ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
ΤΜΗΜΑ ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ

ΠΡΟΓΡΑΜΜΑ ΜΕΤΑΠΤΥΧΙΑΚΩΝ ΣΠΟΥΔΩΝ
«ΨΗΦΙΑΚΕΣ ΕΠΙΚΟΙΝΩΝΙΕΣ ΚΑΙ ΔΙΚΤΥΑ»

«ΑΝΑΠΤΥΞΗ ΔΙΑΔΙΚΤΥΑΚΟΥ ΤΡΟΧΗΛΑΤΟΥ ΡΟΜΠΟΤΙΚΟΥ ΣΥΣΤΗΜΑΤΟΣ ΜΕ ΕΜΦΑΣΗ ΣΤΟΝ ΑΠΟΜΑΚΡΥΣΜΕΝΟ ΡΟΜΠΟΤΙΚΟ ΕΛΕΓΧΟ»

ΠΕΤΡΟΣ ΧΡΥΣΙΚΟΠΟΥΛΟΣ: ΜΨΕ 1619

Επιβλέπων Καθηγητής : Απόστολος Μηλιώνης

Διπλωματική Εργασία υποβληθείσα στο Τμήμα Ψηφιακών Συστημάτων του Πανεπιστημίου Πειραιώς ως μέρους των απαιτήσεων για την απόκτηση Μεταπτυχιακού Διπλώματος Ειδίκευσης στις Ψηφιακές Επικοινωνίες και Δίκτυα

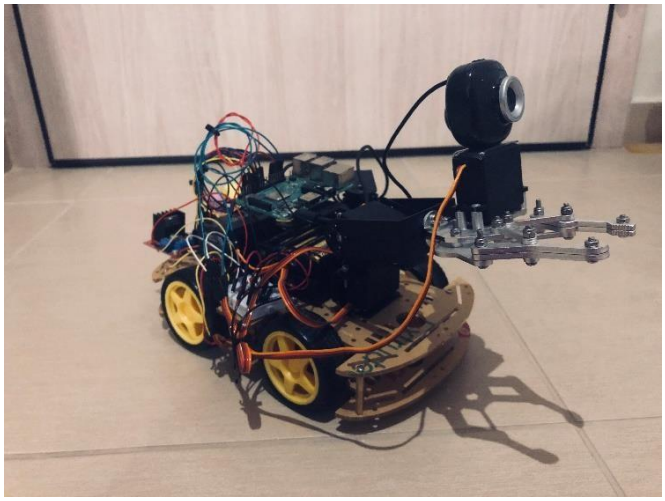
Πειραιάς, Σεπτέμβριος 2018

Περίληψη

Στην παρούσα διπλωματική εργασία σχεδιάστηκε και υλοποιήθηκε εφαρμογή για απομακρυσμένο έλεγχο ρομποτικού οχήματος με βραχίονα το οποίο βασίζεται στο **raspberry pi 3 b+** και κατασκευάστηκε από τον συμφοιτητή μου Ιωάννη Βώσικα. Σκοπός της συγκεκριμένης διπλωματικής εργασίας είναι η σχεδίαση ενός μηχανισμού, με τη λειτουργία του οποίου το όχημα θα μπορεί να κινείται, να συλλέγει αντικείμενα και να μας αποστέλλει εικόνα από το χώρο στον οποίο βρίσκεται.

Για την επίτευξη αυτού του σκοπού επιλέχθηκαν οι κατάλληλες γλώσσες προγραμματισμού και η συγκεκριμένη διανομή Linux για υπολογιστή raspberry ώστε να μπορεί να χρησιμοποιηθεί η εφαρμογή και το ρομποτικό όχημα ακόμα και από αρχάριους χρήστες.

Αφού διαλέξαμε το κατάλληλο για την εργασία μας υλικό, προχωρήσαμε στη υλοποίηση του λογισμικού μας και έτσι προγραμματίσαμε όλες τις απαραίτητες ενέργειες, οι οποίες χρειάζονται τόσο για την κίνηση όσο και για την ασύρματη επικοινωνία και το streaming.



1 Το όχημα μας α.

ΠΕΡΙΕΧΟΜΕΝΑ

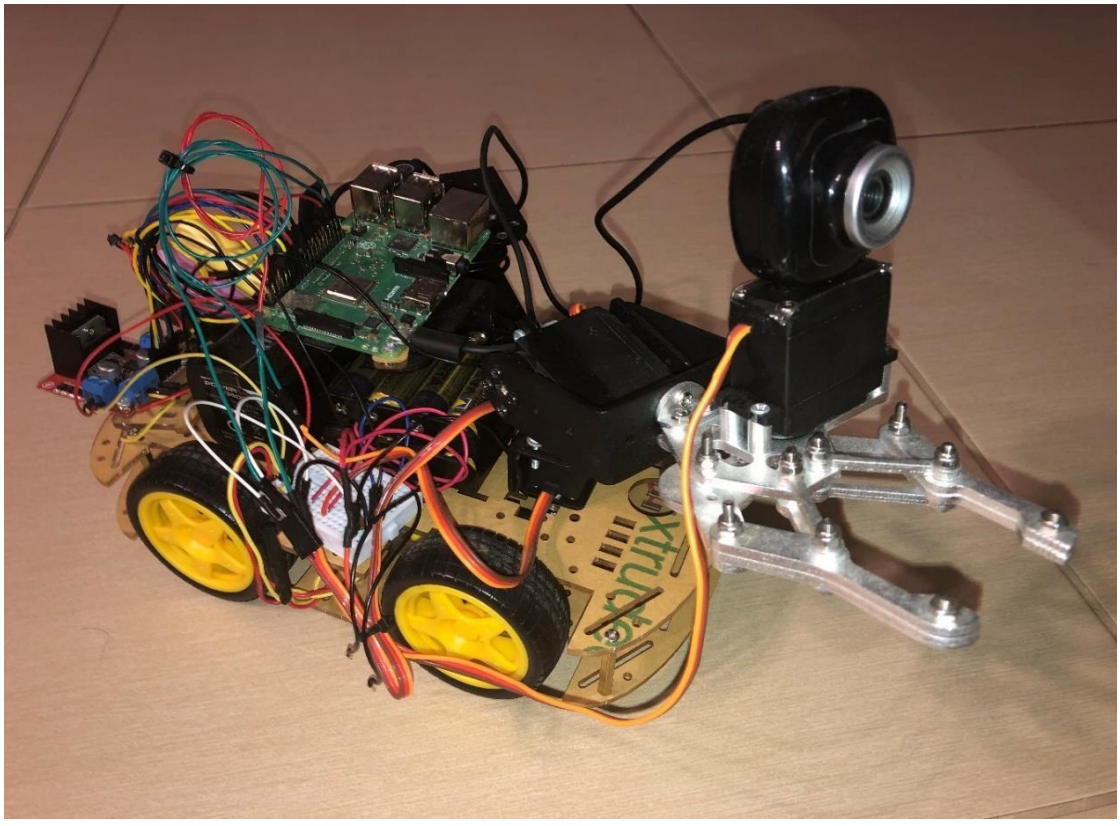
ΚΑΤΑΛΟΓΟΣ ΕΙΚΟΝΩΝ

1. ΕΙΣΑΓΩΓΗ

Στην παρούσα διπλωματική εργασία σχεδιάστηκε και υλοποιήθηκε εφαρμογή μέσω της οποίας θα γίνεται εξ ολοκλήρου έλεγχος ρομποτικού οχήματος με βραχίονα που βασίζεται στο raspberry pi 3 b+.

Το όχημα μας συνδέεται απομακρυσμένα μέσω Wi-Fi με κινητή συσκευή. Η εφαρμογή αυτή μπορεί να ελέγχει και να κινεί το όχημα μας και τον βραχίονα που φέρει αυτό, αλλά και να συλλέγει αντικείμενα από το χώρο που βρίσκεται. Παράλληλα μας δίνει και εικόνα μέσω κάμερας που εχει εγκατασταθεί πάνω στο βραχίονα.

Σκοπός της εφαρμογής αλλα και ολόκληρου του οχήματος είναι ο απομακρυσμένος έλεγχος από άτομα με κινητικά προβλήματα ώστε να τους παραχθεί βοήθεια.



2 Το όχημα μας β

Τι είναι η γλώσσα προγραμματισμού

Γλώσσα προγραμματισμού ονομάζεται η τεχνητή γλώσσα που μπορεί να χρησιμοποιηθεί για τον έλεγχο μιας μηχανής, συνήθως ενός υπολογιστή. Οι γλώσσες προγραμματισμού (όπως άλλωστε και οι ανθρώπινες γλώσσες) ορίζονται από ένα σύνολο συντακτικών και εννοιολογικών κανόνων, που ορίζουν τη δομή και το νόημα, αντίστοιχα, των προτάσεων της γλώσσας.

Οι γλώσσες προγραμματισμού χρησιμοποιούνται για να διευκολύνουν την οργάνωση και διαχείριση πληροφοριών, αλλά και για την ακριβή διατύπωση αλγορίθμων. Ορισμένοι ειδικοί χρησιμοποιούν τον όρο γλώσσα προγραμματισμού μόνο για τυπικές γλώσσες που μπορούν να εκφράσουν όλους τους πιθανούς αλγορίθμους. Μη-υπολογιστικές γλώσσες όπως η HTML ή τυπικές γραμματικές όπως η BNF δεν λέγονται συνήθως γλώσσες προγραμματισμού.

Υπάρχουν χιλιάδες διαφορετικές γλώσσες προγραμματισμού, και κάθε χρόνο δημιουργούνται περισσότερες.

3 Γλώσσες προγραμματισμού

Ιστορία γλωσσών προγραμματισμού

Ο ψηφιακός υπολογιστής μπορεί να χρησιμοποιήσει μόνο δυαδικούς αριθμούς και να εκτελέσει διαδοχικά εντολές που του δίνονται κι αυτές με μορφή δυαδικού αριθμού. Τα προγράμματα που φτιάχνονται με τέτοιες εντολές λέμε πως φτιάχνονται σε γλώσσα μηχανής. Όταν, για παράδειγμα, θέλουμε να αυξήσουμε το περιεχόμενο κάποιου καταμετρητή Ν κατά 2, δίνουμε εντολές σε γλώσσα μηχανής, που μοιάζουν κάπως έτσι:

00000100000000001100101011 00100000000000000000010000
00001000000000001100101011

Αυτές οι μεγάλες σειρές από 0 και 1 ήταν κουραστικές για τον άνθρωπο. Θα βελτιωνόταν κάπως η κατάσταση, αν γράφονταν οι εντολές αυτές με οκταδικούς αριθμούς:

01001453 10000020 02001453

Έτσι οι εντολές διαβάζονταν λίγο πιο εύκολα, πάλι όμως δεν ήταν απλό να δει κανείς αμέσως ποια δουλειά έκαναν αυτές οι εντολές. Επίσης, αν ήθελαν οι προγραμματιστές να κάνουν διορθώσεις, προσθήκες και διαγραφές εντολών σε πρόγραμμα γραμμένο σε γλώσσα μηχανής, αντιμετώπιζαν τεράστιες δυσκολίες σε μια διαδικασία πολύ ευάλωτη από λάθη. Επινοήθηκε λοιπόν μια συμβολική γλώσσα για τις εντολές που καταλάβαινε ο υπολογιστής και γράφτηκε ένα συμβολομεταφραστικό πρόγραμμα (assembler), που μετέτρεπε ένα πρόγραμμα συμβολικής γλώσσας σε ένα πρόγραμμα σε γλώσσα μηχανής.

Το προηγούμενο παράδειγμα θα έμοιαζε σε μια υποθετική συμβολική γλώσσα κάπως έτσι:

```
LDA N ;N ΕΙΝΑΙ Ο ΜΕΤΡΗΤΗΣ

ADD +2 ;ΑΥΞΑΝΕΤΑΙ ΚΑΤΑ 2 Ο Ν

STA N ;ΑΠΟΘΗΚΕΥΕΤΑΙ Η ΝΕΑ ΤΙΜΗ ΣΤΟ Ν
```

Εδώ, εκτός από τις εντολές προγράμματος στο αριστερό μέρος, υπάρχουν και μερικά επεξηγηματικά σχόλια, (που ο συμβολομεταφραστής τα αγνοεί), που βοηθάνε πολύ στην φάση συντήρησης του προγράμματος.

Παρ' όλα αυτά όμως, επειδή τα προγράμματα δεύτερης γενιάς (αυτά σε συμβολική γλώσσα) είχαν ακριβώς τόσες εντολές όσες είχαν και τα αντίστοιχα προγράμματα σε γλώσσα μηχανής, η ανάπτυξη μεγάλων προγραμμάτων ήταν δύσκολη. Έτσι φτιάχτηκαν οι γλώσσες τρίτης γενιάς, όπως είναι η FORTRAN, όπου ενώ οι προγραμματιστές έδιναν μόνο μια εντολή, (το Ν να γίνει όσο ήταν συν 2), αναλάμβανε ένα πρόγραμμα - μεταφραστής (compiler) να την αναλύσει σε περισσότερες εντολές σε συμβολική γλώσσα (και σε γλώσσα μηχανής):

```
N = N + 2
```

Εκτός από το ότι έτσι γράφονταν λιγότερες εντολές για το ίδιο πρόγραμμα, δημιουργήθηκαν και γλώσσες ανώτερου επίπεδου που έμοιαζαν περισσότερο με φυσική γλώσσα. Το ίδιο παράδειγμα σε γλώσσα COBOL θα μπορούσε να γραφτεί έτσι:

```
ADD 2 TO COUNTER–N .
```

Το ίδιο παράδειγμα σε γλώσσα C θα μπορούσε να γραφτεί έτσι:

```
N += 2;
```

Στην εκπαιδευτική γλώσσα προγραμματισμού "ΓΛΩΣΣΑ", που διαθέτει τις εντελώς απαραίτητες εντολές για εξάσκηση στον προγραμματισμό, η εντολή εκχώρησης τιμής γράφεται έτσι:

N <- N + 2

Σημαίνει «κάνε τις πράξεις στην έκφραση στο δεξιό από το βελάκι μέρος και βάλε την προκύπτουσα τιμή στην μεταβλητή που βρίσκεται στο αριστερό μέρος».

Κατηγοριοποίηση γλωσσών προγραμματισμού

Δεν υπάρχει απλός τρόπος να κατηγοριοποιηθούν οι γλώσσες προγραμματισμού. Αυτό συμβαίνει γιατί συνήθως κάθε γλώσσα προγραμματισμού περιέχει επιρροές από πολλές προηγούμενες γλώσσες, συνδυάζοντας θετικά στοιχεία και προσθέτοντας νέα. Χαρακτηριστικά που εμφανίζονται σε μια γλώσσα και έχουν θετική αποδοχή, συνήθως υιοθετούνται από μεταγενέστερες γλώσσες ακόμα και αν πρόκειται για γλώσσες που ανήκουν σε διαφορετική κατηγορία.

Η κατηγοριοποίηση είναι ακόμα πιο περίπλοκη για το λόγο ότι πολλές γλώσσες συνήθως ανήκουν σε παραπάνω από μία κατηγορίες. Για παράδειγμα, η Java είναι τόσο αντικειμενοστρεφής όσο και παράλληλη γλώσσα, δεδομένου ότι υποστηρίζει την οργάνωση των δεδομένων και υπολογισμών σε αντικείμενα, αλλά επιτρέπει επίσης και την δημιουργία προγραμμάτων με ταυτόχρονα νήματα (threads) που εκτελούνται παράλληλα.

Δεδομένης της δυσκολίας στην κατηγοριοποίηση, μπορούμε να κατηγοριοποιήσουμε τις γλώσσες προγραμματισμού με διάφορους τρόπους. Οι συνηθέστεροι τρόποι είναι:

- με βάση τον τρόπο οργάνωσης του προγράμματος
- με βάση τον στόχο που έχει η γλώσσα
- με βάση τον τρόπο που περιγράφουν το ζητούμενο αποτέλεσμα

Στην πρώτη περίπτωση προκύπτουν κατηγορίες όπως:

- **Διαδικαστικές γλώσσες** (procedural) όπου το πρόγραμμα είναι οργανωμένο σε διαδικασίες, που αποτελούνται από σειρές εντολών που περιγράφουν αλγορίθμους. Παραδείγματα γλωσσών που ανήκουν σε αυτή την κατηγορία είναι η Pascal ή η C.
- **Αντικειμενοστρεφείς γλώσσες** (object-oriented) όπου το πρόγραμμα είναι οργανωμένο σε αντικείμενα. Ένα αντικείμενο είναι μια μονάδα που αποτελείται από την περιγραφή κάποιων δεδομένων και την περιγραφή των αλγορίθμων που τα επεξεργάζονται. Ένα αντικειμενοστρεφές πρόγραμμα

αποτελείται από διάφορα αντικείμενα που αλληλεπιδρούν μεταξύ τους. Παραδείγματα αντικειμενοστρεφών γλωσσών είναι η Java ή η C++.

- **Συναρτησιακές γλώσσες** (functional) όπου οι υπολογισμοί εκφράζονται ως εφαρμογές μαθηματικών συναρτήσεων, σε αντίθεση με τα άλλα είδη προγραμματισμού όπου οι υπολογισμοί εκφράζονται ως σειρές εντολών, όπου η κάθε μία αλλάζει με κάποιο τρόπο την κατάσταση του συστήματος. Θεωρητικό τους υπόβαθρο είναι ο λ-λογισμός. Χαρακτηριστικές συναρτησιακές γλώσσες είναι η Lisp, η Haskell και η OCaml.

Στην περίπτωση που η κατηγοριοποίηση των γλωσσών προγραμματισμού γίνει με βάση το στόχο που έχει η γλώσσα, υπάρχουν οι παρακάτω κατηγορίες:

- *Γλώσσες γενικής χρήσης*. Σε αυτήν την κατηγορία ταξινομούνται γλώσσες που δημιουργήθηκαν για τον προγραμματισμό γενικών εφαρμογών, καθώς και πολλές εκπαιδευτικές γλώσσες που αποδείχτηκαν χρήσιμες για την ανάπτυξη γενικών εφαρμογών, όπως η Pascal.
- *Γλώσσες προγραμματισμού συστημάτων*, που χρησιμοποιούνται συνήθως για τον προγραμματισμό λειτουργικών συστημάτων ή οδηγών (drivers) υλικού, όπου χρειάζεται πολλές φορές ο προγραμματιστής να έχει έλεγχο και γνώση του πως λειτουργεί το υλικό. Η πιο συχνά χρησιμοποιούμενη γλώσσα προγραμματισμού συστημάτων είναι η C.
- *Γλώσσες σεναρίων* (scripting). Αυτές οι γλώσσες χρησιμοποιούνται συνήθως για τη γρήγορη ανάπτυξη μικρών προγραμμάτων, σε περιπτώσεις που ο χρόνος του προγραμματιστή είναι πιο πολύτιμος από την ταχύτητα εκτέλεσης του προγράμματος, όπως για παράδειγμα συμβαίνει όταν το πρόγραμμα απλά αυτοματοποιεί απλές λειτουργίες. Παραδείγματα γλωσσών σεναρίων (scripting) είναι η Perl, η Python, η Ruby ή τα κελύφη του λειτουργικού συστήματος Unix (shells).
- *Γλώσσες ειδικών εφαρμογών*. Σε αυτή την κατηγορία ανήκουν γλώσσες που αναπτύχθηκαν ειδικά για μια συγκεκριμένη εφαρμογή. Για παράδειγμα, η γλώσσα PostScript είναι σχεδιασμένη ειδικά για να περιγράφονται με

λεπτομέρεια κείμενα προς εκτύπωση, ενώ η γλώσσα Matlab είναι σχεδιασμένη για την επεξεργασία πινάκων από αριθμητικά δεδομένα.

- *Παράλληλες ή κατανεμημένες γλώσσες.* Στη συγκεκριμένη κατηγορία ταξινομούνται γλώσσες που επιτρέπουν τη ανάπτυξη παράλληλων προγραμμάτων, όπου πολλές εντολές εκτελούνται ταυτόχρονα σε πολλούς υπολογιστές, έτσι ώστε το τελικό αποτέλεσμα να προκύψει γρηγορότερα. Οι παράλληλες γλώσσες προσφέρουν συνήθως εύκολους τρόπους επικοινωνίας μεταξύ των νημάτων που εκτελούνται παράλληλα, καθώς και τρόπους ώστε να δημιουργούνται καινούριες παράλληλες εκτελέσεις. Παραδείγματα γλωσσών που ανήκουν (και) σε αυτή την κατηγορία είναι η Go, η Java, η Erlang, η MultiLisp ή η Cilk.
- *Εκπαιδευτικά προγραμματιστικά περιβάλλοντα*. Σε αυτή την κατηγορία ανήκουν εκπαιδευτικές; γλώσσες προγραμματισμού οι οποίες απευθύνονται σε αρχάριους προγραμματιστές για την κατασκευή μικροεφαρμογών. Είναι κατάλληλες για την εκμάθηση προγραμματισμού σε μικρές ηλικίες. Παραδείγματα τέτοιων γλωσσών είναι η LOGO, το Game Maker και το App Inventor.

Τέλος, στην περίπτωση που η κατηγοριοποίηση γίνεται με βάση τον τρόπο που περιγράφεται το ζητούμενο, υπάρχουν οι παρακάτω κατηγορίες:

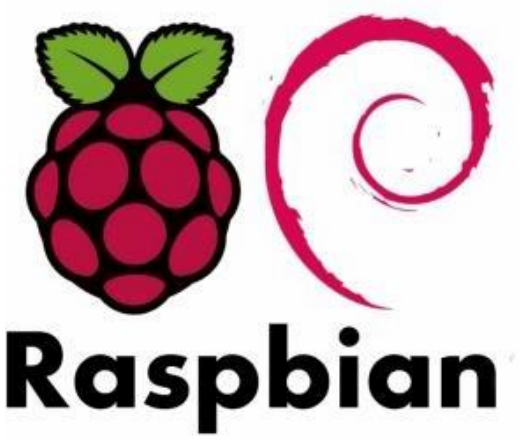
- Προστακτικές γλώσσες προγραμματισμού (imperative) είναι οι γλώσσες που περιγράφουν το ζητούμενο αποτέλεσμα κατασκευαστικά, δίνοντας μια σειρά εντολών που όταν εκτελεστούν παράγουν το ζητούμενο αποτέλεσμα. Τέτοιες γλώσσες είναι η C, η Java αλλά και η OCaml.
- Δηλωτικές γλώσσες προγραμματισμού (declarative) είναι οι γλώσσες που περιγράφουν το ζητούμενο αποτέλεσμα χρησιμοποιώντας τις ιδιότητες που έχει, και όχι τον τρόπο με τον οποίο υπολογίζεται. Παραδείγματα δηλωτικών γλωσσών είναι η Haskell, η SQL και η Prolog.

Γλώσσες προγραμματισμού

- Ada
- Algol
- Applescript
- AWK
- BASIC
- C
- C++
- C#
- Cilk
- Clojure
- COBOL
- Datalog
- Erlang
- Forth
- FORTRAN
- Haskell
- Java
- JavaScript
- Lisp
- Logo
- Lua
- Lucid
- Mathematica
- Matlab
- Miranda
- ML
- OBJ / Σύστημα Maude
- Objective-C
- OCaml
- Pascal
- Perl
- PHP
- Prolog
- Python
- Ruby
- Scala
- Scheme
- Simula
- Smalltalk
- SQL
- Tcl
- Visual Basic
- ΓΛΩΣΣΑ

2. ΛΟΓΙΣΜΙΚΟ

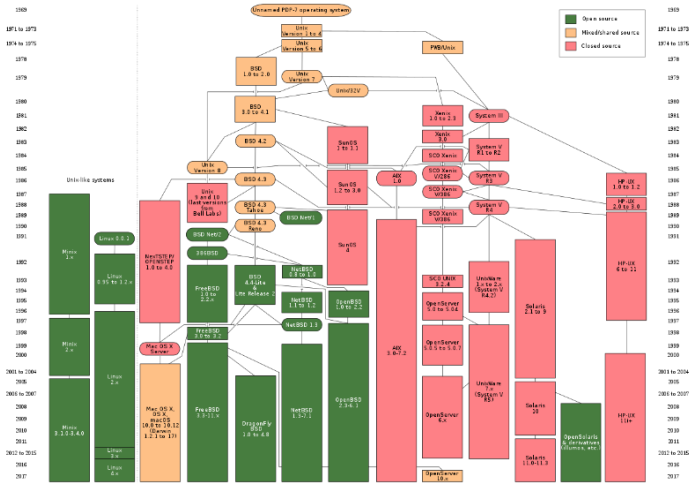
Raspbian computer operating system



4 Λογότυπο Raspbian

Το Raspberry Pi χρησιμοποιεί κυρίως λειτουργικά συστήματα βασισμένα σε Linux-kernel (πυρήνας Linux). Το Linux όπως είναι προφανές είναι βασισμένο σε Linux-kernel, το οποίο είναι ένα λειτουργικό σύστημα το οποίο συμπεριφέρεται όπως ένα Unix σύστημα, χωρίς απαραίτητα να είναι.

Δεν υπάρχει κάποιο πρότυπο που να ορίζει τον όρο Unix-like και γι' αυτό υπάρχουν αντικρουόμενες απόψεις επί του θέματος. Ο όρος μπορεί να αναφέρεται σε ελεύθερα και ανοιχτά λειτουργικά συστήματα που έχουν εμπνευστεί απ' τα unix ή μπορεί να αναφέρεται σε συστήματα που έχουν σχεδιαστεί για να προσομοιώνουν τις λειτουργίες ενός unix.



5 Εξέλιξη των Unix συστημάτων

Το Linux- kernel χρησιμοποιείται τόσο σε παραδοσιακά συστήματα υπολογιστών, όπως ο προσωπικός υπολογιστής και οι διάφοροι servers, αλλά και σε ενσωματωμένα συστήματα, όπως routers, έξυπνες τηλεοράσεις κ.α. Επίσης, το λογισμικό Android που χρησιμοποιούν τα περισσότερα κινητά τηλέφωνα και τάμπλετ είναι επίσης βασισμένο σε Linux-kernel. Αναπτύχθηκε το 1991 απ' τον Φιλανδό φοιτητή Linus Torvalds, για προσωπική χρήση αλλά έκτοτε έχει αναπτυχθεί από περισσότερους από 12.000 προγραμματιστές για πάνω από 1.000 εταιρείες.

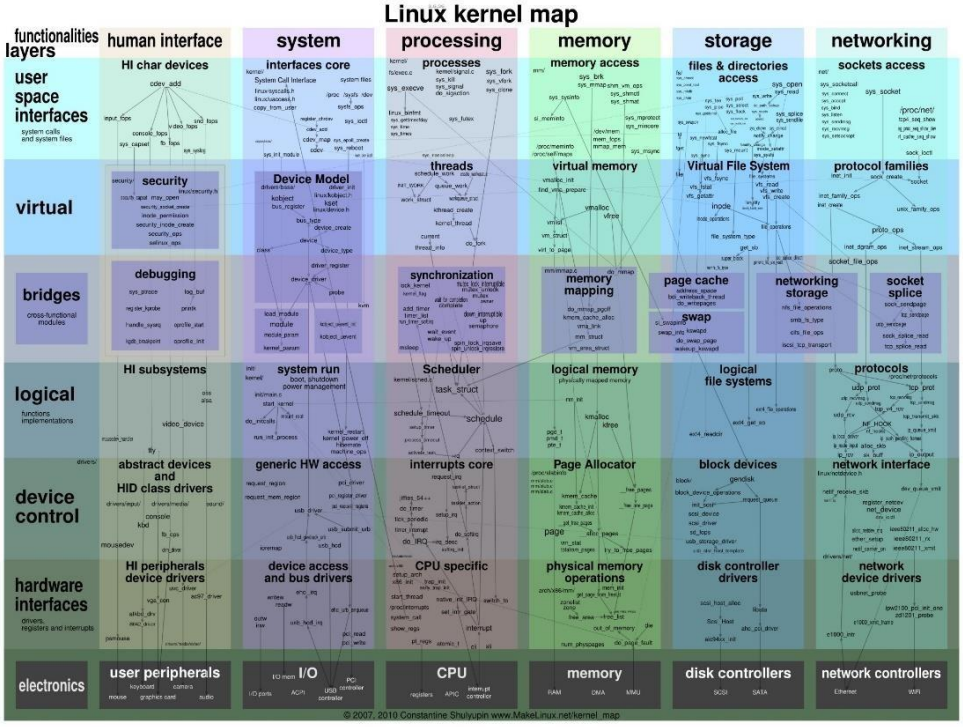
Η διεπαφή για τον προγραμματισμό εφαρμογών (API) με την οποία ο χρήστης επιδρά με το πυρήνα, θεωρείται πολύ σταθερή. Βέβαια ως μειονέκτημα καταγράφεται το γεγονός, ότι η διεπαφή μεταξύ του kernel και των loadable kernel modules, δηλαδή των αρχείων που περιέχουν κώδικα για να επεκτείνουν το ήδη υπάρχον kernel, δεν είναι σταθερή από τον σχεδιασμό του. Το μεγάλο πλεονέκτημά του είναι ότι είναι ελεύθερο και open source, δηλαδή μπορεί οποιοσδήποτε να συμβάλλει και να αναπτύξει εφαρμογές για το Linux. Αξίζει να σημειώσουμε ότι το συγκεκριμένο kernel έχει γραφτεί σε γλώσσα προγραμματισμού C, ενώ υπάρχουν μικρές περιοχές στον κώδικα που έχουν γραφτεί σε γλώσσα assembly.

Το Linux-kernel είναι ένας μονολιθικός πυρήνας, ο οποίος υποστηρίζει ταυτόχρονη εκτέλεση πολλών διεργασιών (τόσο σε επίπεδο χρήστη, όσο και σε επίπεδο πυρήνα),

εικονική μνήμη, κοινές βιβλιοθήκες, διαχείριση μνήμης, σουίτα πρωτοκόλλου ίντερνετ και νήματα. Οι Drivers για τις συσκευές και οι επεκτάσεις του πυρήνα τρέχουν στον χώρου του πυρήνα (χώρος 0 για τις περισσότερες αρχιτεκτονικές επεξεργαστών) με πλήρη πρόσβαση στο υλικό, αν και υπάρχουν κάποιες εξαιρέσεις για τον χώρο του χρήστη.

Το σύστημα γραφικών που οι περισσότεροι έχουν μαζί με τα Linux δεν τρέχει μέσα στον πυρήνα εν αντιθέσει με τα Microsoft Windows. Αντίθετα με τα συνηθισμένα μονολιθικά κέρνελ οι drivers για τις συσκευές είναι εύκολο να διαμορφωθούν σαν ενότητες και να χρησιμοποιηθούν, ενώ το σύστημα τρέχει. Επίσης, από επιλογή το Linux kernel δεν έχει δυαδική διεπαφή πυρήνα.

Ενώ αρχικά δεν είχε κατασκευαστεί για να είναι φορητό, τώρα το Linux είναι ένα απ' τα πιο διαδεδομένα λειτουργικά συστήματα που τρέχουν σε ένα μεγάλο εύρος συστημάτων από αρχιτεκτονικές ARM μέχρι Z/αρχιτεκτονικές.



6 Χάρτης του Linux Kernel

Από το Pi 2 και μετά μπορεί να τρέξει και Windows 10 IoT Core, ενώ υποστηρίζει και RISC OS, το οποίο είναι το λειτουργικό σύστημα του πρώτου υπολογιστή βασισμένου σε ARM. Το πιο συνηθισμένο λειτουργικό σύστημα και αυτό που θα χρησιμοποιήσουμε και εμείς είναι το Raspbian, το οποίο είναι βασισμένο σε Debian (αυτήν την στιγμή έχουμε φτάσει στην έκδοση Jessie (21)) ARM hard-float αρχιτεκτονική. Το συγκεκριμένο λειτουργικό σύστημα έχει αναπτυχθεί από μία ομάδα προγραμματιστών που δεν σχετίζονται με την εταιρεία που «βγάζει» το Raspberry (22) . Ουσιαστικά το συγκεκριμένο λειτουργικό βελτιστοποιεί το Debian για το Raspberry και προσφέρει πολλά παραπάνω πράγματα από ένα απλό λειτουργικό, καθώς είναι προεγκατεστημένα πάνω από 35.000 πακέτα για εύκολη εγκατάσταση (23) . Είναι ελεύθερο λογισμικό και η εγκατάστασή του είναι πολύ εύκολη.

Σκοπός του Raspbian είναι να μείνει όσο πιο πιστό στο Debian, όσο αυτό είναι εφικτό. Το Debian είναι ένα ελεύθερο λειτουργικό σύστημα που περιλαμβάνει βασικά σετ προγραμμάτων και λογισμικών. Στην κοινότητα του Linux έχει την φήμη ότι είναι πολύ σταθερό και παρέχει υψηλή ποιότητα. Θετικό είναι επίσης, ότι η κοινότητά του είναι διαθέσιμη κάθε στιγμή να προσφέρει την βοήθειά της.

Το Debian προσφέρει τρία διαφορετικά πακέτα, το Stable, Testing και το Unstable. Το πρώτο απ’ αυτά είναι το πιο διαδεδομένο κι έχει χρησιμοποιηθεί σαν βάση και για άλλα λειτουργικά όπως στην περίπτωση μας το Raspbian. Τα άλλα δύο με τον καιρό και την ανάπτυξη γίνονται τα καινούργια Stable και περνάνε στην αγορά. Μάλιστα, η πρώτη σταθερή έκδοση κυκλοφόρησε το 1996. Το Debian έχει πρόσβαση σε καταχωρήσεις στο ίντερνετ που περιέχουν πάνω από 50.000 πακέτα λογισμικού. Παρέχει μόνο ελεύθερο λογισμικό, αλλά μπορεί να ληφθεί και λογισμικό επί πληρωμή.

Όσον αφορά τα ενσωματωμένα συστήματα, το Debian υποστηρίζει πληθώρα συσκευών που έχουν βασιστεί σε ARM αρχιτεκτονικές. Ως παράδειγμα, το BeagleBoard που είναι κι ο αμερικάνικος ανταγωνιστής του Raspberry Pi και κατασκευάζεται απ’ την Texas Instrument έχει προεγκατεστημένο πια το Debian

Linux. Επίσης, υπάρχουν προσπάθειες για να μπορέσει το Debian να υποστηρίξει και ασύρματα σημεία πρόσβασης.

Η αρχιτεκτονική hard - float σημαίνει ότι η κάθε εφαρμογή περνάει τις παραμέτρους κινητής υποδιαστολής σε καταχωρητές κινητής υποδιαστολής, εν αντιθέσει με το soft-float που περνάει τις αντίστοιχες παραμέτρους σε καταχωρητές ακεραίων (24) . Είναι εμφανές ότι αφού χρησιμοποιούν διαφορετικούς καταχωρητές οι δύο αυτές αρχιτεκτονικές δεν είναι συμβατές, μπορούμε όμως να χρησιμοποιήσουμε hard floating point με το soft-float αλλά έχουμε μείωση στην απόδοση. Η έκδοση του Debian που έδινε η εταιρεία (Foundation) του Raspberry χρησιμοποιούσε soft-float, γεγονός που μείωνε την απόδοση του Pi μας, μιας και ο επεξεργαστής του έχει floating point hardware.

Python programming language

Η γλώσσα προγραμματισμού για τα προγράμματά μας που θα χρησιμοποιήσουμε είναι η Python (25). Το Raspbian έχει προεγκατεστημένη τόσο την δεύτερη, όσο και την τρίτη έκδοση της Python. Η συγκεκριμένη γλώσσα είναι μία υψηλού επιπέδου γλώσσα, γενικής χρήσεως και δυναμικής συμπεριφοράς. Η φιλοσοφία της γλώσσας αυτής είναι να δίνει έμφαση στην αναγνωσιμότητα του κώδικα και το συντακτικό της επιτρέπει στους χρήστες την έκφραση σε λιγότερες γραμμές κώδικα συγκρινόμενη με την C++ ή την Java. Η συγκεκριμένη γλώσσα παρέχει δομές που έχουν σκοπό να κάνουν δυνατή την συγγραφή ξεκάθαρων προγραμμάτων τόσο σε μικρή όσο και σε μεγάλη κλίμακα.

Υποστηρίζει πολλά πρότυπα προγραμματισμού όπως τον αντικειμενοστραφή. Επίσης, έχει αυτόματη διαχείριση μνήμης, μεγάλη βιβλιοθήκη και ένα δυναμικό σύστημα γραφής κώδικα. Οι διερμηνείς της Python είναι διαθέσιμοι σε πολλά λειτουργικά συστήματα επιτρέποντάς της να τρέχει σε μία πληθώρα συστημάτων. Με χρήση των κατάλληλων προγραμμάτων γίνεται εφικτή η δημιουργία εκτελέσιμων προγραμμάτων με κώδικα Python χωρίς την απαίτηση διερμηνέα.

Η Python δημιουργήθηκε στα τέλη της δεκαετίας του 1980 κι η υλοποίησή της ξεκίνησε το 1989 απ’ τον Guido van Rossum. Η Python 2.0 δημιουργήθηκε το 2000 και περιλάμβανε πολλά νέα χαρακτηριστικά όπως έναν κυκλικό συλλέκτη σκουπιδιών και υποστήριξη για Unicode. Η Python 3.0 βγήκε τον Δεκέμβρη του 2008 και δεν ήταν συμβατή με τις προηγούμενες εκδόσεις.

Ουσιαστικά, κύριος στόχος της Python είναι να προσφέρει μία γλώσσα προγραμματισμού που να είναι εύκολα αναγνώσιμη. Γι’ αυτό εξάλλου τον λόγο έχει σχεδιαστεί, συχνά χρησιμοποιώντας λέξεις του αγγλικού λεξιλογίου εκεί που οι άλλες γλώσσες χρησιμοποιούν σημεία στίξης. Ως αποτέλεσμα, έχει λιγότερες συντακτικές εξαιρέσεις.

Το μεγάλο πλεονέκτημα με την χρήση της για το Raspberry είναι η βιβλιοθήκη rpi gpio που με αυτήν μπορούμε να ελέγξουμε τους ακροδέκτες της GPIO (26) . Να σημειώσουμε ότι ενώ τώρα η συγκεκριμένη βιβλιοθήκη έρχεται προεγκατεστημένη,

παλιότερα δεν ίσχυε το ίδιο και ο χρήστης έπρεπε να την εγκαταστήσει χειρωνακτικά. Ακόμη να επισημάνουμε, ότι είναι καλό να καθαρίζουμε την GPIO μετά από κάθε πρόγραμμα και κάθε διεργασία για να αποφευχθεί τυχόν καταστροφή του Raspberry λόγω βραχυκυκλωμάτων που μπορεί να δημιουργηθούν στους ακροδέκτες της πλακέτας μας.

Ενώ έχουμε και την δεύτερη και την τρίτη έκδοση της Python τα προγράμματα, τα οποία έχουμε αναπτύξει στο πλαίσιο της διπλωματικής αυτής λειτουργούν και στις δύο εκδόσεις με ελάχιστες διαφορές.

Για να «τρέχουν» τα προγράμματα και στην παλιότερη έκδοση της Python το μόνο που πρέπει να κάνουμε είναι να αλλάξουμε την εντολή Print, διότι οι άλλες εντολές δεν επηρεάζονται κι έχουν παραμείνει ίδιες (27) . Ακριβέστερα, πλέον με την τελευταία έκδοση της Python η εντολή print χρειάζεται να έχει τα ορίσματα περικλεισμένα από παρενθέσεις, γεγονός που δεν ίσχυε στις παλιότερες εκδόσεις. Όλα τα προγράμματα έχουν γραφτεί σε Python 3.5.1.

Java Script

Η JavaScript (JS) είναι διερμηνευμένη γλώσσα προγραμματισμού για ηλεκτρονικούς υπολογιστές. Αρχικά αποτέλεσε μέρος της υλοποίησης των φυλλομετρητών Ιστού, ώστε τα σενάρια από την πλευρά του πελάτη (client-side scripts) να μπορούν να επικοινωνούν με τον χρήστη, να ανταλλάσσουν δεδομένα ασύγχρονα και να αλλάζουν δυναμικά το περιεχόμενο του εγγράφου που εμφανίζεται.

Η JavaScript είναι μια γλώσσα σεναρίων που βασίζεται στα πρωτότυπα (prototype-based), είναι δυναμική, με ασθενείς τύπους και έχει συναρτήσεις ως αντικείμενα πρώτης τάξης. Η σύνταξή της είναι επηρεασμένη από τη C. Η JavaScript αντιγράφει πολλά ονόματα και συμβάσεις ονοματοδοσίας από τη Java, αλλά γενικά οι δύο αυτές γλώσσες δε σχετίζονται και έχουν πολύ διαφορετική σημασιολογία. Οι βασικές αρχές σχεδιασμού της JavaScript προέρχονται από τις γλώσσες προγραμματισμού Self και Scheme.[2] Είναι γλώσσα βασισμένη σε διαφορετικά προγραμματιστικά παραδείγματα (multi-paradigm), υποστηρίζοντας αντικειμενοστρεφές,[3] προστακτικό και συναρτησιακό[4][5] στυλ προγραμματισμού.

Η JavaScript χρησιμοποιείται και σε εφαρμογές εκτός ιστοσελίδων — τέτοια παραδείγματα είναι τα έγγραφα PDF, οι εξειδικευμένοι φυλλομετρητές (site-specific browsers) και οι μικρές εφαρμογές της επιφάνειας εργασίας (desktop widgets). Οι νεότερες εικονικές μηχανές και πλαίσια ανάπτυξης για JavaScript (όπως το Node.js) έχουν επίσης κάνει τη JavaScript πιο δημοφιλή για την ανάπτυξη εφαρμογών Ιστού στην πλευρά του διακομιστή (server-side).

HTML

Η **HTML** (αρχικοποίηση του αγγλικού **H**yper**T**ext **M**arkup **L**anguage, ελλ. Γλώσσα Σήμανσης Υπερκειμένου) είναι η κύρια γλώσσα σήμανσης για τις ιστοσελίδες, και τα στοιχεία της είναι τα βασικά δομικά στοιχεία των ιστοσελίδων.

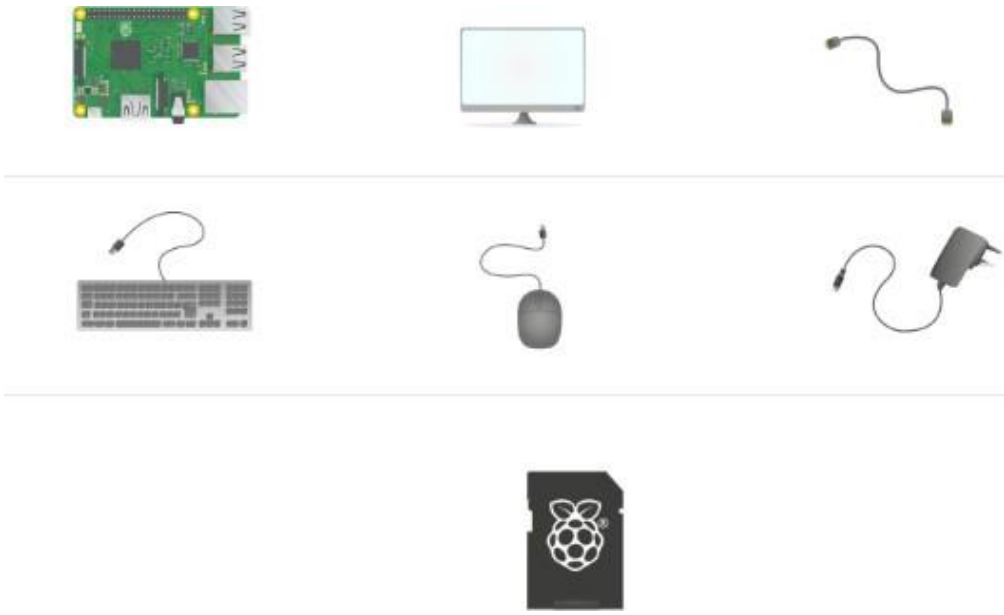
Η HTML γράφεται υπό μορφή στοιχείων HTML τα οποία αποτελούνται από *ετικέτες* (tags), οι οποίες περικλείονται μέσα σε σύμβολα «μεγαλύτερο από» και «μικρότερο από» (για παράδειγμα `<html>`), μέσα στο περιεχόμενο της ιστοσελίδας. Οι ετικέτες HTML συνήθως λειτουργούν ανά ζεύγη (για παράδειγμα `<h1>` και `</h1>`), με την πρώτη να ονομάζεται *ετικέτα έναρξης* και τη δεύτερη *ετικέτα λήξης* (ή σε άλλες περιπτώσεις *ετικέτα ανοίγματος* και *ετικέτα κλεισίματος* αντίστοιχα). Ανάμεσα στις ετικέτες, οι σχεδιαστές ιστοσελίδων μπορούν να τοποθετήσουν κείμενο, πίνακες, εικόνες κλπ. Ο σκοπός ενός web browser είναι να διαβάζει τα έγγραφα HTML και να τα συνθέσει σε σελίδες που μπορεί κανείς να διαβάσει ή να ακούσει. Ο browser δεν εμφανίζει τις ετικέτες HTML, αλλά τις χρησιμοποιεί για να παρουσιάσει το περιεχόμενο της σελίδας.

Τα στοιχεία της HTML χρησιμοποιούνται για να κτίσουν όλους του ιστότοπους. Η HTML επιτρέπει την ενσωμάτωση εικόνων και άλλων αντικειμένων μέσα στη σελίδα, και μπορεί να χρησιμοποιηθεί για να εμφανίσει διαδραστικές φόρμες. Παρέχει τις μεθόδους δημιουργίας δομημένων εγγράφων (δηλαδή εγγράφων που αποτελούνται από το περιεχόμενο που μεταφέρουν και από τον κώδικα μορφοποίησης του περιεχομένου) καθορίζοντας δομικά σημαντικά στοιχεία για το κείμενο, όπως κεφαλίδες, παραγράφους, λίστες, συνδέσμους, παραθέσεις και άλλα. Μπορούν επίσης να ενσωματώνονται σενάρια εντολών σε γλώσσες όπως η JavaScript, τα οποία επηρεάζουν τη συμπεριφορά των ιστοσελίδων HTML και από στατικές τις κάνουν διαδραστικές. Οι Web browsers μπορούν επίσης να αναφέρονται σε στυλ μορφοποίησης CSS για να ορίζουν την εμφάνιση και τη διάταξη του κειμένου και του υπόλοιπου υλικού. Ο οργανισμός W3C, ο οποίος δημιουργεί και συντηρεί τα πρότυπα για την HTML και τα CSS, ενθαρρύνει τη χρήση των CSS αντί διαφόρων στοιχείων της HTML για σκοπούς παρουσίασης του περιεχομένου.[1]

Εγκατάσταση με το σύστημα NOOBS (New Out Of Box Software).

Θα δούμε αναλυτικά τη σύνδεση Raspberry Pi, όλα τα περιφερειακά που χρειαζόμαστε, καθώς και την πρώτη εγκατάσταση Raspberry Pi με το σύστημα NOOBS (New Out Of Box Software).

Τι χρειαζόμαστε;



7 Τι χρειαζόμαστε

Η κάρτα Micro SD, θα πρέπει να έχει μέγεθος τουλάχιστον 4GB και να είναι τουλάχιστον Class 4.

Εγκατάσταση NOOBS στην κάρτα SD

Το NOOBS, από τα αρχικά της φράσης New Out Of Box Software – «καινούριο software του κουτιού», σε ελεύθερη μετάφραση – είναι ένα σύστημα που διευκολύνει σημαντικά την αρχική εγκατάσταση Raspberry Pi για τους αρχάριους.

Το πρώτο που χρειάζεται είναι να κάνουμε είναι να συνδέσουμε την κάρτα Micro SD με τον αντάπτορα στον υπολογιστή μας, και να τη Διαμορφώσουμε κατάλληλα.

Το ίδιο το NOOBS συνιστά να χρησιμοποιήσουμε την επίσημη εφαρμογή για Format σε SD, την οποία θα βρούμε στη διεύθυνση:

https://www.sdcard.org/downloads/formatter_4/eula_windows/

Κατεβάζουμε την εφαρμογή κάνοντας κλικ στο Accept στο κάτω μέρος της σελίδας.

Αποσυμπιέζουμε το .zip, εγκαθιστούμε και τρέχουμε την εφαρμογή.

Στο παράθυρο της εφαρμογής επιλέγουμε το «Option», στο οποίο αλλάζουμε το Format Size Adjustment σε «ON».

Κάνουμε κλικ στο Format, και σε λίγα δευτερόλεπτα η διαδικασία διαμόρφωσης έχει ολοκληρωθεί.

Με την κάρτα SD μας διαμορφωμένη, μπαίνουμε στη διεύθυνση https://www.raspberrypi.org/downloads/noobs/ και κατεβάζουμε την τελευταία έκδοση του NOOBS, είτε απευθείας από την ιστοσελίδα με μορφή συμπιεσμένου φακέλου .zip, είτε μέσω Torrent.

Το πλήρες Noobs έχει ενσωματωμένο το λειτουργικό σύστημα Rasbian, που είναι το πιο δημοφιλές λειτουργικό σύστημα για την εγκατάσταση Raspberry Pi, και μας δίνει πλήθος δυνατοτήτων.

Αφού κατεβάσουμε το συμπιεσμένο φάκελο .zip, κάνουμε εξαγωγή όλων των περιεχομένων του.

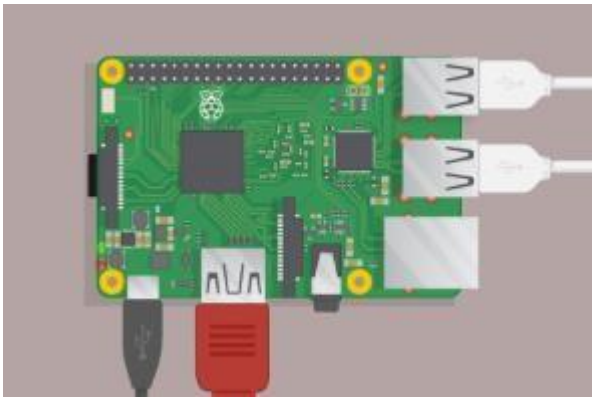
Ανοίγουμε το φάκελο στον οποίο έγιναν αποσυμπίεση και αντιγράφουμε όλα τα αρχεία στην κάρτα SD.

Αφού ολοκληρωθεί η αντιγραφή, κάνουμε ασφαλή αφαίρεση της κάρτας SD, για να ελαχιστοποιήσουμε την πιθανότητα καταστροφής δεδομένων.

Εγκατάσταση Rasbperry Pi με το Raspbian

Γυρίζουμε ανάποδα το Raspberry Pi για να τοποθετήσουμε την Micro SD. Υπάρχει ένας μόνο τρόπος για να μπει, και μπαίνοντας θα «κλειδώσει».

Για να αφαιρέσουμε την SD, την ξαναπατάμε προς τα μέσα, για να ξεκλειδώσει.



8 Εισαγωγή περιφεριακών και SD

Για τα άλλα καλώδια, το μόνο σημαντικό είναι να βάλουμε την τροφοδοσία τελευταία. Το Raspberry Pi δεν έχει διακόπτη On/Off. Μόλις το βάλουμε στην πρίζα, θα ξεκινήσει κατευθείαν.

Πρώτη εκκίνηση του Raspberry Pi

Ξεκινώντας, θα μας δείξει μια οθόνη με διάφορα χρώματα, γνωστή και σαν rainbow screen.

Σύντομα θα φορτώσει το NOOBS. Αν το Raspberry Pi δεν είναι συνδεδεμένο στο Internet, θα εμφανίσει στη λίστα μόνο το Rasbian.

Αν το Raspberry Pi είναι συνδεδεμένο στο Internet, το NOOBS θα μας δείξει και εναλλακτικά λειτουργικά συστήματα που μπορούμε να επιλέξουμε. Θα πρέπει όμως να περιμένουμε να κατεβάσει το καθένα από αυτά από το Internet.

Το πρώτο βήμα είναι στο κάτω μέρος της οθόνης να ορίσουμε σαν γλώσσα English (US) και σαν πληκτρολόγιο us. Τα ελληνικά δεν υπάρχουν σαν επιλογή, θα τα ρυθμίσουμε μετά την εγκατάσταση.

Ο λόγος είναι πως το προεπιλεγμένο πληκτρολόγιο του Ηνωμένου Βασιλείου έχει ελαφρώς διαφορετική διάταξη από τα πληκτρολόγια που έχουμε στην Ελλάδα, κυρίως όσον αφορά κάποια σύμβολα όπως το @ και το #.

Έχοντας τσεκάρει το Raspbian και κάνοντας κλικ στο Install, το σύστημα μας προειδοποιεί πως θα διαγραφεί όλο το περιεχόμενο της SD.

Επιλέγοντας «Yes», ξεκινάει η εγκατάσταση. Ανάλογα με την ταχύτητα της κάρτας SD που έχουμε βάλει, θα πάρει αρκετή ώρα.

Εφόσον όλα πάνε καλά, το σύστημα θα μας εμφανίσει το μήνυμα πως το λειτουργικό σύστημα (ή τα λειτουργικά συστήματα, αν επιλέξαμε πολλαπλά) εγκαταστάθηκαν επιτυχώς.

Κάνοντας κλικ στο OK, το Raspberry Pi θα κάνει επανεκκίνηση. Στην επόμενη εκκίνηση, θα μας βάλει στο περιβάλλον του Raspbian.

Στη συνέχεια ενημερώνουμε (update) τα πακέτα του συστήματος πληκτρολογώντας στο LXTerminal ή από την γραμμή εντολών:

```
sudo apt-get update
```

Μετά αναβαθμίζουμε (upgrade) τα εγκατεστημένα πακέτα με την εντολή

```
sudo apt-get dist-upgrade
```

Όσον αφορά την κάμερα, η εγκατάστασή της ήταν απλή αφού χρησιμοποιήσαμε μια απλή plug-n-play turbo-x, την οποία και αναγνώρισε αμέσως το rasbian.

3. Η ΕΦΑΡΜΟΓΗ

Ο Κώδικας

```
#!/usr/bin/env python3

#Βιβλιοθήκες
from bottle import route, run, template
from picamera import PiCamera
from time import sleep

import time
import RPi.GPIO as GPIO

########################################
## Add your IP address here
########################################
IP_ADDRESS ='192.168.43.87'
PORT = 8080
########################################

##camera=PiCamera()

##camera.start_preview()
##sleep(30)
##camera.stop_preview
```

 # Ορισμός των Pins που θα χρησιμοποιηθούν και την παραμετροποίηση τους.

```
GPIO.setmode(GPIO.BCM)
GPIO.setwarnings(False)
GPIO.setup(17,GPIO.OUT)# Forward Left
GPIO.setup(18,GPIO.OUT)# Backward Left
GPIO.setup(22,GPIO.OUT)# Forward Right
GPIO.setup(23,GPIO.OUT)# Backword Right

GPIO.setup(25,GPIO.OUT)#
pleft=GPIO.PWM(25,1000)# Right EN
pleft.start(100)      #

GPIO.setup(24,GPIO.OUT) #
pright=GPIO.PWM(24,1000)# Left EN
pright.start(100)      #


GPIO.setup(21, GPIO.OUT)#
pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
pArmRotate.start(10) # Initialization


GPIO.setup(20, GPIO.OUT)#
pArmLevel = GPIO.PWM(20, 50) # GPIO 20 for PWM with 50Hz
pArmLevel.start(9.5) # Initialization
```

```
GPIO.setup(16, GPIO.OUT)#
pArmBite = GPIO.PWM(16, 50) # GPIO 16 for PWM with 50Hz
pArmBite.start(10.5) # Initialization


#camera = PiCamera()
#capture_count=0
```

Σχεδιασμός δήλωση των Route με χρήση Python

```
@route('/')
def hello():
return '<b>Hi from RoboCar!</b>'

@route('/remote')
def remote():
```

Στοιχεία εμφάνισης σελίδας με χρήση HTML, CSS

```
return '''
<html>
<head>
<meta name="viewport" content="width=device-width, initial-scale=1">
<link rel="stylesheet" href="http://code.jquery.com/mobile/1.4.2/jquery.mobile-1.4.2.min.css">
<script src="http://code.jquery.com/jquery-1.10.2.min.js"></script>
<script src="http://code.jquery.com/mobile/1.4.2/jquery.mobile-1.4.2.min.js"></script>
```

#Δημιουργία Script με χρήση Javascript και Ajax

```
<script>
$(document).ready(function() {


 $("#moveleft").click(function() {
  $.ajax({
        url: '/remote/left',
        type: 'GET',
        data: { command:'left' }
  });
 });

 $("#moveright").click(function() {
  $.ajax({
        url: '/remote/right',
        type: 'GET',
        data: { command:'right' }
  });
 });

 $("#moveback").click(function() {
  var isRunning = $("#start").is(":checked") ? 1:0;
  $.ajax({
        url: '/remote/back',
```

```
        type: 'GET',
        data: { command:'right' }
  });
});


$("#play").click(function() {
  var cmd ='start';
  $.ajax({
        url: '/remote/play',
        type: 'GET',
        data: { command:cmd }
  });
});

$("#pause").click(function() {
  var cmd ='stop';
  $.ajax({
        url: '/remote/pause',
        type: 'GET',
        data: { command:cmd }
  });
});

$("#moveleftback").click(function(){
  $.ajax({
        url: '/remote/leftback',
        type: 'GET',
```

```
        data: { command:'leftback' }
  });
})

$("#moverightback").click(function(){
  $.ajax({
        url: '/remote/rightback',
        type: 'GET',
        data: { command:'rightback' }
  });
});

$("#armRotate span").click(function(){
  var position=$(this).attr("data-pos");

  $.ajax({
        url: '/remote/armRotate'+position,
        type: 'GET',
        data: {rotate: position}
  });
});

$("#armLevel span").click(function(){
  var positionLevel=$(this).attr("data-pos");

  $.ajax({
        url: '/remote/armLevel'+positionLevel,
```

```
        type: 'GET',
        data: {rotate: positionLevel}
    });
  });


  $("#armBite span").click(function(){
    var positionBite=$(this).attr("data-pos");

    $.ajax({
        url: '/remote/armBite'+positionBite,
        type: 'GET',
        data: {rotate: positionBite}
    });
  });


});
</script>
```

 # Δήλωση του style με χρήση CSS

```
<style>
.ui-content{
    margin:0px;
    padding:0px;
    background:#fff
}
span {
```

```
    padding:10px 15px;
    margin:5px;
    border:solid 2px black;
    text-align:center;
    display:inline-block;
    background:#000;
    color:#eee
}
div{
    text-align:center;
}
.left{
    float:left;
    width:50%;
    margin:0;
    padding:0
}

.diviframe{
    margin-top:50px;
    width:50%;
    float:right;

}

.diviframe iframe{
    border-width:0px!important
```

```
}

h3{
   margin:5px 0;
}

.clear{
   clear:both
}
</style>
</head>
<body>
<div data-role="page">
  <div data-role="main" class="ui-content">
   <div>
     <div class="left">
         <h3>Controls</h3>
         <div style="float:left;width:70%">
           <div>
              <span id="play">F</span>
           </div>
           <div>
              <span id="moveleftback">L</span>
              <span id="moveback">B</span>
              <span id="moverightback">R</span>
           </div>
```

```
    </div>
    <div style="float:right;width:30%;margin-top:25px;">
      <span id="pause" style="background:#c00">Stop</span>
    </div>
    <div class="clear"></div>
    <div>
    <h3>Arm</h3>
      <div id="armRotate">
        <span  data-pos="4">0</span>
        <span  data-pos="3">1</span>
        <span  data-pos="2">2</span>
        <span  data-pos="1">3</span>
        <span  data-pos="0">4</span>
      </div>
      <div id="armLevel">
        <span  data-pos="3">0</span>
        <span  data-pos="2">1</span>
        <span  data-pos="1">2</span>
        <span  data-pos="0">3</span>
      </div>
      <div id="armBite">
        <span data-pos="2">Unbite</span>
        <span data-pos="1">HalfBite</span>
        <span data-pos="0">FullBite</span>
      </div>
    </div>
  </div>
```

```
    <div class="diviframe">
```

#iframe για την εικόνα της κάμερας

```
      <iframe style="width:320px;height:240px;display:inline-block;text-
align:center" src="http://192.168.43.87:8081/"></iframe>



    </div>
 </div>
</div>
</body>
</html>
    '''
```

 # Δημιουργία των Route του χειριστηρίου, οι οποίες προωθούνται με την μέθοδο Ajax, και τις εντολές που θα εκτελεί το Raspberry, με χρήση Python

```
@route('/remote/play')
def play():
    GPIO.output(17,  True)
    GPIO.output(18, False)
    GPIO.output(22,  True)
    GPIO.output(23, False)
```

```
    return 'Starting'


@route('/remote/pause')
def pause():
    GPIO.output(17, False)
    GPIO.output(18, False)
    GPIO.output(22, False)
    GPIO.output(23, False)
    return 'Stopping'


@route('/remote/left')
def left():
    GPIO.output(22,  True)
    GPIO.output(23, False)

    GPIO.output(17,  True)
    GPIO.output(18, False)
    return 'moving left..'

@route('/remote/right')
def right():
    GPIO.output(17,  True)
    GPIO.output(18, False)

    GPIO.output(22, True)
```

```
    GPIO.output(23, False)
    return 'movingright'


@route('/remote/back')
def back():
    GPIO.output(17, False)
    GPIO.output(18,  True)
    GPIO.output(22, False)
    GPIO.output(23,  True)
    return 'reverse'

@route('/remote/leftback')
def leftback():
    GPIO.output(22,  True)
    GPIO.output(23, False)

    GPIO.output(17, False)
    GPIO.output(18, True)
    return 'moving left backwards'

@route('/remote/rightback')
def rightback():
    GPIO.output(17,  True)
    GPIO.output(18, False)

    GPIO.output(22, False)
```

```
    GPIO.output(23, True)
    return 'moving right backwards'


@route('/remote/armRotate0')
def armRotate0():
#     GPIO.setmode(GPIO.BCM)
#     GPIO.setup(21, GPIO.OUT)#
#     pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
    pArmRotate .ChangeDutyCycle(8)
    time.sleep(1)

@route('/remote/armRotate1')
def armRotate1():
#     GPIO.setmode(GPIO.BCM)
#     GPIO.setup(21, GPIO.OUT)#
#     pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
    pArmRotate .ChangeDutyCycle(9)
    time.sleep(1)

@route('/remote/armRotate2')
def armRotate2():
#     GPIO.setmode(GPIO.BCM)
#     GPIO.setup(21, GPIO.OUT)#
#     pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
    pArmRotate .ChangeDutyCycle(10)
    time.sleep(1)
```

```
@route('/remote/armRotate3')
def armRotate3():
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#         pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
        pArmRotate .ChangeDutyCycle(11)
        time.sleep(1)


@route('/remote/armRotate4')
def armRotate4():
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#       pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
        pArmRotate .ChangeDutyCycle(12)
        time.sleep(1)


@route('/remote/armLevel0')
def armLevel0():
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#       pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
        pArmLevel.ChangeDutyCycle(11)
        time.sleep(1)
```

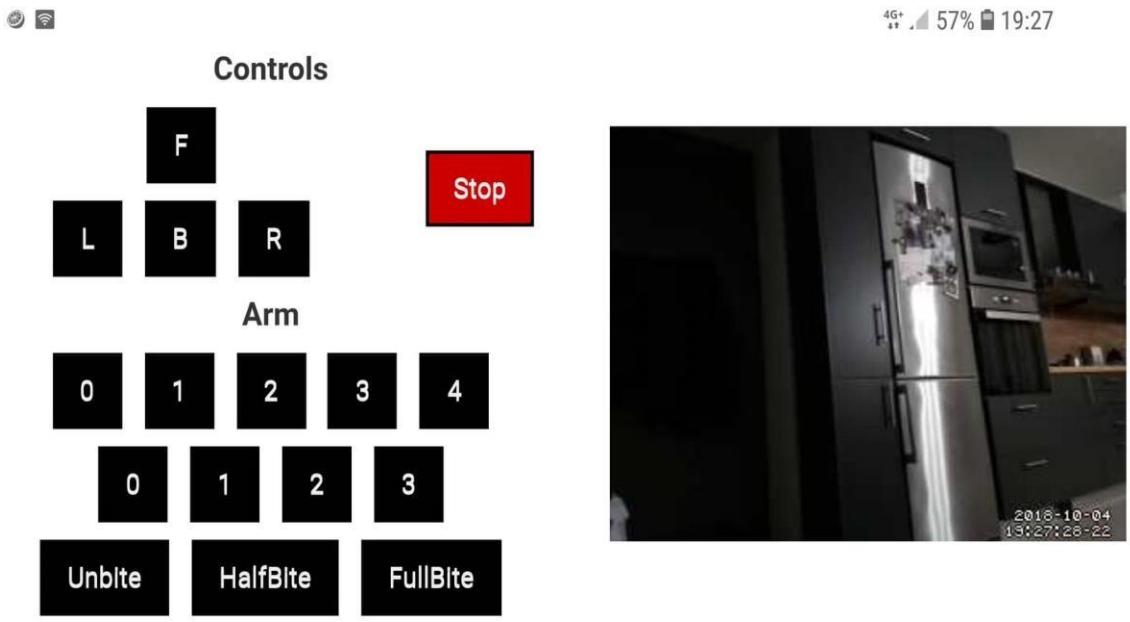
```
@route('/remote/armLevel1')
def armLevel1():
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#       pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
        pArmLevel .ChangeDutyCycle(10)
        time.sleep(1)

@route('/remote/armLevel2')
def armLevel2():
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#       pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
        pArmLevel .ChangeDutyCycle(9)
        time.sleep(1)

@route('/remote/armLevel3')
def armLevel3():
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#       pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
        pArmLevel .ChangeDutyCycle(8)
        time.sleep(1)

@route('/remote/armBite0')
def armBite0():
```

```
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#       pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
      pArmBite .ChangeDutyCycle(13.5)
      time.sleep(1)

@route('/remote/armBite1')
def armBite1():
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#       pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
      pArmBite .ChangeDutyCycle(10.5)
      time.sleep(1)

@route('/remote/armBite2')
def armBite2():
#       GPIO.setmode(GPIO.BCM)
#        GPIO.setup(21, GPIO.OUT)#
#       pArmRotate = GPIO.PWM(21, 50) # GPIO 21 for PWM with 50Hz
      pArmBite .ChangeDutyCycle(9)
      time.sleep(1)

##@route('/remote/armBite2')
##def capture():
##   camera.capture('snapshot'+capture_count+'.jpg')
##   capture_count+=1
```

```
try:
    run(host = IP_ADDRESS, port= PORT)
except(KeyboardInterrupt):
    # If a keyboard interrupt is detected then it exits cleanly!
    print('Finishing up!')
    GPIO.output(17, False)
    GPIO.output(18, False)
    GPIO.output(22, False)
    GPIO.output(23, False)
    GPIO.cleanup()
    quit()
```

Η Λειτουργία

Αρχικά συνδεόμαστε μέσω tethering με την κινητή μας συσκευή. Έπειτα μέσω του Browser που χρησιμοποιούμε θα συνδεθούμε στη διεύθυνση 192.168.43.87:8080/remote και εκεί έχουμε την παρακάτω εικόνα:



9 Η εφαρμογή

Βλέπουμε ότι αριστερά έχουν μπει τα κουμπιά για τις λειτουργίες και την κίνηση του ρομποτικού οχήματος, ενώ δεξιά έχουμε live streaming από την κάμερα που τοποθετήθηκε.

Στα **Controls** υπάρχουν τα κουμπιά **F L B R STOP** τα οποία μας δίνουν και την κίνηση του οχήματος.

- F: Μπροστά Κίνηση
- L: Λειτουργία αριστερής ρόδας για αριστερή στροφή
- R: Λειτουργία δεξιάς ρόδας για αριστερή στροφή
- B: Όπισθεν λειτουργίας
- STOP: Διακοπή κίνησης

Παρακάτω βλέπουμε την εικόνα **ARM** με τα κουμπιά
0 1 2 3 4 / 0 1 2 3 / Unbite HalfBite Fullbite τα οποία μας δίνουν κίνηση στον ρομποτικό μας βραχίονα.
Στα πρώτα αριθμημένα κουμπιά (0-4) έχουμε την κίνηση του βραχίονα στον άξονα των χ με το 2 να είναι στη μέση τα 0 και 1 αριστερά και τα 3 και 4 δεξιά.

Στα δεύτερα αριθμημένα κουμπιά (0-3) έχουμε την κίνηση του βραχίονα στον άξονα των y, δηλαδή την κίνηση «πάνω-κάτω», με το 0 να κατεβάζει το βραχίονα στην πιο κάτω θέση ενώ το 2 στην υψηλότερη.

Τέλος τα κουμπιά Unbite HalfBite Fullbite τα χρησιμοποιούμε για το κλείσιμο και άνοιγμα της δαγκάνας και με το Unbite να είναι τελείως ανοιχτή, το HalfBite στη μέση και το Fullbite να είναι τελείως κλειστή.

Να τονίσουμε ότι κάθε φορά που ενεργοποιούμε το ρομποτικό όχημα η δαγκάνα επανέρχεται σε ευθεία θέση, δηλαδή στην θέση των arm controls 2-1-Unbite.

4. ΣΥΜΕΡΑΣΜΑΤΑ

Η Ρομποτική (Robotics) είναι κλάδος της τεχνολογίας που ασχολείται με τη σχεδίαση, την ανάπτυξη και τη μελέτη ρομπότ. Η επιστήμη της Ρομποτικής αποτελεί συνδυασμό πολλών άλλων επιστημών, κυρίως δε της πληροφορικής, της ηλεκτρονικής και της μηχανολογίας. Η λέξη ρομπότ (Robot) προέρχεται από το Σλαβικό robota που σημαίνει εργασία. Τα ρομπότ είναι αυτόματες μηχανές με προγραμματισμένη συμπεριφορά, η χρήση των οποίων αποσκοπεί στην αντικατάσταση του ανθρώπου στην εκτέλεση έργου, τόσο σε φυσικό επίπεδο όσο και σε επίπεδο λήψης αποφάσεων.

Για τη δημιουργία ενός καινούριου ρομποτικού οχήματος και της εφαρμογής του χρειάζεται καλή γνώση προγραμματισμού, λεπτές κινήσεις κατά την κατασκευή του οχήματος και οφείλω να αναφέρω πως αντιμετωπίστηκαν αρκετές δυσκολίες σε αυτό το κομμάτι και στο στήσιμο της. Μεγάλη βοήθεια καθιστά το διαδίκτυο όπου υπάρχει μεγάλος αριθμός οδηγιών και βίντεο. Κατά τη διάρκεια της εγγραφής της διπλωματικής βρήκαμε εντυπωσιακό το μέγεθος της κοινότητας των προγραμματιστών και την ποικιλία των forums και των απαντήσεων που αντιστοιχεί σε κάθε bug που μπορεί να προκύψει στην κατασκευή και στον κώδικα. Συμπερασματικά, παρά το γεγονός ότι η ανάπτυξη μιας εφαρμογής δεν είναι απλή ασχολία, η κατάλληλη έρευνα, η απαιτούμενη μελέτη και κάποια εξειδικευμένα και εύχρηστα εργαλεία μπορούν να καταστήσουν τη διαδικασία ευχάριστη και να μεγιστοποιήσουν την ικανοποίηση που φέρνει το τελικό αποτέλεσμα.