ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΡΗΤΗΣ

Σχολή Τεχνολογικών Εφαρμογών Τμήμα Εφαρμοσμένης Πληροφορικής & Πολυμέσων

Πτυχιακή Εργασία

"CAN (Controller Area Network) δίκτυο με αισθητήρες"

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: **ΜΑΡΚΑΚΗΣ ΦΩΤΙΟΣ**
ΗΜΕΡΟΜΗΝΙΑ: **19/05/2008**

ΕΙΣΗΓΗΤΗΣ: **Δρ . ΠΑΠΑΔΟΥΡΑΚΗΣ ΓΕΩΡΓΙΟΣ**

Ορισμός στυλ: ΠΠ 2: Ελληνικά, Χωρίς ορθογραφικό ή γραμματικό έλεγχο, Εσοχή: Αριστερά: 0 στ., Στηλοθέτες: 368,5 στ., Δεξιά,Οδηγός: ...

Διαγράφηκε: ¶

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Ευχαριστίρια

Θάθελα να ευχαριστήρω τον καθηγητή του Heriot Watt University Dr.J. McLean όπως και τον καθηγητή Δρ. Γεώργιο Παπαδουράκη για την αμέριστη συμπαράσταση τους στο πέρας αυτής της πτυχιακής .

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Διαγράφηκε: ¶
¶

Κεφάλαιο 1ο

Εισαγωγή

Μορφοποιήθηκε: Ελληνικά

1.i.Περίληψη

Σκοπός της συγκεκριμένης πτυχιακής εργασίας είναι η θεωρητική περιγραφή , η σχεδίαση και η υλοποίηση ενός multi-master πρωτοκόλλου το οποίο έχει εξαπλωθεί ραγδαία στις μέρες μας και ονομάζεται CAN (Controller Area Network) protocol . Η σπουδαιότητα του συγκεκριμένου πρωτοκόλλου φαίνεται από τις εφαρμογές στις οποίες έχει χρησιμοποιηθεί όπως σε οχήματα , σε εργοστασιακούς αυτοματισμούς , σε έλεγχο βιομηχανικών μηχανών , σε ιατρικό εξοπλισμό κ.α. . Ειδικά η τελευταία κατηγορία χρειάζεται υψηλή πιστότητα και αποτελεσματικότητα χαρακτηριστικά που συγκλίνουν με αυτά του πρωτοκόλλου.

Διαγράφηκε: , χ

Το εργαλείο που πρόκειται να χρησιμοποιηθεί για την υλοποίηση αυτού του είδους δικτύου είναι μικροελεγκτές ειδικής κατηγορίας που υποστηρίζουν τη συγκεκριμένη εφαρμογή . Οι μικροελεγκτές αυτοί θα σχηματίζουν ένα κατανεμημένο δίκτυο επικοινωνίας θα βρίσκονται δηλαδή πάνω σε έναν κοινό δίαυλο επικοινωνίας (από ένα απλό καλώδιο έως και οπτική ίνα) , πάνω στον οποίο ο κάθε ελεγκτής έχει την ίδια ακριβώς βαρύτητα .

Μέσω της εφαρμογής μου θα προσπαθήσω να υλοποιήσω την διάταξη ενός «κεντρικού συστήματος θέρμανσης » με την χρησιμοποίηση αισθητήρων θερμοκρασίας . Με λίγα λόγια θα υπάρχουν συνολικά τρεις μικροελεγκτές , οι 2 θα είναι συνδεδεμένοι με έναν αισθητήρα και θα είναι τοποθετημένοι σε ανοιχτό και κλειστό χώρο ο καθένας , και θα μεταφέρουν τα μηνύματα τους στον $3^{ο}$ μικροελεγκτή ο οποίος και ανάλογα με το κατώφλι θερμοκρασίας που του έχουμε ορίσει θα ανοίγει ή θα κλείνει το υποτιθέμενο σύστημα .

- Η πλατφόρμα για τον προγραμματισμό των μικροελεγκτών είναι ένα freeware για κάποιες εκδόσεις πρόγραμμα , το MPLAB .
- Οι μικροελεγκτές που θα χρησιμοποιηθούν είναι το μοντέλο PIC18F4585
- Για τον σχεδιασμό του δικτύου θα χρησιμοποιηθούν επίσης 3 πομποδέκτες (1 για κάθε μικροελεγκτή) και 2 ψηφιακοί αισθητήρες θερμότητας .


Μορφοποιήθηκε: Ελληνικά

1.ii.Εισαγωγή

Διαγράφηκε: .v

Μορφοποιήθηκε: Ελληνικά

Στη σύνοψη της πτυχιακής στο **1ο Κεφάλαιο** γίνεται μια σύντομη και γενική περιγραφή της εργασίας , η λειτουργία του κυκλώματος καθώς και ο τρόπος και η λογική που σχεδιάστηκε και υλοποιήθηκε. Πέντε επιπλέον κεφάλαια ακολουθούν καθένα από το οποίο περιγράφει και αναλύει την πτυχιακή από διαφορετική σκοπιά.

Στο **2οΚεφάλαιο** γίνεται αναφορά στα κατανεμημένα δίκτυα , όπου ανήκουν και τα CAN (Controller Area Networks) , όπου γίνεται σύγκριση με τα δίκτυα "αντίπαλης" τοπολογίας με κάποιον κεντρικό κόμβο .

Διαγράφηκε: ανήκουν

Διαγράφηκε: και

Το **3ο Κεφάλαιο** αποτελεί μια εισαγωγή στο CAN πρωτόκολλο αναλύοντας επίσης και τους λόγους που είναι τόσο ευρέα διαδεδομένο καθώς επίσης και τις εφαρμογές σε όλους τους τομείς που χρησιμοποιείται. Επίσης σε αυτό το κεφάλαιο γίνεται εκτενής ανάλυση του πρωτοκόλλου καθώς και του μηχανισμού λειτουργίας του.

Το **4οΚεφάλαιο** παρουσιάζει όλες τις δυνατές υλοποιήσεις χρησιμοποιώντας το CAN πρωτόκολλο , ανάλυση των μικροελεγκτών που χρησιμοποιήθηκαν σε αυτήν την πτυχιακή καθώς και τα λειτουργικά τους χαρακτηριστικά .

Στο **5οΚεφάλαιο** γίνεται μνεία σε κάθε δυνατή CAN υλοποίηση , και οι λόγοι για τους οποίους χρησιμοποιήθηκε η συγκεκριμένη αρχιτεκτονική. Επίσης γίνεται ανάλυση κάθε στοιχείου του κυκλώματος καθώς και του κώδικα .

Διαγράφηκε: η συγκεκριμένη αρχιτεκτονική

Στο τελευταίο κεφάλαιο (**6οΚεφάλαιο**) γίνεται περίληψη της πτυχιακής , αθροίζοντας όλα τα συμπεράσματα και όλες τις δυνατές μελλοντικές τεχνικές που αφορούν το θέμα .

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

1iii.Πίνακας κεφαλαίων

Μορφοποιήθηκε: Αγγλικά (Η.Β.)

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Κεφάλαιο 2

Background

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Μορφοποιήθηκε: Στοιχισμένο στο κέντρο

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

2.i. Αναφορά στα κατανεμημένα δίκτυα

Το είδος του δικτύου που χαρακτηρίζεται από την κατασκευαστική του απλότητα αλλά την ίδια στιγμή και από την δραστικότητα και αποτελεσματικότητα που έχει , είναι ένα κατανεμημένο δίκτυο. Το κύριο χαρακτηριστικό αυτού του είδους του δικτύου είναι ότι αποτελείται από πολλούς κόμβους καθώς και πολλές συσκευές αποθήκευσης δεδομένων οι οποίες όλες συνδέονται μεταξύ τους άμεσα ή έμμεσα μέσω του ιδίου μέσου.δικτύου .Το παραπάνω γεγονός κάνει τα κατανεμημένα συστήματα περισσότερα δυναμικά και αποτελεσματικά σε σχέση με το είδος των δικτύων που λειτουργούν στηρίζοντας σε έναν κεντρικό κόμβο.

Πρώτα απ΄όλα η μεταφορά μηνυμάτων γίνεται περισσότερο αξιόπιστα , καθώς κάθε διαδικασίες μπορούν να επαναληφθούν πολλές φορές . Πιο συγκεκριμένα όταν ένας κόμβος αποτύχει στην αποστολή ενός μηνύματος , αυτόματα ένας άλλος κόμβος παίρνει προτεραιότητα έτσι ώστε να ολοκληρώσει την διαδικασία .Κοινώς όλα τα δεδομένα μπορούν να μεταφέρονται και να αποθηκεύονται σε πολλούς κόμβους έτσι ώστε η αποτυχία ενός κόμβου να μην επηρεάσει τα δεδομένα που πρέπει να αποσταλούν .Επίσης ένα τέτοιο σύστημα μπορεί να εκτελέσει διεργασίες αρκετά γρήγορα , για τον λόγο ότι υποστηρίζεται ο παραλληλισμός .Αυτά τα δύο χαρακτηριστικά γνωρίσματα , η ανοχή στα λάθη και ο παραλληλισμός δίνουν στο κατανεμημένο σύστημα την δυναμική του και κάτω από αυτές τις ιδιότητες σχεδιάζονται όλα τα κατανεμημένα συστήματα στις μέρες μας . [1]

Μα ποια είναι τα χαρακτηριστικά που κάνουν τα κατανεμημένα συστήματα αναγκαία σε σχέση με τα αντίστοιχα που χρησιμοποιούν έναν κεντρικό κόμβο ?

- *Συναρμολογησιμότητα* : Σ΄ ένα κατανεμημένο σύστημα οι διασυνδέσεις μεταξύ των κόμβων πρέπει να είναι πιο προσεχτικά σχεδιασμένες .
- *Επεκτασιμότητα* :Τα κατανεμημένα συστήματα έχουν αυξημένες δυνατότητες . Ως αποτέλεσμα η ικανότητα αποθήκευσης δεδομένων καθώς και η επεξεργαστική ισχύς μπορεί εύκολα να διευρυνθεί .
- *Αυξομείωση*: Ιδανικά , τα κατανεμημένα συστήματα δεν έχουν κάποιο κύριο κόμβο , όπως συμβαίνει στην "αντίπαλη κατηγορία" δικτύων , έτσι δεν υπάρχει και περιορισμός στο μέγεθος του δικτύου .Πάντως κάτω από πραγματικές συνθήκες , υπάρχει κάποιος περιορισμός ο οποίος συνήθως εξαρτάται από τεχνικά χαρακτηριστικά των κόμβων .
- Αξιοπιστία : Το βασικό γνώρισμα της αξιοπιστίας στα δίκτυα είναι όχι μόνο τα δεδομένα να μεταφέρονται , αλλά αυτή η μεταφορά να γίνεται με όσο το δυνατόν λιγότερα λάθη . Επίσης είναι επιθυμητό οι αλγόριθμοι που θα χρησιμοποιούνται να έχουν την δυνατότητα ανάκτησης δεδομένων σε περίπτωση κάποιας αποτυχίας .

Διαγράφηκε: ς

Διαγράφηκε: και

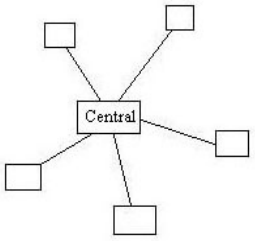
2.ii. Σύγκριση μεταξύ κατανεμημένων δικτύων που χρησιμοποιούν ένα κεντρικό κόμβο

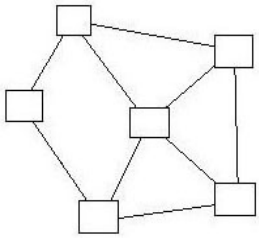
Η κύρια διαφορά μεταξύ ενός δικτύου που χρησιμοποιεί έναν κεντρικό κόμβο , που είναι συνήθως ένα Ethernet δίκτυο , και ενός κατανεμημένου δικτύου , είναι ότι στο δεύτερο μπορούν να τοποθετηθούν και να διαχειριστούν από τους περισσότερους κόμβους του συστήματος σε αντίθεση με το πρώτο που το βάρος συγκεντρώνεται μόνο στον κεντρικό κόμβο.

Επιπλέον αναβαθμίσεις στο λογισμικό του συστήματος δεν συναντάμε συχνά στα κατανεμημένα δίκτυα καθώς δεν είναι τόσο εύκολο να υλοποιηθούν και τέλος περισσότερες διαχειριστικές , λειτουργικές και προφυλάξεις ασφαλείας πρέπει να παίρνονται στα δίκτυα αυτά λόγω της

Διαγράφηκε: τις

ιδιότυπης αρχιτεκτονικής τους





1.1 a . Centralized Network (Usually for Ethernet systems)
A central node and many peripheral components
connected to the central

b . Distributed Network
No central node , many components are
connected in differeant architectures .

Στην μικρή παραπάνω περίληψη , έγινε ανάλυση μερικών από τον πιο σημαντικών χαρακτηριστικών των κατανεμημένων δικτύων . Για να μπορέσουμε όμως να φτάσουμε στο επιθυμητό αποτέλεσμα και να λειτουργήσει το δίκτυο αυτό ιδανικά , πρέπει να ξεπεραστούν όλα τα προβλήματα στο σχεδιασμό και στην υλοποίηση , που τα περισσότερα οφείλονται στην πολυπλοκότητα του δικτύου .Τα προβλήματα αυτά δημιουργούνται εξαιτίας των σύνθετων διασυνδέσεων και αλληλεπιδράσεων μεταξύ των κόμβων .Ακόμα και αν υπάρχει σωστός σχεδιασμός στο σύστημα είναι πιθανό να υπάρξουν νέα προβλήματα. Η πλειοψηφία αυτών των προβλημάτων δεν είναι δυνατό να προβλεφτει και μάλιστα στις περισσότερες περιπτώσεις απέχει πολύ από αυτό που σκεφτήκαμε ότι μπορεί να συμβεί.

Διαγράφηκε: και
Διαγράφηκε: α
Διαγράφηκε: νέα προβλήματα
Διαγράφηκε: θ
Διαγράφηκε: ούν
Διαγράφηκε: ουν

2.iii. Κατανεμημένα συστήματα – Η ιδανική επιλογή

Διαγράφηκε: κ

Η λύση η οποία καθιερώθηκε στα σύγχρονα προβλήματα ελέγχου των δικτύων , είναι ένα είδος κατανεμημένου δικτύου το οποίο στηρίζεται σε έναν κεντρικό δίαυλο . Οι περιφερειακές συσκευές (αισθητήρες , μηχανισμοί κινήσεως κ.α.) μπορούν να συνδεθούν μεταξύ τους δημιουργώντας ένα υποδίκτυο και τα υποδίκτυα αυτά μπορούν πάλι να συνδεθούν μεταξύ τους σε ένα κεντρικό κανάλι διαύλου . Ως αποτέλεσμα το τελικό δίκτυο μπορεί να γίνει λιγότερο σύνθετο καθώς λιγότερες καλωδιώσεις χρησιμοποιούνται και συνεπώς οι αποτυχίες και τα λάθη στο σύστημα είναι πολύ λιγότερα .

Για να επωφεληθούμε όλα τα στοιχεία του συστήματος , δύο στοιχεία πρέπει να προσεχθούν ιδιαίτερα . Το πρώτο έχει να κάνει με τις παραμέτρους που ελέγχουν το σύστημα και είναι η διευθυντοδοτηση και

ο σωστός χρονισμός . Όλες οι διευθύνσεις πρέπει να καθορίζονται από έναν ειδικό δείκτη-επικεφαλίδα έτσι ώστε να ορίζεται η πηγή και ο προορισμός των δεδομένων . Επιπλέον εξαιτίας του συγκεκριμένου τρόπου μηχανισμού αποστολής μηνυμάτων πρέπει αυτά να μεταδίδονται σε συγκεκριμένα χρονικά διαστήματα .

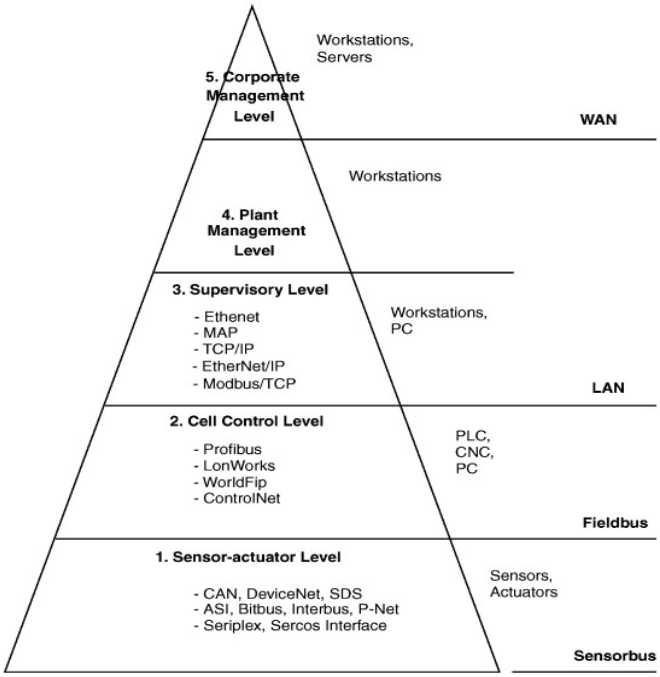
Μια πιο ειδική αναφορά , συμπεριλαμβάνοντας και κάποιο παράδειγμα , για να μπορέσουμε να διακρίνουμε τα χαμηλού επιπέδου στοιχεία ενός δικτύου (αισθητήρες θερμοκρασίας , ταχύτητας , υγρασίας κ.α.) και όλες τις παραμέτρους που χρειάζονται για να έχουμε αποτελεσματική λειτουργία του δικτύου είναι η παρακάτω . Στην περίπτωση ενός κατανεμημένου δικτύου που είναι αρκετά ευρύ , ο αριθμός τον συσκευών είναι επίσης μεγάλος , έτσι η φιλοδοξία για ένα σύστημα με καλή απόδοση , απαιτεί μια πιο εκτενή και πλήρη υλοποίηση του δικτύου .

Διαγράφηκε: ου

Στο σχήμα παρακάτω απεικονίζεται μια τυπική ιεραρχία ενός δικτυακού μοντέλου . Αποτελείται από πέντε διαφορετικά επίπεδα , καθένα από τα οποία έχει την δική του χρησιμότητα και συνεισφορά στο δίκτυο . Με μια γρήγορη ματιά μπορούμε να διακρίνουμε τα διαφορετικά πρωτόκολλα , την χρήση του δικτύου , την τοπολογία κ.α. .

Αναλύοντας το χαμηλότερο επίπεδο το οποίο ονομάζεται (CVD – Continuous – Variable/Device) παρατηρούμε ότι συσκευές όπως αισθητήρες , μηχανισμοί κινήσεως και ελεγκτές δικτύου ανήκουν σε αυτό το επίπεδο . Έτσι πρωταρχικές πληροφορίες όπως θερμοκρασία , υγρασία

, ταχύτητα ανταλλάσσονται μεταξύ αυτών των συσκευών .



1.2

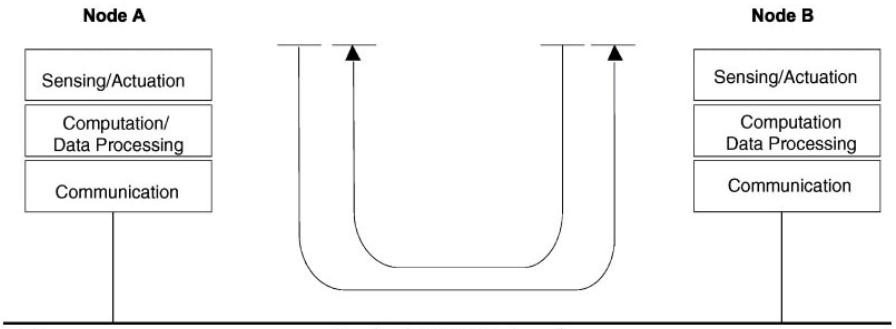
Τα ειδικά χαρακτηριστικά από τις πληροφορίες αυτές , οι οποίες ανταλλάσσονται μέσω των μηνυμάτων , μπορεί να είναι πραγματικού χρόνου ή ανά περιόδους . Το μέγεθος των πακέτων είναι σε γενικές γραμμές μικρό , αλλά η συχνότητα είναι σχετικά μεγάλη .Αυτό συμβαίνει γιατί αυτό το είδος τις πληροφορίας αλλάζει σε πολύ συχνά χρονικά διαστήματα και το σύστημα πρέπει να είναι συνεχώς ενημερωμένο . [2]

Μορφοποιήθηκε: Ελληνικά

Δυστυχώς επειδή είναι απίθανο να συναντήσουμε ένα ιδανικό δίκτυο χωρίς να υπάρχουν καθυστερήσεις , η απώλεια πακέτων είναι πολύ πιθανή .Όλοι οι ειδικοί που ασχολούνται με τα δίκτυα προσπαθούν να περιορίσουν αυτά τα πεδία έτσι ώστε να βελτιωθεί η απόδοση διατηρώντας βέβαια την σταθερότητα του δικτύου .

Ένα τυπικό διάγραμμα για ένα CVD δίκτυο φαίνεται στο σχήμα παρακάτω [σχ.3] . Αποτελείται από 3 υποστρώματα καθένα έχει διαφορετική συνεισφορά και διαφορετικές λειτουργίες .Αρχικά οι αναλογικές πληροφορίες από τους αισθητήρες προωθούνται στα υπολογιστικά/δεδομένα επεξεργαστικά επίπεδα όπου κανονικά

κωδικοποιούνται .



1.3 Typical CVD Network

Σε αυτήν την κατηγορία ανήκουν τα Controller Area Network (CAN) , που είναι και το θέμα αυτής της πτυχιακής .

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Κεφάλαιο 3

Διαγράφηκε: ¶
¶

Εισαγωγή στα Controller Area Networks

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

3.i. Η ιστορία των CAN

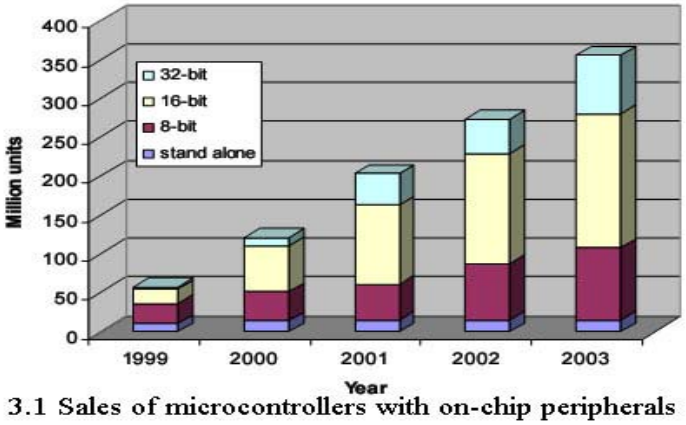
Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Η ιστορία των Controller Area Network αρχίζει στα μέσα της δεκαετία του 80 στην Γερμανία από τον Robert Bosch , όταν η Mercedes του ανάθεσε μια επαγγελματική εργασία . Το κίνητρο ήταν να αυξήσει την απόδοση στον τομέα του ελέγχου στα εντός οχημάτων δίκτυα , με το να μειώσει το μήκος των δικτύων από σημείο σε σημείο και να τα αντικαταστήσει με κάτι λιγότερο σύνθετο και περισσότερο αποτελεσματικό .Καθώς η τεχνολογία και ο ανταγωνισμός μεταξύ των βιομηχανιών αυτοκινήτων συνεχώς αυξάνεται , όλο και περισσότερες ηλεκτρονικές συσκευές που έχουν να κάνουν με την ασφάλεια , την αξιοπιστία και την απόδοση των οχημάτων εισέρχονται στην αγορά .

Ακολουθώντας την Mercedes , άλλες βιομηχανίες αυτοκινήτων όπως είναι η Volvo , η Saab, η BMW , η Volkswagen , η Ford, η Renault, η Fiat κ.α. υιοθέτησαν την τεχνολογία των Controller Area Networks . Ο αριθμός των ηλεκτρονικών δικτυακών μονάδων ελέγχου (Electronic Control Circuit Units , ECUs) σε Mercedes , Audi , BMW και VW ήταν γύρω στα 5 στις αρχές τις δεκαετίας του 90 και έγιναν περίπου 40 στις αρχές του νέου αιώνα . [3]

Ο Robert Bosch κατασκεύασε ένα πρωτοποριακό σειριακό δίαυλο για να ελέγχει μέσω δικτύου συσκευές υψηλού ελέγχου , και ταυτοχρόνως προσπάθησε να περιορίσει τα λάθη κατά την αποστολή δεδομένων , το βάρος και το μήκος των καλωδίων καθώς βέβαια και το κόστος .Υπολογίστηκε ότι περίπου το 10% του βάρους των καλωδίων μειώθηκε με τη χρησιμοποίηση της νέας αυτής τεχνικής .Από το 1993 το CAN θεωρείται standard για το εσωτερικό των οχημάτων και όχι μόνο , και χρησιμοποιήθηκε από τους περισσότερους κατασκευαστές αυτοκινήτων . Η χρήση του σιγά σιγά επεκτάθηκε και σε άλλες εφαρμογές και σε άλλα επίπεδα . Το ISO 11898 υιοθετήθηκε από αυτοκινούμενες εφαρμογές υψηλής ταχύτητας , ενώ το ISO 11519 για τις αντίστοιχες εφαρμογές χαμηλής ταχύτητας .Από το 1994 περισσότερα ISO υιοθετήθηκαν για άλλες βιομηχανικές εφαρμογές



3.1 Sales of microcontrollers with on-chip peripherals

.

Το παραπάνω σχήμα[3.1] *απεικονίζει την τάση πωλήσεως CAN περιφερειακών συστημάτων κατά την περίοδο 1993 2003.Στο επόμενο κεφάλαιο γίνεται αναφορά στις αγορές που οι CAN εφαρμογές έχουν εισχωρήσει* . [3]

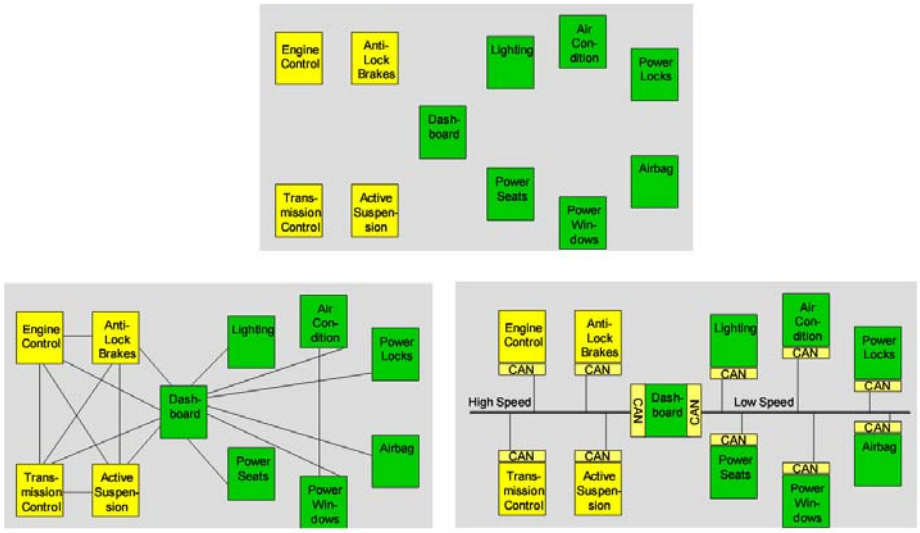
3.ii. Τα οφέλη του CAN

Διαγράφηκε: i

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Τα ωφελεί του CAN μπορούν να συναθροιστούν στο παρακάτω σχήμα [3.2] .



3.2 a Centralize Network b CAN

Εκτός του γεγονότος ότι όλο και λιγότερα καλώδια χρησιμοποιούνται (το οποίο σημαίνει και λιγότερο κόστος) και μειωμένη πολυπλοκότητα του δικτύου , ο έλεγχος του συστήματος γίνεται πιο αποτελεσματικός καθώς κάθε συσκευή διαθέτει τον δικό της ελεγκτή .Στην περίπτωση μάλιστα που θέλουμε

να προσθέσουμε ή να αφαιρέσουμε μια συσκευή από το σύστημα η όλη διαδικασία είναι σχετικά εύκολη καθώς δεν απαιτούνται μεγάλες αλλαγές στο λογισμικό του συστήματος .

Επιπλέον από την υλοποίηση του λογισμικού , δίδεται η δυνατότητα όλα τα μηνύματα να παραλαμβάνονται από όλους τους χρήστες ή με ειδικά φίλτρα ο χρήστης να τα απορρίπτει . Τέλος πρέπει να σημειώσουμε ότι το κάθε μήνυμα έχει την δική του προτεραιότητα , συνεπώς αν δύο μηνύματα εκπεμφθούν ταυτοχρόνως , το μήνυμα με την μεγαλύτερη προτεραιότητα θα παραληφθεί ενώ μετέπειτα θα σταλεί και το άλλο μήνυμα .

Διαγράφηκε: εξαρτώντας και

3.iii. Εφαρμογές του CAN

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Όπως έχουμε προαναφέρει η πρώτη CAN εφαρμογή εφαρμόστηκε για τον έλεγχο των ηλεκτρονικών αυτοκινήτων περίπου πριν δυο δεκαετίες . Μέχρι τώρα τα CAN εξαιτίας της μεγάλης επιτυχίας τους , έχουν υιοθετηθεί και σε άλλες βιομηχανικές εφαρμογές .

CAN βασισμένα σε εφαρμογές αυτοκινήτων

Σε αυτό το είδος χρησιμοποιούνται δύο διαφορετικά δίκτυα βασιζόμενα στο είδος της εφαρμογής . Τυπικές υλοποιήσεις των powertrain δίκτυων ,τα οποία και υποστηρίζουν υψηλούς ρυθμούς μετάδοσης δεδομένων (μέχρι 1Mbit/sec) , είναι κινητήρες με χαμηλότερα επίπεδα θορύβου , μηχανές που καταναλώνουν λιγότερα καύσιμα , καθαρότερα καυσαέρια και βελτιωμένες δυναμικές οδήγησης (εφαρμογές όπου η επικοινωνία είναι υψίστης σημασίας , και τα δεδομένα πρέπει να ανανεώνονται σε πολύ συχνά χρονικά διαστήματα . Από την άλλη , υπάρχουν και τα body-control δίκτυα που δεν έχουν να κάνουν με την απόδοση ή την ασφάλεια του δικτύου αλλά θεωρούνται ως ένα έξτρα κομφόρ . Τυπικά παραδείγματα εφαρμογών είναι τα ηλεκτρικά παράθυρα και οι καθρέφτες αυτοκινήτων , ελεγκτές καθισμάτων και πόρτας , κλιματισμός κ.α. . Η επόμενη γενιά CAN που θα χρησιμοποιηθεί θα είναι ένα ημιαυτόνομο σύστημα παρκαρίσματος αυτοκινήτων , χρησιμοποιώντας αισθητήρες που υπολογίζουν το ελεύθερο διάστημα Μετά ο οδηγός ενημερώνεται ακουστικώς για το πως θα κατευθυνθεί .

Διαγράφηκε: δύο διαφορετικά δίκτυα

Διαγράφηκε: Τ

Διαγράφηκε: α

Διαγράφηκε: α

Διαγράφηκε: τυπικές υλοποιήσεις

Διαγράφηκε: ό

CAN βασισμένα στην ναυτιλία

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Χρησιμοποιούνται σε οποιοδήποτε είδους πλοίου όπως κοντέινερ , επιβατικά , πλοία της γραμμής , φορτηγά πλοία και βάρκες ως ενσωματωμένα συστήματα τα οποία συνδέονται σε άλλα μεγαλύτερα συστήματα .

Χρησιμοποιούνται κυρίως για λόγους ασφαλείας είτε για τον γενικό έλεγχο του πλοίου . Μια αρκετά ενδιαφέρουσα χρήση είναι ένα CANopen σύστημα με ανοχή στα λάθη . Η ιδέα βασίζεται στον χωρισμό του συστήματος σε μικρότερα υποσυστήματα , καθένα από τα οποία εκτελεί διαφορετική διαδικασία .

CAN σε αεροσκάφη και αεροδιαστημικά ηλεκτρονικά

Τα CAN αποτελούν την ραχοκοκαλιά των δικτύων στα αεροσκάφη για αισθητήρες σταθερής πτήσης , κατευθυντικά συστήματα και ερευνητικά PC με οθόνες οδήγησης που είναι εγκατεστημένες στο πιλοτήριο . Αυτά τα CAN δίκτυα έχουν την δυνατότητα να αναλύουν τα δεδομένα της πτήσης ή με την βοήθεια εγκαταστάσεων ήχου/εικόνας να δίνουν αναλυτικές πληροφορίες στο πλήρωμα που στέλνονται από το σύστημα του πιλοτηρίου . Τα CAN επίσης χρησιμοποιούνται σε αεροδιαστημικές εφαρμογές και πιο συγκεκριμένα σε συστήματα ελέγχου όπως έλεγχο καυσίμων και μηχανισμούς κίνησης αντλιών

CAN σε ανυψωτικά συστήματα και ασανσέρ

Τα ανυψωτικά συστήματα και ασανσέρ χρησιμοποιούσαν ανέκαθεν CAN συστήματα . Όλες οι συσκευές που βρίσκουν εφαρμογή στα παραπάνω όπως πλαίσια , ελεγκτές , πόρτες , οδηγοί , φωτισμός κ.α. συνδέονται το καθένα με το άλλο μέσω των CAN . Υπάρχουν επίσης ανελκυστήρες που βασίζονται εξ'ολοκλήρου στα συστήματα CAN .

CAN σε αυτοματισμούς κτιρίων

Σε αυτοματισμούς κτιρίων τα CAN χρησιμοποιούνται για να ελέγξουν τη θερμοκρασία , την ψύξη , το φωτισμό , τον αερισμό κα τις πόρτες . Περαιτέρω χρήσεις είναι συστήματα συναγερμού , συσκευές ψεκασμού κήπων , κρυφά συστήματα ελέγχου όπως επίσης και μηχανισμοί studio με συμπεριλαμβανομένο έλεγχο ήχου και εικόνας .Στην Ελβετία υπάρχουν ήδη αρκετά ολοκληρωμένα δωμάτια ελέγχου σε κτίρια βασισμένα στα CAN .

CANopen σε συσκευές ιατρικού εξοπλισμού

Τα CAN έχουν εισέλθει σε πολλές ιατρικές εφαρμογές . Χαρακτηριστικά παραδείγματα είναι συσκευές ακτινογραφίας , γεννήτριες

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Διαγράφηκε: η

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Διαγράφηκε:

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Διαγράφηκε: ε

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Ελληνικά

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Διαγράφηκε: ούς

Διαγράφηκε: υ

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Ελληνικά

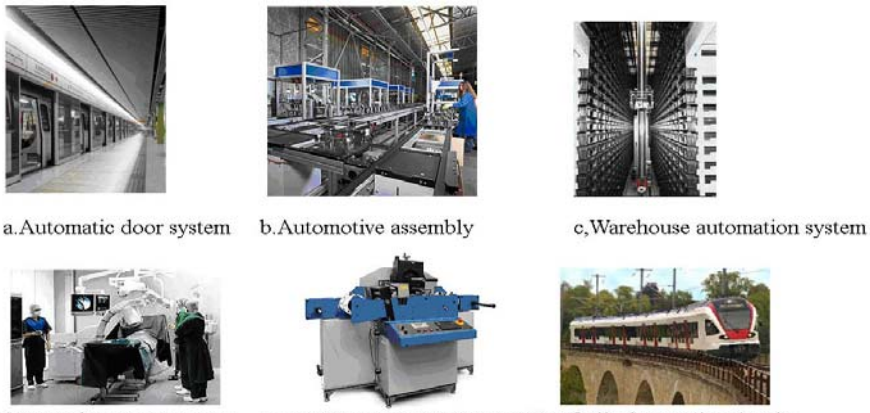
Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

ακτινογραφίας , φάκελοι ασθενών κ.α. . Επίσης ολόκληρα δωμάτια όπως χειρουργεία έχουν υιοθετήσει αυτήν την τεχνολογία .

Άλλες CAN εφαρμογές

Μερικά άλλα παραδείγματα είναι σε εργοστασιακούς αυτοματισμούς , σε βιομηχανικό έλεγχο μηχανών και σε μη-βιομηχανικές εφαρμογές όπως εξοπλισμό εργαστηρίων , αθλητικές κάμερες , αυτόματες πόρτες κ.α. [4] . Το παρακάτω σχήμα [3.3] απεικονίζει μερικές εφαρμογές CAN που χρησιμοποιούνται ευρέος .

Μορφοποιήθηκε: Ελληνικά



a.Automatic door system b.Automotive assembly c,Warehouse automation system

d.Operating surgery room e.special-purpose printing machine f.Flirt interregional train

3.3 [4]

3.iv. Βασικές αρχές των CAN

Τα CAN είναι ένας multicast γραμμικός επικοινωνιακός δίαυλος με πολλούς επεξεργαστές οι οποίοι επικοινωνούν αποτελεσματικά μεταξύ τους . Δεν υπάρχει καμία διάκριση μεταξύ ανάμεσα σε slave και master συσκευές , καθώς όλες οι συσκευές θεωρούνται ισότιμες . Όλοι οι επεξεργαστές δεν διαθέτουν κάποια συγκεκριμένη διεύθυνση αλλά στέλνουν συγκεκριμένα στοιχεία αναγνώρισης με την ανταλλαγή των μηνυμάτων . Όλοι αυτοί οι κωδικοί αναγνώρισης πρέπει να είναι μοναδικοί σε όλο το δίκτυο .Επιπλέον , κάθε μήνυμα περιέχει εκτός από τον κώδικα αναγνώρισης και τα δεδομένα , την προτεραιότητα που έχει κάθε μήνυμα .Αν κάποιος κόμβος θέλει να εκπέμψει κάποιο μήνυμα πρέπει αρχικά να ελέγξει αν ο δίαυλος είναι ελεύθερος και μετά μπορεί να το εκπέμψει . Σύμφωνα με το πρωτόκολλο ποτέ δεν διακόπτεται μια μετάδοση που είναι σε εξέλιξη , αλλά ορίζονται προτεραιότητες στα μηνύματα έτσι ώστε να αποφευχθούν συγκρούσεις και να σιγουρευτούμε ότι τα υψηλής προτεραιότητας μηνύματα παραδίδονται πρώτα

και περιλαμβάνουν επίσης και ελεγκτή λαθών έτσι ώστε να κάνουν την διακίνηση υψηλής αξιοπιστίας . [5]

Η φιλοσοφία του CAN είναι να στέλνει μικρά μηνύματα πολύ συχνά και αποτελεσματικά . Για παράδειγμα τα CAN συχνά στέλνουν μηνύματα σε κάποια συγκεκριμένα γεγονότα όπως αναφορά θερμοκρασίας , on-off σήματα κ.α. . Γι'αυτό το λόγο κάθε μήνυμα είναι μέγεθος μέχρι 8 byte . Ένα CAN σύστημα είναι ικανό να εκπέμψει μέχρι 7600 8-byte μηνύματα ή 18000 αναφορές ανά δευτερόλεπτο . [6]

Ο μέγιστος αριθμός των κόμβων δεν περιορίζεται από το CAN πρωτόκολλο (θεωρητικά υποστηρίζονται μέχρι 2032 κόμβοι) αλλά λόγω του περιορισμού του υλικού υποστηρίζονται μόνο 110 (Στην υλοποίηση αυτής της πτυχιακής εξαιτίας του περιορισμού του υλικού έχει χρησιμοποιηθεί πολύ μικρός αριθμός συσκευών) . [7]

Διαγράφηκε: υποστηρίζονται

Διαγράφηκε: αλλά και για .....

Ένας τυπικός CAN δίαυλος αποτελείται από 2 σύρματα , half duplex , υψηλής ταχύτητας σύστημα που σε ιδανικές συνθήκες έχει throughput μέχρι 1Mbit/sec. Το πιο δημοφιλές μέσο που χρησιμοποιείται σε δίαυλο CAN είναι δύο στροβιλισμένα θωρακισμένα σύρματα χαλκού αλλά μπορεί επίσης να είναι και ένα ομοαξονικό καλώδιο ή οπτική ίνα ή ασύρματο μέσο ή κάτι άλλο . Ένα τρίτο καλώδιο είναι πιθανό να χρησιμοποιηθεί για πιο αξιόπιστη λειτουργία . Αυτό χρησιμοποιείται σαν μια επιπλέον γείωση και είναι απαραίτητη για μακρύς διαύλους . Αντιθέτως στα ενσωματωμένα συστήματα CAN ένα τρίτο καλώδιο είναι άχρηστο επειδή υπάρχει ήδη .

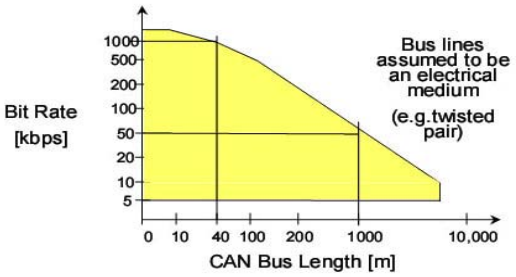
Διαγράφηκε: ς

Διαγράφηκε: ω

Οι τυπικές τιμές του μέγιστου ρυθμού μετάδοσης δεδομένων , οι οποίες εξαρτώνται και από το μέσο μετάδοσης (στην περίπτωση μας δύο στρεφόμενα καλώδια χαλκού) , σε συνάρτηση με το μήκος των καλωδίων των CAN , φαίνονται στα παρακάτω διαγράμματα

Bit Rate	Bus Length
1Mbit/s	25 m
800 kBit/s	50 m
500 kBit/s	100 m
250 kBit/s	250 m
125 kBit/s	500 m
50 kBit/s	1000 m
20 kBit/s	2500 m
10 kBit/s	5000 m

3.4 For a typical two twisted pair wires

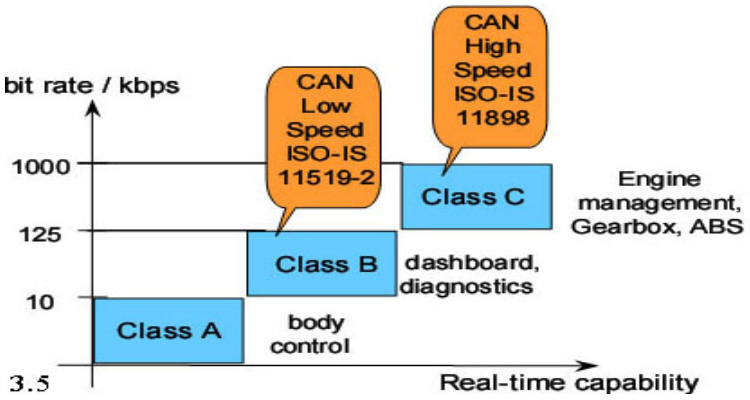


Το σχήμα [σχ.3.1] δείχνει ότι όσο μικρότερο το μήκος του διαύλου τόσο υψηλότερος ο ρυθμός μετάδοσης δεδομένων . Για μήκος διαύλου μικρότερο των 25 μέτρων ο ρυθμός μετάδοσης δεδομένων είναι 1Mbit/sec . Κάτω από το χειρότερο σενάριο ο ρυθμός μετάδοσης αυτός μπορεί να είναι εφικτός αν το μήκος του διαύλου δεν ξεπερνάει το 1 μέτρο μήκος . [4] [6]

Γενικά οι παράμετροι που επηρεάζουν το ρυθμό μετάδοσης είναι λόγοι αγωγιμότητας των καλωδίων και οι συνδέσεις , ο αριθμός των κόμβων του δικτύου καθώς και ο χρόνος καθυστέρησης του κάθε πομποδέκτη .

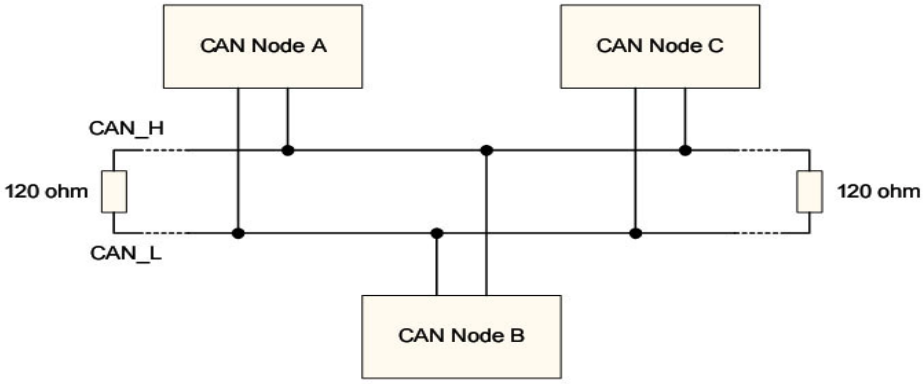
Μερικές φορές δεν είναι αναγκαίο ο ρυθμός μετάδοσης να είναι ο μέγιστος και αυτό εξαρτάται από την εκάστοτε εφαρμογή και κατά πόσο είναι πραγματικού χρόνου . Οι εφαρμογές χωρίζονται σε τρεις κατηγορίες ανάλογα με το πόσο πραγματικού χρόνου είναι .

- Κατηγορία Α : Ρυθμοί μετάδοσης μέχρι 10kbps , για body control εφαρμογές .
- Κατηγορία Β : Ρυθμοί μετάδοσης από 10kbps έως 125kbps , για πίνακες οργάνων και διαγνωστικά
- Κατηγορία Γ : Ρυθμοί μετάδοσης από 125kbps έως 1Mbps για πραγματικού χρόνου εφαρμογές . [4]



Τα ISO-IS 11529-2 και 11898 έγιναν διεθνή standard το 1995 και το 2003 αντίστοιχα . Όπως βλέπουμε στο σχήμα παραπάνω το ISO 11898 αναφέρεται στις εφαρμογές κατηγορίας Γ , και το ISO 11519-2 στις εφαρμογές κατηγορίας Β με χαμηλότερους ρυθμούς μετάδοσης δεδομένων (σχ.3.5) . Αυτές οι διαφορές βρίσκονται στο φυσικό επίπεδο του πρωτοκόλλου . [4]

Το παρακάτω σχήμα παρουσιάζει ένα κλασικό Controller Area Network , το οποίο και περιλαμβάνει τρεις κόμβους , ζεύγη περιελιγμένων καλωδίων και δύο αντιστάσεις οι οποίες και ενώνουν τα CANH και CANL καλώδια . Οι προτεινόμενες τερματικές αντιστάσεις πρέπει να είναι στα 120Ohm για ταχύτητες διαύλου στα 1Mbit/sec , αλλά μακρύτερα καλώδια και δίκτυα χαμηλότερων ρυθμούς μετάδοσης απαιτούν αντιστάσεις 150 με 300 Ohm . [9]



3.6 A typical CAN wiring diagram

Στις επόμενες παραγράφους θα υπάρξει ανάλυση των CAN διαύλων , και θα εξεταστούν τα πλεονεκτήματα και τα λιγοστά μειονεκτήματα . Πρώτα απ'όλα , ο CAN δίαυλος μπορεί να θεωρηθεί ως ένας διαφορετικός δίαυλος μεταφοράς δεδομένων για τον λόγο ότι στην μια περίσταση δυο αντίστροφα σήματα εκπέμπονται και σε μια δεύτερη περίσταση τα ίδια σήματα εκπέμπονται και στα δύο καλώδια . Τα CANH και CANL συμβολίζουν το CAN υψηλό (High) και CAN χαμηλό (Low) και συμβολίζουν το '1' και το ΄0' αντίστοιχα . [σχ.3.7]

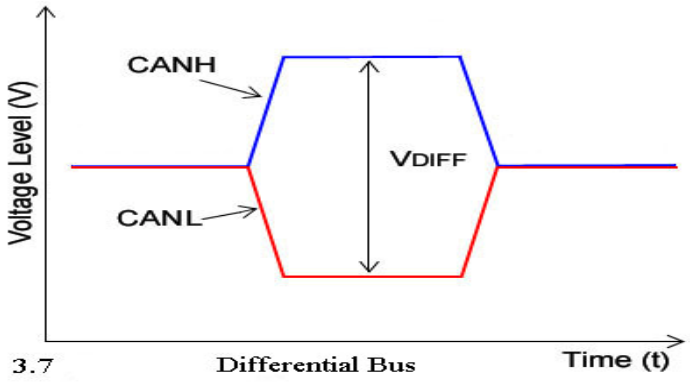
Μορφοποιήθηκε: Ελληνικά

Το πρώτο βασικό πλεονέκτημα των CAN διαύλων είναι η ισχυρή ανοχή στον θόρυβο η οποία σε μερικές περιπτώσεις είναι 10V , το οποίο εξαρτάται από την τάση των CANL και CALH . Μπορούμε να πάρουμε αυτές τις τιμές αν κάνουμε την αφαίρεση του CANH-CANL και το αντίστροφο . Πιο συγκεκριμένα όταν το '1' μεταφέρεται στον δίαυλο CANH-CANL ισούται στο xV και όταν είναι '0' CANH-CANL αντιστοιχεί στο -xV . Αυτή η διαφορά x - (- x) = 2 Xv το οποίο και παριστά την ανοχή του διαύλου και οι τιμές αυτές μπορεί να είναι μέχρι 10V.

Μορφοποιήθηκε: Ελληνικά

Το δεύτερο πλεονέκτημα είναι η ικανότητα στο να απορρίπτει τον θόρυβο . Η αρχιτεκτονική του CAN διαύλου είναι παρόμοια με την αρχιτεκτονική στρεφόμενου ζεύγους καλωδίων . Αυτό σημαίνει ότι λειτουργεί ως ασπίδα στο εκπεμπόμενο σήμα έτσι ώστε να μην υπάρχουν απώλειες . Για παράδειγμα , αν N Volts ενός εξωτερικού σήματος επηρεάσει και τις δύο γραμμές του CAN διαύλου , CANH + N – (CANL + N) το αποτέλεσμα είναι CANH – CANL καθώς το N διαγράφεται .

Το τρίτο πλεονέκτημα του CAN διαύλου έχει να κάνει με τους κανόνες που ισχύουν για την όσο δυνατό καλύτερη λειτουργία του δικτύου . Στον CAN δίαυλο είναι δυνατό να εκπέμψουν πολλοί κόμβοι ταυτοχρόνως .



3.7 Differential Bus

Ένα μειονέκτημα του συγκεκριμένου διαύλου είναι ότι για να λειτουργήσει ιδανικά πρέπει να χρησιμοποιηθούν πολύ συγκεκριμένο hardware καθώς και ανάλογες ρυθμίσεις . Αυτοί οι περιορισμοί αναφέρονται κυρίως σε μια μικρή αντίσταση που ενώνει τα CANH και CANL , και πρέπει να έχει συγκεκριμένη τιμή για την ανάλογη μετάδοση δεδομένων , έτσι ώστε να αποφεύγονται τα εξωτερικά σήματα από ανακλάσεις που μειώνουν το εύρος ζώνης του διαύλου . Τέλος ένα μειονέκτημα το οποίο εύκολα ξεπερνιέται είναι και η χρήση του σωστού μέσου (καλώδιο , ίνα) το οποίο και επηρεάζει το σύστημα .

3.v. Εκτενέστερη ανάλυση του CAN πρωτοκόλλου

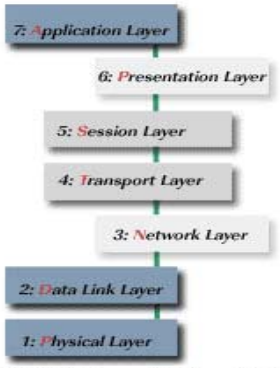
Συνήθως για να γίνει πιο κατανοητή η περιγραφή ενός τηλεπικοινωνιακού δικτύου , γίνεται με βάση του μοντέλου Open System Interconnection , γνωστού ως μοντέλου OSI . Αυτό το μοντέλο έχει ορισθεί ως επίσημο standard για την περιγραφή των 7 στρωμάτων , από το

χαμηλότερο , φυσικό επίπεδο έως και το υψηλότερο επίπεδο αυτό των εφαρμογών . [σχ.3.8]

Διαγράφηκε: εώς

Το CAN πρωτόκολλο ορίζεται μόνο από τα δύο χαμηλότερα στρώματα (1.το φυσικό επίπεδο και 2 . το δίκτυο δεδομένων) και το υψηλότερο στρώμα (7 . το επίπεδο εφαρμογών) . Τα ενδιάμεσα στρώματα δεν είναι υποχρεωτικά έτσι ώστε να γίνει περιγραφεί του δικτύου αλλά μπορεί μερικώς να περιγραφεί από υψηλού επιπέδου CAN όπως το CANopen .

Μορφοποιήθηκε: Ελληνικά



3.8 OSI model for CAN

1. *Φυσικό επίπεδο*

Το πρώτο στρώμα του OSI μπορεί να περιγραφεί από τρεις παραμέτρους

- Αλληλεπίδραση φυσικού μέσου

  Περιλαμβάνει τα καλώδια και τους συνδετήρες . Κάναμε ήδη εκτενή αναφορά στα καλώδια και όλα τα δυνατά μέσα καθώς και για τους διαφορετικούς ρυθμούς μετάδοσης που μπορούν να επιτευχθούν . Σχετικά με τους συνδετήρες , ανάλογα με τις διαφορετικές εφαρμογές υπάρχουν και διαφορετικά ήδη συνδετήρων . Τα βασικά στοιχεία ενός συνδετήρα είναι :

  CANH: Η γραμμή του διαύλου οδηγείται υψηλά στην κυρίαρχη κατάσταση.

  CANL: Η γραμμή του διαύλου οδηγείται χαμηλά στην κυρίαρχη κατάσταση.

  CAN Ground: Είναι μια τρίτη γραμμή που λειτουργεί ως γείωση. Είναι πιθανό να μην χρειάζεται .

  CAN Shield: Προαιρετική θωράκιση μεταξύ των CANH και CANL.

Μορφοποιήθηκε: Ελληνικά

Μερικά από τα πιο δημοφιλή είδη συνδετήρων είναι ο 9-Pin D-Sub (αρσενικός συνδετήρας για τη σύνδεση της συσκευής ή του δικτύου και το θηλυκό το οποίο και προβλέπεται για το καλώδιο) .

- Σύνδεση φυσικού μέσου

Αυτό το μέρος του στρώματος περιλαμβάνει τους οδηγούς , καθώς και τα χαρακτηριστικά του πομπού και του δέκτη . Η διασύνδεση βασικά αποτελείται από έναν ενισχυτή πομπού και έναν ενισχυτή δέκτη (κοινώς πομποδέκτη) . Καθώς ο πομπός προσφέρει ικανοποιητική χωρητικότητα εξόδου και προστατεύει το chip του μικροελεγκτή από υπερφόρτωση δεδομένων . Επίσης έχει την δυνατότητα να ελαχιστοποιεί την ηλεκτρομαγνητική ακτινοβολία . Ακόμη διευρύνει την ακτίνα δράσης της συσκευής σύγκρισης στον CAN controller και προσφέρει αποτελεσματική εσωτερική ευαισθησία . Εκτός αυτού μια δυνατότητα του πομποδέκτη είναι η γαλβανική θωράκιση των CAN κόμβων και των καλωδίων. Τέλος ανιχνεύει λάθη στον δίαυλο όπως κομμένα καλώδια , βραχυκυκλώματα , λανθασμένες γειώσεις κ.α. .

Διαγράφηκε: Μια επιπλέον δυνατότητα αυτού του επιπέδου είναι ότι καθώς ο CAN δέκτης .....

Διαγράφηκε: ου

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Διαγράφηκε: .....

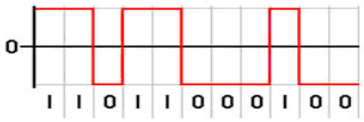
- Φυσικό σήμα

Αυτό το μέρος περιλαμβάνει την κωδικοποίηση/αποκωδικοποίηση και τον συγχρονισμό των bit .

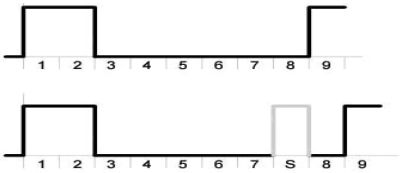
Οι πιο κοινές τεχνικές που χρησιμοποιούνται που χρησιμοποιούνται για κωδικοποίηση και αποκωδικοποίηση στα CAN είναι η NRZ (Non Return to Zero) (σχ.3.9) . Σύμφωνα με αυτόν τον τρόπο κωδικοποίησης , το σήμα παραμένει ‘0’ ή ‘1’ για μια ολόκληρη χρονική σχισμή . Αν το πλαίσιο αποτελείται από μια αλληλουχία άσων και μηδενικών , το σήμα θα παραμείνει σταθερό για όσα bit χρειαστεί . Το μειονέκτημα του NRZ είναι ότι είναι δύσκολο να αντιληφθείς πότε κάθε bit ξεκινάει και σταματάει όταν υπάρχουν πολλοί άσοι και μηδενικά στην σειρά .

Για λόγους συγχρονισμού , τα CAN χρησιμοποιούν την bit stuffing τεχνική με δεδομένα . Οι κορυφές απαιτούνται με κάθε ροή δεδομένων . Για να επιβεβαιώσουμε το παραπάνω πρέπει να υπάρχουν πολλές κορυφές

, ο πομπός εισάγει αυτόματα εισάγει ένα αντίθετο stuff bit στη ροή δεδομένων κάθε πέντε συνεχόμενα bits . [7]



3.9 NRZ Coding

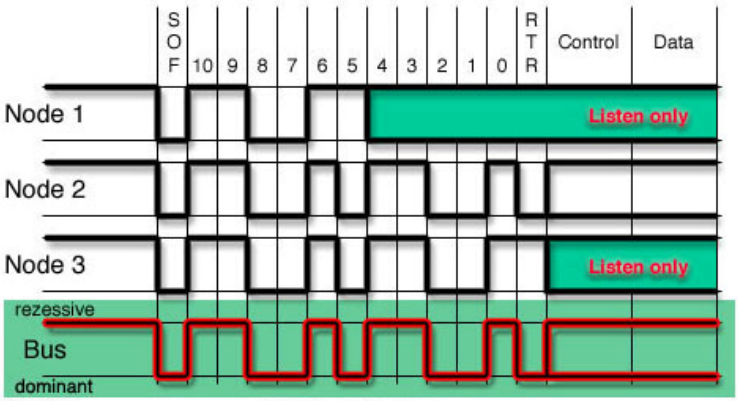


Bit Stuffing

Τέλος πρέπει να κάνουμε τη διάκριση μεταξύ δύο σημαντικών καταστάσεων του σήματος , την "επικρατούσα" και την "υπολειπόμενη" κατάσταση . Η υπολειπόμενη συμβολίζεται με το λογικό '1' , και συμβαίνει όταν τα CANH και CANL είναι στο ίδιο επίπεδο (π.χ. CANH=CANL=περίπου 2.5V , αντιθέτως στην επικρατούσα κατάσταση , που συμβολίζεται με το λογικό '0΄ , και συμβαίνει όταν υπάρχει διαφορά στα επίπεδα των CANH και CANL (π.χ. CANH=3.5V και CANL=1.5V) . Επίσης στην επικρατούσα κατάσταση τα CANL και CANH έχουν συνήθως διαφορά 1V από το επίπεδο που βρίσκονται στην υπολειπόμενη κατάσταση .

Στα CAN πάντα η κυριαρχούσα κατάσταση επικρατεί της υπολειπόμενης . Αν πολλαπλοί κόμβοι προσπαθούν ταυτοχρόνως να εκπέμψουν , το σήμα θα μείνει στην επικρατούσα κατάσταση του . Αυτός ο μηχανισμός χρησιμοποιείται για να ανιχνεύει συγκρούσεις . Στην περίπτωση που ένας κόμβος εκπέμψει σε υπολειπόμενη κατάσταση και διαβάσει τελικά σε επικρατούσα , τότε είναι σίγουρο ότι υπάρχει κάποια σύγκρουση και πρέπει να γίνει κατάσταση ανάκτησης δεδομένων . (σχ.3.10) [7]

Διαγράφηκε: επικρατών
Διαγράφηκε: ς
Διαγράφηκε: ,
Διαγράφηκε: σ



3.10 CAN arbitration

Αυτός ο CAN μηχανισμός έχει ίδια συμπεριφορά με την πύλη AND (Σχ.3.11) . Μόνο στην περίπτωση που όλοι οι κόμβοι εκπέμψουν υπολειπόμενο σήμα , το σήμα του διαύλου θα είναι στην υπολειπόμενη κατάσταση . Σε οποιαδήποτε άλλη περίπτωση το σήμα , το σήμα θα είναι στην επικρατούσα κατάσταση . [5]

Bus state with two nodes transmitting

	dominant	**recessive**
dominant	***dominant***	*dominant*
recessive	**dominant**	***recessive***

Logical AND

	0	1
0	0	0
1	0	1

3.11 Comparison CAN arbitration , " AND " Gate

- Χρονισμός Bit

Το CAN πρωτόκολλο χρησιμοποιεί σύγχρονη μετάδοση δεδομένων για να γίνει η επικοινωνία πιο αποτελεσματική . Τα βασικά χαρακτηριστικά μιας σύγχρονης μετάδοσης είναι ότι δύο ή περισσότεροι κόμβοι χρησιμοποιούν τον ίδιο χρονισμό , και το σημείο αναφοράς ξεκινάει στην αρχή του μηνύματος .Ανεξάρτητα από αυτό είναι αποτελεσματική τεχνική , εξαιτίας του επανασυγχρονισμού των κόμβων και της καθυστέρησης λόγω διασποράς , λάθος φάσης και κατεύθυνσης του ταλαντωτή έτσι ώστε είναι εξαιρετικά δύσκολο να κρατηθεί συγχρονισμένο .

Όπως έχουμε ήδη αναφέρει στις βασικές λειτουργίες του CAN , όλοι οι κόμβοι των CAN πρέπει να έχουν το ίδιο ρυθμό μετάδοσης δεδομένων έτσι ώστε να είναι πάντα συγχρονισμένα μεταξύ τους . Ο ρυθμός

μετάδοσης καθορίζεται από τον αριθμό των bits που περνάνε από ένα συγκεκριμένο σημείο του δικτύου ανά δευτερόλεπτο . Ο ονομαστικός-ιδανικός ρυθμός μετάδοσης είναι ο αριθμός των bit που εκπέμπονται το δευτερόλεπτο από έναν ιδανικό πομπό χωρίς επανασυγχρονισμό

Για να μπορέσουμε να έχουμε ιδανικό χρονισμό των bit και τον ονομαστικό χρόνο των bit (bit time – το οποίο είναι το αντίστροφο του ρυθμού μετάδοσης των bit) , το χωρίζουμε σε τέσσερα τεμάχια . (σχ.3.12)

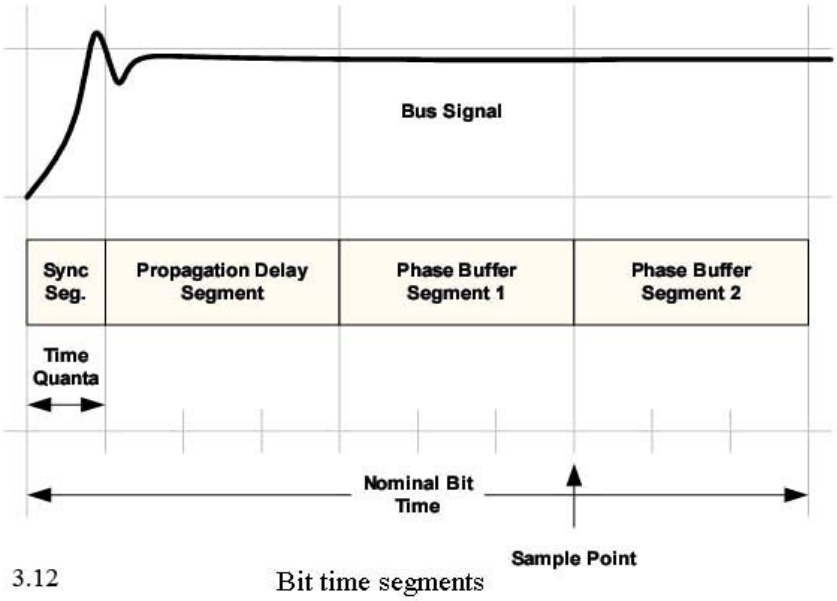
- Τεμάχιο συγχρονισμού : Είναι το πρώτο bit , και βρίσκεται εκεί όπου η αλλαγή στο σήμα περιμένουμε να συμβεί . Το τεμάχιο συγχρονισμού είναι πάντα σταθερό στο ένα κβάντο χρόνου .

- Καθυστέρηση τεμαχίου λόγω διασποράς : Πρέπει να είναι τουλάχιστον διπλάσιο από τον χρόνο που ένα bit χρειάζεται για να φτάσει τον γρηγορότερο κόμβο , για τον λόγο ότι προστίθεται ο χρόνος που χρειάζεται για να γυρίσει πίσω .

- Buffer της φάσης τεμαχίου 1&2 : Αυτά τα buffer εξάπτονται από το προηγούμενο τεμάχιο και έχει αντίστροφη λειτουργία . Το buffer 1 μακραίνει καθώς η μετάβαση από υπολειπόμενο σε επικρατών συμβαίνει κατά την διάρκεια της καθυστέρησης διασποράς . Αντιθέτως το buffer 2 μικραίνει καθώς η μετάβαση από υπολειπόμενο σε επικρατόν συμβαίνει κατά την διάρκεια της αλλαγής φάσης του τεμαχίου . [7]

Διαγράφηκε: ώ

- Σημείο δειγματοληψίας : Βρίσκεται πάντα στο τέλος του buffer της φάσης τεμαχίου .



3.12 Bit time segments

Τα CAN χρησιμοποιούν δύο τρόπους συγχρονισμού , τον αυστηρό συγχρονισμό και τον επανασυγχρονισμό .

- Αυστηρός συγχρονισμός : Συμβαίνει μόνο μια φορά κατά την μετάδοση ενός μηνύματος , από την υπολειπόμενη στην επικρατούσα κατάσταση και την αρχή κάθε bit . Ο χρόνος του κάθε bit (time bit) επαναρχίζει στο τέλος κάθε τεμάχιο συγχρονισμού και μετά ενός αυστηρού συγχρονισμού .

- Επανασυγχρονισμός : Συμβαίνει μετά από κάθε μετάβαση από επικρατών σε κυρίαρχο bit η οποία και δε συμβαίνει κατά την διάρκεια ενός αναμενόμενου τεμάχιου συγχρονισμού . [6] [12]

2 . Data link layer

Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά

Το δεύτερο στρώμα του OSI μπορεί να περιγραφεί από δύο παραμέτρους .

- Medium Access Control (MAC)

Αυτό το μέρος περιλαμβάνει τη βασικό format ενός CAN πλαισίου , την ενθυλάκωση/απενθυλάκωση δεδομένων , συγκρούσεις και λάθη του σήματος καθώς και σειριακή διάταξη και το αντίθετο .

- Τα CAN έχουν τέσσερα διαφορετικά είδη πλαισίου

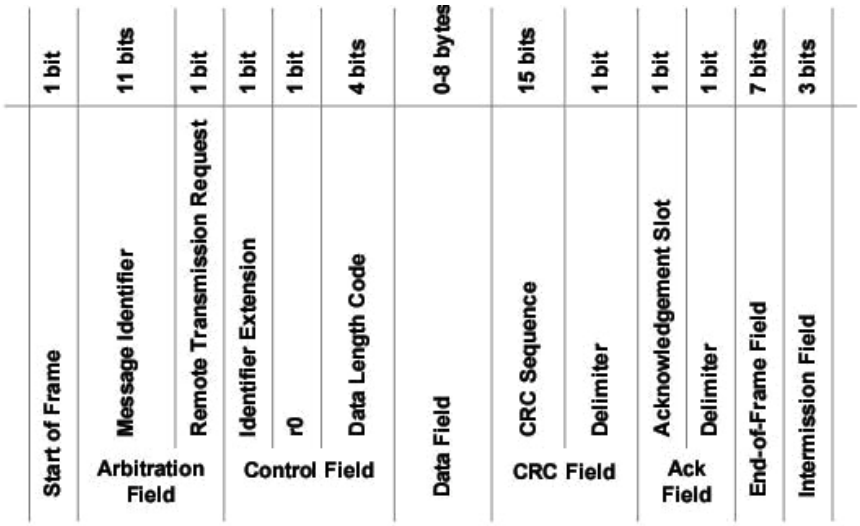
Data frames

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Τα data frames είναι τα "μηνύματα" ή το σήμα που εκπέμπετε από έναν κόμβο στο δίκτυο . Υπάρχουν δύο είδη data frames το βασικό format και το εκτεταμένο format το οποίο είναι και κατά πολύ μεγαλύτερο . Ο λόγος που δημιουργήθηκε το εκτεταμένο format είναι ότι κάποιες εφαρμογές με που απαιτούν υψηλούς ρυθμούς μετάδοσης , όπως στα λεωφορεία και στα φορτηγά , που το βασικό format είναι ανεπαρκές .Στην περίπτωση αυτή "ανεπαρκής" σημαίνει όταν τα δυνατά ID είναι περισσότερα από αυτά που το CAN μπορεί να δημιουργήσει . Τα δυνατά ID εξάπτονται από τον identifier , στο βασικό format είναι 2^{11} περιπτώσεις , ενώ στο εκτεταμένο είναι 2^{29} .

Βασικό format πλαισίου



3.13 Base Format Data Frame

Η έναρξη του διαβάσματος ξεκινάει από αριστερά προς τα δεξιά. Το CAN πλαίσιο δεδομένων ξεκινάει με ένα επικρατών bit . Όταν ο δίαυλος είναι αδρανής , είναι στην υπολειπόμενη κατάσταση , κάθε μετάβαση από αδράνεια σε επικρατούσα κατάσταση θεωρείται ως η έναρξη πλαισίου

Διαγράφηκε: ι

Διαγράφηκε: έ

Διαγράφηκε: ς

Διαγράφηκε: έτσι

Διαγράφηκε: ε

Διαγράφηκε: ί

Τα επόμενα 11 bit αποτελούν τον αναγνωριστή μηνυμάτων (message identifier). Δεν πρέπει να είναι τα ίδια CAN ID πλαίσια στο δίκτυο οποιαδήποτε στιγμή από οποιοδήποτε κόμβο .

Το επόμενο bit είναι τα Remote transmission request bit τα οποία και καθορίζουν αν τα bit είναι δεδομένα ή απομακρυσμένα πλαίσια δεδομένων . Αυτό το bit πρέπει να είναι επικρατές για ένα πλαίσιο δεδομένων .

Τα επόμενα τρία bits είναι τα bit ελέγχου . Τα εκτεταμένα bit αναγνώρισης καθορίζουν το format του πλαισίου , αν είναι δηλαδή βασικό ή εκτεταμένο . Το επόμενο bit είναι το εφεδρικό bit και πρέπει να είναι πάντα επικρατών . Το τελευταίο bit είναι τα bit κώδικα δεδομένων , τα οποία και έχουν τυπικές τιμές από το μηδέν έως το οχτώ . Το πιο σημαντικό bit είναι το επόμενο . Είναι το πεδίο το οποίο και περιέχει το ωφέλιμο φορτίο και είναι το μοναδικό πεδίο που δεν έχει standard μέγεθος . Το κυκλικό bit περίσσιας ελέγχου που είναι μέγεθος 15bit , και είναι τα bit διορθώσεως καθώς ελέγχουν αν η αλληλουχία των bit εισέρχεται ορθά . Το επόμενο είναι το bit παραδοχής και είναι μεγέθους 2bit . Περιέχεται από το bit παραδοχής και τον delimiter .

Εκτεταμένο Format πλαισίου

Η μόνη διαφορά μεταξύ του βασικού και του εκτεταμένου πλαισίου είναι το μήκος των bit του αναγνωριστή bit και το οποίο μπορεί να πάρει 535 εκατομμύρια τιμές σε σχέση με το βασικό που μπορεί να πάρει μόλις 2048 .

Απομακρυσμένο πλαίσιο

Τα απομακρυσμένα πλαίσια χρησιμοποιούνται από τους δέκτες για να ζητούν πληροφορίες από άλλους κόμβους . Έχουν το ίδιο format με το πλαίσια δεδομένων όπως και με τα εκτεταμένα πλαίσια , και η χρήση τους είναι να στέλνουν αιτήσεις σε καθορισμένα χρονικά διαστήματα και να ενημερώνονται από τις συνθήκες του δικτύου . Χρησιμοποιούνται στην περίπτωση που κάποιος κόμβος αποσυνδεθεί από το δίκτυο και υπάρχει δυνατότητα σύνδεσης στο μέλλον .

Λάθη πλαισίου

- Η ενθυλάκωση δεδομένων ενεργοποιεί πολλαπλά πρωτόκολλα για να ενεργοποιηθούν σε ένα τμήμα του CAN διαύλου .

- Αναζήτηση λαθών

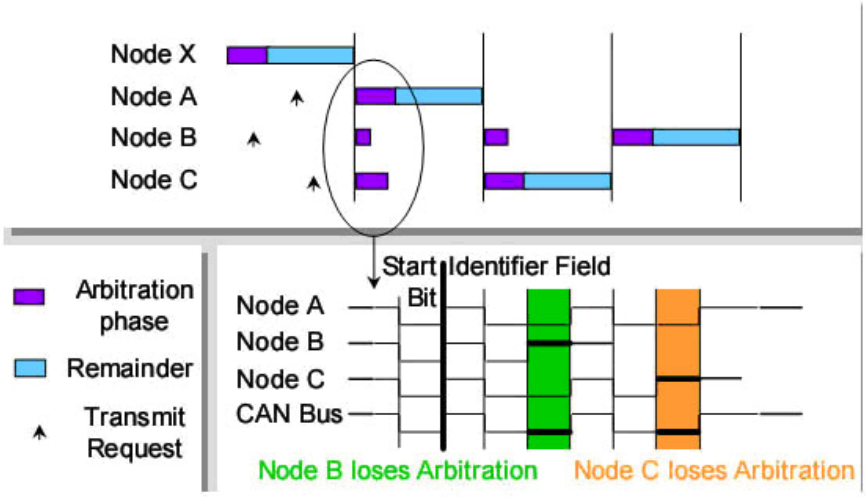
Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Το πρωτόκολλο που χρησιμοποιείται στις περιπτώσεις αυτές για αναζήτηση συγκρούσεων αλλά και για τους κανόνες που πρέπει να επικρατήσουν είναι το CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance) . Η προοπτική αυτού του πρωτοκόλλου είναι να μην σπαταλιέται εύρος ζώνης. “Carrier Sense “ (Αίσθηση του φέροντος) και σημαίνει ότι ο κόμβος συνεχώς “senses the carrier” (αισθάνεται το φέρον) , γνωρίζοντας πάντα την κατάσταση που επικρατεί στο δίκτυο . “Multiple Access” (πολλαπλή πρόσβαση) σημαίνει ότι πολλαπλοί χρήστες έχουν πρόσβαση στο φέρον την ίδια χρονική στιγμή – αν δεν υπάρχει ροή δεδομένων στο δίκτυο – πολλαπλοί χρήστες ίσως προσπαθήσουν να επέμβουν στο δίκτυο την ίδια χρονική στιγμή . Το “Collision Avoidance” (Αποφυγή συγκρούσεων) σημαίνει ότι υπάρχει μηχανισμός ο οποίος προωθεί το σήμα στο δίκτυο , σύμφωνα με την προτεραιότητα του κάθε μηνύματος .

Το μήνυμα με την μεγαλύτερη προτεραιότητα κερδίζει κάθε διαδικασία , αλλά εκτός από αυτό , επαναλαμβάνεται για όλα τα μηνύματα που χάνουν την σειρά τους . Το πρωτόκολλο αυτό υλοποιείται από το hardware έτσι δεν χρειάζεται να γίνονται επεμβάσεις στο λογισμικό .

Το παρακάτω σχήμα ένα τυπικό κύκλο CAN διεργασιών . Στην περίπτωση αυτή πολλαπλοί κόμβοι προσπαθούν να στείλουν κάποιο πλαίσιο , όταν ήδη το δίκτυο χρησιμοποιείται από κάποιον άλλον κόμβο . Πιο συγκεκριμένα οι κόμβοι Α, Βκαι C προσπαθούν να εκπέμψουν ενώ ήδη ο κόμβος Χ στέλνει δεδομένα . Αυτοί οι κόμβοι ενημερώνονται συνεχώς για την χρήση-κατάσταση του δικτύου , και περιμένουν για την λήξη της μετάδοσης από τον κόμβο Χ.

Μορφοποιήθηκε: Ελληνικά



3.14 The CAN arbitration Process

Όπως βλέπουμε στο σχήμα παραπάνω (σχ.3.14) όταν ο κόμβος τελειώνει την εργασία του με το να στείλει και τα υπόλοιπα bit , οι αδρανείς κόμβοι στέλνουν ταυτοχρόνως ένα κυρίαρχο αρχικό bit . Όπως μπορούμε να παρατηρήσουμε ο κόμβος B χάνει την "μάχη" καθώς στέλνει ένα επικρατές bit και ο CAN δίαυλος περιέχει ένα υπολειπόμενο bit , και στο επόμενο στάδιο ο κόμβος C χάνει την μάχη καθώς στέλνει ένα επικρατές bit αλλά στο CAN δίαυλο φαίνεται ένα υπολειπόμενο . [4]

Ο αλγόριθμος πηγαίνει ως ακολούθως :

- Κάθε κόμβος περιέχει μέχρι ο δίαυλος να ελευθερωθεί .
- Στην αρχή της διαδικασίας πάντα στέλνεται ένα επικρατόν bit .
- Στέλνει τα bit αναγνωριστή που είναι 11 ή 29 , και ακολούθως τα υπόλοιπα bit που περιέχουν τα δεδομένα .
- Διαβάζει τα bit από τον δίαυλο .
- Αν τα bit που στέλνονται είναι διαφορετικά από τα bit που παραλαμβάνονται από τους κόμβους τότε ο αλγόριθμος ξεκινάει πάλι από το πρώτο βήμα .
- Αν ο αριθμός των bit που διαχειρίζονται δεν είναι 11 , τότε επανέρχεται στο πρώτο βήμα .
- Εάν όχι , τότε όλα πήγαν σύμφωνα με τους κανόνες , και η διαδικασία ολοκληρώθηκε επιτυχώς .
- Μηχανισμός εύρεσης λαθών .

Διαγράφηκε: ές

Μορφοποιήθηκε: Ελληνικά

Υπάρχουν πέντε μηχανισμοί εύρεσης λαθών τρεις από τους οποίους βρίσκονται στο επίπεδο μηνυμάτων ενώ οι άλλοι δύο βρίσκονται στο επίπεδο των bit .

Μορφοποιήθηκε: Ελληνικά

- CRC (Cyclic Redundancy bit – κυκλικό περίσσιο bit)

Έχοντας την βοήθεια του σχήματος 4.1 μπορούμε να δούμε το CRC , 15 byte (0-14) που ανήκουν στο πρωτόκολλο . Όταν ένα μήνυμα εκπέμπεται από τον πομπό , υπολογίζεται επίσης το άθροισμα κυκλικής περίσσιας ελέγχου μνήμης . Σε αυτή την περίπτωση το άθροισμα ελέγχου μνήμης περιέχει αμφότερα τα bit του αναγνωριστή και τα byte με τα δεδομένα τα οποία εκπέμπονται . Ο δέκτης (άλλου κόμβου) , δέχεται τα μηνύματα μέσω του CANbus και τα συγκρίνει με το υπολογισμένο άθροισμα ελέγχου μνήμης . Αν υπάρχει ταύτιση τότε τραβιούνται τα ομόλογα bit στην κυρίαρχη κατάσταση έτσι ώστε να αποδειχθεί ότι η μεταφορά ήταν επιτυχής Στην αντίθετη περίπτωση ένα CRC λάθος έχει συμβεί , η αποστολή μηνύματος έχει ανασταλεί και ένα πλαίσιο λάθους για επανεκπομπή μηνύματος θα σταλεί .

- Form error check (έλεγχος φόρμας λάθους)

Αν ένα κυρίαρχο bit εκπεμφθεί από τέσσερα προκαθορισμένα τμήματα , τα οποία όλα εκπέμπουν υπολειπόμενα bit , τότε σίγουρα ένα λάθος έχει συμβεί και ένα πλαίσιο λάθους εκπέμπεται . Ως αποτέλεσμα το μήνυμα επανεκπέμπεται .

- Έλεγχος bit παραδοχής

Αν ο πομπός ανιχνεύσει το bit παραδοχής να βρίσκεται στην κυρίαρχη κατάσταση , το μήνυμα δεν έχει ληφθεί ορθά και πρέπει να επανεκπεμφθεί

- Λάθος γέμισμα bit

Λάθος γέμισμα bit συμβαίνει στην περίπτωση που 6 συνεχόμενα bit από την έναρξη ενός πλαισίου μέχρι το CRC έχουν την ίδια πολικότητα , τότε ένα λάθος έχει ανιχνευθεί και το μήνυμα πρέπει να επανεκπεμφθεί .

- Σφάλμα bit

Ένα σφάλμα bit συμβαίνει όταν ένα κυρίαρχο bit εκπέμπεται και ένα υπολειπόμενο bit εισπράττεται και το αντίστροφο . Επομένως αν υπάρχει διαφορά στα επίπεδα που εκπέμπονται και παραλαμβάνονται , καθώς τα παρακολουθούμε , τότε έχει συμβεί σίγουρα κάποιο λάθος .

ΚΕΦΑΛΑΙΟ 4

Σχεδιασμός CAN Controller

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

4.i. Διαφορετικές CAN υλοποιήσεις

Μορφοποιήθηκε: Ελληνικά

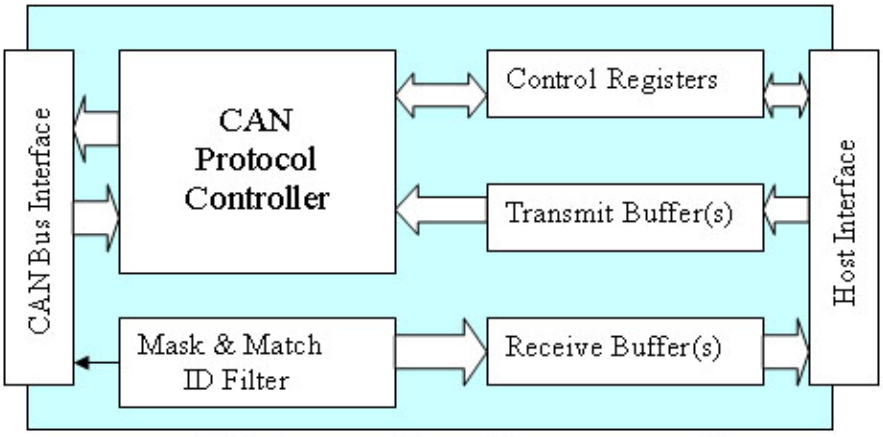
Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Υπάρχουν τέσσερις διαφορετικές CAN υλοποιήσεις , δύο περισσότερο δημοφιλείς και δύο όχι τόσο .Οι διαφορές στην υλοποίηση έγκειται κυρίως στο interface και στον επεξεργαστή .

- Παραδοσιακό βασικό CAN

Αρχικά είχε υλοποιηθεί από τον Philips , και τα βασικά του χαρακτηριστικά είναι η απλότητα και οι βασικές λειτουργίες που διαθέτει . Στις περισσότερες υλοποιήσεις χρησιμοποιεί μόνο το εκπεμπόμενο buffer για τα εξερχόμενα μηνύματα και από το πρώτο έως το τρίτο buffer αποδοχής για τα εισερχόμενα μηνύματα . Στην περίπτωση που ο controller κάθε κόμβου θέλει να έχει την προσοχή του σε περισσότερα από ένα μηνύματα ταυτόχρονα , ο controller πρέπει να διακοπεί έτσι ώστε να γίνεται η λήψη των μηνυμάτων φυσιολογικά χωρίς επικαλύψεις των επόμενων μηνυμάτων . Ως αποτέλεσμα ο δέκτης πρέπει να δέχεται πολλές διακοπές για το λόγο ότι πρέπει συνεχώς να γίνεται έλεγχος για το αν ένα μήνυμα πρέπει να ληφθεί ή να αποβληθεί .



4 1 **Block diagram of a Basic CAN Controller**

Μορφοποιήθηκε: Ελληνικά

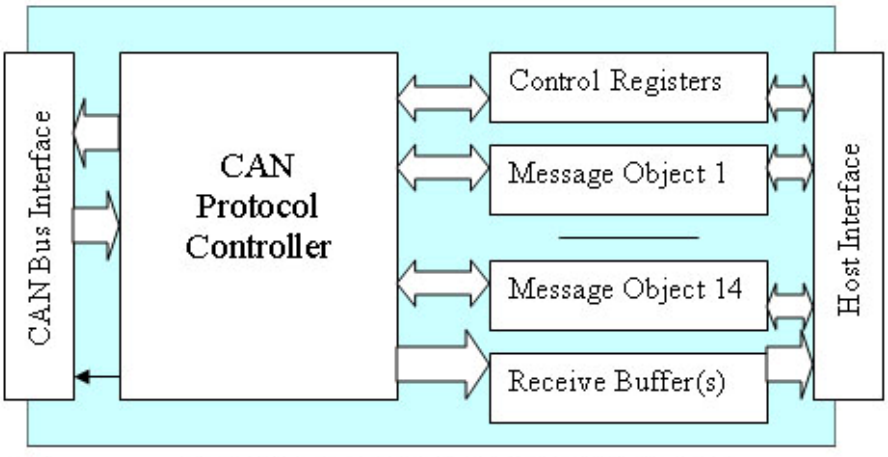
- Παραδοσιακό πλήρες CAN

Το πρόβλημα με τις διακοπές σε μια παραδοσιακή βασική CAN υλοποίηση τελειώνει με την υιοθέτηση ενός παραδοσιακού πλήρες CAN . Η διαφορά στην υλοποίηση είναι η χρησιμοποίηση περισσότερων buffer (τυπικά 15) . Επιπλέον σε αντίθεση με την βασική υλοποίηση κάθε μήνυμα μπορεί να εκπέμπεται και να λαμβάνεται καθώς η κατεύθυνση είναι και προς τις δύο κατευθύνσεις . Εκτός από το παραπάνω κάθε μήνυμα διαθέτει και το ανάλογο φίλτρο του .

Ο controller λειτουργεί ιδανικά στην περίπτωση που ο αριθμός των μηνυμάτων που λαμβάνεται είναι ίσος ή μικρότερος από τον αριθμό των buffers . Όμως αν ο αριθμός των identifier είναι μεγαλύτερος από τον αριθμό των buffer τότε δεν υπάρχει ουσιαστικά καμία διαφορά .

Ένα επιπλέον πλεονέκτημα της πλήρης CAN υλοποίησης είναι ότι τα φίλτρα μηχανισμού αποδοχής μάσκας δε δέχονται άσχετα μηνύματα , με το να χρησιμοποιούν identifiers μηνύματος και να τροφοδοτούν τον controller μόνο με τα μηνύματα που είναι σχετικά με αυτόν .

Από την άλλη μεριά , μια πλήρης CAN υλοποίηση είναι γενικά μια περίπλοκη υλοποίηση . Υπάρχει ένα βασικό μειονέκτημα , καθώς τα δεδομένα στο buffer δεν είναι δυνατό να ενημερωθούν από μόνα τους και η συνταύτιση είναι δύσκολο να διατηρηθεί .Συνεπώς τα δεδομένα που γράφονται στο buffer μπορούν να επανεγραφούν από τον controller και στο τέλος να μην είναι έγκυρα .

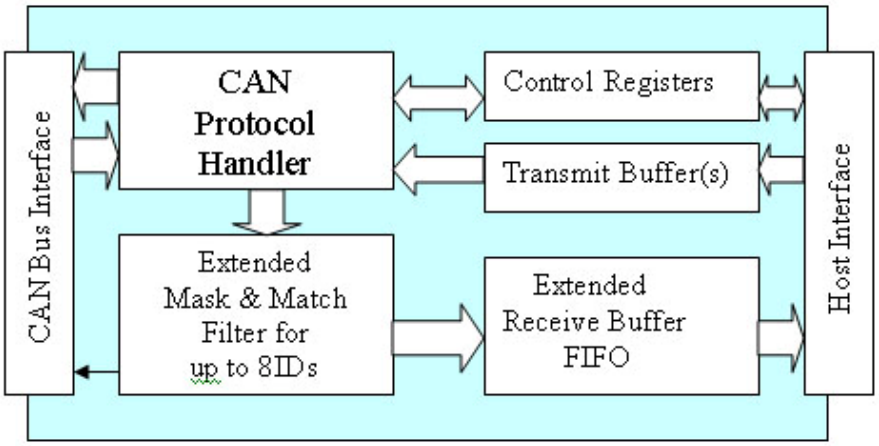


4.2 Block Diagram of a Full CAN Controller

- Υλοποίηση FIFO

Η συγκεκριμένη υλοποίηση είναι ιδανική λύση για απαιτήσεις υψηλής απόδοσης και χρόνου , καθώς χρησιμοποιεί τον FIFO (First-in-first-out) αλγόριθμο . Το σχήμα 4.3 δείχνει τα στάδια έως ότου ένα μήνυμα να εκπεμφθεί . Όπως μπορούμε να δούμε μετά τον controller , υπάρχει ένας μεγάλος αριθμός μασκών και φίλτρων , και αν υπάρχει κάποια ταύτιση φίλτρου , το FIFO buffer ξεκινάει να γεμίζει με μηνύματα , ένα προς ένα . Διακοπή μπορεί να συμβεί στις περιπτώσεις που το buffer είναι γεμάτο ή το υψηλής προτεραιότητας φίλτρο παραλάβει το τελευταίο εισερχόμενο μήνυμα .

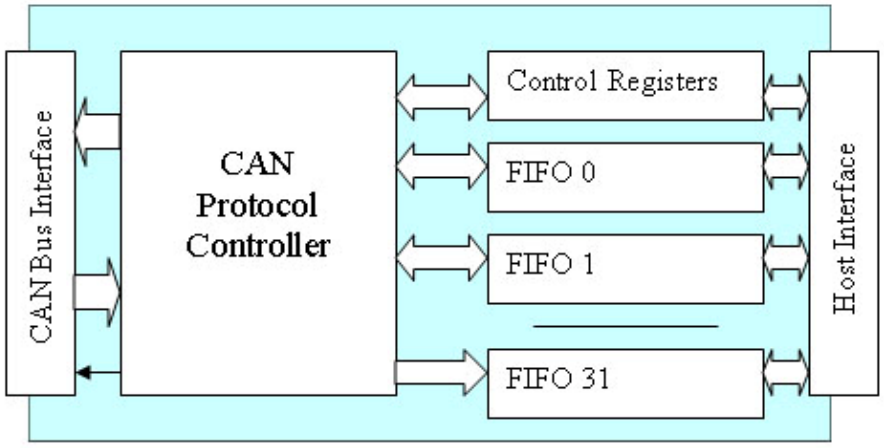
Αν το FIFO είναι σχεδιασμένο να προκαλεί μια διακοπή σε κάθε εισερχόμενο ισότιμο μήνυμα , το MCU (Message Unit Control , μήνυμα ελέγχου) έχει τουλάχιστον 500 bit χρόνο μέχρι να υπερχειλίσει . Αυτό είναι περίπου 10 φορές περισσότερο διαθέσιμο όσο αφορά τον χρόνο του MCU σε σχέση με την βασική και πλήρη CAN υλοποίηση .



4.3 Block diagram of FIFO controller implementation

- Εμπλουτισμένο πλήρες CAN με υλοποίηση FIFO

Η διαφορά με τις παραπάνω υλοποιήσεις είναι ότι χρησιμοποιεί ένα πιο δυναμικό και αποτελεσματικό τρόπο λήψης μηνύματος . Αυτό αποτελείται από τον FIFO για κάθε μήνυμα που λαμβάνεται , όπως βλέπουμε στην υλοποίηση παρακάτω (σχ. 4.4) Είναι η πλέον πολύπλοκη αρχιτεκτονική αλλά και η πιο αποτελεσματική ταυτοχρόνως .



4.4 Block diagram of Philips Enhanced CAN interface(FIFO)

[9]

4.ii. Σύνοψη αρχιτεκτονικής του CAN

Μορφοποιήθηκε: Ελληνικά

Ο CAN controller ο οποίος χρησιμοποιήθηκε σε αυτήν την πτυχιακή ανήκει στην οικογένεια PIC18F και ειδικότερα είναι ο PIC18F4585. Το πλεονέκτημα αυτού του μικροελεγκτή είναι ότι είναι ένας ECAN (Enhanced Controller Area Network) το οποίο σημαίνει ότι περιέχει έναν ενσωματωμένο controller . Είναι πλήρες συμβατό με τα PIC έκδοσης (PIC18C) τα οποία δεν περιέχουν κάποιο ενσωματωμένο controller . [12b]

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Τα βασικά χαρακτηριστικά αυτού του μικροελεγκτή είναι ότι περιέχει τέσσερις διαφορετικές καταστάσεις λειτουργίας του κρυστάλλου του στα 16, 25 , 33 και 40 MHz , ενώ η τάση κάτω από την οποία λειτουργεί είναι μεταξύ 4,2 και 5,5 V και υποστηρίζει 75 εντολές για να λειτουργήσει . Σύμφωνα κάθε φορά με τις συνθήκες (χαμηλότερη κατανάλωση ρεύματος , διόρθωση λαθών) , υποστηρίζει πολλές καταστάσεις λειτουργίας όπως

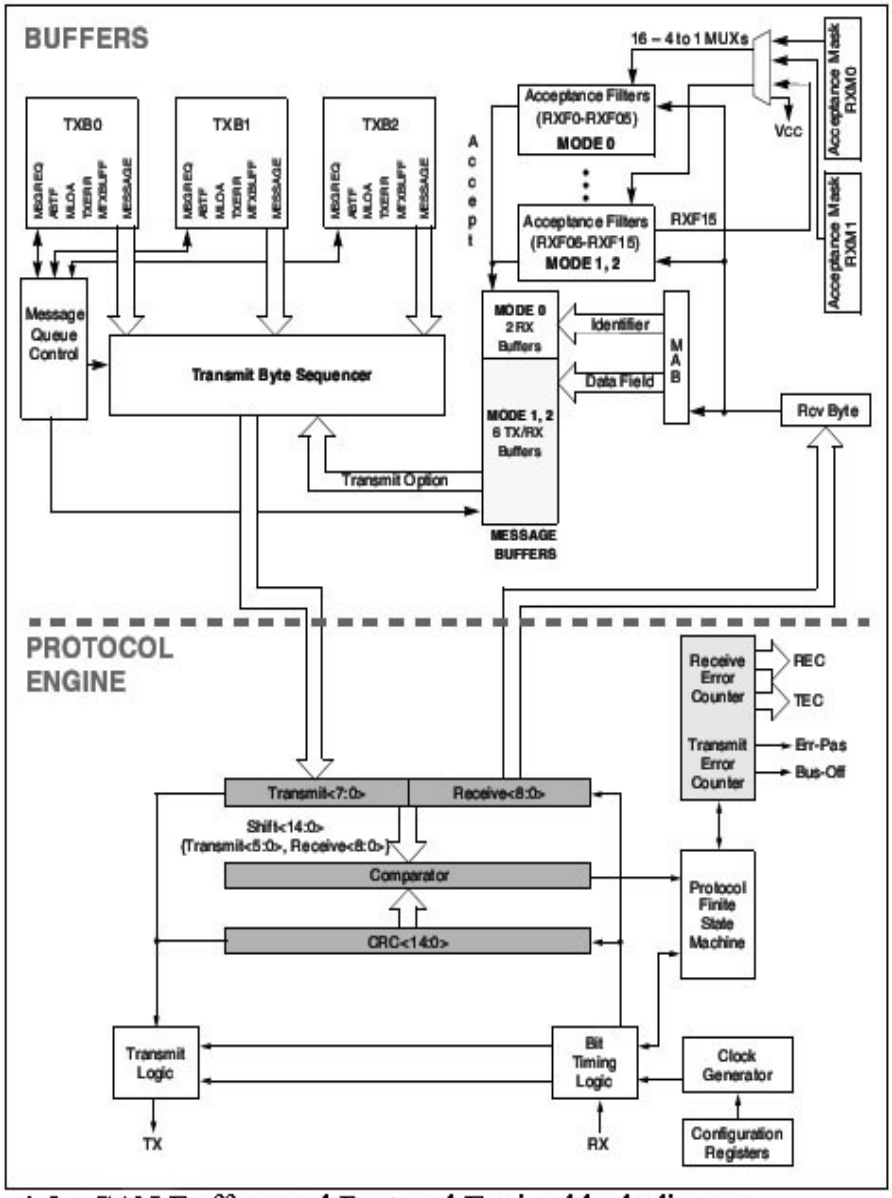
κατάσταση διαμόρφωσης , κανονική κατάσταση , κατάσταση ακρόασης μόνο , κατάσταση επαναλαμβανόμενου βρόγχου και κατάσταση αναγνώρισης λαθών .

Μερικά από τα βασικά χαρακτηριστικά του είναι ότι υποστηρίζει τα πρωτόκολλα CAN 1.2 , 2.0A και 2.0B , υποστηρίζει επίσης βασικά και εκτεταμένα πλαίσια δεδομένων και τέλος έχει τρεις καταστάσεις λειτουργίας . Περιληπτικά είναι η Κατάσταση 0 (Legacy Mode) , Κατάσταση 1 (Enhanced Legacy mode with DeviceNet Support) και Κατάσταση 2 (FIFO mode with DeviceNet Support) .

Τέλος μερικά από τα περιφερικά χαρακτηριστικά είναι ότι έχει τέσσερις διαφορετικούς πολλαπλασιαστές που υποστηρίζουν κοινός SPI και I^2C καταστάσεις καθώς και έναν αναλογικό σε ψηφιακό μετατροπέα 10- bit .

Μορφοποιήθηκε: Ελληνικά

Στην επόμενη σελίδα ακολουθεί ένα σχήμα [4.1] που απεικονίζει τα CAN Buffers και το διάγραμμα που περιέχει τους μηχανισμούς πρωτοκόλλου . Επιπλέον θα υπάρξει και περαιτέρω ανάλυση της κάθε κατάστασης λειτουργίας .



4.5 CAN Buffers and Protocol Engine block diagram

4.iii Καταστάσεις λειτουργίας του ECAN

- Κατάσταση 0 – Legacy Mode

Αυτή είναι η πιο απλή κατάσταση λειτουργίας και είναι η προκαθορισμένη κατάσταση λειτουργίας . Έχοντας την βοήθεια του παραπάνω σχήματος , σε αυτή την κατάσταση 3 buffers που εκπέμπουν υποστηρίζονται (TXB0 ,TXB1 and TXB2) , 2 buffers που λαμβάνουν (RXB0 and RXB1) , 2 μάσκες αποδοχής , μια για κάθε buffer δέκτη (RXM0 and RXM1) και 6 τελικά φίλτρα αποδοχής 2 για το RXB0 (RXF0 και RXF1) και 4 για το RXB1 (RXF2-RXF5) .

Καθένα από τα buffer που εκπέμπουν καταλαμβάνει 14 bytes της SRAM και απεικονίζονται στην SFR μνήμη . Καθένα από τα buffer που εκπέμπουν περιέχει και έναν καταχωρητή ελέγχου (TXBnCON) , 4 καταχωρητές αναγνώρισης (TXBnSIDL,TXBnSIDH, TXBnEIDL, TXBnEIDH) , ένα μετρητή καταχωρητή δεδομένων (TXBnDLC) και 8 καταχωρητές που λειτουργούν στην περίπτωση μη λειτουργίας του FIFO (TXBnDm).

Ακριβώς τα ίδια ισχύουν για τους καταχωρητές λήψης δεδομένων και κάθε buffer καταχωρητή περιέχει έναν καταχωρητή ελέγχου (RXBnCON) , 4 καταχωρητές αναγνώρισης (RXBnSIDL, RXBnSIDH, RXBnEIDL, RXBnEIDH), ένα μετρητή καταχωρητή δεδομένων (RXBnDLC) και 8 καταχωρητές δεδομένων (RXBnDm) .

Όπως μπορούμε να παρατηρήσουμε στο σχήμα παραπάνω υπάρχει ένα προ-Buffer MAB (Message Assembly Buffer) . Το MAB μπορεί να δεχθεί μόνο ένα μήνυμα από τον δίαυλο κάθε φορά και μετά σύμφωνα με το φίλτρο και τον αναγνωριστή μηνύματος, το μήνυμα προωθείται στα επόμενα Buffers ή ματαιώνεται.

- Κατάσταση 1 - Advanced Legacy Mode

Αυτή είναι η προκάτοχος της Κατάστασης 0 , για αυτόν τον λόγο έχουν και κάποιες ομοιότητες . Μπορούμε να πούμε ότι είναι η εκτεταμένη κατάσταση λειτουργίας της Κατάστασης 0 .

Συγκεκριμένα έχει 6 περισσότερα buffers (TX or RX: B0-B5) σε σχέση με την Κατάσταση 0 , που διαθέτει δυναμική λειτουργία και μπορεί να προγραμματιστεί ως buffer εκπομπής ή αποδοχής , 16 δυναμικά buffer αποδοχής (RXF0-RXF15) και 2 καταχωρητές μάσκες αποδοχής . Τα 6 επιπρόσθετα buffers μπορούν να προγραμματιστούν έτσι ώστε να χειρίζονται τα RTR μηνύματα .

Πρέπει να σημειώσουμε ότι οι καταχωρητές buffers εκπομπής-λήψης είναι τα ίδια όπως την Κατάσταση 0 , με τα επιπρόσθετα buffers βέβαια και την MAB λειτουργία επίσης .

- Κατάσταση 2 - Enhanced FIFO mode

Σε αυτή την κατάσταση τουλάχιστον 2 buffers λήψης δεδομένων χρειάζονται για να δημιουργήσουν το FIFO (First In – First Out) buffer . Δεν υπάρχει καμία μια-προς-μια σύνδεση ανάμεσα στα buffer αποδοχής και στους καταχωρητές που περιέχουν φίλτρα αποδοχής . Κάθε φίλτρο το οποίο είναι ενεργό και συνδέεται με κάθε FIFO buffer μπορεί να λειτουργήσει .

Μορφοποιήθηκε: Ελληνικά

Το FIFO δεν έχει κάποιο συγκεκριμένο μήκος , αλλά είναι προ-προγραμματισμένο από τον χρήστη και μπορεί να έχει από 2 μέχρι 8 buffers , από τα οποία το πρώτο θεωρείται ως το buffer εκπομπής δεδομένων .

Εκτός από αυτές τις 2 FIFO λειτουργίες , η Κατάσταση 2 υποστηρίζει κοινές λειτουργίες με τις Καταστάσεις 0 και 1 . [12a]

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

4.iv. Φιλτράρισμα μηνυμάτων

Μορφοποιήθηκε: Ελληνικά

Το φιλτράρισμα μηνυμάτων καθορίζει αν το μήνυμα πρέπει να προωθηθεί στο buffer ή να ματαιωθεί . Όταν ένα μήνυμα φορτώνεται στο MAB , γίνεται σύγκριση ανάμεσα στον αναγνωριστή μηνυμάτων και στις τιμές των φίλτρων . Αν υπάρξει ταύτιση τότε το μήνυμα προωθείται .

Τα φίλτρα-μάσκες χρησιμοποιούνται για να καθορίσουν ποια bit του αναγνωριστή συγκρίνονται κάθε φορά με το φίλτρο .

Ένας πίνακας [4.1] αληθείας ακολουθεί ο οποίος δείχνει πως γίνεται η σύγκριση και με το ανάλογο αποτέλεσμα (αν τελικά το μήνυμα γίνεται αποδεκτό ή όχι)

Mask bit n	**Filter bit n**	**Message Identifier bit n001**	**Accept or Reject bit n**
0	**x**	**X**	**Accept**
1	**0**	**0**	**Accept**
1	**0**	**1**	**Reject**
1	**1**	**0**	**Reject**
1	**1**	**1**	**Accept**

Διαγράφηκε: από

Κάθε κατάσταση υποστηρίζει διαφορετικό αριθμό φίλτρων και μασκών , και πιο συγκεκριμένα στην Κατάσταση 0 το RXB0 buffer σχετίζεται με τα RXF0 και RXF1 φίλτρα και η RXM0 μάσκα και το RXB1 buffer με τα RXF2-RXF5 φίλτρα και τη RXM1 μάσκα . Στις υπόλοιπες καταστάσεις 1 και 2 , υπάρχουν περισσότερα από 10 φίλτρα αποδοχής (RXF6 to RXF15), όπου το τελευταίο μπορεί να χρησιμοποιηθεί είτε για φίλτρο αποδοχής είτε για μάσκα καταχώρησης δεδομένων . [12a]

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Για να γίνει πιο κατανοητή η διαδικασία , τα φίλτρα από το RXF0 έως το RXF5 αναπαρίστανται από τις δυαδικές τιμές 000 έως τη 101 . Για αυτόν ακριβώς τον λόγο οι XOR πύλες χρησιμοποιούνται για την αποδοχή καταχωρητών φίλτρων και η NAND για την αποδοχή καταχωρητών μασκών .

Δίνοντας ένα παράδειγμα:

If the bits of the identifier are	10111100100	
where (1 care bits and 0 don't care bits)		
And the bits of the mask are	11100000000	
The result, identifier and mask	10100000000	
Match with filter	10100000000	
XOR MATCH	00000000000	(ID *AND* MASK)

4.v. Ρυθμίσεις μετάδοσης δεδομένων

Μορφοποιήθηκε: Ελληνικά

Όπως έχει ήδη γραφτεί όλοι οι κόμβοι σε ένα σύστημα CAN δίαυλου πρέπει να έχουν το ίδιον ρυθμό μετάδοσης δεδομένων . Η κωδικοποίηση που χρησιμοποιείται από τα CAN είναι η NRZ , αλλά αυτή η κωδικοποίηση δεν κωδικοποιεί τα δεδομένα σύμφωνα με το ρολόι του κρυστάλλου . Έτσι για να συγχρονίσουμε και να διατηρήσουμε το ρολόι , πρέπει να ληφθεί και από άλλους κόμβους και να συγχρονιστεί με το ρολόι αυτού που εκπέμπει .

Αν θέλουμε να υπολογίσουμε αυτόν τον ρυθμό μετάδοσης δεδομένων , πρέπει πρώτα να υπολογίσουμε τον ονομαστικό ρυθμό μετάδοσης δεδομένων και μετά όλες τις παραμέτρους που περιέχονται σε αυτή την λειτουργία .

Nominal Bit rate = 1 / Nominal Bit time

Nominal Bit time = TQ * (Sync_Seg + Prop_Seg + Phase_Seg1 + Phase_Seg2)

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Οι παράμετροι στις παρενθέσεις έχουν ήδη οριστεί στην παράγραφο [3v1] και στο σχήμα [σχ.3.12] . Το TQ σημαίνει time quant (κβάντο χρόνου) , και είναι μια σταθερή μονάδα η οποία παράγεται από την περίοδο του

ταλαντωτή . Μπορεί να οριστεί από τον προγραμματισμένο ρυθμό μετάδοσης δεδομένων και συχνοτήτων με τιμές από το 1 έως το 256 . Μαθηματικώς μπορούν να υπολογιστούν από τις παρακάτω εξισώσεις .

Διαγράφηκε: ό

$$TQ\ (\mu s) = (2 * (BRP + 1))/Fosc\ (MHz)$$
or
$$TQ\ (\mu s) = (2 * (BRP + 1)) * Tosc\ (\mu s)$$

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Όπου Fosc και Tosc η συχνότητα και η περίοδος του ταλαντωτή αντιστοίχως , και BPR (Baud Rate Prescaler) ένας ακέραιος με τιμές από το 0 έως το 255 . [12a]

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

ΚΕΦΑΛΑΙΟ 5

Αρχές Προσομοίωσης και Αποτελέσματα

5.i. Διαφορετικές CAN κατασκευές

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Ένα Controller Area Network υλοποιείται αν 2 ή περισσότεροι κόμβοι συνδέονται πάνω σε ένα CAN δίαυλο . Υπάρχει η δυνατότητα της επιλογής ανάμεσα σε τρεις διαφορετικές αρχιτεκτονικές για να υλοποιήσουμε το δίκτυο .

Η πρώτη δυνατή CAN υλοποίηση αποτελείτε από 2 ή παραπάνω ανεξάρτητους κόμβους . Στην περίπτωση αυτή ένας κόμβος σχηματίζεται αν χρησιμοποιήσουμε έναν κλασικό μικροελεγκτή (της οικογένειας PIC16F ή PIC16C) και έναν ανεξάρτητο CAN Controller (MCP 2515, από την microchip) με τον οποίο πρόκειται να συνδεθούν αν χρησιμοποιήσουν το SPI πρωτόκολλο , και τελικά τον πομποδέκτη ο οποίος στέλνει και δέχεται σήματα στον δίαυλο . Το πλεονέκτημα αυτής της αρχιτεκτονικής είναι ότι είναι αρκετά φτηνή και εξαρτώμενη από την εφαρμογή(υπάρχει μια ποικιλία από μικροελεγκτές που είναι ιδανικοί σε κάθε περίπτωση) . Το μειονέκτημα είναι ότι θα μπορούσε να χρησιμοποιηθεί ένας εξωτερικός CAN Controller ο οποίος και πρόκειται να χρησιμοποιήσει διαφορετική διασύνδεση σύμφωνα με τα διάφορα ήδη εφαρμογών .

Διαγράφηκε: α

Διαγράφηκε: ε

Διαγράφηκε: ή

Διαγράφηκε: τα

Η δεύτερη λύση είναι η χρησιμοποίηση ενός ολοκληρωμένου CAN Controller , όπως της οικογένειας PIC18F με έναν ενσωματωμένο CAN Controller . Η έκδοση αυτή είναι γενικά πιο αποτελεσματική επειδή χρησιμοποιούνται μόνο δύο μονάδες και η πρόσβαση στα CAN περιφερικά (αισθητήρες στην περίπτωση της πτυχιακής) γίνεται γρηγορότερα . Τα μειονεκτήματα που είναι πιθανό να αντιμετωπίσουμε είναι ότι μπορεί να μην υπάρχει συμβατότητα με την εφαρμογή και επίσης το κόστος είναι αυξημένο σε σχέση με την πρώτη αρχιτεκτονική .

Διαγράφηκε: μόνο δύο μονάδες

Η τελευταία λύση είναι η χρήση ενός πομποδέκτη και αντί του μικροελεγκτή ενός CAN I/O Expander . Αυτή η λύση έχει το πλεονέκτημα ότι δεν χρειάζεται προγραμματισμό καθώς όλες οι παράμετροι καθορίζονται από ένα γραφικό περιβάλλον . Από την άλλη μεριά

Διαγράφηκε: με την

Διαγράφηκε: η

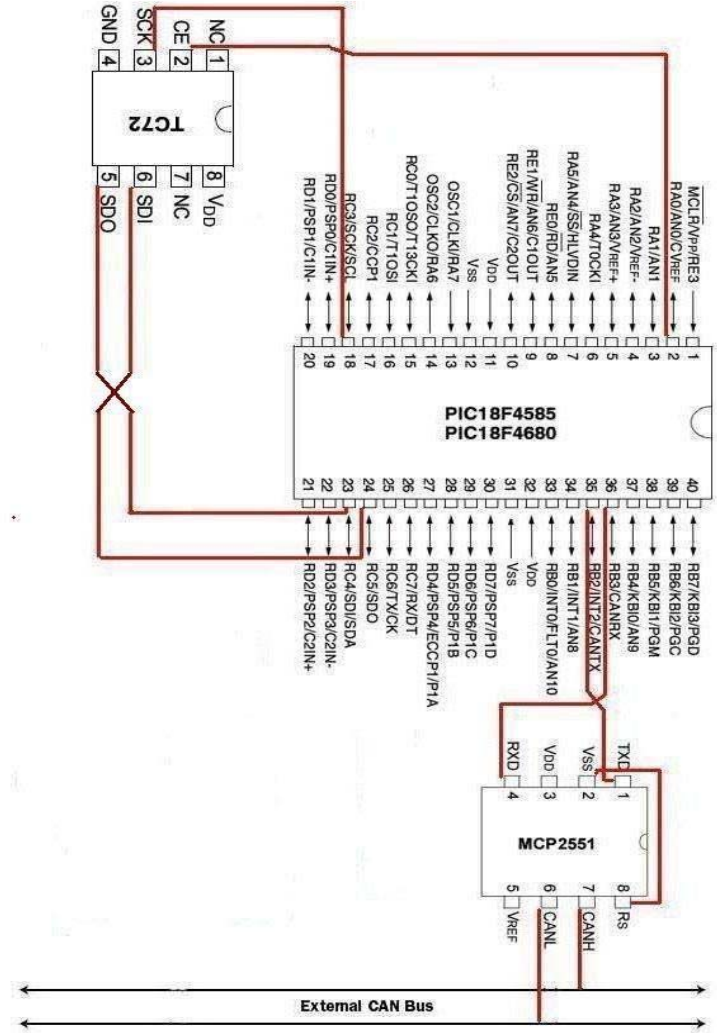
Διαγράφηκε: μοποίηση

υπάρχουν πολλά μειονεκτήματα ένα από τα οποία είναι το υψηλό κόστος (όχι τόσο ο expander , αλλά το απαραίτητο kit έτσι ώστε να προγραμματιστεί) και επίσης το μηχάνημα παρουσιάζει πολλά τεχνικά προβλήματα κατά την χρήση του και είναι γενικά αμφίβολο αν λειτουργήσει πλήρως καθώς δεν έχει τεσταριστεί σε όλα τα επίπεδα . [12c] .

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Συμπερασματικά η ιδανική επιλογή για το δίκτυο αισθητήρων που θέλω να υλοποιήσω είναι η δεύτερη επιλογή με την χρησιμοποίηση ενός ενσωματωμένου CAN . Η υλοποίηση ενός κόμβου παρουσιάζεται στο σχήμα [σχ.5.1] παρακάτω .



5.1 A typical CAN node with a sensor

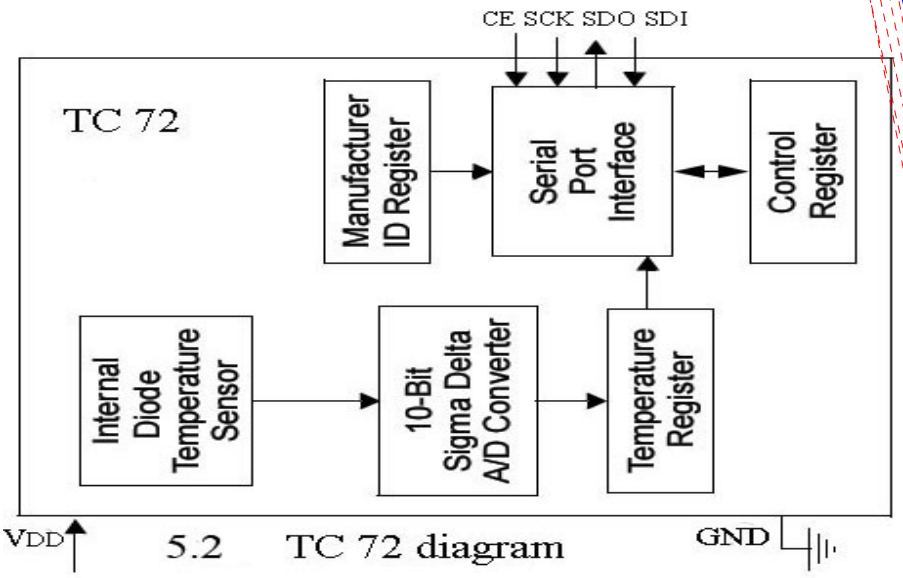
5.ii. Σύνθεση του CAN

Για την υλοποίηση του δικτύου αισθητήρων χρησιμοποιήθηκαν τρία βασικά στοιχεία. Πρώτα από όλα ένας ψηφιακός αισθητήρας θερμοκρασίας (TC72 από την microchip) , το πιο βασικό στοιχείο που είναι ο ECAN μικροελεγκτής (PIC18F4585 από την microchip) και τέλος ο πομποδέκτης που είναι το τελευταίο μέσο επικοινωνίας μεταξύ του κόμβου και του διαύλου . Ο μικροελεγκτής προγραμματίστηκε με την χρήση του ICD (In- Circuit Debugger/Programmer) ο οποίος και υποστηρίζει τεχνολογία USB .

Το φυσικό μέσο του δικτύου αποτελείται από ένας ζεύγος στρεφόμενων καλωδίων και δύο αντιστάσεις 120 Ohm .

- Αισθητήρας TC 72

Ο κύριος λόγος της επιλογής αυτού του αισθητήρα είναι ότι υποστηρίζει το SPI πρωτόκολλο και είναι συμβατός με τον μικροελεγκτή . Επιπλέον χαρακτηριστικό είναι ότι λειτουργεί ως slave συσκευή ενώ είναι και ψηφιακός αισθητήρας καθώς περιέχει A/D μετατροπέα . Τέλος είναι αρκετά φτηνός και έχει πολύ καλή ανάλυση (10-bit ανάλυση και 0.25°C/Bit) και ακρίβεια .



5.2 TC 72 diagram

Επιπλέον υποστηρίζει δύο καταστάσεις λειτουργίας , οι οποίες και έχουν διαφορετική κατανάλωση ισχύος (κατάσταση συνεχόμενης μετατροπής – 250μA και κλειστή λειτουργία με κατανάλωση 1μA) . Η πρώτη λειτουργία παίρνει μετρήσεις σχεδόν κάθε 150ms ενώ η δεύτερη λειτουργία παίρνει μια μέτρηση και μετά επιστρέφει στην δυναμική λειτουργία εξοικονόμησης ενέργειας .

Όπως βλέπουμε στο σχήμα 5.1 παραπάνω , το σειριακό interface αποτελείται από 4 input/output . Το CE(Chip Enable) χρησιμοποιείται για την επιλογή συσκευής όταν περισσότερες από μια συσκευές βρίσκονται στο δίκτυο . Το SCK είναι το σειριακό ρολόι , το οποίο και καθορίζεται από έναν εξωτερικό μικροελεγκτή , και χρησιμοποιείται για τον συγχρονισμό των εξωτερικών και εσωτερικών δεδομένων . Τα δύο τελευταία interfaces SDI και SDO αντιστοιχούν στα δεδομένα που εισέρχονται και εξέρχονται . Πιο συγκεκριμένα το SDI αποθηκεύει δεδομένα στους καταχωρητές του αισθητήρα ενώ το SDO παράγει στην έξοδο δεδομένα από τον αισθητήρα στον μικροελεγκτή .

Τέλος πρέπει να σημειώσουμε ότι οι ακέραιες τιμές του αισθητήρα αναπαρίστανται από το MSB (Most Significant Bits) ενώ οι δεκαδικές τιμές από τα LSB (Least Significant Bits) των καταχωρητών .

Σύμφωνα με υποδείξεις του κατασκευαστή απαραίτητος είναι ένας πυκνωτής των 0.01 ή 0.1 μF στην έξοδο του Vdd pin , έτσι ώστε να λειτουργεί χωρίς προβλήματα . [12d]

- PIC18F4585

Έχουμε κάνει ήδη αναφορά στις βασικές αρχές αυτού του μικροελεγκτή στα προηγούμενα κεφάλαια και τα πλεονεκτήματα που έχει ο ECAN μικροελεγκτής .

- MCP2551 πομποδέκτης

Ο πομποδέκτης που έχει επιλεχθεί υποστηρίζει εύρος ζώνης μέχρι 1Mbit/sec κατά τη λειτουργία του και μπορεί επίσης να υλοποιήσει το ISO-11989 standard ως μικροελεγκτής .

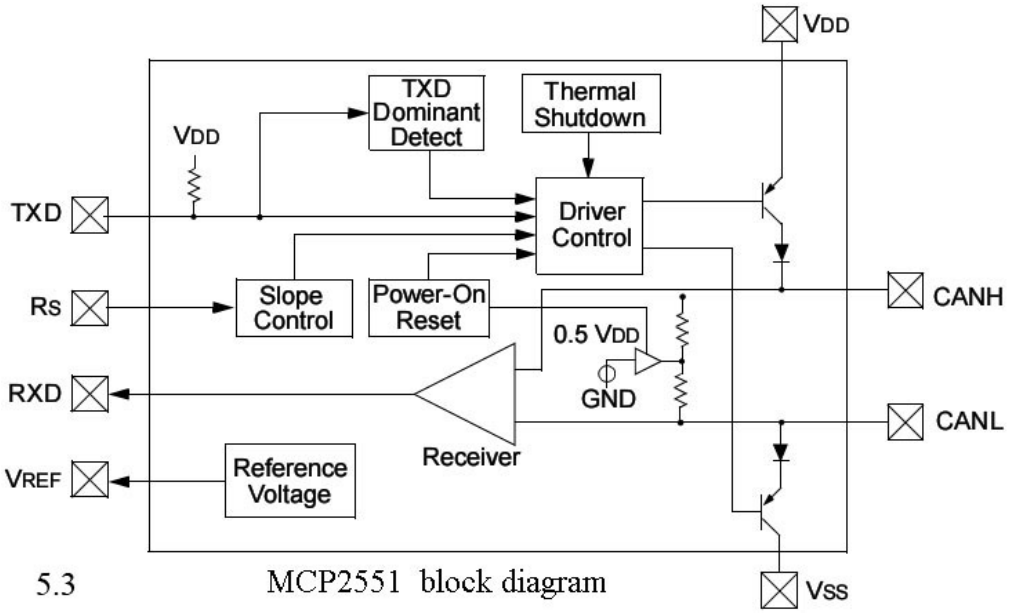
Ο ρόλος του πομποδέκτη είναι να μετατρέπει το ψηφιακό σήμα το οποίο εκπέμπεται (CANTX) ή παραλαμβάνεται (CANRX) από τον CAN Controller σε μια διαφορική έξοδο CANH-CANL η οποία είναι αναγνωρίσιμη από το δίκτυο . Όταν η διαφορά CANH-CANL η οποία παράγεται στην έξοδο είναι στην κυρίαρχη κατάσταση , όταν η τάση είναι περίπου στα 2.5V , ενώ στην περίπτωση που είμαστε στην υπολειπόμενη κατάσταση η διαφορά αυτή είναι σχεδόν στα 0V .

Όπως βλέπουμε στο σχήμα 5.2 παρακάτω , υπάρχει ένας τριφασικός buffer-δέκτης και η RXD έξοδος δημιουργείται κοινός από το CANH και CANL .

Οι τρεις καταστάσεις λειτουργίας καθορίζονται από την συνδεσμολογία του Rs Pin .

Η κατάσταση υψηλής μετάδοσης δεδομένων (high speed mode) σχηματίζεται αν συνδέσουμε το Rs Pin στο Vss Pin . Το χαρακτηριστικό αυτής της κατάστασης είναι η υψηλής συχνότητας πτώσεις και ανυψώσεις του σήματος για να μπορέσει να υποστηρίξει τους υψηλούς ρυθμούς μετάδοσης δεδομένων. Η κατάσταση ελέγχου κλίσης (slope control mode) σχηματίζεται αν συνδέσουμε μια αντίσταση στην έξοδο του Rs Pin . (Η καμπύλη για την κατάσταση ελέγχου κλίσης , για διάφορες τιμές αντιστάσεων , συνοδεύεται στο datasheet του chip) . Ο λόγος για τον οποίο τοποθετούμε την αντίσταση είναι για μπορέσουμε να ελέγξουμε τον λόγο κλίσης (slope rate) , δηλαδή τις ανυψώσεις και τις πτώσεις των CANH και CANL στην μονάδα του χρόνου . Τέλος η κατάσταση αναμονής (standby mode) υφίσταται για καλύτερο έλεγχο της κατανάλωσης ενέργειας , ο πομπός μεταβαίνει σε κατάσταση αδράνειας και ο δέκτης λειτουργεί σε κατάσταση χαμηλής κλίσης (low slope mode) . [12 e]

Μορφοποιήθηκε ... [1]
Μορφοποιήθηκε ... [2]
Μορφοποιήθηκε ... [3]
Μορφοποιήθηκε ... [4]
Μορφοποιήθηκε ... [5]
Μορφοποιήθηκε ... [6]
Μορφοποιήθηκε ... [7]
Μορφοποιήθηκε ... [8]
Μορφοποιήθηκε ... [9]
Μορφοποιήθηκε ... [10]
Μορφοποιήθηκε ... [11]
Μορφοποιήθηκε ... [12]
Μορφοποιήθηκε ... [13]
Μορφοποιήθηκε ... [14]
Μορφοποιήθηκε ... [15]
Μορφοποιήθηκε ... [16]
Μορφοποιήθηκε ... [17]
Μορφοποιήθηκε ... [18]
Μορφοποιήθηκε ... [19]
Μορφοποιήθηκε ... [20]
Διαγράφηκε: ε
Μορφοποιήθηκε ... [21]
Μορφοποιήθηκε ... [22]
Μορφοποιήθηκε ... [23]
Μορφοποιήθηκε ... [24]
Μορφοποιήθηκε ... [25]
Μορφοποιήθηκε ... [26]
Μορφοποιήθηκε ... [27]
Μορφοποιήθηκε ... [28]
Μορφοποιήθηκε ... [29]
Μορφοποιήθηκε ... [30]
Μορφοποιήθηκε ... [31]
Μορφοποιήθηκε ... [32]
Μορφοποιήθηκε ... [33]
Μορφοποιήθηκε ... [34]
Μορφοποιήθηκε ... [35]
Μορφοποιήθηκε ... [36]
Μορφοποιήθηκε ... [37]
Μορφοποιήθηκε ... [38]
Μορφοποιήθηκε ... [39]
Μορφοποιήθηκε ... [40]
Μορφοποιήθηκε ... [41]
Μορφοποιήθηκε ... [42]
Μορφοποιήθηκε ... [43]
Μορφοποιήθηκε ... [44]
Μορφοποιήθηκε ... [45]
Μορφοποιήθηκε ... [46]
Μορφοποιήθηκε ... [47]
Μορφοποιήθηκε ... [48]
Μορφοποιήθηκε ... [49]
Μορφοποιήθηκε ... [50]



5.3 MCP2551 block diagram

5.iii. Στάδια υλοποίησης

Το πρώτο στάδιο υλοποίησης περιλαμβάνει την δημιουργία ενός Controller Area Network με τον μικρότερο δυνατό αριθμό κόμβων . Οι δύο κόμβοι είναι ο μικρότερος δυνατός αριθμός για να έχουμε ένα CAN , και προφανώς είναι η απλούστερη υλοποίηση . Ο ένας κόμβος ο οποίος και θα έχει τον αισθητήρα θερμοκρασίας θα εκπέμπει τα δεδομένα στον CAN δίαυλο , ενώ ο άλλος κόμβος θα δέχεται τα δεδομένα , θα τα αποθηκεύει στον μικροελεγκτή και τέλος η θερμοκρασία είναι δυνατό να φαίνεται στο pc μέσω ενός λογισμικού απεικόνισης .

Η δεύτερη και πιο πολύπλοκη υλοποίηση προσομοιάζει ένα σύστημα κεντρικής θέρμανσης με την χρησιμοποίηση δύο αισθητήρων θερμοκρασίας και τριών κόμβων συνολικά [σχ. 5.4] . Ο κόμβος με τον αισθητήρα θερμοκρασίας πρόκειται να παίρνει μετρήσεις από το εξωτερικό περιβάλλον , ένας άλλος κόμβος με αισθητήρα πρόκειται να παίρνει μετρήσεις από το εσωτερικό περιβάλλον , ενώ τέλος ένας τρίτος κόμβος που θα δέχεται όλες αυτές τις μετρήσεις-σήματα και θα τα μεταφέρει στην οθόνη ενός υπολογιστή όπου θα αναπαρίστανται και την ίδια στιγμή το "κεντρικό σύστημα θέρμανσης" θα ανοίγει ή ένα LED .

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια, Ελληνικά

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια, Ελληνικά

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

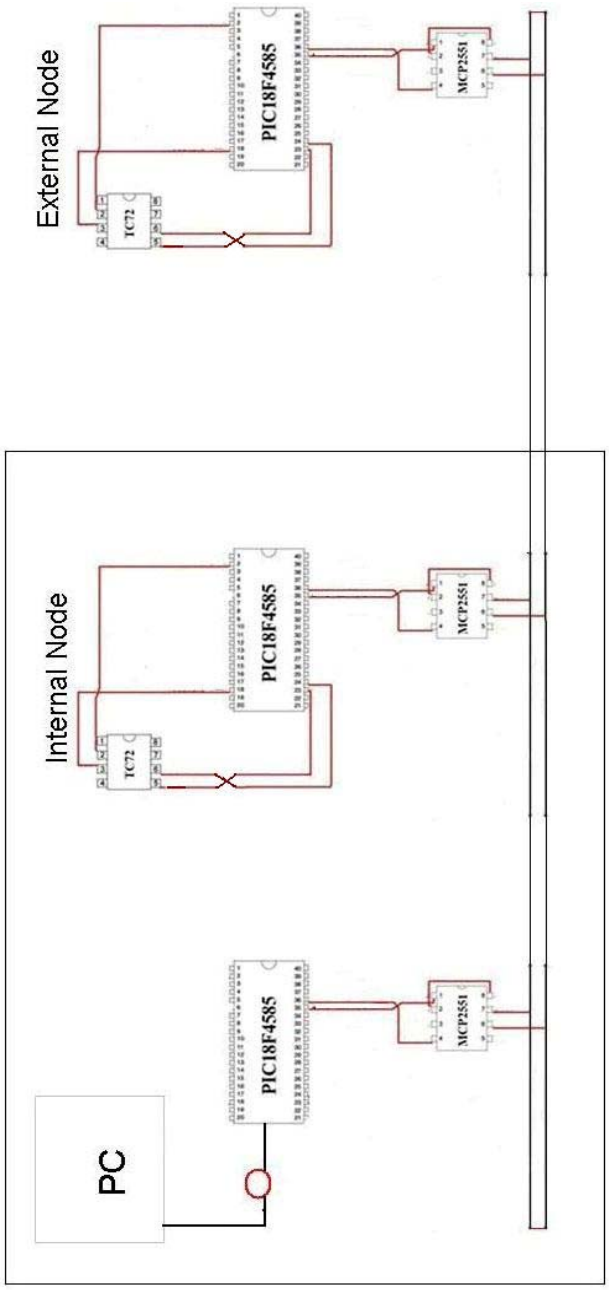
Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά



5.4 CAN sensing network with 3 nodes , 2 sensors

Μορφοποιήθηκε: Ελληνικά

Η διαδικασία που ακολουθείται για να ανοίξει το "κεντρικό σύστημα θέρμανσης" είναι η ακόλουθη :

- Ο εξωτερικός κόμβος έχει προληπτική λειτουργία . Υπολογίζει την μέση τιμή των τιμών που έχει μετρήσει για ένα συγκεκριμένο χρονικό διάστημα , και συγκρίνει αυτές τις τιμές με κάποια σταθερά τιμή " threshold " , που έχουμε προκαθορίσει . Με άλλα λόγια ο κόμβος αυτός λειτουργεί με την λογική να προβλέπει και να ανοίγει το "σύστημα θέρμανσης" προτού ο εσωτερικός χώρος προλάβει να κρυώσει .
- Ο εσωτερικός κόμβος , ο οποίος είναι ανεξάρτητος από τον εξωτερικό , έχει ένα απλό "θερμοστάτη" και όταν η θερμοκρασία πέσει κάτω από ένα επίπεδο το σύστημα ανοίγει . Πιο συγκεκριμένα οι τιμές που μετρώνται από την αντίσταση προωθούνται στον μικροελεγκτή και πάντα συγκρίνονται με μια προκαθορισμένη τιμή . Όταν η τιμή αυτή γίνει μικρότερη της τιμής που έχουμε υπολογίσει , ένα σήμα στέλνεται στον κεντρικό επεξεργαστή και το λαμπάκι που αντιστοιχεί στην "κεντρική θέρμανση" , ανάβει .
- Υπάρχει επίσης ένας επιπλέον εσωτερικός κόμβος που λειτουργεί ως δέκτης πλαισίων από τους άλλους δύο κόμβους και αναπαριστά την τρέχουσα θερμοκρασία στην οθόνη ενός υπολογιστή έτσι ώστε να ανοίγει το LED , εξαρτώντας βέβαια κάθε φορά από την ένδειξη .

5.iv Λεπτομερής παρουσίαση των προγραμματιστικών τεχνικών

Το προγραμματιστικό εργαλείο που χρησιμοποιήθηκε για την μεταγλώτιση (compiling) του κώδικα είναι το γνωστό MPLAB της Microchip , το οποίο είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης για προγραμματισμό της συγκεκριμένης μάρκας μικροελεγκτών . Η γλώσσα προγραμματισμού που χρησιμοποιήθηκε είναι η C , ενώ υπήρχε και ένα τμήμα κώδικα – το οποίο αφορούσε την SPI επικοινωνία – το οποίο και γράφτηκε σε assembly .

Ένα επιπλέον λογισμικό το οποίο χρησιμοποιήθηκε ανήκει και αυτό στην οικογένεια της Microchip και είναι το Microchip Application

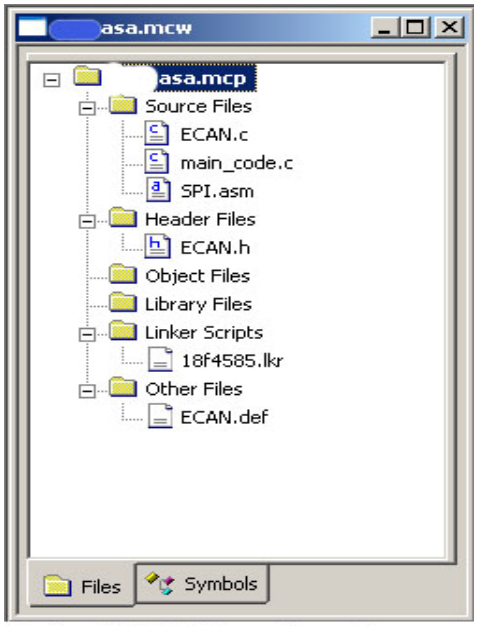
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Κουκκίδα + Επίπεδο: 1 + Στοίχιση: 54 στ. + Στηλοθέτης μετά: 72 στ. + Εσοχή: 72 στ.
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια
Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια, Ελληνικά
Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά

Maestro . Το συγκεκριμένο πρόγραμμα χρησιμοποιείται για τον σχηματισμό του .def αρχείου , το οποίο και είναι απαραίτητο για την μεταγλώττιση (compiling) του C κώδικα και τέλος περιέχει παραμέτρους όπως το ρολόι του μικροελεγκτή , τον αριθμό των buffers κ.α. .

Το τελευταίο λογισμικό που χρησιμοποιήθηκε είναι το Labview η λειτουργία του οποίου ήταν η αναπαράσταση των τιμών της θερμοκρασίας , στην οθόνη του υπολογιστή .

Φορτώνοντας τα αρχεία στο λογισμικό

Στο παρακάτω σχήμα [σχ.5.5] υπάρχει δεντρικό διάγραμμα με τα απαραίτητα αρχεία τα οποία πρέπει να φορτωθούν στον MPLAB Compiler . Στα πηγαία αρχεία (source files) πρέπει να φορτωθεί το main_code.c το οποίο περιέχει τον κώδικα του CAN Controller όπως έχει ήδη αναφερθεί , το SPI.asm ο οποίος είναι κώδικας σε assembly σύμφωνα με τον οποίο ορίζεται και η SPI επικοινωνία ανάμεσα στον μικροελεγκτή και τον αισθητήρα και τέλος το ECAN.c αρχείο το οποίο είναι ένα αρχείο c κώδικα το οποίο και περιέχει μερικές συναρτήσεις σε c κώδικα απαραίτητα για να λειτουργήσει σωστά ο main κώδικας .



5.5 MPLAB configuration

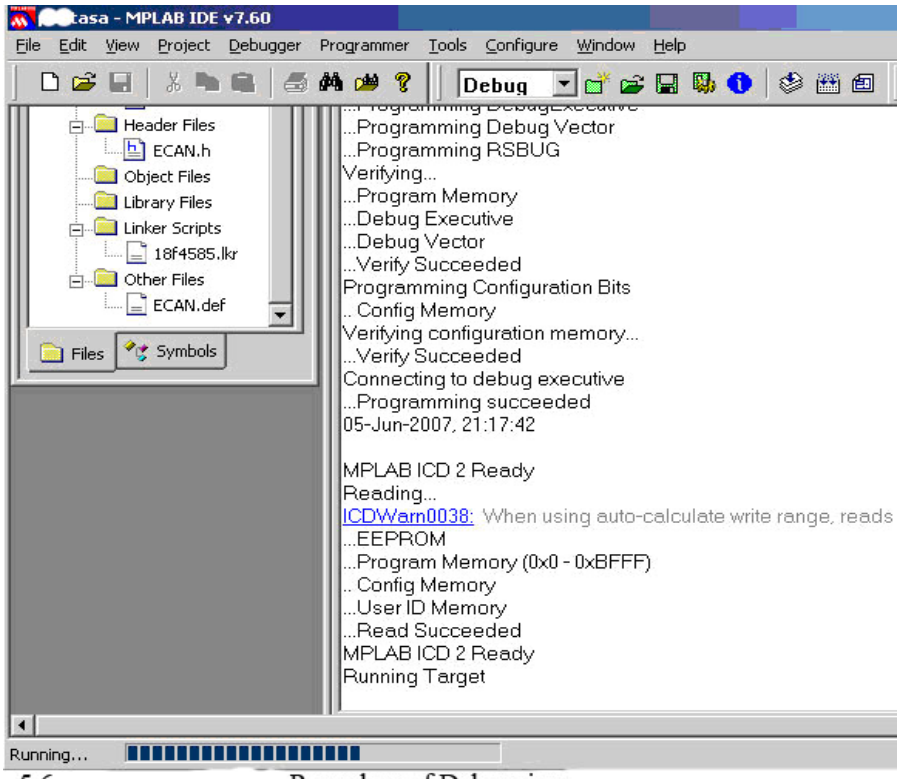
Το αρχείο επικεφαλίδων (header file) , ECAN.h , το οποίο περιλαμβάνει συναρτήσεις , δήλωση δεδομένων και ορίζει οτιδήποτε από αυτά χρειάζεται στον κώδικα .

Ο κώδικας διασύνδεσης , ο οποίος περιλαμβάνει το αρχείο 18F4585.lkr , παρέχει πληροφορίες για την συσκευή η οποία χρησιμοποιείται .

Τέλος το ECAN.def αρχείο για το οποίο έχει ήδη γίνει ανάλυση και από την εφαρμογή Maestro , με την δήλωση της κατάστασης λειτουργίας , των αριθμό των buffers , των φίλτρων και της μάσκας , όπως και της συχνότητας του ταλαντωτή .

Debugger – Programmer

Ο κώδικας αρχικά φορτώνεται στον μικροελεγκτή μέσω του debugger και εν συνεχεία τεστάρεται με την βοήθεια του ιδίου του εργαλείου του MPLAB. Όλα αυτά συμβαίνουν μόνο σε πραγματικές συνθήκες λειτουργίας .



5.6 Procedure of Debugging

Ανάλυση του κώδικα

Στο CAN δίκτυο υπάρχουν συνολικά τρεις κόμβοι δύο εκ των οποίων περιέχουν τους αισθητήρες θερμοκρασίας , εκπέμπουν τα σήματα-πλαίσια στον δίαυλο και υπάρχει επίσης και άλλος ένας κόμβος που δέχεται αυτά τα πλαίσια από τους υπόλοιπους δύο μικροελεγκτές .

- Ο πρώτος κώδικας που πρόκειται να αναλυθεί είναι ο κώδικας του εσωτερικού κόμβου με τον αισθητήρα , ο οποίος στέλνει πλαίσια στο δίκτυο . Η prescaler τιμή ορίζεται στα 128 , η οποία με τη βοήθεια της συνάρτησης που βρίσκεται στο κεφάλαιο [4iiib] .

TQ (μs) = (2 * (BRP + 1))/Fosc (MHz)
= (2 * (128 + 1))/ 16
= 16,125μs

Όταν επιτευχθεί το κατώφλι θερμοκρασίας (15^0C) , θα συμβεί διακοπή στο τέλος του κύκλου , και τα δεδομένα θα σταλούν μέσω του διαύλου .

Όλα τα πλαίσια του διαύλου πρέπει να έχουν διαφορετικό ID , για αυτό το λόγο καλείτε συνάρτηση που παράγει αυξανόμενα ID και η οποία ξεκινάει από το μηδέν μέχρι μια μεγάλη τιμή .

- Ο δεύτερος κόμβος ο οποίος εκπέμπει , και που έχει τον ίδιο ρυθμό ρολογιού με τον παραπάνω κόμβο , για τον λόγο ότι όλοι οι κόμβοι σε ένα Controller Area Network πρέπει να είναι συγχρονισμένα κάτω από την ίδια συχνότητα . Σε αυτή την περίπτωση τα ID's τα οποία δημιουργούνται ξεκινάνε από έναν μεγάλο long ακέραιο και αρχίζουν να μειώνονται . Ο εξωτερικός κόμβος τότε μαζεύει τις τιμές των bytes ,τις οποίες λαμβάνει από τον αισθητήρα , σε έναν πίνακα . Όταν γεμίσει αυτός ο πίνακας , υπολογίζεται η μέση τιμή των τιμών αυτών και συγκρίνεται με ένα προ-ορισμένο κατώφλι θερμοκρασίας (10^0C) . Όταν αυτή η τιμή φτάσει το κατώφλι θερμοκρασίας , τότε δημιουργείται διακοπή του κύκλου και το μήνυμα αποστέλλεται στο δίκτυο .

Μορφοποιήθηκε: Γραμματοσειρά: Πλάγια
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)
Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)
Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)
Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)
Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Διαγράφηκε: η
Μορφοποιήθηκε: Ελληνικά
Διαγράφηκε: αυτή

- Ο τρίτος κόμβος ο οποίος δέχεται τα δεδομένα από τους υπολοίπους κόμβους έχει τον κατάλληλο κώδικα αποδοχής . Με λίγα λόγια ο κώδικας του δέκτη είναι μια μίξη των δύο παραπάνω κωδίκων , ενώ για την αναγνώριση των μηνυμάτων πρέπει να έχουν τον ίδιο ID αλγόριθμο . Όταν συμβεί μια διακοπή από κάποιον από τους κόμβους τότε ένα πλαίσιο δεδομένων πρόκειται να σταλεί μέσω του διαύλου , και όταν κόμβος το λάβει , το LED θα ανάψει .

- Ανάλυση SPI κώδικα : Ο SPI κώδικας χρησιμοποιείται για την διασύνδεση του μικροελεγκτή με τον αισθητήρα θερμοκρασίας . Είναι μια σειριακή πραγματικού χρόνου επικοινωνία μέσω πρωτοκόλλου για ανταλλαγή δεδομένων . Ο μικροελεγκτής θεωρείται ως μια master συσκευή ενώ ο αισθητήρας ως μια slave .

Το είδος της μεταφοράς είναι μια multi-byte μεταφορά , που όταν η επιλογή του chip είναι στο high τότε στέλνονται τα bits διαμόρφωσης τα οποία και διαμορφώνουν την master συσκευή . Η φάση και η πολικότητα του ρολογιού είναι ρυθμισμένα σε ((Mode 0,1) CKE=CKP=0) όπως και ο ρυθμός bit από Fosc/16 έως και Fosc/4 . Τρία bytes συλλέγονται από τον αισθητήρα και αποθηκεύονται προσωρινά στα buffers . Το πρώτο byte είναι το byte ελέγχου το οποίο και δηλώνει την λήψη δύο επιπλέον bytes τα οποία και αναπαριστάνουν τις τιμές της θερμοκρασίας .

Γλώσσες προγραμματισμού

Οι γλώσσες προγραμματισμού που χρησιμοποιήθηκαν για την ολοκλήρωση αυτής της εργασίας είναι μια μίξη της C και της assembly . Η γλώσσα C χρησιμοποιήθηκε για την αρχικοποίηση του συστήματος και των δεδομένων , για τον μέρος που περιλαμβάνει τον έλεγχο του συστήματος , αποστολή και πρόσληψη μηνυμάτων στον δίαυλο CAN ενώ η assembly χρησιμοποιήθηκε για την SPI επικοινωνία μεταξύ των αισθητήρων και των μικροελεγκτών .

Η ένωση αυτών των δύο κωδίκων συνέβη με τον ορισμό μια εξωτερικής global συνάρτησης σε C κώδικα ενώ η κλήση έγινε μέσω της assembly .

ΚΕΦΑΛΑΙΟ 6

Περίληψη

6i Μελλοντική εργασία

Όπως έχουμε ήδη αντιληφθεί το CAN είναι ιδιαίτερα ενδιαφέρον και αποτελεσματικό πρωτόκολλο και για αυτό τον λόγο είναι εξαπλωμένο σε τόσες εφαρμογές , με μεγάλη μάλιστα επιτυχία . Αν θεωρήσουμε αυτή την πτυχιακή σαν ένα πρώτο βήμα τότε η μελλοντική δουλειά μπορεί να έχει ως στόχο τρεις διαφορετικές κατευθύνσεις .

Η πρώτη κατεύθυνση θα μπορούσε να είναι η χρησιμοποίηση εντελώς διαφορετικών διεργασιών ανάλογα με τις ανάγκες και τις απαιτήσεις , έτσι ώστε να γίνεται με διαφορετικούς τρόπους η διαχείριση της θερμοκρασίας . Ένα παράδειγμα θα μπορούσε να είναι ένας PID Controller για την διαχείριση της θερμοκρασίας σε ένα δωμάτιο ή σε ένα ολόκληρο κτίριο . Ένα σύστημα όπως αυτό που περιγράφηκε στην παράγραφο [5iii] , με έναν εσωτερικό και έναν εξωτερικό κόμβο , όπως και έναν κόμβο ελέγχου της θερμοκρασίας θα μπορούσε να διασυνδεθεί με ένα σύστημα θέρμανσης το οποίο προσαρμόζεται αναλογικά έχοντας διάφορα επίπεδα θερμοκρασίας . Σε ένα τέτοιο σύστημα , ο εσωτερικός κόμβος που έχει τον ρόλο του θερμοστάτη , μπορεί να υλοποιήσει τον PID (Proportional-integral-Derivative) αλγόριθμο . Ο PID αλγόριθμος περιλαμβάνει τρεις διαφορετικές καταστάσεις λειτουργίας , την αναλογική κατάσταση , την ολοκληρωμένη κατάσταση και την παράγωγη κατάσταση . Η αναλογική κατάσταση καθορίζει τις αντιδράσεις στα τρέχουσα λάθη , η ολοκληρωμένη κατάσταση καθορίζει τις αντιδράσεις στα προηγούμενα λάθη ενώ η παραγωγική κατάσταση καθορίζει τις αντιδράσεις στις οποίες έχουμε αλλαγή των λαθών . Το άθροισμα αυτών των τριών καταστάσεων μπορεί να χρησιμοποιηθεί για τον έλεγχο ενός μεταβλητού συστήματος θέρμανσης με αυξανόμενη αποδοτικότητα .

Η δεύτερη κατεύθυνση είναι η βελτίωση του κώδικα με την χρησιμοποίηση φίλτρων και μασκών . Αυτό κάνει μεν τον κώδικα κάπως πιο σύνθετο αλλά από την άλλη αυξάνει την λειτουργία ελέγχου τον λαμβανόμενων μηνυμάτων . Σε αυτή την περίπτωση διαφορετικές καταστάσεις λειτουργίας πρέπει να χρησιμοποιηθούν .

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Μορφοποιήθηκε: Στοιχισμένο στο κέντρο

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Στηλοθέτες: 206,25 στ., Αριστερά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Διαγράφηκε: ς

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Μορφοποιήθηκε: Ελληνικά

Διαγράφηκε: ν

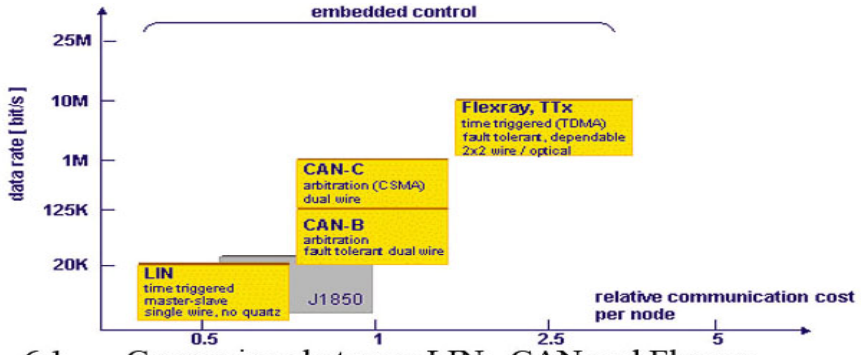
Μορφοποιήθηκε: Ελληνικά

Στην κατάσταση 1 που περισσότερα buffes μπορούν να χρησιμοποιηθούν δυναμικά και η κατάσταση 2 που στα buffer του χρησιμοποιεί τον FIFO αλγόριθμο .

Το τελικό σχέδιο είναι η χρησιμοποίηση διαφορετικού πρωτοκόλλου αντί του CAN . Νέα πρωτόκολλα όπως το Flexray και το LIN έχουν ήδη εισχωρήσει στην αγορά , τα οποία εξαρτώνται κάθε φορά από την εφαρμογή μπορεί να έχουν καλύτερα αποτελέσματα όσο αφορά την απόδοση ή μπορεί να είναι οικονομικότερες λύσεις . Μια μικρή αναφορά καθώς και σύγκριση μεταξύ αυτών τον πρωτοκόλλων θα γίνει αμέσως παρακάτω .

Το LIN (Local Interconnect Network) χρησιμοποιεί την αρχιτεκτονική master-slave και ένα απλό καλώδιο ως δίαυλο . Ο ρυθμός μετάδοσης δεδομένων είναι εως 20Kbit/sec , που είναι ένα σχετικά χαμηλό εύρος μετάδοσης δεδομένων . Θα μπορούσε να είναι υποκατάστατο του CAN κάτω από τις ακόλουθες περιστάσεις . Πρώτα απ΄όλα όταν το κόστος υλοποίησης έχει σημασία (το CAN έχει διπλάσιο κόστος αναλογικά με το LIN) και όταν το εύρος ζώνης και η μεταβλητότητα των CAN δεν χρειάζεται (δίκτυο μηχανισμού κίνησης και αισθητήρων) . [13]

Το Flexray είναι ένα πρωτόκολλο επόμενης γενιάς για κατανεμημένα δίκτυα το διακρίνει η multimaster αρχιτεκτονική του (όπως τα CAN) και τα οφέλη του είναι ότι υποστηρίζει ταχύτητες μετάδοσης εως και τα 20Mbit/sec . Καθώς ο δίαυλος χρησιμοποιεί μια διπλή γραμμή καλωδίου ή για καλύτερα αποτελέσματα οπτική ίνα . Γενικά προσφέρει όλα τα πλεονεκτήματα του CAN αλλά επιπροσθέτως προσφέρει μεγαλύτερες ταχύτητες μετάδοσης με διπλάσιο φυσικά κόστος



6.1 Comparison between LIN , CAN and Flexray

Η ιδανική αρχιτεκτονική είναι μια μίξη αυτών των τριών αρχιτεκτονικών LIN ,CAN και Flexray για τον λόγο ότι μπορούμε να προσαρμόσουμε την υλοποίηση και την αρχιτεκτονική στο ακριβές κόστος καθώς και στο συγκεκριμένο εύρος μετάδοσης .

Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά

6ii Συμπεράσματα

Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια
Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια, Ελληνικά
Μορφοποιήθηκε: Γραμματοσειρά: Έντονα, Πλάγια
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά

Το θέμα αυτής της πτυχιακής , το CAN πρωτόκολλο , ήταν πραγματικά ένα αρκετά ενδιαφέρον θέμα καθώς προσπαθεί να ενώσει δύο διαφορετικές έννοιες όπως τους μικροελεγκτές και τον χώρο των κατανεμημένων δικτύων . Οι βασικές αρχές του πρωτοκόλλου , όπως επεκτασιμότητα και αξιοπιστία , επιβεβαιώθηκαν και στην περίπτωση που το λογισμικό δεν δέχτηκε πολλές μετατροπές . Επιπλέον το κόστος ολόκληρης της υλοποίησης ήταν αρκετά φτηνό . Τα μοναδικά προβλήματα που ουσιαστικά αντιμετώπισα ήταν κάποια προβλήματα που αφορούσαν τον assembly κώδικα για την SPI επικοινωνία και οι θερμοκρασίες τις οποίες λάμβανα οι οποίες δεν ήταν οι αναμενόμενες .

Μορφοποιήθηκε: Ελληνικά
Μορφοποιήθηκε: Ελληνικά

Οι δυσκολίες που αντιμετώπισα όσο αφορά το μέρος του hardware ήταν η ακριβής εύρεση τον συγκεκριμένων αντικειμένων (αισθητήρες , δέκτες , καλώδια) τα οποία και έπρεπε να παρουσιάσουν απολύτως αξιόπιστα αποτελέσματα , σε συνδυασμό με το χαμηλό κόστος .

Μορφοποιήθηκε: Ελληνικά

6iii Λίστα αναφορών

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Βιβλία

[1] Sape Mullender, *Distributed Systems*, Addison-Wesley Publishing Company, New York 1989

[9]Olaf Pfeiffer,Andrew Ayre and Christian Keydel , Embedded Networking with CAN an CANopen , USA November 2003

Papers

[2] Feng-LiLian, John K.Yook, Dawn M.Tilbury, and James Moyne ,IEEE Transactions on industrial informatics, vol.2, NO.1, February 2006

[3] Robert I. Davis and Alan Burns, Reinder J. Bril and Johan J. Lukkien *Controller Area Network (CAN) Schedulability Analysis: Refuted, Revisited and Revised,* 08 May 2007

[4]Siemens Microelectronics, CANPRES Version 2.0, October 1998

[6]Sreeram Krishnamoorthy, Design of an ASIC chip for CAN protocol controller, thesis in Electrical Engineering, August 2006

[7] Daniel Mannisto and Mark Dawson, *An Overview of Controller Area Network (CAN) Technology*, November 12, 2003

[13] XILINK ,LIN Bus ,A Cost Effective Alternative to CAN , 2004, UK

Datasheets

[12] a, PIC18F2585/2680/4585/4680Data Sheet, www.microchip.com 2006 Microchip Technology Inc.

b. Comparing CAN and ECAN Modules , www.microchip.com 2006 Microchip Technology Inc.

c. Design consideration when adding CAN bus to your System www.microchip.com 2006 Microchip Technology Inc., 19 November 2003

d. TC72, Digital Temperature Sensor with SPI Interface, 2002

e. MCP2551, High-Speed CAN Transceiver, 2007

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Websites

[5] http://www.can-cia.org/applications/canopen ,
http://www.can-cia.org/can/protocol/
2004, website of organisation CAN in automation (CiA).
[8] http://www.kvaser.com/index.htm , website of Kvaser advances CAN solutions
[9] http://gramlich.net/projects/rb2/develop.txt , website of Wayne C. Gramlich 1993-2000
[10] a. http://en.wikipedia.org/wiki/Controller_Area_Network , free-content encyclopedia on the Internet
b. http://en.wikipedia.org/wiki/PID_controller , free-content encyclopedia on the Internet
[11] http://www.interfacebus.com/Can_Bus_Connector_Pinout.html#c
Leroy Davis , 1998 – 2007
[14] http://tech.digitimes.com.tw/ , Digitimes Inc.

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Μορφοποιήθηκε: Αγγλικά (Η.Β.)

Μορφοποιήθηκε: Αγγλικά (Η.Π.Α.)

Μορφοποιήθηκε: Στηλοθέτες: 206,25 στ., Αριστερά

APPENDIX A

Κόστος του CAN συστήματος

Όλα τα υλικά που χρησιμοποιήθηκαν σε αυτή την πτυχιακή αγοράστηκαν από την εταιρεία microchip , η οποία και είναι ηγετική εταιρεία στο χώρο των μικροελεγκτών . Πιο συγκεκριμένα χρησιμοποιήθηκαν για το CAN σύστημα 3 μικροελεγκτές , 3 πομποδέκτες και 2 αισθητήρες θερμοκρασίας . Τέλος το πρόγραμμα που χρησιμοποιήθηκε για τη φόρτωση του κώδικα είναι ο ICD2 προγραμματιστής .
More specifically the CAN system contained by 3 microcontrollers , 3 transceivers

Κόστος : 3 x PIC18f4585 = 3 x 4,30 £ = 12.9 £
3 x MCP2551 = 3 x 0,48 £ = 1,44 £
2 x TC72 = 2 x 0,35 £ = 0,70 £
+ 15.04 £

Το κόστος του ICD δεν συμπεριλαμβάνεται για το λόγο ότι δεν είναι μεμονωμένα . Επιπλέον 2 από τους 3 μικροελεγκτές προγραμματίστηκαν μέσω της σειριακής θύρας χρησιμοποιώντας ένα bootloader και ένα κλασικό RS-232 καλώδιο .

Συμπερασματικά το κόστος είναι αρκετά χαμηλό .

APPENDIX B

C and Assembly CODES

C transmit code for internal node

```
#include "ECAN.h"
#include "p18f4585.h"
#include "p18cxxx.h"
#include "adc.h"
#include <timers.h>
#include <delays.h>

unsigned long ID;
unsigned long IDcnt;
BYTE Ttemphigh[8];
BYTE TtemphighLen;
BYTE Ttemplow[8];
BYTE TtemplowLen;
ECAN_RX_MSG_FLAGS MsgFlags;
char rxstr[2];
unsigned long rid;
BYTE rdata[8];
BYTE rdataLen;
ECAN_RX_MSG_FLAGS rMsgFlags;
unsigned char CANTXsqCounter;
unsigned char SendFlag;
unsigned int  Result;

void init(void);
void OpenTimer0(unsigned char config);
void SetTMR0(void);
void CANTIMER_MSG (void);
```

```
void SendCanFrame(void);
void genID(void);
void OutputCanData(void);
void checkTemp(void);
void Rx(void);
void AD(void);
extern void read_the_temp (void);

/*#define Tx4D0  0
#define Tx308  1
#define Tx418  2*/
#define onSIG  0x01
#define offSIG 0x00
#define THRESH 20
#define UMAX 1000
unsigned int asm_temp;

//ISR
#pragma code isrHighVector = 0x08
void atisrHighVector (void)
{
_asm GOTO CANTIMER_MSG _endasm
}

#pragma code
#pragma  interrupt CANTIMER_MSG
void CANTIMER_MSG (void)
{
 if(INTCONbits.TMR0IF == 1)
  {
        T0CONbits.TMR0ON  = 0;
        INTCONbits.TMR0IF = 0;
        SendFlag = 1;
        SetTMR0();
```

```
_____
      }
}

void main ()
{
 init();
 while(1)
 {
    checkTemp();
      }//end wile 1
}

void init()
{
 int i;
 TRISAbits.TRISA0 = 1;              //RA1 as input
 TRISBbits.TRISB2 = 0;              // Set Tx Pin As OUTPUT CAN
 TRISBbits.TRISB3 = 1;              // Set RX pin as Input CAN
 TRISDbits.TRISD0 = 0;              //For LED TX
 TRISDbits.TRISD1 = 0;              //For Led Tx
 OpenTimer0(TIMER_INT_ON & T0_16BIT & T0_SOURCE_INT &
       T0_PS_1_128);  // kathe 128 clocks ayksanei mia timh sto 16 bit pedio
 ADCON0bits.ADON   = 1;
 RCONbits.IPEN     = 1;
 INTCONbits.GIE    = 1;
 INTCONbits.PEIE   = 0;
 INTCONbits.TMR0IE = 1;
 LATDbits.LATD1    = 0;
 CANTXsqCounter    = 0;
 IDcnt             = 0; // init the ID counter to 0
 asm_temp = 0;

for(i=0; i<8; i++){
```

```
        Ttemphigh[i] =0x00;
        Ttemplow[i] =0x00;
}
 ECANInitialize();
 SetTMR0();
}
//Initialize Can Frames
void SendCanFrame(void)
{
 checkTemp();  // Check Temperature & set the necessary bit
 if(asm_temp < THRESH){
  // set signal to turn heating on
  genID();      // generate a new ID
  // now send the first temp byte frame
  Ttemphigh[0] = asm_temp;
  while(!ECANSendMessage(ID,&Ttemphigh[0],8,ECAN_TX_STD_FRAME));
 }else{
  // do nothing
 }
}
// generate a new id
// check limitations with what ECANSendMessage wants as ID !!!
void genID(void)
{
 if(IDcnt < UMAX){
  IDcnt++;
 } else {
  IDcnt=0;
 }
 ID = IDcnt;
}
// Checks the temperature
void checkTemp(void)
{
```

```
  read_the_temp();
}

void Rx()
{
          Delay10KTCYx(100);
          LATDbits.LATD1 ^= 1;
     //LATDbits.LATD0 ^= 1;
}
//set Timer interupt for next 10msec
void SetTMR0(void)
{
 TMR0H = 0xB1;
 TMR0L = 0xDF;
 T0CONbits.TMR0ON = 1;

}
// Send Can frames on to bus

void OutputCanData(void)
{
  if (SendFlag == 1){ // its time to send a frame
        SendCanFrame();    // so send it
        SendFlag = 0x00;   // reset the flag to 0
//LATDbits.LATD0 ^= 1; // useful to close the transmitting pin
  }
}
```

C transmit code for external node

```
#include "ECAN.h"
```

```
#include "p18f4585.h"
#include "p18cxxx.h"
#include "adc.h"
#include <timers.h>
#include <delays.h>

#define onSIG  0x01
#define offSIG 0x00
#define THRESH 10 // temperature threshold
#define ARRAYSIZE 20
#define UMAX 1000

unsigned long ID;
unsigned long IDcnt;
BYTE Ttemphigh[8];
BYTE TtemphighLen;
BYTE Ttemplow[8];
BYTE TtemplowLen;
ECAN_RX_MSG_FLAGS MsgFlags;
char rxstr[2];
unsigned long rid;
BYTE rdata[8];
BYTE rdataLen;
ECAN_RX_MSG_FLAGS rMsgFlags;
unsigned char CANTXsqCounter;
unsigned char SendFlag;
unsigned int  Result;
BYTE tempArray[ARRAYSIZE];
int arraycnt;

void init(void);
void OpenTimer0(unsigned char config);
void SetTMR0(void);
void CANTIMER_MSG (void);
```

```
void SendCanFrame(void);
void genID(void);
void OutputCanData(void);
void checkTemp(void);
void Rx(void);
void AD(void);
extern void read_the_temp (void);

unsigned int asm_temp;
unsigned int cnt;
unsigned int skata;
extern unsigned int asm_variable;

//ISR
#pragma code isrHighVector = 0x08
void atisrHighVector (void)
{
_asm GOTO CANTIMER_MSG _endasm
}

#pragma code
#pragma  interrupt CANTIMER_MSG
void CANTIMER_MSG (void)
{
 if(INTCONbits.TMR0IF == 1)
   {
          T0CONbits.TMR0ON  = 0;
          INTCONbits.TMR0IF = 0;
          SendFlag = 1;
          SetTMR0();

       }
}
void main ()
```

```
{
  init();
  while(1)
  {
     checkTemp();
          }//end wile 1
}
void init()
{
  int i;

  TRISAbits.TRISA0 = 1;                //RA1 as input
  TRISBbits.TRISB2 = 0;               // Set Tx Pin As OUTPUT CAN
  TRISBbits.TRISB3 = 1;               // Set RX pin as Input CAN
  TRISDbits.TRISD0 = 0;                //For LED TX
  TRISDbits.TRISD1 = 0;                //For Led Tx
  OpenTimer0(TIMER_INT_ON & T0_16BIT & T0_SOURCE_INT &
           T0_PS_1_128);  // kathe 128 clocks ayksanei mia timh sto 16 bit pedio
  ADCON0bits.ADON   = 1;
  RCONbits.IPEN     = 1;
  INTCONbits.GIE    = 1;
  INTCONbits.PEIE   = 0;
  INTCONbits.TMR0IE = 1;
  LATDbits.LATD1    = 0;
  CANTXsqCounter    = 0;
  IDcnt             = 0; // init the ID counter to 0
  asm_temp = 0;
  arraycnt = 0;

/* Initialize the array with rather
   high values as we do not want erroneous
   signals sent from the microcontroller until
   the array is filled with real data
*/
```

```
for(i=0; i<ARRAYSIZE; i++){

  // Also initialize our messages
  if (i<8){
        Ttemphigh[i] =0x00;
        Ttemplow[i] =0x00;
  }
  tempArray[i] = 15;
}
 // Initialize CAN module
 ECANInitialize();
 SetTMR0();
}
//Initialize Can Frames
void SendCanFrame(void)
{
 int meanTemp = 0;
 int i;
 for(i=0; i<ARRAYSIZE; i++){
    meanTemp += tempArray[i];
 }
 // normalize
 meanTemp = meanTemp/ARRAYSIZE;

 // now check threshold
 if(meanTemp <= THRESH){
  // set signal to turn heating on
  genID();      // generate a new ID
  // now send the first temp byte frame
  Ttemphigh[0] = meanTemp;
  while(!ECANSendMessage(ID,&Ttemphigh[0],8,ECAN_TX_STD_FRAME));
  }else{
  // do nothing
 }}
```

```
// generate a new id
// check limitations with what ECANSendMessage wants as ID !!!
void genID(void)
{
  if(IDcnt < UMAX){
    IDcnt++;
  } else {
    IDcnt=0;
  }
  ID = IDcnt;
}

// Checks the temperature
void checkTemp(void)
{
  // check if end of array
  if(arraycnt >=ARRAYSIZE){
    // if so reset the counter
    arraycnt = 0;
  }
  // read the temperature values from assembly
  read_the_temp();
  // store it in the next position in the array
  tempArray[arraycnt] = asm_temp;
  // increment the counter
  arraycnt++;
}

void Rx()
{
          Delay10KTCYx(100);
          LATDbits.LATD1 ^= 1;
     //LATDbits.LATD0 ^= 1;
}
```

```
//set Timer interupt for next 10msec

void SetTMR0(void)
{
  TMR0H = 0xB1;
  TMR0L = 0xDF;
  T0CONbits.TMR0ON = 1;

}
// Send Can frames on to bus
void OutputCanData(void)
{
   if (SendFlag == 1){ // its time to send a frame
         SendCanFrame();    // so send it
         SendFlag = 0x00;   // reset the flag to 0
//LATDbits.LATD0 ^= 1; // useful to close the transmitting pin
   }
}
```

SPI code for temperature transfer

```
LIST P=18F4585                ;directive to define processor
#include <P18F4585.INC>     ;processor specific variable definitions

;   Oscillator Selection:
    CONFIG     OSC = HS

PORTA     equ  h'F80'
PORTC     equ  h'F82'
TRISC     equ  h'F94'
SSPSTAT   equ  h'FC7'
SSPCON1   equ  h'FC6'
SSPBUF    equ  h'FC9'
```

```
SSPSTAT   equ  h'FC7'
Ctr0      equ  0x20
BSR       equ h'FE0'
Dly0    equ 0x21          ; Delay Variable (low byte)
Dly1    equ 0x22          ; Delay Variable (high byte)
dummy_1       equ      0x30

 RA0 equ d'0'
;------------BIT EQUATES-----------------------
bf          equ       0
          org 0xc0
          org 0xC0



EXTERN asm_temp   ; defined in C module
asm_variable RES   2  ; 2 byte variable

read_the_temp

 clrf  PORTC           ;Initialize PORTC by clearing output data latches

;Set up the SPI Port
  BANKSEL TRISA      ; ANO is output
  movlw b'00000000'
  movwf  TRISA

BANKSEL TRISC      ; BANK 1   4input all other outputs
  movlw  b'00010000'
  movwf  TRISC

;SPI configuration
  movlw  b'10000000'   ;// STATUS register
  movwf  SSPSTAT
```

```
BANKSEL SSPCON1
  movlw  b'00100001' ;
// CONTROL register , ( MODE 0,1 ) CKE=CKP=0  , 1Mbit
  movwf  SSPCON1

 movlw  b'00000001'     ; ANO sends 1 , a7=0 read
  movwf  PORTA
goto    Send_DT
Send_DT
 bcf     PORTA, RA0          ; Enable Chip Select Output (low)
                             ; Start Transfer
 movf    Ctr0,W              ; Get Ctr (Counter Value) in W
 movf Ctr0, dummy_1
 movwf   SSPBUF              ; put in SSPBUF
                             ; Send SPI Data to Buffer
 BANKSEL SSPSTAT         ; BANK 1

Char1   btfss   SSPSTAT,BF  ;  Data transfer complete? ??????????
                             ;  (Buffer Full?)Check Flag

 goto    Char1               ;  if not, check again
                             ; Get Received Data
 BANKSEL SSPBUF          ; BANK0
 movf    SSPBUF,W            ; Read SSPBUF register
                             ; data is not used

  ;MOVLW SSPBUF
 ;movlw 0xC3
  ;CLRF asm_temp
    movwf asm_variable
  MOVWF asm_temp
   ; put the contents of SSPBUF in the C declared variable
   ;MOVWF asm_temp
```

```
;MOVWF asm_temp+1

 bsf    PORTA, RA0        ; Disable Slave Select Output
                ; (high)

SAVE_BSR MACRO WHERE     ; Save to memory bank 1
           movwf        TEMP_WREG
           movfp        BSR, WREG
           movlb        0                ; must be in page 0 to store to WHERE
           movwf        WHERE
           movfp        TEMP_WREG, WREG
         ENDM

RESTORE_BSR        MACRO WHERE
           movlb        0
           movfp        WHERE, BSR
         ENDM              ; Update_Test_Counter
   incf    Ctr0,F            ; Increment counter variable

 Delay
       movlw   h'F0'        ; Simple Delay loop
       movwf   Dly1         ;       |
       movlw   h'0f'        ;       |
       movwf   Dly0         ;       |

                           ;Delay Loop
  DlyLoop
       decfsz  Dly0,F       ;       |
       goto    DlyLoop      ;       |
                            ;     Makes Increment Rate Visible
       decfsz  Dly1,F       ;       |
       goto    DlyLoop      ;       |
        ; Done Delay        ;       \_/
```

```
                    ;Go To Transfer More SPI Data
                    ;DoAgain
                    ;  goto    Send_DT
                    ; Done, Send Next Byte.
   RETURN
   GLOBAL read_the_temp;
   GLOBAL asm_variable;
                END
```

Σελίδα 48: [1] Μορφοποιήθηκε **Fontas** **18/4/2008 5:30:00 μμ**

Κουκκίδα + Επίπεδο: 1 + Στοίχιση: 54 στ. + Στηλοθέτης μετά: 72 στ. + Εσοχή: 72 στ.

Σελίδα 48: [2] Μορφοποιήθηκε **Fontas** **18/4/2008 5:44:00 μμ**

Εσοχή: Πρώτη γραμμή: 18 στ.

Σελίδα 48: [3] Μορφοποιήθηκε **Fontas** **18/4/2008 5:43:00 μμ**

Ελληνικά

Σελίδα 48: [4] Μορφοποιήθηκε **Fontas** **18/4/2008 5:45:00 μμ**

Ελληνικά

Σελίδα 48: [5] Μορφοποιήθηκε **Fontas** **18/4/2008 5:46:00 μμ**

Ελληνικά

Σελίδα 48: [6] Μορφοποιήθηκε **f** **23/5/2008 7:47:00 πμ**

Ελληνικά

Σελίδα 48: [7] Μορφοποιήθηκε **Fontas** **19/4/2008 4:01:00 πμ**

Ελληνικά

Σελίδα 48: [8] Μορφοποιήθηκε **Fontas** **19/4/2008 4:01:00 πμ**

Ελληνικά

Σελίδα 48: [9] Μορφοποιήθηκε **Fontas** **19/4/2008 4:01:00 πμ**

Ελληνικά

Σελίδα 48: [10] Μορφοποιήθηκε **Fontas** **19/4/2008 4:02:00 πμ**

Ελληνικά

Σελίδα 48: [11] Μορφοποιήθηκε **Fontas** **19/4/2008 4:04:00 πμ**

Ελληνικά

Σελίδα 48: [12] Μορφοποιήθηκε **Fontas** **19/4/2008 4:04:00 πμ**

Ελληνικά

Σελίδα 48: [13] Μορφοποιήθηκε **Fontas** **19/4/2008 4:05:00 πμ**

Ελληνικά

Σελίδα 48: [14] Μορφοποιήθηκε **Fontas** **19/4/2008 4:05:00 πμ**

Ελληνικά

Σελίδα 48: [15] Μορφοποιήθηκε **Fontas** **19/4/2008 4:06:00 πμ**

Ελληνικά

Σελίδα 48: [16] Μορφοποιήθηκε **Fontas** **19/4/2008 4:11:00 πμ**

Ελληνικά

Σελίδα 48: [17] Μορφοποιήθηκε **Fontas** **19/4/2008 4:12:00 πμ**

Ελληνικά

Σελίδα 48: [18] Μορφοποιήθηκε **Fontas** **19/4/2008 4:12:00 πμ**

Ελληνικά

Σελίδα 48: [19] Μορφοποιήθηκε **Fontas** **19/4/2008 4:12:00 πμ**

Ελληνικά

Σελίδα 48: [20] Μορφοποιήθηκε **f** **23/5/2008 7:47:00 πμ**

Ελληνικά

Σελίδα 48: [21] Μορφοποιήθηκε **Fontas** **19/4/2008 4:14:00 πμ**

Ελληνικά

Σελίδα 48: [22] Μορφοποιήθηκε **Fontas** **19/4/2008 4:14:00 πμ**

Ελληνικά

Σελίδα 48: [23] Μορφοποιήθηκε **f** **22/5/2008 11:37:00 μμ**

Ελληνικά

Σελίδα 48: [24] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:17:00 μμ**

Ελληνικά

Σελίδα 48: [25] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:25:00 μμ**

Ελληνικά

Σελίδα 48: [26] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:26:00 μμ**

Ελληνικά

Σελίδα 48: [27] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:26:00 μμ**

Ελληνικά

Σελίδα 48: [28] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:26:00 μμ**

Ελληνικά

Σελίδα 48: [29] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:17:00 μμ**

Ελληνικά

Σελίδα 48: [30] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:18:00 μμ**

Ελληνικά

Σελίδα 48: [31] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:18:00 μμ**

Ελληνικά

Σελίδα 48: [32] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:21:00 μμ**

Ελληνικά

Σελίδα 48: [33] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:27:00 μμ**

Ελληνικά

Σελίδα 48: [34] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:27:00 μμ**

Ελληνικά

Σελίδα 48: [35] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:27:00 μμ**

Ελληνικά

Σελίδα 48: [36] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:27:00 μμ**

Ελληνικά

Σελίδα 48: [37] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:27:00 μμ**

Ελληνικά

Σελίδα 48: [38] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:28:00 μμ**

Ελληνικά

Σελίδα 48: [39] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:29:00 μμ**

Ελληνικά

Σελίδα 48: [40] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:29:00 μμ**

Ελληνικά

Σελίδα 48: [41] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:30:00 μμ**

Ελληνικά

Σελίδα 48: [42] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:31:00 μμ**

Ελληνικά

Σελίδα 48: [43] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:31:00 μμ**

Ελληνικά

Σελίδα 48: [44] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:33:00 μμ**

Ελληνικά

Σελίδα 48: [45] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:33:00 μμ**

Ελληνικά

Σελίδα 48: [46] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:34:00 μμ**

Ελληνικά

Σελίδα 48: [47] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:34:00 μμ**

Ελληνικά

Σελίδα 48: [48] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:36:00 μμ**

Ελληνικά

Σελίδα 48: [49] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:37:00 μμ**

Ελληνικά

Σελίδα 48: [50] Μορφοποιήθηκε	**Fontas**	**19/4/2008 4:37:00 μμ**

Ελληνικά