ΤΕΙ ΚΡΗΤΗΣ ΠΑΡΑΡΤΗΜΑ ΡΕΘΥΜΝΟΥ

ΤΜΗΜΑ ΜΟΥΣΙΚΗΣ ΤΕΧΝΟΛΟΓΙΑΣ

ΚΑΙ ΑΚΟΥΣΤΙΚΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΘΕΜΑ:ΚΑΤΑΣΚΕΥΗ ΣΥΣΚΕΥΗΣ ΕΛΕΓΧΟΥ ΦΩΤΙΣΜΟΥ ΜΕΣΩ ΠΡΩΤΟΚΟΛΛΟΥ DMX 512

ΟΝΟΜΑ : ΝΤΟΚΟΣ ΚΩΝ/ΝΟΣ

ΑΜ:327

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ:ΧΟΥΣΙΔΗΣ ΧΡΗΣΤΟΣ

ΡΕΘΥΜΝΟ 2008

ΠΕΡΙΕΧΟΜΕΝΑ

ΛΙΣΤΑ ΕΙΚΟΝΩΝ

ΛΙΣΤΑ ΠΙΝΑΚΩΝ

1.ΕΙΣΑΓΩΓΗ

Σκοπός της εργασίας αυτής είναι να γίνει μια γνωριμία με της τεράστιες δυνατότητες που προσφέρουν οι μικροελεγκτες μέσω του προγραμματισμού τους. Καθώς επίσης και με το πρωτόκολλο DMX 512, που είναι το πλέον διαδεδομένο πρωτόκολλο στις μέρες μας για την επικοινωνία και τον έλεγχο των συσκευών φωτισμού. Με αυτήν την εργασία διαπιστώνουμε ότι μέσω των δυνατοτήτων που μας δίνουν οι μικροελεγκτες μπορούμε να κατασκευάσουμε οποιαδήποτε format ψηφιακού σήματος. Στην εφαρμογή που παρουσιάζεται έχουμε πετύχει την δημιουργία του DMX 512 σήματος. Σε αυτό το πακέτο φορτώνουμε σαν data τις πληροφορίες που περνούμε από ένα αναλογικό σήμα εισόδου. Αυτό το αναλογικό σήμα μετατρέπεται σε ψηφιακό σήμα μέσω το A/D μετατροπείς που έχει ενσωματωμένους ο μικροελεγκτης ATmega163. Στην συγκεκριμένη εφαρμογή ο προγραμματισμός του μικροελεγκτη ATmega 163 γίνεται σε γλωσσά προγραμματισμού assembly (γλωσσά μηχανής). Ο assembly κώδικας παρουσιάζεται στην συνεχεία της εργασίας.

2.ΑΝΑΛΥΣΗ ΠΡΟΔΙΑΓΡΑΦΩΝ DMX512

2.1 DMX512 ΙΣΤΟΡΙΚΑ

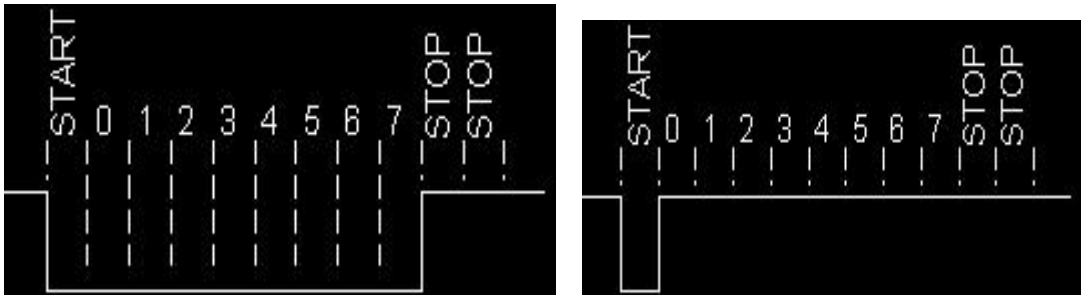
Το πρωτόκολλο DMX επινοήθηκε το 1986 από τον USITT (United States Institute of Theater Technology). Ο αμερικανικός αυτός οργανισμός είναι αρμόδιος για την τυποποίηση τεχνικών προδιαγραφών που αφορούν τον τεχνικό εξοπλισμό τον θεατρικών και των κινηματογραφικών αιθουσών. Ειδικότερα το DMX ορίζει τους τρόπους και τις παραμέτρους επικοινωνίας των ρυθμιστικών έντασης φωτισμού και των αυτομάτων προβολέων που επικοινωνούν ψηφιακά με μια κονσόλα ελέγχου. Μετά την παρουσίαση του πρώτου προτύπου DMX το 1986 έγιναν κάποιες τροποποιήσεις διορθώσεις το 1990 έτσι ώστε να λυθούν κάποια προβλήματα , και τώρα είναι γνωστό ως πρότυπο USITT DMX512(1990).

Πριν την δημιουργία του DMX 512 ο έλεγχος των ρυθμιστικών έντασης φωτισμού και των προβολέων γίνονταν από διαφορά ψηφιακά πρωτοκολλά η από πολύπλοκες αναλογικές συνδέσεις που ήταν διαφορετικές για κάθε κατασκευαστή. Αυτό είχε ως αποτέλεσμα την ασυμβατότητα των συσκευών που ήταν από διαφορετικούς κατασκευαστές, οδηγώντας στον περιορισμό των λύσεων καθώς και στην αύξηση του κόστους [1] [2].

Ωστόσο το DMX 512 δεν είναι η τελεία λύση για τον έλεγχο του φωτισμού αλλά πλέον είναι η πιο ευρέως διαδεδομένη. Η σχεδίαση είναι πολύ απλή καθώς είχε ως σκοπό να πείσει των μεγάλο αριθμό κατασκευαστών να το υιοθετήσουν ως το πρωτόκολλο επικοινωνίας των συσκευών τους. Αυτή η απλή σχεδίαση του DMX512 ήταν πολύ ελκυστική για τους κατασκευαστές καθώς μείωνε τις μεγάλες επενδύσεις για ερευνά, καθώς επίσης και τις δραστικές επανασχεδιάσεις των ήδη υπαρχόντων κατασκευών τους.

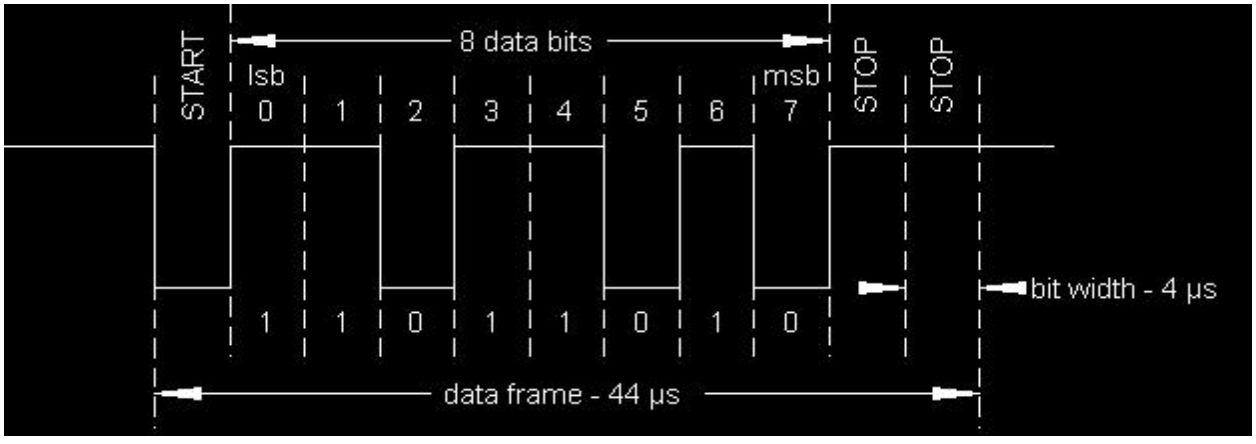
2.2 ΠΡΟΔΙΑΓΡΑΦΕΣ DMX512

Το βασικό στοιχειό κάθε πρωτόκολλου επικοινωνίας είναι ένα σύνολο από κώδικες. Κάθε κώδικας είναι μια μοναδική σειρά από “high” και “low” σήματα που ονομάζονται “bits”. Το κάθε “bit “έχει προκαθορισμένη διάρκεια που για το DMX512 4μsec. Στο DMX512 κάθε κώδικας αποτελείται από 8 bits και ονομάζεται “byte”. Τα 8 bits του κάθε byte επιτρέπουν 256 διαφορετικούς συνδυασμούς.



Εικόνα 2.2.1 : start και stop bit

Στο πρωτόκολλο DMX512 εκτός από τα 8 “bits”υπάρχουν αλλά τρία “bits” που χρησιμοποιούνται για τον διαχωρισμό των “bytes”. Αυτά είναι ένα start bit(low κατάσταση) και δυο stop bits(high κατάσταση)(εικ 2.2.1). Κατά συνέπεια συνολικά υπάρχουν 11 bits για κάθε byte. Αυτός ο κώδικας των 11 bits ονομάζεται “frame” (εικ 2.2.2) και έχει διάρκεια 44 μsec(11 bits*4μsec = 44μsec).



Εικόνα 2.2.2: data frame

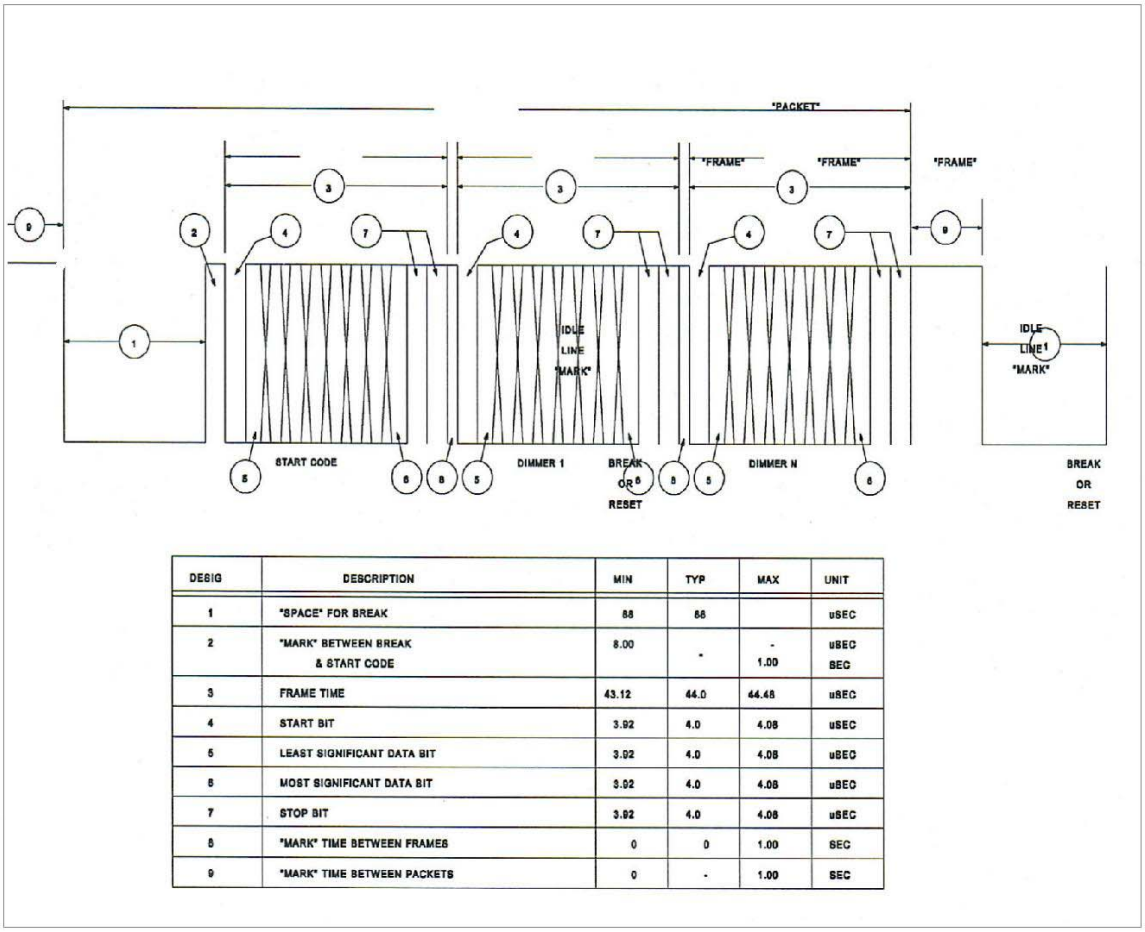
Στο DMX512 ο χρόνος μετάδοσης του κάθε bit είναι 4 μsec αυτό επιτρέπει να έχουμε ταχύτητα μετάδοσης 250000 bit το δευτερόλεπτο που είναι γνωστό με την ονομασία baud rate του DMX 512 και συνήθως δηλώνεται ως 250 kbps.

Το DMX 512 είναι ένα ασύγχρονο πρωτόκολλο που σημαίνει ότι το κάθε frame μπορεί να μεταδοθεί κάθε φορά που η παλμοσειρα είναι σε κατάσταση high “idle”. Στην πράξη οι περισσότερες κονσόλες φωτισμού, τακτικά η περιστασιακά, στέλνουν “idle gap”(θέτουν την παλμοσειρα σε κατάσταση high) μεταξύ των frame. Οι κονσόλες φωτισμού εισάγουν αυτό το idle gap διότι είναι απασχολημένες με την εκτελέσει κάποιον υπολογισμών έτσι ώστε να δεχτούν το επόμενο frame αφού έχει τελειώσει η μετάδοση του προηγούμενου frame.

Το DMX512 δεν επιτρέπει να μεταδοθούν παραπάνω από 512 κανάλια σε μια DMX 512 διασύνδεση. Υπάρχουν κάποιοι τύποι συσκευών ανάλογη των DMX μετατροπέων όπου έτσι και λειτουργήσουν μαζί μπορεί ο αριθμός των μεταδιδόμενων καναλιών να ξεπεράσει τα 512 .

Η χρήση παραπάνω από 512 κανάλια σε μια DMX διασύνδεση δεν είναι μέρος των στάνταρ της USSIT και μπορεί να είναι αίτια προβλημάτων σε μετρικούς δέκτες. Κονσόλες όπου παρέχουν παραπάνω από 512 dimmers έχουν εγκατεστημένα δυο η και παραπάνω DMX εξόδους [2]. Τα κανάλια του DMX δεν πρέπει να μπερδεύονται με τα κανάλια της κονσόλας η τα κανάλια των dimmer.

Στην εικόνα (2.2.3) παρουσιάζονται οι διάρκειες των μερών ενός DMX512 πακέτου όπως έχουν προσδιοριστεί από την USSIT.



DESIG	DESCRIPTION	MIN	TYP	MAX	UNIT
1	"SPACE" FOR BREAK	88	88		uSEC
2	"MARK" BETWEEN BREAK & START CODE	8.00	-	- 1.00	uSEC SEC
3	FRAME TIME	43.12	44.0	44.48	uSEC
4	START BIT	3.92	4.0	4.08	uSEC
5	LEAST SIGNIFICANT DATA BIT	3.92	4.0	4.08	uSEC
6	MOST SIGNIFICANT DATA BIT	3.92	4.0	4.08	uSEC
7	STOP BIT	3.92	4.0	4.08	uSEC
8	"MARK" TIME BETWEEN FRAMES	0	0	1.00	SEC
9	"MARK" TIME BETWEEN PACKETS	0	-	1.00	SEC

Εικόνα 2.2.3: Διαρκειες DMX512

- ***BREAK TIMING:***

Η διάρκεια του break είναι κρίσιμη για τις DMX512 συσκευές καθώς αναγγέλλει την αρχή μεταδόσεις ενός DMX512 πακέτου. Το Break είναι μια συνεχείς “low” κατάσταση που έχει διάρκεια 88 μsec .

- ***MARK AFTER BREAK TIMING:***

Το mark-after-break μεταδίδεται αμέσως μετά από το break είναι μια συνεχείς “high” κατάσταση που έχει διάρκεια 8μsec.

- ***START CODE:***

Ο start code μεταδίδεται αμέσως μετά από το mark-after –break και έχει διάρκεια 44 μsec. Αποτελείται από ένα start και δυο stop “bits” καθώς επίσης και από αλλά 8 “bits” που βρίσκονται σε κατάσταση “low”. Η δομή του start code είναι ιδία με την δομή των frames που ακολουθούν.

- ***MARK-BETWEEN-FRAME TIME*:**

Η διάρκεια του mark- between- frame είναι ο χρόνος μεταξύ του τέλους ενός frame(τέλος του δευτέρου stop bit) και την αρχή του επομένου (αρχή του start bit), και η πιθανή διάρκεια του μπορεί να είναι 0 μέχρι 1 sec .

- ***MARK TIME BETWEEN-PACKETS***

Mark time between packets είναι η διάρκεια μιας HI καταστάσεις ανάμεσα σε δυο πακέτα DMX512. Αυτή η διάρκεια μπορεί να είναι από 0 έως 1 sec. Βεβαία η DMX512 δεκτές πρέπει να μπορούν να δέχονται και συνεχόμενα πακέτα.

2.3 ΣΥΝΔΕΣΕΙΣ DMX512

Οι DMX512 συσκευές συνδέονται με τον εξοπλισμό φωτισμού μέσω 5pin XLR βύσματος (εικ.2.3.1). Το θηλυκό βύσμα είναι εγκατεστημένο στον πομπό ενώ το αρσενικό βύσμα είναι εγκατεστημένο στον δεκτή. Τα βύσματα για το DMX 512 αποτελούνται από 2 ζευγάρια καλωδίων και μια γείωση, για την μεταφορά της DMX512 πληροφορίας απαιτείται μονό το ένα ζευγάρι καλωδίων και η γείωση. Το δεύτερο ζευγάρι καλωδίων παραμένει για αδιευκρίνιστες προαιρετικές χρήσης.

Εικόνα 2.3.1:DMX connector

Το 5pin XLR βύσμα περιγράφεται στα αυθεντικά στανταρ της USSIT για το DMX 512 και πρέπει να είναι ο προτεινόμενος τύπος συνδέσεις.

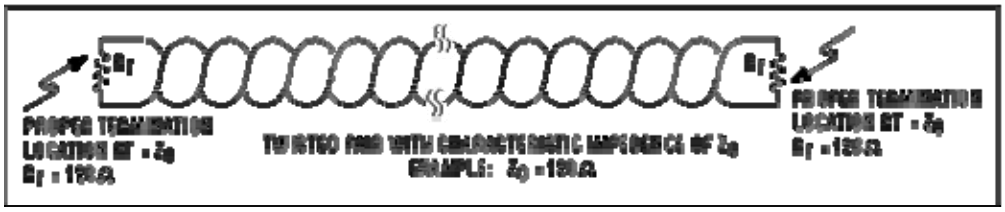
Κάποιοι εξοπλισμοί μπορεί να έχουν εγκατεστημένα 3pin XLR βύσματα όπως αυτά που χρησιμοποιούνται για τις συνδέσεις των μικροφώνων, σε αυτή την περίπτωση τα 3 pins κάνουν την ιδία δουλειά με αυτήν που κάνουν και τα τρία πρώτα pins στα 5pin XLR. Η χρήση 3pin XLR βυσμάτων δεν συνιστάται και δεν είναι μέρος των στανταρ της USSIT για το DMX 512 [2].

Τα audio καλώδια και τα καλώδια μικροφώνων δεν είναι κατάλληλα για την μετάδοση του DMX 512. Τα μονά καλώδια που είναι κατάλληλα για την γρήγορη μεταφορά δεδομένων είναι τα 5 pins XLR.

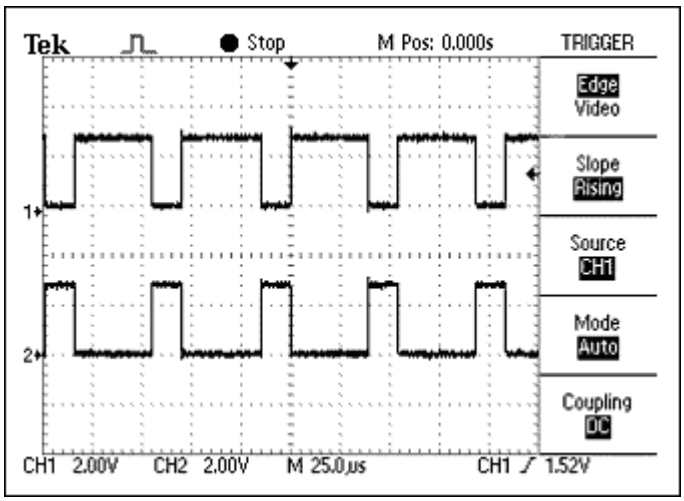
2.4 RS/EIA 485

Το DMX512 είναι σχεδιασμένο να συνδέεσαι με καλώδια μέσω ενός πρωτοκόλλου οπού ονομάζεται EIA 485 . Το EIA 485 είναι το μονό που δίνει μια περιγραφή για τα ηλεκτρικά επίπεδα του πρωτοκόλλου, την τάση, τα ρεύματα κ.τ.λ. Μια DMX512 συσκευή μπορεί να συνδεθεί με μια EIA 485 διάταξη για να γίνει σωστά η μετάδοση των δεδομένων χωρίς κανέναν κίνδυνο.

Το EIA 485 διευκρινίζει ότι η σύνδεση μεταξύ πομπού και δεκτή γίνεται με τρία καλώδια. Το καλώδιο των data,το καλώδιο των ανεστραμμένων data, και την γείωση. Τα δυο σύρματα των data είναι μαζί ένα ζευγάρι κλεισμένα σε ένα μεταλλικό προστατευτικό που είναι διαμορφωμένο στα 0 V. Αυτό βοηθά στο να αποφεύγονται η μαγνητικές επιδράσεις μεταξύ των καλωδίων.



Τα δεδομένα στέλνονται σαν σειρές από “high” και “low” καταστάσεις. Κατάσταση high είναι όταν τα δεδομένα είναι θετικά (ως προς το ρεύμα 0-5 V) και low όταν τα δεδομένα είναι αρνητικά (ως προς το ρεύμα 0- (-5) V). Σε μερικά συστήματα το καλώδιο της γείωσης υπάρχει μονό σαν προστασία και δεν συνδέεσαι με τα κυκλώματα του δεκτή.

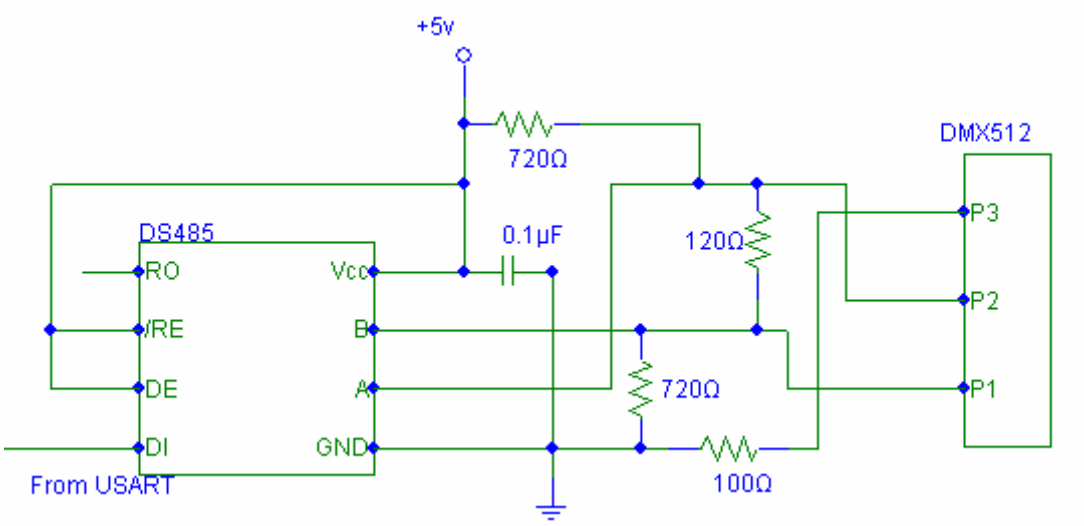


Εικόνα 2.4.1: balance σήμα

Αυτή η μέθοδος πού στέλνονται από το ένα καλώδιο τα δεδομένα και από το δεύτερο τα αντίστροφο η το συμπληρωματικό του, είναι γνωστή σαν “balance” μεταφορά δεδομένων (εικ.2.4.1). Ο δεκτής λαμβάνει τις διαφορές μεταξύ της γραμμής δεδομένων και τις ανεστραμμένης γραμμής δεδομένων για να αποκωδικοποίηση το σήμα. Οποιαδήποτε μαγνητική επίδραση από τα καλώδια θα έχει επιπτώσεις και στις δυο γραμμές δεδομένων αλλά θα αγνοηθούν από τον δεκτή.

Τα καλώδια είναι ζευγάρι για να εντοπιστεί οποιοδήποτε ξένο σήμα οπού υπάρχει και στις δυο γραμμές δεδομένων. Αυτό το ζευγάρωμα είναι πολύ σημαντικό για την μείωση των μαγνητικών επιδράσεων και πιο αποτελεσματικό από την μαγνητική θωράκιση των καλωδίων. Για αυτόν τον λόγω δεν συνιστάται να χρησιμοποιούνται κοινά καλώδια του εμπορίου γιατί δεν έχουν τους αγωγούς τούς ζευγαρωμένους.

Στην παρακάτω εικόνα (εικ.2.4.2) παρουσιάζεται η συνδεσμολογία του EIA-485.



Εικόνα 2.4.2: κύκλωμα EIA 485

3.ΜΙΚΡΟΕΛΕΓΚΤΗΣ ATmega163

3.1 ΓΕΝΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ATmega163

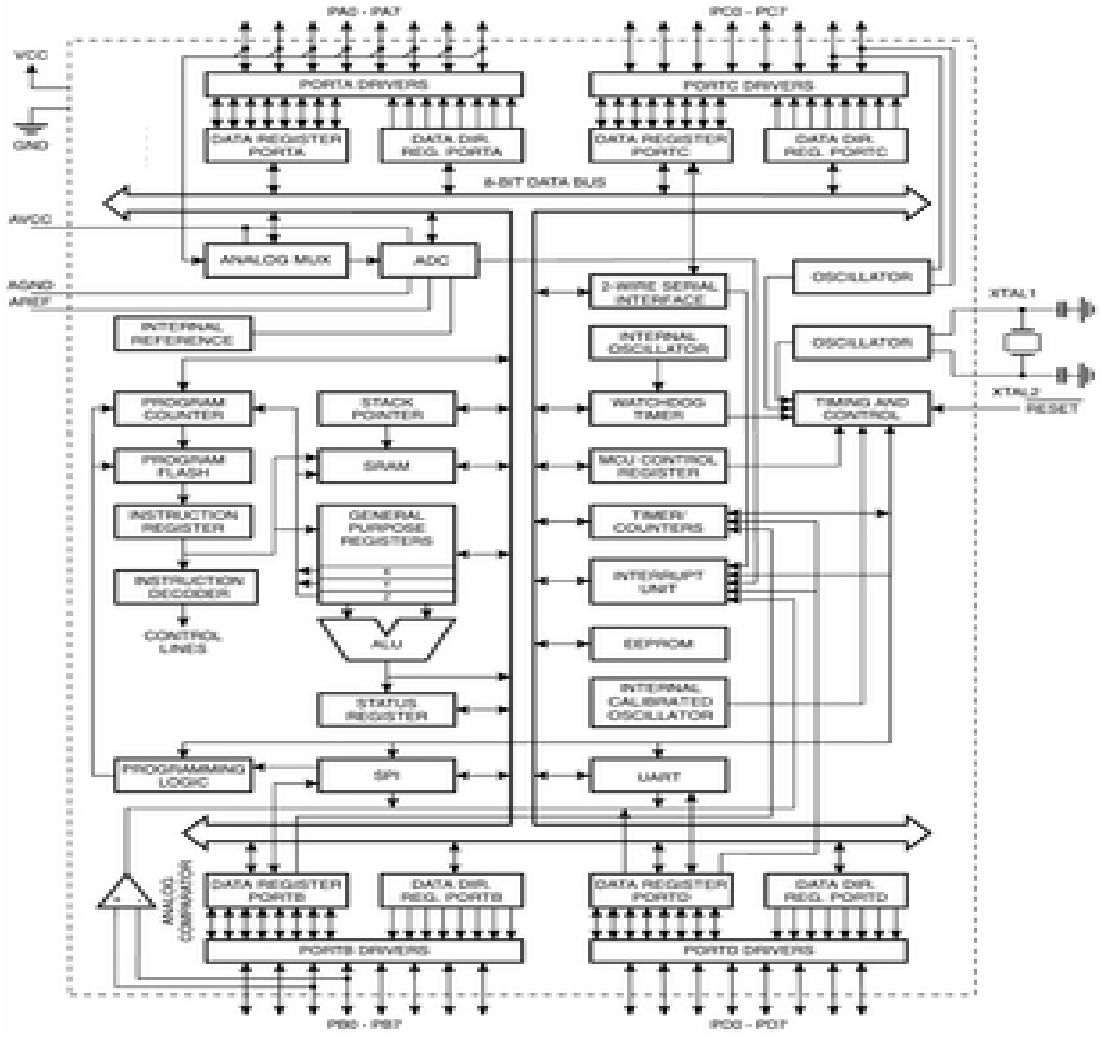
Ο ATmega163 είναι ένας χαμηλής ισχύος CMOS μικροελεγκτης 8 bit βασισμένος στην αρχιτεκτονική AVR RISC. Εκτελώντας ισχυρές εντολές, σε ένα κύκλο ρολογιού ο ATmega163 πετυχαίνει ρυθμό απόδοσης που φτάνει μέχρι 1 MIPS/MHz που επιτρέπει στον σχεδιαστή να βελτιστοποιεί την κατανάλωση ισχύος, σε βάρος τής ταχύτητας επεξεργασίας.

Ο πυρήνας του AVR συνδυάζει ένα σύνολο ποικίλων εντολών με 32 ενεργούς καταχώρησες γενικής χρήσης. Και οι 32 καταχώρησες είναι συνδεδεμένοι με την ALU(Arithmetic Logic Unit) επιτρέποντας την προσέγγιση 2 ανεξαρτήτων καταχωρητων με μια εντολή που εκτελείται σε ένα κύκλο ρολογιού. Η αρχιτεκτονική που προκύπτει συμβάλλει σε έναν πιο αποδοτικό κώδικα ενώ πετυχαίνει ρυθμούς απόδοσης μέχρι και 10 φορές μεγαλύτερους από τους συμβατικούς CISC μικροελεγκτες.

Ο ATmega163 συγκεντρώνει τα παρακάτω χαρακτηριστικά: 8Kb εσωτερικής προγραμματίσιμης ταχείας μνήμης (Flash Memory), 512 bytes EEPROM, 1024 bytes SRAM, 32 γραμμές εισόδου / εξόδου γενικής χρήσης, 32 ενεργούς καταχώρησες γενικής χρήσης, 3 ευκάμπτους αχρόνιστες /μετρητές με λειτουργίες συγκρίσεις, εσωτερικούς και εξωτερικούς διακόπτες (interrupt), 10 bit A/D μετατροπείς, μια προγραμματισημη σειριακή UART, ένα προγραμματισημο ρολόι τύπου “watchdog timer” με ενσωματωμένο ταλαντωτή, μια σειριακή θύρα SPI και 4 λειτουργίες εξοικονόμησης ισχύος με επιλογή λογισμικού. Η λειτουργία idle mode σταματά τον επεξεργαστή, ενώ επιτρέπει στα συστήματα των SRAM, αχρόνιστων/μετρητών, SPI θύρας διακοπτών (interrupt) να συνεχίσουν την λειτουργία τούς λειτουργία. Η Power down mode αποθηκεύει τα περιεχόμενα των καταχωρητων, αλλά παγώνει τον ταλαντωτή, θέτοντας εκτός λειτουργία όλες τις υπόλοιπες λειτουργίες των chip μέχρι την επόμενη εξωτερική διακοπή (interrupt) η αναστοιχειοθέτηση(reset). Στην Power save mode λειτουργία ο ασύγχρονος χρονιζομενος ταλαντωτής συνεχίζει να λειτουργεί και επιτρέπει στον χρηστή να διατηρεί ένα βασικό χρονισμό όταν η υπόλοιπη συσκευή είναι ανενεργοί. Η ADC Noise reduction λειτουργία σταματά τον επεξεργαστή και όλες I/O λειτουργίες

εκτός από τον ασύγχρονο χρονιστεί και τους A/D μετατροπείς, για να ελλατωσει των θόρυβο κατά διάρκεια της A/D μετατροπής.

Ο AT mega163 υποστηρίζεται από μια πλήρη ακολουθία εργαλείων ανάπτυξης προγράμματος και συστήματος όπως C Compilers, macro – assemblers, διορθωτές-εξομοίωτες προγραμμάτων, ενδοκυκλωματικους εξομοίωτες , και εξοπλισμό αναβάθμισης [3].

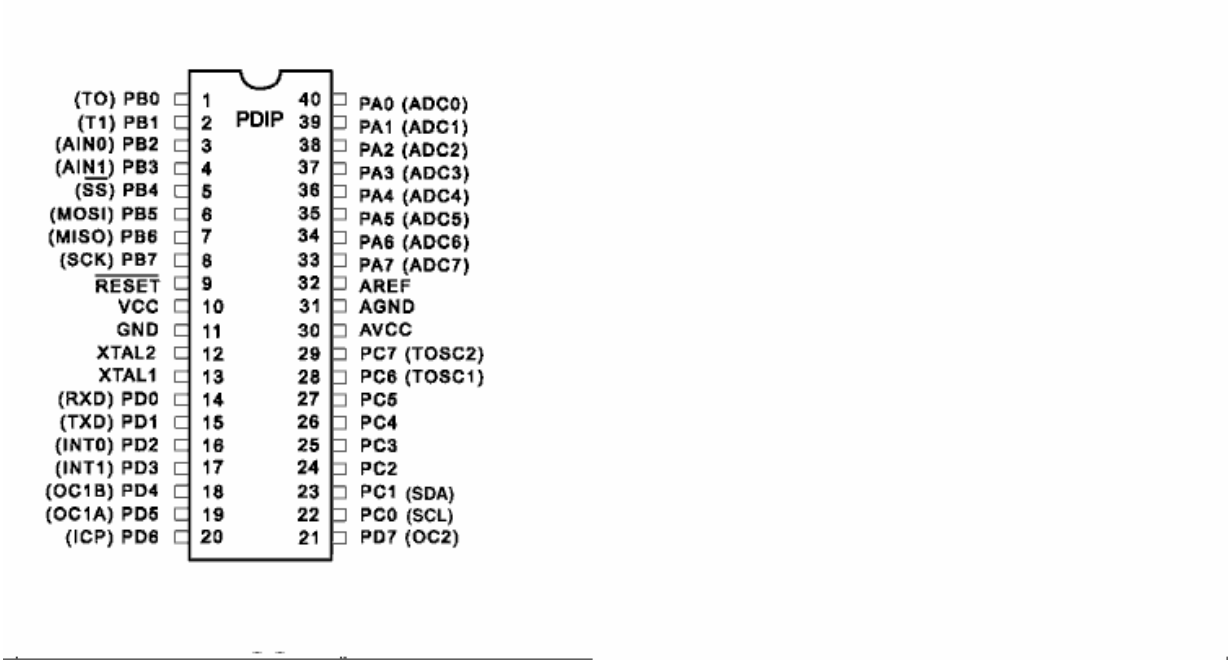


Εικόνα 3.1.1 :block διάγραμμα ATmega163

3.2 ΕΠΙΛΟΓΗ ATmega163

Ο λόγος επιλογής του AT mega163 είναι ότι ο συγκεκριμένο μικροελεγκτης έχει ενσωματωμένους 8 A/D μετατροπείς. Αυτό είναι πολύ σημαντικό για την εφαρμογή καθώς μέσω των A/D μετατροπεών γίνεται ο έλεγχος των συσκευών φωτισμού. Τα αποτελέσματα που δίνουν οι A/D μετατροπείς φορτώνονται καθε φορά σε ένα data frame.

3.3 ΠΕΡΙΓΡΑΦΗ ΑΚΡΟΔΕΚΤΩΝ



Εικόνα 3.3.1 : ATmega163

PORTA (PA7….PA0): Χρησιμοποιούνται σαν αναλογικές είσοδοι για του 8 A/D μετατροπείς.

Η port A είναι μια αμφίδρομη 8bit θύρα. Οι ακροδέκτες των θυρών μπορούν να παρέχουν εσωτερικούς “pull up “ αντιστάτες (επιλεγμένοι για κάθε bit). Οι buffers εξόδου της PortA, μπορούν να δώσουν 20mA και μπορούν να οδηγήσουν τα LEDS απευθείας. Όταν οι ακροδέκτες PA0-PA7 χρησιμοποιούνται σαν είσοδοι και τροφοδοτούνται εξωτερικά θα ρευματοδοτηθουν εφόσον οι εσωτερικοί pull up αντιστάτες, είναι ενεργοποιημένοι. Τα pins του port A βρίσκονται σε κατάσταση

άπειρης αντίστασης όταν μια κατάσταση reset ενεργοποιείται, ακόμα και αν ο χρονιστείς είναι ανενεργός.

PORTB(PB7…PB0):Η port B είναι μια αμφίδρομη 8bit θύρα. Οι buffers εξόδου της PortB, μπορούν να δώσουν 20mA. Όταν οι ακροδέκτες Port B χρησιμοποιούνται σαν είσοδοι και τροφοδοτούνται εξωτερικά θα ρευματοδοτηθουν εφόσον οι εσωτερικοί pull up αντιστάτες είναι ενεργοποιημένοι. Τα pins του port B βρίσκονται σε κατάσταση άπειρης αντίστασης όταν μια κατάσταση reset ενεργοποιείται, ακόμα και αν ο χρονιστείς είναι ανενεργός.

PORTC (PC7…PC0) :Η port C είναι μια αμφίδρομη 8bit θύρα. Οι buffers εξόδου της PortC, μπορούν να δώσουν 20mA. Όταν οι ακροδέκτες Port C χρησιμοποιούνται σαν είσοδοι και τροφοδοτούνται εξωτερικά θα ρευματοδοτηθουν εφόσον οι εσωτερικοί pull up αντιστάτες είναι ενεργοποιημένοι. Τα pins του port C βρίσκονται σε κατάσταση άπειρης αντίστασης, όταν μια κατάσταση reset ενεργοποιείται, ακόμα και αν ο χρονιστείς είναι ανενεργός.

PORTD (PD7…PD0): Η port D είναι μια αμφίδρομη 8bit θύρα. Οι buffers εξόδου της Port D, μπορούν να δώσουν 20mA. Όταν οι ακροδέκτες Port D χρησιμοποιούνται σαν είσοδοι και τροφοδοτούνται εξωτερικά θα ρευματοδοτηθουν εφόσον οι εσωτερικοί pull up αντιστάτες, είναι ενεργοποιημένοι. Τα pins του port D βρίσκονται σε κατάσταση άπειρης αντίστασης, όταν μια κατάσταση reset ενεργοποιείται, ακόμα και αν ο χρονιστείς είναι ανενεργός.

VCC: Τάση ανεφοδιασμού

GND: Γείωση

RESET: Είσοδος μηδενισμού. Μια χαμηλή στάθμη σε αυτόν τον ακροδέκτη, για περισσότερα από 500ns θα αναπαράγει έναν μηδενισμό (reset),ακόμα και αν το ρολόι δεν τρέχει. Μικρότεροι παλμοί δεν εγγυάται ότι θα προκαλέσουν reset.

XTAL1:Είσοδος στον ενισχυτή ανάστροφου ταλαντωτή και είσοδος στο κύκλωμα του internal timer (εσωτερικός χρονιστείς).

XTAL2:Έξοδος από τον ενισχυτή ανάστροφου ταλαντωτή.

AVCC: Τάση ανεφοδιασμού για την Port A και A/D μετατροπεις, πρεπει να συνδεθούν εξωτερικά με το Vic ακόμα και αν οι ADC δεν χρησιμοποιούνται.

AREF: Είναι η αναλογική τάση αναφοράς εισόδου για τον ADC

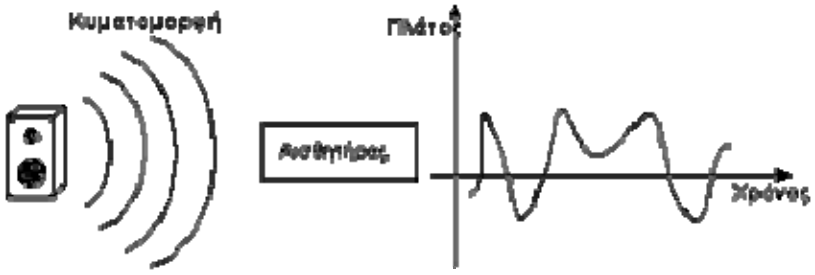
AGND: Αναλογικη γείωση

4. A/D ΜΕΤΑΤΡΟΠΗ

4.1 ΠΛΗΡΟΦΟΡΙΑ ΩΣ ΣΗΜΑ

Η πληροφορία που αντιλαμβανόμαστε μέσω των αισθήσεων μας και επεξεργάζεται ο εγκέφαλος μας, μπορεί να περιγράφει ως μια ή περισσότερες *φυσικές μεταβλητές* η τιμή των οποίων είναι μια συνάρτηση του χρόνου και / ή του χώρου. Να σημειωθεί ότι ως πληροφορία εννοούμε την μορφή της διέγερσης που λαμβάνουμε και όχι το σημασιολογικό περιεχόμενο που αυτή μεταφέρει. Για παράδειγμα, όταν αναφερόμαστε σε ηχητική πληροφορία, η φυσική μεταβλητή περιγράφει την πίεση του αέρα στη θέση ενός παρατηρητή ως συνάρτηση του χρόνου. Αυτή η ηχητική πληροφορία έχει συνήθως και κάποια ερμηνεία, σημασιολογικό περιεχόμενο. Αν ακούμε μια ομιλία, οι λέξεις και οι ιδέες είναι το σημασιολογικό περιεχόμενο του ήχου. Το πως μπορούμε να παρασιτήσουμε τη σημασιολογική πληροφορία δεν θα μας απασχολήσει εδώ.

Αυτή η φυσική μεταβλητή που περιγράφει ένα φαινόμενο, μπορεί να μετρηθεί με κάποιο ειδικά κατασκευασμένο όργανο που ονομάζεται *αισθητήρας*. Ένας αισθητήρας μετατρέπει αυτή την φυσική ποσότητα, στην περίπτωση του ήχου την πίεση του αέρα, σε μια άλλη ποσότητα, όπως μια ηλεκτρική τιμή, που ονομάζεται *σήμα*. Αυτό το σήμα είναι τέτοιο ώστε να άριστα το φυσικό μέγεθος με πιστότητα και μπορεί εύκολα να μετρηθεί. Τα σήματα διακρίνονται σε δύο βασικές κατηγορίες:



Εικ 4.2.1. Η πληροφορία ως σήμα

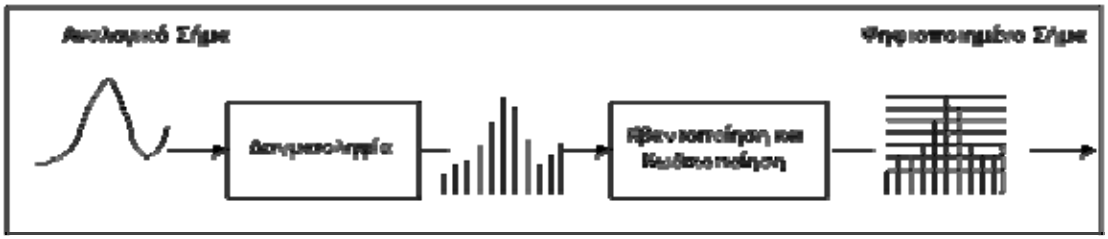
Αναλογικό ονομάζεται ένα σήμα το οποίο είναι συνεχής συνάρτηση του χρόνου και / ή του χώρου. Τότε λέμε επίσης ότι το σήμα είναι ανάλογο της φυσική μεταβλητής που περιγράφει.

Ψηφιακό ονομάζεται ένα σήμα το οποίο αποτελείται από μια ακολουθία διακριτών τιμών που είναι κωδικοποιημένες στο δυαδικό σύστημα και εξαρτώνται από το χρόνο ή το χώρο.

4.2 ΔΕΙΓΜΑΤΟΛΗΨΙΑ, ΚΒΑΝΤΟΠΟΙΗΣΗ ΚΑΙ ΚΩΔΙΚΟΠΟΙΗΣΗ

Το αποτέλεσμα της *ψηφιοποίησης* (ή αλλιώς της Αναλογική / Ψηφιακή μετατροπής ή πιο απλά A/D) είναι ένα σύνολο λέξεων υπολογιστή που περιγράφουν το αναλογικό σήμα που παρέχει ο αισθητήρας. Η ψηφιοποίηση ενός αναλογικού σήματος γίνεται σε τρία βήματα. Πρώτα, γίνεται *δειγματοληψία* του σήματος. Αυτό σημαίνει ότι από το άπειρο πλήθος τιμών του συνεχούς σήματος, κρατάμε μόνο ένα σύνολο διακριτών τιμών, που συνήθως διαφέρουν κατά κάποιο σταθερό χρονικό διάστημα.

Οι τιμές ενός αναλογικού σήματος μπορούν να πάρουν οποιαδήποτε τιμή μέσα από το πεδίο τιμών του. Αφού το πεδίο αυτό είναι γενικά συνεχές, οι τιμές αυτές είναι άπειρες. Μια λέξη μήκους v bits μπορεί να περιγράψει 2^v στάθμες μέσα από το πεδίο τιμών του σήματος. Δηλαδή, δεν γίνεται να περιγράφουν όλες οι δυνατές τιμές του σήματος, αλλά μόνο κάποιο πεπερασμένο υποσύνολο αυτών. Οι τιμές που θα περιγράφουν, επιλέγονται ανάλογα με την ακρίβεια και το μήκος του διαστήματος που θέλουμε να καλύψουμε. Είναι φανερό ότι αυτές οι δύο απαιτήσεις είναι αντικρουόμενες και ότι πρέπει να γίνει απαραίτητα κάποιος συμβιβασμός. Αφού επιλεχθούν οι στάθμες, αντιστοιχίζεται σε κάθε μια από αυτές μια λέξη, γίνεται δηλαδή η *κωδικοποίηση*. Το επόμενο βήμα είναι η *κβαντοποίηση*. Στην κβαντοποίηση, βρίσκουμε την πλησιέστερη στάθμη κάθε τιμής που προέκυψε από τη δειγματοληψία.



Εικ 4.3.1.. Ψηφιοποίηση ενός αναλογικού σήματος

Η ψηφιοποίηση έχει πλέον ολοκληρωθεί αφού κάθε τιμή μπορεί να παρασταθεί με την λέξη που έχουμε αντιστοιχήσει στην πλησιέστερα στάθμη αυτής.

4.3 ΑΝΑΛΟΓΙΚΗ/ΨΗΦΙΑΚΗ ΜΕΤΑΤΡΟΠΗ

Η ψηφιακή αναπαράσταση της πληροφορίας είναι απόλυτα κατανοητή από τον υπολογιστή αλλά δεν είναι καθόλου χρήσιμη στον άνθρωπο. Αυτό σημαίνει ότι για να γίνει η παρουσίαση της από ένα σύστημα πολυμέσων πρέπει πρώτα να μετατραπεί σε αναλογική. Η διαδικασία αυτή είναι η αντίστροφη της A/D και συμβολίζεται ως D/A. Κάθε τύπος πληροφορίας έχει διαφορετικές ανάγκες A/D και D/A μετατροπής:

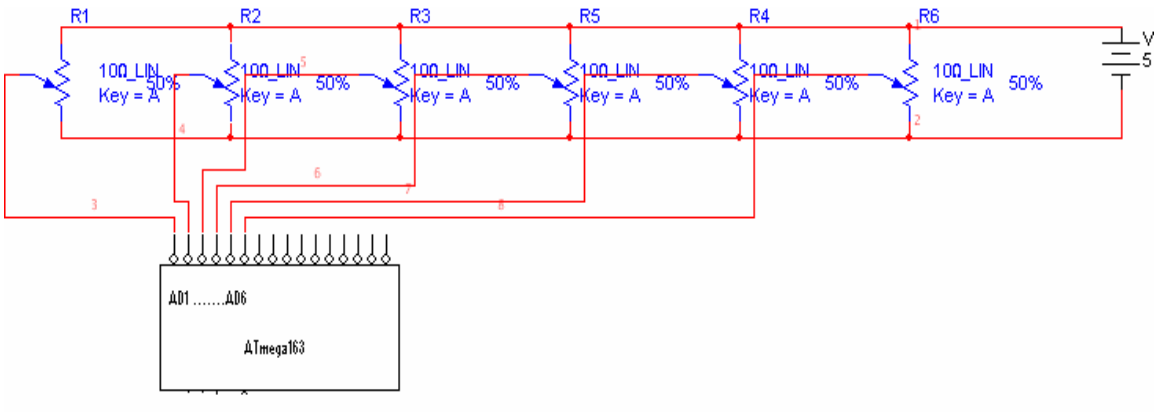
Το κείμενο, τα γραφικά γενικά όλα τα μέσα που έχουν συντεθεί σε υπολογιστή, δεν χρειάζονται A/D μετατροπή αφού δημιουργούνται εξ΄ αρχής σε δυαδική μορφή. Για να τα δούμε όμως στην οθόνη, πρέπει να γίνει κατάλληλη D/A μετατροπή. Αντίθετα ο ηχογραφημένος ήχος, το χειρόγραφο κείμενο και γενικά όλα τα captured media απαιτούν A/D και D/A.

5.ΓΕΝΙΚΗ ΠΕΡΙΓΡΑΦΗ ΕΦΑΡΜΟΓΗΣ (HARDWARE ΜΕΡΟΣ)

5.1 ΔΙΑΤΑΞΗ ΕΦΑΡΜΟΓΗΣ

- ΕΛΕΓΧΟΣ ΕΦΑΡΜΟΓΗΣ ΜΕΣΩ ΜΕΤΑΒΛΗΤΩΝ ΑΝΤΙΣΤΑΣΕΩΝ
- ΜΙΚΡΟΕΛΕΓΚΤΗΣ ATmega 163
- EIA 485
- DMX ΕΞΟΔΟΣ

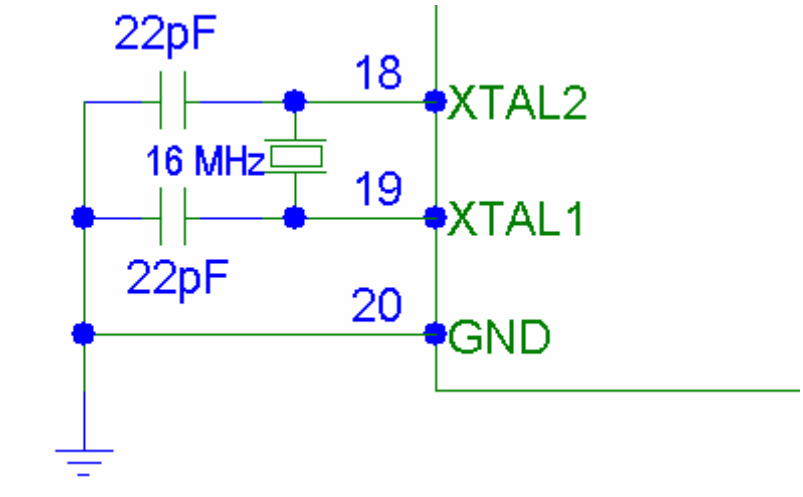
- ΕΛΕΓΧΟΣ ΕΦΑΡΜΟΓΗΣ ΜΕΣΩ ΜΕΤΑΒΛΗΤΩΝ ΑΝΤΙΣΤΑΣΕΩΝ: Για τον έλεγχο της εφαρμογής έχουν χρησιμοποιηθεί 6 μεταβλητή αντιστάτες που τροφοδοτούνται με τάση από 0-5 Volt. Τα αποτελέσματα αυτών των αντιστατών γίνονται είσοδοι στους πρώτους 6 A/D μετατροπείς του μικροελεγκτη ATmega163 (Εικ.5.2.1) .



Εικ.5.1.1 Συνδεσμολογία A/D μετατροπεων

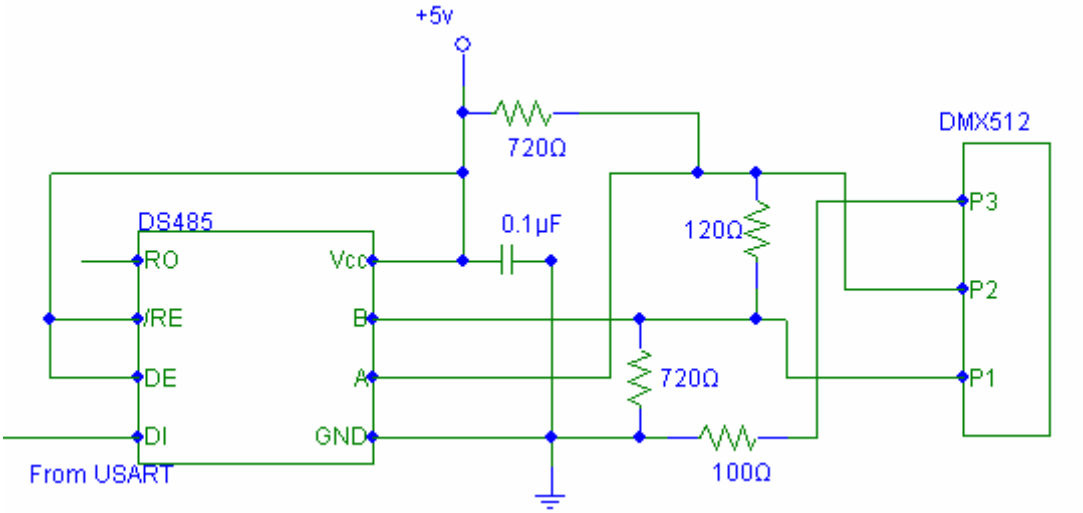
- ΜΙΚΡΟΕΛΕΓΚΤΗΣ :Στους 8 A/D μετατροπείς συνδέονται τα 8 ποτενσιόμετρα που τους τροφοδοτούν με αναλογικό σήμα έτσι ώστε να γίνει η μετατροπή από αναλογικό σε ψηφιακό σήμα. Επίσης συνδέεσαι και με την τροφοδοσία, καθώς επίσης συνδέουμε πάνω στον ATmega163 και έναν κρύσταλλο 16 MHz έτσι ώστε να ρυθμίσουμε το σωστό baud rate

(Εικ.5.2.2). Την έξοδο του μικροελεγκτη την παίρνουμε από το PD1 pin που ονομάζεται Tx.



Εικ.5.1.2 Συνδεσμολογία κρυσταλλου

- EIA 485 : Αποτελείται από το ολοκληρωμένο DS-485 έναν πυκνωτή 0.1μf, δυο αντίστασης 720 Ω μια 120 Ω και μια 100 Ω. Εκτός από την τροφοδοσία του στο pin DI του DS 485 συνδέουμε την Tx του AT mega 163.Απο τα pin A, B έχουμε έξοδο από το pin A το DMX 512 πακέτο και από το pin B το αντιστραμμένο DMX 512 πακέτο, καθώς και μια γείωση (Εικ.5.2.3) [8].



Εικ.5.1.3 Συνδεσμολογία EIA485

- DMX ΕΞΟΔΟΣ : Η DMX έξοδο είναι ένα 3pin XLR που το ένα pin συνδέεται με το A pin του DS-485 και το άλλο pin με το B pin του DS 485. Το τρίτο pin είναι η γείωση

5.2 ΑΝΑΛΥΣΗ ΕΦΑΡΜΟΓΗΣ

Σε αυτό το κεφαλαίο θα εξηγήσουμε αναλυτικά την λειτουργία της σύσκευης. Η συσκευή τροφοδοτείτε με ρεύμα από τους μεταβλητούς αντιστάτες (ποτενσιόμετρα). Αφού υπάρχουν αυτοί η μεταβλητή αντιστάτες υπάρχει η δυνατότητα να μεταβάλλεται η τάση εισόδου. Οι μεταβλητή αντιστάτες είναι συνδεμένη με τους A/D μετατροπείς του μικροελεγκτη AT mega 163 .Μέσω του προγραμματισμού του ATmega 163 σε γλωσσά assembly έχουμε ενεργοποίηση τους μετατροπείς καθώς επίσης έχει προγραμματιστεί να παράγει DMX 512 πακέτα. Δηλαδή έχουμε πετύχει τους χρόνους του “break”,”mark – after – break”, του “start code” καθώς και τους χρόνους των “data frames” με μηδενική πληροφορία. Στα “data frames” φορτώνονται τα δεδομένα που προκύπτουν από την A/D μετατροπή της τάσης είσοδοι. Οι ψηφιακές τιμές που προκύπτουν από κάθε μεταβολή των μεταβλητών αντιστατών φορτώνονται σε ένα κάθε φορά “data frame”. Δηλαδή η τιμές του πρώτου ποτενσιόμετρου στο πρώτο “data frame” , του δευτέρου ποτενσιόμετρου στο δεύτερο “data frame” κτλ. Το DMX 512 έχει την δυνατότητα παραγωγής μέχρι 512 κανάλια (“data frame“), η συγκεκριμένη εφαρμογή παράγει μέχρι και 6 κανάλια.

Επίσης ο AT mega 163 συνδέεται με ένα κρύσταλλο των 16MHz έτσι ώστε να ρυθμίσουμε το σωστό baud rate που για το DMX 512 πρωτόκολλο είναι 250000 bps. Το baud rate αυτό υπολογίζεται από τον τύπο:

$$BAUD = \frac{Fosc}{16(UBRR+1)}$$

BAUD: baud rate

Fosc: Συχνότητα κρυστάλλου ρολογιού

UBRR: περιεχομενα του UART baud rate register

Baud Rate	1 MHz	%Error	1,84 MHz	%Error	2 MHz	%Error	2,458 MHz	%Error
2400	UBR= 25	0,2	UBR= 47	0,0	UBR= 51	0,2	UBR= 63	0,0
4800	UBR= 12	0,2	UBR= 23	0,0	UBR= 25	0,2	UBR= 31	0,0
9600	UBR= 6	7,5	UBR= 11	0,0	UBR= 12	0,2	UBR= 15	0,0
14400	UBR= 3	7,8	UBR= 7	0,0	UBR= 8	3,7	UBR= 10	3,1
19200	UBR= 2	7,8	UBR= 5	0,0	UBR= 6	7,5	UBR= 7	0,0
28800	UBR= 1	7,8	UBR= 3	0,0	UBR= 3	7,8	UBR= 4	6,3
38400	UBR= 1	22,9	UBR= 2	0,0	UBR= 2	7,8	UBR= 3	0,0
57600	UBR= 0	7,8	UBR= 1	0,0	UBR= 1	7,8	UBR= 2	12,5
76800	UBR= 0	22,9	UBR= 1	33,3	UBR= 1	22,9	UBR= 1	0,0
115200	UBR= 0	84,3	UBR= 0	0,0	UBR= 0	7,8	UBR= 0	25,0

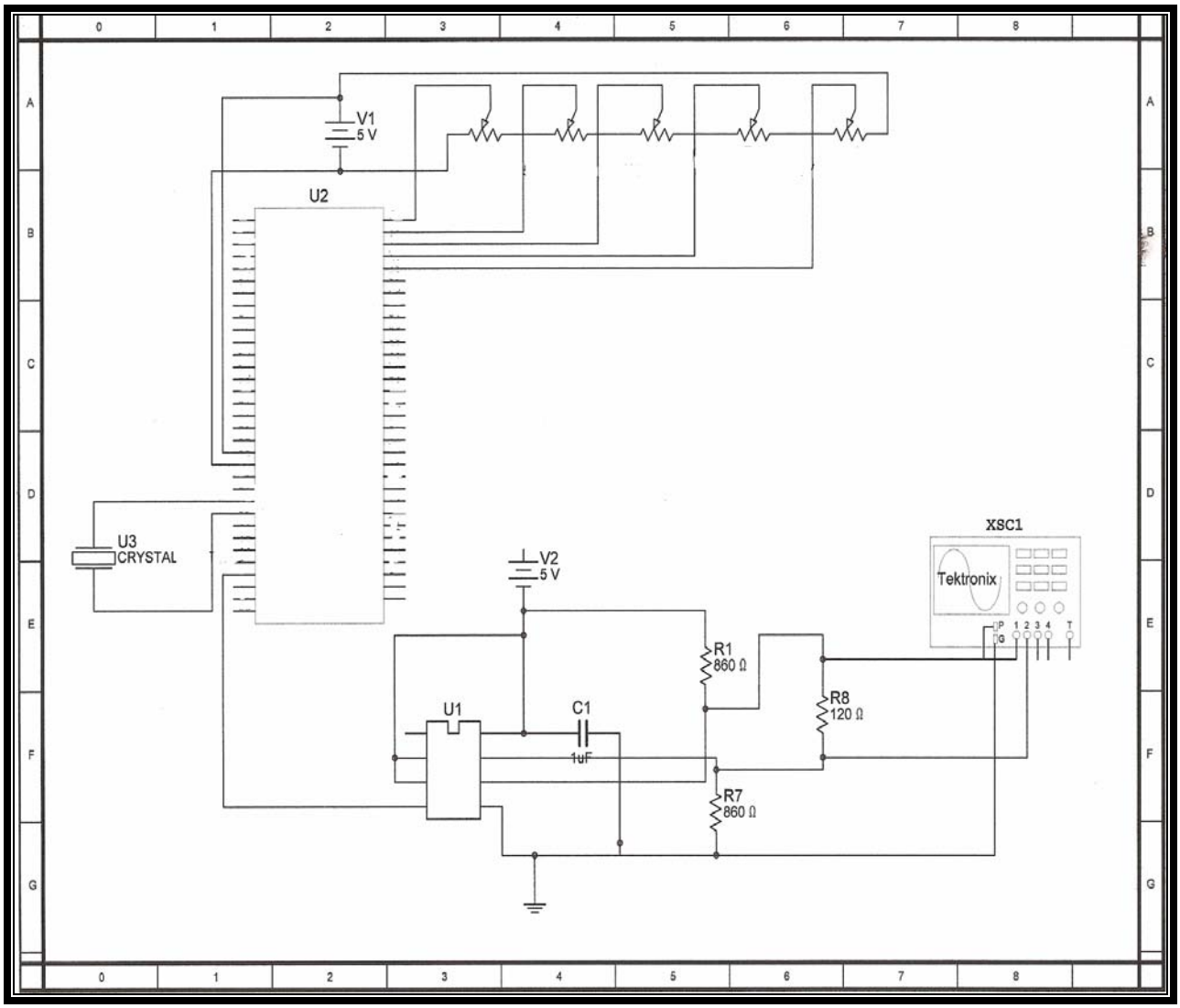
Baud Rate	3,28 MHz	%Error	3,69 MHz	%Error	4 MHz	%Error	4,608 MHz	%Error
2400	UBR= 84	0,4	UBR= 95	0,0	UBR= 103	0,2	UBR= 119	0,0
4800	UBR= 42	0,8	UBR= 47	0,0	UBR= 51	0,2	UBR= 59	0,0
9600	UBR= 20	1,6	UBR= 23	0,0	UBR= 25	0,2	UBR= 29	0,0
14400	UBR= 13	1,6	UBR= 15	0,0	UBR= 16	2,1	UBR= 19	0,0
19200	UBR= 10	3,1	UBR= 11	0,0	UBR= 12	0,2	UBR= 14	0,0
28800	UBR= 6	1,6	UBR= 7	0,0	UBR= 8	3,7	UBR= 9	0,0
38400	UBR= 4	6,3	UBR= 5	0,0	UBR= 6	7,5	UBR= 7	6,7
57600	UBR= 3	12,5	UBR= 3	0,0	UBR= 3	7,8	UBR= 4	0,0
76800	UBR= 2	12,5	UBR= 2	0,0	UBR= 2	7,8	UBR= 3	6,7
115200	UBR= 1	12,5	UBR= 1	0,0	UBR= 1	7,8	UBR= 2	20,0

Baud Rate	7,37 MHz	%Error	8 MHz	%Error
2400	UBR= 191	0,0	UBR= 207	0,2
4800	UBR= 95	0,0	UBR= 103	0,2
9600	UBR= 47	0,0	UBR= 51	0,2
14400	UBR= 31	0,0	UBR= 34	0,8
19200	UBR= 23	0,0	UBR= 25	0,2
28800	UBR= 15	0,0	UBR= 16	2,1
38400	UBR= 11	0,0	UBR= 12	0,2
57600	UBR= 7	0,0	UBR= 8	3,7
76800	UBR= 5	0,0	UBR= 6	7,5
115200	UBR= 3	0,0	UBR= 3	7,8

Πίνακας 5.2.1:Ρυθμισεις UBR σε διαφορές συχνότητες κρυστάλλου

Για κανονικές συχνότητες κρυστάλλου, τα πιο συνηθισμένα baud rate μπορούν να αναπαραχθούν χρησιμοποιώντας τις ρυθμίσεις του UBRR από των παραπάνω πίνακα.

Οι έξοδος των δεδομένων από τον μικροελεγκτη AT mega 163 γίνεται από το pin 15 (PD1) που ονομάζεται TXD και έχει μορφή unbalanced σήματος. Για αυτόν τον λόγω πριν γίνει έξοδος περνάει από το EIA-485 κύκλωμα που το μετατρέπει σε balance σήμα. Το EIA-485 χρησιμοποιεί το DS-485 chip. Μετά από όλη αυτήν την διαδικασία έχουμε την έξοδο του DMX -512 πακέτου από ένα 3 pin XLR.



Εικόνα 5.2.1 : Κύκλωμα εφαρμογής

5.3 ΚΑΤΑΧΩΡΗΤΕΣ ΕΦΑΡΜΟΓΗΣ

Το πρόγραμμα που δημιουργήθηκε για τον μικροελεγκτη ATmega163 έγινε σε γλωσσά προγραμματισμού assembly. Ο προγραμματισμός του μικροελεγκτη έγινε μέσω του αναπτυξιακού STK500. Όλος ο κώδικας δημιουργήθηκε μέσω του προγράμματος “AVR STUDIO”.

Οι βασικές λειτουργίες του συγκεκριμένου προγράμματος είναι οι παρακάτω :

- Η ενεργοποίηση των 6 A/D μετατροπιών που είναι ενσωματωμένη στον ATmega163.
- Οι δημιουργία του σωστού baud rate.

- Και η δημιουργία και μετάδοση DMX512 σήματος.

5.3.1 ΕΝΕΡΓΟΠΟΙΗΣΗ A/D ΜΕΤΑΤΡΟΠΕΩΝ

Στον ATmega163 η ενεργοποίηση των A/D μετατροπεών γίνεται μέσω του καταχώρηση ADCSR.

Bit	7	6	5	4	3	2	1	0
$06 ($26)	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Το κάθε “bit” του καταχώρηση “ADCSR” ανάλογα με την κατάσταση την οποία(0 η 1) βρίσκεται εκτελεί και διαφορετική λειτουργία.
Παρακάτω εξηγούνται το που επηρεάζει το κάθε “bit”

- Bit 7 – ADEN

Οταν αυτό το bit είναι λογικό ένα “1”, ενεργοποιούνται οι A/D μετατροπείς του AT mega 163. Στην αντίθετη περίπτωση δηλαδή όταν αυτό το bit είναι λογικό μηδέν “0” τότε οι μετατροπείς είναι απενεργοποιημένοι.

- Bit 6 – ADSC

Με αυτό το bit ανάλογα ποτέ θα το κάνουμε “1” έχουμε και διαφορετική λειτουργία .Στην πρώτη λειτουργία που ονομάζεται “single conversion mode” αυτό το bit πρέπει να γίνεται λογικό “1” κάθε φορά που έχουμε A/D μετατροπή. Στην δεύτερη λειτουργία που ονομάζεται “free running mode” αυτό το bit πρέπει να γίνει λογικό “1” όταν αρχίζει η πρώτη A/D μετατροπή. Κάνοντας το bit “0” δεν υπάρχει καμιά επίδραση.

- Bit 5 – ADFR

Όταν θέσουμε αυτό το bit λογικό “1” τότε η μετατροπείς δουλεύουν σε “free running mode”. Όταν αυτό το bit το κάνουμε λογικό “0” σταματάει η free running λειτουργία.

- Bit 4 – ADIF

Αυτό το bit θέτεται λογικό “1” όταν ολοκληρώνεται η A/D μετατροπή και η data registers ανανεώνονται

- Bit 3 – ADIE

Οταν αυτό το bit θέτεται λογικό “1” και I – bit του status register είναι λογικό “1” η ADC conversion complete interrupt ειναι ενεργη.

- Bit 2 – ADPS2...0

Αυτό το bit καθορίζει τον χρονισμό ανάμεσα στον κρύσταλλο και το εσωτερικό ρολόι του ADC.

Ο μικροελεκτης ATmega163 έχει 8 A/D μετατροπείς, για την επιλογή ποιος από τους 8 A/D θα χρησιμοποιηθεί υπάρχει ο καταχώρητης “ADMUX” .

Bit	7	6	5	4	3	2	1	0	
$07 ($27)	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Το κάθε “bit” του ADMUX ανάλογα με την κατάσταση την οποία(0 η 1) βρίσκεται εκτελεί και διαφορετική λειτουργία.

Παρακάτω εξηγούνται το που επηρεάζει το κάθε “bit”

- Bit 7,6 –REFS1…0

Αυτα τα bits διαλέγουν την τάση αναφοράς για τους A/D μετατροπείς. Εάν αυτά τα bits αλλάξουν κατά την διάρκεια της μετατροπής, δεν υπάρχει επίδραση της αλλαγής αυτής μέχρι να τελειώσει το conversion. Ο χρηστής πρέπει να αμελήσει τα πρώτα αποτελέσματα τής μετατροπής, εάν έχουν αλλάξει αυτά τα bits, έτσι ώστε να έχει την μέγιστη ακρίβεια.

- Bit 5 – ADLAR

Το ADLAR bit έχει επίδραση στα αποτελέσματα της μετατροπής στον καταχώρηση ADC Data. Εάν το ADLAR bit είναι λογικό “0”, τα αποτελέσματα έχουν δεξιά διευθέτηση. Εάν το ADLAR bit είναι λογικό “1” ,τα αποτελέσματα έχουν αριστερή διευθέτηση. Εάν αλλάξει το ADLAR bit θα υπάρχει άμεση επίδραση άσχετα αν υπάρχει τρέχων μετατροπή.

- 3.Bits 4…0 MUX4….MUX0

Αυτά τα bits επιλεγούν ποια αναλογική είσοδος είναι συνδεμένη στον ADC. Εάν αλλάξουν αυτά τα bits κατά την διάρκεια μιας μετατροπής δεν θα υπάρχει επίδραση μέχρι να ολοκληρωθεί η συγκεκριμένη μετατροπή.

MUX4..0	Single-ended Input
00000	ADC0
00001	ADC1
00010	ADC2
00011	ADC3
00100	ADC4
00101	ADC5
00110	ADC6
00111	ADC7

Πίνακας 5.3.1:πινακας επιλογής A/D μετατροπεων

5.3.2 BAUD RATE

Για τον ορισμό του σωστού baud rate ο μικροελεγκτης ATmega163 παρέχει τον καταχώρηση UBRR (USART Baud Rate Register). Ο UBRR είναι ένας καταχώρησης εγγραφής/ανάγνωσης ο οποίος καθορίζει το baud rate της USART.

Επειδή το πρωτόκολλο DMX 512 έχει συγκεκριμένο baud rate πρέπει να ορίσουμε στον επεξεργαστή το σωστό baud rate.Για το πρωτόκολλο DMX 512 το baud rate είναι 250 Kbps.Αυτό ορίζεται ως εξής, αφού έχουμε κρύσταλλο 16 MHz.

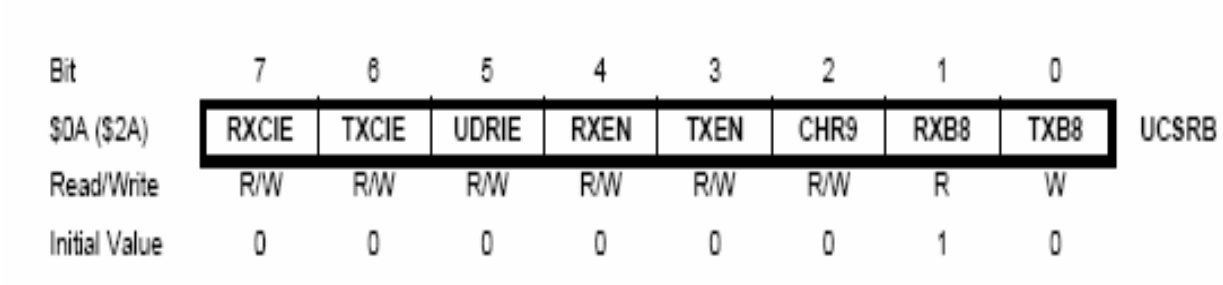
Συμφωνά με την εξίσωση προκύπτει το σωστό baud rate:

$$BAUD = \frac{Fck}{16(UBRR+1)}$$

Οπου Fck=16MHz και UBRR=3 Εχουμε baud rate ίσο με 250 Kbps. Άρα στον UBRR πρέπει να φορτώσουμε την τιμή 3 (UBRR = 3) [3].

5.3.3 ΕΝΕΡΓΟΠΟΙΗΣΗ UCSRB ΓΙΑ ΤΗΝ ΜΕΤΑΔΟΣΗ ΤΟΥ DMX512

Σε αυτό το σημείο της εργασίας θα δούμε πως γίνεται η ενεργοποίηση του UCSRB. Με τον σωστό ορισμό του UCSRB δίνεται η δυνατότητα χρήσης και επιπλέον bits έτσι ώστε να δημιουργηθεί ο start code που χρειάζεται ένα start bit και δυο stop bit. Παρακάτω να εξηγηθεί αναλυτικά η λειτουργία του UCSRB.



Bit	7	6	5	4	3	2	1	0	
$0A ($2A)	RXCIE	TXCIE	UDRIE	RXEN	TXEN	CHR9	RXB8	TXB8	UCSRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	W	
Initial Value	0	0	0	0	0	0	1	0	

- .Bit 7 – RXCIE

Όταν αυτό το bit τίθεται “1” ,θέση (1) του RXC bit στον USR θα προκαλέσει την εκτέλεση της ρουτίνας receive complete interrupt δεδομένου ότι τα global interrupts είναι ενεργοποιημένα.

- Bit 6 - TXCIE

Όταν αυτό το bit τίθεται “1”, η θέση (1) του TXC bit στον USR θα προκαλέσει την εκτέλεση της ρουτίνας transmit complete interrupt δεδομένου ότι τα global interrupts είναι ενεργοποιημένα.

- Bit 5 – UDRIE

Όταν αυτό το bit τίθεται “1”, η θέση (1) του UDRE bit στον USR θα προκαλέσει την εκτέλεση της ρουτίνας UART data register empty interrupt δεδομένου ότι τα global interrupts είναι ενεργοποιημένα.

- Bit 4 – RXEN

Αυτό το bit όταν τίθεται “1” ,ενεργοποιεί τον δεκτή της UART .Όταν ο δεκτής απενεργοποιείται ,οι TXC,OR και FE status flags δεν μπορούν να τοποθετηθούν (1). Εάν αυτές οι σημαίες είναι σε θέση (1), η απενεργοποίηση του RXEN δεν μπορεί να προκαλέσει μηδενισμό.

- Bit 3 - TXEN

Αυτό το bit όταν τίθεται "1" ,ενεργοποιεί τον μεταδότη της UART. Όταν απενεργοποιείται ο μεταδότης την ώρα που μεταδίδει έναν χαρακτήρα ,ο μεταδότης παραμένει ενεργοποιημένος μέχρι ο χαρακτήρας στον shift register και τον χαρακτήρα που ακολουθεί στον UDR, να μεταδοθούν ολοκληρωτικά.

- Bit 2 – CHR9

Όταν αυτό το bit τίθεται "1" , οι σταλθεντες και ληφθεντες χαρακτήρες έχουν μήκος 9 bits,συν τα start και τα stop bits. Το ένατο bit διαβάζεται και εγγράφεται χρησιμοποιώντας τα RXB8 και TXB8 bits στον UCR, αντίστοιχα. Το ένατο data bit μπορεί να χρησιμοποιηθεί σαν επιπλέον stop bit .

- Bit 1 – RXB8

Όταν το CHR9 τίθεται "1" το RXB8 είναι το ένατο bit του ληφθεντος χαρακτήρα.

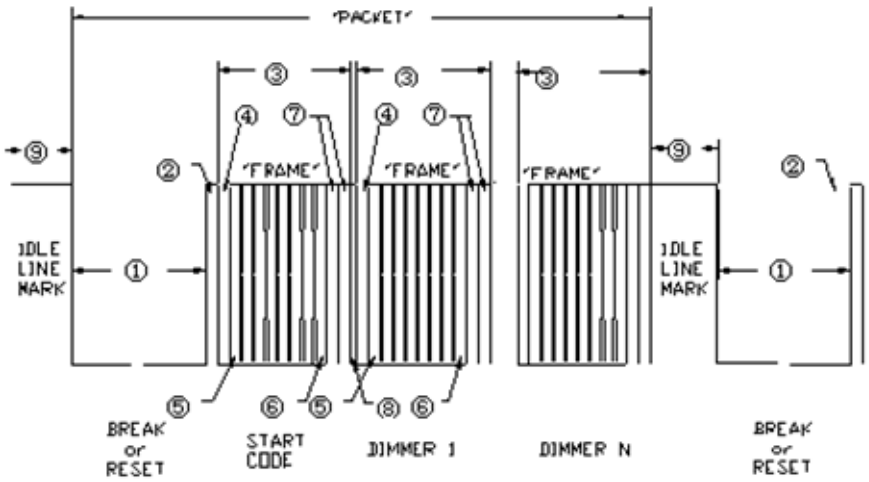
- Bit 0 – TXB8

Όταν το CHR9 τίθεται "1" , το TXB8 είναι το ένατο bit του χαρακτήρα που θα μεταδοθεί.

5.3.4 ΜΕΤΑΔΟΣΗ BREAK ΚΑΙ MAB

Συμφωνά με τα δεδομένα του USSIT η διάρκεια του BREAK και του MAB πρέπει να είναι 88μsec και 8μsec αντίστοιχα. Σε αυτό το σημείο δημιουργείται πρόβλημα καθώς η διάρκεια του BREAK ισούται με τις διάρκειες δυο "data frames", αλλά στα data frame περιέχουν ένα start και δυο stop bit. Όμως το break πρέπει να είναι μια συνεχείς low κατάσταση που έχει διάρκεια 88μsec, αυτό όμως δεν μπορεί να γίνει μέσω της USART καθώς θα υπάρχουν το start και το stop bit. Για την λύση αυτού του προβλήματος, αφού απενεργοποιήσαμε την USART, πήραμε έξοδο από την PORTB του μικροελεγκτη, και μέσω μιας ρουτίνας delay θέσαμε 88μsec μια low κατάσταση. Έτσι δημιουργήσαμε το BREAK. Το ίδιο έγινε και με το MAB αλλά μέσω μιας ρουτίνας delay ορίσαμε την διάρκεια του στα 8 μsec. Μετά από αυτήν την διαδικασία ξανά ενεργοποιήσαμε

την USART για να γίνει η μετάδοση του START CODE και των υπολοίπων DATA FRAME.



Εικόνα 5.3.1: DMX512 packet

6 ΣΧΕΔΙΑΣΜΟΣ SOFTWARE (ΚΩΔΙΚΑ)

6.1 ΓΕΝΙΚΑ

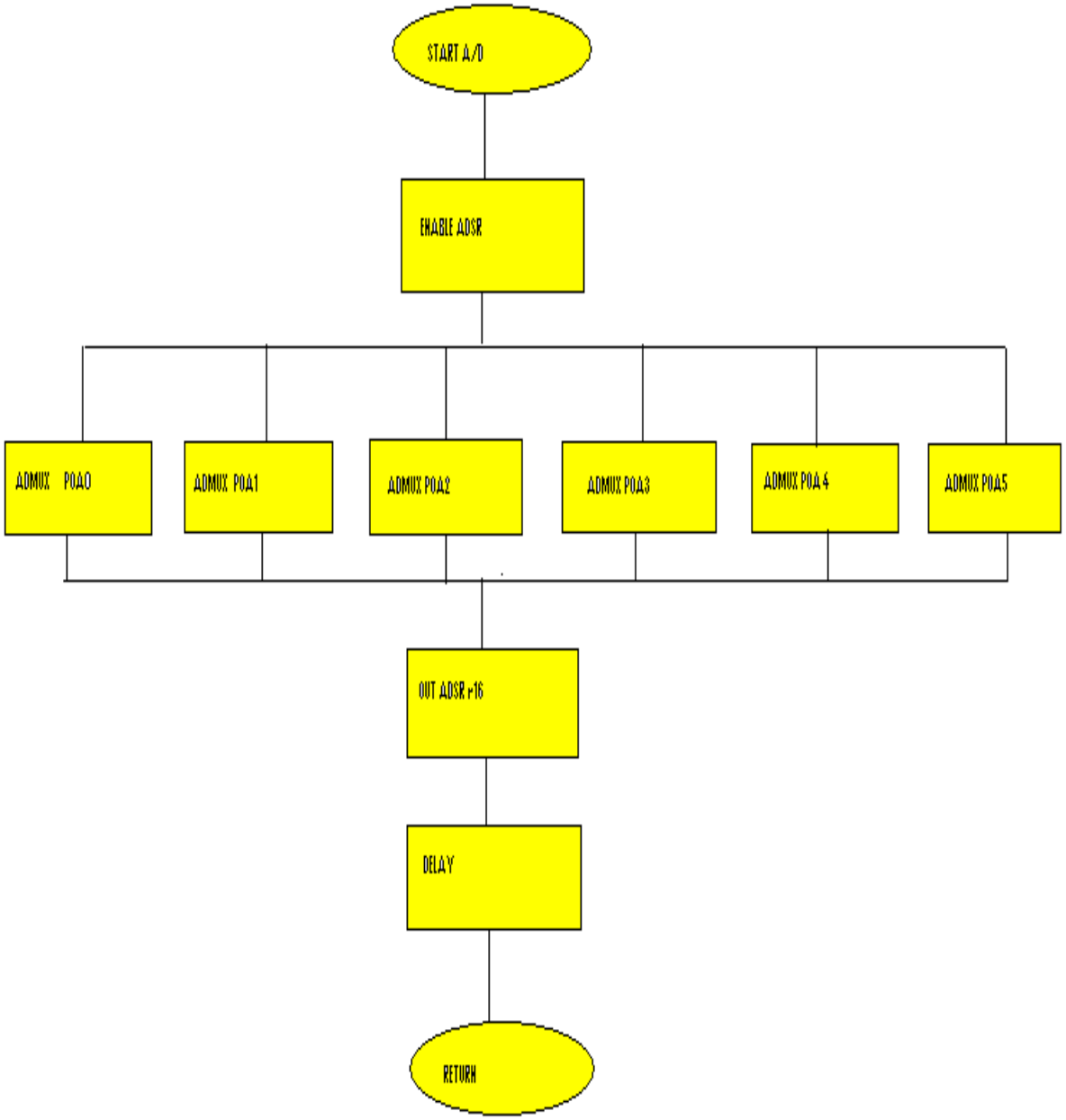
Σε αυτό το μέρος της εργασίας θα περιγράψουμε πως έγινε ο σχεδιασμός του κώδικα. Ο κώδικας γράφτηκε σε γλωσσά assembly μέσω του προγράμματος AVR STUDIO 4. Ο προγραμματισμός του μικροελεγκτη έγινε μέσω του αναπτυξιακού STK 500. Οι τρεις βασικές λειτουργίες του συγκεκριμένου προγράμματος είναι:

- Η ενεργοποίηση της A/D λειτουργίας του AT mega163.
- Η δημιουργία και μετάδοση του DMX512 πακέτου.
- Και η χρήση ως πληροφορίας για τα data frame τα αποτελέσματα της A/D λειτουργίας.

6.2 A/D ΛΕΙΤΟΥΡΓΙΑ ΚΑΙ ΜΕΤΑΔΟΣΗ DMX512

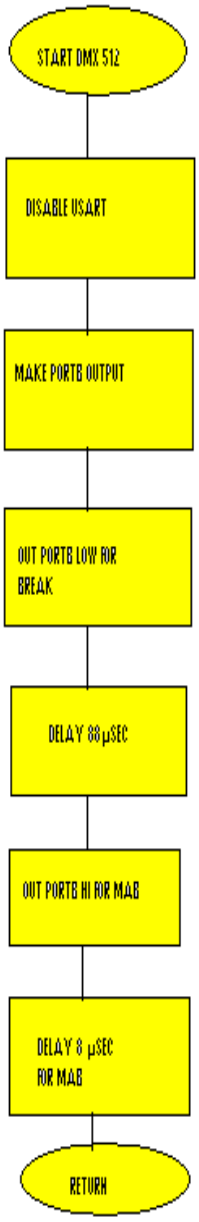
Σε αυτό το πρόγραμμα τρέχουν ταυτόχρονα τρεις ρουτίνες πρώτη ρουτίνα ενεργοποιεί την A/D λειτουργία. Στην δεύτερη ρουτίνα γίνεται απενεργοποίηση της USART και κάνουμε την portB έξοδο έτσι ώστε να δημιουργήσουμε 88μsec low για break, και 8μsec high για mab. Και στην τρίτη ρουτίνα ενεργοποιείται η USART για την μετάδοση του start code και των data frame. Επίσης στα data frame φορτώνονται τα αποτελέσματα της A/D λειτουργίας.Παρακάτω παρουσιάζονται τα block διαγράμματα τους.

6.3 A/D ΡΟΥΤΙΝΑ



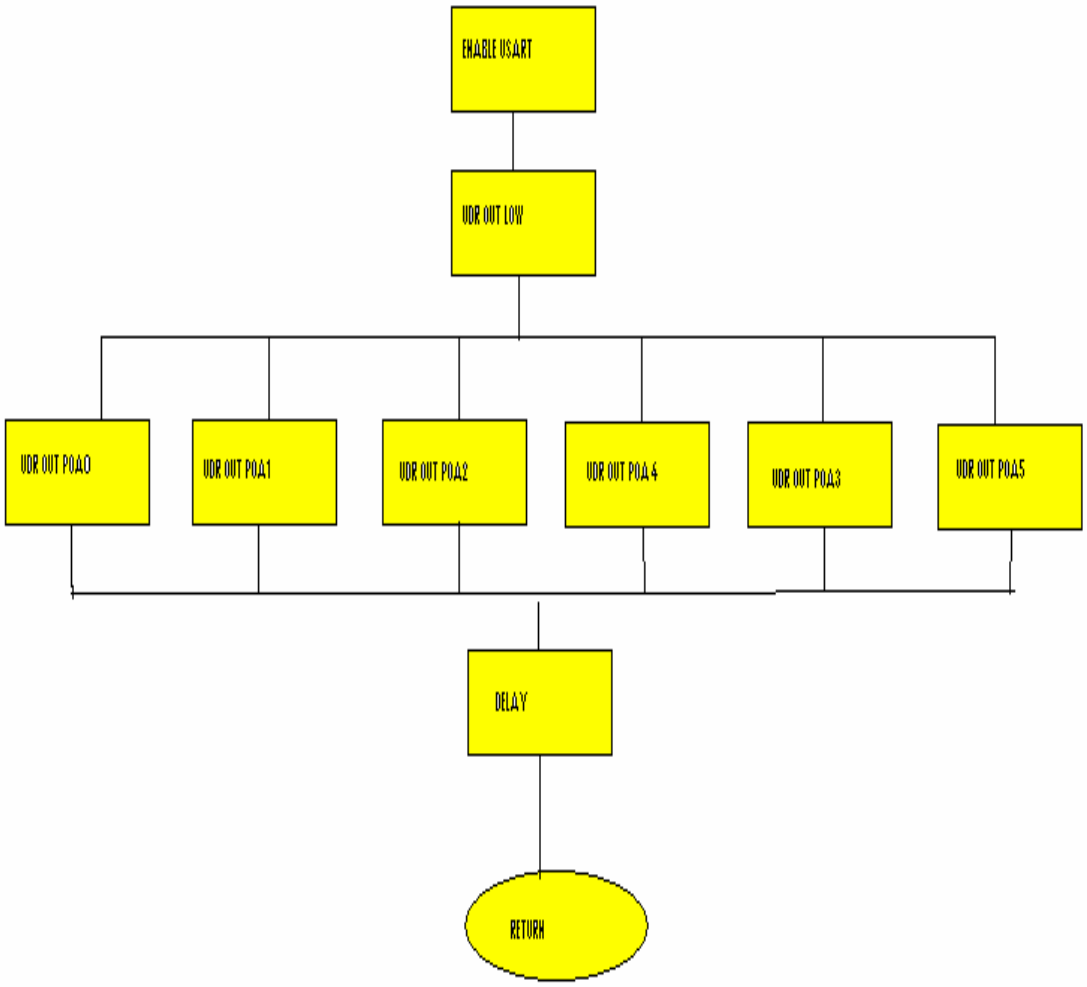
Εικόνα 6.3.1: A/D ρουτινα

6.4 DMX512 ΡΟΥΤΙΝΑ



Εικόνα 6.4.1: DMX512 ρουτινα

6.5 ΜΕΤΑΔΟΣΗ DMX512 ΡΟΥΤΙΝΑ



Εικόνα 6.5.1: μεταδοση DMX512 ρουτινα

7 ΜΕΤΡΗΣΕΙΣ ΚΑΙ ΔΟΚΙΜΕΣ

7.1 ΓΕΝΙΚΑ

Σε αυτό το μέρος της εργασίας έγιναν μια σειρά από μετρήσεις και δοκιμές για να διαπιστωθεί η σωστή λειτουργία της εφαρμογής. Μέσω αυτόν το μετρήσεων και των δοκιμών διαπιστώθηκε ότι και το software μέρος της εφαρμογής και το hardware μέρος λειτουργούν κανονικά. Για τις μετρήσεις που έγιναν χρησιμοποιήθηκε ο ψηφιακός παλμογράφος Tektronix TDS2024B.

Πιο συγκεκριμένα οι δοκιμές έγιναν για να διαπιστωθεί :

- Αν η εφαρμογή λειτουργεί συμφωνά με τα DMX512 στανταρ.
- Αν λειτουργεί σωστά η A/D λειτουργία.
- Και αν EIA/RS-485 μας δίνει balance σήμα.

7.2 ΜΕΤΡΗΣΕΙΣ DMX512 PACKET

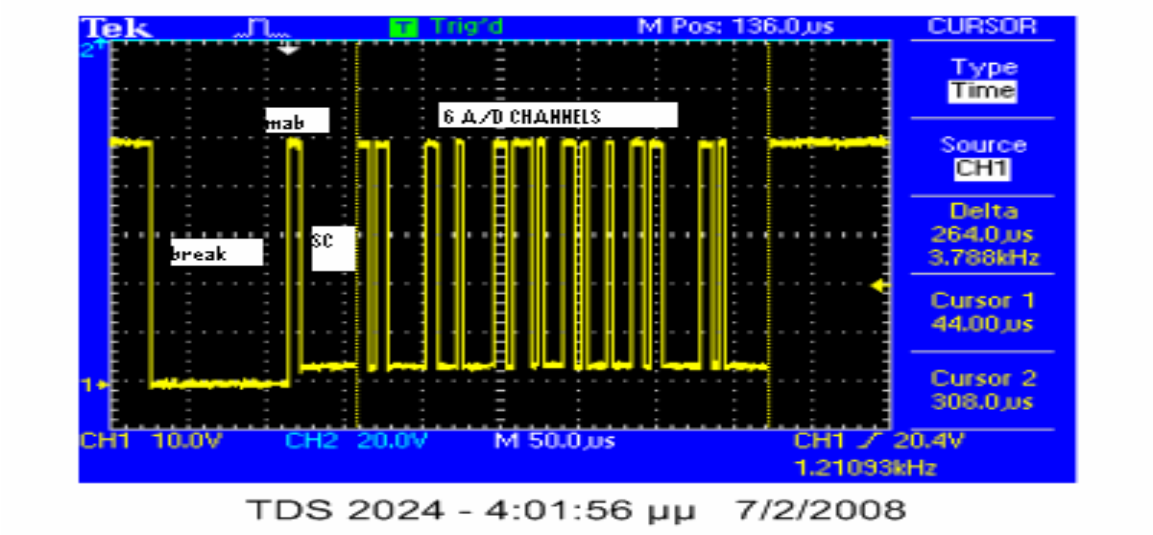
Σε αυτήν την παράγραφο θα διαπιστωθεί ότι η εφαρμογή παράγει και μεταδίδει DMX512 σήμα συμφωνά με τα δεδομένα της USSIT όπως φαίνεται στο παρακάτω πίνακα.

DMX512 (1990) timing chart

Description	MIN	TYP	MAX	UNIT
BREAK	88	88	1000000	usec
MAB		8		usec
FRAME WIDTH		44		usec
START/DATA/STOP BITS		4		usec
MTBF	0	NS	1000000	usec
MTBP	0	NS	1000000	usec

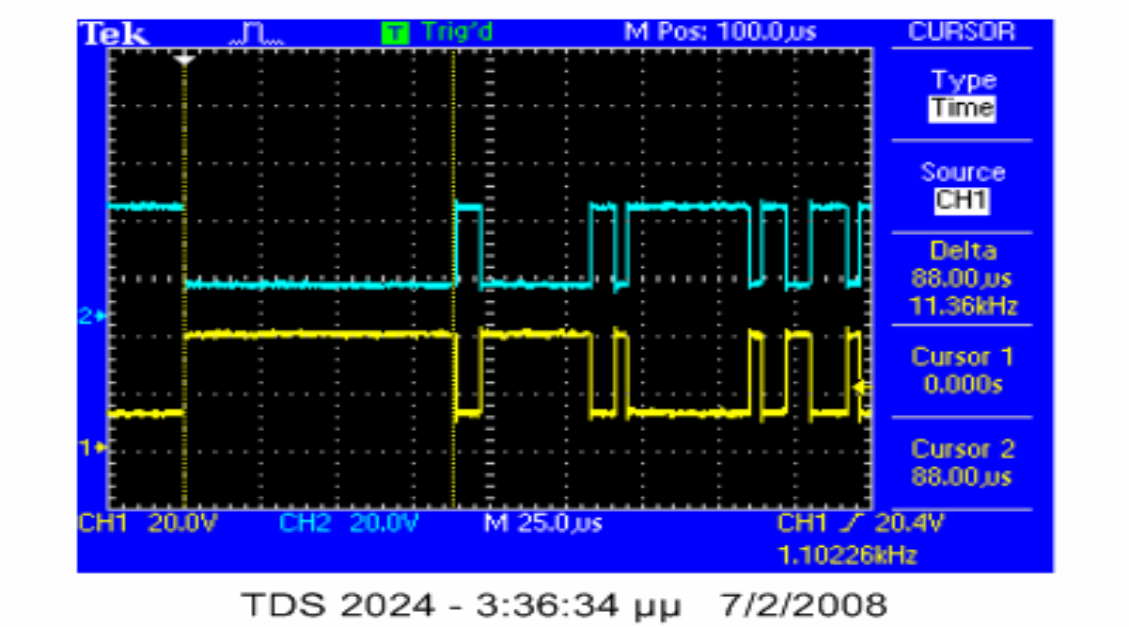
Πινακας 7.2.1 : διαρκειες DMX512

Στην παρακάτω εικόνα φαίνεται μια ολοκληρωμένη αποστολή DMX512 πακέτου με 6 data frame (εικ 7.1.1).

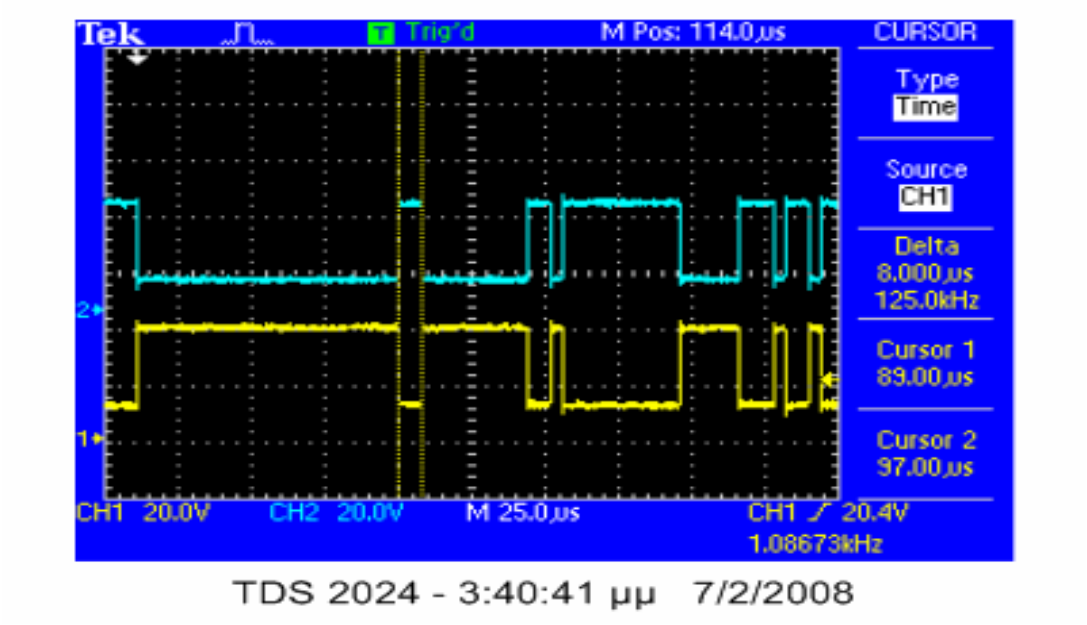


Εικ 7.2.1: DMX512 PACKET

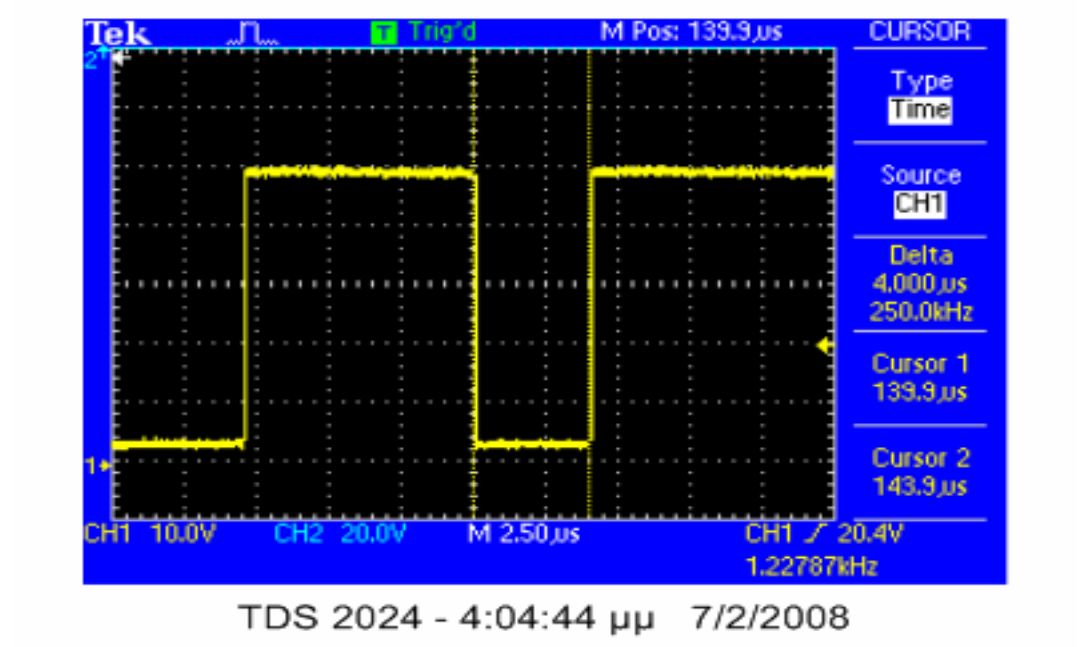
Στις παρακάτω εικόνες φαίνονται οι διάρκειες του break, του Mab , καθώς και η διάρκεια του κάθε bit. Παρατηρώντας τις παρακάτω εικόνες βλέπουμε ότι και πειραματικά προκύπτει ότι το break , mab και η διάρκεια του κάθε bit έχουν τις διάρκειες που έχουν οριστεί από την USSIT (εικ.7.2.2) (εικ.7.2.3) (εικ.7.2.4).



Εικ 7.2.2:Διαρκεια break



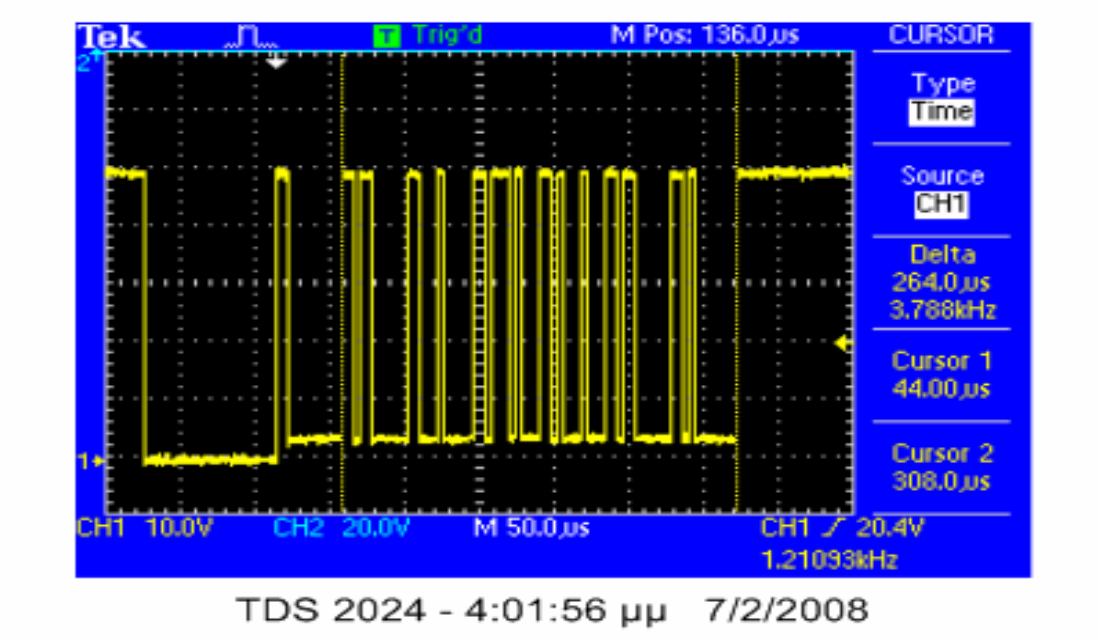
Εικ 7.2.3:Διαρκεια mab



Εικ 7.2.4:Διαρκεια bit

7.3 ΜΕΤΡΗΣΕΙΣ ΓΙΑ A/D ΛΕΙΤΟΥΡΓΙΑ

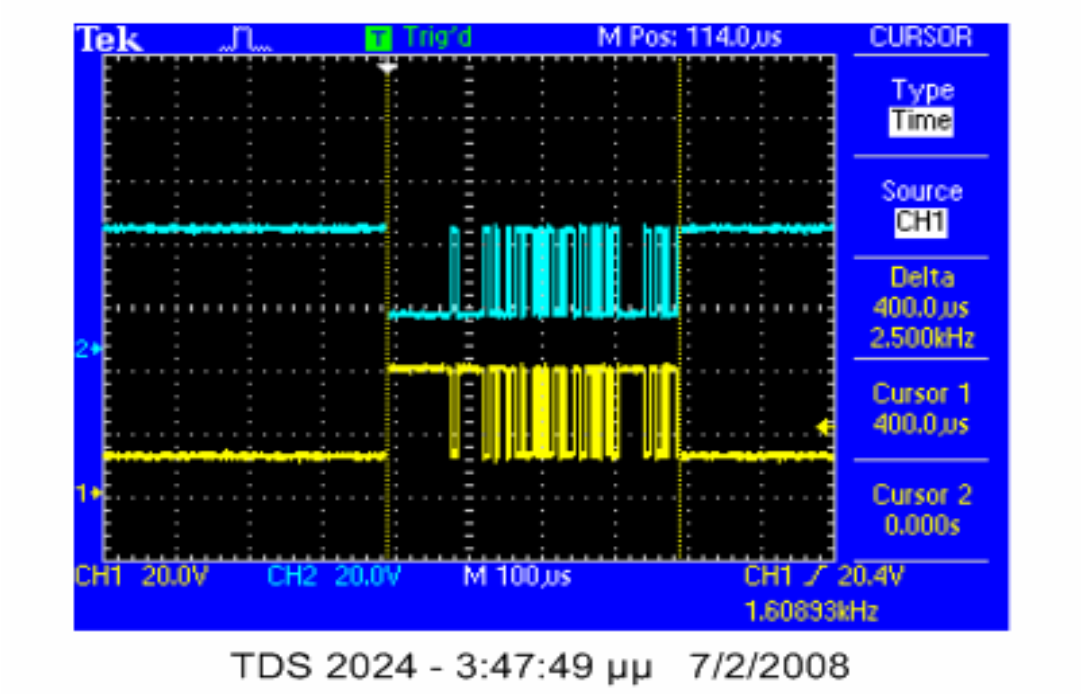
Σε αυτήν την παράγραφο θα δείξουμε μέσω της πειραματικής διαδικασίας την σωστή λειτουργία των A/D μετατροπιών. Μέσω αυτών τον μετρήσεων παρατηρούμε τις αλλαγές καταστάσεων των 6 data frame του DMX512 πακέτου που προκύπτουν από τις μεταβολές των μεταβλητών αντιστατών.



Εικ 7.3.1:6 A/D κανάλια

7.4 ΜΕΤΡΗΣΕΙΣ ΓΙΑ DS/EIA-485

Η τελική μετάδοση του DMX512 σήματος θα γίνει μέσω της DS/EIA-485 διατάξεις. Αυτή η διάταξη μετατραπεί το σήμα που έρχεται από το μικροελεγκτη μέσω της Tx από unbalance σε balance όπως φαίνεται στην παρακάτω εικόνα (εικ.7.4.1) .



Εικ 7.4.1: DMX512 PACKET BALANCE

8.ΑΝΑΛΥΣΗ FIRMWARE

Ο προγραμματισμός του μικροελεγκτη έγινε σε γλωσσά assembly, το firmware φορτωνόταν στον επεξεργαστή από το ηλεκτρονικό υπολογιστή που ήταν συνδεμένος σειριακά με τον αναπτυξιακό STK 500.
Η ανάλυση του κώδικα θα γίνει σε βήματα.

8.1 ΔΗΛΩΣΗ ΜΕΤΑΒΛΗΤΩΝ

```
; Used registers

.def   strT                = r16
.def   delay1              = r17
.def   delay2              = r18
.def   delay3              = r19
.def       chs              = r20
.def   HI                  = r21
.def   accu1               = r22
.def   UART_CONTROL        = r23
.def   LO                  = r24
.def       Poa0             = r25
.def       Poa1             = r26
.def       Poa2             = r27
.def       Poa3             = r28
.def       Poa4             = r29
.def   Poa5                = r30
.def   Poa6                = r31
.equ    ADCSR              = $06
```

Ολα τα δεδομένα των μεταβλητών οπού χρησιμοποιήσαμε φορτώνονται στους 16 καταχώρησες γενικής χρήσεως οπού υπάρχουν στο ATmega163 (r16-r31).

8.2 ΕΝΕΡΓΟΠΟΙΗΣΗ SP ΚΑΙ A/D ΛΕΙΤΟΥΡΓΙΑ

- Ενεργοποίηση stack pointer και ενεργοποίηση A/D λειτουργίας στον ATmega163.

```
main:    ldi     r16,high(RAMEND) ; Initiate Stackpointer.
         out       SPH,r16
          ldi            r16,255
         out  ddrb,r16          ; portB EXODOS
         clr   r16
          out ddra,r16            ; portA EISODOS
          ldi  r16,$8f                    ; pull-up only for inputs that
          out porta,r16          ; are not analog inputs

          ldi  r16,$D3
          out ADCSR,r16
```

Ο stack pointer πρέπει να είναι ορισμένος από το πρόγραμμα πριν από την εκτέλεση κάθε subroutine call η από την ενεργοποίηση interrupt.

Για την ενεργοποίηση του stack pointer φορτώνουμε την τιμή 255 στον r16 καταχώρηση. Αφού καθαρίσουμε των καταχώρηση r16 από τα δεδομένα κάνουμε την port A είσοδος (“out ddra,r16”). Μετά φορτώνουμε στον r16 την τιμή “$D3 και βγάζουμε έξοδο τον r16 στον ADCSR. Μέσω του ADCSR καταχώρηση ενεργοποιούμε την A/D λειτουργία του μικροελεγκτη AT mega 163. Αυτο γίνεται ως εξής:

Bit	7	6	5	4	3	2	1	0
$D6 ($26)	ADEN	ADSC	ADFR	ADIF	ADIE	ADPS2	ADPS1	ADPS0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value	0	0	0	0	0	0	0	0

Το κάθε “bit” ADCSR ανάλογα με την κατάσταση την οποία(0 η 1) βρίσκεται εκτελεί και διαφορετική λειτουργία.

Παρακάτω εξηγούνται το που επηρεάζει το κάθε “bit”

1. Bit 7 – ADEN

Οήταν αυτό το bit είναι λογικό ένα “1”, ενεργοποιούνται οι A/D μετατροπείς του AT mega 163. Στην αντίθετη περίπτωση δηλαδή όταν αυτό το bit είναι λογικό μηδέν “0” τότε οι μετατροπείς είναι απενεργοποιημένοι.

2. Bit 6 – ADSC

Με αυτό το bit ανάλογα ποτέ θα το κάνουμε “1” έχουμε και διαφορετική λειτουργία .Στην πρώτη λειτουργία που ονομάζεται “single conversion mode” αυτό το bit πρέπει να γίνεται λογικό “1” κάθε φορά που έχουμε A/D μετατροπή. Στην δεύτερη λειτουργία που ονομάζεται “free running mode” αυτό το bit πρέπει να γίνει λογικό “1” όταν αρχίζει η πρώτη A/D μετατροπή .Κάνοντας το bit “0” δεν υπάρχει καμιά επίδραση.

3. Bit 5 – ADFR

Όταν θέσουμε αυτό το bit λογικό “1” τότε η μετατροπείς δουλεύουν σε “free running mode”. Όταν αυτό το bit το κάνουμε λογικό “0” σταματάει η free running λειτουργία.

4. Bit 4 – ADIF

Αυτό το bit θέτεται λογικό “1” όταν ολοκληρώνεται η A/D μετατροπή και η data registers ανανεώνονται

5. Bit 3 – ADIE

Οταν αυτό το bit θέτεται λογικό “1” και I – bit του status register είναι λογικό “1” η ADC conversion complete interrupt ειναι ενεργη.

6. Bit 2 – ADPS2...0

Αυτό το bit καθορίζει τον χρονισμό ανάμεσα στον κρύσταλλο και το εσωτερικό ρολόι του ADC.

Table 42. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

Στον κώδικα η τιμή ADCSR είναι “$D3” δηλαδή 10010011 που συμφωνά με τα ανώτερο σημαίνει:
Ότι είναι ενεργοποιημένη η ADC λειτουργία(πρώτο bit “1”),αλλά δεν έχει ξεκινήσει ακόμα στην μετατροπή από αναλογικό σε ψηφιακό σήμα καθώς το bit 6 είναι “0”. Το bit 6 γίνεται λογικό “1” στην συνεχεία του κώδικα. Επειδή το bit 5 είναι λογικό “0” χρησιμοποιείται η single mode conversion. Τα interrupt flag είναι ενεργά(bit 4 λογικό “1”).Το bit 3 είναι λογικό “0” άρα το ADC interrupt είναι ανενεργό. Το bit 1,0 είναι λογικό “1” και από τον παραπάνω πίνακα προκύπτει 1MHz για το ρολόι του ADC(8MHz/8=1MHZ).

8.3 ΕΠΙΛΟΓΗ A/D ΜΕΤΑΤΡΟΠΕΩΝ

- Σε αυτό το μέρος του κώδικα γίνεται η επιλογή ποιος από τους 8 A/D μετατροπείς θα επιλεχθεί ,καθώς θέτουμε και το bit 6 του ADCSR λογικό “1” έτσι ώστε να ξεκινήσει η single conversion λειτουργία.

```
;********************DATA_0************************************


LOOP:

                ldi   chs,$20              ; input ADC0 selected first
sadc1:    out       ADMUX,chs
                in    r16,ADCSR
                sbr   r16,(1<<6)                    ; start coversion, when completed bit6 returns to 0 automatically
                out   ADCSR,r16

             ldi delay3,255
DELAYADC0:
              dec delay3
              brne DELAYADC0

              in       Poa0,ADCH
```

Στην μεταβλητή chs(r20) φορτώνουμε την τιμή $20 που είναι ο πρώτος A/D μετατροπίας και τον κάνουμε έξοδο στον καταχώρηση ADMUX(παρακάτω φαίνεται αναλυτικά η λειτουργία του ADMUX). Μεταφέρουμε τις τιμές του καταχωριστή ADCSR στον καταχωριστή γενικής χρήσεως r16 και με την εντολή sbr θέτουμε το bit 6 του ADCSR σε λογικό “1 “ έτσι ώστε να ξεκινήσει η A/D μετατροπή .Φορτώνουμε τις τιμές του r16 στον καταχωριστή ADCSR και τον κάνουμε έξοδο. Απο τον επεξεργαστή έχει διευσιοδοτηθει τα αποτελέσματα της μετατροπής να αποθηκεύονται στον καταχώρηση ADCH. Τα δεδομένα του ADCH τα μεταφέρουμε σε ένα καταχώρηση γενικής χρήσεως που τον ονομάζουμε poa0. Σε αυτήν όλη την διαδικασία έχει προστεθεί μια ρουτίνα delay έτσι ώστε να προλαβαίνει να γίνονται η υπολογισμοί.

8.4 ADMUX ΛΕΙΤΟΥΡΓΙΑ

Το κάθε “bit” του ADMUX ανάλογα με την κατάσταση την οποία(0 η 1) βρίσκεται εκτελεί και διαφορετική λειτουργία.

Παρακάτω εξηγούνται το που επηρεάζει το κάθε “bit”

Bit	7	6	5	4	3	2	1	0	
$07 ($27)	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

1. Bit 7,6 –REFS1…0

Αυτα τα bits διαλέγουν την τάση αναφοράς για τους A/D μετατροπείς . Εάν αυτά τα bits αλλάξουν κατά την διάρκεια της μετατροπής, δεν υπάρχει επίδραση της αλλαγής αυτής μέχρι να τελειώσει το conversion. Ο χρηστής πρέπει να αμελήσει τα πρώτα αποτελέσματα τής μετατροπής, εάν έχουν αλλάξει αυτά τα bits, έτσι ώστε να έχει την μέγιστη ακρίβεια.

2. Bit 5 – ADLAR

Το ADLAR bit έχει επίδραση στα αποτελέσματα της μετατροπής στον καταχώρηση ADC Data. Εάν το ADLAR bit είναι λογικό “0” , τα αποτελέσματα έχουν δεξιά διευθέτηση. Εάν το ADLAR bit είναι λογικό “1” , τα αποτελέσματα έχουν αριστερή διευθέτηση. Εάν αλλάξει το ADLAR bit θα υπάρχει άμεση επίδραση άσχετα αν υπάρχει τρέχων μετατροπή.

3.Bits 4…0 MUX4….MUX0

Αυτά τα bits επιλεγούν ποια αναλογική είσοδος είναι συνδεμένη στον ADC. Εάν αλλάξουν αυτά τα bits κατά την διάρκεια μιας μετατροπής δεν θα υπάρχει επίδραση μέχρι να ολοκληρωθεί η συγκεκριμένη μετατροπή διευθυνσιοδοτησεις φαίνονται στον παρακάτω πίνακα.

Table 41. Input Channel Selections

MUX4..0	Single-ended Input
00000	ADC0
00001	ADC1
00010	ADC2
00011	ADC3
00100	ADC4
00101	ADC5
00110	ADC6
00111	ADC7

Πίνακας 8.1:πινακας επιλογής A/D μετατροπιών

Αυτή η διαδικασία επαναλαμβάνεται άλλες τέσσερις φορές με την διαφορά ότι κάθε φορά αλλάζει η τιμή του καταχώρηση γενικής χρήσεως chs(r20), έτσι ώστε να χρησιμοποιείται διαφορετικός A/D μετατροπεας. Επίσης κάθε φορά τα αποτελέσματα το ADCH μεταφέρονται και σε άλλον καταχώρηση(poa1, poa2, poa3, poa4) για να χρησιμοποιηθούν σε παρακάτω μέρος του προγράμματος. Αυτό το μέρος του κώδικα φαίνεται παρακάτω.

```
;********************DATA_1************************************

            ldi         chs,$21   ; input ADC1 selected first
sadc2:      out         ADMUX,chs
            in          r16,ADCSR
            sbr         r16,(1<<6)          ; start coversion, when completed bit6 returns to 0 automatically
            out         ADCSR,r16

ldi delay3,255
DELAYADC1:
dec delay3
brne DELAYADC1

            in          Poa1,ADCH




;********************DATA_2************************************


            ldi         chs,$22   ; input ADC2 selected first
sadc3:      out         ADMUX,chs
            in          r16,ADCSR
            sbr         r16,(1<<6)          ; start coversion, when completed bit6 returns to 0 automatically
            out         ADCSR,r16



      in    Poa2,ADCH

ldi delay3,200
DELAYADC2:
dec delay3
brne DELAYADC2

;********************DATA_3************************************


            ldi         chs,$23             ; input ADC3 selected first
sadc4:      out         ADMUX,chs
            in          r16,ADCSR
            sbr         r16,(1<<6)          ; start coversion, when completed bit6 returns to 0 automatically
            out         ADCSR,r16
```

```
            in          Poa3,ADCH

ldi delay3,200
DELAYADC3:
dec delay3
brne DELAYADC3

;********************DATA_4************************************


            ldi         chs,$24   ; input ADC4 selected first
sadc5:      out         ADMUX,chs
            in          r16,ADCSR
            sbr         r16,(1<<6)          ; start coversion, when completed bit6 returns to 0 automatically
            out         ADCSR,r16



            in          Poa4,ADCH

ldi delay3,200
DELAYADC4:
dec delay3
brne DELAYADC4

;********************DATA_5************************************


            ldi         chs,$25   ; input ADC5 selected first
sadc6:      out         ADMUX,chs
            in          r16,ADCSR
            sbr         r16,(1<<6)          ; start coversion, when completed bit6 returns to 0 automatically
            out         ADCSR,r16



            in          Poa5,ADCH

ldi delay3,200
DELAYADC5:
dec delay3
brne DELAYADC5
```

8.5 ΔΗΜΙΟΥΡΓΙΑ DMX512

Σε αυτό μέρος του κώδικα έχει δημιουργηθεί το DMX πακέτο . Για να είναι ένα DMX πακέτο σωστό έτσι ώστε να λαμβάνεται από έναν DMX δεκτή θα πρέπει να επιμέρους μέρη του DMX να έχουν την κατάλληλη χρονική διάρκεια. Οι χρόνοι αυτόν φαίνονται στον παρακάτω πίνακα.

DMX512 (1990) timing chart

Description	MIN	TYP	MAX	UNIT
BREAK	88	88	1000000	usec
MAB		8		usec
FRAME WIDTH		44		usec
START/DATA/STOP BITS		4		usec
MTBF	0	NS	1000000	usec
MTBP	0	NS	1000000	usec

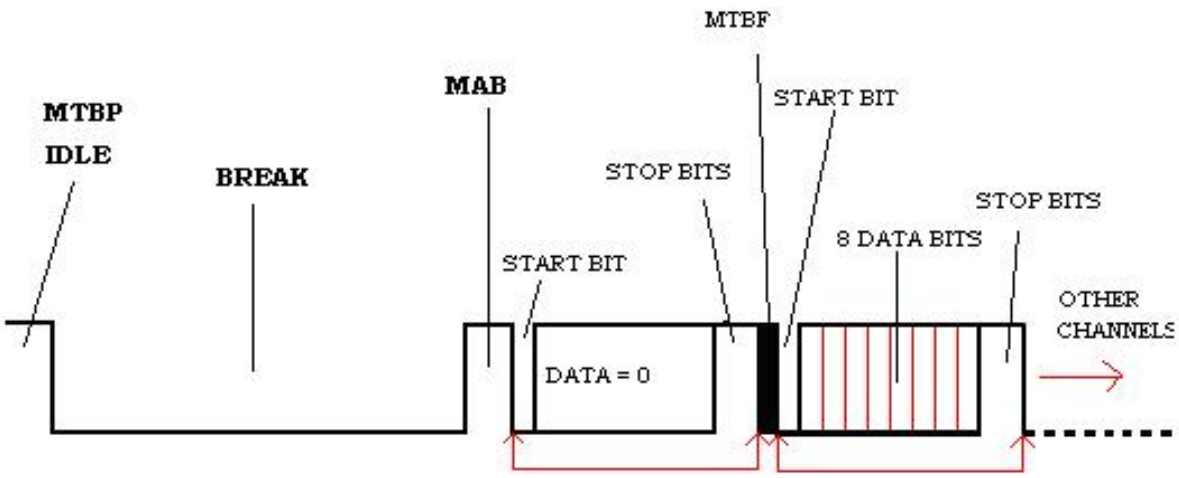
πίνακας 8.2 :χρόνοι DMX512

Στο μέρος του κώδικα που φαίνεται παρακάτω δημιουργείται το break και το mark after break.

```
;*******************DMX PACKET************************************

ldi HI,0b11111111       ;fortonoume 2 katastaseis Hi Logia BREAK & MAB
ldi LO,0b00000000

ldi accu1,1             ;Rithmizei to boud rate 250 Kbps
out UBRR,accu1
ldi UART_CONTROL,0b00001101

ldi accu1,0
```

```
OUT UCSRB,accu1

out DDRD,HI              ;Kanei ti thira D exodo

out PORTD,LO             ;Bfazei LOW gia to BREAK

ldi delay1,234

B:                  ;Kathisterei 88microsecond gia to BREAK
  dec delay1
  brne B


out PORTD,HI             ;Bgazei to HI gaia to MAB

ldi delay2,22
MAB:                ;Kathisterei gia to MAB 9 microsecond
dec delay2
brne MAB
```

Οπως φαίνεται και στο παρακάτω σχήμα το break είναι μια low (λογικο”0”) κατάσταση που έχει διάρκεια 88μsec. Ενώ το mark after break είναι μια hi (λογίκο “1”)κατάσταση που έχει διάρκεια 8μsec.



Εικονα 8.1: break ,mab, start code και 1 data byte

Για να πετύχουμε το break φορτώνουμε στον καταχώρηση LO(r24) την τιμή 0B00000000.Και για την δημιουργία του mark after break φορτώνουμε στον καταχώρηση HI την τιμή 0b11111111.

Επειδή το πρωτόκολλο DMX 512 έχει συγκεκριμένο baud rate πρέπει να ορίσουμε στον επεξεργαστή το σωστό baud rate. Για το πρωτόκολλο DMX 512 το baud rate είναι 250 Kbps. Αυτό ορίζεται ως εξής, αφού έχουμε κρύσταλλο 16 MHz

Φορτώνουμε στον καταχώρηση accu1(r22) την τιμή 3. Τα δεδομένα του accu1 τα μεταφέρουμε στον καταχώρηση UBRR. Ο UBRR είναι ένας καταχώρησης εγγραφής/ανάγνωσης ο οποίος καθορίζει το baud rate της UART. Συμφωνά με την ακολουθεί εξίσωση προκύπτει το σωστό baud rate.

$$BAUD = \frac{Fck}{16(UBRR+1)}$$

Οπου Fck=16MHz και UBRR=3 Εχουμε baud rate ίσο με 250 Kbps.

Αφού έγινε ο ορισμός του baud rate κάνουμε έξοδο την θύρα D. Ββγάζουμε στην έξοδο D το low και μέσω τις ρουτίνας delay 1 πετυχαίνουμε μια καθυστέρηση 88μsec, άρα έχει σχηματιστεί το break. Στην συνεχεία βγάζουμε στην έξοδο D το HI γιατί το mark after break είναι μια high κατάσταση διαρκείας 8μsec τουλάχιστον διάρκεια του mark after break ορίζεται από την ρουτίνα delay 2, που μας δίνει μια καθυστέρηση 8 μsec.

Στο τελευταίο μέρος του κώδικα γίνεται η ενεργοποίηση του UCSRB.Με τον σωστό ορισμό του UCSRB δίνεται η δυνατότητα χρήσης και επιπλέον bits έτσι ώστε να δημιουργηθεί ο start code που χρειάζεται ένα start bit και δυο stop bit. Παρακάτω να εξηγηθεί αναλυτικά η λειτουργία του UCSRB. Στην συνεχεία δημιουργούνται 6 data κανάλια έτσι ώστε να φορτωθούν τα αποτελέσματα από τις A/D μετατροπές.

Ακολουθεί ο κώδικας:

```
ldi accu1,0b00001111        ;Energopoioume tin UART TXE=1, CHR9=1, TXB8=1
out UCSRB,accu1

out UDR,LO                  ;Bgazei to Stast Code
SECOND_DATA1:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA1

out UDR,Poa0                ;Bgazei to Proto DATA BYTE Ch1
SECOND_DATA2:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA2


out UDR,Poa1                ;Bgazei to Deutero DATA BYTE Ch2
SECOND_DATA3:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA3

out UDR ,Poa2               ;Bgazei to Trito DATA BYTE Ch3
SECOND_DATA4:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA4


out UDR,Poa3                ;Bgazei to TETARTO DATA BYTE Ch4
SECOND_DATA5:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA5


out UDR,Poa4                ;Bgazei to PEMTO DATA BYTE Ch5
SECOND_DATA6:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA6


out UDR,Poa5                ;Bgazei to EKTO DATA BYTE Ch6
SECOND_DATA7:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA7



ldi delay2,250
ldi delay3,250
DELAY_PACKET:
```

```
dec delay3
brne DELAY_PACKET
dec delay2
brne DELAY_PACKET


rjmp    LOOP
```

ΕΝΕΡΓΟΠΟΙΗΣΗ UCSRB :

Bit	7	6	5	4	3	2	1	0	
$0A ($2A)	RXCIE	TXCIE	UDRIE	RXEN	TXEN	CHR9	RXB8	TXB8	UCSRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	W	
Initial Value	0	0	0	0	0	0	1	0	

1.Bit 7 – RXCIE

Όταν αυτό το bit τίθεται “1” ,θέση (1) του RXC bit στον USR θα προκαλέσει την εκτέλεση της ρουτίνας receive complete interrupt δεδομένου ότι τα global interrupts είναι ενεργοποιημένα.

2.Bit 6 - TXCIE

Όταν αυτό το bit τίθεται “1”, η θέση (1) του TXC bit στον USR θα προκαλέσει την εκτέλεση της ρουτίνας transmit complete interrupt δεδομένου ότι τα global interrupts είναι ενεργοποιημένα.

3.Bit 5 – UDRIE

Όταν αυτό το bit τίθεται “1”, η θέση (1) του UDRE bit στον USR θα προκαλέσει την εκτέλεση της ρουτίνας UART data register empty interrupt δεδομένου ότι τα global interrupts είναι ενεργοποιημένα.

4.Bit 4 – RXEN

Αυτό το bit όταν τίθεται “1” ,ενεργοποιεί τον δεκτή της UART . Όταν ο δεκτής απενεργοποιείται, οι TXC, OR και FE status flags δεν μπορούν να τοποθετηθούν (1). Εάν αυτές οι σημαίες είναι σε θέση (1), η απενεργοποίηση του RXEN δεν μπορεί να προκαλέσει μηδενισμό.

5.Bit 3 - TXEN

Αυτό το bit όταν τίθεται “1”, ενεργοποιεί τον μεταδότη της UART. Όταν απενεργοποιείται ο μεταδότης την ώρα που μεταδίδει έναν χαρακτήρα, ο μεταδότης παραμένει ενεργοποιημένος μέχρι ο χαρακτήρας στον shift register συν το χαρακτήρα που ακολουθεί στον UDR, να μεταδοθούν ολοκληρωτικά.

6.Bit 2 – CHR9

Όταν αυτό το bit τίθεται “1”, οι σταλθεντες και ληφθεντες χαρακτήρες έχουν μήκος 9 bits, συν τα start και τα stop bits. Το ένατο bit διαβάζεται και εγγράφεται χρησιμοποιώντας τα RXB8 και TXB8 bits στον UCR ,αντίστοιχα. Το ένατο data bit μπορεί να χρησιμοποιηθεί σαν επιπλέον stop bit η σαν ένα bit προστιθέμενο για ισοτιμία (parity bit).

7. Bit 1 – RXB8

Όταν το CHR9 τίθεται “1” το RXB8 είναι το ένατο bit του ληφθεντος χαρακτήρα.

8.Bit 0 – TXB8

Όταν το CHR9 τίθεται “1” , το TXB8 είναι το ένατο bit του χαρακτήρα που θα μεταδοθεί.

Στο συγκεκριμένο κώδικα η τιμή που έχει πάρει ο καταχώρησης USCRB είναι 00001111 μέσω του καταχώρηση γενικής χρήσεως accu1. Αυτο σημαίνει ότι έχει ενεργοποιηθεί ο μεταδότης της UART (bit 3), επίσης επειδή το bit 2 είναι λογικό “1” η σταλθεντες και ληφθεντες χαρακτήρες έχουν μήκος 9 bits και επιπλέον ένα start και ένα stop bit(που είναι απαραίτητα για τον start code, που έχει ένα start και δυο stop bit). Οταν το bit 1 είναι λογικό “1” το RXB8 είναι το 9 bit του ληφθεντος αρχειου. Και όταν το bit 0 είναι λογικό μηδέν το TXB8 είναι το 9 data bit του χαρακτήρα που θα μεταδοθεί.

Οπότε συμφωνά με τα παραπάνω μέσω της UART έχουμε πετύχει την δημιουργία του start code καθώς έχουμε ένα start bit και δυο stop bit. Ο start code γίνεται έξοδο μέσω του καταχώρηση UDR . Στην συνεχεία μέσω του UDR καταχώρηση στέλνουμε στην έξοδο τις τιμές που έχουν προκύψει από την ψηφιοποίηση και έχουν φορτωθεί στους καταχωριστές poa0, poa1, poa2, poa3, poa4, poa5. Αυτές οι τιμές χρησιμοποιούνται σαν τιμές το 5 επομένων data bytes και ανάλογα με τις αλλαγές τον καταστάσεων τους επηρεάζουν την DMX 512 συσκευή φωτισμού.

Επειδή θέλουμε να έχουμε συνεχή μετάδοση DMX 512 πακέτων όλος ο κώδικας ειναι μέσα σε ένα loop.

9.ΣΥΜΠΕΡΑΣΜΑΤΑ-ΠΡΟΒΛΗΜΑΤΑ- ΠΛΕΟΝΕΚΤΗΜΑΤΑ

9.1ΓΕΝΙΚΑ ΣΥΜΠΕΡΑΣΜΑΤΑ

Στο τελευταίο κεφαλαίο της εργασίας συγκεντρώνονται και παρουσιάζονται όλα τα αποτελέσματα αυτής της εφαρμογής. Επίσης παρέχονται προτάσεις για την περαιτέρω ανάπτυξη αυτής της εφαρμογής. Ο κύριος σκοπός αυτής της εφαρμογής ήταν να σχεδιαστή ένα πλήρες σύστημα που να δίνει την δυνατότητα ελέγχου ενός DMX512 συστήματος μέσω αναλογικών σημάτων.

Παρατηρώντας τα αποτελέσματα της πειραματικής διαδικασίας διαπιστώνομαι ότι έχουμε την σωστή λειτουργία της σύσκεψης. Αυτό προκύπτει από την σωστή μετατροπή του αναλογικού σε ψηφιακό σήμα, καθώς και από την σωστή δημιουργία και διάδοση του DMX512 πακέτου συμφωνά με τα στανταρ της USSIT. Στο hardware μέρος της εφαρμογής ιδιαίτερη προσοχή δόθηκε στον απλό σχεδιασμό καθώς και στην χρησιμοποίηση όλων των δυνατοτήτων που έδινε ο μικροελεγκτης ATmega163.

9.2 ΠΡΟΒΛΗΜΑΤΑ ΚΑΤΑ ΤΗΝ ΥΛΟΠΟΙΗΣΗ

Κατά την διάρκεια της ανάπτυξης της εφαρμογής δημιουργηθήκαν διαφορά είδη προβλημάτων.

Ένα πρόβλημα που παρουσιάστηκε ήταν η δημιουργία θορύβου κατά την διάρκεια της A/D μετατροπής. Αυτό το πρόβλημα όφειλωταν στην μη ύπαρξη γείωσης μεταξύ των μεταβλητών αντιστατών και των A/D μετατροπεων, και ξεπεράστηκε με την τοποθέτηση μια γείωσης. Επίσης ένα άλλο πρόβλημα που παρουσιάστηκε ήταν ότι από την έξοδο του μικροελεγκτη (Tx) η τάση του break ήταν μεγαλύτερη από 5 Volt, αυτό οφειλωταν στο ότι ο προγραμματισμός τις διαρκείας του break έγινε μέσω της portB. Βεβαία αυτό το πρόβλημα ξεπεράστηκε με την βοήθεια της διατάξεις RS/EIA-485 που το σήμα σταθμίστηκε και μετατράπηκε σε balance.

9.3 ΠΛΕΟΝΕΚΤΗΜΑΤΑ – ΤΡΟΠΟΙ ΑΝΑΠΤΥΞΗΣ ΣΥΣΚΕΥΗΣ

Με την συγκεκριμένη εφαρμογή μπορούμε να ελέξουμε μια μικρή διάταξη φωτισμού. Μεγάλο πλεονέκτημα της σύσκευης αυτής είναι το πολύ μικρό κόστος, καθώς η βασική του λειτουργία στηρίζετε στις δυνατότητες που μας δίνει ο μικροελεγκτης ATmega163 που έχει σχετικά μικρό κόστος.

Μεγάλο πλεονέκτημα της συγκεκριμένης σύσκευης επίσης είναι ότι είναι μια αρκετά ευέλικτη . Μπορούμε με μικρές αλλαγές στο software και στο hardware να την μετατρέψουμε. Για παράδειγμα θα μπορούσε ο έλεγχος να γινόταν μέσω μηνυμάτων midi.

Επίσης είναι αρκετά εύκολη η επέκταση της συγκεκριμένης σύσκευης έτσι ώστε να ελέγχουμε ακόμα μεγαλύτερες διατάξεις φωτισμού. Αυτό μπορεί να γίνει με την χρήση ενός ακόμα μικροελεγκτη.

Η συγκεκριμένη εφαρμογή μπορεί να χρησιμοποιηθεί για εκπαιδευτικούς σκοπούς λόγω της απλότητα της. Επίσης μπορεί να χρησιμοποιηθεί σαν δοκιμαστική συσκευή έτσι ώστε να διαπιστώνουμε εάν μια διάταξη φωτισμού δέχεται DMX512 πακέτα .

ΒΙΒΛΙΟΡΑΦΙΑ

1. DMX512/1992 Digital Data Transmission Standard for Dimmers and controllers, United States Institute for Theatre Technology,Inc (USSIT)

2. Adam Bennette: Recommended practice for DMX512,(Guide for users and istulers)PLASA,1994

3. Atmel AVR ATmega 163 datasheet,2003

4. Atmel AVR AT90S8515 datasheet,2000

5. AVR STK500 User Guide,Atmel Corporation,2004ation

6. Elector DMX/LPT Interface,5/2002

7. Elector 8 channels DMX de-multiplexer,12/2001

8. National semiconductors,Application note 1057, Ten ways to Bulletproof RS-485 Interfaces,10/1996

9. MIDI to DMX512 Interfacing Protocol. A thesis submitted for the degree of Master of Philosophy By Christos Chousidis

URLs

http://www.dmx512-online.com/

http://www.dmx512.com/

http://www.pangolin.com/LD2000/dmx-about.html

http://www.usitt.org/standards/DMX512.html

http://www.atmel.com/dyn/resources/prod_documents/doc1142.pdf

http://www.telematica.gr/Product/micro/stk500_gr.html

http://www.avrfreaks.net/index.php?module=Freaks%20Devices&func=displayDev&obj

http://www.datasheetcatalog.com/datasheets_pdf/D/S/4/8/DS485.shtml

http://www.national.com/mpf/DS/DS485.html

http://eshop.engineering.uiowa.edu/NI/pdfs/01/28/DS012880.pdf

ΠΑΡΑΡΤΗΜΑ

FIRMWARE

```
.INCLUDE "m163def.inc"


; Used registers
.def    strT      = r16
.def delay1       = r17
.def delay2       = r18
.def delay3       = r19
.def       chs             = r20                    ; channel select
.def HI          = r21
.def accu1        = r22
.def UART_CONTROL  = r23
.def LO          = r24
.def       Poa0       = r25              ; status of outputs
.def       Poa1      = r26
.def       Poa2      = r27
.def       Poa3      = r28
.def       Poa4      = r29
.def       Poa5      = r30
.def    Poa6      = r31




; Code starts here

; ******************* Main program ***************************


main:    ldi       r16,high(RAMEND)         ; Initiate Stackpointer.
         out       SPH,r16
         ldi       r16,255
         out       ddrb,r16           ; portB EXODOS
         clr       r16
```

```
        out     ddra,r16; portA EISODOS
        ldi     r16,$8f         ; pull-up only for inputs that
        out     porta,r16       ; are not analog inputs
        ;clr    Poa0
        ldi     r16,$D3
        out     ADCSR,r16




;*********************DATA_0************************************


LOOP:

        ldi     chs,$20; input ADC0 selected first
sadc1:  out     ADMUX,chs
        in      r16,ADCSR
        sbr     r16,(1<<6)      ; start coversion, when completed bit6 returns to 0 automatically
        out     ADCSR,r16

ldi delay3,255
DELAYADC0:
dec delay3
brne DELAYADC0

 in     Poa0,ADCH

;*********************DATA_1************************************

        ldi     chs,$21 ; input ADC1 selected first
sadc2:  out     ADMUX,chs
        in      r16,ADCSR
        sbr     r16,(1<<6)      ; start coversion, when completed bit6 returns to 0 automatically
        out     ADCSR,r16

ldi delay3,255
DELAYADC1:
dec delay3
brne DELAYADC1

        in      Poa1,ADCH
```

```
;********************DATA_2***********************************


        ldi     chs,$22 ; input ADC2 selected first
sadc3:  out     ADMUX,chs
        in      r16,ADCSR
        sbr     r16,(1<<6)      ; start coversion, when completed bit6 returns to 0 automatically
        out     ADCSR,r16



    in  Poa2,ADCH

ldi delay3,200
DELAYADC2:
dec delay3
brne DELAYADC2

;********************DATA_3***********************************


        ldi     chs,$23         ; input ADC3 selected first
sadc4:  out     ADMUX,chs
        in      r16,ADCSR
        sbr     r16,(1<<6)      ; start coversion, when completed bit6 returns to 0 automatically
        out     ADCSR,r16



        in      Poa3,ADCH

ldi delay3,200
DELAYADC3:
dec delay3
brne DELAYADC3
```

```
;*********************DATA_4************************************


        ldi     chs,$24 ; input ADC4 selected first
sadc5:  out     ADMUX,chs
        in      r16,ADCSR
        sbr     r16,(1<<6)      ; start coversion, when completed bit6 returns to 0 automatically
        out     ADCSR,r16



        in      Poa4,ADCH

ldi delay3,200
DELAYADC4:
dec delay3
brne DELAYADC4
;*********************DATA_5************************************


        ldi     chs,$25 ; input ADC5 selected first
sadc6:  out     ADMUX,chs
        in      r16,ADCSR
        sbr     r16,(1<<6)      ; start coversion, when completed bit6 returns to 0 automatically
        out     ADCSR,r16



        in      Poa5,ADCH


ldi delay3,200
DELAYADC5:
dec delay3
brne DELAYADC5
```

```
;**********DMX PACKET****************************************

ldi HI,0b11111111        ;fortonoume 2 katastaseis Hi Logia BREAK & MAB
ldi LO,0b00000000

ldi accu1,1              ;Rithmizei to boud rate 250 Kbps
out UBRR,accu1
ldi UART_CONTROL,0b00001101



ldi accu1,0
OUT UCSRB,accu1

out DDRD,HI              ;Kanei ti thira D exodo

out PORTD,LO             ;Bfazei LOW gia to BREAK

ldi delay1,234

B:                       ;Kathisterei 88microsecond gia to BREAK
  dec delay1
  brne B


out PORTD,HI             ;Bgazei to HI gaia to MAB

ldi delay2,22
MAB:                     ;Kathisterei gia to MAB 9 microsecond
dec delay2
brne MAB



ldi accu1,0b00001111     ;Energopoioume tin UART TXE=1, CHR9=1, TXB8=1
out UCSRB,accu1

out UDR,LO               ;Bgazei to Stast Code
SECOND_DATA1:            ;Elegxei an oloklirothike h apostoli apo ti UART
```

```
   sbis $0b,5
   rjmp SECOND_DATA1

out UDR,Poa0          ;Bgazei to Proto DATA BYTE Ch1
SECOND_DATA2:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA2


out UDR,Poa1          ;Bgazei to Deutero DATA BYTE Ch2
SECOND_DATA3:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA3

out UDR ,Poa2          ;Bgazei to Trito DATA BYTE Ch3
SECOND_DATA4:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA4


out UDR,Poa3          ;Bgazei to TETARTO DATA BYTE Ch4
SECOND_DATA5:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA5


out UDR,Poa4         ;Bgazei to PEMTO DATA BYTE Ch5
SECOND_DATA6:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA6



out UDR,Poa5          ;Bgazei to EKTO DATA BYTE Ch6
SECOND_DATA7:         ;Elegxei an oloklirothike h apostoli apo ti UART
   sbis $0b,5
   rjmp SECOND_DATA7



ldi delay2,250
```

```
ldi delay3,250
DELAY_PACKET:
dec delay3
brne DELAY_PACKET
dec delay2
brne DELAY_PACKET




rjmp    LOOP
```