ΑΝΩΤΑΤΟ ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΧΑΝΙΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΗΣ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

"DATA LOGGING SYSTEMS USING *1 – WIRE*™ PROTOCOL"

"ΣΥΣΤΗΜΑΤΑ ΣΥΛΛΟΓΗΣ ΔΕΔΟΜΕΝΩΝ ΜΕ ΧΡΗΣΗ ΤΟΥ ΠΡΩΤΟΚΟΛΛΟΥ *1 – WIRE*™ "

ΕΥΑΓΓΕΛΟΣ Χ. ΓΚΑΡΤΖΟΝΙΚΑΣ

ΜΙΧΑΗΛ Ν. ΚΑΤΣΑΡΑΣ

ΕΙΣΗΓΗΤΗΣ: ΕΜΜΑΝΟΥΗΛ ΑΝΤΩΝΙΔΑΚΗΣ

ΜΑΡΤΙΟΣ 2006

INTRODUCTION

The **Dallas Semiconductor** products are a family of devices that all communicate over a single wire following a specific command sequence referred to as the **1–Wire™** Protocol. A key feature of each device is a unique 8–byte ROM code written into each part at the time of manufacture.

The least significant byte contains a family code that identifies the type of the product. Since multiple devices of the same or different family types can reside on the same 1–Wire bus simultaneously, it is important for the host to be able to determine how to properly access each of the devices that it locates on the 1–Wire bus. The family code provides this information.

The next six bytes contain a unique serial number that allows multiple devices within the same family code to be distinguished from each other. This unique serial number can be thought of as an “address” for each device on the 1–Wire bus.

The entire collection of devices plus the host form a type of miniature local area network, or Micro-LAN; they all communicate over the single common wire. The most significant byte in the ROM code of each device contains a Cyclic Redundancy Check (CRC) value based on the previous seven bytes of data for that part.

ΕΥΑΓΓΕΛΟΣ Χ. ΓΚΑΡΤΖΟΝΙΚΑΣ

ΜΙΧΑΗΛ Ν. ΚΑΤΣΑΡΑΣ

ΕΙΣΗΓΗΤΗΣ : ΕΜΜΑΝΟΥΗΛ ΑΝΤΩΝΙΔΑΚΗΣ

ΠΕΡΙΕΧΟΜΕΝΑ

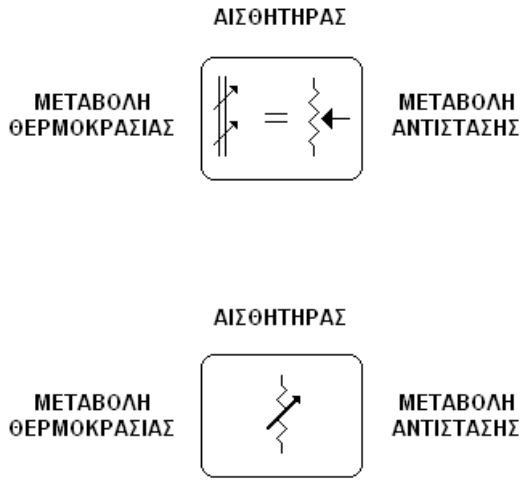
ΚΕΦΑΛΑΙΟ 1

DATA LOGGING SYSTEMS

ΑΙΣΘΗΤΗΡΙΑ-ΜΕΤΑΤΡΟΠΕΙΣ

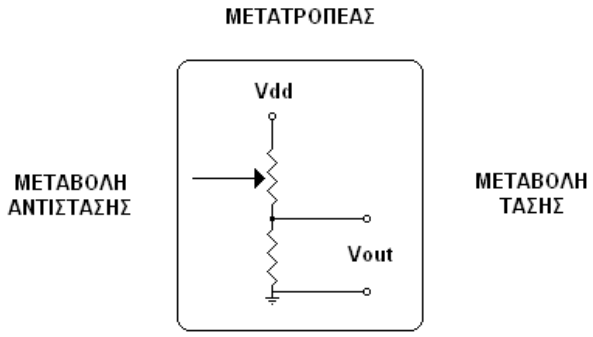
Αισθητήρα καλούμε το φυσικό υλικό, στο οποίο η μεταβολή του προς μέτρηση φυσικού μεγέθους, προκαλεί μεταβολή σε κάποιο ή κάποια από τα φυσικά χαρακτηριστικά του υλικού.

Για παράδειγμα για να μετρήσουμε την θερμότητα και να την εκφράσουμε σε θερμοκρασία, χρησιμοποιούμε ένα αισθητήρα όπου τα χαρακτηριστικά του μεταβάλλονται ανάλογα με την μεταβολή της θερμότητας. Τα χαρακτηριστικά αυτά μπορεί να είναι τάση, αντίσταση, χωρητικότητα, επαγωγή ή ακόμα και μηχανικές μεταβολές. Ένα παράδειγμα αισθητήρα φαίνεται στο **σχήμα 1.1** όπου το μετρούμενο μέγεθος είναι η θερμοκρασία και η χαρακτηριστική ιδιότητα που μεταβάλλεται είναι η αντίσταση.



Σχήμα 1.1: αισθητήρας

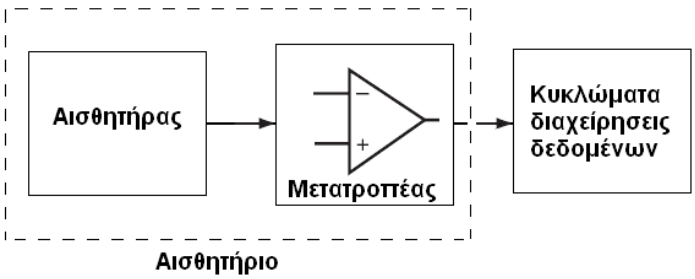
Για να μπορέσουμε να διαχειριστούμε τις μεταβολές αυτών των χαρακτηριστικών χρειαζόμαστε μία κυκλωματική διάταξη, η οποία θα μετατρέψει το χαρακτηριστικό αυτό σε τάση, η κυκλωματική διάταξη καλείται μετατροπέας. Γενικά οι μετατροπείς μετατρέπουν ένα φυσικό μέγεθος σε ένα άλλο όπως φαίνεται στο **σχήμα 1.2**.



σχήμα 1.2: μετατροπείς

Μετά τον Αισθητήρα ακολουθεί ο μετατροπέας, γενικότερα οι μετατροπείς μπορεί να είναι διατάξεις όπως, γέφυρες, τελεστικοί ενισχυτές ή **A/D Converters,** οι οποίοι μεταβάλουν το μετρούμενο μέγεθος σε επιθυμητές στάθμες και τιμές. Ένας μετατροπέας θα αναλάβει να ενισχύσει για παράδειγμα την τάση εάν είναι πολύ μικρή. Ακόμη μπορεί να την ορίσει σε στάθμες για να την μετατρέψουμε σε ψηφιακά δεδομένα, όπως επίσης να μετατρέψει το μετρούμενο μέγεθος, σε κάποιο άλλο φυσικό μέγεθος.

Με τον όρο αισθητήριο εννοούμε την διάταξη η οποία περιλαμβάνει τον φυσικό αισθητήρα και τον μετατροπέα όπως φαίνεται στο **σχήμα 1.3**



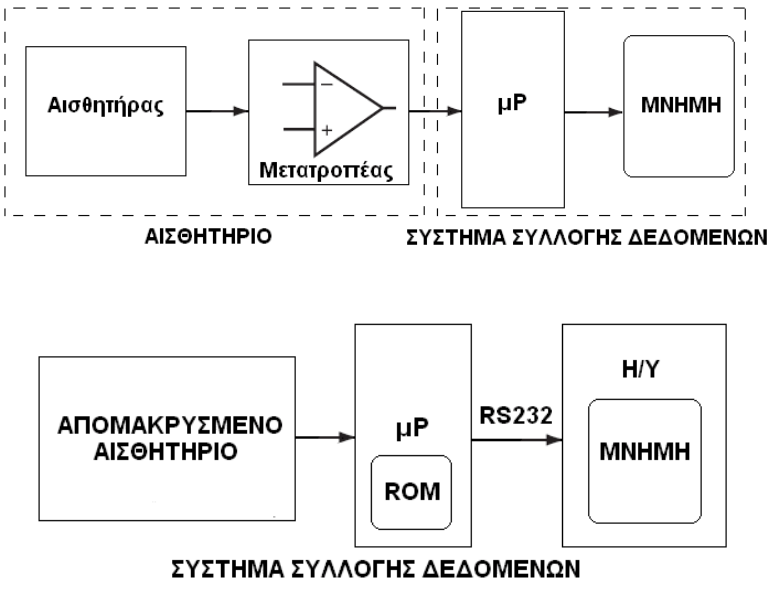
σχήμα 1.3: Αισθητήριο

ΣΥΣΤΗΜΑΤΑ ΣΥΛΛΟΓΗΣ ΔΕΔΟΜΕΝΩΝ

Παρατηρείται μία ραγδαία εξέλιξη στον τομέα των αισθητηρίων, εξέλιξη η οποία πηγάζει από την συνεχώς αυξανόμενη ανάγκη των συστημάτων ελέγχου και αυτοματισμού.

Τα συστήματα συλλογής δεδομένων είναι συνήθως τα κυκλώματα που συνδέονται άμεσα με τα αισθητήρια. Είναι τα συστήματα εκείνα που θα συλλέξουν τα δεδομένα, θα τα επεξεργαστούν και ανάλογα θα ακολουθήσει κάποια ενέργεια (Συστήματα αυτομάτου ελέγχου) ή απλά θα τα αποθηκεύσουν (για περαιτέρω ανάλυση οπότε μιλάμε πλέον για συστήματα συλλογής δεδομένων). Ένα σύστημα συλλογής δεδομένων μπορεί να έχει την μορφή του **σχήματος 1.4**. Οι μεταβολές του μετρούμενου μεγέθους θα προκαλέσουν μεταβολή σε ένα χαρακτηριστικό του αισθητήρα, το οποίο στην συνέχεια θα μετατραπεί σε πληροφορία για το σύστημα μας από τον μετατροπέα.

Τα δεδομένα από τον μετατροπέα συλλέγονται από το σύστημα συλλογής δεδομένων το οποίο αποτελείται από μία μονάδα επεξεργασίας και μία μνήμη. Η μονάδα επεξεργασίας μπορεί να είναι ένας Η/Υ, ή ένας μικροεπεξεργαστής ή και τα δύο.



σχήμα 1.4: Σύστημα συλλογής δεδομένων

ΚΕΦΑΛΑΙΟ 2

ΠΡΩΤΟΚΟΛΛΟ 1-WIRE

ΕΙΣΑΓΩΓΗ

Τα **1-Wire** προϊόντα της **Dallas Semiconductor** είναι μία οικογένεια συσκευών οι οποίες επικοινωνούν μέσω ενός μοναδικού καλωδίου ακολουθώντας μία συγκεκριμένη ακολουθία εντολών, του **1-Wire™** πρωτόκολλου.

Το χαρακτηριστικό κάθε συσκευής είναι ένας μοναδικός **8-Byte ROM** κωδικός, ο οποίος γράφεται στις συσκευές την στιγμή της παραγωγής. Το **LSB** περιέχει έναν **Family Code** όπου διευκρινίζει τον τύπο της συσκευής. Τα επόμενα **6-Bytes** περιέχουν ένα μοναδικό αύξων αριθμό ο οποίος δίνει την δυνατότητα σε συσκευές του ίδιου **Family Code** να ξεχωρίζουν η μία από την άλλη. Μπορούμε να παρομοιάσουμε τον αύξων αριθμό σαν μία “διεύθυνση” για κάθε συσκευή στον δίαυλο **1-Wire**.

Οι συσκευές μαζί με τον **Master** διαμορφώνουν έναν τύπο μινιατούρας τοπικού δικτύου ή **MicroLAN** όπως ονομάζεται και επικοινωνούν μέσω ενός κοινού καλωδίου. Το **8^{o} Byte** του **ROM** κωδικού κάθε συσκευής περιέχει την τιμή ενός κυκλικού ελέγχου πλεονασμού (**CRC**), η οποία απορρέει από τα προηγούμενα **7 bytes** δεδομένων του εξαρτήματος.

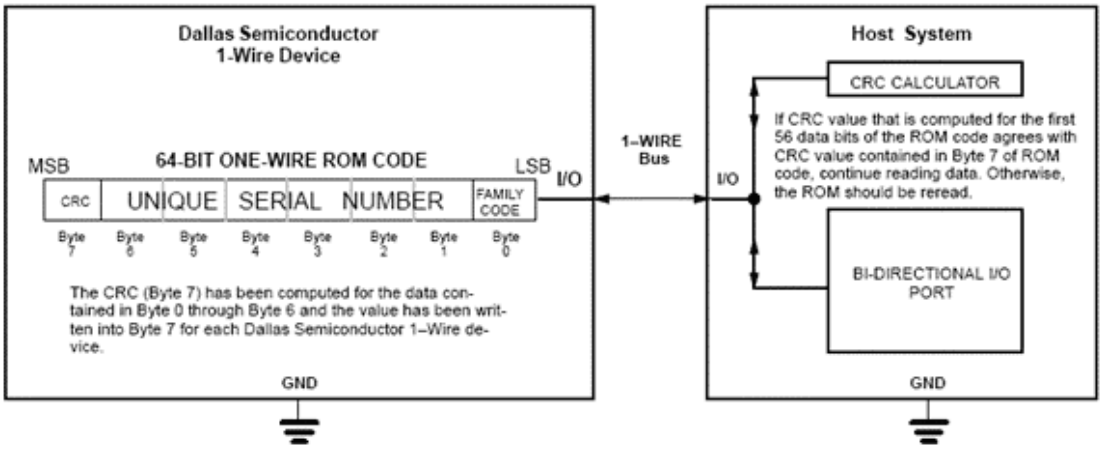
CRC (CYCLIC REDUNDANCY CHECKS)

Η κάθε συσκευή έχει αποθηκευμένη στο **MSB** της **64-bit ROM** μία **8-bit** τιμή **CRC**, την οποία μπορεί να υπολογίσει και ο **Master** από τα πρώτα **56-bits** της **ROM**. Ο **Master** επαληθεύει εάν έλαβε σωστά τα δεδομένα της **ROM** της συσκευής, συγκρίνοντας αυτές τις δύο τιμές, Η ισοδύναμη πολυωνυμική εξίσωση υπολογισμού του **CRC** είναι:

$$\mathbf{CRC = X^8 + X^5 + X^4 + 1}$$

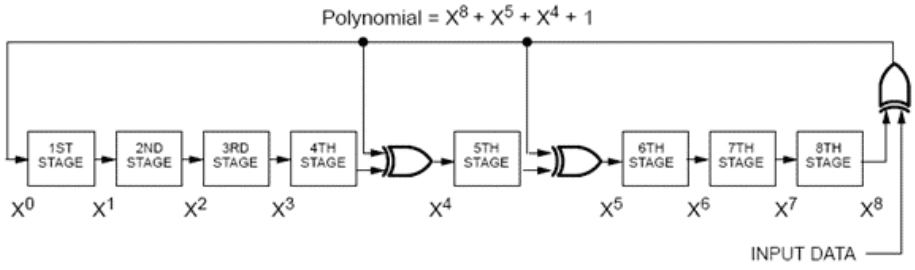
Η συσκευή παράγει επίσης έναν **8bit CRC** χρησιμοποιώντας την ίδια πολυωνυμική εξίσωση όπως φαίνεται και παραπάνω. Έπειτα θα στείλει την τιμή του **CRC** στον επεξεργαστή για να επαληθεύσουν την σωστή μεταφορά

δεδομένων. Σε κάθε περίπτωση όπου το **CRC** χρησιμοποιείται για την επιβεβαίωση της σωστής μεταφοράς δεδομένων, ο **Master** πρέπει να υπολογίζει την τιμή του **CRC** χρησιμοποιώντας την εξίσωση και να συγκρίνει την τιμή είτε με την αποθηκευμένη **CRC** της **ROM** είτε με την τιμή που υπολογίζει η ίδια η συσκευή και βρίσκεται στο **9^{o} Byte** του **Scratchpad**. Από την σύγκριση των τιμών αυτών, ο **Master** θα καθορίσει εάν θα συνεχιστεί ή θα επαναληφθεί η διαδικασία. Όπως φαίνεται και στο **σχήμα 2.1**.



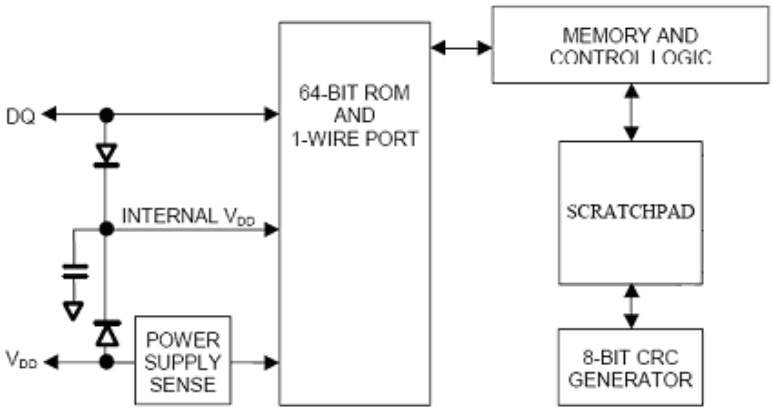
σχήμα 2.1: Cyclic Redundancy Checks

Μπορούμε να αναπαράγουμε το **1-Wire CRC**, φτιάχνοντας μία πολυωνυμική γεννήτρια αποτελούμενη από καταχωρητές μετατόπισης και πύλες **XOR**. Οι καταχωρητές αρχικοποιούνται στο ‘’**0**’’ και αρχίζοντας από το **LSb** του **Family Code** ένα **bit** την φορά μετατοπίζεται μέσα στους καταχωρητές, τα πρώτα **8** για το **Family Code** τα επόμενα **48** για τον αύξων αριθμό και μόλις συμπληρώσουμε τα **56 bit** θα έχουμε την τιμή του **CRC**. Όταν μετατοπίσουμε και τα τελευταία **8 bit** του **CRC** οι καταχωρητές πρέπει να μηδενίσουν. Στο **σχήμα 2.2** φαίνεται η γεννήτρια παραγωγής **CRC**.



σχήμα 2.2: Γεννήτρια παραγωγής CRC

Ο γενικός πυρήνας των συσκευών **1-Wire** όπως φαίνεται στο **σχήμα 2.3** αποτελείται από ένα μηχανισμό τροφοδότησης, από μία μνήμη **ROM 64 bit**, από ένα ελεγκτή ελέγχου λειτουργίας μνήμης, μία μνήμη **Scratchpad** και ένα μηχανισμό αναγέννησης **CRC**.



σχήμα 2.3: Γενικός πυρήνας των συσκευών 1-Wire

Η ισχύς για την ανάγνωση, το γράψιμο και την εκτέλεση εντολών μπορεί να προέλθει από την ίδια την γραμμή δεδομένων χωρίς την ανάγκη εξωτερικής τροφοδότησης των συσκευών.

ΠΑΡΑΣΙΤΙΚΗ ΤΡΟΦΟΔΟΣΙΑ

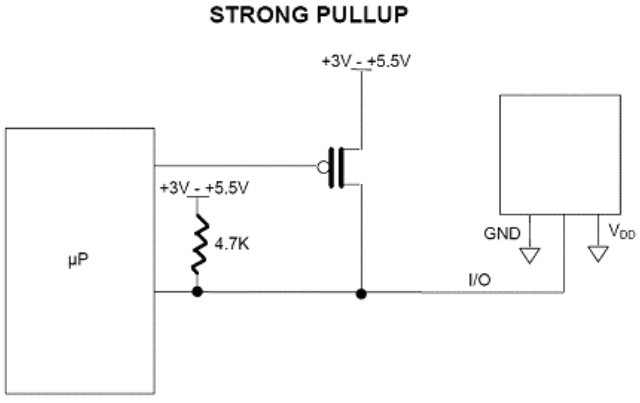
Στο **σχήμα 2.3**, έχουμε την γενική μορφή του κυκλώματος της «**παρασιτικής**» τροφοδοσίας. Κάθε φορά που τα **I/O** και **VDD** pins βρίσκονται σε υψηλό δυναμικό επίπεδο το κύκλωμα αυτό «**κλέβει**» ενέργεια. Η συσκευή θα αντλήσει επαρκή ενέργεια από την γραμμή δεδομένων, εφόσον οι προκαθορισμένες απαιτήσεις χρονισμού και τάσης ικανοποιούνται. Τα πλεονεκτήματα της παρασιτικής τροφοδότησης είναι τα εξής:

Με την χρήση της, δεν απαιτείτε καμία τοπική πηγή ενέργειας για την απομακρυσμένη συσκευή και η **ROM** μπορεί να διαβαστεί χωρίς την ύπαρξη εξωτερικής τροφοδότησης.

Όμως, για να είναι ικανές η συσκευές να εκτελούν διαδικασίες που απαιτούν υψηλές ποσότητες ενέργειας, θα πρέπει να παρέχεται ικανοποιητική ισχύ μέσο της **I/O** γραμμής δεδομένων, κατά την διάρκεια της εκτέλεσης τους. Καθώς το ρεύμα λειτουργίας των συσκευών μπορεί να φτάνει μέχρι και **1,5 mA**, η γραμμή δεδομένων δεν θα είναι ικανή να τα τροφοδοτήσει με την ενέργεια που χρειάζονται, λόγω του **Pull-Up** αντιστάτη των **5k**. Το πρόβλημα αυτό γίνεται εντονότερο, κατά την παρουσία πολλών συσκευών στην γραμμή δεδομένων, τα οποία εκτελούν ταυτόχρονες τέτοιες διαδικασίες.

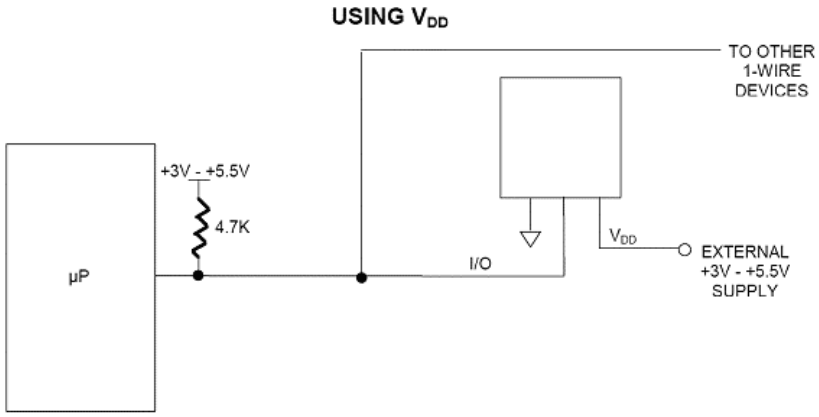
Υπάρχουν δύο τρόποι για να διασφαλίσουμε ότι οι συσκευές θα έχουν ικανοποιητικό ρεύμα τροφοδοσίας, κατά τη διάρκεια του ενεργού κύκλου αυτών των διαδικασιών. Ο πρώτος τρόπος είναι να παρέχουμε μία ισχυρή ανύψωση (**Pull-Up**) στην γραμμή δεδομένων, όποτε λαμβάνουν χώρα τέτοιες διαδικασίες.

Αυτό μπορεί να πραγματοποιηθεί με τη χρήση **Mosfet** το οποίο θα συνδέει άμεσα την γραμμή δεδομένων με την πηγή τροφοδοσίας, (όπως φαίνεται στο **σχήμα 2.4**). Η διαδικασία αυτή θα πρέπει να ολοκληρωθεί μέσα σε **10 ms** το αργότερο, μετά από την έκδοση οποιασδήποτε εντολής που αφορά μία τέτοια διαδικασία η οποία μπορεί να είναι π.χ. εγγραφή στην μνήμη **EEPROM** ή έναρξη μετατροπών θερμοκρασίας. Όταν χρησιμοποιείτε παρασιτική τροφοδότηση, η επαφή $\mathbf{V_{DD}}$* θα πρέπει να είναι συνδεδεμένη στη γη. *(Ο εναλλακτικός τρόπος τροφοδότησης με επαφή $\mathbf{V_{DD}}$ δεν υποστηρίζεται από όλες τις συσκευές.)



σχήμα 2.4: Pull-Up με τη χρήση Mosfet

Ως εναλλακτική μέθοδος τροφοδοσίας των συσκευών, είναι η χρήση εξωτερικής πηγής ενέργειας, η οποία θα συνδέεται στην επαφή $\mathbf{V_{DD}}$ (όπως φαίνεται στο **σχήμα 2.5**). Το πλεονέκτημα αυτής της μεθόδου είναι ότι δεν απαιτείται ισχυρό **Pull-Up** στην γραμμή δεδομένων, με αποτέλεσμα ο ελεγκτής της γραμμής να μην χρειάζεται να παρέμβει κατά την διάρκεια των διαδικασιών που απαιτούν υψηλά ποσά ενέργειας. Αυτή η μέθοδος επιτρέπει την κυκλοφορία δεδομένων στον δίαυλο **1-Wire** κατά την διάρκεια του χρόνου που λαμβάνουν χώρα τέτοιες διαδικασίες. Με την χρήση αυτής μεθόδου αυτής η επαφή **GND** δεν θα πρέπει να έχει διακυμάνσεις.



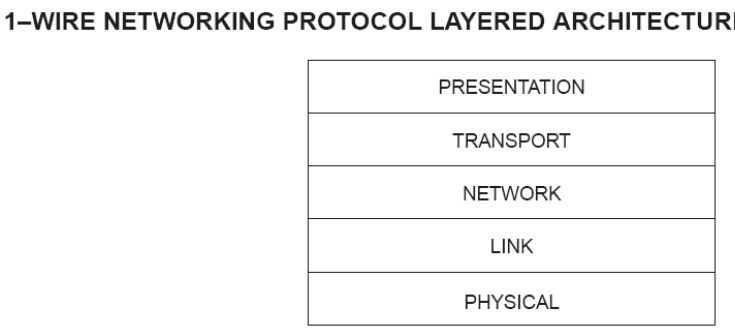
σχήμα 2.5: Εξωτερική πηγή ενέργειας

Στην περίπτωση όπου ο ελεγκτής διαύλου δεν γνωρίζει με ποιόν τρόπο τροφοδοτούνται οι συσκευές που βρίσκονται στην γραμμή, έχει προβλεφθεί σε αυτές τις συσκευές η δυνατότητα να επισημάνουν το χρησιμοποιούμενο σχέδιο παροχής ηλεκτρικού ρεύματος στον ελεγκτή διαύλου.

Ο ελεγκτής διαύλου μπορεί να προσδιορίσει εάν υπάρχουν συσκευές στην γραμμή που να χρειάζονται ισχυρό **Pull-Up** και το επιτυγχάνει με την εντολή **Read Power Supply**. Στην συνέχεια ο ελεγκτής διαύλου, τίθεται σε διαδικασία διαβάσματος περιμένοντας απάντηση από τις συσκευές. Οι συσκευές θα επιστρέψουν ‘’**0**’’ εάν τροφοδοτούνται παρασιτικά ή ‘’**1**’’ εάν τροφοδοτούνται από εξωτερική πηγή.

ΑΡΧΙΤΕΚΤΟΝΙΚΟ ΜΟΝΤΕΛΟ ΠΡΩΤΟΚΟΛΛΟΥ

Το λογισμικό ή το λογισμικό της **ROM** που διαχειρίζεται την μεταφορά δεδομένων προς και από τις συσκευές μπορεί να συσχετιστεί με το μοντέλο αναφοράς **O**pen **S**ystem **I**nterconnection (**OSI**) του **I**nternational **S**tandards **O**rganization (**ISO**), το οποίο καθορίζει ένα πρωτόκολλο **7** επιπέδων, τα οποία είναι: **Physical, Link, Network, Transport, Session, Presentation** και **Application**. Η φύση του πρωτοκόλλου **1-Wire** δεν απαιτεί και τα **7** επίπεδα του **OSI**, αλλά μόνο **5** από αυτά: **Physical, Link, Network, Transport,** και **Presentation**. Στο **σχήμα 2.6** φαίνεται η ιεραρχία αυτών των επιπέδων.



σχήμα 2.6: Η ιεραρχία των επιπέδων 1-Wire.

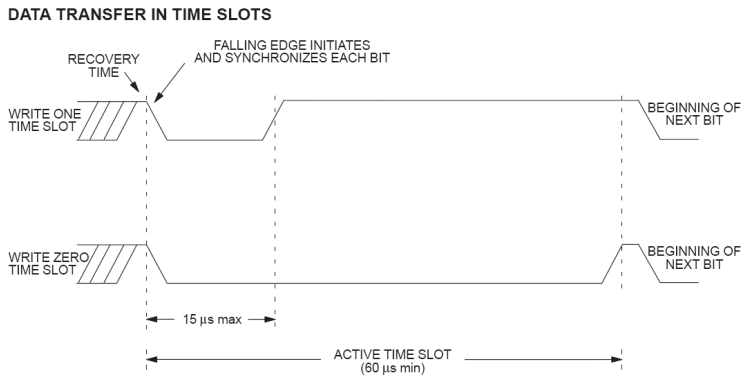
Η παρακάτω περιγραφή ακολουθεί την ιεραρχία αυτών των επιπέδων πρωτοκόλλου.

PHYSICAL LAYER

Σε αυτό το επίπεδο διευκρινίζονται τα ηλεκτρικά χαρακτηριστικά, την τάση των λογικών επιπέδων και γενικού χρονισμού επικοινωνίας.

Το **1-Wire** πρωτόκολλο χρησιμοποιείται για **Bi-Directional** επικοινωνία. Είναι ένα σειριακό **Half Duplex** πρωτόκολλο με ορισμένες διακριτές χρονοθυρίδες (**Time Slots**). Σε κάθε περίπτωση ο ελεγκτής (**Master**) αρχικοποιεί την μεταφορά στέλνοντας μία εντολή στην **1-Wire** συσκευή (**Slave**). Εντολές και δεδομένα στέλνονται ανά **bit** για να δημιουργήσουν **Bytes**, ξεκινώντας από το λιγότερο σημαντικό **bit**.

Ο συγχρονισμός του ελεγκτή (**Master**) και των συσκευών (**Slave**) βασίζεται στην βύθιση που δημιουργεί ο ελεγκτής τραβώντας τη γραμμή δεδομένων σε χαμηλό δυναμικό. Συγκεκριμένο χρόνο μετά την βύθιση, ο οποίος εξαρτάται από την κατεύθυνση των δεδομένων, ανάλογα με το αν δειγματοληπτεί ο **Master** ή ο **Slave** τη γραμμή για τον προσδιορισμό του **bit**. Αυτή η μέθοδος λειτουργίας καλείτε μεταφορά δεδομένων σε χρονοθυρίδες.
Κάθε χρονοθυρίδα είναι ανεξάρτητα χρονισμένη έτσι ώστε αν χρειαστεί να μπορούμε να κάνουμε παύσεις στην επικοινωνία ανάμεσα στα **bit**, χωρίς να δημιουργούνται λάθη. Στο **σχήμα 2.7** φαίνονται τα γενικά χαρακτηριστικά της επικοινωνίας.

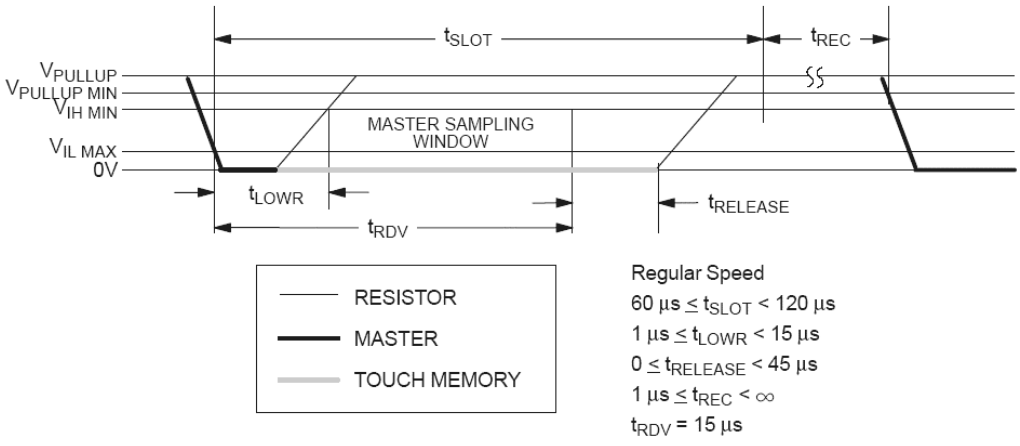


σχήμα 2.7: Γενικά χαρακτηριστικά της επικοινωνίας

Τα ηλεκτρικά χαρακτηριστικά που πρέπει να πληρεί το **1-Wire** πρωτόκολλο φαίνονται στον **πίνακα 2.1** και στο **σχήμα 2.8**.

πίνακας 2.1: Ηλεκτρικά χαρακτηριστικά 1-Wire

$V_{Pull\text{-}Up\ max}$	6 Volt
$V_{Pull\text{-}Up\ min}$	2.8 Volt
$V_{IH\ min}$	2.2 Volt
$V_{IL\ max}$	0.8 Volt
$t_{SLOT\ min}$	60 µs
$t_{SLOT\ max}$	120 µs
t_{RDV}	15 µs



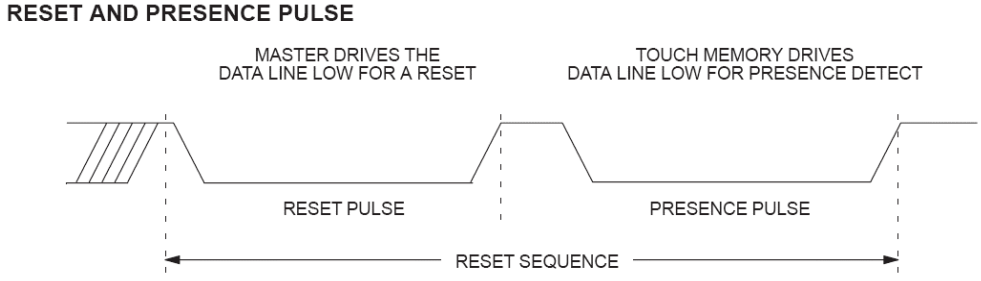
σχήμα 2.8 Ηλεκτρικά χαρακτηριστικά 1-Wire

LINK LAYER

Αυτό το επίπεδο προσδιορίζει τις βασικές λειτουργίες επικοινωνίας. Παρέχει τις βασικές λειτουργίες του **Reset**, ανίχνευσης παρουσίας και μεταφοράς **bit**. Μετά την έκδοση του παλμού παρουσίας η επικοινωνία περνάει στο **Network Layer**.

ΑΡΧΙΚΟΠΟΙΗΣΗ

Η μεταφορά δεδομένων δεν μπορεί να πραγματοποιηθεί πριν την αρχικοποίηση της γραμμής. Ο **Master** εκδίδει ένα παλμό **Reset** ζητώντας έτσι από τις συσκευές να απαντήσουν με ένα παλμό παρουσίας (**Presence**). Μόλις η συσκευή ανιχνεύσει παλμό **Reset** στην γραμμή δεδομένων δημιουργεί ένα παλμό παρουσίας. Μία ολοκληρωμένη ακολουθία παλμών **Reset/Presence** φαίνεται στο **σχήμα 2.9**.



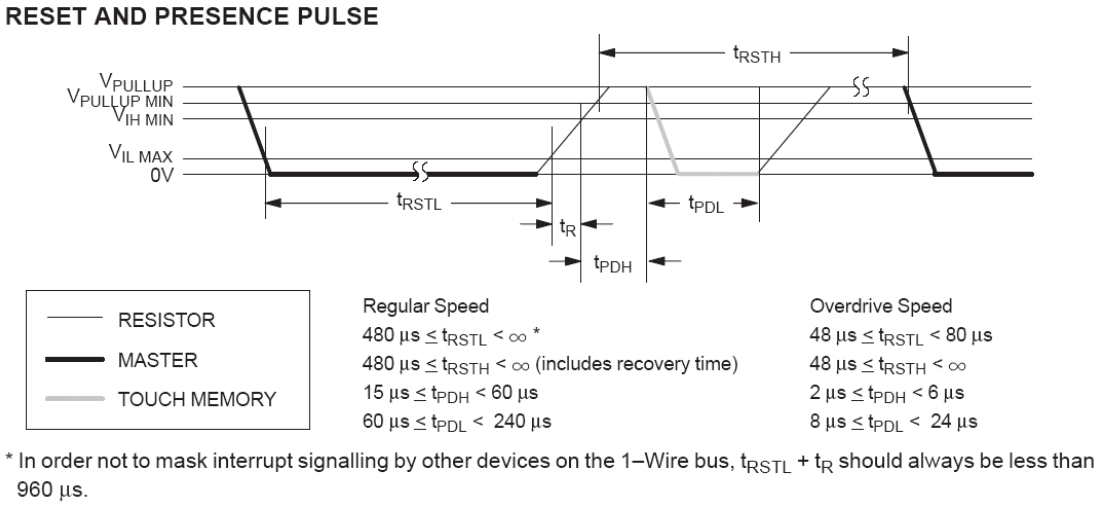
σχήμα 2.9: Ακολουθία παλμών Reset/Presence

ΑΝΙΧΝΕΥΣΗ ΠΑΛΜΟΥ ΠΑΡΟΥΣΙΑΣ

Όπως αναφέρεται παραπάνω ο χρονισμός του **1-Wire** υποστηρίζει έναν παλμό παρουσίας. Ο παλμός αυτός ορίζεται από έναν παλμό χαμηλού δυναμικού ελάχιστης διάρκειας **480 μs** και από ένα παλμό υψηλού δυναμικού τις ίδιας χρονικής διάρκειας. Κατά την διάρκεια του $\mathbf{t_{RSTH}}$ καμία άλλη επικοινωνία δεν επιτρέπεται στην γραμμή **1-Wire**.

Ο παλμός **Reset** έχει σκοπό να παρέχει ξεκάθαρη συνθήκη εκκίνησης που αντικαθιστά κάθε συγχρονισμό χρονοθυρίδας. Όταν ο **Master** στείλει ένα παλμό **Reset** οι συσκευές θα περιμένουν για χρόνο $\mathbf{t_{PDH}}$ και μετά θα απαντήσουν με έναν παλμό παρουσίας διάρκειας $\mathbf{t_{PDL}}$. Αυτό επιτρέπει στον **Master** να διευκρινίσει εύκολα εάν υπάρχουν μία ή περισσότερες συσκευές στην γραμμή.

Οι ονομαστικές τιμές είναι **30μs** για το $\mathbf{t_{PDH}}$ και **120μs** για το $\mathbf{t_{PDL}}$. Η τιμή του $\mathbf{t_{PDH}}$ αντιπροσωπεύει την εσωτερική βάση χρόνου της ποιο γρήγορης συσκευής. Το άθροισμα των τιμών $\mathbf{t_{PDH}}$ και $\mathbf{t_{PDL}}$ είναι **5** φορές η εσωτερική βάση χρόνου τις ποιο αργής συσκευής. Εάν υπάρχει μόνο μια συσκευή στην γραμμή τότε το $\mathbf{t_{PDH}}$ θα είναι ίσο με το $\mathbf{t_{PDL}}$. Οι παραπάνω χρόνοι φαίνονται στο **σχήμα 2.10**.



σχήμα 2.10: Παλμοί Reset και Presence

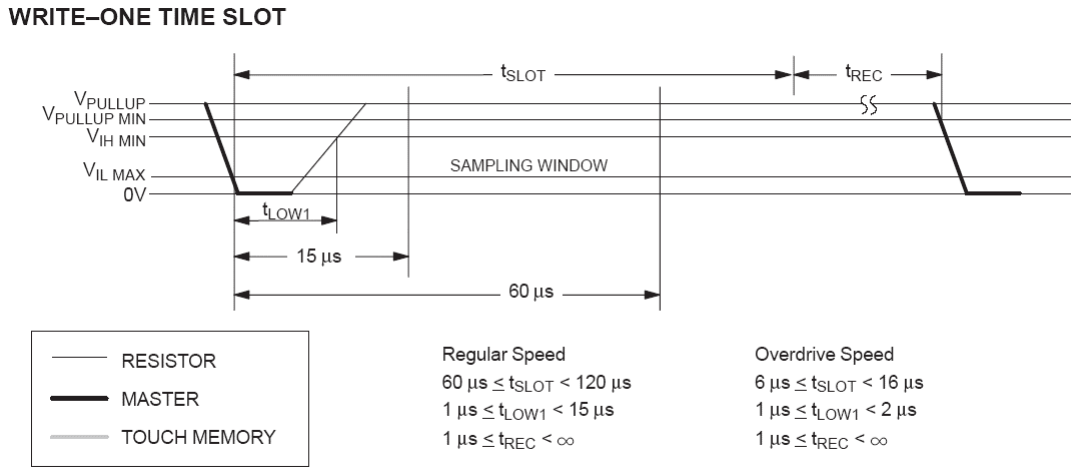
ΧΡΟΝΙΣΜΟΣ

Οι συσκευές **1-Wire** είναι αυτοχρονιζόμενες και απαιτούν για την λειτουργία τους ηλεκτρική επαφή. Η λογική χρονισμού παρέχει τους τρόπους μέτρησης και παραγωγής ψηφιακών παλμών διαφορετικού εύρους. Η μεταφορά των δεδομένων είναι ασύγχρονη και **Half-Duplex**. Τα δεδομένα μπορούν να εκφραστούν σαν εντολές (σύμφωνα με το προκαθορισμένο **Format** το οποίο διευκρινίζεται από το **Family Code**). Λόγω του ότι η κλήση βύθισης εξαρτάται από το χωρητικό φορτίο στα κυκλώματα **Open Drain**, οι συσκευές χρησιμοποιούν αυτή την ακμή για να συγχρονίσουν τα εσωτερικά κυκλώματα χρονισμού τους. Οι συσκευές θεωρούνται ως **Slaves** ενώ ο ελεγκτής που Διαβάζει/Γράφει θεωρείται ως **Master**.

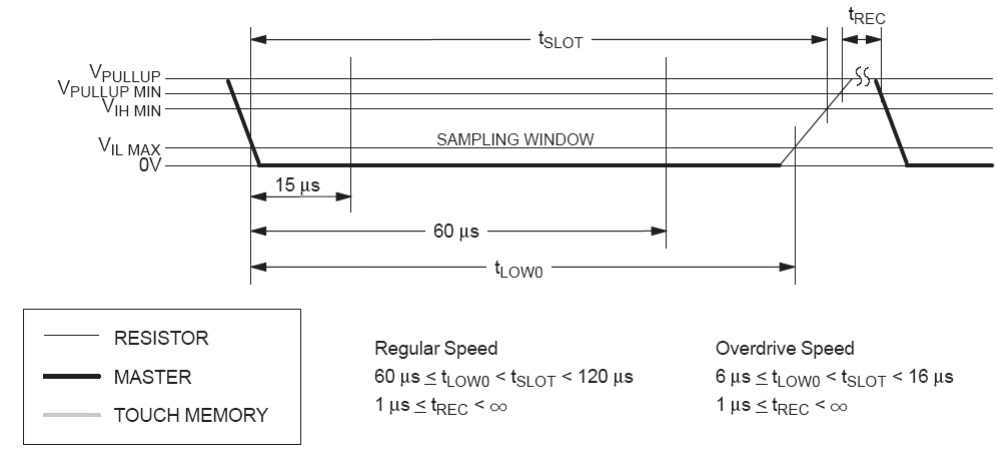
ΧΡΟΝΙΣΜΟΣ ΕΓΓΡΑΦΗΣ

Ο χρονισμός της εγγραφής χωρίζεται σε χρονοθυρίδες οι οποίες είναι συγκεκριμένα χρονικά διαστήματα και ξεκινά από την στιγμή όπου ο ελεγκτής κατεβάζει την γραμμή δεδομένων από υψηλό δυναμικό σε χαμηλό δυναμικό. Υπάρχουν δύο τύποι χρονισμού εγγραφής: εγγραφή με **‘’1’’** και με **‘’0’’**. Οι χρονοθυρίδες πρέπει να έχουν ελάχιστη διάρκεια **60 μs** και **1 μs** χρόνο ανάκτησης ανάμεσα στους κύκλους εγγραφής. Το αισθητήριο δειγματοληπτεί την **I/O** γραμμή ανά τακτά χρονικά διαστήματα, από **15μs** έως **60μs**, όταν αυτή περάσει από υψηλό σε χαμηλό δυναμικό. Όταν η γραμμή βρίσκεται σε υψηλό δυναμικό το αισθητήριο θα γράψει **‘’1’’** και όταν βρίσκεται σε χαμηλό δυναμικό θα γράψει **‘’0’’**.

Για να δημιουργήσει ο ελεγκτής μία χρονοθυρίδα εγγραφής **“1”**, πρέπει να κατεβάσει την γραμμή δεδομένων σε χαμηλό δυναμικό και να την απελευθερώσει για να επιστρέψει σε υψηλό δυναμικό μέσα σε **15 μs**, μετά την έναρξη της χρονοθυρίδας εγγραφής. Για να δημιουργηθεί μία χρονοθυρίδα εγγραφής **“0”** ο ελεγκτής πρέπει να κατεβάσει την γραμμή σε χαμηλό δυναμικό και να την κρατήσει στο επίπεδο αυτό για **60 μs**. Όπως φαίνεται στο **σχήμα 2.11** και **σχήμα 2.12**.



σχήμα 2.11: Χρονοθυρίδα εγγραφής "1"



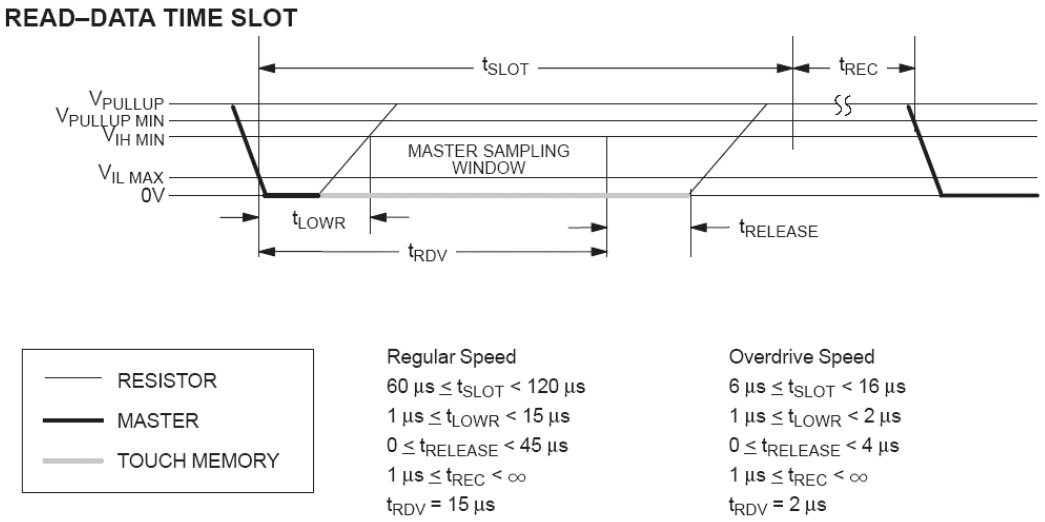
σχήμα 2.12: χρονοθυρίδα εγγραφής "0"

Η διάρκεια του ενεργού μέρους μιας χρονοθυρίδας μπορεί να επεκταθεί πάνω από **60 μs**. Η μέγιστη επέκταση περιορίζεται από το γεγονός ότι η περίοδος **8** ενεργών χρονοθυρίδων (**480μs**) χαμηλού παλμού, ορίζεται ως παλμός **Reset**. Για το λόγο αυτό ορίζεται ως μέγιστη επέκταση τα **120 μs** για να ελαχιστοποιήσουμε την πιθανότητα παρερμηνείας με ένα παλμό **Reset**.

Στο τέλος του ενεργού κύκλου κάθε χρονοθυρίδας οι συσκευές χρειάζονται χρόνο ανάκτησης $\mathbf{t_{REC}}$ τουλάχιστον **1μs** για να είναι έτοιμες για το επόμενο **bit**. Ο χρόνος της ανάκτησης μπορεί να θεωρηθεί ως το ανενεργό μέρος της χρονοθυρίδας και πρέπει να προστεθεί στην συνολική διάρκεια της χρονοθυρίδας για να εξασφαλίσει το χρόνο που χρειάζεται η μεταφορά ενός **bit**.

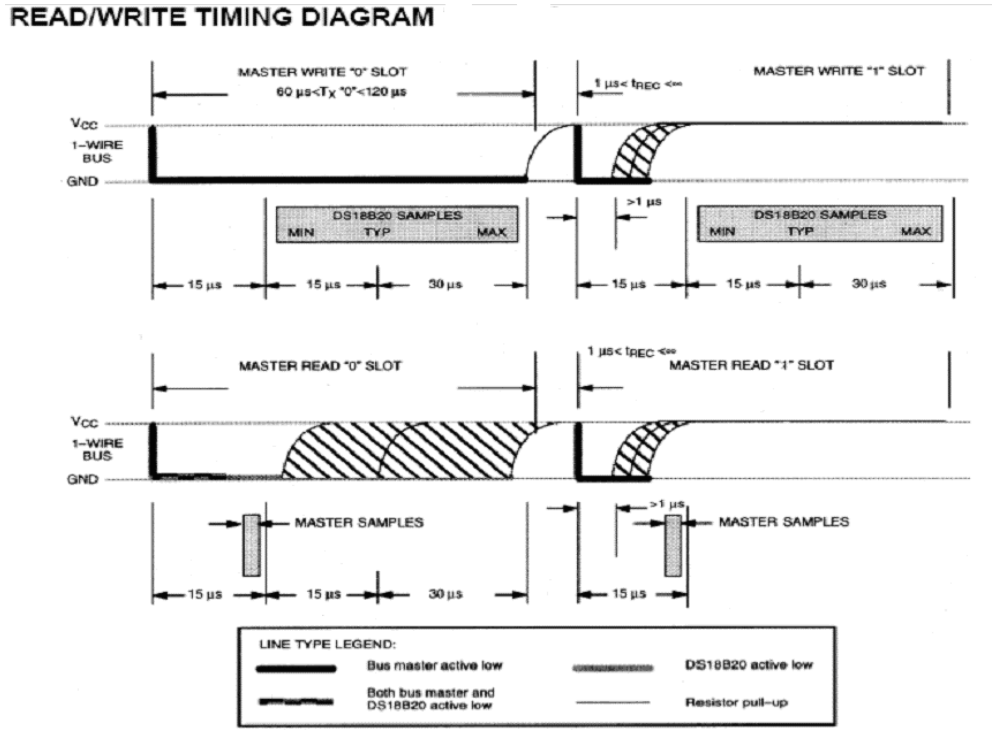
ΧΡΟΝΙΣΜΟΣ ΑΝΑΓΝΩΣΗΣ

Ο ελεγκτής δημιουργεί χρονοθυρίδες ανάγνωσης όταν πρόκειται να διαβάσει δεδομένα από τις συσκευές. Μία χρονοθυρίδα ανάγνωσης ξεκινάει από την στιγμή που ο ελεγκτής κατεβάσει την γραμμή δεδομένων, από υψηλό δυναμικό σε χαμηλό δυναμικό και θα την κρατήσει για **1μs** τουλάχιστον. Τα δεδομένα από τις συσκευές, θα είναι έγκυρα, μετά την πάροδο **15 μs** από την στιγμή που ο παλμός της χρονοθυρίδας διαβάσματος αρχίσει να πέφτει. Επομένως ο ελεγκτής θα πρέπει να απελευθερώσει την γραμμή από το χαμηλό δυναμικό για να διαβάσει την κατάσταση της, **15 μs** μετά την αρχή της χρονοθυρίδας. Με το τέλος της χρονοθυρίδας διαβάσματος η επαφή **I/O** θα περάσει σε υψηλό δυναμικό μέσο του αντιστάτη **Pull-Up**. Οι χρονοθυρίδες πρέπει να έχουν ελάχιστη διάρκεια **60 μs** και **1 μs** χρόνο ανάκτησης ανάμεσα στους κύκλους ανάγνωσης. Όπως φαίνεται στο **σχήμα 2.13**.

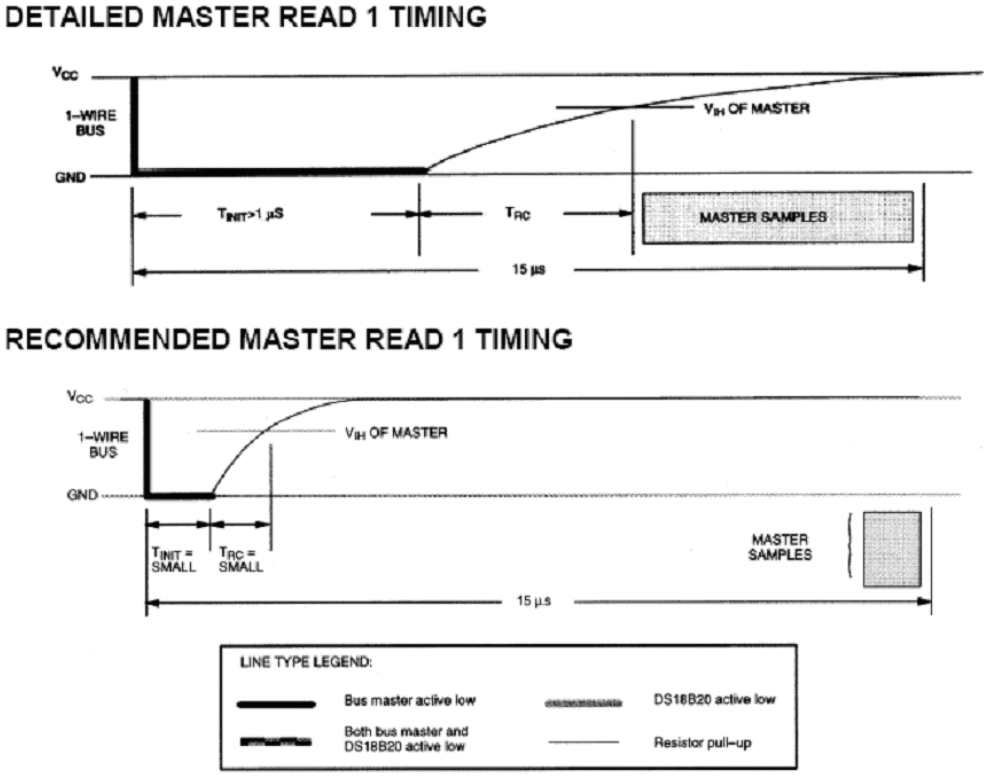


σχήμα 2.13: χρονοθυρίδα ανάγνωσης

Στο **σχήμα 2.14** φαίνεται ότι το άθροισμα των **TINIT**, **TRC** και **TSAMPLE** πρέπει να είναι μικρότερο από **15 μs**. Στο **σχήμα 2.15** φαίνεται ότι το χρονικό περιθώριο αυξάνεται κρατώντας τα **TINIT**, **TRC** όσο το δυνατόν μικρότερα και προσδιορίζοντας στον ελεγκτή τον χρόνο δειγματοληψίας, προς το τέλος της περιόδου των **15 μs**.



σχήμα 2.14: χρονοθυρίδες εγγραφής και ανάγνωσης



σχήμα 2.15

NETWORK LAYER

Αυτό το επίπεδο παρέχει την ταυτοποίηση των συσκευών και τις ικανότητες του υφιστάμενου δικτύου. Κάθε συσκευή έχει τον δικό της μοναδικό αριθμό ταυτοποίησης ο οποίος χαράσσεται στο τμήμα της **ROM** του **Chip**, κατά την διαδικασία κατασκευής του. Εξαιτίας της **ROM**, όλες οι εντολές που λαμβάνουν χώρα στο **Network Layer** ονομάζονται εντολές λειτουργίας **ROM**.

ΕΝΤΟΛΕΣ ΛΕΙΤΟΥΡΓΙΑΣ ROM

Αφού έχει ολοκληρωθεί η διαδικασία της αρχικοποίησης και έχουν οριστεί οι χρονοθυρίδες εγγραφής και ανάγνωσης, ο **Master** μπορεί να στείλει μία από τις παρακάτω εντολές λειτουργίας της **ROM** (**ROM Function Commands**), όλες οι εντολές αποτελούνται από **8-bit**.

Read ROM [33h]

Η εντολή αυτή δίνει την δυνατότητα στον **Master** να διαβάσει τον αύξοντα αριθμό (πυριτίου) της **ROM** του αισθητηρίου. Η εντολή αυτή μπορεί να χρησιμοποιηθεί, εάν στον δίαυλο βρίσκεται μόνο μία συσκευή. Στην περίπτωση που βρίσκονται παραπάνω από μία συσκευές στην γραμμή, θα παρουσιαστεί πρόβλημα, διότι θα απαντήσουν όλες την ίδια χρονική στιγμή. (λόγω του ότι η διάταξη είναι **Open Drain** θα έχουμε αποτέλεσμα πύλης **AND**).

Match ROM [55h]

Μετά την έκδοση της εντολής **Match ROM**, ο ελεγκτής διαύλου στέλνει και την ακολουθία των **64-bit** της **ROM**, όπου τον καθιστά ικανό να διευθετήσει και να απομονώσει μία συγκεκριμένη συσκευή σε δίαυλο όπου υπάρχουν πολλές. Μόνο η συσκευή με τον ίδιο **64-bit ROM** θα ανταποκριθεί στις εντολές που θα ακολουθήσουν, οι υπόλοιπες συσκευές στον δίαυλο θα περιμένουν για ένα παλμό **Reset**. Η εντολή αυτή μπορεί να εκδοθεί σε δίαυλο όπου υπάρχουν μία ή περισσότερες συσκευές.

Overdrive Match ROM [69h]

Η εντολή αυτή έχει την ίδια ακριβώς λειτουργία με την **Match ROM** με την διαφορά ότι εκτελείται σε **Overdrive Mode**. Εφαρμόζουμε αυτή την εντολή μόνο σε συσκευές που υποστηρίζουν **Overdrive Mode**.

Skip ROM [CCh]

Η εντολή αυτή χρησιμοποιείται για να επιτρέψει την άμεση πρόσβαση του ελεγκτή στις **Function Commands** μίας και μόνο συσκευής, χωρίς να χρειαστεί η έκδοση του **64-bit** κωδικού. Στην περίπτωση που στον δίαυλο υπάρχουν περισσότερες συσκευές και ο ελεγκτής εκδώσει μία εντολή ανάγνωσης μετά την εντολή **Skip ROM**, θα δημιουργηθεί σύγχυση δεδομένων καθώς όλες οι συσκευές θα ανταποκριθούν ταυτόχρονα.

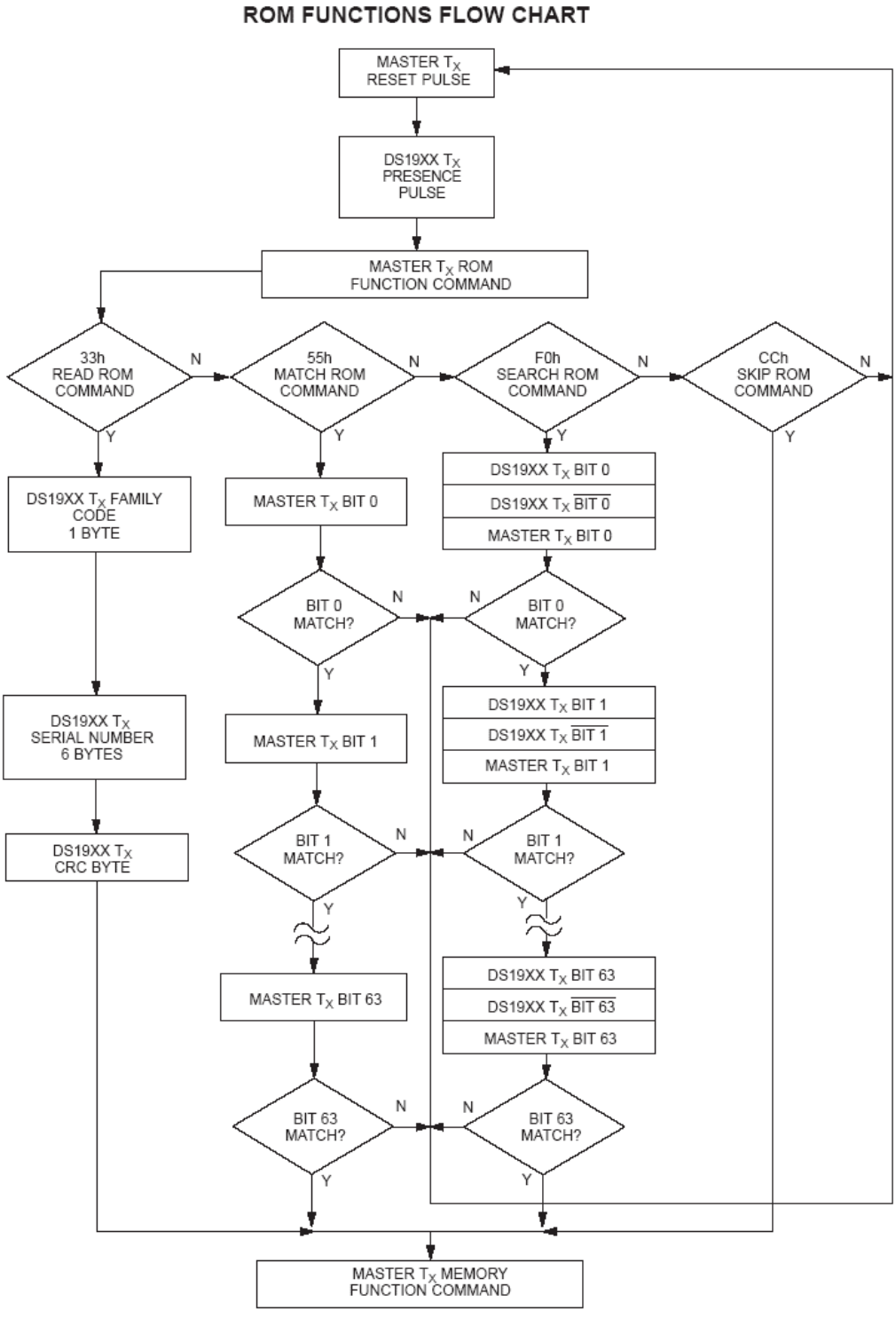
Overdrive Skip ROM [3Ch]

Η εντολή αυτή έχει την ίδια ακριβώς λειτουργία με την **Skip ROM** με την διαφορά ότι εκτελείται σε **Overdrive Mode**. Εφαρμόζουμε αυτή την εντολή μόνο σε συσκευές που υποστηρίζουν **Overdrive Mode**.

Search ROM [F0h]

Όταν ένα σύστημα αρχικοποιηθεί, ο ελεγκτής διαύλου είναι πιθανό να μην γνωρίζει τον ακριβή αριθμό των συσκευών που παρουσιάζονται στον **1-Wire** δίαυλο ή τους **64-bit ROM** κωδικούς τους. Η **Search ROM** εντολή δίνει την δυνατότητα στον ελεγκτή διαύλου να χρησιμοποιήσει μία διαδικασία ανεύρεσης όπου θα διευκρινίσει τους **64-bit ROM** κωδικούς όλων των συσκευών.

Στο **σχήμα 2.16** βλέπουμε το διάγραμμα ροής των εντολών λειτουργίας **ROM**.



σχήμα 2.16: διάγραμμα ροής των εντολών λειτουργίας ROM

ΠΑΡΑΔΕΙΓΜΑ ΑΝΕΥΡΕΣΗΣ ROM

Η διαδικασία ανεύρεσης **ROM** είναι μια ρουτίνα επανάληψης τριών απλών βημάτων: ανάγνωση **bit**, ανάγνωση συμπληρωματικού **bit**, καταχώριση της επιθυμητής αξίας του συγκεκριμένου **bit**. Ο ελεγκτής διαύλου εκτελεί αυτήν την ρουτίνα για κάθε **bit** της **ROM**. Μετά την ολοκλήρωση της ρουτίνας και για τα **64 bit** της **ROM**, ο ελεγκτής διαύλου είναι σε θέση να γνωρίζει το περιεχόμενο της **ROM** μίας συσκευής. Ο υπόλοιπος αριθμός των συσκευών και οι **64 bit ROM** κωδικοί τους μπορούν να ταυτοποιηθούν επαναλαμβάνοντας τη διαδικασία αυτή.

Στο παρακάτω παράδειγμα κατά τη διαδικασία ανεύρεσης **ROM** υποθέτουμε ότι τέσσερις διαφορετικές συσκευές είναι συνδεδεμένες στον ίδιο δίαυλο **1-WIRE**.

Το περιεχόμενο των τεσσάρων συσκευών φαίνεται παρακάτω:

ROM1 00110101...
ROM2 10101010...
ROM3 11110101...
ROM4 00010001...

Η διαδικασία ανεύρεσης είναι η ακόλουθη:

1. Ο ελεγκτής διαύλου ξεκινά την διαδικασία αρχικοποίησης αποστέλλοντας ένα παλμό **Reset**. Οι συσκευές απαντούν αποστέλλοντας ταυτόχρονα παλμούς παρουσίας.

2. Τότε ο ελεγκτής θα αποστείλει την εντολή **Search ROM** στον **1-Wire** δίαυλο.

3. Ο ελεγκτής διαβάζει ένα **bit**. Κάθε συσκευή θα απαντήσει τοποθετώντας την τιμή του πρώτου **bit** τις αντίστοιχης **ROM** στον δίαυλο **1-Wire**. Οι **ROM1** και **ROM4** θα τοποθετήσουν **0** στον δίαυλο (θα τραβήξουν τη γραμμή σε χαμηλό δυναμικό). Οι **ROM2** και **ROM3**

θα τοποθετήσουν **1** στον δίαυλο (επιτρέποντας στην γραμμή να παραμείνει σε υψηλό δυναμικό). Το αποτέλεσμα είναι μία λογική **AND** όλων των συσκευών που είναι συνδεδεμένες στην γραμμή, επομένως ο ελεγκτής βλέπει **0**. Ο ελεγκτής διαύλου διαβάζει άλλο ένα **bit**. Κατά τη διάρκεια εκτέλεσης της εντολής **Search ROM**, όλες οι συσκευές που βρίσκονται στον δίαυλο απαντούν στη δεύτερη ανάγνωση τοποθετώντας το συμπλήρωμα του πρώτου **bit** της αντίστοιχης **ROM** τους στον δίαυλο. Οι **ROM1** και **ROM4** θα τοποθετήσουν **1** στον δίαυλο (επιτρέποντας στην γραμμή να παραμείνει σε υψηλό δυναμικό). Οι **ROM2** και **ROM3** θα τοποθετήσουν **0** στον δίαυλο, συνεπώς θα τραβήξουν τη γραμμή σε χαμηλό δυναμικό. Ο ελεγκτής θα δει ξανά **0** και για το συμπλήρωμα του πρώτου **bit** της **ROM**. Ο ελεγκτής έχει αποφασίσει ότι υπάρχουν συσκευές στον δίαυλο που έχουν ως πρώτο **bit 0** και κάποιες που έχουν **1**.

Τα δεδομένα που αντλούνται από τις δύο αναγνώσεις της ρουτίνας έχουν τις παρακάτω ερμηνείες:

- **00** Υπάρχουν συσκευές συνδεδεμένες που έχουν συγκρουόμενα **bit** σ' αυτή τη θέση.
- **01** Όλες οι συσκευές που παραμένουν συνδεδεμένες έχουν "**0**" σ' αυτή τη θέση.
- **10** Όλες οι συσκευές που παραμένουν συνδεδεμένες έχουν "**1**" σ' αυτή τη θέση.
- **11** Δεν υπάρχουν άλλες συσκευές συνδεδεμένες στον δίαυλο **1-Wire**.

4. Ο ελεγκτής διαύλου γραφεί "**0**". Με αυτόν τον τρόπο απομακρύνονται οι **ROM2** και **ROM3** από τη διαδικασία ανεύρεσης, αφήνοντας μόνο τις **ROM1** και **ROM4** συνδεδεμένες στον δίαυλο.

5. Ο ελεγκτής εκτελεί δύο ακόμα αναγνώσεις και λαμβάνει "**0**" και "**1**". Αυτό υποδεικνύει ότι όλες οι συσκευές που παραμένουν συνδεδεμένες στο δίαυλο έχουν "**0**' στο δεύτερο **bit** της **ROM** τους.

6. Έπειτα ο ελεγκτής γράφει “**0**” και κρατάει και τις δύο **ROM** ενεργές (και η **ROM1** και η **ROM4** θα παραμείνουν συνδεδεμένες στο δίαυλο).

7. Ό ελεγκτής εκτελεί δύο αναγνώσεις και λαμβάνει “**0**” και “**0**”. Αυτό υποδεικνύει ότι υπάρχουν συσκευές συνδεδεμένες που έχουν και “**0**” και “**1**” στο τρίτο **bit** της **ROM** τους.

8. Ο ελεγκτής γράφει “**0**”. Η **ROM1** απενεργοποιείται, αφήνοντας την **ROM4** ως μοναδική συσκευή συνδεδεμένη στον δίαυλο.

9. Ο ελεγκτής διαβάζει τα υπόλοιπα **bit** της **ROM4**. Και έτσι ολοκληρώνεται το πρώτο πέρασμα και προσδιορίζεται η ταυτότητα μίας από τις συσκευές που βρίσκονται στον δίαυλο.

10. Ο ελεγκτής διαύλου ξεκινά μια νέα ακολουθία “**Search ROM**” επαναλαμβάνοντας τα βήματα **1** έως **7**.

11. Ο ελεγκτής γράφει “**1**”. Η **ROM4** απενεργοποιείται, αφήνοντας μόνο την **ROM1** συνδεδεμένη στον δίαυλο.

12. Ο ελεγκτής διαβάζει τα υπόλοιπα **bit** της **ROM1**. Και έτσι ολοκληρώνεται το δεύτερο πέρασμα, στο οποίο προσδιορίζεται η ταυτότητα άλλης μίας συσκευής του δίαυλου.

13. Ο ελεγκτής διαύλου ξεκινά μια νέα “**Search ROM**” επαναλαμβάνοντας τα βήματα **1** έως **3**.

14. Ο ελεγκτής γράφει “**1”**. Η **ROM1** και η **ROM4** απενεργοποιούνται, αφήνοντας μόνο την **ROM2** και την **ROM3** συνδεδεμένες στο σύστημα.

15. Ό ελεγκτής εκτελεί δύο αναγνώσεις και λαμβάνει “**0**” και “**0**”.

16. Ο ελεγκτής γράφει “**0**”. Η **ROM3** απενεργοποιείται, αφήνοντας μόνο την **ROM2** συνδεδεμένη στον δίαυλο.

17. Ο ελεγκτής διαβάζει τα υπόλοιπα **bit** της **ROM2**. Με αυτόν τον τρόπο ολοκληρώνεται το τρίτο πέρασμα, στο οποίο προσδιορίζεται η ταυτότητα άλλης μίας συσκευής του δίαυλου.

18. Ο ελεγκτής διαύλου ξεκινά μια νέα “**Search ROM**” επαναλαμβάνοντας τα βήματα **13** έως **15**.

19. Ο ελεγκτής γράφει “**1**”. Η **ROM2** απενεργοποιείται, αφήνοντας μόνο την **ROM3** συνδεδεμένη στον δίαυλο.

20. Ο ελεγκτής διαβάζει τα υπόλοιπα **bit** της **ROM3**, ολοκληρώνοντας το τέταρτο πέρασμα, στο οποίο προσδιορίζεται η ταυτότητα άλλης μίας συσκευής του δίαυλου.

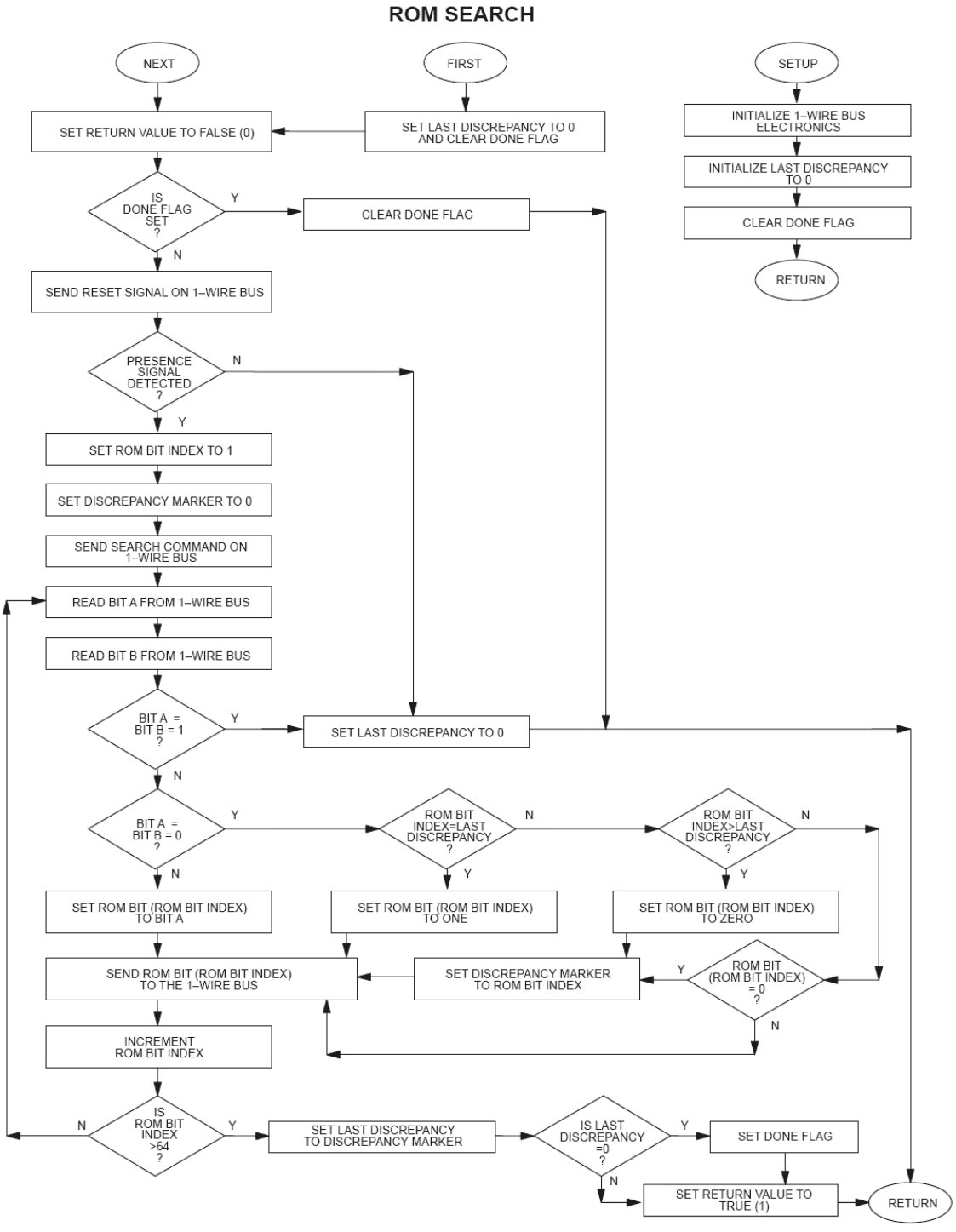
Ο ελεγκτής διαύλου μαθαίνει ένα μοναδικό αριθμό ταυτοποίησης σε κάθε αναζήτηση “**Search ROM**”. Ο χρόνος που απαιτείται για τον προσδιορισμό του μοναδικού κωδικού **ROM** μίας συσκευής είναι:

960 μs + (8 + 3 x 64) 61 μs = 13.16 ms

Αρχικοποίηση = 960 μs
Αποστολή Search ROM = 8 bit * 61μs
Ρουτίνα = 3 bit * 64 bit *61μs
Time Slot = 61μs

Κατά συνέπεια ο ελεγκτής είναι ικανός να ταυτοποιήσει **75** διαφορετικές συσκευές το δευτερόλεπτο. Στο **σχήμα 2.17** φαίνεται το διάγραμμα ροής της διαδικασίας ανεύρεσης **ROM**.

ROM SEARCH FLOW CHART



σχήμα 2.17: διάγραμμα ροής ανεύρεσης της ROM.

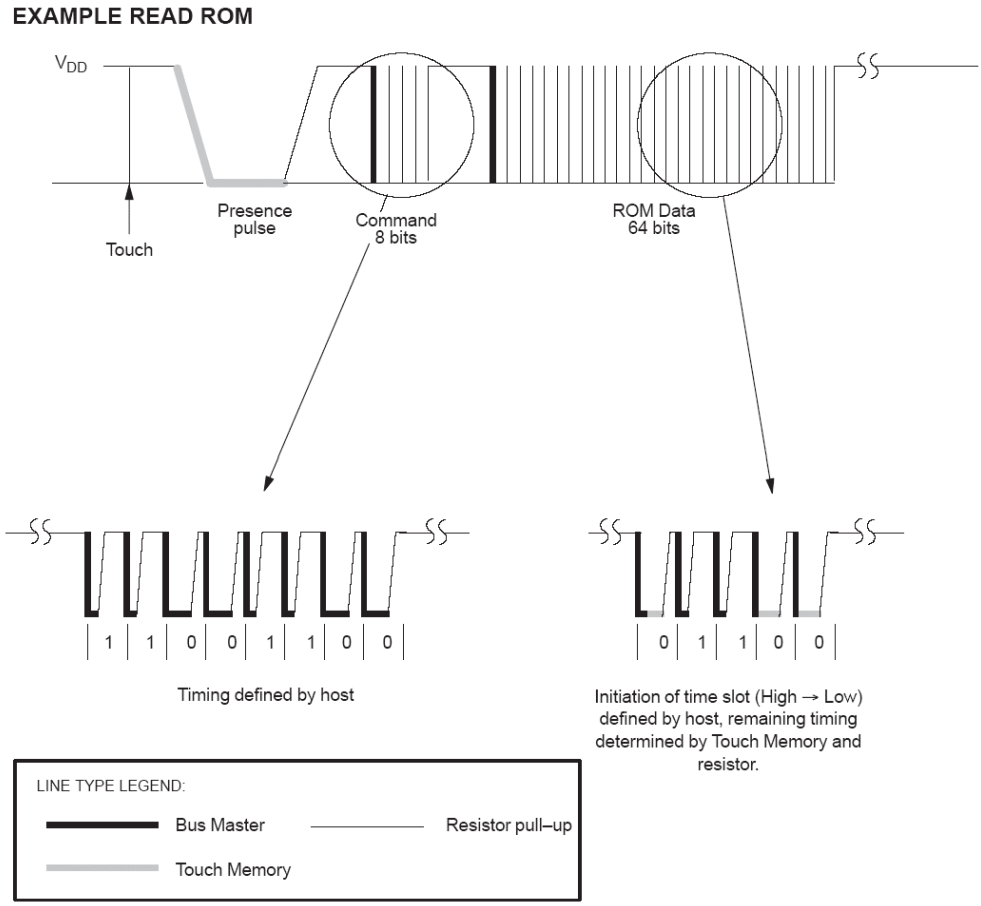
TRANSPORT LAYER

Αυτό το επίπεδο είναι υπεύθυνο για την μεταφορά δεδομένων μεταξύ των συσκευών και **Master** και τις μεταφοράς δεδομένων από το **Scratchpad** και τους ειδικούς καταχωρητές στον τελικό προορισμό αποθήκευσης.

ΜΕΤΑΦΟΡΑ ΔΕΔΟΜΕΝΩΝ

Μετά τον παλμό παρουσίας, οι συσκευές περιμένουν μία εντολή. Κάθε εντολή γράφεται από μία αλληλουχία χρονοθυρίδων εγγραφής λογικού **"1"** και λογικού **"0"** έως ότου δημιουργηθεί ένα ολοκληρωμένο **Byte** εντολής. Η αντίστροφη διαδικασία της μεταφοράς δεδομένων από τις συσκευές στον ελεγκτή χρησιμοποιεί τους ίδιους κανόνες χρονισμού για εγγραφή **"1"** και **"0"** αντίστοιχα. Λόγω του ότι οι συσκευές είναι σχεδιασμένες ως **Slave**, αφήνουν στον **Maser** να ορίσει την έναρξη κάθε χρονοθυρίδας, και για να γίνει αυτό ο **Master** αρχικοποιεί την χρονοθυρίδα εγγραφής **"1"** για να διαβάσει 1 **bit** δεδομένων. Εάν η συσκευή πρέπει να στείλει ένα **"1"** πρέπει να περιμένει την αρχικοποίηση της επόμενης χρονοθυρίδας. Εάν έχει να στείλει **"0"** θα κρατήσει τη γραμμή δεδομένων σε χαμηλό δυναμικό για συγκεκριμένο χρόνο, ακόμα και αν την έχει απελευθερώσει ο **Master**.

Ένα παράδειγμα μίας ολοκληρωμένης ακολουθίας εντολής που αρχίζει με παλμό παρουσίας και τελειώνει με δεδομένα φαίνεται στο **σχήμα 2.18**. Η δραστηριότητα του **Master** είναι σχεδιασμένη με έντονες γραμμές. Με γκρι είναι σχεδιασμένη η ανταπόκριση της συσκευής. Οι λεπτές γραμμές δείχνουν ότι τίποτα δεν είναι ενεργό. Η γραμμή περνάει σε υψηλό δυναμικό μέσω ενός αντιστάτη **Pull-Up**.



σχήμα 2.18: ολοκληρωμένη ακολουθία εντολής.

Οι εντολές που λαμβάνουν χώρα στο **Transport Layer** ονομάζονται εντολές λειτουργίας μνήμης. Οι βασικές εντολές είναι:

Read Scratchpad

Με την εντολή αυτή διαβάζουμε τα περιεχόμενα του **Scratchpad**, ξεκινώντας από το πρώτο (**Byte 0**) μέχρι το τελευταίο **CRC**. Εάν δεν θέλουμε να διαβάσουμε όλα τα δεδομένα του **Scratchpad** η διαδικασία μπορεί να τερματιστεί με την έκδοση ενός παλμού **Reset** από τον ελεγκτή.

Write Scratchpad

Με την εντολή αυτή γράφουμε στο **Scratchpad** της συσκευής, η διαδικασία αυτή πρέπει να ολοκληρωθεί πριν την έκδοση **Reset**.

Copy Scratchpad

Η εντολή αυτή αντιγράφει το **Scratchpad** σε μία μνήμη της συσκευής αποθηκεύοντας έτσι τα δεδομένα του **Scratchpad** σε μία αμετάβλητη μνήμη για να μην χαθούν τα δεδομένα. Εάν ο ελεγκτής κάνει ανάγνωση της γραμμής ενώ η διαδικασία της αντιγραφής είναι σε εξέλιξη, η συσκευή θα επιστρέψει ‘’**0**’’ καθ’ όλη την διάρκεια της αντιγραφής, μόλις η αντιγραφή τελειώσει θα επιστρέψει ‘’**1**’’. Στην περίπτωση που χρησιμοποιούμε παρασιτική τροφοδότηση, ο ελεγκτής θα πρέπει να εφαρμόσει ισχυρό **Pull-Up** για τουλάχιστον **10ms** αμέσως μετά την έκδοση της εντολής αυτής.

Οι παραπάνω εντολές είναι γενικές και χρησιμοποιούνται από όλες τις συσκευές. Ανάλογα με τις λειτουργίες της κάθε συσκευής το **Command Set** διαφοροποιείται για να καλύψει τις ανάγκες της.

PRESENTATION LAYER

Τα επίπεδα **Link**, **Network** και **Transport** είναι θεμελιώδη για το **Presentation Layer**. Αυτό το επίπεδο παρέχει ένα σύστημα αρχείων παρόμοιο με το **DOS** και υποστηρίζει λειτουργίες όπως **Format**, **Directory**, **Type**, **Copy**, **Delete**, **Optimize** και ελέγχους ακεραιότητας. Αυτό το επίπεδο εφαρμόζεται μόνο όταν χρησιμοποιούμε **1-Wire** μνήμες.

ΚΕΦΑΛΑΙΟ 3

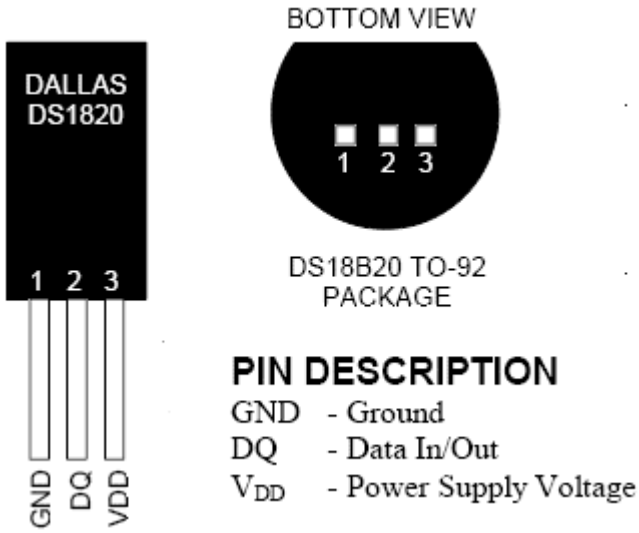
ΠΑΡΑΔΕΙΓΜΑ ΕΦΑΡΜΟΓΗΣ ΜΕΤΡΗΣΗΣ ΘΕΡΜΟΚΡΑΣΙΑΣ ΜΕ ΤΟ DS18B20

ΤΟ ΨΗΦΙΑΚΟ ΑΙΣΘΗΤΗΡΙΟ ΘΕΡΜΟΚΡΑΣΙΑΣ DS18B20

ΓΕΝΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΤΟΥ ΑΙΣΘΗΤΗΡΙΟΥ

Το ψηφιακό αισθητήριο **DS18B20** ανήκει στην κατηγορία των έξυπνων αισθητηρίων, διότι μετατρέπει τη θερμότητα σε ψηφιακή μορφή. Ένα κύριο χαρακτηριστικό του είναι ο μοναδικός **64 bit** αριθμός ταυτότητας που το κάνει να ξεχωρίζει από τα υπόλοιπα αισθητήρια της ίδιας οικογένειας. Αυτό έχει ως αποτέλεσμα τη χρήση δύο ή περισσοτέρων αισθητηρίων πάνω στον ίδιο δίαυλο, όπου χρησιμοποιούν το πρωτόκολλο επικοινωνίας **1-Wire bus**.

Ένα άλλο χαρακτηριστικό του αισθητηρίου είναι το θερμοκρασιακό του εύρος, που κυμαίνεται από **-55 ºC** έως **+125 ºC** με ανάλυση **9, 10, 11, 12 bit,** δηλαδή ανά **0.5, 0.25, 0.125, 0.0625 ºC** αντίστοιχα. Το **DS18B20** μάς δίνει τη δυνατότητα να του θέτουμε ανώτατη και κατώτατη θερμοκρασία σε δύο ξεχωριστούς καταχωρητές που βρίσκονται μέσα στο αισθητήριο, πράγμα που απλοποιεί την υλοποίηση συστημάτων επιτήρησης και ελέγχου θερμοκρασίας. Επίσης το αισθητήριο μπορεί να λειτουργήσει με δύο τρόπους: είτε με χρήση εξωτερικής τροφοδοσίας είτε με παρασιτική τροφοδοσία (τροφοδοσία από την γραμμή δεδομένων). Στο **σχήμα 3.1** φαίνεται το αισθητήριο θερμοκρασίας **DS18B20.**

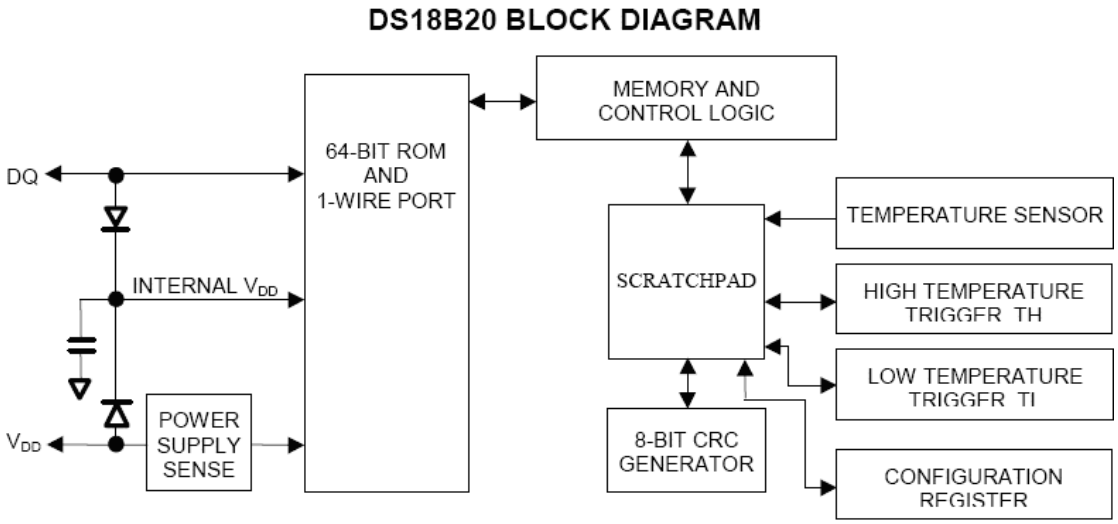


σχήμα 3.1: Αισθητήριο θερμοκρασίας DS18B20.

Η ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΟΥ DS18B20

Στο μπλοκ διάγραμμα του **σχήματος 3.2**, φαίνεται η βασική δομή του **DS18B20**, το οποίο αποτελείτε από τέσσερις κύριες μονάδες

- Από μία μνήμη **Lasered ROM 64-bit**
- Από μία βαθμίδα ελέγχου μνήμης
- Από μία μνήμη **Scratchpad**
  - Ένα αισθητήρα θερμοκρασίας
  - Δύο αμετάβλητους σκανδαλιστές συναγερμού θερμοκρασίας: **TH** και **TL**.
  - Και έναν καταχωρητή ρυθμίσεων.
  - Από μία γεννήτρια **CRC**



σχήμα 3.2: Η βασική δομή του DS18B20

64-BIT LASERED ROM

Κάθε **DS18B20** περιέχει έναν μοναδικό **ROM** κωδικό **64-bit**, τα πρώτα **8 bit** είναι κωδικός της οικογένειας **1-Wire** και μας υποδεικνύουν πια συσκευή έχουμε στο **1-Wire bus**. Για τα αισθητήρια θερμοκρασίας η τιμή του κωδικού αυτού στο δεκαεξαδικό σύστημα είναι (**28h**). Τα επόμενα **48 bits** πού ακολουθούν είναι ένας αύξων αριθμός, μοναδικός για κάθε αισθητήριο, που ουσιαστικά είναι η ταυτότητα του. Τα τελευταία **8 bits** είναι ένας **CRC** έλεγχος

για τα πρώτα **56 bits**. Τα τμήματα, των **64 bits** και ο έλεγχος λειτουργίας, της **ROM**. Επιτρέπουν στο **DS18B20** να λειτουργεί ως μία συσκευή που ακολουθεί το **1-wire** πρωτόκολλο **(σχήμα 3.3).**

(σχήμα 3.3)

MEMORY AND CONTROL LOGIC

Αποτελεί το συνδετικό κρίκο μεταξύ της προηγούμενης βαθμίδας με τις υπόλοιπες.

Η ΟΡΓΑΝΩΣΗ ΤΗΣ ΜΝΗΜΗΣ

Στο **σχήμα 3.4** βλέπουμε πως είναι οργανωμένη εσωτερικά η μνήμη του **DS18B20**. Η μνήμη αποτελείται από ένα **Scratchpad RAM** και μία μη πτητική **E^2** στην οποία αποθηκεύουμε της στάθμες των θερμοκρασιών στους καταχωρητές **TH** και **TL** που θέλουμε να ελέγχουμε και ο καταχωρητής ρυθμίσεως. Το **Scratchpad** μας βοηθάει να εξασφαλίσουμε την ακεραιότητα των δεδομένων μας όταν επικοινωνούμε μέσο του **1-Wire bus**.

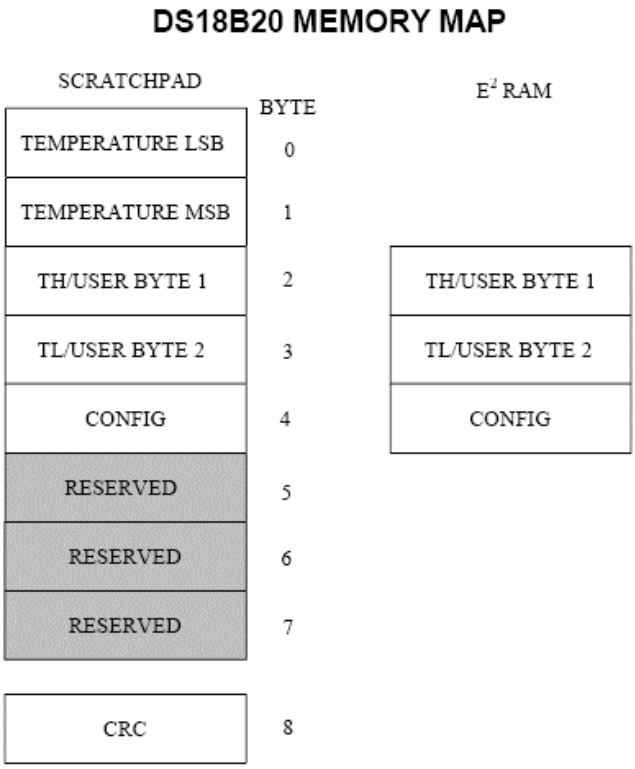
Με την εντολή **Write Scratchpad** τα δεδομένα γράφονται πρώτα στο **Scratchpad** και αφού τα επαληθεύσουμε με την εντολή **Read Scratchpad**, τα μεταφέρουμε στην μη πτητική μνήμη **E^2** με την εντολή **Copy Scratchpad**. Η διαδικασία αυτή μας εξασφαλίζει την ακεραιότητα των δεδομένων όταν τροποποιούμε την μνήμη ή όταν εκτελείται **Reset**.

Το **Scratchpad** αποτελείτε από **9 Bytes** μνήμης, τα δύο πρώτα (**LSB** και **MSB**) περιέχουν την πληροφορία της μετρηθήσας τιμής της θερμοκρασίας αντίστοιχα. Το τρίτο και το τέταρτο περιέχουν τις τιμές των θερμοκρασιών που έχουμε ορίσει στους **TH** και **TL** όπου ανανεώνονται σε κάθε **Reset**. Το πέμπτο **Byte** είναι ο καταχωρητής ρυθμίσεων, με τον οποίο θα ασχοληθούμε εκτενέστερα παρακάτω, ο οποίος επίσης ανανεώνεται με κάθε **Reset**. Τα **6**,**7**

και **8 Bytes** χρησιμοποιούνται από το αισθητήριο για εσωτερικούς υπολογισμούς του αισθητηρίου. Και το **9** και τελευταίο **Byte** περιέχει το **CRC**, του οποίου η τιμή έχει υπολογιστεί από τα **8** προηγούμενα **Bytes**.

SCRATCHPAD

Λειτουργεί ως **RAM** του συστήματος. Εκεί αποθηκεύονται όλα τα προσωρινά δεδομένα όπως θερμοκρασία, καταχωρητές μέγιστης και ελάχιστης θερμοκρασίας, ο **Configuration Register**, και το **CRC (σχήμα 3.4)**.



σχήμα 3.4: Εσωτερική μνήμη του DS18B20.

ΔΙΑΔΙΚΑΣΙΑ ΜΕΤΡΗΣΗΣ ΤΗΣ ΘΕΡΜΟΚΡΑΣΙΑΣ

Η λειτουργικότητα του πυρήνα του **DS18B20** είναι **Direct to Digital Temperature Sensor** και η ανάλυση της θερμοκρασίας είναι ρυθμιζόμενη σε **(9, 10, 11 ή 12 bits)** τα οποία είναι ίσα με υποδιαιρέσεις θερμοκρασίας **0.5 ºC, 0.25 ºC, 0.125 ºC, ή 0.0625 ºC** αντίστοιχα. Τα αισθητήρια είναι προεπιλεγμένα στην ανάλυση των **12 bits**. Όταν εκτελέσουμε την εντολή **Convert T [44h]** γίνεται η διαδικασία της μετατροπής της θερμοκρασίας και τα δεδομένα

αποθηκεύονται στην μνήμη **Scratchpad**. Η πληροφορία της θερμοκρασίας μεταφέρετε μέσω της **1-Wire** γραμμής, από το αισθητήριο στον **Master**, με την εντολή **Read Scratchpad [BEh]** και ξεκινάει από το **LSB**.

Στον **πίνακα 3.1** φαίνεται η σχέση θερμοκρασίας και ανάλυσης δεδομένων. Τα **S** χρησιμοποιούνται για το πρόσημο της θερμοκρασίας. Ενώ τα **bit** τα οποία έχουν αρνητικό εκθέτη, αντιστοιχούν στις υποδιαιρέσεις της θερμοκρασίας από **9 - 12 bits.**

Π.χ. Εάν θέλουμε ανάλυση θερμοκρασίας **0.5 ºC** τότε το **9 bit** θα περιέχει ‘’**1**’’ και τα **10, 11 και 12** θα περιέχουν ‘’**0**’’. Καθώς οι θερμοκρασίες υποδεικνύονται σε κλίμακα **Celsius** η μετατροπή σε **Fahrenheit** θα γίνετε: είτε με την χρήση ρουτίνας, είτε με πίνακα αντιστοίχησης.

πίνακας 3.1: Σχέση θερμοκρασίας και ανάλυσης δεδομένων.

Temperature/Data Relationships

2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	LSB
MSb				(unit = °C)			LSb	
S	S	S	S	S	2^6	2^5	2^4	MSB

TEMPERATURE	DIGITAL OUTPUT (Binary)	DIGITAL OUTPUT (Hex)
+125°C	0000 0111 1101 0000	07D0h
+25.0625°C	0000 0001 1001 0001	0191h
+10.125°C	0000 0000 1010 0010	00A2h
+0.5°C	0000 0000 0000 1000	0008h
0°C	0000 0000 0000 0000	0000h
-0.5°C	1111 1111 1111 1000	FFF8h
-10.125°C	1111 1111 0101 1110	FF5Eh
-25.0625°C	1111 1110 0110 1111	FF6Fh
-55°C	1111 1100 1001 0000	FC90h

ΚΑΤΑΧΩΡΗΤΕΣ ΜΕΓΙΣΤΗΣ ΚΑΙ ΕΛΑΧΙΣΤΗΣ ΘΕΡΜΟΚΡΑΣΙΑΣ

Οι καταχωρητές αυτοί αποθηκεύονται σε μια **EEPROM** των **2 byte**. Οι σκανδαλιστές συναγερμού θερμοκρασίας **TH** και **TL** αποτελούνται από **1 Byte EEPROM** ο κάθε ένας. Εάν η εντολή αναζήτησης συναγερμών δεν εφαρμόζεται στο **DS18B20**, αυτοί οι καταχωρητές μπορούν να χρησιμοποιηθούν ως μνήμη γενικού σκοπού από το χρήστη.

ΛΕΙΤΟΥΡΓΙΑ – ΣΗΜΑΤΟΔΟΤΗΣΗΣ ΤΩΝ ΣΥΝΑΓΕΡΜΩΝ

Αφού το **DS18B20** εκτελέσει μία μετατροπή θερμοκρασίας, η τιμή της μετρηθήσας θερμοκρασίας συγκρίνεται με τις τιμές των καταχωρητών **TH** και **TL**. Οι αμετάβλητοι αυτοί καταχωρητές είναι **8-bit**, έτσι οι υποδιαιρέσεις της θερμοκρασίας που αντιστοιχούν στα **9-12 bit** αγνοούνται στην σύγκριση. Το **MSb** των καταχωρητών **TH** και **TL** αντιστοιχεί στο πρόσημο του **16-bit** καταχωρητή θερμοκρασίας. Εάν το αποτέλεσμα της μετρηθήσας θερμοκρασίας είναι μεγαλύτερο από τον **TH** ή μικρότερο από τον **TL**, τότε ένα **Flag** συναγερμού ενεργοποιείτε, όσο το **Flag** είναι ενεργοποιημένο το **DS18B20** θα ανταποκριθεί στην εντολή **Alarm Search**, το **Flag** ανανεώνεται σε κάθε καινούργια μέτρηση της θερμοκρασίας.

Κάθε φορά που ανιχνεύουμε συναγερμούς και κάποια ή κάποιες συσκευές υπερβούν τις τιμές που έχουμε ορίσει, τότε μπορούμε να τις διευκρινίσουμε και να τις απομονώσουμε διαβάζοντας μόνο αυτές και όχι όλες τις συσκευές.

Ο ΚΑΤΑΧΩΡΗΤΗΣ ΡΥΘΜΙΣΕΩΝ

Η **Scratchpad** περιέχει επίσης ένα **Byte** διαμόρφωσης για να θέτει την επιθυμητή ανάλυση της θερμοκρασίας στην ψηφιακή μετατροπή. Το γράψιμο των **TH, TL** και του **Byte** διαμόρφωσης γίνεται χρησιμοποιώντας μια εντολή λειτουργίας μνήμης **(Write Scratchpad)**. Η πρόσβαση για ανάγνωση αυτών των καταχωρητών γίνεται μέσω του **Scratchpad**. Όλα τα δεδομένα διαβάζονται και γράφονται με το λιγότερο σημαντικό **bit** να αποστέλλεται πρώτο. Ο καταχωρητής αυτός καθορίζει την ανάλυση που θα έχει η μετατροπή της θερμοκρασίας σε ψηφιακή μορφή. Στο **σχήμα 3.5**, έχουμε την εσωτερική δομή του καταχωρητή, τα **bits** από **0-4** είναι πάντα ‘’**1**’’. Το **TM** είναι το **Test Mode bit** και χρησιμοποιείται για να θέσει το **DS18B20**, είτε σε κατάσταση λειτουργίας είτε σε κατάσταση **Test**. Η προεπιλεγμένη τιμή του θα είναι ‘’**0**’’ το οποίο σημαίνει κατάσταση λειτουργίας και δεν θα πρέπει να αλλάξει. Τα **R0** και **R1** είναι τα **bits** όπου ορίζουν την ανάλυση της θερμοκρασίας. Στον **πίνακα 3.2** διευκρινίζεται η ανάλυση της ψηφιακής θερμοκρασίας με βάση τις αλλαγές των δύο **bits**. Υπάρχει άμεση σχέση

μεταξύ ανάλυσης και χρόνου μετατροπής όπως φαίνεται και από τα ηλεκτρικά χαρακτηριστικά. Η προκαθορισμένη ανάλυση του εργοστασίου είναι τα **12 bits**, δηλαδή για **R0=1** και **R1=1**

DS18B20 CONFIGURATION REGISTER

TM	R1	R0	1	1	1	1	1

MSb LSb

σχήμα 3.5: Εσωτερική δομή του καταχωρητή ρυθμίσεων.

πίνακας 3.2: ανάλυση της ψηφιακής θερμοκρασίας

Thermometer Resolution Configuration

R1	R0	Thermometer Resolution	Max Conversion Time
0	0	9-bit	93.75ms
0	1	10-bit	187.5ms
1	0	11-bit	375ms
1	1	12-bit	750ms

CRC (Cyclic Redundancy Check)

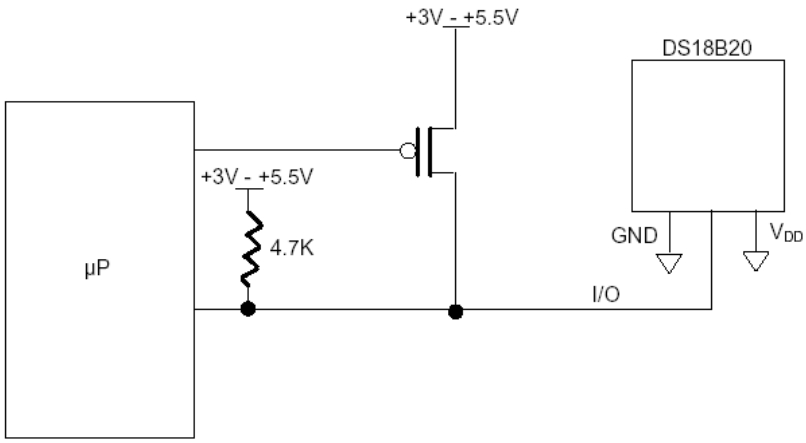
Κυκλικός Έλεγχος Πλεονασμού 8 bit, είναι ο ίδιος μηχανισμός που περιγράφεται στο **κεφάλαιο 3**.

ΠΑΡΑΣΙΤΙΚΗ ΤΡΟΦΟΔΟΤΗΣΗ

Το αισθητήριο αντλεί ενέργεια από το **1-Wire bus**, μέσω της «**παρασιτικής**» τροφοδοσίας. Κατά την διάρκεια που η **1-Wire** γραμμή είναι σε υψηλό επίπεδο δυναμικού το αισθητήριο αποθηκεύει ενέργεια σε έναν εσωτερικό πυκνωτή, ο οποίος γίνεται η πηγή τροφοδοσίας του κατά τις χρονικές περιόδους που η γραμμή είναι σε χαμηλό δυναμικό επίπεδο, κάθε φορά που η **1-Wire** γραμμή εναλλάσσετε από χαμηλό σε υψηλό επίπεδο ανανεώνεται η «**παρασιτική**» ενέργεια στον εσωτερικό πυκνωτή. Σαν εναλλακτική λύση, το

DS18B20 μπορεί επίσης να τροφοδοτηθεί από εξωτερική πηγή **3V** - **5.5V**. Στο **σχήμα 3.6** φαίνεται το σχεδιάγραμμα της παρασιτικής τροφοδότησης.

Η χρήση της παρασιτικής τροφοδότησης δεν συνίσταται για θερμοκρασίες πάνω από **100^{o}C**, δεδομένου ότι μπορεί να μην είναι σε θέση να στηρίξει της επικοινωνίες, λόγω των υψηλότερων ρευμάτων διαρροής που παρουσιάζουν τα αισθητήρια σε αυτές της θερμοκρασίες. Για εφαρμογές της οποίες τέτοιες θερμοκρασίες είναι πιθανές, εφαρμόζεται εξωτερική τροφοδότηση της επαφής **VDD** του αισθητηρίου.



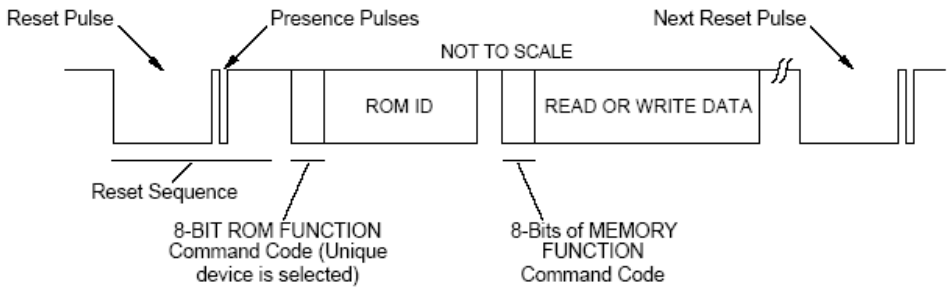
σχήμα 3.6: Σχεδιάγραμμα της παρασιτικής τροφοδότησης.

ΑΚΟΛΟΥΘΙΑ ΣΥΝΑΛΛΑΓΗΣ

Το πρωτόκολλο που χρησιμοποιούμε για να επικοινωνήσουμε με το **DS18B20** μέσω του **1-Wire bus** έχει ως εξής:

- **Αρχικοποίηση**
- **Εντολές λειτουργίας της ROM**
- **Εντολές λειτουργίας της μνήμης**
- **Ανταλλαγή δεδομένων**

Ένας πλήρης κύκλος της ακολουθίας συναλλαγής φαίνεται στο **σχήμα 3.7**.



σχήμα 3.7: Πλήρης κύκλος ακολουθίας συναλλαγής.

ΑΡΧΙΚΟΠΟΙΗΣΗ

Όλες οι συναλλαγές στο **1-Wire** ξεκινάνε με μία ακολουθία αρχικοποίησης. Η ακολουθία αρχικοποίησης αποτελείτε από έναν παλμό **Reset** ο οποίος μεταδίδεται από τον **Master** στον δίαυλο και στην συνέχεια μεταδίδονται παλμοί παρουσίας από τους **Slaves**. Οι παλμοί παρουσίας ενημερώνουν τον **Master** ότι τα **DS18B20** βρίσκονται στον δίαυλο και είναι έτοιμα για λειτουργία.

ΕΝΤΟΛΕΣ ΛΕΙΤΟΥΡΓΙΑΣ ROM ΤΟΥ DS18B20

Όταν ο **Master** ανιχνεύσει παρουσία στον δίαυλο δεδομένων, μπορεί να στείλει μία από τις **5** εντολές λειτουργίας της **ROM** για της οποίες ακολουθεί μία λίστα και όλες οι εντολές αποτελούνται από **8-bit**. (Αναφορά στο διάγραμμα ροής του **σχήματος 3.8**). Το **Command Set** του **DS18B20** αποτελείται από τις **4** βασικές εντολές **ROM** και την **Alarm Search**.

Read ROM [33h]

Η εντολή αυτή δίνει την δυνατότητα στον **Master** να διαβάσει τον αύξοντα αριθμό **(πυριτίου)** της **ROM** του αισθητηρίου. Η εντολή αυτή μπορεί να χρησιμοποιηθεί εάν στον δίαυλο βρίσκεται μόνο ένα **DS18B20** στην περίπτωση που βρίσκονται παραπάνω από ένα αισθητήρια στην γραμμή, θα παρουσιαστή πρόβλημα, διότι θα απαντήσουν όλα την ίδια χρονική στιγμή.
(Η συνδεσμολογία Open **Drain** θα προκαλέσει αποτέλεσμα πύλης **AND**).

Match ROM [55h]

Μετά την έκδοση της εντολής **Match ROM**, ο ελεγκτής διαύλου στέλνει και την ακολουθία των **64-bit** της **ROM**, όπου τον καθιστά ικανό να διευθετήσει και να απομονώσει ένα συγκεκριμένο αισθητήριο σε δίαυλο όπου υπάρχουν πολλά αισθητήρια και μόνο το αισθητήριο με τον ίδιο **64-bit ROM** θα ανταποκριθεί στις εντολές που θα ακολουθήσουν, τα υπόλοιπα αισθητήρια στον δίαυλο θα περιμένουν για ένα παλμό **Reset**. Η εντολή αυτή μπορεί να εκδοθεί σε δίαυλο όπου ένα ή περισσότερα αισθητήρια υπάρχουν.

Skip ROM [CCh]

Η εντολή αυτή χρησιμοποιείται για να επιτρέψει την άμεση πρόσβαση του ελεγκτή στις εντολές λειτουργίας της μνήμης ενός και μόνο αισθητηρίου, χωρίς να χρειαστεί η έκδοση του **64-bit** κωδικού. Στην περίπτωση που στον δίαυλο υπάρχουν περισσότερα αισθητήρια και ο ελεγκτής εκδώσει μία εντολή **Read** μετά την εντολή **Skip ROM** θα δημιουργηθεί σύγχυση δεδομένων καθώς όλα τα αισθητήρια θα ανταποκριθούν ταυτόχρονα.

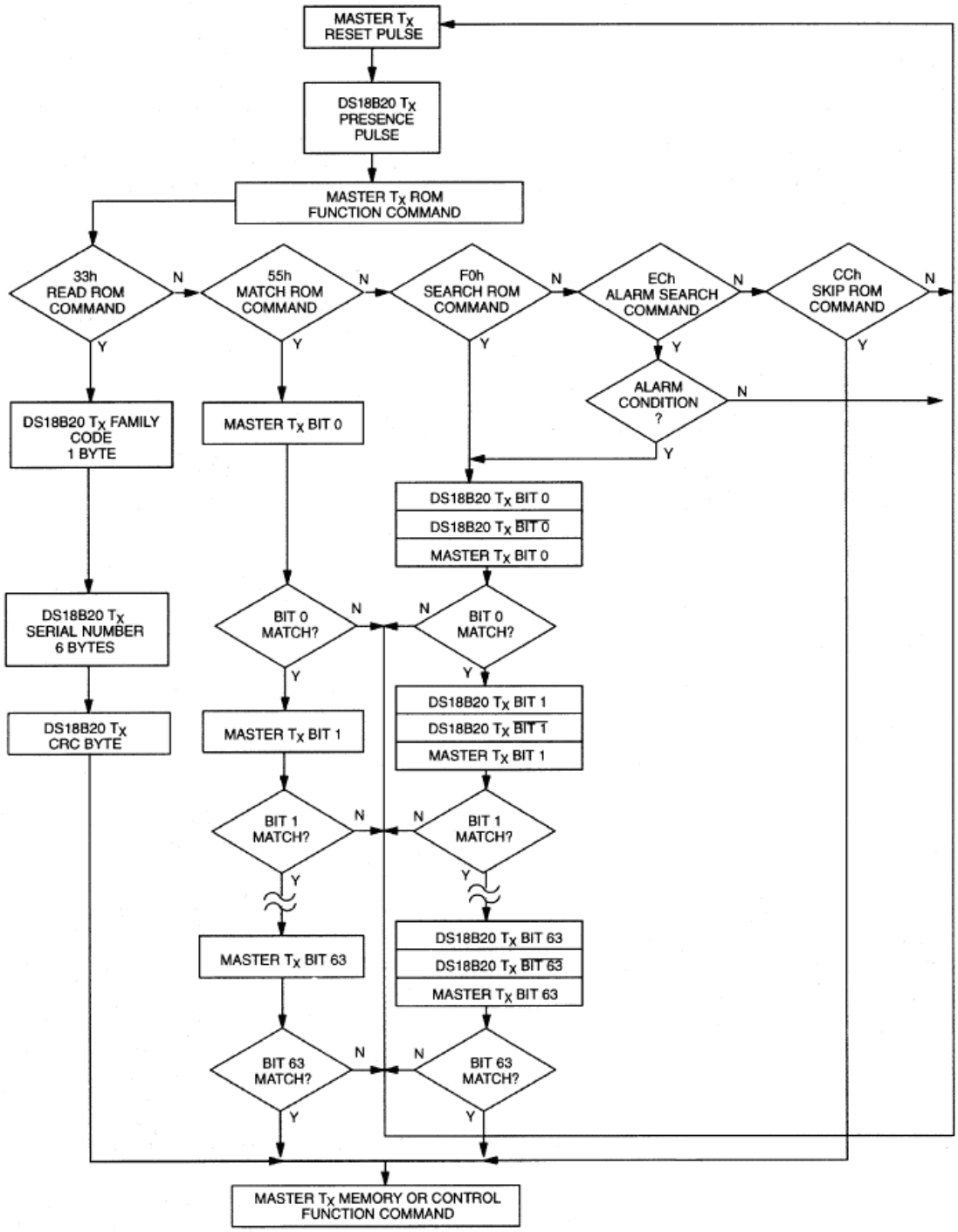
Search ROM [F0h]

Όταν ένα σύστημα αρχικοποιηθεί, ο ελεγκτής διαύλου είναι πιθανό να μην γνωρίζει τον ακριβή αριθμό των αισθητηρίων που παρουσιάζονται στον **1-Wire** δίαυλο ή τους **64-bit ROM** κωδικούς τους. Η **Search ROM** εντολή δίνει την δυνατότητα στον ελεγκτή διαύλου να χρησιμοποιήσει μία διαδικασία αποβολής εξάλειψης απαλοιφής όπου θα διευκρινίσει τους **64-bit ROM** κωδικούς όλων των αισθητηρίων.

Alarm Search [ECh]

Το διάγραμμα ροής της εντολής αυτής είναι ίδιο με της **Search ROM** εντολής. Ωστόσο τα αισθητήρια θα ανταποκριθούν στην εντολή αυτή μόνο εάν μία κατάσταση συναγερμού παρουσιαστεί από την τελευταία μέτρηση της θερμοκρασίας. Ως κατάσταση συναγερμού ορίζουμε την θερμοκρασία η οποία είναι μεγαλύτερη από την τιμή του **TH** και μικρότερη από του **TL**. Η κατάσταση συναγερμού παραμένει ενεργεί όσο το αισθητήριο τροφοδοτείτε ή εωσότου μία καινούργια μέτρηση την αναιρέσει.

ROM FUNCTIONS FLOW CHART



σχήμα 3.8: Εντολές λειτουργίας της ROM

ΕΝΤΟΛΕΣ ΛΕΙΤΟΥΡΓΙΑΣ ΤΗΣ ΜΝΗΜΗΣ

Τα ακόλουθα πρωτόκολλα εντολών συνοψίζονται στον **πίνακα 3.3** και στο διάγραμμα ροής του **σχήματος 3.9**.

Write Scratchpad [4Eh]

Με την εντολή αυτή γράφουμε στο **Scratchpad** του αισθητηρίου, ξεκινώντας από τον καταχωρητή **TH**, τα επόμενα **3 Byte** που θα γραφτούν θα σωθούν στις θέσεις των διευθύνσεων από την **2** μέχρι την **4**. Και τα **3 Byte** πρέπει να γραφτούν πριν από την έκδοση **Reset**.

Read Scratchpad [BEh]

Με την εντολή αυτή διαβάζουμε τα περιεχόμενα του **Scratchpad**, ξεκινώντας από το πρώτο **(Byte 0)** μέχρι το τελευταίο **(Byte 8, CRC)**. Εάν δεν θέλουμε να διαβάσουμε όλα τα δεδομένα του **Scratchpad** η διαδικασία μπορεί να τερματιστεί με την έκδοση ενός παλμού **Reset** από τον ελεγκτή.

Copy Scratchpad [48h]

Η εντολή αυτή αντιγράφει το **Scratchpad** στην μνήμη $\mathbf{E^2}$ του αισθητηρίου αποθηκεύοντας έτσι τους σκανδαλιστές θερμοκρασίας στην αμετάβλητη μνήμη. Εάν ο ελεγκτής κάνει ανάγνωση της γραμμής ενώ η διαδικασία της αντιγραφής στην μνήμη $\mathbf{E^2}$ είναι σε εξέλιξη, το αισθητήριο θα επιστρέψει ‘’**0**’’ καθ’ όλη την διάρκεια της αντιγραφής, μόλις η αντιγραφή τελειώσει θα επιστρέψει ‘’**1**’’. Στην περίπτωση που χρησιμοποιούμε παρασιτική τροφοδότηση, ο ελεγκτής θα πρέπει να εφαρμόσει ισχυρό **Pull-Up** για τουλάχιστον **10ms** αμέσως μετά την έκδοση της εντολής αυτής.

Convert T [44h]

Με την έκδοση της εντολής **Convert T** αρχίζει η μετατροπή της θερμοκρασίας και όταν ολοκληρωθεί, το αισθητήριο θα παραμείνει ανενεργό. Στην περίπτωση που ο ελεγκτής ζητήσει ανάγνωση, το αισθητήριο θα απαντήσει με ‘’**0**’’ εωσότου ολοκληρώσει την μετατροπή, όπου και θα επιστρέψει ‘’**1**’’. Εάν η συνδεσμολογία της τροφοδότησης είναι παρασιτική, η γραμμή θα πρέπει να ανυψωθεί σε ισχυρό **Pull-Up** αμέσως μετά την έκδοση της εντολής για **500ms**.

Recall E2 [B8h]

Η εντολή αυτή επαναφέρει τις τιμές των σκανδαλιστών θερμοκρασίας και του καταχωρητή ρυθμίσεων, από την μνήμη **E^2** στο **Scratchpad**. Και το **Scratchpad** θα έχει διαθέσιμα έγκυρα δεδομένα, μόλις το αισθητήριο τροφοδοτηθεί, η διαδικασία της επαναφοράς γίνεται κάθε φορά αυτόματα, κατά την εκκίνηση του αισθητηρίου. Το αισθητήριο ανταποκρίνεται με ‘’**0**’’ όταν βρίσκεται σε εξέλιξη η διαδικασία και με ‘’**1**’’ όταν είναι έτοιμο.

Read Power Supply [B4h]

Με κάθε αίτημα του ελεγκτή για ανάγνωση μετά την έκδοση της εντολής **Read Power Supply**, το αισθητήριο διευκρινίζει τον τρόπο με τον οποίο τροφοδοτείτε, επιστρέφοντας ‘’**0**’’ για παρασιτική τροφοδότηση και ‘’**1**’’ για εξωτερική τροφοδότηση.

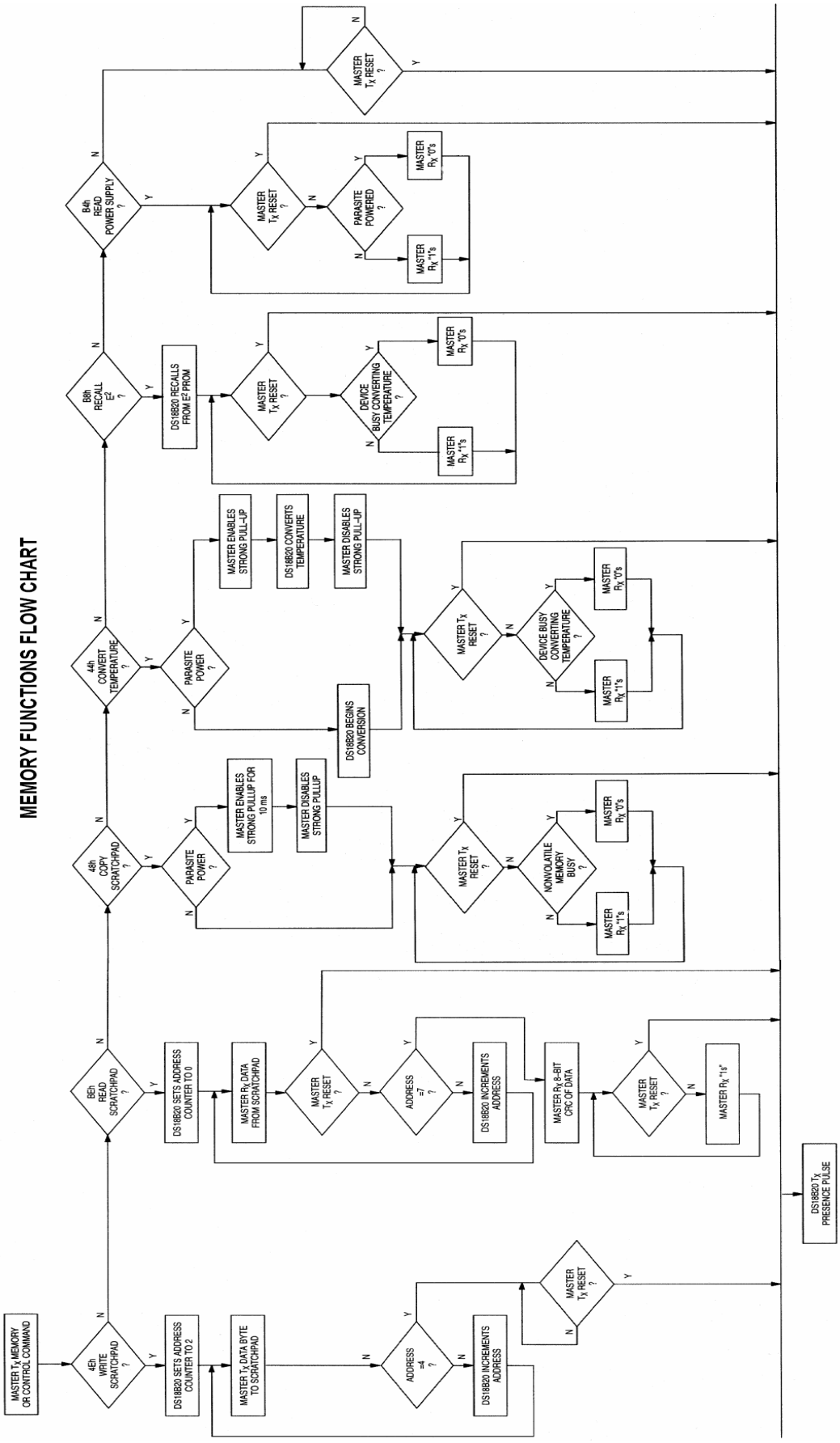
πίνακας 3.3: Set εντολών του DS18B20.

DS18B20 COMMAND SET

INSTRUCTION	DESCRIPTION	PROTOCOL	1-WIRE BUS AFTER ISSUING PROTOCOL	NOTES
TEMPERATURE CONVERSION COMMANDS				
Convert T	Initiates temperature conversion.	44h	<read temperature busy status>	1
MEMORY COMMANDS				
Read Scratchpad	Reads bytes from scratchpad and reads CRC byte.	BEh	<read data up to 9 bytes>	
Write Scratchpad	Writes bytes into scratchpad at addresses 2 through 4 (TH and TL temperature triggers and config).	4Eh	<write data into 3 bytes at addr. 2 through. 4>	3
Copy Scratchpad	Copies scratchpad into nonvolatile memory (addresses 2 through 4 only).	48h	<read copy status>	2
Recall E^2	Recalls values stored in nonvolatile memory into scratchpad (temperature triggers).	B8h	<read temperature busy status>	
Read Power Supply	Signals the mode of DS18B20 power supply to the master.	B4h	<read supply status>	

ΣΗΜΕΙΩΣΕΙΣ ΓΙΑ ΤΟΝ ΠΙΝΑΚΑ 3.3

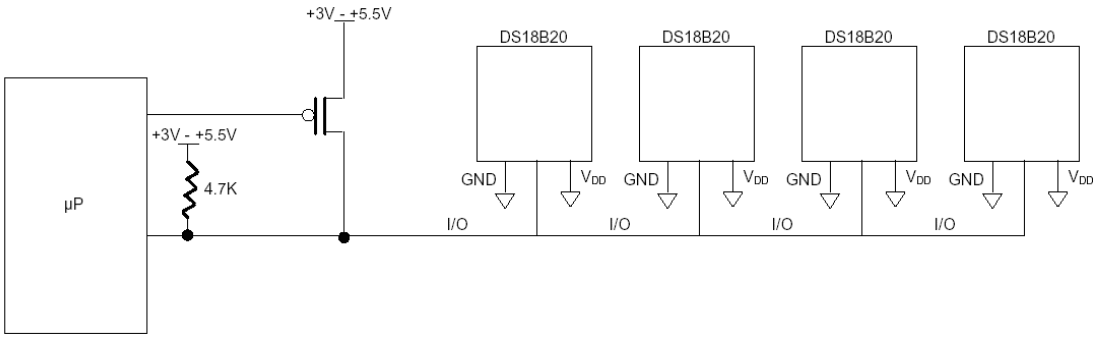
1. Η μετατροπή της θερμοκρασίας χρειάζεται χρόνο μέχρι **500 ms**, για αυτόν τον λόγω όταν το αισθητήριο λάβει την εντολή **Convert T** και δεν τροφοδοτείται εξωτερικά, η **I/O** γραμμή πρέπει να παραμείνει σε υψηλό δυναμικό κατά την διάρκεια της μετατροπής, για τουλάχιστον **500 ms**. Όταν η διαδικασία είναι σε εξέλιξη, καμία άλλη δραστηριότητα δεν λαμβάνει χώρα στον δίαυλο **1-Wire**.
2. Μετά την απολαβή της εντολής **Scratchpad** και εφόσον το αισθητήριο δεν τροφοδοτείτε εξωτερικά, ο δίαυλος παραμένει σε υψηλό δυναμικό για τουλάχιστον **10 ms**, έτσι ώστε να παρέχει στο αισθητήριο την απαιτούμενη ισχύ που χρειάζεται για την εκτέλεση της διαδικασίας. Όταν η διαδικασία είναι σε εξέλιξη, καμία άλλη δραστηριότητα δεν λαμβάνει χώρα στον δίαυλο **1-Wire**.
3. Και τα τρία **Bytes** πρέπει να γραφτούν πριν από την έκδοση ενός παλμού **Reset**.



σχήμα 3.9: Διάγραμμα ροής λειτουργίας της μνήμης.

ΣΥΣΤΗΜΑ ΣΥΛΛΟΓΗΣ ΔΕΔΟΜΕΝΩΝ ΜΕ ΤΟ DS18B20

Στην παρούσα εφαρμογή θα περιγράψουμε ένα σύστημα συλλογής δεδομένων χρησιμοποιώντας το **1-Wire** πρωτόκολλο στα αισθητήρια θερμοκρασίας **DS18B20.** Η κυκλωματική διάταξη που θα χρησιμοποιήσουμε, στηρίζεται στην παρασιτική τροφοδότηση των αισθητηρίων. Το σύστημα αποτελείται από έναν μικροεπεξεργαστή που θα έχει ρόλο ελεγκτή διαύλου **(Master)** και από **4** απομακρυσμένα αισθητήρια θερμοκρασίας **(Slaves)**, από μία διάταξη παθητικού **Pull-Up** με αντιστάτη στα **4k7** και μία διάταξη ενεργού **Pull-Up** με **MOSFET** όπως φαίνεται στο σχήμα **3.10.**



σχήμα 3.10: Σύστημα συλλογής δεδομένων χρησιμοποιώντας το 1-Wire πρωτόκολλο και το αισθητήριο DS18B20.

Την διάταξη του παθητικού **Pull-Up** την χρησιμοποιούμε λόγω του ότι τα **Port** του μικροεπεξεργαστή είναι συνδεσμολογίας ανοιχτού συλλέκτη και δεν παρέχουν αρκετό ρεύμα ώστε να τροφοδοτήσουν τα αισθητήρια.

Την διάταξη του ενεργού **Pull-Up** την χρησιμοποιούμε για να παρέχουμε στα αισθητήρια αρκετή ισχύ κατά τις μετατροπές θερμοκρασίας και τις αντιγραφές του **Scratchpad** στην $\mathbf{E^2}$.

ΤΕΧΝΙΚΑ ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΤΟΥ ΣΥΣΤΗΜΑΤΟΣ

Σε αυτό το σημείο θα πρέπει να προσδιορίσουμε τον τρόπο με τον οποίο θα λειτουργεί το σύστημα μας, τον τρόπο τον καθορίζουν οι απαιτήσεις τις εκάστοτε εφαρμογής. Στην συγκεκριμένη εφαρμογή θα χρησιμοποιήσουμε ανάλυση θερμοκρασίας **9 bit** δηλαδή βήμα θερμοκρασίας **0.5 $\mathbf{C^o}$**. Οι

καταχωρητές συναγερμών **TH** και **TL** δεν θα χρησιμοποιηθούν οπότε μπορούμε να τους χρησιμοποιήσουμε σαν καταχωρητές γενικού σκοπού. Η δειγματοληψία θα γίνεται κάθε ένα λεπτό λόγω του ότι η θερμοκρασία σαν φυσικό μέγεθος μεταβάλετε σχετικά αργά.

ΣΥΝΤΟΜΗ ΑΝΑΛΥΣΗ ΠΡΩΤΟΚΟΛΛΟΥ

Η επικοινωνία με το **DS18B20** γίνεται δια μέσου του ***1-Wire* port**, για να επιτευχθεί όμως κάτι τέτοιο και να ξεκινήσει η επικοινωνία μεταξύ **Master** και αισθητηρίου, θα πρέπει να επαληθευτεί το πρωτόκολλο λειτουργίας της **ROM** ειδάλλως η εντολές ελέγχου και μνήμης του αισθητηρίου δεν θα είναι διαθέσιμες. Θα πρέπει δηλαδή ο **Master** να επικοινωνήσει με το αισθητήριο στέλνοντας του μία από τις πέντε εντολές λειτουργίας της μνήμης **ROM**, οι οποίες είναι:

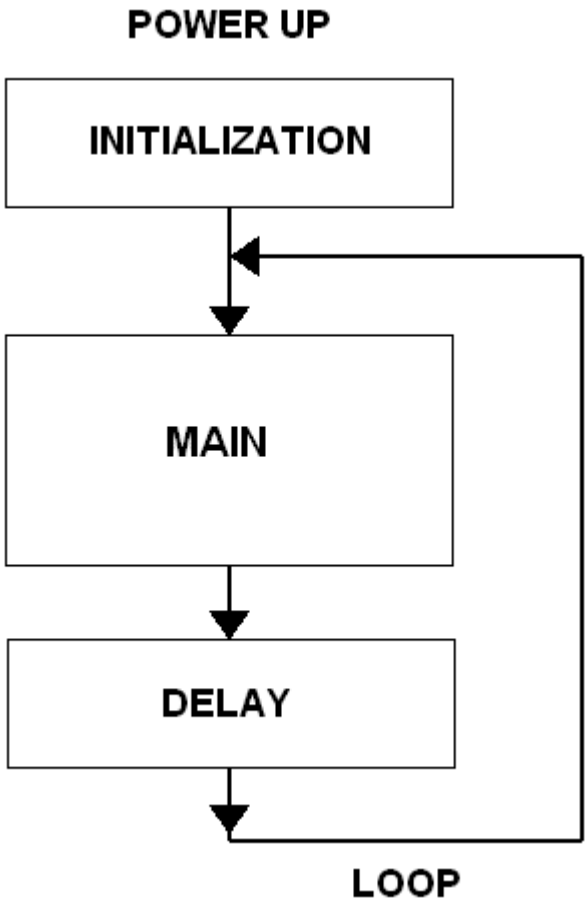
1. **Read ROM**
2. **Match ROM**
3. **Search ROM**
4. **Skip ROM**
5. **Alarm Search**

Οι εντολές αυτές λειτουργούν σε τμήμα της **64-bit ROM** μνήμης κάθε συσκευής και μπορούν να απομονώσουν μία συσκευή, εάν υπάρχουν πολλές στην **1-Wire** γραμμή, όπως επίσης και να διευκρινίσουν στον **Master**, πόσες και τι είδους συσκευές υπάρχουν στον δίαυλο επικοινωνίας.

Αφού εκτελεστεί με επιτυχία η ακολουθία λειτουργίας της μνήμης **ROM**, η μνήμη του αισθητηρίου και οι λειτουργίες ελέγχου του, γίνονται προσβάσιμες από τον **Master**, ο οποίος μπορεί να εκτελέσει οποιαδήποτε από τις έξι εντολές λειτουργίας μνήμης και ελέγχου του αισθητηρίου. Μία εντολή λειτουργίας ελέγχου καθοδηγεί το **DS18B20** για να εκτελέσει μέτρηση θερμοκρασίας. Το αποτέλεσμα αυτής της μέτρησης θα τοποθετηθεί στη μνήμη **Scratchpad** του **DS18B20** και μπορεί να διαβαστεί με την έκδοση μιας εντολής λειτουργίας μνήμης που διαβάζει το περιεχόμενο της **Scratchpad**.

ΑΝΑΠΤΥΣΣΟΝΤΑΣ ΤΟ ΔΙΑΓΡΑΜΜΑ ΡΟΗΣ

Για την δημιουργία του διαγράμματος ροής χωρίζουμε την λειτουργία της εφαρμογής σε τρία βασικά κομμάτια, όπως φαίνονται στο **σχήμα 3.11**.



σχήμα 3.11: βασικό διάγραμμα ροής της εφαρμογής.

Το πρώτο κομμάτι **(Initialization)** περιλαμβάνει την αρχικοποίηση του συστήματος που λαμβάνει χώρα κάθε φορά που τροφοδοτείται το σύστημα. Στο κομμάτι αυτό περιλαμβάνονται οι διαδικασίες αρχικοποίησης του διαύλου, ταυτοποίησης των αισθητηρίων που βρίσκονται συνδεδεμένα στον δίαυλο, παραμετροποίηση των καταχωρητών ρυθμίσεων των αισθητηρίων και αντιγραφή του στην $\mathbf{E^2}$ μνήμη του αισθητηρίου.

Στο δεύτερο και κύριο κομμάτι **(Main)**, περιλαμβάνονται οι λειτουργίες, μετατροπής θερμοκρασίας και ανάγνωσης της μνήμης **Scratchpad**.

Το τρίτο κομμάτι αποτελείται από μία ρουτίνα καθυστέρησης μέχρι την επόμενη δειγματοληψία.

ΑΝΑΛΥΣΗ ΑΡΧΙΚΟΠΟΙΗΣΗΣ (INITIALIZATION)

Το κομμάτι της αρχικοποίησης μπορεί να χωριστεί σε τρία επιμέρους κομμάτια, αρχικοποίηση διαύλου, ταυτοποίηση συσκευών και παραμετροποίηση του καταχωρητή ρυθμίσεων στις απαιτήσεις του συστήματος.

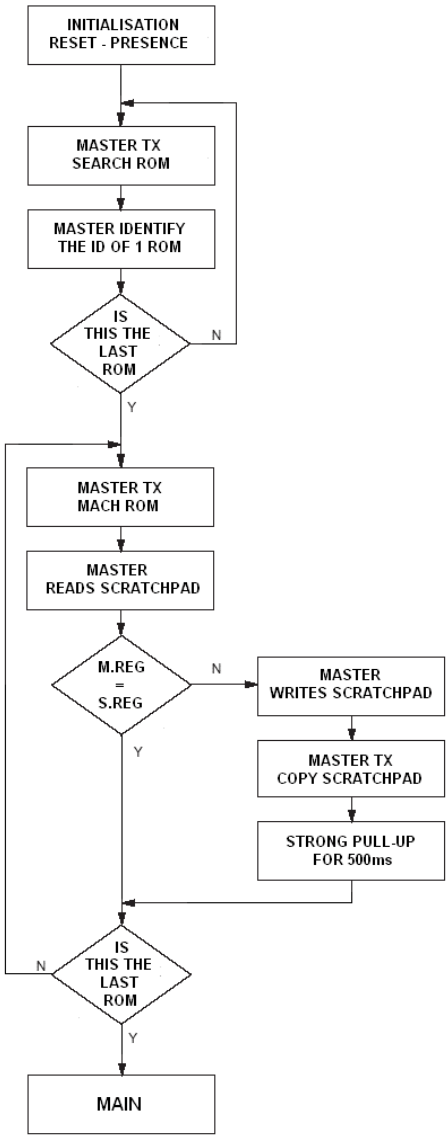
Όπως έχουμε προαναφέρει για να λάβει χώρα οποιαδήποτε επικοινωνία στον δίαυλο θα πρέπει πρώτα να γίνει αρχικοποίηση του αυτό γίνεται με παλμούς **Reset-Presence**.

Μετά την αρχικοποίηση του διαύλου πρέπει να γίνει ταυτοποίηση συσκευών, στην περίπτωσή μας γίνεται με την εντολή **Search ROM**. Λόγω του ότι βρίσκονται περισσότερα από ένα αισθητήρια στον δίαυλο. Ο μηχανισμός ανεύρεσης της **ROM** περιγράφεται αναλυτικά στο **Κεφάλαιο 2**.

Μετά την ταυτοποίηση των αισθητηρίων ακολουθεί η παραμετροποίηση του καταχωρητή ρυθμίσεων του κάθε αισθητηρίου στις ανάγκες του συστήματός μας. Για να έχουμε πρόσβαση στον καταχωρητή θα πρέπει να στείλουμε μια εντολή **Mach ROM** ακολουθούμενη από την ταυτότητα του συγκεκριμένου αισθητηρίου που θέλουμε να μεταβάλουμε. Αφού έχουμε απομονώσει το συγκεκριμένο αισθητήριο, αποστέλλουμε μία εντολή ανάγνωσης **Scratchpad** για να διαβάσουμε τον καταχωρητή ρυθμίσεων **(5^{ο} Byte** του **Scratchpad),** αφού τον διαβάσουμε τον συγκρίνουμε με τις ρυθμίσεις του συστήματός μας που είναι καταχωρημένες στην μνήμη του μικροεπεξεργαστή, αν οι ρυθμίσεις είναι ίδιες τότε προχωράμε στο επόμενο αισθητήριο, αφού λάβει χώρα μια ακολουθία αρχικοποίησης **(Reset-Presence)**. Αν τα περιεχόμενα του καταχωρητή ρυθμίσεων δεν είναι ίδια με αυτά του μικροεπεξεργαστή τότε γράφουμε στον καταχωρητή ρυθμίσεων του **Scratchpad (5^{ο} Byte)** τις ρυθμίσεις του συστήματός μας από τον μικροεπεξεργαστή, αυτό γίνεται μέσω της εντολής **Write Scratchpad**. Μετά την ολοκλήρωση της ρύθμισης του καταχωρητή θα πρέπει να τον αντιγράψουμε στην **E^{2}** μνήμη του αισθητηρίου, έτσι ώστε το αισθητήριο να ρυθμίζεται αυτόματα στις απαιτήσεις του συστήματος μετά από κάθε επανεκκίνηση της συσκευής*. Η αντιγραφή επιτυγχάνεται με την εντολή **Copy Scratchpad**.

* Σε κάθε επανεκκίνηση των αισθητηρίων εκτελείται αυτόματα η εντολή **Recall E^2** η οποία μεταφέρει τα περιεχόμενα της **E^2** στο **Scratchpad**.

Ο τρόπος αυτός μας επιτρέπει την αντικατάσταση οποιουδήποτε αισθητηρίου χωρίς να χρειάζεται να το προ ρυθμίσουμε. Ο μηχανισμός αυτός μας κάνει αυτόματη ρύθμιση μόνο των αισθητηρίων που δεν τηρούν τις προδιαγραφές του συστήματος. Στο **σχήμα 3.12** φαίνεται το διάγραμμα ροής της αρχικοποίησης.



σχήμα 3.12: Διάγραμμα ροής της αρχικοποίησης.

ΚΥΡΙΟΣ ΠΡΟΓΡΑΜΜΑ (MAIN)

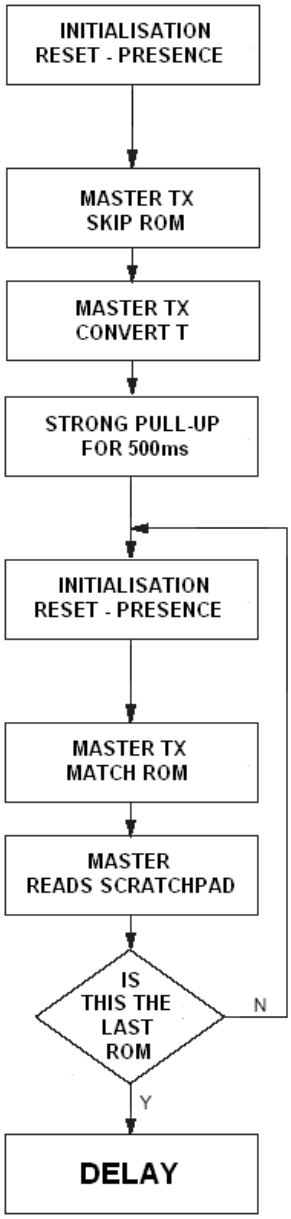
Στην **Main** ακολουθεί η διαδικασία της μετατροπής θερμοκρασίας και ανάγνωσης της από τον μικροεπεξεργαστή. Για να γίνει πιο γρήγορα η διαδικασία αυτή, ο μικροεπεξεργαστής θα στείλει πρώτα την εντολή **Skip ROM,** με την οποία θα έχει πρόσβαση στις λειτουργίες της μνήμης των αισθητηρίων χωρίς να εκδώσει τα **64 bit ROM**. Έπειτα τους ζητάει ταυτόχρονη μετατροπή της θερμοκρασίας με την εντολή **Convert T**.

Επειδή όμως έχουμε παρασιτική τροφοδότηση, εφαρμόζουμε ισχυρό **Pull-Up** για **500ms** και ο μικροεπεξεργαστής περιμένει για τον χρόνο αυτό, έτσι ώστε να δώσει το απαιτούμενο χρονικό περιθώριο και ενέργεια στα αισθητήρια για τις μετατροπές θερμοκρασίας. Τα αισθητήρια καθ' όλη την διάρκεια της μετατροπής, εκδίδουν "**0**" στο **Busy Flag** τους και έτσι απορρίπτεται οποιοδήποτε αίτημα του μικροεπεξεργαστή. Μόλις η διαδικασία της μετατροπής τελειώσει το **Busy Flag** θα γίνει "**1**".

Τότε ο μικροεπεξεργαστής θα αντιγράψει τα αποτελέσματα των θερμοκρασιών με τον εξής τρόπο. Πρώτα θα γίνει η ταυτοποίηση του αισθητηρίου, με την εντολή **Match ROM**, όπου μετά την έκδοση της ακολουθούν τα **64 bit ROM** διευκρινίζοντας με ποίο αισθητήριο θέλει να μιλήσει. Τα υπόλοιπα αισθητήρια μπαίνουν σε αδράνεια και περιμένουν για την έκδοση ενός παλμού **Reset**. Στην συνέχεια διαβάζει το **Scratchpad** του αισθητηρίου με την εντολή **Read Scratchpad**.

Η πληροφορία της θερμοκρασίας βρίσκεται στα δύο πρώτα **Byte** του **Scratchpad,** οπότε εάν δεν θέλουμε να διαβάσουμε και τα **9 Byte,** μπορεί κάλλιστα, μετά την αντιγραφή και του δεύτερου **Byte** ο μικροεπεξεργαστής, να εκδώσει **Reset**.

Μετά την έκδοση **Reset** τα αισθητήρια επανέρχονται από την αδράνεια που τα είχαμε θέσει και ο μικροεπεξεργαστής επαναλαμβάνει την διαδικασία τόσες φορές όσα είναι τα αισθητήρια στον δίαυλό μας. Το διάγραμμα ροής της **Main** φαίνεται στο **σχήμα 3.13**.



σχήμα 3.13: Διάγραμμα ροής της Main

ΧΡΟΝΟΚΑΘΥΣΤΕΡΗΣΗ (DELAY)

Η χρονοκαθυστέρηση του συστήματος, καθορίζεται από τις απαιτήσεις μας. Για παράδειγμα στο σύστημα πού αναλύουμε έχουμε ορίσει ότι ο χρόνος δειγματοληψίας της θερμοκρασίας θα είναι ένα λεπτό, για να είμαστε ακριβής στον χρόνο αυτό, θα πρέπει να υπολογίσουμε πρώτα των χρόνο που θα χρειαστούν οι εντολές της **Main** για να εκτελεστούν και να τον αφαιρέσουμε

από τον συνολικό χρόνο που μεσολαβεί σε κάθε δειγματοληψία. Το υπόλοιπο της αφαίρεσης είναι ο χρόνος που πρέπει να δημιουργήσουμε στο **Delay**.

ΥΠΟΛΟΓΙΣΜΟΣ ΤΟΥ DELAY

Για τον υπολογισμό της χρονικής διάρκειας της **Main**, θα προσθέσουμε τους επιμέρους χρόνους της κάθε διαδικασίας όπως περιγράφεται στο διάγραμμα ροής του σχήματος **3.13**.

Η χρονική διάρκεια της αρχικοποίησης όπως ορίζεται από το πρωτόκολλο είναι **960μs** (**Reset = 480μs, Presence = 480μs)**.

Initialization = *2 * 480 = 960μs*

Για την έκδοση οποιασδήποτε εντολής από τον **Master** απαιτείται χρόνος ίσος με το μέγεθος της εντολής, κάθε εντολή αποτελείται από **8-bit**. Η χρονοθυρίδα εγγραφής όπως ορίζεται από το πρωτόκολλο είναι **61μs**. Άρα ο χρόνος που απαιτείται για την έκδοση μίας εντολής είναι **488μs**.

Master Tx = *8 * 61μs = 488μs*

Για την μετατροπή της θερμοκρασίας όπως έχουμε προαναφέρει απαιτείται ισχυρό **Pull-Up** από τον **Master** για **500.000μs**.

Strong Pull-Up = *500.000μs*

Η εντολή **Match ROM** έχει την ιδιαιτερότητα ότι μετά την έκδοση της ακολουθούν τα **64 bit** της **ROM** για την ταυτοποίηση, άρα ο χρόνος που απαιτείται για την εκτέλεση της **Match ROM** θα είναι **4.392μs (t_{BYTE}= 8*61μs)**.

Master Tx Match ROM = *Master Tx + 8 * t_{BYTE} = 4.392μs*

Η εντολή **Read Scratchpad** μας επιτρέπει να διαβάσουμε και τα **9 Byte** της μνήμης, στην περίπτωσή μας θα διαβάζουμε μόνο τα δύο πρώτα **Byte** που

περιέχουν το αποτέλεσμα της θερμοκρασίας. Άρα ο χρόνος που απαιτείται για την **Read Scratchpad** θα είναι ίσος με τον χρόνο που απαιτείται για την έκδοση της εντολής συν τον χρόνο για την ανάγνωση των δύο **Byte**.

Master Tx Read Scratchpad = $Master\ Tx + 2 * t_{BYTE} = 1.464\mu s$

Ο χρόνος ανάγνωσης της θερμοκρασίας της αισθητηρίου θα είναι ίσος με τον χρόνο αρχικοποίησης συν τον χρόνο που απαιτείται για την εκτέλεση της **Match ROM** συν τον χρόνο που απαιτείται για την εκτέλεση της **Read Scratchpad**.

$$\mathbf{T_{Read}} = 960\mu s + 4.392\mu s + 1.464\mu s = 6.816\mu s$$

Άρα ο συνολικός χρόνος της **Main** θα είναι:

$$\mathbf{t_{Main}} = 960\mu s + 488\mu s + 488\mu s + 500.000\mu s + n * t_{Read} = 501.936\mu s + n*6.816\mu s$$

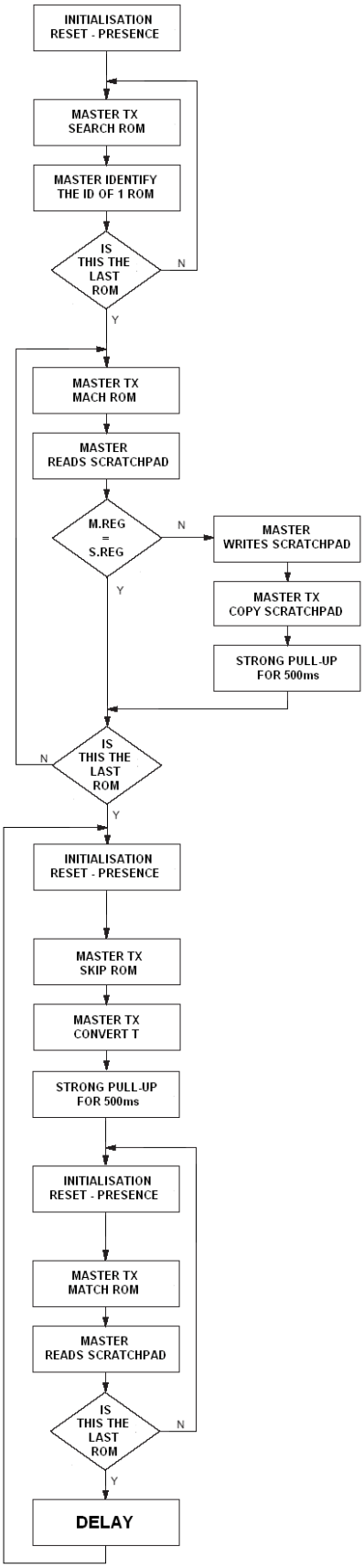
Όπου n ο αριθμός των αισθητηρίων που βρίσκονται συνδεδεμένα στον δίαυλο. Στην περίπτωση της **n=4** οπότε ο χρόνος της **Main** θα είναι:

$$\mathbf{t_{Main}} = 529.200\ \mu s$$

Άρα η χρονοκαθυστέρηση θα είναι:

$$\mathbf{t_{Delay}} = 60s - t_{Main} = 59{,}4708s$$

Στο **σχήμα 3.14** φαίνεται το συνολικό διάγραμμα ροής της εφαρμογής



σχήμα 3.14: Συνολικό διάγραμμα ροής της εφαρμογής

ΚΕΦΑΛΑΙΟ 4

ΔΙΚΤΥΑ MICROLAN

ΚΑΝΟΝΕΣ ΓΙΑ ΤΗΝ ΛΕΙΤΟΥΡΓΙΑ ΤΟΥ MICROLAN

ΕΛΕΓΧΟΣ ΤΟΥ SLEW RATE

Το **MicroLAN** αποτελείται από τρία στοιχεία, τον **Master**, καλώδια και βύσματα και από τις συσκευές **1-Wire**. Είναι ένα ***open drain*** **Master to Multi Slave** δίκτυο που τυπικά χρησιμοποιεί ένα αντιστάτη **Pull-Up** με ονομαστική τάση λειτουργίας **5 Volt**. Ο κύκλος επικοινωνίας αρχίζει όταν το τρανζίστορ στον **Master** τραβάει ενεργά τη γραμμή στο λογικό "**0**", οι **Slave 1-Wire** συσκευές κρατούν τη γραμμή στο λογικό **0** και ο αντιστάτης **Pull-Up** επιστρέφει τη γραμμή στο λογικό "**1**"**,** αφού έχουν απελευθερώσει τη γραμμή τόσο ο **Master** όσο και οι **Slave** συσκευές.

Αν το τρανζίστορ τραβήξει τη γραμμή χαμηλά για λιγότερο χρόνο απ' όσο η μετάβαση χρειάζεται για να διασχίσει το μήκος του καλωδίου, το **MicroLAN** λειτουργεί σε γραμμή μεταφοράς και οι ανακλάσεις από το τέλος της γραμμής μπορεί να προκαλέσουν **bit errors**. Επειδή δεν είναι δυνατόν να τερματίσουμε το καλώδιο του **MicroLAN** στην χαρακτηριστική του σύνθετη αντίσταση, όταν υπάρχουν αυτές οι συνθήκες το **Slew Rate** του τρανζίστορ διαύλου θα πρέπει να ελέγχεται. Ένα **Slew Rate** της τάξης των **3** με **4 microseconds** συνίσταται για μήκη διαύλων που ξεπερνούν τα **300** μέτρα.

Κατά την ανυψούμενη ακμή του παλμού, όσο το χωρητικό φορτίο του **MicroLAN** αυξάνει, είτε με την προσθήκη συσκευών **1-Wire**, είτε με την χωρητικότητα του καλωδίου (αυξάνοντας το μήκος της γραμμής). Ο χρόνος επιστροφής της γραμμής στην τάση τροφοδοσίας αυξάνει. Αν το συνολικό παραγόμενο χωρητικό φορτίο (το οποίο περιλαμβάνει τη γραμμή, τις συσκευές και τις παρασιτικές χωρητικότητες) και η αξία του αντιστάτη **Pull-Up** έχουν ως αποτέλεσμα μία σταθερά χρόνου **RC** η οποία υπερβαίνει την χρόνο θυρίδα που έχει οριστεί από το πρωτόκολλο για το **bit**, τότε παύει η επικοινωνία. Γι' αυτόν τον λόγο αχρησιμοποίητα σύρματα στο καλώδιο θα πρέπει να μένουν ασύνδετα. Γειώνοντας τα μπορεί να αυξηθεί το χωρητικό φορτίο τόσο σημαντικά ώστε το **Pull-Up** να μην μπορεί να ανυψώσει τη γραμμή πάνω από το λογικό κατώφλι μετάβασης του **bit**. Αν η τιμή του

αντιστάτη **Pull-Up** βρίσκεται ήδη στην ελάχιστη τιμή (**1,5K**) που μπορεί να δημιουργήσει ένα αναγνωρίσιμο μηδενικό επίπεδο, τότε θα πρέπει να αντικατασταθεί από ένα ενεργό **Pull-Up**. Βέβαια, οι ίδιοι κανόνες που ισχύουν για το ενεργό **Pull-Up** ισχύουν και για το **Pull-Down**, και το **Slew Rate** πρέπει να ελέγχεται όταν λειτουργεί σε γραμμή μεταφοράς. Το παθητικό **Pull-Up** με αντιστάτη, λίγο μας απασχολεί, γιατί η σταθερά χρόνου **RC** έχει εγγενώς αργό **Slew Rate**. **(βλ. σχήμα 4.1)**

ΣΩΣΤΗ ΕΠΙΛΟΓΗ ΚΑΛΩΔΙΟΥ

Η απόδοση ενός **MicroLAN** δεν εξαρτάται τόσο από τον αριθμό τον συσκευών που βρίσκονται συνδεδεμένες στη γραμμή, αλλά από το καλώδιο που συνδέει τον **Master** με τις **1-Wire** συσκευές. Κατά κύριο μέρος απ' αυτό εξαρτώνται τα όρια του **MicroLAN** δικτύου. Για μικρές αποστάσεις μέχρι **30** μέτρα, η επιλογή καλωδίου για **MicroLAN** είναι απλή, ακόμα και ένα απλό τηλεφωνικό καλώδιο μπορεί να δουλέψει με μικρό αριθμό **1-Wire** συσκευών.

Ωστόσο, όσο επιμηκύνεται το δίκτυο τόσο μεγαλύτερη σημασία έχουν οι ιδιότητες του καλωδίου και κατά συνέπεια και η επιλογή του. Τα καλώδια εμφανίζουν διακριτικές ιδιότητες την αντίσταση, την χωρητικότητα και την αυτεπαγωγή, οι οποίες καθορίζονται από τη γεωμετρία του καλωδίου και από το μέγεθος και την απόσταση των αγωγών και των διηλεκτρικών που τους περιβάλει. Αυτές οι φυσικές ιδιότητες καθορίζουν τη χαρακτηριστική σύνθετη αντίσταση, το εύρος του σήματος που μπορεί να υποστηρίξει και την ταχύτητα διάδοσης του καλωδίου. Συγκεκριμένα η αντίσταση του καλωδίου μειώνει το **Zero Logic Level Noise Margin**, αν και τιμές μέχρι **100 ohm** είναι αποδεκτές.

Ωστόσο η χωρητικότητα του καλωδίου, η οποία μπορεί να κυμαίνεται από **30 pF/m** μέχρι **100 pF/m**, φορτίζει τον οδηγό του **MicroLAN**, αυξάνοντας όχι μόνο τη σταθερά χρόνου **RC** κατά το **Pull-Up**, αλλά επίσης και την κορυφή του ρεύματος που ρέει στο καλώδιο καθώς το τρανζίστορ του **Master** ανοίγει και εκφορτίζει τη γραμμή. Αν αυτό το τρανζίστορ κλείσει πριν προλάβει να εκφορτιστεί τελείως το φορτίο της γραμμής (φορτίο που αποθηκεύεται στη χωρητικότητα της γραμμής), το υπόλοιπο φορτίο που διαρρέει τη γραμμή είναι

αυτό που καθορίζει το πλάτος της επαγωγικής τάσης που δημιουργείται. Το αποτέλεσμα της τάσης που βλέπει ο οδηγός γίνεται αρκετά μεγάλο ώστε να παρεμβάλλεται στην επικοινωνία. Στο απομακρυσμένο τέλος του καλωδίου αυτή η επαγωγική τάση ταλαντεύεται αρνητικά, αντιστρέφοντας την τάση της συσκευής που βρίσκεται πιο κοντά στο τέλος του καλωδίου. Η επαγωγιμότητα που μας ενδιαφέρει είναι η διαφορική αυτεπαγωγή που μετριέται στην είσοδο του καλωδίου με τα απομακρυσμένα άκρα του βραχυκυκλωμένα.

Η διαφορική αυτεπαγωγή είναι σημαντικά μικρότερη από την αυτεπαγωγή ενός μονού καλωδίου διότι το ρεύμα ρέει σε αντίθετες κατευθύνσεις με αποτέλεσμα, σε ιδανικές καταστάσεις να εξουδετερώνεται τελείως. Η διαφορική αυτεπαγωγή μειώνεται καθώς μειώνεται η απόσταση μεταξύ των αγωγών, έτσι η χρήση παράλληλων ζευγών, ή κατά προτίμηση τα συνεστραμμένα ζευγάρια συστήνονται. Τα συνεστραμμένα ζεύγη μειώνουν τις ανεπιθύμητες συζεύξεις με γειτονικές πηγές παρεμβολής διότι τα ρεύματα ρέουν στα καλώδια σε αντίθετες κατευθύνσεις στους δύο αγωγούς και τείνουν να εξαλειφθούν. Ένα τηλεφωνικό καλώδιο κατηγορίας **5** με συνεστραμμένα ζεύγη συστήνεται σε όλες τις εφαρμογές εκτός από τις πιο απαιτητικές.

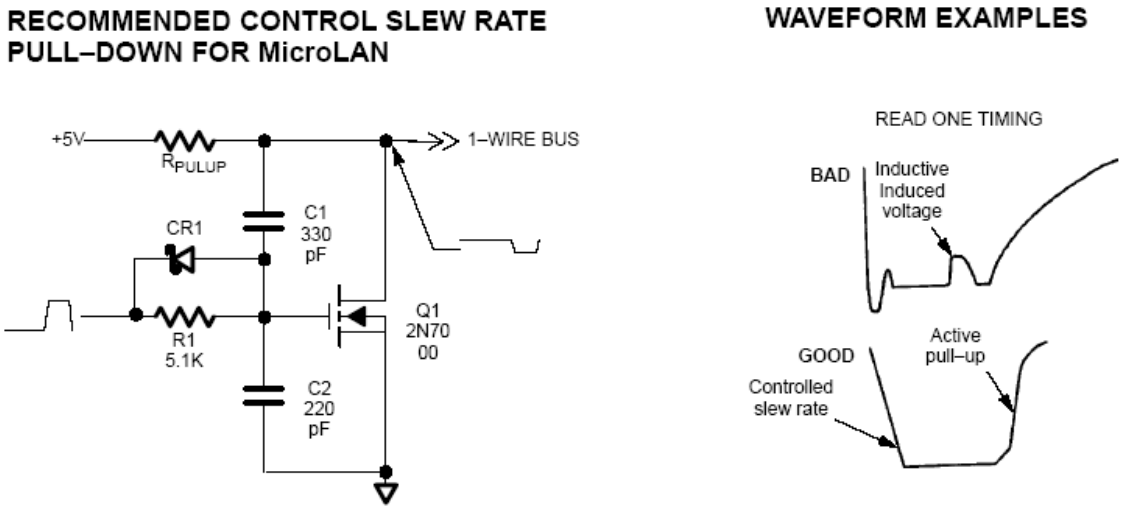
Όταν στο μήκος της γραμμής, ο αριθμός των συσκευών ή περισσότερα από ένα **MicroLAN** ανά καλώδιο απαιτούνται, χρησιμοποιούμε καλώδιο με προδιαγραφές **IEEE 1394** (“**FireWire**”). Το καλώδιο **FireWire** είναι διπλά θωρακισμένο, περιέχει δύο ξεχωριστά θωρακισμένα συνεστραμμένα ζεύγη και δύο ξεχωριστούς αγωγούς για μεταφορά ενέργειας. Η ελεγχόμενη εμπέδηση και η χαμηλή αυτεπαγωγή παρέχουν εξαιρετικές επιδώσεις στα **MicroLAN**.

ΚΡΑΤΩΝΤΑΣ ΤΟ ΔΙΑΥΛΟ ΣΕ ΛΕΙΤΟΥΡΓΙΑ

Όπως αναφέρθηκε προηγουμένως, καθώς η αυτεπαγωγή στη γραμμή αυξάνει η παράγωγος **L di/dt** μπορεί να δημιουργήσει εξάρσεις τάσης οι οποίες προκαλούν **bit errors** και αντιστροφή πολικότητας στην ποιο απομακρυσμένη συσκευή στο τέλος του καλωδίου. Αυτές οι τάσεις δημιουργούνται από ρεύματα που ακόμη ρέουν στις γραμμές δεδομένων και επιστροφής του καλωδίου όταν το τρανζίστορ στον **Master** κλείσει πριν το αποθηκευμένο

χωρητικό φορτίο της γραμμής εκφορτιστεί πλήρως. Η προτεινόμενη και προφανέστερη λύση είναι να διατηρήσουμε το **Pull-Down** τρανζίστορ σε αγώγιμη κατάσταση εωσότου το ρεύμα της γραμμής εκφορτιστεί **(βλ. σχήμα 4.1)**. Αν δεν έχουμε τα χρονικά περιθώρια για να το πραγματοποιήσουμε, τοποθετούμε μια δίοδο **Schottky** στο απομακρυσμένο τέλος του διαύλου για να αποκόψει τις επαγωγικές τάσεις. Συνδέουμε την δίοδο στα άκρα του καλωδίου με την κάθοδο στην γραμμή δεδομένων και την άνοδο στην επιστροφή. Μόνο μια δίοδος απαιτείται για κάθε **MicroLAN** με πρόβλημα αυτεπαγωγής.

Ακολουθώντας τους κανόνες που παρατίθενται εδώ, χρησιμοποιώντας καλώδιο κατηγορίας **5** συνεστραμμένων ζευγών, ελέγχοντας το **Slew Rate** και αντικαθιστώντας με ενεργό **Pull-Up** μπορούμε να πετύχουμε αξιόπιστες επικοινωνίες σε αποστάσεις μεγαλύτερες των **300** μέτρων σε καλώδιο όπου φέρει περισσότερες από **500** διαφορετικές **1-Wire** συσκευές. Χωρίς τον έλεγχο του **Slew Rate** και το ενεργό **Pull-Up** το όριο είναι περίπου **100** μέτρα με **150** συσκευές.



(σχήμα 4.1)

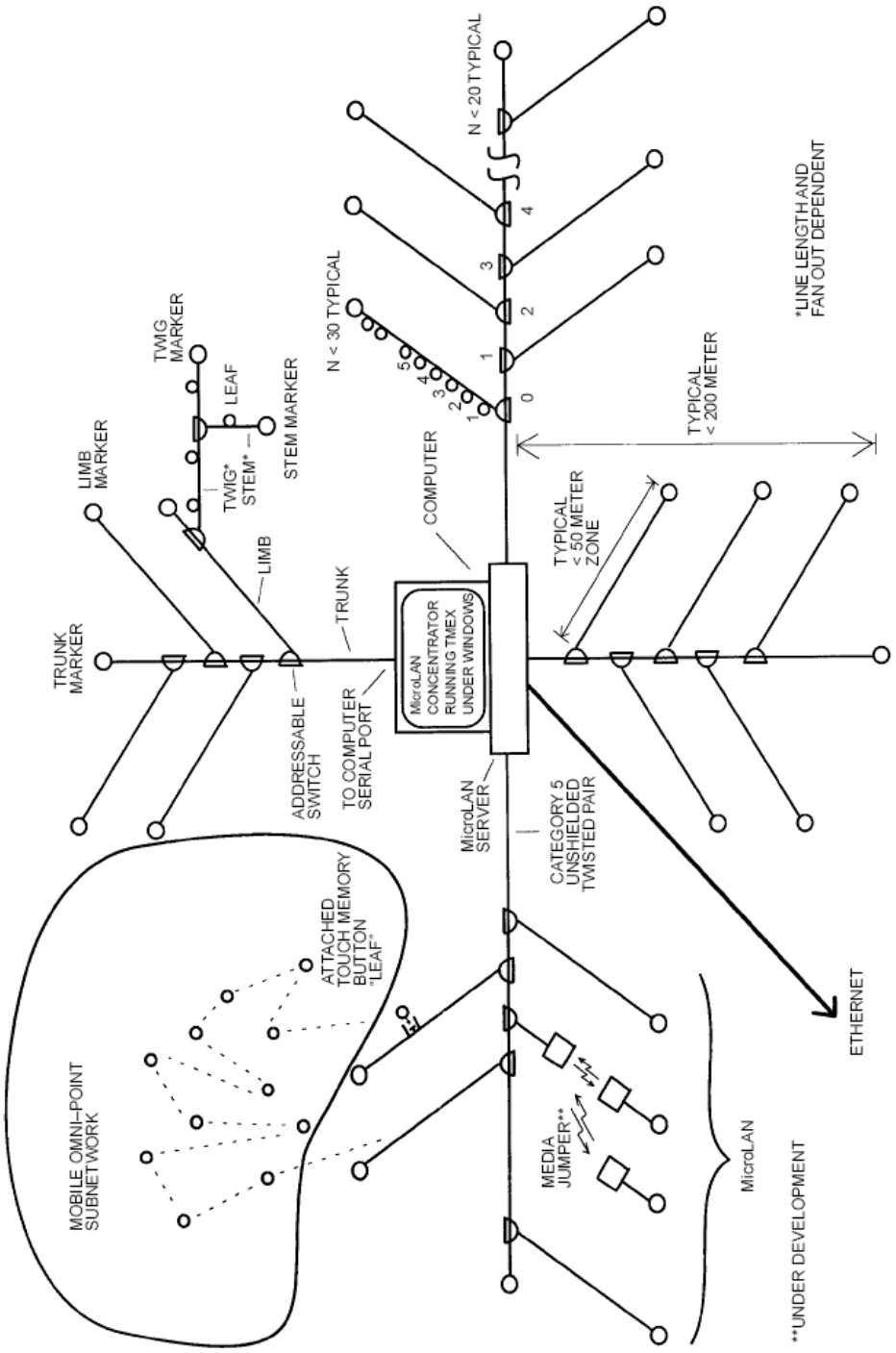
ΣΥΝΘΕΤΑ ΔΙΚΤΥΑ MICROLAN

ΤΟΠΟΛΟΓΙΑ

Το **MicroLAN** είναι ένα αθροιστικό δίκτυο, το οποίο χρησιμοποιεί μία γραμμή δεδομένων και μια γείωση ως αναφορά, για επικοινωνία. Παρέχει ψηφιακή πληροφορία, με πολύ χαμηλό κόστος, στους **H/Y** και στα δίκτυά τους. Το **MicroLAN** (**βλ. σχήμα 4.2**) είναι ένα δίκτυο που μπορούμε να το παρομοιάσουμε με την μορφή ενός δέντρου, αποτελούμενο από ένα κύριο κορμό (***Trunk***) με πολλά κλαδιά. Ο κύριος κορμός του συνδέεται μέσω ενός οδηγού **1-Wire** με την σειριακή θύρα (**RS232**) ενός **H/Y**, ο οποίος θα έχει το ρόλο του ***server MicroLAN*** σε ένα μεγαλύτερο δίκτυο. Αυτός ο **H/Y** επίσης αναφέρεται και ως ελεγκτής διαύλου. Τρέχει ένα λογισμικό επικοινωνίας ***1-Wire*** π.χ. το ***"TMEX"*** κάτω από **DOS** ή **WINDOWS.**

Στο πιο απομακρυσμένο σημείο του κορμού συνδέεται μία μνήμη αφής (***Touch Memory***) που ονομάζεται δείκτης κορμού (***Trunk Marker***). Κατά μήκος του κορμού βρίσκονται διυθυνσιοδοτούμενοι διακόπτες οι οποίοι διακλαδίζουν τον κορμό σε άκρα που ονομάζονται (***Limbs***). Τα άκρα έχουν επίσης τους δείκτες άκρων (***Limb Marker***) στο τέλος της γραμμής τους. Οι διακλαδώσεις που εκτείνονται από τα **Limbs** καλούνται μικροί κλάδοι (***Twigs***) οι οποίοι έχουν διακλαδώσεις που επεκτείνονται σε πιο μικρά κλαδιά (***Stems***).

Σε κάθε περίπτωση, κάθε διακλάδωση έχει τον δικό της δείκτη στο τέλος της διαδρομής της. Οι λειτουργία των δεικτών είναι να επιβεβαιώσουν ότι υπάρχει αξιόπιστη επικοινωνία ως το τέλος της κάθε διαδρομής του δικτύου. Οι δείκτες διαφέρουν από τις υπόλοιπες συσκευές, από το περιεχόμενο των δεδομένων τους. Στην γενική μορφή, φορητές μνήμες αφής, αισθητήρες και **I/O** διακόπτες συνδέονται σαν φύλλα (***Leaves***) στα **Limbs**, **Twigs** και **Stems** αλλά όχι πάνω σε **Trunk**. Στην περίπτωση που δεν υπάρχουν διακλαδώσεις οι συσκευές συνδέονται στο **Trunk**.



σχήμα 4.2: MicroLAN δίκτυο

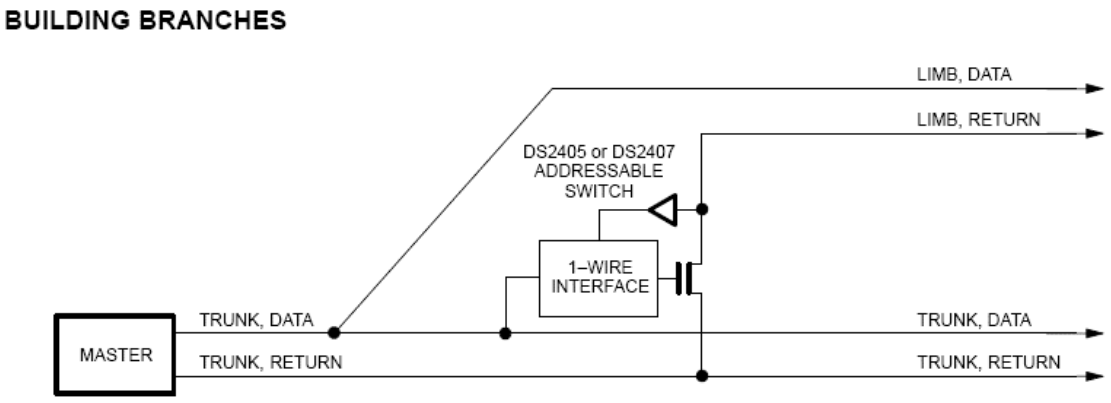
Για να επικοινωνήσουμε με μία συσκευή η οποία βρίσκεται σε ένα **Twig**, θα πρέπει να ενεργοποιηθούν ο διευθυνσιοδοτούμενος διακόπτης που συνδέει το **Limb** στο **Trunk** και ο διακόπτης που συνδέει το **Twig** στο **Limb**. Η δυνατότητα να ενεργοποιούμε διακόπτες ελεγχόμενους από λογισμικό, μας επιτρέπει να στήσουμε τεράστια δίκτυα αποφεύγοντας την υπερφόρτιση της γραμμής με πολλές συσκευές. Επιπλέον ο διακόπτης μας παρέχει πληροφορίες για την φυσική θέση του κλάδου, πράγμα που θα ήταν αδύνατο εάν όλες οι συσκευές ήταν συνδεδεμένες στο **Trunk**.

Τα δεδομένα που επικοινωνούν μέσω του **MicroLAN** είναι σε μορφή πακέτων. Είτε οι συσκευές του **MicroLAN** προσθέτουν **CRC** στα δεδομένα ή τα δεδομένα από μόνα τους πακετάρονται πριν εγγραφούν στην συσκευή. Αν για οποιονδήποτε λόγο το πακέτο καταστραφεί κατά την μεταφορά δεδομένων, ο ελεγκτής διαύλου μπορεί να ανιχνεύσει το σφάλμα ελέγχοντας το **CRC** και να επαναλάβει την μεταφορά δεδομένων. Η επικοινωνία στο **MicroLAN** δεν είναι event driven. Ο **Master** επιλέγει τους κόμβους και ελέγχει αν απαιτείται να λάβει κάποια δραστηριότητα. Ένας **H/Y** μπορεί να υποστηρίξει τόσα ανεξάρτητα δίκτυα **MicroLAN**, όσες οι σειριακές θύρες με τις οποίες είναι εξοπλισμένος. Έτσι αυξάνεται η επίδοση ή το συνολικό μέγεθος το αθροιστικού δικτύου.

ΔΗΜΙΟΥΡΓΩΝΤΑΣ ΔΙΑΚΛΑΔΩΣΕΙΣ

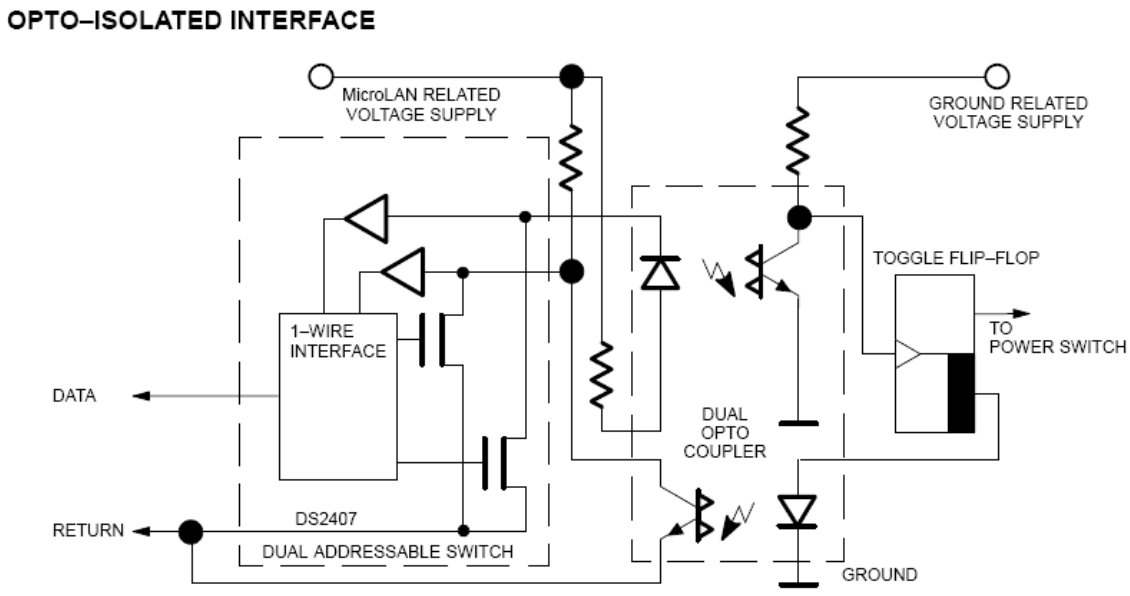
Για να υλοποιήσουμε τεράστια αθροιστικά δίκτυα χωρίς να αυξάνεται ταυτόχρονα το ηλεκτρικό φορτίο. Το **MicroLAN** χρησιμοποιεί τοπολογία δέντρου αποτελούμενη από διάφορα επίπεδα διακλαδώσεων, τα οποία τελικά συνδέονται με τον κορμό. Το κύριο στοιχείο ή συστατικό της ενεργοποίησης και απενεργοποίησης διακλαδώσεων (π.χ. φτιάχνοντας συνδέσεις ελεγχόμενες από λογισμικό) είναι ο διευθυνσιοδοτούμενος διακόπτης. Αυτή η συσκευή είναι βασικά ένα τρανζίστορ το οποίο μπορεί να ελέγχεται απομακρυσμένα (να ανοίγει να κλείνει ή να ανιχνεύεται) μέσω της διασύνδεσης του **MicroLAN**. Ο τρόπος με τον οποίο ο διακόπτης συνδέει ένα **Limb** με τον **Trunk** (ή δημιουργεί ένα νέο επίπεδο διακλάδωσης κάπου στο **MicroLAN**) περιγράφεται στο **σχήμα 4.3**. Η γραμμή δεδομένων του

MicroLAN είναι μόνιμα συνδεδεμένη με όλες τις συσκευές του συστήματος. Ωστόσο η γραμμή της επιστροφής άγει, μόνο στις διακλαδώσεις όπου υπάρχουν συσκευές με τις οποίες ο **Master** προσπαθεί να επικοινωνήσει.



σχήμα 4.3: τρόπος σύνδεσης με διακόπτη ενός Limb με ένα Trunk

Σημειώστε ότι η γραμμή επιστροφής του **MicroLAN** δεν είναι ιδανική ως προς την γείωση του συστήματος. Για αποφυγή βρόγχων γείωσης, χρειάζεται ένα κύκλωμα οπτικής απομόνωσης για επικοινωνία με τα κυκλώματα που πρέπει να γειωθούν. Στο **σχήμα 4.4** παραθέτουμε ένα παράδειγμα διασύνδεσης οπτικής απομόνωσης.



σχήμα 4.4: διασύνδεση οπτικής απομόνωσης.

ΕΞΕΤΑΣΗ ΛΟΓΙΣΜΙΚΟΥ

Μετά την τροφοδότηση του **MicroLAN**, ο ελεγκτής θα δει μόνο τις **1-Wire** συσκευές που βρίσκονται στον κορμό (***Trunk***). Οι συσκευές αυτές θα είναι διευθυνσιοδοτούμενοι διακόπτες και ο δείκτης κορμού (***Trunk Marker***) πριν λάβει χώρα οποιαδήποτε επικοινωνία εκτός του **Trunk** ο ελεγκτής διαύλου πρέπει να μάθει την τοπολογία του δικτύου. Γι' αυτό το πρώτο βήμα είναι να πραγματοποιήσει μία ανάλυση μόνο τον διακοπτών του **MicroLAN**.

Ξεκινώντας από τον **Trunk**, ο ελεγκτής διαύλου ανοίγει συστηματικά τους διακόπτες σε κάθε **Limb** τον ένα μετά τον άλλο και καταγράφει τους αριθμούς καταχώρησης του κάθε διακόπτη που βρίσκει, έπειτα ανοίγει τους διακόπτες αυτούς ένα προς ένα και ούτε καθεξής, μέχρι να σχηματίσει ένα ολοκληρωμένο μοντέλο της τοπολογίας δικτύου στην μνήμη. Στο δεύτερο βήμα ο ελεγκτής πρέπει να ταυτοποιήσει όλες τις υπόλοιπες συσκευές του **MicroLAN** οι οποίες δεν είναι διακόπτες. Ξεκινώντας από τον **Trunk** και χρησιμοποιώντας το εσωτερικό δενδρόγραμμα που έχει αναπαράγει στο πρώτο βήμα. Ο ελεγκτής ανοίγει συστηματικά τους διακόπτες σε κάθε **Limb** τον ένα μετά τον άλλο και καταγράφει τους αριθμούς καταχώρησης των συσκευών που βρίσκει οι οποίες δεν είναι διακόπτες, μέχρι να προσθέσει όλες τις επιπρόσθετες συσκευές στο μοντέλο τοπολογίας δικτύου της μνήμης.

Καθώς ο ελεγκτής γνωρίζει την τοπολογία και το πλήθος όλων των συσκευών στο **MicroLAN**, είναι σε θέση να δημιουργήσει την διαδρομή κάθε συσκευής του δικτύου και να επικοινωνήσει μαζί της. Αφού ο **Master** διευκρινίσει σε πια συσκευή ο χρήστης θέλει να έχει πρόσβαση, ανοίγει τους απαραίτητους διακόπτες για να φτάσει σε αυτή την συσκευή και την απομονώνει από τις υπόλοιπες με μία εντολή **Match ROM**. Σε αυτό το σημείο, όλες οι συσκευές είναι απενεργοποιημένες εκτός από την συγκεκριμένη συσκευή, η οποία είναι έτοιμη να δεχτεί μία εντολή λειτουργίας μνήμης. Αυτό το πλάνο επιτρέπει ανάγνωση και εγγραφή δεδομένων κάθε συσκευής του δικτύου, ανεξάρτητα από την θέση του.

ΤΟΠΟΛΟΓΙΑ ΕΛΑΧΙΣΤΟΥ ΦΟΡΤΙΟΥ

Για την καλύτερη απόδοση εγκατάστασης ενός **MicroLAN** πρέπει να βρούμε τις συνθήκες κατά τις οποίες το φορτίο του **MicroLAN** ελαχιστοποιείται για συγκεκριμένο αριθμό κόμβων. Θεωρούμε μία τοπολογία όπου ο αριθμός των κλάδων κάθε επιπέδου διακλάδωσης είναι σταθερός. Ο βέλτιστος αριθμός επιπέδων διακλάδωσης μπορεί να υπολογιστεί.

Για να εξηγήσουμε πως πραγματοποιείται η βελτιστοποίηση πρέπει να προσδιορίσουμε μερικές παραμέτρους:

M = Ο αριθμός των συσκευών (εκτός των διακοπτών) που πρέπει να προσεγγιστούν μέσω του αθροιστικού δικτύου.

N = Ο αριθμός των επιπέδων διακλάδωσης (**Limb** για πρώτο επίπεδο, **Twig** δεύτερο επίπεδο, **Stem** για τρίτο επίπεδο, κ.τ.λ.).

S = Ο αριθμός των διακλαδώσεων συμπεριλαμβανομένων και των **Leaves** (εκτός από τους δείκτες).

L = Το συνολικό φορτίο, προσδιορίζεται ως ο αριθμός των ενεργών συσκευών σε μία διαδρομή του **MicroLAN**.

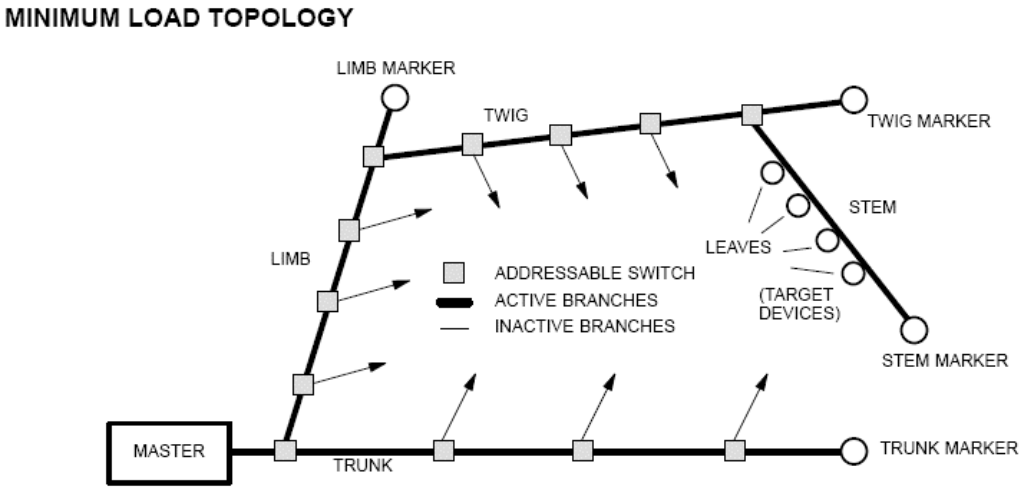
Μπορεί να αποδειχθεί θεωρητικά ότι ο βέλτιστος αριθμός διακλαδώσεων σε κάθε επίπεδο διακλάδωσης είναι περίπου **3,6** και αυτή η τιμή δεν εξαρτάται από το **M**. Αυτό το αποτέλεσμα ισχύει για διάταξη όπου κάθε κλαδί έχει μόνο διακόπτες και ένα δείκτη. Οι συσκευές που θέλουμε να απευθυνθούμε βρίσκονται στα πιο απομακρυσμένα κλαδιά. Ωστόσο η μη ακέραια τιμή του **3,6** δεν μπορεί να πραγματοποιηθεί στην πράξη, για το παράδειγμα που ακολουθεί έχουμε επιλέξει την τιμή **4**. Ο **πίνακας 4.1** δείχνει πως η χωρητικότητα **M** και το φορτίο αυξάνουν σε κάθε επίπεδο διακλάδωσης. Αποδεικνύει επίσης ότι ακόμα και με περισσότερες από **4096** συσκευές το **DC Fan-Out** δεν αποτελεί παράγοντα περιορισμού για το μέγεθος του **MicroLAN**.

πίνακας 4.1: Σχέση χωρητικότητας και φορτίου

MINIMUM LOAD TOPOLOGY

N (NUMBER OF BRANCH LEVELS)	L (LOAD OF LISTENING DEVICES)	M (TARGET DEVICES)
1 (TRUNK ONLY)	5	4
2 (TRUNK, LIMBS)	10	16
3 (TRUNK, LIMBS, TWIGS)	15	64
4 (TRUNK, LIMBS, TWIGS, STEMS)	20	256
5	25	1024
6	30	4096

Στο **σχήμα 4.5** φαίνεται ένα παράδειγμα ενός **MicroLAN** τεσσάρων επιπέδων διακλάδωσης σύμφωνα με τον παραπάνω πίνακα. Ο αριθμός των συσκευών που "**ακούν**" βρίσκεται μετρώντας τους διακόπτες, τους δείκτες των ενεργών κλάδων και των "**φύλλων**" (οι περιφερειακές συσκευές που θέλουμε να προσπελάσουμε ονομάζονται "**φύλλα**") στο πιο απομακρυσμένο κλαδί (**μέγιστο 20**). Εφόσον έχουμε **4** "**φύλλα**" σε κάθε **Stem**, **4 Stem** σε κάθε **Twig**, **4 Twig** σε κάθε **Limb** και **4 Limb** στον **Trunk**, ο αριθμός των περιφερειακών συσκευών (χωρίς την καταμέτρηση διακοπτών και δεικτών) είναι **4*4*4*4 = 256** χωρίς να έχουμε περισσότερες από **20** συσκευές ταυτόχρονα ενεργές.



σχήμα 4.5: MicroLAN τεσσάρων επιπέδων διακλάδωσης

ΣΥΝΟΨΙΖΟΝΤΑΣ

Η εφαρμογή ενός μεγάλου και υπερφορτωμένου **MicroLAN** θα δουλέψει πιο αξιόπιστα αν πληρούνται οι παρακάτω συνθήκες.

- Ελεγχόμενο **Slew Rate** της τάξεως των **3-4 μs.**
- Ελάχιστη χωρητικότητα και αυτεπαγωγή καλωδίου (καλώδιο κατηγορίας **5** συνεστραμμένων ζευγών ή **IEEE 1394** "**FireWire**").
- Πλήρη εκφόρτιση της χωρητικότητας της γραμμής σε κάθε χρονοθυρίδα.
- Τερματισμός της γραμμής με ανάστροφα πολωμένη δίοδο **Schottky**.
- Κύκλωμα οδηγού ενεργού **Pull-Up**.
- Αργή δειγματοληψία στα **2,5T** ή **21,7 μs** αντίστοιχα.
- Μέγιστη **1-Wire** τάση **Pull-Up**.
- Τοπολογία πολλαπλών επιπέδων διακλαδώσεων.

ΠΑΡΑΡΤΗΜΑ Α

DS18B20 “PROGRAMMABLE RESOLUTION 1-WIRE DIGITAL THERMOMETER”

AN 27 “UDERSTANDIN AND USING CYCLIC REUDANDANCY CHECKS”

AN 106 “COMPLEX MICROLANS”

AN 108 “MICROLAN – IN THE LONG RUN”

AN 3438 “SERIAL DIGITAL DATA NETWORKS”

ΠΑΡΑΡΤΗΜΑ Β

ΒΙΒΛΙΟΓΡΑΦΙΑ

Jon S. Wilson
Copyright © 2005
Sensor Technology Handbook
Newnes

Arnold S. Berger
Copyright © 2002
Embedded Systems Design: An Introduction to Processes, Tools, and Techniques
CMP Books

John Catsoulis
Copyright © 2002
Designing Embedded Hardware
O'Reilly

Dan Eisenreich & Brian DeMuth
Copyright © 2003
Designing Embedded Internet Devices
Newnes

Jan Axelson
Copyright © 2003
Embedded ETHERNET AND INTERNET COMPLETE
Lakeview Research LLC

ΒΙΒΛΙΟΓΡΑΦΙΚΕΣ ΑΝΑΦΟΡΕΣ

DALLAS SEMICONDUCTOR
HTTP://WWW.DALSEMI.COM

STANDARD
The Complete Book of iButton™ Memories

DS18B20
Programmable Resolution
1-Wire Digital Thermometer

Tech Brief 1
MicroLAN Design Guide

Application Note 27
Understanding and Using Cyclic Redundancy Checks with Dallas Semiconductor iButton™ Products

Application Note 74
Reading and Writing iButtons via Serial Interfaces

Application Note 106
Complex MicroLANs

Application Note 108
MicroLAN – In The Long Run

Application Note 3438
Serial Digital Data Networks