξει τη ζυγή ισοτιμία στο παραπάνω παράδειγμα το parity bit θα ήταν 0.

4) Stop bits. Το stop bit στέλνεται στο τέλος κάθε λέξης για να επιτρέψει στον παραλήπτη να συγχρονιστεί ξανά με τον αποστολέα.

5) Το Flow Control (έλεγχος ροής): Η σειριακή θύρα μπορεί να σταματήσει και να ξαναρχίσει την αποστολή δεδομένων με το Flow control. Όταν πχ ο παραλήπτης είναι ένα αργό μηχάνημα δηλαδή δεν μπορεί να δεχτεί όλα τα δεδομένα με το ρυθμό που αποστέλλονται τότε χρησιμοποιούνται 2 χαρακτήρες. XON και XOFF. Τα XON και XOFF στέλνονται από τον παραλήπτη στον αποστολέα. Το XOFF λέει στον αποστολέα να σταματήσει να στέλνει δεδομένα και το XON του λέει να συνεχίσει την αποστολή δεδομένων.

Πιο πριν αναφέραμε για το Parity bit που είναι ένας τρόπος να ελέγξουμε αν η πληροφορία στάλθηκε σωστά ή όχι. Τα δεδομένα που λαμβάνω από τη συσκευή πρέπει να ελέγχονται. Πρέπει να βλέπω αν η μετάδοση των δεδομένων είναι η αναμενόμενη. Δεν μπορεί δηλαδή να έχω ένταση του ανέμου 1000 Μποφόρ. Γι αυτό θα χρησιμοποιήσουμε μία πάρα πολύ καλή τεχνική, την τεχνική του CRC (Cyclic Redundancy Check).

3. Κυκλικός έλεγχος πλεονασμού (CRC)

Μία από τις δημοφιλέστερες μεθόδους ανίχνευσης λάθους για ψηφιακά σήματα είναι ο κυκλικός έλεχος πλεονασμού (CRC). Η βασική ιδέα πίσω από τα CRC είναι να μεταχειριστούμε το μήνυμα ως μία ενιαία δυαδική λέξη **M** και να τη διαιρέσουμε με μία λέξη κλειδί **K** που είναι γνωστή και στον αποστολέα και στον παραλήπτη.Το υπόλοιπο **Y** που έμεινε από τη διαίρεση **M/K** αποτελεί τη **λέξη ελέγχου** για το δεδομένο μήνυμα. Ο αποστολέας στέλνει και το μήνυμα **M** και τη λέξη ελέγχου **Y** και έτσι ο παραλήπτης μπορεί να ελέγξει τα δεδομένα επαναλαμβάνοντας τον υπολογισμό **M/K** και επαληθεύοντας ότι το υπόλοιπο είναι **Y**. Το CRC χρησιμοποιεί μία απλουστευμένη μορφή αριθμητικής την οποία θα εξηγήσουμε πιο κάτω.

Επ’ ευκαιρία, αυτή η μέθοδος ελέγχου δεν είναι μη νοθεύσιμη, επειδή υπάρχουν πολλές διαφορετικές σειρές μηνυμάτων τα οποία δίνουν υπόλοιπο **Y** όταν διαιρεθούν δια **K**. Στην πραγματικότητα 1 στις κ τυχαία επιλεγμένες σειρές θα δώσουν συγκεκριμένο υπόλοιπο. Κατά συνέπεια, αν το μήνυμα είναι αλλοιωμένο στη μετάδοση υπάρχει μία πιθανότητα (περίπου 1/κ, που υποθέτει ότι το αλλοιωμένο μήνυμα είναι τυχαίο) ότι η αλλοιωμένη έκδοση θα συμφωνούσε με τη λέξη ελέγχου. Σε μια τέτοια περίπτωση το λάθος δε θα ανιχνευόταν. Εντούτοις, κάνοντας το **K** αρκετά μεγάλο, οι πιθανότητες ενός τυχαίου λάθους να μην ανιχνευθεί είναι εξαιρετικά μικρές.

Το υπόλοιπο αυτής της συζήτησης θα αποτελέσει η βασική ιδέα της βελτιστοποίησης της αποτελεσματικότητας του CRC, περιγράφοντας την απλουστευμένη αριθμητική που χρησιμοποιείται για τη βελτίωση των υπολογισμών έτσι ώστε να επιτευχθεί η μέγιστη αποδοτικότητα κατά την επεξεργασία δυαδικών σειρών.

Κατά τη συζήτηση αυτή είναι σύνηθες να παρουσιάζεται η λέξη κλειδί **K** υπό τη μορφή πολυωνύμου του οποίου οι συντελεστές είναι τα δυαδικά bits

του αριθμού **K**. Παραδείγματος χάριν, ας υποθέσουμε ότι το CRC θέλουμε να χρησιμοποιήσει το κλειδί **K=37**. Αυτός ο αριθμός αν γραφεί δυαδικά είναι ο αριθμός **100101**, και εκφρασμένος ως πολυώνυμο είναι το **x^5 + x^2 + 1**. Για να εφαρμοστεί το CRC σε αυτό το πολυώνυμο πρέπει και ο παραλήπτης και ο αποστολέας να έχουν συμφωνήσει ότι αυτό είναι η λέξη κλειδί που σκοπεύουν να χρησιμοποιήσουν. Ας πούμε λοιπόν ότι έχουμε συμφωνήσει ότι θα χρησιμοποιήσουμε το πολυώνυμο που προκύπτει από το **100101**.

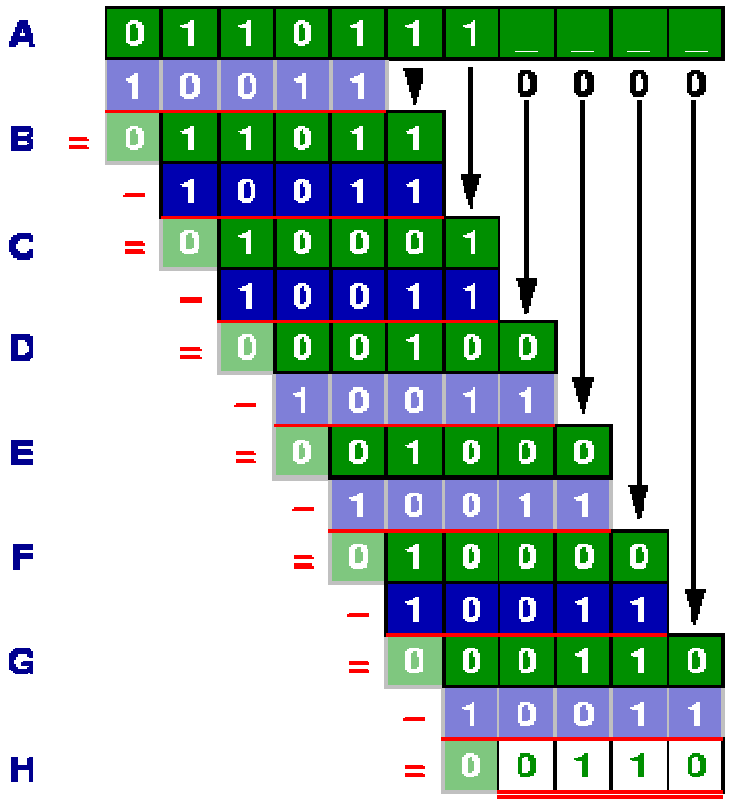
Αξίζει να σημειώσουμε ότι το υπόλοιπο οποιασδήποτε λέξης που θα διαιρεθεί με μία 6-bit λέξη, θα περιέχει το πολύ 5 bits, έτσι οι λέξεις που θα βασιστούν στο πολυώνυμο 100101 θα είναι το πολύ 5 bits. Επομένως, ένα σύστημα CRC που θα βασίζεται σε αυτό το πολυώνυμο θα λέγεται “5-bit CRC”. Γενικά, ένα πολυώνυμο με k bits οδηγεί σε k-1 bit CRC.

Τώρα ας υποθέσουμε ότι θέλουμε να στείλουμε ένα μήνυμα το οποίο αποτελείται από την παρακάτω σειρά bits: **M = 00101100010101110100011** και επίσης να στείλουμε και πρόσθετες πληροφορίες οι οποίες θα μας βοηθήσουν να ελέγξουμε την ακρίβεια του μηνύματος που στάλθηκε. Χρησιμοποιώντας τη συμφωνηθείσα λέξη κλειδί **K=100101** απλά θα διαιρέσω το **M** δια **K** για να σχηματίσω το υπόλοιπο **Y** το οποίο θα αποτελεί τη λέξη ελέγχου CRC. Θα χρησιμοποιήσουμε ένα απλουστευμένο είδος διαίρεσης το οποίο ταιριάζει με τη δυαδική μορφή στην οποία είναι εκφρασμένα και τα δεδομένα μας.

Αν ερμηνεύσουμε το **K** ως ένα συνηθισμένο ακέραιο αριθμό (37), η δυαδική του αναπαράσταση, 100101, στην πραγματικότητα είναι η συντομογραφία για το **(1)2^5 + (0)2^4 + (0)2^3 + (1)2^2 + (0)2^1 + (1)2^0**.

Κάθε ακέραιος αριθμός μπορεί να εκφραστεί μοναδικά με αυτόν τον τρόπο, δηλαδή σαν πολυώνυμο με βάση το 2 και συντελεστές που είναι είτε 0 είτε 1. Αυτή είναι μία πολύ ισχυρή μορφή αναπαράστασης, αλλά στην

πραγματικότητα είναι πιο ισχυρή από αυτή που χρειαζόμαστε για να κάνουμε τον έλεγχο των δεδομένων.



Για να κάνουμε τα πράγματα πιο απλά ας ερμηνεύσουμε το μήνυμα **M**, τη λέξη κλειδί **K** και το υπόλοιπο **Y**, όχι σαν ακέραιους πραγματικούς αριθμούς, αλλά σαν πολυώνυμα με μία μεταβλητή χ (παρά μία καθορισμένη βάση όπως το 2 για τους δυαδικούς ή το 10 για τους δεκαδικούς αριθμούς).

Επίσης θα απλοποιήσουμε κι άλλο με την προϋπόθεση ότι θα δώσουμε προσοχή στην ισότητα των συντελεστών. Δηλαδή αν ένας συντελεστής είναι περιττός αριθμός θα τον θεωρήσουμε απλά σαν 1,και αν είναι ζυγός αριθμός θα τον θεωρήσουμε σαν 0.

Για να δώσουμε μία συνοπτική απεικόνιση, ας θεωρήσουμε τα πολυώνυμα **x^2 + x + 1** και **x^3 + x + 1**. Αν πολλαπλασιάσουμε αυτά τα 2 πολυώνυμα θα πάρουμε το εξής αποτέλεσμα:

(x^2 + x + 1)(x^3 + x + 1) = x^5 + x^4 + 2x^3 + 2x^2 + 2x + 1

και σύμφωνα με την απλοποίησή μας θα κάνουμε κάθε ζυγό συντελεστή 0 και κάθε περιττό συντελεστή 1 και έτσι το αποτέλεσμα του πολλαπλασιασμού θα είναι απλά **x^5 + x^4 + 1**. Θα αναρωτιέστε εαν αυτός ο απλουστευμένος τρόπος είναι αξιόπιστος. Για παράδειγμα, μπορούμε να διαιρέσουμε το αποτέλεσμα του πολλαπλασιασμού **x^5 + x^4 + 1** με έναν από τους παράγοντές του, ας πούμε το **x^2 + x + 1**, και να μας δώσει τον άλλο παράγοντα; Η απάντηση είναι ναι και είναι πιο εύκολο και πιο απλό απ' ότι η κοινή διαίρεση. Για να διαιρέσουμε το πολυώνυμο **110001** δια **111** (που είναι ο τρόπος συντομογραφίας για να εκφράσουμε τα πολυώνυμά μας) απλά εφαρμόζουμε την πράξη Exclusive-OR (XOR) επανειλημμένα ως εξής:

```
  1011
      ______
111 |110001
     111
     ---
     0010
      000
      ---
      0100
       111
       ----
       0111
        111
        ---
        000
```

Αυτό είναι ακριβώς όπως η συνηθισμένη διαίρεση, αλλά πιο απλή, επειδή σε κάθε στάδιο χρειάζεται μόνο να ελέγξουμε εαν το πρώτο bit από τα τρία του

διαιρεταίου είναι 0 ή 1. Αν είναι 0, βάζουμε 0 στο πηλίκο και κάνουμε XOR στα επόμενα 3 bits με το 000. Αν είναι 1, βάζουμε 1 στο πηλίκο και κάνουμε XOR στα επόμενα 3 bits με το 111. Όπως φαίνεται το αποτέλεσμα της διαίρεσης

110001 δια 111 είναι το 1011 το οποίο είναι ο άλλος μας παράγοντας, **x^3 + x + 1**, αφήνοντας υπόλοιπο 000.

Άρα τώρα έχουμε ό,τι χρειαζόμαστε για να υπολογίσουμε το CRC του μηνύματος **M** και της λέξης κλειδί **K** που καθορίσαμε πιο πάνω. Απλά πρέπει να κάνουμε τη διαίρεση χρησιμοποιώντας το απλουστευμένο μας πολυώνυμο. Στην πραγματικότητα είναι ακόμα πιο απλό επειδή δε χρειάζεται να παρακολουθούμε το πηλίκο – αυτό που πραγματικά χρειαζόμαστε είναι το υπόλοιπο. Πρέπει λοιπόν να κάνουμε μία σειρά από 6-bit XOR με τη λέξη κλειδί **K** ξεκινώντας από τον αριστερότερο άσσο (1) του μηνύματος και έκτοτε κατεβάζοντας όσα bits χρειάζονται από το μήνυμα έτσι ώστε να δημιουργηθεί μία 6-bit λέξη που ξεκινάει με 1. Ο υπολογισμός αυτός παρουσιάζεται πιο κάτω:

```
100101 |00101100010101110100011
          100101
          ------
          00100101
            100101
            ------
            0000000101110
                   100101
                   ------
                   00101110
                     100101
                     ------
                     00101100
                       100101
                       ------
                       00100111
```

```
100101
------
000010    Υπόλοιπο = CRC
```

Η λέξη CRC είναι απλά το υπόλοιπο δηλ. το αποτέλεσμα του 6-bit XOR. Το πρώτο bit αυτής της πράξης είναι πάντα 0,έτσι αυτό που χρειαζόμαστε είναι τα τελευταία 5 bits. Και αυτό συμβαίνει γιατί ένα 6-bit κλειδί μας οδηγεί σε 5-bit CRC. Σε αυτήν την περίπτωση, η CRC λέξη μου είναι το **00010**. Έτσι, όταν στείλω το μήνυμα **M**, θα στείλω και την αντίστοιχη λέξη CRC. Ο παραλήπτης μπορεί να επαναλάβει τον παραπάνω υπολογισμό στο **M** και να ελέγξει ότι το υπόλοιπο που θα βρει συμφωνεί με τη λέξη CRC που περιελαμβανόταν στη μετάδοση.

Αυτό που μόλις κάναμε ήταν ένας τέλειος CRC υπολογισμός, και πράγματι πολλές εφαρμογές λειτουργούν με αυτόν τον τρόπο, αλλά υπάρχει και ένα μειονέκτημα στη μέθοδό μας. Όπως μπορούμε να δούμε, ο υπολογισμός που μόλις περιγράψαμε αγνοεί τα 0 που βρίσκονται μπροστά από τον πρώτο άσσο του μηνύματος. Πολλές φορές συμβαίνει στις πραγματικές εφαρμογές να ξεκινάει το μήνυμα με μία μεγάλη σειρά από 0 και έτσι ο αλγόριθμος δε θα λειτουργεί καλά σε αυτές τις περιπτώσεις. Για να αποφύγουμε αυτό το πρόβλημα μπορούμε να συμφωνήσουμε εκ των προτέρων ότι πριν υπολογίσουμε το n-bit CRC, θα ξεκινάμε πάντα κάνοντας XOR στα πρώτα n bits του μηνύματος με μία σειρά από n άσσους. Αυτή τη σύμβαση πρέπει να την έχουν συμφωνήσει και ο παραλήπτης και ο αποστολέας. Με αυτή τη σύμβαση το παράδειγμά μας γίνεται έτσι:

```
00101100010101110100011   <--  Original message string
11111                      <-- "Fix" the leading bits
----------------------
11010100010101110100011    <-- "Fixed" message string
100101
------
0100000
 100101
 ------
 000101001
    100101
    ------
    00110001
```

```
100101
------
0101000
 100101


 ------
 00110111
   100101
   ------
   0100101
    100101
    ------
    0000000100011
           100101
           ------
           000110    Υπόλοιπο = CRC
```

Με τη σύμβαση αυτή, η 5-bit CRC λέξη γι αυτό το μήνυμα βασισμένη στο πολυώνυμο **100101** είναι το **00110**. Έτσι γίνεται ο υπολογισμός του CRC και πολλές εμπορικές εφαρμογές λειτουργούν με τον τρόπο που περιγράψαμε. Μερικοί άνθρωποι χρησιμοποιούν διάφορες ρουτίνες για να επιταχύνουν τις διαιρέσεις, αλλά αυτό δεν αλλάζει τον υπολογισμό ή το αποτέλεσμα. Άλλες φορές συμφωνούν σε άλλες συμβάσεις όπως η ερμηνεία των bits σε αντίστροφη σειρά, αλλά ο ουσιαστικός υπολογισμός παραμένει ο ίδιος. Φυσικά το κρισιμότερο σημείο είναι να συμφωνήσουν και ο αποστολέας και ο παραλήπτης στη σύμβαση που θα χρησιμοποιηθεί.

Τώρα που έχουμε δει πώς υπολογίζεται το CRC για ένα δεδομένο πολυώνυμο, είναι φυσικό να αναρωτηθούμε αν μερικά πολυώνυμα δίνουν πιο «γερούς» ελέγχους από κάποια άλλα. Η απάντηση προφανώς είναι ναι διότι όσο πιο μεγάλη είναι η λέξη κλειδί, τόσο λιγότερες πιθανότητες υπάρχουν τα αλλοιωμένα στοιχεία να μην ανιχνευθούν. Χρησιμοποιώντας 32-bit CRC υπάρχουν πιθανότητες λιγότερες από 1 στο δισεκατομμύριο να μην ανιχνευθούν τα λάθος δεδομένα.

Δεδομένου ότι τα περισσότερα ψηφιακά συστήματα σχεδιάζονται για 8-bit λέξεις,(που ονομάζονται bytes), το πιο κοινό είναι να βρεθούν λέξεις κλειδιά που τα μήκη τους είναι πολλαπλάσια του 8. Στην πράξη, τα 2 πιο

κοινά μήκη είναι τα 16-bit και 32-bit CRC (άρα τα αντίστοιχα πολυώνυμα θα έχουν 17 και 33 bits αντίστοιχα). Υπάρχουν συγκεκριμένα πολυώνυμα τα οποία είναι διαδεδομένα στη χρήση. Για 16-bit CRC μία από τις πιο

δημοφιλέστερες λέξεις κλειδιά είναι η **10001000000100001** και για 32-bit CRC είναι η **10000010011000001000111010110111**. Υπό μορφή ρητών πολυωνύμων γράφονται ως

x^16 + x^12 + x^5 + 1

και

x^32 + x^26 + x^23 + x^22 + x^16 + x^12 + x^11+ x^10 + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1

Το 16-bit πολυώνυμο είναι γνωστό ως “X25 standard” και το 32-bit πολυώνυμο ως “Ethernet standard”. Και τα 2 πολυώνυμα χρησιμοποιούνται ευρέως σε όλα τα είδη εφαρμογών. Ένα άλλο κοινό 16-bit κλειδί είναι το **11000000000000101** το οποίο είναι η βάση του πρωτοκόλλου “CRC-16” το οποίο είναι πολύ γνωστό σε χειριστές modem. Αυτά τα πολυώνυμα δεν είναι ο μοναδικός τρόπος που είναι κατάλληλος για τον υπολογισμό του CRC, αλλά είναι μία πολύ καλή ιδέα να χρησιμοποιηθούν τα καθιερωμένα πρώτυπα για να εκμεταλλευτούμε την εμπειρία πολλών ετών χρήσης.

4. Εισαγωγή στο πρόγραμμα

Το πρόγραμμα που πρέπει να υλοποιήσουμε είναι η απεικόνιση πάνω σε μία φόρμα της ταχύτητας του ανέμου και της κατεύθυνσής του. Τα δεδομένα αυτά πρέπει να τα παίρνουμε από τον αισθητήρα. Σε περίπτωση που τα δεδομένα είναι αλλοιωμένα δεν εμφανίζονται. Για να το πετύχουμε αυτό χρησιμοποιήσαμε την τεχνική του CRC στον κώδικά μας. Επίσης για τις ανάγκες της παρουσίασης χρησιμοποιήσαμε και μία extra ιδιότητα στο πρόγραμμα, το Example Mode. Χρησιμοποιώντας το παράγουμε τυχαίες τιμές για να δείξουμε πώς λειτουργεί η εφαρμογή. Για την υλοποίησή της χρησιμοποιήσαμε τη γλώσσα προγραμματισμού Visual Basic λόγω της μεγάλης γκάμας εργαλείων που μας παρέχει αλλά και λόγω δικής μας γνώσης πάνω σε αυτήν. Παρακάτω ξεκινάμε με τη γενική περιγραφή της εφαρμογής και στη συνέχεια θα ακολουθήσει ανάλυση των επιμέρους τμημάτων της.

- **Έναρξη του προγράμματος**

Όταν η εφαρμογή ξεκινάει γίνονται δηλώσεις κάποιων μεταβλητών οι οποίες θα μας βοηθήσουν για να γίνει η απόκτηση των δεδομένων από τον αισθητήρα καθώς και ο τρόπος που αυτά θα εμφανιστούν στην οθόνη του υπολογιστή.

Έπειτα το πρόγραμμα διαβάζει το αρχείο Application.ini μέσω ενός module που δημιουργήσαμε για να έχουμε τη δυνατότητα αποθήκευσης των τιμών κάθε φορά που τερματίζεται η εφαρμογή. Σε αυτό το αρχείο είναι αποθηκευμένες οι τελευταίες τιμές για την ένταση και την κατεύθυνση του ανέμου, καθώς και ο αριθμός της σειριακής θύρας την οποία χρησιμοποιήσαμε για να πάρουμε τα δεδομένα από τον αισθητήρα. Κάθε φορά που τερματίζεται η εφαρμογή αυτές οι τιμές αποθηκεύονται έτσι ώστε όταν γίνει ξανά εκκίνηση του προγράμματος να εμφανιστούν στην οθόνη οι τελευταίες τιμές.

```
[PARAMETRES]
SerialPort1-COM=1
DummyMode=False
[VALUES]
Angle=357,135688304892
Wind=0,584670782089233
```

Για να το πετύχουμε αυτό χρησιμοποιούμε τις παρακάτω εντολές:

CreateINIFile: Με την εντολή αυτή γίνεται η δημιουργία του αρχείου ini. Για το αρχείο Application.ini η εντολή είναι CreateINIFile("Application.ini")

CreateINIApp: Δημιουργία παραμέτρων του αρχείου. Για να δημιουργήσουμε την παράμετρο Values τότε πρέπει να γράψω στον κώδικα CreateINIApp("VALUES")

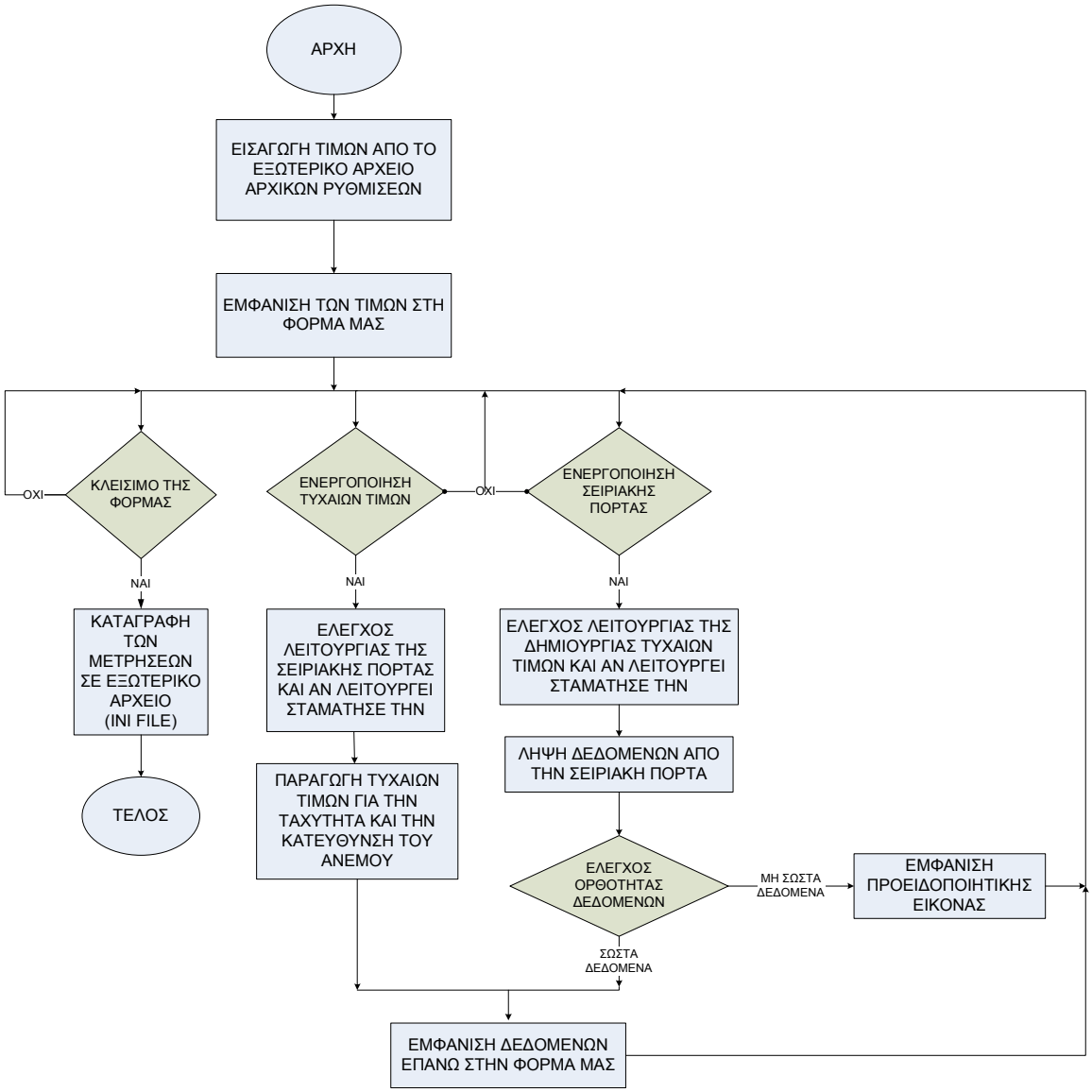
CreateINIKey: Για να φτιάξω ένα κλειδί για κάθε παράμετρο χρησιμοποιώ αυτή την εντολή. Αν θέλω να φτιάξω κλειδί για την παράμετρο Values και θέλω να το ονομάσω Angle θα γράψω CreateINIKey("VALUES", "Angle", "1")

ReadINI: Για να διαβάω την τιμή ενός κλειδιού που βρίσκεται σε κάποια παράμετρο χρησιμοποιώ την εντολή ReadINI. Αν π.χ. θέλω να διαβάσω την τιμή του κλειδιού Angle που βρίσκεται στην παράμετρο Values τότε πρέπει να γράψω ReadINI("VALUES", "Angle")

WriteINI: Αν όμως θέλω να αποθηκεύσω μία τιμή σε ένα κλειδί τότε χρησιμοποιώ αυτή την εντολή π.χ. WriteINI("VALUES", "Wind")

Μετά από την ανάγνωση αυτών των στοιχείων η εφαρμογή προχωράει ανάλογα με τις τιμές που μόλις διάβασε. Δηλ. αν όταν έκλεισε το πρόγραμμα ήταν σε κατάσταση παραδείγματος, τότε θα συνεχίσει να παράγει τυχαίες τιμές για την ένταση και την κατεύθυνση ξεκινώντας από τις τελευταίες αποθηκευμένες τιμές. Αν όμως δε βρισκόταν σε κατάσταση παραδείγματος, τότε θα εμφανίσει στην οθόνη τις τελευταίες τιμές.

Το κεντρικό διάγραμμα ροής του προγράμματός μας παρατίθεται στην επόμενη εικόνα:



Διάγραμμα ροής γενικής φόρμας

Τώρα υπάρχουν στη φόρμα μου 3 κουμπιά τα οποία μπορώ να χρησιμοποιήσω:

Start / Stop Example Mode: Πατώντας το μπορούμε να ξεκινήσουμε την κατάσταση παραδείγματος ή να τη διακόψουμε.

Select Serial Port: Εδώ έχουμε τη δυνατότητα να επιλέξουμε με ποιά σειριακή θύρα του υπολογιστή θα επικοινωνεί ο αισθητήρας μας. Δηλ. με την Com1 ,Com2 κλπ.

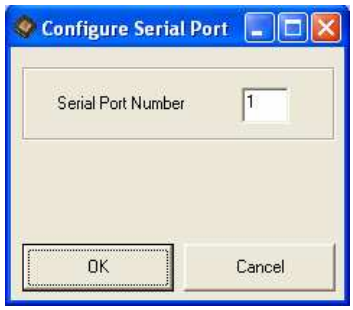
Start / Stop Listening Mode: Όταν πατήσουμε το Start Listening Mode ξεκινάει η επικοινωνία του υπολογιστή με τον αισθητήρα μέσω της σειριακής θύρας RS-232. Έχουμε δηλαδή απόκτηση των τιμών που μετράει ο αισθητήρας. Στη συνέχεια ελέγχονται τα δεδομένα ως προς την ορθότητά τους. Εαν είναι σωστά τότε συνεχίζει το πρόγραμμα και εμφανίζει τις τιμές στη φόρμα. Εαν όμως δεν είναι σωστά εμφανίζει ένα προειδοποιητικό σήμα λάθους. Εαν είναι ενεργοποιημένη η λήψη δεδομένων και πατήσουμε το Stop τότε διακόπτεται η επικοινωνία Αισθητήρα – Υπολογιστή.

Είτε έχουμε κατάσταση παραδείγματος είτε λήψη δεδομένων από τον αισθητήρα οι τιμές της έντασης και της κατεύθυνσης του ανέμου εμφανίζονται πάνω στη φόρμα. Η κατεύθυνση εμφανίζεται σε ένα αντικείμενο flash που είναι μία πυξίδα και η ένταση εμφανίζεται σε χιλιόμετρα, μίλια, κόμβους και στην κλίμακα Μποφορ.

5. Επιλογή σειριακής θύρας

Όπως έχουμε αναφέρει και νωρίτερα, στο πρόγραμμά μας έχουμε τη δυνατότητα να επιλέξουμε τη σειριακή θύρα με την οποία θα επικοινωνεί ο αισθητήρας με τον υπολογιστή. Τώρα θα εξηγήσουμε πώς μπορώ να το κάνω αυτό.

Στην κεντρική φόρμα διακρίνουμε 3 κουμπιά. Πατάμε το μεσαίο κουμπί το οποίο φέρει τον τίτλο Select Serial Port. Στην οθόνη του υπολογιστή εμφανίζεται μία μικρή φόρμα. Δίπλα από το Serial Port Number υπάρχει ένα κουτί μέσα στο οποίο πληκτρολογούμε την επιθυμητή σειριακή. Δηλ. για την Com2 πρέπει να γράψουμε τον αριθμό 2.

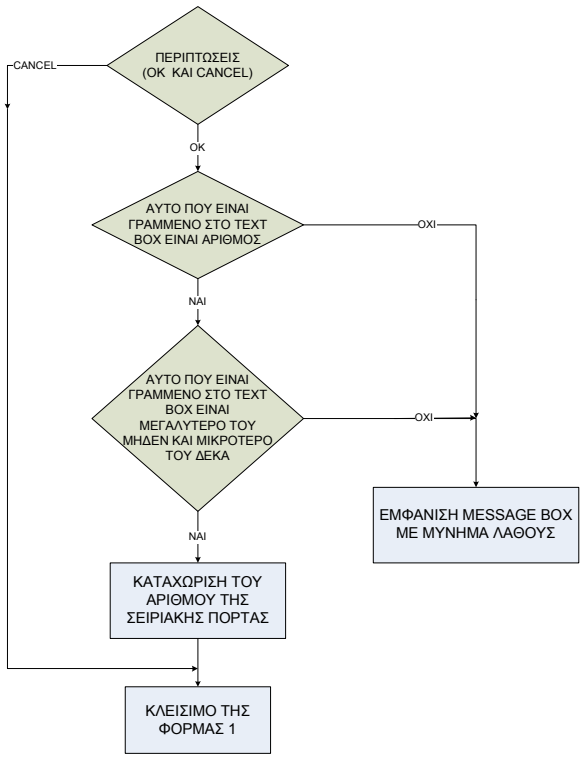


Αν πατηθεί το κουμπί Cancel η φόρμα κλείνει χωρίς να αποθηκευθεί ο αριθμός της σειριακής. Η άλλη περίπτωση είναι να πατηθεί το OK. Με το OK παίρνει την εντολή το πρόγραμμα να ρυθμίσει την επικοινωνία με τη συγκεκριμένη σειριακή. Πρώτα όμως πρέπει να γίνουν οι εξής έλεγχοι έτσι ώστε να εξασφαλίσουμε ότι το project θα δουλέψει σωστά:

1) Ελέγχουμε αν αυτό που πληκτρολογήσαμε είναι αριθμός. Σε περίπτωση που δεν είναι αριθμός, αλλά κάποιος άλλος χαρακτήρας τότε εμφανίζεται στην οθόνη μου ένα μήνυμα λάθους που μου λέει ότι αυτό που έβαλα στη φόρμα δεν είναι έγκυρο. Αν τελικά είναι αριθμός προχωράμε στο δεύτερο και τελευταίο έλεγχο

2) Ελέγχουμε αν αυτό που πληκτρολογήσαμε είναι αριθμός μεταξύ του 1 και του 10. Σε περίπτωση που ισχύει κάτι τέτοιο γίνεται η ρύθμιση επικοινωνίας με τη συγκεκριμένη σειριακή θύρα και η φόρμα κλείνει. Σε αντίθετη περίπτωση εμφανίζεται πάλι ένα μήνυμα λάθους που μας ενημερώνει για τη μη εγκυρότητα της επιλογής μας.

Όλα πλέον είναι έτοιμα. Έχουμε ρυθμίσει τον υπολογιστή να παίρνει δεδομένα από τη Serial Port της επιλογής μας. Το διάγραμμα ροής της λειτουργίας που μόλις περιγράψαμε είναι το παρακάτω:



6. Ενεργοποίηση Example mode

Για να παρουσιάσουμε πώς δουλεύει η εφαρμογή μας έχουμε τοποθετήσει και μία κατάσταση παραδείγματος (Example Mode)

Το Example Mode είναι ένας τρόπος να δούμε πώς δουλεύει το πρόγραμμά μας χωρίς να παίρνουμε δεδομένα από τον αισθητήρα, απλά παράγοντας με μία συνάρτηση τυχαίες τιμές έντασης και κατεύθυνσης.

Όταν πατηθεί το κουμπί Start Example Mode τότε ενεργοποιείται ένας μετρητής (Timer) ο οποίος μας καθορίζει κάθε πότε θα έχουμε παραγωγή τυχαίων τιμών. Αυτός ο μετρητής έχει μία ιδιότητα που ονομάζεται Interval (χρονικό διάστημα) και εκφράζεται σε msec. Η ιδιότητα αυτή στο δικό μας Timer έχει την τιμή 250 που σημαίνει ότι κάθε 250 msec θα καλεί την υπορουτίνα που είναι υπεύθυνη για την παραγωγή τυχαίων τιμών. Έτσι θα βλέπουμε στην οθόνη κάθε 250 msec να αλλάζουν οι τιμές της έντασης και της κατεύθυνσης σαν να παίρνουμε δεδομένα από τον αισθητήρα.

Μετά που θα τελιώσει να μετράει ο μετρητής, ελέγχουμε αν ξεκινάμε ή σταματάμε την κατάσταση παραδείγματος. Αν τη σταματάμε, σταματάει και ο μετρητής να μετράει. Αν όμως θέλουμε να έχουμε κατάσταση παραδείγματος τότε καλείται η υπορουτίνα producedummyvalues (Παραγωγή ψεύτικων τιμών). Για να παράγουμε τιμές με τυχαίο τρόπο έχουμε δημιουργήσει μία συνάρτηση (function) στη Visual Basic την οποία την ονομάσαμε RandomChange(τυχαία αλλαγή). Η συνάρτηση αυτή συντάσσεται ως εξής:

Randomchange(η μεταβλητή που θα αλλάξει, μέγιστη αλλαγή, το ποσοστό αλλαγής)

Για παράδειγμα έστω ότι θέλω να αλλάξω τη μεταβλητή Wind με μέγιστη αλλαγή 20 και ποσοστό αλλαγής 0,55 γράφω το εξής:

RandomChange(Wind,20,0.55). Η συνάρτησή μου θα κάνει τα παρακάτω βήματα:

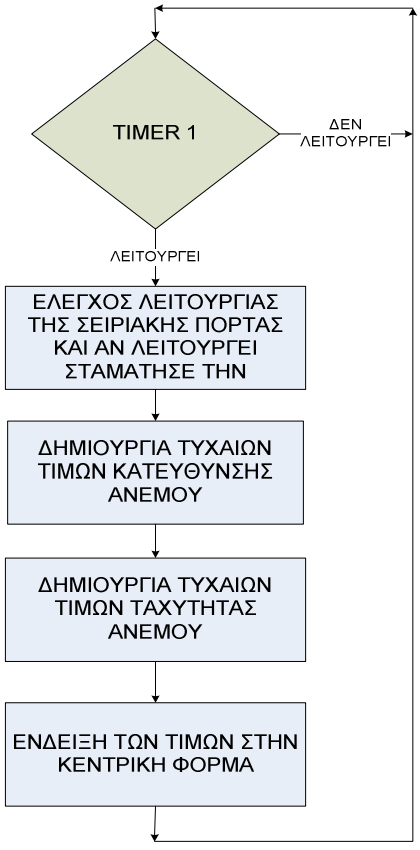
1) Θα χρησιμοποιήσει την εντολή Rnd της Visual Basic η οποία παράγει μία τυχαία τιμή από 0-1

2) Το παραπάνω αποτέλεσμα θα το ελέγξει για 2 περιπτώσεις: αν είναι μεγαλύτερο από το ποσοστό μου δηλ >0,55 ή αν είναι μικρότερο. Στην πρώτη περίπτωση θα αυξήσει την τιμή της μεταβλητής ενώ στη δεύτερη θα τη μειώσει. Έστω ότι είμαστε στην πρώτη περίπτωση με αρχική τιμή Wind = 10 και αποτέλεσμα της Rnd = 0.6.Τότε για να βρω τη νέα τιμή του ανέμου θα πολλαπλασιάσω το αποτέλεσμα της Rnd με τη μέγιστη αλλαγή και αυτό που θα βρω θα το προσθέσω στην ήδη υπάρχουσα τιμή δηλ.

Wind = Wind + (20*0.6) = 10 +(20*0.6) → Wind = 22

Έτσι λοιπόν γίνεται πρώτα η τυχαία αλλαγή της τιμής της κατεύθυνσης του ανέμου, έπειτα γίνεται αλλαγή της τιμής της έντασης και μετά οι νέες τιμές εμφανίζονται στην οθόνη του υπολογιστή στα αντίστοιχα σημεία μέσα στη φόρμα, έχουμε δηλαδή **εμφάνιση δεδομένων.**

Όλη η παραπάνω διαδικασία περιγράφεται στο επόμενο διάγραμμα ροής:



7. Εμφάνιση δεδομένων στη φόρμα μας

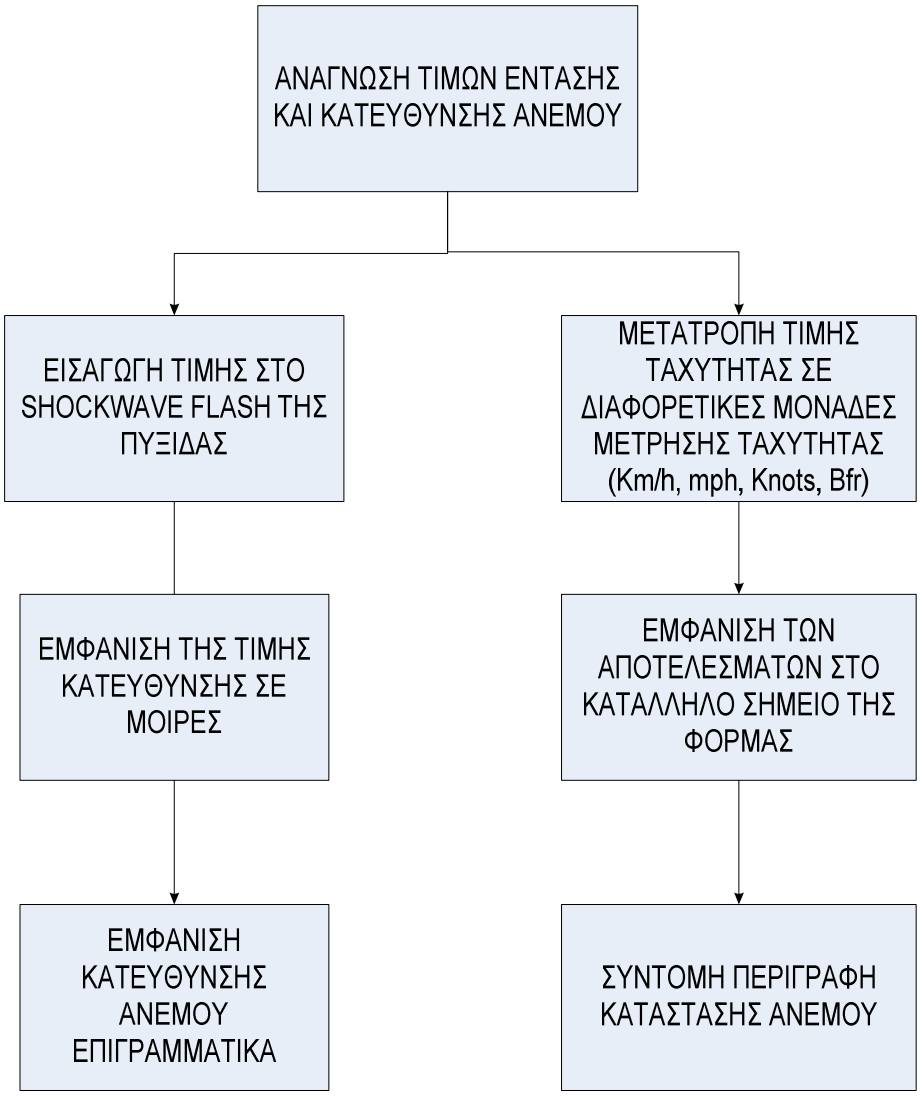
Αναφερόμενοι στον όρο «**εμφάνιση δεδομένων**» εννούμε τον τρόπο με τον οποίο οι μετρήσεις που παίρνουμε από τον αισθητήρα καθώς και οι τυχαίες τιμές της έντασης και της κατεύθυνσης του ανέμου, σε περίπτωση που έχω κατάσταση παραδείγματος, εμφανίζονται στη φόρμα του προγράμματος. Οι απαιτήσεις του προγράμματος είναι η κατεύθυνση του ανέμου να εμφανίζεται σε μοίρες και η έντασή του να εμφανίζεται στις εξής μορφές:

1)Χιλιόμετρα ανά ώρα
2)Μίλια ανά ώρα
3) Κόμβοι ανά ώρα
4) στην κλίμακα Beaufort.

Πρώτα γίνεται η ανάγνωση των μεταβλητών wind και angle. Στη μεταβλητή Angle είναι καταχωρημένη η τιμή της κατεύθυνσης σε μοίρες και στη μεταβλητή wind είναι καταχωρημένη η τιμή της ταχύτητας εκφρασμένη σε **μέτρα ανά δευτερόλεπτο**.

Έπειτα γίνονται παράλληλα δύο διαδικασίες:

1) η εμφάνιση της κατεύθυνσης και
2) η μετατροπή της ταχύτητας στις απαιτούμενες μορφές και εμφάνισή της στα αντίστοιχα σημεία στη φόρμα.



Το επόμενο βήμα είναι η καταχώρηση της τιμής angle στην πυξίδα. Έτσι κάθε φορά που αλλάζει η κατεύθυνση του ανέμου αλλάζει αντίστοιχα και η κατεύθυνση της πυξίδας. Πέρα από αυτό το γραφικό τρόπο αναγραφής έχουμε τοποθετήσει στο κάτω μέρος της πυξίδας και ένα Text Box στο οποίο αναγράφεται η κατεύθυνση σε μοίρες και επιγραμματικά.

Όσον αφορά την ταχύτητα έχουμε τα εξής στάδια μετατροπής:

1) Η τιμή που μου δίνει ο αισθητήρας είναι σε μέτρα ανά δευτερόλεπτο. Αν αυτή την τιμή την πολλαπλασιάσω επί **3,6** το αποτέλεσμα που θα πάρω θα είναι η ταχύτητα εκφρασμένη σε **Χιλιόμετρα ανά ώρα**. Η τιμή αυτή εμφανίζεται στο αντίστοιχο text box που έχουμε τοποθετήσει στη φόρμα.

2) Η τιμή που μόλις βρήκαμε (**χιλιόμετρα ανά ώρα**), αν πολλαπλασιαστεί με τον αριθμό 0.621371 θα μου δώσει την ταχύτητα εκφρασμένη πλέον σε **μίλια ανά ώρα**. Και αυτή η τιμή εμφανίζεται στο ανίστοιχο text box πάνω στη φόρμα.

3) Η ταχύτητα του ανέμου που είναι εκφρασμένη σε **χιλιόμετρα ανά ώρα**, αν πολλαπλασιαστεί αυτή τη φορά με τον αριθμό 0.539957 θα μου δώσει την ταχύτητα σε **κόμβους ανά ώρα**. Την τιμή αυτή την εμφανίζω στο αντίστοιχο Text Box πάνω στη φόρμα μου.

Ο λόγος που βάλαμε αυτούς τους 3 αριθμούς στα παραπάνω βήματα είναι επειδή 1 μέτρο ανά δευτερόλεπτο αντιστοιχεί σε 3,6 Χιλιόμετρα και επειδή το 1 χιλιόμετρο ανά ώρα αντιστοιχεί σε 0,621371 μίλια και 0,539957 κόμβους. Για να γίνει και η τελευταία μετατροπή και να εμφανίζεται η ταχύτητα και στην κλίμακα Beaufort ελέγχουμε την τιμή της ταχύτητας που εκφράζεται σε χιλιόμετρα ανά ώρα. Στο Internet αναζητήσαμε στοιχεία για την κλίμακα Beaufort . Στον πίνακα που βρήκαμε και παραθέτουμε παρακάτω βλέπουμε την αντιστοιχία των χιλιομέτρων ανά ώρα σε Beaufort. Εφαρμόζοντας την αντιστοιχία αυτή στον κώδικα έχουμε πλέον και την ένταση σε Beaufort. Τοποθετήσαμε επίσης και 3 Text Boxes στα οποία αναφέρονται τα εξής στοιχεία: γενική περιγραφή του καιρού, κατάσταση στη στεριά και κατάσταση στη θάλασσα.

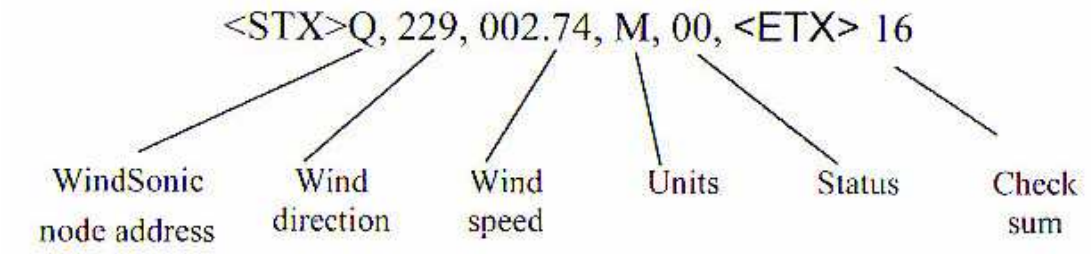
Beaufort number	Wind speed				Mean wind speed (kt / km/h / mph)	Description	Wave height		Sea conditions	Land conditions
	kt	km/h	mph	m/s			m	ft		
0	0	0	0	0-0.2	0 / 0 / 0	Calm	0	0	Flat.	Calm. Smoke rises vertically.
1	1-3	1-6	1-3	0.3-1.5	2 / 4 / 2	Light air	0.1	0.33	Ripples without crests.	Wind motion visible in smoke.
2	4-6	7-11	4-7	1.6-3.3	5 / 9 / 6	Light breeze	0.2	0.66	Small wavelets. Crests of glassy appearance, not breaking	Wind felt on exposed skin. Leaves rustle.
3	7-10	12-19	8-12	3.4-5.4	9 / 17 / 11	Gentle breeze	0.6	2	Large wavelets. Crests begin to break; scattered whitecaps	Leaves and smaller twigs in constant motion.
4	11-16	20-29	13-18	5.5-7.9	13 / 24 / 15	Moderate breeze	1	3.3	Small waves.	Dust and loose paper raised. Small branches begin to move.
5	17-21	30-39	19-24	8.0-10.7	19 / 35 / 22	Fresh breeze	2	6.6	Moderate (1.2 m) longer waves. Some foam and spray.	Smaller trees sway.
6	22-27	40-50	25-31	10.8-13.8	24 / 44 / 27	Strong breeze	3	9.9	Large waves with foam crests and some spray.	Large branches in motion. Whistling heard in overhead wires. Umbrella use becomes difficult.
7	28-33	51-62	32-38	13.9-17.1	30 / 56 / 35	Near gale	4	13.1	Sea heaps up and foam begins to streak.	Whole trees in motion. Effort needed to walk against the wind.
8	34-40	63-75	39-46	17.2-20.7	37 / 68 / 42	Gale	5.5	18	Moderately high waves with breaking crests forming spindrift. Streaks of foam.	Twigs broken from trees. Cars veer on road.
9	41-47	76-87	47-54	20.8-24.4	44 / 81 / 50	Strong gale	7	23	High waves (2.75 m) with dense foam. Wave crests start to roll over. Considerable spray.	Light structure damage.
10	48-55	88-102	55-63	24.5-28.4	52 / 96 / 60	Storm	9	29.5	Very high waves. The sea surface is white and there is considerable tumbling. Visibility is reduced.	Trees uprooted. Considerable structural damage.
11	56-63	103-117	64-72	28.5-32.6	60 / 111 / 69	Violent storm	11.5	37.7	Exceptionally high waves.	Widespread structural damage.
12	>63	>117	>72	>32.7	N/A	Hurricane	14+	46+	Huge waves. Air filled with foam and spray. Sea completely white with driving spray. Visibility very greatly reduced.	Massive and widespread damage to structures.

Έστω όμως ότι σταματάμε την κατάσταση παραδείγματος και ξεκινάμε να λαμβάνουμε δεδομένα από τον αισθητήρα.

8. Λήψη δεδομένων

- **Format δεδομένων αισθητήρα**

Το σημαντικότερο κομάτι της εργασίας είναι ο τρόπος με τον οποίο ο υπολογιστής θα αποκτήσει τα δεδομένα από τον αισθητήρα. Έχουμε ήδη αναφέρει ότι η επικοινωνία θα γίνει μέσω της σειριακής θύρας RS-232. Ο αισθητήρας έχει τη δυνατότητα να μας δώσει τα δεδομένα σε διάφορες μορφές (format). Εμείς θα χρησιμοποιήσουμε το προκαθορισμένο (default) format που μας παρέχει. Στο format αυτό τα δεδομένα μας παρέχονται με τον εξής τρόπο:



Το <STX> είναι ο χαρακτήρας 2 του κώδικα Ascii και χρησιμοποιείται για να δηλώσει την αρχή των δεδομένων.

Ο χαρακτήρας <ETX> είναι ο χαρακτήρας 3 του κώδικα Ascii και με αυτόν ο αισθητήρας δηλώνει το τέλος των δεδομένων. Άρα ότι υπάρχει ανάμεσα σε αυτούς τους δύο χαρακτήρες είναι το καθαρό μήνυμα δεδομένων που θα επεξεργαστώ στη συνέχεια.

WindSonic Node Adress: ονομάζεται διεύθυνση κόμβου. Η προεπιλεγμένη τιμή είναι το Q. Αν όμως στο δίκτυο που θα στήσω υπάρχουν και άλλες συσκευές WindSonic τότε πρέπει να γίνει μετονομασία αυτών των συσκευών με τα γράμματα από R – Z.

Wind Direction: Εδώ μου δείχνει την κατεύθυνση του ανέμου σε μοίρες. Η τιμή που μπορεί να πάρει είναι από 0 έως 359 μοίρες.

Wind Speed: Το wind Speed είναι η ταχύτητα του ανέμου. Δίπλα ακριβώς από αυτήν υπάρχουν τα Units δηλ οι μονάδες μέτρησης της ταχύτητας. Περισσότερες λεπτομέρειες δίνουμε στον παρακάτω πίνακα:

Units	Identifier
Metres per second (default)	**M**
Knots	N
Miles per hour	P
Kilometres per hour	K
Feet per minute	F

Στο παραπάνω παράδειγμα δηλαδή που η μονάδα αναφέρεται με το κωδικό γράμμα M σημαίνει ότι η ταχύτητα του ανέμου είναι 2.74 μέτρα ανά δευτερόλεπτο.

Status: Αν ο αισθητήρας λειτουργεί σωστά τότε το Status είναι 00. Αν υπάρχει κάποιο πρόβλημα τότε στη θέση του 00 εμφανίζεται ο κωδικός του αντίστοιχου λάθους όπως αναφέρεται στα φύλλα δεδομένων που δίνει ο κατασκευαστής.

CheckSum: Μετά από το χαρακτήρα τέλους ο αισθητήρας μου δίνει και το CheckSum του μηνύματος για να μπορέσω μετά να κάνω έλεγχο ορθότητας των δεδομένων. Ο αριθμός αυτός προκύπτει από τη λογική πράξη XOR του κάθε byte του μηνύματος με το διπλανό του.

- Εφαρμογή λήψης δεδομένων

Αφού περιγράψαμε τον τρόπο που αισθητήρας δίνει τα δεδομένα ας προχωρήσουμε τώρα στην επεξεργασία τους μέσω υπολογιστή.

Κατά τη διάρκεια λειτουργίας του αισθητήρα μπορεί κατά λάθος κάποιο καλώδιο να βγει από τη θέση του ή να πέσει κάποιος Driver και να μη μπορεί να συνεχίσει η εφαρμογή. Για οποιοδήποτε τέτοιο τυχαίο λάθος εμφανίζεται

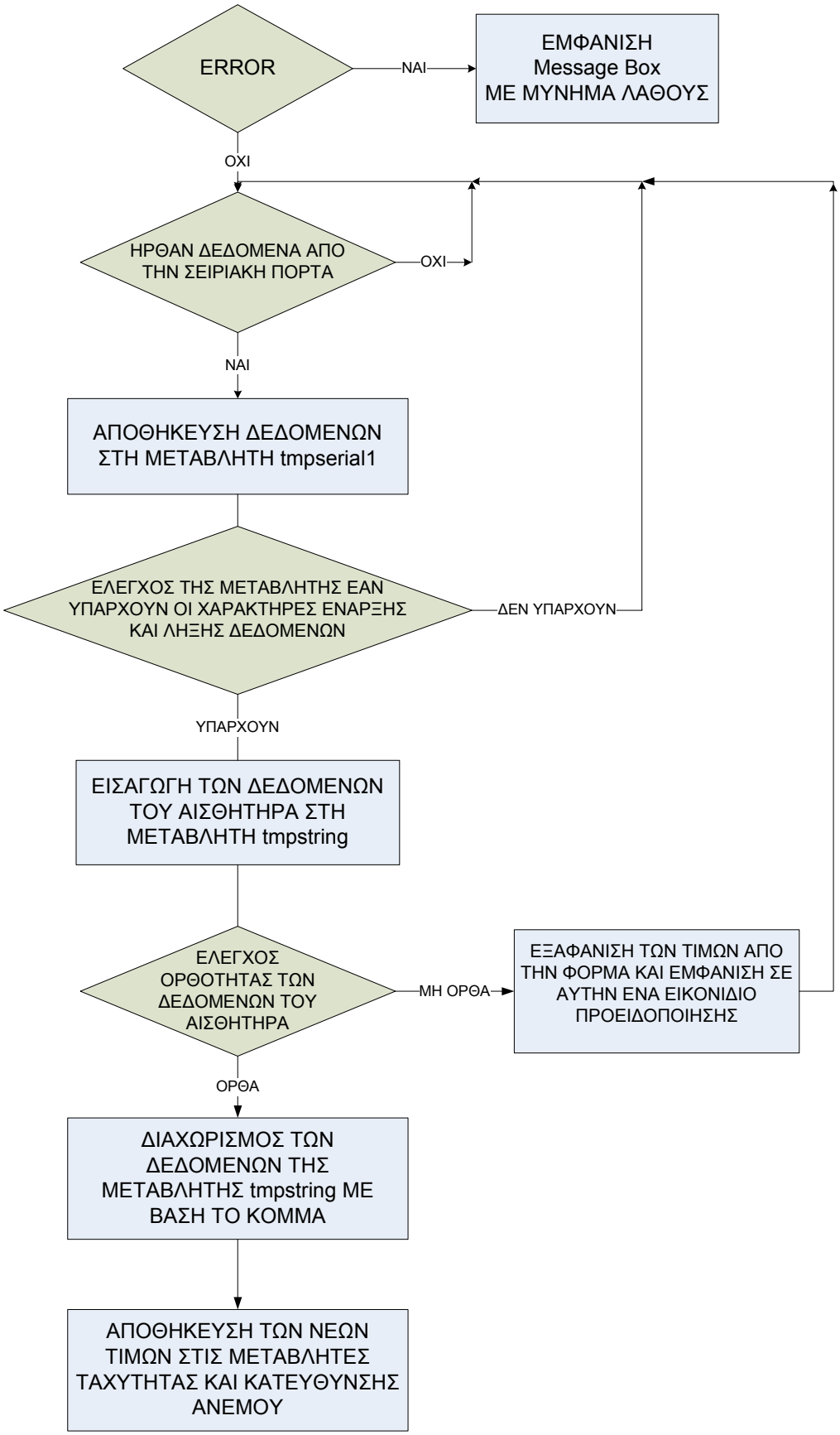
στην οθόνη του υπολογιστή ένα μήνυμα που με ενημερώνει ότι κάτι δεν πάει καλά.

Στη συνέχεια γίνεται ένας έλεγχος για το αν έφτασαν ή όχι δεδομένα στη σειριακή θύρα του υπολογιστή. Μόλις φτάσουν τα δεδομένα ο υπολογιστής κάνει και άλλον ένα έλεγχο για να δει αν υπάρχουν στο μήνυμα ο αρχικός και ο τελικός χαρακτήρας. Αν υπάρχουν σημαίνει ότι έχουν έρθει τα δεδομένα που χρειάζομαι και πηγαίνω πιο κάτω στην επεξεργασία τους. Αν δεν έχουν έρθει τότε ξαναδιαβάζει τα επόμενα δεδομένα, τα αποθηκεύει δίπλα στα προηγούμενα και ξαναγίνεται ο παραπάνω έλεγχος.

Γίνεται εισαγωγή δεδομένων στη μεταβλητή **tmpString**. Για να το κάνει αυτό ελέγχουμε τα δεδομένα που μου ήρθαν και από αυτά αποθηκεύω στην TmpString το καθαρό μήνυμα, δηλαδή τα bytes που υπάρχουν ανάμεσα στον αρχικό και τελικό χαρακτήρα. Πριν προχωρήσω στην εμφάνιση των τιμών στη φόρμα ο υπολογιστής αναλαμβάνει να κάνει επαλήθευση των δεδομένων παράγωντας και εκείνος έναν αριθμό CheckSum.

Σε περίπτωση μη ορθότητας των δεδομένων (Αλλοιωμένα δεδομένα) εμφανίζει μία εικόνα που μου λέει ότι έχω λάθος και ξαναελέγχει τη σειριακή για τα επόμενα δεδομένα που θα έρθουν για να επαναλάβει την ίδια διαδικασία. Αν τα δεδομένα δεν έχουν αλλοιωθεί κατά τη μετάδοσή τους τότε γίνεται ένας διαχωρισμός με βάση το κόμμα. Έτσι έχουμε ξεχωρίσει τις τιμές. Οι νέες τιμές αποθηκεύονται αντικαθιστώντας τις παλιές και εμφανίζονται στη φόρμα του προγράμματος.

Το διάγραμμα ροής που ακολουθεί περιγράφει τη λήψη δεδομένων από τον αισθητήρα:


ERROR
ΝΑΙ
ΕΜΦΑΝΙΣΗ
Message Box
ΜΕ ΜΥΝΗΜΑ ΛΑΘΟΥΣ
ΟΧΙ
ΗΡΘΑΝ ΔΕΔΟΜΕΝΑ ΑΠΟ
ΤΗΝ ΣΕΙΡΙΑΚΗ ΠΟΡΤΑ
ΟΧΙ
ΝΑΙ
ΑΠΟΘΗΚΕΥΣΗ ΔΕΔΟΜΕΝΩΝ
ΣΤΗ ΜΕΤΑΒΛΗΤΗ tmpserial1
ΕΛΕΓΧΟΣ ΤΗΣ ΜΕΤΑΒΛΗΤΗΣ ΕΑΝ
ΥΠΑΡΧΟΥΝ ΟΙ ΧΑΡΑΚΤΗΡΕΣ ΕΝΑΡΞΗΣ
ΚΑΙ ΛΗΞΗΣ ΔΕΔΟΜΕΝΩΝ
ΔΕΝ ΥΠΑΡΧΟΥΝ
ΥΠΑΡΧΟΥΝ
ΕΙΣΑΓΩΓΗ ΤΩΝ ΔΕΔΟΜΕΝΩΝ
ΤΟΥ ΑΙΣΘΗΤΗΡΑ ΣΤΗ
ΜΕΤΑΒΛΗΤΗ tmpstring
ΕΛΕΓΧΟΣ
ΟΡΘΟΤΗΤΑΣ ΤΩΝ
ΔΕΔΟΜΕΝΩΝ ΤΟΥ
ΑΙΣΘΗΤΗΡΑ
ΜΗ ΟΡΘΑ
ΕΞΑΦΑΝΙΣΗ ΤΩΝ ΤΙΜΩΝ ΑΠΟ
ΤΗΝ ΦΟΡΜΑ ΚΑΙ ΕΜΦΑΝΙΣΗ ΣΕ
ΑΥΤΗΝ ΕΝΑ ΕΙΚΟΝΙΔΙΟ
ΠΡΟΕΙΔΟΠΟΙΗΣΗΣ
ΟΡΘΑ
ΔΙΑΧΩΡΙΣΜΟΣ ΤΩΝ
ΔΕΔΟΜΕΝΩΝ ΤΗΣ
ΜΕΤΑΒΛΗΤΗΣ tmpstring ΜΕ
ΒΑΣΗ ΤΟ ΚΟΜΜΑ
ΑΠΟΘΗΚΕΥΣΗ ΤΩΝ ΝΕΩΝ
ΤΙΜΩΝ ΣΤΙΣ ΜΕΤΑΒΛΗΤΕΣ
ΤΑΧΥΤΗΤΑΣ ΚΑΙ ΚΑΤΕΥΘΥΝΣΗΣ
ΑΝΕΜΟΥ

9. Έλεγχος ορθότητας δεδομένων

Πιο πάνω αναφερθήκαμε στον έλεγχο ορθότητας δεδομένων. Γιατί γίνεται αυτός ο έλεγχος και πώς ακριβώς γίνεται θα αναλυθεί στις επόμενες γραμμές.

Ο κόσμος στον οποίο ζούμε δεν είναι όμορφος και αγγελικά πλασμένος. Γι αυτό και στις διάφορες επιστήμες όπως στη φυσική αλλά και στην ηλεκτρονική συναντάμε τον όρο «ιδανικό». Εξετάζουμε το ιδανικό πηνίο, την ιδανική αντίσταση, τον ιδανικό πυκνωτή, τον ιδανικό τελεστικό ενισχυτή κοκ. Έτσι και με την εφαρμογή μας δεν μπορούμε να είμαστε σίγουροι ότι τα δεδομένα που μας στέλνει ο αισθητήρας ανταποκρίνονται στην πραγματικότητα γιατί και η επικοινωνία με τον υπολογιστή δεν είναι ιδανική. Μπορεί κατά τη διάρκεια της μετάδοσης να γίνει αλλοίωση των δεδομένων και αυτό που τελικά θα πάρουμε μπορεί να είναι μία τιμή η οποία είναι λάθος. Δηλαδή αν οι συνθήκες ανέμου είναι 25 Km/h Βόρειοι άνεμοι και ο αισθητήρας μου δώσει 40 Km/h προφανώς υπάρχει λάθος κατά τη μετάδοση δεδομένων.

Γι αυτό λοιπόν τα δεδομένα πρέπει να ελέγχονται ως προς την ορθότητά τους με την τεχνική του CRC. Ο αισθητήρας κάθε φορά που στέλνει δεδομένα μου στέλνει και το CheckSum δηλαδή έναν αριθμό ο οποίος προκύπτει κάθε φορά από τα δεδομένα που έχω. Τα δεδομένα μετατρέπονται σε κώδικα Ascii, γίνεται η λογική πράξη XOR μεταξύ των bytes και στο τέλος γίνεται η μετατροπή αυτού του αριθμού σε δεκαεξαδικό σύστημα. Ο δεκαεξαδικός αυτός αριθμός που στέλνει ο αισθητήρας στον υπολογιστή είναι το CheckSum.

Για να γίνει ο έλεγχος ορθότητας των δεδομένων πρέπει και ο υπολογιστής να επαναλάβει τη διαδικασία εξαγωγής του CheckSum και να πάρω πάλι τον ίδιο αριθμό. Αν δηλαδή ο αισθητήρας μου δώσει CheckSum τον αριθμό 16 και ο υπολογιστής τον αριθμό 24 υπάρχει λάθος κατά τη μετάδοση δεδομένων και εμφανίζεται στην οθόνη ένα προειδοποιητικό

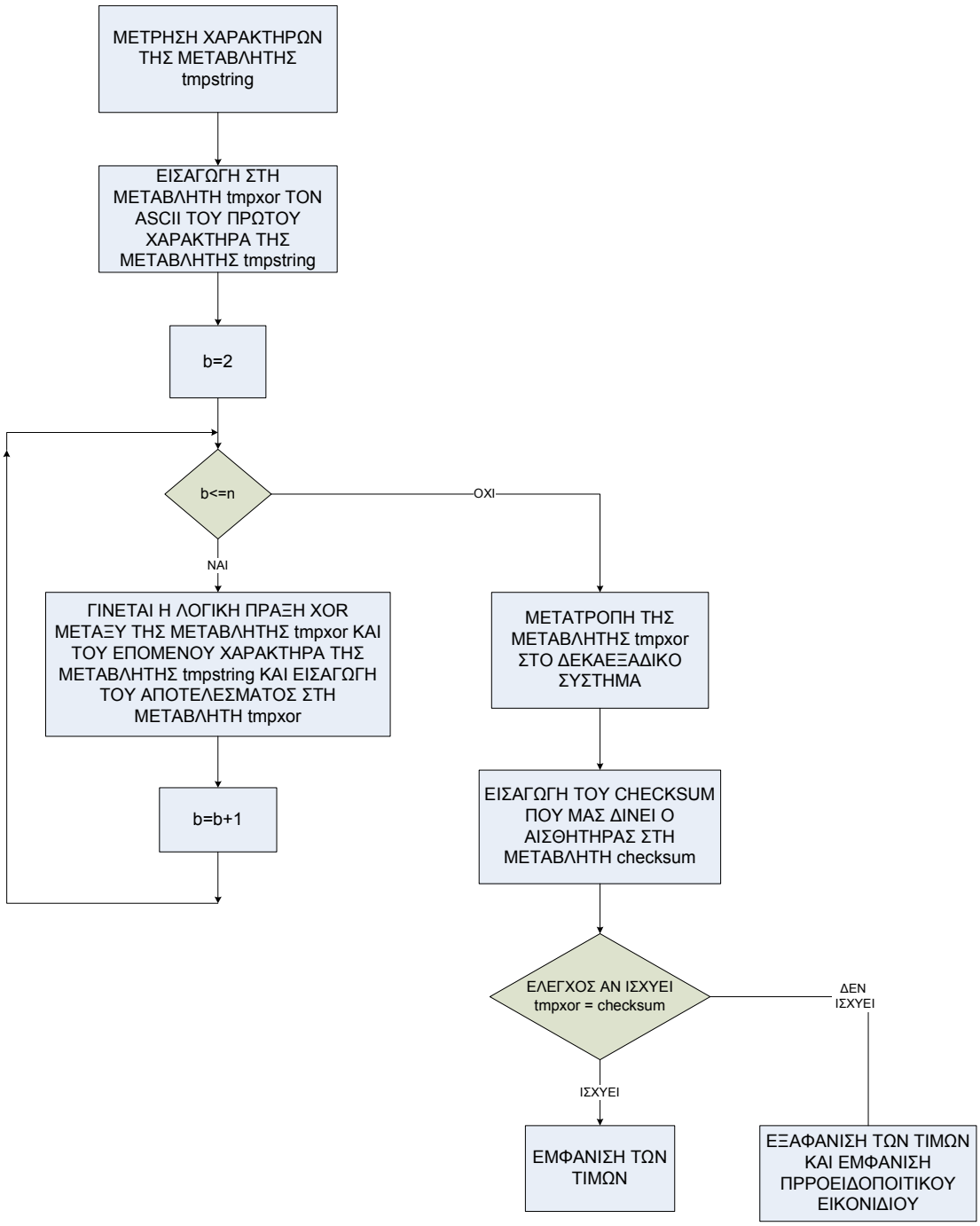
μήνυμα. Αν όμως μου δώσει 16 τότε έχουν έρθει σωστά τα δεδομένα και η εφαρμογή συνεχίζει να παίρνει δεδομένα από τον αισθητήρα.

Σε αυτό το κομάτι της εργασίας θα εξηγήσουμε τον τρόπο με τον οποίο ο υπολογιστής θα επεξεργαστεί τα δεδομένα έτσι ώστε να προκύψει το CheckSum.

Τα δεδομένα που στέλνει ο αισθητήρας αποθηκεύονται στη μεταβλητή tmpString. Αυτό που πρέπει να κάνω τώρα είναι να μετρήσω πόσους χαρακτήρες έχει αυτη η μεταβλητή και να αποθηκεύσω αυτόν τον αριθμό σε μία καινούρια μεταβλητή **n**.

Δημιουργούμε και άλλη μία μεταβλητή την **tmpXor**. Σε αυτήν αποθηκεύουμε το αποτέλεσμα της λογικής πράξης XOR του κάθε byte με το επόμενό του. Η αρχική τιμή αυτής της μεταβλητής είναι ο κωδικός Ascii του πρώτου χαρακτήρα του String. Έπειτα το πρόγραμμα ψάχνει μέσα στο String και δημιουργεί τον κωδικό Ascii του δεύτερου χαρακτήρα. Εκτελείτε η λογική πράξη XOR μεταξύ αυτού και της ήδη υπάρχουσας τιμής της μεταβλητής tmpXor. Τώρα το tmpXor έχει την XOR μεταξύ του πρώτου και του δεύτερου byte. Η διαδικασία συνεχίζεται για το επόμενο byte μέχρι να φτάσει και στο τελευταίο byte (n byte) του tmpString.

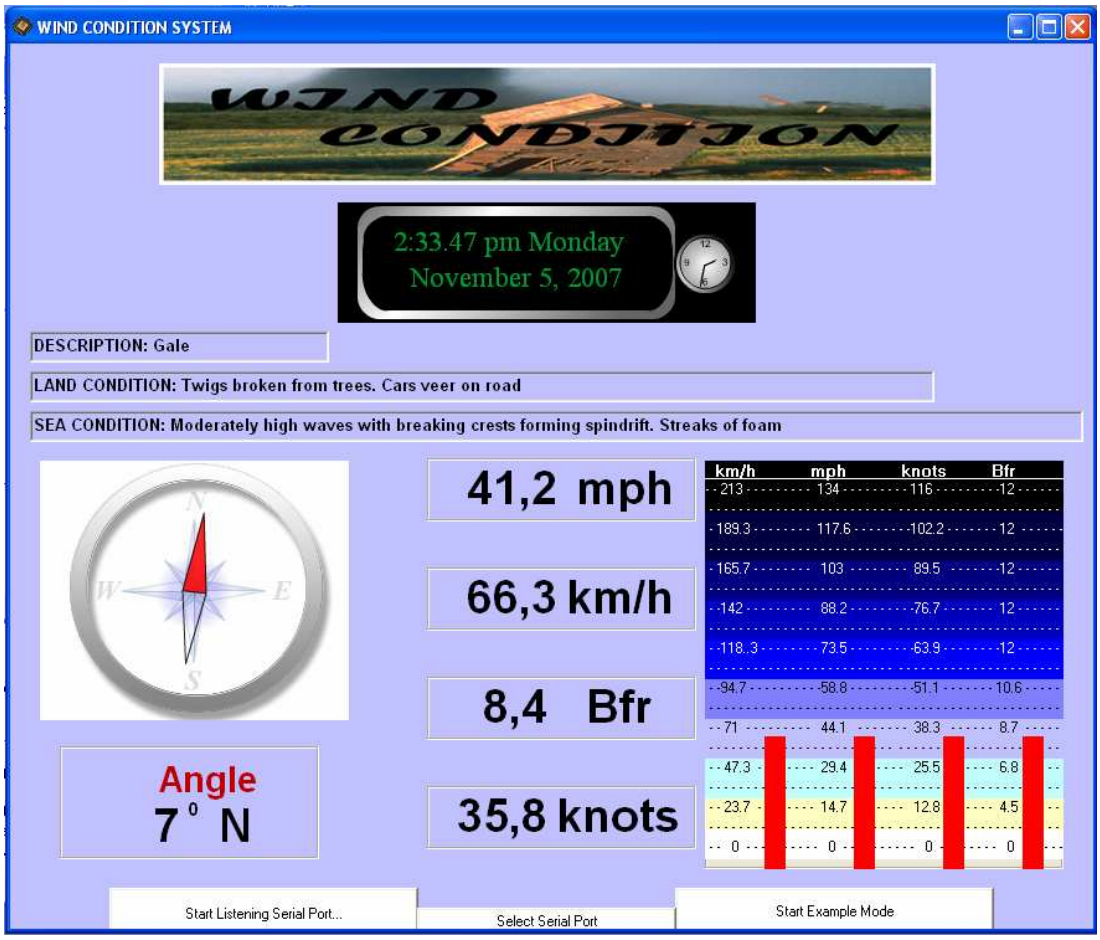
Σε αυτή τη φάση η μεταβλητή tmpXor περιέχει το CheckSum των δεδομένων σε κώδικα Ascii. Αυτή την τιμή τη μετατρέπουμε στο δεκαεξαδικό σύστημα. Τώρα ο υπολογιστής έχει επαναλάβει πλήρως τη διαδικασία παραγωγής του CheckSum. Το επόμενο βήμα είναι να γίνει έλεγχος αν αυτό που μου έστειλε ως CheckSum ο αισθητήρας συμφωνεί με αυτό που βρήκε ο υπολογιστής. Αν Αισθητήρας – Υπολογιστής συμφωνούν τότε οι τιμές



της έντασης και της κατεύθυνσης εμφανίζονται στην οθόνη του υπολογιστή. Αν δε συμφωνούν τότε εμφανίζεται μία προειδοποιητική εικόνα λάθους για 0,5 sec και οι τιμές δεν εμφανίζονται στην οθόνη.

10. Οπτική απεικόνηση της εφαρμογής

Στις προηγούμενες σελίδες εξηγήσαμε εκτενώς τον κώδικα του προγράμματος. Παρακάτω παρατίθεται η τελική οπτική απεικόνιση της εφαρμογής.



Στο πάνω μέρος του παραθύρου έχουμε τοποθετήσει ένα αντικείμενο Flash που είναι ένα ρολόι στο οποίο αναγράφεται η ώρα και η ημερομηνία του συστήματος (του υπολογιστή στον οποίο τρέχουμε την εφαρμογή).

Ακριβώς από κάτω ακολουθεί μία σύντομη περιγραφή της κατάστασης του ανέμου σε 3 μορφές:

1)Γενική περιγραφή
2)Περιγραφή των συνθηκών που επικρατούν στη στεριά και
3)Περιγραφή των συνθηκών που επικρατούν στη θάλασσα

Στην αριστερή πλευρά της φόρμας απεικονίζεται η κατεύθυνση του ανέμου γραφικά μέσω μίας πυξίδας (αντικείμενο flash), σε γωνία (°μοίρες) και επιγραμματικά.

Στο κέντρο της φόρμας εμφανίζεται η ένταση του ανέμου σε διάφορες μονάδες μέτρησης. Δίπλα από αυτές τις τιμές εμφανίζεται η ένταση και γραφικά μέσω τεσσάρων Labels των οποίων η κορυφή αυξομειώνεται ανάλογα με την τιμή της εντάσεως του ανέμου.

Στο κάτω μέρος της φόρμας έχουμε τοποθετήσει τα τρία κουμπιά τα οποία είναι υπεύθυνα για την έναρξη και τον τερματισμό της λήψης δεδομένων, την επιλογή της σειριακής θύρας που θα χρησιμοποιήσουμε καθώς επίσης και για την έναρξη και τον τερματισμό της καταστάσεως παραδείγματος.

- **<u>Επίλογος - Συμπεράσματα</u>**

Το πρόγραμμά μας θα μπορούσε να χρησιμοποιηθεί σε λιμάνια, σε γέφυρες πλοίων, πλωτές εξέδρες, σε φάρους και σε αεροδρόμια. Πέρα από αυτή τη χρήση θα μπορούσε να χρησιμοποιηθεί από κάποιον που τον ενδιαφέρει να παρακολουθεί την κατάσταση του ανέμου στο σπίτι του. Φυσικά το πρόγραμμα μπορεί να δεχτεί παραπάνω τροποποίηση αλλά για τις ανάγκες της πτυχιακής μας εργασίας κρίναμε ότι η μέχρι τώρα δουλειά πάνω στην εφαρμογή είναι αρκετά πλήρης.

Στο παρακάτω παράρτημα παραθέτουμε και τον κώδικα του προγράμματος που γράψαμε στη γλώσσα προγραμματισμού Visual Basic 6.0 Enterprise Edition.

11. Παράρτημα

```
Option Explicit

Dim Angle As Double
Dim Wind As Double
Private ListeningMode As Boolean
Private ValuesChanged As Boolean
Private DummyMode As Boolean
Private bFormLoaded As Boolean


Private Sub Command4_Click()
    Form1.Show 1
End Sub
Private Sub Form_Initialize()
    Call InitializeINIFile

    DummyMode = CBool(ReadINI("PARAMETRES", "DummyMode"))

    Angle = CDbl(ReadINI("VALUES", "Angle"))
    Wind = CDbl(ReadINI("VALUES", "Wind"))

    comSerial1.CommPort = CInt(ReadINI("PARAMETRES", "SerialPort1-
COM"))

End Sub

Private Sub InitializeINIFile()
    Call CreateINIFile("Application.ini")
    Call CreateINIApp("PARAMETRES")
    Call CreateINIKey("PARAMETRES", "SerialPort1-COM", "1")
    Call CreateINIKey("PARAMETRES", "DummyMode", "True")

    Call CreateINIApp("VALUES")
    Call CreateINIKey("VALUES", "Angle", "1")
    Call CreateINIKey("VALUES", "Wind", "1")

End Sub
```

```
Private Sub Form_Load()
    bFormLoaded = True
    Call ShockwaveFlash1.LoadMovie(0, CurDir + "\compass2.swf")
    Call ShockwaveFlash2.LoadMovie(0, CurDir + "\clock\clock106.swf")


    If DummyMode Then
        Timer1.Enabled = True
        Command3.Caption = "Stop Example Mode"
    End If
    Call SetValues
End Sub


Private Sub Form_Unload(Cancel As Integer)

Call WriteINI("PARAMETRES", "SerialPort1-COM", comSerial1.CommPort)
Call WriteINI("PARAMETRES", "DummyMode", CStr(DummyMode))

Call WriteINI("VALUES", "Angle", CStr(Angle))
Call WriteINI("VALUES", "Wind", CStr(Wind))

bFormLoaded = False

Unload Me
Set Form1 = Nothing

End Sub


Private Sub Timer1_Timer()
    If Not (DummyMode) Then
        Timer1.Enabled = False
        Exit Sub
    End If
    Call ProduceDummyValues
    Call SetValues
End Sub
```

```
Private Sub SetValues()
    Call SetAngle
    Call SetWind
    End Sub

Private Sub ProduceDummyValues()
    Angle = RandomChange(Angle, 20, 0.5)
    Wind = RandomChange(Wind, 2, 0.55)
        If Wind > 60 Then
            Wind = 60
        End If
    End Sub


Private Function RandomChange(value As Double, Change As Double,
Frequency As Single) As Double
    If Rnd() > Frequency Then
        If Rnd() > 0.5 Then
            value = value + Change * Rnd()
        Else
            value = value - Change * Rnd()
        End If
    End If

    RandomChange = value
End Function
Private Sub Command3_Click()
    DummyMode = Not (DummyMode)
    If DummyMode Then
        Timer1.Enabled = True
        Command3.Caption = "Stop Example Mode"
            If ListeningMode = "True" Then
                Call EndListeningMode
            End If
        Else
        Command3.Caption = "Start Example Mode"

    End If
End Sub
```

```
Private Sub SetAngle()
Dim Direction As String
    If Angle < 0 Then Angle = Angle + 360
    If Angle > 359 Then Angle = Angle - 360

    Call ShockwaveFlash1.SetVariable("Angle", Format(Angle, "0") +
180)
    Label1(0).Caption = Format(Angle, "0")

    If Angle > 0 And Angle < 23 Then Direction = "N"
    If Angle > 24 And Angle < 68 Then Direction = "NE"
    If Angle > 69 And Angle < 113 Then Direction = "E"
    If Angle > 114 And Angle < 158 Then Direction = "SE"
    If Angle > 159 And Angle < 203 Then Direction = "S"
    If Angle > 204 And Angle < 248 Then Direction = "SW"
    If Angle > 249 And Angle < 293 Then Direction = "W"
    If Angle > 294 And Angle < 338 Then Direction = "NW"
    If Angle > 339 And Angle < 360 Then Direction = "N"

    Label1(1).Caption = Direction
End Sub

Private Sub SetWind()
Dim tempDbl As Double

    If Wind < 0 Then Wind = 0

    'Km/h
    tempDbl = CDbl(Wind * 3.6)
    Label1(10).Caption = Format(tempDbl, "0.0")

    tempDbl = 4680 - CDbl(tempDbl) * 20.28169
    Label5(1).Top = tempDbl
    Label5(2).Top = tempDbl
    Label5(3).Top = tempDbl
    Label5(4).Top = tempDbl
```

```
    'mph
    tempDbl = CDbl(Label1(10).Caption) * 0.621371
    Label1(8).Caption = Format(tempDbl, "##0.0")

    'Knots
    tempDbl = CDbl(Label1(10).Caption) * 0.539957
    Label1(13).Caption = Format(tempDbl, "##0.0")

    'Beaufort
    tempDbl = ConvertMph2Beaufort(Label1(10).Caption)
    Label1(15).Caption = tempDbl

        If tempDbl >= 0 And tempDbl <= 0.7 Then Text1(0).Text =
"DESCRIPTION: Calm"
        If tempDbl >= 0.7 And tempDbl <= 1.9 Then Text1(0).Text =
"DESCRIPTION: Light air"
        If tempDbl >= 1.9 And tempDbl <= 3 Then Text1(0).Text =
"DESCRIPTION: Light breeze"
        If tempDbl >= 3 And tempDbl <= 4 Then Text1(0).Text =
"DESCRIPTION: Gentle breeze"
        If tempDbl >= 4 And tempDbl <= 5 Then Text1(0).Text =
"DESCRIPTION: Moderate breeze"
        If tempDbl >= 5 And tempDbl <= 6 Then Text1(0).Text =
"DESCRIPTION: Fresh breeze"
        If tempDbl >= 6 And tempDbl <= 7 Then Text1(0).Text =
"DESCRIPTION: Strong breeze"
        If tempDbl >= 7 And tempDbl <= 8 Then Text1(0).Text =
"DESCRIPTION: Near gale"
        If tempDbl >= 8 And tempDbl <= 9 Then Text1(0).Text =
"DESCRIPTION: Gale"
        If tempDbl >= 9 And tempDbl <= 10 Then Text1(0).Text =
"DESCRIPTION: Strong gale"
        If tempDbl >= 10 And tempDbl <= 11 Then Text1(0).Text =
"DESCRIPTION: Storm"
        If tempDbl >= 11 And tempDbl <= 11.9 Then Text1(0).Text =
"DESCRIPTION: Violent storm"
        If tempDbl >= 11.9 And tempDbl <= 12 Then Text1(0).Text =
"DESCRIPTION: Hurricane"
```

```
        If tempDbl >= 0 And tempDbl <= 0.7 Then Text1(1).Text = "SEA
CONDITION: Flat"
        If tempDbl >= 0.7 And tempDbl <= 1.9 Then Text1(1).Text =
"SEA CONDITION: Ripples without crests"
        If tempDbl >= 1.9 And tempDbl <= 3 Then Text1(1).Text = "SEA
CONDITION: Small wavelets. Crests of glassy appearance, not breaking"
        If tempDbl >= 3 And tempDbl <= 4 Then Text1(1).Text = "SEA
CONDITION: Large wavelets. Crests begin to break; scattered
whitecaps"
        If tempDbl >= 4 And tempDbl <= 5 Then Text1(1).Text = "SEA
CONDITION: Small waves"
        If tempDbl >= 5 And tempDbl <= 6 Then Text1(1).Text = "SEA
CONDITION: Moderate (1.2 m) longer waves. Some foam and spray"
        If tempDbl >= 6 And tempDbl <= 7 Then Text1(1).Text = "SEA
CONDITION: Large waves with foam crests and some spray"
        If tempDbl >= 7 And tempDbl <= 8 Then Text1(1).Text = "SEA
CONDITION: Sea heaps up and foam begins to streak"
        If tempDbl >= 8 And tempDbl <= 9 Then Text1(1).Text = "SEA
CONDITION: Moderately high waves with breaking crests forming
spindrift. Streaks of foam"
        If tempDbl >= 9 And tempDbl <= 10 Then Text1(1).Text = "SEA
CONDITION: High waves (2.75 m) with dense foam. Wave crests start to
roll over. Considerable spray"
        If tempDbl >= 10 And tempDbl <= 11 Then Text1(1).Text = "SEA
CONDITION: Very high waves. The sea surface is white and there is
considerable tumbling. Visibility is reduced"
        If tempDbl >= 11 And tempDbl <= 11.9 Then Text1(1).Text =
"SEA CONDITION: Exceptionally high waves"
        If tempDbl >= 11.9 And tempDbl <= 12 Then Text1(1).Text =
"SEA CONDITION: Huge waves. Air filled with foam and spray. Sea
completely white with driving spray. Visibility very greatly reduced"
   If tempDbl >= 0 And tempDbl <= 0.7 Then Text1(2).Text = "LAND
CONDITION: Calm. Smoke rises vertically"
        If tempDbl > 0.7 And tempDbl <= 1.9 Then Text1(2).Text =
"LAND CONDITION: Wind motion visible in smoke"
        If tempDbl >= 1.9 And tempDbl <= 3 Then Text1(2).Text = "LAND
CONDITION: Wind felt on exposed skin. Leaves rustle"
        If tempDbl >= 3 And tempDbl <= 4 Then Text1(2).Text = "LAND
CONDITION: Leaves and smaller twigs in constant motion"
```

```
        If tempDbl >= 4 And tempDbl <= 5 Then Text1(2).Text = "LAND
CONDITION: Dust and loose paper raised. Small branches begin to move"
        If tempDbl >= 5 And tempDbl <= 6 Then Text1(2).Text = "LAND
CONDITION: Smaller trees sway"
        If tempDbl >= 6 And tempDbl <= 7 Then Text1(2).Text = "LAND
CONDITION: Large branches in motion. Whistling heard in overhead
wires. Umbrella use becomes difficult"
        If tempDbl >= 7 And tempDbl <= 8 Then Text1(2).Text = "LAND
CONDITION: Whole trees in motion. Effort needed to walk against the
wind"
        If tempDbl >= 8 And tempDbl <= 9 Then Text1(2).Text = "LAND
CONDITION: Twigs broken from trees. Cars veer on road"
        If tempDbl >= 9 And tempDbl <= 10 Then Text1(2).Text = "LAND
CONDITION: Light structure damage"
        If tempDbl >= 10 And tempDbl <= 11 Then Text1(2).Text = "LAND
CONDITION: Trees uprooted. Considerable structural damage"
        If tempDbl >= 11 And tempDbl <= 11.9 Then Text1(2).Text =
"LAND CONDITION: Widespread structural damage"
        If tempDbl >= 11.9 And tempDbl <= 12 Then Text1(2).Text =
"LAND CONDITION: Massive and widespread damage to structures"

End Sub


Private Function ConvertMph2Beaufort(value As Double)
Dim tmpDbl As Double


    If value >= 0 And value <= 0.1 Then tmpDbl = 0.1
    If value >= 0.1 And value <= 0.2 Then tmpDbl = 0.2
    If value >= 0.2 And value <= 0.3 Then tmpDbl = 0.3


    If value >= 0.3 And value <= 0.4 Then tmpDbl = 0.4
    If value >= 0.4 And value <= 0.5 Then tmpDbl = 0.5
    If value >= 0.5 And value <= 0.6 Then tmpDbl = 0.6
    If value >= 0.6 And value <= 0.7 Then tmpDbl = 0.7
    If value >= 0.7 And value <= 0.8 Then tmpDbl = 0.8
    If value >= 0.8 And value <= 0.9 Then tmpDbl = 0.9
```

```
If value >= 0.9 And value <= 1 Then tmpDbl = 1
If value >= 1 And value <= 1.5 Then tmpDbl = 1.1
If value >= 1.5 And value <= 2 Then tmpDbl = 1.2
If value >= 2 And value <= 2.5 Then tmpDbl = 1.3
If value >= 2.5 And value <= 3 Then tmpDbl = 1.4
If value >= 3 And value <= 3.5 Then tmpDbl = 1.5
If value >= 3.5 And value <= 4 Then tmpDbl = 1.6
If value >= 4 And value <= 4.5 Then tmpDbl = 1.7
If value >= 4.5 And value <= 5 Then tmpDbl = 1.8
If value >= 5 And value <= 5.5 Then tmpDbl = 1.9
If value >= 5.5 And value <= 6 Then tmpDbl = 2
If value >= 6 And value <= 6.5 Then tmpDbl = 2.1
If value >= 6.5 And value <= 7 Then tmpDbl = 2.2
If value >= 7 And value <= 7.5 Then tmpDbl = 2.3
If value >= 7.5 And value <= 8 Then tmpDbl = 2.4
If value >= 8 And value <= 8.5 Then tmpDbl = 2.5
If value >= 8.5 And value <= 9 Then tmpDbl = 2.6
If value >= 9 And value <= 9.5 Then tmpDbl = 2.7
If value >= 9.5 And value <= 10 Then tmpDbl = 2.8
If value >= 10 And value <= 10.5 Then tmpDbl = 2.9
If value >= 10.5 And value <= 11 Then tmpDbl = 3
If value >= 11 And value <= 11.8 Then tmpDbl = 3.1
If value >= 11.8 And value <= 12.6 Then tmpDbl = 3.2
If value >= 12.6 And value <= 13.4 Then tmpDbl = 3.3
If value >= 13.4 And value <= 14.2 Then tmpDbl = 3.4
If value >= 14.2 And value <= 15 Then tmpDbl = 3.5
If value >= 15 And value <= 15.8 Then tmpDbl = 3.6
If value >= 15.8 And value <= 16.6 Then tmpDbl = 3.7
If value >= 16.6 And value <= 17.4 Then tmpDbl = 3.8
If value >= 17.4 And value <= 18.2 Then tmpDbl = 3.9
If value >= 18.2 And value <= 19 Then tmpDbl = 4
If value >= 19 And value <= 20 Then tmpDbl = 4.1
If value >= 20 And value <= 21 Then tmpDbl = 4.2
If value >= 21 And value <= 22 Then tmpDbl = 4.3
If value >= 22 And value <= 23 Then tmpDbl = 4.4
If value >= 23 And value <= 24 Then tmpDbl = 4.5
If value >= 24 And value <= 25 Then tmpDbl = 4.6
If value >= 25 And value <= 26 Then tmpDbl = 4.7
If value >= 26 And value <= 27 Then tmpDbl = 4.8
```

```
If value >= 27 And value <= 28 Then tmpDbl = 4.9
If value >= 28 And value <= 29 Then tmpDbl = 5
If value >= 29 And value <= 30 Then tmpDbl = 5.1
If value >= 30 And value <= 31 Then tmpDbl = 5.2
If value >= 31 And value <= 32 Then tmpDbl = 5.3
If value >= 32 And value <= 33 Then tmpDbl = 5.4
If value >= 33 And value <= 34 Then tmpDbl = 5.5
If value >= 34 And value <= 35 Then tmpDbl = 5.6
If value >= 35 And value <= 36 Then tmpDbl = 5.7
If value >= 37 And value <= 37 Then tmpDbl = 5.8
If value >= 37 And value <= 38 Then tmpDbl = 5.9
If value >= 38 And value <= 39 Then tmpDbl = 6
If value >= 39 And value <= 40.1 Then tmpDbl = 6.1
If value >= 40.1 And value <= 41.2 Then tmpDbl = 6.2
If value >= 41.2 And value <= 42.3 Then tmpDbl = 6.3
If value >= 42.3 And value <= 43.4 Then tmpDbl = 6.4
If value >= 43.4 And value <= 44.5 Then tmpDbl = 6.5
If value >= 44.5 And value <= 45.6 Then tmpDbl = 6.6
If value >= 45.6 And value <= 46.7 Then tmpDbl = 6.7
If value >= 46.7 And value <= 47.8 Then tmpDbl = 6.8
If value >= 47.8 And value <= 48.9 Then tmpDbl = 6.9
If value >= 48.9 And value <= 50 Then tmpDbl = 7
If value >= 50 And value <= 51.2 Then tmpDbl = 7.1
If value >= 51.2 And value <= 52.4 Then tmpDbl = 7.2
If value >= 52.4 And value <= 53.6 Then tmpDbl = 7.3
If value >= 53.6 And value <= 54.8 Then tmpDbl = 7.4
If value >= 54.8 And value <= 56 Then tmpDbl = 7.5
If value >= 56 And value <= 57.2 Then tmpDbl = 7.6
If value >= 57.2 And value <= 58.4 Then tmpDbl = 7.7
If value >= 58.4 And value <= 59.6 Then tmpDbl = 7.8
If value >= 59.6 And value <= 60.8 Then tmpDbl = 7.9
If value >= 60.8 And value <= 62 Then tmpDbl = 8
If value >= 62 And value <= 63.3 Then tmpDbl = 8.1
If value >= 63.3 And value <= 64.6 Then tmpDbl = 8.2
If value >= 64.6 And value <= 65.9 Then tmpDbl = 8.3
If value >= 65.9 And value <= 67.2 Then tmpDbl = 8.4
If value >= 67.2 And value <= 68.5 Then tmpDbl = 8.5
If value >= 68.5 And value <= 69.8 Then tmpDbl = 8.6
If value >= 69.8 And value <= 71.1 Then tmpDbl = 8.7
```

```
    If value >= 71.1 And value <= 72.4 Then tmpDbl = 8.8
    If value >= 72.4 And value <= 73.7 Then tmpDbl = 8.9
    If value >= 73.7 And value <= 75 Then tmpDbl = 9
    If value >= 75 And value <= 76.2 Then tmpDbl = 9.1
    If value >= 76.2 And value <= 77.4 Then tmpDbl = 9.2
    If value >= 77.4 And value <= 78.6 Then tmpDbl = 9.3
    If value >= 78.6 And value <= 79.8 Then tmpDbl = 9.4
    If value >= 79.8 And value <= 81 Then tmpDbl = 9.5
    If value >= 81 And value <= 82.2 Then tmpDbl = 9.6
    If value >= 82.2 And value <= 83.4 Then tmpDbl = 9.7
    If value >= 83.4 And value <= 84.6 Then tmpDbl = 9.8
    If value >= 84.6 And value <= 85.8 Then tmpDbl = 9.9
    If value >= 85.8 And value <= 87 Then tmpDbl = 10
    If value >= 87 And value <= 88.5 Then tmpDbl = 10.1
    If value >= 88.5 And value <= 90 Then tmpDbl = 10.2
    If value >= 90 And value <= 91.5 Then tmpDbl = 10.3
    If value >= 91.5 And value <= 93 Then tmpDbl = 10.4
    If value >= 93 And value <= 94.5 Then tmpDbl = 10.5
    If value >= 94.5 And value <= 96 Then tmpDbl = 10.6
    If value >= 96 And value <= 97.5 Then tmpDbl = 10.7
    If value >= 97.5 And value <= 99 Then tmpDbl = 10.8
    If value >= 99 And value <= 100.5 Then tmpDbl = 10.9
    If value >= 100.5 And value <= 102 Then tmpDbl = 11
    If value >= 102 And value <= 103.5 Then tmpDbl = 11.1
    If value >= 103.5 And value <= 105 Then tmpDbl = 11.2
    If value >= 105 And value <= 106.5 Then tmpDbl = 11.3
    If value >= 106.5 And value <= 108 Then tmpDbl = 11.4
    If value >= 108 And value <= 109.5 Then tmpDbl = 11.5
    If value >= 109.5 And value <= 111 Then tmpDbl = 11.6
    If value >= 111 And value <= 112.5 Then tmpDbl = 11.7
    If value >= 112.5 And value <= 114 Then tmpDbl = 11.8
    If value >= 114 And value <= 115.5 Then tmpDbl = 11.9
    If value >= 115.5 And value <= 117 Then tmpDbl = 12
    If value >= 117 Then tmpDbl = 12

 ConvertMph2Beaufort = tmpDbl
End Function
```

```
Private Sub Command1_Click()
    If DummyMode = True Then
        DummyMode = False
        Command3.Caption = "Start Example Mode"
    End If
    Call ListenSerialPort
End Sub



Private Sub EndListeningMode()

    ListeningMode = False
    Command1.Caption = "Start Listening Serial Port"
    Command4.Enabled = True
    If comSerial1.PortOpen Then comSerial1.PortOpen = False
    SerialPortTimer.Enabled = False

End Sub

Private Sub ListenSerialPort()
'Start Listening Serial Port

Dim tmpString  As String, tmpSerial1 As String
Dim i As Integer
Dim DataReceived As Boolean
Dim tmpDate As String
Dim tmpStart As Integer
Dim tmpEnd As Integer
Dim tmpVal As String
Dim tmpValues
Dim tmpPos As Integer
    'Checksum Variables
Dim tmpCheckSum As String
Dim n As Integer
Dim b As Integer
Dim tmpxor As Double
```

```
      If Not (ListeningMode) Then
        ListeningMode = True
        Command1.Caption = "Stop listening Serial Port"
        Command4.Enabled = False

    Else
        Call EndListeningMode
        Exit Sub
    End If

    comSerial1.InputLen = 0
    tmpSerial1 = ""
    ValuesChanged = False


    While ListeningMode

        On Error GoTo Error1

        If Not (comSerial1.PortOpen) Then comSerial1.PortOpen = True


                '~~~~~~~~~~~~~~~~~~~~~~~~~~~
                'Data received in comSerial1
                '~~~~~~~~~~~~~~~~~~~~~~~~~~~

     If comSerial1.InBufferCount Then 'Data Received in Serial Port 1
                    Debug.Print "comSerial1.InBufferCount = " &
comSerial1.InBufferCount

                    tmpSerial1 = tmpSerial1 & CStr(comSerial1.Input)



If InStr(1, tmpSerial1, Chr(3)) > 0 And InStr(1, tmpSerial1, Chr(2))
> 0 Then
      'Full data string arrived in Serial Port 1
  tmpSerial1 = tmpSerial1 & CStr(comSerial1.Input)
ValuesChanged = True
```

```
            'Data format
            'Chr(2) & "Q,229,002.74,M,00," & Chr(3) 16
            ' 229 -> Angle
            ' 002.74 -> Wind

            'Get Real Data..
tmpStart = InStr(1, tmpSerial1, Chr(2))
tmpEnd = InStr(1, tmpSerial1, Chr(3))
tmpString = Mid(tmpSerial1, tmpStart + 1, tmpEnd - tmpStart - 1)
tmpSerial1 = Right(tmpSerial1, Len(tmpSerial1) - tmpEnd)

             'check data start

          n = Len(tmpString)

         tmpxor = Asc("Mid(tmpString, 1, 1")
             For b = 2 To n
                tmpxor = tmpxor Xor Asc(Mid(tmpString, b, 1))
             Next
                tmpxor = Hex(tmpxor)

              tmpCheckSum = tmpSerial1
               If InStr(1, tmpSerial1, Chr(2)) > 0 Then
                 tmpCheckSum = Left(tmpSerial1, Chr(2))
               End If

               If tmpxor = tmpCheckSum Then
                  Timer2 = False
                     Else
                  Timer2 = True
                  Call cleardata
                End If

      tmpValues = Split(tmpString, ",")

      tmpVal = tmpValues(1)
          If IsNumeric(tmpVal) Then
             Angle = CDbl(Replace(tmpVal, ".", ","))
          End If
```

```
            tmpVal = tmpValues(2)

            If IsNumeric(tmpVal) Then
               Wind = CDbl(Replace(tmpVal, ".", ","))
            End If

          SerialPortTimer.Enabled = False
          SerialPortTimer.Enabled = True  'Start / ReStart Counter
to check for connection Time-Out

        End If
 End If


              'Redraw Values
              If ValuesChanged Then
                  Call SetValues
                  ValuesChanged = False
              End If

              DoEvents    'To check whether another button is
pressed

              If Not (bFormLoaded) Then
                  Unload Form1
                  Set Form1 = Nothing
                  Exit Sub
              End If
              If Not (ListeningMode) Then
                  Call EndListeningMode
                  Exit Sub
              End If
   Wend
    'End Listening Mode - Close Ports
    If comSerial1.PortOpen Then comSerial1.PortOpen = False

    Exit Sub
```

```
Error1:
    MsgBox "Error " & Err.Description, vbCritical
End Sub


Private Sub cleardata()
    Label1(8).Caption = ""
    Label1(10).Caption = ""
    Label1(15).Caption = ""
    Label1(13).Caption = ""
    ValuesChanged = False

End Sub
```