ΠΑΝΕΠΙΣΤΗΜΙΟ ΜΑΚΕΔΟΝΙΑΣ
ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
ΤΜΗΜΑΤΟΣ ΕΦΑΡΜΟΣΜΕΝΗΣ ΠΛΗΡΟΦΟΡΙΚΗΣ

ΣΧΕΔΙΑΣΗ ΚΑΙ ΑΝΑΠΤΥΞΗ ΥΛΙΚΟΥ-ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΣΥΣΤΗΜΑ ΕΛΕΓΧΟΥ ΛΕΙΤΟΥΡΓΙΩΝ ΣΕ ΟΧΗΜΑΤΑ

Διπλωματική Εργασία
του
Χριστέλη Χρήστου

Θεσσαλονίκη, 02/2016

ΣΧΕΔΙΑΣΗ ΚΑΙ ΑΝΑΠΤΥΞΗ ΥΛΙΚΟΥ-ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΣΥΣΤΗΜΑ ΕΛΕΓΧΟΥ ΛΕΙΤΟΥΡΓΙΩΝ ΣΕ ΟΧΗΜΑΤΑ

Χριστέλης Χρήστος

Πτυχίο Ηλεκτρονικού Μηχανικού, ΑΤΕΘ, 2012

Διπλωματική Εργασία

υποβαλλόμενη για τη μερική εκπλήρωση των απαιτήσεων του

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΤΙΤΛΟΥ ΣΠΟΥΔΩΝ ΣΤΗΝ ΕΦΑΡΜΟΣΜΕΝΗ ΠΛΗΡΟΦΟΡΙΚΗ

Επιβλέπων Καθηγητής:
Χατζηγεωργίου Αλέξανδρος

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την 17/02/2016

Μαργαρίτης Κωνσταντίνος	**Ξυνόγαλος Στέλιος**	**Χατζηγεωργίου Αλέξανδρος**
.....	.....	.....

Χριστέλης Χρήστος

.....

Στη γιαγιά και το παππού μου που έφυγε πρόσφατα

Περίληψη

Στον 21ο αιώνα ο κλάδος της πληροφορικής λόγω της ραγδαίας ανάπτυξης του την τελευταία δεκαπενταετία εισχώρησε σε άλλους επιστημονικούς κλάδους, με αποτέλεσμα να γίνει αναπόσπαστο κομμάτι τους. Μερικοί κλάδοι, οι οποίοι εξαρτώνται άμεσα από τη δυναμική της πληροφορικής και κατ' επέκταση των πληροφοριακών συστημάτων είναι η ιατρική, η βιομηχανία, η γεωργία, η εκπαίδευση και γενικότερα όλη σχεδόν η σύγχρονη δομή της κοινωνίας του 21ου αιώνα.

Λόγω της διεύρυνσης του τομέα της πληροφορικής σε άλλους τομείς, στα πλαίσια αυτής της διπλωματικής εργασίας μελετάται η υλοποίηση ενός συστήματος ελέγχου λειτουργιών σε οχήματα, το οποίο θα λαμβάνει κάποιες εντολές από τα αντίστοιχα αισθητήρια του αυτοκινήτου και θα δρα αντίστοιχα, με σκοπό την αποφυγή βλάβης στα μηχανικά μέρη του αυτοκινήτου, αλλά και την προστασία του οδηγού από εν δυνάμει ατυχήματα. Η υλοποίηση του συστήματος εκτός από το λογισμικό συμπεριλαμβάνει και τη σχεδίαση ηλεκτρονικού κυκλώματος - πλακέτας, με σκοπό την πειραματική δοκιμή του.

Τέλος, στο πλαίσιο της εργασίας διερευνώνται παρόμοιες προσεγγίσεις που έχουν καταγραφεί στη βιβλιογραφία, καθώς και οι απαιτούμενες τεχνολογίες τόσο σε επίπεδο υλικού όσο και σε επίπεδο λογισμικού.

Abstract

In the 21st century the computer science because of the rapid growth at the last fifteen years has penetrated in other sectors and thereby becoming an integral piece. Some sectors that are directly depended of the computer science dynamic are the medicine, the industry, the agriculture, the education and almost all the modern structure of society in the 21st century.

Due to the expansion of the computer science to other sectors, this thesis studies the implementation of a control system for vehicles, which will receive commands form the respective sensors of the car and will act respectively, in order to avoid a damage at the mechanical parts of the car and from the other hand to protect the driver from potential accidents. The implementation of the system except of the software part includes and hardware part, which is the electronic schematic and the PCB layout, in order to test it at laboratory level.

Finally, this thesis research similar implementations and technologies which have recorded in the literature at the both levels of software-hardware.

Πρόλογος - Ευχαριστίες

Λόγω των αυξημένων δυνατοτήτων που προσφέρει η πληροφορική στις μέρες μας ως προς το κομμάτι των αυτόματων και ημιαυτόματων συστημάτων θεώρησα σκόπιμο να εκμεταλλευτώ τις δυνατότητες αυτές και να τις εφαρμόσω σε κάτι που ασχολούμαι ερασιτεχνικά εδώ και χρόνια, το οποίο είναι το αυτοκίνητο και πιο συγκεκριμένα τα μηχανικά μέρη του αυτοκινήτου. Το σύστημα που ανεπτύχθει περιλαμβάνει λογισμικό και υλικό κομμάτι (πλακέτα). Στο υλικό κομμάτι δεν υπήρξαν ιδιαίτερες δυσκολίες λόγω της εμπειρίας και των γνώσεων λόγω των σπουδών μου ως ηλεκτρονικός μηχανικός (ΑΤΕΙΘ), αλλά στο λογισμικό κομμάτι υπήρξαν αρκετές δυσκολίες και ιδιαίτερα στο κομμάτι της επικοινωνίας μεταξύ των δύο συσκευών (ταμπλέτα - πλακέτα), οι οποίες βέβαια μου παρείχαν χρήσιμες γνώσεις και αυτό είναι πιστεύω το πιο σημαντικό όφελος από την ενασχόληση μου με τη παρούσα εργασία. Τέλος, θα ήθελα να ευχαριστήσω τον επιβλέπων καθηγητή μου, κύριο Αλέξανδρο Χατζηγεωργίου, για την παρότρυνση του να ασχοληθώ με το θέμα αυτό, καθώς και για τη καθοδήγηση του. Επίσης, θα ήθελα να ευχαριστήσω και το κύριο Ξυνόγαλο Στέλιο, ο οποίος με τις άρτια οργανωμένες παραδόσεις μαθημάτων του στην αντικειμενοστραφή γλώσσα Java μου έδωσε τις απαραίτητες βάσεις, ώστε να μπορέσω μετέπειτα να ασχοληθώ με την υλοποίηση εφαρμογών σε Android. Τέλος, θα ήθελα να

Περιεχόμενα

1 Εισαγωγή 10
1.1 Στόχοι 10
1.2 Διάρθρωση της εργασίας 11
2 Βιβλιογραφική Επισκόπηση 13
2.1 Android λογισμικό σε οχήματα 13
2.2 Τεχνολογίες Μικροελεγκτών 15
2.3 Android OS 18
2.3.1 Χαρακτηριστικά του Android 18
2.3.2 Εφαρμογές 19
2.3.3 Διαχείριση Μνήμης 20
2.3.4 ANDROID Software Development Kit (SDK) 22
2.3.5 Αρχιτεκτονική του ANDROID OS 23
2.3.6 Ιστορικό Εκδόσεων του ANDROID OS 25
3 Σχεδίαση εφαρμογής 33
3.1 Λειτουργικότητα συστήματος (hardware) 34
3.1.1 Είσοδοι συστήματος 35
3.1.2 Προσομοίωση αισθητηρίων 41
3.1.3 Έξοδοι συστήματος 42
3.2 Αρχιτεκτονική εφαρμογής Android 48
3.3 Πρωτόκολλο επικοινωνίας (Android USB) 51
4. Υλοποίηση – Ανάπτυξη Κώδικα 53
5 Παρουσίαση λειτουργικότητας του συστήματος 63
6 Συμπεράσματα 68
Βιβλιογραφία – Πηγές Διαδικτύου 69

Κατάλογος Εικόνων

Εικόνα 1 (Αυτοκίνητο της Google) 13
Εικόνα 2 (Γραφικό περιβάλλον Android Auto) 14
Εικόνα 3 (PDIP Package) 17
Εικόνα 4 (TSOP Package) 17
Εικόνα 5 (BGA Package) 17
Εικόνα 6 (Android 6.0 home screen) 19
Εικόνα 7 (Κύρια σελίδα του Google Play) 20
Εικόνα 8 (Activity lifecycle) 21
Εικόνα 9 (Το λογότυπο του Android SDK) 22
Εικόνα 10 (Η αρχιτεκτονική του Android OS) 24
Εικόνα 11 (Περιβάλλον χρήσης Android 1.0) 25
Εικόνα 12 (Λογότυπο Android 1.1) 26
Εικόνα 13 (Λογότυπο Android 1.5) 26
Εικόνα 14 (Η εσωτερική αναζήτηση) 27
Εικόνα 15 (Λογότυπο Android 2.0-2.1) 28
Εικόνα 16 (Λογότυπο Android 2.2-2.2.3) 28
Εικόνα 17 (Λογότυπο Android 2.3-2.3.7) 28
Εικόνα 18 (Λογότυπο Android 3.0-3.2) 29
Εικόνα 19 (Λογότυπο Android 4.0-4.0.4) 29
Εικόνα 20 (Λογότυπο Android 4.1-4.3) 30
Εικόνα 21 (Λογότυπο Android 4.4) 30
Εικόνα 22 (Android 5.0 home screen) 31
Εικόνα 23 (Android 6.0 home screen) 32
Εικόνα 24 (Hardware - PIC24F Accessory Development Starter Kit for Android) 33
Εικόνα 25 (Software - PIC24F Accessory Development Starter Kit for Android) 34
Εικόνα 26 (Μπλοκ διάγραμμα ενδείκτη μπαταρίας) 35
Εικόνα 27 (Πιστόνι αυτοκινήτου) 36
Εικόνα 28 (Μπλοκ διάγραμμα ενδείκτη λαδιού) 36
Εικόνα 29 (Μπλοκ διάγραμμα ενδείκτη θερμοκρασίας) 37
Εικόνα 30 (Μπλοκ διάγραμμα ενδείκτη χειρόφρενου) 37
Εικόνα 31 (Αισθητήρας ζώνης ασφαλείας) 38
Εικόνα 32 (Αισθητήρας θέσης) 38
Εικόνα 33 (Μπλοκ διάγραμμα ενδείκτη μεγάλης σκάλας) 39
Εικόνα 34 (Μπλοκ διάγραμμα δεξιού και αριστερού ενδείκτη φλάς) 40
Εικόνα 35 (Σχηματικό κύκλωμα push-button) 41
Εικόνα 36 (Ηλεκτρομαγνητική κλειδαριά) 42
Εικόνα 37 (Βεντιλατέρ τοποθετημένο στο ψυγείο του αυτοκινήτου) 43
Εικόνα 38 (Θερμαινόμενο τζάμι) 44
Εικόνα 39 (Μηχανισμός ανοίγματος/κλεισίματος πορτ-μπαγκάζ) 44
Εικόνα 40 (Διακόπτης πόρτας αυτοκινήτου) 45
Εικόνα 41 (Σχηματικό κύκλωμα πλαφονιέρας αυτοκινήτου) 45

Εικόνα 42 (Αλάρμ αυτοκινήτου) 46
Εικόνα 43 (Διακόπτης αλάρμ) 46
Εικόνα 44 (Διακόπτης εκκίνησης) 46
Εικόνα 45 (Μίζα) 47
Εικόνα 46 (Κύκλωμα οδήγησης ρελέ) 47
Εικόνα 47 (SplashScreenActivity) 48
Εικόνα 48 (PasswordActivity) 49
Εικόνα 49 (BasicAccessoryDemo - Activity) 50
Εικόνα 50 (InfoActivity) 50
Εικόνα 51 (USB Host and Accessory Modes) 52
Εικόνα 52 (Οθόνη υποδοχής) 63
Εικόνα 53 (Οθόνη εισαγωγής κωδικού) 64
Εικόνα 54 (Οθόνη - καντράν αυτοκινήτου) 64
Εικόνα 55 (Οθόνη - καντράν αυτοκινήτου) 65
Εικόνα 56 (Σχηματικό πλακέτας) 66
Εικόνα 57 (Μπλοκ διάγραμμα πλακέτας) 67

1 Εισαγωγή

1.1 Στόχοι

Ο κλάδος της αυτοκινητοβιομηχανίας τα τελευταία χρόνια αναπτύσσεται ραγδαία προσφέροντες όλο και περισσότερες δυνατότητες στους οδηγούς των αυτοκινήτων, καθώς και γενικότερα στους ανθρώπους που επιβαίνουν σε αυτό. Οι δυνατότητες αυτές αφορούν την ασφάλεια των μηχανικών μερών του αυτοκινήτου από βλάβες, την ασφάλεια των επιβατών από εν δυνάμει ατυχήματα, είτε λόγω βλάβης του αυτοκινήτου, είτε λόγω απροσεξίας του οδηγού, αλλά και την ψυχαγωγία των επιβατών με σκοπό μία ευχάριστη διαδρομή. Οι διεργασίες και οι έλεγχοι που γίνονται για την επίτευξη των προαναφερθεισών θα ήταν αδύνατοι χωρίς τη παρουσία ηλεκτρονικών στοιχείων και χωρίς τη παρουσία κάποιου μικροελεγκτή, ο οποίος να επικοινωνεί με τα αντίστοιχα αισθητήρια του αυτοκινήτου, ώστε να δρα πλήρως αυτόματα σε κάποιες περιπτώσεις και σε κάποιες άλλες με τη συμβολή του οδηγού.

Για αυτούς τους λογούς, στόχος της εργασίας είναι η κατασκευή ενός συστήματος, το οποίο θα περιλαμβάνει μία μονάδα (πλακέτα), η οποία θα επικοινωνεί με τα αισθητήρια του αυτοκινήτου και θα ενημερώνει τον οδηγό μέσω τη οθόνης ενός τάμπλετ για την αντίστοιχη κατάσταση των αισθητηρίων με εικονικό, καθώς και με φωνητικό τρόπο, με σκοπό την αποφυγή δυσάρεστων καταστάσεων τόσο για τους επιβάτες του αυτοκινήτου, όσο και για το ίδιο το αυτοκίνητο. Κύριος στόχος είναι η ενσωμάτωση ενός τάμπλετ στο αυτοκίνητο, όπου ουσιαστικά είναι μία συσκευή με υπολογιστική ισχύ και δυνατότητες συγκρίσιμες με αυτές ενός προσωπικού υπολογιστή. Έτσι, εκτός του ότι με τη χρήση ενός τάμπλετ θα πραγματοποιούνται οι παραπάνω έλεγχοι και θα είναι διαθέσιμες εφαρμογές ψυχαγωγίας και επικοινωνίας (αναπαραγωγή μουσικής, βιντεοκλήσεις, email κτλ.), θα υπάρχει και το μεγάλο πλεονέκτημα ότι όλα αυτά θα είναι εύκολα τροποποιήσιμα και μάλιστα χωρίς το αυτοκίνητο να πάει σε κάποιο συνεργείο. Για παράδειγμα πολλές φορές οι ιδιοκτήτες των αυτοκινήτων θέλουν να έχουν κάποιες έξτρα ενδείξεις σχετικά με τη λειτουργία της μηχανής, όπως για παράδειγμα τη πίεση σε bar του λαδιού ή τη στάθμη σε volt της μπαταρίας, με αποτέλεσμα να χρειάζεται να τοποθετήσουν σε κάποιο σημείο του αυτοκινήτου τα αντίστοιχα όργανα απεικόνισης τους. Με τη χρήση του τάμπλετ, όμως, παρέχετε η δυνατότητα οι απεικονίσεις των μετρήσεων αυτών να γίνονται στην οθόνη του τάμπλετ, μειώνοντας έτσι το κόστος, αφού δεν απαιτούνται επιπλέον όργανα για την απεικόνιση. Επιπλέον, ένα ακόμη πλεονέκτημα είναι ότι ο οδηγός δε χρειάζεται να εστιάζει σε διάφορες περιοχές του αυτοκινήτου με σκοπό να ελέγχει όλα τα όργανα απεικόνισης του αυτοκινήτου.

Άρα ο ένας βασικός στόχος της εργασίας είναι η υλοποίηση ενός συστήματος που θα επικοινωνεί με τα αισθητήρια του αυτοκινήτου και ταυτόχρονα θα παρέχει στον οδηγό και στους συνεπιβάτες του δυνατότητες, όπως αυτές της ψυχαγωγίας και της

επικοινωνίας και ο άλλος στόχος, τα συστήματα ελέγχου των αυτοκινήτων να καταστούν εύκολα τροποποιήσιμα.

1.2 Διάρθρωση της εργασίας

Λόγω του ότι το σύστημα που αναπτύχθηκε δε σχετίζεται μόνο με το τομέα της πληροφορικής, αλλά και τους τομείς των ηλεκτρονικών και των οχημάτων, στα επόμενα κεφάλαια, γίνονται αναφορές στις τεχνολογίες που χρησιμοποιήθηκαν, ώστε να υλοποιηθεί το σύστημα ελέγχου των λειτουργιών ενός οχήματος, καθώς στα στάδια που ακολούθησαν για την υλοποίηση του.

Πιο συγκεκριμένα, στο δεύτερο κεφάλαιο γίνεται μία βιβλιογραφική επισκόπηση, όπου αναφέρονται παρόμοιες τεχνολογίες και εφαρμογές συστημάτων ελέγχου, τα οποία αλληλοεπιδρούν με τις λειτουργίες των οχημάτων, όπου αυτές οι τεχνολογίες είναι είτε σε πειραματικό στάδιο ακόμα, είτε κυκλοφορούν ήδη στην αγορά. Επίσης, στο ίδιο κεφάλαιο αναφέρονται κάποιες από τις τεχνολογίες των μικροελεγκτών που κυκλοφορούν στο εμπόριο και έχουν την επεξεργαστική, ώστε να αναπτυχθούν τέτοιου είδους συστήματα. Επειδή, όμως, το κομμάτι του μικροελεγκτή αποτελεί το ένα μέρος της δομής της εφαρμογής, στη συνέχεια γίνεται αναφορά και στο άλλο κομμάτι, το οποίο είναι το λειτουργικό σύστημα (Operating System - OS) Android. Εξαιτίας του ότι το λειτουργικό σύστημα Android είναι ανοιχτού κώδικα (open source) και πλέον είναι ευρέως διαδεδομένο τόσο σε επίπεδο προγραμματιστών, αλλά και σε επίπεδο χρηστών, υπάρχει ένα μεγάλο πλήθος βιβλιογραφικών αναφορών επάνω σε αυτό, και για αυτούς του λόγους γίνεται μία γενική αναφορά σε αυτό.

Στο τρίτο κεφάλαιο αρχικά, γίνεται αναφορά στο υλικό κομμάτι της κατασκευής, το οποίο αποτελείται από τη πλακέτα και τα αισθητήρια του αυτοκινήτου (τα οποία σχετίζονται με την εφαρμογή). Όσο αφορά τα αισθητήρια του αυτοκινήτου γίνεται αναφορά ως προς τη λειτουργικότητα και τη δομή τους, καθώς και πως αυτά προσομοιώνονται σε πειραματικό επίπεδο. Εν συνεχεία, περιγράφεται η δομή της Android εφαρμογής με λειτουργικά διαγράμματα, καθώς και αναφορικά, όπως ποια activities περιλαμβάνει η εφαρμογή και πως αυτά τα activities συνδέονται μεταξύ τους. Επιπλέον, προς το τέλος του κεφαλαίου περιγράφεται το πρωτόκολλο επικοινωνίας που χρησιμοποιείται για την επικοινωνία των δύο συσκευών, όπου η μία συσκευή είναι η πλακέτα και η άλλη το τάμπλετ.

Στο τέταρτο κεφάλαιο, γίνεται μία πιο εκτενή αναφορά όσο αφορά το λογισμικό της κατασκευής και ιδιαίτερα σε μέρη κώδικα, τα οποία αποτελούν βασικά λειτουργικά κομμάτια του όλου λογισμικού, καθώς και σε μέρη κώδικα.

Στο επόμενο κεφάλαιο παρουσιάζετε το λογισμικό ως προς τη λειτουργικότητα του με τη χρήση screenshots. Επίσης, γίνεται παρουσίαση της πλακέτας με τη χρήση σχηματικών διαγραμμάτων και μπλοκ διαγραμμάτων.

Τέλος, το έκτο κεφάλαιο περιλαμβάνει τα συμπεράσματα που προέκυψαν από την υλοποίηση της εργασίας, όπως και προτάσεις για βελτιώσεις και μελλοντική επέκταση του συστήματος.

2 Βιβλιογραφική Επισκόπηση

2.1 Android λογισμικό σε οχήματα

Λόγω της ραγδαίας ανάπτυξης του λογισμικού Android τα τελευταία χρόνια και της μεγάλης αποδοχής που έχει από χρήστες έξυπνων συσκευών (smartphones, tablets κτλ.) σε παγκόσμιο επίπεδο, πλέον έχουν αρχίσει να γίνονται προσπάθειες και έρευνες, ώστε το λειτουργικό Android να χρησιμοποιηθεί ως «κυρίως» λογισμικό και σε άλλες συσκευές πέρα των προσωπικών-έξυπνων συσκευών. Ένας, κλάδος στον οποίο η εταιρία Google (η οποία και έχει υπό τη κατοχή της το Android) θέλει να ενσωματώσει το λογισμικό Android είναι αυτός των οχημάτων και ιδιαίτερα των αυτοκινήτων. Ως, προς αυτή τη κατεύθυνση έχει επενδύσει σε έρευνα μεγάλα χρηματικά ποσά με σκοπό τη δημιουργία μη επανδρωμένων ηλεκτρικών αυτοκινήτων (driverless cars), των οποίων το λογισμικό και οι εφαρμογές θα στηρίζονται σε κομμάτια του Android.

Η ερευνητική ομάδα της Google έχει εξοπλίσει μια σειρά από διαφορετικούς τύπους αυτοκινήτων συμπεριλαμβανομένης της Toyota Prius, Audi TT, και Lexus RX450h με τον εξοπλισμό, ο οποίος θα καθιστά τα αυτοκίνητα αυτό-οδηγούμενα. Επίσης, η Google έχει αναπτύξει ένα δικό της όχημα με συγκεκριμένες προδιαγραφές, το οποίο κατασκευάζεται από τις επιχειρήσεις Roush και χρησιμοποιεί τον εξοπλισμό από την Bosch, ZF Lenksysteme, LG, και Continental.

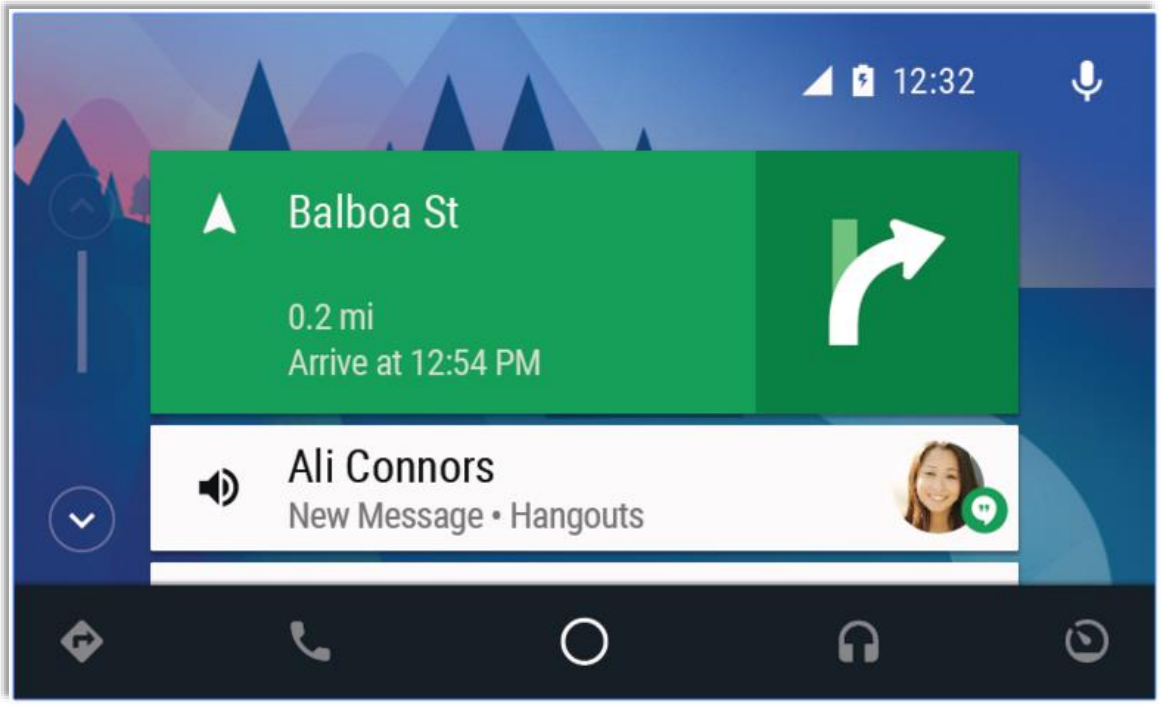
Εικόνα 1 ***(Αυτοκίνητο της Google)***

Τα ρομποτικά αυτοκίνητα της Google «εικόνα 1» έχουν περίπου 150.000$ σε εξοπλισμό συμπεριλαμβανομένου ενός συστήματος LIDAR 70.000$. Ο ανιχνευτής χώρου (range finder) είναι τοποθετημένος στον ουρανό του αυτοκινήτου και περιλαμβάνει ένα Velodyne λέιζερ 64-beam. Αυτό το λέιζερ επιτρέπει στο όχημα να δημιουργεί ένα χάρτη

3D του περιβάλλοντός με μεγάλη λεπτομέρεια. Το αυτοκίνητο εν συνεχεία, παίρνει αυτούς τους 3D χάρτες που δημιουργήθηκαν και τους συνδυάζει με χάρτες υψηλής ανάλυσης του κόσμου που ήδη έχει, παράγοντας έτσι διαφορετικούς τύπους μοντέλων και δεδομένων, τα οποία επιτρέπουν στο αυτοκίνητο να οδηγεί από μόνο του χωρίς τη παρέμβαση ανθρώπου. Η Google σχεδιάζει να βγάλει στη δημόσια κυκλοφορία τα αυτό-οδηγούμενα αυτοκίνητα μέσα στο 2020.

Πηγή: wikipedia - Google self-driving car

Εάν και τα οχήματα της Google θέλουν κάποια χρόνια ακόμα έως ότου να βγουν στην αγορά, παρόλα αυτά η Google δίνει τη δυνατότητα στου οδηγούς, οι οποίοι είναι κάτοχοι αυτοκινήτων, που υποστηρίζουν σύνδεση του ταμπλό τους με smartphone, να χρησιμοποιήσουν την εφαρμογή "Android Auto" «εικόνα 2».



Εικόνα 2 **(Γραφικό περιβάλλον Android Auto)**

Το Android Auto είναι ένα πρότυπο προβολής smartphone που αναπτύχθηκε από την Google για να επιτρέψει κινητές συσκευές που τρέχουν το λειτουργικό σύστημα Android (έκδοση 5.0 "Lollipop" και πάνω) να λειτουργούν και να αλληλοεπιδρούν με την κεντρική μονάδα (ταμπλό) των αυτοκινήτων. Το Android Auto ανακοινώθηκε στις 25

Ιουνίου του 2014, στο Google I/O 2014. Η εφαρμογή Android Auto για smartphone κυκλοφόρησε στις 19 Μαρτίου 2015.

Το Android Auto προσφέρει στους οδηγούς τη δυνατότητα να ελέγχουν το GPS, την αναπαραγωγή μουσικής, τα SMS μηνύματα, τα τηλέφωνα/κλήσεις, και την αναζήτηση στο διαδίκτυο. Τρόποι αλληλεπίδρασης μεταξύ της συσκευής στο αυτοκίνητο είναι η οθόνη αφής, καθώς και η λειτουργία hands-free μέσω φωνητικών, η οποία ως σκοπό έχει την ασφαλή οδήγηση. Άλλες συμβατές εφαρμογές, οι οποίες μπορούν να προβληθούν στο ταμπλό του αυτοκινήτου και να χρησιμοποιηθούν είναι: Google Maps, Μουσική, MLB at Bat, Google Play Music, Spotify, Songza, Stitcher, iHeart Radio και TuneIn.

Εάν και η επίσημη ιστοσελίδα του Android Auto έχει λίστα με τους περισσότερους κατασκευαστές αυτοκινήτων, οι οποίοι υποστηρίζουν το Android Auto, παρόλα αυτά πολλά οχήματα που έχουν κυκλοφορήσει και είναι μοντέλα του 2015 και 2016 δεν το υποστηρίζουν ακόμη επίσημα.

Η εφαρμογή Android Auto είναι μέρος του Open Automotive Alliance, το οποίο είχε ανακοινωθεί στις 6 Ιανουαρίου του 2014, και είναι μια κοινή προσπάθεια με 28 κατασκευαστές αυτοκινήτων και προμηθευτών κινητών συσκευών Nvidia. Τον Μάιο του 2015, η Hyundai έγινε ο πρώτος κατασκευαστής που προσφέρει το Android Auto, καθιστώντας το διαθέσιμο για πρώτη φορά στο Hyundai Sonata (μοντέλο του 2015).

Πηγή: wikipedia - Android Auto

2.2 Τεχνολογίες Μικροελεγκτών

Στην αγορά υπάρχει μία πληθώρα μικροελεγκτών από διάφορες εταιρίες, που δραστηριοποιούνται στο τομέα αυτό. Δύο από τα πολύ βασικά χαρακτηριστικά ενός μικροελεγκτή είναι η συχνότητα λειτουργίας του και η αρχιτεκτονική του (π.χ. 8bit), όπου και τα δύο παίζουν σημαντικό ρόλο στο κόστος κατασκευής του. Κάποια ακόμη βασικά χαρακτηριστικά ενός μικροελεγκτή είναι οι ψηφιακές είσοδοι και έξοδοι που περιέχει, οι αναλογικές είσοδοι, το μέγεθος της flash μνήμης του στην οποία και αποθηκεύετε το πρόγραμμα που τρέχει ο μικροελεγκτής, το μέγεθος της EEPROM μνήμης του κ.α.

Πιο συγκεκριμένα το ολοκληρωμένο «chip» του μικροελεγκτή περιλαμβάνει ένα αριθμό από ποδαράκια «pins», όπου κάποια από αυτά (ανάλογα με το τύπο του μικροελεγκτή), μπορούν να οριστούν προγραμματιστικά, είτε ως είσοδοι ή είτε ως έξοδοι. Όταν ένα ποδαράκι οριστεί ως είσοδος τότε η αντίσταση εισόδου του είναι πολύ μεγάλη (της τάξης των MΩ) και άρα τραβάει πολύ λίγο ρεύμα, οπότε δεν είναι τόσο ευαίσθητο σε καταστροφή. Αντιθέτως, όταν ένα ποδαράκι οριστεί ως έξοδος τότε η αντίσταση εξόδου του είναι μικρή και για αυτό το λόγο χρειάζεται μεγάλη προσοχή στο τι φορτίο θα συνδεθεί σε κάθε έξοδο του μικροελεγκτή, γιατί εάν το φορτίο είναι μικρό και τραβάει ρεύμα μεγαλύτερο από εκείνο, το οποίο μπορεί να παρέχει με ασφάλεια η έξοδος του

μικροελεγκτή (το μέγιστο ρεύμα εξόδου διαφέρει από κάθε τύπο και μάρκα μικροελεγκτή και η μέγιστη τιμή αυτού αναφέρεται στα αντίστοιχα datasheet) τότε στη καλύτερη περίπτωση υπάρχει κίνδυνος να καεί η συγκεκριμένη έξοδος και στη χειρότερη ο ίδιος ο μικροελεγκτής.

Η συχνότητα λειτουργίας ενός μικροελεγκτή καθορίζεται και αυτή προγραμματιστικά (σύμφωνα πάντα με τις προδιαγραφές του κάθε μικροελεγκτή). Η συχνότητα λειτουργείας αναφέρεται ουσιαστικά στο σήμα χρονισμού (clock) του πυρήνα του μικροελεγκτή και όσο πιο μεγάλη είναι τόσο ταχύτερα εκτελούνται οι εντολές και οι διάφορες διεργασίες στο πυρήνα του. Για τη παραγωγή του σήματος χρονισμού υπάρχουν τρεις δυνατότητες συνήθως. Μία επιλογή είναι να χρησιμοποιηθεί το εσωτερικό RC κύκλωμα, το οποίο διαθέτει ο μικροελεγκτής και μπορεί να παρέχει κάποιες μικρές σε τιμή συχνότητες με το μειονέκτημα, όμως, ότι το RC κύκλωμα δεν είναι ακριβές στην παραγωγή του παλμού χρονισμού, με αποτέλεσμα για παράδειγμα εάν κάποιος φτιάξει ένα ρολόι, το ρολόι αυτό να έχει σημαντικές αποκλίσεις στη μέτρηση του χρόνου. Μία δεύτερη επιλογή είναι να χρησιμοποιηθεί ένα κύκλωμα εξωτερικής γεννήτριας παραγωγής σήματος χρονισμού, και μία τρίτη δυνατότητα και η πιο ευρέως χρησιμοποιούμενη είναι η δημιουργία σήματος χρονισμού με τη σύνδεση ενός ηλεκτρονικού εξαρτήματος, το οποίο ονομάζεται κρύσταλλος (crystal - xtal) και παρέχει μεγάλη ακρίβεια χωρίς διακυμάνσεις σε περιβάλλοντα τα οποία δεν παρουσιάζουν έντονες διακυμάνσεις στη θερμοκρασία.

Οι μνήμες που μπορεί να περιλαμβάνει ένας μικροελεγκτής είναι συνήθως τριών ειδών. Μία είναι η μνήμη flash, στην οποία κυρίως κατά βάση αποθηκεύτε το πρόγραμμα που θα εκτελεί ο μικροελεγκτής. Μερικές εταιρίες μικροελεγκτών παρέχουν τη δυνατότητα επέκτασης αυτής της μνήμης με τη χρήση μία εξωτερικής μνήμης flash. Η δεύτερη μνήμη είναι η μνήμη EEPROM, όπου γίνεται αποθήκευση δεδομένων, τα οποία αφορούν το πρόγραμμα (π.χ. τα ψηφία ενός κωδικού εισόδου). Η τρίτη μνήμη είναι η μνήμη RAM, η οποία χρησιμεύει στην προσωρινή αποθήκευση αποτελεσμάτων, τα οποία προκύπτουν κατά τη διάρκεια εκτέλεσης του κώδικα του μικροελεγκτή (π.χ. το αποτέλεσμα μίας σειράς από αριθμητικές πράξεις).

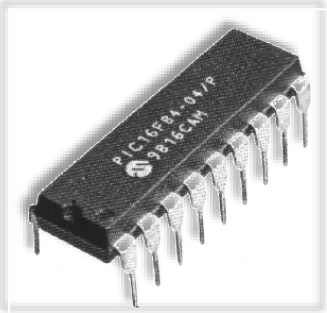
Κάθε μικροελεγκτής ανάλογα με τον αριθμό των ακίδων (pins) που έχει ή από τις δυνατότητες της επεξεργαστικής ισχύος που έχει, καθώς και από άλλους παράγοντες, «έρχεται» και σε διαφορετική μορφή όσο αναφορά τη συσκευασία (footprint – package), όπου στο εσωτερικό της είναι σχεδιασμένο με laser το όλο κύκλωμα του μικροελεγκτή. Σχετικά με το τύπο της συσκευασίας (footprint – package) του ολοκληρωμένου, υπάρχει μία πληθώρα διαφορετικών συσκευασιών από κάθε κατασκευάστρια εταιρία με τις αντίστοιχες προδιαγραφές και ιδιαιτερότητες. Επίσης, ένας τύπος μικροελεγκτή μπορεί να είναι διαθέσιμος σε περισσότερους από ένα τύπο συσκευασίας.

Σχετικά με το προγραμματισμό των μικροελεγκτών, σε επίπεδο λογισμικού (software) απαραίτητο είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (Integrated Development Environment - IDE), καθώς και ένας μεταγλωττιστής (compiler), που

παράγει το «.hex» αρχείο. Κάθε κατασκευάστρια εταιρία παρέχει δικό της ολοκληρωμένο περιβάλλον ανάπτυξης, όπως και μεταγλωττιστή. Εκτός, από το λογισμικό για την ανάπτυξη του κώδικα, απαραίτητη είναι και μία συσκευή προγραμματισμού (programmer) για το προγραμματισμό του μικροελεγκτή. Η συσκευή προγραμματισμού ουσιαστικά «φορτώνει» στο μικροελεγκτή το «.hex» αρχείο, το οποίο παράγει ο μεταγλωττιστής. Όσο αφορά τη συσκευή προγραμματισμού, η κάθε εταιρία παρέχει τη δική της, αλλά υπάρχουν και άλλες universal συσκευές προγραμματισμού στο εμπόριο, οι οποίες υποστηρίζουν το προγραμματισμό περισσότερων τύπων και κατηγοριών μικροελεγκτών, οι οποίοι προέρχονται και από διαφορετικές εταιρίες.

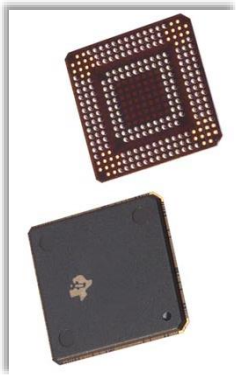
Παρακάτω ακολουθούν κάποιες εταιρίες, που ασχολούνται με τη κατασκευή μικροελεγκτών:

- ARM (δεν κατασκευάζει αλλά παραχωρεί δικαιώματα της αρχιτεκτονικής της)
- Texas Instruments
- Microchip
- Atmel
- Hitachi
- Maxim
- Toshiba



Εικόνα 3 *(PDIP Package)*

Εικόνα 4 *(TSOP Package)*



Εικόνα 5 *(BGA Package)*

2.3 Android OS

Το Android OS είναι ένα λειτουργικό σύστημα βασισμένο στον πυρήνα του Linux. Το περιβάλλον χρήσης του σχεδιάστηκε κυρίως για κινητές συσκευές με οθόνη αφής, όπως έξυπνα κινητά (smartphones) και tablets, με την χρήση εισόδων από επαφή, η οποίες έχουν σαν οπτικό αποτέλεσμα την μίμηση πραγματικών ενεργειών, όπως η κύλιση, με βασικό σκοπό τον χειρισμό αντικειμένων οθόνης και πληκτρολογίων οθόνης. Αν και αρχικά σχεδιάστηκε για είσοδο με αφή, έχει χρησιμοποιηθεί σε τηλεοράσεις, παιχνιδοκονσόλες, ψηφιακές κάμερες και άλλες ηλεκτρονικές συσκευές.

Από το 2011, το Android είναι το περισσότερο εγκατεστημένο λειτουργικό για κινητές συσκευές και από το 2013, οι συσκευές του καταλαμβάνουν μεγαλύτερο μερίδιο της αγοράς σε σύγκριση με τα λειτουργικά των άλλων εταιριών. Συνεπώς, είναι ένα λειτουργικό, το οποίο είναι ραγδαία εξελισσόμενο και οι προοπτικές χρήσης του συνεχώς αυξάνουν.

Ο πηγαίος κώδικας του Android OS κυκλοφόρησε από την Google υπό την προστασία αδειών ανοιχτού κώδικα, παρόλο που οι περισσότερες συσκευές εν τέλει συνδυάζουν τον ανοιχτό κώδικα και ιδιόκτητο λογισμικό.

Το λογισμικό αρχικά αναπτύχθηκε από την εταιρεία Android, Inc., αλλά η Google εξαγόρασε την εταιρεία το 2005. Το Android OS, όμως παρουσιάστηκε το 2007 παράλληλα με την ίδρυση της *Open Handset Alliance*, μιας κοινοπραξίας για εταιρίες υλικού, λογισμικού και τηλεπικοινωνιών με σκοπό την προαγωγή των ανοικτών προτύπων για κινητές συσκευές.

Πηγή: wikipedia - Android (operating system)

2.3.1 Χαρακτηριστικά του Android

Το Android έχει κάποια βασικά χαρακτηριστικά που το κάνουν κατάλληλο για την εφαρμογή του σε κινητές συσκευές. Μερικά από αυτά τα χαρακτηριστικά θα αναφερθούν παρακάτω.

Όπως σημειώθηκε παραπάνω το Android δημιουργήθηκε κατά βάση για να χρησιμοποιηθεί σε κινητές συσκευές με οθόνη αφής. Συνεπώς, με μια κίνηση του χρήστη μπορεί να ενεργοποιηθεί μια λειτουργία για την ενημέρωσή του. Οι λειτουργίες αυτές μπορεί να είναι η εμφάνιση μηνυμάτων στην οθόνη, η ενεργοποίηση κάποιας φωτεινής ένδειξης ή η δόνηση της συσκευής. Με το πάτημα της οθόνης ο χρήστης μπορεί να ανοίξει κάποια εφαρμογή, να δει τα email του και να ακούσει την αγαπημένη του μουσική.

Στην αρχική οθόνη, ο χρήστης μπορεί να τοποθετήσει της αγαπημένες του εφαρμογές, όπως στην επιφάνεια εργασίας ενός υπολογιστή. Έτσι, μπορεί να προσπελάσει απευθείας τα μηνύματά του ή να πλοηγηθεί στο διαδίκτυο με την ενεργοποίηση της κατάλληλης εφαρμογής.

Βασικό χαρακτηριστικό του περιβάλλοντος χρήσης είναι μπάρα ειδοποιήσεων. Η μπάρα ειδοποιήσεων βρίσκεται στο πάνω μέρος της οθόνης και εμφανίζει σχετικά μηνύματα όταν ο χρήστης π.χ. λάβει ένα νέο μήνυμα ή αν έχει χαμηλή μπαταρία ή αν έχει συνδεθεί σε κάποιο ασύρματο δίκτυο.

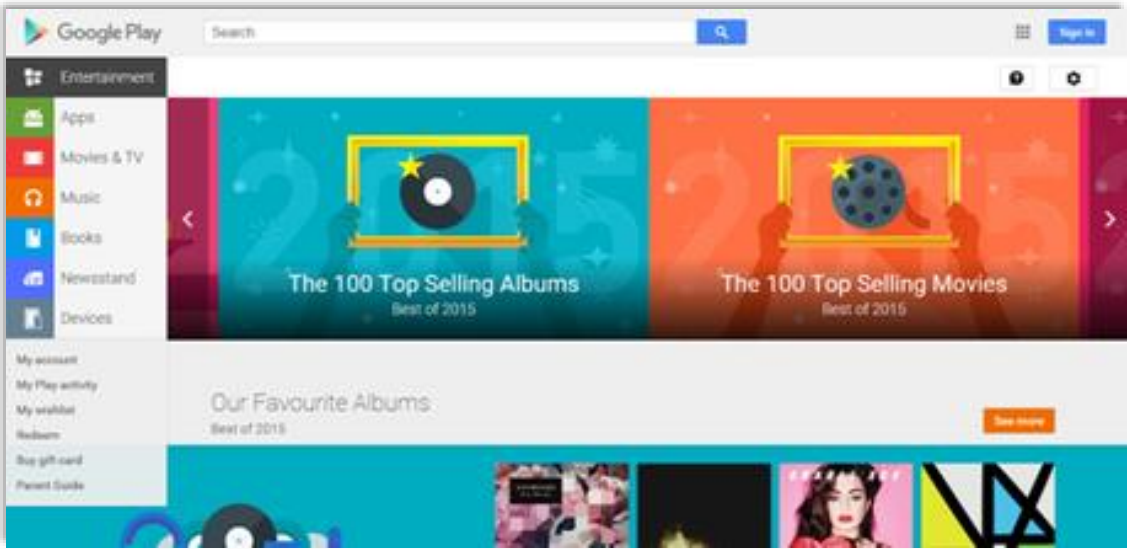
Εικόνα 6 ***(Android 6.0 home screen)***

Πηγή: wikipedia - Android (operating system)

2.3.2 Εφαρμογές

Το Android έχει ένα συνεχώς αυξανόμενο πληθυσμό εφαρμογών που έχουν δημιουργηθεί από τρίτους, οι οποίες είναι διαθέσιμες μέσω του καταστήματος εφαρμογών Google Play. Το Google Play δίνει στους χρήστες την δυνατότητα να ψάξουν, να κατεβάσουν και να αναβαθμίσουν εφαρμογές. Η εφαρμογή για το κατάστημα είναι προ εγκατεστημένη στις συσκευές που συνάδουν με τις απαιτήσεις συμβατότητας και την άδεια της Google για τις κινητές συσκευές. Οι διαθέσιμες εφαρμογές είναι είτε δωρεάν είτε υπό πληρωμή. Σε περίπτωση που κάποιος αγοράσει μια εφαρμογή κατά λάθος εγγυάται επιστροφή χρημάτων μέσα σε 15 λεπτά από την αγορά της εφαρμογής.

Οι διαθέσιμες εφαρμογές έχουν πολλές κατηγορίες όπως παιχνίδια, διαχειριστές αρχείων, εργαλεία για τη εποπτεία του συστήματος, λογισμικά καταπολέμησης ιών, εκπαιδευτικές εφαρμογές και πολλά άλλα. Πρόσφατα προστέθηκε και η δυνατότητα αγοράς ψηφιακών εκδόσεων βιβλίων, μουσικής και ταινιών.



Εικόνα 7 ***(Κύρια σελίδα του Google Play)***

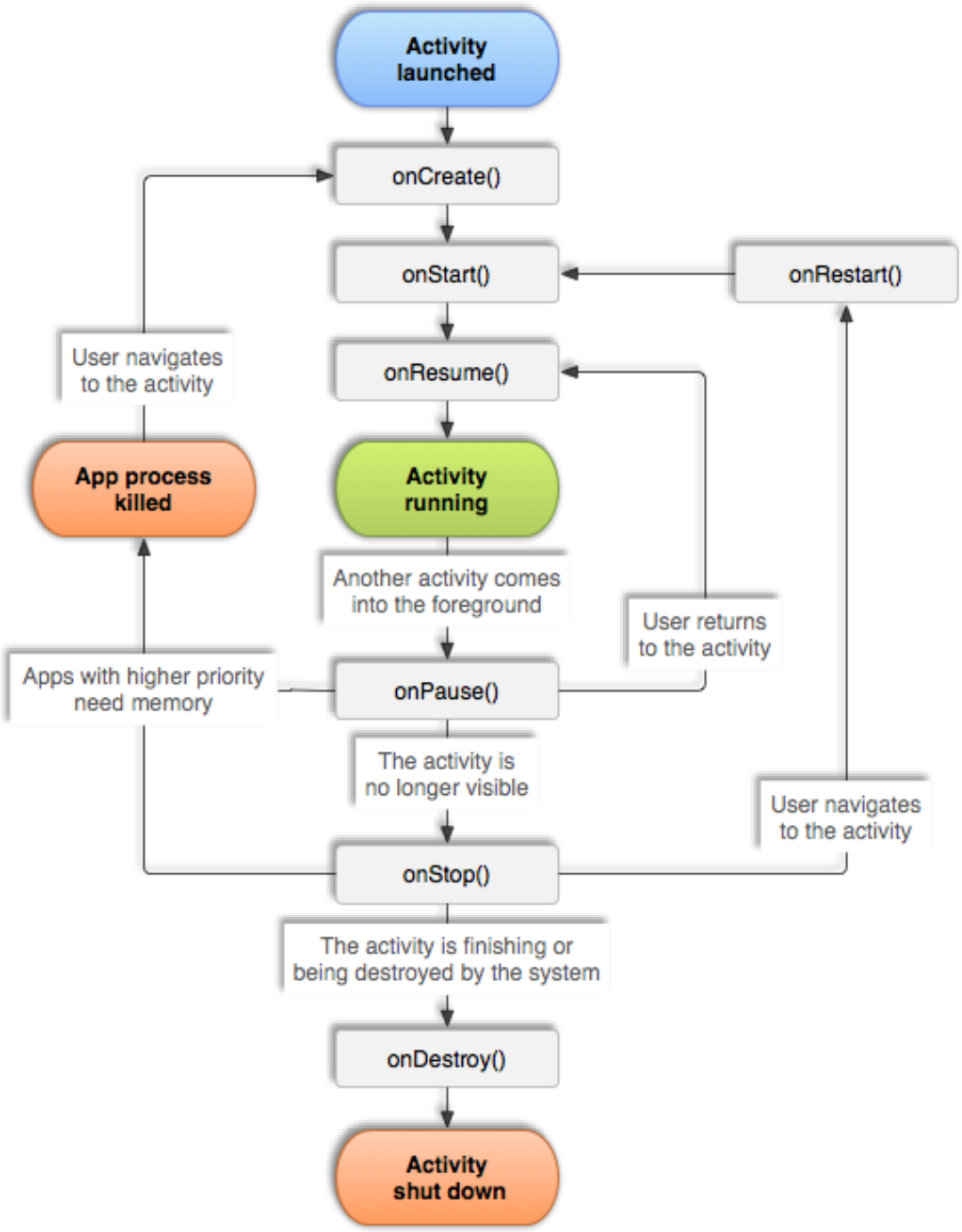
Πηγή: wikipedia - Google Play

2.3.3 Διαχείριση Μνήμης

Οι συσκευές που έχουν Android OS συνήθως τροφοδοτούνται από μπαταρία, συνεπώς το Android είναι σχεδιασμένο να διαχειρίζεται την μνήμη (RAM), έτσι ώστε να διατηρεί την κατανάλωση ενέργειας στο ελάχιστο, σε αντίθεση με τα λειτουργικά συστήματα των επιτραπέζιων υπολογιστών, οι οποίοι έχουν συνεχή τροφοδοσία. Όταν μια εφαρμογή δεν χρησιμοποιείται πλέον, το σύστημα αυτόματα θα την αναστείλει στην μνήμη. Παρόλο που οι εφαρμογή είναι «ακόμη ανοικτή», δεν καταναλώνει πόρους (π.χ. υπολογιστική ισχύ ή ενέργεια από την μπαταρία) και διατηρείται αδρανής στο παρασκήνιο έως ότου χρειαστεί ξανά. Η τεχνική αυτή έχει το διπλό πλεονέκτημα, της αύξησης της απόκρισης της συσκευής, από την στιγμή που οι εφαρμογές δεν χρειάζεται να κλειστούν και να ξανανοιχτούν από την αρχή, καθώς και της σιγουριάς ότι οι εφαρμογές παρασκηνίου δεν καταναλώνουν ενέργεια άσκοπα.

Η διαχείριση των αδρανοποιημένων εφαρμογών στην μνήμη γίνεται αυτόματα: όταν η διαθέσιμη μνήμη είναι λίγη, το σύστημα θα ξεκινήσει να τερματίζει εφαρμογές και διεργασίες, οι οποίες είναι ανενεργές για ώρα, με την ανάποδη χρονική σειρά κατά την οποία αυτές χρησιμοποιήθηκαν (η παλαιότερη τερματίζεται πρώτη). Η όλη διαδικασία είναι σχεδιασμένη, έτσι ώστε να είναι αόρατη στον χρήστη, συνεπώς οι χρήστες δεν χρειάζεται να διαχειρίζονται την μνήμη ή να τερματίζουν εφαρμογές.

Πηγή: wikipedia - Android (operating system)



Εικόνα 8 *(Activity lifecycle)*

Πηγή: Android Developers - Activity

2.3.4 ANDROID Software Development Kit (SDK)

Οι εφαρμογές μπορούν να θεωρηθούν ως επεκτάσεις την λειτουργικότητας της συσκευής και δημιουργούνται κατά κύριο λόγο με την γλώσσα προγραμματισμού Java με την χρήση του Android Software Development Kit (Android SDK). Το SDK περιλαμβάνει ένα σύνολο εργαλείων προγραμματισμού, όπως debugger, βιβλιοθήκες λογισμικού, έναν προσομοιωτή συσκευής βασισμένο σε QEMU, τεκμηρίωση, παραδείγματα κώδικα και οδηγούς εκμάθησης.

Εικόνα 9 ***(Το λογότυπο του Android SDK)***

Η Google έδινε το SDK αρχικά χωρίς κάποιο περιβάλλον ανάπτυξης κώδικα αλλά αργότερα πρότεινε και προτείνει το Eclipse IDE. Στις 16 Μάιου 2013, στα πλαίσια του συνεδρίου Google I/O, ανακοινώθηκε από την διευθυντή προϊόντων της Google, Ellie Powers, ένα νέο περιβάλλον ανάπτυξης κώδικα υπό το όνομα Android Studio. Από τον Ιούνιο του 2013, είναι διαθέσιμη στους χρήστες μια δωρεάν αρχική έκδοση από την επίσημη ιστοσελίδα της Google. Το Android Studio βασίζεται λογισμικό IntelliJ IDEA της JetBrain's και είναι σχεδιασμένο ειδικά για την κατασκευή εφαρμογών Android. Είναι διαθέσιμο για Windows, Mac OS X and Linux.

Πηγή: wikipedia - Android (operating system)

2.3.5 Αρχιτεκτονική του ANDROID OS

Υλικό και πρόσθετα εξαρτήματα

Η κύρια υλική πλατφόρμα (hardware platform) για το Android OS είναι η αρχιτεκτονική 32-bit ARMv7. Το πρόγραμμα Android-x86 παρέχει υποστήριξη για την x86 αρχιτεκτονική. Το 2012, Intel επεξεργαστές έκαναν την εμφάνισή τους σε Android συσκευές, όπως κινητά τηλέφωνα. Κατόπιν, το 2013, η Freescale ανακοίνωσε την υποστήριξη στον i.MX επεξεργαστή της, και ειδικότερα στις σειρές i.MX5X και i.MX6X. Επιπρόσθετα, το Android μπορεί να λειτουργήσει σε x86 αρχιτεκτονική με την χρήση του προσομοιωτή Android από το Android SDK ή με την χρήση του Bluestacks.

Οι συσκευές Android διαθέτουν πολλά προαιρετικά εξαρτήματα υλικού, όπως κάμερες, GPS, αισθητήρες προσανατολισμού, χειριστήρια για παιχνίδια, επιταχυνσιόμετρα, γυροσκόπια, βαρόμετρα, μαγνητόμετρα, αισθητήρες εγγύτητας, αισθητήρες πίεσης, θερμόμετρα και οθόνες αφής. Μερικά εξαρτήματα δεν απαιτούνται, αλλά έγιναν τυπικά σε μερικές κατηγορίες συσκευών, όπως κινητά τηλέφωνα και tablets. Επίσης, μερικά εξαρτήματα δεν απαιτούνται πλέον, όπως για παράδειγμα τα μικρόφωνα, τα οποία με είσοδο των tablets στην αγορά έγιναν προαιρετικά.

Πυρήνας Linux

Το Android βασίζεται στον πυρήνα Linux LTS. Από τον Ιανουάριο του 2014, οι τρέχουσες εκδόσεις του Android χτίζονται στον πυρήνα Linux 3.4 ή μεταγενέστερο, αλλά η ακριβής έκδοση του πυρήνα εξαρτάται από την συσκευή και το chipset αυτής.

Ο πυρήνας Linux έχει περαιτέρω αρχιτεκτονικές αλλαγές, οι οποίες υλοποιήθηκαν από την Google, όπως η ένταξη συστατικών όπως ο Binder, η ashmem, η pmem, δίαφορα wakelocks και διαφορετική διαχείριση στην περίπτωση out-of-memory (OOM).

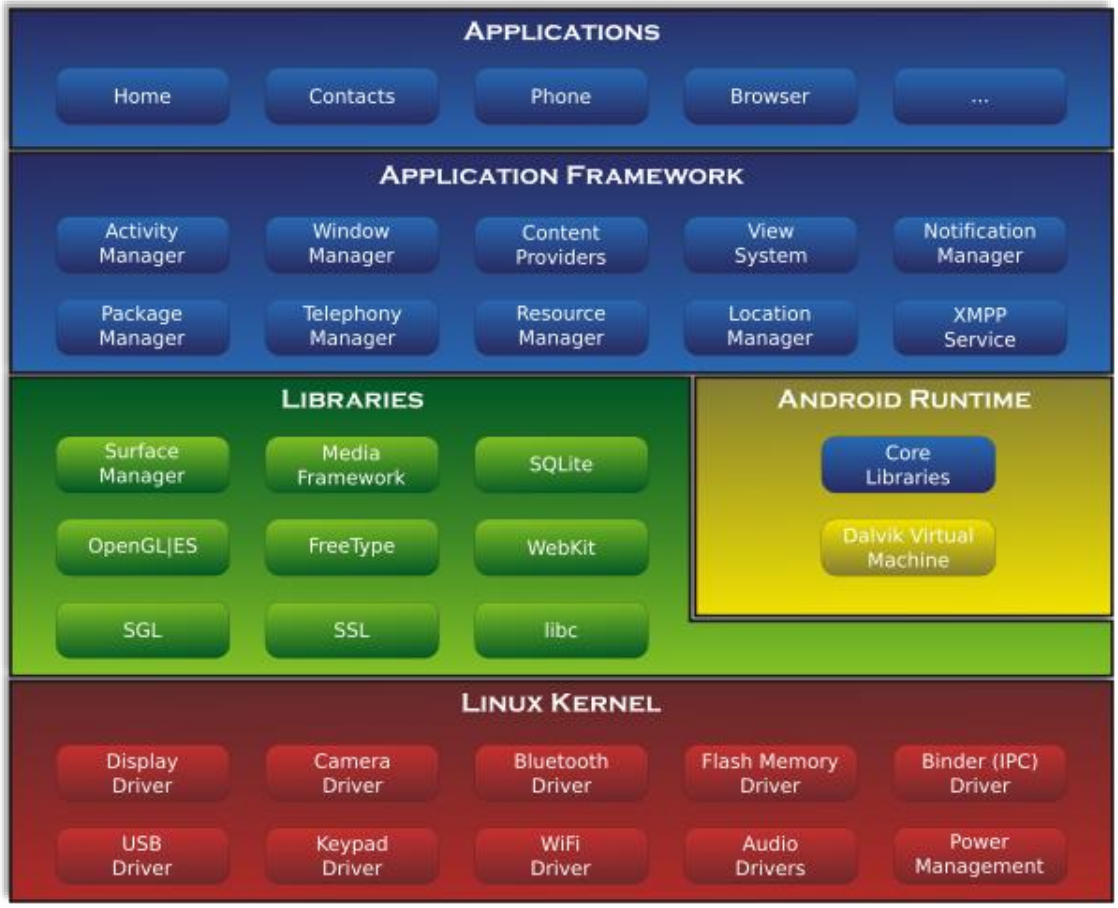
Η αποθήκευση τύπου flash στις Android συσκευές χωρίζεται σε διάφορα μέρη, όπως το /system για το λειτουργικό σύστημα και /data για τα δεδομένα χρήστη και εγκαταστάσεις εφαρμογών. Σε αντίθεση με τις διανομές Linux για επιτραπέζιους υπολογιστές, οι χρήστες Android δεν έχουν δικαιώματα διαχειριστή (root access) στο λειτουργικό σύστημα και «ευαίσθητα» τμήματα όπως το /system είναι μόνο για ανάγνωση. Όμως, τα δικαιώματα διαχειριστή μπορούν να αποκτηθούν μέσω κενών ασφαλείας, τα οποία χρησιμοποιούνται διαρκώς από την κοινότητα ανοιχτού κώδικα για να ενισχύσουν τις δυνατότητες των συσκευών τους.

Σύμφωνα με την Linux Foundation, το Android είναι μια διανομή Linux. Ακόμη, υπάρχει η γνώμη πως το Android δεν είναι μια Linux διανομή με την έννοια της παραδοσιακής Unix διανομής, καθώς το Android δεν περιλαμβάνει την GNU C Library και κάποια άλλα συστατικά που βρίσκονται σε διανομές Linux.

Λογισμικό

Στην κορυφή του πυρήνα Linux, υπάρχουν ενδιάμεσο λογισμικό, βιβλιοθήκες και APIs σε γλώσσα προγραμματισμού C και λογισμικό εφαρμογής, το οποίο τρέχει σε ένα

πλαίσιο εφαρμογής, το οποίο περιλαμβάνει βιβλιοθήκες συμβατές με Java βασισμένες στο Apache Harmony. Το Android χρησιμοποίει την εικονική μηχανή Dalvik με just-in-time compilation έτσι ώστε να τρέχει «κώδικα Dalvik» (Dalvik executable), ο οποίος συνήθως μεταφράζεται από τον bytecode της Java. Από την έκδοση 4.4 υποστηρίζεται και μια νέα πειραματική εικονική μηχανή, η ART, η οποία είναι απενεργοποιημένη από προεπιλογή.
Η βιβλιοθήκη της γλώσσας C στο Android, η Bionic, αναπτύχθηκε από την Google ειδικά για το Android, ως μια διακλάδωση της βιβλιοθήκης του BSD. Η Bionic έχει αρκετά χαρακτηριστικά ειδικά για τον πυρήνα των Linux και η ανάπτυξή του συνεχίζεται ανεξάρτητα από τις άλλες βάσεις κώδικα του Android. Τα προνόμια της χρήσης της Bionic αντί της GNU C Library (glibc) ή της uClibc είναι η διαφορετική αδειοδότηση, μικρότερες απαιτήσεις σε υπολογιστική ισχύ και βελτιστοποίηση σε επεξεργαστές χαμηλών συχνοτήτων. Στοχεύοντας σε ένα πιο κατάλληλο μοντέλο αδειοδότησης, στα τέλη του 2012, η Google άλλαξε την Bluetooth Stack στο Android από την BlueZ στην BlueDroid.



Εικόνα 10 ***(Η αρχιτεκτονική του Android OS)***

Πηγή: wikipedia - Android (operating system)

2.3.6 Ιστορικό Εκδόσεων του ANDROID OS

Το Νοέμβριο του 2007, η Google ανακοίνωσε ότι η εταιρεία σχεδιάζει ένα κινητό τηλέφωνο (Google Phone), καθώς και ένα ολοκαίνουργιο λειτουργικό σύστημα για κινητά τηλέφωνα με το όνομα Android. Το λειτουργικό σύστημα από τότε έχει εγκατασταθεί σε εκατομμύρια συσκευές και έχει αναβαθμιστεί σε εκδόσεις με επιπλέον χαρακτηριστικά και μεγαλύτερη αξιοπιστία.

Android 1.0 (API level 1)

Η πρώτη εμπορική έκδοση του λειτουργικού συστήματος κυκλοφόρησε στις 23 Οκτωβρίου 2008. Η HTC ήταν ο πρώτος κατασκευαστής κινητών τηλεφώνων που διάθεσε στην αγορά κινητό με λειτουργικό Android, το T-Mobile G1 στις ΗΠΑ (ή HTC Dream εκτός των ΗΠΑ), τον Οκτώβριο του 2008.

Εικόνα 11 *(Περιβάλλον χρήσης Android 1.0)*

Τα βασικά χαρακτηριστικά της έκδοσης ήταν:

- Συρόμενο παράθυρο ειδοποιήσεων
- Γραφικά στοιχεία στην αρχική οθόνη (widgets)
- Μια πλήρης διαχείριση της υπηρεσίας Gmail
- Android Market (ηλεκτρονικό κατάστημα εφαρμογών)

Android 1.1 (API level 2)

Η πρώτη αναβάθμιση του Android OS έγινε τον Φεβρουάριο του 2009, τρείς μήνες μετά την διάθεση του HTC Dream. Η αναβάθμιση δεν έφερε μεγάλες αλλαγές αλλά διόρθωσε μια μεγάλη λίστα από σφάλματα και έδωσε στο Android την δυνατότητα να μοιράζει αναβαθμίσεις «over the air» και να απλοποιήσει την διαδικασία για τους χρήστες.

Εικόνα 12 ***(Λογότυπο Android 1.1)***

Android 1.5 (API level 3) "Cupcake"

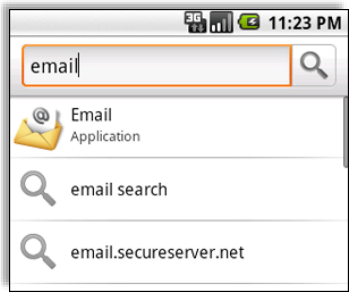
Η πρώτη αναβάθμιση με σημαντικά αποτελέσματα. Οι προηγούμενες δυο εκδόσεις αναφέρονταν σε ένα κινητό με φυσικό πληκτρολόγιο. Έτσι στην αναβάθμιση αυτή παρουσιάστηκε το εικονικό πληκτρολόγιο. Το Ιούλιο του 2009 παρουσιάστηκε το πρώτο κινητό με οθόνη αφής και χωρίς φυσικό πληκτρολόγιο, το HTC Magic. Άλλες βασικές επεκτάσεις που εισήχθησαν ήταν:

- Επεκτάσιμα γραφικά στοιχεία (τρίτοι μπορούσαν πλέον να δημιουργήσουν δικά τους widget)
- Επιλογή και αντιγραφή κειμένου από τον φυλλομετρητή (browser)
- Καταγραφή και αναπαραγωγή βίντεο και εστίαση με την αφή

Εικόνα 13 ***(Λογότυπο Android 1.5)***

Android 1.6 (API level 4) "Donut"

Η αναβάθμιση αυτή έφερε κάποιες μικρές αλλαγές στην εμφάνιση του λειτουργικού, αλλά το σημαντικό σημείο είναι η υποστήριξη CDMA δικτύων. Έτσι οι Android συσκευές μπορούσαν πλέον να λειτουργήσουν σε ασιατικές χώρες. Ακόμη μια σημαντική αλλαγή ήταν η δυνατότητα υποστήριξης διάφορων μεγεθών και αναλύσεων οθόνης. Έτσι έγινε δυνατή η δημιουργία συσκευών με ανάλυση καλύτερη της αρχικής 320x480. Ακόμη, πλέον με το πάτημα του κουμπιού της αναζήτησης ο χρήστης μπορούσε να κάνει και εσωτερική αναζήτηση στο κινητό για επαφές εφαρμογές, κ.α. εκτός από την πάγια λειτουργία αναζήτησης στο διαδίκτυο. Τέλος, αναβαθμίστηκε το περιβάλλον χρήσης του ηλεκτρονικού καταστήματος αγορών «Android Market».



Εικόνα 14 *(Η εσωτερική αναζήτηση)*

Android 2.0 -2.1 (API level 5-7) "Eclair"

Το σημείο όπου το Android έκανε ένα μεγάλο βήμα για την μετέπειτα κυριαρχία του. Η αναβαθμίσεις 2.0 έως 2.1 άλλαξαν ριζικά την αρχιτεκτονική του συστήματος καθώς και το περιβάλλον χρήσης. Οι αλλαγές αυτές ήταν:

- ***Υποστήριξη πολλαπλών λογαριασμών***

  Ο χρήστης μπορούσε να εισάγει τα στοιχεία πολλαπλών λογαριασμών ηλεκτρονικού ταχυδρομείου. Ακόμη, δόθηκε η δυνατότητα στους δημιουργούς των εφαρμογών, να χρησιμοποιήσουν αυτό το πλαίσιο λογαριασμών, έτσι ώστε ο χρήστης να μπορεί να συγχρονίσει και άλλες εφαρμογές εκτός του Gmail.
- ***Πλοήγηση μέσω Google Maps***

  Εισάχθηκε η δυνατότητα πλοήγησης μέσω GPS του κινητού και την χρήση των χαρτών της Google.
- ***Ομιλία σε κείμενο***

  Αντικαταστάθηκε το πλήκτρο με το κόμμα «,» στο εικονικό πληκτρολόγιο και αντικαταστάθηκε από το εικονίδιο ενός μικροφώνου. Με το πάτημα του, ο χρήστης μιλούσε και τα λεγόμενα του μετατρέπονταν σε κείμενο.

Εικόνα 15 *(Λογότυπο Android 2.0-2.1)*

Android 2.2 -2.2.3 (API level 8) "Froyo"

Η έκδοση αυτή εισήγαγε ένα ανανεωμένο γραφικό περιβάλλον, καθώς και μια αναβαθμισμένη έκδοση της εφαρμογής προβολής φωτογραφιών. Ακόμη, δόθηκε η δυνατότητα στους χρήστες να διαμοιράζονται την ασύρματη σύνδεση που παρείχε η εταιρία κινητής τηλεφωνίας με την δημιουργία ενός Wifi-hotspot.

Εικόνα 16 *(Λογότυπο Android 2.2-2.2.3)*

Android 2.3 -2.3.7 (API level 9-10) "Gingerbread"

Με την έκδοση αυτή έγινε μια προσπάθεια μείωσης της κατανάλωσης της ενέργειας της μπαταρίας. Αρχικά, προστέθηκε εφαρμογή επίβλεψης κατανάλωσης της μπαταρίας από τις εφαρμογές. Έτσι ο χρήστης μπορούσε να σταματήσει κάποια εφαρμογή από την άσκοπη κατανάλωση μπαταρίας. Ακόμη, για πρώτη φορά παρουσιάστηκε η δυνατότητα για υποστήριξη κάμερας και στην μπροστινή όψη της συσκευής. Τέλος, δόθηκε πρόσβαση στους προγραμματιστές εφαρμογών σε συναρτήσεις χαμηλότερου επίπεδο όσον αφορά τα γραφικά και τον ήχο, έτσι ώστε να δημιουργηθούν πιο ελκυστικά παιχνίδια.

Εικόνα 17 *(Λογότυπο Android 2.3-2.3.7)*

Android 3.0 -3.2 (API level 11-13) "Honeycomb"

Η πρώτη έκδοση Android που προορίζονταν αποκλειστικά για tablets. Το χρώμα που κυριαρχούσε στο γραφικό περιβάλλον ήταν το πράσινο και παρουσιάστηκαν τα πλήκτρα αφής που αντικατέστησαν τα φυσικά πλήκτρα της αναζήτησης, της αρχικής οθόνης και του μενού.

Εικόνα 18 *(Λογότυπο Android 3.0-3.2)*

Android 4.0-4.0.4 (API level 14-15) "Ice Cream Sandwich"

Η ICS έκδοση του λογισμικού έφερε τις αλλαγές που εισήχθησαν με το Honeycomb στα κινητά (το Honeycomb αποτέλεσε εν τέλει έκδοση μόνο για tablets). Οι κύριες αλλαγές ήταν:

- Αλλαγή του γραφικού περιβάλλοντος
- Βελτίωση πληκτρολογίου
- Ομαδοποίηση εφαρμογών σε φακέλους
- Εισαγωγή εφαρμογών στην μπάρα των αγαπημένων στην αρχική οθόνη
- Android Beam (NFC επικοινωνία μεταξύ δύο συσκευών)
- Ξεκλείδωμα οθόνης με την αναγνώριση του προσώπου του ιδιοκτήτη
- Μετονομασία Android Market σε Google Play
- Εφαρμογή για την παρακολούθηση χρήσης δεδομένων από εφαρμογές.

Εικόνα 19 *(Λογότυπο Android 4.0-4.0.4)*

Android 4.1-4.3 (API level 16-18) "Jelly Bean"

Η Google με την αναβάθμιση αυτή ήθελε να βελτιώσει την λειτουργικότητα και την απόδοση του περιβάλλοντος χρήσης. Η προσπάθεια αυτή περιλάμβανε το "Project Butter", το οποίο χρησιμοποιεί αναμονή αφής, τριπλό buffering, Vsync και ένα σταθερό ρυθμό πλαισίων στα 60 Fps με απώτερο σκοπό την δημιουργία ενός ομαλού περιβάλλοντος. Τα παρακάτω χαρακτηριστικά είναι τα κυριότερα αυτής της αναβάθμισης:

- Πρόβλεψη λέξεων κατά την πληκτρολόγηση
- Δημιουργία χρηστών σε ένα tablet
- Υποστήριξη OpenGL ES3.0
- Bluetooth Smart για συσκευές χαμηλής κατανάλωσης

Εικόνα 20 *(Λογότυπο Android 4.1-4.3)*

Android 4.4 (API level 19) "KitKat"

Παρουσιάστηκε τον Οκτώβριο του 2013. Και πάλι έγινε αλλαγή στο κυρίαρχο χρώμα, από το μπλέ του Honeycomb επιλέχθηκε το άσπρο χρώμα. Τα κύρια σημεία της αναβάθμισης ήταν:

- Μονιμοποίηση της μπάρας "Google Now" στην αρχική οθόνη
- Στροφή σε χρήση του Google Drive αντί εξωτερικής κάρτας επέκτασης μνήμης
- Δυνατότητα εκτύπωσης εγγράφων μέσω εκτυπωτών που έχουν συνδεθεί στο Google Cloud Print
- Νεα τεχνική HDR+ για λήψη καλύτερων φωτογραφιών.

Εικόνα 21 *(Λογότυπο Android 4.4)*

Android 5.0-5.1 (API level 21-22) "Lollipop"

Η τελευταία μέχρι στιγμής έκδοση του Android OS. Παρουσιάστηκε στις 25 Ιουνίου του 2014 κατά τη διάρκεια της ετήσιας ημερίδας της Google I/O, με τη κωδική ονομασία «Android L». Στις 12 Νοεμβρίου του 2014 έγινε διάθεσιμη στο ευρύ κοινό και σε επιλεγμένες συσκευές, οι οποίες «τρέχουν» διανομές του λειτουργικού απευθείας από τη Google (οι συσκευές αυτές είναι οι συσκευές των σειρών Nexus και Google edition). Ο πηγαίος κώδικας (source code) διατέθηκε στις 3 Νοεμβρίου του 2014.

Η έκδοση του λειτουργικού συστήματος, Lolipop, χαρακτηρίζεται από ένα επανασχεδιασμένο γραφικό περιβάλλον χρήσης, το οποίο έχει κατασκευαστεί με τη χρήση μιάς γλώσσας σχεδιασμού, η οποία αναφέρεται ως «material design». Άλλες, αλλαγές συμπεριλαμβάνουν βελτιώσεις στο σύστημα ειδοποιήσεων, το οποίο πλέον μπορεί να είναι προσβάσιμο από την οθόνη κλειδώματος και να εμφανίζεται εν μέσω άλλων εφαρμογών ως καρτέλες ειδοποίησης στο κέντρο της οθόνης. Ακόμη, η Google έκανε εσωτερικές αλλαγές στο λειτουργικό σύστημα, με την επίσημη πλέον αντικατάσταση του περιβάλλοντος εκτέλεσης Dalvik από το Android Runtime (ART). Το ART βελτιώνει την απόδοση των εφαρμογών και διαθέτει αλλαγές προσανατολισμένες στη βελτίωση της χρήσης και της άυξησης του χρόνου ζωής της μπαταρίας. Οι αλλαγές αυτές που στοχεύουν στη μπαταρία είχε ονομαστεί εσωτερικά ως «project Volta».

Εικόνα 22 ***(Android 5.0 home screen)***

Πηγή: wikipedia - Android version history

Android 6.0 "Marshmallow"

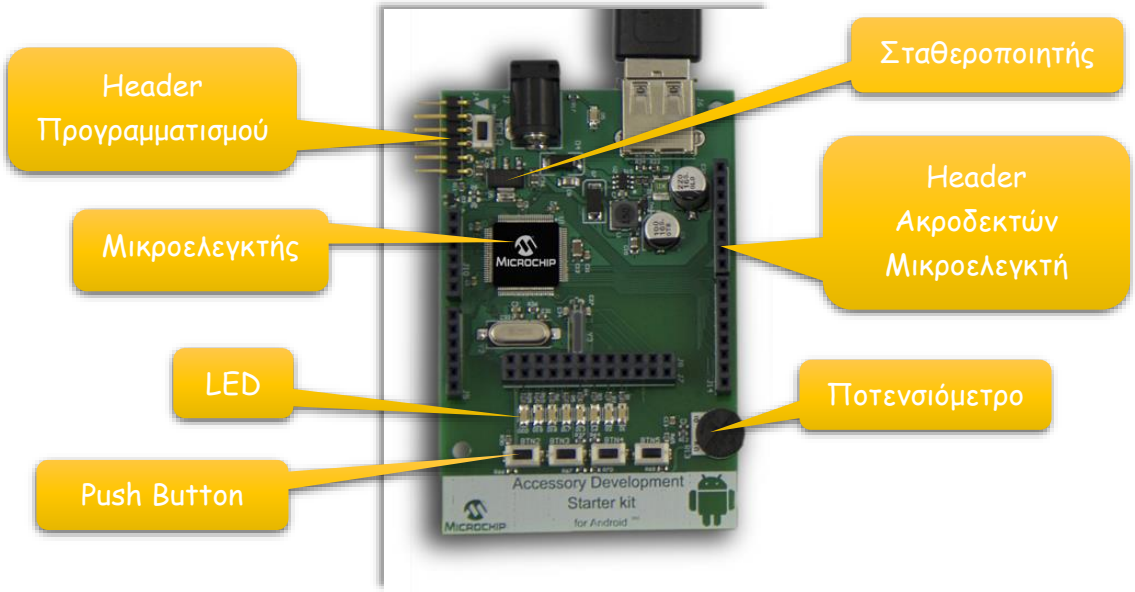
Η έκδοση του λειτουργικού συστήματος Android, με ονομασία Andoid 6.0 "Marshmallow" παρουσιάστηκε με την κωδική ονομασία "Android M" στο συνέδριο Google I/O στις 28 Μαιου 2015. Η αρχική αυτή έκδοση προοριζόταν για τις συσκευες Nexus 5, Nexus 6, Nexus 9 και Nexus Player. Η επίσημη ημερομηνία διάθεσης ήταν ο Οκτώβριος του 2015. Η εν λόγω έκδοση, είχε ως πρωταρχικό στόχο την βελτίωση της εμπειρίας χρήσης της προηγούμενης έκδοσης ("Lollipop"), εισάγοντας μια νέα αρχιτεκτονική δικαιωμάτων, ένα νέο σύστημα διαχείρισης ενέργειας, το οποίο μειώνει τις διεργασίες παρασκηνίου όταν η συσκευή βρίσκεται σε αδράνεια. Ακόμη, προστέθηκε η υποστήριξη συνδέσμων για αναγνώριση δακτυλικών αποτυπωμάτων και συνδέσεων USB Type-C και η δυνατότητα μεταφοράς των δεδομένων και των εφαρμογών στην κάρτα microSD και χρήση αυτής σαν κύρια μονάδα αποθήκευσης.

Εικόνα 23 ***(Android 6.0 home screen)***

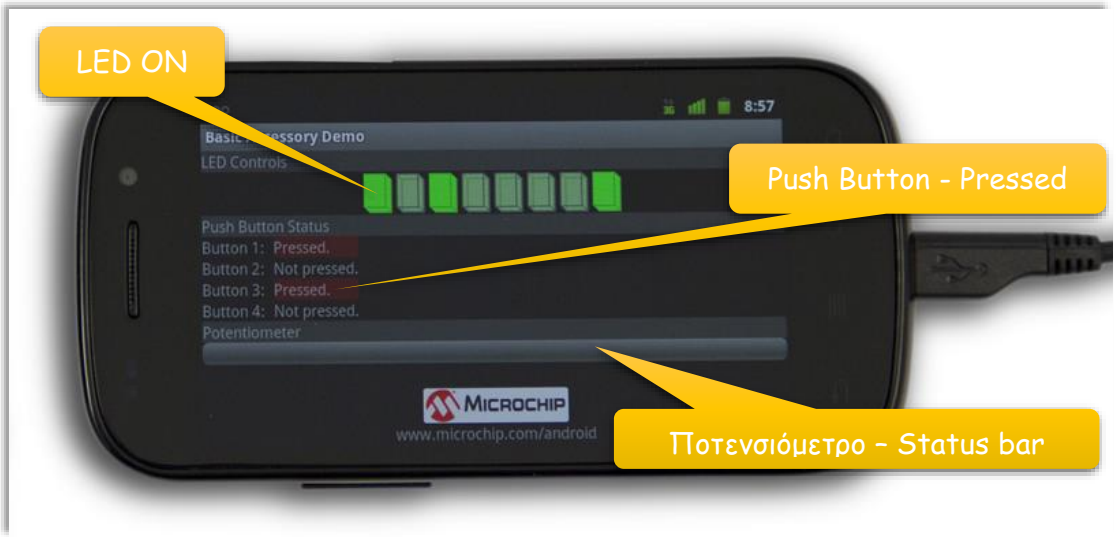
Πηγή: wikipedia - Android Marshmallow

3 Σχεδίαση εφαρμογής

Η υλοποίηση του ***συστήματος ελέγχου των λειτουργειών ενός οχήματος*** στηρίχτηκε και εξελίχτηκε βάση ενός αναπτυξιακού kit της εταιρίας Microchip, η οποία κατασκευάζει μικροελεγκτές και άλλα ολοκληρωμένα κυκλώματα. Η αναπτυξιακή πλακέτα που χρησιμοποιήθηκε είναι αυτή με την ονομασία «PIC24F Accessory Development Starter Kit for Android», η οποία φαίνεται στην *εικόνα 24*. Η συγκεκριμένη πλακέτα αποτελείτε από το μικροελεγκτή «PIC24FJ256GB110-I/PT», ο οποίος είναι ένας 16bit μικροελεγκτής με 100 ποδαράκια (pins). Ο μικροελεγκτής τροφοδοτείται με 3,3Volt μέσω ενός γραμμικού step-down σταθεροποιητή (regulator). Η πλακέτα επίσης περιλαμβάνει ένα ακόμη σταθεροποιητή στα 5Volt. Κάποια από τι pin του μικροελεγκτή συνδέονται σε header 2.54mm με σκοπό την επέκταση του συστήματος, όπως τη λήψη εξωτερικών σημάτων από άλλα κυκλώματα ή το χειρισμό άλλων συσκευών π.χ. μέσω ρελέ. Εκτός, όμως, από το υλικό κομμάτι η εταιρία Microchip μαζί με την αναπτυξιακή πλακέτα παρέχει και το κώδικα μία εφαρμογής Android την οποία έχει αναπτύξει η ίδια η εταιρία. Η δυνατότητες της εφαρμογής (εικόνα 25) αυτής έτσι όπως παρέχεται από την εταιρία είναι να στέλνει δεδομένα-εντολές στον μικροελεγκτή και να λαμβάνει δεδομένα από το μικροελεγκτή, με σκοπό την ενεργοποίηση-απενεργοποίηση οχτώ smd LED, την απεικόνιση της κατάστασης τεσσάρων push button, καθώς και τη θέση του δρομέα ενός ποτενσιόμετρου (potentiometer). **Βάση του προϋπάρχοντάς αναπτυξιακού kit της εταιρίας Microchip, τόσο σε επίπεδο υλικού όσο και λογισμικού, έγιναν οι κατάλληλες τροποποιήσεις και αναβαθμίσεις, ώστε να υλοποιηθεί το «*σύστημα ελέγχου των λειτουργειών ενός οχήματος*» με το οποίο και ασχολείται η εργασία και παρουσιάζεται αναλυτικότερα εν συνέχεια.**



Εικόνα 24 *(Hardware - PIC24F Accessory Development Starter Kit for Android)*



Εικόνα 25 ***(Software - PIC24F Accessory Development Starter Kit for Android)***

3.1 Λειτουργικότητα συστήματος (hardware)

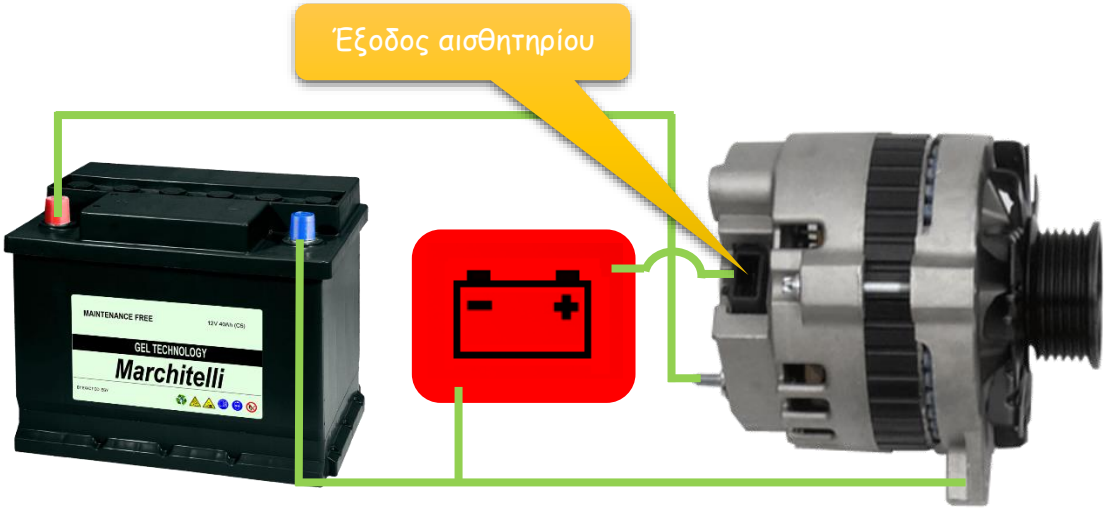
Όσο αφορά το υλικό (hardware) τους συστήματος, σκοπός του είναι να επικοινωνεί με κάποια από τα αισθητήρια-διακόπτες του αυτοκινήτου και να ενημερώνει τον οδηγό μέσω του tablet (είτε με εικονικές ειδοποιήσεις είτε με φωνητικές) για την κατάσταση τους, καθώς επίσης και να ενεργοποιεί-απενεργοποιεί λειτουργίες του αυτοκινήτου. Πιο συγκεκριμένα τα αισθητήρια-διακόπτες των οποίων ο μικροελεγκτής ελέγχει την κατάσταση είναι:

- της μπαταρίας,
- του λαδιού
- της θερμοκρασίας
- του χειρόφρενου
- της εκάστοτε ζώνης ασφαλείας
- της μεγάλης σκάλας,
- του δεξιού φλάς
- του αριστερού φλάς
- του στάθμης του καυσίμου

3.1.1 Είσοδοι συστήματος

Αισθητήρας μπαταρίας

Ο αισθητήρας της μπαταρίας είναι ενσωματωμένος μέσα στη γεννήτρια -δυναμό (alternator-dynamo) και εξετάζει την τιμή της τάσης που παράγει το δυναμό. Αυτό το αισθητήριο λειτουργεί ως εξής. Όταν το δυναμό παράγει την απαιτούμενη τάση τότε η έξοδος του αισθητηρίου είναι 12Volt, ενώ όταν η τάση δεν είναι ικανοποιητική η έξοδος του αισθητηρίου είναι 0Volt, με αποτέλεσμα να ανάβει ένα λαμπάκι με το σύμβολο μίας μπαταρίας στο καντράν του αυτοκινήτου, το οποίο δηλώνει ότι υπάρχει πρόβλημα στην τάση. Το αισθητήριο αυτό είναι απαραίτητο, επειδή αν η τάση της μπαταρίας δεν είναι ικανοποιητική η μίζα δεν θα μπορεί να τροφοδοτηθεί ικανοποιητικά και συνεπώς δεν θα είναι δυνατή η εκκίνηση του κινητήρα.

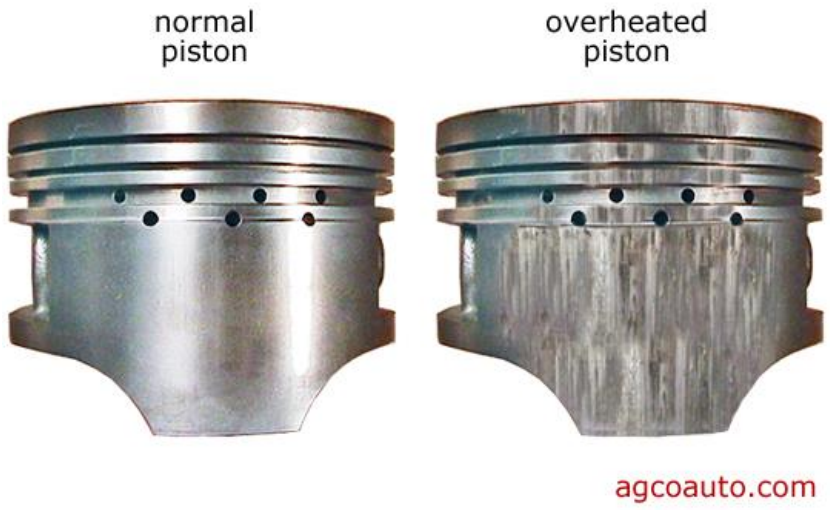


Εικόνα 26 ***(Μπλοκ διάγραμμα ενδείκτη μπαταρίας)***

Αισθητήρας λαδιού

Κατά την διάρκεια λειτουργίας του κινητήρα του αυτοκινήτου θα πρέπει να ελέγχεται εάν τα εξαρτήματα στο εσωτερικό του κινητήρα λιπαίνονται κανονικά. Σε περίπτωση που ένας κινητήρας δεν λιπαίνεται τότε η θερμοκρασία στο εσωτερικό της μηχανής αυξάνεται δραστικά λόγω των τριβών, με αποτέλεσμα να καταστρέφονται τα πιστόνια και οι κύλινδροι του κινητήρα (λόγω της υψηλής θερμοκρασίας που αναπτύσσεται τα μέταλλα σε κάποιες περιπτώσεις λιώνουν). Ο αισθητήρας που ελέγχει τα λάδια είναι ουσιαστικά ένας απλός διακόπτης. Όταν ο κινητήρας ξεκινήσει και η αντλία λαδιού της

μηχανής του αυτοκινήτου πιάσει τη κατάλληλη πίεση τότε ο διακόπτης ανοίγει και το λαμπάκι του λαδιού σβήνει, αντιθέτως σε περίπτωση που η πίεση δεν είναι η κατάλληλη τότε ο διακόπτης δεν ανοίγει, με αποτέλεσμα το λαμπάκι λαδιού να γίνεται φωτεινό. Αυτός είναι και ο λόγος που πάντα όταν το αυτοκίνητο είναι σβηστό και ο οδηγός γυρίσει το διακόπτη τα λαμπάκια της μπαταρίας και του λαδιού ανάβουν αμέσως (σε περίπτωση που δεν ανάψουν υπάρχει κάποιο τεχνικό πρόβλημα) και σβήνουν μόνο όταν ξεκινήσει ο κινητήρας και αρχίσει να δουλεύει το δυναμό και η αντλία λαδιού του κινητήρα.



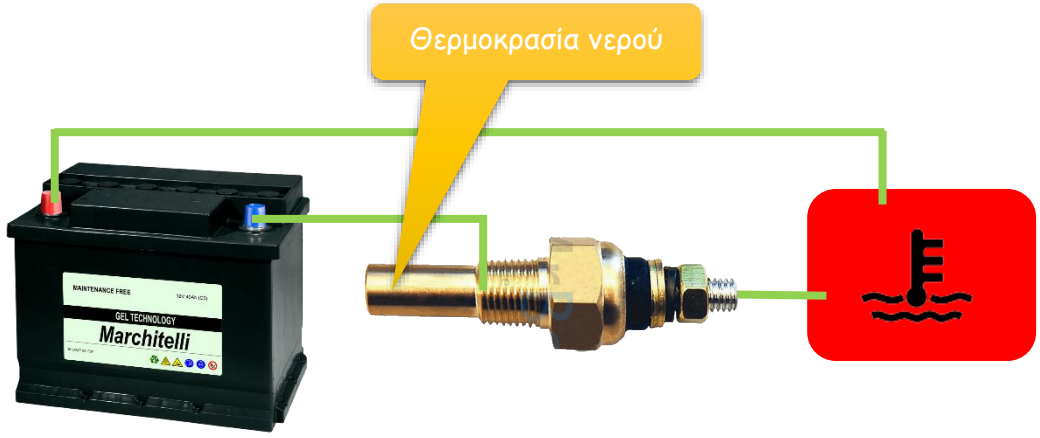
Εικόνα 27 *(Πιστόνι αυτοκινήτου)*

Εικόνα 28 *(Μπλοκ διάγραμμα ενδείκτη λαδιού)*

Αισθητήρας θερμοκρασίας

Ένας αισθητήρας παρόμοιος με αυτό του λαδιού είναι αυτός της θερμοκρασίας. Σε κάποια αυτοκίνητα ο αισθητήρας αυτός έχει αναλογική έξοδο (ειδικά στα μοντέλα των αυτοκινήτων πριν το 2000) με σκοπό την απεικόνιση της θερμοκρασίας της κινητήρα.

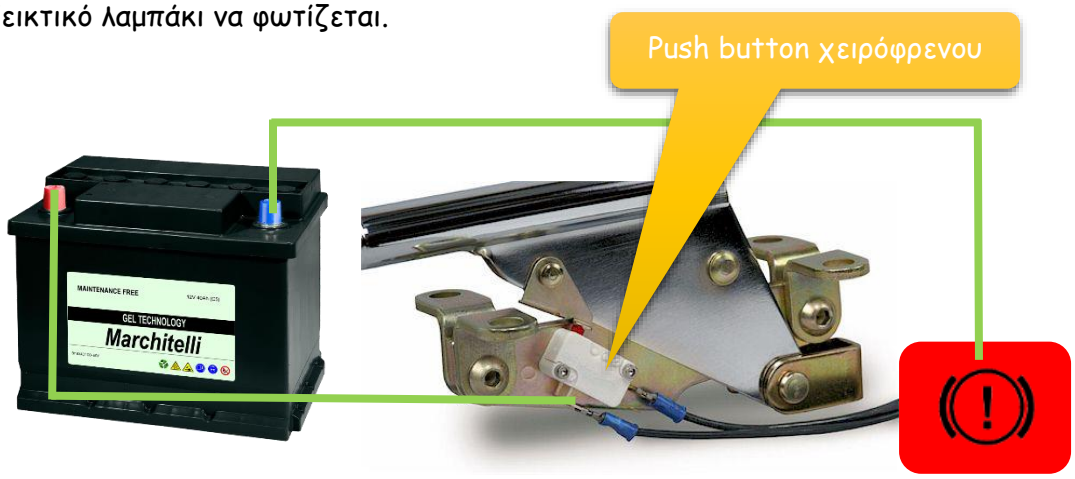
Παρόλα αυτά τα τελευταία χρόνια η απεικόνιση της θερμοκρασίας σε αρκετά αυτοκίνητά δεν υπάρχει και αντί αυτού υπάρχει ένα ενδεικτικό λαμπάκι, που προειδοποίει τον οδηγό όταν η θερμοκρασία του νερού ξεπεράσει ένα κρίσιμο σημείο, το οποίο και ορίζεται από το κατασκευαστή του εκάστοτε αυτοκινήτου. Ο έλεγχος της θερμοκρασίας είναι και αυτός ένας πολύ κρίσιμος έλεγχος, διότι σε περίπτωση, όπου η θερμοκρασία ανέβει υπερβολικά τότε υπάρχει κίνδυνος άμεση καταστροφής **ολόκληρου** του κινητήρα. Το όριο της θερμοκρασίας πάνω από το οποίο καταστρέφετε ο κινητήρας συνήθως για τα νέας τεχνολογίας αυτοκίνητα είναι στους 110°C.



Εικόνα 29 ***(Μπλοκ διάγραμμα ενδείκτη θερμοκρασίας)***

Αισθητήρας χειρόφρενου

Ο αισθητήρας, ο οποίος δείχνει σε τι κατάσταση βρίσκεται το χειρόφρενο, δηλαδή εάν είναι «τραβηγμένο» ή όχι είναι καθαρά ένας απλός διακόπτης και πιο συγκεκριμένα ένα push button, όπου όταν το χειρόφρενο είναι κατεβασμένο τότε το push button πιέζεται με αποτέλεσμα η επαφή του να είναι ανοιχτή και έτσι το ενδεικτικό λαμπάκι που αντιστοιχεί στο χειρόφρενο να είναι σβηστό. Αντιθέτως, όταν ο οδηγός τραβήξει-σηκώσει το χειρόφρενο τότε η επαφή εσωτερικά του push button κλείνει, με αποτέλεσμα το ενδεικτικό λαμπάκι να φωτίζεται.

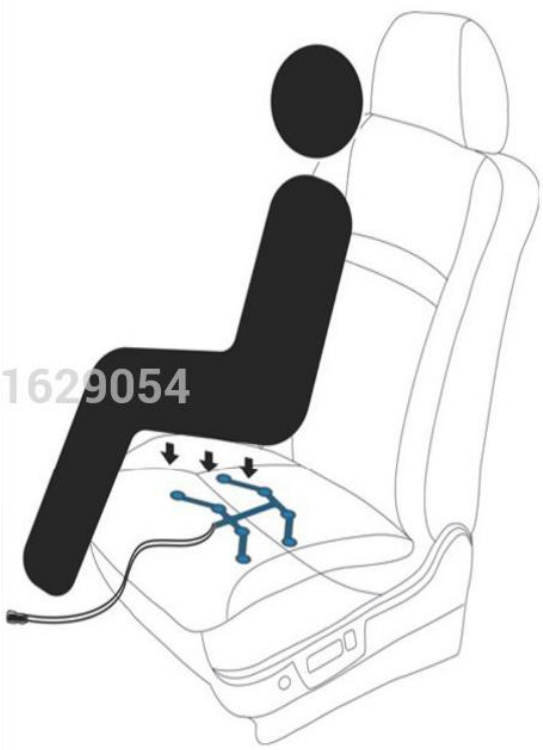


Εικόνα 30 ***(Μπλοκ διάγραμμα ενδείκτη χειρόφρενου)***

Αισθητήρας ζώνης ασφαλείας

Ο αισθητήρας, ο οποίος ελέγχει εάν η ζώνη ασφάλειας είναι δεμένη σε περίπτωση που κάποιος κάθεται στο αντίστοιχο κάθισμα μπορεί να είναι απλός κατασκευαστικά, αφού πρόκειται και εδώ για ένα απλό διακόπτη, αλλά είναι πολύ χρήσιμος ως προς την ασφάλεια των επιβατών του αυτοκινήτου. Εκτός του ότι, σε κάποια οχήματα προειδοποιεί ηχητικά τον οδηγό και τους επιβαίνοντες ότι κάποια ή κάποιες ζώνες δεν έχουν δεθεί, σε κάποια αυτοκίνητα αποτρέπει και την εκκίνηση του κινητήρα, έως ότου δεθούν όλες οι ζώνες εκείνων των καθισμάτων, στα οποία επάνω βρίσκεται άνθρωπος. Ο αισθητήρας αυτός πιο αναλυτικά περιλαμβάνει ένα push button, το οποίο είναι τοποθετημένο στο κάτω μέρος κάθε θέσης και κλείνει με το βάρος που ασκείτε επάνω του από τον επιβάτη που κάθετε στο αντίστοιχο κάθισμα και επίσης, περιλαμβάνει ένα ακόμη ένα push button, το οποίο είναι συνδεδεμένο παράλληλα με το προηγούμενο push button και ταυτόχρονα τοποθετημένο μέσα στο σύνδεσμο της ζώνης ασφάλειας, με αποτέλεσμα να πρέπει και τα δύο push button να είναι ανοιχτά ώστε η έξοδος να είναι απενεργοποιημένη και κατ' επέκταση και η φωτεινή ένδειξη καθώς και η ηχητική ειδοποίηση.

Εικόνα 31 *(Αισθητήρας ζώνης ασφαλείας)*



Εικόνα 32 *(Αισθητήρας θέσης)*

Διακόπτης φώτων μεγάλης σκάλας

Ο διακόπτης ο οποίος είναι υπεύθυνος για τον εξωτερικό φωτισμό του αυτοκινήτου βρίσκεται τοποθετημένος κάτω από το τιμόνι του αυτοκινήτου με σκοπό να είναι εύκολα προσβάσιμος από τον οδηγό. Ο διακόπτης αυτός είναι περιστροφικός και έχει τρείς θέσεις λειτουργίας. Όταν ο οδηγός γυρίσει το διακόπτη στη πρώτη θέση τότε ενεργοποιούνται τα φώτα θέσεως, στη δεύτερη θέση ενεργοποιούνται τα φώτα πορείας και στη τρίτη θέση η μεγάλη σκάλα. Η ενεργοποίηση της φωτεινής ένδειξής της μεγάλης σκάλας στο καντράν του αυτοκινήτου γίνεται άμεσα με τη παράλληλή τροφοδότηση των λαμπτήρων του αυτοκινήτου, χωρίς τη χρήση κάποιου έξτρα διακόπτη ή αισθητήρα.

Εικόνα 33 ***(Μπλοκ διάγραμμα ενδείκτη μεγάλης σκάλας)***

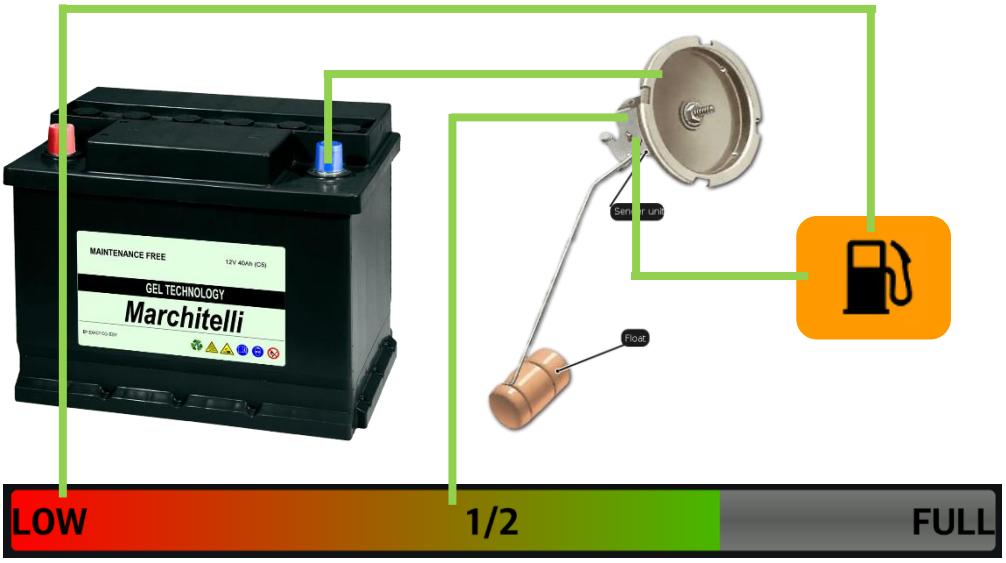
Διακόπτης φλάς

Η ενεργοποίηση και η απενεργοποίηση των φλάς (αριστερού-δεξιού) γίνεται μέσω του διακόπτη που είναι υπεύθυνος για τον εξωτερικό φωτισμό του αυτοκινήτου, μόνο που σε αυτή τη περίπτωση για να ενεργοποιηθεί π.χ. το αριστερό φλάς ο οδηγός δε θα πρέπει να περιστρέψει το διακόπτη, όπως περιγραφή στην προηγούμενη παράγραφο για την ενεργοποίηση του φωτισμού, αλλά να τον σπρώξει ελαφρώς κάθετα προς τα κάτω. Αντιθέτως η απενεργοποίηση των φλάς γίνεται με αυτόματο τρόπο, όταν το τιμόνι του αυτοκινήτου περιστραφεί (εάν το τιμόνι δε περιστραφεί τότε ο οδηγός θα πρέπει από μόνος του να απενεργοποιήσει το φλάς).

Εικόνα 34 *(Μπλοκ διάγραμμα δεξιού και αριστερού ενδείκτη φλάς)*

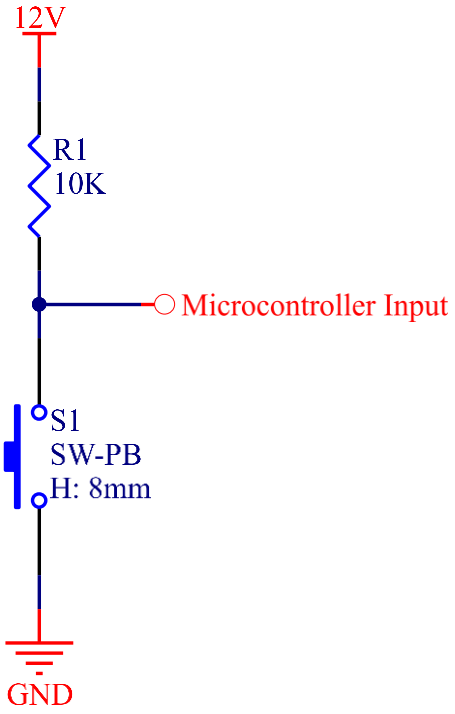
Αισθητήρας στάθμης καυσίμου

Ο αισθητήρας αυτός έχει συνήθως δυο εξόδους, οπού η μία είναι ψηφιακή και ενεργοποιείτε μετά τη πτώση της στάθμης του καυσίμου από ένα όριο και κάτω και η άλλη είναι αναλογική και η τάση εξόδου της της μεταβάλλεται ανάλογα με τη στάθμη του καυσίμου στο ντεπόζιτο του αυτοκινήτου.



3.1.2 Προσομοίωση αισθητηρίων

Τα αισθητήρια-διακόπτες που προαναφέρθηκαν είναι αυτά τα οποία ελέγχει ο μικροελεγκτής με σκοπό να ενημερώνει το οδηγό μέσω του τάμπλετ για την κατάσταση στην οποία βρίσκεται ο αντίστοιχος εξοπλισμός του αυτοκινήτου. Τα συγκεκριμένα αισθητήρια-διακόπτες λόγω του ότι έχουν κάποιο κόστος, στην υλοποίησή του συστήματος που έγινε στα πλαίσια της διπλωματικής εργασίας δεν χρησιμοποιούνται πραγματικά αισθητήρια, αλλά αυτά τα αισθητήρια-διακόπτες προσομοιώνονται με τη χρήση μικροδιακοπτών push buttons. Στην *εικόνα 35* φαίνεται το σχηματικό διάγραμμα των push buttons, όπου όταν ο διακόπτης δεν είναι πατημένος ο μικροελεγκτής διαβάζει στην αντίστοιχη είσοδο του λογικό '1', ενώ όταν ο διακόπτης πατηθεί ο μικροελεγκτής διαβάζει λογικό '0'. Πιο αναλυτικά, όταν το push button δεν είναι πατημένο, τότε η αντίσταση «R1» δε διαρρέεται από ρεύμα, με αποτέλεσμα να μη δημιουργείται πτώση τάσης επάνω της και άρα η είσοδος του μικροελεγκτή να ισούται με τη τάση της πηγής, δηλαδή με 12volt (όταν ο κινητήρας του αυτοκινήτου βρίσκεται σε λειτουργία και κατ' επέκταση δουλεύει και το δυναμό, η τάση σε κάποια αυτοκίνητα αγγίζει μέχρι και τα 14,5Volt). Αντιθέτως, όταν το push button πατηθεί, η αντίσταση διαρρέεται από ρεύμα, με αποτέλεσμα η είσοδος του μικροελεγκτή να συνδέεται στη γείωση και άρα να ισούται με 0Volt.



Εικόνα 35 ***(Σχηματικό κύκλωμα push-button)***

3.1.3 Έξοδοι συστήματος

Εκτός από τα αισθητήρια με τα οποία επικοινωνεί ο μικροελεγκτής, υπάρχουν και κάποιες έξοδοι τις οποίες ελέγχει ο μικροελεγκτής και της ενεργοποιεί κατ' εντολή του οδηγού, μέσω της οθόνης αφής του τάμπλετ. Επίσης, όλες αυτές οι έξοδοι ενεργοποιούνται και απενεργοποιούνται μέσω ρελέ, επειδή τα φορτία που διαχειρίζονται απαιτούν ρεύμα της τάξης των Ampere, ενώ ο μικροελεγκτής στην έξοδο του μπορεί να δώσει μόνο κάποιες δεκάδες mA. Οι ελεγχόμενες από τον οδηγό αυτές έξοδοι σκοπό έχουν την ενεργοποίηση και απενεργοποίηση,

- των κλειδαριών
- του ανεμιστήρα ψύξης του ψυγείου - βεντιλατέρ (ventilating fan)
- των αντιστάσεων θέρμανσης του πίσω παρμπρίζ (defrost window)
- του μοτέρ ανοίγματος/κλεισίματος της πόρτας του πορτ-μπαγκάζ (trunk)
- του εσωτερικού φωτισμού του αυτοκινήτου (πλαφονιέρας - interior light)
- των αλάρμ (hazard lights)
- και της μίζας (starter motor)

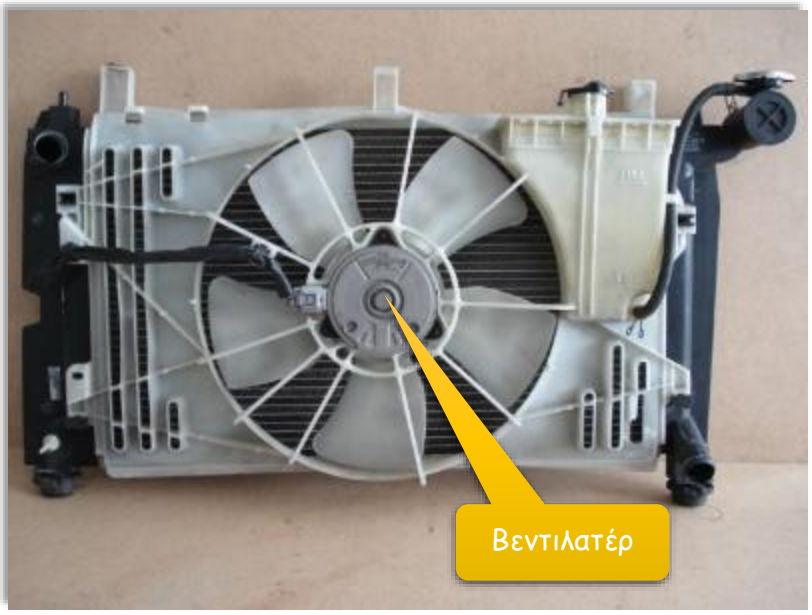
Ηλεκτρομαγνητική κλειδαριά

Οι κλειδαριές του αυτοκινήτου πέρα από το μηχανισμό με τις ντίζες τον οποίο έχουν, περιλαμβάνουν επίσης και μία ηλεκτρομαγνητική «αρπαγή» (High Power Door Lock Actuator), η οποία αποτελείται από ένα έμβολο τοποθετημένο μέσα σε ένα ηλεκτρομαγνήτη. Όταν ο ηλεκτρομαγνήτης ενεργοποιηθεί τότε το έμβολο αυτό έλκετε προς τα πίσω με αποτέλεσμα οι κλειδαριά του αυτοκινήτου να ενεργοποιείται και η πόρτα να ασφαλίζει, ενώ όταν η ηλεκτρομαγνητική αρπαγή απενεργοποιηθεί τότε το έμβολο επανέρχεται στην αρχική του κατάσταση προς τα έξω, με αποτέλεσμα η πόρτα να απ' ασφαλίζει.

Εικόνα 36 *(Ηλεκτρομαγνητική κλειδαριά)*

Βεντιλατέρ

Το βεντιλατέρ του αυτοκινήτου είναι ένας ηλεκτρικός κινητήρας, ο οποίος ενεργοποιείται είτε αυτόματα είτε μέσω του οδηγού και είναι τοποθετημένος μπροστά από το ψυγείο του αυτοκινήτου. Σκοπό έχει την ψύξη του υγρού που υπάρχει μέσα στο ψυγείο του αυτοκινήτου, ώστε η θερμοκρασία του κινητήρα να διατηρείται σε σχετικά χαμηλά όρια (έχει παρόμοια λειτουργεία με αυτή που έχει το ανεμιστηράκι, που είναι τοποθετημένο επάνω στον επεξεργαστή ενός υπολογιστή).



Εικόνα 37 ***(Βεντιλατέρ τοποθετημένο στο ψυγείο του αυτοκινήτου)***

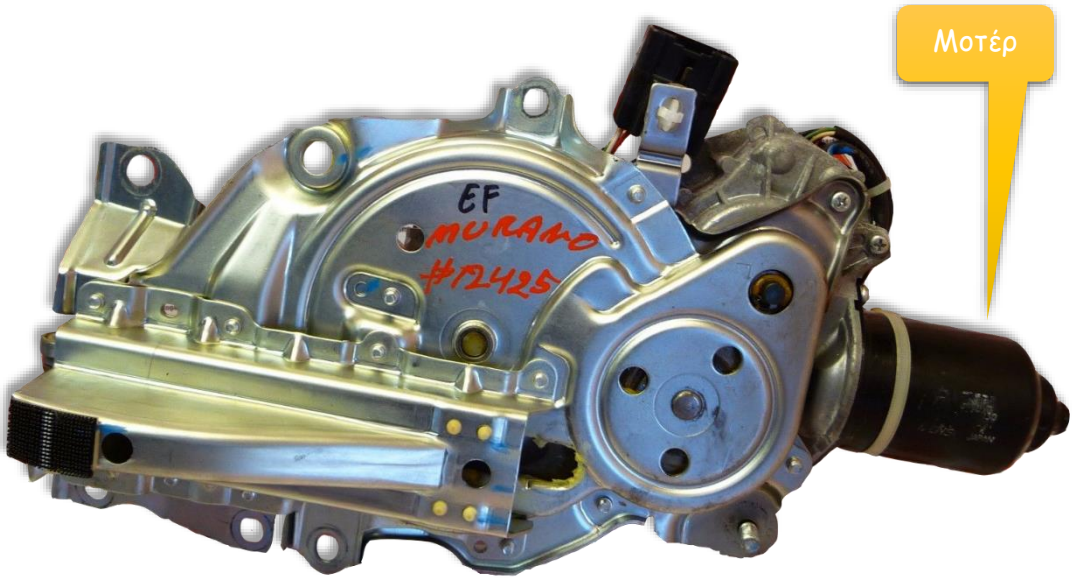
Πίσω θερμαινόμενο τζάμι

Ένας αρκετά χρήσιμο όσο αφορά τον εξοπλισμό του αυτοκινήτου είναι το πίσω θερμαινόμενο τζάμι, το οποίο πλέον αποτελεί βασικό εξοπλισμό και όχι προαιρετικό όπως πριν μία δεκαετία περίπου. Το θερμαινόμενο τζάμι έχει εσωτερικά μία συστοιχία ηλεκτρικών αντιστάσεων, οι οποίες όταν διαρρέονται από ρεύμα θερμαίνονται, με αποτέλεσμα να θερμαίνεται αντίστοιχα και το τζάμι και να μη θολώνει τις κρύες μέρες. Η ενεργοποίηση των αντιστάσεων αυτών γίνεται μέσω ενός διακόπτη και όχι αυτόματα (στα περισσότερα αυτοκίνητα).

Εικόνα 38 *(Θερμαινόμενο τζάμι)*

Πορτ-μπαγκάζ

Τη τελευταία πενταετία όλο και πληθαίνουν τα αυτοκίνητα των οποίων το πορτ-μπαγκάζ δεν ανοίγει πλέον με χειροκίνητο τρόπο, αλλά με το πάτημα ενός διακόπτη, ο οποίος ενεργοποιεί ένα μοτέρ. Όταν σε ένα μοτέρ αλλαχτεί η πολικότητα της τάσης στους ακροδέκτες τροφοδοσίας του, τότε αλλάζει και η φορά περιστροφής του. Με το τρόπο αυτό επιτυγχάνεται και το άνοιγμα και το κλείσιμο του πορτ-μπαγκάζ.

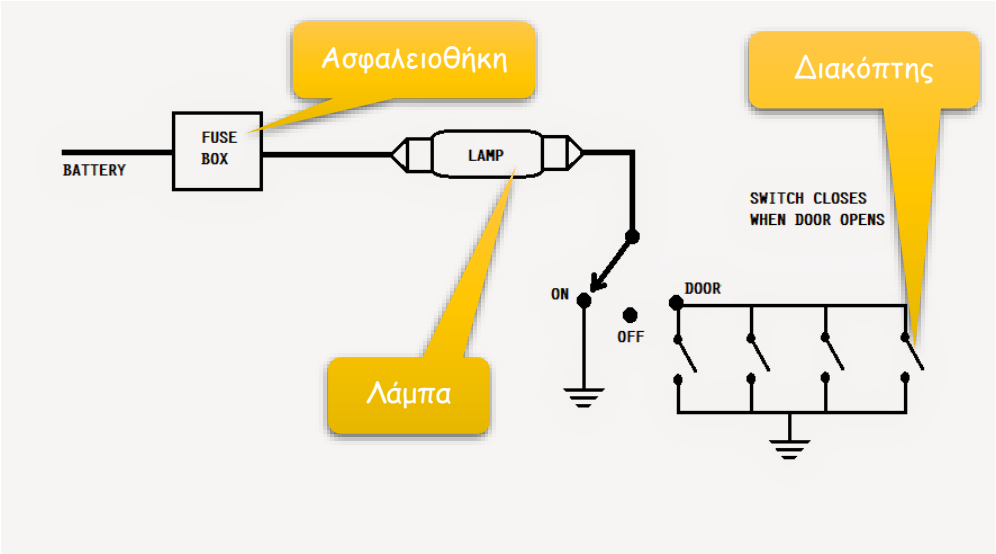


Εικόνα 39 *(Μηχανισμός ανοίγματος/κλεισίματος πορτ-μπαγκάζ)*

Πλαφονιέρα αυτοκινήτου

Η πλαφονιέρα αποτελείτε από ένα απλό λαμπάκι πυράκτωσης ή LED και σκοπό έχει το φωτισμό στο εσωτερικό του αυτοκινήτου. Το λαμπάκι αυτό συνδέετε σε σειρά με τέσσερις διακόπτες, οι οποίοι είναι παράλληλα τοποθετημένοι μεταξύ τους σε κάθε πόρτα του αυτοκινήτου. Σε περίπτωση που κλείσει έστω και ένας διακόπτης, τότε το φως ανάβει στο εσωτερικό του αυτοκινήτου για όσο χρόνο παραμένει τουλάχιστον και ένας διακόπτης κλειστός.

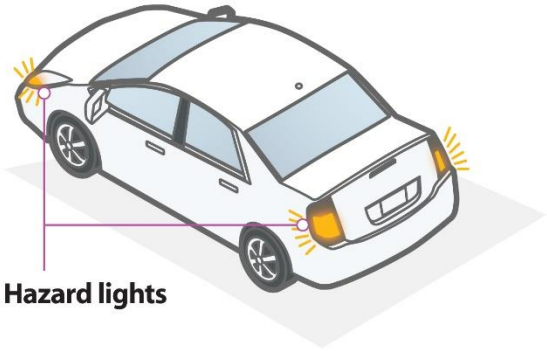
Εικόνα 40 *(Διακόπτης πόρτας αυτοκινήτου)*



Εικόνα 41 *(Σχηματικό κύκλωμα πλαφονιέρας αυτοκινήτου)*

Αλάρμ

Τα αλάρμ του αυτοκινήτου είναι οι τέσσερις λάμπες των φλάς συνδεδεμένες όλες παράλληλα σε ένα διακόπτη, ο οποίος βρίσκεται στο ταμπλό του αυτοκινήτου και είναι σε σημείο τέτοιο, οπού θα είναι εύκολα προσβάσιμος σε περίπτωση έκτακτης ανάγκης κατά τη διάρκεια της οδήγησης.



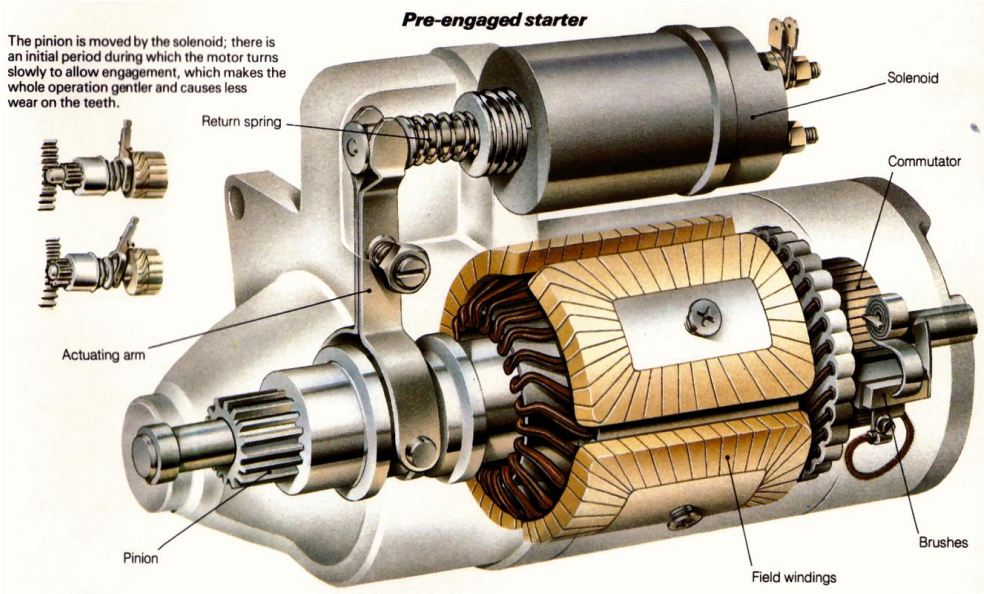
Εικόνα 42 *(Αλάρμ αυτοκινήτου)*

Εικόνα 43 *(Διακόπτης αλάρμ)*

Μίζα

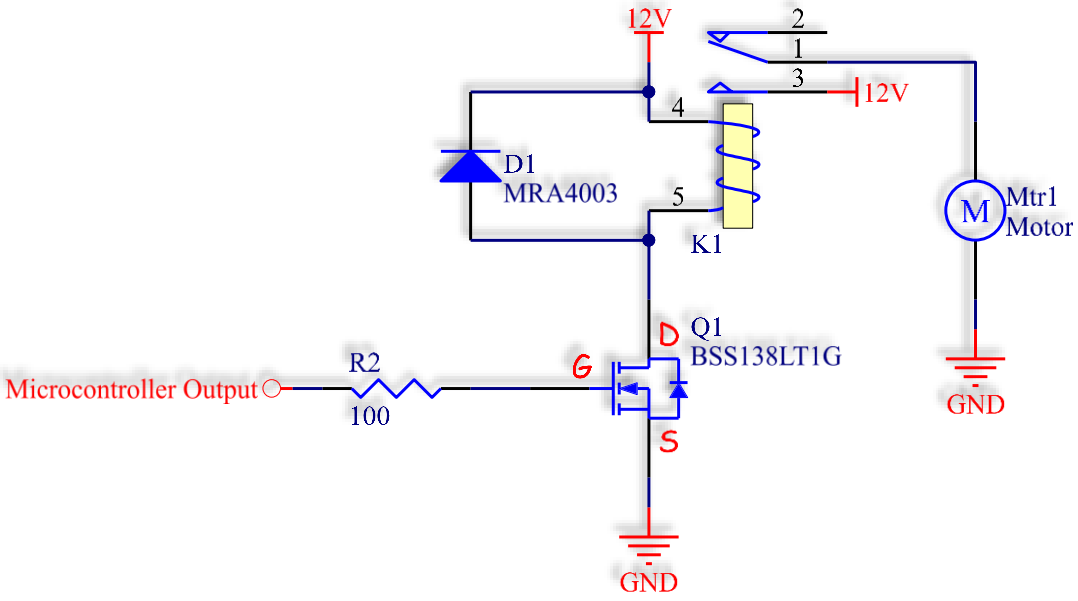
Η μίζα του αυτοκινήτου είναι ένας ηλεκτρικός κινητήρας και μάλιστα ο πιο δυνατός σε ισχύ ήλεκτρο-κινητήρας που έχει το αυτοκίνητο. Το ρεύμα που τραβάει κυμαίνεται στα 60Ampere (ανάλογα με το τύπο της μίζας). Είναι ένα από τα βασικότερα εξαρτήματα του αυτοκινήτου, γιατί εξαιτίας αυτού το εξαρτήματος παίρνει μπρος ο κινητήρας του αυτοκινήτου. Η μίζα του αυτοκινήτου ενεργοποιείται για ένα πολύ μικρό χρονικό διάστημα, το οποίο απαιτείται για να γίνει η εκκίνηση του κινητήρα του αυτοκινήτου και έπειτα σταματάει να δουλεύει. Η ενεργοποίηση της μίζας γίνεται μέσω του κεντρικού διακόπτη (διακόπτης εκκίνησης – ignition switch) του αυτοκινήτου.

Εικόνα 44 *(Διακόπτης εκκίνησης)*



Εικόνα 45 *(Μίζα)*

Στην *εικόνα 46* φαίνεται το σχηματικό διάγραμμα των ρελέ, τα οποία συνδέονται στις εξόδους του μικροελεγκτή με σκοπό την προσομοίωση του εξοπλισμού (θερμαινόμενο τζάμι, μίζα κτλ.) που περιγράφει παραπάνω.



Εικόνα 46 *(Κύκλωμα οδήγησης ρελέ)*

3.2 Αρχιτεκτονική εφαρμογής Android

Το λογισμικό κομμάτι του συστήματος αποτελείται από δύο μέρη. Το ένας μέρος είναι το πρόγραμμα, το οποίο τρέχει ο μικροελεγκτής και είναι γραμμένο σε γλώσσα C και το άλλο μέρος είναι η εφαρμογή Android, η οποία τρέχει σε ένα τάμπλετ (LG GPad v8.3). **Και τα δύο προγράμματα παρέχονται μαζί με την αναπτυξιακή πλακέτα** και περιλαμβάνουν το κομμάτι της USB επικοινωνίας μεταξύ των δύο συσκευών, καθώς και δείγμα (sample) κώδικα, ο οποίος αφορά τον έλεγχο εισόδων και εξόδων του μικροελεγκτή, μέσω του τάμπλετ. **Στους κώδικες τους οποίους παρέχει η εταιρία Microchip έγιναν οι απαραίτητες τροποποιήσεις κυρίως στο γραφικό περιβάλλον της εφαρμογής Android.**

Η εφαρμογή Android όπως υλοποιήθηκε στα πλαίσια της παρούσας εργασίας περιλαμβάνει τις οχτώ παρακάτω activities - κλάσεις:

- SplashScreenActivity
- PasswordActivity
- BasicAccessoryDemo
- InfoActivity
- MyDeviceAdminReceiver
- LEDControl
- USBAccessoryManager
- USBAccessoryManagerMessage

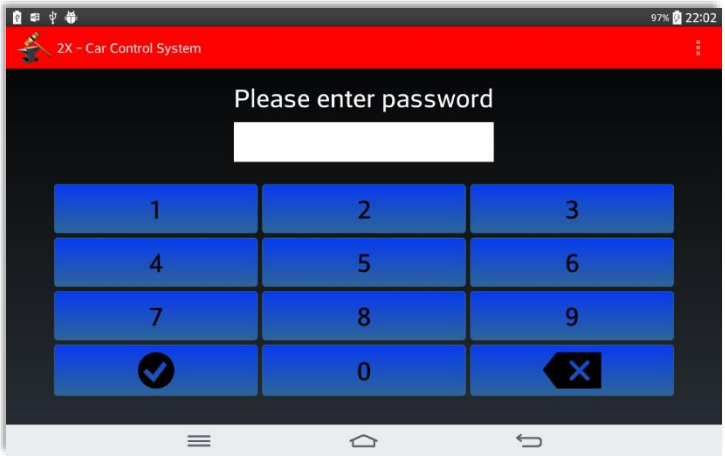
SplashScreenActivity

Η «SplashScreenActivity» αποτελεί την οθόνη υποδοχής της εφαρμογής, στην οποία εμφανίζεται το Logo της μάρκας του αυτοκινήτου για δύο sec περίπου και έπειτα η εφαρμογή μεταβαίνει αυτόματα στο «PasswordActivity».

Εικόνα 47 *(SplashScreenActivity)*

PasswordActivity

Η «PasswordActivity» περιέχει την οθόνη εισαγωγής κωδικού (password), όπου αποτελείται από ένα πληκτρολόγιο με αριθμούς μόνο. Ο χρήστης έχει τη δυνατότητα να εισάγει μέχρι και τρείς φορές το σωστό κωδικό. Σε περίπτωση αποτυχίας εισαγωγής του έγκυρου κωδικού και τις τρείς φορές, τότε ζητείται από το χρήστη να εισάγει τον οκταψήφιο κωδικό ασφαλείας. Σε περίπτωση λάθους ή επιτυχίας εισαγωγής του κωδικού ακούγονται και τα αντίστοιχα φωνητικά μηνύματα.



Εικόνα 48 *(PasswordActivity)*

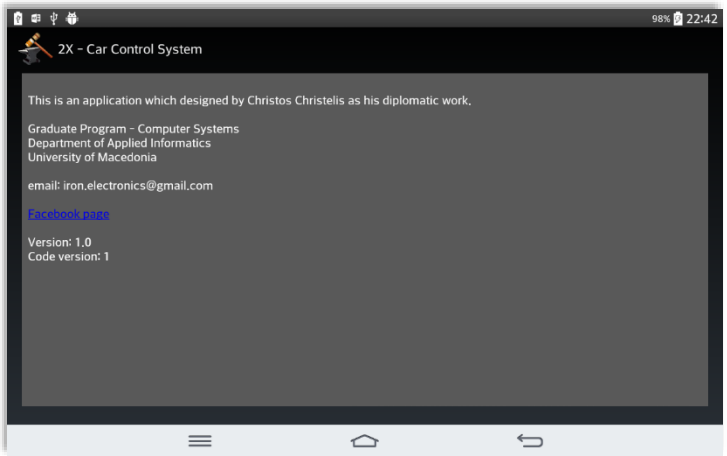
BasicAccessoryDemo

Η «BasicAccessoryDemo» περιέχει την οθόνη απεικόνισης και ελέγχου των εισόδων και εξόδων του μικροελεγκτή. Εκτός από το κομμάτι απεικόνισης, περιλαμβάνει και ένα διαχειριστή (handler), ο οποίος λαμβάνει μηνύματα από ένα νήμα εκτέλεσης (thread) το οποίο ονομάζεται «USB Manager» και είναι υπεύθυνο για την αμφίδρομη USB επικοινωνία μεταξύ τάμπλετ και μικροελεγκτή. Επίσης, περιλαμβάνει και διάφορες μεθόδους (συναρτήσεις), οι οποίες είναι υπεύθυνες για την ανανέωση της κατάστασης (σε γραφικό και λειτουργικό επίπεδο) των εισόδων και εξόδων του μικροελεγκτή στην οθόνη απεικόνισης. **Η activity «BasicAccessoryDemo» υπήρχε υλοποιημένη ως ένα σημείο από την εταιρία Microchip** και τροποποιήθηκε καταλλήλως, ώστε να εξυπηρετεί του σκοπούς και τις λειτουργίες του *συστήματος ελέγχου λειτουργιών ενός οχήματος*.

Εικόνα 49 (*BasicAccessoryDemo* - *Activity*)

InfoActivity

Η «InfoActivity» περιλαμβάνει μία οθόνη με πληροφορίες σχετικά με την υλοποίηση της εφαρμογής και ο χρήστης μπορεί να μεταβεί σε αυτή μόνο μέσω της activity «BasicAccessoryDemo» πατώντας επάνω στο αντίστοιχο κουμπί.



Εικόνα 50 (*InfoActivity*)

MyDeviceAdminReceiver

Η κλάση «MyDeviceAdminReceiver» επεκτείνει (extends) τη κλάση «DeviceAdminReceiver» και περιέχει τις απαραίτητες μεθόδους, ώστε η εφαρμογή να αποκτά διαχειριστικά δικαιώματα στο τάμπλετ, με σκοπό να μπορεί να κλειδώσει και να απενεργοποιεί την οθόνη του όταν αποσυνδεθεί ή διακοπή η τροφοδοσία στη πλακέτα που περιλαμβάνει το μικροελεγκτή και συνδέεται μέσω micro USB με το τάμπλετ.

LEDControl

Η κλάση «LEDControl» επεκτείνει τη κλάση των κουπιών με εικόνα (ImageButton), ώστε σε κάθε πάτημα του χρήστη στο αντίστοιχο εικονίδιο της κάθε εξόδου να ενεργοποιείτε η αντίστοιχη έξοδος μέσω του διαχειριστή (handler), ο οποίος είναι υπεύθυνος για την επικοινωνία με το μικροελεγκτή και ταυτόχρονα να ενημερώνεται και εικονικά η κατάσταση στην οποία βρίσκεται η έξοδος. Με το πρώτο π.χ. πάτημα στο αντίστοιχο κουμπί (button) να ενεργοποιείται η έξοδος (το ρελέ) και να αλλάζει ταυτόχρονα και η εικόνα από το κουμπί και με το δεύτερο πάτημα του χρήστη στο ίδιο το κουμπί να απενεργοποιείτε η έξοδος (με την οποία το κουμπί είναι διασυνδεδεμένο) και συνάμα να αλλάζει εικονική κατάσταση από το κουμπί. **Η κλάση «LEDControl» υπήρχε υλοποιημένη ως ένα σημείο από την εταιρία Microchip** και τροποποιήθηκε καταλλήλως, ώστε να εξυπηρετεί του σκοπούς και τις λειτουργίες του *συστήματος ελέγχου λειτουργιών ενός οχήματος*.

USBAccessoryManager

Η κλάση «USBAccessoryManager» επιτρέπει τη σύνδεση υλικού (hardware) μέσω της «USB» θύρας του τάμπλετ (ή του smartphone). Τα προς σύνδεση εξαρτήματα θα πρέπει να είναι συμβατά ως προς το πρωτόκολλο επικοινωνίας του λογισμικού Android. Αυτό επιτρέπει στις συσκευές Android, οι οποίες δεν μπορούν να λειτουργήσουν ως «USB host» να έχουν τη δυνατότητα αλληλεπίδρασης με «USB» εξαρτήματα-εξοπλισμό. Όταν μία συσκευή με Android λογισμικό βρίσκεται σε λειτουργία «USB accessory mode», τότε το υλικό (π.χ. η πλακέτα με το μικροελεγκτή) που συνδέεται με τη συσκευή Android (π.χ. τάμπλετ) λειτουργεί ως «host», με αποτέλεσμα το ίδιο το υλικό να παρέχει τροφοδοσία στο δίαυλο (bus) «USB».

Πηγή: Android Developers - Device Administration

USBAccessoryManagerMessage

Η κλάση «USBAccessoryManagerMessage» χρησιμοποιείται για την αποστολή μηνυμάτων από το νήμα ανάγνωσης (read thread) του «USBAccessory», στο «GUI» (Graphics User Interface) thread, ώστε να ειδοποιεί «GUI thread» για διάφορα γεγονότα (events), τα οποία σχετίζονται με το «USBAccessory», όπως ότι υπάρχουν διαθέσιμα δεδομένα ή για τη σύνδεση υλικού στη συσκευή Android.

3.3 Πρωτόκολλο επικοινωνίας (Android USB)

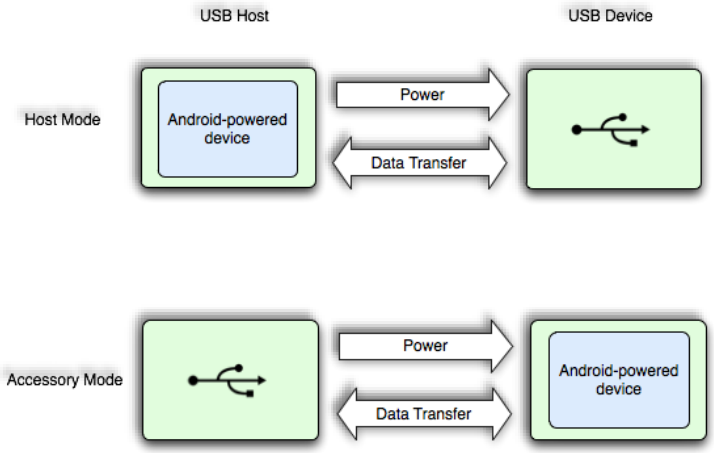
Το λογισμικό Android υποστηρίζει μία πληθώρα «USB» περιφερειακών συσκευών και αξεσουάρ (υλικό και συσκευές, που βασίζονται στο πρωτόκολλο «Android accessory») μέσω δύο λειτουργιών, όπου η μία είναι το «USB accessory» και η άλλη το «USB host».

Στη λειτουργία «USB accessory», η εξωτερική «USB» συσκευή λειτουργεί ως «USB host». Μερικά παραδείγματα αξεσουάρ, τα οποία λειτουργούν ως «USB host» είναι οι ρομποτικοί ελεγκτές (robotics controllers), μουσικός εξοπλισμός (musical equipment), συσκευές αναγνώρισης καρτών (card readers) κ.α. Αυτό επιτρέπει σε συσκευές Android, οι οποίες δεν έχουν δυνατότητες «host», την ικανότητα να αλληλοεπιδρούν με «USB hardware». Τα «USB» αξεσουάρ θα πρέπει να είναι σχεδιασμένα, ώστε να δουλεύουν με συσκευές που έχουν Android λογισμικό, καθώς και να ακολουθούν το πρωτόκολλο «Android accessory communication protocol».

Στη λειτουργία «USB host», η Android συσκευή (π.χ. τάμπλετ) συμπεριφέρεται ως «host». Κάποια παραδείγματα «USB» συσκευών, τα οποία συνδέονται σε Android συσκευές είναι τα πληκτρολόγια, τα ποντίκια, οι κάμερες κ.α. Συσκευές «USB», που έχουν σχεδιαστεί για ένα ευρύ φάσμα εφαρμογών έχουν τη δυνατότητα να αλληλοεπιδρούν με Android εφαρμογές και να μπορούν να επικοινωνούν με έγκυρο τρόπο με την Android συσκευή.

Η *εικόνα 51* δείχνει τις διαφορές μεταξύ λειτουργιών «USB accessory» και «USB host». Όταν η συσκευή Android (τάμπλετ) είναι σε «host» λειτουργία, δρα ως «USB host» και τροφοδοτεί το δίαυλο (bus) επικοινωνίας. Αντιθέτως, όταν η συσκευή Android (τάμπλετ) είναι σε λειτουργία «USB accessory», τότε η συσκευή που συνδέετε επάνω στην Android συσκευή (τάμπλετ) λειτουργεί ως «host» και τροφοδοτεί το δίαυλο επικοινωνίας.

Οι λειτουργίες «USB accessory» και «USB host» που υποστηρίζονται άμεσα στο λογισμικό Android 3.1 (επίπεδο API 12) ή σε νεότερες εκδόσεις του λογισμικού. Η λειτουργία «USB accessory» είναι επίσης «backported» στο Android 2.3.4 (επίπεδο API 10), ως ένα πρόσθετο στοιχείο της βιβλιοθήκης, με σκοπό να υποστηρίξει ένα ευρύτερο φάσμα συσκευών.



Εικόνα 51 ***(USB Host and Accessory Modes)***

Πηγή: Android Developers - USB Host and Accessory

4. Υλοποίηση – Ανάπτυξη Κώδικα

Όπως προαναφέρθηκε στο προηγούμενο κεφάλαιο η Android εφαρμογή περιλαμβάνει 8 activities – κλάσεις. Η υλοποίηση της «SplashScreenActivity» activity έγινε εξολοκλήρου από την αρχή. Η activity αυτή περιλαμβάνει το παρακάτω κώδικα, όπου αρχικά γίνεται παράκαμψη της οθόνης κλειδώματος (μόνο για την συγκεκριμένη εφαρμογή – το τηλέφωνο δεν ξεκλειδώνει τελείως), έπειτα γίνεται αναπαραγωγή ενός ηχητικού μηνύματος «Welcome» και στη συνέχεια, με τη χρήση ενός διαχειριστή γίνεται μία χρονοκαθυστέρηση περίπου των δύο δευτερολέπτων, ώστε ο χρήστης κατά την έναρξη της εφαρμογής να δει το έμβλημα (logo) του αυτοκινήτου του. Μετά το πέρας των δύο δευτερολέπτων η εφαρμογή μεταβαίνει στη «PasswordActivity».

```
// Unlock screen.
Window window = getWindow( );
window.addFlags( WindowManager.LayoutParams.FLAG_SHOW_WHEN_LOCKED
        | WindowManager.LayoutParams.FLAG_DISMISS_KEYGUARD
        | WindowManager.LayoutParams.FLAG_KEEP_SCREEN_ON );

//-------------------------------------------------------------------
----

mPlayer = MediaPlayer.create( SplashScreenActivity.this,
R.raw.welcome );
mPlayer.start(); //Play the audio file
//The audio file is in /res/raw/ and is an OGG file

// Reset Ins, Outs and bars state.
BasicAccessoryDemo.disconnect_fl = true;

//-------------------------------------------------------------------
----

// New Handler to start the Menu-Activity and close this Splash-
Screen after some seconds.
new Handler( ).postDelayed( new Runnable( ) {
    @Override
    public void run( ) {
        // Create an Intent that will start the "Password Activity".
        Intent passwordAct_Int = new Intent(
SplashScreenActivity.this, PasswordActivity.class );
        startActivity( passwordAct_Int );
        finish( );
    }
}, SPLASH_DISPLAY_TIME );
```

Τέλος, με το που συνδέεται η πλακέτα μέσω «USB» στη συσκευή Android (τάμπλετ), πάντα η εφαρμογή ανοίγει πρώτα την activity «SplashScreenActivity». Αυτό

επιτυγχάνεται με τη προσθήκη των παρακάτω «intent-filter», στο Manifest αρχείο της εφαρμογής.

```
<!-- LogoActivity -->
<activity
    android:name=".SplashScreenActivity"
    android:screenOrientation="landscape"
    android:launchMode="singleInstance"
    android:theme="@android:style/Theme.NoTitleBar.Fullscreen" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
    <intent-filter>
        <action
android:name="android.hardware.usb.action.USB_ACCESSORY_ATTACHED" />
    </intent-filter>

    <meta-data

android:name="android.hardware.usb.action.USB_ACCESSORY_ATTACHED"
        android:resource="@xml/accessory_filter" />
</activity>
```

Η «PasswordActivity» υλοποιήθηκε και αυτή εξολοκλήρου από την αρχή. Και αυτή, όπως και η «SplashScreenActivity» περιλαμβάνει τον αντίστοιχο κώδικα ώστε να γίνεται παράκαμψη της κλειδαριάς όταν «ενεργοποιείτε» η συγκεκριμένη activity. Επίσης, περιλαμβάνει ένα «BroadcastReceiver», ο οποίος σκοπό έχει να ελέγχει εάν η πλακέτα είναι συνδεδεμένη και τροφοδοτεί το κανάλι επικοινωνίας (USB bus) με τάση. Σε περίπτωση που η τροφοδοσία διακοπή (π.χ. επειδή ο οδηγός έκλεισε το διακόπτη του αυτοκινήτου) τότε αυτόματα γίνεται απενεργοποίηση της οθόνης του τάμπλετ, καθώς και ενεργοποίηση της κλειδαριάς (εφόσον ο χρήστης έχει ενεργοποιημένο το κλείδωμα της συσκευής) και αναπαραγωγή ενός ηχητικού μηνύματος «Goodbye».

```
BroadcastReceiver receiver = new BroadcastReceiver( ) {
    public void onReceive( Context context, Intent intent ) {

        powerManager = (PowerManager) getSystemService( POWER_SERVICE
);
        int plugged = intent.getIntExtra(
BatteryManager.EXTRA_PLUGGED, -1 );

        if ( plugged == BatteryManager.BATTERY_PLUGGED_AC ) {
            // on AC power
        }
        else if ( plugged == BatteryManager.BATTERY_PLUGGED_USB ) {
```

```
            // on USB power
        }
        else if ( plugged == 0 & powerManager.isScreenOn ( ) ) {
            // on Battery power

                // Enable the Administrator mode.
                DevicePolicyManager deviceManger =
(DevicePolicyManager) getSystemService(
                        Context.DEVICE_POLICY_SERVICE );
                ActivityManager activityManager = (ActivityManager)
getSystemService(
                        Context.ACTIVITY_SERVICE );
                ComponentName compName = new ComponentName(
getApplicationContext( ),
                        MyDeviceAdminReceiver.class);
                Intent device_policy_manager_Int = new Intent(
DevicePolicyManager
                        .ACTION_ADD_DEVICE_ADMIN );
                device_policy_manager_Int.putExtra(
DevicePolicyManager.EXTRA_DEVICE_ADMIN,
                        compName );
                device_policy_manager_Int.putExtra(
DevicePolicyManager.EXTRA_ADD_EXPLANATION,
                        "Additional text explaining why this needs to
be added." );
                startActivityForResult( device_policy_manager_Int,
RESULT_ENABLE );

                // Check if the Administrator is enabled.
                boolean active = deviceManger.isAdminActive( compName
);

                if ( active ) {
                    Log.i("isAdminActive", "Admin enabled!");

                    while ( SplashScreenActivity.mPlayer.isPlaying( )
) { /* Wait here! */ }
                    SplashScreenActivity.mPlayer =
MediaPlayer.create( PasswordActivity.this, R.raw.goodbye );
                    SplashScreenActivity.mPlayer.start( ); //Play the
audio file
                    //The audio file is in /res/raw/ and is an OGG
file

                    // Lock device.
                    deviceManger.lockNow( );

                    while ( SplashScreenActivity.mPlayer.isPlaying( )
) { /* Wait here! */ }

                    finish( );
                    onDestroy( );
                }
        }
        else {
            // intent didnt include extra info
```

```
        }
    }
};

IntentFilter power_type_IntF = new IntentFilter(
Intent.ACTION_BATTERY_CHANGED );
registerReceiver( receiver, power_type_IntF );
```

Η «PasswordActivity» επίσης περιλαμβάνει και του αντίστοιχους ωτακουστές κουμπιών (Button Listeners), ώστε να γίνεται η εισαγωγή και έπειτα ο έλεγχος εγκυρότητας του κωδικού εισόδου.

```
// Κουμπί "1".
final Button one_B = (Button) findViewById( R.id.b_1 );
one_B.setOnClickListener( new View.OnClickListener( ) //theto enan
otakousti
{
    public void onClick( View v ) {

        String str_pass;

        if ( attempts < 3 ) {
            for ( int i = 0; i < temp_four_code_pass.length; i++ )
                if ( temp_four_code_pass[ i ] == 'x' ) {
                    temp_four_code_pass[ i ] = '1';
                    str_pass = String.valueOf( temp_four_code_pass[ i
] );
                    pass_eT.append( str_pass );
                    break;
                }
        }

        else {
            for ( int j = 0; j < temp_recovery_pass.length; j++ )
                if ( temp_recovery_pass[ j ] == 'x' ) {
                    temp_recovery_pass[ j ] = '1';
                    str_pass = String.valueOf( temp_recovery_pass[ j
] );
                    pass_eT.append( str_pass );
                    break;
                }
        }
    }
});
```

Ο «βασική» υλοποίηση του κώδικα της κλάσης «BasicAccessoryDemo» έχει γίνει από την εταιρία «Microchip» και βάση αυτού του κώδικα έγιναν οι απαραίτητες τροποποιήσεις, ώστε να υλοποιηθεί το σύστημα ελέγχου λειτουργιών ενός οχήματος. Η κλάση αυτή περιλαμβάνει ένα διαχειριστή γεγονότων (handler), ο οποίος ανταλλάσσει δεδομένα με το μικροελεγκτή. Τα δεδομένα που ανταλλάσσονται αφορούν την

ενεργοποίηση-απενεργοποίηση εξόδων (ρελέ) του μικροελεγκτή και την κατάσταση στην οποία βρίσκονται κάποιες από τις εισόδους (αισθητήρια-διακόπτες του αυτοκινήτου) του μικροελεγκτή. Ο μικροελεγκτής και το τάμπλετ επικοινωνούν κάθε φορά με την αποστολή δύο «byte», όπου το ένα «byte» περιλαμβάνει την εντολή (δηλαδή εάν πρόκειται για είσοδο ή για έξοδο του μικροελεγκτή) και το άλλο «byte» περιλαμβάνει τα δεδομένα (π.χ. για τη ενεργοποίηση-απενεργοποίηση ποιας εξόδου πρόκειται).

```
    /**
     * Handler for receiving messages from the USB Manager thread or
    *   the LED control modules
     */
    private Handler handler = new Handler() {
       @Override
       public void handleMessage( Message msg ) {
          byte[ ] commandPacket = new byte[ 2 ];

         switch( msg.what )
         {
         case UPDATE_LED_SETTING:
            if( accessoryManager.isConnected( ) == false ) {
               return;
            }

            commandPacket[ 0 ] = UPDATE_LED_SETTING;
            commandPacket[ 1 ] = 0;

            if( ( (LEDControl) findViewById( R.id.out_1 ) ).getState(
) ) {
               commandPacket[ 1 ] |= OUT_1;
            }

            if( ( (LEDControl) findViewById( R.id.out_2 ) ).getState(
) ) {
               commandPacket[ 1 ] |= OUT_2;
            }

            if( ( (LEDControl) findViewById( R.id.out_3 ) ).getState(
) ) {
               commandPacket[ 1 ] |= OUT_3;
            }

            if( ( (LEDControl) findViewById( R.id.out_4 ) ).getState(
) ) {
               commandPacket[ 1 ] |= OUT_4;
            }

            if( ( (LEDControl) findViewById( R.id.out_5 ) ).getState(
) ) {
               commandPacket[ 1 ] |= OUT_5;
            }

               if ( ( (LEDControl) findViewById( R.id.out_6 )
).getState( ) ) {
```

```
                    commandPacket[ 1 ] |= OUT_6;
                    start_button_fl = true;
                 }

             if( ( (LEDControl) findViewById( R.id.out_7 ) ).getState(
) ) {
                 commandPacket[ 1 ] |= OUT_7;
             }

             // If start button pressed one time then keep it on
always.
             if ( start_button_fl )
                 commandPacket[ 1 ] |= OUT_6;

             accessoryManager.write( commandPacket );
             break;

          case USBAccessoryWhat:
             switch(((USBAccessoryManagerMessage)msg.obj).type) {
             case READ:
                 if(accessoryManager.isConnected() == false) {
                    return;
                 }

                 while(true) {
                    if(accessoryManager.available() < 2) {
                       //All of our commands in this example are 2
bytes.  If there are less
                       //  than 2 bytes left, it is a partial command
                       break;
                    }

                    accessoryManager.read( commandPacket );

                    switch(commandPacket[0]) {
                    case POT_STATUS_CHANGE:
                       ProgressBar progressBar = (ProgressBar)
findViewById(R.id.progress_bar);

                       if((commandPacket[1] >= 0) && (commandPacket[1]
<= progressBar.getMax())) {
                          progressBar.setProgress(commandPacket[1]);
                       }
                       break;
                    case PUSHBUTTON_STATUS_CHANGE:
                       if ( commandPacket[1] != -1 ) {
                          updateButton( R.id.battery_icon, ( (
commandPacket[ 1 ] & IN_1) == IN_1 ) ? true : false );
                          updateButton( R.id.oil_icon, ( (
commandPacket[ 1 ] & IN_2) == IN_2 ) ? true : false );
                          updateButton( R.id.temperature_icon, ( (
commandPacket[ 1 ] & IN_3) == IN_3 ) ? true : false );
                          updateButton( R.id.handbrake_icon, ( (
commandPacket[ 1 ] & IN_4) == IN_4 ) ? true : false );
                          updateButton( R.id.fuel_icon, ( (
commandPacket[ 1 ] & IN_5) == IN_5 ) ? true : false );
```

```
                        updateButton(
R.id.right_indicator_light_icon, ( ( commandPacket[ 1 ] & IN_6) ==
IN_6 ) ? true : false );
                        updateButton( R.id.left_indicator_light_icon,
( ( commandPacket[ 1 ] & IN_7) == IN_7 ) ? true : false );
                        updateButton( R.id.seat_belts_icon, ( (
commandPacket[ 1 ] & IN_8) == IN_8 ) ? true : false );
                     }

                     if ( commandPacket[ 1 ] == IN_9 )
                        updateButton( R.id.high_beam_headlights_icon,
true );
                     else
                        updateButton( R.id.high_beam_headlights_icon,
false );     '

                      break;
                  }

               }
               break;
            case CONNECTED:
                    break;
            case READY:

               setTitle( getString( R.string.connected ) );

               Log.d(TAG, "BasicAccessoryDemo:Handler:READY");

               LEDButtonEnable(true);

               String version =
((USBAccessoryManagerMessage)msg.obj).accessory.getVersion();
               firmwareProtocol = getFirmwareProtocol(version);

               switch(firmwareProtocol){
                  case 1:
                     deviceAttached = true;
                     break;
                  case 2:
                     deviceAttached = true;
                     commandPacket[0] = (byte) APP_CONNECT;
                     commandPacket[1] = 0;
                     Log.d(TAG,"sending connect message.");
                     accessoryManager.write(commandPacket);
                     Log.d(TAG,"connect message sent.");
                     break;
                  default:

showErrorPage(ErrorMessageCode.ERROR_FIRMWARE_PROTOCOL);
                     break;
               }
               break;
            case DISCONNECTED:
                    Log.d(TAG, "Disconnected");
               disconnectAccessory();
```

```
                break;
            }


                break;
        default:
            break;
        }  //switch
    } //handleMessage
  }; //handler
```

Η κλάση αυτή επίσης, περιλαμβάνει και διάφορες μεθόδους οι οποίες ενημερώνουν εικονικά τα αντίστοιχα εικονίδια στη οθόνη του τάμπλετ, όπως για την κατάσταση που βρίσκεται κάποια είσοδος του μικροελεγκτή ή ελέγχουν εάν υπάρχει σύνδεση μεταξύ του τάμπλετ και της συσκευής κ.α.

```
// Buttons update.
   private void updateButton( int id, boolean pressed ) {

   ImageView dashboard_icons = (ImageView) findViewById( id );

   if( pressed )
   {
      if ( dashboard_icons.getId( ) == R.id.battery_icon ) {

         dashboard_icons.setBackgroundColor( Color.parseColor(
"#ff0101" ) );

         if ( on_off_icons_fl[ BATTERY_ARRAY_POINTER ] )
            voiceMessages( R.raw.battery );

         on_off_icons_fl[ BATTERY_ARRAY_POINTER ] = false;
      }

      else if ( dashboard_icons.getId( ) == R.id.oil_icon ) {

         dashboard_icons.setBackgroundColor(Color.parseColor(
"#ff0101" ) );

         if ( on_off_icons_fl[ OIL_ARRAY_POINTER ] )
            voiceMessages( R.raw.oil );

         on_off_icons_fl[ OIL_ARRAY_POINTER ] = false;
      }

      else if ( dashboard_icons.getId( ) == R.id.temperature_icon ) {

         dashboard_icons.setBackgroundColor( Color.parseColor(
"#ff0101" ) );

         if ( on_off_icons_fl[ TEMPERATURE_ARRAY_POINTER ] )
```

```
            voiceMessages( R.raw.temperature );

         on_off_icons_fl[ TEMPERATURE_ARRAY_POINTER ] = false;
      }

      else if ( dashboard_icons.getId( ) == R.id.handbrake_icon ) {

         dashboard_icons.setBackgroundColor( Color.parseColor(
"#ff0101" ) );

         handbrake_fl = true;
      }

      else if ( dashboard_icons.getId( ) == R.id.fuel_icon ) {

dashboard_icons.setBackgroundColor(Color.parseColor("#ff6600"));

         if ( on_off_icons_fl[ FUEL_ARRAY_POINTER ] )
            voiceMessages( R.raw.fuel );

         on_off_icons_fl[ FUEL_ARRAY_POINTER ] = false;
      }

      else if ( dashboard_icons.getId( ) ==
R.id.right_indicator_light_icon ) {

dashboard_icons.setBackgroundColor(Color.parseColor("#006600"));
      }

      else if ( dashboard_icons.getId( ) ==
R.id.left_indicator_light_icon ) {

dashboard_icons.setBackgroundColor(Color.parseColor("#006600"));
      }

      else if ( dashboard_icons.getId( ) == R.id.seat_belts_icon ) {

dashboard_icons.setBackgroundColor(Color.parseColor("#ff0101"));

         seat_belts_fl = true;
      }

      else if ( dashboard_icons.getId( ) ==
R.id.high_beam_headlights_icon ) {

dashboard_icons.setBackgroundColor(Color.parseColor("#0000ff"));
      }
   }

   else {

      dashboard_icons.setBackgroundColor( Color.parseColor( "#595959"
) );
```

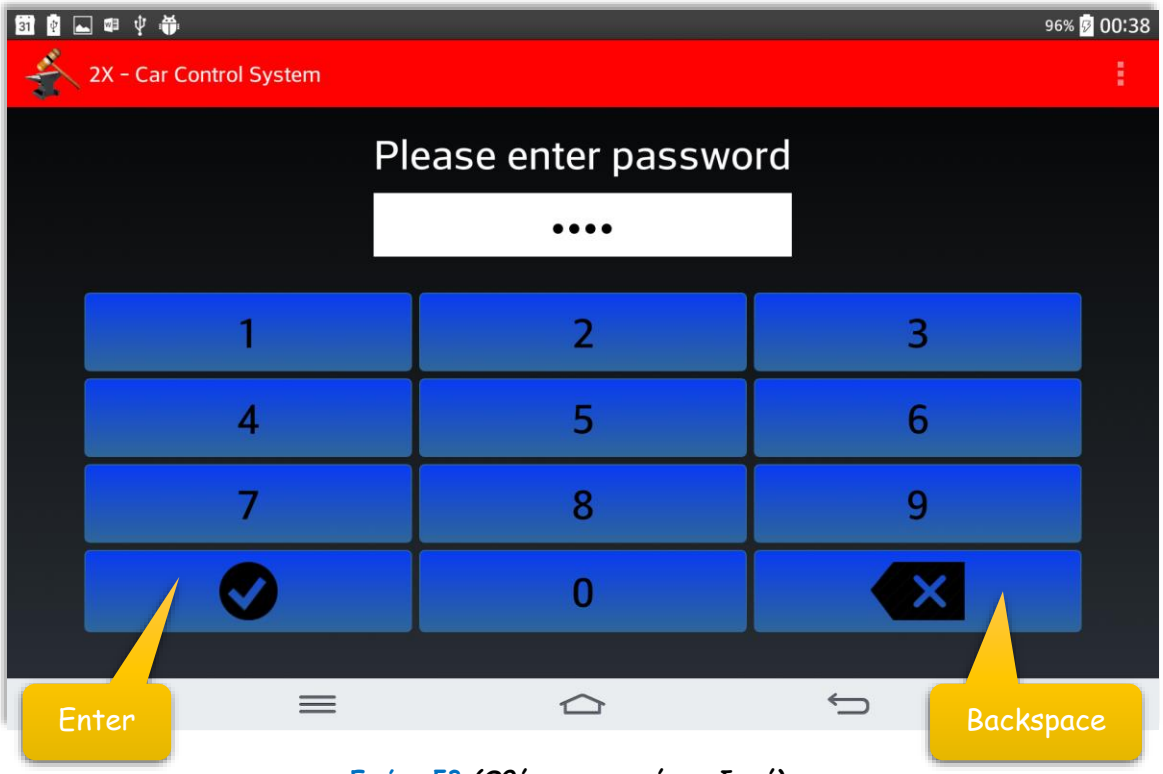
```
      if ( dashboard_icons.getId( ) == R.id.battery_icon )
         on_off_icons_fl[ BATTERY_ARRAY_POINTER ] = true;
      else if ( dashboard_icons.getId( ) == R.id.oil_icon )
         on_off_icons_fl[ OIL_ARRAY_POINTER ] = true;
      else if ( dashboard_icons.getId( ) == R.id.temperature_icon )
         on_off_icons_fl[ TEMPERATURE_ARRAY_POINTER ] = true;
      else if ( dashboard_icons.getId( ) == R.id.fuel_icon )
         on_off_icons_fl[ FUEL_ARRAY_POINTER ] = true;
      else if ( dashboard_icons.getId( ) == R.id.handbrake_icon )
         handbrake_fl = false;
      else if ( dashboard_icons.getId( ) == R.id.seat_belts_icon )
         seat_belts_fl = false;
   }

   }
```

```
   setContentView( R.layout.main );

       accessoryManager = new USBAccessoryManager( handler,
USBAccessoryWhat );

     try {
      //Set the link to the message handler for this class
       LEDControl ledControl;

ledControl = ( (LEDControl) findViewById( R.id.out_1 ) );
ledControl.setHandler( handler );

ledControl = ( (LEDControl) findViewById( R.id.out_2 ) );
ledControl.setHandler( handler );

ledControl = ( (LEDControl) findViewById( R.id.out_3 ) );
ledControl.setHandler( handler );

ledControl = ( (LEDControl) findViewById( R.id.out_4 ) );
ledControl.setHandler( handler );

ledControl = ( (LEDControl) findViewById( R.id.out_5 ) );
ledControl.setHandler( handler );

ledControl = ( (LEDControl) findViewById( R.id.out_6 ) );
ledControl.setHandler( handler );

ledControl = ( (LEDControl) findViewById ( R.id.out_7 ) );
ledControl.setHandler( handler );

     }

     catch ( Exception e ) {
     }
```

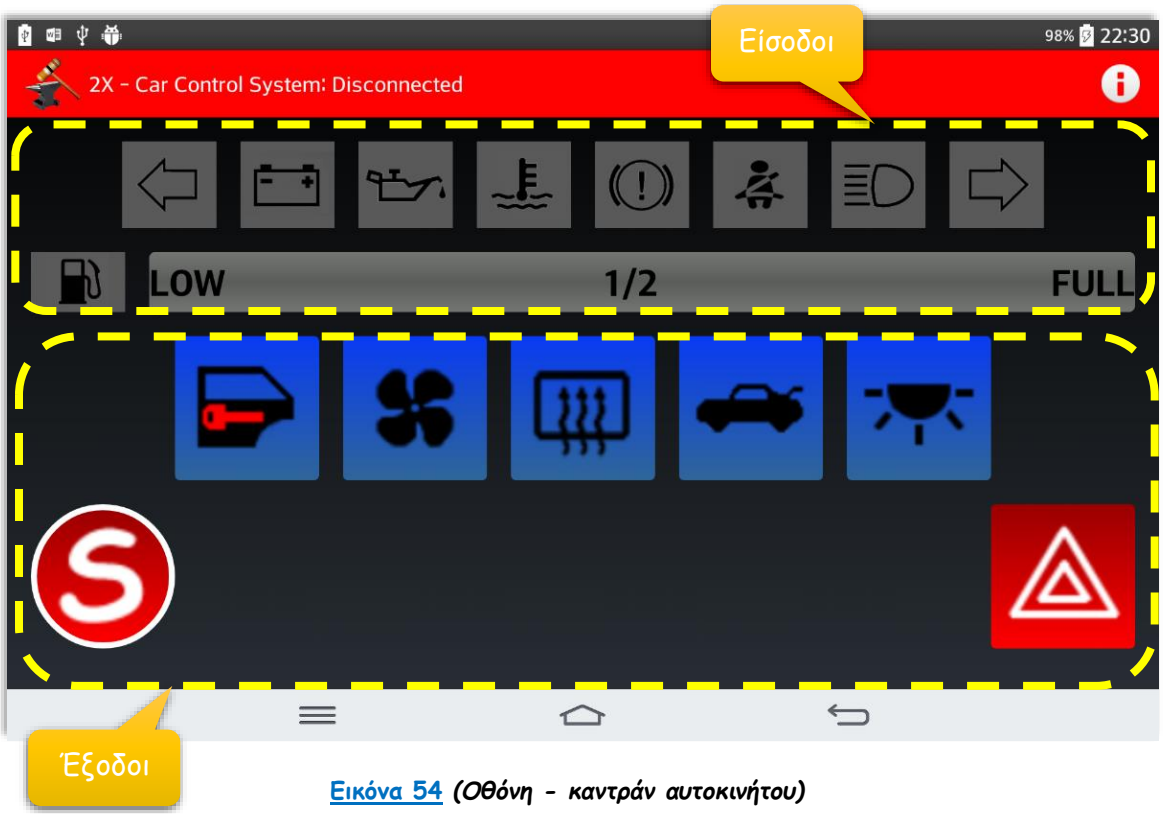
5 Παρουσίαση λειτουργικότητας του συστήματος

Με το που ο οδηγός γυρίσει στη θέση «ON» το διακόπτη εκκίνησης του αυτοκινήτου τότε αυτόματα το τάμπλετ ξεκλειδώνει (μόνο για την εμφάνιση της συγκεκριμένης εφαρμογής) και αρχικά εμφανίζεται η οθόνη υποδοχής (*εικόνα 52*) για περίπου δύο δευτερόλεπτα. Έπειτα εμφανίζεται μία οθόνη με ένα πληκτρολόγιο αριθμών (*εικόνα 53*) και ζητείται από το χρήστη (οδηγό του αυτοκινήτου) η εισαγωγή του τετραψήφιου κωδικού. Σε περίπτωση εισαγωγής λάθος κωδικού για τρείς συνεχόμενες φορές τότε ζητείται από το χρήστη η εισαγωγή ενός οκταψήφιου κωδικού ασφαλείας. Σε περίπτωση επιτυχημένης ή αποτυχημένης εισαγωγής του κωδικού αναπαράγονται και αντίστοιχα μηνύματα, όπως «Access Granted» «Access Denied- You two more attempts to insert a valid password» κ.α. Έπειτα από επιτυχημένη εισαγωγή του τετραψήφιου κωδικού εμφανίζεται η οθόνη (*εικόνα 54*) στην οποία υπάρχουν τα αντίστοιχα εικονίδια που αφορούν τις εισόδους (αισθητήρια-διακόπτες του αυτοκινήτου) και τις εξόδους του μικροελεγκτή. Σε περίπτωση ενεργοποίησης κάποιας ψηφιακής εισόδου τότε το αντίστοιχο εικονίδιο αλλάζει χρώμα (ανάλογα με την είσοδο στην οποία αντιστοιχεί γίνεται κόκκινο ή πράσινο ή μπλε ή πορτοκαλί). Επίσης, κατά την ενεργοποίηση κάποιων εισόδων, οι οποίες είναι κρίσιμες όσο αφορά τη λειτουργία του αυτοκινήτου, όπως π.χ. αυτή της θερμοκρασίας ακούγεται και ένα ηχητικό μήνυμα «Danger! Danger! Engine has overheated» με σκοπό την άμεση ενημέρωση του οδηγού. Η μπάρα, η οποία δείχνει τη στάθμη του καυσίμου είναι μία αναλογική είσοδος και αποτελείται από διαβαθμίσεις των χρωμάτων κόκκινου – πορτοκαλί – πράσινου. Τέλος, όταν ο χρήστης ενεργοποιήσει κάποια έξοδο τότε το αντίστοιχο εικονίδιο της γίνεται πράσινο.

Εικόνα 52 ***(Οθόνη υποδοχής)***

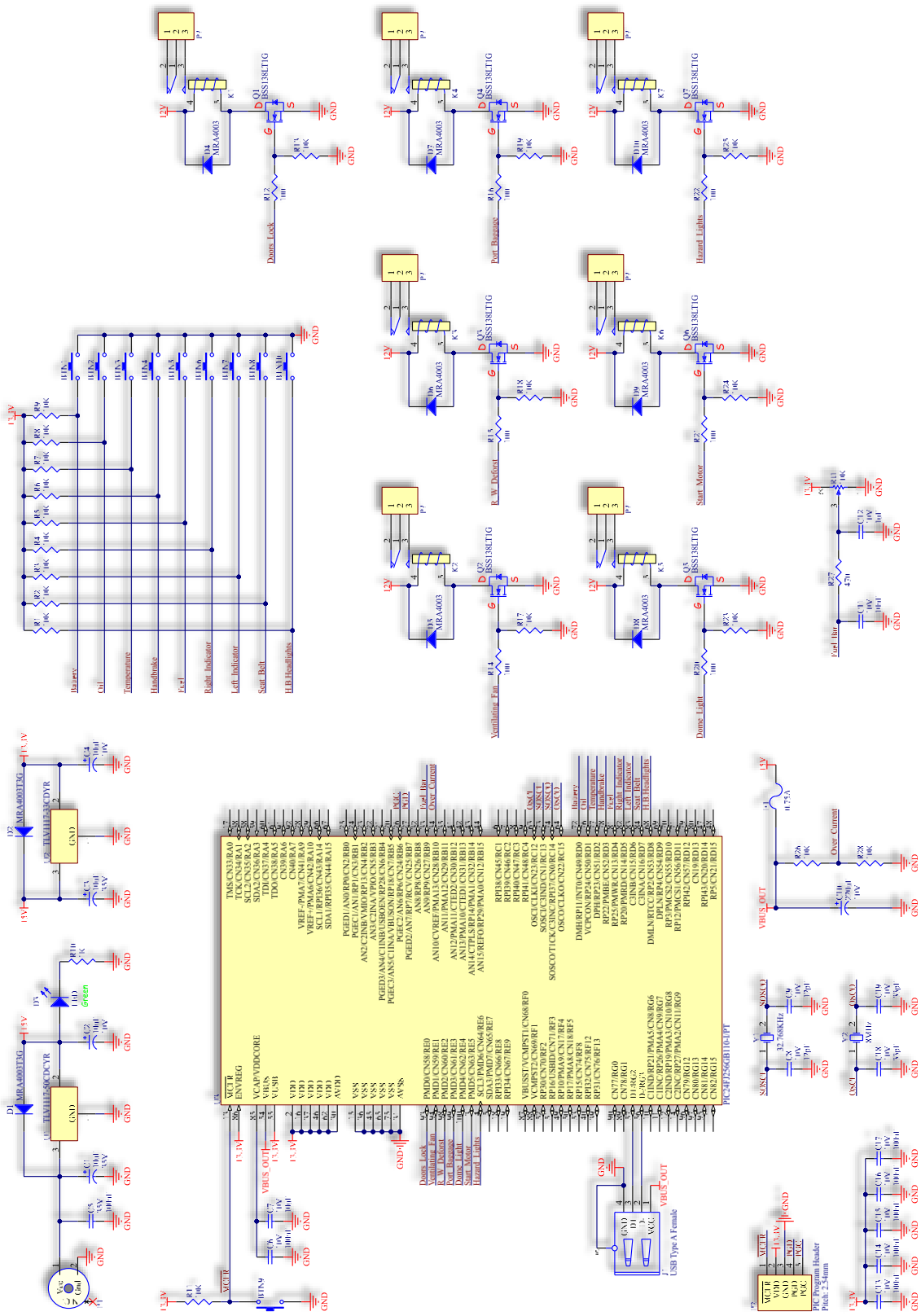


Εικόνα 53 *(Οθόνη εισαγωγής κωδικού)*

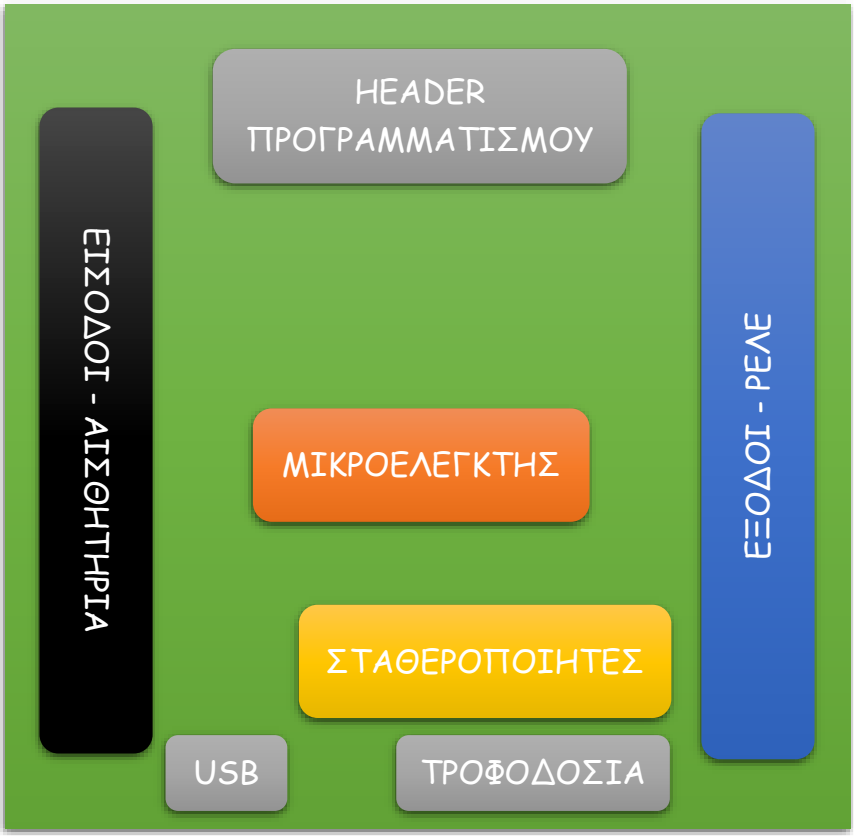


Εικόνα 54 *(Οθόνη - καντράν αυτοκινήτου)*

Εικόνα 55 *(Οθόνη - καντράν αυτοκινήτου)*



Εικόνα 56 (Σχηματικό πλακέτας)



Εικόνα 57 *(Μπλοκ διάγραμμα πλακέτας)*

6 Συμπεράσματα

Στην εργασία αυτή, υλοποιήθηκε ένα σύστημα, του οποίο σκοπός είναι ο έλεγχος κάποιων συγκεκριμένων λειτουργειών (εισόδων-αισθητηρίων και εξόδων-ρελέ) ενός οχήματος και πιο συγκεκριμένα ενός αυτοκινήτου. Ένα από τα βασικά πλεονεκτήματα αυτού του συστήματος είναι η οικονομία χώρου που προσφέρεται από τη συλλογή όλων των απεικονίσεων και διακοπτών ενεργοποίησης/απενεργοποίησης εξοπλισμού του αυτοκινήτου σε μία και μόνο οθόνη. Επίσης, ένα ακόμη πλεονέκτημα είναι τα ηχητικά μηνύματα, τα οποία ενημερώνουν άμεσα τον οδηγό για τυχόν βλάβη του αυτοκινήτου με σκοπό την αποφυγή οδικών ατυχημάτων. Αντιθέτως, το βασικό μειονέκτημα του συστήματος είναι ότι η τροποποίηση του, τόσο σε επίπεδο υλικού όσο και σε επίπεδο λογισμικού χρειάζεται γνώσεις ηλεκτρονικών και προγραμματισμού και συνεπώς, ο κάτοχος του αυτοκινήτου δεν μπορεί να απευθυνθεί σε ένα οποιοδήποτε συνεργείο για την τροποποίηση του συστήματος. Εξαιτίας αυτού του μειονεκτήματος θα μπορούσαν να γίνουν κάποιες τροποποιήσεις και βελτιώσεις όσο αφορά τη λειτουργικότητα του συστήματος, ώστε ο εκάστοτε οδηγός να επιλέγει το αριθμό των εισόδων και εξόδων που θα χρησιμοποιήσει, καθώς και να μπορεί με εύκολο σχετικά τρόπο να τροποποιεί και την γραφική εμφάνιση των εικονιδίων, αλλά και της εφαρμογής γενικά. Επίσης, σε μία νέα έκδοση του συστήματος θα μπορούσε να γίνεται αποστολή μηνύματος «SMS» στον κάτοχο του αυτοκίνητου με τη τοποθεσία του αυτοκινήτου (τις x-y συντεταγμένες) σε περίπτωση ενεργοποίησης του συναγερμού. Μια ακόμα βελτίωση που θα μπορούσε να γίνει είναι η σύνδεση κάμερας οπισθοπορείας στο σύστημα με σκοπό την αυτόματη ενεργοποίησης της σε περίπτωση που ο οδηγός βάλει την όπισθεν ταχύτητα.

Το σύστημα αυτό θα μπορούσε να χρησιμοποιηθεί από κατασκευάστριες εταιρίες αυτοκινήτων ή από κατόχους αυτοκινήτων, οι οποίοι θέλουν κάτι πιο τεχνολογικά εξελιγμένο και καινούργιο για το αυτοκίνητο τους. Η εγκατάσταση τους συστήματος όσο αφορά της σύνδεση της πλακέτας με τα αισθητήρια του αυτοκινήτου, καθώς και τη σύνδεση των ρελέ της πλακέτας στα εξαρτήματα του αυτοκινήτου είναι σχετικά απλή και μπορεί να γίνει σε οποιοδήποτε ηλεκτρολογείο αυτοκινήτων. Επίσης, το ηλεκτρονικό κύκλωμα (πλακέτα) του συστήματος είναι έτσι σχεδιασμένο, ώστε να υπάρχει συμβατότητα με τους περισσότερους αισθητήρες του εκάστοτε αυτοκινήτου.

Η υλοποίηση του συστήματος έγινε στη διάρκεια έξι μηνών, όπου το μεγαλύτερο μέρος του χρόνου αφιερώθηκε στην εφαρμογή Android. Η ανάπτυξη της υλικού και του λογισμικού μέρους έγιναν παράλληλα, επειδή υπάρχει πλήρης αλληλεπίδραση μεταξύ των δύο κατηγοριών.

Βιβλιογραφία – Πηγές Διαδικτύου

Google self-driving car
https://en.wikipedia.org/wiki/Google_self-driving_car

Android Auto
https://en.wikipedia.org/wiki/Android_Auto

Android (operating system)
https://en.wikipedia.org/wiki/Android_(operating_system)

Google Play
https://en.wikipedia.org/wiki/Google_Play

Android Developers - Activity
http://developer.android.com/reference/android/app/Activity.html

Android version history
https://en.wikipedia.org/wiki/Android_version_history

Android Marshmallow
https://en.wikipedia.org/wiki/Android_Marshmallow

Android Developers - Device Administration
http://developer.android.com/guide/topics/admin/device-admin.html

Android Developers - USB Host and Accessory
http://developer.android.com/guide/topics/connectivity/usb/index.html