ΕΘΝΙΚΟ ΜΕΤΣΟΒΕΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΗΛΕΚΤΡΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΩΝ ΔΙΑΤΑΞΕΩΝ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΑΠΟΦΑΣΕΩΝ

ΔΙΑΤΑΞΗ ΣΥΓΧΡΟΝΙΣΜΟΥ ΜΕΤΡΗΣΕΩΝ ΜΕ ΧΡΗΣΗ ΜΙΚΡΟΕΛΕΓΚΤΗ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Κιούσης Γεώργιος

Επιβλέπων καθηγητής: **Τσαραμπάρης Παναγιώτης**

Λέκτορας Ε.Μ.Π.

Αθήνα, Μάρτιος 2017

ΕΘΝΙΚΟ ΜΕΤΣΟΒΕΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΗΛΕΚΤΡΙΚΩΝ ΒΙΟΜΗΧΑΝΙΚΩΝ ΔΙΑΤΑΞΕΩΝ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΑΠΟΦΑΣΕΩΝ

ΔΙΑΤΑΞΗ ΣΥΓΧΡΟΝΙΣΜΟΥ ΜΕΤΡΗΣΕΩΝ ΜΕ ΧΡΗΣΗ ΜΙΚΡΟΕΛΕΓΚΤΗ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Κιούσης Γεώργιος

Επιβλέπων καθηγητής: **Τσαραμπάρης Παναγιώτης**

Λέκτορας Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 02.03.2017

.....	……	……
Π. Τσαραμπάρης	**Ν. Θεοδώρου**	**Μ. Ιωαννίδου**
Λέκτορας Ε.Μ.Π.	**Καθηγητής Ε.Μ.Π.**	**Καθηγήτρια Ε.Μ.Π.**

Αθήνα, Μάρτιος 2017

.....

Κιούσης Γεώργιος

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π

ΠΕΡΙΛΗΨΗ

Στη συγκεκριμένη διπλωματική εργασία σχεδιάζεται και υλοποιείται μία διάταξη συγχρονισμού μετρήσεων βασισμένη στη πλατφόρμα του Arduino. Η διάταξη αυτή θα παρέχει τη δυνατότητα στο χειριστή να εισάγει δεδομένα, μέσω μιας εφαρμογής διεπαφής με στόχο τον έλεγχο και το συντονισμό μέσω ενός μικροελεγκτή μιας ομάδας συσκευών ή συστημάτων. Βασικός προσανατολισμός υπήρξε η ευκολία στη χρήση χωρίς την υποβάθμιση των δυνατοτήτων της, γεγονός που καθιστά ποικίλο το φάσμα της μελλοντική χρήση της διάταξης αυτής.

ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ

Διάταξη συγχρονισμού μετρήσεων, μικροελεγκτής, μικροεπεξεργαστής, Arduino, LazarusIDE, ανοιχτός κώδικας, εφαρμογές χαμηλού κόστους.

ABSTRACT

This thesis covers the design and implementation of a synchronization measurement device based on the Arduino platform. This system will give the opportunity and the ability to the operator to input data, through an interface application, in order to control and synchronize with the help of a microcontroller a group of devices or systems. The main direction of this work was to simplify the use of the device without downgrading the capabilities of the entire system, giving it a varied spectrum of future applications.

KEY WORDS

Synchronization device, measurements, microcontroller, microprocessor, Arduino, Lazarus IDE, open source, low cost.

ΠΡΟΛΟΓΟΣ

Ξεκινώντας, θα ήθελα να κάνω μία ιστορική αναδρομή από τη στιγμή που ξεκίνησα να αναπτύξω την παρούσα εργασία. Έχοντας ακολουθήσει τον τομέα του Ηλεκτρολόγου Μηχανικού και Μηχανικού Ηλεκτρονικών Υπολογιστών, γνώρισα πολύ καλά πως αυτός ο τομέας αγγίζει και συνδέεται με μία πληθώρα εφαρμογών και πεδίων. Σε μία εποχή που ο κόσμος γίνεται κάθε μέρα όλο και πιο ψηφιακός, πλημμυρισμένος από χιλιάδες ηλεκτρονικές συσκευές, από απλά gadgets μέχρι πολύπλοκα συστήματα, απαραίτητα πλέον στην καθημερινότητά μας, δεν υπήρχε άλλος δρόμος από το να αναμιχθώ στη διαδικασία σκέψης, σχεδιασμού και υλοποίησης τέτοιων συστημάτων. Η εμπειρία όταν ξεκίνησα ήταν ελάχιστη. Έτσι αποφάσισα το αντικείμενο της διπλωματικής να είναι τέτοιο ώστε να δίνει τη δυνατότητα να γίνει η ίδια η εργασία ένας δρόμος απόκτησης εμπειριών και γνώσεων, αλλά κυρίαρχα ένα μέσο να κατανόησης και να εμβάθυνσης στο σύνολο του τομέα αυτού. Ακολουθώντας πιστά τη συμβουλή των καθηγητών μου , ότι «οποιαδήποτε ιδέα σκεφτούμε μπορούμε να την υλοποιήσουμε **μόνοι μας»,** ξεκίνησα τον προγραμματισμό και την υλοποίηση της παρούσας διπλωματικής εργασίας.

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία εκπονήθηκε από το φοιτητή Κιούση Γεώργιου του τμήματος Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Ε.Μ.Π. κατά το ακαδημαϊκό έτος 2016-2017.

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα της διπλωματικής,κ. Παναγιώτη Τσαραμπάρη για την ανάθεση, την υποστήριξη και τη καθοδήγηση που μου προσέφερε καθ' όλη τη διάρκεια διεκπεραίωσης της παρούσας διπλωματικής εργασίας.

Θα ήθελα να ευχαριστήσω ακόμα τους καθηγητές του τμήματος για τις πολύτιμες γνώσεις που μου μετέδωσαν κατά τη φοίτησή μου στο τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Ε.Μ.Π.

Τέλος ευχαριστώ θερμά την οικογένειά μου για την ηθική και υλική τους στήριξη προκειμένου να μπορέσω να ολοκληρώσω τις σπουδές μου και να εκπληρώσω απερίσπαστα τους στόχους μου.

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

ΚΕΦΑΛΑΙΟ 1ο

ΕΙΣΑΓΩΓΗ

Στο κεφάλαιο γίνεται αρχικά παρουσίαση στο θεωρητικό κομμάτι που αφορά τις μετρήσεις, αποτελώντας ουσιαστικά μια βάση για την υπόλοιπη διάρθρωση της διπλωματικής αυτής.

Στη συνέχεια αναλύεται η χρονολογική και η τεχνολογική εξέλιξη των μικροεπεξεργαστών και των μικροελεγκτών. Περιγράφονται οι βασικές αρχές λειτουργίας τους όπως έχουν διαμορφωθεί τη σημερινή εποχή, η δομή τους, αλλά και ορισμένες βασικές διαφοροποιήσεις μεταξύ τους.

Ακολουθεί μία αναλυτική περιγραφή του μικροϋπολογιστικού συστήματος Arduino, βασικού στοιχείου στην υλοποίηση της συγκεκριμένης εφαρμογής. Προβάλλεται μια δομημένη και παράλληλα μια ολοκληρωμένη παρουσίαση τόσο του hardware, όσο και του software του Arduino, με αντικειμενικό σκοπό την σφαιρική πληροφόρηση και κατανόηση των δυνατοτήτων του. Επιπλέον έχουν προστεθεί εφαρμογές που είναι βασισμένες στην πλατφόρμα του Arduino.

Στο τελευταίο μέρος της εισαγωγής γίνεται μία ολοκληρωμένη ανάλυση της πλατφόρμας και του περιβάλλοντος προγραμματισμού Lazarus IDE, μέσω του οποίου θα επιτευχθεί η επικοινωνία χρήστη και συστήματος.

1.1) ΜΕΤΡΗΣΕΙΣ

1.1.1) ΟΡΙΣΜΟΣ

Ο όρος μέτρηση μπορεί να σημαίνει είτε απαρίθμηση με χρήση των φυσικών αριθμών, είτε σύγκριση της ποσότητας κάποιου φυσικού μεγέθους με ένα πρότυπο, δηλαδή σύγκριση με κάποια σταθερή ποσότητα του ίδιου φυσικού μεγέθους που αυθαίρετα έχει συμφωνηθεί (κατά «σύμβαση», δηλαδή κατά κοινή συμφωνία) να χρησιμοποιείται ως μονάδα μέτρησης. Οι μετρήσεις είναι εξαιρετικά σημαντικές στην επιστήμη, την τεχνολογία και τη βιομηχανία. Η ανάπτυξη τεχνικών για την ακριβή μέτρηση μεγεθών όπως η μάζα και ο χρόνος αποτέλεσε προϋπόθεση για τη λεπτομερή και προσεκτική παρατήρηση της φύσης και την ανάπτυξη της επιστήμης της φυσικής.

Κλασσικός ορισμός:
Ο κλασσικός ορισμός, ο οποίος είναι κοινός για όλες τις επιστήμες, ορίζει τη μέτρηση ως τον προσδιορισμό ή την εκτίμηση του λόγου των ποσοτήτων. Ποσότητα και μέτρηση ορίζονται αμοιβαία: Ποσοτικές ιδιότητες είναι εκείνες που δύνανται να μετρηθούν, τουλάχιστον κατ' αρχήν. Η κλασσική έννοια της ποσότητας μπορεί να αναχθεί στους John Wallis και Isaac Newton, ενώ οι ρίζες της βρίσκονται στα «Στοιχεία» του Ευκλείδη. Τα είδη των μετρήσεων μπορούν να διακριθούν σε:
Α) Μετρήσεις φυσικών μεγεθών
Β) Μετρήσεις στατιστικών δεδομένων
Γ) Προσδιορισμός σταθερών

1.1.2) ΜΕΤΡΟΛΟΓΙΑ

Σύμφωνα με το Διεθνές Γραφείο Μέτρων και σταθμών (International Bureau of Weights and Measures -IBWM) η Μετρολογία ορίζεται ως η επιστήμη που ασχολείται με τις μετρήσεις, τα όργανα μέτρησης, την αξιοπιστία των οργάνων και των μετρήσεων, τις μονάδες μέτρησης και γενικότερα με ότι αφορά στον ακριβή και αποδεκτό προσδιορισμό των τιμών των μεγεθών που προσδιορίζουν τις ιδιότητες των σωμάτων ή των συστημάτων. Οι αρχές της βρίσκουν εφαρμογή σε όλες τις επιστήμες που χρησιμοποιούν τη μέτρηση αλλά και σε τομείς της καθημερινής ζωής, στις συναλλαγές, στη βιομηχανία, στην ασφάλεια, στην υγεία, στον έλεγχο ποιότητας κ.ά. Η επιστήμη της μέτρησης περιλαμβάνει τόσο πειραματικούς όσο και θεωρητικούς προσδιορισμούς σε κάθε βαθμό αβεβαιότητας σε οποιοδήποτε πεδίο της επιστήμης ή της τεχνολογίας. Κανονικά περιλαμβάνει τεχνικές και μεθόδους μετρήσεων, καθώς και την τεχνολογία των οργάνων μέτρησης (οργανολογία). Όμως είναι συνηθισμένο παραπέμπει πια σε δύο βασικές αποστολές/δραστηριότητες:

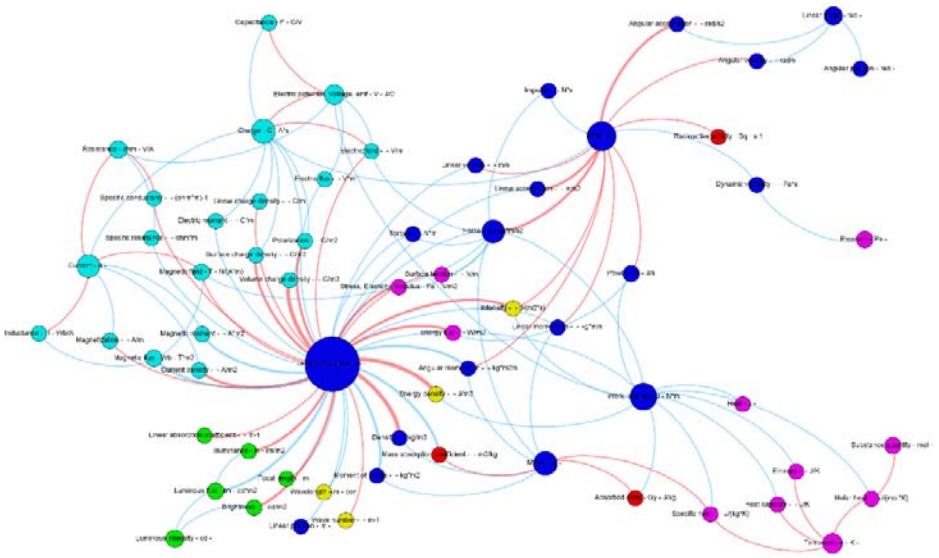
Α) Διασφάλιση ποιότητας (υποστήριξη)
Β) Προτυποποίηση

Η μετρολογία μπορεί να έχει υπόσταση:
Α) Επιστημονική, π.χ. ορισμός πρότυπων μονάδων
Β) Εφαρμοσμένη, π.χ. διακρίβωση και ποιότητα μετρήσεων
Γ) Νομική, π.χ. στη σύνταξη κανονισμών και δημιουργία και έλεγχο προτύπων.

1.1.3) ΣΥΣΤΗΜΑΤΑ ΜΟΝΑΔΩΝ ΚΑΙ ΜΟΝΑΔΕΣ ΜΕΤΡΗΣΗΣ

Ιστορικά οι άνθρωποι δημιούργησαν και χρησιμοποίησαν πολλά διαφορετικά συστήματα μονάδων μέτρησης, αρχικά για τη μέτρηση των αποστάσεων και για τη μέτρηση ποσοτήτων όπως η μάζα (το βάρος) και ο όγκος για εμπορικούς και παρόμοιους σκοπούς. Υπάρχει το βαβυλωνιακό σύστημα, το αιγυπτιακό σύστημα, το ελληνικό, το ρωμαϊκό, το κινέζικο, το βρετανικό και άλλα συστήματα. Για να αποφεύγεται η σύγχυση από τα πολλά, συχνά αντιφατικά και χωρίς αρκετή ακρίβεια συστήματα μονάδων μέτρησης από το 1960 έχει καθιερωθεί και ισχύει παγκοσμίως το σύστημα SI (Systeme Internationale), το οποίο περιλαμβάνει επτά θεμελιώδεις μονάδες. Όλα τα άλλα φυσικά μεγέθη θεωρούνται παράγωγα και κάθε μονάδα μέτρησης τέτοιου μεγέθους μπορεί πάντα να εκφραστεί ως συνάρτηση των θεμελιωδών μονάδων. Ωστόσο, διάφορα άλλα συστήματα μονάδων εξακολουθούν να βρίσκονται σε χρήση.

Η συσχέτιση των θεμελιωδών μεγεθών και των άλλων παράγωγων μεγεθών βάση του συστήματος SI δίδεται σχηματικά στην εικόνα 1.1.3-1, ώστε ο αναγνώστης να έχει μία οπτική αίσθηση της σχέσης μεταξύ τους:



Εικόνα 1.1.3-1. Θεμελιώδη και παράγωγα μεγέθη

1.1.4) ΣΥΣΤΗΜΑΤΑ ΜΕΤΡΗΣΕΩΝ

Η διάταξη σύμφωνα με την οποία μία ποσοτική έξοδος αντιστοιχίζεται στη μετρούμενη μεταβλητή, μέσω συλλογής των προς μέτρηση μεγεθών και κανόνων που διέπουν τη μεταξύ τους σχέση, ονομάζεται σύστημα μέτρησης. Ένα γενικό σύστημα μετρήσεων αποτελείται από τα εξής επιμέρους στοιχεία:

Α) Μετρητικά στοιχεία

Β) Σύστημα προσαρμογής

Γ) Σύστημα μετάδοσης

Δ) Σύστημα επεξεργασίας

Μερικά από τα βασικά χαρακτηριστικά που λαμβάνονται υπόψη είναι τα εξής:

Α) Ακρίβεια - Accuracy

Είναι η απόκλιση της τιμής που δίνει το όργανο μετρήσεων από την πραγματική τιμή του προς μέτρηση μεγέθους, η οποία δεν είναι γνωστή, μόνο κατ' εκτίμηση υπολογίζεται, και ορίζεται μέσω κάποιου προτύπου. Σχετίζεται με το μέγιστο εύρος σφαλμάτων στις ενδείξεις του οργάνου. Για να ελαχιστοποιηθεί το σφάλμα είναι απαραίτητο η κλίμακα του οργάνου να είναι κατά το δυνατόν πλησιέστερα στο εύρος του μεγέθους προς μέτρηση.

Β) Περιοχή μέτρησης – Span/Range

Το εύρος τιμών του φυσικού μεγέθους που μπορεί να μετρηθεί αποτελεί βασικό χαρακτηριστικό ενός μετρητικού συστήματος.

Γ) Ακρίβεια διασποράς, επαναληπτικότητα – Repeatability

Η απόκλιση ενδείξεων ενός οργάνου μεταξύ επαναλαμβανόμενων μετρήσεων, όταν στο όργανο εφαρμόζεται ακριβώς το ίδιο μέγεθος.

Δ) Διακριτική ικανότητα – Resolution

Η ελάχιστη μεταβολή του μετρούμενου φυσικού μεγέθους που μπορεί να ανιχνευθεί.

Ε) Ευαισθησία – Sensitivity

Ο λόγος μεταβολής της εξόδου προς την αντίστοιχη μεταβολή του μετρούμενου φυσικού μεγέθους είναι συνήθως μη σταθερός σε όλη την περιοχή μέτρησης.

ΣΤ) Υπερφόρτιση – Over-ranging

Ουσιαστικά αποτελεί την υπέρβαση του άνω ορίου της περιοχής μέτρησης του οργάνου. Διακρίνεται σε στατική και σε δυναμική και δίνεται ως ποσοστό της τιμής της μέγιστης τιμής.

Ζ) Ταχύτητα απόκρισης – Transient response

Πολύ σημαντικό χαρακτηριστικό αποτελεί η ικανότητα του οργάνου να αποκρίνεται σε ταχείες μεταβολές του μετρούμενου μεγέθους (δυναμική συμπεριφορά του οργάνου). Απαιτείται να είναι μεγαλύτερη όσο μεγαλύτερο είναι το εύρος ζώνης των συχνοτήτων.

Η) Ρύθμιση οργάνου – Calibration

Η διαδικασία αυτή λαμβάνει χώρα κατά την κατασκευή του οργάνου ή του συστήματος μέτρησης, αλλά και όποτε κρίνεται αναγκαία μια τέτοια διαδικασία.

Θ) Αβεβαιότητα - Uncertainty

Η αβεβαιότητα είναι ένα μέτρο της αξιοπιστίας των μετρήσεων. Εκφράζει κατά κάποιο τρόπο την αμφιβολία του κατά πόσον το αποτέλεσμα της μέτρησης είναι σωστό. Στη βιβλιογραφία αναφέρεται και ως σφάλμα, χωρίς να ταυτίζεται όμως με την κυριολεκτική έννοια του σφάλματος.

1.1.4.1) ΣΦΑΛΜΑΤΑ

Ένα σύστημα μέτρησης πρέπει να παράγει ιδανικά ένδειξη που να αντιστοιχεί στο μετρούμενο φυσικό μέγεθος. Αν όχι, αυτό σηματοδοτεί την εισαγωγή σφάλματος μέτρησης. Σφάλμα ονομάζεται η απόκλιση της μετρούμενης τιμής από την πραγματική. Τα σφάλματα μετριούνται με:

Α) Απόλυτο σφάλμα (absolute error)
Β) Σχετικό σφάλμα (relative error)
Γ) Ακρίβεια (precision)

Τα σφάλματα, βέβαια, διακρίνονται και σε επιμέρους κατηγορίες μερικές από τις οποίες είναι:

Α) Ανθρώπινα σφάλματα (λανθασμένη χρήση οργάνων, πρόωρη μέτρηση, παραλλαγή, μεταφορά μέτρησης)
Β) Σφάλματα οργάνων (λανθασμένη βαθμονόμηση, πόλωση, ολίσθηση)

Γ) Φυσικά σφάλματα (θόρυβος, εξωτερικές επιδράσεις)
Δ) Συστηματικά σφάλματα
Ε) Τυχαία σφάλματα.

Όπως είναι φυσικά αντιληπτό κανένα όργανο δεν είναι τέλειο. Όλα εισάγουν κάποιο σφάλμα, εξαιρετικά μικρό, ή μικρό, ή μεγάλο, στις μετρήσεις που κάνουν. Εκτός όμως από το όργανο, στη διαδικασία μέτρησης συμμετέχει και ο χρήστης/παρατηρητής και υπεισέρχονται κι άλλοι παράγοντες, που μπορεί να συντελέσουν σε αύξηση του σφάλματος. Σ΄αυτό το σημείο αξίζει να εμβαθύνουμε σε δύο είδη σφαλμάτων τα οποία έχουν κατά καιρούς αποτελέσει το επίκεντρο του ενδιαφέροντος των επιστημόνων: τα συστηματικά και τα τυχαία σφάλματα.

Τα σφάλματα που οφείλονται στο σύστημα μέτρησης που χρησιμοποιούμε, ή που οφείλονται σε παραμορφώσεις, ανακρίβειες ή μη ορθότητες που προκαλεί το ίδιο το όργανο, αποκαλούνται συστηματικά σφάλματα. Χαρακτηριστικό παράδειγμα είναι η μέτρηση τάσης σε ένα κύκλωμα την ώρα που αυτό λειτουργεί. Το βολτόμετρο, λόγω της εσωτερικής του αντίστασης, θα επηρεάσει τη λειτουργία τού κυκλώματος και η μέτρηση δεν θα αντανακλά την «αλήθεια». Το ίδιο θα συμβεί και σε μια μέτρηση ρεύματος με αμπερόμετρο που παρεμβάλλεται στον κλάδο. Παραμένουν σταθερά κατά τη λήψη επαναλαμβανόμενων μετρήσεων και σχετίζονται με την αξιοπιστία των μετρήσεων. Τείνουν να μετατοπίζουν τη μέση τιμή των μετρήσεων προς μία συγκεκριμένη διεύθυνση με συστηματικό τρόπο. Ο μόνος τρόπος να διορθωθούν είναι η σύγκριση του χρησιμοποιούμενου οργάνου με το αντίστοιχο πρότυπο. Σε πειράματα μεγάλης έκτασης εξαλείφονται κατά το δυνατόν με τη λήψη επαναλαμβανόμενων μετρήσεων με χρήση διαφορετικών μεθόδων. Στα συστηματικά σφάλματα συμπεριλαμβάνονται:

Α) Η κακή βαθμονόμηση (από κατασκευής, π.χ.) ή η κακή ή ελαττωματική διακρίβωση
Β) Ο κακός μηδενισμός (nulling) τού οργάνου.
Γ) Η παλαίωση του οργάνου.
Δ) Εξωτερικοί δυσμενείς παράγοντες, όπως πολύ υψηλές ή πολύ χαμηλές θερμοκρασίες, ηλεκτρικά ή μαγνητικά πεδία, μετεωρολογικά στοιχεία (π.χ. κεραυνοί), κλπ.

Είναι προφανές ότι κάποια συστηματικά σφάλματα δεν μπορούν να αποφευχθούν. Άλλα μπορούν να ελαχιστοποιηθούν με προσεκτική χρήση, συνεπή συντήρηση και κατάλληλες προφυλάξεις.

Υπάρχει και κατάλληλος «λογισμός» που αφορά (κυρίως τα συστηματικά) σφάλματα και μας δίνει επακριβώς το συνολικό σφάλμα σε συγκεκριμένες περιπτώσεις (βλ. αμέσως πιο κάτω). Σε άλλες περιπτώσεις και γενικότερα, ο «νόμος τού Murphy» πρέπει να ληφθεί σοβαρά υπ’ όψη και να θεωρούμε ότι πάντα τα σφάλματα συνδυάζονται με τον χειρότερο δυνατό τρόπο.

Ο λογισμός των σφαλμάτων ακολουθεί την ακόλουθη λογική:

Αν X= A ± B, τότε το σφάλμα στο X= ±[(σφάλμα στο A) + (σφάλμα στο B)]
Αν X= A·B, τότε το % σφάλμα στο X= ±[(% σφάλμα στο A) + (% σφάλμα στο B)]
Αν X= A/B, τότε το % σφάλμα στο X= ±[(% σφάλμα στο A) + (% σφάλμα στο B)]
Αν X= A^B, τότε το % σφάλμα στο X= ± B(% σφάλμα στο A)

Επομένως για ακρίβεια ο σωστός τρόπος αναγραφής της μετρούμενης τιμής είναι:
Τιμή ± Σφάλμα ± Συστηματικό σφάλμα.

Τα τυχαία σφάλματα οφείλονται, με τη σειρά τους, σε τυχαίους παράγοντες, ατυχήματα ή ατυχή συμβάντα. Μπορεί να είναι ανθρώπινα σφάλματα που προκαλούνται από κούραση ή άγχος. Επίσης, μπορεί να οφείλονται σε ξαφνικές αυξήσεις τής τάσης στο τροφοδοτικό, ή μια αλλαγή στη συχνότητα τροφοδοσίας στο δίκτυο και άλλα παρόμοια γεγονότα.

Για να τα αποφύγουμε (κυρίως σε κρίσιμες μετρήσεις), επαναλαμβάνουμε τη μέτρηση, ίσως και πολλές φορές, και επεξεργαζόμαστε στατιστικά τα δεδομένα που προκύπτουν. Διερευνούμε για τυχόν ύπαρξη ακραίων τιμών (outliers) και αποφασίζουμε πώς να τις χειριστούμε. Είναι πιθανό να απορρίψουμε ολόκληρο το σύνολο των μετρήσεων και να επαναλάβουμε σε περίπτωση που «δεν βγάζουμε άκρη».
Σκοπός της θεωρίας των σφαλμάτων είναι σε τελική ανάλυση η κατανόηση και η ελάττωση των αποκλίσεων και των σφαλμάτων στις μετρήσεις. Σε αυτό συμβάλλει σημαντικά η συλλογή περισσοτέρων δεδομένων και η επιλογή ακριβέστερων συστημάτων μέτρησης.

1.1.5) ΗΛΕΚΤΡΙΚΕΣ ΜΕΤΡΗΣΕΙΣ

Ο πίνακας 1.1.5-1 περιγράφει αναλυτικά ορισμένα χαρακτηριστικά παραδείγματα σχετικά με το περιεχόμενο της μετρολογίας και της οργανολογίας (Measurements and Instruments) στα πλαίσια της επιστήμης του Ηλεκτρολόγου Μηχανικού:

Πίνακας 1.2.5-1. Παραδείγματα συσχέτισης της Οργανολογίας και της Μετρολογίας και του Ηλεκτρολόγου Μηχανικού.

ΟΡΓΑΝΑ	ΜΕΤΡΗΣΗ	ΑΛΛΑ ΘΕΜΑΤΑ
Κλασικά Όργανα	Ρεύματος	Σφάλματα
Ηλεκτρονικά Όργανα	Τάσης	Αισθητήρες
Ψηφιακά Όργανα	Τάση/Ρεύμα Εναλλασσόμενου	Επεξεργασία Δεδομέων
Συστήματα Μετρήσεων	Ισχύος/Ενέργειας	Soft Measurements
	Αντίστασης	Αυτοματισμοί
	Χωρητικότητας/Επαγωγής	Συστήματα Ελέγχου
	Ημιαγωγών	
	Άλλων (π.χ. φορτίου)	

Οι ηλεκτρικές μετρήσεις αντιπροσωπεύουν ένα μεγάλο ποσοστό επιστημονικής και οικονομικής ή επιχειρηματικής δραστηριότητας του κλάδου. Το ακριβές ποσό του ετήσιου τζίρου είναι μάλλον αδύνατο να εκτιμηθεί με ακρίβεια (λόγω του εύρους), αλλά είναι τεράστιος. Αρκεί να δει κανείς τους καταλόγους προϊόντων από μερικούς μεγάλους κατασκευαστές (και τιμές) για να πάρει μια ιδέα. Ταυτόχρονα, η ανάπτυξη μεθόδων και συσκευών μέτρησης έχει να επιδείξει εκπληκτικά ιδιοφυείς και εντυπωσιακούς συνδυασμούς ιδεών.

1.1.6) ΣΥΣΚΕΥΕΣ ΣΥΓΧΡΟΝΙΣΜΟΥ ΚΑΙ ΜΕΤΡΗΣΕΩΝ

Αποτελώντας αναπόσπαστο κομμάτι της καθημερινότητας μας, κυρίως στην επαγγελματική, αλλά και στην προσωπική ενασχόλησή μας οι συσκευές συγχρονισμού και μετρήσεων βρίσκουν εφαρμογή σε ένα μεγάλο φάσμα περιπτώσεων, δικαιολογώντας κατά κάποιο τρόπο την ύπαρξη ποικίλων τέτοιων διατάξεων. Στόχος μας στο σημείο αυτό είναι η παρουσίαση μερικών συσκευών συγχρονισμού και μετρήσεων για την εξοικείωση και την ταύτιση αυτών με πτυχές από τη ζωή μας.

Εφαρμογές σε συσκευές μετρήσεων:

Α) Μέτρηση τάσης και ρεύματος
Β) Μετρήσεις σε περιστρεφόμενα συστήματα/αντικείμενα
Γ) Φορητές μετρητικές συσκευές

Εφαρμογές στην καθημερινή ζωή:

Α) Συσκευές ελέγχου και αίσθησης φωτισμού
Β) Συσκευές ελέγχου και αίσθησης θερμοκρασίας
Γ) Συσκευές ασφαλείας και ανίχνευσης φωτιάς
Δ) Συσκευές ελέγχου διαδικασιών

Εφαρμογές βιομηχανικού ελέγχου:

Α) Βιομηχανικές συσκευές οργάνων μέτρησης
Β) Συσκευές ελέγχου διαδικασιών

Επιπλέον κρίνεται σκόπιμη και η παραβολή μερικών ειδικότερων διατάξεων που θα βοηθήσουν στην αρτιότερη κατανόηση της άμεσης σχέσης μέτρησης και συγχρονισμού με τη

τεχνολογία των μικροεπεξεργαστών και μικροελεγκτών, δίνοντας μας έτσι την ευκαιρία μιας πρώτης εισαγωγής για το επόμενο κεφάλαιο.

Ο συγχρονισμός συστημάτων, συσκευών και γενικότερα η υλοποίηση και η εφαρμογή παράλληλων εργασιών για εξοικονόμηση είτε χρόνου είτε κόστους, χωρίς μείωση της απόδοσης και της πολυπλοκότητας, αποτελούσε πάντοτε έναν από τους βασικότερους παράγοντες που ωθούσαν την τεχνολογία στην περαιτέρω εξέλιξη και ανάπτυξη. Ταυτόχρονα βέβαια με την ανάγκη αυτή αυξήθηκε και η απαίτηση για περισσότερες, ταχύτερες και ακριβέστερες μετρήσεις που θα καθιστούσαν και αξιόπιστη την όλη αυτή διαδικασία. Συνεπώς η συνεχής πρόοδος και αναβάθμιση διατάξεων συγχρονισμού και μετρήσεων είναι επιτακτική. Οι διατάξεις συγχρονισμού και μετρήσεων είναι, όπως θα ήταν αναμενόμενο, ευρέως διαδεδομένες σε πολλούς τομείς της βιομηχανίας, της εκπαίδευσης, της έρευνας και γενικότερα της καθημερινότητας. Θέλοντας να αποδείξουμε τη σημασία και το ευρύπεδίο εφαρμογής τους θα παρουσιάσουμε συγκεκριμένες περιπτώσεις στις οποίες λόγω ιδιαιτερότητας και φύσης των απαιτήσεων η συνεχόμενη επεξεργασία ροής μετρήσεων και πληροφοριών αποτελεί υψίστης προτεραιότητας ζήτημα.

Α) Μέτρηση θερμοκρασίας (Temperature control):

Η χρησιμότητα των μετρήσεων σ΄ αυτό το τομέα αποτελεί το βασικό εργαλείο πάνω στο οποίο βασίζονται όλες οι προηγμένες εφαρμογές ελέγχου θερμοκρασίας και ειδικότερα:

- Στην ενίσχυση του χρόνου αντίδρασης μειώνοντας τις υπερβάσεις και τις ταλαντώσεις,
- Στην βελτιστοποίηση ελέγχου διαφορετικών θερμικών χαρακτηριστικών,
- Στην προστασία διαδικασιών και μηχανημάτων μέσω έγκαιρων ειδοποιήσεων,
- Στον συγχρονισμό και στην πρόβλεψη πιθανών μέγιστων επιβαρύνσεων σε πολυζωνικά συστήματα (multizone setups).

Όλα αυτά καθιστούν ευκολότερο το χειρισμό και την παρακολούθηση των συστημάτων με ταυτόχρονη δυνατότητα επέκτασης του ελέγχου.

Β) Έλεγχος θέσης φωτοβολταϊκών συστημάτων (Solar control):

Σ’ αυτή την περίπτωση μία διάταξη συγχρονισμού και μετρήσεων καθιστά γρηγορότερο το υπόβαθρο λειτουργίας του συστήματος, παρέχοντας τις απαραίτητες πληροφορίες στο χρήστη ώστε η προσαρμογή σε διάφορες συνθήκες να οδηγεί στην επίτευξη της μέγιστης απόδοσης της εγκατάστασης.
Παρακολουθείται η τάση, το ρεύμα, και η θερμοκρασία από κάθε ηλιακό πάνελ σε κάθε μία από τις σειρές. Κάθε πίνακας είναι τοποθετημένος σε μια θέση με γραμμικούς ενεργοποιητές

και αισθητήρες φωτός για να μετακινούν τον πίνακα προς τον ήλιο. Το σύστημα θα χρησιμοποιεί bluetooth και WiFi για να αλληλεπιδρά και να αποθηκεύει δεδομένα. Ένα από τα θέματα που πρέπει να αντιμετωπιστούν είναι ο έλεγχος των πολλών Arduino.

Γ) Έλεγχος κίνησης (Motion control):

Η κάλυψη ποικίλων ελέγχων τόσο σε μονοαξονικά όσο και σε πολυαξονικά συστήματα κινήσεως έχει αυξήσει κατά πολύ το βαθμό δυσκολίας παρακολούθησης της ορθής λειτουργίας των συγκεκριμένων πολύπλοκων συστημάτων. Συνεπώς η αξιολόγηση των σημαντικών πληροφοριών, οι οποιές παρέχονται σε μορφή μετρήσεων θα συνεισφέρει στην απλοποίηση και παράλληλα στην συνεχή αναβάθμισή τέτοιων συστημάτων.

Σε όλα τα παραπάνω παραδείγματα η χρήση μικροεπεξεργαστών και μικροελεγκτών συμβάλλει σε μεγάλο βαθμό στην μείωση της πολυπλοκότητας, του κατασκευαστικού και λειτουργικού κόστους και παράλληλα δίνει πολλές δυνατότητες αξιοποίησης.

1.2) ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ ΚΑΙ ΜΙΚΡΟΕΛΕΓΚΤΕΣ

1.2.1) ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΗΣ

1.2.1.1) ΓΕΝΙΚΕΣ ΠΛΗΡΟΦΟΡΙΕΣ

Ο μικροεπεξεργαστής, επίσης γνωστός ως η κεντρική μονάδα επεξεργασίας (CPU), είναι ο εγκέφαλος όλων των υπολογιστών και πολλών οικιακών και ηλεκτρονικών συσκευών. Πολλαπλοί μικροεπεξεργαστές, που εργάζονται μαζί, είναι οι «καρδιές» των κέντρων δεδομένων (datacenters), σούπερ-υπολογιστών, προϊόντων επικοινωνίας, και άλλων ψηφιακών συσκευών.

Ένα από τα σημαντικότερα προβλήματα που παρουσιάστηκαν καθώς οι υπολογιστές εξελίσσονταν, ήταν το πώς θα χρησιμοποιηθούν σε όσο το δυνατόν περισσότερες καθημερινές εφαρμογές. Η «ευφυΐα» και η ευκολία με την οποία κάνουν πράξεις οι υπολογιστές θα μπορούσε να χρησιμοποιηθεί για να γίνουν πολύ πιο λειτουργικές και αποδοτικές πολλές συσκευές (ηλεκτρικές συσκευές, συσκευές ελέγχου μηχανών, συστήματα σηματοδότησης) Φυσικά, ήταν αδύνατο να δεσμευτεί ένας μεγάλος ή έστω ένας μεσαίων δυνατοτήτων υπολογιστής σ’ αυτού του είδους τις εφαρμογές κυρίως λόγω του υπερβολικού τους μεγέθους και κόστους. Προέκυψε έτσι η απαίτηση για την κατασκευή ενός συστήματος που να περιέχει τα κυκλώματα όσο και τη βασική λογική έτσι ώστε να μπορεί να ανταπεξέλθει στο σύνολο των απαιτούμενων εφαρμογών. Συνεπώς, η λύση θα ήταν ένας ολοκληρωμένος αυτοδύναμος υπολογιστής μέσα σ’ ένα μοναδικό ολοκληρωμένο κύκλωμα, ο οποίος θα έπρεπε να είχε τη δυνατότητα αποθήκευσης προγραμμάτων και εκτέλεσης σύνθετων μαθηματικών πράξεων. Βέβαια, όπως συνήθως γίνεται, οι απαιτήσεις της βιομηχανίας και οι δυνατότητες των σχεδιαστών υπολογιστών ήταν αδύνατο να γεφυρωθούν. Φυσικά ήταν δυνατό να δοθεί λύση σε μερικές από τις απαιτήσεις, αλλά η κατασκευή ενός υπολογιστή πάνω σ’ ένα μοναδικό ολοκληρωμένο κύκλωμα φαινόταν σαν μία μακρινή επιδίωξη. Τη λύση φάνηκε να τη δίνει η τεχνολογία ολοκληρωμένων κυκλωμάτων, όταν μπόρεσε να τοποθετήσει χιλιάδες κυκλώματα πάνω σε μια πολύ μικρή επιφάνεια.

Το αποτέλεσμα της εμφάνισης της τεχνολογίας των ολοκληρωμένων κυκλωμάτων ήταν η ενσωμάτωση σε ένα μόνο ολοκληρωμένο κύκλωμα όλης της κεντρικής μονάδας επεξεργασίας, η οποία βέβαια θα έπρεπε να προγραμματίζεται για να περιέχει τις βασικότερες λειτουργίες ενός ψηφιακού υπολογιστή. Το κύκλωμα αυτό ονομάστηκε μικροεπεξεργαστής. Η μνήμη του βρίσκεται σε αρκετά ολοκληρωμένα κυκλώματα περιορισμένων αποθηκευτικών δυνατοτήτων, τα οποία το συνοδεύουν. Επίσης υποστηρίζεται και από μια πλειάδα α) ολοκληρωμένων κυκλωμάτων για να διασυνδέεται κατάλληλα και με τον εξωτερικό κόσμο μια και δεν έχει ενσωματωμένες αυτές τις δυνατότητες και β) ολοκληρωμένων κυκλωμάτων, που επιτελούν τις λειτουργίες χρονισμού και προώθησης δεδομένων στον τελικό τους προορισμό. Η ανάπτυξη της τεχνολογίας των ολοκληρωμένων κυκλωμάτων τις τελευταίες δεκαετίες έδωσε τη

δυνατότητα να μπορούν να ενσωματωθούν σε ένα ολοκληρωμένο κύκλωμα όλο και πιο πολύπλοκα κυκλώματα (από τον πρώτο μικροεπεξεργαστή, που είχε 2 χιλιάδες τρανζίστορ, έχουμε φτάσει πλέον σε επεξεργαστές με πάνω από 2 δισεκατομμύρια τρανζίστορ (ο εξαπύρηνος Intel Core i7-E περιέχει 2.600.000.000 τρανζίστορ) σε ένα και μόνο ολοκληρωμένο κύκλωμα) με αποτέλεσμα τη γρήγορη ανάπτυξη των μικροεπεξεργαστών και την ολοένα και πιο συχνή χρήση τους τόσο σε πολύπλοκες υπολογιστικές συσκευές όσο και σε απλές οικιακές συσκευές ή συστήματα ελέγχου.

Ο πρώτος μικροεπεξεργαστής ήταν ο Intel 4004, ο οποίος μας συστήθηκε το 1971. Ο 4004 δεν ήταν πολύ ισχυρός και κατά κύριο λόγο χρησιμοποιείτο για να εκτελέσει απλές μαθηματικές πράξεις σε έναν υπολογιστή που ονομαζόταν "Busicom." Ακριβώς όπως τα τηλέφωνα, οι συσκευές με μικροεπεξεργαστές έχουν γίνει αναπόσπαστο κομμάτι της καθημερινή μας ζωής, σε σημείο που δεν μπορούμε να φανταστούμε μια ζωή χωρίς αυτά. Είναι μερικές φορές δύσκολο να πιστέψει κανείς ότι μόλις 60 χρόνια πριν, οι υπολογιστές ήταν σπάνιοι και δεν ήταν διαθέσιμοι για το ευρύ κοινό. Δεν ήταν μέχρι τη δεκαετία του '80 που οι υπολογιστές εισήλθαν στα σπίτια μας και - χάρη στον μικροεπεξεργαστή - έκαναν πραγματικά σημαντικό αντίκτυπο στη ζωή του μέσου ατόμου.

Σήμερα, οι σύγχρονοι μικροεπεξεργαστές μπορούν να εκτελέσουν εξαιρετικά πολύπλοκες λειτουργίες σε τομείς όπως η μετεωρολογία, η αεροπορία, η πυρηνική φυσική και η μηχανική, ενώ παράλληλα καταλαμβάνουν πολύ λιγότερο χώρο, καθώς και παρέχουν ανώτερη απόδοση. Κατά τα τελευταία 40 χρόνια, οι μικροεπεξεργαστές έχουν γίνει πιο γρήγοροι και πιο ισχυροί, αλλά και συνάμα οικονομικότεροι. Η κατασκευή ενός CPU είναι μια ιδιαίτερα πολύπλοκη και απαιτητική διαδικασία που περιλαμβάνει πολλαπλά εκατοντάδες βήματα μέσα στους γνωστούς καθαρούς χώρους (cleanrooms).

1.2.1.2) ΔΟΜΗ ΚΑΙ ΧΡΗΣΙΜΟΤΗΤΑ

Ένας σύγχρονος μικροεπεξεργαστής δομείται από τις ακόλουθες μονάδες.

Α) Μονάδα αποκωδικοποίησης (*Decoding Unit*)

Β) Αριθμητική και Λογική Μονάδα (*Arithmetic and Logical Unit, ALU*): Η μονάδα στην οποία εκτελούνται μία προς μία οι αριθμητικές ή λογικές πράξεις, όπως υπαγορεύονται από τις εντολές που έχουν δοθεί στον υπολογιστή.

Γ) Καταχωρητές (*Registers*): Μικρά κελιά μνήμης στο εσωτερικό του επεξεργαστή, που χρησιμοποιούνται για την προσωρινή αποθήκευση των δεδομένων, καθώς αυτά υφίστανται επεξεργασία. Οι καταχωρητές διαφέρουν ανάλογα με τον τύπο του επεξεργαστή και τον κατασκευαστή, τόσο ως προς την οργάνωση όσο και ως προς τη χωρητικότητά τους.

Δ) Μονάδα ελέγχου (*Control Unit*): Ελέγχει τη ροή δεδομένων από και προς την ALU, τους καταχωρητές, τη μνήμη και τις περιφερειακές μονάδες εισόδου/εξόδου.

Ε) Μονάδα προσκόμισης (*Fetch Unit*): Μεταφέρει τις εντολές από τη μνήμη στον επεξεργαστή.

ΣΤ) Μονάδα προστασίας (*Protection Unit*): Εξασφαλίζει το αποδεκτό της κάθε διεργασίας που εκτελεί ο επεξεργαστής, ώστε να μη τροποποιούνται δεδομένα που δεν πρέπει ή να μην εκτελούνται μη αποδεκτές εντολές, όπως π.χ. διαίρεση αριθμού με το μηδέν.

Όσον αφορά τη χρησιμότητα ενός μικροπεξεργαστή βασικά στοιχεία τα οποία πρέπει να ληφθούν σοβαρά υπόψιν είναι:

Η απόδοση: Ο επεξεργαστής είναι ίσως ο πλέον καθοριστικός παράγοντας της απόδοσης του συστήματος. Ενώ και αρκετοί άλλοι παράγοντες έχουν σημαντική θέση στον καθορισμό της απόδοσης, οι ικανότητες του επεξεργαστή υπαγορεύουν τη μέγιστη απόδοση του συστήματος. Τα άλλα μέρη του υπολογιστή απλά επιτρέπουν στον επεξεργαστή να φτάσει στο μέγιστο των δυνατοτήτων του.

Η υποστήριξη λογισμικού: Καινούριοι, πιο γρήγοροι επεξεργαστές κάνουν ικανή τη χρήση καινούριου λογισμικού. Επιπρόσθετα νέοι επεξεργαστές όπως ο Pentium με τεχνολογία MMX, καθιστούν ικανή τη χρήση ειδικού λογισμικού που δεν μπορούσε να τρέξει σε προηγούμενους επεξεργαστές.

Η αξιοπιστία και η σταθερότητα: Η ποιότητα του επεξεργαστή είναι ένας παράγοντας που καθορίζει πόσο αξιόπιστο είναι το σύστημα. Κάποιοι επεξεργαστές είναι αξιόπιστοι, ενώ άλλοι όχι.

Η κατανάλωση ενέργειας και η ψύξη: Οι πρώτοι επεξεργαστές κατανάλωναν μικρή ποσότητα ενέργειας σε σχέση με τα άλλα μέρη του υπολογιστή. Οι σημερινοί υπολογιστές καταναλώνουν μεγάλα ποσά ενέργειας. Η κατανάλωση ενέργειας έχει καθοριστική σημασία σε όλα, από την επιλογή της μεθόδου ψύξης του επεξεργαστή, έως τη συνολική αξιοπιστία του υπολογιστή.

Και η υποστήριξη μητρικής πλακέτας: Ο επεξεργαστής που θα επιλέξουμε για το σύστημα μας είναι ένας βασικός καθοριστικός παράγοντας για το είδος της ομάδας των κεντρικών ολοκληρωμένων κυκλωμάτων (chipset) που θα χρησιμοποιήσουμε και συνεπώς για το τί είδους μητρική πλακέτα θα αγοράσουμε. Η μητρική πλακέτα με τη σειρά της, καθορίζει πολλές διαστάσεις των δυνατοτήτων και της απόδοσης του συστήματος. Όπως κάθε άλλη μονάδα ενός υπολογιστικού συστήματος, έτσι και ο επεξεργαστής βρίσκεται συνεχώς υπό εξέλιξη. Οι επεξεργαστές εδώ και χρόνια διπλασιάζουν την απόδοση τους κάθε 18 μήνες και απ' ότι δείχνουν τα πράγματα αυτός ο ρυθμός μάλλον δε θα σταματήσει. Η ιστορία των επεξεργαστών είναι στενά συνδεδεμένη με δύο εταιρίες, την IBM και κυρίως την Intel. Μέχρι σήμερα μπορούμε να αναγνωρίσουμε έξι γενιές επεξεργαστών. Η αρχή όμως της συμβατότητας χαρακτηρίζει όλες αυτές τις γενιές.

1.2.2) ΜΙΚΡΟΕΛΕΓΚΤΗΣ

1.2.2.1) ΔΙΑΦΟΡΟΠΟΙΗΣΗ ΑΠΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΕΣ – ΔΟΜΗ

Ο μικροελεγκτής (microcontroller) είναι ένας τύπος επεξεργαστή, ουσιαστικά μια παραλλαγή μικροεπεξεργαστή, ο οποίος μπορεί να λειτουργήσει με ελάχιστα εξωτερικά εξαρτήματα, λόγω των πολλών ενσωματωμένων υποσυστημάτων που διαθέτει. Χρησιμοποιείται ευρύτατα σε όλα τα ενσωματωμένα συστήματα (embedded systems) ελέγχου χαμηλού και μεσαίου κόστους, όπως αυτά που χρησιμοποιούνται σε αυτοματισμούς, ηλεκτρονικά καταναλωτικά προϊόντα (από ψηφιακές φωτογραφικές μηχανές έως παιχνίδια), ηλεκτρικές συσκευές και κάθε είδους αυτοκινούμενα τροχοφόρα οχήματα. Ο μικροελεγκτής είναι ένα αυτόνομο υπολογιστικό σύστημα, με πολύ μικρό μέγεθος, σε ένα και μοναδικό ολοκληρωμένο κύκλωμα (computer on a chip). Όπως και όλα τα VLSI κυκλώματα, αποτελείται από μέρη που κατασκευάζονται με διάφορες λιθογραφικές μεθόδους πάνω σε πλάκες πυριτίου, τα λεγόμενα Silicon Wafers. Πάνω σε αυτά σχηματίζονται χιλιάδες έως εκατομμύρια τρανζίστορ και κατ' επέκταση δημιουργούνται τα λεγόμενα ολοκληρωμένα κυκλώματα που είναι συνδυασμός λογικών πυλών. Συνδυάζοντας τις λογικές πύλες, δημιουργούνται υπομονάδες που επιτελούν ορισμένες πιο εξειδικευμένες λειτουργίες στον μικροελεγκτή. Είναι ουσιαστικά ένας τύπος επεξεργαστή, μια παραλλαγή μικροεπεξεργαστή, ο οποίος μπορεί να λειτουργήσει με ελάχιστα εξωτερικά εξαρτήματα, λόγω των πολλών ενσωματωμένων υποσυστημάτων που διαθέτει.

Στο σημείο αυτό κρίνεται σκόπιμο να γίνει ένας βασικός διαχωρισμός μεταξύ μικροεπεξεργαστών και μικροελεγκτών. Στους σύγχρονους μικροεπεξεργαστές για μη ενσωματωμένα συστήματα (πχ τους μικροεπεξεργαστές των προσωπικών υπολογιστών), δίνεται έμφαση στην υπολογιστική ισχύ. Η ευελιξία ανάπτυξης διαφορετικών εφαρμογών είναι μεγάλη, καθώς η λειτουργικότητα του τελικού συστήματος καθορίζεται από τα εξωτερικά περιφερειακά τα οποία διασυνδέονται με την κεντρική μονάδα (μικροεπεξεργαστή), η οποία δεν είναι εξειδικευμένη. Αντίθετα, στους μικροεπεξεργαστές για ενσωματωμένα συστήματα (μικροελεγκτές), οι οποίοι έχουν μικρότερες ή και μηδαμινές δυνατότητες συνεργασίας με εξωτερικά περιφερειακά, αυτού του είδους, η ευελιξία είναι περιορισμένη, καθώς και η υπολογιστική ισχύς. Οι μικροελεγκτές δίνουν έμφαση στο μικρό αριθμό ολοκληρωμένων κυκλωμάτων που απαιτείται για τη λειτουργία μιας συσκευής, το χαμηλό κόστος και την εξειδίκευση.

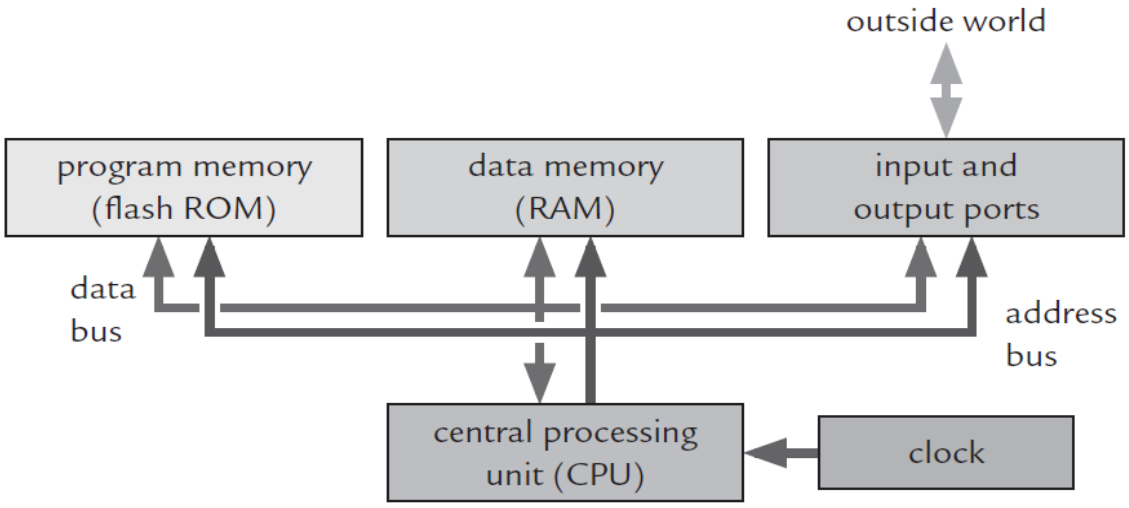
Αναλυτικότερα, τα πλεονεκτήματα των μικροελεγκτών είναι η αυτονομία, μέσω της ενσωμάτωσης σύνθετων περιφερειακών υποσυστημάτων όπως μνήμες και θύρες επικοινωνίας. Έτσι πολλοί μικροελεγκτές δεν χρειάζονται κανένα άλλο ολοκληρωμένο κύκλωμα για να λειτουργήσουν. Η ενσωμάτωση περιφερειακών σημαίνει ευκολότερη υλοποίηση εφαρμογών λόγω των απλούστερων διασυνδέσεων. Επίσης, οδηγεί σε χαμηλότερη κατανάλωση ισχύος, μεγιστοποιώντας τη φορητότητα και ελαχιστοποιεί το κόστος της συσκευής στην οποία ενσωματώνεται ο μικροελεγκτής. Το χαμηλό κόστος. Η μεγαλύτερη

αξιοπιστία, και πάλι λόγω των λιγότερων διασυνδέσεων. Οι μειωμένες εκπομπές ηλεκτρομαγνητικών παρεμβολών και μειωμένη ευαισθησία σε αντίστοιχες παρεμβολές από άλλες ηλεκτρικές και ηλεκτρονικές συσκευές. Το πλεονέκτημα αυτό προκύπτει από το μικρότερο αριθμό και μήκος εξωτερικών διασυνδέσεων καθώς και τις χαμηλότερες ταχύτητες λειτουργίας. Οι περισσότεροι διαθέσιμοι ακροδέκτες για ψηφιακές εισόδους-εξόδους (για δεδομένο μέγεθος ολοκληρωμένου κυκλώματος), λόγω της μη δέσμευσής τους για τη σύνδεση εξωτερικών περιφερειακών. Το μικρό μέγεθος συνολικού υπολογιστικού συστήματος. Τέλος η βασική αρχιτεκτονική των μικροελεγκτών δεν διαφέρει από αυτή των κοινών μικροεπεξεργαστών, αν και στους πρώτους απαντάται συχνά η αρχιτεκτονική μνήμης τύπου Harvard, η οποία χρησιμοποιεί διαφορετικές αρτηρίες σύνδεσης της μνήμης προγράμματος και της μνήμης δεδομένων (π.χ. οι σειρές AVR από την Atmel και PIC από την Microchip). Στους κοινούς μικροεπεξεργαστές συνηθίζεται η ενιαία διάταξη μνήμης τύπου φον Νόιμαν.

Ένας μικροελεγκτής είναι ένας ολόκληρος μικροϋπολογιστής σχεδιασμένος πάνω σε ένα και μόνο ολοκληρωμένο κύκλωμα υψηλής κλίμακας ολοκλήρωσης. Το ολοκληρωμένο αυτό κύκλωμα εμπεριέχει όχι μόνο τον επεξεργαστή αλλά και μνήμη RAM ROM, χρονιστές, πόρτες και άλλες κοινές περιφερειακές λειτουργικές μονάδες εισόδου εξόδου. Ο μικροελεγκτής όπως και ο μικροεπεξεργαστής, έχει σχεδιαστεί για να ανακαλεί δεδομένα, να κάνει πάνω σ' αυτά περιορισμένες πράξεις και με βάση αυτές τις πράξεις να ελέγχει το περιβάλλον του. Όπως κάθε υπολογιστικό κύκλωμα, περιέχει μονάδες εισόδου/εξόδου, μία κεντρική μονάδα επεξεργασίας, έναν αριθμό καταχωρητών και κυκλώματα μνήμης, εσωτερικούς χρονιστές – απαριθμητές, αριθμητική και λογική μονάδα, μνήμη προγράμματος (ROM ή EPROM). Κάθε μικροελεγκτής είναι ικανός να ανταλλάξει σήματα με το εξωτερικό περιβάλλον. Δέχεται σήματα εισόδου από διάφορους αισθητήρες, (θερμότητας πίεσης φωτός κ.τ.λ.). Εκτελεί πράξεις ανάμεσα σε μεταβλητές, καταχωρεί τιμές στη μνήμη RAM που διαθέτει και παράγει σήματα εξόδου που ελέγχουν άλλες συσκευές. Έτσι μπορεί να οδηγεί ηλεκτρονόμους, διόδους LED, κινητήρες και άλλα κατάλληλα κυκλώματα, που συνήθως περιλαμβάνονται σε κάθε μορφής αυτοματισμό. Με λίγα λόγια θα μπορούσαμε να πούμε ότι ένας μικροελεγκτής είναι ένα μικρό υπολογιστικό κύκλωμα σχεδιασμένο σε ένα ολοκληρωμένο κύκλωμα υψηλής κλίμακας ολοκλήρωσης. Έχει δυνατότητα να επικοινωνεί με το εξωτερικό περιβάλλον, να στέλνει σήματα διακοπών, να εκτελεί πράξεις ανάμεσα σε μεταβλητές χρησιμοποιώντας καταχωρητές ειδικού σκοπού. Κάθε μικροελεγκτής περιέχει τα παρακάτω στοιχεία:

Α) έναν αριθμό από καταχωρητές ειδικού σκοπού όπως: καταχωρητή εργασίας, συσσωρευτή, καταχωρητή κατάστασης, μετρητή προγράμματος, καταχωρητή εντολών, καταχωρητή δείκτη.
Β) εσωτερικούς χρονιστές – απαριθμητές.
Γ) αριθμητική και λογική μονάδα εντολών.
Δ) μονάδα αποκωδικοποιήσεις εντολών.
Ε) μνήμη προγράμματος (ROM ή EPROM).
ΣΤ) μνήμη καταχωρητών – μεταβλητών (RAM).

Ζ) κυκλώματα χρονισμού και ελέγχου.
Η) παράλληλες θύρες εισόδου – εξόδου.
Θ) άλλα περιφερειακά κυκλώματα (π.χ. QUART).



Σχήμα 1.2.2.1-1. Η δομή ενός μικροελεγκτή

Ανάλογα, βέβαια, με την εφαρμογή για την οποία προορίζεται ένας μικροελεγκτής, μπορεί να περιέχει και άλλα δομικά στοιχεία όπως:
Α) Μία ή περισσότερες ασύγχρονες σειριακές θύρες επικοινωνίας (Universal Asynchronous Receiver Transmitter, UART).
Β) Σύγχρονες σειριακές θύρες επικοινωνίας (πχ I2C, SPI, Ethernet).
Γ) Ολόκληρα υποσυστήματα για την άμεση υποστήριξη από υλικο λογισμικό (firmware) των πιο σύνθετων πρωτοκόλλων επικοινωνίας όπως CAN, HDLC, ISDN, ADSL.
Δ) Μονάδα άμεσης εκτέλεσης πράξεων κινητής υποδιαστολής (Floating Point Processing Unit, FPU), η οποία είναι πάντοτε πιο γρήγορη από την ALU του επεξεργαστή. Τέτοιες μονάδες χαρακτηρίζουν τους μικροελεγκτές με δυνατότητες ψηφιακής επεξεργασίας σήματος (Digital Signal Processing, DSP). Τα τελευταία χρόνια, με την ευρύτατη διάδοση των φορητών συσκευών ήχου και εικόνας, παρατηρείται μια τάση σύγκλισης των μικροελεγκτών με τους DSP.
Ε) Περισσότερες από μία εισόδους για μετατροπή αναλογικού σήματος σε ψηφιακό (Analog to Digital converter, ADC).
ΣΤ) Μετατροπέα ψηφιακού σε αναλογικό σήμα (Digital to Analog converter, DAC).
Ζ) Ελεγκτή οθόνης υγρών κρυστάλλων (Liquid Crystal Display, LCD).
Η) Υποσύστημα προγραμματισμού πάνω στο κύκλωμα (τύπου ISP, βλ. παραπάνω). Χάρη σε αυτό το κύκλωμα, είναι δυνατός ο επαναπρογραμματισμός (αναβάθμιση λογισμικού) της εφαρμογής, συνδέοντας στη συσκευή μια εξωτερική συσκευή προγραμματισμού (συνήθως σε θύρα UARTRS-232) ή ακόμη και από το διαδίκτυο. Αυτή η δυνατότητα απαιτεί την προϋπαρξη

λογισμικού υποδοχής (bootstrap) μέσα στη μνήμη προγράμματος και επομένως δεν μπορεί να γίνει σε τελείως άδεια μνήμη προγράμματος.
Θ) Υποσύστημα προγραμματισμού (τύπου ISP) και διάγνωσης (συνήθως είναι το καθιερωμένο πρότυπο JTAG). Χάρη σε αυτό, είναι δυνατός ο προγραμματισμός της μνήμης προγράμματος χωρίς να προαπαιτείται κάποιο πρόγραμμα υποδοχής. Γι αυτό το λόγο, το υποσύστημα προγραμματισμού είναι ιδιαίτερα χρήσιμο στον αρχικό προγραμματισμό, πχ κατά τη συναρμολόγηση, ή σε περίπτωση σφάλματος (bug) στο λογισμικό υποδοχής το οποίο να καθιστά αδύνατη την κανονική αναβάθμιση.

1.2.2.2) ΛΟΓΙΣΜΙΚΟ ΜΙΚΡΟΕΛΕΓΚΤΩΝ

Υπάρχουν ποικίλες γλώσσες που μπορούν να χρησιμοποιηθούν για τον προγραμματισμό ενός μικροελεγκτή:

- **Γλώσσα μηχανής (Machine code):** Τα δυαδικά δεδομένα που καταλαβαίνει στην πραγματικότητα ο επεξεργαστής. Κάθε εντολή έχει μια δυαδική τιμή που ονομάζεται «opcode». Δεν είναι αντιληπτή από τον άνθρωπο εκτός αν κάποιος έχει ξοδέψει πάρα πολύ χρόνο σε debugging χαμηλού επιπέδου. Κάποιοι από τους πρώτους υπολογιστές προγραμματίζονταν σε γλώσσα μηχανής, αλλά αυτή η εποχή έχει περάσει ανεπιστρεπτί.
- **Assembly:** Σε λίγο υψηλότερο επίπεδο από τη γλώσσα μηχανής και μεταφρασμένη στα Αγγλικά. Οι εντολές γράφονται σαν λέξεις που λέγονται “μνημονικά”(mnemonics) και ένα πρόγραμμα που ονομάζεται «assembler» μετατρέπει τα mnemonics σε γλώσσα μηχανής. Κάνει κάτι παραπάνω από απλή μετάφραση αλλά σίγουρα όχι πολύ παραπάνω σε σύγκριση με έναν compiler σε μια γλώσσα υψηλού επιπέδου.
- **C:** Η πιο συχνή επιλογή για μικροελεγκτές σήμερα. Ένας μεταγλωττιστής (compiler) μεταφράζει τη C σε γλώσσα μηχανής που μπορεί να αντιληφθεί η CPU. Αυτό μας δίνει όλα τα πλεονεκτήματα μιας γλώσσας υψηλού επιπέδου (δομές δεδομένων, συναρτήσεις κτλ.). Αρχικά το compilation έπρεπε πρώτα να μετατρέψει τηC σε Assembly και μετά από εκεί σε γλώσσα μηχανής αλλά πλέον έχουμε μετατροπή από C σε γλώσσα μηχανής απευθείας.
- **C++:** Μια αντικειμενοστραφής (object-oriented) γλώσσα η οποία χρησιμοποιείται κυρίως για μεγαλύτερες εφαρμογές. Κάποια χαρακτηριστικά της μπορούν να χρησιμοποιηθούν και για μικροελεγκτές αλλά κάποια άλλα δημιουργούν προβλήματα στην παραγωγή του κώδικα. Για αυτό το λόγο δημιουργήθηκε η embedded C++, που αποτελεί ένα κομμάτι της C++ που φτιάχτηκε για ενσωματωμένα (embedded) συστήματα.

1.2.2.3) ΤΡΕΧΟΝΤΕΣ ΜΙΚΡΟΕΛΕΓΚΤΕΣ ΚΑΙ ΑΥΞΗΜΕΝΕΣ ΔΥΝΑΤΟΤΗΤΕΣ ΤΟΥΣ

Η βιομηχανία μικροελεγκτών έχει καταλήξει στην παραγωγή πολύ ανταγωνιστικών μοντέλων μαζικής παραγωγής, καθώς και άλλων με εξειδικευμένες δυνατότητες λόγω του ισχυρού ανταγωνισμού και της τάσης ενσωμάτωσης των μικροελεγκτών σε κάθε ηλεκτρονική συσκευή. Συνεπώς υπάρχει μια κατηγοριοποίηση των διατάξεων αυτών. Ακολούθως παρουσιάζονται οι κυριότερες κατηγορίες τους:

Α) Μικροελεγκτές πολύ χαμηλού κόστους (συνήθως 8 bit), γενικότερης χρήσης με πολύ μεγάλο αριθμό ακροδεκτών. Σχεδιάζονται με έμφαση στη χαμηλή κατανάλωση ισχύος και την αυτάρκεια, ώστε να χρειάζονται ελάχιστα ή και καθόλου εξωτερικά εξαρτήματα και να μη μπορεί να αντιγραφεί εύκολα το εσωτερικό λογισμικό τους. Απουσιάζει η δυνατότητα επέκτασης της μνήμης τους. Μερικά μοντέλα είναι ευρέως γνωστά στους ερασιτέχνες ηλεκτρονικούς, όπως πχ οι περισσότεροι μικροελεγκτές των σειρών PIC (Microchip), AVR (Atmel) και 8051 (Intel, Atmel, Dallas κα).

Β) Μικροελεγκτές χαμηλού κόστους (συνήθως 8-bit), γενικής χρήσης, με μέτριο έως σχετικά μεγάλο αριθμό ακροδεκτών. Διαθέτουν μεγάλο αριθμό κοινών περιφερειακών, όπως θύρες UART, I2C, SPI ή CAN, μετατροπείς αναλογικού σε ψηφιακό και ψηφιακού σε αναλογικό. Στους κατασκευαστές της Άπω Ανατολής (Ιαπωνία, Κορέα), συνηθίζεται η ενσωμάτωση ελεγκτών οθόνης υγρών κρυστάλλων και πληκτρολογίου. Μερικές φορές παρέχουν δυνατότητα εξωτερικής επέκτασης της μνήμης τους.

Γ) Μικροελεγκτές (κυρίως 32-bit) μέσου κόστους, γενικής χρήσης, με μεγάλο αριθμό ακροδεκτών. Χαρακτηρίζονται από έμφαση στην ταχύτητα εκτέλεσης εντολών, υψηλή αυτάρκεια περιφερειακών και μεγάλες δυνατότητες εσωτερικής ή εξωτερικής μνήμης προγράμματος (FLASH) και RAM. Στο χώρο αυτό έχουν ισχυρή παρουσία οι αρχιτεκτονικές με υψηλή μεταφερσιμότητα λογισμικού (portability) από τον ένα στον άλλο κατασκευαστή. Πχ μεταξύ των μικροελεγκτών τύπου ARM ή MIPS, το σύνολο των βασικών εντολών που αναγνωρίζει η ALU είναι ακριβώς το ίδιο, μειώνοντας έτσι τις μεγάλες αλλαγές στο λογισμικό, όταν στο μέλλον ο πελάτης υιοθετήσει ένα μικροελεγκτή άλλου κατασκευαστή (αρκεί, φυσικά, να υποστηρίζει κι αυτός το σύνολο εντολών ARM ή MIPS, αντίστοιχα).

Δ) Μικροελεγκτές εξειδικευμένων εφαρμογών, οι οποίοι ενσωματώνουν συνήθως κάποιο εξειδικευμένο πρωτόκολλο επικοινωνίας το οποίο υλοποιείται πάντοτε σε επίπεδο υλικού (hardware). Τέτοιοι μικροελεγκτές χρησιμοποιούνται σε τηλεπικοινωνιακές συσκευές όπως τα μόντεμ.

Η μεγάλη μερίδα πωλήσεων των μικροελεγκτών εξακολουθεί να αφορά αυτούς των 8-bit, καθώς είναι η κατηγορία με το χαμηλότερο κόστος και το μικρότερο μέγεθος λογισμικού για 28 το ίδιο αποτέλεσμα, ιδίως επειδή οι σύγχρονες οικογένειες μικροελεγκτών 8-bit έχουν πολύ βελτιωμένες επιδόσεις σε σχέση με το παρελθόν.

1.2.3) ΔΥΝΑΤΟΤΗΤΕΣ ΚΑΙ ΠΕΡΙΟΡΙΣΜΟΙ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ ΚΑΙ ΜΙΚΡΟΕΛΕΓΚΤΩΝ

Στους σύγχρονους μικροεπεξεργαστές για μη ενσωματωμένα συστήματα (π.χ. τους μικροεπεξεργαστές των προσωπικών υπολογιστών), δίνεται έμφαση στην υπολογιστική ισχύ. Η ευελιξία ανάπτυξης διαφορετικών εφαρμογών είναι μεγάλη, καθώς η λειτουργικότητα του τελικού συστήματος καθορίζεται από τα εξωτερικά περιφερειακά τα οποία διασυνδέονται με την κεντρική μονάδα (μικροεπεξεργαστή), η οποία δεν είναι εξειδικευμένη. Αντίθετα, στους μικροεπεξεργαστές για ενσωματωμένα συστήματα (μικροελεγκτές), οι οποίοι έχουν μικρότερες ή και μηδαμινές δυνατότητες συνεργασίας με εξωτερικά περιφερειακά, αυτού του είδους, η ευελιξία είναι περιορισμένη, καθώς και η υπολογιστική ισχύς. Οι μικροελεγκτές δίνουν έμφαση στο μικρό αριθμό ολοκληρωμένων κυκλωμάτων που απαιτείται για τη λειτουργία μιας συσκευής, το χαμηλό κόστος και την εξειδίκευση. Αναλυτικότερα ορισμένα από τα βασικότερα πλεονεκτήματα αποτελούν τα ακόλουθα:

Α) Αυτονομία, μέσω της ενσωμάτωσης σύνθετων περιφερειακών υποσυστημάτων όπως μνήμες και θύρες επικοινωνίας. Έτσι πολλοί μικροελεγκτές δεν χρειάζονται κανένα άλλο ολοκληρωμένο κύκλωμα για να λειτουργήσουν.
Β) Η ενσωμάτωση περιφερειακών σημαίνει ευκολότερη υλοποίηση εφαρμογών λόγω των απλούστερων διασυνδέσεων. Επίσης, οδηγεί σε χαμηλότερη κατανάλωση ισχύος, μεγιστοποιώντας τη φορητότητα και ελαχιστοποιεί το κόστος της συσκευής στην οποία ενσωματώνεται ο μικροελεγκτής.
Γ) Χαμηλό κόστος.
Δ) Μεγαλύτερη αξιοπιστία, και πάλι λόγω των λιγότερων διασυνδέσεων.
Ε) Μειωμένες εκπομπές ηλεκτρομαγνητικών παρεμβολών και μειωμένη ευαισθησία σε αντίστοιχες παρεμβολές από άλλες ηλεκτρικές και ηλεκτρονικές συσκευές. Το πλεονέκτημα αυτό προκύπτει από το μικρότερο αριθμό και μήκος εξωτερικών διασυνδέσεων καθώς και τις χαμηλότερες ταχύτητες λειτουργίας.
ΣΤ) Περισσότεροι διαθέσιμοι ακροδέκτες για ψηφιακές εισόδους-εξόδους (για δεδομένο μέγεθος ολοκληρωμένου κυκλώματος), λόγω της μη δέσμευσής τους για τη σύνδεση εξωτερικών περιφερειακών.
Ζ)Μικρό μέγεθος συνολικού υπολογιστικού συστήματος.

1.3) ΤΟ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΙΚΟ ΣΥΣΤΗΜΑ ARDUINO

1.3.1) ΓΕΝΙΚΑ ΣΤΟΙΧΕΙΑ

Το ARDUINO είναι ένας μικροελεγκτής σε μία ενιαία πλακέτα (single-board), δηλαδή μια απλή μητρική πλακέτα ανοικτού κώδικα, με ενσωματωμένο μικροελεγκτή και εισόδους/εξόδους, και η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring (ουσιαστικά πρόκειται για τη γλώσσα προγραμματισμού C++ και ένα σύνολο από βιβλιοθήκες, υλοποιημένες επίσης στην C++). Το Arduino μπορεί να χρησιμοποιηθεί για την ανάπτυξη ανεξάρτητων διαδραστικών αντικειμένων αλλά και να συνδεθεί με υπολογιστή μέσω προγραμμάτων σε Processing, Max/MSP, PureData, SuperCollider. Το Arduino είναι μια open source hardware – software πλατφόρμα ιδανική για τον σχεδιασμό και την ανάπτυξη διαδραστικών (interactive) εφαρμογών βασισμένο στη χρήση απλού λογισμικού.

Οι πλακέτες Arduino είναι σε θέση να διαβάζουν ποικίλες μορφές εισόδων (inputs) και να τις μετατρέπουν αντίστοιχα σε πλήθος διαφορετικών εξόδων (π.χ. ενεργοποίηση ενός LED). Ο χρήστης μπορεί να κατευθύνει τη λειτουργία της πλακέτας στέλνοντας οδηγίες στον μικροελεγκτή που βρίσκεται τοποθετημένος πάνω στην πλακέτα αυτή. Η επίτευξη αυτής της επικοινωνίας βασίζεται στη γλώσσα προγραμματισμού Arduino (Arduino programming language) και στο λογισμικό Arduino (the Arduino software-IDE). Οι μικροελεγκτές Arduino έχουν αποτελέσει την καρδιά και τον εγκέφαλο πολλών εφαρμογών (project), από καθημερινές εφαρμογές έως πολύπλοκα επιστημονικά όργανα.

Ιστορικά τα Arduino γεννήθηκαν το 2005 στο Ivrea Interaction Design Institute της Ιταλίας ως ένα εύκολο και γρήγορο εργαλείο για ανάπτυξη εφαρμογών (prototyping), το οποίο στόχευε κυρίως φοιτητές χωρίς υψηλού επιπέδου υπόβαθρου γνώσεων στην ηλεκτρονική ή στον προγραμματισμό (η βασική ομάδα του Arduino αποτελείτο από τους Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino, και ton David Mellis). Ως αποτέλεσμα γρήγορα μία παγκόσμια ‘κοινότητα’ κατασκευαστών (φοιτητές, επαγγελματίες, ερασιτέχνες, καλλιτέχνες, προγραμματιστές) έδειξε ενδιαφέρον στην αξιοποίηση αυτής της open source πλατφόρμας, συνεισφέροντας όλοι ως ένα βαθμό στην ραγδαία εξάπλωση και αναγνώριση των πλακετών Arduino στο χώρο της τεχνολογίας. Έχοντας φτάσει σε αυτό το επίπεδο η αυξανόμενη ανάγκη για νέες προκλήσεις και η δυνατότητα μεγαλύτερης εμβάθυνσης οδήγησε αναπόφευκτα την εταιρεία στην κατασκευή καινούριων πλακετών που θα ανταποκρίνονται στις νέες απαιτήσεις. Πλέον τα Arduino καλύπτουν ένα ευρύ φάσμα αναγκών, από απλές 8-bit πλακέτες μέχρι προϊόντα για IoT εφαρμογές, 3D-printing και ενσωματωμένα περιβάλλοντα για προγραμματισμό.

Σε αυτό το σημείο κάλλιστα θα μπορούσε κάποιος να αναρωτηθεί για το λόγο χρήσης ενός Arduino, εφόσον υπάρχουν στην αγορά πολλά αντίστοιχα προϊόντα που δύναται να καλύψουν τις ίδιες ανάγκες.

Όμως η χρήση του Arduino φαίνεται κατάλληλη για τους ακόλουθους λόγους:

● **Χαμηλό κόστος**: Οι πλακέτες Arduino είναι σχετικά οικονομικότερες σε σύγκριση με άλλες πλατφόρμες μικροελεγκτών.
● **Cross-platform**: Το λογισμικό Arduino (The Arduino Software-IDE) είναι κατάλληλο για Windows, Macintosh OSX και Linux σε αντίθεση με άλλους μικροελεγκτές που περιορίζονται μόνο σε περιβάλλον Windows.
● **Απλό προγραμματιστικό περιβάλλον**: Το περιβάλλον προγραμματισμού Arduino (IDE) χαρακτηρίζεται για την απλότητα λειτουργίας του από αρχάριους, ωστόσο είναι ταυτόχρονα και αρκετά ευέλικτο για προχωρημένου επιπέδου προγραμματιστές.
● **Opensource λογισμικό**: Το λογισμικό Arduino διατίθεται στη μορφή ανοικτού κώδικα, που μπορεί να επεκταθεί μέσω C++ βιβλιοθηκών. Η κατανόηση των τεχνικών λεπτομερειών είναι εύκολα προσβάσιμη σε γλώσσα AVR C.

Συνεπώς οι πλακέτες Arduino χάρη στην απλή και προσιτή εμπειρία που προσφέρουν κατά την χρήση τους έχουν χρησιμοποιηθεί σε χιλιάδες διαφορετικά σχέδια και εφαρμογές. Εκπαιδευτικοί και μαθητές μπορούν να το χρησιμοποιήσουν για την κατασκευή χαμηλού κόστους επιστημονικών οργάνων, για την απόδειξη αρχών χημείας και φυσικής, ή για την αφετηρία ενασχόλησης με τον προγραμματισμό και τη ρομποτική, σχεδιαστές και αρχιτέκτονες για να χτίσουν διαδραστικά πρωτότυπα, μουσικοί και καλλιτέχνες για να πειραματιστούν με νέα μουσικά όργανα. Με άλλα λόγια τα Arduino αποτελούν ένα βασικό αλλά συνάμα και απλό εργαλείο εκμάθησης νέων πραγμάτων.

1.3.2) SOFTWARE – ΠΕΡΙΒΑΛΛΟΝ ΑΝΑΠΤΥΞΗΣ ΛΟΓΙΣΜΙΚΟΥ

Ένα από τα βασικά στοιχεία την πλατφόρμας Arduino είναι το ολοκληρωμένο περιβάλλον ανάπτυξης του (IDE), μία εφαρμογή γραμμένη σε γλώσσα Java, και το οποίο περιέχει το απαραίτητο και αναγκαίο λογισμικό ώστε να μπορεί ένας υπολογιστής να επικοινωνήσει με το σύστημα. Περιέχει ένα πρόγραμμα επεξεργασίας κειμένου για τη σύνταξη κώδικα, μια περιοχή του μηνύματος, μια κονσόλα κειμένου, μια γραμμή εργαλείων με κουμπιά για κοινές λειτουργίες και μια σειρά από μενού. Συνδέεται με το υλικό (hardware) μέρος του Arduino για να φορτώσει προγράμματα και να επικοινωνεί μαζί τους.
Ένα πρόγραμμα ή κώδικας που γράφτηκε για Arduino ονομάζεται σκίτσο (sketch). Υπάρχουν δυο ειδικές συναρτήσεις που είναι μέρος του κάθε sketch του Arduino οι οποίες είναι η setup() και η loop(). Η setup() καλείται μια φορά, όταν το sketch ξεκινά ή όποτε κάνει επαναφορά (reset) η πλατφόρμα Arduino. Κυρίως, σε αυτήν γίνονται οι αρχικοποιήσεις των μεταβλητών, η ρύθμιση της κατάστασης των ακίδων (pins) και η προετοιμασία των βιβλιοθηκών. Αντιθέτως, η συνάρτηση loop() καλείται ξανά και ξανά επιτρέποντας έτσι στο πρόγραμμα να ανταποκριθεί σε εξωτερικά ερεθίσματα. Και οι δυο συναρτήσεις πρέπει να περιλαμβάνονται στο sketch, ακόμα και αν δεν περιέχουν κάτι και να είναι κενές. Υπάρχουν δύο τρόποι εγγραφής κώδικα στο Arduino IDE:

● Μέσω της γλώσσας του Arduino, μία γλώσσα βασισμένης στη Processing, που υποστηρίζει όλες τις βιβλιοθήκες που έχουν διαμορφωθεί από την κοινότητα του Arduino, δίνοντας τη δυνατότητα ακόμα και σε έναν άπειρο χρήστη να ελέγξει περίπλοκες συσκευές και λειτουργίες.
● Μέσω της γλώσσας C χαμηλού επιπέδου, δίνοντας τη δυνατότητα πιο γρήγορης εκτέλεσης του προγράμματος, αλλά και απευθείας χειρισμού των καταχωρητών του επεξεργαστή, ακόμα και σε λειτουργίες που δεν υποστηρίζει το Arduino IDE.

Το Arduino IDE παρέχει:

• Ένα πρακτικό περιβάλλον για τη συγγραφή των προγραμμάτων, με συντακτική χρωματική σήμανση.
• Μερικές έτοιμες βιβλιοθήκες για προέκταση της.
• Τον compiler για τη μεταγλώττιση των sketch. (Για compiler χρησιμοποιείται ο AVRgcc και ως βασική βιβλιοθήκη C χρησιμοποιείται η AVRlibc. Λόγω της καταγωγής της από τη C , στη γλώσσα του Arduino, μπορούν να χρησιμοποιηθούν ουσιαστικά οι ίδιες βασικές εντολές και συναρτήσεις, με την ίδια σύνταξη, τους ίδιους τύπων δεδομένων και τους ίδιους τελεστές όπως και στη C. Πέρα από αυτές όμως, υπάρχουν κάποιες ειδικές εντολές, συναρτήσεις και σταθερές που βοηθούν για τη διαχείριση του ειδικού hardware του Arduino.)
• Μία σειριακή οθόνη (serial monitor) που παρακολουθεί τις επικοινωνίες της σειριακής (USB), αναλαμβάνει να στείλει αλφαριθμητικά στο Arduino μέσω αυτής και είναι ιδιαίτερα χρήσιμο για την αποσφαλμάτωση των sketch.
• Την επιλογή για ανέβασμα των μεταγλωττισμένων sketch στο Arduino.

Στην εικόνα 1.3.2-1 παρουσιάζεται το περιβάλλον ανάπτυξης.

Σχήμα 1.3.2-1. Περιβάλλον Arduino IDE.

Τα πλήκτρα που φαίνονται στην προηγούμενη εικόνα εκτελούν τις ακόλουθες διαδικασίες:

 Ελέγχει εάν υπάρχουν σφάλματα στον κώδικα. (Verify)

 Περνάει το πρόγραμμα στο Arduino. (Upload)

 Ανοίγει ένα νέο κενό project. (New)

 Άνοιγμα αρχείου. (Open)

 Αποθήκευση αρχείου. (Save)

 Άνοιγμα σειριακής οθόνης. (Serial Monitor)

Οι επιπλέον εντολές βρίσκονται εντός των πέντε μενού:**File**, **Edit**, **Sketch**, **Tools**, **Help.**

Η διαδικασία ανεβάσματος ενός sketch στο Arduino board συνοψίζεται στα ακόλουθα χαρακτηριστικά βήματα: Ο κώδικας μετατρέπεται σε C χαμηλού επιπέδου. Στη συνέχεια μεταγλωττίζεται μέσω του avr-gcc. Ο παραγόμενος δυαδικός κώδικας αποθηκεύεται στη μνήμη του επεξεργαστή απ'όπου και εκτελείται μέσω της σύνδεσης USB (η σύνδεση γίνεται μέσω avrdude, πρόγραμμα που υλοποιεί το πρωτόκολλο επικοινωνίας για την μεταφορά και αποθήκευση του κώδικα στη μνήμη του Arduino).

1.3.3) Ο ΜΙΚΡΟΕΛΕΓΚΤΗΣ ARDUINO UNO

1.3.3.1) ΓΕΝΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ

Το ARDUINO UNO είναι μία πλακέτα μικροελεγκτή βασισμένη στον επεξεργαστή ATmega 328P της Atmel, έναν 8-bit AVR RISC-based 8MHz μικροεπεξεργαστή ο οποίος διαθέτει 32KB ISP Flash μνήμη με δυνατότητα ανάγνωσης-εγγραφής, 1KB EEPROM, 2KB SRAM, 23 E/E γενικού σκοπού, 32 καταχωρητές γενικού σκοπού, έναν 16-bit και δύο 8-bit εύλικτους μετρητές (timers/counters), εσωτερικές και εξωτερικές διακοπές (interrupts), σειριακή προγραμματιζόμενη USART, SPI σειριακή θύρα, 10-bit μετατροπέα αναλογικού σε ψηφιακό σήματος (A/D converter) 6 καναλιών με δυνατότητες μέχρι 200KHz, προγραμματιζόμενο watchdog timer με εσωτερικό κρύσταλλο και 5 λειτουργίες εξοικονόμησης ενέργειας (powersaving). Έχει επίσης εξωτερικό 16 MHz κρύσταλλο, με δυνατότητες επέκτασης στα 20 MHz. Ο κρύσταλλος αυτός συγχρονίζει τη λειτουργία του Arduino, αφού ενεργεί ως πηγή ρολογιού, στέλνοντας σήματα ON-OFF τα οποία αλλάζουν την κατάσταση του συστήματος. Διαθέτει 14 Digital Input / Output ακροδέκτες (6 από τους οποίους χρησιμοποιούνται σαν PWM έξοδοι), 6 ακροδέκτες Analog Inputs που μπορούν επίσης να χρησιμοποιηθούν σαν Digital Inputs, ταλαντωτή 16MHz και USB θύρα μέσω της οποίας συνδέεται με τον Ηλεκτρονικό υπολογιστή (PC). Ο προγραμματισμός του μικροεπεξεργαστή γίνεται μέσω του ειδικού κατάλληλου περιβάλλοντος ανάπτυξης. Διαθέτει τέσσερεις λυχνίες ένδειξης κατάστασης λειτουργίας (Status LEDs). Το πρώτο είναι συνδεδεμένο στο pin 13, σε συνδυασμό με μία on-board αντίσταση του 1ΚΩ και χρησιμοποιείται για έλεγχο της σωστής λειτουργίας του

συστήματος. Serial TX και Rx LEDs τα οποία υποδεικνύουν τη σωστή επικοινωνία με το PC ή κάποια άλλη σειριακή συσκευή (μέσα από τα digital pins 0 και 1). PowerLED, το οποίο γίνεται ON όταν υπάρχει τροφοδοσία στο Arduino. Ο μικροεπεξεργαστής έχει δυνατότητα μόνο σειριακής επικοινωνίας. Το FTDI chip μετατρέπει τα USB σήματα σε σειριακά και αντίστροφα. Επιπλέον έχει έναν on-chip ρυθμιστή (regulator) στα 3.3V, δίνοντας τη δυνατότητα διασύνδεσης με συσκευές που λειτουργούν στα 3.3V όπως το Xbee για ασύρματη επικοινωνία. Τέλος η πλατφόρμα διαθέτει διάφορα στοιχεία όπως πυκνωτές, διόδους και προσαρμοστείς τάσης, για τη σταθεροποίηση ή και υποβιβασμό της τάσης τροφοδοσίας και προστασίας από βραχυκυκλώματα.

1.4.3.2) ΜΕΡΗ ΜΙΚΡΟΕΛΕΓΚΤΗ ARDUINO UNO

Το Arduino Uno έχει 14 ψηφιακούς ακροδέκτες Εισόδου/Εξόδου οι οποίοι μπορούν να τεθούν ως είσοδοι ή ως έξοδοι με εντολές-συναρτήσεις όπως οι pinMode(), digitalWrite(), και digitalRead(). Λειτουργούν στα 5 Volts και έχουν την δυνατότητα να λειτουργούν για ένταση της τάξεως των 40mA. Σε κάθε pin υπάρχει εσωτερικά ένας Pull-up αντιστάτης στα 20-50ΚΩ. Επιπλέον, έχει 6 αναλογικούς ακροδέκτες Εισόδου. Αυτοί μπορούν να διαβάσουν αναλογικές τιμές όπως η τάση μιας μπαταρίας κτλ και να τις μετατρέψουν σε έναν αριθμό από 0-1023. Η μέτρηση της τάσης γίνεται για προκαθορισμένα επίπεδα τάσης τα οποία κυμαίνονται από 0 έως 5 V. Εκτός αυτού, 6 εκ των 14 ψηφιακών ακροδεκτών οι P3, P5, P6, P9, P10 και P11 έχουν την δυνατότητα να προγραμματιστούν ώστε να λειτουργούν ως αναλογικές έξοδοι (προσφέρουν έξοδο με δυνατότητα διαμόρφωσης του πλάτους του πλαμού - Pulse Width Modulation - PWM).

Κάθε μία από τις 14 ψηφιακές ακίδες και τις 6 αναλογικές μπορεί να χρησιμοποιηθεί ως ψηφιακή είσοδος/έξοδος λογικής 5V. Κάθε ακίδα μπορεί να λειτουργεί για μέγιστο ρεύμα 40mA (προτεινόμενο 20mA) και διαθέτει εσωτερική pull-up αντίσταση (αποσυνδεδεμένη εξ ορισμού) που ενεργοποιείται από το λογισμικό. Οι περισσότερες ακίδες έχουν και επιπλέον εξειδικευμένες λειτουργίες. Ειδικότερα έχουμε :

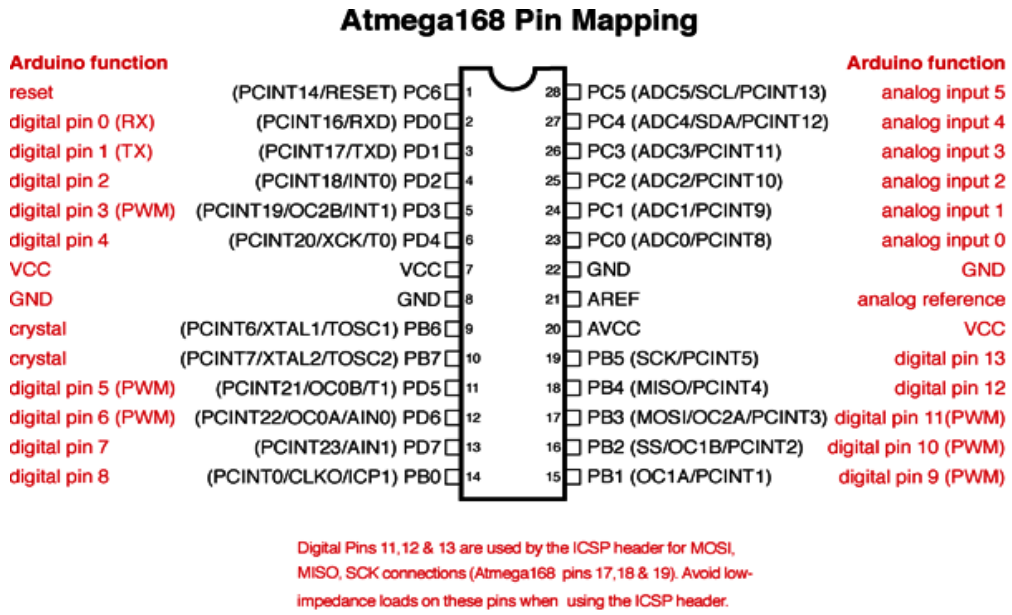
Σειριακές: 0 (Rx) και 1 (Tx). Χρησιμοποιούνται για την παραλαβή και αποστολή σειριακών δεδομένων αντίστοιχα. Αποτελούν απαραίτητη συνθήκη για τη σύνδεση του Arduino με τη USB θύρα, αφού συνδυάζονται με τις αντίστοιχες ακίδες του FTDIUSB-to-Serial chip, αλλά και για ασύρματη επικοινωνία (RF, Xbee, Bluetooth,κτλ).

Εξωτερικά interrupts: Ακίδες 2 και 3. Αφού καθοριστούν ως ψηφιακές είσοδοι, μπορούν να προκαλέσουν εξωτερικά interrupts, ανάλογα με την τιμή τους. (LOW, HIGH, Μετάβαση από LOW σε HIGH ή από HIGH σε LOW).

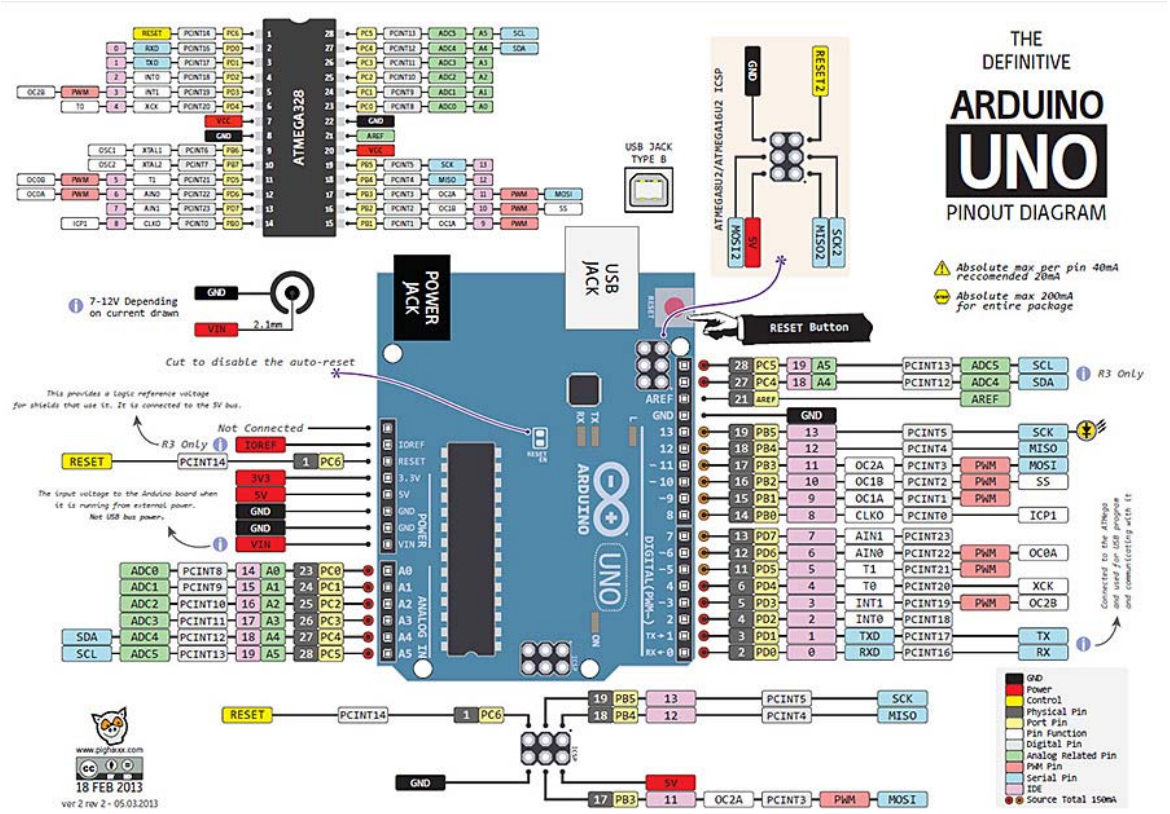
InputCapture: Ακίδα 8. Η λειτουργία inputcapture χρησιμοποιείται κυρίαρχα για τον υπολογισμό της συχνότητας ενός σήματος.

PWM: Ακίδες 3, 5, 6, 9, 10 και 11. Παρέχουν οκτάμπιτη (256 τιμές) παραγωγή PWM(pulse width modulation). Μια τεχνική που ακολουθείται για τη παραγωγή αναλογικών σημάτων με ψηφιακά μέσα.
SPI: Ακίδες 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK). Οι ακίδες αυτές υποστηρίζουν SPI επικοινωνία μεταξύ του Arduino και εξωτερικών σειριακών περιφερειακών μονάδων , όπως εξωτερικές μνήμες και Shift Registers.
Αναλογικές Είσοδοι: Ακίδες A0, A1, A2, A3, A4, A5. Καθεμία από τις οποίες παρέχει 10-bit ανάλυσης Analog to Digital μετατροπή με εξ ορισμού τάση αναφοράς τα 5V. η τάση αναφοράς μπορεί να αλλάξει, χρησιμοποιώντας την **ακίδα AREF**.
I2C: Ακίδες A4 (SDA) και A5 (SCL). Υποστηρίζει I2C (TWI) επικοινωνία, χρησιμοποιώντας την βιβλιοθήκη Wire**.**
Reset: Όταν τίθεται σε τιμή LOW, κάνει επανεκκίνηση (reset) τον μικροεπεξεργαστή.
ICSP: In-circuit Serial Programmer. Δίνει τη δυνατότητα άμεσου ανεβάσματος ενός sketch στο Arduino με χρήση εξωτερικού προγραμματισμού και του κατάλληλου λογισμικού (AVR Dude), χωρίς την ύπαρξη bootloader, κερδίζοντας έτσι χώρο μνήμης και χρόνο ανεβάσματος ενός sketch στην πλατφόρμα. Επιπλέον παρέχει τη δυνατότητα «καψίματος» ενός νέου bootloader στον μικροεπεξεργαστή.

Όπως αναφέρθηκε προηγουμένως “καρδιά” του Arduino Uno είναι ο επεξεργαστής Atmega328. Στα σχήματα 1.3.4.2-1 και 1.3.4.2-2 δίδονται οι αντιστοιχίες των ακροδεκτών του επεξεργαστή και η τελική διαμόρφωση του μικροελεγκτή.



Σχήμα 1.3.3.2-1. Αντιστοιχία θυρών μικροεπεξεργαστή - μικροελεγκτή.



1.3.3.2-2. Λεπτομερές διάγραμμα ακροδεκτών Arduino Uno

1.3.3.3) ΤΕΧΝΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ

Το Arduino Uno τροφοδοτείται είτε από εξωτερική τροφοδοσία που παρέχεται είτε μέσω μιας υποδοχής των 2.1mm (θετικός πόλος στο κέντρο) που βρίσκεται στην κάτω αριστερή γωνία του Arduino είτε απευθείας από την θύρα USB του υπολογιστή. Η επιλογή της πηγής γίνεται αυτόματα από το αναπτυξιακό. Ως εξωτερική τροφοδοσία ορίζεται είτε μια μπαταρία, είτε μετασχηματιστής των 9Volt από 220V. Η μπαταρία μπορεί να συνδεθεί στις υποδοχές του Arduino Vin και GND όπου τοποθετούνται ο θετικός πόλος και ο αρνητικός αντίστοιχα. Από την άλλη αν τροφοδοτηθεί με μετασχηματιστή απλά πρέπει να τοποθετηθεί το βύσμα στην υποδοχή που υπάρχει θετικό πόλο στο κέντρο. Η πλακέτα μπορεί να λειτουργήσει με εξωτερική πηγή από 0 έως 20 Volts. Αν ωστόσο τροφοδοτηθεί με λιγότερα από 7 Volt τα pin εξόδου 5Volt δεν θα καταφέρουν να εξάγουν τάση 5 Volts. Αντίθετα, αν δώσουμε πάνω από 12 Volts θα υπερθερμανθεί ο σταθεροποιητής τάσης στην πλακέτα και ενδεχομένως να καταστραφεί. Συνεπώς, μια ιδανική τάση είναι τα 9 Volts. Οι ακροδέκτες τροφοδοσίας είναι οι εξής:

VIN: Ακροδέκτης για μη σταθεροποιημένη τάση. Συνήθως εδώ συνδέεται μια εξωτερική πηγή τροφοδοσίας.

5V: Ακροδέκτης σταθεροποιημένης τάσης 5 Volt. Η ρυθμιζόμενη παροχή ηλεκτρικού ρεύματος που χρησιμοποιείται για την τροφοδοσία του μικροελεγκτή ή άλλων ηλεκτρονικών στοιχείων της πλακέτας. Αυτό μπορεί να προέρχεται είτε από Vin με ενσωματωμένο ρυθμιστή, ή να παρέχεται από USB ή άλλη ρυθμιζόμενη παροχή 5 V

3V3: Μέγιστη κατανάλωση ρεύματος στην περίπτωση αυτή είναι 50mA.

GND: Γειωμένες ακίδες.

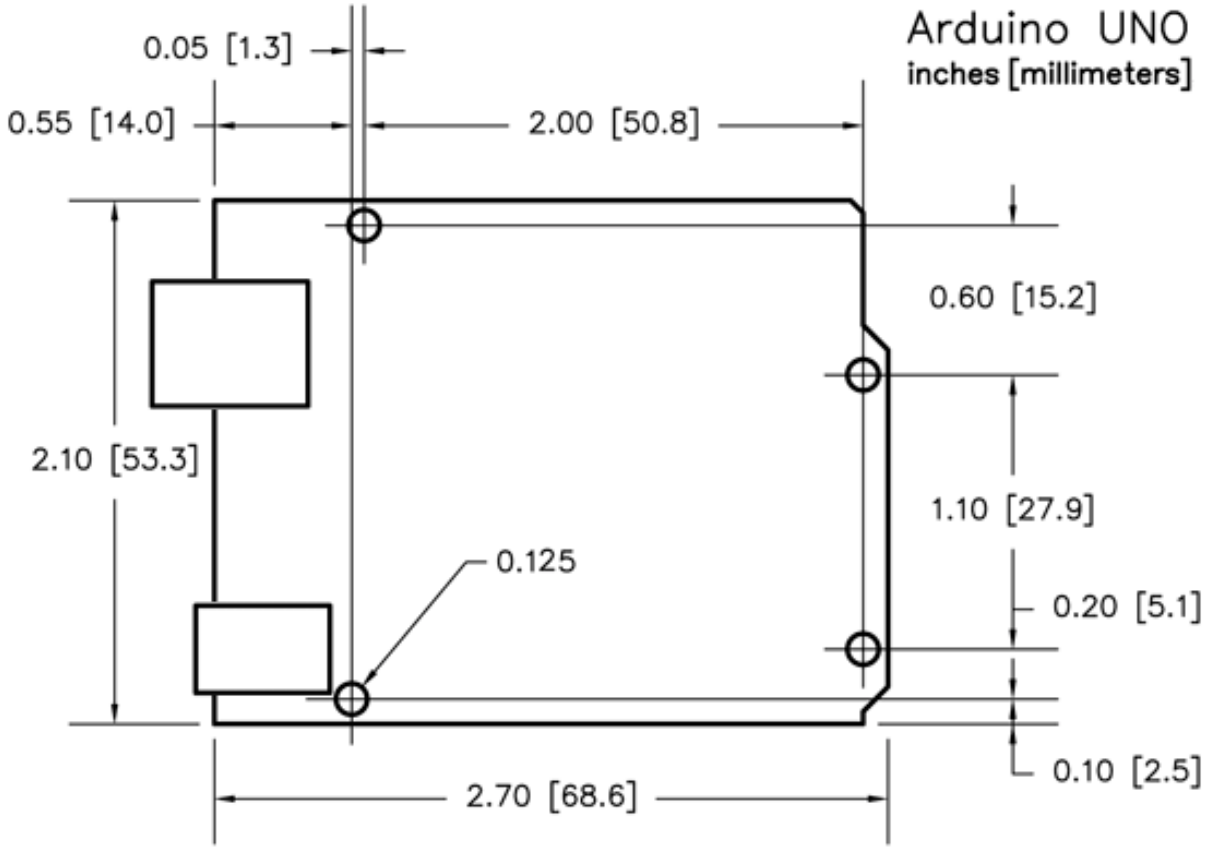
Ο μικροεπεξεργαστής ATmega328P διαθέτει, όπως προείπαμε, τρεις ομάδες μνήμης. Η Flash μνήμη χρησιμοποιείται για την αποθήκευση του Arduino sketch, δηλαδή του κώδικα που θα ανεβάσουμε στην πλατφόρμα. Τα 0.5 KB της Flash χρησιμοποιούνται από τον bootloader (ένα μικρό κομμάτι κώδικα που δίνει τη δυνατότητα ανεβάσματος Arduino sketches στη πλατφόρμα χωρίς την ανάγκη εξωτερικού προγραμματιστή). Η Flash memory δε μπορεί να αλλάξει κατά τη διάρκεια εκτέλεσης του προγράμματος, οπότε χρησιμοποιείται για την αποθήκευση σταθερών δομών δεδομένων, όπως π.χ. πίνακες χαρακτήρων. Την SRAM στην οποία δημιουργείται το sketch και αποθηκεύει τις μεταβλητές όταν τρέχει. Τέλος την EEPROM, η οποία χρησιμοποιείται για την αποθήκευση μακροχρόνιων πληροφοριών.

Το Arduino Uno έχει την δυνατότητα να επικοινωνεί με τον Ηλεκτρονικό Υπολογιστή, έναν άλλον Arduino ή άλλους μικροελεγκτές. Το ολοκληρωμένο ATMega382 παρέχει σειριακή επικοινωνία TTL 5 Volt UART, η οποία είναι διαθέσιμη από τους ακροδέκτες (λήψη RX) 0 και (εκπομπή TX) 1 του ολοκληρωμένου. Επιπλέον, η αναπτυξιακή πλακέτα του Arduino παρέχει σειριακή επικοινωνία με τον Ηλεκτρονικό Υπολογιστή για προγραμματισμό με την βοήθεια ενός ειδικά προγραμματιζόμενου ενσωματωμένου ολοκληρωμένου ATmega382 αντί του chipFTDI. Ωστόσο, αυτό επιτρέπει την πιο γρήγορη ταχύτητα μεταφοράς και γρήγορης σειριακής επικοινωνίας. Με την σύνδεση του Arduino μέσω της θύρας USB αυτό εμφανίζεται ως εικονική σειριακή θύρα COM στο λογισμικό του υπολογιστή. Το firmware ATmega382 χρησιμοποιεί τα προγράμματα οδήγησης USBCOM και δεν χρειάζεται να υπάρχει εξωτερικός παράγοντας. Επομένως, στα Windows απαιτείται μόνο ένα αρχείο .inf . Ένα Arduino περιλαμβάνει ένα τμηματικό όργανο ελέγχου το οποίο επιτρέπει την απλή μορφή κειμένου δεδομένων που αποστέλλονται προς και από τη πλακέτα Arduino. Οι RX και TX λυχνίες LED στην πλακέτα θα αναβοσβήνουν όταν γίνεται μετάδοση δεδομένων μέσω του USB-to-chip σειριακή και USB σύνδεση με τον υπολογιστή (αλλά όχι για σειριακή επικοινωνία στις ακίδες 0 και 1).

Η πιο διαδεδομένη πλακέτα της οικογένειας του Arduino είναι το Arduino Uno, το οποίο διαφέρει από όλες τις άλλες διαθέσιμες πλακέτες της Arduino στο ότι δεν χρησιμοποιεί το FTDIUSB-to-serial chip driver. Αντ 'αυτού, διαθέτει το Atmega16U2 (Atmega8U2 μέχρι την έκδοση R2) που έχει προγραμματιστεί ως USB-σε-σειριακό μετατροπέα.

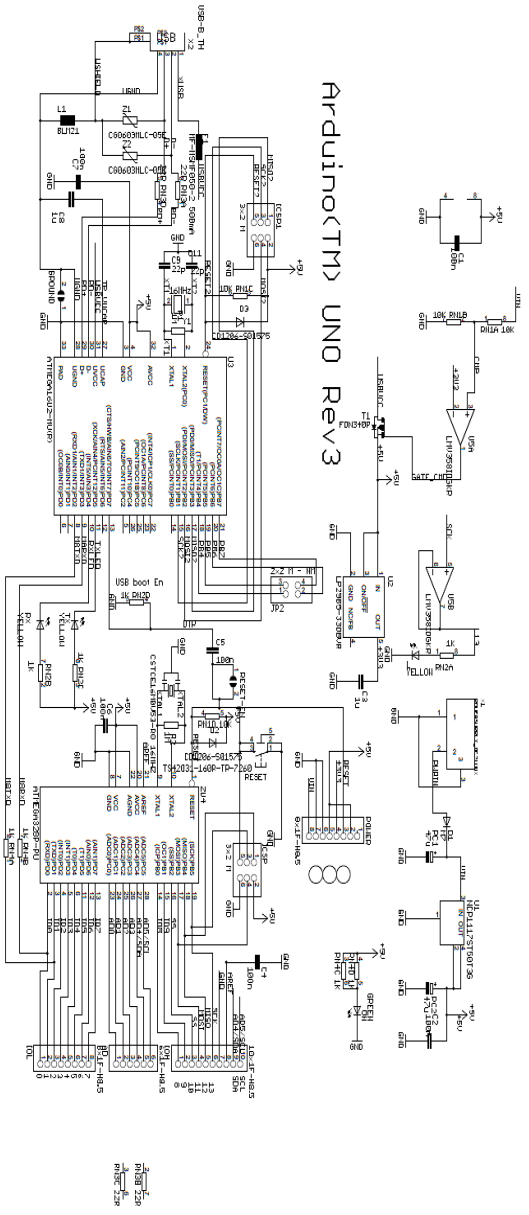
1.3.3.4) ΔΙΑΣΤΑΣΕΙΣ ΚΑΙ ΣΧΗΜΑΤΙΚΟ ΔΙΑΓΡΑΜΜΑ

Στην παρακάτω εικόνα φαίνονται οι διαστάσεις ενός Arduino Uno:



Σχήμα 1.3.3.4-1. Διαστάσεις του Arduino Uno

Τέλος στο σχήμα 1.3.3.4-2 που ακολουθεί παρατίθεται μια συνολική αναλυτικότερη απεικόνιση της πλακέτας όπως αυτή καθορίζεται από την κατασκευάστρια εταιρία:



Σχήμα 1.3.3.4-2. Συνολική απεικόνιση ενός Arduino Uno

1.3.4.4) ΕΦΑΡΜΟΓΕΣ ΒΑΣΙΣΜΕΝΕΣ ΣΤΟ ΠΕΡΙΒΑΛΛΟΝ ARDUINO

Μια αναλυτικότερη προβολή ιδεών-διατάξεων βασισμένων στην πλατφόρμα Arduino που καλύπτουν ένα ευρύ φάσμα αναγκών, θα συμβάλλει κατά κύριο λόγο στην ορθή αντίληψη της ευελιξίας, της χρησιμότητας και του τρόπου σκέψης που κρύβεται πίσω από την τεχνολογία μικροεπεξεργαστών, όπως αυτή του Arduino. Μερικές από αυτές τις «ιδέες» έχουν ήδη κατοχυρωθεί ως πατέντες, ενώ άλλες έχουν παρουσιαστεί σε πλαίσια πτυχιακών εργασιών. Παρακάτω θα αναφερθούμε σε κάποια από τα πιο σημαντικά παραδείγματα εκμετάλλευσης

της τεχνολογίας Arduino σε επίτευξη συγκεκριμένων στόχων, τα οποία συνέβαλαν σε μεγάλο βαθμό στην σύλληψη της ιδέας της παρούσας διπλωματικής εργασίας.

Αρχικά θα παρουσιαστούν ορισμένες εφαρμογές (projects) βασισμένα σε Arduino:

Α)Σύστημα αυτομάτου ελέγχου για μικρούς Ni/H αντιδραστήρες σύντηξης.

Β) Ενεργειακή αποδοτικότητα / αναφορά: χρήση αισθητήρων σε διακόπτες στον πίνακα εναλλασσόμενου ρεύματος (AC) οι οποίοι θα αναφέρουν και ίσως θα καταγράφουν τα δεδομένα (data).

Γ) Σύστημα / ελέγχου ισχύος UPS: Προσθήκη ελέγχου ισχύος και αναφοράς σε κανονική μονάδα UPS (Uninterruptible Power Supply).

Δ) Χρήση Arduino για τη συλλογή και την επεξεργασία / μετεωρολογικών δεδομένων / δεδομένων τηλεμετρίας που στη συνέχεια θα οδηγούνται μέσω σειριακών χορδών δεδομένων εξόδου σε έναν ερασιτεχνικό ραδιοφωνικό πομπό χρησιμοποιώντας APRS (Automatic Packet Reporting System).

Ε) Δημιουργία ρολογιού LED ή άλλων εφαρμογών που περιλαμβάνουν lcd οθόνες.

ΣΤ) «Ηχητικό μάτι» (soniceye) για τυφλούς που θα βοηθά στην μετακίνηση τους χωρίς τη χρήση μπαστουνιού.

Ζ) Υλοποίηση ελεγκτή ανεμιστήρα για ηλεκτρονικούς υπολογιστές.

Η) Ασύρματος θερμοστάτης σπιτιού: Ελεγχόμενος από οπουδήποτε από ένα iPhone μέσω 802.11 συνδεσιμότητας.

Θ) Σύστημα μέτρησης διηλεκτρικής σταθεράς: Εκπέμποντας σήματα και συλλέγοντας τα αντίστοιχα δεδομένα από τα inputs κατάλληλα τοποθετημένων αισθητήρων συνδεδεμένα με μικροελεγκτή δίνει τη δυνατότητα καθορισμού της διηλεκτρικής σταθεράς υλικών με τον πλέον απλό τρόπο.

Ι) Κατασκευή τροφοδοτικού παροχής 30V. Ο χειρισμός του τροφοδοτικού πραγματοποιείται ψηφιακά μέσω ενός κωδικοποιητή. Η απεικόνιση των επιθυμητών τιμών τάσης και έντασης γίνονται μέσω μιας οθόνης lcd. Η επικοινωνία της οθόνης και του κωδικοποιητή με το αναλογικό κύκλωμα του τροφοδοτικού πραγματοποιείται μέσω ενός μικροεπεξεργαστή Arduino.

ΙΑ) Απομακρυσμένος έλεγχος ασύγχρονων μηχανών μέσω συντονισμένων γραπτών μηνυμάτων (sms).Ο δίαυλος επικοινωνίας και η δυνατότητα χειρισμού καθίσταται εφικτή μέσω του κατάλληλου προγραμματισμού που γίνεται στην πλατφόρμα του Arduino.

ΙΒ) Έλεγχος ανοικτού βρόχου βηματικού κινητήρα.

ΙΓ) Υλοποίηση embedded συστημάτων χαμηλού κόστους με σκοπό τη μελλοντική χρήση στο συγχρονισμό των δικτύων ενέργειας.

ΙΔ) Άλλη μία σημαντική διάταξη με χρήση του συγκεκριμένου μικροεπεξεργαστή είναι η κατασκευή ενός κινητού DataLogger με την υποστήριξη GPS δυνατοτήτων, ο οποίος μετρά θερμοκρασία, ποσοστό υγρασίας, επίπεδο ατμοσφαιρικού ηλεκτρικού πεδίου (AtmosphericElectricField, akaAEF) και συντεταγμένες. Η διάταξη αυτή δίνει τη δυνατότητα ελέγχοντας την εξέλιξη του AEF να γίνεται έγκαιρη πρόγνωση πιθανής έντονης κεραυνικής δραστηριότητας.

ΙΕ) Κάμερα ασφαλείας (Arduino controlled safety camera): Η συγκεκριμένη εφαρμογή αποτελεί ένα πολύ χαρακτηριστικό παράδειγμα της δυνατότητας της πλατφόρμας αυτής. Ένα Arduino συνδέεται με ανιχνευτές κίνησης, οι οποίοι σε περίπτωση ανίχνευσης κινητικότητας στέλνουν σήμα πίσω στον μικροελεγκτή. Στη συνέχεια ένας σερβοκινητήρας περιστρέφει την κάμερα καταγράφοντας τυχόν περιστατικά.

Οι μικροεπεξεργαστές- μικροελεγκτές, όπως μπορεί να παρατηρήσει κανείς αποτελούν μέρος ενός εντυπωσιακού αριθμού προϊόντων τα οποία βρίσκονται γύρω μας. Το αυτοκίνητό μας, τα τηλεχειριστήριά, η τηλεόρασή, οι ψηφιακές κάμερες, τα κινητά τηλέφωνα, τα πλυντήριά είναι μερικά από αυτά. Δεν θα ήταν υπερβολή να πούμε ότι η χρήση μικροελεγκτών στις μέρες μας είναι καθολική και γενικά κάθε προϊόν το οποίο αλληλεπιδρά με ένα χρήστη περιλαμβάνει ένα μικροελεγκτή, ο οποίος παίζει το ρόλο του «εγκεφάλου» των ηλεκτρονικών κυκλωμάτων. Δεν είναι τυχαίο πλέον, ότι πολλές βιομηχανίες προσανατολίζονται σε εφαρμογές όπου αποτελούνται από ηλεκτρικά κυκλώματα τα οποία ελέγχονται από μικροελεγκτή σε αντίθεση με την πρακτική του παρελθόντος όπου χρησιμοποιούσαν ογκώδεις ηλεκτρονικούς υπολογιστές (PC) ή ηλεκτρονόμους και σύνθετη αλλά μόνιμη λογική. Όλα τα παραπάνω είναι αποτέλεσμα της ανάπτυξης και του σχεδιασμού των ενσωματωμένων συστημάτων. Ο μικροελεγκτής ουσιαστικά αποτελεί ένα αυτόνομο ολοκληρωμένο σύστημα με επεξεργαστή, μνήμη και περιφερειακά και ως εκ τούτου μπορεί να χρησιμοποιηθεί ως ένα ενσωματωμένο σύστημα. Ενώ κάποια ενσωματωμένα συστήματα είναι αρκετά σύνθετα, τα περισσότερα σχεδιάζονται με ελάχιστη μνήμη και μέγεθος κώδικα, χωρίς την ύπαρξη λειτουργικού συστήματος και με λογισμικό χαμηλής πολυπλοκότητας. Ως συσκευές εισόδου – εξόδου συνήθως χρησιμοποιούνται διακόπτες, ηλεκτρικά χειριζόμενοι διακόπτες (ρελές), πηνία, LEDs,

οθόνες LCD, συσκευές ραδιοσυχνοτήτων και αισθητήρες θερμοκρασίας, υγρασίας, ισχύος φωτός κλπ. Τα ενσωματωμένα συστήματα συνήθως δεν διαθέτουν συσκευές εισόδου – εξόδου που συναντώνται στον προσωπικό υπολογιστή όπως πληκτρολόγιο, οθόνη, οπτικά μέσα αποθήκευσης ή εκτυπωτές και τις περισσότερες φορές δεν παρέχουν δυνατότητα αλληλεπίδρασης με το χρήστη .

1.4) ΠΡΟΓΡΑΜΜΑΤΙΣΤΙΚΟ ΠΕΡΙΒΑΛΛΟΝ LAZARUSIDE

1.4.1) ΓΕΝΙΚΑ ΣΤΟΙΧΕΙΑ

Το LAZARUS IDE είναι ένα δωρεάν ανεξάρτητο πλατφόρμας (cross platform) ολοκληρωμένο περιβάλλον (IDE) για την ταχεία ανάπτυξη εφαρμογών (RAD). Το συγκεκριμένο περιβάλλον χρησιμοποιεί μεταγλωττιστή PASCAL, υποστηρίζοντας, σε διαφόρους βαθμούς, διαλέκτους της object PASCAL. Οι προγραμματιστές λογισμικού χρησιμοποιούν το LAZARUS για να δημιουργήσουν εγγενή κώδικα κονσόλας και εφαρμογές με γραφική διεπαφή χρήστη (GUI) για την επιφάνεια εργασίας, καθώς επίσης και για κινητές συσκευές, web εφαρμογές, υπηρεσίες web, οπτικά εξαρτήματα και βιβλιοθήκες λειτουργίας (.so, .dll, κλπ, για χρήση από άλλα προγράμματα). Ο compiler Δωρεάν Pascal υποστηρίζει μια σειρά από διαφορετικές πλατφόρμες, όπως τα Mac, Linux και Windows.

Το LAZARUS ‘’κληρονομεί’’ τρία χαρακτηριστικά από τη χρήση του compiler της Pascal: compile, ταχύτητα εκτέλεσης, και cross-compile. Οι compiler της Pascal ωφελούνται από τη δομή της γλώσσας Pascal και από τη σταθερή εξέλιξη του σχεδιασμού compiler Pascal για την ανάπτυξη μεγάλων εφαρμογών γρήγορα. Κατά την ανάπτυξη προγραμμάτων αναφοράς για μετρήσεις απόδοσης, το LAZARUS παράγει προγράμματα που εμφανίζουν παρόμοιες επιδόσεις με τα ίδια προγράμματα γραμμένα σε γλώσσα C.

Γενικά μια εφαρμογή που οι προγραμματιστές μπορούν να δημιουργήσουν με τη χρήση του συγκεκριμένου περιβάλλοντος, μπορεί δυνητικά να αναπτυχθεί και να εκτελεστεί σε οποιαδήποτε πλατφόρμα, η οποία διαθέτει ελεύθερο μεταγλωττιστή Pascal.

1.4.2) ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ

Το περιβάλλον LAZARUS προσφέρει ένα εξαιρετικά άρτιο περιβάλλον ανάπτυξης για τη δημιουργία ποικίλων εφαρμογών βασιζόμενο σε τρεις απλούς κανόνες:

Α)ΕΛΕΥΘΕΡΙΑ:
Το LAZARUS αποτελεί μια crossplatform εφαρμογή επιτρέποντας το να λειτουργεί σε διάφορες πλατφόρμες όπως τα Windows, τα MacOS ή τα LINUX, γεγονός που του δίνει την απαιτούμενη ανεξαρτησία.
Β)ΕΥΚΟΛΙΑ ΧΡΗΣΗΣ:
Ο προγραμματιστής από την πρώτη του επαφή με το συγκεκριμένο περιβάλλον αντιλαμβάνεται την μεγάλη ευχέρεια επιλογών. Μεταξύ άλλων του παρέχονται οι δυνατότητες δημιουργίας layout μέσω form designer, code editing, automatic synchronization, debugging, refactoring κ.α.
Γ)ΔΥΝΑΤΟΤΗΤΑ ΕΠΕΚΤΑΣΗΣ:
Το LAZARUS αποτελεί ένα opensource περιβάλλον που επιτρέπει την προσθήκη ή την τροποποίηση υπαρχόντων στοιχείων (components), καθώς διαθέτει πλήθος διαθέσιμων βιβλιοθηκών με δυνατότητα πρόσβασης και άλλων μέσω συγκεκριμένων διασυνδέσεων.

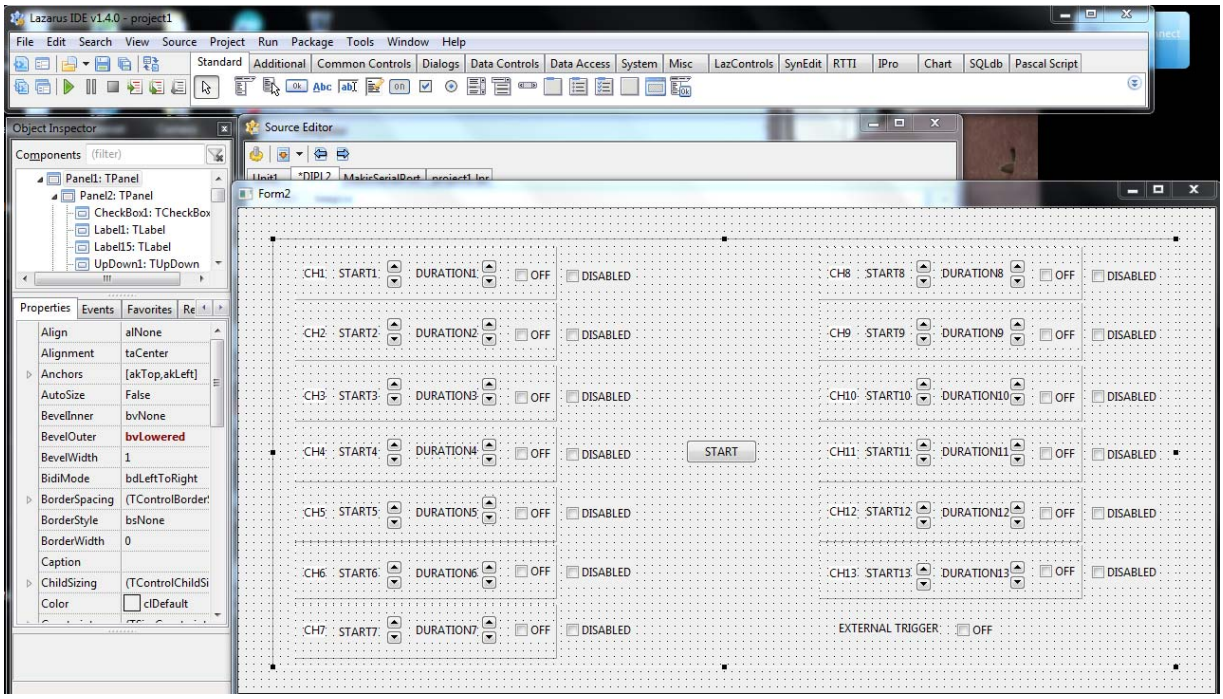
1.4.3) ΤΟ ΠΕΡΙΒΑΛΛΟΝ LAZARUS ΩΣ CROSSPLATFORM

Το LAZARUS χρησιμοποιεί την FreePascal ως τον back-end compiler του. Ως εκ τούτου, μπορεί, θεωρητικά, να χρησιμοποιηθεί για την ανάπτυξη εφαρμογών για όλες τις πλατφόρμες που υποστηρίζονται από την FreePascal. Παράλληλα παρέχει ένα πλαίσιο εφαρμογής cross-platform το οποίο ονομάζεται the Lazarus Component Library (LCL), η οποία παρέχει ένα ενιαίο, ενοποιημένο περιβάλλον εργασίας για τους προγραμματιστές.

1.4.4) ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Η πρώτη προσπάθεια για την ανάπτυξη ενός visual περιβάλλοντος IDE για την free Pascal χρονολογείται το 1998 με το MEGIDO project. Για διάφορους όμως λόγους η προσέγγιση αυτή απέτυχε. Τότε μερικοί από τους προγραμματιστές του Megido project αποφάσισαν να ξεκινήσουν ένα νέο έργο βασιζόμενο σε μια πιο ευέλικτη βάση. Έτσι ξεκίνησε το LAZARUS project τον Φεβρουάριο του 1999 από τους Cliff Baeseman, Shane Miller, Michael A. Hess, Marc Weustink και τον Mattias Gaertner. Σε αντίθεση με το Megido, το LAZARUS ήταν μια μεγάλη επιτυχία. Η πρώτη προκαταρκτική εκδοχή LCL κυκλοφόρησε το 2001, ενώ το 2003 η πρώτη έκδοση beta του LAZARUS (0.9.0.3) φιλοξενήθηκε στο SourceForge. Η πρώτη τελική

έκδοση του LAZARUS (1.0) κυκλοφόρησε το 2012. Τέλος το LAZARUS (1.2) κυκλοφόρησε το 2014 με σημαντικές βελτιώσεις.Πλέον είναι διαθέσιμο για μια σειρά από λειτουργικά συστήματα, συμπεριλαμβανομένων των Linux, MacOS X, BSD, Solaris και Windows. Περισσότερα από τέσσερα εκατομμύρια μεταφορτώσεις (downloads) από το SourceForge (Μάρτιος 2014) σηματοδοτούν την αυξανόμενη δημοτικότητα αυτού του IDE.



Εικόνα 1.4-1. Χαρακτηριστική απεικόνιση του περιβάλλοντος LAZARUS IDE.

ΚΕΦΑΛΑΙΟ 2ο

Η ΔΙΑΤΑΞΗ ΣΥΓΧΡΟΝΙΣΜΟΥ ΜΕΤΡΗΣΕΩΝ ΜΕ ΧΡΗΣΗ ΜΙΚΡΟΕΛΕΓΚΤΗ ARDUINO

Στο κεφάλαιο παρουσιάζεται το κατασκευαστικό κομμάτι της εργασίας αυτής και η υλοποίηση της με τα υλικά και μέσα τα οποία αναλύθηκαν στα προηγούμενα κεφάλαια. Επίσης δίνονται αναλυτικά τα σημαντικότερα μέρη και στοιχεία του κώδικα προγραμματισμού που απαιτήθηκε για την δημιουργία της διάταξης.

2.1) ΑΝΑΛΥΣΗ

Η ανάλυση της διάταξης σε επίπεδο συνιστωσών αποτέλεσε ένα από τα βασικότερα στάδια της κατασκευής, καθώς μια επιτυχημένη και άρτια προ-επεξεργασία καθορίζει κατά μεγάλο βαθμό την ελαχιστοποίηση των σφαλμάτων και την τυχόν εμφάνιση απροσδόκητων μεταβλητών που θα δυσκολέψουν τον βαθμό υλοποίησης. Ο χρόνος λοιπόν αν και αρχικά μπορεί να φαντάζει ελάχιστα ποιοτικά αξιοποιήσιμος, ουσιαστικά είναι αυτός που θα διευκολύνει την μετέπειτα πορεία της διπλωματικής εργασίας.

Στόχος της διάταξης αυτής σε επίπεδο πλήρους αξιοποίησης των δυνατοτήτων της είναι ο συγχρονισμός 12 επιμέρους εξωτερικών διατάξεων μέσω 12 καναλιών του Arduino Uno (μια θύρα- η TX, channel 1- διατηρείται για εφεδρεία). Ο χειριστής μπορεί να καθορίσει το χρόνο έναρξης και το χρόνο διάρκειας της λειτουργίας εισάγοντας στο περιβάλλον διεπαφής τα αντίστοιχα δεδομένα.

2.2) ΚΑΤΑΣΚΕΥΗ

2.2.1) ΣΧΕΔΙΑΣΜΟΣ

Η ευκολία και η απλότητα χρήσης της διάταξης αποτέλεσε τον κυριότερο παράγοντα για την επιλογή των συνιστωσών, ο συνδυασμός των οποίων θα έδινε το επιθυμητό αποτέλεσμα. Αρχικά ως μέσο λειτουργίας και επικοινωνίας χρήστη και διάταξης επιλέχθηκε μεταξύ πολλαπλών επιλογών ο ηλεκτρονικός υπολογιστής λόγω της εξοικείωσης, αλλά και των δυνατοτήτων που προσφέρει. Συνεχίζοντας με βασικό κριτήριο τη διατήρηση του χαμηλού κόστους η επιλογή της οικογένειας μικροελεγκτών Arduino και ειδικότερα του Arduino Uno συμβάδιζε απόλυτα με την ιδέα και το σκοπό της εργασίας αυτής, ενώ από την πλευρά της η πλατφόρμα του Lazarus μηδένισε το κόστος κατασκευής μιας βάσης επικοινωνίας και ελέγχου χρήστη-διάταξης.

2.2.2) ΛΕΠΤΟΜΕΡΗΣ ΠΕΡΙΓΡΑΦΗ ΣΥΝΙΣΤΩΣΩΝ

Ακολουθεί ένας αναλυτικός πίνακας που περιγράφει όλα τα υλικά που χρησιμοποιήθηκαν για την κατασκευή αυτή (πίνακας 2.2.2-1):

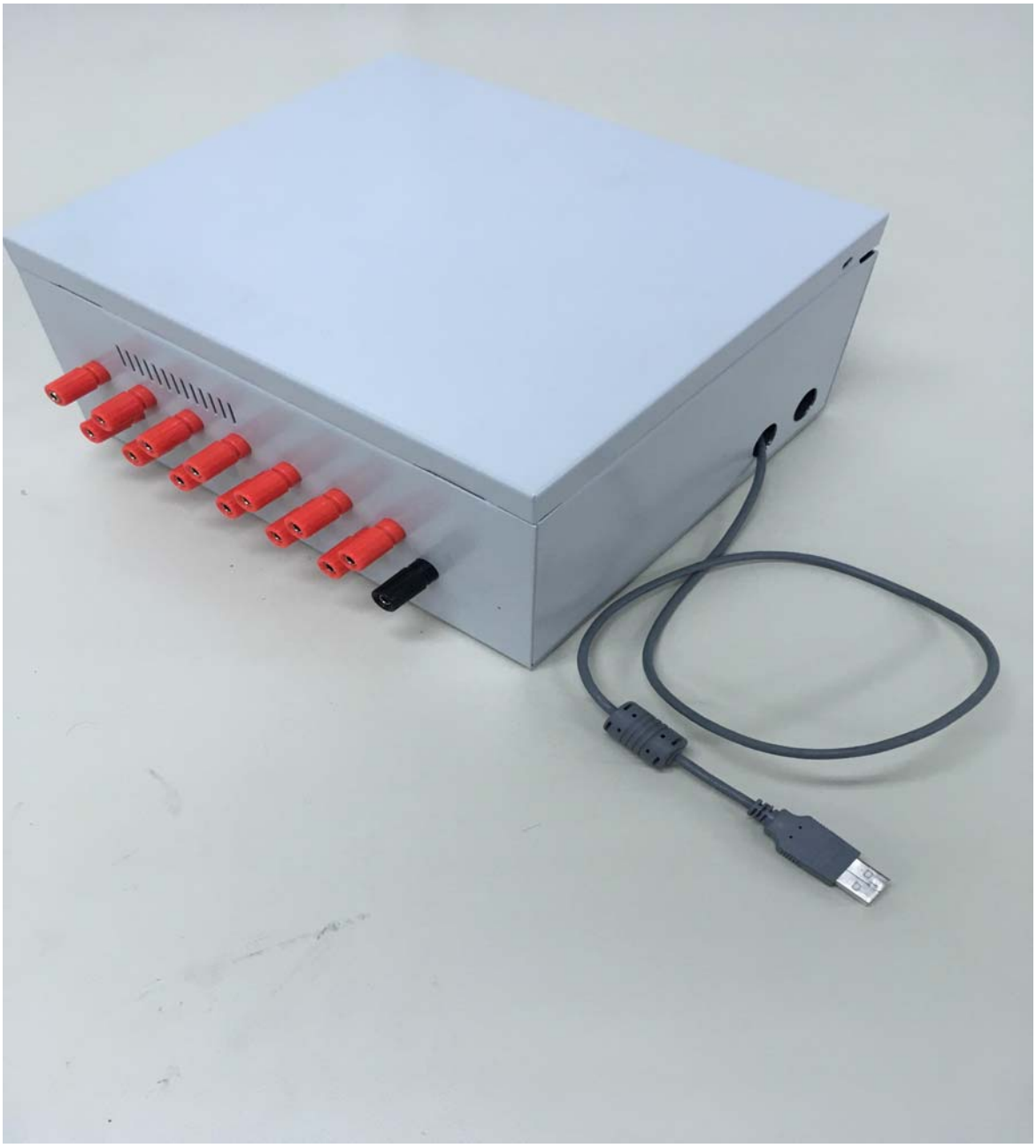
ΠΕΡΙΓΡΑΦΗ ΥΛΙΚΟΥ	ΠΟΣΟΤΗΤΑ
1. Μικροελεγκτής Arduino Uno	1
2. Φορητός Ηλεκτρονικός Υπολογιστής	1
3. Καλώδιο σύνδεσης USB	1
4. Κυτίο (Pillar)	1

Με βάση αυτά τα υλικά και τα αντίστοιχα platforms του LAZARUS και του ARDUINO υλοποιήθηκαν και τα αντίστοιχα προγράμματα τα οποία θα χειρίζεται ο χρήστης.

2.2.3) ΣΥΝΔΕΣΜΟΛΟΓΙΑ ΤΟΥ ΚΥΚΛΩΜΑΤΟΣ

Για τη κατασκευή της διάταξης χρησιμοποιήθηκε η πλακέτα Arduino Uno η οποία τοποθετήθηκε σε κατάλληλο κυτίο (pillar) όπως φαίνεται στο σχήμα Γ. Για την τοποθέτηση της πλακέτας χρησιμοποιήθηκαν κατάλληλα μονωτικά υλικά, ώστε να αποφευχθεί πιθανό βραχυκύκλωμα μεταξύ του μικροελεγκτή και της πλάτης του pillar. Η ορθή αντιστοίχιση καλωδίων και θυρών αποτέλεσε, ίσως, το σημαντικότερο στάδιο του κατασκευαστικού μέρους της υλοποίησης της συγκεκριμένης διάταξης, αφού ένα λάθος θα οδηγούσε σε μη επιθυμητά αποτελέσματα. Επίσης όπως μπορεί κάποιος να διαπιστώσει από τα σχήματα 2.2.3-3 έως 2.2.3-9 υπάρχει επαρκής χώρος χειρισμού εντός της διάταξης σε περίπτωση αναγκαίας εξωτερικής επέμβασης. Παρόλα αυτά η ανάγκη αυτή έχει σχεδόν εκμηδενιστεί διότι όλες οι απαιτούμενες συνδεσμολογίες έχουν ήδη πραγματοποιηθεί. Ο χρήστης καλείται ουσιαστικά να συνδέσει στα κανάλια (σχήμα Α) τις διατάξεις που επιθυμεί να συγχρονίσει χρησιμοποιώντας την κατάλληλη καλωδίωση, και βέβαια να συνδέσει τον ηλεκτρονικό υπολογιστή στη διάταξη συγχρονισμού με το καλώδιο USB που υπάρχει. Ο αερισμός και η ψύξη αν και δεν αποτέλεσε βασικό παράγοντα που εμπόδισε την κατασκευή, επιτυγχάνεται μέσω των σχισμών του κυτίου (pillar).

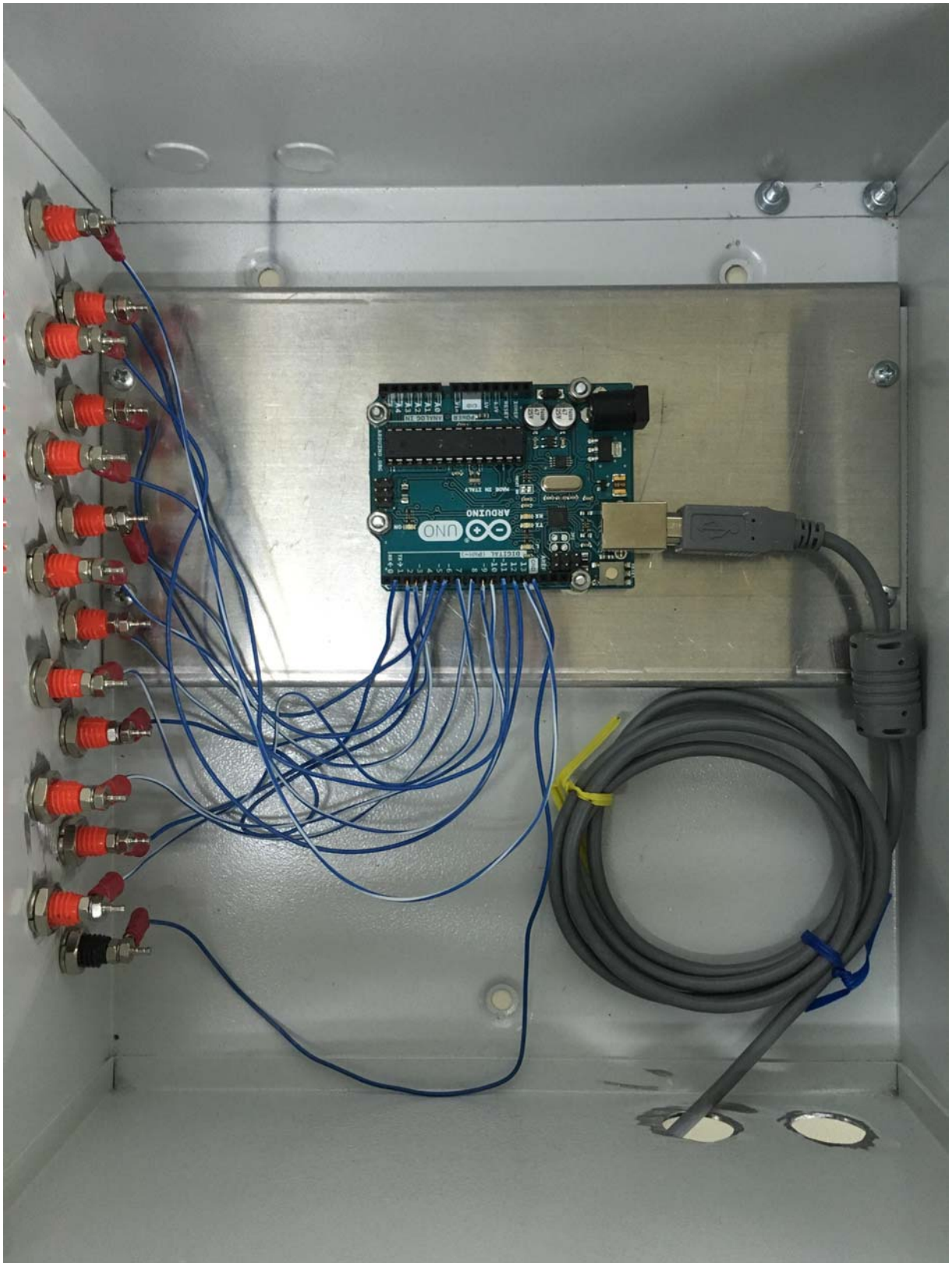
Στο σχήμα 2.2.3-1 που ακολουθεί φαίνεται η εξωτερική εικόνα της διάταξης:



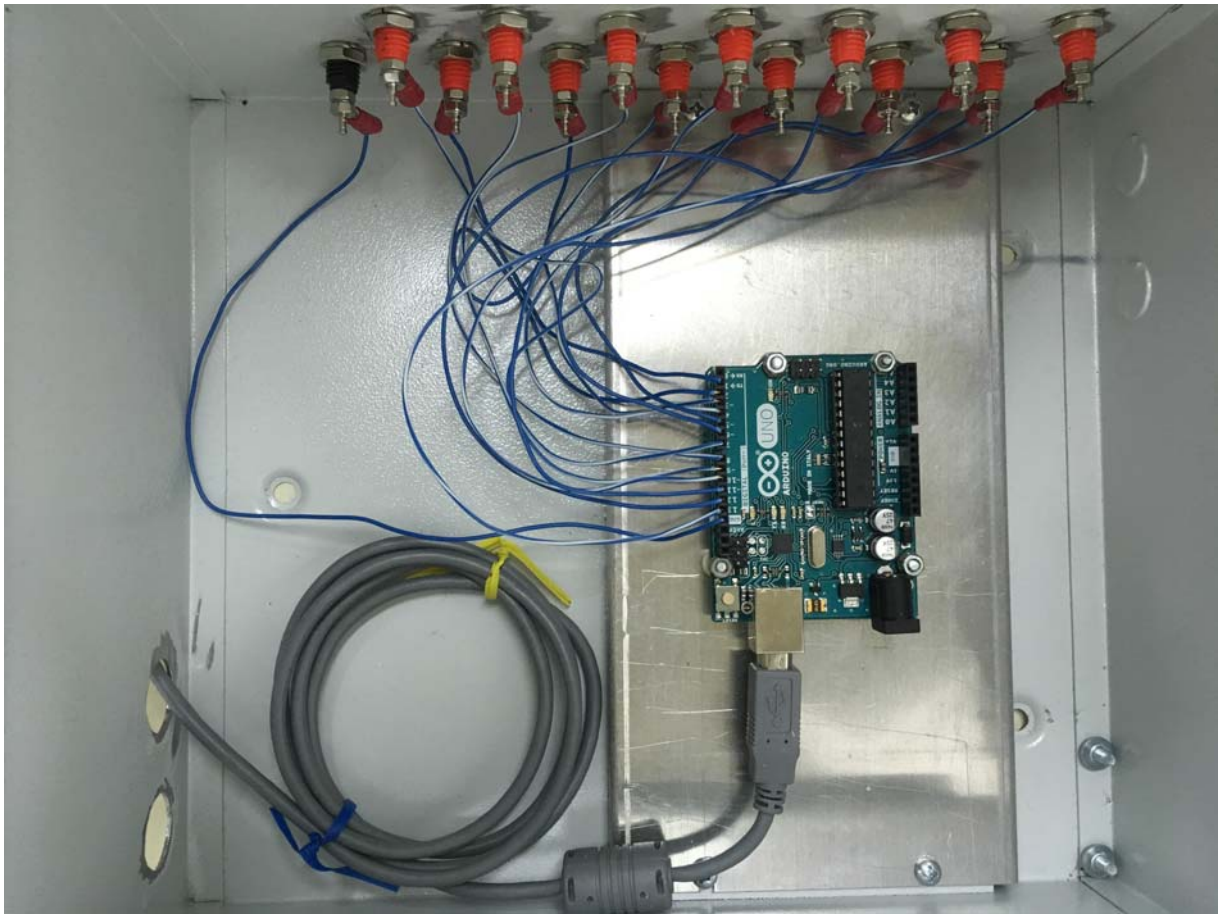
Σχήμα 2.2.3-1. Εξωτερική απεικόνιση της διάταξης συγχρονισμού.

Σχήμα 2.2.3-2. Τα συνολικά κανάλια της διάταξης.

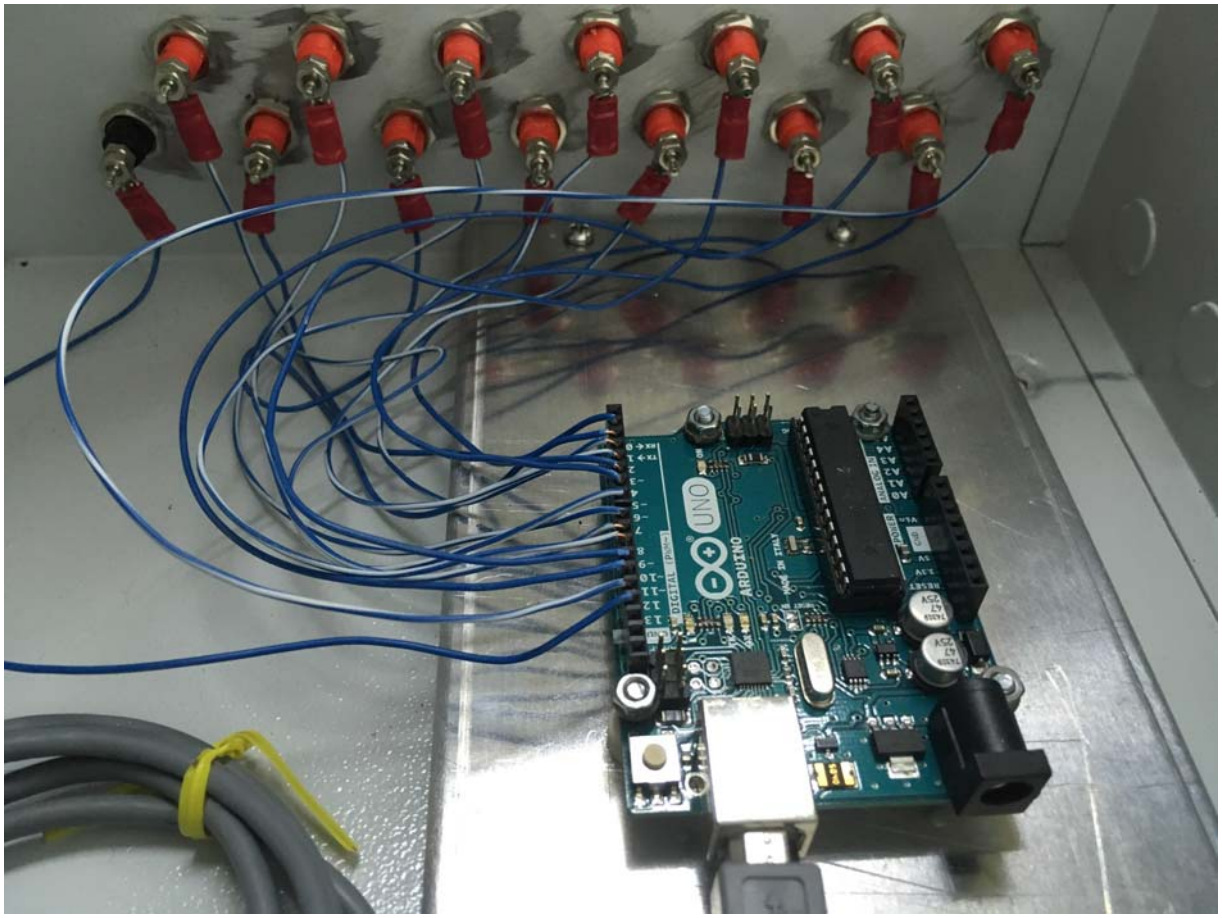
Στην παραπάνω φωτογραφία (Σχήμα 2.2.3-2) απεικονίζονται τα κανάλια μέσω των οποίων γίνεται η σύνδεση της συσκευής συγχρονισμού και των υπό έλεγχο διατάξεων. Με πορτοκαλί χρώμα διακρίνονται τα 12+1 κανάλια που σχηματίστηκαν για τον συγχρονισμό και με μαύρο το κανάλι που συνδέεται η γείωση της διάταξης.



Σχήμα 2.2.3-3. Η εσωτερική συνδεσμολογία της συσκευής συγχρονισμού.

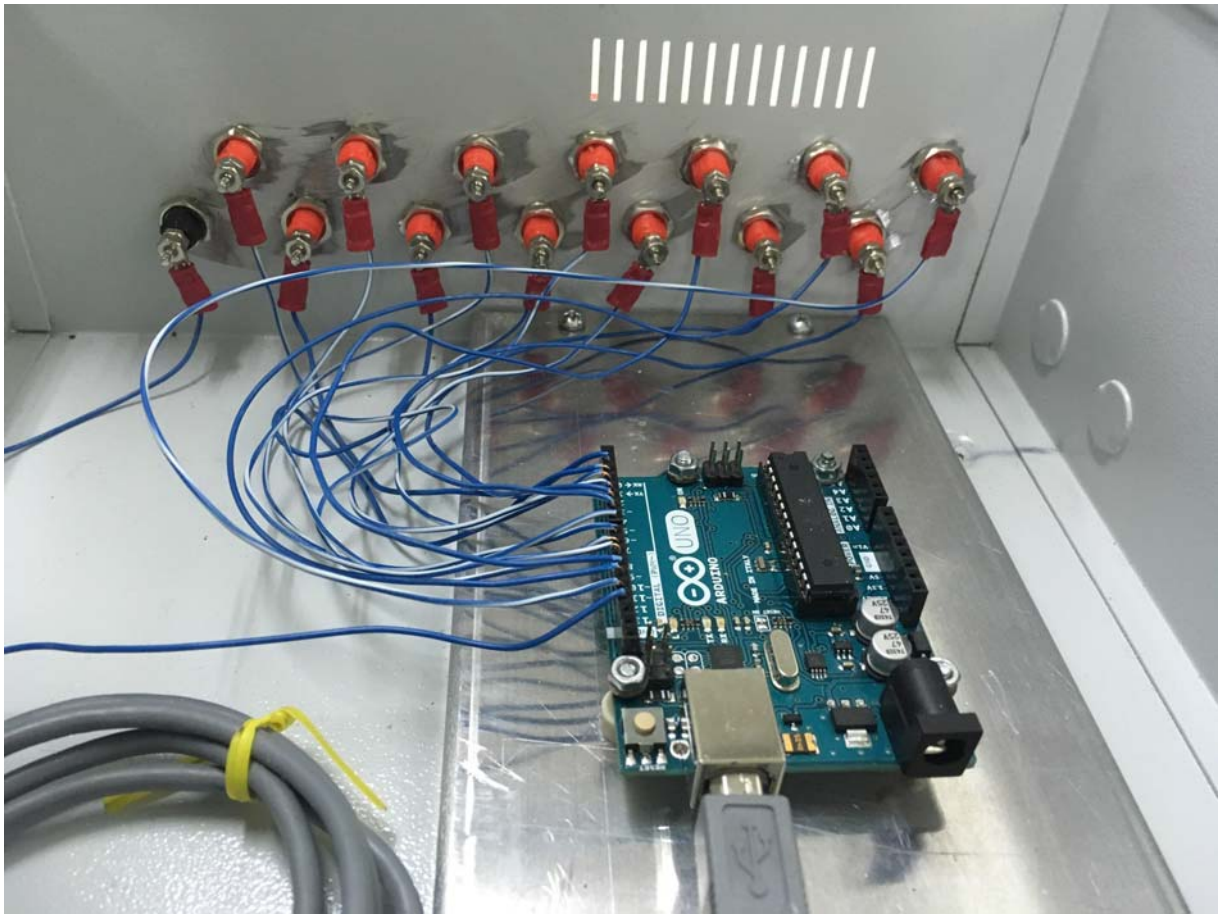


Σχήμα 2.2.3-4. Η εσωτερική συνδεσμολογία της συσκευής συγχρονισμού.



Σχήμα 2.2.3-5. Η εσωτερική συνδεσμολογία της συσκευής συγχρονισμού.

Η καλωδίωση και η συνδεσμολογία των θυρών του Arduino Uno και των εξόδων του κυτίου είναι προκατασκευασμένη. Όλα τα καλώδια είναι μονωμένα.

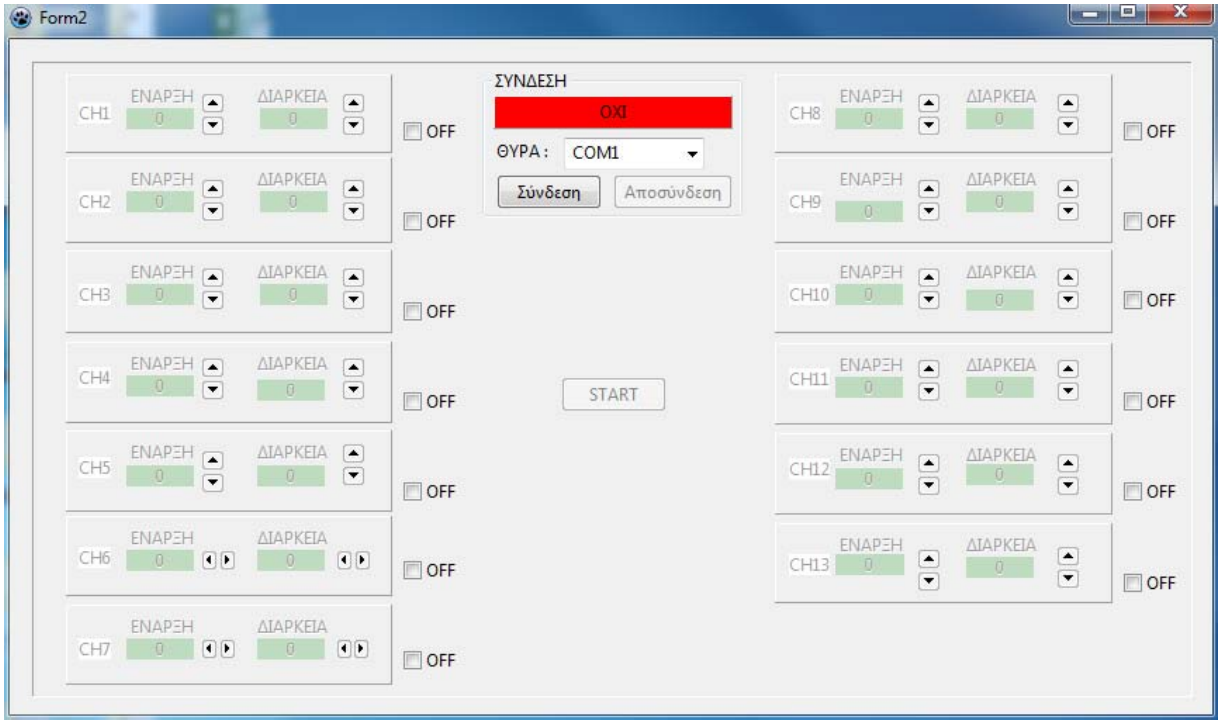


Σχήμα 2.2.3-6. Η εσωτερική συνδεσμολογία της συσκευής συγχρονισμού.

Η σειρά με την οποία εμφανίζονται τα κανάλια εξόδου ακολουθεί αυτή που έχει επισημάνει και η κατασκευάστρια εταιρεία του Arduino Uno. Συγκεκριμένα όπως εμφανίζονται στο παραπάνω σχήμα η πάνω σειρά θυρών αντιστοιχεί σε εξόδους 7-13, ενώ η κάτω σειρά περιλαμβάνει τις εξόδους 1-6 και αυτή της γείωσης (μαύρο χρώμα).

Σ' αυτό το σημείο θα παρουσιαστεί αναλυτικά το γραφικό περιβάλλον διεπαφής του χρήστη και της διάταξης συγχρονισμού που δημιουργήθηκε μέσω του LAZARUS IDE.

Ανοίγοντας λοιπόν την εφαρμογή εμφανίζεται στην οθόνη του ηλεκτρονικού υπολογιστή η εικόνα του σχήματος 2.2.3-7. Χαρακτηριστικό αποτελεί ότι ακόμα δεν έχει επιτευχθεί σύνδεση μεταξύ διάταξης και υπολογιστή, και συνεπώς ο χρήστης βλέπει με κόκκινο χρώμα «ΣΥΝΔΕΣΗ ΟΧΙ». Για να προχωρήσει , αφού βέβαια έχει συνδέσει το καλώδιο USB της διάταξης σε κάποια θύρα USB του Η/Υ, πρέπει να ελεγχθεί η COM της θύρας αυτής και να ενημερώσει κατάλληλα την εφαρμογή επιλέγοντας τη σωστή COM στο πλήκτρο ΘΥΡΑ. Τώρα δύναται να επιτευχθεί η σύνδεση Η/Υ και διάταξης και πατώντας το πλήκτρο σύνδεση η ένδειξη τροποποιείται και γίνεται πράσινου χρώματος υποδεικνύοντας «ΣΥΝΔΕΣΗ ΝΑΙ».



Σχήμα 2.2.3-7. Το περιβάλλον διεπαφής χρήστη και διάταξης σε κατάσταση ΜΗ σύνδεσης.

Για την εισαγωγή των δεδομένων ο χρήστης θα πρέπει να ενεργοποιήσει τα κατάλληλα κανάλια (CHANNELS). Κατά την κατασκευή της πλατφόρμας διεπαφής σχεδιάστηκαν πάνελ πάνω στα οποία τοποθετήθηκαν τα κανάλια, ώστε να υπάρχει η δυνατότητα διακριτής επιλογής των επιθυμητών λειτουργικών καναλιών ανά χρήση. Συγκεκριμένα το πλήκτρο OFF (σχήμα Z) εάν επιλεχθεί μετατρέπει την ένδειξη του σε “ON” , ενεργοποιώντας το αντίστοιχο πάνελ και δη το συγκεκριμένο κανάλι. Σ’ αυτό το σημείο ο χρήστης μπορεί να εισάγει τα δεδομένα για τη λειτουργία της διάταξης. Καλείται να επιλέξει κατάλληλο χρόνο έναρξης λειτουργίας της προς συγχρονισμό εξωτερικής διάταξης, αλλά και τη χρονική διάρκεια λειτουργίας. Τέλος εφόσον έχουν εισαχθεί όλα τα δεδομένα, πατώντας το μπουτόν που βρίσκεται στο κέντρο της πλατφόρμας διεπαφής με ένδειξη «START» η διάταξη συγχρονισμού μεταδίδει τα δεδομένα μέσω της σειριακής σύνδεσης στο Arduino Uno και με αυτό τον τρόπο ξεκινά η λειτουργία της. Με το πέρας χρήσης της διάταξης θα πρέπει να αρχικά να αποσυνδεθεί το περιβάλλον διεπαφής, ακολουθώντας την αντίστροφη πορεία από αυτή της ενεργοποίησης.

2.3) ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ – ΛΕΙΤΟΥΡΓΙΑ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

2.3.1) ΚΩΔΙΚΑΣ ΥΛΟΠΟΙΗΣΗΣ ΠΡΟΓΡΑΜΜΑΤΟΣ

Στο σημείο αυτό παρουσιάζονται τα βασικά σημεία του κώδικα που απαιτήθηκε για την ανάπτυξη τόσο της πλατφόρμας διεπαφής όσο και για την επίτευξη της επικοινωνίας της με το μικροελεγκτή Arduino Uno.

2.3.1.1) ΚΩΔΙΚΟΠΟΙΗΣΗ ΠΛΑΤΦΟΡΜΑΣ ΔΙΕΠΑΦΗΣ ΧΡΗΣΤΗ – ΣΥΣΤΗΜΑΤΟΣ (LAZARUS)

Ήδη από τον ορισμό της έννοιας «πλατφόρμα διεπαφής» μπορούμε να αντιληφθούμε τη σημασία της, καθώς μέσω αυτής καθίσταται ουσιαστικά δυνατή η επικοινωνία χρήστη και διάταξης. Στο σημείο αυτό αναφέρονται οι βασικές ρουτίνες του προγράμματος:

Αρχικά γίνεται η δήλωση του προγράμματος και των απαιτούμενων βιβλιοθηκών:

```
unit DIPL2;
{$mode objfpc}{$H+}
interface
uses
Classes, SysUtils, FileUtil, Forms, Controls, Graphics, Dialogs, ExtCtrls,
StdCtrls, ComCtrls;
type
{ TForm2 }
TForm2 = class(TForm)
```

Παρεμβάλλεται η αναλυτική αναφορά των units της πλατφόρμας και declaration των procedures του προγράμματος:

```
private
{ private declarations }
public
{ public declarations }
end;
var
Form2: TForm2;
var t:integer;
implementation
{$R *.lfm}
{ TForm2 }
```

Έλεγχος των checkboxes για ενημέρωση της φόρμας – ενεργοποίηση καναλιών. Η διαδικασία αυτή είναι αναγκαία για την επιλογή από πλευράς χρήστη των καναλιών που θα χρησιμοποιηθούν σε κάθε λειτουργία της διάταξης. Ενδεικτικά δίδεται για το checkbox. Ο κώδικας επαναλαμβάνεται για όλα τα checkboxes των καναλιών:

```
procedure TFormX.CheckBoxXChange(Sender: TObject);
begin
        if checkboxX.Checked=true then
        begin
                checkboxX.Caption:='ON';
                panelX.enabled:=true;
        end
        else
        begin
                checkboxX.caption:='OFF';
                panelX.enabled:=false;
        end;
end;
```

Ενεργοποίηση της διεπαφής: *πάτημα του πλήκτρου Start. Αποτελεί το στάδιο κατά το οποίο εισάγονται, επεξεργάζονται και αποστέλλονται τα δεδομένα που έχει εισάγει ο χρήστης στο περιβάλλον διεπαφής στο Arduino Uno:*

Αρχικά ορίζονται οι πίνακες που θα χρησιμοποιηθούν από το πρόγραμμα για τα δεδομένα που θα εισαχθούν από το χρήστη.

```
var ts: array [1..13] of integer;
var td: array [1..13] of integer;
var tt: array [1..13] of integer;
var start: array [1..13] of integer;
var duration: array [1..13] of integer;
```

Η διαδικασία εισαγωγής των δεδομένων αποτελείται σε πρώτο στάδιο από την αρχικοποίηση των πινάκων start και duration (πίνακες που θα περιέχουν τη χρονική στιγμή έναρξης και τη χρονική διάρκεια λειτουργίας των καναλιών της διάταξης):

```
Begin
        For k:=1 to 13 do
                begin
                        start[k]:=0;
                        duration[k]:=0;
```

```
        end;
```

Εισαγωγή τιμών στους πίνακες για τα ενεργοποιημένα κανάλια. Τα μή ενεργοποιημένα έχουν τιμή 0:

```
if checkboxX.checked=true then
begin
        start[1]:=updownX.position;
        duration[1]:=updownX.position;
end;
```

Αντιγραφή των πινάκων ts[i]=start[i] και td[i]=duration[i]:

```
for i1:=1 to 13 do
        begin
                ts[i1]:=start[i1];
                td[i1]:=duration[i1];
end;
```

Για τη σωστή λειτουργία απαιτείται η ταξινόμηση των τιμών του πίνακα ts[i] (tstart) με τη μέθοδο bublesort, ώστε να είναι δυνατή η επιλογή από το πρόγραμμα της χρονικής στιγμής που θα ενεργοποιηθεί η λειτουργία της διάταξης:

```
for i2:=1 to 12 do
        begin
                swaps:=false;
                for j1:=1 to 12 do
                        begin
                        if (ts[j1]>ts[j1+1]) then
                                begin
                                        swaps:=true;
                                        swap(ts[j1],ts[j1+1]);
                                end;
                        end;
                        if (NOT swaps) then break;
        end;
```

Αντίστοιχα πραγματοποιείται και η ταξινόμηση των τιμών του πίνακα td[i] (tduration) με τη μέθοδο bublesort:

```
for i3:=1 to 12 do
	begin
		swaps:=false;
		for j2:=1 to 12 do
			begin
			if (td[j2]>td[j2+1]) then
				begin
					swaps:=true;
					swap(td[j2],td[j2+1]);
				end;
			end;
			if (NOT swaps) then break;
	end;
```

Η διάταξη θα πρέπει να παραμείνει σε κατάσταση λειτουργίας για συγκεκριμένο χρονικό διάστημα που επιλέγεται με βάση την ακόλουθη λογική: Αρχικά πραγματοποιείται η πρόσθεση των πινάκων ts[i] και td[i] και δημιουργείται ένας νέος πίνακας tt[i], που περιέχει το σύνολο των χρονικών διαστημάτων ταξινομημένα με αύξουσα σειρά:

```
m:=0;
for m:=1 to 13 do
	begin
		tt[m]:=ts[m]+td[m];
	end;
```

Το στοιχείο tt[13] θα δίνει το μεγαλύτερο χρόνο (0-tt[13] ο κύκλος λειτουργίας των εξόδων κατά το πάτημα του πλήκτρου start), ο οποίος θα αποτελεί ουσιαστικά και το χρόνο κατά τον οποίο η διάταξη συγχρονισμού θα παραμένει σε κατάσταση λειτουργίας:

Για την ενεργοποίηση των καναλιών εξόδου δίνονται αρχικά σε όλα η τιμή 0 (απενεργοποιημένη κατάσταση). Ο χρήστης έχοντας ενεργοποιήσει τα κανάλια πατώντας τα αντίστοιχα πλήκτρα ON-OFF δίνει τιμή 1 σε αυτά που θα ενεργοποιηθούν. Για τα κανάλια αυτά θα πρέπει να υπολογισθεί η συνολική χρονική διάρκεια λειτουργείας, μια διαδικασία που παρουσιάζεται ακολούθως (στο συνολικό σύστημα έχει οριστεί ένα ρολόι –timer- το οποίο κράτα σε συγχρονισμό την λειτουργία ελέγχου των δεδομένων που θα εισαχθούν στο περιβάλλον διεπαφής.). Στη συνέχεια το σύστημα αποστέλλει σε μορφή λέξης B.X.X.X.X.X.X.X.X.X.X.X.X.X. τα δεδομένα στον μικροελεγκτή μέσω μιας εντολής, η οποία ουσιαστικά αποτελεί και τη διασύνδεση Η/Υ και Arduino :

Κατά τη διάρκεια εκτέλεσης του προγράμματος έχει οριστεί ως αρχική τιμή των μεταβλητών εισόδου η μηδενική, δηλαδή:

Inpi:='0' για i:=1 to 13

Στη συνέχεια έχοντας ο χρήστης εισάγει τις τιμές των μεταβλητών στο πρόγραμμα, πραγματοποιείται έλεγχος μέσω της ακόλουθης διαδικασίας για το ποιες είσοδοι έχουν επιλεγεί να ενεργοποιηθούν:

```
t:=0;
timer1.Enabled:=true;
while t<tt[13] do
        begin
                if checkboxX.checked=true then
                        begin
                                if (t>=start[i]) and (t<=start[i]+duration[i]) then
                                        begin
                                                inpi:='1';
                                        end;
                        end;
```

Το πρόγραμμα σε αυτό το σημείο δημιουργεί σε μορφή λέξης την ακολουθία ενεργών και ανενεργών καναλιών που έχει επιλέξει ο χρήστης:

```
                mymess:=Format('%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%s%
                s%s%s%s%s',['B.',inp1,'.',inp2,'.',inp3,'.',inp4,'.',inp5,'.',inp6,'.',inp7,'.',inp8,'.',inp9,
                '.',inp10,'.',inp11,'.',inp12,'.',inp13,'#']);
                Serial1.WriteData(mymess);
                Timer1Timer(Timer1);
        end;

t:=0;
timer1.enabled:=false;
Serial1.WriteData('B.0.0.0.0.0.0.0.0.0.0.0.0.0#');
Serial1.SynSer.Flush;
End;

procedure TForm2.ButtonXClick(Sender: TObject);
var c3,k3:char;
var d3:string[28];
var i3:integer;
        begin
                Serial1.Device:=ComboBoxX.text;
                Serial1.Active:=True;
```

Διαδικασία αρχικοποίησης:

```
k3:='C';
c3:='B';
d3:='B.0.0.0.0.0.0.0.0.0.0.0.0#';
Serial1.WriteData('B.0.0.0.0.0.0.0.0.0.0.0.0#');
d3:=Serial1.ReadData;
k3:=d3[1];
if k3=c3 then
```

Διαδικασία ένδειξης ενεργής ή ανενεργής σύνδεσης περιβάλλοντος διεπαφής και Arduino Uno (main buttons contro)l:

```
                Begin
                        ButtonZ.Enabled:=True;
                        ButtonY.Enabled:=False;
                        ButtonX.Enabled:=True;
                        ComboBox2.Enabled:=False;
                        panelX.Color:=clGreen;
                        panelX.Caption:='Connected';
                        Exit;
                end;
        i3:=0;
        for i3:=1 to 100 do
                begin
                        d3:=Serial1.ReadData;
                        k3:=d3[1];
                        if k3=c3 then
                                begin
                                        ButtonZ.Enabled:=True;
                                        ButtonY.Enabled:=False;
                                        ButtonX.Enabled:=True;
                                        ComboBoxX.Enabled:=False;
                                        panelX.Color:=clGreen;
                                        panelX.Caption:='Connected';
                                        Exit;
                                end;
                end;
    end;

procedure TForm2.Button4Click(Sender: TObject);
    begin
```

```
            Serial1.WriteData('B.0.0.0.0.0.0.0.0.0.0.0#');
            Serial1.Active:=false;
```

Main buttons control:

```
            ButtonZ.Enabled:=False;
            ButtonY.Enabled:=True;
            ButtonX.Enabled:=False;
            ComboBoxY.Enabled:=True;
```

Main panels control:

```
            panelX.Color:=clRed;
            panelX.Caption:='Not Connected';
```

Επικοινωνία Lazarus – Arduino μη εφικτή:

```
            Serial1.Active:=False;
            Serial1.SynSer.Flush;
    end;
```

Παρακάτω παρουσιάζεται η διαδικασία μέσω της οποίας μετατρέπεται ο επιθυμητός χρόνος που εισάγει ο χρήστης πατώντας τα πλήκτρα στα ΕΝΑΡΞΗ και ΔΙΑΡΚΕΙΑ σε δεδομένα τα οποία μπορεί αντιληφθεί το πρόγραμμα και να τα αντιστοιχήσει στις κατάλληλες μεταβλητές των πινάκων ts kai td:

```
procedure TForm2.TimerXTimer(Sender: TObject);
    begin
        t:=t+1;
    end;
```

Διαδικασία μετατροπής τιμών χρόνων εισόδου από χρήστη στο string:

```
procedure TForm2.UpDownXClick(Sender: TObject; Button: TUDBtnType);
    begin
        labelX.caption:=inttostr(upDownX.Position)
    end;
```

Ορισμός διαδικασίας swap. Η swap χρησιμοποιείται εντός της διαδικασίας bubbleshort για την ταξινόμηση των πινάκων:

```
procedure TForm2.swap(var one: Integer; var two: Integer);
var three:integer;
        begin
                three:=0;
                three:=one;
                one:=two;
                two:=three;
        end;
end.
```

2.3.1.2) ΚΩΔΙΚΑΣ ARDUINO - ΣΧΟΛΙΑΣΜΟΣ ΚΩΔΙΚΑ

Ο κώδικας που αναπτύχθηκε στο περιβάλλον Arduino IDE συμβάλλει στη σύνδεση και στην επικοινωνία του μικροεπεξεργαστή με τον Ηλεκτρονικό Υπολογιστή, μέσω ενός καλωδίου USB, για τη μεταφορά των εισαγόμενων δεδομένων από το χρήστη.

Η δήλωση των μεταβλητών (identifiers declaration) του προγράμματος εμπεριέχει όλες τις αντιστοιχήσεις των καναλιών του Arduino και για το λόγο αυτό χρήζει ιδιαίτερης προσοχής:

```
String inData;
String
sub_out_1,sub_out_2,sub_out_3,sub_out_4,sub_out_5,sub_out_6,sub_out_7,sub_out_8,sub_
out_9,sub_out_10,sub_out_11,sub_out_12,sub_out_13;
```

Κάθε κανάλι του Arduino αποτελεί και μία ξεχωριστή μεταβλητή για το πρόγραμμα. Επιλέχθηκε η ακόλουθη λογική :

```
int outPinX = X;
```

Η τιμή X μπορεί να λάβει τιμές από 1 έως και 13 (όσα και τα κανάλια του μικροελεγκτή).

Στη συνέχεια αναπτύσσεται το κύριο μέρος του κώδικα (main body of the program).

```
void setup()
{
Serial.begin(115200);                    //9600
}
```

Παραπάνω φαίνεται η επιλογή του εύρους των bits, μια τιμή που καλείται να επιλεχθεί από το χρήστη βάση του όγκου των δεδομένων που θα κληθεί να χειριστεί το πρόγραμμα.

Η επικοινωνία του περιβάλλοντος διεπαφής και του μικροελεγκτή (serial connection procedure) επιτυγχάνεται με τη χρήση της ακόλουθης διαδικασίας. Τ

```
void loop()
        {
        if (Serial.available() > 0)
                {
                char received = Serial.read();
                inData += received;
                if (received == '#')
                        {
                        if (inData.startsWith("B.") )   //&& inData.endsWith("#")
                                {
```

Στην οθόνη του Η/Υ κρίθηκε χρήσιμη η εμφάνιση της λέξης που αποστέλλεται για την ενεργοποίηση των καναλιών ώστε ο χρήστης να είναι σε θέση να ελέγχει τη σωστή λειτουργία και επικοινωνία των επιμέρους συνιστωσών της διάταξης (Serial.println ("Correct String")).:

```
Serial.println (inData);
```

Κρίσιμο σημείο κατά τη υλοποίηση αποτέλεσε η ορθή δήλωση της αντιστοιχίας των καναλιών του Arduino και των θέσεων της σειριακής λέξης που αποστέλλεται κατά τη διαδικασία εκτέλεσης:

```
sub_out_1= inData.substring(2,3);
sub_out_2= inData.substring(4,5);
sub_out_3= inData.substring(6,7);
sub_out_4= inData.substring(8,9);
sub_out_5= inData.substring(10,11);
sub_out_6= inData.substring(12,13);
sub_out_7= inData.substring(14,15);
sub_out_8= inData.substring(16,17);
sub_out_9= inData.substring(18,19);
sub_out_10= inData.substring(20,21);
sub_out_11= inData.substring(22,23);
sub_out_12= inData.substring(24,25);
sub_out_13= inData.substring(26,27);
```

Στη συνέχεια μέσω μιας εντολής ελέγχου (if-else loop) πρόγραμμα καθορίζει βάση των δεδομένων που θα λάβει από το πρόγραμμα του περιβάλλοντος διεπαφής σε ποια τιμή θα αντιστοιχίσει την αντίστοιχη θύρα του Arduino. Συγκεκριμένα αν από την ανάλυση της λέξης ('B. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1. 0ή1#) που θα λάβει ένα κανάλι που έχει τιμή 1 θα αντιστοιχηθεί με τη τιμή HIGH (υψηλή τάση), ενώ αν έχει τιμή 0 με τη τιμή LOW (χαμηλή τάση). Παρακάτω παρατίθεται ένα χαρακτηριστικό παράδειγμα της εντολής που χρησιμοποιήθηκε για το κανάλι X:

```
if (sub_out_X =="1") {digitalWrite(outPinX,HIGH); } else
{digitalWrite(outPinX,LOW); }
```

Στο τέλος εκτέλεσης του κύκλου κάθε προγράμματος πραγματοποιείται εκκαθάριση του buffer:

```
            inData = ""; // Clear recieved buffer
            }
        }
}
```

Ο κώδικας που παρουσιάστηκε συμβάλλει στην επίτευξη μιας μείζονος σημασίας λειτουργίας της όλης διάταξης: την επικοινωνία του μικροεπεξεργαστή με το περιβάλλον του Lazarus, και άρα με το χειριστή. Αυτή δεν γίνεται αυτόματα, αλλά απαιτείται ένα communication portal. Μέσω αυτού, λοιπόν, μετατρέπονται τα δεδομένα που εισάγονται (χρόνοι εκκίνησης καναλιών, χρόνοι διάρκειας λειτουργίας καναλιών) σε μία σειριακή λέξη της μορφής B.X.X.X.X.X.X.X.X.X.X.X.X.X.X. (όπου X= 0 αν το κανάλι παραμένει απενεργοποιημένο και X= 1 αν το κανάλι είναι ενεργό) και με αυτό τον τρόπο το Arduino αντιλαμβάνεται τα δεδομένα που του παρέχονται και δίνει με τη σειρά του τις κατάλληλες εντολές. Επίσης πρέπει να αναφερθεί ότι η επικοινωνία αυτή δεν είναι στατική, αλλά ανανεώνεται με τη πάροδο κάποιας προκαθορισμένης χρονικής διάρκειας (fixedtime), με συνέπεια την ορθή λειτουργικότητα της διάταξης.

ΚΕΦΑΛΑΙΟ 3^{Ο}

ΑΠΟΤΕΛΕΣΜΑΤΑ – ΣΥΜΠΕΡΑΣΜΑΤΑ – ΜΕΛΛΟΝΤΙΚΕΣ ΕΦΑΡΜΟΓΕΣ ΣΥΣΤΗΜΑΤΟΣ – ΒΕΛΤΙΩΣΕΙΣ

Στο συγκεκριμένο κεφάλαιο δίνεται ο επίλογος, με μία σύνοψη, ανασκόπηση των θεμάτων που μελετήθηκαν στην εργασία και προτάσεις για μελλοντικές επεκτάσεις.

3.1) ΑΠΟΤΕΛΕΣΜΑΤΑ

Ο στόχος της εργασίας ολοκληρώθηκε επιτυχώς και αυτός πρωτίστως ήταν η μελέτη διαφόρων τεχνολογιών και η διασύνδεση τους ώστε να καταφέρουμε να κατασκευάσουμε ένα σύμπλεγμα υποσυστημάτων που θα λειτουργούν αρμονικά και θα υλοποιούν σε πραγματικό χρόνο την λογική του συγχρονισμού μετρήσεων. Προκειμένου να φτάσουμε στο επιθυμό αποτέλεσμα έγιναν πολλαπλές δοκιμές και τροποποιήσεις ώστε να πετύχουμε την καλύτερο δυνατό αποτέλεσμα, χωρίς αυτό να σημαίνει ότι οι δυνατότητες για μελλοντικές επεκτάσεις και βελτιώσεις είναι λίγες. Άξιο αναφοράς είναι ότι τα παραπάνω έγιναν εφικτά με χρήση εργαλείων ανοιχτού κώδικα διαθέσιμα σε όλους μας και με μικρό έως μηδενικό κόστος.

3.2) ΣΥΜΠΕΡΑΣΜΑΤΑ – ΒΕΛΤΙΩΣΕΙΣ

3.2.1) ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΜΕΛΛΟΝΤΙΚΗ ΑΞΙΟΠΟΙΗΣΗ

Κατά τη διαδικασία και διάρκεια εκπόνησης της διπλωματικής αυτής εργασίας έγινε άμεσα αντιληπτό το εύρος και η δυνατότητα των εφαρμογών που δύναται να βασιστούν σε μικροελεγκτές και ως εκ τούτου στην πλατφόρμα του Arduino.

Η συγκεκριμένη διάταξη συγχρονισμού μετρήσεων παρέχει τη δυνατότητα στο χρήστη συντονισμού μετρήσεων με ακρίβεια, ευελιξία και προπάντων αποτελεσματικότητα. Εφόσον επιτευχθεί η ομαλή διασύνδεση των επιμέρους συνιστωσών της διάταξης με τον ηλεκτρονικό υπολογιστή, η επίβλεψη της όλης διαδικασίας γίνεται πλέον εύχρηστα και δίχως κόπο.

Ταυτόχρονα η επιλογή και η αξιοποίηση του μικροελεγκτή Arduinoανοίγει διάπλατα το φάσμα της περαιτέρω αξιοποίησης αυτής της διάταξης στο μέλλον. Ειδικότερα:

Α) Η χρήση άλλων εκδόσεων του Arduino (π.χ. ArduinoMega) θα μπορούσε να οδηγήσει στην αύξηση των αναλογικών και ψηφιακών I/O.

Β) Η εφαρμογή διεπαφής με το χρήστη δύναται να εμπλουτιστεί με επιλογές επεξεργασίας των μετρήσεων με δυνατότητα εμφάνιση διαγραμμάτων και στατιστικών.

Γ) Η επικοινωνία των μερών της διάταξης θα μπορούσε να πραγματοποιηθεί ασύρματα, χωρίς την ανάγκη χρήσης καλωδίων, συμβάλλοντας με αυτό τον τρόπο στην απλούστευση της.

3.2.2) ΒΕΛΤΙΩΣΕΙΣ

Η εργασία αυτή έχει το περιθώριο για κάποιες βελτιώσεις και επεκτάσεις οι οποίες παρατίθενται παρακάτω:

Α) Η βελτίωση και επέκταση της διαδικασίας των μετρήσεων. Αυτό αφορά στην εναρμόνιση της καταγραφής, της αποθήκευσης και της παρουσίασης των μετρήσεων σύμφωνα με τα διεθνή πρότυπα.

Β) Η διεπαφή χρήστη (LAZARUS) να μπορούσε να δεχτεί αρκετές βελτιώσεις όσον αφορά την ευχρηστία και τις δυνατότητες παραμετροποίησης/αρχικοποίησης της εφαρμογής. Επιδέχεται επίσης αρκετές αισθητικές βελτιώσεις, καθώς σχεδιάστηκε αρκετά μινιμαλιστικά για να εξυπηρετήσει τις δοκιμές και τους ελέγχους καλής λειτουργίας του συστήματος.

Γ) Η δημιουργία γραφικού περιβάλλοντος διεπαφής για την παρουσίαση των μετρήσεων. Η υλοποίηση της εφαρμογής στην πλατφόρμα του Arduino, δίνει τη δυνατότητα για εύκολη ανάπτυξη ενός τέτοιου προγράμματος. Γενικότερα, οι δυνατότητες της πλατφόρμας του Arduino προσφέρουν τη ευελιξία για μεγαλύτερη ανάπτυξη των διεπαφών του συστήματος.

Δ) Η ανάπτυξη της διαδραστικότητας της εφαρμογής με το χρήστη με την χρήση διάφορων περιφερειακών συσκευών.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Kenneth C. Smith, Adel S. Sedra, Microelectronic Circuits (The Oxford Series Electrical and Computer Engineering), 6th ed., Oxford University Press, 2009.

[2] Κ.Ζ. Πεκμεστζή, Συστήματα Μικροϋπολογιστών, 2nded., Εκδόσεις Συμμετρία, 1995.

[3] Massimo Banzi, Getting Started with Arduino, 2nd ed., O'Reilly Media/MAake, 2011.

[4] John Davies, MSP430 Microcontroller Basics, 1st ed., Newnes, 2008.

[5] Michael Margolis, Arduino Cookbook, 2nd ed., O'Reilly Media, 2011.

[6] Michael McRoberts, Beginning with Arduino, 1st ed., Apress, 2010.

[7] Simon Monk, 30 Arduino Projects for the evil Genius, 1st ed., McGraw-Hill/TAB Electronics, 2010.

[8] Arduino Official. {Online}. www.arduino.com

[9] Atmel ATmega328. {Online}. www.atmel.com

[10] Hyunwoo Nam, Jan Janak, Henning Schulzrinne, Connecting the Physical World with Arduino in SECE, Department of Computer Science, Columbia University New York.

[11] Alessandro D'Ausilio Arduino, A low-cost multipurpose lab equipment, Psychonomic Society, 2011.

[12] Riccardo Ortolan, Software Engineering of Arduino based art systems, Norwegian University of Science and Technology, 2011.

[13] P. Lengvenis, R. Maskeliunas, V. Raudonis, Arduino based Controller for the Smart Assitive Mobility Hardware, Department of Process Control, Kaunas University of Technology, 2012.

[14] Jeffrey A. Kornuta, Matthew E. Nipper, J. Brandon Dixon, Low-cost microcontroller platform for studying lymphatic biomechanics in vitro, Woodruff School of Mechanical Engineering, Georgia Institute of Technology, Atlanta, GA 30332, United States, Journal of Biomechanics, 2012.

[15] www.microchip.com

[16] www.lazarus.com

[17] Sheikh Ferdoush, Xinrong Li, Department of Electrical Engineering, University of North Texas, Denton, Texas, Wireless Sensor Network System Design using Raspberry Pi and Arduino for Environmental Monitoring Applications, The 9th International Conference on Future Networks and Communications, 2014.

[18] Muhammad Abu Bakar Sidik, Mohd Qamarul Arifin Rusli, Zuraimy Adzis, Zolkafle Buntat, Yanuar Zulardiansyah Arief, Hamizah Shahroom, Zainuddin Nawawi, Muhammad Irfan Jambak, Institute of High Voltage and High Current (IVAT), Universiti Teknologi Malaysia & Department of Electrical Engineering, Faculty of Engineering, Universitas Sriwijaya, Indonesia, Arduino-Uno Based Mobile Data Logger with GPS Feature, TELKOMNIKA, Vol.13, No.1, March 2015.

[19] www.eln.teilam.gr/sites/default/files/Lesson03.pdf

[20] www.microplanet.gr/tutorials/microcontrollers/arduino

[21] Lazarus official website, www.lazarus.freepascal.org

[22] R. Northrop, Introduction to Instrumentation and Measurements, CRC Press, 2005.

[23] J. Noble, Programming Interactivity, O'Reilly Media, 2009.

[24] Ν. Θεοδώρου, Ηλεκτρικές Μετρήσεις: Κλασσικές, Ηλεκτρονικές και Ψηφιακές, Εκδόσεις Συμμετρία, 2016.

[25] Fabio Varesano, Using Arduino for Tangible Human Computer Interaction, University of Torino, April 2011.

[26] Σπύρος Φιορέτος, Διπλωματική Εργασία: Έλεγχος ανοικτού βρόχου βηματικού κινητήρα μέσω του μικροελεγκτή Arduino, Μάρτιος 2015.

[27] Χατζόπουλος Γεώργιος, Χατζόπουλος Γιάννης, Διπλωματική Εργασία: Απομακρυσμένος έλεγχος ασύγχρονης μηχανής μέσω σύντομων γραπτών μηνυμάτων (sms) με Arduino, 2014.

[28] Ισίδωρος Κόλλιας, Διπλωματική Εργασία: Κατασκευή Τροφοδοτικού 0-30V, 0-5A με Δυνατότητα Συμμετρικής Λειτουργίας, Μάρτιος 2014.

[29] Αλέξανδρος Φ. Τουλουζας, Διπλωματική Εργασία: Σύστημα Μέτρησης Διηλεκτρικής Σταθεράς Βασισμένο σε Μικροελεγκτή, Σεπτέμβριος 2003.