ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μετάδοση Δεδομένων Μέσω Του Πρωτοκόλλου I2C

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΝΙΚΟΛΑΟΥ ΜΟΥΖΗ

Επιβλέπων: Ιωάννης Αβαριτσιώτης
Καθηγητής Ε.Μ.Π

Αθήνα, Ιούνιος 2005

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΕΠΙΚΟΙΝΩΝΙΩΝ ΗΛΕΚΤΡΟΝΙΚΗΣ ΚΑΙ ΣΥΣΤΗΜΑΤΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ

Μετάδοση Δεδομένων Μέσω Του Πρωτοκόλλου I2C

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΝΙΚΟΛΑΟΥ ΜΟΥΖΗ

Επιβλέπων: Ιωάννης Αβαριτσιώτης
Καθηγητής Ε.Μ.Π

.....	.....	.....
Ιωάννης Αβαριτσιώτης	Ελευθέριος Καγιάφας	Βασίλειος Λούμος
Καθηγητής Ε.Μ.Π.	Καθηγητής Ε.Μ.Π.	Αναπλ. Καθηγητής Ε.Μ.Π.

.....

ΝΙΚΟΛΑΟΣ ΜΟΥΖΗΣ

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Αθήνα, Ιούνιος 2005

Περίληψη

Σκοπός της διπλωματικής εργασίας είναι ο σχεδιασμός και η κατασκευή συστήματος για την σειριακή επικοινωνία και μετάδοση πληροφοριών, μέσω του πρωτοκόλλου επικοινωνίας IIC (inter-integrated circuit) ή I^2C, μεταξύ δυο μικροελεγκτών και εμφάνιση των δεδομένων που συλλέγονται στην οθόνη του ηλεκτρονικού υπολογιστή.

Το σύστημα έχει τη δυνατότητα να μεταδίδει τον αριθμό πληροφοριών που επιθυμείται χωρίς καμία δέσμευση και η μετάδοση να γίνεται με αρκετά μεγάλες ταχύτητες.

Οι δύο μικροελεγκτές (microcontrollers),που χρησιμοποιούνται στο όλο σύστημα είναι της Microchip και της οικογένειας των PIC .

Λέξεις Κλειδιά

I^2C Bus, CAN Bus, SPI, Μικροελεγκτής PIC, MPLAB IDE, CCS Compiler

Abstract

The target of this final year's project is the design and implementation of a serial communication system which allows data transmission between two microcontrollers connected via IIC (inter-integrated circuit) or I^2C. The transmitted data is then downloaded through a serial connection, with the aid of a personal computer.

The system is able to transmit numbers of bytes with no limits and with an eligible high transmission bit rate.

Two PIC microcontrollers have been used for the completion of this project.

Key Words

I^2C Bus, CAN Bus, SPI, PIC microcontroller, MPLAB IDE, CCS Compiler

Ευχαριστίες.

Θα ήθελα να εκφράσω τις ευχαριστίες μου στον επιβλέποντα καθηγητή κ. Ιωάννη Αβαριτσιώτη για την καθοδήγηση, τις υποδείξεις και την παροχή του εργαστηριακού εξοπλισμού. Επίσης ευχαριστώ τον Θεοφάνη Λάμπρου και Γιώργο Μαζαράκη για την πολύτιμη βοήθειά τους.

Περιεχόμενα

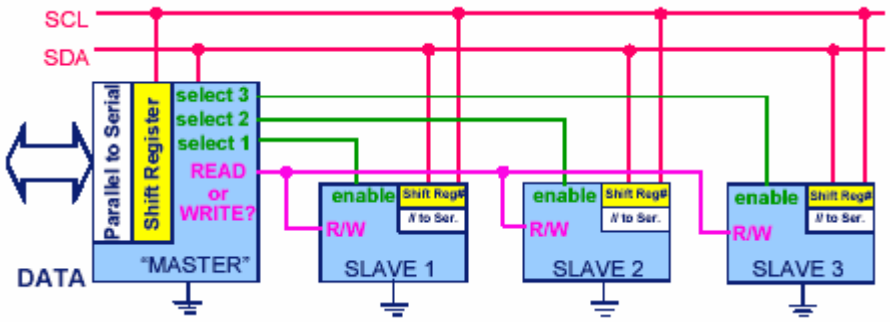
1. Γενική Εισαγωγή

Όπως έχει γίνει πια φανερό, η συνεχής εξέλιξη των δικτύων επικοινωνίας έχει αποφέρει και την ταυτόχρονη αύξηση του μεγέθους τους αλλά και της πολυπλοκότητας τους. Αυτά τα συστήματα περιλαμβάνουν υψηλής ταχύτητας ολοκληρωμένα κυκλώματα με κρίσιμους παράγοντες λειτουργίας. Ο τρόπος διασύνδεσης των επιμέρους μονάδων, του κάθε λογής δικτύου μπορεί να οδηγήσει όχι μόνο στην αποδοτικότερη λειτουργία του αλλά και στον καλύτερο έλεγχο των υπομονάδων του. Η σειριακή επικοινωνία είναι ιδιαίτερα διαδεδομένη για την μετάδοση δεδομένων και χρησιμοποιείται τόσο για την διασύνδεση διαφορετικών ολοκληρωμένων της ίδιας συσκευής όσο και μεταξύ διαφορετικών συσκευών.

Για την διασύνδεση μεταξύ διαφορετικών ολοκληρωμένων έχουν επικρατήσει κυρίως τρία πρωτόκολλα: το I^2C, το SPI και το CAN.

Το I^2C bus λοιπόν προσφέρει μια αρκετά αξιόπιστη λύση με μικρό κόστος υλοποίησης του. Σε συνάρτηση με την χρησιμοποίηση χαμηλού κόστους μικροελεγκτών(MCUs) και των περιφερειακών τους, το I^2C βρίσκει πολλές χρήσιμες εφαρμογές στις οποίες μπορούμε να έχουμε αξιόπιστη μετάδοση αλλά και λήψη δεδομένων.

Στην συγκεκριμένη εφαρμογή θα επιτευχθεί μεταφορά δεδομένων μεταξύ δύο μικροελεγκτών με την βοήθεια του I^2C bus.



Σχήμα 1.1: Γενικό διάγραμμα για σειριακή επικοινωνία

2. Εισαγωγή στο I^2C

2.1 Εισαγωγή

Το I^2C Bus αποτελεί ένα πρωτόκολλο που χρησιμοποιείται για την διασύνδεση μεταξύ διαφορετικών ολοκληρωμένων, όπως των μικροελεγκτών και των υπόλοιπων ολοκληρωμένων περιφερειακών όπως EEPROMs, A/D μετατροπείς, LCD drivers, αισθητήρες πίεσης αλλά και με άλλους μικροελεγκτές. Είναι ένα πρωτόκολλο που μεταδίδει δεδομένα σειριακά και είναι μια από τις λειτουργίες της μονάδας του PICmicro που ονομάζεται

σύγχρονη σειριακή θύρα(SSP). Ο άλλος τρόπος λειτουργίας της είναι ως Σειριακό Περιφερειακό Interface(SPI). Με το I^2C αποφεύγεται η χρησιμοποίηση ενός παράλληλου διαύλου δεδομένων που εισάγει μεγάλη πολυπλοκότητα στη σχεδίαση αλλά και μεγαλύτερο κόστος. Βρίσκει πολλές εφαρμογές στα σύγχρονα ηλεκτρονικά συστήματα όπως σε συσκευές εικόνας και ήχου, σε τηλεφωνικές συσκευές, modems, dip switches, embedded microprocessor boards αλλά και στην επικοινωνία αισθητήρων θερμοκρασίας με τις οθόνες όπου παρουσιάζονται τα αποτελέσματα των μετρήσεων.

Οι ταχύτητες τις οποίες επιτυγχάνει μπορούν να φτάσουν μέχρι και 3.4 Mbps, ταχύτητες ικανές για την ανταλλαγή δεδομένων ανάμεσα στους κόμβους και το μήκος του καλωδίου μερικές δεκάδες μέτρα, μήκος αρκετά ικανό αν θεωρήσουμε ότι οι αποστάσεις συνήθως δεν ξεπερνάνε τα μερικά μέτρα.

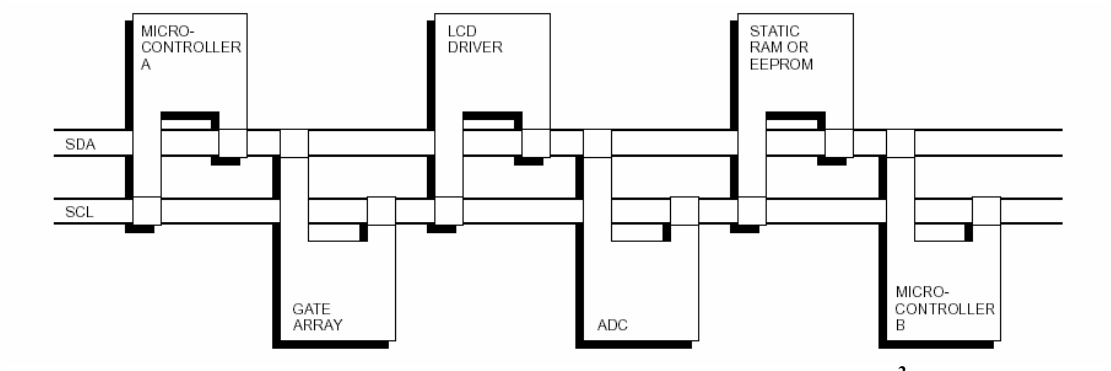
2.2 Ιστορικό Υπόβαθρο

Το πρωτόκολλο I^2C (inter-integrated circuit) αναπτύχθηκε την δεκαετία του 1980 από την Philips αρχικά με σκοπό την εύκολη επικοινωνία μεταξύ μιας Κεντρικής Μονάδας Επεξεργασίας(CPU) με τα περιφερειακά κυκλώματα μιας τηλεόρασης.

Οι περιφερειακές συσκευές σε ενσωματωμένα (embedded) συστήματα συχνά συνδέονται στον MCU σε καταχωρημένες διευθύνσεις μνήμης ως Input/Output(I/O) συσκευές, χρησιμοποιώντας των μικροελεγκτών την παράλληλη διεύθυνση καθώς και το καλώδιο(bus) δεδομένων. Αυτό είχε ως αποτέλεσμα πολυπλοκότητα, μεγάλο αριθμό καλωδιώσεων πάνω στα PCB's (Printed Circuit Board) προκειμένου να δρομολογηθούν οι διευθύνσεις και οι γραμμές δεδομένων, καθώς επίσης και ένα μεγάλο αριθμό από αποκωδικοποιητές διευθύνσεων και συντηρητικής λογικής να συνδεθούν όλα μεταξύ τους. Έτσι σε μαζικής παραγωγής προϊόντα όπως τηλεοράσεις, video και προϊόντα ήχου αυτό δεν ήταν δυνατόν να συμβεί. Σε αυτές τις συσκευές, κάθε εξάρτημα που μπορεί να μικρύνει σε μέγεθος ή ακόμα και να παραληφθεί, σήμαινε κέρδος για τον κατασκευαστή αλλά και πιο οικονομικά ανεκτά προϊόντα για τους καταναλωτές. Επιπλέον, πολλές γραμμές ελέγχου συνεπάγονται πως το σύστημα θα είναι περισσότερο ευαίσθητο στις ηλεκτρομαγνητικές παρεμβολές(EMI) και στις ηλεκτροστατικές εκκενώσεις(ESD).

Η έρευνα για την αντιμετώπιση των παραπάνω προβλημάτων έγινε από την Philips στην Ολλανδία και κατέληξε σε ένα δισύρματο καλώδιο επικοινωνίας που ονομάστηκε I^2C bus. Το I^2C είναι η συντομογραφία του Inter-IC bus και όπως φαίνεται και από την ονομασία του έχει ως σκοπό να παρέχει επικοινωνιακή ζεύξη μεταξύ ολοκληρωμένων κυκλωμάτων.

Σήμερα, το I^2C bus χρησιμοποιείται σε πολύ περισσότερες εφαρμογές απ' αυτές σε εξαρτήματα ήχου και video. Το I^2C bus είναι γενικά αποδεκτό στην βιομηχανία και έχει υιοθετηθεί από πολλούς κατασκευαστές chip όπως Xicor, ST Microelectronics, Infineon Technologies, Intel, Texas Instruments, Maxim, Atmel, Analog Devices και άλλες εταιρίες.



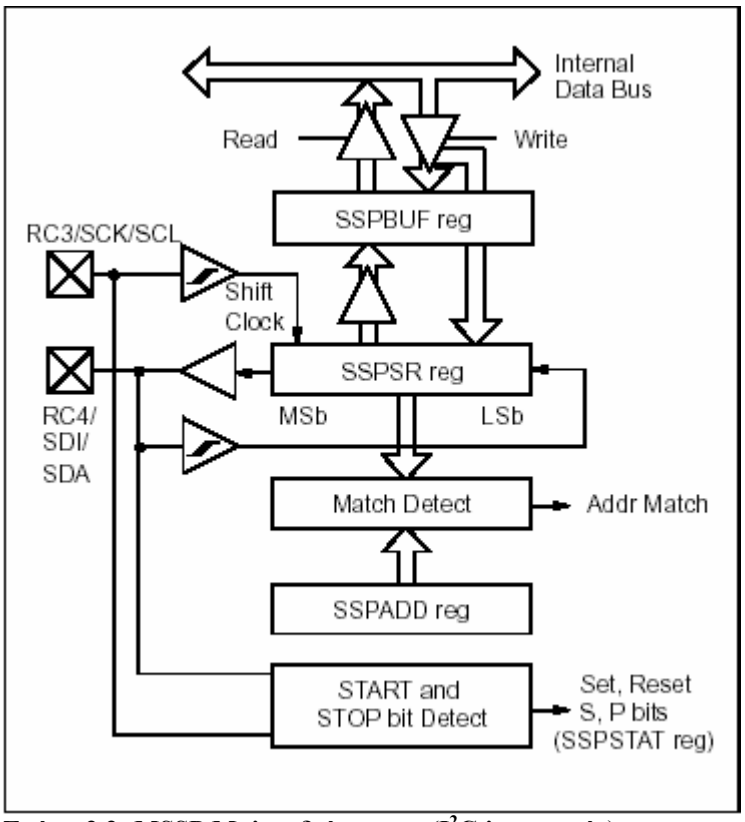
Σχήμα 2.1: Διασύνδεση μεταξύ διαφορετικών ολοκληρωμένων(I^2C MODE)

2.3 Περιγραφή Χαρακτηριστικών του I^2C Bus

2.3.1 Χαρακτηριστικά Μονάδας

Μερικά από τα σημαντικότερα χαρακτηριστικά της μονάδας αυτής είναι:

- Επιλογή χρησιμοποίησης του ως master ή slave
- Υποστήριξη multi-master λειτουργίας, χωρίς να χάνονται μηνύματα κατά την διάρκεια της ανάκτησης ελέγχου από οποιονδήποτε master(arbitration)
- Αυτόματη επαναποστολή αποτυχημένων μηνυμάτων
- Δυνατότητα διευθυνσιοδότησης των 7-bits και των 10-bits
- Κάθε συσκευή που συνδέεται πάνω στο bus έχει τη δικιά της μοναδική διεύθυνση
- Ο slave μπορεί να είναι είτε συσκευή λήψης μόνο είτε συσκευή μετάδοσης με δυνατότητα να λαμβάνει και να στέλνει δεδομένα
- Υποστήριξη γενικής κλήσης(general call addresses) από τον master
- Υποστήριξη επιλογής ταχύτητας μετάδοσης των 8-bit πακέτων μεταξύ του standard clock (SCL) mode που είναι 100Kbps, του fast mode που είναι 400Kbps και του High Speed mode που είναι 3.4Mbps.
- Δεν επηρεάζεται από απότομες μεταβολές της τάσης αλλά και τον θόρυβο
- Επιτρέπει να συνδεθούν 112 συσκευές σε 7-bit διευθυνσιοδότηση και 1024 σε 10-bit διευθυνσιοδότηση



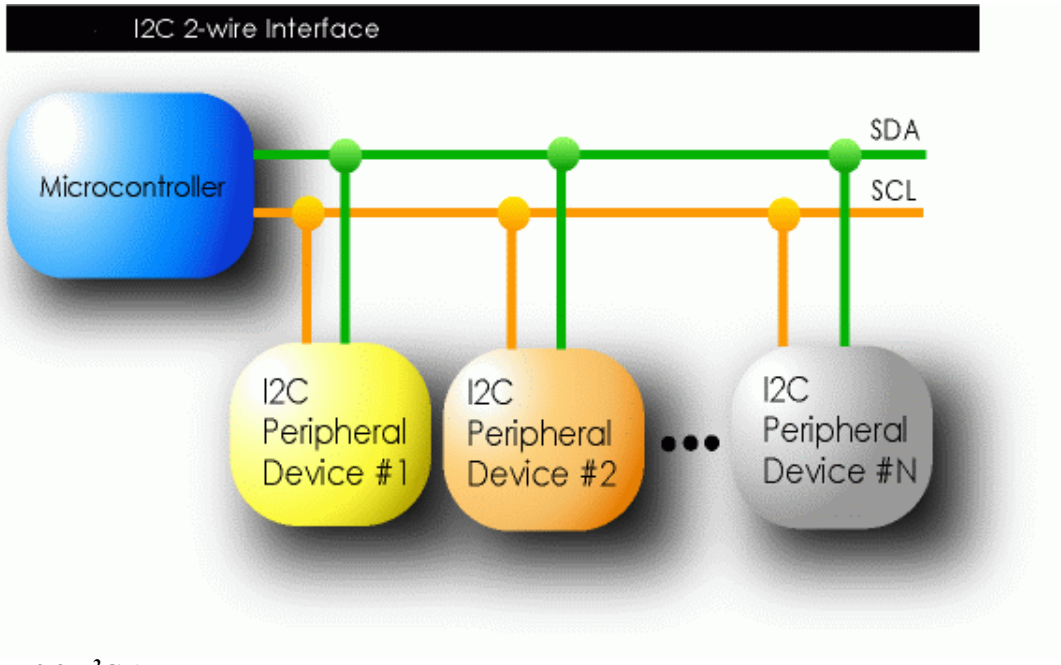
Σχήμα 2.2: MSSP Μπλοκ διάγραμμα (I^2C λειτουργία)

2.3.2 Χαρακτηριστικά του I2C Bus

Το I^2C υλοποιείται με την χρήση δύο καλωδίων διπλής κατεύθυνσης. Τα δύο καλώδια αυτά είναι το SDA, Serial Data, που χρησιμοποιείται για την μεταφορά δεδομένων και το SCL, Serial Clock, για το ρολόι και είναι συνδεδεμένα πάντα σε θετική τροφοδοσία μέσω pull-up αντιστάσεων.

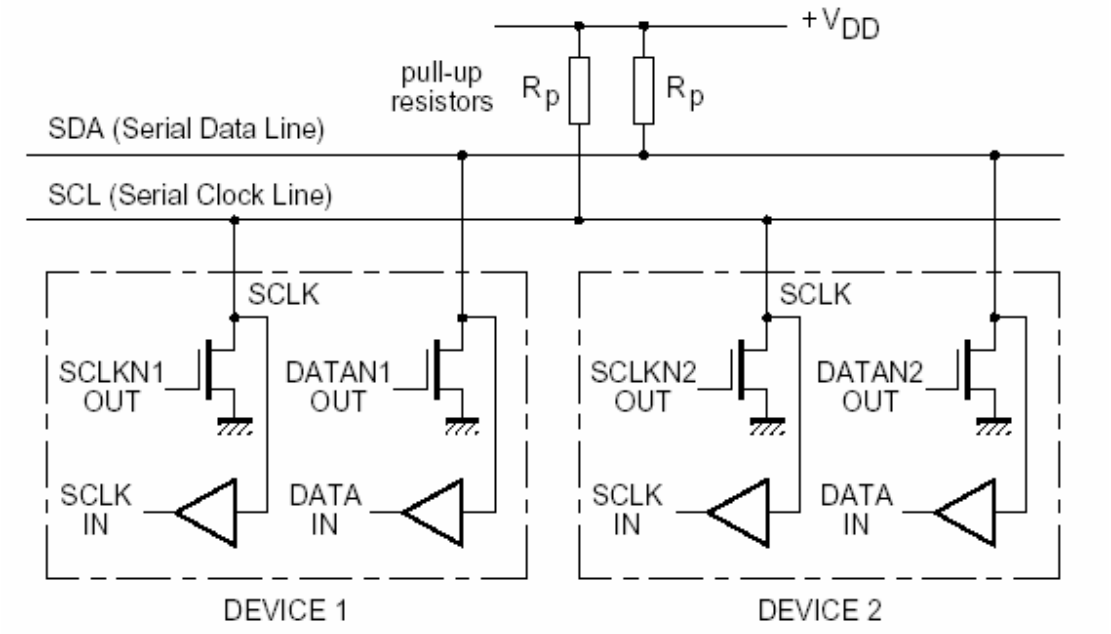
Κάθε συσκευή πάνω στο bus έχει τη δικιά της μοναδική διεύθυνση, καθώς επίσης και το δικαίωμα αποστολής και λήψης δεδομένων από το δίαυλο. Το μήκος του καλωδίου(bus) μπορεί να φτάσει τα 3 με 4 μέτρα αλλά μπορεί και να αυξηθεί με τους λεγόμενους bus extenders έως και 100m.

Επιπλέον κάθε συσκευή πάνω στο bus μπορεί να λειτουργεί είτε ως Master, οπότε αποφασίζει για τις λειτουργίες που επιτελούνται πάνω στο bus, είτε ως slave, οπότε ανταποκρίνεται στις αιτήσεις του Master.
Μια γενική τοπολογία ενός δικτύου I^2C με Master έναν μικροελεγκτή φαίνεται στο επόμενο σχήμα.



Σχήμα 2.3: I^2C Δίκτυο

Εξαιτίας της αμφίδρομης φύσης των καλωδίων, SDA και SCL, τα στάδια εξόδου των ολοκληρωμένων για να μπορούν να αντεπεξέρχονται σε τυχόν συγκρούσεις(collision) χωρίς να καταστρέφονται έχουν στάδια εξόδου ανοικτού συλλέκτη(open collector/drain) έτσι ώστε να διεκπεραιώνουν την καλωδιωμένη-AND(wired-AND) λογική του καλωδίου. Το στάδιο εξόδου του ανοικτού συλλέκτη, όπως φαίνεται στο επόμενο σχήμα, δεν έχει pull-up αντίσταση συνεπώς δεν μπορεί να οδηγήσει την γραμμή σε υψηλή στάθμη. Οι εξωτερικές pull-up αντιστάσεις οδηγούν τις γραμμές SDA, SCL οποτεδήποτε οι γραμμές αυτές δεν οδηγούνται από το τρανζίστορ εξόδου σε υψηλή στάθμη.



Σχήμα 2.4: Τρόπος σύνδεσης ολοκληρωμένων στο I^2C bus

Από την παραπάνω τοπολογία συμπεραίνουμε ότι όταν το bus βρίσκεται σε αδρανή κατάσταση οι δυο γραμμές SDA, SCL, είναι σε υψηλή στάθμη. Το κύκλωμα οδήγησης των διαφόρων συσκευών μπορεί μόνο να θέσει σε μηδενική στάθμη τις γραμμές SDA, SCL ενώ η υψηλή στάθμη δίνεται από τις εξωτερικές pull-up αντιστάσεις. Η ακριβής τιμή των αντιστάσεων δεν παίζει καθοριστικό ρόλο στην λειτουργία ενός συστήματος επικοινωνίας μέσω του πρωτοκόλλου I^2C. Οι τιμές των αντιστάσεων μπορούν να κυμαίνονται από 1k8 (1800 ohms) έως 47k (47000 ohms). Η πιο συχνά χρησιμοποιούμενες τιμές είναι 1k8, 4k7 και 10k. Στο συγκεκριμένο σύστημα χρησιμοποιήθηκαν δύο αντιστάσεις των 4k7. Η απουσία όμως αυτών των αντιστάσεων θα είχε ως αποτέλεσμα οι γραμμές των SCL, SDA να «κρατιούνται» πάντα σε χαμηλό επίπεδο- κοντά στα 0 volts-και το I^2C να μην λειτουργεί. Η τιμή των pull-up αντιστάσεων μαζί με την χωρητικότητα των γραμμών και των εισόδων των συσκευών καθορίζουν την καθυστέρηση στην άνοδο του παλμού και για αυτό θα πρέπει το γινόμενο τους να είναι αρκετά μικρό ώστε να είναι ευδιάκριτο το μέτωπο του παλμού. Για το λόγο αυτό η τιμή των pull-up αντιστάσεων θα πρέπει να είναι αρκετά μικρή. Παράλληλα όμως κάθε συσκευή που προσπαθεί να οδηγήσει μια γραμμή στο μηδέν θα πρέπει να απορροφά ρεύμα λίγο μεγαλύτερο από V_{DD}/R_P. Έτσι είναι κατανοητό ότι η τιμή της αντίστασης R_P δεν μπορεί να είναι όσο μικρή επιθυμούμε.

Τέλος παρατηρούμε ότι οποτεδήποτε δυο συσκευές προσπαθούν ταυτόχρονα να γράψουν σε μια γραμμή, θα υπερισχύει πάντα αυτός που θέτει την γραμμή σε χαμηλή στάθμη. Η ιδιότητα αυτή χρησιμοποιείται για την διαιτησία του διαδρόμου(arbitration) σε περίπτωση πολλών Master συσκευών.

Επίσης ο master και ο slave πρέπει να είναι πάντα σε αντίθετες καταστάσεις (transmitter/receiver) κατά την διάρκεια μιας μετάδοσης δεδομένων. Έτσι μπορούμε να έχουμε τις εξής καταστάσεις πάνω στο I2C bus:

- Master-πομπό(Transmitter) και Slave-δέκτη(receiver)
- Master- δέκτη(receiver)και Slave- πομπό(Transmitter)
- Multi-master περιβάλλον

2.3.3 I^2C Bus Επικοινωνία

Σύμφωνα λοιπόν με όσα έχουμε αναφέρει για να επιτευχθεί η μετάδοση των επιθυμητών δεδομένων είναι απαραίτητη η παρουσία ενός τουλάχιστον Master. Ο Master του bus είναι το εκείνο ολοκληρωμένο που δίνει τις εντολές στα υπόλοιπα ολοκληρωμένα. Ελέγχει την γραμμή του ρολογιού (SCL) και «κηρύσσει» την έναρξη των επιθυμητών διαδικασιών ενώ κάθε άλλο ολοκληρωμένο θεωρείται ως slave.

Θεωρείται σκόπιμο στο κομμάτι αυτό να γίνει μια αναφορά στην ορολογία του I^2C Bus έτσι ώστε να είναι δυνατή η καλύτερη επεξήγηση και κατανόηση του πρωτοκόλλου.

- I^2C Bus Ορολογία

Transmitter (πομπός) - Το ολοκληρωμένο που στέλνει δεδομένα στο καλώδιο. Ο πομπός μπορεί να είναι το ολοκληρωμένο που στέλνει δεδομένα στο bus για λογαριασμό του(Master-Transmitter) ή κατά απαίτηση άλλων ολοκληρωμένων (Slave-Transmitter).

Receiver (Δέκτης) - Το ολοκληρωμένο που λαμβάνει δεδομένα από το καλώδιο.

Master - Το ολοκληρωμένο που αρχικοποιεί την μεταφορά, παράγει τους παλμούς του ρολογιού και τερματίζει την μεταφορά. Ο Master μπορεί να είναι είτε πομπός είτε δέκτης.

Slave - Η συσκευή που διευθυνσιοδοτείται από τον Master. Ο slave μπορεί να είναι είτε πομπός είτε δέκτης.

Multi-master – Η ικανότητα να συνυπάρχουν στο καλώδιο πάνω από ένας master ταυτόχρονα χωρίς συγκρούσεις ή απώλειες δεδομένων

Arbitration (διαιτησία) – Η διαδικασία που εξουσιοδοτεί ποιος master θα ελέγχει το καλώδιο κάθε χρονική στιγμή

Synchronization – Η προκαθορισμένη διαδικασία που συγχρονίζει τους παλμούς του ρολογιού που παρέχονται από δύο ή περισσότερους masters

SDA (Serial DAta) – Η γραμμή μέσω της οποίας μεταφέρονται τα δεδομένα

SCL (Serial CLock) - Η γραμμή μέσω της οποίας μεταφέρονται οι παλμοί του ρολογιού

Οι πιθανές καταστάσεις που μπορούν να συμβαίνουν πάνω στο bus και δηλώνουν το σύνολο των σημείων που ανταλλάσσουν οι συσκευές μεταξύ τους ώστε να μεταφερθούν τα δεδομένα είναι οι εξής:

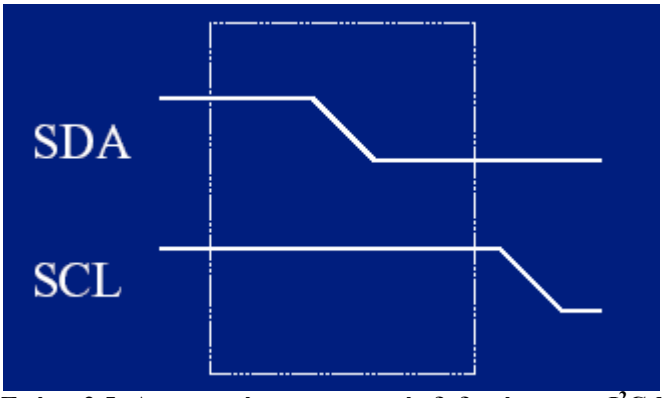
- START Data Transfer (S)
- Acknowledged (Επιβεβαίωση) (ACK)
- Μη-Επιβεβαίωση(NOT-ACK) (NACK)
- Ανάγνωση / Εγγραφή Δεδομένων (READ/WRITE)
- RESTART (R)
- REPEATED START (Rs)
- STOP Data Transfer (P)

2.3.3.1 START Data Transfer (S)

Στην κατάσταση Start σηματοδοτείται η εκκίνηση της μεταφοράς δεδομένων που θέλουμε. Η κατάσταση αυτή σηματοδοτείται μόνο από τον master και σε περίπτωση multi-master περιβάλλοντος από εκείνον τον master που θα εκπέμψει πρώτος. Για να μπει το καλώδιο στην Start κατάσταση πρέπει αρχικά να είναι σε αδρανή (idle) κατάσταση και όταν τελικά μπει στην Start κατάσταση θεωρείται απασχολημένο και δεν μπορεί να το οικειοποιηθεί άλλος master ή οποιαδήποτε άλλη συσκευή που είναι συνδεδεμένη πάνω στο καλώδιο.

Επίσης την κατάσταση αυτή μπορούμε να την συναντήσουμε όχι μόνο στην αρχή αλλά και κατά την διάρκεια της μεταφοράς των δεδομένων.

Η μορφή του σήματος αυτού φαίνεται στο παρακάτω σχήμα:

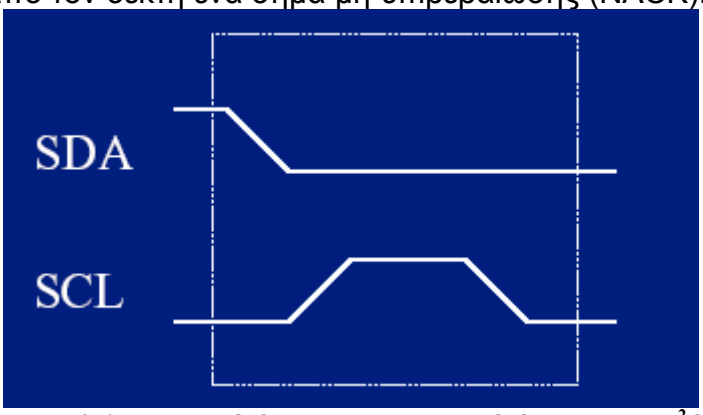


Σχήμα 2.5: Αρχικοποίηση μεταφοράς δεδομένων στο I^2C bus

Όταν λοιπόν το bus δεν χρησιμοποιείται, οι εξωτερικές pull-up αντιστάσεις κρατούν τις εξωτερικές γραμμές σε υψηλή στάθμη σήματος. Όπως παρατηρούμε από το παραπάνω σχήμα ο master που θα εκπέμψει την start κατάσταση θα πρέπει να θέσει την SDA γραμμή από υψηλή σε χαμηλή στάθμη στη διάρκεια που η SCL γραμμή θα είναι σε υψηλή στάθμη.

2.3.3.2 Acknowledged (ACK)

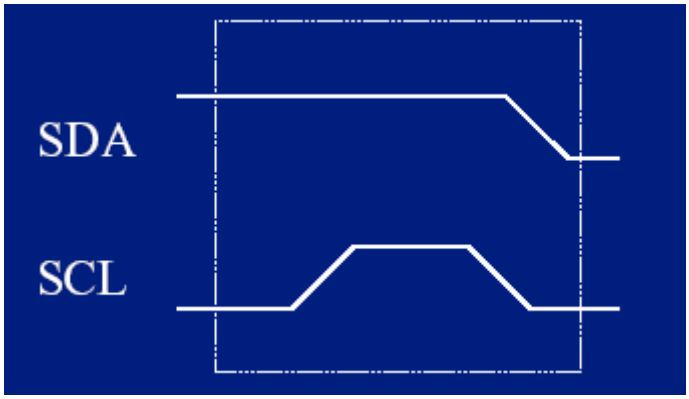
Στο τέλος της διαδικασίας αποστολής ενός byte δεδομένων στον δέκτη του σήματος, αποστέλλεται από αυτόν ένα σήμα επιβεβαίωσης(ACK) ότι έλαβε αυτό το byte. Στην περίπτωση που το byte δεδομένων που αποστέλλεται είναι το τελευταίο της όλης μετάδοσης ή ο δέκτης δεν μπορεί να λάβει άλλα δεδομένα ή γενικά κάτι πήγε στραβά στην όλη μετάδοση, τότε αποστέλλεται από τον δέκτη ένα σήμα μη-επιβεβαίωσης (NACK).



Σχήμα 2.6: Σήμα επιβεβαίωσης μεταφοράς δεδομένων στο I^2C bus

Όπως φαίνεται από το παραπάνω σχήμα τα δεδομένα επιβεβαιώνονται όταν ο δέκτης «σπρώχνει» την SDA γραμμή στην χαμηλή στάθμη στην διάρκεια ενός παλμού του ρολογιού(SCL).

Στην αντίθετη περίπτωση της μη-επιβεβαίωσης(NACK), στην διάρκεια ενός παλμού του ρολογιού(SCL), ο δέκτης «κρατάει» την SDA γραμμή σε υψηλή στάθμη, κάτι που φαίνεται και από το παρακάτω σχήμα.

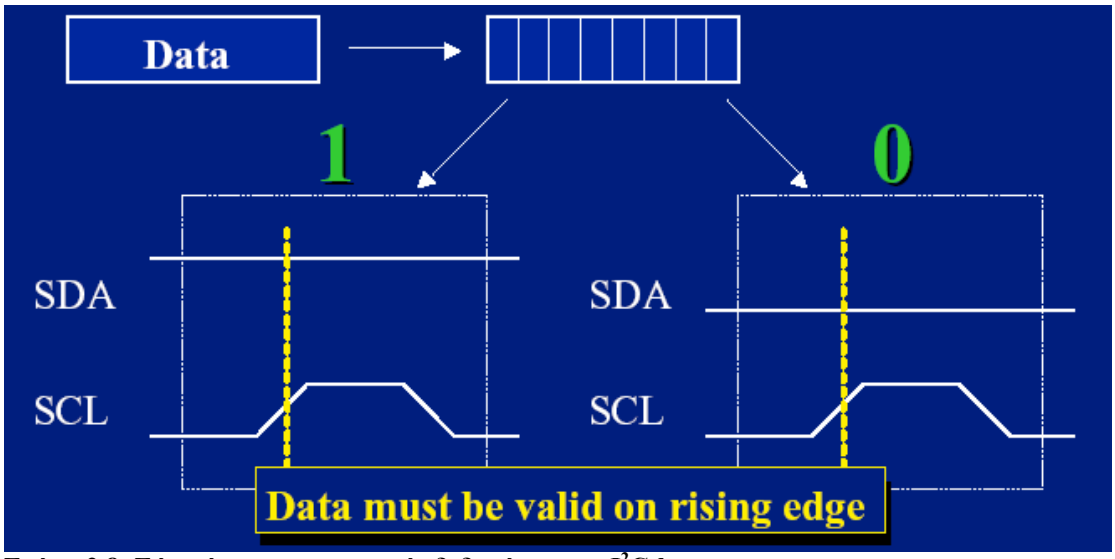


Σχήμα 2.7: Σήμα μη-επιβεβαίωσης μεταφοράς δεδομένων στο I^2C bus

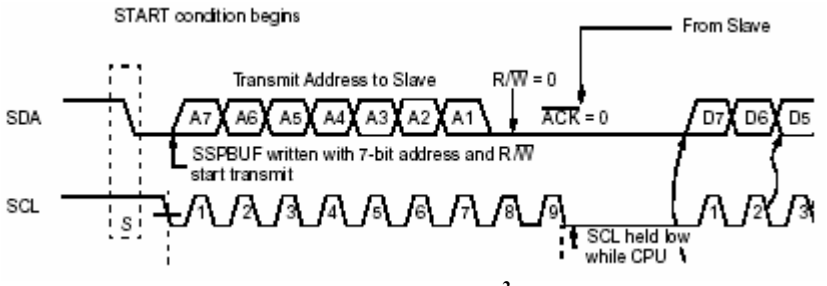
2.3.3.3 Ανάγνωση /Εγγραφή ενός byte (READ/WRITE)

Τόσο η ανάγνωση όσο και η εγγραφή είναι γνωστά μετά την αρχικοποίηση μιας μεταφοράς δεδομένων. Η μεταφορά δεδομένων γίνεται σε πακέτα των 8 bits. Τα δεδομένα θεωρούνται έγκυρα όταν η SCL γραμμή είναι σε υψηλή στάθμη ενώ όταν η SCL δεν είναι σε υψηλή στάθμη δεν επιτρέπεται να αλλάξουν, όπως φαίνεται και από το σχήμα 2.8.

Στην περίπτωση τώρα που ο master θέλει να κάνει εγγραφή (write) δεδομένων στον slave, τοποθετεί τα προς εγγραφή δεδομένα πάνω στην SDA γραμμή και παράγει κατάλληλους παλμούς ρολογιού. Μετά των 8^o παλμό εάν όλα πήγαν καλά, ο slave παίρνει την γραμμή SDA στην κατοχή του και την μηδενίζει, δηλαδή παράγει ένα σήμα ACK. Στην περίπτωση που ο slave δεν ανταποκρίνεται με το παλμό ACK καταλαβαίνουμε αμέσως ότι υπάρχει κάποιο σφάλμα και το λογισμικό θα πρέπει να προβλέπει την διαδικασία που θα ακολουθείται σε μια τέτοια περίπτωση.

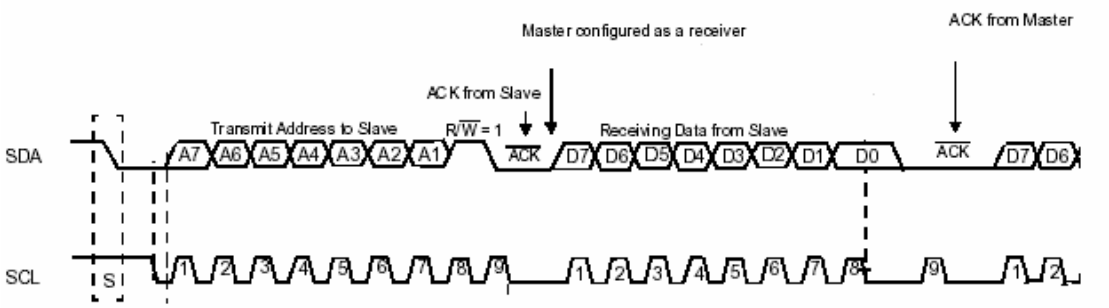


Σχήμα 2.8: Σήμα έγκυρης μεταφοράς δεδομένων στο I^2C bus



Σχήμα 2.9: Σήμα εγγραφής δεδομένων στο I²C bus

Στην περίπτωση που ο master θέλει να κάνει ανάγνωση (read) δεδομένων από τον slave, παράγει κατάλληλους παλμούς ρολογιού και διαβάζει τα δεδομένα στην SDA γραμμή.

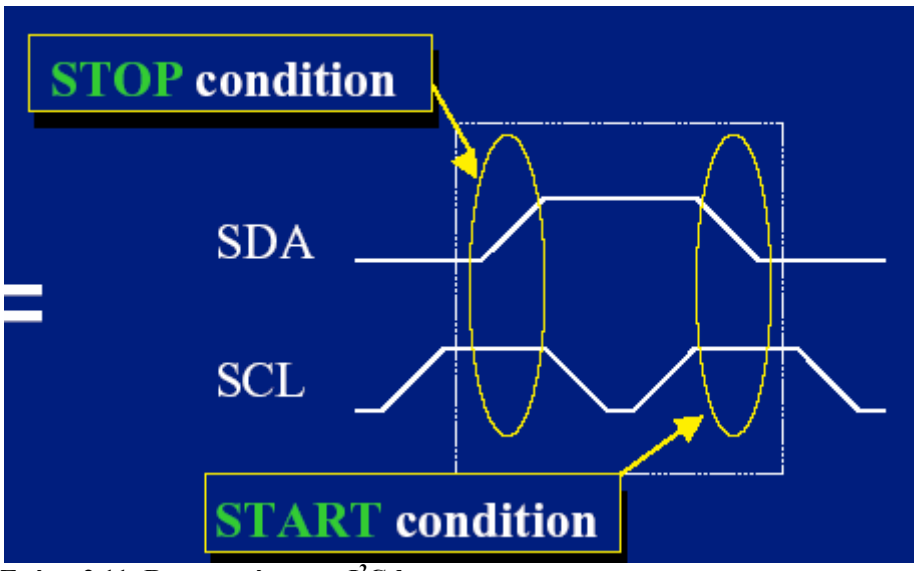


Σχήμα 2.10: Σήμα ανάγνωσης δεδομένων στο I²C bus

2.3.3.4 Restart (R)

Η restart κατάσταση δεν είναι τίποτε άλλο από μια stop κατάσταση που ακολουθείται αμέσως από μια start κατάσταση.

Σε αυτήν την κατάσταση μπορούμε να έχουμε και αλλαγή της κατεύθυνσης των δεδομένων καθώς επίσης και αλλαγή του ολοκληρωμένου που μεταδίδει ή στέλνει τα δεδομένα(νέα διεύθυνση-νέα συσκευή).



Σχήμα 2.11: Restart σήμα στο I²C bus

2.3.3.5 REPEATED START (Rs)

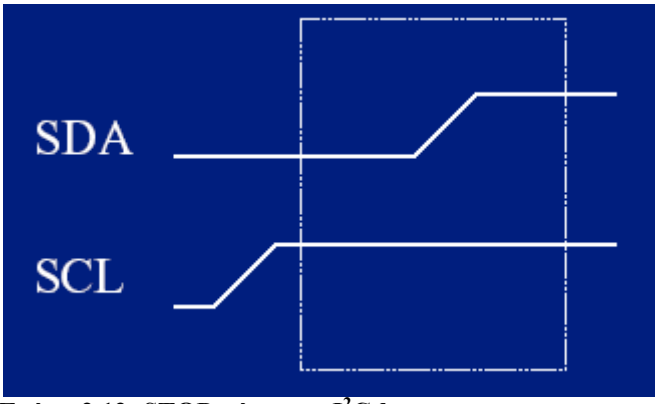
Η repeated start κατάσταση δεν είναι τίποτε άλλο από μια κατάσταση που επιτρέπει στον master να εκτελέσει μια start κατάσταση χωρίς να χρειάζεται η δήλωση κάποιας stop κατάστασης. Ο έλεγχος του bus από τον master δεν χάνεται ποτέ.

Σε αυτήν την κατάσταση μπορούμε να έχουμε και αλλαγή της κατεύθυνσης των δεδομένων καθώς επίσης και αλλαγή του ολοκληρωμένου που μεταδίδει ή στέλνει τα δεδομένα(νέα διεύθυνση-νέα συσκευή).

2.3.3.6 STOP Data Transfer (P)

Η κατάσταση Stop σηματοδοτεί το τέλος μιας μεταφοράς δεδομένων. Μαζί με την κατάσταση Start έχει την μεγαλύτερη προτεραιότητα και μπορεί να εκπεμφθεί σε οποιαδήποτε χρονική στιγμή της μεταφοράς δεδομένων. Με το που σηματοδοτηθεί η stop κατάσταση το καλώδιο απελευθερώνεται και άλλες συσκευές μπορούν να το χρησιμοποιήσουν για να μεταφέρουν δεδομένα.

Η μορφή του σήματος Stop φαίνεται στο παρακάτω σχήμα:

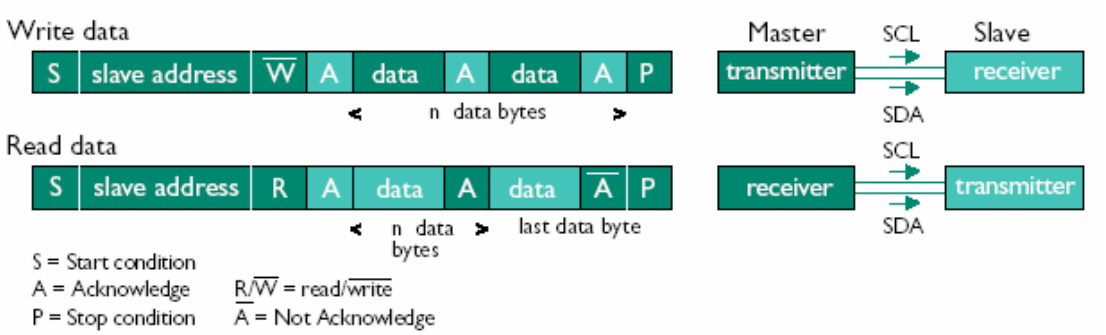


Σχήμα 2.12: STOP σήμα στο I^2C bus

Όπως παρατηρούμε από το παραπάνω σχήμα ο master που θα εκπέμψει την stop κατάσταση θα πρέπει να θέσει την SDA γραμμή από χαμηλή σε υψηλή στάθμη στη διάρκεια που η SCL γραμμή θα είναι σε υψηλή στάθμη. Έτσι όταν ολοκληρωθεί η stop κατάσταση οι SDA, SCL γραμμές θα είναι και οι δύο σε υψηλή στάθμη και το καλώδιο μπαίνει σε αδρανή(idle) κατάσταση.

2.4 Σχηματισμός και Διάταξη Μηνυμάτων Μετάδοσης

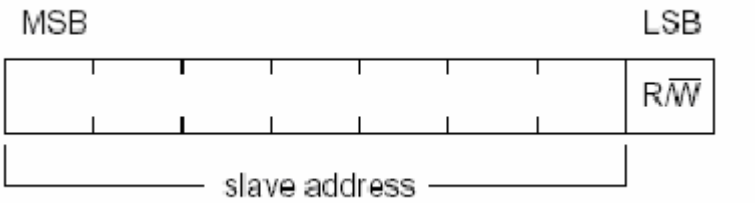
Όπως έχουμε ήδη αναφέρει η αποστολή μηνυμάτων μέσω της sda γραμμής γίνεται με πακέτα των 8 bits, όπου το καθένα επιβεβαιώνεται από τον receiver στον ένατο παλμό του ρολογιού ενώ ο αριθμός των bytes που μπορούν να μεταφερθούν είναι απεριόριστος.



Σχήμα 2.13: Τυπικό σχηματικό μεταφοράς δεδομένων στο I²C bus

Κάθε μήνυμα αρχικοποιείται με την Start κατάσταση και τερματίζεται με την κατάσταση Stop. Ο αριθμός των δεδομένων που μεταφέρονται μεταξύ της Start και Stop κατάστασης καθορίζονται από τον master και τα bytes του κάθε μηνύματος μπορούν να έχουν ειδική σημασία όπως τον καθορισμό της διεύθυνσης ενός slave πάνω στο καλώδιο.

Αμέσως μετά το Start το πρώτο byte είναι της διεύθυνσης του slave από το οποίο τα πρώτα επτά bit καθορίζουν την διεύθυνση του slave ενώ το τελευταίο (LSB) R/W' καθορίζει αν ο master θα είναι transmitter ή receiver. Σε περίπτωση που επιθυμούμε ο master να είναι transmitter το R/W'=0, ενώ αν είναι receiver το R/W'=1.Τα δεδομένα μεταφέρονται με το περισσότερο σημαντικό bit(MSB) πρώτα.



Στον επόμενο (9^{o})παλμό που παράγει ο master, ο slave υποχρεώνεται να στείλει σήμα επιβεβαίωσης(A) για να δείξει ότι έλαβε το byte. Αν ο slave δεν μπορεί να λάβει ή να μεταδώσει άλλο ολοκληρωμένο byte δεδομένων, για οποιοδήποτε λόγο όπως παραδείγματος χάριν γιατί εκτελεί μια εσωτερική διακοπή(interrupt), μπορεί να κρατήσει την SCL γραμμή σε χαμηλή στάθμη έτσι ώστε να αναγκάσει τον master να περιμένει. Όταν ο slave είναι έτοιμος να λάβει ή να μεταδώσει κι άλλο byte απελευθερώνει την SCL γραμμή και η μεταφορά των δεδομένων(8-bit πακέτα) συνεχίζεται.

Ο συσχετιζόμενος με την επιβεβαίωση παλμός παράγεται πάντα από τον Master μετά την αποστολή δεδομένων(data) ενώ με την λήψη ή αποστολή του τελευταίου byte στέλνεται με τον 9^{o} παλμό ένα NACK.

2.5 Καταχωρητές Κατάστασης και Ελέγχου του I²C

Το I²C πρωτόκολλο για την σωστή λειτουργία του χρησιμοποιεί έξι καταχωρητές. Αυτοί είναι:

- SSPSTAT (Status Register)

- SSPCON (Control Register)
- SSPCON2(Control Register)
- SSPBUF (Serial Receive/Transmit Buffer)
- SSPSR (Shift Register- όχι άμεσα προσβάσιμος)
- SSPADD (Address Register)

Οι SSPCON, SSPCON2 και SSPSTAT είναι οι καταχωρητές ελέγχου και κατάστασης της I^2C λειτουργίας. Οι SSPCON, SSPCON2 είναι αναγνώσιμοι και εγγράψιμοι ενώ από τα 8-bit του SSPSTAT τα δύο πρώτα bits(7,6) είναι αναγνώσιμα/εγγράψιμα και τα έξη τελευταία μόνο αναγνώσιμα.

Ο SSPSR είναι καταχωρητής ολίσθησης και χρησιμοποιείται για να ολισθαίνει δεδομένα μέσα ή έξω από το ολοκληρωμένο. Ο SSPBUF είναι ο καταχωρητής όπου τα δεδομένα γράφονται σε αυτόν ή διαβάζονται από αυτόν(αποθηκεύονται). Ο SSPADD είναι ο καταχωρητής που κρατάει την διεύθυνση της slave συσκευής-ολοκληρωμένου όταν η SSP βρίσκεται σε I^2C slave λειτουργία. Όταν η SSP βρίσκεται σε master λειτουργία τα επτά χαμηλότερα bits του SSPADD δηλώνουν τον ρυθμό με τον οποίο θα μεταδίδονται τα δεδομένα(baud rate generator).

Σε λειτουργίες λήψης δεδομένων, οι SSPSR και SSPBUF μαζί δημιουργούν έναν δέκτη διπλής απομόνωσης. Όταν ο SSPSR λαμβάνει ένα ολόκληρο byte, αμέσως μεταφέρεται στον SSPBUF και η σημαία SSPIF γίνεται 1 (interrupt flag) και η διακοπή λαβαίνει χώρα.

Η αναλυτική περιγραφή των σημαντικών αυτών καταχωρητών για την λειτουργία του I^2C ακολουθεί ευθύς αμέσως.

REGISTER 1: SSPSTAT: MSSP STATUS REGISTER (I^2C MODE) (ADDRESS 94h)

R/W-0	R/W-0	R-0	R-0	R-0	R-0	R-0	R-0
SMP	CKE	D/A'	P	S	R/W'	UA	BF
bit 7							bit 0

bit 7 **SMP:** Bit ελέγχου Slew Rate
Σε Master ή Slave λειτουργία :
1= To slew rate control απενεργοποιείται για την λειτουργία standard speed (100 kHz and 1 MHz)
0= To slew rate control ενεργοποιείται για την λειτουργία high speed (400 kHz)

bit 6 **CKE:** Bit επιλογής SMBus
Σε Master ή Slave λειτουργία :
1 = Ενεργοποιεί τα ειδικά δεδομένα εισαγωγής του SMBus
0 = Απενεργοποιεί τα ειδικά δεδομένα εισαγωγής του SMBus

bit 5 **D/A':** Data/Address bit
Σε Master λειτουργία:
Δεσμευμένο
Σε Slave λειτουργία :
1 = Δείχνει ότι το τελευταίο byte που μεταδόθηκε ή ελήφθη ήταν δεδομένα
0 = Δείχνει ότι το τελευταίο byte που μεταδόθηκε ή ελήφθη ήταν διεύθυνση

bit 4 **P:** STOP bit
1 = Δείχνει ότι ένα STOP bit έχει ανιχνευτεί τελευταίο
0 = STOP bit δεν έχει ανιχνευτεί τελευταίο
Σημείωση: Αυτό το bit μηδενίζεται στο RESET and όταν το SSPEN μηδενίζεται.

bit 3 **S:** START bit
1 = Δείχνει ότι ένα START bit έχει ανιχνευτεί τελευταίο

0 = το START bit δεν έχει ανιχνευτεί τελευταίο
Σημείωση: Αυτό το bit μηδενίζεται στο RESET και όταν το SSPEN μηδενίζεται.

bit 2 **R/W':** Read/Write bit πληροφορίας (I2C λειτουργία μόνο)
Σε Slave λειτουργία :
1 = Read
0 = Write
Σημείωση: Αυτό το bit κρατά το R/W bit πληροφορίας που ακολουθεί το τελευταίο ταίριασμα της διεύθυνσης. Αυτό το bit είναι έγκυρο μόνο από το ταίριασμα της διεύθυνσης μέχρι το επόμενο START bit, STOP bit, ή not ACK bit.
Σε Master λειτουργία:
1 = Η μετάδοση είναι σε εξέλιξη
0 = Η μετάδοση δεν είναι σε εξέλιξη
Σημείωση: Κάνοντας τη λογική πράξη OR αυτού του bit με τα SEN, RSEN, PEN, RCEN, ή ACKEN θα φανεί αν το MSSP είναι σε IDLE λειτουργία.

bit 1 **UA:** Update Address (10-bit Slave λειτουργία μόνο)
1 = Δείχνει ότι ο χρήστης πρέπει να ανανεώσει την διεύθυνση στον καταχωρητή SSPADD
0 = Η διεύθυνση δεν χρειάζεται να ανανεωθεί

bit 0 **BF:** Bit Κατάστασης Buffer
Σε λειτουργία Μετάδοσης:
1 = Η λήψη ολοκληρώθηκε, ο SSPBUF είναι γεμάτος
0 = Η λήψη δεν ολοκληρώθηκε, ο SSPBUF είναι άδειος
Σε λειτουργία Λήψης:
1 = Η μετάδοση των δεδομένων είναι σε εξέλιξη (δεν περιλαμβάνει τα ACK και STOP bits), ο SSPBUF είναι γεμάτος
0 = Η μετάδοση των δεδομένων ολοκληρώθηκε(δεν περιλαμβάνει τα ACK και STOP bits), ο SSPBUF είναι άδειος

Υπόμνημα:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 2: SSPCON: MSSP CONTROL REGISTER1 (I^2C MODE) (ADDRESS 14h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0
bit 7							bit 0

bit 7 **WCOL:** bit ανίχνευσης σύγκρουσης κατά την εγγραφή
Σε λειτουργία μετάδοσης του Master:
1 = Επιχειρήθηκε εγγραφή στον καταχωρητή SSPBUF όσο οι I^2C συνθήκες δεν ήταν έτοιμες να ξεκινήσουν μια μετάδοση δεδομένων. (Πρέπει να μηδενίζεται από το software.)
0 = Δεν σημειώθηκε καμία σύγκρουση
Σε λειτουργία μετάδοσης από τον slave:
1 = = Επιχειρήθηκε εγγραφή στον καταχωρητή SSPBUF όσο αυτός ακόμη μετέδιδε την προηγούμενη λέξη. (Πρέπει να μηδενίζεται από το software.)
0 = Δεν σημειώθηκε καμία σύγκρουση
Σε λειτουργία λήψης (Master ή Slave λειτουργία):
Εδώ αυτό το bit «δεν έχει σημασία».

bit 6 **SSPOV:** Bit ένδειξης υπερχείλισης κατά την λήψη
Σε λειτουργία λήψης:
1 =Λήψη ενός νέου byte ενώ στον καταχωρητή SSPBUF βρίσκονται ακόμα προηγούμενα δεδομένα. (Πρέπει να μηδενίζεται από το software.)
0 = Δεν σημειώθηκε καμία υπερχείλιση
Σε λειτουργία Μετάδοσης:

Σε λειτουργία μετάδοσης αυτό το bit «δεν έχει σημασία».
.

bit 5 **SSPEN:** Bit ενεργοποίησης της μονάδας SSP
1 = Ενεργοποιεί την σειριακή θύρα και δηλώνει τα SDA and SCL pins σαν pins της σειριακής θύρας
0 = Απενεργοποιεί την σειριακή θύρα και δηλώνει τα SDA and SCL pins σαν I/O pins
Σημείωση: Όταν ενεργοποιείται, τα SDA και SCL pins πρέπει να είναι σωστά δηλωμένα σαν input ή output.

bit 4 **CKP:** Bit ελέγχου απελευθέρωσης SCK
Σε λειτουργία slave:
1 = Απελευθέρωση ρολογιού
0 = Κρατά το ρολόι σε χαμηλή στάθμη (clock stretch). (Χρησιμοποιείται για να διασφαλίσει τον σωστό χρονισμό των δεδομένων.)
Σε λειτουργία master:
Δεν χρησιμοποιείται σε αυτή την λειτουργία.

bit 3-0 **SSPM3:SSPM0:** Bits ελέγχου της λειτουργίας της θύρας SSP
1111 = I^2C Slave λειτουργία, 10-bit διεύθυνση με START και STOP ενεργοποιημένα bit διακοπών
1110 = I^2C Slave λειτουργία, 7-bit διεύθυνση με START και STOP ενεργοποιημένα bit διακοπών
1011 = I^2C Firmware ελεγχόμενη λειτουργία master (O Slave είναι ΑΔΡΑΝΗΣ)
1000 = I^2C Master λειτουργία, clock = FOSC / (4 * (SSPADD+1))
0111 = I^2C Slave λειτουργία, 10-bit address
0110 = I^2C Slave λειτουργία, 7-bit address
Σημείωση: Συνδυασμοί bit που δεν αναφέρονται παραπάνω είναι είτε δεσμευμένοι, ή λειτουργούν σε SPI λειτουργία μόνο.

Υπόμνημα:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

REGISTER 3: SSPCON2: MSSP CONTROL REGISTER2 (I^2C MODE) (ADDRESS 91h)

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
GCEN	ACKSTAT	ACKDT	ACKEN	RCEN	PEN	RSEN	SEN
bit 7							bit 0

bit 7 **GCEN:** Bit ενεργοποίησης Γενικής Κλήσης(General Call) (Λειτουργία Slave μόνο)
1 = Ενεργοποιεί διακοπές όταν η διεύθυνση Γενικής Κλήσης (0000h) λαμβάνεται στον SSPSR
0 = Απενεργοποίηση διεύθυνσης Γενικής Κλήσης

bit 6 **ACKSTAT:** Bit που δείχνει αν έχουμε επιβεβαίωση (Μόνο σε λειτουργία Μετάδοσης του Master)
1 = Η επιβεβαίωση από τον slave δεν έχει ληφθεί
0 = Η επιβεβαίωση από τον slave έχει ληφθεί

bit 5 **ACKDT:** Bit που δείχνει αν έχουμε Επιβεβαίωση των δεδομένων(Μόνο σε λειτουργία Λήψης του Master)
1 = Δεν υπάρχει επιβεβαίωση
0 = Υπάρχει επιβεβαίωση
Σημείωση: Τιμή που θα μεταδοθεί όταν ο χρήστης αρχικοποιήσει μια ακολουθία Επιβεβαίωσης στο τέλος μιας λήψης δεδομένων.

bit 4 **ACKEN:** Bit ενεργοποίησης ακολουθίας Επιβεβαιώσεων(Μόνο σε λειτουργία Λήψης του Master)
1 = Αρχικοποίηση ακολουθίας επιβεβαιώσεων πάνω στα SDA και SCL pins, και μετάδοση ACKDT data bit.

Μηδενίζεται αυτόματα από το hardware.
0 = Ακολουθία επιβεβαιώσεων ΑΔΡΑΝΗΣ(IDLE)

bit 3 **RCEN:** Receive Enable bit (Λειτουργία Master μόνο)
1 = Ενεργοποιεί για το I²C την λειτουργία λήψης
0 = Η λειτουργία λήψης είναι ΑΔΡΑΝΗΣ(IDLE)

bit 2 **PEN:** Bit ενεργοποίησης κατάστασης STOP (Λειτουργία Master μόνο)
1 = Αρχικοποιεί την STOP κατάσταση στα SDA και SCL pins. Μηδενίζεται αυτόματα από το hardware.
0 = Η STOP κατάσταση είναι ΑΔΡΑΝΗΣ(IDLE)

bit 1 **RSEN:** Bit ενεργοποίησης κατάστασης Repeated START (Λειτουργία Master μόνο)
1 = Αρχικοποιεί την Repeated START κατάσταση στα SDA και SCL pins.
Μηδενίζεται αυτόματα από το hardware.
0 = Η Repeated START κατάσταση είναι ΑΔΡΑΝΗΣ(IDLE)

bit 0 **SEN:** Bit ενεργοποίησης κατάστασης START
Σε λειτουργία Master:
1 = Αρχικοποιεί την START κατάσταση στα SDA και SCL pins. Μηδενίζεται αυτόματα από το hardware.
0 = Η START κατάσταση είναι ΑΔΡΑΝΗΣ(IDLE)
Σε λειτουργία Slave:
1 = Ενεργοποιείται το Clock stretching για τον Slave που μεταδίδει και για τον Slave που λαμβάνει (ενεργοποίηση stretch)
0 = Ενεργοποιείται το Clock stretching για τον Slave που μεταδίδει μόνο(Συμβατότητα PIC16F87X)
Σημείωση: Για τα bits ACKEN, RCEN, PEN, RSEN, SEN: Αν η I²C μονάδα δεν είναι αδρανής (IDLE), αυτό το bit ίσως να μην μπορεί να τεθεί και ο SSPBUF μπορεί να μην είναι εγγράψιμος (ή το να γράψεις στον SSPBUF να μην είναι εφικτό).

Υπόμνημα:			
R = Readable bit	W = Writable bit	U = Unimplemented bit, read as '0'	
- n = Value at POR	'1' = Bit is set	'0' = Bit is cleared	x = Bit is unknown

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on POR, BOR	Value on all other resets
0Bh, 8Bh, 10Bh, 18Bh	INTCON	GIE	PEIE	T0IE	INTE	RBIE	T0IF	INTF	RBIF	0000 000x	0000 000u
0Ch	PIR1	PSPIF(1)	(2)	RCIF	TXIF	SSPIF	CCP1IF	TMR2IF	TMR1IF	0000 0000	0000 0000
8Ch	PIE1	PSPIE(1)	(2)	RCIE	TXIE	SSPIE	CCP1IE	TMR2IE	TMR1IE	0000 0000	0000 0000
13h	SSPBUF	Synchronous Serial Port Receive Buffer/Transmit Register								xxxx xxxx	uuuu uuuu
93h	SSPADD	Synchronous Serial Port (I²C mode) Address Register								0000 0000	0000 0000
14h	SSPCON	WCOL	SSPOV	SSPEN	CKP	SSPM3	SSPM2	SSPM1	SSPM0	0000 0000	0000 0000
94h	SSPSTAT	SMP(3)	CKE(3)	D/$\overline{A}$	P	S	R/$\overline{W}$	UA	BF	0000 0000	0000 0000
87h	TRISC	PORTC Data Direction register								1111 1111	1111 1111

Legend: x = unknown, u = unchanged, - = unimplemented locations read as '0'.
Shaded cells are not used by SSP module in SPI mode.

Σχήμα 2.14: Καταχωρητές που έχουν σχέση με την I²C λειτουργία

2.6 Επικοινωνία Master σε Single-Master περιβάλλον

Όπως έχουμε ήδη αναφέρει η χρησιμότητα αυτού του πρωτοκόλλου θα αξιοποιηθεί στην μεταφορά δεδομένων μεταξύ δυο μικροελεγκτών. Έτσι λοιπόν στην περίπτωση αυτή θα έχουμε έναν master μικροελεγκτή και έναν slave.

Η λειτουργία master ενεργοποιείται θέτοντας και μηδενίζοντας τα κατάλληλα SSPM bits στον SSPCON και θέτοντας το SSPEN bit ίσο με ένα. Οι SDA, SCL γραμμές χειρίζονται από το MSSP(Master Synchronous Serial Port) hardware. Πριν επιλέξουμε οποιαδήποτε λειτουργία του I^2C πρέπει τα SDA, SCL pins να δηλωθούν σαν inputs θέτοντας τα κατάλληλα TRIS bits.

Μόλις η master λειτουργία ενεργοποιηθεί, ο χρήστης έχει έξη επιλογές:

- Δήλωση μιας Start κατάστασης πάνω στα SDA, SCL
- Δήλωση μιας Repeated Start κατάστασης πάνω στα SDA, SCL
- Αρχικοποίηση μετάδοσης δεδομένων/διεύθυνσης, γράφοντας στον καταχωρητή SSPBUF
- Υποχρέωση της I^2C θύρας να λάβει δεδομένα
- Παραγωγή μιας κατάστασης Επιβεβαίωσης(ACK) αφότου έχει ληφθεί κάποιο byte ή παραγωγή μιας κατάστασης Μη-Επιβεβαίωσης (NACK) αν το byte που λαμβάνεται είναι το τελευταίο
- Παραγωγή μιας Stop κατάστασης πάνω στα SDA, SCL

Ακόμη τα παρακάτω γεγονότα έχουν ως αποτέλεσμα την ενεργοποίηση του SSP Interrupt Flag bit, SSPIF(τίθεται ίσο με 1), το οποίο βρίσκεται στον καταχωρητή PIR1(ADDRESS 0Ch):

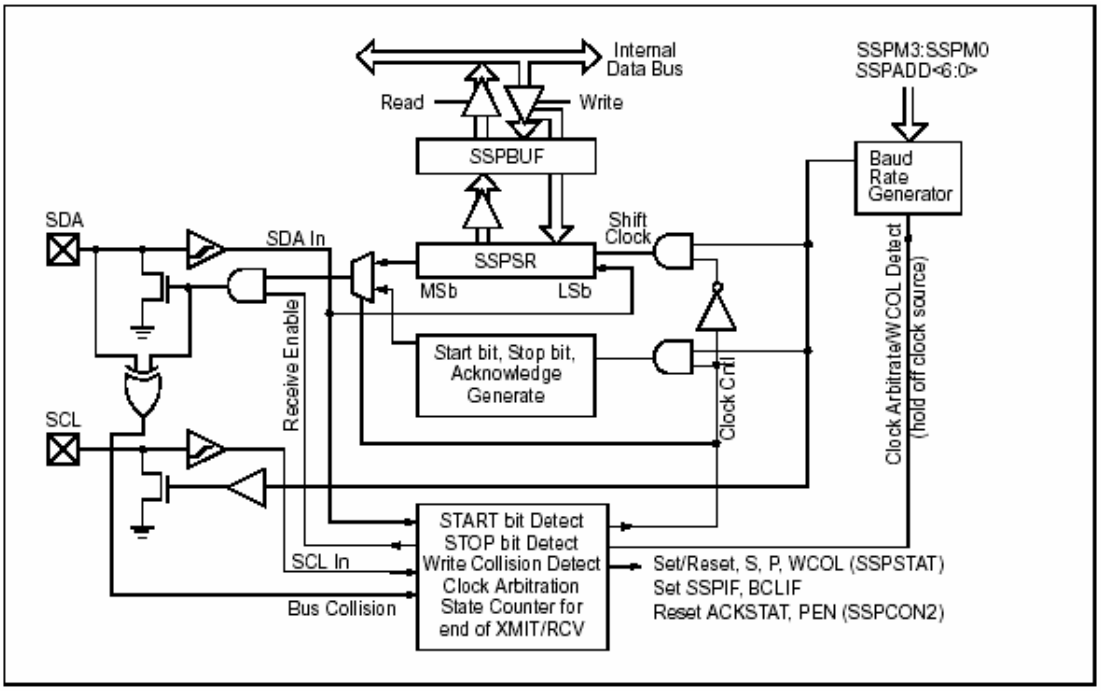
- Start κατάσταση
- Stop κατάσταση
- Όταν μεταδίδεται ή λαμβάνεται κάποιο byte
- Μετάδοση ACK bit
- Repeated Start κατάσταση

Έχουμε ήδη πει ότι το master ολοκληρωμένο είναι που παράγει τους παλμούς και τις Start και Stop καταστάσεις.

Όταν ο Master λειτουργεί ως μεταδότης, τα δεδομένα μεταφέρονται σειριακά στον slave μέσω της SDA όσο η SCL δίνει τους σειριακούς παλμούς με το ρολόι. Το πρώτο byte που μεταδίδεται περιέχει την διεύθυνση(7-bits) του slave του ολοκληρωμένου που λαμβάνει και το bit Ανάγνωσης/Εγγραφής (R/W'). Σε αυτή την περίπτωση το R/W' bit παίρνει την λογική τιμή '0'. Τα δεδομένα μεταδίδονται κάθε φορά σειριακά σε πακέτα των 8- bits. Μετά από κάθε byte που έχει μεταδοθεί ένα ACK bit λαμβάνεται. Οι Start και Stop καταστάσεις δείχνουν την αρχή και το τέλος της σειριακής μεταφοράς.

Όταν ο Master τώρα λειτουργεί ως δέκτης, το πρώτο byte που μεταδίδεται περιέχει την διεύθυνση(7-bits) του ολοκληρωμένου που θα μεταδώσει και το R/W' bit. Σε αυτή την περίπτωση το R/W' bit παίρνει την λογική τιμή '1' και έτσι φαίνεται ότι θα λάβει δεδομένα. Τα δεδομένα λαμβάνονται σειριακά μέσω της SDA όσο η SCL δίνει τους σειριακούς παλμούς με το ρολόι. Μετά από κάθε byte που έχει ληφθεί ένα ACK bit μεταδίδεται. Οι Start και Stop καταστάσεις δείχνουν την αρχή και το τέλος της σειριακής μετάδοσης.

Η γεννήτρια ρυθμού μετάδοσης των δεδομένων(BRG) χρησιμοποιείται για να θέσει την SCL σε συχνότητα 100KHz, 400KHz, 3.4MHz.



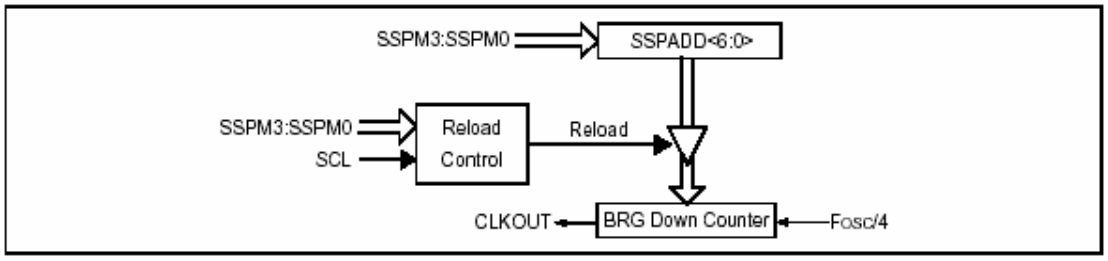
Σχήμα 2.15: I^2C Μπλοκ διάγραμμα σε λειτουργία Master

Μια τυπική ακολουθία μετάδοσης περιγράφεται παρακάτω:

1. Ο χρήστης παράγει Start κατάσταση θέτοντας το SEN bit του SSPCON2.(SSPCON2<0>).
2. Η σημαία SSPIF τίθεται και η μονάδα MSSP περιμένει τον απαραίτητο χρόνο για να αρχίσει νέα λειτουργία.
3. Ο χρήστης «φορτώνει» τον SSPBUF με την slave διεύθυνση στην οποία θέλει να μεταδώσει τα δεδομένα.
4. Η διεύθυνση μεταδίδεται μέσω του SDA pin μέχρι όλα τα bits να μεταφερθούν.
5. Η μονάδα MSSP δέχεται το ACK bit από τον slave και γράφει την τιμή του στον SSPCON2(SSPSON<6>).
6. Η μονάδα MSSP παράγει διακοπή στο τέλος του ένατου παλμού θέτοντας το SSPIF bit(PIR1<3>).
7. Ο χρήστης «φορτώνει» τον SSPBUF με τα οκτώ bits δεδομένων.
8. Τα δεδομένα μετακινούνται μέσω του SDA pin μέχρι να μεταδοθούν όλα.
9. Η μονάδα MSSP δέχεται το ACK bit από τον slave και γράφει την τιμή του στον SSPCON2(SSPSON<6>).
10. Η μονάδα MSSP παράγει διακοπή στο τέλος του ένατου παλμού θέτοντας το SSPIF bit(PIR1<3>).
11. Ο χρήστης παράγει Stop κατάσταση θέτοντας το PEN bit του SSPCON2.(SSPCON2<2>).
12. Η διακοπή παράγεται μόλις η Stop κατάσταση ολοκληρωθεί.

2.6.1 BAUD RATE GENERATOR (BRG)

Όταν το I^2C βρίσκεται σε master λειτουργία, η τιμή της γεννήτριας (BRG) τοποθετείται στα 7 χαμηλότερα bits στου καταχωρητή SSPADD. Με το που συμβεί μια εγγραφή στον SSPBUF, η γεννήτρια αυτόματα αρχίζει να μετράει αντίστροφα, καταλήγοντας δηλαδή στο μηδέν και σταματώντας εκεί που θα συμβεί μια δεύτερη μετάδοση. Η BRG μέτρηση σε κάθε κύκλο εντολής μειώνεται κατά δύο. Στην master λειτουργία ξαναρχίζει την αντίστροφη μέτρηση αυτόματα.



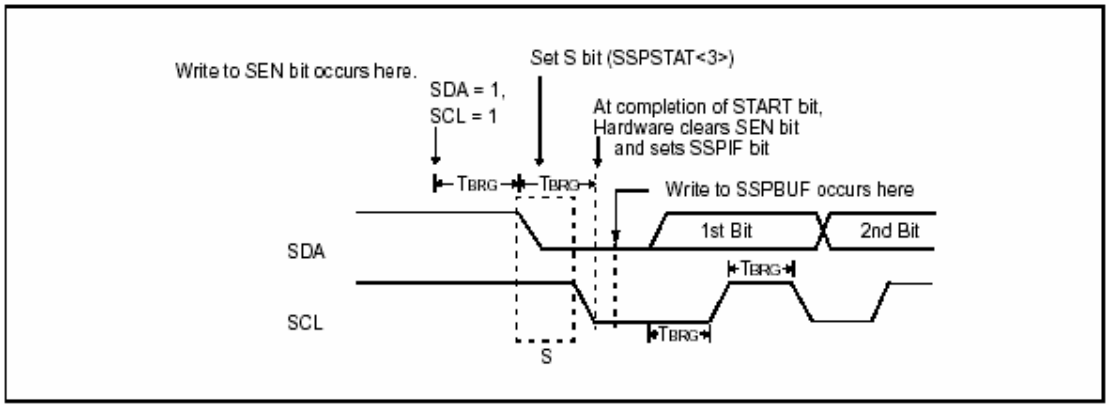
Σχήμα 2.16: Μπλοκ διάγραμμα Baud Rate Generator

Ανάλογα με το Baud Rate που επιθυμούμε ο καταχωρητής SSPADD παίρνει την τιμή του σύμφωνα με τον τύπο:SSPADD = (FOSC / (4 * BAUD))-1

2.6.2 Λειτουργία Κατάστασης Έναρξης I^2C Master

Για να αρχικοποιήσουμε μια Start κατάσταση πρέπει να ενεργοποιήσουμε το SEN bit(SSPCON2<0>). Αν τα SDA, SCL pin είναι ήδη σε υψηλή στάθμη, η BRG επανατροφοδοτείται από το περιεχόμενο του SSPADD<6:0> και αρχίζει να μετράει. Αν τα SDA, SCL pin είναι ήδη σε υψηλή στάθμη όταν η BRG τελειώνει την μέτρηση (T_{BRG}), το SDA pin οδηγείται σε χαμηλή στάθμη και όσο η SCL είναι σε υψηλή στάθμη, προκαλείται η Start κατάσταση, θέτοντας παράλληλα το S bit(SSPSTAT<3>) ίσο με 1. Μετά από αυτό η BRG επανατροφοδοτείται από το περιεχόμενο του SSPADD<6:0> και αρχίζει να μετράει ξανά. Όταν η BRG μηδενιστεί το SEN bit(SSPSTAT<3>) θα μηδενιστεί αυτόματα από το hardware και η BRG παραμερίζεται αφήνοντας την SDA γραμμή να παραμένει σε χαμηλή στάθμη.

Αξίζει να σημειώσουμε ότι αν στην αρχή μιας Start κατάστασης τα SDA, SCL pin είναι ήδη σε χαμηλή στάθμη ή κατά την διάρκεια της Start κατάστασης η SCL γραμμή δειγματοληπτείται σε χαμηλή στάθμη πριν η SDA γραμμή οδηγηθεί σε χαμηλή στάθμη, παρατηρείται μια σύγκρουση στο καλώδιο και το Bus Collision Interrupt Flag(BCLIF) ενεργοποιείται.



Σχήμα 2.17:Διάγραμμα Πρώτου Start bit

2.6.2.1 Σημαία Κατάστασης WCOL

Αν ο χρήστης γράφει στον SSPBUF όταν μια Start ακολουθία είναι σε εξέλιξη, το WCOL bit(SSPCON<7>) τίθεται και τα περιεχόμενα του απομονωτή παραμένουν τα ίδια.

2.6.3 Λειτουργία Κατάστασης Repeated Start του Master στο I^2C

Μια Repeated Start κατάσταση συμβαίνει όταν το bit RSEN (SSPCON2<1>) τίθεται ίσο με ένα και το SCL δειγματοληπτείται σε χαμηλή στάθμη. Τότε η γεννήτρια ρυθμού μετάδοσης των δεδομένων(BRG) φορτώνεται με το περιεχόμενο του καταχωρητή SSPADD<5:0> και ξεκινά να μετρά αντίστροφα. Το SDA pin απελευθερώνεται(πάει σε υψηλή στάθμη) για χρόνο ίσο με T_{BRG}. Όταν η γεννήτρια ρυθμού μετάδοσης των δεδομένων (BRG) μηδενιστεί αν το SDA pin δειγματοληπτηθεί σε υψηλή στάθμη, τότε το SCL pin θα πάει και αυτό σε υψηλή στάθμη. Όταν το SCL pin δειγματοληπτηθεί σε υψηλή στάθμη, η BRG ξαναφορτώνεται με τα περιεχόμενα του SSPADD<6:0> και ξεκινά να μετρά. Τα SDA, SCL pins πρέπει να δειγματοληπτηθούν σε υψηλή στάθμη για το χρόνο μιας περιόδου της BRG. Μετά από αυτό το bit RSEN (SSPCON2<1>) θα μηδενιστεί αυτόματα και η γεννήτρια ρυθμού μετάδοσης δεν θα ξαναφορτωθεί, αφήνοντας την SDA γραμμή σε χαμηλή στάθμη. Μόλις μια κατάσταση Έναρξης ανιχνευτεί , το bit S(SSPSTAT<3>) θα τεθεί και η σημαία SSPIF δεν θα τεθεί μέχρι που η BRG μηδενιστεί. Αμέσως μετά τον μηδενισμό της BRG η SSPIF θα τεθεί και ο χρήστης μπορεί να γράψει στον SSPBUF.

2.6.3.1 Σημαία Κατάστασης WCOL

Αν ο χρήστης γράφει στον SSPBUF όταν μια Repeated Start ακολουθία είναι σε εξέλιξη, το WCOL bit(SSPCON<7>) τίθεται και τα περιεχόμενα του απομονωτή παραμένουν τα ίδια(δεν συμβαίνει κανένα γράψιμο).

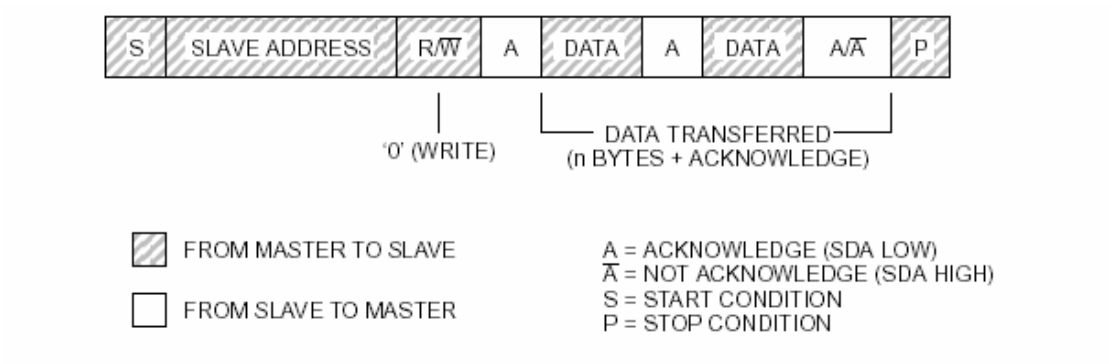
2.6.4 Λειτουργία Μετάδοσης του Master στο I^2C

Η μετάδοση ενός byte δεδομένων, μιας 7-bit διεύθυνσης ή του άλλου μισού μιας 10-bit διεύθυνσης πραγματοποιείται γράφοντας απλά την τιμή που θέλουμε στον SSPBUF. Αυτή η πράξη θα έχει ως αποτέλεσμα να γίνει η σημαία του απομονωτή, BF, ίση με 1 και ο BRG να αρχίζει την αντίστροφη μέτρηση και να ξεκινάει την επόμενη μετάδοση. Κάθε bit διεύθυνσης ή δεδομένων σπρώχνεται προς το SDA pin. Το SCL κρατείται σε χαμηλή στάθμη για μια ολόκληρη μέτρηση του BRG (T_{BRG}). Τα δεδομένα πρέπει να

είναι έγκυρα προτού το SCL μεταβεί σε υψηλή στάθμη. Όταν το SCL pin μεταβεί σε υψηλή στάθμη, κρατείται εκεί για T_{BRG}. Τα δεδομένα στο SDA pin πρέπει να παραμένουν σταθερά για διάρκεια ίση με T_{BRG} και λίγο χρόνο μετά την αρνητική ακμή του SCL. Αφού και το όγδοο bit απομακρυνθεί, η BF σημαία μηδενίζεται και ο master απελευθερώνει το SDA.

Αυτό επιτρέπει στο σωστό slave ολοκληρωμένο να απαντήσει με ένα ACK bit κατά την διάρκεια που ένατου παλμού αν η διεύθυνση ταιριάζει ή τα δεδομένα ελήφθησαν σωστά. Η κατάσταση ACK καταγράφεται στο ACKDT bit. Αν ο master λάβει μια επιβεβαίωση, το bit επιβεβαίωσης ACKSTAT (SSPCON2<6>)μηδενίζεται. Αν όχι το bit τίθεται ίσο με ένα. Μετά τον ένατο παλμό το SSPIF bit γίνεται ένα και το ρολόι του master(BRG) παραμένει σταθερό μέχρι το επόμενο byte δεδομένων να φορτωθεί στον SSPBUF, αφήνοντας το SCL χαμηλή στάθμη και το SDA αμετάβλητο.

Μετά την εγγραφή στον SSPBUF, κάθε bit της διεύθυνσης θα ολισθήσει προς τα έξω κατά την διάρκεια της αρνητική ακμή του SCL μέχρι όλα τα bits και το R/W' να συμπληρωθούν. Στο τέλος του όγδοου παλμού ο master θα απελευθερώσει το SDA, επιτρέποντας στον slave να απαντήσει με μια Επιβεβαίωση. Στο τέλος του ένατου παλμού ο master θα δειγματοληπτεί το SDA pin για να δει αν η διεύθυνση αναγνωρίστηκε από τον slave. Η κατάσταση του ACK bit φαίνεται από το ACKSTAT bit(SSPCON2<6>). Ακολουθώντας την αρνητική ακμή του ένατου παλμού μετάδοσης της διεύθυνσης, η σημαία SSPIF τίθεται, η σημαία BF μηδενίζεται και η BRG σταματάει να μετρά έως ότου μια άλλη εγγραφή πραγματοποιηθεί στον SSPBUF, κρατώντας παράλληλα το SCL σε χαμηλή στάθμη.



Σχήμα 2.18: Κατάσταση μετάδοσης του I^2C Master

2.6.4.1 Σημαία Κατάστασης BF

Στην λειτουργία μετάδοσης, το BF bit(SSPSTAT<0>) τίθεται όταν ο CPU γράφει στον SSPBUF και μηδενίζεται όταν και τα οκτώ bits έχουν απομακρυνθεί από αυτόν.

2.6.4.2 Σημαία Κατάστασης WCOL

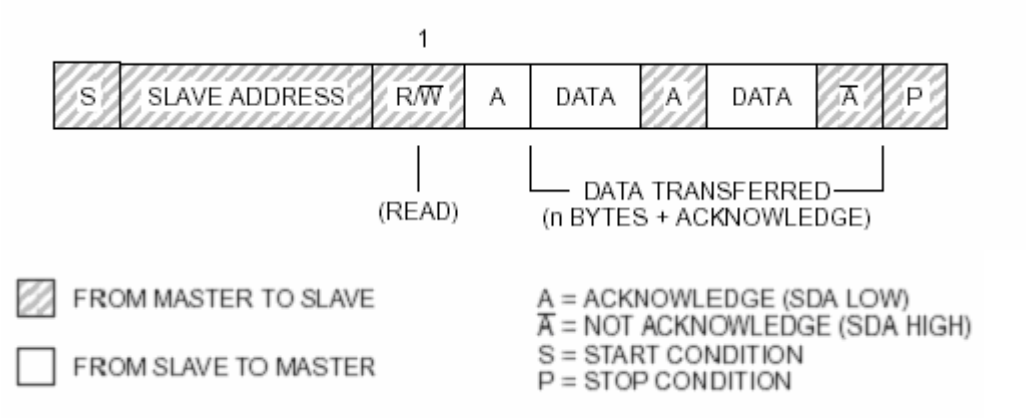
Αν ο χρήστης γράφει στον SSPBUF όταν μια μετάδοση είναι σε εξέλιξη, το WCOL bit(SSPCON<7>) τίθεται και τα περιεχόμενα του απομονωτή παραμένουν τα ίδια. Το WCOL πρέπει να μηδενίζεται από το λογισμικό.

2.6.4.3 Σημαία Κατάστασης ACKSTAT

Στην λειτουργία μετάδοσης, το ACKSTAT bit(SSPCON2<6>) μηδενίζεται όταν ο slave έχει στείλει μια Επιβεβαίωση και τίθεται όταν ο slave δεν έχει στείλει μια Επιβεβαίωση. Ο slave στέλνει Επιβεβαίωση όταν έχει αναγνωρίσει την διεύθυνση του ή έχει λάβει τα δεδομένα του σωστά.

2.6.5 Λειτουργία Λήψης του Master στο I^2C

Η λειτουργία λήψης του Master ενεργοποιείται προγραμματίζοντας το bit λήψης, RCEN (SSPCON2<3>). Η BRG ξεκινάει την αντίστροφη μέτρηση και κάθε φορά που μηδενίζεται, η κατάσταση του SCL pin αλλάζει (υψηλή/χαμηλή και το αντίστροφο) και τα δεδομένα ολισθαίνουν στον SSPSR. Μετά την αρνητική ακμή του όγδοου παλμού, η σημαία που ενεργοποιεί την λήψη αυτόματα μηδενίζεται, τα περιεχόμενα του SSPSR αποθηκεύονται στον SSPBUF, το BF bit τίθεται, η σημαία SSPIF τίθεται και η γεννήτρια παραγωγής ρυθμού μετάδοσης (BRG) σταματά να μετρά κρατώντας έτσι την SCL σε χαμηλή στάθμη. Το MSSP είναι τώρα σε αδρανή κατάσταση, περιμένοντας την επόμενη εντολή. Όταν ο απομονωτής διαβαστεί από την CPU, η σημαία BF μηδενίζεται αυτόματα. Ο χρήστης τότε μπορεί να στείλει μια Επιβεβαίωση στο τέλος της λήψης των δεδομένων, θέτοντας το ACKEN bit(SSPCON2<4>) ίσο με ένα.



Σχήμα 2.19: Κατάσταση Λήψης του I^2C Master

2.6.5.1 Σημαία Κατάστασης BF

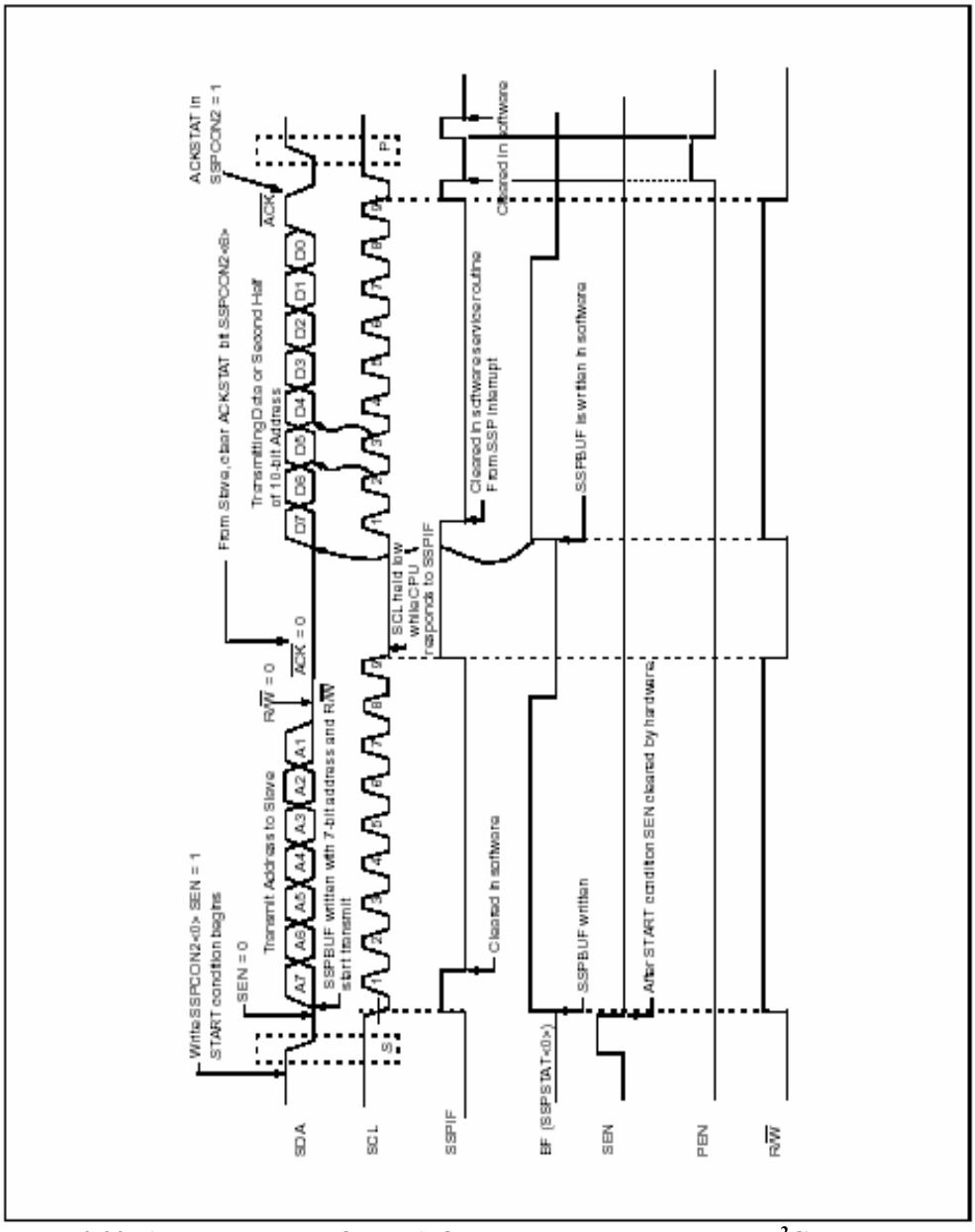
Στην λειτουργία λήψης, το BF bit(SSPSTAT<0>) τίθεται όταν μια διεύθυνση ή ένα byte δεδομένων γράφεται στον SSPBUF από τον SSPSR και μηδενίζεται όταν και ο SSPBUF έχει διαβαστεί.

2.6.5.2 Σημαία Κατάστασης SSPOV

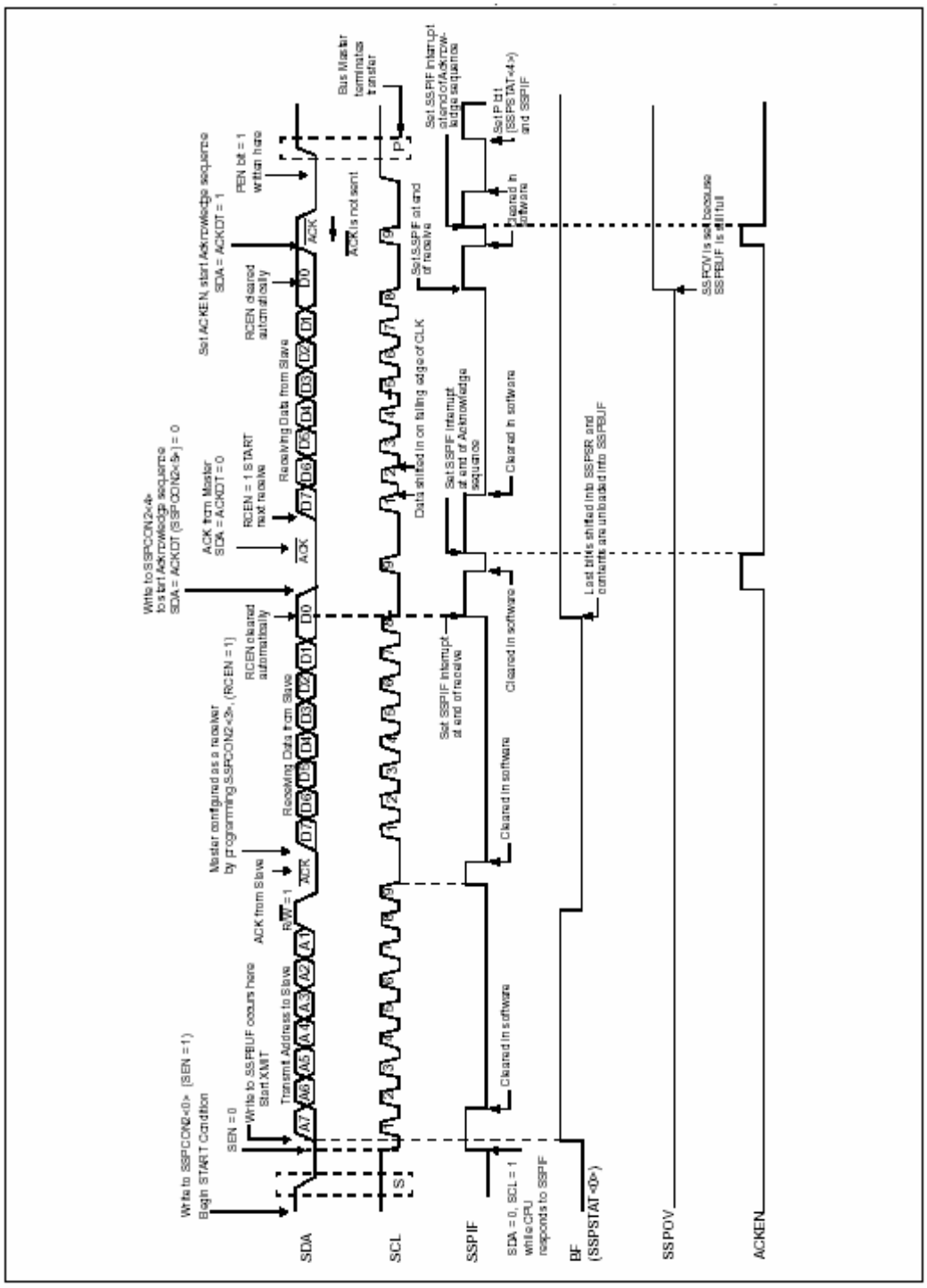
Στην λειτουργία λήψης, το SSPOV bit(SSPSTAT<6>) τίθεται όταν και τα 8 bits ληφθούν από τον SSPSR και BF bit έχει ήδη τεθεί από προηγούμενη λήψη.

2.6.5.3 Σημαία Κατάστασης WCOL

Αν ο χρήστης γράψει στον SSPBUF όταν αυτός λαμβάνει δεδομένα που είναι ακόμα σε εξέλιξη, το WCOL bit τίθεται και τα περιεχόμενα του απομονωτή μένουν αμετάβλητα(δεν συμβαίνει κανένα γράψιμο).



Σχήμα 2.20: Λειτουργία Μετάδοσης Δεδομένων από τον Master στο I^2C

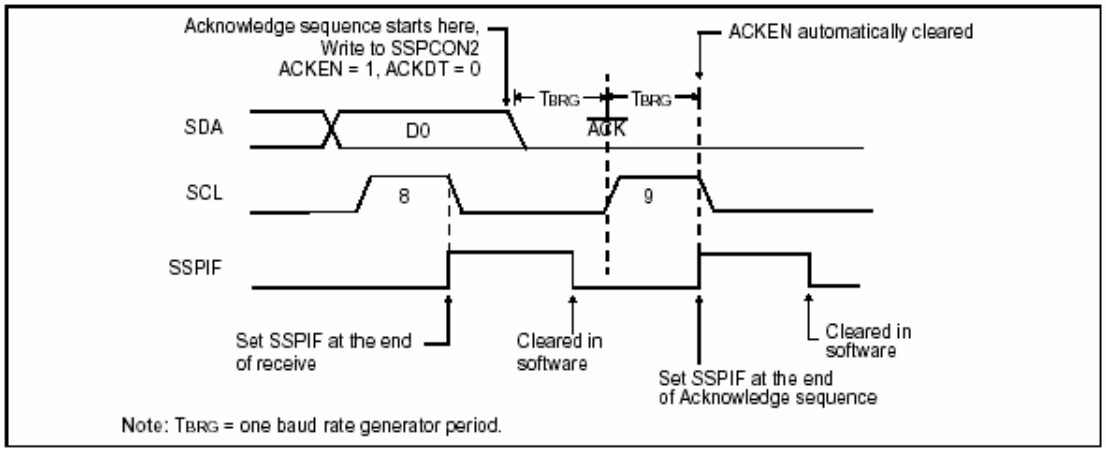


Σχήμα 2.21: Λειτουργία Λήψης Δεδομένων από τον Master στο I^2C

2.6.6 Χρονισμός Ακολουθίας Επιβεβαίωσης

Μια ακολουθία επιβεβαίωση ενεργοποιείται θέτοντας το ACKEN bit (SSPCON2<4>). Όταν το bit αυτό τεθεί, το SCL πάει σε χαμηλή στάθμη και τα περιεχόμενα των δεδομένων της Επιβεβαίωσης φαίνονται από το SDA pin. Αν ο χρήστης επιθυμεί να παράγει μια Επιβεβαίωση, τότε το ACKDT bit πρέπει να μηδενιστεί. Αν όχι, ο χρήστης πρέπει να θέσει το ACKDT bit πριν αρχίσει η ακολουθία της επιβεβαίωσης. Μετά η γεννήτρια παραγωγής ρυθμού

μετάδοσης(BRG) αρχίζει να μετράει αντίστροφα για T_{BRG} και το SCL pin πάει σε υψηλή στάθμη. Όταν το SCL pin δειγματοληπτείται, η BRG μετράει για T_{BRG}. Το SCL pin μετά κρατείται σε χαμηλή στάθμη. Έτσι μετά από αυτό το ACKEN bit θα μηδενιστεί αυτόματα, η γεννήτρια παραγωγής ρυθμού μετάδοσης θα σταματήσει και η μονάδα MSSP θα μείνει αδρανής.



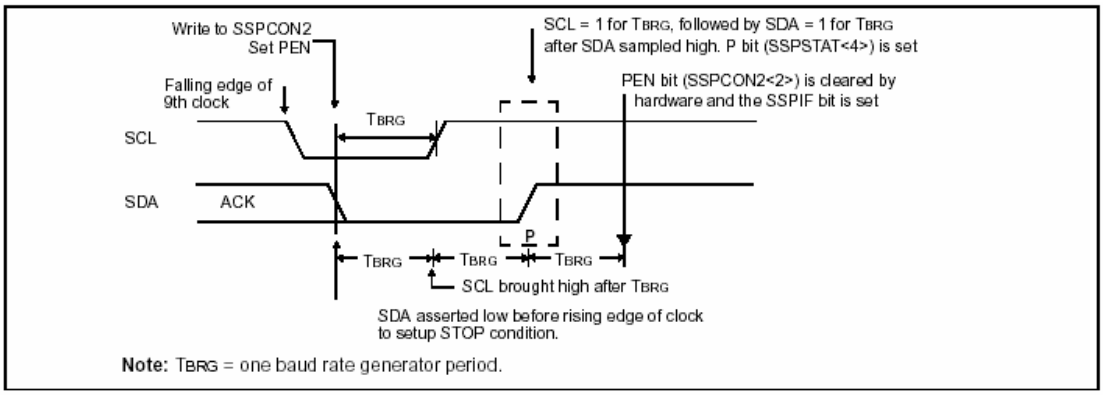
Σχήμα 2.22: Ακολουθία Επιβεβαίωσης στο I^2C

2.6.6.1 Σημαία Κατάστασης WCOL

Αν ο χρήστης γράφει στον SSPBUF όταν μια Επιβεβαίωση είναι σε εξέλιξη, τότε το WCOL bit τίθεται και τα περιεχόμενα του απομονωτή μένουν αμετάβλητα(δεν συμβαίνει κανένα γράψιμο).

2.6.7 Χρονισμός Κατάστασης Stop

Όπως έχουμε αναφέρει ένα stop bit βεβαιώνεται από το SDA pin στο τέλος μιας λήψης ή μετάδοσης. Η stop ακολουθία ενεργοποιείται θέτοντας το PEN bit(SSPCON2<2>). Στο τέλος της λήψης ή μετάδοσης, η SCL γραμμή είναι σε χαμηλή στάθμη μετά τη αρνητική ακμή του ένατου παλμού. Όταν το PEN bit τίθεται, ο master κρατήσει την SDA γραμμή σε χαμηλή στάθμη. Όταν η SDA γραμμή δειγματοληπτηθεί θα είναι σε χαμηλή στάθμη η γεννήτρια παραγωγής ρυθμού μετάδοσης ξαναρχίζει να μετρά αντίστροφα μέχρι το μηδέν. Όταν η γεννήτρια παραγωγής ρυθμού μετάδοσης μηδενιστεί, η SCL πάει σε υψηλή στάθμη και ένα T_{BRG} χρόνο αργότερα, η SDA ξαναπηγαίνει σε υψηλή στάθμη. Έτσι το P bit (SSPSTAT<4>) τίθεται όταν το SDA δειγματοληπτηθεί σε υψηλή στάθμη όσο το SCL είναι σε χαμηλή στάθμη. Ένα T_{BRG} χρόνο αργότερα, το PEN bit μηδενίζεται και το SSPIF bit τίθεται ίσο με ένα.



Σχήμα 2.23: Stop Κατάσταση σε Λειτουργία Λήψης ή Μετάδοσης

2.6.7.1 Σημαία Κατάστασης WCOL

Αν ο χρήστης γράφει στον SSPBUF όταν μια Stop κατάσταση είναι σε εξέλιξη, τότε το WCOL bit τίθεται και τα περιεχόμενα του απομονωτή μένουν αμετάβλητα(δεν συμβαίνει κανένα γράψιμο).

2.6.8 Λειτουργία SLEEP

Κατά την διάρκεια λειτουργίας σε κατάσταση sleep, η μονάδα I^2C μπορεί να λαμβάνει διευθύνσεις ή δεδομένα. Όταν συμβεί το ταίριασμα διεύθυνσης ή ολοκληρωμένη μεταφορά ενός byte, θα παραχθεί μια διακοπή ssp η οποία θα «ξυπνήσει» τον επεξεργαστή από την κατάσταση sleep(εφόσον βέβαια είναι ενεργοποιημένη η διακοπή SSP).

2.6.9 Αποτελέσματα RESET

Όταν γίνει reset απενεργοποιείται η μονάδα SSP και τερματίζεται η τρέχουσα μεταφορά δεδομένων.

2.7 Λειτουργία Master σε Multi-Master περιβάλλον

Σε λειτουργία Multi-Master, η παραγωγή διακοπών μόλις ανιχνευτεί μια Start ή Stop κατάσταση καθορίζει πότε το καλώδιο είναι αδρανές. Τα bits αρχής (S) και τέλους (P) μηδενίζονται κατά το reset ή όταν η μονάδα απενεργοποιείται. Ο έλεγχος του διαύλου I^2C μπορεί να ληφθεί όταν το bit (P) (SSPSTAT<4>) γίνει ένα, ή όταν ο διάδρομος είναι ανενεργός οπότε και τα δυο bits S, P είναι μηδέν.

Επίσης η SDA γραμμή πρέπει να ελέγχεται(διαιτησία διαδρόμου) έτσι ώστε να ξέρουμε αν το σήμα εξόδου είναι το αναμενόμενο. Το αποτέλεσμα γίνεται αντιληπτό από το BCLIF bit.

Τα στάδια όπου μια διαιτησία διαδρόμου(arbitration) μπορεί να είναι:

- Μεταφορά διεύθυνσης

- Μεταφορά δεδομένων
- Κατάσταση Stop
- Κατάσταση Repeated Start
- Κατάσταση Επιβεβαίωσης

2.7.1 Επικοινωνία Multi-master, Διαιτησία Διαδρόμου, Συγκρούσεις Στο Καλώδιο

Η λειτουργία Multi-Master υποστηρίζεται από την διαιτησία διαδρόμου. Η διαιτησία διαδρόμου διαδραματίζεται όταν ο ένας master στέλνει '1' στην SDA και ο άλλος master στέλνει '0'. Όταν το SCL pin παραμένει σε υψηλή στάθμη τα δεδομένα πρέπει να είναι σταθερά.

Κάθε φορά που ο master προσπαθεί να θέσει μια γραμμή σε υψηλή στάθμη θα πρέπει στην συνέχεια να ελέγχει αν αυτή όντως βρίσκεται σε υψηλή στάθμη. Αν αυτή δεν βρίσκεται σε υψηλή στάθμη τότε κάποιος άλλος οδηγεί την γραμμή σε χαμηλή στάθμη οπότε ο master θα πρέπει να αφήσει τον δίαυλο και να περιμένει κάποια κατάσταση Stop για να τον πάρει πάλι στην κατοχή του. Το γεγονός ότι αν προσπαθούν δυο διαφορετικές συσκευές να γράψουν διαφορετικά δεδομένα στο bus, δεν εγκυμονεί κανένα κίνδυνο για τις συσκευές και το ότι πάντα αυτός που εκπέμπει την σε χαμηλή στάθμη επικρατεί στον δίαυλο διασφαλίζει ότι πάντα θα υπάρχει μόνο ένας master στο bus και ότι η μετάδοση του δεν θα διακόπτεται.

2.7.2 Σύγκρουση στο Bus κατά την διάρκεια της Start Κατάστασης

Κατά την διάρκεια της Start κατάστασης, μια σύγκρουση συμβαίνει όταν:

- Οι SDA ή SCL δειγματοληπτούνται σε χαμηλή στάθμη στην αρχή της Start κατάστασης
- SCL δειγματοληπτείται σε χαμηλή στάθμη πριν η SDA βεβαιωθεί ότι είναι σε χαμηλή στάθμη

Κατά την διάρκεια της Start κατάστασης, και οι δύο γραμμές SDA, SCL παρακολουθούνται.

Αν το SDA pin ή το SCL pin είναι ήδη σε χαμηλή στάθμη, τότε συμβαίνουν τα εξής:

- η Start κατάσταση διακόπτεται
- η BCLIF τίθεται και
- η MSSP μονάδα πηγαίνει σε αδρανή κατάσταση

2.7.3 Σύγκρουση στο Bus κατά την διάρκεια μιας Repeated Start Κατάστασης

Κατά την διάρκεια της Repeated Start κατάστασης, μια σύγκρουση συμβαίνει αν:

- Η SDA δειγματοληπτείται σε χαμηλή στάθμη όταν η SCL πηγαίνει από χαμηλή σε υψηλή στάθμη
- Η SCL πηγαίνει σε χαμηλό επίπεδο πριν βεβαιώσουμε ότι η SDA είναι σε χαμηλό επίπεδο, οπότε ένας άλλος master προσπαθεί να μεταδώσει δεδομένα

2.7.4 Σύγκρουση στο Bus κατά την διάρκεια μιας Stop Κατάστασης

Κατά την διάρκεια της Stop κατάστασης, μια σύγκρουση συμβαίνει αν:

- Αφού το SDA pin βρίσκεται σε υψηλή στάθμη, τότε το SCL δειγματοληπτείται σε χαμηλή στάθμη μετά από χρόνο μια περιόδου(T_{BRG}) της γεννήτριας παραγωγής ρυθμού μετάδοσης των δεδομένων.
- Αφού το SCL pin βρίσκεται σε υψηλή στάθμη, το SCL δειγματοληπτείται σε χαμηλή στάθμη πριν το SDA πάει σε υψηλή στάθμη.

2.8 Λειτουργία I^2C Slave

Προτού επιλεγεί οποιαδήποτε λειτουργία στο I^2C, τα pin SDA, SCL πρέπει να προγραμματιστούν ως είσοδοι θέτοντας 1 στα αντίστοιχα TRIS bits(TRISC<4:3>). Η μονάδα SSP θα υπερκαλύψει την κατάσταση εισόδου με τα δεδομένα εξόδου όταν αυτό απαιτείται(slave που στέλνει δεδομένα).

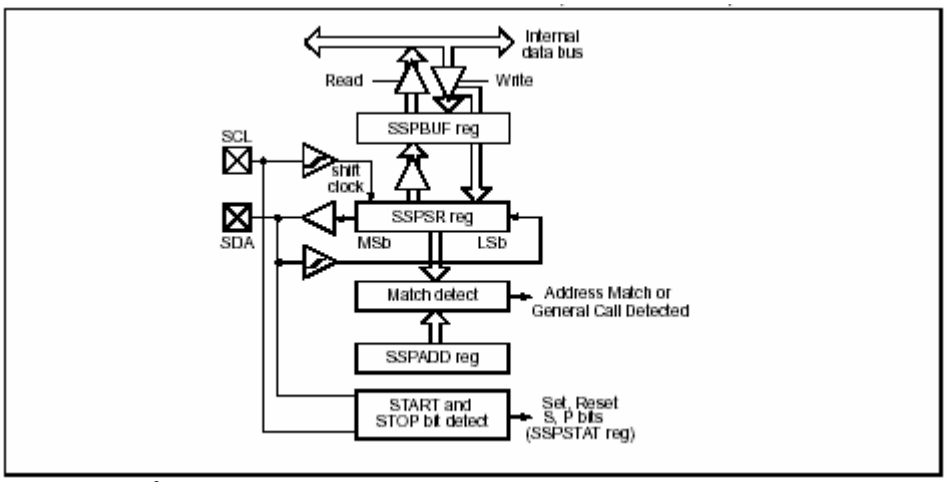
Όταν μια διεύθυνση ταιριάζει ή τα δεδομένα μετά το ταίριασμα της διεύθυνσης ληφθούν, το hardware αυτόματα παράγει ένα παλμό αναγνώρισης (ACK) και φορτώνει τον καταχωρητή SSPBUF με την τιμή που έλαβε και που την στιγμή αυτή βρισκόταν στον SSPSR.

Κάτω από ορισμένες συνθήκες η μονάδα SSP δεν θα δώσει αυτόν τον παλμό αναγνώρισης (ACK) . Οι συνθήκες αυτές μπορούν να ισχύουν είτε και οι δύο είτε η μια με το ίδιο αποτέλεσμα και είναι οι εξής:

- Το bit του απομονωτή BF (SSPSTAT<0>) έγινε 1 προτού ολοκληρωθεί το μήνυμα
- Το bit υπερχείλισης SSPOV (SSPCON<6>) έγινε 1 προτού ολοκληρωθεί το μήνυμα

Σε αυτή την περίπτωση η τιμή του καταχωρητή SSPSR δεν φορτώνεται στον SSPBUF αλλά τα bits SSPIF(PIR1<3>) και SSPOV γίνονται ένα(1).

Η είσοδος του ρολογιού πρέπει να έχει έναν ελάχιστο χρόνο high και low για την σωστή λειτουργία. Οι χρόνοι όπως ορίζονται στις προδιαγραφές I^2C, μπορούν να βρεθούν στις ηλεκτρικές προδιαγραφές του μικροελεγκτή στο φυλλάδιο δεδομένων στις παραμέτρους 100 και 101.



Σχήμα 2.24: I^2C Μπλοκ διάγραμμα σε λειτουργία Slave

2.8.1 Διευθυνσιοδότηση

Όταν η μονάδα MSSP ενεργοποιηθεί, περιμένει να συμβεί μια κατάσταση έναρξης. Μόλις αυτό συμβεί, τα 8 bits ολισθαίνουν μέσα στον καταχωρητή SSPSR. Όλα τα εισερχόμενα bits δειγματοληπτούνται κατά την θετική ακμή της γραμμής σήματος του ρολογιού(SCL). Η τιμή του καταχωρητή SSPSR <7:1> συγκρίνεται με την τιμή του καταχωρητή SSPADD, κατά την αρνητική ακμή του όγδοου παλμού ρολογιού. Αν οι δύο διευθύνσεις ταιριάζουν, τα bits BF και SSPOV μηδενίζονται και συμβαίνουν τα ακόλουθα:

1. Το περιεχόμενο του καταχωρητή SSPSR φορτώνεται στον SSPBUF κατά την αρνητική τιμή του όγδοου παλμού του ρολογιού.
2. Το bit BF (γεμάτος απομονωτής) γίνεται 1 στην αρνητική τιμή του όγδοου παλμού του ρολογιού.
3. Δημιουργείται ένας παλμός ACK'
4. Το bit διακοπής SSPIF γίνεται 1(και προκαλείται αν είναι ενεργοποιημένη η αντίστοιχη διακοπή) κατά την αρνητική ακμή του ένατου παλμού SCL.

Στην περίπτωση διευθυνσιοδότησης των 10-bits, ο slave χρειάζεται να λάβει 2 bytes διεύθυνσης. Τα πέντε πιο σημαντικά bits (MSBs) του πρώτου byte διεύθυνσης δείχνουν αν πρόκειται για διεύθυνση των10-bits. Το bit R/W' (SSPSTAT<2>) καθορίζει αν πρόκειται να γίνει εγγραφή οπότε ο σκλάβος θα λάβει το δεύτερο byte διεύθυνσης. Για μια διεύθυνση των 10bits το πρώτο byte θα ισούται με '1110 A9 A8 0' όπου A9, A8 τα δύο πιο σημαντικά bits(MSBs) της διεύθυνσης. Η ακολουθία των γεγονότων για μια διεύθυνση των 10bits, έχει ως εξής(τα βήματα 7-9 ισχύουν και στην περίπτωση του slave που μεταδίδει δεδομένα):

1. Λήψη του πρώτου (high) byte της διεύθυνσης(τα bits SSPIF, BF και UA (SSPSTAT<1>) γίνονται 1).
2. Ενημέρωση του καταχωρητή SSPADD με το δεύτερο (low) byte της διεύθυνσης (μηδενίζεται το bit UA και απελευθερώνεται η γραμμή SCL).

3. Ανάγνωση του SSPBUF(μηδενίζεται το bit BF) και μηδενισμός του bit της σημαίας SSPIF.
4. Λήψη του δεύτερου(low) byte της διεύθυνσης (τα bits SSPIF, BF και UA γίνονται 1).
5. Ενημέρωση του SSPADD με το high byte της διεύθυνσης (μηδενίζεται το bit UA και απελευθερώνεται η γραμμή SCL).
6. Ανάγνωση του SSPBUF (μηδενίζεται το bit BF) και μηδενισμός του SSPIF.
7. Λήψη Επαναλαμβανόμενης κατάστασης Έναρξης.
8. Λήψη του πρώτου (high) byte της διεύθυνσης (τα bits SSPIF, BF γίνονται 1).
9. Ανάγνωση του SSPBUF (μηδενίζεται το bit BF) και μηδενισμός του SSPIF.

2.8.2 Λήψη Δεδομένων

Όταν το bit R/W' του byte διεύθυνσης είναι μηδέν και έχουμε ταίριασμα διεύθυνσης, το bit R/W' του SSPSTAT γίνεται μηδέν. Η διεύθυνση που λήφθηκε φορτώνεται στον SSPBUF και η γραμμή SDA κρατείται σε χαμηλή στάθμη.

Στην περίπτωση υπερχείλισης του byte διεύθυνσης, παράγεται παλμός Μη-αναγνώρισης (NACK). Αυτή η περίπτωση ανιχνεύεται όταν είτε το bit BF (SSPSTAT<0>) είναι 1, είτε το SSPOV(SSPCON<6>) είναι 1. Έτσι, όταν λαμβάνεται ένα byte με αυτές τις συνθήκες και προσπαθεί να μεταφερθεί από τον SSPSR στον SSPBUF, δεν δίνεται παλμός αναγνώρισης.

Για κάθε μεταφορά ενός byte, δημιουργείται μια διακοπή SSP. Το bit σημαίας SSPIF πρέπει να καθαρίζεται από το software. Ο καταχωρητής SSPSTAT χρησιμοποιείται για να καθοριστεί η κατάσταση του byte που λαμβάνεται.

Αν το SEN (SSPCON<0>) bit ενεργοποιηθεί, τα RC3/SCK/SCL θα κρατηθούν σε χαμηλή στάθμη(clock stretch) ακολουθώντας κάθε μεταφορά δεδομένων. Το ρολόι πρέπει τότε να απελευθερωθεί θέτοντας το bit CKP (SSPCON<4>) ίσο με 1.

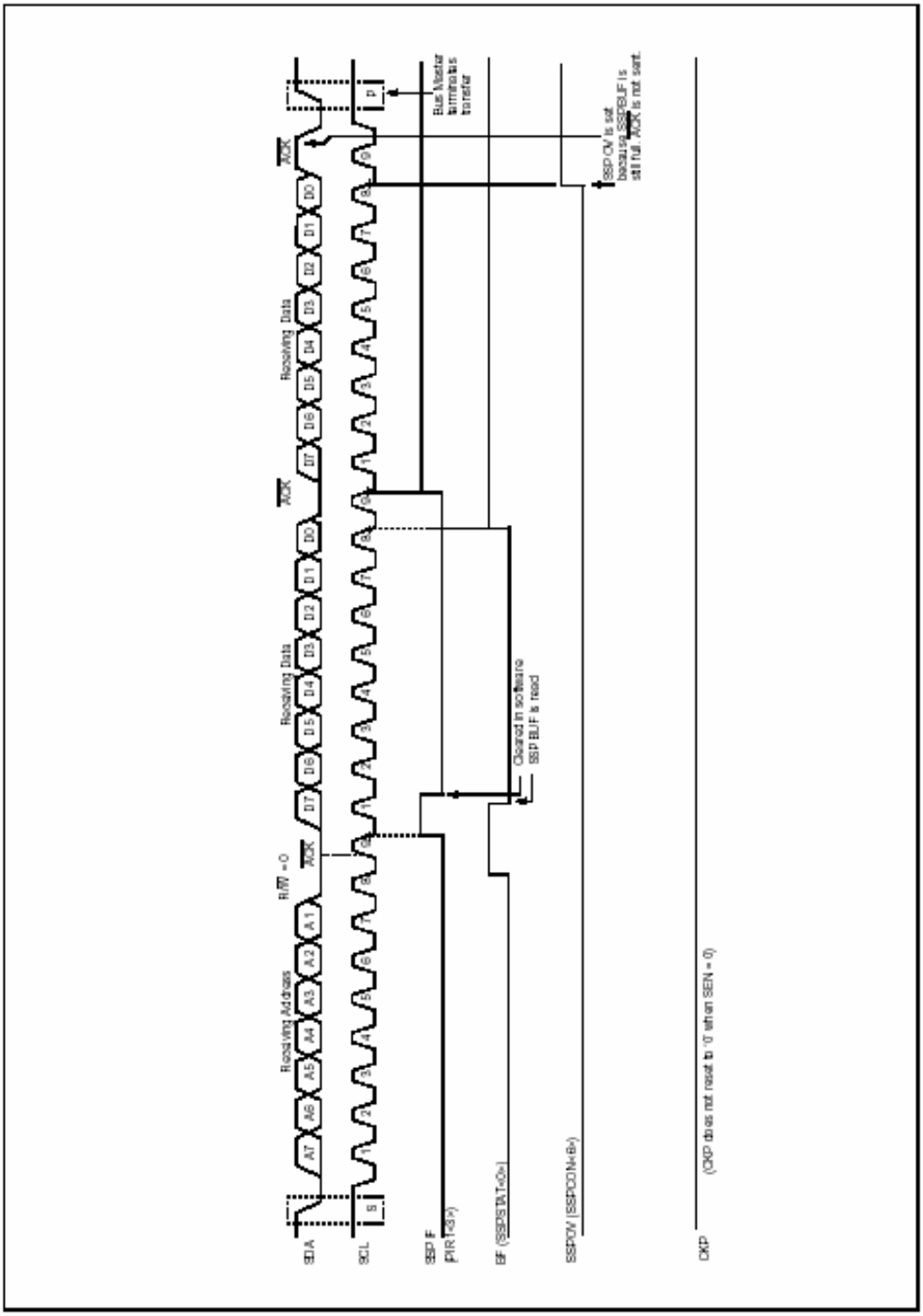
2.8.3 Αποστολή Δεδομένων

Όταν το bit R/W' του εισερχόμενου byte είναι 1 και έχουμε ταίριασμα διεύθυνσης, το bit R/W' του SSPSTAT γίνεται και αυτό ένα και η διεύθυνση που λήφθηκε στέλνεται στον SSPBUF. Στο ένατο bit στέλνεται ένας παλμός ACK' και το pin SCL κρατείται σε κατάσταση low. Τα προς μετάδοση δεδομένα πρέπει να φορτωθούν στον SSPBUF, που φορτώνει και τον SSPSR και πρέπει τότε να ενεργοποιηθεί το pin SCL κάνοντας το bit CKP ίσο με ένα. Ο αφέντης πρέπει να ελέγχει το pin SCL προτού δημιουργήσει νέο παλμό ρολογιού αφού οι συσκευές-σκλάβοι μπορούν να «παγώνουν» τον αφέντη κρατώντας το ρολόι. Τα 8 bits δεδομένων ολισθαίνονται προς τα έξω κατά την αρνητική ακμή του ρολογιού στην είσοδο του SCL. Αυτό διασφαλίζει

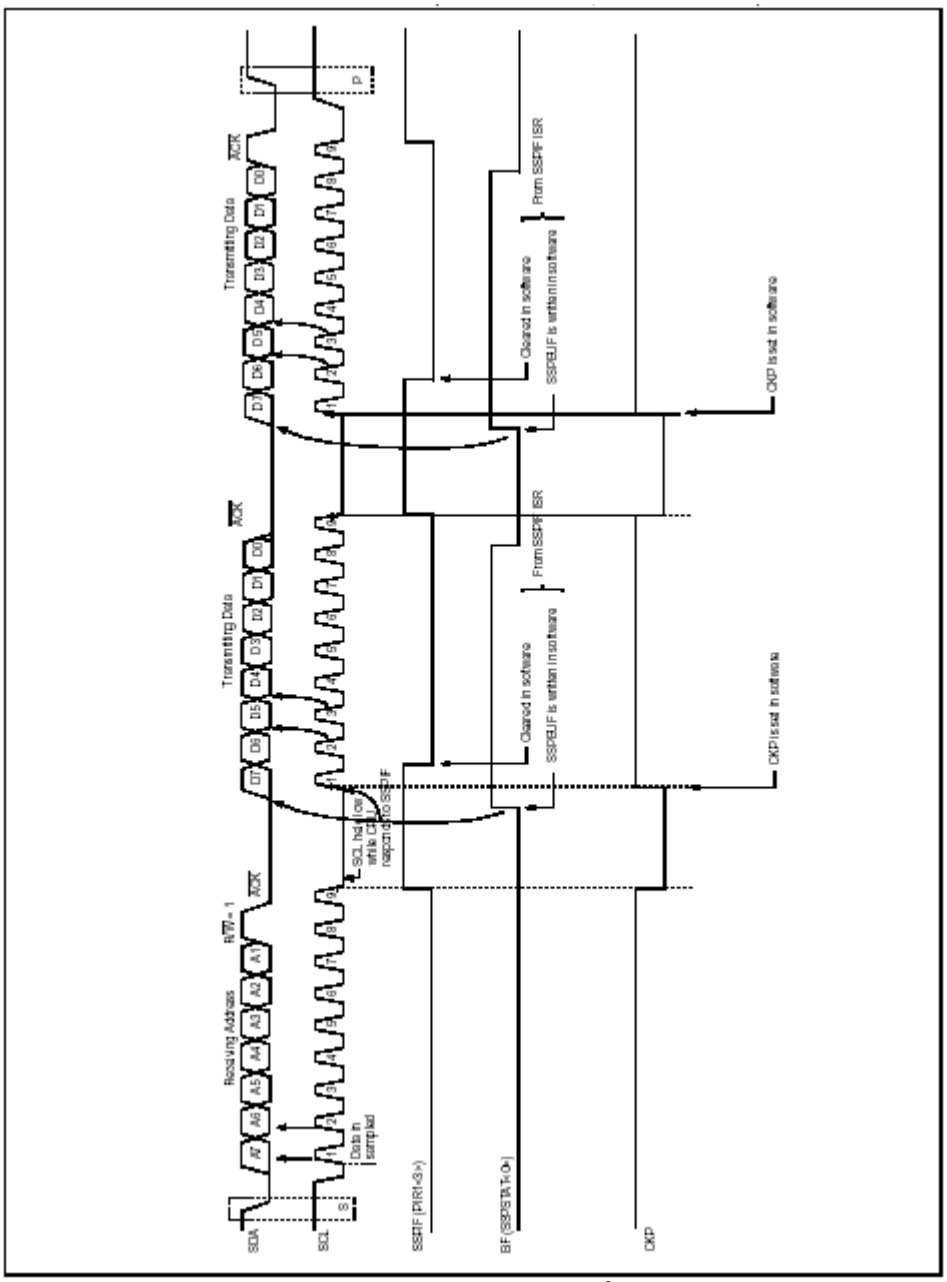
ότι το σήμα SDA είναι έγκυρο κατά την διάρκεια που ο παλμός SCL είναι σε κατάσταση high.

Για κάθε μεταφερόμενο byte δημιουργείται και μια διακοπή SSP. Η σημαία SSPIF πρέπει να καθαρίζεται από το software, ενώ ο καταχωρητής SSPSTAT χρησιμεύει για την αναγνώριση της κατάστασης μεταφοράς του byte. Η σημαία SSPIF γίνεται 1 κατά την αρνητική ακμή του ένατου παλμού του ρολογιού.

Σε έναν slave που εκπέμπει δεδομένα, ο παλμός ACK' του master-δέκτη λαμβάνεται κατά την θετική ακμή του ένατου παλμού του ρολογιού στην είσοδο SCL. Αν η γραμμή SDA βρισκόταν σε κατάσταση high(not ACK'), η μεταφορά δεδομένων έχει ολοκληρωθεί. Όταν το not ACK' ληφθεί από τον σκλάβο, το λογικό κύκλωμα του σκλάβου κάνει reset και ελέγχει τότε την έλευση του νέου bit έναρξης. Αν η γραμμή sda ήταν σε κατάσταση low(ACK'), τα προς μεταφορά δεδομένα πρέπει να φορτωθούν στον SSPBUF που με τη σειρά του φορτώνει τον SSPSR ενώ πρέπει να ενεργοποιηθεί και το pin SCL κάνοντας ένα το bit CKP.



Σχήμα 2.25: Κατάσταση Λήψης σε Λειτουργία I^2C Slave



Σχήμα 2.26: Κατάσταση Μετάδοσης σε Λειτουργία I^2C Slave

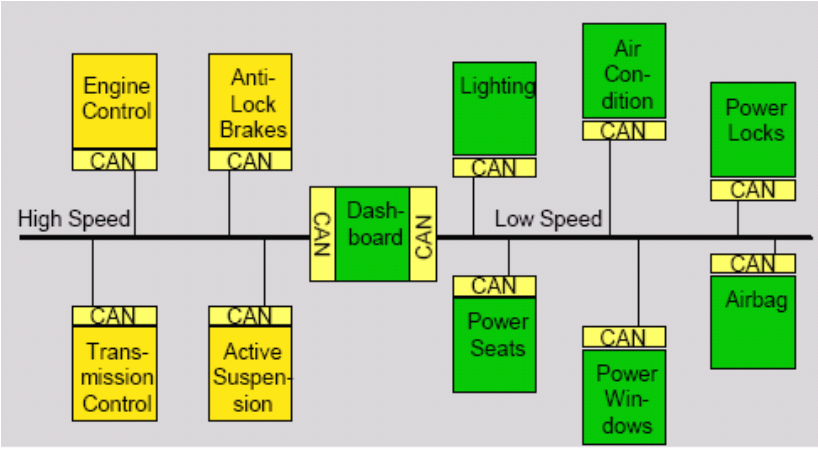
2.8.4 Επίτρεψη ρολογιού

Η επίτρεψη ρολογιού ελέγχεται από το pin SCL και μπορεί να απαγορεύει από τον αφέντη την αποστολή του επόμενου παλμού. Η μονάδα SSP κατά την λειτουργία σκλάβου I^2C κρατά το SCL pin σε κατάσταση low όταν ο επεξεργαστής χρειάζεται να ανταποκριθεί στην διακοπή SSP (το bit SSPIF είναι 1 και το CKP 0). Τότε τα δεδομένα που πρόκειται να σταλούν πρέπει να γραφούν στον SSPBUF και κατόπιν πρέπει να γίνει 1 το bit CKP ώστε να επιτραπεί στον αφέντη να στείλει τους απαιτούμενους παλμούς ρολογιού.

3. Εισαγωγή στο CAN

Το 1983 η Bosch αρχίζει να αναπτύσσει δίκτυο για το εσωτερικό των αυτοκινήτων. Τρία χρόνια αργότερα γίνεται και η επίσημη παρουσίαση του πρωτοκόλλου CAN για αυτό το σκοπό. Το 1991 έχουμε τη δεύτερη έκδοση του πρωτοκόλλου (με το extended Identifier και κάποιες άλλες μικρές αλλαγές) και ένα χρόνο αργότερα υλοποιείται για πρώτη φορά στην παραγωγή από τα αυτοκίνητα της Mercedes-Benz.

Το πρωτόκολλο CAN αναπτύχθηκε για να οργανώσει τα διαρκώς αυξανόμενα σε αριθμό και πολυπλοκότητα ηλεκτρονικά των σύγχρονων αυτοκινήτων όπως το ABS, ESP, έλεγχο των φώτων κτλ. Όπως φαίνεται και στο σχήμα παρακάτω δημιούργησε μία ιεραρχία σε σχέση με την «αναρχία» που επικρατούσε μέχρι εκείνη την περίοδο και περιόρισε σημαντικά των αριθμό των καλωδιώσεων άρα και το κόστος στα σύγχρονα αυτοκίνητα.



Σχήμα 3.1: CAN BUS

Δηλαδή πλέον έχουμε το διαχωρισμό των κόμβων σε δίκτυα υψηλών και χαμηλών ταχυτήτων και γενικότερα την δημιουργία κάποιας τάξης στη μεταξύ τους επικοινωνία.

Το CAN BUS αποτελεί ένα πρωτόκολλο δικτύου που χρησιμοποιείται στα σύγχρονα αυτοκίνητα και γενικότερα στα μέσα μεταφοράς για την επικοινωνία των διάφορων ηλεκτρονικών μεταξύ τους και κυρίως με τον κεντρικό εγκέφαλο. Είναι ένα σειριακό πρωτόκολλο επικοινωνίας το οποίο υποστηρίζει αποτελεσματικά κατανεμημένο έλεγχο σε πραγματικό χρόνο.

Τα σύγχρονα αυτοκίνητα διαθέτουν μία πληθώρα από ηλεκτρονικά συστήματα. Αυτά τα συστήματα επικοινωνούν μεταξύ τους ανταλλάσσοντας πληροφορίες και μηνύματα. Η επικοινωνία αυτή κάνει πιο εύρυθμη την λειτουργία ενός αυτοκινήτου. Προκειμένου όμως να επικοινωνήσουν όλα αυτά τα συστήματα μεταξύ τους, θα απαιτούνταν η ανά ζεύγος διασύνδεσή τους. Αυτό θα σήμαινε μεγάλο κόστος και μεγάλο πλήθος καλωδίωσης. Σε αυτό το πρόβλημα έρχεται να δώσει λύσει το πρωτόκολλο CAN. Έρχεται να αντικαταστήσει αυτή την πληθώρα καλωδίων με ένα δίαυλο, πάνω στον οποίο όλοι επικοινωνούν. Αν μάλιστα αναλογιστούμε ότι το κόστος στα ηλεκτρονικά του αυτοκινήτου, φτάνει μέχρι και το 10% της αξίας του και διαρκώς

αυξάνεται, η μείωση κόστους που έχει επιφέρει είναι τεράστια. Όλοι πλέον στις εφαρμογές που δημιουργούν για το αυτοκίνητο συμπεριλαμβάνουν και το πρωτόκολλο CAN.

Οι ταχύτητες τις οποίες και επιτυγχάνει φτάνουν και έως 1Mbps ταχύτητες ικανές για την ανταλλαγή δεδομένων ανάμεσα στους κόμβους, καθώς είναι μικρός ο αριθμός των δεδομένων που ανταλλάσσονται και καλύπτει τις απαιτήσεις των συγχρόνων εφαρμογών και υλοποιήσεων.

Τα κύρια χαρακτηριστικά του πρωτοκόλλου CAN είναι:

- Προτεραιότητα στα μηνύματα
- Εξασφαλισμένοι χρόνοι καθυστέρησης
- Ευελιξία στην παραμετροποίηση
- Δυνατότητα πολλαπλών παραληπτών μηνυμάτων με δυνατότητα συγχρονισμού
- Multimaster
- Ανίχνευση λαθών
- Αυτόματη επαναποστολή κατεστραμμένων μηνυμάτων μόλις ο διάδρομος ελευθερωθεί
- Διάκριση μεταξύ προσωρινών λαθών και μόνιμων αποτυχιών κόμβων και αυτόματη απομάκρυνση αυτών

3.1 Το Πρωτόκολλο CAN (Controller Area Network)

Για το πρωτόκολλο CAN υπάρχουν πολλές απόψεις όσον αφορά την αξιοπιστία του διαύλου όταν σημαντικά δεδομένα πρέπει να σταλούν αφού υπάρχει η πιθανότητα κάποια bits ενός μηνύματος να επηρεαστούν από τον θόρυβο ή διάφορες λανθασμένες ηλεκτρομηχανικές λειτουργίες.

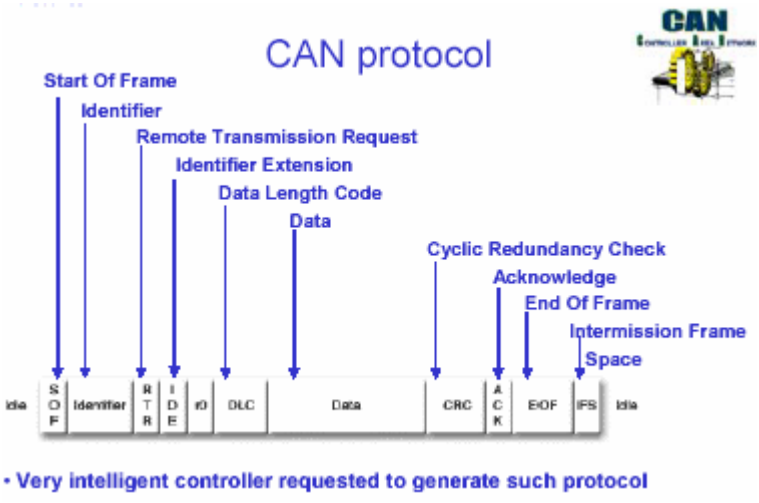
Τουλάχιστον ένας κεραμικός ταλαντωτής(resonator) και πιθανόν ένας κρύσταλλος συχνότητας χρειάζονται για να παραχθούν οι ακριβείς χρονισμοί που χρειάζονται. Το ρολόι και τα δεδομένα συνδυάζονται και 6 ‘high’ bits στην σειρά θεωρούνται σαν λάθος στο καλώδιο. Έτσι το ρολόι και ο χρονισμός των bits είναι πολύ σημαντικά. Όλες οι συνδεδεμένες μονάδες πρέπει να έχουν τα ίδια timings. Όλες οι μονάδες ελέγχουν για λάθη στα δεδομένα σε οποιοδήποτε σημεία των καλωδιώσεων, έτσι ώστε να μπορούν να αναφέρουν το λάθος και το μήνυμα να ξανασταλθεί.

Όπως στο I^2C έτσι και στο CAN είναι απαραίτητη η σύνδεση εξωτερικών pull-up αντιστάσεων στο Bus. Όταν ο πομποδέκτης οδηγεί τον δίαυλο, του εφαρμόζει μια τάση που ονομάζεται «κυρίαρχη»(‘dominant’) κατάσταση. Ο αναγνωριστής(identifier) δείχνει την σημασία των δεδομένων και όχι των παραλήπτη. Έτσι όλοι οι συνδεδεμένοι κόμβοι λαμβάνουν, διαβάζουν τον identifier και αποφασίζουν αν θα αντιδράσουν στα δεδομένα ή όχι. Έτσι όλες οι μονάδες ελέγχουν για λάθη το μήνυμα που μεταδίδεται. Η διαιτησία διαδρόμου είναι όπως το I^2C, η συσκευή που στέλνει bit ‘1’ υπερισχύει αυτής που στέλνει bit ‘0’, ενώ όταν ο δίαυλος είναι ελεύθερος οποιοσδήποτε κόμβος μπορεί να στείλει μηνύματα.

- DLC: Μήκος κώδικα
- CRC: Ελέγχει την ορθότητα του μηνύματος
- ACK: Επιβεβαίωση

- Error Frame: Μεταδίδεται από έναν κόμβο για να δηλώσει σφάλμα στον δίαυλο

Παρακάτω φαίνεται το τυπικό σχηματικό μεταφοράς δεδομένων στο CAN bus:



Τα προϊόντα του I^2C είναι συμβατά με αυτά πολλών κατασκευαστών, ενώ το CAN hardware θα επιλεγεί και θα χρησιμοποιηθεί για ένα συγκεκριμένο σύστημα. Μερικοί CAN πομποδέκτες μπορεί να είναι συμβατοί με άλλους αλλά αυτό θα είναι η εξαίρεση στον κανόνα. Το CAN συνήθως σχεδιάζεται για ξεχωριστά συστήματα που δεν πρόκειται να τροποποιηθούν. Συνήθως χρησιμοποιείται για την δυνατότητα του να αποσυνδέεται και να αντικαθιστά κομμάτια που κατακλύζονται από λάθη.

4. Εισαγωγή στο SPI

Το SPI(Serial Peripheral Interface), όπως και το I^2C αναπτύχθηκε με σκοπό την εύκολη επικοινωνία μεταξύ ολοκληρωμένων και τον καλύτερο τρόπο διασύνδεσης των περιφερειακών μονάδων και των μικροελεγκτών μεταξύ τους.

Το SPI επιτρέπει σε δεδομένα των 8-bits να αποστέλλονται σύγχρονα και ταυτόχρονα να λαμβάνονται σύγχρονα δεδομένα με ταχύτητα που φτάνει το 1Mbps.

Για να επιτευχθεί επικοινωνία το SPI χρησιμοποιεί 4 ακροδέκτες:

1. Τον ακροδέκτη SDO(Σειριακά δεδομένα εξόδου)
2. Τον ακροδέκτη SDI(Σειριακά δεδομένα εισόδου)
3. Το σειριακό ρολόι (SCK)

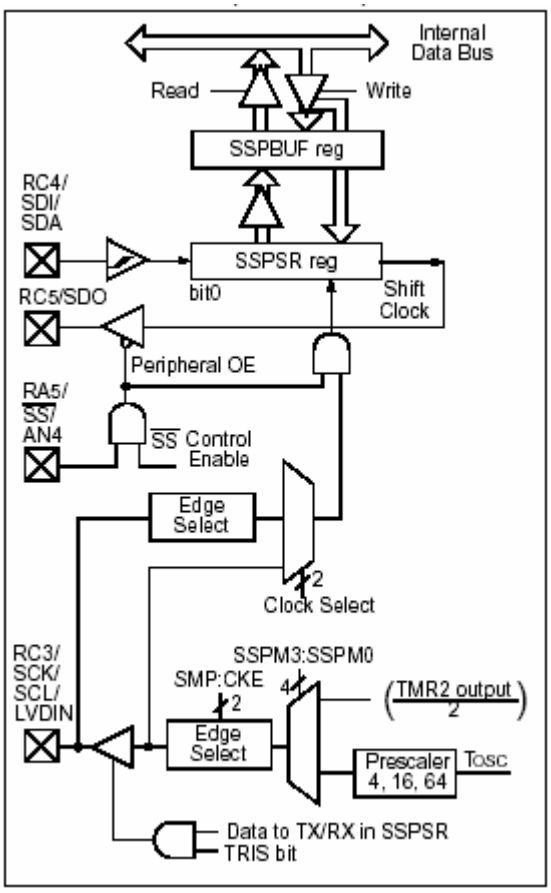
Επιπρόσθετα μπορεί να χρησιμοποιηθεί και ένας τέταρτος ακροδέκτης στην λειτουργία slave: Επιλογή slave (SS'). Σε όλες τις μεταφορές στο SPI το ψηφίο υψηλότερης αξίας στέλνεται πρώτο, όπως εξάλλου συμβαίνει και στο I^2C.

Όταν αρχικοποιείται το SPI, πρέπει να καθοριστούν ορισμένα χαρακτηριστικά. Αυτό γίνεται προγραμματίζοντας τα αντίστοιχα bits των

καταχωρητών SSPCON<5:0> και SSPSTAT<7:6>. Αυτά τα bits ελέγχου επιτρέπουν να καθοριστούν τα ακόλουθα:

- Λειτουργία Master (το SCK είναι έξοδος του ρολογιού)
- Λειτουργία Slave (το SCK είναι είσοδος του ρολογιού)
- Πολικότητα του ρολογιού (ανενεργός κατάσταση του SCK)
- Ακμή ρολογιού (έξοδος δεδομένων κατά την θετική/αρνητική ακμή του SCK)
- Φάση δειγματοληψίας δεδομένων εισόδου
- Επιλογή τύπου λειτουργίας slave(μόνο για λειτουργία slave)

Στο επόμενο σχήμα φαίνεται το μπλοκ διάγραμμα της μονάδας SSP, όταν λειτουργεί το SPI.



Σχήμα 4.1: Μπλοκ διάγραμμα της μονάδας SSP σε λειτουργία SPI

Παρατηρούμε λοιπόν με βάση τα παραπάνω ότι για την λειτουργία SPI χρησιμοποιούνται τέσσερις καταχωρητές των οποίων τα bits έχουν ξεχωριστή σημασία ανάλογα με το αν χρησιμοποιούνται σε SPI ή I^2C λειτουργία. Αυτοί είναι οι: SSPCON, SSPSTAT, SSPBUF, SSPSR(μη-προσβάσιμος άμεσα).

Όπως και στην περίπτωση του I^2C bus το πρωτόκολλο μας προδιαγράφει μόνο το πλήθος των γραμμών, τη λειτουργικότητα τους και την μορφή των δεδομένων μέσα στον δίαυλο. Ανάλογα με τις συσκευές που «συνομιλούν» θα πρέπει να ανταλλάσσονται κάποια δεδομένα που αρχικοποιούν το είδος της μεταφοράς. Επίσης αξίζει να τονίσουμε ότι στο SPI

ο αριθμός των συσκευών που είναι συνδεδεμένες στον δίαυλο είναι περίπου ανάλογος με τον αριθμό των καλωδιώσεων.

5. Σύγκριση I^2C, CAN, SPI

Μετά την αναφορά μας στα πρωτόκολλα SPI, I^2C και CAN μπορούμε να συνοψίσουμε τα κυριότερα πλεονεκτήματα και μειονεκτήματα τους.

Οι περισσότερες CAN συσκευές δεν είναι plug & play και αυτό γιατί τα περισσότερα chips χρησιμοποιούνται σε συγκεκριμένο σύστημα στο οποίο δεν μπορεί αργότερα να συνδεθεί τίποτα παραπάνω. Αυτό συμβαίνει γιατί κάθε προστιθέμενο chip αναμένεται να πάρει μέρος σε όλες τις μεταδόσεις δεδομένων χωρίς όμως να ξέρει την ταχύτητα του ρολογιού και έτσι δεν θα μπορεί να συγχρονιστεί με αποτέλεσμα να μην αφήνει το σύστημα να λειτουργήσει καθόλου. Το CAN και το I^2C χρησιμοποιούν software διευθυνσιοδότηση για να δηλώσουν τις συνδεδεμένες συσκευές που συμμετέχουν στην μετάδοση των δεδομένων αφού όλες είναι συνδεδεμένες στο ίδιο καλώδιο. Το I^2C είναι το καλύτερο Bus που μπορεί να χρησιμοποιηθεί όταν θέλουμε συσκευές να προστίθενται ή να αφαιρούνται από ένα σύστημα.

Για το SPI μπορούμε να πούμε ότι το πλήθος των καλωδιώσεων καθώς και οι χαμηλότερες ταχύτητες που μπορεί να φτάσει το κάνουν να μειονεκτεί έναντι στο I^2C.

Τέλος περιληπτικά έχουμε για τα 3 αυτά πρωτόκολλα έχουμε τα εξής:

CAN: γρήγορο αλλά σύνθετο, κυρίως χρησιμοποιείται στα ηλεκτρονικά των αυτοκινήτων και έχει ακριβό firmware.

SPI: γρήγορο, αποδεκτό παγκοσμίως, χαμηλού κόστους, μεγάλη γκάμα χρησιμοποίησης του αλλά δεν είναι plug & play

I^2C: Απλό, αποδεκτό παγκοσμίως, plug & play δυνατότητα, μεγάλη γκάμα χρησιμοποίησης του και σχετικά χαμηλό κόστος

6. Σύστημα Μεταφοράς Δεδομένων Μέσω I^2C

6.1 Περιγραφή του Συστήματος Μεταφοράς Δεδομένων

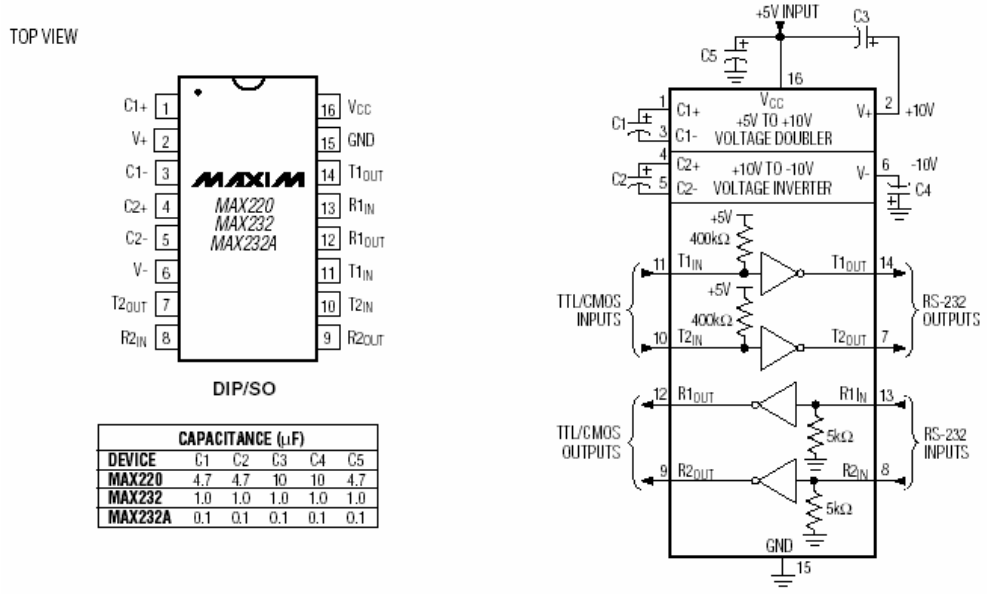
Το σύστημα αποτελείται από δύο microcontrollers (μικροελεκτές) PIC16F876A τοποθετημένους σε δυο διαφορετικές πλακέτες. Ο ένας microcontroller έχει το ρόλο του Master, οπότε και ελέγχει το I^2C bus. Ο δεύτερος microcontroller έχει το ρόλο του Slave και ανταποκρίνεται στις απαιτήσεις του Master. Στην πλακέτα του Slave υπάρχει και μια σειριακή θύρα για την επικοινωνία με τον ηλεκτρονικό υπολογιστή. Η διασύνδεση των

δυο πλακετών γίνεται μέσω των κατάλληλων ακροδεκτών των μικροελεγκτών, δηλαδή των SDA, SCL και ενός breadboard. Σημαντικός είναι και ο ρόλος των pull-up αντιστάσεων που συνδέονται πάνω στο I^2C bus.

Ο Slave δέχεται τα δεδομένα από τον Master και στην συνέχεια τα παρουσιάζει μέσω της σειριακής θύρας στην οθόνη του υπολογιστή όταν συνδεθεί με αυτόν.

6.1.1 Υποσύστημα σειριακής επικοινωνίας

Ο μικροελεγκτής που χρησιμοποιήθηκε διαθέτει ενσωματωμένο υποσύστημα σειριακής επικοινωνίας. Το μόνο που χρειάζεται για την επικοινωνία με ένα ηλεκτρονικό υπολογιστή είναι η μετατροπή των σημάτων του μικροελεγκτή σε τάσεις κατάλληλες για επικοινωνία με την σειριακή θύρα του Υπολογιστή. Για το λόγο αυτό χρησιμοποιήθηκε το ολοκληρωμένο MAX232A της εταιρίας MAXIM (Σχήμα 6.1). Το ολοκληρωμένο αυτό περιλαμβάνει ένα κύκλωμα γεννήτριας τάσεων +10V και -10V από μια μονή τροφοδοσία των +5V. Στο ίδιο ολοκληρωμένο περιλαμβάνονται 2 δέκτες και 2 οδηγοί εξόδου. Το πρόγραμμα που χρησιμοποιείται για τη λήψη δεδομένων, είναι το ενσωματωμένο πρόγραμμα της CCS, PCWH C compiler IDE v3.212 και συγκεκριμένα το Serial Input/Output Monitor. Το υποσύστημα είναι ρυθμισμένο για εκπομπή δεδομένων σε ταχύτητα 19200 bps με χρήση 8 bits για δεδομένα (data), κανένα bit ισοτιμίας (parity), ένα bit σταματήματος (stop) και με έλεγχο ροής μέσω υλικού. (Hardware flow control).

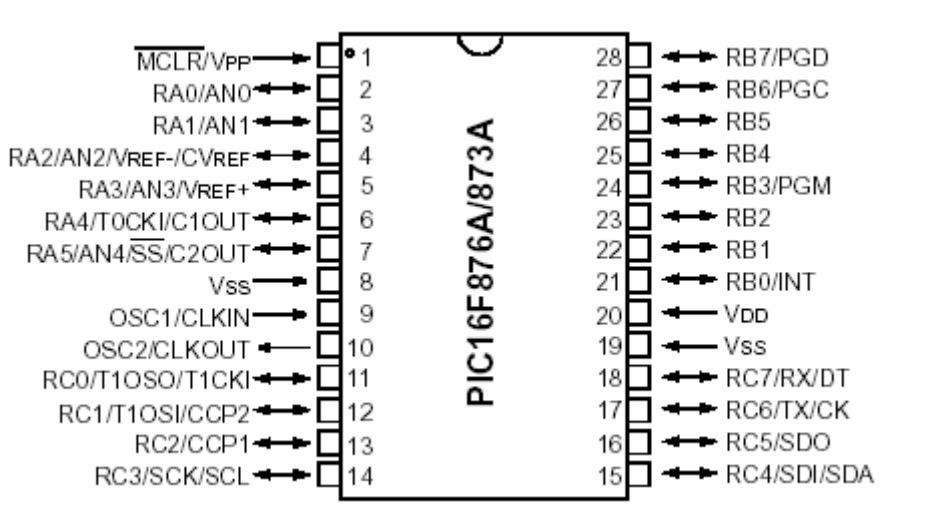


CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Σχήμα 6.1: Το ολοκληρωμένο MAX232A και ο τρόπος σύνδεσής του

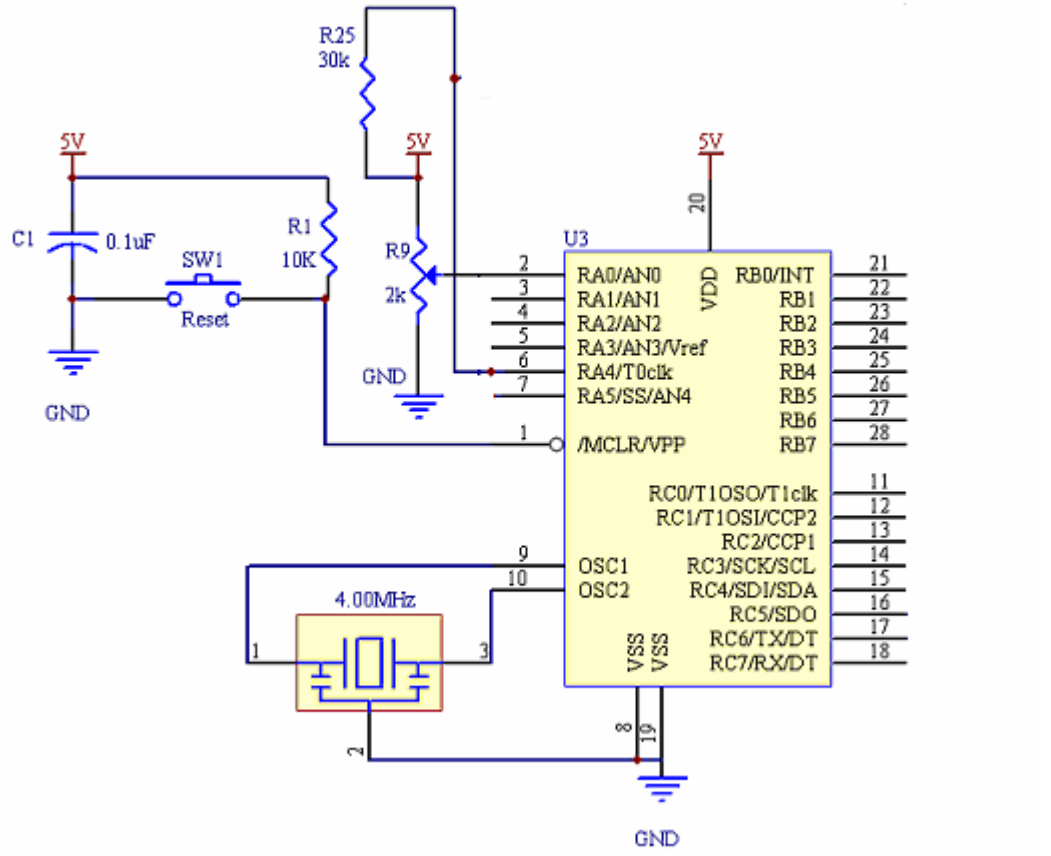
6.2 Μικροελεγκτής

Όλες οι λειτουργίες του συστήματος ελέγχονται από τον μικροελεγκτή PIC16F876A της εταιρείας Microchip.



Σχήμα 6.2: Μικροελεγκτής PIC 16F876A

Στο παρακάτω σχήμα φαίνεται το κύκλωμα του Master PIC16F876A.



Σχήμα 6.3: Σχηματικό της διασύνδεσης του Master μικροελεγκτή PIC με διακόπτη για μηδενισμό του ολοκληρωμένου

Ο πιεστικός διακόπτης χρησιμοποιείται για να κάνει μηδενισμό (Reset) της συσκευής. Πιέζοντας τον διακόπτη γειώνουμε τον ακροδέκτη MCLR και ο μικροελεγκτής διακόπτει όλες τις λειτουργίες του και ξεκινά την εκτέλεση του προγράμματος από την αρχή.

Το σύστημα χρονισμού αποτελείται από ένα κρύσταλλο συχνότητας 4MHz (XT) και οι τιμές των πυκνωτών προσδιορίζονται από τα φυλλάδια προδιαγραφών.

Η οικογένεια PIC ή Peripheral Interface Controller, όπως είναι το πλήρες όνομά τους, αναφέρεται σε μια οικογένεια mid-range μικροελεγκτών της εταιρείας Microchip, με μήκος λέξης εντολής των 14-bit.

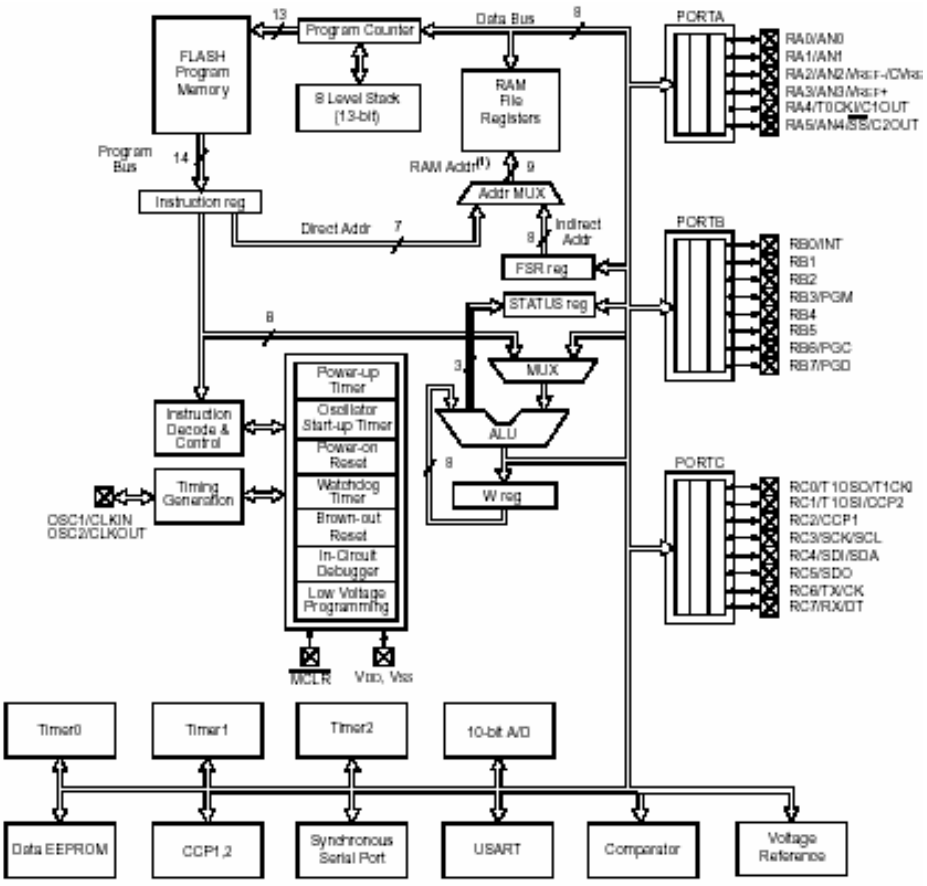
Η δομή τους στηρίζεται στην αρχιτεκτονική Harvard. Η αρχιτεκτονική αυτή επιτρέπει οι εντολές να έχουν διαφορετικό μήκος σε δυαδικά ψηφία από τα δεδομένα. Δίνεται η δυνατότητα να επιλέγεται, ανάλογα με το πλήθος των εντολών, το κατάλληλο μήκος της λέξης εντολής ώστε να επιτυγχάνεται η κωδικοποίηση της κάθε εντολής σε μία μόνο λέξη. Γίνεται δυνατό με αυτό τον τρόπο να μειωθεί σημαντικά την ταχύτητα ανάκλησης (fetch) της κάθε εντολής.

Ένα άλλο χαρακτηριστικό των PIC είναι ότι, για την εκτέλεση μιας εντολής χρειάζεται μόνο ένας κύκλος μηχανής (εκτός των εντολών που αλλάζουν την ροή του προγράμματος). Η ανάκληση μιας εντολής χρειάζεται επίσης μόνο ένα κύκλο μηχανής. Δηλαδή αν έχουμε έναν PIC που δουλεύει με έναν κρύσταλλο των 4MHz τότε θα εκτελεί μια εντολή κάθε 1μs. Οι PIC διαθέτουν επιπλέον μια απλή μονάδα συνεχούς διοχέτευσης (pipeline) με την οποία πετυχαίνουν την εκτέλεση μιας εντολής ανά κύκλο μηχανής χωρίς να χρειάζεται ιδιαίτερα πολύπλοκη αρχιτεκτονική.

Ένα ακόμα χαρακτηριστικό των PIC είναι ότι όλες οι εντολές επιτρέπεται να εκτελούνται σε οποιοδήποτε καταχωρητή ακόμα και σε καταχωρητές ειδικού σκοπού (SFR – Special Function Registers). Το γεγονός ότι δεν υπάρχουν ειδικές περιπτώσεις στη διαχείριση των εντολών σε συνδυασμό με τον μικρό αριθμό, επιτρέπει την γρήγορη και εύκολη εκμάθησή τους. Ο λόγος αυτός, μαζί με τις αυξημένες δυνατότητες και τα περιφερειακά που διαθέτει, ήταν και ο λόγος που επιλέχθηκε ο PIC για την υλοποίηση του συστήματος.

Αρχιτεκτονική της οικογένειας PIC 16FXXX

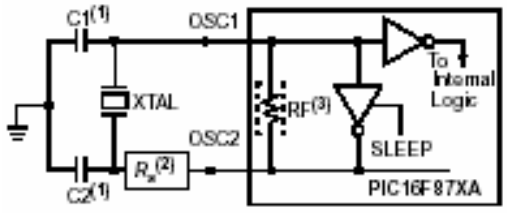
Η δομή ενός PIC διακρίνεται σε τρία μέρη: Τον πυρήνα (Core), τα περιφερειακά (Peripherals) και τα ειδικά χαρακτηριστικά (Special Features) (Σχήμα 6.4).



Device	Program FLASH	Data Memory	Data EEPROM
PIC16F873A	4K words	192 Bytes	128 Bytes
PIC16F876A	8K words	368 Bytes	256 Bytes

Σχήμα 6.4: Μπλοκ διάγραμμα του μικροελεγκτή PIC16F876A

Ο πυρήνας περιέχει όλες τις απαραίτητες συσκευές για τη λειτουργία του μικροελεγκτή, όπως τον ταλαντωτή (Oscillator), τα απαραίτητα κυκλώματα για τη σωστή εκκίνηση του μικροελεγκτή (Reset logic), την κεντρική μονάδα επεξεργασίας (CPU), την αριθμητική μονάδα (ALU), την μνήμη και τη λογική διακοπών (interrupt operation). Στη συγκεκριμένη περίπτωση, ο ταλαντωτής είναι ένας εξωτερικός κρύσταλλος (XT) που λειτουργεί στα 4MHz και συνδέεται με τον μικροελεγκτή μέσω δύο πυκνωτών (Σχήμα 6.5) .



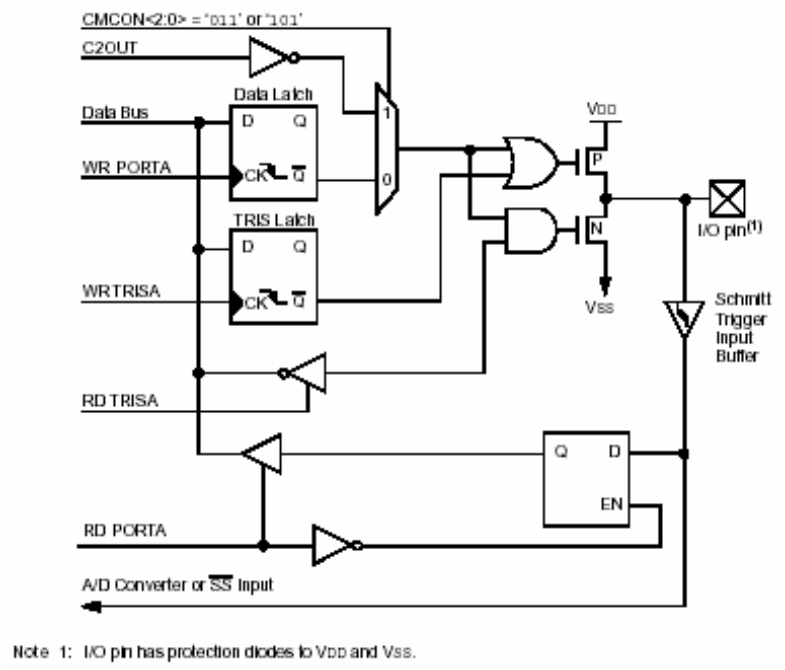
Σχήμα 6.5: Σύνδεση εξωτερικού ταλαντωτή XT

Τα περιφερειακά είναι το πιο ενδιαφέρον κομμάτι ενός μικροελεγκτή, αφού αποτελούν το σημαντικότερο στοιχείο για να αποφασίσει ο χρήστης αν ο συγκεκριμένος μικροελεγκτής είναι κατάλληλος για την εφαρμογή του. Τα περιφερειακά που μπορούν να χρησιμοποιηθούν στην συγκεκριμένη εφαρμογή είναι οι πόρτες εισόδου και εξόδου (I/O Ports), οι χρονιστές (Timers) για την μέτρηση πραγματικού χρόνου και για την παραγωγή διακοπών με συγκεκριμένη συχνότητα, ο μετατροπέας αναλογικού σήματος σε ψηφιακό (A/D Converter) και η σειριακή θύρα (USART).

6.2.1 Ψηφιακές θύρες Εισόδου / Εξόδου

Στις περισσότερες θύρες η κατεύθυνση του κάθε ακροδέκτη, δηλαδή το εάν ένας ακροδέκτης λειτουργεί ως είσοδος ή ως έξοδος, ελέγχεται από τον καταχωρητή ελέγχου κατεύθυνσης που καλείται TRIS. Ο καταχωρητής TRIS<X> ελέγχει αντίστοιχα τη διεύθυνση στη θύρα PORT<X>. Εάν κάποιο bit του καταχωρητή TRIS<X> είναι μονάδα τότε ο αντίστοιχος ακροδέκτης της θύρας συμπεριφέρεται ως είσοδος, ενώ, αν είναι μηδέν, ο ακροδέκτης συμπεριφέρεται ως έξοδος.

Ο καταχωρητής PORT<X> περιέχει τα δεδομένα εξόδου της θύρας. Όταν διαβάζονται τα δεδομένα του καταχωρητή αυτού, δε διαβάζεται ο ίδιος ο καταχωρητής αλλά ότι εμφανίζεται στους ακροδέκτες της θύρας (Σχήμα 6.6).



Note 1: I/O pin has protection diodes to VDD and VSS.

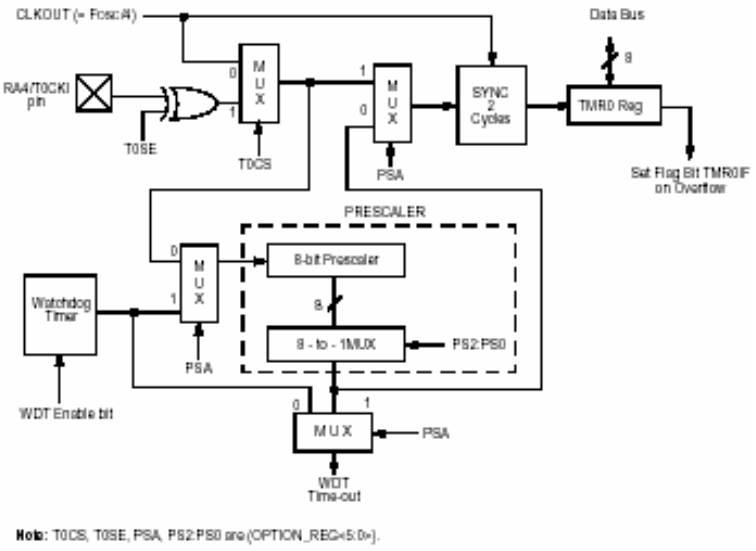
Σχήμα 6.6: Τυπική αρχιτεκτονική ακροδέκτη μιας θύρας

Για οικονομία στο πλήθος των ακροδεκτών οι είσοδοι και οι έξοδοι των περιφερειακών του PIC, όπως είναι οι Α/Ψ μετατροπείς, οι σειριακές θύρες κτλ., χρησιμοποιούν του ίδιους ακροδέκτες με τις ψηφιακές θύρες. Το περιφερειακό αποφασίζει τον τρόπου με τον οποίο λειτουργεί ο ακροδέκτης

που χρησιμοποιεί και μπορεί να παρακάμψει τη λειτουργικότητα του καταχωρητή TRIS.

6.2.2 Χρονιστής

Οι χρονιστές (timers) είναι περιφερειακές συσκευές που αυξάνουν ή μειώνουν περιοδικά την τιμή ενός μετρητή σύμφωνα με κάποια συχνότητα ενός ρολογιού, στην παρούσα εφαρμογή κάθε 4 MHz / 4 = 1 MHz ή σε χρόνο 1 μsec. Ο χρονιστής Timer0 που χρησιμοποιήθηκε, είναι ένας 8 bit μετρητής με ένα διαιρέτη συχνότητας 8 bit (Σχήμα 21). Μπορεί να παράγει μία και μόνο διακοπή όταν το περιεχόμενο του καταχωρητή TMR0 υπερχειλίζει από την τιμή FF στο δεκαεξαδικό σύστημα. Όταν συμβεί υπερχείλιση, η σημαία διακοπής T0IF (interrupt flag), τίθεται. Αν η διακοπή του Timer0 είναι ενεργοποιημένη (bit T0IE), τότε προκαλείται διακοπή.



Σχήμα 6.7: Μπλοκ διάγραμμα χρονιστή Timer0/WDT και διαιρέτη συχνότητας

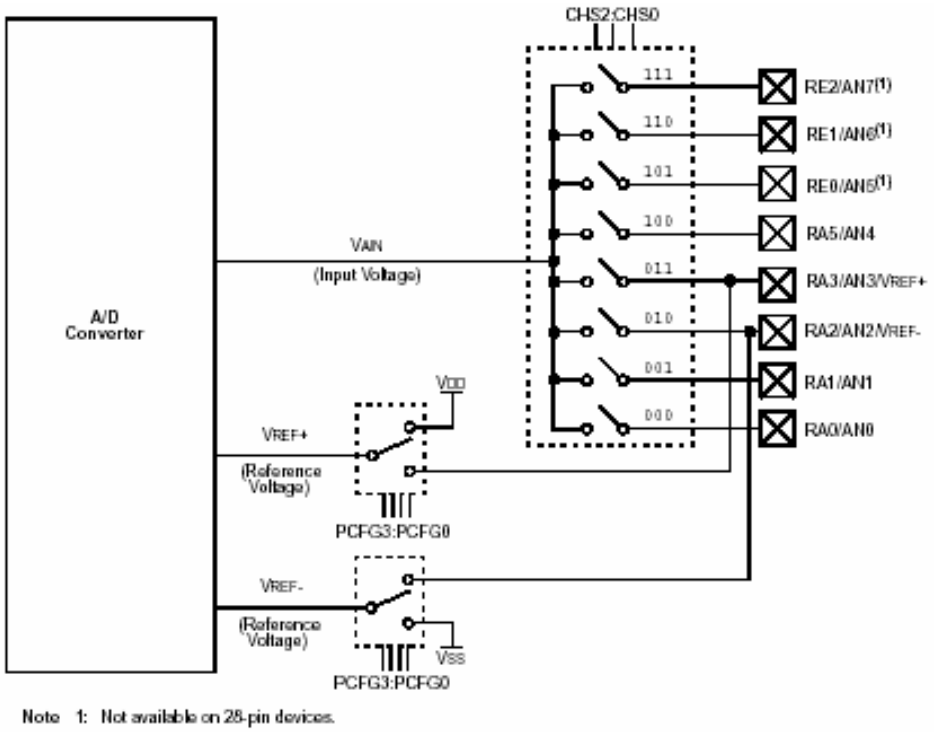
Με την βοήθεια του χρονιστή, ο οποίος μπορεί να μετρήσει μέχρι το 255, και με διαίρεση συχνότητας παραδείγματος χάριν κατά 2, παράγονται διακοπές κάθε 500 μsec. Με χρήση του χρόνου αυτού σαν μονάδα, μπορούμε να μετρήσουμε πραγματικούς χρόνους.

6.2.3 Μετατροπέας αναλογικού σήματος σε ψηφιακό

Ο μετατροπέας αναλογικού σε ψηφιακό παίρνει ως είσοδο ένα αναλογικό σήμα και ως αποτέλεσμα δίνει ένα 10-bit αριθμό που δίνει το ποσοστό της μετρούμενης τάσης σε σχέση με τη τάση αναφοράς. Η μετατροπή γίνεται με την μέθοδο των διαδοχικών προσεγγίσεων (Successive Approximation Method). Ο μετατροπέας είναι ικανός να πραγματοποιήσει μια ακριβή μετατροπή μόνο αν η τάση εισόδου είναι εντός της εμβέλειας του μετατροπέα, που είναι συνήθως από τα 0V έως μία τάση αναφοράς, που

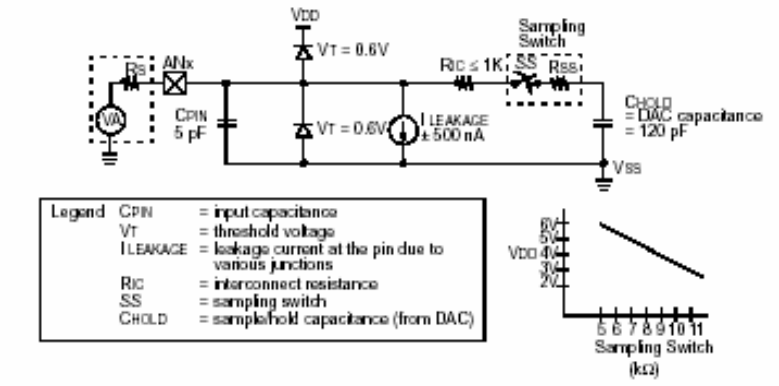
ορίζεται από τον χρήστη. Η τάση αναφοράς μπορεί να είναι, είτε η τάση τροφοδοσίας V_{DD}, είτε η τάση που εμφανίζεται στον ακροδέκτη V_{REF}.

Ο συγκεκριμένος μικροελεγκτής έχει 5 αναλογικές εισόδους. Ως τάση αναφοράς χρησιμοποιήθηκε η τάση τροφοδοσίας V_{DD} που ήταν τα +5V. Επομένως, η τάση που μπορεί να μετατρέψει με ακρίβεια ο μικροελεγκτής κυμαίνεται από 0 έως +5V. Δίνεται επίσης η δυνατότητα να μη χρησιμοποιηθούν και τα 10 bit του αποτελέσματος της μετατροπής, αλλά μόνο τα 8 bit. Με χρήση μόνο των 8 σημαντικότερων bits διευκολύνονται κατά πολύ οι πράξεις και οι υπολογισμοί κατά την επεξεργασία των δεδομένων.



Σχήμα 6.8: Μπλοκ διάγραμμα αναλογικού ψηφιακού μετατροπέα

Οι ακροδέκτες αναλογικής εισόδου είναι συνδεδεμένοι στις εισόδους ενός αναλογικού πολυπλέκτη, που συνδέει το επιλεγμένο κανάλι σε έναν πυκνωτή συγκράτησης (holding capacitor) (Σχήμα 6.9). Ο αναλογικός πολυπλέκτης επιτρέπει πολλαπλές εισόδους να είναι διαθέσιμες για μετατροπή.



Σχήμα 6.9: Μοντέλο αναλογικής εισόδου

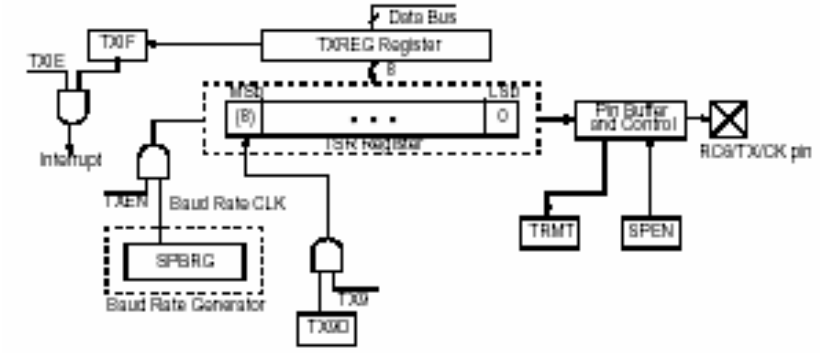
Ο πυκνωτής συγκράτησης (holding capacitor), είναι συνδεδεμένος με την έξοδο του αναλογικού πολυπλέκτη. Όταν η μετατροπή ξεκινάει, ο πολυπλέκτης αποσυνδέει όλες τις εισόδους από τον πυκνωτή συγκράτησης και ο μετατροπέας πραγματοποιεί την μετατροπή στην τάση που είναι αποθηκευμένη στον πυκνωτή με την μέθοδο των διαδοχικών προσεγγίσεων.

Δεδομένου ότι στην ουσία φορτίζεται ένας πυκνωτής από τη τάση εισόδου του Α/Ψ μετατροπέα, ο χρόνος λήψης (acquisition time) είναι ο χρόνος που χρειάζεται να φορτιστεί πλήρως ο πυκνωτής συγκράτησης. Για να γίνει σωστά η μετατροπή θα πρέπει να δοθεί ο απαραίτητος χρόνος σ' αυτό τον πυκνωτή να φορτιστεί και να σταθεροποιηθεί η τάση του στο επίπεδο της τάσης της αναλογικής εισόδου.

Μετά την αναμονή του κατάλληλου χρόνου λήψης, πρέπει να γίνει και η μετατροπή της τάσης από αναλογική σε ψηφιακή ξεκινώντας από το περισσότερο σημαντικό ψηφίο (MSB – Most Significant Bit).

6.2.4 Σειριακή επικοινωνία (USART)

Το USART (Universal Synchronous Asynchronous Receiver Transmitter) μπορεί να λειτουργήσει είτε με σύγχρονο είτε με ασύγχρονο τρόπο για την μετάδοση δεδομένων. Στην συγκεκριμένη εφαρμογή χρησιμοποιήθηκε η ασύγχρονη μετάδοση λόγω στο μικρό αριθμό δεδομένων που πρέπει να σταλούν και της απλότητας στην υλοποίησή της. Στον ασύγχρονο τρόπο λειτουργίας δεν απαιτείται ρολόι για τον συγχρονισμό του συστήματος. Έτσι χρησιμοποιείται ένας ακροδέκτης για την μετάδοση και ένας για την λήψη δεδομένων (Σχήμα 6.10). Τόσο η μετάδοση όσο και η λήψη μπορούν να πραγματοποιηθούν ταυτόχρονα (full duplex). Η μετάδοση και η λήψη ενεργοποιούνται ανεξάρτητα, όταν όμως είναι ενεργοποιημένο το περιφερειακό σειριακής επικοινωνίας, ο έλεγχος και των δύο ακροδεκτών περνάει σ' αυτό και δεν μπορούν αυτοί να χρησιμοποιηθούν σαν ακροδέκτες εισόδου / εξόδου.



Σχήμα 6.10: Μπλοκ διάγραμμα ακροδέκτη λήψης δεδομένων

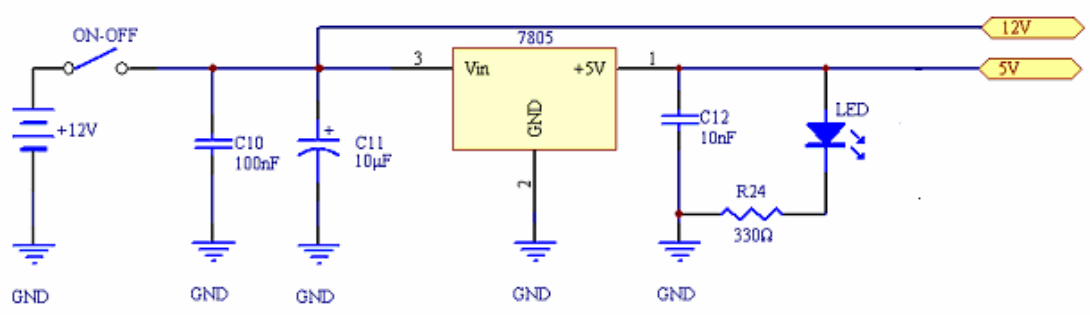
Η πιο διαδεδομένη χρήση του USART και αυτή που χρησιμοποιείται και στο σύστημα αυτό, είναι η επικοινωνία με μία σειριακή θύρα ενός ηλεκτρονικού υπολογιστή, χρησιμοποιώντας το πρωτόκολλο RS-232. Ο κατάλληλος μετατροπέας των τάσεων στα επιθυμητά επίπεδα είναι το ολοκληρωμένο MAX232A όπως έχει ήδη αναφερθεί.

Το USART μπορεί να ρυθμιστεί να στέλνει 8 ή 9 bits δεδομένων με χρήση και του TX9 bit του καταχωρητή TXSTA. Για την εφαρμογή αυτή τα 8 bit κρίθηκαν αρκετά. Μόλις τα δεδομένα είναι έτοιμα προς αποστολή, τα 8 bit μεταφέρονται στον καταχωρητή TXREG. Μόλις εγγραφούν τότε αυτά μεταφέρονται στον καταχωρητή μετάδοσης με ολίσθηση (transmit shift register). Από εκεί, με κάθε παλμό ενός ρολογιού χρονισμού, μεταδίδονται από τον ακροδέκτη TX αφού τους προστεθεί ένα bit ξεκινήματος (start bit) και ένα bit σταματήματος (stop bit). Η χρήση του ξεχωριστού καταχωρητή μετάδοσης με ολίσθηση επιτρέπει σε νέα δεδομένα να γραφτούν στον καταχωρητή TXREG ενώ ακόμα στέλνονται τα προηγούμενα δεδομένα, βελτιώνοντας έτσι την ταχύτητα της μετάδοσης.

Ο μικροελεγκτής έχει προγραμματιστεί μέσω του καταχωρητή TXSTA και SPBRG για να μεταδίδει δεδομένα με ταχύτητα 19200 bps με χρήση 8 bits για δεδομένα (data), κανένα bit ισοτιμίας (parity), ένα bit σταματήματος (stop) και με έλεγχο ροής μέσω υλικού (Hardware flaw control). Η μετάδοση των δεδομένων αρχίζει μόνο έπειτα από χειρισμό του χρήστη, δηλαδή πάτημα ενός διακόπτη. Με το πάτημα του διακόπτη, πραγματοποιείται αλλαγή της κατάστασης του ακροδέκτη RB0. Ο ακροδέκτης έχει την ιδιότητα να προκαλεί διακοπή όταν συμβεί αλλαγή της κατάστασής του. Με το που θα γίνει μια τέτοια διακοπή, σταματάει η ροή του προγράμματος και εξυπηρετείται η διακοπή, δηλαδή στέλνονται τα δεδομένα μέσω της σειριακής θύρας σε έναν ηλεκτρονικό υπολογιστή.

6.2.5 Τροφοδοσία- Σταθεροποιητής Τάσης 5V

Ο σταθεροποιητής τάσης χρησιμοποιείται για την δημιουργία σταθερής τάσης 5V για την τροφοδοσία του PIC και άλλων κυκλωμάτων. Στο κύκλωμα αυτό χρησιμοποιείται ένα κόκκινο LED για ένδειξη της κατάστασης λειτουργίας ON. Ο σταθεροποιητής τάσης 7805 είναι υπεύθυνος για ένα μεγάλο ποσοστό κατανάλωσης ισχύος στο σύστημα.



Σχήμα 6.11: Σχηματικό της τροφοδοσίας, σταθεροποιητή τάσης 5V

Οι πυκνωτές στο κύκλωμα ονομάζονται και decoupling capacitors και σκοπό έχουν τη μείωση του πλάτους του κωδωνισμού της τάσης τροφοδοσίας.

6.2.6 Λειτουργίες του μικροελεγκτή

Ο μικροελεγκτής είναι υπεύθυνος για την λειτουργία και τον συγχρονισμό όλων των υποσυστημάτων του συστήματος. Έτσι ο μικροελεγκτής είναι υπεύθυνος για τις παρακάτω λειτουργίες:

- Παραγωγή και εξυπηρέτηση των απαραίτητων διακοπών του I^2C αλλά και άλλων γενικών διακοπών.
- Παραγωγή και εξυπηρέτηση της διακοπής για την παρουσίαση και αποστολή των αποτελεσμάτων. Κατά τη διάρκεια αυτής της διακοπής, στέλνονται τα δεδομένα, μέσω του πρωτοκόλλου I^2C ή μέσω της σειριακής θύρας, σε ηλεκτρονικό υπολογιστή.

7. Περιγραφή και Επεξήγηση του Προγράμματος του Master και Slave Μικροελεγκτή PIC16F876A

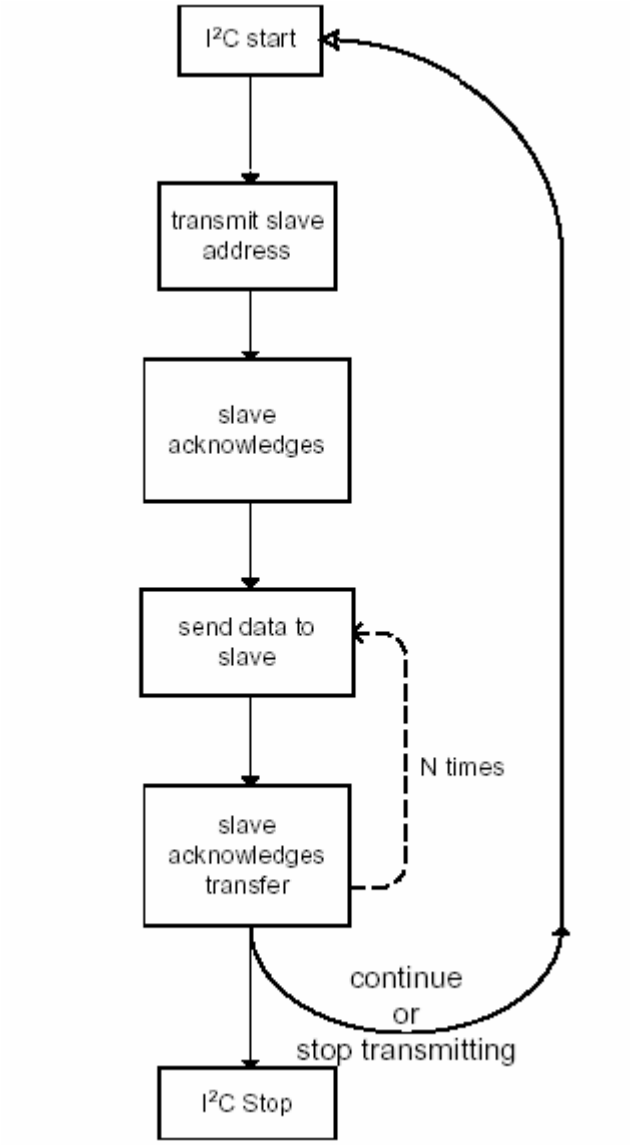
7.1 MASTER

Το πρόγραμμα του Master μικροελεγκτή είναι υπεύθυνο για τον έλεγχο του συστήματος. Ο μικροελεγκτής οδηγεί το I^2C Bus και ορίζει κάθε στιγμή την κατάσταση που θα βρίσκεται.

Παρακάτω θα γίνει μια περιληπτική επεξήγηση του κώδικα. Ολόκληρος ο κώδικας (firmware) του Master μικροελεγκτή παρουσιάζεται στο παράρτημα Δ με λεπτομερείς επεξηγήσεις για κάθε εντολή. Επίσης στο παράρτημα Ε παρουσιάζεται ένα output μέσω της σειριακής θύρας του slave και του προγράμματος του PCWH C, Serial Input/Output Monitor.

Ο κώδικας είναι γραμμένος σε γλώσσα C++ και ο compiler που χρησιμοποιήθηκε είναι της CCS, ο PCWH C compiler IDE v3.212 ενώ με το κατάλληλο plug-in του MPLAB IDE v7.01 και της συσκευής PICSTART PLUS γίνεται η «μετάφραση» σε γλώσσα assembly.

Στο παρακάτω σχήμα φαίνονται οι διάφορες λειτουργίες που επιτελούνται από τον Master PIC για την μεταφορά(εγγραφή) δεδομένων στον slave.



Σχήμα 7.1: Διάγραμμα Ροής Μετάδοσης του Master στο I^2C

Οποιοδήποτε πρόγραμμα που προορίζεται για έναν μικροελεγκτή PIC ξεκινά με δήλωση του μικροελεγκτή π.χ. #include <16F876A.H> και τη δήλωση #fuses που χρησιμοποιείται για τον έλεγχο των εσωτερικών προγραμματιζόμενων ενωτήρων του ελεγκτή PIC.

Στην δήλωση #fuses λοιπόν τα configuration bits παίρνουν τις παρακάτω τιμές :

Oscillator : XT
Watchdog timer : OFF(NOWDT)
Power-up timer : ON(PUT)
Brown Out Detect : OFF(NONROWNOUT)
Low Voltage : OFF(NOLVP)
Code Protect: OFF(NOPROTECT)

Το configuration Word σύμφωνα με τα παραπάνω γίνεται 3F31h.

Δήλωση μεταβλητών

Η δήλωση των μεταβλητών γίνεται με την εντολή #define η οποία εξισώνει κάθε καταχωρητή (μεταβλητή) με την τιμή στην μνήμη του PIC που εμείς επιθυμούμε. Ενώ με την εντολή #byte δηλώνεται σε ποια διεύθυνση μνήμης του pic βρίσκεται ο κάθε καταχωρητής του I^2C.

Αρχή Προγράμματος

Μετά τις εντολές αρχής προγράμματος και αρχής της εξυπηρέτησης των διακοπών γίνεται αρχικοποίηση των πόρτων εισόδου / εξόδου του PIC.

Αρχικοποίηση των Ακροδεκτών του PIC

Η πόρτα RC3/SCK/SCL χρησιμοποιείται σαν είσοδος για την γραμμή ρολογιού του I^2C και η πόρτα RC4/SDI/SDA σαν είσοδος για την γραμμή δεδομένων του I^2C. Επίσης οι πόρτες RC6/TX/CK και RC7/RX/DT χρησιμοποιούνται σαν είσοδοι σε περίπτωση που θέλουμε να ενεργοποιήσουμε την σειριακή θύρα του PIC. Οι υπόλοιπες αρχικοποιούνται σαν πόρτες εξόδου.

Αρχικοποίηση της Ενεργοποίησης της I^2C Master λειτουργίας

Σε αυτό το μέρος του κώδικα αρχικοποιείται η λειτουργία I^2C με την βοήθεια της εντολής #use I^2C καθώς επίσης και ο ρυθμός μετάδοσης των bits.

Ενεργοποίηση Διακοπών

Θέτοντας κατάλληλα τον καταχωρητή INTCON ενεργοποιούμε τις διακοπές του timer0, τις διακοπές των περιφερειακών και γενικά μπορούμε να ενεργοποιήσουμε όλες τις διακοπές του PIC (GIE=1), όλες τις περιφερειακές διακοπές με το PEIE=1, αλλά και τις διακοπές της SSP θέτοντας το bit SSPIE=1 (PIE1<3>). Μετά από αυτό ο μικροελεγκτής τίθεται σε κατάσταση αναμονής περιμένοντας να συμβεί κάποια διακοπή.

Αρχικοποίηση της Σειριακής Θύρας του PIC

Σε αυτό το μέρος του κώδικα αρχικοποιείται η λειτουργία μετάδοσης των δεδομένων μέσω του RS-232 στον υπολογιστή, με την βοήθεια της εντολής #use rs232, όπου αναφέρεται και η ταχύτητα μετάδοσης των δεδομένων καθώς και οι κατάλληλοι ακροδέκτες του PIC που χρησιμεύουν για την μετάδοση και λήψη των δεδομένων.

7.2 ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΛΕΙΤΟΥΡΓΊΑΣ ΤΟΥ MASTER

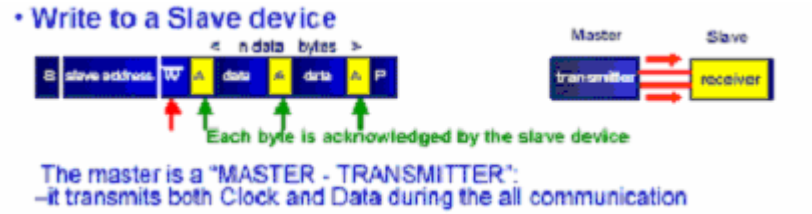
Στο σύστημα μας ο Master ενεργοποιεί την έναρξη της I^2C διαδικασίας για την μεταφορά των επιθυμητών δεδομένων θέτοντας το SEN bit του SSPCON2(SSPCON2<0>). Προτού όμως γίνει η δήλωση της έναρξης μεταφοράς πρέπει να δηλώσουμε τα pin RC3, RC4 του Master PIC σαν εισόδους, να θέσουμε το bit SSPEN (SSPCON<5>) ίσο με ένα, τα bits SSPCON <3:0> ίσα με 1000 για να δηλωθεί ο PIC ως Master αλλά και τον επιθυμητό ρυθμό μετάδοσης των δεδομένων μας, που στην συγκεκριμένη περίπτωση είναι 100KHz(standard mode).

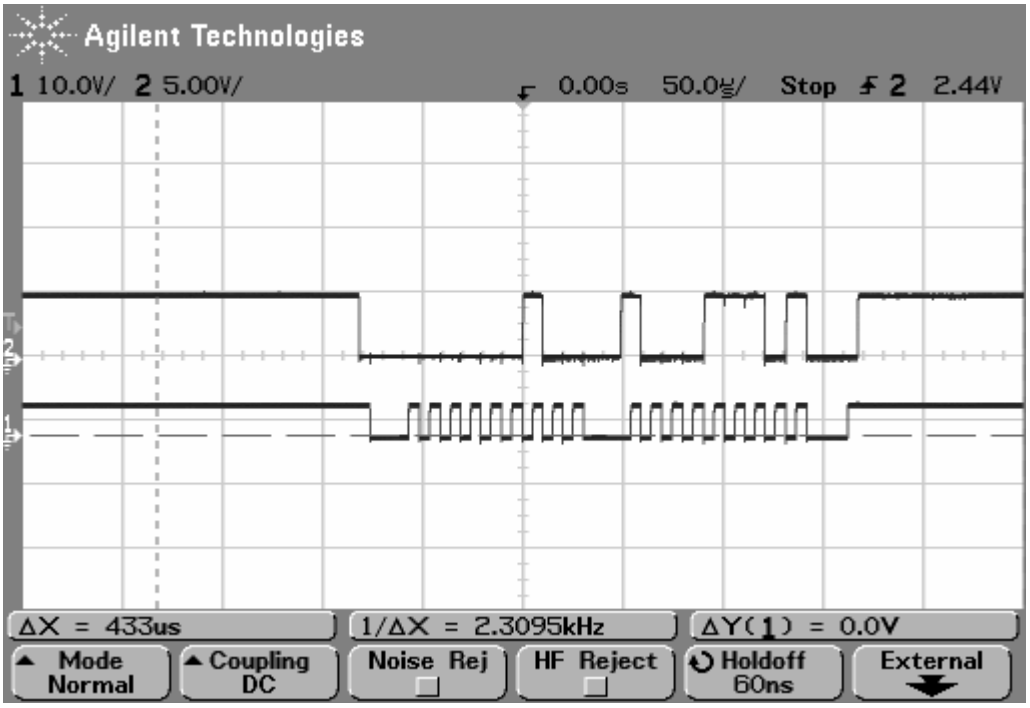
Κατόπιν θέτουμε το bit SEN και η σημαία SSPIF τίθεται ίση με ένα και η μονάδα MSSP περιμένει τον απαραίτητο χρόνο για να αρχίσει νέα λειτουργία. Ο μηδενισμός του SSPIF καθορίζεται από το software.

Στην συνέχεια δηλώνουμε την διεύθυνση του Slave μικροελεγκτή που θέλουμε να μεταδώσουμε τα δεδομένα, Η διεύθυνση αυτή «φορτώνεται» στον SSPBUF, οπότε τίθεται και το bit BF(SSPSTAT<0>) ίσο με ένα και εν συνεχεία μεταδίδεται μέσω του SDA pin μέχρι όλα τα bits να μεταφερθούν. Εδώ πρέπει να προσέξουμε το τελευταίο bit της διεύθυνσης, R/W', να είναι μηδέν αφού αυτό είναι που δηλώνει πως ο Master πρόκειται να μεταδώσει δεδομένα στον Slave. Έτσι δηλώνουμε την διεύθυνση του Slave ως '00000010'=0x02.

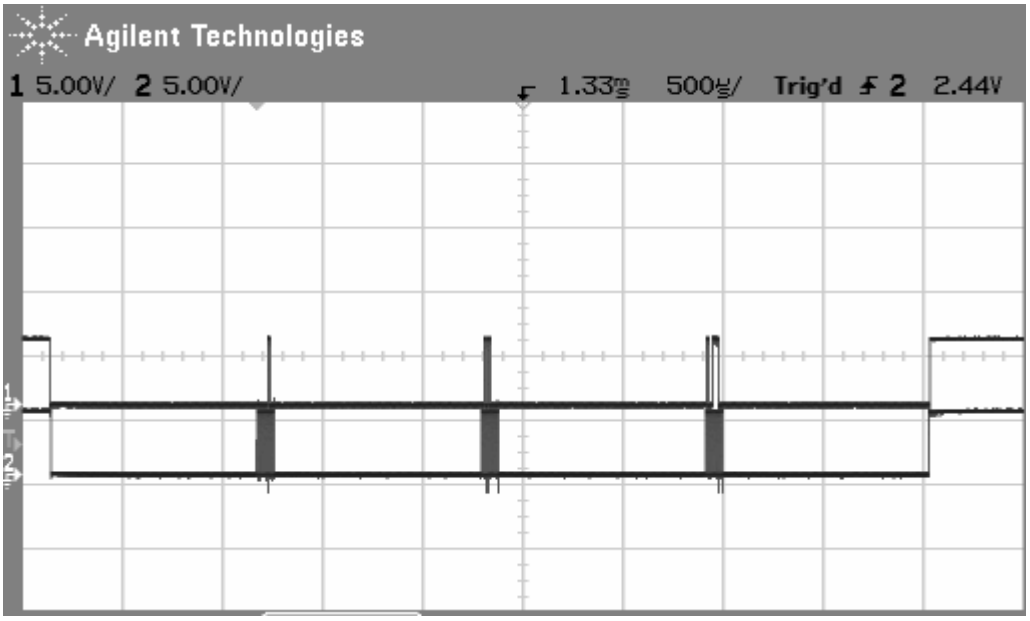
Όταν μεταδοθούν και τα 8 bits η σημαία BF μηδενίζεται και ο Master απελευθερώνει την γραμμή SDA. Ο Master έπειτα αναμένει την επιβεβαίωση(ACK) ή την μη- επιβεβαίωση(NACK) από τον Slave για να καταλάβει αν έλαβε ή όχι αντίστοιχα, την διεύθυνση του σωστά. Σε περίπτωση που η μονάδα MSSP λάβει ACK, το bit ACKSTAT(SSPCON2<6>) γίνεται μηδέν. Αμέσως μετά την λήψη του ACK, η μονάδα MSSP παράγει διακοπή στο τέλος του ένατου παλμού θέτοντας το SSPIF bit(PIR1<3>) ίσο με ένα. Αφού λοιπόν έχει λάβει τις επιβεβαιώσεις του ο Master μετά αρχίζει «φορτώνει» τον SSPBUF με τα οκτώ bits δεδομένων. Τα δεδομένα μετακινούνται μέσω του SDA pin μέχρι να μεταδοθούν όλα τα bits. Με την σωστή λήψη των δεδομένων ο Slave στέλνει ένα ACK bit που η τιμή του γράφεται στον SSPCON2(SSPSON<6>). Η μονάδα MSSP παράγει διακοπή στο τέλος του ένατου παλμού θέτοντας το SSPIF bit(PIR1<3>) ίσο με ένα. Μετά τον μηδενισμό του SSPIF και με την ίδια διαδικασία φορτώνεται στον SSPBUF το δεύτερο byte δεδομένων που θέλουμε να μεταδώσουμε. Ο Master έπειτα αναμένει την επιβεβαίωση(ACK) ή την μη- επιβεβαίωση(NACK) των δεδομένων που έστειλε από τον Slave, για να καταλάβει αν τα έλαβε ή όχι αντίστοιχα. Λαμβάνει λοιπόν το ACK bit στην αρνητική ακμή του ένατου παλμού, το SSPIF bit(PIR1<3>) τίθεται ίσο με ένα, το bit BF μηδενίζεται και ο SSPBUF αδειάζει περιμένοντας το επόμενο byte δεδομένων. Ο αριθμός των bytes που στέλνουμε στην συγκεκριμένη εφαρμογή είναι τέσσερα.

Η όλη διαδικασία μετάδοσης δεδομένων επαναλαμβάνεται και σταματά μόλις ο Master ενεργοποιήσει μια Stop κατάσταση θέτοντας το PEN bit του SSPCON2.(SSPCON2<2>). Η διακοπή παράγεται με το που το bit SSPIF γίνει ένα και μόλις η κατάσταση λήξης ολοκληρωθεί. Η διαδικασία εγγραφής φαίνεται στο παρακάτω σχήμα.





Σχήμα 7.2: Μετάδοση Διεύθυνσης και Ενός Byte Δεδομένων στον Slave(0x02)



Σχήμα 7.3: Μετάδοση Διεύθυνσης και Δύο Byte Δεδομένων στον Slave(0x02)

7.3 ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΟΥ SLAVE

Η επιλογή του PIC σε λειτουργία slave στο I^2C, προϋποθέτει ότι τα pin SDA, SCL θα προγραμματιστούν ως είσοδοι θέτοντας 1 στα αντίστοιχα TRIS bits(TRISC<4:3>). Ακόμα τα bits SSPCON <3:0> παίρνουν τιμή 0110 για να δηλωθεί ο PIC ως Slave. Επίσης γίνεται αρχικοποίηση της SSP διακοπής, που κατά την διάρκεια της γίνεται ο έλεγχος από τον slave pic των δεδομένων που λαμβάνει από τον master και ανάλογα αν τα bytes είναι διεύθυνσης ή δεδομένων και αν ο master απαιτεί εγγραφή ή λήψη δεδομένων πηγαίνει στην σωστή κατάσταση. Ο έλεγχος γίνεται μέσω των bits του καταχωρητή SSPSTAT. Απαραίτητη είναι η δήλωση της διεύθυνσης του slave pic και ενός καταχωρητή όπου θα αποθηκεύονται τα δεδομένα που θα στέλνει ο master. Ο καταχωρητής αυτός στην περίπτωση μας είναι ο slave_buffer[ ], έχει μήκος 32 bytes και έχουμε προβλέψει πως σε περίπτωση γεμίσματος του slave_buffer[ ], αυτός κατόπιν να καθαρίζεται.

Όταν λοιπόν η διεύθυνση που στέλνει ο master ταιριάζει, το hardware αυτόματα παράγει ένα παλμό αναγνώρισης (ACK) και φορτώνει τον καταχωρητή SSPBUF με την τιμή που έλαβε και που την στιγμή αυτή βρισκόταν στον SSPSR.

Κάτω από ορισμένες συνθήκες η μονάδα SSP δεν θα δώσει αυτόν τον παλμό αναγνώρισης (ACK) . Οι συνθήκες αυτές μπορούν να ισχύουν είτε και οι δύο είτε η μια με το ίδιο αποτέλεσμα και είναι οι εξής:

- Το bit του απομονωτή BF (SSPSTAT<0>) έγινε 1 προτού ολοκληρωθεί το μήνυμα
- Το bit υπερχείλισης SSPOV (SSPCON<6>) έγινε 1 προτού ολοκληρωθεί το μήνυμα

Σε αυτή την περίπτωση η τιμή του καταχωρητή SSPSR δεν φορτώνεται στον SSPBUF αλλά τα bits SSPIF(PIR1<3>) και SSPOV γίνονται ένα(1).

Όταν η μονάδα MSSP ενεργοποιηθεί, περιμένει να συμβεί μια κατάσταση έναρξης. Μόλις αυτό συμβεί, τα 8 bits ολισθαίνουν μέσα στον καταχωρητή SSPSR. Όλα τα εισερχόμενα bits δειγματοληπτούνται κατά την θετική ακμή της γραμμής σήματος του ρολογιού(SCL). Η τιμή του καταχωρητή SSPSR <7:1> συγκρίνεται με την τιμή του καταχωρητή SSPADD, κατά την αρνητική ακμή του όγδοου παλμού ρολογιού. Αν οι δύο διευθύνσεις ταιριάζουν, τα bits BF και SSPOV μηδενίζονται και συμβαίνουν τα ακόλουθα:

- Το περιεχόμενο του καταχωρητή SSPSR φορτώνεται στον SSPBUF κατά την αρνητική τιμή του όγδοου παλμού του ρολογιού.
- Το bit BF (γεμάτος απομονωτής) γίνεται 1 στην αρνητική τιμή του όγδοου παλμού του ρολογιού.
- Δημιουργείται ένας παλμός ACK
- Το bit διακοπής SSPIF γίνεται 1(και προκαλείται αν είναι ενεργοποιημένη η αντίστοιχη διακοπή) κατά την αρνητική ακμή του ένατου παλμού SCL.

Όταν το bit R/W' του byte διεύθυνσης είναι μηδέν και έχουμε ταίριασμα διεύθυνσης, το bit R/W' του SSPSTAT γίνεται μηδέν. Η διεύθυνση που λήφθηκε φορτώνεται στον SSPBUF και η γραμμή SDA κρατείται σε χαμηλή στάθμη.

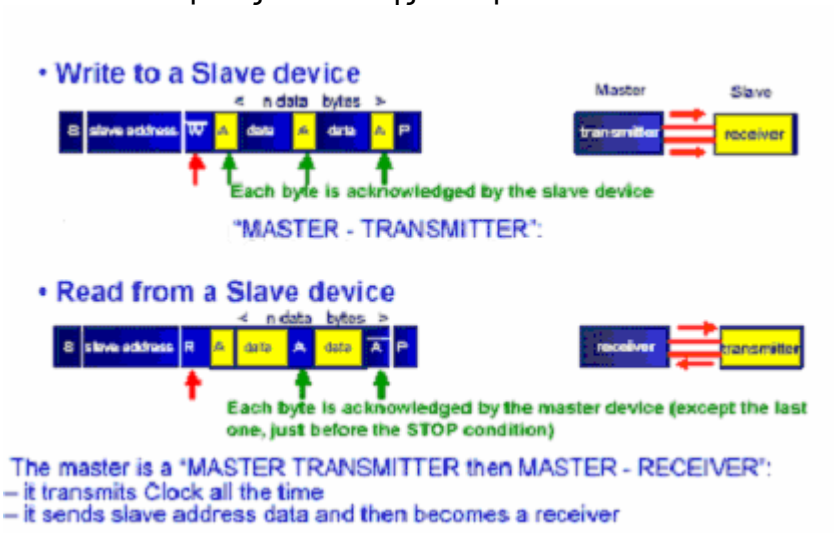
Στην περίπτωση υπερχείλισης του byte διεύθυνσης, παράγεται παλμός Μη-αναγνώρισης (NACK). Έτσι, όταν λαμβάνεται ένα byte με συνθήκη υπερχείλισης και προσπαθεί να μεταφερθεί από τον SSPSR στον SSPBUF, δεν δίνει παλμό αναγνώρισης.

Οι καταστάσεις του SSPSTAT που μπορούν να συμβούν είναι οι εξής:

1. Λειτουργία εγγραφής I2C, το τελευταίο byte ήταν byte διεύθυνσης
   SSPSTAT bits: S = 1, D_A = 0, R_W = 0, BF = 1
2. Λειτουργία εγγραφής I2C, το τελευταίο byte ήταν byte δεδομένων
   SSPSTAT bits: S = 1, D_A = 1, R_W = 0, BF = 1
3. Λειτουργία ανάγνωσης I2C, το τελευταίο byte ήταν byte διεύθυνσης
   SSPSTAT bits: S = 1, D_A = 0, R_W = 1, BF = 0
4. Λειτουργία ανάγνωσης I2C, το τελευταίο byte ήταν byte δεδομένων
   SSPSTAT bits: S = 1, D_A = 1, R_W = 1, BF = 0
5. Τερματισμός λογικής σκλάβου μετά την μετάδοση NACK από τον master
   SSPSTAT bits: S = 1, D_A = 1, R_W = 0, BF = 0

Τέλος αν το SEN (SSPCON<0>) bit ενεργοποιηθεί, τα RC3/SCK/SCL θα κρατηθούν σε χαμηλή στάθμη(clock stretch) ακολουθώντας κάθε μεταφορά δεδομένων. Το ρολόι πρέπει τότε να απελευθερωθεί θέτοντας το bit CKP (SSPCON<4>) ίσο με 1.

Παρακάτω παρουσιάζεται ένα σχήμα όπου διακρίνεται η λειτουργία ενός master σαν πομπός και δέκτης δεδομένων.



8. Συμπεράσματα και Βελτιώσεις του Συστήματος

8.1 Συμπεράσματα

Το σύστημα που έχει κατασκευαστεί δουλεύει αρκετά αποδοτικά. Από την όλη διαδικασία σχεδίασης, κατασκευής και μέτρησης των επιδόσεων του συστήματος, έγιναν αρκετές παρατηρήσεις για την λειτουργικότητα του συστήματος αλλά και για την αξιοπιστία των αποτελεσμάτων.

Βεβαίως σε μία δεύτερη έκδοσή του συστήματος θα μπορούσαν να γίνουν βελτιώσεις ώστε το σύστημα να μπορεί να χρησιμοποιηθεί για περισσότερες εφαρμογές, με μεγαλύτερες ταχύτητες μεταφοράς δεδομένων αλλά και σε μεγαλύτερες αποστάσεις από ότι το σύστημα που έχει κατασκευαστεί στα πλαίσια αυτής της διπλωματικής εργασίας.

8.2 Βελτιώσεις

Όπως έχουμε ήδη αναφέρει το I^2C πρωτόκολλο είναι αρκετά απλό στο να κατανοηθεί και να χρησιμοποιηθεί σε σχέση με άλλα πρωτόκολλα επικοινωνίας όπως το CAN ή το SPI. Αυτή η απλότητα του βέβαια δεν το εμποδίζει στο να βρίσκει πολλές εφαρμογές σε πολλά συστήματα, ακόμα και στα ποιο σύνθετα, αφού ο μόνος περιορισμός του είναι στην φαντασία του σχεδιαστή.

Ανάλογα λοιπόν με το σκοπό της εργασίας που θέλουμε να χρησιμοποιήσουμε το I^2C Bus μπορούν να γίνουν κάποιες προσθήκες στην ήδη υπάρχουσα κατασκευή.

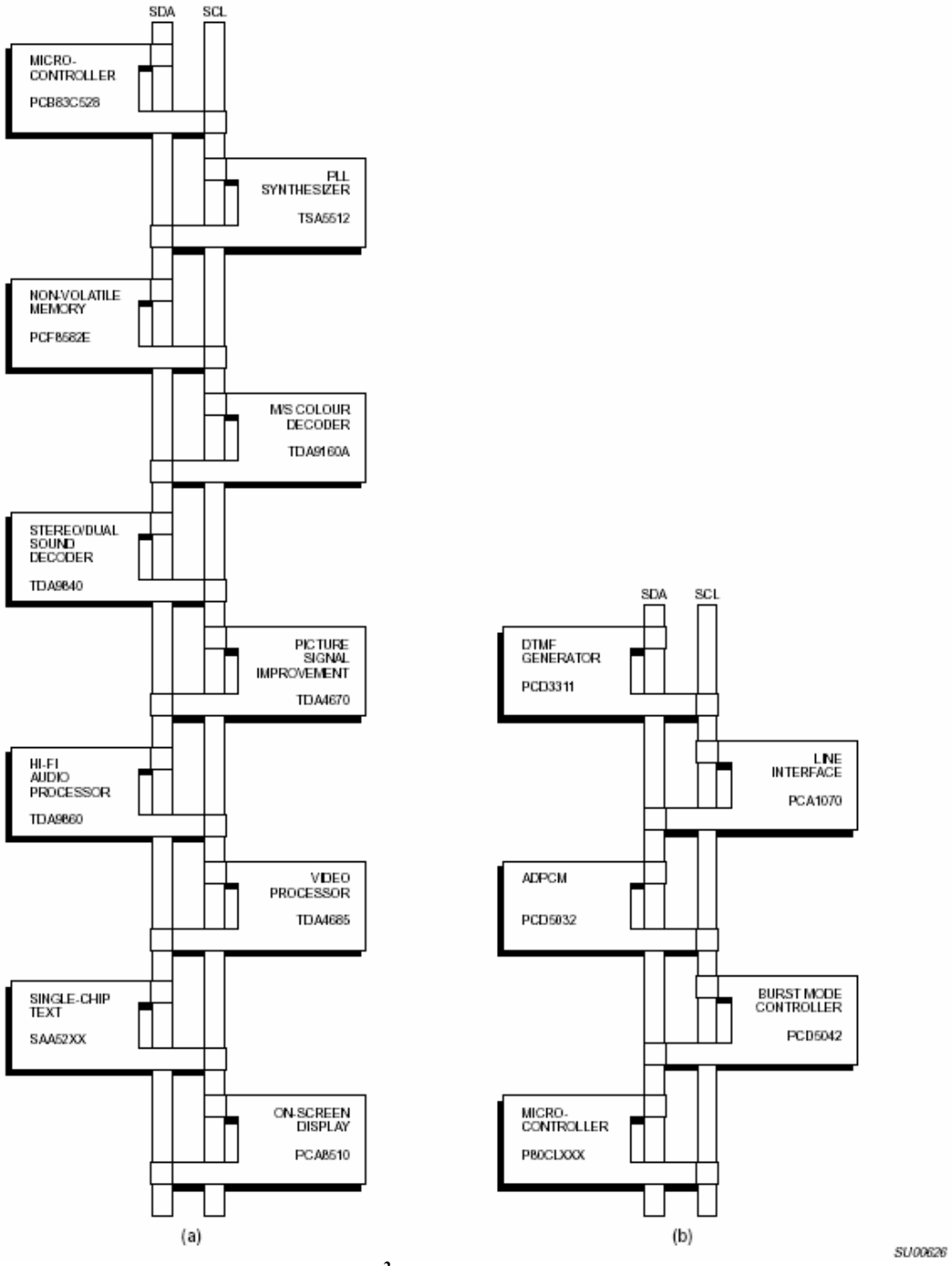
Η προσθήκη σειριακής επικοινωνίας μεταξύ του master και του υπολογιστή μέσω του rs-232 θα ήταν μια πρώτη βελτίωση, Αυτό θα βοηθούσε στην περίπτωση που ο master ήταν συνδεδεμένος με περισσότερους από έναν slave και είχε ως σκοπό την περισυλλογή δεδομένων από τους slaves και επεξεργασία των δεδομένων τους από τον χρήστη. Φυσικά τα υπόλοιπα slave ολοκληρωμένα θα μπορούσαν να είναι μικροελεγκτές, μνήμες RAM, EEPROMs κ.α. Η αφαίρεση ή προσθήκη ολοκληρωμένων είναι ένα από τα πλεονεκτήματα του I^2C πρωτοκόλλου. Στο παράρτημα Γ δίνεται ένα σχηματικό όπου φαίνεται ο master με δυνατότητα σειριακής επικοινωνίας μέσω του rs-232.

Ακόμα με μια απλή αλλαγή στον ήδη υπάρχοντα κώδικα θα μπορούσαμε να είχαμε αύξηση του ρυθμού μετάδοσης των δεδομένων με την λειτουργία του I^2C σε Fast-Mode(400Kbps). Η χρησιμοποίηση εργαλείων όπως ο PCWH C compiler IDE, ο οποίος διαθέτει κάποιες έτοιμες βιβλιοθήκες διευκολύνουν στον προγραμματισμό του PIC.

Το μήκος του διαύλου I^2C μπορεί να φτάσει στην συγκεκριμένη εφαρμογή τα 2-3 μέτρα αλλά με την βοήθεια των bus extenders σε συνδυασμό με μείωση του ρυθμού μετάδοσης των δεδομένων και ανάλογα με τον αριθμό των ολοκληρωμένων που είναι συνδεδεμένα πάνω του μπορεί να φτάσει μέχρι και 100 μέτρα.

Όπως έχουμε πει το I^2C βρίσκει πολλές εφαρμογές στα ολοκληρωμένα κυκλώματα μιας ψηφιακής τηλεόρασης, σε ασύρματες τηλεφωνικές βάσεις, σε συστήματα συναγερμών, σε συστήματα ελέγχου θερμοκρασίας και αλλού.

Ακόμα εφικτή θα ήταν στο σύστημα μας η προσθήκη αισθητήρων, παραδείγματος χάριν αισθητήρων θερμοκρασίας και η συγκομιδή των δεδομένων μέσω του I^2C και του Master microcontroller, ενώ η παρουσίαση θα γίνεται μέσω της rs-232 του Master στην οθόνη του υπολογιστή.



Σχήμα 8.1: Παραδείγματα Χρησιμοποίησης του I^2C Bus

a. Υψηλής ολοκλήρωσης ψηφιακή τηλεόραση

b. DECT ασύρματος σταθμός βάσης τηλεφώνου

9. Επίλογος

Με την πρόοδο της τεχνολογίας οι καινούργιες εφαρμογές που αναπτύσσονται γίνονται ολοένα πιο μικρές και πολύπλοκες με αποτέλεσμα ο έλεγχος τους να δυσκολεύει. Η εύρεση μιας αποδοτικής λύσης μας οδήγησε στην χρησιμοποίηση του I^2C πρωτοκόλλου, ενός πρωτοκόλλου ευρύτατα αναγνωρισμένου από πολλούς κατασκευαστές chip. Πολλά χαρακτηριστικά του είναι ιδιαίτερα ελκυστικά στους σχεδιαστές κυκλωμάτων όπως ότι δεν είναι αναγκαία η χρησιμοποίηση εξωτερικών επαφών αφού είναι ενσωματωμένο σε πολλά chips αλλά και η χαμηλή κατανάλωση, η ανέχεια του στο θόρυβο, και η μεγάλη εμβέλεια σε θερμοκρασιακές αλλαγές, που το κάνουν ιδιαίτερα ελκυστικό ακόμα και στην περίπτωση που πρέπει να ενσωματωθεί σε κάποιο φορητό σύστημα.

Σκοπός της διπλωματικής εργασίας είναι η σχεδίαση και η κατασκευή ενός συστήματος το οποίο να μπορεί με την βοήθεια ενός μικροελεγκτή PIC να μπορεί να συλλέγει πληροφορίες και να τις μεταδίδει σε έναν δεύτερο PIC και από αυτόν στην οθόνη του ηλεκτρονικού υπολογιστή όπου θα μπορούμε να επεξεργαστούμε τις πληροφορίες όπως εμείς θέλουμε.

Το σύστημα μπορεί να αποτελέσει υποσύστημα μιας μεγαλύτερης εφαρμογής ενώ υπάρχουν αρκετές αχρησιμοποίητες πόρτες εισόδου / εξόδου για περαιτέρω ανάπτυξη του συστήματος.

Κατά την ανάπτυξη του συστήματος αποκτήθηκε αρκετή γνώση σε θέματα μικροελεγκτών PIC, όπως προγραμματισμός σε assembly και C++ language, εκμάθηση περιφερειακών μονάδων του μικροελεγκτή κ.α. Τέλος αποκτήθηκε μια πρώτη εμπειρία στην ανάπτυξη ενός ολοκληρωμένου συστήματος.

Με το σύστημα που έχει κατασκευαστεί καθίσταται δυνατή η μετάδοση οποιωνδήποτε δεδομένων επιθυμούμε στην standard mode (100Kbps) ταχύτητα του I^2C Bus.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Using the PICmicro® MSSP Module for Master I2C™ Communications AN735

[2] Using the PICmicro® SSP for Slave I2C™ Communication AN734

[3] Use of the SSP Module in the I2C™ Multi-Master Environment AN578

[4] An I2C™ Network Protocol for Environmental Monitoring AN736

[5] Προγραμματίζοντας τον μικροελεγκτή PIC
Myke Predko

[6] PIC 16F87XA Data Sheet
Microchip

[7] I2C_BUS_SPECIFICATION

[8] C Compiler for MICROCHIP PICMICRO MCUs
http://www.ccsinfo.com/picc.shtml

[9] Hardware and Software Advice
http://www.pic-c.com/links/tips.html

[10] I2C Manual AN10216_01

[11] C Compiler Reference Manual
http://www.ccsinfo.com/download.shtml#CompilerManual

[12] Κ.Ζ. Πεκεμεστζή, καθηγητή ΕΜΠ, “Σεμινάριο Χρήση μικροελεγκτών στη λήψη δεδομένων και στον έλεγχο”

ΠΑΡΑΡΤΗΜΑΤΑ

ΠΑΡΑΡΤΗΜΑ Α

Σχηματικό του συστήματος μετάδοσης δεδομένων

ΠΑΡΑΡΤΗΜΑ Β

Σχηματικά διασύνδεσης του Master και του Slave ξεχωριστά

ΠΑΡΑΡΤΗΜΑ Γ

Σχηματικό του συστήματος μετάδοσης δεδομένων με λειτουργία δύο σειριακών θυρών

ΠΑΡΑΡΤΗΜΑ Δ

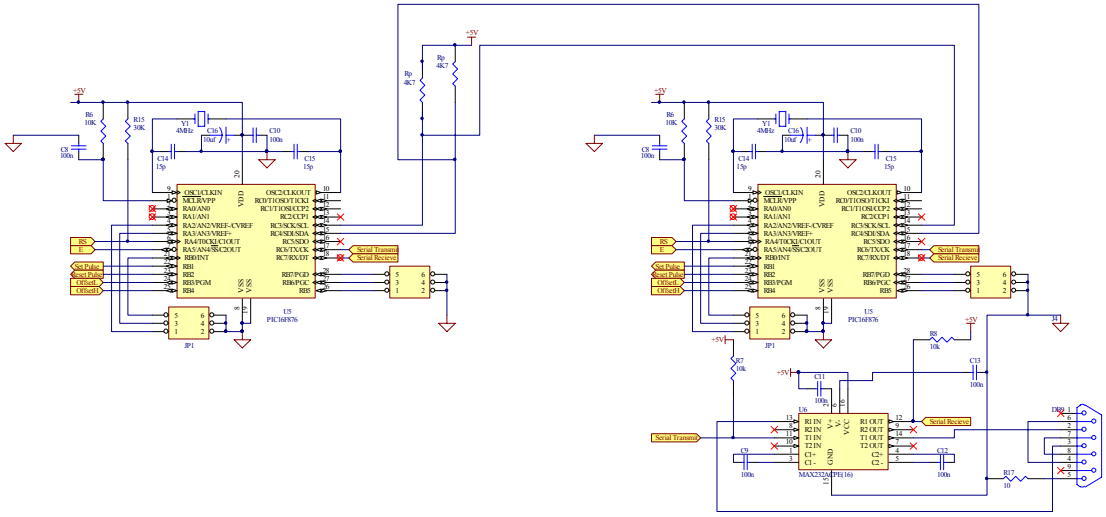
Κώδικας προγράμματος των μικροελεγκτών PIC16F876A

ΠΑΡΑΡΤΗΜΑ Ε

Εμφάνιση των δεδομένων αποστολής μέσω του Serial Input/ Output Monitor

ΠΑΡΑΡΤΗΜΑ Α

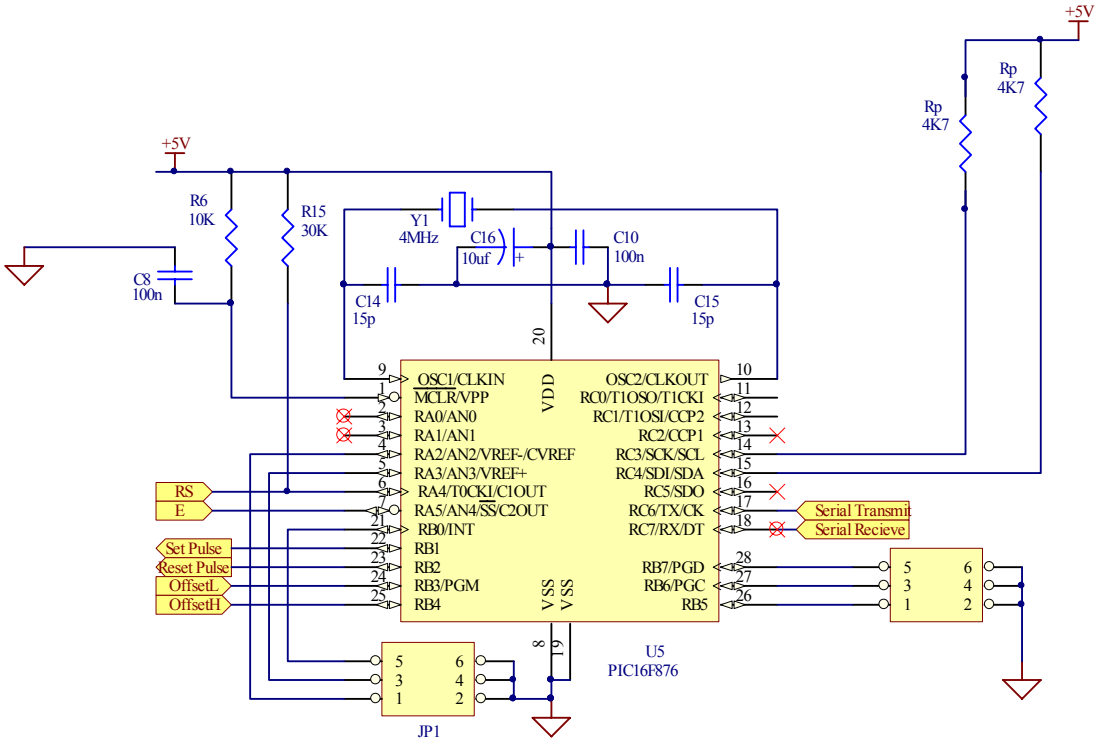
Σχηματικό του συστήματος μετάδοσης δεδομένων



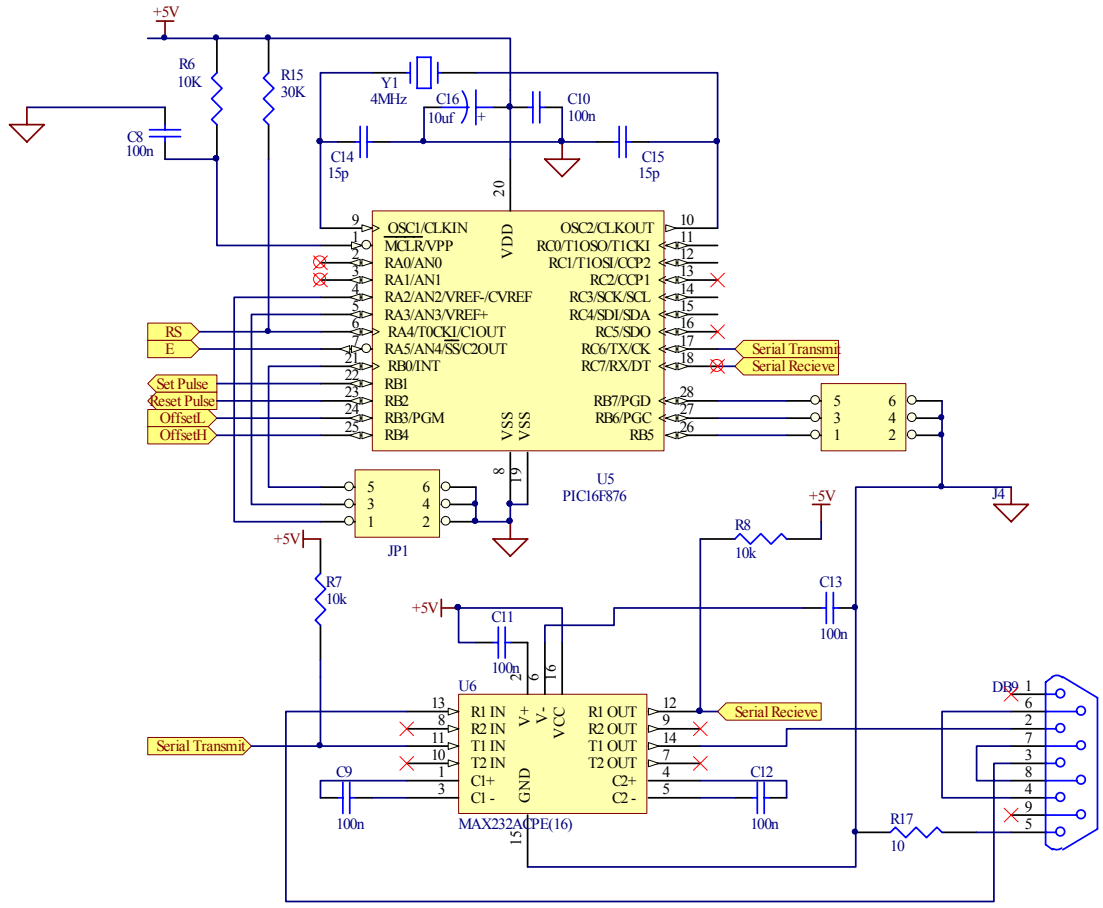
ΠΑΡΑΡΤΗΜΑ Β

Σχηματικά διασύνδεσης του Master και του Slave ξεχωριστά

Master

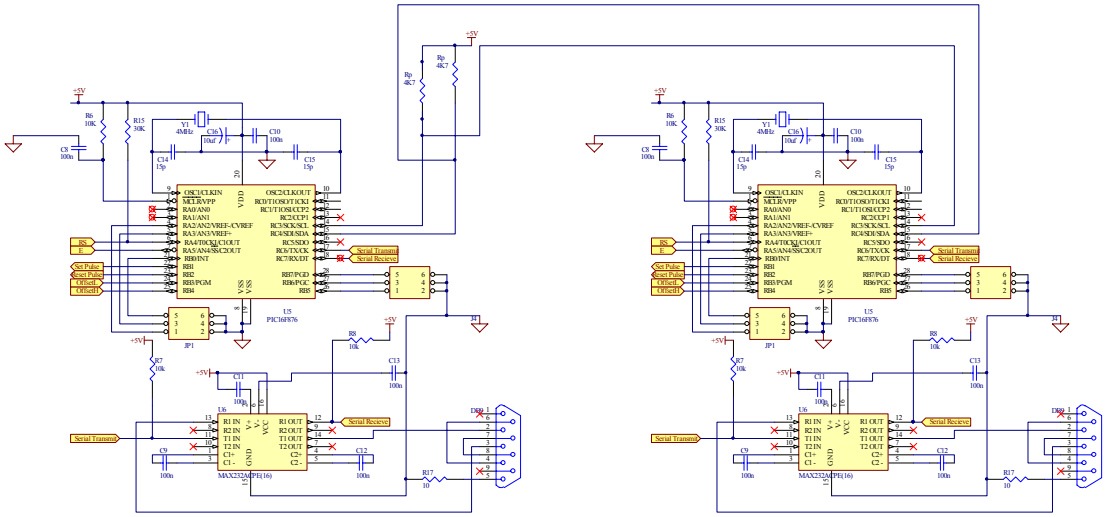


Slave



ΠΑΡΑΡΤΗΜΑ Γ

Σχηματικό του συστήματος μετάδοσης δεδομένων με λειτουργία δύο σειριακών θυρών



ΠΑΡΑΡΤΗΜΑ Δ

Κώδικας προγράμματος των μικροελεγκτών PIC16F876A

ΚΩΔΙΚΑΣ MASTER

```
#include <16F876A.H>

// 10-bit A/D conversion
#device ADC=10
#fuses XT,NOWDT,PUT,NOBROWNOUT,NOLVP,NOPROTECT

#use Delay(Clock=4000000)

#users232(baud=19200,xmit=PIN_C6,rcv=PIN_C7,brgh1ok)

#include <input.c>

#define EEPROM_SDA  PIN_C4
#define EEPROM_SCL  PIN_C3
#define SLAVE_ADDRESS 0x02

#usei2c(master,   sda=EEPROM_SDA,   scl=EEPROM_SCL,   Slow,
FORCE_HW)

void
initI2C()
{
   output_float(EEPROM_SCL);
   output_float(EEPROM_SDA);
}

void
mywrite()
{
   i2c_start();
   delay_ms(1);
   i2c_write(SLAVE_ADDRESS);  /* Device Address */
   delay_ms(1);
   i2c_write(0x4E);           /* DATA  NICK*/
   delay_ms(1);
   i2c_write(0x49);
   delay_ms(1);
   i2c_write(0x43);
   delay_ms(1);
   i2c_write(0x4B);
   delay_ms(1);
   i2c_stop();
```

```
}
void main() {

   initI2C();


   mywrite();
}
```

ΚΩΔΙΚΑΣ SLAVE

```
#include <16F876A.H>


// 10-bit A/D conversion
#device ADC=10
#fuses XT, NOWDT, PUT, NOBROWNOUT, NOLVP, NOPROTECT

#use Delay(Clock=4000000)
#use rs232(baud=19200,xmit=PIN_C6,rcv=PIN_C7,brgh1ok)

unsigned char read_i2c(void);
void i2c_interrupt_handler(void);
void i2c_initialize(void);
void i2c_error(void);
void write_i2c(unsigned char transmit_byte);

#INT_SSP
void ssp_interupt ()
{
    i2c_interrupt_handler();
}


/* 16f87XA bytes */
/* Change it per chip */
#byte PIC_SSPBUF=0x13
#byte PIC_SSPADD=0x93
#byte PIC_SSPSTAT=0x94
#byte PIC_SSPCON1=0x14
#byte PIC_SSPCON2=0x91

/* Bit defines */
#define PIC_SSPSTAT_BIT_SMP     0x80
#define PIC_SSPSTAT_BIT_CKE     0x40
#define PIC_SSPSTAT_BIT_DA      0x20
#define PIC_SSPSTAT_BIT_P       0x10
#define PIC_SSPSTAT_BIT_S       0x08
```

```
#define PIC_SSPSTAT_BIT_RW      0x04
#define PIC_SSPSTAT_BIT_UA      0x02
#define PIC_SSPSTAT_BIT_BF      0x01

#define PIC_SSPCON1_BIT_WCOL    0x80
#define PIC_SSPCON1_BIT_SSPOV   0x40
#define PIC_SSPCON1_BIT_SSPEN   0x20
#define PIC_SSPCON1_BIT_CKP     0x10
#define PIC_SSPCON1_BIT_SSPM3   0x08
#define PIC_SSPCON1_BIT_SSPM2   0x04
#define PIC_SSPCON1_BIT_SSPM1   0x02
#define PIC_SSPCON1_BIT_SSPM0   0x01

#define PIC_SSPCON2_BIT_GCEN    0x80
#define PIC_SSPCON2_BIT_ACKSTAT 0x40
#define PIC_SSPCON2_BIT_ACKDT   0x20
#define PIC_SSPCON2_BIT_ACKEN   0x10
#define PIC_SSPCON2_BIT_RCEN    0x08
#define PIC_SSPCON2_BIT_PEN     0x04
#define PIC_SSPCON2_BIT_RSEN    0x02
#define PIC_SSPCON2_BIT_SEN     0x01


#define RX_BUF_LEN  32
#define NODE_ADDR   0x02     /* I2C address of the slave
node */

unsigned char slave_buffer[RX_BUF_LEN];
int buffer_index;
int comms_error;
int debug_state;


void i2c_initialize(void)
{
   /* Set up SSP module for 7-bit */
   PIC_SSPCON1 = 0x36;   /* 0011 0101 */
   PIC_SSPADD = NODE_ADDR;  /* Set the slave's address */
   PIC_SSPSTAT = 0x00;     /* Clear the SSPSTAT register.
*/
   enable_interrupts(INT_SSP);        /*    Enable    MSSP
interrupts. */
}

void i2c_interrupt_handler(void)
{

    unsigned char i2c_mask = 0x2D;  /* 0010 1101 */
    unsigned char temp_sspstat;
    unsigned char this_byte;
    unsigned char tx_byte;
```

```
    int x;

    /* Mask out the unnecessary bits */
    temp_sspstat = PIC_SSPSTAT & i2c_mask;

    switch (temp_sspstat)
    {
        /* Write operation, last byte was an address,
buffer is full */
        case 0x09:   /* 0000 1001 */
            /* Clear the receive buffer */
            for (x=0; x<RX_BUF_LEN; x++)
            {
               slave_buffer[x] = 0x00;
            }
            buffer_index = 0;  /* Clear the buffer index
*/
            this_byte = read_i2c();    /* Do a read of
PIC_SSPBUF */

            debug_state = 1;
            break;

        /* Write operation, last byte was data, buffer is
full */
        case 0x29:   /* 0010 1001 */
            /* Point to the buffer */
            this_byte = read_i2c();  /* Get the byte from
the SSP */
            slave_buffer[buffer_index] = this_byte;  /*
Put it into the buffer */
            buffer_index++;  /*  Increment  the  buffer
pointer */
            /* Get the current buffer index */
            /* Substract the buffer length */
            /* Has the index exceeded the buffer length?
*/
            if (buffer_index >= RX_BUF_LEN)
            {
               buffer_index = 0; /* Yes, clear the buffer
index. */
            }
            debug_state = 2;
            break;

        /* Read operation; last byte was an address,
buffer is empty */
        case 0x0C:   /* 0000 1100 */
            buffer_index = 0; /* Clear the buffer index
*/
            /* Point to the buffer */
```

```
            tx_byte = slave_buffer[buffer_index]; /* Get
byte from the buffer */
            write_i2c(tx_byte);  /*  Write  the  byte  to
PIC_SSPBUF */
            buffer_index++; /* increment the buffer index
*/
            debug_state = 3;
            break;

        /* Read operation; last byte was data, buffer is
empty */
        case 0x2C:   /* 0010 1100 */
            /* Get the current buffer index */
            /* Subtract the buffer length */
            /* Has the index exceeded the buffer length?
*/
            if (buffer_index >= RX_BUF_LEN)
            {
                buffer_index  =  0;  /*  Yes,  clear  the
buffer index */
            }
            /* Point to the buffer */
            /* Get the byte */
            tx_byte = slave_buffer[buffer_index];
            write_i2c(tx_byte);   /* Write to PIC_SSPBUF
*/
            buffer_index++; /* increment the buffer index
*/
            debug_state = 4;
            break;

        /*  A  NACK  was  received  when  transmitting  data
back from the master. */
        /*  Slave  logic  is  reset  in  this  case.  R_W=0,
D_A=1, and BF=0. */
        /* If we don't stop in this state, then something
is wrong!! */
        case 0x28:   /* 0010 1000 */
            debug_state = 5;
            break;

        /* Something went wrong!! */
        default:
            i2c_error();
            break;
    }
}

void i2c_error(void)
{
    comms_error = 1;
```

```
    printf ("I2C ERROR!\r\n");
}

void write_i2c(unsigned char transmit_byte)
{
    unsigned char write_collision = 1;

    while (PIC_SSPSTAT & PIC_SSPSTAT_BIT_BF) /* Is BF bit
set in PIC_SSPSTAT? */
    {
        /* If yes, then keep waiting */
    }

    while (write_collision)
    {
        /* If not, then do the i2c_write. */
        PIC_SSPCON1 &= ~PIC_SSPCON1_BIT_WCOL;   /* Clear
the WCOL flag */
        PIC_SSPBUF = transmit_byte;

        /* Was there a write collision? */
        if (PIC_SSPCON1 & PIC_SSPCON1_BIT_WCOL)
        {
            /* Yes there was a write collision. */
            write_collision = 1;
        }
        else
        {
            /* NO, there was no write collision. */
            /* The transmission was successful */
            write_collision = 0;
        }
    }
    PIC_SSPCON1 |= PIC_SSPCON1_BIT_CKP;   /* Release the
clock. */
}

/* This function returns the byte in SSPBUF */
unsigned char read_i2c(void)
{
    return PIC_SSPBUF;
}

void main(void)
{
    int x;
    debug_state = 0;
    i2c_initialize();
    enable_interrupts(GLOBAL);
```

```
    while (1)
    {
        if (debug_state)
        {
         printf ("debug state = %d\r\n", debug_state);
         delay_ms(1);
          for (x=0; x<buffer_index; x++)
            {
               printf("data = %c\r\n",slave_buffer[x]);
            }

            debug_state = 0;
        }


    }
}
```

ΠΑΡΑΡΤΗΜΑ Ε

Εμφάνιση των δεδομένων αποστολής μέσω του Serial Input/ Output Monitor