ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ ΚΕΝΤΡΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ

ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ Τ.Ε.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΛΟΠΟΙΗΣΗ ΣΕΙΡΑΣ ΕΦΑΡΜΟΓΩΝ ΜΕ ΤΟ RASPBERRY PI®

Σταύρος Φραγκόπουλος ΑΜ : 3577

Επιβλέπων: Ιωάννης Καλόμοιρος, Αναπληρωτής Καθηγητής

ΣΕΡΡΕΣ, ΙΟΥΝΙΟΣ 2017

Υπεύθυνη Δήλωση : Βεβαιώνω ότι είμαι συγγραφέας αυτής της πτυχιακής εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην πτυχιακή εργασία. Επίσης έχω αναφέρει τις πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται επακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η πτυχιακή εργασία προετοιμάστηκε από εμένα προσωπικά, ειδικά για τις απαιτήσεις του προγράμματος σπουδών του Τμήματος Μηχανικών Πληροφορικής Τ.Ε. του Τ.Ε.Ι. Κεντρικής Μακεδονίας.

ΠΕΡΙΛΗΨΗ

Ο σκοπός αυτής της πτυχιακής εργασίας είναι να γίνει μια εισαγωγή για το τι είναι ένα “single board computer (SBC)”, παρουσιάζοντας το Raspberry Pi ως αντιπρόσωπο αυτής της κατηγορίας υπολογιστών. Θα γίνει μια αναφορά των χαρακτηριστικών της συσκευής, της εγκατάστασης και χρήσης του λειτουργικού της συστήματος, καθώς και των πρωτοκόλλων διασύνδεσης που διαθέτει. Επίσης θα γίνει η αναλυτική περιγραφή των βημάτων που είναι απαραίτητα ώστε να τεθεί σε λειτουργία η συσκευή. Θα γίνει εισαγωγή στον προγραμματισμό της συσκευής με την χρήση της γλώσσας προγραμματισμού Python. Θα εξηγηθούν τρόποι σύνδεσης και προγραμματισμού συσκευών και αισθητήρων. Τέλος, θα εξηγηθεί ο τρόπος χρήσης της συσκευής μέσω VNC.

Περιεχόμενα

ΕΙΣΑΓΩΓΗ

Στην συγκεκριμένη εργασία θα γίνει μια προσπάθεια να προετοιμαστεί ο αναγνώστης ώστε να είναι σε θέση να ετοιμάσει, να χρησιμοποιήσει και να προγραμματίσει μια συσκευή Raspberry Pi.
Το παρόν εγχειρίδιο προϋποθέτει ότι ο αναγνώστης έχει γνώσεις λειτουργικών συστημάτων, γλωσσών προγραμματισμού, και γνώσεις που σχετίζονται με κυκλώματα.

Τι Έπεται

Το πρώτο κεφάλαιο εξηγεί τι είναι το Raspberry Pi και ποια είναι τα χαρακτηριστικά του. Στο δεύτερο κεφάλαιο θα εξηγηθούν τα βήματα που είναι απαραίτητα ώστε να προετοιμαστεί το σύστημα raspberry για χρήση. Στο τρίτο κεφάλαιο θα γίνει μια μικρή περιήγηση στο λειτουργικό σύστημα Raspbian. Το τέταρτο κεφάλαιο είναι ένα συνοπτικό εγχειρίδιο της γλώσσας προγραμματισμού Python3. Στο πέμπτο κεφάλαιο θα εξηγηθούν μερικοί τρόποι σύνδεσης και προγραμματισμού συσκευών και αισθητήρων. Στο έκτο κεφάλαιο θα εξηγηθεί ο τρόπος ρύθμισης του raspberry ώστε να λειτούργει μέσω VNC. Στο έβδομο κεφάλαιο θα εξηγηθεί ένας απλός τρόπος λήψης στιγμιότυπων από μια USB webcam.

Κεφάλαιο 1

Το Raspberry Pi

Σε αυτό το κεφάλαιο θα γίνει παρουσίαση για το τι είναι το Raspberry Pi, ποια είναι τα χαρακτηριστικά ενός τέτοιου συστήματος και οι κυριότεροι τομείς στους οποίους μπορεί να χρησιμοποιηθεί αυτή η συσκευή, καθώς και η αναλυτική περιγραφή των δυο συστημάτων Raspberry Pi που θα χρησιμοποιηθούν σε αυτήν την πτυχιακή.

1.1 Τι είναι ένα “Single board computer (SBC)” και τι ένα “Σύστημα σε ψηφίδα (SoC)”

Ένα “Single board Computer (SBC)”, είναι η μέθοδος με την οποία όλα τα απαραίτητα ηλεκτρονικά στοιχεία που είναι απαραίτητα ώστε ένα σύστημα να χαρακτηριστεί ηλεκτρονικός υπολογιστής, τοποθετούνται σε μια πλακέτα.

Ως “System on a chip (SoC)” ή χαρακτηρίζεται ένα σύστημα το οποίο εμπεριέχει συστήματα όπως μικροεπεξεργαστή (ή μικροελεγκτή), κάρτα γραφικών και άλλα υποσυστήματα σε ένα μόνο μικροτσίπ.

1.2 Τι είναι το Raspberry Pi;

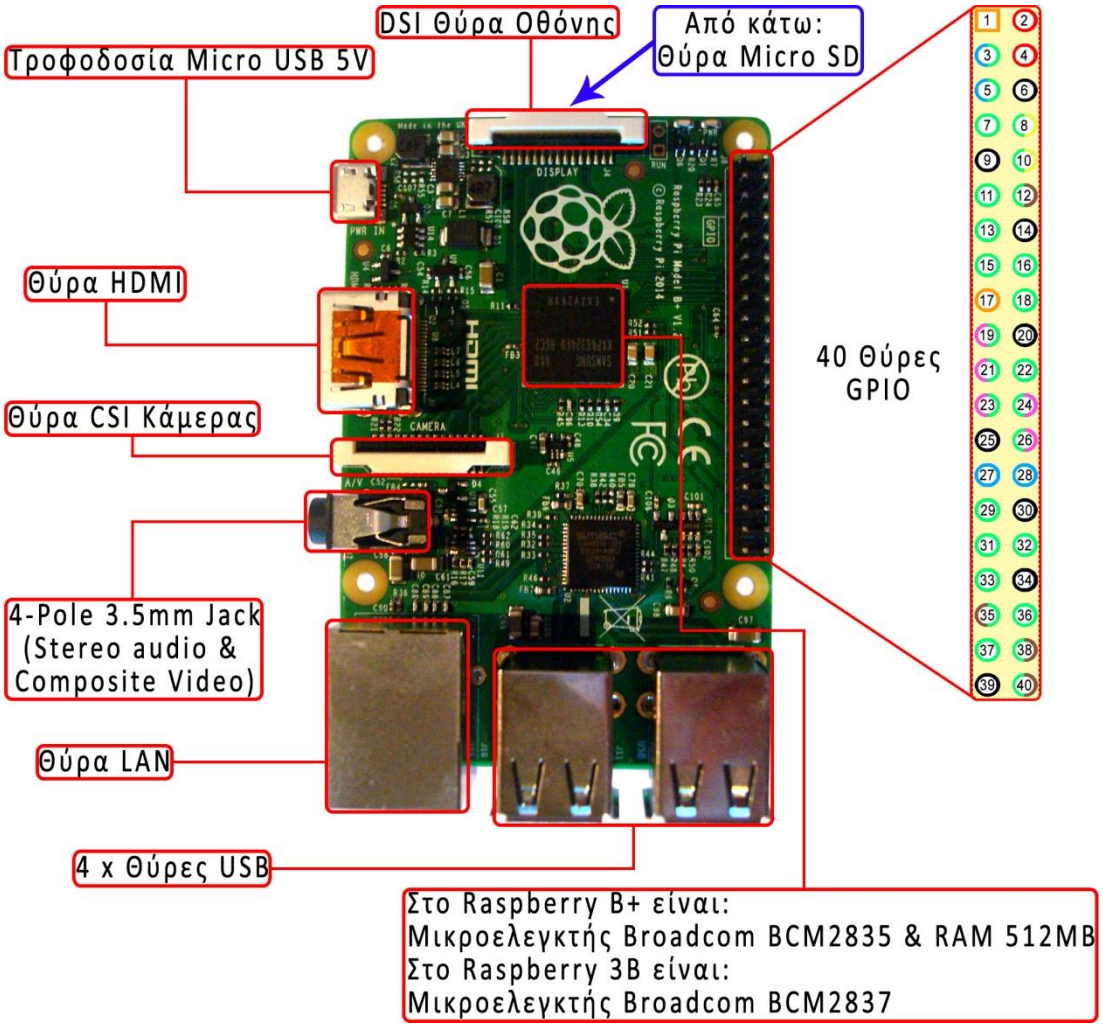
Το Raspberry Pi είναι ένας μικρός υπολογιστής σε μέγεθος πιστωτικής κάρτας. Είναι ικανό να χρησιμοποιηθεί σε εφαρμογές ηλεκτρονικών κυκλωμάτων, καθώς και να χρησιμοποιηθεί σαν απλός προσωπικός υπολογιστής κάνοντας απλές εργασίες όπως επεξεργασία κειμένου, εικόνας, σύνδεση στο διαδίκτυο, αλλά έχει και την ικανότητα αναπαραγωγής video υψηλής ανάλυσης.

Το όνομα του προέκυψε από την παράδοση που υπήρχε παλιά στην ονοματολογία των μικροϋπολογιστών να παίρνουν ονόματα φρούτων, όπως για παράδειγμα η Tangerine Computer Systems, η Apricot Computers και η Acorn.

Η κατάληξη Pi (προφέρεται: ”πάϊ”) στο όνομα του, είναι λόγω του ότι αρχικά η εταιρία ήθελε να δημιουργήσει έναν υπολογιστή ο όποιος θα έτρεχε μόνο προγράμματα Python, άλλα το τελικό προϊόν κατέληξε να λειτουργεί ως κανονικός υπολογιστής.

Το Raspberry Pi ξεκίνησε να παράγεται στην Αγγλία, από την Raspberry Pi Foundation η οποία ιδρύθηκε το 2009, και είναι μη κερδοσκοπική εταιρία, που ξεκίνησε με βασικό σκοπό να προωθήσει την διδασκαλία στις αρχές της επιστήμης των υπολογιστών και του προγραμματισμού στα σχολεία.

1.3 Τα χαρακτηριστικά των συστημάτων Raspberry Pi, B+ και 3 B



Εικόνα 1.3.1 (Μητρική πλακέτα Raspberry)

Με εξαίρεση τα μοντέλα Raspberry Pi 2B και 3B, όλες οι υπόλοιπες εκδόσεις Raspberry Pi, έχουν τον μικροελεγκτή Broadcom BCM2835, ο οποίος περιέχει τον μονοπύρηνο επεξεργαστή ARM1176JZFS με ικανότητα πράξεων κινητής υποδιαστολής, ο οποίος τρέχει στα 700MHz.

Όλα τα μοντέλα Raspberry Pi έχουν την ίδια κάρτα γραφικών (Video core IV), αλλά διαφέρουν στα χαρακτηριστικά όπως η μνήμη Ram, αλλά και τους τρόπους διασύνδεσης όπως θύρες USB, HDMI, audio jack…

Η κάρτα γραφικών έχει την ικανότητα αναπαραγωγής video Blu-Ray ποιότητας, χρησιμοποιώντας H.264 στα 40MBits/s. Ο πυρήνας της κάρτας γραφικών είναι προσβάσιμος από τις βιβλιοθήκες OpenGL ES2.0 και OpenVG.

Το μοντέλο 2B χρησιμοποιεί τον μικροελεγκτή Broadcom BCM2836, ο οποίος εμπεριέχει έναν τετραπύρηνο επεξεργαστή (900 MHz 32-bit) ARM Cortex-a7 με ικανότητα πράξεων κινητής υποδιαστολής.
Το μοντέλο 3B χρησιμοποιεί τον μικροελεγκτή Broadcom BCM2837, ο οποίος εμπεριέχει έναν τετραπύρηνο επεξεργαστή(1.2 GHz 64-bit) ARM Cortex-a53.

Τα πλήρες χαρακτηριστικά των συστημάτων Raspberry Pi B+ και Raspberry Pi 3B που θα χρησιμοποιηθούν στην πτυχιακή εργασία:

Χαρακτηριστικά του Raspberry Pi B+:

System on a chip (SoC): Broadcom BCM2835(CPU+GPU. H SDRAM βρίσκεται πάνω στον μικροελεγκτή σε ξεχωριστό μικροτσίπ).
CPU: 700MHz ARM11 ARM1176JZF-S core.
GPU: Broadcom VideoCore IV, OpenGL ES 2.0, OpenVG 1080p30 H.264 high-profile encode/decode, 250 MHz.
Memory (SDRAM): 512MB.
Θύρες USB: 4xUSB 2.0 (μέσω ενσωματωμένου USB hub in LAN9512).
Έξοδος Βίντεο: Composite video/Composite RCA, HDMI (not at the same time).
Έξοδος ήχου: TRS connector/3.5mm jack, HDMI
Είσοδος ήχου: None, but a USB mic or sound-card could be added.
Μέσω αποθήκευσης: Secure Digital| SD/ MMC/ SDIO card slot.
Διασύνδεση Δικτύου: 10/100 wired Ethernet RJ45
Περιφερικά χαμηλού επιπέδου: 26 General Purpose Input/Output(GPIO) pins, Serial Peripheral Interface Bus(SPI), I2C, PCM(I2S), Universal asynchronous receiver/transmitter(UART)
Κατανάλωση Ισχύος: 600mA, (3 W)
Τροφοδοσία: 5V(DC) via Micro USB type B or GPIO header.
Διαστάσεις: 85.0mm x 56mm x 15mm (μέγεθος πιστωτικής κάρτας)

Χαρακτηριστικά του Raspberry Pi 3B :

System on a chip(SoC): Broadcom BCM2837(CPU+GPU).

CPU: 1.2 GHz 64-bit quad-core ARM Cortex-A53.

GPU: Broadcom VideoCore IV, OpenGL ES 2.0, OpenVG 1080p30 H.264 high-profile encode/decode, 250 MHz.

Memory(SDRAM): 1024MB.

Θύρες USB: 4xUSB 2.0(μέσω ενσωματωμένου USB hub in LAN9512).

Έξοδος Βίντεο: Composite video/Composite RCA, HDMI (not at the same time).

Έξοδος ήχου: TRS connector/3.5mm jack, HDMI

Είσοδος ήχου: None, but a USB mic or sound-card could be added.

Μέσω αποθήκευσης: Secure Digital| SD/ MMC/ SDIO card slot.

Διασύνδεση Δικτύου: 10/100 wired Ethernet RJ45,wifi,bluetooth

Περιφεριακά χαμηλού επιπέδου: 26 General Purpose Input/Output(GPIO) pins, Serial Peripheral Interface Bus(SPI), I2C, PCM(I2S), Universal asynchronous receiver/transmitter(UART)

Κατανάλωση Ισχύος: 800mA, (4 W)

Τροφοδοσία: 5V(DC) via Micro USB type B or GPIO header.

Διαστάσεις: 85.0mm x 56mm x 15mm (μέγεθος πιστωτικής κάρτας)

1.4 Ποιοι είναι οι τομείς στους οποίους αξιοποιείται αυτή η συσκευή

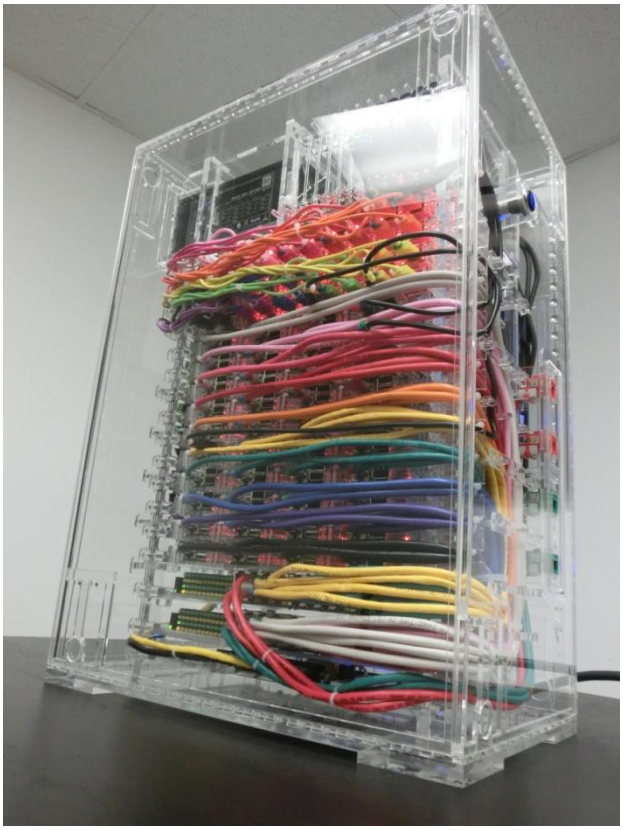
Ο βασικός τομέας για τον οποίο αρχικά αναπτύχτηκε η συσκευή ήταν για τον τομέα της εκπαίδευσης και αρχικός σκοπός της ήταν να εισάγει τις βασικές αρχές της επιστήμης των υπολογιστών και του προγραμματισμού στα σχολεία. Αλλά, τελικά, η χαμηλή τιμή της και οι ικανότητές της, την βοήθησαν να γίνει δημοφιλής και σε άλλους τομείς, όπως για παράδειγμα, ο τομέας αυτοματισμών σπιτιών και ο τομέας αυτοματισμών επιχειρήσεων (με το RI Compute Module).

Επίσης, χρησιμοποιήθηκε σε εφαρμογές που σχετίζονται με: τη ρομποτική, τα ηλεκτρονικά παιχνίδια, media centers, 3d printers, μηχανική όραση και αναγνώριση προτύπων, στον απομακρυσμένο έλεγχο, ως Virtual assistant(Amazon Alexa, Google Home), καθώς και σε πολλές άλλες εφαρμογές.

1.5 Αξιόλογες κατασκευές με το Raspberry Pi

Στην συνέχεια θα αναφερθούν μερικές αξιόλογες κατασκευές που έχουν γίνει με την χρήση του συστήματος Raspberry Pi.

Raspberry Pi Cluster



Εικόνα 1.5.1 (Raspberry Pi Cluster)

Ένα raspberry Cluster, μπορεί να μιμηθεί τις λειτουργιές των μοντέρνων Cluster υπερυπολογιστών. Είναι προσβάσιμο από ιδιώτες και οργανισμούς χάρη στο ιδιαίτερα χαμηλό κόστος κατασκευής του. Παρέχει στους χρήστες του έναν τρόπο να χειριστούν εφαρμογές που τρέχουν σε υπερυπολογιστές, και έτσι είναι χρήσιμος για εκπαιδευτικούς σκοπούς.

3D Printer

Εικόνα 1.5.2 (Κατασκευή με το RPi 3D εκτυπωτή)

Πλήρες 3D εκτυπωτής βασισμένος σε raspberry pi. Το ίδιο το raspberry τρέχει το λογισμικό αλλά και τα μηχανικά μέρη του εκτυπωτή.

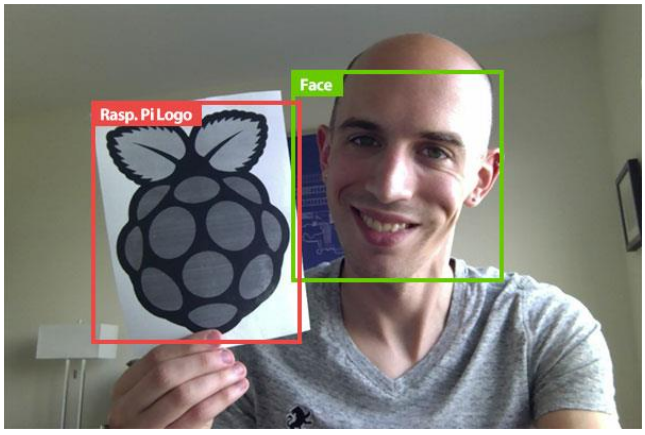
NAS

Δημιουργία μιας μονάδας αποθήκευσης που είναι προσαρτημένη στο δίκτυο (Network Attached Storage).

Web server

Δημιουργία ενός πλήρους λειτουργικού web server

Μηχανική όραση (Open Cv)



Εικόνα 1.5.3 (Αναγνώριση με Open CV)

Από το τρίτο μοντέλο της σειράς raspberry και μετά είναι εφικτή η χρήση του λογισμικού Open CV σε συνεργασία με μια κάμερα, ώστε να έχουμε αναγνώριση προσώπου/αντικειμένων.

Κεφάλαιο 2

Προετοιμασία ενός συστήματος Raspberry Pi

Σε αυτό το κεφάλαιο θα ασχοληθούμε με τα βήματα που είναι απαραίτητα ώστε να τεθεί ένα σύστημα Raspberry Pi σε λειτουργία.

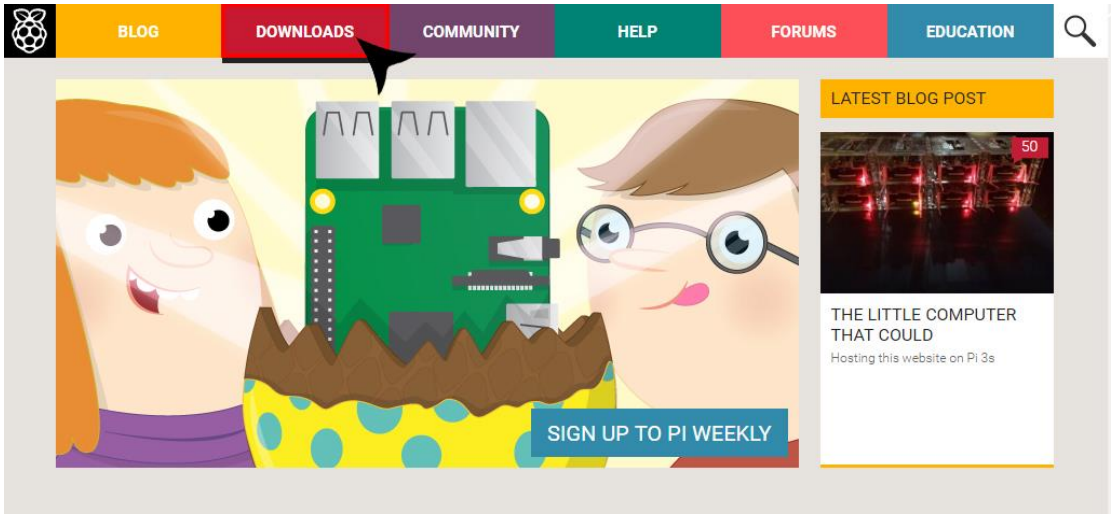
2.1 Απαραίτητος Εξοπλισμός

Για να είναι εφικτό να εργαστούμε με το Raspberry Pi είναι απαραίτητα τα εξής αντικείμενα:

- Μια κάρτα MicroSD με χωρητικότητα 8GB και άνω.
- Ένα MicroSD προσαρμογέα σε USB, ώστε να συνδέσουμε την MicroSD κάρτα σε υπολογιστή, για να εγκαταστήσουμε το λειτουργικό σύστημα στην κάρτα.
- Ένα Micro USB τροφοδοτικό (είναι εφικτή η χρήση φορτιστή Android κινητού). Για την βέλτιστη χρήση του Raspberry Pi ενδείκνυται τροφοδοτικό που να παρέχει περίπου 5V, και 2.5A.
- Καλώδιο για την σύνδεση του Raspberry σε οθόνη. Το Raspberry μας παρέχει τρεις βασικούς τρόπους διασύνδεσης, HDMI, Composite RCA jack, DSI display port. Είναι εφικτή και η διασύνδεση σε DVI με προσαρμογέα HDMI σε DVI. Σημειώνεται ότι είναι εφικτό να δοθεί σήμα σε όλες τις θύρες ταυτόχρονα.
- Ένα USB πληκτρολόγιο και ένα USB ποντίκι.
- Ένα καλώδιο δικτύου ή ένας προσαρμογέας ασυρμάτου δικτύου Wi-Fi.

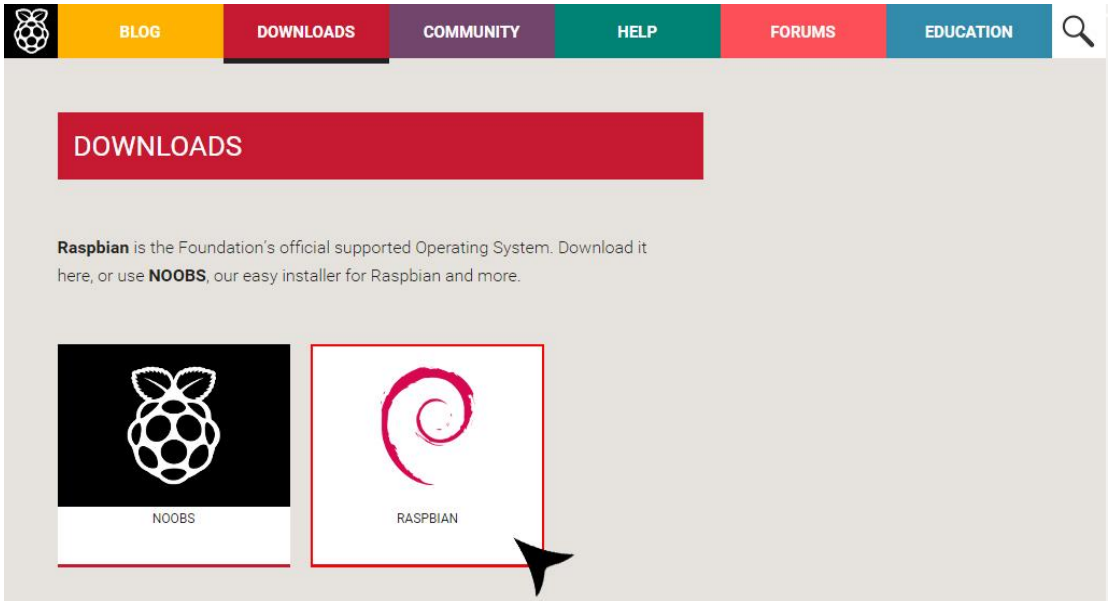
2.2 Λήψη και εγκατάσταση του RaspbianOS

Για να κατεβάσουμε το λειτουργικό σύστημα το οποίο θα τρέξουμε στο Raspberry Pi πρέπει να επισκεφτούμε τον ισότοπο www.raspberrypi.org και να επιλέξουμε την καρτέλα Downloads:



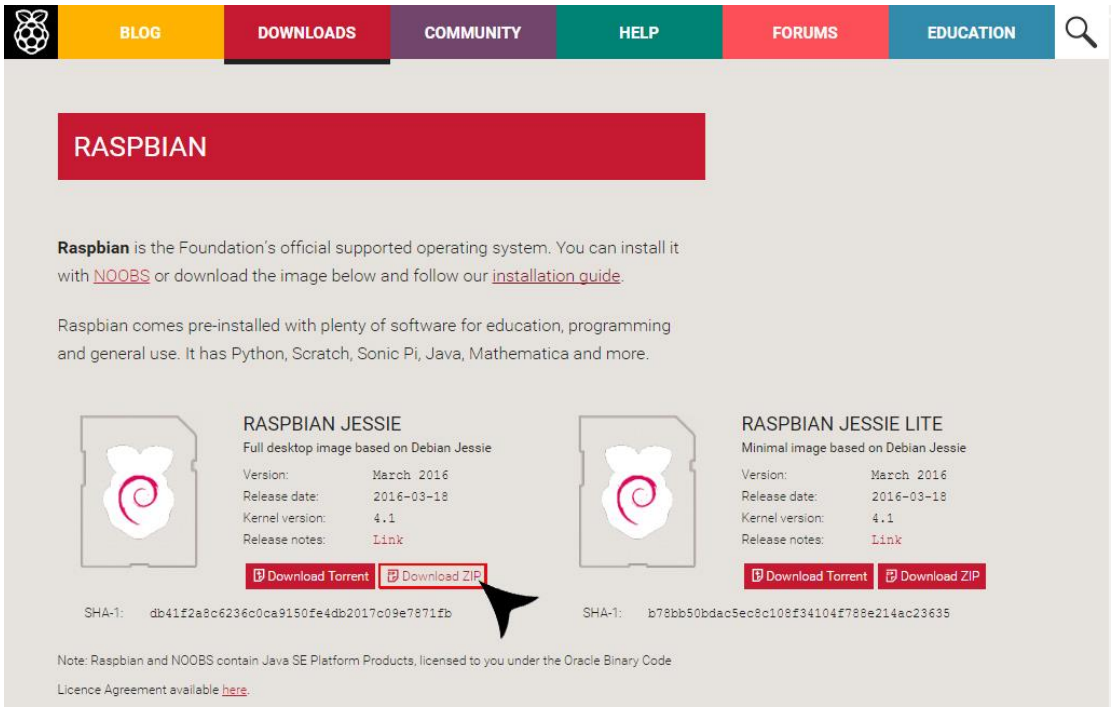
Εικόνα 2.2.1 (αρχική σελίδα του raspberrypi.org)

Στη σελίδα που θα μας εμφανίσει επιλέγουμε τον σύνδεσμο Raspbian:



Εικόνα 2.2.2 (η καρτέλα downloads του ισότοπου raspberrypi.org)

Στην σελίδα που θα φορτώσει επιλέγουμε την έκδοση του Raspbian που θέλουμε να κατεβάσουμε, αλλά και τον τρόπο λήψης. Επιλέγουμε να κατεβάσουμε την πλήρη έκδοση σε μορφή zip.



Εικόνα 2.2.3 (επιλογή λήψης του raspbian)

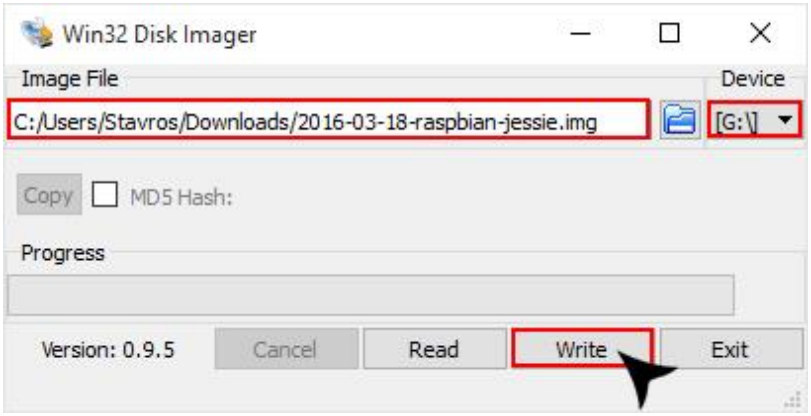
Μόλις κατέβει η εικόνα εγκατάστασης (ISO) του λειτουργικού Raspbian θα πρέπει να την περάσουμε στην Micro SD Card που θα χρησιμοποιήσουμε ως αποθηκευτικό μέσο για το Raspberry Pi μας. Για να το κατορθώσουμε αυτό, θα πρέπει να κατεβάσουμε και να εγκαταστήσουμε (για Windows) το βοηθητικό πρόγραμμα εγκατάστασης Win32DiskImager το οποίο θα αναλάβει την εγκατάσταση του λειτουργικού Raspbian στην MicroSD μας.

Για να κατεβάσουμε το βοηθητικό πρόγραμμα θα πρέπει να επισκεφτούμε τον ισότοπο https://sourceforge.net/projects/win32diskimager/ και να κατεβάσουμε το Win32 DiskImager. Από την σελίδα που θα μας φορτώσει επιλέγουμε το κουμπί που γράφει Download ώστε να κατεβάσουμε το Win32 DiskImager.

Κάνουμε εγκατάσταση το Win32 DiskImager.

Μόλις τελειώσει η εγκατάσταση συνδέουμε την MicroSDcard στον υπολογιστή μας με την βοήθεια ενός cardreader.

Αποσυμπιέζουμε το zip αρχείο στο οποίο βρίσκεται η εικόνα του λειτουργικού Raspbian.
Ύστερα τρέχουμε το Win32 DiskImager και στο παράθυρο που μας ανοίγει επιλέγουμε την εικόνα του λειτουργικού Raspbian που αποσυμπιέσαμε προηγουμένως, καθώς και την μονάδα δίσκου στην οποία αντιστοιχεί η MicroSD μας.
Μόλις βεβαιωθούμε ότι τα κάναμε όλα σωστά πατάμε το κουμπί Write και περιμένουμε να ολοκληρωθεί η εγγραφή του λειτουργικού Raspbian στην MicroSD.



Εικόνα 2.2.7 (εγγραφή του λειτουργικού στην SD)

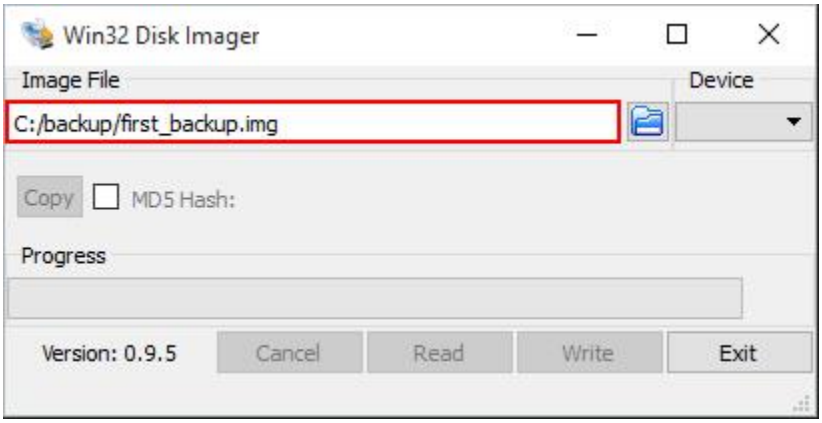
Ύστερα από αυτά τα βήματα μπορούμε πλέον να συνδέσουμε την MicroSDCard στο RaspberryPi και να τρέξουμε το λειτουργικό Raspbian.

2.3 Δημιουργία εφεδρικού αντιγράφου των περιεχομένων της MicroSD κάρτας.

Σε αυτό το παράδειγμα θα παρουσιαστούν τα βήματα που είναι απαραίτητα για την δημιουργία εφεδρικού αντιγράφου του λειτουργικού συστήματος καθώς και όλων των αρχείων που βρίσκονται πάνω στην MicroSD.

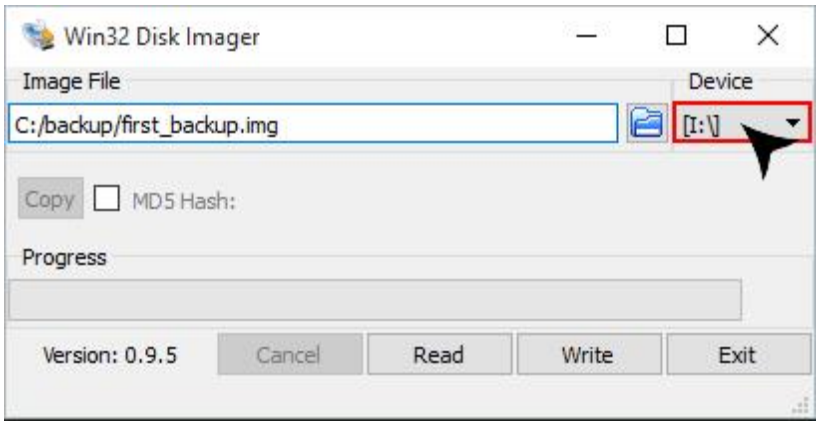
Για την δημιουργία του εφεδρικού αντιγράφου θα χρησιμοποιήσουμε το εργαλείο Win32 Disk Imager.

Συνδέουμε πρώτα τον cardreader με την MicroSD μας στον Υπολογιστή.
Ύστερα ανοίγουμε το Win32 Disk Imager και πληκτρολογούμε την διαδρομή στην οποία θέλουμε να αποθηκευτεί το εφεδρικό αντίγραφο, καθώς και το όνομα που θα έχει το εφεδρικό αντίγραφο μας το οποίο θα πρέπει να τελειώνει σε ".img" όπως για παράδειγμα first_backup.img:



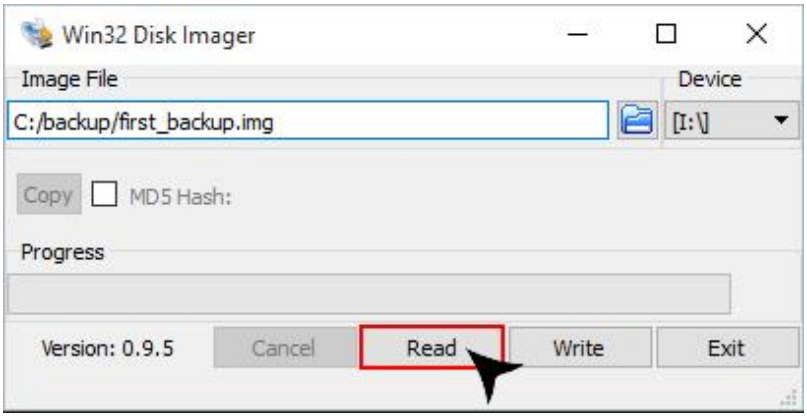
Παράθυρο 2.3.1 (ορισμός διαδρομής και ονόματος αρχείου)

Ύστερα επιλέγουμε την μονάδα δίσκου στην οποία αντιστοιχεί η MicroSD μας:



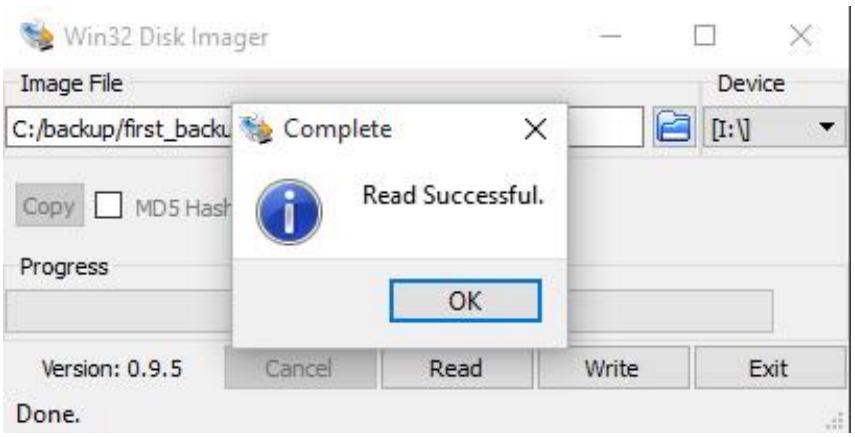
Παράθυρο 2.3.2 (επιλογή μονάδας από την οποία θα παρθεί το backup)

Και ύστερα πατάμε στο κουμπί που γράφει Read:



Παράθυρο 2.3.3 (εκκίνηση λήψης backup)

Περιμένουμε να τελειώσει η διαδικασία αποθήκευσης, και να μας εμφανίσει το μήνυμα “Read Successful.”



Παράθυρο 2.3.4 (επιτυχής ολοκλήρωση της διαδικασίας)

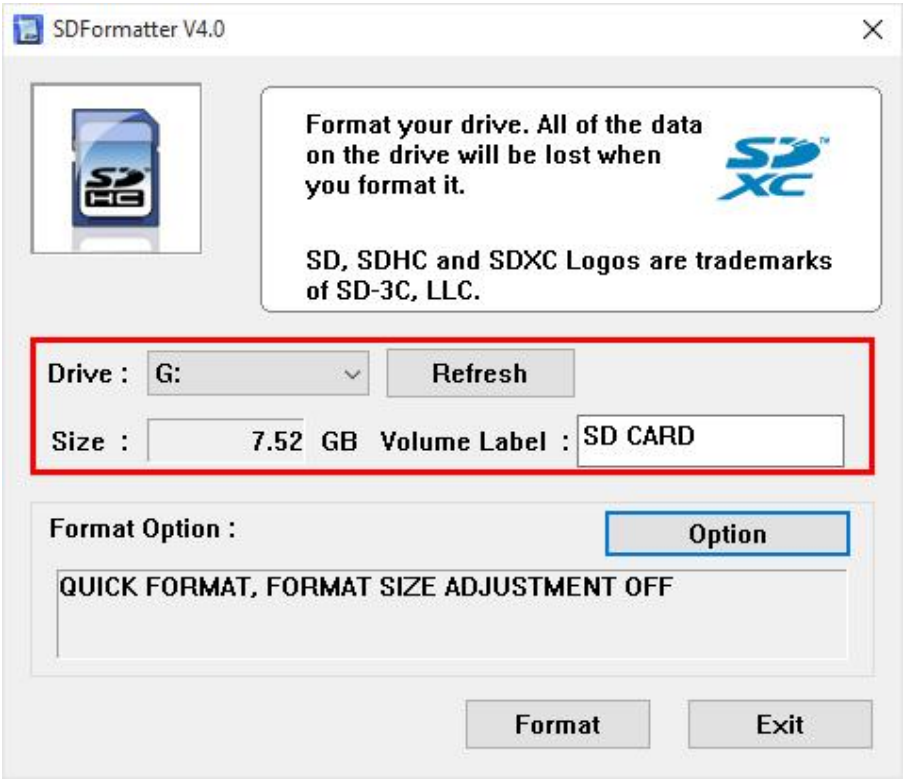
2.4 Διαμόρφωση της κάρτας microSD, έπειτα από εγκατάσταση Raspbian

2.4.1 Γρήγορη διαμόρφωση με το εργαλείο SD Formatter

Η χρήση του εργαλείου SDFormatter διευκολύνει την διαμόρφωση του αποθηκευτικού μας μέσου σε περιβάλλον Windows ύστερα από εγκατάσταση Raspbian σε αυτό. Για να κατεβάσουμε το βοηθητικό πρόγραμμα θα πρέπει να επισκεφτούμε τον ισότοπο: https://www.sdcard.org/downloads/formatter_4/eula_windows/index.html.

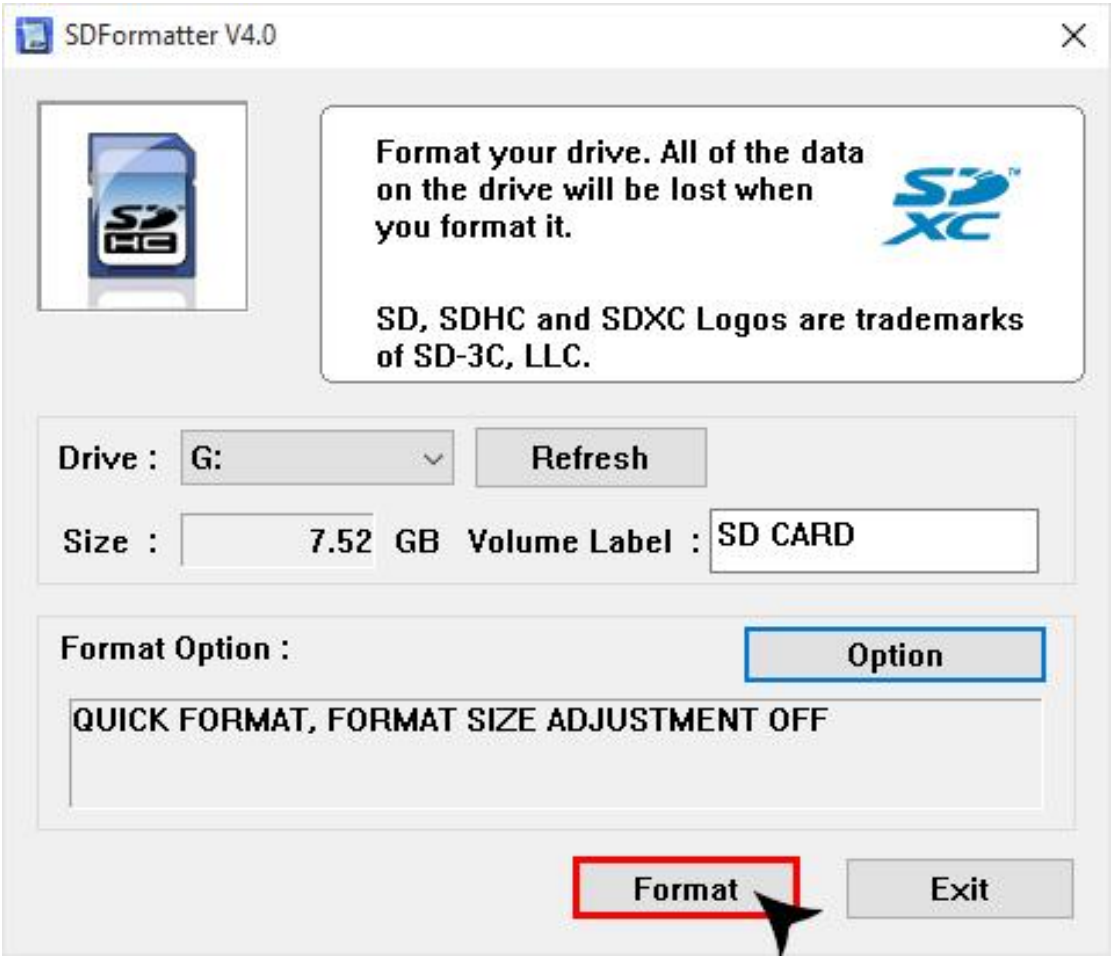
Τα βήματα που είναι απαραίτητα για την διαμόρφωση της MicroSD με το εργαλείο SDFormatter είναι τα εξής:

Πρώτα πρέπει να συνδέσουμε το μέσο αποθήκευσης στο οποίο θέλουμε να εφαρμόσουμε διαμόρφωση. Ύστερα τρέχουμε την εφαρμογή SDFormatter και φροντίζουμε να επιλέξουμε το μέσο στο οποίο θέλουμε να εφαρμόσουμε διαμόρφωση.



Παράθυρο 2.4.1.1 (επιλογές σχετικά με το μέσο αποθήκευσης)

Μόλις είμαστε σίγουροι ότι επιλέξαμε το σωστό μέσο αποθήκευσης πατάμε το κουμπί διαμόρφωσης (Format), και περιμένουμε να ολοκληρωθεί η διεργασία.



Παράθυρο 2.4.1.2 (εκκίνηση διαμόρφωσης)

Η MicroSD μας έχει διαμορφωθεί πλέον και είναι έτοιμη για χρήση.

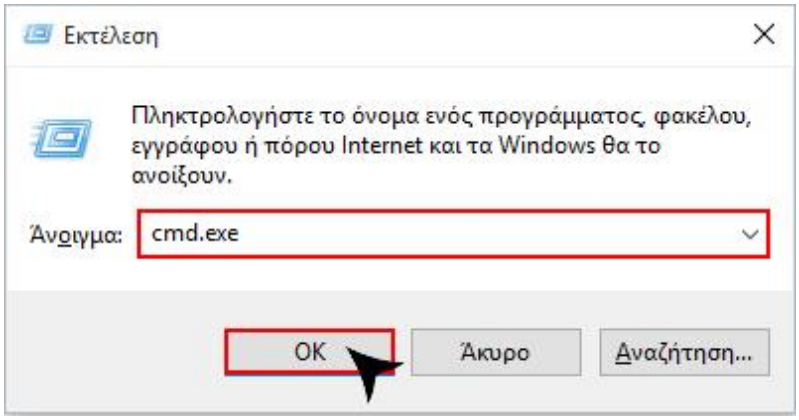
2.4.2 Διαμόρφωση της κάρταςMicroSD μέσω εργαλείων των Windows.

Ο τρόπος διαμόρφωσης της κάρτας MicroSD για χρήστες Windows χωρίς την εγκατάσταση βοηθητικών εφαρμογών είναι λίγο σύνθετος μιας και δεν είναι εφικτός από το κλασσικό μενού διαμόρφωσης που προσφέρουν τα Windows.

Τα βήματα που είναι απαραίτητα για την διαμόρφωση τις MicroSD είναι τα εξής:

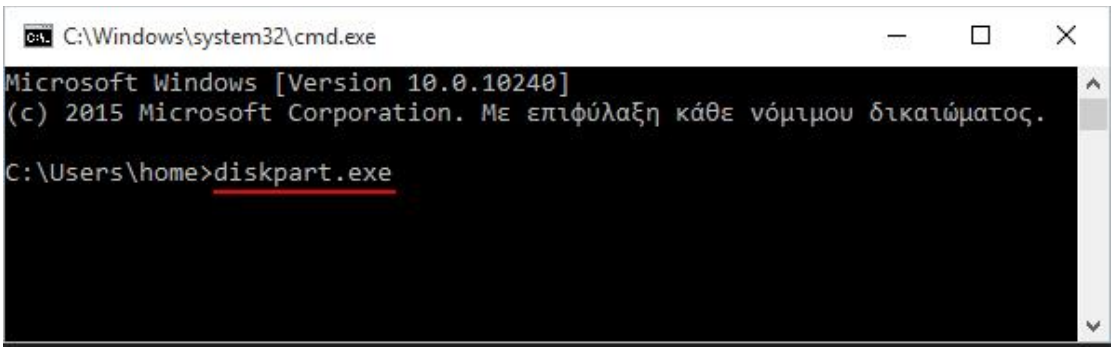
Πρώτα πρέπει να συνδέσουμε το μέσο αποθήκευσης στο οποίο θέλουμε να εφαρμόσουμε διαμόρφωση.

Ύστερα πρέπει να τρέξουμε το CommandPrompt πατώντας τα πλήκτρα WIN+R(WIN είναι το πλήκτρο που έχει το λογότυπο τον Windows). Στο παράθυρο που θα μας ανοίξει θα πληκτρολογήσουμε “cmd.exe” και πατάμε OK.



Παράθυρο 2.4.2.1 (παράθυρο εκτέλεσης)

Μόλις πατήσουμε OK θα ανοίξει ένα παράθυρο CMD. Στο οποίο θα πληκτρολογήσουμε “diskpart.exe” και ύστερα θα πατήσουμε Enter.

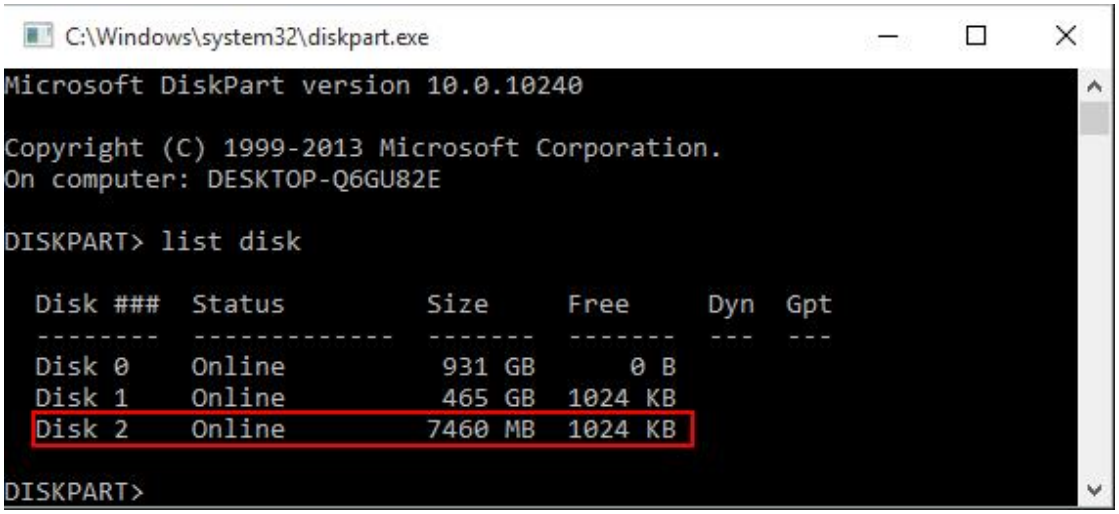


Παράθυρο 2.4.2.2 (εκτέλεση της εφαρμογής diskpart)

Ανάλογα με την έκδοση των Windows που έχουμε, το εργαλείο diskpart θα ανοίξει σε άλλο ένα CMD παράθυρο ή στο ίδιο το παράθυρο που πληκτρολογήσαμε την προηγούμενη εντολή. Στο παράθυρο του diskpart πληκτρολογούμε “listdisk” και πατάμε Enter.

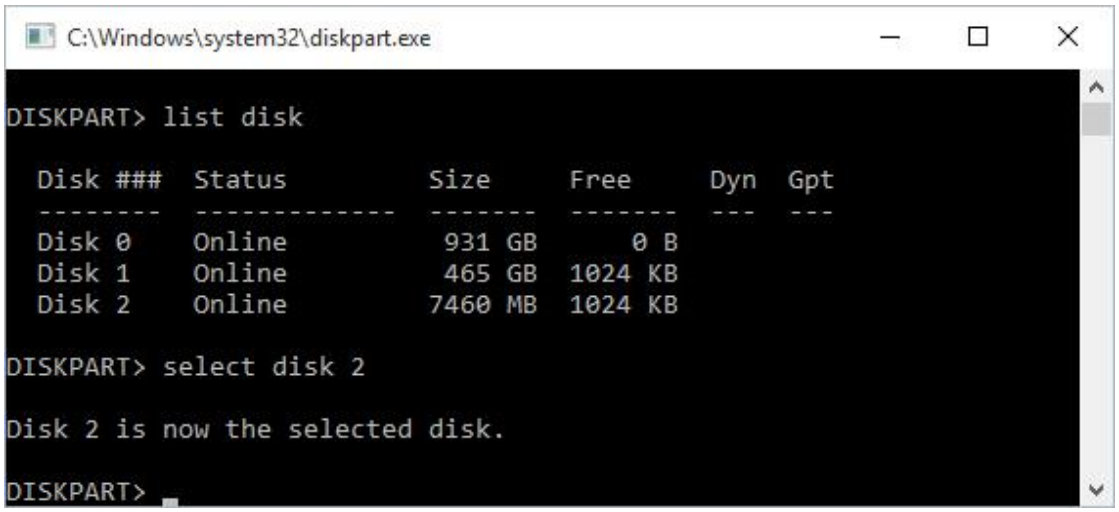
Στην λίστα που θα εμφανιστεί πρέπει να βρούμε ποιος από τους δίσκους είναι η MicroSD μας, αυτό είναι εφικτό κοιτώντας το μέγεθος του δίσκου, ο δίσκος αυτού του παραδείγματος που φαίνεται στο παράθυρο 2.4.2.3 έχει μέγεθος 8GB άρα είναι ο τελευταίος μιας και 7460MB είναι περίπου 8GB.

Σημείωση: Καλό θα ήταν να αφαιρεθούν όλες οι περιττές αποθηκευτικές μονάδες ώστε να αποφευχθεί σύγχυση, αλλά και ο κίνδυνος να διαγραφούν δεδομένα σε λάθος αποθηκευτικό μέσο.



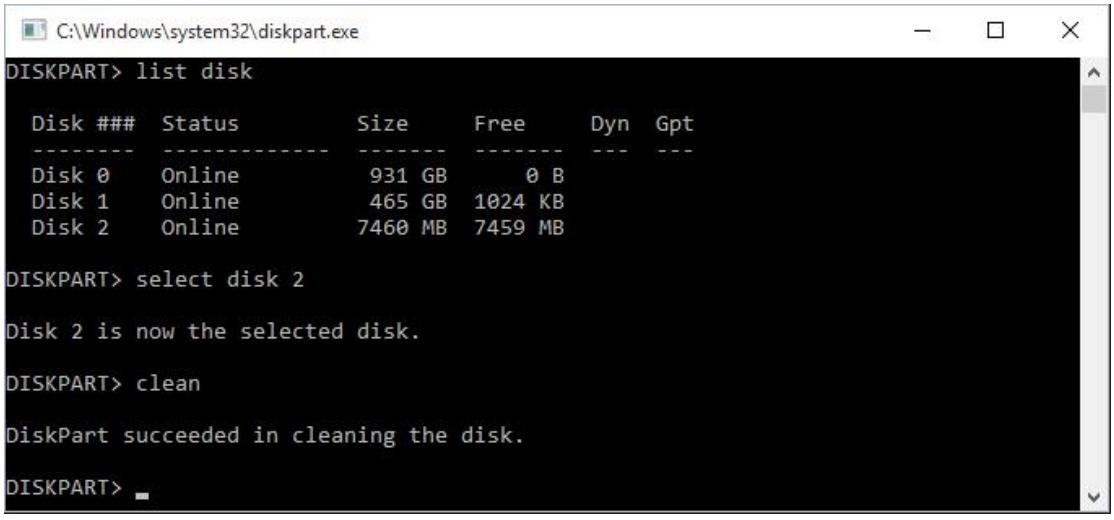
Παράθυρο 2.4.2.3 (εύρεση του σωστού μέσου)

Μόλις σιγουρευτούμε ότι βρήκαμε τον σωστό δίσκο πληκτρολογούμε την εντολή “select disk X”, όπου “X” ο αριθμός του δίσκου μας στην λίστα, στην περίπτωση του παραδείγματος μας είναι ο “2”. Ύστερα πατάμε Enter.



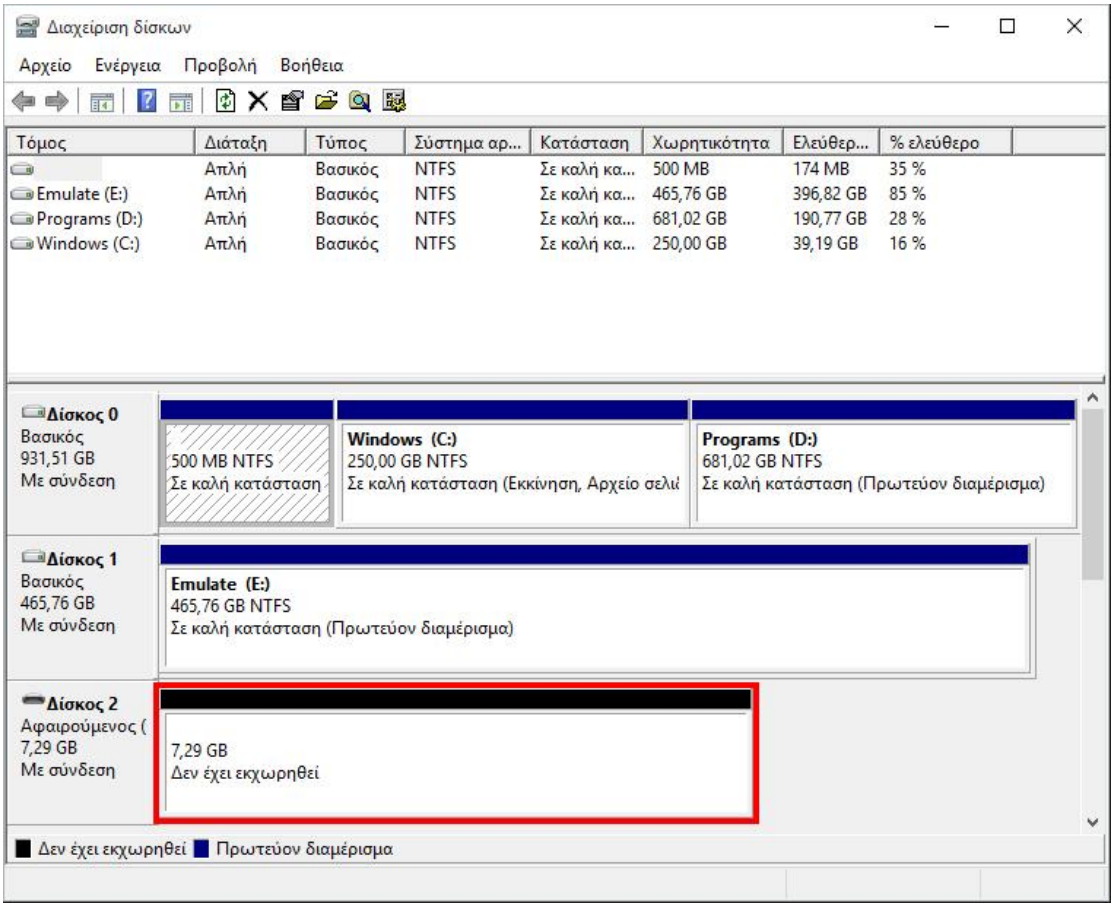
Παράθυρο 2.4.2.4 (επιλογή του μέσου)

Αφού επιλέξαμε τον σωστό δίσκο πληκτρολογούμε την εντολή “clean”, πατάμε Enter και περιμένουμε μέχρι να μας εμφανίσει το μήνυμα “DiskPart succeeded in cleaning the disk.”. Αν εμφανίσει μήνυμα σφάλματος, ότι δεν επιτρέπεται η πρόσβαση, θα πρέπει να αφαιρέσουμε το μέσο και να το επανασυνδέσουμε, και να επαναλάβουμε όλα τα βήματα από την αρχή.
Αν πετύχει η διαγραφή κλείνουμε το παράθυρο του diskpart.



Παράθυρο 2.4.2.5 (εκκαθάριση του μέσου)

Τώρα έμεινε να δεσμεύσουμε τον αποθηκευτικό χώρο του μέσου μας και να το διαμορφώσουμε σε Fat32. Για να το κατορθώσουμε αυτό πατάμε τα πλήκτρα WIN+R(WIN είναι το κουμπί που έχει το λογότυπο τον Windows). Στο παράθυρο που θα μας ανοίξει θα πληκτρολογήσουμε “diskmgmt.msc” και ύστερα OK. Και θα μας εμφανίσει το εξής παράθυρο, στο οποίο θα ψάξουμε να βρούμε τον δίσκο στον οποίο γραφεί “Δεν έχει εκχωρηθεί” και κάνουμε δεξί click πάνω του:



Παράθυρο 2.4.2.6 (επιλογή μη εκχωρημένου δίσκου)

Στο παράθυρο που θα εμφανίσει αφότου κάναμε δεξί click πάνω του θα επιλέξουμε “Νέος απλός τόμος…”, στον οδηγό που θα μας εμφανίσει θα πατάμε “Επόμενο” μέχρι να τελειώσει, και πατάμε “Τέλος”. Η MicroSD μας έχει διαμορφωθεί πλέον και είναι έτοιμη για χρήση.

Κεφάλαιο 3

Λειτουργικό σύστημα

Σε αυτό το κεφάλαιο θα γίνει μια παρουσίαση του λειτουργικού Raspbian, αναφορά στις σημαντικότερες ρυθμίσεις, στους τρόπους εγκατάστασης/απεγκατάστασης εφαρμογών, καθώς και στην αναβάθμιση του λειτουργικού συστήματος. Επίσης θα γίνει μια αναφορά άλλων αξιόλογων λειτουργικών συστημάτων που υπάρχουν για το Raspberry Pi.

Τι λειτουργικό;

Το λειτουργικό που χρησιμοποιείται σε αυτήν την πτυχιακή για την υλοποίηση εφαρμογών στο Raspberry Pi είναι το Raspbian. Το Rasbian είναι μια τροποποιημένη έκδοση του λειτουργικού συστήματος Debian, προσαρμοσμένη για χρήση με το Raspberry Pi.

Γιατί Raspbian;

Διότι είναι το βασικό λειτουργικό σύστημα που διατίθεται δωρεάν, το οποίο δημιουργήθηκε από τους δημιουργούς του raspberry, και ο σκοπός του είναι να αξιοποιήσει όλες τις ικανότητες της συσκευής. Παρέχει συμβατότητα με πολλές συσκευές που μπορούμε να συνδέσουμε μέσω USB όπως κάμερες, εξωτερικές αποθηκευτικές μονάδες, ασύρματοι ελεγκτές κτλ.

Συμβατότητα;

Το λειτουργικό Raspbian στην τελευταία του έκδοση (Jessie) είναι συμβατό με όλες τις εκδόσεις Raspberry Pi. Έτσι έχουμε την δυνατότητα να τρέχουμε τα προγράμματα που θα γράψουμε σε μια έκδοση του Raspberry Pi (όπως το Raspberry Pi B+ το οποίο χρησιμοποιούμε σε αυτή την πτυχιακή) σε όλες τις άλλες εκδόσεις με μικρές αλλαγές στον κώδικα (λόγω του ότι οι ακροδέκτες GPIO διαφέρουν από μοντέλο σε μοντέλο).

3.1 Η Επιφάνεια εργασίας του Raspbian

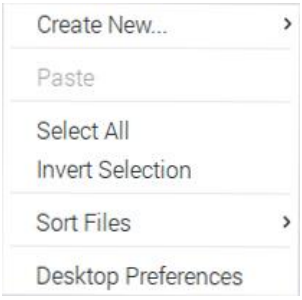
Στην εικόνα 3.1.1 βλέπουμε το απλό περιβάλλον εργασίας του λειτουργικού Raspbian.

Εικόνα 3.1.1 (επιφάνεια εργασίας του raspbian)

Στοιχεία της επιφάνειας εργασίας:

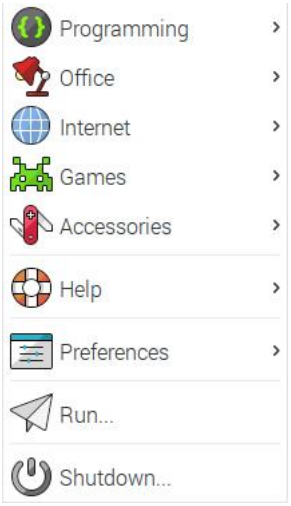
1. Επιφάνεια εργασίας
2. Μενού έναρξης
3. Εικονίδια επιφάνειας εργασίας
4. Γραμμή γρήγορης εκκίνησης
5. Περιοχή ειδοποιήσεων
6. Αποσύνδεση περιφερειακών συσκευών

3.1.1 Το μενού του δεξιού κλικ



Από το μενού του δεξιού κλικ έχουμε τις επιλογές να δημιουργήσουμε νέα αρχεία και φακέλους, αλλά και τις επιλογές αντιγραφής, αποκοπής , επικόλλησης, ταξινόμησης, πολλαπλής επιλογής καθώς και το άνοιγμα των ρυθμίσεων της επιφάνειας εργασίας. Το μενού αυτό προσαρμόζεται ανάλογα με το πού θα γίνει δεξί κλικ.

3.1.2 Το μενού έναρξης



Στο μενού έναρξης μπορούμε να βρούμε όλες τις εγκατεστημένες εφαρμογές κατηγοριοποιημένες. Επίσης έχουμε πρόσβαση στις ρυθμίσεις του συστήματος, στο μενού εκτέλεσης(Run) αλλά και στο μενού απενεργοποίησης του συστήματος.

3.1.3 Οι ρυθμίσεις της επιφάνειας εργασίας

Οι ρυθμίσεις επιφάνειας εργασίας είναι προσβάσιμες πατώντας δεξί κλικ σε κενό χώρο της επιφάνειας εργασίας και επιλέγοντας “Desktop Preferences”, είτε από το μενού έναρξης Preferences/Appearance Settings. Αυτές οι ρυθμίσεις μας επιτρέπουν να αλλάξουμε το φόντο της επιφάνειας εργασίας, καθώς και την εμφάνιση και γραμματοσειρά του συστήματος.

3.1.4 Ρυθμίσεις ζώνης ώρας, πληκτρολογίου και περιοχής WIFI

Από την έναρξη επιλέγουμε Preferences/Raspberry Pi Configuration και στο παράθυρο που ανοίγει πηγαίνουμε στην καρτέλα Localisation. Για να ορίσουμε την ζώνη ώρας πατάμε στο “Set Timezone” και στο παράθυρο που θα μας ανοίξει επιλέγουμε Area το “Europe” και Location το “Athens” και πατάμε OK. Για την ρύθμιση του πληκτρολογίου επιλέγουμε “Set Keyboard” και στο παράθυρο που θα

ανοίξει επιλέγουμε από την λίστα “Country” το United States, ύστερα επιλέγουμε από την λίστα “Variant” το English(US, with euro on 5) και πατάμε OK. Τελειώνοντας για να ορίζουμε την περιοχή WiFi επιλέγουμε “Set WiFi Country” και στο παράθυρο που ανοίγει επιλέγουμε GR Greece και πατάμε OK. Πατάμε OK και στο παράθυρο των ρυθμίσεων και επιλέγουμε να κάνει επανεκκίνηση το σύστημα ώστε να εφαρμοστούν οι αλλαγές.

3.1.5 Ρυθμίσεις οθόνης

Για να αλλάξουμε τις ρυθμίσεις ανάλυσης της οθόνης ανοίγουμε τις ρυθμίσεις από την έναρξη > Preferences> Raspberry Pi Configuration. Στο παράθυρο που μας ανοίγει επιλέγουμε Set Resolution για να αλλάξουμε την ανάλυση. Αν έχουμε ένα μαύρο πλαίσιο γύρω από την οθόνη τότε θα πρέπει να απενεργοποιήσουμε την επιλογή Underscan που βρίσκεται στο κάτω μέρος του παράθυρου.

3.2 Το σύστημα αρχείων

Για να έχουμε πρόσβαση στο σύστημα αρχείων μπορούμε να πατήσουμε το εικονίδιο με το σύμβολο φακέλων () από την γραμμή γρήγορης εκκίνησης ή να επιλέξουμε από την έναρξη Accessories/File Manager. Στην συνέχεια, εξηγούνται οι βασικοί κατάλογοι του συστήματος.

boot: — Εδώ υπάρχει ο πυρήνας του λειτουργικού (Linux kernel) αλλά και άλλες βοηθητικές εφαρμογές που είναι απαραίτητες για την εκκίνηση του συστήματος.

bin: — Σημαντικά αρχεία του λειτουργικού συστήματος, όπως τα αρχεία για να λειτουργήσει το γραφικό περιβάλλον διεπαφής χρήστη είναι αποθηκευμένα εδώ.

dev: Αυτός είναι εικονικός κατάλογος, τα στοιχεία αυτού του καταλόγου είναι η φυσικές συσκευές που είναι συνδεδεμένες στο σύστημα μας.

etc: Σε αυτόν τον φάκελο περιέχονται οι ρυθμίσεις του συστήματος καθώς και το αρχείο χρηστών με τους κρυπτογραφημένους κωδικούς.

home: Μέσα στον κατάλογο home δημιουργείται ένας υποκατάλογος για κάθε χρήστη του συστήματος, όπου αποθηκεύονται τα προσωπικά αρχεία του κάθε χρήστη. Μέσα σε αυτόν τον κατάλογο βρίσκεται και ο κατάλογος pi που είναι ο κατάλογος του τρέχοντος χρήστη όταν εκκινούμε το σύστημα για πρώτη φόρα.

lib: Εδώ αποθηκεύονται βιβλιοθήκες, που είναι απαραίτητες για την εκτέλεση διάφορων εφαρμογών.

lost+found: Σε αυτό τον φάκελο αποθηκεύονται κομμάτια αρχείων σε περίπτωση που το σύστημα καταρρεύσει.

man: Εδώ αποθηκεύονται τα εγχειρίδια χρήσης των εγκατεστημένων εφαρμογών.

media: Αυτός ο κατάλογος είναι υπεύθυνος να εμφανίζει τα προσωρινά αφαιρούμενα μέσα αποθήκευσης που συνδέονται στο σύστημα.

mnt: Αυτός ο κατάλογος είναι υπεύθυνος να εμφανίζει μόνιμα μέσα αποθήκευσης, όπως εξωτερικούς σκληρούς δίσκους.

opt: Εδώ είναι ο χώρος στον οποίο αποθηκεύονται οι εφαρμογές που εγκαθιστούν οι χρήστες.

proc: Αυτός είναι ένας εικονικός κατάλογος, στον οποίο περιέχονται πληροφορίες σχετικά με τις εφαρμογές που είναι υπό εκτέλεση, γνωστές και ως διεργασίες.

sbin: Σε αυτόν τον κατάλογο είναι αποθηκευμένα πηγαία αρχεία που είναι απαραίτητα για την συντήρηση του λειτουργικού συστήματος

sys: Εδώ αποθηκεύονται σημαντικά αρχεία του λειτουργικού συστήματος.

tmp: Εδώ αποθηκεύονται αυτόματα τα προσωρινά αρχεία.

usr: Αυτός ο κατάλογος προσφέρει αποθηκευτικό χώρο σε εφαρμογές προσβάσιμες από τους χρήστες.

var: Αυτός είναι ένας εικονικός κατάλογος στον οποίο οι εφαρμογές αποθηκεύουν κάποιες μεταβλητές περιβάλλοντος.

3.3 Το τερματικό Linux (terminal/bash)

Το τερματικό είναι ένα από τα πιο σημαντικά εργαλεία που μας προσφέρει το λειτουργικό σύστημα, μιας και μέσω αυτού μας είναι εφικτό να διευκολύνουμε πολλές διεργασίες. Για να έχουμε πρόσβαση στο τερματικό μπορούμε να πατήσουμε το εικονίδιο με το σύμβολο γραμμής εντολών () από την γραμμή γρήγορης εκκίνησης ή να επιλέξουμε από την έναρξη Accessories/Terminal.

3.3.1 Βασικές Εντολές

Θα εξηγηθούν κάποιες σημαντικές εντολές που είναι απαραίτητες για την χρήση του Raspberry.

Αυτό είναι το βασικό σώμα για τις περισσότερες εντολές του τερματικού:

[sudo] **εντολή** [προαιρετική παράμετρος] [διεύθυνση αρχείου ή φακέλου]

Εδώ υπενθυμίζονται οι γνωστές εντολές του τερματικού:

sudo: Μπαίνει ως πρόθεμα σε κάθε εντολή που θέλουμε να εκτελέσουμε με δικαιώματα υπερχρήστη.

Παράδειγμα: sudo apt-get update ↲

cd: Περιήγηση στους φακέλους του συστήματος.

Παραδείγματα: cd ↲ *#Μεταφέρει των χρήστη στον κατάλογο home* cd / ↲ *#Μεταφέρει των χρήστη στον κατάλογο root* cd .. ↲ *#Μεταφέρει των χρήστη έναν κατάλογο πιο πάνω από εκεί που βρίσκεται* cd /home/desktop ↲ *#Μεταφέρει των χρήστη στον κατάλογο desktop χρησιμοποιώντας την απόλυτη διαδρομή*

cp: Αντιγραφή αρχείων και φακέλων

Δομή εντολής:

cp <αρχείο> <αρχείο2>

Παράδειγμα:

cp tst1.py /home/desktop/folder1 ↲

Εξήγηση: Το αρχείο tst1.py θα αντιγράφει στον κατάλογο folder1 που βρίσκεται στην διαδρομή /home/desktop/

man: Προβολή εγχειριδίου βοήθειας εντολής/προγράμματος. Μπαίνει ως πρόθεμα σε κάθε εντολή, της οποίας θέλουμε να διαβάσουμε το εγχειρίδιο χρήσης.

Δομή εντολής:

man <εντολή>

Παράδειγμα:

man cp ↲

Εξήγηση: Εμφανίζει το εγχειρίδιο χρήσης της εντολής cp

mkdir: Δημιουργία φακέλου.

Δομή εντολής:

mkdir <όνομα_ νέου_ καταλόγου>

Παράδειγμα:

mkdir folder1 ↲

Εξήγηση: Δημιουργήθηκε ένας νέος κατάλογος με όνομα folder1

rm: Διαγραφή αρχείων.

Δομή εντολής:

rm <όνομα_ αρχείου>

Παράδειγμα:

rm tst1.py ↲

Εξήγηση: Διαγράφηκε το αρχείο με όνομα tst1.py

rmdir: Διαγραφή φακέλου υπό την προϋπόθεση ότι αυτός είναι κενός.

Δομή εντολής:

rmdir <όνομα_ φακέλου>

Παράδειγμα:

rmdir folder1 ↲

Εξήγηση: Διαγράφηκε ο φάκελος με όνομα folder1

chmod: Αλλάζει τα δικαιώματα χρήσης των αρχείων.

Δομή εντολής:

chmod [ομάδα_ χρηστών][τελεστής][δικαιώματα] <όνομα_ αρχείου>

Στην **ομάδα_ χρηστών** ορίσουμε σε ποιους θα απευθύνονται τα δικαιώματα.

Η πιθανές επιλογές είναι **u** (ιδιοκτήτης), **g** (ομάδα που έχει πρόσβαση στον κατάλογο), **o** (οι υπόλοιποι χρήστες του συστήματος), **a** (όλοι οι χρήστες του συστήματος).

Στον **τελεστή** ορίζουμε αν θα προστεθούν η αν θα αφαιρεθούν δικαιώματα.

Η πιθανές επιλογές είναι **+** (πρόσθεση δικαιωμάτων) και **–** (αφαίρεση δικαιωμάτων)

Ως **δικαιώματα** ορίζουμε τα δικαιώματα που θα έχουνε οι χρήστες πάνω στο αρχείο.

Η πιθανά δικαιώματα είναι **r** (ανάγνωσης), **w** (εγράφης), **x** (εκτέλεσης).

Παράδειγμα:

chmod gu+rx folder1 ↲

Εξήγηση: Προσθήκη δικαιωμάτων ανάγνωσης και εκτέλεσης για το αρχείου folder1 από τον ιδιοκτήτη και την ομάδα που έχει πρόσβαση στον κατάλογο.

pwd: Εκτύπωση της τρέχουσας διαδρομής.

Παράδειγμα: pwd ↲ Έξοδος: /home/pi

whoami: εκτυπώνει το username μας.

Παράδειγμα: whoami ↲ Έξοδος: pi

reboot: επανεκκινεί το σύστημα.

shutdown: απενεργοποιεί το σύστημα.

nano: Απλός επεξεργαστής κειμένου, ανοίγει έγγραφα κειμένου πάνω στο ίδιο το τερματικό, και μας δίνει την δυνατότητα να τα επεξεργαστούμε.

nano [**αρχείο κειμένου**]

Μπορούμε επίσης να τρέξουμε το nano χωρίς παραμέτρους, για να δημιουργήσουμε ένα νέο αρχείο κειμένου.
Αρκεί να θυμάστε πως οι εντολές που αναγράφονται στο κάτω μέρος λειτουργούν όλες σε συνδυασμό με το Ctrl, δηλαδή πατάμε Ctrl+X για να βγούμε από το πρόγραμμα, Ctrl+G για τη βοήθεια κλπ.

leafpad: Επεξεργαστής κειμένου με οπτικό περιβάλλον, ανοίγει έγγραφα κειμένου παρέχοντας μας και ένα οπτικό περιβάλλον και εργαλεία διευκολύνοντας την επεξεργασία κειμένου.

leafpad [**αρχείο κειμένου**]

Μπορούμε επίσης να τρέξουμε το leafpad χωρίς παραμέτρους, για δημιουργήσουμε ένα νέο αρχείο κειμένου.

git: Η συγκεκριμένη εφαρμογή μας επιτρέπει να διαχειριζόμαστε κώδικα στο αποθετήριο git hub. Παρακάτω θα εξηγηθεί μόνο η εντολή clone της εφαρμογής git που θα μας χρησιμεύσει ώστε να κατεβάζουμε έτοιμο κώδικα.

Δομή εντολής:

git clone [URL_ διεύθυνση _του αποθετηρίου]

Παράδειγμα:

git clone https://github.com/scanner/HDC100X_Python.git ↲

apt-get: Αυτή η εντολή μας επιτρέπει την εγκατάσταση/απεγκατάσταση και αναβάθμιση εφαρμογών, καθώς και την αναβάθμιση του λειτουργικού συστήματος μας. Προκειμένου να λειτουργήσει η συγκεκριμένη εντολή θα πρέπει να χρησιμοποιηθεί η εντολή sudo ως πρόθεμα. Στην συνέχεια θα παρουσιαστούν μερικές χρήσιμες παράμετροι αυτής της εντολής. Οι παράμετροι για την αναβάθμιση του λειτουργικού θα εξηγηθούν σε ξεχωριστή ενότητα.

Αυτή η εντολή εγκαθιστά μια νέα εφαρμογή, εάν το όνομα της βρίσκεται στην λίστα του αποθετηρίου (repository).

```
sudo apt-get install <όνομα_εφαρμογής>
```

Αυτή η εντολή επιδιορθώνει μια κατεστραμμένη εφαρμογή.

```
sudo apt-get -f install <όνομα_εφαρμογής>
```

Αυτή η εντολή αφαιρεί την καθορισμένη εφαρμογή από το σύστημα.

```
sudo apt-get remove <όνομα_εφαρμογής>
```

Αυτή η εντολή καθαρίζει απομεινάρια εφαρμογών που απεγκαταστάθηκαν από το σύστημα.

```
sudo apt-get autoclean
```

Οι εντολές python και bash προϋποθέτουν ότι βρισκόμαστε στον ίδιο κατάλογο με τα αρχεία που θέλουμε να εκτελέσουμε.

python3 και **python:** Με αυτές τις εντολές μας είναι εφικτό να εκτελούμε κώδικα από python αρχεία. Ένα αρχείο python έχει κατάληξη ".py". Δυστυχώς δεν είναι εφικτό να διακρίνουμε σε ποια έκδοση python έχει γραφτεί ένα αρχείο python από την κατάληξη του.

```
python <όνομα_ αρχείου >.py #Για αρχείο Python2
python3 <όνομα_ αρχείου >.py # Για αρχείο Python3
```

bash: Με αυτήν την εντολή μπορούμε να εκτελούμε αρχεία με κώδικα bash. Ένα αρχείο bash έχει κατάληξη ".sh".

```
bash <όνομα_ αρχείου >.sh
```

3.3.2 Δημιουργία και χρήση ενός αρχείου bash

Το αρχείο bash είναι ένα αρχείο το οποίο περιέχει εντολές του τερματικού, με σκοπό να αυτοματοποιήσει μια διαδικασία. Όπως για παράδειγμα η ομαδική μετονομασία αρχείων, ή η ακολουθία εκτέλεσης συγκεκριμένων εντολών.
Για να δημιουργήσουμε ένα αρχείο bash στο λειτουργικό Raspbian, θα πρέπει να κάνουμε δεξί κλικ στην επιφάνεια εργασίας, και στο μενού που θα ανοίξει επιλέγουμε Create New/Empty File. Στο παράθυρο που θα ανοίξει πληκτρολογούμε ένα όνομα και προσθέτουμε στο τέλος του ονόματος την κατάληξη ".sh".

Για να επεξεργαστούμε το αρχείο που δημιουργήσαμε προηγούμενος θα πρέπει να κάνουμε δεξί κλικ πάνω του, και στο μενού επιλογών που θα ανοίξει να επιλέξουμε "Text Editor".

Υπενθυμίζεται ότι όλες οι εντολές του τερματικού είναι εφικτό να χρησιμοποιηθούνε σε ένα bash αρχείο. Στην συνέχεια θα εξηγηθούν οι βασικές εντολές που είναι απαραίτητες για την συγγραφή κώδικα σε bash αρχείο:

#!/bin/bash : Η συγκεκριμένη γραμμή κώδικα μπαίνει πάντα στην κορυφή ενός προγράμματος bash και βοηθάει το τερματικό να καταλάβει με ποια εφαρμογή να εκτελέσει τον κώδικα που στην περίπτωση μας είναι του bash.

#: Το σύμβολο της δίεσης μας επιτρέπει να προσθέτουμε σχόλια στον κώδικα μας.

echo: Η συγκεκριμένη εντολή επιτρέπει να τυπώνουμε κείμενο στο παράθυρο του τερματικού. Αυτό είναι χρήσιμο στην περίπτωση που η εφαρμογή θα επιστρέφει κάποια αποτελέσματα.

Δομή εντολής:

echo “κείμενο “

Παράδειγμα:

echo “hello world“ ↲

Έξοδος: hello world

Ακολουθεί παράδειγμα ενός bash αρχείου.

Ας υποθέσουμε ότι δημιουργήσαμε ένα αρχείο bash με όνομα file1.sh στην επιφάνεια εργασίας (διαδρομή επιφάνειας εργασίας: “/home/pi/Desktop”).

Για να τρέξουμε το αρχείο file1.sh θα πρέπει να ανοίξουμε ένα τερματικό και να πάμε στον κατάλογο στον οποίο βρίσκεται το αρχείο file1.sh με την χρήση της εντολής “cd”.

Μόλις βρεθούμε στον ίδιο κατάλογο με το file1.sh (με την εντολή “ls” ελέγχουμε τα αρχεία που βρίσκονται στον κατάλογο) είμαστε πλέον σε θέση να το τρέξουμε.

Για να τρέξουμε ένα αρχείο bash θα πρέπει να του δώσουμε δικαιώματα εκτέλεσης με την εντολή “chmod”. Ύστερα είμαστε σε θέση να το τρέξουμε χρησιμοποιώντας το πρόθεμα “./” και άμεσος μετά το όνομα του αρχείου.

Άρα ο κώδικας που θα πληκτρολογήσουμε στο τερματικό για να τρέξουμε το αρχείο file1.sh θα είναι:

```
cd /home/Desktop
chmod +x file1.sh
./file1.sh
```

Ας υποθέσουμε ότι το αρχείο file1.she περιέχει τον εξής κώδικα:

```
1:  #!/bin/bash
2:  echo “Hello World!”
3:  pwd
```

Αν τρέξουμε το αρχείο θα μας εμφανίσει:

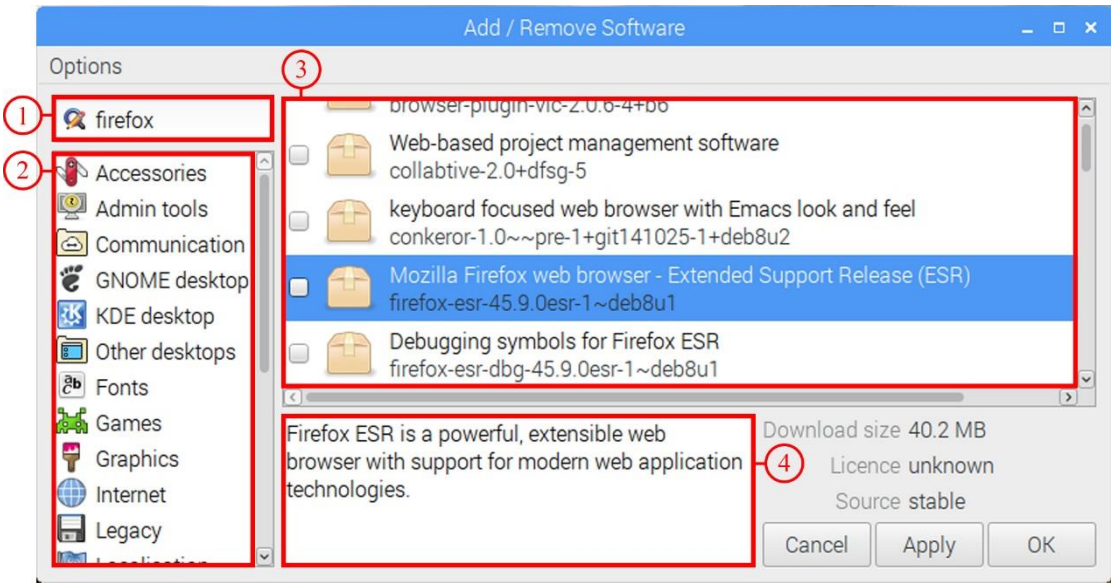
```
Hello World!
home/pi/Desktop
```

3.4 Εγκατάσταση/Απεγκατάσταση Εφαρμογών

Το λειτουργικό μάς παρέχει ένα εργαλείο μέσω του οποίου μπορούμε να βρίσκουμε και να κάνουμε εγκατάσταση/απεγκατάσταση εφαρμογών. Το εργαλείο αυτό λέγεται Package Manager και μπορούμε να το τρέξουμε από την έναρξη επιλέγοντας Preferences > Add / Remove Software.

Για να μας είναι εφικτό να χρησιμοποιήσουμε σωστά αυτήν την εφαρμογή καλό θα είναι κάθε φορά που την τρέχουμε, να πηγαίνουμε στο μενού Options > Refresh Package Lists ώστε να ενημερώνουμε την λίστα με τις διαθέσιμες εφαρμογές προς εγκατάσταση.

Παρακάτω απεικονίζεται το περιβάλλον του Package Manager.



Εικόνα 3.4.1 (περιβάλλον του package manager)

Στοιχεία του Package Manager:

1. Αναζήτηση
2. Λίστα κατηγοριών
3. Λίστα εφαρμογών
4. Πληροφορίες επιλεγμένης εφαρμογής

Όπως φαίνεται στην εικόνα 3.4.1 μας είναι εφικτό να βρούμε την εφαρμογή που ψάχνουμε μέσω της λίστας κατηγοριών ή αναζητώντας την. Μόλις βρούμε την εφαρμογή που θέλουμε να εγκαταστήσουμε βάζουμε ένα τικ στο πλαίσιο επιλογής αριστερά της. Είναι εφικτό να επιλέξουμε περισσότερες από μια εφαρμογές για εγκατάσταση. Αν βγάλουμε το τικ από εφαρμογή που είναι εγκατεστημένη τότε αυτή θα αφαιρεθεί στην συνέχεια.

Μόλις τελειώσουμε με την επιλογή των εφαρμογών που θέλουμε να εγκαταστήσουμε/αφαιρέσουμε τότε πατάμε στο Apply που βρίσκετε κάτω δεξιά στο παράθυρο, και θα μας εμφανίσει ένα νέο παράθυρο. Στο πεδίο “password” θα πληκτρολογήσουμε “raspberry”(που είναι ο κωδικός του συστήματος), και ελέγχουμε το “username” να είναι “pi”. Μόλις πατήσουμε OK θα ξεκινήσει η εγκατάσταση/αφαίρεση των εφαρμογών.

3.5 Αναβάθμιση εφαρμογών και λειτουργικού συστήματος

Υπάρχουν δυο τρόποι για να αναβαθμίσουμε τις εφαρμογές μας. Ο ένας είναι μέσω του γραφικού περιβάλλοντος του Package Manager. Ο δεύτερος τρόπος είναι να χρησιμοποιήσουμε το εργαλείο apt-get μέσω του terminal, που είναι ο καλύτερος τρόπος, μιας και μας επιτρέπει να αναβαθμίσουμε και το firmware του Raspberry.

Παρακάτω θα εξηγηθεί μόνο ο τρόπος αναβάθμισης εφαρμογών, λειτουργικού και firmware με την χρήση της εφαρμογής apt-get. Για να χρησιμοποιήσουμε την εφαρμογή apt-get πρέπει να ανοίξουμε το terminal.

Πριν από κάθε αναβάθμιση θα πρέπει πάντα να ανανεώνουμε την λίστα των αποθετηρίων ώστε να βρεθούν οι νεότερες αναβαθμίσεις που είναι διαθέσιμες. Αυτό γίνεται αν τρέξουμε την εντολή:

```
sudo apt-get update
```

Αφού ενημερωθούν οι λίστες των αποθετηρίων μπορούμε να ξεκινήσουμε την αναβάθμιση όλων των εφαρμογών, αλλά και στοιχείων του λειτουργικού συστήματος, τρέχοντας την εξής εντολή:

```
sudo apt-get upgrade
```

Μόλις τρέξουμε αυτήν την εντολή η εφαρμογή θα συγκεντρώσει όλες τις πληροφορίες για την αναβάθμιση και θα τις προβάλει, αν συμφωνούμε και θέλουμε να γίνει η αναβάθμιση τότε πληκτρολογούμε τον χαρακτήρα “Y”(σε αγγλικά) και πατάμε Enter. Μερικές φορές κατά την διάρκεια της εγκατάστασης των ενημερώσεων υπάρχει περίπτωση να εμφανίσει διαλόγους που θα εξηγούν τα νέα στοιχεία της εκάστοτε εφαρμογής, αν συμβεί αυτό ακολουθούμε τα βήματα ώστε να ολοκληρωθεί ο διάλογος και να συνεχιστεί η εγκατάσταση των εφαρμογών. Η διαδικασία της αναβάθμισης μπορεί να διαρκέσει αρκετή ώρα, μόλις τελειώσει καλό είναι να κάνουμε επανεκκίνηση ώστε να εφαρμοστούν οι νέες αλλαγές.

Κατά καιρούς καλό θα είναι να κάνουμε έλεγχο για το αν υπάρχει ολική αναβάθμιση του λειτουργικού συστήματος σε νεότερη έκδοση. Η σύνταξη της εντολής για αυτή τη δουλειά είναι:

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

Τέλος για να αναβαθμίσουμε το firmware του raspberry στην νεότερη έκδοση χρησιμοποιούμε την εντολή:

```
sudo rpi-update
```

3.6 Άλλα αξιόλογα λειτουργικά συστήματα για το Raspberry

Kali Linux : Λειτουργικό σύστημα με σκοπό τον έλεγχο ασφάλειας τοπικών δίκτυων.

Windows 10 IoT Core : Λειτουργικό σύστημα Windows για την δημιουργία-ανάπτυξη IoT (Internet of Things) εφαρμογών. Δυστυχώς τρέχει μέχρι στιγμής μόνο στο Raspberry Pi 2.

Android Things: Λειτουργικό σύστημα της Google για την δημιουργία και ανάπτυξη Android εφαρμογών και συσκευών.

Arch Linux ARM: Λειτουργικό σύστημα Linux που παρέχει τον πλήρη έλεγχο στην παραμετροποίηση του συστήματος από τον χρήστη.

FreeBSD: Λειτουργικό σύστημα που χρησιμοποιείται σε servers, επιτραπέζιους υπολογιστές άλλα και σε ενσωματωμένα συστήματα, είναι ευρέως διαδεδομένο και αξιόπιστο λειτουργικό.

OSMC: Λειτουργικό σύστημα τύπου Media center.

Κεφάλαιο 4

Συνοπτικό Εγχειρίδιο στην γλώσσα προγραμματισμού Python 3

Η ύπαρξη αυτού του εγχειριδίου έχει ως σκοπό να βοηθήσει τον αναγνώστη να είναι σε θέση να κατανοήσει τον κώδικα που θα βρει σε αυτήν την πτυχιακή αλλά και για να δει της ιδιαιτερότητες της γλώσσας προγραμματισμού Python 3 και το συντακτικό της.

Τι είναι η Python?

Η Python είναι μια γλώσσα προγραμματισμού υψηλού επίπεδου η οποία δημιουργήθηκε από τον Ολλανδό Γκβίντο βαν Ρόσσουμ (Guido van Rossum) το 1990.

Ο κύριος σκοπός της είναι η αναγνωσιμότητα του κώδικα της και η ευκολία χρήσης της, που χάρη στο συντακτικό της επιτρέπει στους προγραμματιστές να εκφράσουν έννοιες σε λιγότερες γραμμές κώδικα σε σχέση με άλλες γλώσσες προγραμματισμού, όπως παραδείγματος χάριν η C ή η Java.

Οι έτοιμες βιβλιοθήκες που διαθέτει διευκολύνουν και επιταχύνουν την συγγραφή κώδικα, και επομένως την εκμάθηση της γλώσσας αυτής.

Αξίζει να αναφερθεί ότι η Python αναπτύσσεται ως ανοιχτό λογισμικό (open source) και η διαχείριση της γίνεται από τον μη κερδοσκοπικό οργανισμό Python Software Foundation.

Ποια έκδοση Python θα χρησιμοποιηθεί;

Θα χρησιμοποιηθεί η 3η έκδοση της γλώσσας Python. Θα γίνει εξαίρεση σε κάποια παραδείγματα, όπου θα χρησιμοποιηθούν βιβλιοθήκες της Python2.

Σε τι περιβάλλον θα γράφω κώδικα?

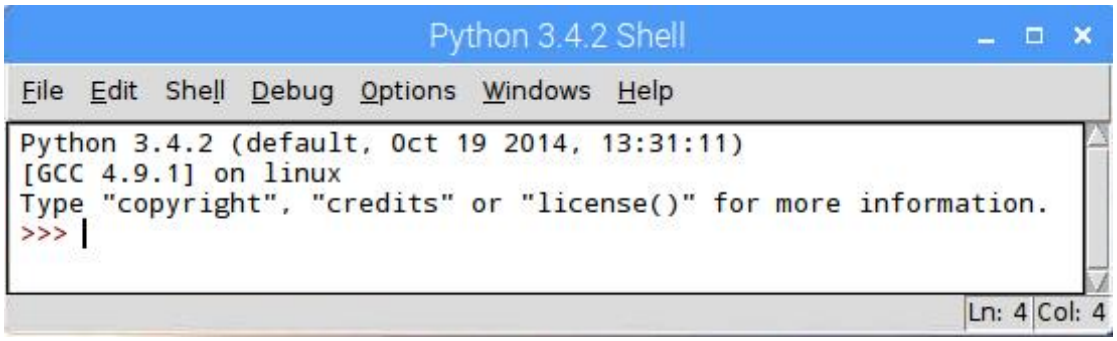
Πλέον στο λειτουργικό Raspbian υπάρχουν προεγκατεστημένα δύο περιβάλλοντα προγραμματισμού, το IDLE που είναι ένα πολύ απλό περιβάλλον και διατίθεται από την ίδια την Python Software Foundation, και το Geany το οποίο είναι ένα ολοκληρωμένο περιβάλλον προγραμματισμού. Θα γίνει παρακάτω μια εισαγωγή στο περιβάλλον IDLE.

4.1 Το περιβάλλον προγραμματισμού IDLE

Το περιβάλλον IDLE είναι η εφαρμογή που θα χρησιμοποιούμε για να γράφουμε και να τρέχουμε εφαρμογές Python.

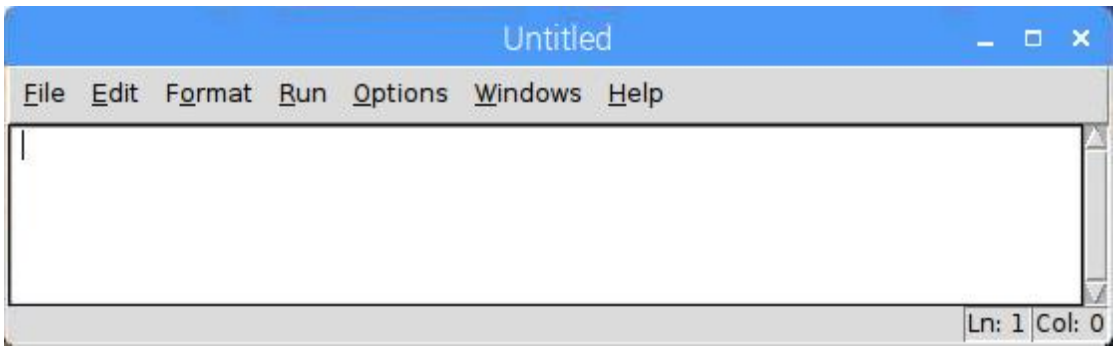
Για να τρέξουμε το περιβάλλον IDLE πηγαίνουμε στην έναρξη και ύστερα στο Programming και επιλέγουμε την έκδοση IDLE που επιθυμούμε. Τα μενού σε και στις δυο εκδόσεις IDLE (2 και 3) είναι τα ίδια.

Όταν ανοίγει το περιβάλλον Idle μας εμφανίζει το command window, στο όποιο μπορούμε να εκτελούμε τον κώδικά μας γραμμή προς γραμμή. Αυτό το παράθυρο χρησιμεύει όταν εκτελούμε μια εφαρμογή και θέλουμε άμεσα να τρέξουμε κάποιες εντολές, που δεν περιλαμβάνονται στον βασικό κώδικα.



Εικόνα 4.1.1 (command window του IDLE)

Για να ανοίξουμε το βασικό παράθυρο (Editor window) στο οποίο είναι εφικτό να γράψουμε προγράμματα, θα πρέπει να πάμε στο μενού File του Command window και να επιλέξουμε New File ή αν έχουμε ήδη ένα αρχείο προγράμματος επιλέγουμε Open… ώστε να το ανοίξουμε.



Εικόνα 4.1.2 (editor window του IDLE)

Το Idle έχει δυο τρόπους εμφάνισης το Shell window που είναι ένα είδος γραμμής εντολών στην οποία μπορούμε να εισάγουμε κώδικα βήμα-βήμα, και το Editor window που είναι ένας κειμενογράφος με ικανότητες αυτόματης συμπλήρωσης και χρωματισμό κειμένου. Είναι προφανές ότι για να συγγράψουμε κώδικα θα χρησιμοποιήσουμε το Editor window το οποίο ανοίγει όταν από το Python Shell δημιουργούμε ή ανοίγουμε ένα αρχείο.

Στην συνέχεια εξηγούνται τα μενού και τον δύο παραθύρων.

4.1.1 Το μενού File

New File: Δημιουργία ενός νέου Editor Window.

Open…: Άνοιγμα ενός υπάρχοντος αρχείου.

Open Module...: Άνοιγμα ενός υπάρχοντος module. (ψάχνει στο sys.path)

Recent Files...: Ανοίγει την λίστα τον πρόσφατα ανοιγμένων αρχείων.

Class Browser: Εμφανίζει της κλάσεις και της μεθόδους του παρόντος αρχείου.

Path Browser: Εμφανίζει τους καταλόγους, modules, κλάσεις και μεθόδους του sys.path

Save: Αποθηκεύει το τρέχον αρχείο.

Save As...: Αποθηκεύει το τρέχον παράθυρο σε νέο αρχείο, και γίνεται το τρέχον αρχείο που επεξεργαζόμαστε.

Save Copy As...: Αποθηκεύει το τρέχον παράθυρο ως αντίγραφο σε νέο αρχείο.

Print Window: Εκτυπώνει το τρέχον παράθυρο.

Close: Κλείνει το τρέχων παράθυρο.

Exit: Κλείνει όλα τα παράθυρα και τερματίζει την εφαρμογή.

4.1.2 Το μενού Edit

Undo: Αναίρεση

Redo: Ακύρωση αναίρεσης.

Cut: Αποκοπή

Copy: Αντιγραφή

Paste: Επικόλληση

Select All: Επιλογή όλων

Find...: Εύρεση

Find Again: Εύρεση επόμενου.

Find Selection: Εύρεση στο επιλεγμένο τμήμα κειμένου.

Find in Files...: Εύρεση σε αρχείο.

Replace...: Εύρεση και αντικατάσταση.

Go to Line: Εμφάνιση ζητούμενης γραμμής, με χρήση αριθμού γραμμής.

Expand Word: Αυτόματη συμπλήρωση κειμένου ώστε να ισούται με παρόμοιο κείμενο που πληκτρολογήθηκε στο παρελθόν.

Show Calltip: Ύστερα από άνοιγμα παρένθεσης μιας συνάρτησης ανοίγει ένα μικρό παράθυρο με πληροφορίες παραμέτρων για την συγκεκριμένη συνάρτηση.

Show Parens: Τονίζει/Εξυψώνει/Επιλέγει τις παρενθέσεις που περικλείουν το συγκεκριμένο block κώδικα.

Show Completions: Εμφανίζει ένα παράθυρο επιλογής που μας επιτρέπει να επιλέξουμε λέξεις κλειδιά.

4.1.3 Το μενού Format (Editor window)

Indent Region: Προσθέτει 4 κενά (Εσοχής) στην αρχή της επιλεγμένης περιοχής γραμμών.

Dedent Region: Κάνει το αντίθετο από την προηγούμενη εντολή.

Comment Out Region: Κάνει σχόλιο τις επιλεγμένες γραμμές κειμένου προσθέτοντας '##' στην αρχή κάθε γραμμής.

Uncomment Region: Κάνει το αντίθετο από την προηγούμενη εντολή.

Tabify Region: Μετατρέπει τα κενά που βρίσκονται στην αρχή τον επιλεγμένων γραμμών σε tabs. (Δεν προτείνεται για χρήση σε κώδικα Python.)

Untabify Region: Κάνει το αντίθετο από την προηγούμενη εντολή.

Toggle tabs–Ανοίγει Διάλογος για την εναλλαγή από κενά εσοχής σε Tab.

New Indent Width...: Ανοίγει διάλογο για την αλλαγή του προκαθορισμένου αριθμού κενόν εσοχής.

Format Paragraph: Αλλάζει την διάταξη τον κενόν σε μια παράγραφο, άλλες οι γραμμές μιας παραγράφου θα μειωθούν σε λιγότερες από 80 στήλες.

Strip trailing whitespace: Αφαιρεί όλους τους χαρακτήρες κενού μετά το τέλος του τελευταίου μη-κενού χαρακτήρα.

4.1.4 Το μενού Run (Editor window)

Python Shell: Άνοιγμα η επαναφορά του Python shell window

Check Module: Έλεγχος του κώδικα για πιθανά συντακτικά λάθη.

Run Module: Εκτέλεση του Κώδικα.

4.1.5 Το μενού Shell (Shell window)

View Last Restart: Πηγαίνει τον κέρσορα στο σημείο της τελευταίας επανεκκίνησης του Shell Window.

Restart Shell: Επανεκκίνηση του shell.

4.1.6 Το μενού Debug (Shell window)

Go to File/Line: Πηγαίνει στο σημείο οπού ορίστικε όνομα αρχείου και αριθμός γραμμής, ανοίγει το αρχείο και εμφανίζει την γραμμή.

Debugger (toggle): Πειραματική λειτουργία Debugger.

Stack Viewer: Εμφανίζει την στοίβα ιστορικού προηγούμενων μηνυμάτων σφάλματος.

Auto-open Stack Viewer (toggle): Επιλογή αυτόματης εμφάνισης του Stack Viewer.

4.1.7 Το μενού Options

Configure IDLE: Ανοίγει τον Διάλογο ρυθμίσεων του IDLE.

Code Context (toggle): Ανοίγει ένα πλαίσιο στην κορυφή του Editor Window, το οποίο εμφανίζει το block context της περιοχής του κώδικα που βγαίνει εκτός του παράθυρου.

4.1.8 Το μενού Window

Zoom Height: Μεγιστοποιεί το ύψος του παράθυρου

4.2 Λέξεις κλειδιά στην Python

Οι λέξεις κλειδιά στην Python είναι δεσμευμένες λέξεις οι οποίες επιτελούν κάποια λειτουργία. Δεν μπορούμε να χρησιμοποιήσουμε μια λέξη κλειδί σαν όνομα μεταβλητής ή όνομα συνάρτησης. Οι λέξεις κλειδιά στην Python έχουν διάκριση πεζών-κεφαλαίων, και υπάρχουν συνολικά 33 λέξεις κλειδιά. Παρακάτω παρουσιάζεται ο πίνακας με όλες της λέξεις κλειδιά:

FALSE	and	as	assert	break	class
continue	def	del	elif	else	except
finally	for	from	global	if	import
in	is	lambda	None	nonlocal	not
or	pass	raise	return	try	while
with	yield	TRUE			

4.3 Αναγνωριστικά στην Python

Αναγνωριστικά είναι τα ονόματα που δίνουμε στης μεταβλητές, συναρτήσεις, κλάσεις κτλ.

4.3.1 Κανόνες συγγραφής αναγνωριστικών

1. Μπορούν να αποτελούνται από συνδυασμό πεζών/κεφαλαίων χαρακτήρων, και αριθμών ή και του χαρακτήρα underscore (_).
2. Δεν είναι εφικτό να ξεκινάνε με αριθμό.
3. Δεν είναι εφικτό να χρησιμοποιήσουμε λέξεις κλειδιά ως αναγνωριστικό.
4. Δεν μπορούμε να χρησιμοποιήσουμε τους ειδικούς χαρακτήρες όπως: !,@,#,$,% κτλ.
5. Ισχύει ο κανόνας διάκρισης πεζών-κεφαλαίων.

4.4 Σύντομη αναφορά σε όλους τους τελεστές της γλώσσας Python

Πινάκας αριθμητικών τελεστών:

Τελεστής	Σημασία
+	Πρόσθεση
-	Αφαίρεση
*	Πολλαπλασιασμός
/	Διαίρεση
%	Υπόλοιπο ακεραίας διαίρεσης
//	Πηλίκο ακεραίας διαίρεσης
**	Ύψωση σε δύναμη

Πινάκας τελεστών σύγκρισης:

Τελεστής	Σημασία
>	Μεγαλύτερο
<	Μικρότερο
==	Ισότητα
!=	Ανισότητα
>=	Μεγαλύτερο η ίσο
<=	Μικρότερο η ίσο

Πινάκας λογικών τελεστών:

Τελεστής	Σημασία
And	Λογικό AND
Or	Λογικό OR
not	Λογικό NOT

Τελεστές επίπεδου bit:

Τελεστής	Σημασία
&	AND επίπεδου bit
|	OR επίπεδου bit
~	NOT επίπεδου bit
^	XOR επίπεδου bit
>>	Ολίσθηση προς τα δεξιά επίπεδου bit
<<	Ολίσθηση προς τα αριστερά επίπεδου bit

Τελεστές εκχώρησης:

Τελεστής	Σημασία
=	Εκχώρηση σε μεταβλητή
+=	Πρόσθεσή σε μεταβλητή
-=	Αφαίρεση από μεταβλητή
*=	Πολλαπλασιασμός μεταβλητής
/=	Διαίρεσή μεταβλητής
%=	Υπόλοιπο ακεραίας διαίρεσης μεταβλητής
//=	Πηλίκο ακεραίας διαίρεσης μεταβλητής
**=	Ύψωση σε δύναμη της μεταβλητής
&=	Σύζευξη (AND) επίπεδου bit μεταβλητής
|=	Διάζευξη (OR) επίπεδου bit μεταβλητής
^=	Αποκλειστική διάζευξη (XOR) επίπεδου bit μεταβλητής
>>=	Ολίσθηση προς τα δεξιά επίπεδου bit μεταβλητής
<<=	Ολίσθηση προς τα αριστερά επίπεδου bit μεταβλητής

Τελεστές ταυτότητας:

Τελεστής	Σημασία	Παράδειγμα
is	Έλεγχος για το αν ισούνται οι τελεστέοι αλλιώς επιστρέφει False	x is True
is not	Έλεγχος για το αν είναι άνισοι οι τελεστέοι αλλιώς επιστρέφει False	x is not True

Τελεστές μέλους:

Τελεστής	Σημασία	Παράδειγμα
in	Επιστρέφει True αν ο πρώτος τελεστέος περιέχεται στον δεύτερο τελεστέο	5 in x
not in	Επιστρέφει True αν ο πρώτος τελεστέος ΔΕΝ περιέχεται στον δεύτερο τελεστέο	5 not in x

4.5 Εντολές πολλών γραμμών στην Python

Εδώ θα αναφερθούνε οι τρόποι με τους οποίους μπορούμε να σπάσουμε μια μεγάλη εντολή ή να ενώσουμε μικρές εντολές σε μια γραμμή.

Χρησιμοποιώντας τον χαρακτήρα "\".

```
x =  1* 3 + \
    5 + 6
```

Παράδειγμα 4.5.1

Χρησιμοποιώντας αγκύλες "()", τετράγωνες αγκύλες "[]" ή braces "{}".

```
x = (1 *3
5 + 6)
```

```
x = [1 *3
5 + 6]
```

```
x = {1 *3
5 + 6}
```

Παράδειγμα 4.5.2

Χρησιμοποιώντας ερωτηματικά ";" μπορούμε να συρρικνώσουμε πολλαπλές γραμμές κώδικα σε μια γραμμή.

```
x = 5; y = 2.2; z = "george"
```

Παράδειγμα 4.5.3

4.6 Εσοχές στην Python

Οι εσοχές στην Python έχουνε μεγάλη σημασία μιας και από αυτές ο μεταγλωττιστής θα καταλάβει που ξεκινάει και που τελειώνει ένα block εντολών. Ένα block εντολών αρχίζει με την πρώτη εσοχή και τελειώνει μόλις φτάσουμε σε γραμμή οπού δεν υφίσταται πλέον εσοχή.
Για να δημιουργήσουμε μια εσοχή απαιτείται να εισάγουμε 4 φορές τον χαρακτήρα Space. Εδώ βλέπουμε την δομή επιλογής If στην οποία είναι απαραίτητη εσοχή για να λειτουργήσει.

```
if True:
    print('Hello  World')
```

Παράδειγμα 4.6.1

4.7 Σχόλια στην Python

Για να δημιουργήσουμε σχόλιο στην Python χρησιμοποιούμε την δίεση “#” στην αρχή της γραμμής.

```
#Σχόλιο γραμμής
```

Παράδειγμα 4.7.1

Είναι εφικτό να δημιουργήσουμε σχόλια πολλών γραμμών βάζοντας την δίεση στην αρχή κάθε γραμμής. Ένας άλλος τρόπος για να δημιουργήσουμε σχόλια πολλών γραμμών είναι να χρησιμοποιήσουμε τριπλά εισαγωγικά <’’’> ή <”””>.

```
"""Σχόλιο πολλαπλών
γραμμών"""
```

Παράδειγμα 4.7.2

4.8 Docstring στην Python

Ένα docstring είναι μια σύντομή περιγραφή για το τι κάνει μια συνάρτηση/κλάση/μέθοδο και το γράφουμε μετά την δήλωση της συνάρτησης/κλάσης/μεθόδου χρησιμοποιώντας τριπλά εισαγωγικά <’’’> ή <”””>.

```
def add(x,y):
    """Συνάρτηση που επιστρέφει το αποτέλεσμα της πρόσθεσης x με y"""
    return x+y
```

Παράδειγμα 4.8.1

Για να έχουμε πρόσβαση στο Docstring μιας συνάρτησης/κλάσης/μεθόδου χρησιμοποιούμε το όνομα τις μαζί με τον τελεστή τελείας “.” και στην συνέχεια την εντολή “__doc__” (με δυο κάτω παύλες από κάθε μεριά). Στο παράδειγμα που ακολουθεί χρησιμοποιούμε την βοήθεια της συνάρτησης add ώστε να μας εμφανίσει το docstring του παραδείγματος 4.7 στην οθόνη του υπολογιστή.

```
print(add.__doc__)
```

Παράδειγμα 4.8.2

4.9 Μεταβλητές στην Python

Δεν είναι απαραίτητο να ορίσουμε τον τύπο μιας μεταβλητής στην Python απλά καταχωρούμε την τιμή που θέλουμε να αποθηκεύσουμε και αυτομάτως θα αναγνωριστεί ο τύπος της μεταβλητής και η μεταβλητή θα δημιουργηθεί. Προϋπόθεση για να δημιουργήσουμε μια μεταβλητή είναι να της καταχωρίσουμε μια αρχική τιμή. Με την χρήση του τελεστή εκχώρησης(=) μας είναι εφικτή η καταχώριση τιμών σε μια μεταβλητή.

```
x = 2
y = 4.2
z = "abc"
```

Παράδειγμα 4.9.1

Στο Παράδειγμα 4.9.1 βλέπουμε την καταχώρηση ενός ακεραίου (integer) στην μεταβλητή x, την καταχώριση ενός πραγματικού αριθμού (float) στην μεταβλητή y, αλλά και την καταχώριση κειμένου(String) στην μεταβλητή z.

Πολλαπλές αρχικοποιήσεις μεταβλητών

Μας είναι εφικτή η αρχικοποίηση πολλών μεταβλητών στην ίδια γραμμή.

```
x, y, z = 1, 5.6, "test"
a = b = c = "Same text"
```

Παράδειγμα 4.9.2

4.9.1 Λίστα τιμών/Αμετάβλητη Λίστα

Οι Λίστες τιμών μας επιτρέπουν να αποθηκεύουμε πολλά δεδομένα μέσα τους. Τα δεδομένα που μπορούμε να εισάγουμε στις λίστες δεν είναι απαραίτητο να είναι του ίδιου τύπου. Επίσης, είναι δυνατό μέσα σε μια λίστα να περιέχονται άλλες λίστες. Η διαφορά στην δημιουργία απλής λίστας με λίστα σταθερών τιμών είναι στις αγκύλες, αν βάλουμε τετράγωνες αγκύλες“[]” δημιουργούμε απλή λίστα αν βάλουμε απλές αγκύλες “()” δημιουργούμε λίστα σταθερών τιμών.

```
# άδεια λίστα
myList = []

# αμετάβλητη λίστα ακέραιων
myList = (1, 2, 3)

# λίστα ανάμεικτων δεδομένων
myList = [1, "Hello", 3.4]
```

Παράδειγμα 4.9.3

4.9.2 Τύποι δεδομένων στην Python

Στην Python υπάρχουν οι εξής τύποι δεδομένων:

- Αριθμητικοί τύποι(int, float, complex)
- Λίστες, Αμετάβλητες Λίστες(Tuple)
- Ακολουθίες χαρακτήρων(Strings)
- Set (αταξινόμητες συλλογές από διάφορους τύπους δεδομένων)
- Λεξικό (αταξινόμητη συλλογή από δεδομένα που συνοδεύονται από αντιπροσωπευτικό κλειδί)

4.10 Είσοδος και έξοδος στην Python

4.10.1 Έξοδος

Με την συνάρτηση Print() μπορούμε να εμφανίζουμε δεδομένα στην οθόνη μας.

Το βασικό σώμα της συνάρτησης Print() είναι το εξής:

```
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
```

Οπού το *objects είναι οι τιμές/κείμενο που θα τυπωθεί, στο sep ορίζουμε το πώς θα διαχωρίζονται η μεταβλητές μας και έχει ως προκαθορισμένη τιμή το κενό διάστημα, στο end ορίζουμε τι θα συμβαίνει στο τέλος της γραμμής, δηλαδή αν θα αλλάζει γραμμή. Στο file ορίζουμε αν οι τιμές/κείμενο θα τυπωθούν σε αρχείο η στην οθόνη(με την εντολή sys.stdout τυπώνει στην οθόνη).

```
>>> print(1,2,3,4)
Έξοδος: 1 2 3 4


>>> print(1,2,3,4,sep='*')
Έξοδος: 1*2*3*4


>>> print(1,2,3,4,sep='#',end='&')
Έξοδος: 1#2#3#4&
```

Παράδειγμα 4.10.1

Μορφοποίηση Εξόδου

Με την εντολή str.format() μας είναι εφικτό να δημιουργήσουμε κείμενο το οποίο θα ενημερώνεται δυναμικά από τις μεταβλητές μας. Ακολουθούνε μερικά παραδείγματα σχετικά με αυτήν την εντολή:

```
>>> x = 2; y = 5
>>> print('Η τιμή του x είναι {} και του y είναι {}'.format(x,y))
Έξοδος: Η τιμή του x είναι 2 και του y είναι 5
```

Παράδειγμα 4.10.2

Τα Braces “{}” Κρατάνε τον χώρο στον οποίο θα εμφανιστούν τα περιεχόμενα των μεταβλητών μας με την σειρά που είναι στην παρένθεση. Μας είναι εφικτό να αλλάξουμε την σειρά με την οποία θα εμφανίζονται οι μεταβλητές με την χρήση του αριθμού σειράς του κάθε στοιχείου στην παρένθεση, το πρώτο στοιχείο ξεκάνει από τον αριθμό 0.
Εδώ έχουμε μερικά παραδείγματα για τον καθορισμό της σειράς με την οποία θα εμφανιστούν οι μεταβλητές μας:

```
>>> print('έχουμε {0} και {1}'.format('μεταβλητές', 'σταθερές'))
Έξοδος: έχουμε μεταβλητές και σταθερές


>>> print(' έχουμε {1} και {0}'.format('μεταβλητές ', 'σταθερές'))
Έξοδος: έχουμε σταθερές και μεταβλητές
```

Παράδειγμα 4.10.3

Μας είναι εφικτό να τυπώσουμε κείμενο με τρόπο παρόμοιο με αυτόν της γλώσσας προγραμματισμού C χρησιμοποιώντας τον ειδικό χαρακτήρα %:

```
>>> a = 56.7890123
>>> print('Η τιμή του a είναι %3.2f' %a)
Έξοδος: Η τιμή του a είναι 56.78
>>> print('Η τιμή του a είναι %3.4f' %a)
Έξοδος: Η τιμή του a είναι 56.7890
```

Παράδειγμα 4.10.4

4.10.2 Είσοδος

Με την εντολή input() είναι εφικτό να γίνει εισαγωγή δεδομένων από τον χρήστη κατά την εκτέλεση της εφαρμογής. Το βασικό σώμα της συνάρτησης input() είναι το εξής:

```
input([prompt])
```

Όπου στο **prompt** μπορούμε να γράψουμε ένα κείμενο το οποίο θα εξηγεί στον χρήστη ποια είναι τα απαιτούμενα δεδομένα προς πληκτρολόγηση.

```
>>> num = input('Δώσε ένα αριθμό: ')
Έξοδος: Δώσε ένα αριθμό:
Είσοδος: 10
>>> num
Έξοδος: '10'
```

Παράδειγμα 4.10.5

Τα Δεδομένα εισάγει ο χρήστης με την εντολή input() αποθηκεύονται ως συμβολοσειρά(String). Ευτυχώς είναι εφικτό να μετατρέψουμε αριθμητικά δεδομένα που είναι αποθηκευμένα ως συμβολοσειρές σε αριθμούς με την χρήση τον εντολών int() και float().

```
>>> num = '10' #Συμβολοσειρά
>>> int(num)
Έξοδος: 10
>>> float(num)
Έξοδος: 10.0
```

Παράδειγμα 4.10.6

4.11 Εισαγωγή έτοιμων βιβλιοθηκών

Μια βιβλιοθήκη η χώρος ονομάτων είναι ένα αρχείο(με κατάληξη .py) το οποίο περιέχει μέσα του έτοιμες συναρτήσεις που μπορούν να επεκτείνουν την λειτουργικότητα τον εφαρμογών μας. Με την εντολή **import** είμαστε σε θέση να εισάγουμε έτοιμες βιβλιοθήκες στο πρόγραμμα μας.

Στο παρακάτω παράδειγμα εισάγουμε την βιβλιοθήκη math που ανήκει στης βασικές βιβλιοθήκες της Python.

```
>>> import math
>>> print(math.pi) #Η εντολή math.pi μας επιστρέφει το π.
Έξοδος: 3.141592653589793
```

Παράδειγμα 4.11.1

Μας παρέχεται η δυνατότητα να ορίζουμε ψευδώνυμα με την εντολή **as** κατά την δήλωση των βιβλιοθηκών ώστε να διευκολυνθεί η συγγραφή του κώδικα.

```
>>> import math as m
>>> print(m.pi) #Η εντολή m.pi μας επιστρέφει το π.
Έξοδος: 3.141592653589793
```

Παράδειγμα 4.11.2

4.12 Δομή επιλογής

4.12.1 Απλή δομή της εντολής If

Παρακάτω παρουσιάζεται το σώμα της δομής επιλογής If. Προσέχουμε η εντολές που θα εκτελεστούν εντός της If να έχουν εσοχή τέσσερα(4) κενά διαστήματα σε σχέση με την If:

```
if λογική_έκφραση:
    εντολές
```

Οι εντολές που βρίσκονται εντός της If θα εκτελεστούν μόνο αν η λογική έκφραση της If ισχύει, αλλιώς το πρόγραμμα θα συνεχίσει από τις εντολές που βρίσκονται στην ίδια εσοχή με την If.

Παράδειγμα χρήσης της εντολής If:

```
num=3
if num > 0:
   print(num, "είναι θετικός αριθμός.")
print("αυτό θα εκτυπώνεται πάντα.")
```

Έξοδος: 3 είναι θετικός αριθμός.
αυτό θα εκτυπώνεται πάντα.

Παράδειγμα 4.12.1

4.12.2 Δομή της εντολής If…else

```
if λογική_έκφραση:
   εντολές που θα εκτελεστούν αν ισχύει η συνθήκη
else:
   εντολές που θα εκτελεστούν αν δεν ισχύει η συνθήκη
```

Παράδειγμα χρήσης της εντολής If…else:

```
num=3
if num >=0:
   print("Θετικός η Μηδέν")
else:
   print("Αρνητικός Αριθμός")
```

Έξοδος: Θετικός η Μηδέν

Παράδειγμα 4.12.2

4.12.3 Δομή εντολής If…elif…else

```
if λογική_έκφραση_1:
   εντολές που θα εκτελεστούν αν ισχύει η συνθήκη_1
elif λογική_έκφραση_2:
   εντολές που θα εκτελεστούν αν ισχύει η συνθήκη_2
else:
   εντολές που θα εκτελεστούν αν δεν ισχύει καμία συνθήκη
```

Παράδειγμα χρήσης της εντολής If…elif…else:

```
num = 3.4
if num > 0:
   print("Θετικός Αριθμός")
elif num == 0:
   print("Μηδέν")
else:
   print("Αρνητικός Αριθμός")
```

Έξοδος: Θετικός Αριθμός

Παράδειγμα 4.12.3

4.12.4 Εμφωλευμένες If

Στο παρακάτω παράδειγμα θα γίνει παρουσίαση εμφωλευμένων/φωλιασμένων If ώστε να ξεκαθαριστεί η χρήση της εσοχής:

```
num = float(input("Παρακαλώ δώστε έναν αριθμό:"))
if num >= 0:
   if num == 0:
      print("Μηδέν")
   else:
      print("Θετικός Αριθμός")
else:
   print("Αρνητικός Αριθμός")
```

Παράδειγμα:

>>> 5

Έξοδος: Θετικός Αριθμός

Παράδειγμα 4.12.4

4.13 Δομές Επανάληψης

4.13.1 Δομή Επανάληψης For

Παρακάτω παρουσιάζεται το σώμα της δομής επανάληψης For:

```
for val in sequence:
   σώμα της for
```

Οπού **val** είναι η μεταβλητή η όποια θα κρατεί σε κάθε πέρασμα/επανάληψη την επόμενη τιμή που υπάρχει μέσα στο **sequence** μέχρι να εξαντληθούν όλες οι τιμές που περιέχει. Μόλις εξαντληθούν οι τιμές που περιέχει το sequence ο βρόχος θα σταματήσει. Σε κάθε επανάληψη θα εκτελείται ο κώδικας που βρίσκεται στο σώμα της for.

Ακλουθεί ένα παράδειγμα χρήσης της for:

```
numbers = [6, 5, 3, 8, 4, 2, 5, 4, 11] #λίστα 9 τιμών
sum = 0

for val in numbers:
   sum = sum+val
print("The sum is", sum)

Έξοδος: The sum is 48
```

Παράδειγμα 4.13.1

Δομή για πεπερασμένο αριθμό επαναλήψεων

Αν θέλουμε να συμβεί μια επανάληψη για ένα πεπερασμένο σύνολο τιμών τότε, θα χρειαστεί να χρησιμοποιήσουμε την εντολή range() μέσα στον βρόχο For. Η δομή της συνάρτησης range() είναι η εξής:

```
range(start,stop,step_size)
```

Οπού **start** είναι η τιμή εκκίνησης, **stop** είναι η τιμή τερματισμού και **step_size** το βήμα.
Αν βάλουμε ως όρισμα μόνο μια τιμή π.χ. range(10) θα δημιουργήσει την ακολουθία από 0 μέχρι το 9.

Στην συνέχεια παρουσιάζεται ένα παράδειγμα του βρόχου For σε συνεργασία με την συνάρτηση range():

```
sum = 0
for i in range(10): #Ο βρόχος θα τρέξει 10 φορές
   sum+=i
print("Το αποτέλεσμα είναι", sum)

Έξοδος: Το αποτέλεσμα είναι 45
```

Παράδειγμα 4.13.2

4.13.2 Δομή Επανάληψης While

Παρακάτω παρουσιάζεται το σώμα της δομής επανάληψης While:

```
while συνθήκη:
    σώμα της while
```

Όσο ισχύει η **συνθήκη** της while ο βρόχος θα εκτελείται.

Στην συνεχεία παρουσιάζεται ένα παράδειγμα του βρόχου While:

```
sum = 0
i = 1

while i <= 10:
    sum = sum + i
    i = i + 1

print("Το αποτέλεσμα είναι",sum)

Έξοδος: Το αποτέλεσμα είναι 55
```

Παράδειγμα 4.13.3

4.14 Δημιουργία Συναρτήσεων

Παρακάτω παρουσιάζεται το σώμα μιας συνάρτησης:

```
def function_name(parameters):
    """docstring"""
    statement(s)
```

Όπου **function_name** το όνομα με το οποίο θα καλούμε την συνάρτηση, **parameters** οι παράμετροι της συνάρτησης, **docstring** είναι η σύντομη περιγραφή για το τι κάνει η συνάρτηση και **statement(s)** το σώμα της συνάρτησης.

Στην συνεχεία παρουσιάζεται ένα παράδειγμα μιας συνάρτησης:

```
def greet(name):
    """Η συνάρτηση χαιρετά το άτομο του οποίου το όνομα
    περάστηκε ως παράμετρος"""
    print("Hello "+name+" Good morning!")
```

Παράδειγμα 4.14.1

Για να καλέσουμε την συνάρτηση χρησιμοποιούμε το όνομα της συνάρτησης και μέσα στην παρένθεση εισάγουμε μια παράμετρο που στην δική μας περίπτωση είναι ένα όνομα:

```
>>> greet('George')
Έξοδος: Hello George Good morning!
```

Παράδειγμα 4.14.2

4.14.1 Επιστροφή δεδομένων από συνάρτηση

Είναι εφικτή η επιστροφή δεδομένων από μια συνάρτηση με τη χρήση της εντολής return:

```
def absolute_value(num):
    """Αυτή η συνάρτηση επιστρέφει
    την απόλυτη τιμή του αριθμού που
    θα περαστεί ως παράμετρος"""
    if num >= 0:
        return num
    else:
        return -num
```

Παράδειγμα:

```
>>> print(absolute_value(-2))
```

Έξοδος: 2

Παράδειγμα 4.14.3

4.15 Δημιουργία Κλάσεων

Παρακάτω παρουσιάζεται το σώμα μιας κλάσης:

```
class NewClassName:
    """docstring"""
    #Μεταβλητές και συναρτήσεις μέλη...
```

Όπου **NewClassName** το όνομα το όνομα της κλάσης και **docstring** είναι η σύντομη περιγραφή για το τι κάνει η κλάση.

Στην συνεχεία παρουσιάζεται ένα παράδειγμα μιας κλάσης:

```
class TestClass:
    """Πειραματική κλάση"""
    x = 2.2
    def func(self):
        print('Test Class')
```

Παράδειγμα 4.15.1

Για να ορίσουμε μια συνάρτηση μέλος μιας κλάσης βάζουμε το πρόθεμα def.
Το πρώτο όρισμα μιας συνάρτησης μέλους(και ας μη έχει άλλα ορίσματα) πρέπει να είναι πάντα self.

Για να δημιουργήσουμε ένα αντικείμενο κλάσης απλά ορίζουμε το όνομα που θέλουμε να έχει το αντικείμενο μας, και μετά χρησιμοποιώντας τον τελεστή εκχώρησης (=) βάζουμε το όνομα της κλάσης για την οποία θέλουμε να δημιουργήσουμε αντικείμενο.

```
tst1 = TestClass()
```

Στο παράδειγμα που ακολουθεί θα χρησιμοποιηθεί η κλάση του παραδείγματος 4.15.1 ώστε να δούμε της λειτουργίες αυτής της κλάσης:

```
1:  tst1 = TestClass()
2:  tst1.__doc__
3:  tst1.x
4:  tst1.func()
```

Έξοδος:

```
'Πειραματική κλάση'
2.2
Test Class
```

Παράδειγμα 4.15.2

Ανάλυση:

1: Ορισμός ενός αντικείμενου TestClass με όνομα tst1.

2: Εμφάνιση του docstring του αντικείμενου tst1.

3: Εμφάνιση της δημόσιας μεταβλητής μέλους x του αντικείμενου tst1.

4: Εκτέλεση της συνάρτησης μέλους func() του αντικείμενου tst1, που έχει ως αποτέλεσμα να μας τυπώσει “Test Class” στην οθόνη.

4.15.1 Συνάρτηση δόμησης

Το σώμα της συνάρτησης δόμησης είναι το έξης:

```
def __init__(self,par1=0,par2=2,…):
   self.x=par1
   self.y=par2
   ...
```

Όπου **par1**,**par2** οι παράμετροι της κλάσης και **x**,**y** οι μεταβλητές μέλη της κλάσης, στις όποιες θα αποθηκευτούν οι τιμές των παραμέτρων. Είναι εφικτό να δώσουμε αρχικές τιμές στις παραμέτρους.

Παρακάτω θα μελετήσουμε μια κλάση με συνάρτηση δόμησης.

```
class TestClass2:
   def __init__(self,par1=0,par2=2):
      self.x=par1
      self.y=par2
   def func(self):
      print('το x είναι: %d'%self.x)
      print('το y είναι: %d'%self.y)
```

Παράδειγμα 4.15.3

Όταν δημιουργείται αντικείμενο της κλάσης TestClass2, χωρίς ορίσματα οι τιμές που θα έχουν οι μεταβλητές θα είναι x=0 και y=2.

Παρακάτω θα μελετηθούν οι λειτουργίες της κλάσης παρουσιάζοντας την χρησιμότητα της συνάρτησης δόμησης:

```
1:    tst1 = TestClass2()
2:    tst2 = TestClass2(5,7)
3:    print("tst1:")
4:    tst1.func()
5:    print("tst2:")
6:    tst2.func()
```

Έξοδος:

```
tst1:
το x είναι: 0
το y είναι: 2
tst2:
το x είναι: 5
το y είναι: 7
```

Παράδειγμα 4.15.4

Ανάλυση:

1: Δημιουργία αντικείμενου tst1 χωρίς παραμέτρους.

2: Δημιουργία αντικείμενου tst2 με παραμέτρους.

3: Τυπώνεται το κείμενο “tst1:” στην οθόνη.

4,5: Εκτελείται η συνάρτηση μέλους func του αντικείμενου tst1, που έχει ως αποτέλεσμα να τυπώσει της μεταβλητές μέλη x και y, οι όποιες πρόεκυψαν από τις προκαθορισμένες τιμές της συνάρτησης δόμησης μιας και δεν περάστηκαν παράμετροι.

6,7: Εκτελείται η συνάρτηση μέλους func του αντικείμενου tst2, που έχει ως αποτέλεσμα να τυπώσει της μεταβλητές μέλη x και y, οι οποίες προέκυψαν από τις παραμέτρους που ορίστηκαν κατά την δημιουργία του αντικειμένου.

4.15.2 Διαγραφή αντικείμενου

Μας είναι εφικτό να διαγράψουμε ένα αντικείμενο η μια ιδιότητα ενός αντικειμένου με την εντολή **del**.

```
# διαγραφή της μεταβλητής x του αντικειμένου tst1
del tst1.x
#διαγραφή του αντικειμένου tst2
del tst2
```

Παράδειγμα 4.15.5

4.16 Η βιβλιοθήκη time

Η βιβλιοθήκη time είναι σημαντική λόγω του ότι μας παρέχει λειτουργίες όπως την χρονοκαθυστέρηση στις εφαρμογές μας. Στην συνέχεια θα παρουσιαστούν οι σημαντικότερες συναρτήσεις αυτής της βιβλιοθήκης.

Για να συμπεριλάβουμε την βιβλιοθήκη time στο πρόγραμμα μας θα πρέπει στην κορυφή του προγράμματος να γράψουμε:

```
import time
```

Παρακάτω παρουσιάζονται μερικές συναρτήσεις τις βιβλιοθήκης time:

start = time.time()= Αποθηκεύει σε μια μεταβλητή με όνομα start την τρέχουσα χρονική στιγμή (με βάση Unix Timestamp).
time.sleep(x) = Δημιουργεί χρονοκαθυστέρηση, για **x** δευτερόλεπτα. Οι τιμές που δέχεται είναι float(0.5, 1, 2.4, κτλ).

4.17 Εκτέλεση κώδικα python από το τερματικό

Για να είμαστε σε θέση να τρέχουμε κώδικα python από το τερματικό θα πρέπει να προσθέτουμε στην κορυφή των προγραμμάτων python την εξής γραμμή κώδικα: “#!/usr/bin/env python”. Και θα πρέπει να δώσουμε δικαιώματα εκτέλεσης στο python αρχείο με την εντολή “chmod +x <όνομα_ αρχείου>.py”.

Ύστερα μπορούμε να εκτελέσουμε το αρχείο χρησιμοποιώντας το πρόθεμα “./”.

Αν για παράδειγμα είχαμε ένα αρχείο με όνομα “prog1.py” η εντολή για να το εκτελέσουμε θα ήταν “./prog1.py”. Τονίζεται ότι θα πρέπει να βρισκόμαστε στον ίδιο κατάλογο με το αρχείο python που θέλουμε να τρέξουμε.

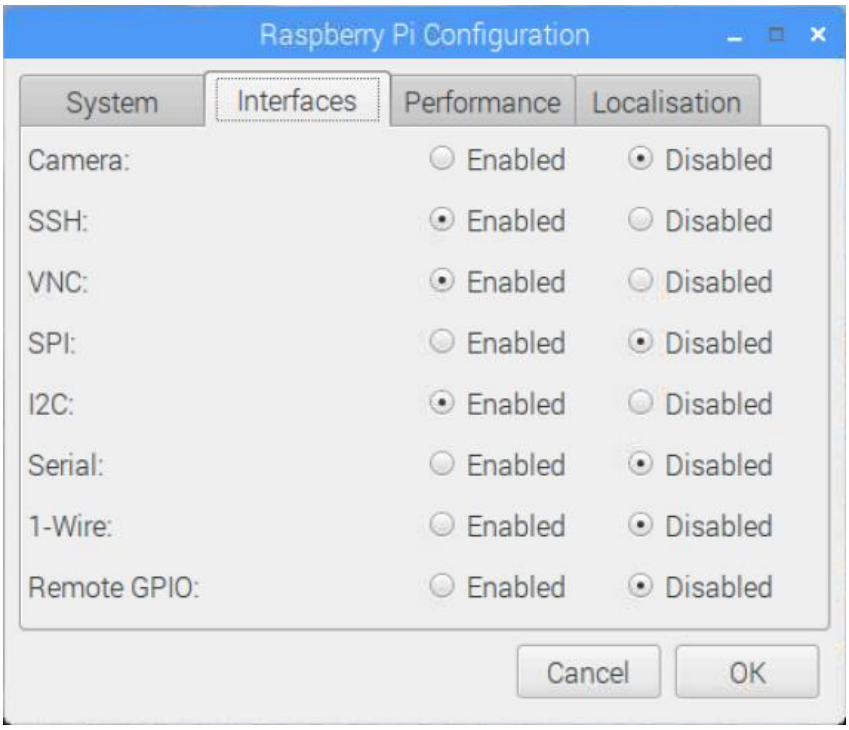
Κεφάλαιο 5

Σύνδεση συσκευών και αισθητήρων

Σε αυτό το κεφάλαιο θα γίνει σύνδεση και προγραμματισμός διαφορών συσκευών και αισθητήρων στο Raspberry Pi, με την χρήση της γλώσσας προγραμματισμού Python.

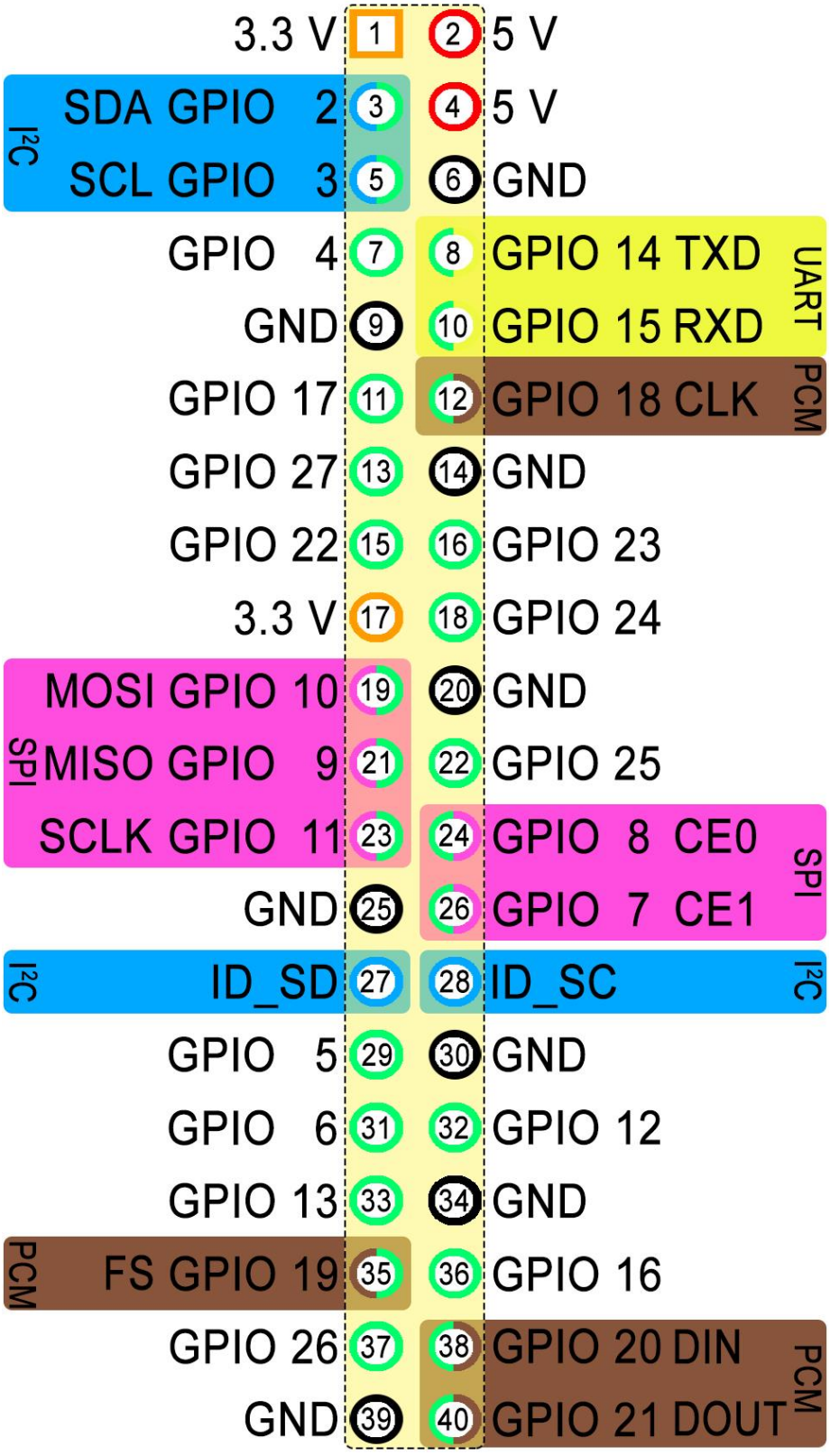
Τι θα πρέπει να προσέξουμε;

Για να λειτουργήσουν τα πειράματα θα πρέπει να δοθεί προσοχή στα ενεργά πρωτόκολλα που θα χρησιμοποιήσουμε. Για να ρυθμιστεί σωστά το raspberry θα πρέπει να πάμε από την έναρξη στο Preferences>Raspberry Pi Configuration και από εκεί στην καρτέλα Interfaces, και να αλλάξουμε της ρυθμίσεις όπως στην εικόνα 5.1.



Εικόνα 5.1 (παράθυρο ρυθμίσεων του raspbian)

5.1 Διάγραμμα Θυρών GPIO(Για τα Μοντέλα B και Zero)



Σχήμα 5.1.1 (οι θύρες GPIO)

5.1.1 Χαρακτηρίστηκα και Αρίθμηση των θυρών GPIO

Υπάρχουν συνολικά 40 θύρες GPIO στα μοντέλα B και Zero, από τις οποίες οι 26 είναι προγραμματίσιμες. Στο πίνακα που ακλουθεί εξηγούνται η ομάδες θυρών που υπάρχουν:

Σύνολο	**Τύπος θύρας**	**Ονομασίες**	**Α/Α θυρών**
8	GND	-	6,9,14,30,25,30,34,39
2	5V	-	2,4
2	3.3V	-	1,17
2	Ειδικές θύρες I2C	ID_SD, ID_SC	27,28
26	Προγραμματιζόμενες Θύρες GPIO	-	3,5,7,8,10,11,12,13,15,16,18,19,21,22, 23,24,26,29,31,32,33,35,36,37,38,40

Πίνακας 5.1.2

Όπου οι 26 προγραμματιζόμενες θύρες χωρίζονται στης εξής υποκατηγορίες:

Σύνολο	**Τύπος θύρας**	**Ονομασίες**	**Α/Α θυρών**
2	I2C	SDA, SCL	3,5
2	UART	TXD, RXD	8,10
5	SPI	MOSI, MISO, SCLK, CE0, CE1	19,21,23,24,26
4	PCM	CLK, FS, DIN, DOUT	12,35,38,40
13	-	-	7,11,13,15,16,18,22, 29,31,32,33,36,37

Πίνακας 5.1.3

Υπάρχουν δυο τρόποι με τους οποίους μπορούμε να αναφερθούμε στις θύρες GPIO. Ο ένας τρόπος είναι να κάνουμε αναφορά στην φυσική σειρά των ακροδεκτών GPIO. Ο άλλος τρόπος είναι να κάνουμε αναφορά στον αριθμό σειράς BCM των ακροδεκτών. Το καλό της φυσικής αναφοράς είναι ότι θα γίνει η χρήση των ίδιων ακροδεκτών σε όλες εκδόσεις Raspberry. Αν κάνουμε αναφορά στον αριθμό BCM θα χρειαστεί να δουλεύουμε με το διάγραμμα της έκδοσης Raspberry που χρησιμοποιούμε (Σχήμα 5.1.1). Επίσης υπάρχει κίνδυνος ο κώδικας μας να μην λειτουργεί σε όλες τις εκδόσεις τον Raspberry.

Στο σχήμα 5.1.1 φαίνεται η φυσική αρίθμηση των θυρών (οι κυκλωμένοι αριθμοί) καθώς και η αρίθμηση BCM (οι αριθμοί που ξεκινάμε με την λέξη GPIO). Για να συμφωνεί η πλακέτα του raspberry με το σχήμα 5.1.1 θα πρέπει να γυρίσουμε την πλακέτα έτσι ώστε οι θύρες USB να βρίσκονται κάτω και η θύρα DSI στην κορυφή. Η θύρα που βρίσκεται προς την κατεύθυνση της θύρας DSI είναι η θύρα 1.

5.1.2 Υποστηριζόμενα πρωτόκολλα διασύνδεσης συσκευών και αισθητήρων

Τα υποστηριζόμενα πρωτόκολλα διασύνδεσης των μοντέλων B και Zero είναι τα εξής:

1. UART. (Universal Asynchronous Receiver/Transmitter)
2. SPI. (Serial Peripheral Interface Bus)
3. I2C. (Inter-Integrated Circuit)
4. PCM (I2S – Πρωτόκολλο διασύνδεσης ψηφιακών συσκευών ήχου).

5.2 Εντολές GPIO σε Python

5.2.1 Εισαγωγή βιβλιοθήκης GPIO

Για να είναι εφικτή η χρήση των συναρτήσεων που μας επιτρέπουν να χειριστούμε τις GPIO θύρες θα πρέπει πρώτα να γίνει εισαγωγή της βιβλιοθήκης GPIO στον Python κώδικα. Η εισαγωγή της βιβλιοθήκης GPIO γίνεται με τον εξής τρόπο:

```
import RPi.GPIO as GPIO
```

5.2.2 Επιλογή τρόπου αρίθμησης GPIO θυρών

Ανάλογα με τον τρόπο αρίθμησης των θυρών που θα θελήσουμε να χρησιμοποιήσουμε, θα πρέπει να συμπεριλάβουμε την ανάλογη γραμμή κώδικα στο πρόγραμμα μας:

```
GPIO.setmode(GPIO.BOARD)#Για Φυσικό τρόπο αρίθμησης
#ή
GPIO.setmode(GPIO.BCM)#Για BCM τρόπο αρίθμησης
```

Είναι εφικτό το πρόγραμμα να μας επιστρέψει τον τρέχοντα τρόπο αρίθμησης των θυρών GPIO με την εξής εντολή:

```
mode = GPIO.getmode()
```

Όπου αν τυπώσουμε το περιεχόμενο της μεταβλητής mode στην οθόνη, θα μας επιστρέψει μια από τις πιθανές τιμές που είναι: **GPIO.BOARD** που σημαίνει φυσική αρίθμηση των θυρών, **GPIO.BCM** που σημαίνει BCM αρίθμηση των θυρών ή **None** που σημαίνει ότι δεν έχει ακόμα καθοριστεί τρόπος αρίθμησης των θυρών GPIO.

5.2.3 Προειδοποιήσεις (Warnings)

Στην περίπτωση που έχουν τροποποιηθεί οι θύρες GPIO από κάποια script που εκτελέστηκαν προηγουμένως, υπάρχει πιθανότητα το RPi.GPIO να επιστρέψει μήνυμα προειδοποίησης αν κάποια από τις θύρες GPIO βρίσκεται σε κατάσταση διαφορετική της αρχικής. Για να απενεργοποιήσουμε τα μηνύματα προειδοποιήσεων θα πρέπει να συμπεριλάβουμε την εξής γραμμή κώδικα στο πρόγραμμα μας:

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

5.2.4 Αρχικοποίηση θυρών/καναλιών (channel)

5.2.4.1 Ορισμός θύρας ως Είσοδος

Για να ορίσουμε αν μια θύρα θα είναι είσοδος στο πρόγραμμα, θα πρέπει να χρησιμοποιήσουμε την εξής εντολή:

```
GPIO.setup(channel,GPIO.IN)
GPIO.setup(channel, GPIO.IN, pull_up_down=mode)
```

όπου **channel** ο αριθμός της θύρας.

όπου **mode**: GPIO.PUD_UP για Pull-up αντίσταση και GPIO.PUD_DOWN για Pull-down αντίσταση. Με αυτήν την ρύθμιση μας είναι εφικτό να συνδέσουμε διακόπτη χωρίς την χρήση αντίστασης pull up/down.

5.2.4.2 Ορισμός θύρας ως Έξοδος

Για να ορίσουμε αν μια θύρα θα είναι είσοδος ή έξοδος στο πρόγραμμα μας θα πρέπει να χρησιμοποιήσουμε την εξής εντολή:

```
GPIO.setup(channel,GPIO.OUT)
```

Σε περίπτωση που η θύρα μας είναι έξοδος μπορούμε να ορίσουμε και την αρχική κατάστασή της, δηλαδή αν θα είναι σε κατάσταση HIGH ή LOW:

```
GPIO.setup(channel, GPIO.OUT, initial=state)
```

Οπού στο **state** βάζουμε **`GPIO.HIGH`** ή **`GPIO.LOW`**

5.2.5 Αρχικοποίηση περισσότερων θυρών/καναλιών

Είναι εφικτή η αρχικοποίηση πολλαπλών θυρών με μιας, χρησιμοποιώντας μια λίστα με τις επιθυμητές θύρες:

```
chan_list=[11,12]#Δημιουργία λίστας
GPIO.setup(chan_list,GPIO.OUT)
```

5.2.6 Είσοδος από θύρα

Για να διαβάσουμε δεδομένα από μια θύρα χρησιμοποιούμε την εντολή:

```
GPIO.input(channel)
```

όπου **channel** ο αριθμός της θύρας από την όποια θα διαβάσουμε. Τα δεδομένα θα είναι της μορφής HIGH/LOW.

5.2.7 Έξοδος από θύρα

Για να γράψουμε δεδομένα σε μια θύρα χρησιμοποιούμε την εντολή:

```
GPIO.output(channel, state)
```

Όπου **channel** ο αριθμός της θύρας στην οποία θα γράψουμε.

Όπου **state** η τιμή που θα γράψουμε στην θύρα και η οποία μπορεί να είναι είτε HIGH/1/True είτε LOW/0/False.

5.2.8 Έξοδος σε πολλαπλές θύρες

Είναι εφικτό να δώσουμε έξοδο σε πολλαπλές θύρες από την ίδια εντολή χρησιμοποιώντας πίνακα θυρών, αλλά και πίνακα με την τιμή που θα πάρει κάθε θύρα ξεχωριστά:

```
chan_list=[11,12]#πίνακας θυρών


# θέτει όλες της εξόδους σε GPIO.LOW
GPIO.output(chan_list,GPIO.LOW)


# θέτει την πρώτη σε HIGH και την δεύτερη σε LOW
GPIO.output(chan_list,(GPIO.HIGH, GPIO.LOW))
```

5.2.9 Εύρεση κατάστασης λειτουργίας μιας θύρας

Μας είναι εφικτό να βρούμε σε τι κατάσταση έχει τεθεί μια θύρα με την εντολή:

```
func = GPIO.gpio_function(pin)
```

Όπου **pin** ο αριθμός της θύρας.

Όπου **func** η μεταβλητή στην οποία θα αποθηκευτή η κατάσταση της θύρας. Οι πιθανές καταστάσεις που μπορεί να έχει μια θύρα GPIO είναι οι εξής: GPIO.IN, GPIO.OUT, GPIO.SPI, GPIO.I2C, GPIO.HARD_PWM, GPIO.SERIAL, GPIO.UNKNOWN

5.2.10 Απελευθέρωση θυρών (Cleanup)

Όταν τερματίζει το πρόγραμμα μας θα πρέπει να απελευθερώνουμε της θύρες που χρησιμοποιήσαμε ώστε να είναι εφικτή η χρήση τους από τα επόμενα προγράμματα που θα τρέξουμε. Για να γίνει αυτό, στο τέλος κάθε εφαρμογής μας θα πρέπει να γράφουμε την εντολή:

```
GPIO.cleanup()
```

Αυτή η εντολή επίσης θέτει και τον τρόπο αρίθμησης των θυρών GPIOσε **None**.

5.2.11 Διακοπές (Interrupts) και αναγνώριση Ακμής (Edge detection)

Μια ακμή είναι η εναλλαγή της κατάστασης από LOW σε HIGH (rising edge) ή από HIGH σε LOW (falling edge). Για να είναι το πρόγραμμα μας σε θέση να αναγνωρίσει μια ακμή μας παρέχονται δυο συναρτήσεις, η **wait_for_edge()** και η **event_detected().**

5.2.11.1 Η συνάρτηση wait_for_edge()

Η Λειτουργία της συνάρτησης wait_for_edge() είναι να διακόπτει την λειτουργία του προγράμματος μέχρι να γίνει αναγνώριση μιας ακμής.

```
#Απλή σύνταξη εντολής
GPIO.wait_for_edge(channel,mode)


#Εδώ περιμένει 5 δευτερόλεπτα να δοθεί είσοδος
channel =GPIO.wait_for_edge(channel, GPIO_RISING, timeout=5000)
if channel is None:
   print('Timeout occurred')
else:
   print('Edge detected on channel', channel)
```

Όπου **channel** η θύρα.

Όπου στο **mode** μπαίνει μια από αυτές τις εντολές: GPIO.RISING, GPIO.FALLING, GPIO.BOTH

Υπάρχει παράδειγμα χρήσης της εντολής στην σελίδα 91.

5.2.11.2 Η συνάρτηση event_detected()

Η συνάρτηση event_detected() είναι σχεδιασμένη ώστε να μπαίνει σε μια δομή επανάληψης.

```
#Απλή σύνταξη εντολής
GPIO.add_event_detect(channel,mode)


#Έλεγχος για το αν έχει αναγνωρίσει ακμή
if GPIO.event_detected(channel):
   print('πατήθηκε πλήκτρο')
```

Όπου **channel** η θύρα.

Όπου στο **mode** μπαίνει μια από αυτές τις εντολές: GPIO.RISING, GPIO.FALLING, GPIO.BOTH

Υπάρχει παράδειγμα χρήσης της εντολής στην σελίδα 92.

5.2.11.3 Αναγνώριση ακμής χωρίς να διακόπτεται το πρόγραμμα

Το RPi.GPIO τρέχει ένα παράλληλο thread που υπάρχει για συναρτήσεις τύπου callback. Αυτό σημαίνει ότι μπορούμε να περιμένουμε να συμβεί ένα γεγονός (event) χωρίς να είναι απαραίτητη η διακοπή του προγράμματος.

Ακολουθεί παράδειγμα χρήσης συνάρτησης callback:

```
def my callback(channel):
   print('Αυτό είναι μια callback συνάρτηση αναγνώρισης ακμής!')
   print('Αναγνωρίστηκε ακμή στην θύρα %s'%channel)
   print('Αυτή η συνάρτηση τρέχει σε ξεχωριστό thread από το βασικό πρόγραμμα')
   #προσθήκηαναγνώρισηςακμήςστο channel
   GPIO.add_event_detect(channel, GPIO.RISING, callback=my_callback,
   bouncetime = 200)
...η συνέχεια της εφαρμογής...
```

Όπου **bouncetime** ο χρόνος αναμονής για την αποφυγή κλυδωνισμού.

Υπάρχει παράδειγμα χρήσης της εντολής στην σελίδα 94.

5.2.12 Απενεργοποίηση αναγνώρισης ακμής/γεγονότος (event)

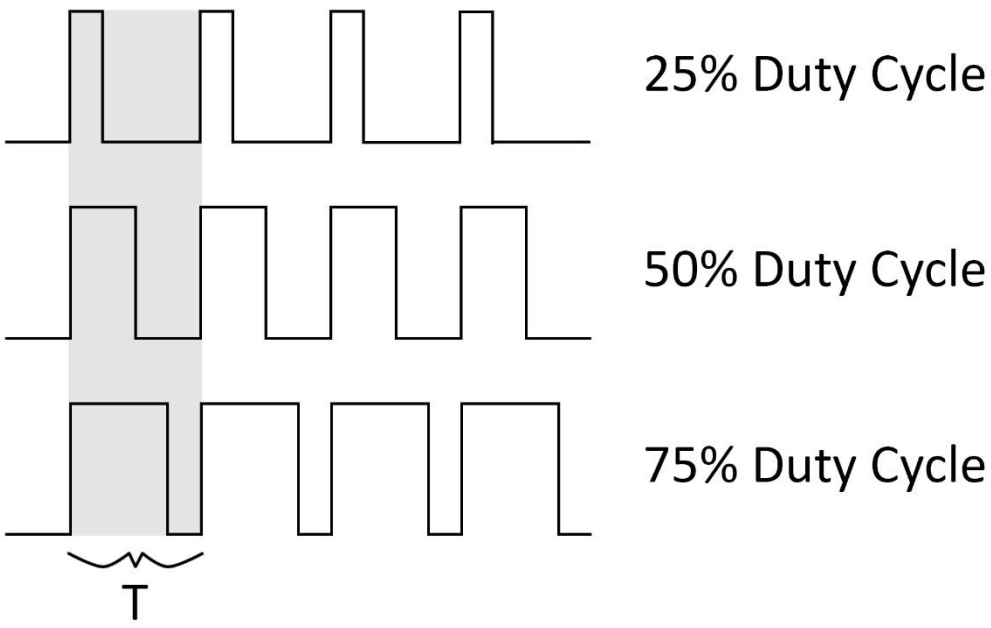
Η συγκεκριμένη εντολή είναι χρήσιμη αν θελήσουμε σε κάποιο σημείο του προγράμματος να παύσει η αναγνώριση ακμής για κάποιο κανάλι/θύρα.

```
GPIO.remove_event_detect(channel)
```

5.2.13 Προετοιμασία και χρήση του PWM

Το PWM (Pulse With Modulation) είναι ο τρόπος με τον όποιο μπορούμε να ρυθμίσουμε μια GPIO θύρα έτσι ώστε να παράγει μια ψηφιακή περιοδική κυμματομορφή. Με την χρήση του PWM είναι εφικτό να χειριστούμε φορτία όπως κινητήρες servo ή να ρυθμίσουμε την φωτεινότητα μιας οθόνης.

Το σήμα που παράγεται εξαρτάται από την *συχνότητα* και το *Duty cycle,* όπου Duty cycle είναι ο χρόνος στον οποίο μια θύρα βρίσκεται σε HIGH κατά το διάστημα μιας περιόδου.



Σχήμα 5.2.13.1 (παλμοί PWM)

Για να προετοιμάσουμε έναν παλμό PWM χρησιμοποιούμε την εξής εντολή:

```
p=GPIO.PWM(channel,frequency)
```

όπου **channel** η θύρα, και **frequency**η συχνότητα.

Για να ορίσουμε το dutycycle και να ξεκινήσει ταυτόχρονα η παραγωγή του παλμού PWM χρησιμοποιούμε την εξής εντολή:

```
p.start(dc)
```

Όπου dc βάζουμε το επιθυμητό dutycycle. Το εύρος των τιμών που μπορεί να πάρει το dutycycle κυμαίνεται από “0.0” μέχρι “100.0”.

Για να αλλάξουμε την συχνότητα χρησιμοποιούμε αυτήν την εντολή:

```
p.ChangeFrequency(freq)
```

Όπου freq η συχνότητα σε Hz.

Για να αλλάξουμε το dutycycle χρησιμοποιούμε την εντολή:

```
p.ChangeDutyCycle(dc)
```

Όπου dc βάζουμε το επιθυμητό dutycycle. Το εύρος των τιμών που μπορεί να πάρει το dutycycle κυμαίνεται από “0.0” μέχρι “100.0”.

Για να διακόψουμε την παραγωγή του PWM πληκτρολογούμε:

```
p.stop()
```

5.2.14 Πληροφορίες σχετικά με την έκδοση Raspberry και την έκδοση του RPi.GPIO

Μπορούμε να δούμε τις πληροφορίες σχετικά με το Raspberry με την εντολή:

```
GPIO.RPI_INFO
```

Για να δούμε πληροφορίες σχετικά με την έκδοση της πλακέτας του Raspberry χρησιμοποιούμε τις εντολές:

```
GPIO.RPI_REVISION
```

Μπορούμε να δούμε την τρέχουσα έκδοση του RPi.GPIO με την εντολή:

```
GPIO.VERSION
```

5.3 Είσοδος από διακόπτη (Push button)

Παρακάτω θα εξηγηθούν οι τρόποι με τους οποίους μπορούμε να συνδέσουμε διακόπτη μπουτόν και να διαβάσουμε είσοδο από αυτόν. Υπάρχουν δύο τρόποι με τους οποίους μπορούμε να συνδέσουμε έναν διακόπτη στο σύστημα μας, ανάλογα με την αρχική κατάσταση που θέλουμε να έχει ο διακόπτης μας.

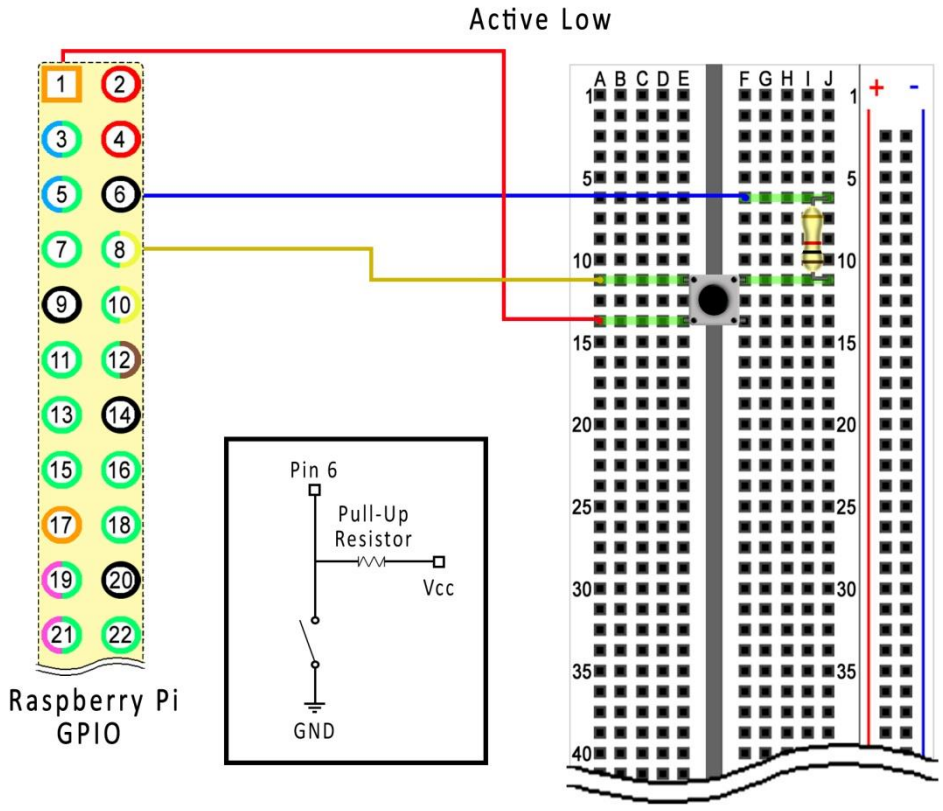
Για το πείραμα θα χρειαστούμε:

1 x διακόπτη push button.

1 x 1ΚΩ αντίσταση.

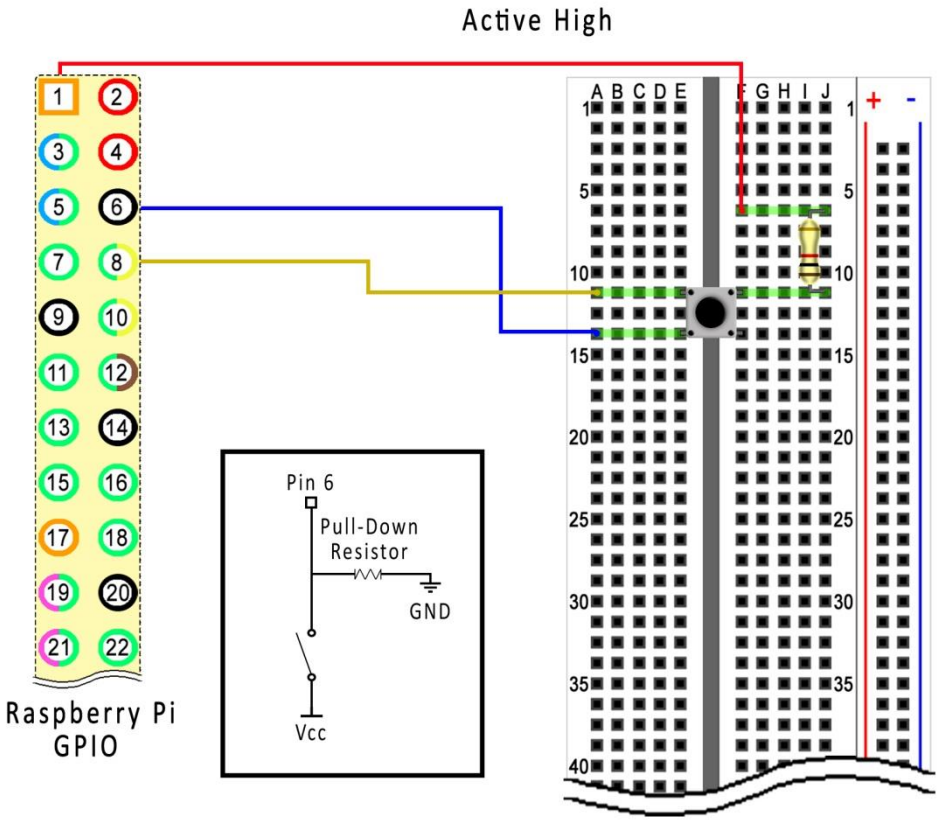
1 x Ράστερ και καλώδια.

Οι αρχικές καταστάσεις είναι δύο. Στην μια κατάσταση ο διακόπτης διατρέχεται μόνιμα από τάση μέχρις ότου να διακοπεί η διέλευση τάσης με το πάτημα του διακόπτη η συγκεκριμένη διασύνδεση του διακόπτη ονομάζεται **Active Low**.



Σχήμα 5.2.1 (active low τρόπος σύνδεσης διακόπτη)

Στην δεύτερη κατάσταση ο διακόπτης είναι γειωμένος και μόλις πατηθεί επιτρέπει την διέλευση τάσης. Η διασύνδεση του διακόπτη με αυτόν τον τρόπο ονομάζεται **Active High**.



Σχήμα 5.2.2 (active high τρόπος σύνδεσης διακόπτη)

Για να εξασφαλίσουμε την ορθή λειτουργία του διακόπτη θα πρέπει να προσθέσουμε και μια αντίσταση στο κύκλωμα. Σκοπός της αντίστασης είναι να διατηρεί σταθερή την ουδέτερη κατάσταση του διακόπτη, δηλαδή όταν ο διακόπτης δεν είναι πατημένος.

Όταν η ουδέτερη κατάσταση του διακόπτη είναι να διατηρεί τάση high τότε η αντίσταση που συνδέεται στο κύκλωμα ονομάζεται **pull-up resistor** (Σχήμα 5.2.1). Αν η ουδέτερη κατάσταση του διακόπτη είναι να διατηρεί τάση low, τότε η αντίσταση που συνδέεται στο κύκλωμα ονομάζεται **pull-down resistor** (Σχήμα 5.2.2).

Τώρα που είδαμε τους δυο βασικούς τρόπους διασύνδεσης θα εξηγηθούν οι τρόποι με τους οποίους μπορούμε να προγραμματίσουμε τους διακόπτες. Υπάρχουν τρείς τρόποι με τους οποίους είναι εφικτός ο προγραμματισμός εισόδου από διακόπτες. Τα προγράμματα είναι γραμμένα ώστε να λειτουργήσουν με τον τρόπο διασύνδεσης Active Low.

Παύση προγράμματος μέχρι να συμβεί αναγνώριση ακμής.

```
1:  #!/usr/bin/env python
2:  import time
3:  import RPi.GPIO as GPIO
4:
5:  GPIO.setmode(GPIO.BOARD)
6:  GPIO.setup(8, GPIO.IN)
7:
8:  print('Push the button to stop looping')
9:  GPIO.wait_for_edge(8, GPIO.FALLING)
10: print('Button was Pressed')
11: GPIO.cleanup()
```

Έξοδος: Θα εμφανιστεί το μήνυμα “Push the button to stop looping” και το πρόγραμμα θα περιμένει να πατηθεί ο διακόπτης. Μόλις πατηθεί ο διακόπτης θα εμφανίσει τα μηνύματα “Button was Pressed” και θα τερματίσει η εφαρμογή.

Ανάλυση:

1: Στην γραμμή 1 κάνουμε τον κώδικα εκτελέσιμο μέσω του Bash.

2,3: Γίνεται εισαγωγή των απαραίτητων βιβλιοθηκών.

5: Ορίζουμε τον τρόπο αρίθμησης των θυρών GPIO σε φυσική αρίθμηση.

6: Ορίζουμε την θύρα 8 (GPIO 14 σε BCM) ως είσοδο.

8: Τυπώνει μήνυμα στην οθόνη.

9: Αναμονή μέχρι να γίνει αναγνώριση ακμής από την θύρα 8.

10: Τυπώνει μήνυμα στην οθόνη.

11: Απελευθερώνουμε τις θύρες που δεσμεύσαμε, και γίνετε επανεκκίνηση του GPIO.

Έλεγχος σε επανάληψη μέχρι να συμβεί ακμή.

```
1:  #!/usr/bin/env python
2:  import time
3:  import RPi.GPIO as GPIO
4:
5:  GPIO.setmode(GPIO.BOARD)
6:  GPIO.setup(8, GPIO.IN)
7:
8:  print('Push the button to stop looping')
9:  GPIO.add_event_detect(8, GPIO.FALLING)
10:
11: whileTrue:
12:     time.sleep(0.01)
13:     if GPIO.event_detected(8):
14:         print('Button pressed')
15:         break
16: GPIO.cleanup()
```

Έξοδος: Θα εμφανιστεί το μήνυμα “Push the button to stop looping” και το πρόγραμμα θα περιμένει να πατηθεί ο διακόπτης. Μόλις πατηθεί ο διακόπτης θα εμφανίσει τα μηνύματα “Button pressed” και θα τερματίσει η εφαρμογή.

Ανάλυση:

1: Στην γραμμή 1 κάνουμε τον κώδικα εκτελέσιμο μέσω του Bash.

2,3: Γίνεται εισαγωγή των απαραίτητων βιβλιοθηκών.

5: Ορίζουμε τον τρόπο αρίθμησης των θυρών GPIO σε φυσική αριθμηση.

6: Ορίζουμε την θύρα 8 (GPIO 14 σε BCM) ως είσοδο.

Ανάλυση (Συνέχεια):

8: Τυπώνει μήνυμα στην οθόνη.

9: Ορισμός αναγνώρισης για πτώση ακμής στην θύρα 8.

11: Είσοδος σε ατέρμονα βρόχο.

12: Παύση για 10ms ώστε ο επεξεργαστής να αναλάβει άλλες διεργασίες.

13: Έλεγχος για το αν έγινε αναγνώριση ακμής από την θύρα 8.

14: Τυπώνει μήνυμα στην οθόνη.

15: Έξοδος από τον ατέρμονα βρόχο.

16: Απελευθερώνουμε τις θύρες που δεσμεύσαμε και γίνεται επανεκκίνηση του GPIO.

Αναγνώριση ακμής σε παράλληλη διεργασία (thread).

```
1:  #!/usr/bin/env python
2:  import time
3:  import RPi.GPIO as GPIO
4:
5:  GPIO.setmode(GPIO.BOARD)
6:  GPIO.setup(8, GPIO.IN)
7:
8:  def my_callback_one(channel):
9:      print('callback one')
10:
11: GPIO.add_event_detect(8, GPIO.RISING, callback=my_callback_one)
12: print('Push the button to stop looping')
13: whileTrue:
14:     time.sleep(0.01)# wait 10 ms to give CPU chance to do other things
15:     if GPIO.event_detected(8)
16:         break
17: GPIO.cleanup()
```

Έξοδος: Θα εμφανιστεί το μήνυμα “Push the button to stop looping” και το πρόγραμμα θα περιμένει να πατηθεί ο διακόπτης. Μόλις πατηθεί ο διακόπτης θα εμφανίσει τα μηνύματα “callback one” και θα τερματίσει η εφαρμογή.

Ανάλυση:

1: Στην γραμμή 1 κάνουμε τον κώδικα εκτελέσιμο μέσω του Bash.

2,3: Γίνεται εισαγωγή των απαραίτητων βιβλιοθηκών.

5: Ορίζουμε τον τρόπο αρίθμησης των θυρών GPIO σε φυσικό τρόπο αρίθμησης.

6: Ορίζουμε την θύρα 8(GPIO 14 σε BCM) ως είσοδο.

8: Δημιουργία συνάρτησης callback που θα εκτελεστεί μόλις γίνει αναγνώριση ακμής.

Ανάλυση (Συνέχεια):

9: Τυπώνει μήνυμα στην οθόνη.

11: Ορισμός αναγνώρισης για πτώση ακμής στην θύρα 8. Η συγκεκριμένη αναγνώριση πτώσης ακμής θα εκτελείται σε παράλληλη διεργασία.

12: Τυπώνει μήνυμα στην οθόνη.

13: Είσοδος σε ατέρμονα βρόχο.

14: Παύση για 10ms ώστε ο επεξεργαστής να αναλάβει άλλες διεργασίες.

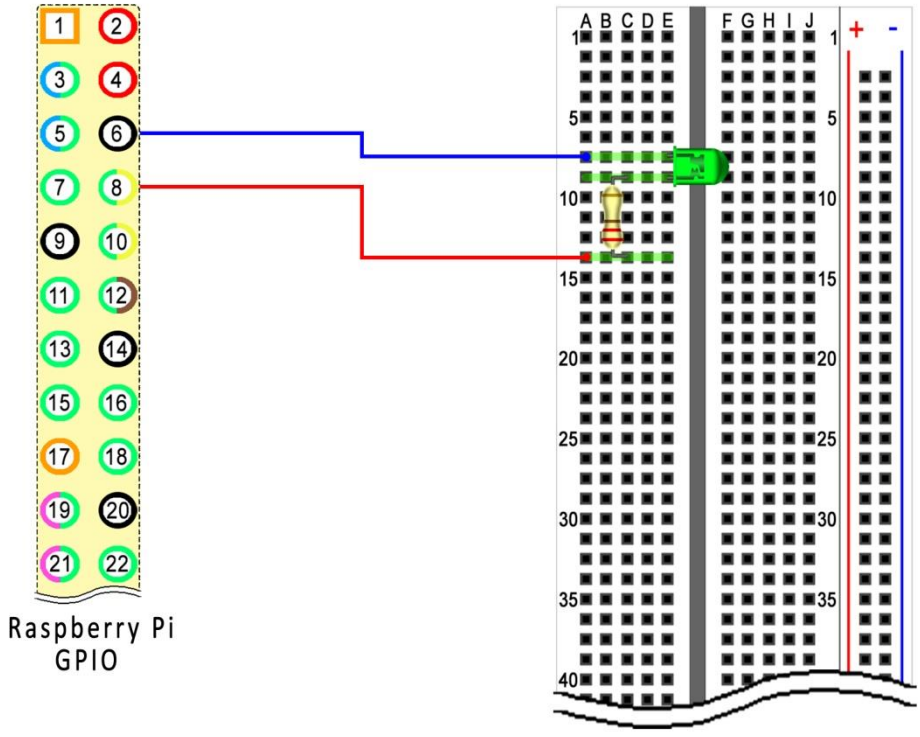
15: Έλεγχος για το αν έγινε αναγνώριση ακμής από την θύρα 8.

16: Έξοδος από τον ατέρμονα βρόχο.

17: Απελευθερώνουμε τις θύρες που δεσμεύσαμε, και γίνεται επανεκκίνηση του GPIO.

5.4 Έξοδος σε δίοδο φωτοεκπομπής Led

Παρακάτω θα εξηγηθεί ένας τρόπος ώστε να συνδέσουμε και να προγραμματίσουμε μια δίοδο φωτοεκπομπής Led. Υπενθυμίζεται ότι το μακρύ ποδαράκι του Led (άνοδος) είναι αυτό που συνδέετε πάντα από την “θετική” πλευρά της τάσης τροφοδοσίας. Φροντίζουμε πάντα να υπάρχει και μια αντίσταση συνδεδεμένη σε σειρά με μια φωτοδίοδο Led ώστε να αποτρέψουμε την διέλευση μεγάλου ρεύματος μέσα από το στοιχείο, κάτι που θα είχε ως αποτέλεσμα την καταστροφή του. Δεν έχει σημασία από ποια πλευρά θα συνδέσουμε την αντίσταση. Για λόγους ευκολίας χρησιμοποιούμε ως συνήθως μια αντίσταση των 220Ω, αλλά είναι εφικτή και η χρήση μεγαλυτέρων αντιστάσεων.



Σχήμα 5.1.1 (σύνδεση διόδου φωτοεκπομπής)

Θα χρειαστούμε:

1 x 220Ω αντίσταση.

1 x Δίοδο φωτοεκπομπής LED.

1 x Ράστερ και καλώδια.

Όπως βλέπουμε στο σχήμα 5.1.1 η κάθοδος (κοντό ποδαράκι) του Led καταλήγει στην θύρα 6 του Raspberry που είναι **γείωση** (GND), η άνοδος (μακρύ ποδαράκι) του Led συνδέεται στην αντίσταση 220Ω και η αντίσταση με την σειρά της στην **θύρα 8** (GPIO 14 σε BCM) του Raspberry.

Ο παρακάτω κώδικας σε python θα χρησιμοποιηθεί ώστε να δώσουμε έξοδο στο Led.

```
1:  #!/usr/bin/env python
2:  Import time
3:  import RPi.GPIO as GPIO
4:
5:  GPIO.setmode(GPIO.BOARD)
6:  GPIO.setup(8, GPIO.OUT)
7:
8:  GPIO.output(8,True)
9:  time.sleep(2)
10: GPIO.output(8,False)
11:
12: GPIO.cleanup()
```

Έξοδος: Το Led θα ανάψει για 2 δευτερόλεπτα και ύστερα θα σβήσει.

Ανάλυση:

1: Στην γραμμή 1 κάνουμε τον κώδικα εκτελέσιμο μέσω του Bash.

2,3: Γίνεται εισαγωγή των απαραίτητων βιβλιοθηκών.

5: Ορίζουμε τον τρόπο αρίθμησης των θυρών GPIO σε φυσικό.

6: Ορίζουμε την θύρα 8 (GPIO 14 σε BCM) ως έξοδο.

8: Θέτουμε την τιμή της θύρας 8 σε HIGH (ανάβει το Led).

9: Βάλαμε χρονική καθυστέρηση 2 δευτερολέπτων ώστε να φανεί το αποτέλεσμα.

10: Θέτουμε την τιμή της θύρας 8 σε LOW (σβήνει το Led).

12: Απελευθερώνουμε τις θύρες που δεσμεύσαμε και γίνεται επανεκκίνηση του GPIO.

5.5 Σύνδεση οθόνης LCD (2X16) με την χρήση τον ακροδεκτών GPIO

Εδώ θα εξηγηθεί ένας τρόπος ώστε να συνδεθεί μια LCD οθόνη 16X2 στο Raspberry Pi.

Για αρχή, χρειάζεται να κατεβάσουμε της βιβλιοθήκες που θα μας επιτρέψουνε να επικοινωνήσουμε με την LCD οθόνη. Παρακάτω εξηγούνται τα βήματα:

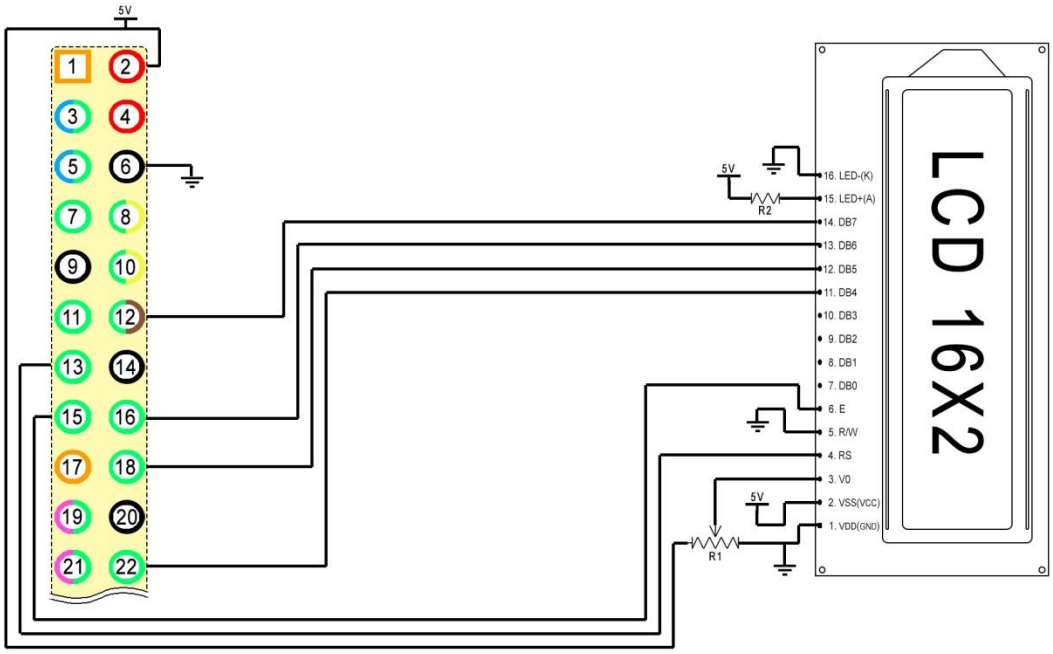
Ανοίγουμε το terminal και πληκτρολογούμε της εξής εντολές:

```
git clone https://github.com/adafruit/Adafruit_Python_GPIO.git
cd Adafruit_Python_GPIO
sudo python setup.py install
cd ..
git clone https://github.com/adafruit/Adafruit_Python_CharLCD.git
cd Adafruit_Python_CharLCD
sudo python setup.py install
```

Μόλις τελειώσουν οι εγκαταστάσεις κάνουμε επανεκκίνηση το σύστημα ώστε να εφαρμοστούν οι αλλαγές.

Σημειώνεται ότι η βιβλιοθήκη της οθόνης LCD είναι γραμμένη σε Python 2 άρα για να προγραμματίσουμε τον αισθητήρα θα χρησιμοποιήσουμε το IDLE για Python2.

Είναι εφικτό να συνδεθεί μια οθόνη 16X2 οποιουδήποτε κατασκευαστή αρκεί να δοθεί σημασία στο αν η φωτοδίοδος (LED) έχει δικιά της αντίσταση ή αν χρειαστεί να προσθέσουμε μια στο κύκλωμα. Οι ακροδέκτες σε αυτές της οθόνες έχουνε ως συνήθως την ίδια διάταξη, προτείνεται όμως να συμβουλευτούμε το εγχειρίδιο του κατασκευαστή(datasheet).



Σχήμα 5.5.1 (σύνδεση οθόνης LCD)

Στο Σχήμα 5.5.1 βλέπουμε τον τρόπο σύνδεσης της οθόνης LCD 16X2 στο Raspberry. Η αντίσταση R2 είναι προαιρετική και εξαρτάται από το αν υπάρχει ενσωματωμένη αντίσταση στην φωτοδίοδος(LED) της οθόνης, αν δεν υπάρχει ενσωματωμένη αντίσταση θα πρέπει να προσθέσουμε μια αντίσταση 220Ω. Το R1 είναι το ποτενσιόμετρο που ελέγχει την αντίθεση της οθόνης. Οι εντολές σε γλώσσα Python για να χρησιμοποιηθεί η οθόνη προϋποθέτουν το εξής τμήμα δηλώσεων (για μονόχρωμη οθόνη):

```
#!/usr/bin/python
import time
import Adafruit_CharLCD as LCD
#Αρχικοποίηση των ακροδεκτών του Raspberry Pi:
lcd_rs = 27
lcd_en = 22
lcd_d4 = 25
lcd_d5 = 24
lcd_d6 = 23
lcd_d7 = 18
lcd_backlight = 4 #για την περίπτωση που το raspberry θα ελέγχει τη φωτεινότητα.
#Ορίζουμε το μέγεθος της γραμμής και της στήλης για 16x2 LCD οθόνη.
lcd_columns = 16 #μέγεθος στήλης.
lcd_rows = 22 #μέγεθος γραμμής.
#Αρχικοποίηση της LCD οθόνης χρησιμοποιώντας τους ακροδέκτες που ορίστηκαν προηγουμένως.
lcd = LCD.Adafruit_CharLCD(lcd_rs,lcd_en,lcd_d4,lcd_d5,lcd_d6,lcd_d7,
lcd_columns,lcd_rows,lcd_backlight)
```

Σημείωση: Παρατηρούμε ότι στην αρχικοποίηση τον ακροδεκτών του Raspberry Pi έχουμε την ελευθερία να επιλέξουμε εμείς ποιους ακροδέκτες του Raspberry θέλουμε να δεσμεύσουμε ώστε να επικοινωνήσει με την LCD οθόνη. Πρέπει όμως να φροντίσουμε ώστε να γίνει η σωστή αρχικοποίηση και αντιστοίχιση των ακροδεκτών του Raspberry με την οθόνη LCD, όπως για παράδειγμα ο ακροδέκτης **27** του Raspberry που έχει αρχικοποιηθεί ως **lcd_rs** να συνδέεται με τον ακροδέκτη **RS** της οθόνης LCD.
Έχουμε αρχικοποιήσει και τον ακροδέκτη **lcd_backlight** προαιρετικά για την περίπτωση που θελήσουμε να τον χρησιμοποιήσουμε σε μελλοντικές εφαρμογές, στο παράδειγμα μας δεν γίνεται χρήση του ακροδέκτη **lcd_backlight (**μιας και συνδέσαμε ποτενσιόμετρο, ώστε να ρυθμίζουμε την φωτεινότητα της LCD οθόνης.) Αν χρειαστεί να χρησιμοποιήσουμε τον ακροδέκτη **lcd_backlight** θα πρέπει να ρυθμιστεί ο ακροδέκτης (στην περίπτωση μας ο ακροδέκτης **4**) σε λειτουργία PWM, ώστε να μπορούμε να ρυθμίσουμε την φωτεινότητα μέσω του Raspberry.

Όπως φαίνεται στο διάγραμμά διασύνδεσης θα χρησιμοποιήσουμε μόνο 6 ακροδέκτες (λόγω του ότι θα χρησιμοποιήσουμε την LCD οθόνη se 4bit mode) για την επικοινωνία του Raspberry με την οθόνη LCD καθώς και άλλους 2 ακροδέκτες για την τροφοδοσία της οθόνης (VCC,GND).

Οι βασικές εντολές για να χρησιμοποιηθεί η οθόνη είναι οι εξής:

lcd.home() = Μετακινεί τον κέρσορα στην πρώτη γραμμή και πρώτη στήλη.

lcd.clear() = Καθαρίζει/Μηδενίζει την οθόνη LCD.

lcd.set_cursor(col, row) = Θέτει τον κέρσορα στην γραμμή (row) και στήλη (col) που ορίσαμε.

lcd.enable_display(True) = Ενεργοποιεί (True)/Απενεργοποιεί (False) την οθόνη.

lcd.show_cursor(True) = Εμφανίζει (True)/Αποκρύπτει (False) τον κέρσορα.

lcd.blink(True) = Ενεργοποιεί (True)/Απενεργοποιεί (False) το αναβόσβημα του κέρσορα.

lcd.move_left() = Μετακινεί το κείμενο μια θέση προς τα αριστερά.

lcd.move_right() = Μετακινεί το κείμενο μια θέση προς τα δεξιά.

lcd.set_left_to_right() = Ορίζει την κατεύθυνση εισαγωγής του κειμένου από αριστερά σε δεξιά.

lcd.set_right_to_left() = Ορίζει την κατεύθυνση εισαγωγής του κειμένου από δεξιά σε αριστερά.

lcd.autoscroll(True) = Αν είναι ενεργό (True) θα επιτρέπει την αυτόματη κύλιση του κείμενου.

lcd.message(text) = Θα εμφανίσει στην οθόνη LCD το κείμενο που περιέχει η μεταβλητή **text** (μπορεί να περιέχει και αλλαγή γραμμής).

lcd.set_backlight(backlight) = Ρυθμίζει την φωτεινότητα της οθόνης, στην περίπτωση που το PWM είναι ενεργό οι τιμές φωτεινότητας ξεκινάνε από 0.0 (ανενεργό) μέχρι 1.0 (μέγιστη ένταση φωτισμού)

5.5.1 Απλό παράδειγμα χρήσης LCD οθόνης

Στο παράδειγμα που ακολουθεί θα τυπωθεί στην οθόνη LCD το κείμενο “Hello World!”. Για να είναι εφικτή η εκτέλεση του προγράμματος θα πρέπει να ανοίξουμε το περιβάλλον IDLE για Python2.

```
 1:  #!/usr/bin/python
 2:  import time
 3:  import Adafruit_CharLCD as LCD
 4:  lcd_rs = 27
 5:  lcd_en = 22
 6:  lcd_d4 = 25
 7:  lcd_d5 = 24
 8:  lcd_d6 = 23
 9:  lcd_d7 = 18
10:  lcd_backlight = 4
11:  lcd_columns = 16
12:  lcd_rows = 22
13:  lcd = LCD.Adafruit_CharLCD(lcd_rs,lcd_en,lcd_d4,lcd_d5,lcd_d6,lcd_d7\
14:  ,lcd_columns,lcd_rows,lcd_backlight)
15:  lcd.message("Hello World!")
```

Έξοδος: Θα εμφανιστούν το κείμενο “Hello World!” στην οθόνη LCD.

Ανάλυση:

1: Στην γραμμή 1 κάνουμε τον κώδικα εκτελέσιμο μέσω του Bash.

2,3: Γίνεται εισαγωγή των απαραίτητων βιβλιοθηκών.

4-14: Αρχικοποίηση ρυθμίσεων οθόνης LCD.

15: Με την βοήθεια της συνάρτησης message τυπώνεται το κείμενο “Hello World!” στην LCD οθόνη.

5.6 Σύνδεση αισθητήρα υγρασίας/θερμοκρασίας (HDC100X)

Παρακάτω θα εξηγηθεί ο τρόπος εγκατάστασης και χρήσης της βιβλιοθήκης του αισθητήρα υγρασίας/θερμοκρασίας και ύστερα θα εξηγηθεί ο τρόπος σύνδεσης του αισθητήρα με το Raspberry καθώς και ο τρόπος προγραμματισμού του.

Σημειώνεται ότι η βιβλιοθήκη του αισθητήρα υγρασίας είναι γραμμένη σε Python 2 άρα για να προγραμματίσουμε τον αισθητήρα θα χρησιμοποιήσουμε το IDLE για Python2.

5.6.1 Εγκατάσταση βιβλιοθήκης για τον αισθητήρα HDC100X

Για να είναι εφικτή η χρήση του αισθητήρα θα πρέπει να κατεβάσουμε την βιβλιοθήκη του. Παρακάτω εξηγούνται τα απαραίτητα βήματα:

Ανοίγουμε το terminal και πληκτρολογούμε τις εντολές:

```
git clone https://github.com/scanner/HDC100X_Python.git
cd HDC100X_Python
sudo python setup.py install
```

Μόλις τελειώσει η εγκατάσταση κάνουμε επανεκκίνηση του συστήματος ώστε να εφαρμοστούν οι αλλαγές.

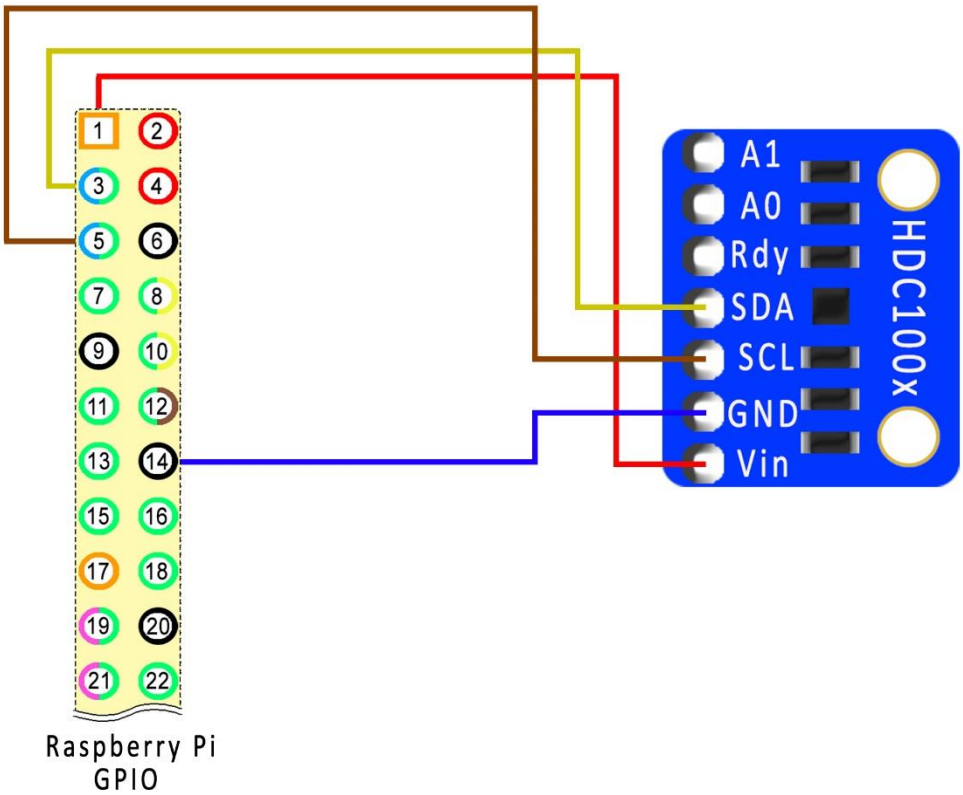
5.6.2 Ενεργοποίηση του πρωτοκόλλου I2C

Λόγω του ότι ο αισθητήρας χρησιμοποιεί το πρωτόκολλο I2C για να μας είναι εφικτή η επικοινωνία, με αυτόν θα πρέπει να το ενεργοποιήσουμε από τις ρυθμίσεις του raspberry.

Για την ενεργοποίηση πηγαίνουμε από την έναρξη στο Preferences/Raspberry Pi Configuration και στο παράθυρο που μας ανοίγει επιλέγουμε την καρτέλα Interfaces ενεργοποιούμε το I2C και πατάμε OK.

5.6.3 Σύνδεση του αισθητήρα HDC100X με το Raspberry και προγραμματισμός του

Ο αισθητήρας για να λειτουργήσει θα πρέπει να συνδεθεί όπως στο σχήμα 5.6.3.1



Σχήμα 5.6.3.1 (σύνδεση του αισθητήρα υγρασίας/θερμοκρασίας)

Θα χρειαστούμε:

Τον αισθητήρα HDC100x.

1 x Ράστερ και καλώδια.

Ο αισθητήρας τροφοδοτείται με τάση 3.3V. Λόγω του ότι ο αισθητήρας χρησιμοποιεί το πρωτόκολλο I2C για να λειτουργήσει, πρέπει αναγκαστικά να χρησιμοποιήσουμε τις θύρες 3 και 5. Η θύρα 3 (GPIO 2 σε BCM) όταν είναι σε λειτουργία I2C αντιστοιχεί στον ακροδέκτη SDA. Η θύρα 5 (GPIO 3 σε BCM) όταν είναι σε λειτουργία I2C αντιστοιχεί στον ακροδέκτη SCL.

Οι βασικές εντολές για να χρησιμοποιηθεί ο αισθητήρας:

hdc = HDC100x() = Δημιουργία ενός αντικείμενου της κλάσης HDC100x ώστε να είναι εφικτή η ανάγνωση των δεδομένων από τον αισθητήρα.

hdc.dry_sensor() = Αρχικοποίηση αισθητήρα, έπειτα από μέτρηση. Είναι χρήσιμο μόνο αν περνούμε διαδοχικές μετρήσεις.

temp, humid = hdc.data() = Ανάγνωση δεδομένων από τον αισθητήρα, η συνάρτηση επιστρέφει 2 τιμές η πρώτη είναι η θερμοκρασία και η δεύτερη η υγρασία.

hdc.reset() = Επανεκκίνηση του αισθητήρα.

Ο κώδικας για να μας επιστρέψει ο αισθητήρας την θερμοκρασία και την υγρασία είναι ο εξής:

```
1:  #!/usr/bin/env python
2:  from HDC100x.HDC100x import HDC100x
3:
4:  hdc = HDC100x()
5:  temp, humid = hdc.data()
6:  print"Θερμοκρασία: {}, Υγρασία: {}".format(temp, humid)
```

Έξοδος: Θα εμφανιστούν η θερμοκρασία και η υγρασία στην οθόνη.

Ανάλυση:

1: Στην γραμμή 1 κάνουμε τον κώδικα εκτελέσιμο μέσω του Bash.

2: Γίνεται εισαγωγή της απαραίτητης βιβλιοθηκης.

4: Δημιουργούμε ένα αντικείμενο τύπου HDC100x.

5: Εμφανίζει την θερμοκρασία και υγρασία στην οθόνη, παρατηρούμε ότι η συνάρτηση print ΔΕΝ έχει παρενθέσεις λόγω του ότι γράφουμε σε Python2.

6: Δημιουργούμε τις μεταβλητές temp και humid στις οποίες θα αποθηκευτούν οι τιμές θερμοκρασίας και υγρασίας.

5.7 Σύνδεση και χρήση servo κινητήρα

Παρακάτω θα εξηγηθεί ο τρόπος σύνδεσης και προγραμματισμού κινητήρων servo. Οι Servo κινητήρες δέχονται ως είσοδο παλμό PWM ώστε να λειτουργήσουν. Οι περισσότεροι έχουνε συχνότητα λειτουργιάς κοντά στα 50Hz, ο τρόπος με τον όποιο καθορίζουμε την θέση ενός servo κινητήρα, είναι τροποποιώντας το πλάτος του παλμού. Ένας τυπικός servo κινητήρας με πλάτος παλμού 1ms περιστρέφεται στην τέρμα αριστερή θέση, με πλάτος παλμού 1.5 ms βρίσκεται στο κέντρο, και με πλάτος παλμού 2 ms περιστρέφεται στην τέρμα δεξιά θέση.
Για να υπολογίσουμε το Duty cycle για κάθε θέση (αριστερά, κέντρο, δεξιά) πρώτα θα πρέπει να βρούμε την περίοδο του παλμού. Ο τύπος για να βρούμε την περίοδο Τ είναι:

$$\pi\epsilon\rho\acute{\iota}o\delta o\varsigma\ T = \frac{1}{\Sigma\upsilon\chi\nu\acute{o}\tau\eta\tau\alpha}$$

Για παλμό 50Hz η περίοδος Τ είναι 1/50 = **0.2s** ή **20ms**. Έχοντας την περίοδο του παλμού και γνωρίζοντας ότι τα πλάτη κύματος για κάθε θέση είναι: αριστερά = **1ms**, κέντρο = **1.5ms** και δεξιά = **2ms** εφαρμόζουμε τον τύπο:

$$\textbf{duty cycle} = \frac{\pi\lambda\acute{\alpha}\tau o\varsigma\ \kappa\acute{\upsilon}\mu\alpha\tau o\varsigma}{\pi\epsilon\rho\acute{\iota}o\delta o\varsigma\ T}$$

$$= \pi\lambda\acute{\alpha}\tau o\varsigma\ \kappa\acute{\upsilon}\mu\alpha\tau o\varsigma\ /\left(\frac{1}{\Sigma\upsilon\chi\nu\acute{o}\tau\eta\tau\alpha}\right)$$

$$= \boldsymbol{\pi\lambda\acute{\alpha}\tau o\varsigma\ \kappa\acute{\upsilon}\mu\alpha\tau o\varsigma} * \boldsymbol{\sigma\upsilon\chi\nu\acute{o}\tau\eta\tau\alpha}$$

Έτσι το duty cycle για κάθε θέση είναι:

duty cycle (αριστερά) = πλάτος κύματος* συχνότητα = 0.001*50 = 0.05 = **5%**

duty cycle (κέντρο) =0.0015*50= 0.075 = **7.5%**

duty cycle (δεξιά) = 0.002*50= 0.1= **10%**

Ο Servo κινητήρας του συγκεκριμένου παραδείγματος είναι ο Futaba S3113. Ο συγκεκριμένος servo κινητήρας έχει συχνότητα λειτουργίας στα 50Hz και τα πλάτη κύματος για κάθε θέση είναι: αριστερά = 0.5ms, κέντρο = 1ms και δεξιά = 1.5ms. Επίσης έχουμε:

duty cycle (αριστερά) = **2.5%**

duty cycle (κέντρο) = **5%**

duty cycle (δεξιά) = **7.5%**

Τώρα που γνωρίζουμε τα duty cycle τον βασικών θέσεων του servo κινητήρα, μπορούμε να δημιουργήσουμε μια γραμμική εξίσωση ώστε να ορίσουμε την περιστροφή που θα κάνει σε μοίρες(από 0-180). Για να δημιουργήσουμε την γραμμική εξίσωση χρειαζόμαστε δυο σημεία. Αν θέσουμε ότι για 0 μοίρες ο servo κινητήρας θα βρίσκεται τέρμα αριστερά(2.5% duty cycle) τότε προκύπτει το σημείο (0,2.5). Άρα για 180 μοίρες ο servo κινητήρας θα βρίσκεται τέρμα δεξιά(7.5% duty cycle) και έτσι προκύπτει το δεύτερο σημείο (180,7.5). Τώρα που έχουμε δυο σημεία μπορούμε να υπολογίσουμε την κλήση της ευθείας με τον τύπο:

m=(y2-y1)/(x2-x1)

=(7.5-2.5)/(180-0)

m=5/180

Τώρα που έχουμε την κλήση της ευθείας μπορούμε να βρούμε την εξίσωση γραμμής αντικαθιστώντας σε αυτόν τον τύπο:

$$y - y1 = m(x - x1)$$

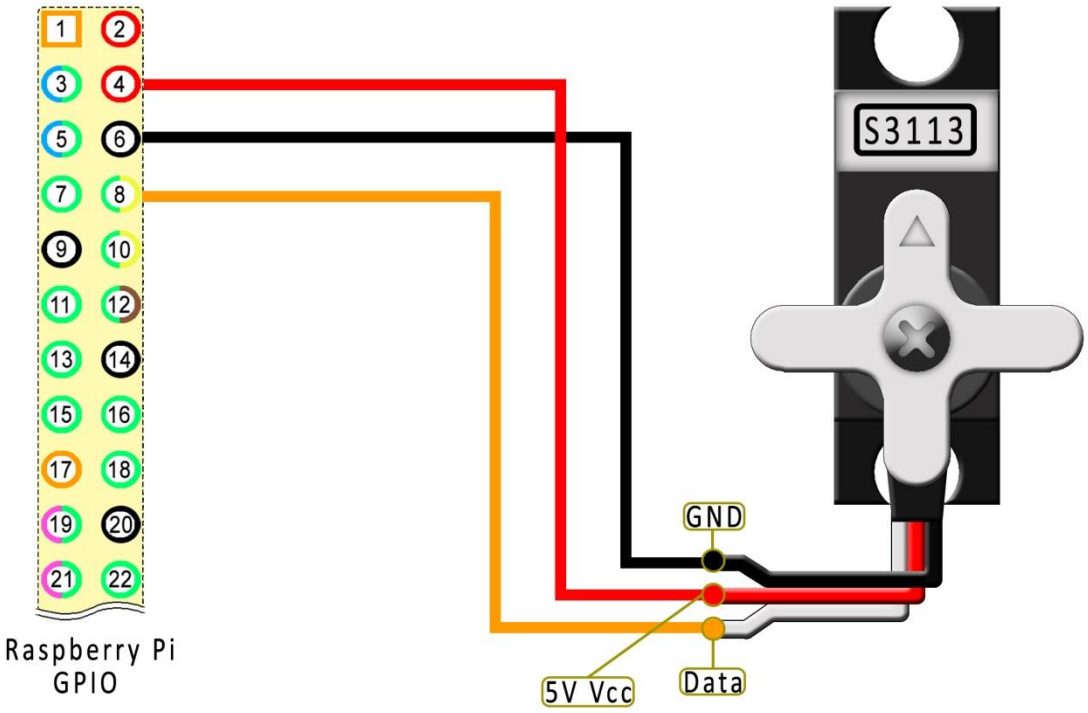
$$y - 2.5 = \frac{5}{180} * (x - 0)$$

$$\boldsymbol{y = \frac{5}{180} * x + 2.5}$$

Και έτσι έχουμε:

$$\boldsymbol{DutyCycle = \frac{5}{180} * (\text{επιθυμητή γωνία}) + 2.5}$$

Στην συνεχεία παρουσιάζεται ο τρόπος σύνδεσης του servo κινητήρα:



Σχήμα 5.7.1 (σύνδεση του servo κινητήρα)

<u>Για το πείραμα θα χρειαστούμε:</u>

Τον Servo κινητήρα.

1 x Ράστερ και καλώδια.

Στην συνέχεια παρουσιάζεται ένα πρόγραμμα στο όποιο, ο servo κινητήρας ξεκινά από το κέντρο, ύστερα πηγαίνει τέρμα αριστερά (0 μοίρες), κινείται με βήμα 45 μοίρες προς τα δεξιά και ύστερα τερματίζει καταλήγοντας στο κέντρο.

```
1:  #!/usr/bin/env python
2:  import RPi.GPIO as GPIO
3:  import time
4:  GPIO.setmode(GPIO.BOARD)
5:  GPIO.setup(8, GPIO.OUT)
6:  pwm = GPIO.PWM(8, 50) # (GPIO_Port14)channel=8 frequency=50Hz
7:  pwm.start(5) #min=0 max=100, servo S3113: L=2.5,C=5.5,R=7.5
8:  time.sleep(0.5)
```

```
 9:
10:     for x in range(0,181,45):
11:         y=(5/180)*(x)+2.5
12:         pwm.ChangeDutyCycle(y)
13:         time.sleep(0.5)
14:
15:     pwm.ChangeDutyCycle(5)
16:     time.sleep(0.5)
17:
18:     pwm.stop()
19:     GPIO.cleanup()
```

Ανάλυση:

1: Στην γραμμή 1 κάνουμε τον κώδικα εκτελέσιμο μέσω του Bash.

2,3: Γίνεται εισαγωγή των απαραίτητων βιβλιοθηκών.

4: Ορίζουμε τον τρόπο αρίθμησης των θυρών GPIO σε φυσικό.

5: Ορίζουμε την θύρα 8 (GPIO 14 σε BCM) ως έξοδο.

6: Προετοιμασία παλμού PWM στα 50Hz, στην θύρα 8.

7: Εκκίνηση παράγωγης παλμού PWM με duty cycle 5, ο συγκεκριμένος παλμός θα μετακινήσει τον servo κινητήρα στην κεντρική θέση.

8: Σε αυτό το σημείο το σύστημα περιμένει 0.5ms ώστε να προλάβει ο servo κινητήρας να περιστραφεί.

10-13: Σε αυτό το σημείο το πρόγραμμα μπαίνει μια επανάληψη, όπου η τιμή x του βρόχου καθορίζει την επιθυμητή γωνιά περιστροφής του servo κινητήρα. Η τιμή x στην συνέχεια μπαίνει στην εξίσωση, η οποία επιστρέφει το επιθυμητό duty cycle ώστε ο servo κινητήρας να περιστραφεί x μοίρες.

15-16: Σε αυτό το σημείο ο servo κινητήρας επιστρέφει στην κεντρική θέση.

19: Απελευθερώνουμε τις θύρες που δεσμεύσαμε και γίνεται επανεκκίνηση του GPIO.

5.8 Σύνδεση αισθητήρα υπερήχων

Παρακάτω θα εξηγηθεί ο τρόπος με τον όποιο είναι εφικτό να συνδέσουμε έναν αισθητήρα υπερήχων στο σύστημα μας.

Πρώτα όμως θα γίνει μια μικρή αναφορά για το πώς είναι δυνατόν να υπολογίσουμε την απόσταση με την χρήση των υπερήχων. Ο αισθητήρας εκπέμπει ένα σύνολο από υπέρηχους οι οποίοι ταξιδεύουν μέχρι το κοντινότερο φυσικό εμπόδιο και υστέρα επιστρέφουν στον αισθητήρα. Κρατάμε τις χρονικές στιγμές που επέστρεψαν ο πρώτος και ο τελευταίος υπέρηχος στον αισθητήρα. Αφαιρώντας την χρονική στιγμή του πρώτου υπέρηχου από την χρονική στιγμή του δευτέρου βρίσκουμε πόσο χρόνο χρειάζεται ο ήχος ώστε να ταξιδέψει ως το φυσικό εμπόδιο και να επιστρέψει στον αισθητήρα. Χρησιμοποιώντας τον τύπο:

$$\text{Ταχύτητα} = \frac{\text{Απόσταση}}{\text{Χρόνο}}$$

Γνωρίζοντας ότι η ταχύτητα του ήχου είναι 343 m/s (34300cm/s) στον αέρα, και γνωρίζοντας ότι θέλουμε μόνο τον χρόνο που ο ήχος χρειάστηκε να φτάσει μέχρι το αντικείμενο, συμπεραίνουμε ότι πρέπει να τον διαιρέσουμε με το 2. Ο τύπος θα πάρει την εξής μορφή:

$$34300 = \frac{Απόσταση}{Χρόνο/2}$$

Καταλήγουμε στον τύπο:

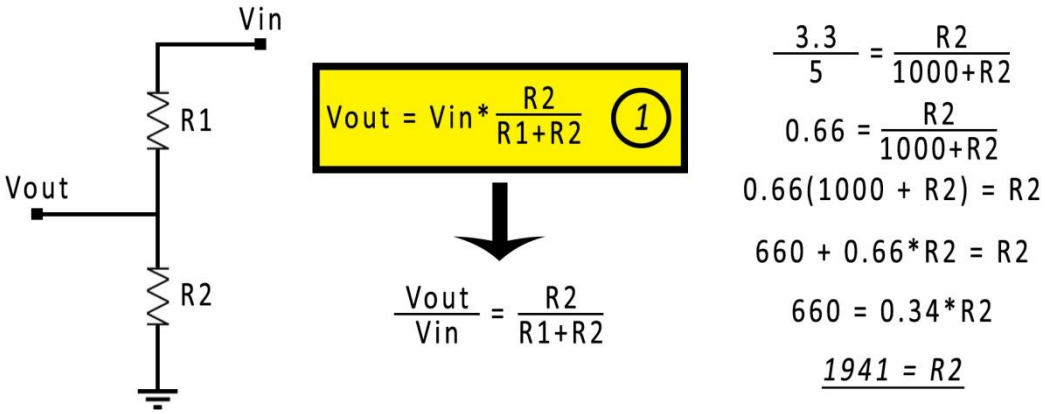
17150 * Χρόνο = Απόσταση

Με την βοήθεια του πάνω τύπου είναι εφικτό να υπολογίσουμε την απόσταση σε εκατοστά (cm).

Τώρα θα εξηγηθεί ο τρόπος διασύνδεσης του αισθητήρα υπερήχων με το Raspberry καθώς και ο προγραμματισμός του.

Για να συνδέσουμε τον αισθητήρα υπερήχων ο όποιος απαιτεί τάση 5V για να λειτουργήσει κάτι που δεν θα είναι πρόβλημα διότι το Raspberry παρέχει τροφοδοσία 5V. Το πρόβλημα μας είναι ότι οι θύρες του Raspberry αντέχουν τάσεις μόνο μέχρι 3.3V. Έτσι θα χρειαστεί να ρίξουμε την τάση επιστροφής της θύρας Echo του αισθητήρα στα 3.3V, για να το κατορθώσουμε αυτό θα χρειαστεί να κάνουμε χρήση ενός διαιρετή τάσης.

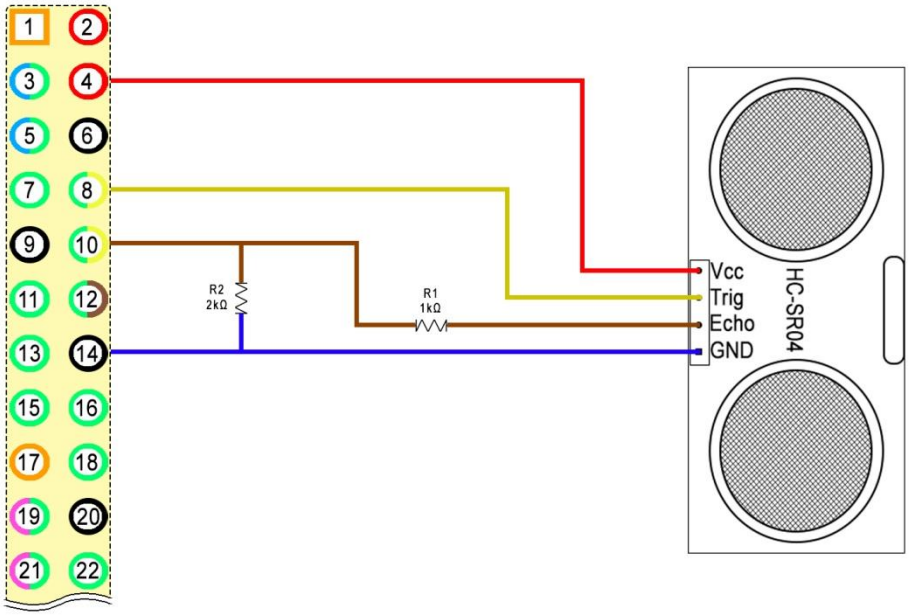
Ένας διαιρέτης τάσης αποτελείται από δυο αντιστάσεις R1,R2. Οπού ο σκοπός των αντιστάσεων είναι να ρίξουν την τάση που μπαίνει στον διαιρετή από την Vin ώστε να καταλήξει στην έξοδο του διαιρετή Vout.



Σχήμα 5.8.1 (υπολογισμός των αντιστάσεων ενός διαιρετή τάσης)

Για να βρούμε τι αντιστάσεις θα χρειαστούμε ώστε να λειτουργήσει ο διαιρέτης πήραμε αυθαίρετα την τιμή 1000Ω για την αντίσταση R1 και αντικαθιστώντας στον τύπο 1 του σχήματος 5.8.1 καταλήγουμε στο ότι η αντίσταση που θα χρειαστούμε θα πρέπει να είναι 2000Ω.

Γνωρίζοντας πλέον τι αντιστάσεις θα χρειαστούν μπορούμε να συνδέσουμε τον αισθητήρα υπερήχων στο Raspberry όπως στο παρακάτω σχήμα:



Σχήμα 5.8.2 (σύνδεση του αισθητήρα υπερήχων)

Θα χρειαστούμε:

Τον αισθητήρα υπερήχων HC-SR04.

1 x 1ΚΩ αντίσταση.

1 x 2ΚΩ αντίσταση.

1 x Ράστερ και καλώδια.

Στην συνέχεια θα παρουσιαστεί και θα αναλυθεί ο κώδικας που χρησιμοποιήθηκε για την λειτουργία του αισθητήρα.

```
#!/usr/bin/env python
import time
import RPi.GPIO as GPIO

TRIG =8
ECHO =10
GPIO.setmode(GPIO.BOARD)
GPIO.setup(TRIG,GPIO.OUT)
GPIO.setup(ECHO,GPIO.IN)

print("Εκτελείται υπολογισμός απόστασης")
GPIO.output(TRIG,False)
print("Αναμονή αρχικοποίησης αισθητήρα")
time.sleep(1)

GPIO.output(TRIG,True)
time.sleep(0.00001)
GPIO.output(TRIG,False)

while GPIO.input(ECHO)==0:
    pulse_start = time.time()

while GPIO.input(ECHO)==1:
    pulse_end = time.time()

pulse_duration = pulse_end - pulse_start
distance = pulse_duration *17150
distance = round(distance,2)
print("Απόσταση:",distance,"cm")
```

```
30:
31:     GPIO.cleanup()
```

Έξοδος: Θα εμφανιστούν τα μηνύματα “Εκτελείται υπολογισμός απόστασης ” και “Αναμονή αρχικοποίησης αισθητήρα”, ύστερα θα συμβεί παύση για 1 δευτερόλεπτο και στην συνέχεια θα γίνει υπολογισμός της απόστασης και εμφάνιση του τελικού αποτελέσματος στην οθόνη.

Ανάλυση:

1: Στην γραμμή 1 κάνουμε τον κώδικα εκτελέσιμο μέσω του Bash.

2,3: Γίνεται εισαγωγή των απαραίτητων βιβλιοθηκών.

5: Ορίζουμε μια μεταβλητή με όνομα TRIG να έχει την τιμή 8

6: Ορίζουμε μια μεταβλητή με όνομα ECHO να έχει την τιμή 10

7: Ορίζουμε τον φυσικό τρόπο αρίθμησης των θυρών GPIO.

8: Ορίζουμε την θύρα 8 (GPIO 14 σε BCM) ως έξοδο.

9: Ορίζουμε την θύρα 10 (GPIO 15σε BCM) ως είσοδο.

11: Τυπώνει μήνυμα στην οθόνη.

12: Θέτουμε την τιμή της θύρας 8 σε LOW.

13: Τυπώνει μήνυμα στην οθόνη.

14: Χρονική καθυστέρηση 1 δευτερόλεπτο.

16-18: Θέτουμε την τιμή της θύρας 8 σε High για 10μs ώστε να στείλει ο αισθητήρας υπέρηχους.

20-21: Αναμονή μέχρι να επιστρέψει ο πρώτος υπέρηχος και αποθήκευση της χρονικής στιγμής.

23-24: Αναμονή μέχρι να επιστρέψει ο τελευταίος υπέρηχος και αποθήκευση της χρονικής στιγμής.

26-28: Υπολογισμός τις απόστασης.

29: Τυπώνει την απόσταση στην οθόνη.

31: Απελευθερώνουμε τις θύρες που δεσμεύσαμε και γίνεται επανεκκίνηση του GPIO.

Κεφάλαιο 6

Χρήση του Raspberry μέσω VNC

Σε αυτό το κεφάλαιο θα εξηγηθεί ο τρόπος με τον οποίο είναι εφικτή η χρήση του Raspberry μέσω ενός υπολογιστή. Αυτός ο τρόπος είναι ιδιαίτερα χρήσιμος μιας και έτσι δεν χρειάζεται να συνδεθούνε οθόνη, πληκτρολόγιο και ποντίκι στο raspberry διευκολύνοντας τη σύνδεση του στο breadboard/raster.

6.1 Τι είναι VNC (Virtual Network Computing);

Το VNC είναι μια εφαρμογή μέσω της οποίας είναι εφικτός ο απομακρυσμένος έλεγχος ενός υπολογιστή, μεταδίδοντας πληροφορίες του γραφικού περιβάλλοντος, του ποντικιού αλλά και του πληκτρολογίου.

6.2 Προετοιμασία του Raspberry

Η εφαρμογή που θα χρησιμοποιηθεί ώστε να είναι εφικτή η απομακρυσμένη σύνδεση είναι το VNC Viewer της RealVNC και είναι προεγκατεστημένη στο σύστημα Raspbian. Παρακάτω θα εξηγηθούνε τα βήματα που είναι απαραίτητα για την ορθή λειτουργία της εφαρμογής. Σημειώνεται ότι για την ρύθμιση θα πρέπει το Raspberry να συνδεθεί σε οθόνη και πληκτρολόγιο.

6.2.1 Ρυθμίσεις εκκίνησης της εφαρμογής VNC

Είναι απαραίτητο η εφαρμογή να εκκινεί μαζί με το λειτουργικό σε περίπτωση που δεν θα είναι διαθέσιμα περιφερειακά για την εκκίνηση και ρύθμιση της συσκευής.

Το πρώτο βήμα που πρέπει να κάνουμε είναι να ανοίξουμε το terminal και να πληκτρολογήσουμε:

```
sudo leafpad /boot/config.txt
```

Στο παράθυρο που μας ανοίγει ψάχνουμε την γραμμή που λέει “#hdmi_force_hotplug=1” και αφαιρούμε την δίεση (#) από αυτήν την γραμμή. Αυτό

το κάνουμε για να είναι εφικτή η εκκίνηση του raspberry χωρίς να έχει συνδεμένη οθόνη πάνω του. Ύστερα αποθηκεύουμε και κλείνουμε το έγγραφο.

Για ενεργοποίηση της αυτόματης εκκίνησης του VNC πηγαίνουμε από την έναρξη στο Preferences/Raspberry Pi Configuration και στο παράθυρο που μας ανοίγει επιλέγουμε την καρτέλα Interfaces ενεργοποιούμε το VNC και πατάμε OK. Κάνουμε επανεκκίνηση το σύστημα ώστε να αποθηκευτούνε οι ρυθμίσεις.

6.2.2 Ορισμός στατικής IP για ασύρματο η ενσύρματο δίκτυο

Για να ορίσουμε στατική IP θα πρέπει να ανοίξουμε το terminal και να πληκτρολογήσουμε την εντολή:

```
sudo leafpad /etc/dhcpcd.conf
```

Και στο παράθυρο που ανοίγει θα προσθέσουμε κώδικα στο τέλος του αρχείου, για τον προσαρμογέα δικτύου που μας ενδιαφέρει. Ο κώδικας θα πρέπει να είναι όπως στο σχήμα 6.2.1.1. Φροντίζουμε να βάλουμε την διεύθυνση του δικού μας router, και να ορίσουμε μια στατική διεύθυνση για το raspberry.

```
interface [συσκευή]
inform [στατική διεύθυνση του raspberry]
static routers = [IP του router]  #Προαιρετικό


Παράδειγμα:
interface wlan0
inform 192.168.1.31
static routers=192.168.1.1  #Προαιρετικό
```

Σχήμα 6.2.2.1

Όπου "**συσκευή"** βάζουμε **eth0** για την Ethernet σύνδεση και **wlan0** για την ασύρματη σύνδεση. Είναι εφικτό να δούμε αναλυτικό πίνακα τον συσκευών δικτύου πληκτρολογώντας της εντολές "**sudo ifconfig**" στο τερματικό. Μας δίνεται η δυνατότητα να ορίσουμε ξεχωριστή στατική IP για το ενσύρματο(eth0) και το ασύρματο(wlan0) δίκτυο, καθώς και για άλλες συσκευές δικτύου συνδεμένες στο raspberry.Μόλις τελειώσουμε αποθηκεύουμε το έγγραφο και το κλείνουμε. Κάνουμε επανεκκίνηση του συστήματος ώστε να αποθηκευτούν οι νέες ρυθμίσεις.

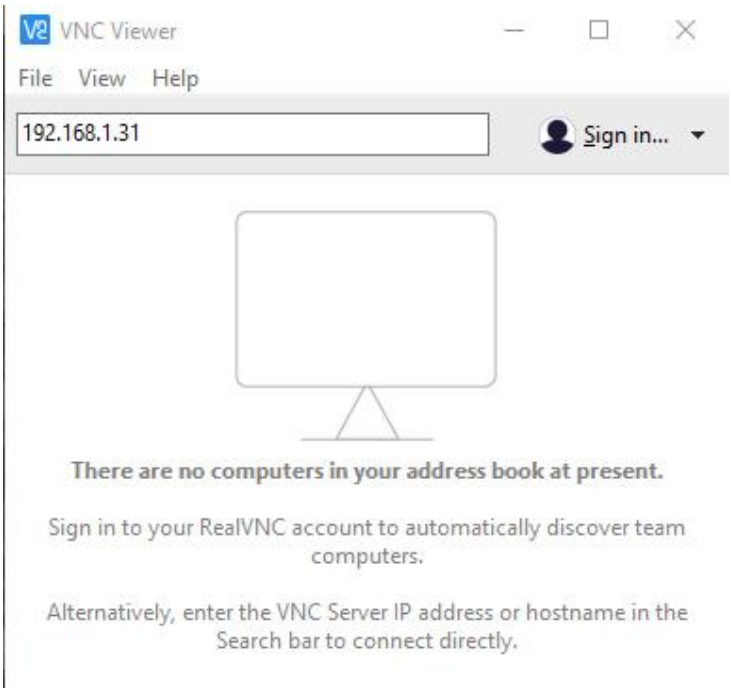
Μετά την επανεκκίνηση πατάμε στο εικονίδιο του VNC viewer ( ) που βρίσκεται πάνω δεξιά στην περιοχή ειδοποιήσεων, και ελέγχουμε αν όντως η διεύθυνση IP είναι αυτή που ορίσαμε προηγούμενος.

Για την περίπτωση που θέλουμε να συνδέσουμε το Raspberry κατευθείαν στον προσαρμογέα δικτύου Lan ενός Η/Υ, θα πρέπει να έχουμε ορίσει στατική διεύθυνση IP για τον Η/Υ.

6.3 Προετοιμασία του υπολογιστή client (πελάτη)

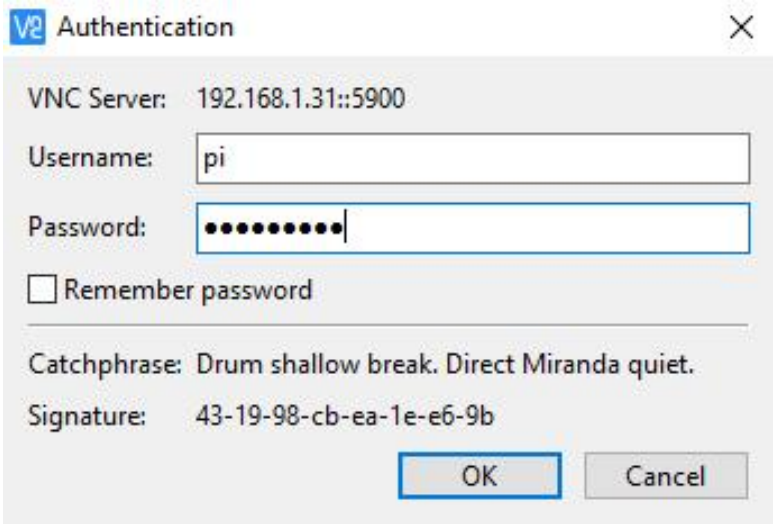
Κατεβάζουμε και τρέχουμε την εφαρμογή VNC Viewer από την ιστοσελίδα: https://www.realvnc.com/download/viewer/ .

Στο παράθυρο που θα μας ανοίξει πληκτρολογούμε στο πεδίο την στατική διεύθυνση IP που θέσαμε στο raspberry και πατάμε OK.



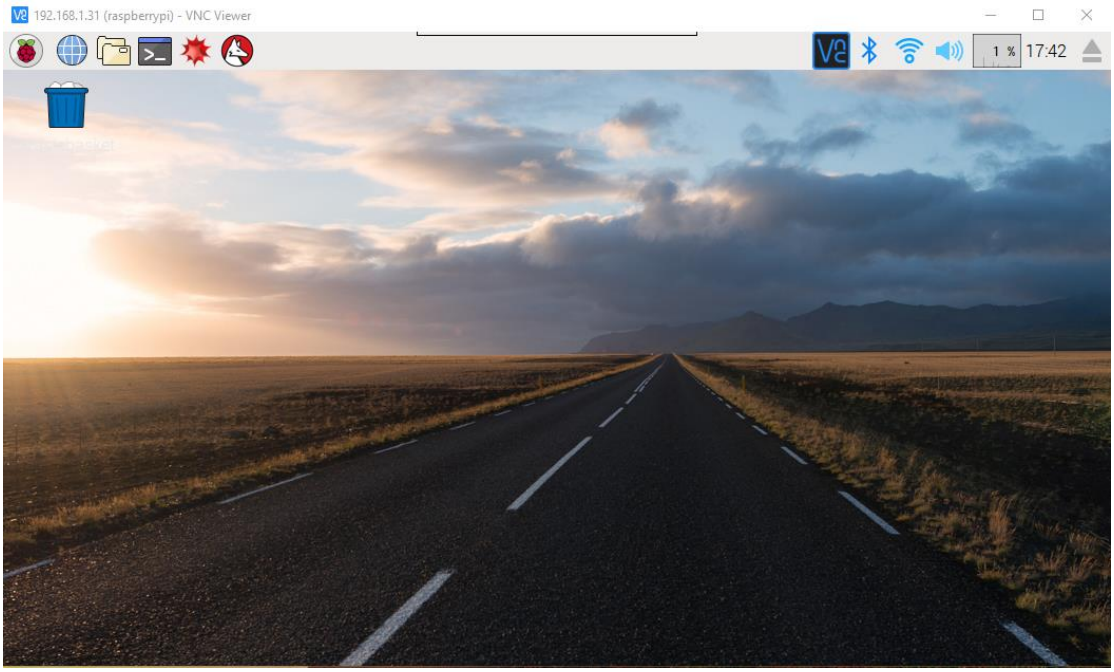
Εικόνα 6.3.1 (το παράθυρο του VNC Viewer)

Στο παράθυρο που ανοίγει στο πεδίο **username** γράφουμε “pi” και στο πεδίο **password** γράφουμε “raspberry” και πατάμε OK.



Εικόνα 6.3.2 (εισαγωγή τον στοιχείων χρήστη του Raspbian)

Μόλις πατήσουμε OK θα ανοίξει ένα παράθυρο από το οποίο θα έχουμε πρόσβαση στην επιφάνεια εργασίας του raspberry.



Εικόνα 6.3.3 (επιφάνεια εργασίας του Raspbian όπως φαίνεται μέσο VNC viewer)

Κεφάλαιο 7

Σύνδεση κάμερας μέσω USB

Σε αυτό το κεφάλαιο θα γίνει σύνδεση και χρήση μιας απλής USB webcam.

7.1 Λήψη εφαρμογής χειρισμού webcam

Η εφαρμογή που θα χρησιμοποιηθεί για την λήψη στιγμιότυπων από την sub κάμερα είναι η webcam. Για να κατεβάσουμε την εφαρμογή πρέπει να ανοίξουμε το terminal και να πληκτρολογήσουμε την εξής εντολή:

```
sudo apt-get install fswebcam
```

Μόλις κατέβει η εφαρμογή είμαστε πλέον σε θέση να λαμβάνουμε στιγμιότυπα.

Η εφαρμογή fswebcam τρέχει μέσω του terminal, για να τραβήξουμε ένα στιγμιότυπο το σώμα της εντολής πρέπει να έχει την εξής μορφή:

fswebcam <-r **ανάλυση**> <**όνομα_ εικόνας**>.< **τύπος**>

Παραδείγματα:

fswebcam image1.jpg

fswebcam -r 640x480 image2.png

Παράδειγμα 7.1.1

Μας είναι εφικτό να ορίσουμε και την ανάλυση που θα έχει η φωτογραφία που θα τραβήξουμε. Η μεγίστη ανάλυση εξαρτάται από την webcam. Υπάρχουν δυο υποστηριζόμενοι τύποι εικόνας .jpg και .png

Παρακάτω βλέπουμε ένα στιγμιότυπο που τραβήχτηκε με το fswebcam:

Εικόνα 7.1.1 (στιγμιότυπο από την USB webcam)

Κεφάλαιο 8

Συμπεράσματα, κριτική, και προτάσεις

Γενικά συμπεράσματα της μελέτης

Το raspberry pi έχει ενεργή κοινότητα, χάρη σε αυτό είναι εφικτό να βρει κανείς έτοιμο κώδικα για σχεδόν τα πάντα.

Επίσης οι δημιουργοί του προθέτουνε στο βασικό του λειτουργικό (Raspbian), όλο και περισσότερες ικανότητες, κάτι που διευκολύνει και την χρήση της συσκευής.

Κατά την σύνδεση αισθητήρων/συσκευών στης θύρες GPIO, θα πρέπει να δοθεί προσοχή στα ενεργά πρωτόκολλα (από την έναρξη στο Preferences>Raspberry Pi Configuration και από εκεί στην καρτέλα Interfaces), διότι υπάρχει κίνδυνος να μην λειτουργήσουν σωστά κάποια πειράματα. Αν για παράδειγμα είναι ενεργό το Serial οι θύρες 8 και 10 θα δεσμεύουν από το σύστημα για την λειτουργία UART.

Το Raspberry είχε σταθερή συμπεριφορά σε όλα τα πειράματα και επέστρεφε πάντα τα αναμενόμενα αποτελέσματα.

Κριτική

Δυστυχώς δεν είχα τα κατάλληλα περιφερικά ώστε να πειραματιστώ και να εξηγήσω την σύνδεση συσκευών όπως: αισθητήρες πρωτοκόλλου SPI και UART, συσκευή ψηφιακού ήχου πρωτοκόλλου I2S (PCM), οθόνη με τρόπο διασύνδεσης DSI, κάμερα με τρόπο διασύνδεσης CSI. Επίσης δεν βρήκα καλώδιο για να ελέγξω τον τρόπο σύνδεσης σε οθόνη με την χρήση του 4pole 3.5mm Jack.
Δυστυχώς το Raspberry δεν έχει αναλογικές θύρες.

Προτάσεις για μελλοντική έρευνα

Είναι εφικτό πλέον από το Raspberry Pi 3 να τρέξει το λογισμικό OpenCV. Με την χρήση του OpenCV είναι εφικτές οι εργασίες όπως η αναγνώριση: προσώπων, αντικείμενων/σχημάτων, χρωμάτων. Η διαδικασία της αναγνώρισης είναι εφικτό να γίνει σε φωτογραφίες σε video, άλλα και πάνω στην λήψη video.

Είναι εφικτό να προγραμματίσουμε το raspberry και σε άλλες γλώσσες προγραμματισμού όπως η C και η JAVA, που προσφέρουν γρηγορότερη απόκριση του raspberry σε σχέση με την Python.

Βιβλιογραφία

Shotts, Jr. , W. E., n.d. *Writing Your First Script And Getting It To Work.* [Ηλεκτρονικό]
Available at: http://linuxcommand.org/lc3_wss0010.php
[Πρόσβαση 24 06 2017].

Adafruit, 2016. *Adafruit_Python_CharLCD.* [Ηλεκτρονικό]
Available at: https://github.com/adafruit/Adafruit_Python_CharLCD
[Πρόσβαση 06 06 2017].

Adafruit, 2017. *Adafruit_Python_GPIO.* [Ηλεκτρονικό]
Available at: https://github.com/adafruit/Adafruit_Python_GPIO
[Πρόσβαση 06 06 2017].

Anon., 2009. *Οδηγός χρήσης του Debian/Debian για πρώτη φορά.* [Ηλεκτρονικό]
Available at: https://el.wikibooks.org/wiki/
[Πρόσβαση 02 06 2017].

Anon., 2010. *Linux: Show / Display Available Network Interfaces.* [Ηλεκτρονικό]
Available at: https://www.cyberciti.biz/faq/linux-list-network-interfaces-names-command/
[Πρόσβαση 05 06 2017].

Anon., 2014. *HC-SR04 Ultrasonic Range Sensor on the Raspberry Pi.* [Ηλεκτρονικό]
Available at: http://www.modmypi.com/blog/hc-sr04-ultrasonic-range-sensor-on-the-raspberry-pi
[Πρόσβαση 25 05 2017].

Anon., 2015. *How to setup a static IP address on your Raspberry Pi.* [Ηλεκτρονικό]
Available at: https://thepihut.com/blogs/raspberry-pi-tutorials/16683276-how-to-setup-a-static-ip-address-on-your-raspberry-pi
[Πρόσβαση 05 06 2017].

Anon., 2015. *Raspberry Pi LESSON 28: Controlling a Servo on Raspberry Pi with Python.* [Ηλεκτρονικό]
Available at: http://www.toptechboy.com/raspberry-pi/raspberry-pi-lesson-28-controlling-a-servo-on-raspberry-pi-with-python/
[Πρόσβαση 25 05 2017].

Anon., 2016. *Linux για αρχάριους/Βασικές γνώσεις τερματικού.* [Ηλεκτρονικό]
Available at:
https://el.wikibooks.org/wiki/Linux_για_αρχάριους/Βασικές_γνώσεις_τερματικού
[Πρόσβαση 03 06 2017].

Anon., 2016. *RPi Hardware.* [Ηλεκτρονικό]
Available at: http://elinux.org/RPi_Hardware
[Πρόσβαση 25 05 2017].

Anon., 2016. *Οδηγίες χρήσης του Github.* [Ηλεκτρονικό]
Available at: https://ellak.gr/wiki/index.php?title=Οδηγίες_χρήσης_του_Github
[Πρόσβαση 03 06 2017].

Browning, J., 2012. *So you got a Raspberry Pi now what?.* [Ηλεκτρονικό]
Available at: https://www.engadget.com/2012/09/04/raspberry-pi-getting-started-guide-how-to/
[Πρόσβαση 25 05 2017].

Croston, B., n.d. *RPi.GPIO Python Module Examples.* [Ηλεκτρονικό]
Available at: https://sourceforge.net/p/raspberry-gpio-python/wiki/Examples/
[Πρόσβαση 25 05 2017].

Dee, J., n.d. *Pulse-width Modulation.* [Ηλεκτρονικό]
Available at: https://learn.sparkfun.com/tutorials/pulse-width-modulation
[Πρόσβαση 06 06 2017].

Fried, L., 2015. *Adafruit HDC1008 Temperature and Humidity Sensor Breakout.* [Ηλεκτρονικό]
Available at: https://learn.adafruit.com/adafruit-hdc1008-temperature-and-humidity-sensor-breakout
[Πρόσβαση 25 05 2017].

Guill, D., 2015. *Raspberry Pi Cluster.* [Ηλεκτρονικό]
Available at: http://likemagicappears.com/projects/raspberry-pi-cluster/
[Πρόσβαση 17 06 2017].

Hawkins, M., 2015. *How To Format Pi SD Cards Using SD Formatter.* [Ηλεκτρονικό]
Available at: http://www.raspberrypi-spy.co.uk/2015/03/how-to-format-pi-sd-cards-using-sd-formatter/
[Πρόσβαση 25 05 2017].

Krassenstein, E., 2014. *World's First Raspberry Pi Controlled 3D Printer is Created by Engineering Student.* [Ηλεκτρονικό]
Available at: http://3dprint.com/16060/raspberry-pi-3d-printer/
[Πρόσβαση 17 06 2017].

Leventeas, D., 2009. *Οδηγός Εκμάθησης Python Βήμα Βήμα.* [Ηλεκτρονικό]
Available at: http://python.org.gr/index.php/files/file/9-python
[Πρόσβαση 25 05 2017].

Linux Format, 2016. *How to build your own Raspberry Pi NAS.* [Ηλεκτρονικό]
Available at: http://www.techradar.com/how-to/computing/how-to-build-your-own-raspberry-pi-nas-1315968
[Πρόσβαση 17 06 2017].

Perry, C., 2016. *AptGet/Howto.* [Ηλεκτρονικό]
Available at: https://help.ubuntu.com/community/AptGet/Howto
[Πρόσβαση 07 06 2017].

Programiz, n.d. *Learn Python Programming.* [Ηλεκτρονικό]
Available at: https://www.programiz.com/python-programming
[Πρόσβαση 12 06 2017].

Python Software Foundation, 2017. *Python 3.6.1 documentation.* [Ηλεκτρονικό]
Available at: https://docs.python.org/3/
[Πρόσβαση 25 05 2017].

Raspberry Pi Foundation, 2012. *How to Backup your SDcard.* [Ηλεκτρονικό]
Available at: https://www.raspberrypi.org/forums/viewtopic.php?p=239331
[Πρόσβαση 25 05 2017].

Raspberry Pi Foundation, n.d. *Installing operating system images.* [Ηλεκτρονικό]
Available at: https://www.raspberrypi.org/documentation/installation/installing-images/README.md
[Πρόσβαση 25 05 2017].

Raspberry Pi Foundation, n.d. *Raspberry Pi 3 is out now! Specs, benchmarks & more.* [Ηλεκτρονικό]
Available at: https://www.raspberrypi.org/magpi/raspberry-pi-3-specs-benchmarks/
[Πρόσβαση 02 06 2017].

Raspberry Pi Foundation, n.d. *Raspbian.* [Ηλεκτρονικό]
Available at: https://www.raspberrypi.org/documentation/raspbian/
[Πρόσβαση 25 05 2017].

Raspberry Pi Foundation, n.d. *SETTING UP AN APACHE WEB SERVER ON A RASPBERRY PI.* [Ηλεκτρονικό]
Available at: https://www.raspberrypi.org/documentation/remote-access/web-server/apache.md
[Πρόσβαση 17 06 2017].

Raspberry Pi Foundation, n.d. *USING A STANDARD USB WEBCAM.* [Ηλεκτρονικό]
Available at: https://www.raspberrypi.org/documentation/usage/webcams/
[Πρόσβαση 05 06 2017].

Raspberry Pi Foundation, n.d. *Welcome to Raspbian.* [Ηλεκτρονικό]
Available at: https://www.raspbian.org/
[Πρόσβαση 02 06 2017].

Raspberry Pi Foundation, n.d. *WHAT IS A RASPBERRY PI?.* [Ηλεκτρονικό]
Available at: https://www.raspberrypi.org/help/faqs/#introWhatIs
[Πρόσβαση 02 06 2017].

Raspberry Pi Foundation, n.d. *WHAT IS AN SOC?.* [Ηλεκτρονικό]
Available at: https://www.raspberrypi.org/help/faqs/#generalSoCdefined
[Πρόσβαση 25 05 2017].

Rosenbrock, A., 2016. *Install guide: Raspberry Pi 3 + Raspbian Jessie + OpenCV 3.* [Ηλεκτρονικό]
Available at: http://www.pyimagesearch.com/2016/04/18/install-guide-raspberry-pi-3-raspbian-jessie-opencv-3/
[Πρόσβαση 17 06 2017].

Rouse, M., 2015. *system-on-a-chip (SoC).* [Ηλεκτρονικό]
Available at: http://internetofthingsagenda.techtarget.com/definition/system-on-a-chip-SoC
[Πρόσβαση 25 05 2017].

Scanner, L., 2016. *HDC100X_Python.* [Ηλεκτρονικό]
Available at: https://github.com/scanner/HDC100X_Python
[Πρόσβαση 06 06 2017].

Shabaz, 2015. *Raspberry Pi GPIO Explained.* [Ηλεκτρονικό]
Available at: https://www.element14.com/community/docs/DOC-78315/l/raspberry-pi-gpio-explained
[Πρόσβαση 25 05 2017].

Shawn, 2010. *Disk - Clean and Clean All with Diskpart Command.* [Ηλεκτρονικό]
Available at: https://www.sevenforums.com/tutorials/52129-disk-clean-clean-all-diskpart-command.html
[Πρόσβαση 25 05 2017].

Sklar, M., 2015. *Drive a 16x2 LCD with the Raspberry Pi.* [Ηλεκτρονικό]
Available at: https://learn.adafruit.com/drive-a-16x2-lcd-directly-with-a-raspberry-pi
[Πρόσβαση 25 05 2017].

Upton, E. & Halfacree, G., 2014. *Raspberry Pi User Guide.* Second Edition επιμ. s.l.:WILEY.

Vilches, J., 2017. *Interview with Raspberry's Founder Eben Upton.* [Ηλεκτρονικό]
Available at: http://www.techspot.com/article/531-eben-upton-interview/
[Πρόσβαση 16 06 2017].

Wikipedia, 2017. *Python (programming language).* [Ηλεκτρονικό]
Available at: https://en.wikipedia.org/wiki/Python_(programming_language)
[Πρόσβαση 25 05 2017].

Wikipedia, 2017. *Raspberry Pi.* [Ηλεκτρονικό]
Available at: http://em.m.wikipedia.org/wiki/Raspberry_Pi
[Πρόσβαση 16 06 2017].

Wikipedia, 2017. *Virtual Network Computing.* [Ηλεκτρονικό]
Available at: https://en.wikipedia.org/wiki/Virtual_Network_Computing
[Πρόσβαση 05 06 2017].

Wikipedia, n.d. *Single-board computer.* [Ηλεκτρονικό]
Available at: https://en.wikipedia.org/wiki/Single-board_computer
[Πρόσβαση 02 06 2017].

Wikipedia, n.d. *System on a chip.* [Ηλεκτρονικό]
Available at: https://en.wikipedia.org/wiki/System_on_a_chip
[Πρόσβαση 02 06 2017].

Κυρίτσης, Α., 2014. *Απομακρυσμένος Έλεγχος Υπολογιστή - Οι Καλύτερες μέθοδοι.* [Ηλεκτρονικό]
Available at: https://www.pcsteps.gr/1350-απομακρυσμένος-έλεγχος-υπολογιστή/
[Πρόσβαση 05 06 2017].

Κυρίτσης, Α., 2015. *Εντολές στο Τερματικό Linux (Linux Terminal) για Αρχάριους.* [Ηλεκτρονικό]
Available at: www.pcsteps.gr/746-εντολές-τερματικό-linux-terminal/
[Πρόσβαση 03 06 2017].

Εικόνες:

Εικόνα 1.5.1 :
Available at: http://likemagicappears.com/site-content/uploads/2014/02/RPCI-00-00000007.jpg
[Πρόσβαση 19 06 2017].

Εικόνα 1.5.2 :
Available at: https://3dprint.com/wp-content/uploads/2014/09/rpi4.jpg
[Πρόσβαση 19 06 2017].

Εικόνα 1.5.3 :
Available at: http://www.pyimagesearch.com/wp-content/uploads/2015/02/raspberry_pi_in_post.jpg
[Πρόσβαση 19 06 2017].