



ΕΛΛΗΝΙΚΟ ΜΕΣΟΓΕΙΑΚΟ  
ΠΑΝΕΠΙΣΤΗΜΙΟ  
ΕΛ.ΜΕ.ΠΑ.  
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ  
ΤΜΗΜΑ  
ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ



ΗΛΕΚΤΡΟΝΙΚΑ ΣΥΣΤΗΜΑΤΑ  
ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ &  
ΑΥΤΟΜΑΤΙΣΜΟΥ  
(TeleAutoS)

**ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ**  
----- **THESIS** -----

**MEDIBERRY**

Automatic **ME**dication **DI**spenser using Rasp **BERRY** Pi

**Όνοματεπώνυμο:**

**Αντώνιος Ι. Ζερβουδάκης**

**Εισηγητής:**

**Δρ. Αντώνιος Κωνσταντάρας**

**(c) 2020**

## Ευχαριστίες:

Σε αυτό το σημείο θα ήθελα να ευχαριστήσω:

- Πρώτα την γυναίκα μου Γεωργία και τα παιδιά μου Γιάννη και Στέλιο που έκαναν τόσο καιρό υπομονή για να ολοκληρώσω το Μεταπτυχιακό αυτό.
- Τους γονείς μου που με μεγαλώσανε και με σπούδασαν και μου δώσανε όλα τα εφόδια για να γίνω σωστός άνθρωπος.
- Όλο το προσωπικό της Σχολής Εφαρμοσμένων Επιστημών του ΤΕΙ Κρήτης νυν Ελληνικό Μεσογειακό Πανεπιστήμιο που μου δώσανε με τις Σπουδές μου στο Προπτυχιακό Κύκλο Σπουδών τα εφόδια να τελειώσω το Τμήμα Ηλεκτρονικής.
- Τον εισηγητή μου Δρ. Αντώνη Κωνσταντάρα που μου έδωσε την ευκαιρία να εμπλουτίσω τις γνώσεις μου πάνω στο αντικείμενο της IoT Τεχνολογίας.
- Και στο τέλος τον μέντορα και συνεργάτη μου Γεώργιο Λιοδάκη για τις πολλές και πολύωρες συζητήσεις που κάναμε πάνω σε θέματα δικτύων Η/Υ και IoT.

## Abstract:

In this Thesis we are dealing with the Objective for another possibility to deliver the per scripted medication to the patient by using IoT Technology.

The terminology IoT Technology is in the present day a way to make things easier and better. It has penetrated in all possible dimensions and ways into everyday work and flows.

One of these innovations is an autonomous vehicle following a designated route using the following the Line Algorithm and Technology. This automation seems to be used also in other possible ways.

Our difference is that we use the same implementation supervised by a microcomputer Raspberry PI for the specific and special deliverance of Medication on an autonomous vehicle following a designated route. The implementation of this system consisted of a wireless network, devices and sensors which have been deployed in a way to fulfill the requirements. The program in this network is achieved by using Python as a programming language. The purpose of this network is supervising the autonomous vehicle, input of the different deliverance variables (which medication box port applies to which patient) designed from Scratch.

We call this Project **MEDIBERRY**. Automatic Medication **D**ispenser using **Rasp**berry Pi.

## Σύνοψη:

Σε αυτή την εργασία θα ασχοληθούμε με στόχο την δυνατότητα παράδοσης φαρμάκων σε ασθενείς χρησιμοποιώντας την τεχνολογία IoT.

Η τεχνολογία IoT είναι σήμερα ένας τρόπος για να διευκολύνουμε και να βελτιώσουμε τα πράγματα. Έχει διεισδύσει σε όλες τις δυνατές πτυχές και διαστάσεις και τρόπους της καθημερινότητας στον εργασιακό και ιδιωτικό χώρο και όχι μόνο.

Μία από αυτές τις καινοτομίες είναι η χρήση ενός αυτόνομου οχήματος ακολουθώντας μια προκαθορισμένη διαδρομή χρησιμοποιώντας τον Αλγόριθμο παρακολούθησης γραμμής. Αυτή η αυτοματοποίηση φαίνεται να χρησιμοποιείται και με άλλους πιθανούς τρόπους και συνδυασμούς.

Η διαφορά μας είναι ότι χρησιμοποιούμε την ίδια εφαρμογή που ελέγχεται από έναν μικροϋπολογιστή Raspberry PI για την ειδική παράδοση φαρμάκων πάνω σε αυτόνομο όχημα ακολουθώντας μια προκαθορισμένη διαδρομή. Η εφαρμογή αυτού του συστήματος περιλαμβάνει ένα ασύρματο δίκτυο, διάφορες συσκευές και αισθητήρες που αλληλοεπιδρούν κατά τρόπο που να ικανοποιεί τις απαιτήσεις μας.

Ο προγραμματισμός σε αυτό το δίκτυο επιτυγχάνεται με τη χρήση της γλώσσας προγραμματισμού Python.

Ο σκοπός αυτού του δικτύου είναι η επίβλεψη του αυτόνομου οχήματος, η εισαγωγή των διαφορετικών μεταβλητών απελευθέρωσης (της αντίστοιχης θυρίδας φαρμάκων που ισχύει για τον ασθενή και που σχεδιάστηκε από το μηδέν).

**Ονομάζουμε αυτό το Project **MEDIBERRY** προερχόμενο από τα αρχικά**

**Αυτόματος Διανομέας Φαρμάκων με χρήση Raspberry Pi.**



## Περιεχόμενα

### Κεφάλαιο 1°

1.1 Περίληψη.....	10
1.2 Κίνητρο για την Διεξαγωγή της Εργασίας.....	11
1.3 Σκοπός και Στόχοι Εργασίας.....	12
1.4 Δομή Εργασίας.....	13

### Κεφάλαιο 2°

2.1 Τι είναι Τεχνολογίες του Internet (Internet of Things – IoT) .....	15
2.2 Τεχνολογίες που χρησιμοποιούμε στο Internet of Things.....	15
2.3 Κίνδυνοι και προκλήσεις των Internet of Things.....	16
2.4 Οφέλη της χρήσης του Internet of Things (IoT).....	17
2.5 Προκλήσεις του IoT.....	17
2.6 Πεδίο εφαρμογών του Internet of Things (IoT).....	17

### Κεφάλαιο 3°

3.1 Γνωριμία με το Raspberry Pi.....	19
--------------------------------------	----

### Κεφάλαιο 4°

4.1 Follow the Line με Raspberry Pi.....	21
4.2 Λειτουργία και τεχνική ακολούθησης γραμμής (Line Follower).....	21
4.3 Κυκλωματικό διάγραμμα για την ακολούθηση γραμμής.....	25
4.4 Προγραμματισμός για την ρουτίνα για την ακολούθηση γραμμής.....	28
4.5 Τροχοφόρο όχημα σε κίνηση με αλγόριθμο αντίχενυσης γραμμής.....	30
4.6 RFID.....	36
4.7 Relays.....	43
4.8 Μαγνητικός αισθητήρας.....	45

4.9 Servo κινητήρες.....	47
--------------------------	----

## Κεφάλαιο 5°

5.1 Εισαγωγικά.....	51
---------------------	----

5.2 Βήματα για την διασύνδεση της μονάδας Raspberry Pi με τον Υπολογιστή.....	51
---	----

5.3 Βήματα για την αυτοματοποιημένη εκκίνηση της μονάδας Raspberry.....	54
---	----

## Κεφάλαιο 6°

6.1 Εισαγωγικά.....	58
---------------------	----

6.2 Διάγραμμα ροής της υλοποίησης.....	58
--	----

6.3 Το πλήρες σχέδιο σε Fritzing.....	60
---------------------------------------	----

6.4 Το πλήρες πρόγραμμα σε Python.....	61
--	----

### 6.5 Σενάρια Υλοποίησης

a. Υλοποίηση σε πλήρη λειτουργία.....	67
---------------------------------------	----

b. Υλοποίηση με RFID σε λάθος θέσεις.....	67
---	----

c. Υλοποίηση με εμπόδιο στην διαδρομή.....	67
--	----

## Κεφάλαιο 7°

Μελλοντικές επεκτάσεις.....	69
-----------------------------	----

Μειονεκτήματα.....	69
--------------------	----

Πλεονεκτήματα.....	69
--------------------	----

## Κεφάλαιο 8°

Συμπεράσματα.....	70
-------------------	----

## Κεφάλαιο 9°

9.1 Βιβλιογραφία.....	71
-----------------------	----

9.2 Data Sheets.....	Appendix 1-156
----------------------	----------------

• <i>L298 Driver Motor</i> .....	<i>Appendix 2</i>
----------------------------------	-------------------

- *L298N-module-Information* ..... *Appendix 16*
- *IR Sensor* ..... *Appendix 21*
- *MFRC522 Contactless Reader IC* ..... *Appendix 26*
- *Raspberry PI* ..... *Appendix 120*
- *Server Motor SG90* ..... *Appendix 156*

## Λίστα Σχημάτων

<b>Σχήμα 1:</b> Το '20 θα έχουμε τουλάχιστον 50 δις IoT συσκευές.....	14
<b>Σχήμα 2α:</b> Διασύνδεση των πάντων.....	15
<b>Σχήμα 2β:</b> Τρόποι και λόγοι διασύνδεσης.....	15
<b>Σχήμα 3:</b> Πεδία εφαρμογών του Internet of Things.....	15
<b>Σχήμα 4:</b> Smart Document management σε γραφεία.....	18
<b>Σχήμα 5:</b> Πεδία εφαρμογών IoT τεχνολογίας.....	18
<b>Σχήμα 6:</b> Διάφορα μοντέλα Raspberry Pi και οι δυνατότητες τους.....	19
<b>Σχήμα 7:</b> Raspberry Pi 3 model B.....	20
<b>Σχήμα 8:</b> Raspberry Pi δίτροχο αμαξίδιο με αισθητήρια για ανίχνευση γραμμής.....	21
<b>Σχήμα 9:</b> α) με άσπρη ανακλώμενη επιφάνεια    β) σε μαύρη ανακλώμενη επιφάνεια .....	22
<b>Σχήμα 10:</b> IR αισθητήρες HW-006 για την υλοποίηση.....	22
<b>Σχήμα 11:</b> Μπλοκ διάγραμμα οδήγησης των τροχών από τους IR αισθητήρες.....	25
<b>Σχήμα 12:</b> Follow the Line with Fritzing.....	26
<b>Σχήμα 13:</b> Διάταξη του GPIO Header στις διάφορες εκδόσεις Raspberry Pi.....	26
<b>Σχήμα 14:</b> Raspberry Pi Zero v1.3 vs v1.1.....	27
<b>Σχήμα 15:</b> Έτοιμο όχημα για την υλοποίηση Follow the Line.....	28
<b>Σχήμα 16:</b> L298 Motor Driver.....	32
<b>Σχήμα 17:</b> Εικονική δοκιμαστική διαδρομή.....	33
<b>Σχήμα 18:</b> Πραγματική δοκιμαστική διαδρομή.....	34
<b>Σχήμα 19:</b> Αισθητήρας υπερήχων HC-SR04.....	34
<b>Σχήμα 20:</b> Αρχή λειτουργίας Ultrasonic Sensor.....	35

<b>Σχήμα 21:</b> Σύνδεση του αισθητήρα υπερήχων HC-SR04 στο Raspberry.....	35
<b>Σχήμα 22:</b> RFID tags τύπου κάρτας και ετικέτας.....	37
<b>Σχήμα 23:</b> Pin layout στο RFID-RC522.....	39
<b>Σχήμα 24:</b> Διασύνδεση του Raspberry Pi με το RFID-RC522.....	39
<b>Σχήμα 25:</b> Relays 4 καναλιών.....	44
<b>Σχήμα 26:</b> Αποτύπωση σε Fritzing για τη χρήση 4 Relays .....	45
<b>Σχήμα 27:</b> Μαγνητικός αισθητήρας ON/OFF.....	46
<b>Σχήμα 28:</b> Παραστατικός τρόπος λειτουργίας του μαγνητικού αισθητήρα.....	47
<b>Σχήμα 29:</b> Micro Step motor τύπου Tower Pro και διάταξη ακροδεκτών.....	47
<b>Σχήμα 30:</b> Αποτύπωση σε Fritzing για τη χρήση 4 Micro Server Motor.....	48
<b>Σχήμα 31:</b> Α) Pill box ευρείας χρήσης B) Κάτοψη Pill box                   Γ) Πλαϊνή όψη με εγκοπές χωρίς αισθητήρια.....	49
<b>Σχήμα 32:</b> Pill box με αισθητήρες μαγνητικούς και micro servo κινητήρα.....	50
<b>Σχήμα 33:</b> Ρυθμίσεις για πρόσβαση ssh.....	53
<b>Σχήμα 34:</b> Αρχική οθόνη στο Crontab.....	55
<b>Σχήμα 35:</b> Οθόνη στο Crontab μετά την εισαγωγή του script μας.....	56
<b>Σχήμα 36:</b> Αρχική οθόνη εκτέλεσης.....	57
<b>Σχήμα 37:</b> Πλήρες διάγραμμα ροής “Mediberry”.....	59
<b>Σχήμα 38:</b> Τελικό σχέδιο διάταξης μαζί με συνδέσεις.....	60

## Κεφάλαιο 1<sup>ο</sup>

### 1.1 Περίληψη

Σήμερα, το Internet of Things πιάνει ένα όλο και μεγαλύτερο κομμάτι στην αγορά των τεχνολογιών καθώς προσφέρει λύσεις και ευκολία στην καθημερινότητα ενός ανθρώπου. Ένα απλό δίκτυο αισθητήρων μπορεί να προσφέρει λύσεις από την πιο απλούστερη π.χ. αισθητήρια για την παρακολούθηση της θερμοκρασίας σε δωμάτιο μέχρι και την πιο σύνθετη αυτής, της απομακρυσμένης παρακολούθησης μιας μονάδας π.χ. ενός ρομπότ σε άλλο πλανήτη. Επιπλέον, με τα Smartphones να αποτελούν πλέον αναπόσπαστο κομμάτι της ζωής μας η διαχείριση και η παρακολούθηση τέτοιων συστημάτων έχει γίνει πιο εύκολη και προσιτή από ποτέ και συν τοις άλλης μας επιτρέπει την παραμετροποίηση και παρέμβαση οποιαδήποτε στιγμή και αν χρειαστεί.

Οι οικιακοί αυτοματισμοί είναι ένα άλλο πεδίο εφαρμογής του *Internet of Things* που ανήκει στην κατηγορία των εφαρμογών για καταναλωτές και είναι συνεχώς σε εξέλιξη με νέες πλατφόρμες, τεχνολογίες και Hardware να βγαίνουν κάθε τόσο. Ένας οικιακός αυτοματισμός μπορεί να προσφέρει οικονομία καθώς κάνει την παρακολούθηση σε κατανάλωση ενέργειας πιο εύκολη από ποτέ, προσφέρει ασφάλεια με την εγκατάσταση ενός δικτύου αισθητήρων και κάμερες και την ακόμη ευκολότερη εξ' αποστάσεως διαχείριση όλων των διασυνδεδεμένων έξυπνων οικιακών συσκευών.

Η υλοποίηση προσφέρει επίσης ευφυής αυτοματισμούς καθώς στην αλλαγή της κατάστασης κάποιων αντικειμένων ενεργοποιούνται άλλοι μηχανισμοί του συστήματος μέσω της υλοποίησης κάποιων κανόνων που θα εξηγήσουμε σε παρακάτω ενότητα. Το σύστημα αυτό με τις κατάλληλες τροποποιήσεις και χρήση αισθητήρων μπορεί να χρησιμοποιηθεί και για άλλες εφαρμογές αυτοματισμού στο πλαίσιο υλοποιήσεων των *Internet of Things* όπως για παράδειγμα, εφαρμογές τηλεϊατρικής, εκπαίδευσης ή διανομής εμπορευμάτων.

## 1.2 Κίνητρο για την Διεξαγωγή της Εργασίας

Τα τελευταία χρόνια, λόγω της ραγδαίας ανάπτυξης των συσκευών που συνδέονται στο διαδίκτυο και ελέγχονται μέσω αυτού, έχει οδηγήσει το *Internet of Things* να είναι αρκετά δημοφιλές και να αναπτύσσονται διάφορες εφαρμογές στο πεδίο αυτό. Όλο και περισσότερες επιχειρήσεις έχουν αρχίσει να υιοθετούν εφαρμογές που να πουλούν τεχνολογικά προϊόντα σχετικά με το Internet of Things για την χρήση στους καταναλωτές.

Παρ' όλα αυτά αν κάποιος χρήστης θέλει να στήσει ένα ολοκληρωμένο σύστημα αυτοματισμού από τα προϊόντα αυτά θα χρειαστεί να παραμετροποιήσει τις συσκευές ώστε να πληρούν τα διαφορετικά ζητούμενα των εφαρμογών.

Για αυτό τον λόγο σε αυτή την εργασία αναπτύσσεται ένα σύστημα αυτοματισμού που να υλοποιεί την αρχική ιδέα της στοχευμένης παράδοσης φαρμάκων σε πελάτες σε έναν Smart κόσμο με την χρήση ενός δικτύου μικροελεγκτών και αισθητήρων και τις εξ' αποστάσεως διαχείρισης και ελέγχου. Ένα άλλο ζητούμενο είναι πως μπορούμε να κρατήσουμε το κόστος χαμηλό και τι ευελιξία μας προσφέρει αυτό το σύστημα, ώστε να μπορεί ανά πάσα στιγμή να παραμετροποιηθεί και για άλλα πεδία εφαρμογών και χώρων.

### 1.3 Σκοπός και Στόχοι Εργασίας

Σκοπός της εργασίας είναι η δημιουργία ενός ευέλικτου συστήματος με χαμηλό κόστος για την απομακρυσμένη παράδοση χαπιών σε πελάτες με τη χρήση ενός αυτόνομου τροχοφόρου οχήματος μέσα σε έναν κλειστό και προδιαγραμμένο χώρο. Στόχος είναι η ανάπτυξη και εφαρμογή ενός λογισμικού για την εισαγωγή αρχικών παραμέτρων και δεδομένων, της διαδρομής, τους τελικούς αποδέκτες, σε πλατφόρμα ικανή να τρέξει σε οποιοδήποτε Smartphone ή και έναν ηλεκτρονικό υπολογιστή καθώς και την ανάπτυξη και υλοποίηση εφ' όσον το δίκτυο των συσκευών μας είναι συνδεδεμένο με το διαδίκτυο να παρακολουθεί την κατάσταση του οχήματος, αλλά και να αλληλοεπιδρά με τις συσκευές οποιαδήποτε στιγμή αυτό χρειαστεί.

Σε αυτή την εργασία θα ασχοληθούμε με στόχο την δυνατότητα στοχευμένης παράδοσης φαρμάκων σε ασθενείς χρησιμοποιώντας τη τεχνολογία IoT.

Η τεχνολογία IoT είναι σήμερα ένας τρόπος για να διευκολύνουμε και να βελτιώσουμε τα πράγματα. Έχει διεισδύσει σε όλες τις δυνατές διαστάσεις και τρόπους της καθημερινότητας στον εργασιακό χώρο και όχι μόνο.

Μία από αυτές τις καινοτομίες είναι και η χρήση ενός αυτόνομου οχήματος ακολουθώντας μια προκαθορισμένη διαδρομή χρησιμοποιώντας έναν Αλγόριθμο του τύπου παρακολούθησης γραμμής. Αυτή η αυτοματοποίηση φαίνεται να χρησιμοποιείται και με άλλους πιθανούς τρόπους και συνδυασμούς.

Η ουσιαστική διαφορά μας είναι ότι θα χρησιμοποιούμε ως καρδιά του συστήματος έναν μικροϋπολογιστή τύπου Raspberry Pi. Η ειδική παράδοση των φαρμάκων πάνω σε ένα αυτόνομο τροχοφόρο όχημα ακολουθώντας μια προκαθορισμένη διαδρομή. Η εφαρμογή αυτού του συστήματος περιλαμβάνει ένα ασύρματο δίκτυο, διάφορες συσκευές και αισθητήρες που αλληλοεπιδρούν κατά τρόπο που να ικανοποιεί τις απαιτήσεις μας.

Όλα αυτά θα δρουν αυτόνομα (σχεδιάστηκε από το μηδέν) και η επικοινωνία σε αυτό το δίκτυο επιτυγχάνεται με τη χρήση της γλώσσας προγραμματισμού Python.

Ο σκοπός αυτού του δικτύου είναι η επίβλεψη του αυτόνομου οχήματος, η εισαγωγή των διαφορετικών μεταβλητών απελευθέρωσης (της αντίστοιχης θυρίδας φαρμάκων που ισχύει για τον ασθενή) σε αλληλεπίδραση με τον αποδέκτη εάν αυτός είναι παρόν ή όχι. Η συνέχιση στον επόμενο σταθμό παράδοσης και την επιτυχή ολοκλήρωση της διαδρομής και επαναφορά στην βάση του.

**Ονομάζουμε αυτό το Project MEDIBERRY προερχόμενο από τα αρχικά Αυτόματος Διανομέας Φαρμάκων με χρήση Raspberry Pi.**



## 1.4 Δομή Εργασίας

Στην παρούσα εργασία στο κεφάλαιο 2<sup>ο</sup> γίνεται αναφορά στις τεχνολογίες του *Internet of Things* και των ασυρμάτων δικτύων αισθητήρων, οι οποίες αποτελούν την βάση για την υλοποίηση ενός σύγχρονου αυτοματισμού.

Στο κεφάλαιο 3<sup>ο</sup> περιγράφεται ο μικροεπεξεργαστής Raspberry Pi και τι δυνατότητες έχει με το εξωτερικό περιβάλλον και θα γίνει και μια εισαγωγή στον τρόπο χρήσης και παραμετροποίησης για το δικό μας σκοπό.

Στο κεφάλαιο 4<sup>ο</sup> περιγράφονται αναλυτικά τα προ απαιτούμενα στάδια υλοποίησης για κάθε σημείο αναφοράς και παράδοσης και παρουσίαση σεναρίων λειτουργίας του συστήματος με εικόνες.

Στο κεφάλαιο 5<sup>ο</sup> πως θα χρησιμοποιηθεί λεπτομερώς βήμα προς βήμα το λογισμικό Python για την διασύνδεση Raspberry Pi με το σύστημα του H/Y ή και φορητού.

Εν συνεχεία, στο κεφάλαιο 6<sup>ο</sup> ακολουθώντας όλα τα στάδια που προαναφέρθηκαν θα παρουσιαστεί στο σύνολο της η υλοποίηση για την δικιά μας περίπτωση και θα αναφερθούμε σε βασικά σενάρια κατάστασης του αυτοματισμού σε πραγματικό περιβάλλον βάζοντας τις αντίστοιχες παραμέτρους.

Στο 7<sup>ο</sup> κεφάλαιο περιγράφονται μελλοντικές επεκτάσεις του συστήματος, αναφορά σε μειονεκτήματα και συνολική αξιολόγηση της εργασίας που θα μπορούσε να έχει η παρούσα εφαρμογή.

Τέλος στο 8<sup>ο</sup> κεφάλαιο θα έχουμε αναφορά στα Συμπεράσματα.

Η παρούσα διατριβή ολοκληρώνετε με το 9<sup>ο</sup> κεφάλαιο το οποίο περιλαμβάνει την **Βιβλιογραφία**, **Πηγές** και **Τεχνικά Φυλλάδια** των χρησιμοποιημένων συσκευών και αισθητήρων καθώς και τον αναλυτικό κώδικα της εφαρμογής.

## Κεφάλαιο 2°

### 2.1. Τι είναι Τεχνολογίες του Internet (Internet of Things – IoT)

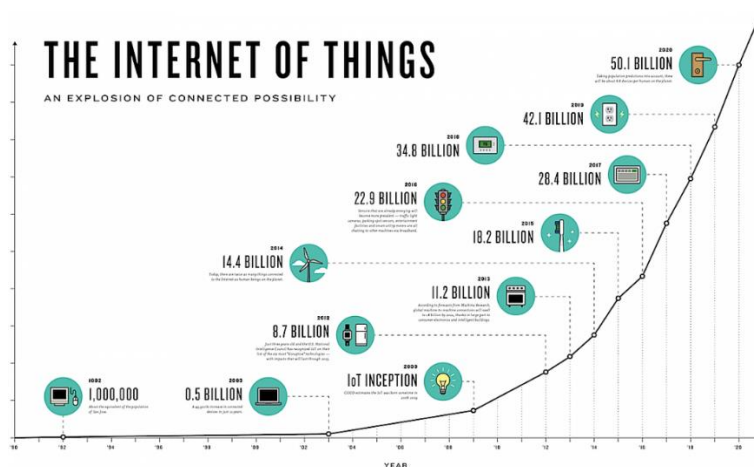
Το Internet of Things ή διαδίκτυο των πραγμάτων είναι μια νέα εφαρμογή τεχνολογιών που ανθίζει από τα τέλη της δεκαετίας του '90 και πιο συγκεκριμένα το 1999. Ο Kevin Ashton<sup>1</sup> αποτελεί μέλος της ομάδας που ανακάλυψε τον τρόπο σύνδεσης των συσκευών με το διαδίκτυο, κατά την παρουσίαση του επιτεύγματος αυτού αναφέρθηκε για πρώτη φορά ο όρος **the Internet of Things** και από τότε και ύστερα καθιερώθηκε η χρήση τους.

*Περιγράψτε με απλά λόγια ως οι συσκευές και περιφερειακά που χρησιμοποιούν ενσωματωμένους αισθητήρες για τη συλλογή και μετάδοση δεδομένων ή την ανάληψη κάποιας δράσης βάσει αυτών.*

Οι συσκευές αυτές είναι συνδεδεμένες σε τοπικό δίκτυο ή και το Internet και συλλέγουν, ανταλλάσσουν πληροφορίες και επικοινωνούν μεταξύ τους με απλές διακριτές μεθόδους και κατά τρόπο τέτοιο, ώστε να υλοποιούν τα προ απαιτούμενα.

*Η ιδέα πίσω από αυτό είναι η διασύνδεση όλων των ηλεκτρονικών συσκευών (αισθητήρων και μη) μεταξύ τους ή και ακόμα η διασύνδεση τους με το Internet.*

Η Cisco σε μια έρευνα που έκανε και βάση της διαφαινόμενης ανάπτυξης προέβλεψε ότι αρχές της επόμενης δεκαετίας (δεκαετία του '20) θα έχουμε κατ' εκτίμηση περίπου 50 δισεκατομμύρια συσκευές διασυνδεδεμένες με το Internet.

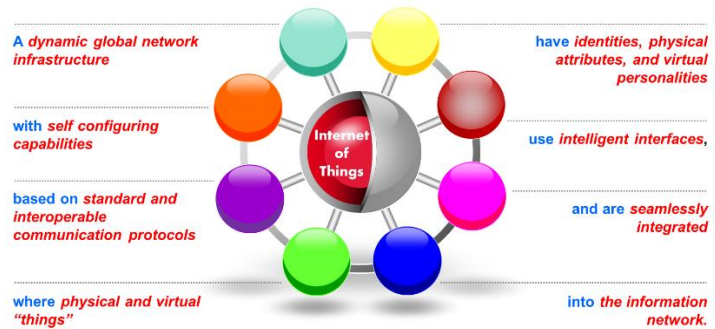
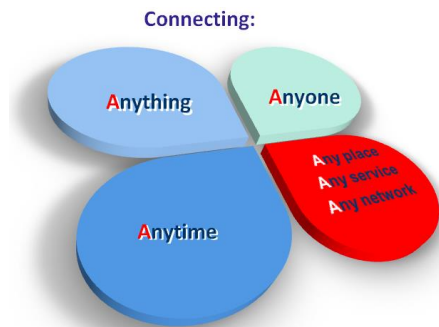


**Σχήμα 1:** Το '20 θα έχουμε τουλάχιστον 50 δις IoT συσκευές<sup>2</sup>

Και αυτές θα αυξηθούν ακόμη περισσότερο όσο θα υλοποιούμε όλο και περισσότερες εφαρμογές στο Internet. Εμάς σκοπός μας δεν είναι να κρίνουμε ή να ερευνήσουμε αν το ίδιο το Internet στην παρούσα φάση έχει την δυνατότητα να ανταπεξέλθει στο πλήθος των διαφαινόμενων εφαρμογών ούτε μας ενδιαφέρει αν θα μεταβεί ταχέως σε 5G ή και ακόμα 6G.

<sup>1</sup> [Wikipedia](#) αναφορά στον Kevin Ashton ερευνητή και πρωτοστάτη της Τεχνολογίας IoT.

<sup>2</sup> [What is the Internet of Things.](#)



Σχήμα 2α: Διασύνδεση των πάντων

Σχήμα 2β: Τρόποι και λόγοι διασύνδεσης

## 2.2. Τεχνολογίες που χρησιμοποιούμε στο Internet of Things

Οι τεχνολογίες που χρησιμοποιούμε αφορούν σε ποια πεδία θα θέλαμε να εφαρμόσουμε αυτή τη νέα τεχνολογία. Σε γενικές γραμμές βγαίνει το slogan ...

**Μπορούμε να συνδέσουμε τα πάντα με τα πάντα !!!**



Σχήμα 3: Πεδία εφαρμογών του Internet of Things

## 2.2 Οφέλη της χρήσης του Internet of Things (IoT)

Καμία τεχνολογική εφαρμογή δεν παράγει περισσότερα **DATA** από το **Internet of Things** και όταν πρόκειται για **BIG DATA** μιλάμε για τεράστιους όγκους δεδομένων.

*Ακούμε συνεχώς για το Internet of Things αλλά που είναι;*

*Πότε θα γίνει μέρος της καθημερινότητάς μας;*

*Που είναι οι αμέτρητες συσκευές-αισθητήρες που θα συνδέονται στο Internet;*

*Υπάρχει ζήτηση στην αγορά;*

*Έχει ωριμάσει η τεχνολογία;*

Πρέπει να ξεκαθαρίσουμε την κατάσταση παρουσιάζοντας μερικά χαρακτηριστικά παραδείγματα από βιομηχανίες που ενσωματώνουν τεχνολογίες IoT όχι στο μέλλον, αλλά στο παρόν.

### ***Ο χώρος του IoT χωρίζεται σε δύο βασικές κατηγορίες.***

**Στην πρώτη** έχουμε την καταναλωτική αγορά και τις διάφορες συνδεδεμένες συσκευές και **στη δεύτερη** τις συσκευές βιομηχανικού αυτοματισμού διασυνδεδεμένες μεταξύ τους.

Ωστόσο, ο χρήστης χρειάζεται τρανταχτά παραδείγματα βγαλμένα μέσα από τη ζωή του για να αντιληφθεί το πραγματικό όφελος του IoT.

Ένα από αυτά είναι η αυτοκινητοβιομηχανία η οποία από την πρώτη στιγμή υιοθέτησε τη συγκεκριμένη τεχνολογία. Μπορεί στην Ελλάδα να μην την έχουμε στην πράξη ακόμα, αλλά σε ΗΠΑ και Δ. Ευρώπη τα περισσότερα οχήματα πωλούνται με ενσωματωμένη συνδεσιμότητα και (συνδρομητικές) υπηρεσίες υποστήριξης προς τον οδηγό (τεράστιο το όφελος στον επαγγελματικό κλάδο). Έτσι, όχι μόνο μπορεί να γίνει πρόβλεψη βλαβών και αντικατάσταση εξαρτημάτων (της μπαταρίας για παράδειγμα ή του προγραμματισμού για service στο πλησιέστερο συνεργείο) αλλά και η εταιρεία υποστήριξης μπορεί να ειδοποιήσει τον οδηγό ότι το αυτοκίνητό του είναι ξεκλειδωτό και να του το κλειδώσει εξ' αποστάσεως<sup>3</sup>. Δεν μιλάμε ότι οι λύσεις σε περιβάλλον cloud προκαλούν τις εταιρίες λογισμικού στο να αξιοποιήσουν και να ενσωματώσουν όλο και περισσότερες ιδέες και εφαρμογές στα διασυνδεδεμένα οχήματα. Αλλά τους βάζουν συνεχώς να εξελίσσουν όλο και περισσότερες γραμμές κώδικα και να αναπτύξουν μικροελεκτές και αισθητήρια για την κάλυψη των αναγκών. Και τι θα δούμε ακόμα στο μέλλον!

### **2.3 Κίνδυνοι και προκλήσεις των IoT**

Οι κίνδυνοι και προκλήσεις του IoT αφορούν στο πόσο μπορούμε τεχνολογικά να συνδέσουμε τα πάντα με τα πάντα. Υπάρχουν κάποια πεδία στην ιατρική π.χ. που δεν μπορούν ακόμα να αξιοποιήσουν στην πράξη την τεχνολογία αυτή. Είτε γιατί δεν έχει ακόμα ωριμάσει η τεχνολογία υλοποίησης τους είτε δεν είναι ακόμα εφικτό να παραμετροποιηθούν πλήρως τα δεδομένα ώστε να προχωρήσουμε στην υλοποίησή τους και να έχουμε όφελος τελικά. Οι κίνδυνοι από την άλλη είναι κυρίως η ασφάλεια των IoT που είναι πολύ πιο πολύπλοκη από την ασφάλεια συνηθισμένων και τυπικών τερματικών συσκευών. Στο IoT έχουμε μια πλειάδα έξυπνων συσκευών, όπως αισθητήρες, διασυνδεδεμένες πλακέτες, υποπλακέτες, συσκευές γενικότερα που συνδέονται σε κάποιο τοπικό δίκτυο και από εκεί στο Internet ή και κατ' ευθείαν στο Internet. Μπορεί επίσης να υλοποιούν και κάποια διαδικτυακή εφαρμογή την οποία έχουμε σχεδιάσει από το μηδέν ή την παρέχει ο κατασκευαστής με ή όχι πρόσβαση σε υπηρεσίες cloud. Σε όλη την διαδρομή και ακόμη περισσότερα τα ίδια τα IoT έχουν ευπάθειες και τρωτά σημεία. Η ασφάλεια του αισθητηρίου ή της συσκευής δεν αποτελεί προτεραιότητα στο μυαλό κάθε σχεδιαστή εφαρμογής. Απλά υλοποιούν τις σχεδιαστικές ανάγκες αγνοώντας το θέμα ασφάλειας. Παράδειγμα θα μπορούσε να είναι ότι “εύκολα“ μπορούμε να

<sup>3</sup> Τα πραγματικά οφέλη του IoT, [Biztech.gr](http://Biztech.gr)

παραβιάσουμε κάποιες Webcam αφού με μια καλή αναζήτηση στο Internet μπορούμε να βρούμε τα συνθηματικά για την διαχειριστική πρόσβαση πάνω τους και έτσι να εκμεταλλευτούμε και να αποσπάσουμε τα δεδομένα τους. Το λογισμικό που τρέχουν πολλές φορές δεν ενημερώνετε ή δεν αξιοποιεί όπως θα έπρεπε κάποια επίπεδα ασφαλείας και έτσι παραμένει εκτεθειμένο και απροστάτευτο σε εξωτερικές κακόβουλες επιθέσεις. Από την άλλη και οι ίδιες εφαρμογές που ελέγχουν αυτές δεν ενημερώνονται ή ο κατασκευαστής/σχεδιαστής δεν ενδιαφέρετε να τις ενημερώσεις δίνοντας έτσι άλλο ένα σημείο τροφής σε επιθέσεις. Δεν αποτελεί κάποιο φανταστικό σενάριο άλλα δυστυχώς είναι η καθημερινότητα και το σύνθημα σε αυτές τις περιπτώσεις.

#### **2.4 Οφέλη της χρήσης του Internet of Things (IoT)**

Το IoT γενικότερα έχει τεράστια οφέλη. Δεν θα αναλύσουμε εδώ την ποικιλία των οφελών που έχει να προσφέρει αυτή η τεχνολογία για κάθε πτυχή της καθημερινότητας. Ας αναφερθεί ενδεικτικά σαν παράδειγμα η χρήση έξυπνων ετικετών σε φάρμακα. Έτσι μπορεί οποιοσδήποτε να παρακολουθήσει και να ελέγξει την διαδρομή από την παρασκευή μέχρι την κατανάλωση του φαρμάκου, κάτι το οποίο μέχρι σήμερα ήταν αδιανόητο χωρίς την νέα αυτή τεχνολογία. Πολλά είναι επίσης τα οικονομικά οφέλη που μπορούμε να έχουμε από την ορθή χρήση των IoT αισθητήρων και συσκευών.

Για παράδειγμα η εγκατάσταση ενός πλήρες ευφυούς σπιτιού μας επιτρέπει αν έχουμε σωστά παραμετροποίηση τα δεδομένα και τις αρχικές συνθήκες το σύστημα από μόνο του μόνο με επικοινωνία machine 2 machine (M2M) να ελέγχει το σπίτι ανοιγοκλείνοντας περιττά φώτα, ενεργοποίηση/απενεργοποίηση συναγερμού, ενεργοποίηση/απενεργοποίηση των κλιματιστικών, συμβάλλοντας ουσιαστικά στην ορθότερη ενεργειακή διαχείριση και εξοικονόμηση χρημάτων και ορθότερη χρήση πόρων.

#### **2.5 Προκλήσεις του IoT**

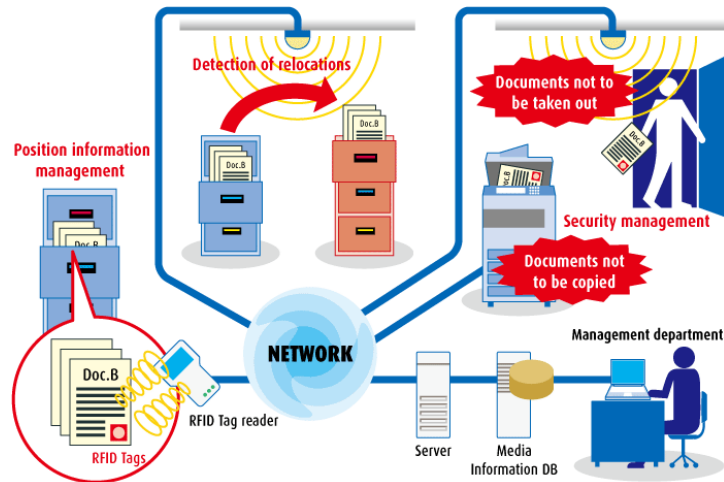
Αναμφίβολα η νέα τάση στον τομέα της τεχνολογίας είναι η αξιοποίηση της τεχνολογίας των IoT. Όλο και περισσότερες συσκευές και αισθητήρες διασυνδέονται είτε τοπικά είτε με τη χρήση του IP πρωτοκόλλου στο Internet. Αυτό εγείρει νέες προκλήσεις και κινδύνους που σπαζοκεφαλιάζουν οι τεχνικοί λογισμικού και δικτυακής ασφάλειας να προσπαθήσουν να ανιχνεύουν και να αποτρέψουν τρωτότητες ή εισόδους σε κακόβουλο λογισμικό. Οι πολλές φορές απλή φύση ή η σχεδίαση των αισθητήρων. Η πρόχειρη σχεδίαση και υλοποίηση στην ανάπτυξη του κώδικα, η πίεση χρόνου για την υλοποίηση της εφαρμογής, βοηθούν τους επίδοξους χάκερς να εισβάλλουν και να αντλούν ευαίσθητα δεδομένα ή να παίρνουν πλήρως τον έλεγχο των συσκευών αυτών. Άλλη πρόκληση είναι να σχεδιαστούν και να γίνουν άτρωτα τα συστήματα αυτά ως προς τις κυβερνοεπιθέσεις.

#### **2.6 Πεδίο εφαρμογών του Internet of Things (IoT)**

Τα πεδία εφαρμογών των IoT δεν είναι μόνο σε ένα πεδίο. Όλες οι καθημερινές λειτουργίες είτε είναι οι πιο απλές είτε είναι οι πιο σύνθετες ενδέχεται από πίσω να τρέχουν κάποιο IoT αυτοματισμό. Δεν νοείτε ότι χρειάζεται ιδιαίτερη μνεία αν με αυτό το τρόπο ξεπερνάμε κάποιους σκόπελους που σε προηγούμενα χρόνια ήταν αδιανόητο τεχνικά κάποια πράγματα να

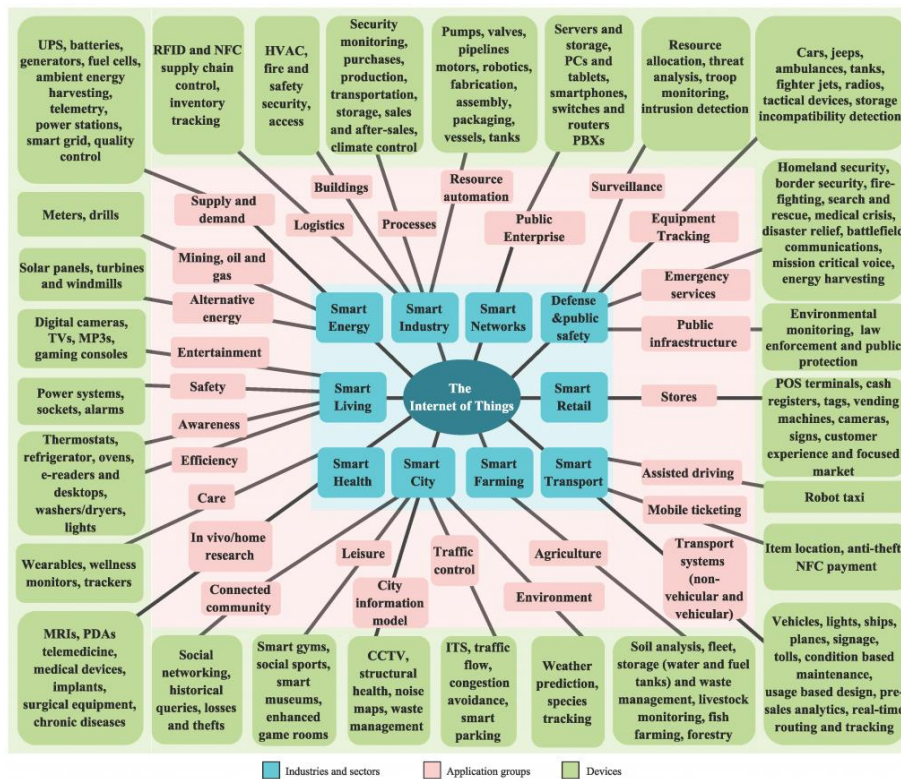


υλοποιηθούν, π.χ. η παρακολούθηση της αρτηριακής πίεσης από αυτοκόλλητο IoT αισθητήριο σε πραγματικό χρόνο, ή τη χρήση IoT αισθητηρίου για την παρακολούθηση της διαδρομής αντικειμένου στην γραμμή παραγωγής ή και την παρακολούθηση της ακριβής θέσης ενός διαβαθμισμένου εγγράφου μέσα σε μια υπηρεσία και την αποτροπή της κλοπής ή της αδικαιολόγητης εξόδου του ή της παραποίησης του.



Σχήμα 4: Smart Document management σε γραφεία

Παραστατικά στο παρακάτω σχήμα θα δούμε ενδεικτικά τα διάφορα πεδία εφαρμογών των IoT στην καθημερινότητα.



Σχήμα 5: Πεδία εφαρμογών IoT τεχνολογίας<sup>4</sup>

<sup>4</sup> Πεδία εφαρμογών των IoT τεχνολογιών, [πηγή](#)

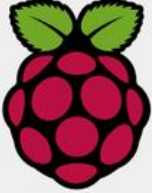
## Κεφάλαιο 3<sup>ο</sup>

Στο κεφάλαιο 3<sup>ο</sup> περιγράφεται η αναπτυξιακή πλακέτα με τον μικροεπεξεργαστή Raspberry Pi και τι δυνατότητες έχει με τον έξω κόσμο και θα γίνει και μια εισαγωγή στον τρόπο χρήσης και παραμετροποίησης για το δικό μας σκοπό.

### 3.1 Γνωριμία με το Raspberry Pi

*Το Raspberry Pi είναι ένας πλήρης υπολογιστής σε μέγεθος μιας πιστωτικής κάρτας.*

Οι διάφορες γενιές του διαφέρουν στην ταχύτητα του επεξεργαστή και στην μνήμη που διαθέτουνε.

	Raspberry Pi 3 Model B	Raspberry Pi Zero	Raspberry Pi 2 Model B	Raspberry Pi Model B+
Introduction Date	2/29/2016	11/25/2015	2/2/2015	7/14/2014
SoC	BCM2837	BCM2835	BCM2836	BCM2835
CPU	Quad Cortex A53 @ 1.2GHz	ARM11 @ 1GHz	Quad Cortex A7 @ 900MHz	ARM11 @ 700MHz
Instruction set	ARMv8-A	ARMv6	ARMv7-A	ARMv6
GPU	400MHz VideoCore IV	250MHz VideoCore IV	250MHz VideoCore IV	250MHz VideoCore IV
RAM	1GB SDRAM	512 MB SDRAM	1GB SDRAM	512MB SDRAM
Storage	micro-SD	micro-SD	micro-SD	micro-SD
Ethernet	10/100	none	10/100	10/100
Wireless	802.11n / Bluetooth 4.0	none	none	none
Video Output	HDMI / Composite	HDMI / Composite	HDMI / Composite	HDMI / Composite
Audio Output	HDMI / Headphone	HDMI	HDMI / Headphone	HDMI / Headphone
GPIO	40	40	40	40
Price	\$35	\$5	\$35	\$35

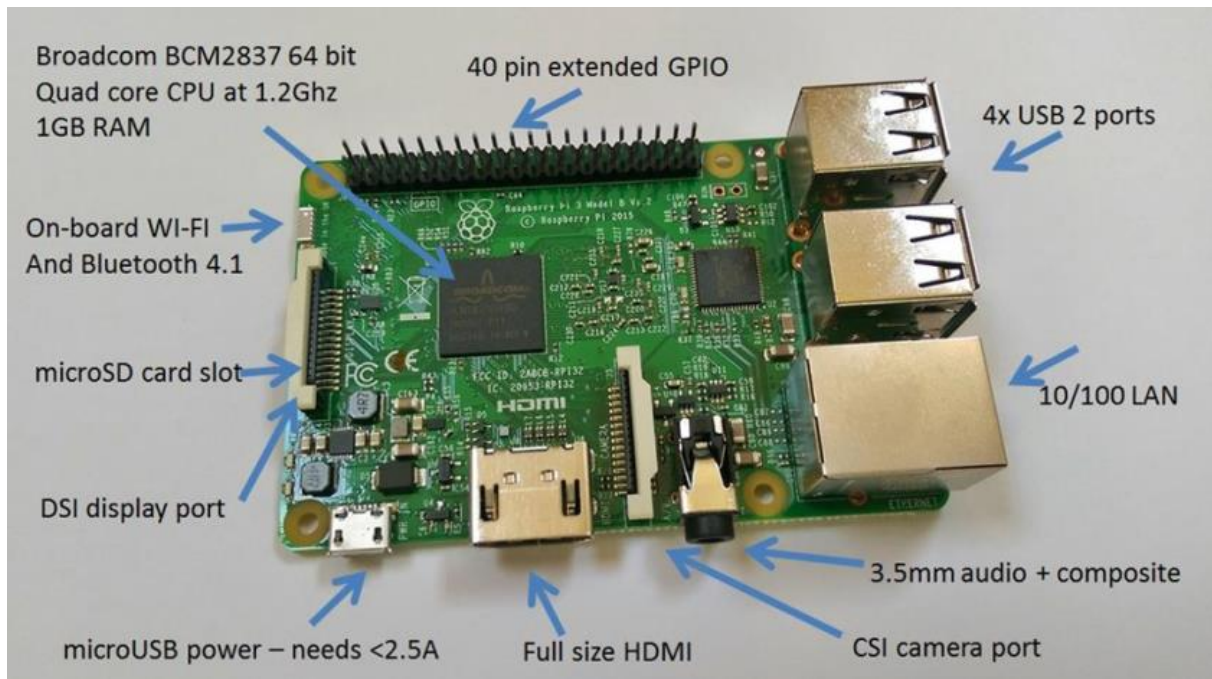
**Σχήμα 6:** Διάφορα μοντέλα Raspberry Pi και οι δυνατότητες τους

Το Raspberry Pi 3 (με το οποίο θα υλοποιήσουμε την παρούσα εργασία) διαθέτει έναν τετραπύρινο επεξεργαστή χρονισμένο στα 1.2GHz τεχνολογίας 64-Bit, με εσωτερική μνήμη 1GB RAM.

Για τη διασύνδεση με διάφορα περιφερειακά έχουμε στη διάθεση μας 4 θύρες USB 2.0 για σύνδεση με πληκτρολόγιο, ποντίκι και άλλα περιφερειακά, μια θύρα Ethernet ταχύτητας 10/100Mbps. Αυτό που το διακρίνει και το κάνει ιδιαίτερα δημοφιλή είναι η ενσωματωμένη δυνατότητα ασύρματης σύνδεσης τόσο σε Wifi όσο και με το πρωτόκολλο Bluetooth 4.1.

Επειδή είπαμε ότι πρόκειται για πλήρη υπολογιστή δεν θα πρέπει να ξεχάσουμε ότι διαθέτει έξοδο HDMI για την σύνδεση σε οθόνη H/Y (αν διαθέτει σύνδεση) ή ακόμη και στην σύνδεση σε τηλεοράσεις TFT/LCD. Όλο αυτό ολοκληρώνονται με τη δυνατότητα σύνδεσης ήχου στην

υποδοχή mini jack και τέλος microUSB υποδοχή για να την τροφοδοσία 5V με μέγιστο τα 2.5A.



**Σχήμα 7:** Raspberry Pi 3 model B

Για να λειτουργήσει το Pi 3 χρειάζεται και μια κάρτα microSD όπου θα εγκατασταθεί το λειτουργικό σύστημα γιατί από μόνο του δεν διαθέτει ενσωματωμένο λειτουργικό ούτε τρέχει με την ενεργοποίηση του κάποιο έτοιμο set ρουτίνων και κωδικών σε αντίθεση με το Arduino.

Ανάλογα με την χρήση του στις διάφορες υλοποιήσεις υπάρχουν έτοιμες διανομές λειτουργικού ανοιχτού κώδικα τις οποίες κυρίως χρησιμοποιούμε ή ο χρήστης τις διαμορφώνει, τροποποιεί ή και εξελίσσει κατάλληλα για την χρήση του.

Το Raspberry Pi 3 αποτελεί ιδανική λύση για χρήση στην εκπαίδευση και την εκμάθηση προγραμματισμού διότι δίνετε από τους προγραμματιστές διάφορες εκδόσεις λειτουργικού με ανοιχτό κώδικα για την χρήση ή περ εταίρου παραμετροποίηση.

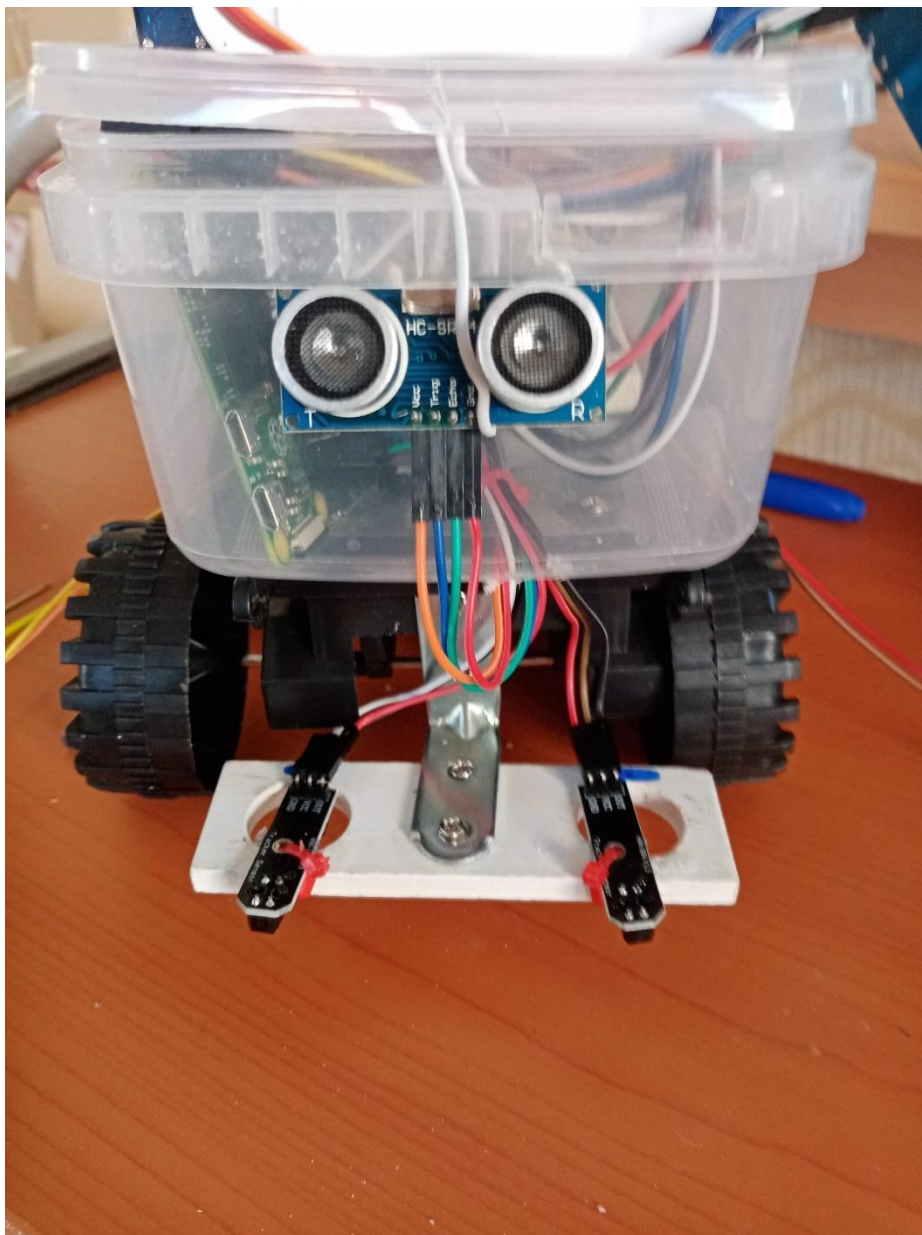


## Κεφάλαιο 4°

Στο κεφάλαιο 4° περιγράφονται αναλυτικά τα προ απαιτούμενα στάδια υλοποίησης για κάθε σημείο αναφοράς και παράδοσης του συστήματος με λεπτομερή επεξήγηση τόσο τεχνικά όσο και με εικόνες.

### 4.1 Follow the Line με Raspberry Pi

Όπως γνωρίζουμε το Raspberry Pi έχει αμέτρητες εφαρμογές χάρη στην πλατφόρμα ανάπτυξης που βασίζεται ο επεξεργαστής ARM που περιέχει. Μια από αυτές είναι η δυνατότητα με χρήση κατάλληλων κυκλωματικών στοιχείων της παρακολούθησης γραμμής είτε το έχουμε σταθερά κάπου τοποθετημένο είτε το έχουμε σε μια αυτοκινούμενη κατασκευή.



**Σχήμα 8:** Raspberry Pi δίτροχο αμαξίδιο με αισθητήρια για ανίχνευση γραμμής

## 4.2 Λειτουργία και τεχνική ακολούθησης γραμμής (Line Follower)

Η τεχνική που ακολουθείτε είναι ευρέως γνωστή σαν “**Τεχνική ακολούθησης Γραμμής - Line Follower**”. Η αρχή στην οποία στηρίζετε όλη αυτή η τεχνική είναι ότι ένα ζευγάρι αισθητηρίων (μπορεί να γίνει και στην μάντα των υπέρυθρων και στην μάντα των ακουστικών σημάτων) ανιχνεύει την όδευση μιας προκαθορισμένης διαδρομής. Αυτή η διαδρομή μπορεί να είναι αόρατη οπότε έχουμε εκ των προτέρων εισάγει στο λογισμικό την ακριβή διαδρομή με σημεία αναφοράς (waypoints). Ή δε άλλη περίπτωση είναι η εμφανή οπτική παρακολούθηση μιας γραμμής (μαύρη στην πλειονότητα και χρωματιστή όπου χρειαστεί).

Στη δικιά μας περίπτωση θα χρησιμοποιήσουμε ένα ζευγάρι υπέρυθρων αισθητηρίων. Κάθε αισθητήριο έχει ένα ζευγάρι από πομποδέκτη IR. Ο πομπός εκπέμπει μια δέσμη φωτός στη μάντα των υπέρυθρων το οποίο οδεύει προς τα μπρος. Εάν η δέσμη ανακλασθεί σε μια επιφάνεια θα υπάρξει επιστροφή και ο δέκτης που στη δικιά μας περίπτωση είναι μια φωτοδίοδος θα το αναγνωρίσει και θα δώσει σήμα στις επόμενες βαθμίδες του κυκλώματος.

*Στη μάντα των υπέρυθρων το φως ανακλάται από όλες τις επιφάνειες αλλά υπάρχουν 2 ακραίες καταστάσεις που μας βοηθούν ιδιαίτερα.*

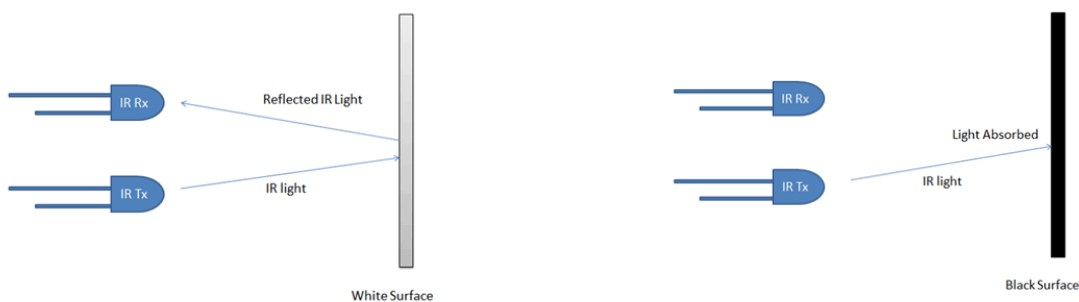
### 1<sup>η</sup> περίπτωση:

Εάν η προσπίπτουσα επιφάνεια είναι άσπρη η ανάκλαση θα είναι ολική και θα ανιχνεύει ο δέκτης ισχυρό IR σήμα (Σχήμα 9α).

### 2<sup>η</sup> περίπτωση:

Εάν η προσπίπτουσα επιφάνεια είναι μαύρη η ανάκλαση θα είναι μηδενική γιατί θα έχει υποστεί ολική απορρόφηση της IR ακτινοβολίας (Σχήμα 9β).

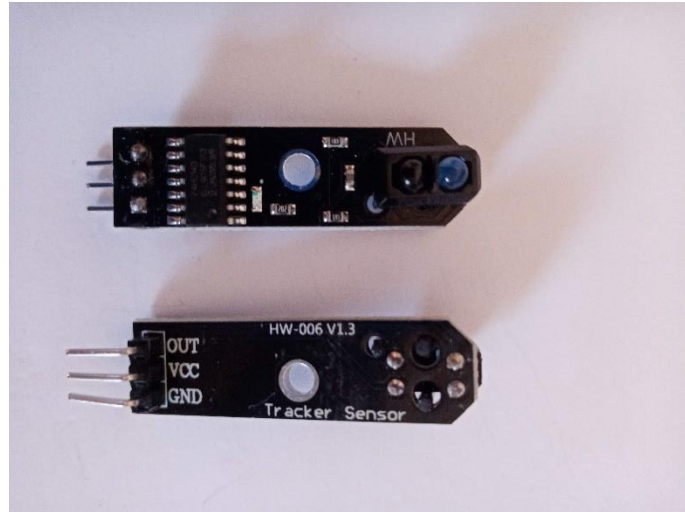
Παραστατικά στο παρακάτω σχήμα βλέπουμε τις 2 καταστάσεις.



**Σχήμα 9:** α) με άσπρη ανακλώμενη επιφάνεια β) σε μαύρη ανακλώμενη επιφάνεια

Η λογική συνέχεια είναι πως θα αποτυπωθούν αυτές οι δυο προαναφερόμενες καταστάσεις κυκλωματικά.

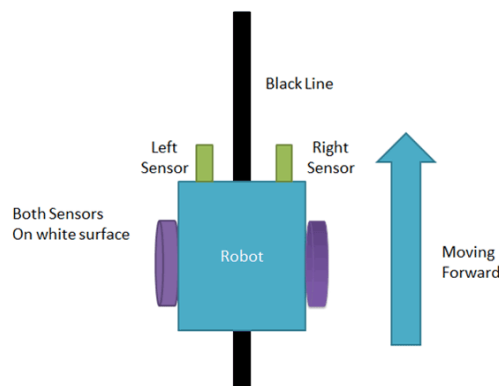
Στο παρακάτω σχήμα θα δούμε μια υλοποίηση σε πραγματική πλακέτα



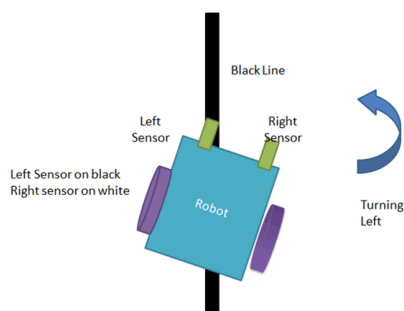
Σχήμα 10: IR αισθητήρες HW-006 για την υλοποίηση

Βάση αυτής της αρχής και το τρόπο λειτουργίας στην παρούσα υλοποίηση θα χρησιμοποιήσουμε **2 IR αισθητήρες**. Οι αισθητήρες είναι έτσι τοποθετημένοι σταθερά στο εμπρός τμήμα ώστε να ανιχνεύουν την πορεία του οχήματος. Το όχημα θα είναι εντός πορείας αν συνεχίζει να ανιχνεύει άσπρο από το ζευγάρι των αισθητηρίων. Σε περίπτωση που ανιχνεύσει μαύρο ένα από τα 2 αισθητήρια τότε θα υπάρχει πρόβλημα πορείας και θα πρέπει να επέλθει διόρθωση της πορείας.

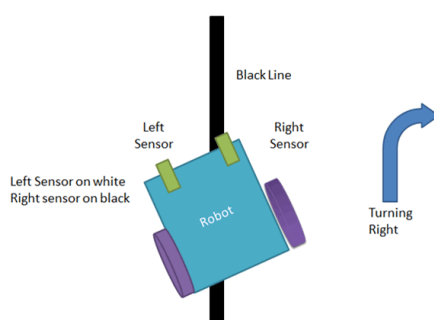
**Αν κανένας από τους 2 αισθητήρες IR δεν ανιχνεύει μαύρη γραμμή η εντολή από το λογισμικό είναι να προχωρήσει ευθεία.**



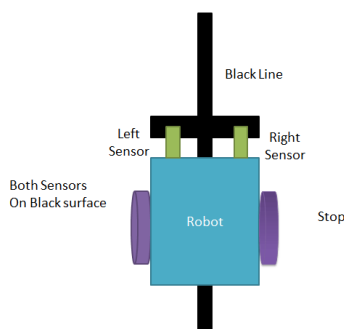
**Εάν το αριστερό αισθητήριο ανιχνεύσει μαύρο και το δεξί ανιχνεύει άσπρο τότε έχουμε πάτημα γραμμής από αριστερά και πρέπει να γίνει διόρθωση της πορείας προς τα αριστερά περιστρέφοντας το όχημα τόσο όσο να ανιχνεύσουν και τα 2 αισθητήρια πάλι άσπρο και να συνεχιστεί η ευθεία πορεία.**



**Εάν το δεξί αισθητήριο ανιχνεύσει μαύρο και το αριστερό ανιχνεύει άσπρο τότε έχουμε πάτημα γραμμής από δεξιά και πρέπει να γίνει διόρθωση της πορείας προς τα δεξιά περιστρέφοντας το όχημα τόσο όσο να ανιχνεύσουν και τα 2 αισθητήρια πάλι άσπρο και να συνεχιστεί η ευθεία πορεία.**



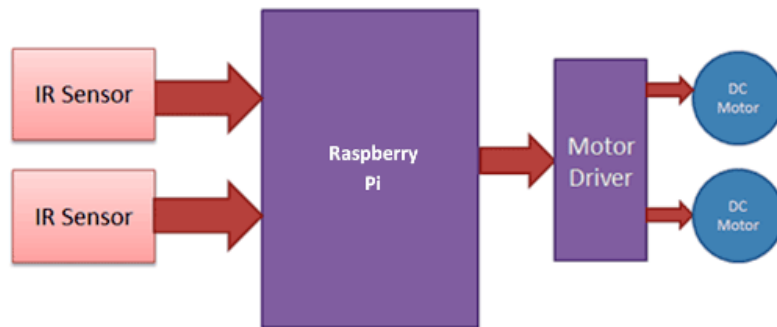
**Κωδικοποιούμε την κατάσταση για σταμάτημα του οχήματος, δίνοντας στον κώδικα το απαιτούμενο έναυσμα για δέσμη ενεργειών όταν και τα 2 αισθητήρια αντιληφθούν μαύρη οριζόντια γραμμή, αυτό το πετυχαίνουμε με τη χρήση συνεχόμενης μαύρης γραμμής όπως φαίνετε και στο παρακάτω σχήμα (μπορεί να γίνει και με διακεκομμένη γραμμή αρκεί να είναι λίγο μεγαλύτερη από το πεδίο "όρασης" των αισθητηρίων).**



Η όλη ανίχνευση γίνεται στα εξής στάδια.

Τα 2 αισθητήρια IR ανιχνεύουν την πορεία και το εάν το όχημα βρίσκεται στην προκαθορισμένη ευθεία. Εκείνα με τη σειρά τους οδηγούνται σε ένα Raspberry Pi το οποίο με τον κατάλληλο

λογισμικό έχει προγραμματιστεί έτσι ώστε να ελέγχει και κινεί από κάθε πλευρά την αντίστοιχη ρόδα που με τη σειρά της διορθώνει την πορεία όπως προαναφέραμε. Μια άλλη απαίτηση είναι οι κινητήρες αυτοί να έχουν την δυνατότητα να κινούνται και προς τα μπρος και προς τα πίσω για να διορθώνουν την πορεία.



**Σχήμα 11:** Μπλοκ διάγραμμα οδήγησης των τροχών από τους IR αισθητήρες

Για αυτή την απλοϊκή υλοποίηση θα χρειαστούμε τα παρακάτω υλικά:

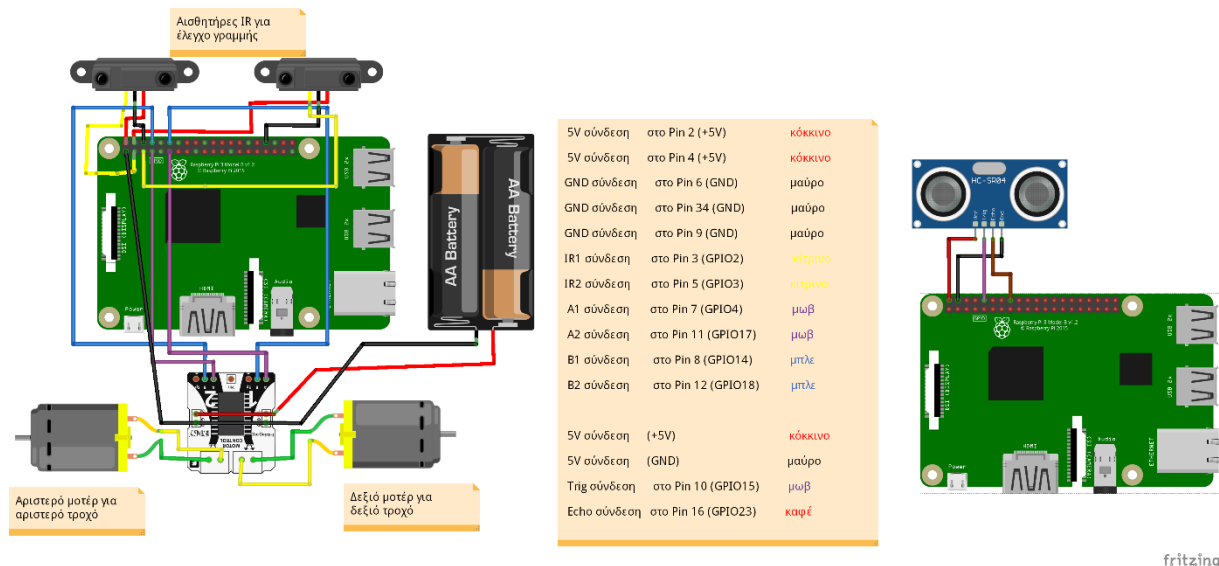
- *Raspberry Pi 3 (καρδιά του συστήματος μας)*
- *IR Sensors (2 αισθητήρια IR)*
- *DC Gear Motor (2 βηματικούς κινητήρες για τους τροχούς του οχήματος)*
- *L293D Motor Driver*
- *Τροχοφόρο όχημα (ένα σασί από χαρτόνι ή όχημα από παροπλισμένο τηλεκατευθυνόμενο (ευχαριστώ το γιο μου για αυτό).*
- *Power bank (Πηγή τροφοδοσίας του συστήματος και των υπολοίπων υλικών)*

#### 4.3 Κυκλωματικό διάγραμμα για την ακολούθηση γραμμής

Παρακάτω θα επικεντρωθούμε στο κυκλωματικό διάγραμμα της υλοποίησης δίνοντας έμφαση στον τρόπο. **Θα κάνουμε χρήση προγράμματος ανοικτού κώδικα για την υλοποίηση των κυκλωματικών διατάξεων βαθμίδα προς βαθμίδα εξηγώντας επ' ακριβώς τον τρόπο λειτουργίας τους και στο τέλος θα παρουσιάσουμε την όλη διάταξη συγκεντρωτικά.**

Το παρακάτω σχήμα είναι υλοποιημένο στο πρόγραμμα **Fritzing**<sup>5</sup> (ανοικτού κώδικα - πρότυπο σχέδιο υπάρχει ως σχέδιο με όνομα **follow the line 1.fzz**).

<sup>5</sup> Fritzing πρόγραμμα, <https://fritzing.org/>



**Σχήμα 12:** Follow the Line with Fritzing

Αναλυτικά όπως φαίνεται στο παραπάνω σχήμα στο κύκλωμα υπάρχουν 2 αισθητήρες IR και ένα ζευγάρι από DC μοτέρ που κινούν τους τροχούς ανεξάρτητα αριστερά και δεξιά. Το όλο κύκλωμα τροφοδοτείται από μια συστοιχία από μπαταρίες ή ένα power bank για να δώσει τα απαιτούμενα 5V που χρειάζεται (στο σχήμα φαίνεται ως μπαταρία).

Επειδή κάθε έκδοση του Raspberry Pi έχει διαφορετική διάταξη στον connector σύνδεσης (δες παρακάτω σχήμα) θα πρέπει για καλύτερη κατανόηση θα δείξουμε αναλυτικά την διάταξη τους.

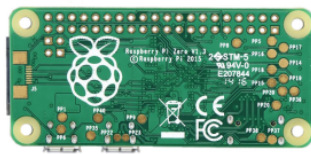
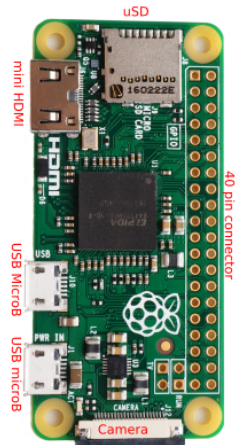
Raspberry Pi (Rev1)			Raspberry Pi (Rev 2)			Raspberry Pi B+, 2, 3 & Zero			Key
3V3	1	2	3V3	1	2	3V3	1	2	
GPIO0	3	4	GPIO2	3	4	GPIO2	3	4	GND
GPIO1	5	6	GPIO3	5	6	GPIO3	5	6	GND
GPIO4	7	8	GPIO4	7	8	GPIO4	7	8	GPIO14
GND	9	10	GND	9	10	GND	9	10	GPIO15
GPIO17	11	12	GPIO17	11	12	GPIO17	11	12	GPIO18
GPIO27	13	14	GPIO27	13	14	GPIO27	13	14	GND
GPIO22	15	16	GPIO22	15	16	GPIO22	15	16	GPIO23
3V3	17	18	3V3	17	18	3V3	17	18	GPIO24
GPIO10	19	20	GPIO10	19	20	GPIO10	19	20	GND
GPIO9	21	22	GPIO9	21	22	GPIO9	21	22	GPIO25
GPIO11	23	24	GPIO11	23	24	GPIO11	23	24	GPIO8
GND	25	26	GND	25	26	GND	25	26	GPIO7
						DNC	27	28	DNC
						GPIO5	29	30	GND
						GPIO6	31	32	GPIO12
						GPIO13	33	34	GND
						GPIO19	35	36	GPIO16
						GPIO26	37	38	GPIO20
						GND	39	40	GPIO21

**Σχήμα 13:** Διάταξη του GPIO Header στις διάφορες εκδόσεις Raspberry Pi

Στη δικιά μας υλοποίηση θα χρησιμοποιήσουμε το Raspberry Pi Zero v1.3 το οποίο είναι πιο οικονομικό (στα 15€) και έχει τις ίδιες δυνατότητες στο Header με το μεγάλο αδερφάκι του Raspberry Pi 3.



## Raspberry Pi Zero v1.3



Position: **Power** **Ground** **Control** **GPIO**  
**Wiring** **BCM** **Serial** **PWM** **Misc**  
 Different places use different pin numbers  
 GPIO, Wiring, and BCM have been included.

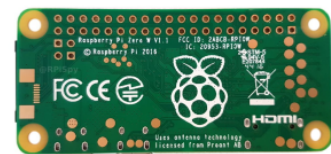
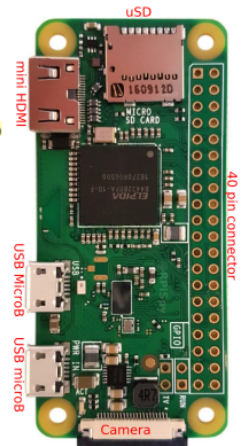
		3.3V	1	2	5V		
SDA	8	2	3	4	5V		
SCL	9	3	5	6	GND		
GPCLK0	4	7	4	7		14	15
		GND	9	10	15	16	TXD
spi1 CS1	17	0	17	11	12	18	1
		27	27	13	14	GND	
		22	3	22	15	16	23
		3.3V	17	18	24	5	24
MOSI	12	10	19	20	GND		
MISO	13	9	21	22	25	6	25
SCLK	14	11	23	24	8	10	SPI CS0
		GND	25	26	7	11	SPI CS1
ID_SD	30	0	DNC	27	28	DNC	1
GPCLK1	5	21	5	29	30	GND	
GPCLK2	6	22	6	31	32	12	26
PWM1	13	23	13	33	34	GND	
miso1	19	24	19	35	36	16	27
		26	25	26	37	20	28
		GND	39	40	21	29	21

PP1	USB
PP6	GND
PP8	3.3V
PP14	SD CLK
PP15	SD CMD
PP16	SD DAT0
PP17	SD DAT1
PP18	SD DAT2
PP19	SD CD
PP22	USB D+
PP23	USB D-

TV +	TV	Run	Run
TV -	TV	Run	Run

GPIO 0 and 1 are reserved - Do Not Connect  
 PAL or NTSC via composite video on TV pads  
 Run - temporarily connect pins to reset chip (or start chip after a shutdown)  
 Camera Connector (not on Zero 1.1 or 1.2) - 22pin, 0.5mm  
 Board Dimensions - 65mm x 30mm x 0.2mm  
 Mounting holes M2.5

## Raspberry Pi Zero W v1.1



**Processor - BCM2835**  
 ARM 11  
 Single Core  
 1GHz

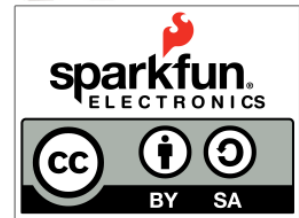
**Memory**  
 512MB RAM  
 uSD slot to run OS

**Video**  
 mini HDMI  
 PAL or NTSC via pads  
 HDMI capable of 1080p

**USB**  
 microB for power  
 microB for OTG

**Audio**  
 from HDMI port only

**Wireless**  
 2.4GHz  
 802.11n  
 Bluetooth 4.1/BLE



Σχήμα 14: Raspberry Pi Zero v1.3 vs v1.1

Οι ακροδέκτες 2 και 4 είναι για να τροφοδοτήσουν τα αισθητήρια IR με τα απαιτούμενα +5V (κόκκινοι αγωγοί). Στη συνέχεια γειώνουμε τα ίδια αισθητήρια IR στις θέσεις 6 και 34 (μαύρη σύνδεση). Με κίτρινο συνδέουμε τις εξόδους των αισθητήρων IR1 – pin 3 (GPIO2) και IR2 – pin 5 (GPIO3). Οι συνδέσεις αυτές θα μας χρειαστούν στην σύνταξη πιο κάτω του προγράμματος για την αναγνώριση της μαύρης γραμμής.

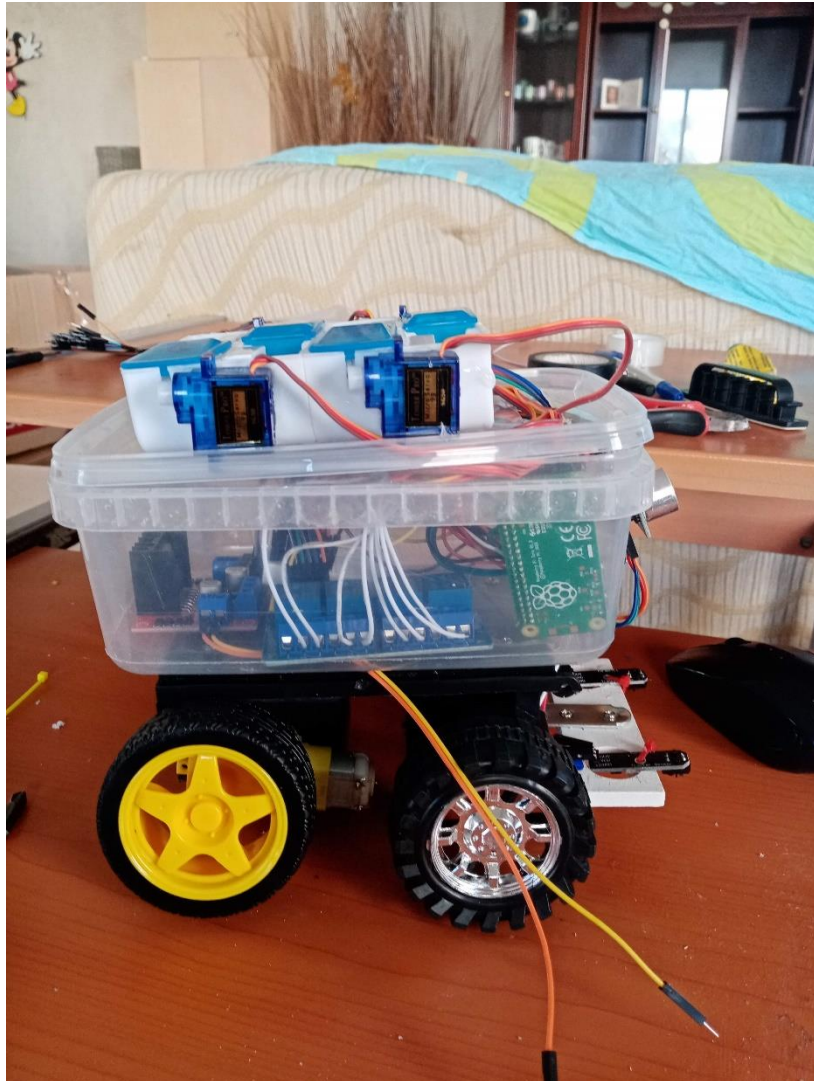
Για την οδήγηση των μοτέρ θα χρησιμοποιήσουμε 4 pins (μωβ το A και μπλε το B) αυτά συνδέονται με τα pins του Raspberry GPIO4 (pin7), GPIO14 (pin8), GPIO17 (pin11) και GPIO18 (pin12). Το πορτοκαλί και κίτρινο ζευγάρι καλωδίων αποτελούν το ζευγάρι σύνδεσης του εκάστοτε μοτέρ. Οι σύνδεση τους γίνεται από μια πλακέτα οδήγησης L293D που τροφοδοτείται αντίστοιχα από μια πηγή τροφοδοσίας ( μπαταρίες ή power bank). Η άνοδος θα τροφοδοτήσει την πλακέτα οδήγησης και η κάθοδος της τροφοδοσίας θα πάει στον κονέκτορα μας στη γείωση (pin 9) για να κλείσει το κύκλωμα.

❖ 5V σύνδεση	στο Pin 2	(+5V)	κόκκινο
❖ 5V σύνδεση	στο Pin 4	(+5V)	κόκκινο
❖ GND σύνδεση	στο Pin 6	(GND)	μαύρο
❖ GND σύνδεση	στο Pin 34	(GND)	μαύρο
❖ GND σύνδεση	στο Pin 9	(GND)	μαύρο
❖ IR1 σύνδεση	στο Pin 3	(GPIO2)	κίτρινο
❖ IR2 σύνδεση	στο Pin 5	(GPIO3)	κίτρινο

❖ A1 σύνδεση	στο Pin 7	(GPIO4)	μωβ
❖ A2 σύνδεση	στο Pin 11	(GPIO17)	μωβ
❖ B1 σύνδεση	στο Pin 8	(GPIO14)	μπλε
❖ B2 σύνδεση	στο Pin 12	(GPIO18)	μπλε

#### 4.4 Προγραμματισμός για την ρουτίνα για την ακολούθηση γραμμής

Μετά την επιτυχή σύνδεση της πλακέτας με τα υπόλοιπα κυκλωματικά στοιχεία η υλοποίηση θα είναι κάπως έτσι:



**Σχήμα 15:** Έτοιμο όχημα για την υλοποίηση Follow the Line

Το επόμενο βήμα είναι ο προγραμματισμός των ρουτίνων για να υλοποιήσουμε το Follow the Line Αλγόριθμο γραμμένο σε έκδοση Python.

Στην παρούσα διπλωματική δεν θα επεκταθούμε ούτε θα αναλύσουμε τους τρόπους προγραμματισμού σε περιβάλλον Python<sup>6</sup>. Θα θεωρηθεί ότι ο αναγνώστης θα ανατρέξει σε

<sup>6</sup> [Python](#) programming language



βιβλιογραφία και παραδείγματα του τρόπου προγραμματισμού σε αυτή τη γλώσσα προγραμματισμού.

Προχωράμε στην υλοποίηση.

Αρα πρώτα εισάγουμε από την βιβλιοθήκη για να ενεργοποιηθεί ο προγραμματισμός των ακροδεκτών της σύνδεσης GPIO στο Raspberry Pi. Για λόγους συντομίας θα κωδικοποιήσουμε τον όρο “GPIO” σε “IO” για να έχουμε μια πιο σαφή εικόνα κατά τον προγραμματισμό.

```
import RPi.GPIO as GPIO      # calling header file for GPIO's of PI
```

Για να αποφύγουμε λάθη στον προγραμματισμό των ακροδεκτών θα βάλουμε μια εντολή να αγνοήσει αυτό το πρόβλημα και να προχωρήσει με την εκτέλεση του προγράμματος εξόδου αν έχει.

```
GPIO.setwarnings (False)
```

Μπορούμε να αναφερθούμε όπως και στον προγραμματισμό σε Arduino είτε με τον αριθμό του ακροδέκτη είτε με την ιδιότητα που έχει το συγκεκριμένο pin (π.χ. PIN8 που είναι το GPIO14).

```
GPIO.setmode (GPIO.BOARD)
```

Πρέπει στη συνέχεια να ορίσουμε 2 ακροδέκτες ως εισοδοι σημάτων και 4 ακροδέκτες ως εξόδους. Τα πρώτα 2 θα είναι οι εισοδοι από τους αισθητήρες IR ενώ οι άλλοι 4 είναι ανά δυο για τον έλεγχο των τροχών αριστερά και δεξιά (Η διάταξη των μοτέρ 1 και 2 είναι όπως φαίνονται στο σχήμα).

```
GPIO.setup(3,GPIO.IN) #GPIO2      → IR1 έξοδος
```

```
GPIO.setup(5,GPIO.IN) #GPIO3      → IR2 έξοδος
```

```
GPIO.setup(7,GPIO.OUT) #GPIO4     → κινητήρας1 (δεξιά) θέση A
```

```
IO.setup(8,GPIO.OUT) #GPIO14     → κινητήρας1 (δεξιά) θέση B
```

```
GPIO.setup(11,GPIO.OUT) #GPIO17 → κινητήρας2 (αριστερά) θέση A
```

```
GPIO.setup(12,GPIO.OUT) #GPIO18 → κινητήρας2 (αριστερά) θέση B
```

Τα υπέρυθρα αισθητήρια δίνουν έξοδο «αληθής» όσο ανιχνεύουν άσπρη επιφάνεια.

Ως επακόλουθο όταν και οι δυο αισθητήρες μας δίνουν αληθής αποτέλεσμα μπορούμε να οδηγήσουμε το όχημα μας να προχωρήσει ευθεία.

Οι ρουτίνες είναι οι εξής:

```
if(GPIO.input(3)==True and GPIO.input(5)==True): #άσπρη επιφάνεια και ευθεία πορεία
```

```
GPIO.output (4,True)             #1A+
```

```
GPIO.output (14,False)          #1B-
```

```
GPIO.output (17,True)           #2A+
```

```
GPIO.output (18,False)         #2B-
```

Εάν τώρα βρεθεί ο αριστερός αισθητήρας να ανιχνεύει μαύρη γραμμή θα πρέπει να κάνουμε στροφή προς τα αριστερά. Θα πρέπει να σταματήσει ο δεξιός κινητήρας και να γυρίσει ο αριστερός έως ότου σταματήσει η ανίχνευση της μαύρης γραμμής.

```
elif(GPIO.input(3)==False and GPIO.input(5)==True): #στροφή προς τα αριστερά
```

```
GPIO.output(4,True)      #1A+  
GPIO.output(14,True)     #1B-  
GPIO.output(17,True)     #2A+  
GPIO.output(18,False)    #2B-
```

Σε αντιδιαστολή αν ανιχνεύσουμε δεξιά μαύρη γραμμή και αριστερά άσπρο θα πρέπει το όχημα μας να κάνει στροφή προς τα δεξιά. Ο κώδικας είναι:

```
elif(GPIO.input(3)==True and GPIO.input(5)==False): #στροφή προς τα δεξιά
```

```
GPIO.output(4,True)      #1A+  
GPIO.output(14,False)    #1B-  
GPIO.output(17,True)     #2A+  
GPIO.output(18,True)     #2B-
```

Αν και οι 2 αισθητήρες πατήσουν μαύρη γραμμή τότε το όχημα μας θα πρέπει να σταματήσει να κινείται γιατί θα είμαστε σε θέση που απαιτεί να καλέσουμε κάποια υπορουτίνα.

```
else: #σταμάτημα του οχήματος
```

```
GPIO.output(4,True)      #1A+  
GPIO.output(14,True)     #1B-  
GPIO.output(17,True)     #2A+  
GPIO.output(18,True)     #2B-
```

#### 4.5 Τροχοφόρο όχημα σε κίνηση με αλγόριθμο ανίχνευσης γραμμής

Ο ολοκληρωμένος κώδικας για το τροχοφόρο όχημα σε κίνηση με τον αλγόριθμο ανίχνευσης γραμμής για την ρουτίνα “**Follow the Black Line**” είναι ο παρακάτω:

```
import RPi.GPIO as GPIO      # calling header file for GPIO's of PI  
  
GPIO.setwarnings(False)  
  
GPIO.setmode (GPIO.BOARD)    # The Rpi uses the physical pin numbering  
  
# setup pins as Inputs and Outputs  
GPIO.setup(3,GPIO.IN) #GPIO2    IR1 έξοδος  
GPIO.setup(5,GPIO.IN) #GPIO3    IR2 έξοδος  
  
GPIO.setup(7,GPIO.OUT) #GPIO4    κινητήρας 1 (δεξιά) θέση A  
GPIO.setup(8,GPIO.OUT) #GPIO14   κινητήρας 1 (δεξιά) θέση B  
  
GPIO.setup(11,GPIO.OUT) #GPIO17  κινητήρας 2 (αριστερά) θέση A
```

```
GPIO.setup(12,GPIO.OUT) #GPIO18
```

κινητήρας 2 (αριστερά) θέση B

```
while 1:
```

```
    if(GPIO.input(3)==True and GPIO.input(5)==True): #άσπρη επιφάνεια και ευθεία πορεία
```

```
        GPIO.output(4,True)      #1A
        GPIO.output(14,False)     #1B
        GPIO.output(17,True)      #2A
        GPIO.output(18,False)     #2B
```

```
    elif(GPIO.input(3)==False and GPIO.input(5)==True): #στροφή προς τα αριστερά
```

```
        GPIO.output(4,True)      #1A
        GPIO.output(14,True)      #1B
        GPIO.output(17,True)      #2A
        GPIO.output(18,False)     #2B
```

```
    elif(GPIO.input(3)==True and GPIO.input(5)==False): #στροφή προς τα δεξιά
```

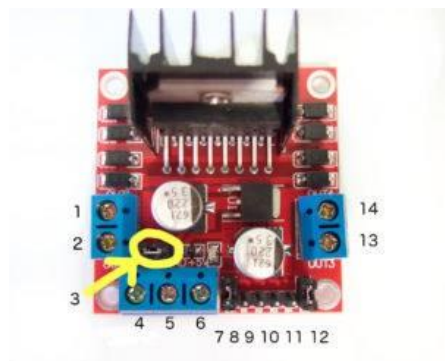
```
        GPIO.output(4,True)      #1A
        GPIO.output(14,False)     #1B
        GPIO.output(17,True)      #2A
        GPIO.output(18,True)      #2B
```

```
    else: #σταμάτημα του οχήματος
```

```
        GPIO.output(4,True)      #1A
        GPIO.output(14,True)      #1B
        GPIO.output(17,True)      #2A
        GPIO.output(18,True)      #2B
```

Ο πλήρης κώδικας για όλη την διάταξη που υλοποιήθηκε υπάρχει στην αρχή του κεφαλαίου 6.

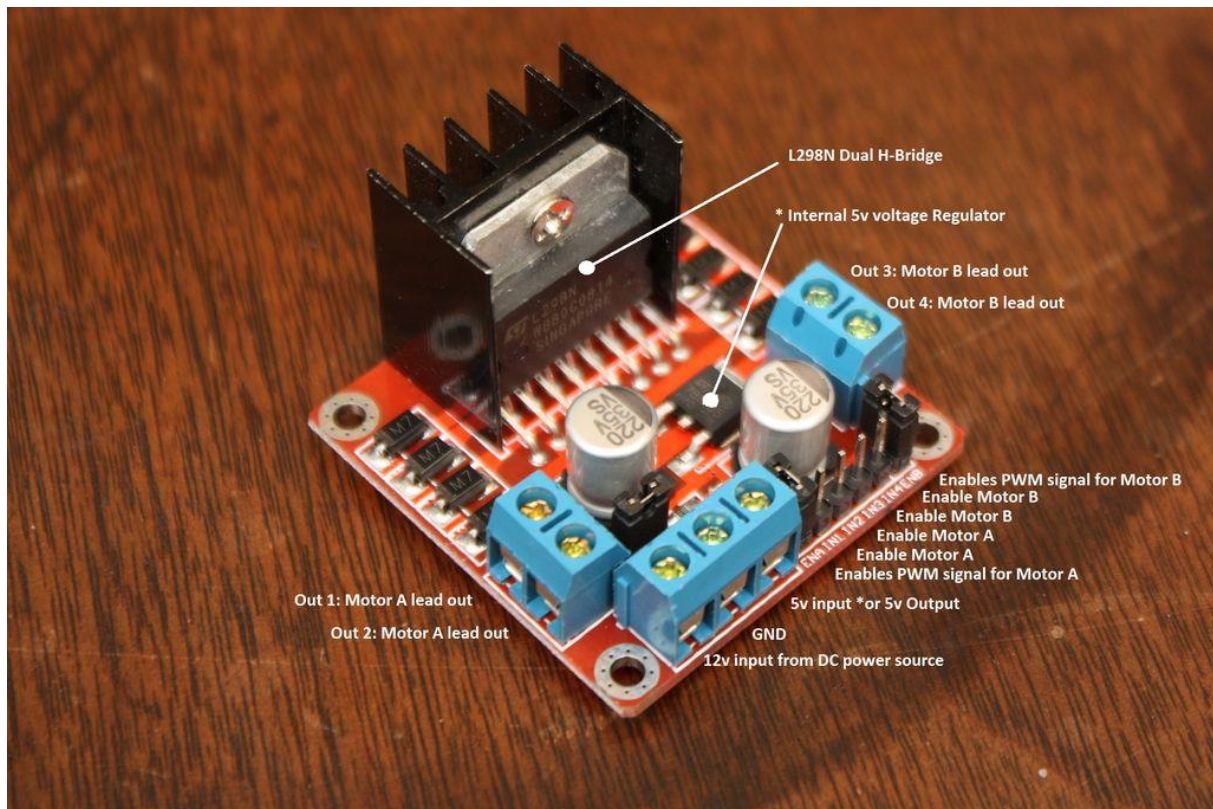
Η Συνδεσμολογία στην πλακέτα είναι η παρακάτω<sup>7</sup>:



<sup>7</sup> Official connection description for L298N Motor Driver

1. DC motor 1 “+” or stepper motor A+
2. DC motor 1 “-” or stepper motor A-
3. 12V jumper – **remove this if using a supply voltage greater than 12V DC**. This enables power to the onboard 5V regulator
4. Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5. GND
6. 5V output if 12V jumper in place, ideal for powering your Arduino (etc)
7. DC motor 1 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
8. IN1
9. IN2
10. IN3
11. IN4
12. DC motor 2 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
13. DC motor 2 “+” or stepper motor B+
14. DC motor 2 “-” or stepper motor B-

Και μια καλύτερη εικόνα του ίδιου



Σχήμα 16: L298 Motor Driver







Η αρχή λειτουργίας φαίνεται στο παρακάτω σχήμα.



Σχήμα 20: Αρχή λειτουργίας Ultrasonic Sensor

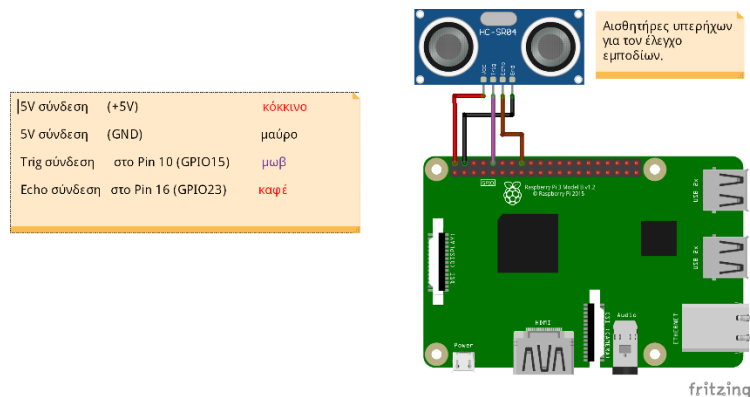
Ο αισθητήρας έχει 4 pins:

- |   |      |                              |         |
|---|------|------------------------------|---------|
| • | VCC  | 5V Supply                    | κόκκινο |
| • | TRIG | Trigger Input pin16 (GPIO23) | καφέ    |
| • | Echo | Echo Input pin10 (GPIO15)    | μωβ     |
| • | GND  | Ground                       | μαύρο   |

Η συνδεσμολογία απαιτεί τροφοδοσία στα 5V την οποία του παρέχουμε ήδη από το υπόλοιπο κύκλωμα σύνδεσης «Follow the Line». Το pin Echo δεν πρέπει να συνδεθεί κατευθείαν σε είσοδο Pin του Raspberry Pi γιατί αλλιώς θα κάψει την είσοδο. Για να μην το κάνει αυτό θα χρησιμοποιήσουμε έναν διαιρέτη τάσης ώστε να αποφύγουμε τα προαναφερόμενα.

Ένα απόσπασμα του κώδικα είναι το παρακάτω όπου ορίζουμε το pin16 ως είσοδο του παλμού Trigger από τον αισθητήρα υπερήχων και το pin10 ως είσοδο του παλμού Echo.

Παραστατικά βλέπουμε και το παρακάτω σχέδιο με τις συνδέσεις στο Fritzing (**ultrasonic HC-SR04 thesis.fzz**).



Σχήμα 21: Σύνδεση του αισθητήρα υπερήχων HC-SR04 στο Raspberry

Η συνδεσμολογία είναι συνέχεια του προηγούμενου κυκλώματος και δεν αποτελεί ξεχωριστή οντότητα όπως όλα τα σχέδια σε αυτή τη διπλωματική.

Αναδεικνύουμε κάθε φορά το επιμέρους κύκλωμα μαζί με τις επεξηγήσεις για να αποτυπώσουμε πως δουλεύει και πως είναι ο κώδικας για να λειτουργήσει.

Ο κώδικας σε Python είναι:

```

# We setup the pins as Inputs and Outputs for controlling the Ultrasonic Sensor for
# obstacle avoidance

GPIO.setup(10,GPIO.IN) #TRIGGER PULSE

GPIO.setup(16,GPIO.IN) #ECHO PULSE

sensor = DistanceSensor(echo=16, trigger=10, max_distance=1, threshold_distance=0.2)

led = LED(30)

sensor.when_in_range = led.on    #turns led on when sensor in range

sensor.when_out_of_range = led.off #turns led off when sensor out of range

pause()

```

Όταν ο αισθητήρας ανιχνεύσει «εμπόδιο» μπροστά του και είναι σε απόσταση **20cm** τότε ανάβει ένα κόκκινο Led το οποίο το έχουμε συνδέσει στην έξοδο στο pin **30**.

Μια διαφοροποίηση αυτού του προγράμματος θα ήταν αφού ανιχνευτεί το εμπόδιο (το οποίο εμείς απλά βλέπουμε όσο είναι μπροστά μας) να σταματήσει και το όχημα, εκτός από το οπτικό σημάδι που θα ανάβει, για όσο διάστημα βρίσκετε μπροστά του να τρέξει και μια ρουτίνα αποφυγής ή ακόμα αποφυγής και συνέχισης της διαδρομής. Αυτό έχει και τα υπέρ και τα κατά του.

#### ΥΠΕΡ:

- Το όχημα δεν μετακινείται όσο «βλέπει» μπροστά του εμπόδιο
- Δεν χρειάζεται επιπλέον πολύπλοκος κώδικας για την εξυπηρέτηση της ρουτίνας.

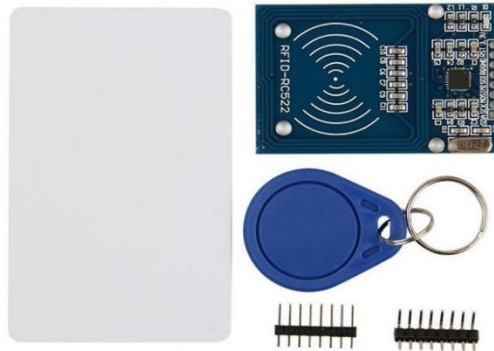
#### ΚΑΤΑ:

- Πρέπει να δοθεί έναυσμα από τον κώδικα για τη συνέχιση του προγράμματος

### 4.6 RFID

Οι ετικέτες RFID (**R**adio **F**requency **I**dentification) είναι μία πολλά υποσχόμενη και ταυτόχρονα ταχύτατα νέα αναπτυσσόμενη τεχνολογία. Τι είναι όμως οι ετικέτες RFID; Ουσιαστικά πρόκειται για μικροσκοπικά chips με ενσωματωμένη κεραία, τα οποία χρησιμοποιούνται ως αυτοκόλλητες ετικέτες σε διάφορα προϊόντα. Έτσι όταν βρεθούν κοντά σε μία συσκευή ανάγνωσης αυτής (RFID reader) εκπέμπουν τα δεδομένα που έχουν μέσα τους όπως, π.χ. την ταυτότητα του προϊόντος, την ημερομηνία κατασκευής ή παραγωγής ή κάποιο άλλο κωδικό που εμείς επιλέγουμε κ.τ.λ.





**Σχήμα 22:** RFID tags τύπου κάρτας και ετικέτας

Σήμερα για τη σήμανση των προϊόντων χρησιμοποιείται συνήθως για λόγους οικονομικούς και ευκολίας παγκοσμίως ο γραμμικός κώδικας (Barcode).

Γρήγορα όμως αυτός θα αντικατασταθεί από τον RFID, για τους εξής λόγους:

1. **Δεν απαιτεί αυστηρή οπτική επαφή όπως γίνεται με το barcode.** Για παράδειγμα στα εμπορικά καταστήματα (supermarket) κάθε προϊόν περνάει από ένα "μάτι" στο ταμείο για να "διαβαστεί" το barcode ώστε να αποτυπωθεί στην οθόνη του ταμείου το προϊόν και το κόστος του. Εάν δεν περάσει το εμπόρευμα περνάει απαρατήρητο και δεν θα χρεωθεί καν.
2. **Η ανάγνωση των στοιχείων γίνεται από απόσταση.** Η απόσταση του αναγνώστη είναι πολύ μεγαλύτερη από αυτή του barcode reader που είναι το πολύ κάποια εκατοστά.
3. **Αυξημένη ασφάλεια.** Τοποθετούνται σε σημεία που δεν γίνονται εύκολα αντιληπτά π.χ. μέσα στα ρούχα, στις ετικέτες ή στο εσωτερικό διαφόρων προϊόντων ή και συσκευασιών.
4. **Οι συσκευές ανάγνωσης των RFID tags μπορούν να τοποθετηθούν στην οροφή, στο δάπεδο, σε σημεία αναφοράς κτλ.** Οι συσκευές αυτές είναι μικροπομποί, οι οποίοι δημιουργούν ένα ηλεκτρομαγνητικό πεδίο. Έτσι μόλις περάσει από μέσα τους το προϊόν, ενεργοποιείται η ετικέτα και μεταδίδει τις πληροφορίες της πίσω στον μικροπομπού.
5. **Δεν χρειάζονται μπαταρίες για την λειτουργία τους.**

Θα μπορούσαμε να πούμε ότι η τεχνολογία αυτή έχει απεριόριστα πεδία εφαρμογών. Με την ανθρώπινη φαντασία μπορεί να βρούμε και νέα πεδία εφαρμογών. Τέτοια πεδία κάνουν ήδη εφαρμογή αυτής της τεχνολογίας και είναι κάποια από τα παρακάτω όπως:

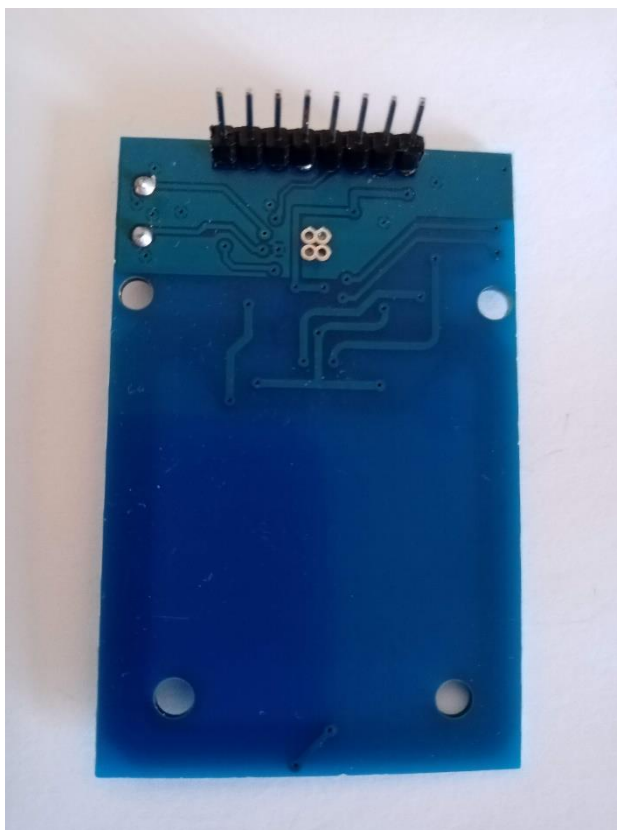
- *Σήμανση ζώων για εντοπισμό τους ή και την άντληση στοιχείων*
- *Αντικλεπτικά συστήματα θυρών*
- *Βιβλιοθήκες, παρακολούθηση κίνησης βιβλίων*

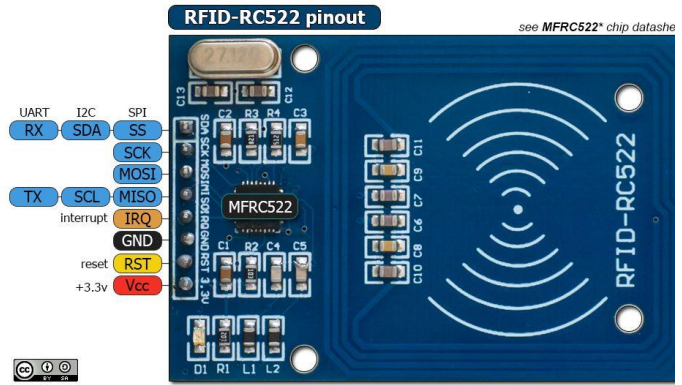
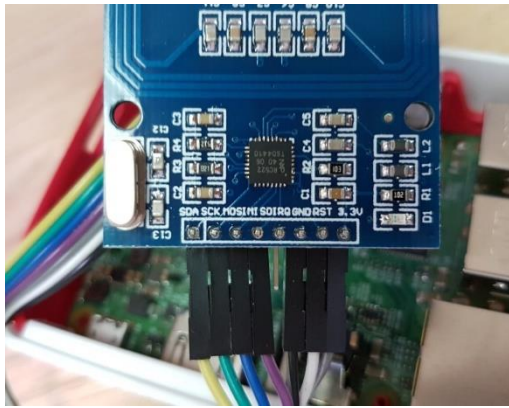
- *Αεροδρόμια, ανίχνευση μετακίνησης αποσκευών*
- *Κάρτες ανέπαφων συναλλαγών (πιστωτικές/χρεωστικές κάρτες)*

Εμείς θα κάνουμε χρήση αυτής της τεχνολογίας ως ακολούθως. Πάνω στο δικό μας RFID RC522 έχουμε 8 σημεία σύνδεσης. Αυτά είναι:

- ✓ **SDA** (Serial Data Signal),
- ✓ **SCK** (Serial Clock),
- ✓ **MOSI** (Master Out Slave In),
- ✓ **MISO** (Master In Slave Out),
- ✓ **IRQ** (Interrupt Request),
- ✓ **GND** (Ground Power),
- ✓ **RST** (Reset Circuit) και
- ✓ **3.3v** (3.3V Power In).

Θα πρέπει να συνδέσουμε όλους τους ακροδέκτες στο GPIO του Raspberry Pi εκτός από τη σύνδεση του **IRQ**. Η σύνδεση μπορεί να γίνει απευθείας καλωδιακά ή μέσα από συνδέσεις πάνω σε breadboard. Η διάταξη των ακροδεκτών είναι οφθαλμοφανές και δεν απαιτεί κάποιες άλλες επι πλέον συνδέσεις.

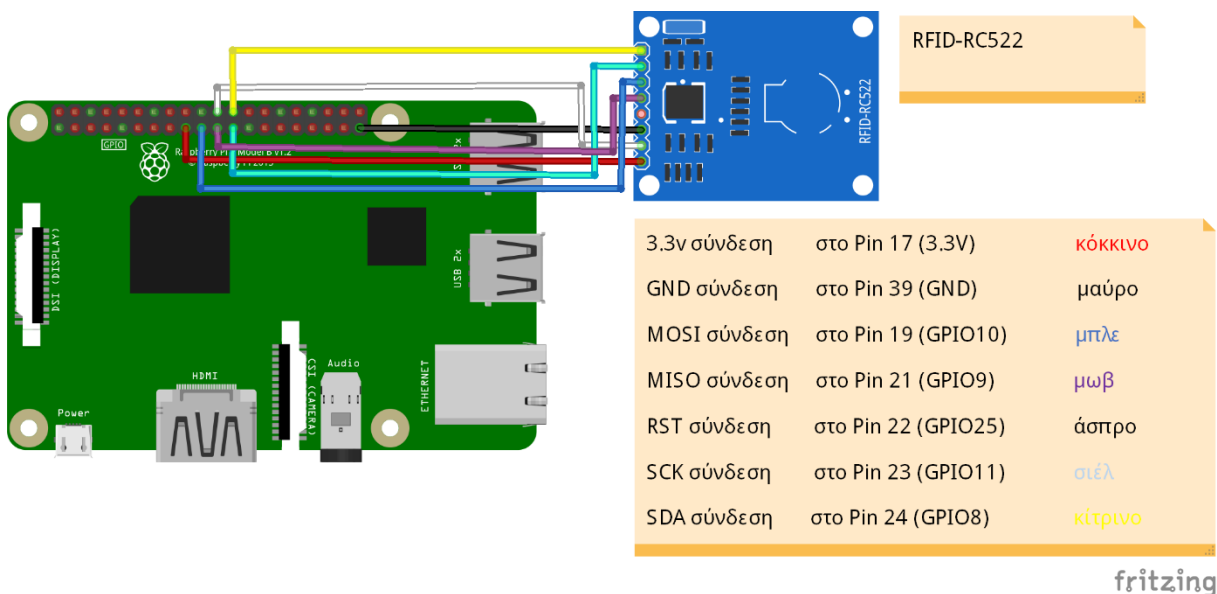




Σχήμα 23: Pin layout στο RFID-RC522

Στο παρακάτω σχήμα παραθέτουμε την κυκλωματική διασύνδεση με το Raspberry Pi μας.

Παρατηρούμε ότι έχουμε κάνει συνδέσεις σε ελεύθερες θέσεις στο GPIO μας, κάτι που θα μας βοηθήσει στο τέλος όταν ολοκληρώσουμε την διάταξη. Προς το παρόν οι τροφοδοσίες των 3.3V, 5V και GND είναι όλες ανεξάρτητα συνδεδεμένες, αλλά για λόγους ευκολίας μπορούμε στην ολική υλοποίηση να τις ομαδοποιήσουμε. Παραστατικά βλέπουμε και το παρακάτω σχέδιο με τις συνδέσεις στο Fritzing ([rfid-rc522 thesis.fzz](http://fritzing.org/projects/rfid-rc522-thesis-fzz)).



Σχήμα 24: Διασύνδεση του Raspberry Pi με το RFID-RC522

- ✓ 3.3v σύνδεση στο Pin 17 (3.3V) κόκκινο
- ✓ GND σύνδεση στο Pin 39 (GND) μαύρο
- ✓ MOSI σύνδεση στο Pin 19 (GPIO10) μπλε
- ✓ MISO σύνδεση στο Pin 21 (GPIO9) μωβ
- ✓ RST σύνδεση στο Pin 22 (GPIO25) άσπρο
- ✓ SCK σύνδεση στο Pin 23 (GPIO11) σιέλ

- ✓ SDA σύνδεση
- ✓ IRQ σύνδεση

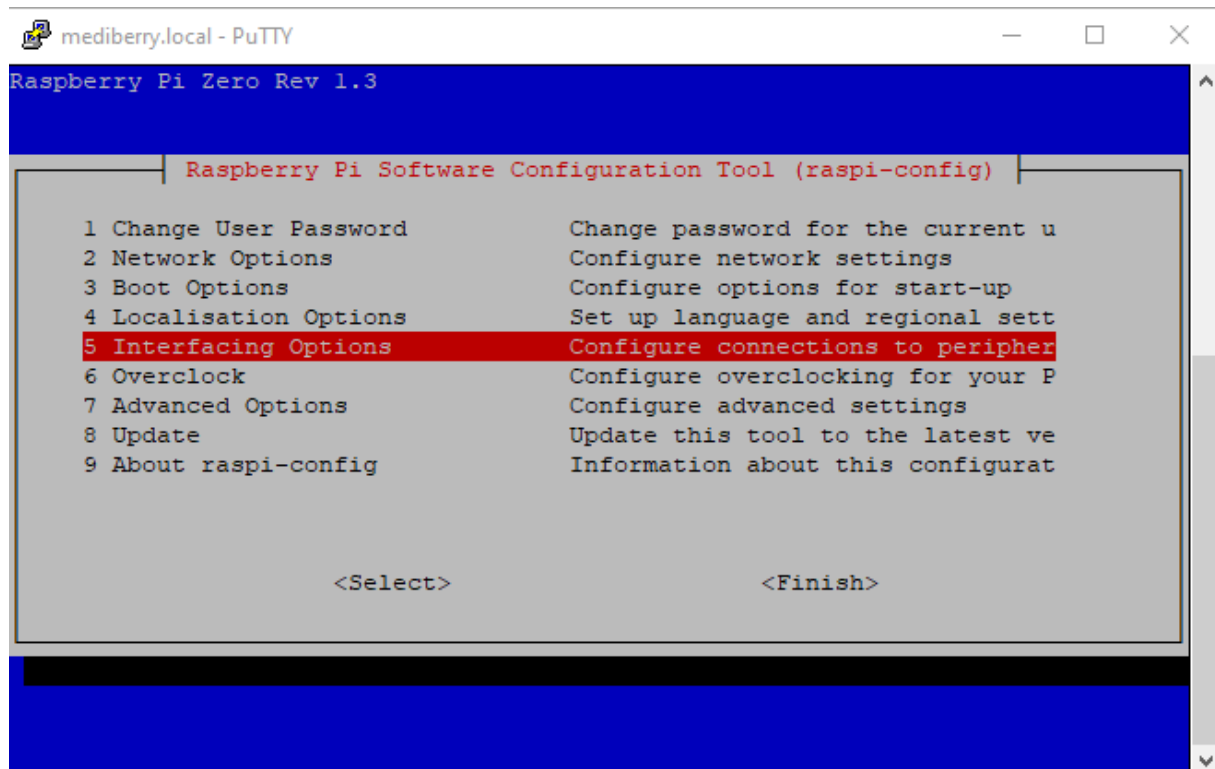
στο **Pin 24 (GPIO8)**  
ασύνδετο

κίτρινο

Έχοντας την διασύνδεση μας έρθει και εκατέρωθεν μπορούμε να αναπτύξουμε τον κώδικα σε python για την αναγνώριση.

Τα προ απαιτούμενα είναι να έχουμε ενεργοποιήσει το SPI Interface στο Raspberry Pi από τις ρυθμίσεις του raspi-config.

### Sudo raspi-config



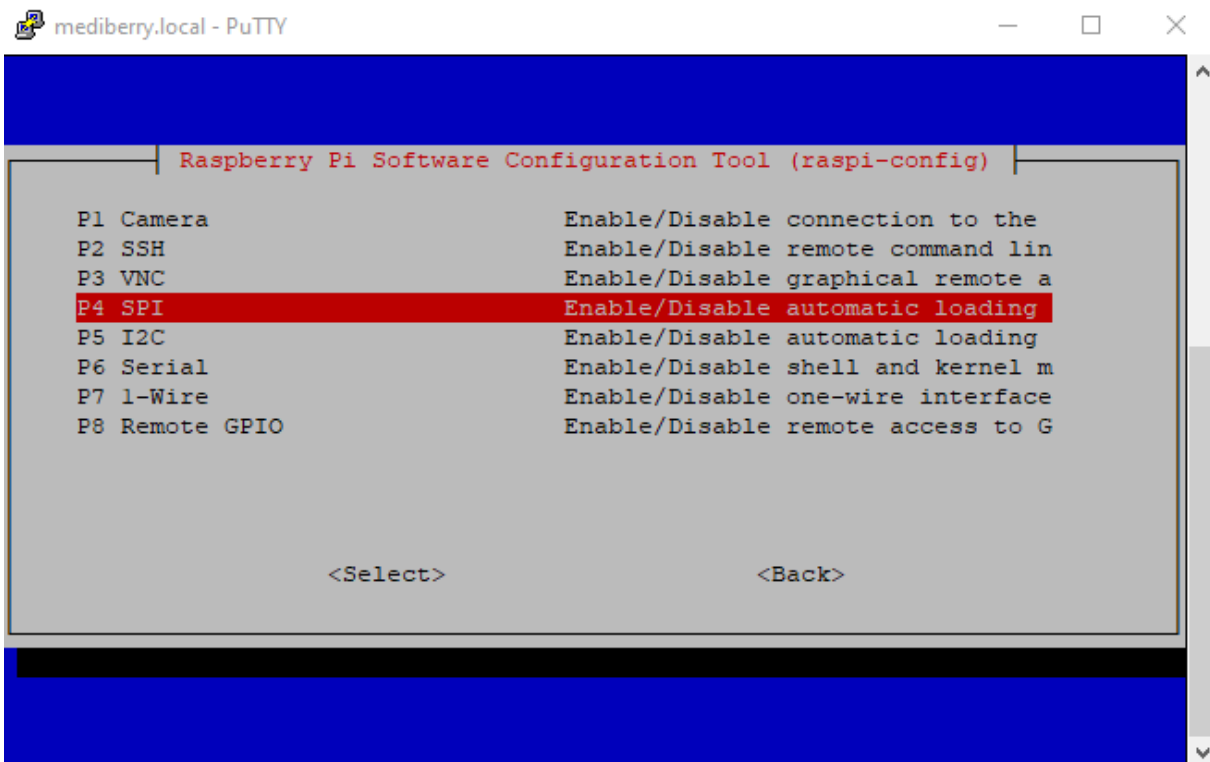
```
mediberry.local - PuTTY
Raspberry Pi Zero Rev 1.3

Raspberry Pi Software Configuration Tool (raspi-config)

1 Change User Password      Change password for the current u
2 Network Options           Configure network settings
3 Boot Options              Configure options for start-up
4 Localisation Options      Set up language and regional sett
5 Interfacing Options       Configure connections to peripher
6 Overclock                 Configure overclocking for your P
7 Advanced Options          Configure advanced settings
8 Update                    Update this tool to the latest ve
9 About raspi-config        Information about this configurat

<Select>                    <Finish>
```

### Μετά Interfacing Options



και enable SPI Interface

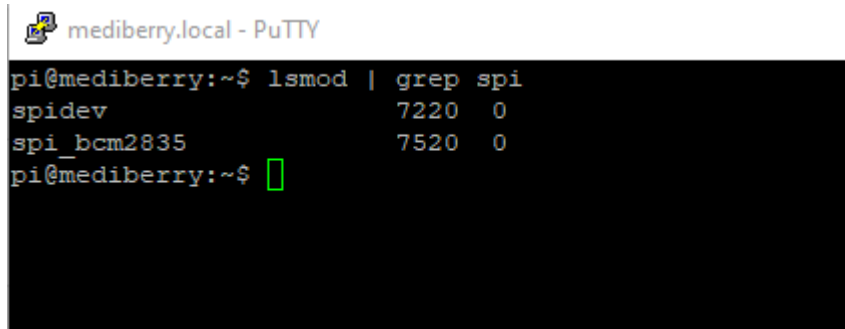


Αφού ολοκληρωθεί η διαδικασία της ενεργοποίησης θα σώσουμε την κατάσταση και θα προβούμε σε επανεκκίνηση για να ενεργοποιηθεί πλήρως.

**Sudo reboot**

Αφού γίνει αυτό θα πρέπει να ελέγξουμε αν έχει ενεργοποιηθεί το SPI Interface.

### lsmod | grep spi



```
mediberry.local - PuTTY
pi@mediberry:~$ lsmod | grep spi
spidev                7220 0
spi_bcm2835           7520 0
pi@mediberry:~$
```

Και βλέπουμε ότι είναι ενεργό το **spi\_bcm2835 (Raspberry Pi)**.

Στην δικιά μας υλοποίηση ζητούμενο είναι να τα εξής στάδια:

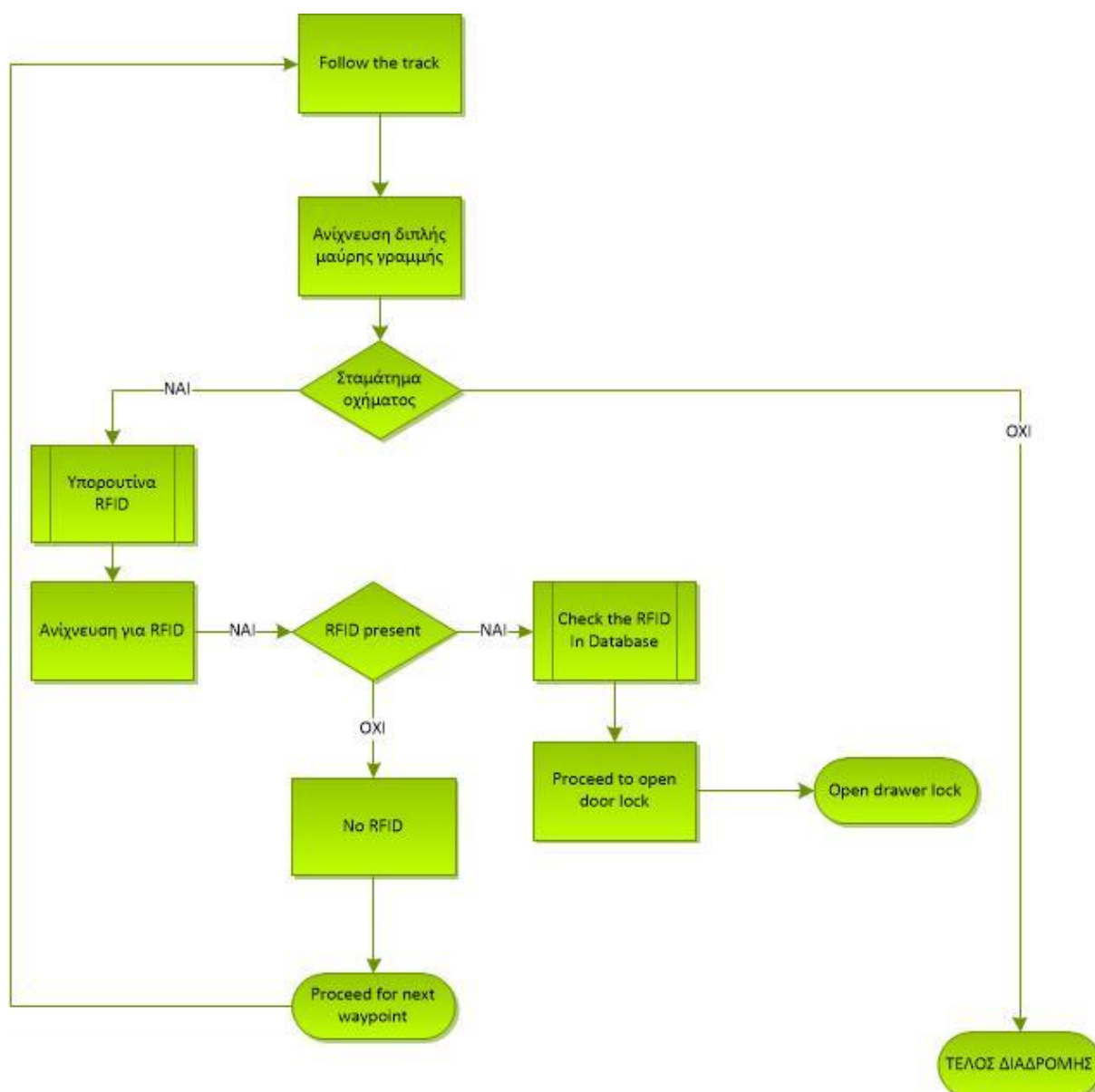
- 1) Αφού έχει έλθει σε ακινησία το τροχοφόρο όχημα ενεργοποιείτε ρουτίνα στον κώδικα του για να δουλέψει ο RFID αναγνώστης.
- 2) Ο RFID αναγνώστης είναι σε «**αναμονή**» για τυχόν σήμα από RFID πομπό τύπου κάρτας.
- 3) Αφού ταυτοποιηθεί ότι ανήκει σε «**γνωστό**» κλειδιά θα προχωρήσει ο κώδικας και θα τσεκάρει αν είναι στην σωστή θέση (π.χ. σταμάτησε το όχημα στη θέση 1 και είναι το RFID της θέσης 1 τότε προχωράει σαν αληθής η ρουτίνα, αλλιώς βγάζει λάθος).
- 4) Αν είναι στη σωστή θέση
- 5) Είναι και το αντίστοιχο κλειδί της θέσης;
- 6) Προχωράμε στον κώδικα δίνοντας εντολή στο αντίστοιχο πορτάκι (Θα εξηγήσουμε παρακάτω τι είναι το πορτάκι).
- 7) Εμφανίζουμε ενδεικτική λυχνία ότι είναι λάθος θέση ή κλειδί.
- 8) Αν δεν είναι το αντίστοιχο κλειδί της θέσης προχωράμε στον κώδικα και δίνουμε εντολή για συνέχιση της κίνησης του οχήματος.

Μετά την λεκτική περιγραφή ακολουθεί ένα διάγραμμα ροής σε Microsoft Visio για την υλοποίηση του.

***Προσοχή δίνει μόνο σημασία στο κομμάτι το πως δουλεύει η ανίχνευση του σήματος RFID και τι ρουτίνες δουλεύουν και τι στάδια υπάρχουν.***

**Το Πλήρες διάγραμμα ροής της όλης διάταξης θα αναλυθεί και θα παρουσιαστεί στο κεφάλαιο 7.**





Μετά την λεκτική περιγραφή ακολουθεί το πρόγραμμα σε κώδικα python για την υλοποίηση του.

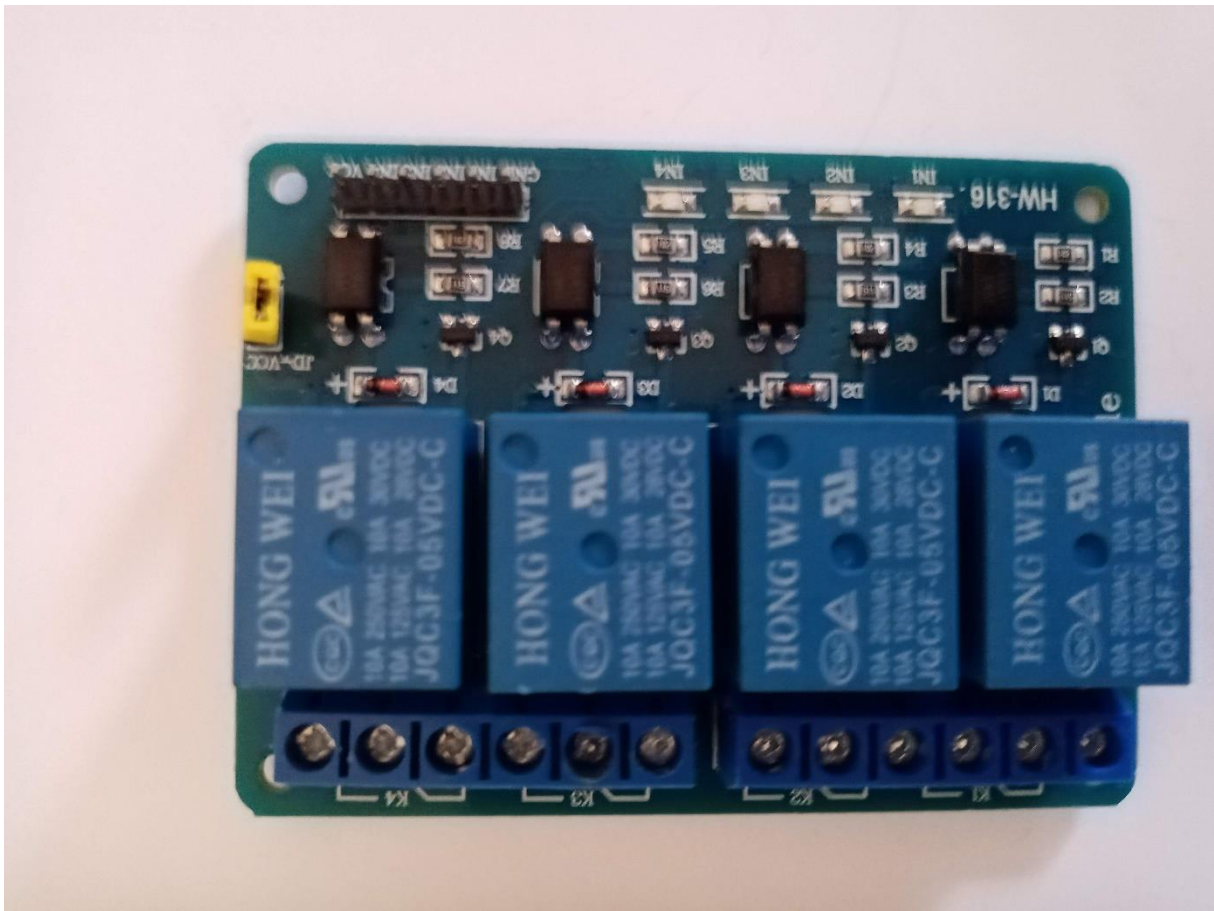
### Python script RFID

#### 4.7 Relays

Το επόμενο στάδιο είναι το πως θα ανοίξουμε και θα κλείσουμε τα πορτάκια στα κελιά στο τροχοφόρο όχημα.

Για αυτό το σκοπό θα χρησιμοποιήσουμε ένα relay 4 καναλιών ειδικά σχεδιασμένο για Arduino/Raspberry τύπου HW-316 στα 5V για να κάνουμε την οδήγηση των επι μέρους κελιών φαρμάκων (medicine cell). Κάθε αριθμημένο κελί (από το 1 μέχρι το 4) περιέχει μέσα του το αντίστοιχο φάρμακο.

Στην προκειμένη περίπτωση πρέπει να είμαστε σε θέση να οδηγήσουμε κάθε κελί ξεχωριστά από την αντίστοιχη ρουτίνα του προγράμματος.

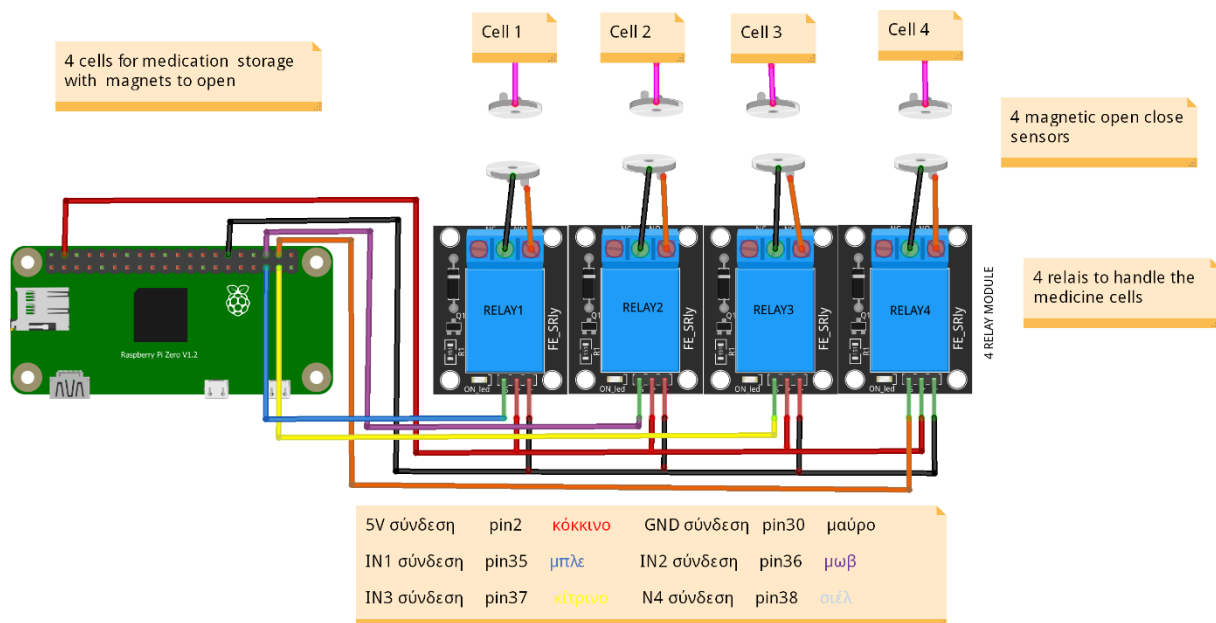


Σχήμα 25: Relay 4 καναλιών

Η διαδικασία είναι η εξής:

Αφού γίνει η επιβεβαίωση από το Rfid (δες πιο πάνω για την περιγραφή) τότε θα δοθεί από το πρόγραμμα η εντολή στα άκρα του Relays και συγκεκριμένα στις εισόδους IN1 έως IN4 ανάλογα με την περίπτωση για ενεργοποίηση εξόδου. Αυτό με τη σειρά του θα ενεργοποιήσει την αντίστοιχη έξοδο στο relays K1 έως K4. Οι εξοδοί των relays είναι συνδεδεμένα με μαγνητικούς αισθητήρες.

Παρακάτω βλέπουμε το σχηματικό σχεδιασμένο στο Fritzing (4 relais board thesis.fzz)



fritzing

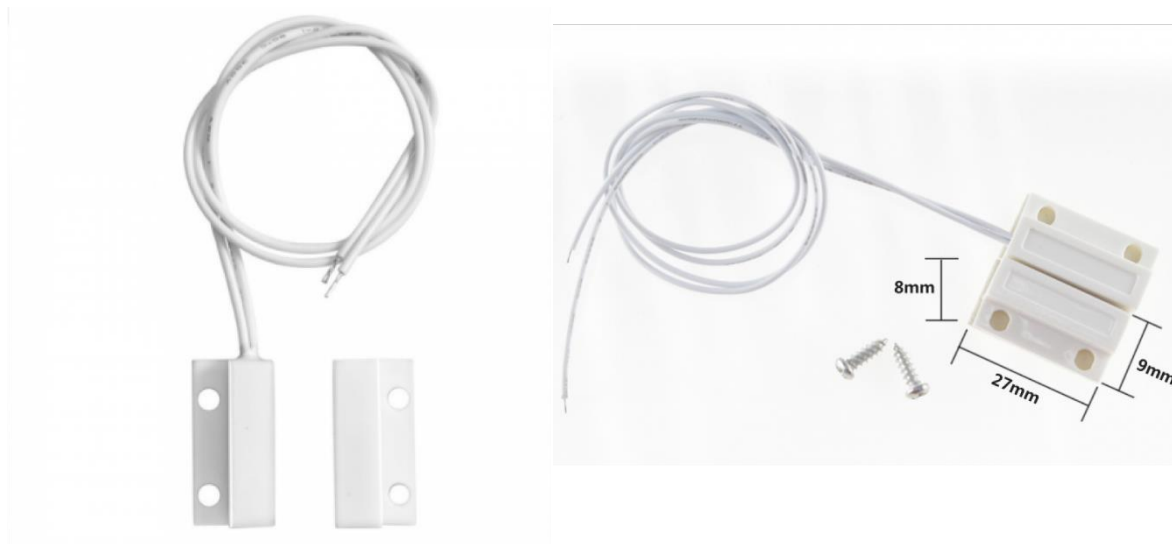
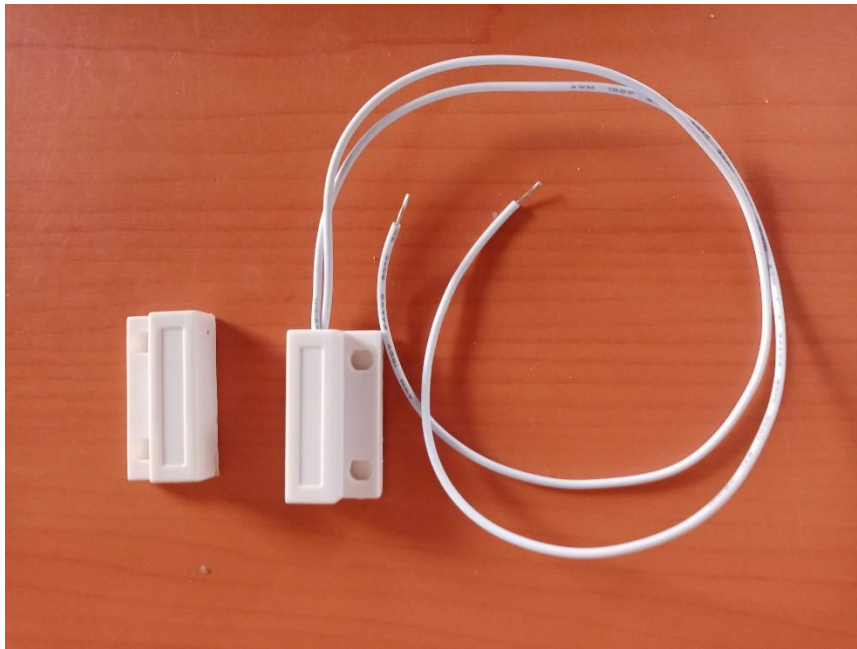
Σχήμα 26: Αποτύπωση σε Fritzing για τη χρήση 4 Relays

#### 4.8 Μαγνητικός αισθητήρας

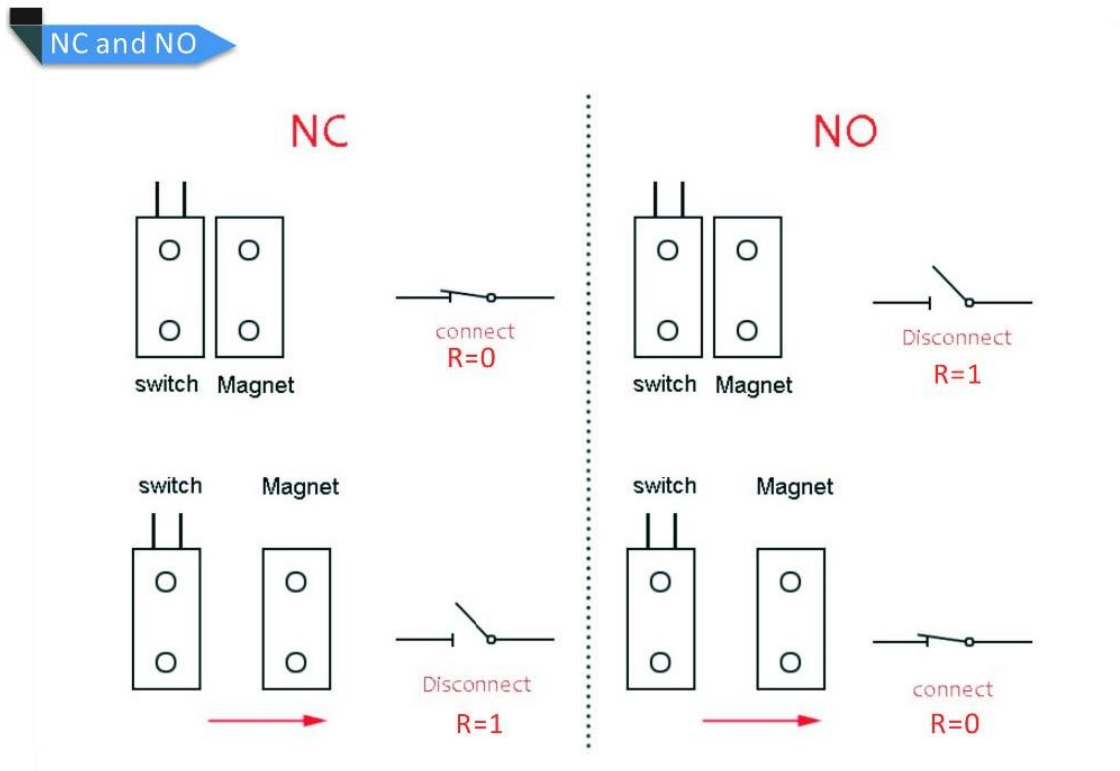
Το πως λειτουργεί κάθε μαγνητικός αισθητήρας ON/OFF θα γίνει περιγραφή πιο κάτω.

Ο μαγνητικός αισθητήρας ON/OFF που είναι ο ίδιος τύπος που έχουμε στους συναγερμούς στα παράθυρα θα είναι συνεχώς οπλισμένος δηλαδή ενεργοποιημένος (η εντολή έρχεται από τα Relay), ώστε να μην μπορεί να ανοίξει το κελί του φαρμάκου. Έτσι καταφέρνουμε να έχουμε την απαιτούμενη ασφάλεια και να μην μπορεί να ανοίξει το κελί από οποιονδήποτε άλλο.

Μόλις γίνει η επιβεβαίωση του ασθενούς ότι πρόκειται δηλαδή για την σωστή θέση και τον αντίστοιχο ασθενή τότε το relay δίνει εντολή απόπλισης του μαγνητικού αισθητήρα και είναι πλέον δυνατόν να ανοίξει το κελί. Έτσι δεν διαρρέει ρεύμα στο εσωτερικό μέρος του μαγνητικού αισθητήριου και απενεργοποιεί η μαγνήτιση.



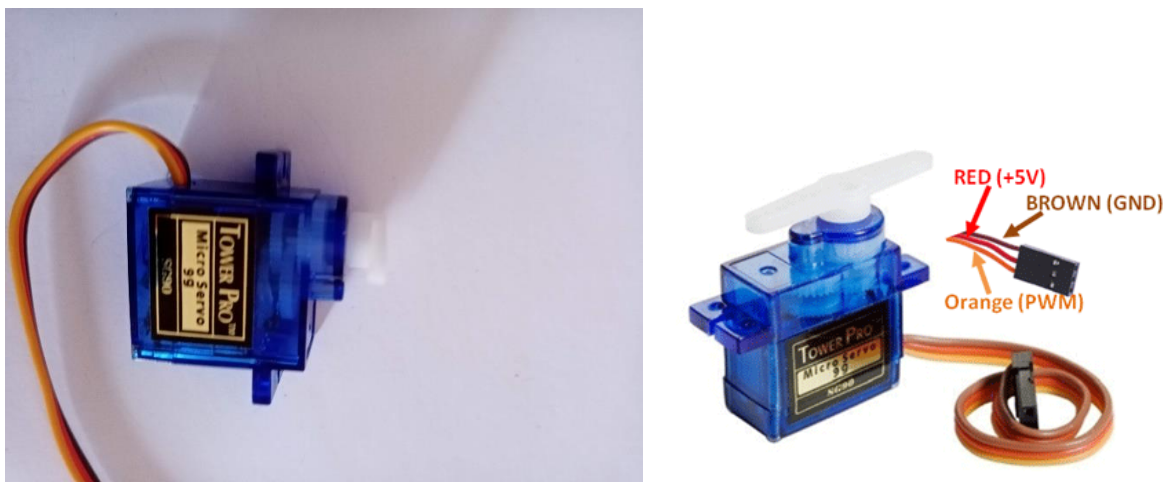
Σχήμα 27: Μαγνητικός αισθητήρας ON/OFF



Σχήμα 28: Παραστατικός τρόπος λειτουργίας του μαγνητικού αισθητήρα

#### 4.9 Servo κινητήρες

Οι Servo κινητήρες τύπου Tower Pro είναι βηματικοί κινητήρες με δυνατότητα περιστροφής του άξονα από 0° έως 360°. Είναι τριών ακροδεκτών με τροφοδοσία +5V, GND και πορτοκαλί την εντολή PWM ώστε να δίνει τον παλμό που θέλουμε.



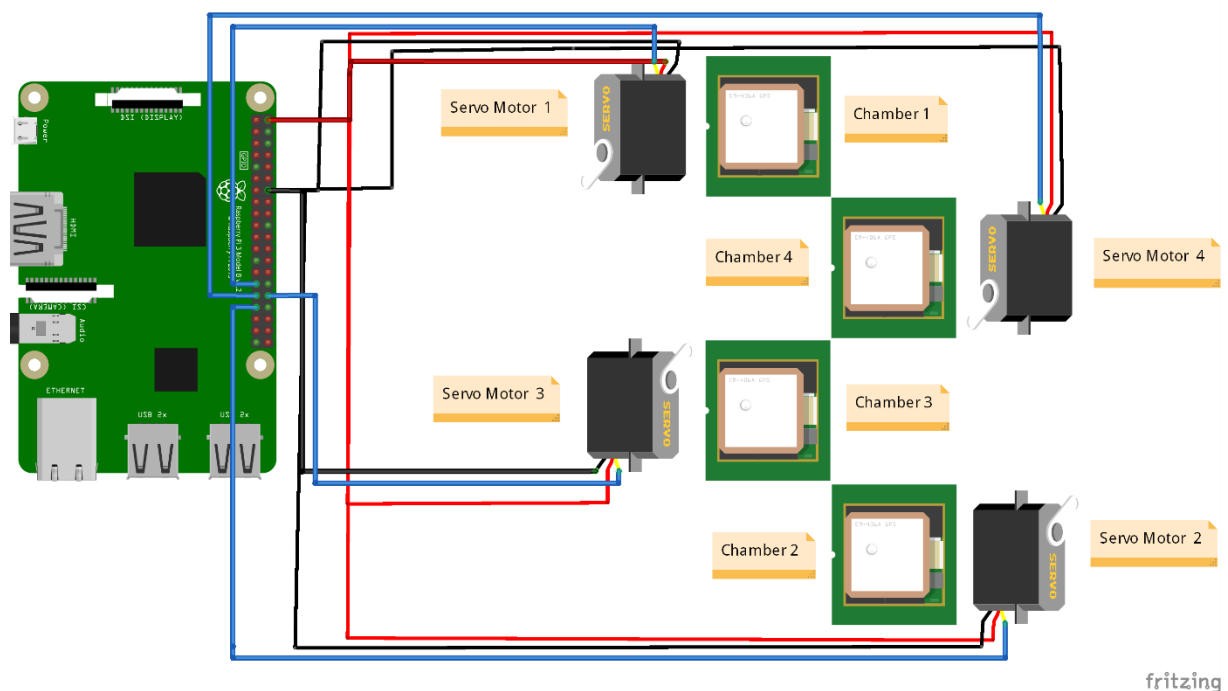
Σχήμα 29: Micro Step motor τύπου Tower Pro και διάταξη ακροδεκτών

Σε αυτούς τους Servo κινητήρες έρχεται η εντολή από το πρόγραμμα να ανοιγοκλείσσουνε κατά μια γωνία γύρω στις 90° ώστε να ανοίξει το καπάκι στο κελί (*Chamber*) και έτσι ο ασθενής να παραλάβει το φάρμακο του.

Μετά την περιστροφική κίνηση του ανοίγματος και το ξετάπωμα του κελιού ο άξονας περιστροφής θα κάνει την αντίστροφη κίνηση να επανέρθει στην αρχική του θέση αφού περάσουν κάποια λεπτά που εμείς θα ορίσουμε σαν κατώφλι μέσα στο πρόγραμμα. Όταν θα ολοκληρωθεί αυτή η επιστροφή του άξονα, ο ασθενής θα βεβαιωθεί οπτικά ότι επανήλθε στην θέση του και θα ταπώσει το κελί. Αυτό θα δώσει και το έναυσμα στον μαγνητικό αισθητήρα ON/OFF να οπλίσει και το τροχοφόρο να συνεχίσει την πορεία του.

Παραστατικά το σχέδιο σε Fritzing (servo motor control.fzz)

5V σύνδεση	στο Pin 2 (+5V)	κόκκινο	GND σύνδεση	στο Pin 14 (GND)	μαύρο
Chamber 1 σύνδεση	στο Pin 29 (GPIO5)	μπλε	Chamber 2 σύνδεση	στο Pin 31 (GPIO6)	μπλε
Chamber 3 σύνδεση	στο Pin 32 (GPIO12)	μπλε	Chamber 4 σύνδεση	στο Pin 33 (GPIO13)	μπλε



Σχήμα 30: Αποτύπωση σε Fritzing για τη χρήση 4 Micro Servo Motor

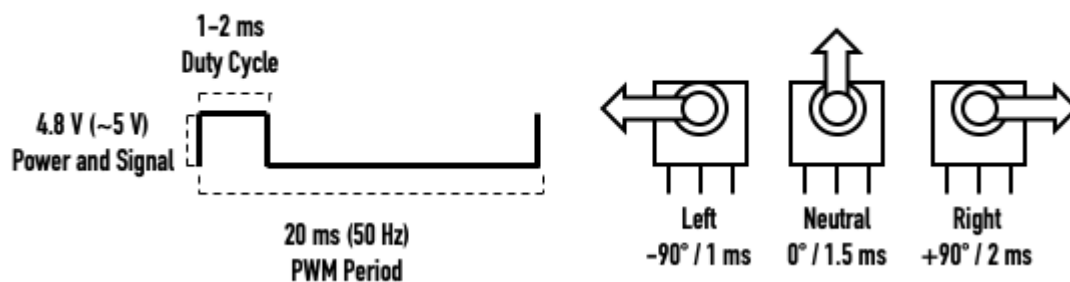
Στην τελική έκδοση βάζουμε και μια αντίσταση 1 KΩ σε κάθε ακροδέκτη PWM για να μην έχουμε θέμα, σε κάθε μπλε αγωγό που φαίνεται στο σχήμα 30 και πριν πάει η γραμμή στο Raspberry.

Τεχνικά οι εγκοπές πάνω στο κουτί των φαρμάκων και ακριβέστερα στα πλάγια των κελιών είναι για να μπορούμε να περάσουμε τον άξονα του micro servo κινητήρα μέσα για να ανοιγοκλείσουμε το καπάκι. Επίσης στις παρακάτω εικόνες φαίνεται και η θέση των μαγνητικών αισθητηρίων πως έχουν τοποθετηθεί στο εσωτερικό και εξωτερικό περιμετρικά.

Για να υπολογιστεί το πόσο θα περιστρέφει ο άξονας του Step Motor θα πρέπει να υπολογίσουμε για 3 θέσεις το Duty Cycle και την αντίστοιχη γωνία περιστροφής. Θεωρούμε, ως αφετηρία την κεντρική (0°) και θα περιστρεφόμαστε σε 2 θέσεις αριστερά 90° και δεξιά 90°. Στο παρακάτω σχήμα φαίνονται οι γωνίες περιστροφής για αριστερά κέντρο δεξιά.



## LEARN ROBOTICS



Για να υπολογίσουμε τον κύκλο εργασίας επί της 100 για κάθε θέση θα πρέπει να κάνουμε διαίρεση της συνολικής περιόδου της πλήρης περιστροφής. Next, convert the duty cycle times to percentages by dividing the position's duty cycle by the total PWM Period (20 ms). For example, the left position has a duty cycle = 1 ms. Therefore,  $1/20 \times 100\% = 5\%$ . I've included this handy chart for your reference.

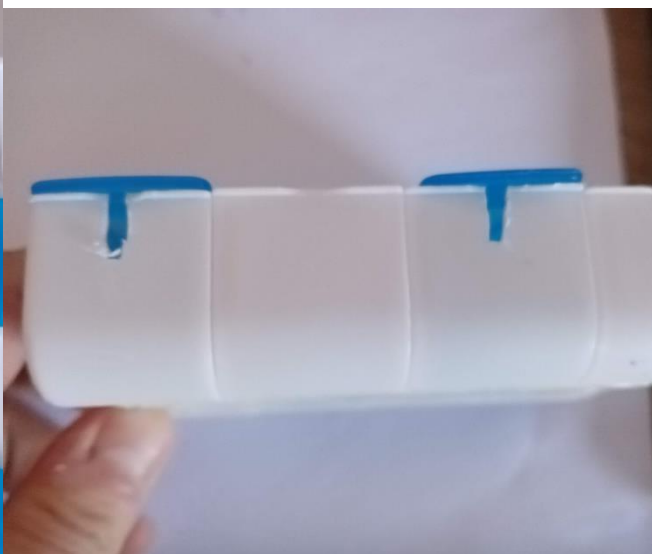
	Left	Neutral	Right
Angle (deg)	-90	0	+90
Duty Cycle (ms)	1 ms	1.5 ms	2 ms
Duty Cycle (%)	5%	7.5%	10%



Σχήμα 31: Α) Pill box ευρείας χρήσης



**Β)** Κάτοψη Pill box



**Γ)** Πλαϊνή όψη με εγκοπές χωρίς αισθητήρια



**Σχήμα 32:** Pill box με αισθητήρες μαγνητικούς και micro servo κινητήρα

## Κεφάλαιο 5°

### 5.1 Εισαγωγικά

Στο κεφάλαιο 5° θα δείξουμε λεπτομερώς βήμα προς βήμα το λογισμικό χρήσης για τη διασύνδεση του Raspberry Pi με τον Υπολογιστή για τον προγραμματισμό του καθώς και πως αναπτύχθηκε και υλοποιήθηκε όλο το λογισμικό μέχρι το τελικό στάδιο.

### 5.2 Βήματα για την διασύνδεση της μονάδας Raspberry Pi με τον Υπολογιστή

Ακολουθούν τα βήματα για την διασύνδεση της μονάδας **Raspberry Pi3, Pi Zero** ή **Pi Zero W** για να έχουμε SSH πρόσβαση μέσω της θύρας USB σε περιβάλλον Windows.

Μια από τις βασικότερες ερωτήσεις προκύπτει όταν έχουμε αγοράσει την μονάδα πως θα έχουμε πρόσβαση σε προγραμματιστικό περιβάλλον ή καλύτερα την διαχείριση χωρίς να χρειάζεται να συνδέσουμε επιπλέον εξοπλισμό όπως ένα USB Hub για την σύνδεση USB ποντίκι, USB πληκτρολόγιο ή και οθόνης για την αποτύπωση και λοιπών συσκευών όπως και σαφώς αντάπτορα για να συνδέσουμε από micro HDMI (που έχει το **Pi Zero**) σε HDMI για τη σύνδεση με οθόνη νέας τεχνολογίας ή ακόμα και σε τηλεόραση.

Όλα αυτά θα δούμε πως είναι περιττά και πως θα συνδέσουμε την μονάδα απλά μόνο με καλώδιο USB σε υπολογιστή / φορητό υλοποιώντας τα παρακάτω βήματα.

#### **Βήμα 1°:**

Κατεβάζουμε μια διανομή **Raspbian Stretch Lite** για το Raspberry Pi από την επίσημη ιστοσελίδα<sup>9</sup> (υπάρχουν τα images στις πηγές της Διπλωματικής).

Την περνάμε με ένα πρόγραμμα εγγραφής<sup>10</sup> σε μια micro SD κάρτα χωρητικότητα τουλάχιστον 8GB.

#### **Βήμα 2°:**

Το ssh πρωτόκολλο δεν είναι ενεργοποιημένο για λόγους ασφαλείας εξ αρχής στο λειτουργικό **Raspbian** και θα πρέπει χειροκίνητα να ενεργοποιηθεί εκ των υστέρων από το χρήστη πριν από την 1<sup>η</sup> χρήση. Για να το ενεργοποιήσουμε ακολουθούμε τα εξής βήματα:

1. Τρέχουμε το **Σημειωματάριο**.
2. Κάνουμε ένα κενό μέσο σε νέο έγγραφο.
3. Αποθηκεύουμε στο **root της SD κάρτας** ως νέο αρχείο με το όνομα **ssh και κατάληξη «όλα τα αρχεία»**.
4. Κλείνουμε το αρχείο.

---

<sup>9</sup> <https://www.raspberrypi.org/downloads/raspbian/>

<sup>10</sup> Etcher πρόγραμμα εγγραφής εικόνων σε αποθηκευτικά μέσα.

### Βήμα 3<sup>ο</sup>:

Επεξεργασία του αρχείου **config.txt** για να έχουμε πρόσβαση ssh προσθέτοντας στο τέλος του αρχείου το παρακάτω

**dtoverlay=dwc2**

### Βήμα 4<sup>ο</sup>:

Επεξεργασία και αποθήκευση του αρχείου **cmdline.txt** για να έχουμε πρόσβαση **ssh** προσθέτοντας μετά το σημείο **rootwait** τα παρακάτω:

**modules-load=dwc2,g\_ether**

*Προσοχή να υπάρχει μονό κενό πριν και μετά το πρόσθετο στοιχείο γιατί αλλιώς δεν θα παίζει.*

### Βήμα 5<sup>ο</sup>:

Εγκατάσταση του **Bonjour** σε περιβάλλον Windows.

Για να έχουμε πρόσβαση **ssh** σε περιβάλλον Windows στο Raspberry Pi βάζοντας το με hostname ακολουθούμενο από **.local** (π.χ.: **raspberrypi.local**) θα πρέπει να εγκατασταθεί στο λειτουργικό μας η υπηρεσία Bonjour<sup>11</sup>.

### Βήμα 6<sup>ο</sup>:

Ενεργοποίηση του **Pi Zero** ή **οποιοδήποτε Raspberry Pi**

1. Βάζουμε την micro SD card μέσα στον ειδικό οδηγό στο Pi Zero.
2. Συνδέουμε το καλώδιο Micro-USB στην υποδοχή **data/peripherals** χωρίς να χρειαστεί να συνδέσουμε επιπλέον τροφοδοσία.
3. Δίνουμε τον απαιτούμενο χρόνο 2-3 λεπτών μέχρι που να φορτωθεί το λειτουργικό στο Pi Zero και να είναι έτοιμο προς χρήση.

### Βήμα 7<sup>ο</sup>:

Ανοίγουμε το πρόγραμμα **putty** για την επικοινωνία **ssh**.

Οι ρυθμίσεις είναι όπως φαίνονται στο παρακάτω σχήμα.

Στο περιβάλλον εκκίνησης μόλις ζητηθεί εισαγωγή στοιχείων βάζουμε:

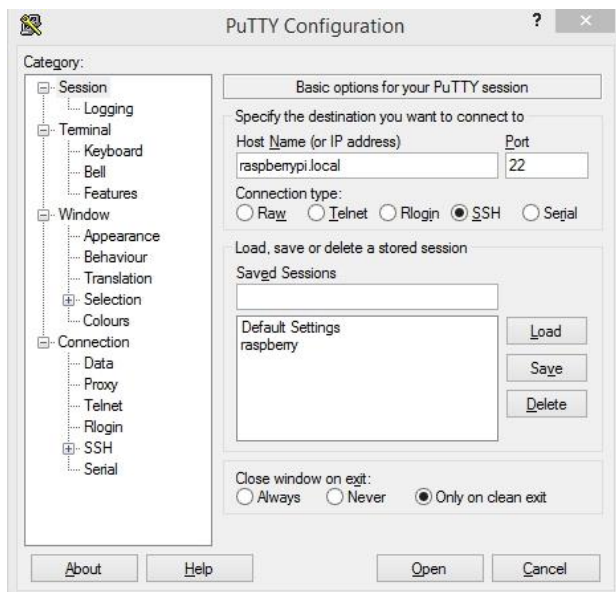
1. Στο πλαίσιο **Host Name (or IP address)** το όνομα **raspberrypi.local**
2. Η πόρτα (**Port**) εξ' ορισμού είναι η **22** και η σύνδεση **Connection type** σετταρισμένη σε **SSH**.

<sup>11</sup> [https://support.apple.com/kb/DL999?locale=en\\_US](https://support.apple.com/kb/DL999?locale=en_US)

3. Πατάμε το **Open** και περιμένουμε να μας βγάλει στο πλαίσιο διαλόγου για τα διαπιστευτήρια στα οποία εισάγουμε:

*Username: pi και password: raspberry.*

**Μετά την επιτυχή είσοδο θα πρέπει να είμαστε στο root του Raspberry Pi.**



**Σχήμα 33:** Ρυθμίσεις για πρόσβαση ssh

### **Βήμα 8<sup>ο</sup>:**

Πρόσβαση στο δίκτυο από το σύστημα των Windows

Επειδή εξ' αρχής η σύνδεση η ενσύρματη στο σύστημα των Windows δεν είναι ρυθμισμένη να παρέχει και ταυτόχρονα διασύνδεση στο Internet, θα πρέπει από μεριάς μας να πραγματοποιήσουμε κάποιες χειροκίνητες ενέργειες ώστε να δώσουμε στην μονάδα πλήρη πρόσβαση.

Αυτά τα βήματα περιγράφονται στη συνέχεια.

1. Ανοίγουμε τον **Πίνακα Ελέγχου** και τρέχουμε το **Κέντρο Δικτύου και κοινής Χρήσης**.
2. Εξαρτάται από το τρόπο με τον οποίο είμαστε συνδεδεμένοι για να επιλέξουμε ή την ενσύρματη ή την ασύρματη σύνδεση μας και μετά
3. **Click Ιδιότητες**
4. **Click στην καρτέλα κοινή χρήση.**
5. Ενεργοποιούμε την επιλογή **Να επιτρέπεται η χρήση άλλων χρηστών του διαδικτύου από αυτή τη σύνδεση.**

6. Click **OK** και μετά **Κλείσιμο**.
7. Για έλεγχο θα μπορούσα μέσα στο παράθυρο να βάζαμε ping [www.hmu.gr](http://www.hmu.gr) και να δούμε εάν έχουμε απάντηση στα αιτήματα ping.

Ολοκληρώνοντας αυτά τα βήματα θα έχουμε πετύχει την διασύνδεση της μονάδας Raspberry με τον Η/Υ ή και φορητό μας.

**Προσοχή !!!**

**Πρέπει οπωσδήποτε να εκτελεστούν τα βήματα πάντα πριν την πρώτη χρήση και σύνδεση της πλακέτας ανεξαρτήτως μηχανήματος.**

### **5.3 Βήματα για την αυτοματοποιημένη εκκίνηση της μονάδας Raspberry Pi για την εκτέλεση του προγράμματος**

Ένα άλλο ζήτημα είναι πως θα κάνουμε το Raspberry Pi να τρέξει αυτοματοποιημένα με την εκκίνηση το πρόγραμμα που θέλουμε.

Το Raspberry δεν διαθέτει autogun όπως διαθέτει το Arduino με αποτέλεσμα να έχουμε πρόβλημα για την αυτόματη εκτέλεση προγράμματος κάθε φορά που αυτό ενεργοποιείτε ή που τίθεται σε λειτουργία μετά από εκκίνηση.

Με το **Cron** που υπάρχει στα συστήματα Linux, μπορούμε να κάνουμε προγραμματισμό πότε θέλουμε να επαναληφθούν κάποιες εργασίες στο χώρο-χρόνο δίνοντας στον προγραμματιστή άπειρες δυνατότητες. Εμείς θα προσπαθήσουμε να το προσαρμόσουμε στα μέτρα μας τρέχοντας ένα script σε python που θα τρέχει το πρόγραμμα μας κάθε φορά που εκκινείτε το Raspberry Pi.

#### **Βήμα 1<sup>ο</sup>: Δημιουργία του Python Script**

Το πρώτο βήμα είναι να δημιουργήσουμε το script σε Python. Αυτό θα τρέχει όταν θα κάνει boot. Είναι πολύ σημαντικό να γνωρίζουμε το ακριβές όνομα και τη θέση που βρίσκετε. Στη δικιά μας υλοποίηση θα το ονομάσουμε «**Mediberry.py**» και θα το αποθηκεύσουμε στην τοποθεσία «**/home/pi/**».

Βλέπουμε και τσεκάρουμε αν έχουμε δώσει τη σωστή θέση στο αρχείο

```
cat /home/pi/Mediberry.py
```

Θα πρέπει να μας δείξει το περιεχόμενο του script που φτιάξαμε.

Εδώ σημειώνουμε ότι θα πρέπει να έχουμε διαπιστώσει ότι το συντάξαμε σωστά γιατί τρέχει κατά την εκκίνηση και δεν είναι εύκολο να το διορθώσουμε άπαξ και τρέξει.

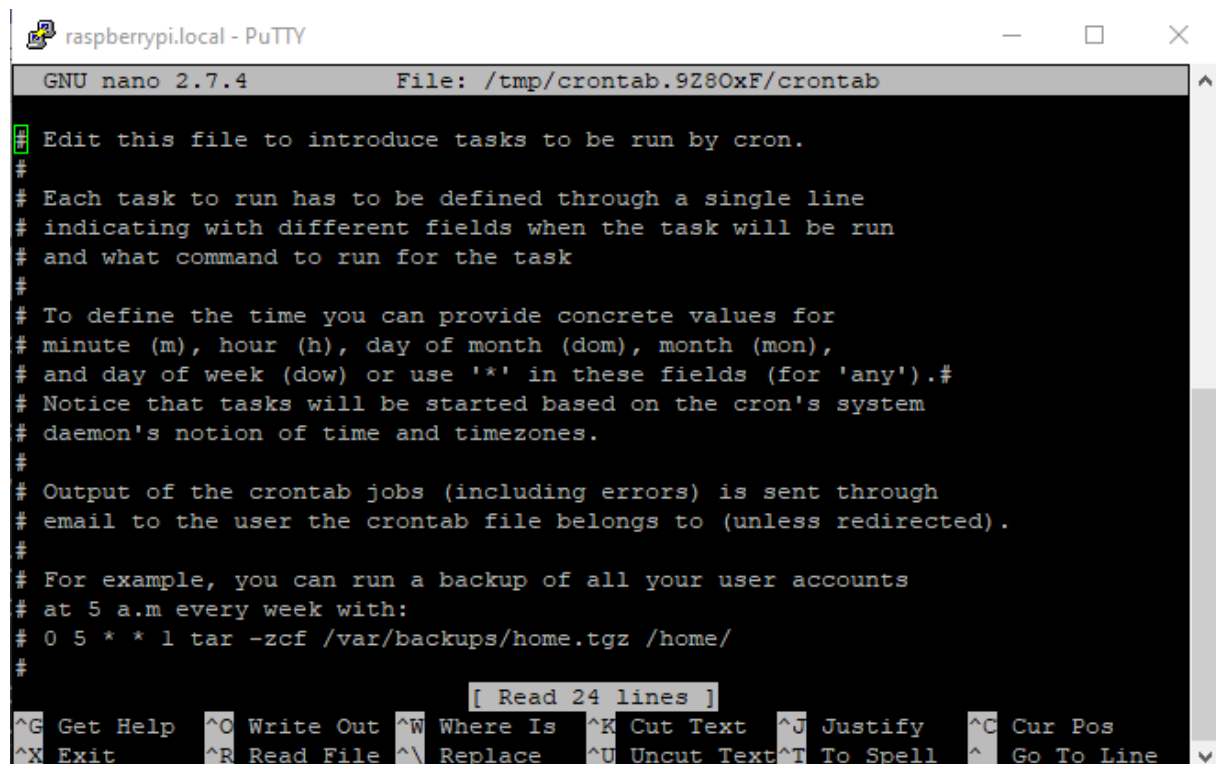
#### **Βήμα 2<sup>ο</sup>: Δημιουργία νέου Cron job**



Για τη δημιουργία ενός νέου **Cron job** θα πρέπει να αλλάξουμε το «**crontab**». Είναι στην ουσία ένας πίνακας με λίστα από αυτοματοποιημένες εργασίες που ο **Cron** εποπτεύει και εκτελεί βάσει χρονοδιαγράμματος. Για να το τροποποιήσουμε θα εκτελέσουμε τις εξής εντολές:

```
sudo crontab -e
```

Κάθε χρήστης του λειτουργικού συστήματος έχει το δικό του Crontab, οπότε για να αποφύγουμε τυχόν δυσλειτουργίες που μεταφράζετε να αστοχεί να τρέχει το προαναφερθέν script από όλους τους χρήστες, στόχος μας είναι να το περάσουμε ως διαχειριστής admin οπότε γι' αυτό βάζουμε ως πρόθεμα το «**sudo**» στο «**crontab -e**». Οπότε θα πάρουμε ως αποτέλεσμα αυτό που βλέπουμε στην παρακάτω εικόνα.



```
raspberrypi.local - PuTTY
GNU nano 2.7.4 File: /tmp/crontab.9Z80xF/crontab
# Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
[ Read 24 lines ]
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line
```

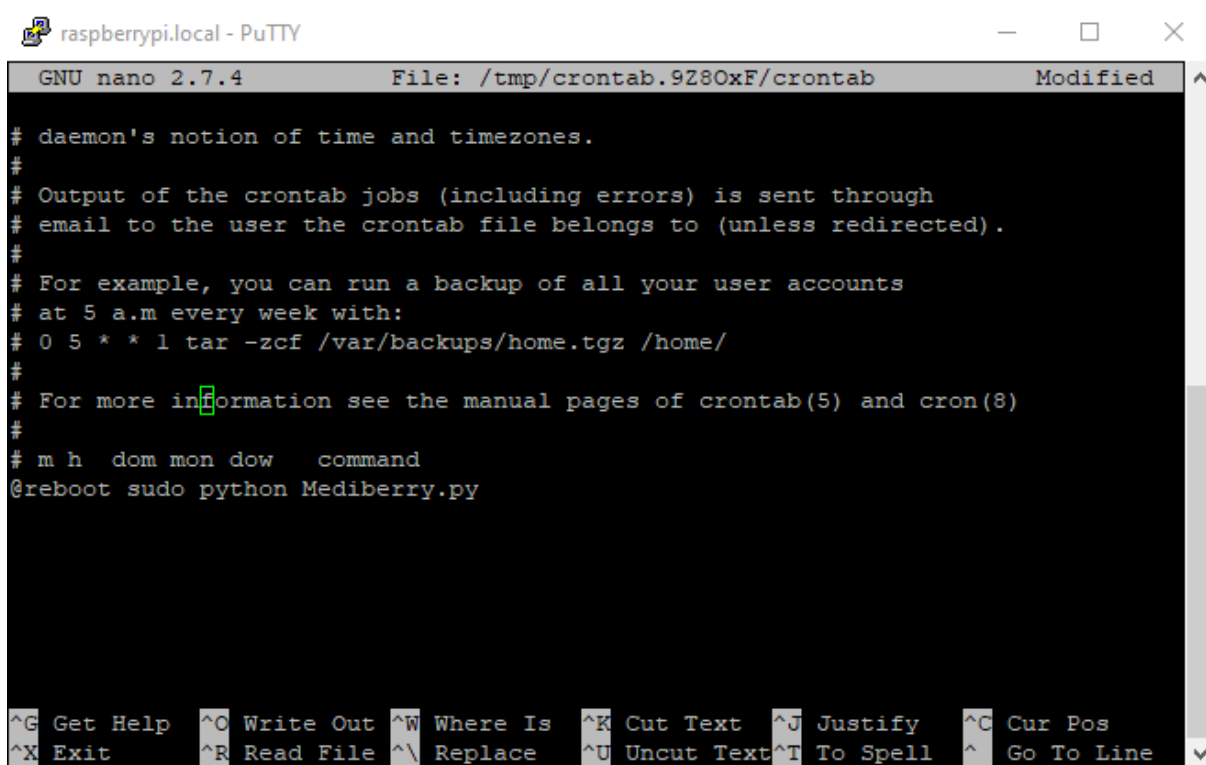
Σχήμα 34: Αρχική οθόνη στο Crontab

Στο τέλος αυτού θα πρέπει να περάσουμε το εξής για να τρέχει κατά την εκκίνηση το script

```
@reboot sudo python Mediberry.py &
```

Το Cron τώρα μαθαίνει ότι σε κάθε εκκίνηση boot (or reboot or start-up) θέλουμε να τρέξουμε το **Mediberry.py** αυτόματα στην Python. Το «και» στο τέλος «&» έχει την έννοια να τρέχει στο παρασκήνιο και δε σταματάει όπως πριν.

Στην οθόνη θα βλέπουμε τα εξής:



```
raspberrypi.local - PuTTY
GNU nano 2.7.4 File: /tmp/crontab.9Z80xF/crontab Modified
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h dom mon dow   command
@reboot sudo python Mediberry.py
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify   ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text ^T To Spell  ^_ Go To Line
```

Σχήμα 35: Οθόνη στο Crontab μετά την εισαγωγή του script μας

Για να το σώσουμε πατάμε «CTRL-X» για έξοδο, μετά «Y» για επιβεβαίωση και στο τέλος «Return». Μετά από αυτή τη διαδικασία επιστρέφουμε στο Command Prompt και κάνουμε τα τεστ μας στην επόμενη επανεκκίνηση.

#### Sudo reboot

*Αφού το έχουμε τώρα σετάρει κάθε φορά θα τρέχει με την εκκίνηση του Pi.*

**Θα υπάρξουν όμως και φορές που δεν θα θέλουμε να τρέχει αυτόματα το σκριπτάκι και για αυτό το λόγο θα πρέπει να το σταματήσουμε.**

Στόχος είναι πληκτρολογώντας την παρακάτω εντολή να βρούμε τον αριθμό εκτέλεσης της διαδικασίας και να του κάνουμε «kill».

#### Ps aux | grep /home/pi/Mediberry.py

Μέσα στα όλα αποτελέσματα μας βγάζει και τα παρακάτω από όπου και σταματάμε την εκτέλεση δίνοντας του την τελική εντολή «kill» με δικαιώματα διαχειριστή.

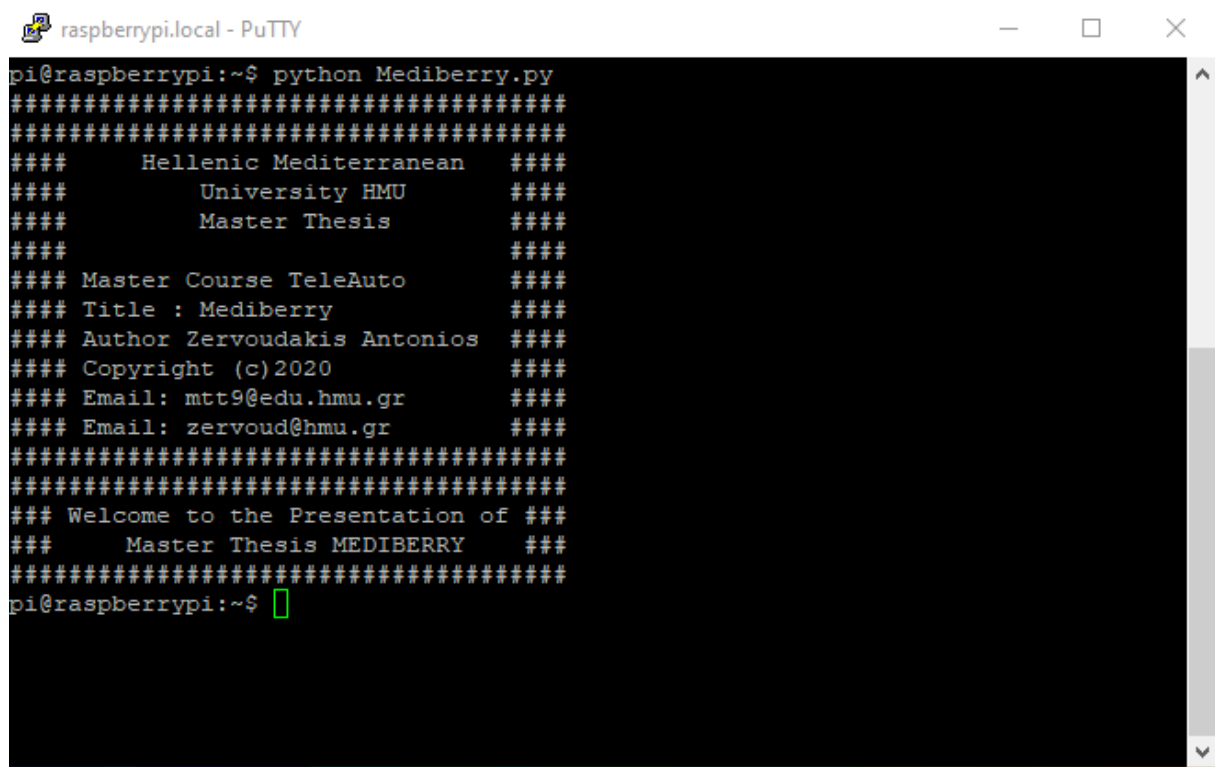
```
root 1863 0.0 1.0 24908 4012 ? Sl 19:45 0:00 python /home/pi/Mediberry.py
```

```
sudo kill 1863
```

Στη δικιά μας περίπτωση για την εκτέλεση του προγράμματος θα ενεργήσουμε απλούστατα καλώντας το αντίστοιχο αρχείο Mediberry.py οπότε η εντολή στην κονσόλα θα είναι:

**python Mediberry.py**

και η εκτέλεση του προγράμματος θα μας βγάλει το welcome screen.



```
raspberrypi.local - PuTTY
pi@raspberrypi:~$ python Mediberry.py
#####
#####
####   Hellenic Mediterranean   ####
####   University HMU           ####
####   Master Thesis            ####
####                               ####
#### Master Course TeleAuto     ####
#### Title : Mediberry          ####
#### Author Zervoudakis Antonios ####
#### Copyright (c)2020          ####
#### Email: mtt9@edu.hmu.gr     ####
#### Email: zervoud@hmu.gr      ####
#####
#####
### Welcome to the Presentation of ###
###   Master Thesis MEDIBERRY   ###
#####
pi@raspberrypi:~$
```

**Σχήμα 36:** Αρχική οθόνη εκτέλεσης

*Στη συνέχεια στο κεφάλαιο 6 παραθέτουμε το πρόγραμμα της υλοποίησης*

## Κεφάλαιο 6°

### 6.1 Εισαγωγικά

Εν συνεχεία, στο κεφάλαιο 6° ακολουθώντας όλα τα στάδια που προαναφέρθηκαν θα παρουσιαστούν όλα τα στάδια υλοποίησης μαζί αφού έχουν διεξοδικά αναλυθεί και από το εργαστηριακό κομμάτι προχωράμε στην αυτού καθαυτού υλοποίηση και στα σενάρια υλοποίησης σε πραγματικό περιβάλλον.

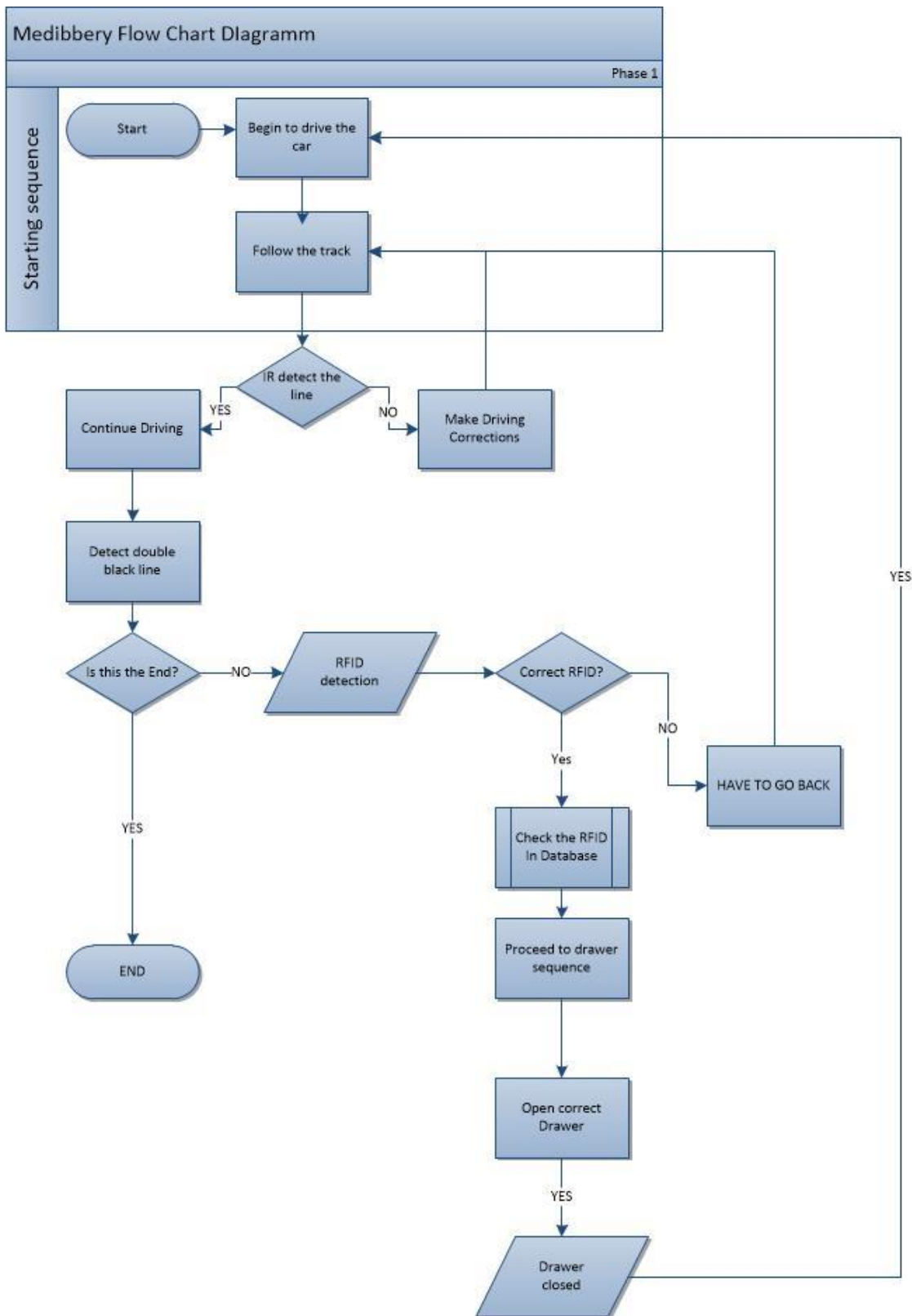
*Τρία Σενάρια* θα μας απασχολήσουν σε αυτή την υλοποίηση

1. Υλοποίηση σε πλήρη λειτουργία
2. Υλοποίηση με RFID σε λάθος θέσεις
3. Υλοποίηση με εμπόδιο στην διαδρομή

Πρώτα από όλα όμως θα δώσουμε το πλήρες διάγραμμα ροής της υλοποίησης.

### 6.2 Διάγραμμα ροής της υλοποίησης

Το πλήρες διάγραμμα ροής της υλοποίησης σας μας φαίνεται στο παρακάτω σχήμα

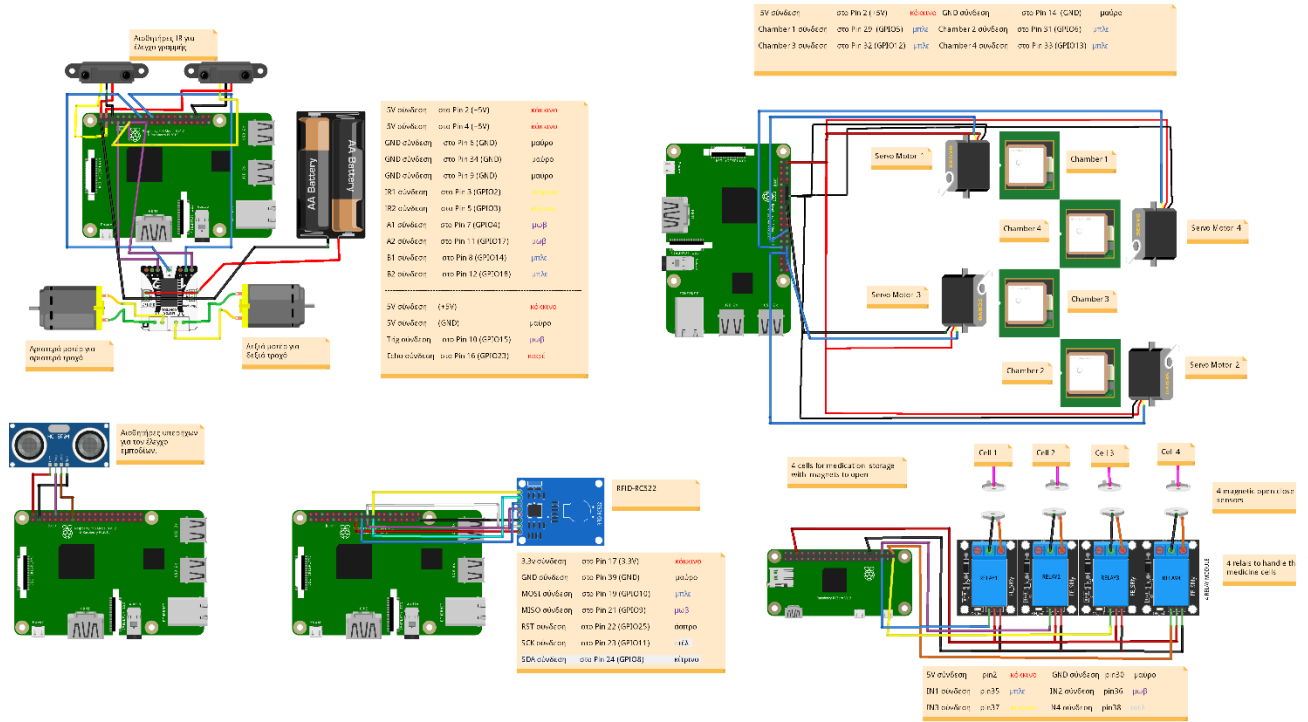


Σχήμα 37: Πλήρες διάγραμμα ροής “Mediberry”

### 6.3 Το πλήρες σχέδιο σε Fritzing

Ολοκληρώνοντας όλα τα στάδια της υλοποίησης και της ηλεκτρονικής σχεδίασης με το πρόγραμμα Fritzing ακολουθεί το πλήρες ηλεκτρονικό σχέδιο με όλες τα επι μέρους τμήματα τα οποία έχουν αναλυθεί σε προηγούμενες ενότητες.

Αποτύπωση σε Fritzing του όλου κυκλώματος της υλοποίησης (thesis final shematic.fzz).



Σχήμα 38: Τελικό σχέδιο διάταξης μαζί με συνδέσεις

Στο σχήμα αυτό δεν τα βάλουμε όλα μαζί για να δείξουμε επ’ ακριβώς τα διάφορα μέρη της υλοποίησης.

Πάνω στο GPIO του Raspberry έχουμε τις εξής συνδέσεις πραγματοποιήσει:

5V σύνδεση	στο Pin 2	GND σύνδεση	στο Pin 6
5V σύνδεση	στο Pin 4	GND σύνδεση	στο Pin 9
3.3v σύνδεση	στο Pin 17	GND σύνδεση	στο Pin 14
LED RED 1	στο Pin 13	GND σύνδεση	στο pin30
LED RED 2	στο Pin 18	GND σύνδεση	στο Pin 34
Switch	στο Pin 15	GND σύνδεση	στο Pin 39
Trig σύνδεση	στο Pin 10 (GPIO15)	IN1 σύνδεση	στο pin35
Echo σύνδεση	στο Pin 16 (GPIO23)	IN2 σύνδεση	στο pin36
Chamber 1	στο Pin 29 (GPIO5)	IN3 σύνδεση	στο pin37
Chamber 2	στο Pin 31 (GPIO6)	IN4 σύνδεση	στο pin38
Chamber 3	στο Pin 32 (GPIO12)	MOSI σύνδεση	στο Pin 19 (GPIO10)
Chamber 4	στο Pin 33 (GPIO13)	MISO σύνδεση	στο Pin 21 (GPIO9)
IR1 σύνδεση	στο Pin 3 (GPIO2)	RST σύνδεση	στο Pin 22 (GPIO25)
IR2 σύνδεση	στο Pin 5 (GPIO3)		
A1 σύνδεση	στο Pin 7 (GPIO4)		



<i>A2 σύνδεση στο Pin 11 (GPIO17)</i>
<i>B1 σύνδεση στο Pin 8 (GPIO14)</i>
<i>B2 σύνδεση στο Pin 12 (GPIO18)</i>

<i>SCK σύνδεση στο Pin 23 (GPIO11)</i>
<i>SDA σύνδεση στο Pin 24 (GPIO8)</i>

#### **6.4 Το πλήρες πρόγραμμα σε Python**

Η τελική παράγραφος αυτού του κεφαλαίου είναι ο πλήρες κώδικας της υλοποίησης σε python όπως ακριβώς τρέχει μέσα στο Raspberry Pi.

Ο κώδικας είναι γραμμή προς γραμμή και δεν θα γίνει ιδιαίτερη αναφορά ούτε κάποια επεξήγηση του σε αυτό το σημείο, προϋποθέτουμε να έχει ο αναγνώστης κάποιες βασικές γνώσεις σε Python για να τον καταλάβει.

#### **Κώδικας Mediberry**

```
#!/usr/bin/python:
print("#####")
print("### Master Thesis          ###")
print("### Hellenic Mediterranean University  ###")
print("### Master Course Teleauto      ###")
print("### Title : Mediberry           ###")
print("### Author Zervoudakis Antonios  ###")
print("### Copyright 2020              ###")
print("### Email: mtt9@edu.hmu.gr      ###")
print("### zervoud@hmu.gr             ###")
print("#####")
print("#####")
print("### Welcome to the Presentation  ###")
print("#####")
import RPi.GPIO as GPIO # calling header file for GPIO's of PI
import time
from time import sleep #import sleep function to call later
GPIO.setwarnings(False) # Ignore multiple warnings
GPIO.setmode(GPIO.BOARD)
GPIO.setup(13,GPIO.OUT) # RED LED Indicator
GPIO.setup(15,GPIO.IN) # Switch On/Off
#while True:
    if GPIO.input(15):
        GPIO.output(13,GPIO.HIGH)
        print ("LED ON")
    else:
        GPIO.output(13,GPIO.LOW)
        print ("LED OFF")
print("Running full Mediberry programm")
print("proceeding to start wheels")

# *****
```

```

# *          START OF ULTRASONIC FUNCTION          *
# *****
def ultrasonic_function():
# We setup the pins as Inputs and Outputs
# for controlling the Ultrasonic Sensor and
# obstacle recognition to halt
GPIO.setup(10,GPIO.OUT) #TRIGGER PULSE
GPIO.setup(16,GPIO.IN) #ECHO PULSE
GPIO.setup(18,GPIO.OUT) #RED INDICATOR for Distance
print ("Distance Measurement In Process")
GPIO.output(10, False)
print ("Waiting For Sensor To Travel forward")
time.sleep(.1)
GPIO.output(10,GPIO.HIGH)
time.sleep(0.00001)
GPIO.output(10,GPIO.LOW)

while GPIO.input(16) == 0:
    pass
    pulse_start1 = time.time()

while GPIO.input(16) == 1:
    pass
    pulse_end1 = time.time()

pulse_duration1 = pulse_end1 - pulse_start1

distance1 = pulse_duration1 * 17150
distance1= round(distance1, 2)
print ("Distance1:",distance1, "cm")

time.sleep(10)
#while True:

    if distance1<=0.2:
        GPIO.output(18,GPIO.HIGH)
        print("LED ON OBSTACLE IN RANGE UNDER 20cm")
    else:
        GPIO.output(18,GPIO.LOW)
        print("LED OFF OBSTACLE OUT OF RANGE")
pause()
GPIO.cleanup() #this ensures a clean exit

# *****
# *          START OF MAIN FUNCTION          *
# *****
def Main_function(): print("starts Main Function")
# First we setup pins as Inputs and Outputs

```

```

# for controlling the direction of the Wheels

GPIO.setup(3,GPIO.IN)
GPIO.setup(5,GPIO.IN)
GPIO.setup(7,GPIO.OUT)
GPIO.setup(8,GPIO.OUT)
GPIO.setup(11,GPIO.OUT)
GPIO.setup(12,GPIO.OUT)
try:
    while True:
        if GPIO.input(3)==False and GPIO.input(5)==False:
            print("Forward movement")
            GPIO.output(7,True)
            GPIO.output(8,False)
            GPIO.output(11,False)
            GPIO.output(12,True)

        if GPIO.input(3)==True and GPIO.input(5)==False:
            print("Left turn")
            GPIO.output(7,True)
            GPIO.output(8,True)
            GPIO.output(11,True)
            GPIO.output(12,False)

        if GPIO.input(3)==False and GPIO.input(5)==True:
            print("Right turn")
            GPIO.output(7,False)
            GPIO.output(8,True)
            GPIO.output(11,False)
            GPIO.output(12,False)

        if GPIO.input(3)==True and GPIO.input(5)==True:
            print("Reverse movement")
            GPIO.output(7,True)
            GPIO.output(8,True)
            GPIO.output(11,True)
            GPIO.output(12,True)
    finally:
        GPIO.cleanup() #this ensures a clean exit

# *****
# *           START OF Rfid FUNCTION           *
# *****

def rfid_function():
GPIO.setup(19,GPIO.IN) # MOSI
GPIO.setup(21,GPIO.IN) # MISO
GPIO.setup(22,GPIO.IN) # RST
GPIO.setup(23,GPIO.IN) # SCK

```

```
GPIO.setup(24,GPIO.IN) # SDA
```

```
# *****  
# *          START OF Magnetic lock FUNCTION          *  
# *****
```

```
def magnet_function():
```

```
GPIO.setup(35,GPIO.OUT) # IN1 Magnetic Lock 1  
GPIO.setup(36,GPIO.OUT) # IN2 Magnetic Lock 2  
GPIO.setup(37,GPIO.OUT) # IN3 Magnetic Lock 3  
GPIO.setup(38,GPIO.OUT) # IN4 Magnetic Lock 4  
GPIO.output(35,GPIO.HIGH)  
print ("IN1 Locked")  
GPIO.output(36,GPIO.HIGH)  
print ("IN2 Locked")  
GPIO.output(37,GPIO.HIGH)  
print ("IN3 Locked")  
GPIO.output(38,GPIO.HIGH)  
print ("IN4 Locked")  
GPIO.output(35,GPIO.LOW)  
print ("IN1 Opened")  
GPIO.output(36,GPIO.LOW)  
print ("IN2 Opened")  
GPIO.output(37,GPIO.LOW)  
print ("IN3 Opened")  
GPIO.output(38,GPIO.LOW)  
print ("IN4 Opened")
```

```
def servo_function():
```

```
# *****  
# *          START OF Servo rotation FUNCTION          *  
# *****
```

```
GPIO.setmode(GPIO.BOARD)  
GPIO.setup(29,GPIO.OUT) # Servo 1  
if GPIO.output(29,LOW)  
print ("Servo 1 Closed")  
else GPIO.output(29,HIGH)  
print ("Servo 2 open")  
GPIO.setup(31,OUT) # Servo 2  
if GPIO.output(31,GPIO.LOW)  
print ("Servo 2 Closed")
```

```

else GPIO.output(31,GPIO.HIGH)
print ("Servo 2 open")
GPIO.setup(32,GPIO.OUT) # Servo 3
GPIO.output(32,GPIO.LOW)
print ("Servo 3 Closed")
else GPIO.output(32,GPIO.HIGH)
print ("Servo 3 Open")
GPIO.setup(33,GPIO.OUT) # Servo 4
GPIO.output(33,GPIO.LOW)
print ("Servo 4 Closed")
else GPIO.output(32,GPIO.HIGH)
print ("Server 4 Open")

```

```

# Now we need an output to send our PWM signal on correct servo motor
if IN1=true:
pwm=GPIO.PWM(29, 50)
pwm.start(7)
#stay for 60 seconds open
time.sleep(60)
pwm.stop()
pwm.start(0)

```

```

elif IN2=True :
pwm=GPIO.PWM(31, 50)
pwm.start(7)
#stay for 60 seconds open
time.sleep(60)
pwm.stop()

```

```

elif IN3=True :
pwm=GPIO.PWM(32, 50)
pwm.start(7)
#stay for 60 seconds open
time.sleep(60)
pwm.stop()

```

```

else IN4=True :
GPIO.setup(33, GPIO.OUT) #Chamber 4
pwm=GPIO.PWM(33, 50)
pwm.start(7)
#stay for 60 seconds open
time.sleep(60)
pwm.stop()
GPIO.cleanup()

```

```

# *****
# *           START OF Cell FUNCTION           *

```

```

# *****
def cell_function():

# *****
# *           START OF THE MAIN PROGRAM           *
# *****

ultrasonic_function() #Ultrasonic Function to handle distance and
obstacles
Main_function() #Main Function to handle wheels
rfid_function() #Main Function to handle wheels
servo_function() #Servo function to handle servos
magnet_function() #Main Function to handle wheels
cell_function() #cell Function to handle opening and closing cells on

pill box
exit_function() # Function to exit properly

# *****
# *           FUNCTION TO EXIT THE PROGRAM CLEANLY           *
# *****
# * We use this function to turn off both motor and to ensure the *
# * GPIO ports are reset properly on exit *
# *****

def exit_program():
    z = motor_drive (False) # Turn off all the GPIO outputs
    print ("\n\n") # Print a couple of blank lines
    GPIO.cleanup() # Clean up GPIO ports with any
    exit() # And quit the program

```



## 6.5 Σενάρια Υλοποίησης

Παραθέτουμε παρακάτω σενάρια υλοποίησης της παρούσας υλοποίησης.

### a. Υλοποίηση σε πλήρη λειτουργία

**Πρώτο σενάριο** η υλοποίηση σε πλήρη λειτουργία περιλαμβάνει να κάνει το τροχοφόρο την πλήρη διαδρομή χωρίς εμπόδιο και λάθος ανίχνευση RFID's. **Αυτό αποτελεί και το ιδανικό και επιθυμητό σενάριο.** Εδώ δεν έχουμε να πούμε τίποτα άλλο παρά ότι το τροχοφόρο όχημα ακολουθεί κατά γράμμα και την διαδρομή και το πρόγραμμα της ρυθμ.

### b. Υλοποίηση με RFID σε λάθος θέσεις

**Στο δεύτερο σενάριο** έχουμε την ίδια ακριβώς πλήρη διαδρομή για το τροχοφόρο μόνο που αντιμετωπίζουμε στα σημεία 2 και 3 τους δέκτες RFid's μεταξύ τους. Αυτό έχει το νόημα να δείξουμε τι θα συμβεί αν 2 σημεία αλλάξουν την θέση τους ή καλύτερα αν οι ασθενείς δεν βρίσκονται στην θέση τους. Αυτό θα έχει σαν επίπτωση το τροχοφόρο να παραδίδει κανονικά στο σημείο 1. Αφού φτάσει στο σημείο 2 θα κάνει όλη την επικοινωνία στο κομμάτι του RFid και μόλις διαπιστώσει ότι δεν είναι ο σωστός παραλήπτης θα του ανοίξει το φάρμακο με το σωστό κελί. Δηλαδή εδώ στη θέση 2 θα του ανοίξει το κελί 3 και στη συνέχεια θα προχωρήσει στο σημείο 3 όπου απλά θα διαπιστώσει ότι είναι ο 2 και θα παραδώσει το φάρμακο στο κελί 2.

- **Μια πρώτη παραλλαγή** θα ήταν όλες οι θέσεις να ήταν ανακατεμένες και να παρέδιδε τα σωστά φάρμακα από τα αντίστοιχα κελιά στους σωστούς ασθενείς μετά από ορθή ταυτοποίηση των RFid τους.
- **Μια δεύτερη παραλλαγή** θα ήταν να πήγαινε από όλα τα σημεία αναφοράς ένα προς ένα (για το δικό μας παράδειγμα από το 1 έως το 4) και να παρέδιδε **μόνο** τα φάρμακα στις σωστές θέσεις και τους σωστούς ασθενείς και να περνούσε απλά από τα υπόλοιπα σημεία που δεν θα είχε σωστή αντιστοίχιση μέχρι εξάντληση της διαδρομής.

### c. Υλοποίηση με εμπόδιο στην διαδρομή

**Στο τρίτο σενάριο** έχουμε την ίδια ακριβώς πλήρη διαδρομή για το τροχοφόρο μόνο που βάζουμε πάνω στην διαδρομή ένα εμπόδιο μεταξύ των σημείων 2 έως 4 να καλύπτει την μαύρη γραμμή ακολουθίας. Αυτό θα έχει σαν επίπτωση το τροχοφόρο να παραδίδει κανονικά στο σημείο 1. Αφού φτάσει χωρίς εμπόδιο θα παραδώσει και στο σημείο 2 και θα συνεχίσει στο επόμενο. Αν διαπιστώσει καθ' οδόν ότι έχει εμπόδιο θα σταματήσει και θα ενεργοποιήσει ένα **κόκκινο LED** και **Buzzer** για ηχητικό σήμα εμποδίου.

Αυτό το εμπόδιο μπορεί να είναι κάποιος που περπατάει πάνω στην διαδρομή ή κάποιο αντικείμενο που βρίσκετε πάνω στην διαδρομή.

Ολοκληρώσαμε επιτυχώς τα πιθανά σενάρια σε αυτό το σημείο και θεωρούμε ότι έχουμε εξαντλήσει τις πιθανές παραλλαγές.



## Κεφάλαιο 7<sup>ο</sup>

Στο 7<sup>ο</sup> κεφάλαιο περιγράφονται μελλοντικές επεκτάσεις του συστήματος, αναφορά μειονεκτημάτων και συνολική αξιολόγηση της εργασίας που θα μπορούσε να έχει η παρούσα εφαρμογή.

### Μελλοντικές Επεκτάσεις:

- Οι μελλοντικές επεκτάσεις του συστήματος θα μπορούσαν να είναι να αναπτυχθεί επίσης ο αλγόριθμος για την πλήρη αποφυγή των εμποδίων όπου παρουσιάζονται κατά την διαδρομή.
- Επίσης θα μπορούσε η ίδια η διαδρομή να ήταν πιο εκτενή όχι μόνο στον ίδιο όροφο ή επίπεδο αλλά και να άλλαζε επίπεδο ή να κινούταν σε περισσότερους ορόφους.
- Άλλη επέκταση θα ήταν να μην είναι μόνο για φάρμακα αλλά να μεταφέρει και επι πλέον άλλα στοιχεία όπως φαΐ.
- Μπορούμε να το έχουμε σε εφαρμογή σε οποιονδήποτε επαγγελματικό κλάδο.

### Μειονεκτήματα:

Τα μειονεκτήματα της παρούσας εργασίας ήταν ότι το Raspberry Pi χρησιμοποιήθηκε ως ένας απλός μικροεπεξεργαστής / μικροϋπολογιστής. Δεν δόθηκε η απαιτούμενη προσοχή ούτε ήταν αντικείμενο να εμπλουτιστεί το όλο εγχείρημα με ότι θα μπορούσε να έκανε. Πρέπει να ξέρει κάποιος που θα θέλει να υλοποιήσει αρκετά καλά γλώσσα προγραμματισμού και στοιχειώδες ηλεκτρονικά.

### Πλεονεκτήματα:

Στα πλεονεκτήματα θα πρέπει να πούμε ότι η κατασκευή είναι αρκετά φτηνή, εύκολα υλοποιήσιμη με εύκολο εύρετα υλικά. Μπορούμε να χρησιμοποιήσουμε υπάρχουσες υποδομές και τεχνικές και να τις προσαρμόσουμε πάνω σε αυτές. Ο ίδιος ο Raspberry Pi δίνει τις πολλές έως άπειρες ιδιότητες δικτύωσης και εφαρμογής.

## Κεφάλαιο 8<sup>ο</sup>

Τέλος στο 8<sup>ο</sup> κεφάλαιο θα έχουμε μια αναφορά στα Συμπεράσματα.

Συμπεράσματα:

- Απλή εγκατάσταση και συντήρηση της κατασκευής και του εξοπλισμού
- Εύκολη τυποποίηση σε οποιαδήποτε εφαρμογή
- Άμεση αλλαγή στην αρχιτεκτονική του Raspberry Pi με την με χρήση μιας SD κάρτας και του αντίστοιχου λειτουργικού υλοποίησης
- Εύκολη αναβάθμιση και χρήση “παλιών ή εν μέρη υλοποιήσεων”
- Χαμηλό κόστος κατασκευής και εγκατάστασης
- Δυνατότητα εμπλουτισμού με επι πλέον αισθητήρια και στοιχεία
- Αποκαλύπτει πλευρές του υλικού και τρόπους επικοινωνίας με άλλες εφαρμογές
- Ενθαρρύνει την δημιουργικότητα

## Κεφάλαιο 9<sup>ο</sup>

### 9.1 Βιβλιογραφία

Η πτυχιακή ολοκληρώνετε με το 9<sup>ο</sup> κεφάλαιο το οποίο περιλαμβάνει την Βιβλιογραφία, Πηγές και τεχνικά φυλλάδια των χρησιμοποιημένων συσκευών και αισθητηρίων.

Επίσης σε ηλεκτρονική μορφή διαθέσιμα στο DVD

- <https://pinout.xyz/#> , Complete GPIO pinout description
- Controlling Servo Motors with Raspberry Pi, You Tube
- Raspberry Pi with Linux LESSON 1\_ Introduction to the Raspberry Pi, You Tube
- Raspberry Pi Linux LESSON 18\_ Remotely Connect from Windows with Putty SSH Telnet Client, You Tube
- Raspberry Pi Linux LESSON 24\_ Running Python on the Raspberry Pi, You Tube
- Raspberry Pi Linux LESSON 25\_ GPIO Pinout for the Raspberry Pi 2, You Tube
- Raspberry Pi Linux LESSON 26\_ Controlling the Raspberry Pi GPIO pins from Python, You Tube
- Raspberry Pi Linux LESSON 27\_ PWM Output on GPIO Pins from Python, You Tube
- Raspberry Pi LESSON 28. Controlling a Servo with Raspberry Pi and Python, You Tube
- Raspberry Pi LESSON 29\_ Using GPIO pins as Inputs and Reading them in Python, You Tube
- Raspberry Pi LESSON 32\_ Analog Input for the Raspberry Pi, You Tube
- Raspberry Pi 2 - Magnetic Switch and LED light demo, You Tube video
- Raspberry Pi Zero\_ GPIO Pins
- Raspberry Pi\_ Using GPIO Inputs
- RFID RC522 (Raspberry Pi)
- PN532 NFC RFID MODULE v3
- Review of Raspberry PI Zero

Τα DATA SHEETS και MANUALS περιλαμβάνονται παρακάτω και έχουν αριθμηθεί παρακάτω με σελίδες από 1-156.

L298 Driver motor	2
L298N-module-information	16
IR sensor	21
MFRC522 Contactless reader IC	26
Raspberry PI	120
SERVO MOTOR SG90	DATA
SHEET	156

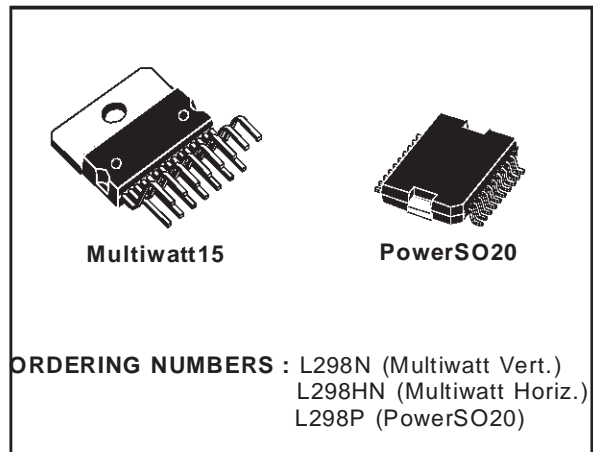


## DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)

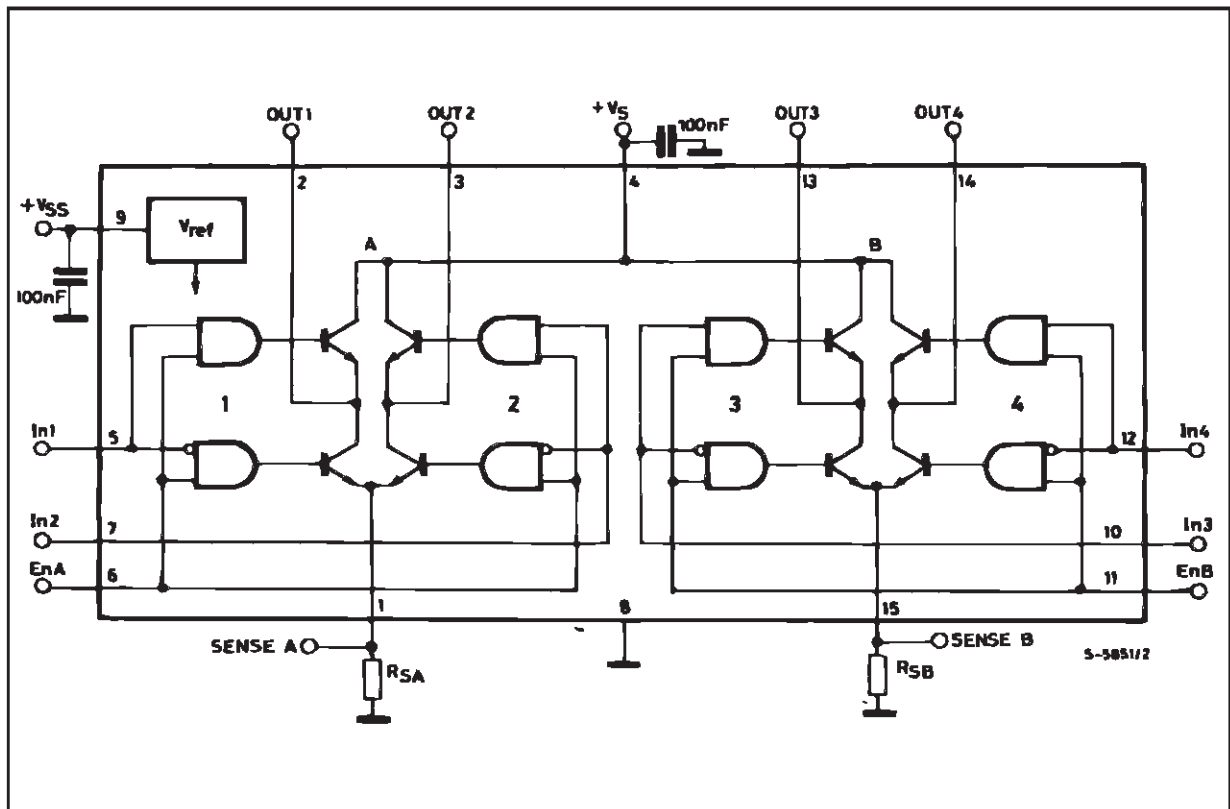
### DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the con-



nection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

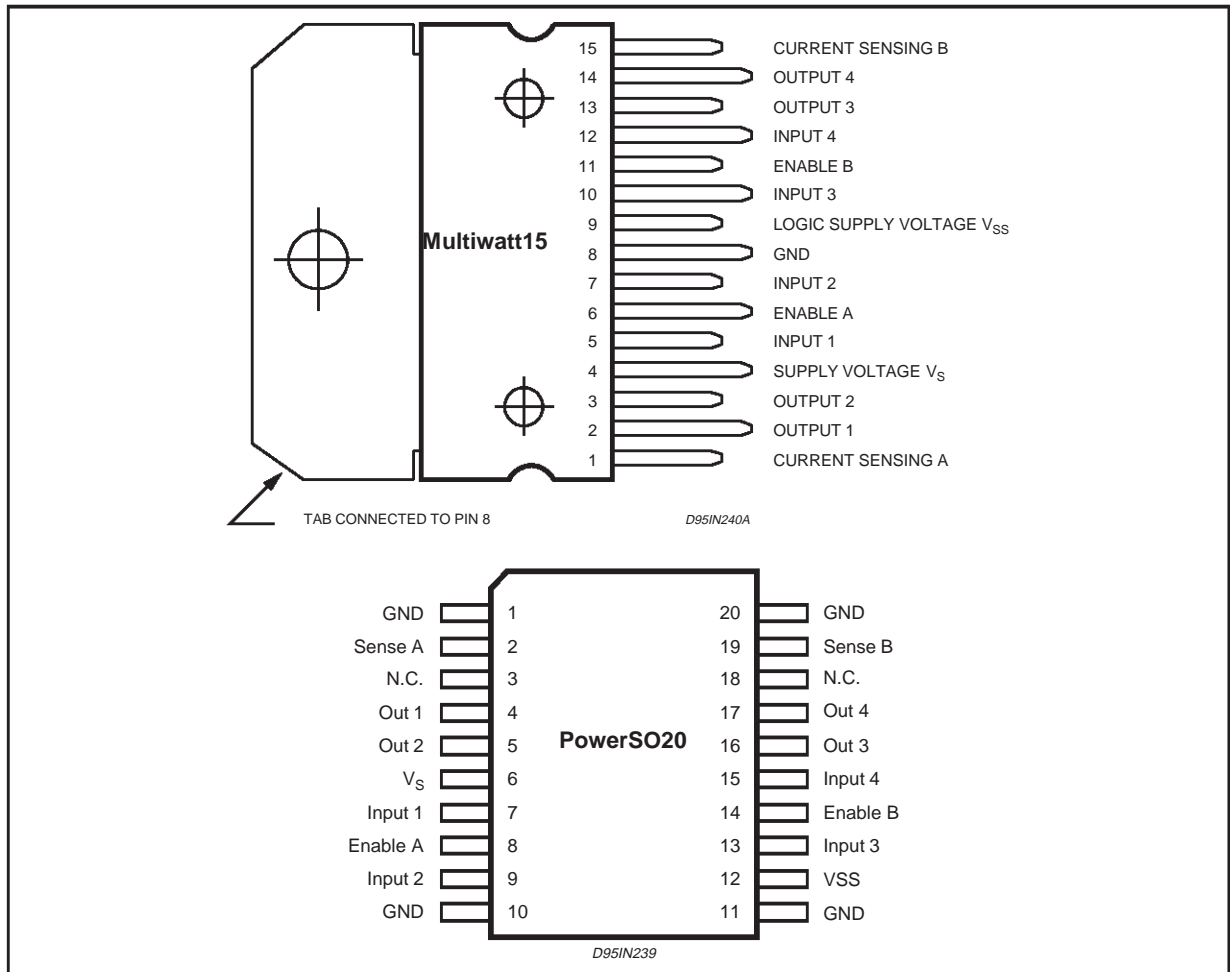
### BLOCK DIAGRAM



**ABSOLUTE MAXIMUM RATINGS**

Symbol	Parameter	Value	Unit
$V_S$	Power Supply	50	V
$V_{SS}$	Logic Supply Voltage	7	V
$V_I, V_{en}$	Input and Enable Voltage	-0.3 to 7	V
$I_O$	Peak Output Current (each Channel)		
	- Non Repetitive ( $t = 100\mu s$ )	3	A
	- Repetitive (80% on -20% off; $t_{on} = 10ms$ )	2.5	A
	-DC Operation	2	A
$V_{sens}$	Sensing Voltage	-1 to 2.3	V
$P_{tot}$	Total Power Dissipation ( $T_{case} = 75^\circ C$ )	25	W
$T_{op}$	Junction Operating Temperature	-25 to 130	$^\circ C$
$T_{stg}, T_j$	Storage and Junction Temperature	-40 to 150	$^\circ C$

**PIN CONNECTIONS (top view)**



**THERMAL DATA**

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th j-case}$	Thermal Resistance Junction-case	Max.	-	3	$^\circ C/W$
$R_{th j-amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	$^\circ C/W$

(\*) Mounted on aluminum substrate

## PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V <sub>S</sub>	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	V <sub>SS</sub>	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V<sub>S</sub> = 42V; V<sub>SS</sub> = 5V, T<sub>j</sub> = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V <sub>S</sub>	Supply Voltage (pin 4)	Operative Condition	V <sub>IH</sub> +2.5		46	V
V <sub>SS</sub>	Logic Supply Voltage (pin 9)		4.5	5	7	V
I <sub>S</sub>	Quiescent Supply Current (pin 4)	V <sub>en</sub> = H; I <sub>L</sub> = 0 V <sub>i</sub> = L V <sub>i</sub> = H		13 50	22 70	mA mA
		V <sub>en</sub> = L V <sub>i</sub> = X			4	mA
I <sub>SS</sub>	Quiescent Current from V <sub>SS</sub> (pin 9)	V <sub>en</sub> = H; I <sub>L</sub> = 0 V <sub>i</sub> = L V <sub>i</sub> = H		24 7	36 12	mA mA
		V <sub>en</sub> = L V <sub>i</sub> = X			6	mA
V <sub>iL</sub>	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V <sub>iH</sub>	Input High Voltage (pins 5, 7, 10, 12)		2.3		V <sub>SS</sub>	V
I <sub>iL</sub>	Low Voltage Input Current (pins 5, 7, 10, 12)	V <sub>i</sub> = L			–10	μA
I <sub>iH</sub>	High Voltage Input Current (pins 5, 7, 10, 12)	V <sub>i</sub> = H ≤ V <sub>SS</sub> –0.6V		30	100	μA
V <sub>en</sub> = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V <sub>en</sub> = H	Enable High Voltage (pins 6, 11)		2.3		V <sub>SS</sub>	V
I <sub>en</sub> = L	Low Voltage Enable Current (pins 6, 11)	V <sub>en</sub> = L			–10	μA
I <sub>en</sub> = H	High Voltage Enable Current (pins 6, 11)	V <sub>en</sub> = H ≤ V <sub>SS</sub> –0.6V		30	100	μA
V <sub>CEsat(H)</sub>	Source Saturation Voltage	I <sub>L</sub> = 1A I <sub>L</sub> = 2A	0.95	1.35 2	1.7 2.7	V V
V <sub>CEsat(L)</sub>	Sink Saturation Voltage	I <sub>L</sub> = 1A (5) I <sub>L</sub> = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V <sub>CEsat</sub>	Total Drop	I <sub>L</sub> = 1A (5) I <sub>L</sub> = 2A (5)	1.80		3.2 4.9	V V
V <sub>sens</sub>	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T <sub>1</sub> (V <sub>i</sub> )	Source Current Turn-off Delay	0.5 V <sub>i</sub> to 0.9 I <sub>L</sub> (2); (4)		1.5		μs
T <sub>2</sub> (V <sub>i</sub> )	Source Current Fall Time	0.9 I <sub>L</sub> to 0.1 I <sub>L</sub> (2); (4)		0.2		μs
T <sub>3</sub> (V <sub>i</sub> )	Source Current Turn-on Delay	0.5 V <sub>i</sub> to 0.1 I <sub>L</sub> (2); (4)		2		μs
T <sub>4</sub> (V <sub>i</sub> )	Source Current Rise Time	0.1 I <sub>L</sub> to 0.9 I <sub>L</sub> (2); (4)		0.7		μs
T <sub>5</sub> (V <sub>i</sub> )	Sink Current Turn-off Delay	0.5 V <sub>i</sub> to 0.9 I <sub>L</sub> (3); (4)		0.7		μs
T <sub>6</sub> (V <sub>i</sub> )	Sink Current Fall Time	0.9 I <sub>L</sub> to 0.1 I <sub>L</sub> (3); (4)		0.25		μs
T <sub>7</sub> (V <sub>i</sub> )	Sink Current Turn-on Delay	0.5 V <sub>i</sub> to 0.9 I <sub>L</sub> (3); (4)		1.6		μs
T <sub>8</sub> (V <sub>i</sub> )	Sink Current Rise Time	0.1 I <sub>L</sub> to 0.9 I <sub>L</sub> (3); (4)		0.2		μs
f <sub>c</sub> (V <sub>i</sub> )	Commutation Frequency	I <sub>L</sub> = 2A		25	40	KHz
T <sub>1</sub> (V <sub>en</sub> )	Source Current Turn-off Delay	0.5 V <sub>en</sub> to 0.9 I <sub>L</sub> (2); (4)		3		μs
T <sub>2</sub> (V <sub>en</sub> )	Source Current Fall Time	0.9 I <sub>L</sub> to 0.1 I <sub>L</sub> (2); (4)		1		μs
T <sub>3</sub> (V <sub>en</sub> )	Source Current Turn-on Delay	0.5 V <sub>en</sub> to 0.1 I <sub>L</sub> (2); (4)		0.3		μs
T <sub>4</sub> (V <sub>en</sub> )	Source Current Rise Time	0.1 I <sub>L</sub> to 0.9 I <sub>L</sub> (2); (4)		0.4		μs
T <sub>5</sub> (V <sub>en</sub> )	Sink Current Turn-off Delay	0.5 V <sub>en</sub> to 0.9 I <sub>L</sub> (3); (4)		2.2		μs
T <sub>6</sub> (V <sub>en</sub> )	Sink Current Fall Time	0.9 I <sub>L</sub> to 0.1 I <sub>L</sub> (3); (4)		0.35		μs
T <sub>7</sub> (V <sub>en</sub> )	Sink Current Turn-on Delay	0.5 V <sub>en</sub> to 0.9 I <sub>L</sub> (3); (4)		0.25		μs
T <sub>8</sub> (V <sub>en</sub> )	Sink Current Rise Time	0.1 I <sub>L</sub> to 0.9 I <sub>L</sub> (3); (4)		0.1		μs

- 1) Sensing voltage can be -1 V for t ≤ 50 μsec; in steady state V<sub>sens</sub> min ≥ -0.5 V.
- 2) See fig. 2.
- 3) See fig. 4.
- 4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current.

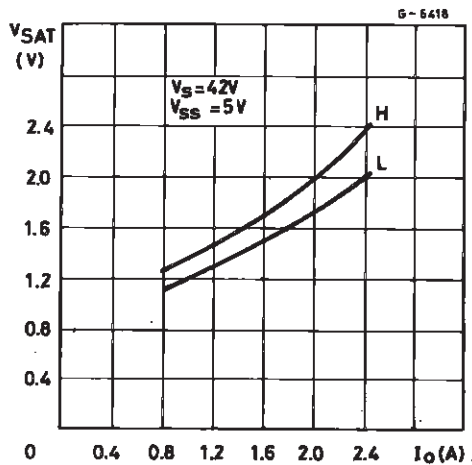
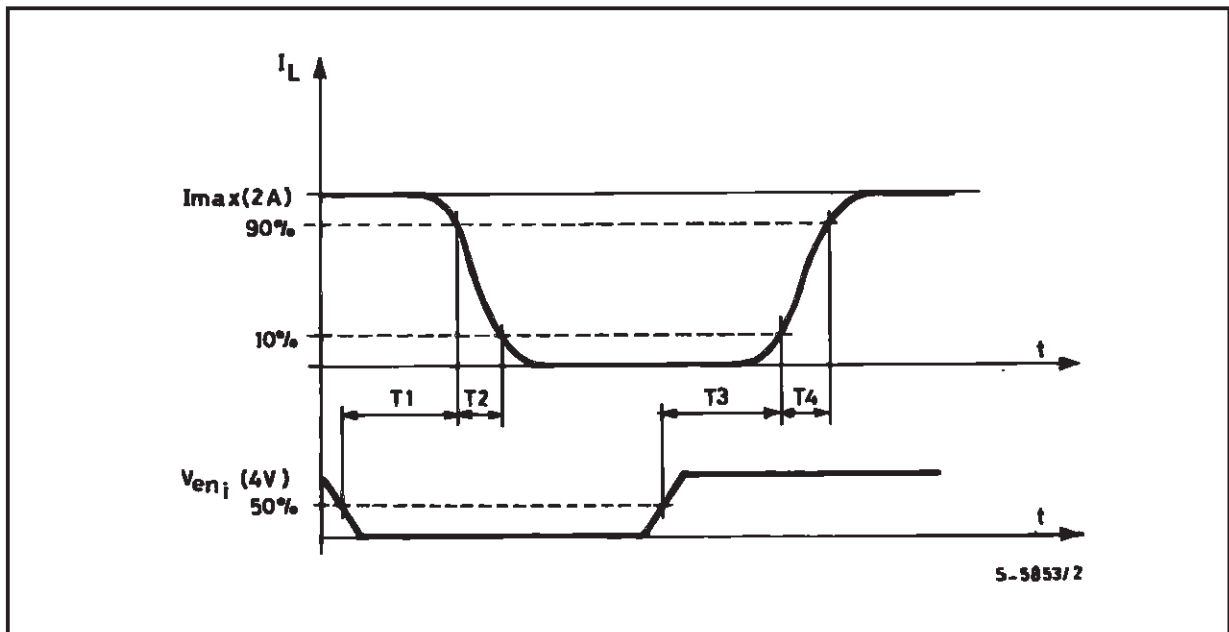


Figure 2 : Switching Times Test Circuits.

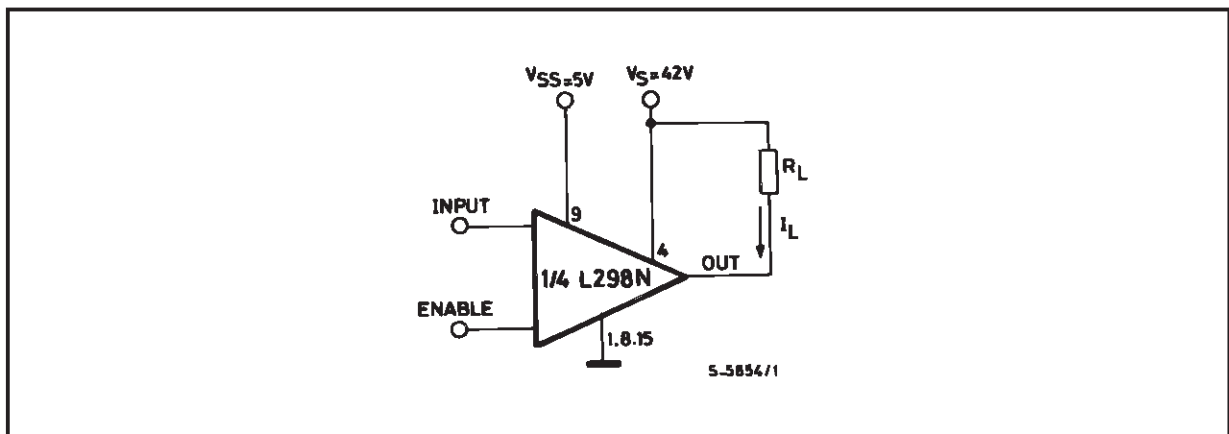


Note: For INPUT Switching, set EN = H  
 For ENABLE Switching, set IN = H

**Figure 3 :** Source Current Delay Times vs. Input or Enable Switching.



**Figure 4 :** Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H  
For ENABLE Switching, set IN = L

Figure 5 : Sink Current Delay Times vs. Input 0 V Enable Switching.

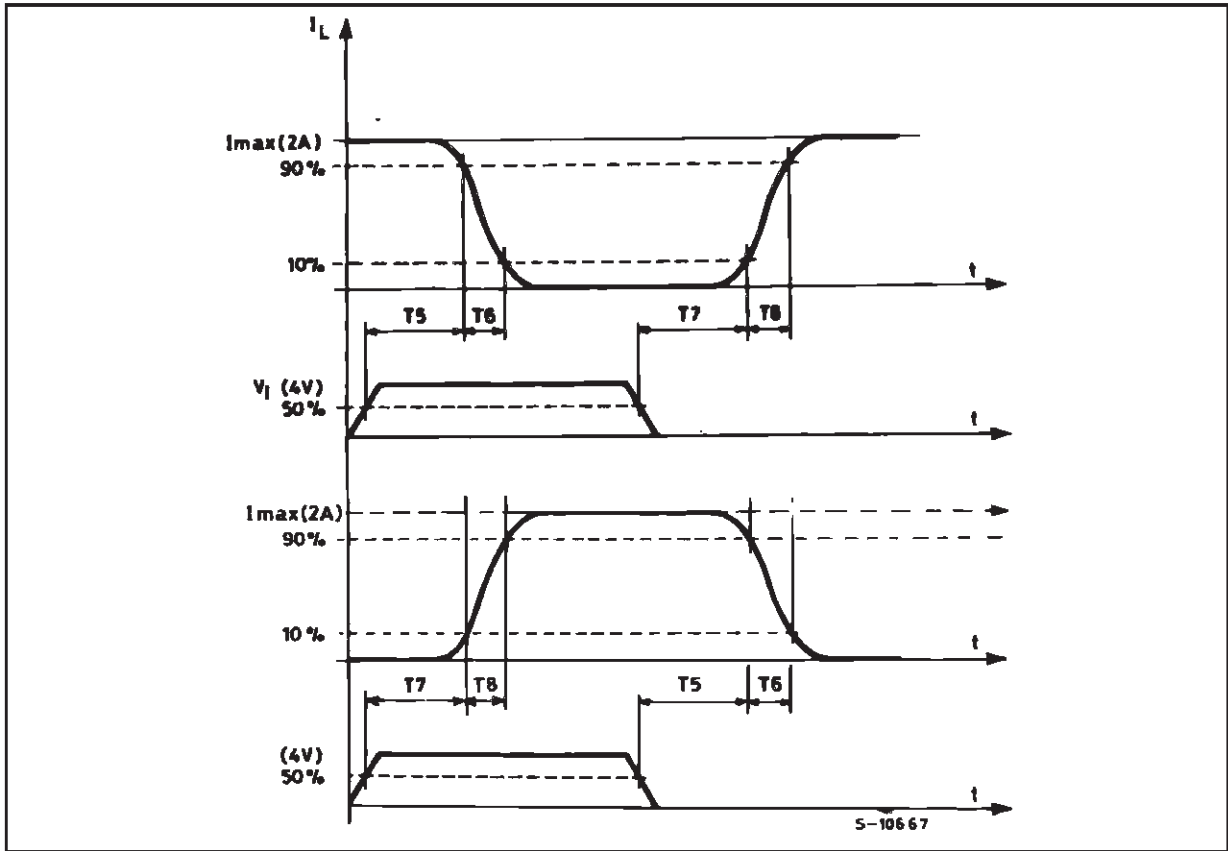
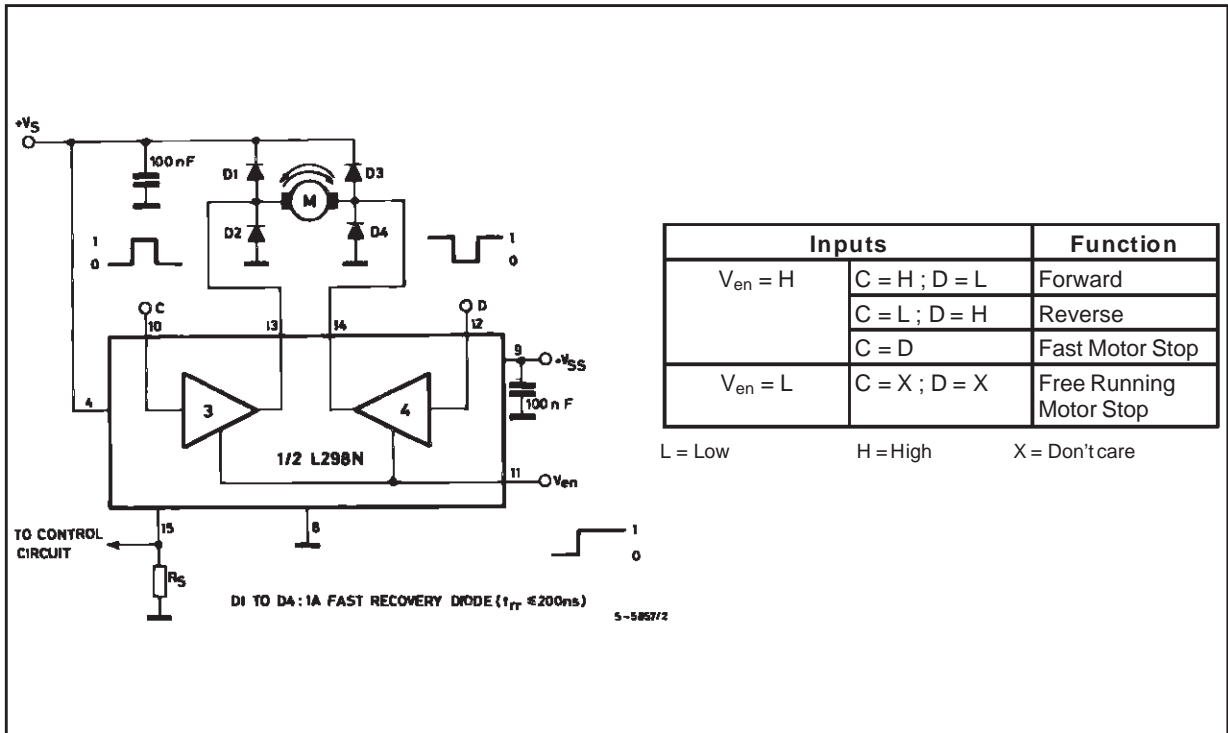
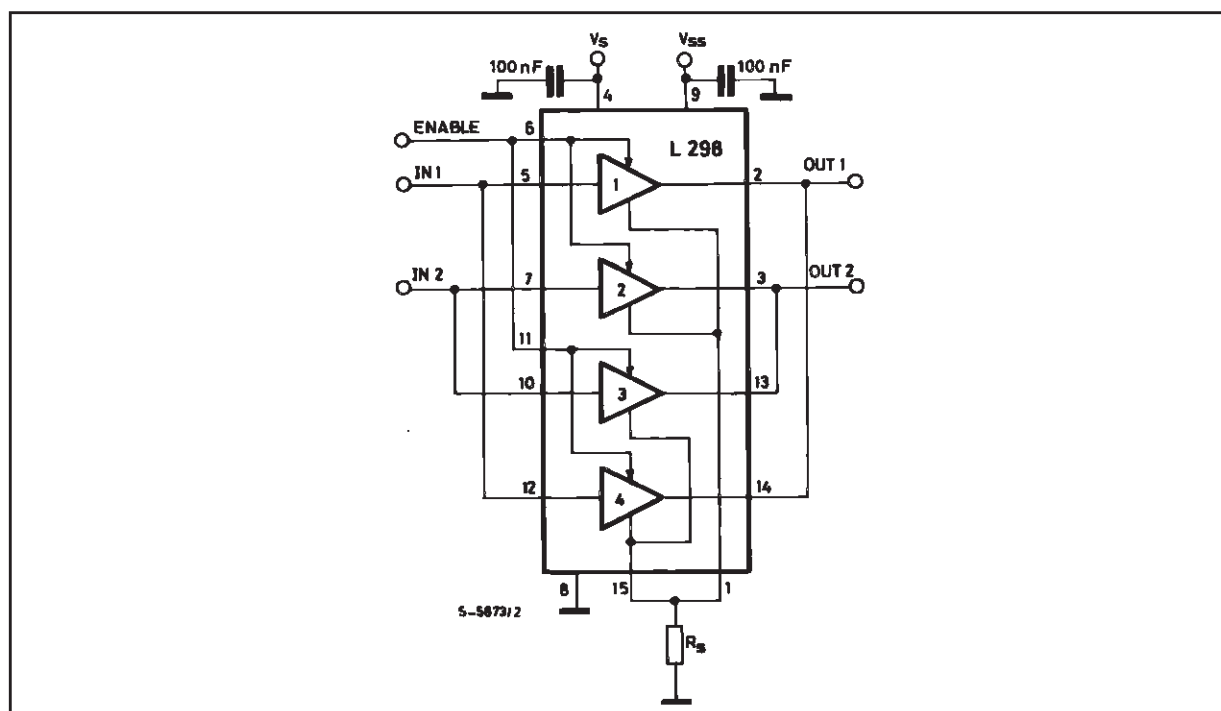


Figure 6 : Bidirectional DC Motor Control.



**Figure 7** : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



## APPLICATION INFORMATION (Refer to the block diagram)

### 1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differential mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output: an external resistor ( $R_{SA}$ ;  $R_{SB}$ ) allows to detect the intensity of this current.

### 1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are  $In1$ ;  $In2$ ;  $EnA$  and  $In3$ ;  $In4$ ;  $EnB$ . The  $In$  inputs set the bridge state when The  $En$  input is high; a low state of the  $En$  input inhibits the bridge. All the inputs are TTL compatible.

### 2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both  $V_s$  and  $V_{ss}$ , to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of  $V_s$  that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

### 3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes D1 to D4 is made by four fast recovery elements ( $t_{tr} \leq 200$  nsec) that must be chosen of a  $V_F$  as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped; Schottky diodes would be preferred.



This solution can drive until 3 Amps In DC operation and until 3.5 Amps of a repetitive peak current.

On Fig 8 it is shown the driving of a two phase bipolar stepper motor ; the needed signals to drive the inputs of the L298 are generated, in this example, from the IC L297.

Fig 9 shows an example of P.C.B. designed for the application of Fig 8.

**Figure 8** : Two Phase Bipolar Stepper Motor Circuit.

This circuit drives bipolar stepper motors with winding currents up to 2 A. The diodes are fast 2 A types.

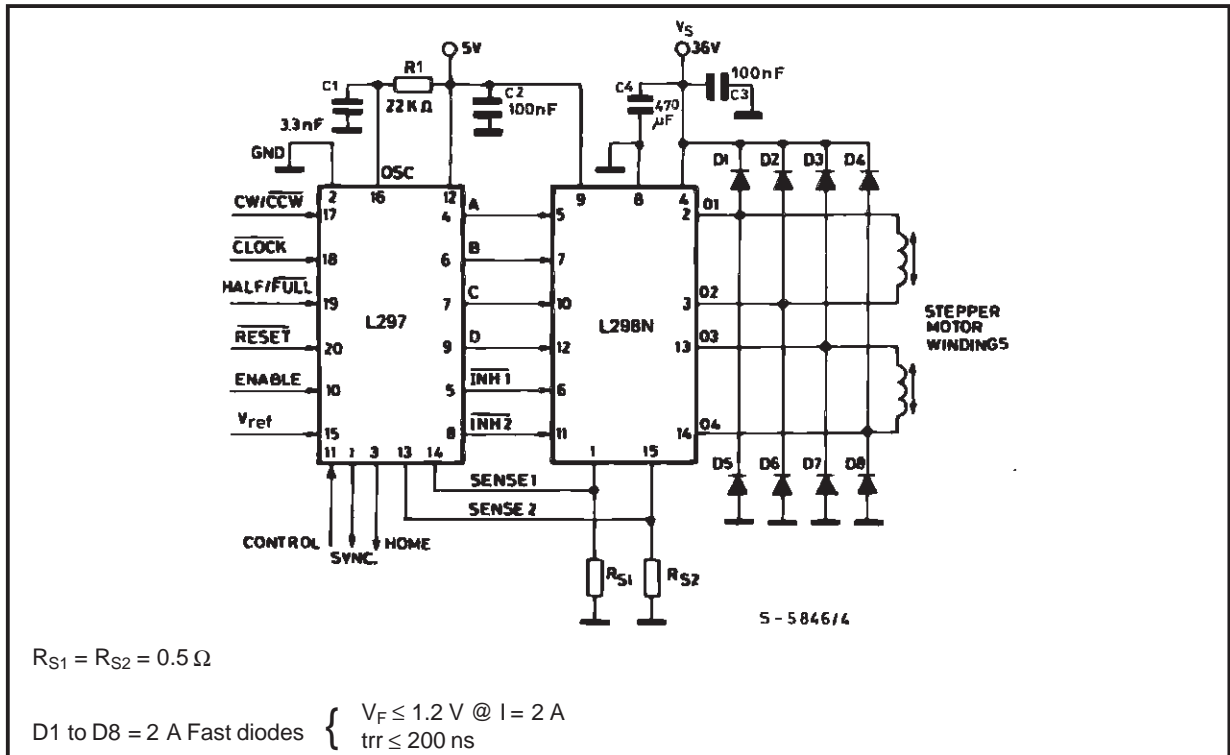


Fig 10 shows a second two phase bipolar stepper motor control circuit where the current is controlled by the I.C. L6506.

Figure 9 : Suggested Printed Circuit Board Layout for the Circuit of fig. 8 (1:1 scale).

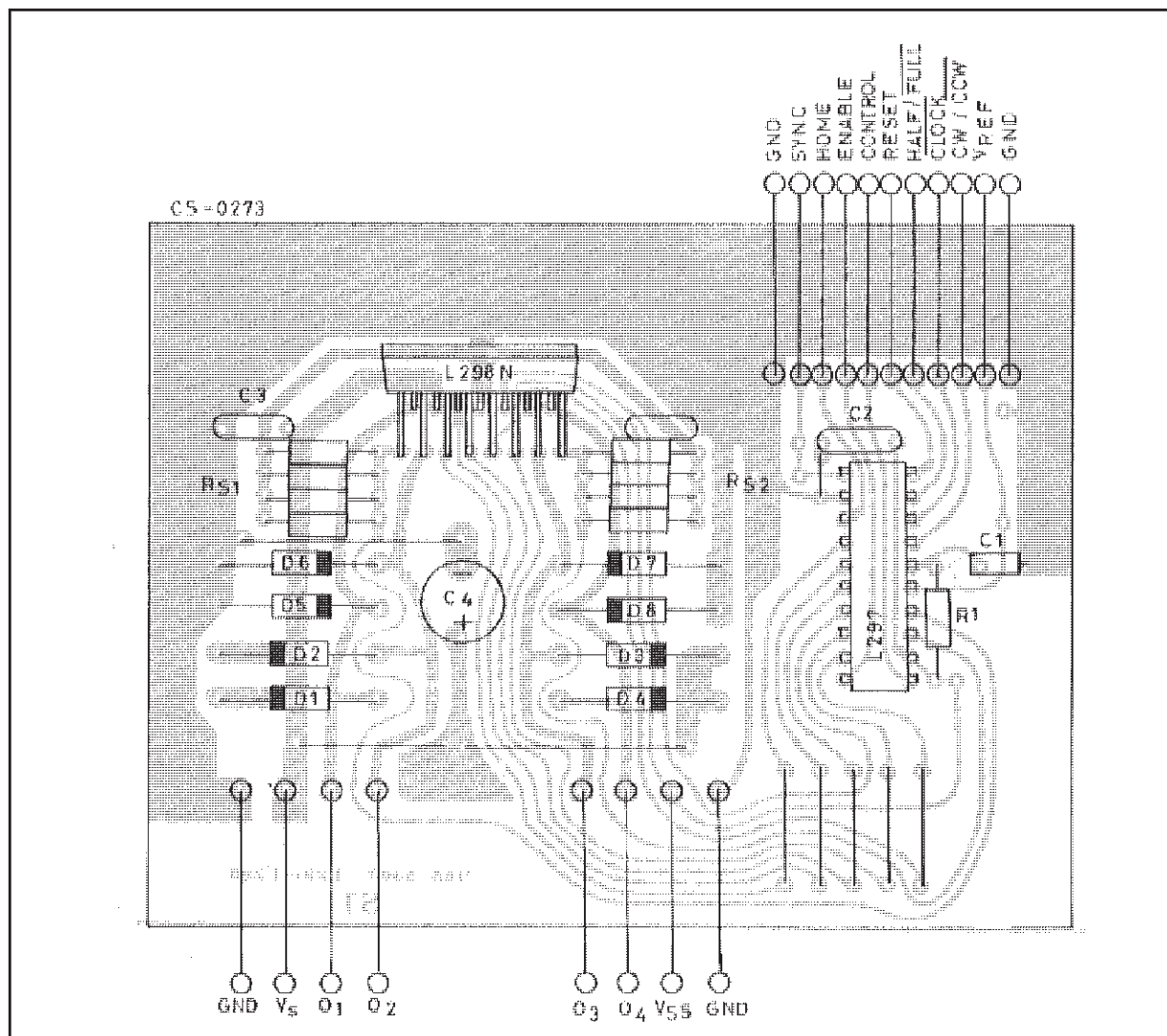
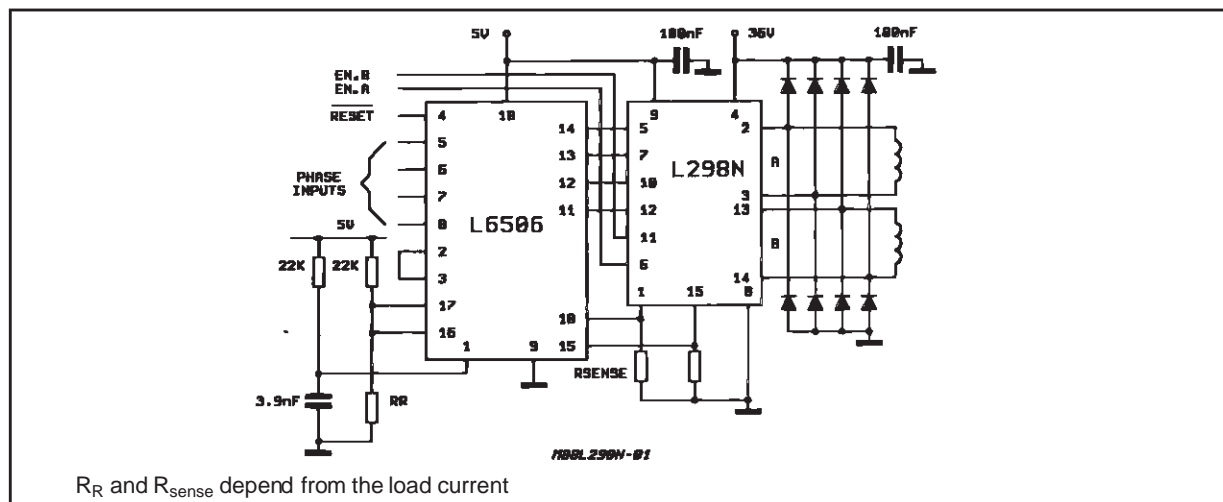
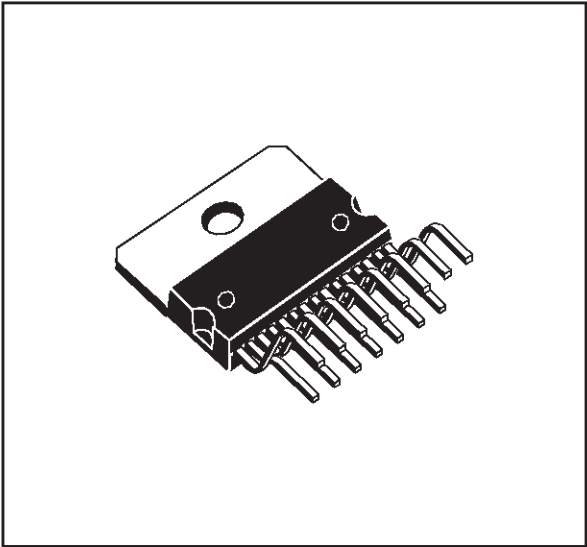


Figure 10 : Two Phase Bipolar Stepper Motor Control Circuit by Using the Current Controller L6506.

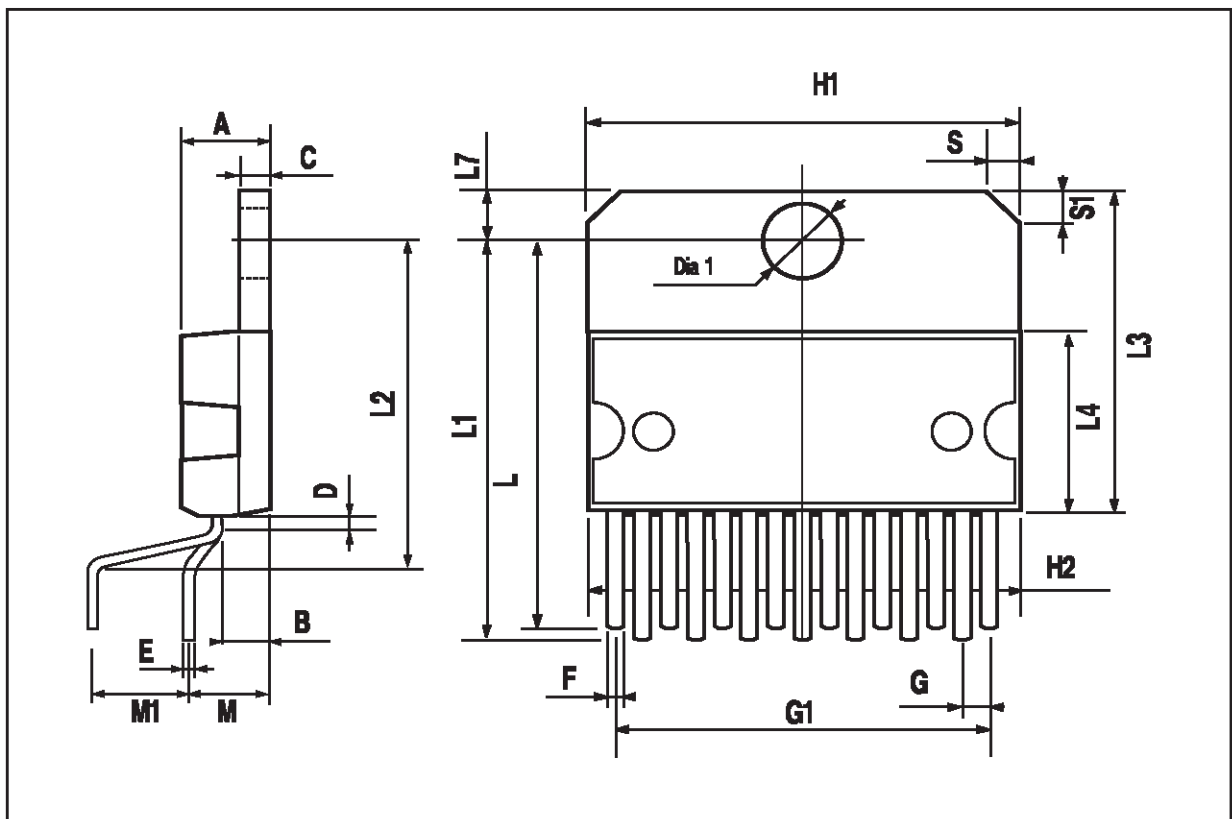


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
D		1			0.039	
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.02	1.27	1.52	0.040	0.050	0.060
G1	17.53	17.78	18.03	0.690	0.700	0.710
H1	19.6			0.772		
H2			20.2			0.795
L	21.9	22.2	22.5	0.862	0.874	0.886
L1	21.7	22.1	22.5	0.854	0.870	0.886
L2	17.65		18.1	0.695		0.713
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L7	2.65		2.9	0.104		0.114
M	4.25	4.55	4.85	0.167	0.179	0.191
M1	4.63	5.08	5.53	0.182	0.200	0.218
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

**OUTLINE AND MECHANICAL DATA**

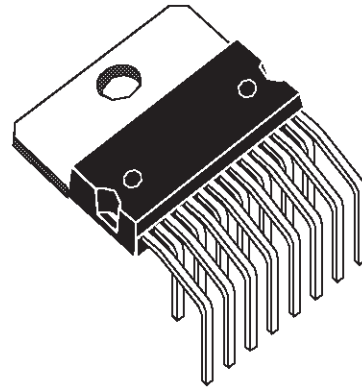


**Multiwatt15 V**

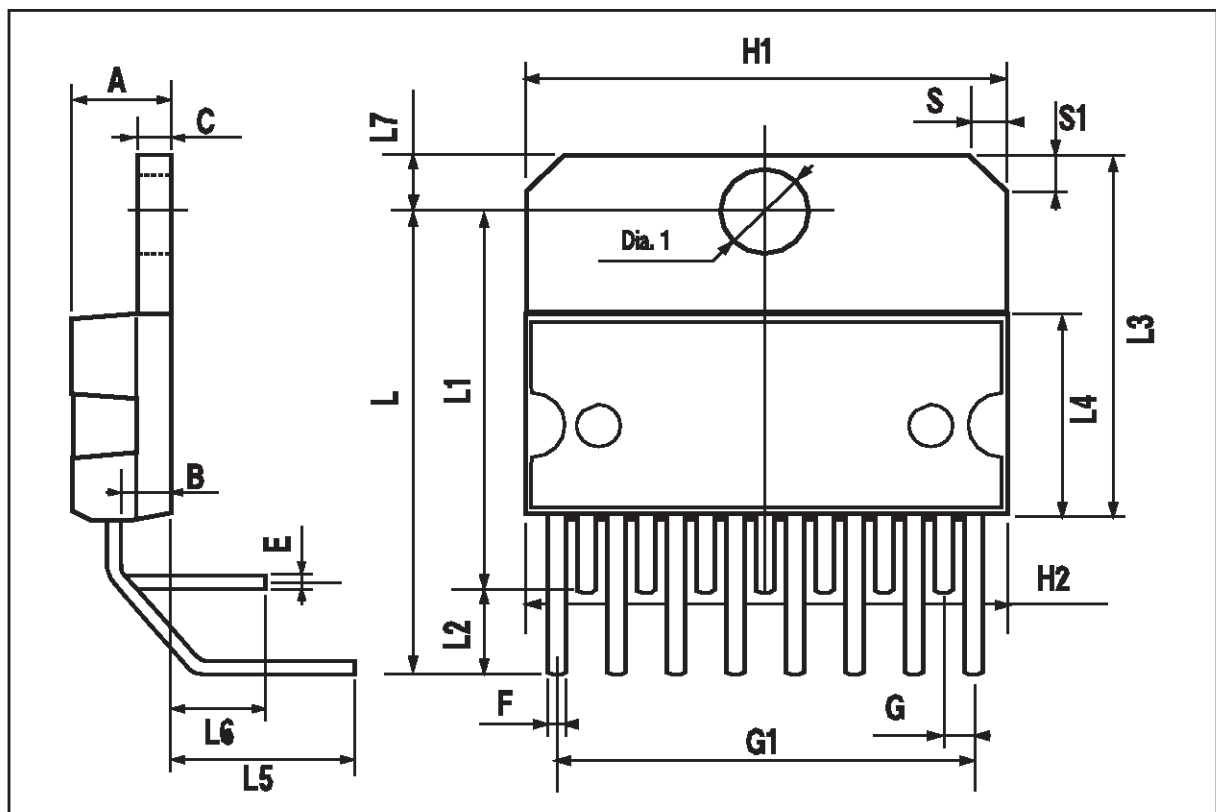


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.14	1.27	1.4	0.045	0.050	0.055
G1	17.57	17.78	17.91	0.692	0.700	0.705
H1	19.6			0.772		
H2			20.2			0.795
L		20.57			0.810	
L1		18.03			0.710	
L2		2.54			0.100	
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L5		5.28			0.208	
L6		2.38			0.094	
L7	2.65		2.9	0.104		0.114
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

## OUTLINE AND MECHANICAL DATA



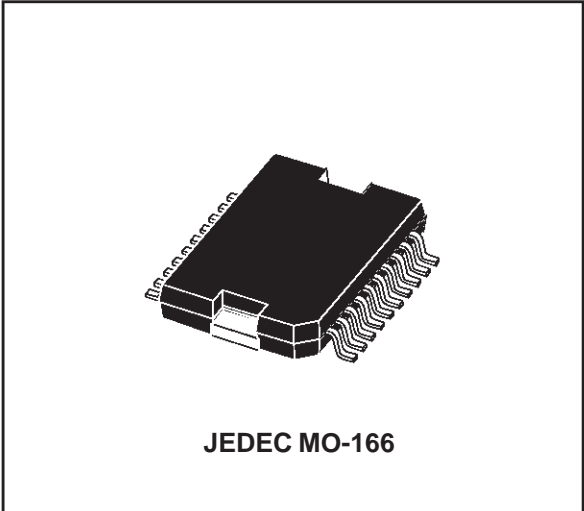
**Multiwatt15 H**



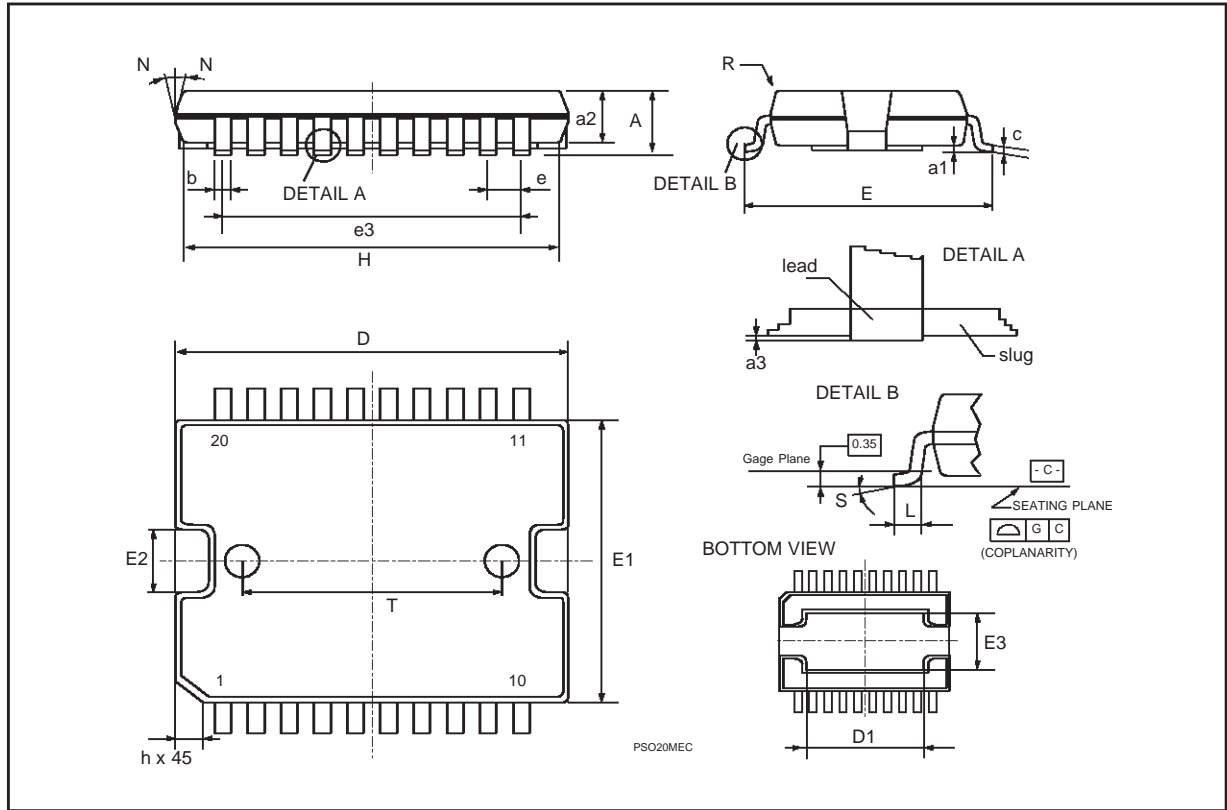
DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			3.6			0.142
a1	0.1		0.3	0.004		0.012
a2			3.3			0.130
a3	0		0.1	0.000		0.004
b	0.4		0.53	0.016		0.021
c	0.23		0.32	0.009		0.013
D (1)	15.8		16	0.622		0.630
D1	9.4		9.8	0.370		0.386
E	13.9		14.5	0.547		0.570
e		1.27			0.050	
e3		11.43			0.450	
E1 (1)	10.9		11.1	0.429		0.437
E2			2.9			0.114
E3	5.8		6.2	0.228		0.244
G	0		0.1	0.000		0.004
H	15.5		15.9	0.610		0.626
h			1.1			0.043
L	0.8		1.1	0.031		0.043
N	10° (max.)					
S	8° (max.)					
T		10			0.394	

(1) "D and F" do not include mold flash or protrusions.  
 - Mold flash or protrusions shall not exceed 0.15 mm (0.006").  
 - Critical dimensions: "E", "G" and "a3"

**OUTLINE AND MECHANICAL DATA**



**PowerSO20**



Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specification mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without express written approval of STMicroelectronics.

The ST logo is a registered trademark of STMicroelectronics  
© 2000 STMicroelectronics – Printed in Italy – All Rights Reserved  
STMicroelectronics GROUP OF COMPANIES

Australia - Brazil - China - Finland - France - Germany - Hong Kong - India - Italy - Japan - Malaysia - Malta - Morocco -  
Singapore - Spain - Sweden - Switzerland - United Kingdom - U.S.A.

<http://www.st.com>

This datasheet has been download from:

[www.datasheetcatalog.com](http://www.datasheetcatalog.com)

Datasheets for electronics components.

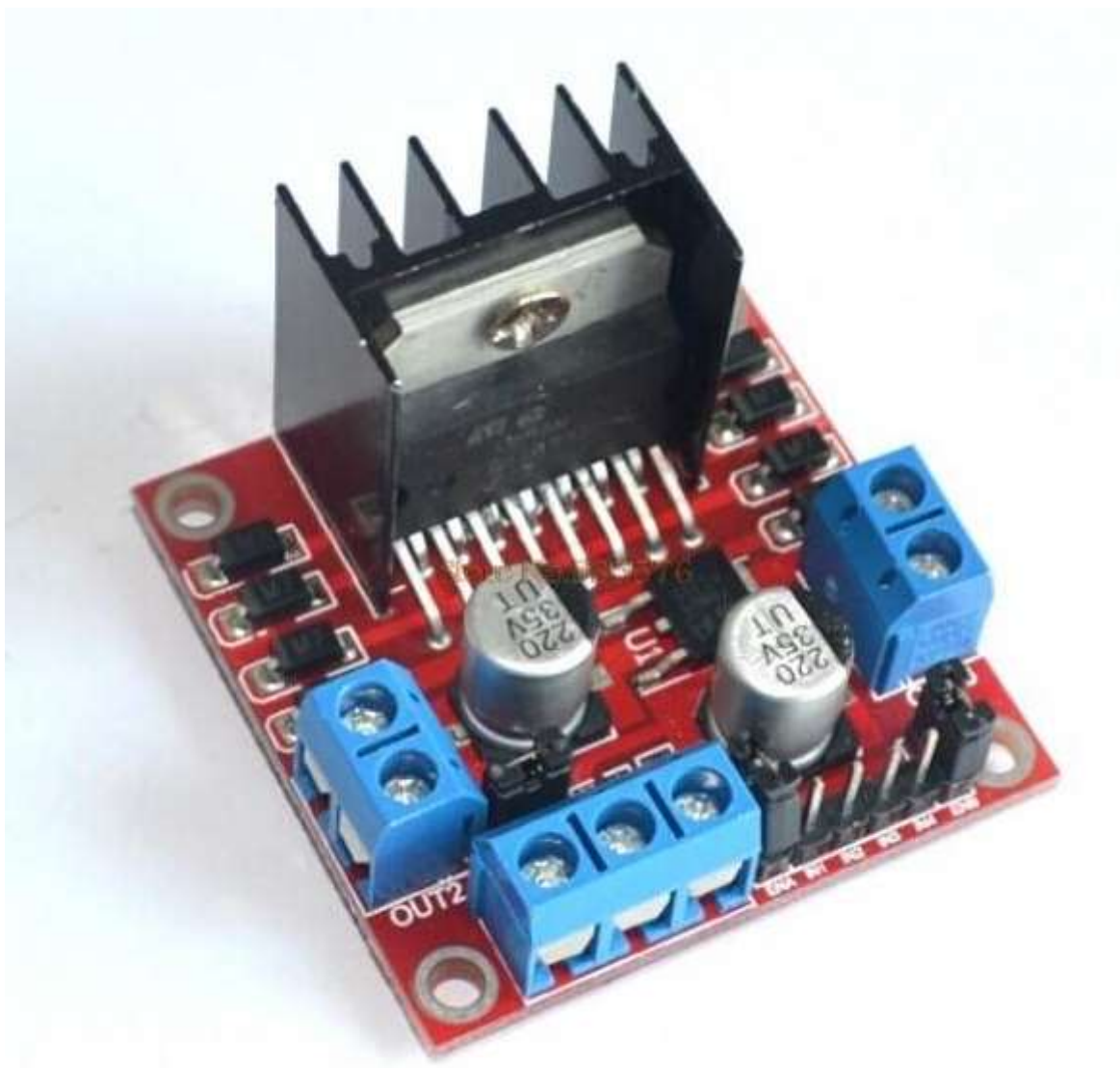


# Tutorial - L298N Dual Motor Controller Module 2A and Arduino

In this tutorial we'll explain how to use our L298N H-bridge Dual Motor Controller Module 2A with Arduino. This allows you to control the speed and direction of two DC motors, or control one bipolar stepper motor with ease. The L298N H-bridge module can be used with motors that have a voltage of between 5 and 35V DC.

There is also an onboard 5V regulator, so if your supply voltage is up to 12V you can also source 5V from the board.

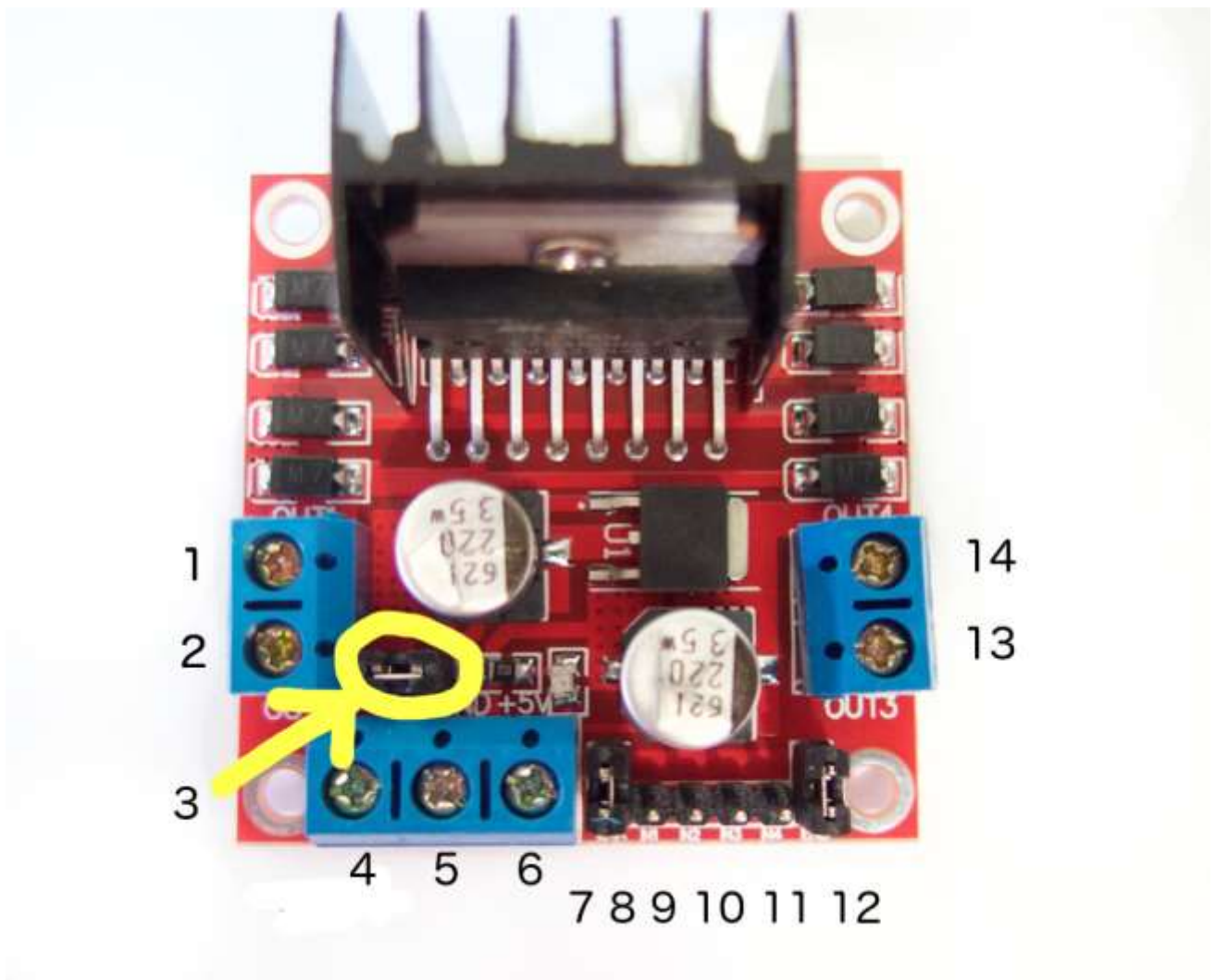
*So let's get started!*



First we'll run through the connections, then explain how to control DC motors then a stepper motor.

## Module pinouts

Consider the following image - match the numbers against the list below the image:



1. DC motor 1 "+" or stepper motor A+
2. DC motor 1 "-" or stepper motor A-
3. 12V jumper - remove this if using a supply voltage greater than 12V DC. This enables power to the onboard 5V regulator
4. Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5. GND
6. 5V output if 12V jumper in place, ideal for powering your Arduino (etc)
7. DC motor 1 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
8. IN1
9. IN2
10. IN3
11. IN4
12. DC motor 2 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.

13. DC motor 2 "+" or stepper motor B+
14. DC motor 2 "-" or stepper motor B-

## Controlling DC Motors

To control one or two DC motors is quite easy. First connect each motor to the A and B connections on the L298N module. If you're using two motors for a robot (etc) ensure that the polarity of the motors is the same on both inputs. Otherwise you may need to swap them over when you set both motors to forward and one goes backwards!

Next, connect your power supply - the positive to pin 4 on the module and negative/GND to pin 5. If your supply is up to 12V you can leave in the 12V jumper (point 3 in the image above) and 5V will be available from pin 6 on the module. This can be fed to your Arduino's 5V pin to power it from the motors' power supply. Don't forget to connect Arduino GND to pin 5 on the module as well to complete the circuit.

Now you will need six digital output pins on your Arduino, two of which need to be PWM (pulse-width modulation) pins. PWM pins are denoted by the tilde ("~") next to the pin number, for example:



Finally, connect the Arduino digital output pins to the driver module. In our example we have two DC motors, so digital pins D9, D8, D7 and D6 will be connected to pins IN1, IN2, IN3 and IN4 respectively. Then connect D10 to module pin 7 (remove the jumper first) and D5 to module pin 12 (again, remove the jumper).

The motor direction is controlled by sending a HIGH or LOW signal to the drive for each motor (or channel). For example for motor one, a HIGH to IN1 and a LOW to IN2 will cause it to turn in one direction, and a LOW and HIGH will cause it to turn in the other direction.

However the motors will not turn until a HIGH is set to the enable pin (7 for motor one, 12 for motor two). And they can be turned off with a LOW to the same pin(s). However if you need to control the speed of the motors, the PWM signal from the digital pin connected to the enable pin can take care of it.

This is what we've done with the DC motor demonstration sketch. Two DC motors and an Arduino Uno are connected as described above, along with an external power supply. Then enter and upload the following sketch:

```
// connect motor controller pins to Arduino digital pins
// motor one
int enA = 10;
int in1 = 9;
int in2 = 8;
// motor two
int enB = 5;
int in3 = 7;
int in4 = 6;
```

```

void setup()
{
  // set all the motor control pins to outputs
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
}
void demoOne()
{
  // this function will run the motors in both directions at a fixed speed
  // turn on motor A
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  // set speed to 200 out of possible range 0~255
  analogWrite(enA, 200);
  // turn on motor B
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  // set speed to 200 out of possible range 0~255
  analogWrite(enB, 200);
  delay(2000);
  // now change motor directions
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  delay(2000);
  // now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
}
void demoTwo()
{
  // this function will run the motors across the range of possible speeds
  // note that maximum speed is determined by the motor itself and the
operating voltage
  // the PWM values sent by analogWrite() are fractions of the maximum
speed possible
  // by your hardware
  // turn on motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  // accelerate from zero to maximum speed
  for (int i = 0; i < 256; i++)
  {
    analogWrite(enA, i);
    analogWrite(enB, i);
    delay(20);
  }
  // decelerate from maximum speed to zero
  for (int i = 255; i >= 0; --i)
  {
    analogWrite(enA, i);
    analogWrite(enB, i);
  }
}

```

```
    delay(20);
  }
  // now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
}
void loop()
{
  demoOne();
  delay(1000);
  demoTwo();
  delay(1000);
}
```

So what's happening in that sketch? In the function *demoOne()* we turn the motors on and run them at a PWM value of 200. This is not a speed value, instead power is applied for 200/255 of an amount of time at once.

Then after a moment the motors operate in the reverse direction (see how we changed the HIGHS and LOWs in the *digitalWrite()* functions?).

To get an idea of the range of speed possible of your hardware, we run through the entire PWM range in the function *demoTwo()* which turns the motors on and then runs through PWM values zero to 255 and back to zero with the two *for* loops.

Finally this is demonstrated in the following video - using a well-worn tank chassis with two DC motors:

# velleman®

## VMA326

---

LINE TRACKING SENSOR TCRT5000 MODULE



USER MANUAL



# USER MANUAL

## 1. Introduction

To all residents of the European Union

### Important environmental information about this product



This symbol on the device or the package indicates that disposal of the device after its lifecycle could harm the environment. Do not dispose of the unit (or batteries) as unsorted municipal waste; it should be taken to a specialized company for recycling. This device should be returned to your distributor or to a local recycling service. Respect the local environmental rules.

**If in doubt, contact your local waste disposal authorities.**

Thank you for choosing Velleman®! Please read the manual thoroughly before bringing this device into service. If the device was damaged in transit, do not install or use it and contact your dealer.

## 2. Safety Instructions



- This device can be used by children aged from 8 years and above, and persons with reduced physical, sensory or mental capabilities or lack of experience and knowledge if they have been given supervision or instruction concerning the use of the device in a safe way and understand the hazards involved. Children shall not play with the device. Cleaning and user maintenance shall not be made by children without supervision.



- Indoor use only.  
Keep away from rain, moisture, splashing and dripping liquids.

## 3. General Guidelines



- Refer to the Velleman® Service and Quality Warranty on the last pages of this manual.
- Familiarise yourself with the functions of the device before actually using it.
- All modifications of the device are forbidden for safety reasons. Damage caused by user modifications to the device is not covered by the warranty.
- Only use the device for its intended purpose. Using the device in an unauthorised way will void the warranty.
- Damage caused by disregard of certain guidelines in this manual is not covered by the warranty and the dealer will not accept responsibility for any ensuing defects or problems.
- Nor Velleman nv nor its dealers can be held responsible for any damage (extraordinary, incidental or indirect) – of any nature (financial, physical...) arising from the possession, use or failure of this product.
- Due to constant product improvements, the actual product appearance might differ from the shown images.
- Product images are for illustrative purposes only.
- Do not switch the device on immediately after it has been exposed to changes in temperature. Protect the device against damage by leaving it switched off until it has reached room temperature.
- Keep this manual for future reference.

## 4. What is Arduino®

Arduino® is an open-source prototyping platform based in easy-to-use hardware and software. Arduino® boards are able to read inputs – light-on sensor, a finger on a button or a Twitter message – and turn it into an output – activating of a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so, you use the Arduino programming language (based on Wiring) and the Arduino® software IDE (based on Processing).

Surf to [www.arduino.cc](http://www.arduino.cc) and [www.arduino.org](http://www.arduino.org) for more information.



## 5. Overview

### VMA326

The line tracking sensor based on TCRT5000 is a type of infrared reflectance sensor, and is commonly used in line following robots, mounted at the bottom of the robot chassis. The line tracking sensor works by detecting reflected light coming from its own infrared LED and by measuring the amount of reflected infrared light, it can detect transitions from light to dark (lines) or even objects directly in front of it.

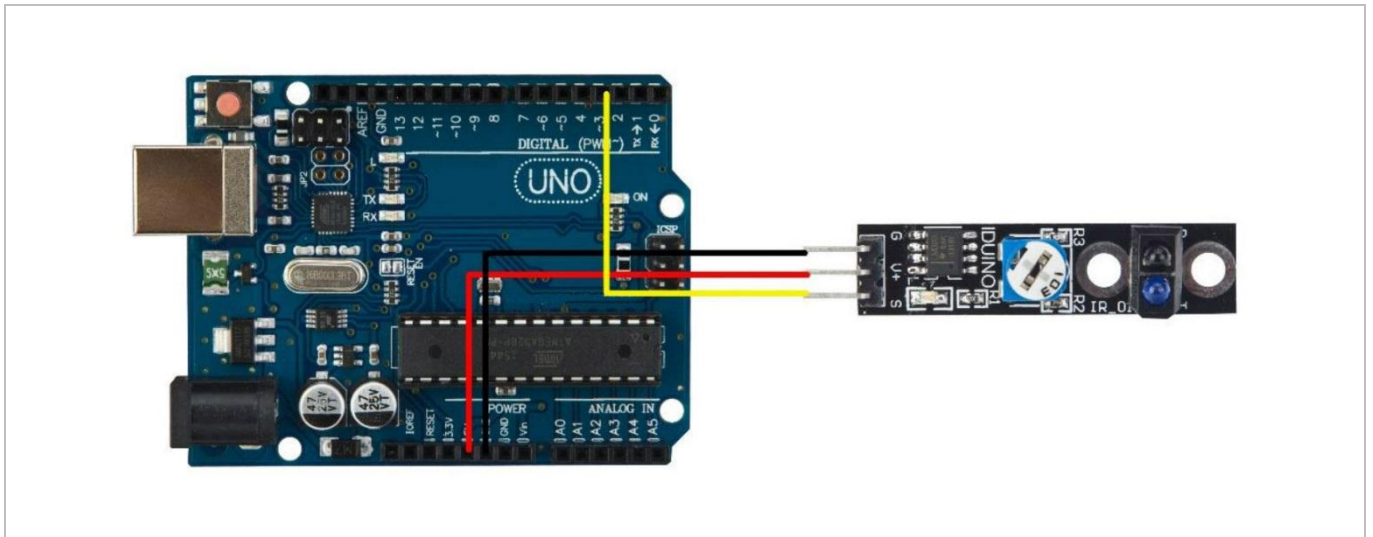
sensor adopts TCRT5000, high sensitivity  
 sensitivity adjustable by potentiometer  
 Working voltage 3.3 V to 5 V  
 operating current : 20 mA  
 digital switch output (0 and 1)  
 two fixed bolt holes for convenient installation  
 small board PCB size: 42 x 10.5 mm  
 power indicator light (red) and digital switch output indicator light (green)  
 detection reflection distance: 1-25 mm

## 6. Pin Layout

S	digital output (black = low, white = high)
V+	5 VDC power
G	ground

## 7. Example

The example shows that when the sensor detects a black area, the S pin's output is a low TTL signal. The LED13 switches off while the light L on this module switches on. On the contrary, LED13 switches on.



```

// CODE BEGIN
int Led=13; // variable Led = connected to digital pin 13 (Which is on your VMA100 connected to a LED as well)
int buttonpin=3; // Variable buttonpin = connected to digital line 3, this is where the VMA326 output has to be connected for this test
int val;
void setup()
{ pinMode(Led,OUTPUT); // declare Led (digital 13) as output
  pinMode(buttonpin,INPUT); // declare buttonpin (digital 3) as input
}
void loop()
{
  val=digitalRead(buttonpin); // read the value of buttonpin (digital 3)
  if(val==HIGH) // if this value is high ...
  { digitalWrite(Led,HIGH); } // then switch Led (digital 13) to High
  else { digitalWrite(Led,LOW); } // else .. switch to low
}
// CODE END

```

## 8. More Information

Please refer to the VMA326 product page on [www.velleman.eu](http://www.velleman.eu) for more information.

**Use this device with original accessories only. Velleman nv cannot be held responsible in the event of damage or injury resulting from (incorrect) use of this device. For more info concerning this product and the latest version of this manual, please visit our website [www.velleman.eu](http://www.velleman.eu). The information in this manual is subject to change without prior notice.**

### © COPYRIGHT NOTICE

**The copyright to this manual is owned by Velleman nv. All worldwide rights reserved.** No part of this manual may be copied, reproduced, translated or reduced to any electronic medium or otherwise without the prior written consent of the copyright holder.

# Velleman® Service and Quality Warranty

Since its foundation in 1972, Velleman® acquired extensive experience in the electronics world and currently distributes its products in over 85 countries.

All our products fulfil strict quality requirements and legal stipulations in the EU. In order to ensure the quality, our products regularly go through an extra quality check, both by an internal quality department and by specialized external organisations. If, all precautionary measures notwithstanding, problems should occur, please make appeal to our warranty (see guarantee conditions).

## General Warranty Conditions Concerning Consumer Products (for EU):

- All consumer products are subject to a 24-month warranty on production flaws and defective material as from the original date of purchase.
- Velleman® can decide to replace an article with an equivalent article, or to refund the retail value totally or partially when the complaint is valid and a free repair or replacement of the article is impossible, or if the expenses are out of proportion.

You will be delivered a replacing article or a refund at the value of 100% of the purchase price in case of a flaw occurred in the first year after the date of purchase and delivery, or a replacing article at 50% of the purchase price or a refund at the value of 50% of the retail value in case of a flaw occurred in the second year after the date of purchase and delivery.

### • Not covered by warranty:

- all direct or indirect damage caused after delivery to the article (e.g. by oxidation, shocks, falls, dust, dirt, humidity...), and by the article, as well as its contents (e.g. data loss), compensation for loss of profits;
- consumable goods, parts or accessories that are subject to an aging process during normal use, such as batteries (rechargeable, non-rechargeable, built-in or replaceable), lamps, rubber parts, drive belts... (unlimited list);
- flaws resulting from fire, water damage, lightning, accident, natural disaster, etc....;
- flaws caused deliberately, negligently or resulting from improper handling, negligent maintenance, abusive use or use contrary to the manufacturer's instructions;
- damage caused by a commercial, professional or collective use of the article (the warranty validity will be reduced to six (6) months when the article is used professionally);
- damage resulting from an inappropriate packing and shipping of the article;
- all damage caused by modification, repair or alteration performed by a third party without written permission by Velleman®.
- Articles to be repaired must be delivered to your Velleman® dealer, solidly packed (preferably in the original packaging), and be completed with the original receipt of purchase and a clear flaw description.
- Hint: In order to save on cost and time, please reread the manual and check if the flaw is caused by obvious causes prior to presenting the article for repair. Note that returning a non-defective article can also involve handling costs.
- Repairs occurring after warranty expiration are subject to shipping costs.
- The above conditions are without prejudice to all commercial warranties.

**The above enumeration is subject to modification according to the article (see article's manual).**



# MFRC522

## Contactless reader IC

Rev. 3.6 — 14 December 2011  
112136

Product data sheet  
COMPANY PUBLIC

## 1. Introduction

---

This document describes the functionality and electrical specifications of the contactless reader/writer MFRC522.

**Remark:** The MFRC522 supports all variants of the MIFARE Mini, MIFARE 1K, MIFARE 4K, MIFARE Ultralight, MIFARE DESFire EV1 and MIFARE Plus RF identification protocols. To aid readability throughout this data sheet, the MIFARE Mini, MIFARE 1K, MIFARE 4K, MIFARE Ultralight, MIFARE DESFire EV1 and MIFARE Plus products and protocols have the generic name MIFARE.

## 2. General description

---

The MFRC522 is a highly integrated reader/writer IC for contactless communication at 13.56 MHz. The MFRC522 reader supports ISO/IEC 14443 A/MIFARE mode.

The MFRC522's internal transmitter is able to drive a reader/writer antenna designed to communicate with ISO/IEC 14443 A/MIFARE cards and transponders without additional active circuitry. The receiver module provides a robust and efficient implementation for demodulating and decoding signals from ISO/IEC 14443 A/MIFARE compatible cards and transponders. The digital module manages the complete ISO/IEC 14443 A framing and error detection (parity and CRC) functionality.

The MFRC522 supports MF1xxS20, MF1xxS70 and MF1xxS50 products. The MFRC522 supports contactless communication and uses MIFARE higher transfer speeds up to 848 kBd in both directions.

The following host interfaces are provided:

- Serial Peripheral Interface (SPI)
- Serial UART (similar to RS232 with voltage levels dependant on pin voltage supply)
- I<sup>2</sup>C-bus interface

### 2.1 Differences between version 1.0 and 2.0

The MFRC522 is available in two versions:

- MFRC52201HN1, hereafter referred to version 1.0 and
- MFRC52202HN1, hereafter referred to version 2.0.

The MFRC522 version 2.0 is fully compatible to version 1.0 and offers in addition the following features and improvements:



- Increased stability of the reader IC in rough conditions
- An additional timer prescaler, see [Section 8.5](#).
- A corrected CRC handling when RX Multiple is set to 1

This data sheet version covers both versions of the MFRC522 and describes the differences between the versions if applicable.

### 3. Features and benefits

- Highly integrated analog circuitry to demodulate and decode responses
- Buffered output drivers for connecting an antenna with the minimum number of external components
- Supports ISO/IEC 14443 A/MIFARE
- Typical operating distance in Read/Write mode up to 50 mm depending on the antenna size and tuning
- Supports MF1xxS20, MF1xxS70 and MF1xxS50 encryption in Read/Write mode
- Supports ISO/IEC 14443 A higher transfer speed communication up to 848 kBd
- Supports MFIN/MFOUT
- Additional internal power supply to the smart card IC connected via MFIN/MFOUT
- Supported host interfaces
  - ◆ SPI up to 10 Mbit/s
  - ◆ I<sup>2</sup>C-bus interface up to 400 kBd in Fast mode, up to 3400 kBd in High-speed mode
  - ◆ RS232 Serial UART up to 1228.8 kBd, with voltage levels dependant on pin voltage supply
- FIFO buffer handles 64 byte send and receive
- Flexible interrupt modes
- Hard reset with low power function
- Power-down by software mode
- Programmable timer
- Internal oscillator for connection to 27.12 MHz quartz crystal
- 2.5 V to 3.3 V power supply
- CRC coprocessor
- Programmable I/O pins
- Internal self-test

### 4. Quick reference data

**Table 1. Quick reference data**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V <sub>DDA</sub>	analog supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ;	<a href="#">[1][2]</a> 2.5	3.3	3.6	V
V <sub>DDD</sub>	digital supply voltage	V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	2.5	3.3	3.6	V
V <sub>DD(TVDD)</sub>	TVDD supply voltage		2.5	3.3	3.6	V
V <sub>DD(PVDD)</sub>	PVDD supply voltage		<a href="#">[3]</a> 1.6	1.8	3.6	V
V <sub>DD(SVDD)</sub>	SVDD supply voltage	V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	1.6	-	3.6	V

**Table 1. Quick reference data ...continued**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit	
I <sub>pd</sub>	power-down current	V <sub>D<sub>DA</sub></sub> = V <sub>D<sub>DD</sub></sub> = V <sub>D<sub>D(TVDD)</sub></sub> = V <sub>D<sub>D(PVDD)</sub></sub> = 3 V					
		hard power-down; pin NRSTPD set LOW	[4]	-	-	5	μA
		soft power-down; RF level detector on	[4]	-	-	10	μA
I <sub>DD</sub>	digital supply current	pin DVDD; V <sub>D<sub>DD</sub></sub> = 3 V	-	6.5	9	mA	
I <sub>D<sub>DA</sub></sub>	analog supply current	pin AVDD; V <sub>D<sub>DA</sub></sub> = 3 V, CommandReg register's RcvOff bit = 0	-	7	10	mA	
		pin AVDD; receiver switched off; V <sub>D<sub>DA</sub></sub> = 3 V, CommandReg register's RcvOff bit = 1	-	3	5	mA	
I <sub>D<sub>D(PVDD)</sub></sub>	PVDD supply current	pin PVDD	[5]	-	40	mA	
I <sub>D<sub>D(TVDD)</sub></sub>	TVDD supply current	pin TVDD; continuous wave	[6][7][8]	-	60	100	mA
T <sub>amb</sub>	ambient temperature	HVQFN32	-25	-	+85	°C	

- [1] Supply voltages below 3 V reduce the performance in, for example, the achievable operating distance.
- [2] V<sub>D<sub>DA</sub></sub>, V<sub>D<sub>DD</sub></sub> and V<sub>D<sub>D(TVDD)</sub></sub> must always be the same voltage.
- [3] V<sub>D<sub>D(PVDD)</sub></sub> must always be the same or lower voltage than V<sub>D<sub>DD</sub></sub>.
- [4] I<sub>pd</sub> is the total current for all supplies.
- [5] I<sub>D<sub>D(PVDD)</sub></sub> depends on the overall load at the digital pins.
- [6] I<sub>D<sub>D(TVDD)</sub></sub> depends on V<sub>D<sub>D(TVDD)</sub></sub> and the external circuit connected to pins TX1 and TX2.
- [7] During typical circuit operation, the overall current is below 100 mA.
- [8] Typical value using a complementary driver configuration and an antenna matched to 40 Ω between pins TX1 and TX2 at 13.56 MHz.

## 5. Ordering information

**Table 2. Ordering information**

Type number	Package		
	Name	Description	Version
MFRC52201HN1/TRAYB <sup>[1]</sup>	HVQFN32	plastic thermal enhanced very thin quad flat package; no leads; 32 terminal; body 5 × 5 × 0.85 mm	SOT617-1
MFRC52201HN1/TRAYBM <sup>[2]</sup>	HVQFN32	plastic thermal enhanced very thin quad flat package; no leads; 32 terminal; body 5 × 5 × 0.85 mm	SOT617-1
MFRC52202HN1/TRAYB <sup>[1]</sup>	HVQFN32	plastic thermal enhanced very thin quad flat package; no leads; 32 terminal; body 5 × 5 × 0.85 mm	SOT617-1
MFRC52202HN1/TRAYBM <sup>[2]</sup>	HVQFN32	plastic thermal enhanced very thin quad flat package; no leads; 32 terminal; body 5 × 5 × 0.85 mm	SOT617-1

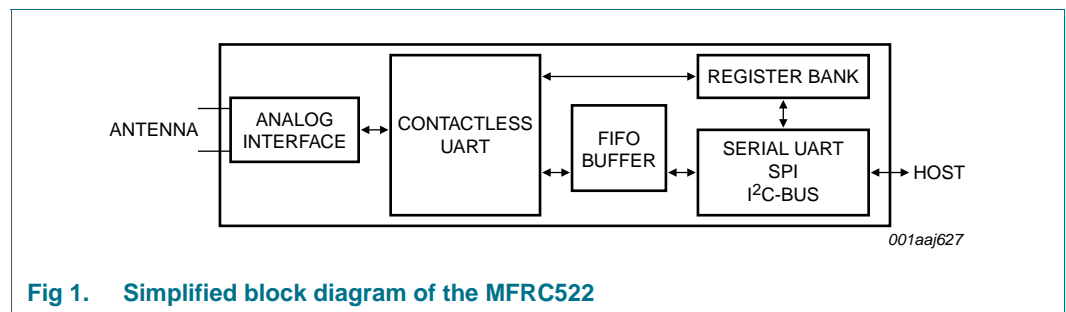
- [1] Delivered in one tray.
- [2] Delivered in five trays.

## 6. Block diagram

The analog interface handles the modulation and demodulation of the analog signals.

The contactless UART manages the protocol requirements for the communication protocols in cooperation with the host. The FIFO buffer ensures fast and convenient data transfer to and from the host and the contactless UART and vice versa.

Various host interfaces are implemented to meet different customer requirements.



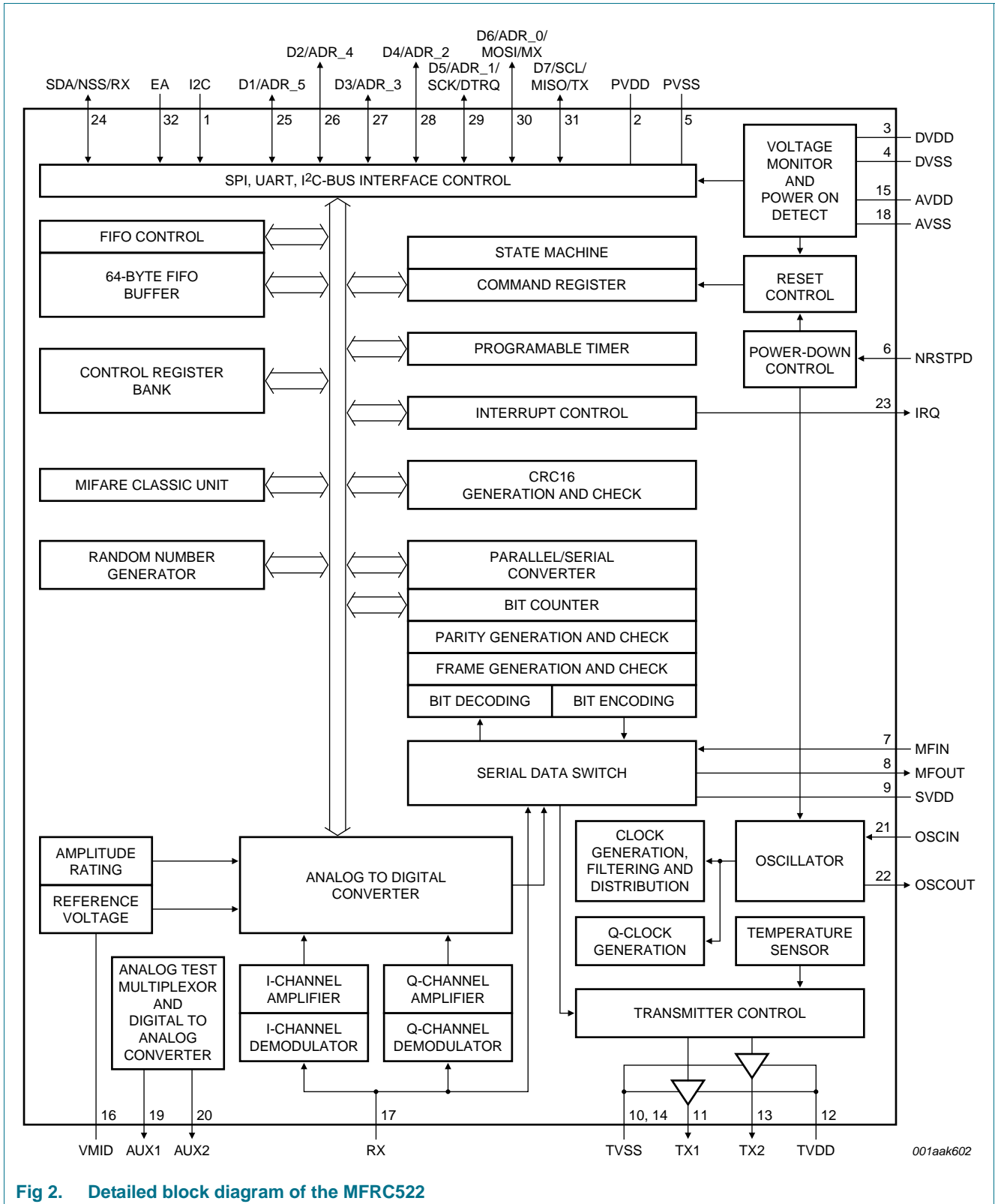
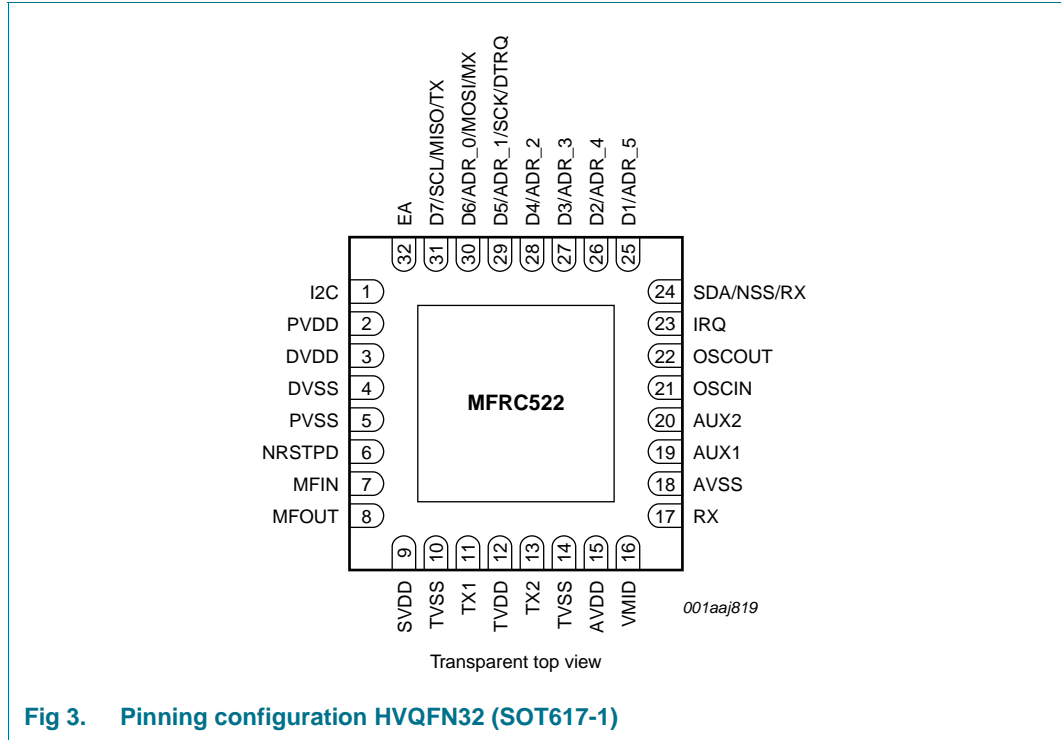


Fig 2. Detailed block diagram of the MFRC522



## 7. Pinning information



**Fig 3. Pinning configuration HVQFN32 (SOT617-1)**

### 7.1 Pin description

**Table 3. Pin description**

Pin	Symbol	Type <sup>[1]</sup>	Description
1	I2C	I	I <sup>2</sup> C-bus enable input <sup>[2]</sup>
2	PVDD	P	pin power supply
3	DVDD	P	digital power supply
4	DVSS	G	digital ground <sup>[3]</sup>
5	PVSS	G	pin power supply ground
6	NRSTPD	I	reset and power-down input: power-down: enabled when LOW; internal current sinks are switched off, the oscillator is inhibited and the input pins are disconnected from the outside world reset: enabled by a positive edge
7	MFIN	I	MIFARE signal input
8	MFOUT	O	MIFARE signal output
9	SVDD	P	MFIN and MFOUT pin power supply
10	TVSS	G	transmitter output stage 1 ground
11	TX1	O	transmitter 1 modulated 13.56 MHz energy carrier output
12	TVDD	P	transmitter power supply: supplies the output stage of transmitters 1 and 2
13	TX2	O	transmitter 2 modulated 13.56 MHz energy carrier output
14	TVSS	G	transmitter output stage 2 ground
15	AVDD	P	analog power supply

Table 3. Pin description ...continued

Pin	Symbol	Type <sup>[1]</sup>	Description
16	VMID	P	internal reference voltage
17	RX	I	RF signal input
18	AVSS	G	analog ground
19	AUX1	O	auxiliary outputs for test purposes
20	AUX2	O	auxiliary outputs for test purposes
21	OSCIN	I	crystal oscillator inverting amplifier input; also the input for an externally generated clock ( $f_{clk} = 27.12$ MHz)
22	OSCOUT	O	crystal oscillator inverting amplifier output
23	IRQ	O	interrupt request output: indicates an interrupt event
24	SDA	I/O	I <sup>2</sup> C-bus serial data line input/output <sup>[2]</sup>
	NSS	I	SPI signal input <sup>[2]</sup>
	RX	I	UART address input <sup>[2]</sup>
25	D1	I/O	test port <sup>[2]</sup>
	ADR_5	I/O	I <sup>2</sup> C-bus address 5 input <sup>[2]</sup>
26	D2	I/O	test port
	ADR_4	I	I <sup>2</sup> C-bus address 4 input <sup>[2]</sup>
27	D3	I/O	test port
	ADR_3	I	I <sup>2</sup> C-bus address 3 input <sup>[2]</sup>
28	D4	I/O	test port
	ADR_2	I	I <sup>2</sup> C-bus address 2 input <sup>[2]</sup>
29	D5	I/O	test port
	ADR_1	I	I <sup>2</sup> C-bus address 1 input <sup>[2]</sup>
	SCK	I	SPI serial clock input <sup>[2]</sup>
	DTRQ	O	UART request to send output to microcontroller <sup>[2]</sup>
30	D6	I/O	test port
	ADR_0	I	I <sup>2</sup> C-bus address 0 input <sup>[2]</sup>
	MOSI	I/O	SPI master out, slave in <sup>[2]</sup>
	MX	O	UART output to microcontroller <sup>[2]</sup>
31	D7	I/O	test port
	SCL	I/O	I <sup>2</sup> C-bus clock input/output <sup>[2]</sup>
	MISO	I/O	SPI master in, slave out <sup>[2]</sup>
	TX	O	UART data output to microcontroller <sup>[2]</sup>
32	EA	I	external address input for coding I <sup>2</sup> C-bus address <sup>[2]</sup>

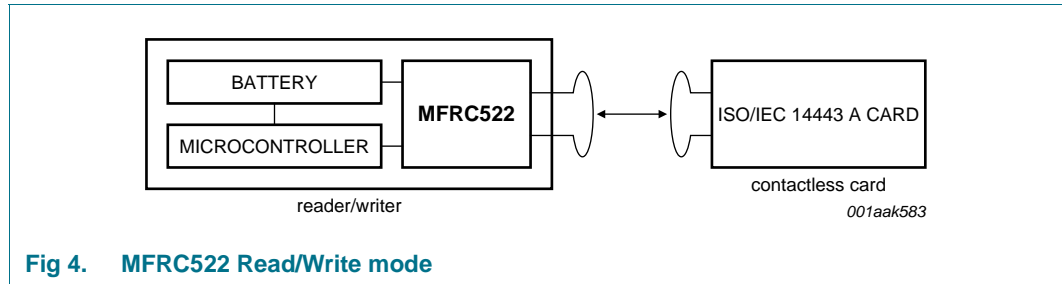
[1] Pin types: I = Input, O = Output, I/O = Input/Output, P = Power and G = Ground.

[2] The pin functionality of these pins is explained in [Section 8.1 "Digital interfaces"](#).

[3] Connection of heatsink pad on package bottom side is not necessary. Optional connection to pin DVSS is possible.

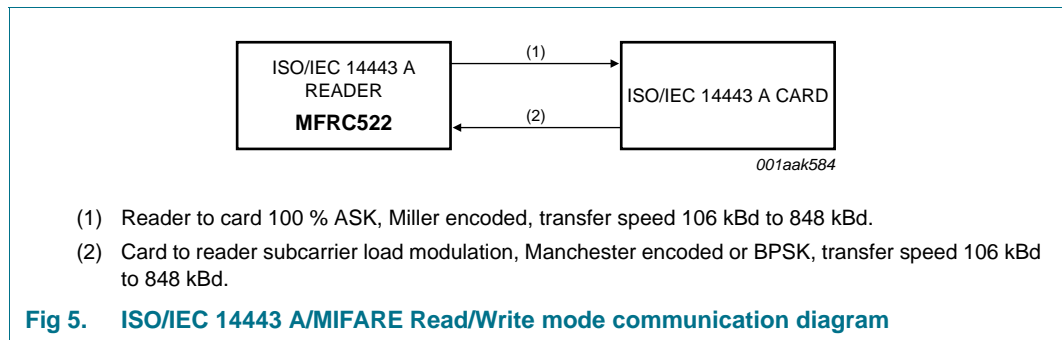
## 8. Functional description

The MFRC522 transmission module supports the Read/Write mode for ISO/IEC 14443 A/MIFARE using various transfer speeds and modulation protocols.



**Fig 4. MFRC522 Read/Write mode**

The physical level communication is shown in [Figure 5](#).



**Fig 5. ISO/IEC 14443 A/MIFARE Read/Write mode communication diagram**

The physical parameters are described in [Table 4](#).

**Table 4. Communication overview for ISO/IEC 14443 A/MIFARE reader/writer**

Communication direction	Signal type	Transfer speed			
		106 kBd	212 kBd	424 kBd	848 kBd
Reader to card (send data from the MFRC522 to a card)	reader side modulation	100 % ASK	100 % ASK	100 % ASK	100 % ASK
	bit encoding	modified Miller encoding	modified Miller encoding	modified Miller encoding	modified Miller encoding
	bit length	128 (13.56 μs)	64 (13.56 μs)	32 (13.56 μs)	16 (13.56 μs)
Card to reader (MFRC522 receives data from a card)	card side modulation	subcarrier load modulation	subcarrier load modulation	subcarrier load modulation	subcarrier load modulation
	subcarrier frequency	13.56 MHz / 16	13.56 MHz / 16	13.56 MHz / 16	13.56 MHz / 16
	bit encoding	Manchester encoding	BPSK	BPSK	BPSK

The MFRC522's contactless UART and dedicated external host must manage the complete ISO/IEC 14443 A/MIFARE protocol. [Figure 6](#) shows the data coding and framing according to ISO/IEC 14443 A/MIFARE.

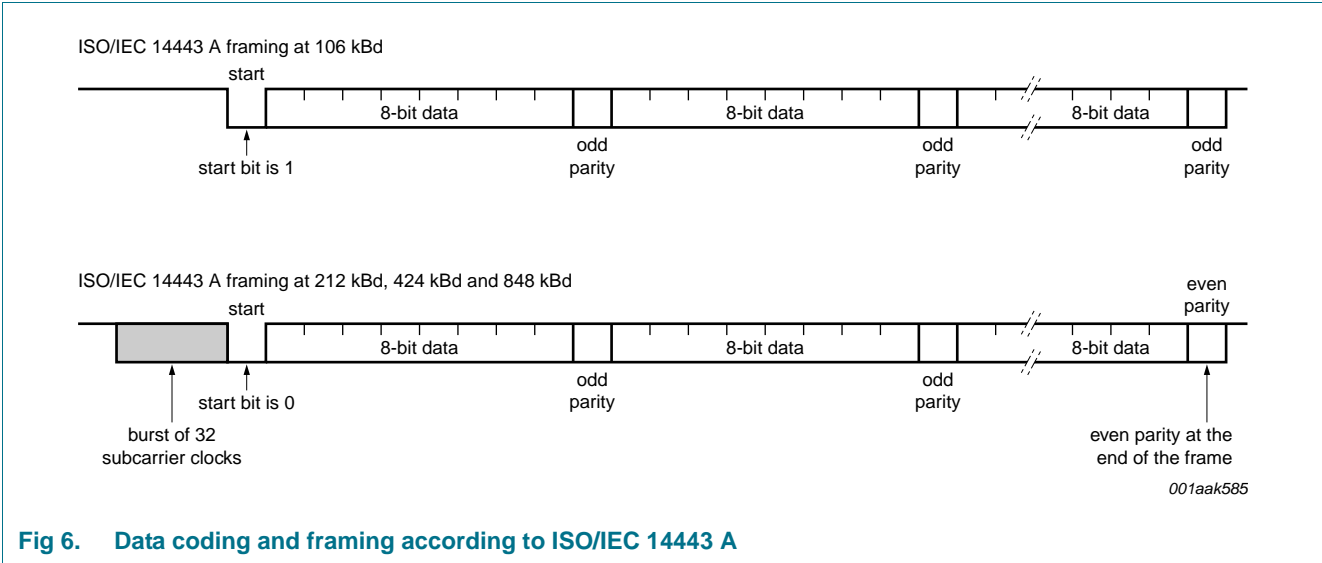


Fig 6. Data coding and framing according to ISO/IEC 14443 A

The internal CRC coprocessor calculates the CRC value based on ISO/IEC 14443 A part 3 and handles parity generation internally according to the transfer speed. Automatic parity generation can be switched off using the MfRxReg register’s ParityDisable bit.

## 8.1 Digital interfaces

### 8.1.1 Automatic microcontroller interface detection

The MFRC522 supports direct interfacing of hosts using SPI, I<sup>2</sup>C-bus or serial UART interfaces. The MFRC522 resets its interface and checks the current host interface type automatically after performing a power-on or hard reset. The MFRC522 identifies the host interface by sensing the logic levels on the control pins after the reset phase. This is done using a combination of fixed pin connections. Table 5 shows the different connection configurations.

Table 5. Connection protocol for detecting different interface types

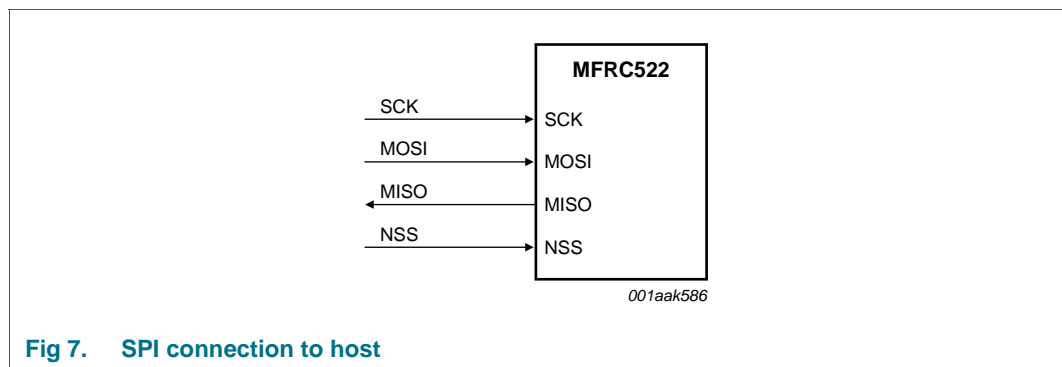
Pin	Interface type		
	UART (input)	SPI (output)	I <sup>2</sup> C-bus (I/O)
SDA	RX	NSS	SDA
I2C	0	0	1
EA	0	1	EA
D7	TX	MISO	SCL
D6	MX	MOSI	ADR_0
D5	DTRQ	SCK	ADR_1
D4	-	-	ADR_2
D3	-	-	ADR_3
D2	-	-	ADR_4
D1	-	-	ADR_5

**8.1.2 Serial Peripheral Interface**

A serial peripheral interface (SPI compatible) is supported to enable high-speed communication to the host. The interface can handle data speeds up to 10 Mbit/s. When communicating with a host, the MFRC522 acts as a slave, receiving data from the external host for register settings, sending and receiving data relevant for RF interface communication.

An interface compatible with SPI enables high-speed serial communication between the MFRC522 and a microcontroller. The implemented interface is in accordance with the SPI standard.

The timing specification is given in [Section 14.1 on page 77](#).



**Fig 7. SPI connection to host**

The MFRC522 acts as a slave during SPI communication. The SPI clock signal SCK must be generated by the master. Data communication from the master to the slave uses the MOSI line. The MISO line is used to send data from the MFRC522 to the master.

Data bytes on both MOSI and MISO lines are sent with the MSB first. Data on both MOSI and MISO lines must be stable on the rising edge of the clock and can be changed on the falling edge. Data is provided by the MFRC522 on the falling clock edge and is stable during the rising clock edge.

**8.1.2.1 SPI read data**

Reading data using SPI requires the byte order shown in [Table 6](#) to be used. It is possible to read out up to n-data bytes.

The first byte sent defines both the mode and the address.

**Table 6. MOSI and MISO byte order**

Line	Byte 0	Byte 1	Byte 2	To	Byte n	Byte n + 1
MOSI	address 0	address 1	address 2	...	address n	00
MISO	X <sup>[1]</sup>	data 0	data 1	...	data n – 1	data n

[1] X = Do not care.

**Remark:** The MSB must be sent first.

8.1.2.2 SPI write data

To write data to the MFRC522 using SPI requires the byte order shown in [Table 7](#). It is possible to write up to n data bytes by only sending one address byte.

The first send byte defines both the mode and the address byte.

Table 7. MOSI and MISO byte order

Line	Byte 0	Byte 1	Byte 2	To	Byte n	Byte n + 1
MOSI	address 0	data 0	data 1	...	data n – 1	data n
MISO	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	...	X <sup>[1]</sup>	X <sup>[1]</sup>

[1] X = Do not care.

**Remark:** The MSB must be sent first.

8.1.2.3 SPI address byte

The address byte must meet the following format.

The MSB of the first byte defines the mode used. To read data from the MFRC522 the MSB is set to logic 1. To write data to the MFRC522 the MSB must be set to logic 0. Bits 6 to 1 define the address and the LSB is set to logic 0.

Table 8. Address byte 0 register; address MOSI

7 (MSB)	6	5	4	3	2	1	0 (LSB)
1 = read 0 = write	address						0

8.1.3 UART interface

8.1.3.1 Connection to a host

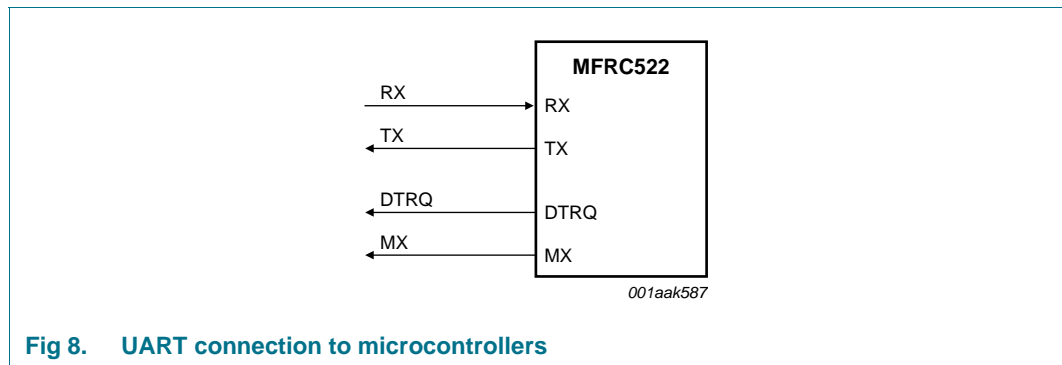


Fig 8. UART connection to microcontrollers

**Remark:** Signals DTRQ and MX can be disabled by clearing TestPinEnReg register's RS232LineEn bit.

### 8.1.3.2 Selectable UART transfer speeds

The internal UART interface is compatible with an RS232 serial interface.

The default transfer speed is 9.6 kBd. To change the transfer speed, the host controller must write a value for the new transfer speed to the SerialSpeedReg register. Bits BR\_T0[2:0] and BR\_T1[4:0] define the factors for setting the transfer speed in the SerialSpeedReg register.

The BR\_T0[2:0] and BR\_T1[4:0] settings are described in [Table 9](#). Examples of different transfer speeds and the relevant register settings are given in [Table 10](#).

**Table 9. BR\_T0 and BR\_T1 settings**

BR_Tn	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
BR_T0 factor	1	1	2	4	8	16	32	64
BR_T1 range	1 to 32	33 to 64	33 to 64	33 to 64	33 to 64	33 to 64	33 to 64	33 to 64

**Table 10. Selectable UART transfer speeds**

Transfer speed (kBd)	SerialSpeedReg value		Transfer speed accuracy (%) <sup>[1]</sup>
	Decimal	Hexadecimal	
7.2	250	FAh	-0.25
9.6	235	EBh	0.32
14.4	218	DAh	-0.25
19.2	203	CBh	0.32
38.4	171	ABh	0.32
57.6	154	9Ah	-0.25
115.2	122	7Ah	-0.25
128	116	74h	-0.06
230.4	90	5Ah	-0.25
460.8	58	3Ah	-0.25
921.6	28	1Ch	1.45
1228.8	21	15h	0.32

[1] The resulting transfer speed error is less than 1.5 % for all described transfer speeds.

The selectable transfer speeds shown in [Table 10](#) are calculated according to the following equations:

If BR\_T0[2:0] = 0:

$$transfer\ speed = \frac{27.12 \times 10^6}{(BR\_T0 + 1)} \tag{1}$$

If BR\_T0[2:0] > 0:

$$transfer\ speed = \left( \frac{27.12 \times 10^6}{(BR\_T1 + 33)} \right)_{2^{(BR\_T0 - 1)}} \tag{2}$$

**Remark:** Transfer speeds above 1228.8 kBd are not supported.

8.1.3.3 UART framing

Table 11. UART framing

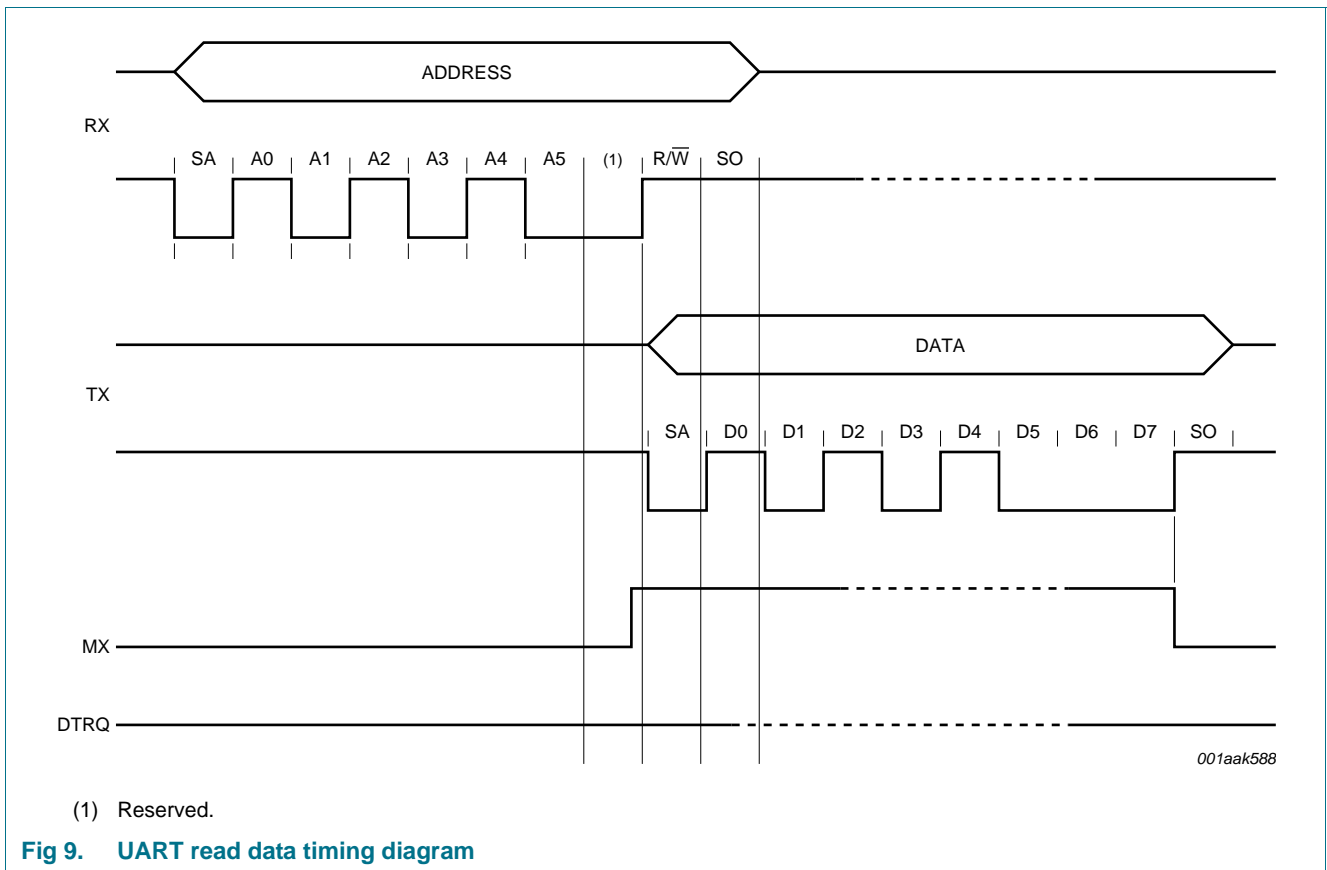
Bit	Length	Value
Start	1-bit	0
Data	8 bits	data
Stop	1-bit	1

**Remark:** The LSB for data and address bytes must be sent first. No parity bit is used during transmission.

**Read data:** To read data using the UART interface, the flow shown in Table 12 must be used. The first byte sent defines both the mode and the address.

Table 12. Read data byte order

Pin	Byte 0	Byte 1
RX (pin 24)	address	-
TX (pin 31)	-	data 0



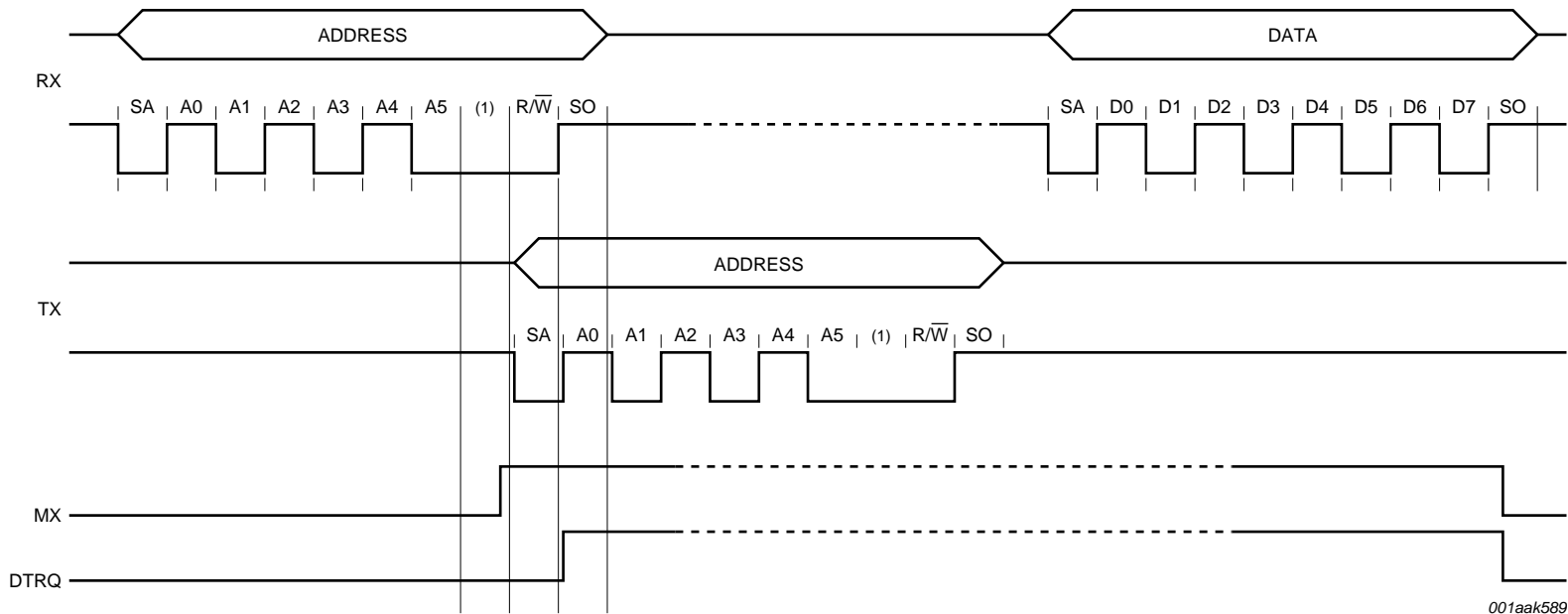
**Write data:** To write data to the MFRC522 using the UART interface, the structure shown in Table 13 must be used.

The first byte sent defines both the mode and the address.



**Table 13. Write data byte order**

<b>Pin</b>	<b>Byte 0</b>	<b>Byte 1</b>
RX (pin 24)	address 0	data 0
TX (pin 31)	-	address 0



001aak589

(1) Reserved.

**Fig 10. UART write data timing diagram**

**Remark:** The data byte can be sent directly after the address byte on pin RX.

**Address byte:** The address byte has to meet the following format:

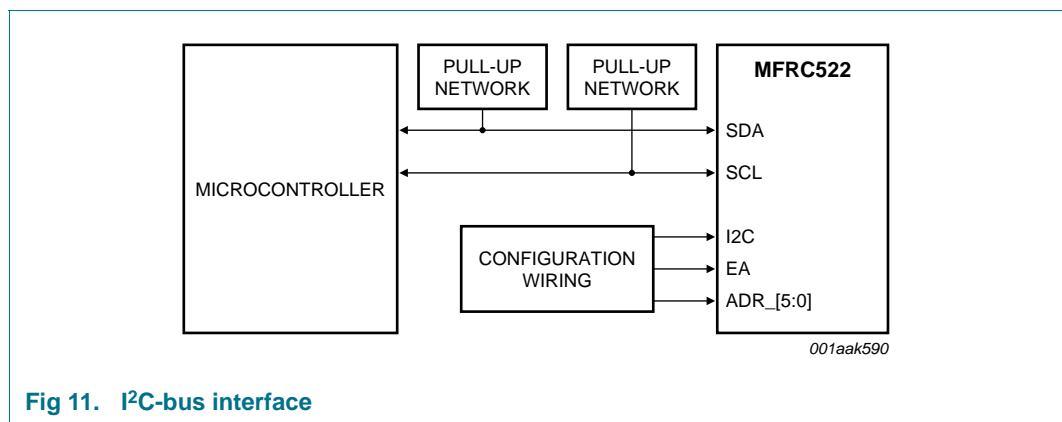
The MSB of the first byte sets the mode used. To read data from the MFRC522, the MSB is set to logic 1. To write data to the MFRC522 the MSB is set to logic 0. Bit 6 is reserved for future use, and bits 5 to 0 define the address; see [Table 14](#).

**Table 14. Address byte 0 register; address MOSI**

7 (MSB)	6	5	4	3	2	1	0 (LSB)
1 = read 0 = write	reserved	address					

**8.1.4 I<sup>2</sup>C-bus interface**

An I<sup>2</sup>C-bus (Inter-IC) interface is supported to enable a low-cost, low pin count serial bus interface to the host. The I<sup>2</sup>C-bus interface is implemented according to NXP Semiconductors' *I<sup>2</sup>C-bus interface specification, rev. 2.1, January 2000*. The interface can only act in Slave mode. Therefore the MFRC522 does not implement clock generation or access arbitration.



**Fig 11. I<sup>2</sup>C-bus interface**

The MFRC522 can act either as a slave receiver or slave transmitter in Standard mode, Fast mode and High-speed mode.

SDA is a bidirectional line connected to a positive supply voltage using a current source or a pull-up resistor. Both SDA and SCL lines are set HIGH when data is not transmitted. The MFRC522 has a 3-state output stage to perform the wired-AND function. Data on the I<sup>2</sup>C-bus can be transferred at data rates of up to 100 kBd in Standard mode, up to 400 kBd in Fast mode or up to 3.4 Mbit/s in High-speed mode.

If the I<sup>2</sup>C-bus interface is selected, spike suppression is activated on lines SCL and SDA as defined in the I<sup>2</sup>C-bus interface specification.

See [Table 155 on page 78](#) for timing requirements.

8.1.4.1 Data validity

Data on the SDA line must be stable during the HIGH clock period. The HIGH or LOW state of the data line must only change when the clock signal on SCL is LOW.

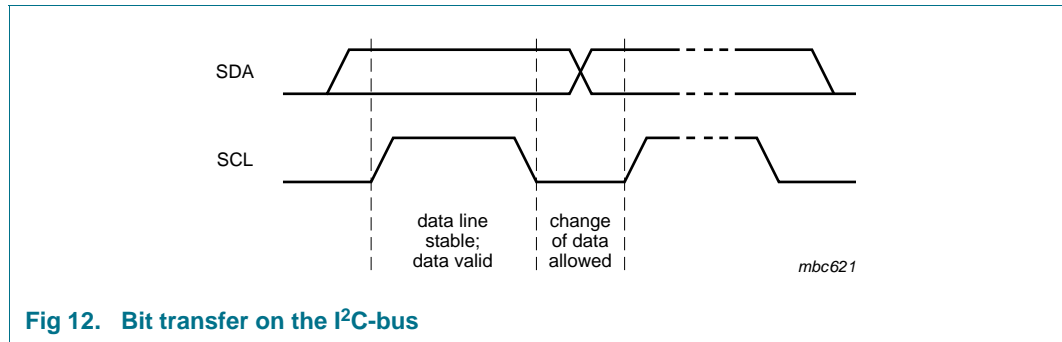


Fig 12. Bit transfer on the I<sup>2</sup>C-bus

8.1.4.2 START and STOP conditions

To manage the data transfer on the I<sup>2</sup>C-bus, unique START (S) and STOP (P) conditions are defined.

- A START condition is defined with a HIGH-to-LOW transition on the SDA line while SCL is HIGH.
- A STOP condition is defined with a LOW-to-HIGH transition on the SDA line while SCL is HIGH.

The I<sup>2</sup>C-bus master always generates the START and STOP conditions. The bus is busy after the START condition. The bus is free again a certain time after the STOP condition.

The bus stays busy if a repeated START (Sr) is generated instead of a STOP condition. The START (S) and repeated START (Sr) conditions are functionally identical. Therefore, S is used as a generic term to represent both the START (S) and repeated START (Sr) conditions.

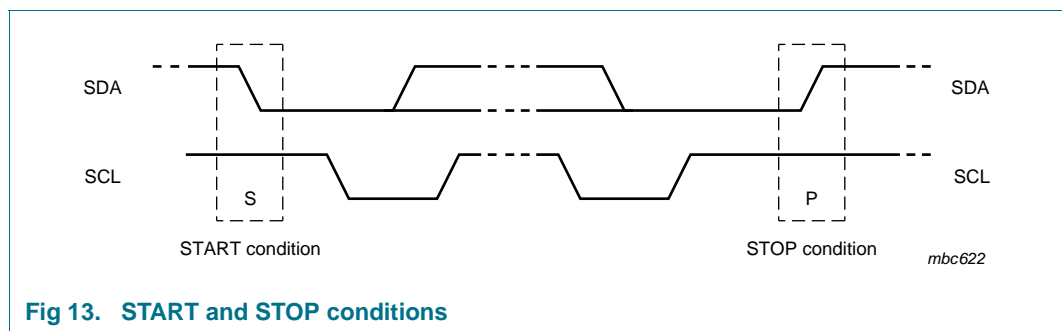


Fig 13. START and STOP conditions

8.1.4.3 Byte format

Each byte must be followed by an acknowledge bit. Data is transferred with the MSB first; see Figure 16. The number of transmitted bytes during one data transfer is unrestricted but must meet the read/write cycle format.

8.1.4.4 Acknowledge

An acknowledge must be sent at the end of one data byte. The acknowledge-related clock pulse is generated by the master. The transmitter of data, either master or slave, releases the SDA line (HIGH) during the acknowledge clock pulse. The receiver pulls down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse.

The master can then generate either a STOP (P) condition to stop the transfer or a repeated START (Sr) condition to start a new transfer.

A master-receiver indicates the end of data to the slave-transmitter by not generating an acknowledge on the last byte that was clocked out by the slave. The slave-transmitter releases the data line to allow the master to generate a STOP (P) or repeated START (Sr) condition.

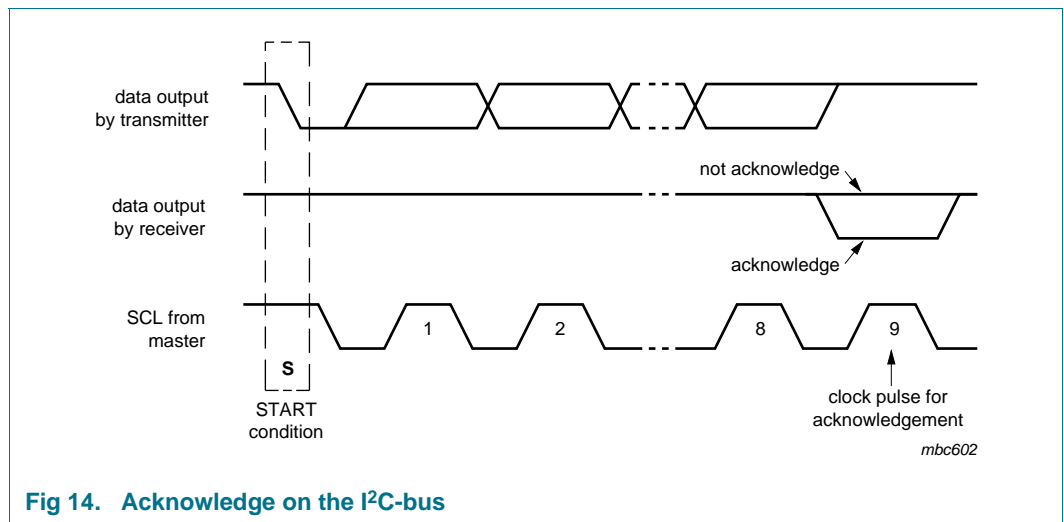


Fig 14. Acknowledge on the I<sup>2</sup>C-bus

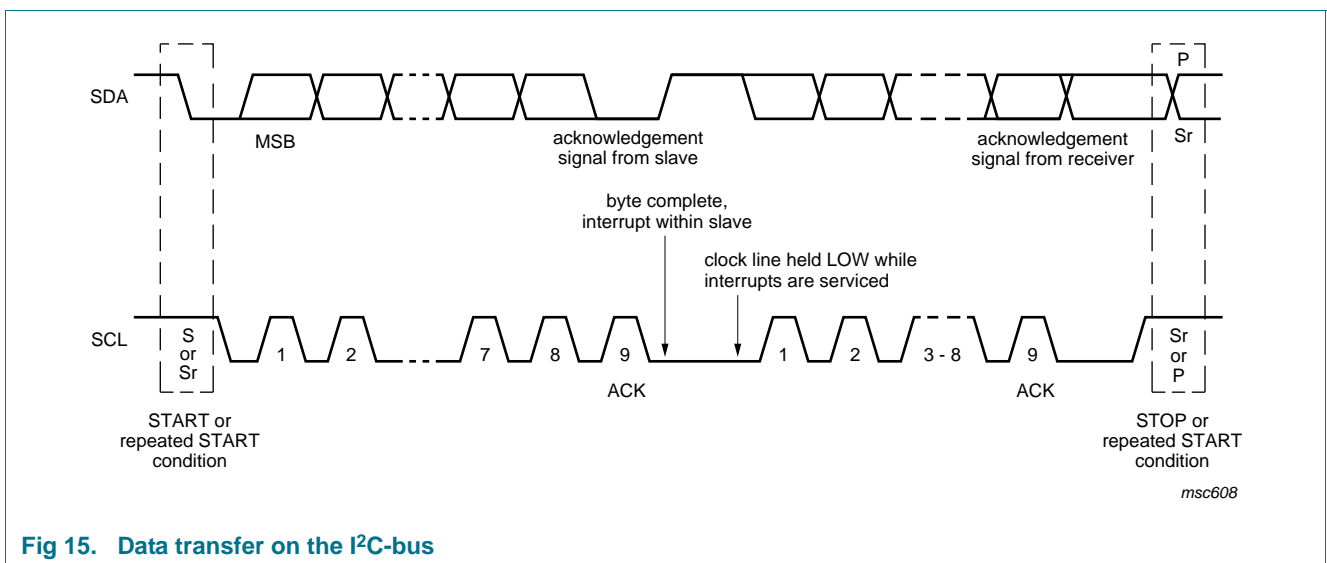


Fig 15. Data transfer on the I<sup>2</sup>C-bus

**8.1.4.5 7-Bit addressing**

During the I<sup>2</sup>C-bus address procedure, the first byte after the START condition is used to determine which slave will be selected by the master.

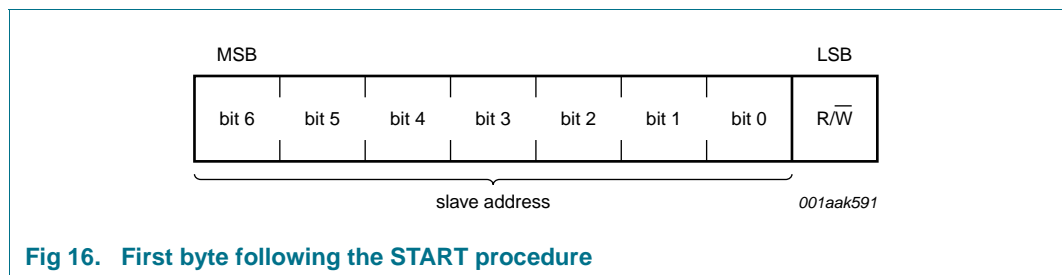
Several address numbers are reserved. During device configuration, the designer must ensure that collisions with these reserved addresses cannot occur. Check the *I<sup>2</sup>C-bus specification* for a complete list of reserved addresses.

The I<sup>2</sup>C-bus address specification is dependent on the definition of pin EA. Immediately after releasing pin NRSTPD or after a power-on reset, the device defines the I<sup>2</sup>C-bus address according to pin EA.

If pin EA is set LOW, the upper 4 bits of the device bus address are reserved by NXP Semiconductors and set to 0101b for all MFRC522 devices. The remaining 3 bits (ADR\_0, ADR\_1, ADR\_2) of the slave address can be freely configured by the customer to prevent collisions with other I<sup>2</sup>C-bus devices.

If pin EA is set HIGH, ADR\_0 to ADR\_5 can be completely specified at the external pins according to [Table 5 on page 9](#). ADR\_6 is always set to logic 0.

In both modes, the external address coding is latched immediately after releasing the reset condition. Further changes at the used pins are not taken into consideration. Depending on the external wiring, the I<sup>2</sup>C-bus address pins can be used for test signal outputs.



**Fig 16. First byte following the START procedure**

**8.1.4.6 Register write access**

To write data from the host controller using the I<sup>2</sup>C-bus to a specific register in the MFRC522 the following frame format must be used.

- The first byte of a frame indicates the device address according to the I<sup>2</sup>C-bus rules.
- The second byte indicates the register address followed by up to n-data bytes.

In one frame all data bytes are written to the same register address. This enables fast FIFO buffer access. The Read/Write (R/ $\overline{W}$ ) bit is set to logic 0.

8.1.4.7 Register read access

To read out data from a specific register address in the MFRC522, the host controller must use the following procedure:

- Firstly, a write access to the specific register address must be performed as indicated in the frame that follows
- The first byte of a frame indicates the device address according to the I<sup>2</sup>C-bus rules
- The second byte indicates the register address. No data bytes are added
- The Read/Write bit is 0

After the write access, read access can start. The host sends the device address of the MFRC522. In response, the MFRC522 sends the content of the read access register. In one frame all data bytes can be read from the same register address. This enables fast FIFO buffer access or register polling.

The Read/Write (R/W) bit is set to logic 1.

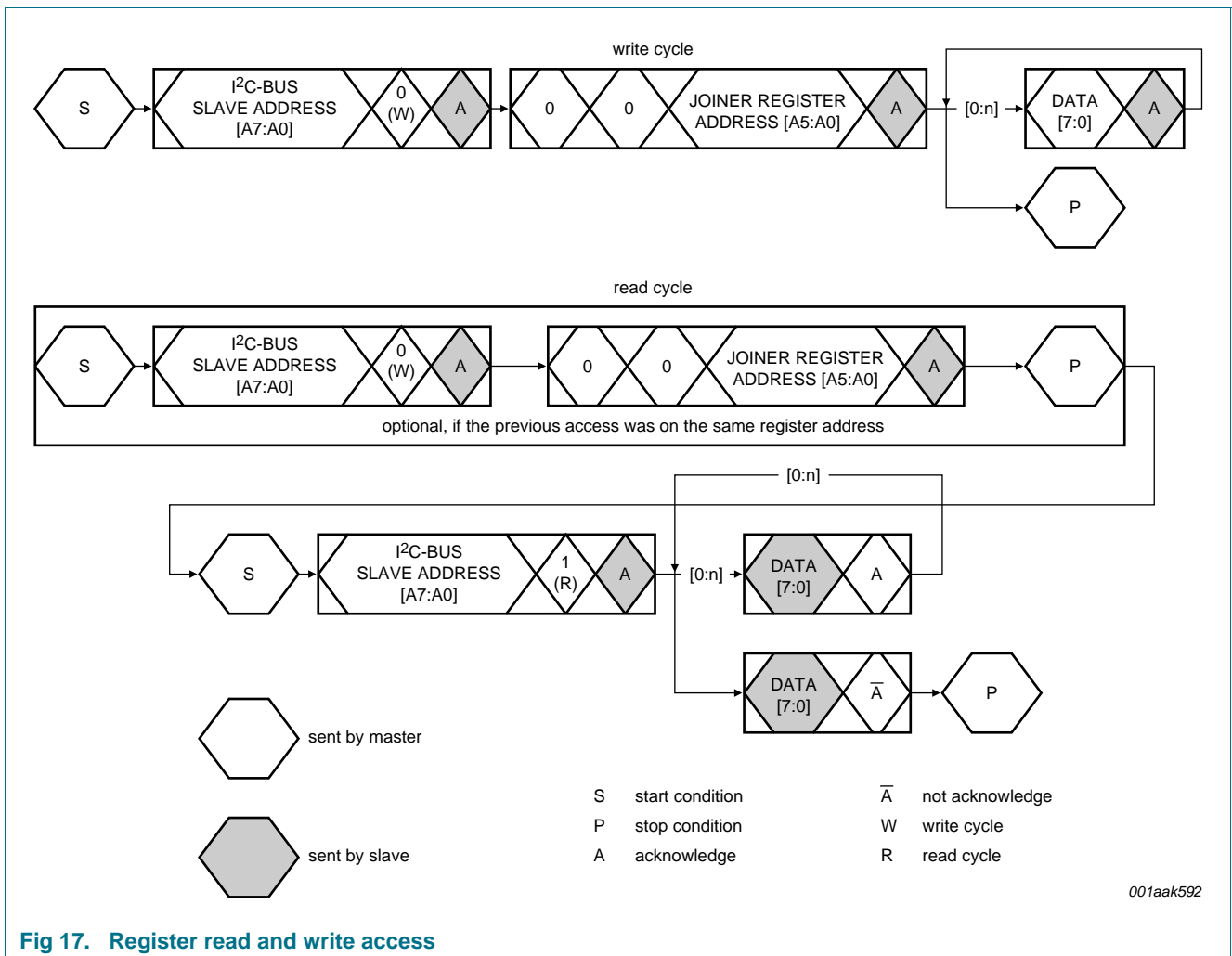


Fig 17. Register read and write access

**8.1.4.8 High-speed mode**

In High-speed mode (HS mode), the device can transfer information at data rates of up to 3.4 Mbit/s, while remaining fully downward-compatible with Fast or Standard mode (F/S mode) for bidirectional communication in a mixed-speed bus system.

**8.1.4.9 High-speed transfer**

To achieve data rates of up to 3.4 Mbit/s the following improvements have been made to I<sup>2</sup>C-bus operation.

- The inputs of the device in HS mode incorporate spike suppression, a Schmitt trigger on the SDA and SCL inputs and different timing constants when compared to F/S mode
- The output buffers of the device in HS mode incorporate slope control of the falling edges of the SDA and SCL signals with different fall times compared to F/S mode

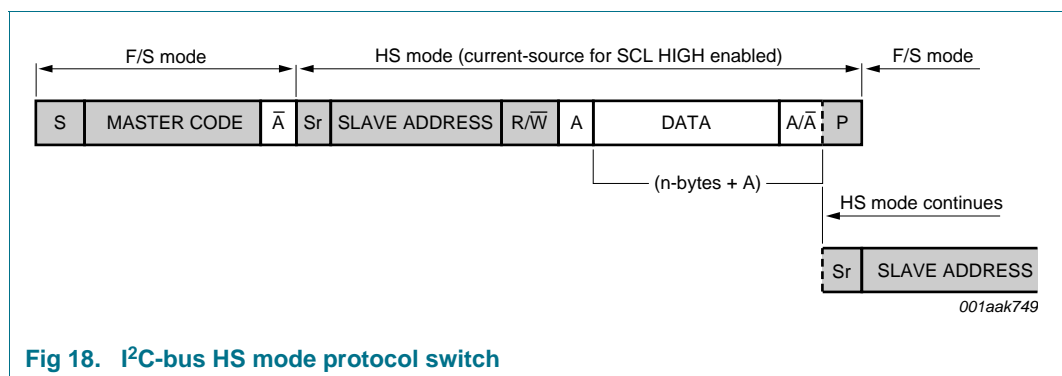
**8.1.4.10 Serial data transfer format in HS mode**

The HS mode serial data transfer format meets the Standard mode I<sup>2</sup>C-bus specification. HS mode can only start after all of the following conditions (all of which are in F/S mode):

1. START condition (S)
2. 8-bit master code (00001XXXb)
3. Not-acknowledge bit ( $\bar{A}$ )

When HS mode starts, the active master sends a repeated START condition (Sr) followed by a 7-bit slave address with a R/W bit address and receives an acknowledge bit (A) from the selected MFRC522.

Data transfer continues in HS mode after the next repeated START (Sr), only switching back to F/S mode after a STOP condition (P). To reduce the overhead of the master code, a master links a number of HS mode transfers, separated by repeated START conditions (Sr).



**Fig 18. I<sup>2</sup>C-bus HS mode protocol switch**



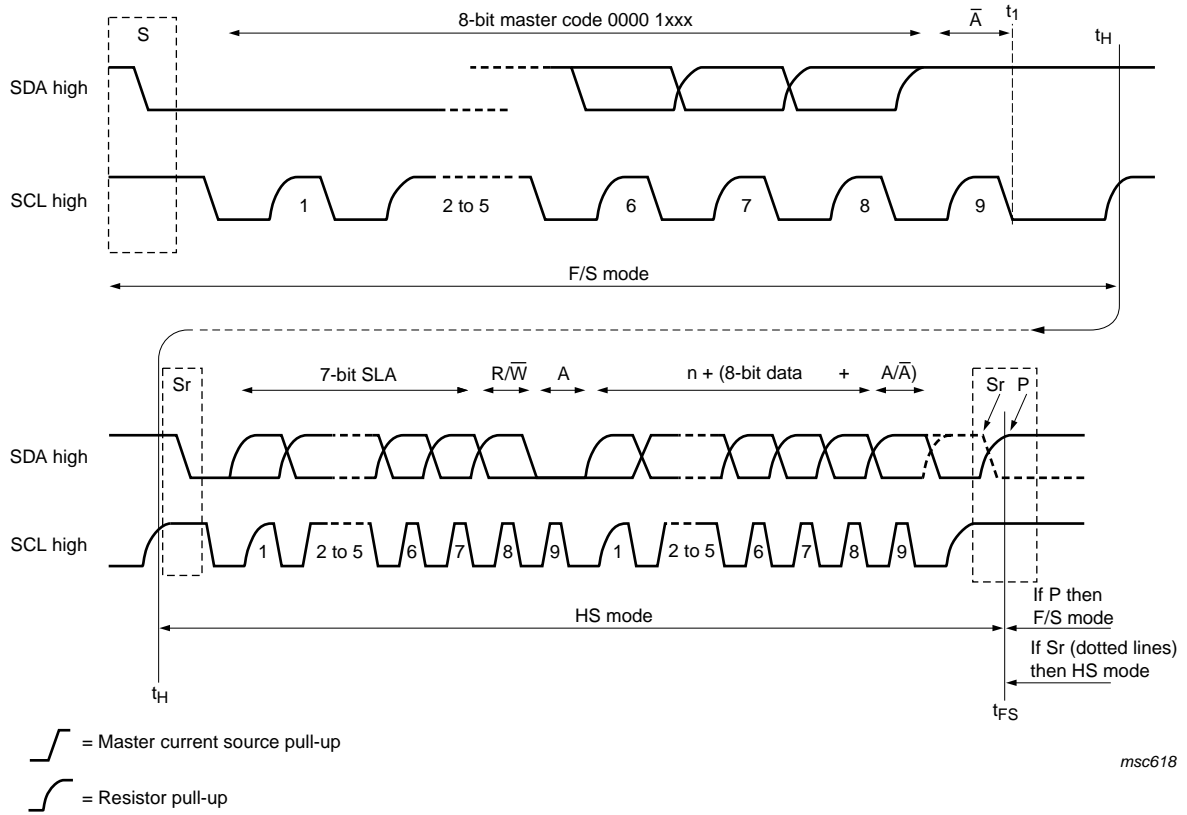


Fig 19. I<sup>2</sup>C-bus HS mode protocol frame

#### 8.1.4.11 Switching between F/S mode and HS mode

After reset and initialization, the MFRC522 is in Fast mode (which is in effect F/S mode as Fast mode is downward-compatible with Standard mode). The connected MFRC522 recognizes the "S 00001XXX A" sequence and switches its internal circuitry from the Fast mode setting to the HS mode setting.

The following actions are taken:

1. Adapt the SDA and SCL input filters according to the spike suppression requirement in HS mode.
2. Adapt the slope control of the SDA output stages.

It is possible for system configurations that do not have other I<sup>2</sup>C-bus devices involved in the communication to switch to HS mode permanently. This is implemented by setting Status2Reg register's I<sup>2</sup>CForceHS bit to logic 1. In permanent HS mode, the master code is not required to be sent. This is not defined in the specification and must only be used when no other devices are connected on the bus. In addition, spikes on the I<sup>2</sup>C-bus lines must be avoided because of the reduced spike suppression.

#### 8.1.4.12 MFRC522 at lower speed modes

MFRC522 is fully downward-compatible and can be connected to an F/S mode I<sup>2</sup>C-bus system. The device stays in F/S mode and communicates at F/S mode speeds because a master code is not transmitted in this configuration.

## 8.2 Analog interface and contactless UART

### 8.2.1 General

The integrated contactless UART supports the external host online with framing and error checking of the protocol requirements up to 848 kBd. An external circuit can be connected to the communication interface pins MFIN and MFOUT to modulate and demodulate the data.

The contactless UART handles the protocol requirements for the communication protocols in cooperation with the host. Protocol handling generates bit and byte-oriented framing. In addition, it handles error detection such as parity and CRC, based on the various supported contactless communication protocols.

**Remark:** The size and tuning of the antenna and the power supply voltage have an important impact on the achievable operating distance.

### 8.2.2 TX p-driver

The signal on pins TX1 and TX2 is the 13.56 MHz energy carrier modulated by an envelope signal. It can be used to drive an antenna directly using a few passive components for matching and filtering; see [Section 15 on page 80](#). The signal on pins TX1 and TX2 can be configured using the TxControlReg register; see [Section 9.3.2.5 on page 49](#).

The modulation index can be set by adjusting the impedance of the drivers. The impedance of the p-driver can be configured using registers CWGsPReg and ModGsPReg. The impedance of the n-driver can be configured using the GsNReg register. The modulation index also depends on the antenna design and tuning.

The TxModeReg and TxSelReg registers control the data rate and framing during transmission and the antenna driver setting to support the different requirements at the different modes and transfer speeds.

**Table 15. Register and bit settings controlling the signal on pin TX1**

Bit Tx1RFEn	Bit Force 100ASK	Bit InvTx1RFOn	Bit InvTx1RFOff	Envelope	Pin TX1	GSPMos	GSNMos	Remarks
0	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	not specified if RF is switched off
1	0	0	X <sup>[1]</sup>	0	RF	pMod	nMod	100 % ASK: pin TX1 pulled to logic 0, independent of the InvTx1RFOff bit
				1	RF	pCW	nCW	
	0	1	X <sup>[1]</sup>	0	RF	pMod	nMod	
				1	RF	pCW	nCW	
1	1	X <sup>[1]</sup>	X <sup>[1]</sup>	0	0	pMod	nMod	
				1	RF_n	pCW	nCW	

[1] X = Do not care.

**Table 16. Register and bit settings controlling the signal on pin TX2**

Bit Tx1RFEn	Bit Force 100ASK	Bit Tx2CW	Bit InvTx2RFOOn	Bit InvTx2RFOff	Envelope	Pin TX2	GSPMos	GSNMos	Remarks	
0	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	not specified if RF is switched off	
1	0	0	0	X <sup>[1]</sup>	0	RF	pMod	nMod	-	
				1	RF	pCW	nCW			
				X <sup>[1]</sup>	0	RF_n	pMod	nMod		
				1	RF_n	pCW	nCW			
	1	0	0	X <sup>[1]</sup>	X <sup>[1]</sup>	RF	pCW	nCW	conductance always CW for the Tx2CW bit	
				X <sup>[1]</sup>	X <sup>[1]</sup>	RF_n	pCW	nCW		
				1	0	0	pMod	nMod		100 % ASK: pin TX2 pulled to logic 0 (independent of the InvTx2RFOOn/InvTx2RFOff bits)
					1	0	0	pMod		
1	1	0	0	X <sup>[1]</sup>	0	0	pMod	nMod	100 % ASK: pin TX2 pulled to logic 0 (independent of the InvTx2RFOOn/InvTx2RFOff bits)	
				X <sup>[1]</sup>	1	RF	pCW	nCW		
				X <sup>[1]</sup>	0	0	pMod	nMod		
				X <sup>[1]</sup>	1	RF_n	pCW	nCW		
1	1	1	0	X <sup>[1]</sup>	X <sup>[1]</sup>	RF	pCW	nCW	InvTx2RFOOn/InvTx2RFOff bits)	
				X <sup>[1]</sup>	X <sup>[1]</sup>	RF_n	pCW	nCW		

[1] X = Do not care.

The following abbreviations have been used in [Table 15](#) and [Table 16](#):

- RF: 13.56 MHz clock derived from 27.12 MHz quartz crystal oscillator divided by 2
- RF\_n: inverted 13.56 MHz clock
- GSPMos: conductance, configuration of the PMOS array
- GSNMos: conductance, configuration of the NMOS array
- pCW: PMOS conductance value for continuous wave defined by the CWGsPReg register
- pMod: PMOS conductance value for modulation defined by the ModGsPReg register
- nCW: NMOS conductance value for continuous wave defined by the GsNReg register's CWGsN[3:0] bits
- nMod: NMOS conductance value for modulation defined by the GsNReg register's ModGsN[3:0] bits
- X = do not care.

**Remark:** If only one driver is switched on, the values for CWGsPReg, ModGsPReg and GsNReg registers are used for both drivers.

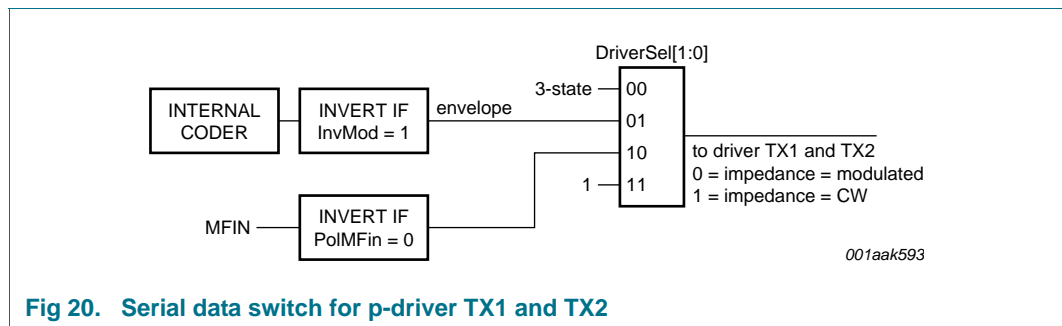
**8.2.3 Serial data switch**

Two main blocks are implemented in the MFRC522. The digital block comprises the state machines, encoder/decoder logic. The analog block comprises the modulator and antenna drivers, the receiver and amplifiers. It is possible for the interface between these two blocks to be configured so that the interfacing signals are routed to pins MFIN and MFOUT.

This topology allows the analog block of the MFRC522 to be connected to the digital block of another device.

The serial signal switch is controlled by the TxSelReg and RxSelReg registers.

Figure 20 shows the serial data switch for p-driver TX1 and TX2.



**Fig 20. Serial data switch for p-driver TX1 and TX2**

**8.2.4 MFIN and MFOUT interface support**

The MFRC522 is divided into a digital circuit block and an analog circuit block. The digital block contains state machines, encoder and decoder logic and so on. The analog block contains the modulator and antenna drivers, receiver and amplifiers. The interface between these two blocks can be configured so that the interfacing signals can be routed to pins MFIN and MFOUT; see Figure 21 on page 27. This configuration is implemented using TxSelReg register’s MFOutSel[3:0] and DriverSel[1:0] bits and RxSelReg register’s UARTSel[1:0] bits.

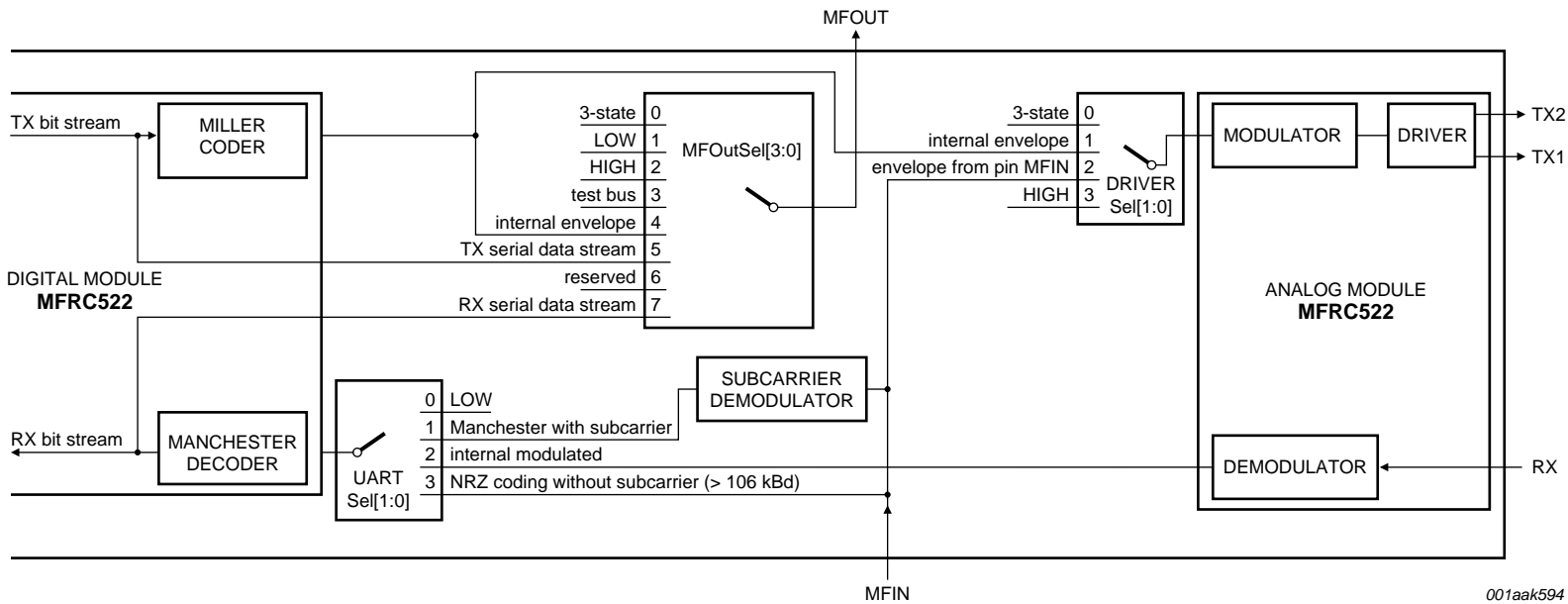
This topology allows some parts of the analog block to be connected to the digital block of another device.

Switch MFOutSel in the TxSelReg register can be used to measure MIFARE and ISO/IEC14443 A related signals. This is especially important during the design-in phase or for test purposes as it enables checking of the transmitted and received data.

The most important use of pins MFIN and MFOUT is found in the active antenna concept. An external active antenna circuit can be connected to the MFRC522’s digital block. Switch MFOutSel must be configured so that the internal Miller encoded signal is sent to pin MFOUT (MFOutSel = 100b). UARTSel[1:0] must be configured to receive a Manchester signal with subcarrier from pin MFIN (UARTSel[1:0] = 01).

It is possible to connect a passive antenna to pins TX1, TX2 and RX (using the appropriate filter and matching circuit) and an active antenna to pins MFOUT and MFIN at the same time. In this configuration, two RF circuits can be driven (one after another) by a single host processor.

**Remark:** Pins MFIN and MFOUT have a dedicated supply on pin SVDD with the ground on pin PVSS. If pin MFIN is not used it must be connected to either pin SVDD or pin PVSS. If pin SVDD is not used it must be connected to either pin DVDD, pin PVDD or any other voltage supply pin.



001aak594

Fig 21. Overview of MFIN and MFOUT signal routing

### 8.2.5 CRC coprocessor

The following CRC coprocessor parameters can be configured:

- The CRC preset value can be either 0000h, 6363h, A671h or FFFFh depending on the ModeReg register's CRCPreset[1:0] bits setting
- The CRC polynomial for the 16-bit CRC is fixed to  $x^{16} + x^{12} + x^5 + 1$
- The CRCResultReg register indicates the result of the CRC calculation. This register is split into two 8-bit registers representing the higher and lower bytes.
- The ModeReg register's MSBFirst bit indicates that data will be loaded with the MSB first.

**Table 17. CRC coprocessor parameters**

Parameter	Value
CRC register length	16-bit CRC
CRC algorithm	algorithm according to ISO/IEC 14443 A and ITU-T
CRC preset value	0000h, 6363h, A671h or FFFFh depending on the setting of the ModeReg register's CRCPreset[1:0] bits

## 8.3 FIFO buffer

An 8 × 64 bit FIFO buffer is used in the MFRC522. It buffers the input and output data stream between the host and the MFRC522's internal state machine. This makes it possible to manage data streams up to 64 bytes long without the need to take timing constraints into account.

### 8.3.1 Accessing the FIFO buffer

The FIFO buffer input and output data bus is connected to the FIFODataReg register. Writing to this register stores one byte in the FIFO buffer and increments the internal FIFO buffer write pointer. Reading from this register shows the FIFO buffer contents stored in the FIFO buffer read pointer and decrements the FIFO buffer read pointer. The distance between the write and read pointer can be obtained by reading the FIFOLevelReg register.

When the microcontroller starts a command, the MFRC522 can, while the command is in progress, access the FIFO buffer according to that command. Only one FIFO buffer has been implemented which can be used for input and output. The microcontroller must ensure that there are not any unintentional FIFO buffer accesses.

### 8.3.2 Controlling the FIFO buffer

The FIFO buffer pointers can be reset by setting FIFOLevelReg register's FlushBuffer bit to logic 1. Consequently, the FIFOLevel[6:0] bits are all set to logic 0 and the ErrorReg register's BufferOvfl bit is cleared. The bytes stored in the FIFO buffer are no longer accessible allowing the FIFO buffer to be filled with another 64 bytes.

### 8.3.3 FIFO buffer status information

The host can get the following FIFO buffer status information:

- Number of bytes stored in the FIFO buffer: FIFOLevelReg register's FIFOLevel[6:0]
- FIFO buffer almost full warning: Status1Reg register's HiAlert bit

- FIFO buffer almost empty warning: Status1Reg register's LoAlert bit
- FIFO buffer overflow warning: ErrorReg register's BufferOvfl bit. The BufferOvfl bit can only be cleared by setting the FIFOLevelReg register's FlushBuffer bit.

The MFRC522 can generate an interrupt signal when:

- ComIEnReg register's LoAlertIEn bit is set to logic 1. It activates pin IRQ when Status1Reg register's LoAlert bit changes to logic 1.
- ComIEnReg register's HiAlertIEn bit is set to logic 1. It activates pin IRQ when Status1Reg register's HiAlert bit changes to logic 1.

If the maximum number of WaterLevel bytes (as set in the WaterLevelReg register) or less are stored in the FIFO buffer, the HiAlert bit is set to logic 1. It is generated according to [Equation 3](#):

$$HiAlert = (64 - FIFOLength) \leq WaterLevel \quad (3)$$

If the number of WaterLevel bytes (as set in the WaterLevelReg register) or less are stored in the FIFO buffer, the LoAlert bit is set to logic 1. It is generated according to [Equation 4](#):

$$LoAlert = FIFOLength \leq WaterLevel \quad (4)$$

## 8.4 Interrupt request system

The MFRC522 indicates certain events by setting the Status1Reg register's IRq bit and, if activated, by pin IRQ. The signal on pin IRQ can be used to interrupt the host using its interrupt handling capabilities. This allows the implementation of efficient host software.

### 8.4.1 Interrupt sources overview

[Table 18](#) shows the available interrupt bits, the corresponding source and the condition for its activation. The ComIrqReg register's TimerIRq interrupt bit indicates an interrupt set by the timer unit which is set when the timer decrements from 1 to 0.

The ComIrqReg register's TxIRq bit indicates that the transmitter has finished. If the state changes from sending data to transmitting the end of the frame pattern, the transmitter unit automatically sets the interrupt bit. The CRC coprocessor sets the DivIrqReg register's CRCIRq bit after processing all the FIFO buffer data which is indicated by CRCReady bit = 1.

The ComIrqReg register's RxIRq bit indicates an interrupt when the end of the received data is detected. The ComIrqReg register's IdleIRq bit is set if a command finishes and the Command[3:0] value in the CommandReg register changes to idle (see [Table 149 on page 69](#)).

The ComIrqReg register's HiAlertIRq bit is set to logic 1 when the Status1Reg register's HiAlert bit is set to logic 1 which means that the FIFO buffer has reached the level indicated by the WaterLevel[5:0] bits.

The ComIrqReg register's LoAlertIRq bit is set to logic 1 when the Status1Reg register's LoAlert bit is set to logic 1 which means that the FIFO buffer has reached the level indicated by the WaterLevel[5:0] bits.



The ComIrqReg register's ErrIRq bit indicates an error detected by the contactless UART during send or receive. This is indicated when any bit is set to logic 1 in register ErrorReg.

**Table 18. Interrupt sources**

Interrupt flag	Interrupt source	Trigger action
IRq	timer unit	the timer counts from 1 to 0
TxIRq	transmitter	a transmitted data stream ends
CRCIRq	CRC coprocessor	all data from the FIFO buffer has been processed
RxIRq	receiver	a received data stream ends
IdleIRq	ComIrqReg register	command execution finishes
HiAlertIRq	FIFO buffer	the FIFO buffer is almost full
LoAlertIRq	FIFO buffer	the FIFO buffer is almost empty
ErrIRq	contactless UART	an error is detected

## 8.5 Timer unit

The MFRC522A has a timer unit which the external host can use to manage timing tasks. The timer unit can be used in one of the following timer/counter configurations:

- Timeout counter
- Watchdog counter
- Stop watch
- Programmable one shot
- Periodical trigger

The timer unit can be used to measure the time interval between two events or to indicate that a specific event occurred after a specific time. The timer can be triggered by events explained in the paragraphs below. The timer does not influence any internal events, for example, a time-out during data reception does not automatically influence the reception process. Furthermore, several timer-related bits can be used to generate an interrupt.

The timer has an input clock of 13.56 MHz derived from the 27.12 MHz quartz crystal oscillator. The timer consists of two stages: prescaler and counter.

The prescaler (TPrescaler) is a 12-bit counter. The reload values (TReloadVal\_Hi[7:0] and TReloadVal\_Lo[7:0]) for TPrescaler can be set between 0 and 4095 in the TModeReg register's TPrescaler\_Hi[3:0] bits and TPrescalerReg register's TPrescaler\_Lo[7:0] bits.

The reload value for the counter is defined by 16 bits between 0 and 65535 in the TReloadReg register.

The current value of the timer is indicated in the TCounterValReg register.

When the counter reaches 0, an interrupt is automatically generated, indicated by the ComIrqReg register's TimerIRq bit setting. If enabled, this event can be indicated on pin IRQ. The TimerIRq bit can be set and reset by the host. Depending on the configuration, the timer will stop at 0 or restart with the value set in the TReloadReg register.

The timer status is indicated by the Status1Reg register's TRunning bit.

The timer can be started manually using the ControlReg register's TStartNow bit and stopped using the ControlReg register's TStopNow bit.

The timer can also be activated automatically to meet any dedicated protocol requirements by setting the TModeReg register's TAuto bit to logic 1.

The delay time of a timer stage is set by the reload value + 1. The total delay time ( $t_{d1}$ ) is calculated using [Equation 5](#):

$$t_{d1} = \frac{(TPrescaler \times 2 + 1) \times (TReloadVal + 1)}{13.56 \text{ MHz}} \quad (5)$$

An example of calculating total delay time ( $t_d$ ) is shown in [Equation 6](#), where the TPrescaler value = 4095 and TReloadVal = 65535:

$$39.59 \text{ s} = \frac{(4095 \times 2 + 1) \times (65535 + 1)}{13.56 \text{ MHz}} \quad (6)$$

**Example:** To give a delay time of 25  $\mu\text{s}$  requires 339 clock cycles to be counted and a TPrescaler value of 169. This configures the timer to count up to 65535 time-slots for every 25  $\mu\text{s}$  period.

The MFRC522 version 2.0 offers in addition a second prescaler timer. Due to the fact that the prescaler counts down to 0 the prescaler period always count an odd number of clocks (1, 3, 5, ..). This may lead to inaccuracy. The second available prescaler timer implements the possibility to change the prescaler reload value to odd numbers, which results in an even prescaler period. This new prescaler can be enabled only in version 2.0 using the register bit DemodeReg, see [Table 72](#). Within this option, the total delay time ( $t_{d2}$ ) is calculated using [Equation 5](#):

$$t_{d2} = \frac{(TPrescaler \times 2 + 2) \times (TReloadVal + 1)}{13.56 \text{ MHz}} \quad (7)$$

**8.6 Power reduction modes**

**8.6.1 Hard power-down**

Hard power-down is enabled when pin NRSTPD is LOW. This turns off all internal current sinks including the oscillator. All digital input buffers are separated from the input pins and clamped internally (except pin NRSTPD). The output pins are frozen at either a HIGH or LOW level.

**8.6.2 Soft power-down mode**

Soft Power-down mode is entered immediately after the CommandReg register's PowerDown bit is set to logic 1. All internal current sinks are switched off, including the oscillator buffer. However, the digital input buffers are not separated from the input pins and keep their functionality. The digital output pins do not change their state.

During soft power-down, all register values, the FIFO buffer content and the configuration keep their current contents.

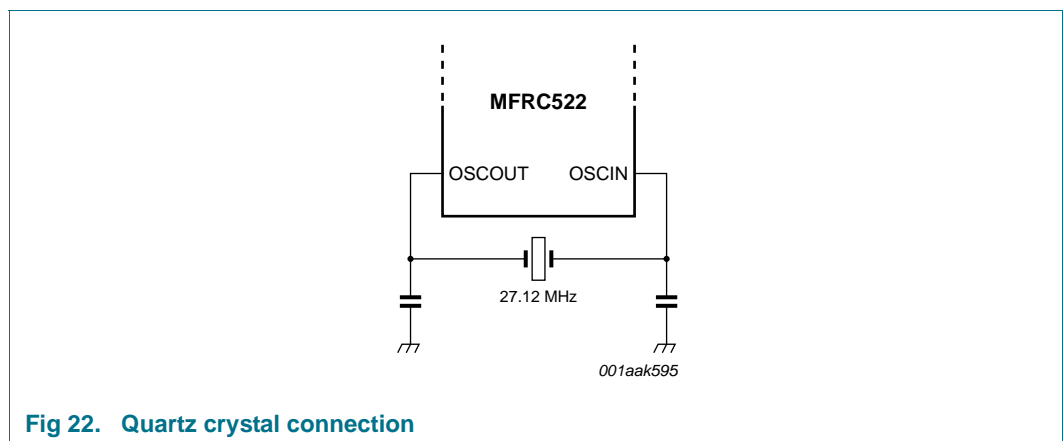
After setting the PowerDown bit to logic 0, it takes 1024 clocks until the Soft power-down mode is exited indicated by the PowerDown bit. Setting it to logic 0 does not immediately clear it. It is cleared automatically by the MFRC522 when Soft power-down mode is exited.

**Remark:** If the internal oscillator is used, you must take into account that it is supplied by pin AVDD and it will take a certain time ( $t_{osc}$ ) until the oscillator is stable and the clock cycles can be detected by the internal logic. It is recommended for the serial UART, to first send the value 55h to the MFRC522. The oscillator must be stable for further access to the registers. To ensure this, perform a read access to address 0 until the MFRC522 answers to the last read ready command with the register content of address 0. This indicates that the MFRC522 is ready.

**8.6.3 Transmitter power-down mode**

The Transmitter Power-down mode switches off the internal antenna drivers thereby, turning off the RF field. Transmitter power-down mode is entered by setting either the TxControlReg register's Tx1RFEn bit or Tx2RFEn bit to logic 0.

**8.7 Oscillator circuit**



**Fig 22. Quartz crystal connection**

The clock applied to the MFRC522 provides a time basis for the synchronous system's encoder and decoder. The stability of the clock frequency, therefore, is an important factor for correct operation. To obtain optimum performance, clock jitter must be reduced as much as possible. This is best achieved using the internal oscillator buffer with the recommended circuitry.

If an external clock source is used, the clock signal must be applied to pin OSCIN. In this case, special care must be taken with the clock duty cycle and clock jitter and the clock quality must be verified.

**8.8 Reset and oscillator start-up time**

**8.8.1 Reset timing requirements**

The reset signal is filtered by a hysteresis circuit and a spike filter before it enters the digital circuit. The spike filter rejects signals shorter than 10 ns. In order to perform a reset, the signal must be LOW for at least 100 ns.

**8.8.2 Oscillator start-up time**

If the MFRC522 has been set to a Power-down mode or is powered by a V<sub>DDX</sub> supply, the start-up time for the MFRC522 depends on the oscillator used and is shown in [Figure 23](#).

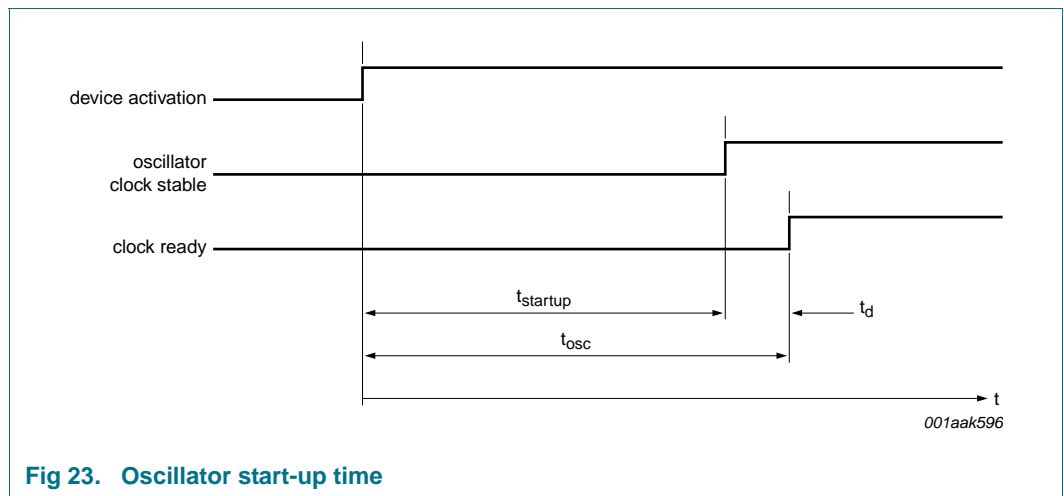
The time (t<sub>startup</sub>) is the start-up time of the crystal oscillator circuit. The crystal oscillator start-up time is defined by the crystal.

The time (t<sub>d</sub>) is the internal delay time of the MFRC522 when the clock signal is stable before the MFRC522 can be addressed.

The delay time is calculated by:

$$t_d = \frac{1024}{27 \mu s} = 37.74 \mu s \tag{8}$$

The time (t<sub>osc</sub>) is the sum of t<sub>d</sub> and t<sub>startup</sub>.



**Fig 23. Oscillator start-up time**

## 9. MFRC522 registers

### 9.1 Register bit behavior

Depending on the functionality of a register, the access conditions to the register can vary. In principle, bits with same behavior are grouped in common registers. The access conditions are described in [Table 19](#).

**Table 19. Behavior of register bits and their designation**

Abbreviation	Behavior	Description
R/W	read and write	These bits can be written and read by the microcontroller. Since they are used only for control purposes, their content is not influenced by internal state machines, for example the ComIEnReg register can be written and read by the microcontroller. It will also be read by internal state machines but never changed by them.
D	dynamic	These bits can be written and read by the microcontroller. Nevertheless, they can also be written automatically by internal state machines, for example the CommandReg register changes its value automatically after the execution of the command.
R	read only	These register bits hold values which are determined by internal states only, for example the CRCReady bit cannot be written externally but shows internal states.
W	write only	Reading these register bits always returns zero.
reserved	-	These registers are reserved for future use and must not be changed. In case of a write access, it is recommended to always write the value "0".
RFT	-	These register bits are reserved for future use or are for production tests and must not be changed.

## 9.2 Register overview

**Table 20. MFRC522 register overview**

Address (hex)	Register name	Function	Refer to
<b>Page 0: Command and status</b>			
00h	Reserved	reserved for future use	<a href="#">Table 21 on page 37</a>
01h	CommandReg	starts and stops command execution	<a href="#">Table 23 on page 37</a>
02h	ComIEnReg	enable and disable interrupt request control bits	<a href="#">Table 25 on page 37</a>
03h	DivIEnReg	enable and disable interrupt request control bits	<a href="#">Table 27 on page 38</a>
04h	ComIrqReg	interrupt request bits	<a href="#">Table 29 on page 38</a>
05h	DivIrqReg	interrupt request bits	<a href="#">Table 31 on page 39</a>
06h	ErrorReg	error bits showing the error status of the last command executed	<a href="#">Table 33 on page 40</a>
07h	Status1Reg	communication status bits	<a href="#">Table 35 on page 41</a>
08h	Status2Reg	receiver and transmitter status bits	<a href="#">Table 37 on page 42</a>
09h	FIFODataReg	input and output of 64 byte FIFO buffer	<a href="#">Table 39 on page 43</a>
0Ah	FIFOLevelReg	number of bytes stored in the FIFO buffer	<a href="#">Table 41 on page 43</a>
0Bh	WaterLevelReg	level for FIFO underflow and overflow warning	<a href="#">Table 43 on page 43</a>
0Ch	ControlReg	miscellaneous control registers	<a href="#">Table 45 on page 44</a>
0Dh	BitFramingReg	adjustments for bit-oriented frames	<a href="#">Table 47 on page 45</a>
0Eh	CollReg	bit position of the first bit-collision detected on the RF interface	<a href="#">Table 49 on page 45</a>
0Fh	Reserved	reserved for future use	<a href="#">Table 51 on page 46</a>
<b>Page 1: Command</b>			
10h	Reserved	reserved for future use	<a href="#">Table 53 on page 46</a>
11h	ModeReg	defines general modes for transmitting and receiving	<a href="#">Table 55 on page 47</a>
12h	TxModeReg	defines transmission data rate and framing	<a href="#">Table 57 on page 47</a>
13h	RxModeReg	defines reception data rate and framing	<a href="#">Table 59 on page 48</a>
14h	TxControlReg	controls the logical behavior of the antenna driver pins TX1 and TX2	<a href="#">Table 61 on page 49</a>
15h	TxASKReg	controls the setting of the transmission modulation	<a href="#">Table 63 on page 50</a>
16h	TxSelReg	selects the internal sources for the antenna driver	<a href="#">Table 65 on page 50</a>
17h	RxSelReg	selects internal receiver settings	<a href="#">Table 67 on page 51</a>
18h	RxThresholdReg	selects thresholds for the bit decoder	<a href="#">Table 69 on page 52</a>
19h	DemodReg	defines demodulator settings	<a href="#">Table 71 on page 52</a>
1Ah	Reserved	reserved for future use	<a href="#">Table 73 on page 53</a>
1Bh	Reserved	reserved for future use	<a href="#">Table 75 on page 53</a>
1Ch	MfTxReg	controls some MIFARE communication transmit parameters	<a href="#">Table 77 on page 54</a>
1Dh	MfRxReg	controls some MIFARE communication receive parameters	<a href="#">Table 79 on page 54</a>
1Eh	Reserved	reserved for future use	<a href="#">Table 81 on page 54</a>
1Fh	SerialSpeedReg	selects the speed of the serial UART interface	<a href="#">Table 83 on page 54</a>
<b>Page 2: Configuration</b>			
20h	Reserved	reserved for future use	<a href="#">Table 85 on page 56</a>

**Table 20. MFRC522 register overview ...continued**

Address (hex)	Register name	Function	Refer to
21h	CRCResultReg	shows the MSB and LSB values of the CRC calculation	<a href="#">Table 87 on page 56</a>
22h			<a href="#">Table 89 on page 56</a>
23h	Reserved	reserved for future use	<a href="#">Table 91 on page 57</a>
24h	ModWidthReg	controls the ModWidth setting	<a href="#">Table 93 on page 57</a>
25h	Reserved	reserved for future use	<a href="#">Table 95 on page 57</a>
26h	RFCfgReg	configures the receiver gain	<a href="#">Table 97 on page 58</a>
27h	GsNReg	selects the conductance of the antenna driver pins TX1 and TX2 for modulation	<a href="#">Table 99 on page 58</a>
28h	CWGsPReg	defines the conductance of the p-driver output during periods of no modulation	<a href="#">Table 101 on page 59</a>
29h	ModGsPReg	defines the conductance of the p-driver output during periods of modulation	<a href="#">Table 103 on page 59</a>
2Ah	TModeReg	defines settings for the internal timer	<a href="#">Table 105 on page 59</a>
2Bh	TPrescalerReg		<a href="#">Table 107 on page 60</a>
2Ch	TReloadReg	defines the 16-bit timer reload value	<a href="#">Table 109 on page 61</a>
2Dh			<a href="#">Table 111 on page 61</a>
2Eh	TCounterValReg	shows the 16-bit timer value	<a href="#">Table 113 on page 62</a>
2Fh			<a href="#">Table 115 on page 62</a>

**Page 3: Test register**

30h	Reserved	reserved for future use	<a href="#">Table 117 on page 62</a>
31h	TestSel1Reg	general test signal configuration	<a href="#">Table 119 on page 62</a>
32h	TestSel2Reg	general test signal configuration and PRBS control	<a href="#">Table 121 on page 63</a>
33h	TestPinEnReg	enables pin output driver on pins D1 to D7	<a href="#">Table 123 on page 63</a>
34h	TestPinValueReg	defines the values for D1 to D7 when it is used as an I/O bus	<a href="#">Table 125 on page 64</a>
35h	TestBusReg	shows the status of the internal test bus	<a href="#">Table 127 on page 64</a>
36h	AutoTestReg	controls the digital self test	<a href="#">Table 129 on page 65</a>
37h	VersionReg	shows the software version	<a href="#">Table 131 on page 65</a>
38h	AnalogTestReg	controls the pins AUX1 and AUX2	<a href="#">Table 133 on page 66</a>
39h	TestDAC1Reg	defines the test value for TestDAC1	<a href="#">Table 135 on page 67</a>
3Ah	TestDAC2Reg	defines the test value for TestDAC2	<a href="#">Table 137 on page 67</a>
3Bh	TestADCReg	shows the value of ADC I and Q channels	<a href="#">Table 139 on page 67</a>
3Ch to 3Fh	Reserved	reserved for production tests	<a href="#">Table 141 to Table 147 on page 68</a>

## 9.3 Register descriptions

### 9.3.1 Page 0: Command and status

#### 9.3.1.1 Reserved register 00h

Functionality is reserved for future use.

**Table 21. Reserved register (address 00h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 22. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	-	reserved

#### 9.3.1.2 CommandReg register

Starts and stops command execution.

**Table 23. CommandReg register (address 01h); reset value: 20h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol:	reserved		RcvOff	PowerDown	Command[3:0]			
Access:	-		R/W	D	D			

**Table 24. CommandReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 6	reserved	-	reserved for future use
5	RcvOff	1	analog part of the receiver is switched off
4	PowerDown	1	Soft power-down mode entered
		0	MFRC522 starts the wake up procedure during which this bit is read as a logic 1; it is read as a logic 0 when the MFRC522 is ready; see <a href="#">Section 8.6.2 on page 32</a>
			<b>Remark:</b> The PowerDown bit cannot be set when the SoftReset command is activated
3 to 0	Command[3:0]	-	activates a command based on the Command value; reading this register shows which command is executed; see <a href="#">Section 10.3 on page 69</a>

#### 9.3.1.3 ComIEnReg register

Control bits to enable and disable the passing of interrupt requests.

**Table 25. ComIEnReg register (address 02h); reset value: 80h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	IRqInv	TxIEn	RxIEn	IdleIEn	HiAlertIEn	LoAlertIEn	ErrIEn	TimerIEn
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W



**Table 26. ComIEnReg register bit descriptions**

Bit	Symbol	Value	Description
7	IRqInv	1	signal on pin IRQ is inverted with respect to the Status1Reg register's IRq bit
		0	signal on pin IRQ is equal to the IRq bit; in combination with the DivIEnReg register's IRqPushPull bit, the default value of logic 1 ensures that the output level on pin IRQ is 3-state
6	TxIEn	-	allows the transmitter interrupt request (TxIRq bit) to be propagated to pin IRQ
5	RxIEn	-	allows the receiver interrupt request (RxIRq bit) to be propagated to pin IRQ
4	IdleIEn	-	allows the idle interrupt request (IdleIRq bit) to be propagated to pin IRQ
3	HiAlertIEn	-	allows the high alert interrupt request (HiAlertIRq bit) to be propagated to pin IRQ
2	LoAlertIEn	-	allows the low alert interrupt request (LoAlertIRq bit) to be propagated to pin IRQ
1	ErrIEn	-	allows the error interrupt request (ErrIRq bit) to be propagated to pin IRQ
0	TimerIEn	-	allows the timer interrupt request (TimerIRq bit) to be propagated to pin IRQ

#### 9.3.1.4 DivIEnReg register

Control bits to enable and disable the passing of interrupt requests.

**Table 27. DivIEnReg register (address 03h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	IRQPushPull	reserved	MfinActIEn	reserved	CRCIEn	reserved		
Access	R/W	-	R/W	-	R/W	-		

**Table 28. DivIEnReg register bit descriptions**

Bit	Symbol	Value	Description
7	IRQPushPull	1	pin IRQ is a standard CMOS output pin
		0	pin IRQ is an open-drain output pin
6 to 5	reserved	-	reserved for future use
4	MfinActIEn	-	allows the MFIN active interrupt request to be propagated to pin IRQ
3	reserved	-	reserved for future use
2	CRCIEn	-	allows the CRC interrupt request, indicated by the DivIrqReg register's CRCIRq bit, to be propagated to pin IRQ
1 to 0	reserved	-	reserved for future use

#### 9.3.1.5 ComIrqReg register

Interrupt request bits.

**Table 29. ComIrqReg register (address 04h); reset value: 14h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	Set1	TxIRq	RxIRq	IdleIRq	HiAlertIRq	LoAlertIRq	ErrIRq	TimerIRq
Access	W	D	D	D	D	D	D	D

**Table 30. ComlRqReg register bit descriptions**

All bits in the ComlRqReg register are cleared by software.

Bit	Symbol	Value	Description
7	Set1	1	indicates that the marked bits in the ComlRqReg register are set
		0	indicates that the marked bits in the ComlRqReg register are cleared
6	TxIRq	1	set immediately after the last bit of the transmitted data was sent out
5	RxIRq	1	receiver has detected the end of a valid data stream if the RxModeReg register's RxNoErr bit is set to logic 1, the RxIRq bit is only set to logic 1 when data bytes are available in the FIFO
4	IdleIRq	1	If a command terminates, for example, when the CommandReg changes its value from any command to the Idle command (see <a href="#">Table 149 on page 69</a> ) if an unknown command is started, the CommandReg register Command[3:0] value changes to the idle state and the IdleIRq bit is set The microcontroller starting the Idle command does not set the IdleIRq bit
3	HiAlertIRq	1	the Status1Reg register's HiAlert bit is set in opposition to the HiAlert bit, the HiAlertIRq bit stores this event and can only be reset as indicated by the Set1 bit in this register
2	LoAlertIRq	1	Status1Reg register's LoAlert bit is set in opposition to the LoAlert bit, the LoAlertIRq bit stores this event and can only be reset as indicated by the Set1 bit in this register
1	ErrIRq	1	any error bit in the ErrorReg register is set
0	TimerIRq	1	the timer decrements the timer value in register TCounterValReg to zero

### 9.3.1.6 DivlRqReg register

Interrupt request bits.

**Table 31. DivlRqReg register (address 05h); reset value: x0h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	Set2	reserved	MfinActIRq	reserved	CRCIRq	reserved	reserved	reserved
Access	W	-	D	-	D	-	-	-

**Table 32. DivlRqReg register bit descriptions**

All bits in the DivlRqReg register are cleared by software.

Bit	Symbol	Value	Description
7	Set2	1	indicates that the marked bits in the DivlRqReg register are set
		0	indicates that the marked bits in the DivlRqReg register are cleared
6 to 5	reserved	-	reserved for future use
4	MfinActIRq	1	MFIN is active this interrupt is set when either a rising or falling signal edge is detected
3	reserved	-	reserved for future use
2	CRCIRq	1	the CalcCRC command is active and all data is processed
1 to 0	reserved	-	reserved for future use

### 9.3.1.7 ErrorReg register

Error bit register showing the error status of the last command executed.

**Table 33. ErrorReg register (address 06h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	WrErr	TempErr	reserved	BufferOvfl	CollErr	CRCErr	ParityErr	ProtocolErr
Access	R	R	-	R	R	R	R	R

**Table 34. ErrorReg register bit descriptions**

Bit	Symbol	Value	Description
7	WrErr	1	data is written into the FIFO buffer by the host during the MFAuthent command or if data is written into the FIFO buffer by the host during the time between sending the last bit on the RF interface and receiving the last bit on the RF interface
6	TempErr <sup>[1]</sup>	1	internal temperature sensor detects overheating, in which case the antenna drivers are automatically switched off
5	reserved	-	reserved for future use
4	BufferOvfl	1	the host or a MFRC522's internal state machine (e.g. receiver) tries to write data to the FIFO buffer even though it is already full
3	CollErr	1	a bit-collision is detected cleared automatically at receiver start-up phase only valid during the bitwise anticollision at 106 kBd always set to logic 0 during communication protocols at 212 kBd, 424 kBd and 848 kBd
2	CRCErr	1	the RxModeReg register's RxCRCEn bit is set and the CRC calculation fails automatically cleared to logic 0 during receiver start-up phase
1	ParityErr	1	parity check failed automatically cleared during receiver start-up phase only valid for ISO/IEC 14443 A/MIFARE communication at 106 kBd
0	ProtocolErr	1	set to logic 1 if the SOF is incorrect automatically cleared during receiver start-up phase bit is only valid for 106 kBd during the MFAuthent command, the ProtocolErr bit is set to logic 1 if the number of bytes received in one data stream is incorrect

[1] Command execution clears all error bits except the TempErr bit. Cannot be set by software.

### 9.3.1.8 Status1Reg register

Contains status bits of the CRC, interrupt and FIFO buffer.

**Table 35. Status1Reg register (address 07h); reset value: 21h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved	CRCOk	CRCReady	IRq	TRunning	reserved	HiAlert	LoAlert
Access	-	R	R	R	R	-	R	R

**Table 36. Status1Reg register bit descriptions**

Bit	Symbol	Value	Description
7	reserved	-	reserved for future use
6	CRCOk	1	the CRC result is zero for data transmission and reception, the CRCOk bit is undefined: use the ErrorReg register's CRCErr bit indicates the status of the CRC coprocessor, during calculation the value changes to logic 0, when the calculation is done correctly the value changes to logic 1
5	CRCReady	1	the CRC calculation has finished only valid for the CRC coprocessor calculation using the CalcCRC command
4	IRq	-	indicates if any interrupt source requests attention with respect to the setting of the interrupt enable bits: see the ComIEnReg and DivIEnReg registers
3	TRunning	1	MFRC522's timer unit is running, i.e. the timer will decrement the TCounterValReg register with the next timer clock <b>Remark:</b> in gated mode, the TRunning bit is set to logic 1 when the timer is enabled by TModeReg register's TGated[1:0] bits; this bit is not influenced by the gated signal
2	reserved	-	reserved for future use
1	HiAlert	1	the number of bytes stored in the FIFO buffer corresponds to equation: $HiAlert = (64 - FIFOLength) \leq WaterLevel$ example: FIFO length = 60, WaterLevel = 4 → HiAlert = 1 FIFO length = 59, WaterLevel = 4 → HiAlert = 0
0	LoAlert	1	the number of bytes stored in the FIFO buffer corresponds to equation: $LoAlert = FIFOLength \leq WaterLevel$ example: FIFO length = 4, WaterLevel = 4 → LoAlert = 1 FIFO length = 5, WaterLevel = 4 → LoAlert = 0

**9.3.1.9 Status2Reg register**

Contains status bits of the receiver, transmitter and data mode detector.

**Table 37. Status2Reg register (address 08h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TempSensClear	I <sup>2</sup> CForceHS	reserved		MFCrypto1On	ModemState[2:0]		
Access	R/W	R/W	-		D	R		

**Table 38. Status2Reg register bit descriptions**

Bit	Symbol	Value	Description
7	TempSensClear	1	clears the temperature error if the temperature is below the alarm limit of 125 °C
6	I <sup>2</sup> CForceHS	1	the I <sup>2</sup> C-bus input filter is set to the High-speed mode independent of the I <sup>2</sup> C-bus protocol
		0	the I <sup>2</sup> C-bus input filter is set to the I <sup>2</sup> C-bus protocol used
5 to 4	reserved	-	reserved
3	MFCrypto1On	-	indicates that the MIFARE Crypto1 unit is switched on and therefore all data communication with the card is encrypted can only be set to logic 1 by a successful execution of the MFAuthent command only valid in Read/Write mode for MIFARE standard cards this bit is cleared by software
2 to 0	ModemState[2:0]	-	shows the state of the transmitter and receiver state machines:
		000	idle
		001	wait for the BitFramingReg register's StartSend bit
		010	TxWait: wait until RF field is present if the TModeReg register's TxWaitRF bit is set to logic 1 the minimum time for TxWait is defined by the TxWaitReg register
		011	transmitting
		100	RxWait: wait until RF field is present if the TModeReg register's TxWaitRF bit is set to logic 1 the minimum time for RxWait is defined by the RxWaitReg register
		101	wait for data
		110	receiving

### 9.3.1.10 FIFODataReg register

Input and output of 64 byte FIFO buffer.

**Table 39. FIFODataReg register (address 09h); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	FIFOData[7:0]							
Access	D							

**Table 40. FIFODataReg register bit descriptions**

Bit	Symbol	Description
7 to 0	FIFOData[7:0]	data input and output port for the internal 64-byte FIFO buffer FIFO buffer acts as parallel in/parallel out converter for all serial data stream inputs and outputs

### 9.3.1.11 FIFOLevelReg register

Indicates the number of bytes stored in the FIFO.

**Table 41. FIFOLevelReg register (address 0Ah); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	FlushBuffer	FIFOLevel[6:0]						
Access	W	R						

**Table 42. FIFOLevelReg register bit descriptions**

Bit	Symbol	Value	Description
7	FlushBuffer	1	immediately clears the internal FIFO buffer's read and write pointer and ErrorReg register's BufferOvfl bit reading this bit always returns 0
6 to 0	FIFOLevel [6:0]	-	indicates the number of bytes stored in the FIFO buffer writing to the FIFODataReg register increments and reading decrements the FIFOLevel value

### 9.3.1.12 WaterLevelReg register

Defines the level for FIFO under- and overflow warning.

**Table 43. WaterLevelReg register (address 0Bh); reset value: 08h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		WaterLevel[5:0]					
Access	-		R/W					

**Table 44. WaterLevelReg register bit descriptions**

Bit	Symbol	Description
7 to 6	reserved	reserved for future use
5 to 0	WaterLevel [5:0]	<p>defines a warning level to indicate a FIFO buffer overflow or underflow:</p> <p>Status1Reg register's HiAlert bit is set to logic 1 if the remaining number of bytes in the FIFO buffer space is equal to, or less than the defined number of WaterLevel bytes</p> <p>Status1Reg register's LoAlert bit is set to logic 1 if equal to, or less than the WaterLevel bytes in the FIFO buffer</p> <p><b>Remark:</b> to calculate values for HiAlert and LoAlert see <a href="#">Section 9.3.1.8 on page 41</a>.</p>

**9.3.1.13 ControlReg register**

Miscellaneous control bits.

**Table 45. ControlReg register (address 0Ch); reset value: 10h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TStopNow	TStartNow	reserved			RxLastBits[2:0]		
Access	W	W	-			R		

**Table 46. ControlReg register bit descriptions**

Bit	Symbol	Value	Description
7	TStopNow	1	timer stops immediately reading this bit always returns it to logic 0
6	TStartNow	1	timer starts immediately reading this bit always returns it to logic 0
5 to 3	reserved	-	reserved for future use
2 to 0	RxLastBits[2:0]	-	indicates the number of valid bits in the last received byte if this value is 000b, the whole byte is valid

### 9.3.1.14 BitFramingReg register

Adjustments for bit-oriented frames.

**Table 47. BitFramingReg register (address 0Dh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	StartSend	RxAlign[2:0]			reserved	TxLastBits[2:0]		
Access	W	R/W			-	R/W		

**Table 48. BitFramingReg register bit descriptions**

Bit	Symbol	Value	Description
7	StartSend	1	starts the transmission of data only valid in combination with the Transceive command
6 to 4	RxAlign[2:0]		used for reception of bit-oriented frames: defines the bit position for the first bit received to be stored in the FIFO buffer example:
		0	LSB of the received bit is stored at bit position 0, the second received bit is stored at bit position 1
		1	LSB of the received bit is stored at bit position 1, the second received bit is stored at bit position 2
		7	LSB of the received bit is stored at bit position 7, the second received bit is stored in the next byte that follows at bit position 0  These bits are only to be used for bitwise anticollision at 106 kBd, for all other modes they are set to 0
3	reserved	-	reserved for future use
2 to 0	TxLastBits[2:0]	-	used for transmission of bit oriented frames: defines the number of bits of the last byte that will be transmitted 000b indicates that all bits of the last byte will be transmitted

### 9.3.1.15 CollReg register

Defines the first bit-collision detected on the RF interface.

**Table 49. CollReg register (address 0Eh); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	ValuesAfterColl	reserved	CollPosNotValid	CollPos[4:0]				
Access	R/W	-	R	R				

**Table 50. CollReg register bit descriptions**

Bit	Symbol	Value	Description
7	ValuesAfterColl	0	all received bits will be cleared after a collision only used during bitwise anticollision at 106 kBd, otherwise it is set to logic 1
6	reserved	-	reserved for future use
5	CollPosNotValid	1	no collision detected or the position of the collision is out of the range of CollPos[4:0]



**Table 50. CollReg register bit descriptions ...continued**

Bit	Symbol	Value	Description
4 to 0	CollPos[4:0]	-	shows the bit position of the first detected collision in a received frame only data bits are interpreted example:
		00h	indicates a bit-collision in the 32 <sup>nd</sup> bit
		01h	indicates a bit-collision in the 1 <sup>st</sup> bit
		08h	indicates a bit-collision in the 8 <sup>th</sup> bit
			These bits will only be interpreted if the CollPosNotValid bit is set to logic 0

**9.3.1.16 Reserved register 0Fh**

Functionality is reserved for future use.

**Table 51. Reserved register (address 0Fh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 52. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

**9.3.2 Page 1: Communication**

**9.3.2.1 Reserved register 10h**

Functionality is reserved for future use.

**Table 53. Reserved register (address 10h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 54. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

### 9.3.2.2 ModeReg register

Defines general mode settings for transmitting and receiving.

**Table 55. ModeReg register (address 11h); reset value: 3Fh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	MSBFirst	reserved	TxWaitRF	reserved	PolMFin	reserved	CRCPreset[1:0]	
Access	R/W	-	R/W	-	R/W	-	R/W	

**Table 56. ModeReg register bit descriptions**

Bit	Symbol	Value	Description
7	MSBFirst	1	CRC coprocessor calculates the CRC with MSB first in the CRCResultReg register the values for the CRCResultMSB[7:0] bits and the CRCResultLSB[7:0] bits are bit reversed <b>Remark:</b> during RF communication this bit is ignored
6	reserved	-	reserved for future use
5	TxWaitRF	1	transmitter can only be started if an RF field is generated
4	reserved	-	reserved for future use
3	PolMFin		defines the polarity of pin MFIN <b>Remark:</b> the internal envelope signal is encoded active LOW, changing this bit generates a MFinActIRq event
		1	polarity of pin MFIN is active HIGH
		0	polarity of pin MFIN is active LOW
2	reserved	-	reserved for future use
1 to 0	CRCPreset [1:0]		defines the preset value for the CRC coprocessor for the CalcCRC command <b>Remark:</b> during any communication, the preset values are selected automatically according to the definition of bits in the RxModeReg and TxModeReg registers
		00	0000h
		01	6363h
		10	A671h
		11	FFFFh

### 9.3.2.3 TxModeReg register

Defines the data rate during transmission.

**Table 57. TxModeReg register (address 12h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TxCRCEn	TxSpeed[2:0]			InvMod	reserved		
Access	R/W	D			R/W	-		

**Table 58. TxModeReg register bit descriptions**

Bit	Symbol	Value	Description
7	TxCRCEn	1	enables CRC generation during data transmission <b>Remark:</b> can only be set to logic 0 at 106 kBd
6 to 4	TxSpeed[2:0]		defines the bit rate during data transmission the MFRC522 handles transfer speeds up to 848 kBd
		000	106 kBd
		001	212 kBd
		010	424 kBd
		011	848 kBd
		100	reserved
		101	reserved
		110	reserved
		111	reserved
3	InvMod	1	modulation of transmitted data is inverted
2 to 0	reserved	-	reserved for future use

### 9.3.2.4 RxModeReg register

Defines the data rate during reception.

**Table 59. RxModeReg register (address 13h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	RxCRCEn	RxSpeed[2:0]			RxNoErr	RxMultiple	reserved	
Access	R/W	D			R/W	R/W	-	

**Table 60. RxModeReg register bit descriptions**

Bit	Symbol	Value	Description
7	RxCRCEn	1	enables the CRC calculation during reception <b>Remark:</b> can only be set to logic 0 at 106 kBd
6 to 4	RxSpeed[2:0]		defines the bit rate while receiving data the MFRC522 handles transfer speeds up to 848 kBd
		000	106 kBd
		001	212 kBd
		010	424 kBd
		011	848 kBd
		100	reserved
		101	reserved
		110	reserved
		111	reserved
3	RxNoErr	1	an invalid received data stream (less than 4 bits received) will be ignored and the receiver remains active

**Table 60. RxModeReg register bit descriptions ...continued**

Bit	Symbol	Value	Description
2	RxMultiple	0	receiver is deactivated after receiving a data frame
		1	able to receive more than one data frame only valid for data rates above 106 kBd in order to handle the polling command after setting this bit the Receive and Transceive commands will not terminate automatically. Multiple reception can only be deactivated by writing any command (except the Receive command) to the CommandReg register, or by the host clearing the bit if set to logic 1, an error byte is added to the FIFO buffer at the end of a received data stream which is a copy of the ErrorReg register value. For the MFRC522 version 2.0 the CRC status is reflected in the signal CRCOK, which indicates the actual status of the CRC coprocessor. For the MFRC522 version 1.0 the CRC status is reflected in the signal CRCErr.
1 to 0	reserved	-	reserved for future use

### 9.3.2.5 TxControlReg register

Controls the logical behavior of the antenna driver pins TX1 and TX2.

**Table 61. TxControlReg register (address 14h); reset value: 80h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	InvTx2RF On	InvTx1RF On	InvTx2RF Off	InvTx1RF Off	Tx2CW	reserved	Tx2RFEn	Tx1RFEn
Access	R/W	R/W	R/W	R/W	R/W	-	R/W	R/W

**Table 62. TxControlReg register bit descriptions**

Bit	Symbol	Value	Description
7	InvTx2RFOn	1	output signal on pin TX2 inverted when driver TX2 is enabled
6	InvTx1RFOn	1	output signal on pin TX1 inverted when driver TX1 is enabled
5	InvTx2RFOff	1	output signal on pin TX2 inverted when driver TX2 is disabled
4	InvTx1RFOff	1	output signal on pin TX1 inverted when driver TX1 is disabled
3	Tx2CW	1	output signal on pin TX2 continuously delivers the unmodulated 13.56 MHz energy carrier
		0	Tx2CW bit is enabled to modulate the 13.56 MHz energy carrier
2	reserved	-	reserved for future use
1	Tx2RFEn	1	output signal on pin TX2 delivers the 13.56 MHz energy carrier modulated by the transmission data
0	Tx1RFEn	1	output signal on pin TX1 delivers the 13.56 MHz energy carrier modulated by the transmission data

**9.3.2.6 TxASKReg register**

Controls transmit modulation settings.

**Table 63. TxASKReg register (address 15h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved	Force100ASK	reserved					
Access	-	R/W	-					

**Table 64. TxASKReg register bit descriptions**

Bit	Symbol	Value	Description
7	reserved	-	reserved for future use
6	Force100ASK	1	forces a 100 % ASK modulation independent of the ModGsPReg register setting
5 to 0	reserved	-	reserved for future use

**9.3.2.7 TxSelReg register**

Selects the internal sources for the analog module.

**Table 65. TxSelReg register (address 16h); reset value: 10h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol:	reserved		DriverSel[1:0]		MFOutSel[3:0]			
Access:	-		R/W		R/W			

**Table 66. TxSelReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 6	reserved	-	reserved for future use
5 to 4	DriverSel [1:0]	-	selects the input of drivers TX1 and TX2
		00	3-state; in soft power-down the drivers are only in 3-state mode if the DriverSel[1:0] value is set to 3-state mode
		01	modulation signal (envelope) from the internal encoder, Miller pulse encoded
		10	modulation signal (envelope) from pin MFIN
		11	HIGH; the HIGH level depends on the setting of bits InvTx1RFOOn/InvTx1RFOff and InvTx2RFOOn/InvTx2RFOff

Table 66. TxSelReg register bit descriptions ...continued

Bit	Symbol	Value	Description
3 to 0	MFOutSel [3:0]		selects the input for pin MFOUT
		0000	3-state
		0001	LOW
		0010	HIGH
		0011	test bus signal as defined by the TestSel1Reg register's TstBusBitSel[2:0] value
		0100	modulation signal (envelope) from the internal encoder, Miller pulse encoded
		0101	serial data stream to be transmitted, data stream before Miller encoder
		0110	reserved
		0111	serial data stream received, data stream after Manchester decoder
1000 to 1111		reserved	

### 9.3.2.8 RxSelReg register

Selects internal receiver settings.

Table 67. RxSelReg register (address 17h); reset value: 84h bit allocation

Bit	7	6	5	4	3	2	1	0
Symbol	UARTSel[1:0]			RxWait[5:0]				
Access	R/W			R/W				

Table 68. RxSelReg register bit descriptions

Bit	Symbol	Value	Description
7 to 6	UARTSel [1:0]		selects the input of the contactless UART
		00	constant LOW
		01	Manchester with subcarrier from pin MFIN
		10	modulated signal from the internal analog module, default
	11	NRZ coding without subcarrier from pin MFIN which is only valid for transfer speeds above 106 kBd	
5 to 0	RxWait [5:0]	-	<p>after data transmission the activation of the receiver is delayed for RxWait bit-clocks, during this 'frame guard time' any signal on pin RX is ignored</p> <p>this parameter is ignored by the Receive command</p> <p>all other commands, such as Transceive, MFAuthent use this parameter</p> <p>the counter starts immediately after the external RF field is switched on</p>

### 9.3.2.9 RxThresholdReg register

Selects thresholds for the bit decoder.

**Table 69. RxThresholdReg register (address 18h); reset value: 84h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	MinLevel[3:0]			reserved		CollLevel[2:0]		
Access	R/W			-		R/W		

**Table 70. RxThresholdReg register bit descriptions**

Bit	Symbol	Description
7 to 4	MinLevel [3:0]	defines the minimum signal strength at the decoder input that will be accepted if the signal strength is below this level it is not evaluated
3	reserved	reserved for future use
2 to 0	CollLevel [2:0]	defines the minimum signal strength at the decoder input that must be reached by the weaker half-bit of the Manchester encoded signal to generate a bit-collision relative to the amplitude of the stronger half-bit

### 9.3.2.10 DemodReg register

Defines demodulator settings.

**Table 71. DemodReg register (address 19h); reset value: 4Dh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	AddIQ[1:0]		FixIQ	TPrescal Even	TauRcv[1:0]		TauSync[1:0]	
Access	R/W		R/W	R/W	R/W		R/W	

**Table 72. DemodReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 6	AddIQ [1:0]	-	defines the use of I and Q channel during reception <b>Remark:</b> the FixIQ bit must be set to logic 0 to enable the following settings:
		00	selects the stronger channel
		01	selects the stronger channel and freezes the selected channel during communication
		10	reserved
		11	reserved
5	FixIQ	1	if AddIQ[1:0] are set to X0b, the reception is fixed to I channel if AddIQ[1:0] are set to X1b, the reception is fixed to Q channel

**Table 72. DemodReg register bit descriptions ...continued**

Bit	Symbol	Value	Description
4	TPrescalEven	R/W	Available on RC522 version 1.0 and version 2.0: If set to logic 0 the following formula is used to calculate the timer frequency of the prescaler: $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 1)$ . Only available on version 2.0: If set to logic 1 the following formula is used to calculate the timer frequency of the prescaler: $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 2)$ . Default TPrescalEven bit is logic 0, find more information on the prescaler in <a href="#">Section 8.5</a> .
3 to 2	TauRcv[1:0]	-	changes the time-constant of the internal PLL during data reception <b>Remark:</b> if set to 00b the PLL is frozen during data reception
1 to 0	TauSync[1:0]	-	changes the time-constant of the internal PLL during burst

**9.3.2.11 Reserved register 1Ah**

Functionality is reserved for future use.

**Table 73. Reserved register (address 1Ah); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 74. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

**9.3.2.12 Reserved register 1Bh**

Functionality is reserved for future use.

**Table 75. Reserved register (address 1Bh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 76. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

**9.3.2.13 MfTxReg register**

Controls some MIFARE communication transmit parameters.



**Table 77. MfTxReg register (address 1Ch); reset value: 62h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved						TxWait[1:0]	
Access	-						R/W	

**Table 78. MfTxReg register bit descriptions**

Bit	Symbol	Description
7 to 2	reserved	reserved for future use
1 to 0	TxWait	defines the additional response time 7 bits are added to the value of the register bit by default

### 9.3.2.14 MfRxReg register

**Table 79. MfRxReg register (address 1Dh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved			ParityDisable	reserved			
Access	-			R/W	-			

**Table 80. MfRxReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 5	reserved	-	reserved for future use
4	ParityDisable	1	generation of the parity bit for transmission and the parity check for receiving is switched off the received parity bit is handled like a data bit
3 to 0	reserved	-	reserved for future use

### 9.3.2.15 Reserved register 1Eh

Functionality is reserved for future use.

**Table 81. Reserved register (address 1Eh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 82. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

### 9.3.2.16 SerialSpeedReg register

Selects the speed of the serial UART interface.

**Table 83. SerialSpeedReg register (address 1Fh); reset value: EBh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	BR_T0[2:0]				BR_T1[4:0]			
Access	R/W				R/W			

**Table 84. SerialSpeedReg register bit descriptions**

Bit	Symbol	Description
7 to 5	BR_T0[2:0]	factor BR_T0 adjusts the transfer speed: for description, see <a href="#">Section 8.1.3.2 on page 12</a>
4 to 0	BR_T1[4:0]	factor BR_T1 adjusts the transfer speed: for description, see <a href="#">Section 8.1.3.2 on page 12</a>

### 9.3.3 Page 2: Configuration

#### 9.3.3.1 Reserved register 20h

Functionality is reserved for future use.

**Table 85. Reserved register (address 20h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	-							
Access	reserved							

**Table 86. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

#### 9.3.3.2 CRCResultReg registers

Shows the MSB and LSB values of the CRC calculation.

**Remark:** The CRC is split into two 8-bit registers.

**Table 87. CRCResultReg (higher bits) register (address 21h); reset value: FFh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	CRCResultMSB[7:0]							
Access	R							

**Table 88. CRCResultReg register higher bit descriptions**

Bit	Symbol	Description
7 to 0	CRCResultMSB [7:0]	shows the value of the CRCResultReg register's most significant byte only valid if Status1Reg register's CRCReady bit is set to logic 1

**Table 89. CRCResultReg (lower bits) register (address 22h); reset value: FFh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	CRCResultLSB[7:0]							
Access	R							

**Table 90. CRCResultReg register lower bit descriptions**

Bit	Symbol	Description
7 to 0	CRCResultLSB [7:0]	shows the value of the least significant byte of the CRCResultReg register only valid if Status1Reg register's CRCReady bit is set to logic 1

### 9.3.3.3 Reserved register 23h

Functionality is reserved for future use.

**Table 91. Reserved register (address 23h); reset value: 88h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 92. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

### 9.3.3.4 ModWidthReg register

Sets the modulation width.

**Table 93. ModWidthReg register (address 24h); reset value: 26h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	ModWidth[7:0]							
Access	R/W							

**Table 94. ModWidthReg register bit descriptions**

Bit	Symbol	Description
7 to 0	ModWidth[7:0]	defines the width of the Miller modulation as multiples of the carrier frequency ( $\text{ModWidth} + 1 / f_{\text{clk}}$ ) the maximum value is half the bit period

### 9.3.3.5 Reserved register 25h

Functionality is reserved for future use.

**Table 95. Reserved register (address 25h); reset value: 87h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 96. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

### 9.3.3.6 RFCfgReg register

Configures the receiver gain.

**Table 97. RFCfgReg register (address 26h); reset value: 48h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved	RxGain[2:0]			reserved			
Access	-	R/W			-			

**Table 98. RFCfgReg register bit descriptions**

Bit	Symbol	Value	Description
7	reserved	-	reserved for future use
6 to 4	RxGain [2:0]		defines the receiver's signal voltage gain factor:
		000	18 dB
		001	23 dB
		010	18 dB
		011	23 dB
		100	33 dB
		101	38 dB
		110	43 dB
		111	48 dB
3 to 0	reserved	-	reserved for future use

### 9.3.3.7 GsNReg register

Defines the conductance of the antenna driver pins TX1 and TX2 for the n-driver when the driver is switched on.

**Table 99. GsNReg register (address 27h); reset value: 88h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	CWGsN[3:0]				ModGsN[3:0]			
Access	R/W				R/W			

**Table 100. GsNReg register bit descriptions**

Bit	Symbol	Description
7 to 4	CWGsn [3:0]	<p>defines the conductance of the output n-driver during periods without modulation which can be used to regulate the output power and subsequently current consumption and operating distance</p> <p><b>Remark:</b> the conductance value is binary-weighted during soft Power-down mode the highest bit is forced to logic 1 value is only used if driver TX1 or TX2 is switched on</p>
3 to 0	ModGsN [3:0]	<p>defines the conductance of the output n-driver during periods without modulation which can be used to regulate the modulation index</p> <p><b>Remark:</b> the conductance value is binary weighted during soft Power-down mode the highest bit is forced to logic 1 value is only used if driver TX1 or TX2 is switched on</p>

### 9.3.3.8 CWGsPReg register

Defines the conductance of the p-driver output during periods of no modulation.

**Table 101. CWGsPReg register (address 28h); reset value: 20h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		CWGsP[5:0]					
Access	-		R/W					

**Table 102. CWGsPReg register bit descriptions**

Bit	Symbol	Description
7 to 6	reserved	reserved for future use
5 to 0	CWGsP[5:0]	defines the conductance of the p-driver output which can be used to regulate the output power and subsequently current consumption and operating distance <b>Remark:</b> the conductance value is binary weighted during soft Power-down mode the highest bit is forced to logic 1

### 9.3.3.9 ModGsPReg register

Defines the conductance of the p-driver output during modulation.

**Table 103. ModGsPReg register (address 29h); reset value: 20h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		ModGsP[5:0]					
Access	-		R/W					

**Table 104. ModGsPReg register bit descriptions**

Bit	Symbol	Description
7 to 6	reserved	reserved for future use
5 to 0	ModGsP[5:0]	defines the conductance of the p-driver output during modulation which can be used to regulate the modulation index <b>Remark:</b> the conductance value is binary weighted during soft Power-down mode the highest bit is forced to logic 1 if the TxASKReg register's Force100ASK bit is set to logic 1 the value of ModGsP has no effect

### 9.3.3.10 TModeReg and TPrescalerReg registers

These registers define the timer settings.

**Remark:** The TPrescaler setting higher 4 bits are in the TModeReg register and the lower 8 bits are in the TPrescalerReg register.

**Table 105. TModeReg register (address 2Ah); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TAuto	TGated[1:0]		TAutoRestart	TPrescaler_Hi[3:0]			
Access	R/W	R/W		R/W	R/W			

**Table 106. TModeReg register bit descriptions**

Bit	Symbol	Value	Description
7	TAuto	1	timer starts automatically at the end of the transmission in all communication modes at all speeds  if the RxModeReg register's RxMultiple bit is not set, the timer stops immediately after receiving the 5th bit (1 start bit, 4 data bits)  if the RxMultiple bit is set to logic 1 the timer never stops, in which case the timer can be stopped by setting the ControlReg register's TStopNow bit to logic 1
		0	indicates that the timer is not influenced by the protocol
6 to 5	TGated[1:0]		internal timer is running in gated mode  <b>Remark:</b> in gated mode, the Status1Reg register's TRunning bit is logic 1 when the timer is enabled by the TModeReg register's TGated[1:0] bits  this bit does not influence the gating signal
		00	non-gated mode
		01	gated by pin MFIN
		10	gated by pin AUX1
		11	-
4	TAutoRestart	1	timer automatically restarts its count-down from the 16-bit timer reload value instead of counting down to zero
		0	timer decrements to 0 and the ComlRqReg register's TimerIRq bit is set to logic 1
3 to 0	TPrescaler_Hi[3:0]	-	defines the higher 4 bits of the TPrescaler value  The following formula is used to calculate the timer frequency if the DemodReg register's TPrescalEven bit in Demot Regis set to logic 0:  $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 1).$ Where TPreScaler = [TPrescaler_Hi:TPrescaler_Lo] (TPrescaler value on 12 bits) (Default TPrescalEven bit is logic 0)  The following formula is used to calculate the timer frequency if the DemodReg register's TPrescalEven bit is set to logic 1:  $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 2).$  See <a href="#">Section 8.5 "Timer unit"</a> .

**Table 107. TPrescalerReg register (address 2Bh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TPrescaler_Lo[7:0]							
Access	R/W							

**Table 108. TPrescalerReg register bit descriptions**

Bit	Symbol	Description
7 to 0	TPrescaler_Lo[7:0]	<p>defines the lower 8 bits of the TPrescaler value</p> <p>The following formula is used to calculate the timer frequency if the DemodReg register's TPrescalEven bit is set to logic 0:</p> $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 1).$ <p>Where TPreScaler = [TPrescaler_Hi:TPrescaler_Lo] (TPrescaler value on 12 bits) (Default TPrescalEven bit is logic 0)</p> <p>The following formula is used to calculate the timer frequency if the DemodReg register's TPrescalEven bit in DemoReg is set to logic 1:</p> $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 2).$ <p>See <a href="#">Section 8.5 "Timer unit"</a>.</p>

**9.3.3.11 TReloadReg register**

Defines the 16-bit timer reload value.

**Remark:** The reload value bits are contained in two 8-bit registers.

**Table 109. TReloadReg (higher bits) register (address 2Ch); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TReloadVal_Hi[7:0]							
Access	R/W							

**Table 110. TReloadReg register higher bit descriptions**

Bit	Symbol	Description
7 to 0	TReloadVal_Hi[7:0]	<p>defines the higher 8 bits of the 16-bit timer reload value</p> <p>on a start event, the timer loads the timer reload value</p> <p>changing this register affects the timer only at the next start event</p>

**Table 111. TReloadReg (lower bits) register (address 2Dh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TReloadVal_Lo[7:0]							
Access	R/W							

**Table 112. TReloadReg register lower bit descriptions**

Bit	Symbol	Description
7 to 0	TReloadVal_Lo[7:0]	<p>defines the lower 8 bits of the 16-bit timer reload value</p> <p>on a start event, the timer loads the timer reload value</p> <p>changing this register affects the timer only at the next start event</p>

**9.3.3.12 TCounterValReg register**

Contains the timer value.

**Remark:** The timer value bits are contained in two 8-bit registers.



**Table 113. TCounterValReg (higher bits) register (address 2Eh); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TCounterVal_Hi[7:0]							
Access	R							

**Table 114. TCounterValReg register higher bit descriptions**

Bit	Symbol	Description
7 to 0	TCounterVal_Hi [7:0]	timer value higher 8 bits

**Table 115. TCounterValReg (lower bits) register (address 2Fh); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TCounterVal_Lo[7:0]							
Access	R							

**Table 116. TCounterValReg register lower bit descriptions**

Bit	Symbol	Description
7 to 0	TCounterVal_Lo [7:0]	timer value lower 8 bits

### 9.3.4 Page 3: Test

#### 9.3.4.1 Reserved register 30h

Functionality is reserved for future use.

**Table 117. Reserved register (address 30h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 118. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

#### 9.3.4.2 TestSel1Reg register

General test signal configuration.

**Table 119. TestSel1Reg register (address 31h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved					TstBusBitSel[2:0]		
Access	-					R/W		

**Table 120. TestSel1Reg register bit descriptions**

Bit	Symbol	Description
7 to 3	reserved	reserved for future use
2 to 0	TstBusBitSel [2:0]	selects a test bus signal which is output at pin MFOUT if AnalogSelAux2[3:0] = FFh in AnalogTestReg register, test bus signal is also output at pins AUX1 or AUX2

**9.3.4.3 TestSel2Reg register**

General test signal configuration and PRBS control.

**Table 121. TestSel2Reg register (address 32h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TstBusFlip	PRBS9	PRBS15	TestBusSel[4:0]				
Access	R/W	R/W	R/W	R/W				

**Table 122. TestSel2Reg register bit descriptions**

Bit	Symbol	Value	Description
7	TstBusFlip	1	test bus is mapped to the parallel port in the following order: TstBusBit4, TstBusBit3, TstBusBit2, TstBusBit6, TstBusBit5, TstBusBit0; see <a href="#">Section 16.1 on page 81</a>
6	PRBS9	-	starts and enables the PRBS9 sequence according to ITU-TO150 <b>Remark:</b> all relevant registers to transmit data must be configured before entering PRBS9 mode the data transmission of the defined sequence is started by the Transmit command
5	PRBS15	-	starts and enables the PRBS15 sequence according to ITU-TO150 <b>Remark:</b> all relevant registers to transmit data must be configured before entering PRBS15 mode the data transmission of the defined sequence is started by the Transmit command
4 to 0	TestBusSel[4:0]	-	selects the test bus; see <a href="#">Section 16.1 “Test signals”</a>

**9.3.4.4 TestPinEnReg register**

Enables the test bus pin output driver.

**Table 123. TestPinEnReg register (address 33h); reset value: 80h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	RS232LineEn	TestPinEn[5:0]					reserved	
Access	R/W	R/W					-	

**Table 124. TestPinEnReg register bit descriptions**

Bit	Symbol	Value	Description
7	RS232LineEn	0	serial UART lines MX and DTRQ are disabled
6 to 1	TestPinEn [5:0]	-	enables the output driver on one of the data pins D1 to D7 which outputs a test signal  <b>Example:</b> setting bit 1 to logic 1 enables pin D1 output setting bit 5 to logic 1 enables pin D5 output  <b>Remark:</b> If the SPI is used, only pins D1 to D4 can be used. If the serial UART interface is used and the RS232LineEn bit is set to logic 1 only pins D1 to D4 can be used.
0	reserved	-	reserved for future use

#### 9.3.4.5 TestPinValueReg register

Defines the HIGH and LOW values for the test port D1 to D7 when it is used as I/O.

**Table 125. TestPinValueReg register (address 34h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	UseIO	TestPinValue[5:0]						reserved
Access	R/W	R/W						-

**Table 126. TestPinValueReg register bit descriptions**

Bit	Symbol	Value	Description
7	UseIO	1	enables the I/O functionality for the test port when one of the serial interfaces is used  the input/output behavior is defined by value TestPinEn[5:0] in the TestPinEnReg register  the value for the output behavior is defined by TestPinValue[5:0]
6 to 1	TestPinValue [5:0]	-	defines the value of the test port when it is used as I/O and each output must be enabled by TestPinEn[5:0] in the TestPinEnReg register  <b>Remark:</b> Reading the register indicates the status of pins D6 to D1 if the UseIO bit is set to logic 1. If the UseIO bit is set to logic 0, the value of the TestPinValueReg register is read back.
0	reserved	-	reserved for future use

#### 9.3.4.6 TestBusReg register

Shows the status of the internal test bus.

**Table 127. TestBusReg register (address 35h); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TestBus[7:0]							
Access	R							

**Table 128. TestBusReg register bit descriptions**

Bit	Symbol	Description
7 to 0	TestBus[7:0]	shows the status of the internal test bus the test bus is selected using the TestSel2Reg register; see <a href="#">Section 16.1 on page 81</a>

**9.3.4.7 AutoTestReg register**

Controls the digital self-test.

**Table 129. AutoTestReg register (address 36h); reset value: 40h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved	AmpRcv	RFT		SelfTest[3:0]			
Access	-	R/W	-		R/W			

**Table 130. AutoTestReg register bit descriptions**

Bit	Symbol	Value	Description
7	reserved	-	reserved for production tests
6	AmpRcv	1	internal signal processing in the receiver chain is performed non-linearly which increases the operating distance in communication modes at 106 kBd <b>Remark:</b> due to non-linearity, the effect of the RxThresholdReg register's MinLevel[3:0] and the CollLevel[2:0] values is also non-linear
5 to 4	RFT	-	reserved for production tests
3 to 0	SelfTest[3:0]	-	enables the digital self test the self test can also be started by the CalcCRC command; see <a href="#">Section 10.3.1.4 on page 70</a> the self test is enabled by value 1001b <b>Remark:</b> for default operation the self test must be disabled by value 0000b

**9.3.4.8 VersionReg register**

Shows the MFRC522 software version.

**Table 131. VersionReg register (address 37h); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	Version[7:0]							
Access	R							

**Table 132. VersionReg register bit descriptions**

Bit	Symbol	Description
7 to 4	Chiptype	'9' stands for MFRC522
3 to 0	Version	'1' stands for MFRC522 version 1.0 and '2' stands for MFRC522 version 2.0.

MFRC522 version 1.0 software version is: 91h.

MFRC522 version 2.0 software version is: 92h.

### 9.3.4.9 AnalogTestReg register

Determines the analog output test signal at, and status of, pins AUX1 and AUX2.

**Table 133. AnalogTestReg register (address 38h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	AnalogSelAux1[3:0]				AnalogSelAux2[3:0]			
Access	R/W				R/W			

**Table 134. AnalogTestReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 4	AnalogSelAux1 [3:0]		controls pin AUX1
		0000	3-state
		0001	output of TestDAC1 (AUX1), output of TestDAC2 (AUX2) <sup>[1]</sup>
		0010	test signal Corr1 <sup>[1]</sup>
		0011	reserved
		0100	DAC: test signal MinLevel <sup>[1]</sup>
		0101	DAC: test signal ADC_I <sup>[1]</sup>
		0110	DAC: test signal ADC_Q <sup>[1]</sup>
		0111	reserved
		1000	reserved, test signal for production test <sup>[1]</sup>
		1001	reserved
		1010	HIGH
		1011	LOW
		1100	TxActive: at 106 kBd: HIGH during Start bit, Data bit, Parity and CRC at 212 kBd: 424 kBd and 848 kBd: HIGH during data and CRC
		1101	RxActive: at 106 kBd: HIGH during Data bit, Parity and CRC at 212 kBd: 424 kBd and 848 kBd: HIGH during data and CRC
		1110	subcarrier detected: 106 kBd: not applicable 212 kBd: 424 kBd and 848 kBd: HIGH during last part of data and CRC
		1111	test bus bit as defined by the TestSel1Reg register's TstBusBitSel[2:0] bits <b>Remark:</b> all test signals are described in <a href="#">Section 16.1 on page 81</a>
3 to 0	AnalogSelAux2 [3:0]	-	controls pin AUX2 (see bit descriptions for AUX1)

[1] **Remark:** Current source output; the use of 1 kΩ pull-down resistor on AUXn is recommended.

### 9.3.4.10 TestDAC1Reg register

Defines the test value for TestDAC1.

**Table 135. TestDAC1Reg register (address 39h); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		TestDAC1[5:0]					
Access	-		R/W					

**Table 136. TestDAC1Reg register bit descriptions**

Bit	Symbol	Description
7	reserved	reserved for production tests
6	reserved	reserved for future use
5 to 0	TestDAC1[5:0]	defines the test value for TestDAC1 output of DAC1 can be routed to AUX1 by setting value AnalogSelAux1[3:0] to 0001b in the AnalogTestReg register

### 9.3.4.11 TestDAC2Reg register

Defines the test value for TestDAC2.

**Table 137. TestDAC2Reg register (address 3Ah); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		TestDAC2[5:0]					
Access	-		R/W					

**Table 138. TestDAC2Reg register bit descriptions**

Bit	Symbol	Description
7 to 6	reserved	reserved for future use
5 to 0	TestDAC2[5:0]	defines the test value for TestDAC2 output of DAC2 can be routed to AUX2 by setting value AnalogSelAux2[3:0] to 0001b in the AnalogTestReg register

### 9.3.4.12 TestADCReg register

Shows the values of ADC I and Q channels.

**Table 139. TestADCReg register (address 3Bh); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	ADC_I[3:0]				ADC_Q[3:0]			
Access	R				R			

**Table 140. TestADCReg register bit descriptions**

Bit	Symbol	Description
7 to 4	ADC_I[3:0]	ADC I channel value
3 to 0	ADC_Q[3:0]	ADC Q channel value

### 9.3.4.13 Reserved register 3Ch

Functionality reserved for production test.

**Table 141. Reserved register (address 3Ch); reset value: FFh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol					RFT			
Access					-			

**Table 142. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for production tests

**Table 143. Reserved register (address 3Dh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol					RFT			
Access					-			

**Table 144. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for production tests

**Table 145. Reserved register (address 3Eh); reset value: 03h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol					RFT			
Access					-			

**Table 146. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for production tests

**Table 147. Reserved register (address 3Fh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol					reserved			
Access					-			

**Table 148. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for production tests

## 10. MFRC522 command set

### 10.1 General description

The MFRC522 operation is determined by a state machine capable of performing a set of commands. A command is executed by writing a command code (see [Table 149](#)) to the CommandReg register.

Arguments and/or data necessary to process a command are exchanged via the FIFO buffer.

### 10.2 General behavior

- Each command that needs a data bit stream (or data byte stream) as an input immediately processes any data in the FIFO buffer. An exception to this rule is the Transceive command. Using this command, transmission is started with the BitFramingReg register's StartSend bit.
- Each command that needs a certain number of arguments, starts processing only when it has received the correct number of arguments from the FIFO buffer.
- The FIFO buffer is not automatically cleared when commands start. This makes it possible to write command arguments and/or the data bytes to the FIFO buffer and then start the command.
- Each command can be interrupted by the host writing a new command code to the CommandReg register, for example, the Idle command.

### 10.3 MFRC522 command overview

**Table 149. Command overview**

Command	Command code	Action
Idle	0000	no action, cancels current command execution
Mem	0001	stores 25 bytes into the internal buffer
Generate RandomID	0010	generates a 10-byte random ID number
CalcCRC	0011	activates the CRC coprocessor or performs a self test
Transmit	0100	transmits data from the FIFO buffer
NoCmdChange	0111	no command change, can be used to modify the CommandReg register bits without affecting the command, for example, the PowerDown bit
Receive	1000	activates the receiver circuits
Transceive	1100	transmits data from FIFO buffer to antenna and automatically activates the receiver after transmission
-	1101	reserved for future use
MFAuthent	1110	performs the MIFARE standard authentication as a reader
SoftReset	1111	resets the MFRC522



### 10.3.1 MFRC522 command descriptions

#### 10.3.1.1 Idle

Places the MFRC522 in Idle mode. The Idle command also terminates itself.

#### 10.3.1.2 Mem

Transfers 25 bytes from the FIFO buffer to the internal buffer.

To read out the 25 bytes from the internal buffer the Mem command must be started with an empty FIFO buffer. In this case, the 25 bytes are transferred from the internal buffer to the FIFO.

During a hard power-down (using pin NRSTPD), the 25 bytes in the internal buffer remain unchanged and are only lost if the power supply is removed from the MFRC522.

This command automatically terminates when finished and the Idle command becomes active.

#### 10.3.1.3 Generate RandomID

This command generates a 10-byte random number which is initially stored in the internal buffer. This then overwrites the 10 bytes in the internal 25-byte buffer. This command automatically terminates when finished and the MFRC522 returns to Idle mode.

#### 10.3.1.4 CalcCRC

The FIFO buffer content is transferred to the CRC coprocessor and the CRC calculation is started. The calculation result is stored in the CRCResultReg register. The CRC calculation is not limited to a dedicated number of bytes. The calculation is not stopped when the FIFO buffer is empty during the data stream. The next byte written to the FIFO buffer is added to the calculation.

The CRC preset value is controlled by the ModeReg register's CRCPreset[1:0] bits. The value is loaded in to the CRC coprocessor when the command starts.

This command must be terminated by writing a command to the CommandReg register, such as, the Idle command.

If the AutoTestReg register's SelfTest[3:0] bits are set correctly, the MFRC522 enters Self Test mode. Starting the CalcCRC command initiates a digital self test. The result of the self test is written to the FIFO buffer.

#### 10.3.1.5 Transmit

The FIFO buffer content is immediately transmitted after starting this command. Before transmitting the FIFO buffer content, all relevant registers must be set for data transmission.

This command automatically terminates when the FIFO buffer is empty. It can be terminated by another command written to the CommandReg register.

#### 10.3.1.6 NoCmdChange

This command does not influence any running command in the CommandReg register. It can be used to manipulate any bit except the CommandReg register Command[3:0] bits, for example, the RcvOff bit or the PowerDown bit.

### 10.3.1.7 Receive

The MFRC522 activates the receiver path and waits for a data stream to be received. The correct settings must be chosen before starting this command.

This command automatically terminates when the data stream ends. This is indicated either by the end of frame pattern or by the length byte depending on the selected frame type and speed.

**Remark:** If the RxModeReg register's RxMultiple bit is set to logic 1, the Receive command will not automatically terminate. It must be terminated by starting another command in the CommandReg register.

### 10.3.1.8 Transceive

This command continuously repeats the transmission of data from the FIFO buffer and the reception of data from the RF field. The first action is transmit and after transmission the command is changed to receive a data stream.

Each transmit process must be started by setting the BitFramingReg register's StartSend bit to logic 1. This command must be cleared by writing any command to the CommandReg register.

**Remark:** If the RxModeReg register's RxMultiple bit is set to logic 1, the Transceive command never leaves the receive state because this state cannot be cancelled automatically.

### 10.3.1.9 MFAuthent

This command manages MIFARE authentication to enable a secure communication to any MIFARE Mini, MIFARE 1K and MIFARE 4K card. The following data is written to the FIFO buffer before the command can be activated:

- Authentication command code (60h, 61h)
- Block address
- Sector key byte 0
- Sector key byte 1
- Sector key byte 2
- Sector key byte 3
- Sector key byte 4
- Sector key byte 5
- Card serial number byte 0
- Card serial number byte 1
- Card serial number byte 2
- Card serial number byte 3

In total 12 bytes are written to the FIFO.

**Remark:** When the MFAuthent command is active all access to the FIFO buffer is blocked. However, if there is access to the FIFO buffer, the ErrorReg register's WrErr bit is set.

This command automatically terminates when the MIFARE card is authenticated and the Status2Reg register's MFCrypto1On bit is set to logic 1.

This command does not terminate automatically if the card does not answer, so the timer must be initialized to automatic mode. In this case, in addition to the IdleIRq bit, the TimerIRq bit can be used as the termination criteria. During authentication processing, the RxIRq bit and TxIRq bit are blocked. The Crypto1On bit is only valid after termination of the MFAuthent command, either after processing the protocol or writing Idle to the CommandReg register.

If an error occurs during authentication, the ErrorReg register's ProtocolErr bit is set to logic 1 and the Status2Reg register's Crypto1On bit is set to logic 0.

#### 10.3.1.10 SoftReset

This command performs a reset of the device. The configuration data of the internal buffer remains unchanged. All registers are set to the reset values. This command automatically terminates when finished.

**Remark:** The SerialSpeedReg register is reset and therefore the serial data rate is set to 9.6 kBd.

## 11. Limiting values

**Table 150. Limiting values**

In accordance with the Absolute Maximum Rating System (IEC 60134).

Symbol	Parameter	Conditions	Min	Max	Unit
V <sub>DDA</sub>	analog supply voltage		-0.5	+4.0	V
V <sub>DDD</sub>	digital supply voltage		-0.5	+4.0	V
V <sub>DD(PVDD)</sub>	PVDD supply voltage		-0.5	+4.0	V
V <sub>DD(TVDD)</sub>	TVDD supply voltage		-0.5	+4.0	V
V <sub>DD(SVDD)</sub>	SVDD supply voltage		-0.5	+4.0	V
V <sub>i</sub>	input voltage	all input pins except pins MFIN and RX	V <sub>SS(PVSS)</sub> - 0.5	V <sub>DD(PVDD)</sub> + 0.5	V
		pin MFIN	V <sub>SS(PVSS)</sub> - 0.5	V <sub>DD(SVDD)</sub> + 0.5	V
P <sub>tot</sub>	total power dissipation	per package; and V <sub>DDD</sub> in shortcut mode	-	200	mW
T <sub>j</sub>	junction temperature		-	100	°C
V <sub>ESD</sub>	electrostatic discharge voltage	HBM; 1500 Ω, 100 pF; JESD22-A114-B	-	2000	V
		MM; 0.75 μH, 200 pF; JESD22-A114-A	-	200	V

## 12. Recommended operating conditions

**Table 151. Operating conditions**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V <sub>DDA</sub>	analog supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ; V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	[1][2] 2.5	3.3	3.6	V
V <sub>DDD</sub>	digital supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ; V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	[1][2] 2.5	3.3	3.6	V
V <sub>DD(TVDD)</sub>	TVDD supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ; V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	[1][2] 2.5	3.3	3.6	V
V <sub>DD(PVDD)</sub>	PVDD supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ; V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	[3] 1.6	1.8	3.6	V
V <sub>DD(SVDD)</sub>	SVDD supply voltage	V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	1.6	-	3.6	V
T <sub>amb</sub>	ambient temperature	HVQFN32	-25	-	+85	°C

[1] Supply voltages below 3 V reduce the performance (the achievable operating distance).

[2] V<sub>DDA</sub>, V<sub>DDD</sub> and V<sub>DD(TVDD)</sub> must always be the same voltage.

[3] V<sub>DD(PVDD)</sub> must always be the same or lower voltage than V<sub>DDD</sub>.

## 13. Thermal characteristics

**Table 152. Thermal characteristics**

Symbol	Parameter	Conditions	Package	Typ	Unit
R <sub>th(j-a)</sub>	thermal resistance from junction to ambient	in still air with exposed pin soldered on a 4 layer JEDEC PCB	HVQFN32	40	K/W

## 14. Characteristics

Table 153. Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
<b>Input characteristics</b>						
<b>Pins EA, I2C and NRSTPD</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DD(PVDD)}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DD(PVDD)}$	V
<b>Pin MFIN</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DD(SVDD)}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DD(SVDD)}$	V
<b>Pin SDA</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DD(PVDD)}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DD(PVDD)}$	V
<b>Pin RX<sup>[1]</sup></b>						
$V_i$	input voltage		-1	-	$V_{DDA} + 1$	V
$C_i$	input capacitance	$V_{DDA} = 3\text{ V}$ ; receiver active; $V_{RX(p-p)} = 1\text{ V}$ ; 1.5 V (DC) offset	-	10	-	pF
$R_i$	input resistance	$V_{DDA} = 3\text{ V}$ ; receiver active; $V_{RX(p-p)} = 1\text{ V}$ ; 1.5 V (DC) offset	-	350	-	$\Omega$
<i>Input voltage range; see <a href="#">Figure 24</a></i>						
$V_{i(p-p)(min)}$	minimum peak-to-peak input voltage	Manchester encoded; $V_{DDA} = 3\text{ V}$	-	100	-	mV
$V_{i(p-p)(max)}$	maximum peak-to-peak input voltage	Manchester encoded; $V_{DDA} = 3\text{ V}$	-	4	-	V
<i>Input sensitivity; see <a href="#">Figure 24</a></i>						
$V_{mod}$	modulation voltage	minimum Manchester encoded; $V_{DDA} = 3\text{ V}$ ; $RxGain[2:0] = 111\text{b}$ (48 dB)	-	5	-	mV
<b>Pin OSCIN</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DDA}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DDA}$	V
$C_i$	input capacitance	$V_{DDA} = 2.8\text{ V}$ ; DC = 0.65 V; AC = 1 V (p-p)	-	2	-	pF
<b>Input/output characteristics</b>						
<b>pins D1, D2, D3, D4, D5, D6 and D7</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DD(PVDD)}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DD(PVDD)}$	V

**Table 153. Characteristics ...continued**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DD(PVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>DD(PVDD)</sub> - 0.4	-	V <sub>DD(PVDD)</sub>	V
V <sub>OL</sub>	LOW-level output voltage	V <sub>DD(PVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>SS(PVSS)</sub>	-	V <sub>SS(PVSS)</sub> + 0.4	V
I <sub>OH</sub>	HIGH-level output current	V <sub>DD(PVDD)</sub> = 3 V	-	-	4	mA
I <sub>OL</sub>	LOW-level output current	V <sub>DD(PVDD)</sub> = 3 V	-	-	4	mA
<b>Output characteristics</b>						
<b>Pin MFOUT</b>						
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DD(SVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>DD(SVDD)</sub> - 0.4	-	V <sub>DD(SVDD)</sub>	V
V <sub>OL</sub>	LOW-level output voltage	V <sub>DD(SVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>SS(PVSS)</sub>	-	V <sub>SS(PVSS)</sub> + 0.4	V
I <sub>OL</sub>	LOW-level output current	V <sub>DD(SVDD)</sub> = 3 V	-	-	4	mA
I <sub>OH</sub>	HIGH-level output current	V <sub>DD(SVDD)</sub> = 3 V	-	-	4	mA
<b>Pin IRQ</b>						
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DD(PVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>DD(PVDD)</sub> - 0.4	-	V <sub>DD(PVDD)</sub>	V
V <sub>OL</sub>	LOW-level output voltage	V <sub>DD(PVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>SS(PVSS)</sub>	-	V <sub>SS(PVSS)</sub> + 0.4	V
I <sub>OL</sub>	LOW-level output current	V <sub>DD(PVDD)</sub> = 3 V	-	-	4	mA
I <sub>OH</sub>	HIGH-level output current	V <sub>DD(PVDD)</sub> = 3 V	-	-	4	mA
<b>Pins AUX1 and AUX2</b>						
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DDD</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>DDD</sub> - 0.4	-	V <sub>DDD</sub>	V
V <sub>OL</sub>	LOW-level output voltage	V <sub>DDD</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>SS(PVSS)</sub>	-	V <sub>SS(PVSS)</sub> + 0.4	V
I <sub>OL</sub>	LOW-level output current	V <sub>DDD</sub> = 3 V	-	-	4	mA
I <sub>OH</sub>	HIGH-level output current	V <sub>DDD</sub> = 3 V	-	-	4	mA
<b>Pins TX1 and TX2</b>						
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DD(TVDD)</sub> = 3 V; I <sub>DD(TVDD)</sub> = 32 mA; CWGsP[5:0] = 3Fh	V <sub>DD(TVDD)</sub> - 0.15	-	-	V
		V <sub>DD(TVDD)</sub> = 3 V; I <sub>DD(TVDD)</sub> = 80 mA; CWGsP[5:0] = 3Fh	V <sub>DD(TVDD)</sub> - 0.4	-	-	V
		V <sub>DD(TVDD)</sub> = 2.5 V; I <sub>DD(TVDD)</sub> = 32 mA; CWGsP[5:0] = 3Fh	V <sub>DD(TVDD)</sub> - 0.24	-	-	V
		V <sub>DD(TVDD)</sub> = 2.5 V; I <sub>DD(TVDD)</sub> = 80 mA; CWGsP[5:0] = 3Fh	V <sub>DD(TVDD)</sub> - 0.64	-	-	V

Table 153. Characteristics ...continued

Symbol	Parameter	Conditions	Min	Typ	Max	Unit	
V <sub>OL</sub>	LOW-level output voltage	V <sub>DD(TVDD)</sub> = 3 V; I <sub>DD(TVDD)</sub> = 32 mA; CWGsP[5:0] = 0Fh	-	-	0.15	V	
		V <sub>DD(TVDD)</sub> = 3 V; I <sub>DD(TVDD)</sub> = 80 mA; CWGsP[5:0] = 0Fh	-	-	0.4	V	
		V <sub>DD(TVDD)</sub> = 2.5 V; I <sub>DD(TVDD)</sub> = 32 mA; CWGsP[5:0] = 0Fh	-	-	0.24	V	
		V <sub>DD(TVDD)</sub> = 2.5 V; I <sub>DD(TVDD)</sub> = 80 mA; CWGsP[5:0] = 0Fh	-	-	0.64	V	
<b>Current consumption</b>							
I <sub>pd</sub>	power-down current	V <sub>D<sub>DA</sub></sub> = V <sub>D<sub>DD</sub></sub> = V <sub>DD(TVDD)</sub> = V <sub>DD(PVDD)</sub> = 3 V					
		hard power-down; pin NRSTPD set LOW	[2]	-	-	5	μA
		soft power-down; RF level detector on	[2]	-	-	10	μA
I <sub>DD</sub>	digital supply current	pin DVDD; V <sub>D<sub>DD</sub></sub> = 3 V	-	6.5	9	mA	
I <sub>D<sub>DA</sub></sub>	analog supply current	pin AVDD; V <sub>D<sub>DA</sub></sub> = 3 V; CommandReg register's bit RcvOff = 0	-	7	10	mA	
		pin AVDD; receiver switched off; V <sub>D<sub>DA</sub></sub> = 3 V; CommandReg register's bit RcvOff = 1	-	3	5	mA	
I <sub>DD(PVDD)</sub>	PVDD supply current	pin PVDD	[3]	-	40	mA	
I <sub>DD(TVDD)</sub>	TVDD supply current	pin TVDD; continuous wave	[4][5][6]	-	60	100	mA
I <sub>DD(SVDD)</sub>	SVDD supply current	pin SVDD	[7]	-	4	mA	
<b>Clock frequency</b>							
f <sub>clk</sub>	clock frequency		-	27.12	-	MHz	
δ <sub>clk</sub>	clock duty cycle		40	50	60	%	
t <sub>jit</sub>	jitter time	RMS	-	-	10	ps	
<b>Crystal oscillator</b>							
V <sub>OH</sub>	HIGH-level output voltage	pin OSCOUT	-	1.1	-	V	
V <sub>OL</sub>	LOW-level output voltage	pin OSCOUT	-	0.2	-	V	
C <sub>i</sub>	input capacitance	pin OSCOUT	-	2	-	pF	
		pin OSCIN	-	2	-	pF	
<b>Typical input requirements</b>							
f <sub>xtal</sub>	crystal frequency		-	27.12	-	MHz	
ESR	equivalent series resistance		-	-	100	Ω	
C <sub>L</sub>	load capacitance		-	10	-	pF	
P <sub>xtal</sub>	crystal power dissipation		-	50	100	mW	

[1] The voltage on pin RX is clamped by internal diodes to pins AVSS and AVDD.

- [2]  $I_{pd}$  is the total current for all supplies.
- [3]  $I_{DD(PVDD)}$  depends on the overall load at the digital pins.
- [4]  $I_{DD(TVDD)}$  depends on  $V_{DD(TVDD)}$  and the external circuit connected to pins TX1 and TX2.
- [5] During typical circuit operation, the overall current is below 100 mA.
- [6] Typical value using a complementary driver configuration and an antenna matched to  $40 \Omega$  between pins TX1 and TX2 at 13.56 MHz.
- [7]  $I_{DD(SVDD)}$  depends on the load at pin MFOUT.

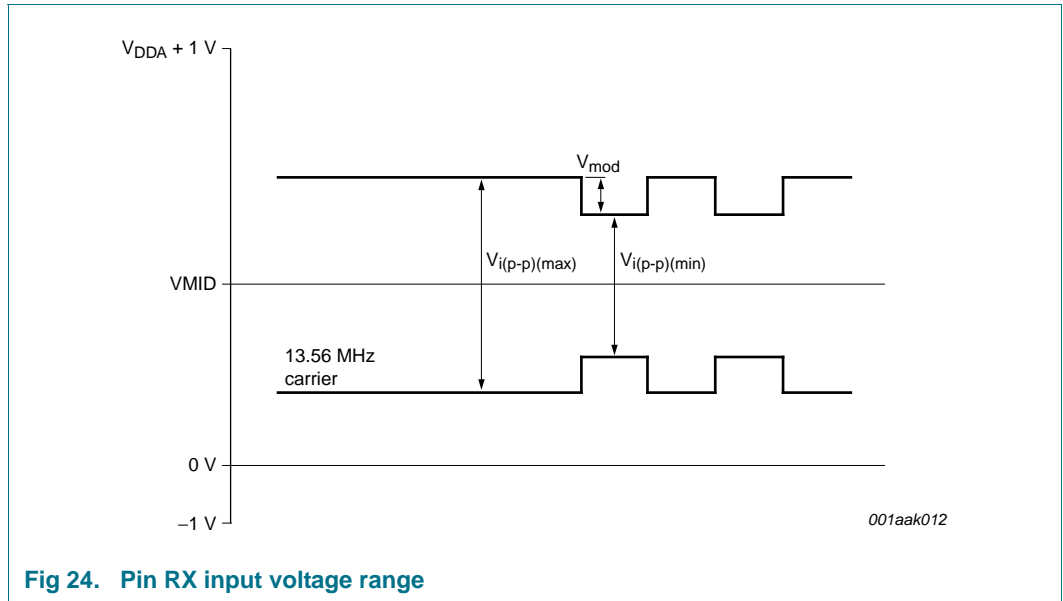


Fig 24. Pin RX input voltage range

## 14.1 Timing characteristics

Table 154. SPI timing characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{WL}$	pulse width LOW	line SCK	50	-	-	ns
$t_{WH}$	pulse width HIGH	line SCK	50	-	-	ns
$t_{h(SCKH-D)}$	SCK HIGH to data input hold time	SCK to changing MOSI	25	-	-	ns
$t_{su(D-SCKH)}$	data input to SCK HIGH set-up time	changing MOSI to SCK	25	-	-	ns
$t_{h(SCKL-Q)}$	SCK LOW to data output hold time	SCK to changing MISO	-	-	25	ns
$t_{(SCKL-NSSH)}$	SCK LOW to NSS HIGH time		0	-	-	ns
$t_{NHNL}$	NSS high before communication		50	-	-	ns



Table 155. I<sup>2</sup>C-bus timing in Fast mode

Symbol	Parameter	Conditions	Fast mode		High-speed mode		Unit
			Min	Max	Min	Max	
f <sub>SCL</sub>	SCL clock frequency		0	400	0	3400	kHz
t <sub>HD;STA</sub>	hold time (repeated) START condition	after this period, the first clock pulse is generated	600	-	160	-	ns
t <sub>SU;STA</sub>	set-up time for a repeated START condition		600	-	160	-	ns
t <sub>SU;STO</sub>	set-up time for STOP condition		600	-	160	-	ns
t <sub>LOW</sub>	LOW period of the SCL clock		1300	-	160	-	ns
t <sub>HIGH</sub>	HIGH period of the SCL clock		600	-	60	-	ns
t <sub>HD;DAT</sub>	data hold time		0	900	0	70	ns
t <sub>SU;DAT</sub>	data set-up time		100	-	10	-	ns
t <sub>r</sub>	rise time	SCL signal	20	300	10	40	ns
t <sub>f</sub>	fall time	SCL signal	20	300	10	40	ns
t <sub>r</sub>	rise time	SDA and SCL signals	20	300	10	80	ns
t <sub>f</sub>	fall time	SDA and SCL signals	20	300	10	80	ns
t <sub>BUF</sub>	bus free time between a STOP and START condition		1.3	-	1.3	-	μs

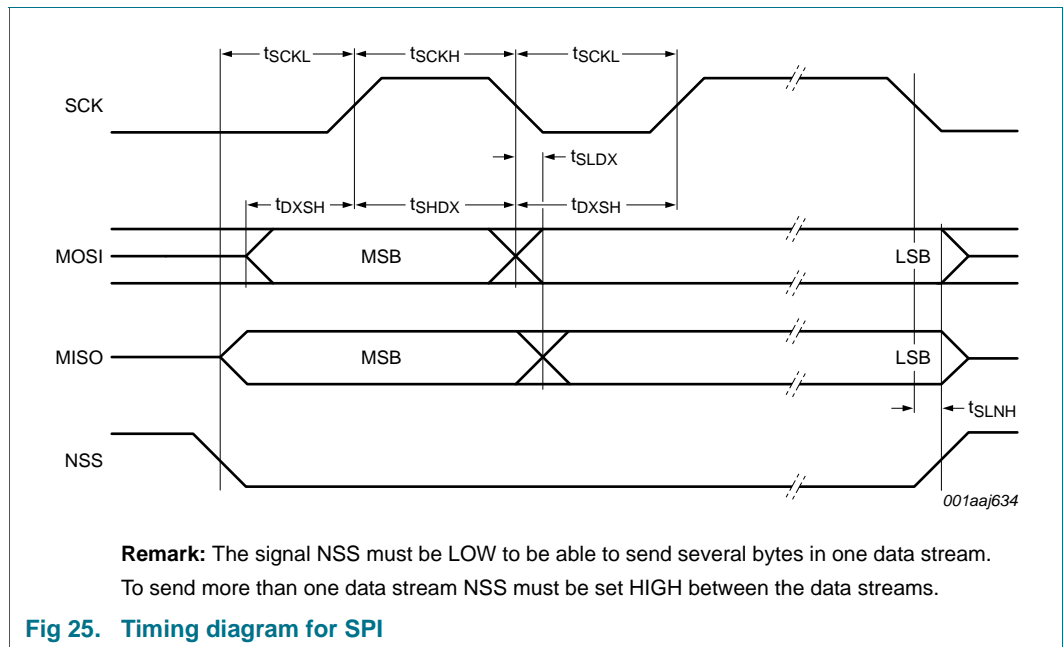


Fig 25. Timing diagram for SPI

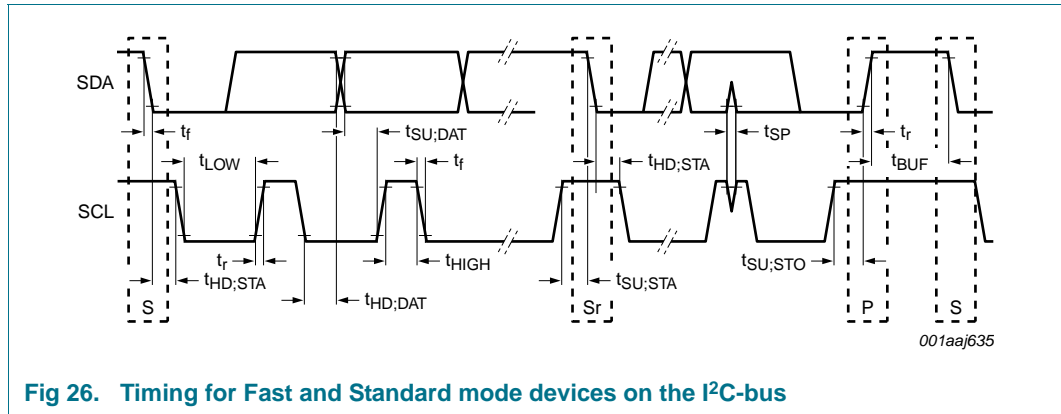


Fig 26. Timing for Fast and Standard mode devices on the I<sup>2</sup>C-bus

### 15. Application information

A typical application diagram using a complementary antenna connection to the MFRC522 is shown in [Figure 27](#).

The antenna tuning and RF part matching is described in the application note [Ref. 1](#) and [Ref. 2](#).

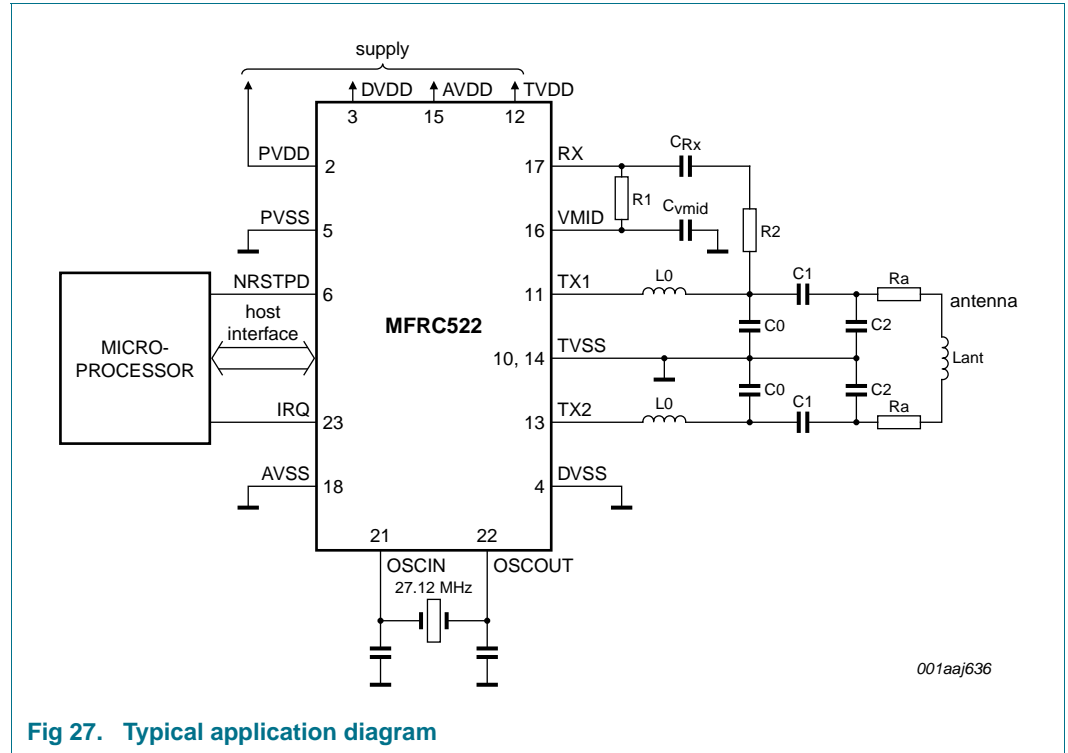


Fig 27. Typical application diagram

## 16. Test information

### 16.1 Test signals

#### 16.1.1 Self test

The MFRC522 has the capability to perform a digital self test. The self test is started by using the following procedure:

1. Perform a soft reset.
2. Clear the internal buffer by writing 25 bytes of 00h and implement the Config command.
3. Enable the self test by writing 09h to the AutoTestReg register.
4. Write 00h to the FIFO buffer.
5. Start the self test with the CalcCRC command.
6. The self test is initiated.
7. When the self test has completed, the FIFO buffer contains the following 64 bytes:

FIFO buffer byte values for MFRC522 version 1.0:

00h, C6h, 37h, D5h, 32h, B7h, 57h, 5Ch,  
C2h, D8h, 7Ch, 4Dh, D9h, 70h, C7h, 73h,  
10h, E6h, D2h, AAh, 5Eh, A1h, 3Eh, 5Ah,  
14h, AFh, 30h, 61h, C9h, 70h, DBh, 2Eh,  
64h, 22h, 72h, B5h, BDh, 65h, F4h, ECh,  
22h, BCh, D3h, 72h, 35h, CDh, AAh, 41h,  
1Fh, A7h, F3h, 53h, 14h, DEh, 7Eh, 02h,  
D9h, 0Fh, B5h, 5Eh, 25h, 1Dh, 29h, 79h

FIFO buffer byte values for MFRC522 version 2.0:

00h, EBh, 66h, BAh, 57h, BFh, 23h, 95h,  
D0h, E3h, 0Dh, 3Dh, 27h, 89h, 5Ch, DEh,  
9Dh, 3Bh, A7h, 00h, 21h, 5Bh, 89h, 82h,  
51h, 3Ah, EBh, 02h, 0Ch, A5h, 00h, 49h,  
7Ch, 84h, 4Dh, B3h, CCh, D2h, 1Bh, 81h,  
5Dh, 48h, 76h, D5h, 71h, 061h, 21h, A9h,  
86h, 96h, 83h, 38h, CFh, 9Dh, 5Bh, 6Dh,  
DCh, 15h, BAh, 3Eh, 7Dh, 95h, 03Bh, 2Fh

#### 16.1.2 Test bus

The test bus is used for production tests. The following configuration can be used to improve the design of a system using the MFRC522. The test bus allows internal signals to be routed to the digital interface. The test bus comprises two sets of test signals which are selected using their subaddress specified in the TestSel2Reg register's TestBusSel[4:0] bits. The test signals and their related digital output pins are described in [Table 156](#) and [Table 157](#).

**Table 156. Test bus signals: TestBusSel[4:0] = 07h**

Pins	Internal signal name	Description
D6	s_data	received data stream
D5	s_coll	bit-collision detected (106 kBd only)
D4	s_valid	s_data and s_coll signals are valid
D3	s_over	receiver has detected a stop condition
D2	RCV_reset	receiver is reset
D1	-	reserved

**Table 157. Test bus signals: TestBusSel[4:0] = 0Dh**

Pins	Internal test signal name	Description
D6	clkstable	oscillator output signal
D5	clk27/8	oscillator output signal divided by 8
D4 to D3	-	reserved
D2	clk27	oscillator output signal
D1	-	reserved

### 16.1.3 Test signals on pins AUX1 or AUX2

The MFRC522 allows the user to select internal signals for measurement on pins AUX1 or AUX2. These measurements can be helpful during the design-in phase to optimize the design or used for test purposes.

[Table 158](#) shows the signals that can be switched to pin AUX1 or AUX2 by setting AnalogSelAux1[3:0] or AnalogSelAux2[3:0] in the AnalogTestReg register.

**Remark:** The DAC has a current output, therefore it is recommended that a 1 k $\Omega$  pull-down resistor is connected to pin AUX1 or AUX2.

**Table 158. Test signal descriptions**

AnalogSelAux1[3:0] or AnalogSelAux2[3:0] value	Signal on pin AUX1 or pin AUX2
0000	3-state
0001	DAC: register TestDAC1 or TestDAC2
0010	DAC: test signal Corr1
0011	reserved
0100	DAC: test signal MinLevel
0101	DAC: test signal ADC_I
0110	DAC: test signal ADC_Q
0111 to 1001	reserved
1010	HIGH
1011	LOW
1100	TxActive

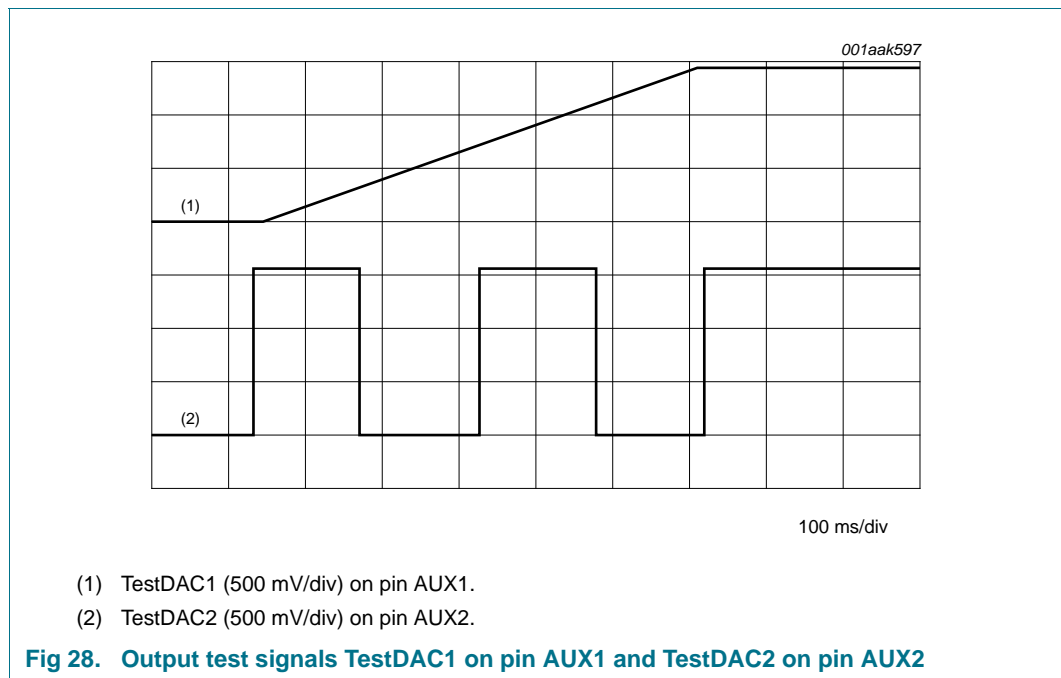
Table 158. Test signal descriptions ...continued

AnalogSelAux1[3:0] or AnalogSelAux2[3:0] value	Signal on pin AUX1 or pin AUX2
1101	RxActive
1110	subcarrier detected
1111	TstBusBit

16.1.3.1 Example: Output test signals TestDAC1 and TestDAC2

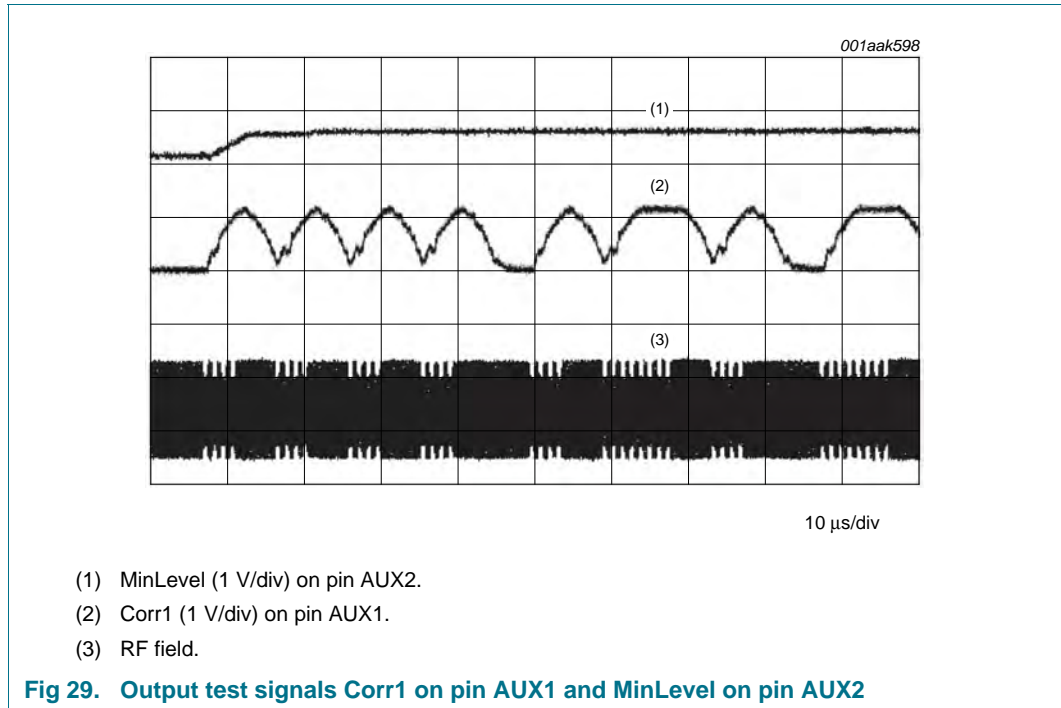
The AnalogTestReg register is set to 11h. The output on pin AUX1 has the test signal TestDAC1 and the output on pin AUX2 has the test signal TestDAC2. The signal values of TestDAC1 and TestDAC2 are controlled by the TestDAC1Reg and TestDAC2Reg registers.

Figure 28 shows test signal TestDAC1 on pin AUX1 and TestDAC2 on pin AUX2 when the TestDAC1Reg register is programmed with a slope defined by values 00h to 3Fh and the TestDAC2Reg register is programmed with a rectangular signal defined by values 00h and 3Fh.



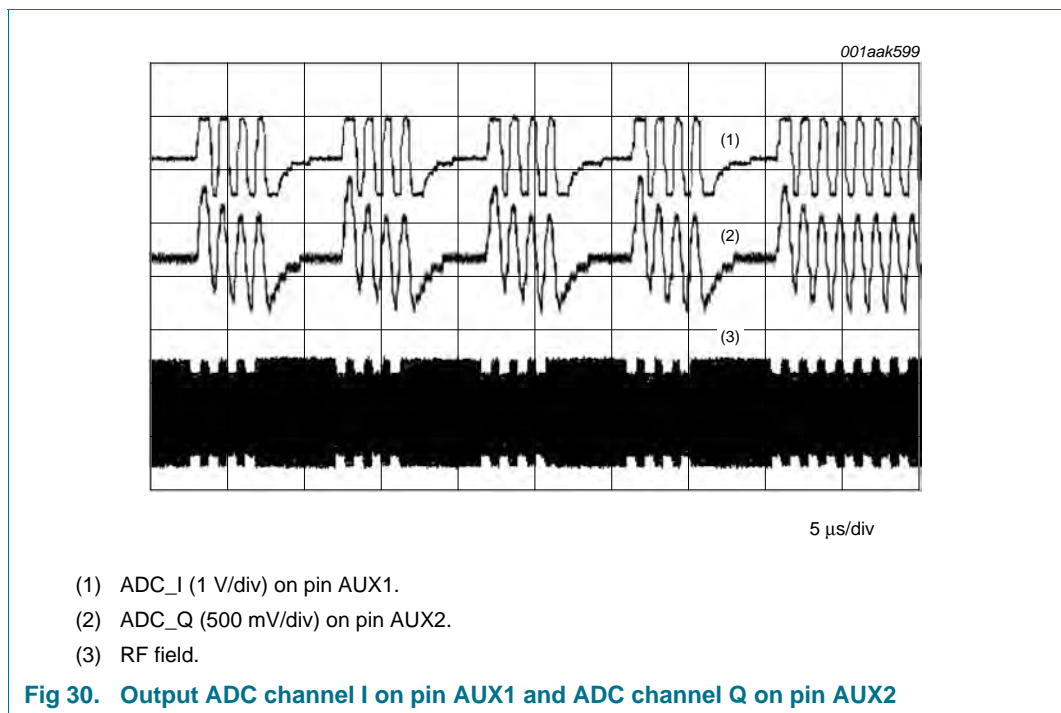
16.1.3.2 Example: Output test signals Corr1 and MinLevel

Figure 29 shows test signals Corr1 and MinLevel on pins AUX1 and AUX2, respectively. The AnalogTestReg register is set to 24h.



**16.1.3.3 Example: Output test signals ADC channel I and ADC channel Q**

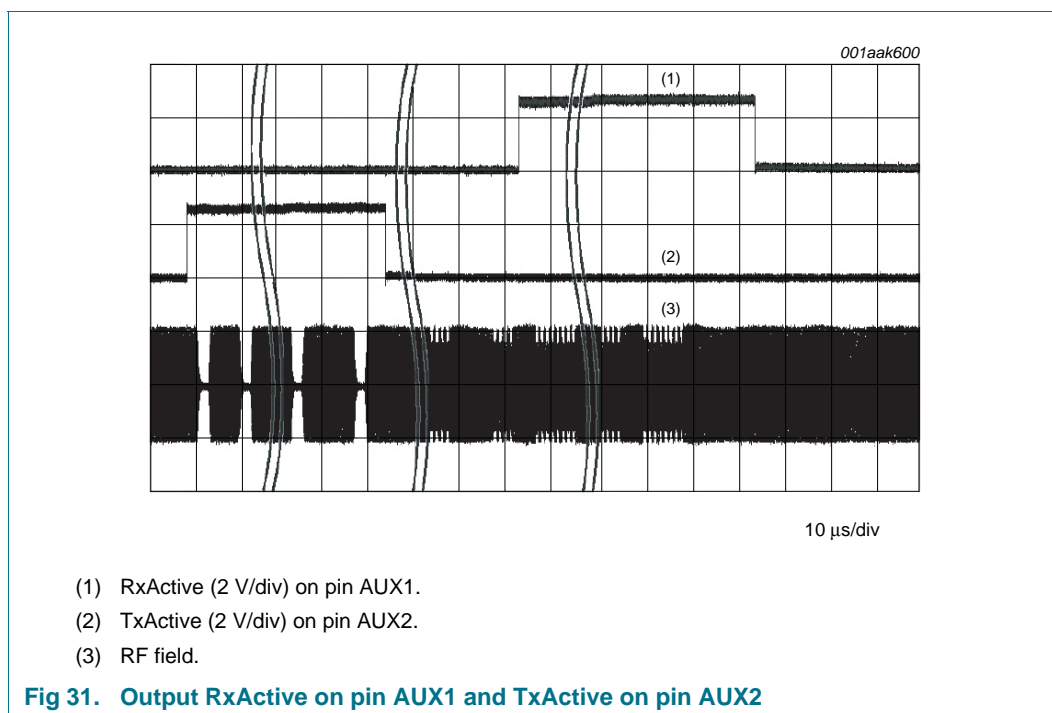
Figure 30 shows the channel behavior test signals ADC\_I and ADC\_Q on pins AUX1 and AUX2, respectively. The AnalogTestReg register is set to 56h.



**16.1.3.4 Example: Output test signals RxActive and TxActive**

Figure 31 shows the RxActive and TxActive test signals relating to RF communication. The AnalogTestReg register is set to CDh.

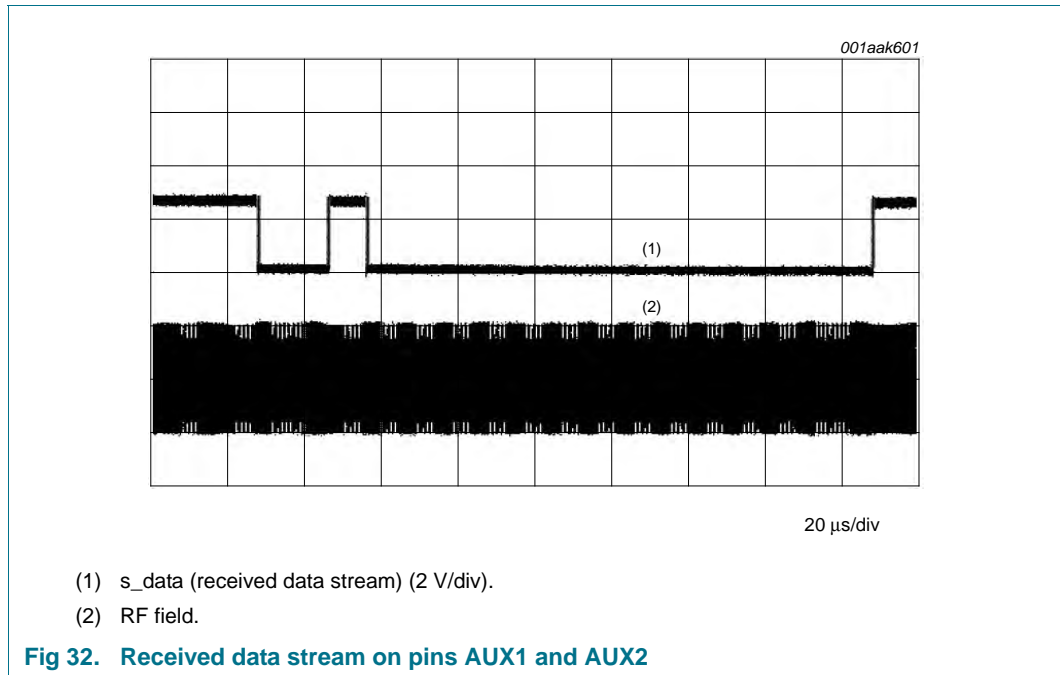
- At 106 kBd, RxActive is HIGH during data bits, parity and CRC reception. Start bits are not included
- At 106 kBd, TxActive is HIGH during start bits, data bits, parity and CRC transmission
- At 212 kBd, 424 kBd and 848 kBd, RxActive is HIGH during data bits and CRC reception. Start bits are not included
- At 212 kBd, 424 kBd and 848 kBd, TxActive is HIGH during data bits and CRC transmission





16.1.3.5 Example: Output test signal RX data stream

Figure 32 shows the data stream that is currently being received. The TestSel2Reg register's TestBusSel[4:0] bits are set to 07h to enable test bus signals on pins D1 to D6; see Section 16.1.2 on page 81. The TestSel1Reg register's TstBusBitSel[2:0] bits are set to 06h (pin D6 = s\_data) and AnalogTestReg register is set to FFh (TstBusBit) which outputs the received data stream on pins AUX1 and AUX2.



16.1.3.6 PRBS

The pseudo-random binary sequences PRBS9 and PRBS15 are based on ITU-T0150 and are defined with the TestSel2Reg register. Transmission of either data stream is started by the Transmit command. The preamble/sync byte/start bit/parity bit are automatically generated depending on the mode selected.

**Remark:** All relevant registers for transmitting data must be configured in accordance with ITU-T0150 before selecting PRBS transmission.

17. Package outline

HVQFN32: plastic thermal enhanced very thin quad flat package; no leads; 32 terminals; body 5 x 5 x 0.85 mm

SOT617-1

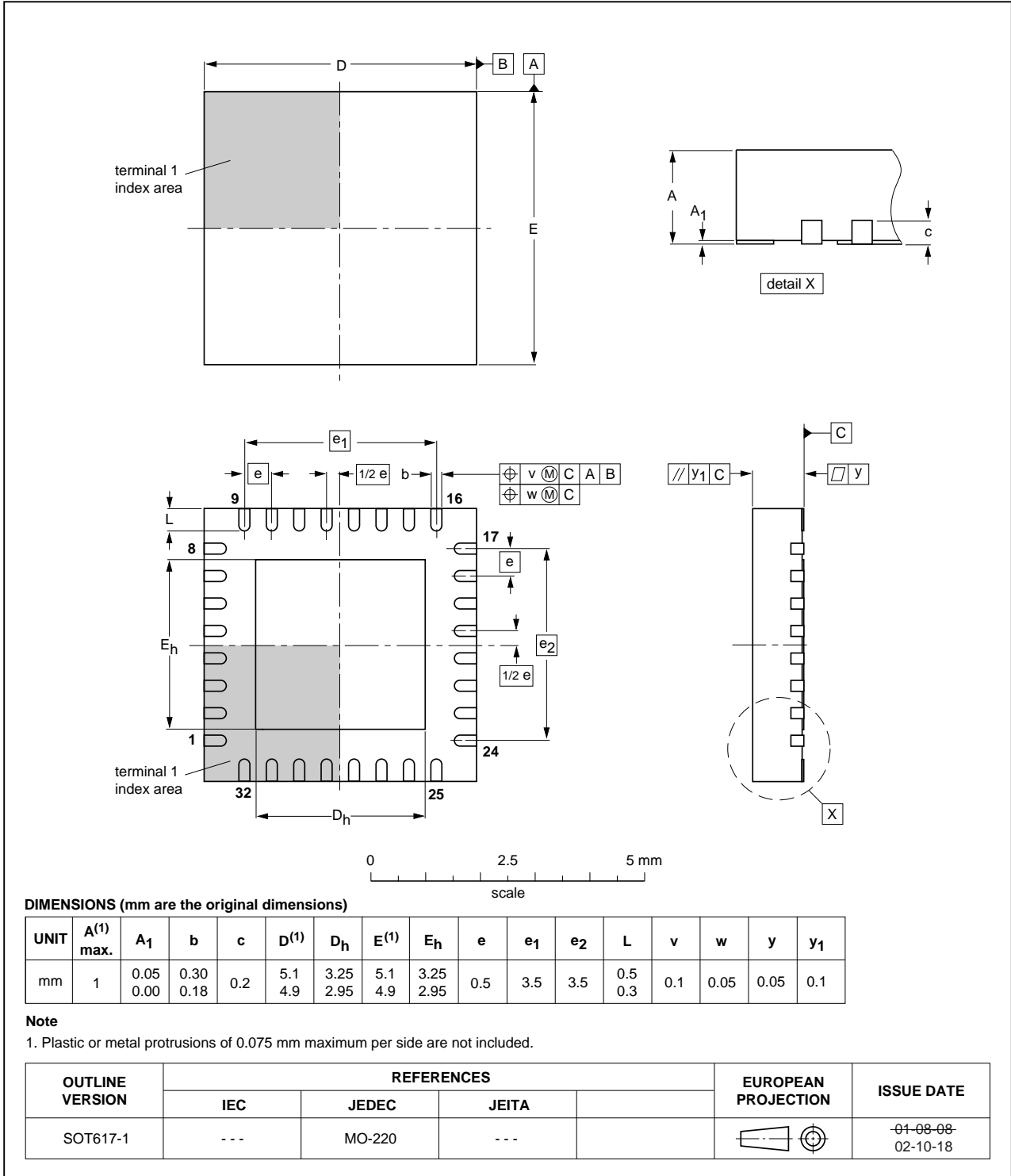


Fig 33. Package outline SOT617-1 (HVQFN32)

Detailed package information can be found at:  
<http://www.nxp.com/package/SOT617-1.html>.

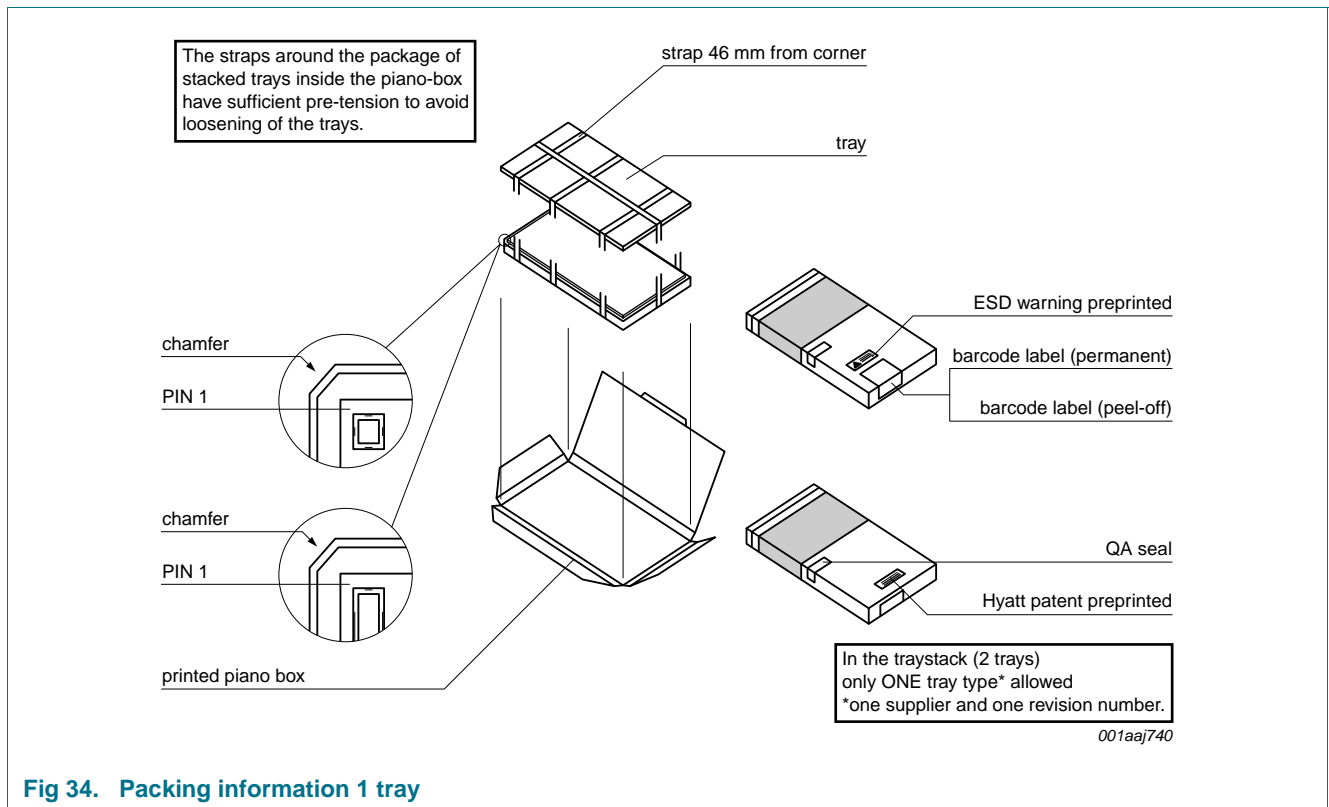
## 18. Handling information

Moisture Sensitivity Level (MSL) evaluation has been performed according to *SNW-FQ-225B rev.04/07/07 (JEDEC J-STD-020C)*. MSL for this package is level 1 which means 260 °C convection reflow temperature.

Dry pack is not required.

Unlimited out-of-pack floor life at maximum ambient 30 °C/85 % RH.

## 19. Packing information



## 20. Abbreviations

**Table 159. Abbreviations**

Acronym	Description
ADC	Analog-to-Digital Converter
BPSK	Binary Phase Shift Keying
CRC	Cyclic Redundancy Check
CW	Continuous Wave
DAC	Digital-to-Analog Converter
HBM	Human Body Model
I <sup>2</sup> C	Inter-integrated Circuit
LSB	Least Significant Bit
MISO	Master In Slave Out
MM	Machine Model
MOSI	Master Out Slave In
MSB	Most Significant Bit
NRZ	Not Return to Zero
NSS	Not Slave Select
PLL	Phase-Locked Loop
PRBS	Pseudo-Random Bit Sequence
RX	Receiver
SOF	Start Of Frame
SPI	Serial Peripheral Interface
TX	Transmitter
UART	Universal Asynchronous Receiver Transmitter

## 21. References

- [1] **Application note** — *MFRC52x Reader IC Family Directly Matched Antenna Design*
- [2] **Application note** — *MIFARE (ISO/IEC 14443 A) 13.56 MHz RFID Proximity Antennas*

## 22. Revision history

Table 160. Revision history

Document ID	Release date	Data sheet status	Change notice	Supersedes
MFRC522 v.3.6	20111214	Product data sheet	-	MFRC522_35
Modifications:				
				<ul style="list-style-type: none"> <li>• <a href="#">Section 2.1 “Differences between version 1.0 and 2.0” on page 1</a>: added</li> <li>• <a href="#">Table 2 “Ordering information” on page 3</a>: updated</li> <li>• <a href="#">Section 9.3.2.10 “DemodReg register” on page 52</a>: register updated and add reference to Timer unit</li> <li>• <a href="#">Section 8.5 “Timer unit” on page 30</a>: Pre Scaler Information for version 2.0 added</li> <li>• <a href="#">Section 9.3.4.8 “VersionReg register” on page 65</a>: version information structured in chip information and version information updated, including version 1.0 and 2.0</li> <li>• <a href="#">Section 16.1 “Test signals” on page 81</a>: selftest result including values for version 1.0 and 2.0</li> </ul>
MFRC522_35	20100621	Product data sheet		MFRC522_34
Modifications:				
				<ul style="list-style-type: none"> <li>• <a href="#">Section 9.3.2.10 “DemodReg register” on page 52</a>: register updated</li> <li>• <a href="#">Section 9.3.3.10 “TModeReg and TPrescalerReg registers” on page 59</a>: register updated</li> <li>• <a href="#">Section 8.5 “Timer unit” on page 30</a>: timer calculation updated</li> <li>• <a href="#">Section 9.3.4.8 “VersionReg register” on page 65</a>: version B2h updated</li> <li>• <a href="#">Section 16.1 “Test signals” on page 81</a>: selftest result updated</li> </ul>
MFRC522_34	20100305	Product data sheet		MFRC522_33
Modifications:				
				<ul style="list-style-type: none"> <li>• <a href="#">Section 8.5 “Timer unit”</a>: information added</li> <li>• <a href="#">Table 106 “TModeReg register bit descriptions”</a>: bit 7 updated</li> <li>• <a href="#">Table 154 “SPI timing characteristics”</a>: row added</li> </ul>
MFRC522_33	20091026	Product data sheet	-	112132

## 23. Legal information

### 23.1 Data sheet status

Document status <sup>[1][2]</sup>	Product status <sup>[3]</sup>	Definition
Objective [short] data sheet	Development	This document contains data from the objective specification for product development.
Preliminary [short] data sheet	Qualification	This document contains data from the preliminary specification.
Product [short] data sheet	Production	This document contains the product specification.

[1] Please consult the most recently issued document before initiating or completing a design.

[2] The term 'short data sheet' is explained in section "Definitions".

[3] The product status of device(s) described in this document may have changed since this document was published and may differ in case of multiple devices. The latest product status information is available on the Internet at URL <http://www.nxp.com>.

### 23.2 Definitions

**Draft** — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

**Short data sheet** — A short data sheet is an extract from a full data sheet with the same product type number(s) and title. A short data sheet is intended for quick reference only and should not be relied upon to contain detailed and full information. For detailed and full information see the relevant full data sheet, which is available on request via the local NXP Semiconductors sales office. In case of any inconsistency or conflict with the short data sheet, the full data sheet shall prevail.

**Product specification** — The information and data provided in a Product data sheet shall define the specification of the product as agreed between NXP Semiconductors and its customer, unless NXP Semiconductors and customer have explicitly agreed otherwise in writing. In no event however, shall an agreement be valid in which the NXP Semiconductors product is deemed to offer functions and qualities beyond those described in the Product data sheet.

### 23.3 Disclaimers

**Limited warranty and liability** — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the *Terms and conditions of commercial sale* of NXP Semiconductors.

**Right to make changes** — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

**Suitability for use** — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or

malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

**Applications** — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

**Limiting values** — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) will cause permanent damage to the device. Limiting values are stress ratings only and (proper) operation of the device at these or any other conditions above those given in the Recommended operating conditions section (if present) or the Characteristics sections of this document is not warranted. Constant or repeated exposure to limiting values will permanently and irreversibly affect the quality and reliability of the device.

**Terms and conditions of commercial sale** — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at <http://www.nxp.com/profile/terms>, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

**No offer to sell or license** — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

**Export control** — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

**Quick reference data** — The Quick reference data is an extract of the product data given in the Limiting values and Characteristics sections of this document, and as such is not complete, exhaustive or legally binding.

**Non-automotive qualified products** — Unless this data sheet expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

## 24. Contact information

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: [salesaddresses@nxp.com](mailto:salesaddresses@nxp.com)

## 23.4 Licenses

### Purchase of NXP ICs with ISO/IEC 14443 type B functionality



**RATP/Innovatron  
Technology**

This NXP Semiconductors IC is ISO/IEC 14443 Type B software enabled and is licensed under Innovatron's Contactless Card patents license for ISO/IEC 14443 B. The license includes the right to use the IC in systems and/or end-user equipment.

## 23.5 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

**I<sup>2</sup>C-bus** — logo is a trademark of NXP B.V.

**MIFARE** — is a trademark of NXP B.V.

## 25. Contents

<b>1</b>	<b>Introduction</b> .....	<b>1</b>	8.6	Power reduction modes .....	32
<b>2</b>	<b>General description</b> .....	<b>1</b>	8.6.1	Hard power-down .....	32
2.1	Differences between version 1.0 and 2.0 .....	1	8.6.2	Soft power-down mode .....	32
<b>3</b>	<b>Features and benefits</b> .....	<b>2</b>	8.6.3	Transmitter power-down mode .....	32
<b>4</b>	<b>Quick reference data</b> .....	<b>2</b>	8.7	Oscillator circuit .....	32
<b>5</b>	<b>Ordering information</b> .....	<b>3</b>	8.8	Reset and oscillator start-up time .....	33
<b>6</b>	<b>Block diagram</b> .....	<b>4</b>	8.8.1	Reset timing requirements .....	33
<b>7</b>	<b>Pinning information</b> .....	<b>6</b>	8.8.2	Oscillator start-up time .....	33
7.1	Pin description .....	6	<b>9</b>	<b>MFRC522 registers</b> .....	<b>34</b>
<b>8</b>	<b>Functional description</b> .....	<b>8</b>	9.1	Register bit behavior .....	34
8.1	Digital interfaces .....	9	9.2	Register overview .....	35
8.1.1	Automatic microcontroller interface detection ..	9	9.3	Register descriptions .....	37
8.1.2	Serial Peripheral Interface .....	10	9.3.1	Page 0: Command and status .....	37
8.1.2.1	SPI read data .....	10	9.3.1.1	Reserved register 00h .....	37
8.1.2.2	SPI write data .....	11	9.3.1.2	CommandReg register .....	37
8.1.2.3	SPI address byte .....	11	9.3.1.3	ComEnReg register .....	37
8.1.3	UART interface .....	11	9.3.1.4	DivlEnReg register .....	38
8.1.3.1	Connection to a host .....	11	9.3.1.5	ComlRqReg register .....	38
8.1.3.2	Selectable UART transfer speeds .....	12	9.3.1.6	DivlRqReg register .....	39
8.1.3.3	UART framing .....	13	9.3.1.7	ErrorReg register .....	40
8.1.4	I <sup>2</sup> C-bus interface .....	16	9.3.1.8	Status1Reg register .....	41
8.1.4.1	Data validity .....	17	9.3.1.9	Status2Reg register .....	42
8.1.4.2	START and STOP conditions .....	17	9.3.1.10	FIFODataReg register .....	43
8.1.4.3	Byte format .....	17	9.3.1.11	FIFOLevelReg register .....	43
8.1.4.4	Acknowledge .....	18	9.3.1.12	WaterLevelReg register .....	43
8.1.4.5	7-Bit addressing .....	19	9.3.1.13	ControlReg register .....	44
8.1.4.6	Register write access .....	19	9.3.1.14	BitFramingReg register .....	45
8.1.4.7	Register read access .....	20	9.3.1.15	CollReg register .....	45
8.1.4.8	High-speed mode .....	21	9.3.1.16	Reserved register 0Fh .....	46
8.1.4.9	High-speed transfer .....	21	9.3.2	Page 1: Communication .....	46
8.1.4.10	Serial data transfer format in HS mode .....	21	9.3.2.1	Reserved register 10h .....	46
8.1.4.11	Switching between F/S mode and HS mode ..	23	9.3.2.2	ModeReg register .....	47
8.1.4.12	MFRC522 at lower speed modes .....	23	9.3.2.3	TxModeReg register .....	47
8.2	Analog interface and contactless UART .....	24	9.3.2.4	RxModeReg register .....	48
8.2.1	General .....	24	9.3.2.5	TxControlReg register .....	49
8.2.2	TX p-driver .....	24	9.3.2.6	TxASKReg register .....	50
8.2.3	Serial data switch .....	26	9.3.2.7	TxSelReg register .....	50
8.2.4	MFIN and MFOUT interface support .....	26	9.3.2.8	RxSelReg register .....	51
8.2.5	CRC coprocessor .....	28	9.3.2.9	RxThresholdReg register .....	52
8.3	FIFO buffer .....	28	9.3.2.10	DemodReg register .....	52
8.3.1	Accessing the FIFO buffer .....	28	9.3.2.11	Reserved register 1Ah .....	53
8.3.2	Controlling the FIFO buffer .....	28	9.3.2.12	Reserved register 1Bh .....	53
8.3.3	FIFO buffer status information .....	28	9.3.2.13	MfTxReg register .....	53
8.4	Interrupt request system .....	29	9.3.2.14	MfRxReg register .....	54
8.4.1	Interrupt sources overview .....	29	9.3.2.15	Reserved register 1Eh .....	54
8.5	Timer unit .....	30	9.3.2.16	SerialSpeedReg register .....	54
			9.3.3	Page 2: Configuration .....	56
			9.3.3.1	Reserved register 20h .....	56

continued >>



9.3.3.2	CRCResultReg registers	56	16.1.2	Test bus	81
9.3.3.3	Reserved register 23h	57	16.1.3	Test signals on pins AUX1 or AUX2	82
9.3.3.4	ModWidthReg register	57	16.1.3.1	Example: Output test signals TestDAC1 and TestDAC2	83
9.3.3.5	Reserved register 25h	57	16.1.3.2	Example: Output test signals Corr1 and MinLevel	83
9.3.3.6	RFCfgReg register	58	16.1.3.3	Example: Output test signals ADC channel I and ADC channel Q	84
9.3.3.7	GsNReg register	58	16.1.3.4	Example: Output test signals RxActive and TxActive	85
9.3.3.8	CWGsPReg register	59	16.1.3.5	Example: Output test signal RX data stream	86
9.3.3.9	ModGsPReg register	59	16.1.3.6	PRBS	86
9.3.3.10	TModeReg and TPrescalerReg registers	59	<b>17</b>	<b>Package outline</b>	<b>87</b>
9.3.3.11	TReloadReg register	61	<b>18</b>	<b>Handling information</b>	<b>88</b>
9.3.3.12	TCounterValReg register	61	<b>19</b>	<b>Packing information</b>	<b>88</b>
9.3.4	Page 3: Test	62	<b>20</b>	<b>Abbreviations</b>	<b>89</b>
9.3.4.1	Reserved register 30h	62	<b>21</b>	<b>References</b>	<b>89</b>
9.3.4.2	TestSel1Reg register	62	<b>22</b>	<b>Revision history</b>	<b>90</b>
9.3.4.3	TestSel2Reg register	63	<b>23</b>	<b>Legal information</b>	<b>91</b>
9.3.4.4	TestPinEnReg register	63	23.1	Data sheet status	91
9.3.4.5	TestPinValueReg register	64	23.2	Definitions	91
9.3.4.6	TestBusReg register	64	23.3	Disclaimers	91
9.3.4.7	AutoTestReg register	65	23.4	Licenses	92
9.3.4.8	VersionReg register	65	23.5	Trademarks	92
9.3.4.9	AnalogTestReg register	66	<b>24</b>	<b>Contact information</b>	<b>92</b>
9.3.4.10	TestDAC1Reg register	67	<b>25</b>	<b>Contents</b>	<b>93</b>
9.3.4.11	TestDAC2Reg register	67			
9.3.4.12	TestADCReg register	67			
9.3.4.13	Reserved register 3Ch	67			
<b>10</b>	<b>MFRC522 command set</b>	<b>69</b>			
10.1	General description	69			
10.2	General behavior	69			
10.3	MFRC522 command overview	69			
10.3.1	MFRC522 command descriptions	70			
10.3.1.1	Idle	70			
10.3.1.2	Mem	70			
10.3.1.3	Generate RandomID	70			
10.3.1.4	CalcCRC	70			
10.3.1.5	Transmit	70			
10.3.1.6	NoCmdChange	70			
10.3.1.7	Receive	71			
10.3.1.8	Transceive	71			
10.3.1.9	MFAuthent	71			
10.3.1.10	SoftReset	72			
<b>11</b>	<b>Limiting values</b>	<b>73</b>			
<b>12</b>	<b>Recommended operating conditions</b>	<b>73</b>			
<b>13</b>	<b>Thermal characteristics</b>	<b>73</b>			
<b>14</b>	<b>Characteristics</b>	<b>74</b>			
14.1	Timing characteristics	77			
<b>15</b>	<b>Application information</b>	<b>80</b>			
<b>16</b>	<b>Test information</b>	<b>81</b>			
16.1	Test signals	81			
16.1.1	Self test	81			

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

© NXP B.V. 2011.

All rights reserved.

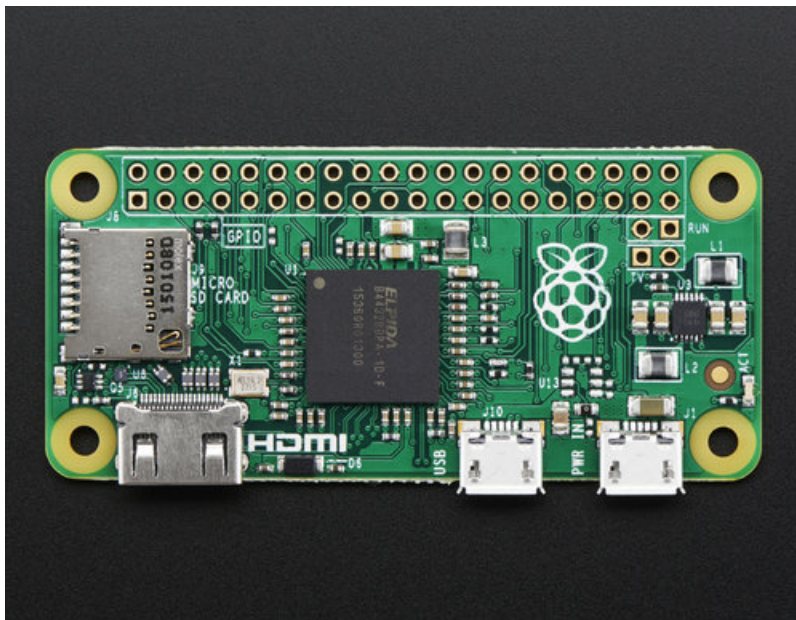
For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: [salesaddresses@nxp.com](mailto:salesaddresses@nxp.com)

Date of release: 14 December 2011  
112136

## Introducing the Raspberry Pi Zero

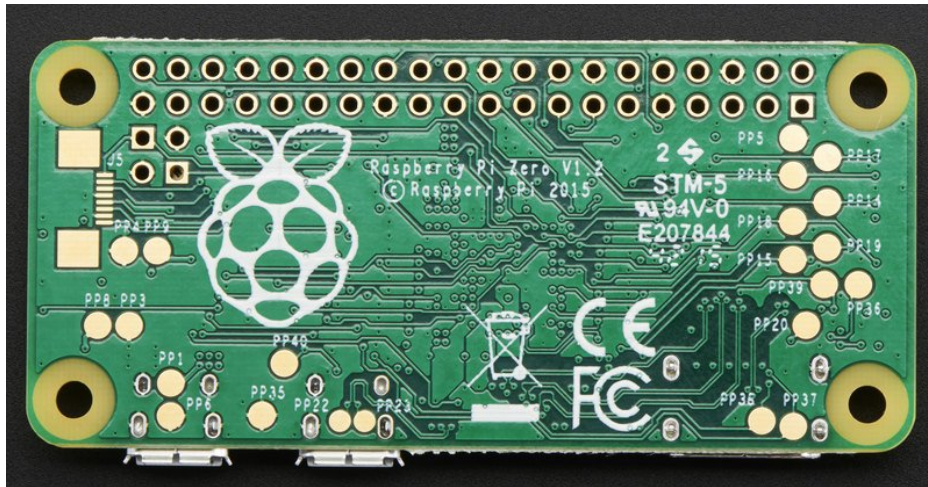
Created by lady\_ada



Last updated on 2019-10-31 09:34:18 PM UTC





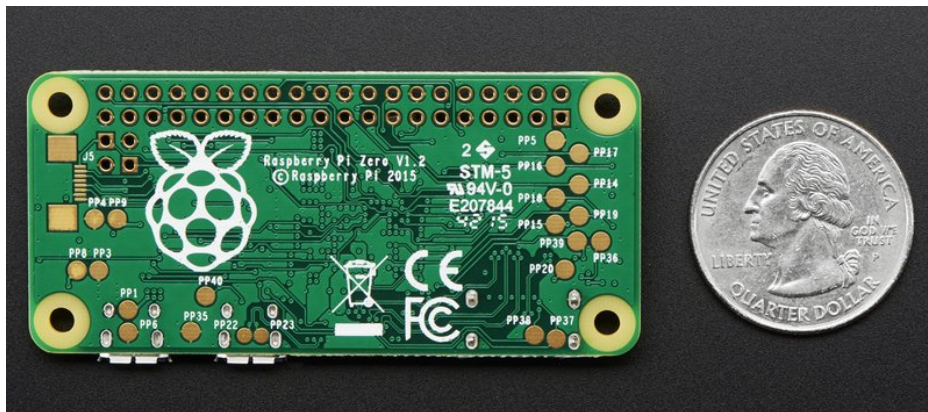


## Size

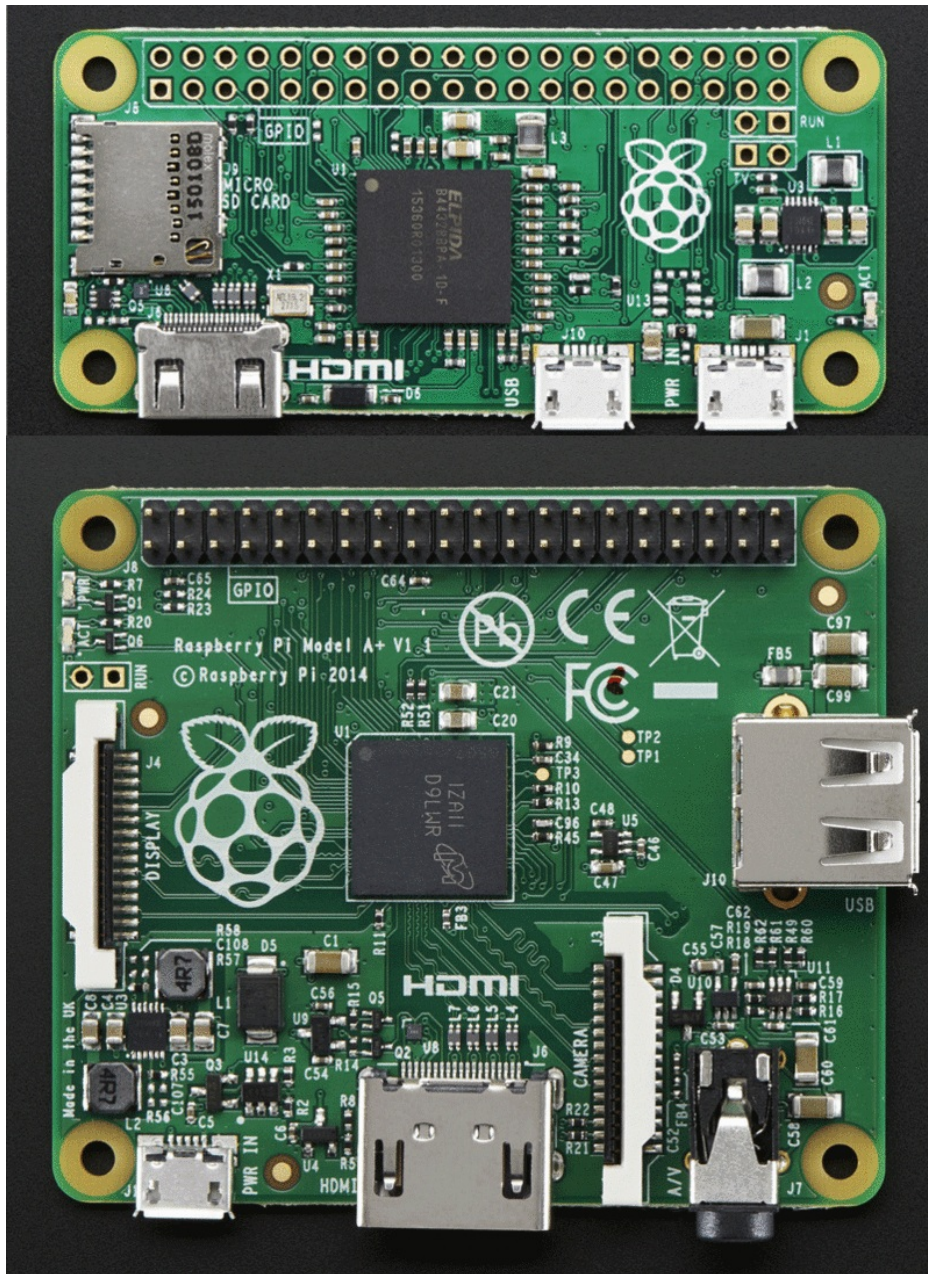
First up, the Pi Zero is **small and thin**

**65mm long x 30mm wide x 5mm thick**

(31mm if you include the little sticky-out bits of the microUSB jacks)

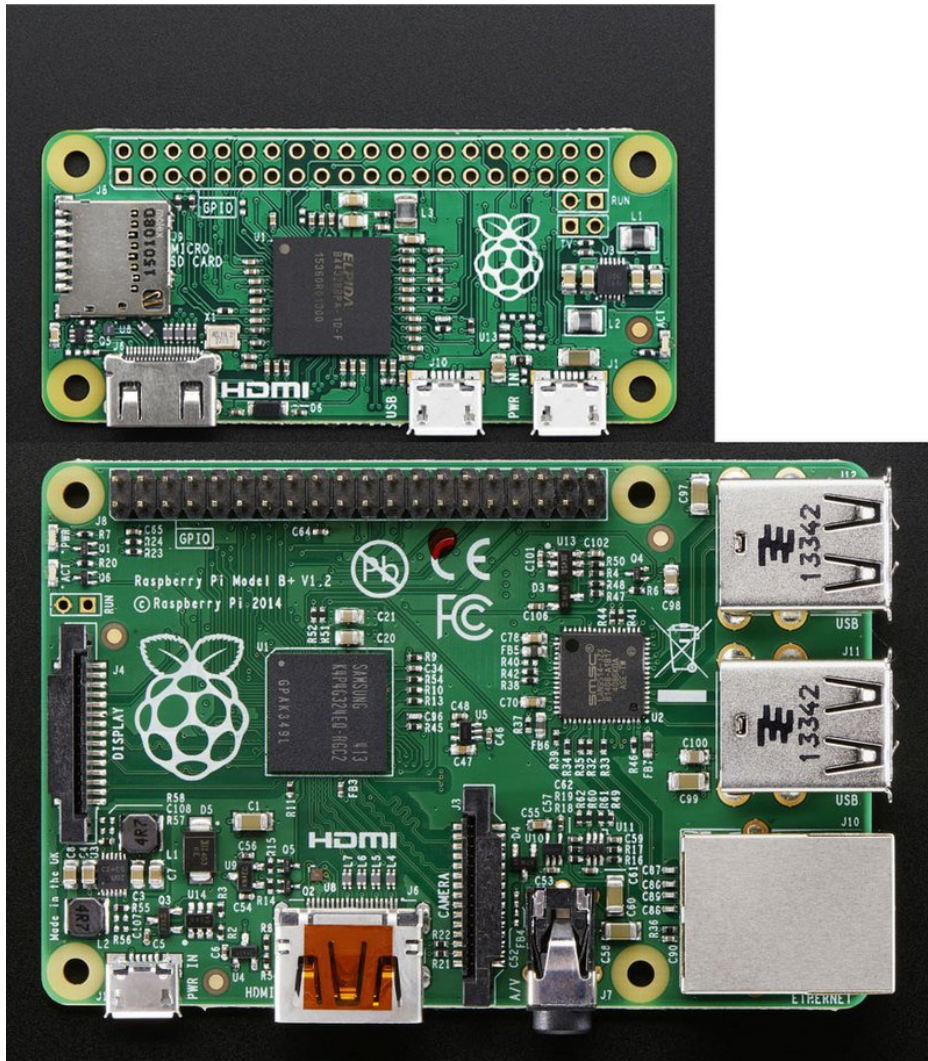


Way smaller than the Pi 2 or B+ and even smaller than the A+, its 60% the size of the A+: same length, and about half the width:



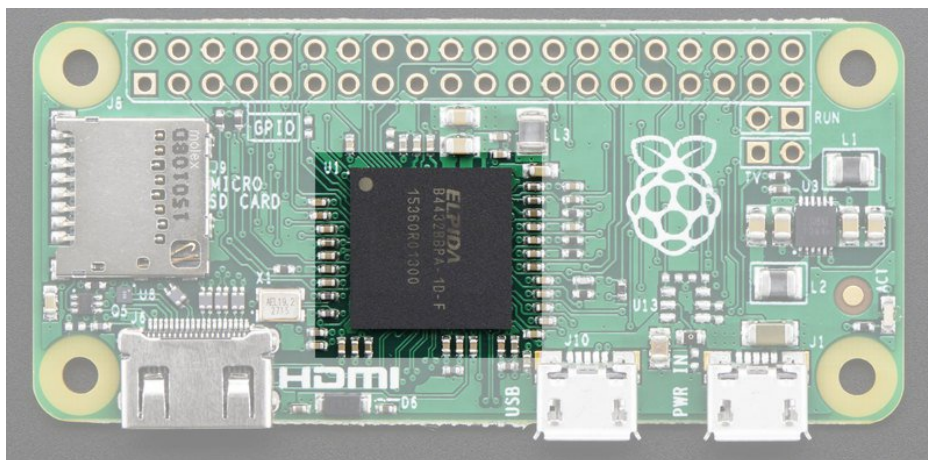
And about 40% the size of the Pi 2 or B+





## Processor and Speed

To keep the Pi Zero low cost, the processor and RAM are kept pretty basic. Instead of the Pi 2's zippy quad core ARM v7, we're back to a single-core 1GHz ARM (same processor in the Pi Model B+ and A+). We also have 512 MB of RAM with a 'package-on-package' setup. The chip shown here:



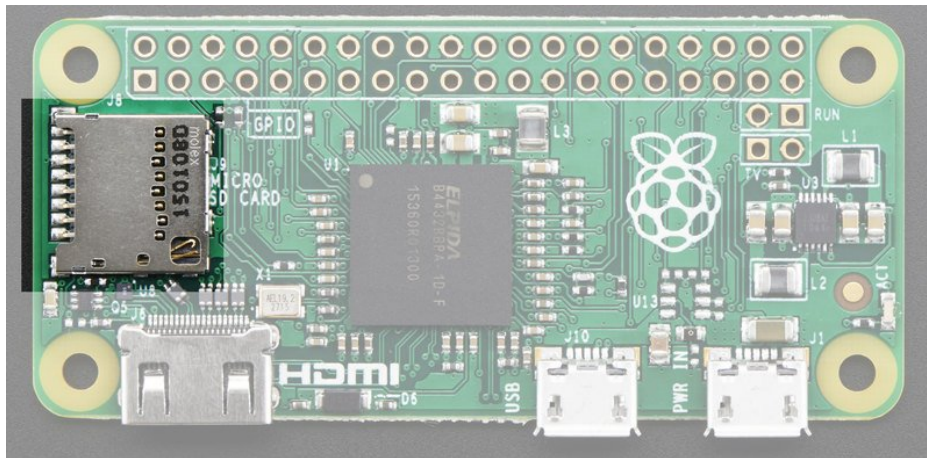
Is the RAM that is sitting *on top* of the main processor.

For maker and hacker projects, this isn't a big deal. You're essentially going to get the same performance as the Pi A+ or B+. If you're looking for something that can do some more serious processing, [check out the Pi 2](http://adafru.it/2358) (<http://adafru.it/2358>)

## Micro SD Card Holder

---

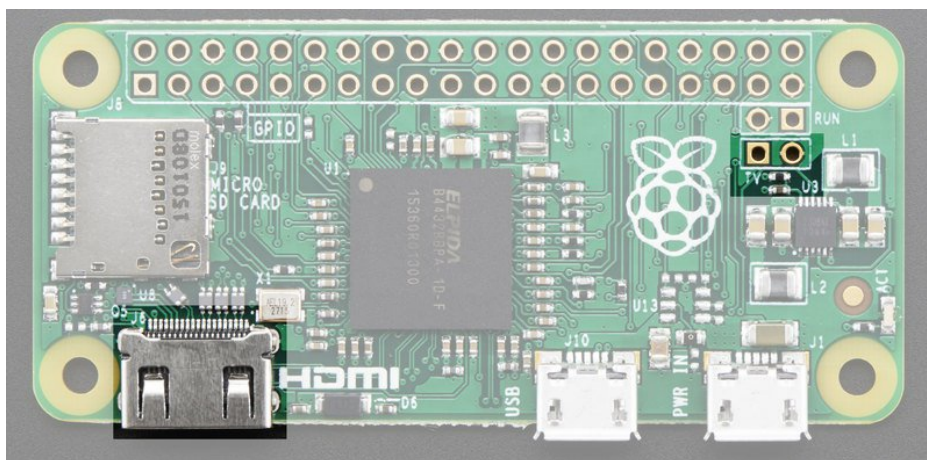
Not much has changed here, we're still going with MicroSD for size and ease of use (they're the most common card size these days!) This time the card holder is up top and is **push-pull** style not **push-push**. Honestly, I prefer it this way since you won't accidentally 'push-pop' the card out



## Video Out

---

HDMI Video is still available, you'll want to use a [Mini to Standard HDMI adapter](http://adafru.it/2819) (<http://adafru.it/2819>) to connect an HDMI cable. There's no 3.5mm jack with composite out, however you can get PAL or NTSC out via two 0.1" pads. [We've got a bigger write-up here about Pi Zero video outputs.](https://adafru.it/jEf) (<https://adafru.it/jEf>)



## Audio out

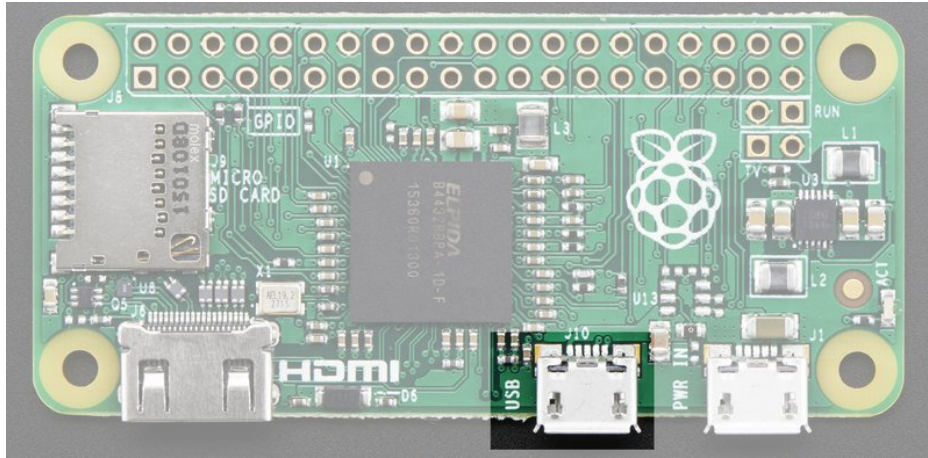
---

No analog audio out, but if you connect HDMI to a monitor with speakers you will get HDMI digital audio. It's also possible to hack analog audio out with a few passive components, [see our more detailed look at Pi Zero audio output options.](https://adafru.it/jEh) (<https://adafru.it/jEh>)



## USB Port

Like the Pi Model A+, the Pi Zero **does not have a USB Hub built in** which means you get **one USB port!** Moreover that USB port is not a standard type A port, instead it is a 'USB On-The-Go' port



In order to connect a USB device (mouse, keyboard, WiFi) etc you'll need a **USB OTG micro B to A cable** (<http://adafru.it/1099>):



If you need to connect multiple USB devices, a simple USB hub will do what you need. **A powered hub is even better** (<http://adafru.it/961>), and will let you power high-current USB devices like WiFi adapters and even external USB hard-drives.



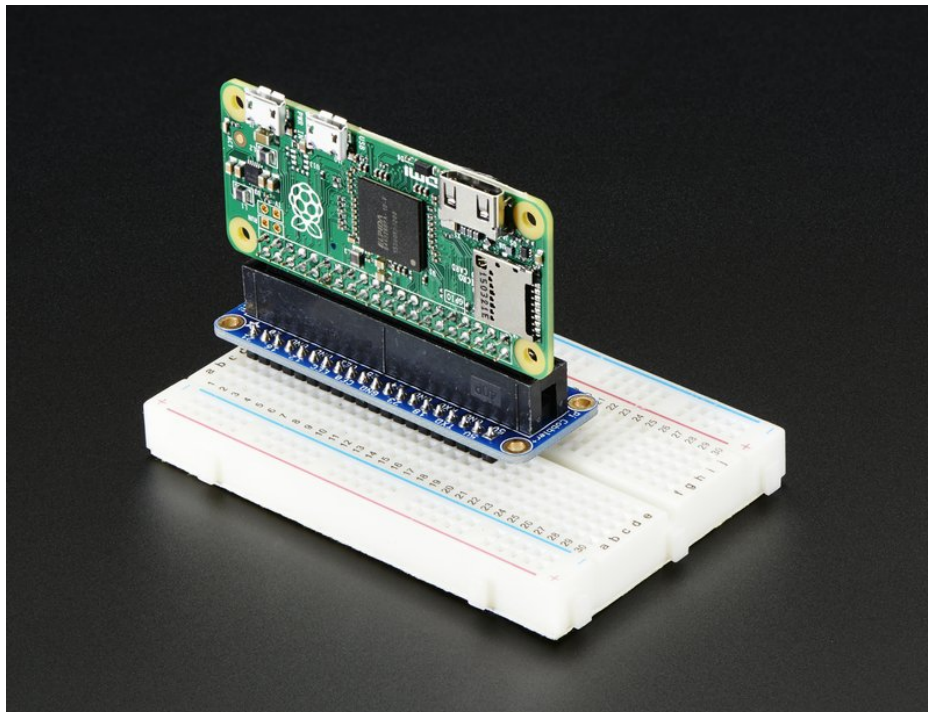


As a bonus you can power the Pi Zero *from* the hub (the power cable does not pass any data) - just plug the power micro USB cable into one of the ports.

## GPIO Header

---

To keep the Zero as simple and small as possible, the 'normal' GPIO header spot has been left blank! Normally, [a 2x20 male header is soldered in there](http://adafru.it/2822) (<http://adafru.it/2822>). While you could grab one of those and solder them in, the empty spot has a lot of potential. For example, you can solder in right-angle socket header, and turn the Pi Zero it a sort of 'daughter card'



We've got more ideas and suggestions on our GPIO header detail page (<https://adafru.it/jEi>)

## Setting up your SD card



Before you can power up your Pi Zero, you will need to program in the SD card with an **Operating System**

Much like your computer has Windows, Mac OS X or Linux on it to make it run, the Raspberry Pi needs something to help it boot and run software. That software is **Raspbian Linux** (a *flavor* of Debian Linux). [You can check out our tutorial on What Is Linux if you're curious to learn more \(https://adafru.it/jDZ\)](https://adafru.it/jDZ)

If you just want to get rockin, grab the [latest \(https://adafru.it/fQi\)](https://adafru.it/fQi) **Raspbian Jessie** operating system [from the Raspberry Pi downloads page \(https://adafru.it/fi7\)](https://adafru.it/fi7)

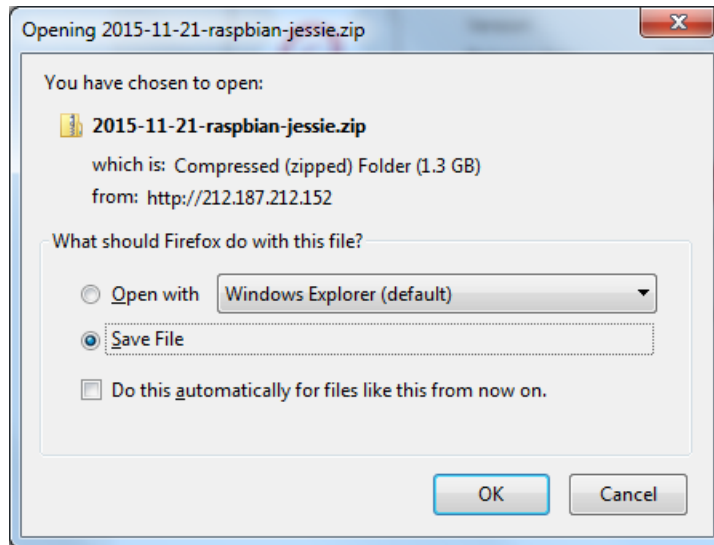
Just click the button below!

<https://adafru.it/jE0>

<https://adafru.it/jE0>



Raspbian Wheezy 5-15 or earlier do not support the Zero! Try Jessie instead

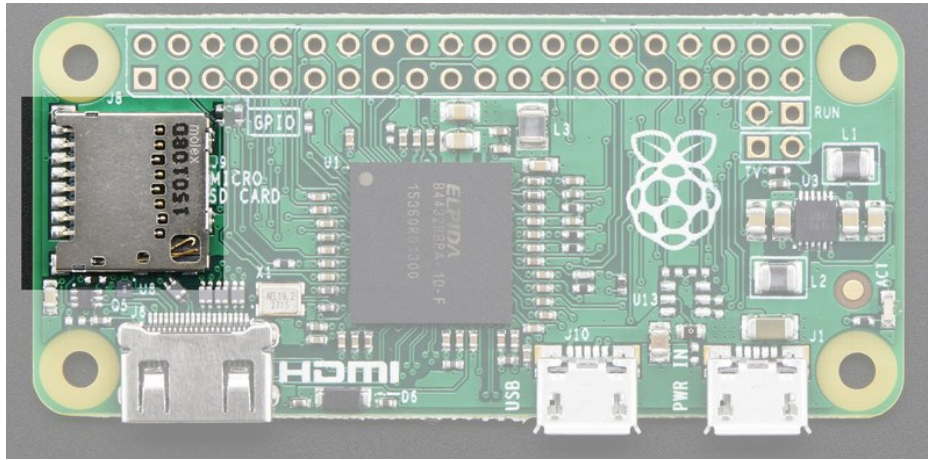


Once downloaded, unzip the zip file, the full image is about 4.5 Gigabytes.

Next up grab your SD or micro SD card reader and plug it into your computer



Now follow our guide for [Windows](https://adafru.it/jE4) (<https://adafru.it/jE4>) or [Mac OS X](https://adafru.it/jE5) (<https://adafru.it/jE5>) to burn the image



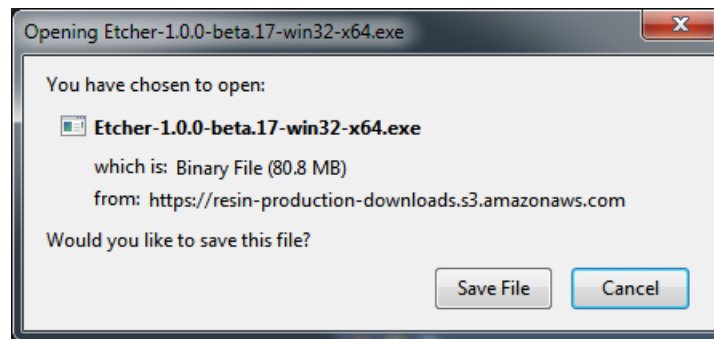
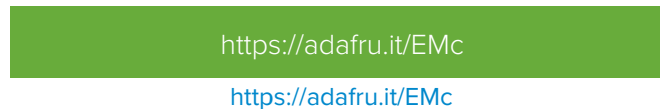
Once you're done, plug the micro SD card into the slot indicated. It will fit snugly in place but you won't hear or feel a 'click'

## Making an SD Card – Using Windows

We really like using balenaEtcher for burning SD cards. Works great on any version of Windows, macOS and Linux. It will not over-write your backup disk drive, and can handle compressed images so you do not need to unzip them!

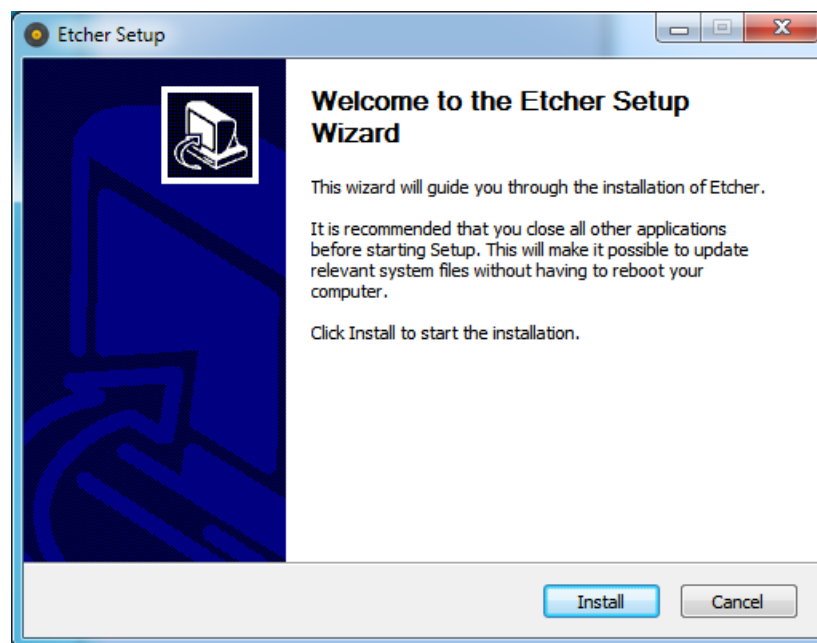
### Step 1.

Download Etcher from: <https://www.balena.io/etcher/> (<https://adafru.it/EMc>)



### Step 2.

Run the downloaded app to install!



You can start immediately, doubleclick the Etcher desktop icon, or select it from the Start menu

### Step 3.

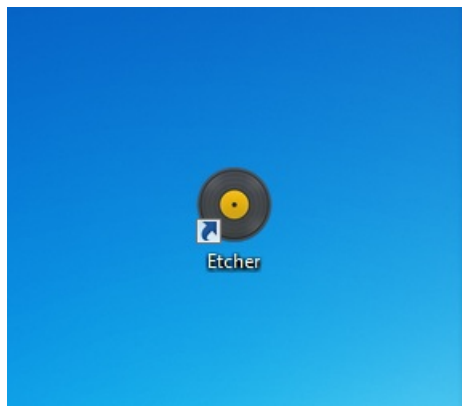


Eject any external storage devices such as USB flash drives and backup hard disks. This makes it easier to identify the SD card. Then insert the SD card into the slot on your computer or into the reader.

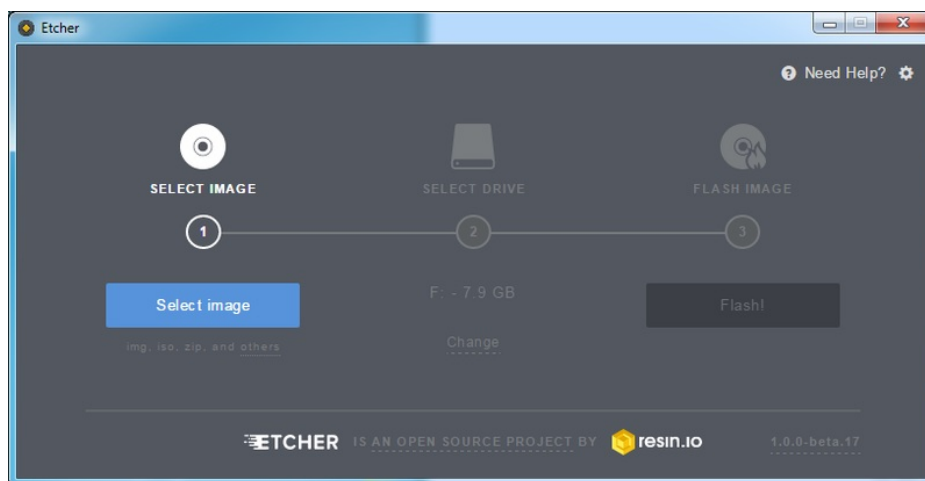
## Step 4.

---

Run the Etcher program



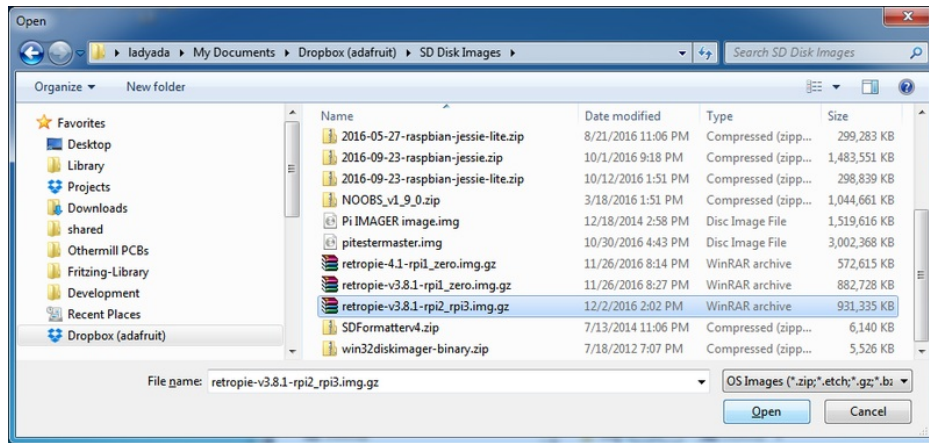
This will launch the following application.



## Step 5.

---

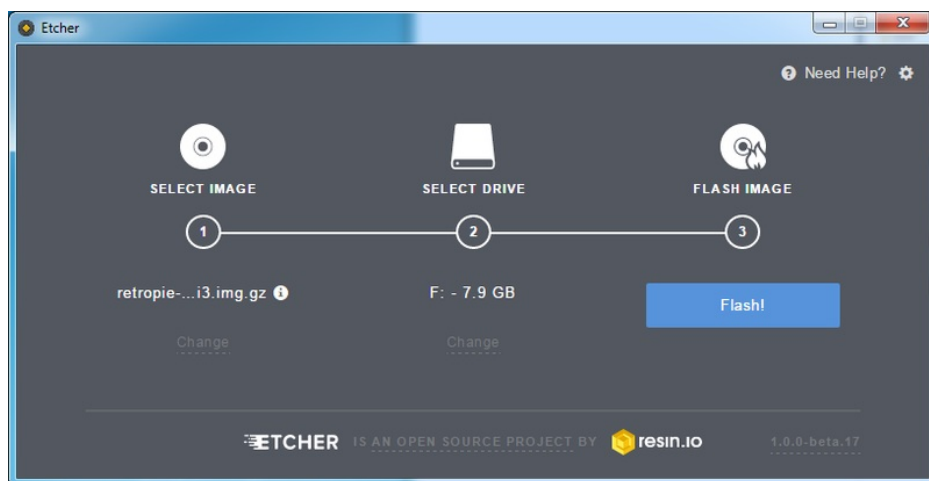
Select the image file by clicking **Select Image** you can select a compressed file such as a **.zip** or **.gz**



## Step 6.

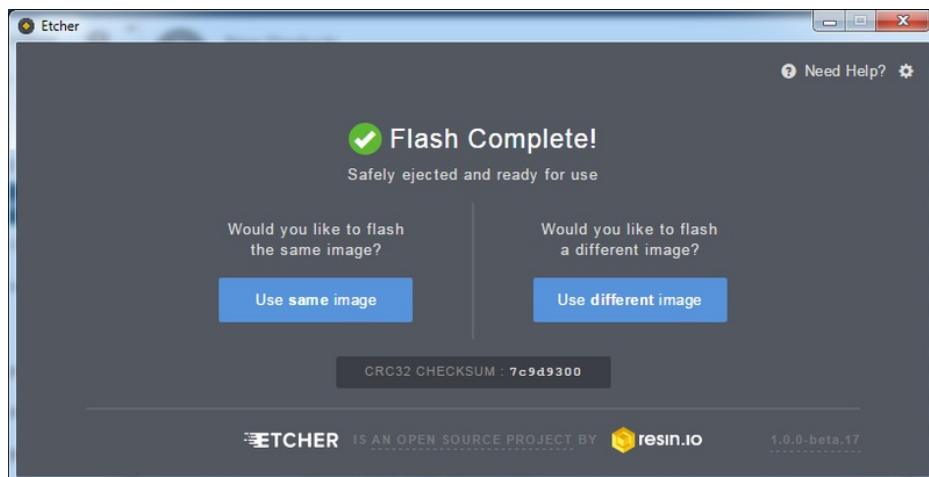
Etcher will automatically try to detect the SD drive, check the size to make sure its the right one

Then click **Flash!**



Check that you have the right device, as it will be reformatted, and then click Install.

It will take a few minutes to install, but once the SD card is ready, you will see the following.



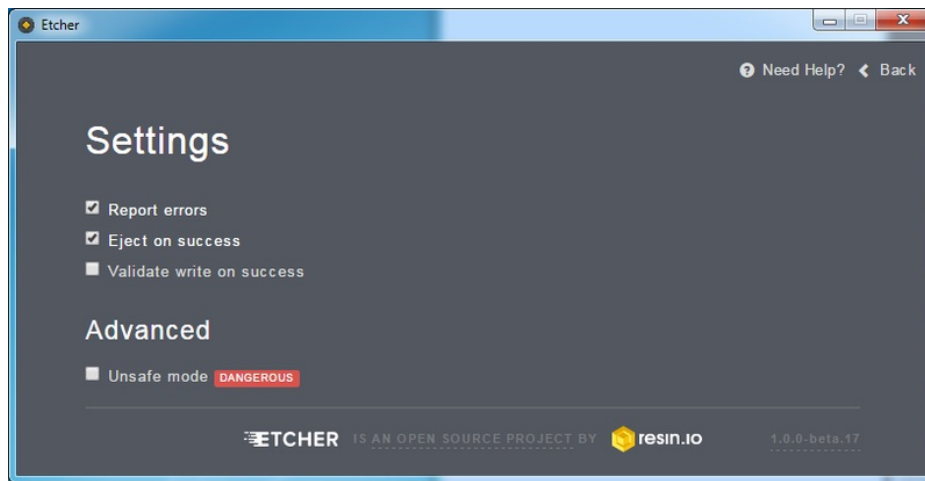


That's all there is to it. Your SD card is ready for use in your Raspberry Pi.

## Faster writes

---

If you burn a lot of cards, speed it up by turning off **Validate write on success**



## Making an SD Card – Using a Mac

We really like using balena**Etcher** for burning SD cards. Works great on Mac OS X 10.9 or later, won't over-write your backup disk drive, and can handle compressed images so you do not need to unzip them!

### Mac OS Catalina Issues

If you are having issues running Etcher on the Catalina release of Mac OS, see the links below for more information and some suggested workarounds.

- [Issue 2833 \(https://adafru.it/GB4\)](https://adafru.it/GB4)
- [Issue 2911 \(https://adafru.it/GB5\)](https://adafru.it/GB5)
- [Balena forum post \(https://adafru.it/GB7\)](https://adafru.it/GB7)

Most success has been reported by simply running Etcher from the command line using sudo:

```
sudo /Applications/balenaEtcher.app/Contents/MacOS/balenaEtcher
```

### Step 1.

---

Download Etcher from <https://www.balena.io/etcher/> (<https://adafru.it/EMc>)

<https://adafru.it/EMc>

<https://adafru.it/EMc>

### Step 2.

---

Open the downloaded disk image and drag the balenaEtcher application to the Applications folder. You can then eject the disk image.

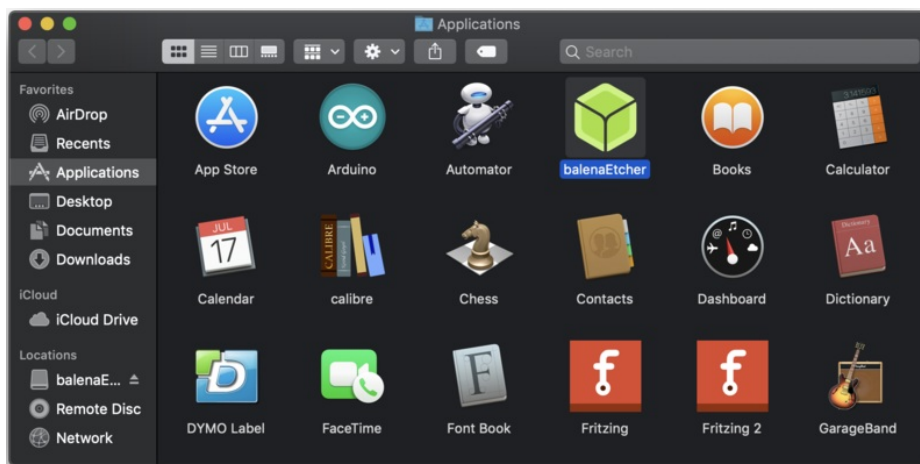


### Step 3.

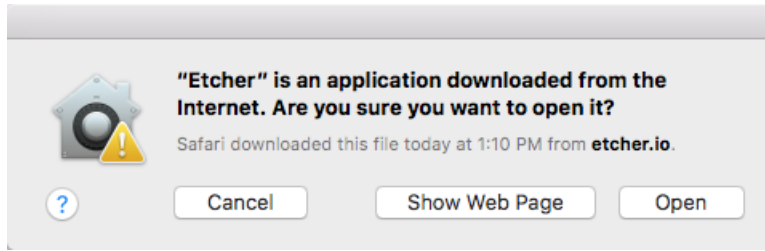
Eject any external storage devices such as USB flash drives and backup hard disks. This makes it easier to identify the SD card. Then insert the SD card into the slot on your computer or into the reader.

### Step 4.

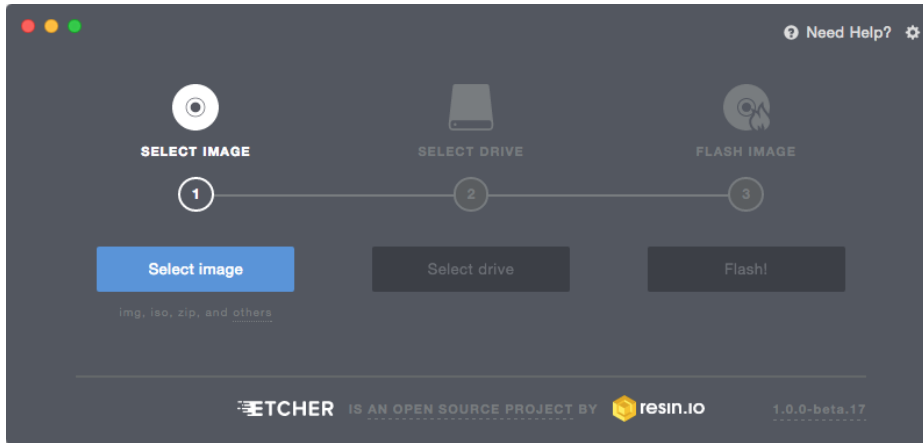
Run the Etcher application.



The first time you run Etcher you'll be asked to confirm the download. Click "Open" to continue.

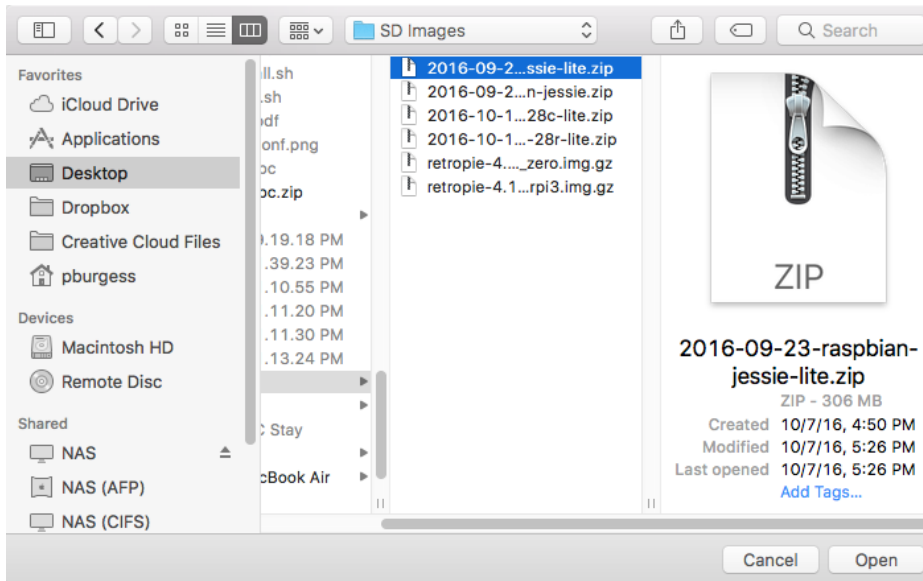


This will launch the Etcher application...



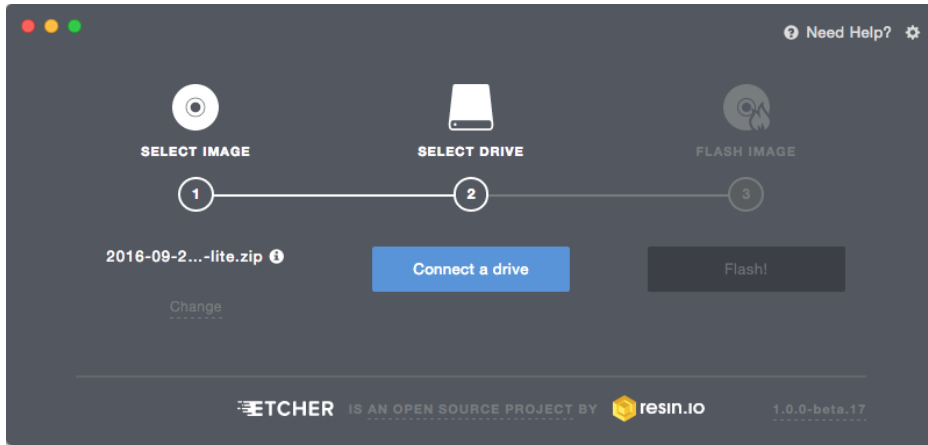
## Step 5.

Select the SD card image file by clicking **Select Image**. You can choose a compressed SD image file such as a **.zip** or **.gz** or an uncompressed **.img**, it's all good!



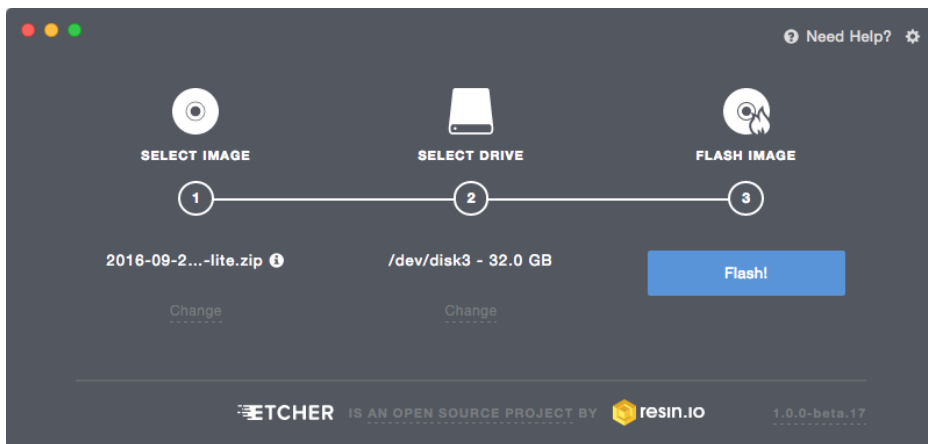
## Step 6.

Etcher will automatically try to detect the SD drive. If you don't have an SD card currently inserted, you'll be prompted to connect one.



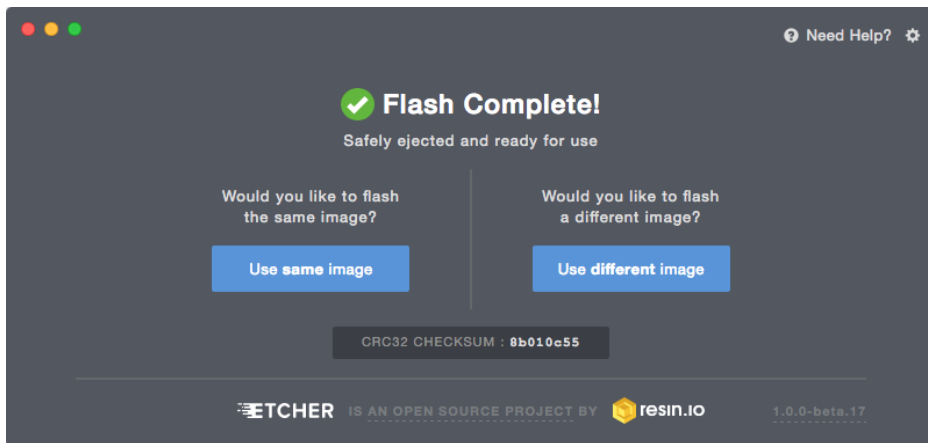
Check the disk size to make sure its the right one, that it's not overwriting your main drive or anything nasty.

Then click **Flash!** *A-ah!*



Etcher will work for a few minutes to “burn” the SD image to the card. You’ll see a progress bar as it works. This is about the time you’ll wish you’d splurged on a high-speed card.

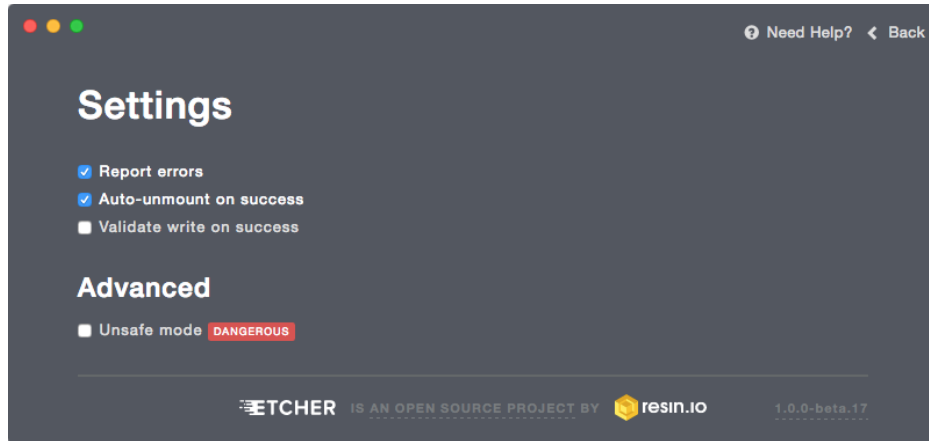
Once the SD card is ready, you will see the following:



The card will be unmounted automatically, so you can pull it out now and use it in your Raspberry Pi.

## Faster Writes

If you find yourself burning a lot of SD cards, you can speed things up by clicking the gear icon at the top-right, then turn off the “Validate write” option. I’ve written *hundreds* of cards and only had *one* fail validation.

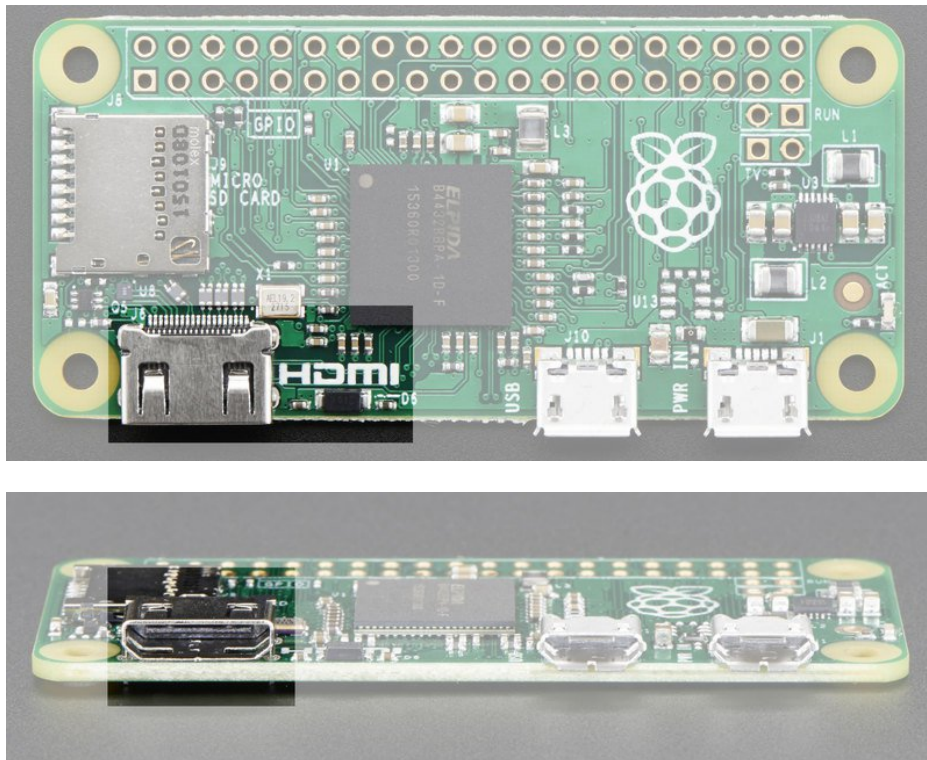


## Video Outputs

The Raspberry Pi chipset was originally designed to be a HDMI/graphics co-processor for mobile devices. For that reason, it has quite a bit of 'HDMI horsepower' and can, despite it's small size, play 1080p video at full screen.

### HDMI Video Out

The easiest & fastest way to get video going is to connect up an HDMI display (<https://adafru.it/jsb>). We have a ton of options, and any HDMI display size from 640x480 up to 1920x1080 will work. The Mini HDMI port is conveniently labeled and shown below:



For example, our 5" HDMI touch backpack which is the smallest all-in-one display we carry can be powered from the Pi Zero's USB port and provide a touchscreen at the same time (<http://adafru.it/2260>)



*(Shown here with a Pi 2 because, well, the Pi Zero wasn't out at the time)*

To connect an HDMI device, you'll need 2 things, a [Mini HDMI to HDMI Adapter](http://adafru.it/2819) (<http://adafru.it/2819>) and an [HDMI Cable](http://adafru.it/608) (<http://adafru.it/608>)

The HDMI cable is pretty straight-forward to understand, and you can get one anywhere. The HDMI adapter is required because the Pi Zero does not have a standard size HDMI port, instead the port is slimmer and smaller to keep the Zero petite. The adapter is pretty straight forward to use - plug it into the Pi Zero and the port is now large enough for any standard HDMI cable





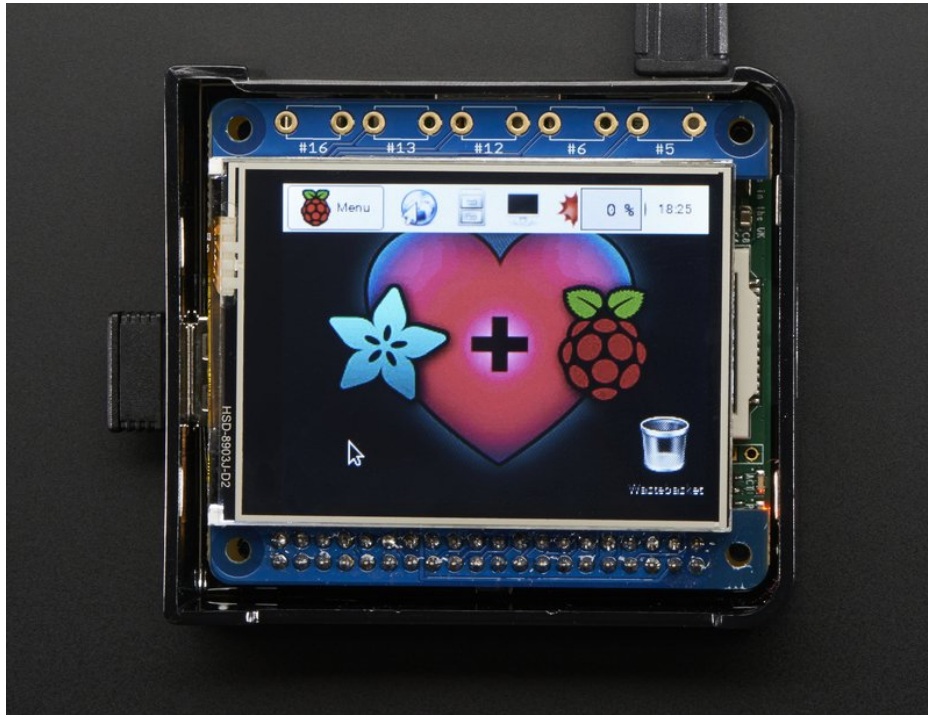
## PiTFT Video

---

Even though it is 'half size' of the A+, [you can still use any of our PiTFT's on the Pi Zero](https://adafru.it/jE7) (https://adafru.it/jE7) You can use any size from our 2.2" 320x240 PiTFT HAT, up to our 3.5" Touchscreen 480x320.

Before you can plug in a HAT or PiTFT [you'll need to solder in the 2x20 male header](http://adafru.it/2822) (http://adafru.it/2822)

Then follow the tutorial for the PiTFT of your choice! Be sure to pick the Jessie install image



## VGA Video Out

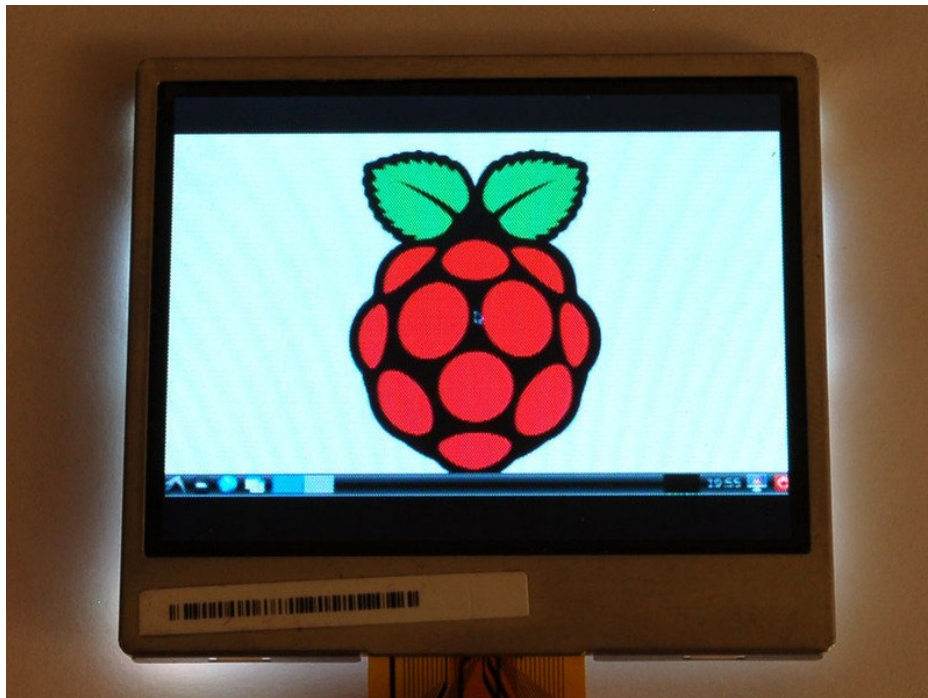
---

This one is pretty easy, just use the HDMI adapter above, and an [HDMI to VGA adapter](http://adafru.it/1151) (this also has the benefit of giving you an audio output) (http://adafru.it/1151)

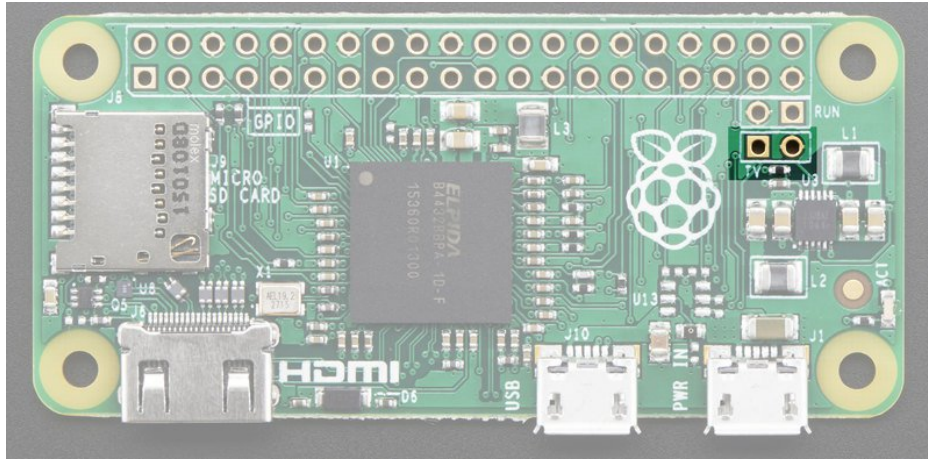


## NTSC/PAL Composite Video

OK so you want TV video? Maybe for one of our [very tiny composite video screens \(https://adafru.it/jE8\)](https://adafru.it/jE8)?



Well, the quality is not going to be nearly as nice as with VGA or HDMI but you can do it. Find the two pads marked **TV** on the 'Zero



The hole on the left, nearest to the **TV** text, is the signal (+) line, the pin to the right of it is the ground (-) line. Solder two wires to these pads and connect them to an [RCA Jack](http://adafru.it/2792) (<http://adafru.it/2792>) like this one



Make sure to not have HDMI plugged in, it should auto-switch to TV out. If you have somehow set your Pi for HDMI out only, plug your HDMI screen back in, or use a console cable to connect and log into the Pi. Then run `sudo raspi-config` at a command line to set video output to composite! You'll also want to tweak your Pi to use composite in the nicest resolution possible (<https://adafru.it/diN>)



# Audio Outputs

Uh, well, there aren't any! That's right, to keep the Pi Zero small and low cost, the headphone audio filter isn't included. You can still get digital audio out via HDMI so if you plug it your Pi into a monitor with speakers, that will work fine.

Well, ok that's not the whole truth

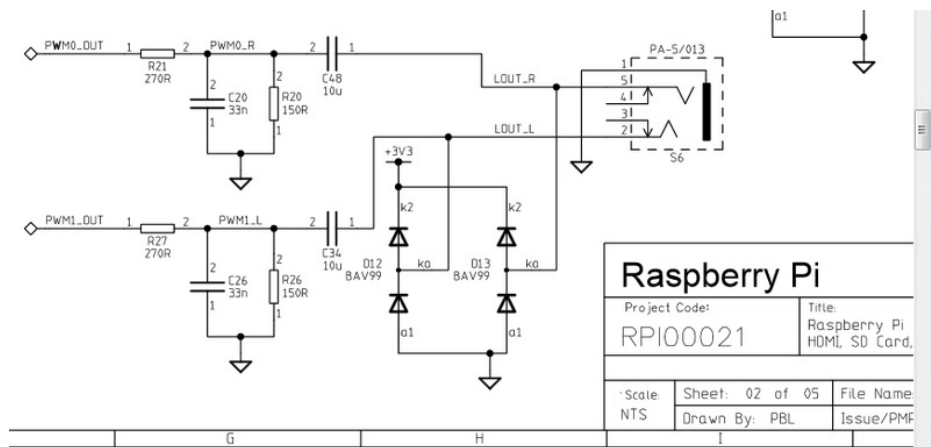
## How to Add Audio Outputs to your Pi Zero

Hey, wanna do the below but with step-by-step instructions? We wrote a tutorial!

[Click here to do the thing @ https://learn.adafruit.com/adding-basic-audio-ouput-to-raspberry-pi-zero](https://learn.adafruit.com/adding-basic-audio-ouput-to-raspberry-pi-zero) (<https://adafru.it/jZD>)

## How Other Pi's Create Audio

GPIO #18 is also known as PWM0 and in the original Pi was coupled with a very basic RC filter to create the audio output:



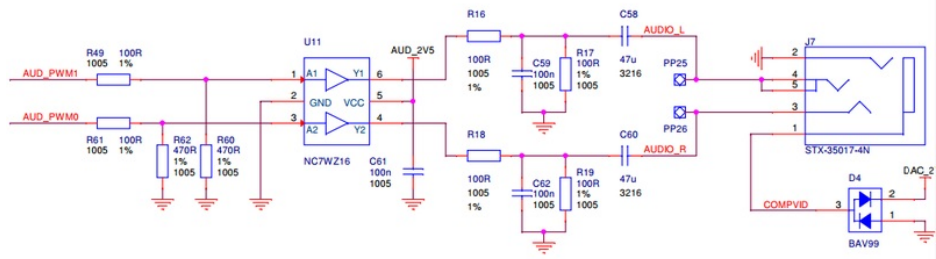
If you don't mind getting a few 150 and 270 ohm resistors, and two each of about 33nF (also known as 0.033uF) and 10uF capacitors, you can basically recreate those two filters.

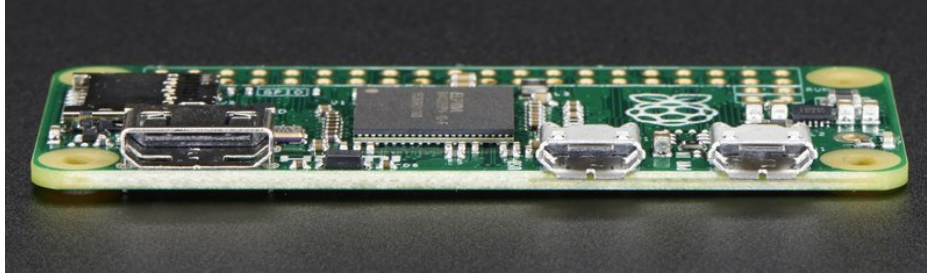
Now all you need is access to PWM0\_OUT and PWM1\_OUT, which are...on GPIO #40 and #45 and are not brought out on the Pi Zero. Tragedy? Give up? No! You can get to **PWM0** on GPIO #18 (ALT5) and **PWM1** on GPIO #13 (ALT0) or GPIO #19 (ALT5) - [see the full list of pins and alternate functions here \(https://adafru.it/jEa\)](https://adafru.it/jEa)

You can do *that* by adjusting the device tree overlay to change the PWM audio pins from pins #40 and #45 (which are not accessible) to pins #18 and #13 [This very nice Pi forum thread will tell you how! \(https://adafru.it/jEb\)](https://adafru.it/jEb)

See here for a [program that will let you set the alt forms of GPIO pins \(https://adafru.it/jEc\)](https://adafru.it/jEc)

If you want a higher quality audio output, the B+ and Pi 2 use this schematic - it has a driving buffer on the audio PWM lines for better current drive *and* it uses a cleaner 2.5V reference for better quality audio.





The most intriguing difference for hackers and makers is that the Pi Zero does not come with the soldered GPIO header. Partially this is to save cost, but it also allows the Pi Zero to be very thin and gives you the option of embedding it easily into a project box.

### Cons:

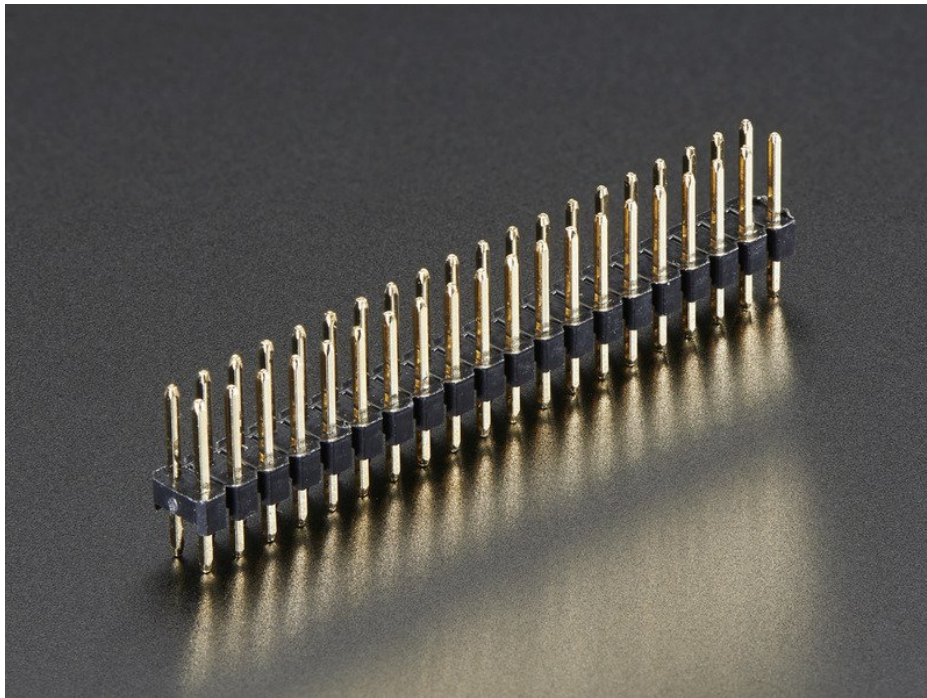
- You have to solder in the header to use Pi HATs and Pi toppers

### Pros:

- You can practice your soldering!
- Can skip the GPIO header to keep the Pi Zero super slim
- Solder wires directly into the GPIO pads, use only what you need
- Try different, exotic headers such as right angle or socket header

## Go Classic with 2x20 Male Header

Like blue jeans and Coca-Cola, [the 2x20 male header is the classic option.](http://adafru.it/2822) (<http://adafru.it/2822>)



Once soldered in, you can plug in any HAT or topper. The pinout is completely identical to the 2x20 headers on the Pi 2 and Pi A+ & B+



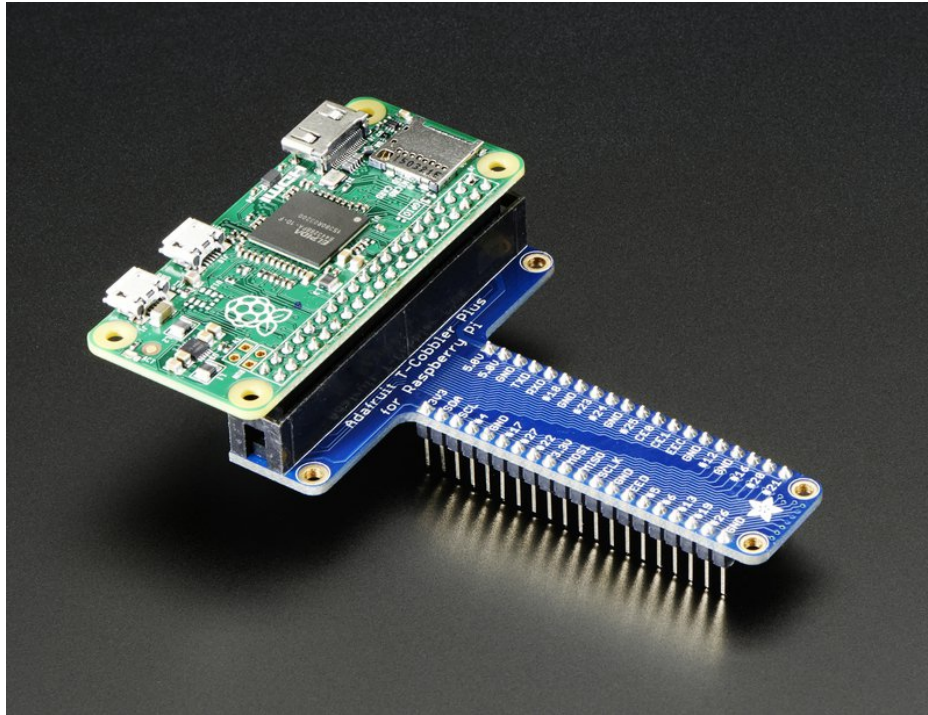
## Or 2x20 Female Socket Header

This one is interesting, [if you solder in a 2x20 female socket header](http://adafru.it/2222) (<http://adafru.it/2222>)



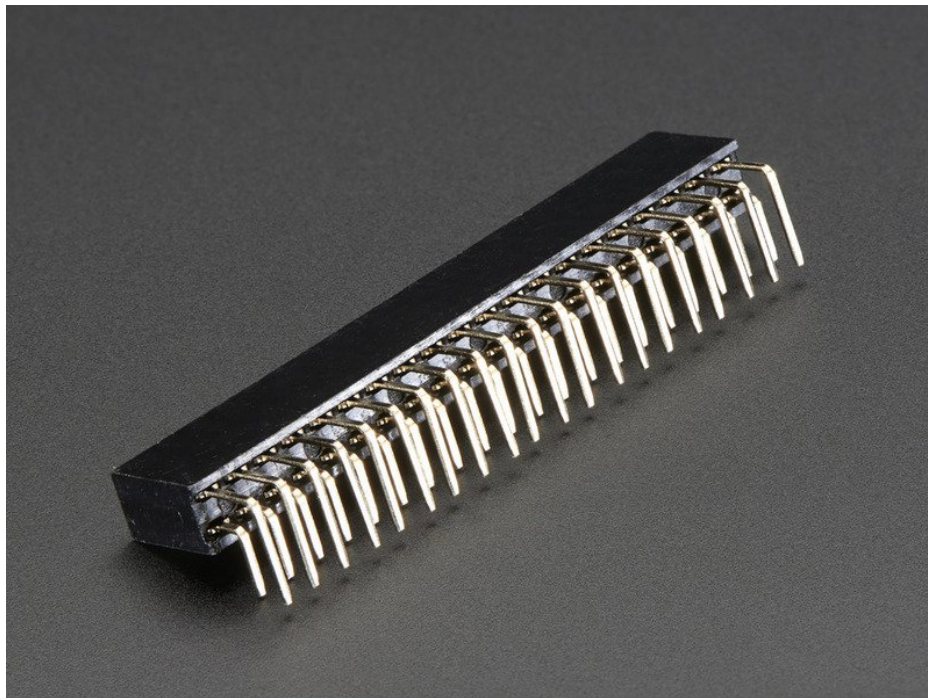
and attach it upside down you can plug it right into a T-Cobbler!





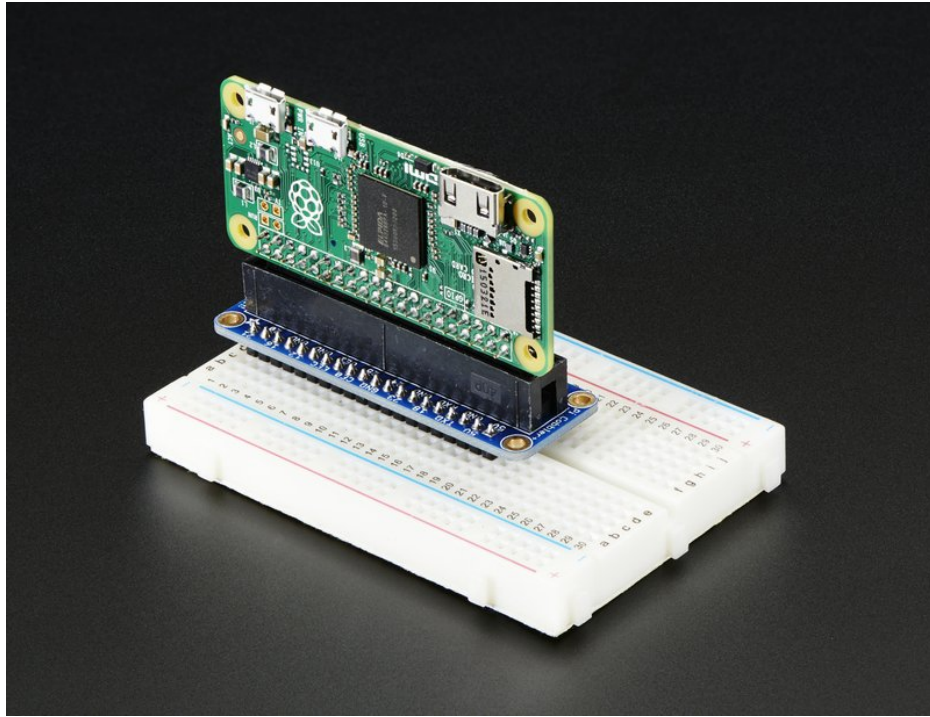
## Advanced 2x20 Right-Angle Female Socket Header

Or, take it even more *extreme* with [2x20 right angle female header](http://adafru.it/2823) (<http://adafru.it/2823>)



Now you can stick it into a Cobbler or T-Cobbler and it will sit sort of like a computer daughtercard!





## Is My Pi Zero Dead?

The Pi Zero is so minimal, it can be tough to tell if its working at all. Here's how to do a quick check (from [this sticky \(https://adafru.it/upa\)](https://adafru.it/upa))!

- Take your Zero, with nothing in any slot or socket (yes, no SD-card is needed or wanted to do this test!).
- Take a normal micro-USB to USB-A **DATA SYNC** cable (not a charge-only cable! make sure its a true data sync cable!)
- Connect the USB cable to your PC, plugging the micro-USB into the Pi's USB, (*not the PWR\_IM*).
- If the Zero is alive, your Windows PC will go ding for the presence of new hardware & you should see "BCM2708 Boot" in Device Manager.
- Or on linux, run `sudo lsusb` or run `dmesg` and look for a `ID 0a5c:2763 Broadcom Corp` message. If you see that, so far so good, you know the Zero's not dead.

I tested on Linux and here's my actual dmesg:

```
[226314.048026] usb 4-2: new full-speed USB device number 82 using uhci_hcd
[226314.213273] usb 4-2: New USB device found, idVendor=0a5c, idProduct=2763
[226314.213280] usb 4-2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[226314.213284] usb 4-2: Product: BCM2708 Boot
[226314.213288] usb 4-2: Manufacturer: Broadcom
```

but there will be NO LED LIGHT (looks so dead but its alive!)

## Is There Even Life?

You can skip this section unless you have reason to believe your Pi Zero isn't alive.

### THE ZERO DOES NOT HAVE A POWER LED

The Pi Zero doesn't have much in the way of blinky LEDs to give you a warm fuzzy that it's doing anything or even alive. And if the GPU doesn't find a valid OS image, it doesn't even turn on the green ACT LED and looks totally dead. Typically this just means something is up with the SD card. Bad card. Bad image. Out of date image. Whatever. **It does not mean the Pi Zero is dead.**

### Here's how to run a sanity check to verify if the Pi Zero is OK.

(taken from [here \(https://adafru.it/upa\)](https://adafru.it/upa) and also provided [here \(https://adafru.it/vle\)](https://adafru.it/vle))

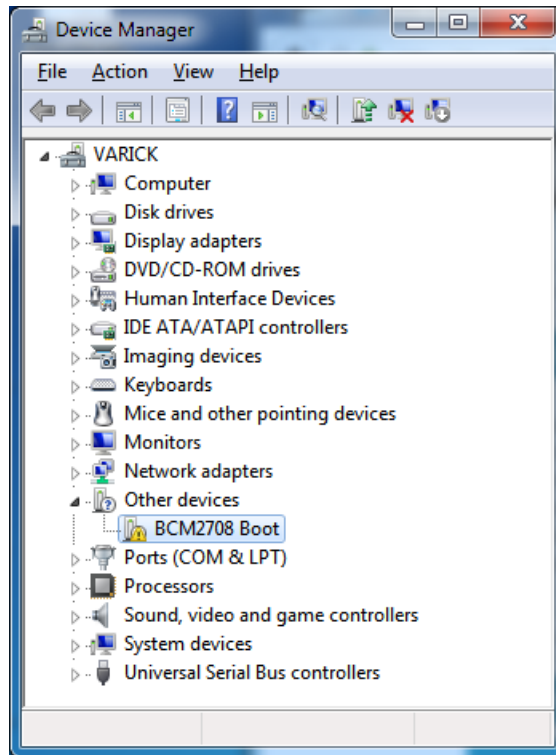
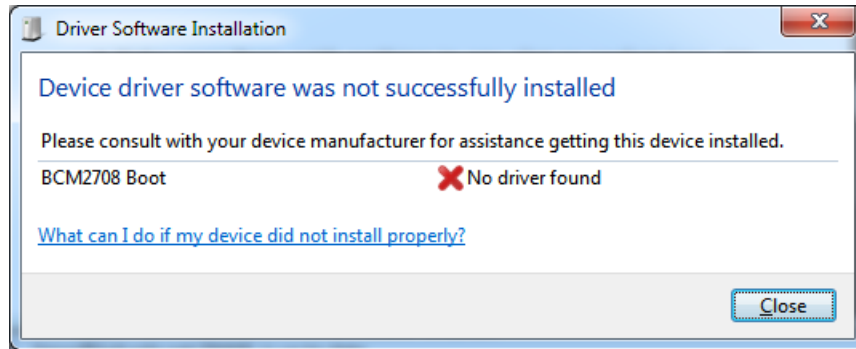
- Take your Zero, with nothing in any slot or socket (yes, **no SD-card is needed** or wanted to do this test!).
- Take a normal micro-USB to USB-A **DATA SYNC** cable (not a charge-only cable! make sure its a true data sync cable!)
- Connect the USB cable to your PC, plugging the micro-USB into the Pi's USB, (*not the PWR\_IM*).
- If the Zero is alive, your Windows PC will go ding for the presence of new hardware & you should see "BCM2708 Boot" in Device Manager.
- Or on linux, run `sudo lsusb` or run `dmesg` and look for a `ID 0a5c:2763 Broadcom Corp` message. If you see that, so far so good, you know the Zero's not dead.

It may take a few seconds for the messages to show up.

Below is a Pi Zero connected to a Linux computer via a USB cable and the resulting dmesg output. **Note: there is no SD card installed, USB cable is in USB port, and there are no lights.**

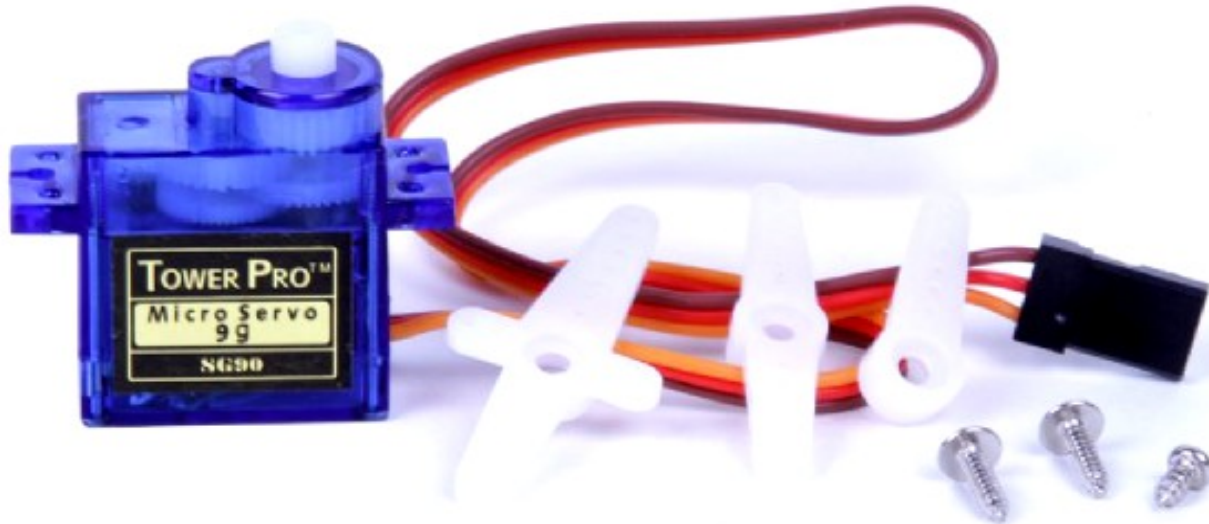


Here's what our Windows machine showed:

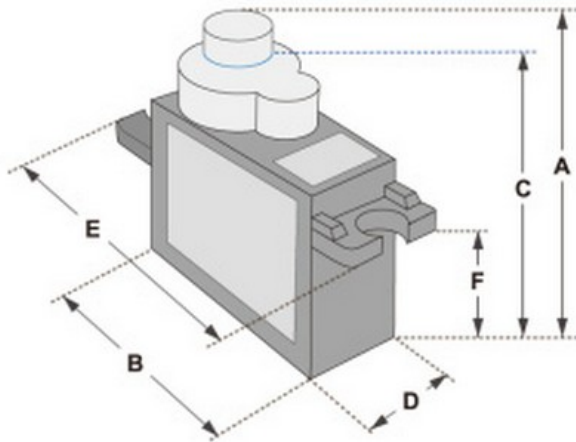


Looks dead, but it's not.



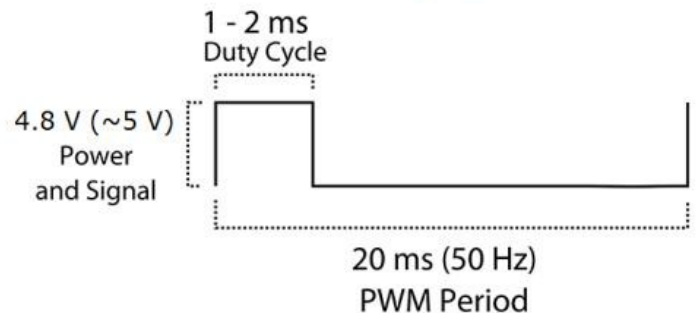
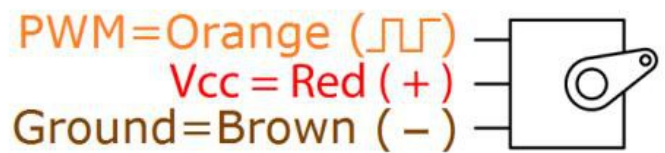


Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.



Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is middle, is all the way to the right, "-90" (~1ms pulse) is all the way to the left.



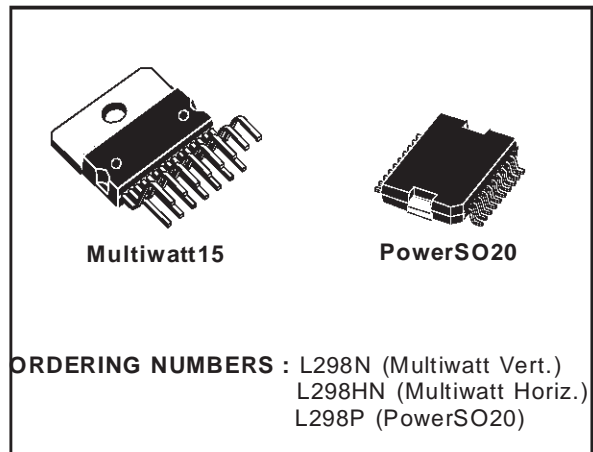
L298 Driver motor	2
L298N-module-information	16
IR sensor	21
MFRC522 Contactless reader IC	26
Raspberry PI	120
SERVO MOTOR SG90	DATA
SHEET	156

## DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)

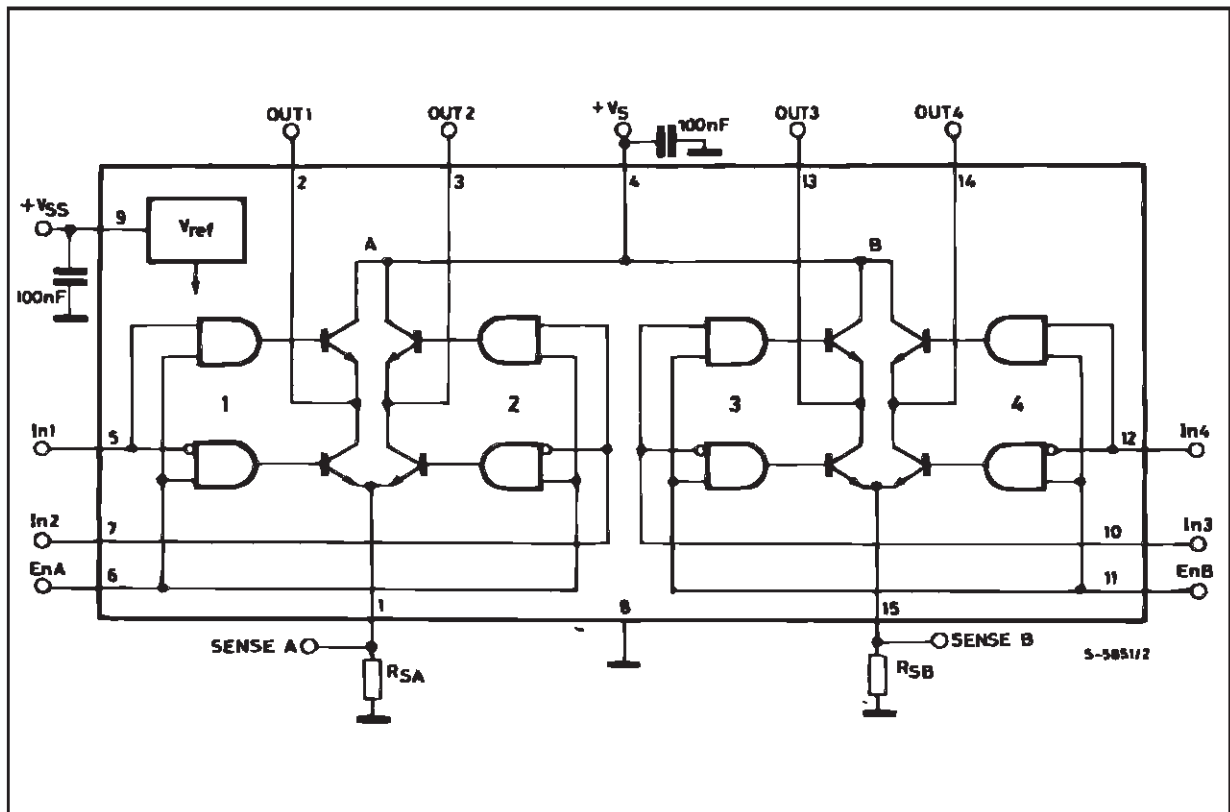
### DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the con-



nection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

### BLOCK DIAGRAM

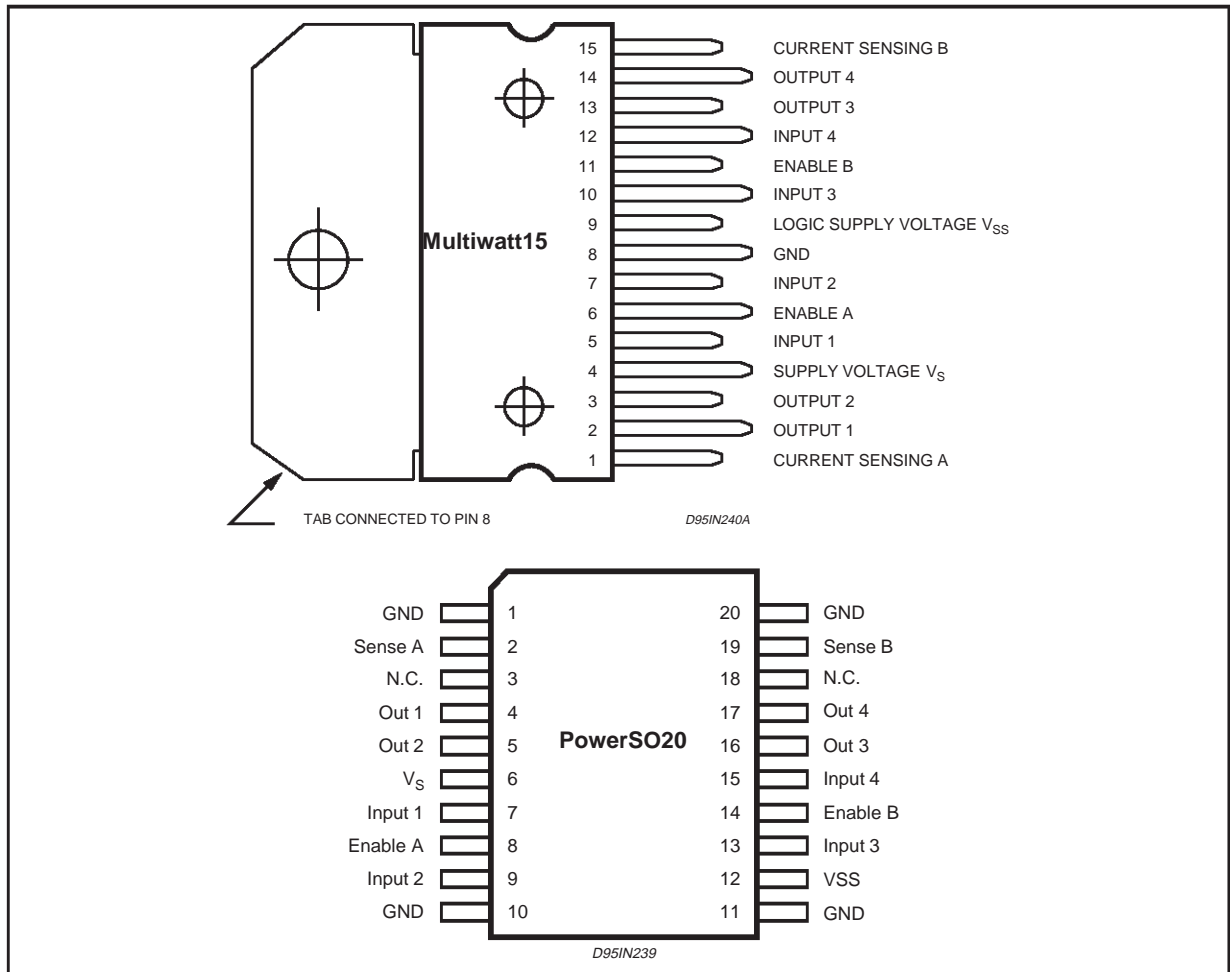




**ABSOLUTE MAXIMUM RATINGS**

Symbol	Parameter	Value	Unit
$V_S$	Power Supply	50	V
$V_{SS}$	Logic Supply Voltage	7	V
$V_I, V_{en}$	Input and Enable Voltage	-0.3 to 7	V
$I_O$	Peak Output Current (each Channel)		
	- Non Repetitive ( $t = 100\mu s$ )	3	A
	- Repetitive (80% on -20% off; $t_{on} = 10ms$ )	2.5	A
	-DC Operation	2	A
$V_{sens}$	Sensing Voltage	-1 to 2.3	V
$P_{tot}$	Total Power Dissipation ( $T_{case} = 75^\circ C$ )	25	W
$T_{op}$	Junction Operating Temperature	-25 to 130	$^\circ C$
$T_{stg}, T_j$	Storage and Junction Temperature	-40 to 150	$^\circ C$

**PIN CONNECTIONS (top view)**



**THERMAL DATA**

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th j-case}$	Thermal Resistance Junction-case	Max.	-	3	$^\circ C/W$
$R_{th j-amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	$^\circ C/W$

(\*) Mounted on aluminum substrate

## PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V <sub>S</sub>	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	V <sub>SS</sub>	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V<sub>S</sub> = 42V; V<sub>SS</sub> = 5V, T<sub>j</sub> = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V <sub>S</sub>	Supply Voltage (pin 4)	Operative Condition	V <sub>IH</sub> +2.5		46	V
V <sub>SS</sub>	Logic Supply Voltage (pin 9)		4.5	5	7	V
I <sub>S</sub>	Quiescent Supply Current (pin 4)	V <sub>en</sub> = H; I <sub>L</sub> = 0 V <sub>i</sub> = L V <sub>i</sub> = H		13 50	22 70	mA mA
		V <sub>en</sub> = L V <sub>i</sub> = X			4	mA
I <sub>SS</sub>	Quiescent Current from V <sub>SS</sub> (pin 9)	V <sub>en</sub> = H; I <sub>L</sub> = 0 V <sub>i</sub> = L V <sub>i</sub> = H		24 7	36 12	mA mA
		V <sub>en</sub> = L V <sub>i</sub> = X			6	mA
V <sub>iL</sub>	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V <sub>iH</sub>	Input High Voltage (pins 5, 7, 10, 12)		2.3		V <sub>SS</sub>	V
I <sub>iL</sub>	Low Voltage Input Current (pins 5, 7, 10, 12)	V <sub>i</sub> = L			–10	μA
I <sub>iH</sub>	High Voltage Input Current (pins 5, 7, 10, 12)	V <sub>i</sub> = H ≤ V <sub>SS</sub> –0.6V		30	100	μA
V <sub>en</sub> = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V <sub>en</sub> = H	Enable High Voltage (pins 6, 11)		2.3		V <sub>SS</sub>	V
I <sub>en</sub> = L	Low Voltage Enable Current (pins 6, 11)	V <sub>en</sub> = L			–10	μA
I <sub>en</sub> = H	High Voltage Enable Current (pins 6, 11)	V <sub>en</sub> = H ≤ V <sub>SS</sub> –0.6V		30	100	μA
V <sub>CEsat(H)</sub>	Source Saturation Voltage	I <sub>L</sub> = 1A I <sub>L</sub> = 2A	0.95	1.35 2	1.7 2.7	V V
V <sub>CEsat(L)</sub>	Sink Saturation Voltage	I <sub>L</sub> = 1A (5) I <sub>L</sub> = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V <sub>CEsat</sub>	Total Drop	I <sub>L</sub> = 1A (5) I <sub>L</sub> = 2A (5)	1.80		3.2 4.9	V V
V <sub>sens</sub>	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T <sub>1</sub> (V <sub>i</sub> )	Source Current Turn-off Delay	0.5 V <sub>i</sub> to 0.9 I <sub>L</sub> (2); (4)		1.5		μs
T <sub>2</sub> (V <sub>i</sub> )	Source Current Fall Time	0.9 I <sub>L</sub> to 0.1 I <sub>L</sub> (2); (4)		0.2		μs
T <sub>3</sub> (V <sub>i</sub> )	Source Current Turn-on Delay	0.5 V <sub>i</sub> to 0.1 I <sub>L</sub> (2); (4)		2		μs
T <sub>4</sub> (V <sub>i</sub> )	Source Current Rise Time	0.1 I <sub>L</sub> to 0.9 I <sub>L</sub> (2); (4)		0.7		μs
T <sub>5</sub> (V <sub>i</sub> )	Sink Current Turn-off Delay	0.5 V <sub>i</sub> to 0.9 I <sub>L</sub> (3); (4)		0.7		μs
T <sub>6</sub> (V <sub>i</sub> )	Sink Current Fall Time	0.9 I <sub>L</sub> to 0.1 I <sub>L</sub> (3); (4)		0.25		μs
T <sub>7</sub> (V <sub>i</sub> )	Sink Current Turn-on Delay	0.5 V <sub>i</sub> to 0.9 I <sub>L</sub> (3); (4)		1.6		μs
T <sub>8</sub> (V <sub>i</sub> )	Sink Current Rise Time	0.1 I <sub>L</sub> to 0.9 I <sub>L</sub> (3); (4)		0.2		μs
f <sub>c</sub> (V <sub>i</sub> )	Commutation Frequency	I <sub>L</sub> = 2A		25	40	KHz
T <sub>1</sub> (V <sub>en</sub> )	Source Current Turn-off Delay	0.5 V <sub>en</sub> to 0.9 I <sub>L</sub> (2); (4)		3		μs
T <sub>2</sub> (V <sub>en</sub> )	Source Current Fall Time	0.9 I <sub>L</sub> to 0.1 I <sub>L</sub> (2); (4)		1		μs
T <sub>3</sub> (V <sub>en</sub> )	Source Current Turn-on Delay	0.5 V <sub>en</sub> to 0.1 I <sub>L</sub> (2); (4)		0.3		μs
T <sub>4</sub> (V <sub>en</sub> )	Source Current Rise Time	0.1 I <sub>L</sub> to 0.9 I <sub>L</sub> (2); (4)		0.4		μs
T <sub>5</sub> (V <sub>en</sub> )	Sink Current Turn-off Delay	0.5 V <sub>en</sub> to 0.9 I <sub>L</sub> (3); (4)		2.2		μs
T <sub>6</sub> (V <sub>en</sub> )	Sink Current Fall Time	0.9 I <sub>L</sub> to 0.1 I <sub>L</sub> (3); (4)		0.35		μs
T <sub>7</sub> (V <sub>en</sub> )	Sink Current Turn-on Delay	0.5 V <sub>en</sub> to 0.9 I <sub>L</sub> (3); (4)		0.25		μs
T <sub>8</sub> (V <sub>en</sub> )	Sink Current Rise Time	0.1 I <sub>L</sub> to 0.9 I <sub>L</sub> (3); (4)		0.1		μs

- 1) Sensing voltage can be -1 V for t ≤ 50 μsec; in steady state V<sub>sens</sub> min ≥ -0.5 V.
- 2) See fig. 2.
- 3) See fig. 4.
- 4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current.

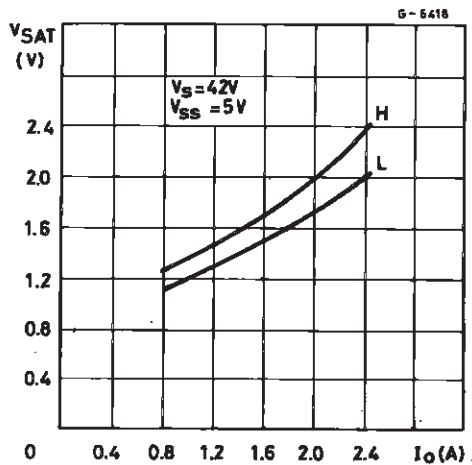
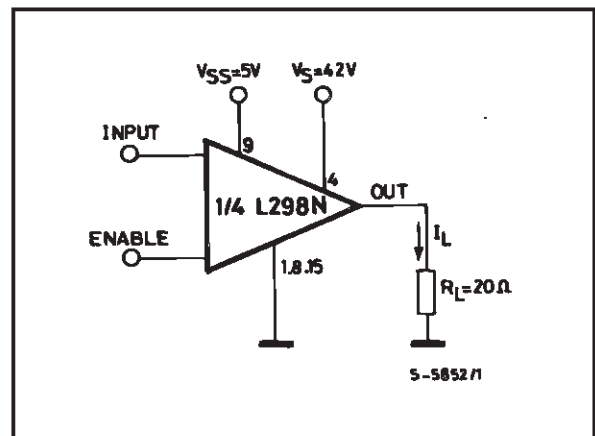
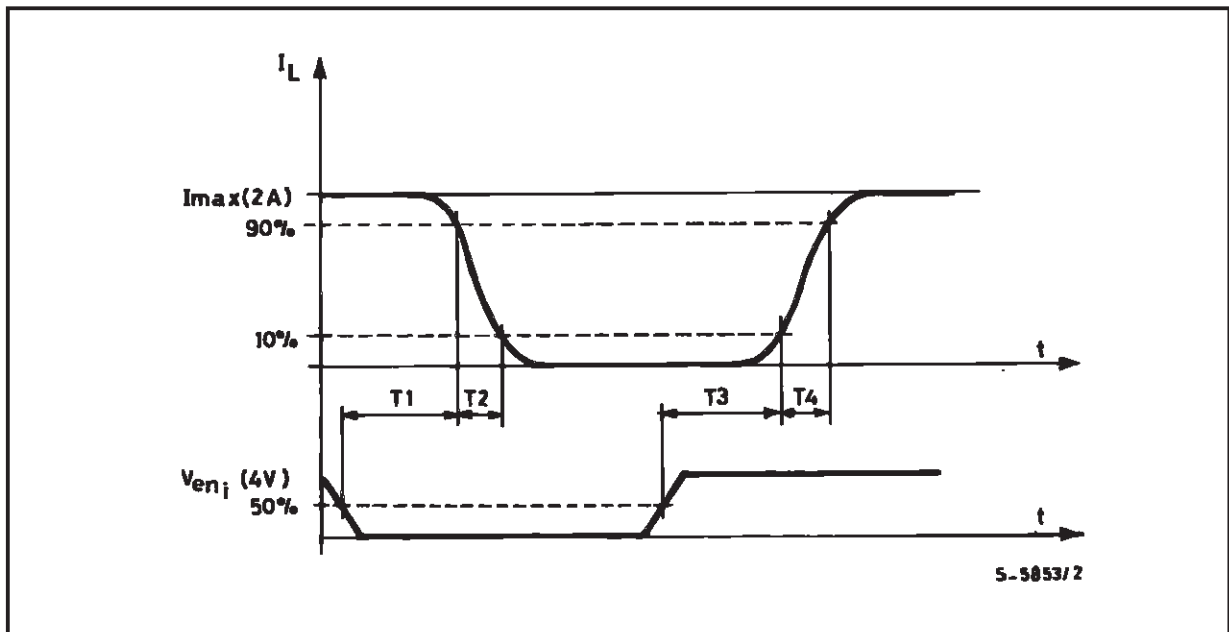


Figure 2 : Switching Times Test Circuits.

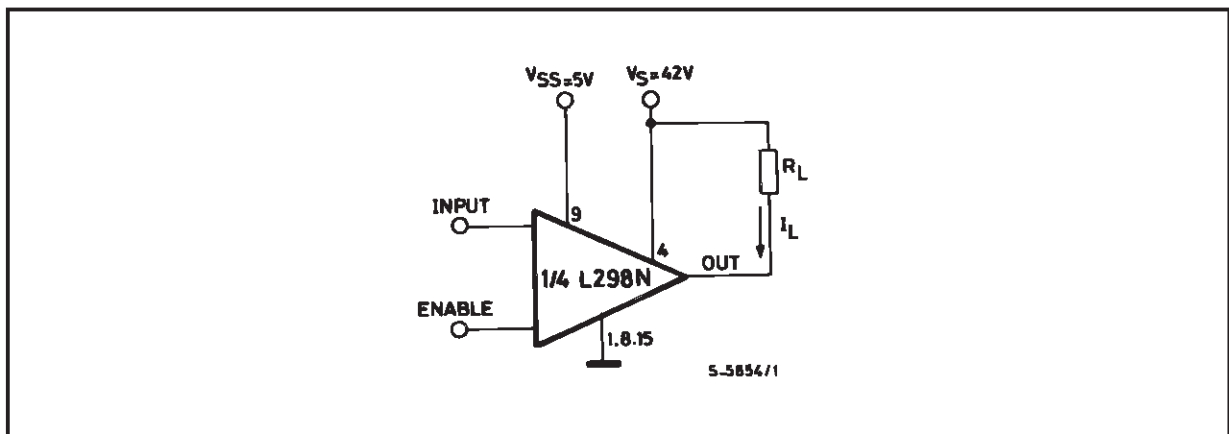


Note: For INPUT Switching, set EN = H  
 For ENABLE Switching, set IN = H

**Figure 3 :** Source Current Delay Times vs. Input or Enable Switching.



**Figure 4 :** Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H  
For ENABLE Switching, set IN = L

Figure 5 : Sink Current Delay Times vs. Input 0 V Enable Switching.

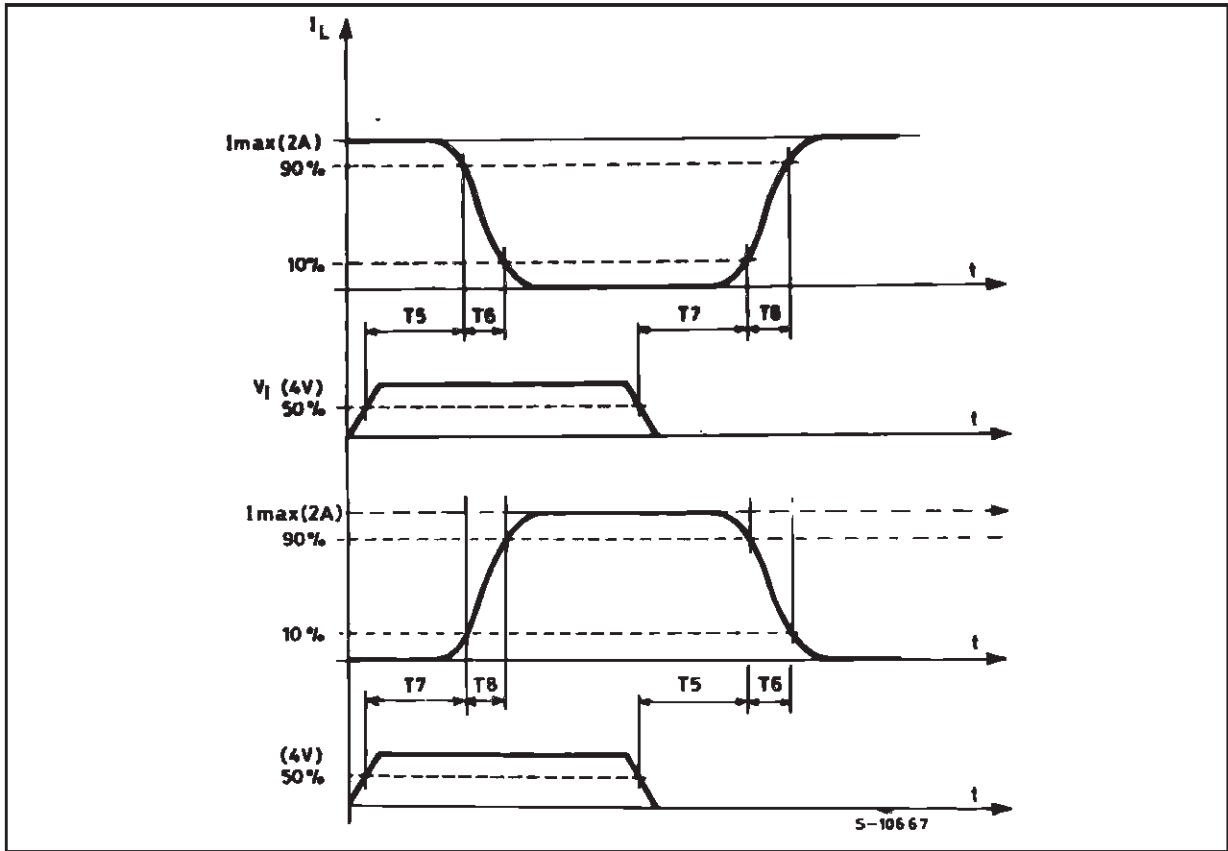
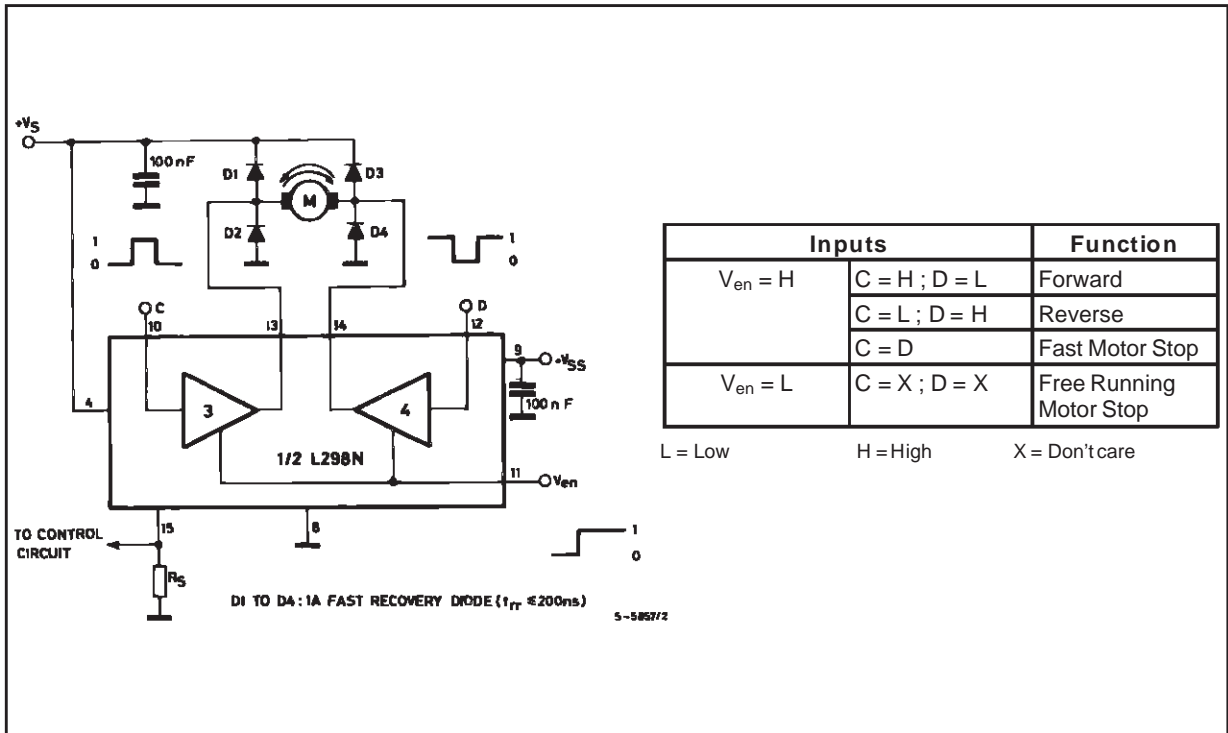
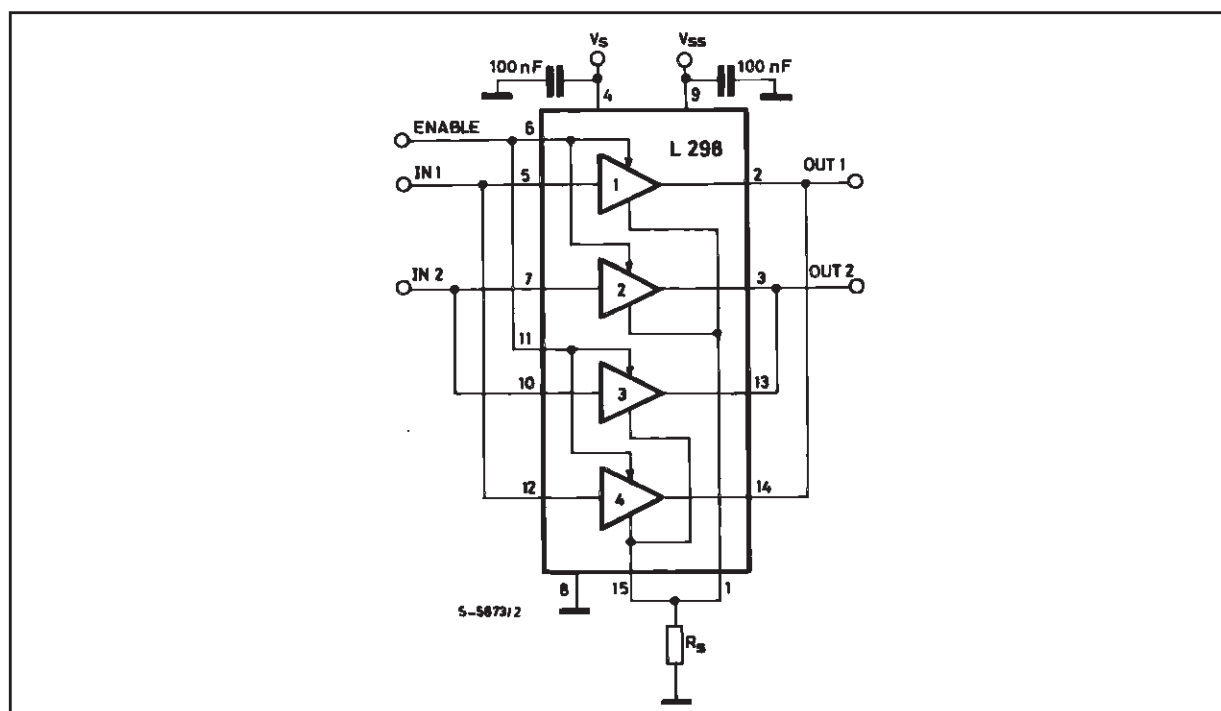


Figure 6 : Bidirectional DC Motor Control.



**Figure 7** : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



## APPLICATION INFORMATION (Refer to the block diagram)

### 1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differential mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output: an external resistor ( $R_{SA}$ ;  $R_{SB}$ ) allows to detect the intensity of this current.

### 1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are  $In_1$ ;  $In_2$ ;  $EnA$  and  $In_3$ ;  $In_4$ ;  $EnB$ . The  $In$  inputs set the bridge state when The  $En$  input is high; a low state of the  $En$  input inhibits the bridge. All the inputs are TTL compatible.

### 2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both  $V_s$  and  $V_{ss}$ , to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of  $V_s$  that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

### 3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes D1 to D4 is made by four fast recovery elements ( $t_{tr} \leq 200$  nsec) that must be chosen of a  $V_F$  as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped; Schottky diodes would be preferred.

This solution can drive until 3 Amps In DC operation and until 3.5 Amps of a repetitive peak current.

On Fig 8 it is shown the driving of a two phase bipolar stepper motor ; the needed signals to drive the inputs of the L298 are generated, in this example, from the IC L297.

Fig 9 shows an example of P.C.B. designed for the application of Fig 8.

**Figure 8** : Two Phase Bipolar Stepper Motor Circuit.

This circuit drives bipolar stepper motors with winding currents up to 2 A. The diodes are fast 2 A types.

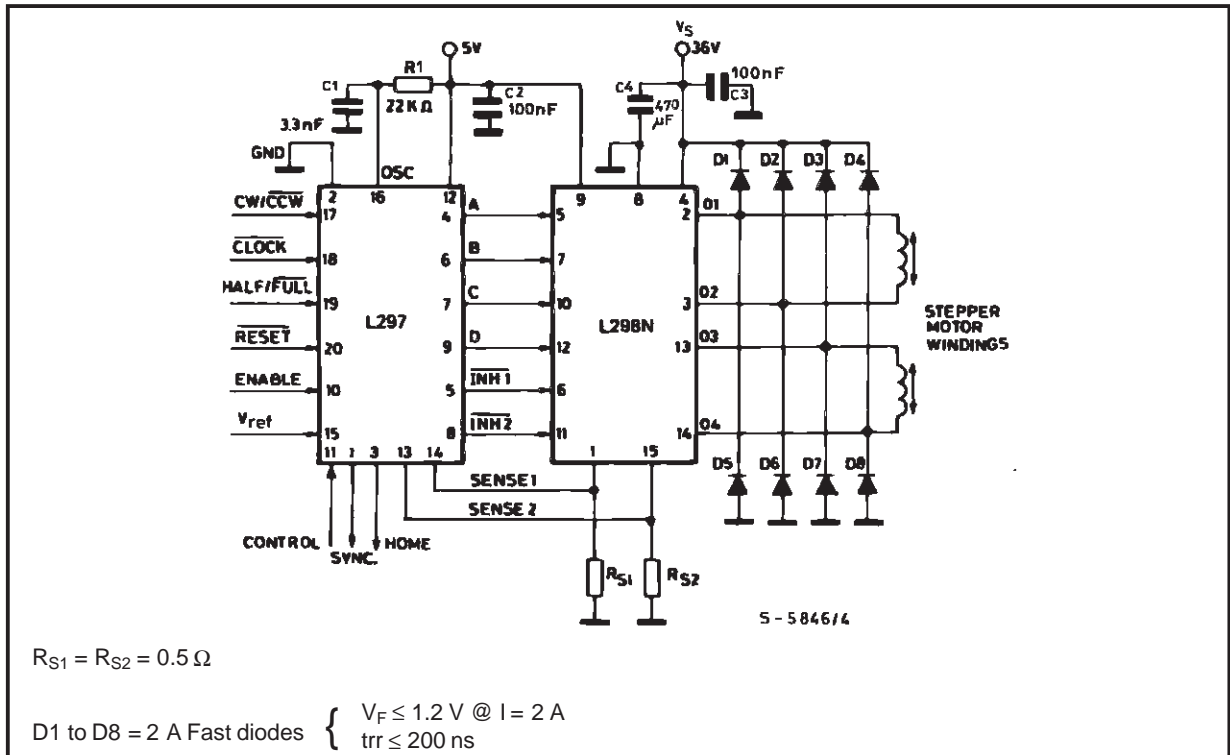


Fig 10 shows a second two phase bipolar stepper motor control circuit where the current is controlled by the I.C. L6506.

Figure 9 : Suggested Printed Circuit Board Layout for the Circuit of fig. 8 (1:1 scale).

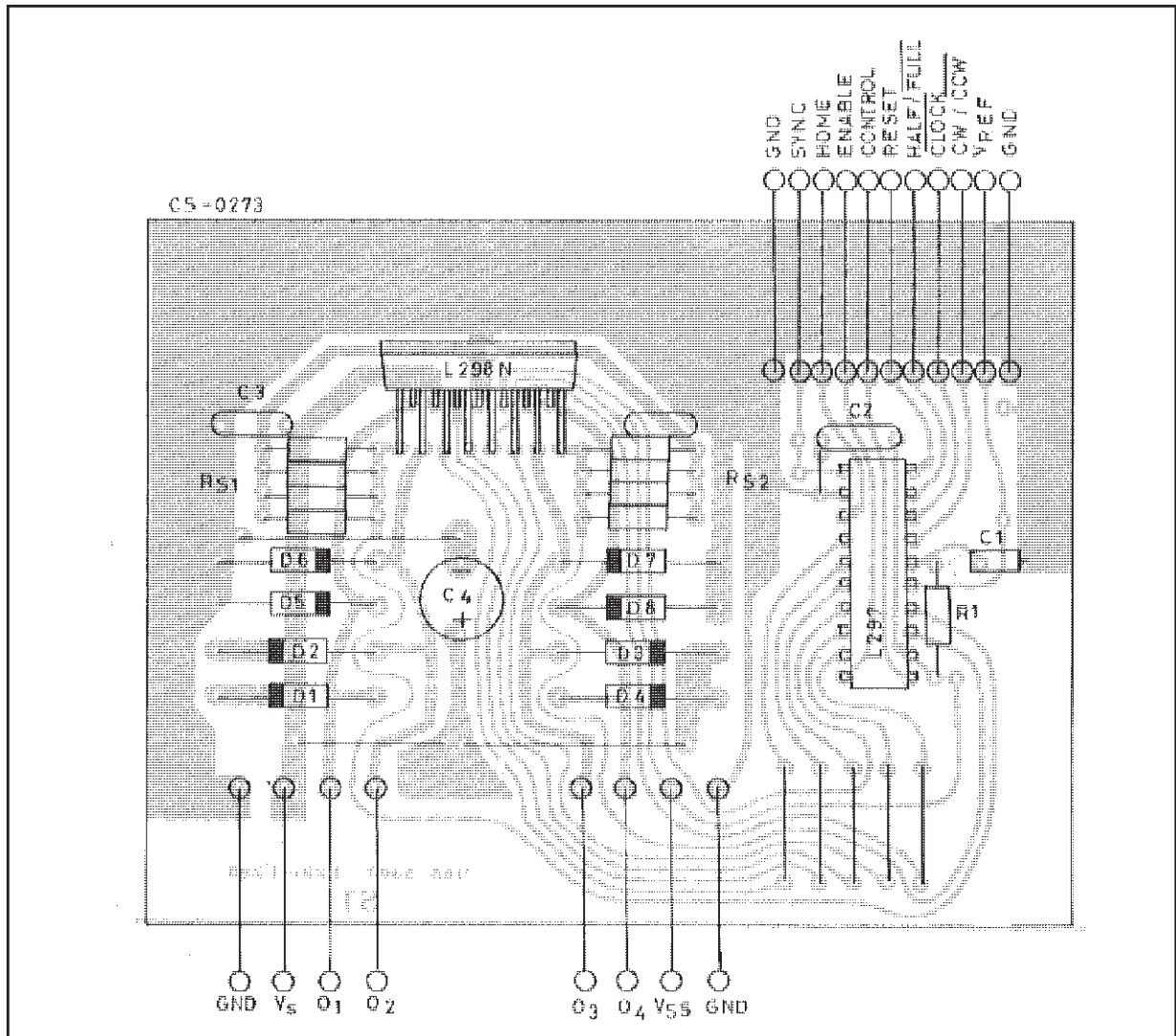
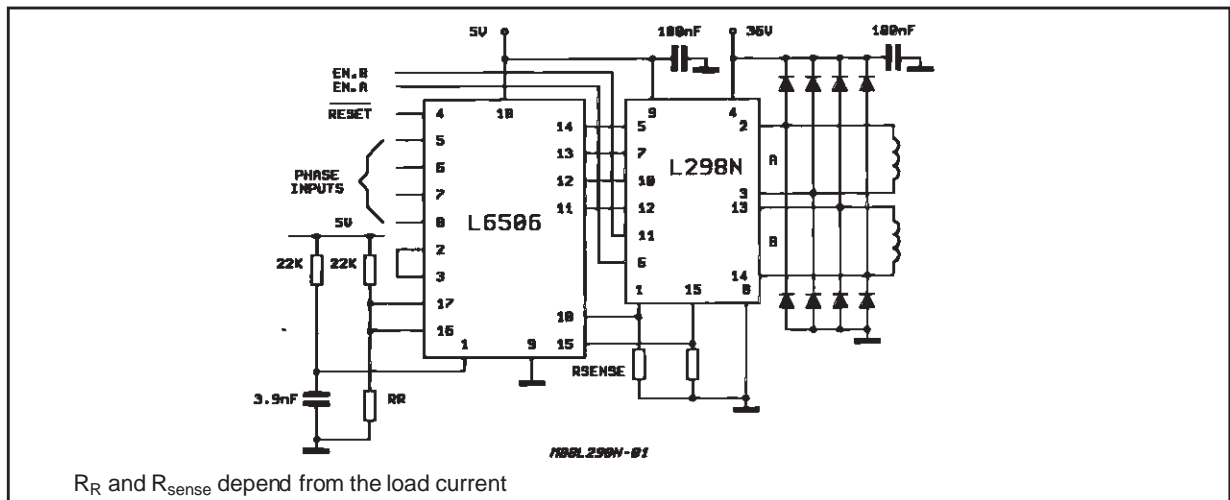


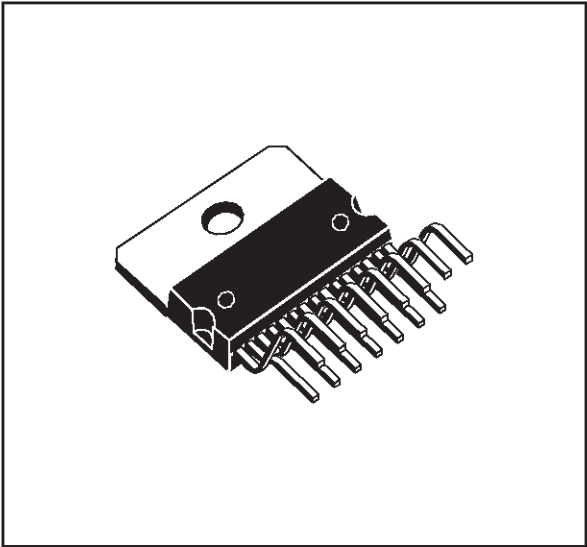
Figure 10 : Two Phase Bipolar Stepper Motor Control Circuit by Using the Current Controller L6506.



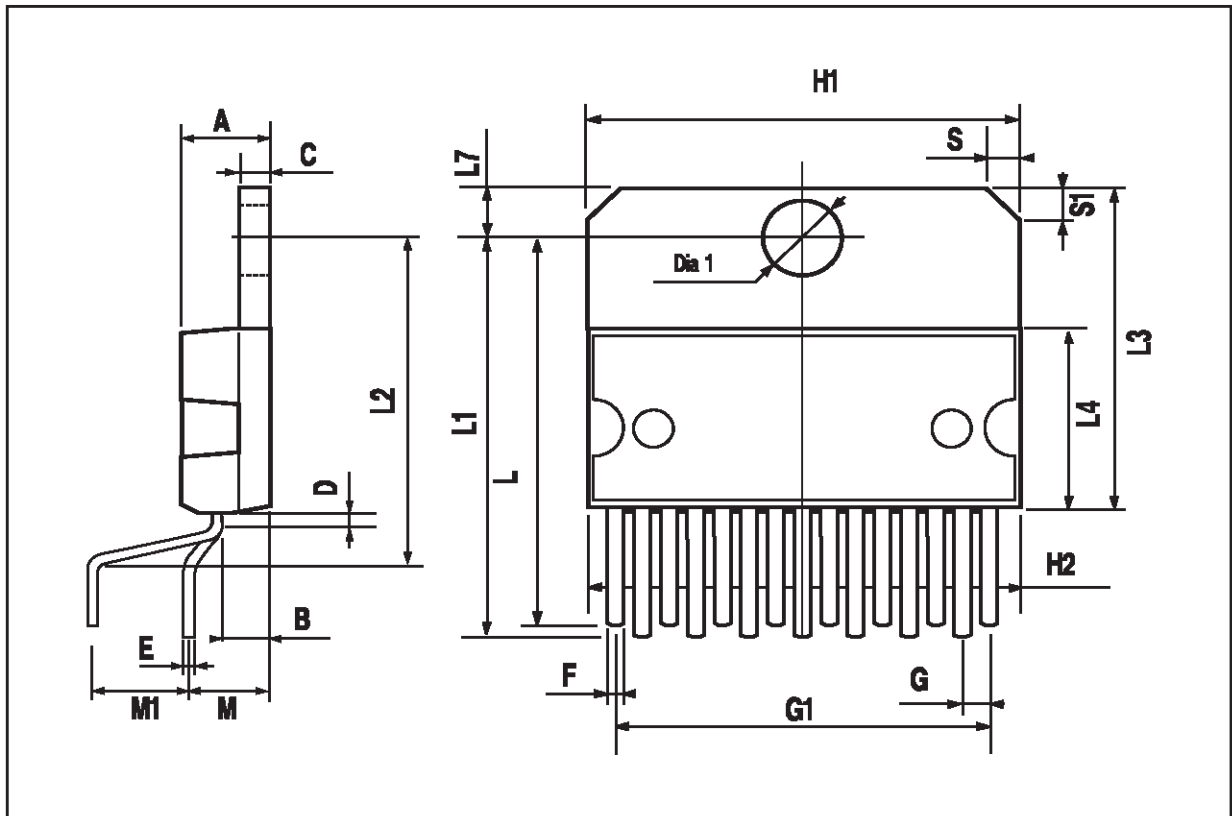


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
D		1			0.039	
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.02	1.27	1.52	0.040	0.050	0.060
G1	17.53	17.78	18.03	0.690	0.700	0.710
H1	19.6			0.772		
H2			20.2			0.795
L	21.9	22.2	22.5	0.862	0.874	0.886
L1	21.7	22.1	22.5	0.854	0.870	0.886
L2	17.65		18.1	0.695		0.713
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L7	2.65		2.9	0.104		0.114
M	4.25	4.55	4.85	0.167	0.179	0.191
M1	4.63	5.08	5.53	0.182	0.200	0.218
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

**OUTLINE AND MECHANICAL DATA**

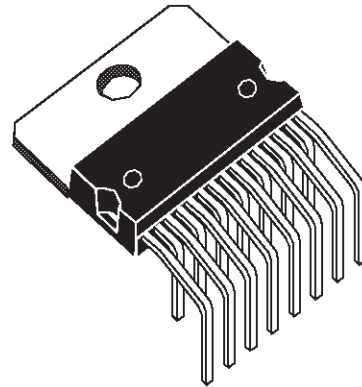


**Multiwatt15 V**

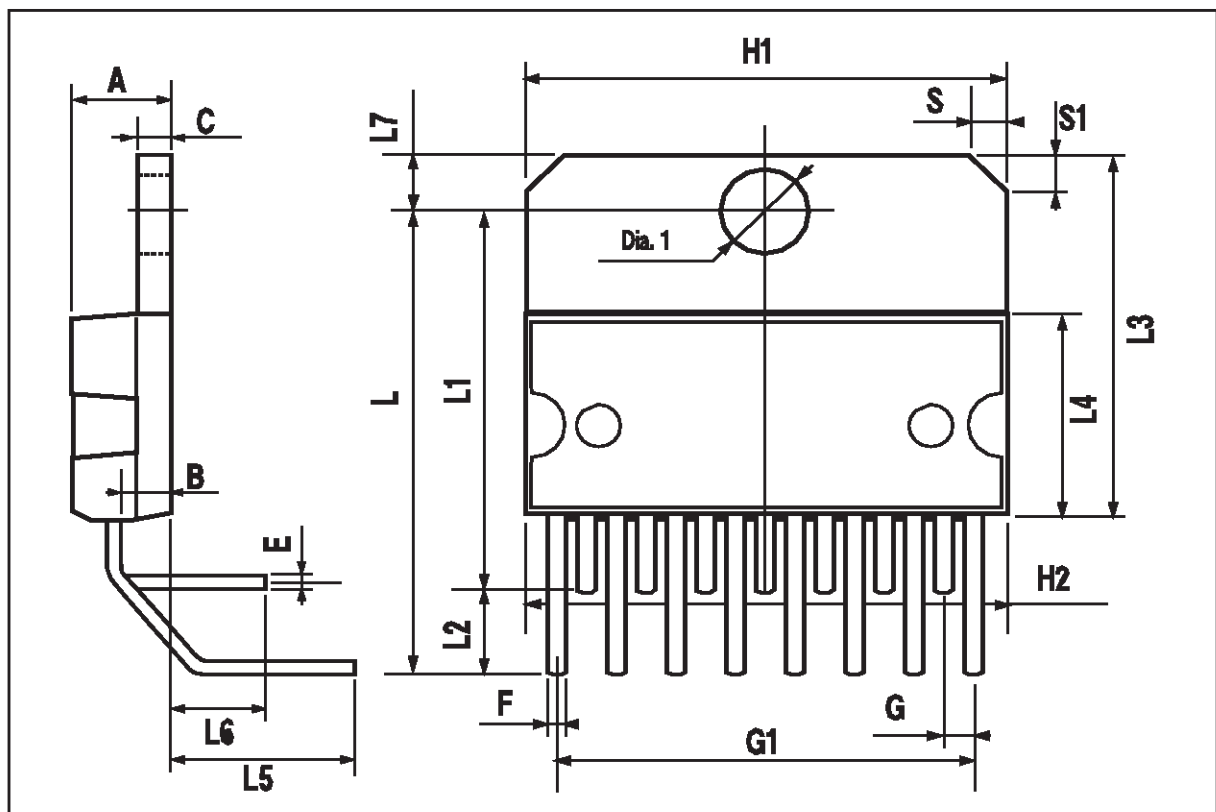


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.14	1.27	1.4	0.045	0.050	0.055
G1	17.57	17.78	17.91	0.692	0.700	0.705
H1	19.6			0.772		
H2			20.2			0.795
L		20.57			0.810	
L1		18.03			0.710	
L2		2.54			0.100	
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L5		5.28			0.208	
L6		2.38			0.094	
L7	2.65		2.9	0.104		0.114
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

## OUTLINE AND MECHANICAL DATA



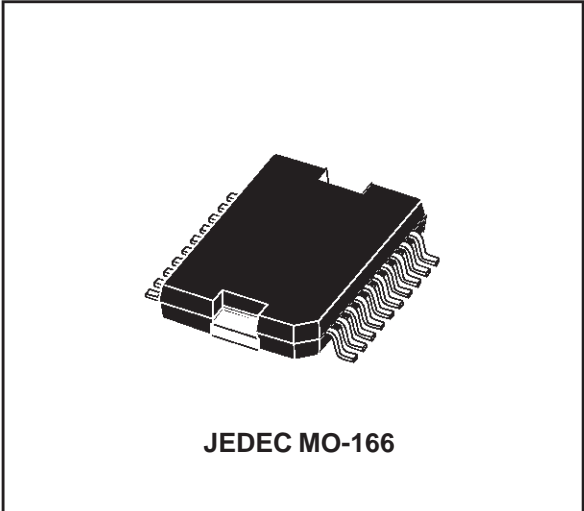
**Multiwatt15 H**



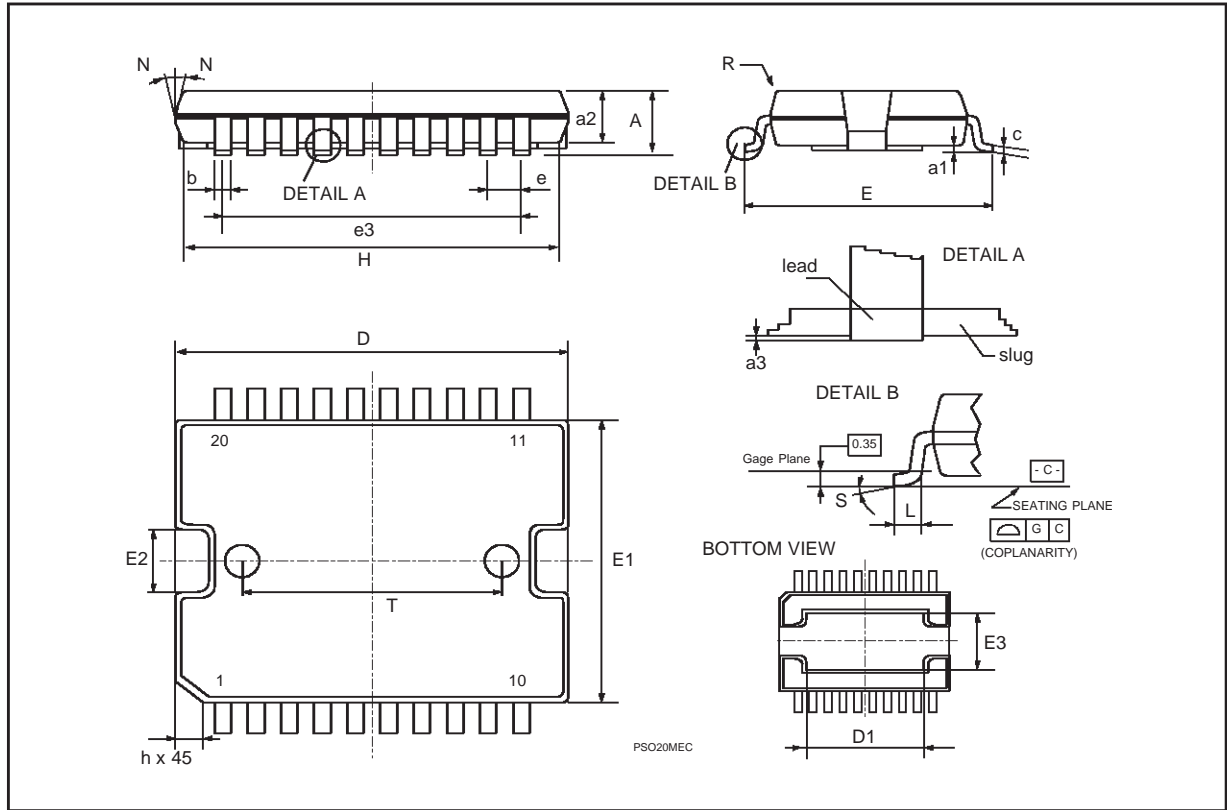
DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			3.6			0.142
a1	0.1		0.3	0.004		0.012
a2			3.3			0.130
a3	0		0.1	0.000		0.004
b	0.4		0.53	0.016		0.021
c	0.23		0.32	0.009		0.013
D (1)	15.8		16	0.622		0.630
D1	9.4		9.8	0.370		0.386
E	13.9		14.5	0.547		0.570
e		1.27			0.050	
e3		11.43			0.450	
E1 (1)	10.9		11.1	0.429		0.437
E2			2.9			0.114
E3	5.8		6.2	0.228		0.244
G	0		0.1	0.000		0.004
H	15.5		15.9	0.610		0.626
h			1.1			0.043
L	0.8		1.1	0.031		0.043
N	10° (max.)					
S	8° (max.)					
T		10			0.394	

(1) "D and F" do not include mold flash or protrusions.  
 - Mold flash or protrusions shall not exceed 0.15 mm (0.006").  
 - Critical dimensions: "E", "G" and "a3"

**OUTLINE AND MECHANICAL DATA**



**PowerSO20**



Information furnished is believed to be accurate and reliable. However, STMicroelectronics assumes no responsibility for the consequences of use of such information nor for any infringement of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of STMicroelectronics. Specification mentioned in this publication are subject to change without notice. This publication supersedes and replaces all information previously supplied. STMicroelectronics products are not authorized for use as critical components in life support devices or systems without express written approval of STMicroelectronics.

The ST logo is a registered trademark of STMicroelectronics  
© 2000 STMicroelectronics – Printed in Italy – All Rights Reserved  
STMicroelectronics GROUP OF COMPANIES

Australia - Brazil - China - Finland - France - Germany - Hong Kong - India - Italy - Japan - Malaysia - Malta - Morocco -  
Singapore - Spain - Sweden - Switzerland - United Kingdom - U.S.A.

<http://www.st.com>

This datasheet has been download from:

[www.datasheetcatalog.com](http://www.datasheetcatalog.com)

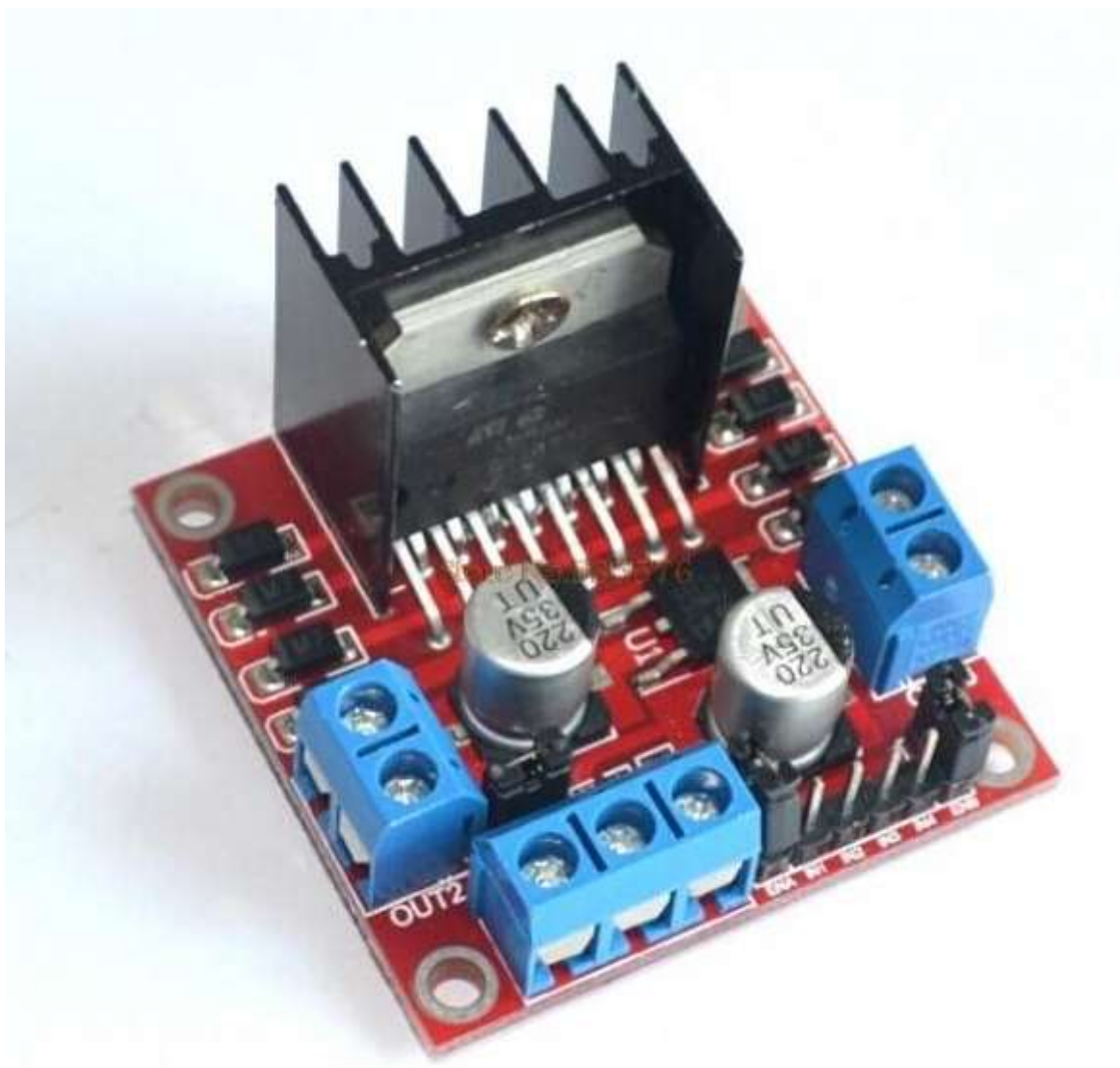
Datasheets for electronics components.

# Tutorial - L298N Dual Motor Controller Module 2A and Arduino

In this tutorial we'll explain how to use our L298N H-bridge Dual Motor Controller Module 2A with Arduino. This allows you to control the speed and direction of two DC motors, or control one bipolar stepper motor with ease. The L298N H-bridge module can be used with motors that have a voltage of between 5 and 35V DC.

There is also an onboard 5V regulator, so if your supply voltage is up to 12V you can also source 5V from the board.

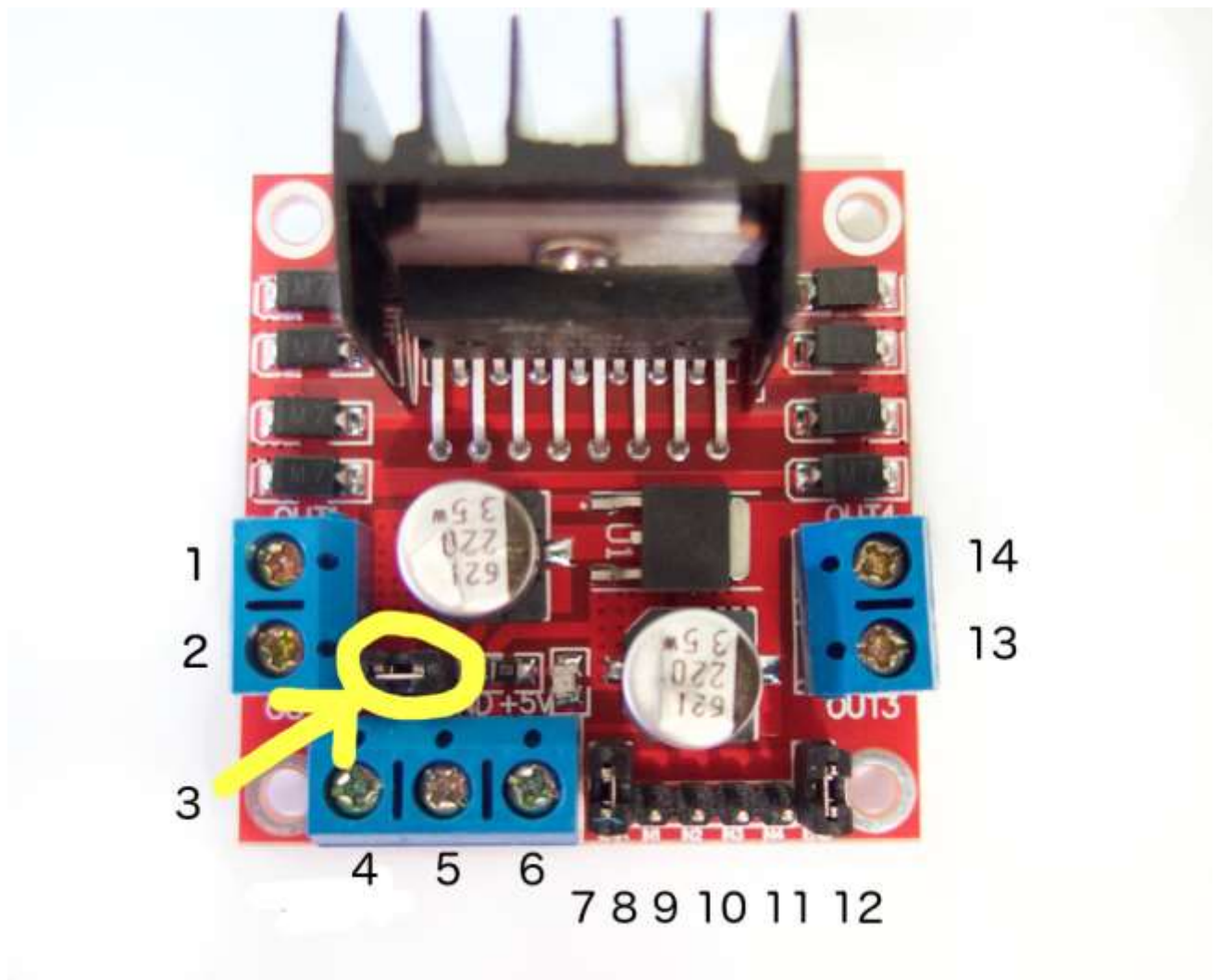
*So let's get started!*



First we'll run through the connections, then explain how to control DC motors then a stepper motor.

## Module pinouts

Consider the following image - match the numbers against the list below the image:



1. DC motor 1 "+" or stepper motor A+
2. DC motor 1 "-" or stepper motor A-
3. 12V jumper - remove this if using a supply voltage greater than 12V DC. This enables power to the onboard 5V regulator
4. Connect your motor supply voltage here, maximum of 35V DC. Remove 12V jumper if >12V DC
5. GND
6. 5V output if 12V jumper in place, ideal for powering your Arduino (etc)
7. DC motor 1 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.
8. IN1
9. IN2
10. IN3
11. IN4
12. DC motor 2 enable jumper. Leave this in place when using a stepper motor. Connect to PWM output for DC motor speed control.

13. DC motor 2 "+" or stepper motor B+
14. DC motor 2 "-" or stepper motor B-

## Controlling DC Motors

To control one or two DC motors is quite easy. First connect each motor to the A and B connections on the L298N module. If you're using two motors for a robot (etc) ensure that the polarity of the motors is the same on both inputs. Otherwise you may need to swap them over when you set both motors to forward and one goes backwards!

Next, connect your power supply - the positive to pin 4 on the module and negative/GND to pin 5. If your supply is up to 12V you can leave in the 12V jumper (point 3 in the image above) and 5V will be available from pin 6 on the module. This can be fed to your Arduino's 5V pin to power it from the motors' power supply. Don't forget to connect Arduino GND to pin 5 on the module as well to complete the circuit.

Now you will need six digital output pins on your Arduino, two of which need to be PWM (pulse-width modulation) pins. PWM pins are denoted by the tilde ("~") next to the pin number, for example:



Finally, connect the Arduino digital output pins to the driver module. In our example we have two DC motors, so digital pins D9, D8, D7 and D6 will be connected to pins IN1, IN2, IN3 and IN4 respectively. Then connect D10 to module pin 7 (remove the jumper first) and D5 to module pin 12 (again, remove the jumper).

The motor direction is controlled by sending a HIGH or LOW signal to the drive for each motor (or channel). For example for motor one, a HIGH to IN1 and a LOW to IN2 will cause it to turn in one direction, and a LOW and HIGH will cause it to turn in the other direction.

However the motors will not turn until a HIGH is set to the enable pin (7 for motor one, 12 for motor two). And they can be turned off with a LOW to the same pin(s). However if you need to control the speed of the motors, the PWM signal from the digital pin connected to the enable pin can take care of it.

This is what we've done with the DC motor demonstration sketch. Two DC motors and an Arduino Uno are connected as described above, along with an external power supply. Then enter and upload the following sketch:

```
// connect motor controller pins to Arduino digital pins
// motor one
int enA = 10;
int in1 = 9;
int in2 = 8;
// motor two
int enB = 5;
int in3 = 7;
int in4 = 6;
```



```

void setup()
{
  // set all the motor control pins to outputs
  pinMode(enA, OUTPUT);
  pinMode(enB, OUTPUT);
  pinMode(in1, OUTPUT);
  pinMode(in2, OUTPUT);
  pinMode(in3, OUTPUT);
  pinMode(in4, OUTPUT);
}
void demoOne()
{
  // this function will run the motors in both directions at a fixed speed
  // turn on motor A
  digitalWrite(in1, HIGH);
  digitalWrite(in2, LOW);
  // set speed to 200 out of possible range 0~255
  analogWrite(enA, 200);
  // turn on motor B
  digitalWrite(in3, HIGH);
  digitalWrite(in4, LOW);
  // set speed to 200 out of possible range 0~255
  analogWrite(enB, 200);
  delay(2000);
  // now change motor directions
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  delay(2000);
  // now turn off motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, LOW);
  digitalWrite(in3, LOW);
  digitalWrite(in4, LOW);
}
void demoTwo()
{
  // this function will run the motors across the range of possible speeds
  // note that maximum speed is determined by the motor itself and the
operating voltage
  // the PWM values sent by analogWrite() are fractions of the maximum
speed possible
  // by your hardware
  // turn on motors
  digitalWrite(in1, LOW);
  digitalWrite(in2, HIGH);
  digitalWrite(in3, LOW);
  digitalWrite(in4, HIGH);
  // accelerate from zero to maximum speed
  for (int i = 0; i < 256; i++)
  {
    analogWrite(enA, i);
    analogWrite(enB, i);
    delay(20);
  }
  // decelerate from maximum speed to zero
  for (int i = 255; i >= 0; --i)
  {
    analogWrite(enA, i);
    analogWrite(enB, i);
  }
}

```

```
    delay(20);
}
// now turn off motors
digitalWrite(in1, LOW);
digitalWrite(in2, LOW);
digitalWrite(in3, LOW);
digitalWrite(in4, LOW);
}
void loop()
{
    demoOne();
    delay(1000);
    demoTwo();
    delay(1000);
}
```

So what's happening in that sketch? In the function *demoOne()* we turn the motors on and run them at a PWM value of 200. This is not a speed value, instead power is applied for 200/255 of an amount of time at once.

Then after a moment the motors operate in the reverse direction (see how we changed the HIGHS and LOWs in the *digitalWrite()* functions?).

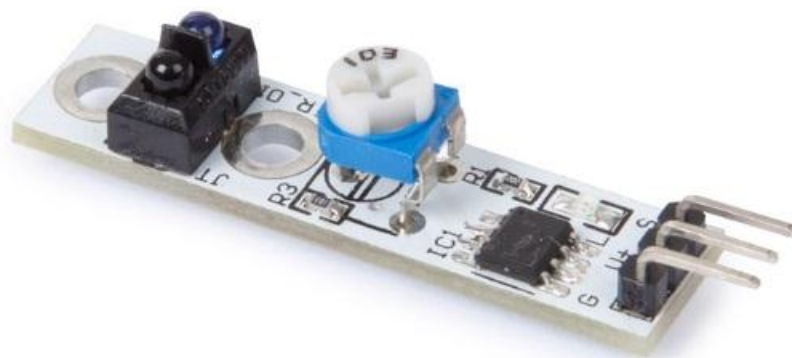
To get an idea of the range of speed possible of your hardware, we run through the entire PWM range in the function *demoTwo()* which turns the motors on and then runs through PWM values zero to 255 and back to zero with the two *for* loops.

Finally this is demonstrated in the following video - using a well-worn tank chassis with two DC motors:

## VMA326

---

### LINE TRACKING SENSOR TCRT5000 MODULE



USER MANUAL



# USER MANUAL

## 1. Introduction

To all residents of the European Union

### Important environmental information about this product



This symbol on the device or the package indicates that disposal of the device after its lifecycle could harm the environment. Do not dispose of the unit (or batteries) as unsorted municipal waste; it should be taken to a specialized company for recycling. This device should be returned to your distributor or to a local recycling service. Respect the local environmental rules.

**If in doubt, contact your local waste disposal authorities.**

Thank you for choosing Velleman®! Please read the manual thoroughly before bringing this device into service. If the device was damaged in transit, do not install or use it and contact your dealer.

## 2. Safety Instructions



- This device can be used by children aged from 8 years and above, and persons with reduced physical, sensory or mental capabilities or lack of experience and knowledge if they have been given supervision or instruction concerning the use of the device in a safe way and understand the hazards involved. Children shall not play with the device. Cleaning and user maintenance shall not be made by children without supervision.



- Indoor use only.  
Keep away from rain, moisture, splashing and dripping liquids.

## 3. General Guidelines



- Refer to the Velleman® Service and Quality Warranty on the last pages of this manual.
- Familiarise yourself with the functions of the device before actually using it.
- All modifications of the device are forbidden for safety reasons. Damage caused by user modifications to the device is not covered by the warranty.
- Only use the device for its intended purpose. Using the device in an unauthorised way will void the warranty.
- Damage caused by disregard of certain guidelines in this manual is not covered by the warranty and the dealer will not accept responsibility for any ensuing defects or problems.
- Nor Velleman nv nor its dealers can be held responsible for any damage (extraordinary, incidental or indirect) – of any nature (financial, physical...) arising from the possession, use or failure of this product.
- Due to constant product improvements, the actual product appearance might differ from the shown images.
- Product images are for illustrative purposes only.
- Do not switch the device on immediately after it has been exposed to changes in temperature. Protect the device against damage by leaving it switched off until it has reached room temperature.
- Keep this manual for future reference.

## 4. What is Arduino®

Arduino® is an open-source prototyping platform based in easy-to-use hardware and software. Arduino® boards are able to read inputs – light-on sensor, a finger on a button or a Twitter message – and turn it into an output – activating of a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so, you use the Arduino programming language (based on Wiring) and the Arduino® software IDE (based on Processing).

Surf to [www.arduino.cc](http://www.arduino.cc) and [www.arduino.org](http://www.arduino.org) for more information.

## 5. Overview

### VMA326

The line tracking sensor based on TCRT5000 is a type of infrared reflectance sensor, and is commonly used in line following robots, mounted at the bottom of the robot chassis. The line tracking sensor works by detecting reflected light coming from its own infrared LED and by measuring the amount of reflected infrared light, it can detect transitions from light to dark (lines) or even objects directly in front of it.

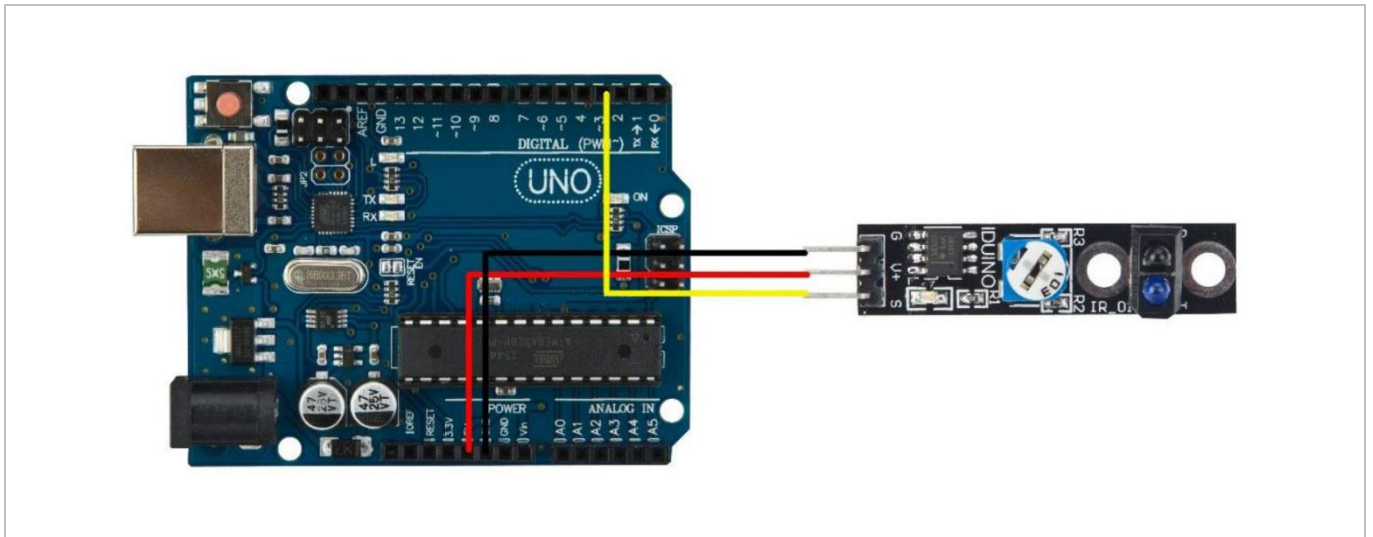
sensor adopts TCRT5000, high sensitivity  
 sensitivity adjustable by potentiometer  
 Working voltage 3.3 V to 5 V  
 operating current : 20 mA  
 digital switch output (0 and 1)  
 two fixed bolt holes for convenient installation  
 small board PCB size: 42 x 10.5 mm  
 power indicator light (red) and digital switch output indicator light (green)  
 detection reflection distance: 1-25 mm

## 6. Pin Layout

S	digital output (black = low, white = high)
V+	5 VDC power
G	ground

## 7. Example

The example shows that when the sensor detects a black area, the S pin's output is a low TTL signal. The LED13 switches off while the light L on this module switches on. On the contrary, LED13 switches on.



```

// CODE BEGIN
int Led=13; // variable Led = connected to digital pin 13 (Which is on your VMA100 connected to a LED as well)
int buttonpin=3; // Variable buttonpin = connected to digital line 3, this is where the VMA326 output has to be connected for this test
int val;
void setup()
{ pinMode(Led,OUTPUT); // declare Led (digital 13) as output
  pinMode(buttonpin,INPUT); // declare buttonpin (digital 3) as input
}
void loop()
{
  val=digitalRead(buttonpin); // read the value of buttonpin (digital 3)
  if(val==HIGH) // if this value is high ...
  { digitalWrite(Led,HIGH); } // then switch Led (digital 13) to High
  else { digitalWrite(Led,LOW); } // else .. switch to low
}
// CODE END

```

## 8. More Information

Please refer to the VMA326 product page on [www.velleman.eu](http://www.velleman.eu) for more information.

**Use this device with original accessories only. Velleman nv cannot be held responsible in the event of damage or injury resulting from (incorrect) use of this device. For more info concerning this product and the latest version of this manual, please visit our website [www.velleman.eu](http://www.velleman.eu). The information in this manual is subject to change without prior notice.**

### © COPYRIGHT NOTICE

**The copyright to this manual is owned by Velleman nv. All worldwide rights reserved.** No part of this manual may be copied, reproduced, translated or reduced to any electronic medium or otherwise without the prior written consent of the copyright holder.

# Velleman® Service and Quality Warranty

Since its foundation in 1972, Velleman® acquired extensive experience in the electronics world and currently distributes its products in over 85 countries.

All our products fulfil strict quality requirements and legal stipulations in the EU. In order to ensure the quality, our products regularly go through an extra quality check, both by an internal quality department and by specialized external organisations. If, all precautionary measures notwithstanding, problems should occur, please make appeal to our warranty (see guarantee conditions).

## General Warranty Conditions Concerning Consumer Products (for EU):

- All consumer products are subject to a 24-month warranty on production flaws and defective material as from the original date of purchase.
- Velleman® can decide to replace an article with an equivalent article, or to refund the retail value totally or partially when the complaint is valid and a free repair or replacement of the article is impossible, or if the expenses are out of proportion.

You will be delivered a replacing article or a refund at the value of 100% of the purchase price in case of a flaw occurred in the first year after the date of purchase and delivery, or a replacing article at 50% of the purchase price or a refund at the value of 50% of the retail value in case of a flaw occurred in the second year after the date of purchase and delivery.

### • Not covered by warranty:

- all direct or indirect damage caused after delivery to the article (e.g. by oxidation, shocks, falls, dust, dirt, humidity...), and by the article, as well as its contents (e.g. data loss), compensation for loss of profits;
- consumable goods, parts or accessories that are subject to an aging process during normal use, such as batteries (rechargeable, non-rechargeable, built-in or replaceable), lamps, rubber parts, drive belts... (unlimited list);
- flaws resulting from fire, water damage, lightning, accident, natural disaster, etc....;
- flaws caused deliberately, negligently or resulting from improper handling, negligent maintenance, abusive use or use contrary to the manufacturer's instructions;
- damage caused by a commercial, professional or collective use of the article (the warranty validity will be reduced to six (6) months when the article is used professionally);
- damage resulting from an inappropriate packing and shipping of the article;
- all damage caused by modification, repair or alteration performed by a third party without written permission by Velleman®.
- Articles to be repaired must be delivered to your Velleman® dealer, solidly packed (preferably in the original packaging), and be completed with the original receipt of purchase and a clear flaw description.
- Hint: In order to save on cost and time, please reread the manual and check if the flaw is caused by obvious causes prior to presenting the article for repair. Note that returning a non-defective article can also involve handling costs.
- Repairs occurring after warranty expiration are subject to shipping costs.
- The above conditions are without prejudice to all commercial warranties.

**The above enumeration is subject to modification according to the article (see article's manual).**



# MFRC522

## Contactless reader IC

Rev. 3.6 — 14 December 2011  
112136

Product data sheet  
COMPANY PUBLIC

## 1. Introduction

---

This document describes the functionality and electrical specifications of the contactless reader/writer MFRC522.

**Remark:** The MFRC522 supports all variants of the MIFARE Mini, MIFARE 1K, MIFARE 4K, MIFARE Ultralight, MIFARE DESFire EV1 and MIFARE Plus RF identification protocols. To aid readability throughout this data sheet, the MIFARE Mini, MIFARE 1K, MIFARE 4K, MIFARE Ultralight, MIFARE DESFire EV1 and MIFARE Plus products and protocols have the generic name MIFARE.

## 2. General description

---

The MFRC522 is a highly integrated reader/writer IC for contactless communication at 13.56 MHz. The MFRC522 reader supports ISO/IEC 14443 A/MIFARE mode.

The MFRC522's internal transmitter is able to drive a reader/writer antenna designed to communicate with ISO/IEC 14443 A/MIFARE cards and transponders without additional active circuitry. The receiver module provides a robust and efficient implementation for demodulating and decoding signals from ISO/IEC 14443 A/MIFARE compatible cards and transponders. The digital module manages the complete ISO/IEC 14443 A framing and error detection (parity and CRC) functionality.

The MFRC522 supports MF1xxS20, MF1xxS70 and MF1xxS50 products. The MFRC522 supports contactless communication and uses MIFARE higher transfer speeds up to 848 kBd in both directions.

The following host interfaces are provided:

- Serial Peripheral Interface (SPI)
- Serial UART (similar to RS232 with voltage levels dependant on pin voltage supply)
- I<sup>2</sup>C-bus interface

### 2.1 Differences between version 1.0 and 2.0

The MFRC522 is available in two versions:

- MFRC52201HN1, hereafter referred to version 1.0 and
- MFRC52202HN1, hereafter referred to version 2.0.

The MFRC522 version 2.0 is fully compatible to version 1.0 and offers in addition the following features and improvements:





- Increased stability of the reader IC in rough conditions
- An additional timer prescaler, see [Section 8.5](#).
- A corrected CRC handling when RX Multiple is set to 1

This data sheet version covers both versions of the MFRC522 and describes the differences between the versions if applicable.

### 3. Features and benefits

- Highly integrated analog circuitry to demodulate and decode responses
- Buffered output drivers for connecting an antenna with the minimum number of external components
- Supports ISO/IEC 14443 A/MIFARE
- Typical operating distance in Read/Write mode up to 50 mm depending on the antenna size and tuning
- Supports MF1xxS20, MF1xxS70 and MF1xxS50 encryption in Read/Write mode
- Supports ISO/IEC 14443 A higher transfer speed communication up to 848 kBd
- Supports MFIN/MFOUT
- Additional internal power supply to the smart card IC connected via MFIN/MFOUT
- Supported host interfaces
  - ◆ SPI up to 10 Mbit/s
  - ◆ I<sup>2</sup>C-bus interface up to 400 kBd in Fast mode, up to 3400 kBd in High-speed mode
  - ◆ RS232 Serial UART up to 1228.8 kBd, with voltage levels dependant on pin voltage supply
- FIFO buffer handles 64 byte send and receive
- Flexible interrupt modes
- Hard reset with low power function
- Power-down by software mode
- Programmable timer
- Internal oscillator for connection to 27.12 MHz quartz crystal
- 2.5 V to 3.3 V power supply
- CRC coprocessor
- Programmable I/O pins
- Internal self-test

### 4. Quick reference data

**Table 1. Quick reference data**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V <sub>DDA</sub>	analog supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ;	[1][2] 2.5	3.3	3.6	V
V <sub>DDD</sub>	digital supply voltage	V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	2.5	3.3	3.6	V
V <sub>DD(TVDD)</sub>	TVDD supply voltage		2.5	3.3	3.6	V
V <sub>DD(PVDD)</sub>	PVDD supply voltage		[3] 1.6	1.8	3.6	V
V <sub>DD(SVDD)</sub>	SVDD supply voltage	V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	1.6	-	3.6	V

**Table 1. Quick reference data ...continued**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit	
I <sub>pd</sub>	power-down current	V <sub>D<sub>DA</sub></sub> = V <sub>D<sub>DD</sub></sub> = V <sub>D<sub>D(TVDD)</sub></sub> = V <sub>D<sub>D(PVDD)</sub></sub> = 3 V					
		hard power-down; pin NRSTPD set LOW	[4]	-	-	5	μA
		soft power-down; RF level detector on	[4]	-	-	10	μA
I <sub>DD<sub>D</sub></sub>	digital supply current	pin DVDD; V <sub>D<sub>DD</sub></sub> = 3 V	-	6.5	9	mA	
I <sub>DD<sub>A</sub></sub>	analog supply current	pin AVDD; V <sub>D<sub>DA</sub></sub> = 3 V, CommandReg register's RcvOff bit = 0	-	7	10	mA	
		pin AVDD; receiver switched off; V <sub>D<sub>DA</sub></sub> = 3 V, CommandReg register's RcvOff bit = 1	-	3	5	mA	
I <sub>DD(PVDD)</sub>	PVDD supply current	pin PVDD	[5]	-	40	mA	
I <sub>DD(TVDD)</sub>	TVDD supply current	pin TVDD; continuous wave	[6][7][8]	-	60	100	mA
T <sub>amb</sub>	ambient temperature	HVQFN32	-25	-	+85	°C	

- [1] Supply voltages below 3 V reduce the performance in, for example, the achievable operating distance.
- [2] V<sub>D<sub>DA</sub></sub>, V<sub>D<sub>DD</sub></sub> and V<sub>D<sub>D(TVDD)</sub></sub> must always be the same voltage.
- [3] V<sub>D<sub>D(PVDD)</sub></sub> must always be the same or lower voltage than V<sub>D<sub>DD</sub></sub>.
- [4] I<sub>pd</sub> is the total current for all supplies.
- [5] I<sub>DD(PVDD)</sub> depends on the overall load at the digital pins.
- [6] I<sub>DD(TVDD)</sub> depends on V<sub>D<sub>D(TVDD)</sub></sub> and the external circuit connected to pins TX1 and TX2.
- [7] During typical circuit operation, the overall current is below 100 mA.
- [8] Typical value using a complementary driver configuration and an antenna matched to 40 Ω between pins TX1 and TX2 at 13.56 MHz.

## 5. Ordering information

**Table 2. Ordering information**

Type number	Package		
	Name	Description	Version
MFRC52201HN1/TRAYB <sup>[1]</sup>	HVQFN32	plastic thermal enhanced very thin quad flat package; no leads; 32 terminal; body 5 × 5 × 0.85 mm	SOT617-1
MFRC52201HN1/TRAYBM <sup>[2]</sup>	HVQFN32	plastic thermal enhanced very thin quad flat package; no leads; 32 terminal; body 5 × 5 × 0.85 mm	SOT617-1
MFRC52202HN1/TRAYB <sup>[1]</sup>	HVQFN32	plastic thermal enhanced very thin quad flat package; no leads; 32 terminal; body 5 × 5 × 0.85 mm	SOT617-1
MFRC52202HN1/TRAYBM <sup>[2]</sup>	HVQFN32	plastic thermal enhanced very thin quad flat package; no leads; 32 terminal; body 5 × 5 × 0.85 mm	SOT617-1

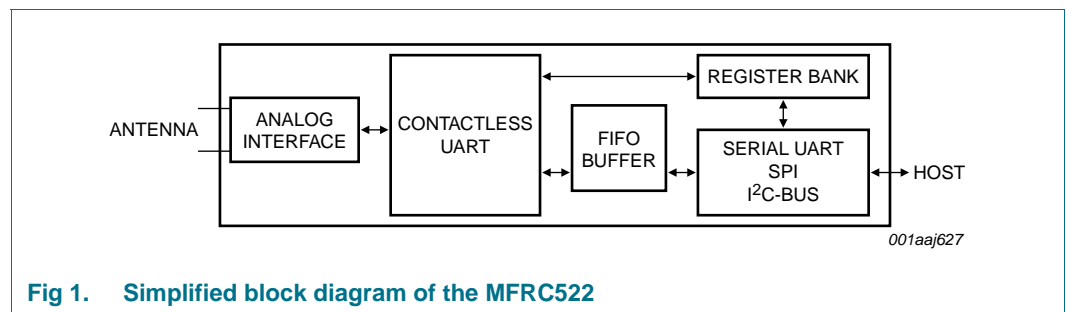
- [1] Delivered in one tray.
- [2] Delivered in five trays.

## 6. Block diagram

The analog interface handles the modulation and demodulation of the analog signals.

The contactless UART manages the protocol requirements for the communication protocols in cooperation with the host. The FIFO buffer ensures fast and convenient data transfer to and from the host and the contactless UART and vice versa.

Various host interfaces are implemented to meet different customer requirements.



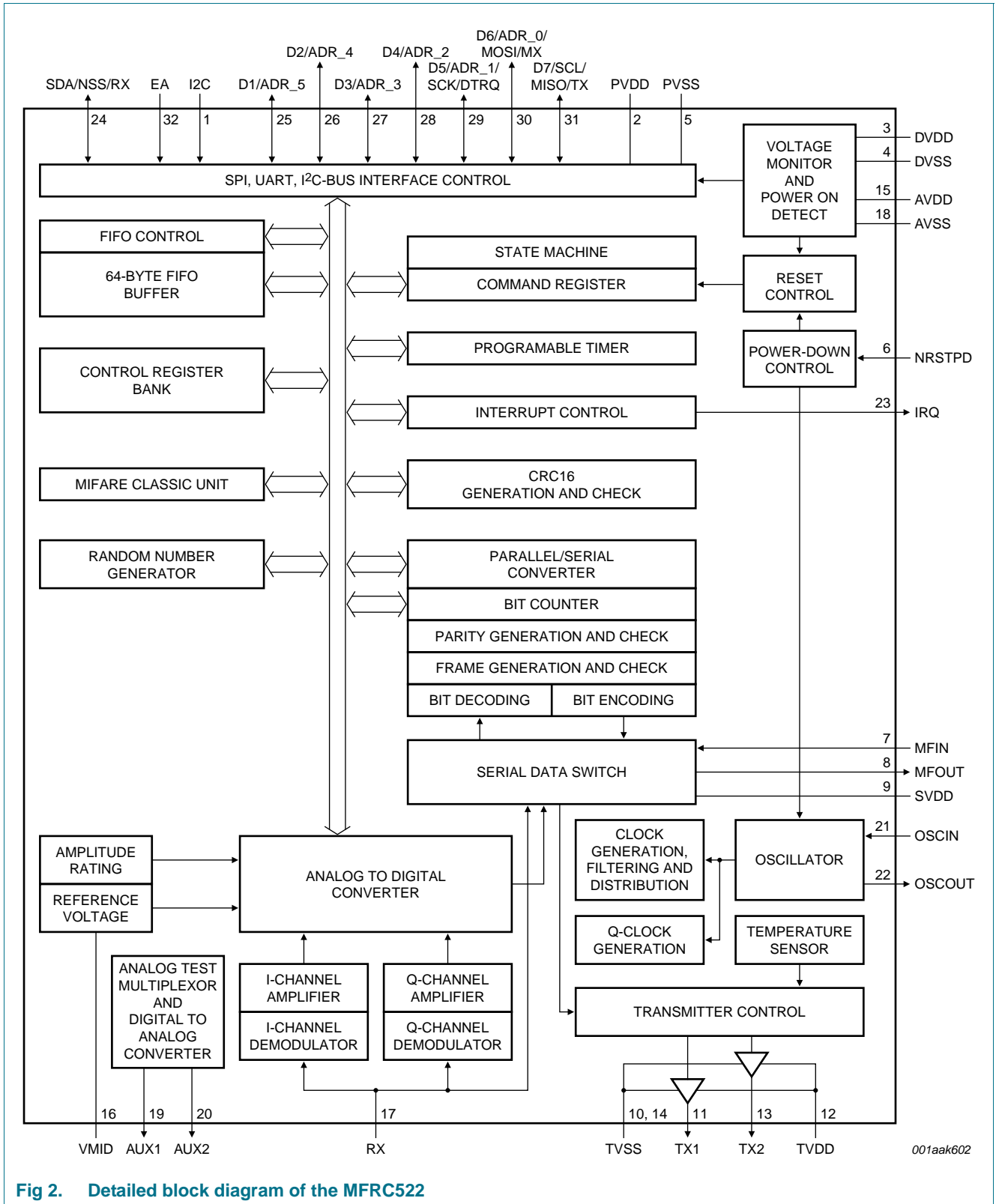
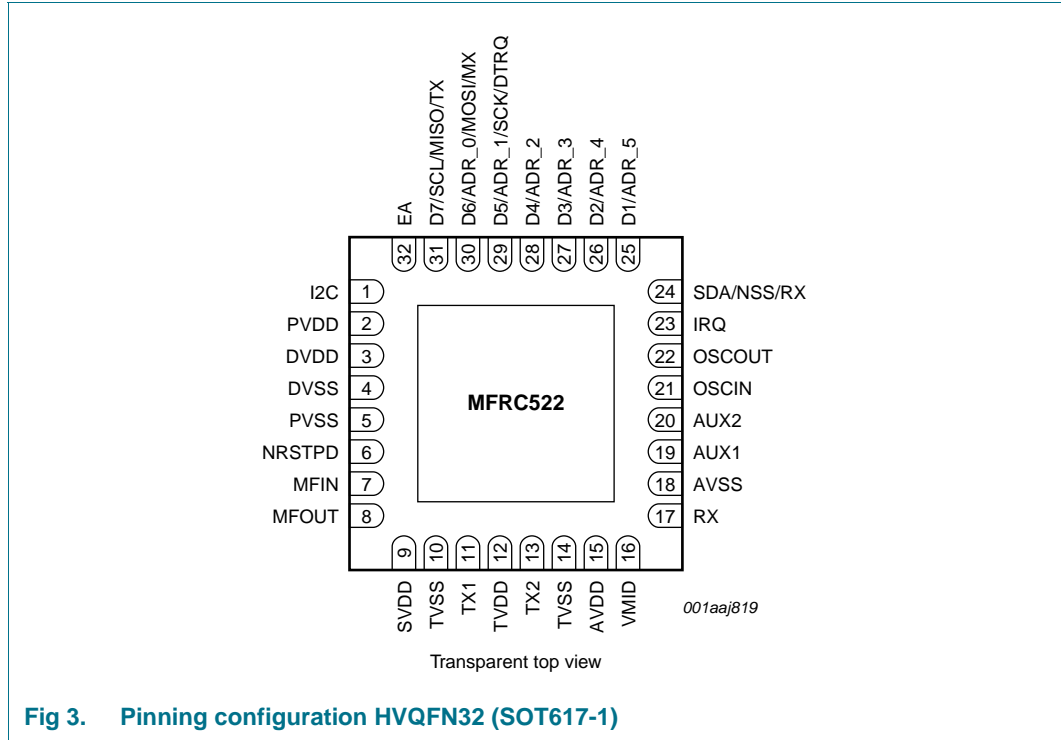


Fig 2. Detailed block diagram of the MFRC522

## 7. Pinning information



**Fig 3. Pinning configuration HVQFN32 (SOT617-1)**

### 7.1 Pin description

**Table 3. Pin description**

Pin	Symbol	Type <sup>[1]</sup>	Description
1	I2C	I	I <sup>2</sup> C-bus enable input <sup>[2]</sup>
2	PVDD	P	pin power supply
3	DVDD	P	digital power supply
4	DVSS	G	digital ground <sup>[3]</sup>
5	PVSS	G	pin power supply ground
6	NRSTPD	I	reset and power-down input: power-down: enabled when LOW; internal current sinks are switched off, the oscillator is inhibited and the input pins are disconnected from the outside world reset: enabled by a positive edge
7	MFIN	I	MIFARE signal input
8	MFOUT	O	MIFARE signal output
9	SVDD	P	MFIN and MFOUT pin power supply
10	TVSS	G	transmitter output stage 1 ground
11	TX1	O	transmitter 1 modulated 13.56 MHz energy carrier output
12	TVDD	P	transmitter power supply: supplies the output stage of transmitters 1 and 2
13	TX2	O	transmitter 2 modulated 13.56 MHz energy carrier output
14	TVSS	G	transmitter output stage 2 ground
15	AVDD	P	analog power supply

Table 3. Pin description ...continued

Pin	Symbol	Type <sup>[1]</sup>	Description
16	VMID	P	internal reference voltage
17	RX	I	RF signal input
18	AVSS	G	analog ground
19	AUX1	O	auxiliary outputs for test purposes
20	AUX2	O	auxiliary outputs for test purposes
21	OSCIN	I	crystal oscillator inverting amplifier input; also the input for an externally generated clock ( $f_{clk} = 27.12$ MHz)
22	OSCOU	O	crystal oscillator inverting amplifier output
23	IRQ	O	interrupt request output: indicates an interrupt event
24	SDA	I/O	I <sup>2</sup> C-bus serial data line input/output <sup>[2]</sup>
	NSS	I	SPI signal input <sup>[2]</sup>
	RX	I	UART address input <sup>[2]</sup>
25	D1	I/O	test port <sup>[2]</sup>
	ADR_5	I/O	I <sup>2</sup> C-bus address 5 input <sup>[2]</sup>
26	D2	I/O	test port
	ADR_4	I	I <sup>2</sup> C-bus address 4 input <sup>[2]</sup>
27	D3	I/O	test port
	ADR_3	I	I <sup>2</sup> C-bus address 3 input <sup>[2]</sup>
28	D4	I/O	test port
	ADR_2	I	I <sup>2</sup> C-bus address 2 input <sup>[2]</sup>
29	D5	I/O	test port
	ADR_1	I	I <sup>2</sup> C-bus address 1 input <sup>[2]</sup>
	SCK	I	SPI serial clock input <sup>[2]</sup>
	DTRQ	O	UART request to send output to microcontroller <sup>[2]</sup>
30	D6	I/O	test port
	ADR_0	I	I <sup>2</sup> C-bus address 0 input <sup>[2]</sup>
	MOSI	I/O	SPI master out, slave in <sup>[2]</sup>
	MX	O	UART output to microcontroller <sup>[2]</sup>
31	D7	I/O	test port
	SCL	I/O	I <sup>2</sup> C-bus clock input/output <sup>[2]</sup>
	MISO	I/O	SPI master in, slave out <sup>[2]</sup>
	TX	O	UART data output to microcontroller <sup>[2]</sup>
32	EA	I	external address input for coding I <sup>2</sup> C-bus address <sup>[2]</sup>

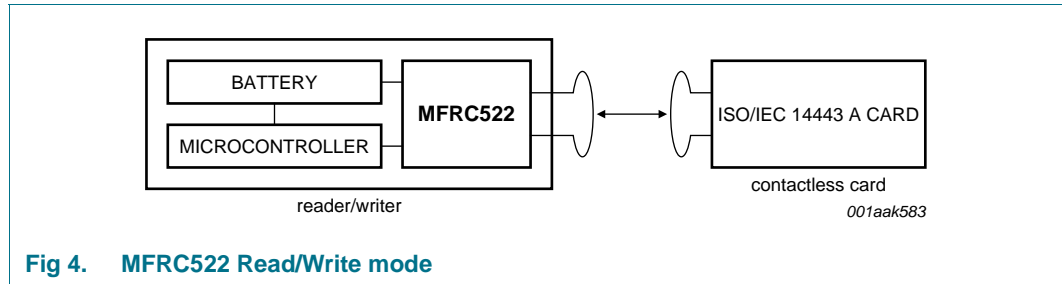
[1] Pin types: I = Input, O = Output, I/O = Input/Output, P = Power and G = Ground.

[2] The pin functionality of these pins is explained in [Section 8.1 "Digital interfaces"](#).

[3] Connection of heatsink pad on package bottom side is not necessary. Optional connection to pin DVSS is possible.

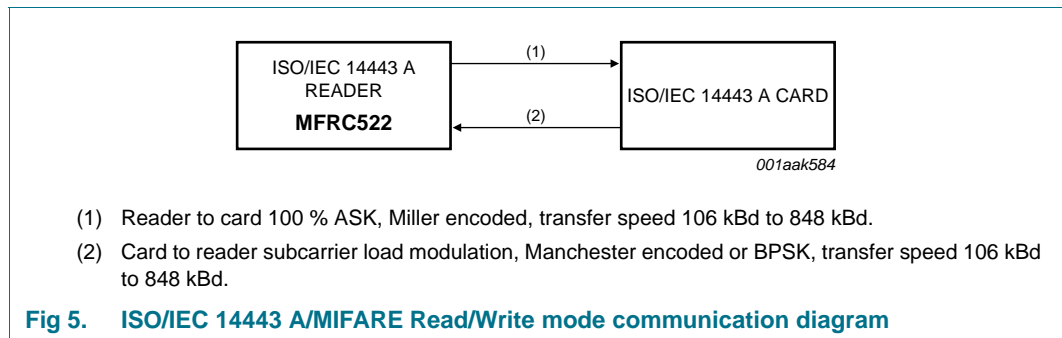
## 8. Functional description

The MFRC522 transmission module supports the Read/Write mode for ISO/IEC 14443 A/MIFARE using various transfer speeds and modulation protocols.



**Fig 4. MFRC522 Read/Write mode**

The physical level communication is shown in [Figure 5](#).



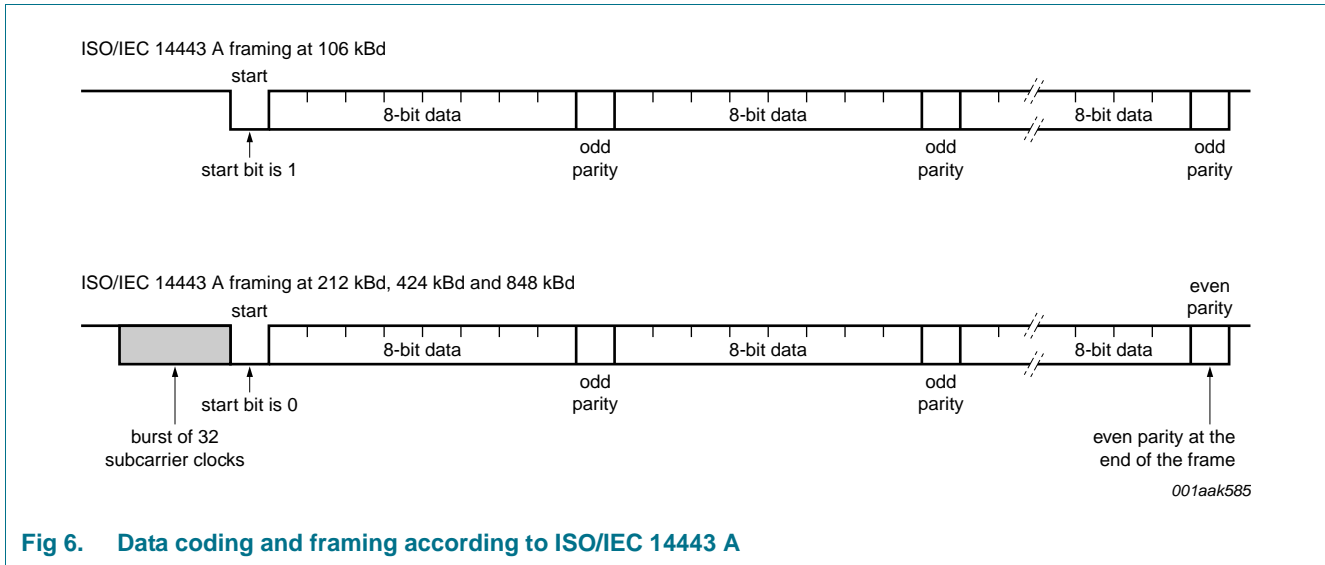
**Fig 5. ISO/IEC 14443 A/MIFARE Read/Write mode communication diagram**

The physical parameters are described in [Table 4](#).

**Table 4. Communication overview for ISO/IEC 14443 A/MIFARE reader/writer**

Communication direction	Signal type	Transfer speed			
		106 kBd	212 kBd	424 kBd	848 kBd
Reader to card (send data from the MFRC522 to a card)	reader side modulation	100 % ASK	100 % ASK	100 % ASK	100 % ASK
	bit encoding	modified Miller encoding	modified Miller encoding	modified Miller encoding	modified Miller encoding
	bit length	128 (13.56 μs)	64 (13.56 μs)	32 (13.56 μs)	16 (13.56 μs)
Card to reader (MFRC522 receives data from a card)	card side modulation	subcarrier load modulation	subcarrier load modulation	subcarrier load modulation	subcarrier load modulation
	subcarrier frequency	13.56 MHz / 16	13.56 MHz / 16	13.56 MHz / 16	13.56 MHz / 16
	bit encoding	Manchester encoding	BPSK	BPSK	BPSK

The MFRC522's contactless UART and dedicated external host must manage the complete ISO/IEC 14443 A/MIFARE protocol. [Figure 6](#) shows the data coding and framing according to ISO/IEC 14443 A/MIFARE.



**Fig 6. Data coding and framing according to ISO/IEC 14443 A**

The internal CRC coprocessor calculates the CRC value based on ISO/IEC 14443 A part 3 and handles parity generation internally according to the transfer speed. Automatic parity generation can be switched off using the MfRxReg register's ParityDisable bit.

## 8.1 Digital interfaces

### 8.1.1 Automatic microcontroller interface detection

The MFRC522 supports direct interfacing of hosts using SPI, I<sup>2</sup>C-bus or serial UART interfaces. The MFRC522 resets its interface and checks the current host interface type automatically after performing a power-on or hard reset. The MFRC522 identifies the host interface by sensing the logic levels on the control pins after the reset phase. This is done using a combination of fixed pin connections. [Table 5](#) shows the different connection configurations.

**Table 5. Connection protocol for detecting different interface types**

Pin	Interface type		
	UART (input)	SPI (output)	I <sup>2</sup> C-bus (I/O)
SDA	RX	NSS	SDA
I2C	0	0	1
EA	0	1	EA
D7	TX	MISO	SCL
D6	MX	MOSI	ADR_0
D5	DTRQ	SCK	ADR_1
D4	-	-	ADR_2
D3	-	-	ADR_3
D2	-	-	ADR_4
D1	-	-	ADR_5

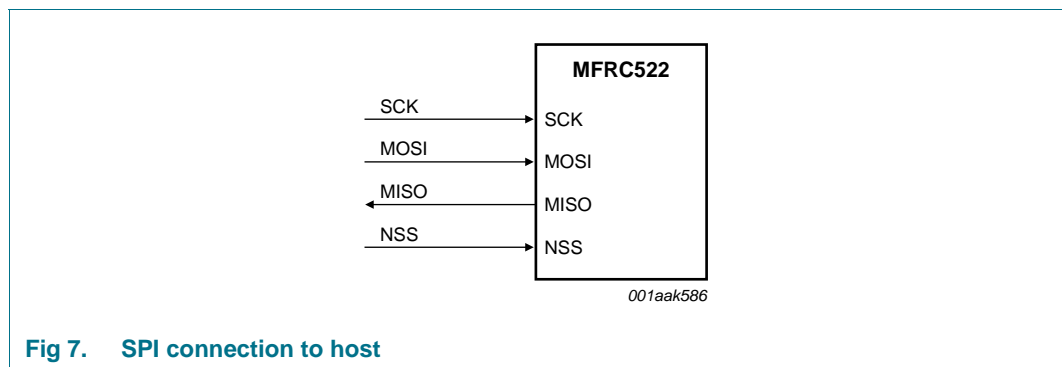


**8.1.2 Serial Peripheral Interface**

A serial peripheral interface (SPI compatible) is supported to enable high-speed communication to the host. The interface can handle data speeds up to 10 Mbit/s. When communicating with a host, the MFRC522 acts as a slave, receiving data from the external host for register settings, sending and receiving data relevant for RF interface communication.

An interface compatible with SPI enables high-speed serial communication between the MFRC522 and a microcontroller. The implemented interface is in accordance with the SPI standard.

The timing specification is given in [Section 14.1 on page 77](#).



The MFRC522 acts as a slave during SPI communication. The SPI clock signal SCK must be generated by the master. Data communication from the master to the slave uses the MOSI line. The MISO line is used to send data from the MFRC522 to the master.

Data bytes on both MOSI and MISO lines are sent with the MSB first. Data on both MOSI and MISO lines must be stable on the rising edge of the clock and can be changed on the falling edge. Data is provided by the MFRC522 on the falling clock edge and is stable during the rising clock edge.

**8.1.2.1 SPI read data**

Reading data using SPI requires the byte order shown in [Table 6](#) to be used. It is possible to read out up to n-data bytes.

The first byte sent defines both the mode and the address.

**Table 6. MOSI and MISO byte order**

Line	Byte 0	Byte 1	Byte 2	To	Byte n	Byte n + 1
MOSI	address 0	address 1	address 2	...	address n	00
MISO	X <sup>[1]</sup>	data 0	data 1	...	data n – 1	data n

[1] X = Do not care.

**Remark:** The MSB must be sent first.

8.1.2.2 SPI write data

To write data to the MFRC522 using SPI requires the byte order shown in [Table 7](#). It is possible to write up to n data bytes by only sending one address byte.

The first send byte defines both the mode and the address byte.

Table 7. MOSI and MISO byte order

Line	Byte 0	Byte 1	Byte 2	To	Byte n	Byte n + 1
MOSI	address 0	data 0	data 1	...	data n – 1	data n
MISO	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	...	X <sup>[1]</sup>	X <sup>[1]</sup>

[1] X = Do not care.

**Remark:** The MSB must be sent first.

8.1.2.3 SPI address byte

The address byte must meet the following format.

The MSB of the first byte defines the mode used. To read data from the MFRC522 the MSB is set to logic 1. To write data to the MFRC522 the MSB must be set to logic 0. Bits 6 to 1 define the address and the LSB is set to logic 0.

Table 8. Address byte 0 register; address MOSI

7 (MSB)	6	5	4	3	2	1	0 (LSB)
1 = read 0 = write	address						0

8.1.3 UART interface

8.1.3.1 Connection to a host

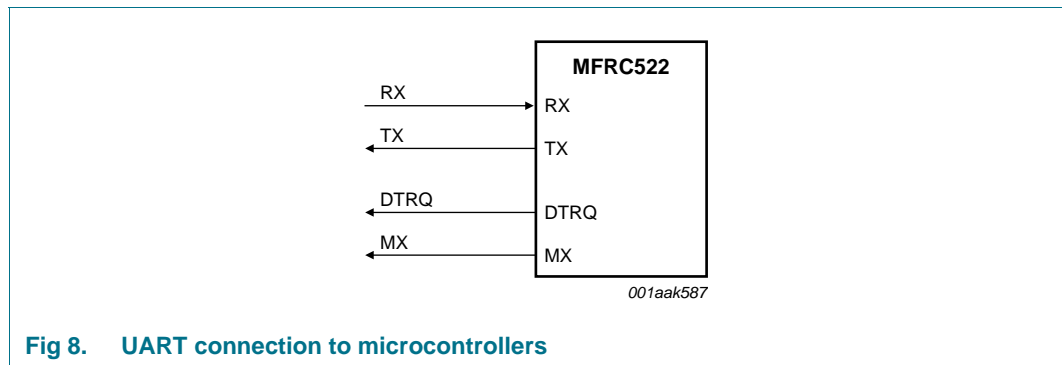


Fig 8. UART connection to microcontrollers

**Remark:** Signals DTRQ and MX can be disabled by clearing TestPinEnReg register's RS232LineEn bit.

### 8.1.3.2 Selectable UART transfer speeds

The internal UART interface is compatible with an RS232 serial interface.

The default transfer speed is 9.6 kBd. To change the transfer speed, the host controller must write a value for the new transfer speed to the SerialSpeedReg register. Bits BR\_T0[2:0] and BR\_T1[4:0] define the factors for setting the transfer speed in the SerialSpeedReg register.

The BR\_T0[2:0] and BR\_T1[4:0] settings are described in [Table 9](#). Examples of different transfer speeds and the relevant register settings are given in [Table 10](#).

**Table 9. BR\_T0 and BR\_T1 settings**

BR_Tn	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
BR_T0 factor	1	1	2	4	8	16	32	64
BR_T1 range	1 to 32	33 to 64	33 to 64	33 to 64	33 to 64	33 to 64	33 to 64	33 to 64

**Table 10. Selectable UART transfer speeds**

Transfer speed (kBd)	SerialSpeedReg value		Transfer speed accuracy (%) <sup>[1]</sup>
	Decimal	Hexadecimal	
7.2	250	FAh	-0.25
9.6	235	EBh	0.32
14.4	218	DAh	-0.25
19.2	203	CBh	0.32
38.4	171	ABh	0.32
57.6	154	9Ah	-0.25
115.2	122	7Ah	-0.25
128	116	74h	-0.06
230.4	90	5Ah	-0.25
460.8	58	3Ah	-0.25
921.6	28	1Ch	1.45
1228.8	21	15h	0.32

[1] The resulting transfer speed error is less than 1.5 % for all described transfer speeds.

The selectable transfer speeds shown in [Table 10](#) are calculated according to the following equations:

If BR\_T0[2:0] = 0:

$$transfer\ speed = \frac{27.12 \times 10^6}{(BR\_T0 + 1)} \tag{1}$$

If BR\_T0[2:0] > 0:

$$transfer\ speed = \left( \frac{27.12 \times 10^6}{(BR\_T1 + 33)} \right)_{2^{(BR\_T0 - 1)}} \tag{2}$$

**Remark:** Transfer speeds above 1228.8 kBd are not supported.

8.1.3.3 UART framing

Table 11. UART framing

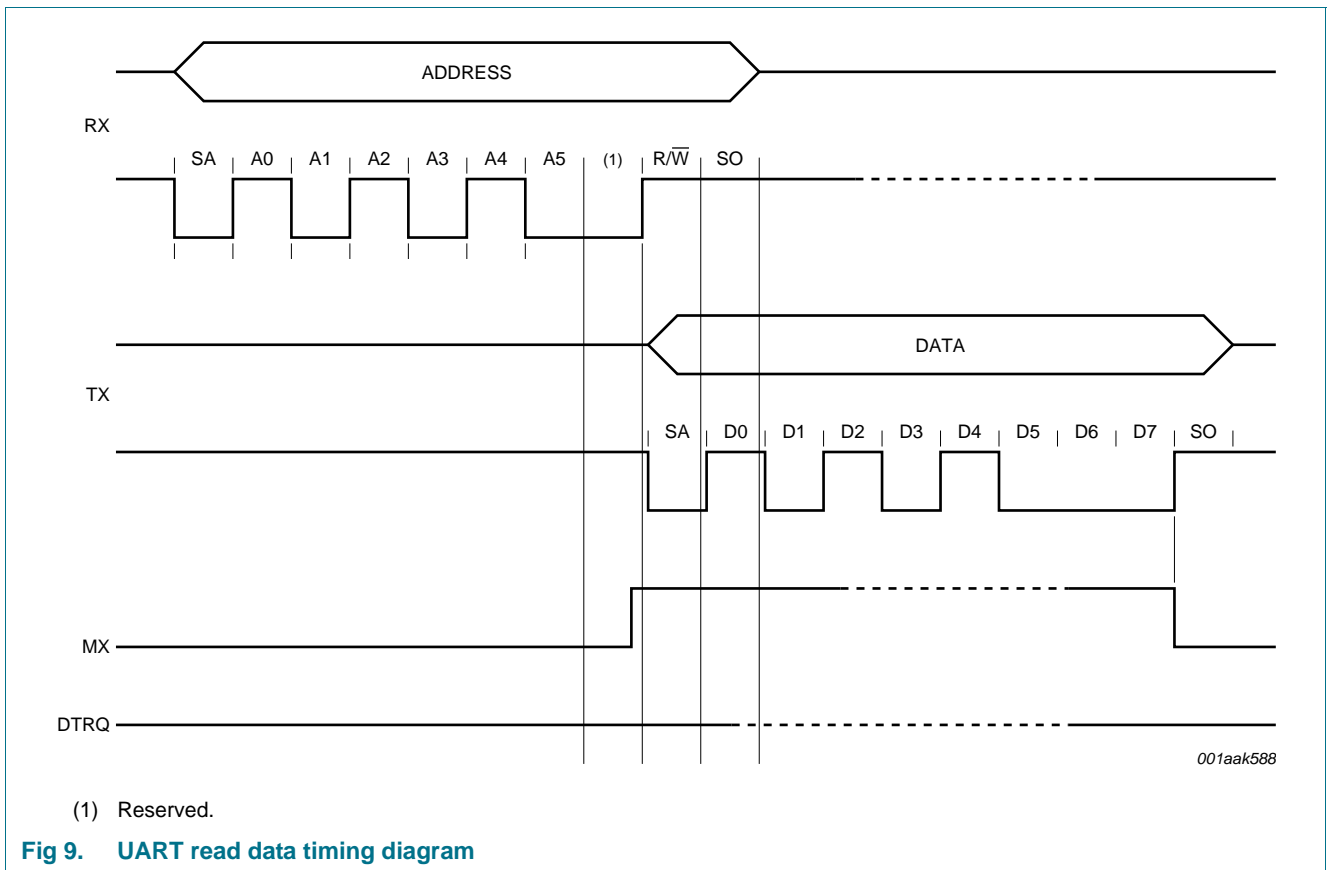
Bit	Length	Value
Start	1-bit	0
Data	8 bits	data
Stop	1-bit	1

**Remark:** The LSB for data and address bytes must be sent first. No parity bit is used during transmission.

**Read data:** To read data using the UART interface, the flow shown in Table 12 must be used. The first byte sent defines both the mode and the address.

Table 12. Read data byte order

Pin	Byte 0	Byte 1
RX (pin 24)	address	-
TX (pin 31)	-	data 0

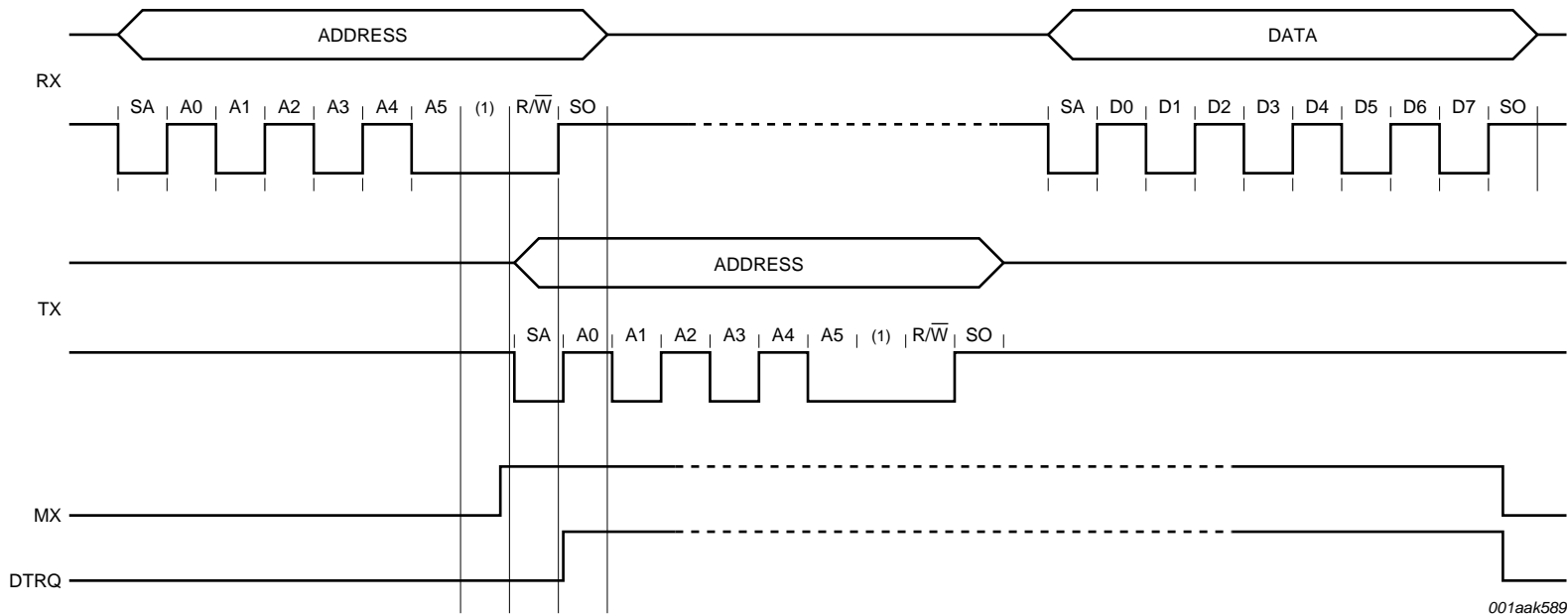


**Write data:** To write data to the MFRC522 using the UART interface, the structure shown in Table 13 must be used.

The first byte sent defines both the mode and the address.

**Table 13. Write data byte order**

<b>Pin</b>	<b>Byte 0</b>	<b>Byte 1</b>
RX (pin 24)	address 0	data 0
TX (pin 31)	-	address 0



(1) Reserved.

**Fig 10. UART write data timing diagram**

**Remark:** The data byte can be sent directly after the address byte on pin RX.

**Address byte:** The address byte has to meet the following format:

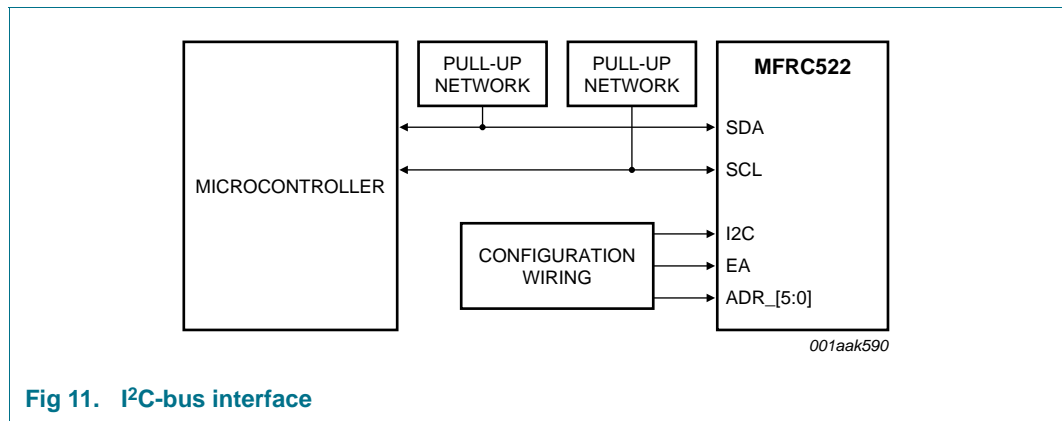
The MSB of the first byte sets the mode used. To read data from the MFRC522, the MSB is set to logic 1. To write data to the MFRC522 the MSB is set to logic 0. Bit 6 is reserved for future use, and bits 5 to 0 define the address; see [Table 14](#).

**Table 14. Address byte 0 register; address MOSI**

7 (MSB)	6	5	4	3	2	1	0 (LSB)
1 = read 0 = write	reserved	address					

**8.1.4 I<sup>2</sup>C-bus interface**

An I<sup>2</sup>C-bus (Inter-IC) interface is supported to enable a low-cost, low pin count serial bus interface to the host. The I<sup>2</sup>C-bus interface is implemented according to NXP Semiconductors' *I<sup>2</sup>C-bus interface specification, rev. 2.1, January 2000*. The interface can only act in Slave mode. Therefore the MFRC522 does not implement clock generation or access arbitration.



**Fig 11. I<sup>2</sup>C-bus interface**

The MFRC522 can act either as a slave receiver or slave transmitter in Standard mode, Fast mode and High-speed mode.

SDA is a bidirectional line connected to a positive supply voltage using a current source or a pull-up resistor. Both SDA and SCL lines are set HIGH when data is not transmitted. The MFRC522 has a 3-state output stage to perform the wired-AND function. Data on the I<sup>2</sup>C-bus can be transferred at data rates of up to 100 kBd in Standard mode, up to 400 kBd in Fast mode or up to 3.4 Mbit/s in High-speed mode.

If the I<sup>2</sup>C-bus interface is selected, spike suppression is activated on lines SCL and SDA as defined in the I<sup>2</sup>C-bus interface specification.

See [Table 155 on page 78](#) for timing requirements.

8.1.4.1 Data validity

Data on the SDA line must be stable during the HIGH clock period. The HIGH or LOW state of the data line must only change when the clock signal on SCL is LOW.

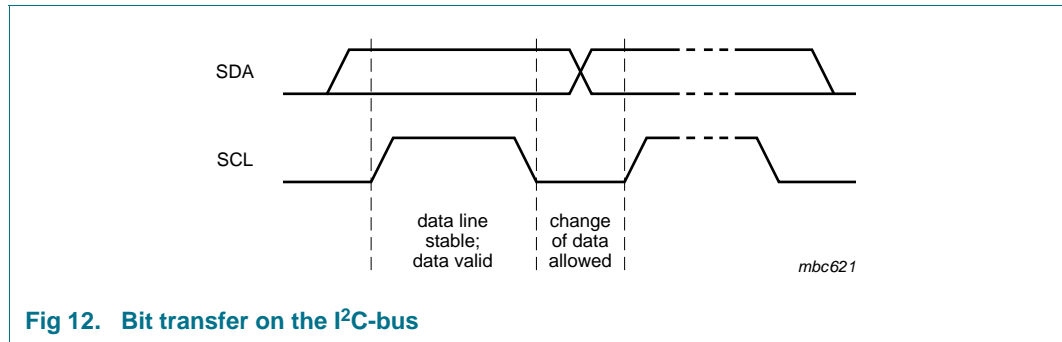


Fig 12. Bit transfer on the I<sup>2</sup>C-bus

8.1.4.2 START and STOP conditions

To manage the data transfer on the I<sup>2</sup>C-bus, unique START (S) and STOP (P) conditions are defined.

- A START condition is defined with a HIGH-to-LOW transition on the SDA line while SCL is HIGH.
- A STOP condition is defined with a LOW-to-HIGH transition on the SDA line while SCL is HIGH.

The I<sup>2</sup>C-bus master always generates the START and STOP conditions. The bus is busy after the START condition. The bus is free again a certain time after the STOP condition.

The bus stays busy if a repeated START (Sr) is generated instead of a STOP condition. The START (S) and repeated START (Sr) conditions are functionally identical. Therefore, S is used as a generic term to represent both the START (S) and repeated START (Sr) conditions.

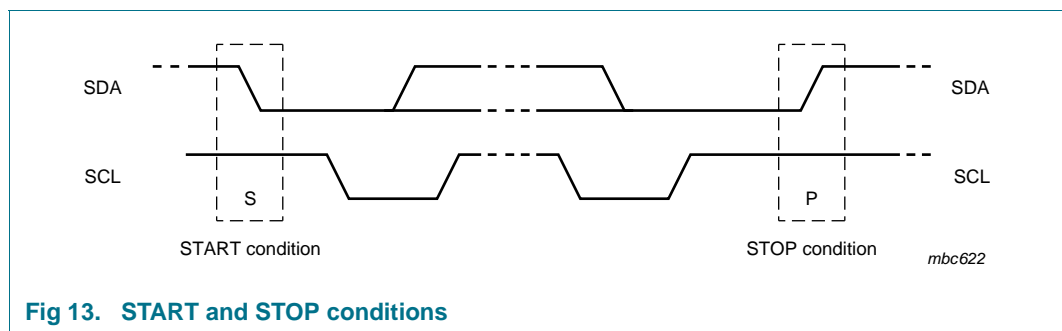


Fig 13. START and STOP conditions

8.1.4.3 Byte format

Each byte must be followed by an acknowledge bit. Data is transferred with the MSB first; see [Figure 16](#). The number of transmitted bytes during one data transfer is unrestricted but must meet the read/write cycle format.



8.1.4.4 Acknowledge

An acknowledge must be sent at the end of one data byte. The acknowledge-related clock pulse is generated by the master. The transmitter of data, either master or slave, releases the SDA line (HIGH) during the acknowledge clock pulse. The receiver pulls down the SDA line during the acknowledge clock pulse so that it remains stable LOW during the HIGH period of this clock pulse.

The master can then generate either a STOP (P) condition to stop the transfer or a repeated START (Sr) condition to start a new transfer.

A master-receiver indicates the end of data to the slave-transmitter by not generating an acknowledge on the last byte that was clocked out by the slave. The slave-transmitter releases the data line to allow the master to generate a STOP (P) or repeated START (Sr) condition.

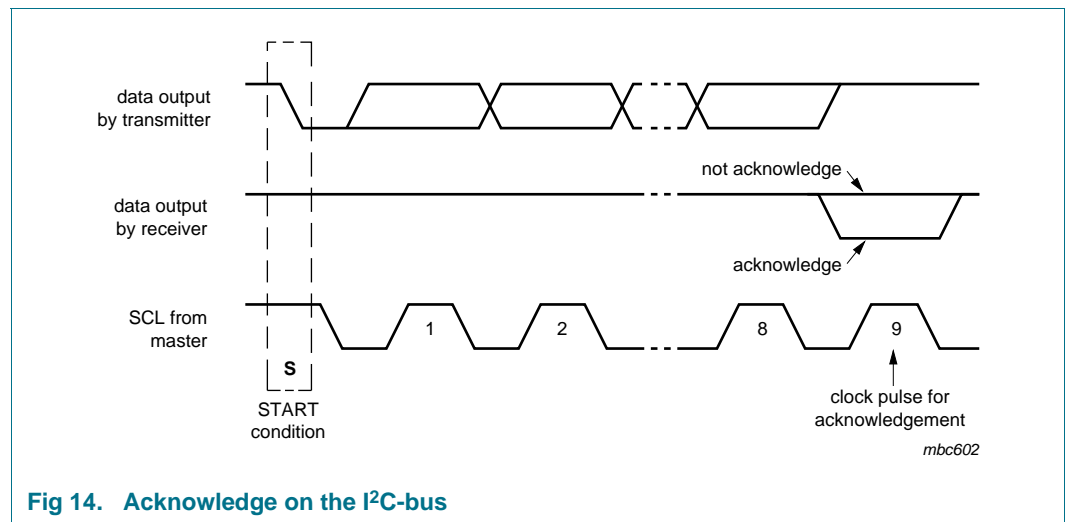


Fig 14. Acknowledge on the I<sup>2</sup>C-bus

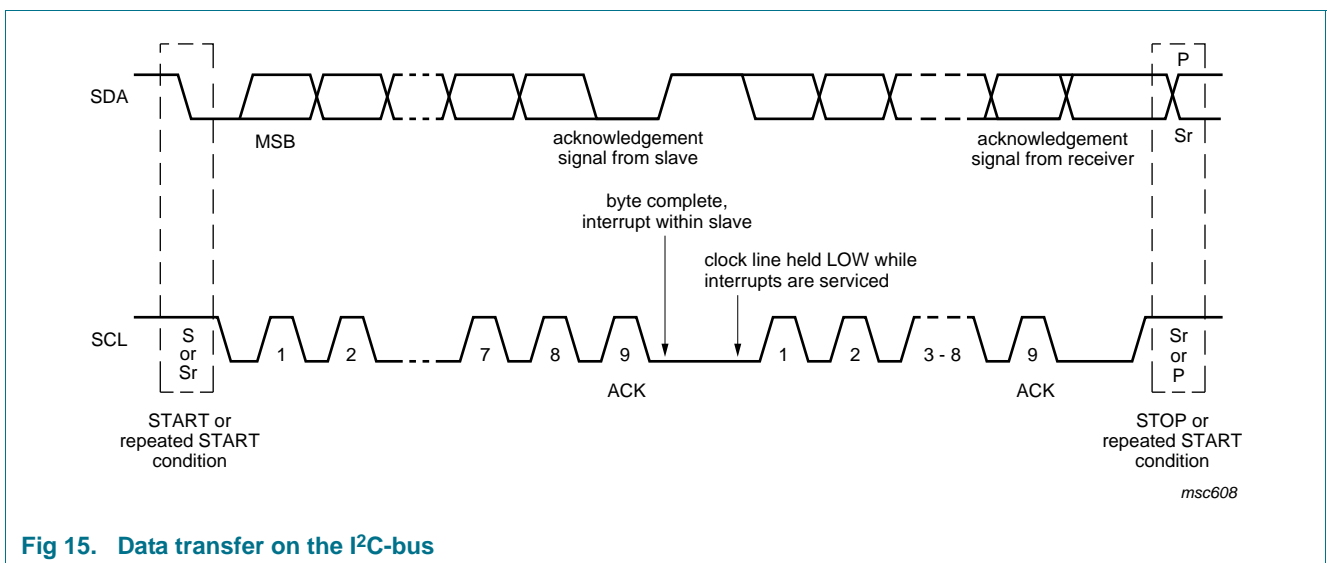


Fig 15. Data transfer on the I<sup>2</sup>C-bus

**8.1.4.5 7-Bit addressing**

During the I<sup>2</sup>C-bus address procedure, the first byte after the START condition is used to determine which slave will be selected by the master.

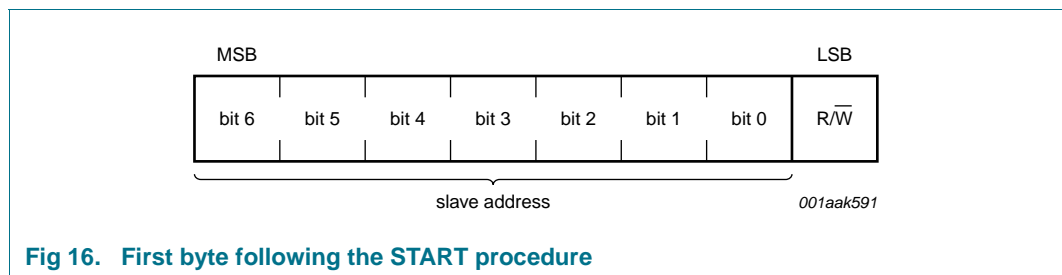
Several address numbers are reserved. During device configuration, the designer must ensure that collisions with these reserved addresses cannot occur. Check the *I<sup>2</sup>C-bus specification* for a complete list of reserved addresses.

The I<sup>2</sup>C-bus address specification is dependent on the definition of pin EA. Immediately after releasing pin NRSTPD or after a power-on reset, the device defines the I<sup>2</sup>C-bus address according to pin EA.

If pin EA is set LOW, the upper 4 bits of the device bus address are reserved by NXP Semiconductors and set to 0101b for all MFRC522 devices. The remaining 3 bits (ADR\_0, ADR\_1, ADR\_2) of the slave address can be freely configured by the customer to prevent collisions with other I<sup>2</sup>C-bus devices.

If pin EA is set HIGH, ADR\_0 to ADR\_5 can be completely specified at the external pins according to [Table 5 on page 9](#). ADR\_6 is always set to logic 0.

In both modes, the external address coding is latched immediately after releasing the reset condition. Further changes at the used pins are not taken into consideration. Depending on the external wiring, the I<sup>2</sup>C-bus address pins can be used for test signal outputs.



**Fig 16. First byte following the START procedure**

**8.1.4.6 Register write access**

To write data from the host controller using the I<sup>2</sup>C-bus to a specific register in the MFRC522 the following frame format must be used.

- The first byte of a frame indicates the device address according to the I<sup>2</sup>C-bus rules.
- The second byte indicates the register address followed by up to n-data bytes.

In one frame all data bytes are written to the same register address. This enables fast FIFO buffer access. The Read/Write (R/W) bit is set to logic 0.

8.1.4.7 Register read access

To read out data from a specific register address in the MFRC522, the host controller must use the following procedure:

- Firstly, a write access to the specific register address must be performed as indicated in the frame that follows
- The first byte of a frame indicates the device address according to the I<sup>2</sup>C-bus rules
- The second byte indicates the register address. No data bytes are added
- The Read/Write bit is 0

After the write access, read access can start. The host sends the device address of the MFRC522. In response, the MFRC522 sends the content of the read access register. In one frame all data bytes can be read from the same register address. This enables fast FIFO buffer access or register polling.

The Read/Write (R/W) bit is set to logic 1.

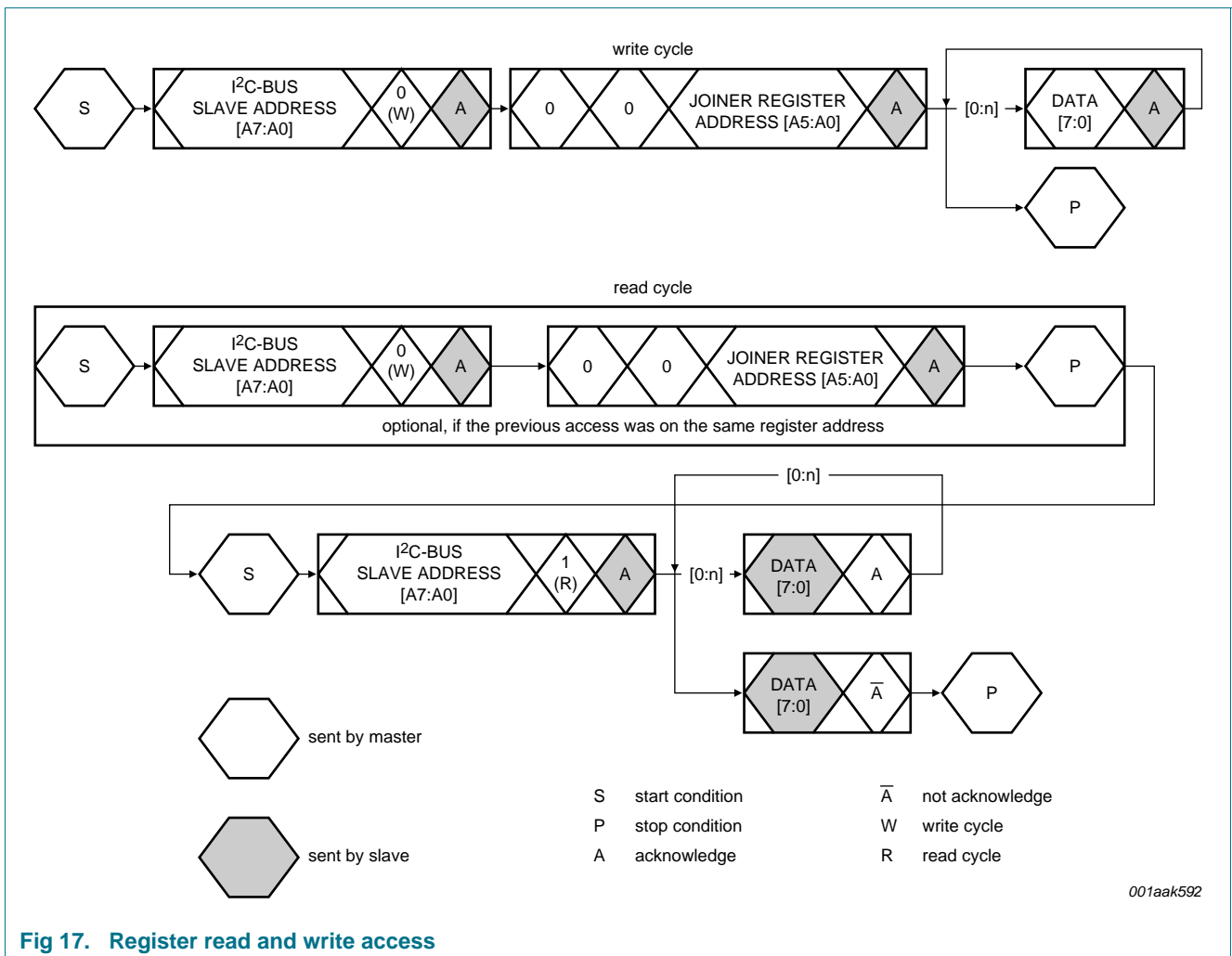


Fig 17. Register read and write access

**8.1.4.8 High-speed mode**

In High-speed mode (HS mode), the device can transfer information at data rates of up to 3.4 Mbit/s, while remaining fully downward-compatible with Fast or Standard mode (F/S mode) for bidirectional communication in a mixed-speed bus system.

**8.1.4.9 High-speed transfer**

To achieve data rates of up to 3.4 Mbit/s the following improvements have been made to I<sup>2</sup>C-bus operation.

- The inputs of the device in HS mode incorporate spike suppression, a Schmitt trigger on the SDA and SCL inputs and different timing constants when compared to F/S mode
- The output buffers of the device in HS mode incorporate slope control of the falling edges of the SDA and SCL signals with different fall times compared to F/S mode

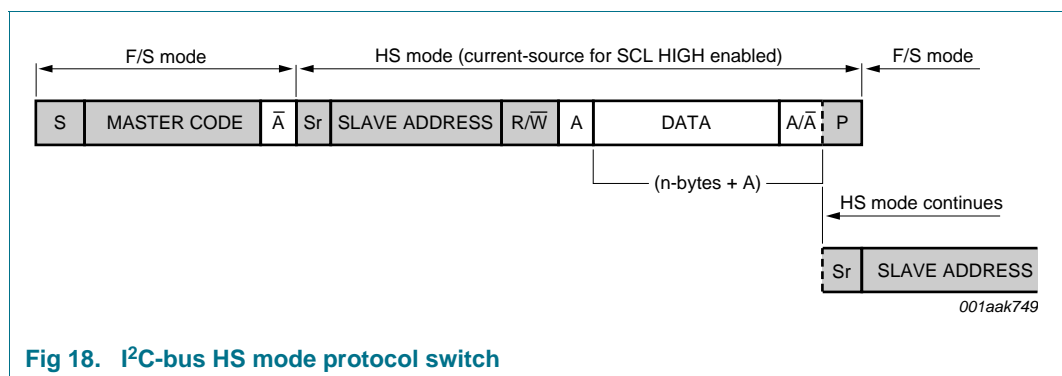
**8.1.4.10 Serial data transfer format in HS mode**

The HS mode serial data transfer format meets the Standard mode I<sup>2</sup>C-bus specification. HS mode can only start after all of the following conditions (all of which are in F/S mode):

1. START condition (S)
2. 8-bit master code (00001XXXb)
3. Not-acknowledge bit ( $\bar{A}$ )

When HS mode starts, the active master sends a repeated START condition (Sr) followed by a 7-bit slave address with a R/W bit address and receives an acknowledge bit (A) from the selected MFRC522.

Data transfer continues in HS mode after the next repeated START (Sr), only switching back to F/S mode after a STOP condition (P). To reduce the overhead of the master code, a master links a number of HS mode transfers, separated by repeated START conditions (Sr).



**Fig 18. I<sup>2</sup>C-bus HS mode protocol switch**

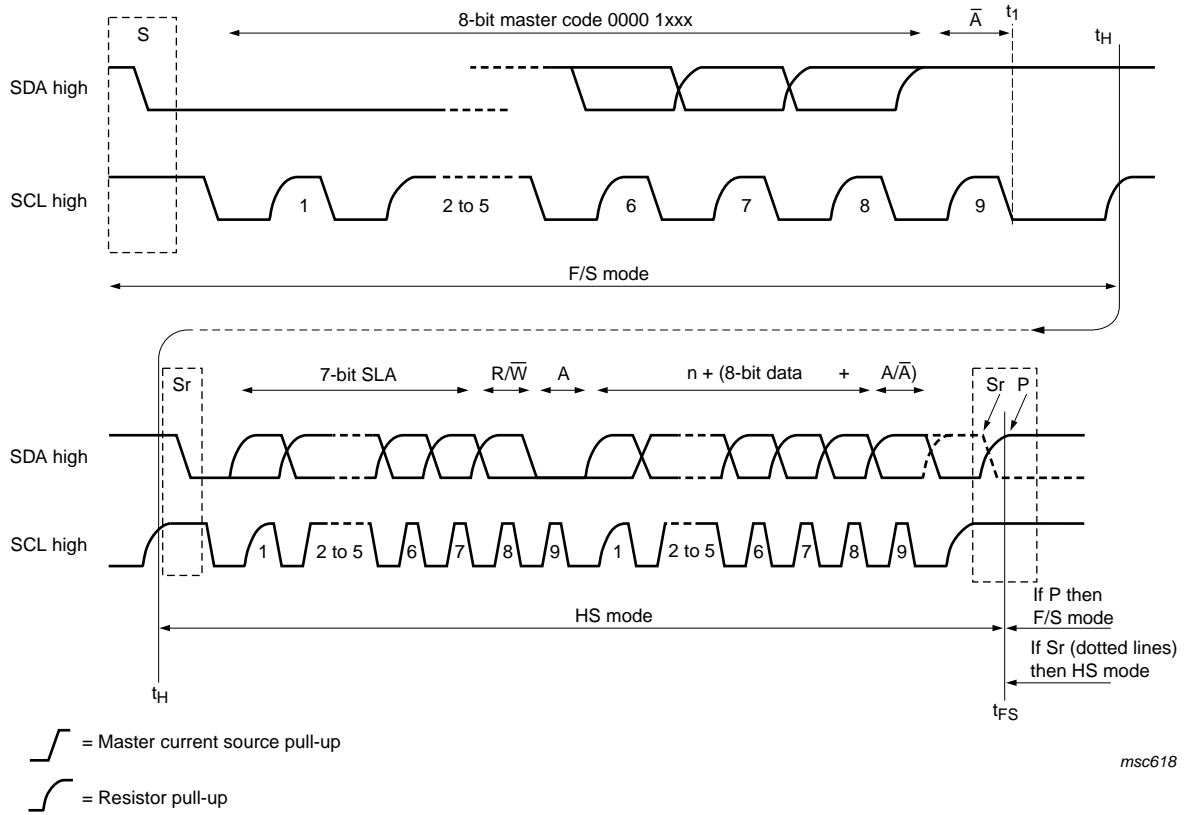


Fig 19. I<sup>2</sup>C-bus HS mode protocol frame

#### 8.1.4.11 Switching between F/S mode and HS mode

After reset and initialization, the MFRC522 is in Fast mode (which is in effect F/S mode as Fast mode is downward-compatible with Standard mode). The connected MFRC522 recognizes the "S 00001XXX A" sequence and switches its internal circuitry from the Fast mode setting to the HS mode setting.

The following actions are taken:

1. Adapt the SDA and SCL input filters according to the spike suppression requirement in HS mode.
2. Adapt the slope control of the SDA output stages.

It is possible for system configurations that do not have other I<sup>2</sup>C-bus devices involved in the communication to switch to HS mode permanently. This is implemented by setting Status2Reg register's I<sup>2</sup>CForceHS bit to logic 1. In permanent HS mode, the master code is not required to be sent. This is not defined in the specification and must only be used when no other devices are connected on the bus. In addition, spikes on the I<sup>2</sup>C-bus lines must be avoided because of the reduced spike suppression.

#### 8.1.4.12 MFRC522 at lower speed modes

MFRC522 is fully downward-compatible and can be connected to an F/S mode I<sup>2</sup>C-bus system. The device stays in F/S mode and communicates at F/S mode speeds because a master code is not transmitted in this configuration.

## 8.2 Analog interface and contactless UART

### 8.2.1 General

The integrated contactless UART supports the external host online with framing and error checking of the protocol requirements up to 848 kBd. An external circuit can be connected to the communication interface pins MFIN and MFOUT to modulate and demodulate the data.

The contactless UART handles the protocol requirements for the communication protocols in cooperation with the host. Protocol handling generates bit and byte-oriented framing. In addition, it handles error detection such as parity and CRC, based on the various supported contactless communication protocols.

**Remark:** The size and tuning of the antenna and the power supply voltage have an important impact on the achievable operating distance.

### 8.2.2 TX p-driver

The signal on pins TX1 and TX2 is the 13.56 MHz energy carrier modulated by an envelope signal. It can be used to drive an antenna directly using a few passive components for matching and filtering; see [Section 15 on page 80](#). The signal on pins TX1 and TX2 can be configured using the TxControlReg register; see [Section 9.3.2.5 on page 49](#).

The modulation index can be set by adjusting the impedance of the drivers. The impedance of the p-driver can be configured using registers CWGsPReg and ModGsPReg. The impedance of the n-driver can be configured using the GsNReg register. The modulation index also depends on the antenna design and tuning.

The TxModeReg and TxSelReg registers control the data rate and framing during transmission and the antenna driver setting to support the different requirements at the different modes and transfer speeds.

**Table 15. Register and bit settings controlling the signal on pin TX1**

Bit Tx1RFEn	Bit Force 100ASK	Bit InvTx1RFOn	Bit InvTx1RFOff	Envelope	Pin TX1	GSPMos	GSNMos	Remarks
0	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	not specified if RF is switched off
1	0	0	X <sup>[1]</sup>	0	RF	pMod	nMod	100 % ASK: pin TX1 pulled to logic 0, independent of the InvTx1RFOff bit
				1	RF	pCW	nCW	
	0	1	X <sup>[1]</sup>	0	RF	pMod	nMod	
				1	RF	pCW	nCW	
1	1	X <sup>[1]</sup>	X <sup>[1]</sup>	0	0	pMod	nMod	
				1	RF_n	pCW	nCW	

[1] X = Do not care.

Table 16. Register and bit settings controlling the signal on pin TX2

Bit Tx1RFEn	Bit Force 100ASK	Bit Tx2CW	Bit InvTx2RFOOn	Bit InvTx2RFOff	Envelope	Pin TX2	GSPMos	GSNMos	Remarks			
0	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	X <sup>[1]</sup>	not specified if RF is switched off			
1	0	0	0	X <sup>[1]</sup>	0	RF	pMod	nMod	-			
				1	RF	pCW	nCW					
				X <sup>[1]</sup>	0	RF_n	pMod	nMod				
				1	RF_n	pCW	nCW					
	1	0	0	X <sup>[1]</sup>	X <sup>[1]</sup>	RF	pCW	nCW	conductance always CW for the Tx2CW bit			
				X <sup>[1]</sup>	X <sup>[1]</sup>	RF_n	pCW	nCW				
				1	0	X <sup>[1]</sup>	0	0		pMod	nMod	100 % ASK: pin TX2 pulled to logic 0 (independent of the InvTx2RFOOn/InvTx2RFOff bits)
						X <sup>[1]</sup>	0	0		pMod	nMod	
1	0	0	X <sup>[1]</sup>	X <sup>[1]</sup>	RF	pCW	nCW	-				
			X <sup>[1]</sup>	X <sup>[1]</sup>	RF_n	pCW	nCW					

[1] X = Do not care.

The following abbreviations have been used in [Table 15](#) and [Table 16](#):

- RF: 13.56 MHz clock derived from 27.12 MHz quartz crystal oscillator divided by 2
- RF\_n: inverted 13.56 MHz clock
- GSPMos: conductance, configuration of the PMOS array
- GSNMos: conductance, configuration of the NMOS array
- pCW: PMOS conductance value for continuous wave defined by the CWGsPReg register
- pMod: PMOS conductance value for modulation defined by the ModGsPReg register
- nCW: NMOS conductance value for continuous wave defined by the GsNReg register's CWGsN[3:0] bits
- nMod: NMOS conductance value for modulation defined by the GsNReg register's ModGsN[3:0] bits
- X = do not care.

**Remark:** If only one driver is switched on, the values for CWGsPReg, ModGsPReg and GsNReg registers are used for both drivers.



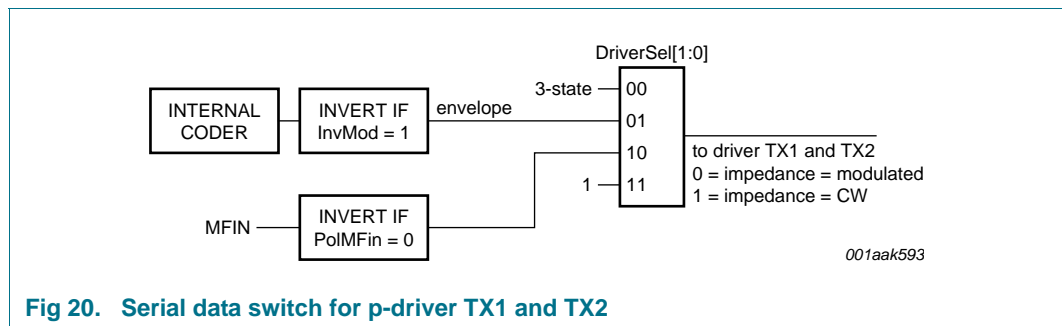
**8.2.3 Serial data switch**

Two main blocks are implemented in the MFRC522. The digital block comprises the state machines, encoder/decoder logic. The analog block comprises the modulator and antenna drivers, the receiver and amplifiers. It is possible for the interface between these two blocks to be configured so that the interfacing signals are routed to pins MFIN and MFOUT.

This topology allows the analog block of the MFRC522 to be connected to the digital block of another device.

The serial signal switch is controlled by the TxSelReg and RxSelReg registers.

Figure 20 shows the serial data switch for p-driver TX1 and TX2.



**Fig 20. Serial data switch for p-driver TX1 and TX2**

**8.2.4 MFIN and MFOUT interface support**

The MFRC522 is divided into a digital circuit block and an analog circuit block. The digital block contains state machines, encoder and decoder logic and so on. The analog block contains the modulator and antenna drivers, receiver and amplifiers. The interface between these two blocks can be configured so that the interfacing signals can be routed to pins MFIN and MFOUT; see Figure 21 on page 27. This configuration is implemented using TxSelReg register’s MFOutSel[3:0] and DriverSel[1:0] bits and RxSelReg register’s UARTSel[1:0] bits.

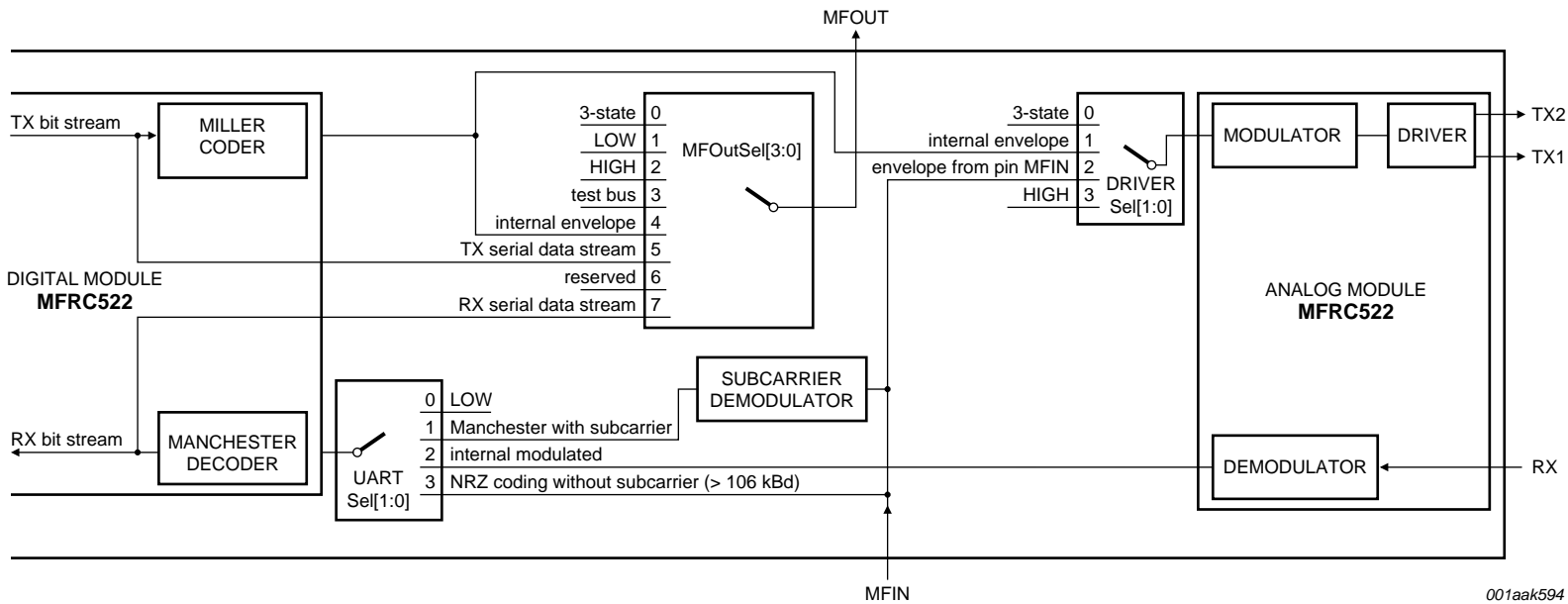
This topology allows some parts of the analog block to be connected to the digital block of another device.

Switch MFOutSel in the TxSelReg register can be used to measure MIFARE and ISO/IEC14443 A related signals. This is especially important during the design-in phase or for test purposes as it enables checking of the transmitted and received data.

The most important use of pins MFIN and MFOUT is found in the active antenna concept. An external active antenna circuit can be connected to the MFRC522’s digital block. Switch MFOutSel must be configured so that the internal Miller encoded signal is sent to pin MFOUT (MFOutSel = 100b). UARTSel[1:0] must be configured to receive a Manchester signal with subcarrier from pin MFIN (UARTSel[1:0] = 01).

It is possible to connect a passive antenna to pins TX1, TX2 and RX (using the appropriate filter and matching circuit) and an active antenna to pins MFOUT and MFIN at the same time. In this configuration, two RF circuits can be driven (one after another) by a single host processor.

**Remark:** Pins MFIN and MFOUT have a dedicated supply on pin SVDD with the ground on pin PVSS. If pin MFIN is not used it must be connected to either pin SVDD or pin PVSS. If pin SVDD is not used it must be connected to either pin DVDD, pin PVDD or any other voltage supply pin.



001aak594

Fig 21. Overview of MFIN and MFOUT signal routing

### 8.2.5 CRC coprocessor

The following CRC coprocessor parameters can be configured:

- The CRC preset value can be either 0000h, 6363h, A671h or FFFFh depending on the ModeReg register's CRCPreset[1:0] bits setting
- The CRC polynomial for the 16-bit CRC is fixed to  $x^{16} + x^{12} + x^5 + 1$
- The CRCResultReg register indicates the result of the CRC calculation. This register is split into two 8-bit registers representing the higher and lower bytes.
- The ModeReg register's MSBFirst bit indicates that data will be loaded with the MSB first.

**Table 17. CRC coprocessor parameters**

Parameter	Value
CRC register length	16-bit CRC
CRC algorithm	algorithm according to ISO/IEC 14443 A and ITU-T
CRC preset value	0000h, 6363h, A671h or FFFFh depending on the setting of the ModeReg register's CRCPreset[1:0] bits

## 8.3 FIFO buffer

An 8 × 64 bit FIFO buffer is used in the MFRC522. It buffers the input and output data stream between the host and the MFRC522's internal state machine. This makes it possible to manage data streams up to 64 bytes long without the need to take timing constraints into account.

### 8.3.1 Accessing the FIFO buffer

The FIFO buffer input and output data bus is connected to the FIFODataReg register. Writing to this register stores one byte in the FIFO buffer and increments the internal FIFO buffer write pointer. Reading from this register shows the FIFO buffer contents stored in the FIFO buffer read pointer and decrements the FIFO buffer read pointer. The distance between the write and read pointer can be obtained by reading the FIFOLevelReg register.

When the microcontroller starts a command, the MFRC522 can, while the command is in progress, access the FIFO buffer according to that command. Only one FIFO buffer has been implemented which can be used for input and output. The microcontroller must ensure that there are not any unintentional FIFO buffer accesses.

### 8.3.2 Controlling the FIFO buffer

The FIFO buffer pointers can be reset by setting FIFOLevelReg register's FlushBuffer bit to logic 1. Consequently, the FIFOLevel[6:0] bits are all set to logic 0 and the ErrorReg register's BufferOvfl bit is cleared. The bytes stored in the FIFO buffer are no longer accessible allowing the FIFO buffer to be filled with another 64 bytes.

### 8.3.3 FIFO buffer status information

The host can get the following FIFO buffer status information:

- Number of bytes stored in the FIFO buffer: FIFOLevelReg register's FIFOLevel[6:0]
- FIFO buffer almost full warning: Status1Reg register's HiAlert bit

- FIFO buffer almost empty warning: Status1Reg register's LoAlert bit
- FIFO buffer overflow warning: ErrorReg register's BufferOvfl bit. The BufferOvfl bit can only be cleared by setting the FIFOLevelReg register's FlushBuffer bit.

The MFRC522 can generate an interrupt signal when:

- ComIEnReg register's LoAlertIEn bit is set to logic 1. It activates pin IRQ when Status1Reg register's LoAlert bit changes to logic 1.
- ComIEnReg register's HiAlertIEn bit is set to logic 1. It activates pin IRQ when Status1Reg register's HiAlert bit changes to logic 1.

If the maximum number of WaterLevel bytes (as set in the WaterLevelReg register) or less are stored in the FIFO buffer, the HiAlert bit is set to logic 1. It is generated according to [Equation 3](#):

$$HiAlert = (64 - FIFOLength) \leq WaterLevel \quad (3)$$

If the number of WaterLevel bytes (as set in the WaterLevelReg register) or less are stored in the FIFO buffer, the LoAlert bit is set to logic 1. It is generated according to [Equation 4](#):

$$LoAlert = FIFOLength \leq WaterLevel \quad (4)$$

## 8.4 Interrupt request system

The MFRC522 indicates certain events by setting the Status1Reg register's IRq bit and, if activated, by pin IRQ. The signal on pin IRQ can be used to interrupt the host using its interrupt handling capabilities. This allows the implementation of efficient host software.

### 8.4.1 Interrupt sources overview

[Table 18](#) shows the available interrupt bits, the corresponding source and the condition for its activation. The ComIrqReg register's TimerIRq interrupt bit indicates an interrupt set by the timer unit which is set when the timer decrements from 1 to 0.

The ComIrqReg register's TxIRq bit indicates that the transmitter has finished. If the state changes from sending data to transmitting the end of the frame pattern, the transmitter unit automatically sets the interrupt bit. The CRC coprocessor sets the DivIrqReg register's CRCIRq bit after processing all the FIFO buffer data which is indicated by CRCReady bit = 1.

The ComIrqReg register's RxIRq bit indicates an interrupt when the end of the received data is detected. The ComIrqReg register's IdleIRq bit is set if a command finishes and the Command[3:0] value in the CommandReg register changes to idle (see [Table 149 on page 69](#)).

The ComIrqReg register's HiAlertIRq bit is set to logic 1 when the Status1Reg register's HiAlert bit is set to logic 1 which means that the FIFO buffer has reached the level indicated by the WaterLevel[5:0] bits.

The ComIrqReg register's LoAlertIRq bit is set to logic 1 when the Status1Reg register's LoAlert bit is set to logic 1 which means that the FIFO buffer has reached the level indicated by the WaterLevel[5:0] bits.

The ComIrqReg register's ErrIRq bit indicates an error detected by the contactless UART during send or receive. This is indicated when any bit is set to logic 1 in register ErrorReg.

**Table 18. Interrupt sources**

Interrupt flag	Interrupt source	Trigger action
IRq	timer unit	the timer counts from 1 to 0
TxIRq	transmitter	a transmitted data stream ends
CRCIRq	CRC coprocessor	all data from the FIFO buffer has been processed
RxIRq	receiver	a received data stream ends
IdleIRq	ComIrqReg register	command execution finishes
HiAlertIRq	FIFO buffer	the FIFO buffer is almost full
LoAlertIRq	FIFO buffer	the FIFO buffer is almost empty
ErrIRq	contactless UART	an error is detected

## 8.5 Timer unit

The MFRC522A has a timer unit which the external host can use to manage timing tasks. The timer unit can be used in one of the following timer/counter configurations:

- Timeout counter
- Watchdog counter
- Stop watch
- Programmable one shot
- Periodical trigger

The timer unit can be used to measure the time interval between two events or to indicate that a specific event occurred after a specific time. The timer can be triggered by events explained in the paragraphs below. The timer does not influence any internal events, for example, a time-out during data reception does not automatically influence the reception process. Furthermore, several timer-related bits can be used to generate an interrupt.

The timer has an input clock of 13.56 MHz derived from the 27.12 MHz quartz crystal oscillator. The timer consists of two stages: prescaler and counter.

The prescaler (TPrescaler) is a 12-bit counter. The reload values (TReloadVal\_Hi[7:0] and TReloadVal\_Lo[7:0]) for TPrescaler can be set between 0 and 4095 in the TModeReg register's TPrescaler\_Hi[3:0] bits and TPrescalerReg register's TPrescaler\_Lo[7:0] bits.

The reload value for the counter is defined by 16 bits between 0 and 65535 in the TReloadReg register.

The current value of the timer is indicated in the TCounterValReg register.

When the counter reaches 0, an interrupt is automatically generated, indicated by the ComIrqReg register's TimerIRq bit setting. If enabled, this event can be indicated on pin IRQ. The TimerIRq bit can be set and reset by the host. Depending on the configuration, the timer will stop at 0 or restart with the value set in the TReloadReg register.

The timer status is indicated by the Status1Reg register's TRunning bit.

The timer can be started manually using the ControlReg register's TStartNow bit and stopped using the ControlReg register's TStopNow bit.

The timer can also be activated automatically to meet any dedicated protocol requirements by setting the TModeReg register's TAuto bit to logic 1.

The delay time of a timer stage is set by the reload value + 1. The total delay time ( $t_{d1}$ ) is calculated using [Equation 5](#):

$$t_{d1} = \frac{(TPrescaler \times 2 + 1) \times (TReloadVal + 1)}{13.56 \text{ MHz}} \quad (5)$$

An example of calculating total delay time ( $t_d$ ) is shown in [Equation 6](#), where the TPrescaler value = 4095 and TReloadVal = 65535:

$$39.59 \text{ s} = \frac{(4095 \times 2 + 1) \times (65535 + 1)}{13.56 \text{ MHz}} \quad (6)$$

**Example:** To give a delay time of 25  $\mu\text{s}$  requires 339 clock cycles to be counted and a TPrescaler value of 169. This configures the timer to count up to 65535 time-slots for every 25  $\mu\text{s}$  period.

The MFRC522 version 2.0 offers in addition a second prescaler timer. Due to the fact that the prescaler counts down to 0 the prescaler period always count an odd number of clocks (1, 3, 5, ..). This may lead to inaccuracy. The second available prescaler timer implements the possibility to change the prescaler reload value to odd numbers, which results in an even prescaler period. This new prescaler can be enabled only in version 2.0 using the register bit DemodeReg, see [Table 72](#). Within this option, the total delay time ( $t_{d2}$ ) is calculated using [Equation 5](#):

$$t_{d2} = \frac{(TPrescaler \times 2 + 2) \times (TReloadVal + 1)}{13.56 \text{ MHz}} \quad (7)$$

**8.6 Power reduction modes**

**8.6.1 Hard power-down**

Hard power-down is enabled when pin NRSTPD is LOW. This turns off all internal current sinks including the oscillator. All digital input buffers are separated from the input pins and clamped internally (except pin NRSTPD). The output pins are frozen at either a HIGH or LOW level.

**8.6.2 Soft power-down mode**

Soft Power-down mode is entered immediately after the CommandReg register's PowerDown bit is set to logic 1. All internal current sinks are switched off, including the oscillator buffer. However, the digital input buffers are not separated from the input pins and keep their functionality. The digital output pins do not change their state.

During soft power-down, all register values, the FIFO buffer content and the configuration keep their current contents.

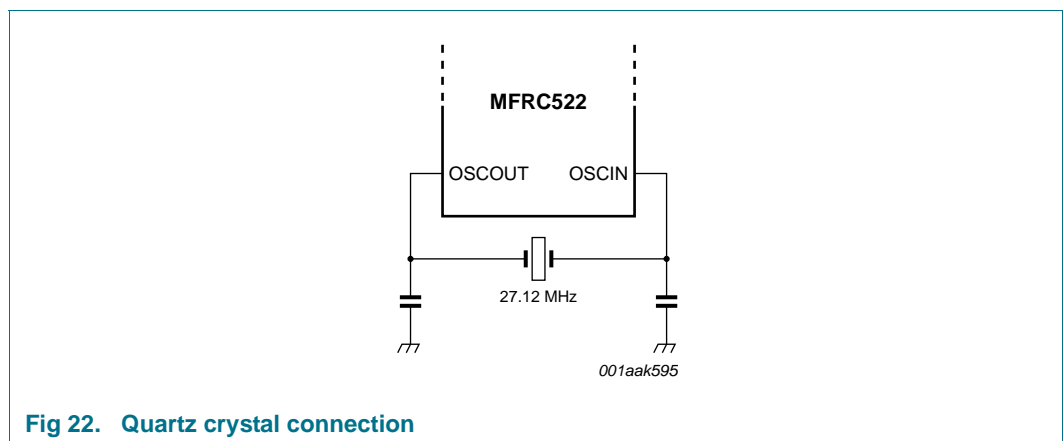
After setting the PowerDown bit to logic 0, it takes 1024 clocks until the Soft power-down mode is exited indicated by the PowerDown bit. Setting it to logic 0 does not immediately clear it. It is cleared automatically by the MFRC522 when Soft power-down mode is exited.

**Remark:** If the internal oscillator is used, you must take into account that it is supplied by pin AVDD and it will take a certain time ( $t_{osc}$ ) until the oscillator is stable and the clock cycles can be detected by the internal logic. It is recommended for the serial UART, to first send the value 55h to the MFRC522. The oscillator must be stable for further access to the registers. To ensure this, perform a read access to address 0 until the MFRC522 answers to the last read ready command with the register content of address 0. This indicates that the MFRC522 is ready.

**8.6.3 Transmitter power-down mode**

The Transmitter Power-down mode switches off the internal antenna drivers thereby, turning off the RF field. Transmitter power-down mode is entered by setting either the TxControlReg register's Tx1RFEn bit or Tx2RFEn bit to logic 0.

**8.7 Oscillator circuit**



**Fig 22. Quartz crystal connection**

The clock applied to the MFRC522 provides a time basis for the synchronous system's encoder and decoder. The stability of the clock frequency, therefore, is an important factor for correct operation. To obtain optimum performance, clock jitter must be reduced as much as possible. This is best achieved using the internal oscillator buffer with the recommended circuitry.

If an external clock source is used, the clock signal must be applied to pin OSCIN. In this case, special care must be taken with the clock duty cycle and clock jitter and the clock quality must be verified.

**8.8 Reset and oscillator start-up time**

**8.8.1 Reset timing requirements**

The reset signal is filtered by a hysteresis circuit and a spike filter before it enters the digital circuit. The spike filter rejects signals shorter than 10 ns. In order to perform a reset, the signal must be LOW for at least 100 ns.

**8.8.2 Oscillator start-up time**

If the MFRC522 has been set to a Power-down mode or is powered by a V<sub>DDX</sub> supply, the start-up time for the MFRC522 depends on the oscillator used and is shown in [Figure 23](#).

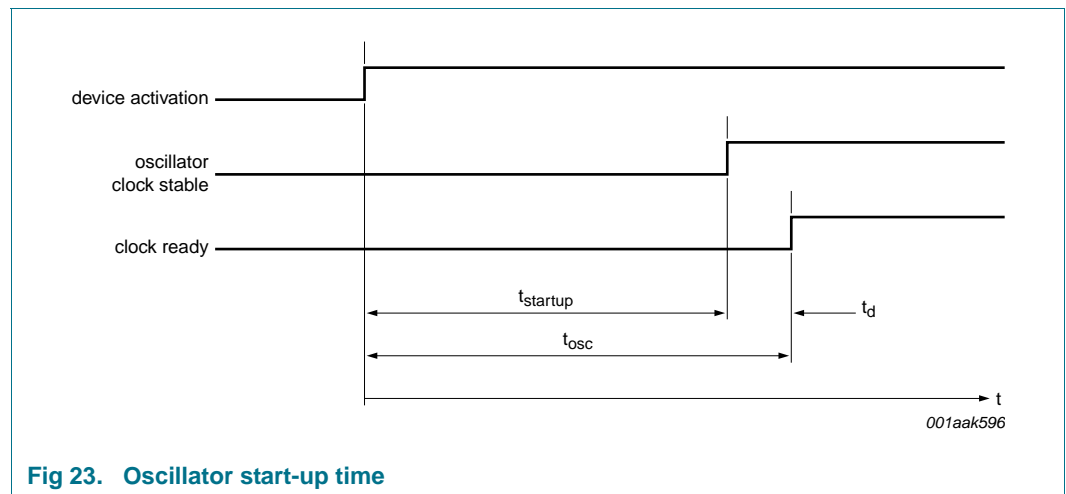
The time (t<sub>startup</sub>) is the start-up time of the crystal oscillator circuit. The crystal oscillator start-up time is defined by the crystal.

The time (t<sub>d</sub>) is the internal delay time of the MFRC522 when the clock signal is stable before the MFRC522 can be addressed.

The delay time is calculated by:

$$t_d = \frac{1024}{27 \mu s} = 37.74 \mu s \tag{8}$$

The time (t<sub>osc</sub>) is the sum of t<sub>d</sub> and t<sub>startup</sub>.



**Fig 23. Oscillator start-up time**



## 9. MFRC522 registers

### 9.1 Register bit behavior

Depending on the functionality of a register, the access conditions to the register can vary. In principle, bits with same behavior are grouped in common registers. The access conditions are described in [Table 19](#).

**Table 19. Behavior of register bits and their designation**

Abbreviation	Behavior	Description
R/W	read and write	These bits can be written and read by the microcontroller. Since they are used only for control purposes, their content is not influenced by internal state machines, for example the ComIEnReg register can be written and read by the microcontroller. It will also be read by internal state machines but never changed by them.
D	dynamic	These bits can be written and read by the microcontroller. Nevertheless, they can also be written automatically by internal state machines, for example the CommandReg register changes its value automatically after the execution of the command.
R	read only	These register bits hold values which are determined by internal states only, for example the CRCReady bit cannot be written externally but shows internal states.
W	write only	Reading these register bits always returns zero.
reserved	-	These registers are reserved for future use and must not be changed. In case of a write access, it is recommended to always write the value "0".
RFT	-	These register bits are reserved for future use or are for production tests and must not be changed.

## 9.2 Register overview

**Table 20. MFRC522 register overview**

Address (hex)	Register name	Function	Refer to
<b>Page 0: Command and status</b>			
00h	Reserved	reserved for future use	<a href="#">Table 21 on page 37</a>
01h	CommandReg	starts and stops command execution	<a href="#">Table 23 on page 37</a>
02h	ComIEnReg	enable and disable interrupt request control bits	<a href="#">Table 25 on page 37</a>
03h	DivIEnReg	enable and disable interrupt request control bits	<a href="#">Table 27 on page 38</a>
04h	ComIrqReg	interrupt request bits	<a href="#">Table 29 on page 38</a>
05h	DivIrqReg	interrupt request bits	<a href="#">Table 31 on page 39</a>
06h	ErrorReg	error bits showing the error status of the last command executed	<a href="#">Table 33 on page 40</a>
07h	Status1Reg	communication status bits	<a href="#">Table 35 on page 41</a>
08h	Status2Reg	receiver and transmitter status bits	<a href="#">Table 37 on page 42</a>
09h	FIFODataReg	input and output of 64 byte FIFO buffer	<a href="#">Table 39 on page 43</a>
0Ah	FIFOLevelReg	number of bytes stored in the FIFO buffer	<a href="#">Table 41 on page 43</a>
0Bh	WaterLevelReg	level for FIFO underflow and overflow warning	<a href="#">Table 43 on page 43</a>
0Ch	ControlReg	miscellaneous control registers	<a href="#">Table 45 on page 44</a>
0Dh	BitFramingReg	adjustments for bit-oriented frames	<a href="#">Table 47 on page 45</a>
0Eh	CollReg	bit position of the first bit-collision detected on the RF interface	<a href="#">Table 49 on page 45</a>
0Fh	Reserved	reserved for future use	<a href="#">Table 51 on page 46</a>
<b>Page 1: Command</b>			
10h	Reserved	reserved for future use	<a href="#">Table 53 on page 46</a>
11h	ModeReg	defines general modes for transmitting and receiving	<a href="#">Table 55 on page 47</a>
12h	TxModeReg	defines transmission data rate and framing	<a href="#">Table 57 on page 47</a>
13h	RxModeReg	defines reception data rate and framing	<a href="#">Table 59 on page 48</a>
14h	TxControlReg	controls the logical behavior of the antenna driver pins TX1 and TX2	<a href="#">Table 61 on page 49</a>
15h	TxASKReg	controls the setting of the transmission modulation	<a href="#">Table 63 on page 50</a>
16h	TxSelReg	selects the internal sources for the antenna driver	<a href="#">Table 65 on page 50</a>
17h	RxSelReg	selects internal receiver settings	<a href="#">Table 67 on page 51</a>
18h	RxThresholdReg	selects thresholds for the bit decoder	<a href="#">Table 69 on page 52</a>
19h	DemodReg	defines demodulator settings	<a href="#">Table 71 on page 52</a>
1Ah	Reserved	reserved for future use	<a href="#">Table 73 on page 53</a>
1Bh	Reserved	reserved for future use	<a href="#">Table 75 on page 53</a>
1Ch	MfTxReg	controls some MIFARE communication transmit parameters	<a href="#">Table 77 on page 54</a>
1Dh	MfRxReg	controls some MIFARE communication receive parameters	<a href="#">Table 79 on page 54</a>
1Eh	Reserved	reserved for future use	<a href="#">Table 81 on page 54</a>
1Fh	SerialSpeedReg	selects the speed of the serial UART interface	<a href="#">Table 83 on page 54</a>
<b>Page 2: Configuration</b>			
20h	Reserved	reserved for future use	<a href="#">Table 85 on page 56</a>

**Table 20. MFRC522 register overview ...continued**

Address (hex)	Register name	Function	Refer to
21h	CRCResultReg	shows the MSB and LSB values of the CRC calculation	<a href="#">Table 87 on page 56</a>
22h			<a href="#">Table 89 on page 56</a>
23h	Reserved	reserved for future use	<a href="#">Table 91 on page 57</a>
24h	ModWidthReg	controls the ModWidth setting	<a href="#">Table 93 on page 57</a>
25h	Reserved	reserved for future use	<a href="#">Table 95 on page 57</a>
26h	RFCfgReg	configures the receiver gain	<a href="#">Table 97 on page 58</a>
27h	GsNReg	selects the conductance of the antenna driver pins TX1 and TX2 for modulation	<a href="#">Table 99 on page 58</a>
28h	CWGsPReg	defines the conductance of the p-driver output during periods of no modulation	<a href="#">Table 101 on page 59</a>
29h	ModGsPReg	defines the conductance of the p-driver output during periods of modulation	<a href="#">Table 103 on page 59</a>
2Ah	TModeReg	defines settings for the internal timer	<a href="#">Table 105 on page 59</a>
2Bh	TPrescalerReg		<a href="#">Table 107 on page 60</a>
2Ch	TReloadReg	defines the 16-bit timer reload value	<a href="#">Table 109 on page 61</a>
2Dh			<a href="#">Table 111 on page 61</a>
2Eh	TCounterValReg	shows the 16-bit timer value	<a href="#">Table 113 on page 62</a>
2Fh			<a href="#">Table 115 on page 62</a>

**Page 3: Test register**

30h	Reserved	reserved for future use	<a href="#">Table 117 on page 62</a>
31h	TestSel1Reg	general test signal configuration	<a href="#">Table 119 on page 62</a>
32h	TestSel2Reg	general test signal configuration and PRBS control	<a href="#">Table 121 on page 63</a>
33h	TestPinEnReg	enables pin output driver on pins D1 to D7	<a href="#">Table 123 on page 63</a>
34h	TestPinValueReg	defines the values for D1 to D7 when it is used as an I/O bus	<a href="#">Table 125 on page 64</a>
35h	TestBusReg	shows the status of the internal test bus	<a href="#">Table 127 on page 64</a>
36h	AutoTestReg	controls the digital self test	<a href="#">Table 129 on page 65</a>
37h	VersionReg	shows the software version	<a href="#">Table 131 on page 65</a>
38h	AnalogTestReg	controls the pins AUX1 and AUX2	<a href="#">Table 133 on page 66</a>
39h	TestDAC1Reg	defines the test value for TestDAC1	<a href="#">Table 135 on page 67</a>
3Ah	TestDAC2Reg	defines the test value for TestDAC2	<a href="#">Table 137 on page 67</a>
3Bh	TestADCReg	shows the value of ADC I and Q channels	<a href="#">Table 139 on page 67</a>
3Ch to 3Fh	Reserved	reserved for production tests	<a href="#">Table 141 to Table 147 on page 68</a>

## 9.3 Register descriptions

### 9.3.1 Page 0: Command and status

#### 9.3.1.1 Reserved register 00h

Functionality is reserved for future use.

**Table 21. Reserved register (address 00h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 22. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	-	reserved

#### 9.3.1.2 CommandReg register

Starts and stops command execution.

**Table 23. CommandReg register (address 01h); reset value: 20h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol:	reserved		RcvOff	PowerDown	Command[3:0]			
Access:	-		R/W	D	D			

**Table 24. CommandReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 6	reserved	-	reserved for future use
5	RcvOff	1	analog part of the receiver is switched off
4	PowerDown	1	Soft power-down mode entered
		0	MFRC522 starts the wake up procedure during which this bit is read as a logic 1; it is read as a logic 0 when the MFRC522 is ready; see <a href="#">Section 8.6.2 on page 32</a> <b>Remark:</b> The PowerDown bit cannot be set when the SoftReset command is activated
3 to 0	Command[3:0]	-	activates a command based on the Command value; reading this register shows which command is executed; see <a href="#">Section 10.3 on page 69</a>

#### 9.3.1.3 ComIEnReg register

Control bits to enable and disable the passing of interrupt requests.

**Table 25. ComIEnReg register (address 02h); reset value: 80h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	IRqInv	TxIEn	RxIEn	IdleIEn	HiAlertIEn	LoAlertIEn	ErrIEn	TimerIEn
Access	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**Table 26. ComIEnReg register bit descriptions**

Bit	Symbol	Value	Description
7	IRqInv	1	signal on pin IRQ is inverted with respect to the Status1Reg register's IRq bit
		0	signal on pin IRQ is equal to the IRq bit; in combination with the DivIEnReg register's IRqPushPull bit, the default value of logic 1 ensures that the output level on pin IRQ is 3-state
6	TxIEn	-	allows the transmitter interrupt request (TxIRq bit) to be propagated to pin IRQ
5	RxIEn	-	allows the receiver interrupt request (RxIRq bit) to be propagated to pin IRQ
4	IdleIEn	-	allows the idle interrupt request (IdleIRq bit) to be propagated to pin IRQ
3	HiAlertIEn	-	allows the high alert interrupt request (HiAlertIRq bit) to be propagated to pin IRQ
2	LoAlertIEn	-	allows the low alert interrupt request (LoAlertIRq bit) to be propagated to pin IRQ
1	ErrIEn	-	allows the error interrupt request (ErrIRq bit) to be propagated to pin IRQ
0	TimerIEn	-	allows the timer interrupt request (TimerIRq bit) to be propagated to pin IRQ

#### 9.3.1.4 DivIEnReg register

Control bits to enable and disable the passing of interrupt requests.

**Table 27. DivIEnReg register (address 03h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	IRQPushPull	reserved	MfinActIEn	reserved	CRCIEn	reserved		
Access	R/W	-	R/W	-	R/W	-		

**Table 28. DivIEnReg register bit descriptions**

Bit	Symbol	Value	Description
7	IRQPushPull	1	pin IRQ is a standard CMOS output pin
		0	pin IRQ is an open-drain output pin
6 to 5	reserved	-	reserved for future use
4	MfinActIEn	-	allows the MFIN active interrupt request to be propagated to pin IRQ
3	reserved	-	reserved for future use
2	CRCIEn	-	allows the CRC interrupt request, indicated by the DivIrqReg register's CRCIRq bit, to be propagated to pin IRQ
1 to 0	reserved	-	reserved for future use

#### 9.3.1.5 ComIrqReg register

Interrupt request bits.

**Table 29. ComIrqReg register (address 04h); reset value: 14h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	Set1	TxIRq	RxIRq	IdleIRq	HiAlertIRq	LoAlertIRq	ErrIRq	TimerIRq
Access	W	D	D	D	D	D	D	D

**Table 30. ComlRqReg register bit descriptions**

All bits in the ComlRqReg register are cleared by software.

Bit	Symbol	Value	Description
7	Set1	1	indicates that the marked bits in the ComlRqReg register are set
		0	indicates that the marked bits in the ComlRqReg register are cleared
6	TxIRq	1	set immediately after the last bit of the transmitted data was sent out
5	RxIRq	1	receiver has detected the end of a valid data stream if the RxModeReg register's RxNoErr bit is set to logic 1, the RxIRq bit is only set to logic 1 when data bytes are available in the FIFO
4	IdleIRq	1	If a command terminates, for example, when the CommandReg changes its value from any command to the Idle command (see <a href="#">Table 149 on page 69</a> ) if an unknown command is started, the CommandReg register Command[3:0] value changes to the idle state and the IdleIRq bit is set The microcontroller starting the Idle command does not set the IdleIRq bit
3	HiAlertIRq	1	the Status1Reg register's HiAlert bit is set in opposition to the HiAlert bit, the HiAlertIRq bit stores this event and can only be reset as indicated by the Set1 bit in this register
2	LoAlertIRq	1	Status1Reg register's LoAlert bit is set in opposition to the LoAlert bit, the LoAlertIRq bit stores this event and can only be reset as indicated by the Set1 bit in this register
1	ErrIRq	1	any error bit in the ErrorReg register is set
0	TimerIRq	1	the timer decrements the timer value in register TCounterValReg to zero

### 9.3.1.6 DivlRqReg register

Interrupt request bits.

**Table 31. DivlRqReg register (address 05h); reset value: x0h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	Set2	reserved	MfinActIRq	reserved	CRCIRq	reserved	reserved	reserved
Access	W	-	D	-	D	-	-	-

**Table 32. DivlRqReg register bit descriptions**

All bits in the DivlRqReg register are cleared by software.

Bit	Symbol	Value	Description
7	Set2	1	indicates that the marked bits in the DivlRqReg register are set
		0	indicates that the marked bits in the DivlRqReg register are cleared
6 to 5	reserved	-	reserved for future use
4	MfinActIRq	1	MFIN is active this interrupt is set when either a rising or falling signal edge is detected
3	reserved	-	reserved for future use
2	CRCIRq	1	the CalcCRC command is active and all data is processed
1 to 0	reserved	-	reserved for future use

### 9.3.1.7 ErrorReg register

Error bit register showing the error status of the last command executed.

**Table 33. ErrorReg register (address 06h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	WrErr	TempErr	reserved	BufferOvfl	CollErr	CRCErr	ParityErr	ProtocolErr
Access	R	R	-	R	R	R	R	R

**Table 34. ErrorReg register bit descriptions**

Bit	Symbol	Value	Description
7	WrErr	1	data is written into the FIFO buffer by the host during the MFAuthent command or if data is written into the FIFO buffer by the host during the time between sending the last bit on the RF interface and receiving the last bit on the RF interface
6	TempErr <sup>[1]</sup>	1	internal temperature sensor detects overheating, in which case the antenna drivers are automatically switched off
5	reserved	-	reserved for future use
4	BufferOvfl	1	the host or a MFRC522's internal state machine (e.g. receiver) tries to write data to the FIFO buffer even though it is already full
3	CollErr	1	a bit-collision is detected cleared automatically at receiver start-up phase only valid during the bitwise anticollision at 106 kBd always set to logic 0 during communication protocols at 212 kBd, 424 kBd and 848 kBd
2	CRCErr	1	the RxModeReg register's RxCRCEn bit is set and the CRC calculation fails automatically cleared to logic 0 during receiver start-up phase
1	ParityErr	1	parity check failed automatically cleared during receiver start-up phase only valid for ISO/IEC 14443 A/MIFARE communication at 106 kBd
0	ProtocolErr	1	set to logic 1 if the SOF is incorrect automatically cleared during receiver start-up phase bit is only valid for 106 kBd during the MFAuthent command, the ProtocolErr bit is set to logic 1 if the number of bytes received in one data stream is incorrect

[1] Command execution clears all error bits except the TempErr bit. Cannot be set by software.

### 9.3.1.8 Status1Reg register

Contains status bits of the CRC, interrupt and FIFO buffer.

**Table 35. Status1Reg register (address 07h); reset value: 21h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved	CRCOk	CRCReady	IRq	TRunning	reserved	HiAlert	LoAlert
Access	-	R	R	R	R	-	R	R

**Table 36. Status1Reg register bit descriptions**

Bit	Symbol	Value	Description
7	reserved	-	reserved for future use
6	CRCOk	1	the CRC result is zero for data transmission and reception, the CRCOk bit is undefined: use the ErrorReg register's CRCErr bit indicates the status of the CRC coprocessor, during calculation the value changes to logic 0, when the calculation is done correctly the value changes to logic 1
5	CRCReady	1	the CRC calculation has finished only valid for the CRC coprocessor calculation using the CalcCRC command
4	IRq	-	indicates if any interrupt source requests attention with respect to the setting of the interrupt enable bits: see the ComIEnReg and DivIEnReg registers
3	TRunning	1	MFRC522's timer unit is running, i.e. the timer will decrement the TCounterValReg register with the next timer clock <b>Remark:</b> in gated mode, the TRunning bit is set to logic 1 when the timer is enabled by TModeReg register's TGated[1:0] bits; this bit is not influenced by the gated signal
2	reserved	-	reserved for future use
1	HiAlert	1	the number of bytes stored in the FIFO buffer corresponds to equation: $HiAlert = (64 - FIFOLength) \leq WaterLevel$ example: FIFO length = 60, WaterLevel = 4 → HiAlert = 1 FIFO length = 59, WaterLevel = 4 → HiAlert = 0
0	LoAlert	1	the number of bytes stored in the FIFO buffer corresponds to equation: $LoAlert = FIFOLength \leq WaterLevel$ example: FIFO length = 4, WaterLevel = 4 → LoAlert = 1 FIFO length = 5, WaterLevel = 4 → LoAlert = 0



**9.3.1.9 Status2Reg register**

Contains status bits of the receiver, transmitter and data mode detector.

**Table 37. Status2Reg register (address 08h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TempSensClear	I2CForceHS	reserved		MFCrypto1On	ModemState[2:0]		
Access	R/W	R/W	-		D	R		

**Table 38. Status2Reg register bit descriptions**

Bit	Symbol	Value	Description
7	TempSensClear	1	clears the temperature error if the temperature is below the alarm limit of 125 °C
6	I2CForceHS	1	I2C-bus input filter settings: the I2C-bus input filter is set to the High-speed mode independent of the I2C-bus protocol
		0	the I2C-bus input filter is set to the I2C-bus protocol used
5 to 4	reserved	-	reserved
3	MFCrypto1On	-	indicates that the MIFARE Crypto1 unit is switched on and therefore all data communication with the card is encrypted can only be set to logic 1 by a successful execution of the MFAuthent command only valid in Read/Write mode for MIFARE standard cards this bit is cleared by software
2 to 0	ModemState[2:0]	-	shows the state of the transmitter and receiver state machines:
		000	idle
		001	wait for the BitFramingReg register's StartSend bit
		010	TxWait: wait until RF field is present if the TModeReg register's TxWaitRF bit is set to logic 1 the minimum time for TxWait is defined by the TxWaitReg register
		011	transmitting
		100	RxWait: wait until RF field is present if the TModeReg register's TxWaitRF bit is set to logic 1 the minimum time for RxWait is defined by the RxWaitReg register
		101	wait for data
		110	receiving

### 9.3.1.10 FIFODataReg register

Input and output of 64 byte FIFO buffer.

**Table 39. FIFODataReg register (address 09h); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	FIFOData[7:0]							
Access	D							

**Table 40. FIFODataReg register bit descriptions**

Bit	Symbol	Description
7 to 0	FIFOData[7:0]	data input and output port for the internal 64-byte FIFO buffer FIFO buffer acts as parallel in/parallel out converter for all serial data stream inputs and outputs

### 9.3.1.11 FIFOLevelReg register

Indicates the number of bytes stored in the FIFO.

**Table 41. FIFOLevelReg register (address 0Ah); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	FlushBuffer	FIFOLevel[6:0]						
Access	W	R						

**Table 42. FIFOLevelReg register bit descriptions**

Bit	Symbol	Value	Description
7	FlushBuffer	1	immediately clears the internal FIFO buffer's read and write pointer and ErrorReg register's BufferOvfl bit reading this bit always returns 0
6 to 0	FIFOLevel [6:0]	-	indicates the number of bytes stored in the FIFO buffer writing to the FIFODataReg register increments and reading decrements the FIFOLevel value

### 9.3.1.12 WaterLevelReg register

Defines the level for FIFO under- and overflow warning.

**Table 43. WaterLevelReg register (address 0Bh); reset value: 08h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		WaterLevel[5:0]					
Access	-		R/W					

**Table 44. WaterLevelReg register bit descriptions**

Bit	Symbol	Description
7 to 6	reserved	reserved for future use
5 to 0	WaterLevel [5:0]	defines a warning level to indicate a FIFO buffer overflow or underflow: Status1Reg register's HiAlert bit is set to logic 1 if the remaining number of bytes in the FIFO buffer space is equal to, or less than the defined number of WaterLevel bytes Status1Reg register's LoAlert bit is set to logic 1 if equal to, or less than the WaterLevel bytes in the FIFO buffer <b>Remark:</b> to calculate values for HiAlert and LoAlert see <a href="#">Section 9.3.1.8 on page 41</a> .

**9.3.1.13 ControlReg register**

Miscellaneous control bits.

**Table 45. ControlReg register (address 0Ch); reset value: 10h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TStopNow	TStartNow	reserved			RxLastBits[2:0]		
Access	W	W		-			R	

**Table 46. ControlReg register bit descriptions**

Bit	Symbol	Value	Description
7	TStopNow	1	timer stops immediately reading this bit always returns it to logic 0
6	TStartNow	1	timer starts immediately reading this bit always returns it to logic 0
5 to 3	reserved	-	reserved for future use
2 to 0	RxLastBits[2:0]	-	indicates the number of valid bits in the last received byte if this value is 000b, the whole byte is valid

### 9.3.1.14 BitFramingReg register

Adjustments for bit-oriented frames.

**Table 47. BitFramingReg register (address 0Dh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	StartSend	RxAlign[2:0]			reserved	TxLastBits[2:0]		
Access	W	R/W			-	R/W		

**Table 48. BitFramingReg register bit descriptions**

Bit	Symbol	Value	Description
7	StartSend	1	starts the transmission of data only valid in combination with the Transceive command
6 to 4	RxAlign[2:0]		used for reception of bit-oriented frames: defines the bit position for the first bit received to be stored in the FIFO buffer example:
		0	LSB of the received bit is stored at bit position 0, the second received bit is stored at bit position 1
		1	LSB of the received bit is stored at bit position 1, the second received bit is stored at bit position 2
		7	LSB of the received bit is stored at bit position 7, the second received bit is stored in the next byte that follows at bit position 0  These bits are only to be used for bitwise anticollision at 106 kBd, for all other modes they are set to 0
3	reserved	-	reserved for future use
2 to 0	TxLastBits[2:0]	-	used for transmission of bit oriented frames: defines the number of bits of the last byte that will be transmitted 000b indicates that all bits of the last byte will be transmitted

### 9.3.1.15 CollReg register

Defines the first bit-collision detected on the RF interface.

**Table 49. CollReg register (address 0Eh); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	ValuesAfterColl	reserved	CollPosNotValid	CollPos[4:0]				
Access	R/W	-	R	R				

**Table 50. CollReg register bit descriptions**

Bit	Symbol	Value	Description
7	ValuesAfterColl	0	all received bits will be cleared after a collision only used during bitwise anticollision at 106 kBd, otherwise it is set to logic 1
6	reserved	-	reserved for future use
5	CollPosNotValid	1	no collision detected or the position of the collision is out of the range of CollPos[4:0]

**Table 50. CollReg register bit descriptions ...continued**

Bit	Symbol	Value	Description
4 to 0	CollPos[4:0]	-	shows the bit position of the first detected collision in a received frame only data bits are interpreted example:
		00h	indicates a bit-collision in the 32 <sup>nd</sup> bit
		01h	indicates a bit-collision in the 1 <sup>st</sup> bit
		08h	indicates a bit-collision in the 8 <sup>th</sup> bit
			These bits will only be interpreted if the CollPosNotValid bit is set to logic 0

**9.3.1.16 Reserved register 0Fh**

Functionality is reserved for future use.

**Table 51. Reserved register (address 0Fh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 52. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

**9.3.2 Page 1: Communication****9.3.2.1 Reserved register 10h**

Functionality is reserved for future use.

**Table 53. Reserved register (address 10h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 54. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

### 9.3.2.2 ModeReg register

Defines general mode settings for transmitting and receiving.

**Table 55. ModeReg register (address 11h); reset value: 3Fh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	MSBFirst	reserved	TxWaitRF	reserved	PolMFin	reserved	CRCPreset[1:0]	
Access	R/W	-	R/W	-	R/W	-	R/W	

**Table 56. ModeReg register bit descriptions**

Bit	Symbol	Value	Description
7	MSBFirst	1	CRC coprocessor calculates the CRC with MSB first in the CRCResultReg register the values for the CRCResultMSB[7:0] bits and the CRCResultLSB[7:0] bits are bit reversed <b>Remark:</b> during RF communication this bit is ignored
6	reserved	-	reserved for future use
5	TxWaitRF	1	transmitter can only be started if an RF field is generated
4	reserved	-	reserved for future use
3	PolMFin		defines the polarity of pin MFIN <b>Remark:</b> the internal envelope signal is encoded active LOW, changing this bit generates a MFinActIRq event
		1	polarity of pin MFIN is active HIGH
		0	polarity of pin MFIN is active LOW
2	reserved	-	reserved for future use
1 to 0	CRCPreset [1:0]		defines the preset value for the CRC coprocessor for the CalcCRC command <b>Remark:</b> during any communication, the preset values are selected automatically according to the definition of bits in the RxModeReg and TxModeReg registers
		00	0000h
		01	6363h
		10	A671h
		11	FFFFh

### 9.3.2.3 TxModeReg register

Defines the data rate during transmission.

**Table 57. TxModeReg register (address 12h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TxCRCEn	TxSpeed[2:0]			InvMod	reserved		
Access	R/W	D			R/W	-		

**Table 58. TxModeReg register bit descriptions**

Bit	Symbol	Value	Description
7	TxCRCEn	1	enables CRC generation during data transmission <b>Remark:</b> can only be set to logic 0 at 106 kBd
6 to 4	TxSpeed[2:0]		defines the bit rate during data transmission the MFRC522 handles transfer speeds up to 848 kBd
		000	106 kBd
		001	212 kBd
		010	424 kBd
		011	848 kBd
		100	reserved
		101	reserved
		110	reserved
		111	reserved
3	InvMod	1	modulation of transmitted data is inverted
2 to 0	reserved	-	reserved for future use

### 9.3.2.4 RxModeReg register

Defines the data rate during reception.

**Table 59. RxModeReg register (address 13h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	RxCRCEn	RxSpeed[2:0]			RxNoErr	RxMultiple	reserved	
Access	R/W	D			R/W	R/W	-	

**Table 60. RxModeReg register bit descriptions**

Bit	Symbol	Value	Description
7	RxCRCEn	1	enables the CRC calculation during reception <b>Remark:</b> can only be set to logic 0 at 106 kBd
6 to 4	RxSpeed[2:0]		defines the bit rate while receiving data the MFRC522 handles transfer speeds up to 848 kBd
		000	106 kBd
		001	212 kBd
		010	424 kBd
		011	848 kBd
		100	reserved
		101	reserved
		110	reserved
		111	reserved
3	RxNoErr	1	an invalid received data stream (less than 4 bits received) will be ignored and the receiver remains active

**Table 60. RxModeReg register bit descriptions ...continued**

Bit	Symbol	Value	Description
2	RxMultiple	0	receiver is deactivated after receiving a data frame
		1	able to receive more than one data frame only valid for data rates above 106 kBd in order to handle the polling command after setting this bit the Receive and Transceive commands will not terminate automatically. Multiple reception can only be deactivated by writing any command (except the Receive command) to the CommandReg register, or by the host clearing the bit if set to logic 1, an error byte is added to the FIFO buffer at the end of a received data stream which is a copy of the ErrorReg register value. For the MFRC522 version 2.0 the CRC status is reflected in the signal CRCOK, which indicates the actual status of the CRC coprocessor. For the MFRC522 version 1.0 the CRC status is reflected in the signal CRCErr.
1 to 0	reserved	-	reserved for future use

### 9.3.2.5 TxControlReg register

Controls the logical behavior of the antenna driver pins TX1 and TX2.

**Table 61. TxControlReg register (address 14h); reset value: 80h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	InvTx2RF On	InvTx1RF On	InvTx2RF Off	InvTx1RF Off	Tx2CW	reserved	Tx2RFEn	Tx1RFEn
Access	R/W	R/W	R/W	R/W	R/W	-	R/W	R/W

**Table 62. TxControlReg register bit descriptions**

Bit	Symbol	Value	Description
7	InvTx2RFOn	1	output signal on pin TX2 inverted when driver TX2 is enabled
6	InvTx1RFOn	1	output signal on pin TX1 inverted when driver TX1 is enabled
5	InvTx2RFOff	1	output signal on pin TX2 inverted when driver TX2 is disabled
4	InvTx1RFOff	1	output signal on pin TX1 inverted when driver TX1 is disabled
3	Tx2CW	1	output signal on pin TX2 continuously delivers the unmodulated 13.56 MHz energy carrier
		0	Tx2CW bit is enabled to modulate the 13.56 MHz energy carrier
2	reserved	-	reserved for future use
1	Tx2RFEn	1	output signal on pin TX2 delivers the 13.56 MHz energy carrier modulated by the transmission data
0	Tx1RFEn	1	output signal on pin TX1 delivers the 13.56 MHz energy carrier modulated by the transmission data



### 9.3.2.6 TxASKReg register

Controls transmit modulation settings.

**Table 63. TxASKReg register (address 15h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved	Force100ASK				reserved		
Access	-	R/W				-		

**Table 64. TxASKReg register bit descriptions**

Bit	Symbol	Value	Description
7	reserved	-	reserved for future use
6	Force100ASK	1	forces a 100 % ASK modulation independent of the ModGsPReg register setting
5 to 0	reserved	-	reserved for future use

### 9.3.2.7 TxSelReg register

Selects the internal sources for the analog module.

**Table 65. TxSelReg register (address 16h); reset value: 10h bit allocation**

Bit	7	6	5	4	3	2	1	0	
Symbol:	reserved		DriverSel[1:0]		MFOutSel[3:0]				
Access:	-		R/W		R/W				

**Table 66. TxSelReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 6	reserved	-	reserved for future use
5 to 4	DriverSel [1:0]	-	selects the input of drivers TX1 and TX2
		00	3-state; in soft power-down the drivers are only in 3-state mode if the DriverSel[1:0] value is set to 3-state mode
		01	modulation signal (envelope) from the internal encoder, Miller pulse encoded
		10	modulation signal (envelope) from pin MFIN
		11	HIGH; the HIGH level depends on the setting of bits InvTx1RFOOn/InvTx1RFOff and InvTx2RFOOn/InvTx2RFOff

**Table 66. TxSelReg register bit descriptions ...continued**

Bit	Symbol	Value	Description
3 to 0	MFOutSel [3:0]		selects the input for pin MFOUT
		0000	3-state
		0001	LOW
		0010	HIGH
		0011	test bus signal as defined by the TestSel1Reg register's TstBusBitSel[2:0] value
		0100	modulation signal (envelope) from the internal encoder, Miller pulse encoded
		0101	serial data stream to be transmitted, data stream before Miller encoder
		0110	reserved
		0111	serial data stream received, data stream after Manchester decoder
		1000 to 1111	reserved

**9.3.2.8 RxSelReg register**

Selects internal receiver settings.

**Table 67. RxSelReg register (address 17h); reset value: 84h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	UARTSel[1:0]				RxWait[5:0]			
Access	R/W				R/W			

**Table 68. RxSelReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 6	UARTSel [1:0]		selects the input of the contactless UART
		00	constant LOW
		01	Manchester with subcarrier from pin MFIN
		10	modulated signal from the internal analog module, default
		11	NRZ coding without subcarrier from pin MFIN which is only valid for transfer speeds above 106 kBd
5 to 0	RxWait [5:0]	-	after data transmission the activation of the receiver is delayed for RxWait bit-clocks, during this 'frame guard time' any signal on pin RX is ignored  this parameter is ignored by the Receive command  all other commands, such as Transceive, MFAuthent use this parameter  the counter starts immediately after the external RF field is switched on

### 9.3.2.9 RxThresholdReg register

Selects thresholds for the bit decoder.

**Table 69. RxThresholdReg register (address 18h); reset value: 84h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	MinLevel[3:0]			reserved		CollLevel[2:0]		
Access	R/W			-		R/W		

**Table 70. RxThresholdReg register bit descriptions**

Bit	Symbol	Description
7 to 4	MinLevel [3:0]	defines the minimum signal strength at the decoder input that will be accepted if the signal strength is below this level it is not evaluated
3	reserved	reserved for future use
2 to 0	CollLevel [2:0]	defines the minimum signal strength at the decoder input that must be reached by the weaker half-bit of the Manchester encoded signal to generate a bit-collision relative to the amplitude of the stronger half-bit

### 9.3.2.10 DemodReg register

Defines demodulator settings.

**Table 71. DemodReg register (address 19h); reset value: 4Dh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	AddIQ[1:0]		FixIQ	TPrescal Even	TauRcv[1:0]		TauSync[1:0]	
Access	R/W		R/W	R/W	R/W		R/W	

**Table 72. DemodReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 6	AddIQ [1:0]	-	defines the use of I and Q channel during reception <b>Remark:</b> the FixIQ bit must be set to logic 0 to enable the following settings:
		00	selects the stronger channel
		01	selects the stronger channel and freezes the selected channel during communication
		10	reserved
		11	reserved
5	FixIQ	1	if AddIQ[1:0] are set to X0b, the reception is fixed to I channel if AddIQ[1:0] are set to X1b, the reception is fixed to Q channel

**Table 72. DemodReg register bit descriptions ...continued**

Bit	Symbol	Value	Description
4	TPrescalEven	R/W	Available on RC522 version 1.0 and version 2.0: If set to logic 0 the following formula is used to calculate the timer frequency of the prescaler: $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 1)$ . Only available on version 2.0: If set to logic 1 the following formula is used to calculate the timer frequency of the prescaler: $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 2)$ . Default TPrescalEven bit is logic 0, find more information on the prescaler in <a href="#">Section 8.5</a> .
3 to 2	TauRcv[1:0]	-	changes the time-constant of the internal PLL during data reception <b>Remark:</b> if set to 00b the PLL is frozen during data reception
1 to 0	TauSync[1:0]	-	changes the time-constant of the internal PLL during burst

**9.3.2.11 Reserved register 1Ah**

Functionality is reserved for future use.

**Table 73. Reserved register (address 1Ah); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 74. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

**9.3.2.12 Reserved register 1Bh**

Functionality is reserved for future use.

**Table 75. Reserved register (address 1Bh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 76. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

**9.3.2.13 MfTxReg register**

Controls some MIFARE communication transmit parameters.

**Table 77. MfTxReg register (address 1Ch); reset value: 62h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved						TxWait[1:0]	
Access	-						R/W	

**Table 78. MfTxReg register bit descriptions**

Bit	Symbol	Description
7 to 2	reserved	reserved for future use
1 to 0	TxWait	defines the additional response time 7 bits are added to the value of the register bit by default

### 9.3.2.14 MfRxReg register

**Table 79. MfRxReg register (address 1Dh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved			ParityDisable	reserved			
Access	-			R/W	-			

**Table 80. MfRxReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 5	reserved	-	reserved for future use
4	ParityDisable	1	generation of the parity bit for transmission and the parity check for receiving is switched off the received parity bit is handled like a data bit
3 to 0	reserved	-	reserved for future use

### 9.3.2.15 Reserved register 1Eh

Functionality is reserved for future use.

**Table 81. Reserved register (address 1Eh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 82. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

### 9.3.2.16 SerialSpeedReg register

Selects the speed of the serial UART interface.

**Table 83. SerialSpeedReg register (address 1Fh); reset value: EBh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	BR_T0[2:0]				BR_T1[4:0]			
Access	R/W				R/W			

**Table 84. SerialSpeedReg register bit descriptions**

Bit	Symbol	Description
7 to 5	BR_T0[2:0]	factor BR_T0 adjusts the transfer speed: for description, see <a href="#">Section 8.1.3.2 on page 12</a>
4 to 0	BR_T1[4:0]	factor BR_T1 adjusts the transfer speed: for description, see <a href="#">Section 8.1.3.2 on page 12</a>

### 9.3.3 Page 2: Configuration

#### 9.3.3.1 Reserved register 20h

Functionality is reserved for future use.

**Table 85. Reserved register (address 20h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	-							
Access	reserved							

**Table 86. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

#### 9.3.3.2 CRCResultReg registers

Shows the MSB and LSB values of the CRC calculation.

**Remark:** The CRC is split into two 8-bit registers.

**Table 87. CRCResultReg (higher bits) register (address 21h); reset value: FFh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	CRCResultMSB[7:0]							
Access	R							

**Table 88. CRCResultReg register higher bit descriptions**

Bit	Symbol	Description
7 to 0	CRCResultMSB [7:0]	shows the value of the CRCResultReg register's most significant byte only valid if Status1Reg register's CRCReady bit is set to logic 1

**Table 89. CRCResultReg (lower bits) register (address 22h); reset value: FFh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	CRCResultLSB[7:0]							
Access	R							

**Table 90. CRCResultReg register lower bit descriptions**

Bit	Symbol	Description
7 to 0	CRCResultLSB [7:0]	shows the value of the least significant byte of the CRCResultReg register only valid if Status1Reg register's CRCReady bit is set to logic 1

### 9.3.3.3 Reserved register 23h

Functionality is reserved for future use.

**Table 91. Reserved register (address 23h); reset value: 88h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 92. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

### 9.3.3.4 ModWidthReg register

Sets the modulation width.

**Table 93. ModWidthReg register (address 24h); reset value: 26h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	ModWidth[7:0]							
Access	R/W							

**Table 94. ModWidthReg register bit descriptions**

Bit	Symbol	Description
7 to 0	ModWidth[7:0]	defines the width of the Miller modulation as multiples of the carrier frequency ( $\text{ModWidth} + 1 / f_{\text{clk}}$ ) the maximum value is half the bit period

### 9.3.3.5 Reserved register 25h

Functionality is reserved for future use.

**Table 95. Reserved register (address 25h); reset value: 87h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 96. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use



### 9.3.3.6 RFCfgReg register

Configures the receiver gain.

**Table 97. RFCfgReg register (address 26h); reset value: 48h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved	RxGain[2:0]			reserved			
Access	-	R/W			-			

**Table 98. RFCfgReg register bit descriptions**

Bit	Symbol	Value	Description
7	reserved	-	reserved for future use
6 to 4	RxGain [2:0]		defines the receiver's signal voltage gain factor:
		000	18 dB
		001	23 dB
		010	18 dB
		011	23 dB
		100	33 dB
		101	38 dB
		110	43 dB
		111	48 dB
3 to 0	reserved	-	reserved for future use

### 9.3.3.7 GsNReg register

Defines the conductance of the antenna driver pins TX1 and TX2 for the n-driver when the driver is switched on.

**Table 99. GsNReg register (address 27h); reset value: 88h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	CWGsN[3:0]				ModGsN[3:0]			
Access	R/W				R/W			

**Table 100. GsNReg register bit descriptions**

Bit	Symbol	Description
7 to 4	CWGsn [3:0]	<p>defines the conductance of the output n-driver during periods without modulation which can be used to regulate the output power and subsequently current consumption and operating distance</p> <p><b>Remark:</b> the conductance value is binary-weighted during soft Power-down mode the highest bit is forced to logic 1 value is only used if driver TX1 or TX2 is switched on</p>
3 to 0	ModGsN [3:0]	<p>defines the conductance of the output n-driver during periods without modulation which can be used to regulate the modulation index</p> <p><b>Remark:</b> the conductance value is binary weighted during soft Power-down mode the highest bit is forced to logic 1 value is only used if driver TX1 or TX2 is switched on</p>

### 9.3.3.8 CWGsPReg register

Defines the conductance of the p-driver output during periods of no modulation.

**Table 101. CWGsPReg register (address 28h); reset value: 20h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		CWGsP[5:0]					
Access	-		R/W					

**Table 102. CWGsPReg register bit descriptions**

Bit	Symbol	Description
7 to 6	reserved	reserved for future use
5 to 0	CWGSP[5:0]	defines the conductance of the p-driver output which can be used to regulate the output power and subsequently current consumption and operating distance <b>Remark:</b> the conductance value is binary weighted during soft Power-down mode the highest bit is forced to logic 1

### 9.3.3.9 ModGsPReg register

Defines the conductance of the p-driver output during modulation.

**Table 103. ModGsPReg register (address 29h); reset value: 20h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		ModGsP[5:0]					
Access	-		R/W					

**Table 104. ModGsPReg register bit descriptions**

Bit	Symbol	Description
7 to 6	reserved	reserved for future use
5 to 0	ModGsP[5:0]	defines the conductance of the p-driver output during modulation which can be used to regulate the modulation index <b>Remark:</b> the conductance value is binary weighted during soft Power-down mode the highest bit is forced to logic 1 if the TxASKReg register's Force100ASK bit is set to logic 1 the value of ModGsP has no effect

### 9.3.3.10 TModeReg and TPrescalerReg registers

These registers define the timer settings.

**Remark:** The TPrescaler setting higher 4 bits are in the TModeReg register and the lower 8 bits are in the TPrescalerReg register.

**Table 105. TModeReg register (address 2Ah); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TAuto	TGated[1:0]		TAutoRestart	TPrescaler_Hi[3:0]			
Access	R/W	R/W		R/W	R/W			

**Table 106. TModeReg register bit descriptions**

Bit	Symbol	Value	Description
7	TAuto	1	timer starts automatically at the end of the transmission in all communication modes at all speeds  if the RxModeReg register's RxMultiple bit is not set, the timer stops immediately after receiving the 5th bit (1 start bit, 4 data bits)  if the RxMultiple bit is set to logic 1 the timer never stops, in which case the timer can be stopped by setting the ControlReg register's TStopNow bit to logic 1
		0	indicates that the timer is not influenced by the protocol
6 to 5	TGated[1:0]		internal timer is running in gated mode  <b>Remark:</b> in gated mode, the Status1Reg register's TRunning bit is logic 1 when the timer is enabled by the TModeReg register's TGated[1:0] bits  this bit does not influence the gating signal
		00	non-gated mode
		01	gated by pin MFIN
		10	gated by pin AUX1
		11	-
4	TAutoRestart	1	timer automatically restarts its count-down from the 16-bit timer reload value instead of counting down to zero
		0	timer decrements to 0 and the ComlRqReg register's TimerIRq bit is set to logic 1
3 to 0	TPrescaler_Hi[3:0]	-	defines the higher 4 bits of the TPrescaler value  The following formula is used to calculate the timer frequency if the DemodReg register's TPrescalEven bit in Demot Regis set to logic 0:  $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 1)$ .  Where TPreScaler = [TPrescaler_Hi:TPrescaler_Lo] (TPrescaler value on 12 bits) (Default TPrescalEven bit is logic 0)  The following formula is used to calculate the timer frequency if the DemodReg register's TPrescalEven bit is set to logic 1:  $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 2)$ .  See <a href="#">Section 8.5 "Timer unit"</a> .

**Table 107. TPrescalerReg register (address 2Bh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TPrescaler_Lo[7:0]							
Access	R/W							

**Table 108. TPrescalerReg register bit descriptions**

Bit	Symbol	Description
7 to 0	TPrescaler_Lo[7:0]	<p>defines the lower 8 bits of the TPrescaler value</p> <p>The following formula is used to calculate the timer frequency if the DemodReg register's TPrescalEven bit is set to logic 0:</p> $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 1).$ <p>Where TPreScaler = [TPrescaler_Hi:TPrescaler_Lo] (TPrescaler value on 12 bits) (Default TPrescalEven bit is logic 0)</p> <p>The following formula is used to calculate the timer frequency if the DemodReg register's TPrescalEven bit in DemoReg is set to logic 1:</p> $f_{\text{timer}} = 13.56 \text{ MHz} / (2 * \text{TPreScaler} + 2).$ <p>See <a href="#">Section 8.5 "Timer unit"</a>.</p>

**9.3.3.11 TReloadReg register**

Defines the 16-bit timer reload value.

**Remark:** The reload value bits are contained in two 8-bit registers.

**Table 109. TReloadReg (higher bits) register (address 2Ch); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TReloadVal_Hi[7:0]							
Access	R/W							

**Table 110. TReloadReg register higher bit descriptions**

Bit	Symbol	Description
7 to 0	TReloadVal_Hi[7:0]	<p>defines the higher 8 bits of the 16-bit timer reload value</p> <p>on a start event, the timer loads the timer reload value</p> <p>changing this register affects the timer only at the next start event</p>

**Table 111. TReloadReg (lower bits) register (address 2Dh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TReloadVal_Lo[7:0]							
Access	R/W							

**Table 112. TReloadReg register lower bit descriptions**

Bit	Symbol	Description
7 to 0	TReloadVal_Lo[7:0]	<p>defines the lower 8 bits of the 16-bit timer reload value</p> <p>on a start event, the timer loads the timer reload value</p> <p>changing this register affects the timer only at the next start event</p>

**9.3.3.12 TCounterValReg register**

Contains the timer value.

**Remark:** The timer value bits are contained in two 8-bit registers.

**Table 113. TCounterValReg (higher bits) register (address 2Eh); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TCounterVal_Hi[7:0]							
Access	R							

**Table 114. TCounterValReg register higher bit descriptions**

Bit	Symbol	Description
7 to 0	TCounterVal_Hi [7:0]	timer value higher 8 bits

**Table 115. TCounterValReg (lower bits) register (address 2Fh); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TCounterVal_Lo[7:0]							
Access	R							

**Table 116. TCounterValReg register lower bit descriptions**

Bit	Symbol	Description
7 to 0	TCounterVal_Lo [7:0]	timer value lower 8 bits

### 9.3.4 Page 3: Test

#### 9.3.4.1 Reserved register 30h

Functionality is reserved for future use.

**Table 117. Reserved register (address 30h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved							
Access	-							

**Table 118. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for future use

#### 9.3.4.2 TestSel1Reg register

General test signal configuration.

**Table 119. TestSel1Reg register (address 31h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved					TstBusBitSel[2:0]		
Access	-					R/W		

**Table 120. TestSel1Reg register bit descriptions**

Bit	Symbol	Description
7 to 3	reserved	reserved for future use
2 to 0	TstBusBitSel [2:0]	selects a test bus signal which is output at pin MFOUT if AnalogSelAux2[3:0] = FFh in AnalogTestReg register, test bus signal is also output at pins AUX1 or AUX2

**9.3.4.3 TestSel2Reg register**

General test signal configuration and PRBS control.

**Table 121. TestSel2Reg register (address 32h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TstBusFlip	PRBS9	PRBS15	TestBusSel[4:0]				
Access	R/W	R/W	R/W	R/W				

**Table 122. TestSel2Reg register bit descriptions**

Bit	Symbol	Value	Description
7	TstBusFlip	1	test bus is mapped to the parallel port in the following order: TstBusBit4, TstBusBit3, TstBusBit2, TstBusBit6, TstBusBit5, TstBusBit0; see <a href="#">Section 16.1 on page 81</a>
6	PRBS9	-	starts and enables the PRBS9 sequence according to ITU-TO150 <b>Remark:</b> all relevant registers to transmit data must be configured before entering PRBS9 mode the data transmission of the defined sequence is started by the Transmit command
5	PRBS15	-	starts and enables the PRBS15 sequence according to ITU-TO150 <b>Remark:</b> all relevant registers to transmit data must be configured before entering PRBS15 mode the data transmission of the defined sequence is started by the Transmit command
4 to 0	TestBusSel[4:0]	-	selects the test bus; see <a href="#">Section 16.1 “Test signals”</a>

**9.3.4.4 TestPinEnReg register**

Enables the test bus pin output driver.

**Table 123. TestPinEnReg register (address 33h); reset value: 80h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	RS232LineEn	TestPinEn[5:0]					reserved	
Access	R/W	R/W					-	

**Table 124. TestPinEnReg register bit descriptions**

Bit	Symbol	Value	Description
7	RS232LineEn	0	serial UART lines MX and DTRQ are disabled
6 to 1	TestPinEn [5:0]	-	enables the output driver on one of the data pins D1 to D7 which outputs a test signal  <b>Example:</b> setting bit 1 to logic 1 enables pin D1 output setting bit 5 to logic 1 enables pin D5 output  <b>Remark:</b> If the SPI is used, only pins D1 to D4 can be used. If the serial UART interface is used and the RS232LineEn bit is set to logic 1 only pins D1 to D4 can be used.
0	reserved	-	reserved for future use

#### 9.3.4.5 TestPinValueReg register

Defines the HIGH and LOW values for the test port D1 to D7 when it is used as I/O.

**Table 125. TestPinValueReg register (address 34h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	UseIO	TestPinValue[5:0]						reserved
Access	R/W	R/W						-

**Table 126. TestPinValueReg register bit descriptions**

Bit	Symbol	Value	Description
7	UseIO	1	enables the I/O functionality for the test port when one of the serial interfaces is used  the input/output behavior is defined by value TestPinEn[5:0] in the TestPinEnReg register  the value for the output behavior is defined by TestPinValue[5:0]
6 to 1	TestPinValue [5:0]	-	defines the value of the test port when it is used as I/O and each output must be enabled by TestPinEn[5:0] in the TestPinEnReg register  <b>Remark:</b> Reading the register indicates the status of pins D6 to D1 if the UseIO bit is set to logic 1. If the UseIO bit is set to logic 0, the value of the TestPinValueReg register is read back.
0	reserved	-	reserved for future use

#### 9.3.4.6 TestBusReg register

Shows the status of the internal test bus.

**Table 127. TestBusReg register (address 35h); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	TestBus[7:0]							
Access	R							

**Table 128. TestBusReg register bit descriptions**

Bit	Symbol	Description
7 to 0	TestBus[7:0]	shows the status of the internal test bus the test bus is selected using the TestSel2Reg register; see <a href="#">Section 16.1 on page 81</a>

### 9.3.4.7 AutoTestReg register

Controls the digital self-test.

**Table 129. AutoTestReg register (address 36h); reset value: 40h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved	AmpRcv	RFT		SelfTest[3:0]			
Access	-	R/W	-		R/W			

**Table 130. AutoTestReg register bit descriptions**

Bit	Symbol	Value	Description
7	reserved	-	reserved for production tests
6	AmpRcv	1	internal signal processing in the receiver chain is performed non-linearly which increases the operating distance in communication modes at 106 kBd <b>Remark:</b> due to non-linearity, the effect of the RxThresholdReg register's MinLevel[3:0] and the CollLevel[2:0] values is also non-linear
5 to 4	RFT	-	reserved for production tests
3 to 0	SelfTest[3:0]	-	enables the digital self test the self test can also be started by the CalcCRC command; see <a href="#">Section 10.3.1.4 on page 70</a> the self test is enabled by value 1001b <b>Remark:</b> for default operation the self test must be disabled by value 0000b

### 9.3.4.8 VersionReg register

Shows the MFRC522 software version.

**Table 131. VersionReg register (address 37h); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	Version[7:0]							
Access	R							

**Table 132. VersionReg register bit descriptions**

Bit	Symbol	Description
7 to 4	Chiptype	'9' stands for MFRC522
3 to 0	Version	'1' stands for MFRC522 version 1.0 and '2' stands for MFRC522 version 2.0.

MFRC522 version 1.0 software version is: 91h.

MFRC522 version 2.0 software version is: 92h.



### 9.3.4.9 AnalogTestReg register

Determines the analog output test signal at, and status of, pins AUX1 and AUX2.

**Table 133. AnalogTestReg register (address 38h); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	AnalogSelAux1[3:0]				AnalogSelAux2[3:0]			
Access	R/W				R/W			

**Table 134. AnalogTestReg register bit descriptions**

Bit	Symbol	Value	Description
7 to 4	AnalogSelAux1 [3:0]		controls pin AUX1
		0000	3-state
		0001	output of TestDAC1 (AUX1), output of TestDAC2 (AUX2) <sup>[1]</sup>
		0010	test signal Corr1 <sup>[1]</sup>
		0011	reserved
		0100	DAC: test signal MinLevel <sup>[1]</sup>
		0101	DAC: test signal ADC_I <sup>[1]</sup>
		0110	DAC: test signal ADC_Q <sup>[1]</sup>
		0111	reserved
		1000	reserved, test signal for production test <sup>[1]</sup>
		1001	reserved
		1010	HIGH
		1011	LOW
		1100	TxActive: at 106 kBd: HIGH during Start bit, Data bit, Parity and CRC at 212 kBd: 424 kBd and 848 kBd: HIGH during data and CRC
		1101	RxActive: at 106 kBd: HIGH during Data bit, Parity and CRC at 212 kBd: 424 kBd and 848 kBd: HIGH during data and CRC
		1110	subcarrier detected: 106 kBd: not applicable 212 kBd: 424 kBd and 848 kBd: HIGH during last part of data and CRC
		1111	test bus bit as defined by the TestSel1Reg register's TstBusBitSel[2:0] bits  <b>Remark:</b> all test signals are described in <a href="#">Section 16.1 on page 81</a>
3 to 0	AnalogSelAux2 [3:0]	-	controls pin AUX2 (see bit descriptions for AUX1)

[1] **Remark:** Current source output; the use of 1 kΩ pull-down resistor on AUXn is recommended.

### 9.3.4.10 TestDAC1Reg register

Defines the test value for TestDAC1.

**Table 135. TestDAC1Reg register (address 39h); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		TestDAC1[5:0]					
Access	-		R/W					

**Table 136. TestDAC1Reg register bit descriptions**

Bit	Symbol	Description
7	reserved	reserved for production tests
6	reserved	reserved for future use
5 to 0	TestDAC1[5:0]	defines the test value for TestDAC1 output of DAC1 can be routed to AUX1 by setting value AnalogSelAux1[3:0] to 0001b in the AnalogTestReg register

### 9.3.4.11 TestDAC2Reg register

Defines the test value for TestDAC2.

**Table 137. TestDAC2Reg register (address 3Ah); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	reserved		TestDAC2[5:0]					
Access	-		R/W					

**Table 138. TestDAC2Reg register bit descriptions**

Bit	Symbol	Description
7 to 6	reserved	reserved for future use
5 to 0	TestDAC2[5:0]	defines the test value for TestDAC2 output of DAC2 can be routed to AUX2 by setting value AnalogSelAux2[3:0] to 0001b in the AnalogTestReg register

### 9.3.4.12 TestADCReg register

Shows the values of ADC I and Q channels.

**Table 139. TestADCReg register (address 3Bh); reset value: xxh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol	ADC_I[3:0]				ADC_Q[3:0]			
Access	R				R			

**Table 140. TestADCReg register bit descriptions**

Bit	Symbol	Description
7 to 4	ADC_I[3:0]	ADC I channel value
3 to 0	ADC_Q[3:0]	ADC Q channel value

### 9.3.4.13 Reserved register 3Ch

Functionality reserved for production test.

**Table 141. Reserved register (address 3Ch); reset value: FFh bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol					RFT			
Access					-			

**Table 142. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for production tests

**Table 143. Reserved register (address 3Dh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol					RFT			
Access					-			

**Table 144. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for production tests

**Table 145. Reserved register (address 3Eh); reset value: 03h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol					RFT			
Access					-			

**Table 146. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for production tests

**Table 147. Reserved register (address 3Fh); reset value: 00h bit allocation**

Bit	7	6	5	4	3	2	1	0
Symbol					reserved			
Access					-			

**Table 148. Reserved register bit descriptions**

Bit	Symbol	Description
7 to 0	reserved	reserved for production tests

## 10. MFRC522 command set

### 10.1 General description

The MFRC522 operation is determined by a state machine capable of performing a set of commands. A command is executed by writing a command code (see [Table 149](#)) to the CommandReg register.

Arguments and/or data necessary to process a command are exchanged via the FIFO buffer.

### 10.2 General behavior

- Each command that needs a data bit stream (or data byte stream) as an input immediately processes any data in the FIFO buffer. An exception to this rule is the Transceive command. Using this command, transmission is started with the BitFramingReg register's StartSend bit.
- Each command that needs a certain number of arguments, starts processing only when it has received the correct number of arguments from the FIFO buffer.
- The FIFO buffer is not automatically cleared when commands start. This makes it possible to write command arguments and/or the data bytes to the FIFO buffer and then start the command.
- Each command can be interrupted by the host writing a new command code to the CommandReg register, for example, the Idle command.

### 10.3 MFRC522 command overview

**Table 149. Command overview**

Command	Command code	Action
Idle	0000	no action, cancels current command execution
Mem	0001	stores 25 bytes into the internal buffer
Generate RandomID	0010	generates a 10-byte random ID number
CalcCRC	0011	activates the CRC coprocessor or performs a self test
Transmit	0100	transmits data from the FIFO buffer
NoCmdChange	0111	no command change, can be used to modify the CommandReg register bits without affecting the command, for example, the PowerDown bit
Receive	1000	activates the receiver circuits
Transceive	1100	transmits data from FIFO buffer to antenna and automatically activates the receiver after transmission
-	1101	reserved for future use
MFAuthent	1110	performs the MIFARE standard authentication as a reader
SoftReset	1111	resets the MFRC522

### 10.3.1 MFRC522 command descriptions

#### 10.3.1.1 Idle

Places the MFRC522 in Idle mode. The Idle command also terminates itself.

#### 10.3.1.2 Mem

Transfers 25 bytes from the FIFO buffer to the internal buffer.

To read out the 25 bytes from the internal buffer the Mem command must be started with an empty FIFO buffer. In this case, the 25 bytes are transferred from the internal buffer to the FIFO.

During a hard power-down (using pin NRSTPD), the 25 bytes in the internal buffer remain unchanged and are only lost if the power supply is removed from the MFRC522.

This command automatically terminates when finished and the Idle command becomes active.

#### 10.3.1.3 Generate RandomID

This command generates a 10-byte random number which is initially stored in the internal buffer. This then overwrites the 10 bytes in the internal 25-byte buffer. This command automatically terminates when finished and the MFRC522 returns to Idle mode.

#### 10.3.1.4 CalcCRC

The FIFO buffer content is transferred to the CRC coprocessor and the CRC calculation is started. The calculation result is stored in the CRCResultReg register. The CRC calculation is not limited to a dedicated number of bytes. The calculation is not stopped when the FIFO buffer is empty during the data stream. The next byte written to the FIFO buffer is added to the calculation.

The CRC preset value is controlled by the ModeReg register's CRCPreSet[1:0] bits. The value is loaded in to the CRC coprocessor when the command starts.

This command must be terminated by writing a command to the CommandReg register, such as, the Idle command.

If the AutoTestReg register's SelfTest[3:0] bits are set correctly, the MFRC522 enters Self Test mode. Starting the CalcCRC command initiates a digital self test. The result of the self test is written to the FIFO buffer.

#### 10.3.1.5 Transmit

The FIFO buffer content is immediately transmitted after starting this command. Before transmitting the FIFO buffer content, all relevant registers must be set for data transmission.

This command automatically terminates when the FIFO buffer is empty. It can be terminated by another command written to the CommandReg register.

#### 10.3.1.6 NoCmdChange

This command does not influence any running command in the CommandReg register. It can be used to manipulate any bit except the CommandReg register Command[3:0] bits, for example, the RcvOff bit or the PowerDown bit.

### 10.3.1.7 Receive

The MFRC522 activates the receiver path and waits for a data stream to be received. The correct settings must be chosen before starting this command.

This command automatically terminates when the data stream ends. This is indicated either by the end of frame pattern or by the length byte depending on the selected frame type and speed.

**Remark:** If the RxModeReg register's RxMultiple bit is set to logic 1, the Receive command will not automatically terminate. It must be terminated by starting another command in the CommandReg register.

### 10.3.1.8 Transceive

This command continuously repeats the transmission of data from the FIFO buffer and the reception of data from the RF field. The first action is transmit and after transmission the command is changed to receive a data stream.

Each transmit process must be started by setting the BitFramingReg register's StartSend bit to logic 1. This command must be cleared by writing any command to the CommandReg register.

**Remark:** If the RxModeReg register's RxMultiple bit is set to logic 1, the Transceive command never leaves the receive state because this state cannot be cancelled automatically.

### 10.3.1.9 MFAuthent

This command manages MIFARE authentication to enable a secure communication to any MIFARE Mini, MIFARE 1K and MIFARE 4K card. The following data is written to the FIFO buffer before the command can be activated:

- Authentication command code (60h, 61h)
- Block address
- Sector key byte 0
- Sector key byte 1
- Sector key byte 2
- Sector key byte 3
- Sector key byte 4
- Sector key byte 5
- Card serial number byte 0
- Card serial number byte 1
- Card serial number byte 2
- Card serial number byte 3

In total 12 bytes are written to the FIFO.

**Remark:** When the MFAuthent command is active all access to the FIFO buffer is blocked. However, if there is access to the FIFO buffer, the ErrorReg register's WrErr bit is set.

This command automatically terminates when the MIFARE card is authenticated and the Status2Reg register's MFCrypto1On bit is set to logic 1.

This command does not terminate automatically if the card does not answer, so the timer must be initialized to automatic mode. In this case, in addition to the IdleIRq bit, the TimerIRq bit can be used as the termination criteria. During authentication processing, the RxIRq bit and TxIRq bit are blocked. The Crypto1On bit is only valid after termination of the MFAuthent command, either after processing the protocol or writing Idle to the CommandReg register.

If an error occurs during authentication, the ErrorReg register's ProtocolErr bit is set to logic 1 and the Status2Reg register's Crypto1On bit is set to logic 0.

#### 10.3.1.10 SoftReset

This command performs a reset of the device. The configuration data of the internal buffer remains unchanged. All registers are set to the reset values. This command automatically terminates when finished.

**Remark:** The SerialSpeedReg register is reset and therefore the serial data rate is set to 9.6 kBd.

## 11. Limiting values

**Table 150. Limiting values**

In accordance with the Absolute Maximum Rating System (IEC 60134).

Symbol	Parameter	Conditions	Min	Max	Unit
V <sub>DDA</sub>	analog supply voltage		-0.5	+4.0	V
V <sub>DDD</sub>	digital supply voltage		-0.5	+4.0	V
V <sub>DD(PVDD)</sub>	PVDD supply voltage		-0.5	+4.0	V
V <sub>DD(TVDD)</sub>	TVDD supply voltage		-0.5	+4.0	V
V <sub>DD(SVDD)</sub>	SVDD supply voltage		-0.5	+4.0	V
V <sub>i</sub>	input voltage	all input pins except pins MFIN and RX	V <sub>SS(PVSS)</sub> - 0.5	V <sub>DD(PVDD)</sub> + 0.5	V
		pin MFIN	V <sub>SS(PVSS)</sub> - 0.5	V <sub>DD(SVDD)</sub> + 0.5	V
P <sub>tot</sub>	total power dissipation	per package; and V <sub>DDD</sub> in shortcut mode	-	200	mW
T <sub>j</sub>	junction temperature		-	100	°C
V <sub>ESD</sub>	electrostatic discharge voltage	HBM; 1500 Ω, 100 pF; JESD22-A114-B	-	2000	V
		MM; 0.75 μH, 200 pF; JESD22-A114-A	-	200	V

## 12. Recommended operating conditions

**Table 151. Operating conditions**

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V <sub>DDA</sub>	analog supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ; V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	[1][2] 2.5	3.3	3.6	V
V <sub>DDD</sub>	digital supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ; V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	[1][2] 2.5	3.3	3.6	V
V <sub>DD(TVDD)</sub>	TVDD supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ; V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	[1][2] 2.5	3.3	3.6	V
V <sub>DD(PVDD)</sub>	PVDD supply voltage	V <sub>DD(PVDD)</sub> ≤ V <sub>DDA</sub> = V <sub>DDD</sub> = V <sub>DD(TVDD)</sub> ; V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	[3] 1.6	1.8	3.6	V
V <sub>DD(SVDD)</sub>	SVDD supply voltage	V <sub>SSA</sub> = V <sub>SSD</sub> = V <sub>SS(PVSS)</sub> = V <sub>SS(TVSS)</sub> = 0 V	1.6	-	3.6	V
T <sub>amb</sub>	ambient temperature	HVQFN32	-25	-	+85	°C

[1] Supply voltages below 3 V reduce the performance (the achievable operating distance).

[2] V<sub>DDA</sub>, V<sub>DDD</sub> and V<sub>DD(TVDD)</sub> must always be the same voltage.

[3] V<sub>DD(PVDD)</sub> must always be the same or lower voltage than V<sub>DDD</sub>.

## 13. Thermal characteristics

**Table 152. Thermal characteristics**

Symbol	Parameter	Conditions	Package	Typ	Unit
R <sub>th(j-a)</sub>	thermal resistance from junction to ambient	in still air with exposed pin soldered on a 4 layer JEDEC PCB	HVQFN32	40	K/W



## 14. Characteristics

Table 153. Characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
<b>Input characteristics</b>						
<b>Pins EA, I2C and NRSTPD</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DD(PVDD)}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DD(PVDD)}$	V
<b>Pin MFIN</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DD(SVDD)}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DD(SVDD)}$	V
<b>Pin SDA</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DD(PVDD)}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DD(PVDD)}$	V
<b>Pin RX<sup>[1]</sup></b>						
$V_i$	input voltage		-1	-	$V_{DDA} + 1$	V
$C_i$	input capacitance	$V_{DDA} = 3\text{ V}$ ; receiver active; $V_{RX(p-p)} = 1\text{ V}$ ; 1.5 V (DC) offset	-	10	-	pF
$R_i$	input resistance	$V_{DDA} = 3\text{ V}$ ; receiver active; $V_{RX(p-p)} = 1\text{ V}$ ; 1.5 V (DC) offset	-	350	-	$\Omega$
<i>Input voltage range; see <a href="#">Figure 24</a></i>						
$V_{i(p-p)(min)}$	minimum peak-to-peak input voltage	Manchester encoded; $V_{DDA} = 3\text{ V}$	-	100	-	mV
$V_{i(p-p)(max)}$	maximum peak-to-peak input voltage	Manchester encoded; $V_{DDA} = 3\text{ V}$	-	4	-	V
<i>Input sensitivity; see <a href="#">Figure 24</a></i>						
$V_{mod}$	modulation voltage	minimum Manchester encoded; $V_{DDA} = 3\text{ V}$ ; $RxGain[2:0] = 111\text{b}$ (48 dB)	-	5	-	mV
<b>Pin OSCIN</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DDA}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DDA}$	V
$C_i$	input capacitance	$V_{DDA} = 2.8\text{ V}$ ; DC = 0.65 V; AC = 1 V (p-p)	-	2	-	pF
<b>Input/output characteristics</b>						
<b>pins D1, D2, D3, D4, D5, D6 and D7</b>						
$I_{LI}$	input leakage current		-1	-	+1	$\mu\text{A}$
$V_{IH}$	HIGH-level input voltage		$0.7V_{DD(PVDD)}$	-	-	V
$V_{IL}$	LOW-level input voltage		-	-	$0.3V_{DD(PVDD)}$	V

Table 153. Characteristics ...continued

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DD(PVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>DD(PVDD)</sub> - 0.4	-	V <sub>DD(PVDD)</sub>	V
V <sub>OL</sub>	LOW-level output voltage	V <sub>DD(PVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>SS(PVSS)</sub>	-	V <sub>SS(PVSS)</sub> + 0.4	V
I <sub>OH</sub>	HIGH-level output current	V <sub>DD(PVDD)</sub> = 3 V	-	-	4	mA
I <sub>OL</sub>	LOW-level output current	V <sub>DD(PVDD)</sub> = 3 V	-	-	4	mA
<b>Output characteristics</b>						
<b>Pin MFOUT</b>						
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DD(SVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>DD(SVDD)</sub> - 0.4	-	V <sub>DD(SVDD)</sub>	V
V <sub>OL</sub>	LOW-level output voltage	V <sub>DD(SVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>SS(PVSS)</sub>	-	V <sub>SS(PVSS)</sub> + 0.4	V
I <sub>OL</sub>	LOW-level output current	V <sub>DD(SVDD)</sub> = 3 V	-	-	4	mA
I <sub>OH</sub>	HIGH-level output current	V <sub>DD(SVDD)</sub> = 3 V	-	-	4	mA
<b>Pin IRQ</b>						
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DD(PVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>DD(PVDD)</sub> - 0.4	-	V <sub>DD(PVDD)</sub>	V
V <sub>OL</sub>	LOW-level output voltage	V <sub>DD(PVDD)</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>SS(PVSS)</sub>	-	V <sub>SS(PVSS)</sub> + 0.4	V
I <sub>OL</sub>	LOW-level output current	V <sub>DD(PVDD)</sub> = 3 V	-	-	4	mA
I <sub>OH</sub>	HIGH-level output current	V <sub>DD(PVDD)</sub> = 3 V	-	-	4	mA
<b>Pins AUX1 and AUX2</b>						
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DDD</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>DDD</sub> - 0.4	-	V <sub>DDD</sub>	V
V <sub>OL</sub>	LOW-level output voltage	V <sub>DDD</sub> = 3 V; I <sub>O</sub> = 4 mA	V <sub>SS(PVSS)</sub>	-	V <sub>SS(PVSS)</sub> + 0.4	V
I <sub>OL</sub>	LOW-level output current	V <sub>DDD</sub> = 3 V	-	-	4	mA
I <sub>OH</sub>	HIGH-level output current	V <sub>DDD</sub> = 3 V	-	-	4	mA
<b>Pins TX1 and TX2</b>						
V <sub>OH</sub>	HIGH-level output voltage	V <sub>DD(TVDD)</sub> = 3 V; I <sub>DD(TVDD)</sub> = 32 mA; CWGsP[5:0] = 3Fh	V <sub>DD(TVDD)</sub> - 0.15	-	-	V
		V <sub>DD(TVDD)</sub> = 3 V; I <sub>DD(TVDD)</sub> = 80 mA; CWGsP[5:0] = 3Fh	V <sub>DD(TVDD)</sub> - 0.4	-	-	V
		V <sub>DD(TVDD)</sub> = 2.5 V; I <sub>DD(TVDD)</sub> = 32 mA; CWGsP[5:0] = 3Fh	V <sub>DD(TVDD)</sub> - 0.24	-	-	V
		V <sub>DD(TVDD)</sub> = 2.5 V; I <sub>DD(TVDD)</sub> = 80 mA; CWGsP[5:0] = 3Fh	V <sub>DD(TVDD)</sub> - 0.64	-	-	V

Table 153. Characteristics ...continued

Symbol	Parameter	Conditions	Min	Typ	Max	Unit	
V <sub>OL</sub>	LOW-level output voltage	V <sub>DD(TVDD)</sub> = 3 V; I <sub>DD(TVDD)</sub> = 32 mA; CWGsP[5:0] = 0Fh	-	-	0.15	V	
		V <sub>DD(TVDD)</sub> = 3 V; I <sub>DD(TVDD)</sub> = 80 mA; CWGsP[5:0] = 0Fh	-	-	0.4	V	
		V <sub>DD(TVDD)</sub> = 2.5 V; I <sub>DD(TVDD)</sub> = 32 mA; CWGsP[5:0] = 0Fh	-	-	0.24	V	
		V <sub>DD(TVDD)</sub> = 2.5 V; I <sub>DD(TVDD)</sub> = 80 mA; CWGsP[5:0] = 0Fh	-	-	0.64	V	
<b>Current consumption</b>							
I <sub>pd</sub>	power-down current	V <sub>D<sub>DA</sub></sub> = V <sub>D<sub>DD</sub></sub> = V <sub>DD(TVDD)</sub> = V <sub>DD(PVDD)</sub> = 3 V					
		hard power-down; pin NRSTPD set LOW	[2]	-	-	5	μA
		soft power-down; RF level detector on	[2]	-	-	10	μA
I <sub>DD</sub>	digital supply current	pin DVDD; V <sub>D<sub>DD</sub></sub> = 3 V	-	6.5	9	mA	
I <sub>D<sub>DA</sub></sub>	analog supply current	pin AVDD; V <sub>D<sub>DA</sub></sub> = 3 V; CommandReg register's bit RcvOff = 0	-	7	10	mA	
		pin AVDD; receiver switched off; V <sub>D<sub>DA</sub></sub> = 3 V; CommandReg register's bit RcvOff = 1	-	3	5	mA	
I <sub>DD(PVDD)</sub>	PVDD supply current	pin PVDD	[3]	-	40	mA	
I <sub>DD(TVDD)</sub>	TVDD supply current	pin TVDD; continuous wave	[4][5][6]	-	60	100	mA
I <sub>DD(SVDD)</sub>	SVDD supply current	pin SVDD	[7]	-	4	mA	
<b>Clock frequency</b>							
f <sub>clk</sub>	clock frequency		-	27.12	-	MHz	
δ <sub>clk</sub>	clock duty cycle		40	50	60	%	
t <sub>jit</sub>	jitter time	RMS	-	-	10	ps	
<b>Crystal oscillator</b>							
V <sub>OH</sub>	HIGH-level output voltage	pin OSCOUT	-	1.1	-	V	
V <sub>OL</sub>	LOW-level output voltage	pin OSCOUT	-	0.2	-	V	
C <sub>i</sub>	input capacitance	pin OSCOUT	-	2	-	pF	
		pin OSCIN	-	2	-	pF	
<b>Typical input requirements</b>							
f <sub>xtal</sub>	crystal frequency		-	27.12	-	MHz	
ESR	equivalent series resistance		-	-	100	Ω	
C <sub>L</sub>	load capacitance		-	10	-	pF	
P <sub>xtal</sub>	crystal power dissipation		-	50	100	mW	

[1] The voltage on pin RX is clamped by internal diodes to pins AVSS and AVDD.

- [2]  $I_{pd}$  is the total current for all supplies.
- [3]  $I_{DD(PVDD)}$  depends on the overall load at the digital pins.
- [4]  $I_{DD(TVDD)}$  depends on  $V_{DD(TVDD)}$  and the external circuit connected to pins TX1 and TX2.
- [5] During typical circuit operation, the overall current is below 100 mA.
- [6] Typical value using a complementary driver configuration and an antenna matched to  $40 \Omega$  between pins TX1 and TX2 at 13.56 MHz.
- [7]  $I_{DD(SVDD)}$  depends on the load at pin MFOUT.

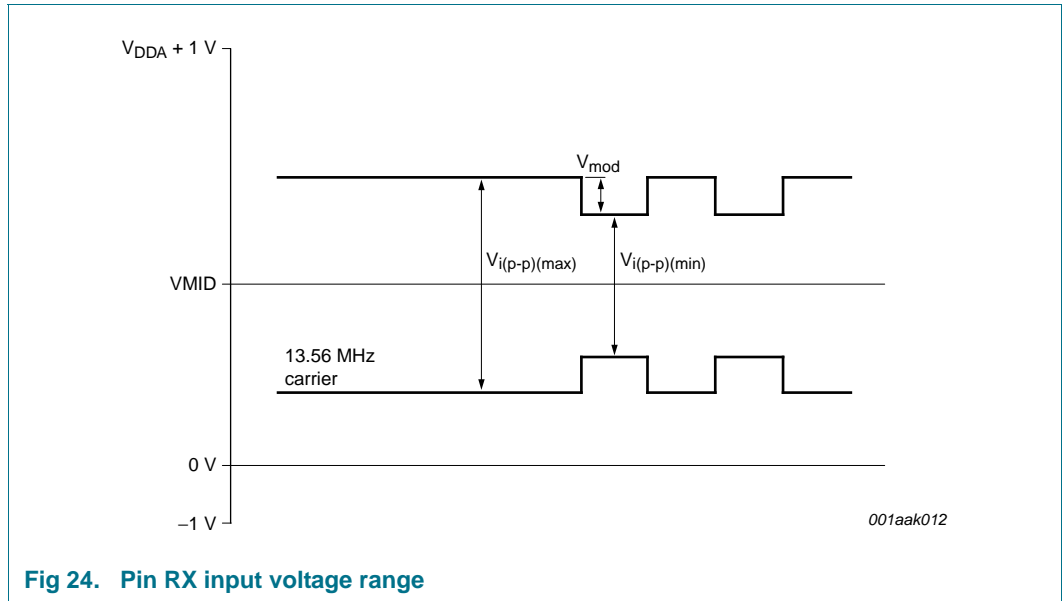


Fig 24. Pin RX input voltage range

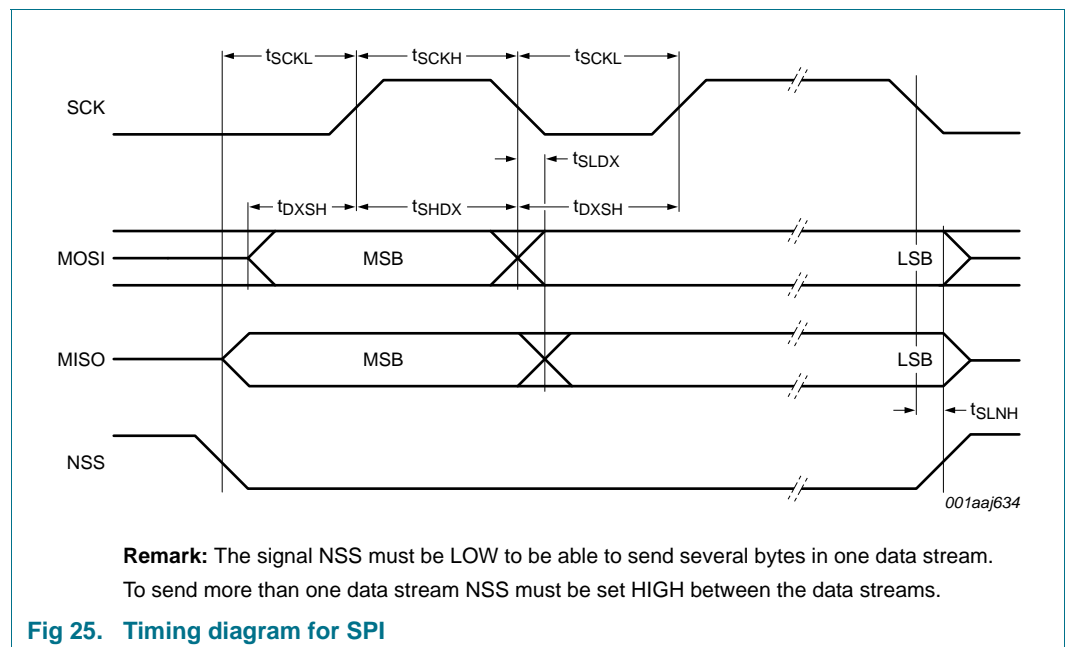
## 14.1 Timing characteristics

Table 154. SPI timing characteristics

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
$t_{WL}$	pulse width LOW	line SCK	50	-	-	ns
$t_{WH}$	pulse width HIGH	line SCK	50	-	-	ns
$t_{h(SCKH-D)}$	SCK HIGH to data input hold time	SCK to changing MOSI	25	-	-	ns
$t_{su(D-SCKH)}$	data input to SCK HIGH set-up time	changing MOSI to SCK	25	-	-	ns
$t_{h(SCKL-Q)}$	SCK LOW to data output hold time	SCK to changing MISO	-	-	25	ns
$t_{(SCKL-NSSH)}$	SCK LOW to NSS HIGH time		0	-	-	ns
$t_{NHNL}$	NSS high before communication		50	-	-	ns

Table 155. I<sup>2</sup>C-bus timing in Fast mode

Symbol	Parameter	Conditions	Fast mode		High-speed mode		Unit
			Min	Max	Min	Max	
f <sub>SCL</sub>	SCL clock frequency		0	400	0	3400	kHz
t <sub>HD;STA</sub>	hold time (repeated) START condition	after this period, the first clock pulse is generated	600	-	160	-	ns
t <sub>SU;STA</sub>	set-up time for a repeated START condition		600	-	160	-	ns
t <sub>SU;STO</sub>	set-up time for STOP condition		600	-	160	-	ns
t <sub>LOW</sub>	LOW period of the SCL clock		1300	-	160	-	ns
t <sub>HIGH</sub>	HIGH period of the SCL clock		600	-	60	-	ns
t <sub>HD;DAT</sub>	data hold time		0	900	0	70	ns
t <sub>SU;DAT</sub>	data set-up time		100	-	10	-	ns
t <sub>r</sub>	rise time	SCL signal	20	300	10	40	ns
t <sub>f</sub>	fall time	SCL signal	20	300	10	40	ns
t <sub>r</sub>	rise time	SDA and SCL signals	20	300	10	80	ns
t <sub>f</sub>	fall time	SDA and SCL signals	20	300	10	80	ns
t <sub>BUF</sub>	bus free time between a STOP and START condition		1.3	-	1.3	-	μs



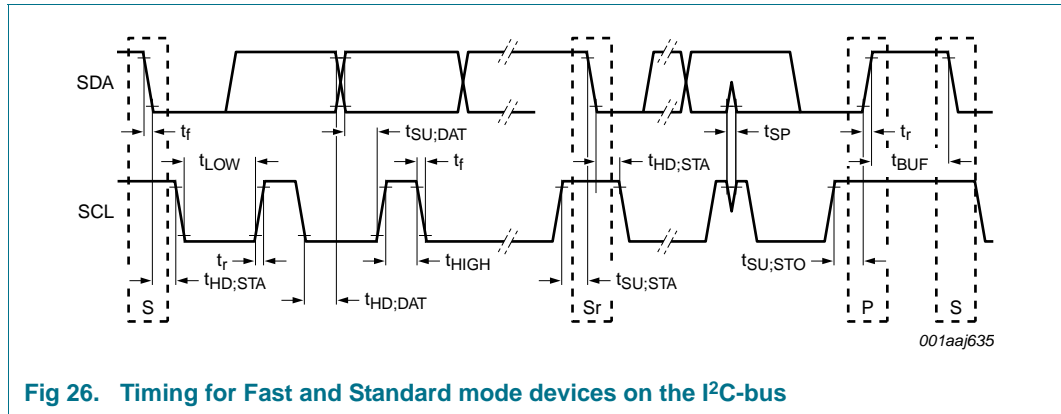
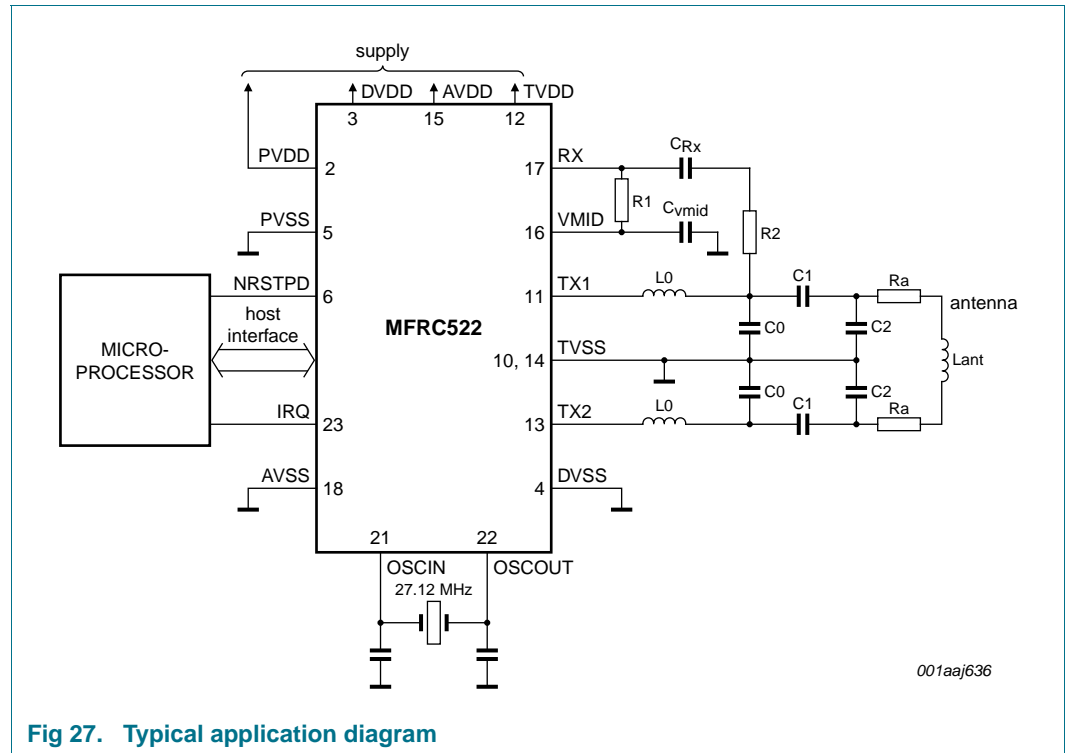


Fig 26. Timing for Fast and Standard mode devices on the I<sup>2</sup>C-bus

## 15. Application information

A typical application diagram using a complementary antenna connection to the MFRC522 is shown in [Figure 27](#).

The antenna tuning and RF part matching is described in the application note [Ref. 1](#) and [Ref. 2](#).



**Fig 27. Typical application diagram**

## 16. Test information

### 16.1 Test signals

#### 16.1.1 Self test

The MFRC522 has the capability to perform a digital self test. The self test is started by using the following procedure:

1. Perform a soft reset.
2. Clear the internal buffer by writing 25 bytes of 00h and implement the Config command.
3. Enable the self test by writing 09h to the AutoTestReg register.
4. Write 00h to the FIFO buffer.
5. Start the self test with the CalcCRC command.
6. The self test is initiated.
7. When the self test has completed, the FIFO buffer contains the following 64 bytes:

FIFO buffer byte values for MFRC522 version 1.0:

00h, C6h, 37h, D5h, 32h, B7h, 57h, 5Ch,  
C2h, D8h, 7Ch, 4Dh, D9h, 70h, C7h, 73h,  
10h, E6h, D2h, AAh, 5Eh, A1h, 3Eh, 5Ah,  
14h, AFh, 30h, 61h, C9h, 70h, DBh, 2Eh,  
64h, 22h, 72h, B5h, BDh, 65h, F4h, ECh,  
22h, BCh, D3h, 72h, 35h, CDh, AAh, 41h,  
1Fh, A7h, F3h, 53h, 14h, DEh, 7Eh, 02h,  
D9h, 0Fh, B5h, 5Eh, 25h, 1Dh, 29h, 79h

FIFO buffer byte values for MFRC522 version 2.0:

00h, EBh, 66h, BAh, 57h, BFh, 23h, 95h,  
D0h, E3h, 0Dh, 3Dh, 27h, 89h, 5Ch, DEh,  
9Dh, 3Bh, A7h, 00h, 21h, 5Bh, 89h, 82h,  
51h, 3Ah, EBh, 02h, 0Ch, A5h, 00h, 49h,  
7Ch, 84h, 4Dh, B3h, CCh, D2h, 1Bh, 81h,  
5Dh, 48h, 76h, D5h, 71h, 061h, 21h, A9h,  
86h, 96h, 83h, 38h, CFh, 9Dh, 5Bh, 6Dh,  
DCh, 15h, BAh, 3Eh, 7Dh, 95h, 03Bh, 2Fh

#### 16.1.2 Test bus

The test bus is used for production tests. The following configuration can be used to improve the design of a system using the MFRC522. The test bus allows internal signals to be routed to the digital interface. The test bus comprises two sets of test signals which are selected using their subaddress specified in the TestSel2Reg register's TestBusSel[4:0] bits. The test signals and their related digital output pins are described in [Table 156](#) and [Table 157](#).



**Table 156. Test bus signals: TestBusSel[4:0] = 07h**

Pins	Internal signal name	Description
D6	s_data	received data stream
D5	s_coll	bit-collision detected (106 kBd only)
D4	s_valid	s_data and s_coll signals are valid
D3	s_over	receiver has detected a stop condition
D2	RCV_reset	receiver is reset
D1	-	reserved

**Table 157. Test bus signals: TestBusSel[4:0] = 0Dh**

Pins	Internal test signal name	Description
D6	clkstable	oscillator output signal
D5	clk27/8	oscillator output signal divided by 8
D4 to D3	-	reserved
D2	clk27	oscillator output signal
D1	-	reserved

### 16.1.3 Test signals on pins AUX1 or AUX2

The MFRC522 allows the user to select internal signals for measurement on pins AUX1 or AUX2. These measurements can be helpful during the design-in phase to optimize the design or used for test purposes.

[Table 158](#) shows the signals that can be switched to pin AUX1 or AUX2 by setting AnalogSelAux1[3:0] or AnalogSelAux2[3:0] in the AnalogTestReg register.

**Remark:** The DAC has a current output, therefore it is recommended that a 1 k $\Omega$  pull-down resistor is connected to pin AUX1 or AUX2.

**Table 158. Test signal descriptions**

AnalogSelAux1[3:0] or AnalogSelAux2[3:0] value	Signal on pin AUX1 or pin AUX2
0000	3-state
0001	DAC: register TestDAC1 or TestDAC2
0010	DAC: test signal Corr1
0011	reserved
0100	DAC: test signal MinLevel
0101	DAC: test signal ADC_I
0110	DAC: test signal ADC_Q
0111 to 1001	reserved
1010	HIGH
1011	LOW
1100	TxActive

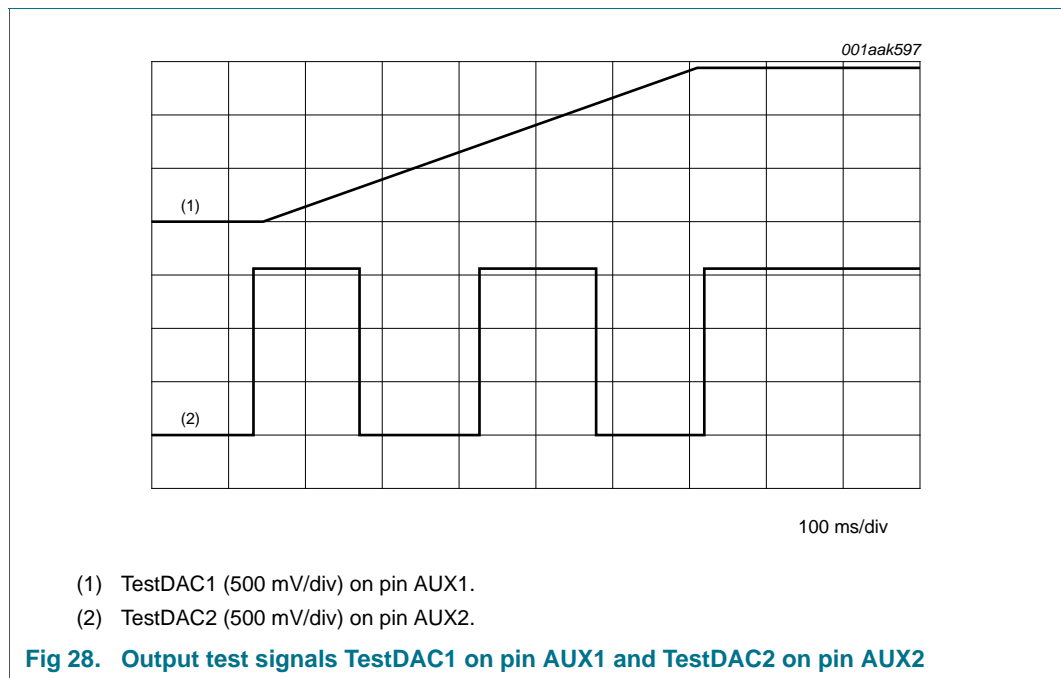
Table 158. Test signal descriptions ...continued

AnalogSelAux1[3:0] or AnalogSelAux2[3:0] value	Signal on pin AUX1 or pin AUX2
1101	RxActive
1110	subcarrier detected
1111	TstBusBit

16.1.3.1 Example: Output test signals TestDAC1 and TestDAC2

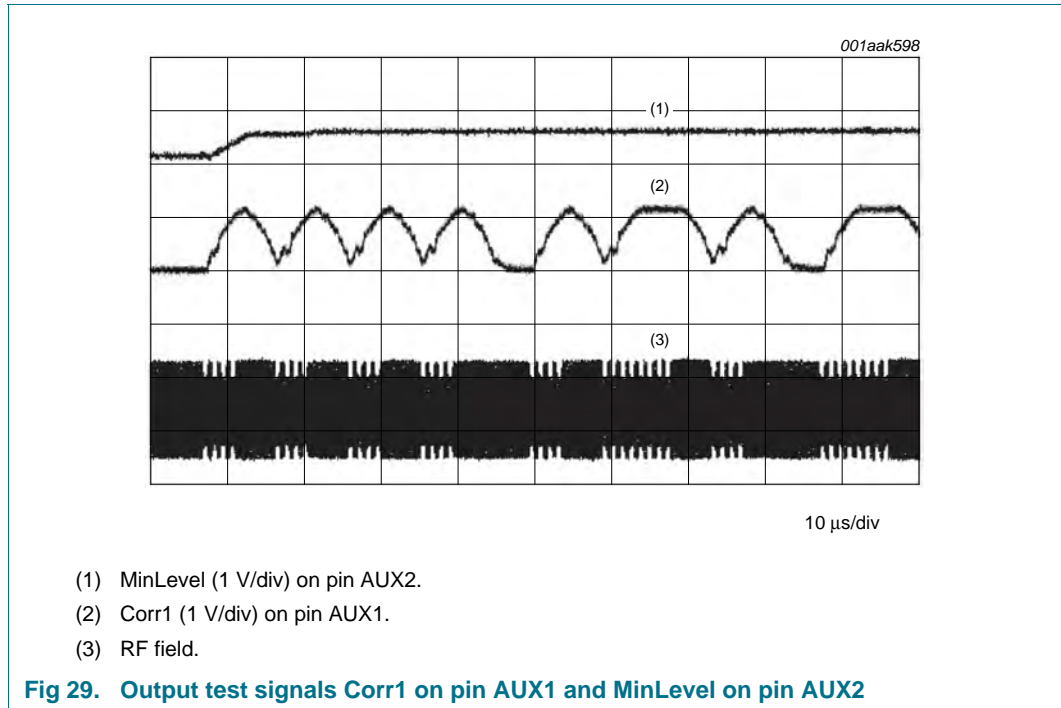
The AnalogTestReg register is set to 11h. The output on pin AUX1 has the test signal TestDAC1 and the output on pin AUX2 has the test signal TestDAC2. The signal values of TestDAC1 and TestDAC2 are controlled by the TestDAC1Reg and TestDAC2Reg registers.

Figure 28 shows test signal TestDAC1 on pin AUX1 and TestDAC2 on pin AUX2 when the TestDAC1Reg register is programmed with a slope defined by values 00h to 3Fh and the TestDAC2Reg register is programmed with a rectangular signal defined by values 00h and 3Fh.



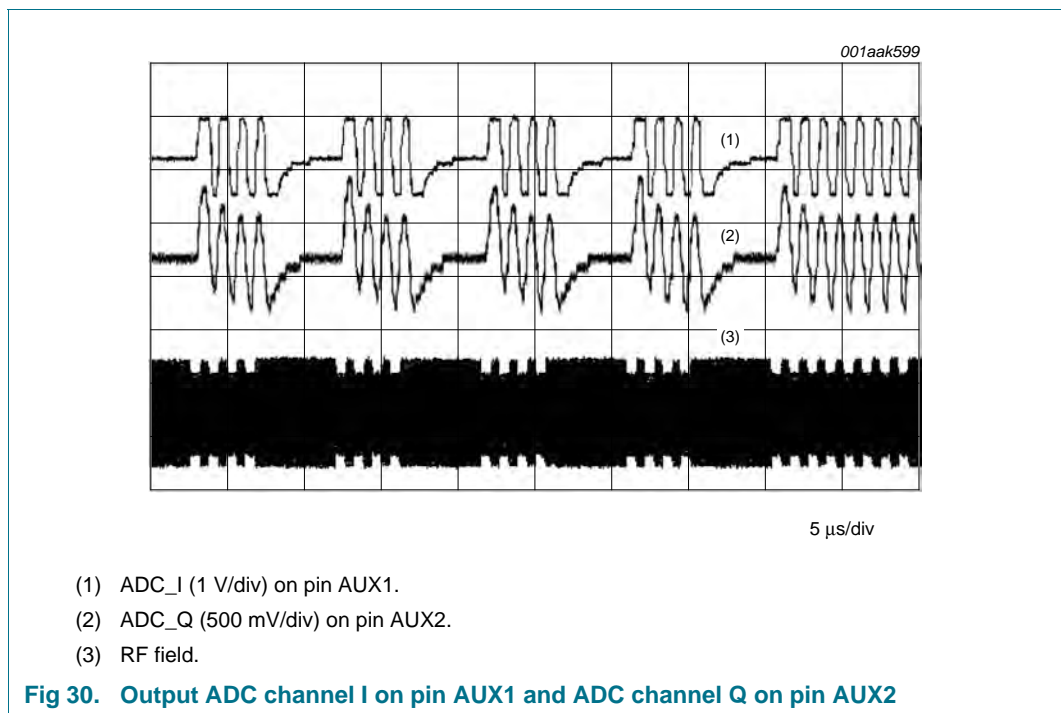
16.1.3.2 Example: Output test signals Corr1 and MinLevel

Figure 29 shows test signals Corr1 and MinLevel on pins AUX1 and AUX2, respectively. The AnalogTestReg register is set to 24h.



16.1.3.3 Example: Output test signals ADC channel I and ADC channel Q

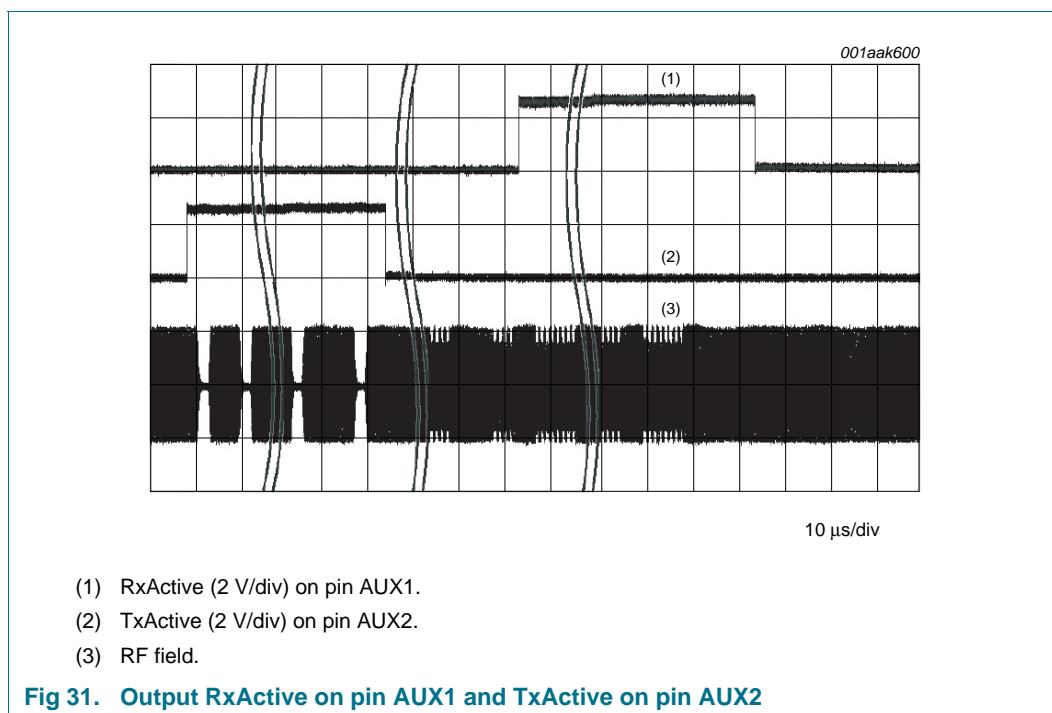
Figure 30 shows the channel behavior test signals ADC\_I and ADC\_Q on pins AUX1 and AUX2, respectively. The AnalogTestReg register is set to 56h.



16.1.3.4 Example: Output test signals RxActive and TxActive

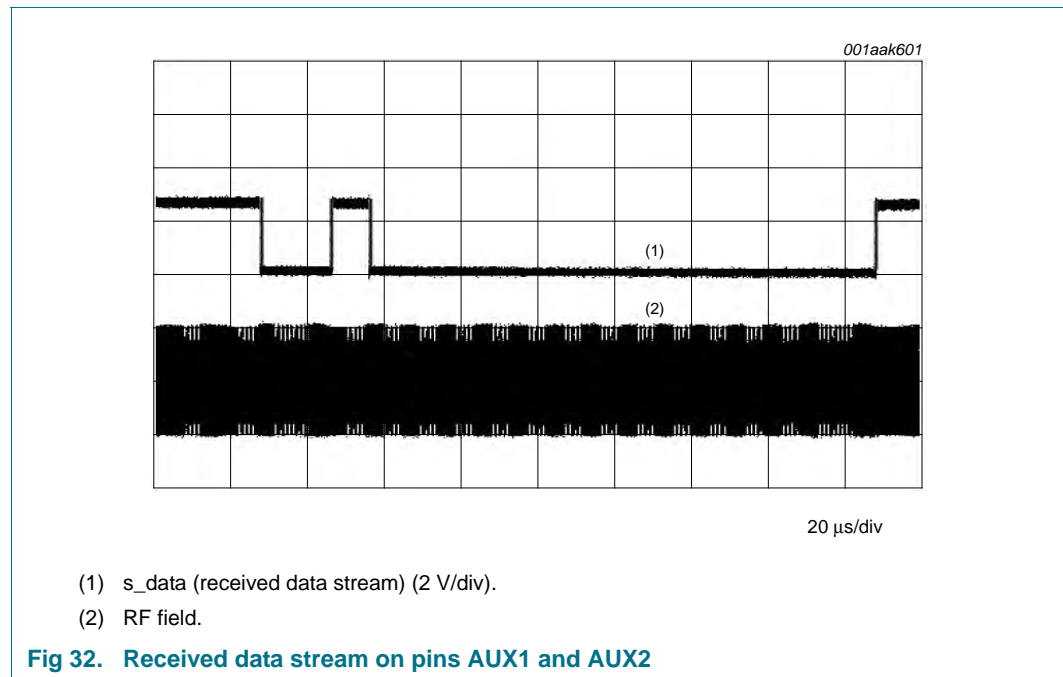
Figure 31 shows the RxActive and TxActive test signals relating to RF communication. The AnalogTestReg register is set to CDh.

- At 106 kBd, RxActive is HIGH during data bits, parity and CRC reception. Start bits are not included
- At 106 kBd, TxActive is HIGH during start bits, data bits, parity and CRC transmission
- At 212 kBd, 424 kBd and 848 kBd, RxActive is HIGH during data bits and CRC reception. Start bits are not included
- At 212 kBd, 424 kBd and 848 kBd, TxActive is HIGH during data bits and CRC transmission



16.1.3.5 Example: Output test signal RX data stream

Figure 32 shows the data stream that is currently being received. The TestSel2Reg register's TestBusSel[4:0] bits are set to 07h to enable test bus signals on pins D1 to D6; see Section 16.1.2 on page 81. The TestSel1Reg register's TstBusBitSel[2:0] bits are set to 06h (pin D6 = s\_data) and AnalogTestReg register is set to FFh (TstBusBit) which outputs the received data stream on pins AUX1 and AUX2.



16.1.3.6 PRBS

The pseudo-random binary sequences PRBS9 and PRBS15 are based on ITU-T0150 and are defined with the TestSel2Reg register. Transmission of either data stream is started by the Transmit command. The preamble/sync byte/start bit/parity bit are automatically generated depending on the mode selected.

**Remark:** All relevant registers for transmitting data must be configured in accordance with ITU-T0150 before selecting PRBS transmission.

17. Package outline

HVQFN32: plastic thermal enhanced very thin quad flat package; no leads; 32 terminals; body 5 x 5 x 0.85 mm

SOT617-1

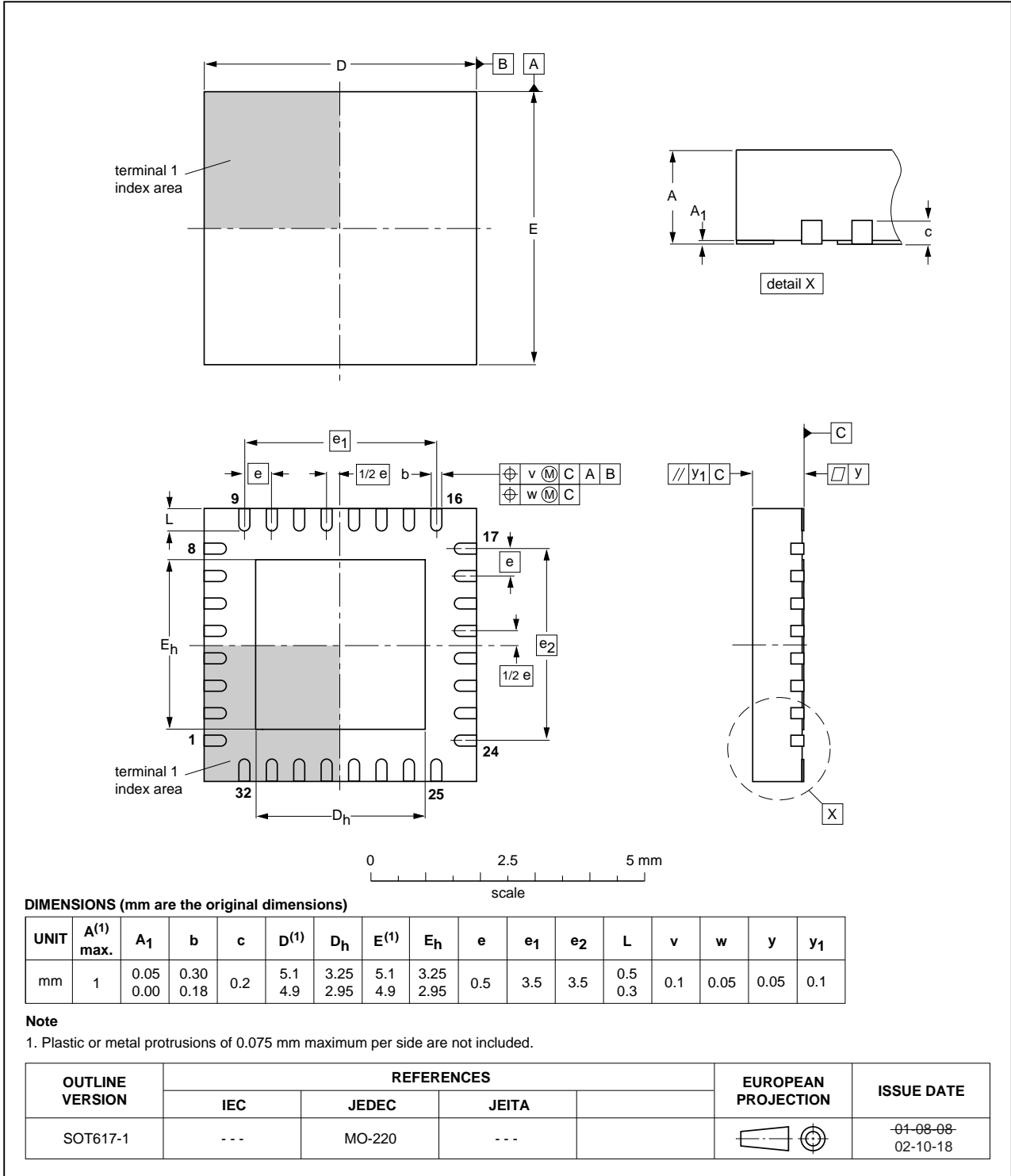


Fig 33. Package outline SOT617-1 (HVQFN32)

Detailed package information can be found at:  
<http://www.nxp.com/package/SOT617-1.html>.

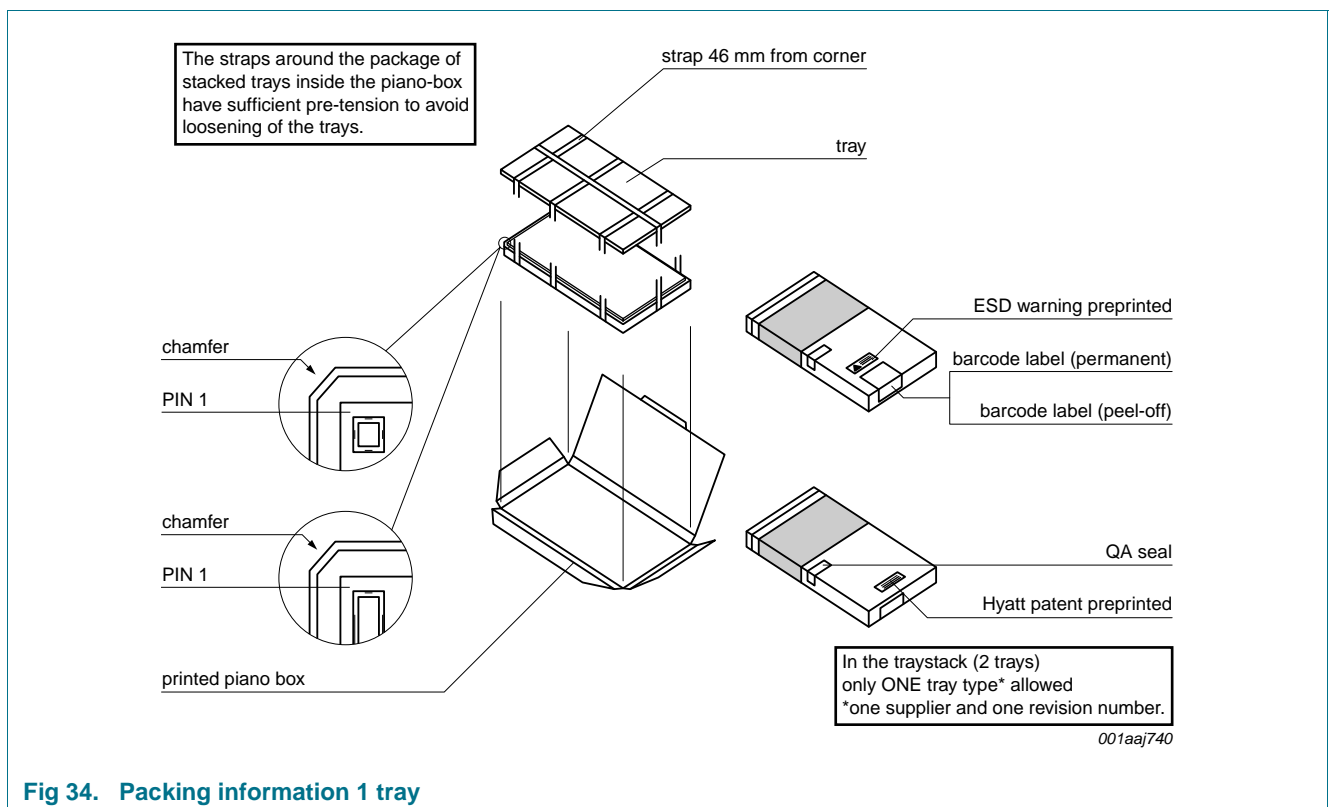
## 18. Handling information

Moisture Sensitivity Level (MSL) evaluation has been performed according to *SNW-FQ-225B rev.04/07/07 (JEDEC J-STD-020C)*. MSL for this package is level 1 which means 260 °C convection reflow temperature.

Dry pack is not required.

Unlimited out-of-pack floor life at maximum ambient 30 °C/85 % RH.

## 19. Packing information



## 20. Abbreviations

**Table 159. Abbreviations**

Acronym	Description
ADC	Analog-to-Digital Converter
BPSK	Binary Phase Shift Keying
CRC	Cyclic Redundancy Check
CW	Continuous Wave
DAC	Digital-to-Analog Converter
HBM	Human Body Model
I <sup>2</sup> C	Inter-integrated Circuit
LSB	Least Significant Bit
MISO	Master In Slave Out
MM	Machine Model
MOSI	Master Out Slave In
MSB	Most Significant Bit
NRZ	Not Return to Zero
NSS	Not Slave Select
PLL	Phase-Locked Loop
PRBS	Pseudo-Random Bit Sequence
RX	Receiver
SOF	Start Of Frame
SPI	Serial Peripheral Interface
TX	Transmitter
UART	Universal Asynchronous Receiver Transmitter

## 21. References

- [1] **Application note** — *MFRC52x Reader IC Family Directly Matched Antenna Design*
- [2] **Application note** — *MIFARE (ISO/IEC 14443 A) 13.56 MHz RFID Proximity Antennas*



## 22. Revision history

Table 160. Revision history

Document ID	Release date	Data sheet status	Change notice	Supersedes
MFRC522 v.3.6	20111214	Product data sheet	-	MFRC522_35
Modifications:				
				<ul style="list-style-type: none"> <li>• <a href="#">Section 2.1 “Differences between version 1.0 and 2.0” on page 1</a>: added</li> <li>• <a href="#">Table 2 “Ordering information” on page 3</a>: updated</li> <li>• <a href="#">Section 9.3.2.10 “DemodReg register” on page 52</a>: register updated and add reference to Timer unit</li> <li>• <a href="#">Section 8.5 “Timer unit” on page 30</a>: Pre Scaler Information for version 2.0 added</li> <li>• <a href="#">Section 9.3.4.8 “VersionReg register” on page 65</a>: version information structured in chip information and version information updated, including version 1.0 and 2.0</li> <li>• <a href="#">Section 16.1 “Test signals” on page 81</a>: selftest result including values for version 1.0 and 2.0</li> </ul>
MFRC522_35	20100621	Product data sheet		MFRC522_34
Modifications:				
				<ul style="list-style-type: none"> <li>• <a href="#">Section 9.3.2.10 “DemodReg register” on page 52</a>: register updated</li> <li>• <a href="#">Section 9.3.3.10 “TModeReg and TPrescalerReg registers” on page 59</a>: register updated</li> <li>• <a href="#">Section 8.5 “Timer unit” on page 30</a>: timer calculation updated</li> <li>• <a href="#">Section 9.3.4.8 “VersionReg register” on page 65</a>: version B2h updated</li> <li>• <a href="#">Section 16.1 “Test signals” on page 81</a>: selftest result updated</li> </ul>
MFRC522_34	20100305	Product data sheet		MFRC522_33
Modifications:				
				<ul style="list-style-type: none"> <li>• <a href="#">Section 8.5 “Timer unit”</a>: information added</li> <li>• <a href="#">Table 106 “TModeReg register bit descriptions”</a>: bit 7 updated</li> <li>• <a href="#">Table 154 “SPI timing characteristics”</a>: row added</li> </ul>
MFRC522_33	20091026	Product data sheet	-	112132

## 23. Legal information

### 23.1 Data sheet status

Document status <sup>[1][2]</sup>	Product status <sup>[3]</sup>	Definition
Objective [short] data sheet	Development	This document contains data from the objective specification for product development.
Preliminary [short] data sheet	Qualification	This document contains data from the preliminary specification.
Product [short] data sheet	Production	This document contains the product specification.

[1] Please consult the most recently issued document before initiating or completing a design.

[2] The term 'short data sheet' is explained in section "Definitions".

[3] The product status of device(s) described in this document may have changed since this document was published and may differ in case of multiple devices. The latest product status information is available on the Internet at URL <http://www.nxp.com>.

### 23.2 Definitions

**Draft** — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

**Short data sheet** — A short data sheet is an extract from a full data sheet with the same product type number(s) and title. A short data sheet is intended for quick reference only and should not be relied upon to contain detailed and full information. For detailed and full information see the relevant full data sheet, which is available on request via the local NXP Semiconductors sales office. In case of any inconsistency or conflict with the short data sheet, the full data sheet shall prevail.

**Product specification** — The information and data provided in a Product data sheet shall define the specification of the product as agreed between NXP Semiconductors and its customer, unless NXP Semiconductors and customer have explicitly agreed otherwise in writing. In no event however, shall an agreement be valid in which the NXP Semiconductors product is deemed to offer functions and qualities beyond those described in the Product data sheet.

### 23.3 Disclaimers

**Limited warranty and liability** — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the *Terms and conditions of commercial sale* of NXP Semiconductors.

**Right to make changes** — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

**Suitability for use** — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or

malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors accepts no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

**Applications** — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

**Limiting values** — Stress above one or more limiting values (as defined in the Absolute Maximum Ratings System of IEC 60134) will cause permanent damage to the device. Limiting values are stress ratings only and (proper) operation of the device at these or any other conditions above those given in the Recommended operating conditions section (if present) or the Characteristics sections of this document is not warranted. Constant or repeated exposure to limiting values will permanently and irreversibly affect the quality and reliability of the device.

**Terms and conditions of commercial sale** — NXP Semiconductors products are sold subject to the general terms and conditions of commercial sale, as published at <http://www.nxp.com/profile/terms>, unless otherwise agreed in a valid written individual agreement. In case an individual agreement is concluded only the terms and conditions of the respective agreement shall apply. NXP Semiconductors hereby expressly objects to applying the customer's general terms and conditions with regard to the purchase of NXP Semiconductors products by customer.

**No offer to sell or license** — Nothing in this document may be interpreted or construed as an offer to sell products that is open for acceptance or the grant, conveyance or implication of any license under any copyrights, patents or other industrial or intellectual property rights.

**Export control** — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

**Quick reference data** — The Quick reference data is an extract of the product data given in the Limiting values and Characteristics sections of this document, and as such is not complete, exhaustive or legally binding.

**Non-automotive qualified products** — Unless this data sheet expressly states that this specific NXP Semiconductors product is automotive qualified, the product is not suitable for automotive use. It is neither qualified nor tested in accordance with automotive testing or application requirements. NXP Semiconductors accepts no liability for inclusion and/or use of non-automotive qualified products in automotive equipment or applications.

In the event that customer uses the product for design-in and use in automotive applications to automotive specifications and standards, customer (a) shall use the product without NXP Semiconductors' warranty of the product for such automotive applications, use and specifications, and (b) whenever customer uses the product for automotive applications beyond NXP Semiconductors' specifications such use shall be solely at customer's own risk, and (c) customer fully indemnifies NXP Semiconductors for any liability, damages or failed product claims resulting from customer design and use of the product for automotive applications beyond NXP Semiconductors' standard warranty and NXP Semiconductors' product specifications.

## 23.4 Licenses

### Purchase of NXP ICs with ISO/IEC 14443 type B functionality



**RATP/Innovatron  
Technology**

This NXP Semiconductors IC is ISO/IEC 14443 Type B software enabled and is licensed under Innovatron's Contactless Card patents license for ISO/IEC 14443 B. The license includes the right to use the IC in systems and/or end-user equipment.

## 23.5 Trademarks

Notice: All referenced brands, product names, service names and trademarks are the property of their respective owners.

**I<sup>2</sup>C-bus** — logo is a trademark of NXP B.V.

**MIFARE** — is a trademark of NXP B.V.

## 24. Contact information

For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: [salesaddresses@nxp.com](mailto:salesaddresses@nxp.com)

**25. Contents**

<b>1</b>	<b>Introduction</b> .....	<b>1</b>	8.6	Power reduction modes .....	32
<b>2</b>	<b>General description</b> .....	<b>1</b>	8.6.1	Hard power-down .....	32
2.1	Differences between version 1.0 and 2.0 .....	1	8.6.2	Soft power-down mode .....	32
<b>3</b>	<b>Features and benefits</b> .....	<b>2</b>	8.6.3	Transmitter power-down mode .....	32
<b>4</b>	<b>Quick reference data</b> .....	<b>2</b>	8.7	Oscillator circuit .....	32
<b>5</b>	<b>Ordering information</b> .....	<b>3</b>	8.8	Reset and oscillator start-up time .....	33
<b>6</b>	<b>Block diagram</b> .....	<b>4</b>	8.8.1	Reset timing requirements .....	33
<b>7</b>	<b>Pinning information</b> .....	<b>6</b>	8.8.2	Oscillator start-up time .....	33
7.1	Pin description .....	6	<b>9</b>	<b>MFRC522 registers</b> .....	<b>34</b>
<b>8</b>	<b>Functional description</b> .....	<b>8</b>	9.1	Register bit behavior .....	34
8.1	Digital interfaces .....	9	9.2	Register overview .....	35
8.1.1	Automatic microcontroller interface detection ..	9	9.3	Register descriptions .....	37
8.1.2	Serial Peripheral Interface .....	10	9.3.1	Page 0: Command and status .....	37
8.1.2.1	SPI read data .....	10	9.3.1.1	Reserved register 00h .....	37
8.1.2.2	SPI write data .....	11	9.3.1.2	CommandReg register .....	37
8.1.2.3	SPI address byte .....	11	9.3.1.3	ComEnReg register .....	37
8.1.3	UART interface .....	11	9.3.1.4	DivlEnReg register .....	38
8.1.3.1	Connection to a host .....	11	9.3.1.5	ComlRqReg register .....	38
8.1.3.2	Selectable UART transfer speeds .....	12	9.3.1.6	DivlRqReg register .....	39
8.1.3.3	UART framing .....	13	9.3.1.7	ErrorReg register .....	40
8.1.4	I <sup>2</sup> C-bus interface .....	16	9.3.1.8	Status1Reg register .....	41
8.1.4.1	Data validity .....	17	9.3.1.9	Status2Reg register .....	42
8.1.4.2	START and STOP conditions .....	17	9.3.1.10	FIFODataReg register .....	43
8.1.4.3	Byte format .....	17	9.3.1.11	FIFOLevelReg register .....	43
8.1.4.4	Acknowledge .....	18	9.3.1.12	WaterLevelReg register .....	43
8.1.4.5	7-Bit addressing .....	19	9.3.1.13	ControlReg register .....	44
8.1.4.6	Register write access .....	19	9.3.1.14	BitFramingReg register .....	45
8.1.4.7	Register read access .....	20	9.3.1.15	CollReg register .....	45
8.1.4.8	High-speed mode .....	21	9.3.1.16	Reserved register 0Fh .....	46
8.1.4.9	High-speed transfer .....	21	9.3.2	Page 1: Communication .....	46
8.1.4.10	Serial data transfer format in HS mode .....	21	9.3.2.1	Reserved register 10h .....	46
8.1.4.11	Switching between F/S mode and HS mode ..	23	9.3.2.2	ModeReg register .....	47
8.1.4.12	MFRC522 at lower speed modes .....	23	9.3.2.3	TxModeReg register .....	47
8.2	Analog interface and contactless UART .....	24	9.3.2.4	RxModeReg register .....	48
8.2.1	General .....	24	9.3.2.5	TxControlReg register .....	49
8.2.2	TX p-driver .....	24	9.3.2.6	TxASKReg register .....	50
8.2.3	Serial data switch .....	26	9.3.2.7	TxSelReg register .....	50
8.2.4	MFIN and MFOUT interface support .....	26	9.3.2.8	RxSelReg register .....	51
8.2.5	CRC coprocessor .....	28	9.3.2.9	RxThresholdReg register .....	52
8.3	FIFO buffer .....	28	9.3.2.10	DemodReg register .....	52
8.3.1	Accessing the FIFO buffer .....	28	9.3.2.11	Reserved register 1Ah .....	53
8.3.2	Controlling the FIFO buffer .....	28	9.3.2.12	Reserved register 1Bh .....	53
8.3.3	FIFO buffer status information .....	28	9.3.2.13	MfTxReg register .....	53
8.4	Interrupt request system .....	29	9.3.2.14	MfRxReg register .....	54
8.4.1	Interrupt sources overview .....	29	9.3.2.15	Reserved register 1Eh .....	54
8.5	Timer unit .....	30	9.3.2.16	SerialSpeedReg register .....	54
			9.3.3	Page 2: Configuration .....	56
			9.3.3.1	Reserved register 20h .....	56

continued >>

9.3.3.2	CRCResultReg registers	56	16.1.2	Test bus	81
9.3.3.3	Reserved register 23h	57	16.1.3	Test signals on pins AUX1 or AUX2	82
9.3.3.4	ModWidthReg register	57	16.1.3.1	Example: Output test signals TestDAC1 and TestDAC2	83
9.3.3.5	Reserved register 25h	57	16.1.3.2	Example: Output test signals Corr1 and MinLevel	83
9.3.3.6	RFCfgReg register	58	16.1.3.3	Example: Output test signals ADC channel I and ADC channel Q	84
9.3.3.7	GsNReg register	58	16.1.3.4	Example: Output test signals RxActive and TxActive	85
9.3.3.8	CWGsPReg register	59	16.1.3.5	Example: Output test signal RX data stream	86
9.3.3.9	ModGsPReg register	59	16.1.3.6	PRBS	86
9.3.3.10	TModeReg and TPrescalerReg registers	59	<b>17</b>	<b>Package outline</b>	<b>87</b>
9.3.3.11	TReloadReg register	61	<b>18</b>	<b>Handling information</b>	<b>88</b>
9.3.3.12	TCounterValReg register	61	<b>19</b>	<b>Packing information</b>	<b>88</b>
9.3.4	Page 3: Test	62	<b>20</b>	<b>Abbreviations</b>	<b>89</b>
9.3.4.1	Reserved register 30h	62	<b>21</b>	<b>References</b>	<b>89</b>
9.3.4.2	TestSel1Reg register	62	<b>22</b>	<b>Revision history</b>	<b>90</b>
9.3.4.3	TestSel2Reg register	63	<b>23</b>	<b>Legal information</b>	<b>91</b>
9.3.4.4	TestPinEnReg register	63	23.1	Data sheet status	91
9.3.4.5	TestPinValueReg register	64	23.2	Definitions	91
9.3.4.6	TestBusReg register	64	23.3	Disclaimers	91
9.3.4.7	AutoTestReg register	65	23.4	Licenses	92
9.3.4.8	VersionReg register	65	23.5	Trademarks	92
9.3.4.9	AnalogTestReg register	66	<b>24</b>	<b>Contact information</b>	<b>92</b>
9.3.4.10	TestDAC1Reg register	67	<b>25</b>	<b>Contents</b>	<b>93</b>
9.3.4.11	TestDAC2Reg register	67			
9.3.4.12	TestADCReg register	67			
9.3.4.13	Reserved register 3Ch	67			
<b>10</b>	<b>MFRC522 command set</b>	<b>69</b>			
10.1	General description	69			
10.2	General behavior	69			
10.3	MFRC522 command overview	69			
10.3.1	MFRC522 command descriptions	70			
10.3.1.1	Idle	70			
10.3.1.2	Mem	70			
10.3.1.3	Generate RandomID	70			
10.3.1.4	CalcCRC	70			
10.3.1.5	Transmit	70			
10.3.1.6	NoCmdChange	70			
10.3.1.7	Receive	71			
10.3.1.8	Transceive	71			
10.3.1.9	MFAuthent	71			
10.3.1.10	SoftReset	72			
<b>11</b>	<b>Limiting values</b>	<b>73</b>			
<b>12</b>	<b>Recommended operating conditions</b>	<b>73</b>			
<b>13</b>	<b>Thermal characteristics</b>	<b>73</b>			
<b>14</b>	<b>Characteristics</b>	<b>74</b>			
14.1	Timing characteristics	77			
<b>15</b>	<b>Application information</b>	<b>80</b>			
<b>16</b>	<b>Test information</b>	<b>81</b>			
16.1	Test signals	81			
16.1.1	Self test	81			

Please be aware that important notices concerning this document and the product(s) described herein, have been included in section 'Legal information'.

© NXP B.V. 2011.

All rights reserved.

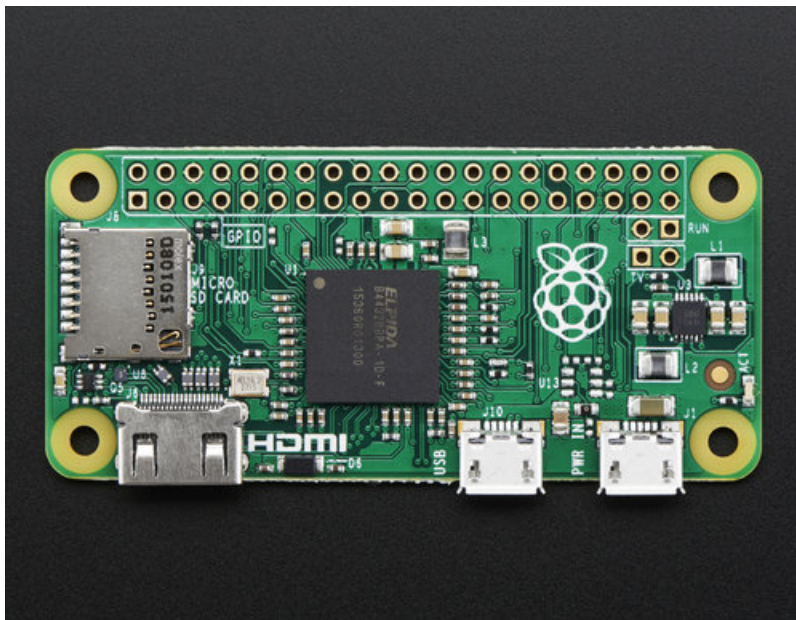
For more information, please visit: <http://www.nxp.com>

For sales office addresses, please send an email to: [salesaddresses@nxp.com](mailto:salesaddresses@nxp.com)

Date of release: 14 December 2011  
112136

## Introducing the Raspberry Pi Zero

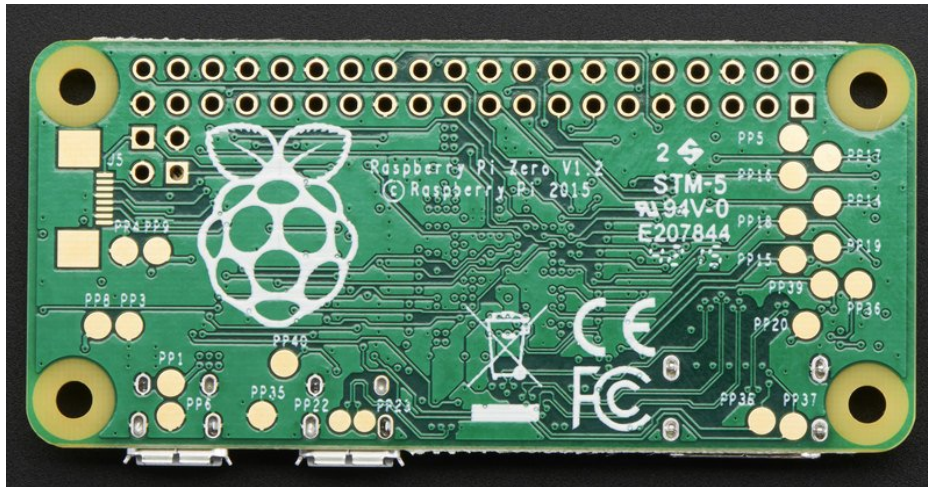
Created by lady ada



Last updated on 2019-10-31 09:34:18 PM UTC





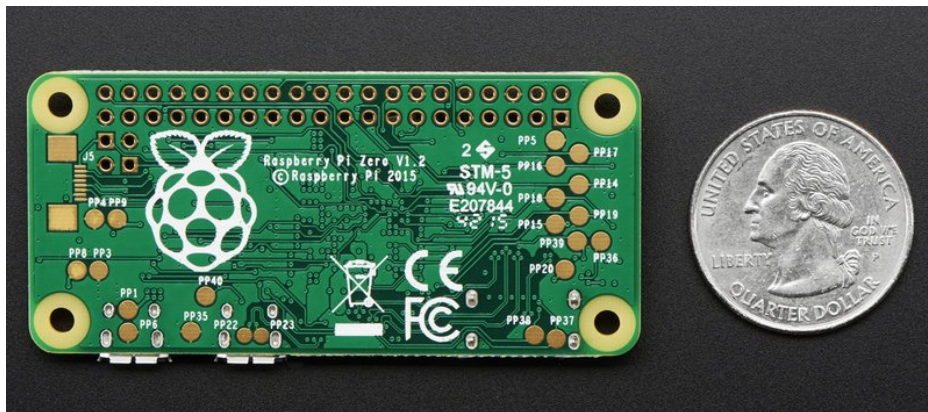


## Size

First up, the Pi Zero is **small and thin**

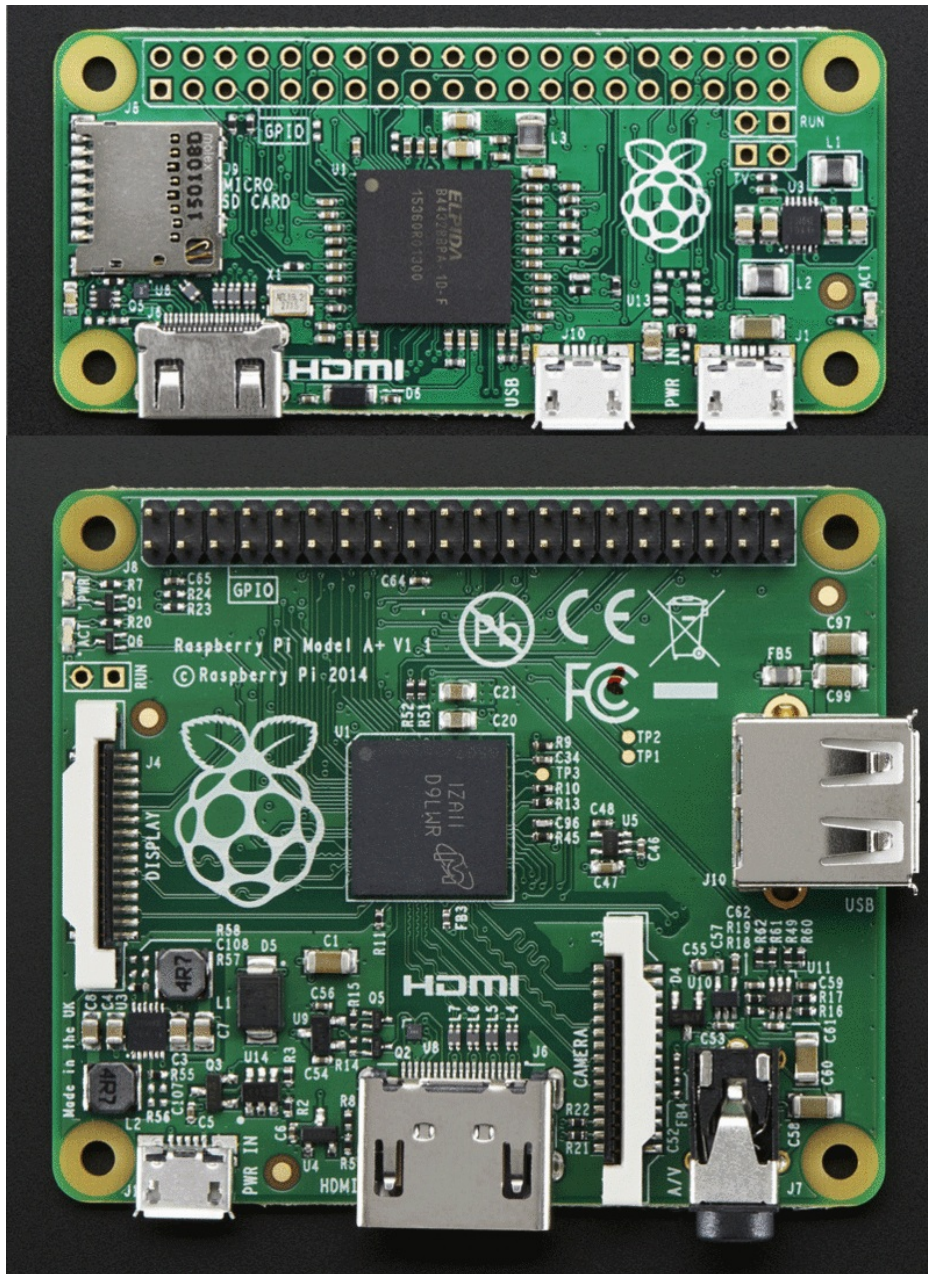
**65mm long x 30mm wide x 5mm thick**

(31mm if you include the little sticky-out bits of the microUSB jacks)



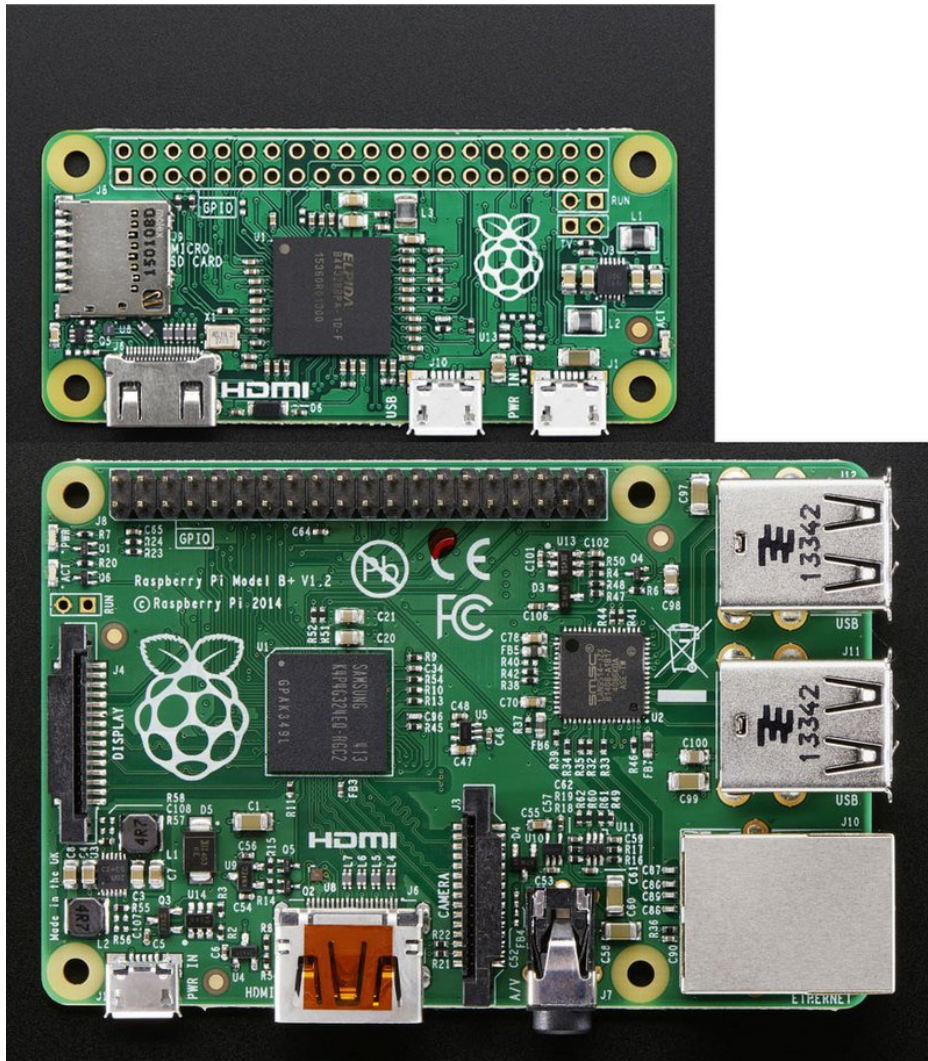
Way smaller than the Pi 2 or B+ and even smaller than the A+, its 60% the size of the A+: same length, and about half the width:





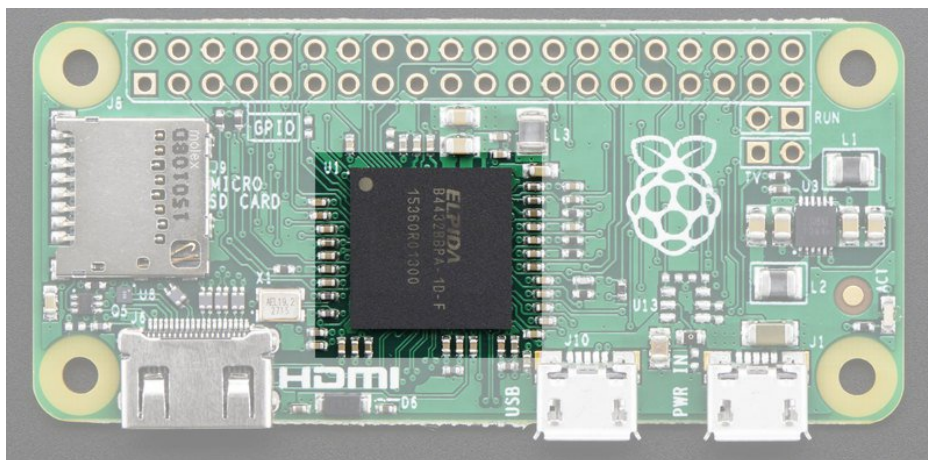
And about 40% the size of the Pi 2 or B+





## Processor and Speed

To keep the Pi Zero low cost, the processor and RAM are kept pretty basic. Instead of the Pi 2's zippy quad core ARM v7, we're back to a single-core 1GHz ARM (same processor in the Pi Model B+ and A+). We also have 512 MB of RAM with a 'package-on-package' setup. The chip shown here:



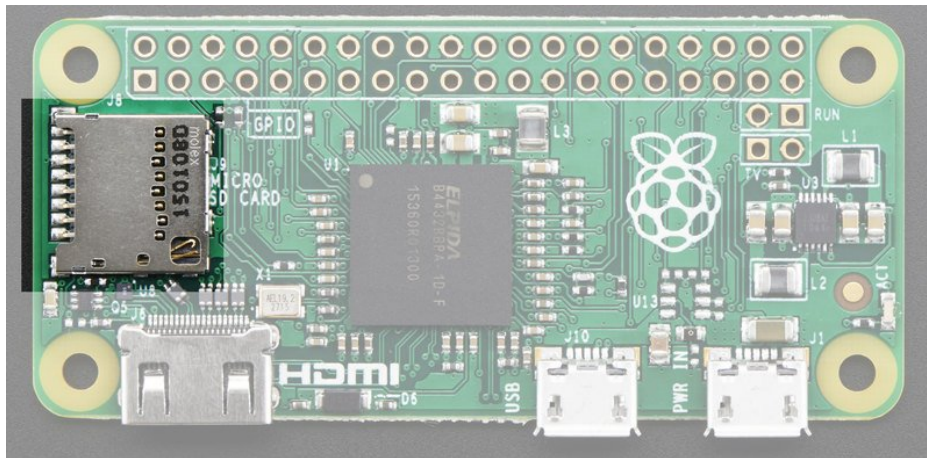
Is the RAM that is sitting *on top* of the main processor.

For maker and hacker projects, this isn't a big deal. You're essentially going to get the same performance as the Pi A+ or B+. If you're looking for something that can do some more serious processing, [check out the Pi 2](http://adafru.it/2358) (<http://adafru.it/2358>)

## Micro SD Card Holder

---

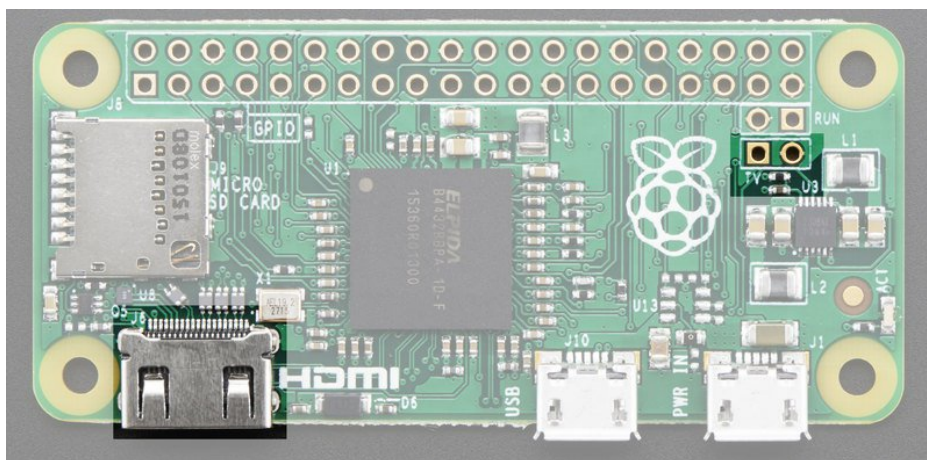
Not much has changed here, we're still going with MicroSD for size and ease of use (they're the most common card size these days!) This time the card holder is up top and is **push-pull** style not **push-push**. Honestly, I prefer it this way since you won't accidentally 'push-pop' the card out



## Video Out

---

HDMI Video is still available, you'll want to use a [Mini to Standard HDMI adapter](http://adafru.it/2819) (<http://adafru.it/2819>) to connect an HDMI cable. There's no 3.5mm jack with composite out, however you can get PAL or NTSC out via two 0.1" pads. [We've got a bigger write-up here about Pi Zero video outputs.](https://adafru.it/jEf) (<https://adafru.it/jEf>)



## Audio out

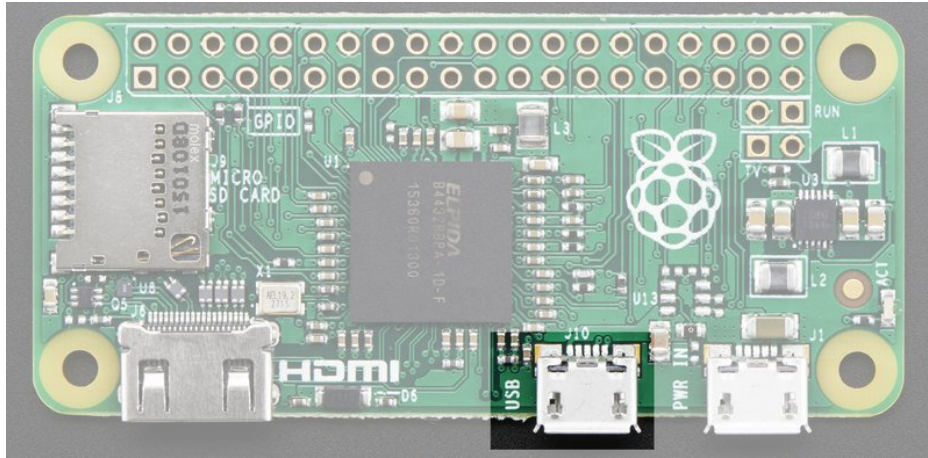
---

No analog audio out, but if you connect HDMI to a monitor with speakers you will get HDMI digital audio. It's also possible to hack analog audio out with a few passive components, [see our more detailed look at Pi Zero audio output options.](https://adafru.it/jEh) (<https://adafru.it/jEh>)



## USB Port

Like the Pi Model A+, the Pi Zero **does not have a USB Hub built in** which means you get **one USB port!** Moreover that USB port is not a standard type A port, instead it is a 'USB On-The-Go' port



In order to connect a USB device (mouse, keyboard, WiFi) etc you'll need a **USB OTG micro B to A cable** (<http://adafru.it/1099>):



If you need to connect multiple USB devices, a simple USB hub will do what you need. **A powered hub is even better** (<http://adafru.it/961>), and will let you power high-current USB devices like WiFi adapters and even external USB hard-drives.

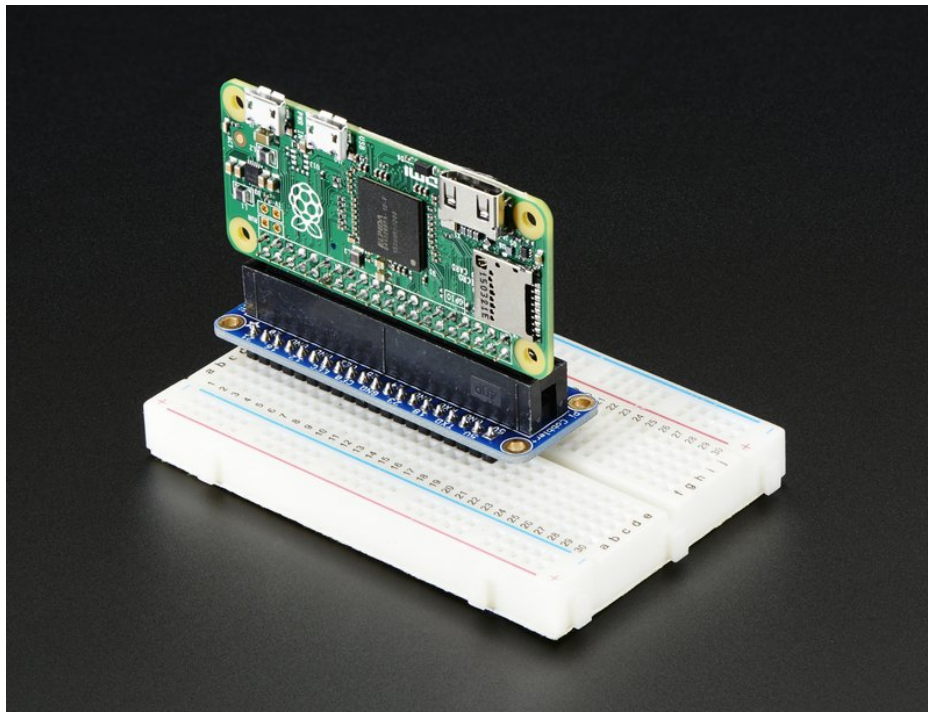


As a bonus you can power the Pi Zero *from* the hub (the power cable does not pass any data) - just plug the power micro USB cable into one of the ports.

## GPIO Header

---

To keep the Zero as simple and small as possible, the 'normal' GPIO header spot has been left blank! Normally, [a 2x20 male header is soldered in there](http://adafru.it/2822) (<http://adafru.it/2822>). While you could grab one of those and solder them in, the empty spot has a lot of potential. For example, you can solder in right-angle socket header, and turn the Pi Zero it a sort of 'daughter card'



We've got more ideas and suggestions on our GPIO header detail page (<https://adafru.it/jEi>)

## Setting up your SD card



Before you can power up your Pi Zero, you will need to program in the SD card with an **Operating System**

Much like your computer has Windows, Mac OS X or Linux on it to make it run, the Raspberry Pi needs something to help it boot and run software. That software is **Raspbian Linux** (a *flavor* of Debian Linux). [You can check out our tutorial on What Is Linux if you're curious to learn more \(https://adafru.it/jDZ\)](https://adafru.it/jDZ)

If you just want to get rockin, grab the [latest \(https://adafru.it/fQi\)](https://adafru.it/fQi) **Raspbian Jessie** operating system [from the Raspberry Pi downloads page \(https://adafru.it/fi7\)](https://adafru.it/fi7)

Just click the button below!

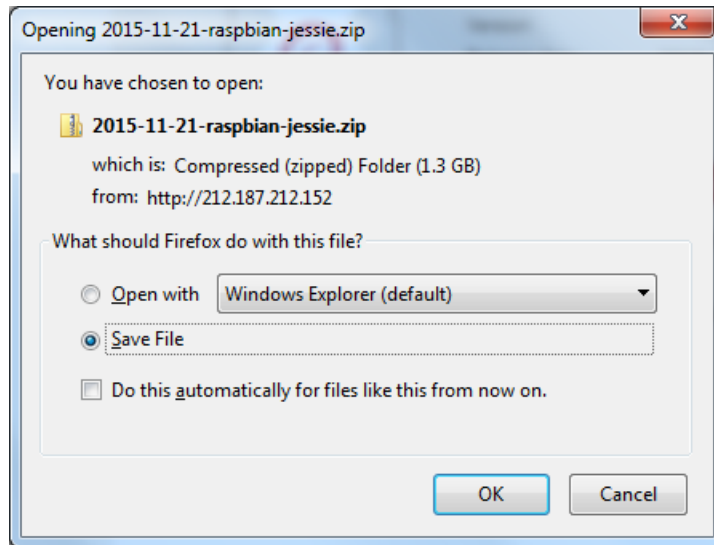
<https://adafru.it/jE0>

<https://adafru.it/jE0>



Raspbian Wheezy 5-15 or earlier do not support the Zero! Try Jessie instead





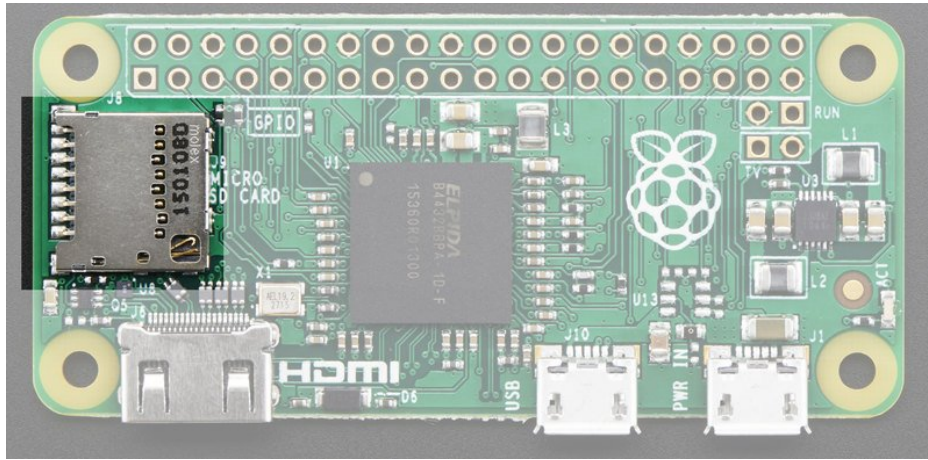
Once downloaded, unzip the zip file, the full image is about 4.5 Gigabytes.

Next up grab your SD or micro SD card reader and plug it into your computer



Now follow our guide for [Windows](https://adafru.it/jE4) (<https://adafru.it/jE4>) or [Mac OS X](https://adafru.it/jE5) (<https://adafru.it/jE5>) to burn the image





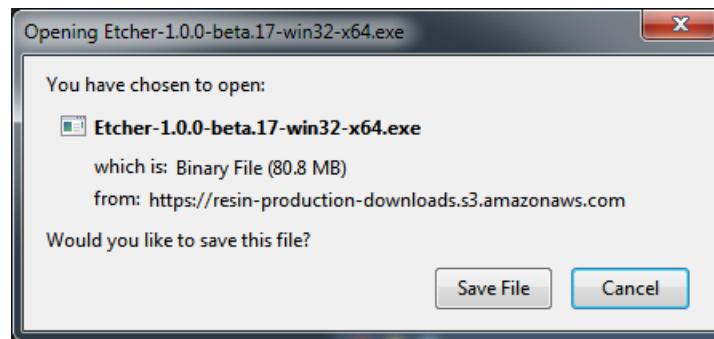
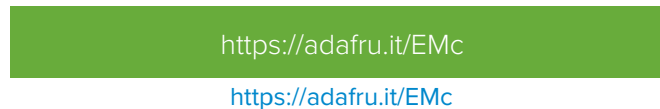
Once you're done, plug the micro SD card into the slot indicated. It will fit snugly in place but you won't hear or feel a 'click'

## Making an SD Card – Using Windows

We really like using balenaEtcher for burning SD cards. Works great on any version of Windows, macOS and Linux. It will not over-write your backup disk drive, and can handle compressed images so you do not need to unzip them!

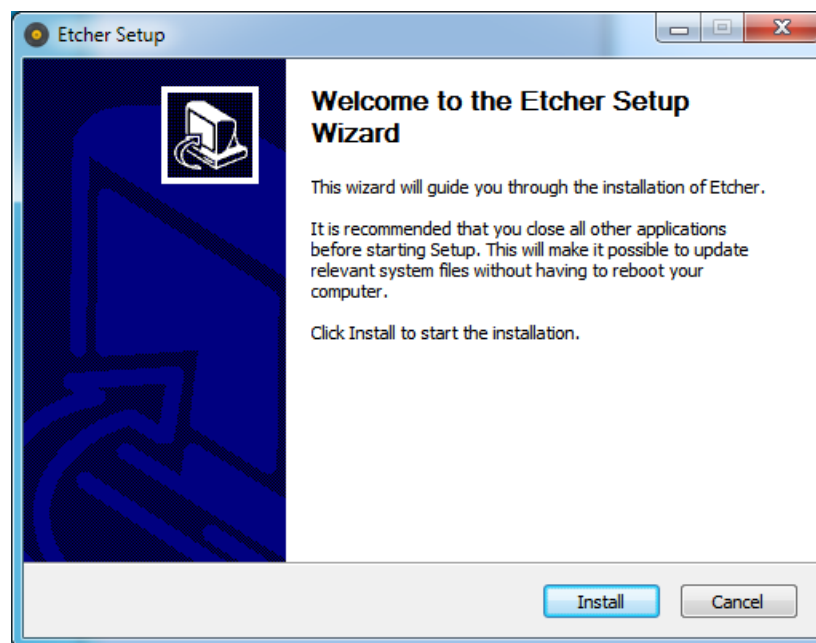
### Step 1.

Download Etcher from: <https://www.balena.io/etcher/> (<https://adafru.it/EMc>)



### Step 2.

Run the downloaded app to install!



You can start immediately, doubleclick the Etcher desktop icon, or select it from the Start menu

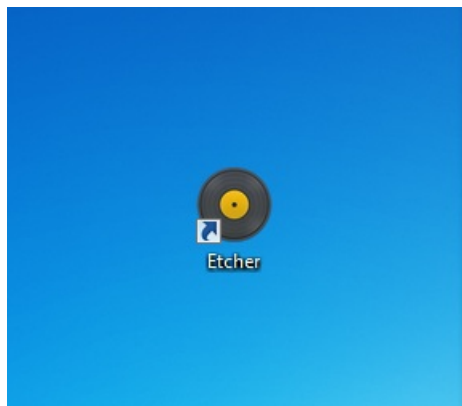
### Step 3.

Eject any external storage devices such as USB flash drives and backup hard disks. This makes it easier to identify the SD card. Then insert the SD card into the slot on your computer or into the reader.

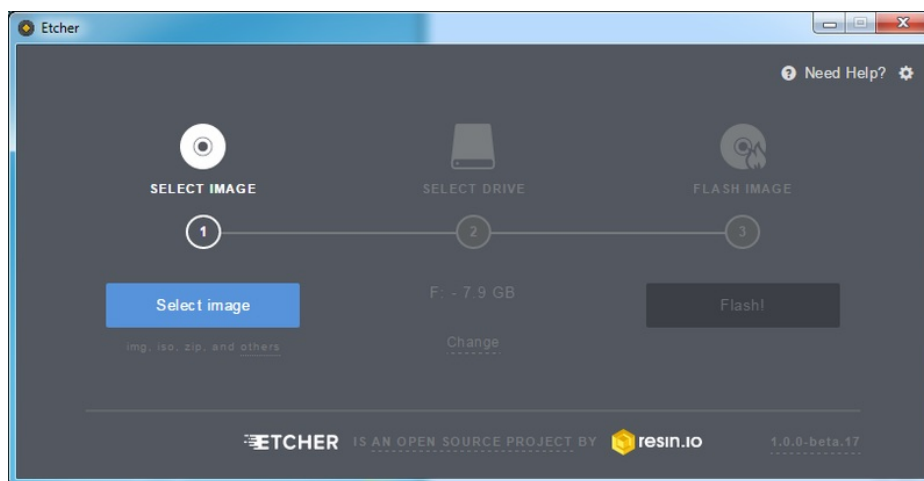
## Step 4.

---

Run the Etcher program



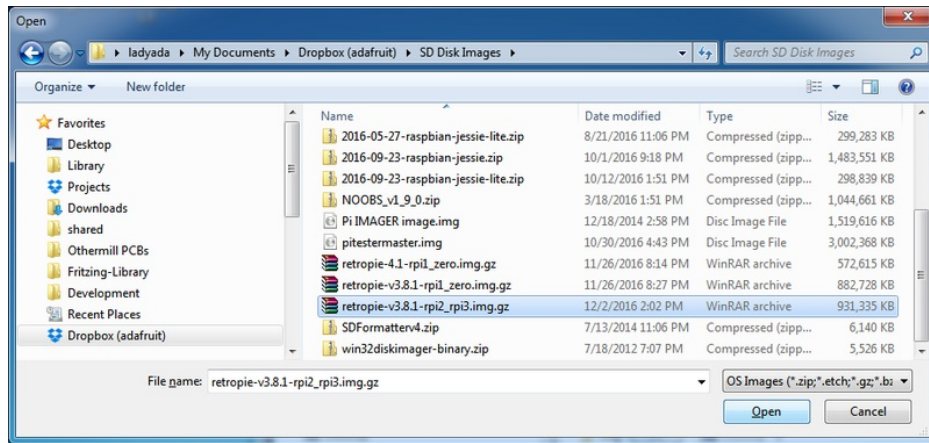
This will launch the following application.



## Step 5.

---

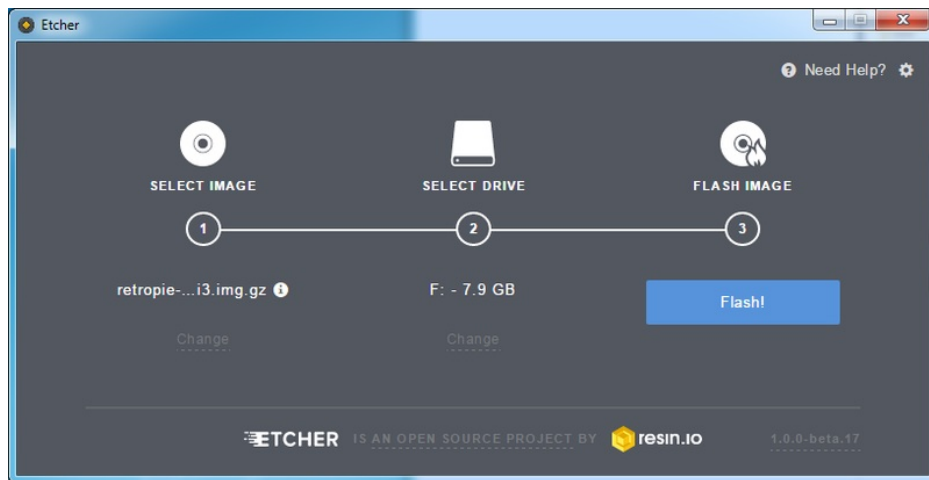
Select the image file by clicking **Select Image** you can select a compressed file such as a **.zip** or **.gz**



## Step 6.

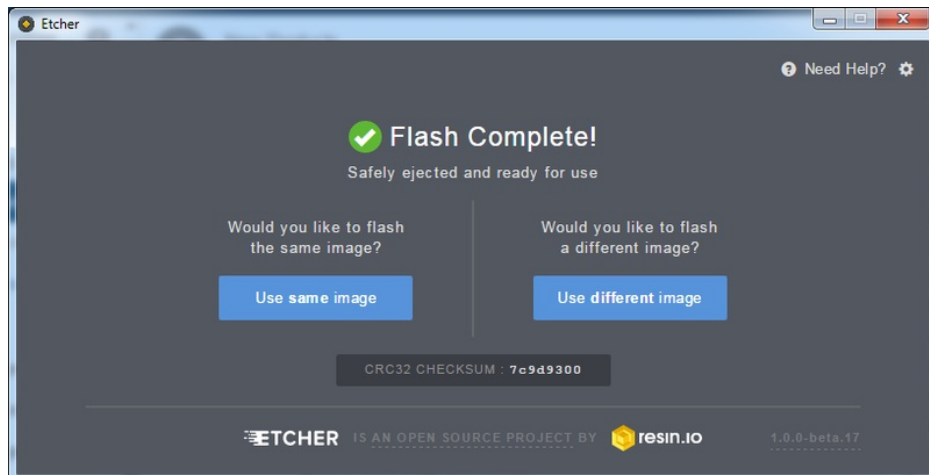
Etcher will automatically try to detect the SD drive, check the size to make sure its the right one

Then click **Flash!**



Check that you have the right device, as it will be reformatted, and then click Install.

It will take a few minutes to install, but once the SD card is ready, you will see the following.

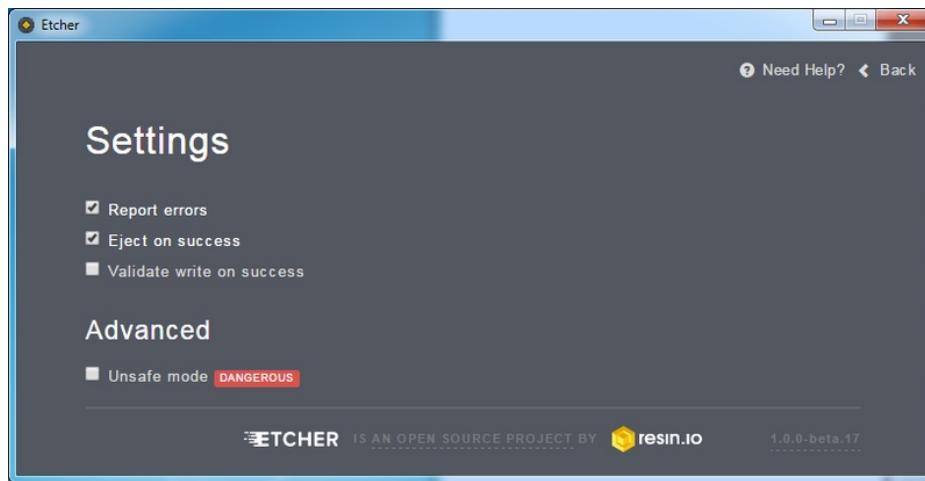


That's all there is to it. Your SD card is ready for use in your Raspberry Pi.

## Faster writes

---

If you burn a lot of cards, speed it up by turning off **Validate write on success**



## Making an SD Card – Using a Mac

We really like using balena**Etcher** for burning SD cards. Works great on Mac OS X 10.9 or later, won't over-write your backup disk drive, and can handle compressed images so you do not need to unzip them!

### Mac OS Catalina Issues

If you are having issues running Etcher on the Catalina release of Mac OS, see the links below for more information and some suggested workarounds.

- [Issue 2833 \(https://adafru.it/GB4\)](https://adafru.it/GB4)
- [Issue 2911 \(https://adafru.it/GB5\)](https://adafru.it/GB5)
- [Balena forum post \(https://adafru.it/GB7\)](https://adafru.it/GB7)

Most success has been reported by simply running Etcher from the command line using sudo:

```
sudo /Applications/balenaEtcher.app/Contents/MacOS/balenaEtcher
```

### Step 1.

---

Download Etcher from <https://www.balena.io/etcher/> (<https://adafru.it/EMc>)

<https://adafru.it/EMc>

<https://adafru.it/EMc>

### Step 2.

---

Open the downloaded disk image and drag the balenaEtcher application to the Applications folder. You can then eject the disk image.

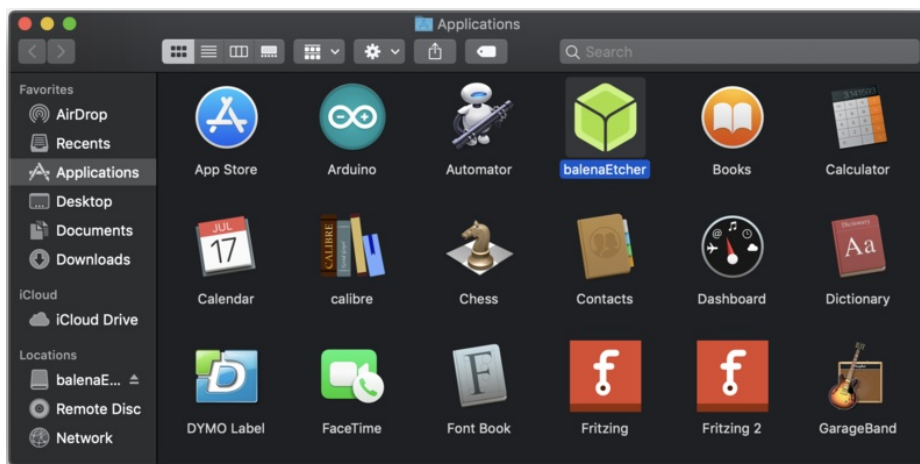


### Step 3.

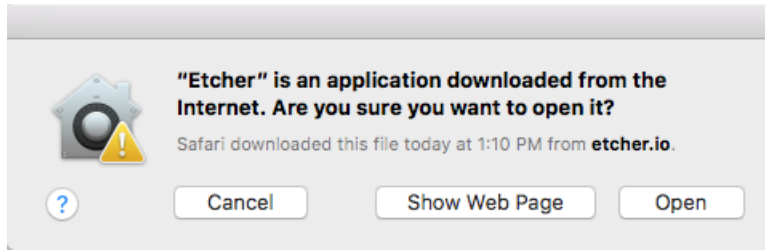
Eject any external storage devices such as USB flash drives and backup hard disks. This makes it easier to identify the SD card. Then insert the SD card into the slot on your computer or into the reader.

### Step 4.

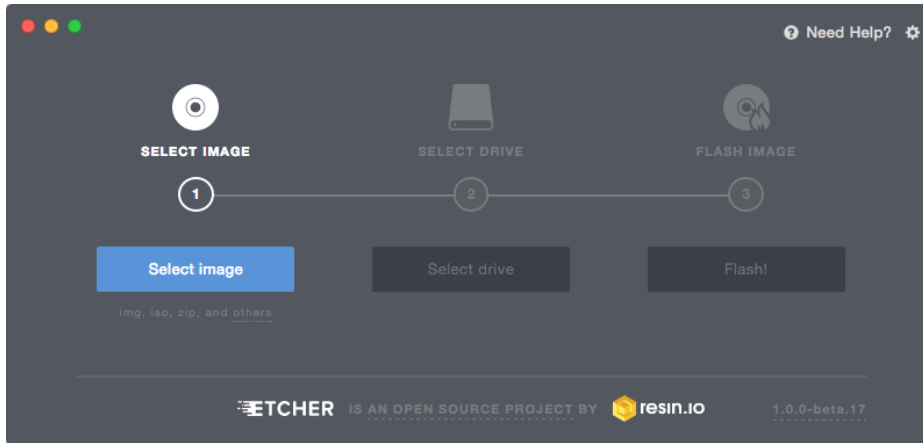
Run the Etcher application.



The first time you run Etcher you'll be asked to confirm the download. Click "Open" to continue.

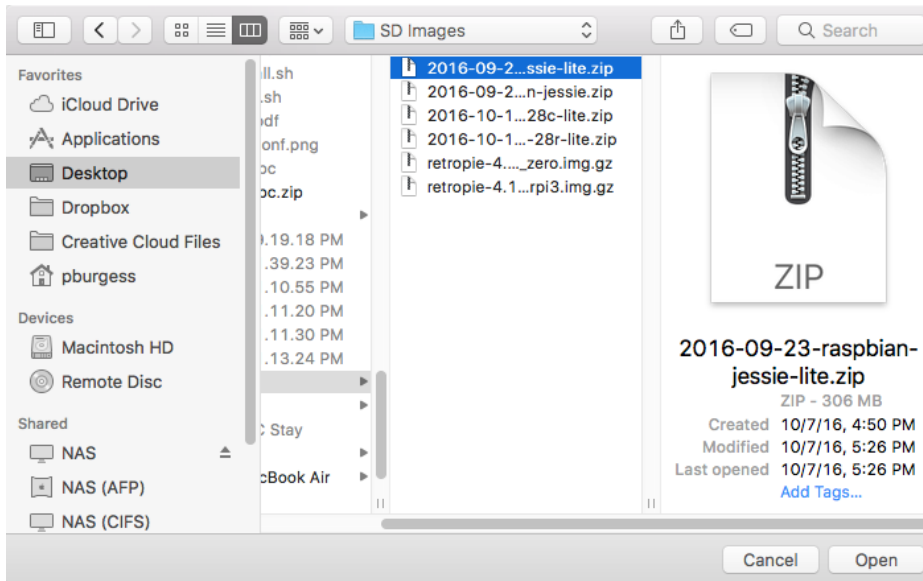


This will launch the Etcher application...



## Step 5.

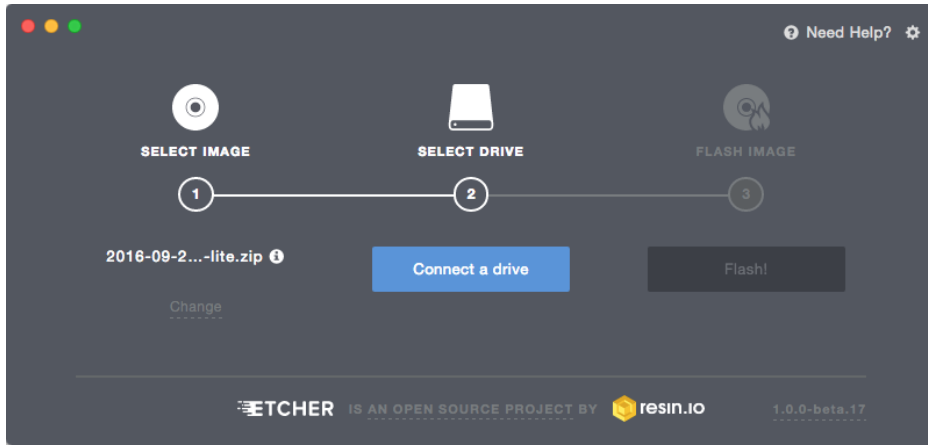
Select the SD card image file by clicking **Select Image**. You can choose a compressed SD image file such as a **.zip** or **.gz** or an uncompressed **.img**, it's all good!



## Step 6.

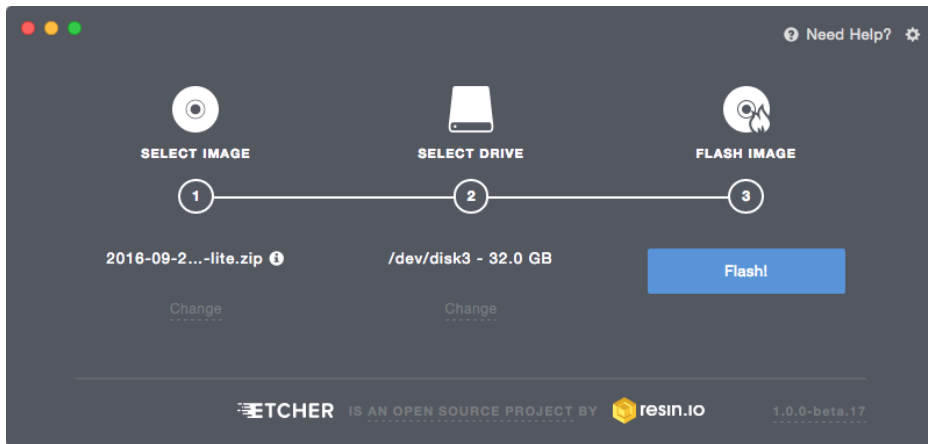
Etcher will automatically try to detect the SD drive. If you don't have an SD card currently inserted, you'll be prompted to connect one.





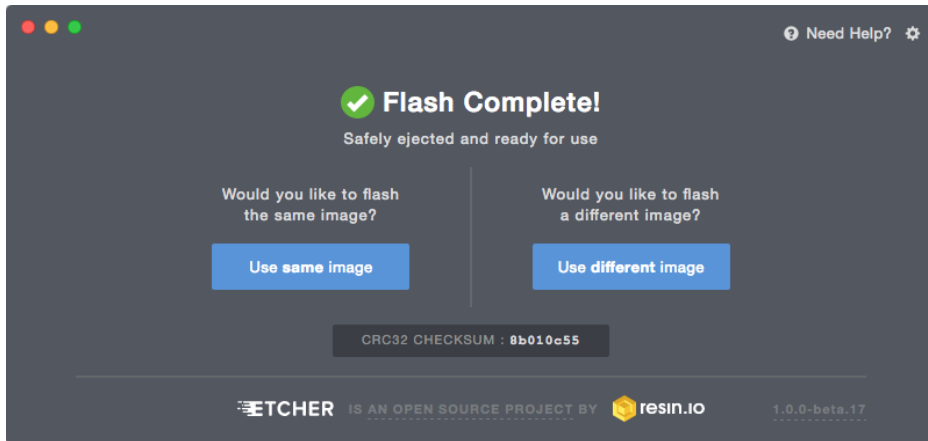
Check the disk size to make sure its the right one, that it's not overwriting your main drive or anything nasty.

Then click **Flash!** *A-ah!*



Etcher will work for a few minutes to “burn” the SD image to the card. You’ll see a progress bar as it works. This is about the time you’ll wish you’d splurged on a high-speed card.

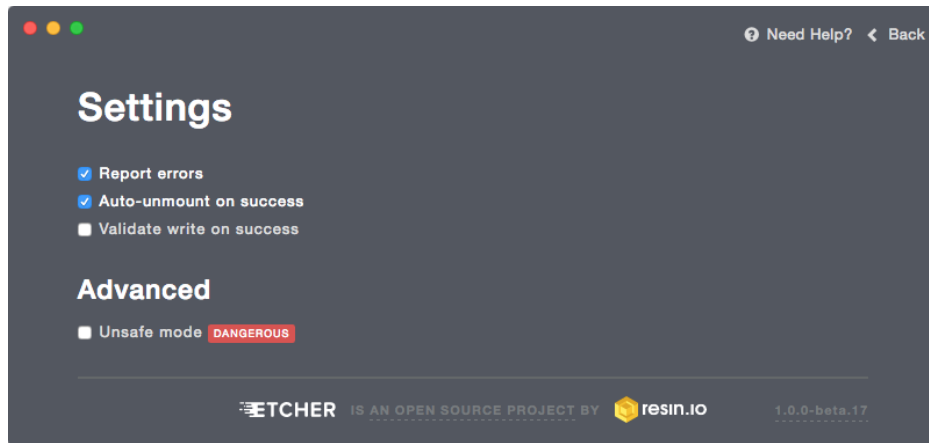
Once the SD card is ready, you will see the following:



The card will be unmounted automatically, so you can pull it out now and use it in your Raspberry Pi.

## Faster Writes

If you find yourself burning a lot of SD cards, you can speed things up by clicking the gear icon at the top-right, then turn off the “Validate write” option. I’ve written *hundreds* of cards and only had *one* fail validation.

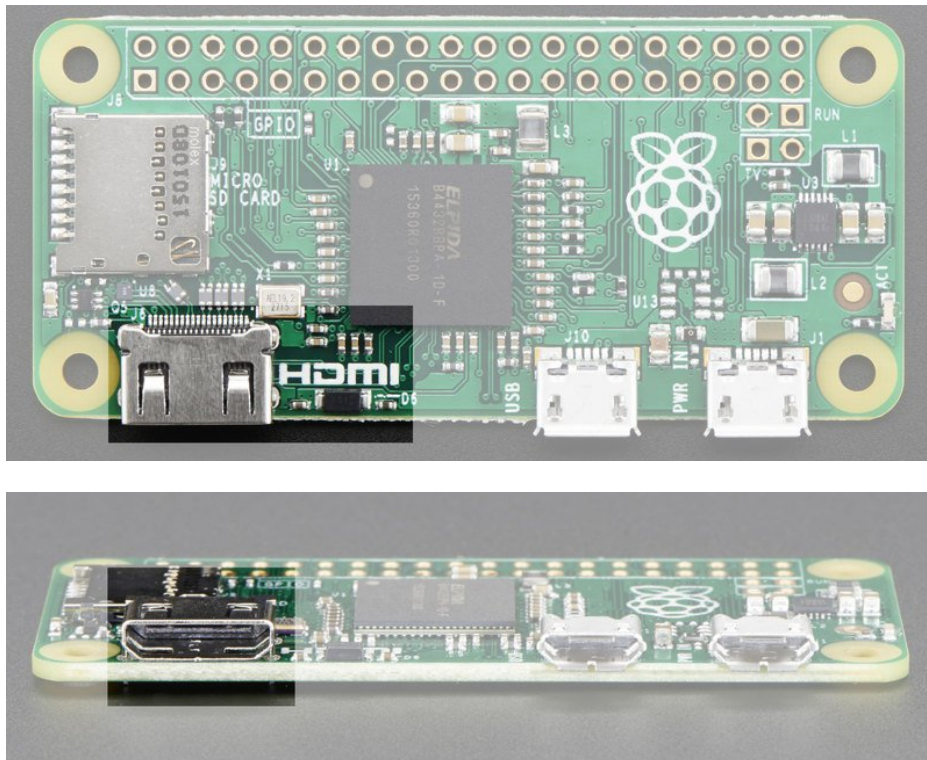


## Video Outputs

The Raspberry Pi chipset was originally designed to be a HDMI/graphics co-processor for mobile devices. For that reason, it has quite a bit of 'HDMI horsepower' and can, despite it's small size, play 1080p video at full screen.

### HDMI Video Out

The easiest & fastest way to get video going is to connect up an HDMI display (<https://adafru.it/jsb>). We have a ton of options, and any HDMI display size from 640x480 up to 1920x1080 will work. The Mini HDMI port is conveniently labeled and shown below:



For example, our 5" HDMI touch backpack which is the smallest all-in-one display we carry can be powered from the Pi Zero's USB port and provide a touchscreen at the same time (<http://adafru.it/2260>)



*(Shown here with a Pi 2 because, well, the Pi Zero wasn't out at the time)*

To connect an HDMI device, you'll need 2 things, a [Mini HDMI to HDMI Adapter](http://adafru.it/2819) (<http://adafru.it/2819>) and an [HDMI Cable](http://adafru.it/608) (<http://adafru.it/608>)

The HDMI cable is pretty straight-forward to understand, and you can get one anywhere. The HDMI adapter is required because the Pi Zero does not have a standard size HDMI port, instead the port is slimmer and smaller to keep the Zero petite. The adapter is pretty straight forward to use - plug it into the Pi Zero and the port is now large enough for any standard HDMI cable



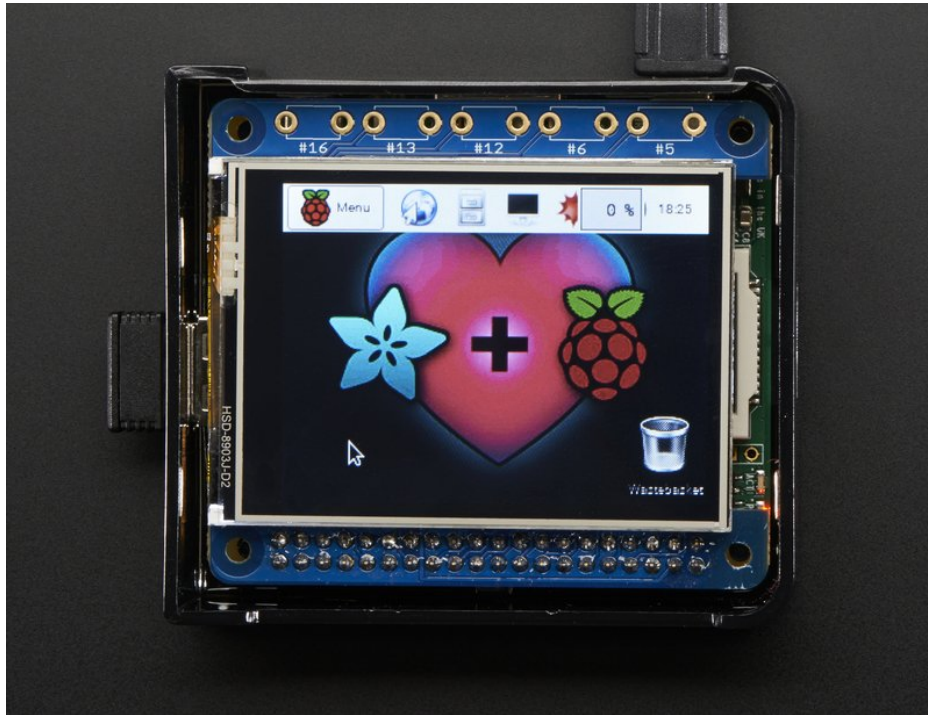
## PiTFT Video

---

Even though it is 'half size' of the A+, [you can still use any of our PiTFT's on the Pi Zero](https://adafru.it/jE7) (https://adafru.it/jE7) You can use any size from our 2.2" 320x240 PiTFT HAT, up to our 3.5" Touchscreen 480x320.

Before you can plug in a HAT or PiTFT [you'll need to solder in the 2x20 male header](http://adafru.it/2822) (http://adafru.it/2822)

Then follow the tutorial for the PiTFT of your choice! Be sure to pick the Jessie install image



## VGA Video Out

---

This one is pretty easy, just use the HDMI adapter above, and an [HDMI to VGA adapter](http://adafru.it/1151) (this also has the benefit of giving you an audio output) (http://adafru.it/1151)

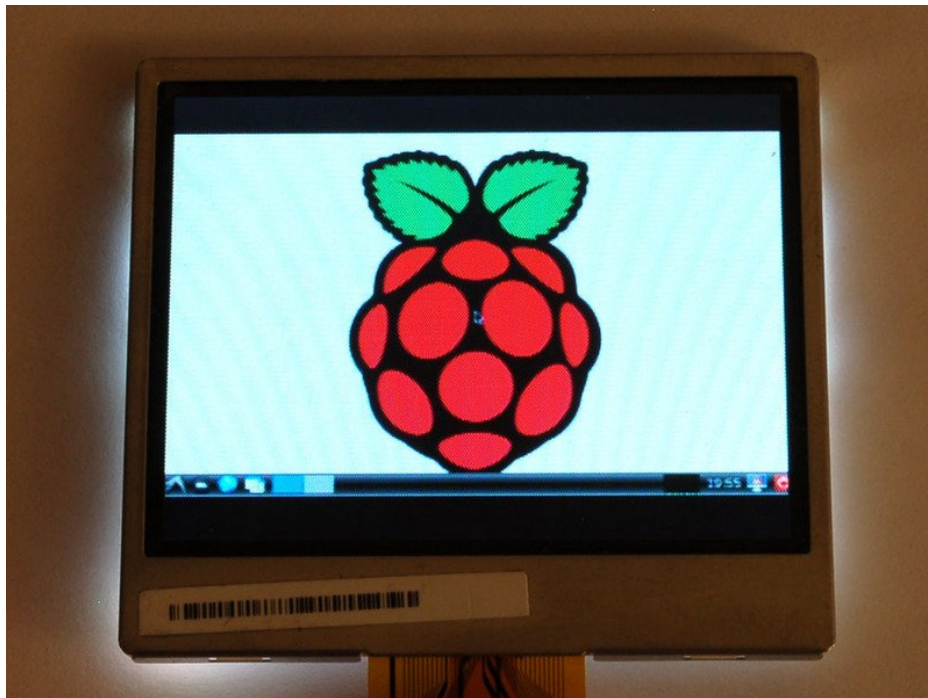




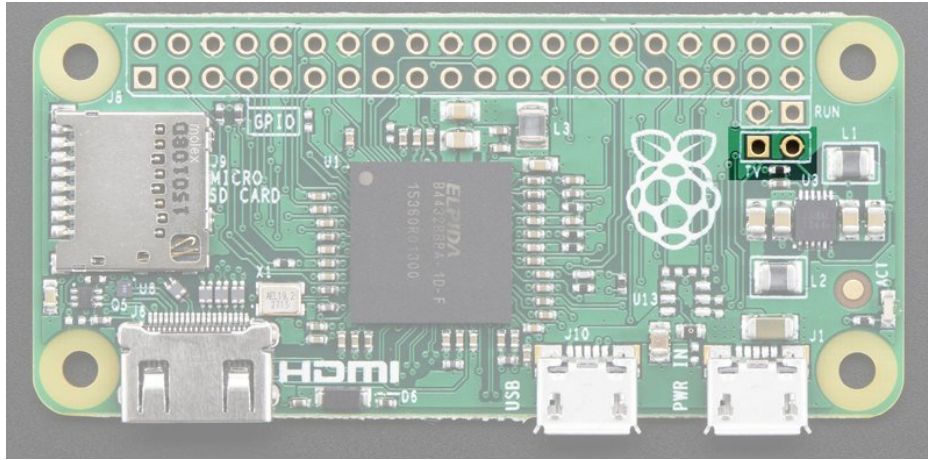
## NTSC/PAL Composite Video

---

OK so you want TV video? Maybe for one of our [very tiny composite video screens \(https://adafru.it/jE8\)](https://adafru.it/jE8)?



Well, the quality is not going to be nearly as nice as with VGA or HDMI but you can do it. Find the two pads marked **TV** on the 'Zero



The hole on the left, nearest to the **TV** text, is the signal (+) line, the pin to the right of it is the ground (-) line. Solder two wires to these pads and connect them to an [RCA Jack](http://adafru.it/2792) (<http://adafru.it/2792>) like this one



Make sure to not have HDMI plugged in, it should auto-switch to TV out. If you have somehow set your Pi for HDMI out only, plug your HDMI screen back in, or use a console cable to connect and log into the Pi. Then run `sudo raspi-config` at a command line to set video output to composite! You'll also want to tweak your Pi to use composite in the nicest resolution possible (<https://adafru.it/diN>)

# Audio Outputs

Uh, well, there aren't any! That's right, to keep the Pi Zero small and low cost, the headphone audio filter isn't included. You can still get digital audio out via HDMI so if you plug it your Pi into a monitor with speakers, that will work fine.

Well, ok that's not the whole truth

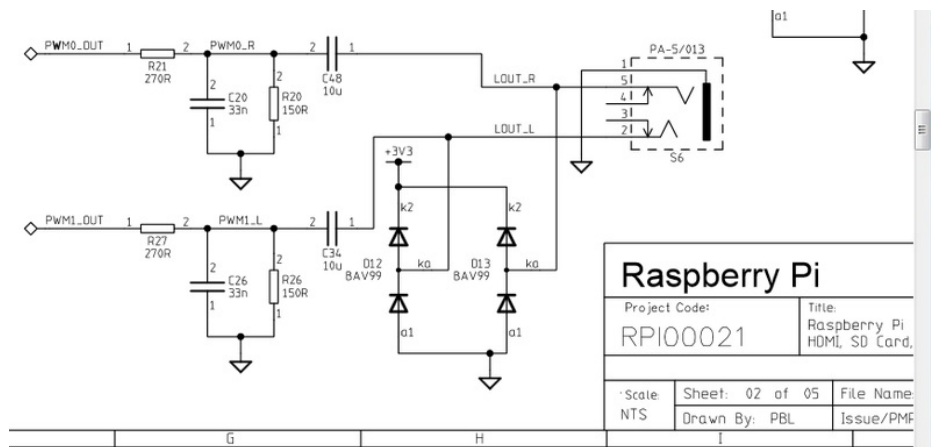
## How to Add Audio Outputs to your Pi Zero

Hey, wanna do the below but with step-by-step instructions? We wrote a tutorial!

[Click here to do the thing @ https://learn.adafruit.com/adding-basic-audio-ouput-to-raspberry-pi-zero](https://learn.adafruit.com/adding-basic-audio-ouput-to-raspberry-pi-zero) (<https://adafru.it/jZD>)

## How Other Pi's Create Audio

GPIO #18 is also known as PWM0 and in the original Pi was coupled with a very basic RC filter to create the audio output:



If you don't mind getting a few 150 and 270 ohm resistors, and two each of about 33nF (also known as 0.033uF) and 10uF capacitors, you can basically recreate those two filters.

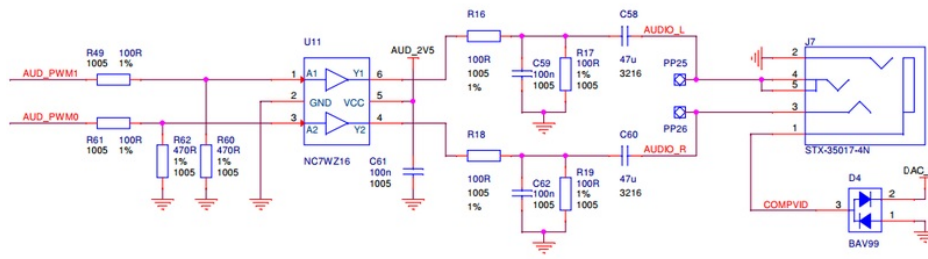
Now all you need is access to PWM0\_OUT and PWM1\_OUT, which are...on GPIO #40 and #45 and are not brought out on the Pi Zero. Tragedy? Give up? No! You can get to **PWM0** on GPIO #18 (ALT5) and **PWM1** on GPIO #13 (ALT0) or GPIO #19 (ALT5) - [see the full list of pins and alternate functions here \(https://adafru.it/jEa\)](https://adafru.it/jEa)

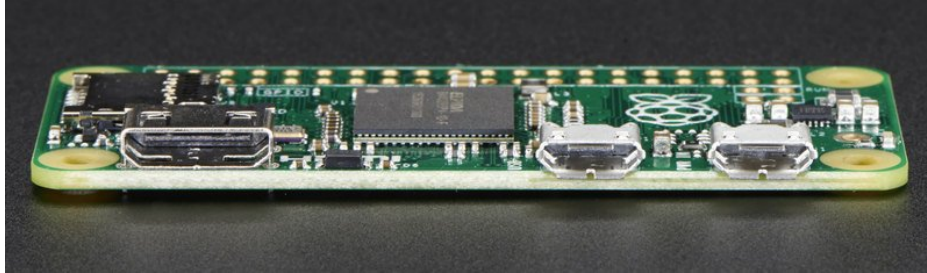
You can do *that* by adjusting the device tree overlay to change the PWM audio pins from pins #40 and #45 (which are not accessible) to pins #18 and #13 [This very nice Pi forum thread will tell you how! \(https://adafru.it/jEb\)](https://adafru.it/jEb)

See here for a [program that will let you set the alt forms of GPIO pins \(https://adafru.it/jEc\)](https://adafru.it/jEc)

If you want a higher quality audio output, the B+ and Pi 2 use this schematic - it has a driving buffer on the audio PWM lines for better current drive *and* it uses a cleaner 2.5V reference for better quality audio.







The most intriguing difference for hackers and makers is that the Pi Zero does not come with the soldered GPIO header. Partially this is to save cost, but it also allows the Pi Zero to be very thin and gives you the option of embedding it easily into a project box.

### Cons:

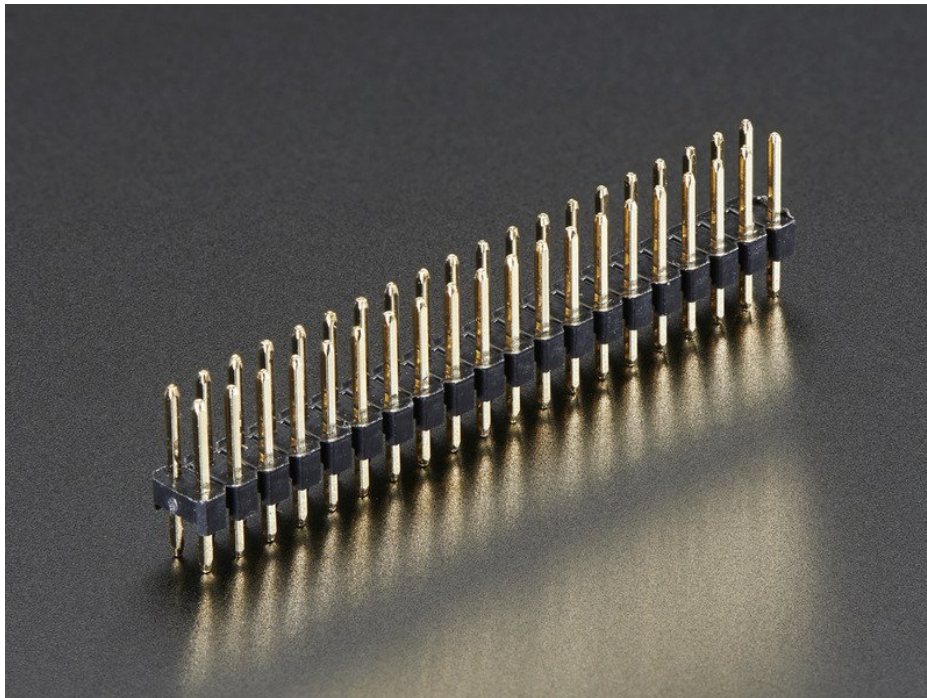
- You have to solder in the header to use Pi HATs and Pi toppers

### Pros:

- You can practice your soldering!
- Can skip the GPIO header to keep the Pi Zero super slim
- Solder wires directly into the GPIO pads, use only what you need
- Try different, exotic headers such as right angle or socket header

## Go Classic with 2x20 Male Header

Like blue jeans and Coca-Cola, [the 2x20 male header is the classic option.](http://adafru.it/2822) (<http://adafru.it/2822>)



Once soldered in, you can plug in any HAT or topper. The pinout is completely identical to the 2x20 headers on the Pi 2 and Pi A+ & B+



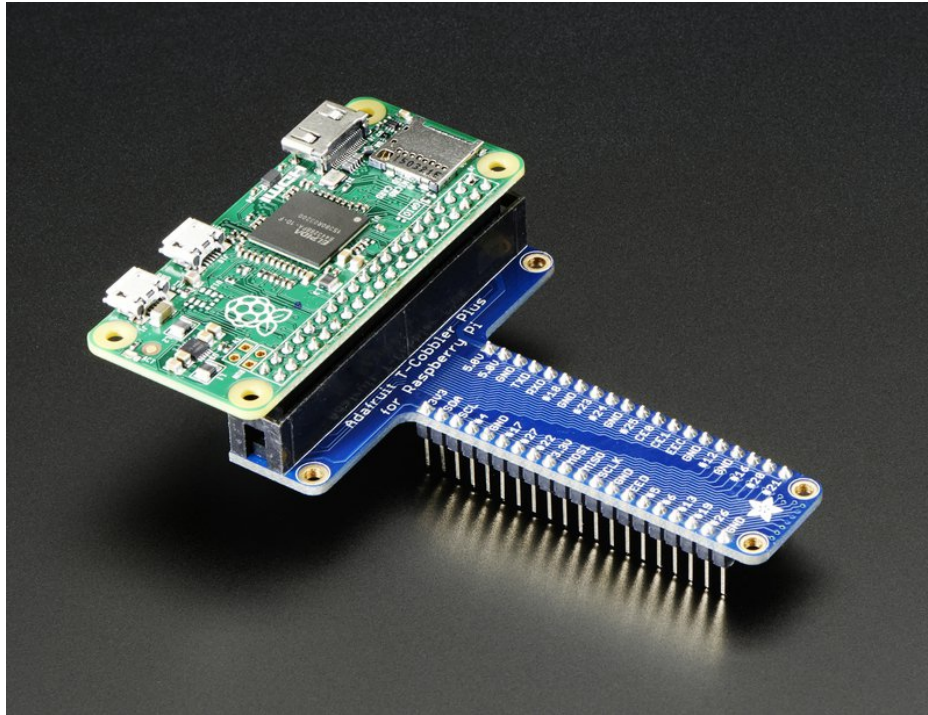
## Or 2x20 Female Socket Header

This one is interesting, [if you solder in a 2x20 female socket header](http://adafru.it/2222) (<http://adafru.it/2222>)



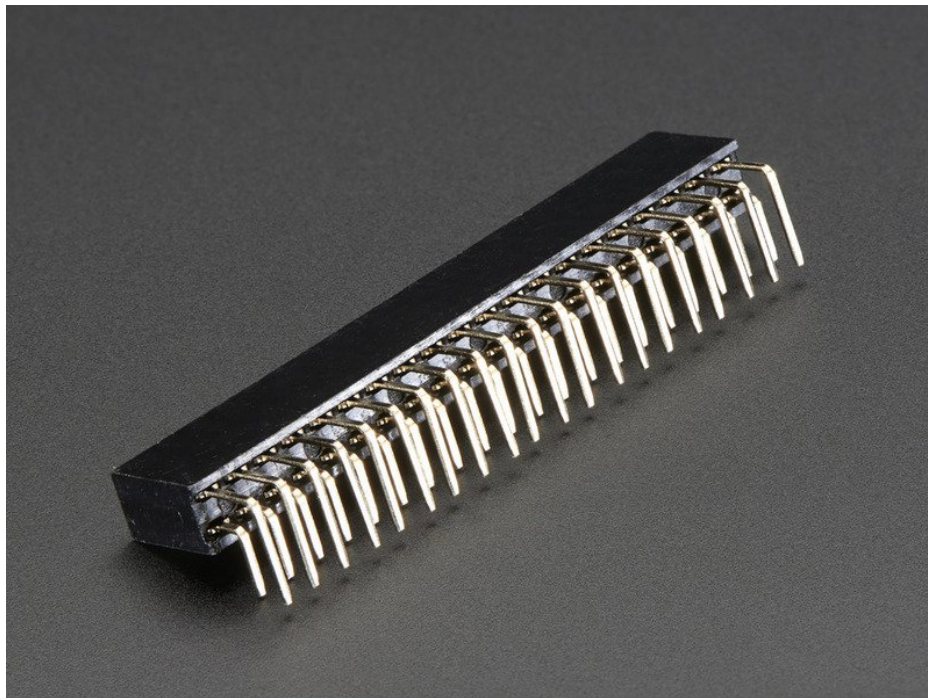
and attach it upside down you can plug it right into a T-Cobbler!



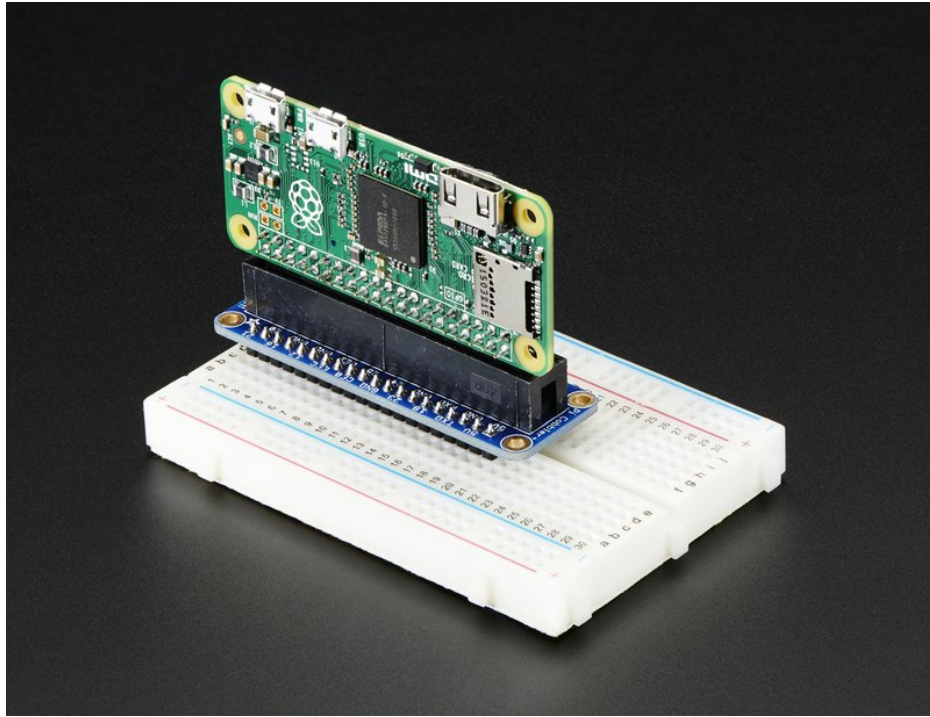


## Advanced 2x20 Right-Angle Female Socket Header

Or, take it even more *extreme* with [2x20 right angle female header](http://adafru.it/2823) (<http://adafru.it/2823>)



Now you can stick it into a Cobbler or T-Cobbler and it will sit sort of like a computer daughtercard!



## Is My Pi Zero Dead?

The Pi Zero is so minimal, it can be tough to tell if its working at all. Here's how to do a quick check (from [this sticky \(https://adafru.it/upa\)](https://adafru.it/upa))!

- Take your Zero, with nothing in any slot or socket (yes, no SD-card is needed or wanted to do this test!).
- Take a normal micro-USB to USB-A **DATA SYNC** cable (not a charge-only cable! make sure its a true data sync cable!)
- Connect the USB cable to your PC, plugging the micro-USB into the Pi's USB, (*not the PWR\_IM*).
- If the Zero is alive, your Windows PC will go ding for the presence of new hardware & you should see "BCM2708 Boot" in Device Manager.
- Or on linux, run `sudo lsusb` or run `dmesg` and look for a `ID 0a5c:2763 Broadcom Corp` message. If you see that, so far so good, you know the Zero's not dead.

I tested on Linux and here's my actual dmesg:

```
[226314.048026] usb 4-2: new full-speed USB device number 82 using uhci_hcd
[226314.213273] usb 4-2: New USB device found, idVendor=0a5c, idProduct=2763
[226314.213280] usb 4-2: New USB device strings: Mfr=1, Product=2, SerialNumber=0
[226314.213284] usb 4-2: Product: BCM2708 Boot
[226314.213288] usb 4-2: Manufacturer: Broadcom
```

but there will be NO LED LIGHT (looks so dead but its alive!)

## Is There Even Life?

You can skip this section unless you have reason to believe your Pi Zero isn't alive.

### THE ZERO DOES NOT HAVE A POWER LED

The Pi Zero doesn't have much in the way of blinky LEDs to give you a warm fuzzy that it's doing anything or even alive. And if the GPU doesn't find a valid OS image, it doesn't even turn on the green ACT LED and looks totally dead. Typically this just means something is up with the SD card. Bad card. Bad image. Out of date image. Whatever. **It does not mean the Pi Zero is dead.**

### Here's how to run a sanity check to verify if the Pi Zero is OK.

(taken from [here \(https://adafru.it/upa\)](https://adafru.it/upa) and also provided [here \(https://adafru.it/vle\)](https://adafru.it/vle))

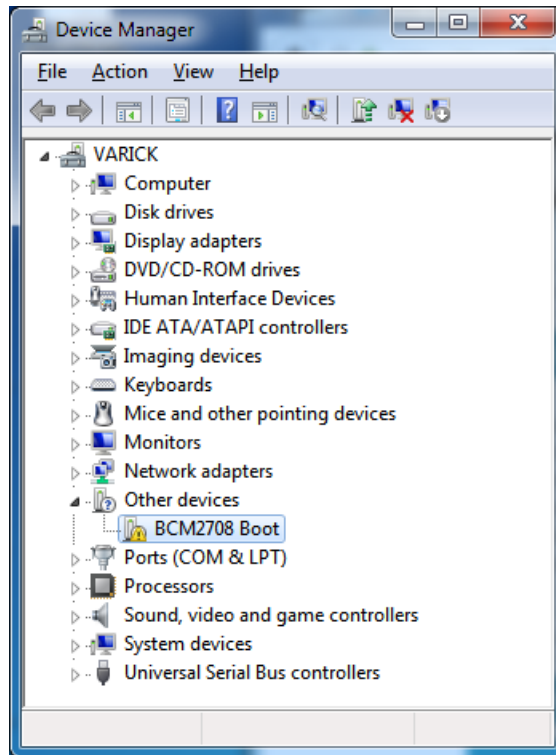
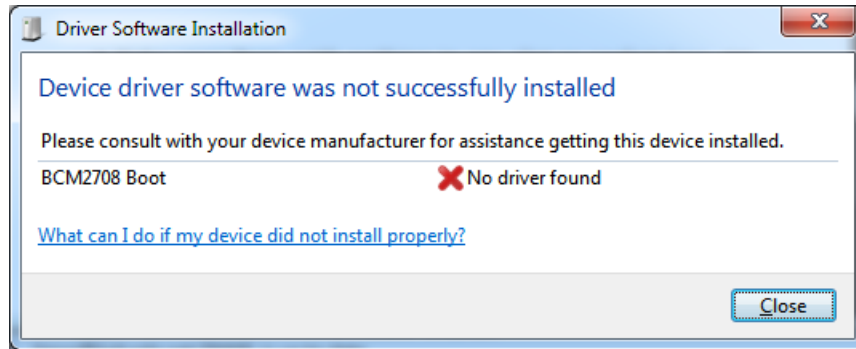
- Take your Zero, with nothing in any slot or socket (yes, **no SD-card is needed** or wanted to do this test!).
- Take a normal micro-USB to USB-A **DATA SYNC** cable (not a charge-only cable! make sure its a true data sync cable!)
- Connect the USB cable to your PC, plugging the micro-USB into the Pi's USB, (*not the PWR\_IM*).
- If the Zero is alive, your Windows PC will go ding for the presence of new hardware & you should see "BCM2708 Boot" in Device Manager.
- Or on linux, run `sudo lsusb` or run `dmesg` and look for a `ID 0a5c:2763 Broadcom Corp` message. If you see that, so far so good, you know the Zero's not dead.

It may take a few seconds for the messages to show up.

Below is a Pi Zero connected to a Linux computer via a USB cable and the resulting dmesg output. **Note: there is no SD card installed, USB cable is in USB port, and there are no lights.**



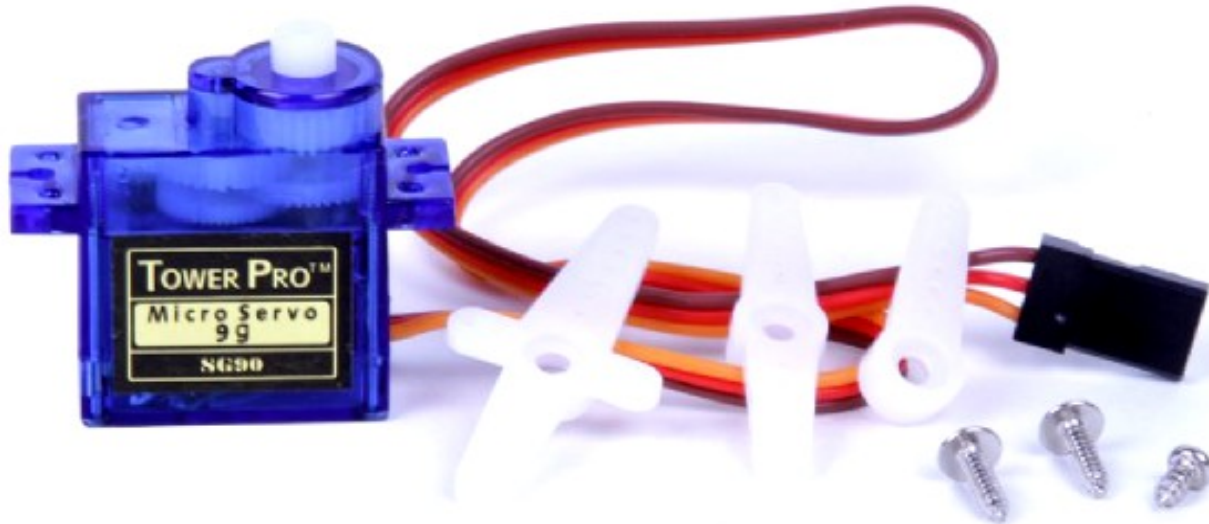
Here's what our Windows machine showed:



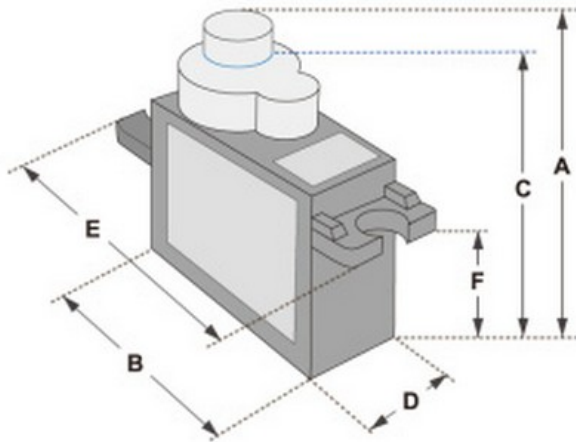
Looks dead, but it's not.







Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction), and works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.



Dimensions & Specifications
A (mm) : 32
B (mm) : 23
C (mm) : 28.5
D (mm) : 12
E (mm) : 32
F (mm) : 19.5
Speed (sec) : 0.1
Torque (kg-cm) : 2.5
Weight (g) : 14.7
Voltage : 4.8 - 6

Position "0" (1.5 ms pulse) is middle, "90" (~2ms pulse) is middle, is all the way to the right, "-90" (~1ms pulse) is all the way to the left.

