ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΔΥΤΙΚΗΣ ΕΛΛΑΔΑΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ Τ.Ε.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ 1565

ΣΧΕΔΙΑΣΗ ΚΑΙ ΥΛΟΠΟΙΗΣΗ ΕΝΟΣ ΜΟΥΣΙΚΟΥ ΚΟΥΤΙΟΥ (JUKEBOX) ΜΕ ΔΥΝΑΤΟΤΗΤΑ ΑΝΑΠΑΡΑΓΩΓΗΣ ΨΗΦΙΑΚΩΝ ΑΡΧΕΙΩΝ ΜΟΥΣΙΚΗΣ (MP3) ΣΕ ΜΙΚΡΟΕΛΕΓΚΤΗ ARDUINO ΜΕ ΤΟΠΙΚΟ ΚΑΙ ΑΠΟΜΑΚΡΥΣΜΕΝΟ ΕΛΕΓΧΟ.

ΣΠΟΥΔΑΣΤΗΣ: ΜΟΥΛΙΟΣ ΓΡΗΓΟΡΗΣ (6208)

ΕΠΟΠΤΕΥΩΝ ΚΑΘΗΓΗΤΗΣ: ΤΟΠΑΛΗΣ ΕΥΑΓΓΕΛΟΣ

ΠΑΤΡΑ 2016

Ευχαριστίες

Πρώτα απ' όλα, θα ήθελα να ευχαριστήσω τον εισηγητή της πτυχιακής εργασίας μου, Καθηγητή κ. Ευάγγελο Τοπάλη, για την βοήθεια και καθοδήγηση καθ' όλη τη διάρκεια της εκπόνησης της εργασίας μου. Επίσης, ευχαριστώ τον Καθηγητή κ. Λουκά Χαδέλλη που μου έδωσε την ευκαιρία να κάνω την πρακτική μου άσκηση στο εργαστήριο Μικροϋπολογιστικών Συστημάτων του Τμήματος, εμπειρία που με βοήθησε στο να φέρω εις πέρας την πτυχιακή εργασία. Θα ήθελα να τον ευχαριστήσω μαζί με τον Καθηγητή κ. Βασίλειο Καψάλη επίσης για την προσεκτική ανάγνωση της εργασίας μου και για τις υποδείξεις τους, ως μέλη της εξεταστικής επιτροπής της πτυχιακής. Οφείλω ευχαριστίες στον Καθηγητή κ. Μίμο Ευάγγελο για την βοήθεια του στον προγραμματισμό του Arduino. Τέλος θα ήθελα να ευχαριστήσω τους γονείς μου για την υποστήριξη τους όλα αυτά τα χρόνια. Αφιερώνω αυτή την εργασία στην μητέρα μου και στον πατέρα μου.

Γρηγόρης Μούλιος

ΠΕΡΙΛΗΨΗ

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η δημιουργία ενός jukebox, μέσω της πλατφόρμας Arduino. Ο έλεγχος του θα επιτυγχάνεται σε τοπικό επίπεδο και θα υπάρχει δυνατότητα σύνδεσης στο διαδίκτυο για να γίνεται απομακρυσμένος έλεγχος και τη συλλογή πληροφοριών όπως θερμοκρασία τοποθεσία κλπ.

Πιο συγκεκριμένα, για την υλοποίηση του συστήματος θα χρησιμοποιηθεί το Arduino Mega 2560, το οποίο είναι βασισμένο στον μικροελεγκτή της Atmel Atmega 2560. Το Arduino είναι μία πλατφόρμα ανάπτυξης εφαρμογών ανοιχτού κώδικα και περιλαμβάνει μία εύχρηστη πλακέτα δωρεάν λογισμικό για τον προγραμματισμό του μικροελεγκτή σε γλώσσα Wiring C. Το λογισμικό που χρησιμοποιήθηκε για τον προγραμματισμό του Arduino είναι το Arduino IDE, το οποίο είναι γραμμένο σε γλώσσα java και είναι βασισμένο στο IDE για τη γλώσσα προγραμματισμού Processing.

Στην πλακέτα Arduino Mega 2560 έχουν συνδεθεί διάφορες επεκτάσεις(shields) οι οποίες δίνουν δυνατότητες που δεν παρέχει η πλατφόρμα από μόνη της. Χρησιμοποιήθηκε επέκταση για την αποκωδικοποίηση αρχείων ήχου και αναπαραγωγή τους και μία επέκταση η οποία δίνει δυνατότητα σύνδεσης με το διαδίκτυο. Αρχικά, ο τοπικός έλεγχος θα γινόταν μέσω κανονικών κουμπιών αλλά στην πορεία αυτά τα κουμπιά εξομοιώθηκαν σε μία οθόνη αφής.

Τέλος, δημιουργήθηκε ιστοσελίδα όπου το jukebox θα μπορεί να ελέγχεται απομακρυσμένα μέσω κάποιου υπολογιστή ή smartphone, διευκολύνοντας έτσι τον χρήστη.

Περιεχόμενα

Κεφάλαιο 1: Εισαγωγή

1.1 Γενικά

Αντικείμενο της παρούσας πτυχιακής εργασίας είναι η δημιουργία ενός jukebox, μέσω της πλατφόρμας Arduino. Ο έλεγχος του θα επιτυγχάνεται σε τοπικό επίπεδο με τη χρήση μίας οθόνης αφής και θα υπάρχει δυνατότητα σύνδεσης στο διαδίκτυο για να γίνεται απομακρυσμένος έλεγχος και τη συλλογή πληροφοριών όπως θερμοκρασία τοποθεσία κλπ.

Αρχικά θα γίνει μία ιστορική αναδρομή στα βασικότερα μέσα αναπαραγωγής μουσικής που ενέπνευσαν αυτή την εργασία και στην κυρίαρχη κωδικοποίηση ήχου που θα χρησιμοποιηθεί, της MPEG Audio Layer III ευρύτερα γνωστό ως MP3, και τα πλεονεκτήματα της. Στη συνέχεια θα γίνει εκτεταμένη αναφορά στους μικροελεγκτές Arduino και το πρόγραμμα που χρησιμοποιήθηκε για τον προγραμματισμό τους, και την γλώσσα σήμανσης HTML που χρησιμοποιήθηκε για τον προγραμματισμό διαδικτύου.

Ακολουθεί το βασικό μέρος της πτυχιακής εργασίας στο οποίο θα αναλυθεί το υλικό και το λογισμικό μέρος του συστήματος που υλοποιήθηκε. Συγκεκριμένα θα γίνει αναφορά στον μικροελεγκτή Arduino Mega 2560 που χρησιμοποιήθηκε στην εργασία και στα περιφερειακά του που έκαναν δυνατή την αναπαραγωγή μουσικής και τη σύνδεση με το διαδίκτυο καθώς και στον προγραμματισμό τους και τις απαιτούμενες συνδέσεις τους για την σωστή λειτουργία του τελικού κυκλώματος.

Τέλος θα γίνει ανάλυση του τελικού αποτελέσματος και θα γίνει αναφορά σε πιθανές μελλοντικές επεκτάσεις και βελτιώσεις του συστήματος που υλοποιήθηκε.

1.2 Η κωδικοποίηση MP3 και η ιστορία του

Το mp3 ή στην πλήρη ονομασία του, MPEG Audio Layer III, είναι μία μορφή κωδικοποίησης ήχου σε ψηφιακή μορφή το οποίο μας δίνει τη δυνατότητα να έχουμε ένα αρχείο μουσικής μικρό σε μέγεθος με λίγη ή χωρίς καθόλου απώλεια στην ποιότητα του ήχου. Το mp3 είναι μέρος της MPEG οικογένειας της οποίας τα αρχικά σημαίνουν Moving Pictures Experts Group που σχετίζεται με τη χρήση της συμπίεσης lossy σε βίντεο και ήχο.

Το 1987 το γερμανικό ερευνητικό κέντρο Fraunhofer Institut Integrierte Schaltungen (μέρος της εταιρείας Fraunhofer-Gesellschaft) ξεκίνησε το “EUREKA project EU147, Digital Audio Broadcasting (DAB)” με σκοπό να πετύχει μία κωδικοποίηση ήχου υψηλής ποιότητας αλλά με χαμηλό ρυθμό μετάδοσης δεδομένων (bit-rate).

Το 1997 και μετά από την αποτυχημένη προσπάθεια της Fraunhofer στις αρχές της δεκαετίας του 1990 ο προγραμματιστής Tomislav Uzelac της εταιρείας Advanced Multimedia Products εφηύρε το AMP MP3 Playback Engine, πρώτο MP3 Player. Αργότερα δύο φοιτητές, οι Justin Frankel και Dmitry Boldyrev, συνέδεσαν το AMP με τα Windows και δημιούργησαν την εφαρμογή Winamp. Το 1998 το Winamp έγινε ένα δωρεάν MP3 Player και σε συνδυασμό με τη μη απαίτηση κατοχής αδειών για την χρησιμοποίηση του, η επιτυχία της MP3 κωδικοποίησης γιγαντώθηκε.

1.3 Οι συσκευές αναπαραγωγής ήχου

- Το 1857 ο Leon Scott De Martinville δημιουργεί τον φωνοαυτογράφο, την πρώτη συσκευή ικανή να ηχογραφήσει οποιονδήποτε ήχο χωρίς να μπορεί να τον αναπαράξει όμως.
- Το 1877 ο Thomas Edison εφηύρε την πρώτη συσκευή ικανή να ηχογραφήσει και να αναπαράξει την ανθρώπινη φωνή, τον φωνογράφο, που χρησιμοποιούνταν αποκλειστικά για επαγγελματικούς σκοπούς.
- Το 1887 ο Emile Berliner εξέλιξε την ιδέα του Martinville σε μία συσκευή αναπαραγωγής δίσκων μουσικής, το γραμμόφωνο.
- Το 1954 με τη βοήθεια της εφεύρεσης του τρανζίστορ δημιουργήθηκαν τα ραδιόφωνα τρανζίστορ τα οποία θα άλλαζαν για πάντα τον τρόπο με τον οποίο οι άνθρωποι θα άκουγαν μουσική. Ο άνθρωπος πλέον μπορούσε να ακούσει μουσική οπουδήποτε, εισάγοντας αυτή τη δυνατότητα στην ψυχοσύνθεση του ανθρώπου με αποτέλεσμα δισεκατομμύρια τρανζίστορ να φτιαχτούν τις δεκαετίες του 1970-1980.
- Το 1958 ο μουσικόφιλος John Koss ανακάλυψε τα πρώτα στερεοφωνικά ακουστικά που ήταν τα πρώτα που δημιουργήθηκαν αποκλειστικά για την αναπαραγωγή μουσικής. Πλέον η μουσική απέκτησε ένα πιο προσωπικό ύφος.
- Το 1962 ο Henry Kloss έφτιαξε το πρώτο φορητό stereo αναπαραγωγής δίσκων.
- Το 1966 και από τον δημιουργό της audio compact κασέτας δημιουργήθηκαν τα πρώτα φορητά κασετόφωνα γνωστά στην πορεία και ως “Boom Box” τα οποία ήταν μεγάλο κομμάτι της “pop” κουλτούρας κατά τις δεκαετίες του 1980-1990.
- Το 1979 η Sony έφερε στην καθημερινότητα μας το πρώτο πραγματικά αυτοδύναμο φορητό μουσικό σύστημα. Η πραγματικά μεγάλη καινοτομία του “Walkman” ήταν το μέγεθος του το οποίο ήταν λίγο μεγαλύτερο από αυτό της κασέτας. Σε συνδυασμό με την χρησιμοποίηση AA μπαταριών και ελαφριών ακουστικών το “Walkman” άνοιξε το δρόμο για πολλές καινοτομίες στο χώρο των φορητών συσκευών.
- Το 1982 η Sony ξανά, δημιούργησε το “Discman” το οποίο ήταν το πρώτο φορητό σύστημα που μπορούσε να αναπαράγει μουσική που ήταν σε CD.
- Το 1998 ξεκίνησε η εποχή των φορητών μουσικών συσκευών που χρησιμοποίησαν την mp3 κωδικοποίηση η οποία επικρατεί μέχρι σήμερα. Οι δύο πρώτες εταιρείες ήταν η Αμερικάνικη Diamond Multimedia με το μοντέλο “Rio 100” και η Κορεάτικη Saehan Information Systems με το μοντέλο “MPMAN”. Ήταν οι πρώτες φορητές συσκευές αναπαραγωγής μουσικής που χρησιμοποίησαν μνήμη flash για την αποθήκευση και αναπαραγωγή mp3 αρχείων.

1.4 Ιδέα- Σύστημα προς υλοποίηση

Η αρχική ιδέα ήταν δημιουργία ενός συστήματος αναπαραγωγής μουσικής σε MP3 κωδικοποίηση (MP3 Player) με τον μικροελεγκτή Arduino και τον χειρισμό του είτε με τη χρήση εξωτερικών κουμπιών είτε με την εξομοίωση των κουμπιών αυτών σε οθόνη αφής. Αργότερα με την βοήθεια διάφορων επεκτάσεων του Arduino η ιδέα μου εξελίχτηκε κρατώντας μεν την βασική ιδέα αλλά απέκτησε και έναν χαρακτήρα συσκευής πολυμέσων με την προσθήκη ενός σταθμού καιρού (weather station) και τη

δυνατότητα να ακολουθεί κάποιος τα social media ή κάποια ιστοσελίδα. Επίσης μέσω ενός υπολογιστή ή κινητού τηλεφώνου με πρόσβαση στο internet υπάρχει η δυνατότητα να ελέγξει το σύστημα μέσω μίας ιστοσελίδας στο internet.

Κεφάλαιο 2: Τεχνολογίες και Εργαλεία Ανάπτυξης που χρησιμοποιήθηκαν.

2.1 Μικροελεγκτές

Ο μικροελεγκτής είναι ένα σύστημα σε ολοκληρωμένο κύκλωμα με ενσωματωμένη μνήμη και θύρες Εισόδου-Εξόδου, σε αντίθεση με τους μικροεπεξεργαστές που συνδέονται ως περιφερειακά κυκλώματα. Συνήθως ένας μικροελεγκτής χρησιμοποιείται για τρείς σκοπούς,

- Να συλλέγει δεδομένα από κάποιο άλλο μέσο.
- Να αποθηκεύει και να επεξεργάζεται αυτά τα δεδομένα.
- Να εφαρμόζει τα επεξεργασμένα δεδομένα και μέσω κάποιων εργασιών να εξάγει τα καινούρια πια δεδομένα.

Συναντάμε τους μικροελεγκτές σε προϊόντα και συσκευές αυτομάτου ελέγχου στην βιομηχανία αλλά και στην καθημερινότητα, για παράδειγμα σε τηλεχειριστήρια, παιχνίδια, ηλεκτρικές συσκευές και άλλα συστήματα.

Οι περισσότεροι μικροελεγκτές βασίζονται στην αρχιτεκτονική Von Neumann, η οποία καθόρισε σαφώς τα τέσσερα βασικά συστατικά που απαιτούνται για ένα ψηφιακό σύστημα. Αυτά περιλαμβάνουν έναν επεξεργαστικό πυρήνα (CPU), τη μνήμη για το πρόγραμμα και τα δεδομένα (RAM), χώρο μόνιμης αποθήκευσης (FLASH σε έναν Μικροελεγκτή), καθώς επίσης και τις θύρες I/O για επικοινωνία με εξωτερικές περιφερειακές μονάδες - όλο αυτά σε ένα ενιαίο ολοκληρωμένο κύκλωμα.

Οι μικροελεγκτές συνήθως περιλαμβάνουν θύρες I/O για σύνδεση με ποικίλες συσκευές εισόδου/εξόδου, όπως αναλογικές (με ενσωματωμένους μετατροπείς από αναλογικό σε ψηφιακό), UARTs ή ειδικευμένα σειριακά interface επικοινωνιών όπως το PC, το SPI και το Controller Area Network (CAN). Μερικοί σύγχρονοι μικροελεγκτές έχουν ενσωματωμένη και μία υψηλού επιπέδου γλώσσα προγραμματισμού όπως η BASIC.

2.1.1 Arduino

Το Arduino δημιουργήθηκε στο πανεπιστήμιο της Ιβρέα στην Ιταλία το 2005 ως ένα εργαλείο ανάπτυξης εφαρμογών, για φοιτητές χωρίς παλαιότερη ενασχόληση με τον τομέα των ηλεκτρονικών και του προγραμματισμού. Είναι μία πλατφόρμα ανάπτυξης εφαρμογών (φυσικού υπολογισμού) ανοιχτού κώδικα, δίνοντας τη δυνατότητα στον χρήστη να κάνει όποια προσαρμογή χρειάζεται για να καλύψει τις ανάγκες του. Η πλατφόρμα αυτή περιλαμβάνει μία εύχρηστη πλακέτα και δωρεάν λογισμικό για τον προγραμματισμό του μικροελεγκτή. Μέχρι το 2015 το Arduino χρησιμοποιούσε αποκλειστικά τη σειρά μικροελεγκτών megaAVR της Atmel. Η γλώσσα προγραμματισμού είναι η

Wiring C μία απλοποιημένη έκδοση της γλώσσας C++ και το περιβάλλον ανάπτυξης είναι βασισμένο στο project Processing. Υπάρχουν πολλές πλατφόρμες μικροελεγκτών φυσικού υπολογισμού αλλά το Arduino παρέχει πλεονεκτήματα στους χρήστες του, καθηγητές, φοιτητές και αρχάριους σε σχέση με τις υπόλοιπες πλατφόρμες. Τα βασικά πλεονεκτήματα του είναι τα εξής (http://arduino.cc) :

- Χαμηλό κόστος. Οι πλακέτες Arduino είναι φθηνότερες από τις υπόλοιπες πλατφόρμες μικροελεγκτών με το κόστος να μην ξεπερνά τα 50 € σε κανένα από τα προϊόντα και έχοντας τη δυνατότητα με πολύ λίγα χρήματα να κατασκευάσεις το δικό σου.

- Συμβατότητα. Το λογισμικό του Arduino είναι συμβατό σε Windows, Macintosh και Linux.
- Απλό προγραμματιστικό περιβάλλον. Το λογισμικό του Arduino είναι αρκετά εύχρηστο για να βοηθήσει τους αρχάριους αλλά και με αρκετές δυνατότητες για τους πιο έμπειρους προγραμματιστές.
- Επεκτάσιμο λογισμικό ανοιχτού κώδικα. Τα προγράμματα του Arduino είναι ανοιχτού κώδικα που σημαίνει ότι οποιοσδήποτε μπορεί να βρει τμήματα πηγαίου κώδικα, να τα μελετήσει και να τα τροποποιήσει σύμφωνα με τις ανάγκες του. Επίσης, μέσα από την προσθήκη βιβλιοθηκών C++ μπορούν να προστεθούν δυνατότητες στο περιβάλλον ανάπτυξης. Κάποιος πεπειραμένος χρήστης μπορεί να γράψει δικό του κώδικα χαμηλού επιπέδου και να τον καταστήσει διαθέσιμο μέσω βιβλιοθηκών, ενώ οι αρχάριοι χρήστες μπορούν να προσθέσουν υποστήριξη για νέες διατάξεις (αισθητήρες, κ.τ.λ.) χωρίς να χρειάζεται να γνωρίζουν τις χαμηλού επιπέδου λεπτομέρειες του προγραμματισμού των διατάξεων αυτών.

2.1.2 Εκδόσεις Arduino

Από το 2005 που κυκλοφόρησε το πρώτο Arduino έχουν δημιουργηθεί πολλές εκδόσεις και παραλλαγές. Ορισμένες έχουν αποσυρθεί για να τις διαδεχτούν νεότερες και πιο εξελιγμένες. Κάθε έκδοση υποστηρίζει συγκεκριμένο μικροελεγκτή και διαθέτει συγκεκριμένο αριθμό ακροδεκτών. Υπάρχουν ακροδέκτες τροφοδοσίας και ακροδέκτες εισόδου/εξόδου για σύνδεση με περιφερειακά κλπ. Το Arduino μπορεί να διαβάσει ψηφιακά σήματα (0 ή 5 V) και αναλογικά σήματα (από 0 έως 5 V).Οι έξοδοι μπορούν να δεχτούν μόνο ψηφιακά σήματα.

Οι ακροδέκτες τροφοδοσίας στα Arduino board μπορούν να τροφοδοτήσουν με 3.3 ή 5 V εξωτερικές συσκευές και ρεύμα μικρότερο από 30-40 mA. Υπάρχει δυνατότητα σύνδεσης με τον υπολογιστή μέσω θύρας USB για τον προγραμματισμό του Arduino αλλά και για την τροφοδοσία του με 5 V. Είναι δυνατόν να λειτουργήσει και χωρίς την σύνδεση με υπολογιστή αρκεί να τροφοδοτείται από την ειδική είσοδο.

Σας παραθέτω τις πιο διαδεδομένες εκδόσεις Arduino

1. Το Arduino UNO το οποίο είναι βασισμένο στον μικροελεγκτή Atmega328P της Atmel και είναι το ιδανικό board για να ξεκινήσει κάποιος. Έχει 14 ακροδέκτες ψηφιακών εισόδων/εξόδων (από τις οποίες οι 6 είναι PWM έξοδοι), 6 ακροδέκτες αναλογικών εισόδων, έχει συχνότητα 16 MHz, και 32 Kb εσωτερικής μνήμης Flash.
2. Το Arduino 101 χρησιμοποιεί τον μικροελεγκτή Intel Curie και έχει την ίδια εμφάνιση και λίστα περιφερειακών με το Arduino UNO με την προσθήκη δυνατότητας Bluetooth LE και ένα επιταχυνσιόμετρο/γυροσκόπιο 6 αξόνων. Λειτουργεί σε συχνότητα 32 MHz και έχει 196 Kb

εσωτερικής μνήμης Flash.

3. Το Arduino Pro το οποίο έχει δύο διαφορετικές εκδόσεις, η πρώτη λειτουργεί στα 3.3 V και έχει συχνότητα 8 MHz και η δεύτερη στα 5 V και έχει συχνότητα 16 MHz. Υπάρχει η επιλογή επίσης να διαλέξεις ποιόν μικροελεγκτή θα χρησιμοποιήσεις καθώς υποστηρίζεται και από τον Atmega168 αλλά και τον Atmega328 με τον πρώτο να σου δίνει 16 Kb Flash μνήμης και τον δεύτερο 32Kb. Τα περιφερειακά του είναι τα ίδια με του Arduino UNO.
4. Το Arduino Micro είναι το μικρότερο Arduino χρησιμοποιεί τον Atmega32U4 μικροελεγκτή και φτιάχτηκε σε συνεργασία με την εταιρεία Adafruit. Έχει 20 ψηφιακές εισόδους/εξόδους (από τις οποίες οι 7 μπορούν να χρησιμοποιηθούν ως PWM έξοδοι και 12 ως αναλογικές εισόδους). Λειτουργεί στα 5 V, έχει συχνότητα 16 MHz και 32 Kb μνήμης Flash.
5. Το Arduino Pro Mini το οποίο είναι μία μικρότερη έκδοση του Arduino Pro, έχει τα ίδια χαρακτηριστικά, μόνο που είναι βασισμένο αποκλειστικά στον μικροελεγκτή Atmega328 και έχει 32 Kb Flash μνήμη.
6. Το Arduino Nano το οποίο είναι ένα μικρό board με δύο εκδόσεις, η μία είναι βασισμένη στον Atmega168 και η άλλη στον Atmega328. Έχει συχνότητα 16 MHz, 14 ψηφιακές εισόδους/εξόδους εκ των οποίων οι 6 είναι και PWM έξοδοι και 8 αναλογικές εισόδους και έχει 16 Kb μνήμης Flash στην πρώτη περίπτωση και 32 Kb στη δεύτερη.
7. Το Arduino Mega 2560 είναι βασισμένο στον Atmega2560 μικροελεγκτή και είναι σχεδιασμένο για πιο πολύπλοκα projects. Έχει 54 ακροδέκτες ψηφιακών εισόδων/εξόδων (εκ των οποίων οι 15 μπορούν να χρησιμοποιηθούν και ως PWM έξοδοι), 16 ακροδέκτες αναλογικών εισόδων και 4 UART. Η συχνότητα του είναι 16 MHz και η μνήμη Flash είναι 256 Kb.
8. Το Arduino Zero είναι μία επέκταση της υπάρχουσας πλατφόρμας του Arduino UNO στα 32 bit προσφέροντας πολλές δυνατότητες και ευρεία γκάμα νέων πιθανών project. Βασίζεται στον μικροελεγκτή της Atmel SAMD21 MCU, που έχει για πυρήνα έναν 32bit ARM Cortex M0+. Επίσης έχει ενσωματωμένο τον Embedded Debugger (EBDG) της Atmel μεγαλώνοντας σημαντικά την ευκολία του εντοπισμού σφαλμάτων. Λειτουργεί αυστηρά στα 3.3 V, διαθέτει 20 ακροδέκτες ψηφιακών εισόδων/εξόδων (όλες εκτός από την 2 και την 7 μπορούν να χρησιμοποιηθούν ως PWM έξοδοι),2 ακροδέκτες UART, 6 12-bit ADC κανάλια και 1 10-bit DAC κανάλι, έχει 256 Kb Flash μνήμη και συχνότητα 48 MHz.
9. Το Arduino Due που είναι το πρώτο Arduino βασισμένο σε 32-bit ARM μικροελεγκτή. Πιο συγκεκριμένα χρησιμοποιεί τον SAM3X8E ARM Cortex-M3 CPU της Atmel. Έχει 54 ακροδέκτες ψηφιακών εισόδων/εξόδων με 12 από αυτούς να μπορούν να χρησιμοποιηθούν και ως PWM έξοδοι, 12 αναλογικές εισόδους, 4 UART, USB OTG σύνδεση, συχνότητα 84 MHz, 2 DAC κανάλια, 2 TWI, και θύρα για JTAG debugger. Επίσης έχει και 512 Kb μνήμης Flash πράγμα που το κάνει το καταλληλότερο board για μεγάλης κλίμακας εφαρμογές.

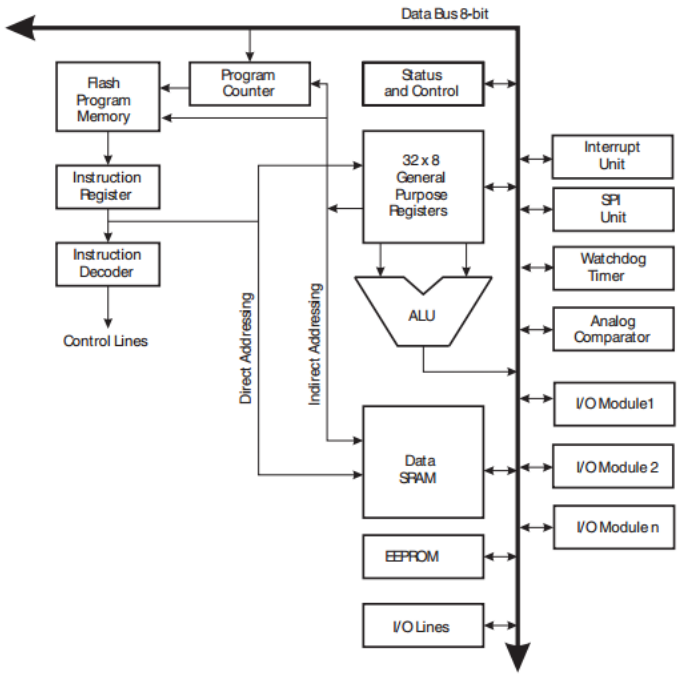
Η έκδοση του Arduino που χρησιμοποιήθηκε αρχικά σε αυτή την εργασία ήταν το Arduino UNO. Εξαιτίας των περιορισμένων δυνατοτήτων του όμως, προορίζεται ως εκπαιδευτικό για την εξοικείωση με την οικογένεια Arduino, και της μη συμβατότητας του με την οθόνη αφής που χρησιμοποιήθηκε έγινε επιλογή του Arduino Mega 2560.

Υπάρχουν σημαντικές διαφορές μεταξύ των δύο μικροελεγκτών με τις πιο σημαντικές από αυτές να είναι η μνήμη flash (32 Kb το UNO και 256 Kb το Mega 2560) και ο αριθμός των ακροδεκτών. Το Arduino UNO σύνολο 20 ακροδέκτες εισόδων/εξόδων ενώ το Arduino Mega 2560 έχει 70.

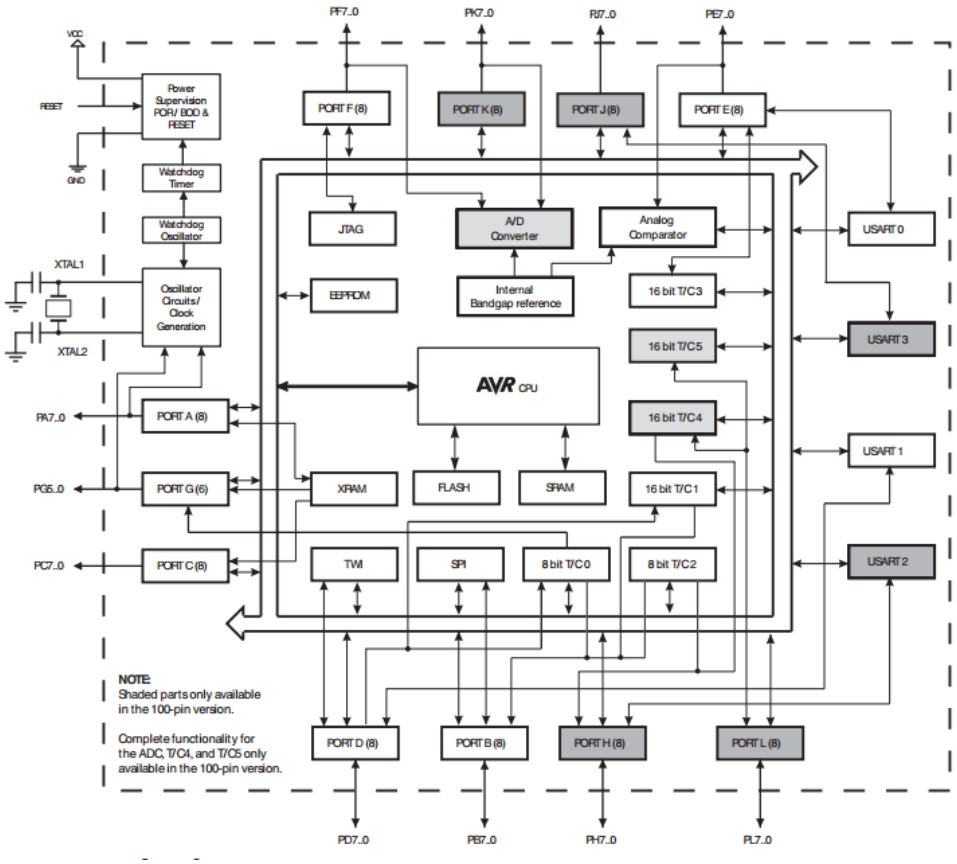
2.1.3 Ο μικροελεγκτής AtMega2560

Ο AtMega2560 είναι ένας υψηλής απόδοσης CMOS 8-bit μικροελεγκτής βασισμένος στην αρχιτεκτονική RISC της AVR. Έχοντας την δυνατότητα να εκτελεί απαιτητικές εντολές σε έναν κύκλο ρολογιού, ο μικροελεγκτής επιτυγχάνει διακίνηση η οποία πλησιάζει τα 1 MIPS ανά MHz εξισορροπώντας την κατανάλωση με την ταχύτητα επεξεργασίας.

Ο AtMega2560 παρέχει 256K bytes ενσωματωμένη προγραμματιζόμενη μνήμη Flash με δυνατότητα ανάγνωσης-εγγραφής, 4 Kbytes EEPROM, 8Kbytes SRAM, 86 εισόδους-εξόδους γενικής χρήσης, 32 γενικής χρήσης καταχωρητές, μετρητή πραγματικού χρόνου (RTC), 6 ευέλικτους μετρητές με καταστάσεις σύγκρισης και PWM, 4 USART, σειριακή διασύνδεση 2 καλωδίων, προγραμματιζόμενο χρονόμετρο φυλάκων με εσωτερικό ταλαντωτή, μία σειριακή θύρα SPI και έναν 16 καναλιών 10 bit A/D μετατροπέα.

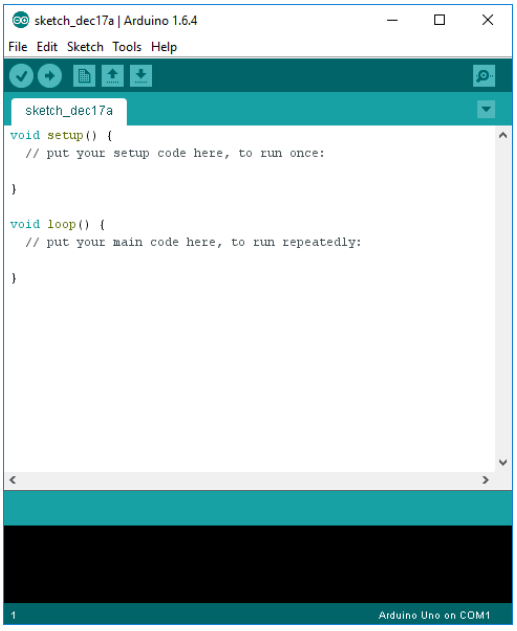


Εικόνα 2.1-Σχηματικό διάγραμμα της Αρχιτεκτονικής του AVR



Εικόνα 2.2-Σχηματικό διάγραμμα AtMega2560

2.2 Περιβάλλον ανάπτυξης κώδικα ARDUINO IDE 1.6.4



Εικόνα 2.3-Arduino IDE

Το λογισμικό που χρησιμοποιήθηκε για τον προγραμματισμό του Arduino είναι το Arduino IDE. Πρόκειται για ένα πρόγραμμα ανοιχτού κώδικα και είναι συμβατό σε Windows, Linux και Mac OS X. Το περιβάλλον είναι γραμμένο σε γλώσσα java και είναι βασισμένο στο project Processing ενώ η γλώσσα προγραμματισμού είναι η Wiring C.

2.2.1 Εισαγωγή στο Arduino IDE.

Κάθε εφαρμογή του Arduino αποτελείται από ένα κύκλωμα και ένα κώδικα. Ο κώδικας εκτελείται από τον μικροελεγκτή και τα αποτελέσματα του φαίνονται στο κύκλωμα.

Ο κώδικας αυτός είναι σε γλώσσα προγραμματισμού Wiring C που ακολουθεί τους κανόνες του δομημένου προγραμματισμού. Κάθε πρόγραμμα αποτελείται τουλάχιστον από δύο διαδικασίες, τη **void setup** και τη **void loop**.

Η διαδικασία void setup εκτελείται μόνο μία φορά στην αρχή ενός προγράμματος και περιλαμβάνει τις αρχικοποιήσεις του συστήματος.

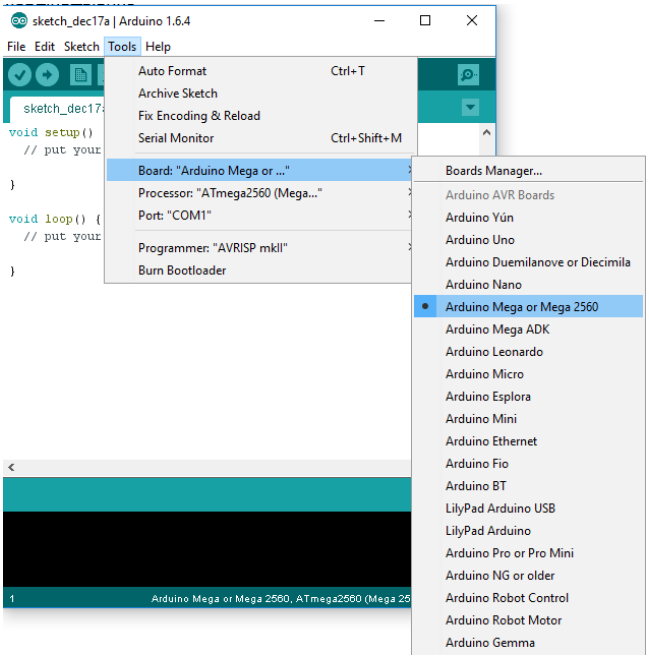
Η διαδικασία void loop εκτελείται συνεχώς μέχρι να σταματήσει να τροφοδοτείται το Arduino.

```
void setup()
{
//εντολές που εκτελούνται μόνο μία φορά
}
void loop()
{
//εντολές που εκτελούνται συνέχεια
}
```

2.2.2 Βασικές ρυθμίσεις προγράμματος και εκτέλεση απλού προγράμματος.

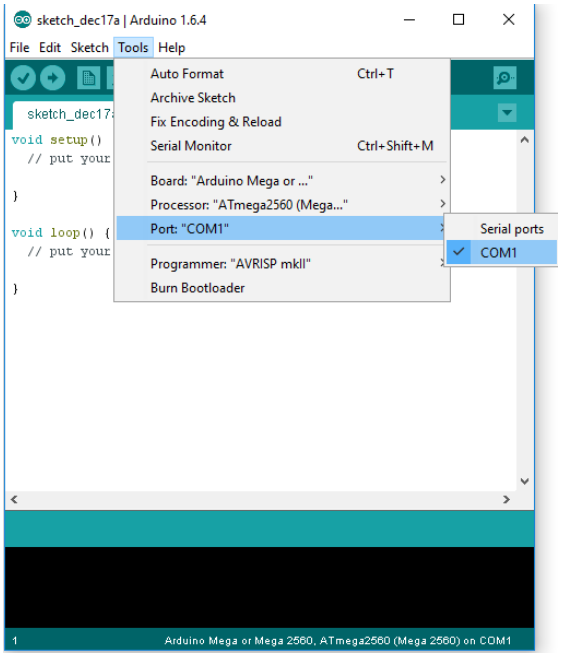
Για να γίνει σωστή χρήση του Arduino IDE θα πρέπει να κάθε φορά πριν την εκτέλεση κάποιου προγράμματος να γίνεται έλεγχος των βασικών ρυθμίσεων του.

- Αρχικά συνδέουμε το Arduino board μέσω του USB στον υπολογιστή.
- Στη συνέχεια κάνουμε κλικ στην επιλογή tools και στην επιλογή Board επιλέγουμε την έκδοση του Arduino που χρησιμοποιούμε.



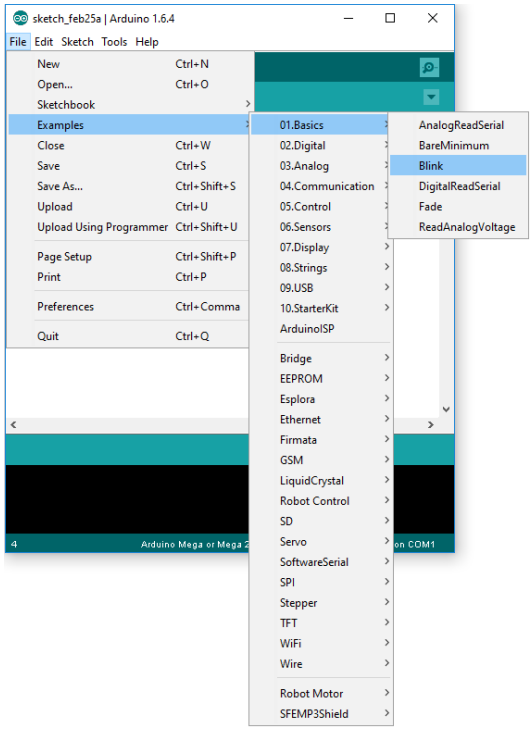
Εικόνα 2.4-Επιλογή πλακέτας

- Αφού επιλέξουμε την πλακέτα που χρησιμοποιούμε επιλέγουμε την θύρα στην οποία έχουμε συνδέσει το Arduino.

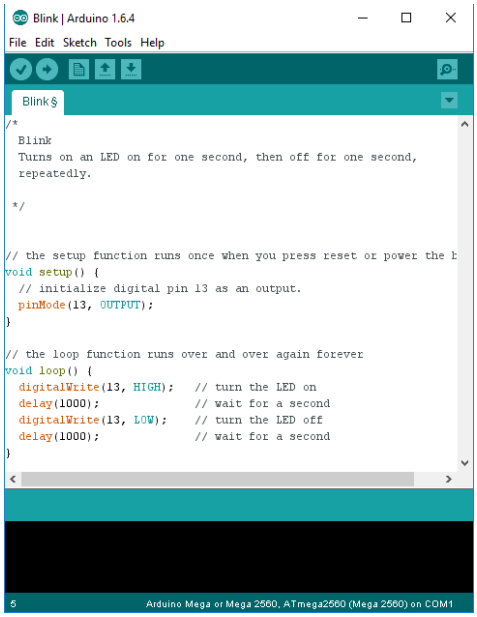


Εικόνα 2.5-Επιλογή θύρας

Αφού κάναμε τις κατάλληλες ρυθμίσεις στο IDE ήρθε η ώρα να τρέξουμε ένα από τα απλά έτοιμα παραδείγματα του IDE για να εξοικειωθούμε με το περιβάλλον του. Θα προσπαθήσουμε να τρέξουμε το παράδειγμα Blink, το οποίο θα αναβοσβήνει το ενσωματωμένο στην πλακέτα LED στον ακροδέκτη 13. Επιλέγουμε την καρτέλα File, μετά Examples, Basics και Blink και βλέπουμε το παράδειγμα (Εικόνα 2.6).



Εικόνα 2.6-Επιλογή παραδείγματος



Εικόνα 2.7-Παράδειγμα Blink

Πατώντας το κουμπί upload στο πάνω αριστερά μέρος του IDE φορτώνουμε το πρόγραμμα στο Arduino. Όταν τελειώσει το upload το LED στον ακροδέκτη 13 του board μας θα πρέπει να αναβοσβήνει ανά ένα δευτερόλεπτο.

Ας αναλύσουμε τα μέρη του προγράμματος για καλύτερη κατανόηση.

1. Ξεκινάμε με ένα σχόλιο πολλών γραμμών. Η αρχή τους συμβολίζεται με /* και τελειώνουν με */. Οτιδήποτε γραφτεί ανάμεσα από τα δύο αυτά σύμβολα δεν θα μεταφραστεί ως κώδικας.
2. Μετά έχουμε τα σχόλια μονής γραμμής. Όταν σε μία γραμμή τοποθετηθεί το σύμβολο // από εκεί και πέρα η γραμμή θεωρείται σχόλιο
3. `void setup()` είναι μία από τις δύο διαδικασίες που πρέπει να υπάρχουν σε όλα τα προγράμματα Arduino. Μία διαδικασία είναι ένα κομμάτι κώδικα που εκτελεί ένα συγκεκριμένο σκοπό. Ο κώδικας εντός της διαδικασίας `setup()` εκτελείται μία φορά στην αρχή κάθε προγράμματος. Αυτό είναι χρήσιμο για ρυθμίσεις όπως ενεργοποίηση κάποιας πλατφόρμας επικοινωνίας με το Arduino, τη συμπεριφορά κάποιων ακροδεκτών και άλλα.
4. Οι ψηφιακοί ακροδέκτες του Arduino μπορούν να λειτουργήσουν ως είσοδοι ή έξοδοι. Για να καθοριστεί η λειτουργία τους, χρησιμοποιείται η συνάρτηση `pinMode()`. Αυτή η συνάρτηση παίρνει δύο παραμέτρους. Η παράμετρος δίνει στις εντολές πληροφορίες για το πώς πρέπει να λειτουργήσουν. Οι παράμετροι τοποθετούνται μέσα σε παρενθέσεις και ακολουθούν μία συνάρτηση. Η πρώτη παράμετρος στην συνάρτηση `pinMode()` καθορίζει ποιού ακροδέκτη τη συμπεριφορά θέτουμε. Θέλουμε να ελέγξουμε το led που βρίσκεται στο 13^{ο} ακροδέκτη άρα ως πρώτη παράμετρο θα έχουμε τον αριθμό 13. Η δεύτερη παράμετρος θέτει την συμπεριφορά του ακροδέκτη: INPUT(Είσοδος) ή OUTPUT(Έξοδος). Οι ακροδέκτες από προεπιλογή θεωρούνται είσοδοι, άρα πρέπει να τους θέσουμε ως εξόδους αν θέλουμε οι διαδικασίες να τους αναγνωρίζουν σωστά. Επειδή θέλουμε να αναβοσβήσουμε ένα led, πρέπει να θέσουμε τον ακροδέκτη ως έξοδο. Αυτή η δήλωση πρέπει να γίνει μόνο μία φορά. Για το υπόλοιπο του προγράμματος θα λειτουργεί ως έξοδο, μέχρι να το αλλάξουμε.
5. Η δεύτερη διαδικασία που είναι απαραίτητη σε όλα τα προγράμματα Arduino είναι η `void loop()`. Τα περιεχόμενα της loop διαδικασίας επαναλαμβάνονται όσο το Arduino είναι ενεργοποιημένο.
6. Η συνάρτηση `digitalWrite()` χρησιμοποιείται για να θέσει την κατάσταση κάποιου ακροδέκτη εξόδου σε 5V ή 0V(HIGH ή LOW). Όταν ένα led και μία αντίσταση είναι συνδεδεμένα σε έναν ακροδέκτη, η κατάσταση HIGH του ακροδέκτη θα ανάψει το led. Η πρώτη παράμετρος στην συνάρτηση `digitalWrite()` είναι ποιόν ακροδέκτη θέλουμε να ελέγξουμε. Η δεύτερη παράμετρος είναι η τιμή που θέλουμε να καταχωρήσουμε σε αυτό, είτε HIGH(5V) ή LOW(0V). Ο ακροδέκτης παραμένει σε αυτή σε αυτή την κατάσταση μέχρι να αλλαχτεί στον κώδικα.
7. Η συνάρτηση `delay()` δέχεται μία παράμετρο: έναν χρόνο καθυστέρησης σε milliseconds. Όταν καλείται η `delay()`, το Arduino σταματά οποιαδήποτε διαδικασία για τον χρόνο που έχουμε δηλώσει. Σε αυτή την περίπτωση, καθυστερούμε το πρόγραμμα για 1000ms, ή ένα δευτερόλεπτο. Αυτό έχει ως αποτέλεσμα το led να παραμένει αναμμένο για ένα δευτερόλεπτο πριν εκτελεστεί η επόμενη εντολή.
8. Εδώ, η `digitalWrite()` χρησιμοποιείται για να σβήσει το led θέτοντας τον ακροδέκτη στην κατάσταση LOW.
9. Ξανά, καθυστερούμε για 1 δευτερόλεπτο για να κρατήσει το led στην κατάσταση LOW προτού η loop επαναληφθεί και αλλάξει πάλι η κατάσταση του ακροδέκτη.

2.2.3 Βασικές δομές και λειτουργίες προγραμματισμού

Παρακάτω ακολουθούν μερικές από τις πιο βασικές δομές και λειτουργίες που μπορούν να αξιοποιηθούν ως εργαλεία κατά την συγγραφή ενός προγράμματος Arduino:

Δομές ελέγχου ροής

- **if** (δομή ελέγχου μίας συνθήκης)
- **if … else** (δομή ελέγχου πολλαπλών συνθηκών)
- **for** (δομή επαναληπτικού ελέγχου συνθήκης)
- **while** (δομή επαναληπτικού ελέγχου συνθήκης)
- **do … while** (δομή επαναληπτικού ελέγχου συνθήκης)
- **switch … case** (δομή ελέγχου περιπτώσεων)
- **break** (εντολή διακοπής μίας επαναληπτικής δομής)
- **continue** (εντολή παράλειψης της τρέχουσας επανάληψης)
- **return** (εντολή επιστροφής από μία συνάρτηση)
- **go to** (εντολή μετάβασης σε κάποιο σημείο του κώδικα)

Αριθμητικοί τελεστές

- **=** (τελεστής εκχώρησης)
- **+** (τελεστής πρόσθεσης)
- **-** (τελεστής αφαίρεσης)
- ***** (τελεστής πολλαπλασιασμού)
- **/** (τελεστής διαίρεσης)
- **%** (τελεστής υπόλοιπου ακεραίας διαίρεσης)

Λογικοί τελεστές

- **&&** (λογική σύζευξη)
- **||** (λογική διάζευξη)
- **!** (λογική άρνηση)

Δυαδικοί τελεστές

- **&** (δυαδική σύζευξη)
- **|** (δυαδική διάζευξη)
- **^** (δυαδική αποκλειστική διάζευξη)
- **~** (δυαδική άρνηση)
- **<<** (δυαδική αριστερή ολίσθηση)
- **>>** (δυαδική δεξιά ολίσθηση)

Τελεστές αύξησης και μείωσης

- **++** (αύξηση κατά μία ακέραιη μονάδα)
- **- -** (μείωση κατά μία ακέραιη μονάδα)

Σύνθετοι τελεστές

- **+= , -= , *= , /= , %=** (σύνθετοι αριθμητικοί τελεστές)
- **&= , |= , ^= , ~= , <<= , >>=** (σύνθετοι δυαδικοί τελεστές)

Τελεστές σύγκρισης

- **==** (ισότητα)
- **!=** (ανισότητα)
- **<** (μικρότερο)
- **>** (μεγαλύτερο)

- **<=** (μικρότερο ή ίσο)
- **>=** (μεγαλύτερο ή ίσο)

Τελεστές δεικτών
- ***** (τελεστής απόκτησης περιεχομένου)
- **&** (τελεστής απόκτησης διεύθυνσης)

Σταθερές
- **HIGH** (τιμή υψηλής στάθμης για μία επαφή εισόδου ή εξόδου)
- **LOW** (τιμή χαμηλής στάθμης για μία επαφή εισόδου ή εξόδου)
- **false** (λογικό επίπεδο ψεύδους σε μία συνθήκη)
- **true** (λογικό επίπεδο αλήθειας σε μία συνθήκη)
- **INPUT** (χρησιμοποιείται για τον ορισμό μίας επαφής ως είσοδο)
- **OUTPUT** (χρησιμοποιείται για τον ορισμό μία επαφής ως έξοδο)
- **A0 , … , A5** (συμβολοσταθερές για τις αναλογικές επαφές εισόδου)

Τύποι δεδομένων
- **boolean** (λογική δυαδική τιμή)
- **char** (προσημασμένος χαρακτήρας 8 ψηφίων)
- **unsigned char** (μη προσημασμένος χαρακτήρας 8 ψηφίων)
- **byte** (μη προσημασμένος χαρακτήρας 8 ψηφιών)
- **int** (προσημασμένος ακέραιος αριθμός 16 ψηφίων)
- **unsigned int** (μη προσημασμένος ακέραιος αριθμός 16 ψηφίων)
- **word** (μη προσημασμένος ακέραιος αριθμός 16 ψηφίων)
- **long** (προσημασμένος ακέραιος αριθμός 32 ψηφίων)
- **unsigned long** (μη προσημασμένος ακέραιος αριθμός 32 ψηφίων)
- **float, double** (αριθμός κινητής υποδιαστολής απλής ακρίβειας)
- **String** (αντικείμενο αλφαριθμητικού με χρήσιμες μεθόδους)
- Ως αλφαριθμητικό μπορεί να θεωρηθεί και ο πίνακας χαρακτήρων

Συναρτήσεις μετατροπής τύπων
- **char() , byte()**
- **int() , word() , long()**
- **float() , double()**

Συναρτήσεις εισόδου και εξόδου
- **pinMode()** (ορίζει μία επαφή ως είσοδο ή έξοδο)

Συναρτήσεις ψηφιακής εισόδου και εξόδου
- **digitalWrite()** (γράφει σε μία ψηφιακή επαφή εξόδου)
- **digitalRead()** (διαβάζει από μία ψηφιακή επαφή εισόδου)

Συναρτήσεις αναλογικής εισόδου και εξόδου
- **analogReference()** (ορίζει την τάση αναλογικής αναφοράς)
- **analogWrite()** (γράφει PWM σήματα σε μία επαφή εξόδου)
- **analogRead()** (διαβάζει από μια αναλογική επαφή εισόδου)

Προηγμένες συναρτήσεις εισόδου και εξόδου
- **tone()** (παράγει ένα τετραγωνικό σήμα ορισμένης συχνότητας)
- **noTone()** (διακόπτει την παραγωγή τετραγωνικών σημάτων)
- **shiftOut()** (ολισθαίνει τα ψηφία μίας τιμής σε μία επαφή εξόδου)

- **pulseIn()** (επιστρέφει την διάρκεια σε μs ενός παλμού HIGH ή LOW)

Συναρτήσεις χρόνου
- **millis()** (διάρκεια εκτέλεσης του προγράμματος σε ms)
- **micros()** (διάρκεια εκτέλεσης του προγράμματος σε μs)
- **delay()** (παύση προγράμματος – η διάρκεια δίδεται σε ms)
- **delayMicroseconds()** (παύση προγράμματος – η διάρκεια δίδεται σε μs)

Μαθηματικές και Τριγωνομετρικές συναρτήσεις
- **max()** (βρίσκει τον μεγαλύτερο ανάμεσα σε δύο αριθμούς)
- **min()** (βρίσκει τον μικρότερο ανάμεσα σε δύο αριθμούς)
- **abs()** (επιστρέφει την απόλυτη τιμή ενός αριθμού)
- **constrain()** (ελέγχει για υπερχείλιση ή υποχείλιση ορίων)
- **map()** (πραγματοποιεί γραμμικό μετασχηματισμό ορίων)
- **pow()** (επιστρέφει το αποτέλεσμα μίας δύναμης)
- **sqrt()** (επιστρέφει την ρίζα ενός αριθμού)
- **sin()** (υπολογίζει το ημίτονο ενός αριθμού)
- **cos()** (υπολογίζει το συνημίτονο ενός αριθμού)
- **tan()** (υπολογίζει την εφαπτομένη ενός αριθμού)

Συναρτήσεις γεννήτριας ψευδοτυχαίων αριθμών
- **random()** (δίδεται ένας νέος αριθμός από την γεννήτρια)
- **randomSeed()** (θέτει τον σπόρο της γεννήτριας παραγωγής)

Συναρτήσεις επεξεργασίας δυαδικών αριθμών
- **lowByte()** (επιστρέφει το δεξιότερο byte μίας μεταβλητής)
- **highByte()** (επιστρέφει το αριστερότερο byte μίας μεταβλητής)
- **bitRead()** (διαβάζει ένα συγκεκριμένο ψηφίο μίας μεταβλητής)
- **bitWrite()** (γράφει σε ένα συγκεκριμένο ψηφίο μίας μεταβλητής)
- **bitSet()** (γράφει την τιμή 1 σε κάποιο ψηφίο μίας μεταβλητής)
- **bitClear()** (γράφει την τιμή 0 σε κάποιο ψηφίο μίας μεταβλητής)
- **bit()** (υπολογίζει συγκεκριμένη δύναμη με βάση το 2)

Συναρτήσεις χρήσης ρουτινών εξυπηρέτησης διακοπών
- **attachInterrupt()** (ενεργοποιεί μία ρουτίνα εξυπηρέτησης διακοπής)
- **detachInterrupt()** (απενεργοποιεί μία ρουτίνα εξυπηρέτησης διακοπών)

Συναρτήσεις ενεργοποίησης και απενεργοποίησης διακοπών
- **interrupts()** (ενεργοποιεί τα σήματα διακοπής)
- **noInterrupts()** (απενεργοποιεί τα σήματα διακοπής)

Υποστήριξη σειριακής επικοινωνίας
- **Serial** (αντικείμενο σειριακής επικοινωνίας με χρήσιμες μεθόδους)

2.3 Γλώσσα σήμανσης HTML

Η HTML (αρχικοποίηση του αγγλικού Hyper Text Markup Language ή στα ελληνικά Γλώσσα Σήμανσης Υπερκειμένου) είναι η κύρια γλώσσα σήμανσης για τις ιστοσελίδες, και τα στοιχεία της είναι τα βασικά δομικά στοιχεία των ιστοσελίδων.

Η HTML γράφεται υπό μορφή στοιχείων HTML τα οποία αποτελούνται από ετικέτες (tags), οι οποίες περικλείονται μέσα σε σύμβολα «μεγαλύτερο από» και «μικρότερο από» (για παράδειγμα <html>), μέσα στο περιεχόμενο της ιστοσελίδας. Οι ετικέτες HTML συνήθως λειτουργούν ανά ζεύγη (για παράδειγμα <h1> και </h1>), με την πρώτη να ονομάζεται ετικέτα έναρξης και την δεύτερη ετικέτα λήξης (ή σε άλλες περιπτώσεις ετικέτα ανοίγματος και ετικέτα κλεισίματος αντίστοιχα). Ανάμεσα στις ετικέτες, οι σχεδιαστές ιστοσελίδων μπορούν να τοποθετήσουν κείμενο, πίνακες, εικόνες κλπ.

Ο σκοπός ενός web browser είναι να διαβάζει τα έγγραφα HTML και να τα συνθέτει σε σελίδες που μπορεί κανείς να διαβάσει ή να ακούσει. Ο browser δεν εμφανίζει τις ετικέτες HTML, αλλά τις χρησιμοποιεί για να ερμηνεύσει το περιεχόμενο της σελίδας.

Τα στοιχεία της HTML χρησιμοποιούνται για να κτίσουν όλους τους ιστότοπους. Η HTML επιτρέπει την ενσωμάτωση εικόνων και άλλων αντικειμένων μέσα στη σελίδα, και μπορεί να χρησιμοποιηθεί για να εμφανίσει διαδραστικές φόρμες. Παρέχει τις μεθόδους δημιουργίας δομημένων εγγράφων (δηλαδή εγγράφων που αποτελούνται από το περιεχόμενο που μεταφέρουν και από τον κώδικα μορφοποίησης του περιεχομένου) καθορίζοντας δομικά σημαντικά στοιχεία για το κείμενο, όπως κεφαλίδες, παραγράφους, λίστες, συνδέσμους, παραθέσεις, και άλλα. Μπορούν επίσης να ενσωματώνονται σενάρια εντολών σε γλώσσες όπως η JavaScript, τα οποία επηρεάζουν τη συμπεριφορά των ιστοσελίδων HTML.

2.3.1 Σήμανση

Η σήμανση HTML αποτελείται από μερικά βασικά συστατικά, συμπεριλαμβανομένων των στοιχείων και των ιδιοτήτων τους, τους βασισμένους σε χαρακτήρες τύπους δεδομένων, τις αναφορές χαρακτήρων και τις αναφορές οντοτήτων. Ένα ξεχωριστό σημαντικό συστατικό είναι η δήλωση τύπου εγγράφου (document type declaration), η οποία ορίζει στον browser τον τρόπο εμφάνισης της σελίδας.

Στην HTML, το πρόγραμμα Hello world, ένα συνηθισμένο πρόγραμμα υπολογιστή που χρησιμεύει για τη σύγκριση γλωσσών προγραμματισμού, γλωσσών σεναρίων και γλωσσών σήμανσης, φτιάχνεται με 9 γραμμές κώδικα, παρότι οι νέες γραμμές είναι προαιρετικές στην HTML:

```
<!DOCTYPE html>
<html>
<head>
<title>Hello HTML</title>
</head>
<body>
<p>Hello world</p>
</body>
</html>
```

Το κείμενο στο `<html>` και το `</html>` περιγράφει την ιστοσελίδα, και το κείμενο μεταξύ του `<body>` και του `</body>` είναι το ορατό μέρος της. Το σημασμένο κείμενο `<title>Hello HTML</title>` καθορίζει τον τίτλο που θα εμφανίζεται στην μπάρα τίτλου του browser.

Το Document Type Declaration στον πιο πάνω κώδικα είναι για την HTML5. Αν δεν συμπεριλαμβάνεται η δήλωση `<!DOCTYPE html>` τότε μερικοί browser θα καταφύγουν στην λειτουργία quirks για την εμφάνιση της σελίδας.

2.3.2 Παραδείγματα Στοιχείων HTML

Κεφαλίδα του εγγράφου HTML: `<head>…</head>`. Συνήθως περιέχει τον τίτλο, π.χ.:

```
<head>
```

```
  <title>Ο τίτλος</title>
</head>
```

Επικεφαλίδες: οι επικεφαλίδες στην HTML ορίζονται με τις ετικέτες <h1> έως <h6>:

```
<h1>Επικεφαλίδα1</h1>
<h2>Επικεφαλίδα2</h2>
<h3>Επικεφαλίδα3</h3>
<h4>Επικεφαλίδα4</h4>
<h5>Επικεφαλίδα5</h5>
<h6>Επικεφαλίδα6</h6>
```

Παράγραφοι:

```
<p>Παράγραφος 1</p>  <p>Παράγραφος 2</p>
```

Αλλαγή γραμμής:
. Η διαφορά ανάμεσα στο
 και το <p> είναι ότι το
 αλλάζει γραμμή χωρίς να αλλάζει την σημαντική δομή της σελίδας, ενώ το <p> τεμαχίζει τη σελίδα σε παραγράφους. Το
 είναι ένα άδειο στοιχείο, δηλαδή δεν έχει περιεχόμενο, ούτε και χρειάζεται ετικέτα τερματισμού.

```
<p>Αυτή <br>είναι μία παράγραφος <br> με <br> αλλαγή γραμμής</p>
```

Αυτός είναι ένας σύνδεσμος σε HTML: Για τη δημιουργία του χρησιμοποιείται η ετικέτα <a>. Η ιδιότητα href περιέχει τη σελίδα προορισμού του συνδέσμου.

```
<a href="http://el.wikipedia.org/">Ενας σύνδεσμος στη Βικιπαίδεια</a>
```

2.3.3 Ιδιότητες της HTML

Οι περισσότερες ιδιότητες των στοιχείων είναι ζεύγη ονομάτων και τιμών, τα οποία διαχωρίζονται με ένα «=» και γράφονται μέσα στην ετικέτα εκκίνησης ενός στοιχείου, μετά το όνομα του στοιχείου. Η τιμή μπορεί να περικλείεται σε μονά ή διπλά εισαγωγικά, παρότι τιμές που αποτελούνται από συγκεκριμένους χαρακτήρες μπορούν να γράφονται χωρίς εισαγωγικά στην HTML, αλλά όχι στην XHTML. Το να μένουν οι τιμές των ιδιοτήτων χωρίς εισαγωγικά θεωρείται ανασφαλές. Εκτός από τις ιδιότητες που γράφονται ως ζεύγη ονομάτων και τιμών, υπάρχουν και μερικές οι οποίες επηρεάζουν το στοιχείο απλά με την παρουσία τους μέσα στην ετικέτα εκκίνησης, όπως η ιδιότητα ismap του στοιχείου img.

Υπάρχουν και μερικές κοινές ιδιότητες οι οποίες εμφανίζονται σε πολλά στοιχεία:

- Η ιδιότητα id παρέχει ένα αναγνωριστικό για ένα στοιχείο το οποίο είναι μοναδικό σε ολόκληρο το έγγραφο. Χρησιμοποιείται για να ταυτοποιεί το στοιχείο ώστε τα CSS να μπορούν να αλλάξουν τον τρόπο που αυτό εμφανίζεται, καθώς και τα σενάρια μπορούν να αλλάξουν, να μετακινήσουν ή να διαγράψουν τα περιεχόμενα ή την εμφάνισή του. Ακόμα, αν ένα id προστεθεί στο URL μίας σελίδας, παρέχει ένα μοναδικό αναγνωριστικό για ένα τμήμα της σελίδας, για παράδειγμα: http://el.wikipedia.org/wiki/HTML#**Ιδιότητες**
- Η ιδιότητα class παρέχει τη δυνατότητα ταξινόμησης παρόμοιων αντικειμένων στην ίδια κλάση. Μπορεί να χρησιμοποιηθεί για να αποδώσει κάποια σημασία στο στοιχείο, ή για σκοπούς εμφάνισης. Για παράδειγμα, ένα έγγραφο HTML μπορεί να χρησιμοποιεί την επισήμανση class=”notation” σε μερικά στοιχεία για ξεχωρίσει από το υπόλοιπο κείμενο του εγγράφου. Κατά την εμφάνιση του εγγράφου, αυτά τα στοιχεία μπορεί –για παράδειγμα- να εμφανίζονται όλα μαζί στο τέλος της σελίδας ως υποσημειώσεις, άσχετα με την θέση που εμφανίζονται μέσα στον κώδικα. Επίσης οι ιδιότητες class χρησιμοποιούνται σημασιολογικά στα microformat. Ένα στοιχείο μπορεί να έχει πολλαπλές κλάσεις, για παράδειγμα το class=”notation important” βάζει

το στοιχείο τόσο στην κλάση «notation» όσο και στην «important».

- Η ιδιότητα style εφαρμόζει στυλ εμφάνισης σε συγκεκριμένα στοιχεία. Θεωρείται καλύτερη τακτική να χρησιμοποιούνται οι ιδιότητες id ή class ώστε να επιλέγεται το στοιχείο μέσα σε ένα CSS, αλλά μερικές φορές μπορεί να είναι πιο απλά να ανατεθούν κατευθείαν στο στοιχείο.
- Η ιδιότητα title προσθέτει μία εξήγηση στο στοιχείο στο οποίο εφαρμόζεται. Στους περισσότερους φυλλομετρητές αυτή η ιδιότητα εμφανίζεται ως αναδυόμενο παράθυρο βοήθειας.
- Η ιδιότητα lang ταυτοποιεί την φυσική γλώσσα των περιεχόμενων του στοιχείου, η οποία μπορεί να είναι διαφορετική από το υπόλοιπο έγγραφο.

Για παράδειγμα, μέσα σε ένα έγγραφο μπορεί να υπάρχει το κείμενο:

```
<p>Τι να κανουμε;<span lang="fr">c'est la vie</span>, όπως λένε και οι γάλλοι</p>
```

Το στοιχείο abbr μπορεί να χρησιμοποιηθεί για να επιδείξει μερικές από τις πιο πάνω ιδιότητες:

```
<abbr id="anId" class="jargon" style="color:purple;" title="Hypertext Markup Language">HTML</abbr>
```

Αυτό το παράδειγμα εμφανίζεται ως: HTML. Στους περισσότερους φυλλομετρητές, όταν σταθεί ο δείκτης του ποντικιού πάνω από αυτό το στοιχείο, τότε θα εμφανιστεί ένα αναδυόμενο παράθυρο με το περιεχόμενο «Hypertext Markup Language».

Τα περισσότερα στοιχεία μπορούν επίσης να δεχτούν την ιδιότητα dir η οποία καθορίζει την κατεύθυνση του κειμένου. Η τιμή «rtl» της ιδιότητας, δείχνει κείμενο που διαβάζεται από δεξιά προς τα αριστερά, όπως για παράδειγμα στα Αραβικά, στα Περσικά ή στα Εβραϊκά.

2.3.4 Τύποι δεδομένων

Στην HTML ορίζονται μερικοί τύποι δεδομένων για το περιεχόμενο των στοιχείων, όπως σενάρια εντολών ή stylesheet, και μία πληθώρα τύπων για τις τιμές των ιδιοτήτων, συμπεριλαμβανομένων των ID, των name, των URL, διαφόρων αριθμών και μονάδων μήκους, γλωσσών, τύπων αρχείων πολυμέσων, χρωμάτων, κωδικοποιήσεων χαρακτήρων, ημερομηνιών κλπ.

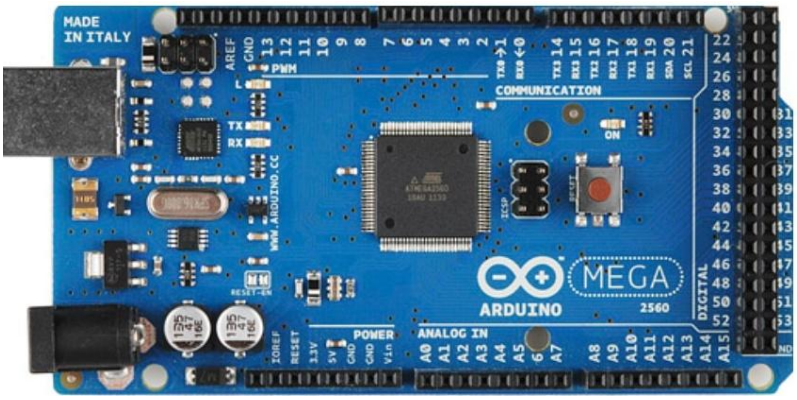
2.3.5 Δήλωση τύπου εγγράφου

Τα έγγραφα HTML πρέπει να αρχίζουν με μία Δήλωση τύπου εγγράφου (Document Type Declaration, ανεπίσημα λέγεται και «doctype»). Αυτή η δήλωση βοηθάει τους browser να καταλάβουν πώς πρέπει να διαβάσουν το περιεχόμενο του εγγράφου και πώς να το παρουσιάσουν μετά, και ιδιαίτερα όταν χρησιμοποιείται το quirks mode.

Ο αρχικός σκοπός του doctype ήταν να επιτρέπει την ανάλυση και επιβεβαίωση των εγγράφων HTML από εργαλεία SGML τα οποία ήταν βασισμένα στο Document Type Definition (DTD). Το DTD στο οποίο αναφέρεται το DOCTYPE περιέχει μία γραμματική σε γλώσσα μηχανής, η οποία καθορίζει τι επιτρέπεται και τι απαγορεύεται να υπάρχει μέσα στο έγγραφο. Οι browser, από την άλλη, δεν υλοποιούν την HTML ως εφαρμογή της SGML και συνεπώς δεν διαβάζουν το DTD. Στην HTML5 δεν ορίζεται κανένα DTD, λόγω εγγενών τεχνολογικών περιορισμών. Έτσι το doctype <!doctype html> δεν αναφέρεται σε κανένα.

ΚΕΦΑΛΑΙΟ 3: Υλικό Μέρος του Συστήματος

3.1 Arduino MEGA 2560



Εικόνα 3.1-Arduino MEGA 2560

Το Arduino /Genuino Mega 2560 είναι βασισμένο στον ATmega2560 μικροελεγκτή και πρόκειται για αναβάθμιση του Arduino Mega το οποίο αντικαθιστά. Έχει 54 digital input/output pins (από τα οποία τα 15 μπορούν να χρησιμοποιηθούν ως PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), ένα κρυσταλλικό ταλαντωτή 16 MHz, μία θύρα USB, ένα power jack, μία κεφαλή ICSP, και ένα reset button. Περιέχει ότι χρειάζεται για να υποστηρίξει τον μικροελεγκτή. Συνδέεται με τον υπολογιστή μέσω USB, ή τροφοδοτείται με ένα τροφοδοτικό AC-DC 7-12Volt ή με μία μπαταρία 9 Volt. Το Arduino Mega 2560 είναι συμβατό με τα περισσότερα shield που έχουν σχεδιαστεί για Arduino/Genuino Uno αλλά και για προηγούμενα boards όπως Duemilanove ή Diecimila.

Τεχνικά Χαρακτηριστικά

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12 V
Input Voltage (limit)	6-20 V
Digital I/O Pins	54 (of which 15 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 Kb of which 8 Kb used by bootloader

SRAM	8 Kb
EEPROM	4 Kb
Clock Speed	16 MHz
Length	101.52 mm
Width	53.3 mm
Weight	37g

3.1.1 Προγραμματισμός του ArduinoBoardMega2560

Το Mega 2560 μπορεί να προγραμματιστεί με το Arduino IDE. Ο μικροελεγκτής Atmega2560 που περιέχει το Mega 2560 έχει προεγκατεστημένο έναν bootloader που επιτρέπει τον προγραμματισμό του χωρίς τη χρήση ενός εξωτερικού προγραμματιστή υλικού και επικοινωνεί χρησιμοποιώντας το STK500 πρωτόκολλο.

Υπάρχει επίσης η δυνατότητα προγραμματισμού του μικροελεγκτή μέσω της κεφαλής ICSP (In-Circuit Serial Programming) με ένα Arduino ISP.

3.1.2 Προειδοποιήσεις για τη χρήση του ArduinoBoardMega2560

Το Mega 2560 έχει ασφάλεια που προστατεύει τις θύρες USB του υπολογιστή από βραχυκύκλωμα και υπερφόρτιση. Παρόλο που οι περισσότεροι υπολογιστές διαθέτουν τη δικιά τους εσωτερική προστασία, η ασφάλεια αυτή παρέχει ένα επιπλέον επίπεδο προστασίας. Εάν η θύρα USB δεχτεί παραπάνω από 500 mA η ασφάλεια θα σταματήσει την επικοινωνία μέχρι το βραχυκύκλωμα ή η υπερφόρτιση αποκατασταθεί.

3.1.3 Τροφοδοσία

Το ArduinoBoardMega2560 μπορεί να τροφοδοτηθεί είτε μέσω της σύνδεσης σε υπολογιστή είτε με εξωτερική τροφοδοσία ρεύματος. Η πηγή τροφοδοσίας επιλέγεται αυτόματα. Η εξωτερική τροφοδοσία μπορεί να είναι ένα τροφοδοτικό AC σε DC ή μία μπαταρία. Το τροφοδοτικό μπορεί να συνδεθεί με ένα 2.1 mm βύσμα με θετικό κέντρο στη θύρα τροφοδοσίας της πλακέτας. Η μπαταρία μπορεί να συνδεθεί στους GND και Vin ακροδέκτες. Η πλακέτα μπορεί να λειτουργήσει με εξωτερική τροφοδοσία από 6 έως 20 volts. Όμως αν τροφοδοτηθεί με λιγότερο από 7 V ο 5V ακροδέκτης ίσως παρέχει λιγότερα από πέντε volt και η πλακέτα μπορεί να μην λειτουργεί σωστά. Επίσης αν χρησιμοποιηθούν παραπάνω από 12V, ο ρυθμιστής τάσης μπορεί να υπερθερμανθεί και να προκαλέσει βλάβη στην πλακέτα. Το συνιστώμενο εύρος τάσης είναι 7 έως 12 volts.

3.1.4 Οι ακροδέκτες τροφοδοσίας του ArduinoBoardMega2560

- **Vin.** Η τάση εισόδου της πλακέτας όταν τροφοδοτείται από εξωτερική πηγή (σε αντίθεση με τα 5 V από τη σύνδεση μέσω USB ή άλλες ρυθμιζόμενες πηγές ενέργειας). Μπορούμε να παρέχουμε τάση με αυτό το Pin, ή, αν έχουμε τροφοδοσία μέσω της θύρας τροφοδοσίας της πλακέτας, να έχουμε πρόσβαση σε αυτή την τάση.

- **5V.** Αυτός ο ακροδέκτης παράγει στην έξοδο μία ρυθμιζόμενη τάση 5V από τον ρυθμιστή πάνω στην πλακέτα. Το board μπορεί να τροφοδοτηθεί είτε από την DC θύρα τροφοδοσίας(7-12 V), τη θύρα USB (5V), ή τον ακροδέκτη Vin (7-12V). Τυχόν τροφοδοσία μέσω των ακροδεκτών 5V ή 3.3V παρακάμπτει τον ρυθμιστή τάσης, και μπορεί να προκαλέσει βλάβη στην πλακέτα.

- **3.3V.** Μία παροχή 3.3 V από τον ρυθμιστή της πλακέτας με μέγιστη παροχή ρεύματος 50 mA.

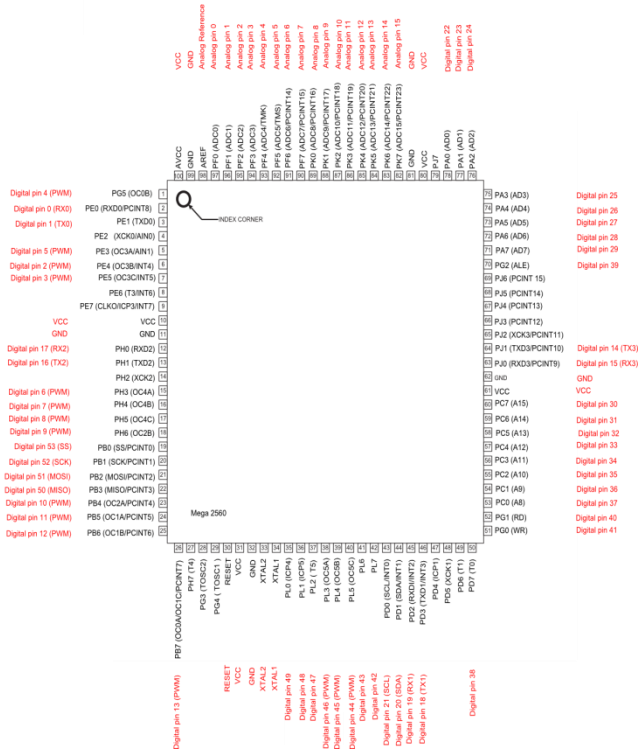
- **GND.** Ακροδέκτες γείωσης της πλακέτας.

- **IOREF.** Αυτός ο ακροδέκτης της πλακέτας παρέχει την τάση αναφοράς με την οποία λειτουργεί ο μικροελεγκτής. Ένα κατάλληλα ρυθμισμένο Shield μπορεί να διαβάσει την τάση στον ακροδέκτη IOREF και να επιλέξει την κατάλληλη πηγή ενέργειας, ή να ενεργοποιήσει τους μεταφραστές τάσης στις εξόδους για λειτουργία στα 5 ή 3.3 Volts.

3.1.5 Μνήμη του ArduinoBoardMega2560

Ο μικροελεγκτής Atmega2560 έχει 256 Kb μνήμης flash για την αποθήκευση κώδικα (από τα οποία τα 8 Kb χρησιμοποιούνται για τον bootloader), 8 Kb SRAM και 4Kb EEPROM (η οποία μπορεί να διαβαστεί ή να γραφεί με την βιβλιοθήκη EEPROM).

3.1.6 Είσοδοι και έξοδοι του ArduinoBoardMega2560



Εικόνα 3.2 – Διάγραμμα pin του Arduino Mega 2560

Κάθε ένα από τα ψηφιακά pins στο ArduinoBoardMega2560 μπορεί να χρησιμοποιηθεί ως είσοδος ή έξοδος, χρησιμοποιώντας τις συναρτήσεις pinMode(), digitalWrite() και την digitalRead() και λειτουργούν στα 5 V. Κάθε pin μπορεί να παρέχει ή να λάβει 20 mA σε συνιστώμενες συνθήκες και έχει εσωτερικές αντιστάσεις ανύψωσης 20-50 k ohm που είναι αρχικά αποσυνδεδεμένες. Τα 40 mA είναι η μέγιστη τιμή που δεν πρέπει να ξεπεραστεί για να αποφύγουμε μόνιμες βλάβες στον μικροελεγκτή.

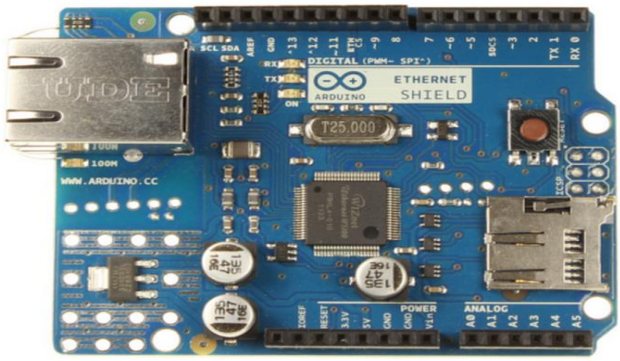
Επιπλέον, κάποια από τα Pin έχουν ειδικές λειτουργίες.

- **Serial: 0(RX) και 1(TX); Serial 1: 19(RX) και 18(TX); Serial 2: 17(RX) και 16(TX); Serial 3: 15(RX) και 14(TX).** Χρησιμοποιούνται για την αποδοχή(RX) και αποστολή (TX) TTL σειριακών δεδομένων. Τα pins 0 και 1 συνδέονται στα αντίστοιχα pins του Atmega16U2 USB - σε - TTL Serial Chip.
- **External Interrupts: 2(interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), και 21 (interrupt 2).** Αυτά τα pins μπορούν να ρυθμιστούν έτσι ώστε να προκαλούν μία διακοπή (interrupt) σε χαμηλό επίπεδο, μία αύξηση ή πτώση ή την αλλαγή επιπέδου.
- **PWM: 2 έως 13 και 44 έως 46.** Παρέχουν έξοδο 8-bit PWM με την διαδικασία analogWrite().
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** Αυτά τα pins παρέχουν SPI επικοινωνία μέσω της βιβλιοθήκης SPI. Τα pins SPI υπάρχουν και στην κεφαλή ICSP, η οποία είναι συμβατή με αυτή του Arduino/Genuino UNO και των παλιών Duemilanove και Duecimila Arduino.
- **LED: 13.** Υπάρχει ένα ενσωματωμένο LED συνδεδεμένο με το ψηφιακό pin 13. Όταν η τιμή του Pin είναι HIGH το LED είναι αναμμένο, ενώ όταν είναι LOW είναι σβηστό.
- **TWI: 20 (SDA) και 21(SCL).** Υποστηρίζει την TWI επικοινωνία χρησιμοποιώντας την Wire βιβλιοθήκη.

Το Mega 2560 έχει 16 αναλογικές εισόδους, και κάθε μία από αυτές παρέχει 10 bits ανάλυσης δηλαδή 1024 διαφορετικές τιμές. Από προεπιλογή μετρούν από 0 έως 5 V, αλλά είναι πιθανό να αλλάξει το πάνω μέρος της κλίμακας χρησιμοποιώντας το AREF pin και την συνάρτηση analogReference().

- **AREF.** Τάση αναφοράς για τις αναλογικές εισόδους. Χρησιμοποιείται με την συνάρτηση analogReference().
- **Reset.** Θέτοντας αυτό το pin LOW κάνουμε επαναφορά τον μικροελεγκτή. Συνήθως χρησιμοποιείται για την προσθήκη κουμπιού επαναφοράς όταν κάποιο shield εμποδίζει αυτό που υπάρχει στην πλακέτα.

3.2 Επικοινωνία Arduino με το διαδίκτυο



Εικόνα 3.3- Arduino Ethernet Shield

Το Arduino Ethernet Shield συνδέει το Arduino με το internet. Τοποθετώντας το πάνω στο Arduino board και με ένα καλώδιο RJ45 και με κάποια απλά βήματα μπορούμε να το ελέγξουμε από το internet. Οι απαιτήσεις του Ethernet Shield είναι:

- Μία πλακέτα Arduino
- Τάση τροφοδοσίας 5V που λαμβάνει από το Arduino.
- Ethernet Controller: W5100 με εσωτερικό buffer 16K
- Ταχύτητα σύνδεσης 10/100 Mb
- Σύνδεση με την θύρα SPI του Arduino.

Το Arduino Ethernet Shield είναι βασισμένο στο W5100 ολοκληρωμένο κύκλωμα της Wiznet και παρέχει δίκτυο (IP) κατάλληλο για TCP και UDP. Με την χρήση της βιβλιοθήκης Ethernet μπορούμε να γράψουμε προγράμματα για σύνδεση στο internet. Το Ethernet Shield συνδέεται με την πλακέτα με μακριά σύρματα-ακίδες που εξέχουν από το Shield. Έτσι η σειρά των Pin δεν μεταβάλλεται ώστε να υπάρχει δυνατότητα να τοποθετηθεί και άλλο Shield από πάνω.

Το Arduino Ethernet Shield έχει μία τυπική RJ-45 σύνδεση με ενσωματωμένο μετασχηματιστή για τροφοδοσία μέσω Ethernet.

Υπάρχει ενσωματωμένη θύρα για micro-SD κάρτα μνήμης. Που μπορεί να χρησιμοποιηθεί για αποθήκευση δεδομένων για την εξυπηρέτηση μέσω του δικτύου. Η ενσωματωμένη micro-SD κάρτα γίνεται προσβάσιμη μέσω της βιβλιοθήκης SD. Όσο χρησιμοποιούμε αυτή την βιβλιοθήκη το SS καταλαμβάνει το 4^{o} pin. Το Shield μας έχει επίσης κουμπί reset, για να είναι σίγουρο ότι το W510 Ethernet module αρχικοποιημένο κατά την εκκίνηση του.

Το Arduino επικοινωνεί με το W5100 και την κάρτα SD μέσω του διαύλου SPI (μέσω της κεφαλής ICSP), ο οποίος βρίσκεται στα ψηφιακά pins 10,11,12 και 13 στο Arduino Uno και στα 50,51 και 52 στο Mega. Και στις δύο πλακέτες ο ακροδέκτης 10 χρησιμοποιείται για την επιλογή του W5100 και ο ακροδέκτης 4 για την SD κάρτα. Αυτοί οι ακροδέκτες παραμένουν απασχολημένοι και δεν μπορούν να χρησιμοποιηθούν ως γενικής χρήσης είσοδοι/έξοδοι. Στο Mega ,ο SS ακροδέκτης, 53, δεν χρησιμοποιείται για κάποια από τις παραπάνω λειτουργίες, αλλά πρέπει να θεωρείται συνεχώς έξοδος γιατί αλλιώς δεν θα λειτουργεί το περιβάλλον SPI.

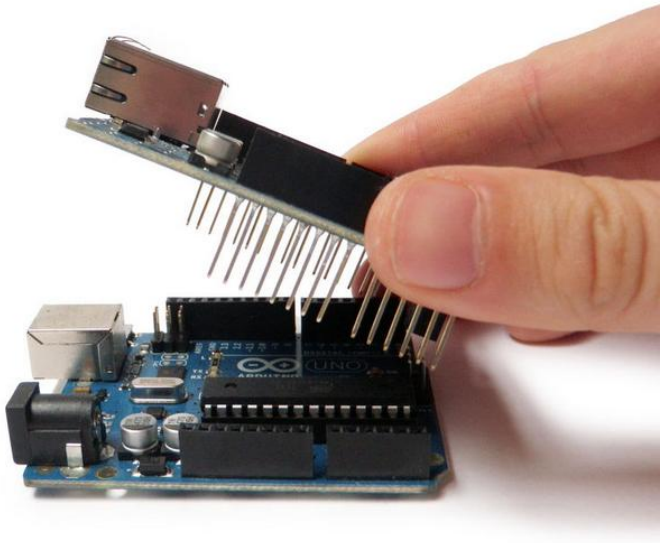
Επειδή το W5100 και η SD κάρτα μοιράζονται τον δίαυλο SPI, δεν μπορούν να λειτουργούν ταυτόχρονα. Αν χρησιμοποιούνται και τα δύο περιφερειακά από το πρόγραμμα, το πότε θα λειτουργεί η καθεμία αναλαμβάνεται από τις αντίστοιχες βιβλιοθήκες. Αν δεν χρησιμοποιείται ένα από τα δύο, πρέπει οπωσδήποτε να απενεργοποιηθεί. Για να γίνει αυτό στην SD κάρτα, θέτουμε τον ακροδέκτη 4 ως HIGH έξοδο, ενώ για το W5100 θέτουμε τον ακροδέκτη 10 ως HIGH έξοδο.

Το reset κουμπί στο Ethernet Shield επανεκκινεί τον W5100 αλλά και το Arduino Board.

Επίσης το Arduino Ethernet Shield έχει έναν αριθμό LED ενδείξεων:

- PWR: Ένδειξη πως η πλακέτα και το Shield τροφοδοτούνται.
- LINK: Ένδειξη πως υπάρχει σύνδεση με το διαδίκτυο και αναβοσβήνει όταν εκπέμπει η δέχεται δεδομένα.
- FULLD: Ένδειξη πως η σύνδεση είναι αμφίδρομη στο ίδιο κανάλι (full duplex).
- 100M: Ένδειξη ότι υπάρχει σύνδεση 100 Mb/s(Αντί για την συνηθισμένη 10 Mb/s).
- RX: Αναβοσβήνει όταν το Shield δέχεται δεδομένα.
- TX: Αναβοσβήνει όταν το Shield στέλνει δεδομένα.
- COLL: Αναβοσβήνει όταν αναγνωρίζονται συγκρούσεις μεταξύ δικτύων.

Στην παρακάτω φωτογραφία (Εικόνα 3.4) φαίνεται ο τρόπος σύνδεσης του Arduino Ethernet Shield με το Arduino Board.



Εικόνα 3.4 - Σύνδεση Ethernet Shield με Arduino.

3.3 Mp3 module

Εικόνα 3.5 – VS1053 MP3 Breakout Board

Η πλακέτα αυτή βασίζεται στον επεξεργαστή VS1053B της VLSI. Ο VS1053 είναι ένας πολύπλευρος κωδικοποιητής MP3 που είναι ικανός να αποκωδικοποιήσει αρκετές μορφές μουσικών αρχείων, περιλαμβάνοντας τα Ogg Vorbis/MP3/AAC/WMA/MIDI audio. Για την καλύτερη δυνατή ακρόαση μέσω ακουστικών, το VS1053 περιλαμβάνει την EarSpeaker επεξεργασία η οποία προσομοιώνει το πώς θα ακουγόταν ένα δωμάτιο με στερεοφωνικά μεγάφωνα. Εκτός του ότι μπορεί να αποκωδικοποιήσει τα επικρατέστερα format ήχου, είναι ικανό να ηχογραφήσει σε μορφή Ogg Vobis.

Το board έχει ενσωματωμένη θύρα για SD κάρτα η οποία μας δίνει τη δυνατότητα αναπαραγωγής MP3 αρχείων.

Χαρακτηριστικά:

- Can play a variety of music formats, support for OGG encoding real-time recording
- SPI interface, the control signal lines are led out
- A headphone and stereo output
- A microphone for recording
- A line_ in input interface
- Power indicator
- 3.3V and 2.8V of LDO chip AMS-1117 on board, provides up to 800 mA current
- A single power supply: +5VDC
- 12.288 MHz crystal
- SD card slot

Μορφές ήχου που αποκωδικοποιεί:

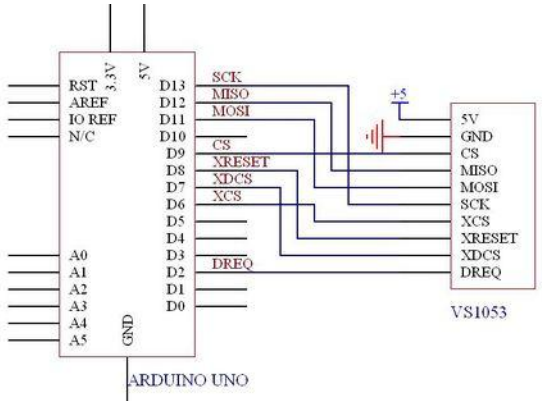
- Ogg Vorbis
- MP3 = MPEG 1&2 audio layer III (CBR+VBR+ABR)
- MP1 & MP2 = MPEG 1&2 audio layers I&II optional
- MPEG4 / 2 AAC-LC(+PNS), HE-AAC v2 (Level 3) (SBR + PS)
- WMA4.0/4.1/7/8/9 all profiles (5-384 kbps)
- FLAC lossless audio with software plugin (up to 24 bits, 48 kHz)
- WAV (PCM + IMA ADPCM)

- General MIDI 1 / SP-MIDI format 0

Μορφές κωδικοποιήσεων της ηχογράφησης μέσω μικροφώνου:

- Ogg Vorbis with software plugin
- IMA ADPCM
- 16-bit PCM

3.3.1 Σύνδεση με το Arduino Uno/Mega



Εικόνα 3.6 – Σύνδεση του VS1053 με το Arduino UNO

Η συνδεσμολογία του VS1053 με τα Arduino UNO και Mega 2560 είναι σχεδόν κοινή με τη μόνη διαφορά να είναι ότι οι ακροδέκτες SCK,MISO και MOSI ,του κωδικοποιητή ήχου, συνδέονται στους ακροδέκτες 50,51,52 του Arduino Mega 2560.
Για να λειτουργήσει σωστά η κάρτα SD και το VS1053 πρέπει να χρησιμοποιηθούν οι βιβλιοθήκες SPI.h για να υπάρξει επικοινωνία μεταξύ του Arduino και του VS1053 board, η SdFat.h και η SdFatUtil.h για την σωστή λειτουργία της SD κάρτας και η βιβλιοθήκη SFEMP3Shield.h για να ενεργοποιηθούν λειτουργίες του επεξεργαστή του breakout board.

Για να προστεθούν οι παραπάνω βιβλιοθήκες αρκεί να προσθέσουμε τις εξής εντολές:

```
#include <SPI.h>
#include <SdFat.h>
#include <SdFatUtil.h>
#include <SFEMP3Shield.h>
```

Επίσης και οι παρακάτω εντολές για να αλλάξουν τα ονόματα για δική μας ευκολία:

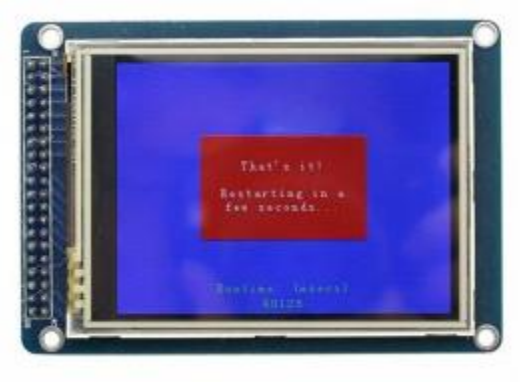
```
SdFat sd;
SFEMP3Shield MP3player;
```

Και στο void setup γίνονται οι αρχικοποιήσεις:

```
Serial.begin(115200); // Έναρξη της σειριακής επικοινωνίας με baudrate 115200
if(!sd.begin(9, SPI_HALF_SPEED)) sd.initErrorHalt();
if (!sd.chdir("/")) sd.errorHalt("sd.chdir"); // Αρχικοποίηση της κάρτας SD
MP3player.begin(); //Εκκίνηση της λειτουργίας του VS1053 breakout Board
```

Αρκετά σημαντικό είναι ότι τα αρχεία MP3 που βρίσκονται στην κάρτα SD πρέπει να είναι ονομασμένα “track001.mp3”, “track002.mp3”, κλπ, για να μπορούν να διαβαστούν από το module.

3.4 Οθόνη Sainsmart 3.2" TFT LCD



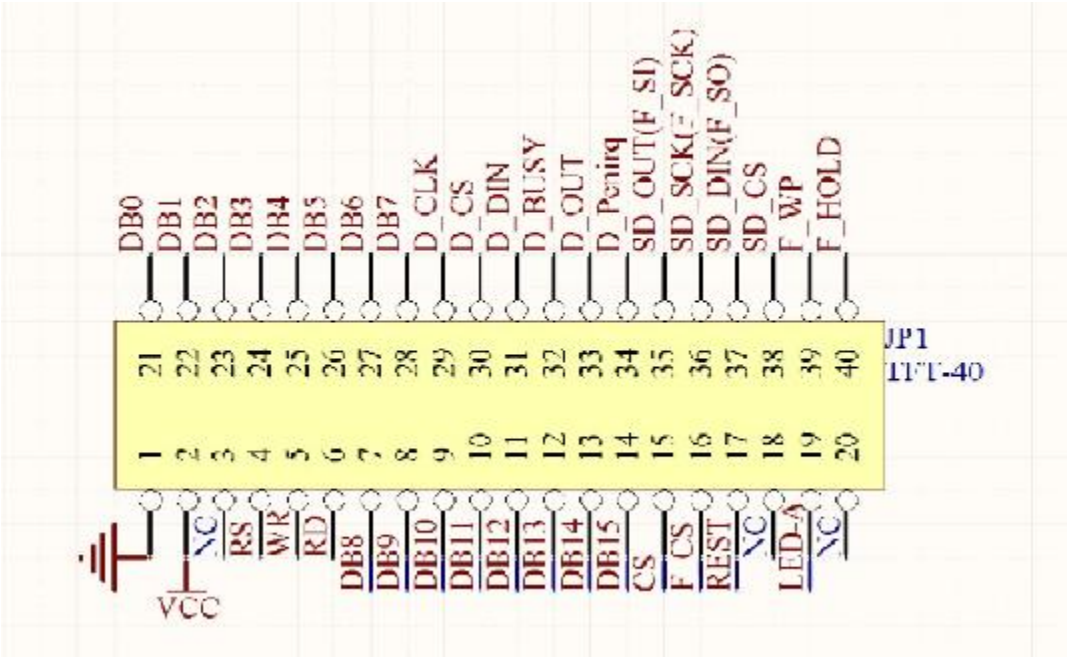
Εικόνα 3.7-Οθόνη Sainsmart 3.2" TFT LCD

Η οθόνη που χρησιμοποιήθηκε είναι μία LCD οθόνη αφής. Έχει σαράντα ακροδέκτες διεπαφών ενσωματωμένο reader για κάρτα SD και Flash. Η οθόνη συμπεριλαμβάνει τον SSD1289 μικροελεγκτή, ο οποίος υποστηρίζει δεδομένα 8/16 bit, και είναι εύκολο να ελεγχτεί από πολλούς μικροελεγκτές όπως ο STM32, ο AVR και ο 8051. Ο μικροελεγκτής της λειτουργίας αφής είναι ο ADS7843, και οι επαφές του συμπεριλαμβάνονται στους σαράντα ακροδέκτες.

Χαρακτηριστικά:

- 3.2" Color TFT LCD
- 320 x 240 Resolution
- 65,536 Colors (16 bit)
- Powerful 16 bit Microcontroller (SSD1289)
- Integrated Resistive Touchscreen
- Integrated SD card
- 5V Operating Voltage
- LED backlight

3.4.1 Περιγραφή των Ακροδεκτών της οθόνης



Εικόνα 3.8 – Διάγραμμα pin της οθόνης

Pin(s)	Περιγραφή
VCC	+5V supply
GND	LCD ground bus
LED-A	LCD backlight supply
DB0-DB15	LCD 16-bit parallel data interface
RS, WR, RD, CS, RST	LCD data flow control lines
D_CLK, D_CS, D_DIN, D_BUSY, D_OUT, D_Pernirq	Touch Panel data flow control lines
SD_OUT(F_SI), SD_SCK(F_SCK),	SD card slot data control lines(SPI)

Βιβλιοθήκες που πρέπει να χρησιμοποιήσουμε για την σωστή λειτουργία της οθόνης.

```
#include <UTouch.h>  // Βιβλιοθήκη για την σωστή λειτουργία της αφής
#include <UTFT.h>   // Βιβλιοθήκη για την σωστή λειτουργία της οθόνης
```

Πρέπει να δηλώσουμε ποια γραμματοσειρά, την οθόνη, και τους ακροδέκτες που θα χρησιμοποιήσουμε.

```
extern uint8_t Retro8x16[];  // Γραμματοσειρά που χρησιμοποιείται
UTFT myGLCD(ITDB32S, 38, 39, 40, 41); // Το ITDB32S είναι η οθόνη και τα νούμερα
στην παρένθεση είναι οι ακροδέκτες στους οποίους έχουμε συνδέσει τα RS, WR, RD και
CS της οθόνης
UTouch myTouch(42, 43, 44, 45, 46);  //τα νούμερα στην παρένθεση είναι οι
ακροδέκτες στους οποίους έχουμε συνδέσει τα D_CLK, D_CS, D_DIN, D_OUT και D_Penirq
που αφορούν την λειτουργία αφής
```

Επίσης στο `void setup()` πρέπει να συμπεριλάβουμε τις παρακάτω γραμμές κώδικα για να γίνει η σωστή αρχικοποίηση της οθόνης.

```
myGLCD.InitLCD();  //Ενεργοποίηση της λειτουργίας της οθόνης
myGLCD.clrScr();   //καθαρισμός της οθόνης
myTouch.InitTouch();  //Ενεργοποίηση της λειτουργίας αφής
myTouch.setPrecision(PREC_MEDIUM);  //Θέτουμε την ευαισθησία της αφής
myGLCD.setFont(Retro8x16);  //Θέτουμε την γραμματοσειρά που θα χρησιμοποιήσουμε
myGLCD.setBackColor(0, 0, 16);  //Θέτουμε το χρώμα του φόντου της οθόνης
```

3.5 Κύκλωμα

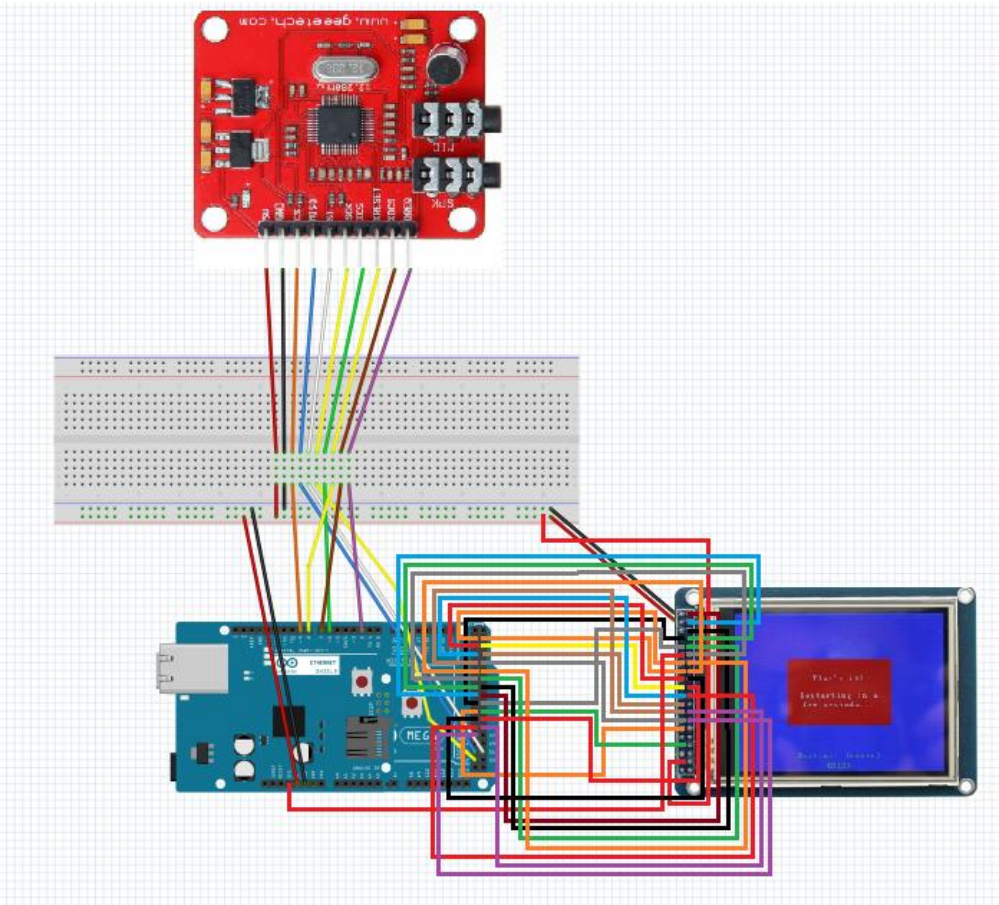
Το κύκλωμα που υλοποιήθηκε αποτελείται από έναν μικροελεγκτή Arduino Mega 2560, το Ethernet Shield για επικοινωνία με το διαδίκτυο και απομακρυσμένο έλεγχο, το VS1053 Mp3 Module με την κάρτα SD για να γίνει εφικτή η αναπαραγωγή μουσικής στο σύστημα, και την 3,2” οθόνη αφής στην οποία έχουν προσομοιωθεί τα κουμπιά με τα οποία χειρίζεται τοπικά το JUKEBOX και εμφανίζονται τα αποτελέσματα και οι πληροφορίες της κάθε αναζήτησης.

Για την υλοποίηση του κυκλώματος πρέπει να συνδεθεί αρχικά το Ethernet Shield στο Arduino και στη συνέχεια το Mp3 Module μέσω ενός breadboard για να είναι πιο εύκολη η σύνδεση με το υπόλοιπο κύκλωμα. Για να λειτουργήσει κατάλληλα το Mp3 Module πρέπει να υλοποιηθεί η παρακάτω συνδεσμολογία.

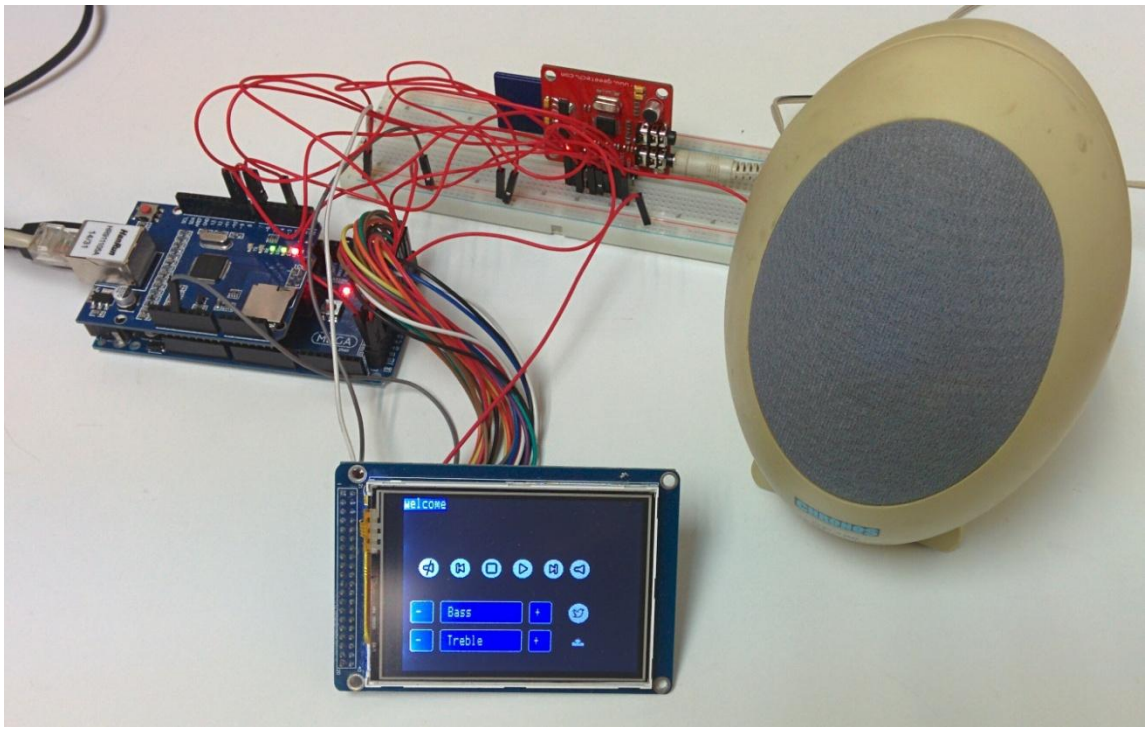
5V → 5V
GND → GND
CS → D9
MISO → D50
MOSI → D51
SCK → D52
XCS → D6
XDCS → D7
DREQ → D2

Μετά θα συνδεθεί η οθόνη αφής κάνοντας την συνδεσμολογία που ακολουθεί.

LEDA → 5V
VCC → 5V
RD → 3.3 V
GND → GND
DB0 έως DB7 αντίστοιχα στα pin D37 έως D30
DB8 έως DB15 αντίστοιχα στα pin D22 έως D29
RS → D38
WR → D39
CS → D40
REST → D41
D_CLK → D42
D_CS → D43
D_IN → D44
D_OUT → D45
D_ Penirq → D46



Εικόνα 3.9-Συνδεσμολογία κυκλώματος που υλοποιήθηκε



Εικόνα 3.10-Το κύκλωμα που υλοποιήθηκε

Κεφάλαιο 4: Λογισμικό Μέρος Συστήματος

4.1 Λογισμικό MP3 player, Weather Station και Twitter

Στην αρχή του προγράμματος γίνεται η εισαγωγή των βιβλιοθηκών. Χρησιμοποιήθηκαν βιβλιοθήκες για την οθόνη αφής, το περιβάλλον SPI, το Ethernet Shield, την κάρτα SD, το MP3 module και η βιβλιοθήκη "TextFinder.h" η οποία μας δίνει τη δυνατότητα να διαβάσουμε τιμές η λέξεις σε οποιοδήποτε κείμενο.

```
#include <UTouch.h> //βιβλιοθήκη για τη λειτουργία αφής της οθόνης
#include <UTFT.h>  //βιβλιοθήκη για τη γενικότερη χρήση της οθόνης
#include <SPI.h>
#include <Ethernet.h>
#include <TextFinder.h>
#include <SdFat.h>
#include <SdFatUtil.h>
#include <SFEMP3Shield.h>
```

Στη συνέχεια θα δηλώσουμε τη γραμματοσειρά και το φόντο που θα χρησιμοποιήσουμε στην οθόνη, να δηλώσουμε σε ποιους ακροδέκτες έχουμε θέσει την κανονική λειτουργία της οθόνης και σε ποιους την λειτουργία της αφής και να δηλώσουμε εξωτερικές μεταβλητές ,όπως εικόνες, αν χρησιμοποιούμε.

```
extern uint8_t Retro8x16[]; // Δήλωση της γραμματοσειράς που χρησιμοποιείται.
UTFT myGLCD(ITDB32S, 38, 39, 40, 41); // Το ITDB32S είναι η οθόνη και δίπλα τα pins
που
UTouch myTouch(42, 43, 44, 45, 46);
extern unsigned int next[1024];
extern unsigned int play[1024];
extern unsigned int prev[1024];
extern unsigned int stop[1024];
extern unsigned int audio[1024];
extern unsigned int twitter[1024];
extern unsigned int weather[1024];
extern unsigned int audiooff[1024];
```

Έπειτα δηλώνουμε τις διευθύνσεις MAC και IP που έχουμε θέσει στο Arduino καθώς και τις πληροφορίες του δικτύου που βρισκόμαστε. Πρέπει επίσης να δηλώσουμε και τις IP των Server των ιστοσελίδων που θα "τραβήξουμε" πληροφορίες.

```
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xAD }; // MAC Address Arduino
byte ip[] = { 195, 251, 14, 230}; // IP Address Arduino
//Πληροφορίες δικτύου
byte gateway[] = { 195, 251, 14, 129 };
byte subnet[] = { 255, 255, 255, 128 };
byte DNS[] = {195, 251, 8, 37};
IPAddress server2(46,43,0,52); // IP Address Twitter Server

char tweet[140]; // Μεταβλητή που ορίζει μέχρι πόσους χαρακτήρες θα είναι το Tweet
που θα διαβαστεί
IPAddress server(188, 125, 93, 38); //IP Address Yahoo Server

EthernetClient client;
TextFinder  finder( client );
```

```
char place[50]; // Μεταβλητές για τις πληροφορίες από το Yahoo Weather
char hum[30];  // Μεταβλητές για τις πληροφορίες από το Yahoo Weather
char date[50]; // Μεταβλητές για τις πληροφορίες από το Yahoo Weather
```

Ακολουθούν τρεις υπορουτίνες, από τις οποίες οι δύο περιέχουν τις συντεταγμένες για το που θα τοποθετηθούν τα κουμπιά στην οθόνη και η τρίτη είναι δείκτης για το ποιο κουμπί πατιέται κάθε φορά.

Εικόνα 4.1-Η διάταξη των κουμπιών στην αρχική οθόνη

```
void drawButtons()
{
   myGLCD.drawBitmap (60, 80, 32, 32, prev);

   myGLCD.drawBitmap (102, 80, 32, 32, stop);

   myGLCD.drawBitmap(144, 80, 32, 32, play);

   myGLCD.drawBitmap(186, 80, 32, 32, next);

   myGLCD.setColor(0,0,255);
   myGLCD.fillRoundRect (10, 142, 40, 172);
   myGLCD.setColor(255,255,255);
   myGLCD.drawRoundRect (10, 142, 40, 172);
   myGLCD.print("-", 17,150);

   myGLCD.setColor(0,0,255);
   myGLCD.fillRoundRect (50, 142, 160, 172);
   myGLCD.setColor(255,255,255);
   myGLCD.drawRoundRect (50, 142, 160, 172);
   myGLCD.print("Bass", 60,150);

   myGLCD.setColor(0,0,255);
   myGLCD.fillRoundRect (170, 142, 200, 172);
   myGLCD.setColor(255,255,255);
   myGLCD.drawRoundRect (170, 142, 200, 172);
   myGLCD.print("+", 178,150);

   myGLCD.setColor(0,0,255);
   myGLCD.fillRoundRect (10, 182, 40, 212);
   myGLCD.setColor(255,255,255);
   myGLCD.drawRoundRect (10, 182, 40, 212);
   myGLCD.print("-", 17,190);

   myGLCD.setColor(0,0,255);
   myGLCD.fillRoundRect (50, 182, 160, 212);
```

```
    myGLCD.setColor(255,255,255);
    myGLCD.drawRoundRect (50, 182, 160, 212);
    myGLCD.print("Treble", 60,190);

    myGLCD.setColor(0,0,255);
    myGLCD.fillRoundRect (170, 182, 200, 212);
    myGLCD.setColor(255,255,255);
    myGLCD.drawRoundRect (170, 182, 200, 212);
    myGLCD.print("+", 177,190);

    myGLCD.drawBitmap(18, 80, 32, 32, audiooff);
    myGLCD.drawBitmap(222, 80, 32, 32, audio);
    myGLCD.drawBitmap(222, 184, 32, 32, weather);
    myGLCD.drawBitmap (222, 142, 32, 32, twitter);

    myGLCD.setBackColor (0,0,0);
}

void drawButtons2()
{
  myGLCD.setColor(0,0,255);
  myGLCD.fillRoundRect (270, 142, 290, 169);
  myGLCD.setColor(255,255,255);
  myGLCD.drawRoundRect (270, 142, 290, 169);
  myGLCD.print("b", 273,145);

  myGLCD.setBackColor (0,0,0);
}
void waitForIt(int x1, int y1, int x2, int y2)
{
  myGLCD.setColor(255, 0, 0);
  myGLCD.drawRoundRect (x1, y1, x2, y2);
  while (myTouch.dataAvailable())
  myTouch.read();
  myGLCD.setColor(255, 255, 255);
  myGLCD.drawRoundRect (x1, y1, x2, y2);
}
```

Στη συνέχεια υπάρχει το `void setup()` το οποίο εκτελείται μόνο μία φορά και περιέχει τις αρχικοποιήσεις του συστήματός μας. Ενεργοποιούμε την σειριακή επικοινωνία, την λειτουργία της οθόνης, το Ethernet Shield, την SD κάρτα και το MP3 module.

```
void setup()
{
  Serial.begin(115200); //Εκκίνηση σειριακής επικοινωνίας

//Αρχικοποιήσεις οθόνης
  myGLCD.InitLCD();
  myGLCD.clrScr();
  myTouch.InitTouch();
  myTouch.setPrecision(PREC_MEDIUM);
  myGLCD.setFont(Retro8x16);
  myGLCD.setBackColor(0, 0, 255);

  Ethernet.begin(mac, ip); //Εκκίνηση λειτουργίας του Ethernet shield

//Αρχικοποιήσεις κάρτας SD
  if (!sd.begin(9, SPI_HALF_SPEED)) sd.initErrorHalt();
```

```
  if (!sd.chdir("/")) sd.errorHalt("sd.chdir");
  sd.ls(LS_DATE | LS_SIZE);

//Αρχικοποιήσεις MP3 module
  uint16_t TrebleFrequency = MP3player.getTrebleFrequency();
  int8_t TrebleAmplitude = MP3player.getTrebleAmplitude();
  MP3player.begin();
  MP3player.setVolume(40, 40);

//Μήνυμα που χαιρετίζει τον χρήστη
  Serial.println(F("Looking for Buttons to be pressed..."));
  myGLCD.print("welcome",0,0);
  delay (5000);
//Εμφάνιση των κουμπιών και άλλων πληροφοριών
  drawButtons();
  Serial.print(F("Former TrebleFrequency = "));
  Serial.println(TrebleFrequency, DEC);
  Serial.print(F("Former TrebleAmplitude = "));
  Serial.println(TrebleAmplitude, DEC);
  union twobyte mp3_vol;
  mp3_vol.word = MP3player.getVolume();
}
```

Η διαδικασία που ακολουθεί είναι το void loop που εκτελείται συνεχώς μέχρι να σταματήσει να τροφοδοτείται το Arduino.

```
void loop()
{
  while (true)
  {
    {
      char title[50]; //buffer για να αποθηκευτεί ο τίτλος του αρχείου που έχει
επιλεχτεί
      char artist[50]; //buffer για να αποθηκευτεί το όνομα του καλλιτέχνη
      char album[50]; //buffer για να αποθηκευτεί το όνομα του άλμπουμ
```

Εδώ αν η οθόνη λειτουργεί κανονικά, ανάλογα τις συντεταγμένες που θα δεχτεί θα αυξήσει ή θα μειώσει τη συχνότητα και το πλάτος του μπάσου.

```
    if (myTouch.dataAvailable())
      {
        myTouch.read();
        x = myTouch.getX();
        y = myTouch.getY();

        uint16_t BassFrequency = MP3player.getBassFrequency();
        uint16_t BassAmplitude = MP3player.getBassAmplitude();
        if ((y >=142) && (y <=172))
         {
          if ((x>=10) && (x<=40))
           {
             waitForIt(10, 142, 40, 172);
             if ((BassFrequency >= 150) && (BassAmplitude >= 15)) {
             BassFrequency = 0;
             BassAmplitude = 0;
            } else
           {
```

```
          BassFrequency -= 10;
          BassAmplitude--;
         }
          MP3player.setBassFrequency(BassFrequency);
          MP3player.setBassAmplitude(BassAmplitude);
          Serial.print(F("New BassFrequency = "));
          Serial.println(MP3player.getBassFrequency(), DEC);
          Serial.print(F("New BassAmplitude = "));
          Serial.println(MP3player.getBassAmplitude(), DEC);
        }
       }
       if ((y >=142) && (y <=172))
        {
         if ((x>=170) && (x<=200))
          {
            waitForIt(170, 142, 200, 172);
            if ((BassFrequency >=150) && (BassAmplitude >= 15))
            {
              BassFrequency = 0;
              BassAmplitude = 0;
             } else
           {
             BassFrequency += 10;
             BassAmplitude++;
            }
            MP3player.setBassFrequency(BassFrequency);
            MP3player.setBassAmplitude(BassAmplitude);
            Serial.print(F("New BassFrequency = "));
            Serial.println(MP3player.getBassFrequency(), DEC);
            Serial.print(F("New BassAmplitude = "));
            Serial.println(MP3player.getBassAmplitude(), DEC);
          }
         }
```

Στη συνέχεια ανάλογα με τις συντεταγμένες που θα δεχτεί θα αλλάξει τη συχνότητα και το πλάτος του Treble.

```
uint16_t TrebleFrequency = MP3player.getTrebleFrequency();
 int8_t TrebleAmplitude = MP3player.getTrebleAmplitude();

 if ((y>=182) && (y<=212))
 {
   if ((x>=10) && (x<=40))
   {
     waitForIt(10, 182, 40, 212);
     if ((TrebleFrequency >= 15000) && (TrebleAmplitude >= 7)) {
        TrebleFrequency = 0;
        TrebleAmplitude = -8;
     } else {
       TrebleFrequency -= 1000;
       TrebleAmplitude--;
     }
     MP3player.setTrebleFrequency(TrebleFrequency);
     MP3player.setTrebleAmplitude(TrebleAmplitude);
     Serial.print(F("New TrebleFrequency = "));
     Serial.println(MP3player.getTrebleFrequency(), DEC);
     Serial.print(F("New TrebleAmplitude = "));
     Serial.println(MP3player.getTrebleAmplitude(), DEC);
```

```
        }
      }
      if ((y>=182) && (y<=212))
      {
        if ((x>=170) && (x<=200))
        {
          waitForIt(170, 182, 200, 212);
          if ((TrebleFrequency >= 15000) && (TrebleAmplitude >= 7)) {
            TrebleFrequency = 0;
            TrebleAmplitude = -8;
          } else {
            TrebleFrequency += 1000;
            TrebleAmplitude++;
          }
          MP3player.setTrebleFrequency(TrebleFrequency);
          MP3player.setTrebleAmplitude(TrebleAmplitude);
          Serial.print(F("New TrebleFrequency = "));
          Serial.println(MP3player.getTrebleFrequency(), DEC);
          Serial.print(F("New TrebleAmplitude = "));
          Serial.println(MP3player.getTrebleAmplitude(),DEC);
        }
      }
```

Εδώ εξετάζουμε την αύξηση ή τη μείωση της έντασης.

```
if ((y >= 80) && (y <= 112))
      {
        if ((x>=222) && (x <= 254))
        {

         waitForIt(222, 80, 254, 112);
         union twobyte mp3_vol;
         mp3_vol.word = MP3player.getVolume();
          if(mp3_vol.byte[1] <= 2) {
             mp3_vol.byte[1] = 2;
            } else {
             mp3_vol.byte[1] -= 2;
            }
            MP3player.setVolume(mp3_vol.byte[1], mp3_vol.byte[1]); // commit new
volume
            Serial.print(F("Volume changed to -"));
            Serial.print(mp3_vol.byte[1]>>1, 1);
            Serial.println(F("[dB]"));
        }
      }
      if ((y >= 80) && (y <= 112))
      {
        if ((x >= 18) && (x <= 50))
        {
          waitForIt(18, 80, 50, 112);
          union twobyte mp3_vol;
          mp3_vol.word = MP3player.getVolume();
          if(mp3_vol.byte[1] >= 254)
          {
            mp3_vol.byte[1] = 254;
          } else {
            mp3_vol.byte[1] +=2;
          }
          MP3player.setVolume(mp3_vol.byte[1], mp3_vol.byte[1]); // commit new
volume
```

```
          Serial.print(F("Volume changed to -"));
          Serial.print(mp3_vol.byte[1]>>1, 1);
          Serial.println(F("[dB]"));
        }
      }
```

Αν πατηθεί το κουμπί που δείχνει τον καιρό θα μεταφερθούμε σε άλλη σελίδα στην οποία θα πάρουμε πληροφορίες από το Yahoo Weather για την υγρασία τη θερμοκρασία και το όνομα της πόλης που βρισκόμαστε.

Εικόνα 4.2-Σελίδα σύνδεσης με το Yahoo Weather

```
if ((y >= 184) && (y<=216))
      {
        if ((x >= 222) && (x<=254))
        {
          waitForIt(222, 184, 254, 216);
          myGLCD.clrScr();
        myGLCD.setFont(Retro8x16);
        if (client.connect(server, 80))
        {
          // Call Wetter-API
          // w: ID from your City
          // http://xm.weather.yahoo.com/forecastrss?w=959401
          ///
          //Serial.println("Connect to Yahoo Weather...");
          client.println("GET /forecastrss?w=959401 HTTP/1.0");
          client.println("HOST:weather.yahooapis.com\n\n");
          client.println();
          myGLCD.print("Connect to Yahoo Weather...", 0, 0);
          myGLCD.print("connected", 0, 16);
        }
        else
        {
          myGLCD.print("connection failed", 0, 0);
        }


        if (client.connected())
        {

          // Humidity
          if ( (finder.getString("<yweather:atmosphere humidity=\"", "\"", hum, 4)
!= 0) )
          {
```

```
            myGLCD.print("Humidity: ", 0, 32);
            myGLCD.print(hum, 160, 32);
            myGLCD.setColor(0, 0, 255);
          }
          else
          {
            Serial.print("No Humidity Data");
            myGLCD.print("No Humidity Data",0,32);
          }


          // Place/City
          if ( (finder.getString("<title>Conditions for ", " ", place, 50) != 0) )
          {
            myGLCD.print("City: ", 0, 48);
            myGLCD.print(place, 96, 48);
          }


          // Temperature
          if (finder.find("temp=") )
          {
            int temperature = finder.getValue();
            temperature = ((temperature - 32) * 5/9);
            myGLCD.print("Temp C: ", 0, 64);
            myGLCD.printNumI(temperature, 138, 64);
          }
          else
          {
            Serial.print("No Temperature Data");
          }

          if ((finder.getString("<pubDate>"," ", date, 50) !=0))
          {
            myGLCD.print("wea: ", 0,80);
            myGLCD.print(date, 32, 80);
            Serial.print(date);
          }
          // END XML
        }
        else
        {
          Serial.println("Disconnected");
        }
        MP3player.stopTrack();// Ενώ βρισκόμαστε στη σελίδα που παρακολουθούμε τον
καιρό σταματάει η                    αναπαραγωγή μουσικής για να αποφύγουμε τυχόν
bugs.
        client.stop();
        client.flush();
        delay(10000); //Έχω επιλέξει να μένει στη σελίδα αυτή για δέκα δευτερόλεπτα
και μετά να επιστρέφει στην αρχικη σελιδα.
        myGLCD.clrScr();
        drawButtons();

      }
    }
```

Αν πατηθεί το κουμπί με το σήμα του Twitter θα μεταφερθούμε σε μία σελίδα που θα μας δείχνει το τελευταίο tweet οποιουδήποτε λογαριασμού επιλέξουμε.

Εικόνα 4.3-Τελευταίο Tweet του λογαριασμού της F1 που επιλέχτηκε

```
if ((y >= 132) && (y<=164))
      {
        if ((x >= 222) && (x<=254))
        {

          waitForIt(144, 100, 176, 132);
          jj:
          myGLCD.clrScr();

        myGLCD.setFont(Retro8x16);
        drawButtons2 ();

    if (client.connect(server2, 80)) {
    client.println("GET https://twitrss.me/twitter_user_to_rss/?user=F1 HTTP/1.0");
// twitter rss for fgranelli
    client.println();
  }
  else {
    myGLCD.print("Connection failed",0,0);
    Serial.println("Connection failed");
  }
  if (client.connected()) {
     // get the last tweet by simply parsing the item and title tags

if((finder.find("<item>")&&(finder.getString("<title>","</title>",tweet,140)!=0)))
     {
         Serial.println(tweet);
          myGLCD.print (tweet,0,0);
       }


    else
      Serial.println("Could not find item field");
  }
    else{
    myGLCD.print("Disconnected",0,16);
  }
  client.stop();
  client.flush();

  delay(60000); // wait a minute before next update
    if ((y >= 142) && (y <= 169))
  {
    if ((x >=270) && (x <= 290))
    {
       waitForIt(270, 142, 290, 169);
```

```
        goto gg;
      }
    }
    goto jj;

        gg:
        myGLCD.clrScr();
        drawButtons();

}
```

Αν πατηθεί το κουμπί play θα ξεκινήσει να παίζει το πρώτο αρχείο στην κάρτα SD και θα εμφανιστούν στην οθόνη οι πληροφορίες για το όνομα του τραγουδιού, του καλλιτέχνη και του άλμπουμ στο οποίο βρίσκεται το κάθε τραγούδι που θα παίζει.

Εικόνα 4.4-Πάτημα κουμπιού play

```
if ((y >= 80) && (y <= 112))
      {
        if ((x >= 144) && (x <= 176)) // Button: Play
        {
          waitForIt(144, 80, 176, 112);
          myGLCD.clrScr();
          drawButtons();
          myGLCD.setFont(Retro8x16);
          myGLCD.setColor(0, 255, 0);
          MP3player.playTrack(current_track);
          MP3player.trackArtist((char*)&artist);
          myGLCD.print("Play Pressed ", 0, 0);
          MP3player.trackTitle((char*)&title);
          Serial.write((byte*)&title, 50);
          Serial.println();
          MP3player.trackAlbum((char*)&album);
          myGLCD.print(title, 0, 16);
          Serial.write((byte*)&artist, 50);
          Serial.println();
          myGLCD.print(artist, 0, 32);
          Serial.write((byte*)&album, 50);
          myGLCD.print(album, 0, 48);
          Serial.println();
        }
      }
```

Αν πατηθεί το κουμπί Stop θα σταματήσει η αναπαραγωγή του κομματιού και θα εμφανίζονται οι πληροφορίες του μέχρι να πατηθεί κάποιο άλλο κουμπί.

Εικόνα 4.5-Πάτημα κουμπιού Stop

```
if ((y >= 80) && (y <= 112))
      {
        if ((x >= 102) && (x <= 134)) // Button: Stop
        {
          waitForIt(102, 80, 134, 112);
          myGLCD.clrScr();
          drawButtons();
          myGLCD.setFont(Retro8x16);
          myGLCD.setColor(255, 0, 0);
          MP3player.stopTrack();
          myGLCD.print("Stop Pressed ", 0, 0);
          myGLCD.print(title, 0, 16);
          myGLCD.print(artist, 0, 32);
          myGLCD.print(album, 0, 48);

        }
      }
```

Αν πατήσουμε το κουμπί Next θα προχωρήσουμε στο επόμενο τραγούδι ενώ αν πατήσουμε το κουμπί Previous θα πάμε στο προηγούμενο.

Εικόνα 4.6-Πάτημα κουμπιού Next

```
if ((y >= 80) && (y <= 112))
     {
       if ((x >= 186) && (x <= 218))// Button: next
       {
```

```
        waitForIt(186, 80, 218, 112);
        myGLCD.clrScr();
        drawButtons();
        myGLCD.setFont(Retro8x16);
        myGLCD.setColor(0, 255, 0);
        MP3player.stopTrack();
        Serial.println(++current_track);
        MP3player.playTrack(current_track);
        myGLCD.print("Next Pressed ", 0, 0);
        MP3player.trackTitle((char*)&title);
        Serial.write((byte*)&title, 50);
        Serial.println();
        myGLCD.print(title, 0, 16);
        MP3player.trackArtist((char*)&artist);
        Serial.write((byte*)&artist, 50);
        Serial.println();
        myGLCD.print(artist, 0, 32);
        MP3player.trackAlbum((char*)&album);
        Serial.write((byte*)&album, 50);
        Serial.println();
        //myGLCD.print("album: ",0,48);
        myGLCD.print(album, 0, 48);
      }
    }
```

Εικόνα 4.7-Πάτημα κουμπιού Previous

```
if ((y >= 80) && (y <= 112))
    {
      if ((x >= 60) && (x <= 92)) // Button: Previous
      {
        waitForIt(60, 80, 92, 112);
        myGLCD.clrScr();
        drawButtons();
        myGLCD.setFont(Retro8x16);
        myGLCD.setColor(0, 255, 0);
        MP3player.stopTrack();
        Serial.println(--current_track);
        MP3player.playTrack(current_track);
        myGLCD.print("Prev Pressed ", 0, 0);
        MP3player.trackTitle((char*)&title);
        Serial.write((byte*)&title, 50);
        Serial.println();
```

```
        myGLCD.print(title, 0, 16);
        MP3player.trackArtist((char*)&artist);
        Serial.write((byte*)&artist, 50);
        Serial.println();
        myGLCD.print(artist, 0, 32);
        MP3player.trackAlbum((char*)&album);
        Serial.write((byte*)&album, 50);
        Serial.println();
        myGLCD.print(album, 0, 48);
      }
    }
```

4.2 Λογισμικό Web Server

Σε αυτή την έκδοση του προγράμματος που υλοποιήθηκε θα εξετάσουμε την δημιουργία ενός Web Server για τον απομακρυσμένο έλεγχο του JUKEBOX μέσω του Web Browser του υπολογιστή ή του κινητού τηλεφώνου (smartphone). Θα υπάρχει δυνατότητα τοπικού ελέγχου επίσης όπως παραπάνω.

Αρχικά δηλώνουμε τις βιβλιοθήκες και θέτουμε διευθύνσεις IP και MAC στο Arduino και τα ονόματα των μεταβλητών.

```
#include <UTouch.h> //βιβλιοθήκη για τη λειτουργία αφής της οθόνης
#include <UTFT.h>  //βιβλιοθήκη για τη γενικότερη χρήση της οθόνης
#include <SPI.h>
#include <Ethernet.h>
#include <TextFinder.h>
#include <SdFat.h>
#include <SdFatUtil.h>
#include <SFEMP3Shield.h>

extern uint8_t Retro8x16[]; // Δήλωση της γραμματοσειράς που χρησιμοποιείται.
UTFT myGLCD(ITDB32S, 38, 39, 40, 41); // Το ITDB32S είναι η οθόνη και δίπλα τα pins
που
UTouch myTouch(42, 43, 44, 45, 46);
extern unsigned int next[1024];
extern unsigned int play[1024];
extern unsigned int prev[1024];
extern unsigned int stop[1024];
extern unsigned int audio[1024];
extern unsigned int twitter[1024];
extern unsigned int weather[1024];
extern unsigned int audiooff[1024];

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xAD }; // MAC Address Arduino
byte ip[] = { 195, 251, 14, 230}; // IP Address Arduino
//Πληροφορίες δικτύου
byte gateway[] = { 195, 251, 14, 129 };
byte subnet[] = { 255, 255, 255, 128 };
byte DNS[] = {195, 251, 8, 37};
EthernetServer server(80); // Δήλωση του Server
```

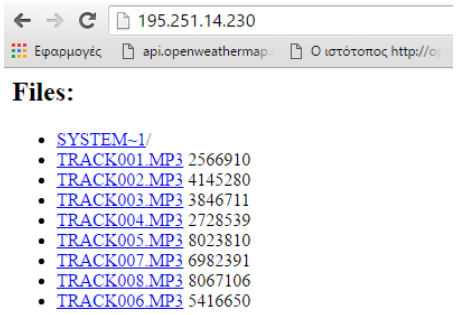
Ακολουθεί το `void setup()` που εκτελείται μόνο την πρώτη φορά που τρέχει το πρόγραμμα.

```
void setup() {
  Serial.begin(115200); //Εκκίνηση σειριακής επικοινωνίας

//Αρχικοποιήσεις οθόνης
```

```
  myGLCD.InitLCD();
  myGLCD.clrScr();
  myTouch.InitTouch();
  myTouch.setPrecision(PREC_MEDIUM);
  myGLCD.setFont(Retro8x16);
  myGLCD.setBackColor(0, 0, 255);

  Serial.print(F("Free RAM: "));
  Serial.println(FreeRam());

  // εκκίνηση της λειτουργίας της SD κάρτας σε SPI_HALF_SPEED για να αποφύγουμε
τυχόν σφάλματα με το
  // breadboard.  Χρήση SPI_FULL_SPEED για καλύτερη απόδοση.
  pinMode(10, OUTPUT); // Θέτουμε το SS pin ως έξοδο για τη σωστή λειτουργία της SD
κάρτας
  digitalWrite(10, HIGH); // Θέτουμε το W5100 chip του Ethernet Shield εκτός
λειτουργίας

  pinMode(8, OUTPUT);
  digitalWrite(8, LOW); // θέτουμε το VS1053 σε επανεκκίνηση

  if(!sd.begin(9, SPI_HALF_SPEED)) sd.initErrorHalt();
  if (!sd.chdir("/")) sd.errorHalt("sd.chdir");

  Serial.print(F("Volume is FAT"));
  Serial.println(sd.vol()->fatType(),DEC);
  Serial.println();

  // Εμφάνισε τη λίστα με τα αρχεία με ημερομηνία και μέγεθος
  Serial.println(F("Files found in root:"));
  sd.ls(LS_DATE | LS_SIZE);
  Serial.println();


  Serial.println(F("Done"));

  // Ολοκλήρωση Debugging, εκκίνηση Server
  Ethernet.begin(mac, ip);
  server.begin();

//Αρχικοποιήσεις MP3 module
  uint16_t TrebleFrequency = MP3player.getTrebleFrequency();
  int8_t TrebleAmplitude = MP3player.getTrebleAmplitude();

  result = MP3player.begin();
  //Έλεγχος του result,δηλαδή έλεγχος για σωστή εκκίνηση του MP3 module
  if(result != 0) {
    Serial.println(F("Error code: "));
    Serial.print(result);
    Serial.println(F(" when trying to start MP3 player"));
    }
  MP3player.setVolume(40,40);
//Εμφάνιση των κουμπιών και άλλων πληροφοριών
  drawButtons();
  Serial.print(F("Former TrebleFrequency = "));
  Serial.println(TrebleFrequency, DEC);
  Serial.print(F("Former TrebleAmplitude = "));
  Serial.println(TrebleAmplitude, DEC);
  union twobyte mp3_vol;
```

```
  mp3_vol.word = MP3player.getVolume();

}
```

Ακολουθεί υπορουτίνα η οποία εμφανίζει τα αρχεία της κάρτας SD. Για να είναι πιο εύκολο εμφανίζει μόνο τα αρχεία του κύριου φακέλου.

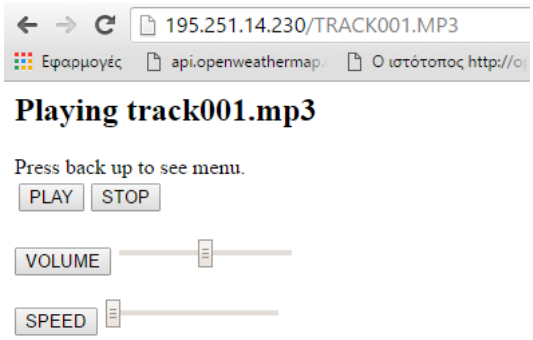


Εικόνα 4.8-Αρχική σελίδα του Web Server

```
void ListFiles(EthernetClient client, uint8_t flags) {

  dir_t p;

  sd.vwd()->rewind(); // Εδώ κάνουμε reset κατεύθυνση στον αρχικό φάκελο
  client.println(F("<ul>"));
  while (sd.vwd()->readDir(&p) > 0) {
    // done if past last used entry
    if (p.name[0] == DIR_NAME_FREE) break;

    // προσπέρασε διαγραμμένα ονόματα
    if (p.name[0] == DIR_NAME_DELETED || p.name[0] == '.') continue;

    // εμφάνισε μόνο υποκαταλόγους και φακέλους
    if (!DIR_IS_FILE_OR_SUBDIR(&p)) continue;

    // print any indent spaces
    client.print(F("<li><a href=\""));
    for (uint8_t i = 0; i < 11; i++) {
      if (p.name[i] == ' ') continue;
      if (i == 8) {
        client.print('.');
      }
      client.print((char)p.name[i]);
    }
    client.print(F("\">"));

    // Ακολουθεί η εκτύπωση του ονόματος με τα πιθανά κενά
    for (uint8_t i = 0; i < 11; i++) {
      if (p.name[i] == ' ') continue;
      if (i == 8) {
        client.print('.');
      }
      client.print((char)p.name[i]);
    }
```

```
    client.print(F("</a>"));

    if (DIR_IS_SUBDIR(&p)) {
      client.print('/');
    }

    // Εμφάνισε ή μετέτρεψε την ώρα/ημερομηνία αν ζητηθεί
    if (flags & LS_DATE) {
       sd.vwd()->printFatDate(p.lastWriteDate);
       client.print(' ');
       sd.vwd()->printFatTime(p.lastWriteTime);
    }
    // Εμφάνισε το μέγεθος του αρχείου αν ζητηθεί
    if (!DIR_IS_SUBDIR(&p) && (flags & LS_SIZE)) {
      client.print(' ');
      client.print(p.fileSize);
    }
    client.println(F("</li>"));
  }
  client.println(F("</ul>"));
}
```

Η παρακάτω διαδικασία είναι το void loop που εκτελείται συνεχώς μέχρι να σταματήσει να τροφοδοτείται το Arduino. Ο κώδικας που εκτελείται έχει τις εξής λειτουργίες. Αρχικά δημιουργούμε έναν buffer στον οποίο θα αποθηκεύονται με τη σειρά οι χαρακτήρες του ονόματος του κάθε αρχείου μέχρι να δεχτούμε τον χαρακτήρα \n ή \r που σημαίνει ότι πρέπει να αλλάξει γραμμή. Για να τελειώσει η συμβολοσειρά (string) πρέπει να τοποθετηθεί ένας κενός χαρακτήρας (0) στο τέλος. Στη συνέχεια χρησιμοποιείται η εντολή `strstr` για να γίνει αναζήτηση για συμβολοσειρές σε υποφακέλους. Αν δεχτούμε ένα αίτημα “GET / HTTP” συνεχίζουμε την εκτύπωση του ονόματος όπως πριν. Αν μετά το “GET /” δεν υπάρχει κενό όπως για παράδειγμα “GET /file” θα πρέπει να εξάγουμε το όνομα του αρχείου. Δημιουργούμε έναν δείκτη στη συμβολοσειρά και τον ξεκινάμε μετά την κάθετο. Μετά αναζητούμε την αρχή της συμβολοσειράς “HTTP/1.1” που ακολουθεί το αίτημα για την ανάγνωση του ονόματος του αρχείου και μετατρέπουμε τον πρώτο χαρακτήρα σε κενό χαρακτήρα, για να έχουμε το όνομα του αρχείου. Αν αποτύχει το άνοιγμα του φακέλου ένα σφάλμα 404 θα συμβεί. Αλλιώς, θα εμφανιστούν όλα τα περιεχόμενα του φακέλου.



Εικόνα 4.9-Μενού επιλογών μετά την επιλογή κομματιού

```
void loop()
```

```
{
  char clientline[BUFSIZ];// Το BUFSIZ δηλώνεται πριν το void loop (#define BUFSIZ
100) και είναι το επιλεγμένο μέγεθος της κάθε γραμμής
  int index = 0;

  EthernetClient client = server.available();
  if (client) {
    // Ένα http request τελείωνει πάντα με μία κενή γραμμή
    boolean current_line_is_blank = true;

    // Αρχικοποίηση του buffer εισαγωγής
    index = 0;

    while (client.connected()) {
      if (client.available()) {
        char c = client.read();

        // Αν δεν δηλώνεται καινούρια γραμμή, πρόσθεσε τον χαρακτήρα στον buffer
        if (c != '\n' && c != '\r') {
          clientline[index] = c;
          index++;
          // Αν η λέξη είναι πολύ μεγάλη για τον buffer πρέπει να αφαιρέσουμε
δεδομένα
          if (index >= BUFSIZ)
            index = BUFSIZ -1;

          // Συνέχιση ανάγνωσης
          continue;
        }

        // Αν δεχτούμε \n ή \r ,σημαίνει ότι τελείωσε η εμφάνιση του ονόματος
        clientline[index] = 0;

        // Εμφάνιση για να γίνει debugging
        Serial.println(clientline);

        // Look for substring such as a request to get the root file
        if (strstr(clientline, "GET / ") != 0) {
          // send a standard http response header
          client.println(F("HTTP/1.1 200 OK"));
          client.println(F("Content-Type: text/html"));
          client.println();

          // Εμφάνιση όλων των αρχείων
          client.println(F("<h2>Files:</h2>"));
          ListFiles(client, LS_SIZE);

        } else if (strstr(clientline, "GET /") != 0) {
          // Δεν υπάρχει κενό μετά το /, άρα αναφερόμαστε σε υποφάκελο
          char *filename;

          filename = clientline + 5; // Έλεγχος 5 χαρακτήρων μετά το  "GET /"
          // Για να γίνει καθαρισμός ελέγχουμε την " HTTP/1.1" string και
          // και μετατρέπουμε τον πρώτο χαρακτήρα σε 0.
          (strstr(clientline, " HTTP"))[0] = 0;

          // Εμφάνιση του ονόματος του αρχείου που θέλουμε
          Serial.println(filename);
```

```
            char *get_arg;
            get_arg = strstr(filename, "?");
            Serial.println(get_arg);
            if (get_arg) {
              get_arg[0] = 0;
              get_arg += 5;
              Serial.println(F("get arg invoked "));
              Serial.println(get_arg);
              Serial.println(filename);
              client.println("HTTP/1.1 204 OK");
              client.println();

              // Ακολουθούν οι εντολές που θα υπάρχουν στην ιστοσελίδα

              //PLAY
              if (strstr(get_arg, "PLAY") != 0) {
                result = MP3player.playMP3(filename);
                if(result != 0) {
                  Serial.print(F("Error code: "));
                  Serial.print(result);
                  Serial.println(F(" when trying to play track"));
                }

              // STOP
              } else if (strstr(get_arg, "STOP") != 0) {
                MP3player.stopTrack();

              //PAUSE
              } else if (strstr(get_arg, "PAUSE") != 0) {
                if( MP3player.getState() == playback) {
                  MP3player.pauseMusic();
                  Serial.println(F("Pausing"));
                } else if( MP3player.getState() == paused_playback) {
                  MP3player.resumeMusic();
                  Serial.println(F("Resuming"));
                } else {
                  Serial.println(F("Not Playing!"));
                }

              // VOLUME
              } else if(strstr(get_arg, "VOLUME") != 0) {
                int8_t val = atol(strstr(get_arg, "VAL=") + 4);
                Serial.print(F("Val="));
                Serial.println(val, DEC);

                // push byte[1] into both left and right assuming equal balance.
                MP3player.setVolume(val, val); // Αλλαγή της έντασης
                Serial.print(F("Volume changed to -"));
                Serial.print(val>>1, 1);
                Serial.println(F("[dB]"));

              // PLAY SPEED
              } else if(strstr(get_arg, "SPEED") != 0) {
                int16_t val = atol(strstr(get_arg, "VAL=") + 4);
                Serial.print(F("Val="));
                Serial.println(val, DEC);

                MP3player.setPlaySpeed(val); // Εισαγωγή νέας ταχύτητας αναπαραγωγής
του τραγουδιού
```

```
            Serial.print(F("playspeed to "));
            Serial.println(val, DEC);

          }

          //
        } else if (  strstr(strlwr(filename), ".mp3")
           || strstr(strlwr(filename), ".aac")
           || strstr(strlwr(filename), ".wma")
           || strstr(strlwr(filename), ".wav")
           || strstr(strlwr(filename), ".fla")
           || strstr(strlwr(filename), ".mid")
          ) {

          // Ενεργοποίηση της ιστοσελίδας
          client.println(F("HTTP/1.1 200 OK"));
          client.println(F("Content-Type: text/html"));
          client.println(F("<html><head></head><body>"));

          client.println();
          client.print(F("<h2>Playing "));
          client.print(filename);
          client.println(F("</h2>"));
          client.println(F("Press back up to see menu."));

          client.println(F("<table>"));
          client.println(F("<tr><td>"));
          ButtonForm(client, "CMD", "PLAY");
          client.println(F("</td><td>"));
//          ButtonForm(client, "CMD", "PAUSE");
//          client.println(F("</td><td>"));
          ButtonForm(client, "CMD", "STOP");
          client.println(F("</table>"));

          client.println(F("<form method=get>"));
          client.println(F("<input type='submit' name='CMD' value='VOLUME'>"));
          client.println(F("<input id='VAL' name='VAL' type='range' min='0'
max='100' >"));
          client.println(F("</form>"));

          client.println(F("<form method=get>"));
          client.println(F("<input type='submit' name='CMD' value='SPEED'>"));
          client.println(F("<input id='VAL' name='VAL' type='range' min='1'
max='4' value ='1'>"));
          client.println(F("</form>"));

          Serial.print(F("Music file:"));
          Serial.println(filename);

        } else {
          if (! file.open(filename, O_READ)) {
            client.println(F("HTTP/1.1 404 Not Found"));
            client.println(F("Content-Type: text/html"));
            client.println();
            client.println(F("<h2>File Not Found!</h2>"));
            break;
          }

          Serial.println(F("Opened!"));
```

```
            client.println(F("HTTP/1.1 200 OK"));
//            client.println(F("Content-Type: text/plain"));
            client.println(F("Content-Type: "));
            Serial.println(F("Content-Type: "));
            if (strstr(strlwr(filename), ".htm") != 0) { //Sets content type.
              client.println(F("text/html"));
              Serial.println(F("text/html"));
            } else if (strstr(strlwr(filename), ".css") != 0) {
              client.println(F("text/css"));
              Serial.println(F("text/css"));
            } else if (strstr(strlwr(filename), ".png") != 0) {
              client.println(F("image/png"));
              Serial.println(F("image/png"));
            } else if (strstr(strlwr(filename), ".jpg") != 0) {
              client.println(F("image/jpeg"));
              Serial.println(F("image/jpeg"));
            } else if (strstr(strlwr(filename), ".gif") != 0) {
              client.println(F("image/gif"));
              Serial.println(F("image/gif"));
            } else if (strstr(strlwr(filename), ".3gp") != 0) {
              client.println(F("video/mpeg"));
              Serial.println(F("video/mpeg"));
            } else if (strstr(strlwr(filename), ".pdf") != 0) {
              client.println(F("application/pdf"));
              Serial.println(F("application/pdf"));
            } else if (strstr(strlwr(filename), ".js") != 0) {
              client.println(F("application/x-javascript"));
              Serial.println(F("application/x-javascript"));
            } else if (strstr(strlwr(filename), ".xml") != 0) {
              client.println(F("application/xml"));
              Serial.println(F("application/xml"));
            } else {
              client.println(F("text"));
              Serial.println(F("text"));
            }
            client.println();

            int16_t c;
            while ((c = file.read()) > 0) {
                // uncomment the serial to debug (slow!)
                Serial.print((char)c);
                client.print((char)c);
            }
            file.close();
          }
        } else {
          // οτιδήποτε άλλο είναι σφάλμα 404
          client.println(F("HTTP/1.1 404 Not Found"));
          client.println(F("Content-Type: text/html"));
          client.println();
          client.println(F("<h2>File Not Found!</h2>"));
        }
        break;
      }
    }
    // καθυστέρηση προγράμματος για να γίνει σωστή επεξεργασία δεδομένων
    delay(1000);
    client.stop();
```

```
    }
}
```

Κεφάλαιο 5: Συμπεράσματα και Μελλοντικές Επεκτάσεις

Η υλοποίηση της πτυχιακής εργασίας με βοήθησε να ανακαλύψω τους μικροελεγκτές Arduino και τις πολλές δυνατότητες που προσφέρουν σε ερασιτέχνες και επαγγελματίες προγραμματιστές. Μέσω της πλατφόρμας αυτής εξοικειώθηκα, μετά από αρκετές ώρες διαβάσματος και δοκιμών, στον προγραμματισμό σε γλώσσες που δεν είχα προηγούμενη επαφή αλλά και σε πιο πρακτικούς τομείς όπως η συγκόλληση καλωδίων και ανάγνωση πιθανών συνδεσμολογιών για την επιλογή της καλύτερης.

Το τελικό αποτέλεσμα, είναι μία συσκευή πολυμέσων με κύρια λειτουργία την αναπαραγωγή μουσικής η οποία ελέγχεται τοπικά, μέσω κουμπιών που εξομοιώθηκαν σε μία οθόνη αφής, και απομακρυσμένα μέσω μίας ιστοσελίδας που δημιουργήσαμε. Για να γίνει το τελευταίο χρησιμοποιήσαμε μία επέκταση στο Arduino η οποία δίνει τη δυνατότητα σύνδεσης μέσω Ethernet στο διαδίκτυο. Οι υπόλοιπες λειτουργίες είναι η σύνδεση με τη σελίδα Yahoo Weather για την παρουσίαση πληροφοριών για την πόλη, την υγρασία, και την θερμοκρασία καθώς και η σύνδεση με το Tweeter.

Τέλος, θα ήθελα να αναφέρω μερικές βελτιώσεις και επεκτάσεις που θα μπορούσαν να εφαρμοστούν μελλοντικά στο σύστημά μας:

- Μείωση του όγκου του συστήματος για τη δυνατότητα δημιουργίας φορητής συσκευής.
- Χρησιμοποίηση του WiFi Shield για το Arduino για να μην υπάρχει ο περιορισμός της σύνδεσης με καλώδιο Ethernet.
- Δημιουργία λειτουργίας ραδιοφώνου μέσω internet.
- Περαιτέρω ανάπτυξη της ιστοσελίδας, όπως π.χ. αναγνώριση τραγουδιών και πληροφοριών για τους καλλιτέχνες (βιογραφίες, νέα κλπ.)

Βιβλιογραφία

Βιβλία:

Jeremy Blum, Exploring Arduino: Tools and Techniques for Engineering Wizardry, John Wiley & Sons, Inc. , 2013

Διαδίκτυο:

3.2” TFT LCD with SD and Touch for Arduino Mega with adjustable TFT shield Quickstart Guide, http://sainsmart.com/skin/frontend/base/default/document/3_2%20inch%20TFT%20with%20SD%20and%20Touch%20Quickstart.pdf

https://el.wikipedia.org/wiki/HTML

Atmel ATmega640/V-1280/V-1281/V-2560/V-2561/V Datasheet, http://www.atmel.com/images/atmel-2549-8-bit-avr-microcontroller-atmega640-1280-1281-2560-2561_datasheet.pdf

http://www.arduino.cc/en/Main/ArduinoBoardMega2560

http://www.arduino.cc/en/Main/ArduinoEthernetShield

http://www.arduino.cc/en/Main/Products

http://www.sainsmart.com/sainsmart-3-2-tft-lcd-display-touch-panel-pcb-adapter-sd-slot-for-arduino-2560.html

http://www.geeetech.com/wiki/index.php/Arduino_MP3_shield_board_with_TF_card

http://inventors.about.com/od/mstartinventions/a/MPThree.htm