ΑΕΙ ΠΕΙΡΑΙΑΤ.Τ.
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝΤ.Ε.

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Εκπαιδευτικό Σύστημα Εκμάθησης Ενσωματωμένων Συστημάτων

Πελέκης Μάριος

Επιβλέπων: Αν. Βελώνη
Καθηγήτρια ΑΕΙ Πειραιά

Αθήνα, Οκτώβριος 2016

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Εκπαιδευτικό Σύστημα Εκμάθησης Ενσωματωμένων Συστημάτων

Πελέκης Μάριος
Α.Μ. 42077

Επιβλέπων:

Αν. Βελώνη, Καθηγήτρια

Εξεταστική Επιτροπή:

.....
.....

Ημερομηνία εξέτασης 17/5/2015

.....

Πελέκης Μάριος

Διπλωματούχος Μηχανικών Ηλεκτρονικών Υπολογιστικών Συστημάτων

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο/Η κάτωθι υπογεγραμμένος/η ……,
του ……, με αριθμό μητρώου …….. φοιτητής/τρια του Τμήματος Μηχανικών Η/Υ Συστημάτων Τ.Ε. του Α.Ε.Ι. Πειραιά Τ.Τ. πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μου, δηλώνω ότι ενημερώθηκα για τα παρακάτω:

«Η Πτυχιακή Εργασία (Π.Ε.) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και πρωτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρει θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε., ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

Πέραν των όποιων ποινικών ευθυνών του συγγραφέα σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του Τμήματος. Η Συνέλευση του Τμήματος με νέα απόφασης της, μετά από αίτηση του ενδιαφερόμενου, του αναθέτει εκ νέου την εκπόνηση της Π.Ε. με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε. πρέπει να ολοκληρωθεί εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσης της. Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18, παρ. 5 του ισχύοντος Εσωτερικού Κανονισμού.»

ΕΥΧΑΡΙΣΤΙΕΣ

Αφιερώνω την διπλωματική αυτή εργασία, στους γονείς μου και στον αδερφό μου Ηλία.

Πρώτη από όλους θα ήθελα να ευχαριστήσω την υπεύθυνη και επιβλέπων της διπλωματικής αυτής, Κα Αναστασία Βελώνη, Καθηγήτρια Ηλεκτρολόγος Μηχανικός Ε.Μ.Π, για την αμέριστη εμπιστοσύνη που μου έδειξε και καθοδήγηση που μου πρόσφερε καθ' όλη τη διάρκεια εκτέλεσης της παρούσας πτυχιακής. Ευχαριστώ την οικογένειά μου για την αμέριστη συμπαράσταση, βοήθεια και συμβουλές που μου προσέφεραν ώστε να ολοκληρωθεί με τον καλύτερο τρόπο αυτή η πτυχιακή. Επίσης ευχαριστώ τους καθηγητές του πανεπιστημίου μου, για τις πληροφορίες που μου προσέφεραν.
Τέλος, (last but not the least) θα ήθελα να ευχαριστήσω τους φίλους μου, για την ανεκτίμητη παρέα καθ' όλη τη διάρκεια εκπόνησης της εργασίας.

ΠΕΡΙΛΗΨΗ

Σύμφωνα με τον σπουδαίο Γάλλο φιλόσοφο Jacques Ellul,
"*Η Σύγχρονη τεχνολογία έχει αποτελέσει ένα πραγματικό φαινόμενο για τον πολιτισμό, την καθοριστική δύναμη μιας νέας κοινωνικής τάξης στην οποία η αποτελεσματικότητα δεν αποτελεί επιλογή αλλά αναγκαιότητα που επιβάλλεται σε κάθε ανθρώπινη δραστηριότητα.".*

Τα λόγια αυτά φαντάζουν όλο και πιο πιστευτά όσο περνάνε τα χρόνια και οι ζωή του ανθρώπου ορίζεται και λειτουργεί μέσα σε ένα κόσμο ηλεκτρονικών συσκευών και αυτοματοποιημένων διεργασιών. Τα πάντα γύρω μας λειτουργούν με έναν προγραμματισμένο και πολύ αυτόματο τρόπο, δημιουργώντας μια κατάσταση εξάρτισης του ανθρώπου από έναν τεχνολογικό κόσμο που τείνει να επεκτείνεται σε κάθε πτυχή της ανθρώπινη ζωής. Τα θεμέλια αυτού του τεράστιου και περίπλοκου τεχνολογικού οικοδομήματος αποτελούνται από τα Ενσωματωμένα Συστήματα ή Embedded System όπως είναι ευρέως γνωστά.

Το αντικείμενο, λοιπόν, αυτής της διπλωματικής εργασίας, είναι η ανάλυση και κατανόηση των Ενσωματωμένων Συστημάτων (Embedded Systems) μέσα από μια μορφή εκπαιδευτικού χαρακτήρα. Στόχος του παρόντος συγγράμματος είναι η εμπέδωση των βασικότερων εννοιών καθώς και της γενικότερης φιλοσοφίας της σχεδίαση, προγραμματισμού και υλοποίησης ενσωματωμένων συστημάτων. Η ανάλυση αυτή, ξεδιπλώνεται μέσα από 6 διαφορετικού περιεχόμενου, πληροφοριακές ενότητες σχετικές με τον τρόπο λειτουργίας και διαχείρισης των microcontrollers με στοιχεία προγραμματισμού.

ΕΠΙΣΤΗΜΟΝΙΚΗ ΠΕΡΙΟΧΗ: Αρχιτεκτονική των Embedded System
ΛΕΞΕΙΣ ΚΛΕΙΔΙΑ: MCU, C, Assembly, Core, GPIO,RAM, Clock

ABSTRACT

According to the great French philosopher Jacques Ellul,
“Modern technology has become a total phenomenon for civilization, the defining force of a new social order in which efficiency is no longer an option but a necessity imposed on all human activity.”.
These words seems more and more believable over passing years and the human life is defined and operates in a world of electronic devices and automatic processes. Everything around us, is working in a very specific and automated way, causing a state of dependence on a technological world that tends to penetrate in every sector of human life. The foundations of this huge and complex edifice are based on the Embedded Systems.

The object of this thesis, is the analysis and understanding of Embedded Systems in a more educational form. The aim of this writing is to make the reader assimilate the basic meanings, also the general philosophy of designing, programming and implementation of Embedded System. This analysis unfolds through 6 information sections with different content, related to the operation and management of microcontrollers in combination of programming.

SCIENTIFIC AREA: Structure of Embedded Systems
KEYWORDS: MCU, C, Assembly, Core, GPIO,RAM, Clock

ΠΕΡΙΕΧΟΜΕΝΑ

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ **13**

ΕΙΣΑΓΩΓΗ **16**

Κεφάλαιο 1 – Εμβαθύνοντας στα Ενσωματωμένα Συστήματα **24**

1.1 Κεντρική Μονάδα Επεξεργασίας (CPU) *26*

1.2 Εισόδους και Εξόδους (I/O) *28*

1.3 Λογισμικό *30*

1.4 Μνήμη *33*

1.5 Επικοινωνία *36*

Κεφάλαιο 2 – Θεωρία **42**

2.1 LAB 1 – General Clock Module *43*

2.1.1 RC Ταλαντωτής 46

2.1.2 LC Ταλαντωτής 47

2.1.3 Ταλαντωτής Κρυστάλλου 48

2.2 LAB 2 – Digital Inputs Outputs (GPIO / KBI) *49*

2.2.1 GPIO (General Purpose Inputs Outputs) 49

2.2.2 KBI (Keyboard Interrupts) 52

2.3 LAB 3 – Timers (FTM / MTIM) *54*

2.3.1 FTM (FlexTimer Module) 58

2.3.2 MTIM (8-Bit Modulo Timer) 62

2.4 LAB 4 – Serial Communication (SCI / SPI) *63*

2.4.1 SCI –Serial Communication Interface (UART) 66

2.4.2 SPI - Serial Peripheral Interface 69

2.5 LAB 5 –Serial Communication and Analog to Digital converter *72*

2.5.1 I2C - (Inter-Integrated Circuit) 72

2.5.2 ADC - (Analog-to-Digital Converter) 76

2.6 LAB 6 – Touch Sense Input *78*

2.6.1 TSI (Touch Sense Input) 80

Κεφάλαιο 3 – Εφαρμογή **82**

3.1 LAB 1 – Clock Module (External Oscillator) *83*

Στη Πλακέτα 83

Στη Πράξη 84

3.2 LAB 2 – GPIO/KBI (Keyboard) *86*

Στη Πλακέτα 86

Στη Πράξη 87

3.3 LAB 3 – FTM/MTIM (PWM the LED) *89*

Στη Πλακέτα 89

Στη Πράξη 90

3.4 LAB 4 – Serial Communications (ADT7301 and RS232) *92*
Στη Πλακέτα 92
Στη Πράξη 94

3.5 LAB 5 – Serial Communications (BH1750) & A2D *96*
Στη Πλακέτα 96
Στη Πράξη 98

3.6 LAB 6 – Touch Sense Input *100*
Στη Πλακέτα 100
Στη Πράξη 101

Κεφάλαιο 4 - Build a Project **102**

Κεφάλαιο 5 - Debug **106**

5.1 Resume (Συνέχεια) *108*

5.2 Suspend (Παύση) *108*

5.3 Terminate (Τερματισμός) *108*

5.4 Disconnect (Διακοπή) *108*

5.5 Step Into (Μετάβαση Εντός) *108*

5.6 Step Over (Μετάβαση Πάνω) *109*

5.7 Step Return (Μετάβαση Πίσω) *109*

5.8 Reset (Επανατοποθέτηση) *109*

5.9 Restart (Επανεκκίνηση) *109*

ΒΙΒΛΙΟΓΡΑΦΙΑ ***110***

ΣΥΝΤΟΜΟΓΡΑΦΙΕΣ

ACK	Acknowledge Bit
ADC	Analog to Digital
ASCII	American Standard Code for Information Interchange
BUSOUT	Bus Output
CLK	Clock
CNT	Count
CS	Chip Select
FLL	Frequency-locked loop
FTM	FlexTimer Module
GPIO	General purpose Input/Output
I2C	Inter-Integrated Circuit
IC	Integrated Circuit
IRC	Internal Reference Clock
KBI	Keyboard Interrupts
KHZ	Kilo Hertz
LC	Inductor Capacitor
LED	light-emitting diode
LPC	Low Power Counter
LPM	Low Power Mode
LSB	Less Significant Byte
MCU	Microcontroller Unit
MHZ	Mega Hertz
MISO	Master In – Slave Out
MOD	Modulo
MOSI	Master Out – Slave In
MS	Millisecond
MSB	Most Significant Byte
MTIM	Modulo Timer
PWM	Pulse With Modulation
R/W	Read/Write
RAM	Random-Access Memory
RC	Resistant Capacitor
RX	Receive Chanel
SCI	Serial Communication Interface
SCL	Serial Clock
SCLK	SPI Clock
SDA	Serial Data
SPI	Serial Peripheral Interface
SS	Slave Select
TDP	Timer Digital Pin
TSI	Touch Sense Input

TX	Transceiver Channel
UART	universal asynchronous receiver/transmitter
UV	Ultraviolet
USB	Universal Serial Bus

ΕΙΣΑΓΩΓΗ

Το βιβλίο της εξέλιξης του ανθρώπου, περιλαμβάνει θα έλεγε κανείς, μεγάλα κεφάλαια στην πορεία της εξέλιξής του, κεφάλαια που άλλαξαν για πάντα την αντίληψή του για το περιβάλλον, την ύπαρξή του άλλα και τον τρόπο που σκέφτεται. Επίσης, μεγάλο ρόλο σε αυτή την εξέλιξη έπαιξαν οι ανά τα χρόνια ανακαλύψεις που έφερναν σιγά-σιγά τον άνθρωπο όλο και πιο κοντά σε αυτό που είναι σήμερα.

Μια από της σπουδαιότερες ανακαλύψεις της ανθρωπότητας είναι η ανακάλυψη του ηλεκτρικού φαινομένου. Με ρίζες από το 600π.χ. όταν ο Θαλής Μιλίσιος πρώτο-παρατήρησε το φαινόμενο του στατικού ηλεκτρισμού, έως και τους πατέρες του ηλεκτρισμού όπως ο Βιεναμίν Φραγκλίνος, Αλεσάντρο Βόλτα, Χανς Κρίστιαν Έρστεντ, Αντρέ Μαρί Αμπέρ και Μάικλ Φαραντέι, το ηλεκτρικό φαινόμενο αποτελεί μια από τις μεγαλύτερες ανακαλύψεις, ίσως και την μεγαλύτερη, που έκανε ποτέ ο άνθρωπος.

Η συνεχής τριβή με τον ηλεκτρισμό έφερε στην επιφάνεια μερικά από τα πιο γνωστά και συνήθη χαρακτηριστικά όπως τα γνωρίζουμε σήμερα. Επίσης με το πέρασμα των χρόνων, ο ηλεκτρισμός, τα γνωρίσματα και οι ιδιότητές του αναλύθηκαν από τον άνθρωπο σε επίπεδο τέτοιο, που η χρήση του σε καθημερινή βάση με σκοπό την βελτίωση της ζωής του ανθρώπου, καθιέρωσαν την ανακάλυψη του φαινομένου αυτού, ως το ηλεκτρικό ρεύμα όπως ακριβώς το γνωρίζουμε σήμερα και που έχουμε στα σπίτια μας.

Η ανακάλυψη του ηλεκτρικού ρεύματος αποτέλεσε το έναυσμα για μια σειρά μεγάλων τεχνολογικών εφευρέσεων που θα βοήθαγαν την ανθρωπότητα να πραγματοποιήσει τεράστια τεχνολογικά και επιστημονικά επιτεύγματα και να ξεπεράσει κατά πολύ τον έως τότε εαυτό της. Το στοίχημα που έπρεπε να πετύχει ο άνθρωπος, ήταν να μπορέσει με κάποιο τρόπο να δαμάσει το ηλεκτρικό ρεύμα και τις ιδιότητές του, ώστε να μπορεί να το διαχειριστεί προς όφελος του.

Το δρόμο για αυτό το επίτευγμα, ανοίγει ο Γιούλιους Έντγκαρ Λιλινφελντ το 1928 και ακολουθεί ο Γερμανός φυσικός Όσκαρ Χάιλ, που το 1934 κατοχυρώνει για πρώτη φορά στην ιστορία, την ευρεσιτεχνία του Τρανζίστορ (Transistor). Το Τρανζίστορ αποτελούσε μια τρίοδο, σχηματισμένη από

ημιαγωγή φυσικά στοιχεία, η οποία είχε την δυνατότητα να ελέγχει την διέλευση ρεύματος στα δύο από τα τρία της άκρα σύμφωνα με το φορτίο ρεύματος που εμφανιζόταν στο τρίτο άκρο της τριόδου.

Η τελειοποίηση αυτού του ηλεκτρονικού εξαρτήματος (τρανζίστορ) φέρνει τον άνθρωπο αντιμέτωπο με το επόμενο μεγάλο εμπόδιο που καλείτε να ξεπεράσει. Αυτό δεν είναι άλλο από την οργάνωση πολλών τέτοιων τρανζίστορ στο χώρο, με τρόπο τέτοιο, ώστε να δημιουργούνται συγκεκριμένες διατάξεις για την κυκλοφορία του ρεύματος με πολύ προκαθορισμένο και “προγραμματισμένο” τρόπο. Για να γίνει πιο κατανοητό το παραπάνω, θα μπορούσαμε να συγκρίνουμε την διαδικασία αυτή με το κυκλοφοριακό σύστημα που συναντάμε κάθε μέρα στις πόλεις μας. Δηλαδή, ένα σύστημα που αποτελείτε από αυτόματους ελεγκτές διέλευσης κυκλοφορίας όπως είναι τα φανάρια, ώστε να διατηρείτε η ομαλότητα ανάμεσα σε πεζούς, ποδηλάτες και αυτοκίνητα.

Την ιδέα αυτή, έρχεται να εφαρμόσει στην πράξη το Πανεπιστήμιο του MIT (Massachusetts Institute of Technology), όπου το 1956 κατασκευάζει το πρώτο ηλεκτρονικό υπολογιστή, ονόματι TX-0, ο οποίος λειτουργεί με Τρανζίστορ. Αξίζει σε αυτό το σημείο να αναφερθεί πως ο TX-0 δεν αποτελεί τον πρώτο υπολογιστή που κατασκευάστηκε ποτέ, καθώς και πριν από τον TX-0 βλέπουμε υλοποιήσεις υπολογιστικών συστημάτων, όπως το “ENIAC” (Electronic Numerical Integrator and Computer) και το “Colossus Mark I”. Τα δύο αυτά μοντέλα δημιουργούνται στο Πανεπιστήμιο της Πενσηλβάνιας με κρατικές επιχορηγήσεις για αντικατασκοπευτικούς λόγους. Σε αντίθεση με τον “TX-0”, οι “ENIAC” και ”Colosus Mark I” δεν αποτελούν ηλεκτρονικά υπολογιστικά συστήματα καθώς αποτελούνται από ηλεκτρικούς λαμπτήρες και όχι από ηλεκτρονικά Τρανζίστορ.

Στη συνέχεια, ο Τζακ Κίλμπυ (Jack Kilby) ένας υπάλληλος της Texas Instruments, μια εταιρίας κολοσσό του σήμερα, έρχεται να “ταράξει τα νερά” για τα καλά στον έως τότε διαμορφωμένο χώρο, δημιουργώντας κάτι που θα άλλαζε για πάντα τον κόσμο των ηλεκτρονικών. Το 1958 ο Jack Kilby γίνεται ο πρώτος άνθρωπος που κατασκεύασε Ολοκληρωμένο Σύστημα με Τρανζίστορ, αντιστάσεις, πυκνωτές και άλλα ηλεκτρονικά εξαρτήματα ενσωματώνοντάς όλα αυτά μαζί σε ένα μόνο κομμάτι πυριτίου. Το δημιούργημα αυτό, ανοίγει τον ορίζοντα για δημιουργία μικρών υπολογιστικών

μονάδων, που έδωσε την δυνατότητα στους επιστήμονες να χρησιμοποιήσουν αυτή την μικρή σε φυσικές διαστάσεις υπολογιστική δύναμη, σε πληθώρα εφαρμογών.

Με το τελευταίο αυτό τεχνολογικό επίτευγμα του ανθρώπου, ανοίγει ένας νέος δρόμος για την τεχνολογία των υπολογιστών, εκείνος των Embedded Systems. Δηλαδή ένα σύνολο από ηλεκτρονικά υλικά και εξαρτήματα που όλα μαζί συμβάλουν στη δημιουργία ενός σύνθετου συστήματος αποτελούμενο από ηλεκτρικό και μηχανικό μέρος. Το παρών σύστημα πλεον, έχει την δυνατότητα να εκτελεί ενέργειες με αυτόματο και πολύ προκαθορισμένο τρόπο, επίσης μπορεί να επεξεργάζεται διάφορα ερεθίσματα από το περιβάλλον καθώς και να επικοινωνεί με άλλα συστήματα. Τα συστήματα αυτά είναι οι σύγχρονοι Microprocessor (μικροεπεξεργαστές) και Microcontroller (μικροελεγκτές), που συναντάμε σε οποιαδήποτε ηλεκτρονική συσκευή υπάρχει γύρω μας.

Οι microcontroller και οι microprocessors, που σαν συστήματα αναφέρονται ως Embedded Systems, αποτελούν μικρές ηλεκτρονικές υπολογιστικές μονάδες, στις οποίες ο άνθρωπος έχει την δυνατότητα να ορίσει την λογική με την οποία θα δουλεύουν και τον τρόπο με τον οποίο θα αλληλεπιδρούν με το περιβάλλον τους. Η παραπάνω διαδικασία με μια λέξη ονομάζεται “προγραμματισμός”.

Σε αυτό το σημείο θα πρέπει να διευκρινιστεί η έννοια των λέξεων microcontroller και microprocessor καθώς και η βασική διαφορά των δύο αυτών συστημάτων.

Ο microprocessor ο οποίος ιστορικά εμφανίζεται και πρώτος, αποτελεί ένα ηλεκτρονικό σύστημα το οποίο χαρακτηρίζεται από ένα κεντρικό ρολόι, διάφορους καταχωρητές που με αυτούς ο χρήστης μπορεί να προγραμματίσει τον τρόπο λειτουργίας του και ένα πυρήνα ο οποίος έχει την δυνατότητα να εκτελεί πράξεις. Οι microprocessors αποτελούν την μικρότερη και απλούστερη μονάδα Embedded System καθώς εκτός από ελάχιστες δυνατότητες που παρέχουν στον χρήστη, πραγματοποιούν μόνο πράξεις στο εσωτερικό του πυρήνα τους.

Ο microcontroller ωστόσο αποτελείται και αυτός από κάθε χαρακτηριστικό ενός microprocessor, όπως αναφέρθηκε παραπάνω, ενώ παράλληλα εμπεριέχει πληθώρα ανεξάρτητων μονάδων, όπου κάθε μία από αυτές κάνει κάτι το διαφορετικό, όπως παραδείγματος χάρη το πρωτόκολλο επικοινωνίας I2S.

Η βασική διαφορά ανάμεσα σε αυτά τα δύο συστήματα είναι οι δυνατότητες που τα ορίζουν. Οι microprocessors αποτελούνται μόνο από υπολογιστική δύναμη για το λόγο αυτό, χρησιμοποιούνται σε εφαρμογές που απαιτείτε μόνο ταχύτητα εκτέλεσης πράξεων και εντολών, από την άλλη πλευρά οι microcontroller εφαρμόζονται σε περιπτώσεις που απαιτείται ολοκληρωμένη λύση αυτόματης διαχείρισης και ευελιξίας. Ένα εύλογο ερώτημα που γενάτε όμως είναι το γιατί χρησιμοποιούμε ακόμα και microcontrollers και microprocessors, καθώς σύμφωνα με τα παραπάνω, ένας microcontroller καλύπτει και τις δύο ανάγκες.

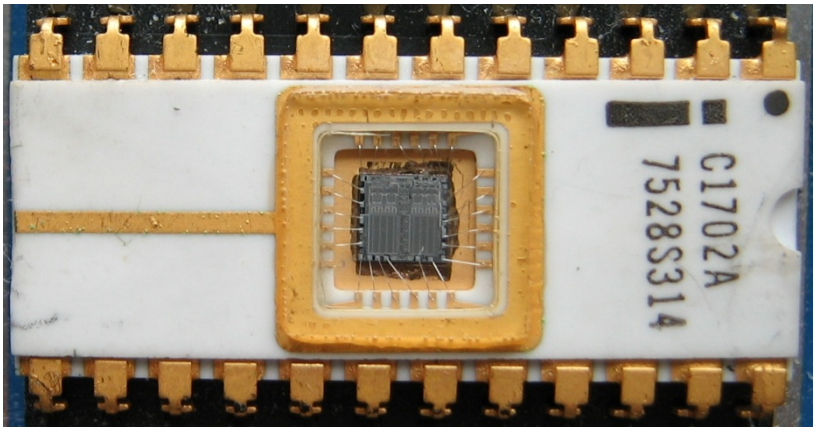
Λάθος! Τα πάντα είναι θέμα χώρου. Ένας microcontroller δεν θα μπορέσει ποτέ να επιτύχει την υπολογιστική δύναμη ενός microprocessor ενώ παράλληλα να διατηρεί όλα εκείνα τα πολύ σημαντικά χαρακτηριστικά που τον κάνουν να ξεχωρίζει, σε χώρο ίσο με εκείνο ενός microprocessor. Συνεπώς κατανοούμε πως κάθε σύστημα τείνει να τελειοποιηθεί στον τομέα για τον οποίο προορίζεται.

Ιστορικά το πρώτο ολοκληρωμένο Embedded System που εμφανίζεται με γνωρίσματα, αρκετά πλησίον σε αυτά που αναφέρθηκαν παραπάνω κατασκευάζεται το 1971 από μια εταιρεία κολοσσό. Πρωτοπόρα στο τομέα της τεχνολογίας η INTEL παρουσιάζει τον πρώτο IC(Integrated Circuit) microprocessor, ο οποίος στηρίζεται 4-Bit αρχιτεκτονική. Η αρχιτεκτονική αυτή επέτρεπε στον microprocessor να εκτελεί λογικές πράξεις ταυτόχρονα σε 4-Bit σε κάθε χτύπημα του κεντρικού του ρολογιού. Τον αμέσως επόμενο χρόνο, η εταιρεία παρουσιάζει το νέο της μοντέλο “8008” το οποίο στηρίζεται σε 8-bit αρχιτεκτονική. Το επίτευγμα αυτό αποτέλεσε το έναυσμα για την υλοποίηση του ψηφιακού κόσμου όπως τον γνωρίζουμε σήμερα, και ακολουθήθηκε από πληθώρα εγχειρημάτων και από άλλες εταιρείες.

Τρια(3) χρόνια έπειτα από την δημιουργία του πρώτου microprocessor, είναι η ώρα του πρώτου microcontroller να εμφανιστεί. Για ακόμη μια φορά η εταιρεία που πρωτοστάτησε στον χώρο των ημιαγωγών, παρουσιάζει τον

πρώτο microcontroller. Η Texas Instruments παρουσιάζει το 1974 το “TMS1000”, ένα microcontroller που συνδύαζε 1 read-only χώρο αποθήκευσης, 1 read-only read/write χώρο αποθήκευσης, επεξεργαστή καθώς και δικό του κεντρικό ρολόι.

Αξίζει να σημειωθεί πως οι πρώτοι microcontrollers και microprocessors προγραμματίζονταν με έναν πολύ ιδιαίτερο τρόπο. Το κέλυφος από το ολοκληρωμένο ήταν διάφανο σε τέτοιο σημείο που ακόμα και το ανθρώπινο μάτι μπορούσε να δει το εσωτερικό του controller. Η διαγραφή των δεδομένων καθώς και ο προγραμματισμός γίνονταν με ειδικές λάμπες που εξέπεμπαν UV(Ultraviolet) ή αλλιώς υπεριώδη ακτινοβολία στο εσωτερικό του microcontroller. Χάρη στην παραπάνω τεχνική ακόμα και σήμερα θα ακούσουμε τον προγραμματισμό ενός microcontroller να αναφέρεται και ως “κάψιμο”.



Η πάροδος των χρόνων λειτούργησε πολύ ευεργετικά στον συγκεκριμένο τομέα, καθώς κάθε νέα ανακάλυψη που γινόταν, ερχόταν να μεγαλώσει εκθετικά τις γνώσεις και τις δυνατότητες της συγκεκριμένης επιστήμης. Έτσι λοιπόν εμφανίστηκαν microprocessor και συνεπώς microcontroller, με 12-Bit, 32-Bit και 64-bit αρχιτεκτονική που συνεχώς εξελίσσονται ακόμα και σήμερα. Ενδεικτικό της κατάστασης είναι πως ακόμα και σήμερα, οι μεγαλύτερες εταιρείες παρουσιάζουν συνεχώς καινούργιους microcontrollers 8-Bit αρχιτεκτονικής, ακόμα κι αν η τεχνολογία έχει ήδη καταφέρει microcontrollers μεγαλύτερης ισχύς. Αυτό γίνεται γιατί, όπως αναφέρθηκε και προηγουμένως, κάθε κατηγορία embedded system κάνει κάτι το διαφορετικό και είναι απαραίτητη.

Κλείνοντας αυτήν την ιστορική αναδρομή, θα ήθελα να αναφέρω πως μεγάλο ρόλο στην εξέλιξη των microprocessors και microcontrollers έπαιξαν κι άλλες πολύ μεγάλες εταιρίες, που δεν αναφέρθηκαν καθότι στην παρούσα εισαγωγή έγινε παρουσίαση της προϊστορίας των embedded system, με όσο το δυνατό πιο λακωνικό τρόπο. Εταιρίες όπως η Motorola, National Semiconductors, Freescale, Renesas, NXP, Infineon, Silicon Labs, Microchip, Atmel και πολλές άλλες προσέφεραν πάρα πολλά στην ανακάλυψη νέων τεχνικών και τεχνολογιών στον τομέα των Embedded System. Διαδραματίζοντας τεράστιο ρόλο στον παγκόσμιο χάρτη της τεχνολογικής εξέλιξης των ημιαγωγών, κάθε μια από αυτές τις εταιρείες κολοσσούς άφησαν και αφήνουν ακόμα και σήμερα το στίγμα τους με μερικές από τις σπουδαιότερες ανακαλύψεις σε αυτόν τον κλάδο.

Όπως αναφέρθηκε νωρίτερα, κάθε εταιρία του κλάδου διαφέρει από τις υπόλοιπες, καθώς δίνει περισσότερη σημασία σε κάποιους συγκεκριμένους τομείς ενώ κάποια άλλη σε κάποιους άλλους τομείς. Επίσης καθώς αναφερόμαστε σε εταιρείες πρωτοπόρους, που κατέχουν τεχνολογική και επιστημονική ισχύ να διαμορφώνουν τον κόσμο με την ανακάλυψη νέων τεχνικών και ισχυρότερων επεξεργαστών, γίνεται εύκολα αντιληπτή η ανάγκη για διαφορετικότητα. Συνεπώς, η διαφορετικότητα έγκειται τόσο στην τεχνογνωσία όσο και στην γενικότερη φιλοσοφία που η κάθε εταιρία έχει.

Η επιλογή λοιπόν, microcontroller συνεπώς και της εταιρίας που τον παρέχει, με τον οποίο θα ασχοληθεί το παρόν σύγγραμμα είναι πολύ προσεχτικά μελετημένη. Για να γίνω πιο συγκεκριμένος, ο microcontroller με τον οποίο θα ασχοληθούμε είναι από την 8-Bit σειρά S08 της εταιρίας Freescale και πιο συγκεκριμένα είναι ο MC9S08PT60.

Λίγα λόγια για την εταιρία

Η Freescale/NXP αποτελεί κυρίαρχη δύναμη στον χώρο των ημιαγωγών εδώ και πολλά χρόνια. Παρέχοντας ένα πολύ μεγάλο εύρος ημιαγωγών στο καταναλωτικό κοινό, έχει εδραίωση την θέση τις σε όλες τι κατηγορίες επεξεργαστών που διαθέτει (8/12/16/32/64bit). Η εταιρία έχει χαρακτηριστεί μοναδικός κυρίαρχος στο κλάδο των αυτοματισμών (automotive) και παρέχει την μεγαλύτερη γκάμα επεξεργαστών ισχύος (power architecture). Επίσης είναι η εταιρία που εφηύρε το σειριακό πρωτόκολλο επικοινωνίας “SPI” (Serial Peripheral Interface). Η Freescale/NXP μένει σταθερά προσηλωμένη στους στόχους της όλα αυτά τα χρόνια πρωτοπορώντας πάντα στο κλάδο του automotive και διατηρώντας μια ενιαία φιλοσοφία σε όλα της τα προϊόντα, γεγονός που βοηθάει υπερβολικά στην μετάβαση γνώσεων και τεχνογνωσίας μεταξύ των διαφόρων κατηγοριών των προϊόντων της.

Λίγα λόγια για τον microcontroller

Ο επιλεγμένος microcontroller (MC9S08PT60) αποτελεί ιδιαίτερο “στοιχείο” της σειράς S08 επεξεργαστών της Freescale/NXP, καθώς συνδυάζει πληθώρα περιφερειακών μονάδων, με την παράλληλη ενσωμάτωση 15 καναλιών αφής, γεγονός που αποτελεί σπάνιο γνώρισμα των microcontroller, τόσο στον ανταγωνισμό όσο και στην ίδια την εταιρεία. Ο συγκεκριμένος microcontroller υποστηρίζει ρολόι έως και 20MHz, αποτελείτε από 60KB μνήμη Flash 4096KB RAM και 256B EEPROM. Επίσης εμπεριέχει το 99% των περιφερειακών μονάδων που συναντάται στους microcontrollers αυτής της κατηγορίας (8-Bit), όπως 1xADC(16ch), 3xFTM, 1xI2C, 2xMTIM,1xRTC(Counter), 3xSCI(UART), 1xSPI(8Bit), 1xSPI(16Bit), 1xTSI(16ch),1xACMP, 1xCRC.

Συνεπώς αποτελεί ιδανική επιλογή για την καλύτερη κατανόηση των περιφερειακών μονάδων που εμπεριέχουν οι microcontrollers καθώς και της φιλοσοφίας των Embedded Systems και τον προγραμματισμό τους.

Κεφάλαιο 1 – Εμβαθύνοντας στα Ενσωματωμένα Συστήματα

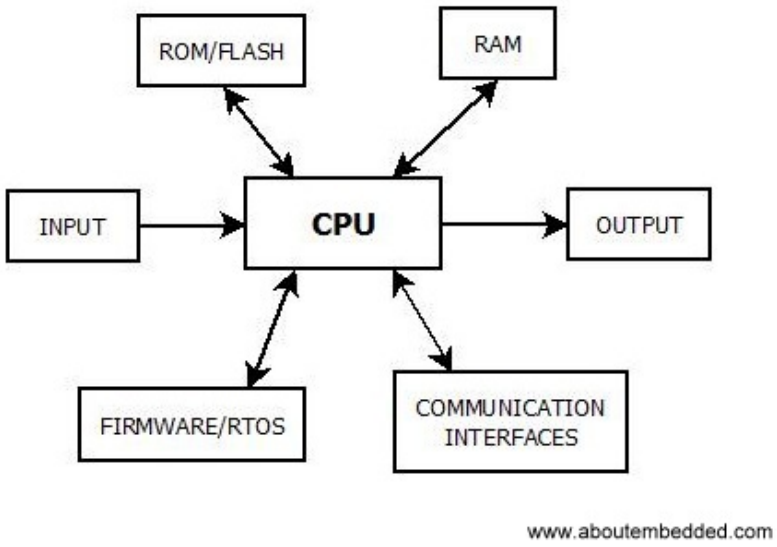
Αρχικά θα πρέπει να γίνει κατανοητός ο όρος Ενσωματωμένα Συστήματα, που στο εξής θα αναφέρονται ως Embedded Systems σύμφωνα με την διεθνή ορολογία. Ομολογούμενος ο συγκεκριμένος όρος είναι αρκετά δύσκολο να αποδοθεί σε μία πρόταση. Αυτό συμβαίνει διότι ο όρος έχει δανειστεί σε διάφορες εφαρμογές με αποτέλεσμα να επικρατεί μια σύγχυση σχετικά με την πραγματική έννοια.

Η βασική έννοια του όρου σημαίνει, ένα σύνολο ηλεκτρικών και μηχανικών στοιχείων ενωμένα σε ένα μεγάλο σύνολο με στόχο την επίτευξη ενός συγκεκριμένου στόχο στα πλαίσια προκαθορισμένων κανόνων. Ένα σύνολο, στο οποίο όλα τα επιμέρους στοιχεία που το αποτελούν, εργάζονται και συνεργάζονται, βάση ενός προκαθορισμένου σχεδίου. Το σύνολο αυτό μπορεί να αποτελεί από μόνο του μια μονάδα με καθορισμένη λειτουργικότητα καθώς επίσης μπορεί να αποτελεί στοιχείο ενός μεγαλύτερου συνόλου. Επιπροσθέτως η έννοια Embedded προσδίδει στον όρο με τον οποίο ασχολούμαστε, μια επιπλέον, ιδιαίτερης βαρύτητας σημασία. Η έννοια Embedded περιγράφει ένα σύστημα από ηλεκτρικά και μηχανικά μέρη το οποίο βασίζεται σε έναν microcontroller. Με τον microcontroller λοιπόν, το παρών σύστημα έχει την δυνατότητα να δέχεται Software το οποίο εκτελείτε σε πραγματικό χρόνο και ορίζει τις διεργασίες αλλά και τον τρόπο με τον οποίο θα εκτελεστούν. Η ενσωμάτωση λοιπόν τμήματος Software σε ένα σύνολο Hardware με στόχο την εκτέλεση περίπλοκων και σύνθετων ενεργειών σε πραγματικό χρόνο ονομάζεται με δύο λέξεις Embedded Systems.

Για να χαρακτηριστεί ένα ηλεκτρικό κύκλωμα ως Embedded System θα πρέπει το Hardware τμήμα του να αποτελείτε απαραιτήτως από:

- Κεντρική μονάδα επεξεργασίας (CPU)
- Εισόδους και Εξόδους (Inputs/Outputs)
- Λογισμικό (Software)
- Μνήμη (ROM/RAM)
- Δυνατότητα Επικοινωνίας (Communication Interface)

Το ποιο απλό παράδειγμα Embedded System παρουσιάζεται με το παρακάτω διάγραμμα.



Κάθε τομέας από τους οποίους αποτελείται ένα Embedded System όπως αναφέρθηκαν παραπάνω παίζει τον δικό του καθοριστικό ρόλο στην υλοποίηση, αποτελεσματικότητα του συστήματος. Η κάθε μονάδα, έχει συγκεκριμένες ευθύνες που δεν συγχέονται με εκείνες των άλλων μονάδων. Παράλληλα όμως, κάθε μονάδα εξαρτάται από τις άλλες για τη σωστή λειτουργία του συστήματος. Παρακάτω θα αναλύσουμε τα πέντε(5) τμήματα από τα οποία αποτελούνται τα Embedded Systems εις βάθος.

1.1Κεντρική Μονάδα Επεξεργασίας (CPU)

Η μονάδα αυτή αποτελεί αναπόσπαστο κομμάτι της φιλοσοφίας των Embedded Systems καθώς αποτελεί κατά βάση των συνδετικό κρίκο ανάμεσα στις υπόλοιπες μονάδες από τις οποίες αποτελείται ένα τέτοιο σύστημα. Ο ρόλος του είναι πολυδιάστατος και άκρος σημαντικός καθώς αποτελεί των πυρήνα του συστήματος. Η Κεντρική μονάδα επεξεργασίας, που στο εξής θα αναφέρεται ως CPU (Central Processor Unit), αποτελεί τον “εγκέφαλο” του συστήματός μας, αποτελώντας καθοριστικό ρόλος παρόμοιο με εκείνον στον ανθρώπινο οργανισμό. Ουσιαστικά, αποτελεί το σημείο του συστήματος, στο οποίο πραγματοποιούνται όλες οι επικοινωνίες ανάμεσα στις υπόλοιπες μονάδες, εκτελούνται λογικές αριθμητικές πράξεις, καθώς και γίνεται λήψη αποφάσεων συμφώνα με τις οποίες το σύστημα πραγματοποιεί ενέργειες, που ορίζονται από το πρόγραμμα(Λογισμικό) το οποίο ο ίδιος εκτελεί.

Πιο συγκεκριμένα η CPU στα Embedded Systems είναι υπεύθυνη για τα εξής:

- ***Διασύνδεση μονάδων*** με νοητό τρόπο, βάση του οποίου, ένα τμήμα του συστήματος στέλνει πληροφορίες σε ένα άλλος υπό την εποπτεία και εξουσιοδότηση της κεντρικής μονάδας επεξεργασίας. Η CPU, συνδέεται με όλες τις υπόλοιπες μονάδες και παρέχει ένα σύστημα συγχρονισμού σύμφωνα με το οποίο κάθε τμήμα του συστήματος πραγματοποιεί ενέργειες με πολύ συγκεκριμένο ρυθμό (συχνότητα) . Επίσης υπεύθυνος για την εκτέλεση μιας ενέργειας σε μια από τις μονάδες είναι η CPU και συνεπώς υπό αυτήν την έννοια αποτελεί τον “τροχονόμο” του συστήματός μας.

- ***Εκτέλεση Λογισμικού (Software):*** Με την φράση αυτή εννοούμε πως η CPU είναι σε θέση να πραγματοποιεί

ένα σύνολο ενεργειών και να ακολουθεί μια φιλοσοφία λειτουργίας σύμφωνα με το Software το οποίο δύναται να εκτελεί στο εσωτερικό του. Πιο συγκεκριμένα στα σύγχρονα Embedded Systems ο χρήστης έχει την δυνατότητα να επηρεάσει και να διαμορφώσει τον τρόπο λειτουργίας του συστήματος μέσα από την δημιουργία ενός λογισμικού. Το λογισμικό, που στο εξής θα αναφέρεται ως Software, επεμβαίνει με δύο τρόπους στην ροή λειτουργίας του συστήματος. Πρώτον, επηρεάζει και παραμετροποιεί τον Hardware μηχανισμό του συστήματος (CPU Frequency, Interrupts, Προτεραιότητα μονάδων κ.α.) μέσα από ειδικούς Register που υπάρχουν στο εσωτερικό της CPU. Δεύτερον, δημιουργεί ένα σενάριο δομημένων αποφάσεων και ενεργειών, που εκτελούνται με συγκεκριμένη σειρά και χαρακτηρίζουν την λειτουργία του συστήματος.

- ***Επεξεργασία Πληροφοριών:*** Δηλαδή τον μηχανισμό συγκέντρωσης πληροφοριών και εντολών από τις υπόλοιπες μονάδες του συστήματος και η παράλληλη λήψη αποφάσεων σύμφωνα με τις πληροφορίες που συγκεντρώθηκαν και της υποδείξεις του Software. Καθοριστικό ρόλο σε αυτό το στάδιο επεξεργασίας παίζει η Αριθμητική Λογική Μονάδα ή αλλιώς, ALU(Arithmetic Logic Unit), η οποία πραγματοποιεί, όπως αναφέρει και το όνομά της, λογικές αριθμητικές πράξεις. Με τη χρήση αυτού του μηχανισμού η CPU έχει την δυνατότητα να πραγματοποιεί λογικές πράξεις όπως η AND, OR, NAND, NOR, XOR, XNOR που συμβάλουν στην λήψη αποφάσεων.

1.2 Εισόδους και Εξόδους (I/O)

Η μονάδα αυτή αποτελεί με μια φράση, το φυσικό μέσω αλληλεπίδρασης του συστήματος με το εξωτερικό περιβάλλον. Αν κοιτάξουμε της ηλεκτρονικές συσκευές γύρω μας, μπορούμε εύκολα να καταλάβουμε την σημασία αλληλεπίδρασης των ηλεκτρονικών και ειδικότερα των Embedded System με το ευρύτερο περιβάλλον στο οποίο βρίσκονται . Ας αναλογιστούμε για παράδειγμα την νόημα θα είχε το ηλεκτρονικός υπολογιστής τσέπης(κομπιουτεράκι) αν δεν είχε κουμπιά ή αν ένα ηλεκτρονικό χρονόμετρο δεν είχε ένδειξη λήξης χρόνου. Γίνεται αντιληπτό λοιπόν, πως ένα Embedded System δεν έχει λόγο ύπαρξης εάν δεν έχει την δυνατότητα να αλληλεπιδρά με το εξωτερικό περιβάλλον. Σε αυτό, βοηθούν η γενικής χρήσης είσοδοι και έξοδοι που έχει κάθε Embedded System, ώστε το σύστημα να αναγνωρίζει αλλαγές του περιβάλλοντος μέσω των εισόδων και να το επηρεάζει μέσω των εξόδων.

Η μονάδα αυτή ενεργεί καθ΄υπόδειξη της Κεντρικής Μονάδας Επεξεργασίας του συστήματος και μεταφέρει πληροφορίες από το περιβάλλον στο σύστημα καθώς και πραγματοποιεί επιθυμητές ενέργειες του συστήματος επηρεάζοντας τον εξωτερικό “κόσμο”.

Η υλοποίηση αυτού του μοντέλου αλληλεπίδρασης χρήζει αξιοποίηση, πολλών διαφορετικών τεχνικών και πληθώρα πρωτοκόλλων επικοινωνίας με τους μηχανισμούς που το ορίζουν. Συνεπώς, ανεξαρτήτως τρόπου επικοινωνίας με τους μηχανισμούς εισόδων και εξόδων τους οποίους χρησιμοποιεί το σύστημα, η μονάδα αυτή είναι εντελώς διαφορετική από την *μονάδα Επικοινωνίας* που θα συναντήσουμε αργότερα.

Ο όρος Είσοδοι και Έξοδοι (I/O) δεν θα πρέπει να συγχέεται με τα GPIO (General Propose Inputs Outputs) των microcontroller, καθώς στο σημείο αυτό αναφερόμαστε, σε κάθε μηχανισμό που ενσωματώνεται στο συστήματος και το βοηθάει να αλληλεπιδρά με το περιβάλλον του και όχι στην ύπαρξη γενικής χρήσης εισόδων εξόδων που συναντάμε στου microcontroller, λογικής 0/1.

Για να γίνω πιο κατανοητός θα αναφέρω μερικούς από τους πιο συνηθισμένους μηχανισμούς Εισόδων Εξόδων που συναντάμε σε κάποιες εφαρμογές των Embedded Systems.

Είσοδοι

- Πληκτρολόγιο
- Μικρόφωνο
- Κουμπιά
- Αισθητήρες Θερμοκρασίας, Κίνησης, Υγρασίας κ.α.
- Αναλογικές είσοδοι (περιστροφική διακόπτες, Ποτενσιόμετρα)

Έξοδοι

- Λυχνίες ένδειξης
- Μεγάφωνο, Buzzer
- Οθόνη προβολής πληροφοριών
- Θέρμανση, Ψύξη

1.3 Λογισμικό

Το λογισμικό στα Embedded Systems, που αναφέρεται κοινώς ως Embedded Software, αποτελεί μια πολύ σημαντική μονάδα του συστήματος. Συνοπτικά, το λογισμικό αποτελεί το σημείο αναφοράς για κάθε ενέργεια που εκτελεί το σύστημα καθώς και απόφαση που καλείτε να πάρει.

Πιο συγκεκριμένα το λογισμικό αποτελεί το αποτέλεσμα σύνταξης ενός κείμενου από κανόνες, περιορισμούς και υποθέσεις, που έχει γράψει ένας άνθρωπος, το οποίο έπειτα από μια συγκεκριμένη διαδικασία εισάγεται στην μνήμη του συστήματος μας. Από το σημείο αυτό και έπειτα το σύστημα λειτουργεί σύμφωνα με τον τρόπο που ορίζει το λογισμικό που είναι αποθηκευμένο στη μονάδα μνήμη.

Το λογισμικό που απευθύνεται σε Embedded Systems είναι απόλυτα συγκεκριμένο και προσδιορισμένο για το Hardware στο οποίο προορίζεται να εκτελεστεί καθώς το τελευταίο ορίζεται από απόλυτα συγκεκριμένους χρόνους(χρόνος εκτέλεσης εντολών), χώρους(χώρος αποθήκευσης-μνήμη) και ιδιότητες. Ειδικότερα το τελευταίο αποτελεί και την κατ΄ουσίαν αδυναμία συμβατότητας ανάμεσα στα διάφορα λογισμικά και στα hardware. Κάθε σύστημα χαρακτηρίζεται από διαφορετικό Hardware το οποίο χρήζει διαφορετικής μεταχείριση, όπως θα δούμε και παρακάτω, συνεπώς, απαιτείτε απόλυτη συμβατότητα ανάμεσα σε Software και Hardware στον επίπεδο των Embedded Systems.

Θα μπορούσαμε, σε αυτό το σημείο, να διαχωρίσουμε τον τρόπο με τον οποίο το λογισμικό επεμβαίνει και επηρεάζει την λειτουργία του συστήματός μας σε δύο κατηγορίες. Όπως προαναφέρθηκε το λογισμικό που απευθύνεται στα Embedded Systems επεμβαίνει με δύο τρόπους στο σύστημα.

- **Ρυθμίζοντας** τον τρόπο με τον οποίο **το Hardware** του συστήματος θα λειτουργεί. Σε αυτή την περίπτωση ο χρήστης μέσω του Software έχει την δυνατότητα να ρυθμίσει πολύ συγκεκριμένα πράγματα τα οποία θα ορίζουν τον τρόπο με τον οποίο όλοι οι εσωτερικοί μηχανισμοί θα λειτουργούν και θα συνεργάζονται μεταξύ τους. Αυτό συνήθως γίνεται μέσω ειδικών παραμέτρων που υπάρχουν στη μονάδα CPU και ονομάζονται *Registers,* οι οποίοι δέχονται πού συγκεκριμένες τιμές που είναι άρρηκτα συνδεδεμένες με το Hardware τμήμα του συστήματος. Ενδεικτικά, μερικές από τις ρυθμίσει που δύναται να επηρεάσουν οι Registers είναι οι, επιλογή τρόπου επικοινωνίας με τις περιφερικές μονάδες, επιλογή συχνότητας συγχρονισμού επικοινωνίας ανάμεσα στις μονάδες, επιλογή χρήσης διακοπτόμενων λειτουργιών(interrupts) και πολλά άλλα.

- **Ορίζοντας τον τρόπο λειτουργίας** του συστήματος. Εδώ αναφερόμαστε για την γενικότερη φιλοσοφία εκτέλεσης του προγράμματος που σε αντίθεση με πριν δεν είναι αλληλένδετη με το Hardware του συστήματος. Ουσιαστικά αναφερόμαστε σε διαδικασίες τις οποίες θέλουμε να εκτελέσει το σύστημά μας με μια προκαθορισμένη σειρά καθώς και στην δημιουργία σεναρίων αντιμετώπισης στα ερεθίσματα και στα αποτελέσματα των παραπάνω διαδικασιών. Στο σημείο αυτό ο χρήστη συμφώνα με την γλώσσα προγραμματισμού που έχει επιλέξει δομεί αυτό που λέμε “πρόγραμμα” αξιοποιώντας με τον καλύτερο δυνατό τρόπο την λογική τον εντολών που του παρέχονται.

Το πρώτο βήμα στην διαδικασία ενσωμάτωσης του Software σε ένα Embedded System είναι η σύνταξη του κειμένου(λογισμικό) σε ηλεκτρονική μορφή με την βοήθεια κάποιου ηλεκτρονικού υπολογιστή και κάποιας γλώσσας προγραμματισμού. Μερικές από τις διαθέσιμες γλώσσες προγραμματισμού για Embedded System είναι οι εξής:

- Γλώσσα Μηχανής
- Assembly
- C
- C++
- Python
- Fortran

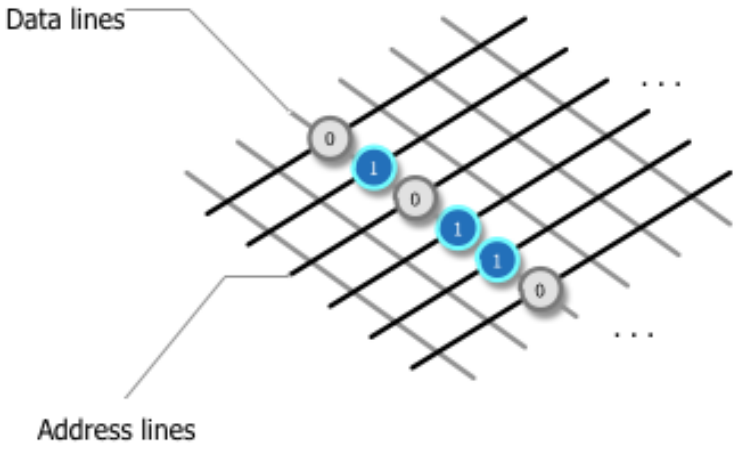
Έπειτα από τη σύνταξη του προγράμματος με κάποια από τις παραπάνω γλώσσες, θα πρέπει να μετατρέψουμε – μεταφράσουμε το σύνολο του κειμένου σε γλώσσα μηχανής. Η γλώσσα μηχανής είναι το μόνο κείμενο που μπορεί να καταλάβει το σύστημά μας καθώς αποτελείτε από άσσους και μηδενικά. Συνεπώς με την χρήση ειδικών εργαλείων που ονομάζονται Compilers μπορούμε να ετοιμάσουμε το πρόγραμμά μας ώστε να μπορεί να εκτελεστεί από το σύστημά μας. Τέλος, μεταφέρουμε το μεταφρασμένο πλέον πρόγραμμα στο εσωτερικό του συστήματός μας και ειδικότερα στην μονάδα μνήμης. Το σημείο αυτό αποτελεί και το τέλος της διαδικασίας προγραμματισμού.

1.4 Μνήμη

Η μνήμη στα Embedded Systems είναι το φυσικό μέσω στο οποίο αποθηκεύεται όλη η πληροφορία του συστήματός μας. Πληροφορίες σε ένα Embedded System αποτελούν όλα εκείνα τα δεδομένα, που χρειάζεται το σύστημά μας, τόσο για να πραγματοποίηση της διαδικασία εκτέλεσης της ροής εργασίας για την οποία έχει σχεδιαστεί, όσο και για την διαδικασία αποθήκευσης προσωρινών και μόνιμων δεδομένων που προκύπτουν από την αλληλεπίδραση με το περιβάλλον αλλά και από την εσωτερική λειτουργία. Η μνήμη, αποτελεί ένα απλό προς την κατανόηση τμήμα, ενός Embedded System καθώς παράλληλα αποτελεί βασικό κριτήριο σύγκρισης ανάμεσα στα διάφορα συστήματα με άμεσο αντίκτυπο στο κόστος παραγωγή.

Αν και υπάρχουν διαφορετικοί τύποι και κατηγορίες μνημών, που θα αναφερθούν στη συνέχεια, στο σύνολο του, η μονάδα *Μνήμης* σε όλα τα Embedded Systems έχει κοινά γνωρίσματα και λειτουργεί υπό μια βασική αρχή – φιλοσοφία.
Ξεκινώντας λοιπόν, θα πρέπει πρώτα από όλα να αναφερθεί πως κύριο χαρακτηριστικό μιας μνήμης αποτελεί η αρχιτεκτονική στην οποία στηρίζεται. Με τον όρο αυτό εννοούμε τον τρόπο με τον οποίο μια μνήμη κάνει προσπέλαση στα δεδομένα της. Για το λόγο αυτό η μνήμες χαρακτηρίζονται ως 8/16/32/64Bit αρχιτεκτονικής, σύμφωνα με την οργάνωση που επικρατεί στο εσωτερικό τους. Η παραπάνω πληροφορία είναι ιδιαιτέρα σημαντική για την κατανόηση της λειτουργίας των μνημών.

Για να κατανοήσουμε την τρόπο λειτουργίας μιας μνήμης, αρκεί να συγκρίνουμε την εσωτερική της διάταξη με ένα πίνακα από το Excel(εκζσελ) , όπου κάθε κελί χαρακτηρίζεται από δύο αριθμούς. Ο ένας αριθμός ορίζει την στήλη, ενώ ο άλλος την σειρά στην οποία βρίσκεται κάθε κελί. Με πολύ παρόμοιο τρόπο είναι οργανωμένη και μια μνήμη στο εσωτερικό της, δηλαδή ένα δεδομένο που τοποθετείτε στο εσωτερικό της ορίζεται από δυο διαφορετικούς αριθμός.



Εικόνα 1.

Όπως φαίνεται και στην *Εικόνα 1* η μνήμη είναι σαν ένα πλέγμα. Από την μία μεριά του πλέγματος προσδιορίζουμε την διεύθυνση της μνήμης την οποία θέλουμε να προσπελάσουμε(Address Lines) και από την άλλη πλευρά προσδιορίζονται τα δεδομένα της μνήμης τα οποία μας απασχολούν. Το μέγεθος των δεδομένων, ανά μονάδα, καθώς και ο τρόπος με τον οποίο γίνεται η διευθυνσιοδότησης των δεδομένων μιας μνήμης, διαφέρουν ανάλογα με την αρχιτεκτονική της. Αυτό σημαίνει πως μια μνήμη 32Bit αρχιτεκτονικής, αποθηκεύει 32Bit μεγέθους δεδομένα στο εσωτερικό της που ορίζονται από 32Bit μεγέθους, διευθύνσεις μνήμης. Παράλληλα μια μνήμη 8Bit αρχιτεκτονικής αποθηκεύει δεδομένα 8Bit μεγέθους, χρησιμοποιώντας 8Bit διευθύνσεις.

Η αρχιτεκτονική με την οποία είναι δομημένη μια μνήμη στα περισσότερα Embedded System είναι άρρηκτα συνδεδεμένη με την αρχιτεκτονική της Κεντρικής Μονάδας Επεξεργασίας CPU του συστήματος για μεγαλύτερη ταχύτητα και καλύτερη αποδοτικότητα του συστήματος.

Επιπλέον, αξίζει να σημειωθεί πως, υπάρχουν και μνήμες με διαφορετικό μέγεθος δεδομένων και διαφορετικό μέγεθος διευθύνσεων να τις ορίζει. Όπως αναφέρθηκε και προηγουμένως, οι μνήμες δια-χωρίζονται σε δύο πολύ μεγάλες και γενικές κατηγορίες.

Volatile Memory

Αυτή η κατηγορία μνημών, που στα Ελληνικά αναφέρονται ως *Μεταβαλλόμενες Μνήμες*, έχει την ιδιότητα να διατηρεί τα δεδομένα που αποθηκεύονται σε αυτή κατά την διάρκεια ύπαρξης ηλεκτρικής παροχής, ενώ παράλληλα αδυνατεί να τα διατηρήσει κατά την διακοπή παροχής ενέργειας. Συνεπώς τα δεδομένα διατηρούνται όσο η μνήμη έχει ηλεκτρική παροχή. Η συγκεκριμένη τεχνική είναι ιδιαιτέρως γνωστή και χρησιμοποιείτε σε τεράστιο βαθμό στα Embedded System για την διατήρηση ευαίσθητων ή προσωρινών δεδομένων που είναι απαραίτητα να διαγράφοντα κατά την απενεργοποίηση του συστήματος για λόγους ασφαλείας. Επιπλέον η συγκεκριμένη κατηγορία μνημών χαρακτηρίζεται από υψηλή ταχύτητα λειτουργίας, γεγονός που καθιστά τις μνήμες αυτές ιδανικές για εκτέλεση προγράμματος και ταχύτατη μεταφορά μεγάλου όγκου πληροφοριών, από και προς την Κεντρική Μονάδα Επεξεργασίας CPU. Το πιο γνωστό παράδειγμα αυτής της κατηγορίας μνημών είναι η RAM(Random Access Memory).

Non-Volatile Memory

Σε αυτή την κατηγορία μνημών, το χαρακτηριστικό γνώρισμα που συναντάμε είναι η διατήρηση των δεδομένων ακόμα και όταν η παροχή ηλεκτρικής ενέργειας εκλείψει από την μονάδα. Με αυτό τον τρόπο, οποιαδήποτε πληροφορία και δεδομένο υπάρχει αποθηκευμένο στο εσωτερικό μια *Μη Μεταβαλλόμενης Μνήμης*, όπως είναι και ο όρος της στα Ελληνικά, διατηρείτε ανέπαφη σε κάθε διακοπή παροχής ενέργειας. Ο συγκεκριμένος τύπος μνήμης είναι, επίσης πολύ διαδεδομένος καθώς χρησιμοποιείτε για την αποθήκευση μεγάλου όγκου πληροφορίας, όπου δεν απαιτείται μεγάλη ταχύτητα και μεγάλος αριθμός προσπελάσεων στο περιεχόμενό τους. Τα πιο κοινά παραδείγματα τέτοιου τύπου μνήμης, είναι ο δίσκος SSD και το USB Flash Disc.

1.5 Επικοινωνία

Μια από τις μονάδες που συμπληρώνουν ένα Embedded System είναι και η Επικοινωνία. Με τον όρο αυτό, αναφερόμαστε σε κάθε υποσύστημα η μηχανισμό που παρέχει την δυνατότητα στο σύστημά μας, σαν ένας οργανισμός, να αλληλεπιδρά με τρόπο οργανωμένο και δομημένο, με το περιβάλλον στο οποίο λειτουργεί. Στην περίπτωση αυτή δεν αναφερόμαστε στην δυνατότητα του συστήματος να αισθάνεται και να επηρεάζει το εξωτερικό του περιβάλλον που είδαμε στη μονάδα *Εισόδων και Εξόδων.* Στην προκειμένη περίπτωση η μονάδα Επικοινωνίας, είναι υπεύθυνη να επικοινωνεί, στα πλαίσια πολύ αυστηρών κανόνες και προτύπων, με την χρήση συγκεκριμένων και τυποποιημένων πρωτοκόλλων, με διάφορες συσκευές η ακόμα και άλλα συστήματα που βρίσκονται εκτός συστήματος.

Η μονάδα Επικοινωνίας είναι πολύ σημαντική στα Embedded Systems καθώς παρέχει την δυνατότητα δημιουργίας μεγαλύτερων μονάδων, που στηρίζονται στον διαχωρισμό των εργασιών, με στόχο την μεγαλύτερη ακρίβεια και αποτελεσματικότητα. Επιπλέον η μονάδα αυτή παρέχει την δυνατότητα για αλληλοσυμπλήρωση των δυνατοτήτων μεταξύ διαφορετικών συστημάτων. Δηλαδή την δυνατότητα παραχώρησης μιας πολύ συγκεκριμένης διεργασίας σε ένα άλλο σύστημα, ώστε να επιτευχθεί καλύτερο και πιο γρήγορο αποτέλεσμα. Ένα τέτοια παράδειγμα αποτελεί, η αξιοποίηση της οπτικής μονάδας CD, από ένα Embedded System για την αναγνώριση οπτικών αποθηκευτικών μέσων από το σύστημα, που συναντάμε σε έναν ηλεκτρονικό υπολογιστή. Επίσης, η χρήση ενός εξωτερικού αισθητήρα μέτρησης θερμοκρασίας που ενσωματώνεται σε ένα Embedded System για τον υπολογισμό της θερμοκρασίας, αποτελεί ένα παρόμοιο παράδειγμα.

Η δυνατότητα επικοινωνίας παρέχει στα Embedded Systems την δυνατότητα να έρχονται σε επαφή με γεγονότα και δεδομένα του εξωτερικού περιβάλλοντος, τα οποία είναι αναλογικά και άκρως δυσνόητα και απροσπέλαστα από την απλή ψηφιακή λογική που επικρατεί στο εσωτερικό του συστήματος. Επιπλέον, η ενσωμάτωση

της μονάδας επικοινωνίας στα Embedded Systems έδωσε την δυνατότητα να δια-μοιραστούν οι περίπλοκες λειτουργίες ενός μεγάλου συνόλου σε επιμέρους μικρότερα σύνολα, γεγονός που δημιούργησε μεγαλύτερη ευελιξία στον τομέα των Embedded Systems.
Η επικοινωνία στον κλάδο αυτό πραγματοποιείτε μέσα από συγκεκριμένα διεθνή πρωτόκολλα επικοινωνία, κοινώς αναγνωρισμένα από κάθε σύστημα το οποίο παρέχει μονάδα επικοινωνίας. Κάθε πρωτόκολλο από αυτά έχει τους δικούς του κανόνες, τις δικές του δυνατότητες και την δική του φιλοσοφία. Επιπλέον τα πρωτόκολλα επικοινωνίας χαρακτηρίζονται από μια δομή παρόμοια της ανθρώπινης επικοινωνίας, και για το λόγο αυτό απαιτούν τουλάχιστον δύο πλευρές, ενώ αποτελούνται από διαδικασίες, ερωτήσεως και απαντήσεως

Η μονάδα της Επικοινωνίας διαχωρίζεται σε δυο μεγάλες κατηγορίες καθώς και σε επιμέρους μικρότερες κατηγορίες σύμφωνα με τα διάφορα πρωτόκολλα που έχουν εφευρεθεί ανά τα χρόνια.

Ασύγχρονη Επικοινωνία

Η ασύγχρονη επικοινωνία αποτελεί την πιο απλή μορφή επικοινωνίας καθώς είναι μια από τις πρώτες, χρονολογικά, μορφές επικοινωνίας που συναντάμε στα Embedded Systems. Η Λειτουργία της βασίζεται στον διαχωρισμό της πληροφορία κατά την μετάδοση της σε μικρά και πολύ συγκεκριμένα, ως προς της ιδιότητές τους, πακέτα καθώς και σε μια τεχνική άτυπου συγχρονισμού στην εσωτερική λειτουργία των δύο πλευρών. Πιο συγκεκριμένα, η τεχνική αυτή ορίζει την αρχή και το τέλος μετάδοσης της πληροφορίας ανά πακέτο με ειδικές σημαίες που ονομάζονται Start Bit και Stop Bit. Με αυτό τον τρόπο οι δύο πλευρές μπορούν να αναγνωρίσουν την αρχή και το τέλος ενός πακέτου δεδομένων. Επιπλέον, καθώς δεν υπάρχει κάποιου είδους σφιχτού Handshake(χειραψίας) ανάμεσα στις δύο πλευρές, κρίνεται απαραίτητος ο συγχρονισμός δύο πλευρών στον τρόπο με τον οποίο “σπρώχνουν” ένα πακέτο ή το διαβάζουν αντίστοιχα δια μέσου του φυσικού αγωγού.Ενδεικτικά, να αναφέρουμε πως τυπικά παραδείγματα τέτοιου είδους επικοινωνίας αποτελούν οι εκτυπωτές γραφείου.

Σύγχρονη Επικοινωνία

Από την άλλη πλευρά συναντάμε την Σύγχρονη επικοινωνία η οποία παρόλο που εφευρέθηκε μόλις λίγα χρόνο ύστερα από την Ασύγχρονη έχει επικρατήσει ως η πλέον καλή γρήγορη και σφιχτή ως προς το Handshake, επικοινωνία ανάμεσα στα Embedded Systems. Στην κατηγορία αυτή συναντάμε πρωτόκολλα τα οποία χρησιμοποιούν περισσότερα από 2 καλώδια(φυσικό μέσω) για την μεταφορά δεδομένων και αυτό διότι σε αυτή την κατηγορία απαιτείται η ύπαρξη φυσικού συγχρονισμού των 2 πλευρών. Εδώ οι δεν υπάρχει η ανάγκη για εσωτερικό μηχανισμό στις δύο πλευρές για συγχρονισμό της διαδικασία επικοινωνίας ενώ αντίθετα η αποστολή δεδομένων δεν γίνεται με διαχωρισμό σε πακέτα. Επίσης στις σύγχρονες επικοινωνίες υπάρχει η δυνατότητα επικοινωνίας με παραπάνω από 2 πλευρές καθώς απαιτείται απαραιτήτως ένας μηχανισμός διευθυνσιοδότησης που υλοποιείτε σε κάθε πρωτόκολλο σύγχρονης επικοινωνίας.

Αναλύοντας λοιπόν τα Embedded Systems στις μονάδες από τις οποίες αποτελούνται καταφέραμε να διακρίνουμε τις λειτουργίες της κάθε μονάδας αλλά και τον ξεχωριστό ρόλο της κάθε μιας σε ένα Embedded System. Επίσης καταλήγουμε στο συμπέρασμα πως, παρ' όλο που κάθε μονάδα του συστήματος είναι φτιαγμένη για να κάτι εντελώς διαφορετικό από τις υπόλοιπες, όταν συνδυάσουν τις δυνάμεις τους κατορθώνουν να εκτελούν συγκεκριμένες λειτουργίες με πολύ αποτελεσματικό τρόπο. Συνεπώς ο συνδυασμός όλων αυτόν των μηχανισμών ως ένα σύνολο αποδεικνύεται πως λειτούργει πολύ αποτελεσματικά, με τρόπο, καθόλου τυχαίο. Ο συνδυασμός αυτός με μία λέξη ονομάζεται Embedded Systems.

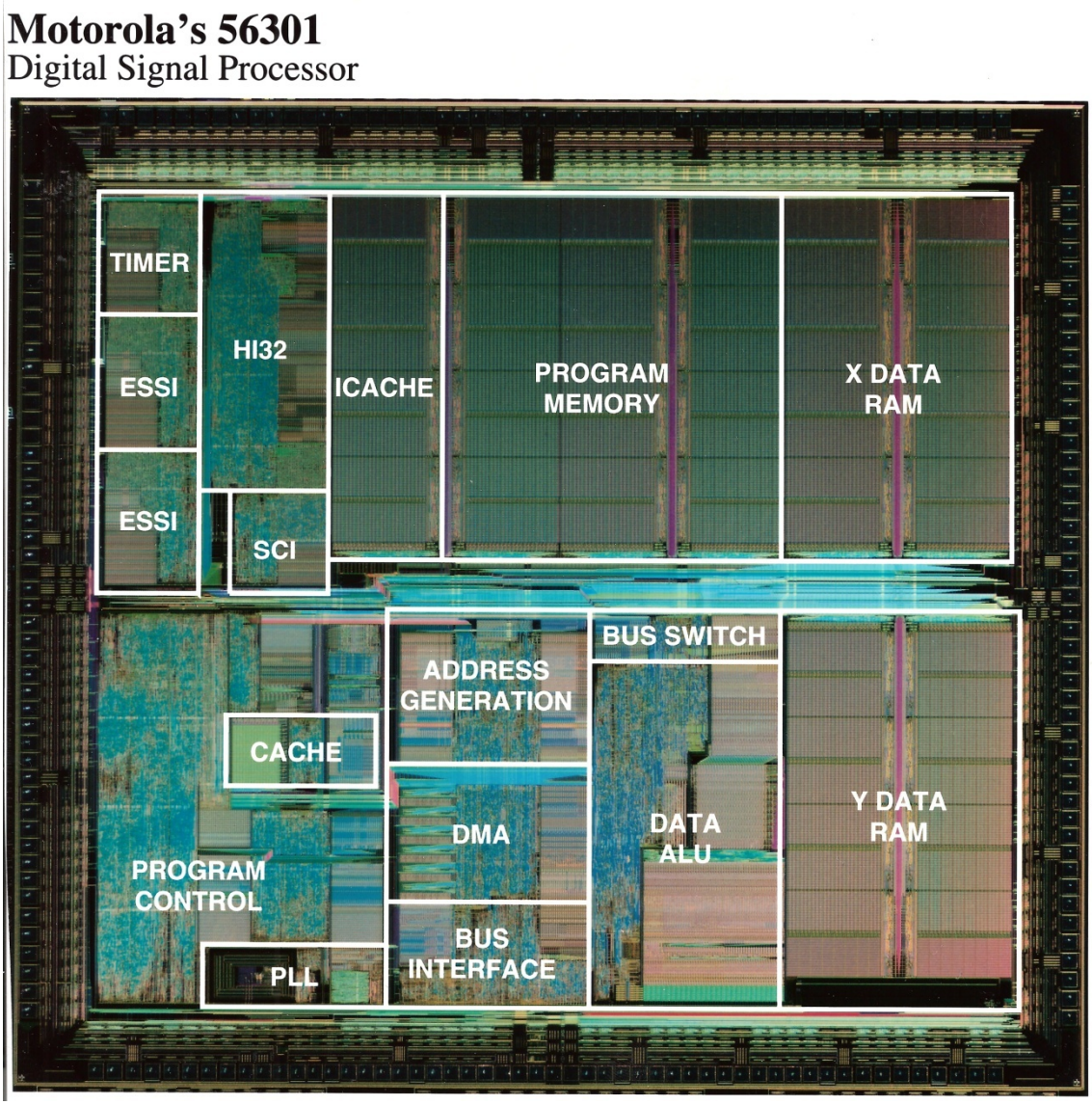
Η επιτυχία, όμως του παραπάνω μοντέλου δεν έμεινε απαρατήρητη οδηγώντας σε μια συνεχόμενη προσπάθεια για δημιουργία μικρότερων συστημάτων σαν και αυτά. Συστήματα όπως εκείνα των Embedded Systems που να μπορούν να λειτουργούν αυτόνομα με άμεσο στόχο την εφαρμογή τους σε διάφορες κατασκευές με διαφορετικές απαιτήσεις και λειτουργίες.

Το επίτευγμα αυτό συναντάται στους σύγχρονους ηλεκτρονικούς microcontroller που κατακλύζουν οποιαδήποτε μικρή ή μεγάλη ηλεκτρονική συσκευή υπάρχει γύρο μας. Όσο παράξενο και αν ακούγεται, κάθε μικρός ή μεγάλος microcontroller αποτελεί ένα πλήρως ανεπτυγμένο Embedded Systems, με όλα τα γνωρίσματα και χαρακτηριστικά των Embedded System όπως αυτά αναλύθηκαν προηγουμένως. Πιο συγκεκριμένα κάθε microcontroller της αγοράς περιέχει τις πέντε(5) βασικές μονάδες στο εσωτερικό του, σε ένα πολύ συνεκτικό και αποδοτικό σύνολο.

Όπως αναφέρθηκε στην αρχή ο όρος Embedded System περικλείει πολλά πράγματα και έννοιες και είναι δύσκολο να περιγραφή με μία μόνο πρόταση. Το γεγονός πως όλοι οι microcontroller της αγοράς διαθέτουν όλα εκείνα τα στοιχεία που αποτελούν ένα Embedded System έχει δημιουργήσει μια σύγχυση με την πραγματική έννοια του όρου καθώς πλέον, ο τελευταίος είναι άρρηκτα συνδεδεμένος, με εκείνον του microcontroller.

Η συνεχής εξέλιξη στην τεχνολογία υλικών καθώς και τα επιτεύγματα στον πεδίο των χημικών επιστημών κατόρθωσαν να ελαχιστοποιήσουν τον όγκο

των εξαρτημάτων σε τέτοιο βαθμό που να μπορούν όλα μαζί να συνδυαστούν σε έναν πολύ μικρό χώρο και να αποτελέσουν το σύγχρονο microcontroller.



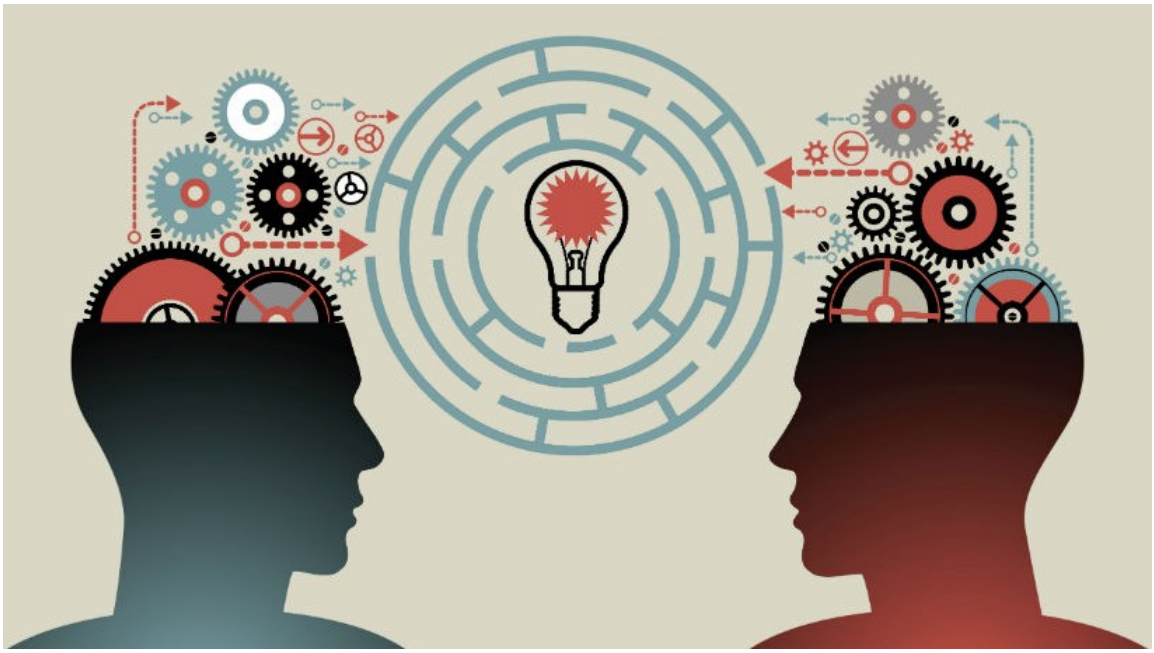
Όπως φαίνεται στην παραπάνω μεγεθυμένη εικόνα από τον πυρήνα ενός microcontroller πλέον όλα τα τμήματα είναι διακριτά και πολύ μελετημένα τοποθετημένα στο εσωτερικό του. Έτσι λοιπόν πλέον κάθε microcontroller σαν ένα πλήρως ανεπτυγμένο Embedded System αποτελείτε από κεντρική μονάδα επεξεργασίας, μνήμες, μονάδα εσόδων εξόδων, και μονάδα επικοινωνίας. Παράλληλα στο σύνολο αυτό υπάρχει η δυνατότητα να ενσωματωθεί το λογισμικό, που αποτελεί και την τελευταία μονάδα του συστήματός. Με τον τρόπο αυτό ένας microcontroller, σαν ένα Embedded

System έχει πλέον την δυνατότητα να πραγματοποιεί συγκεκριμένες και πολύ αυστηρά προκαθορισμένες ενέργειες και λειτουργίες στα πλαίσια των κανόνων και των περιορισμών που ορίζουν οι φυσικές δυνατότητες του συστήματος καθώς και το λογισμικό (Software).

Η συνεχόμενη εξέλιξη στον τομέα των microcontroller οδήγησε στην ενσωμάτωση κι άλλον περιφερειακών μονάδων στο εσωτερικό των microcontroller που έως τότε θεωρούνταν ως περιφερικές συσκευές ή ξεχωριστά Embedded Systems. Περιφερειακές μονάδες σαν και αυτές είναι οι διάφοροι μηχανισμοί επικοινωνίας, οι διάφοροι αισθητήρες όπως, αισθητήρας θερμοκρασίας αναλογικών τιμών ή ακόμα και αφής. Επιπλέον ενσωματώθηκε, ρολόι πραγματικού χρόνου με δυνατότητα διατήρησης ημερολογίου (Real Time Clock) καθώς και περισσότερες από μία μνήμες με διαφορετικές ιδιότητες και χαρακτηριστικά.

Όλα τα παραπάνω, συν πολλών άλλων ακόμα περιφερειακών μονάδων αποτελούν τους σύγχρονους microcontroller. Υπάρχουν εκατοντάδες διαφορετικοί microcontroller στην αγορά που διαφέρουν σε χαρακτηριστικά και προσανατολισμό μεταξύ τους. Συνεπώς ανάλογα με τα χαρακτηριστικά που κάθε εφαρμογή έχει, μπορούμε να επιλέξουμε τον κατάλληλο ως προς τις εσωτερικές περιφερειακές μονάδες microcontroller, μέσα από ένα τεράστιο εύρος.

Κεφάλαιο 2 – Θεωρία



2.1 LAB 1 – General Clock Module

Σε αυτό το μάθημα(LAB_1), γίνεται παρουσίαση του βασικότερου καθώς και καθοριστικού, για την λειτουργία, συστήματος ενός microcontroller και αυτό είναι, το **Clock Module**.

Με τον όρο Clock Module εννοείται το γενικότερο σύστημα χρονισμού ενός microcontroller. Κάθε microcontroller που παράγεται στο κόσμο, το minimum και το πρώτο σύστημα που απαιτεί για να λειτουργήσει, είναι το κεντρικό του ρολόι. Ένας microcontroller χρησιμοποιεί το κεντρικό του ρολόι, που στο εξής θα αναφέρεται ως “MCU Clock”, για να πραγματοποιεί – εκτελεί ενέργειες στον πυρήνα του με συγκεκριμένη σειρά και με μεγάλη ακρίβεια ως προς την συχνότητα εκτέλεσης τους. Το MCU Clock είναι ένα ψηφιακός παλμός εναλλαγής τάσης ρεύματος **“0”** και **“1”** , που βοηθά στον συντονισμό και στην επικοινωνία, με σειριακό τρόπο, όλων των εσωτερικών περιφερειακών συστημάτων ενός microcontroller με τον πυρήνα, καθώς και όλων των περιφερειακών συστημάτων μεταξύ τους.

Υπεύθυνος για τον συντονισμό αυτής της επικοινωνίας των συστημάτων, είναι ο πυρήνας. Για τον λόγο αυτό, ο πυρήνας του συστήματος θα πρέπει να δουλεύει σε πολύ μεγαλύτερες ταχύτητες από τα περιφερειακά συστήματα τα οποία έχει να διαχειριστεί. Το παρακάτω block diagram παρουσιάζει της εσωτερική διασύνδεση του Clock Module με τα περιφερικά. Όπως μπορούμε να δούμε το Clock Module συνδέεται με κάθε ένα σύστημα ξεχωριστά. Επίσης βλέπουμε, πως ανάλογα με τον επιλογή του κρυστάλλου, το Clock Module συνδέεται σε λιγότερα ή περισσότερα συστήματα καθώς και πως όλα τα εσωτερικά συστήματα χρησιμοποιούν το ½ της συχνότητας του κεντρικού ρολογιού MCU Clock.

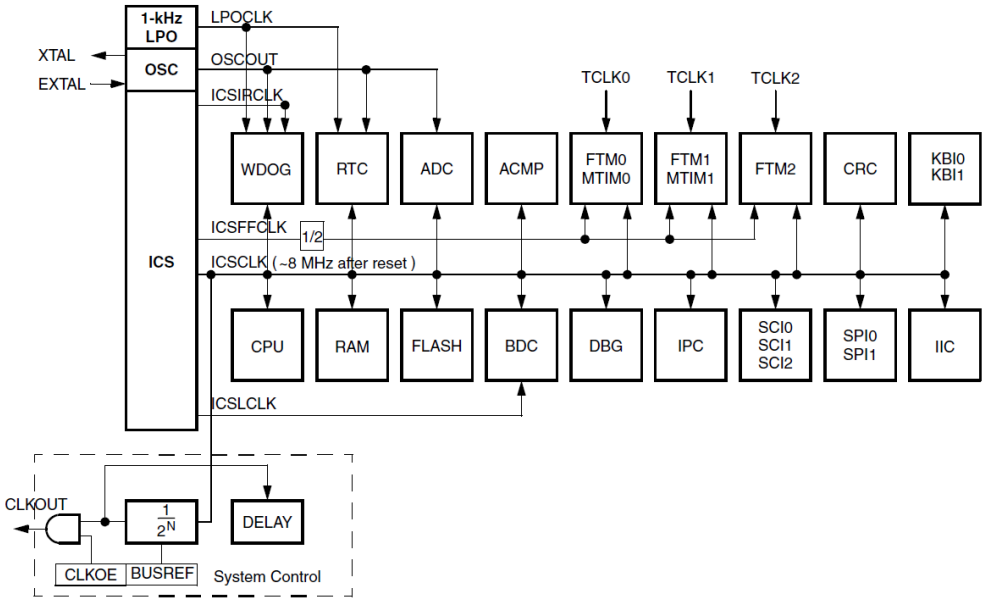


Figure 1-2. System clock distribution diagram

Σε κάθε χτύπημα του MCU Clock (γνωστό και ως Κύκλος Μηχανής) το παρών σύστημα εκτελεί αυστηρά μια ενέργεια, το τελευταίο αποτελεί την βασική αρχή λειτουργίας των microcontrollers. Για την καλύτερη κατανόηση αυτής της αρχής θα πρέπει να παρουσιαστεί ένα παράδειγμα εκτέλεσης εντολών, δανειζόμενοι ένα τμήμα κώδικα της γλώσσας assembly.

```
; compute (A + B) - (C + D) (assumes no carry or borrow)
            lda    oprC          ;oprC -> accumulator
            add    oprD          ;oprC + oprD -> accumulator
            psha                 ;intermediate result to SP+1
            lda    oprA          ;oprA -> accumulator
            add    oprB          ;oprA + oprB -> accumulator
            sub    1,sp          ;(A+B)-(C+D) to accumulator
            ais    #1            ;deallocate local space
```

Στο παραπάνω παράδειγμα παρατηρούμε ένα τμήμα κώδικα γραμμένο σε Assembly το οποίο πραγματοποιεί 2 προσθέσεις και μια αφαίρεση. Συμβουλευόμενοι τον παρακάτω πίνακα που μας δείχνει πόσους κύκλους μηχανής απαιτεί κάθε instruction set για να εκτελεστεί, θα υπολογίσουμε πόσο χρόνο θα χρειαστεί ο microcontroller να μας δώσει το επιθυμητό αποτέλεσμα.

(Επιλέγουμε την συγκεκριμένη γλώσσα καθώς, ο κάθε πυρήνας επεξεργαστή έχει δημιουργηθεί για να καταλαβαίνει συγκεκριμένες δεσμευμένες λέξεις που ονομάζονται “Instruction Set”. Τα παραπάνω Instruction Set είναι πολύ συγκεκριμένα για κάθε επεξεργαστή, “μιλάνε” κατευθείαν στον πυρήνα του και ο χρόνος εκτέλεσής τους μετριέται σε Clocks ή αλλιώς κύκλοι μηχανής).

Source Form	Operation	Object Code	Cycles
LDA oprC	Load Accumulator from Memory A ← (M)	B6 dd	3
ADD oprD	Logical AND A ← (A) & (M)	B4 dd	3
PSHA	Push Accumulator onto Stack Push (A); SP ← (SP) – $0001	87	2
LDA oprC	Load Accumulator from Memory A ← (M)	B6 dd	3
ADD oprD	Logical AND A ← (A) & (M)	B4 dd	3
SUB oprx8,SP	Subtract A ← (A) – (M)	9E E0 ff	4
AIS #opr8i	Add Immediate Value (Signed) to Stack Pointer SP ← (SP) + (M)	A7 ii	2
ΣΥΝΟΛΟ	**18 Cycles**		

Συνεπώς συμφώνα με τον παραπάνω πίνακα, ο κώδικας που δόθηκε στον microcontroller απαιτεί 18 κύκλους μηχανής για να εκτελεστεί. Αν υποθέσουμε λοιπόν πως το κεντρικό ρολόι του microcontroller είναι ρυθμισμένο να λειτουργεί στα 16MHz προκύπτει περίοδος T=62,5ns.

Συχνότητα:	F=16MHz
Περίοδος:	T=1/F => T=62,5ns

Η περίοδος 62,5ns αποτελεί το χρόνο που μεσολαβεί ανάμεσα στους κύκλους μηχανής, και καθώς γνωρίζουμε τους κύκλους μηχανής που απαιτεί ο κώδικας μας μπορούμε να υπολογίσουμε τον χρόνο εκτέλεσής.

Χρόνος Εκτέλεσης 1^{ος} Κύκλου	62,5ns
Απαιτούμενοι κύκλοι μηχανής	18 Cycles

Συνολικός χρόνος εκτέλεσης προγράμματος:

18 x 62,5ns => 18 x 0,0000000625s => 0,00001125 => 11,25us

Το ερώτημα όμως που γεννάτε είναι το πώς δημιουργείτε αυτό το χτύπημα του ρολογιού και πως διασφαλίζεται η ακρίβεια και η διατήρηση της συχνότητας του;

Την απάντηση στην ερώτηση μας την δίνουν οι **Ταλαντωτές**. Η μονάδα του κεντρικού ρολογιού του microcontroller περιλαμβάνει έναν ή περισσότερους ταλαντωτές (oscillators).Οι ταλαντωτές είναι ειδικά κυκλώματα ή κρύσταλλοι, που έχουν την ιδιότητα να δημιουργούν το φαινόμενο της ταλάντωσης, δηλαδή παραγωγή περιοδικού σήματος του οποίου η συχνότητα καθορίζεται εξολοκλήρου από τις τιμές των εξαρτημάτων που τα αποτελούν, χωρίς να εφαρμόζεται σήμα στην είσοδό τους.

Οι ταλαντωτές διαχωρίζονται σε δύο μεγάλες κατηγορίες:

1. **Αρμονικοί :** Η έξοδός τους είναι ένα ημιτονοειδές σήμα.
2. **Ανατροπής :** Η έξοδός τους δεν αποτελεί ημιτονοειδές σήμα αλλά οποιοιδήποτε άλλη περιοδική κατάσταση.

Επίσης οι ταλαντωτές κατηγοριοποιούνται αναλόγως με την μορφή εξόδου του σήματός που παράγουν

- Ημιτόνιο
- Τετραγωνική
- Τριγωνική
- Πριονωτή

και την συχνότητα στην οποία λειτουργούν

- Υψηλή
- Χαμηλή
-

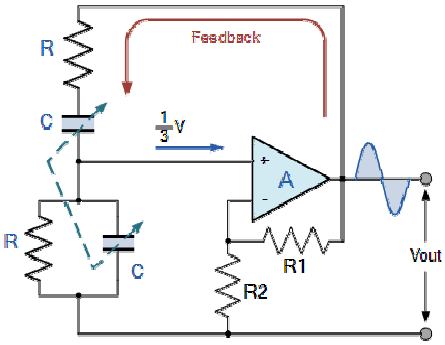
Πιο συγκεκριμένα, στους microcontroller χρησιμοποιούνται 3 κατηγορίες ταλαντωτών:

1. RC ταλαντωτής.
2. LC ταλαντωτής.
3. Ταλαντωτής κρυστάλλου.

2.1.1 RC Ταλαντωτής

Αποτελεί το πιο διαδεδομένο κύκλωμα ταλαντωτή καθώς και το πιο οικονομικό. Όπως αναφέρει και το όνομά του, το συγκεκριμένο κύκλωμα ταλάντωσης αποτελείται από δύο βασικά ηλεκτρονικά εξαρτήματα, αυτά είναι οι αντιστάσεις και οι πυκνωτές. Η RC διάταξη ταλάντωσης είναι καταδικασμένη να αποσταθεροποιείτε με τον χρόνο λόγο των απωλειών ενέργειας μέσω των ηλεκτρονικών εξαρτημάτων, για το λόγο αυτό

απαιτείτε ένα κύκλωμα ανατροφοδότησης(feedback) για την διατήρηση της ενέργειας και την σταθεροποίηση της περιοδικότητας. Παρακάτω παρουσιάζεται ένα τυπικό κύκλωμα RC ταλαντωτή καθώς και ο τύπος υπολογισμού της τιμής συχνότητας εξόδου.



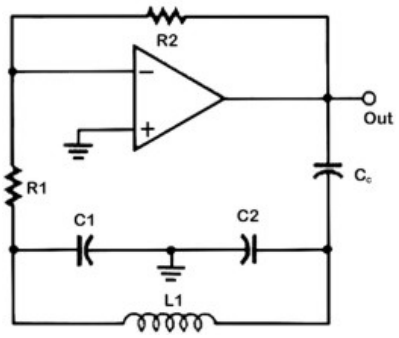
$$f_r = \frac{1}{2\pi RC}$$

2.1.2 LC Ταλαντωτής

Επίσης ένα από τα πιο διαδεδομένα κυκλώματα ταλάντωση είναι αυτό του LC ταλαντωτή. Εδώ τα γράμματα L και C αντιστοιχούν στα ηλεκτρονικά εξαρτήματα, του πηνίου και του πυκνωτή αντίστοιχα.

Η αρχή λειτουργίας του συγκεκριμένου κυκλώματος, αποτελεί την αναγνωρισμένη φυσική ιδιότητα πηνίου-πυκνωτή(LC Thomson) σύμφωνα με την οποία η συγκεκριμένη διάταξη πραγματοποιεί ηλεκτρομαγνητική ταλάντωση αντίστοιχη με αυτή της μηχανικής ταλάντωσης ελατηρίου-σώματος.

Η LC διάταξη ταλάντωσης είναι καταδικασμένη να αποσταθεροποιείτε με τον χρόνο λόγο των απωλειών ενέργειας μέσω των ηλεκτρονικών εξαρτημάτων, για το λόγο αυτό και η παρούσα διάταξη απαιτεί ένα κύκλωμα ανατροφοδότησης(feedback) για την διατήρηση της ενέργειας και την σταθεροποίηση της περιοδικότητας. Παρακάτω παρουσιάζεται ένα τυπικό κύκλωμα LC ταλαντωτή καθώς και ο τύπος υπολογισμού της τιμής συχνότητας εξόδου.



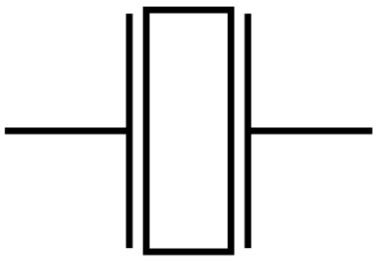
$$f_r = \frac{1}{2\pi\sqrt{LC}}$$

2.1.3 Ταλαντωτής Κρυστάλλου

Σε αυτή την κατηγορία ταλαντωτών συναντάμε τους κρυστάλλους. Η φυσική σύσταση των κρυστάλλων που υπάρχουν στην φύση και που χρησιμοποιούνται στους microcontrollers, πιο συγκεκριμένα του κρυστάλλου χαλαζία και τουρμαλίνη, είναι τέτοια ώστε εάν σε ένα κομμάτι κρυστάλλου εφαρμόσουμε μηχανική πίεση τότε παρατηρείτε πως στα άκρα του ο κρύσταλλος εφαρμόζει μια ηλεκτρική τάση αντίστοιχα. Η παραπάνω παρατήρηση αποτελεί, την αρχή του πιεζοηλεκτρικού φαινομένου, καθώς και φυσική ιδιότητα των κρυστάλλων. Στους παραπάνω αναφερόμενους κρυστάλλους όμως, παρατηρήθηκε πως το πιεζοηλεκτρικό φαινόμενο διατηρείτε και αν αντιστρέψουμε την παραπάνω διαδικασία. Δηλαδή, αν εφαρμόσουμε μια εναλλασσόμενη τάση στα άκρα του κρυστάλλου τότε ο τελευταίος τείνει να αλλάζει τις φυσικές του διαστάσεις άρα να διαστέλλεται και να συστέλλετε σύμφωνα με την εφαρμοζόμενης, σε αυτόν, τάση αντίστοιχα με απόλυτα συγκεκριμένη συχνότητα.

Τέλος, το ποσοστό διαστολής-συστολής καθώς και η συχνότητα ταλάντωσης εξαρτώνται από το πάχος και το σχήμα των κρυστάλλων και όχι από τον ρυθμό μεταβολής της τάσης στα άκρα τους.

Εκμεταλλευόμενοι την παραπάνω παρατήρηση οι σύγχρονοι μηχανικοί τοποθέτησαν μια πλάκα κρυστάλλου ανάμεσα σε δυο οπλισμούς δημιουργώντας με αυτόν τον τρόπο, μια διάταξη διακόπτη, ο οποίος ανοιγοκλείνει με την συχνότητα ταλάντωσης του τοποθετημένου κρυστάλλου. Παρακάτω, παρουσιάζεται η γραφική αναπαράσταση του ηλεκτρονικού εξαρτήματος του κρυστάλλου.



2.2 LAB 2 – Digital Inputs Outputs (GPIO / KBI)

Στην ενότητα αυτή θα ασχοληθούμε με τα GPIO, δηλαδή γενικής χρήσης pins καθώς και με την KBI(Keyboard Interrupts) περιφερειακή μονάδα, που όπως αναφέρει και το όνομά της, μας δίνει τη δυνατότητα να ελέγξουμε την αλλαγή κατάστασης ενός pin του microcontroller με την μορφή Interrupt.

2.2.1 GPIO (General Purpose Inputs Outputs)

Τα GPIO αποτελούν μια εσωτερική περιφερειακή μονάδα μέσα σε έναν microcontroller. Το συγκεκριμένο περιφερειακό χρησιμοποιεί κάποια από τα “πόδια”, που στο εξής θα αναφέρονται ως pins, πολλές φορές και όλα, σαν γενικής χρήσης ακροδέκτες. Με το παραπάνω εννοούμε πως ο microcontroller δύναται να χρησιμοποιήσει κάθε ένα ή σε ομάδες, pin\s, άλλοτε ως εξόδους και άλλοτε ως εισόδους, από και προς τον microcontroller.

Ρυθμίζοντας ένα pin σε λειτουργία εξόδου, μπορούμε να ελέγξουμε την ύπαρξη ή την απουσία ρεύματος, πάνω στο pin. Με τον τρόπο αυτό μπορούμε για παράδειγμα να ανοιγοκλείσουμε έναν λαμπτήρα καθώς και να επικοινωνήσουμε με εξωτερικές συσκευές γενικότερα.

Ρυθμίζοντας ένα pin σε λειτουργία εισόδου, μπορούμε να διαβάσουμε την κατάσταση του, δηλαδή μπορούμε να αναγνωρίσουμε την ύπαρξη ή την απουσία ρεύματος επάνω του. Λειτουργώντας ένα pin, λοιπόν ως είσοδο μπορούμε να αντιληφθούμε ένα ερέθισμα από το εξωτερικό περιβάλλον του microcontroller και να το αξιοποιήσουμε για να πραγματοποιήσουμε μια ενέργεια.

Η GPIO μονάδα χωρίζει σε ομάδες, συνήθως των 8 για 8bit αρχιτεκτονική και 32 για 32bit αρχιτεκτονική αντίστοιχα, τα pin του microcontroller τις οποίες αναφέρει ως πόρτες (PORT). Για να προγραμματίσουμε ένα pin λοιπόν ως έξοδο ή είσοδο, χρειάζεται να αναφερθούμε σε αυτό μέσω της πόρτας που το ορίζει “PORTx” (π.χ. PORTB) καθώς και τον αριθμό του pin πάνω στην πόρτα.

Στον συγκεκριμένο microcontroller κάθε pin αποτελεί μέρος της GPIO μονάδα πλην εκείνων της τροφοδοσίας.

Η μονάδα GPIO διαφέρει σε κάθε microcontroller ως προς τις δυνατότητες που προσφέρει αν pin. Με αυτό εννοούμε πως πέραν των δυο βασικών λειτουργιών που δύναται να διεκπεραιώσει ένα pin(είσοδος-έξοδος) , υπάρχουν κάποιες επιπλέον καταστάσεις και δυνατότητες που έρχονται να τελειοποιήσουν τις δυνατότητες του συστήματος. Παρακάτω παρουσιάζονται αυτές οι δυνατότητες καθώς και η επεξήγησή τους.

- **Output:** Έξοδος σήματος από το Pin του microcontroller στο περιβάλλον.
- **Input:** Είσοδος από το περιβάλλον στο εσωτερικό του microcontroller.
- **High-Z** Το pin βρίσκετε σε ηρεμία, δίχως να έχει οριστεί κάποια συγκεκριμένη λειτουργία.
- **PULL-UP** Δυνατότητα εφαρμογής pull-up αντίστασης στο pin.
- **PULL-DOWN** Δυνατότητα εφαρμογής pull-down αντίστασης στο pin**.**
- **High Current Drive** Δυνατότητα ενισχυμένου ρεύματος στο pin.
- **True Open-Drain** Ανοιχτού συλλέκτη pin.

Για να γίνουν πιο κατανοητά τα παραπάνω θα δούμε την φυσική δομή των pin στο εσωτερικό του microcontroller, για τρείς (3) διαφορετικές περιπτώσεις:

1) Δομή ενός απλού GPIO.

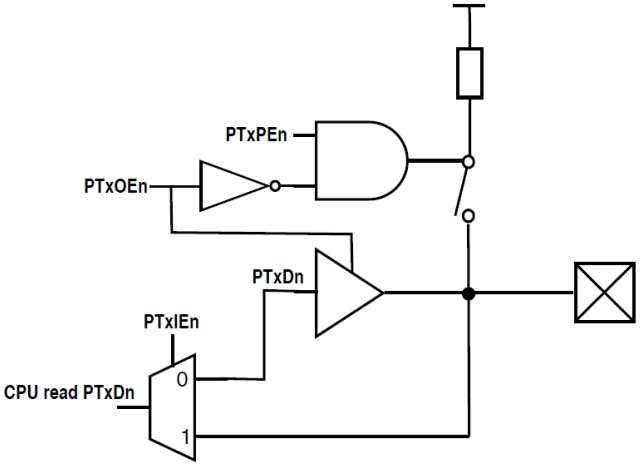


Figure 7-1. Normal I/O structure

2) Δομή GPIO σε διάταξη True-Open-Drain (ανοιχτού συλλέκτη).

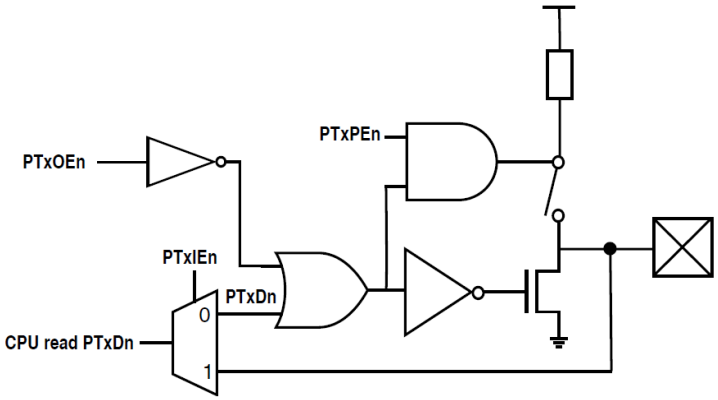


Figure 7-2. SDA(PTA2)/SCL(PTA3) structure

3) Δομή GPIO σε διάταξη High Current Drive

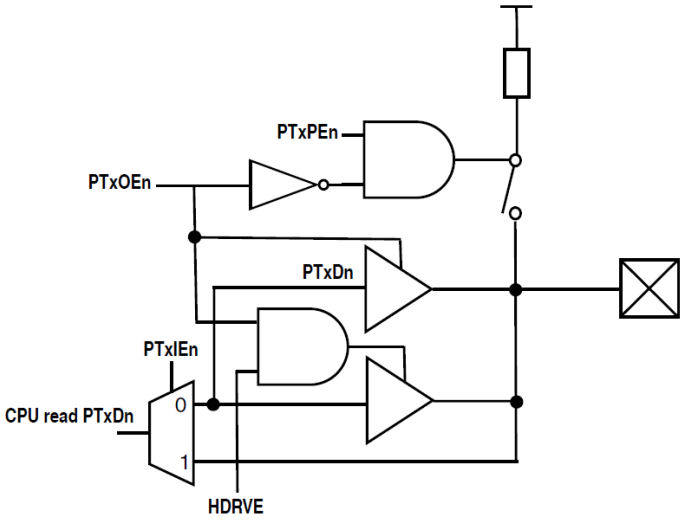


Figure 7-3. High drive I/O structure

2.2.2 KBI (Keyboard Interrupts)

Τα Keyboard Interrupts αποτελούν μια εσωτερική μονάδα στους microcontroller, η οποία μπορεί και διαχειρίζεται κάποια από τα pin, ακόμα και όλα σε κάποιες περιπτώσεις, ώστε να λειτουργούν σαν σηματοδότες τύπου Interrupt. Με το παραπάνω εννοούμε, πως εφαρμόζοντας το KBI module σε ένα pin, μας δίνεται η δυνατότητα να παρακολουθούμε την κατάσταση του, με ασύγχρονο τρόπο. Αυτό σημαίνει πως ανάλογα με τον προγραμματισμό του KBI πάνω σε ένα pin, το KBI module πρόκειται να διακόψει με την μορφή “interrupt” την κανονική λειτουργία του προγράμματός μας ,αν η παραπάνω προγραμματισμένη συνθήκη επιβεβαιωθεί.

Τα KBI pins, βοηθούν το σύστημα να πάρει ένα ή και περισσότερα ακαριαία σήματα από τον έξω κόσμο και να τα διαχειριστή σύμφωνα με το πρόγραμμα, καθώς επίσης βοηθούν στην διαδικασία εξόδου από χαμηλής ενέργειας λειτουργίες (low-power modes) του microcontroller.

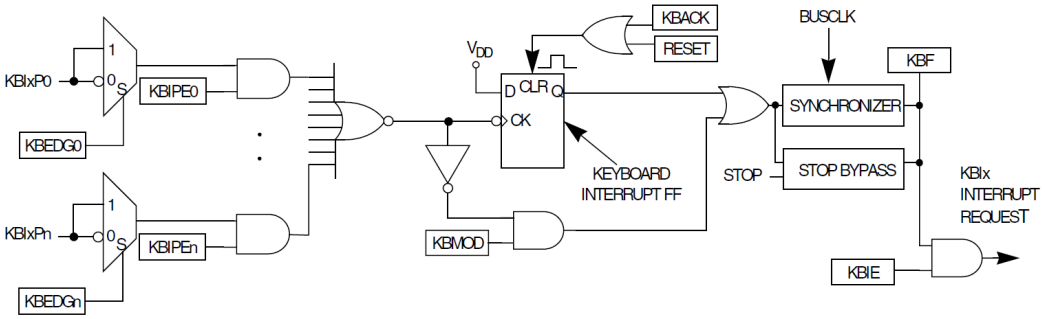


Figure 11-1. KBI block diagram

Στον συγκεκριμένο microcontroller μας δίνεται η δυνατότητα να προγραμματίσουμε
έως και 8 KBI pins. Οι προγραμματιζόμενες καταστάσεις που μπορούμε να διαχειριστούμε ανά pin με το KBI περιφερειακό είναι οι παρακάτω:

- Falling edge sensitivity only.
- Rising edge sensitivity only.
- Both falling edge and low-level sensitivity
- Both rising edge and high-level sensitivity

Πιο συγκεκριμένα:

Edge-Only Sensitivity: Σε αυτή την περίπτωση μόνο οι αλλαγές κατάστασης στο pin γίνονται αντιληπτές και στέλνουν σήμα στον microcontroller. Δηλαδή, αν η κατάσταση του pin αλλάξει από Hi σε Low ή το αντίθετο, τότε εκείνη τη στιγμή ο microcontroller θα διακόψει την κανονική του λειτουργία και θα ενημερώσει το πρόγραμμά του, μέσω ενός interrupt για την αλλαγή κατάστασης στο pin.

Edge and Level Sensitivity: Σε αυτή την περίπτωση γίνονται αντιληπτές τόσο οι αλλαγές κατάστασης στο pin όσο και το λογικό επίπεδο του pin. Δηλαδή, ο microcontroller δύναται να αναγνωρίσει δύο(2) διαφορετικές καταστάσεις και όταν αυτές επαληθευτούν να διακόψει την κανονική του λειτουργία του και να στείλει σήμα στο πρόγραμμα. Οι παραπάνω δύο(2) καταστάσεις είναι οι εξής:

1) Η κατάσταση του pin αλλάζει από Hi σε Low ή το αντίθετο, **2)** το προγραμματισμένο pin διατηρεί λογικό επίπεδο.

2.3 LAB 3 – Timers (FTM / MTIM)

Στην ενότητα αυτή θα ασχοληθούμε με ένα από τα βασικότερα περιφερικά συστήματα που συναντάμε στους microcontrollers, τους Timers. Οι Timers που έχει ένας microcontroller στο εσωτερικό του, είναι μονάδες οι οποίες βάση κάποιου συστήματος συγχρονισμού στην είσοδό τους είναι εξειδικευμένες να μετρούν μονάδες χρόνου με ακρίβεια.

Οι Timers σε ένα σύστημα μας βοηθούν, να μετράμε χρόνο και να πραγματοποιούμε ενέργειες, σε πολύ συγκεκριμένα χρονικά διαστήματα. Οι Timers κατηγοριοποιούνται σύμφωνα με την ανάλυσή τους , δηλαδή ανάλογα με την υποδιαίρεση που μπορούν να κάνουν στο μηχανισμό συγχρονισμού (π.χ. MCU Clock) που εφαρμόζεται στην είσοδό τους. Για το λόγο αυτό στα Embedded Systems συναντάμε timers με 8bit,16bit ακόμα και 32bit ανάλυσης, δηλαδή ο prescaller τους, που θα συναντήσουμε και παρακάτω, μπορεί να πραγματοποιήσει υποδιαιρέσεις στο ρολόι εισόδου με ανάλυση 8/16/32bit και άρα να δημιουργήσει μικρότερες περιόδους και μεγαλύτερες συχνότητες στην έξοδο του περιφερειακού όσο η ανάλυση μεγαλώνει.

Ένας Timer αποτελείτε από τρία(3) μέρη:

1. **Είσοδος (Input)**
   Κάθε Timer περιφερειακό απαιτεί στην είσοδό του μια μονάδα χρονισμού, δηλαδή ένα ρολόι που θα χρησιμοποιεί σαν βάση για την λειτουργία του, όπως κάθε περιφερειακό σε έναν microcontroller.Οι διαφορά του συγκεκριμένου περιφερειακού με όλα τα άλλα είναι πως οι Timers μπορούν να χρησιμοποιήσουν διάφορα ρολόγια ως βασικό σύστημα χρονισμού πέραν του κεντρικού ρολογιού (MCU Clock), όπως παραδείγματος χάρη έναν εξωτερικό κρύσταλλο(external oscillator),τον εσωτερικό κρύσταλλο IRC(internal reference clock) 4Mhz,τον ειδικό εσωτερικό κρύσταλλο χαμηλής κατανάλωσης LPC(low power clock) 1Mhz.

2. **Μηχανισμός Υπολογισμού Συχνότητας (Prescaller-Register-Functionality)**
   Στο σημείο αυτό, πραγματοποιούνται όλες οι ρυθμίσεις του περιφερειακού (Timer) και διαμορφώνεται ο τρόπος λειτουργίας του. Συμφώνα με το ρολόι εισόδου που έχει οριστεί στο προηγούμενο σκέλος και σύμφωνα με την ανάλυση(8bit-16bit-32bit) που δύναται να προσφέρει ο microcontroller στον Timer, προγραμματίζουμε τον Prescaller, δηλαδή το σύστημα υποδιαίρεσης που θα εφαρμοστεί στο ρολόι εισόδου ώστε να διαμορφώσουμε την συχνότητα εξόδου του Timer. Το παραπάνω στάδιο είναι το πλέον καθοριστικό, για την σωστή λειτουργία του Timer.Επιπλέον στο σημείο αυτό μπορούμε να προγραμματίσουμε το γενικότερο τρόπο λειτουργίας του Timer, επηρεάζοντας ρυθμίσεις όπως την ύπαρξη interrupt ή μη κατά την ολοκλήρωση του μετρήματος, τον τρόπο μέτρησης Up-Counting/Down-Counting και πολλών άλλων.

3. **Έξοδος (Output)**
   Αυτό αποτελεί το τελικό στάδιο ενός Timer. Στο σημείο αυτό και έπειτα από τον σωστό προγραμματισμό, το περιφερειακό Timer δίνει στην έξοδό του ένα σήμα σε τακτά χρονικά διαστήματα διαμορφώνοντας συχνότητα σύμφωνα με τον προγραμματισμό στο σκέλος 2.

Λειτουργία

Καθώς οι Timers αποτελούν ένα σύνθετο μηχανισμό-περιφερειακό για τους microcontroller, έχουν διάφορους τρόπους λειτουργίας, παρακάτω θα παρουσιαστεί το τυπικό μοντέλο λειτουργίας ενός Timer που θα μας βοηθήσει να καταλάβουμε τον τρόπο λειτουργίας του.

Το πιο απλό μοντέλο, είναι ένας Timer 8bit ανάλυσης ο οποίος δουλεύει με λογική Up-Counting δηλαδή, ο εσωτερικός μετρητής μετράει με αύξουσα λογική.

Μέσω λοιπόν, των Registers του περιφερειακού κάνουμε τις απαραίτητες ρυθμίσεις, ώστε ο Timer να λειτουργεί με τον επιθυμητό τρόπο:

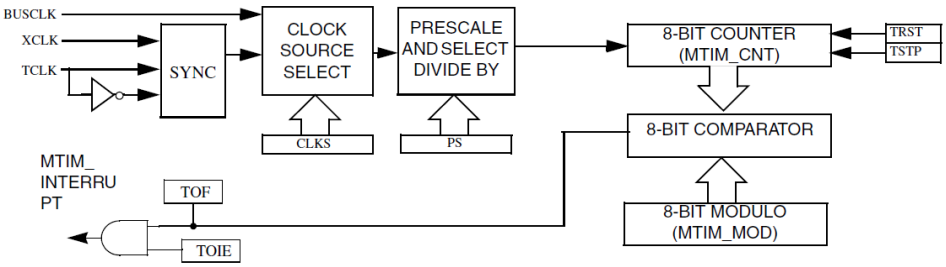
1. Ρυθμίζουμε συχνότητα στην είσοδο του Timer-Μπορούμε να επιλέξουμε είτε εξωτερικό είτε εσωτερικό κρύσταλλο.
2. Ρυθμίσουμε τον prescaller του περιφερειακού-Δηλαδή τοποθετούμε έναν αριθμό ο οποίος διαιρώντας την συχνότητα εισόδου δημιουργεί την τελική συχνότητα λειτουργίας του Timer.
3. Ρυθμίζουμε τον τρόπο λειτουργίας του Timer-Up-Counting/Down-Counting, Interrupt-on-match, λειτουργία σε Low Power Modes, μηδενισμός του μετρητή στο άνω όριο ή λειτουργία “ράμπας”(ο μετρητής ξεκινά από το κατώτατο όριο, μετρά μέχρι το ανώτατο όριο και ξανά μετρά μέχρι το κατώτατο όριο για να ολοκληρώσει τον κύκλο του)

Η βασική φιλοσοφία ενός Timer “θέλει”, έναν μετρητή ο οποίος συνεχώς αυξάνεται να προσπαθεί να φτάσει μια προκαθορισμένη τιμή από τον χρήστη, ώστε είτε να κάνει reset και να μετρήσει από την αρχή είτε να ειδοποιήσει μέσω ενός Interrupt το πρόγραμμα και ύστερα να κάνει reset.

Για το παραπάνω λοιπόν, απαιτείτε ένας καταχωρητής, όπου στον συγκεκριμένο microcontroller ονομάζεται “MOD”, στον οποίο ο χρήστης τοποθετεί έναν αριθμό μεγέθους μικρότερου ή ίσου του αριθμού που δύναται να προσφέρει η ανάλυση του περιφερειακού(8bit=256, 16bit=65535, 32bit=4294967296), καθώς και ένας καταχωρητής ονόματι “CNT” (read-only) ο οποίος συμφώνα με την συχνότητα εξόδου που έχει ρυθμιστεί το περιφερειακό αυξάνει συνεχώς την τιμή του.

Στην περίπτωση λοιπόν που το περιφερειακό έχει ρυθμιστεί να λειτουργεί σε Up-Counting μέτρηση, τότε ο “CNT” καταχωρητή θα αυξάνει συνεχώς την τιμή του έως ότου φθάσει το ανώτατο όριο (το οποίο υπολογίζεται ανάλογα με την ανάλυση του περιφερειακού) ή έως ότου κάποιος μηχανισμός τον μηδενίσει. Ρυθμίζοντας μόνο τον παραπάνω καταχωρητή(CNT) έχουμε ολοκληρώσει μόνο το μισό σκέλος για την λειτουργία του επιθυμητού Timer. Σε αυτό το σημείο έχουμε απλά ενεργοποιήσει ένα μετρητή ο οποίος ατέρμονα μετρά με αύξουσα λογική ανάμεσα στα δύο του όρια.

Το δεύτερο σκέλος απαιτεί να ενημερώσουμε τον μηχανισμό ο οποίος θα μας μηδενίζει το μέτρημα σε πολύ συγκεκριμένο σημείο και άρα θα διαμορφώνει συνεχώς ένα πολύ συγκεκριμένο(το επιθυμητό) χρονικό περιθώριο, από την στιγμή που ξεκινά η μέτρηση έως την στιγμή που θα μηδενίζεται. Αυτό επιτυγχάνεται με τον καταχωρητή “MOD”. Ο εσωτερικός μηχανισμός του περιφερειακού παρακολουθεί συνεχώς τον μετρητή “CNT” και όταν ο τελευταίος γίνει ίσος(match) με τον αριθμό που υπάρχει στον καταχωρητή “MOD” τότε μηδενίζει τον “CNT”, το μέτρημα αρχίζει ξανά από την αρχή και ενημερώνει το πρόγραμμα για την ολοκλήρωση της διαδικασίας μέσω ενός interrupt, εάν έχει ρυθμιστεί.



Με την παραπάνω τεχνική είμαστε σε θέσει να μπορούμε να υπολογίσουμε χρονικά περιθώρια με ακρίβεια και αρά να θέσουμε σε λειτουργία έναν Timer στο εσωτερικό ενός micrcontroller.

Παρακάτω παρουσιάζεται με αριθμούς ο υπολογισμός χρονισμού ενός Timer.

Έστω **16MHz** συχνότητα λειτουργίας του περιφερειακού(συχνότητα εισόδου).
Αν θέσουμε στον Prescaller την τιμή **256** μπορούμε να διαμορφώσουμε την συχνότητα εξόδου, άρα και την συχνότητα με την οποία θα μετράει-αυξάνει ο εσωτερικός μηχανισμό, σε τιμή ίση με 62500Hz (62,5MHz):

F = (Input Frequency) / (Prescaller)
=> F = 16.000.000 / 256
=> F=62500Hz (62,5MHz)

Με τον παραπάνω υπολογισμό λοιπόν έχουμε ρυθμίσει την συχνότητα με την οποία θα μετράει-αυξάνει ο μετρητής στο εσωτερικό του μηχανισμού.
Το επόμενο βήμα είναι να ρυθμίσου τον Timer να μας μετρά χρόνο 1ms. Συνεπώς θα πρέπει να υπολογίσουμε πόσα “χτυπήματα” του εσωτερικού μηχανισμού πρέπει να γίνουν ώστε να πάρουμε αποτέλεσμα **1ms**.

F = 62500
=>T = 1 / F
=>T = 0,000.016 seconds (16μs)

Με απλή διαίρεση λοιπόν βρίσκουμε πως χρειαζόμαστε περίπου 62,5 “χτυπήματα” ώστε να πάρουμε αποτέλεσμα **1ms**

MOD = 1ms / T
=>MOD = 0,001 / 0,000.016
=> **MOD ~ 62.5**

Τα παραπάνω αποτελούν τα βασικά μέρη, γενικότερα των Timer περιφερειακών που συναντάμε στους microcontrollers,πρέπει να αναφερθεί όμως πως σε κάθε εταιρία και σε κάθε οικογένεια microcontroller, εφαρμόζονται επιπλέον δυνατότητες στο συγκεκριμένο περιφερειακό ώστε να γίνει πιο ευέλικτο και πιο “δυνατό”.

2.3.1 FTM (FlexTimer Module)

Η FTM μονάδα αποτελείται από 2 μηχανισμούς με 8 κανάλια timer ο καθένας και αποτελεί μια εκ των δυο περιφερειακών μονάδων Timer του microcontroller. Η συγκεκριμένη περιφερειακή μονάδα είναι και η πιο “δυνατή” στον συγκεκριμένο microcontroller, καθώς εκτός από την ανάλυση του εσωτερικού μηχανισμού της (16bit Counter), έχει και μια πληθώρα δυνατοτήτων όπως input capture,
output compare, και δυνατότητα δημιουργίας PWM(pulse with modulation).
Η FTM περιφερειακή μονάδα δύναται να κάνει πολλά πράγματα παραπάνω από έναν απλό Timer, για το λόγο αυτό, έχει πάρει και το όνομα FlexTimer από τους κατασκευαστές, αυτό δεν σημαίνει όμως πως δεν κάνει και ότι ακριβώς κάνει οποιοσδήποτε Timer. Με το παραπάνω, πρέπει να γίνει αντιληπτό πως και η FTM μονάδα λειτούργει με την βασική αρχή λειτουργίας όλων των Timer, δηλαδή βάση ενός μετρητή ο οποίος δουλεύει ατέρμονα έως ότου να τερματίσει στο ανώτατο όριο ή έως ότου κάποιος μηχανισμός τον μηδενίσει, καθώς επίσης και τον μηχανισμό (πχ καταχωρητής MOD) ο οποίος βοηθάει στην δημιουργία χρονικών ορίων και άρα συγκεκριμένων συχνοτήτων εξόδου.
Στην επόμενη σελίδα παρουσιάζεται το Block Diagram του συγκεκριμένου περιφερειακού.

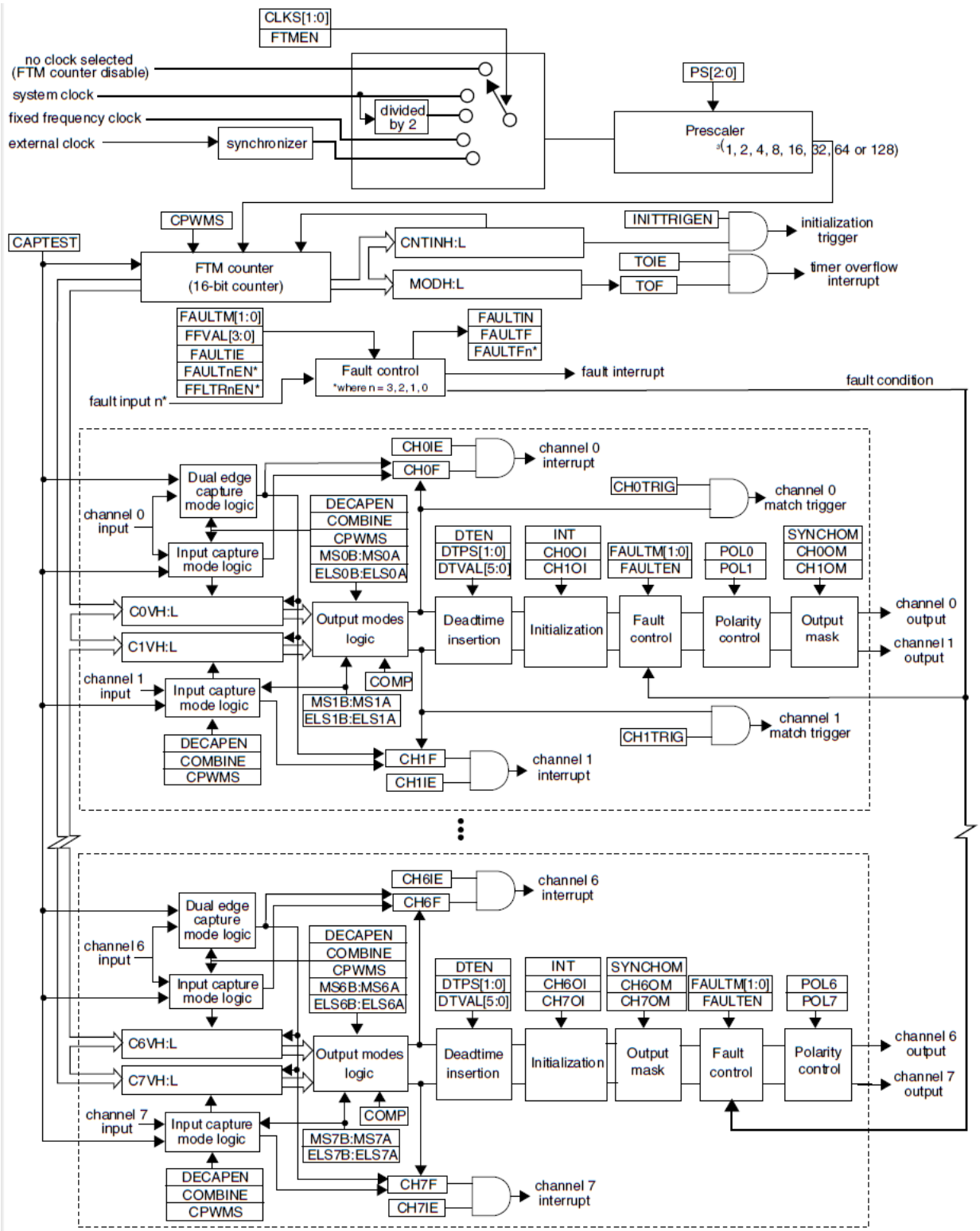


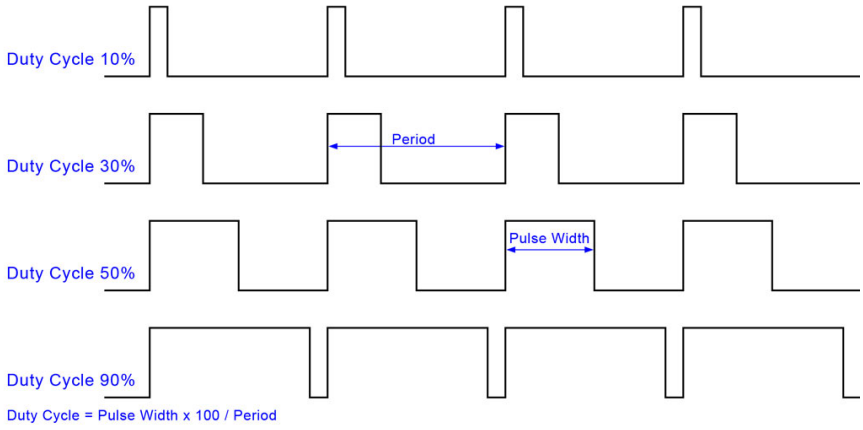
Figure 12-1. FTM block diagram

Παρακάτω περιγράφονται αναλυτικά μερικές από τις επιπλέον δυνατότητες του συγκεκριμένου Timer περιφερειακού που το καθιστούν αρκετά πιο “ενισχυμένο” από τα άλλα:

1. *Input Capture:* Ο συγκεκριμένος τρόπος λειτουργίας δίνει την δυνατότητα στο κανάλι του Timer στο οποίο εφαρμόζεται, να παρακολουθεί την κατάσταση από ένα pin του microcontroller. Όταν το λογικό επίπεδο του pin ταιριάξει με ένα προκαθορισμένο, από τον προγραμματισμό των registers επίπεδο, τότε ο FTM μηχανισμός αποθηκεύει την τιμή στην οποία βρίσκεται εκείνη την στιγμή ο μετρητής (CNT) και παράλληλα δίνεται η δυνατότητα ενημέρωσης του προγράμματος για αυτή την εξέλιξη μέσω ενός interrupt.
   Η παραπάνω δυνατότητα βοηθάει σε περιπτώση που η ταχύτητα ελέγχου ενός pin είναι αρκετά κρίσιμη ώστε απαιτείται παρακολούθηση σε hardware επίπεδο, επίσης όταν απαιτείτε καταγραφή του χρόνου εκείνου που μεσολάβησε έως ότου η κατάσταση από το pin αλλάξει σε μια προκαθορισμένη.

2. *Output Compare:* Με αυτό τον τρόπο λειτουργίας δίνεται η δυνατότητα να προγραμματίσουμε τον τρόπο λειτουργίας-συμπεριφορά ενός pin του microcontroller συμφώνα με κάποια προγραμματιζόμενα χρονικά περιθώρια. Πιο συγκεκριμένα θέτοντας ένα κανάλι σε λειτουργία output compare μπορούμε να προγραμματίσουμε την ένα χρονικό διάστημα, στο πέρας του οποίου, το pin μας να αλλάξει το λογικό του επίπεδο σύμφωνα με τον προγραμματισμό που του έχουμε κάνει. Επίσης μας δίνεται η δυνατότητα να αλλάζουμε επαναλαμβανόμενα την κατάσταση του pin(Toggle) σε τακτά χρονικά διαστήματα ρυθμιζόμενα από τον προγραμματισμό του μηχανισμού.

3. *PWM(Pulse with Modulation):* Ένα από τα επιπλέον χαρακτηριστικά του FTM είναι η δυνατότητα εφαρμογής PWM στα κανάλια του. Με αυτό τον τρόπο μπορούμε να προγραμματίσουμε κάποιο από τα pin του microcontroller να λειτουργεί ως PWM pin. Το PWM είναι μια τεχνική κατά την οποία μπορούμε να εναλλάσσουμε το λογικό επίπεδο ενός pin με υψηλό ρυθμό διαμορφώνοντας έτσι μια συγκεκριμένη "ποσότητα ενέργειας" στο pin. Πιο συγκεκριμένα ρυθμίζοντας το PWM σε μια συγκεκριμένη συχνότητα διαμορφώνουμε την αντίστοιχη περίοδο στην όποια ο μηχανισμός μας θα δουλεύει, πάνω σε αυτές τις περιόδους, η παραπάνω τεχνική μας επιτρέπει να ορίζουμε την διάρκεια της περιόδου για την οποία το pin μας θα εμφανίζει λογικό επίπεδο 1 (Hi), συνεπώς και την διάρκεια στην οποία το pin μας θα εμφανίζει λογικό επίπεδο 0 (Low). Με αυτό τον τρόπο είμαστε σε θέσει να ορίζουμε σε τιμές επί της εκατό του χρόνου που ένα λογικό επίπεδο θα εμφανίζεται στο pin. Για να γίνει πιο κατανοητή η παραπάνω τεχνική ας παρατηρήσουμε τις παρακάτω γραφικές παραστάσεις.



2.3.2 MTIM (8-Bit Modulo Timer)

Ο MTIM αποτελεί μια περιφερειακή μονάδα Timer που βρίσκεται στον συγκεκριμένο microcontroller στην πιο απλή εκδοχή της. Έχοντας μόλις τέσσερεις (4) καταχωρητές αποτελεί τον πιο κοινό και πιο εύχρηστο Timer που συναντάμε σε αυτήν την οικογένεια microcontroller. Σαφώς και αυτό το περιφερειακό, όντας ένας απλός Timer, λειτουργεί βάση της βασικής αρχής λειτουργίας των Timer όπως αυτή περιγράφηκε κατά την εισαγωγή.

Ο MTIM χρησιμοποιεί 8bit ανάλυση μετρητή στον εσωτερικό του μηχανισμό και σε αντίθεση με τον FTM μπορεί να λειτούργει σε καταστάσεις χαμηλής κατανάλωσης ενέργειας-LPM(Low Power Modes).

Με αυτό τον τρόπο ο MTIM χρησιμοποιείτε σε πολλές περιπτώσεις για να επαναφέρει το σύστημα από καταστάσεις LPM.

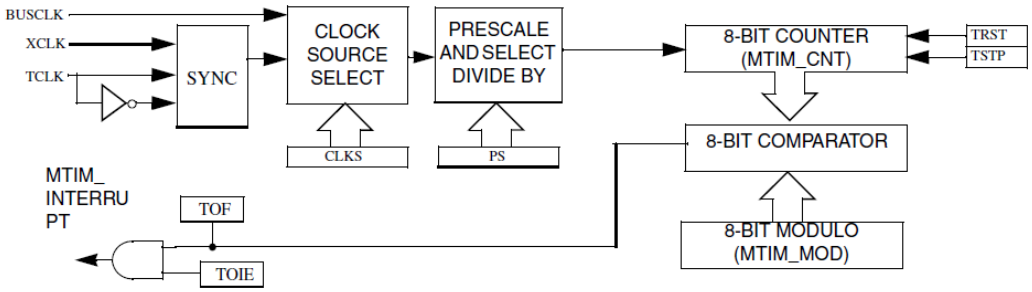


Figure 13-1. Modulo timer (MTIM) block diagram

2.4 LAB 4 – Serial Communication (SCI / SPI)

Στην ενότητα αυτή θα ασχοληθούμε με 2 περιφερειακά που βοηθούν το σύστημα μας να επικοινωνεί με τον έξω κόσμο, αυτά είναι η SCI(UART) και το SPI. Στα Embedded συστήματα, η ανάγκη για επικοινωνία, εμφανίζεται πολύ συχνά όταν θέλουμε να στείλουμε ή να πάρουμε εντολές από άλλα συστήματα, όταν πρέπει να μεταφερθεί πληροφορία από το ένα σύστημα στο άλλο, καθώς και στην περίπτωση του Debug. Καθώς μιλάμε για “επικοινωνία” πρέπει να αναφερθεί πως παρόμοιος με την ανθρώπινη επικοινωνία και εδώ, απαιτούνται πάντα, τουλάχιστον 2 μονάδες-συστήματα για να επιτευχθεί η παραπάνω διαδικασία. Τα σύστημα επικοινωνίας στα Embedded συστήματα κατηγοριοποιούνται σε δύο(2) μεγάλες κατηγορίες, αυτές είναι, η παράλληλη επικοινωνία, η οποία αποτελεί και την απαρχή της διαδικασίας επικοινωνίας στους microcontrollers, και η σειριακή επικοινωνία, η οποία αναπτύχθηκε λίγο αργότερα και η οποία χρησιμοποιείτε κατά κόρο στα σύγχρονα συστήματα.

Στην παράλληλη επικοινωνία, έχουμε δια-μεταγωγή δεδομένων με την πλευρά του συνομιλητή, δια μέσου πολλών διαύλων επικοινωνίας παράλληλα, δηλαδή για να φέρουμε πληροφορία μεγέθους 8Bit θα χρειαζόμασταν 8 διαφορετικά καλώδια ώστε να επικοινωνήσουμε με την απέναντι πλευρά. Η παράλληλη επικοινωνία κατηγοριοποιείτε και αυτή με την σειρά της συμφώνα με μέγεθος της πληροφορίας που μπορεί να μετάγει, συνεπώς συναντάμε πρωτόκολλα παράλληλης επικοινωνίας με ανάλυση 8bit,16bit,32bit και εν συνεχεία, ανάλογα με το εκάστοτε σύστημα συναντάμε και 128bit παράλληλης επικοινωνίας ή ακόμα και 256bit.

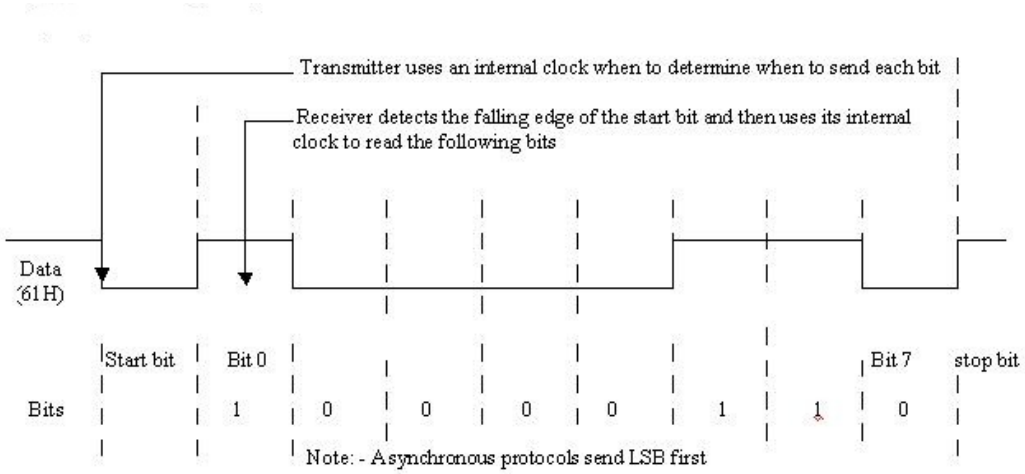
Στην σειριακή επικοινωνία τα πράγματα είναι πιο περίπλοκα και υπάρχουν πιο πολλές παραλλαγές καθώς και μια πληθώρα πρωτοκόλλων που τα συνοδεύει. Όπως λέει και το όνομα αυτού του είδους επικοινωνίας, τα δεδομένα μεταφέρονται με σειριακό τρόπο, δηλαδή μια πληροφορία 8Bit θα σταλθεί σειριακά σε 8 διαφορετικούς χρόνους .Όπως μπορεί να γίνει αντιληπτό αυτή η διαδικασία είναι πιο χρονοβόρα από την παράλληλη δια-μεταγωγή δεδομένων, όμως απαιτεί τον ελάχιστο δυνατό αριθμό διαύλων, καθώς θα χρειαστούμε μόνο ένα καλώδιο για να πραγματοποιήσουμε την παραπάνω επικοινωνία.

Και τα δύο περιφερικά σύστημα(UART-SPI) τα οποία αναφέρονται στην ενότητα αυτήν, είναι σειριακά πρωτόκολλα επικοινωνίας και για το λόγο αυτό ας αναλύσουμε περεταίρω την φιλοσοφία της σειριακή επικοινωνίας.

Όπως αναφέρθηκε και νωρίτερα η σειριακή επικοινωνία, είναι η πλέον διαδεδομένη στις σύγχρονες τεχνολογίες καθώς απαιτεί μικρότερο φυσικό όγκο hardware, αποτελεσματικότητα σε μεγαλύτερες αποστάσεις και δυνατότητα επικοινωνίας με περισσότερους από μία μονάδες με το ίδιο φυσικό επίπεδο.

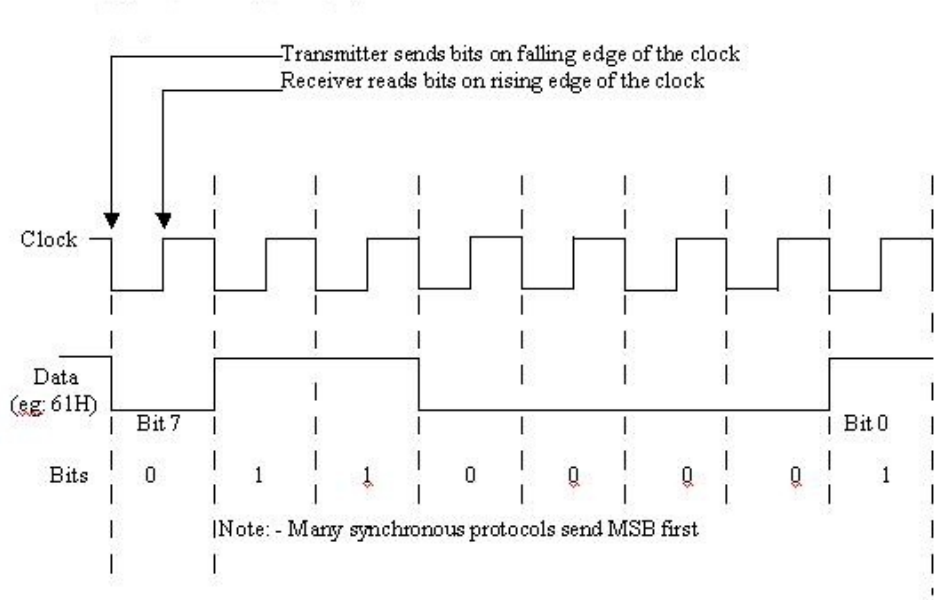
Όπως κάθε μέθοδο μεταφοράς δεδομένων, η σειριακή επικοινωνία απαιτεί επίσης το συντονισμό μεταξύ του αποστολέα και του δέκτη. Για παράδειγμα, πότε να ξεκινήσει η μετάδοση και πότε να τελειώσει, πού ένα συγκεκριμένο bit ή byte τελειώνει και που ένα άλλο αρχίζει, πότε ο δέκτη είναι σε θέση για να λάβει δεδομένα και πότε είναι σε κατάσταση busy, και ούτω καθεξής. Στις σειριακές επικοινωνίες υπάρχουν δύο ειδών συντονισμοί, ο σύγχρονος και ο ασύγχρονος. Το τελευταίο αποτελεί και τη βασική κατηγοριοποίηση των σειριακών επικοινωνιών γενικότερα.

Ασύγχρονη επικοινωνία: Η ασύγχρονη επικοινωνία είναι η πιο συνηθισμένη και η πιο απλή, απαιτεί μόνο ένα καλώδιο ανά κατεύθυνση, χαρακτηρίζεται από αργές ταχύτητες μεταφοράς δεδομένων και, εν σχέση με την σύγχρονη επικοινωνία, έχει περισσότερες πιθανότητες εμφάνισης σφάλματος. Στον συγκεκριμένο τύπο επικοινωνίας δεν υπάρχει κανένα είδος handshake και για το λόγο αυτό κάθε πλευρά μπορεί να στέλνει πληροφορία όποτε θέλει χωρίς πρώτα να ενημερώσει την απέναντι πλευρά για την αρχή και το τέλος του μηνύματος ή για το μέγεθος της πληροφορίας. Εδώ βλέπουμε δύο σήματα ονόματι Tx(Transmit signal) και Rx (Receive signal) τα οποία κάνουν την μεταφορά της πληροφορίας από και προς τις δύο πλευρές.
Στην *εικόνα 1* της επόμενης σελίδας παρουσιάζεται η μεταφορά πληροφορίας από τον αποστολέα στον παραλήπτη. Με την ασύγχρονη επικοινωνίας δεν είναι εφικτή η σύνδεση πολλών μονάδων δια μέσω του ίδιου φυσικού επιπέδου. Περισσότερες πληροφορίες σχετικά με την ασύγχρονη επικοινωνία θα δούμε στην ανάλυση του περιφερειακού UART που αναφέρεται παρακάτω.



Εικόνα 1.

Σύγχρονη επικοινωνία: Στην σύγχρονη επικοινωνία τα δεδομένα μεταφέρονται με έναν δίαυλο επικοινωνίας ανά κατεύθυνση, και παράλληλα σε ένα δεύτερο καλώδιο υπάρχει ένα σήμα, που συχνά ονομάζεται Clock (CLK), που συντονίζει την διαδικασία μεταφοράς, δηλαδή ενημερώνει συνεχώς τον δέκτη πότε αρχίζει και πότε τελειώνει κάθε bit από την λέξη που μεταφέρεται εκείνη την στιγμή. Στην *εικόνα 2* παρουσιάζεται ένα τυπικό διάγραμμα στο οποίο βλέπουμε την διαδικασία αποστολής δεδομένων με σύγχρονο τρόπο. Με αυτό τον τρόπο επικοινωνίας γίνετε εφικτή η δια-μεταγωγή δεδομένων με μεγαλύτερες ταχύτητες καθώς και εκμηδενίζονται οι πιθανότητες σφάλματος κατά την μεταφορά των δεδομένων. Όπως μπορούμε να καταλάβουμε στις σύγχρονες επικοινωνίες υπάρχει ένα σφιχτό handshake ανάμεσα στις πλευρές που επικοινωνούν, γεγονός που μας επιτρέπει να τις εφαρμόζουμε σε πιο κρίσιμα συστήματα, όπως παραδείγματος χάρη, στην μεταφορά διευθύνσεων ανάμεσα στα συστήματα(κάρτες γραφικών, κάρτες RAM, σκληρός δίσκος) του ηλεκτρονικού υπολογιστή. Περισσότερες πληροφορίες σχετικά με την ασύγχρονη επικοινωνία θα δούμε στην ανάλυση του περιφερειακού UART που αναφέρεται παρακάτω.



Εικόνα 2.

2.4.1 SCI –Serial Communication Interface (UART)

Η SCI περιφερειακή μονάδα ή αλλιώς UART (Universal Asynchronous Receiver Transmitter) αποτελεί τμήμα των microcontroller το οποίο επιτρέπει την σειριακή επικοινωνία του συστήματος με τον έξω κόσμο, δηλαδή με άλλα περιφερειακά chips και συστήματα. Το περιφερικό SCI που στο εξής θα αναφέρεται ως UART αποτελεί ένα σύστημα ασύγχρονης σειριακής επικοινωνίας στους microcontroller το οποίο χρησιμοποιεί ένα αγωγό μετάδοσης πληροφορίας ανά κατεύθυνση. Συνεπώς με 2 αγωγούς μετάδοσης η UART αποτελεί τον απλούστερο τρόπο επικοινωνίας σε ένα Embedded System. Στην ασύγχρονη επικοινωνία, συνεπώς και στην UART, δεν χρησιμοποιείτε σήμα συντονισμού της διαδικασίας επικοινωνίας ανάμεσα στις δύο πλευρές, γεγονός που καθιστά το συγκεκριμένο είδος επικοινωνίας ιδανικό για μικρές αποστάσεις και γρήγορη υλοποίηση.

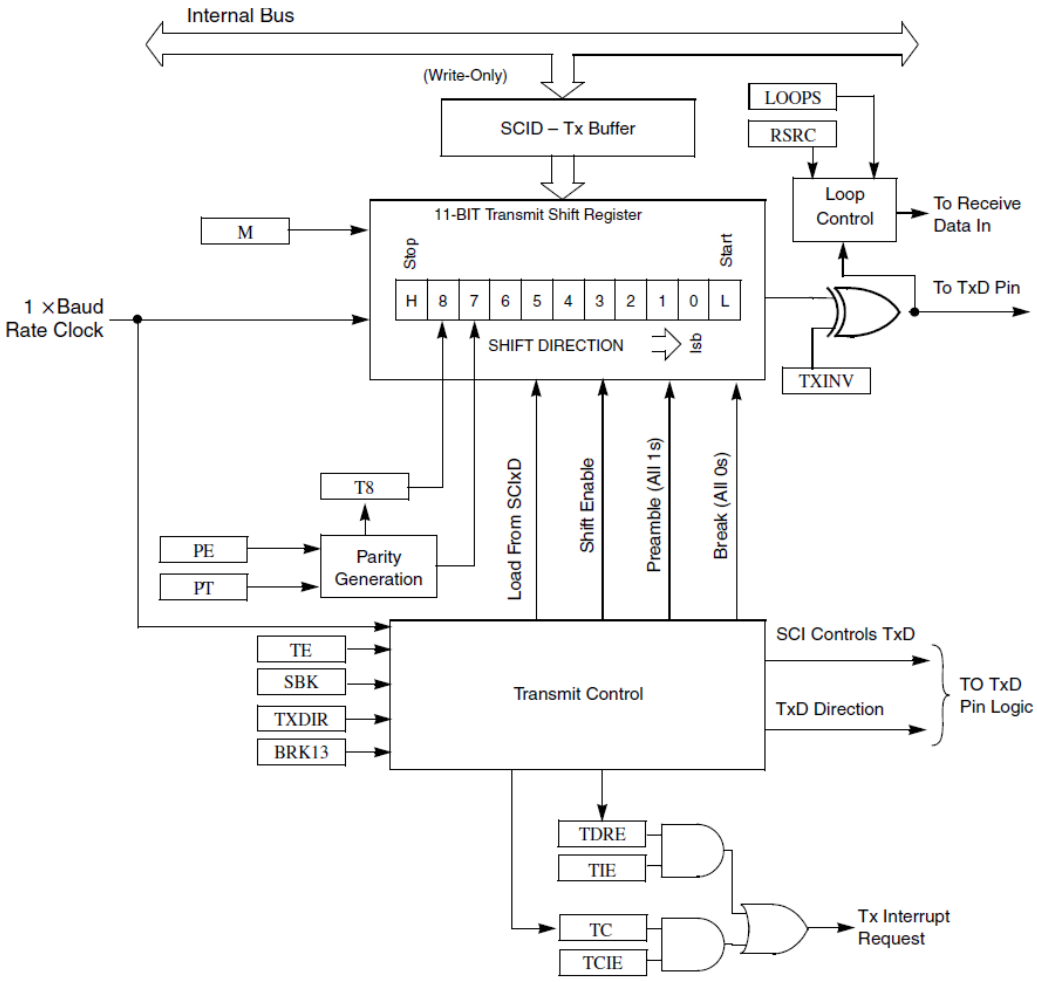


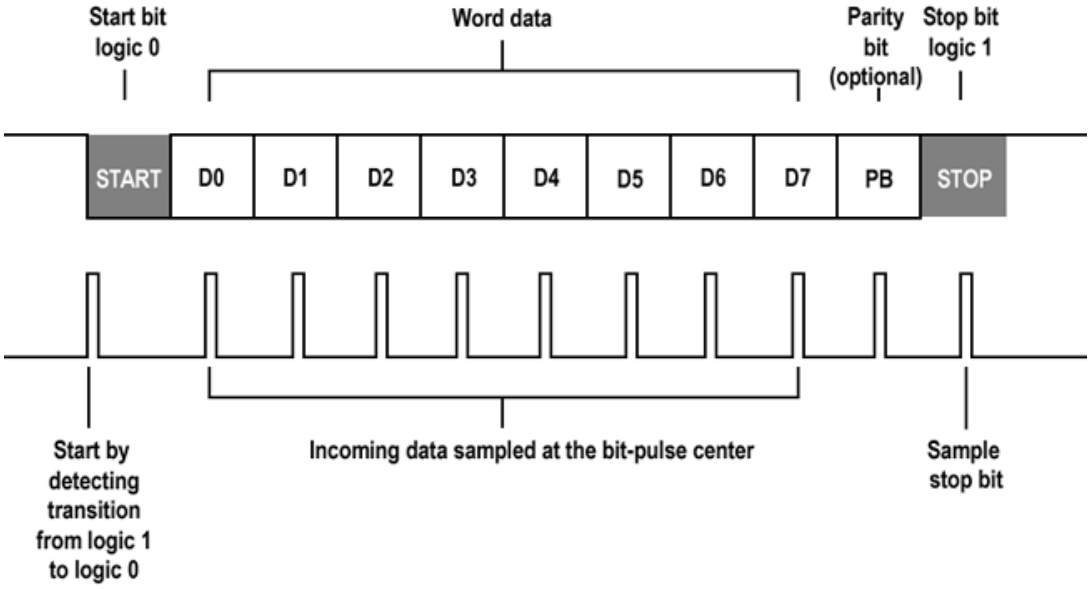
Figure 15-1. SCI transmitter block diagram

Η UART χαρακτηρίζεται από 5 βασικές παραμέτρους. Οι παρακάτω παράμετροι απαιτούνται για την υλοποίησή μιας τυπικής σύνδεσης σειριακής επικοινωνίας τύπου UART ανάμεσα σε 2 πλευρές:

1. **Baud Rate:** αποτελεί το βασικότερο χαρακτηριστικό της UART επικοινωνίας και χαρακτηρίζει την ταχύτητα με την οποία τα Bits ταξιδεύουν πάνω στον αγωγό ανά δευτερόλεπτο. Πιο συγκεκριμένα η έννοια Baud Rate, χαρακτηρίζει τον ρυθμό εναλλαγής της κατάστασης του λογικού επιπέδου 0 ή 1 πάνω στον αγωγό σε ένα δευτερόλεπτο. Ο όρος Baud Rate στην προκειμένη περίπτωση είναι αλληλένδετος με τον όρο Bit per Seconds(bps).Συνεπώς, αν υποθέσουμε πως η ταχύτητα του Baud Rate είναι ρυθμισμένη στα 115200, εννοούμε πως πραγματοποιείτε μεταφορά δεδομένων με ταχύτητα 115200 bits το δευτερόλεπτο.
2. **Number of Bits:** χαρακτηρίζει τον αριθμό των Bits ο οποίος αποστέλλετε κάθε φορά από την μια πλευρά στην άλλη. Αυτή η παράμετρος παίρνει τιμές από 5 έως 8, με πιο κοινές τις τιμές 7 και 8, καθώς για την μετάδοση ενός ASCII απαιτούνται 7 bits και για την μετάδοση ενός τυπικού byte απαιτούνται 8 bits.
3. **Parity:** ή αλλιώς Parity check είναι μια λειτουργία κατά την οποία επαληθεύεται από τον παραλήπτη η σωστή μετάδοση της πληροφορίας που στάλθηκε από τον αποστολέα, μέσω ενός bit που ονομάζεται Parity Bit. Με αυτή την τεχνική ο αποστολέας καταμετρά τα bit με λογικό επίπεδο 1 και καταχωρεί έναν αριθμό 1 ή 0 στη θέση του Parity Bit ο οποίος σχετίζεται τον αριθμό των 1 που καταμετρήθηκαν. Με τον τρόπο αυτό ο παραλήπτης μπορεί να επαληθεύσει την παραπάνω συσχέτιση κατά την παραλαβή της πληροφορίας και να αποφανθεί για την εγκυρότητά της. Η παράμετρος Parity μπορεί να ρυθμιστεί ως even, odd, mark and space. Αν η παράμετρος Parity έχει ρυθμιστεί ως even(άρτιος) και ο αριθμός των άσσων(1) μέσα στο byte είναι άρτιος τότε το Parity Bit παίρνει την τιμή 0, αλλιώς παίρνει την τιμή 1. Αν η παράμετρος Parity έχει ρυθμιστεί ως odd(περιττός) και ο αριθμός των άσσων(1) μέσα στο byte είναι περιττός, το Parity Bit παίρνει την τιμή 0,αλλιώς παίρνει την τιμή 1.Στην περίπτωση που η παράμετρος Parity έχει ρυθμιστεί ως mark ή space το Parity Bit θα πάρει τιμή 1 ή 0 ανάλογα με τον αριθμό των άσσων(1) στο byte, δηλαδή αν οι άσοι μέσα στο Byte είναι περιττός αριθμός τότε το Parity Bit θα πάρει την τιμή 1, αντιστοίχως αν οι άσσοι στο byte είναι άρτιος αριθμός τότε το Parity Bit θα πάρει τιμή 0.
4. **Stop Bits:** Το χαρακτηριστικό αυτό σηματοδοτεί το ελάχιστο κενό(χρόνου) που θα πρέπει να υπάρχει, κατά την αποστολή, ανάμεσα στα bytes. Η παράμετρος αυτή μπορεί να πάρει τιμές 1

και 2. Σε κάποιες περιπτώσεις όπου το hardware το υποστηρίζει, υπάρχει η δυνατότητα χρήσης μισού bit ώστε να υλοποιείτε η λογική του 1,5 και του 2,5 Stop Bits. Συνεπώς με τον όρο Stop Bits χαρακτηρίζεται το κενό χρόνου, μετρούμενο σε bits(δηλαδή χρόνος μετάδοσης bits), που θα πρέπει να μεσολαβεί από byte σε byte.

5. **Endianess:** η παράμετρος αυτή μπορεί να ρυθμιστεί ως LSB ή MSB και ρυθμίζει τον τρόπο με τον οποίο ο αποστολέας θα τοποθετεί το Byte πάνω στον αγωγό για αποστολή. Κατά την ρύθμιση της παραμέτρου Endianess ως LSB(Less Significant Bit) ο αποστολέας στέλνει τα bit του byte ξεκινώντας από το λιγότερο σημαντικό , ενώ κατά την ρύθμιση της παραμέτρου ως MSB(Most Significant Bit), ο αποστολέας στέλνει τα bit ξεκινώντας από αυτά με την μεγαλύτερη σημασία.



2.4.2 SPI - Serial Peripheral Interface

Το SPI, δηλαδή η περιφερειακή μονάδα σειριακής επικοινωνίας όπως αναφέρει και το όνομά του, αποτελεί την δεύτερη πιο διαδεδομένη μορφή σειριακής επικοινωνίας στα Embedded Systems μετά από την UART. Το SPI είναι ένα σύγχρονο πρωτόκολλο επικοινωνίας και συνεπώς απαιτεί το συγχρονισμό των δύο πλευρών επικοινωνίας, μέσω ενός αγωγού ο οποίος συντονίζει την διαδικασία της επικοινωνίας. Για την ιστορία να αναφέρουμε πως το πρωτόκολλο SPI εφευρέθηκε και υλοποιήθηκε για πρώτη φορά από την Motorola και αποτελεί προσωπική κληρονομιά της εταιρείας, η ονομασία του πρωτοκόλλου είναι κατοχυρωμένη και δεν μπορεί να χρησιμοποιηθεί από άλλους. Για το λόγο αυτό, το παραπάνω πρωτόκολλο , το συναντάμε και με την ονομασία "Four wire serial bus" δηλαδή σειριακός δίαυλος επικοινωνίας με 4 αγωγούς.

Στο πρωτόκολλο SPI συναντάμε 4 σήματα τα οποία είναι τα παρακάτω:

- **MOSI - Master out Slave in**
- **MISO - Master in Slave out**
- **SCLK - SPI Clock**
- **CS - Chip Select**

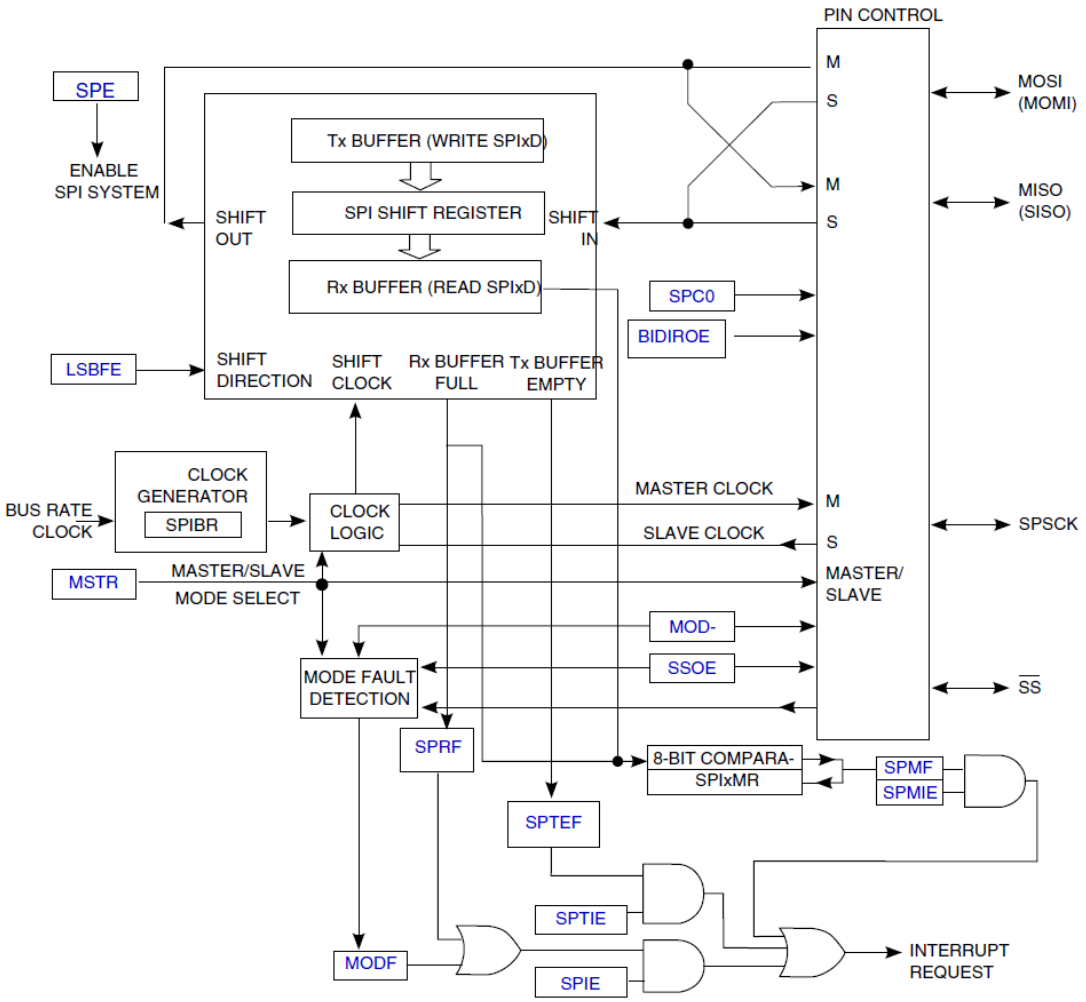


Figure 16-2. SPI Module Block Diagram without FIFO

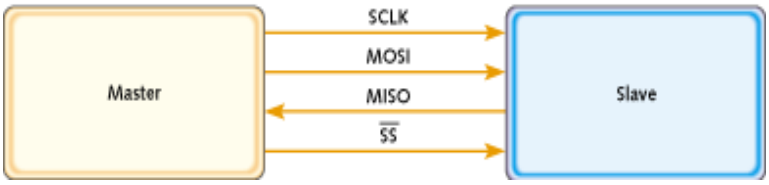
Το SPI, όπως και κάθε πρωτόκολλο επικοινωνίας απαιτεί δύο(2) πλευρές, που στην προκειμένη περίπτωση χωρίζονται και ονομάζονται αντίστοιχα σε Master και Slave. Καθώς μιλάμε για ένα σύγχρονο πρωτόκολλο επικοινωνίας, απαιτείται ένα σύστημα συντονισμού της διαδικασίας μεταφοράς πληροφορίας από την μια πλευρά στην άλλη, που στο συγκεκριμένο περιφερειακό (SPI) υπεύθυνος για την υλοποίηση της διαδικασίας αυτής είναι ο Master. Κατά την μεταφορά δεδομένων λοιπόν ο Master εμφανίζει ένα σήμα που ονομάζεται Clock(SCLK), με το οποίο συντονίζεται ολόκληρη η διαδικασία, καθώς κατά την εναλλαγή του λογικού επιπέδου του Clock(SCLK) από 0 σε 1 ο Slave είναι σε θέση να "διαβάζει", με απόλυτα συντονισμένο τρόπο, ένα προς ένα τα Bit που καταφθάνουν από το εκάστοτε Byte.
Το SPI πρωτόκολλο επιτρέπει την διασύνδεση του Master, με παραπάνω από ένα Slave, γεγονός που καθιστά το SPI ιδανικό για τα Embedded Systems. Το παραπάνω πραγματοποιείτε χάρη στο CS(εμφανίζεται και ως SS - Slave Select) σήμα το οποίο είναι υπεύθυνο, όπως αναφέρει και το όνομά του, για την επιλογή του chip στο οποίο θέλουμε να "μιλήσουμε", δηλαδή την Slave μονάδα με την οποία είναι συνδεδεμένος ο Master. Το CS σήμα είναι ένα ή περισσότερα GPIO το\α οποίο\α συνδέει τον Master με κάθε ένα από τις Slave μονάδες, με αυτό τον τρόπο ο Master είναι σε θέσει να μπορεί να επιλέξει σε ποιά μονάδα από τις Slave θα μιλήσει, κατεβάζοντας το λογικό επίπεδο του εκάστοτε CS σήματος από 1 σε 0.

Πιο αναλυτικά περιγράφονται παρακάτω τα σήματα το πρωτοκόλλου SPI.

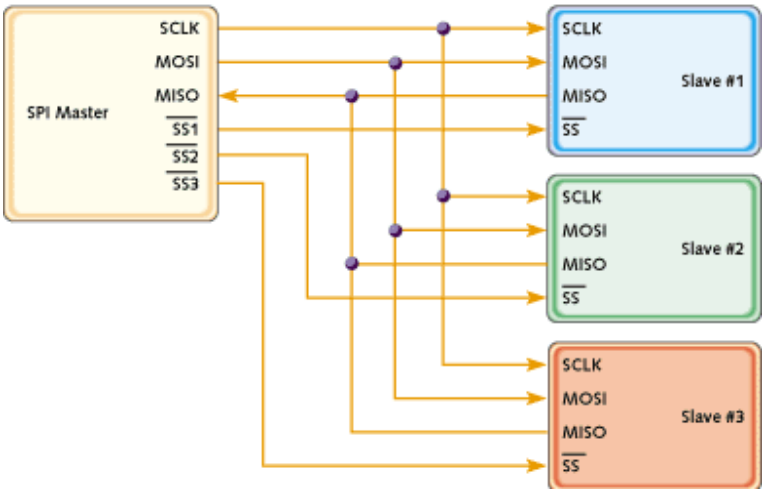
- **MOSI:** Όπως αναφέρει και η περιγραφή του, το συγκεκριμένο σήμα είναι υπεύθυνο για την δια-μεταγωγή δεδομένων από την μονάδα Master στην μονάδα Slave. Κατά την αποστολή ενός byte από τον Master στον Slave σε αυτόν τον αγωγό εμφανίζονται διαδοχικά τα Bits συντονισμένα με το σήμα SCLK που εφαρμόζει ο Master παράλληλα.
- **MISO:** Αντίστοιχα, και το σήμα MISO χρησιμοποιείτε για την μεταφορά δεδομένων. Σε αυτή την περίπτωση το σήμα MISO μεταφέρει πληροφορία από τους Slave στον Master του συστήματος. Για την αποστολή ενός byte από έναν μια Slave μονάδα στον Master απαιτείτε, όπως και πριν, η εφαρμογή του σήματος SCLK παράλληλα με την αποστολή του byte για τον συντονισμό της διαδικασίας.
- **SCLK:** Το σήμα αυτό, είναι υπεύθυνο για τον συντονισμό της επικοινωνίας ανάμεσα στις δύο πλευρές. Κατά την μεταφορά δεδομένων από και προς τις δύο πλευρές, ο Master εμφανίζει

στο συγκεκριμένο σήμα έναν παλμό "κλειδωμένο" σε μία συχνότητα και βάση του συγκεκριμένου παλμού συντονίζονται τα Bit της πληροφορίας που αποστέλλεται. Πιο αναλυτικά ο Master εφαρμόζει κάθε Bit του Byte που θέλει να στείλει σε κάθε φάση του παλμού με λογικό επίπεδο 1 με τη σειρά, με αυτό τον τρόπο ο Slave Είναι σε θέση να συγχρονιστεί με τον παλμό αυτό και να "διαβάσει" με αντίστοιχη λογική, ένα προς ένα τα bit του byte που καταφθάνουν .

- **CS:** ή αλλιώς και SS(Slave Select) είναι ένα απλό GPIO ανάστροφης λογικής το οποίο διασύνδεει μια μονάδα Slave με τον Master με στόχο ο τελευταίος να είναι σε θέση, εφαρμόζοντας λογικό επίπεδο 0, να μπορεί να επιλέξει σε πια από της μονάδες Slave που είναι συνδεδεμένες πάνω του, θα μιλήσει κάθε φορά.



Εικόνα 1. *Απλή σύνδεση SPI με 1 Master και 1 Slave*



Εικόνα 2. *Παράλληλη σύνδεση SPI με 1 Master και 3 Slave*

2.5 LAB 5 –Serial Communication and Analog to Digital converter

Σε αυτή την ενότητα εν συνεχεία της προηγούμενη θα αναλύσουμε ακόμα μια περιφερειακή μονάδα επικοινωνίας που συναντάμε στου microcontroller, την I2C. Επίσης θα παρουσιάσουμε και θα αναλύσουμε τον τρόπο λειτουργίας του περιφερειακού ADC, το οποίο αποτελεί το πιο συνηθισμένο τρόπο ψηφιοποίησης οποιασδήποτε αναλογικής τιμής από τον microcontroller. Στην πρώτη περίπτωση (I2C) αναλύουμε ένα ακόμα σειριακό πρωτόκολλο σύγχρονης επικοινωνίας, το οποίο συναντάτε σε πολύ μεγάλο βαθμό σε κάθε μικρό η μεγάλο, σε δυνατότητες, microcontroller. Το συγκεκριμένο πρωτόκολλο επικοινωνίας (I2C) χρησιμοποιεί 2 σήματα και είναι ιδανικό για τη διασύνδεση μονάδων με χαμηλές ταχύτητες. Στο δεύτερο χρόνο αναλύουμε ένα από τα πιο συνηθισμένα περιφερειακά των microcontrollers και αυτό είναι το ADC ή αλλιώς Analogue to Digital converter. Το παραπάνω περιφερειακό βοηθάει τον microcontroller να "αντιληφτεί" τις αναλογικές τιμές του περιβάλλοντος με το οποίο αλληλεπιδρά. Καθώς οι microcontroller είναι ψηφιακά συστήματα δεν έχουν την δυνατότητα να επεξεργαστούν αναλογικές τιμές, πρόβλημα που έρχεται να λύσει, η συγκεκριμένη περιφερειακή μονάδα.

2.5.1 I2C - (Inter-Integrated Circuit)

Η περιφερειακή μονάδα I2C υλοποιεί ακόμη έναν σύγχρονο τρόπο επικοινωνίας που συναντάμε στους microcontroller. Ευρέως γνωστός και ως IIC ή I^2C το I2C πρωτόκολλο εφευρέθηκε από την εταιρία Philips στις αρχές του 1980 και χρησιμοποιήθηκε πολύ στις εσωτερικές επικοινωνία των περιφερειακών μέσα στις τηλεοράσεις.

Το I2C πρωτόκολλο που υλοποιείτε από την I2C περιφερειακή μονάδα του microcontroller,καθώς αναφερόμαστε σε ένα πρωτόκολλο σύγχρονης επικοινωνίας, απαιτείτε ένας συντονιστής για την σωστή λειτουργία του μηχανισμού μεταφοράς δεδομένων από την μια πλευρά στην άλλη. Για το λόγο αυτό το I2C περιφερειακό ορίζει την μια πλευρά ως Master και την άλλη ως Slave. Στο I2C όπως και στο SPI δίνεται η δυνατότητα να συνδέσουμε παραπάνω από μια Slave μονάδες με μια Master μονάδα, γεγονός που καθιστά το συγκεκριμένο πρωτόκολλο ιδανικό για εφαρμογή σε πολύπλοκα συστήματα που απαιτείτε μεταφορά δεδομένων ανάμεσα σε πολλές μονάδες, όπως οι motherboards, τα κινητά τηλέφωνα και γενικότερα στα Embedded Systems.

Το I2C χρησιμοποιεί μόνο δύο(2) αγωγούς που ονομάζονται SDA(Serial Data) και SCL(Serial Clock) και σε αντίθεση με άλλα σύγχρονα πρωτόκολλα επικοινωνίας, όπως το SPI, δεν απαιτεί κάποιο μηχανισμό για την επιλογής της μονάδας στην οποία θέλει να "μιλήσει" κάθε φορά. Αντί αυτού υλοποιεί την παραπάνω διαδικασία με την χρήση ψηφιακών διευθύνσεων. αυτό σημαίνει πως κάθε σύστημα που είναι σχεδιασμένο να

επικοινωνεί με το συγκεκριμένο πρωτόκολλο(I2C) φέρει στο εσωτερικό μια ψηφιακή διεύθυνση μεγέθους 7Bit που λειτουργεί σαν όνομα για την μονάδα. Συνεπώς για την επιλογή της μονάδα στην οποία ο Master θέλει να μιλήσει, πρέπει να χρησιμοποιεί την αντίστοιχη διεύθυνση του, προκειμένου να "ακούσει" ο εκάστοτε Slave.
Πολύ σημαντικό ρόλο σε μια τέτοιου είδους πολύ σφιχτή επικοινωνία είναι, η δυνατότητα επαλήθευσης(Feedback) προς τον Master για την πορεία της διαδικασίας. Για το λόγο αυτό στο παρών πρωτόκολλο δίνετε προαιρετικά η δυνατότητα στον χρήστη να ενεργοποιήσει μια λειτουργία που ονομάζεται "Acknowledge". Η παραπάνω λειτουργία ενσωματώνει σε κάθε ένα Byte που στέλνει πάνω στον αγωγό μέσω του SDA, ένα επιπλέον παλμό στο SCL σήμα, στη διάρκεια του οποίου η Slave μονάδα απαντάει θετικά ή αρνητικά με 0 ή 1 λογικό επίπεδο αντίστοιχα, ενημερώνοντας με τον τρόπο αυτό τον Master για την εξέλιξη της διαδικασίας.

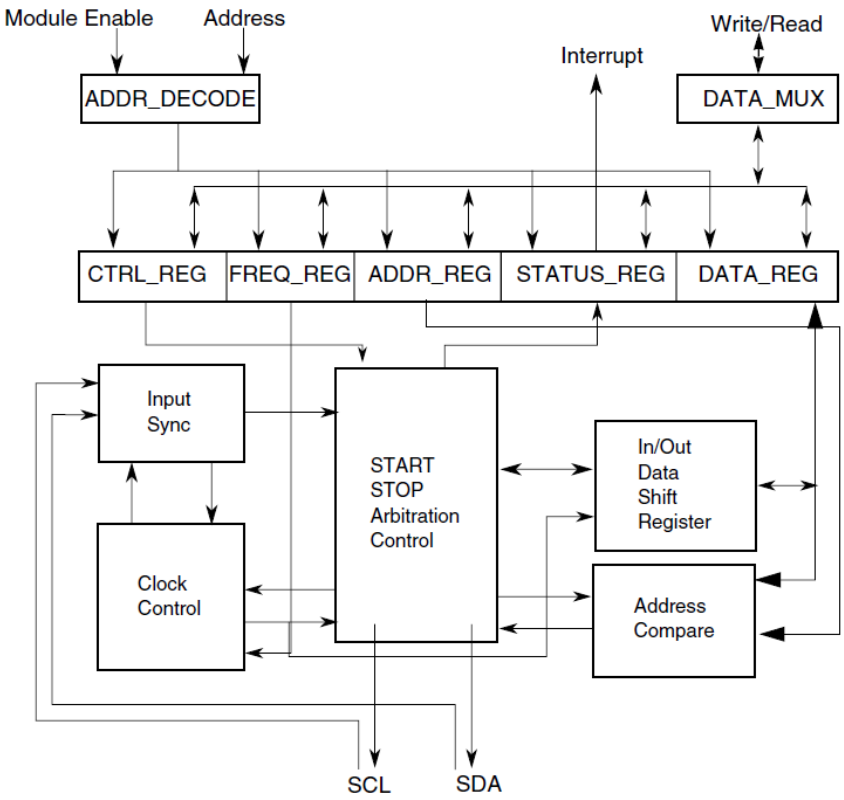
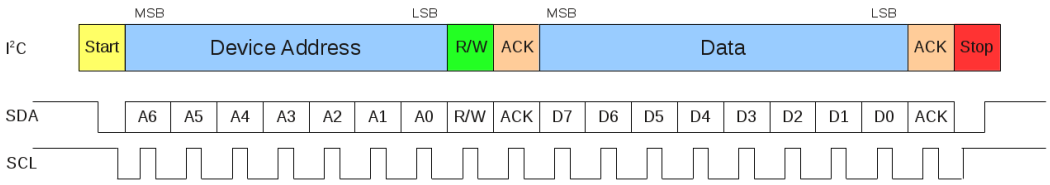


Figure 18-1. I2C Functional block diagram

Λειτουργία

Το συγκεκριμένο περιφερειακό πραγματοποιεί την διαδικασία της επικοινωνίας με ένα αρκετά περίπλοκο τρόπο, ο οποίος αποτελείτε, θα λέγαμε, από τέσσερεις(4) διαφορετικές και πολύ σημαντικές φάσεις (Εικόνα 2).
Οι παρακάτω 4 φάσεις είναι απολύτως απαραίτητες για κάθε επικοινωνία που υλοποιείτε βάση του πρωτοκόλλου I2C και η σειρά εκτέλεσης των φάσεων είναι πολύ συγκεκριμένη:

1. **START Signal**
2. **Slave Address Transmission**
3. **Data Transfer**
4. **STOP Signal**



Εικόνα 2. *Αποστολή μιας φράσης με την χρήση I2C πρωτοκόλλου.*

1. **START Signal (Σήμα ΕΝΑΡΞΗΣ)**
   Αποτελεί τη σήμανση έναρξης μιας νέα αποστολής δεδομένων πάνω στο καλώδιο. Ο συντονιστής της επικοινωνίας Master, όταν θέλει να χρησιμοποιήσει τον αγωγό μεταφοράς δεδομένων για να αποστείλει μια πληροφορία και εφόσον το "κανάλι" είναι ελεύθερο, τότε ξεκινά την διαδικασία αποστολής στέλνοντας το START Signal. Το START Signal είναι μια πολύ συγκεκριμένη κατάσταση, με τον συνδυασμός των λογικών επιπέδων ρεύματος των δυο σημάτων SDA και SCL, όπου σηματοδοτεί όλες τις παράλληλα συνδεδεμένες μονάδες Slave, για την εκκίνηση αποστολής μιας πληροφορίας πάνω στον αγωγό, με αποτέλεσμα να τις ενεργοποιεί και να τις βγάζει από την κατάσταση ηρεμίας(Idle State). START Signal πραγματοποιείτε μόνο όταν το λογικό επίπεδο του σήματος SDA αλλάξει από 1 σε 0, ενώ παράλληλα το λογικό επίπεδο του σήματος SCL είναι 1.

2. **Slave Address Transmission (Αποστολή διεύθυνσης του Slave)**
   Σε αυτή την φάση ο Master αποστέλλει πάνω στον αγωγό την ψηφιακή διεύθυνση της Slave μονάδας με την οποία θέλει να επικοινωνήσει**.**
   Αμέσως μετά το START Signal ο Master στέλνει, μέσω του αγωγού, την 7ψήφια διεύθυνση της μονάδας με την οποία θέλει να επικοινωνήσει, ακολουθούμενη από ένα(1) bit το οποίο ονομάζεται R/W bit. Η 7ψήφια διεύθυνση αποτελεί προϋπόθεση για την επικοινωνία με την αντίστοιχη Slave μονάδα η οποία θα την επιβεβαιώσει και το R/W bit αποτελεί σήμανση για την Slave μονάδα που "ακούει" στην παραπάνω διεύθυνση για το αν θα ενεργοποιηθεί σε λειτουργία διαβάσματος μιας πληροφορίας ή εγγραφής. Συνεπώς, στην περίπτωση που το 8ο bit αυτής της φάσης είναι λογικό 0, τότε ο Master ενημερώνει τον Slave πως ετοιμάζεται να στείλει και άρα με την σειρά του ο Slave πρέπει να διαβάσει την πληροφορία. Στην αντίθετη περίπτωση που το 8ο bit έχει λογικό επίπεδο 1, τότε ο Master ζητά από τον Slave να γράψει-στείλει μια πληροφορία, ώστε ο Master να την διαβάσει.

3. **Data Transfer (Μεταφορά Πληροφορίας)**
   Έπειτα από την σωστή αποστολή της διεύθυνσης και της ρύθμισης R/W πάνω στον αγωγό και έπειτα από την ανταπόκριση κάποιας Slave μονάδας σε αυτή την διαδικασία, το επόμενο βήμα είναι η αποστολή της πληροφορίας. Σε αυτή την φάση και ανάλογα με την ρύθμιση R/W που πραγματοποιήθηκε στο προηγούμενο βήμα, η Master ή η Slave μονάδα αντίστοιχα αποστέλλει, παράλληλα με τους παλμούς που διοχετεύει ο Master πάνω στον αγωγό μέσω του SCL σήματος, την πληροφορία στον παραλήπτη μέσω του αγωγού SDA.

4. **STOP Signal (Σήμα ΤΕΛΟΥΣ)**
   Η φάση αυτή τερματίζει την διαδικασία αποστολής δεδομένων μεταξύ μιας Master και μιας Slave μονάδας καθώς και απελευθερώνει τα κανάλια μεταφοράς SDA και SCL ώστε να μπορούν να χρησιμοποιηθούν για μια νέα μεταφορά δεδομένων. Όπως και το START Signal, έτσι και το STOP Signal πραγματοποιείτε από την εφαρμογή ενός συγκεκριμένου συνδυασμού λογικών καταστάσεων πάνω στα σήματα SDA και SCL. Για να πραγματοποιηθεί λοιπόν STOP Signal θα πρέπει το σήμα SDA να αλλάξει το λογικό του επίπεδο από 0 σε 1, ενώ παράλληλα το λογικό επίπεδο του σήματος SCL είναι 1.

2.5.2 ADC - (Analog-to-Digital Converter)

Από τους μικρότερους έως και τους μεγαλύτερους, σε δυνατότητες, microcontroller στον κόσμο, η περιφερειακή μονάδα ADC αποτελεί αναπόσπαστο κομμάτι των Embedded System. Η συγκεκριμένη περιφερειακή μονάδα όπως αναφέρει και το όνομά της δίνει την δυνατότητα στον microcontroller να μετατρέπει μια αναλογική τιμή σε ψηφιακή.

Στον πραγματικό κόσμο όλα τα σήματα του περιβάλλοντος συναντιούνται σε αναλογικές τιμές, όπως παραδείγματος χάρη ο ήχος, το φώς του ήλιου, η βαρύτητα και πολλά άλλα. Καθώς λοιπόν, οι microcontroller αποτελούν ψηφιακά συστήματα, συνεπώς επεξεργάζονται και λειτουργούν με ψηφιακά σήματα και καθώς η εξέλιξη του κόσμου μας απαιτεί την αλληλεπίδραση τεχνολογίας-περιβάλλοντος , δημιουργείτε η ανάγκη για την εύρεση ενός τρόπου κατανόηση των αναλογικών σημάτων του πραγματικού κόσμου από τους microcontroller. Το παραπάνω πρόβλημα ήρθε να λύσει, από τα πρώτα κιόλας βήματα του ψηφιακού κόσμου, η εφεύρεση του ADC μηχανισμού.

Πιο συγκεκριμένα λοιπόν στους microcontroller πλέον, η περιφερειακή μονάδα ADC είναι αρμόδια για την μετατροπή, μέσω ενός σύνθετου μηχανισμού, οποιασδήποτε αναλογικής τιμής σε ψηφιακή. Η γενική φιλοσοφία αυτού του σύνθετου μηχανισμού στηρίζεται στην επαναλαμβανόμενη και σε πολύ σύντομα χρονικά διαστήματα, λήψη μετρήσεων.

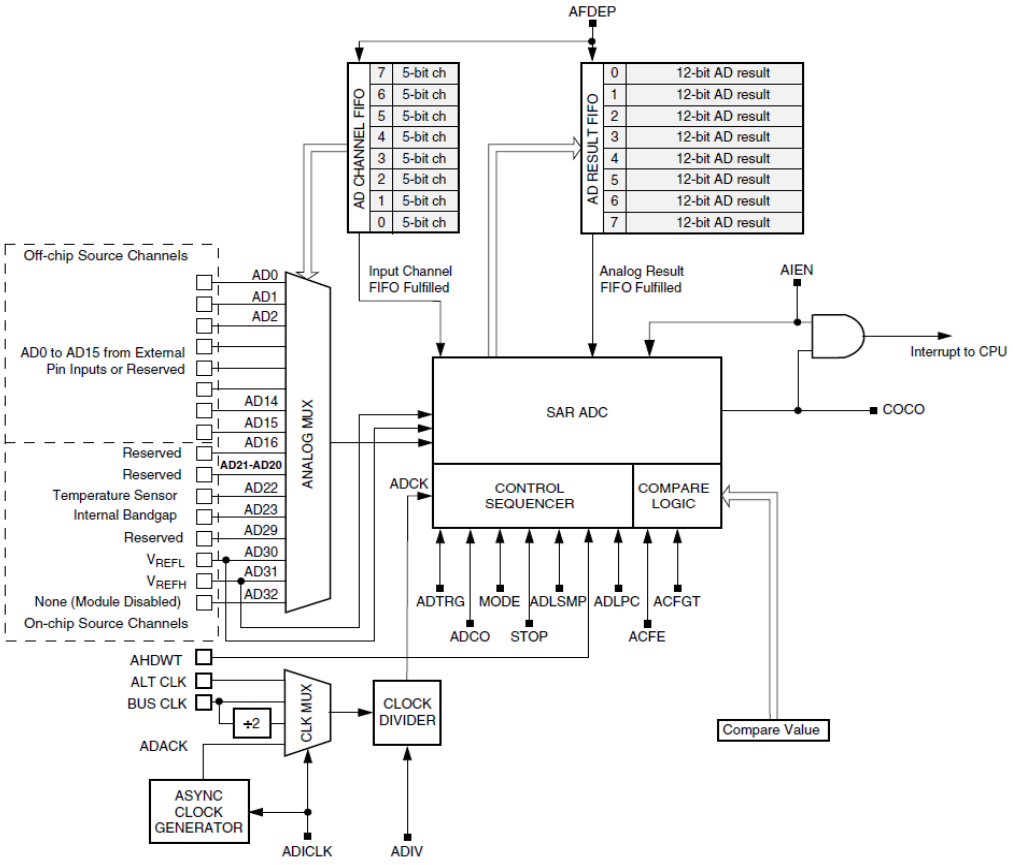
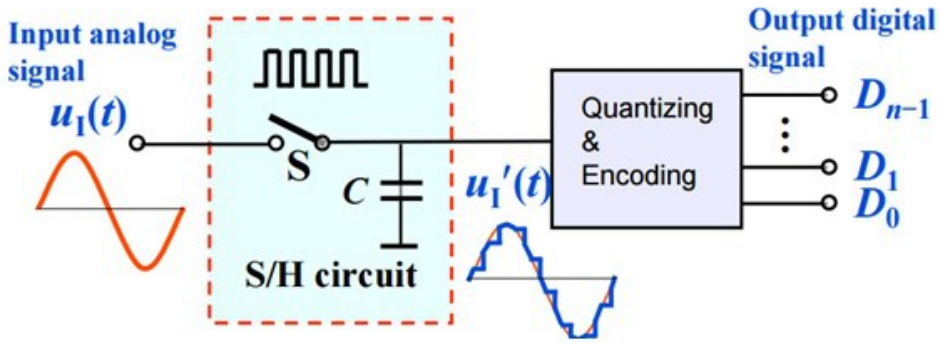


Figure 19-1. ADC Block Diagram

Λειτουργία

Η περιφερειακή μονάδα ADC λειτουργεί σε όλους τους microcontroller βάση μιας πολύ απλής λογικής που θα αναλυθεί παρακάτω. Ο λειτουργία του ADC βασίζεται σε έναν μηχανισμό που περιλαμβάνει έναν πυκνωτή και μια αντίσταση pull-down. Ανά δειγματοληψία επαναλαμβάνεται το φαινόμενο της ακαριαίας φόρτισης του συγκεκριμένου πυκνωτή, με τάση ίση με αυτή στο pin. Έπειτα από την στιγμιαία φόρτιση, ο μηχανισμός αποκόπτεται εντελώς από τον έξω κόσμο συνεπώς και από το pin και η διαδικασία περνά στο επόμενο στάδιο της καταμέτρησης. Εφόσον λοιπόν σύμφωνα με το προηγούμενο βήμα έχουμε τάση στα άκρα του πυκνωτή, εφαρμόζοντας μια pull-down αντίσταση συγκεκριμένης τιμής και αποφορτίζουμε τον πυκνωτή δια του πλην (-) μετρώντας χρόνο έως ότου το φορτίο του πυκνωτή γίνει ίσο με μηδέν (0). Η διαδικασία αυτή είναι γνωστή και ως Sample and Hold.
Ο χρόνος αυτό που μεσολάβησε από την στιγμή που αποκόπηκε ο μηχανισμός από τον έξω κόσμο έως ότου το φορτίο του πυκνωτή γίνει ίσο με μηδέν (0) μας βοηθάει να υπολογίσουμε την τιμή του ADC.
Περνώντας λοιπόν στο επόμενο βήμα, στον εσωτερικό μηχανισμό του ADC σύμφωνα με την τάση αναφοράς που έχει το συγκεκριμένο περιφερειακό υπολογίζεται δια μέσου της απλής μέθοδου των τριών ένας αριθμός από 0-256 αν η ανάλυση είναι 8 bit, 0-1024 αν η ανάλυση είναι 10Bit ο οποίος αντιστοιχεί στην αναλογική τιμή που καταγράφηκε από τα προηγούμενα βήματα.



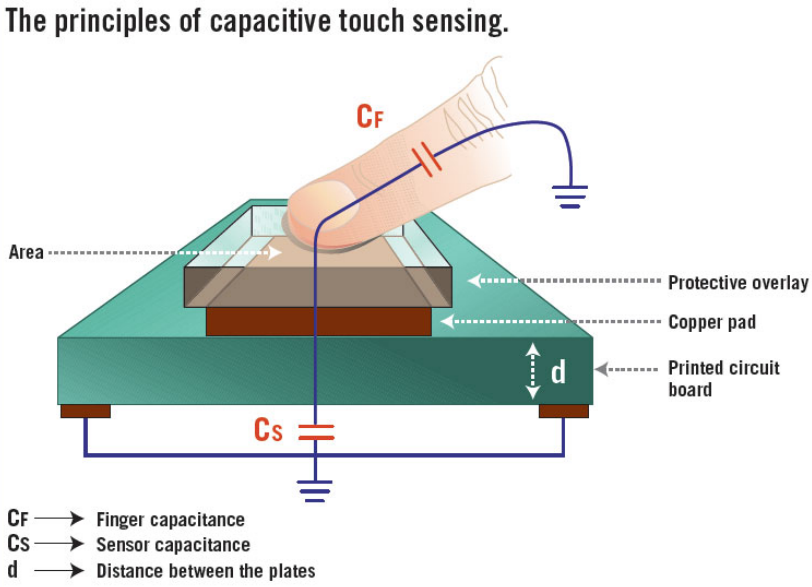
Εικόνα 3 : *Block diagram του μηχανισμού ADC με τεχνική Sample & Hold.*

2.6 LAB 6 – Touch Sense Input

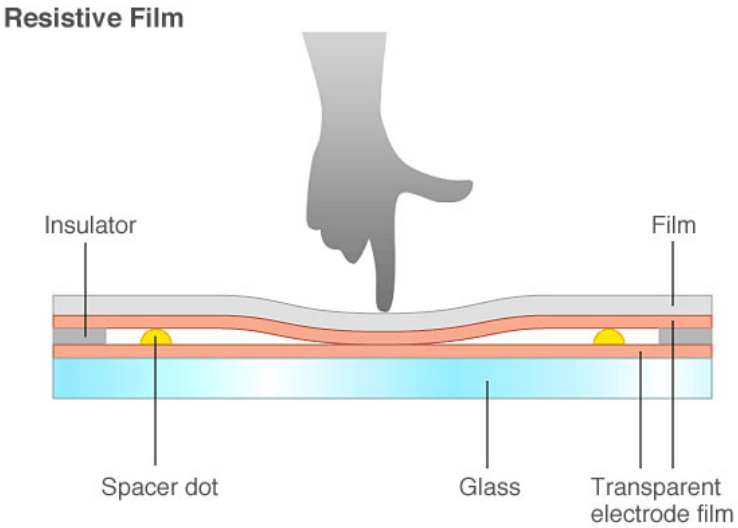
Σε αυτήν την τελευταία ενότητα θα αναλύσουμε ένα περιφερειακό που δεν συναντάται συχνά στους microcontrollers, το TSI όπου αποτελεί μηχανισμό που υλοποιεί την λειτουργία ενός αισθητήρα αφής. Αισθητήρες αφής χρησιμοποιούνται σε πολύ μεγάλο βαθμό στις μέρες μας καθώς με την τεχνολογική εξέλιξη της εποχής καταλαμβάνουν όλο και περισσότερο χώρο στην αγορά, παραμερίζοντας έτσι τους απλούς συμβατικούς διακόπτες από συσκευές όπως τα πλυντήρια ρούχων, ηλεκτρικές κουζίνες, gadget καθώς και πολλά άλλα. Το TSI (Touch Sense Input) είναι μια περιφερειακή μονάδα που, σε αντίθεση με άλλους microcontroller, στον συγκεκριμένο είναι ενσωματωμένο στο εσωτερικό του. Όπως λέει και το όνομά του το TSI δίνει την δυνατότητα στο σύστημα να υλοποιεί τον μηχανισμό καθώς και την λογική ενός Touch Button. Δηλαδή δίνει την δυνατότητα στον προγραμματιστή να κάνει χρήση αισθητήρων αφής στο σύστημα. Συνήθως η συγκεκριμένη μονάδα αποτελεί εξωτερικό περιφερειακό για τους περισσότερους microcontroller, δηλαδή ένα ξεχωριστό chip υπεύθυνο για την λειτουργία ανίχνευσης σημάτων αφής, με το οποίο αλληλεπιδρούν μέσω ενός πρωτοκόλλου επικοινωνίας. Ενσωματώνοντας λοιπόν τον παραπάνω μηχανισμό στο εσωτερικό του συγκριμένου microcontroller μας δίνεται η δυνατότητα να ρυθμίσουμε με ακρίβεια τον τρόπο λειτουργίας του, καθώς και να τελειοποιήσουμε την διαδικασία ανίχνευσης της αφής, σύμφωνα με τον hardware του συστήματος μας.

Υπάρχουν δυο τεχνολογίες για την ανίχνευση της αφής στα Embedded System, που λειτουργούν με τελείως διαφορετικό τρόπο σε κάθε περίπτωση:

1. **Capacitive Touch Sensing:** Όπως αναφέρει και το όνομά της, η τεχνολογία αυτή της χωρητικής ανίχνευση βασίζεται στην χρήση χωρητικών πεδίων που αναπτύσσονται ανάμεσα σε αισθητήρια και άνθρωπο που η αλληλεπίδρασή τους έχει ως αποτέλεσμα τον διέγερση του αισθητήρα συνεπώς και την αντίληψη του φαινομένου της αφής. Πιο συγκεκριμένα στην τεχνολογία αυτή, από την πλευρά του μηχανισμού ανίχνευσης απαιτείτε ένα ηλεκτρικό κύκλωμα σε διάταξη LC το οποίο δημιουργεί ένα ηλεκτρομαγνητικό πεδίο στην επιφάνεια του αισθητήρα. Το ανθρώπινο σώμα από την άλλη πλευρά λόγο φυσικής σύστασης αποτελεί από μόνο του ένα μεγάλο πυκνωτή ως προς τη γη και αντίστοιχα με ένα LC κύκλωμα δημιουργεί με τη σειρά του ένα ηλεκτρομαγνητικό πεδίο. Σύμφωνα με τα παραπάνω όταν η δυο αυτές πλευρές αλληλεπιδρούν δημιουργείτε το φαινόμενο της στρέβλωσης και επηρεασμού του ηλεκτρομαγνητικού πεδίου στον αισθητήρα. Από την πλευρά του αισθητήρα, παρακολουθείτε συνεχώς το Feedback του ηλεκτρομαγνητικού πεδίου που ο ίδιος δημιουργεί και με αυτόν τον τρόπο, μπορεί να αναγνωρίσει την αλλαγή που αυτό υφίσταται και συνεπώς να αναγνωρίσει το φαινόμενο της αφής.



2. **Resistive Touch Sensing:** Η τεχνική αυτή είναι πιο σπάνια και χρησιμοποιείτε σε πολύ μικρότερο βαθμό. Η τεχνική αυτή στηρίζεται στην λειτουργία ενός κυκλώματος διάταξης διαιρέτη τάσης. Πιο αναλυτικά, στην τεχνική αυτή χρησιμοποιείτε μια ειδική αντίσταση που έχει την ιδιότητα να μεταβάλει την τιμή που την ορίζει, ανάλογα με την στρέβλωση των φυσικών της διαστάσεων. Το παραπάνω σημαίνει, πως αν η αρχική τιμή της συγκεκριμένης αντίστασης είναι 4K7, τότε μπορούμε να αλλάξουμε την τιμή αυτή απλά και μόνο στρεβλώνοντας την φυσική του διάσταση. Συνεπώς αν ασκήσουμε μια πίεση στην αντίσταση αυτή με το χέρι μας τότε θα παρατηρήσουμε πως η αναγραφόμενη τιμή της αντίστασης θα αλλάξει αναλογικά. Με την χρήση της συγκεκριμένης αντίστασης λοιπόν σε διάταξη διαιρέτη τάσης με μια αντίσταση σταθερής τιμής, το σύστημα μπορεί να αντιλαμβάνεται μεταβολές στην τάση εξόδου από το κύκλωμα και με αυτό τον τρόπο να αντιλαμβάνεται το φαινόμενο της αφής στο άκρο του αισθητήρα.



2.6.1 TSI (Touch Sense Input)

Ο Συγκεκριμένος microcontroller είναι ο πρώτος της σειράς επεξεργαστών της εταιρείας καθώς και ένας από τους ελάχιστους επεξεργαστές της αγοράς, που περιλαμβάνουν το TSI περιφερειακό στο εσωτερικό του. Το TSI περιφερειακό στηρίζεται στο μηχανισμό της χωρητικής ανίχνευσης αφής (Capacitive Touch Sensing) και συνεπώς λειτουργεί με την τεχνολογία όπως ακριβώς αναλύθηκε παραπάνω.

Στο εσωτερικό του microcontroller εμπεριέχονται έως και 16 κανάλια αφής γεγονός που καθιστά τον συγκεκριμένο microcontroller άκρως ανταγωνίσιμο και ευέλικτο.

Το συγκεκριμένο περιφερειακό είναι απόλυτα παραμετροποιήσιμο μέσα από 4 Control Register που οποίοι ρυθμίζουν πλήρως την λειτουργία και την ανάλυση του.

Ο προγραμματιστής έχει την δυνατότητα να ρυθμίσει την ευαισθησία κάθε αισθητηρίου ξεχωριστά μέσω του ειδικού καταχωρητή TSI_CS2 (Control/Status Register 2), σύμφωνα με τον οποίο μπορεί να επηρεάσει το Hardware τμήμα του αισθητήρα, όπως παραδείγματος χάρη την ένταση του ρεύματος σε uA(microampere) με την οποία θα φορτίζεται και θα αποφορτίζεται ένας αισθητήρας, ή ακόμα και την στάθμη της τάσεις με την οποία θα λειτουργεί η λογική του TSI περιφερειακού.

Λειτουργία

Για να γίνουμε πιο αναλυτικοί, και αυτό το περιφερειακό λειτουργεί με την μέθοδο της σάρωσης.

Συνεπώς, έχουμε μια επιφάνεια εξωτερικά του microcontroller την οποία χρησιμοποιεί το περιφερειακό για να καταλαβαίνει το φαινόμενο της αφής και αναφέρεται ως Electrode(ηλεκτρόδιο) για το περιφερειακό, ενώ για εμάς είναι ένα απλό PAD, δηλαδή μια απλή μεταλλική επιφάνεια.

Η TSI περιφερειακή μονάδα χρησιμοποιεί 2 περιοδικά σήματα για την αναγνώριση του φαινομένου της αφής. Και τα δύο αυτά σήματα δημιουργούνται από τον microcontroller ενώ απευθύνονται σε διαφορετικές διαδικασίες. Το ένα σήμα δημιουργεί μια συχνότητα στο εσωτερικό του microcontroller και το άλλο μια συχνότητα που χρησιμοποιείτε για το electrode.Ο χρήστης έχει την δυνατότητα να ρυθμίσει και της δύο συχνότητες μέσα από τους καταχωρητές του περιφερειακού.

Η συχνότητα στο εσωτερικό του microcontroller είναι υπεύθυνη για την λειτουργία του περιφερειακού καθώς επίσης και για την παρατήρηση, μέτρηση και σύγκριση φαινομένων με την μέθοδο της σάρωσης. Παράλληλα η συχνότητα που απευθύνεται στο εξωτερικό κομμάτι του αισθητήρα βοηθά στην διαδικασία φόρτισης-αποφόρτισης του electrode με τάση. Καθώς λοιπόν το σύστημα μας θα πρέπει να

παρακολουθεί την διαδικασία φόρτισης-αποφόρτισης που εκτελείτε στο εξωτερικό κομμάτι του αισθητήρα, καταλαβαίνουμε πως η συχνότητα λειτουργίας στο εσωτερικό του microcontroller θα πρέπει να είναι αρκετά μεγαλύτερη για να πραγματοποιήσουμε μια αποτελεσματική σάρωση και η περιφερειακή μονάδα να καταγράψει οποιαδήποτε αλλαγή στο εξωτερικό κομμάτι του αισθητήρα.
Στην πράξη λοιπόν, όταν ένα μέρος του ανθρώπινου σώματος πλησιάσει το ηλεκτρομαγνητικό πεδίου που εμφανίζει ο αισθητήρας πάνω στο PAD, δημιουργείτε μια αλληλεπίδραση μεταξύ του ηλεκτρομαγνητικού πεδίου του αισθητήρα και του ηλεκτρομαγνητικού πεδίου του ανθρώπινου σώματος. Αυτό έχει ως αποτέλεσμα να επηρεάζεται η χωρητική ιδιότητα του αισθητήρα στο εξωτερικό του τμήμα και συνεπώς η ανάλογη συχνότητα να μειώνεται όπως ακριβώς βλέπουμε και στην αμέσως επόμενη εικόνα.

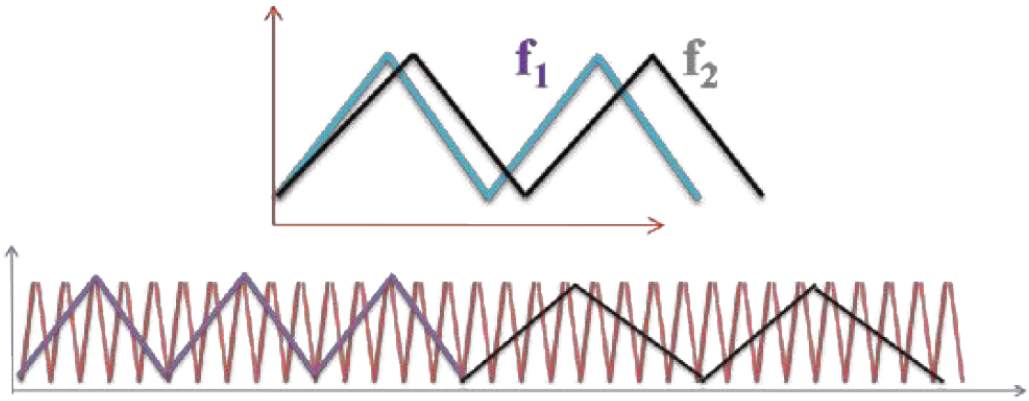


Figure 1. Internal reference oscillations (Blue) vs. external reference oscillations (Black)

Στο πρώτο διάγραμμα παρουσιάζεται η μεταβολή της συχνότητας από F2 σε F1 κατά την “επαφή” του ανθρώπινου σώματος με τον αισθητήρα, ενώ στο δεύτερο διάγραμμα βλέπουμε τις δύο διαφορετικές συχνότητες που αναφέρθηκαν προηγουμένως. Με κόκκινο χρώμα εμφανίζεται η εσωτερική συχνότητα η οποία είναι αρκετά μεγαλύτερη της εξωτερικής (μοβ-μαύρο), ενώ παράλληλα μπορούμε να διακρίνουμε την αλλαγή της εξωτερικής συχνότητας στο χρόνο από μοβ χρώμα σε μαύρο όταν ο αισθητήρας έρχεται σε “επαφή” με το ανθρώπινο σώμα, όπου και αλλάζει η τιμή της.
Με αυτή την τεχνική, η περιφερειακή μονάδα TSI είναι σε θέση παρατηρεί την μεταβολή της εξωτερικής συχνότητας εν σχέση με τον χρόνο και συνεπώς να μπορεί να αναγνωρίσει το φαινόμενο της αφής, πότε δηλαδή, ένα ανθρώπινο άκρο ακούμπησε την επιφάνεια του αισθητήρα.

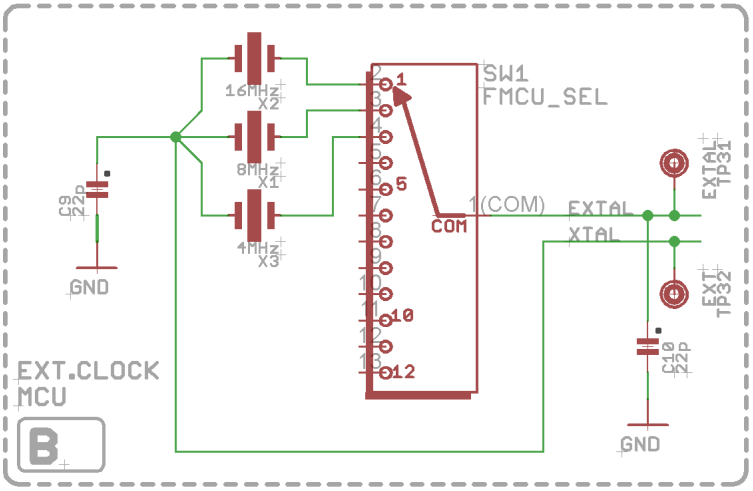
Κεφάλαιο 3 – Εφαρμογή

3.1 LAB 1 – Clock Module (External Oscillator)

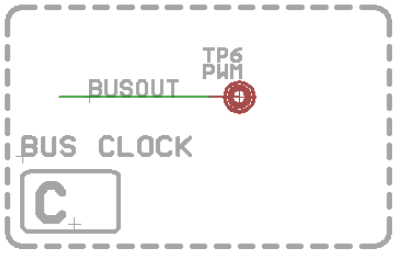
Στη Πλακέτα

Το LAB_1 χρησιμοποιεί το τμήμα **1** της πλακέτας. Σε αυτό το τμήμα της πλακέτας παρατηρούμε τα παρακάτω εξαρτήματα:

- Περιστροφικός διακόπτης 3 θέσεων για την επιλογή του εξωτερικού κρυστάλλου με τον οποίο θα δουλεύει ο microcontroller.
- 3 κρυστάλλους 4/8/18MHz για βασικό ρολόι του microcontroller.
- Σημείο ελέγχου **BUSOUT** του BusClock του microcontroller.



Σχέδιο 1.Επιλογέας κρυστάλλων σε microcontroller.



Σχέδιο 2. BUSCLK pin από τον microcontroller

LAB_1	Direction	Microcontroller
EXTAL	->	PTB7/SCL/**EXTAL**
XTAL	->	PTB6/SDA/**XTAL**
BUSOUT	<-	PTH2/**BUSOUT**

Πίνακας 1. Συνδεσμολογία σημάτων.

Στη Πράξη

Επισκόπηση :

Το LAB_1 παρουσιάζει την λειτουργία του κεντρικού ρολογιού του microcontroller. Σε αυτό το παράδειγμα μπορούμε, έπειτα από τις ανάλογες ρυθμίσεις στο Software, να δούμε τον εσωτερικό χρονισμό του συστήματος δηλαδή την συχνότητα λειτουργίας του microcontroller.

Απαραίτητος εξοπλισμός :

- Pemicro Multilink Universal
- Ηλεκτρονικός υπολογιστής
- USB καλώδιο (Multilink <-> EduBoard)
- Παλμογράφος
- EduBoard

Προετοιμασία Πλακέτας :

1. Συνδέουμε το EduBoard με τον ηλεκτρονικό υπολογιστή με την χρήση ενός καλωδίου τύπου USB.
2. Ανοίγουμε το Terminal στον ηλεκτρονικό υπολογιστή με της παρακάτω ρυθμίσεις:
    - 115200 baud rate.
    - 8 data bits.
    - No parity.
    - One stop bit.
    - No flow control.
3. Κατεβάζουμε το πρόγραμμά μας στο EduBoard (target) συμφώνα με την διαδικασία RUN.
4. Πραγματοποιούμε επανεκκίνηση του EduBoard από τον διακόπτη ON/OFF.

Εκτέλεση Προγράμματος :

Όταν το πρόγραμμα ξεκινήσει επιτυχώς θα πρέπει να εμφανιστεί στο Terminal εισαγωγικό μήνυμα με πληροφορίες σχετικά με το παρόν πρόγραμμα που εκτελείτε.

Στο παρόν παράδειγμα μας δίνεται η δυνατότητα μέσω του Software να ρυθμίσουμε τον τρόπο λειτουργίας του General Clock Module τόσο ως προς τον μηχανισμό εισόδου, ανάμεσα σε:
1)FLL - Output of FLL
2)IRC - Internal Reference Clock
3)EXT -External Reference Clock

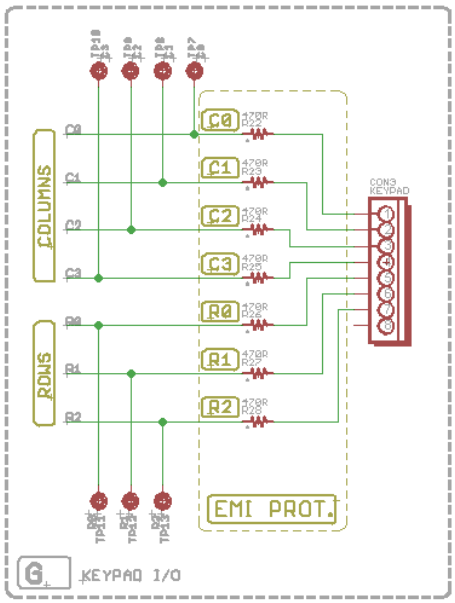
Στην περίπτωση που ρυθμίσουμε το General Clock Module σε λειτουργία EXT, μπορούμε να επιλέξουμε τον εξωτερικό κρύσταλλο μέσω του επιλογέα που βρίσκετε πάνω στο EduBoard.
Τέλος και έπειτα από τις απαραίτητες ρυθμίσεις, χρησιμοποιώντας έναν παλμογράφο μπορούμε να διαβάσουμε την εσωτερική συχνότητα του microcontroller και άρα του συστήματος, από το BUSOUT pin που βρίσκεται τοποθετημένο στην **περιοχή 1** πάνω στο EduBoard.

3.2 LAB 2 – GPIO/KBI (Keyboard)

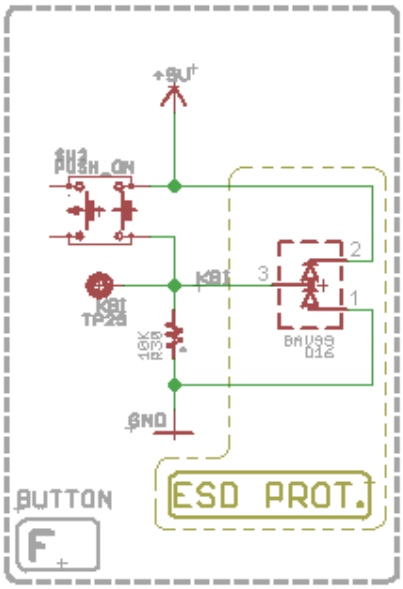
Στη Πλακέτα

Το LAB_2 χρησιμοποιεί το τμήμα **2** της πλακέτας. Σε αυτό το τμήμα της πλακέτας παρατηρούμε τα παρακάτω εξαρτήματα:

- 4 x Pins/LED σε συνδεσμολογία εξόδου (Outputs).
- 3 x Pins/LED σε συνδεσμολογία εισόδου (Inputs).
- 1 x Button/LED/Pin σε λειτουργία KBI (KeyBoard Interrupt)
- 1 x φύσα (1x8) για προσαρμογή εξωτερικού πληκτρολογίου(Keypad)



Σχέδιο 1.GPIO Interface.



*Σχέδιο 2.*Button σε συνδεσμολογία KBI με τον microcontroller.

LAB_2	Direction	Microcontroller
C0	<->	**PTB0**/KBI0P4/RxD0/ADP4/TSI2
C1	<->	**PTB1**/KBI0P5/TxD0/ADP5/TSI3
C2	<->	**PTB2**/KBI0P6/SPSCK0/ADP6/TSI4
C3	<->	**PTB3**/KBI0P7/MOSI0/ADP7/TSI5
R0	<->	**PTC0**/FTM2CH0/ADP8/TSI6
R1	<->	**PTC1**/FTM2CH1/ADP9/TSI7
R2	<->	**PTC2**/FTM2CH2/ADP10
KBI	<-	PTD4/**KBI1P4**

Πίνακας 1. Συνδεσμολογία σημάτων.

Στη Πράξη

Επισκόπηση :

Το LAB_2 παρουσιάζει την λειτουργία του τω περιφερειακών του GPIO και του KBI στον συγκεκριμένο microcontroller. Επίσης στο παρόν παράδειγμα γίνεται χρήση του Keypad Driver με το οποίο είμαστε σε θέση να παρατηρήσουμε την διαδικασίας της σάρωσης που απαιτείτε για την λειτουργία του.

Απαραίτητος εξοπλισμός :

- Pemicro Multilink Universal
- Ηλεκτρονικός υπολογιστής
- USB καλώδιο (Multilink <-> EduBoard)
- Παλμογράφος(optional)
- Keypad (Μεταλλικό πληκτρολόγιο)
- EduBoard

Προετοιμασία Πλακέτας :

1. Συνδέουμε το μεταλλικό πληκτρολόγιο (Keypad) στην φύσα που βρίσκετε στην **περιοχή 2** του EduBoard.
2. Συνδέουμε το EduBoard με τον ηλεκτρονικό υπολογιστή με την χρήση ενός καλωδίου τύπου USB.
3. Ανοίγουμε το Terminal στον ηλεκτρονικό υπολογιστή με τις παρακάτω ρυθμίσεις:
    - 115200 baud rate.
    - 8 data bits.
    - No parity.
    - One stop bit.
    - No flow control.
4. Κατεβάζουμε το πρόγραμμά μας στο EduBoard (target) σύμφωνα με την διαδικασία RUN.
5. Πραγματοποιούμε επανεκκίνηση του EduBoard από τον διακόπτη ON/OFF.

Εκτέλεση Προγράμματος :

Όταν το πρόγραμμα ξεκινήσει επιτυχώς θα πρέπει να εμφανιστεί στο Terminal εισαγωγικό μήνυμα με πληροφορίες σχετικά με το παρόν πρόγραμμα που εκτελείτε.

Πατώντας οποιοδήποτε πλήκτρο από το πληκτρολόγιο παρατηρούμε πως ανάβει ένα LED από τα INPUTS, αντίστοιχο με την στήλη του κουμπιού που

πατήθηκε από το πληκτρολόγιο και παράλληλα εμφανίζεται στο Terminal ο αριθμός/γράμμα που αντιστοιχεί στο κουμπί που πατήθηκε. Με τον τρόπο αυτό παρατηρούμε το αποτέλεσμα της σάρωσης που πραγματοποιείτε στο πληκτρολόγιο.

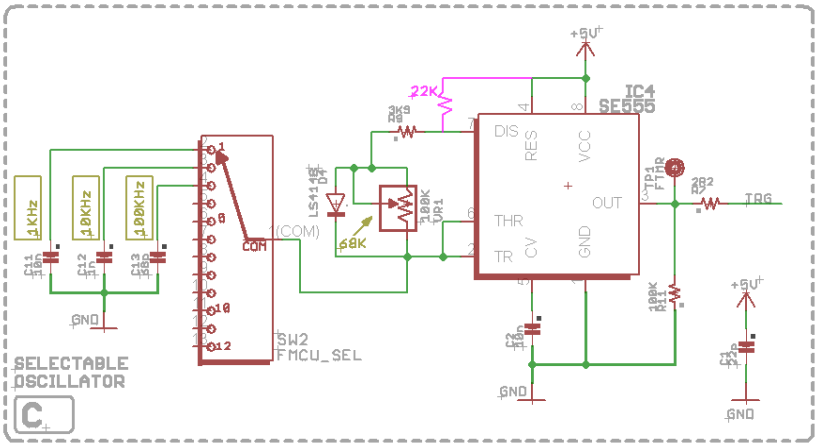
Παράλληλα, στην **περιοχή 2** παρατηρούμε το KBI κουμπί το οποίο πατώντας το ανακόπτεται η κανονική λειτουργία στο εσωτερικό του microcontroller και ένα ηχητικό σήμα ακούγεται από το buzzer της πλακέτας. Η παραπάνω διαδικασία ελέγχεται και από το Terminal το οποίο παρακολουθεί την διαδικασία αναφέροντας αντίστοιχα την αλλαγή κατάσταση Edge του KBI.

3.3 LAB 3 – FTM/MTIM (PWM the LED)

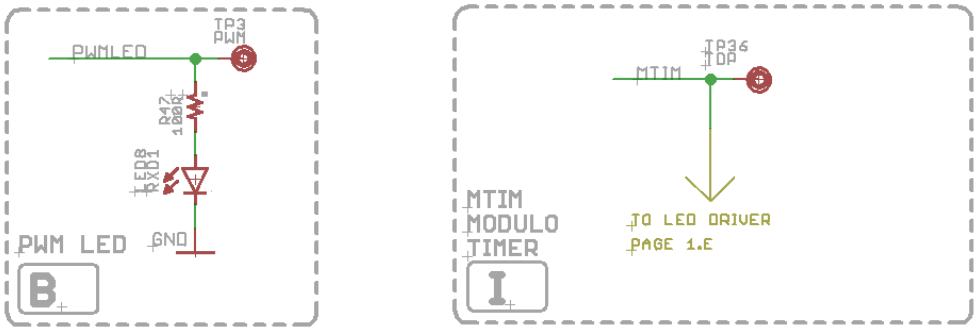
Στη Πλακέτα

Το LAB_3 χρησιμοποιεί το τμήμα **3** της πλακέτας. Σε αυτό το τμήμα της πλακέτας παρατηρούμε τα παρακάτω εξαρτήματα:

- IC NE555 για δημιουργία συχνότητας λειτουργίας του εσωτερικό Timer (FTM) του microcontroller.
- 1 x Ποτενσιόμετρο - Περιστροφικός διακόπτης 3 θέσεων για την επιλογή της συχνότητας εισόδου στον εσωτερικό Timer (FTM) του microcontroller.
- 1 x Πολυστροφικός ρότορας για μικρορύθμιση της συχνότητας του IC NE555.
- 1 x Pin/LED **PWM** για ένδειξη της εξόδου τύπου PWM.
- 1 x Pin/LED για ένδειξη της συχνότητας λειτουργίας του δεύτερου εσωτερικού Timer (MTIM).



Σχέδιο 1.Συνδεσμολογία IC NE555



Σχέδιο 2.Συνδεσμολογία ενδεικτικών Pin/LED με τον microcontroller.

LAB_3	Direction	Microcontroller
FTM	<->	PTE7/**TCLK2**
PWMLED	<->	PTD1/KBI1P1/**FTM2CH3**/MOSI11
MTIM	<->	**PTC4**/FTM1CH0

Πίνακας 1. Συνδεσμολογία σημάτων.

Στη Πράξη

Επισκόπηση:

Το LAB_3 παρουσιάζει την λειτουργία των περιφερειακών του MTIM και του FTM στον συγκεκριμένο microcontroller. Για την χρήση του FTM υπάρχει η δυνατότητα επιλογής της εσωτερικής συχνότητας λειτουργίας του περιφερειακού από έξω και όχι από το εσωτερικό General Clock Module. Παράλληλα παρουσιάζεται η υλοποίηση ενός PWM καναλιού FTM.

Απαραίτητος εξοπλισμός :

- Pemicro Multilink Universal
- Ηλεκτρονικός υπολογιστής
- USB καλώδιο (Multilink <-> EduBoard)
- Παλμογράφος
- Κατσαβίδι ηλεκτρολογικό (ίσιο)
- EduBoard

Προετοιμασία Πλακέτας :

6. Συνδέουμε το EduBoard με τον ηλεκτρονικό υπολογιστή με την χρήση ενός καλωδίου τύπου USB.
7. Ανοίγουμε το Terminal στον ηλεκτρονικό υπολογιστή με τις παρακάτω ρυθμίσεις:
    - 115200 baud rate.
    - 8 data bits.
    - No parity.
    - One stop bit.
    - No flow control.
8. Κατεβάζουμε το πρόγραμμά μας στο EduBoard (target) συμφώνα με την διαδικασία RUN.
9. Πραγματοποιούμε επανεκκίνηση του EduBoard από τον διακόπτη ON/OFF.

Εκτέλεση Προγράμματος :

Όταν το πρόγραμμα ξεκινήσει επιτυχώς θα πρέπει να εμφανιστεί στο Terminal εισαγωγικό μήνυμα με πληροφορίες σχετικά με το παρόν πρόγραμμα που εκτελείτε.

Στην **περιοχή 3** της πλακέτας στο πεδίο MTIM συνδέουμε το άκρο από το παλμογράφο στο pin TDP που βρίσκεται δίπλα στο TDP LED, ώστε να μετρήσουμε την συχνότητα στην οποία είναι χρονισμένο το περιφερειακό MTIM εντός του microcontroller. Η παραπάνω συχνότητα είναι ρυθμισμένη

από το Software στα 1000Hz δηλαδή στο 1KHz. Συνεπώς παρατηρούμε ως το LED TDP ανάβει 1000 φορές το δευτερόλεπτο.

Επίσης στο πεδίο FTM της πλακέτας υπάρχει ένας ακροδέκτης PWM και ένας TRG. Καθώς το περιφερειακό έχει ρυθμιστεί να λειτουργεί με εξωτερικό ρολόι, ακουμπώντας τον παλμογράφο στο pin TGR παρατηρούμε την συχνότητα χρονισμού του MTIM από τον εξωτερικό κόσμο, ενώ ακουμπώντας τον παλμογράφο στο pin PWM μπορούμε να παρατηρήσουμε την συχνότητα εξόδου από το περιφερειακό.

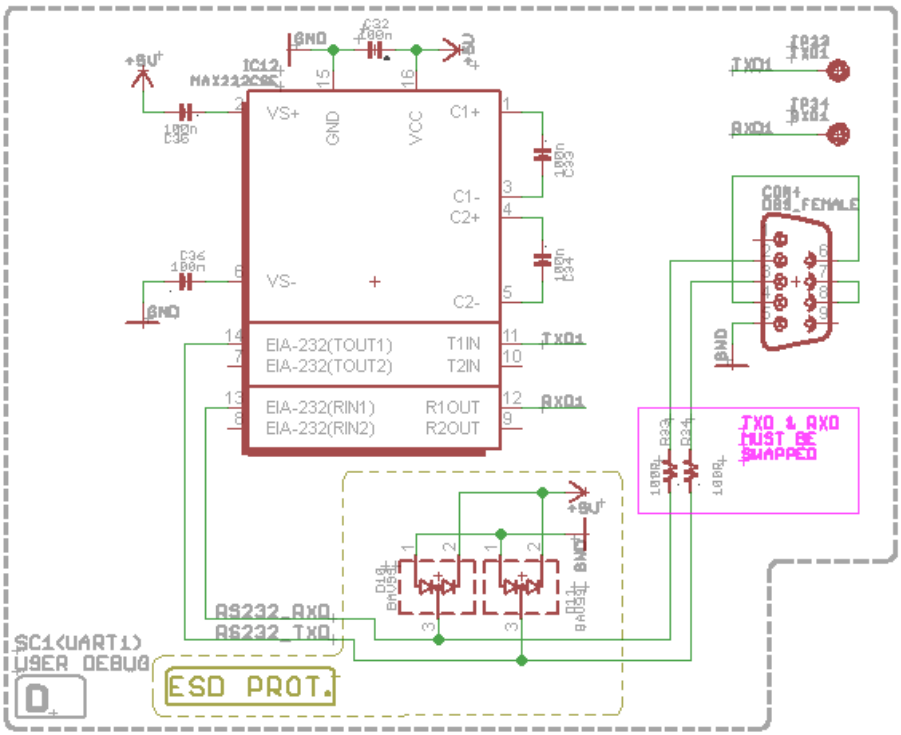
Παράλληλα το Terminal μας προτείνει να τοποθετήσουμε έναν αριθμό από το 1 έως το 99, σύμφωνα με τον οποίο στο pin PWM θα εμφανιστεί σήμα τύπου PWM με το ανάλογο Duty Cycle.

3.4 LAB 4 – Serial Communications (ADT7301 and RS232)

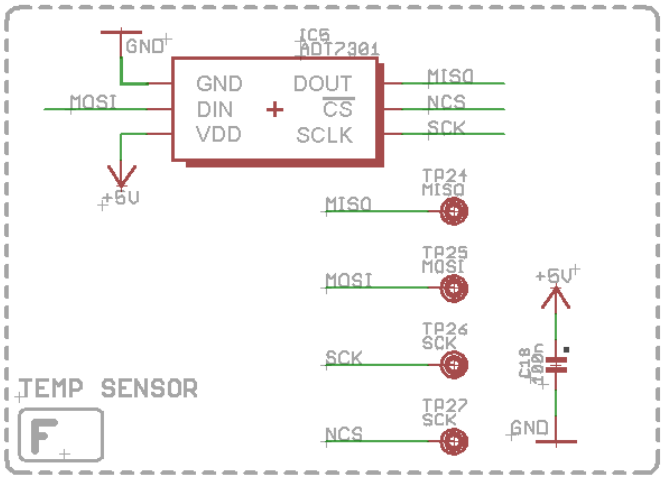
Στη Πλακέτα

Το LAB_4 χρησιμοποιεί το τμήμα **4** της πλακέτας. Σε αυτό το τμήμα της πλακέτας παρατηρούμε τα παρακάτω εξαρτήματα:

- 1 x MAX232 converter RS232 to UART
- 1 x Pin/LED σημείο εξόδου του TxD του UART συστήματος.
- 1 x Pin/LED σημείο εξόδου του RxD του UART συστήματος.
- 1 x Connector για την σύνδεση RS232 βύσματος.
- 1 x ADT7301 IC για τον υπολογισμό εξωτερικής θερμοκρασίας χώρου.
- 4 x Pins/LED για κάθε ένα σήμα του SPI πρωτοκόλλου (MOSI, MISO, SCK, NCS)



Σχέδιο 1.Συνδεσμολογία IC MAX232 (RS232 to UART) με τον microcontroller.



Σχέδιο 2.Συνδεσμολογία IC ADT7301 με τον microcontroller.

LAB_4	Direction	Microcontroller
TxD1	<-	PTC7/**TxD1**/TSI9
RxD1	->	PTC6/**RxD1**/TSI8
MOSI	<-	PTE1/**MOSI0**
MISO	->	PTE2/**MISO0**
SCK	<-	PTE0/**SPSCK0**/TCLK11
NCS	<-	**PTC5**/FTM1CH1

Πίνακας 1. Συνδεσμολογία σημάτων.

Στη Πράξη

Επισκόπηση :

Το LAB_4 παρουσιάζει την λειτουργία των περιφερειακών του SPI και του SCI στον συγκεκριμένο microcontroller. Στον παρόν παράδειγμα θα χρησιμοποιήσουμε το πρωτόκολλο SPI για να συνδέσουμε τον microcontroller με το εξωτερικό chip ADT7301 για να μετρήσουμε θερμοκρασία χώρου. Επίσης θα χρησιμοποιήσουμε το πρωτόκολλο επικοινωνίας SCI για να υλοποιήσουμε την λειτουργία της UART, ώστε να μπορούμε να συνδέσουμε τον microcontroller με τον ηλεκτρονικό μας υπολογιστή.

Απαραίτητος εξοπλισμός :

- Pemicro Multilink Universal
- Ηλεκτρονικός υπολογιστής
- USB καλώδιο (Multilink <-> EduBoard)
- Παλμογράφος
- Καλώδια για την σειριακή επικοινωνίας μέσω UART τύπου PS/2 (RS232)
- EduBoard

Προετοιμασία Πλακέτας :

10. Συνδέουμε με το σειριακό καλώδιο επικοινωνίας PS/2(RS232) τον ηλεκτρονικό υπολογιστή με το EduBoard στο πεδίο SCI στην **περιοχή 4**.
11. Συνδέουμε το EduBoard με τον ηλεκτρονικό υπολογιστή με την χρήση ενός καλωδίου τύπου USB.
12. Ανοίγουμε το Terminal στον ηλεκτρονικό υπολογιστή με τις παρακάτω ρυθμίσεις:
    - 115200 baud rate.
    - 8 data bits.
    - No parity.
    - One stop bit.
    - No flow control.
13. Κατεβάζουμε το πρόγραμμά μας στο EduBoard (target) συμφώνα με την διαδικασία RUN.
14. Πραγματοποιούμε επανεκκίνηση του EduBoard από τον διακόπτη ON/OFF.

Εκτέλεση Προγράμματος :

Όταν το πρόγραμμα ξεκινήσει επιτυχώς θα πρέπει να εμφανιστεί στο Terminal

εισαγωγικό μήνυμα με πληροφορίες σχετικά με το παρόν πρόγραμμα που εκτελείτε.

Στην **περιοχή 4** της πλακέτας στο πεδίο SPI παρατηρούμε ένα μικρό chip στην άκρη της πλακέτας. Το συγκεκριμένο Chip αποτελεί τον αισθητήρα θερμοκρασίας από τον οποίο παίρνουμε συνεχώς τιμές θερμοκρασίας του χώρου χρησιμοποιώντας το πρωτόκολλο SPI.Παράλληλα μπορούμε να παρατηρήσουμε την δραστηριότητα των LED ανά σήμα SPI κατά την επικοινωνία.

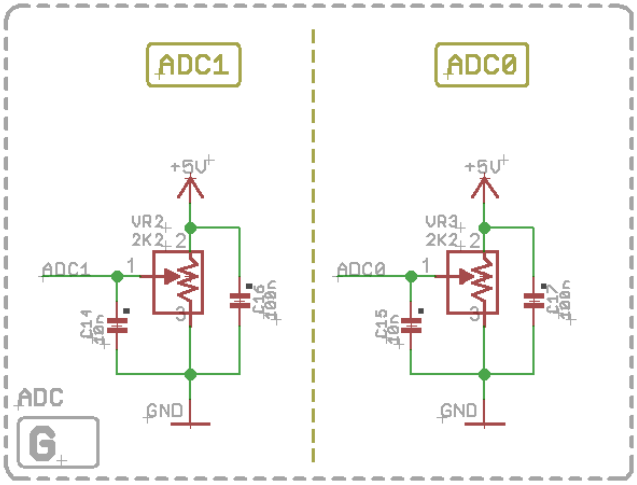
Για την λειτουργία του της σειριακής επικοινωνίας που βλέπουμε στο πεδίο SCI της ίδια περιοχής, πρέπει να συνδέσουμε το αντίστοιχο τμήμα με τον ηλεκτρονικό υπολογιστή. Έπειτα θα πρέπει να ανοίξουμε ένα δεύτερο Terminal με ρυθμίσεις ανάλογες της σύνδεσης του USB που αναγράφονται παραπάνω. Με αυτό τον τρόπο πληκτρολογώντας ένα γράμμα ή μια φράση στο Terminal ο microcontroller επιστρέφει το αντίστοιχο γράμμα ή την αντίστοιχη πρόταση μέσω της UART πίσω στο Terminal (μέθοδος γνωστή και ως ECHO).
Προσοχή! Πολλές φορές δεν είναι εφικτή η παράλληλη λειτουργία 2 Terminal. Συνεπώς για να τρέξουμε το παράδειγμα με την UART, θα πρέπει να απενεργοποιήσουμε την σύνδεση με την USB.

3.5 LAB 5 – Serial Communications (BH1750) & A2D

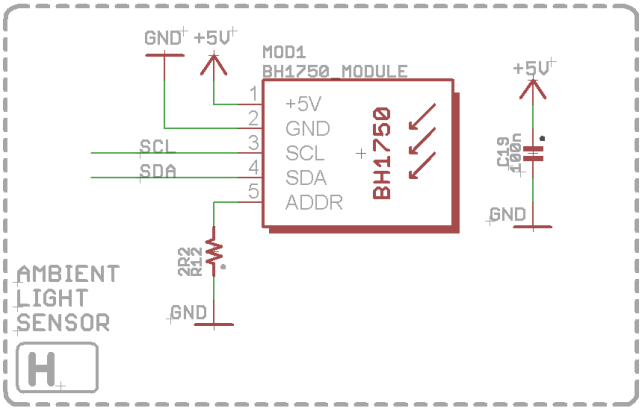
Στη Πλακέτα

Το LAB_5 χρησιμοποιεί το τμήμα **5** της πλακέτας. Σε αυτό το τμήμα της πλακέτας παρατηρούμε τα παρακάτω εξαρτήματα:

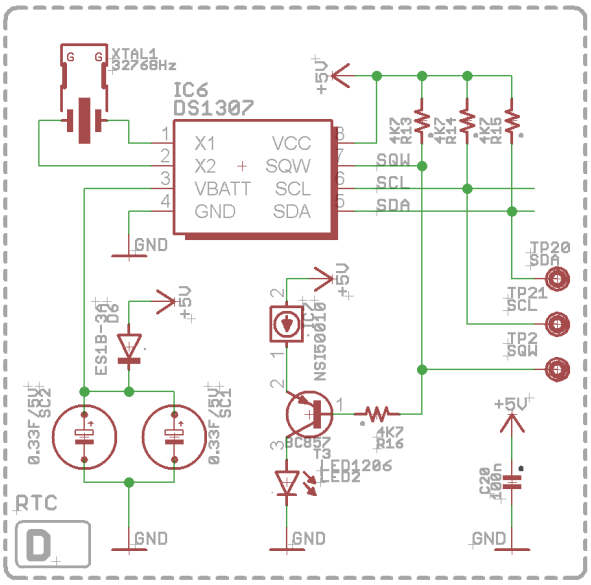
- 2 x Pin/A2D Channels αναλογικών εισόδων προς τον microcontroller.
- 2 x Trimmer για αλλαγή της αναλογικής τιμής εισόδου στον microcontroller.
- 1 x **BH1750** IC αισθητήρας έντασης φωτός.
- 1 x **DS1307** IC πραγματικού χρόνου ρολόι με λειτουργία ημερολογίου.
- 2 x SuperCap EC για την παροχή τάσης στο **BH1750**.
- 3 x Pin/LED σημεία ελέγχου του πρωτοκόλλου I2C.



Σχέδιο 1.Συνδεσμολογία αναλογικών εισόδων A2D με τον microcontroller.



Σχέδιο 2.Συνδεσμολογία IC BH1750 με τον microcontroller.



Σχέδιο 2.Συνδεσμολογία IC DS1307 με τον microcontroller.

LAB_5	Direction	Microcontroller
ADC0	->	PTA0/KBI0P0/FTM0CH0/ACMP0/**ADP0**
ADC1	->	PTA1/KBI0P1/FTM1CH0/ACMP1/**ADP1**
SDA	<->	PTA2/KBI0P2/RxD0/**SDA**
SCL	<-	PTA3/KBI0P3/TxD0/**SCL**

Πίνακας 1. Συνδεσμολογία σημάτων.

Στη Πράξη

Επισκόπηση :

Το LAB_5 παρουσιάζει την λειτουργία των περιφερειακών του I2C και του ADC στον συγκεκριμένο microcontroller. Στον παρόν παράδειγμα θα χρησιμοποιήσουμε το πρωτόκολλο I2C για να συνδέσουμε τον microcontroller με το εξωτερικό chip BH1750 για να μετρήσουμε την ένταση της φωτεινότητας στο χώρο. Επίσης θα χρησιμοποιήσουμε το περιφερειακό ADC για να κάνουμε ψηφιακή ανάλυση αναλογικών σημάτων.

Απαραίτητος εξοπλισμός :

- Pemicro Multilink Universal
- Ηλεκτρονικός υπολογιστής
- USB καλώδιο (Multilink <-> EduBoard)
- Παλμογράφος (προορατικό)
- EduBoard

Προετοιμασία Πλακέτας :

15. Συνδέουμε το EduBoard με τον ηλεκτρονικό υπολογιστή με την χρήση ενός καλωδίου τύπου USB.
16. Ανοίγουμε το Terminal στον ηλεκτρονικό υπολογιστή με τις παρακάτω ρυθμίσεις:
    - 115200 baud rate.
    - 8 data bits.
    - No parity.
    - One stop bit.
    - No flow control.
17. Κατεβάζουμε το πρόγραμμά μας στο EduBoard (target) συμφώνα με την διαδικασία RUN.
18. Πραγματοποιούμε επανεκκίνηση του EduBoard από τον διακόπτη ON/OFF.

Εκτέλεση Προγράμματος :

Όταν το πρόγραμμα ξεκινήσει επιτυχώς θα πρέπει να εμφανιστεί στο Terminal εισαγωγικό μήνυμα με πληροφορίες σχετικά με το παρόν πρόγραμμα που εκτελείτε.

Στο πεδίο I2C της **περιοχής 5** της πλακέτας παρατηρούμε το BH1750 (ανιχνευτής ένταση φωτός).
Με την εκκίνηση του προγράμματος ο microcontroller επικοινωνεί με το

BH1750 κάθε 0.5 δευτερόλεπτα και ενημερώνεται για την ένταση του φωτός στο περιβάλλον και την στέλνει στο Terminal που είναι συνδεδεμένο με τον ηλεκτρονικό υπολογιστή. Με αυτό τον τρόπο είτε επικαλύπτοντας τον αισθητήρα, είτε εκθέτοντάς τον σε πηγή φωτός μπορούμε να γνωρίζουμε την ένταση του φωτός που μετράει ο αισθητήρας.

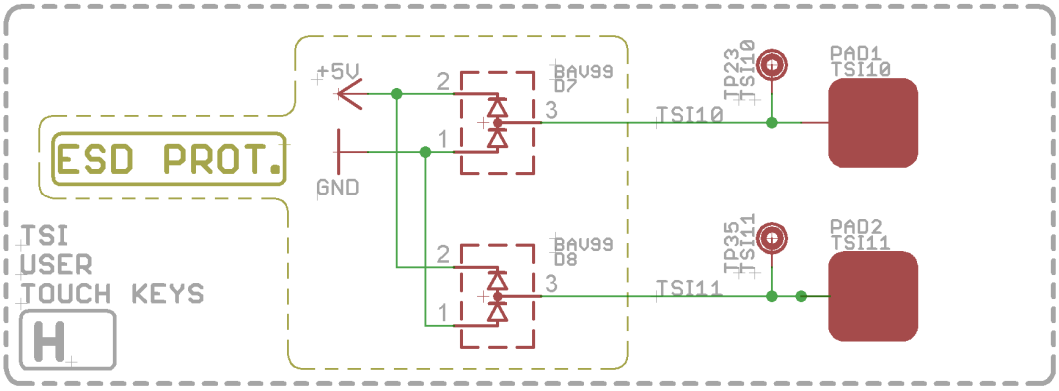
Παράλληλα, στο πεδίο ADC παρατηρούμε 2 ρυθμιζόμενες αντιστάσεις τύπου Trimmer όπου μας βοηθούν στο παράδειγμα της μετατροπής αναλογικών τιμών σε ψηφιακές. Σε αυτό το πεδίο ο χρήστης έχει την δυνατότητα να "παίξει" με τα Trimmer και να επηρεάσει την αναλογική τιμή που καλείτε να μετατρέψει ο microcontroller. Με αυτό τον τρόπο ο microcontroller μετατρέπει τα αναλογικά σήματα στις εισόδους του σε ψηφιακές και της στέλνει στο Terminal ηλεκτρονικού υπολογιστή σε πραγματικό χρόνο.

3.6 LAB 6 – Touch Sense Input

Στη Πλακέτα

Το LAB_6 χρησιμοποιεί το τμήμα **6** της πλακέτας. Σε αυτό το τμήμα της πλακέτας παρατηρούμε τα παρακάτω εξαρτήματα:

- 2 x Pad/TSI επιφάνειες για αισθητήρες αφής.



Σχέδιο 1.Συνδεσμολογία αναλογικών εισόδων A2D με τον microcontroller.

LAB_6	Direction	Microcontroller
TSI10	->	PTD2/KBI1P2/MISO1/**TSI10**
TSI12	->	PTD3/KBI1P3/SS1/**TSI11**

Πίνακας 1. Συνδεσμολογία σημάτων.

Στη Πράξη

Επισκόπηση :

Το LAB_6 παρουσιάζει την λειτουργία του TSI περιφερειακού του συγκεκριμένου microcontroller. Στον παρόν παράδειγμα θα χρησιμοποιήσουμε το περιφερειακό TSI για να αναγνωρίσουμε ερεθίσματα αφής από τον έξω κόσμο. Σε αυτό το παράδειγμα γίνεται χρήση 2 αισθητήρων αφής που προσομοιώνουν την λειτουργία 2 κανονικών κουμπιών πάνω σε ένα Embedded System.

Απαραίτητος εξοπλισμός :

- Pemicro Multilink Universal
- Ηλεκτρονικός υπολογιστής
- USB καλώδιο (Multilink <-> EduBoard)
- EduBoard

Προετοιμασία Πλακέτας :

19. Συνδέουμε το EduBoard με τον ηλεκτρονικό υπολογιστή με την χρήση ενός καλωδίου τύπου USB.
20. Ανοίγουμε το Terminal στον ηλεκτρονικό υπολογιστή με τις παρακάτω ρυθμίσεις:
    - 115200 baud rate.
    - 8 data bits.
    - No parity.
    - One stop bit.
    - No flow control.
21. Κατεβάζουμε το πρόγραμμά μας στο EduBoard (target) συμφώνα με την διαδικασία RUN.
22. Πραγματοποιούμε επανεκκίνηση του EduBoard από τον διακόπτη ON/OFF.

Εκτέλεση Προγράμματος :

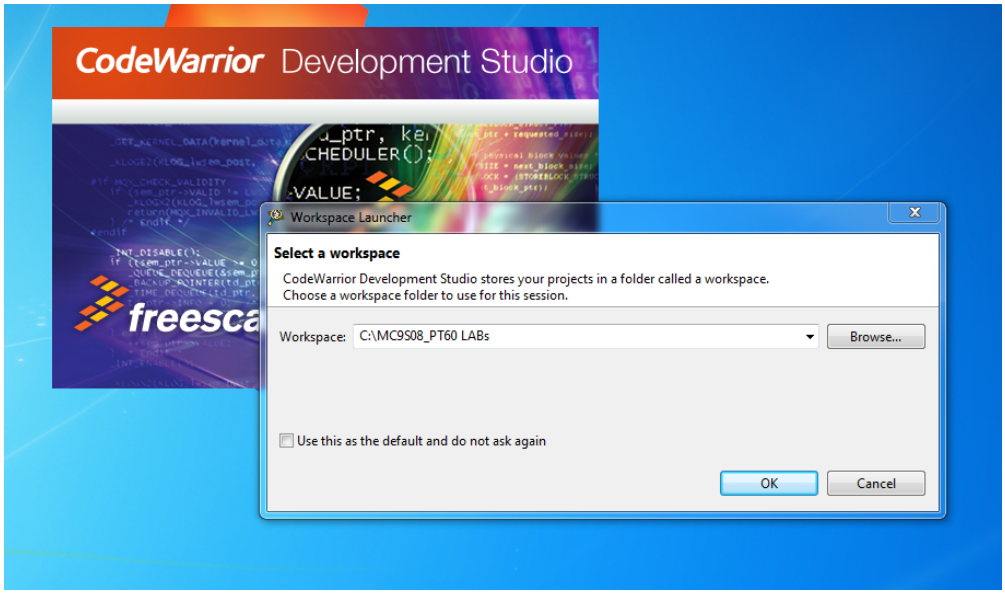
Όταν το πρόγραμμα ξεκινήσει επιτυχώς θα πρέπει να εμφανιστεί στο Terminal εισαγωγικό μήνυμα με πληροφορίες σχετικά με το παρόν πρόγραμμα που εκτελείτε.

Το πρόγραμμα που εκτελείτε στον microcontroller, παρακολουθεί σε πραγματικό χρόνο την κατάσταση των αισθητηρίων αφής που υπάρχουν στην περιοχή 6 και ενημερώνει τόσο το Terminal όσο και την LCD οθόνη για την αλλαγή κατάστασης.

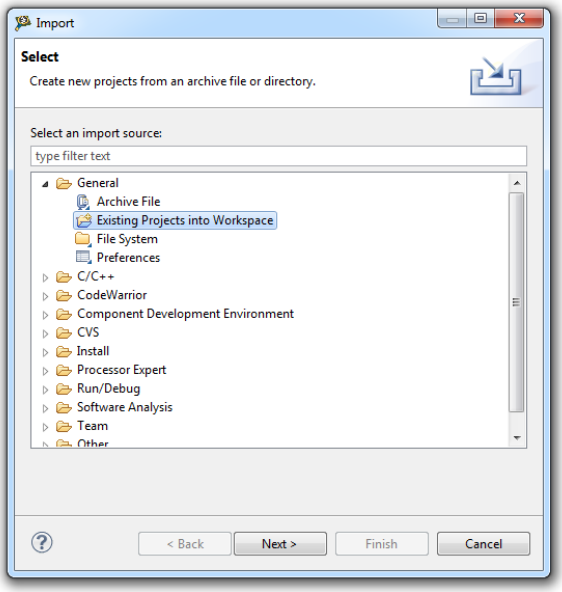
Κεφάλαιο 4 - Build a Project

Τα παρακάτω βήματα δείχνουν τον ενδεικτικό τρόπο εισαγωγής του LAB_1 project στο CW10.6 IDE.

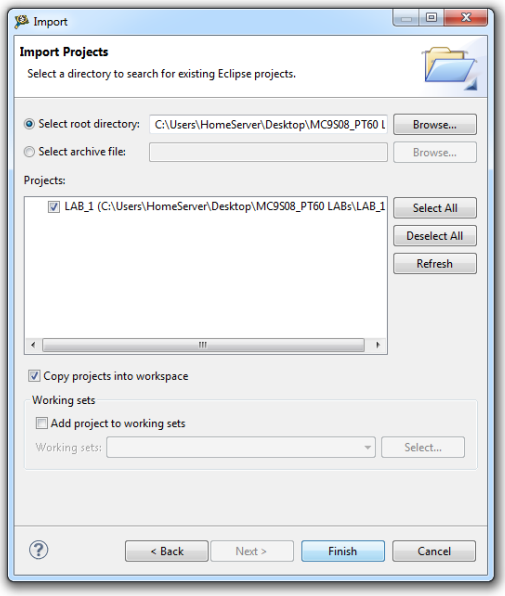
1. Ανοίγουμε το IDE σε μια προεπιλεγμένη θέση, η οποία από το εξής θα χρησιμοποιείτε ως **workspace** για το IDE.



2. Ανοίγουμε το παράθυρο **“Import”** πατώντας **“File->Import…”**.
3. Στο παράθυρο **“Import”** που εμφανίζεται, επιλέγουμε από τον φάκελο **“General”** την επιλογή **“Existing Project into Workspace”**

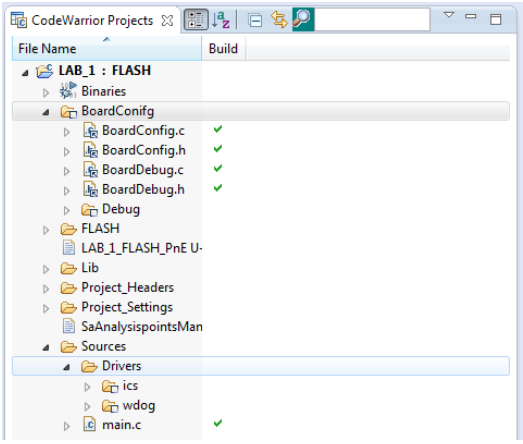


4. Στο παράθυρο που εμφανίζεται, επιλέγουμε την επιλογή **“Select root directory”** και αφού πατήσουμε το κουμπί **“Browse”,** αναζητούμε από τον explorer την διαδρομή του LAB_1 από τον φάκελο “\Firmwares” με τα έτοιμα project.
5. Επιλέγουμε την επιλογή **“Copy project into workspace”**



6. Πατάμε το κουμπί “**Finish”.**

7. Έπειτα από την σωστή διαδικασία εφαρμογής του Project στο IDE, θα πρέπει να βλέπουμε το παρακάτω.



8. Αφού καθαρίσουμε(Clean) το Project, ρυθμίζουμε στον κώδικά τον γενικότερο τρόπο λειτουργίας του EduBoard. Πηγαίνοντας στο αρχείο **“BoardConfig.h”** μας δίνεται η δυνατότητα να επιλέξουμε με την χρήση των “#defines” τα παρακάτω:

```
/*LAB Select*/
#define LAB                 LAB_1

/*LCD Function*/
#define LCD_Enable          1           /* 1=ON , 0=OFF*/

/*Buzzer Function*/
#define Buzzer_Enable       1           /* 1=ON , 0=OFF*/

/*Usb Function*/
#define USB_Enable          1           /* 1=ON , 0=OFF*/

/*Bus Clock Output*/
#define BusClockOut         1           /* 1=ON , 0=OFF*/
```

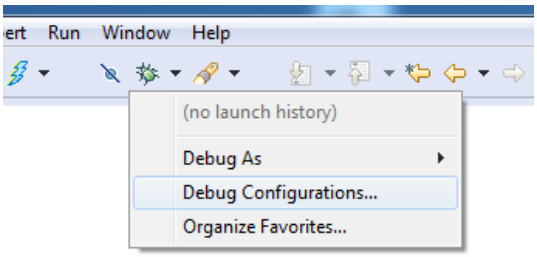
9. Έπειτα από την διαδικασία ρύθμιση του EduBoard καθώς και την ολοκλήρωση οποιασδήποτε αλλαγής του κώδικα, πατάμε το κουμπί **“Build”** για να ‘χτίσουμε’ το Project μας.

Κεφάλαιο 5 - Debug

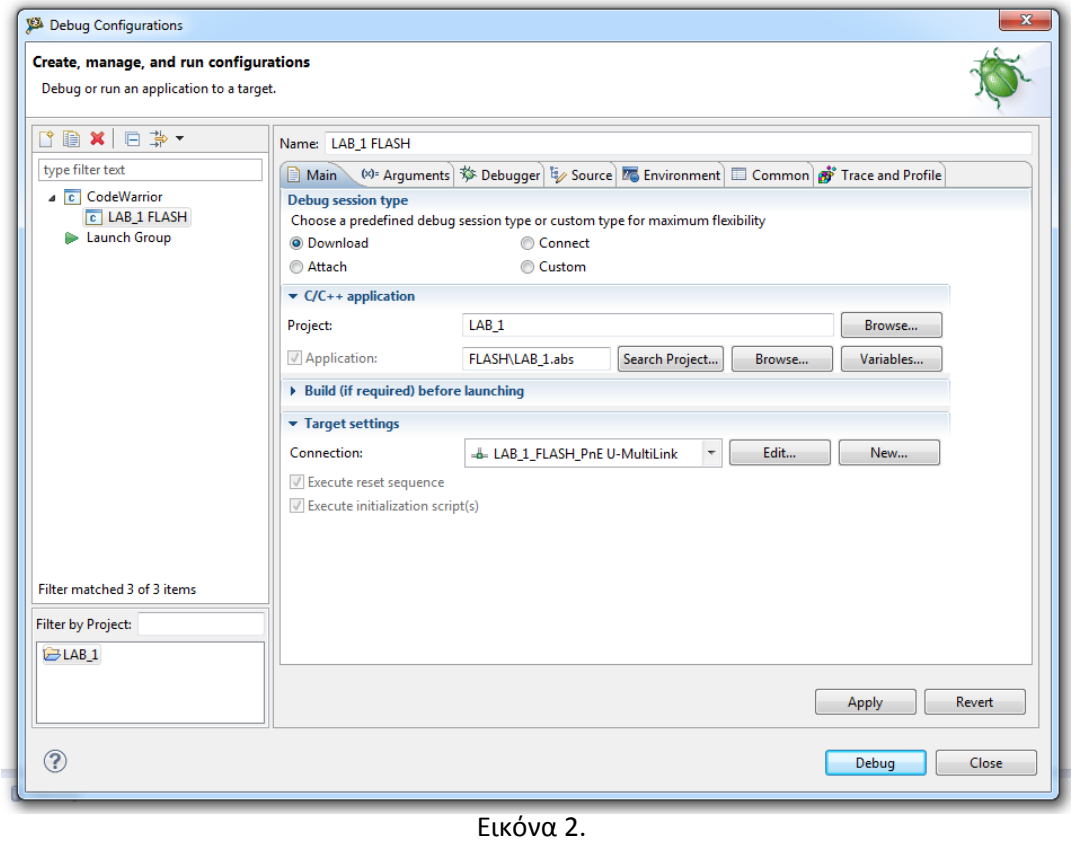
Τα παρακάτω βήματα δείχνουν τον ενδεικτικό τρόπο εκτέλεσης ενός Project πάνω στο EduBoard με την δυνατότητα **“Debug”**.

1. Συνδέουμε το **“Debugger”** στο PC μέσω του βύσματος USB.

2. Έπειτα από την επιτυχή διαδικασία χτισίματος (**Build)** του Project, πατάμε το βελάκι δεξιά από το κουμπί **(Debug)** επιλέγοντας τη δυνατότητα Debug Configuration για να δημιουργήσουμε ένα Debug Profile για το συγκεκριμένο Project.
   Προσοχή! Αν δεν εμφανίζεται το “LAB_x FLASH” της εικόνας 2 πατήστε διπλό κλικ στο “CodeWarrior”.



Εικόνα 1.



Εικόνα 2.

3. Τέλος περιμένουμε μέχρι να ξεκινήσει η διαδικασία του **“Debug”**.
4. Γνωρίζουμε ότι είμαστε έτοιμοι για την διαδικασία **“Debug”** όταν το γραφικό περιβάλλον αλλάξει και μια πράσινη επισκίαση βρίσκετε πάνω στην πρώτη γραμμή κώδικα του προγράμματός μας. Για να ξεκινήσουμε πατάμε το κουμπί **“Start”.**

Κατά την διαδικασία του **“Debug”** το IDE αλλάζει εμφάνιση ώστε να μας βοηθήσει να πραγματοποιήσουμε την διαδικασία του Debug με τον καλύτερο δυνατό τρόπο ως προς την εμπειρία του χρήστης. Παρακάτω, περιγράφονται αναλυτικά μερικές από τις πιο σημαντικές δυνατότητες που παρέχονται την περάτωση της διαδικασίας προσθέτοντας τα παρακάτω κουμπιά:

- Resume
- Suspend
- Terminate
- Disconnect
- Step Into
- Step Over
- Step Return
- Reset
- Restart

5.1 Resume (Συνέχεια)
Πραγματοποιεί συνέχεια εκτέλεσης του προγράμματος από το σημείο στο οποίο έχει σταματήσει. Επιπλέον πραγματοποιεί εκκίνηση της διαδικασίας εκτέλεσης προγράμματος κατά την εκκίνηση του Debug.

5.2 Suspend (Παύση)
Πραγματοποιεί παύση της εκτέλεση του προγράμματος σε οποιοδήποτε σημείο και αν βρίσκεται. Η διαδικασία της παύση σκιαγραφεί με πράσινο χρώμα την γραμμή κώδικα που θα εκτελέσει ο microcontroller αμέσως μετά την συνέχεια εκτέλεσης του προγράμματος.

5.3 Terminate (Τερματισμός)
Πραγματοποιεί τερματισμό της εκτέλεσης του προγράμματος ενώ παράλληλα τερματίζει και την διαδικασία του Debug.

5.4 Disconnect (Διακοπή)
Πραγματοποιεί διακοπή της διαδικασία του Debug. Τα αποτελέσματά αυτή της ενέργειας είναι παρόμοια με εκείνα του τερματισμού (Terminate) με μόνη διαφορά πως το κουμπί του τερματισμού ενεργοποιείται μόνο κατα την διαδικασία εκτέλεσης του Debug.

5.5 Step Into (Μετάβαση Εντός)
Πραγματοποιεί εκτέλεση της αμέσως επόμενης γραμμής κώδικα σύμφωνα με τους παρακάτω κανόνες:

- Αν η επόμενη γραμμή κώδικα, δεν αποτελεί συνάρτηση, τότε εκτελείται η συγκεκριμένη γραμμή κώδικα.
- Αν η επόμενη γραμμή κώδικα, αποτελεί συνάρτηση, τότε εκτελείται η πρώτη γραμμή κώδικα στο εσωτερικό της συνάρτησης.
- Αν η επόμενη γραμμή κώδικα αποτελεί το τέλος μιας συνάρτησης τότε εκτελείται η επόμενη γραμμή κώδικα εκτός της συνάρτησης.

5.6 Step Over (Μετάβαση Πάνω)
Πραγματοποιεί εκτέλεση του προγράμματος έως ότου συναντήσει:

- Το τέλος της εκτελούμενης συνάρτησης
- Breakpoint
- Watchpoint
- Ένα eventpoint το οποίο σταματά την εκτέλεση

5.7 Step Return (Μετάβαση Πίσω)
Πραγματοποιεί εκτέλεση της επόμενης ρουτίνα και μετά την εκτέλεσή της επιστέφει στην γραμμή κώδικα που την κάλεσε σε κατάσταση *Suspend.*

5.8 Reset (Επανατοποθέτηση)
Πραγματοποιεί επανεκκίνηση του microcontroller δίχως την επαναφόρτωση του προγράμματος και συνεπώς δίχως να επηρεάζει τυχόν μεταβλητές που έχουν αποθηκευθεί στην μνήμη Flash από το πρόγραμμα κατά την εκτέλεσή του. Η μέθοδος αυτή βοηθάει τον χρήστη να παρατηρεί την διαδικασία επανεκκίνησης του microcontroller σε πραγματικές συνθήκες, όντας σε διαδικασία Debug.

5.9 Restart (Επανεκκίνηση)
Πραγματοποιεί επανεκκίνηση της διαδικασίας του Debug. Με την λειτουργία του Restart το περιβάλλον ξαναφορτώνει από την αρχή το πρόγραμμα στον microcontroller. Η διαδικασία αυτή βοηθάει τον χρήστη να πραγματοποιεί εκτέλεση του προγράμματος από την αρχή δίχως αρχικές ρυθμίσεις στον microcontroller.

ΒΙΒΛΙΟΓΡΑΦΙΑ

1. “MICROCONTROLERS MOTOROLA – FREESCALE” - Juan Carlos Vesga 2008
2. “Theory and Design with Freescale microcontrollers: Family Flexis 32-bit tm MCF51QE” - Antonio Díaz Estrella 2008
3. “Embedded Systems Interfacing for Engineers using the Freescale HCS08 Microcontroller I: Digital and Analog Hardware Interfacing” - Douglas Summerville 2009
4. **“**Embedded Systems Interfacing for Engineers using the Freescale HCS08 Microcontroller II: Digital and Analog Hardware Interfacing**” -** Douglas Summerville 2009
5. “68HC12 Microcontroller: Theory and Applications” - Daniel J. Pack, Steven Frank Barrett 2002
6. “C Programming for Microcontrollers” - Joe Pardue 2005
7. “An Embedded Software Primer” - David E. Simon 1999
8. “Programming Embedded Systems in C and C++” - Michael Barr 1999
9. “Vacuum Electronics: Components and Devices” - Joseph A. Eichmeier, Manfred Thumm 2008
10. “Modern digital and analog communication systems” - Bhagwandas Pannalal Lathi 1983
11. “Test-driven Development for Embedded C” - James W. Grenning 2011
12. “Wireless Sensors and Instruments: Networks, Design, and Applications” - Halit Eren 2005
13. “Measurement, Instrumentation, and Sensors Handbook, Second Edition: Electromagnetic, Optical, Radiation, Chemical, and Biomedical Measurement” - John G. Webster, Halit Eren 2014
14. “Measurement, Instrumentation, and Sensors Handbook, Second Edition: Spatial, Mechanical, Thermal, and Radiation Measurement” - John G. Webster, Halit Eren 2014
15. “Analog Communication” - U.A.Bakshi A.P.Godse 2010
16. “Analog Electronic Circuits” - U.A.Bakshi A.P.Godse 2009

17. “Semiconductor Devices & Circuits” - U.A.Bakshi A.P.Godse 2008

18. “A Guide to Printed Circuit Board Design” - Charles Hamilton 2013

19. “EMC and the Printed Circuit Board: Design, Theory, and Layout Made Simple” - Mark I. Montrose 2004

20. “Introduction to Robotics” - Saeed Niku 2010

21. “Embedded Systems Design” - Steve Heath 2002

22. “Microprocessors and Microcomputer-Based System Design” - Mohamed Rafiquzzaman 1995

23. “Embedded Microprocessor Systems: Real World Design” - Stuart Ball 2002

24. “Microprocessor Architectures and Systems: RISC, CISC and DSP” - Steve Heath 2014

25. “Embedded Systems” - Jack G. Ganssle 2008

26. “Real-Time Concepts for Embedded Systems” - Qing Li, Caroline Yao 2003

27. “Microcontrollers: Theory and Applications” - Ajay V Deshmukh 2005

28. “Direct Sensor-to-microcontroller Interface Circuits: Design and Characterisation” - Ferran Reverter, Ramón Pallás Areny 2005

29. “Embedded Microcomputer Systems: Real Time Interfacing” - Jonathan Valvano 2011

30. “Embedded Systems Know It All Bundle” - Jean J. Labrosse, Jack Ganssle 2008

31. “Power Estimation and Optimization Methodologies for VLIW-based Embedded Systems” - Vittorio Zaccaria 2003

32. “Electronics & Microprocessors” - A.P.Godse, D.A.Godse 2010

33. “Light-Matter Interaction: Physics and Engineering at the Nanoscale” - John Weiner, Frederico Nunes 2012

34. “Quantum Electronics for Atomic Physics and Telecommunication” - Warren Nagourney 2014

35. “Fluctuating Nonlinear Oscillators: From Nanomechanics to Quantum Superconducting Circuits” - Mark Dykman 2012

36. “High-Frequency Oscillator Design for Integrated Transceivers” - J. van der Tang, Dieter Kasperkovitz, Arthur van Roermund 2006

37. “Mastering the I2C Bus” - Vincent Himpe 2011

38. “Embedded Hardware: Know It All” - Jack Ganssle, Tammy Noergaard, Fred Eady, Lewin Edwards, David J. Katz, Rick Gentile, Ken Arnold, Kamal Hyder, Bob Perrin 2007

39. “The I²C bus: from theory to practice” - Dominique Paret, Carl Fenger 1997

40. “Higher Electronics” - Mike James 1999