Erasmus+ KA-202

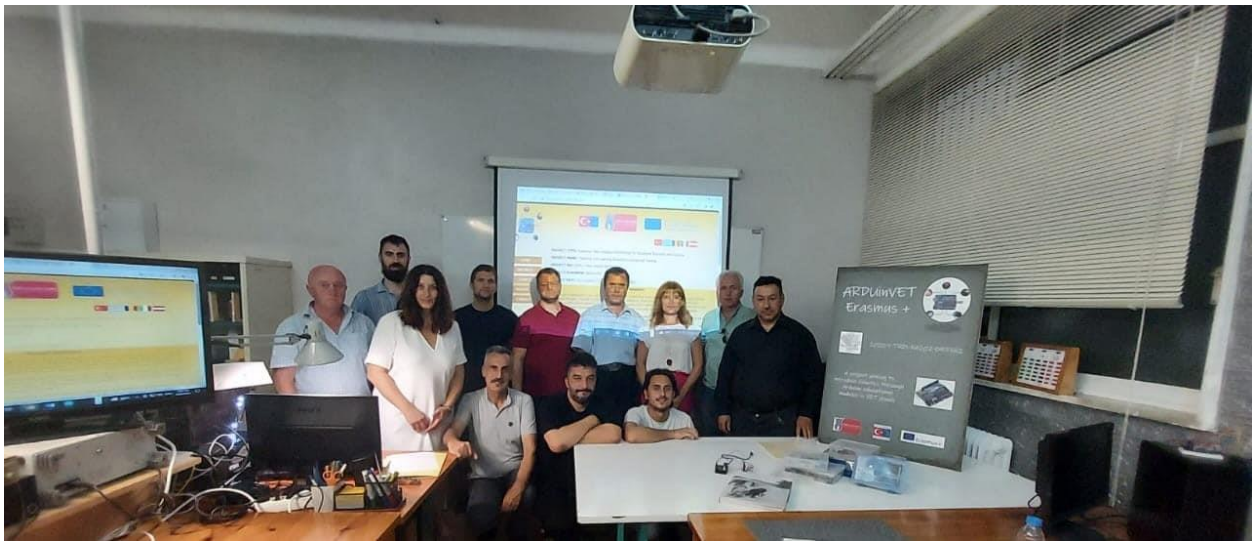
Πρόγραμμα Στρατηγικής Σύμπραξης στην Επαγγελματική Εκπαίδευση και Κατάρτιση

Τίτλος Προγράμματος: "Teaching and Learning Arduinos in Vocational Training"

Ακρονύμιο Προγράμματος: " ARDUinVET "

Κωδικός Προγράμματος: "2020-1-TR01-KA202-093762"

***ΕΓΧΕΙΡΙΔΙΟ ARDUinVET ***



« Το πρόγραμμα χρηματοδοτείται από το Πρόγραμμα Erasmus+ της Ευρωπαϊκής Ένωσης. Η υποστήριξη της Ευρωπαϊκής Επιτροπής και του Οργανωτικού Φορέα της Τουρκίας για την παραγωγή αυτού του υλικού δεν αποτελεί έγκριση του περιεχομένου το οποίο αντικατοπτρίζει τις απόψεις αποκλειστικά των συγγραφέων και η Επιτροπή δεν μπορεί να θεωρηθεί υπεύθυνη για οποιαδήποτε χρήση των πληροφοριών που περιέχονται στο παρόν.»

"Bu proje Erasmus+ Programı kapsamında Avrupa Komisyonu tarafından desteklenmektedir. Ancak burada yer alan görüşlerden Avrupa Komisyonu ve Türkiye Ulusal Ajansı sorumlu tutulamaz."

“This project is Funded by the Erasmus+ Program of the European Union. However, European Commission and Turkish National Agency cannot be held responsi-ble for any use which may be made of the information contained therein”

Co-funded by the Erasmus+ Programme of the European Union

ΣΥΝΤΟΝΙΣΤΗΣ:

Gölbaşı Mesleki ve Teknik Anadolu Lisesi (Ankara / ΤΟΥΡΚΙΑ)

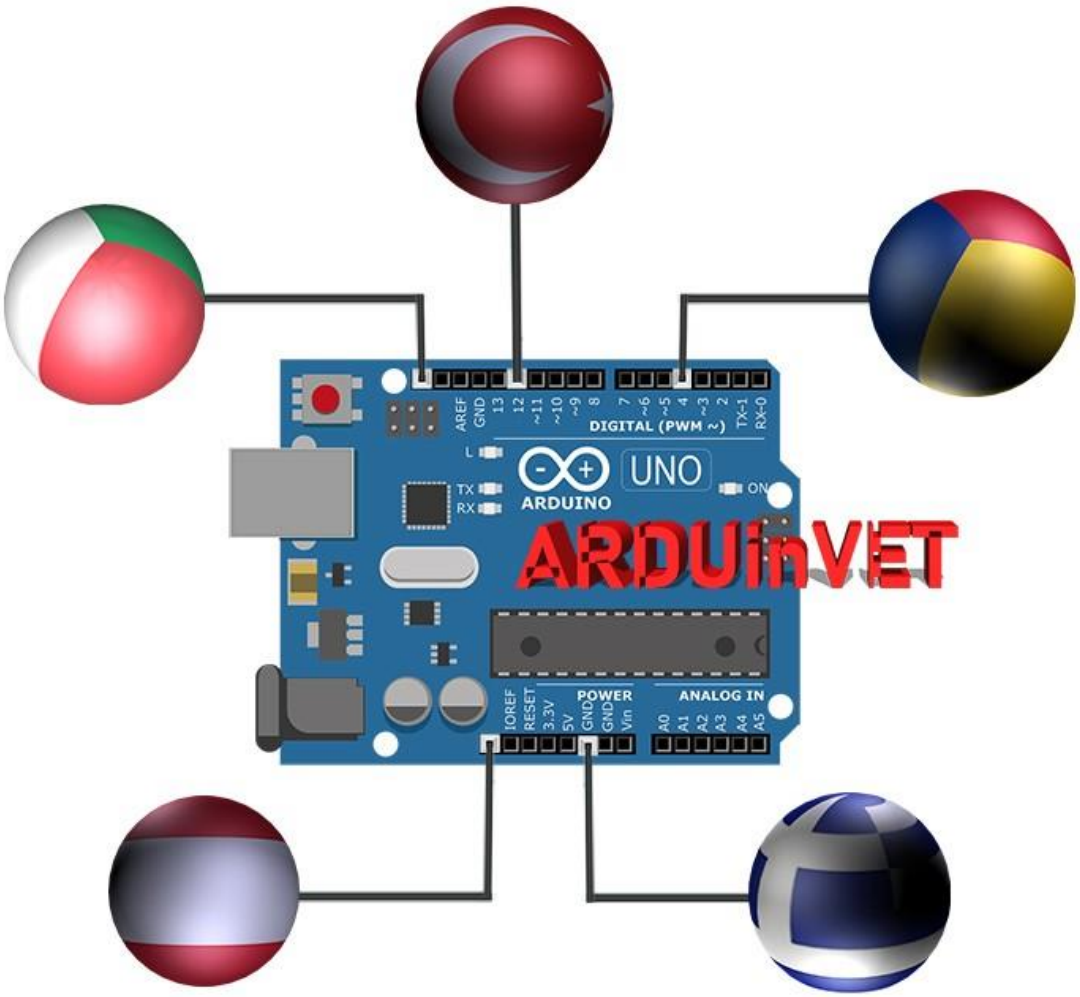
ΣΥΜΜΕΤΕΧΟΝΤΕΣ:

2 ΕΚ Πειραιά (Πειραιάς / Ελλάς)

Liceul Tehnologic Grigore Moisil Braila (Braila / Ρουμανία)

Istituto di Istruzione Superiore Einstein De Lorenzo Potenza (Potenza / Ιταλία)

HTBLA Wolfsberg / (College for Engineering Wolfsberg) (Wolfsberg / Αυστρία)



Περίληψη έργου

"Διδασκαλία και εκμάθηση Arduino στην Επαγγελματική Εκπαίδευση"

Η τεχνολογία αναπτύσσεται καθημερινά με ταχύτατους ρυθμούς. Είναι γεγονός ότι οι τεχνολογικές εξελίξεις εξαναγκάζουν τους ανθρώπους να αναπτύσσονται συνεχώς στην επαγγελματική τους ζωή. Περισσότερο από οποιονδήποτε άλλο τομέα, η ηλεκτρονική και η τεχνολογία πληροφοριών και επικοινωνιών (ΤΠΕ) αναπτύσσονται ταχύτερα. Αυτές οι εξελίξεις οδήγησαν στην παραγωγή πιο πολύπλοκων συστημάτων ελέγχου. Πλέον, τα Arduino χρησιμοποιούνται για τον έλεγχο αυτών των πολύπλοκων συστημάτων.

Δεδομένου ότι είμαστε εκπαιδευτικά ιδρύματα που διδάσκουν τεχνολογία, είμαστε υποχρεωμένοι να παρακολουθούμε τις παγκόσμιες τεχνολογικές εξελίξεις. Το πρόγραμμα «Teaching and Learning Arduinos in Vocational Training» στοχεύει στην προσαρμογή των εφαρμογών Arduino στην επαγγελματική εκπαίδευση και κατάρτιση και στην ανάπτυξη ενός πιο αποτελεσματικού εκπαιδευτικού πακέτου και ενός οδηγού για τα εργαστήρια και τα εργαστηριακά μαθήματα επαγγελματικής και τεχνικής εκπαίδευσης.

Ο κύριος στόχος αυτού του έργου είναι η ανάπτυξη ενός βιβλίου ορθών πρακτικών και ενός εκπαιδευτικού κιτ Arduino, η εισαγωγή μοντέλων εκπαίδευσης arduino στους υπόλοιπους συμμετέχοντες κατά τη διάρκεια των επισκέψεών τους σε κάθε χώρα υποδοχής, η σύγκριση διαφορετικών εκπαιδευτικών συστημάτων και μεθόδων κατάρτισης με τα υπόλοιπα συμμετέχοντα άλλα σχολεία και ο διαμοιρασμός βέλτιστων πρακτικών. Συμμετέχοντες στα έργα είναι: καθηγητές Ηλεκτρικής, Ηλεκτρονικής, ΤΠΕ, Αυτοματισμού ΕΕΚ. Οι συμμετέχοντες διδάσκουν arduino σε σχολεία ΕΕΚ. Υπάρχει ένας συντονιστής και συνολικά 4 εταίροι στο έργο. Οι εταίροι είναι επαγγελματικά σχολεία από την Τουρκία, την Ιταλία, τη Ρουμανία, την Αυστρία, την Ελλάδα. Ο συνολικός αριθμός κινητικότητων του έργου είναι 40. Συνολικά θα πραγματοποιηθεί καθ 'όλη τη διάρκεια του έργου 5 διεθνικές συναντήσεις και κάθε εταίρος θα συμμετέχει σε κάθε συνάντηση με 2 συμμετέχοντες. Σε κάθε χώρα θα πραγματοποιηθεί μια συνάντηση.

Με τις δραστηριότητες του έργου, θα διασφαλιστεί ότι θα διαμοιραστούν βέλτιστες πρακτικές για τη διδασκαλία Arduino και οι εταίροι θα τις προσαρμόσουν στο δικό τους εκπαιδευτικό τους περιβάλλον. Οι εταίροι του έργου θα παράγουν πειραματικά κιτ Arduino και έναν οδηγό βέλτιστων πρακτικών. Οι δραστηριότητες του έργου μας στοχεύουν στην παροχή ενός επιτυχημένου περιβάλλοντος μάθησης και διδασκαλίας για όλους τους εκπαιδευτικούς και τους μαθητές της ΕΕΚ. Ως προϋπόθεση της επιτυχούς μάθησης, οι εκπαιδευτικοί πρέπει να ενισχύσουν το ρόλο τους στη διευκόλυνση της μάθησης. Οι εκπαιδευτικοί χρειάζονται νέα πειραματικά κιτ και ενότητες για καινοτομία, ομαδική εργασία, ανατροφοδότηση και αξιολόγηση. Είναι υποχρέωση μας να δοθεί η ευκαιρία στους εκπαιδευτικούς για συνεχή επαγγελματική ανάπτυξη.

Τα κύρια αποτελέσματα του έργου είναι: ένα εκπαιδευτικό κιτ Arduino για διδασκαλία Arduino στην επαγγελματική εκπαίδευση και ένας οδηγός για αυτό το κιτ, μια ιστοσελίδα έργου και ένα

DVD έργου. Η μεθοδολογία που θα χρησιμοποιηθεί για την εκτέλεση του έργου είναι "Make-Develop-Share". Στο έργο μας, οι καλές πρακτικές αχικά θα δοκιμαστούν πρώτα, στη συνέχεια θα αναπτυχθούν και τελικά θα διαμοιραστούν. Όλα είναι ανοιχτά και διαφανή στο έργο μας. Κάθε εργασία και ευθύνη θα καταγράφονται και θα περιγράφονται στο έργο. Τα προϊόντα του έργου δεν είναι μόνο γραπτά ή ντοκιμαντέρ, αλλά και ένα πρακτικό σετ και κιτ εκπαίδευσης Arduino.

Μακροπρόθεσμα, στοχεύουμε να διαδώσουμε το έργο σε δασκάλους, μαθητές και σχολεία επαγγελματικής εκπαίδευσης, τοπικά εκπαιδευτικά ιδρύματα, ηλεκτρονική αγορά εργασίας και ΤΠΕ. Πιστεύουμε ότι τα αποτελέσματα και τα προϊόντα των έργων μας θα έχουν βραχυπρόθεσμες και μακροπρόθεσμες επιπτώσεις στην επαγγελματική εκπαίδευση και την αγορά εργασίας από τις δραστηριότητες διάδοσης. Ο συνολικός προϋπολογισμός του έργου για τους 5 εταίρους είναι 59.000 Ευρώ.

Περιεχόμενα

Εισαγωγή στα ARDUINOS 8

Arduino Input/Output και κιτ 24

Οθόνη LCD και εκπαιδευτικό κιτ 72

Εκπαιδευτικό κιτ Πληκτρολογίου 97

Εκπαιδευτικό κιτ Dot Matrix 117

Εκπαιδευτικό κιτ Κινητήρων Arduino 139

Αισθητήρες και εκπαιδευτικό κιτ 163

Ελεύθερα Projects Arduino 206

Παραρτήματα 238

Co-funded by the
Erasmus+ Programme
of the European Union

ΕΝΟΤΗΤΕΣ ΕΡΓΟΥ και KITS του ARDUinVET

Co-funded by the Erasmus+ Programme of the European Union

Erasmus+ KA-202

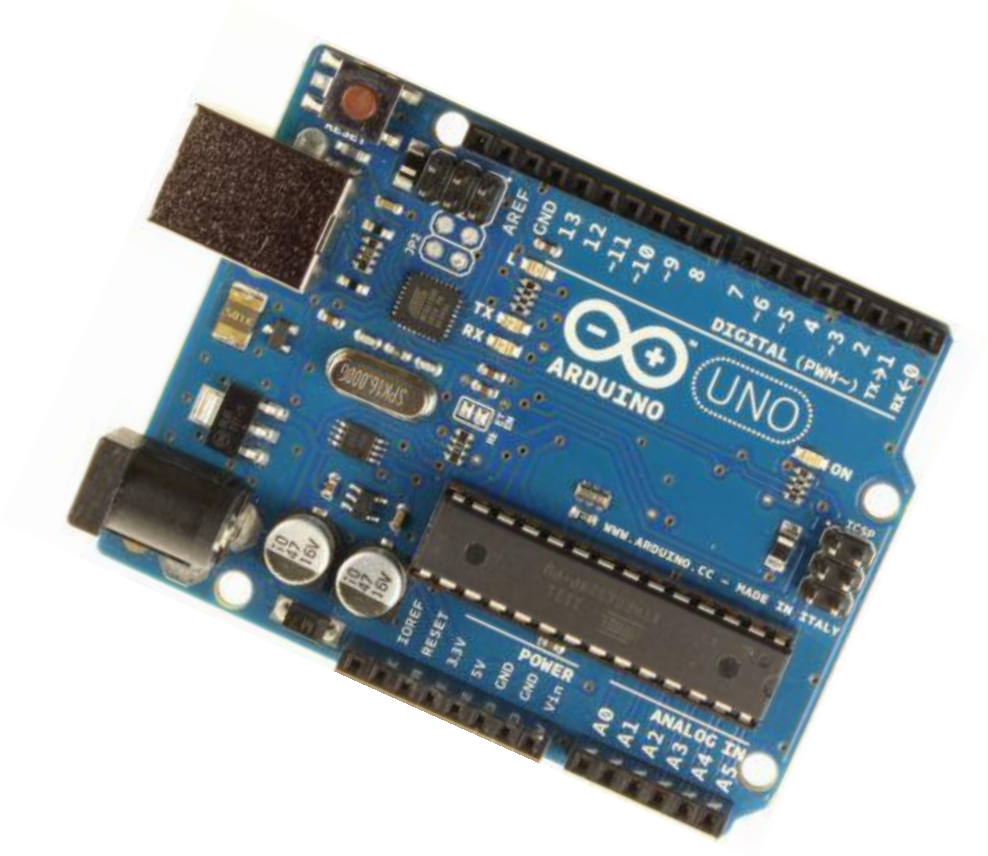
Πρόγραμμα Στρατηγικής Σύμπραξης στην Επαγγελματική Εκπαίδευση και Κατάρτιση

Τίτλος Προγράμματος: "Teaching and Learning Arduinos in Vocational Training"

Ακρονύμιο Προγράμματος: " ARDUinVET "

Κωδικός Προγράμματος: "2020-1-TR01-KA202-093762"

Εισαγωγή στα ARDUINOS
(ΒΑΣΙΚΑ ΣΤΟΙΧΕΙΑ των ARDUINOS)



Εισαγωγή στο ARDUINO

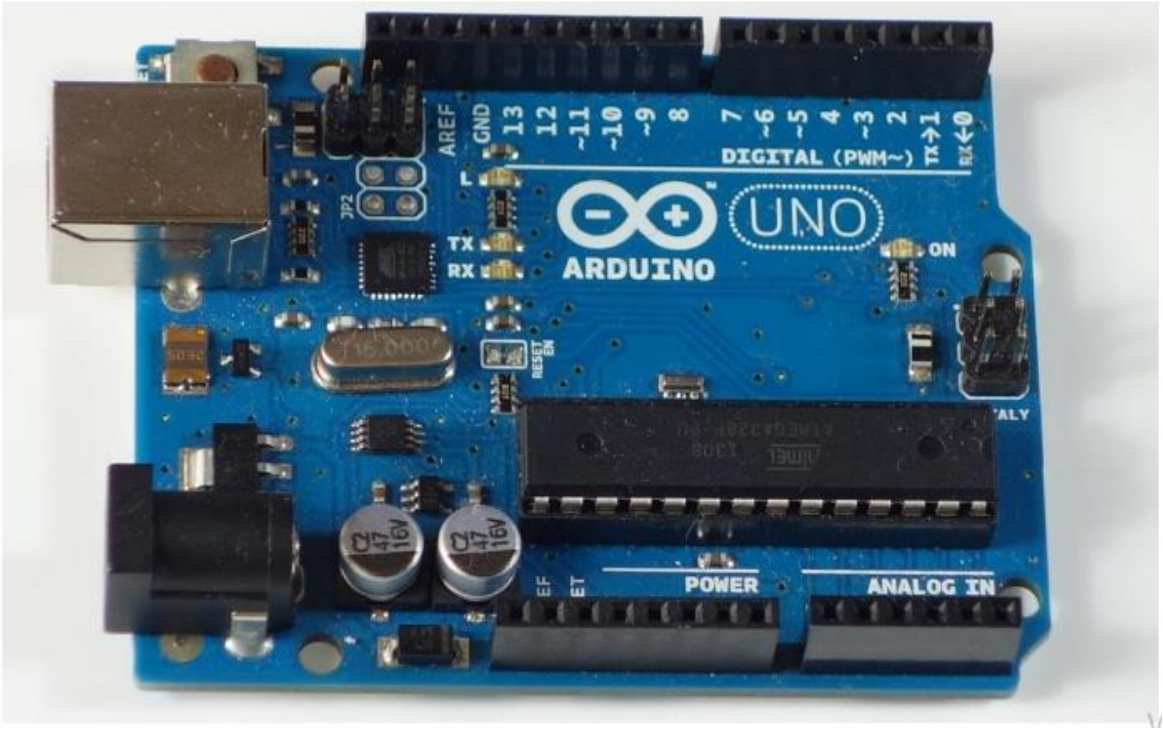
Το Arduino είναι μια πρωτότυπη πλατφόρμα (ανοιχτού κώδικα) που βασίζεται σε ένα εύχρηστο υλικό και λογισμικό. Αποτελείται από έναν τυπωμένο κύκλωμα (πλακέτα), το οποίο μπορεί να προγραμματιστεί (αναφέρεται ως μικροελεγκτής) και ένα έτοιμο λογισμικό που ονομάζεται Arduino IDE (Integrated Development Environment), το οποίο χρησιμοποιείται για τη σύνταξη και αποστολή του κώδικα από τον υπολογιστή στη φυσική πλακέτα.

Με άλλα λόγια, το Arduino είναι ένας μικρός υπολογιστής που μπορείτε να προγραμματίσετε για να διαβάζει πληροφορίες από τον φυσικό κόσμο γύρω σας και να στέλνει εντολές στον έξω κόσμο. Όλα αυτά είναι εφικτά επειδή μπορείτε να συνδέσετε πολλές συσκευές και εξαρτήματα στο Arduino για να κάνετε αυτό που θέλετε. Μπορείτε να κάνετε εκπληκτικά έργα, δεν υπάρχει όριο σε αυτό που μπορείτε να κάνετε και χρησιμοποιώντας τη φαντασία σας όλα είναι πιθανά!

Με απλά λόγια, το Arduino είναι ένα μικροσκοπικό σύστημα υπολογιστή που μπορεί να προγραμματιστεί με τις οδηγίες σας να αλληλεπιδρά με διάφορες μορφές εισόδου και εξόδου. Το τρέχον μοντέλο πλακέτας Arduino, το Uno, είναι αρκετά μικρό σε μέγεθος σε σύγκριση με το μέσο ανθρώπινο χέρι.

Τι είναι το Arduino?

Το Arduino είναι η πλακέτα που φαίνεται στο παρακάτω σχήμα.

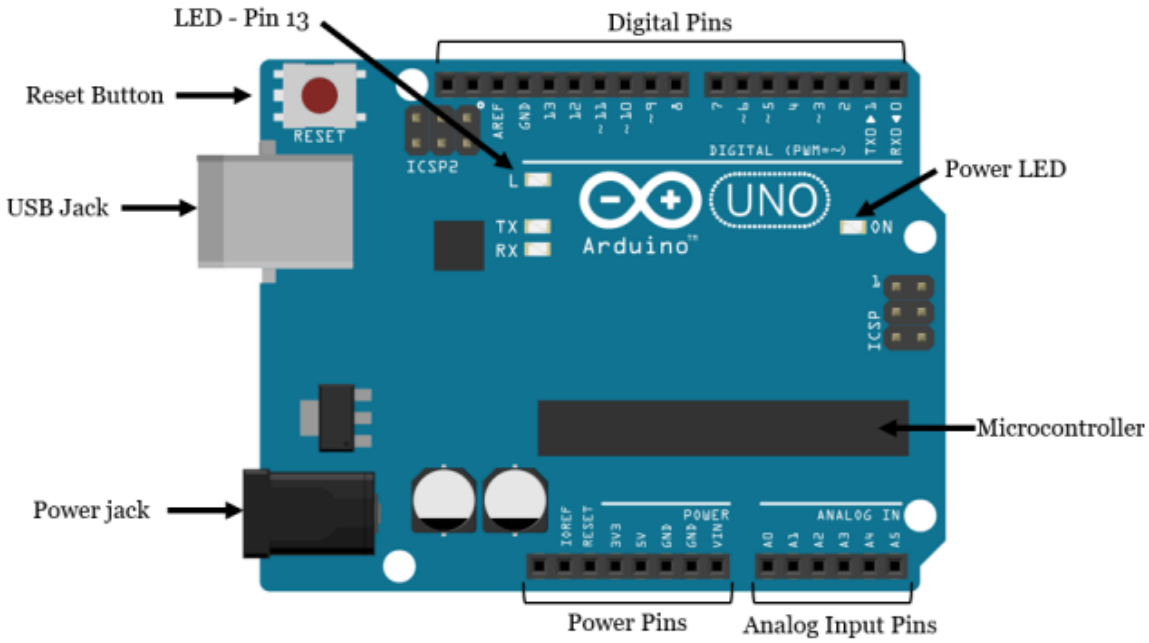


Βασικά, είναι ένας μικρό προγραμματιζόμενο κύκλωμα με επεξεργαστή (επίσης γνωστός ως μικροελεγκτής) που μπορεί να συνδέθεί σε διάφορα ηλεκτρικονικά κυκλώματα. Αυτό διευκολύνει την ανάγνωση εισόδου - ανάγνωση δεδομένων από τον φυσικό κόσμο - και έλεγχο εξόδου - αποστολή εντολής προς τον έξω κόσμο. Ο επεξεργαστής αυτής της πλακέτας (Arduino

Uno) είναι ένα τσιπ ATmega328p όπου μπορείτε να αποθηκεύσετε τα προγράμματά σας που θα προγραμματίσουν στο Arduino στο τι να κάνει.

Εξερευνώντας την πλακέτα Arduino Uno

Στο παρακάτω σχήμα, βλέπετε έναν πίνακα Arduino με ετικέτες. Ας δούμε τι κάνει το κάθε μέρος.



• **Microcontroller (Μικροελεγκτής):** Το chip ATmega328p είναι ο εγκέφαλος του Arduino. Όλα στην πλακέτα Arduino προορίζονται για την υποστήριξη αυτού του μικροελεγκτή. Εδώ αποθηκεύονται τα προγράμματα που δίνουν τις εντολές στο Arduino.

• **Digital pins (Ψηφιακές θύρες):** Το Arduino διαθέτει 14 ψηφιακές θύρες, με ετικέτα από 0 έως 13 που μπορούν να λειτουργήσουν ως είσοδοι ή έξοδοι. Όταν ρυθμιστούν ως είσοδοι, αυτοί οι ακροδέκτες μπορούν να διαβάσουν την τάση. Μπορούν να διαβάσουν μόνο δύο καταστάσεις: HIGH ή LOW. Όταν ορίζονται ως έξοδοι, αυτοί οι ακροδέκτες μπορούν να εφαρμόσουν τάση. Μπορούν να εφαρμόσουν μόνο 5V (HIGH) ή 0V (LOW).

• **PWM pins (PWM θύρες):** Πρόκειται για ψηφιακές θύρες που σημειώνονται με a ~ (ακίδες 11, 10, 9, 6, 5 και 3). Το PWM σημαίνει "διαμόρφωση πλάτους παλμού" και επιτρέπει στις ψηφιακές θύρες να παράγουν "εικονικές" μεταβλητές ποσότητες τάσης. Θα μάθετε περισσότερα για το PWM αργότερα.

• **TX and RX pins (TX και RX θύρες):** ψηφιακές θύρες 0 και 1. Το T σημαίνει «μετάδοση» και το R σημαίνει «λήψη». Το Arduino χρησιμοποιεί αυτές τις θύρες για να επικοινωνεί με άλλα ηλεκτρονικά μέσω Serial. Το Arduino χρησιμοποιεί επίσης αυτές τις θύρες για να επικοινωνεί με τον υπολογιστή σας κατά τη μεταφόρτωση νέου κώδικα. Αποφύγετε τη χρήση αυτών των θυρών για άλλες εργασίες εκτός από τη σειριακή επικοινωνία, εκτός εάν εξαντληθούν οι υπόλοιπες θύρες.

• **Φωτάκι LED στη ψηφιακή θύρα 13:** Αυτό είναι χρήσιμο για έναν εύκολο εκσφαλμάτωση στα διάφορα σενάρια του Arduino.

• **Φωτάκια LED TX και RX:** αυτά τα led αναβοσβήνουν όταν αποστέλλονται πληροφορίες μεταξύ του υπολογιστή και του Arduino.

• **Analog pins (Αναλογικές θύρες):** οι αναλογικές θύρες επισημαίνονται ως A0 έως A5 και χρησιμοποιούνται συχνά για την ανάγνωση αναλογικών αισθητήρων. Μπορούν να διαβάσουν διαφορετικές ποσότητες τάσης μεταξύ 0 και 5V. Επιπλέον, μπορούν επίσης να χρησιμοποιηθούν ως ψηφιακές θύρες εξόδου/εισόδου όπως οι ψηφιακές θύρες.

• **Power pins (θύρες** τροφοδοσίας): το Arduino παρέχει 3,3V ή 5V μέσω αυτών των θυρών. Αυτό είναι πολύ χρήσιμο αφού τα περισσότερα εξαρτήματα απαιτούν 3,3V ή 5V για να λειτουργήσουν. Οι θύρες με την ένδειξη "GND" είναι θύρες γείωσης.

• **Reset button (Κουμπί reset):** όταν πατάτε αυτό το κουμπί, επανεκκινείται το πρόγραμμα που τρέχει αυτήν τη στιγμή στο Arduino. Υπάρχει ακόμα μια θύρα επαναφοράς δίπλα στους ακροδέκτες τροφοδοσίας που λειτουργεί ως κουμπί επαναφοράς. Όταν εφαρμόζετε μια μικρή τάση σε αυτήν την θύρα, θα επαναφέρει το Arduino.

• **Power ON LED (LED λειτουργείας):** θα είναι αναμμένο όταν εφαρμοστεί τροφοδοσία στο Arduino.

• **USB jack (Υποδοχή USB):** χρειάζεστε αρσενικό καλώδιο USB A σε αρσενικό USB B (φαίνεται στο παρακάτω σχήμα) για να μεταφέρετε προγράμματα από τον υπολογιστή σας στην πλακέτα Arduino. Αυτό το καλώδιο τροφοδοτεί επίσης το Arduino σας.

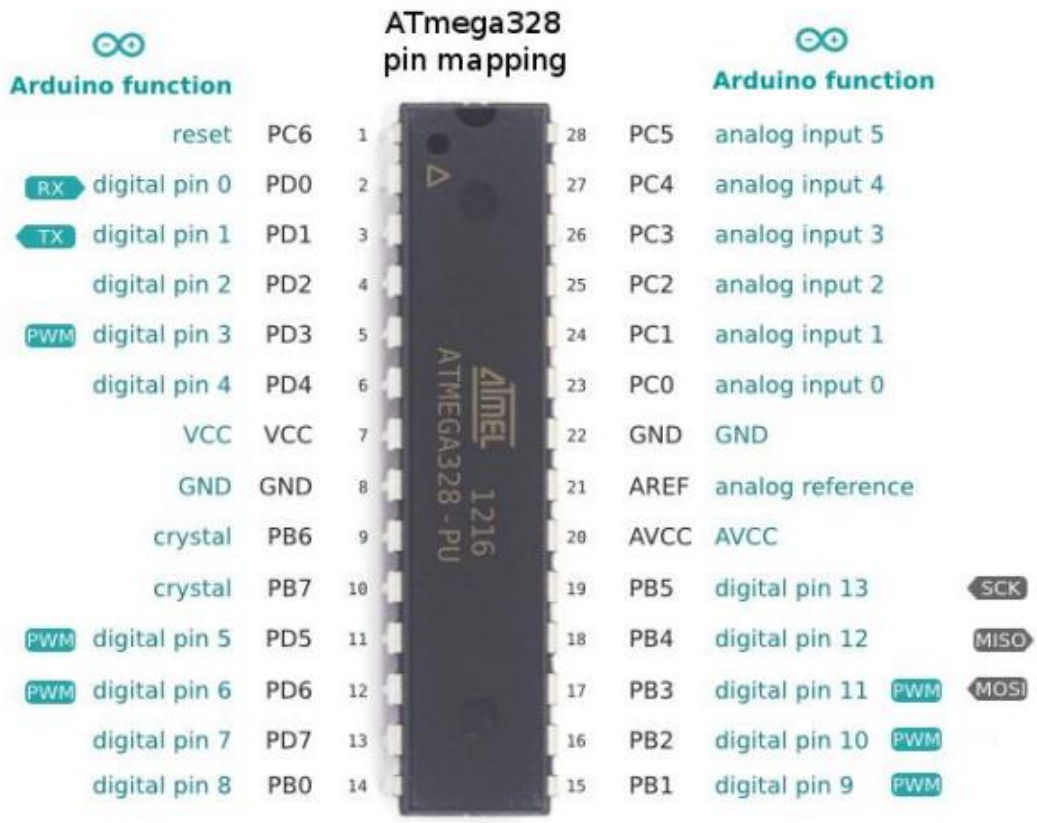
- **Power jack(**Υποδοχή τροφοδοσίας): μπορείτε να τροφοδοτήσετε το Arduino μέσω της υποδοχής τροφοδοσίας. Η συνιστώμενη τάση εισόδου είναι 7V έως 12V. Υπάρχουν διάφοροι τρόποι για να τροφοδοτήσετε το Arduino σας: για παράδειγμα. επαναφορτιζόμενες μπαταρίες, μπαταρίες μιας χρήσης και ηλιακά πάνελ.

Χαρακτηριστικά Arduino

Το Arduino Uno: διαθέτει μικροελεγκτή Atmel Atmega 328P και επίσης έχει είσοδο USB, είσοδο τροφοδοσίας, κουμπί επαναφοράς. Το Arduino διαθέτει όλα όσα πρέπει να έχει ένας μικροελεγκτής.

Μικροελεγκτής	Atmega328P
Τάση λειτουργίας	5V
Τάση εισόδου (συνιστόμενη)	7-12V
Τάση εισόδου (όριο)	6-20V
Ψηφιακές θύρες input / output	14
Θύρες PWM input / output	6
Αναλογικές θύρες input	6
Dc ένταση ανά θύρα input / output	20mA
DC ένταση για 3.3V	50mA
Flash memory	32 KB
Sram	2KB
EEPROM	1 KB
Clock Speed (συχνότητα ρολογιού)	16 MHz
Μήκος	68.6 mm
Πλάτος	53.4 mm
Βάρος	25 g

Πίνακας: Χαραστηριστικά Arduino Uno



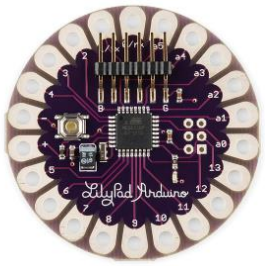
Σχήμα: Atmega 328P Pins

ΑΛΛΟΙ ΤΥΠΟΙ ARDUINOS

ARDUINO MEGA

Διαθέτει τον μικροελεγκτή Atmega 2560. Έχει 54 ψηφιακές θύρες εισόδου-εξόδου, 16 αναλογικές εισόδους, 4 σειριακές θύρες υλικού και έναν ταλαντωτή κρυστάλλου 16 mhz. Τροφοδοτείται από αντάπτορα USB και DC. Γενικά, η κάρτα, η οποία έχει τα ίδια χαρακτηριστικά με το Arduno UNO, προτιμάται σε μεγαλύτερα έργα επειδή έχει περισσότερες θύρες.	

ARDUINO LILYPAD

Το Lilypad έχει σχεδιαστεί για να ράβεται σε φορέματα και υφάσματα. Με αυτόν τον τρόπο, μπορεί να χρησιμοποιηθεί σε ενδιαφέροντα έργα που μπορούν να σχεδιαστούν ώστε να προσαρμοστούν σε ένδυμα. Διαθέτει μικροελεγκτή Atmega 168V.	

ARDUINO ETHERNET

Διαθέτει ένα Ethernet chip και μια θύρα Ethernet για την κατασκευή έργων που συνδέονται με το Διαδίκτυο. Υπάρχει επίσης μια υποδοχή κάρτας SD στη πλακέτα, η οποία έχει το μοντέλο Atmega 328 ως μικροελεγκτή.	

ARDUINO BLUETOOTH

Υπάρχει μια μονάδα Bluetooth στο Arduino BT, ιδανική για την επικοινωνία εφαρμογών με το πρωτόκολλο Bluetooth. Αυτή η ενότητα μπορεί επίσης να χρησιμοποιηθεί για τον προγραμματισμό του Arduino μέσω Bluetooth.	

ARDUINO MİNİ

Είναι ένα μοντέλο Arduino που έχει σχεδιαστεί για να λειτουργεί σε ένα breadboard ή να ενσωματωθεί σε άλλο σχέδιο. Υπάρχει μικροελεγκτής Atmega 168 ή Atmega 328 σε αυτό. Είναι ιδανικό για εφαρμογές όπου το μικρό μέγεθος είναι ιδιαίτερα σημαντικό.	

ARDUINO NANO

Είναι ένα πολύ μικρό μοντέλο κατάλληλο για εφαρμογές πάνω στην πλακέτα και διαθέτει μικροελεγκτή Atmega 328 ή Atmega 168, ρυθμιστή τάσης, τσιπ μετατροπέα σειριακού σε USB, θύρα εισόδου DC τάσης και θύρα mini USB.	

ARDUINO LEONARDO

Είναι μια από τις πλακέτες Arduino που περιέχουν έναν μικροελεγκτή Atmega 32u4 και δεν απαιτεί επιπλέον τσιπ για σύνδεση USB. Με 20 ψηφιακές εισόδους / εξόδους και 12 αναλογικές εισόδους, ο μικροελεγκτής στην πλακέτα διαθέτει κάλυμμα επιφάνειας. Χάρη στις δυνατότητες σύνδεσης USB, το Leonardo μπορεί να συνδεθεί στον υπολογιστή ως ποντίκι ή πληκτρολόγιο.	

ARDUINO ESPLORA

Το Esplora είναι μια πλακέτα Arduino που περιέχει διάφορους αισθητήρες, σε αντίθεση με τις άλλες. Χάρη στους αισθητήρες στην πλακέτα, είναι δυνατή η εκτέλεση πολλών εφαρμογών χωρίς να χρειάζονται άλλες προσθήκες και υπερβολική γνώση ηλεκτρονικών. Το Esplora είναι εξοπλισμένο με ποτενσιόμετρο, αισθητήρα φωτός και ήχου, αισθητήρα θερμοκρασίας, γεννήτρια ήχου, μίνι αναλογικό χειριστήριο 2 αξόνων, LED 3 χρωμάτων και επιταχυνσιόμετρο. Το Esplora είναι επίσης εξοπλισμένο με μικροελεγκτή Atmega 32U4 AVR όπως το Leonardo. Εφαρμογές που μπορούν να λειτουργήσουν ως ποντίκι ή πληκτρολόγιο μπορούν να αναπτυχθούν όταν συνδέθεί σε υπολογιστή μέσω σύνδεσης micro USB.	

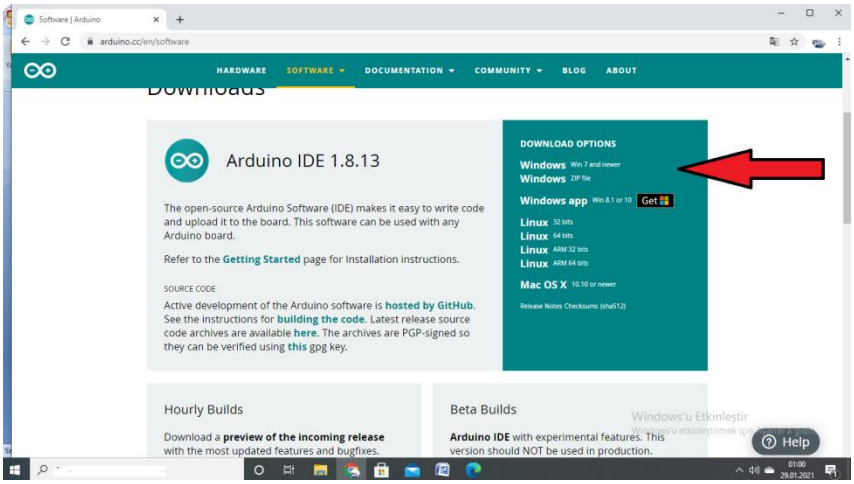
Κατεβάζοντας το Arduino IDE

Το Arduino IDE (Integrated Development Environment - Ολοκληρωμένο Περιβάλλον Ανάπτυξης) είναι το περιβάλλον όπου αναπτύσσετε τα προγράμματά σας που περιέχουν τις εντολές στο Arduino.

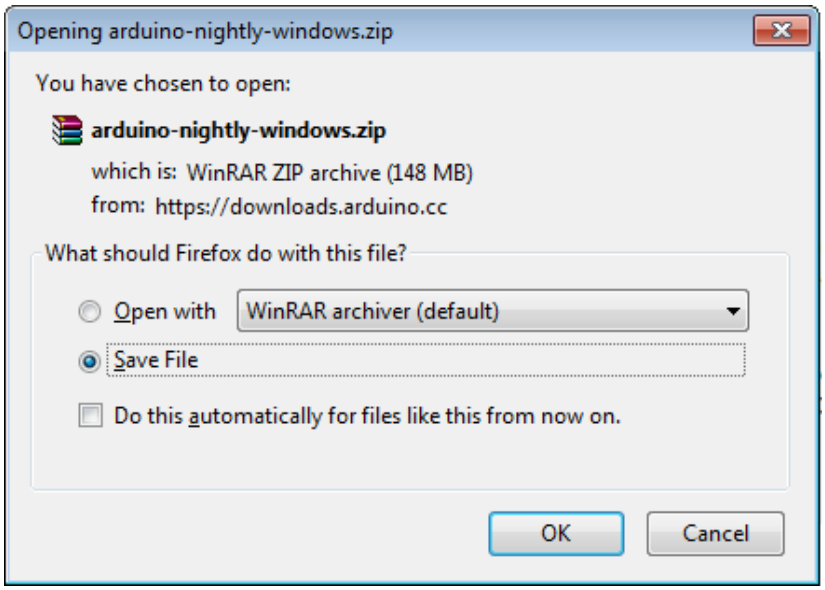
Μπορείτε να φορτώσετε νέα προγράμματα στο κύριο τσιπ, το ATmega328p, μέσω USB χρησιμοποιώντας το Arduino IDE.

Για να εγκαταστήσετε το Arduino IDE για Windows, πρέπει να ακολουθήσετε τις οδηγίες.

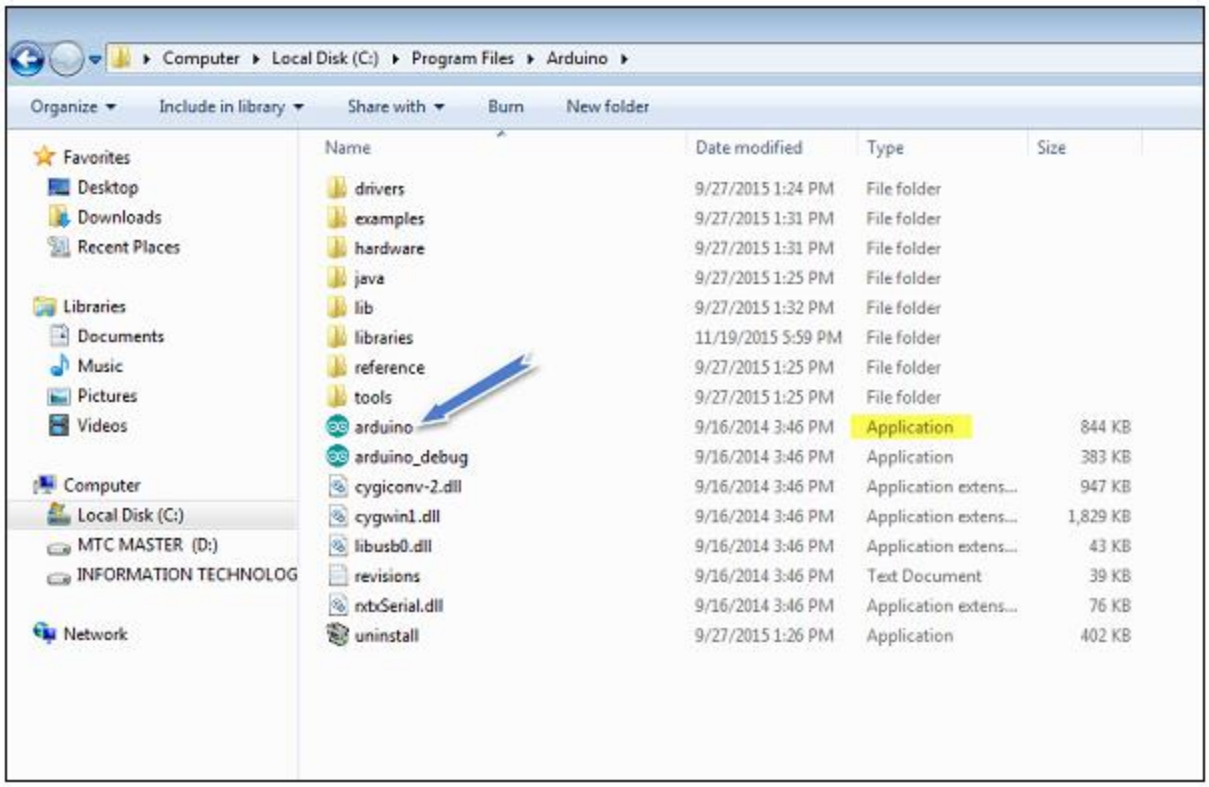
Για να κατεβάσετε το Arduino IDE, κάντε κλικ στον ακόλουθο σύνδεσμο: https://www.arduino.cc/en/Main/Software



Επιλέξτε ποιο λειτουργικό σύστημα χρησιμοποιείτε και κατεβάστε το. Μετά τη λήψη του λογισμικού Arduino IDE, πρέπει να αποσυμπιέσετε το φάκελο.

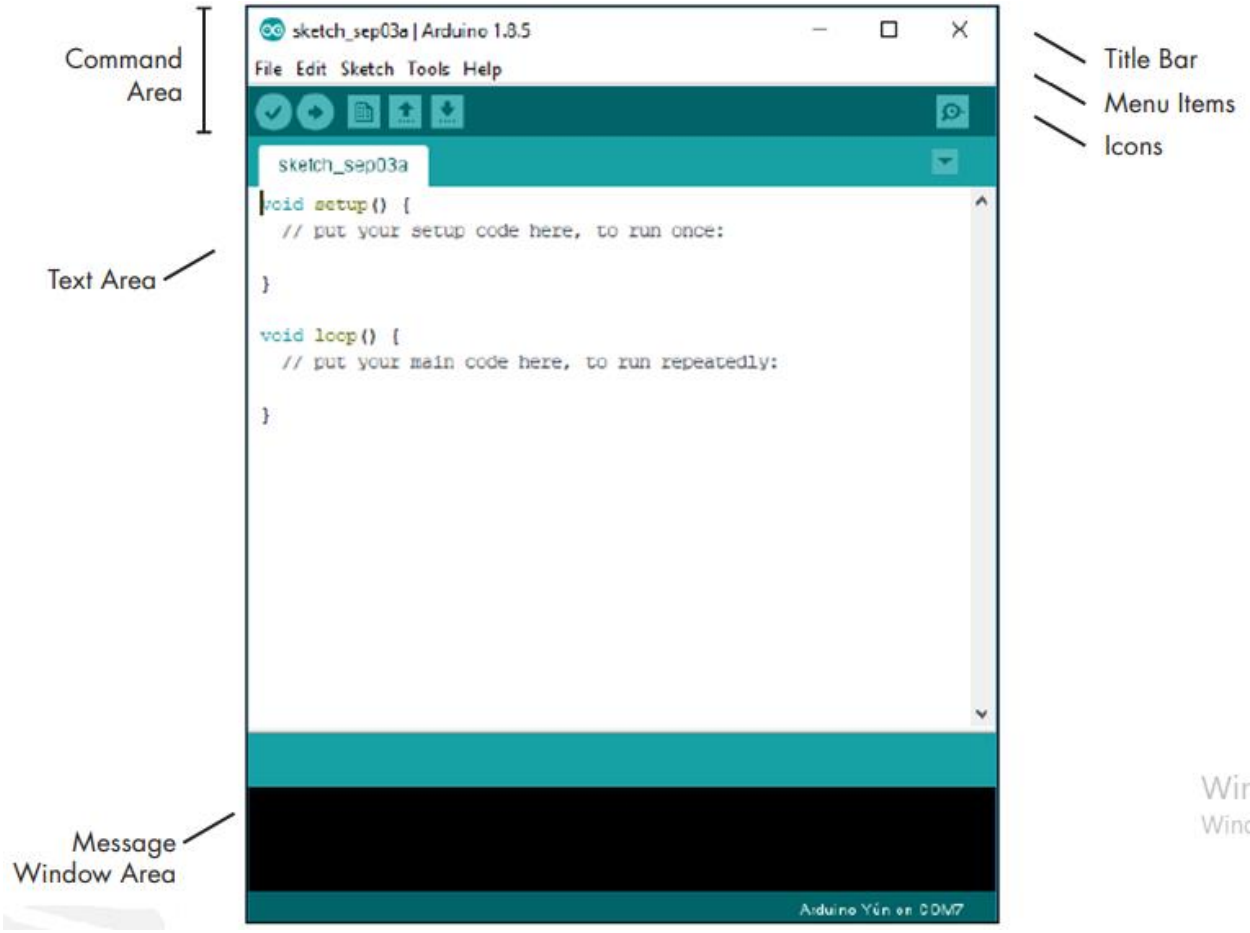


Μέσα στο φάκελο, μπορούμε να βρούμε το εικονίδιο της εφαρμογής (arduino.exe). Κάντε διπλό κλικ στο εικονίδιο για να ξεκινήσει το IDE. Στη συνέχεια, απλώς ακολουθήστε τον οδηγό εγκατάστασης για να εγκαταστήσετε το Arduino IDE.



Εφαρμογή Arduino IDE Windows για την κατασκευή προγραμμάτων

Όταν ανοίγετε για πρώτη φορά το Arduino IDE, θα πρέπει να δείτε κάτι παρόμοιο με το παρακάτω σχήμα. Όπως φαίνεται στο παρακάτω σχήμα, το Arduino IDE μοιάζει με έναν απλό επεξεργαστή κειμένου. Το IDE χωρίζεται σε τρεις κύριες περιοχές: την περιοχή εντολών, την περιοχή κειμένου και την περιοχή του παραθύρου μηνυμάτων.



Επιλογές Menu

Όπως συμβαίνει με οποιονδήποτε επεξεργαστή κειμένου ή επεξεργαστή κειμένου, μπορείτε να κάνετε κλικ σε ένα από τα στοιχεία του μενού για να εμφανίσετε τις διάφορες επιλογές του.

Αρχείο: Περιέχει επιλογές για αποθήκευση, φόρτωση και εκτύπωση σκίτσων. ένα πλήρες σύνολο παραδειγμάτων σκίτσων για άνοιγμα. καθώς και το υπομενού Προτιμήσεις.

Επεξεργασία: Περιέχει τις συνήθεις λειτουργίες αντιγραφής, επικόλλησης και αναζήτησης που είναι κοινές σε οποιονδήποτε επεξεργαστή κειμένου

Σκίτσο: Περιέχει τη λειτουργία επαλήθευσης του σκίτσου σας πριν από τη μεταφόρτωση σε έναν πίνακα και ορισμένους φακέλους σκίτσων και επιλογές εισαγωγής.

Εργαλεία: Περιέχει μια ποικιλία λειτουργιών καθώς και τις εντολές για να επιλέξετε τον τύπο της πλακέτας Arduino και τη θύρα USB.

Βοήθεια: Περιέχει συνδέσμους προς διάφορα θέματα που ενδιαφέρουν και την έκδοση του IDE.

File: Περιέχει επιλογές για αποθήκευση, φόρτωση και εκτύπωση σεναρίων. ένα πλήρες σύνολο παραδειγμάτων σεναρίων για επισκόπηση. καθώς και το υπομενού Preferences (Προτιμήσεις).

Edit: Περιέχει τις συνήθεις λειτουργίες αντιγραφής, επικόλλησης και αναζήτησης που είναι κοινές σε οποιονδήποτε επεξεργαστή κειμένου

Sketch: Περιέχει τη λειτουργία επαλήθευσης του σεναρίου σας πριν από τη μεταφόρτωση σε μια πλακλετα και ορισμένους φακέλους σεναρίων και επιλογές εισαγωγής.

Tools: Περιέχει διάφορες λειτουργίες καθώς και τις εντολές για να επιλέξετε τον τύπο της πλακέτας Arduino και τη θύρα USB.

Help: Περιέχει συνδέσμους προς διάφορα θέματα ενδιαφέροντος και την έκδοση του IDE.

Τι είναι το Σενάριο

Ένα σενάριο Arduino είναι ένα σύνολο εντολών που δημιουργείτε για να ολοκληρώσετε μια συγκεκριμένη εργασία. Με άλλα λόγια, ένα σενάριο είναι ένα πρόγραμμα.

Το σενάριο δεν είναι παρά μια σειρά εντολών που πρέπει να εκτελέσει το Arduino. Τα σενάρια που δημιουργήθηκαν χρησιμοποιώντας το Arduino IDE αποθηκεύονται ως αρχεία .pde. Ένα σενάριο περιέχει τα τρία κύρια μέρη: Δηλώσεις μεταβλητών, συνάρτηση Setup και κύρια συνάρτηση Loop.

Κουμπιά Arduino IDE Toolbar

Κάτω από τη γραμμή εργαλείων του μενού υπάρχουν έξι εικονίδια. Περάστε πάνω από κάθε εικονίδιο για να εμφανιστεί το όνομά του. Τα εικονίδια, από αριστερά προς τα δεξιά, είναι:

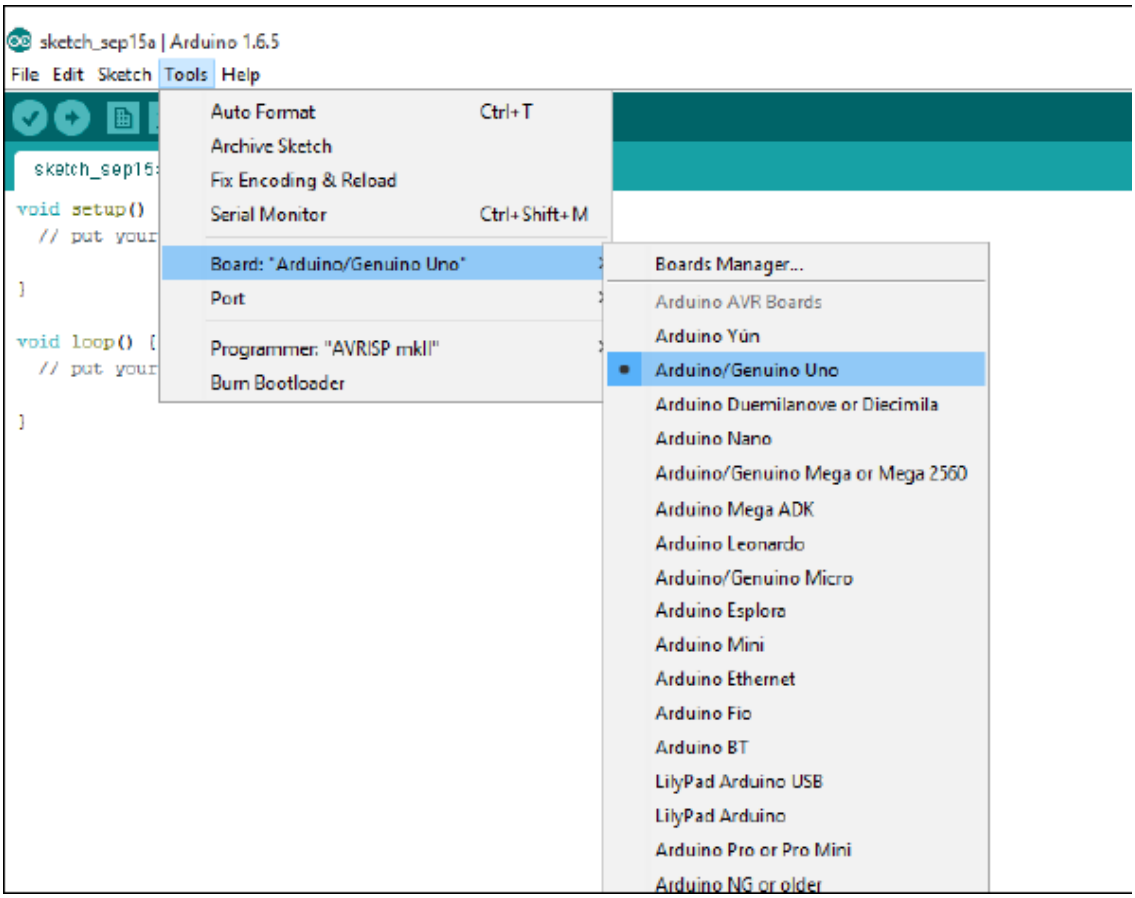
	**Verify (Compile):** Κάντε κλικ σε αυτό για να ελέγξετε ότι το σκίτσο Arduino είναι έγκυρο και δεν περιέχει προγραμματιστικά σφάλματα.
	**New:** Επιλέξτε για να ανοίξετε ένα νέο κενό σενάριο σε νέο παράθυρο.
	**Open:** Κάντε κλικ σε αυτό για να ανοίξετε ένα αποθηκευμένο σενάριο.
	**Save:** Κάντε κλικ σε αυτό για να αποθηκεύσετε το ανοιχτό σενάριο. Εάν το σενάριο δεν έχει όνομα, θα σας ζητηθεί να δώσετε ένα.
	**Upload:** Κάντε κλικ σε αυτό για επαλήθευση και, στη συνέχεια, το ανέβασμα υ σεναρίου σας στην πλακέτα Arduino.
	**Serial Monitor:** Κάντε κλικ σε αυτό για να ανοίξετε ένα νέο παράθυρο για χρήση στην αποστολή και λήψη δεδομένων μεταξύ του Arduino και του IDE.

Σύνδεση του Arduino σας

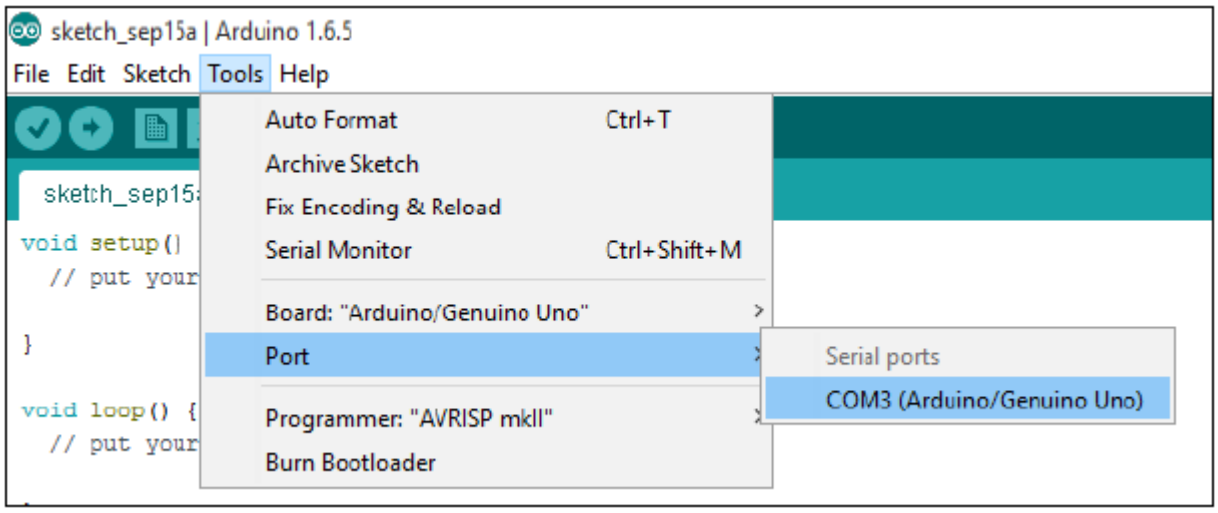
Συνδέστε το Arduino UNO στον υπολογιστή σας μέσω USB.

Αφού συνδέσετε το Arduino με καλώδιο USB, πρέπει να βεβαιωθείτε ότι το Arduino IDE έχει επιλέξει τη σωστή πλακέτα.

Στην περίπτωσή μας, χρησιμοποιούμε το Arduino Uno, οπότε επιλέγουμε το **Tools ▸ Board: ▸ Arduino/Genuino Uno.**



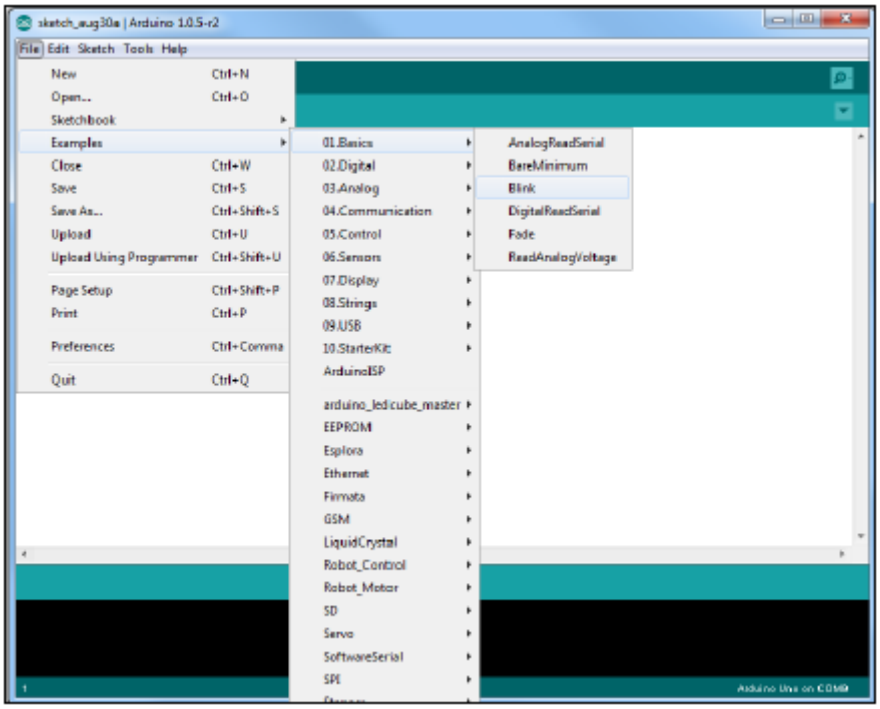
Στη συνέχεια, θα πρέπει να επιλέξετε τη σειριακή θύρα στην οποία είναι συνδεδεμένο το Arduino. Μεταβείτε στο **Tools ▸ Port** και επιλέξτε τη σωστή θύρα.



Ανεβάζοντας ένα Σενάριο Arduino

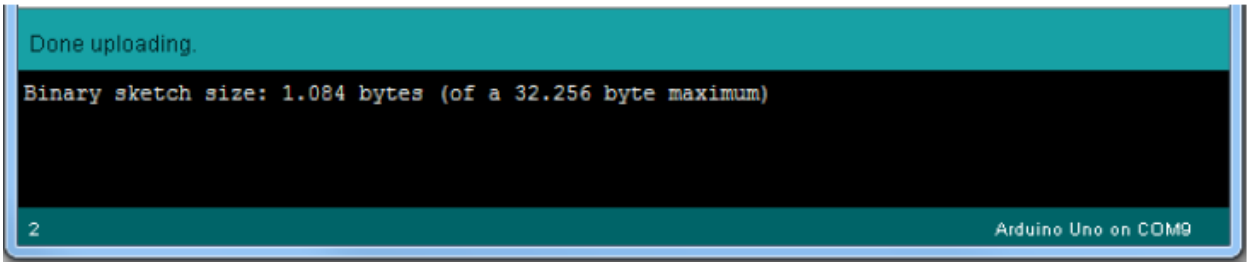
Για να σας δείξουμε πώς να ανεβάζετε κώδικα στην πλακέτα Arduino, θα σας δείξουμε ένα απλό παράδειγμα. Αυτό είναι ένα από τα πιο βασικά παραδείγματα - συνίσταται στο να αναβοσβήνει το ενσωματωμένο LED στη ψηφιακή θύρα 13 κάθε δευτερόλεπτο.

1. Ανοίξτε το Arduino IDE.
2. Επιλέξτε **File ▸ Examples ▸ 01.Basics ▸ Blink**



Το Arduino IDE έχει ως προεπιλογή το Arduino UNO. Κάντε κλικ στο κουμπί **Upload** και περιμένετε μερικά δευτερόλεπτα.

Μετά από μερικά δευτερόλεπτα, θα πρέπει να δείτε ένα μήνυμα **Done uploading**.



Αυτός ο κωδικός αναβοσβήνει απλά το ενσωματωμένο LED στο Arduino UNO (επισημαίνεται με κόκκινο χρώμα). Θα πρέπει να δείτε το μικρό LED να ανάβει για ένα δευτερόλεπτο και να απενεργοποιείται για ένα άλλο δευτερόλεπτο επανειλημμένα.

Διάβασμα εισόδου και έλεγχος εξόδου

Ένας πίνακας Arduino περιέχει ψηφιακές θύρες, αναλογικές θύρες και θύρες PWM.

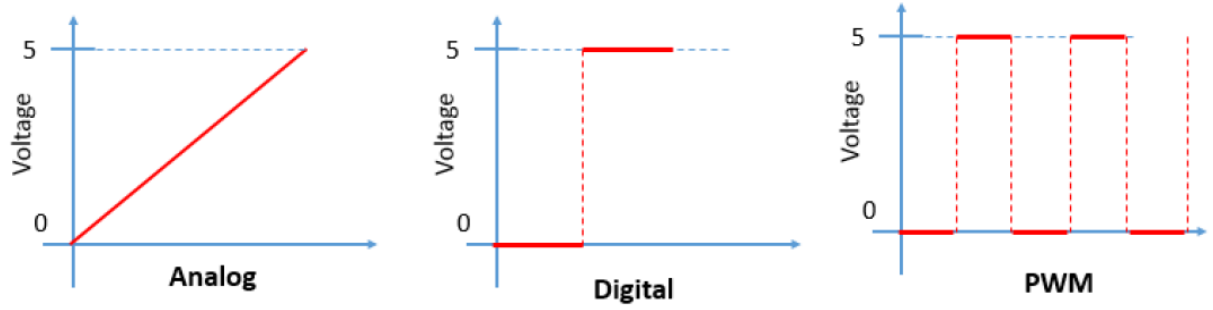
Διαφορά μεταξύ ψηφιακών, αναλογικών και θυρών PWM

Στις **ψηφιακές θύρες**, έχετε μόνο δύο πιθανές καταστάσεις, οι οποίες είναι on ή off. Αυτά μπορούν επίσης να αναφέρονται ως High ή Low, 1 ή 0 και 5V ή 0V.

Για παράδειγμα, εάν ένα LED είναι αναμμένο, τότε, η κατάστασή του είναι High ή 1 ή 5V. Εάν είναι απενεργοποιημένο, θα έχει Low ή 0 ή 0V.

Στις **αναλογικές θύρες**, έχετε απεριόριστες πιθανές καταστάσεις μεταξύ 0 και 1023. Αυτό σας επιτρέπει να διαβάζετε τιμές αισθητήρων. Για παράδειγμα, με έναν αισθητήρα φωτός, εάν είναι πολύ σκοτεινό, θα διαβάσετε 1023, αν είναι πολύ φωτεινό θα διαβάσετε 0 Εάν υπάρχει φωτεινότητα μεταξύ σκοτεινού και πολύ φωτεινού θα διαβάσετε μια τιμή μεταξύ 0 και 1023.

Οι **θύρες PWM** είναι ψηφιακές θύρες, επομένως έχουν έξοδο είτε 0 είτε 5V. Ωστόσο, αυτές οι θύρες μπορούν να εξάγουν "εικονικές" ενδιάμεσες τιμές τάσης μεταξύ 0 και 5V, επειδή μπορούν να εκτελέσουν «Διαμόρφωση πλάτους παλμού» (PWM). Το PWM επιτρέπει την «προσομοίωση» διαφόρων επιπέδων ισχύος ταλαντεύοντας την τάση εξόδου του Arduino.



Έλεγχος μιας εξόδου

Για τον έλεγχο μιας ψηφιακής εξόδου, χρησιμοποιείτε τη λειτουργία digitalWrite () και ανάμεσα σε αγκύλες που γράφετε, τη θύρα που θέλετε να ελέγξετε και, στη συνέχεια, HIGH ή LOW.

Για να ελέγξετε μια θύρα PWM, χρησιμοποιείτε τη λειτουργία analogWrite () και ανάμεσα σε αγκύλες γράφετε τη θύρα που θέλετε να ελέγξετε και έναν αριθμό μεταξύ 0 και 255.

Διαβάζοντας μια είσοδο

Για να διαβάσετε μια αναλογική είσοδο χρησιμοποιείτε τη συνάρτηση analogRead () και για μια ψηφιακή είσοδο χρησιμοποιείτε το digitalRead ().

Σημείωση: Ο καλύτερος τρόπος για να μάθετε το Arduino είναι η εξάσκηση. Κάντε, λοιπόν, πολλά έργα και ξεκινήστε να χτίζετε κάτι.

Erasmus+ KA-202

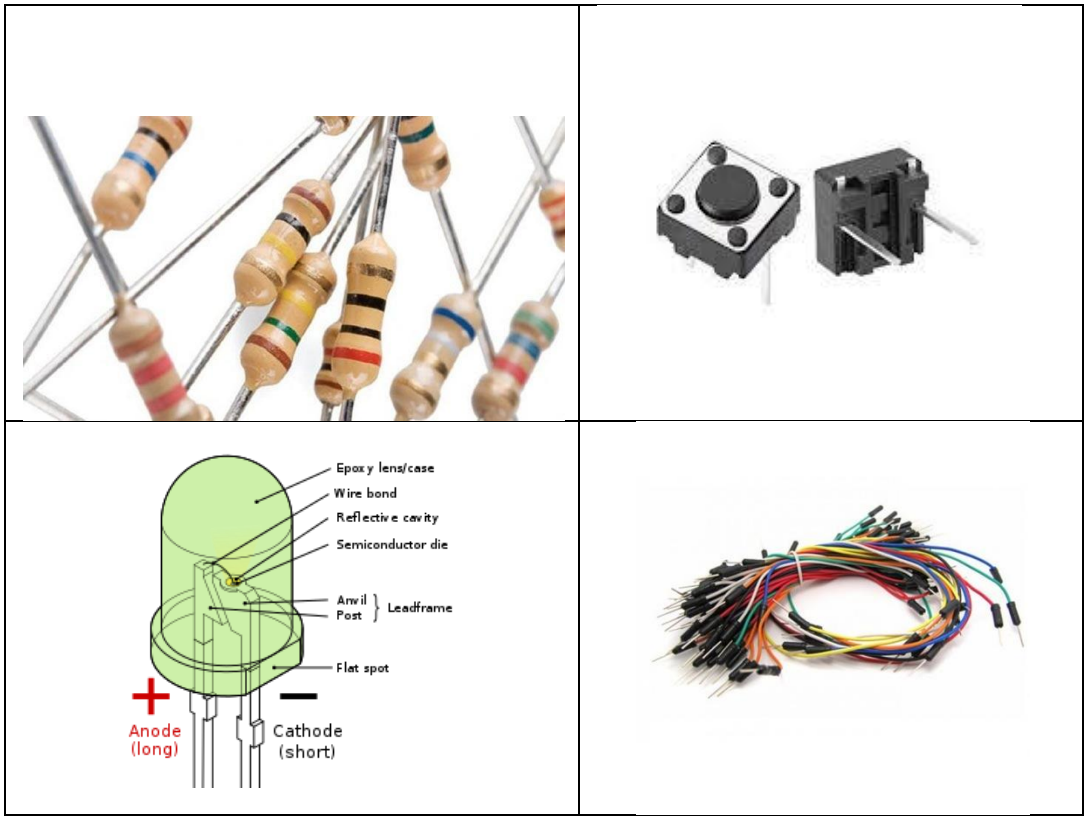
Πρόγραμμα Στρατηγικής Σύμπραξης στην Επαγγελματική Εκπαίδευση και Κατάρτιση

Τίτλος Προγράμματος: "Teaching and Learning Arduinos in Vocational Training"

Ακρονύμιο Προγράμματος: " ARDUinVET "

Κωδικός Προγράμματος: "2020-1-TR01-KA202-093762"

Arduino Input/Output και κιτ



Προγραμματισμός των έργων σας

Κατά την έναρξη των πρώτων σας έργων, μπορεί να μπείτε στον πειρασμό να γράψετε το σενάριο σας αμέσως μετά την αρχική ιδέα. Πριν όμως ξεκινήσετε να γράφετε, πρέπει να ακολουθήσετε μερικά βασικά προπαρασκευαστικά βήματα. Άλλωστε, η πλακέτα Arduino δεν μπορεί να διαβάσει τις σκέψεις σας. χρειάζεται ακριβείς οδηγίες. Ακόμη και αν αυτές οι οδηγίες μπορούν να εκτελεστούν από το Arduino, τα αποτελέσματα μπορεί να μην είναι αυτά που περιμένατε αν έχετε παραβλέψει έστω και μια μικρή λεπτομέρεια.

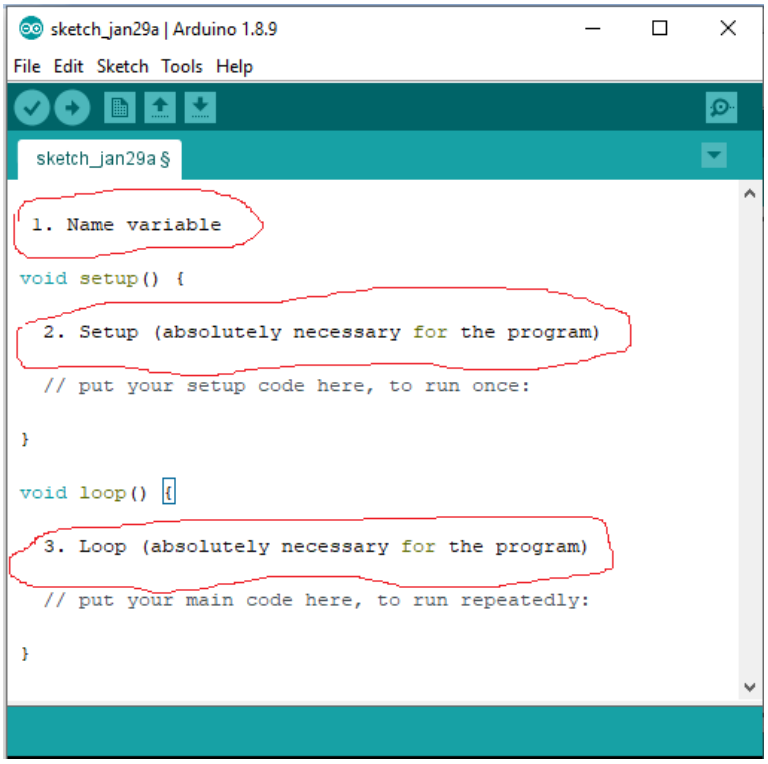
Είτε δημιουργείτε ένα έργο που απλά αναβοσβήνει ένα LED είτε ένα αυτοματοποιημένο σήμα σιδηροδρομικού μοντέλου, ένα λεπτομερές σχέδιο είναι το θεμέλιο της επιτυχίας. Κατά τον σχεδιασμό των έργων σας στο Arduino, ακολουθήστε αυτά τα βασικά βήματα:

1. Καθορίστε τον στόχο σας. Καθορίστε τι θέλετε να επιτύχετε.
2. Γράψτε τον αλγόριθμό σας. Ένας αλγόριθμος είναι ένα σύνολο οδηγιών που περιγράφουν πώς να ολοκληρώσετε το έργο σας. Ο αλγόριθμός σας θα πρέπει να αναφέρει τα απαραίτητα βήματα για να επιτύχετε τον στόχο του έργου σας.
3. Επιλέξτε το υλικό σας. Καθορίστε πώς θα συνδεθεί με το Arduino.
4. Γράψτε το σενάριο σας. Δημιουργήστε το αρχικό σας πρόγραμμα που λέει στο Arduino τι να κάνει.
5. Συνδέστε το. Συνδέστε το υλικό σας, τα κυκλώματα και άλλα στοιχεία στην πλακέτα Arduino.
6. Δοκιμάστε και εντοπίστε σφάλματα. Λειτουργεί; Κατά τη διάρκεια αυτού του σταδίου, εντοπίζετε σφάλματα και βρίσκετε τις αιτίες τους, είτε στο σενάριο, το υλικό ή τον αλγόριθμο.

Όσο περισσότερο χρόνο αφιερώνετε στον προγραμματισμό του έργου σας, τόσο πιο εύκολη θα είναι η διαδικασία του σταδίου δοκιμών και εντοπισμού σφαλμάτων.

Βασική δομή ενός σεναρίου

Ένα πρόγραμμα Arduino ονομάζεται “σενάριο”. Ένα σενάριο χωρίζεται σε 3 μέρη.



1. Δήλωση μεταβλητών:
Στο πρώτο μέρος δηλώνονται οι μεταβλητές του προγράμματος. Αυτό το μέρος δεν είναι απολύτως απαραίτητο.

2. Setup (απολύτως απαραίτητο για το πρόγραμμα):
Η συνάρτηση setup θα εκτελεστεί μόνο μία φορά στην εκκίνηση. Εδώ ορίζετε στο πρόγραμμα για παράδειγμα ποιά θύρα (υποδοχή για καλώδια) πρέπει να είναι είσοδος και ποιά έξοδος.
Ορισμός ως έξοδος: Η θύρα πρέπει να βγάζει τάση. Για παράδειγμα: Με αυτόν τη θύρα μπορεί να ανάψει ένα LED.
Ορισμός ως είσοδος: Η πλακέτα πρέπει να διαβάζει μια τάση. Για παράδειγμα: Ένας διακόπτης αλλάζει κατάσταση. Ο πίνακας το αναγνώρισε αυτό, επειδή παίρνει μια τάση στη θύρα εισόδου.

3. Loop (απολύτως απαραίτητο για το πρόγραμμα):
Η συνάρτηση loop θα επαναλαμβάνεται συνεχώς από την πλακέτα. Το σενάριο εκτελείται από Την αρχή ως το τέλος, κατόπιν ξεκινά ξανά από την αρχή και ούτω καθεξής.

Επιπλέον συντακτικοί κανόνες

Είναι απαραίτητο να δώσετε προσοχή σε αυτούς τους κανόνες κατά τη σύνταξη προγραμμάτων Arduino. Διαφορετικά, το πρόγραμμά σας θα αποτύχει.

; (ερωτηματικό):

; (ερωτηματικό) χρησιμοποιείται για το τέλος μιας εντολής. Το να ξεχάσετε να τερματίσετε μια εντολή με ερωτηματικό θα οδηγήσει σε σφάλμα μεταγλωττιστή.

Παράδειγμα: int a=13;

{} (Αγκύλες):

Οι αγκύλες είναι ένα σημαντικό μέρος της γλώσσας προγραμματισμού Arduino. Χρησιμοποιούνται σε πολλές διαφορετικές κατασκευές και αυτό μπορεί μερικές φορές να προκαλέσει σύγχυση στους αρχάριους. Μια αρχική αγκύλη "{" πρέπει πάντα να ακολουθείται από μία τελική αγκύλη "}".

Οι κύριες χρήσεις των αγκυλών: Functions, Loops, Δηλώσεις

Παράδειγμα:

```
void myfunction(datatype argument)
{        εντολή/ές        }
```

// (Σχόλιο μιας γραμμής) and /* */ (Σχόλιο πολλών γραμμών):

Αυτές είναι γραμμές του προγράμματος που χρησιμοποιούνται για να ενημερώσετε τον εαυτό σας ή άλλους σχετικά με τον τρόπο λειτουργίας του προγράμματος. Αγνοούνται από τον μεταγλωττιστή και δεν εξάγονται στον επεξεργαστή, οπότε δεν καταλαμβάνουν καθόλου χώρο στο chip Atmega. Ο σκοπός των σχολίων είναι μόνο να σας βοηθήσει να καταλάβετε (ή να θυμηθείτε) πώς λειτουργεί το πρόγραμμά σας ή να ενημερώσετε άλλους πώς λειτουργεί. Υπάρχουν δύο διαφορετικοί τρόποι για να επισημάνετε μια γραμμή ως σχόλιο:

Παράδειγμα:

x = 5; // Αυτό είναι ένα σχόλιο μιας γραμμής. Οτιδήποτε μετά τις καθέτους είναι σχόλιο //μέχρι το τέλος της γραμμής

x = 5; /* Αυτό είναι ένα σχόλιο πολλών γραμμών. …..... Αυτό είναι το τέλος ενός σχολίου πολλών γραμμών. */

#define:

#define επιτρέπει στον προγραμματιστή να δώσει ένα όνομα σε μια σταθερά πριν την μεταγλώττιση του προγράμματος. Οι ορισμένες σταθερές στο Arduino δεν καταλαμβάνουν χώρο μνήμης στο τσιπ. Γενικά, η δεσμευμένη λέξη const προτιμάται για τον καθορισμό σταθερών και πρέπει να χρησιμοποιείται αντί της #define.

Παράδειγμα:

#define ledPin 3 // Ο μεταγλωττιστής θα αντικαταστήσει κάθε αναφορά του ledPin με την τιμή 3 κατά τη μεταγλώττιση.

#include:

#include Χρησιμοποιείται για να συμπεριλάβει εξωτερικές βιβλιοθήκες στο σενάριό σας. Αυτό δίνει στον προγραμματιστή πρόσβαση σε μια μεγάλη ομάδα βιβλιοθηκών C (ομάδες προκαθορισμένων function), καθώς και βιβλιοθήκες που έχουν γραφτεί ειδικά για το Arduino.

Παράδειγμα:

#include <servo.h>

Arduino – Τύποι Δεδομένων

Οι τύποι δεδομένων χρησιμοποιούνται για τη δήλωση μεταβλητών ή συναρτήσεων διαφορετικών τύπων. Ο τύπος μιας μεταβλητής καθορίζει πόσο χώρο καταλαμβάνει στο χώρο αποθήκευσης και πώς ερμηνεύεται η αποθηκευμένη ακολουθία bit.

Οι εκφράσεις που χρησιμοποιούνται για την αποθήκευση οποιασδήποτε πληροφορίας στη μνήμη και μπορούν να αλλάξουν την τιμή κατά τη ροή του προγράμματος ονομάζονται μεταβλητές. Οι μεταβλητές μπορεί να είναι αριθμοί, χαρακτήρες ή λογικές εκφράσεις. Ο κατάλληλος τύπος δεδομένων θα πρέπει να επιλέγεται σύμφωνα με τον τύπο της μεταβλητής. Μια συγκεκριμένη περιοχή κατανέμεται ανάλογα με τον τύπο των δεδομένων που χρησιμοποιούνται για τον καθορισμό της μεταβλητής στη μνήμη.

Ο παρακάτω πίνακας παρέχει όλους τους τύπους δεδομένων που θα χρησιμοποιήσετε κατά τη διάρκεια του προγραμματισμού Arduino.

Τύπος	**Τιμές**
boolean	Περιέχει true ή false
char	-128 to 127
byte	0 to 255
unsigned char	0 to 255
int	-32,768 to 32,767
unsigned int	0 to 65,535
word	(το ίδιο όπως unsigned int)
long (or long int)	-2,147,483,648 to 2,147,483,647
unsigned long	0 to 4,294,967,295
float	-3.4028235E+38 to 3.4028235E+38
double	(το ίδιο όπως float)

Arduino – Μεταβλητές και σταθερές

Κατά τον ορισμό της μεταβλητής, πρέπει να καθοριστεί το όνομα της μεταβλητής, η τιμή της μεταβλητής και ο κατάλληλος τύπος δεδομένων για τη μεταβλητή.

Ο ορισμός γίνεται όπως φαίνεται στο παρακάτω παράδειγμα:

int LED =12;

Εδώ: **int**=τύπος δεδομένων **LED**=όνομα μεταβλητής **12**=τιμή μεταβλητής

Οι μεταβλητές, τις οποίες χρησιμοποιεί το Arduino, έχουν μια ιδιότητα που ονομάζεται εμβέλεια. Η εμβέλεια αφορά μια περιοχή του προγράμματος και υπάρχουν τρία μέρη όπου μπορούν να δηλωθούν μεταβλητές. Αυτοί είναι:

• Μέσα σε μια συνάρτηση ή ένα μπλοκ, και ονομάζονται τοπικές μεταβλητές.

• Στον ορισμό των παραμέτρων συνάρτησης, και ονομάζονται τυπικές παράμετροι.

• Εκτός όλων των συναρτήσεων, και ονομάζονται καθολικές μεταβλητές.

Μια σταθερά είναι ένας *προσδιοριστής* μεταβλητής που τροποποιεί τη συμπεριφορά της μεταβλητής, καθιστώντας μια μεταβλητή "*μόνο για ανάγνωση*". Αυτό σημαίνει ότι η μεταβλητή μπορεί να χρησιμοποιηθεί όπως κάθε άλλη μεταβλητή του τύπου της, αλλά η τιμή της δεν μπορεί να αλλάξει. Αν προσπαθήσετε να αλλάξετε μια τιμή σε μια μεταβλητή const θα λάβετε ένα σφάλμα μεταγλώτισσης.

Οι σταθερές που ορίζονται με τη δεσμευμένη λέξη const υπακούουν στους κανόνες της εμβέλειας μεταβλητών που διέπουν και τις άλλες μεταβλητές. Αυτό, και οι παγίδες της χρήσης της #define, καθιστά τη δεσμευμένη λέξη const καλύτερη πρακτική για τον καθορισμό σταθερών και προτιμάται από τη χρήση #define

Παράδειγμα:
const float pi = 3.14;

Παρατήρηση:
#define ή const: Μπορείτε να χρησιμοποιήσετε είτε const είτε #define για τη δημιουργία αριθμητικών ή σταθερών συμβολοσειρών. Για πίνακες, θα χρειαστεί να χρησιμοποιήσετε const. Γενικά το const προτιμάται από το #define για τον καθορισμό σταθερών.

Arduino – Τελεστές

Ο τελεστής είναι ένα σύμβολο που λέει στον μεταγλωττιστή να εκτελέσει συγκεκριμένες μαθηματικές ή λογικές συναρτήσεις. Στο Arduinos, μπορούν να χρησιμοποιηθούν οι ακόλουθοι τύποι τελεστών.

Αριθμητικοί τελεστές:

Ας υποθέσουμε ότι η μεταβλητή A έχει τιμή 10 και η μεταβλητή B έχει τιμή 20 τότε -

Ονομα Τελεστή	Απλή μορφή	Παράδειγμα
εκχώρηση	=	A = B
πρόσθεση	+	A + B will give 30
αφαίρεση	-	A - B will give -10
πολλαπλασιασμός	*	A * B will give 200
διαίρεση	/	B / A will give 2
Ακέραια διαίρεση	%	B % A will give 0

Σχεσιακοί τελεστές:

Ας υποθέσουμε ότι η μεταβλητή A έχει τιμή 10 και η μεταβλητή B έχει τιμή 20 τότε -

Ονομα Τελεστή	Σύμβολο	Παράδειγμα
ίσο με	==	(A == B) is not true
διάφορο	!=	(A != B) is true
μικρότερο από	<	(A < B) is true
μεγαλύτερο από	>	(A > B) is not true
μικρότερο ή ίσο με	<=	(A <= B) is true
μεγαλύτερο ή ίσο με	>=	(A >= B) is not true

Λογικοί τελεστές (Boolean):

Ας υποθέσουμε ότι η μεταβλητή A έχει τιμή 10 και η μεταβλητή B έχει τιμή 20 τότε -

Ονομα Τελεστή	Απλή μορφή	Παράδειγμα
and	&&	(A && B) : true
or	||	(A || B) : true
not	!	!(A && B) : false

Τελεστές μετατόπισης Bit (Bitwise):

Ας υποθέσουμε ότι η μεταβλητή A έχει τιμή 60 και η μεταβλητή B έχει τιμή 13 τότε -

Ονομα Τελεστή	Απλή μορφή	Παράδειγμα
and	&	(A & B) επιστρέφει 12 που είναι 0000 1100
or	|	(A | B) επιστρέφει 61 που είναι 0011 1101
xor	^	(A ^ B) επιστρέφει 49 που είναι 0011 0001
not	~	(~A) επιστρέφει -60 που είναι 1100 0011
shift left	<<	A << 2 επιστρέφει 240 που είναι 1111 0000
shift right	>>	A >> 2 επιστρέφει 15 που είναι 0000 1111

Arduino – Συναρτήσεις I/O (Εντολές)

Οι συναρτήσεις επιτρέπουν τη δόμηση των προγραμμάτων για την εκτέλεση μεμονωμένων εργασιών. Η συνήθης περίπτωση για τη δημιουργία μιας συνάρτησης είναι όταν κάποιος χρειάζεται να εκτελέσει την ίδια ενέργεια πολλές φορές σε ένα πρόγραμμα. Θα είναι πιο σαγές σαφέστερα όταν δείξουμε παρακάτω παραδείγματα κώδικα σε κυκλώματα και προγράμματα Arduino παρακάτω. Για να εξηγήσουμε τις διάφορες συναρτήσεις εντολών, τις χωρίσαμε σε ξεχωριστές Εντολές.

Οι θύρες στην πλακέτα Arduino μπορούν να οριστούν είτε ως είσοδοι είτε ως έξοδοι. Θα εξηγήσουμε τη λειτουργία των θυρών σε αυτούς τους ορισμούς. Είναι σημαντικό να σημειωθεί ότι η πλειοψηφία των αναλογικών θυρών Arduino, μπορεί να διαμορφωθεί και να χρησιμοποιηθεί, με τον ίδιο ακριβώς τρόπο όπως και οι ψηφιακές θύρες.

pinMode() Function:

Η συνάρτηση pinMode () χρησιμοποιείται για τον ορισμό μιας συγκεκριμένης θύρας ώστε να συμπεριφέρεται είτε ως είσοδος είτε ως έξοδος. Είναι δυνατή η ενεργοποίηση των εσωτερικών αντιστάσεων έλξης με τη λειτουργία INPUT_PULLUP.

Σύνταξη: pinMode(pin, mode)

```
Void setup () {
 pinMode (pin , mode);
                }
```

Παράμετροι:

pin: η θύρα Arduino: αριθμός θύρας που θα οριστεί.

mode: INPUT, OUTPUT, or INPUT_PULLUP.

Παράδειγμα:
pinMode(13, OUTPUT); // ορίζει τη ψηφιακή θύρα 13 σαν output
pinMode(5, INPUT); // ορίζει τη ψηφιακή θύρα 5 σαν input

digitalWrite() Function:

Η συνάρτηση digitalWrite () χρησιμοποιείται για την εγγραφή τιμής HIGH ή LOW σε ψηφιακή θύρα. Εάν η θύρα έχει διαμορφωθεί ως OUTPUT με pinMode (), η τάση της θα οριστεί στην αντίστοιχη τιμή: 5V για HIGH, 0V (γείωση) για LOW. Εάν η θύρα έχει ρυθμιστεί ως INPUT, το digitalWrite () θα ενεργοποιήσει (HIGH) ή θα απενεργοποιήσει (LOW) την εσωτερική αντίσταση. Συνιστάται να ορίσετε το pinMode () σε INPUT_PULLUP για να ενεργοποιήσετε την εσωτερική αντίσταση έλξης.

Εάν δεν ορίσετε το pinMode () σε OUTPUT και συνδέσετε μια λυχνία LED σε μια θύρα, όταν καλείτε το digitalWrite (HIGH), η λυχνία LED μπορεί να φαίνεται αμυδρά. Χωρίς ρητή ρύθμιση του pinMode (), το digitalWrite () θα έχει ενεργοποιήσει την εσωτερική αντίσταση έλξης, η οποία λειτουργεί σαν μια μεγάλη αντίσταση περιορισμού ρεύματος.

Σύνταξη:

digitalWrite (pin ,value);

pin: αριθμός θύρας που θα οριστεί

value: HIGH (1), or LOW(0).

Παράδειγμα:

digitalWrite(LED, HIGH); // άναμμα led
digitalWrite(LED, LOW); // σβήσιμο led

digitalRead() Function:
Η συνάρτηση digitalRead () διαβάζει την τιμή από ένα ορισμένη ψηφιακή θύρα HIGH ή LOW.

Σύνταξη:
digitalRead(pin)
pin: ο αριθμός της θύρας Arduino που θέλετε να διαβάσετε.

Παράδειγμα:
val = digitalRead(inPin); // read the input pin

Παρατήρηση: Οι αναλογικές θύρες μπορούν να οριστού ως ψηφιακές, με ονομασία A0, A1, κλπ.

delay() Function:
Η συνάρτηση delay () θέτει σε παύση το πρόγραμμα για το χρονικό διάστημα (σε χιλιοστά του δευτερολέπτου) που καθορίζεται ως παράμετρος. (Ένα δευτερόλεπτο έχει 1000 χιλιοστά)

Σύνταξη: delay(ms):

ms: αριθμός milliseconds για παύση. Επιτρεπόμενοι τύποι τιμών: unsigned long.

Παράδειγμα:

```
delay(1000);                    // waits for 1 second
delay(2000);                    // waits for 2 seconds
```

analogWrite() Function:
Η συνάρτηση analogWrite () εγγράφει μια αναλογική τιμή (κύμα PWM) σε μια θύρα. Μπορεί να χρησιμοποιηθεί για να ανάψει ένα LED σε διάφορα επίπεδα φωτεινότητας ή να οδηγήσει έναν κινητήρα σε διάφορες ταχύτητες. Μετά από μια κλήση της analogWrite (), η θύρα θα δημιουργήσει ένα σταθερό ορθογώνιο κύμα του καθορισμένου κύκλου μέχρι την επόμενη κλήση στο analogWrite () (ή μια κλήση προς digitalRead () ή digitalWrite ()) στην ίδια θύρα.

Παράδειγμα:

Ρυθμίζει την έξοδο στο LED ανάλογα με την τιμή που διαβάζεται από το ποτενσιόμετρο.

```
val = analogRead(analogPin);   // διάβασμα της θύρας input
 analogWrite(ledPin, val / 4);   // οι τιμές της analogRead values είναι από  0 έως 1023,
analogWrite δέχεται τιμές από 0 ως 255
```

analogRead( ) Function

Το Arduino είναι σε θέση να ανιχνεύσει εάν εφαρμόζεται τάση σε κάποια θύρα του και να το αναφέρει μέσω της λειτουργίας digitalRead (). Υπάρχει διαφορά μεταξύ ενός αισθητήρα on/off (ο οποίος ανιχνεύει την παρουσία ενός αντικειμένου) και ενός αναλογικού αισθητήρα, του οποίου η τιμή αλλάζει συνεχώς. Για να διαβάσουμε αυτόν τον τύπο αισθητήρα, χρειαζόμαστε μια θύρα διαφορετικού τύπου.

Στο κάτω δεξί μέρος της πλακέτας Arduino, θα δείτε έξι θύρες με την ένδειξη "Analog In". Αυτές οι ειδικές θύρες όχι μόνο επιστρέφουν αν εφαρμόζεται τάση σε αυτές, αλλά και την τιμή της. Χρησιμοποιώντας τη λειτουργία analogRead (), μπορούμε να διαβάσουμε την τάση που εφαρμόζεται σε μία από τις θύρες αυτές.

Αυτή η συνάρτηση επιστρέφει έναν αριθμό μεταξύ 0 και 1023, ο οποίος αντιπροσωπεύει τάσεις μεταξύ 0 και 5 βολτ. Για παράδειγμα, εάν υπάρχει τάση 2,5 V στον αριθμό pin 0, το analogRead (0) επιστρέφει 512.

Σύνταξη: analogRead(pin);

pin: Ο αριθμός της αναλογικής θύρας input με τιμές από 0 έως 5.

Παράδειγμα:

```
val = analogRead(analogPin);        // διάβασμα της τιμής input
Serial.println(val);                // εμφάνιση της τιμής
```

if Function:

Η εντολή if ελέγχει μια συνθήκη και εκτελεί την ακόλουθη πρόταση ή ένα σύνολο δηλώσεων εάν η συνθήκη είναι «αληθής».

Σύνταξη:

```
if (condition) {
//statement(s)
}
```

συνθήκη: μια λογική (boolean) έκφραση (π.χ, μπορεί να είναι true ή false).

Παράδειγμα: (Οι αγκύλες μπορεί να παραλειφθούν μετά από μια πρόταση if. Εάν γίνει αυτό, η εκτελείται ανάλογα με τη συνθήκη μόνο η επόμενη γραμμή (ορίζεται με το ερωτηματικό))

```
if (x > 120) digitalWrite(LEDpin, HIGH);

if (x > 120)
digitalWrite(LEDpin, HIGH);

if (x > 120) {digitalWrite(LEDpin, HIGH);}

if (x > 120) {
  digitalWrite(LEDpin1, HIGH);
  digitalWrite(LEDpin2, HIGH);
}
          // όλες είναι ορθές
```

if-else Εντολή:

Η εντολή if… else επιτρέπει μεγαλύτερο έλεγχο της ροής του κώδικα από τη βασική δήλωση if, επιτρέποντας την ομαδοποίηση πολλών συνθηκών. Η συνθήκη else (εάν υπάρχει) θα εκτελεστεί εάν η συνθήκη στην εντολή if έχει αποτέλεσμα false. Η else μπορεί να περιέχει και άλλη

συνθήκη if, έτσι ώστε να εκτελούνται ταυτόχρονα πολλαπλές, αμοιβαίως αποκλειόμενες συνθήκες.

Κάθε έλεγχος συνθήκες θα προχωρήσει στην επόμενη έως ότου βρεθεί μια συνθήκη με τιμή true. Όταν βρεθεί μια συνθήκη με τιμή true, εκτελείται το σχετικό μπλοκ εντολών και το πρόγραμμα στη συνέχεια μεταβαίνει στη γραμμή που είναι μετά την δομή if/else. Εάν καμία συνθήκη δεν έχει τιμή true, εκτελείται το μπλοκ else, εάν υπάρχει, και ορίζει την προεπιλεγμένη συμπεριφορά. Επιτρέπονται απεριόριστος αριθμός φωλιασμένων if/else.

Σύνταξη:

```
if (condition is TRUE) {
  // do Thing A
}
else
  //if NOT,  do Thing B
}
```

```
if (condition1) {
  // do Thing A
}
else if (condition2) {
  // do Thing B
}
else {
  // do Thing C
}
```

Παράδειγμα: (Παρακάτω είναι ένα απόσπασμα κώδικα για αισθητήρες θερμοκρασίας.)

```
if (temperature >= 70) {
  // Danger! Shut down the system.
}
else if (temperature >= 60) { // 60 <= temperature < 70
  // Warning! User attention required.
}
else { // temperature < 60
  // Safe! Continue usual tasks.
}
```

for Εντολή:

Η εντολή for χρησιμοποιείται για την επανάληψη ενός μπλοκ εντολών που περικλείονται σε αγκύλες. Ένας μετρητής χρησιμοποιείται συνήθως για να τερματίσει τον βρόχο. Η εντολή for είναι χρήσιμη για κάθε επαναλαμβανόμενη λειτουργία και συχνά χρησιμοποιείται σε συνδυασμό με πίνακες για τη λειτουργία συλλογών δεδομένων/θυρών.

Σύνταξη:

```
for (initialization; condition; increment) {
  // statement(s);
}
```

Παράμετροι:

αρχικοποίηση: συμβαίνει στην αρχή και μόνο μία φορά.

συνθήκη: κάθε φορά μέσω του βρόχου, η κατάσταση ελέγχεται. αν είναι αληθής, το μπλοκ εντολών και η αύξηση του μετρητή εκτελείται, και κατόπιν η συνθήκη δοκιμάζεται ξανά. Όταν η συνθήκη γίνει ψευδής, ο βρόχος τελειώνει.

προσαύξηση: εκτελείται κάθε φορά μέσω του βρόχου όταν η συνθήκη είναι αληθής.

Παράδειγμα**:**

```
for (int i = 0; i <= 255; i++)            {
   analogWrite(PWMpin, i);                 }

for (int x = 2; x < 100; x = x * 1.5) {
        println(x);                          }

for (int i = 0; i > -1; i = i + x)   {
  analogWrite(PWMpin, i);        }
```

switch...case εντολή

Όπως και η εντολή **if**, η switch/ case ελέγχει τη ροή των προγραμμάτων επιτρέποντας στους προγραμματιστές να καθορίσουν διαφορετικό κώδικα που πρέπει να εκτελεστεί σε διάφορες συνθήκες. Συγκεκριμένα, η εντολή switch συγκρίνει την τιμή μιας μεταβλητής με τις τιμές που καθορίζονται στις εντολές case. Όταν βρεθεί μια εντολή case της οποίας η τιμή ταιριάζει με εκείνη της μεταβλητής, εκτελείται ο συγκεκριμένος κώδικας.

Η δεσμευμένη λέξη -κλειδί **break** διακόπτει την εκτέλεση από της εντολής switch και χρησιμοποιείται συνήθως στο τέλος κάθε περίπτωσης. Χωρίς thην εντολή break, η εντολή switc θα συνεχίσει να εκτελεί τις επόμενες εντολές ("σειριακά") έως ότου βρε μια εντολή break ή φτάσει στο τέλος της εντολής switch.

Σύνταξη:

```
switch (var) {
  case label1:
    // statements
    break;
  case label2:
    // statements
    break;
default:
    // statements
```

```
  break;
}
```

Κώδικας Παράδειγμα:

```
switch (var) {
  case 1:
    //do something when var equals 1
    break;
  case 2:
    //do something when var equals 2
    break;
  default:
    // if nothing else matches, do the default
    // default is optional
    break;
}
```

var: μια μεταβλητή για σύγκριση με τις τιμές. Επιτρεπόμενοι τύποι μεταβλητών: int, char.

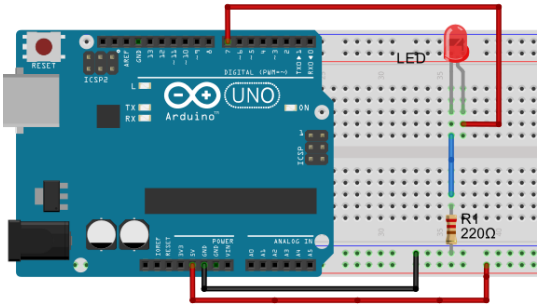
label1, label2: σταθερές. Επιτρεπόμενοι τύποι μεταβλητών: int, char.

Παρατήρηση:

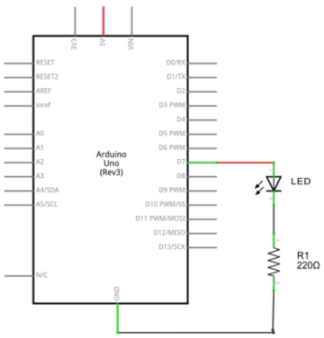
Τα κυκλώματα Arduino απεικονίζονται με δύο τρόπους;

1-) Όψη Breadboard

2-) Σχηματική όψη



1-) Όψη Breadboard



2-) Σχηματική όψη

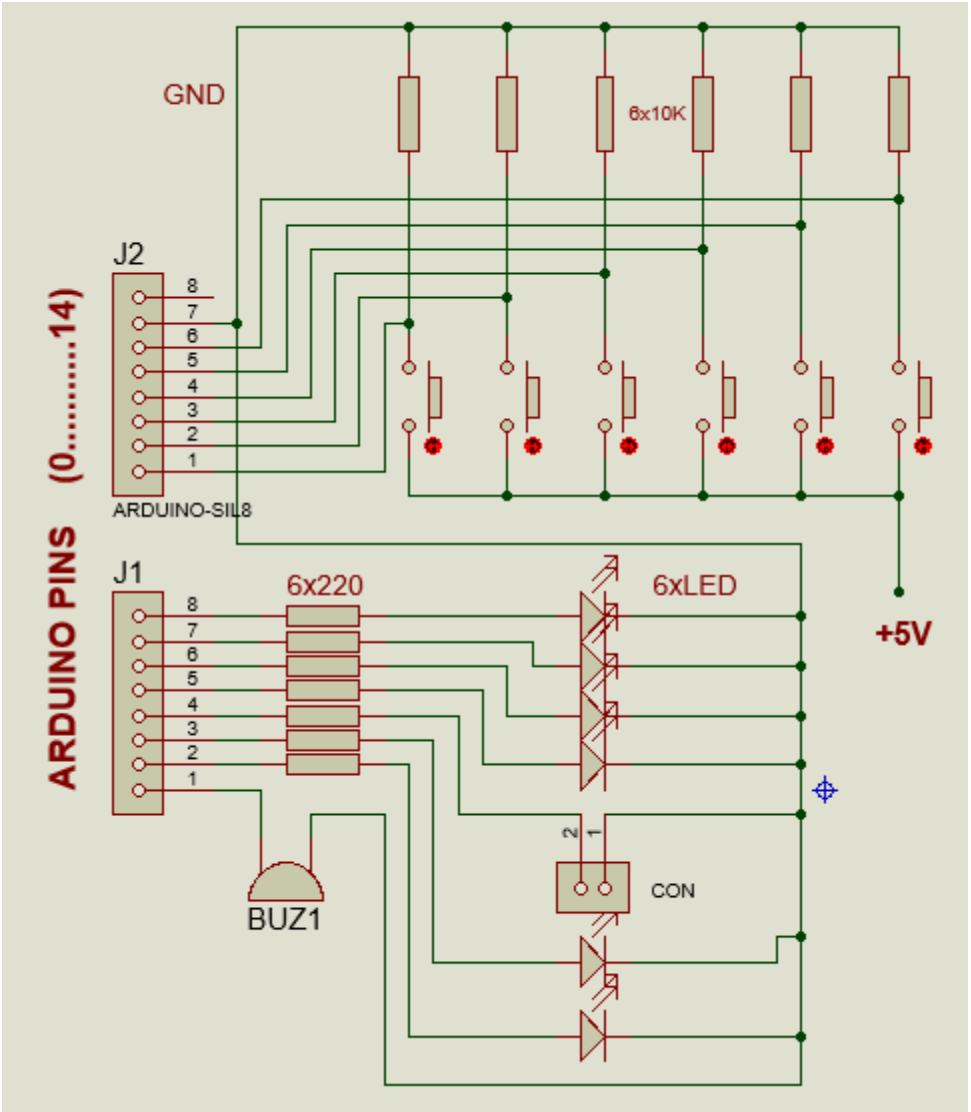
Θα χρησιμοποιήσουμε την όψη Breadboard καθώς είναι η πιο σύνηθης.

Αρκετά μιλήσαμε! Ας φτιάξουμε κάτι!

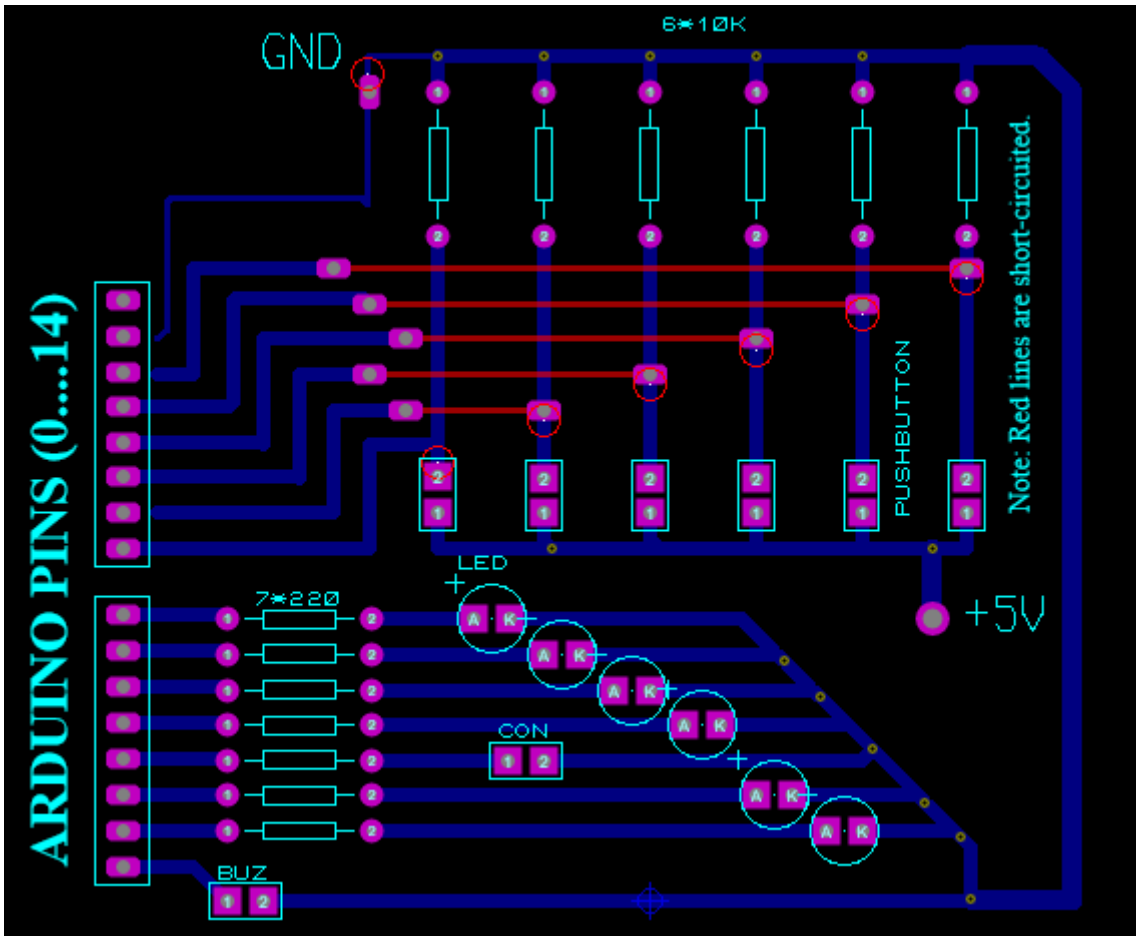
Κυκλώματα ενότητας κουμπιών και LED Arduino

Οι περισσότερες θύρες στο Arduino μπορούν να διαμορφωθούν ως είσοδος ή έξοδος. Η ενότητα το κιτ κουμπιών και LED Module είναι η πρώτη εκπαίδευση του κιτ του ARDUinVET για να βοηθήσει τους μαθητές στην εκμάθηση συστημάτων εισόδου/εξόδου του Arduino. Έτσι, μπορεί να ονομαστεί ως κιτ εκπαίδευσης I/O Arduino. Σε αυτό το εκπαιδευτικό κιτ, όπως φαίνεται παρακάτω, 6 κουμπιά συνδέονται στο Arduino ως είσοδοι. Και ένας βομβητής, ένας σύνδεσμος 2 pin και 6 LED είναι συνδεδεμένοι ως έξοδοι.

Οι μαθητές μπορούν να χρησιμοποιήσουν αυτό το κιτ εκπαίδευσης για να μάθουν συστήματα εισόδου/εξόδου του Arduino. Αυτό το εκπαιδευτικό κιτ μπορεί να τους κάνει να πειραματιστούν πιο εύκολα το κύκλωμα Arduino. Επίσης, μπορούν να χρησιμοποιήσουν ένα breadboard για να δοκιμάσουν κυκλώματα και πειράματα Arduino.



Σχήμα: Ανοικτό κύκλωμα κιτ κουμπιών και LED



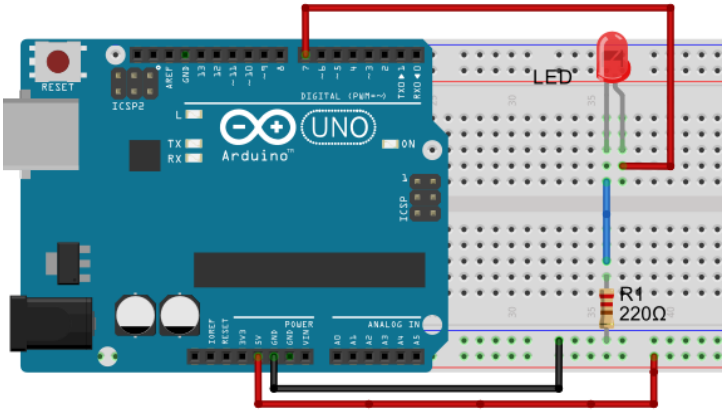
Σχήμα: PCB Σχήμα ανοικτού κύκλωματος κιτ κουμπιών και LED

Πρότυπα κυκλώματα και προγραμμάτα Arduio

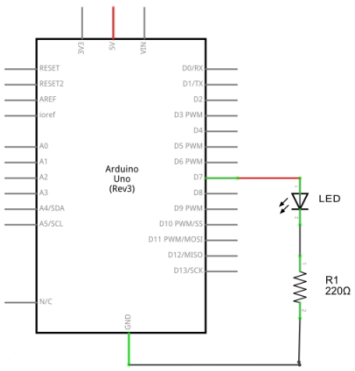
Κύκλωμα 1:

Τίτλος κυκλώματος: Πρόγραμμα αναβοσβήματος LED

Περιγραφή κυκλώματος: Ένα LED είναι συνδεδεμένη στην 13η ψηφιακή θύρα του Arduino. Η λυχνία LED αναβοσβήνει συνεχώς με διάστημα 1 δευτερολέπτου.



1-) Οψη Breadboard view



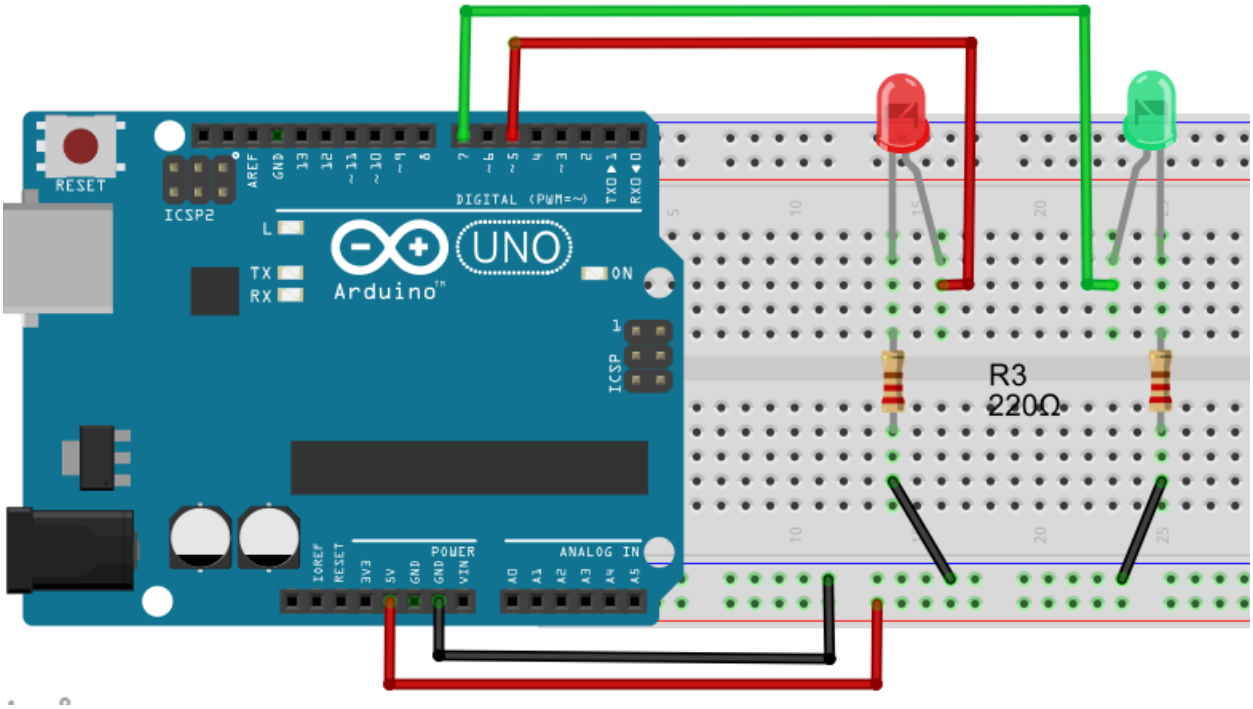
2-) Σχηματική όψη

```
/*    LED flashing Program, Switching a LED on and off    */

int led = 7;                    // integer variable led is declared

void setup() {                  // the setup() method is executed only once
pinMode(led, OUTPUT);           // the led PIN is declared as digital output
}

void loop() {                   // the loop() method is repeated
digitalWrite(led, HIGH);        // switching on the led
delay(1000);                    // stopping the program for 1000 milliseconds
digitalWrite(led, LOW);         // switching off the led
delay(1000);                    // stopping the program for 1000 milliseconds
 }
```

Κύκλωμα 2:

Τίτλος κυκλώματος: Flip-Flop , Πρόγραμμα αναβοσβήματος 2 LED

Περιγραφή κυκλώματος: 2 LED αναβοσβήνουν διαδοχικά με διάστημα 2 δευτερολέπτων.

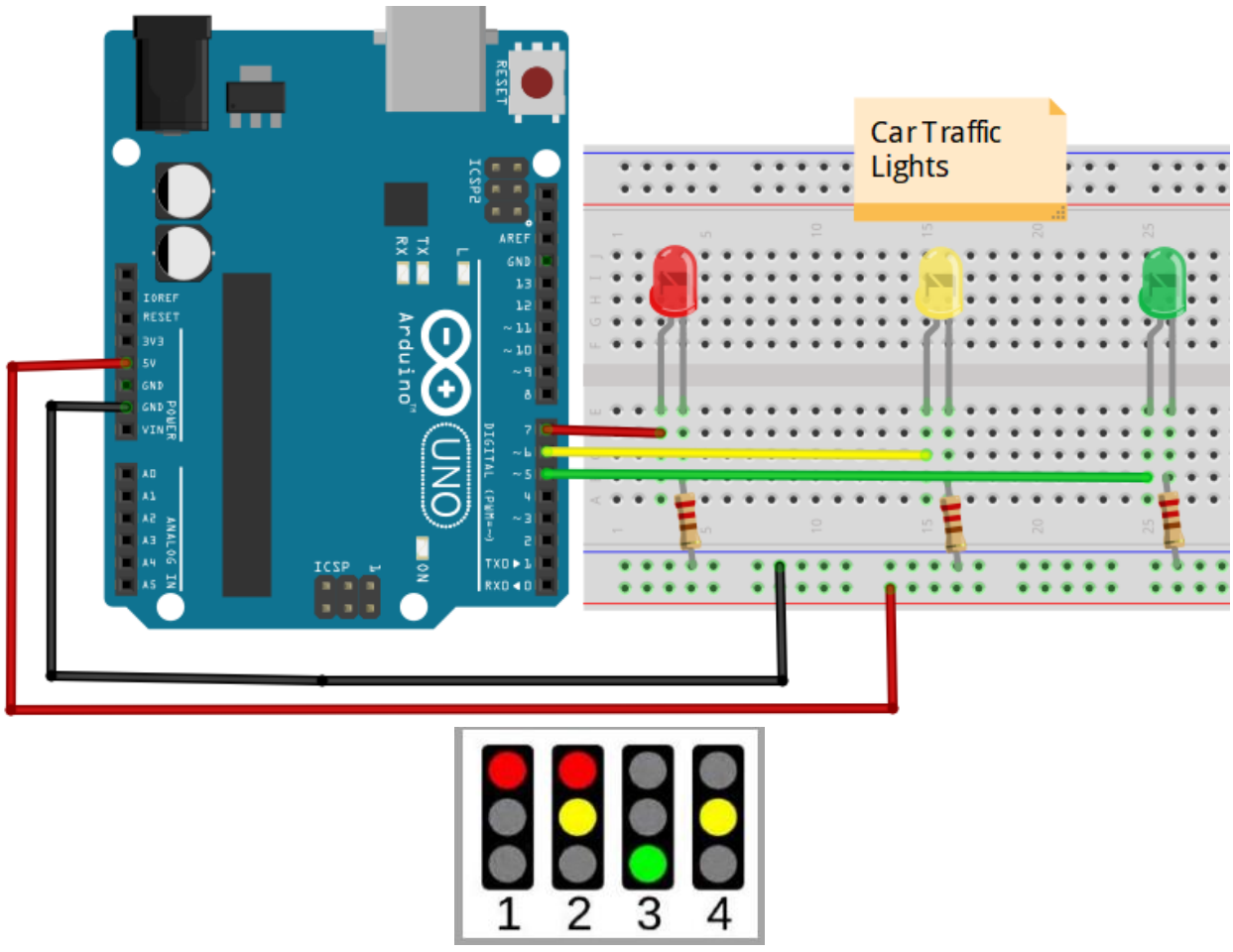



```
/*    Flip Flop     */

int greenLED=5;                    // Pin where the green LED is attached
int redLED=7;                      // Pin where the red LED is attached

void setup()  {
pinMode(greenLED, OUTPUT);         // green LED pin is initialised as OUTPUT
pinMode(redLED, OUTPUT);           // red LED pin is initialised as OUTPUT
                  }

void loop(){
digitalWrite(greenLED,  HIGH);          // switch on green LED
digitalWrite(redLED,  LOW);             // switch off red LED
pause(2000);

digitalWrite(greenLED,  LOW);           // switch off green LED
digitalWrite(redLED,  HIGH);            // switch on red LED
pause(2000);
                  }
```

Κύκλωμα 3:

Τίτλος κυκλώματος: Φανάρι

Περιγραφή κυκλώματος: Σε αυτό το έργο, πρόκειται να κατασκευάσουμε ένα σύστημα φαναριών. Υπάρχουν 3 LED με διαφορετικά χρώματα (πράσινο, κίτρινο και κόκκινο).



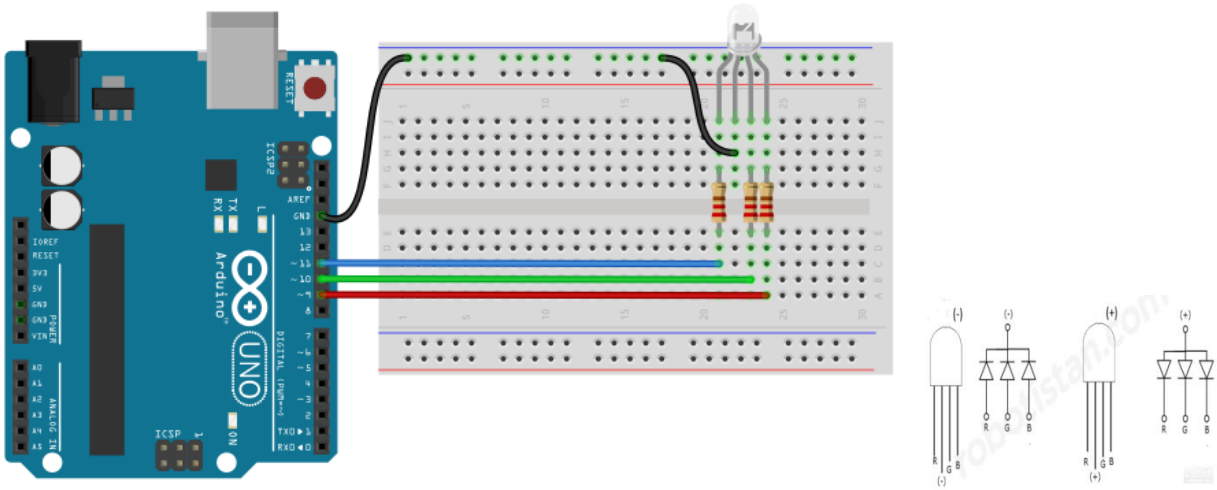

```
/*      Traffic Lights    */

int redLED = 7;
int yellowLED =6;
int greenLED = 5;

void setup() {                          //  here, we are initializing our pins as outputs
pinMode(redLED,  OUTPUT);
pinMode(yellowLED,  OUTPUT);
pinMode(greenLED,  OUTPUT);
}
```

```
void loop() {

digitalWrite(redLED,  HIGH);          // redLED is ON for 9 seconds
digitalWrite(yellowLED,  LOW);
digitalWrite(greenLED,  LOW);
delay(9000);

digitalWrite(redLED,  HIGH);          // redLED and  yellowLED are  ON for 2seconds
digitalWrite(yellowLED,  HIGH);
digitalWrite(greenLED,  LOW);
delay(2000);

digitalWrite(redLED,  LOW);           // greenLED is ON for 9 seconds
digitalWrite(yellowLED,  LOW);
digitalWrite(greenLED,  HIGH);
delay(9000);

digitalWrite(redLED,  LOW);           // Again, yellowLED is ON for 2seconds
digitalWrite(yellowLED,  HIGH);
digitalWrite(greenLED,  LOW);
delay(2000);

/* The loop starts again    */
 }
```

Κύκλωμα 4:

Τίτλος κυκλώματος: Εφαρμογή RGB LED

Περιγραφή κυκλώματος: Το RGB LED είναι τα 3 ενωμένα LED, τοποθετημένα σε μία μόνο θήκη. Είναι δυνατός ο ψηφιακός έλεγχος της έντασης του φωτός τριών χρωμάτων. Επιπλέον, τα χρώματα μπορούν να οριστούν χρησιμοποιώντας την τεχνική PWM.



/* Θα αναβοσβήνουμε κάθε χρώμα RGB LED για 1 δευτερολέπτο. Αν θέλουμε να εμφανίσουμε λευκό φως, πρέπει να ανάψουμε όλες τις λυχνίες LED ..*/

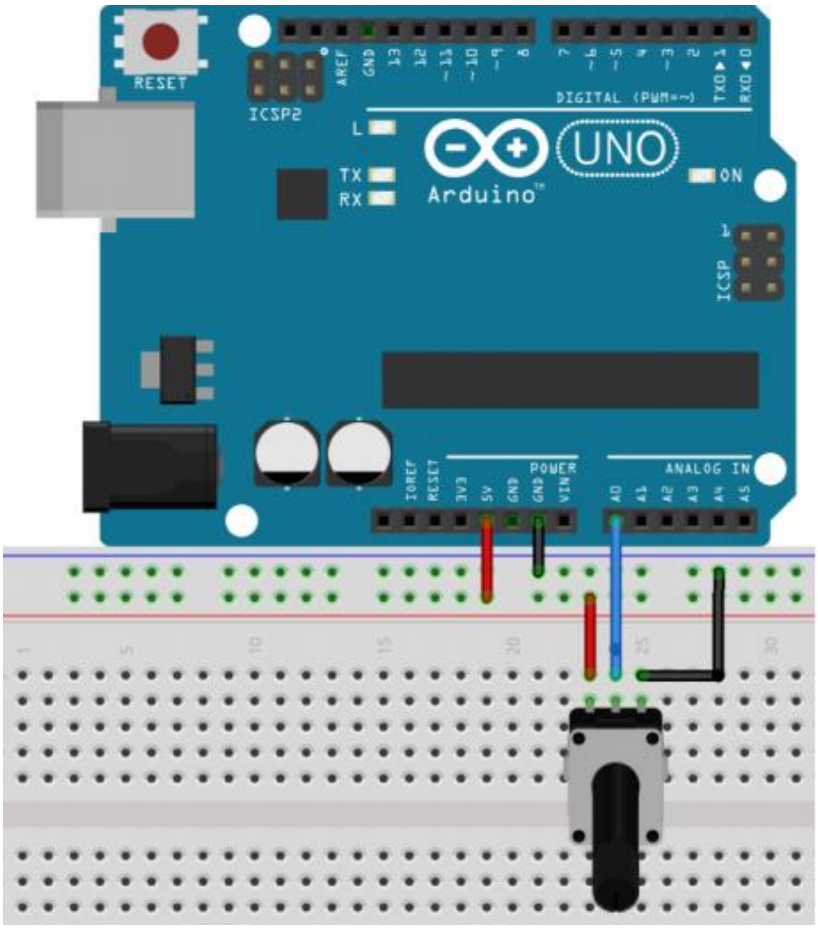
```
const int BlueLed=11;        // we connect the blue led to pin-11
const int GreenLed=10;       //  we connect the green led to pin-10
const int RedLed=9;          // we connect the red led to pin-11

        // We assign the pins to which the LEDs are connected as outputs.
void setup()  {
pinMode(BlueLed,OUTPUT);
pinMode(GreenLed,OUTPUT);
pinMode(RedLed,OUTPUT);     }

        //  The loop  starts here.

void loop() {
digitalWrite(BlueLed, LOW);        // RedLed is ON.
digitalWrite(GreenLed, LOW);
digitalWrite(RedLed, HIGH);
delay(1000);

 digitalWrite(BlueLed,  LOW );     // GreenLed is ON.
digitalWrite(GreenLed, HIGH);
digitalWrite(RedLed, LOW );
delay(1000);

 digitalWrite(BlueLed,  HIGH);          // BlueLed is ON.
digitalWrite(GreenLed,  LOW);
digitalWrite(RedLed, LOW);
delay(1000);

        // We display the white color by activating all the leds.
digitalWrite(BlueLed,  HIGH);
digitalWrite(GreenLed,  HIGH);
digitalWrite(RedLed,  HIGH);
delay(1000);
}
```

Κύκλωμα 5:

Τίτλος κυκλώματος: Ανάγνωση αναλογικής τάσης, ανάγνωση της τιμής από το ποτενσιόμετρο

Περιγραφή κυκλώματος: Εδώ μαθαίνουμε πώς να διαβάζουμε μια αναλογική είσοδο σε αναλογικό pin-0. Η είσοδος μετατρέπεται από analogRead () σε τάση και εκτυπώνεται στη σειριακή οθόνη του Arduino IDE.

Ποτενσιόμετρο: Ένα ποτενσιόμετρο είναι ένας απλός ηλεκτρομηχανικός μετατροπέας. Μετατρέπει την περιστροφική ή γραμμική κίνηση από τον χειριστή εισόδου σε αλλαγή αντίστασης. Αυτή η αλλαγή χρησιμοποιείται (ή μπορεί να χρησιμοποιηθεί) για τον έλεγχο οτιδήποτε από τον όγκο ενός συστήματος hi-fi έως την κατεύθυνση ενός τεράστιου πλοίου μεταφοράς εμπορευματοκιβωτίων.	



/* ReadAnalogVoltage: Διαβάζει μια αναλογική είσοδο στον ακροδέκτη 0, το μετατρέπει σε τάση και εκτυπώνει το αποτέλεσμα στη σειριακή οθόνη. Η γραφική αναπαράσταση είναι διαθέσιμη χρησιμοποιώντας σειριακό σχεδιαστή (μενού Tools > Serial Plotter). Συνδέστε τον κεντρικό ακροδέκτη ενός ποτενσιόμετρου στην θύρα A0 και τους εξωτερικούς ακροδέκτες στα +5V και τη γείωση. */

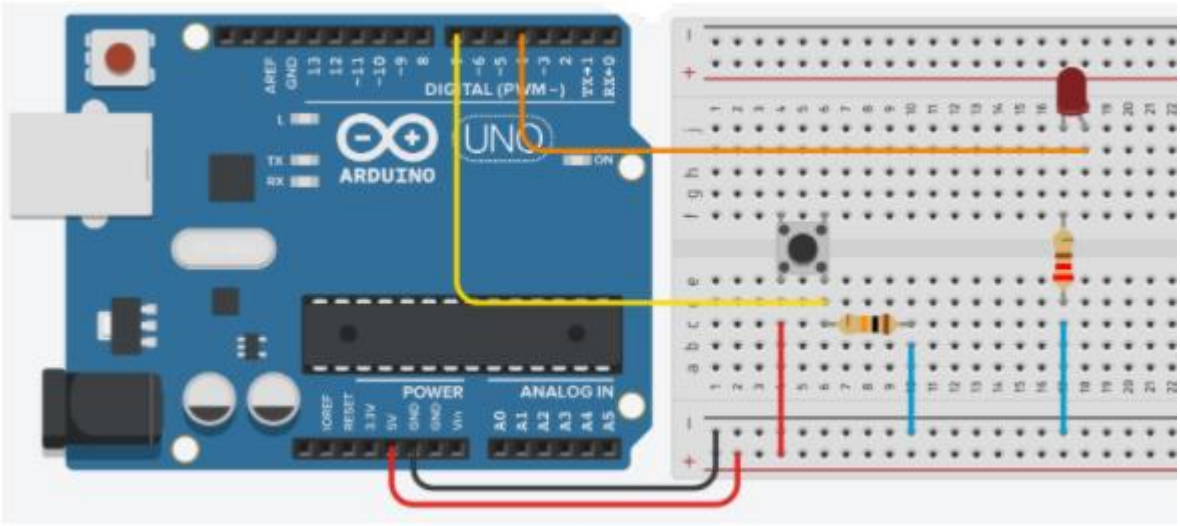
// the setup routine runs once when you press reset:

```
void setup()        {
        // initialize serial communication at 9600 bits per second:
  Serial.begin(9600); }
       // the loop routine runs over and over again forever:
void loop() {
        // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
        // print out the value you read:
  Serial.println(sensorValue);
  delay(1000);        // delay between reads for stability
}
```

Κύκλωμα 6:

Τίτλος κυκλώματος: Χρήση κουμπιών στο Arduino

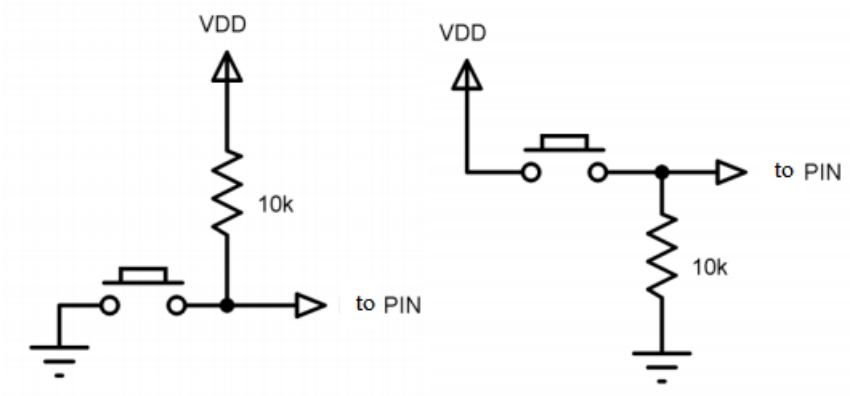
Περιγραφή κυκλώματος: Με το πάτημα ενός πιεζόμενου κουμπιού που είναι προσαρτημένο στη θύρα 7, το κουμπί ανάβει και σβήνει ένα LED, συνδεδεμένο με τον ψηφιακό ακροδέκτη 4.



/* Το κουμπί ανάβει και σβηνει ένα φωτάκι LED συνδεδεμένο στη ψηφιακή θύρα 4, όταν πιέζεται ένα κουμπί που είναι προσαρτημένο στη ψηφιακή θύρα 7. */

```
void setup()
{
  pinMode(4, OUTPUT);  // pin-4 is output
  pinMode(7, INPUT);     //  pin7 is input. 7
}
void loop()
{
   if (digitalRead(7) == HIGH)              //  If pin7 is high,
    digitalWrite(4, HIGH);                  // Led is ON,
    if (digitalRead(7) == LOW)              // If pin7 is low,
     digitalWrite(4, LOW);                  // Led is OFF.
}
```

ΠΑΡΑΤΗΡΗΣΗ: Η εντολή IF-THEN μπορεί επίσης να χρησιμοποιηθεί για τη σύνδεση κουμπιών σε ακροδέκτες εισόδου του Arduino. Αυτή η σύνδεση μπορεί να γίνει με δύο διαφορετικούς τρόπους. Σύνδεση Pull_Up και σύνδεση Pull-down.

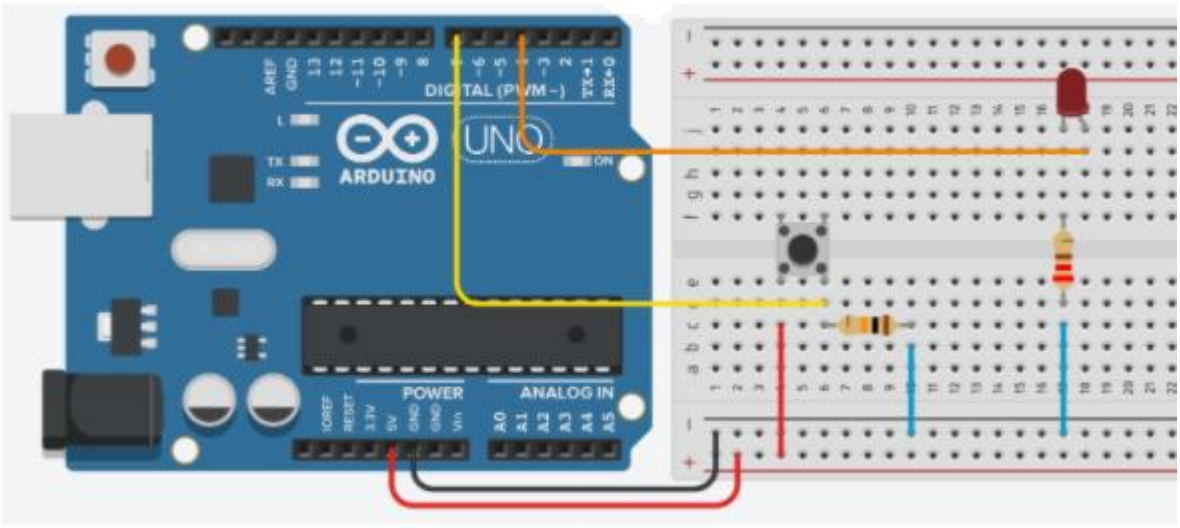


Σχήμα: Διακόπτες και κουμπιά που μπορούν να χρησημοποιηθούν με την εντολή IF…THEN

Κύκλωμα 7:

Τίτλος κυκλώματος: Χρήση εντολής **ELSE** στο Arduino

Περιγραφή κυκλώματος: Με το πάτημα ενός πιεζόμενου κουμπιού που είναι προσαρτημένο στη θύρα 7, το κουμπί ανάβει και σβήνει ένα LED, συνδεδεμένο με τον ψηφιακό ακροδέκτη 4. Επίσης, θα μάθουμε να καθορίζουμε τις θύρες μας με τη μεταβλητή "int".



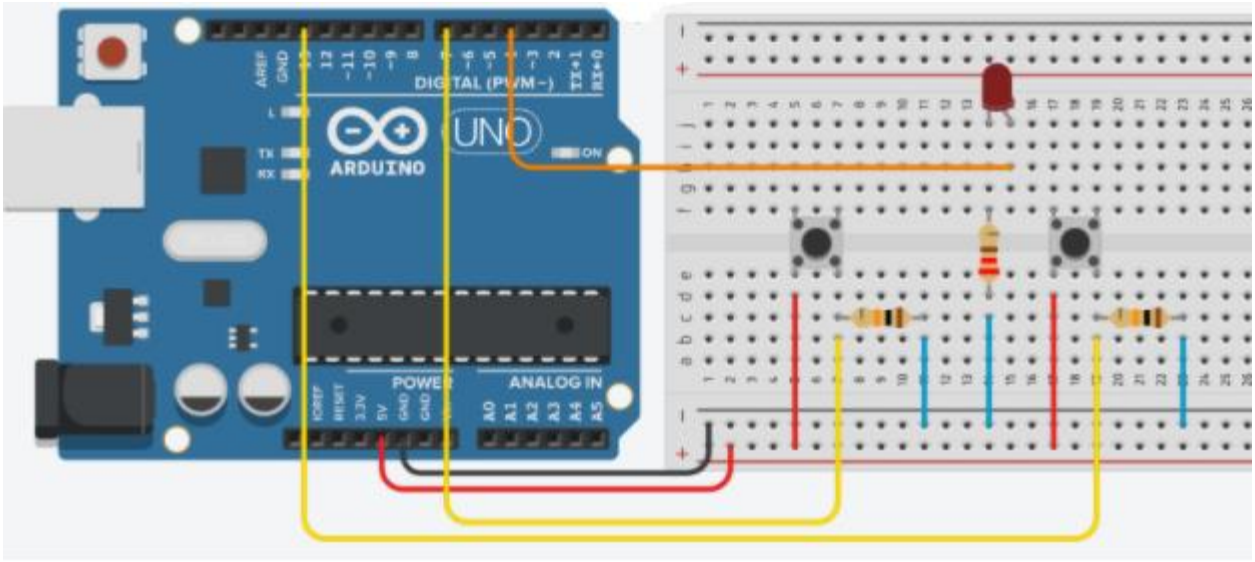
Σύνταξη:

```
 int led = 4;
int buton =7;
void setup()
{
  pinMode(led, OUTPUT);
  pinMode(buton, INPUT);
}
void loop()
{
    if (digitalRead(buton) == HIGH)          // The button is ON,
       digitalWrite(led, HIGH);              // Led is ON
     else                                    // If  not,
       digitalWrite(led, LOW);               // Led is OFF.
}
```

Κύκλωμα 8:

Τίτλος κυκλώματος: 2 κουμπιά με 1 LED

Περιγραφή κυκλώματος: Το ένα κουμπί ανάβει ένα LED, το άλλο κουμπί το σβήνει




```
/*      2 κουμπιά με 1 LED
Τα κουμπιά είναι συνδεδεμένα με αντιστάσεις 10K resistors σε σειρά.      */

int led = 4;                    // The pin-4 is assigned as "led"
int button1 = 7;                // The pin-7 is assigned as "button1"
int button2 = 13;               // The pin-13 is assigned as "button2"
void setup()
{
  pinMode(led, OUTPUT);                 // led=Output
  pinMode(button1, INPUT);              // button1=Input
  pinMode(button2, INPUT);              // button2=Input
}
void loop()
{
  if (digitalRead(button1) == HIGH)     // IF "button1" is ON (active),
     digitalWrite(led, HIGH);           // Led is ON.
   if (digitalRead(button2) == HIGH)    // IF "button2" is ON (active),
     digitalWrite(led, LOW);            // Led is OFF.
}
```

Χρήση του ARDUINO SERIAL MONITOR

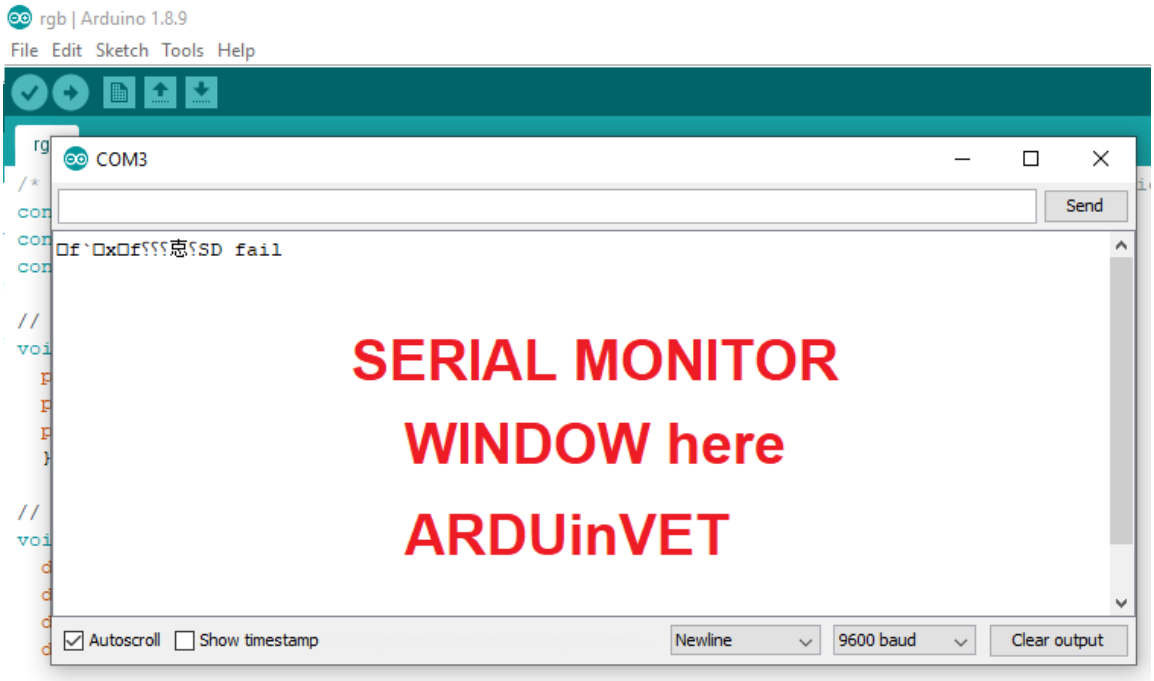
Το Arduino IDE διαθέτει μια δυνατότητα που μπορεί να είναι μια μεγάλη βοήθεια για τον εντοπισμό σφαλμάτων ή τον έλεγχο του Arduino από το πληκτρολόγιο του υπολογιστή σας.

Το Serial Monitor είναι ένα ξεχωριστό αναδυόμενο παράθυρο που λειτουργεί ως ξεχωριστό τερματικό που επικοινωνεί με τη λήψη και την αποστολή σειριακών δεδομένων. Δείτε το εικονίδιο στην άκρη δεξιά της εικόνας εδώ. 

Τα σειριακά δεδομένα αποστέλλονται μέσω ενός καλωδίου (στην περίπτωσή μας, συνήθως μέσω σύνδεσης USB) και αποτελούνται από μια ακολουθία bit 1 και 0 που αποστέλλονται μέσω του καλωδίου. Τα δεδομένα μπορούν να σταλούν και προς τις δύο κατευθύνσεις (Στην περίπτωσή μας μέσω δύο καλωδίων).

Θα χρησιμοποιήσετε τη Σειριακή οθόνη για την εκσφαλμάτωση σεναρίων κώδικα Arduino ή για να δείτε δεδομένα που αποστέλλονται από ένα σενάριο. Πρέπει να έχετε ένα Arduino συνδεδεμένο μέσω USB στον υπολογιστή σας για να μπορείτε να ενεργοποιήσετε τη σειριακή οθόνη (Serial Monitor).

Ανοίξτε το Serial Monitor κάνοντας κλικ στο κουμπί Serial Monitor στο IDE. Θα πρέπει να μοιάζει με το παρακάτω στιγμιότυπο οθόνης. Βεβαιωθείτε ότι η baud (ταχύτητα) έχει οριστεί σε 9600. Βρίσκεται στην κάτω δεξιά γωνία. (Ειναι σημαντικό να έχει οριστεί το ίδιο στο πρόγραμμά μας και εδώ. Δεδομένου ότι η προεπιλογή εδώ είναι 9600, ορίζουμε τη ταχύτητα.



Κύριες εντολές για χρήση σειριακής οθόνης

Serial.begin(); Ορίζει το ρυθμό δεδομένων σε bits ανά δευτερόλεπτο (baud) για σειριακή μετάδοση δεδομένων. Για την επικοινωνία με το Serial Monitor, βεβαιωθείτε ότι χρησιμοποιείτε έναν από τους ρυθμούς baud που αναφέρονται στο μενού στην κάτω δεξιά γωνία της οθόνης του. Μπορείτε, ωστόσο, να καθορίσετε άλλους ρυθμούς - για παράδειγμα, για επικοινωνία μέσω των θυρών 0 και 1 με ένα εξάρτημα που απαιτεί συγκεκριμένο ρυθμό baud.

Σύνταξη: Serial.begin(speed);
Παράδειγμα: Serial.begin(9600);

Serial.print(); Εκτυπώνει δεδομένα στη σειριακή θύρα σε αναγνώσιμη μορφή ASCII. Αυτή η εντολή μπορεί να λάβει πολλές μορφές. Οι αριθμοί εκτυπώνονται χρησιμοποιώντας έναν χαρακτήρα ASCII για κάθε ψηφίο. Οι αριθμοί float εκτυπώνονται ομοίως ως ψηφία ASCII, προεπιλεγμένα με δύο δεκαδικά ψηφία. Τα bytes αποστέλλονται ως ένας μόνο χαρακτήρας. Οι χαρακτήρες και οι συμβολοσειρές αποστέλλονται ως έχουν. Για παράδειγμα:

Serial.print(78;) επιστρέφει"78"

Serial.print('N'); επιστρέφει "N"

Serial.print("Hello ArduinVet."); επιστρέφει "Hello ArduinVet."

Serial.println(); Εκτυπώνει δεδομένα στη σειριακή θύρα σε αναγνώσιμη μορφή ASCII ακολουθούμενα από χαρακτήρα επιστροφής (ASCII 13, ή '\ r') και χαρακτήρα νέας γραμμής (ASCII 10, ή '\ n'). Αυτή η εντολή παίρνει τις ίδιες μορφές με το Serial.print ().

Serial.println(val);

Serial.println(val, format);

Serial.printle(13, BIN); επιστρέφει "1101", δυαδική τιμή του 15 στο Serial Monitor.

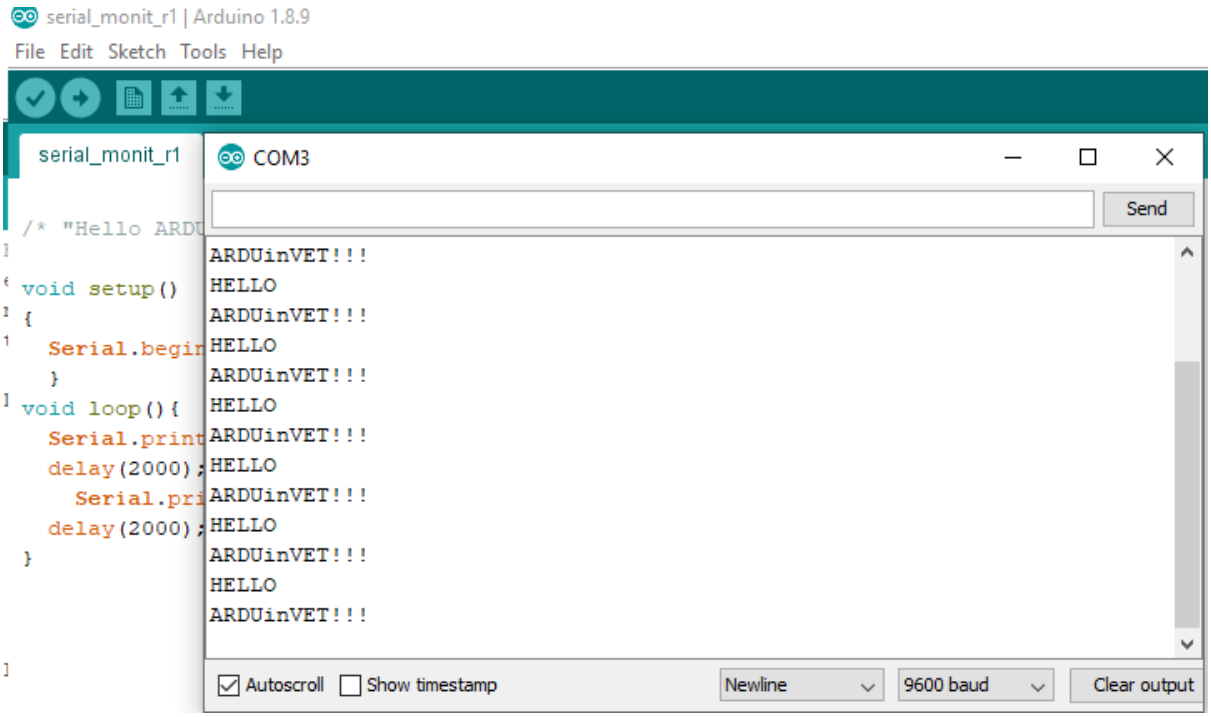
ΠΑΡΑΤΗΡΗΣΗ: Η μόνη διαφορά μεταξύ Serial.print και Serial.println είναι ότι το Serial.println σημαίνει ότι το επόμενο πράγμα που θα αποσταλλεί στη σειριακή θύρα θα ξεκινήσει στην επόμενη γραμμή. Υπάρχει ένα τρίτο νέο πράγμα που ίσως έχετε παρατηρήσει. Υπάρχει κάτι σε εισαγωγικά ("). Αυτό ονομάζεται συμβολοσειρά (string).

Κύκλωμα 9:

Τίτλος κυκλώματος: Εκτύπωση "Hello ARDUinVET" στο Serial Monitor

Περιγραφή κυκλώματος: Θα εμφανιστεί το μήνυμα "Hello ARDUinVET" στο Serial Monitor

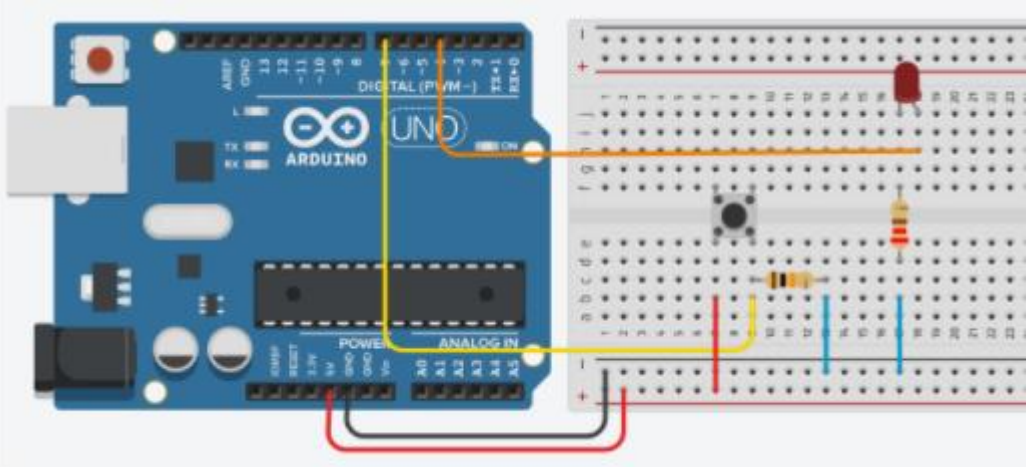
```
/*      Εκτύπωση "Hello ARDUinVET" στο Serial Monitor */
void setup()
{
  Serial.begin(9600);     // Serial communication speed
}
void loop()
{
  Serial.println("  HELLO ");
  delay(2000);
    Serial.println("ARDUinVET!!!");
  delay(2000);
}
```



Κύκλωμα 10:

Τίτλος κυκλώματος: Εκτύπωση κατάστασης κουμπιού στο Serial Monitor

Περιγραφή κυκλώματος: Αν είναι πατημένο το κουμπί, θα ανάβει το Led και θα εμφανίζεται στη σειριακή οθόνη το μήνυμα "Led is ligthing, LED = ON". Αν το κουμπί δεν είναι πατημένο, το Led θα είναι σβηστό και θα εμφανίζεται στη σειριακή οθόνη το μήνυμα ("Button is not pressed ")




```
/*      Εκτύπωση κατάστασης κουμπιού στο Serial Monitor */

void setup()                    {
 pinMode(4, OUTPUT);
 pinMode(7, INPUT);
   Serial.begin(9600);          }
void loop()                      {
 if (digitalRead(7) == HIGH)          // Read the digital signal on Pin-7
   {
   digitalWrite(4, HIGH);
   Serial.println("Led is ligthing, LED=ON");
   delay(250);
   }
  else                          // If the previous condition is not  met.
   {
   digitalWrite(4, LOW);
   Serial.println("Button is not pressed");
```

```
  delay(1000);
  }
}                          //The view of the serial monitor is seen below.
```

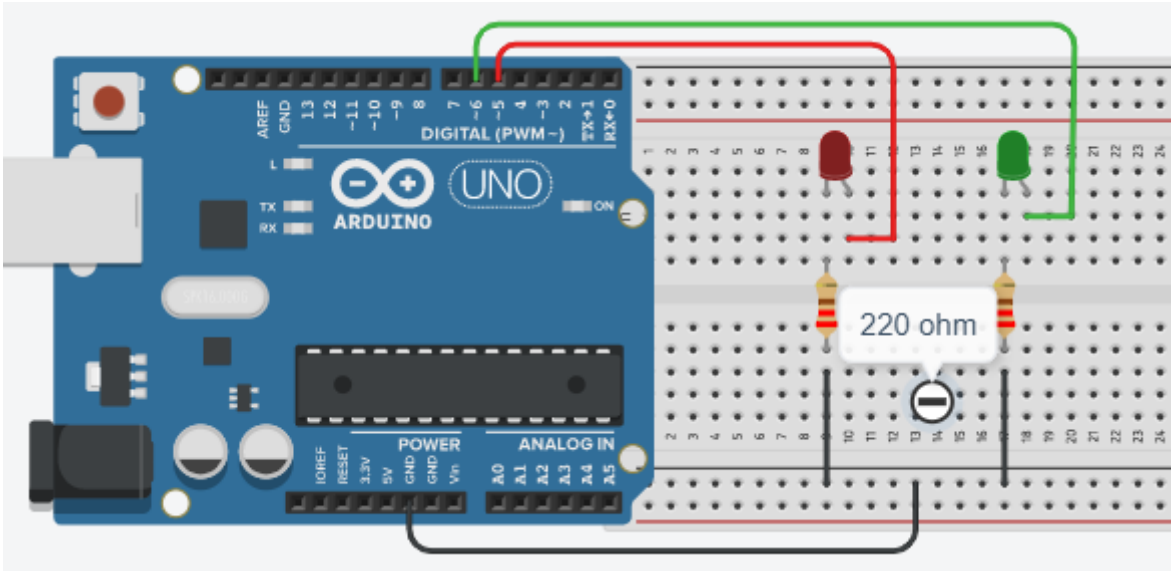
```
Button is not pressed
Button is not pressed
Led is ligthing, LED=ON
Led is ligthing, LED=ON
Led is ligthing, LED=ON
Led is ligthing, LED=ON
Led is ligthing, LED=ON
Button is not pressed
```

Κύκλωμα 11:

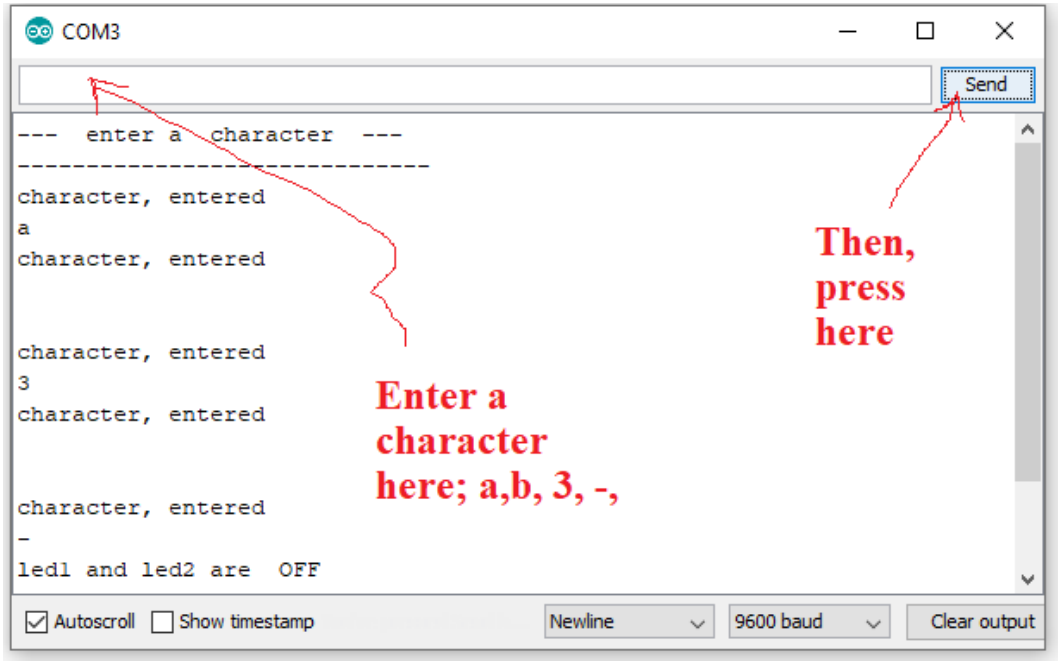
Τίτλος κυκλώματος: Αποστολή δεδομένων από το Serial Monitor

Περιγραφή κυκλώματος: Εάν πατηθεί ο χαρακτήρας "A" στο πληκτρολόγιο, το led1 είναι ON. Εάν πατηθεί ο χαρακτήρας "3" στο πληκτρολόγιο, το led2 είναι ON.

Εάν πατηθεί ο χαρακτήρας "-" στο πληκτρολόγιο, το led1 και το led2 είναι OFF.



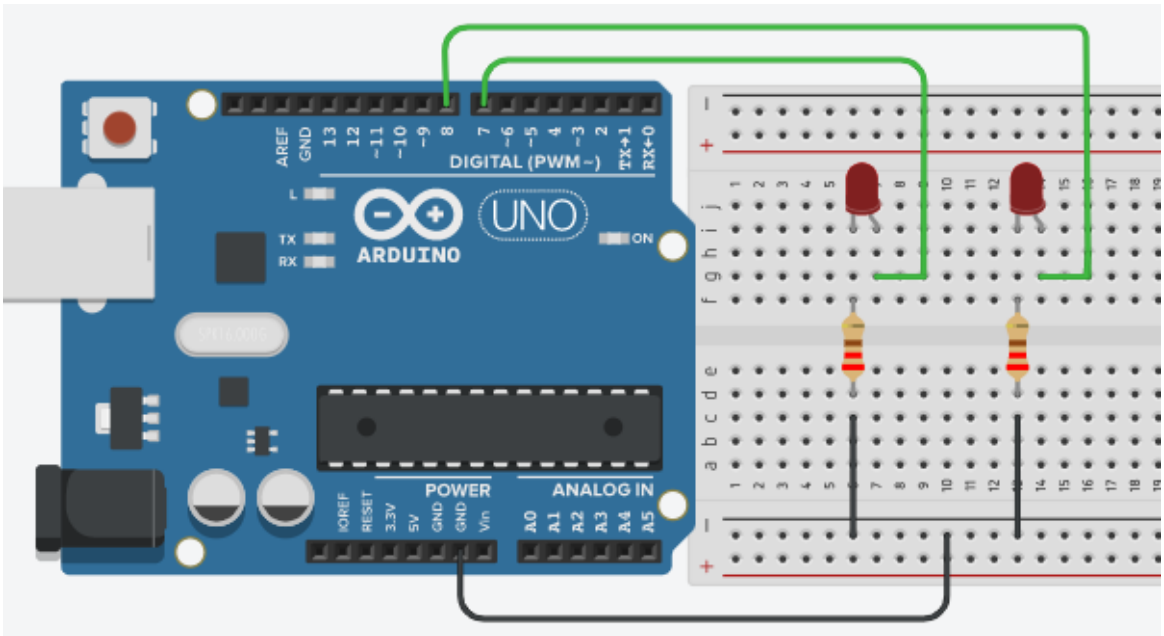
```
/*     Αποστολή δεδομένων από το Serial Monitor */

char character;
#define  led1 5
#define  led2 6

 void setup()   {
pinMode(led1, OUTPUT);
pinMode(led2, OUTPUT);
Serial.begin(9600);
Serial.println ("---  enter a  character  --- ");
Serial.println ("-----------------------------");
}

void loop()
{                                                                    "
if (Serial.available()>0)
{
character=Serial.read();
Serial.println ("character, entered ");
Serial.println (character);

 if (character=='A') digitalWrite (led1, 1);

if (character=='a') digitalWrite (led1, 1);

if (character=='3') digitalWrite (led2, 1);

if (character=='-')
{
digitalWrite(led1,0);
digitalWrite(led2,0);
Serial.println ("led1 and led2 are  OFF ");
}
}
}
```



Κύκλωμα 12:

Τίτλος κυκλώματος: Χρήση εντολής FOR

Περιγραφή κυκλώματος:



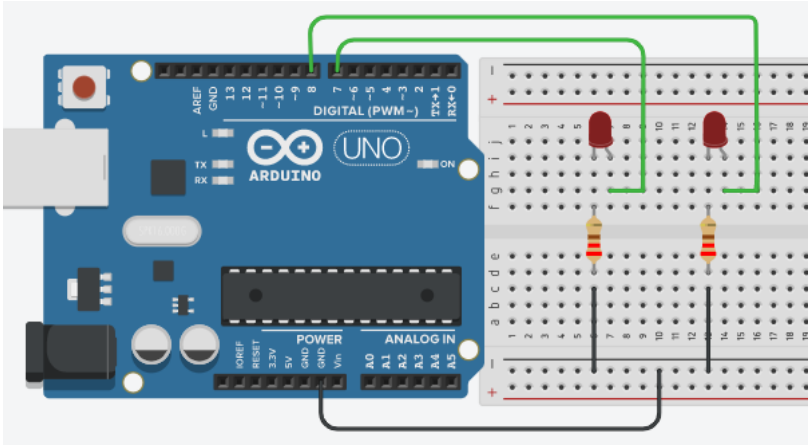
```
/*    "Χρήση εντολής  FOR ”   */

int   led1 = 7;
int   led2 = 8;

void setup() {
Serial.begin(9600);
pinMode(7,OUTPUT);
pinMode(8,OUTPUT);
}

void loop()
{
for(int i=1; i<6; i++)
{
Serial.println(i);
digitalWrite(led1, 1);
digitalWrite(led2, 1);
delay(1000);

 digitalWrite(led1, 0);
digitalWrite(led2, 0);
delay(1000);
}
}
```

Κύκλωμα 13:

Τίτλος κυκλώματος: Χρήση εντολής FOR μέσω Serial Monitor

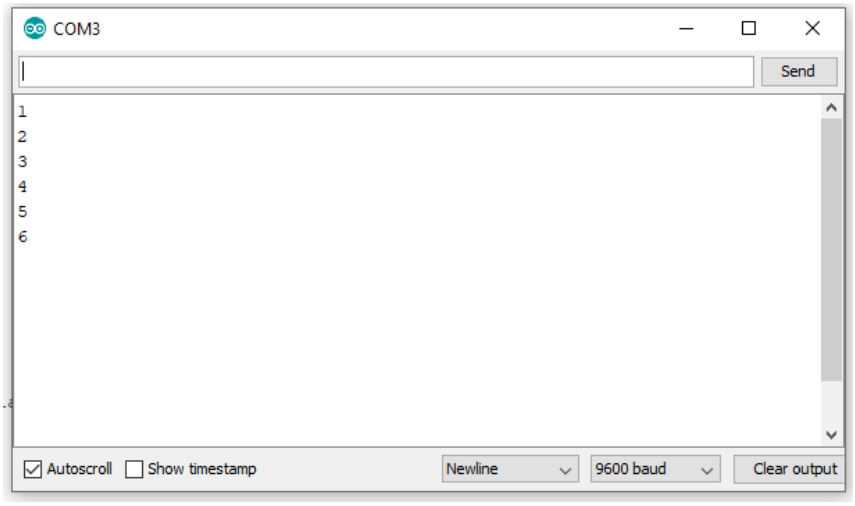
Περιγραφή κυκλώματος: Σε αυτήν την εφαρμογή, τα LED θα αναβοσβήσουν 6 φορές. Οι αριθμοί 1, 2, 3, 4, 5,6 θα εμφανιστούν στη σειριακή οθόνη. Στη συνέχεια θα εισαχθεί στον βρόχο while βγαίνοντας από τον βρόχο for. Και το πρόγραμμα θα σταματήσει εδώ. Για επανεκκίνηση, πρέπει να πατήσετε το κουμπί επαναφοράς.

Εδώ, χρησιμοποιούμε το ίδιο κύκλωμα με το προηγούμενο.




```
/*     "Χρήση εντολής FOR μέσω Serial Monitor"   */

int   led1 = 7;
int   led2 = 8;

void setup()                    {
Serial.begin(9600);
pinMode(7,OUTPUT);
pinMode(8,OUTPUT);       }

void loop() {

  for(int i=1; i<=6; i++)                // the value of i = 1, 2,3, 4,5, 6

  {

 Serial.println(i);                      // Write the values of i, on the serial monitor
digitalWrite(led1, 1);
digitalWrite(led2, 1);
delay(1000);
digitalWrite(led1, 0);
digitalWrite(led2, 0);
delay(1000);
}
while(1);                  //  the for loop doesn't get into an infinite loop.

}
```

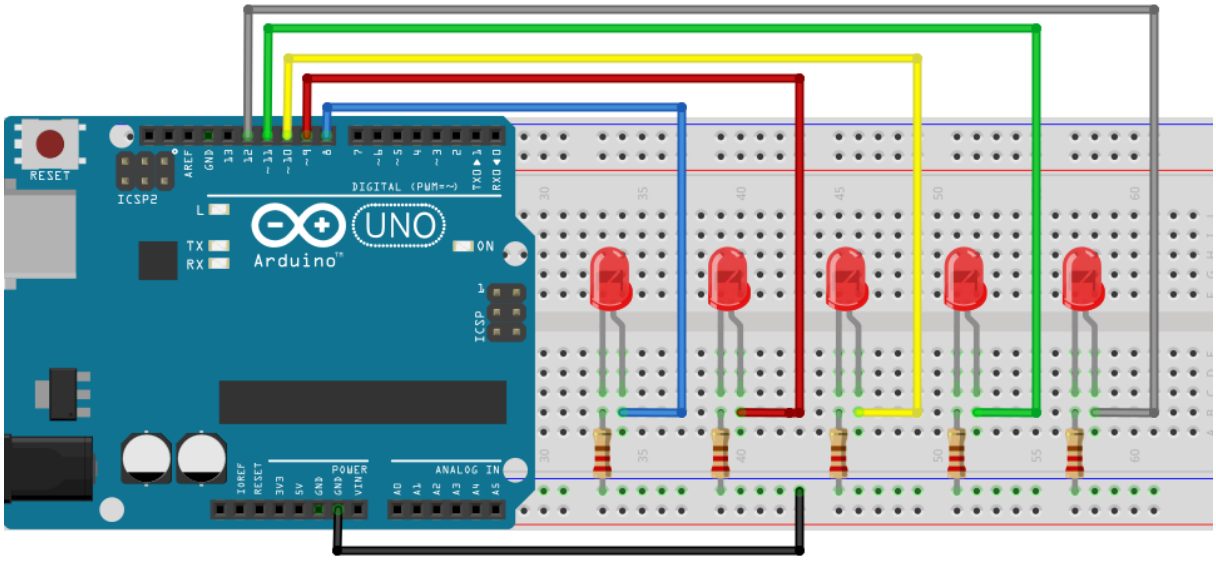
ΠΑΡΑΤΗΡΗΣΗ: Για επανεκκίνηση, πρέπει να πατηθεί το κουμπί επαναφοράς.



Κύκλωμα 14:

Τίτλος κυκλώματος: Knight Rider με 5 Led με χρήση εντολής FOR

Περιγραφή κυκλώματος:



/* Knight Rider με 5 Led με χρήση εντολής FOR */

```
void setup() {
pinMode(8, OUTPUT);
pinMode(9, OUTPUT);
pinMode(10, OUTPUT);
pinMode(11, OUTPUT);
pinMode(12, OUTPUT);    }
```

```
void loop() {

 for (int b=8; b<=12; b++)                    // Upcounter  starts here

  {

 digitalWrite(b, HIGH);
delay(150);
digitalWrite (b, LOW);
delay(150);

  }

  for (int b=12; b>=8; b--)                    // Downcounter  starts here

  {

 digitalWrite (b, HIGH);
delay(150);
digitalWrite (b, LOW);
delay(150);

  }
}
```

Κύκλωμα 15:

Τίτλος κυκλώματος: Παραγωγή τυχαίων χρωμάτων με led RGB, Χρήση της εντολής switch/case

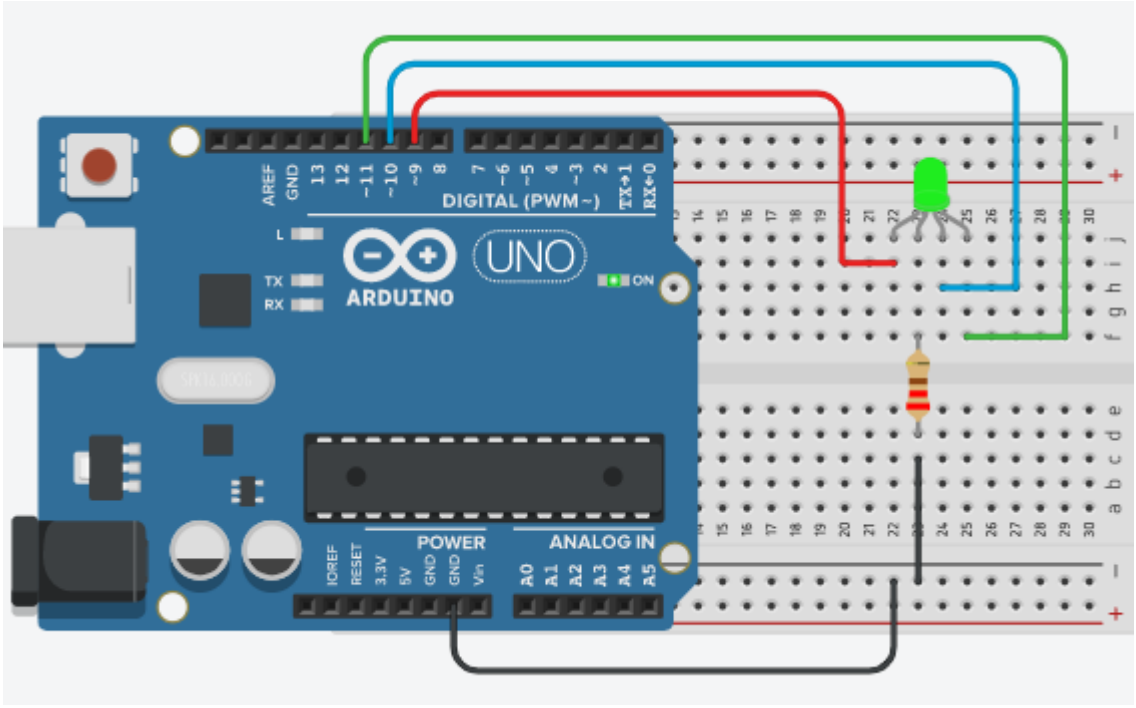
Περιγραφή κυκλώματος: Σε αυτήν την εφαρμογή χρησιμοποιούνται η συνάρτηση random και η εντολή Switch/Case. Σε αυτό το κύκλωμα, τα κόκκινα, πράσινα, μπλε χρώματα θα ληφθούν τυχαία.

ΠΑΡΑΤΗΡΗΣΗ:

random() : Η συνάρτηση random παράγει τυχαίους αριθμούς.
Σύνταξη: random(max), random(min, max)

min: κάτω όριο της τυχαίας τιμής, (προαιρετικό).

max: άνω όριο της τυχαίας τιμής.



/* Παραγωγή τυχαίων χρωμάτων με led RGB, Χρήση της εντολής switch/case */

```
#define R  9
#define G 10
#define B 11
int colour;
int dly=3000;

void setup() {
pinMode(R, OUTPUT);
pinMode(G, OUTPUT);
pinMode(B, OUTPUT);
Serial.begin(9600);
}

void loop()
{
colour=random(4);
Serial.println(colour);

switch(colour)  {

 case 0:
digitalWrite (R, 1);              //RedLED=ON
digitalWrite (G, 0);
digitalWrite (B, 0);
delay(dly);
break;
```

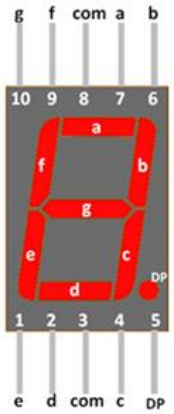
```
 case 1:
digitalWrite (R, 0);          //GreenLED=ON
digitalWrite (G, 1);
digitalWrite (B, 0);
delay(dly);
break;

 case 2:                      //BlueLED=ON
digitalWrite (R, 0);
digitalWrite (G, 0);
digitalWrite (B, 1);
delay(dly);
break;

 case 3:                      //No colour
digitalWrite (R, 0);
digitalWrite (G, 0);
digitalWrite (B, 0);
delay(dly);
break;
}
}
```

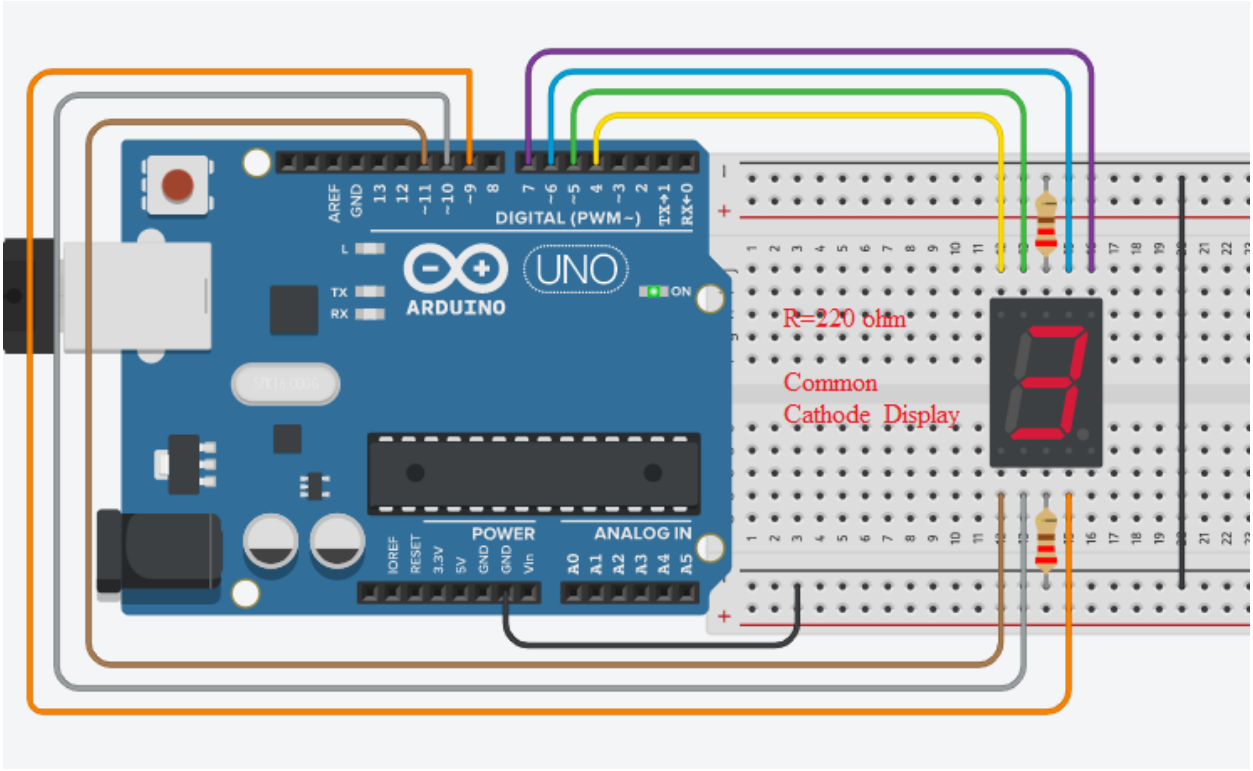
Κύκλωμα 16:

Τίτλος κυκλώματος: Αυξανόμενος μετρητής 0-9 σε αριθμητική οθόνη 7 τμημάτων

Περιγραφή κυκλώματος: Για να εμφανιστεί ένας αριθμός στην οθόνη, ανάβουν τα αντίστοιχα LED από τα 7 LED, που αντιστοιχούν στα γράμματα a, b, c, d, e, f, g.

ΠΑΡΑΤΗΡΗΣΗ: Η οθόνη επτά τμημάτων είναι μια μορφή ηλεκτρονικής συσκευής απεικόνισης για την εμφάνιση δεκαδικών αριθμών.






```
/*    " Αυξανώμενος μετρητής 0-9 σε αριθμητική οθόνη 7 τμημάτων "    */

int       a=6,  b=7,  c=9,  d=10,  e=11,  f=5,  g=4;
int  number;

void setup() {
pinMode(a,  OUTPUT);
pinMode(b,  OUTPUT);
pinMode(c,  OUTPUT);
pinMode(d,  OUTPUT);
pinMode(e,  OUTPUT);
pinMode(f,  OUTPUT);
pinMode(g,  OUTPUT);
}

void loop()  {
for(number=0;  number< =9;  number++)    {
delay(1000);
switch(number)  {

 case 0:
digitalWrite (a, HIGH);
digitalWrite (b, HIGH);
digitalWrite (c, HIGH);
digitalWrite (d, HIGH);
digitalWrite (e, HIGH);
digitalWrite (f, HIGH);
digitalWrite (g, LOW);
break;

case 1:
digitalWrite (a, LOW);
digitalWrite (b, HIGH);
digitalWrite (c, HIGH);
```

```
digitalWrite (d, LOW);
digitalWrite (e, LOW);
digitalWrite (f, LOW);
digitalWrite (g, LOW);
break;

case 2:
digitalWrite (a, HIGH);
digitalWrite (b, HIGH);
digitalWrite (c, LOW);
digitalWrite (d, HIGH);
digitalWrite (e, HIGH);
digitalWrite (f, LOW);
digitalWrite (g, HIGH);
break;

case 3:
digitalWrite (a, HIGH);
digitalWrite (b, HIGH);
digitalWrite (c, HIGH);
digitalWrite (d, HIGH);
digitalWrite (e, LOW);
digitalWrite (f, LOW);
digitalWrite (g, HIGH);
break;

case 4:
digitalWrite (a,LOW );
digitalWrite (b, HIGH);
digitalWrite (c, HIGH);
digitalWrite (d, LOW);
digitalWrite (e, LOW);
digitalWrite (f, HIGH);
digitalWrite (g, HIGH);
break;

case 5:
digitalWrite (a, HIGH);
digitalWrite (b, LOW);
digitalWrite (c, HIGH);
digitalWrite (d, HIGH);
digitalWrite (e, LOW);
digitalWrite (f, HIGH);
digitalWrite (g, HIGH);
break;

case 6:
digitalWrite (a, HIGH);
digitalWrite (b, LOW);
digitalWrite (c, HIGH);
digitalWrite (d, HIGH);
digitalWrite (e, HIGH);
digitalWrite (f, HIGH);
digitalWrite (g, HIGH);
break;

case 7:
digitalWrite (a, HIGH);
digitalWrite (b, HIGH);
digitalWrite (c, HIGH);
digitalWrite (d, LOW);
digitalWrite (e, LOW);
digitalWrite (f, LOW);
digitalWrite (g, LOW);
break;

case 8:
digitalWrite (a, HIGH);
digitalWrite (b, HIGH);
digitalWrite (c, HIGH);
digitalWrite (d, HIGH);
digitalWrite (e, HIGH);
digitalWrite (f, HIGH);
digitalWrite (g, HIGH);
break;

case 9:
digitalWrite (a, HIGH);
digitalWrite (b, HIGH);
digitalWrite (c, HIGH);
digitalWrite (d, HIGH);
digitalWrite (e, LOW);
digitalWrite (f, HIGH);
digitalWrite (g, HIGH);
break;
}
}
}
```

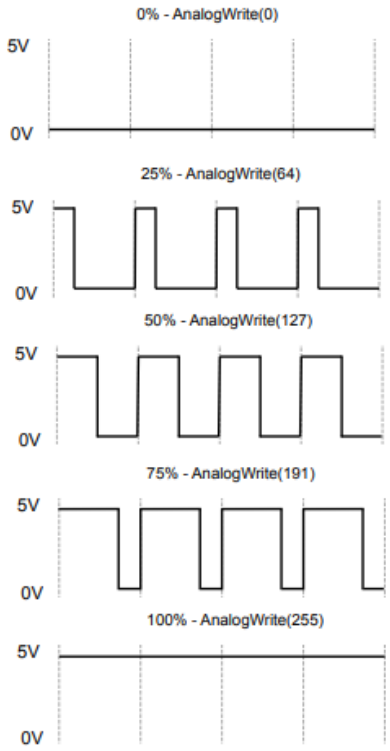
Κύκλωμα 17:

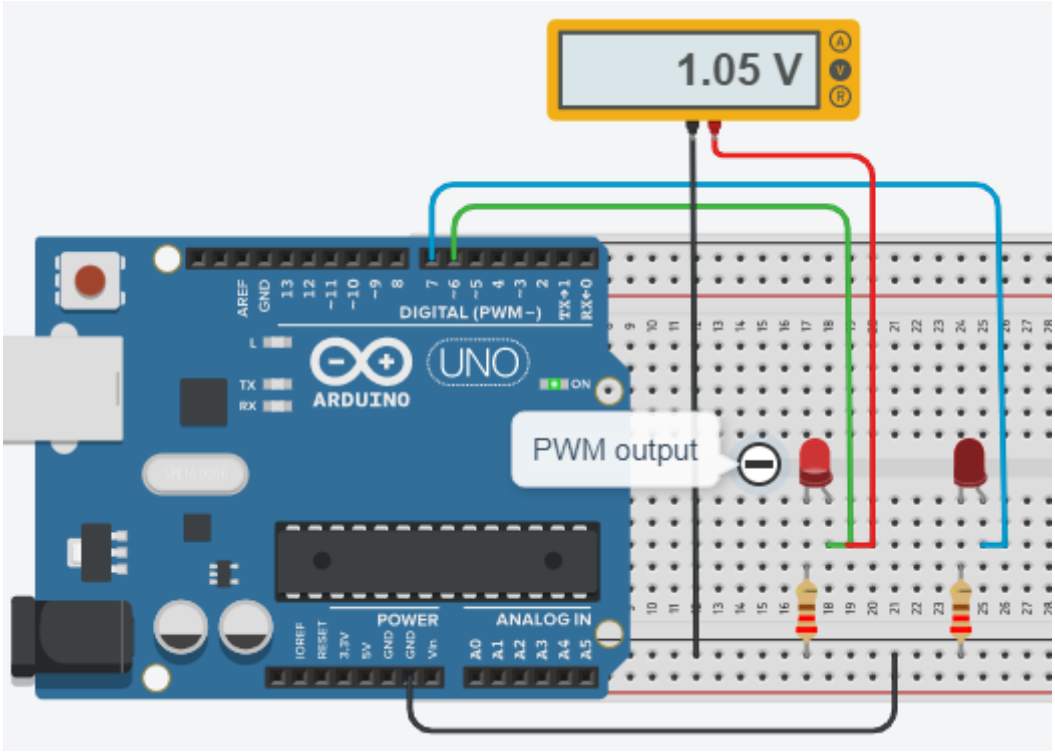
Τίτλος κυκλώματος: Χρήση της τεχνικής PWM

Περιγραφή κυκλώματος: Στην πράξη, χρησιμοποιείται το pin-6 ως έξοδος PWM και το pin-7 ως ψηφιακή έξοδος. Έτσι, μπορούμε να παρατηρήσουμε τη διαφορά μεταξύ τους. Εφαρμογές όπως ρύθμιση φωτεινότητας LED, έλεγχος ταχύτητας κινητήρα κλπ. μπορούν να πραγματοποιηθούν με τη μέθοδο PWM.

ΠΑΡΑΤΗΡΗΣΗ: Το Arduino υποστηρίζει PWM (σε ορισμένες θύρες που σημειώνονται με μια περισπωμένη (~) στην πλακέτα Arduino - θύρες 3, 4,5,9,10 και 11) στα 500Hz. (500 φορές το δευτερόλεπτο.) Μπορείτε να δώσετε μια τιμή μεταξύ 0 και 255. 0 σημαίνει ότι δεν είναι ποτέ 5V. 255 σημαίνει ότι είναι πάντα 5V. Για να το κάνετε αυτό, πραγματοποιείτε μια κλήση στο analogWrite () με την τιμή. Ο λόγος του χρόνου "ON" προς τον συνολικό χρόνο ονομάζεται "κύκλος λειτουργίας". Μια έξοδος PWM που είναι ενεργοποιημένη στο μισό χρόνο λέγεται ότι έχει κύκλο λειτουργίας 50%.

Μπορείτε να σκεφτείτε ότι το PWM είναι ενεργοποιημένο για x/255 όπου x είναι η τιμή που στέλνετε με το analogWrite (). Παρακάτω είναι ένα παράδειγμα που δείχνει πώς μοιάζουν οι παλμοί:



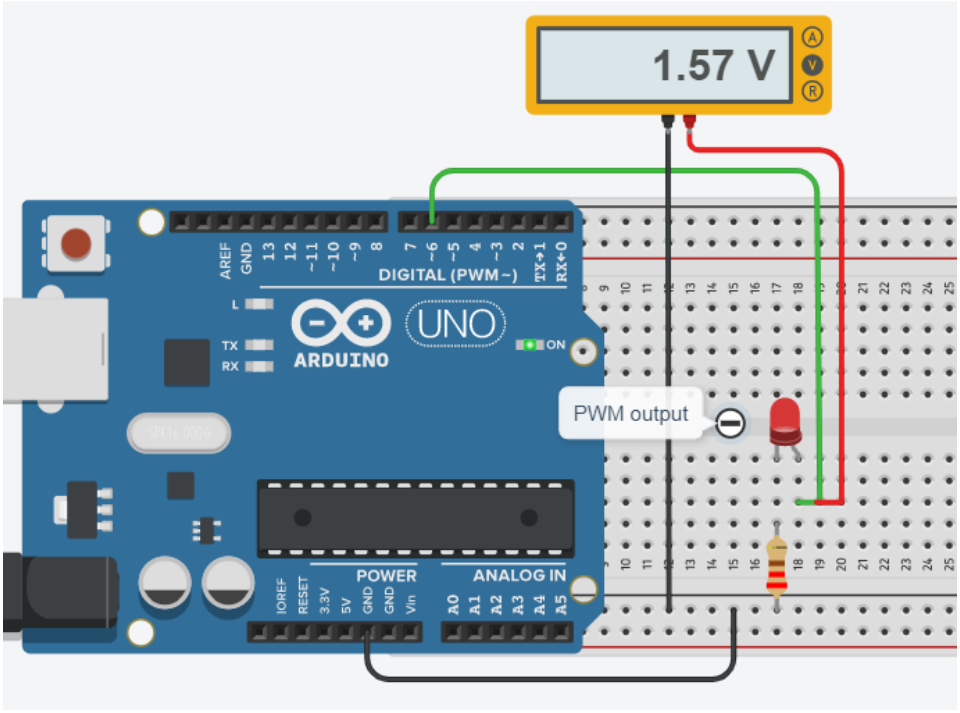



```
/* Χρήση της τεχνικής PWM μέσω της analogWrite()   */

#define    led1  6
#define    led2  7

void setup()   {
pinMode(led1, OUTPUT);
pinMode(led2, OUTPUT);       }

void loop()                  {

 analogWrite(led1, 60);        //  The value of 60 is sent to Led1 pin, i.e 1.05 V.

 analogWrite(led2,60);         //  The value of 60 is sent to Led2 pin, i.e 1.05 V.

delay (100);                   // only, the led1   lights.
}
```

Κύκλωμα 18:

Τίτλος κυκλώματος: Αλλάγή φωτεινότητας μιας λυχνίας LED από ελάχιστη σε μέγιστη.

Περιγραφή κυκλώματος: Χρησιμοποιώντας την τεχνική PWM, η φωτεινότητα ενός LED θα αλλάξει από το ελάχιστο στο μέγιστο. Εδώ θα χρησιμοποιηθούν μαζί η συνάρτηση analogWrite () και οι εντολές for.



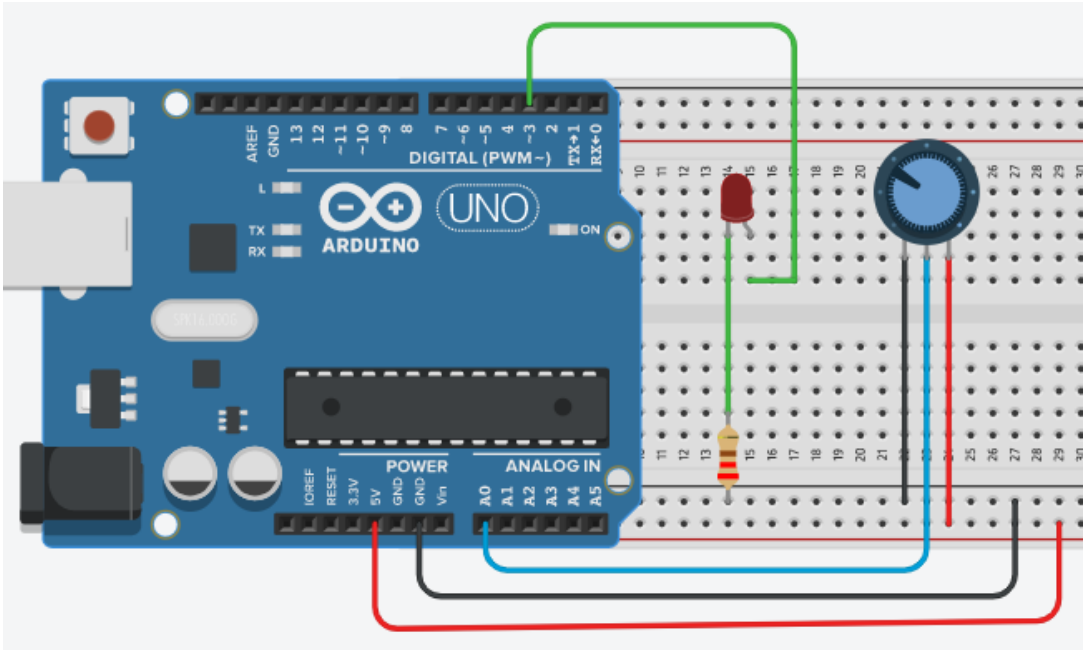

```
/* Αλλάγή φωτεινότητας μιας λυχνίας LED από ελάχιστη σε μέγιστη  */
#define led1    6
int a;
void setup()          {
Serial.begin(9600);
pinMode(led1, OUTPUT);    }
void loop() {
 for (a=0; a<=255; a++) {
 Serial.println(a);
analogWrite(led1, a);

 delay (100);
}               }
```

ΠΑΡΑΤΗΡΗΣΗ: Στο βολτόμετρο εμφανίζονται τιμές που κυμαίνονται από 0 έως 5 Volts. Οι τιμές από 0 έως 255 εμφανίζονται στη σειριακή οθόνη.

Κύκλωμα 19:

Τίτλος κυκλώματος: Ο φωτομετρητής χαμηλώνει το led.

Περιγραφή κυκλώματος: Χρησιμοποιώντας το ποτενσιόμετρο (10K), μεταβάλεται η φωτεινότητα ενός LED από το ελάχιστο στο μέγιστο. Εδώ, χρησιμοποιείται η συνάρτηση map().



ΠΑΡΑΤΗΡΗΣΗ:

map() Function:

Επανα-ορίζει έναν αριθμό από το ένα εύρος στο άλλο. Δηλαδή, μια τιμή **fromLow** θα αντιστοιχιζόταν στο **toLow**, μια τιμή **fromHigh** σε **toHigh**, ενδιάμεσες τιμές σε ενδιάμεσες τιμές κ.λπ.

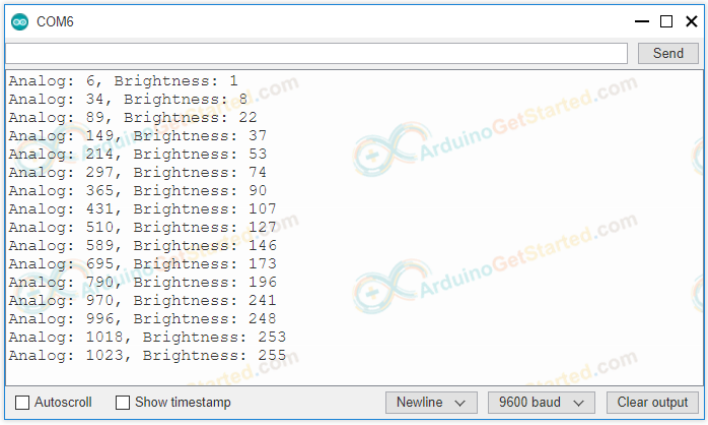
Δεν περιορίζει τις τιμές εντός του εύρους, επειδή μερικές φορές οι τιμές εκτός εύρους είναι σκόπιμες και χρήσιμες. Η συνάρτηση constrain () μπορεί να χρησιμοποιηθεί είτε πριν είτε μετά από αυτήν τη συνάρτηση, εάν απαιτούνται όρια στις περιοχές τιμών.

Η συνάρτηση map () χρησιμοποιεί ακέραια μαθηματικά, οπότε δεν θα παράγει κλάσματα, όταν τα μαθηματικά ενδέχεται να οδηγούν σε αυτό. Τα κλασματικά υπολείμματα περικόπτονται και δεν στρογγυλεύονται ή δεν υπολογίζονται κατά μέσο όρο.

Σύνταξη: map(value, fromLow, fromHigh, toLow, toHigh);

Παράδειγμα: val = map(val, 0, 1023, 0, 255);

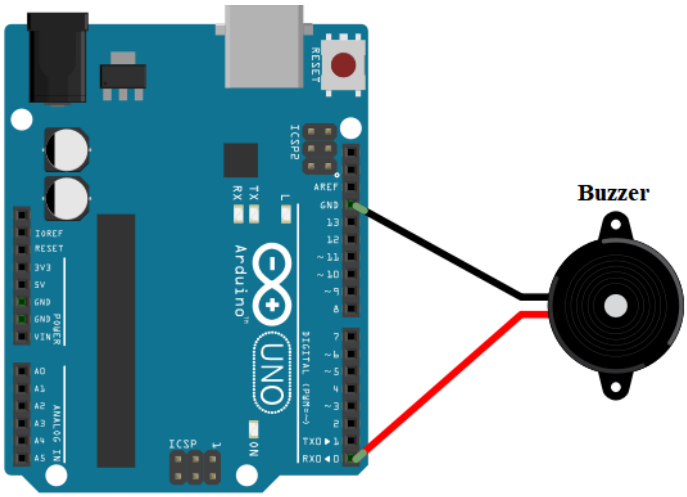
```
/*     Ο φωτομετρητή χαμηλώνει το led.   */

int LED_PIN = 3;

void setup() {
Serial.begin(9600);
pinMode(LED_PIN, OUTPUT);       }

void loop() {

  // reads the input on analog pin A0 (value between 0 and 1023)

 int analogValue = analogRead(A0);

  // scales it to brightness (value between 0 and 255)

 int brightness = map(analogValue, 0, 1023, 0, 255);

  // sets the brightness LED that connects to  pin 3

 analogWrite(LED_PIN, brightness);

  // print out the value

  Serial.print("Analog: ");

  Serial.print(analogValue);

  Serial.print(" Brightness: ");

  Serial.println(brightness);

 delay(100);
// Serial Monitor screen i below
}
```



Κύκλωμα 20:

Τίτλος Κυκλώματος: Χρήση buzzer

Περιγραφή Κυκλώματος: Ο βομβητής είναι μια συσκευή ηχητικού σήματος, η οποία μπορεί να είναι μηχανική, ηλεκτρομηχανική ή πιεζοηλεκτρική (piezo για συντομία). Οι τυπικές χρήσεις των βομβητών στη βιομηχανία είναι ως συσκευές συναγερμού.

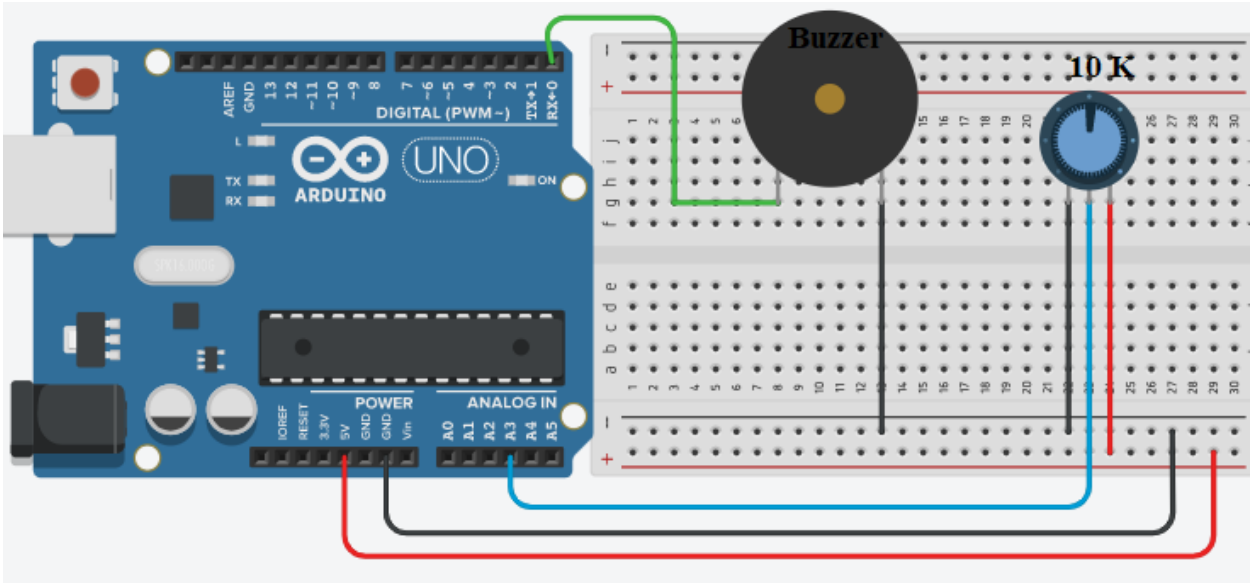



```
/* To use a buzzer  in Arduino circuits*/

#define buzzer_pin  0

 void setup()  {

 pinMode(buzzer_pin, OUTPUT);        }

void loop()       {

 digitalWrite(buzzer_pin, HIGH);

 delay(500);

 digitalWrite(buzzer_pin, LOW);

 delay(500);      }
```

Κύκλωμα 21:

Τίτλος Κυκλώματος: Έλεγχος buzzer με Ποτενσιόμετρο

Περιγραφή Κυκλώματος: Όταν η τιμή του ποτενσιόμετρου είναι μεγαλύτερη από 500, ακούγεται ο βομβητής.




```
/* Έλεγχος  buzzer με Ποτενσιόμετρο  */

const int POT_PIN = A3;              // Arduino pin connected to Pot pin
const int BUZZER_PIN  = 0;           // Arduino pin connected to Buzzer's pin
const int ANALOG_THRESHOLD  = 500;
int analogValue;

void setup() {
  pinMode(BUZZER_PIN, OUTPUT);       // set arduino pin to output mode
}
void loop() {
 analogValue = analogRead(POT_PIN);              // read the input on analog pin
if(analogValue > ANALOG_THRESHOLD)
digitalWrite(BUZZER_PIN, HIGH);                  // turn on Buzzer

else
digitalWrite(BUZZER_PIN, LOW);                   // turn off Buzzer

}
```

Erasmus+ KA-202

Strategic Partnerships Project for Vocational Education and Training

Project Title: "Teaching and Learning Arduinos in Vocational Training"

Project Acronym: "ARDUinVET"

Project No: "2020-1-TR01-KA202-093762"

Οθόνη LCD και εκπαιδευτικό κιτ

Μονάδα LCD και εκπαιδευτικό KIT

Σε αυτήν την ενότητα θέλουμε να εξηγήσουμε πώς να εμφανίζουμε μηνύματα κατάστασης ή μετρήσεις αισθητήρων του Arduino σε οθόνες LCD. Είναι εξαιρετικά συνηθισμένο και ένας γρήγορος τρόπος για να προσθέσετε μια ευανάγνωστη διεπαφή στο έργο σας.

Η οθόνη LCD αποτελεί συντομογραφία μιας οθόνης υγρών κρυστάλλων. Είναι μια μονάδα προβολής που χρησιμοποιεί υγρούς κρυστάλλους για να παράγει μια ορατή εικόνα. Όταν εφαρμόζεται ρεύμα σε αυτό το ειδικό είδος κρυστάλλου, γίνεται αδιαφανές εμποδίζοντας τον οπίσθιο φωτισμό που βρίσκεται πίσω από την οθόνη. Ως αποτέλεσμα, η συγκεκριμένη περιοχή θα γίνει σκοτεινή σε σύγκριση με άλλες. Και έτσι εμφανίζονται οι χαρακτήρες στην οθόνη.

Διασύνδεση μονάδας LCD 16×2 χαρακτήρων με Arduino

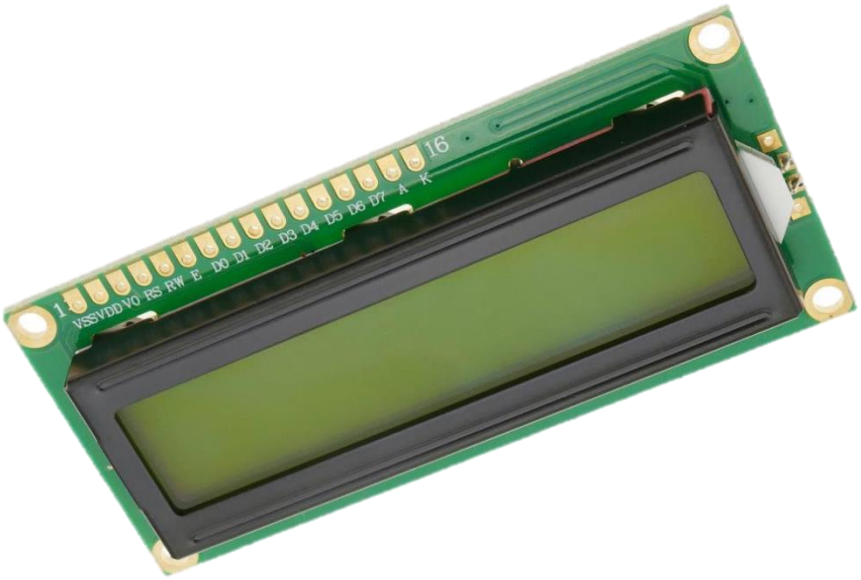
Υπάρχουν διάφορα είδη οθονών LCD που μπορείτε να συνδέσετε στο Arduino σας. Το πιο συνηθισμένο βασίζεται σε τσιπ ελεγκτή LCD παράλληλης διεπαφής της Hitachi που ονομάζεται HD44780.

Αυτές οι οθόνες LCD είναι ιδανικές για την εμφάνιση μόνο κειμένου/χαρακτήρων, εξ ου και η ονομασία «Character LCD». Η οθόνη διαθέτει οπίσθιο φωτισμό LED και μπορεί να εμφανίσει 32 χαρακτήρες ASCII σε δύο σειρές με 16 χαρακτήρες σε κάθε σειρά.

Υπάρχουν λίγα ορθογώνια για κάθε χαρακτήρα στην οθόνη, καθένα από αυτά τα ορθογώνια είναι ένα πλέγμα 5×8 pixel.

Αν και εμφανίζουν μόνο κείμενο, διατίθενται σε πολλά μεγέθη και χρώματα: για παράδειγμα, 16×1, 16×4, 20×4, με λευκό κείμενο σε μπλε φόντο, με μαύρο κείμενο σε πράσινο και πολλά άλλα.

Η κοινότητα του Arduino έχει ήδη αναπτύξει μια βιβλιοθήκη για να χειρίζεται οθόνες LCD HD44780 (Liquid Crystal Library), με τις οποίες θα έχουμε διασύνδεση σε λίγο.



Παρακάτω βλέπουμε τους ακροδέκτες οθόνης LCD:

- **VSS:** είναι ένας ακροδέκτης γείωσης και θα πρέπει να συνδεθεί με τη γείωση του Arduino;
- **VDD:** συνδεδεμένο σε 5 V;
- **VD:** για ρύθμιση αντίθεσης (συνδεδεμένη με την κεντρική ακίδα του ποτενσιόμετρου) ;
- **RS:** ελέγχει σε ποια ζώνη μνήμης lcd αποθηκεύονται τα απεσταλμένα δεδομένα;
- **R/W:** επιλογή λειτουργίας ανάγνωσης/εγγραφής;
- **E:** εάν είναι ενεργοποιημένο, επιτρέπει στη μονάδα LCD να εκτελεί ειδικές οδηγίες;
- **D0 to D7:** ακροδέκτες μετάδοσης δεδομένων;
- **A and K:** άνοδος και κάθοδος για την παροχή οπίσθιου φωτισμού στη μονάδα LCD.

Arduino - LCD Λειτουργίες (εντολές)

LiquidCrystal lcd() - Δημιουργεί μια μεταβλητή τύπου LiquidCrystal. Η οθόνη μπορεί να ελεγχθεί χρησιμοποιώντας 4 ή 8 γραμμές δεδομένων. Εάν προηγουμένως, έχουν παραλειφθεί οι αριθμοί των ακίδων από το d0 έως το d3 και έχουν αφεθεί αυτές οι γραμμές ασύνδετες. Ο ακροδέκτης RW μπορεί να συνδεθεί στη γείωση αντί να συνδεθεί με έναν ακροδέκτη στο Arduino. Εάν ναι, παραλείψτε το από τις παραμέτρους αυτής της συνάρτησης. Σύνταξη:

LiquidCrystal(rs, enable, d4, d5, d6, d7)
LiquidCrystal(rs, rw, enable, d4, d5, d6, d7)
LiquidCrystal(rs, enable, d0, d1, d2, d3, d4, d5, d6, d7)
LiquidCrystal(rs, rw, enable, d0, d1, d2, d3, d4, d5, d6, d7)

Παράμετροι:

rs: ο αριθμός του ακροδέκτη Arduino που είναι συνδεδεμένος με τον ακροδέκτη RS στην οθόνη LCD

rw: ο αριθμός του ακροδέκτη Arduino που είναι συνδεδεμένος με τον ακροδέκτη RW στην οθόνη LCD (προαιρετικό)

enable: ο αριθμός της ακίδας Arduino που είναι συνδεδεμένος στην ακίδα ενεργοποίησης στην οθόνη LCD

d0, d1, d2, d3, d4, d5, d6, d7: οι αριθμοί των ακίδων Arduino που συνδέονται με τις αντίστοιχες ακίδες δεδομένων στην οθόνη LCD. Τα d0, d1, d2 και d3 είναι προαιρετικά. Εάν παραληφθεί, η οθόνη LCD θα ελέγχεται χρησιμοποιώντας μόνο τις τέσσερις γραμμές δεδομένων (d4, d5, d6, d7).

lcd.begin() - Αρχικοποιεί τη διεπαφή με την οθόνη LCD και καθορίζει τις διαστάσεις (πλάτος και ύψος) της οθόνης. Η συνάρτηση begin() πρέπει να κληθεί πριν από οποιεσδήποτε άλλες εντολές βιβλιοθήκης LCD. Σύνταξη:

lcd.begin(cols, rows)

Παράμετροι:

lcd: a variable of type LiquidCrystal

cols: ο αριθμός των στηλών που έχει η οθόνη

rows: ο αριθμός των σειρών που έχει η οθόνη

lcd.print() – Εκτυπώνει κείμενο στην οθόνηLCD. Σύνταξη:

lcd.print(data)

lcd.print(data, BASE)

Παράμετροι:

lcd: μεταβλητή τύπου LiquidCrystal

data: Τα δεδομένα προς εκτύπωση (char, byte, int, long, or string)

BASE (optional): η βάση στην οποία εκτυπώνονται οι αριθμοί: BIN για δυαδικό (βάση 2), DEC για δεκαδικό (βάση 10), OCT για οκταδικό (βάση 8), HEX για δεκαεξαδικό (βάση 16).

Επιστροφές

byte print() επιστρέφει τον αριθμό των byte που γράφτηκαν, αν και η ανάγνωση αυτού του αριθμού είναι προαιρετική

lcd.setCursor() - Τοποθετήστε τον κέρσορα LCD. Δηλαδή, ορίστε τη θέση στην οποία θα εμφανίζεται το επόμενο κείμενο που θα γραφτεί στην οθόνη LCD. Σύνταξη:

lcd.setCursor(col, row)

Παράμετροι:

lcd: παράμετρος τύπου LiquidCrystal

col: η στήλη στην οποία θα τοποθετηθεί ο κέρσορας (με το 0 να είναι η πρώτη στήλη)

row: η σειρά στην οποία θα τοποθετηθεί ο κέρσορας (με το 0 να είναι η πρώτη σειρά)

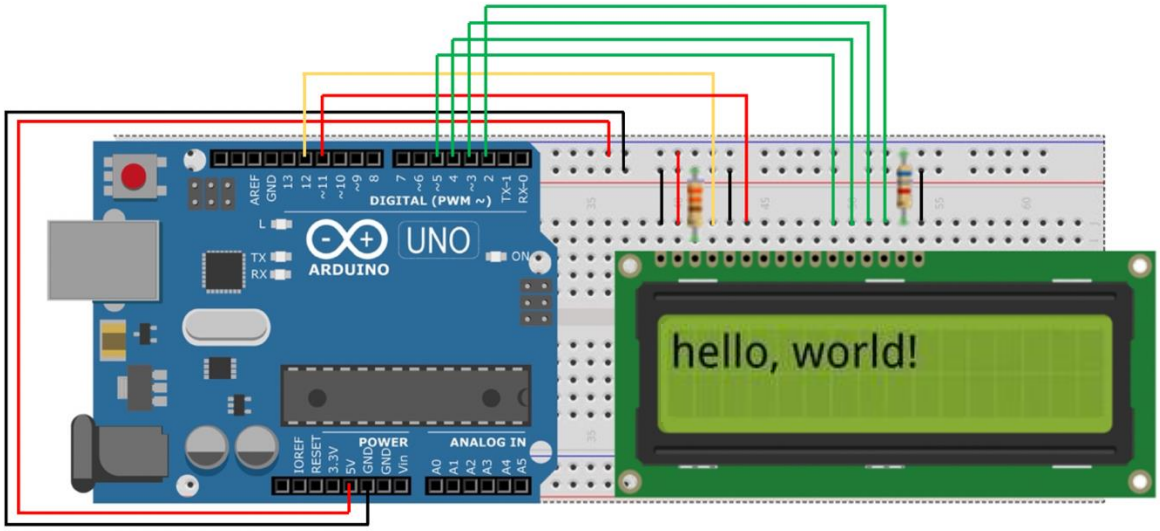
lcd.clear(); - Καθαρίζει την οθόνη LCD και τοποθετεί τον κέρσορα στην επάνω αριστερή γωνία.

Syntax: lcd.clear()

Parameters: lcd: παράμετρος τύπου LiquidCrystal

Κύκλωμα 1:
Τίτλος κυκλώματος: "Εκτύπωση μηνύματος στην οθόνη LCD"
Περιγραφή κυκλώματος: παρέχει δυνατότητα σύνδεσης μιας οθόνης LCD στον μικροελεγκτή και εκτύπωση επιθυμητών μηνυμάτων στην οθόνη.

Σημείωση: πρέπει να συμπεριλάβετε τη βιβλιοθήκη LiquidCrystal.h για να χρησιμοποιήσετε τις λειτουργίες LCD που περιγράφονται παρακάτω.




```
/*  Print a message  */

#include <LiquidCrystal.h> // include the library code

//constants for the number of rows and columns in the LCD

const int numRows = 2;

const int numCols = 16;

// initialize the library with the numbers of the interface pins

LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup()

{

lcd.begin(numCols, numRows);

lcd.print("hello, world!"); // Print a message to the LCD.

}
```

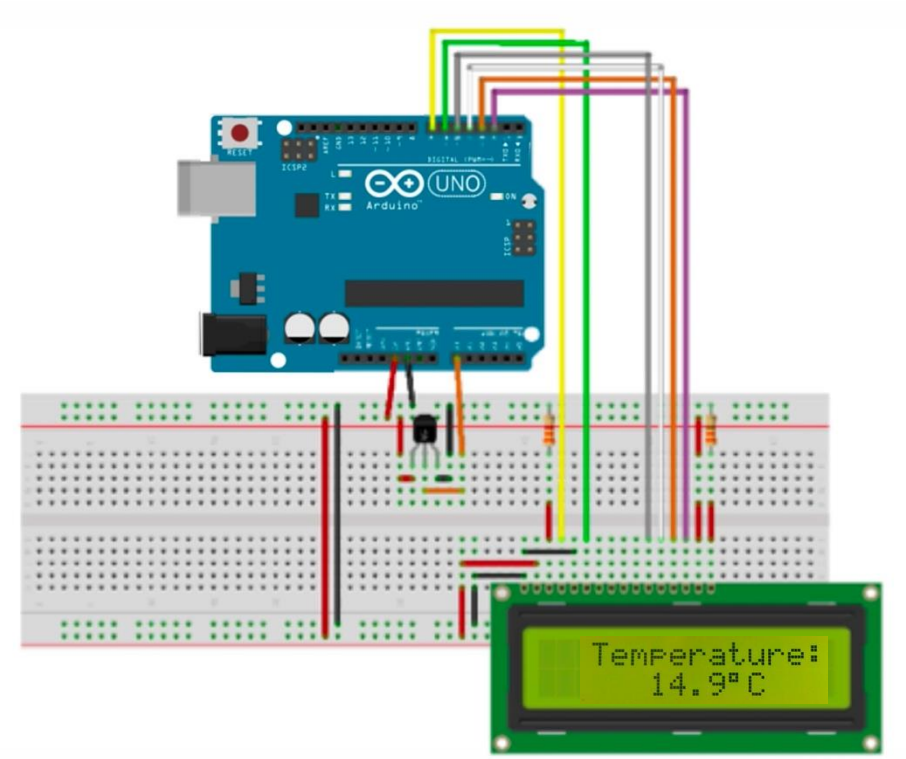
Κύκλωμα 2:
Τίτλος κυκλώματος: " Ψηφιακό Θερμόμετρο"

Περιγραφή κυκλώματος: Δημιουργήστε ένα ψηφιακό θερμόμετρο με το Arduino. Η θερμοκρασία θα μετριέται από αισθητήρα θερμοκρασίας LM35 και θα εμφανίζεται σε οθόνη LCD η οποία θα ενημερώνεται κάθε δευτερόλεπτο.

Η ακίδα Vo της οθόνης LCD προσαρμόζει την αντίθεση των χαρακτήρων που εμφανίζονται στην οθόνη. Όπως προαναφέρθηκε, πρέπει να συνδεθεί με αντίσταση 3,3 kΩ συνδεδεμένη στο

GND. Ωστόσο, σε ορισμένες οθόνες μπορεί να χρειαστεί να συνδέσετε την καρφίτσα Vo απευθείας στο GND

Σημείωση: ο αισθητήρας LM35 έχει 3 ακροδέκτες: έναν για την παροχή ρεύματος, έναν για τη γείωση και έναν για την έξοδο της τάσης ανάλογη με την ανιχνευόμενη θερμοκρασία, η οποία είναι ίση με 10 mV για κάθε βαθμό Κελσίου και βαθμονομείται σε βαθμούς Κελσίου.




```
/*  Digital Thermometer  */
#include <LiquidCrystal.h> //Library to drive LCD display
#define pin_temp A0 //Temperature sensor Vout foot connection pin
float temp = 0; //Variable in which the detected temperature will be stored
```

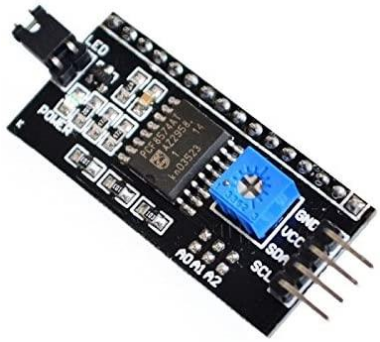
```
LiquidCrystal lcd(7, 6, 5, 4, 3, 2); //Initializing the library with LCD display pins
void setup()
{
  lcd.begin(16, 2); //Setting the number of columns and rows in the display
  LCD  lcd.setCursor(0, 0); //Move the cursor over the first row (row 0) and the first
  column lcd.print ("Temperature:"); Print the message 'Temperature:' on the first line
 //*Imposed ADC Vref at 1.1V
 (for greater accuracy in temperature calculation)
  IMPORTANT: If you use Arduino Mega replace INTERNAL with INTERNAL1V1 */
  analogReference(INTERNAL);
}
void loop()
{
  /*calculate the temperature ==============================================*/
  temp = 0;
  for (int i = 0; i < 5; i++) { //It executes the next statement 5 times
    temp += (analogRead(pin_temp) / 9.31); // It calculates temperature and sum at variable
    'temp'
  }
  temp /= 5; //It calculates the mathematical average of temperature values
  /*=====================================================================*/

  /*I see the temperature on the LCD display
  ==========================================*/
  lcd.setCursor(0, 1); //lcd.print(temp); Move the cursor over the first column
  and the second row lcd.print(temp);
  // Mold on LCD display temperature
  lcd.print(" C"); // Mold a space and the 'C' font on the display
 /*=========================================*/
  delay(1000); //Delay by one second (it can be changed)
}
```

Διασύνδεση I2C LCD with Arduino

Εάν θέλετε να συνδέσετε μια οθόνη LCD με το Arduino, πρέπει να καταναλώσετε πολλές ακίδες στο Arduino. Ακόμη και σε λειτουργία 4-bit, το Arduino εξακολουθεί να απαιτεί συνολικά επτά συνδέσεις, που είναι οι μισές από τις διαθέσιμες ψηφιακές ακίδες I/O.

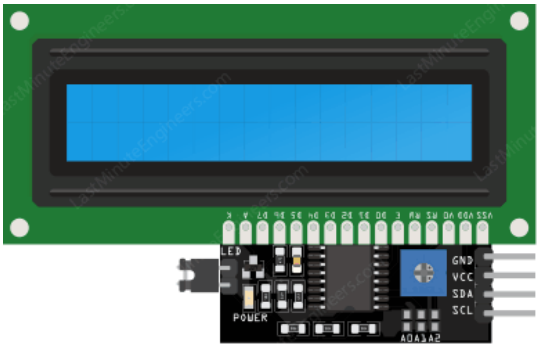
Η λύση είναι να χρησιμοποιήσετε μια οθόνη LCD που διασυνδέεται με το πρωτόκολλο I2C. Καταναλώνει μόνο δύο ακίδες I/O, οι οποίες δεν μπορούν να αποτελούν καν μέρος ενός ψηφιακού συνόλου ακίδων I/O και μπορούν να μοιραστούν και με άλλες συσκευές I2C.

Η πιο διάχυτη οθόνη LCD I2C αποτελείται από μια οθόνη LCD χαρακτήρων HD44780 και έναν προσαρμογέα LCD I2C.



Το πιο σημαντικό μέρος του προσαρμογέα είναι το τσιπ 8-bit I/O Expander – PCF8574. Αυτό το τσιπ μετατρέπει τα δεδομένα I2C από ένα Arduino στα παράλληλα δεδομένα που απαιτούνται από την οθόνη LCD. Η πλακέτα περιλαμβάνει ένα ποτενσιόμετρο με το οποίο μπορείτε να κάνετε λεπτές ρυθμίσεις στην αντίθεση της οθόνης. Εάν χρησιμοποιείτε περισσότερες από μία συσκευές στον ίδιο δίαυλο I2C, ίσως χρειαστεί να ορίσετε διαφορετική διεύθυνση I2C για την πλακέτα, ώστε να μην έρχεται σε διένεξη με άλλη συσκευή I2C. Για το λόγο αυτό, η πλακέτα διαθέτει τρεις βραχυκυκλωτήρες συγκόλλησης (A0, A1 και A2).

Μια οθόνη LCD I2C έχει μόνο 4 ακίδες που τη διασυνδέουν με το Arduino.



Οι ακίδες εξόδου έχουν ως εξής:

GND: είναι ένας ακροδέκτης γείωσης και θα πρέπει να συνδεθεί με τη γείωση του Arduino.;

VCC: τροφοδοτεί τη μονάδα και την οθόνη LCD. Συνδέστε το στην έξοδο 5V του Arduino ή σε ξεχωριστό τροφοδοτικό.

SDA: είναι μια ακίδα σειριακών δεδομένων. Αυτή η γραμμή χρησιμοποιείται τόσο για μετάδοση όσο και για λήψη. Συνδεθέστε τον ακροδέκτη SDA στην ακίδα του Arduino.

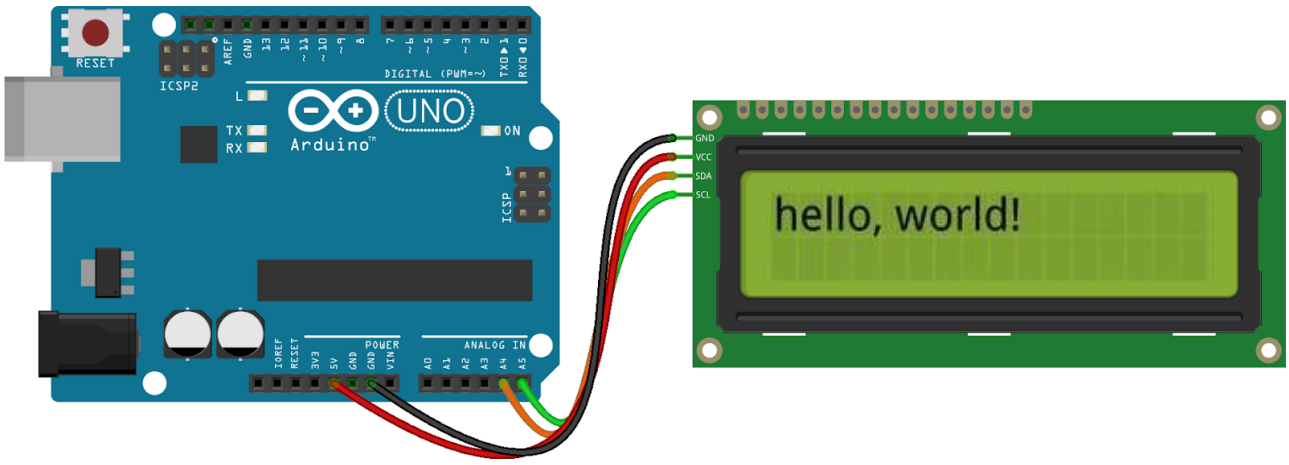
SCL: είναι μια ακίδα σειριακού ρολογιού. Αυτό είναι ένα σήμα χρονισμού που παρέχεται από τη συσκευή Bus Master. Συνδέστε τον ακροδέκτη SCL στην ακίδα του Arduino.

Στις πλακέτες Arduino με διάταξη R3, το SDA (γραμμή δεδομένων) και το SCL (γραμμή ρολογιού) βρίσκονται στις κεφαλίδες των ακροδεκτών κοντά στον ακροδέκτη AREF. Είναι επίσης γνωστά ως A5 (SCL) και A4 (SDA). Ανατρέξτε στον παρακάτω πίνακα για να προσδιορίσετε τις σωστές ακίδες ανάλογα με το μοντέλο Arduino.

	SCL	SDA
Arduino Uno	A5	A4
Arduino Nano	A5	A4
Arduino Mega	21	20
Leonardo/Micro	3	2

Κύκλωμα 3:
Τίτλος κυκλώματος: "Εκτύπωση μηνύματος σε οθόνη I2C LCD”
Περιγραφή κυκλώματος: είναι δυνατό να συνδέσετε μια οθόνη LCD στον μικροελεγκτή με μόνο 4 ακίδες και να εκτυπώσετε τα επιθυμητά μηνύματα στην οθόνη.

Σημείωση: πρέπει να εγκαταστήσετε μια βιβλιοθήκη που ονομάζεται LiquidCrystal_I2C. Αυτή η βιβλιοθήκη είναι μια βελτιωμένη έκδοση της βιβλιοθήκης LiquidCrystal που συνοδεύεται από το Arduino IDE.



```
/*  Print a message on a I2C Display */

#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x3F,16,2);  // set the LCD address to 0x3F for a 16 chars and 2 line display

void setup() {

  lcd.init();

  lcd.clear();

  lcd.backlight();      // Make sure backlight is on

  // Print a message on both lines of the LCD.

  lcd.setCursor(2,0);   //Set cursor to character 2 on line 0

  lcd.print("Hello world!");

  lcd.setCursor(2,1);   //Move cursor to character 2 on line 1

  lcd.print("with I2C protocol!");

}

void loop() {

}
```

Διασύνδεση μονάδας οθόνης γραφικών OLED με Arduino

Μια άλλη δυνατότητα χρήσης του πρωτοκόλλου I2C είναι η επιλογή μιας οθόνης OLED (Organic Light-Emitting Diode). Είναι εξαιρετικά ελαφριά και λεπτή και παράγει μια πιο φωτεινή και καθαρή εικόνα.

Μια οθόνη OLED λειτουργεί χωρίς οπίσθιο φωτισμό. Αυτός είναι ο λόγος για τον οποίο η οθόνη έχει τόσο υψηλή αντίθεση, εξαιρετικά ευρεία γωνία θέασης και μπορεί να εμφανίσει βαθιές στάθμες μαύρου. Η απουσία οπίσθιου φωτισμού μειώνει σημαντικά την ισχύ που απαιτείται για τη λειτουργία της OLED.
Όπως μπορούμε να δούμε από το σχήμα, το pinout είναι η κλασική διασύνδεση I2C τεσσάρων ακίδων που φαίνεται ήδη παραπάνω.
Η κοινότητα του Arduino έχει ήδη αναπτύξει μερικές βιβλιοθήκες για να χειριστεί αυτές τις οθόνες OLED, όπως η βιβλιοθήκη SSD1306 του Adafruit. Για να εγκαταστήσετε τη βιβλιοθήκη, μεταβείτε στο Σκίτσο > Συμπερίληψη βιβλιοθήκης > Διαχείριση βιβλιοθηκών… Περιμένετε μέχρι ο Διαχειριστής βιβλιοθήκης να πραγματοποιήσει λήψη του ευρετηρίου βιβλιοθηκών και να ενημερώσει τη λίστα των εγκατεστημένων βιβλιοθηκών.

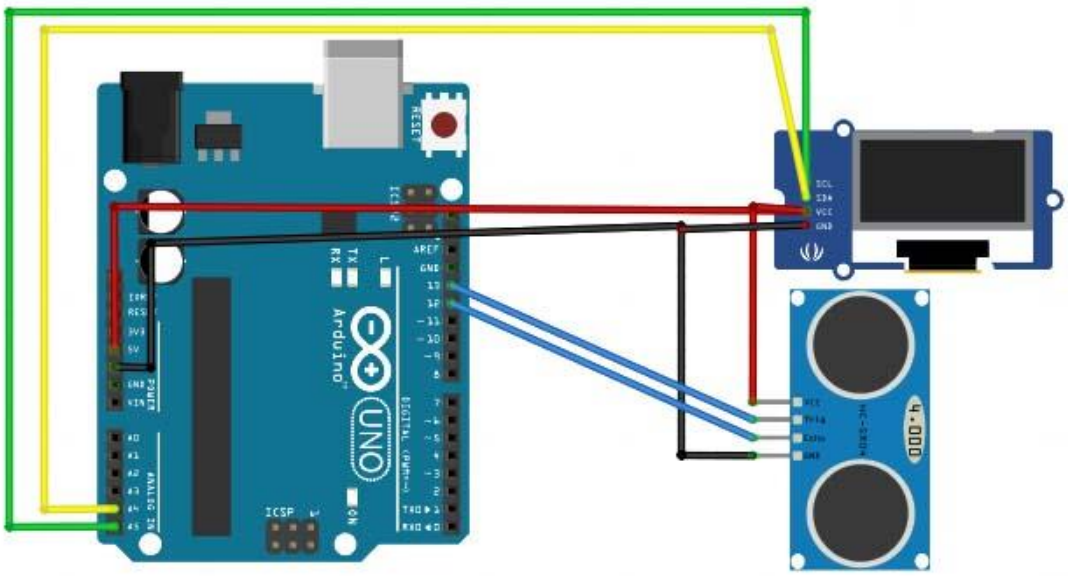
Arduino - Λειτουργίες μονάδας οθόνης γραφικών OLED (εντολές)

pinMode();

display.begin()

display.clearDisplay();

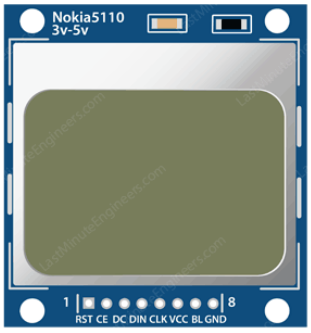
Κύκλωμα 4:
Τίτλος κυκλώματος: "Αισθητήρας απόστασης"
Περιγραφή κυκλώματος: Η απόσταση θα ληφθεί με τον αισθητήρα υπερήχων HC-SR04 και θα εμφανίζεται σε οθόνη OLED.
Σημείωση: Υπάρχουν μόνο τέσσερις ακίδες για τις οποίες πρέπει να ανησυχείτε στο HC-SR04: VCC (Power), Trig (Trigger), Echo (Receive) και GND (Ground).



```
/*  Distance sensor  */
#include <SPI.h>        // this library allows you to communicate with SPI devices, with the
                        Arduino as the master device.

#include <Wire.h>     // this library allows you to communicate with I2C / TWI devices
#include <Adafruit_GFX.h>            // the OLED display's libraries
#include <Adafruit_SSD1306.h>
#define CommonSenseMetricSystem
#define trigPin 13      // define the pins of the sensor
#define echoPin 12

void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  display.begin(SSD1306_SWITCHCAPVCC, 0x3C); //initialize with the I2C addr 0x3C
(128x64)
  display.clearDisplay();
}

void loop() {
  long duration, distance;
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = (duration/2) / 29.1;
  display.setCursor(22,20);    //oled display setting cursor
  display.setTextSize(3);        //size of the text
  display.setTextColor(WHITE);       //if you write black it erases things
  display.println(distance);     //print our variable
  display.setCursor(85,20);
  display.setTextSize(3);
    display.println("cm");
  display.display();
  delay(500);
  display.clearDisplay();
  Serial.println(distance);//debug
}
```

Διασύνδεση γραφικής οθόνης LCD Nokia 5110 με Arduino

Μπορείτε να διασυνδέσετε το Arduino με μικρές οθόνες LCD παρόμοιες με αυτές που χρησιμοποιεί η Nokia στα κινητά τηλέφωνα 3310 και 5110. Αυτές οι οθόνες είναι μικρές (μόνο περίπου 1,5″), φθηνές, εύχρηστες, αρκετά χαμηλής ισχύος (μόνο από 6 έως 7 mA) και μπορούν να εμφανίζουν κείμενο καθώς και bitmaps.



Πρόκειται για γραφική οθόνη 84×48 pixel. Διασυνδέονται με μικροελεγκτές μέσω μιας διεπαφής σειριακού διαύλου παρόμοια με το SPI. Η οθόνη LCD έρχεται επίσης με οπίσθιο φωτισμό σε διαφορετικά χρώματα όπως κόκκινο, πράσινο, μπλε και λευκό. Ο οπίσθιος φωτισμός δεν είναι παρά τέσσερα LED που βρίσκονται σε απόσταση γύρω από τις άκρες της οθόνης.

Έχει 8 ακίδες που το διασυνδέουν με το Arduino, το pinout έχει ως εξής:

- **RST:** επαναφέρει την οθόνη. Με τη λογική χαμηλής στάθμης. πιέζοντας αυτό μπορείτε να επαναφέρετε την οθόνη. Μπορείτε επίσης να συνδέσετε αυτήν την ακίδα στην επαναφορά του Arduino ώστε να επαναφέρει αυτόματα την οθόνη.
- **CE(Chip Enable**): χρησιμοποιείται για την επιλογή μιας από τις πολλές συνδεδεμένες συσκευές που μοιράζονται τον ίδιο δίαυλο SPI.
- **D/C(Data/Command):** Η ακίδαενημερώνει την πλακέτα οθόνης εάν τα δεδομένα που λαμβάνει είναι εντολή ή δεδομένα με δυνατότητα προβολής.
- **DIN:** είναι μια σειριακή ακίδα δεδομένων για τη διεπαφή SPI.
- **CLK:** είναι μια σειριακή ακίδα ρολογιού για τη διεπαφή SPI.
- **VCC:** τροφοδοτεί την οθόνη LCD την οποία συνδέουμε στον ακροδέκτη 3,3 V του Arduino.
- **BL(Backlight):** ελέγχει τον οπίσθιο φωτισμό της οθόνης. Για να ελέγξετε τη φωτεινότητά του, μπορείτε να προσθέσετε ένα ποτενσιόμετρο ή να συνδέσετε αυτόν τον ακροδέκτη σε οποιαδήποτε ακίδα Arduino με δυνατότητα PWM.
- **GND:** είναι ένας ακροδέκτης γείωσης και θα πρέπει να συνδεθεί με τη γείωση του Arduino.

Μπορείτε να συνδέσετε ακροδέκτες μετάδοσης δεδομένων σε οποιαδήποτε ψηφιακή ακίδα I/O. Η οθόνη LCD έχει επίπεδα επικοινωνίας 3v, επομένως δεν μπορούμε να συνδέσουμε απευθείας αυτές τις ακίδες στο Arduino. Ένας τρόπος είναι να προσθέσετε αντιστάσεις ενσωματωμένες σε

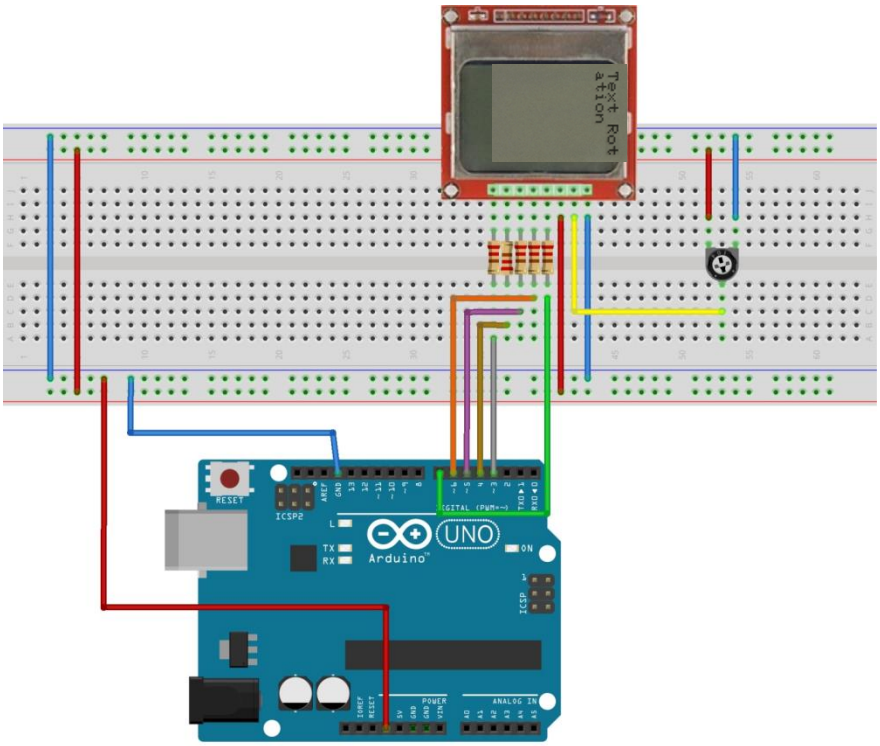
κάθε ακίδα μετάδοσης δεδομένων. Απλώς προσθέστε αντιστάσεις 10kΩ μεταξύ των ακίδων CLK, DIN, D/C και RST και μια αντίσταση 1kΩ μεταξύ CE. Η ακίδα οπίσθιου φωτισμού (BL) συνδέεται με 3,3 V μέσω αντίστασης περιορισμού ρεύματος 330Ω. Μπορείτε να προσθέσετε ένα ποτενσιόμετρο ή να συνδέσετε αυτόν τον ακροδέκτη σε οποιοδήποτε ακροδέκτη Arduino με δυνατότητα PWM, εάν θέλετε να ελέγξετε τη φωτεινότητά του.

Η κοινότητα του Arduino έχει ήδη αναπτύξει μερικές βιβλιοθήκες για να χειριστεί αυτές τις οθόνες NOKIA, όπως τη βιβλιοθήκη LCD του Nokia 5110 PCD8544 της Adafruit. Για να εγκαταστήσετε τη βιβλιοθήκη, μεταβείτε στο Sketch > Include Library > Manage Libraries… Περιμένετε μέχρι ο Διαχειριστής βιβλιοθήκης να πραγματοποιήσει λήψη του ευρετηρίου βιβλιοθηκών και να ενημερώσει τη λίστα των εγκατεστημένων βιβλιοθηκών.

Κύκλωμα 5:
Τίτλος κυκλώματος: "Περιστροφή κειμένου"
Περιγραφή κυκλώματος: Μπορείτε να περιστρέψετε τα περιεχόμενα της οθόνης καλώντας τη συνάρτηση setRotation(). Σας επιτρέπει να βλέπετε την οθόνη σας σε κατακόρυφη λειτουργία ή να την αναστρέφετε ανάποδα.

Σημείωση: Η συνάρτηση δέχεται μόνο μία παράμετρο που αντιστοιχεί σε 4 βασικές περιστροφές. Αυτή η τιμή μπορεί να είναι οποιοσδήποτε μη αρνητικός ακέραιος αριθμός που ξεκινά από το 0. Κάθε φορά που αυξάνετε την τιμή, τα περιεχόμενα της οθόνης περιστρέφονται κατά 90 μοίρες αριστερόστροφα. Για παράδειγμα:

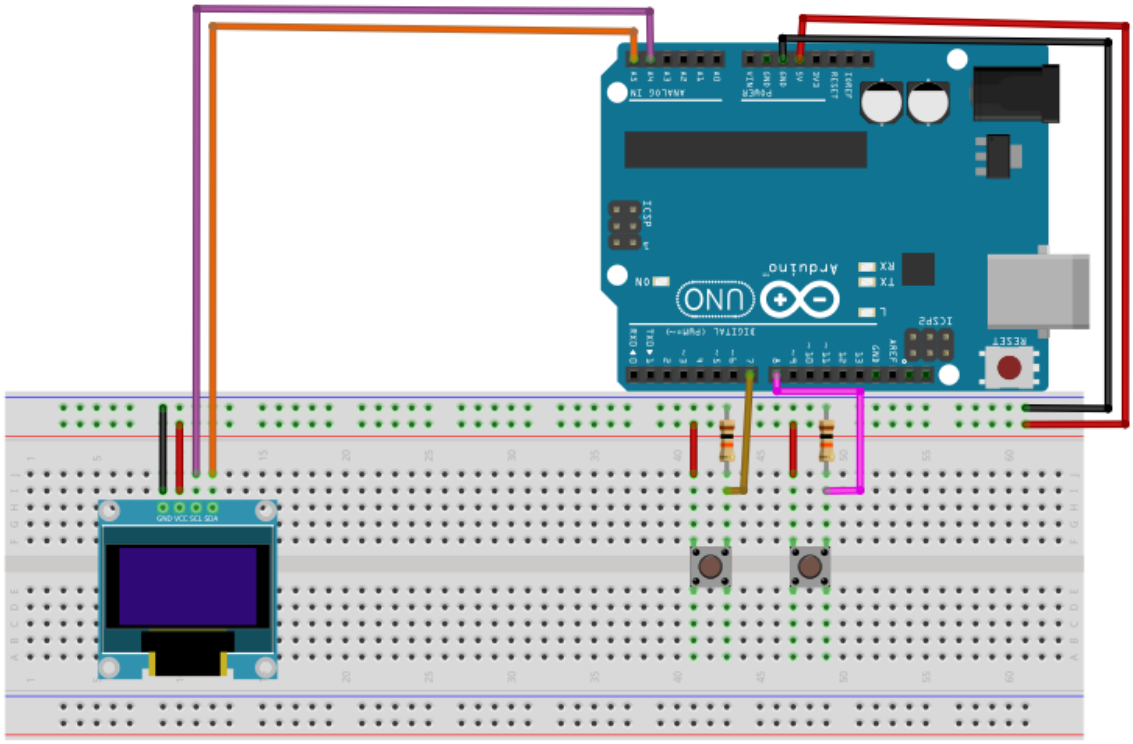
- 0-Διατηρεί την οθόνη στον τυπικό οριζόντιο προσανατολισμό.
- 1-Περιστρέφει την οθόνη κατά 90° προς τα δεξιά.
- 2-Γυρίζει την οθόνη ανάποδα.
- 3-Περιστρέφει την οθόνη κατά 90° προς τα αριστερά.




```
/*  Text Rotation  */

#include <SPI.h>        // this library allows you to communicate with SPI devices, with the
                        Arduino as the master device.
#include <Adafruit_GFX.h>           // the OLED display's libraries
#include <Adafruit_PCD8544.h>
// Declare LCD object for software SPIAdafruit_PCD8544(CLK,DIN,D/C,CE,RST);

Adafruit_PCD8544 display = Adafruit_PCD8544(7, 6, 5, 4, 3);

void setup()   {

        Serial.begin(9600);

        //Initialize Display

        display.begin();

        display.setContrast(57);        // you can change the contrast around to adapt the display

        display.clearDisplay();         // clear the buffer.

// Text Rotation

while(1)

{
```

```
        display.clearDisplay();
        display.setRotation(rotatetext);
        display.setTextSize(1);
        display.setTextColor(BLACK);
        display.setCursor(0,0);
        display.println("Text Rotation");
        display.display();
        delay(1000);
        display.clearDisplay();
        rotatetext++;
}
}
void loop() {}
```

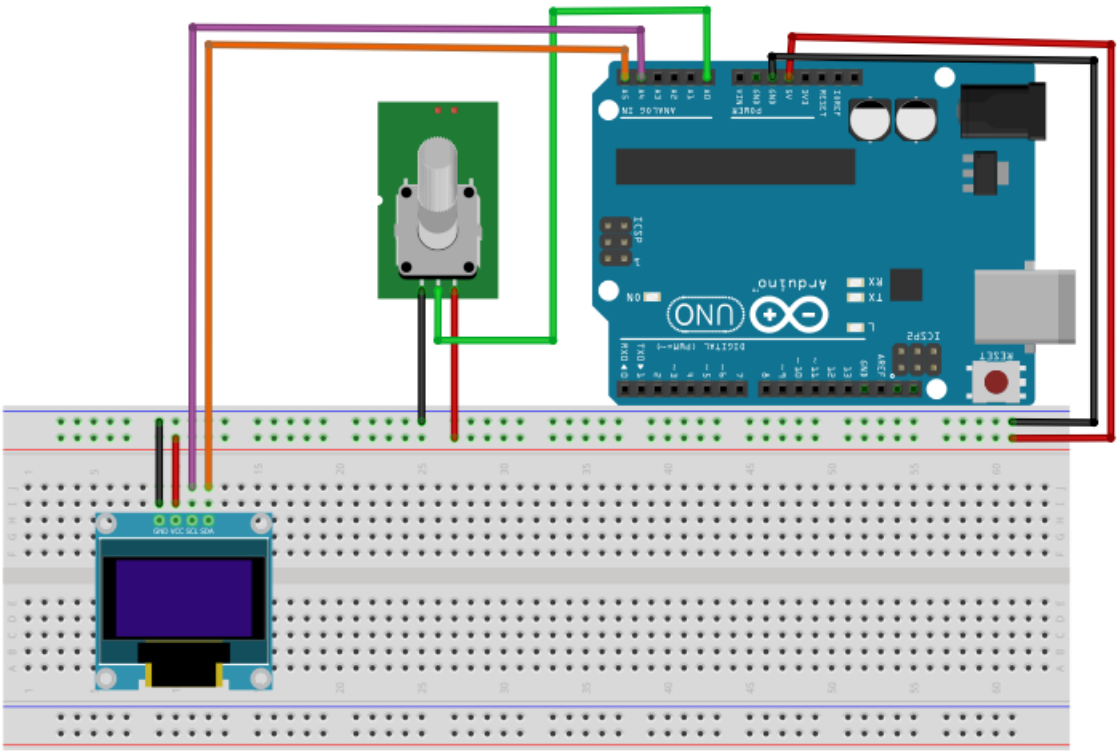
Κύκλωμα 6:
Τίτλος κυκλώματος: "Μετρητής πλήκτρων"
Περιγραφή κυκλώματος: μια οθόνη OLED που δείχνει έναν αριθμό, ο οποίος μπορεί να αυξηθεί και να μειωθεί με το πάτημα δύο διαφορετικών κουμπιών.



```
#include <U8glib.h> //lcd libraries
#include "U8glib.h"
U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE|U8G_I2C_OPT_DEV_0);  //display
model

int button_plus = 8, button_minus = 7; //declaration pin buttons
int state_plus, state_minus, number=0;

void setup() {

  pinMode(button_plus,INPUT);
  pinMode(button_minus,INPUT);

  Serial.begin(9600);

  u8g.setFont(u8g_font_fub25n); //font to be used for the write of the number
}

void write_number(void) {
  u8g.setPrintPos(10,50);
  u8g.print(number);
}

void loop() {
```

```
  //buttons
  state_plus = digitalRead(button_plus);
  state_minus = digitalRead(button_minus);

  if(state_plus==1)
  {
     number++;
     delay(50);
  }

  if(state_minus==1)
  {
    number--;
    delay(50);
  }

  //write de number sul display

  u8g.firstPage();
  do {
    write_number();
  } while  ( u8g.nextPage() );

    u8g.firstPage();
}
```

Κύκλωμα 7:
Τίτλος κυκλώματος: "Μετρητής με ποτενσιόμετρο"
Περιγραφή κυκλώματος: μια οθόνη OLED που δείχνει έναν αριθμό, που αυξάνεται και μειώνεται καθώς περιστρέφεται ένα ποτενσιόμετρο.

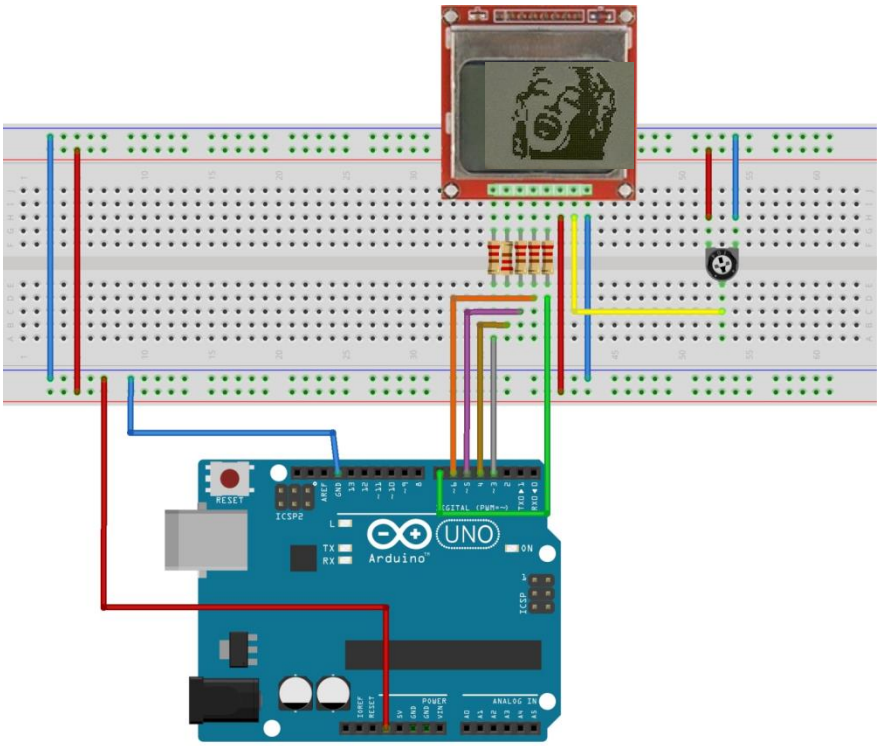


```
#include <U8glib.h> //lcd libraries

#include "U8glib.h"

U8GLIB_SSD1306_128X64 u8g(U8G_I2C_OPT_NONE|U8G_I2C_OPT_DEV_0);  //display model

int potentiometer = 0; //declaration and potentiometer

int state_pot, stop_pot, difference, number=0;

void setup() {

  Serial.begin(9600);

  u8g.setFont(u8g_font_fub25n);//font to be used for the write of the number

}
```

```
void write_number(void) {
  u8g.setPrintPos(10,50);
  u8g.print(number);
}
void loop() {
  state_pot = analogRead(potentiometer);
  //potentiometer
  stop_pot = state_pot;//save the value when it is stop to see if the potentiometer turns clockwise or counterclockwise
  delay(100);
  state_pot = analogRead(potentiometer);
  difference = stop_pot-state_pot;
  if((difference>2)||(difference<2))//is used to avoid making trades if the potentiometer is stop
  {
    number=number-difference/5;//if difference is greater than 0 then it turns clockwise and adds otherwise it subtracts
    delay(100);
  }
  //write the number on display
  u8g.firstPage();
  do {
    write_number();
  } while
  ( u8g.nextPage() );
    u8g.firstPage();
}
```

Κύκλωμα 8:
Circuit title: "Εικόνα Marilyn Bmp"
Circuit Explanation: Πώς να σχεδιάσετε εικόνες bitmap στην οθόνη LCD του Nokia 5110. Σε αυτό το παράδειγμα υπάρχει ένα πορτρέτο της Μέριλιν Μονρόε.

Σημείωση: η ανάλυση οθόνης της οθόνης LCD του Nokia 5110 είναι 84×48 pixel, επομένως οι εικόνες μεγαλύτερες από αυτήν δεν θα εμφανίζονται σωστά. Για να εμφανίσουμε εικόνα bitmap στην οθόνη LCD του Nokia 5110 πρέπει να καλέσουμε τη συνάρτηση drawBitmap(). Χρειάζονται έξι παραμέτρους:. επάνω αριστερή γωνία Συντεταγμένη X, επάνω αριστερή γωνία Συντεταγμένη Y, πίνακας byte μονόχρωμου bitmap, πλάτος bitmap σε pixel, ύψος bitmap σε pixel και Χρώμα. Στο παράδειγμά μας, η εικόνα bitmap έχει μέγεθος 84×48. Έτσι, οι συντεταγμένες X & Y ορίζονται στο 0 ενώ το πλάτος και το ύψος ορίζονται σε 84 και 48.



```
/*  Marilyn Bmp Image */

#include <SPI.h>
#include <Adafruit_GFX.h>
#include <Adafruit_PCD8544.h>

Adafruit_PCD8544 display = Adafruit_PCD8544(7, 6, 5, 4, 3);

// 'Marilyn Monroe 84x48', 84x48px

const unsigned char MarilynMonroe [] PROGMEM = {
```

0x00, 0x00, 0x00, 0x7f, 0x00, 0x02, 0xfe, 0xf8, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xbe, 0x00,

0x00, 0x1f, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x20, 0x00, 0x00, 0x3f, 0x80, 0x00, 0x00,

0x00, 0x00, 0x00, 0x00, 0xf0, 0x00, 0x00, 0x1f, 0xe1, 0x80, 0x00, 0x00, 0x00, 0x00, 0x00, 0xc0,

0x00, 0x00, 0x0f, 0xf1, 0xc0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x0e, 0xd8, 0xe0,

0x00, 0x00, 0x00, 0x00, 0x00, 0x1f, 0x80, 0x00, 0x07, 0xe0, 0x70, 0x00, 0x00, 0x00, 0x00, 0x03,

0x3f, 0xe0, 0x00, 0x07, 0xf0, 0x78, 0x00, 0x00, 0x00, 0x00, 0x01, 0xe0, 0x70, 0x00, 0x0f, 0xee,

0x7c, 0x00, 0x00, 0x00, 0x00, 0x03, 0xc0, 0x00, 0x00, 0x0f, 0xf7, 0x1c, 0x00, 0x00, 0x00, 0x00,

0x07, 0x80, 0x00, 0x0f, 0xc7, 0xf3, 0x1e, 0x00, 0x00, 0x00, 0x00, 0x07, 0xc0, 0x00, 0x0f, 0xf3,

0xdf, 0x7f, 0x80, 0x00, 0x00, 0x00, 0x07, 0xfe, 0x00, 0x08, 0x7d, 0xef, 0xff, 0xc0, 0x00, 0x00,

0x00, 0x7f, 0xff, 0x80, 0x30, 0x0f, 0xfc, 0xe0, 0xc0, 0x00, 0x00, 0x01, 0x9e, 0x73, 0xc0, 0xe0,

0x07, 0xf8, 0xc1, 0xc0, 0x00, 0x00, 0x03, 0xfc, 0x00, 0x01, 0xc0, 0x0f, 0xfd, 0xe1, 0x80, 0x00,

0x00, 0x03, 0xf8, 0x00, 0x01, 0x9c, 0x0f, 0xff, 0xc1, 0xc0, 0x00, 0x00, 0x02, 0xc0, 0x00, 0x01,

0x9f, 0xbf, 0xfe, 0x01, 0x40, 0x00, 0x00, 0x02, 0x60, 0x00, 0x03, 0x07, 0xef, 0xff, 0x01, 0x40,

0x00, 0x00, 0x00, 0x60, 0x00, 0x07, 0x01, 0xf7, 0xff, 0x80, 0xc0, 0x00, 0x00, 0x00, 0x50, 0x01,

0xdf, 0x00, 0x7f, 0xff, 0x1c, 0x80, 0x00, 0x00, 0x00, 0x40, 0x01, 0xff, 0x00, 0x1f, 0xff, 0x1e,

0xe0, 0x00, 0x00, 0x02, 0x08, 0x00, 0x3f, 0x80, 0x07, 0xef, 0x03, 0xe0, 0x00, 0x00, 0x06, 0x08,

0x00, 0x03, 0xc0, 0x07, 0xdf, 0x07, 0xc0, 0x00, 0x00, 0x06, 0x08, 0x0f, 0x81, 0x80, 0x1f, 0xdf,

0x1f, 0x80, 0x00, 0x00, 0x03, 0x08, 0x1f, 0x98, 0x00, 0x3f, 0xfe, 0x19, 0x80, 0x00, 0x00, 0x18,

0x08, 0x3f, 0xfe, 0x00, 0x7f, 0xfe, 0x3f, 0x00, 0x00, 0x00, 0x08, 0x08, 0x30, 0x3f, 0x00, 0xff,

0xff, 0x3f, 0x00, 0x00, 0x00, 0x01, 0xe0, 0x76, 0x0f, 0x89, 0xff, 0xff, 0x9f, 0x00, 0x00, 0x00,

0x03, 0xe0, 0x7f, 0xc3, 0x81, 0xff, 0xfe, 0x9f, 0x80, 0x00, 0x00, 0x03, 0xf0, 0x7f, 0xf3, 0xc3,

0xff, 0xfe, 0x1f, 0x00, 0x00, 0x00, 0x03, 0xf0, 0x7f, 0xfd, 0xc3, 0xff, 0xfe, 0x5e, 0x00, 0x00,

0x00, 0x03, 0xf0, 0x7f, 0xff, 0xc3, 0xff, 0xf3, 0x1e, 0x00, 0x00, 0x00, 0x03, 0xf0, 0x71, 0xff,

0x87, 0xff, 0xe3, 0xff, 0x00, 0x00, 0x00, 0x07, 0xf0, 0x7c, 0x3f, 0x87, 0xff, 0xe3, 0xfe, 0x00,

0x00, 0x00, 0x0f, 0xf0, 0x3c, 0xff, 0x05, 0xff, 0xf3, 0xfc, 0x00, 0x00, 0x00, 0x0f, 0xf0, 0x0f,

0xfe, 0x09, 0xff, 0xf7, 0xfc, 0x00, 0x00, 0x00, 0x08, 0xf8, 0x01, 0xfc, 0x19, 0xff, 0xff, 0xf8,

0x00, 0x00, 0x00, 0x0c, 0x78, 0x00, 0x00, 0x13, 0xff, 0xff, 0xf8, 0x00, 0x00, 0x00, 0x0e, 0x78,

0x00, 0x00, 0x23, 0xff, 0xff, 0xf0, 0x00, 0x00, 0x00, 0x0e, 0xf8, 0x00, 0x00, 0x47, 0xff, 0xff,

0xf0, 0x00, 0x00, 0x00, 0x0c, 0xfa, 0x00, 0x01, 0x8f, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x00, 0x08,

0x7b, 0x00, 0x03, 0x3f, 0xff, 0xff, 0xe0, 0x00, 0x00, 0x00, 0x0c, 0x3b, 0xf8, 0x0f, 0xff, 0xff,

0xff, 0xe0, 0x00, 0x00, 0x00, 0x0f, 0xbb, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf0, 0x00, 0x00, 0x00,

0x07, 0xfb, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf0, 0x00, 0x00, 0x00, 0x00, 0x71, 0xff, 0xff, 0xff,

```
        0xff, 0xff, 0xe0, 0x00, 0x00, 0x00, 0x00, 0x41, 0xff, 0xff, 0xff, 0xff, 0xff, 0xe0, 0x00,
0x00
};
void setup()  {
        Serial.begin(9600);
        display.begin();
        display.setContrast(57);
        display.clearDisplay();
        // Display bitmap
        display.drawBitmap(0, 0,  MarilynMonroe, 84, 48, BLACK);
        display.display();
        // Invert Display
        //display.invertDisplay(1);
}
void loop() {}
```

Erasmus+ KA-202

Πρόγραμμα Στρατηγικής Σύμπραξης στην Επαγγελματική Εκπαίδευση και Κατάρτιση

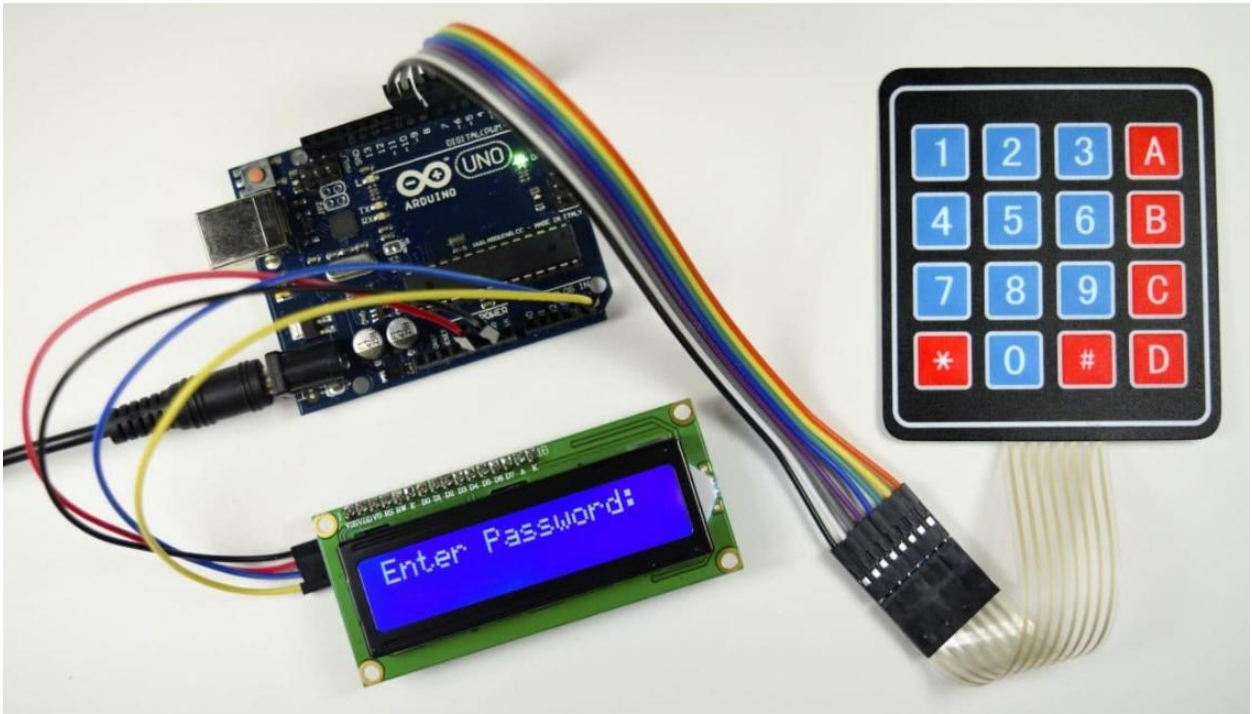
Τίτλος Προγράμματος: "Teaching and Learning Arduinos in Vocational Training"

Ακρονύμιο Προγράμματος: " ARDUinVET "

Κωδικός Προγράμματος: "2020-1-TR01-KA202-093762"

Erasmus+ KA-202

Πληκτρολογίο και εκπαιδευτικό κιτ



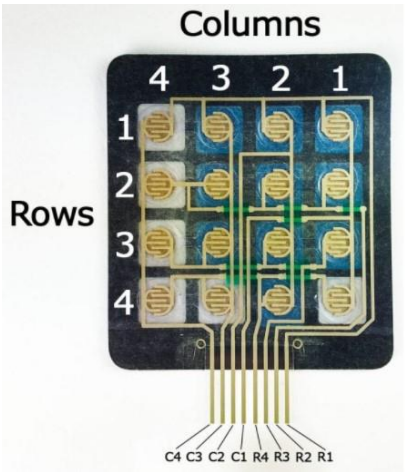
Τι είναι το πληκτρολόγιο;

Το πληκτρολόγιο είναι ένα σύστημα πλήκτρων διατεταγμένων σε μια μήτρα που λειτουργεί ως συσκευή μεταγωγής για παροχή μιας σύνδεσης μεταξύ μιας γραμμής και μιας στήλης.

Θα χρησιμοποιήσουμε πληκτρολόγιο μεμβράνης με μήτρα 4X4, γιατί είναι λεπτό και έχει συγκολλητική ενίσχυση, ώστε να μπορεί να κολληθεί στις περισσότερες επίπεδες επιφάνειες.

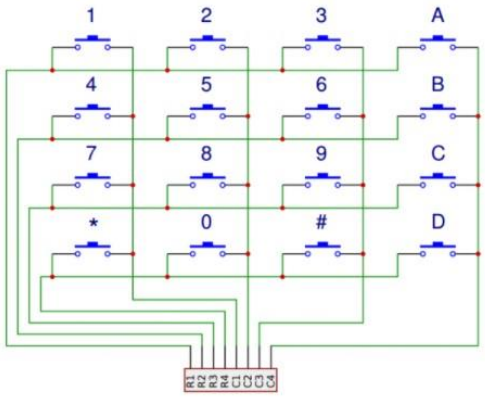
Κάτω από κάθε πλήκτρο υπάρχει ένας διακόπτης μεμβράνης. Κάθε διακόπτης στη σειρά συνδέεται με τους άλλους διακόπτες της ίδιας σειράς μέσω ενός αγωγού κάτω από το pad. Κάθε διακόπτης σε μια στήλη συνδέεται με τον ίδιο τρόπο - η μία πλευρά του διακόπτη συνδέεται με όλους τους άλλους διακόπτες της ίδιας στήλης μέσω αγωγού.

Κάθε γραμμή και στήλη μεταφέρεται σε μία μόνο ακίδα, για συνολικά 8 ακίδες σε ένα πληκτρολόγιο 4X4.



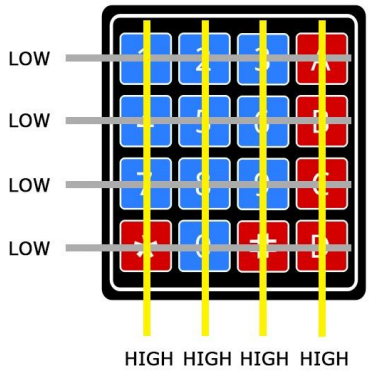
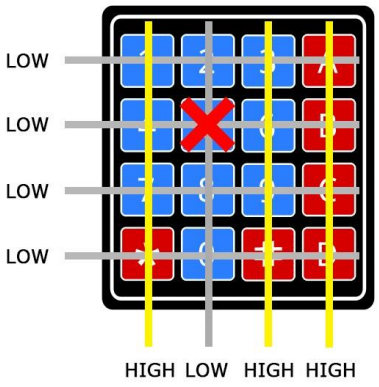
Πατώντας ένα πλήκτρο κλείνει ο διακόπτης μεταξύ μιας στήλης και μιας γραμμής, επιτρέποντας στο ρεύμα να ρέει μεταξύ μιας ακίδας στήλης και μιας ακίδας γραμμής.

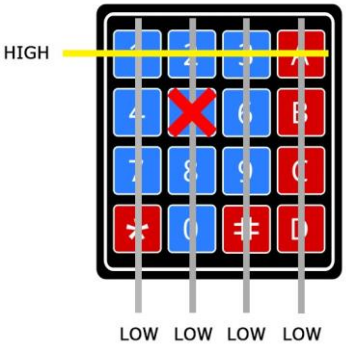
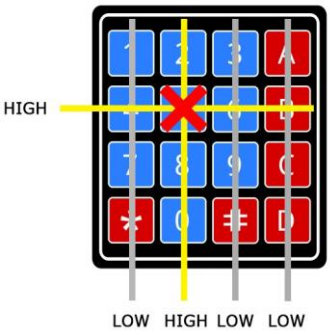
Το παρακάτω σχήμα δείχνει πώς συνδέονται οι σειρές και οι στήλες σε ένα πληκτρολόγιο 4X4:



Το Arduino ανιχνεύει ποιο πλήκτρο πατιέται ανιχνεύοντας την ακίδα γραμμής και στήλης που είναι συνδεδεμένη με το πλήκτρο.

Αυτό συμβαίνει σε τέσσερα βήματα:

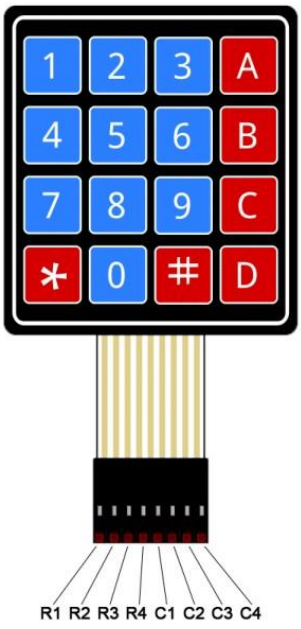
1. Όταν δεν πατιέται κανένα πλήκτρο, όλες οι ακίδες στηλών κρατιούνται σε ΥΨΗΛΟ δυναμικό και όλες οι ακίδες της σειράς κρατιούνται σε ΧΑΜΗΛΟ δυναμικό	
2. Όταν πατηθεί ένα πλήκτρο, η ακίδα της στήλης πέφτει σε ΧΑΜΗΛΟ δυναμικό, καθώς το ρεύμα από τη στήλη HIGH ρέει στην ακίδα της σειράς LOW:	

3. Το Arduino γνωρίζει τώρα σε ποια στήλη βρίσκεται το πλήκτρο, οπότε τώρα χρειάζεται απλώς να βρει και τη σειρά στην οποία βρίσκεται. Αυτό το κάνει περνώντας διαδοχικά κάθε γραμμή σε ΥΨΗΛΟ δυναμικό διαβάζοντας κάθε φορά τις ακίδες των στηλών μέχρι να ανιχνευτεί η ακίδα στην οποία επιστρέφεται το ΥΨΗΛΟ αυτό δυναμικό.	
4. Όταν η ακίδα της στήλης περνάει ξανά σε ΥΨΗΛΟ δυναμικό, ο Arduino έχει βρει την ακίδα της γραμμής που είναι συνδεδεμένη στο πλήκτρο:	

Από το παραπάνω διάγραμμα, μπορείτε να δείτε ότι ο συνδυασμός της σειράς 2 και της στήλης 2 θα μπορούσε να σημαίνει ότι πατήθηκε μόνο το κουμπί με τον αριθμό 5.

ΣΥΝΔΕΣΤΕ ΤΟ ΠΛΗΚΤΡΟΛΟΓΙΟ ΣΤΟΝ ARDUINO

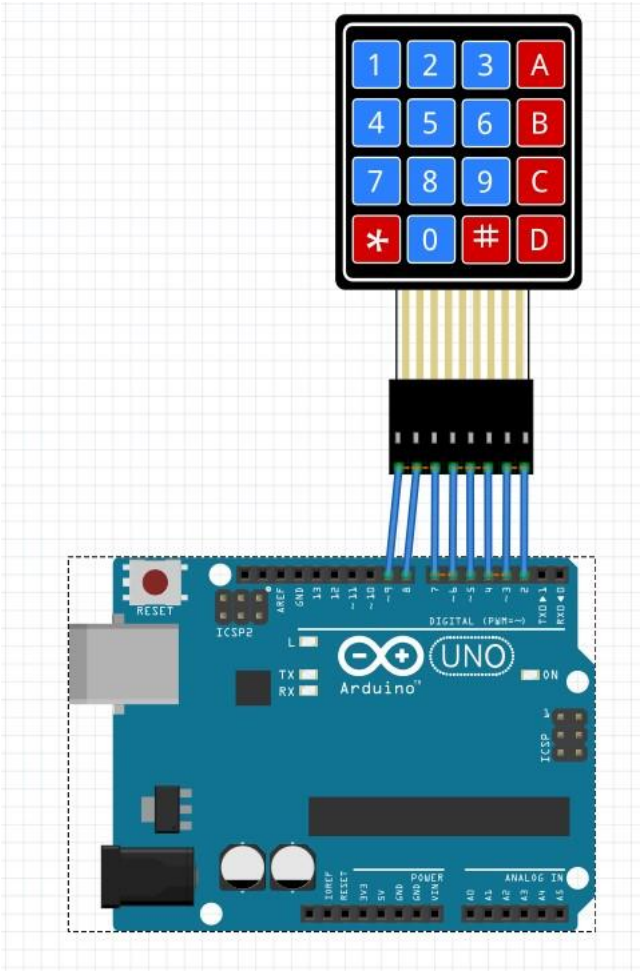
Η διάταξη των ακροδεκτών για τα περισσότερα πληκτρολόγια μεμβράνης θα μοιάζει με αυτό:



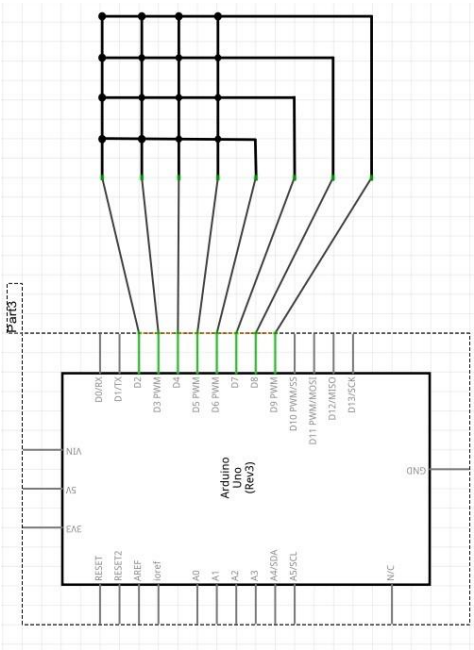
Κύκλωμα 1

Τίτλος Κυκλώματος: Η σειριακή οθόνη εμφανίζει το πατημένο πλήκτρο

Περιγραφή Κυκλώματος: Το πρόγραμμα θα μας δείξει πώς να εμφανίζουμε κάθε πλήκτρο που πατιέται στη σειριακή οθόνη.



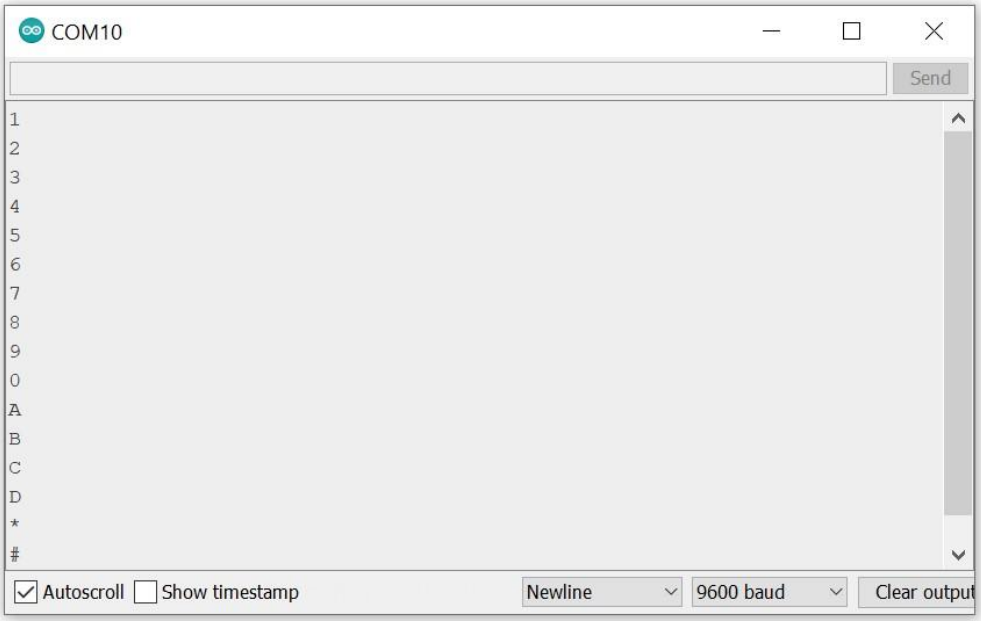
1-) Όψη πλακέτας δοκιμών



2-) Σχηματικό διάγραμμα

```
/*    "Η σειριακή οθόνη εμφανίζει το πλήκτρο που πατήθηκε"   */
#include <Keypad.h>
const byte ROWS = 4;
const byte COLS = 4;
char hexaKeys[ROWS][COLS] = {
  {'1', '2', '3', 'A'},
  {'4', '5', '6', 'B'},
  {'7', '8', '9', 'C'},
  {'*', '0', '#', 'D'}
};
```

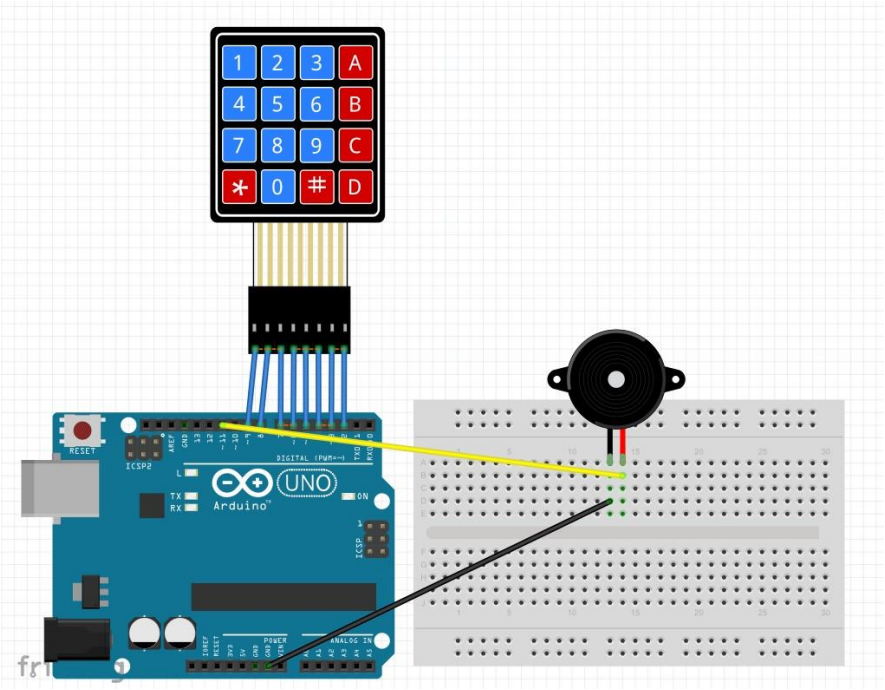
```
byte rowPins[ROWS] = {9, 8, 7, 6};
byte colPins[COLS] = {5, 4, 3, 2};
Keypad customKeypad = Keypad(makeKeymap(hexaKeys), rowPins, colPins, ROWS, COLS);
void setup() {
    Serial.begin(9600);
}
void loop() {
    char customKey = customKeypad.getKey();
    if (customKey){
        Serial.println(customKey);
    }
}
```



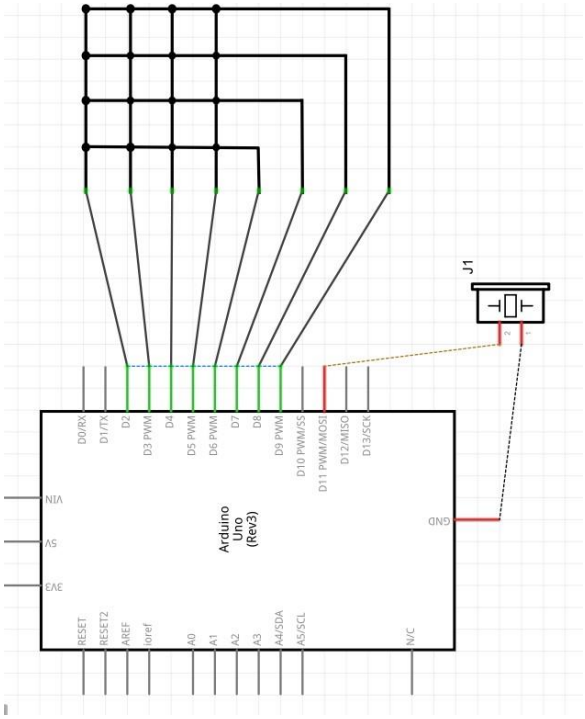
Κύκλωμα 2

Τίτλος Κυκλώματος: Ήχος πληκτρολογίου με Arduino

Περιγραφή Κυκλώματος: Όταν πατηθεί ένα πλήκτρο στο πληκτρολόγιο, ηχεί ο πιεζοηλεκτρικός βομβητής



1-) Όψη πλακέτας δοκιμών



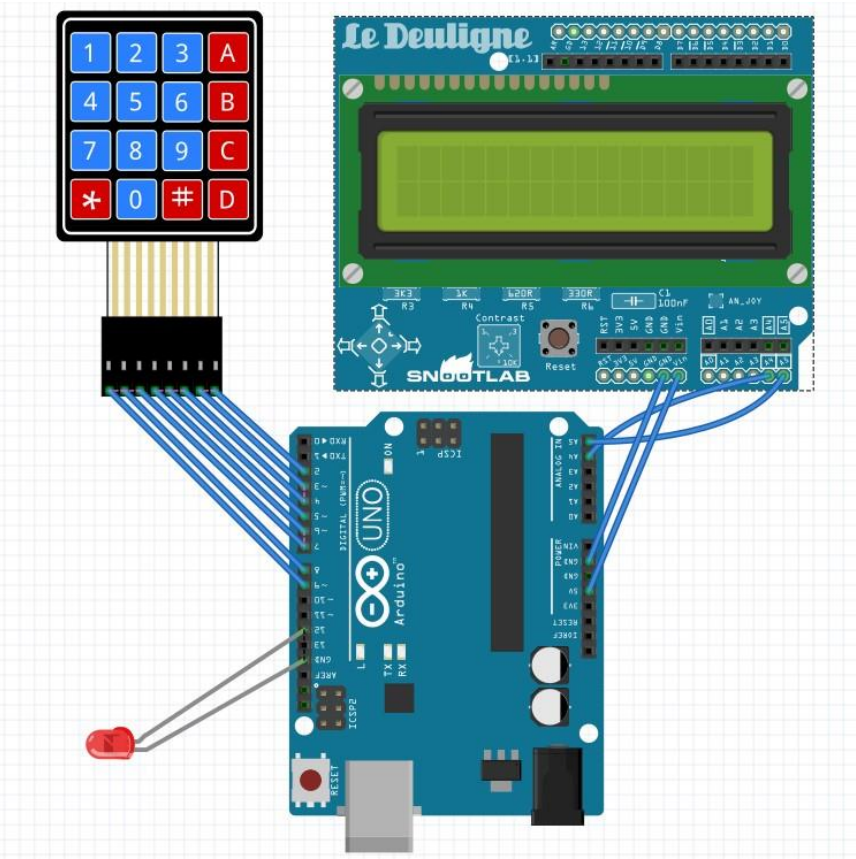
2-) Σχεδιάγραμμα

```
/*     "Ηχος πληκτρολογίου με Arduino"   */
#include <Keypad.h>
#include <ezBuzzer.h>
const int BUZZER_PIN = 11;
const int ROW_NUM    = 4; // τέσσερις σειρές
const int COLUMN_NUM = 4; // τέσσερις στήλες
 keys[ROW_NUM][COLUMN_NUM] = {
    {'1', '2', '3', 'A'},
    {'4', '5', '6', 'B'},
    {'7', '8', '9', 'C'},
    {'*', '0', '#', 'D'}
};
byte pin_rows[ROW_NUM] = {9, 8, 7, 6};      /* συνδέστε τους ακροδέκτες σειρών του
* πληκτρολογίου*/
byte pin_column[COLUMN_NUM] = {5, 4, 3, 2}; /* συνδέστε τους ακροδέκτες των          *
στηλών του πληκτρολογίου*/
Keypad keypad = Keypad(makeKeymap(keys), pin_rows, pin_column, ROW_NUM,
COLUMN_NUM );
ezBuzzer buzzer(BUZZER_PIN); /* δημιουργία αντικειμένου ezBuzzer που συνδέεται σε  *
έναν ακροδέκτη.*/
void setup() {
    Serial.begin(9600);
}
void loop(){
    buzzer.loop();      /*μέσα στην loop() ΠΡΕΠΕΙ να καλέσετε την συνάρτηση
    * buzzer.loop() */
    char key = keypad.getKey();
    if (key) {
        Serial.print(key); // τυπώνει το κλειδί στη σειριακή οθόνη
        buzzer.beep(100);  // παράγει ένα ηχητικό σήμα για 100ms
    }
 }
```

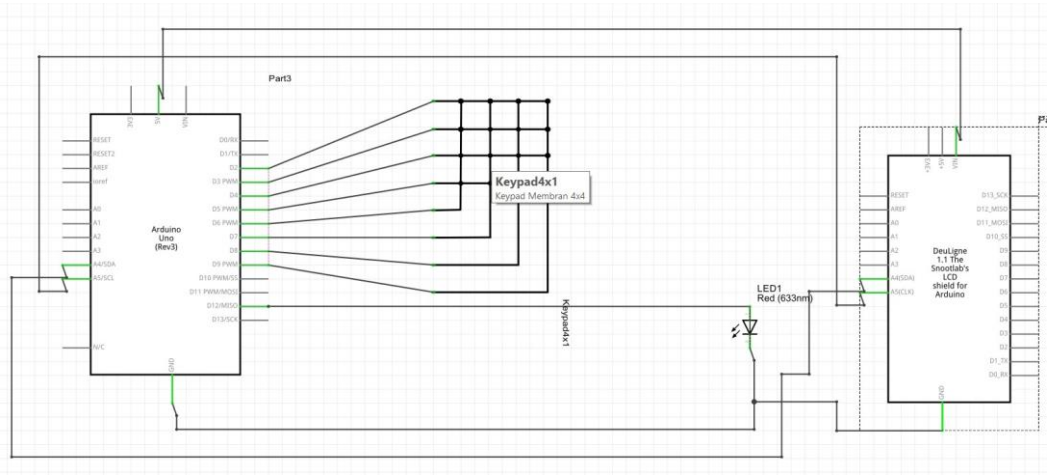
Κύκλωμα 3

Τίτλος Κυκλώματος: Κωδικός Ξεκλειδώματος σε Arduino

Περιγραφή Κυκλώματος: Πρόγραμμα που ρυθμίζει ένα πληκτρολόγιο στον Arduino. Ένα LED θα ανάψει όταν πληκτρολογήσετε τον σωστό κωδικό



1-) Όψη πλακέτας δοκιμών



2-) Σχηματικό διάγραμμα

```
/* Κωδικός Ξεκλειδώματος σε Arduino */
#include <Keypad.h> //Μπορείτε να κατεβάσετε Βιβλιοθήκες μέσω Arduino IDE
#include <Wire.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
#define Solenoid 12          /* Στην πραγματικότητα η πύλη του τρανζίστορ που ελέγχει
* ένα ηλεκτρομαγνητικό εξάρτημα(πηνίο)*/
//Στην περίπτωσή μου χρησιμοποιώ ένα απλό LED
#include <liquidcrystal_i2c.h></liquidcrystal_i2c.h></lcd.h></wire.h></keypad.h>
#define I2C_ADDR 0x27     // LCD i2c Address and pins
#define BACKLIGHT_PIN 3
#define En_pin 2
#define Rw_pin 1
#define Rs_pin 0
#define D4_pin 4
#define D5_pin 5
#define D6_pin 6
#define D7_pin 7
LiquidCrystal_I2C
lcd(I2C_ADDR,En_pin,Rw_pin,Rs_pin,D4_pin,D5_pin,D6_pin,D7_pin);
const byte numRows= 4; // αριθμός σειρών στο πληκτρολόγιο
const byte numCols= 4; // αριθμός στηλών στο πληκτρολόγιο
int code = 1234;  // Εδώ είναι ο κωδικός
int tot,i1,i2,i3,i4;
char c1,c2,c3,c4;
/* Το keymap ορίζει το πλήκτρο που πατιέται σύμφωνα με τη γραμμή και τις στήλες όπως
* ακριβώς εμφανίζεται στο πληκτρολόγιο*/
char keymap[numRows][numCols]= {
   {'1', '2', '3', 'A'},
   {'4', '5', '6', 'B'},
   {'7', '8', '9', 'C'},
   {'*', '0', '#', 'D'}
};
// Κώδικας που δείχνει τις συνδέσεις του πληκτρολογίου στους ακροδέκτες του arduino
byte rowPins[numRows] = {9,8,7,6}; // Γραμμές από 0 έως 3
```

```
 byte colPins[numCols]= {5,4,3,2}; // Στήλες από 0 έως 3
// Αρχικοποιεί ένα στιγμιότυπο της κλάσης Keypad
Keypad myKeypad= Keypad(makeKeymap(keymap), rowPins, colPins, numRows,
numCols);
void setup(){
    lcd.begin (16,2);
    lcd.setBacklightPin(BACKLIGHT_PIN,POSITIVE);
    lcd.setBacklight(HIGH);
    lcd.home ();
    lcd.print("ROMANIA DoorLock");
    lcd.setCursor(9, 1);
    lcd.print("Standby");
    pinMode(Solenoid,OUTPUT);
    delay(2000);
}
void loop(){
    char keypressed = myKeypad.getKey();  /*Η συνάρτηση getKey δεν επιτρέπει να
    τρέξει το πρόγραμμα ξεκλειδώματος έως ότου πατηθεί το πλήκτρο "*"  */
    if (keypressed == '*') {          /* Και μπορείτε να χρησιμοποιείτε απλά τον
    * υπόλοιπο κώδικά σας*/
        lcd.clear();
        lcd.setCursor(0, 0);
        lcd.print("Enter Code");       /* όταν πατήσετε το πλήκτρο "*" μπορείτε να
        * εισάγετε τον κωδικό πρόσβασης*/
        keypressed = myKeypad.waitForKey();              /* εδώ όλα τα προγράμματα
        * σταματούν μέχρι να εισάγετε τα τέσσερα ψηφία και στη συνέχεια συγκρίνεται με
        * τον παραπάνω κωδικό*/
        if (keypressed != NO_KEY) {
              c1 = keypressed;
              lcd.setCursor(0, 1);
              lcd.print("*");
        }
        keypressed = myKeypad.waitForKey();
        if (keypressed != NO_KEY) {
            c2 = keypressed; lcd.setCursor(1, 1);
            lcd.print("*");
```

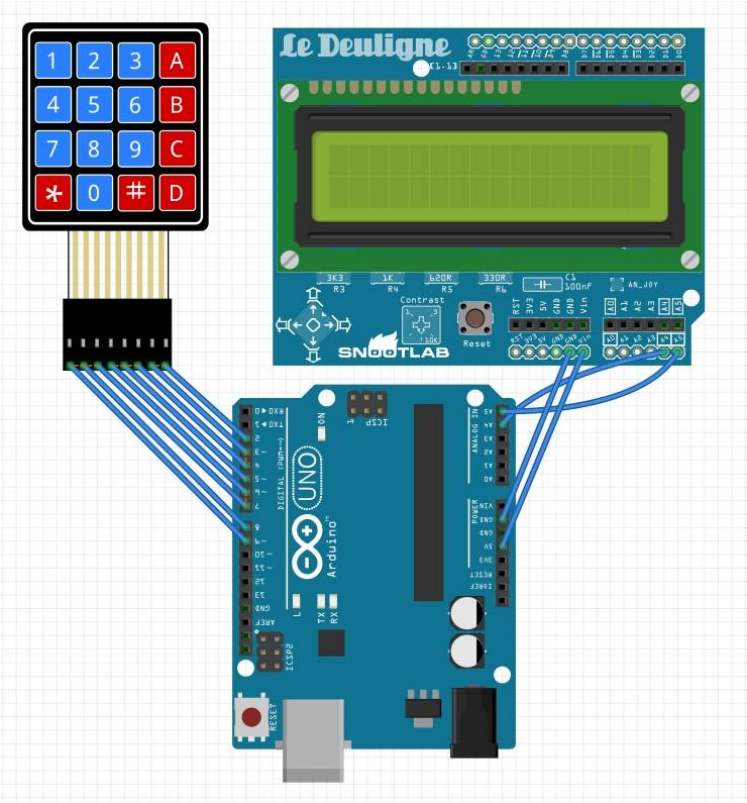
```
}
keypressed = myKeypad.waitForKey();
if (keypressed != NO_KEY) {
    c3 = keypressed;
    lcd.setCursor(2, 1);
    lcd.print("*");
}
keypressed = myKeypad.waitForKey();
if (keypressed != NO_KEY) {
    c4 = keypressed;
    lcd.setCursor(3, 1);
    lcd.print("*");
}
i1=(c1-48)*1000;        /* Τα πλήκτρα που πατάτε αποθηκεύονται ως χαρακτήρες τους
* οποίους για να τους μετατρέψω σε ακέραιους αριθμούς κάνω κάποιες πράξεις  και
* παίρνω τον κωδικό ως μεταβλητή ακεραίου xxxx.
*/
i2=(c2-48)*100;
i3=(c3-48)*10;
i4=c4-48;
tot=i1+i2+i3+i4;
if (tot == code)            /* Αν ο κωδικός είναι σωστός οτιδήποτε χρειαστείς                *
εκτυπώνεται στην οθόνη σε μήνυμα */
{
    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Welcome");
    digitalWrite(Solenoid,HIGH);
    delay(3000);
    digitalWrite(Solenoid,LOW);
    lcd.setCursor(7,1);
    lcd.print("ROMANIA DoorLock");
    delay(3000);
    lcd.clear();
    lcd.print("ROMANIA DoorLock");
```

```
        lcd.setCursor(9, 1);
        lcd.print("Standby");
    }
        else                    //Αν ο κωδικός είναι λάθος λαμβάνετε κάτι άλλο
        {
            lcd.clear();
            lcd.setCursor(0, 0);
            lcd.print("WRONG CODE");
            delay(3000);
            lcd.clear();
            lcd.print("ROMANIA DoorLock");
            lcd.setCursor(9, 1);
            lcd.print("Standby");
        }
    }
}
```

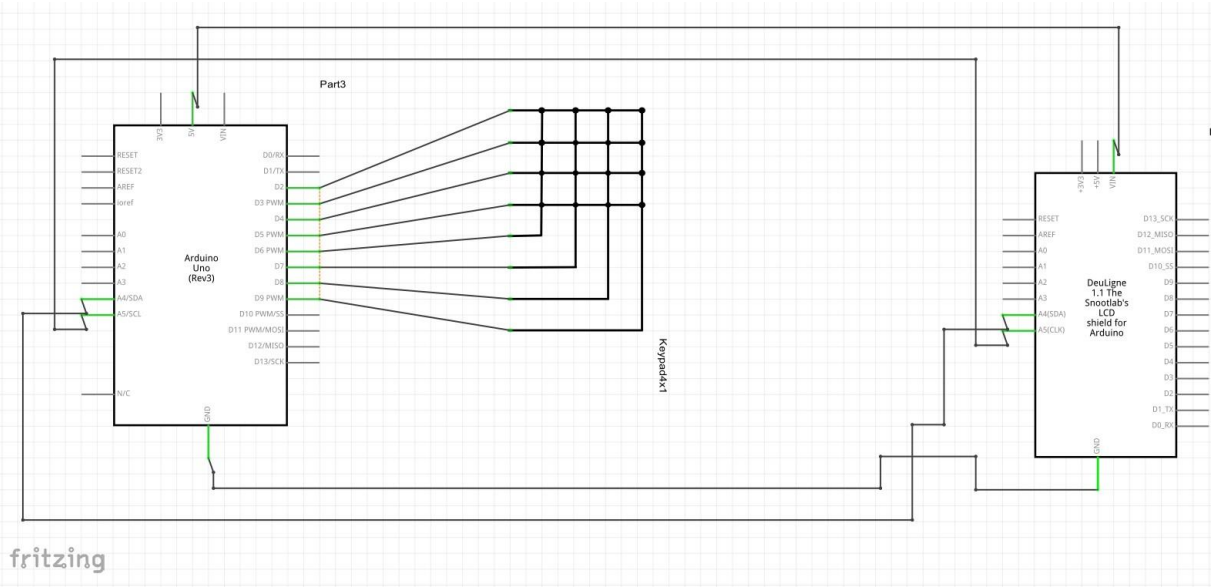
Κύκλωμα 4

Τίτλος Κυκλώματος: Αριθμομηχανή Arduino

Περιγραφή Κυκλώματος: Το πρόγραμμα κάνει βασικές μαθηματικές πράξεις



1-) Όψη πλακέτας δοκιμών



2-) Σχηματικό διάγραμμα

```
/* Αριθμομηχανή Arduino */
#include <Keypad.h>
#include <EEPROM.h>
#include <LCD.h>
#include <LiquidCrystal_I2C.h>
#define I2C_ADDR 0x27                    // Διεύθυνση I2C
#define BACKLIGHT_PIN 3                  // Δήλωση ακίδων LCD
#define En_pin 2
#define Rw_pin 1
#define Rs_pin 0
#define D4_pin 4
#define D5_pin 5
#define D6_pin 6
#define D7_pin 7
const byte ROWS = 4;                     // Τέσσερις γραμμές
const byte COLS = 4;                     // Τέσσερις στήλες
// Οριστικοποίηση του χάρτη κλειδιών
char keys[ROWS][COLS] = {
   {'1','2','3','A'},
   {'4','5','6','B'},
   {'7','8','9','C'},
   {'*','0','#','D'}
};
byte rowPins[ROWS] = {9,8,7,6};          //Γραμμές από 0 έως 3
byte colPins[COLS]= {5,4,3,2};           //Στήλες από 0 έως 3
LiquidCrystal_I2C
lcd(I2C_ADDR,En_pin,Rw_pin,Rs_pin,D4_pin,D5_pin,D6_pin,D7_pin);
// Δημιουργία του πληκτρολογίου
Keypad kpd = Keypad( makeKeymap(keys), rowPins, colPins, ROWS, COLS );
long Num1,Num2,Number;
char key,action;
boolean result = false;
void setup(){
   lcd.begin (16,2);
   lcd.setBacklightPin(BACKLIGHT_PIN,POSITIVE);
```

```
    lcd.setBacklight(HIGH);          // Οπίσθιος φωτισμός
    lcd.print("Calculator Ready");          // Εμφάνιση εισαγωγικού μηνύματος
    lcd.setCursor(0, 1);                 //  Τοποθέτηση του κέρσορα στην στήλη 0, γραμμή 1
    lcd.print("A=+ B=- C=* D=/"); // Εμφάνιση εισαγωγικού μηνύματος
    delay(6000);                         //  Καθυστέρηση  για εμφάνιση πληροφοριών στην οθόνη
    lcd.clear();                         // Μετά καθάρισε την οθόνη
// for(i=0 ; i<sizeof(code);i++){    /* Την πρώτη φορά που θα φορτώσετε τον κώδικα
* κρατήστε αυτή την εντολή ως σχόλιο */
    //EEPROM.get(i, code[i]);         /* Ενημερώνεται ο κώδικας και αποθηκεύονται οι
    * αλλαγές στην EEPROM */
    //          }                          /* Στη συνέχεια αποδεσμεύστε  την εντολή επανάληψης
    * από το σύμβολο του σχολίου  και ξανατρέξτε τον κώδικα( Να γίνει μόνο μία φορά) */
}
void loop() {
    key = kpd.getKey();                 /* Αποθήκευση της τιμής του πατημένου πλήκτρου σε
* μία μεταβλητή τύπου char(χαρακτήρα)*/
    if (key!=NO_KEY)
        DetectButtons();
    if (result==true)
        CalculateResult();
    DisplayResult();
}
void DetectButtons(){
    lcd.clear();                         // Μετά καθάρισε την οθόνη
    if (key=='*'){                       //Αν πατηθεί το πλήκτρο ακύρωσης
      Serial.println ("Button Cancel");
      Number=Num1=Num2=0;
      result=false;
    }
    if (key == '1') {                    // Αν πατηθεί το πλήκτρο 1
      Serial.println ("Button 1");
      if (Number==0)
        Number=1;
    else
        Number = (Number*10) + 1;                //Πατήθηκε δύο φορές
```

```
}
if (key == '4') {                          //Αν πατηθεί το πλήκτρο 4
      Serial.println ("Button 4");
    if (Number==0)
       Number=4;
    else
       Number = (Number*10) + 4;           //Πατήθηκε δύο φορές
}
if (key == '7') {                           //Αν πατηθεί το πλήκτρο 7
    Serial.println ("Button 7");
    if (Number==0)
       Number=7;
    else
       Number = (Number*10) + 7;           //Πατήθηκε δύο φορές
}
if (key == '0'){
    Serial.println ("Button 0");           //Αν πατηθεί το πλήκτρο 0
    if (Number==0)
        Number=0;
    else
       Number = (Number*10) + 0;           //Πατήθηκε δύο φορές
}
if (key == '2') {                           //Αν πατηθεί το πλήκτρο 2
    Serial.println ("Button 2");
    if (Number==0)
       Number=2;
    else
       Number = (Number*10) + 2;           //Πατήθηκε δύο φορές
}
if (key == '5') {
    Serial.println ("Button 5");
     if (Number==0)
       Number=5;
    else
```

```
        Number = (Number*10) + 5;              //Πατήθηκε δύο φορές
}
if (key == '8') {
    Serial.println ("Button 8");
    if (Number==0)
        Number=8;
    else
        Number = (Number*10) + 8;              //Πατήθηκε δύο φορές
}
if (key == '#') {
    Serial.println ("Button Equal");
    Num2=Number;
    result = true;
}
 if (key == '3'){
    Serial.println ("Button 3");
    if (Number==0)
        Number=3;
    else
        Number = (Number*10) + 3;              //Πατήθηκε δύο φορές
}
if (key == '6') {
    Serial.println ("Button 6");
    if (Number==0)
        Number=6;
    else
        Number = (Number*10) + 6;              //Πατήθηκε δύο φορές
}
if (key == '9') {
    Serial.println ("Button 9");
    if (Number==0)
        Number=9;
    else
        Number = (Number*10) + 9;                      //Πατήθηκε δύο φορές
```

```
  }
  if (key == 'A' || key == 'B' || key == 'C' || key == 'D') {   /*Ανιχνεύονται τα πλήκτρα
* της στήλης 4*/
      Num1 = Number;
      Number =0;
      if (key == 'A') {
          Serial.println ("Addition");
          action = '+';
      }
      if (key == 'B') {
          Serial.println ("Subtraction");
          action = '-';
      }
      if (key == 'C') {
          Serial.println ("Multiplication");
          action = '*';
      }
      if (key == 'D') {
          Serial.println ("Division");
          action = '/';
      }
      delay(100);
    }
 }
void CalculateResult() {
    if (action=='+')
        Number = Num1+Num2;
    if (action=='-')
        Number = Num1-Num2;
    if (action=='*')
        Number = Num1*Num2;
    if (action=='/')
    Number = Num1/Num2;
}
void DisplayResult() {
```

```
    lcd.setCursor(0, 0);        // ρύθμιση του κέρσορα στην στήλη 0, γραμμή 1
    lcd.print(Num1);
    lcd.print(action);
    lcd.print(Num2);
    if (result==true) {
        lcd.print(" =");
        lcd.print(Number);      //Εμφάνιση του αποτελέσματος
    }
    lcd.setCursor(0, 1);        // ρύθμιση του κέρσορα στην στήλη 0, γραμμή 1
    lcd.print(Number);          // Εμφάνιση του αποτελέσματος
}
```

Erasmus+ KA-202

Πρόγραμμα Στρατηγικής Σύμπραξης στην Επαγγελματική Εκπαίδευση και Κατάρτιση

Τίτλος Προγράμματος: "Teaching and Learning Arduinos in Vocational Training"

Ακρωνύμιο Προγράμματος: " ARDUinVET "

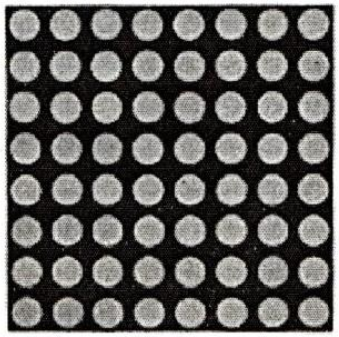
Κωδικός Προγράμματος: "2020-1-TR01-KA202-093762"

Οθόνη Dot Matrix και εκπαιδευτικό κιτ

Τι είναι το Dot Matrix Display;

Η οθόνη Dot Matrix είναι μια ομάδα LED σε ένα συγκεκριμένη διάταξη. Μια τυπική συστοιχία LED αποτελείται από 5x7 ή 8x8 LED διατεταγμένα σε σειρές και στήλες όπως φαίνεται στο παρακάτω σχήμα. Μια οθόνη Dot Matrix 5*7 έχει 5 σειρές και 7 στήλες LED και μια οθόνη Dot Matrix 8*8 έχει 8 σειρές και 8 στήλες LED συνδεδεμένες μεταξύ τους.

5*7 DOT Matrix Display



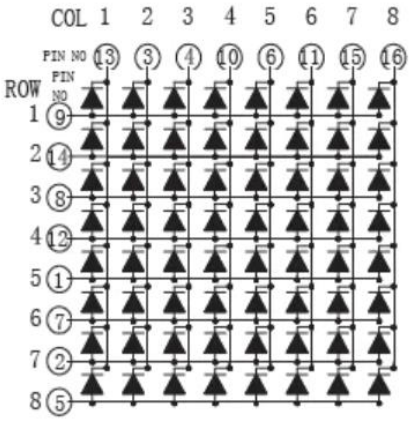
8*8 DOT Matrix Display

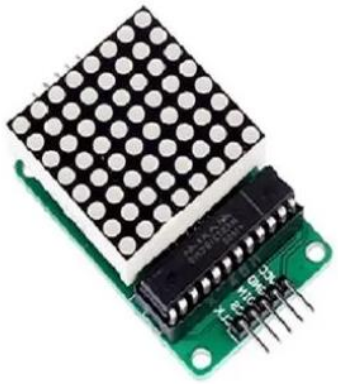
Αν δούμε ένα κομμάτι του πίνακα 8x8 κουκκίδων, περιέχει 16 ακίδες στις οποίες 8 ακίδες χρησιμοποιούνται για γραμμές και 8 για στήλες. Αυτό σημαίνει σε γραμμές και στήλες συνολικά 64 LED.

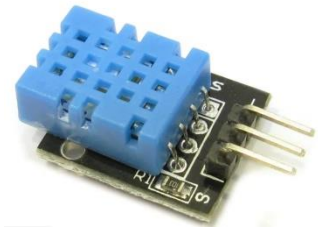
Ξεκινάμε από την ακίδα # 1 έως την ακίδα # 8. Ο αριθμός ακίδας 1 είναι R5 (Σειρά-5) και ο αριθμός ακίδας 8 είναι R3 (Σειρά-3) στην κάτω πλευρά.

Στην επάνω πλευρά βρίσκεται η ακίδα 9 (Σειρά-1) έως την ακίδα 16 (στήλη-1).

Ένας αρχάριος πάντα μπερδεύεται και ξεκινά από το μηδέν. Βλέποντας την εικόνα/διάγραμμα, συχνά χρησιμοποιούμε κάποια πηγή. Επίσης πρέπει να ξεχωρίσουμε τη πηγή +VE και -VE. Ένας έμπειρος μπορεί να καταλάβει από τον κοινό τύπο καθόδου/ανόδου.



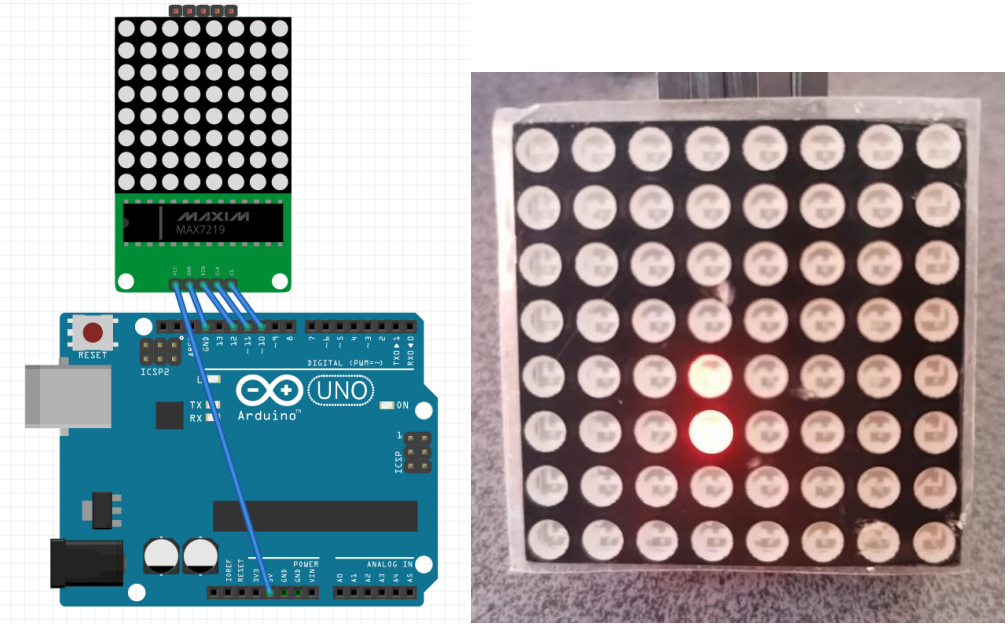
Η οθόνη MAX7219 LED Dot matrix είναι μία από τις δημοφιλείς που διατίθενται στην αγορά και χρησιμοποιείται από φοιτητές, χομπίστες ηλεκτρονικών ειδών, ειδικά όμως σε βιομηχανικές εφαρμογές οθονών. Υπάρχουν δύο τύποι κυκλωμάτων που είναι γενικά διαθέσιμοι. Αυτά είναι τα καθολικά (generic) κυκλώματα. Κάθε κύκλωμα αποτελείται από δύο μονάδες. Το 8X8 LED dot matrix και το MAX7219 IC. Το **MAX7219** είναι ένα IC προγράμματος οδήγησης κοινής καθόδου με σειριακές εισόδους και ακίδες εξόδου. Διαθέτει δυνατότητα ρύθμισης ρεύματος που μπορεί να ρυθμιστεί χρησιμοποιώντας μόνο μία εξωτερική αντίσταση. Επιπλέον, έχει μια σειριακή διεπαφή τεσσάρων καλωδίων που μπορεί εύκολα να συνδεθεί με όλους τους μικροεπεξεργαστές. Μπορεί να ελέγξει 64 μεμονωμένα LED συνδεδεμένα στις ακίδες εξόδου του χρησιμοποιώντας μόνο 4 καλώδια χρησιμοποιώντας το Arduino. Επιπλέον, μπορεί να δημιουργήσει απεικονίσεις dot matrix, απεικονίσεις 7 τμημάτων και γραφήματα ράβδων. Επιπλέον, το MAX7219 διαθέτει ενσωματωμένο αποκωδικοποιητή BCD που το καθιστά εύκολο στη χρήση με αριθμητικές απεικονίσεις επτά τμημάτων. Επιπλέον, διαθέτει στατική RAM 8×8 που μπορούμε να χρησιμοποιήσουμε για την αποθήκευση αριθμών. Είναι ένα από τα πιο δημοφιλή IC προγραμμάτων οδήγησης οθόνης.	

	Ο DHT11 είναι ένας ψηφιακός αισθητήρας θερμοκρασίας και υγρασίας. Χρησιμοποιεί έναν χωρητικό αισθητήρα υγρασίας και ένα θερμίστορ για τη μέτρηση του περιβάλλοντος αέρα και εκπέμπει ένα ψηφιακό σήμα στην ακίδα δεδομένων (δεν

	απαιτούνται αναλογικές ακίδες εισόδου).

Κύκλωμα 1:

Τίτλος κυκλώματος: Εμφάνιση όλων των LED ένα προς ένα
Περιγραφή κυκλώματος: Το Dot Matrix εμφανίζει όλα τα LED ένα προς ένα.



1-) Breadboard view

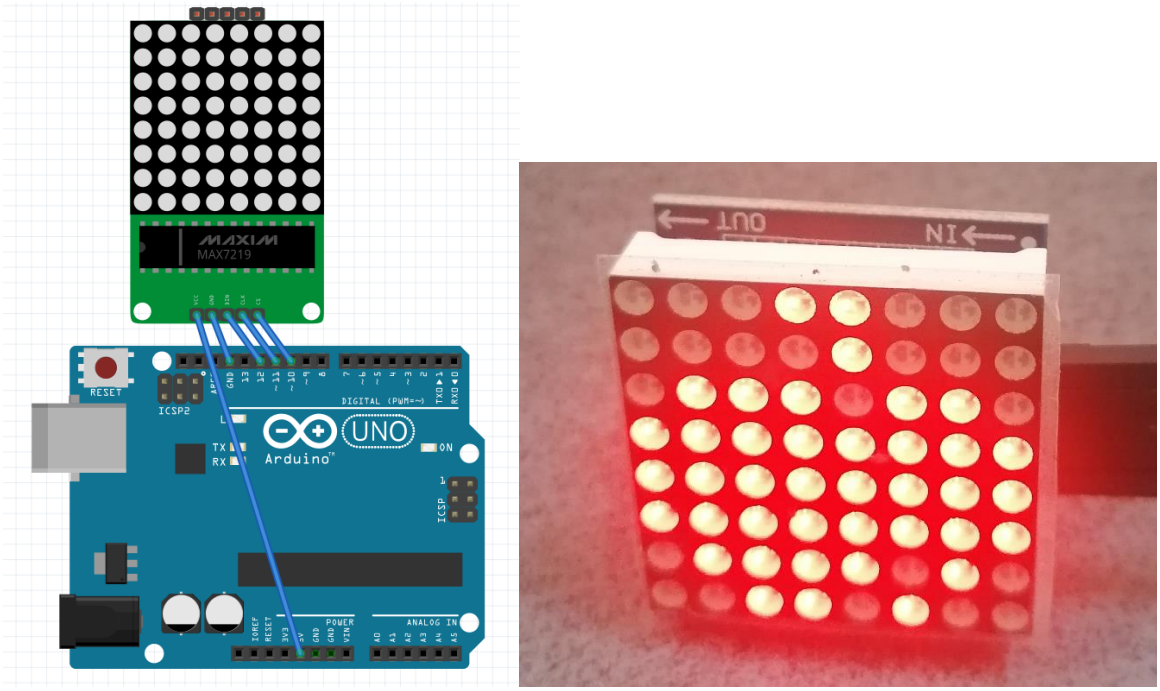
```
/*    Display all LEDs one by one  */

#include <LedControl.h>
int DIN = 12;
int CS = 10;
int CLK = 11;
LedControl lc=LedControl(DIN, CLK, CS,0);
void setup() {
  lc.shutdown(0,false);
  lc.setIntensity(0,0);
  lc.clearDisplay(0);}
void loop() {
  for(int j=0;j<8;j++){
     for(int i=0;i<8;i++){
       lc.setLed(0,j,i,true);
```

```
delay(100);
lc.setLed(0,j,i,false); } }}
```

Κύκλωμα 2:

Τίτλος κυκλώματος: Εμφάνιση του εικονιδίου Apple

Περιγραφή κυκλώματος: Η διεπαφή Dot Matrix εμφανίζει το εικονίδιο της Apple



1-) Breadboard view

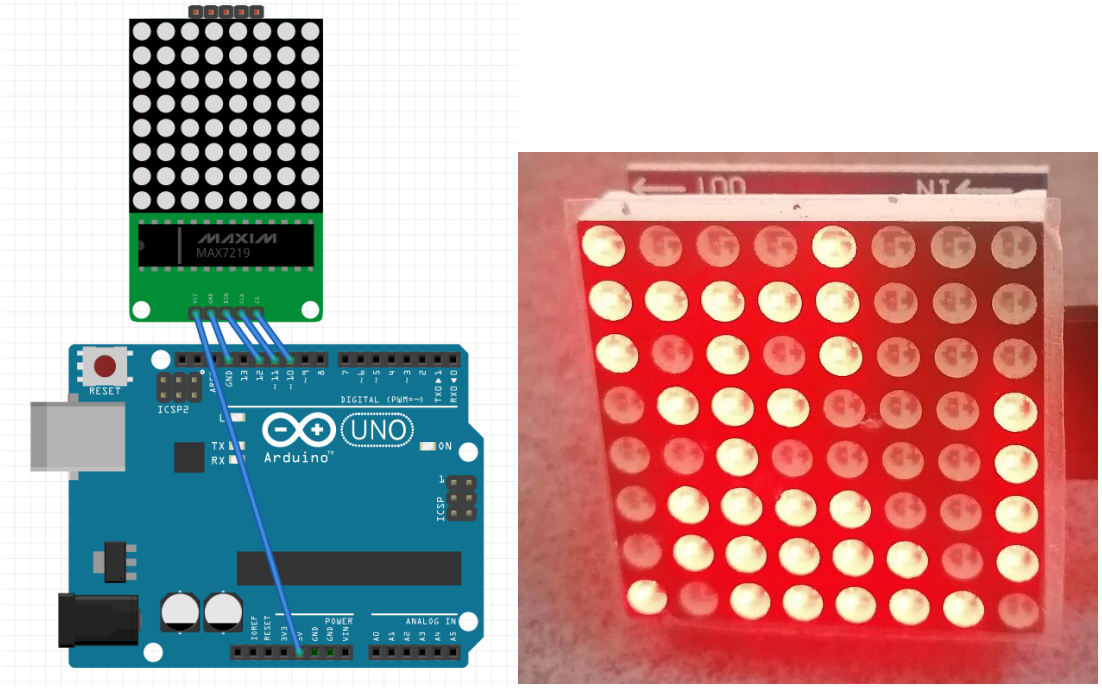
```
/*  Display the Apple icon*/
#include <LedControl.h>
int DIN = 12;
int CS = 10;
int CLK = 11;
LedControl lc=LedControl(DIN, CLK, CS,0);
byte Apple
[8]={B00011000,B00001000,B01110110,B11111111,B11111111,B11111111,B01111010,B001
10100};
void setup() {
  lc.shutdown(0,false);
```

```
 lc.setIntensity(0,0);
 lc.clearDisplay(0);  }
 void loop(){
 for(int i=0;i<8;i++) lc.setRow(0,i,Apple[i]);   }
```

Κύκλωμα 3:

Τίτλος κυκλώματος: Εμφάνιση εικονιδίου γάτας

Περιγραφή κυκλώματος: Η διεπαφή Dot Matrix εμφανίζει ένα εικονίδιο γάτας



1-) Breadboard view

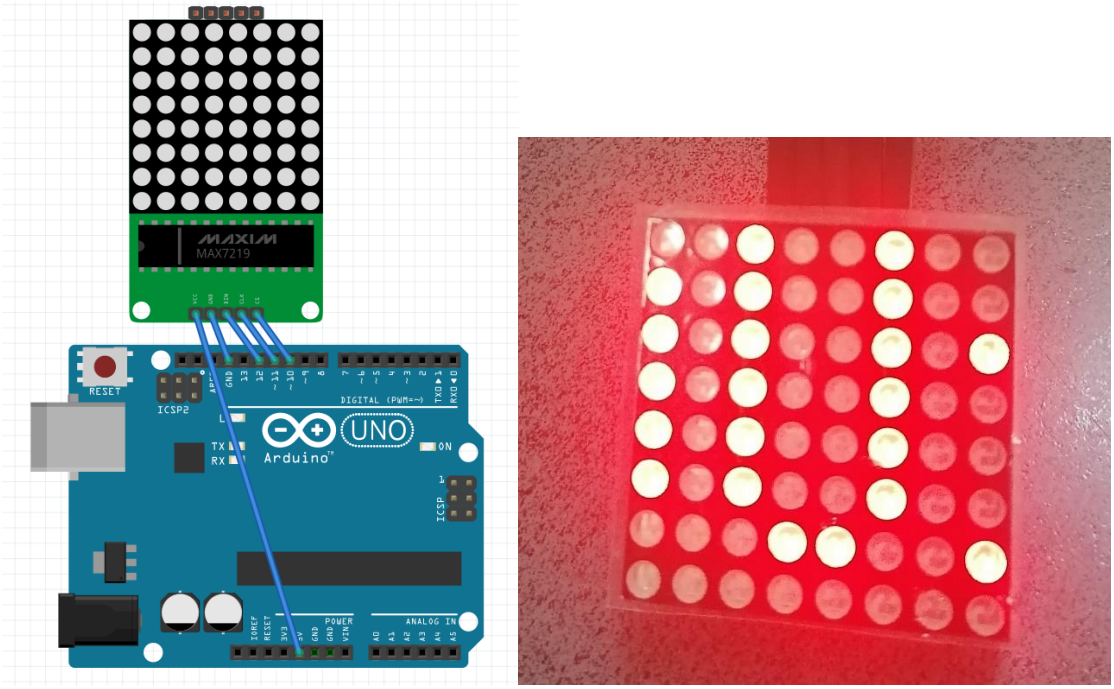
```
/*  Display a cat icon*/
#include <LedControl.h>
int DIN = 12;
int CS = 10;
int CLK = 11;
LedControl lc=LedControl(DIN, CLK, CS,0);
int Cat[8]
={B10001000,B11111000,B10101000,B01110001,B00100001,B01111001,B01111101,B10111
110 };
void setup() {
  lc.shutdown(0,false);
```

```
lc.setIntensity(0,0);
lc.clearDisplay(0);  }
void loop(){
  for(int i=0;i<8;i++) lc.setRow(0,i,Cat[i]); }
```

Κύκλωμα 4:

Τίτλος κυκλώματος: Εμφάνιση ARDUINVET ρέοντος κειμένου

Περιγραφή κυκλώματος: Η διεπαφή Dot Matrix εμφανίζει ARDUINVET ρέοντος κειμένου.



1-) Breadboard view

```
/*  Display flowing text ARDUINVET*/
//D10=CS
//D11=CLK
//D12=DIN
#include <MaxMatrix.h> //include matrix library
#include <avr/pgmspace.h>
#include <stdlib.h>
PROGMEM const unsigned char CH[] = {
3, 8, B00000000, B00000000, B00000000, B00000000, B00000000, // space
1, 8, B01011111, B00000000, B00000000, B00000000, B00000000, // !
```

```
3, 8, B00000011, B00000000, B00000011, B00000000, B00000000, // "
5, 8, B00010100, B00111110, B00010100, B00111110, B00010100, // #
4, 8, B00100100, B01101010, B00101011, B00010010, B00000000, // $
5, 8, B01100011, B00010011, B00001000, B01100100, B01100011, // %
5, 8, B00110110, B01001001, B01010110, B00100000, B01010000, // &
1, 8, B00000011, B00000000, B00000000, B00000000, B00000000, // '
3, 8, B00011100, B00100010, B01000001, B00000000, B00000000, // (
3, 8, B01000001, B00100010, B00011100, B00000000, B00000000, // )
5, 8, B00101000, B00011000, B00001110, B00011000, B00101000, // *
5, 8, B00001000, B00001000, B00111110, B00001000, B00001000, // +
2, 8, B10110000, B01110000, B00000000, B00000000, B00000000, // ,
4, 8, B00001000, B00001000, B00001000, B00001000, B00000000, // -
2, 8, B01100000, B01100000, B00000000, B00000000, B00000000, // .
4, 8, B01100000, B00011000, B00000110, B00000001, B00000000, // /
4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // 0
3, 8, B01000010, B01111111, B01000000, B00000000, B00000000, // 1
4, 8, B01100010, B01010001, B01001001, B01000110, B00000000, // 2
4, 8, B00100010, B01000001, B01001001, B00110110, B00000000, // 3
4, 8, B00011000, B00010100, B00010010, B01111111, B00000000, // 4
4, 8, B00100111, B01000101, B01000101, B00111001, B00000000, // 5
4, 8, B00111110, B01001001, B01001001, B00110000, B00000000, // 6
4, 8, B01100001, B00010001, B00001001, B00000111, B00000000, // 7
4, 8, B00110110, B01001001, B01001001, B00110110, B00000000, // 8
4, 8, B00000110, B01001001, B01001001, B00111110, B00000000, // 9
2, 8, B01010000, B00000000, B00000000, B00000000, B00000000, // :
2, 8, B10000000, B01010000, B00000000, B00000000, B00000000, // ;
3, 8, B00010000, B00101000, B01000100, B00000000, B00000000, // <
3, 8, B00010100, B00010100, B00010100, B00000000, B00000000, // =
3, 8, B01000100, B00101000, B00010000, B00000000, B00000000, // >
4, 8, B00000010, B01011001, B00001001, B00000110, B00000000, // ?
5, 8, B00111110, B01001001, B01010101, B01011101, B00001110, // @
4, 8, B01111110, B00010001, B00010001, B01111110, B00000000, // A
4, 8, B01111111, B01001001, B01001001, B00110110, B00000000, // B
```

4, 8, B00111110, B01000001, B01000001, B00100010, B00000000, // C

4, 8, B01111111, B01000001, B01000001, B00111110, B00000000, // D

4, 8, B01111111, B01001001, B01001001, B01000001, B00000000, // E

4, 8, B01111111, B00001001, B00001001, B00000001, B00000000, // F

4, 8, B00111110, B01000001, B01001001, B01111010, B00000000, // G

4, 8, B01111111, B00001000, B00001000, B01111111, B00000000, // H

3, 8, B01000001, B01111111, B01000001, B00000000, B00000000, // I

4, 8, B00110000, B01000000, B01000001, B00111111, B00000000, // J

4, 8, B01111111, B00001000, B00010100, B01100011, B00000000, // K

4, 8, B01111111, B01000000, B01000000, B01000000, B00000000, // L

5, 8, B01111111, B00000010, B00001100, B00000010, B01111111, // M

5, 8, B01111111, B00000100, B00001000, B00010000, B01111111, // N

4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // O

4, 8, B01111111, B00001001, B00001001, B00000110, B00000000, // P

4, 8, B00111110, B01000001, B01000001, B10111110, B00000000, // Q

4, 8, B01111111, B00001001, B00001001, B01110110, B00000000, // R

4, 8, B01000110, B01001001, B01001001, B00110010, B00000000, // S

5, 8, B00000001, B00000001, B01111111, B00000001, B00000001, // T

4, 8, B00111111, B01000000, B01000000, B00111111, B00000000, // U

5, 8, B00001111, B00110000, B01000000, B00110000, B00001111, // V

5, 8, B00111111, B01000000, B00111000, B01000000, B00111111, // W

5, 8, B01100011, B00010100, B00001000, B00010100, B01100011, // X

5, 8, B00000111, B00001000, B01110000, B00001000, B00000111, // Y

4, 8, B01100001, B01010001, B01001001, B01000111, B00000000, // Z

2, 8, B01111111, B01000001, B00000000, B00000000, B00000000, // [

4, 8, B00000001, B00000110, B00011000, B01100000, B00000000, // \ backsplash

2, 8, B01000001, B01111111, B00000000, B00000000, B00000000, //]

3, 8, B00000010, B00000001, B00000010, B00000000, B00000000, // hat

4, 8, B01000000, B01000000, B01000000, B01000000, B00000000, // _

2, 8, B00000001, B00000010, B00000000, B00000000, B00000000, // `

4, 8, B00100000, B01010100, B01010100, B01111000, B00000000, // a

4, 8, B01111111, B01000100, B01000100, B00111000, B00000000, // b

4, 8, B00111000, B01000100, B01000100, B00101000, B00000000, // c

```
4, 8, B00111000, B01000100, B01000100, B01111111, B00000000, // d
4, 8, B00111000, B01010100, B01010100, B00011000, B00000000, // e
3, 8, B00000100, B01111110, B00000101, B00000000, B00000000, // f
4, 8, B10011000, B10100100, B10100100, B01111000, B00000000, // g
4, 8, B01111111, B00000100, B00000100, B01111000, B00000000, // h
3, 8, B01000100, B01111101, B01000000, B00000000, B00000000, // i
4, 8, B01000000, B10000000, B10000100, B01111101, B00000000, // j
4, 8, B01111111, B00010000, B00101000, B01000100, B00000000, // k
3, 8, B01000001, B01111111, B01000000, B00000000, B00000000, // l
5, 8, B01111100, B00000100, B01111100, B00000100, B01111000, // m
4, 8, B01111100, B00000100, B00000100, B01111000, B00000000, // n
4, 8, B00111000, B01000100, B01000100, B00111000, B00000000, // o
4, 8, B11111100, B00100100, B00100100, B00011000, B00000000, // p
4, 8, B00011000, B00100100, B00100100, B11111100, B00000000, // q
4, 8, B01111100, B00001000, B00000100, B00000100, B00000000, // r
4, 8, B01001000, B01010100, B01010100, B00100100, B00000000, // s
3, 8, B00000100, B00111111, B01000100, B00000000, B00000000, // t
4, 8, B00111100, B01000000, B01000000, B01111100, B00000000, // u
5, 8, B00011100, B00100000, B01000000, B00100000, B00011100, // v
5, 8, B00111100, B01000000, B00111100, B01000000, B00111100, // w
5, 8, B01000100, B00101000, B00010000, B00101000, B01000100, // x
4, 8, B10011100, B10100000, B10100000, B01111100, B00000000, // y
3, 8, B01100100, B01010100, B01001100, B00000000, B00000000, // z
3, 8, B00001000, B00110110, B01000001, B00000000, B00000000, // {
1, 8, B01111111, B00000000, B00000000, B00000000, B00000000, // |
3, 8, B01000001, B00110110, B00001000, B00000000, B00000000, // }
4, 8, B00001000, B00000100, B00001000, B00000100, B00000000, // ~
};
int data = 12;    // DIN pin of MAX7219 module
int load = 10;    // CS pin of MAX7219 module
int clock = 11;  // CLK pin of MAX7219 module
int maxInUse = 1;    //change this variable to set how many MAX7219's you'll use
MaxMatrix m(data, load, clock, maxInUse); // define module
```

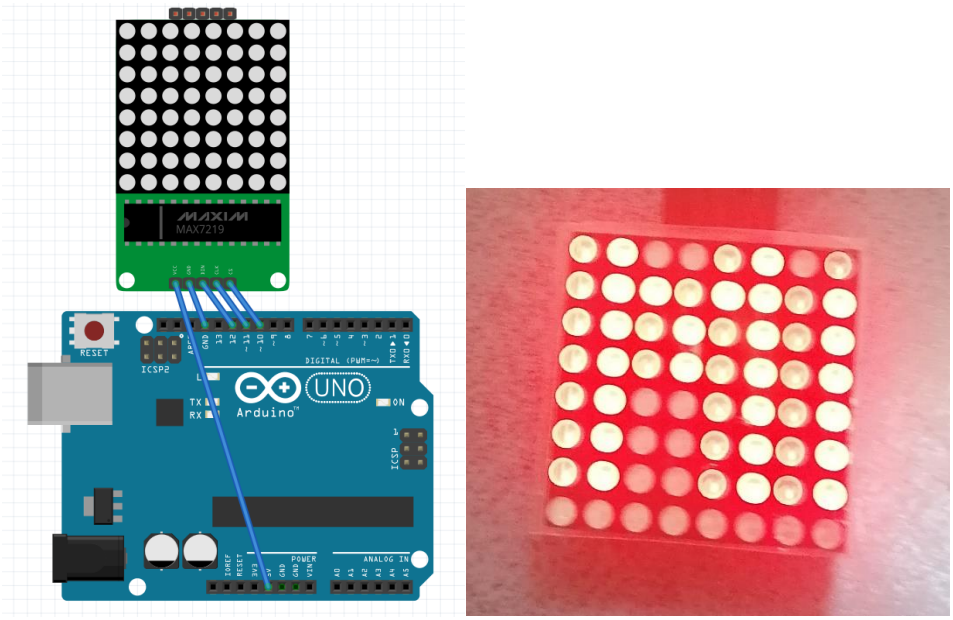
```
byte buffer[10];
void setup(){
  m.init(); // module initialize
  m.setIntensity(2); // dot matix intensity 0-15
  Serial.begin(9600); // serial communication initialize
  Serial.println("ARDUinVET");  }
void loop(){
     printStringWithShift("ARDUinVET    ", 100);
   m.shiftLeft(false, true);  }
void printCharWithShift(char c, int shift_speed){
  if (c < 32) return;
  c -= 32;
  memcpy_P(buffer, CH + 7*c, 7);
  m.writeSprite(32, 0, buffer);
  m.setColumn(32 + buffer[0], 0);
  for (int i=0; i<buffer[0]+1; i++)
  {    delay(shift_speed);
    m.shiftLeft(false, false);  }
}
void printStringWithShift(char* s, int shift_speed){
  while (*s != 0){
    printCharWithShift(*s, shift_speed);
    s++;  }
}
void printString(char* s)
{  int col = 0;
  while (*s != 0)
  {    if (*s < 32) continue;
    char c = *s - 32;
    memcpy_P(buffer, CH + 7*c, 7);
    m.writeSprite(col, 0, buffer);
    m.setColumn(col + buffer[0], 0);
    col += buffer[0] + 1;
```

s++; } }

Κύκλωμα 5:

Τίτλος κυκλώματος: Εμφάνιση κάθε γράμματος του αλφαβήτου

Περιγραφή κυκλώματος: Η διεπαφή Dot Matrix εμφανίζει κάθε γράμμα του αλφαβήτου μετά από 1 δευτερόλεπτο.



1-) Breadboard view

/* Display every letter of alphabet*/

#include <MaxMatrix.h>//download from https://code.google.com/archive/p/arudino-maxmatrix-library/downloads

int DIN = 12; // DIN pin of MAX7219 module

int CLK = 11; // CLK pin of MAX7219 module

int CS = 10; // CS pin of MAX7219 module

int maxInUse = 1;

MaxMatrix m(DIN, CS, CLK, maxInUse);

char A[] = {8, 8,

```
B00111000,B01000100,B01000100,B01000100,B01111100,B01000100,B01000100,B0100010
0};
char B[] = {8, 8,
B01111000,B01000100,B01000100,B01111000,B01000100,B01000100,B01111000,B0000000
0};
char c[] = {8, 8,
B00000000,B00111100,B01000000,B01000000,B01000000,B01000000,B00111100,B0000000
0};
char d[] = {8, 8,
B00000000,B01111000,B01000100,B01000100,B01000100,B01000100,B01111000,B0000000
0};
char e[] = {8, 8,
B00000000,B01111100,B01000000,B01000000,B01111100,B01000000,B01000000,B0111110
0};
char f[] = {8, 8,
B00000000,B01111100,B01000000,B01000000,B01111100,B01000000,B01000000,B0100000
0};
char g[] = {8, 8,
B00000000,B00111100,B01000000,B01000000,B01001110,B01000010,B0011111
0};
char h[] = {8, 8,
B00000000,B01000100,B01000100,B01111100,B01000100,B01000100,B01000100,B0000000
0};
char i[] = {8, 8,
B00000000,B01111100,B00010000,B00010000,B00010000,B00010000,B01111100,B0000000
0};
char j[] = {8, 8,
B00000000,B00000100,B00000100,B00000100,B00000100,B01000100,B00111000,B0000000
0};
char k[] = {8, 8,
B00000000,B00100100,B00101000,B00110000,B00101000,B00100100,B00100010,B0000000
0};
char l[] = {8, 8,
```

```
B00000000,B01000000,B01000000,B01000000,B01000000,B01000000,B01111100,B00000000};
char n[] = {8, 8,
B00000000,B01000010,B01100010,B01010010,B01001010,B01000110,B01000010,B00000000};
char o[] = {8, 8,
B00111100,B01000010,B01000010,B01000010,B01000010,B01000010,B01000010,B00111100};
char p[] = {8, 8,
B00000000,B00111000,B00100100,B00100100,B00111000,B00100000,B00100000,B00000000};
char q[] = {8, 8,
B00111100,B01000010,B01000010,B01000010,B01001010,B01000110,B00111110,B00000001};
char r[] = {8, 8,
B01111000,B01000100,B01000100,B01111000,B01100000,B01010000,B01001000,B01000100};
char s[] = {8, 8,
B00111100,B01000000,B01000000,B00111000,B00000100,B00000100,B01111000,B00000000};
char t[] = {8, 8,
B00000000,B01111100,B00010000,B00010000,B00010000,B00010000,B00010000,B00000000};
char u[] = {8, 8,
B00000000,B01000010,B01000010,B01000010,B01000010,B01000010,B00111100,B00000000};
char v[] = {8, 8,
B00000000,B00100100,B00100100,B00100100,B00100100,B00100100,B00011000,B00000000};
char w[] = {8, 8,
B00000000,B01010100,B01010100,B01010100,B01010100,B01010100,B00111000,B00000000};
char x[] = {8, 8,
```

```
B00000000,B01000100,B00101000,B00010000,B00101000,B01000100,B00000000,B0000000
0};
char y[] = {8, 8,
B10000001,B01000010,B00100100,B00011000,B00011000,B00011000,B00011000,B0001100
0};
char z[] = {8, 8,
B00000000,B01111100,B00001000,B00010000,B00100000,B01111100,B00000000,B0000000
0};void setup() {
  m.init(); // MAX7219 initialization
  m.setIntensity(5); // initial led matrix intensity, 0-15     }
void loop() {
  // Seting the LEDs On or Off at x,y or row,column position
  m.setDot(6,2,true);
  delay(1000);
  m.setDot(6,3,true);
  delay(1000);
  m.clear(); // Clears the display
  for (int i=0; i<8; i++){
    m.setDot(i,i,true);
    delay(300);}
  m.clear();
  // Displaying the character at x,y (upper left corner of the character)
  m.writeSprite(0, 0, A);
  delay(1000);
  m.writeSprite(0, 0, B);
  delay(1000);
  m.writeSprite(0, 0, c);
  delay(1000);
  m.writeSprite(0, 0, d);
  delay(1000);
  m.writeSprite(0, 0, e);
  delay(1000);
  m.writeSprite(0, 0, f);
```

```
 delay(1000);
m.writeSprite(0, 0, g);
 delay(1000);
m.writeSprite(0, 0, h);
 delay(1000);
m.writeSprite(0, 0, i);
 delay(1000);
m.writeSprite(0, 0, j);
 delay(1000);
m.writeSprite(0, 0, k);
 delay(1000);
 m.writeSprite(0, 0, l);
 delay(1000);
m.writeSprite(0, 0, n);
 delay(1000);
m.writeSprite(0, 0, o);
 delay(1000);
  m.writeSprite(0, 0, p);
 delay(1000);
  m.writeSprite(0, 0, q);
 delay(1000);
  m.writeSprite(0, 0, r);
 delay(1000);
  m.writeSprite(0, 0, s);
 delay(1000);
  m.writeSprite(0, 0, t);
 delay(1000);
  m.writeSprite(0, 0, u);
 delay(1000);
  m.writeSprite(0, 0, v);
 delay(1000);
  m.writeSprite(0, 0, w);
 delay(1000);
  m.writeSprite(0, 0, x);
 delay(1000);
m.writeSprite(0, 0, y);
 delay(1000);
```

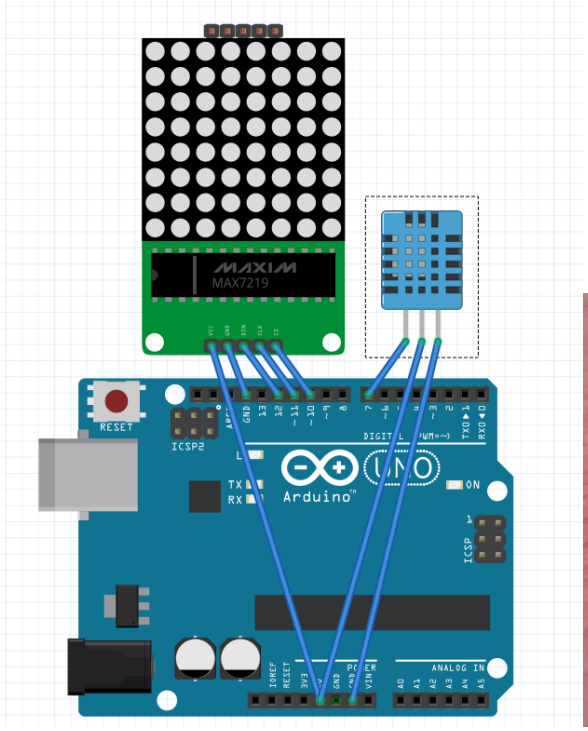
m.writeSprite(0, 0, z);

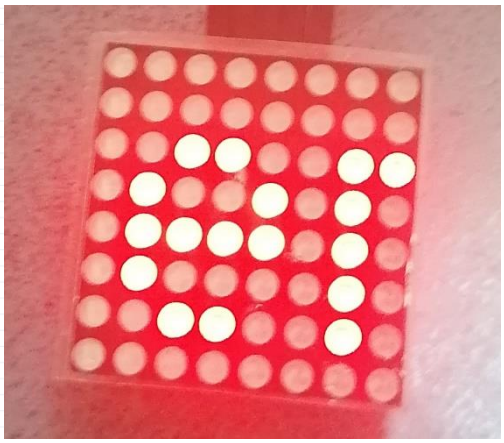
delay(1000);}

Κύκλωμα 6:

Τίτλος κυκλώματος: Interface Dot Matrix με Arduino

Περιγραφή κυκλώματος: Η διασύνδεση Dot Matrix εμφανίζει τη θερμοκρασία που μετρήθηκε χρησιμοποιώντας τον αισθητήρα θερμοκρασίας DHT11.





1-) Breadboard view

/* Interface Dot Matrix with Arduino */

//D7= temp

//D10=CS

//D11=CLK

//D12=DIN

#include <MaxMatrix.h> //include matrix library

#include <avr/pgmspace.h>

```
#include <stdlib.h>
#include "DHT.h" //include the temp sensor library
#define DHTPIN 7     // what pin we're connected to
#define DHTTYPE DHT11   // DHT 11 temp&humid sensor
DHT dht(DHTPIN, DHTTYPE);
PROGMEM const unsigned char CH[] = {
3, 8, B00000000, B00000000, B00000000, B00000000, B00000000, // space
1, 8, B01011111, B00000000, B00000000, B00000000, B00000000, // !
3, 8, B00000011, B00000000, B00000011, B00000000, B00000000, // "
5, 8, B00010100, B00111110, B00010100, B00111110, B00010100, // #
4, 8, B00100100, B01101010, B00101011, B00010010, B00000000, // $
5, 8, B01100011, B00010011, B00001000, B01100100, B01100011, // %
5, 8, B00110110, B01001001, B01010110, B00100000, B01010000, // &
1, 8, B00000011, B00000000, B00000000, B00000000, B00000000, // '
3, 8, B00011100, B00100010, B01000001, B00000000, B00000000, // (
3, 8, B01000001, B00100010, B00011100, B00000000, B00000000, // )
5, 8, B00101000, B00011000, B00001110, B00011000, B00101000, // *
5, 8, B00001000, B00001000, B00111110, B00001000, B00001000, // +
2, 8, B10110000, B01110000, B00000000, B00000000, B00000000, // ,
4, 8, B00001000, B00001000, B00001000, B00001000, B00000000, // -
2, 8, B01100000, B01100000, B00000000, B00000000, B00000000, // .
4, 8, B01100000, B00011000, B00000110, B00000001, B00000000, // /
4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // 0
3, 8, B01000010, B01111111, B01000000, B00000000, B00000000, // 1
4, 8, B01100010, B01010001, B01001001, B01000110, B00000000, // 2
4, 8, B00100010, B01000001, B01001001, B00110110, B00000000, // 3
4, 8, B00011000, B00010100, B00010010, B01111111, B00000000, // 4
4, 8, B00100111, B01000101, B01000101, B00111001, B00000000, // 5
4, 8, B00111110, B01001001, B01001001, B00110000, B00000000, // 6
4, 8, B01100001, B00010001, B00001001, B00000111, B00000000, // 7
4, 8, B00110110, B01001001, B01001001, B00110110, B00000000, // 8
4, 8, B00000110, B01001001, B01001001, B00111110, B00000000, // 9
2, 8, B01010000, B00000000, B00000000, B00000000, B00000000, // :
```

2, 8, B10000000, B01010000, B00000000, B00000000, B00000000, // ;

3, 8, B00010000, B00101000, B01000100, B00000000, B00000000, // <

3, 8, B00010100, B00010100, B00010100, B00000000, B00000000, // =

3, 8, B01000100, B00101000, B00010000, B00000000, B00000000, // >

4, 8, B00000010, B01011001, B00001001, B00000110, B00000000, // ?

5, 8, B00111110, B01001001, B01010101, B01011101, B00001110, // @

4, 8, B01111110, B00010001, B00010001, B01111110, B00000000, // A

4, 8, B01111111, B01001001, B01001001, B00110110, B00000000, // B

4, 8, B00111110, B01000001, B01000001, B00100010, B00000000, // C

4, 8, B01111111, B01000001, B01000001, B00111110, B00000000, // D

4, 8, B01111111, B01001001, B01001001, B01000001, B00000000, // E

4, 8, B01111111, B00001001, B00001001, B00000001, B00000000, // F

4, 8, B00111110, B01000001, B01001001, B01111010, B00000000, // G

4, 8, B01111111, B00001000, B00001000, B01111111, B00000000, // H

3, 8, B01000001, B01111111, B01000001, B00000000, B00000000, // I

4, 8, B00110000, B01000000, B01000001, B00111111, B00000000, // J

4, 8, B01111111, B00001000, B00010100, B01100011, B00000000, // K

4, 8, B01111111, B01000000, B01000000, B01000000, B00000000, // L

5, 8, B01111111, B00000010, B00001100, B00000010, B01111111, // M

5, 8, B01111111, B00000100, B00001000, B00010000, B01111111, // N

4, 8, B00111110, B01000001, B01000001, B00111110, B00000000, // O

4, 8, B01111111, B00001001, B00001001, B00000110, B00000000, // P

4, 8, B00111110, B01000001, B01000001, B10111110, B00000000, // Q

4, 8, B01111111, B00001001, B00001001, B01110110, B00000000, // R

4, 8, B01000110, B01001001, B01001001, B00110010, B00000000, // S

5, 8, B00000001, B00000001, B01111111, B00000001, B00000001, // T

4, 8, B00111111, B01000000, B01000000, B00111111, B00000000, // U

5, 8, B00001111, B00110000, B01000000, B00110000, B00001111, // V

5, 8, B00111111, B01000000, B00111000, B01000000, B00111111, // W

5, 8, B01100011, B00010100, B00001000, B00010100, B01100011, // X

5, 8, B00000111, B00001000, B01110000, B00001000, B00000111, // Y

4, 8, B01100001, B01010001, B01001001, B01000111, B00000000, // Z

2, 8, B01111111, B01000001, B00000000, B00000000, B00000000, // [

4, 8, B00000001, B00000110, B00011000, B01100000, B00000000, // \ backsplash

2, 8, B01000001, B01111111, B00000000, B00000000, B00000000, //]

3, 8, B00000010, B00000001, B00000010, B00000000, B00000000, // hat

4, 8, B01000000, B01000000, B01000000, B01000000, B00000000, // _

2, 8, B00000001, B00000010, B00000000, B00000000, B00000000, // `

4, 8, B00100000, B01010100, B01010100, B01111000, B00000000, // a

4, 8, B01111111, B01000100, B01000100, B00111000, B00000000, // b

4, 8, B00111000, B01000100, B01000100, B00101000, B00000000, // c

4, 8, B00111000, B01000100, B01000100, B01111111, B00000000, // d

4, 8, B00111000, B01010100, B01010100, B00011000, B00000000, // e

3, 8, B00000100, B01111110, B00000101, B00000000, B00000000, // f

4, 8, B10011000, B10100100, B10100100, B01111000, B00000000, // g

4, 8, B01111111, B00000100, B00000100, B01111000, B00000000, // h

3, 8, B01000100, B01111101, B01000000, B00000000, B00000000, // i

4, 8, B01000000, B10000000, B10000100, B01111101, B00000000, // j

4, 8, B01111111, B00010000, B00101000, B01000100, B00000000, // k

3, 8, B01000001, B01111111, B01000000, B00000000, B00000000, // l

5, 8, B01111100, B00000100, B01111100, B00000100, B01111000, // m

4, 8, B01111100, B00000100, B00000100, B01111000, B00000000, // n

4, 8, B00111000, B01000100, B01000100, B00111000, B00000000, // o

4, 8, B11111100, B00100100, B00100100, B00011000, B00000000, // p

4, 8, B00011000, B00100100, B00100100, B11111100, B00000000, // q

4, 8, B01111100, B00001000, B00000100, B00000100, B00000000, // r

4, 8, B01001000, B01010100, B01010100, B00100100, B00000000, // s

3, 8, B00000100, B00111111, B01000100, B00000000, B00000000, // t

4, 8, B00111100, B01000000, B01000000, B01111100, B00000000, // u

5, 8, B00011100, B00100000, B01000000, B00100000, B00011100, // v

5, 8, B00111100, B01000000, B00111100, B01000000, B00111100, // w

5, 8, B01000100, B00101000, B00010000, B00101000, B01000100, // x

4, 8, B10011100, B10100000, B10100000, B01111100, B00000000, // y

3, 8, B01100100, B01010100, B01001100, B00000000, B00000000, // z

3, 8, B00001000, B00110110, B01000001, B00000000, B00000000, // {

1, 8, B01111111, B00000000, B00000000, B00000000, B00000000, // |

```
3, 8, B01000001, B00110110, B00001000, B00000000, B00000000, // }
4, 8, B00001000, B00000100, B00001000, B00000100, B00000000, // ~
};
int data = 12;    // DIN pin of MAX7219 module
int load = 10;    // CS pin of MAX7219 module
int clock = 11;  // CLK pin of MAX7219 module
int maxInUse = 1;    //change this variable to set how many MAX7219's you'll use
MaxMatrix m(data, load, clock, maxInUse); // define module
byte buffer[10];
void setup(){
  m.init(); // module initialize
  m.setIntensity(2); // dot matix intensity 0-15
  Serial.begin(9600); // serial communication initialize
  Serial.println("DHTxx test!");
  dht.begin();
}
void loop(){
  int t = dht.readTemperature();
  char temp[4];
  itoa(t,temp,10); //convert int to char!!!!
  Serial.println(temp);
   printStringWithShift("BRAILA ROMANIA", 100);
  printStringWithShift("  temp: ", 100);
  printStringWithShift(temp, 100);
  printStringWithShift(" C   ", 100);
  m.shiftLeft(false, true);
  }
void printCharWithShift(char c, int shift_speed){
  if (c < 32) return;
  c -= 32;
  memcpy_P(buffer, CH + 7*c, 7);
  m.writeSprite(32, 0, buffer);
  m.setColumn(32 + buffer[0], 0);
```

```
 for (int i=0; i<buffer[0]+1; i++)
 {
  delay(shift_speed);
  m.shiftLeft(false, false);
 }
}
void printStringWithShift(char* s, int shift_speed){
 while (*s != 0){
  printCharWithShift(*s, shift_speed);
  s++;
 }
}
void printString(char* s)
{
 int col = 0;
 while (*s != 0)
 {
  if (*s < 32) continue;
  char c = *s - 32;
  memcpy_P(buffer, CH + 7*c, 7);
  m.writeSprite(col, 0, buffer);
  m.setColumn(col + buffer[0], 0);
  col += buffer[0] + 1;
  s++;
 }
}
```

Co-funded by the Erasmus+ Programme of the European Union

Erasmus+ KA-202

Πρόγραμμα Στρατηγικής Σύμπραξης στην Επαγγελματική Εκπαίδευση και Κατάρτιση

Τίτλος Προγράμματος: "Teaching and Learning Arduinos in Vocational Training"

Ακρονύμιο Προγράμματος: " ARDUinVET "

Κωδικός Προγράμματος: "2020-1-TR01-KA202-093762"

Erasmus+ KA-202

Κινητήρες Arduino και εκπαιδευτικό κιτ

(DC, Step, Servo)

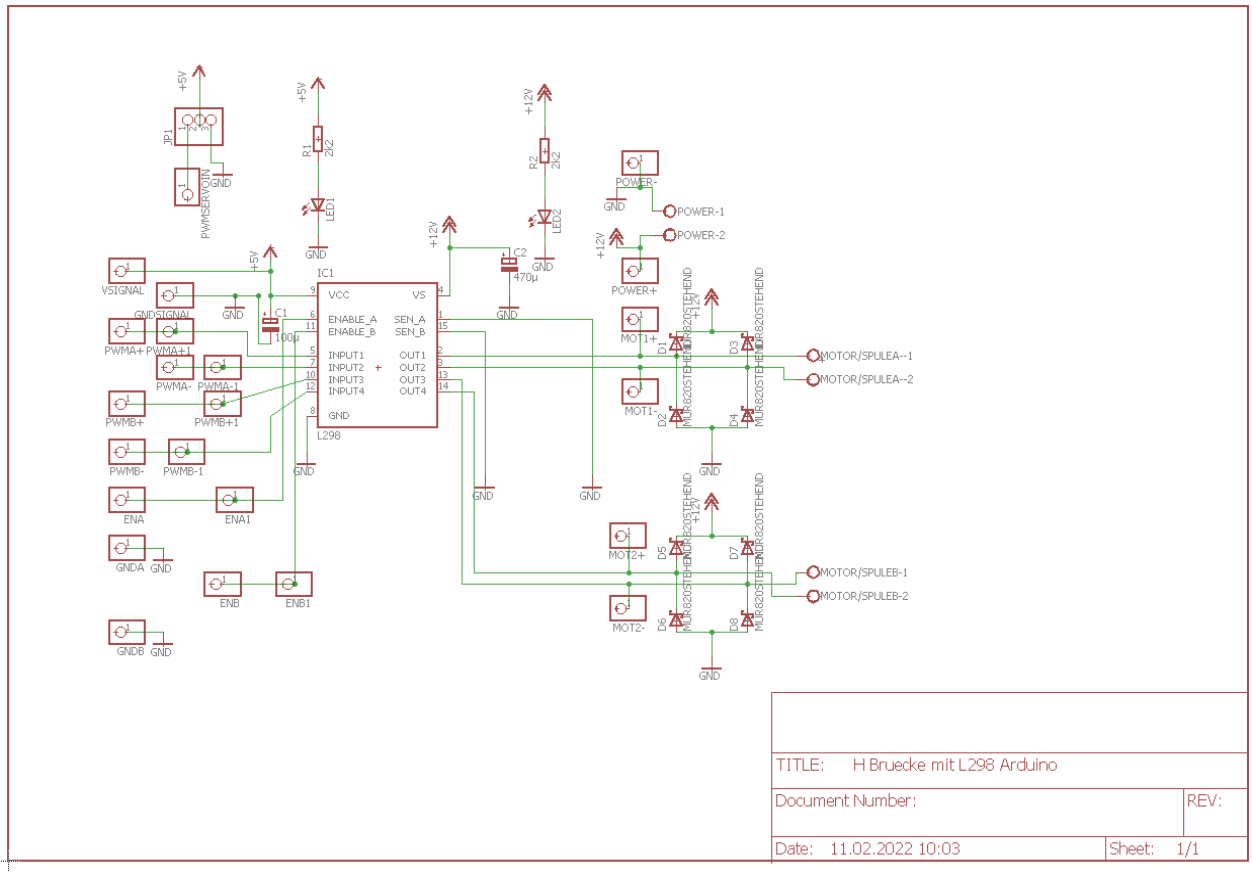
Ενότητα και εκπαιδευτικό κιτ Κινητήρων Arduino

Το εκπαιδευτικό κιτ μονάδων κινητήρα είναι ένα διδακτικό εργαλείο που αναπτύχθηκε στο πλαίσιο του έργου ARDUinVET, με στόχο να εξηγήσει με απλό τρόπο τη λειτουργία διαφορετικών ειδών των κινητήρων (Servo, Step και DC) που ελέγχονται από το Arduino..

Χρησιμοποιώντας την πλατφόρμα προγραμματισμού του Arduino, σκοπεύουμε να προσεγγίσουμε με απλό τρόπο τον προγραμματισμό των πλακετών Arduino ώστε να λειτουργούν οι διαφορετικοί κινητήρες, με τη χρήση του υλικού που έχουμε χρησιμοποιήσει.

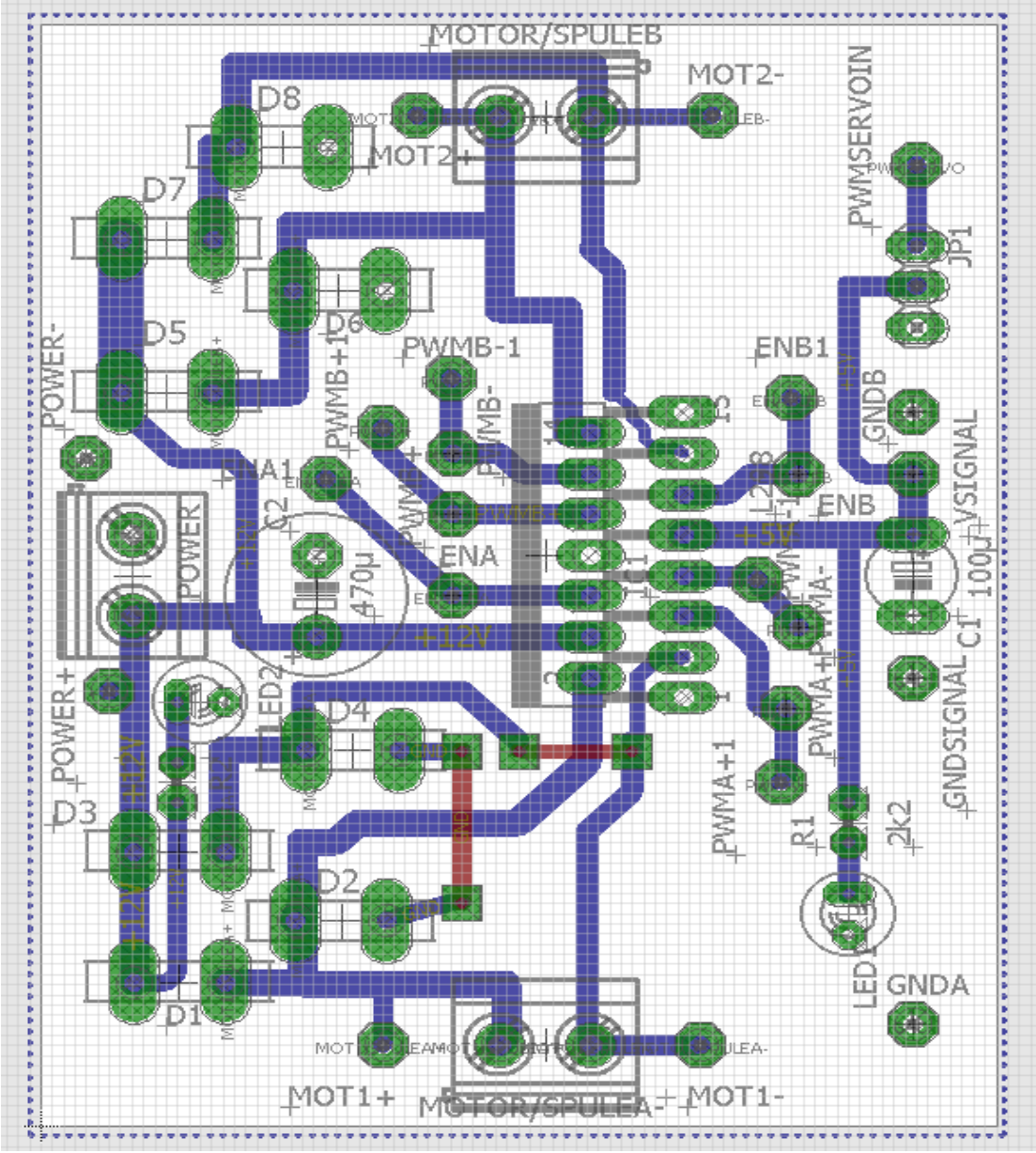
Σκοπός μας είναι να δείξουμε στους μαθητές πώς να κατασκευάζουν και να προγραμματίζουν μια πλακέτα τυπωμένου κυκλώματος (εικόνα 1), καθώς και τη δυνατότητα διαφόρων χρήσεων της, κάνοντας τους κινητήρες να λειτουργούν και να εκτελούν διαφορετικές εργασίες, στη συγκεκριμένη περίπτωση.

Διάγραμμα ηλεκτρονικού κυκλώματος Κινητήρα



Εικόνα: 1

Διάταξη κυκλωματος κινητήρα



Εικόνα: 2

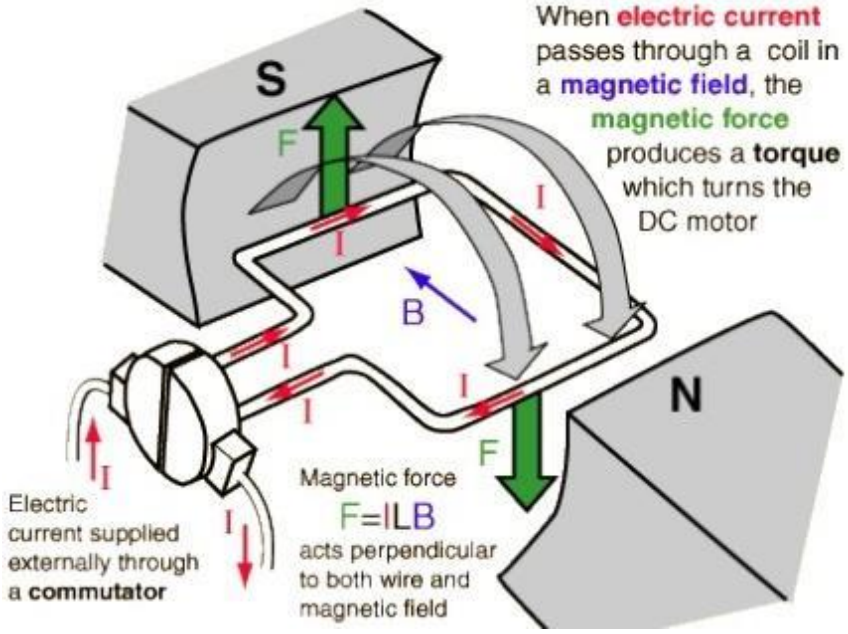
ΚΙΝΗΤΗΡΕΣ

Κινητήρες DC

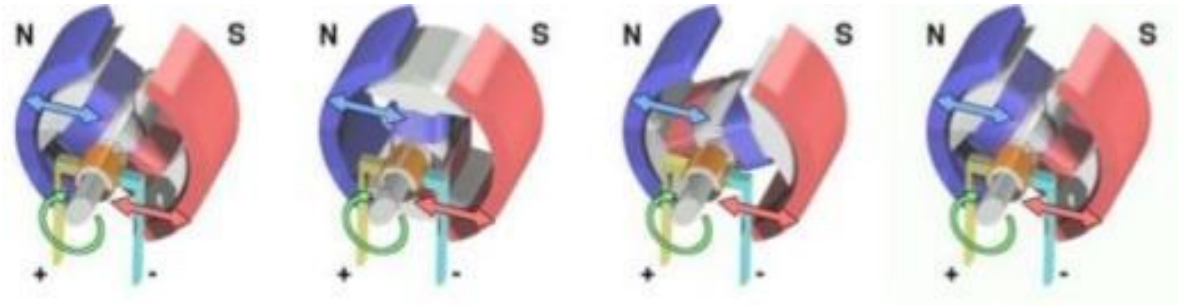
Οι κινητήρες DC, (Συνεχούς ρεύματος, Direct Current, εικόνα 3) είναι ηλεκτρονικές συσκευές που λειτουργούν μέσω των δυνάμεων έλξης και απώθησης που δημιουργούνται από τους μαγνήτες.

Κινητήρας DC

Παράδειγμα:



Εικόνα 3: Λειτουργία κινητήρων DC



1 **2** **3** **4**

1. Όταν το πηνίο τροφοδοτείται, δημιουργείται ένα μαγνητικό πεδίο γύρω από τον ρότορα, προκαλώντας απώθηση μεταξύ των μαγνητών, κάνοντας τον κινητήρα να περιστρέφεται δεξιόστροφα.

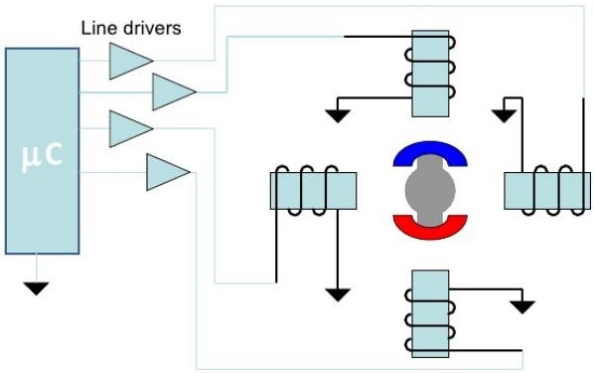
2. Ο ρότορας συνεχίζει να περιστρέφεται.

3. Όταν ο ρότορας ευθυγραμμιστεί οριζόντια, το μαγνητικό πεδίο αναστρέφεται, συνεχίζοντας την κίνηση δεξιόστροφα.

4. Η διαδικασία επαναλαμβάνεται συνεχώς ενόσω ο κινητήρας τροφοδοτείται.

Κινητήρες Step Motors

Ο step motor (βηματικός κινητήρας) είναι ένας ηλεκτροκινητήρας που κινείται σύμφωνα με έναν παλμό που λαμβάνεται μέσω ενός ελεγκτή. Ο αριθμός των βημάτων που κινείται ο κινητήρας είναι ακριβώς ο ίδιος με τον αριθμό των παλμών που λαμβάνει και η ταχύτητα του κινητήρα είναι ίδια με τη συχνότητα των παλμών. Επομένως είναι μια απλή συσκευή (χωρίς ψύκτρες, χωρίς διακόπτες και χωρίς κωδικοποιητές). Είναι κινητήρες που χρησιμοποιούνται σε διάφορους τομείς ηλεκτρονικών και αυτοματισμών, όπως η ρομποτική, η κίνηση των καρτεσιανών αξόνων, μεταξύ άλλων..



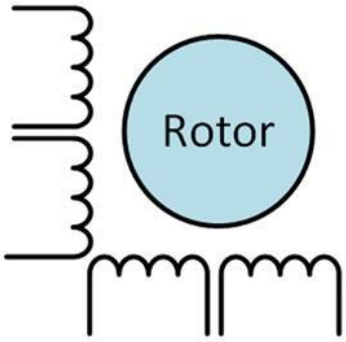
Εικόνα 4: Απλό διάγραμμα Step Motor

Εικόνα 5: Step Motor

Παραδείγματα step motors

Μονοπολικός κινητήρας

Ένας μονοπολικός κινητήρας έχει δύο πηνία ανά φάση, ένα για κάθε κατεύθυνση ηλεκτρικού ρεύματος. Σε αυτή τη σύνθεση ένας μαγνητικός πόλος μπορεί να αναστραφεί χωρίς αλλαγή της κατεύθυνσης του ηλεκτρικού ρεύματος, το κύκλωμα του διακόπτη μπορεί να γίνει πολύ απλά για κάθε πηνίο.



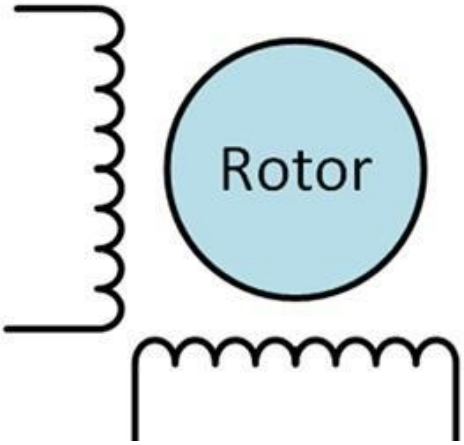
Εικόνα 6: Μονοπολικός κινητήρας

Cycle	C1	C2	C3	C4
A	1	0	0	0
B	0	1	0	0
C	0	0	1	0
D	0	0	0	1

Πίνακας 1: Πίνακας κατάστασης.

Διοπολικός κινητήρας

Οι διπολικοί κινητήρες έχουν ένα μοναδικό πηνίο ανά φάση. Το ηλεκτρικό ρεύμα σε ένα πηνίο πρέπει να αναστραφεί για να αναστραφεί ένας μαγνητικός πόλος. Έτσι το ηλεκτρικό κύκλωμα είναι λίγο πιο περίπλοκο, απαιτώντας μια γέφυρα Η. Υπάρχουν δύο συνδέσεις ανά φάση και καμία δεν είναι κοινή.

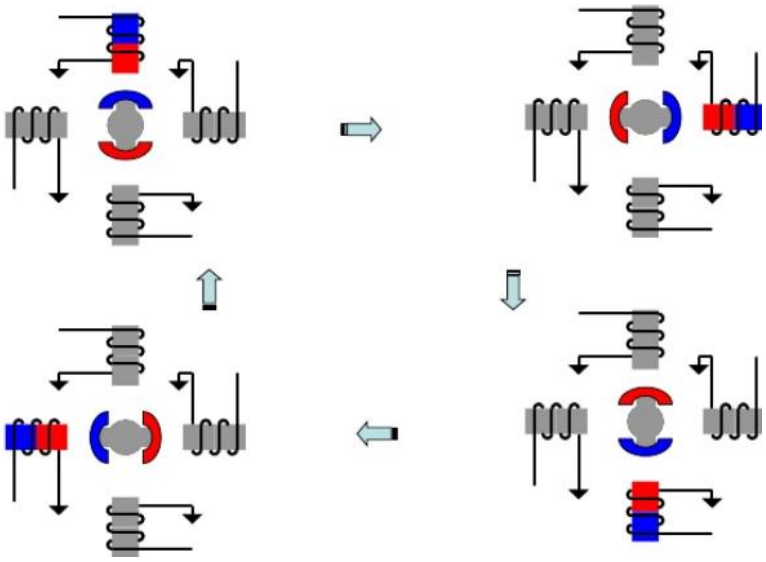


Εικόνα 7: *Διοπολικός κινητήρας*

Cycle	C1	C2	C3	C4
A	1	0	0	0
B	0	1	0	0
C	0	0	1	0
D	0	0	0	1

Πίνακας 2: Πίνακας κατάστασης.

Λειτουργία Step motor



Εικόνα 8: Σχήμα Step motor

Κινητήρες Servo

Ο σερβοκινητήρας είναι μια ηλεκτρομηχανική συσκευή που μέσω ηλεκτρικού σήματος τοποθετεί τον άξονα σε γωνιακή θέση. Σε κανονική λειτουργία, αυτός ο τύπος κινητήρα είναι συμπαγής, επιτρέποντας μια ακριβή θέση του άξονά του. Επί του παρόντος, οι σερβοκινητήρες χρησιμοποιούνται στη ρομποτική και τη μοντελοποίηση.

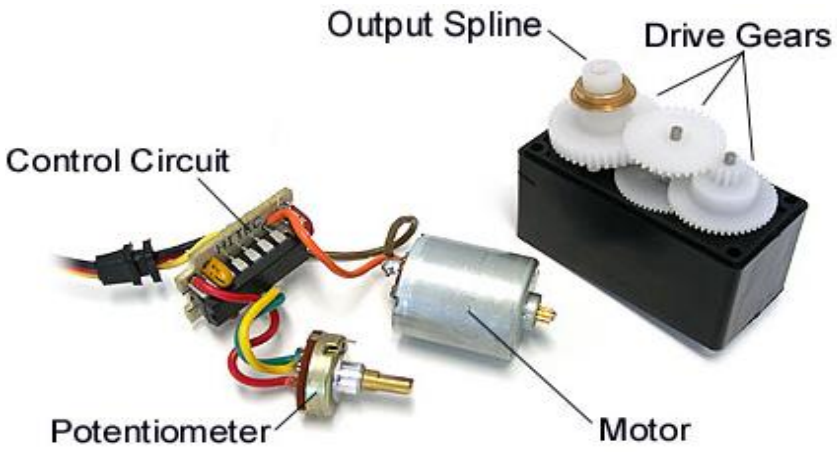
Εικόνα 1: Κινητήρας Servo

Λειτουργία:

Ένας κινητήρας servo μπορεί να χωριστεί σε 4 μέρη:

- **Κύκλωμα ελέγχου** – Υπεύθυνο για τη λήψη των σημάτων PWM και της ισχύος. Ελέγχει τη θέση του ποτενσιόμετρου και ελέγχει τον κινητήρα σύμφωνα με το σήμα PWM που λαμβάνεται.
- **Ποτενσιόμετρο**– Συνδέεται στον άξονα εξόδου του σερβομηχανισμού, βάζοντας τον στη θέση του.
- **Κινητήρας** – Μετακινεί το γρανάζι και τον κύριο άξονα του σερβομηχανισμού.

- **Γρανάζια κίνησης** – Μειώνουν την περιστροφή του κινητήρα, μειώνοντας την αντοχή έτσι ώστε να εφαρμόζει ανώτερη ροπή στον κύριο άξονα. Κινούν το ποτενσιόμετρο μαζί με τον άξονα.

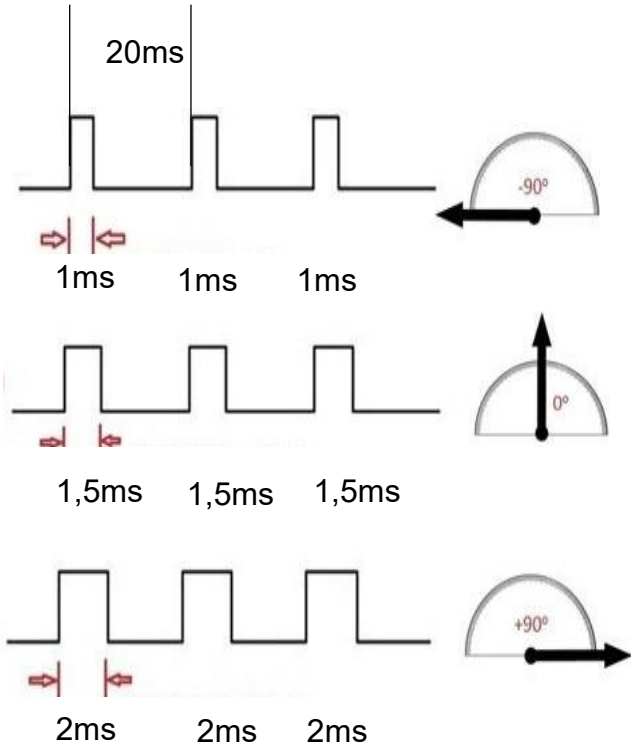


Εικόνα 10: Μέρη κινητήρα Servo

Θέσεις κινητήρα Servo

Ο έλεγχος ενός σερβοκινητήρα λαμβάνεται από ένα σήμα εισόδου που παρουσιάζει επίπεδα τάσης TTL τα οποία καθορίζουν τη θέση του. Αυτή η μορφή σήματος ακολουθεί τη διαμόρφωση PWM (Pulse Width Modulation) όπως φαίνεται στην εικόνα 11. Η κωδικοποίηση στο σήμα PWM γίνεται μέσω παλμού υψηλού επιπέδου σε σχέση με τη συνολική περίοδο ταλάντωσης, η οποία στην περίπτωση των σερβοκινητήρων είναι 20ms.

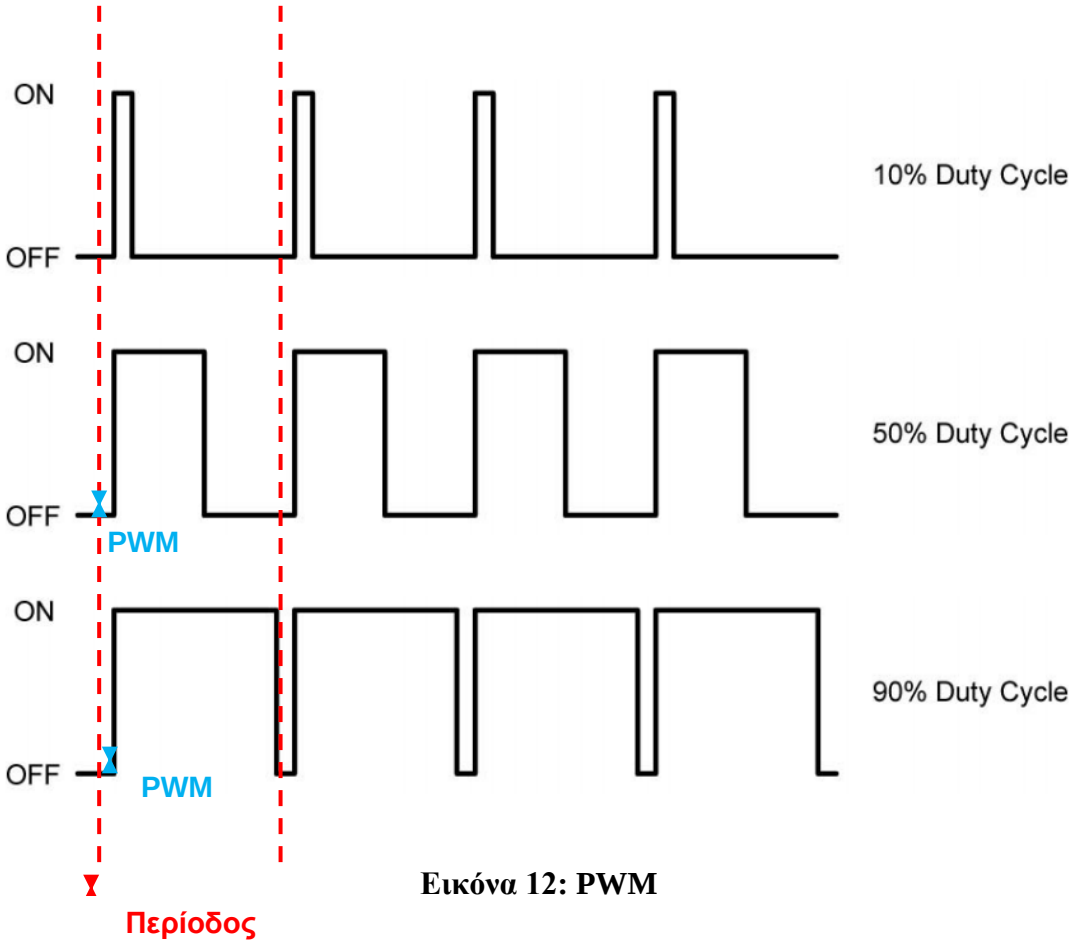
Στη συγκεκριμένη περίπτωση, εάν σε 20 χιλιοστά του δευτερολέπτου το 1 χιλιοστό του δευτερολέπτου είναι σε υψηλό επίπεδο, ο σερβοκινητήρας πρέπει να βρίσκεται στην χαμηλότερη θέση, όπως φαίνεται στην εικόνα 11.



Εικόνα 11: PWM ενός κινητήρα servo

PWM (Pulse Width Modulation)

Το PWM μπορεί να εφαρμοστεί σε διάφορους τομείς της ηλεκτρονικής. Χρήσεις του είναι στην παροχή ρεύματος, τον έλεγχο ταχύτητας κινητήρων DC, τον έλεγχο φωτός, τον έλεγχο σερβοκινητήρων και πολλές άλλες εφαρμογές. Μέσω του PWM μπορούμε να ελέγξουμε την ταχύτητα και την ισχύ.



Εικόνα 12: PWM

Λειτουργία PWM

Θεωρώντας ένα τετράγωνο κύμα, για να επιτύχουμε τη σωστή λειτουργία του PWM πρέπει να μεταβάλλουμε το πλάτος παλμού του κύματος. Για να υπολογίσουμε χρειαζόμαστε την περίοδο και το πλάτος του παλμού και το αποτέλεσμά του ονομάζεται κύκλος λειτουργίας, όπως ορίζεται από την εξίσωση:

$$Κύκλος\ Λειτουργίας = 100\frac{Πλάτος\ Παλμού}{Περίοδος}$$

Κύκλος Λειτουργίας: Ποσοστιαία τιμή

Πλάτος παλμού: χρονική διάρκεια κατά την οποία το σήμα είναι σε υψηλό επίπεδο.

Περίοδος: διάρκεια ενός κύκλου.

ΣΗΜΕΙΩΣΗ: Τα παρακάτω πειράματα θα γίνουν χρησιμοποιώντας το κιτ κινητήρα Arduino, το οποίο θα φτιάξουν μόνοι **τους οι μαθητές.**

Τίτλος κυκλώματος 1 (με κινητήρες DC): "Ενας απλός κινητήρας DC "

Περιγραφή κυκλώματος: Το πρόγραμμα δείχνει τις διαδικασίες εκκίνησης και σβησίματος ενός κινητήρα DC

Πρόγραμμα:

// One DC-Motor simple

```
/*
    Connection:
Arduino   | MotorShielt
PIN       | PIN
-------------------------
+5V       | VSIGNAL
GND       | GNDSIGNAL
4         | ENA
5         | PWMA+
6         | PWMA-
*/
//-------------------------------------------------------------
//  variables
//-------------------------------------------------------------
int ena = 4;    // Arduino pin 4 connected to ENA
int right = 5;  // Arduino pin 5 connected to PWMA+
int left = 6;   // Arduino pin 5 connected to PWMA-

//-------------------------------------------------------------
//  setup function
//-------------------------------------------------------------
void setup()
{
  pinMode(ena, OUTPUT);
  pinMode(right, OUTPUT);
  pinMode(left, OUTPUT);
  digitalWrite(ena, HIGH);  //Enable ENA
```

```
  delay(900);
}
//-----------------------------------------------------------
// loop Function
//-----------------------------------------------------------
void loop()
{
  analogWrite(right, 255);  // turn right with maximum speed
  delay(1000);
  analogWrite(right, 0);    // turn off
  delay(1000);
  analogWrite(left, 255);   // turn left with maximum speed
  delay(1000);
  analogWrite(left, 0);     // turn off
  delay(1000);
  analogWrite(right, 127);  // turn right with half speed
  delay(1000);
  analogWrite(right, 0);    // turn off
  delay(1000);
  analogWrite(left, 127);   // turn left with half speed
  delay(1000);
  analogWrite(left, 0);     // turn off
  delay(1000);
}
```

Τίτλος κυκλώματος 2: "Κίνηση ενός κινητήρα DC "

Περιγραφή κυκλώματος: Το πρόγραμμα ελέγχει ένα κινητήρα DC. Οι εντολές ελέγχου δίνονται από το σειριακό παράθυρο (serial monitor).

The control commands:

Write into the Serial Monitor	*Action*
stop	*The motor stops*
go right	*Motor turn right*
go left	*Motor turn left*
+	*Increase speed*
-	*Decrease speed*

Πρόγραμμα:

// One DC-Motor advance

```
/*
    Connection:
```

```
Arduino   | MotorShielt
PIN       | PIN
------------------------
+5V       | VSIGNAL
GND       | GNDSIGNAL
4         | ENA
5         | PWMA+
6         | PWMA-
*/
//-------------------------------------------------------------
//  variables
//-------------------------------------------------------------
int en_a = 4;              //ArduPin4 connected to enable Pin
int right_a = 5;           //ArduPin5 connected to PWMA+ Pin
int left_a = 6;            //ArduPin6 connected to PWMA- Pin
String message = "";       //save the message of the seriel interface
int val = 80;              //defalt value for the speed
int del = 10;              //value in- and decrease by command "+" and "-"
int minimumVal = 70;       //minimum speed / value
int maximumVal = 250;      //maximum speed / value
bool flag_go_right = false; //becomes true when the command "go right" is issued
bool flag_go_left = false;  //becomes true when the command "go left" is issued

//-------------------------------------------------------------
//  setup function
//-------------------------------------------------------------
void setup()
{
  Serial.begin(9600);
  pinMode(en_a, OUTPUT);
  pinMode(right_a, OUTPUT);
  pinMode(left_a, OUTPUT);
  digitalWrite(en_a, LOW);
  analogWrite(right_a, 0);
  analogWrite(left_a, 0);
  delay(900);
}
//-------------------------------------------------------------
// loop Function
//-------------------------------------------------------------
void loop()
{
  if(Serial.available()>0)
  {
    message = Serial.readString();
    Serial.println(" --> Incoming message: " + message ); //You see the incoming message

    if(message.equals("stop\n"))
    {
```

```
    digitalWrite(en_a, LOW);
          analogWrite(right_a, 0);
    analogWrite(left_a, 0);
    flag_go_right = false;
    flag_go_left = false;
    Serial.println(" <-- Outgoing message: stop \n");
   }
   else if(message.equals("go right\n"))  //check message if "go right"
   {
    digitalWrite(en_a, HIGH);          //set enable Pin of the Modul to HIGH
          analogWrite(right_a, val);         //write the value to the PWM Pin of turn right
    analogWrite(left_a, 0);            //write zero to the PWM Pin of turn left
    flag_go_left = false;              //set the flags
    flag_go_right = true;
    Serial.println(" <-- Outgoing message: go right\n");  //feedback
   }
   else if(message.equals("go left\n"))
   {
    digitalWrite(en_a, HIGH);
          analogWrite(right_a, 0);
    analogWrite(left_a, val);
    flag_go_right = false;
    flag_go_left = true;
    Serial.println(" <-- Outgoing message: go left\n");
   }
   else if(message.equals("+\n"))      //check message if "+"
   {
    if(val >= maximumVal)          // reach maximum value
    {
     Serial.println(" <-- Outgoing message: Maximum value! \n");
    }
    else if (val < maximumVal)
    {
     val = val + del;    //increase the value
     if(flag_go_right)   // check witch direction should be increase
     {
      analogWrite(right_a, val);
     }
     else if(flag_go_left)
     {
      analogWrite(left_a, val);
     }
     Serial.print(" <-- Outgoing message: value = "); Serial.println(val); Serial.println("");
//feedback
    }
   }
   else if(message.equals("-\n"))
   {
    if(val <= minimumVal)                      // reach Minimum Speed
```

```
    {
      Serial.println(" <-- Outgoing message: Minimum value! \n");
    }
    else if (val > minimumVal)
    {
      val = val - del;
      if(flag_go_right)
      {
        analogWrite(right_a, val);
      }
      else if(flag_go_left)
      {
        analogWrite(left_a, val);
      }
      Serial.print(" <-- Outgoing message: value = "); Serial.println(val); Serial.println("");
    }
  }
  else //generate message if the message is unknown
  {
    Serial.println("<<-- Incoming message " + message + "      IS UNKNOWN \n\n");
  }
 }
}
```

Τίτλος κυκλώματος 3: "Κίνηση δύο κινητήρων DC "

Περιγραφή κυκλώματος: Το πρόγραμμα ελέγχει δύο κινητήρες DC. Οι εντολές ελέγχου δίνονται από το σειριακό παράθυρο (serial monitor).

The control command:

Write into the Serial Monitor	*Action*
stop	The motor stops
stop a	Stop Motor a
stop b	Stop Motor b
go right a	Motor a turn right
go left a	Motor a turn left
go right b	Motor b turn right
go left b	Motor b turn left

+	Increase speed
-	Decrease speed

Πρόγραμμα:

// two DC-Motors advance

```
/*
 Connection:
Arduino  | MotorShielt
PIN      | PIN
------------------------
+5V      | VSIGNAL
GND      | GND
4        | ENA
5        | PWMA+
6        | PWMA-
8        | ENB
9        | PWMB+
10       | PWMB-
*/

//-------------------------------------------------------------
// variables
//-------------------------------------------------------------
String message = "";

int en_a = 4;
int right_a = 5;
int left_a = 6;
bool flag_go_right_a = false;
bool flag_go_left_a = false;

int en_b = 8;
int right_b = 9;
int left_b = 10;
bool flag_go_right_b = false;
bool flag_go_left_b = false;

int val = 255;
int del = 10;
int minimumVal = 70;
int maximumVal = 250;

//-------------------------------------------------------------
// setup Function
//-------------------------------------------------------------
```

```
void setup()
{
  Serial.begin(9600);

  pinMode(en_a, OUTPUT);
  pinMode(right_a, OUTPUT);
  pinMode(left_a, OUTPUT);

  pinMode(en_b, OUTPUT);
  pinMode(right_b, OUTPUT);
  pinMode(left_b, OUTPUT);

  digitalWrite(en_a, LOW);
  analogWrite(right_a, 0);
  analogWrite(left_a, 0);

  digitalWrite(en_b, LOW);
  analogWrite(right_b, 0);
  analogWrite(left_b, 0);

  delay(900);
}

//-------------------------------------------------------------
// loop Function
//-------------------------------------------------------------
void loop()
{
  if(Serial.available()>0)
  {
    message = Serial.readString();
    Serial.println(" --> Incoming message: " + message);

    if(message.equals("stop\n"))                    // both motors stop
    {
      digitalWrite(en_a, LOW);
            analogWrite(right_a, 0);
      analogWrite(left_a, 0);
      flag_go_right_a = false;
      flag_go_left_a = false;

      digitalWrite(en_b, LOW);
      analogWrite(right_b, 0);
      analogWrite(left_b, 0);
      flag_go_right_b = false;
      flag_go_left_b = false;

      Serial.println(" <-- Outgoing message: stop all\n");
    }
```

```
else if(message.equals("stop a\n"))                 // motor a stop
{
  digitalWrite(en_a, LOW);
  analogWrite(right_a, 0);
  analogWrite(left_a, 0);
  flag_go_right_a = false;
  flag_go_left_a = false;
  Serial.println(" <-- Outgoing message: stop a\n");
}
else if(message.equals("stop b\n"))                 // motor b stop
{
  digitalWrite(en_b, LOW);
  analogWrite(right_b, 0);
  analogWrite(left_b, 0);
  flag_go_right_b = false;
  flag_go_left_b = false;
  Serial.println(" <-- Outgoing message: stop b\n");
 }
else if(message.equals("go right a\n"))
{
  digitalWrite(en_a, HIGH);
        analogWrite(right_a, val);
  analogWrite(left_a, 0);
  flag_go_left_a = false;
  flag_go_right_a = true;
  Serial.println(" <-- Outgoing message: go right a\n");
}
else if(message.equals("go right b\n"))
{
  digitalWrite(en_b, HIGH);
  analogWrite(left_b, 0);
  analogWrite(right_b, val);
  flag_go_left_b = false;
  flag_go_right_b = true;
  Serial.println(" <-- Outgoing message: go right b\n");
}
else if(message.equals("go left a\n"))
{
  digitalWrite(en_a, HIGH);
        analogWrite(right_a, 0);
  analogWrite(left_a, val);
  flag_go_right_a = false;
  flag_go_left_a = true;
  Serial.println(" <-- Outgoing message: go left a\n");
}
else if(message.equals("go left b\n"))
{
  digitalWrite(en_b, HIGH);
  analogWrite(right_b, 0);
```

```
    analogWrite(left_b, val);
    flag_go_right_b = false;
    flag_go_left_b = true;
    Serial.println(" <-- Outgoing message: go left b \n");
   }
   else if(message.equals("+\n"))
   {
    if(val >= maximumVal)                    // reach maximum value
    {
     Serial.println(" <-- Outgoing message: Maximum value! \n");
    }
    else if (val < maximumVal)
    {
     val = val + del;
     if(flag_go_right_a) analogWrite(right_a, val);
     if(flag_go_right_b) analogWrite(right_b, val);
     if(flag_go_left_a) analogWrite(left_a, val);
     if(flag_go_left_b) analogWrite(left_b, val);
     Serial.print(" <-- Outgoing message: value = "); Serial.println(val); Serial.println("");
    }
   }
   else if(message.equals("-\n"))
   {
    if(val <= minimumVal)                    // reach Minimum Value
    {
     Serial.println(" <-- Outgoing message: Minimum value! \n");
    }
    else if (val > minimumVal)
    {
     val = val - del;
     if(flag_go_right_a) analogWrite(right_a, val);
     if(flag_go_right_b) analogWrite(right_b, val);
     if(flag_go_left_a) analogWrite(left_a, val);
     if(flag_go_left_b) analogWrite(left_b, val);

     Serial.print(" <-- Outgoing message: value = "); Serial.println(val); Serial.println("");
    }
   }
   else
   {
    Serial.println("  <<-- Incoming message " + message + "      IS UNKNOWN \n\n");
   }
  }
}
```

Τίτλος κυκλώματος 4: "Απλό πρόγραμμα step Motor "

Περιγραφή κυκλώματος: "Ενας κινητήρας step Motor κινείται με το φορά των δεικτών του ρολογιού (δεξιόστροφα) και αντίστροφα.

Πρόγραμμα:

```
// Simple step Motor program
/*
   Connection:
Arduino  | MotorShielt
PIN      | PIN
-------------------------
+5V      | VSIGNAL
GND      | GNDSIGNAL
4        | ENA
5        | PWMA+
6        | PWMA-
8        | ENB
9        | PWMB+
10       | PWMB-
*/

//-------------------------------------------------------------
//  librarys
//-------------------------------------------------------------
#include <Stepper.h>
// libraty for the stepper
//-------------------------------------------------------------
//  constants
//-------------------------------------------------------------
#define STEPS 200
// Step Angle of used stepmotor = 1,8 degrees
// 360 / 1,8° = 200

//-------------------------------------------------------------
//  class
//-------------------------------------------------------------
Stepper Motor(STEPS, 5,6,9,10);
  //create a new instance of the Stepper class
  //Parameter:
  //  STEPS -> the number fo steps in one revolution of the Motor
  //  5,6,9,10 -> used Pins

//-------------------------------------------------------------
// variables
```

```
//---------------------------------------------------------------
int spe = 50;

//---------------------------------------------------------------
// setup function
//---------------------------------------------------------------
void setup()
{
  Serial.begin(9600);
  pinMode(4, INPUT);
  pinMode(8, INPUT);
  digitalWrite(4 ,HIGH);    //activate ENA Pin of the MotorShielt
  digitalWrite(8 ,HIGH);    //activate ENB Pin of the MotorShielt

  Motor.setSpeed(spe);  //Object Motor get the speed in "rotation per minute"
}

void loop()
{
  Motor.step(100);    //Turns the motor a specific number of steps,
                      //at a speed determined by the most recent call to setSpeed();
                      //in this case 100 steps clockwise;
  delay(200);

  Motor.step(-50);    //50 steps counterclockwise;
  delay(200);
}
```

Τίτλος κυκλώματος 5: "Προχωρημένο πρόγραμμα step Motor"

Περιγραφή κυκλώματος: Το πρόγραμμα ελέγχει ένα κινητήρα step - motor. Οι εντολές ελέγχου δίνονται από το σειριακό παράθυρο (serial monitor).

Παράδειγμα: Στέλνοντας 100 μέσω της σειριακής οθόνης στο Arduino ο κινητήρας Step-Motor κάνει 100 βήματα. Εάν στείλετε -100 ο κινητήρας Step-Motor θα κάνει 100 βήματα προς την άλλη κατεύθυνση.

Πρόγραμμα:

```
// Advanced step Motor program
/*
   Connection:
Arduino  | MotorShielt
PIN      | PIN
-------------------------
+5V      | VSIGNAL
GND      | GNDSIGNAL
4        | ENA
5        | PWMA+
6        | PWMA-
8        | ENB
9        | PWMB+
10       | PWMB-
*/

//------------------------------------------------------------
//  librarys
//------------------------------------------------------------
#include <Stepper.h>
//------------------------------------------------------------
//  constants
//------------------------------------------------------------
#define STEPS 200
//------------------------------------------------------------
//  class
//------------------------------------------------------------
Stepper Motor(STEPS, 5,6,9,10);
//------------------------------------------------------------
// variables
//------------------------------------------------------------

int spe = 100;
String message = "";

//------------------------------------------------------------
```

```
// setup function
//-------------------------------------------------------------

void setup()
{
  Serial.begin(9600);
  pinMode(4, INPUT);
  pinMode(8, INPUT);
  digitalWrite(4 ,HIGH);
  digitalWrite(8 ,HIGH);
  Motor.setSpeed(spe);
}

void loop()
{
  if(Serial.available()>0)
  {
    message = Serial.readString();
    Serial.println("\n\n --> Incoming message: " + message );

    if(message.toInt())
    {
      Serial.print(" <-- Outgoing message:  Steps -> " + message);
      Motor.step(message.toInt());
    }
    else
    {
      Serial.println("  <<-- Incoming message " + message + "  !!! -->  is not a number <-- !!!
\n\n");
    }
  }

}
```

Τίτλος κυκλώματος 6 (με κινητήρα Servo): ” Απλό πρόγραμμα κινητήρα servo ”

Περιγραφή κυκλώματος: Σε αυτό το πρόγραμμα ένας κινητήρας servo ελέγχεται από ένα σήμα PWM. Χρησιμοποιείτα ο βρόγχος επανάληψης **for**.

Πρόγραμμα:

```
// Simple servo Motor program
/*
   Connection:
Arduino  | MotorShielt
PIN      | PIN
-------------------------
+5V      | VSIGNAL
GND      | GNDSIGNAL
3        | PWMSERVOIN
*/
//-------------------------------------------------------------
// variables
//-------------------------------------------------------------
int del = 100;  // time for delay

//-------------------------------------------------------------
// setup function
//-------------------------------------------------------------
void setup()
{
 Serial.begin(9600);
 pinMode(3, OUTPUT);
}

void loop()
{
 for(int i = 0; i<255; i++)
   {
    analogWrite(3,i);
    Serial.println(i);
    delay(del);
  }

  delay(1000);
}
```

Co-funded by the Erasmus+ Programme of the European Union

Erasmus+ KA-202

Strategic Partnerships Project for Vocational Education and Training

Project Title: “Teaching and Learning Arduinos in Vocational Training”

Project Acronym: “ ARDUinVET ”

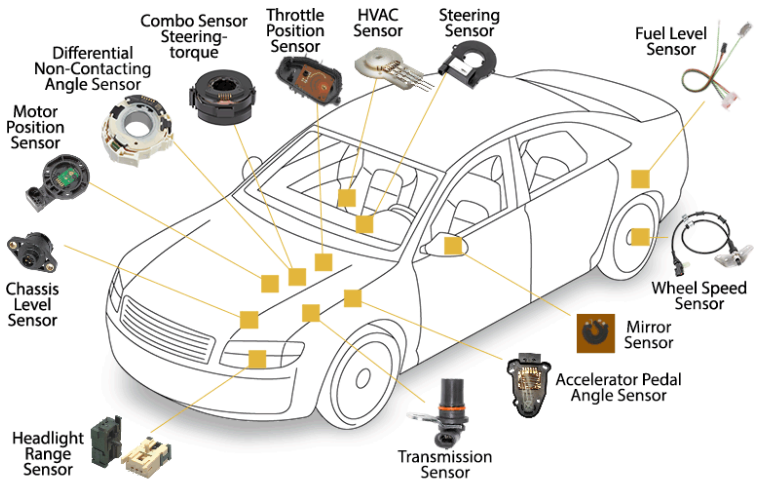
Project No: “2020-1-TR01-KA202-093762”

Αισθητήρες και εκπαιδευτικό κιτ

Αισθητήρες

Ο αισθητήρας είναι μια συσκευή, μονάδα ή υποσύστημα της Ηλεκτρονικής, σκοπός της οποίας είναι να δημιουργήσει μια διεπαφή μεταξύ ενός κυκλώματος και του περιβάλλοντος του. Είναι το σύστημα που λαμβάνει πληροφορίες από το περιβάλλον, τον φυσικό κόσμο, και στέλνει αυτές τις πληροφορίες ως τιμή στο κύκλωμα. Στην πραγματικότητα ανιχνεύει γεγονότα ή αλλαγές που συμβαίνουν στο περιβάλλον. Για να ανιχνεύσει αυτά τα συμβάντα ή αλλαγές, ο αισθητήρας πρέπει να μετρήσει ορισμένες τιμές σε φυσική κλίμακα. Μετρά μια ιδιότητα, όπως η πίεση, η θέση, η θερμοκρασία ή η επιτάχυνση και αποκρίνεται με ανάδραση. Στη συνέχεια μετατρέπει αυτές τις τιμές σε ένα ηλεκτρικό -συνήθως- σήμα.

Έτσι, ένας αισθητήρας χρησιμοποιείται πάντα με άλλα ηλεκτρονικά. Οι αισθητήρες μπορούν να βρεθούν σε πολλά καθημερινά αντικείμενα, όπως συσκευές ευαίσθητες στην αφή ή κίνηση, συσκευές που λειτουργούν με φως ή ήχο κ.λπ. Η χρήση αισθητήρων έχει επεκταθεί πέρα από τα παραδοσιακά πεδία μέτρησης θερμοκρασίας, πίεσης ή ροής, π.χ. Αισθητήρες GPS. Οι αναλογικοί αισθητήρες εξακολουθούν να χρησιμοποιούνται ευρέως. Οι εφαρμογές περιλαμβάνουν την κατασκευή και τα μηχανήματα, τα αεροπλάνα και την αεροδιαστημική, τα αυτοκίνητα, την ιατρική, τη ρομποτική και πολλές άλλες πτυχές της καθημερινής μας ζωής.



Εικόνα 1. Αισθητήρες αυτοκινήτου

Χαρακτηριστικά αισθητήρων

Ένας καλός αισθητήρας υπακούει στους ακόλουθους κανόνες:

- πρέπει να είναι ευαίσθητος στη μετρούμενη ιδιότητα
- δεν πρέπει να είναι ευαίσθητος σε οποιαδήποτε άλλη ιδιότητα που ενδέχεται να συναντήσει στην εφαρμογή του
- δεν πρέπει να επηρεάζει τη μετρούμενη ιδιότητα.

Τα βασικότερα χαρακτηριστικά των αισθητήρων είναι:

- **Γραμμικότητα:** Ο αισθητήρας έχει μια ιδιότητα ή χαρακτηριστικό, της οποίας η τιμή αλλάζει, όταν αλλάζει και η φυσική ποσότητα που μετρά. Είναι επιθυμητό οι παραλλαγές του στη μέτρηση της φυσικής ποσότητας να προκαλούν αισθητές αλλαγές στην ιδιότητα του αισθητήρα. Αυτή η ιδιότητα ονομάζεται γραμμικότητα και είναι κύριας σημασίας.
- **Ακρίβεια:** Η εγγύτητα της τιμής εξόδου στην τιμή εισόδου.
- **Σφάλμα:** Η διαφορά μεταξύ της μετρούμενης τιμής και της πραγματικής τιμής.
- **Ανοχή:** Το μέγιστο σφάλμα που μπορεί να δημιουργήσει ο αισθητήρας.
- **Full - Scale Output (FSO):** Ορίζει τις τιμές που μπορεί να λάβει η τάση ή το ρεύμα στην έξοδο του αισθητήρα
- **Ευαισθησία:** Εκφράζει πόσο ψηλά το σήμα εξόδου εξάγει τον αισθητήρα για κάθε μονάδα του μετρούμενου φυσικού μεγέθους
- **Ανάλυση:** Εκφράζει τη μικρότερη αλλαγή στο φυσικό μέγεθος που μπορεί να εντοπίσει ο αισθητήρας και κατά συνέπεια να αλλάξει την τιμή εξόδου του
- **Υστέρηση:** Ένα σφάλμα υστέρησης προκαλεί ότι η τιμή εξόδου ποικίλλει ανάλογα με τις προηγούμενες τιμές εισόδου. Εάν η έξοδος ενός αισθητήρα είναι διαφορετική, ανάλογα με το αν μια συγκεκριμένη τιμή εισόδου επιτεύχθηκε με αύξηση έναντι μείωσης της εισόδου, τότε ο αισθητήρας έχει σφάλμα υστέρησης
- **Καθυστέρηση:** Είναι η καθυστέρηση της αλλαγής της τιμής εξόδου μετά από αλλαγή εισόδου.
- **Νεκρή ζώνη:** Το μέγιστο ποσό αλλαγής της τιμής εισόδου που δεν επηρεάζει την τιμή εξόδου.

Κατηγορίες αισθητήρων

Υπάρχουν διάφοροι τρόποι κατηγοριοποίησης των αισθητήρων, μερικοί από τους οποίους παρατίθενται παρακάτω.

Η πρώτη κατηγοριοποιεί τους αισθητήρες ανάλογα με τη μορφή της τιμής εξόδου, διαιρώντας τους αναλογικούς και ψηφιακούς αισθητήρες.

Η δεύτερη κατηγοριοποίηση αφορά το τι μπορεί να μετρήσει ένας αισθητήρας, με μια πιο σημαντική διάκριση μεταξύ φυσικών και χημικών αισθητήρων. Οι φυσικοί αισθητήρες ελέγχουν φυσικά μεγέθη όπως η θέση, η μάζα, το ρεύμα, ο χρόνος και τα σχετικά μεγέθη τους, ενώ οι χημικοί αισθητήρες ελέγχουν την παρουσία διαφορετικών αερίων σε μια συγκεκριμένη ατμόσφαιρα.

Ένας άλλος τρόπος σχετίζεται με υλικά στις φυσικές ιδιότητες των οποίων λειτουργεί ο αισθητήρας, με κύριες κατηγορίες αισθητήρες με αγώγιμα, ημιαγωγά, διηλεκτρικά, μαγνητικά και υπεραγώγιμα υλικά.

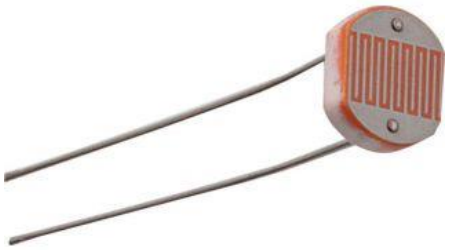
Τέλος, μια ακόμη μέθοδος ταξινόμησης αναφέρεται στο πεδίο χρήσης του αισθητήρα, με μεγάλες κατηγορίες όπως βιομηχανικούς, ιατρικούς, στρατιωτικούς, περιβαλλοντικούς αισθητήρες, καθώς και αισθητήρες για εφαρμογές μεταφοράς και αυτοματισμού.

Εικόνα 2: Διάφοροι αισθητήρες

Βασικοί αισθητήρες

Ακολουθούν ορισμένοι βασικοί αισθητήρες για κοινές εφαρμογές:

Photo Resistor LDR: Η αντίσταση είναι μια μεταβλητή αντίσταση της οποίας η τιμή αλλάζει ανάλογα με το φως που πέφτει πάνω της.



Εικόνα 3: Photo Resistor

Ψηφιακός αισθητήρας φωτεινότητας / Lux / Φωτός: Ο αισθητήρας φωτεινότητας TSL2561 είναι ένας προηγμένος ψηφιακός αισθητήρας φωτός, ιδανικός για χρήση σε ένα ευρύ φάσμα καταστάσεων φωτός. Αυτός ο αισθητήρας είναι πιο ακριβής, επιτρέποντας ακριβείς υπολογισμούς lux και μπορεί να διαμορφωθεί για διαφορετικά εύρη απολαβής/χρονισμού για ανίχνευση εύρους φωτός από 0,1 - 40.000+ Lux εν κινήσει. Περιέχει διόδους υπερύθρων και πλήρους φάσματος! Αυτό σημαίνει ότι μπορείτε να μετρήσετε ξεχωριστά το υπέρυθρο, το πλήρες φάσμα ή το ορατό από τον άνθρωπο φως.

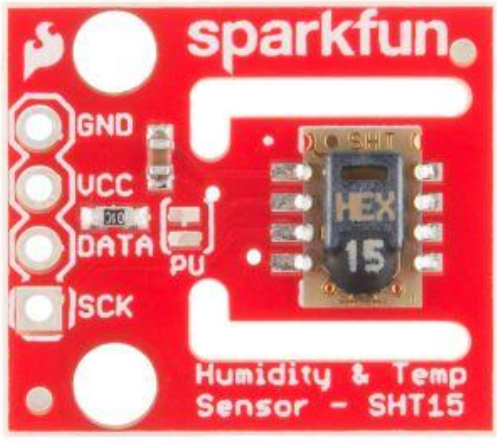
Εικόνα 3: Ψηφιακός αισθητήρας φωτεινότητας / Lux / Φωτός

Αισθητήρας θερμοκρασίας (με αναλογική έξοδο): Το εύρος θερμοκρασίας που μετρά είναι συνήθως -40 ° C έως + 100 ° C με ακρίβεια ± 1 ° C.

Εικόνα 4: Αισθητήρας θερμοκρασίας

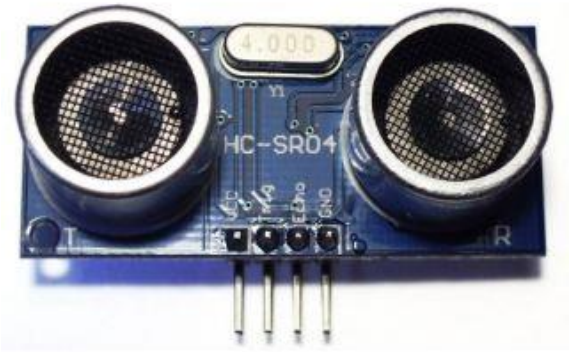
Αισθητήρας υγρασίας και θερμοκρασίας: υψηλής ακρίβειας, ψηφιακός αισθητήρας θερμοκρασίας και υγρασίας. Διαθέτει εύρος μέτρησης RH 0-100% με ακρίβεια θερμοκρασίας +/- 0,3°C @ 25°

C.



Εικόνα 5: Αισθητήρας υγρασίας και θερμοκρασίας

Αισθητήρας υπερύθρων: Χρησιμοποιείται για τον υπολογισμό της απόστασης. Η απόσταση που μπορείτε να υπολογίσετε είναι από 2 cm. έως 400 εκ. με ακρίβεια ενός εκατοστού.



Εικόνα 6: Αισθητήρας υπερήχων

Αισθητήρας κίνησης: Έχει την ικανότητα να ανιχνεύει την κίνηση ενός ανθρώπου ή ενός κατοικίδιου μέσα σε ένα δωμάτιο εντός έξι μέτρων. Ο αισθητήρας διαθέτει δύο αντιστάσεις trimmer όπου μπορεί να ρυθμιστεί η ευαισθησία και ο χρόνος ενεργοποίησης από τη στιγμή που ανιχνεύσει κίνηση.

Εικόνα 7: Αισθητήρας κίνησης

Αισθητήρας κραδασμών: Όταν ο αισθητήρας ανιχνεύσει κραδασμούς, δημιουργείται μια τάση, χρησιμοποιώντας αντίσταση 1 Mohm

Εικόνα 8: Αισθητήρας κραδασμών

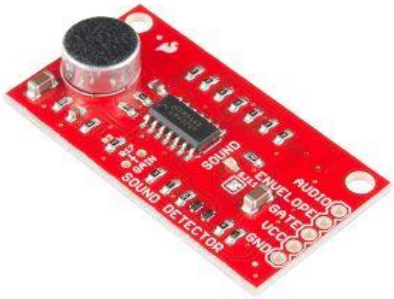
Επιταχυνσιόμετρο τριπλού άξονα και αισθητήρας γυροσκοπίου: Συνδυάζοντας ένα γυροσκόπιο 3 αξόνων και ένα επιταχυνσιόμετρο 3 αξόνων στην ίδια μήτρα πυριτίου μαζί με έναν ενσωματωμένο ψηφιακό επεξεργαστή κίνησης (DMP) ικανό να επεξεργάζεται σύνθετους αλγόριθμους Motion Fusion 9 αξόνων, ο αισθητήρας μπορεί να επιστρέψει όλες τις τιμές ευθυγράμμισης σταυροαξονικά.

Εικόνα 9: Επιταχυνσιόμετρο τριπλού άξονα και αισθητήρας γυροσκοπίου

Αισθητήρας αερίου: Ευαίσθητος σε υγραέριο, φυσικό αέριο, αέριο άνθρακα. Η τάση εξόδου αυξάνεται μαζί με την αύξηση της συγκέντρωσης των μετρούμενων αερίων.

Εικόνα 11: Αισθητήρας αερίου

Ανιχνευτής ήχου: Αυτός ο αισθητήρας παρέχει μια έξοδο ήχου, αλλά και μια δυαδική ένδειξη της παρουσίας ήχου και μια αναλογική αναπαράσταση του πλάτους του.



Εικόνα 12: Ανιχνευτής ήχου

ΕΡΓΟ 1- ΟΝΟΜΑ: ΑΙΣΘΗΤΗΡΑΣ ΥΠΕΡΗΧΩΝ ΣΕ ΣΗΜΑΤΟΔΟΤΗΣΗ ΚΥΚΛΟΦΟΡΙΑΣ

Σκοπός του Έργου: Σε αυτό το έργο οι μαθητές θα δουν πώς μπορούμε να χρησιμοποιήσουμε έναν αισθητήρα υπερήχων σε σήματα κυκλοφορίας χρησιμοποιώντας μια εξωτερική αντίσταση έλξης.

Μέσω αυτού του έργου, οι μαθητές θα μπορούν να πειραματιστούν με έναν αισθητήρα υπερήχων, μια συσκευή που μετρά την απόσταση από ένα αντικείμενο χρησιμοποιώντας ηχητικά κύματα.

Μεθοδολογία

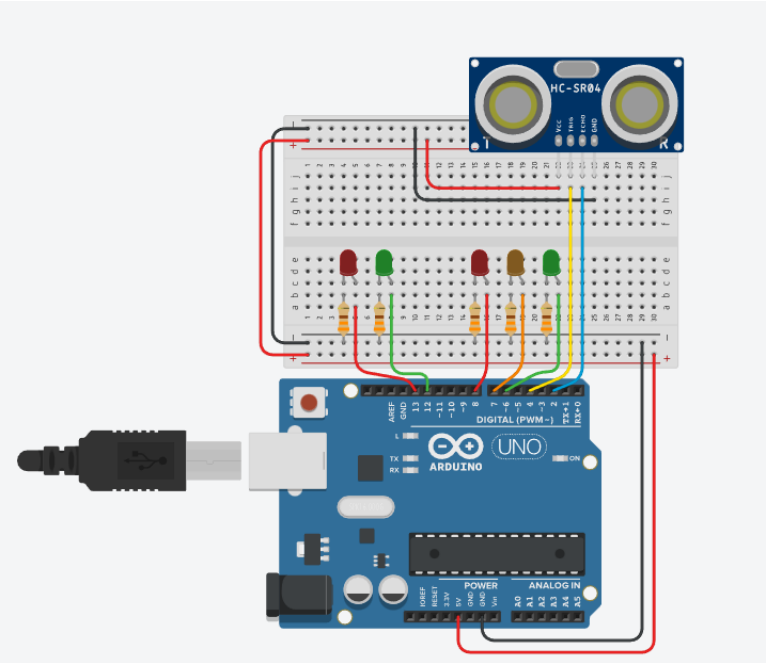
Αρχικά, περιγράφουμε την επιθυμητή ροή εργασιών του έργου. Στη συνέχεια ζητάμε από τους μαθητές να επιλέξουν τα απαραίτητα στοιχεία και να χρησιμοποιήσουν το Tinkercad για να σχεδιάσουν και να σχεδιάσουν το κύκλωμα. Θα χρησιμοποιήσουν το Εργαλείο κώδικα Tinkercad για να γράψουν τον κώδικα.

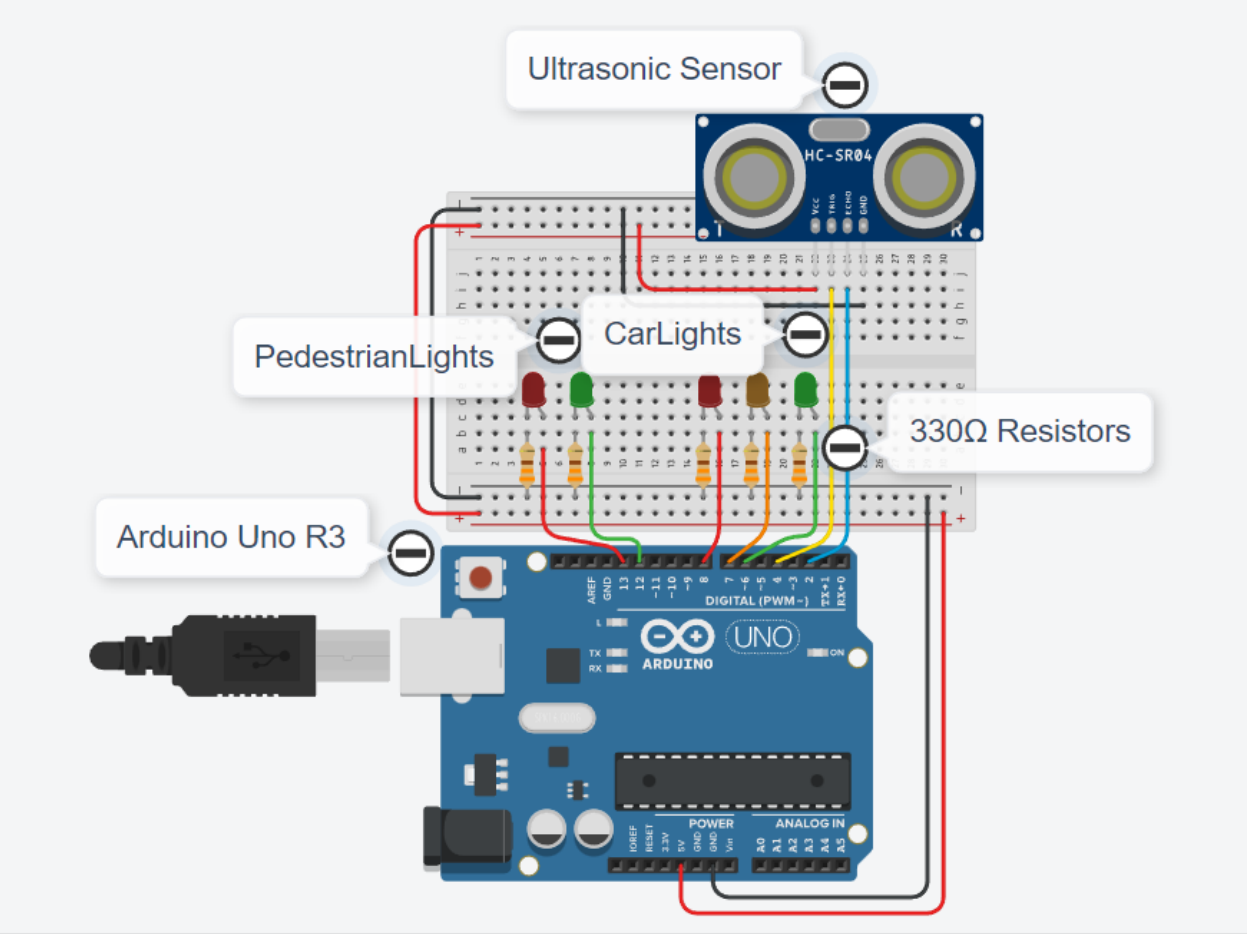
Στη συνέχεια θα δημιουργήσουν το κύκλωμα, θα γράψουν τον κώδικα στο εργαλείο λογισμικού Arduino και θα τον ανεβάσουν στην πλακέτα του Arduino.

Η προσομοίωση στο Tinkercad θα επαληθεύσει ότι το κύκλωμα κατασκευάστηκε σωστά.

Αυτό το έργο απευθύνεται σε μαθητές βασικού επιπέδου, οι οποίοι αρχίζουν να επωφελούνται από τα αποτελέσματα και τα αποτελέσματα από το έργο Στρατηγικών Συνεργασιών Erasmus+ KA-202 στην ΕΕΚ, που ονομάζεται «ArduinVET».

Κύκλωμα έργου.





Πως δουλεύει:

Ο αισθητήρας υπερήχων λειτουργεί στέλνοντας ένα ηχητικό κύμα σε συχνότητα υπερήχων και περιμένοντας να αναπηδήσει από το αντικείμενο. Η χρονική καθυστέρηση μεταξύ της μετάδοσης του ήχου και της λήψης του ήχου χρησιμοποιείται στη συνέχεια για τον υπολογισμό της απόστασης.

Οι μαθητές μπορούν να ενεργοποιήσουν τον αισθητήρα τοποθετώντας το χέρι τους μπροστά από τον αισθητήρα, με αποτέλεσμα να μειωθεί η χρονική καθυστέρηση μεταξύ της μετάδοσης του ήχου και της λήψης του ήχου. Η απόσταση θα εμφανιστεί στη σειριακή οθόνη. Οι μαθητές μπορούν να δουν την ενεργοποίηση των φωτεινών σηματοδοτών παρατηρώντας τα φώτα.

Στο κύκλωμα, μαζί με τον αισθητήρα υπερήχων χρησιμοποιούμε μια εξωτερική αντίσταση έλξης.

- Η pull-up αντίσταση διατηρεί το Pin 3 μόνιμα σε κατάσταση HIGH (+5V).
- Όταν ενεργοποιηθεί ο αισθητήρας υπερήχων, ο ακροδέκτης 3 γειώνεται (κατάσταση ΧΑΜΗΛΟ ΔΥΝΑΜΙΚΟ) στιγμιαία.

Πίνακας χρονοδιαγράμματος

Φάσεις	Οχήματα	πεζοί	Χρόνος	Περιγραφή
0				Ο αισθητήρας δεν έχει ενεργοποιηθεί
1			3sec	Μόλις ενεργοποιηθεί ο αισθητήρας, ενεργοποιούνται οι φάσεις 1 έως 4 και στη συνέχεια η συσκευή επιστρέφει στην προηγούμενη κατάστασή της (Φάση 0)
2			3sec	
3			4sec	
4			3sec	
0				Μέχρι να ενεργοποιηθεί ξανά ο αισθητήρας

Λίστα εξαρτημάτων:

Τα εξαρτήματά μας είναι:

- Ultrasonic Sensor HC-SR04
- Breadboard and Jump Wires
- Wire USB
- Arduino UNO R3
- 5XLED
- 5X330 ohm

Arduino Code of Project:

```
// Όνομα έργου: ΑΙΣΘΗΤΗΡΑΣ ΥΠΕΡΗΧΩΝ ΣΕ ΦΑΝΑΡΙΑ
// Σηματοδότηση οδικής κυκλοφορίας με φανάρι πεζών και αισθητήρα υπερήχων
```

```
// Δήλωση σταθερών
  const byte greenCar = 6;
  const byte orangeCar = 7;
  const byte redCar = 8;
  const byte greenPedestrian = 12;
  const byte redPedestrian = 13;
// Δήλωση παραμέτρων
  boolean buttonOn=false;

// Parameter declaration Ultrasonic Sensor HC-SR04 * ******
// defines pins numbers
  const int trigPin = 4;  //D4
  const int echoPin = 2;  //D3


void setup() {
 // Αρχικοποίηση Σταθερών Ultrasonic Sensor HC-SR04 * ******
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.begin(9600); // Starts the serial communication
 // αρχικοποίηση παραμέτρων
  pinMode(greenCar,OUTPUT);
  pinMode(orangeCar,OUTPUT);
  pinMode(redCar,OUTPUT);

  pinMode(greenPedestrian,OUTPUT);
  pinMode(redPedestrian,OUTPUT);

  // Εκκίνηση τιμών για σηματοδότηση πεζών
  digitalWrite(greenPedestrian,LOW);
  digitalWrite(redPedestrian,HIGH);

  // Εκκίνηση τιμών για σηματοδότηση οχημάτων
  digitalWrite(greenCar,HIGH);
  digitalWrite(orangeCar,LOW);
  digitalWrite(redCar,LOW);
  pinMode (2, INPUT_PULLUP);
}

void loop() {
 //****** * 3. Code Ultrasonic Sensor HC-SR04 * ******
 // Clears the trigPin
  delay(500);
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);

// Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
```

```
  digitalWrite(trigPin, LOW);

// Διαβάζει το echoPin, επιστρέφει το χρόνο διαδρομής του ηχητικού κύματος σε
//microseconds
  const long duration = pulseIn(echoPin, HIGH);
  Serial.print("Duration: ");
  Serial.println(duration);
// Υπολογισμός της απόστασης
  const long distance= duration/58.2;
// Εκτυπώνει την απόσταση στη σειριακή οθόνη
  Serial.print("Distance: ");
  Serial.println(distance);

  delay(2000);
  if (distance<20){
     buttonOn=true;
  }
  if(buttonOn == true)
    {
     buttonOn = false;

     delay(3000);
     digitalWrite(greenCar,LOW);
     digitalWrite(orangeCar,HIGH);

     delay(3000);
     digitalWrite(orangeCar,LOW);
     digitalWrite(redCar,HIGH);

     digitalWrite(greenPedestrian,HIGH);
     digitalWrite(redPedestrian,LOW);

     delay(4000);
     digitalWrite(greenPedestrian,LOW);
     digitalWrite(redPedestrian,HIGH);

     delay(3000);
     digitalWrite(greenCar,HIGH);
     digitalWrite(redCar,LOW);

     delay(5000);
    }else{
     digitalWrite(greenPedestrian,LOW);
     digitalWrite(redPedestrian,HIGH);

     digitalWrite(greenCar,HIGH);
     digitalWrite(orangeCar,LOW);
     digitalWrite(redCar,LOW);
     }
```

}

ΕΡΓΟ 2 - ΟΝΟΜΑ: ΣΥΣΤΗΜΑ ΣΥΝΑΓΕΡΜΟΥ

Σκοπός του Έργου: Σε αυτό το έργο, οι μαθητές θα δουν πώς λειτουργεί ένα σύστημα συναγερμού που μπορεί να ανιχνεύσει υγρό αέριο, κίνηση σε κλειστό χώρο καθώς και αδικαιολόγητη αύξηση της θερμοκρασίας. Το σύστημα συναγερμού θα ελέγχεται από ένα κουμπί και έναν αισθητήρα φωτοαντίστασης. Πατώντας το κουμπί οι μαθητές μπορούν να ενεργοποιήσουν το σύστημα συναγερμού. Όταν ηχεί ο βομβητής πατώντας το κουμπί μπορούν να τον σταματήσουν. Όταν συσχετίζουν έναν ενεργοποιημένο αισθητήρα με το πάτημα του κουμπιού, μπορούν να απενεργοποιήσουν το σύστημα συναγερμού. Η φωτοαντίσταση ενεργοποιεί το σύστημα συναγερμού σε περίπτωση που το σύστημα συναγερμού δεν είναι ενεργοποιημένο και ο φωτισμός είναι χαμηλός.

Μέσα από αυτό το έργο, οι μαθητές θα μπορούσαν να πειραματιστούν με:

- ✓ μια μονάδα αισθητήρα ενεργού βομβητή που έχει ενσωματωμένο κύκλωμα ταλαντωτή που παράγει σταθερή συχνότητα ήχου. Ανάβει και σβήνει με μια καρφίτσα Arduino
- ✓ ένας αισθητήρας υπερήχων, μια συσκευή που μετρά την απόσταση από ένα αντικείμενο χρησιμοποιώντας ηχητικά κύματα
- ✓ έναν αισθητήρα αερίου και
- ✓ δύο ενδεικτικές λυχνίες και ένα κουμπί
- ✓ αισθητήρα φωτοαντίστασης
- ✓ αισθητήρα θερμοκρασίας

Μεθοδολογία

Αρχικά, περιγράφουμε την επιθυμητή ροή εργασίας του έργου που θα αναπτυχθεί σε βήματα. Στη συνέχεια ζητείται από τους μαθητές να επιλέξουν τα απαραίτητα στοιχεία για να κατασκευάσουν το κύκλωμα.

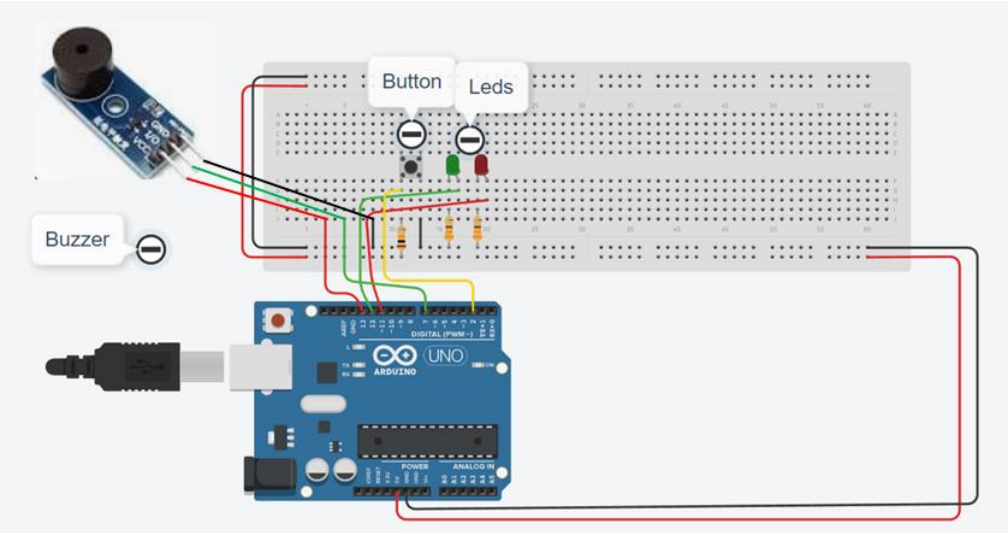
Στη συνέχεια θα αναπτύξουν τον κώδικα σε περιβάλλον Arduino IDE και θα τον ανεβάσουν στην πλακέτα του Arduino.

Οι μαθητές θα επαληθεύσουν ότι το κύκλωμα κατασκευάστηκε σωστά.

Αυτό το έργο απευθύνεται σε μαθητές βασικού επιπέδου, οι οποίοι αρχίζουν να επωφελούνται από τα αποτελέσματα και τα αποτελέσματα από το έργο Στρατηγικών Συνεργασιών Erasmus+ KA-202 στην ΕΕΚ, που ονομάζεται «ArduinVET».

Βήμα 1 (Βομβητής με διαφορετικό τόνο ήχου στο σύστημα συναγερμού)

Κύκλωμα Έργου.



Πως δουλεύει:

Μια μονάδα αισθητήρα ενεργού βομβητή έχει ενσωματωμένο κύκλωμα ταλάντωσης, επομένως η συχνότητα του ήχου είναι σταθερή. Είναι σε θέση να παράγει τον ίδιο τον ήχο.
Μέσα σε αυτό το έργο ο τόνος του ήχου παύει να έχει τον ίδιο ήχο μέσω κώδικα. Μόλις πατηθεί το κουμπί, αρχίζει να ηχεί με μια ακίδα Arduino, το κόκκινο Led που περιμένει σβήνει και το πράσινο Led ανάβει
Στο κύκλωμα, μαζί με τον αισθητήρα υπερήχων χρησιμοποιούμε μια εξωτερική αντίσταση έλξης,

- Η pull-up αντίσταση διατηρεί το Pin 2 μόνιμα σε κατάσταση HIGH (+5V).
- Όταν πατηθεί το κουμπί, ο ακροδέκτης 2 γειώνεται (κατάσταση ΧΑΜΗΛΗ) στιγμιαία.

Λίστα εξαρτημάτων:

Τα εξαρτήματά μας είναι:

- Buzzer
- Breadboard and Jump Wires
- Wire USB
- Arduino UNO R3
- 2XLED
- 2X330 ohm
- Button
- 1X10Kohm

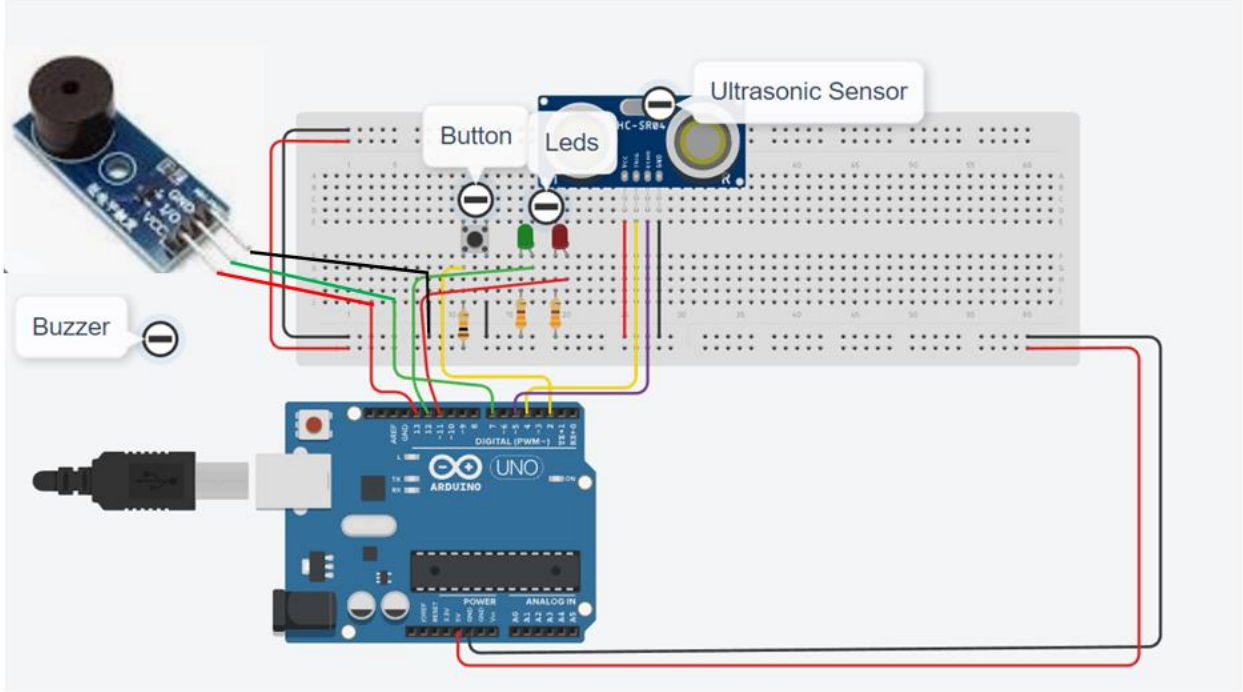
Arduino Code of Step 1:

```
//Project Name: Alarm System

// Step 1  (Buzzer with different sound tone)

//Declaration-define of variables - constants
int buzz= 7; // I/O-pin from buzzer Module connects here
int buzzVcc= 13; //Vcc-pin from buzzer Module connects here
const int wpm = 20; // Morse speed in WPM
const int dotL = 1200/wpm; // Calculated dot-length
const int dashL = 3*dotL; // Dash = 3 x dot
const int sPause = dotL; // Symbol pause = 1 dot
const int lPause = dashL; // Letter pause = 3 dots
const int wPause = 7*dotL; // Word pause = 7 dots

int red_led=12;
int green_led=11;

boolean buttonOn=false;

void setup()
{
  // Initialization of variables - constants
  pinMode(buzz,OUTPUT);
  pinMode(buzzVcc,OUTPUT);

  pinMode(red_led,OUTPUT);
  pinMode(green_led,OUTPUT);
  digitalWrite(red_led, HIGH);

  pinMode (2, INPUT_PULLUP);
}

void loop(){

  if (digitalRead(2)== LOW){
     buttonOn=true;
     }

    //Buzzer
    if(buttonOn==true){

      digitalWrite(red_led, LOW);
      digitalWrite(green_led, HIGH);

      digitalWrite(buzz, LOW); // Tone ON
```

```
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause
    digitalWrite(buzzVcc, HIGH);
    digitalWrite(buzz, LOW); // Tone ON
    delay(dotL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    digitalWrite(buzz, LOW); // Tone ON
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    digitalWrite(buzz, LOW); // Tone ON
    delay(dotL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    delay(lPause-sPause); // Subtracts pause already taken

    digitalWrite(buzz, LOW); // Tone ON
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause
    digitalWrite(buzz, LOW); // Tone ON
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    digitalWrite(buzz, LOW); // Tone ON
    delay(dotL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    digitalWrite(buzz, LOW); // Tone ON
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause
    delay(wPause-sPause); // Subtracts pause already taken
    delay(5000);
    //Variable return to their original status
    buttonOn=false;
    digitalWrite(buzz, LOW); // Tone ON
    digitalWrite(buzzVcc, LOW);
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, LOW);
  }
}
```

Βήμα 2 (Προστέθηκε αισθητήρας υπερήχων HC-SR04 στο σύστημα συναγερμού)

Συνδέστε τον αισθητήρα υπερήχων HC-SR04



Πως δουλεύει:

Ο αισθητήρας υπερήχων λειτουργεί στέλνοντας ένα ηχητικό κύμα σε συχνότητα υπερήχων και περιμένει να αναπηδήσει από το αντικείμενο. Η χρονική καθυστέρηση μεταξύ της μετάδοσης του ήχου και της λήψης του ήχου χρησιμοποιείται στη συνέχεια για τον υπολογισμό της απόστασης. Οι μαθητές μπορούν να ενεργοποιήσουν το σύστημα συναγερμού πατώντας το κουμπί, τότε το πράσινο led θα ανάψει και εάν τοποθετήσουν το χέρι τους μπροστά από τον αισθητήρα, η χρονική καθυστέρηση μεταξύ της μετάδοσης του ήχου και της λήψης του ήχου θα μειωθεί και το πράσινο led θα σβήσει, θα ανάψει το κόκκινο led και ο ηχητικός βομβητής θα αρχίσει να ηχεί. Η απόσταση θα εμφανιστεί στη σειριακή οθόνη. Οι μαθητές μπορούν επίσης να απενεργοποιήσουν το σύστημα συναγερμού πατώντας ξανά το κουμπί.

Arduino Code of Step 2:

```
// code of Step 2 - Added Ultrasonic Sensor HC-SR04
int red_led=12;
int green_led=11;
int sensorThres=400;
```

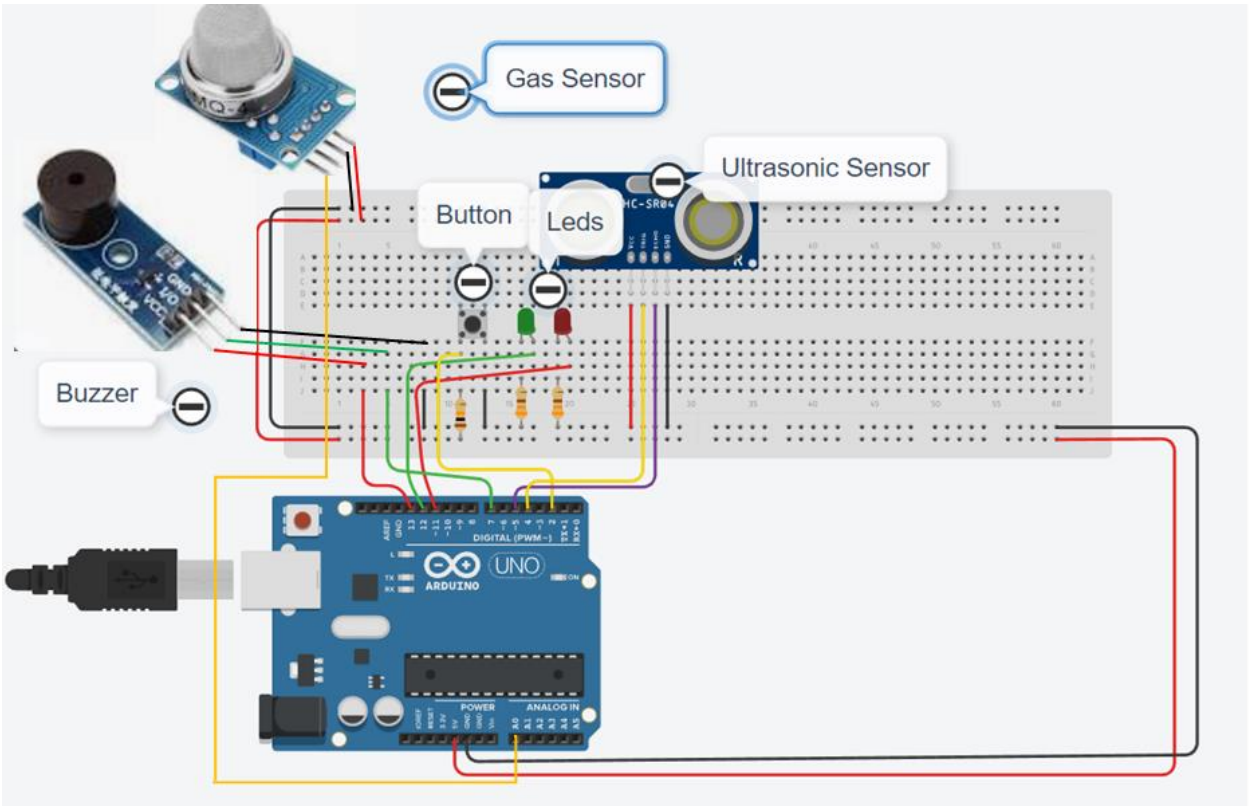
```
//buzzer
int buzz= 13; // I/O-pin from buzzer connects here
int buzzVcc= 7; //Vcc-pin from buzzer connects here
const int wpm = 20; // Morse speed in WPM
const int dotL = 1200/wpm; // Calculated dot-length
const int dashL = 3*dotL; // Dash = 3 x dot
const int sPause = dotL; // Symbol pause = 1 dot
const int lPause = dashL; // Letter pause = 3 dots
const int wPause = 7*dotL; // Word pause = 7 dots

//Ultrasonic Sensor
#define trigPin 4
#define echoPin 5

//Button
boolean buttonOn=false;

void setup()
{
pinMode(red_led,OUTPUT);
pinMode(buzz,OUTPUT);
pinMode(green_led,OUTPUT);

pinMode(buzz,OUTPUT); // Set buzzer-pin as output
pinMode(buzzVcc,OUTPUT); //Vcc Buzzer

//Ultrasonic Sensor
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);

pinMode (2, INPUT_PULLUP);

Serial.begin(9600);
}



void loop()
{
//Button
 if (digitalRead(2)== LOW){
     buttonOn=true;
     digitalWrite(green_led, HIGH);
     }

//Ultrasoc Sensor code
long duration, distance;
digitalWrite(trigPin, LOW);
```

```
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = (duration/2) / 29.1;

digitalWrite(buzzVcc, LOW);

if (distance < 20)
 {
 while(buttonOn==true){

  if (digitalRead(2)== LOW){
    buttonOn=false;
    digitalWrite(red_led, LOW);
    digitalWrite(green_led, LOW);
    digitalWrite( buzz, LOW);
 }else{
 digitalWrite(red_led, HIGH);
 digitalWrite(green_led, LOW);
 digitalWrite( buzz, HIGH);



 //buzzer
 digitalWrite(buzzVcc, HIGH);
 digitalWrite(buzz, LOW); // Tone ON
 delay(dashL); // Tone length
 digitalWrite(buzz, HIGH); // Tone OFF
 delay(sPause); // Symbol pause

 digitalWrite(buzz, LOW); // Tone ON
 delay(dotL); // Tone length
 digitalWrite(buzz, HIGH); // Tone OFF
 delay(sPause); // Symbol pause

 digitalWrite(buzz, LOW); // Tone ON
 delay(dashL); // Tone length
 digitalWrite(buzz, HIGH); // Tone OFF
 delay(sPause); // Symbol pause

 digitalWrite(buzz, LOW); // Tone ON
 delay(dotL); // Tone length
 digitalWrite(buzz, HIGH); // Tone OFF
 delay(sPause); // Symbol pause

 delay(lPause-sPause); // Subtracts pause already taken
```

```
 digitalWrite(buzz, LOW); // Tone ON
 delay(dashL); // Tone length
 digitalWrite(buzz, HIGH); // Tone OFF
 delay(sPause); // Symbol pause
 digitalWrite(buzz, LOW); // Tone ON
 delay(dashL); // Tone length
 digitalWrite(buzz, HIGH); // Tone OFF
 delay(sPause); // Symbol pause

 digitalWrite(buzz, LOW); // Tone ON
 delay(dotL); // Tone length
 digitalWrite(buzz, HIGH); // Tone OFF
 delay(sPause); // Symbol pause

 digitalWrite(buzz, LOW); // Tone ON
 delay(dashL); // Tone length
 digitalWrite(buzz, HIGH); // Tone OFF
 delay(sPause); // Symbol pause
 delay(wPause-sPause); // Subtracts pause already taken

  }  //end else
  } //end while
} //end if

delay(100);
}
```

Βήμα 3 (Προστέθηκε αισθητήρας αερίου στο σύστημα συναγερμού)

Συνδέστε τον αισθητήρα αερίου



Πως δουλεύει:

Όταν ο ανιχνευτής αερίου ανιχνεύσει αέριο και το σύστημα συναγερμού είναι ήδη ενεργοποιημένο, το πράσινο φως σβήνει, το κόκκινο φως ανάβει και ηχεί ο βομβητής.

Arduino Code of Step 3:

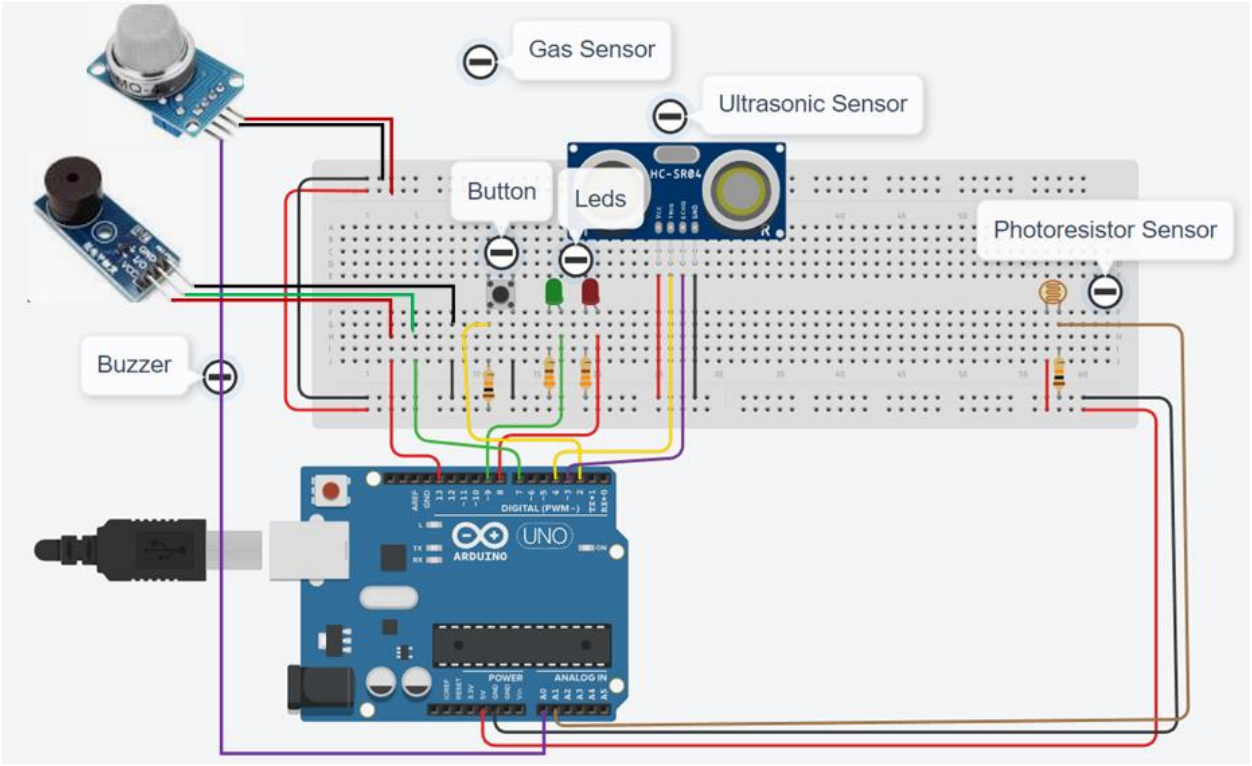
```
//WORK : ALARM SYSTEM
// ADDED AN GAS SENSOR IN SYSTEM
int red_led=11;
int green_led=12;
int gas_value;
int sensorThres=400;

//buzzer
```

```
int buzz= 13; // I/O-pin from buzzer connects here
int buzzVcc= 7; //Vcc-pin from buzzer connects here
const int wpm = 20; // Morse speed in WPM
const int dotL = 1200/wpm; // Calculated dot-length
const int dashL = 3*dotL; // Dash = 3 x dot
const int sPause = dotL; // Symbol pause = 1 dot
const int lPause = dashL; // Letter pause = 3 dots
const int wPause = 7*dotL; // Word pause = 7 dots

//Ultrasonic Sensor
#define trigPin 4
#define echoPin 5

//Button
boolean buttonOn=false;

void setup()
{
pinMode(red_led,OUTPUT);
pinMode(green_led,OUTPUT);
pinMode(A0,INPUT); //A0 Gass

pinMode(buzz,OUTPUT); // Set buzzer-pin as output
pinMode(buzzVcc,OUTPUT); //Vcc Buzzer

//Ultrasonic Sensor
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);

pinMode (2, INPUT_PULLUP);

Serial.begin(9600);
}

void loop()
{
//Button
 if (digitalRead(2)== LOW){
     buttonOn=true;
     digitalWrite(green_led, HIGH);
     Serial.println("The alarm system is enabled!");
     }

//Ultrasoc Sensor code
long duration, distance;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
```

```
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = (duration/2) / 29.1;


digitalWrite(buzzVcc, LOW);

gas_value=analogRead(A0);

if (gas_value > sensorThres or distance < 20)
 {
  while(buttonOn==true){
  if (digitalRead(2)== LOW){
     buttonOn=false;
     digitalWrite(red_led, LOW);
     digitalWrite(green_led, LOW);
     digitalWrite( buzz, LOW);
     Serial.println("The alarm system is disabled!");
  }else{
  digitalWrite(red_led, HIGH);
  digitalWrite(green_led, LOW);
  digitalWrite( buzz, HIGH);
  Serial.println("DANGER!!!!");
  Serial.print("Gas Value: ");
  Serial.println(gas_value);
  Serial.print("Distance: ");
  Serial.println(distance);

  //buzzer
  digitalWrite(buzzVcc, HIGH);
  digitalWrite(buzz, LOW); // Tone ON
  delay(dashL); // Tone length
  digitalWrite(buzz, HIGH); // Tone OFF
  delay(sPause); // Symbol pause

  digitalWrite(buzz, LOW); // Tone ON
  delay(dotL); // Tone length
  digitalWrite(buzz, HIGH); // Tone OFF
  delay(sPause); // Symbol pause

  digitalWrite(buzz, LOW); // Tone ON
  delay(dashL); // Tone length
  digitalWrite(buzz, HIGH); // Tone OFF
  delay(sPause); // Symbol pause

  digitalWrite(buzz, LOW); // Tone ON
  delay(dotL); // Tone length
  digitalWrite(buzz, HIGH); // Tone OFF
  delay(sPause); // Symbol pause
```

```
  delay(lPause-sPause); // Subtracts pause already taken

  digitalWrite(buzz, LOW); // Tone ON
  delay(dashL); // Tone length
  digitalWrite(buzz, HIGH); // Tone OFF
  delay(sPause); // Symbol pause
  digitalWrite(buzz, LOW); // Tone ON
  delay(dashL); // Tone length
  digitalWrite(buzz, HIGH); // Tone OFF
  delay(sPause); // Symbol pause

  digitalWrite(buzz, LOW); // Tone ON
  delay(dotL); // Tone length
  digitalWrite(buzz, HIGH); // Tone OFF
  delay(sPause); // Symbol pause

  digitalWrite(buzz, LOW); // Tone ON
  delay(dashL); // Tone length
  digitalWrite(buzz, HIGH); // Tone OFF
  delay(sPause); // Symbol pause
  delay(wPause-sPause); // Subtracts pause already taken
        } //end else
   }//end while
  } //end if

delay(100);
}
```

Βήμα 4 (Προσθήκη αισθητήρα φωτοαντίστασης στο σύστημα συναγερμού)



Συνδέστε τον αισθητήρα φωτοαντίστασης

Σε αυτό το βήμα θα χρειαστούμε έναν πομπό λέιζερ, μια φωτοαντίσταση για τον δέκτη και μια αντίσταση 1 KOhm για την προστασία του.

Πως δουλεύει:

Ο πομπός λέιζερ μαζί με τον αισθητήρα φωτοαντίστασης που λειτουργεί ως δέκτης δίνει εντολή στο Arduino να ενεργοποιήσει το σύστημα όταν για κάποιο λόγο διακόπτεται η λήψη. Επίσης, το Σύστημα Συναγερμού ενεργοποιείται όταν η τιμή του αισθητήρα θερμοκρασίας υπερβεί μια συγκεκριμένη τιμή.

Arduino Code of Step 4:

```
//WORK : ALARM SYSTEM
//ADDED A PHOTORESISTOR SENSOR IN SYSTEM
int red_led=8;
int green_led=9;
int gas_value;
int sensorThres=400;

//------------------buzzer-----------------------
```

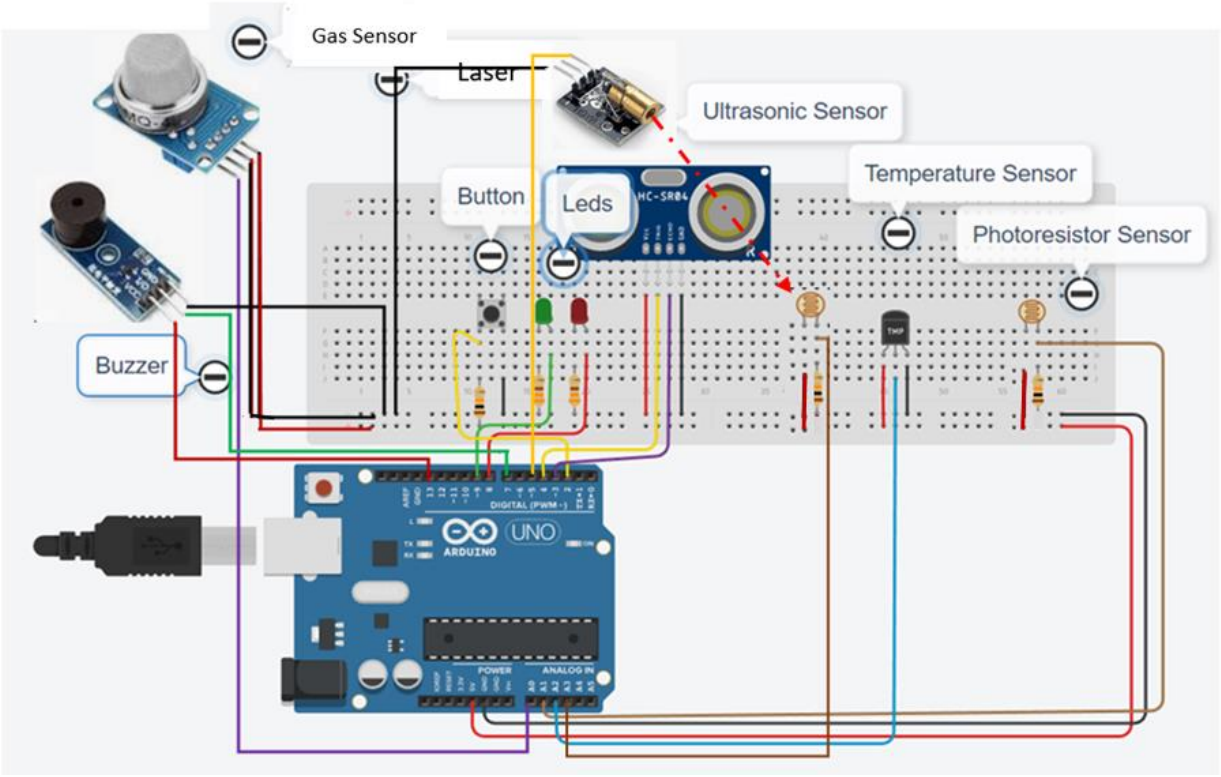
```
int buzz= 13; // I/O-pin from buzzer connects here
int buzzVcc= 7; //Vcc-pin from buzzer connects here
const int wpm = 20; // Morse speed in WPM
const int dotL = 1200/wpm; // Calculated dot-length
const int dashL = 3*dotL; // Dash = 3 x dot
const int sPause = dotL; // Symbol pause = 1 dot
const int lPause = dashL; // Letter pause = 3 dots
const int wPause = 7*dotL; // Word pause = 7 dots

//------------Ultrasonic Sensor--------------------
#define trigPin 4
#define echoPin 5

//------------------Button-------------------------
boolean buttonOn=false;

void setup()
{
pinMode(red_led,OUTPUT);
pinMode(green_led,OUTPUT);
digitalWrite(red_led, LOW);
digitalWrite(green_led, LOW);
pinMode(A0,INPUT); //A0 Gass

pinMode(buzz,OUTPUT); // Set buzzer-pin as output
pinMode(buzzVcc,OUTPUT); //Vcc Buzzer

//Ultrasonic Sensor
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);

pinMode (2, INPUT_PULLUP);

Serial.begin(9600);
}
void loop()
{
  //------------photoresistor---------------
  int valuePhotor = analogRead(A1);
   Serial.println("Analog valuePhotor : ");
   Serial.println(valuePhotor);
   delay(250);

  //------------------Button------------------
   if (digitalRead(2)== LOW or valuePhotor<110){
       buttonOn=true;
       digitalWrite(red_led, LOW);
       digitalWrite(green_led, HIGH);
       Serial.println("The alarm system is enabled!");
```

```
    }

//-------------Ultrasoc Sensor--------------
long duration, distance;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = (duration/2) / 29.1;
Serial.println("Distance");
Serial.println(distance);

digitalWrite(buzzVcc, LOW);

gas_value=analogRead(A0);

if (gas_value > sensorThres or distance < 20){
  while(buttonOn==true){
    if (digitalRead(2)== LOW){
      buttonOn=false;
      digitalWrite(red_led, LOW);
      digitalWrite(green_led, LOW);
      digitalWrite( buzz, LOW);
      Serial.println("The alarm system is disabled!");
   }else{
   digitalWrite(red_led, HIGH);
   digitalWrite(green_led, LOW);
   digitalWrite( buzz, HIGH);
   Serial.println("DANGER!!!!");
   Serial.print("Gas Value: ");
   Serial.println(gas_value);
   Serial.print("Distance: ");
   Serial.println(distance);

   //buzzer
   digitalWrite(buzzVcc, HIGH);
   digitalWrite(buzz, LOW); // Tone ON
   delay(dashL); // Tone length
   digitalWrite(buzz, HIGH); // Tone OFF
   delay(sPause); // Symbol pause

   digitalWrite(buzz, LOW); // Tone ON
   delay(dotL); // Tone length
   digitalWrite(buzz, HIGH); // Tone OFF
   delay(sPause); // Symbol pause

   digitalWrite(buzz, LOW); // Tone ON
```

```
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    digitalWrite(buzz, LOW); // Tone ON
    delay(dotL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    delay(lPause-sPause); // Subtracts pause already taken

    digitalWrite(buzz, LOW); // Tone ON
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause
    digitalWrite(buzz, LOW); // Tone ON
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    digitalWrite(buzz, LOW); // Tone ON
    delay(dotL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause

    digitalWrite(buzz, LOW); // Tone ON
    delay(dashL); // Tone length
    digitalWrite(buzz, HIGH); // Tone OFF
    delay(sPause); // Symbol pause
    delay(wPause-sPause); // Subtracts pause already taken
    } //end else
  } //end while
 } //end if
 delay(100);
}
```

Βήμα 5 (Προστέθηκε αισθητήρας θερμοκρασίας και μια μονάδα πομπού λέιζερ στο σύστημα συναγερμού)

Συνδέστε τον αισθητήρα θερμοκρασίας και τη μονάδα πομπού λέιζερ στο σύστημα συναγερμού)



Σε αυτό το βήμα θα χρειαστούμε έναν αισθητήρα θερμοκρασίας, έναν πομπό λέιζερ, μια φωτοαντίσταση για τον δέκτη και μια αντίσταση 1 KOhm για την προστασία του.

Πως δουλεύει:

Ο πομπός λέιζερ μαζί με τον αισθητήρα φωτοαντίστασης που λειτουργεί ως δέκτης δίνει εντολή στο Arduino να ενεργοποιήσει το σύστημα όταν για κάποιο λόγο διακόπτεται η λήψη. Επίσης, το Σύστημα Συναγερμού ενεργοποιείται όταν η τιμή του αισθητήρα θερμοκρασίας υπερβεί μια συγκεκριμένη τιμή.

Arduino Code of Step 5:

```
//WORK : ALARM SYSTEM
//ADDED A TEMPERATURE SENSOR IN SYSTEM
int red_led=8;
int green_led=9;
int gas_value;
int sensorThres=400;

//----------------buzzer-------------------------
int buzz= 13; // I/O-pin from buzzer connects here
int buzzVcc= 7; //Vcc-pin from buzzer connects here
const int wpm = 20; // Morse speed in WPM
const int dotL = 1200/wpm; // Calculated dot-length
const int dashL = 3*dotL; // Dash = 3 x dot
const int sPause = dotL; // Symbol pause = 1 dot
const int lPause = dashL; // Letter pause = 3 dots
const int wPause = 7*dotL; // Word pause = 7 dots

//-------------Ultrasonic Sensor------------------
#define trigPin 3
#define echoPin 4

//----------------Button-------------------------
boolean buttonOn=false;

//-----------temperature sensor-------------------
float tempf;
int tempPin = A2;

//-----------Laser-------------------
int Laser = 5; // creating a variable named Laser which is assigned to digital pin 5
int valueLaserPh=0;// creating a variable named valueLaserPh and setting is value to zero
float valueLaserPhF=0;// creating a variable named valueLaserPhF and setting is value to
zero

// room temperature in Fa
const float baselineTemp = 100.0;

void setup()
{
  pinMode(red_led,OUTPUT);
  pinMode(buzz,OUTPUT);
  pinMode(green_led,OUTPUT);
  pinMode(A0,INPUT); //A0 Gas

  pinMode(buzz,OUTPUT); // Set buzzer-pin as output
  pinMode(buzzVcc,OUTPUT); //Vcc Buzzer

  //Ultrasonic Sensor
  pinMode(trigPin, OUTPUT);
```

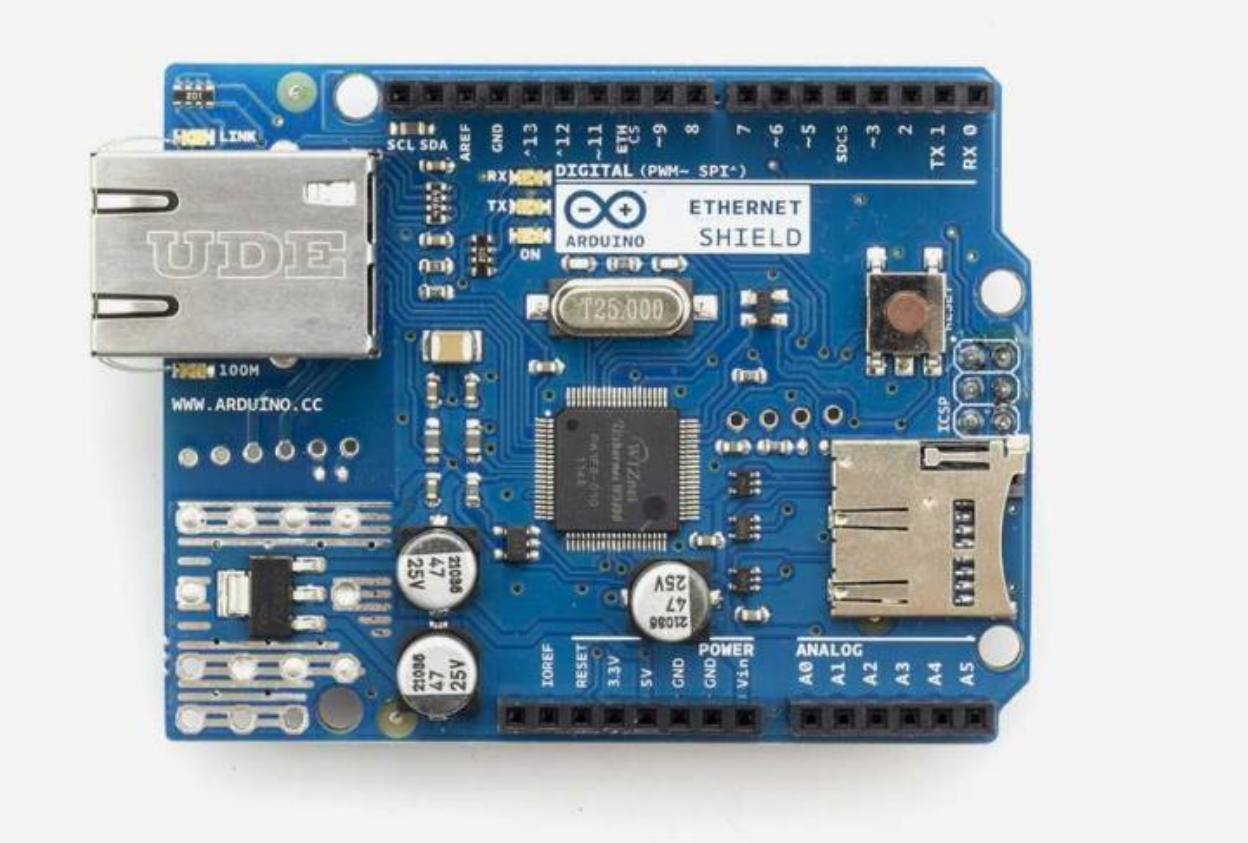
```
  pinMode(echoPin, INPUT);

//----------Laser-------------------
  pinMode (Laser,OUTPUT); // designating digital pin 5 for output
  digitalWrite(Laser,LOW); // just making sure the laser is off at startup or reset


  pinMode (2, INPUT_PULLUP);

  //temperature sensor
  pinMode(tempPin,INPUT);

  Serial.begin(9600);
}


void loop()
{
  digitalWrite(red_led, LOW);
  digitalWrite(Laser,HIGH);
  //-----------------Laser-------------------------------
  valueLaserPh=analogRead(A4); //reading the voltage on A4 and storing the value
received in "voltage"
  valueLaserPhF = valueLaserPh * (5.0 / 1023.0); // transforming the value stored in
"voltage" to readable information

  //----------temperature sensor--------------------------
  tempf = analogRead(tempPin);
  tempf=(tempf*5)/10;
  // convert the analog volt to its temperature equivalent
  Serial.print("TEMPERATURE = ");
  Serial.print(tempf); // display temperature value
  Serial.print("*F");
  Serial.println();
  delay(1000); // update sensor reading each one second

 //------------photoresistor-----------------
  int valuePhotor = analogRead(A1);
  Serial.print("Photoresistor value : ");
  Serial.println(valuePhotor);
  delay(250);

//---------------Button---------------------
 if (digitalRead(2)== LOW or valuePhotor<20){
     buttonOn=true;
     digitalWrite(red_led, LOW);
     digitalWrite(green_led, HIGH);
     digitalWrite(Laser,HIGH); // turning the laser on
     Serial.println("The alarm system is enabled!");
```

```
    }

//------------Ultrasoc Sensor -----------
long duration, distance;
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
duration = pulseIn(echoPin, HIGH);
distance = (duration/2) / 29.1;
Serial.print("distance : ");
Serial.println(distance);
digitalWrite(buzzVcc, LOW);
gas_value=analogRead(A0);
Serial.print("Gas Value : ");
Serial.println(gas_value);
Serial.print("Laser value : ");
Serial.println(valueLaserPhF);

if (gas_value > sensorThres or distance < 10 or tempf>baselineTemp or valueLaserPhF<1)
 {
   while(buttonOn==true){
    if (digitalRead(2)== LOW){
       buttonOn=false;
       //digitalWrite(red_led, LOW);
       digitalWrite(green_led, LOW);
       digitalWrite( buzz, LOW);
       Serial.println("The alarm system is disabled!");
       distance=100;
       tempf=0.0;
   }else{
        digitalWrite(red_led, HIGH);
        digitalWrite(green_led, LOW);
        digitalWrite( buzz, HIGH);
        Serial.println("DANGER!!!!");
        Serial.print("Gas Value: ");
        Serial.println(gas_value);
        Serial.print("Distance: ");
        Serial.println(distance);
        Serial.print("Temperature: ");
        Serial.println(tempf);
        Serial.print("Photoresistor value : ");
        Serial.println(valuePhotor);
        Serial.print("Laser value : ");
        Serial.println(valueLaserPhF);

        //--------------buzzer--------------
        digitalWrite(buzzVcc, HIGH);
```

```
      digitalWrite(buzz, LOW); // Tone ON
      delay(dashL); // Tone length
      digitalWrite(buzz, HIGH); // Tone OFF
      delay(sPause); // Symbol pause

      digitalWrite(buzz, LOW); // Tone ON
      delay(dotL); // Tone length
      digitalWrite(buzz, HIGH); // Tone OFF
      delay(sPause); // Symbol pause

      digitalWrite(buzz, LOW); // Tone ON
      delay(dashL); // Tone length
      digitalWrite(buzz, HIGH); // Tone OFF
      delay(sPause); // Symbol pause

      digitalWrite(buzz, LOW); // Tone ON
      delay(dotL); // Tone length
      digitalWrite(buzz, HIGH); // Tone OFF
      delay(sPause); // Symbol pause

      delay(lPause-sPause); // Subtracts pause already taken

      digitalWrite(buzz, LOW); // Tone ON
      delay(dashL); // Tone length
      digitalWrite(buzz, HIGH); // Tone OFF
      delay(sPause); // Symbol pause
      digitalWrite(buzz, LOW); // Tone ON
      delay(dashL); // Tone length
      digitalWrite(buzz, HIGH); // Tone OFF
      delay(sPause); // Symbol pause

      digitalWrite(buzz, LOW); // Tone ON
      delay(dotL); // Tone length
      digitalWrite(buzz, HIGH); // Tone OFF
      delay(sPause); // Symbol pause

      digitalWrite(buzz, LOW); // Tone ON
      delay(dashL); // Tone length
      digitalWrite(buzz, HIGH); // Tone OFF
      delay(sPause); // Symbol pause
      delay(wPause-sPause); // Subtracts pause already taken
    }//end else
  }//end while
 } //end if
delay(100);
}
```

ΕΡΓΟ 3- ΟΝΟΜΑ: WebServer

Ο στόχος του έργου: Σε αυτό το έργο, οι μαθητές θα δουν πώς συνδέουμε ένα arduino σε ένα τοπικό δίκτυο χρησιμοποιώντας μια ασπίδα ethernet. Θα πειραματιστούν με αισθητήρες και θα



μπορούν να δουν τις τιμές που έχουν λάβει στην οθόνη του υπολογιστή τους μέσω του τοπικού δικτύου.

Μέσα από αυτό το έργο, οι μαθητές θα μπορούσαν να πειραματιστούν με:

- ✓ a gas sensor
- ✓ a photoresistor sensor
- ✓ a temperature sensor

Η ασπίδα Ethernet επιτρέπει σε μια πλακέτα Arduino να συνδεθεί στο διαδίκτυο.

Λίστα εξαρτημάτων:

Για την υλοποίηση του έργου θα χρειαστείτε επιπλέον τα ακόλουθα στοιχεία:

- Ethernet Shield
- Ethernet cable
- Breadboard and Jump Wires
- Wire USB
- Arduino UNO R3
- a photoresistor sensor
- a gas sensor
- a temperature sensor
- a resistor 10KΩ

Πως δουλεύει:

Το Arduino Ethernet Shield 2 επιτρέπει σε μια πλακέτα Arduino να συνδεθεί στο διαδίκτυο. Βασίζεται στο (τσιπ Wiznet W5500 Ethernet). Το Wiznet W5500 παρέχει μια στοίβα δικτύου (IP) ικανή για TCP και UDP. Υποστηρίζει έως και οκτώ ταυτόχρονες συνδέσεις πρίζας.

Σε αυτό το παράδειγμα οι μαθητές θα αναζητήσουν την IP της ασπίδας ethernet σε ένα πρόγραμμα περιήγησης τοπικού δικτύου και θα μπορούν να δουν τις τιμές των αισθητήρων που συνέδεσαν στην ασπίδα ethernet σε μια φιλική σελίδα. Η σελίδα ανανεώνεται αυτόματα κάθε 5 δευτερόλεπτα.

Κύκλωμα Έργου.

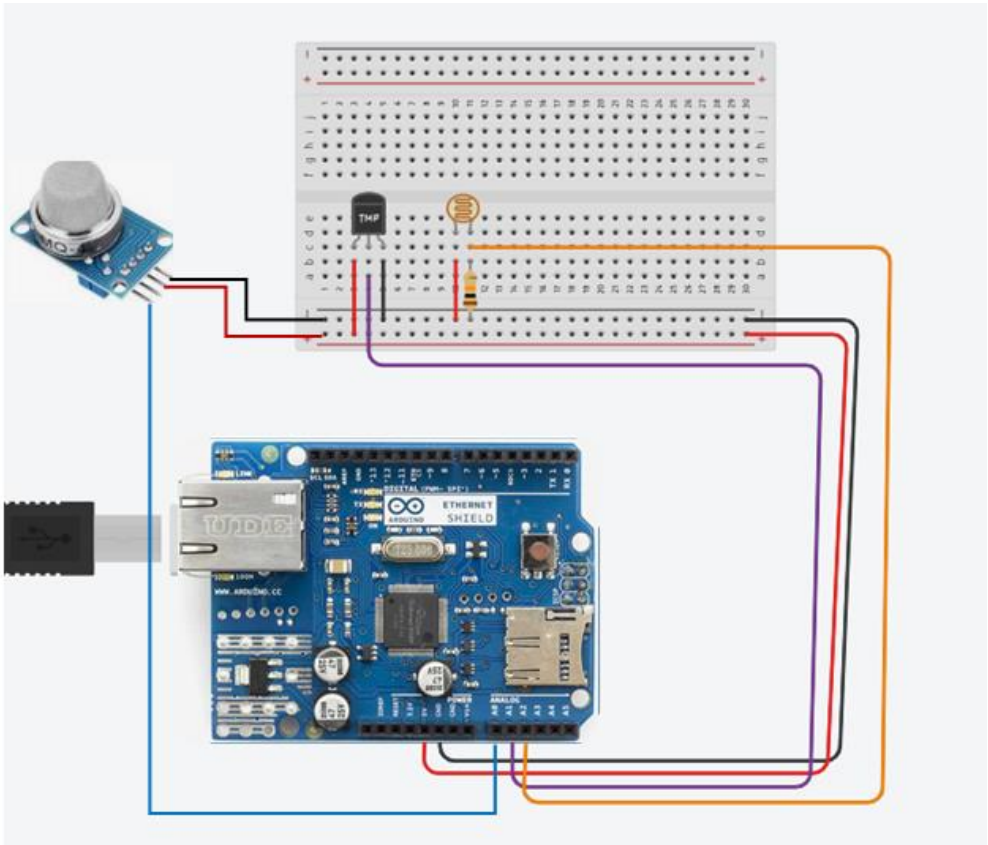
Μεθοδολογία

Αρχικά, περιγράφουμε την επιθυμητή ροή εργασίας του έργου που θα αναπτυχθεί σε βήματα. Στη συνέχεια ζητείται από τους μαθητές να επιλέξουν τα απαραίτητα στοιχεία για κατασκευάσουν το κύκλωμα. Θα χρησιμοποιήσουν την εφαρμογή "Arduino IDE" για να γράψουν τον κώδικα.

Στη συνέχεια θα δημιουργήσουν το κύκλωμα, θα γράψουν τον κώδικα στο εργαλείο λογισμικού Arduino και θα τον ανεβάσουν στην πλακέτα του Arduino.

Οι μαθητές θα επαληθεύσουν ότι το κύκλωμα κατασκευάστηκε σωστά.

Αυτό το έργο απευθύνεται σε μαθητές βασικού επιπέδου, οι οποίοι αρχίζουν να επωφελούνται από τα αποτελέσματα και τα αποτελέσματα από το έργο Στρατηγικών Συνεργασιών Erasmus+ KA-202 στην ΕΕΚ, που



ονομάζεται «ArduInVet».

Arduino Code of Project:

```
/*
  Web Server

 A simple web server that shows the value of the analog input pins.
 using an Arduino Wiznet Ethernet shield.

 Circuit:
 * Ethernet shield attached to pins 10, 11, 12, 13
 * Analog inputs attached to pins A0 through A5 (optional)

 */

#include <SPI.h>
#include <Ethernet.h>
unsigned long refreshCounter  = 0;
// Enter a MAC address and IP address for your controller below.
// The IP address will be dependent on your local network:
byte mac[] = {
  0xA8, 0x61, 0x0A, 0xAE, 0x63, 0xA8
};
IPAddress ip(169, 254, 122, 108);

// Initialize the Ethernet server library
// with the IP address and port you want to use
// (port 80 is default for HTTP):
EthernetServer server(80);

EthernetClient client;
void setup() {
  // You can use Ethernet.init(pin) to configure the CS pin
  //Ethernet.init(10);  // Most Arduino shields
  //Ethernet.init(5);   // MKR ETH shield
  //Ethernet.init(0);   // Teensy 2.0
  //Ethernet.init(20);  // Teensy++ 2.0
  //Ethernet.init(15);  // ESP8266 with Adafruit Featherwing Ethernet
  //Ethernet.init(33);  // ESP32 with Adafruit Featherwing Ethernet

  // Open serial communications and wait for port to open:
  Serial.begin(9600);
  while (!Serial) {
        ; // wait for serial port to connect. Needed for native USB port only
  }
  Serial.println("ARDUinVET in Ethernet WebServer ");
```

```
 // start the Ethernet connection and the server:
 Ethernet.begin(mac, ip);

 // Check for Ethernet hardware present
 if (Ethernet.hardwareStatus() == EthernetNoHardware) {
  Serial.println("Ethernet shield was not found.  Sorry, can't run without hardware. :(");
        while (true) {
                delay(1); // do nothing, no point running without Ethernet hardware
        }
 }
 if (Ethernet.linkStatus() == LinkOFF) {
        Serial.println("Ethernet cable is not connected.");
 }
 // start the server
 server.begin();
 Serial.print("server is at ");
 Serial.println(Ethernet.localIP());
}

void loop() {
    // listen for incoming clients
   client = server.available();
   if (client) {
        Serial.println("ARDUinVET in client");
       // an http request ends with a blank line
       boolean currentLineIsBlank = true;
       while (client.connected()) {
           if (client.available()) {
              char c = client.read();
             Serial.write(c);
             // if you've gotten to the end of the line (received a newline
            // character) and the line is blank, the http request has ended,
            // so you can send a reply
               if (c == '\n' && currentLineIsBlank) {
                    // send a standard http response header
                    client.println("HTTP/1.1 200 OK");
                    client.println("ARDinVET - HTTP");
                    client.println("Content-Type: text/html");
                    client.println("Connection: close");  // the connection will be closed after
//completion of the response
                    client.println("Refresh: 5");  // refresh the page automatically every 5 sec
                   client.println();
                   webpage(client);
                   break;
                   }
                   if (c == '\n') {
                          // you're starting a new line
                          currentLineIsBlank = true;
                          } else if (c != '\r') {
```

```
                    // you've gotten a character on the current line
                    currentLineIsBlank = false;
                }
            }
        }
        // give the web browser time to receive the data
        delay(1);
        // close the connection:
        client.stop();
        Serial.println("client disconnected");
    }
}

void webpage(EthernetClient client) { /* function webpage */
////Send wbepage to client
client.println("ARDUinVET - HTTP");
//client.println("Content-Type: text/html");
client.println(""); //  do not forget this one
client.println("<!DOCTYPE HTML>");
client.println("<html>");
client.println("<head>");
client.println("<title>ARDUinVET</title>");
//client.println("<script src='https://smtpjs.com/v3/smtp.js'></script>");
//client.println("<script>");
//client.println("function sendEmail() {");
/*client.println("Email.send({");
client.println("  Host: 'smtp.gmail.com',");
client.println("  Username : '**********@gmail.com',");
client.println("  Password : '******************',");
client.println("  To : '**********@gmail.com',");
client.println("  From : '**********@gmail.com',");
client.println("  Subject : 'Data from Arduino',");
client.println("  Body : 'Sensors value....',");
client.println("  }).then(");
client.println("    message => alert('mail sent successfully')");
client.println("  );");*/
//client.println("}");
//client.println("</script>");
client.println("</head>");
client.println("<body bgcolor = '#cccc00'>");
client.println("<hr/><hr>");
client.println("<h1 style='color : #0000cc;'><center>  Alarm System </center></h1>");
client.println("<hr/><hr>");
client.println("<br><br>");
client.println("<br><br>");
client.println("<center>");
client.println("<br>Sensors Output<br>");
client.println("Gas.....:");
client.println("  <input value=" + String(analogRead(A0)) + " readonly></input>");
```

```
client.println("<br>");
client.println("  Photoresistor..:");
client.println("  <input value=" + String(analogRead(A1)) + " readonly></input>");
client.println("<br>");
client.println("  Temperature....:");
client.println("  <input value=" + String(analogRead(A2)) + " readonly></input>");
client.println("<br>");
client.println("  </center>");
client.println("<br><br>");
client.println("<center>");
client.println("<a style='color : #0000cc;'
href='https://www.arduinvet.com/'>www.arduinvet.com</a>");
client.println("</center>");
client.println("<br><br>");
client.println("</body></html>");
client.println();
delay(1);
}
```


ARDUinVET
Μη ασφαλής | 169.254.122.108
Gmail
YouTube
Χάρτες
ARDUinVET - HTTP
Sensors-Arduino Ethernet Shield
Sensors Output
Gas.....: 142
Photoresistor..: 23
Temperature....: 169
www.arduinvet.com

Co-funded by the Erasmus+ Programme of the European Union

Erasmus+ KA-202

Πρόγραμμα Στρατηγικής Σύμπραξης στην Επαγγελματική Εκπαίδευση και Κατάρτιση

Τίτλος Προγράμματος: "Teaching and Learning Arduinos in Vocational Training"

Ακρονύμιο Προγράμματος: " ARDUinVET "

Κωδικός Προγράμματος: "2020-1-TR01-KA202-093762"

Erasmus+ KA-202

Ελεύθερα Projects Arduino

(Αυτά τα projects δημιουργήθηκαν από τους μαθητές των σχολείων - εταίρων στη σύμπραξη ARDUinVET.)

Α/Α	Τίτλος Project	Χώρα
1	ΚΑΠΕΛΟ ΚΟΙΝΩΝΙΚΗΣ ΑΠΟΣΤΑΣΗΣ	ΤΟΥΡΚΙΑ
2	ΘΕΡΜΟΜΕΤΡΟ	ΕΛΛΑΣ
3	ΦΩΣ ΦΡΕΝΟΥ ΕΚΤΑΚΤΗΣ ΑΝΑΓΚΗΣ - ΠΡΟΣΑΡΜΟΖΟΜΕΝΟ ΦΩΣ ΦΡΕΝΟΥ	ΡΟΥΜΑΝΙΑ
4	LINEFOLLOWER	ΑΥΣΤΡΙΑ
5	ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΟ ΘΕΡΜΟΚΗΠΙΟ	ΙΤΑΛΙΑ

1ο PROJECT: ΚΑΠΕΛΟ ΚΟΙΝΩΝΙΚΗΣ ΑΠΟΣΤΑΣΗΣ (ΤΟΥΡΚΙΑ)

Στόχος του Project: Στις μέρες μας, που η πανδημία Covid-19 είναι παρούσα, η τήρηση του κανόνα της «ΚΟΙΝΩΝΙΚΗΣ ΑΠΟΣΤΑΣΗΣ» είναι μια από τις βασικές προϋποθέσεις για να μην προσβληθεί κανείς από τη νόσο.

Προκειμένου να επιστήσουμε την προσοχή στο θέμα της «κοινωνικής αποστασιοποίησης», αποφασίσαμε να σχεδιάσουμε ένα σχετικό έργο Arduino.

Ο κανόνας υπενθυμίζεται με το «Καπέλο κοινωνικής απόστασης», το οποίο θα ετοιμάσουμε. Θα δώσει έναν ηχητικό ήχο βομβητή και ταυτόχρονα μια φωτεινή προειδοποίηση.

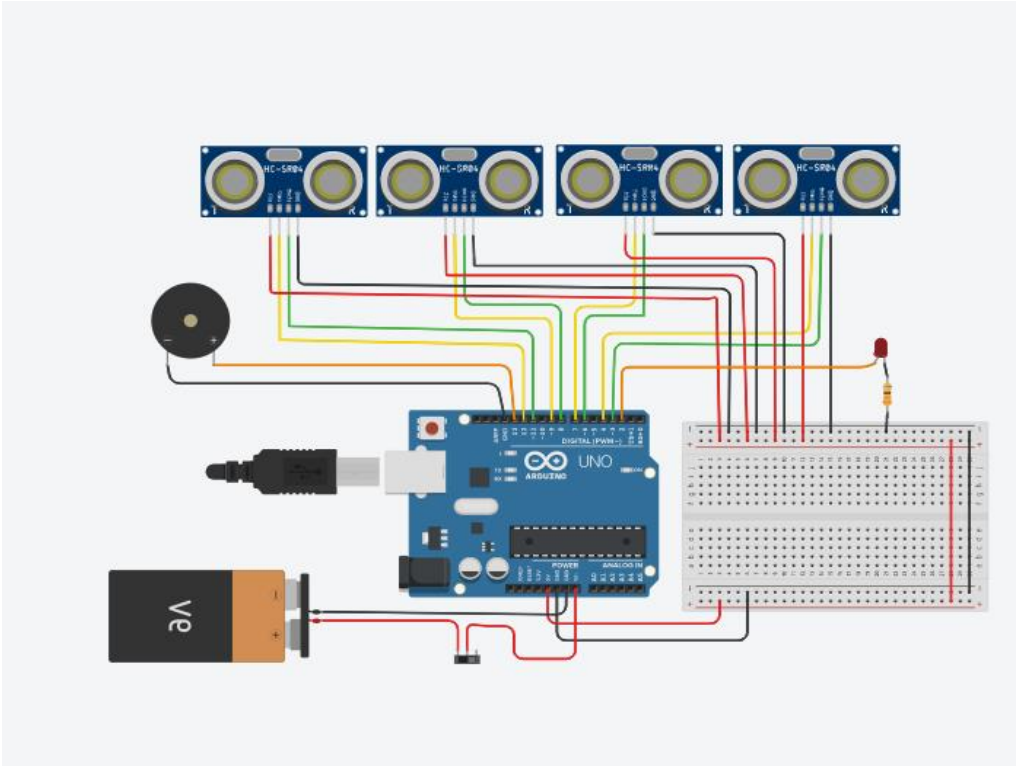
Μεθοδολογία του Project:

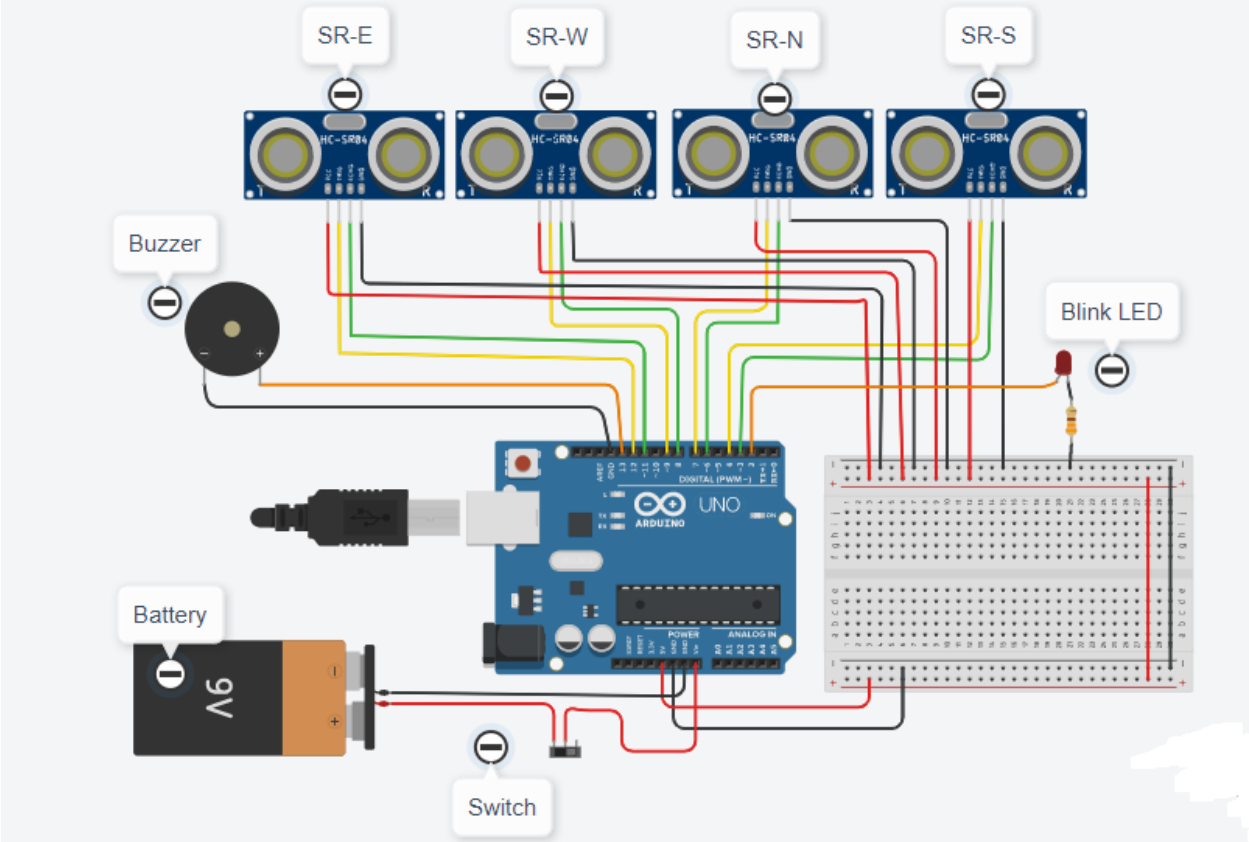
Αρχικά, προετοιμάστηκε ο αλγόριθμος του έργου. Σύμφωνα τον αλγόριθμο, έγινε το λογικό διάγραμμα και γράφηκε ο κώδικας Arduino. Το Arduino χρησιμοποιήθηκε στο σχεδιασμό του κυκλώματος. Κατά τη δημιουργία αυτών των σχεδίων, οι μαθητές επωφελήθηκαν από όλα τα αποτελέσματα και τα προϊόντα του έργου Erasmus+ KA-202 VET, «ArduinVET».

Το τελικό κύκλωμα μας ετοιμάζεται με τον απαραίτητο εξοπλισμό και προγραμματισμό του Arduino. Το τελικό κύκλωμα μας τοποθετείται οπτικά στο καπέλο.

Αυτό το έργο εκτέθηκε και σε εκθέσεις και υπενθυμίζει στους επισκέπτες τον κανόνα της «Κοινωνικής Αποστασιοποίησης» και τη σημασία του..

Κύκλωμα του Project.





Λειτουργία:

4 αισθητήρες απόστασης υπερήχων τοποθετούνται στις 4 κατευθύνσεις του καπέλου. Όταν οι αισθητήρες εντοπίσουν κάποιον που πλησιάζει κάτω από τα 100 cm, θα ηχήσει ο βομβητής και το led θα αναβοσβήσει. Το κύκλωμα τροφοδοτείται από μπαταρία 9V.

Κατάλογος υλικών:

Ένα καπέλο, ένα Arduino Uno R3, 4 αισθητήρες απόστασης υπερήχων, ένας βομβητής, μια αντίσταση 330 Ω, μια κόκκινη λυχνία LED, μια μπαταρία 9V, ένας διακόπτης ολίσθησης.

Κώδικας Arduino του Project:

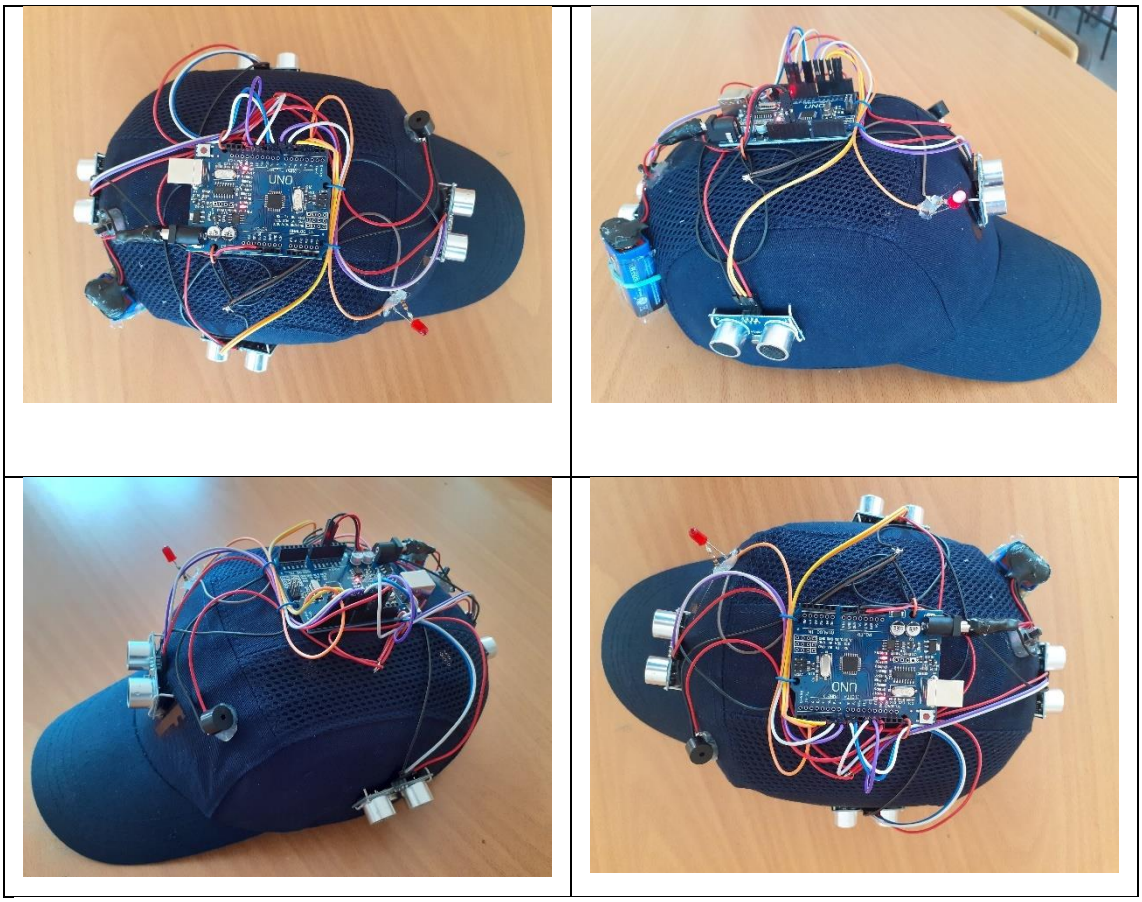
//Project Name: SOCIAL DISTANCING HAT:

```
int trigPin=12;
int echoPin=11;
int trigPin2=9;
int echoPin2=8;
int trigPin3=7;
int echoPin3=6;
int trigPin4=4;
int echoPin4=3;
void setup()
{
pinMode(trigPin, OUTPUT);
pinMode(echoPin, INPUT);
 pinMode(trigPin2, OUTPUT);
pinMode(echoPin2, INPUT);
pinMode(trigPin3, OUTPUT);
pinMode(echoPin3, INPUT);
 pinMode(trigPin4, OUTPUT);
pinMode(echoPin4, INPUT);
pinMode(13,OUTPUT);
pinMode(2,OUTPUT);
 Serial.begin(9600);
}
void loop() {
 digitalWrite(trigPin, LOW);
```

```
delay(3);
digitalWrite(trigPin, HIGH);
delay(3);
digitalWrite(trigPin, LOW);
int time1 = pulseIn(echoPin, HIGH);
int distance1 = (time1/2) / 28.97;         //Social distance is equal to and less than 100cm.
//-----------------------------------------------------
 digitalWrite(trigPin2, LOW);
delay(3);
digitalWrite(trigPin2, HIGH);
delay(3);
digitalWrite(trigPin2, LOW);

int time2 = pulseIn(echoPin2, HIGH);
int distance2 = (time2/2)  / 28.97;    //Social distance is equal to and less than 100cm.
//-----------------------------------------------------
 digitalWrite(trigPin3, LOW);
delay(3);
digitalWrite(trigPin3, HIGH);
delay(3);
digitalWrite(trigPin3, LOW);
int time3 = pulseIn(echoPin3, HIGH);
int distance3 = (time3/2)  / 28.97;       //Social distance is equal to and less than 100cm.
//-----------------------------------------------------
digitalWrite(trigPin4, LOW);
delay(3);
```

```
digitalWrite(trigPin4, HIGH);
delay(3);
digitalWrite(trigPin4, LOW);
int time4 = pulseIn(echoPin4, HIGH);
int distance4 = (time4/2)  / 28.97;         //Social distance is equal to and less than 100cm.
//---------------------------------------------------------
if(distance1<75)
 {
  digitalWrite(13,HIGH);
  digitalWrite(2,HIGH);
  }
 else if(distance2<75)
 {
  digitalWrite(13,HIGH);
   digitalWrite(2,HIGH);
  }
 else if(distance3<75)
 {
  digitalWrite(13,HIGH);
   digitalWrite(2,HIGH);
  }
 else if(distance4<75)
 {
  digitalWrite(13,HIGH);
   digitalWrite(2,HIGH);
  }
```

```
else
{
digitalWrite(13,LOW);
digitalWrite(2,LOW);
}
 Serial.println(distance1);      // To see which ultrosonc sensor is activated on the Serial monitor
 Serial.println(distance2);
 Serial.println(distance3);
 Serial.println(distance4);
 delay(500);
}
```



Εικόνα: τελική μορφή του κυκλώματος σε καπέλο

2^{o} PROJECT: ΘΕΡΜΟΜΕΤΡΟ (ΕΛΛΑΣ)

Στόχος του Project: Αυτό το έργο είναι μια επέκταση του απλού βασικού project Love Meter που βρίσκεται στο γνήσιο εγχειρίδιο Arduino. Σε αυτήν την εκτεταμένη έκδοση, οι μαθητές θα χειριστούν έναν αισθητήρα (θερμοκρασίας), θα πειραματιστούν με το ποτενσιόμετρο, θα ανάψουν ένα RGB LED και θα γνωρίσουν την οθόνη LCD (αν είναι διαθέσιμη, διαφορετικά η οθόνη θα είναι απλώς η σειριακή οθόνη του λογισμικού Arduino).

Είναι ένα project που επιτρέπει στους μαθητές να πειραματιστούν με πολλά στοιχεία διαφορετικών τύπων και να εξοικειωθούν με τις τιμές εισόδου και τις μεθόδους εμφάνισης.

Μεθοδολογία του Project:

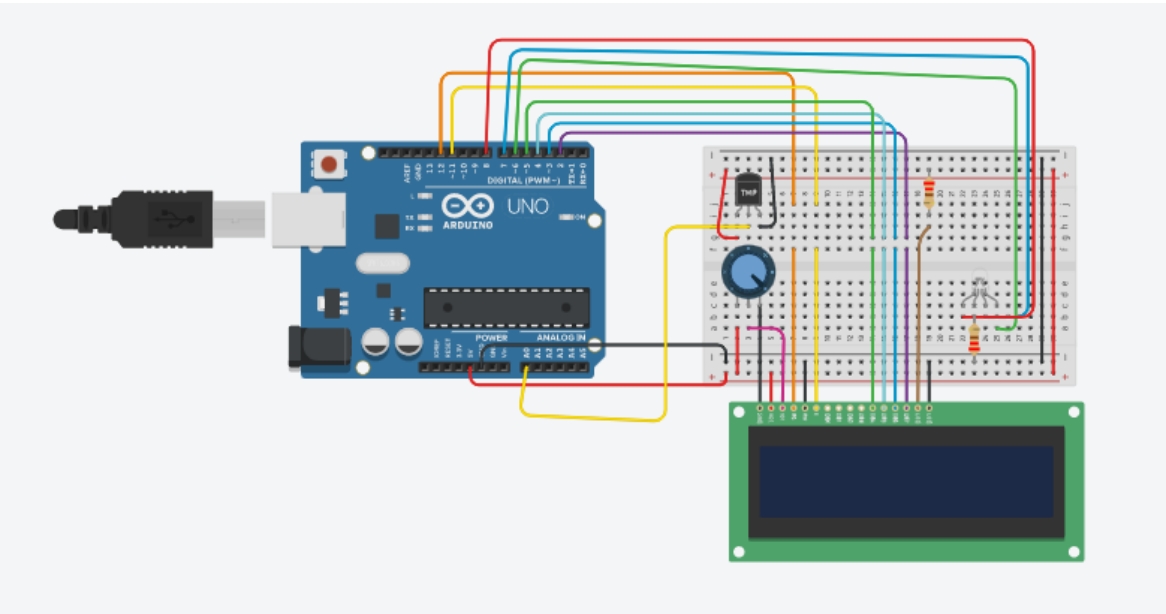
Αρχικά, περιγράφουμε την επιθυμητή ροή εργασιών του έργου. Στη συνέχεια ζητάμε από τους μαθητές να επιλέξουν τα απαραίτητα υλικά και να χρησιμοποιήσουν το Tinkercad για να δημιουργήσουν και να σχεδιάσουν το κύκλωμα. Θα χρησιμοποιήσουν το Εργαλείο Κώδικα Tinkercad για να γράψουν τον κώδικα.

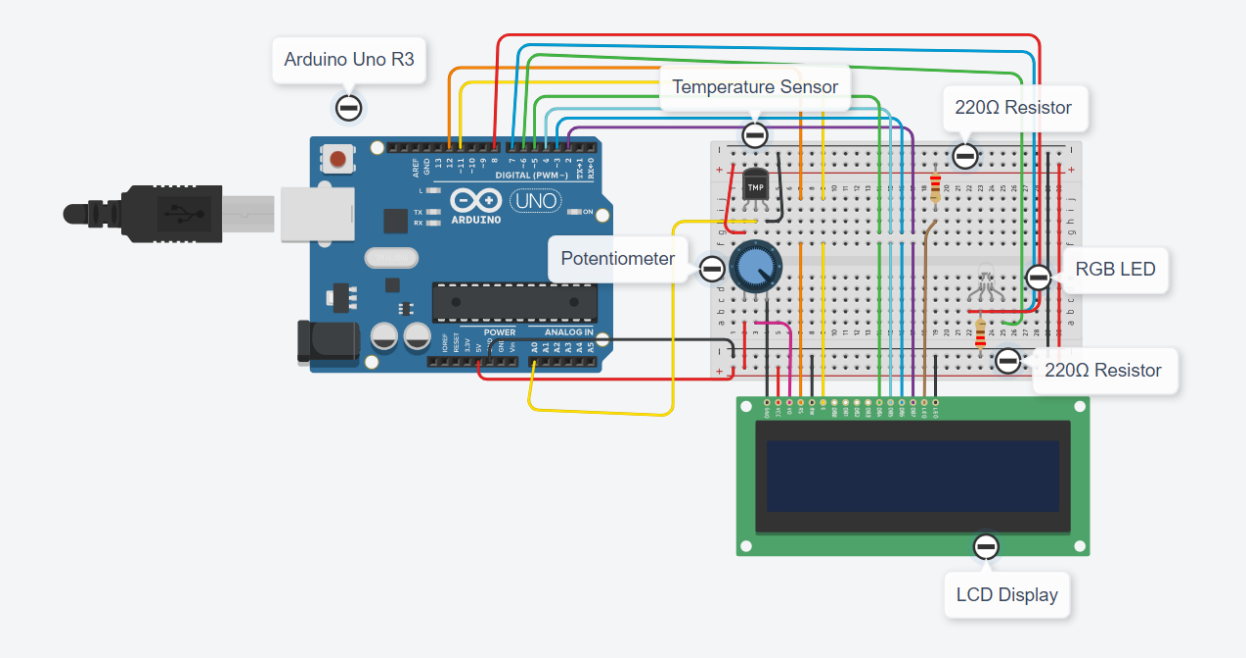
Η προσομοίωση στο Tinkercad θα επαληθεύσει ότι το κύκλωμα κατασκευάστηκε σωστά

Στη συνέχεια θα δημιουργήσουν το κύκλωμα, θα γράψουν τον κώδικα στο εργαλείο λογισμικού Arduino και θα τον ανεβάσουν στην πλακέτα του Arduino.

.Αυτό το έργο απευθύνεται σε μαθητές βασικού επιπέδου, οι οποίοι αρχίζουν να επωφελούνται από τα αποτελέσματα και τα προϊόντα του έργου μας Erasmus+ KA-202 Στρατηγικές Συμπράξεις στην ΕΕ, που ονομάζεται «ArduinVET».

Κύκλωμα του Project.





Λειτουργία:

Ο αισθητήρας θερμοκρασίας εισάγει τις τιμές της θερμοκρασίας δωματίου. Φυσικά, οι μαθητές μπορούν να αλλάξουν τη θερμοκρασία χρησιμοποιώντας τα δάχτυλά τους. Η θερμοκρασία θα εμφανίζεται στην οθόνη LCD (εάν υπάρχει) και στη σειριακή οθόνη. Το LED RGB θα έχει διαφορετικά χρώματα (απενεργοποιημένο, πράσινο, μπλε, κόκκινο) ανάλογα με την τιμή θερμοκρασίας. Το ποτενσιόμετρο θα χρησιμοποιηθεί ως διακόπτης για την οθόνη LCD. Το κύκλωμα θα τροφοδοτείται από το καλώδιο USB (εναλλακτικά μπορείτε να χρησιμοποιήσετε ένα κουτί μπαταρίας).

Κατάλογος υλικών:

Τα εξαρτήματά μας είναι: ένα Arduino Uno R3, ένας αισθητήρας θερμοκρασίας, δύο (2) αντιστάσεις 220Ω, ένα RGB LED και (προαιρετικά) ένα ποτενσιόμετρο και μια οθόνη LCD (16*2).

Κώδικας Arduino του Project:

//Project Name: TEMPERATURE METER

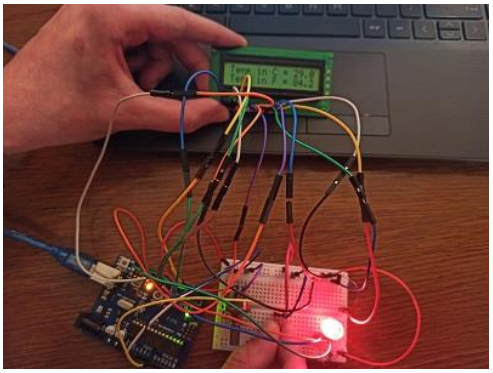
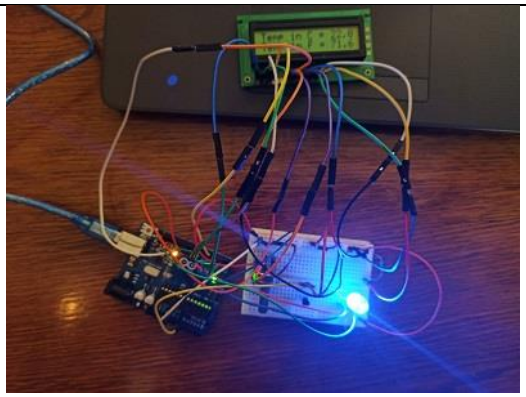
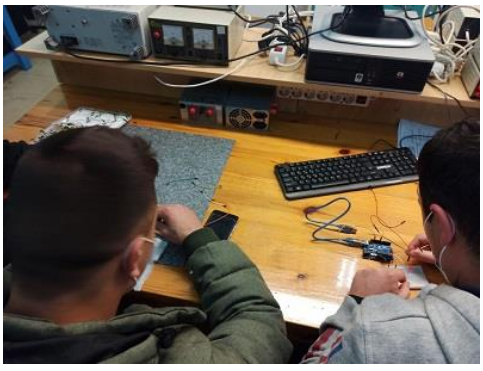
// include the library code:

```
#include <LiquidCrystal.h>
int t=0;
int sensor = A0;
float temp;
float tempc;
float tempf;
// initialize the library with the numbers of the interface pins
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);
//**********RGB**********
const int sensorPin = A0;
// room temperature in Celsius
const float baselineTemp = 20.0;
//**************************
void setup() {
  // set up the LCD's number of columns and rows:
pinMode(sensor,INPUT);
lcd.setCursor (0,0);
lcd.begin(16, 2);
  // Print a message to the LCD.
lcd.print ("    ARDUINO     ");
lcd.setCursor (0,1);
lcd.print ("TEMPERATURE METER");
```

```
delay (3000);
Serial.begin(9600);
for(int pinNumber = 6; pinNumber<8; pinNumber++){
    pinMode(pinNumber,OUTPUT); // pin6:Blue pin7:Green  pin8:Red
    digitalWrite(pinNumber, LOW);
   }
}
void loop() {
delay(2000);
t=t+2;
temp=analogRead(sensor);
//tempc=(temp*5)/10;
tempc=map(((temp-20)*3.04), 0, 1023, -40, 125);
tempf=(tempc*1.8)+32;
Serial.println("_______");
Serial.println("Temperature Logger");
Serial.print("Time in Seconds= ");
Serial.println(t);
Serial.print("Temp in deg Celcius = ");
Serial.println(tempc);
Serial.print("Temp in deg Fahrenheit = ");
Serial.println(tempf);
lcd.setCursor(0,0);
lcd.print("Temp in C = ");
lcd.println(tempc);
lcd.setCursor(0,1);
```

```
lcd.print("Temp in F = ");
lcd.println(tempf);


//*************code RGB***************
  // read the value on AnalogIn pin 0
  // and store it in a variable
  int sensorVal = analogRead(sensorPin);
  // send the 10-bit sensor value out the serial port
  Serial.println("sensor Value: ");
  Serial.println(sensorVal);

  // convert the ADC reading to voltage
  float voltage = (sensorVal/1024.0) * 5.0;
  // Send the voltage level out the Serial port
  Serial.println(", Volts: ");
  Serial.println(voltage);
   if(tempc < 20){
   digitalWrite(6, LOW);
   digitalWrite(7, LOW);
   digitalWrite(8, LOW);
   }
   else if(tempc >= 20 && tempc < 80){
         digitalWrite(6, HIGH);
   digitalWrite(7, LOW);
   digitalWrite(8, LOW);
  }
```

```
 else if(tempc >= 80 && tempc < 140){
        digitalWrite(6, LOW);
  digitalWrite(7, HIGH);
  digitalWrite(8, LOW);
}
 else if(tempc >= 140){
        digitalWrite(6, LOW);
  digitalWrite(7, LOW);
  digitalWrite(8, HIGH);
}
delay(100);
}
```

Εικόνα: Εκπαιδευτικό κιτ του Project και μαθητές που εργάζονται στο project.

3ο PROJECT: ΦΩΣ ΦΡΕΝΟΥ ΕΚΤΑΚΤΗΣ ΑΝΑΓΚΗΣ - ΠΡΟΣΑΡΜΟΖΟΜΕΝΟ ΦΩΣ ΦΡΕΝΟΥ (ΡΟΥΜΑΝΙΑ)

Στόχος του Project: Σε περίπτωση ισχυρού φρεναρίσματος, το προσαρμοζόμενο φως φρένων μπορεί να προειδοποιήσει την επερχόμενη κυκλοφορία και έτσι να βοηθήσει στην αποφυγή συγκρούσεων από το πίσω μέρος. Ενεργοποιούνται εάν κατά το φρενάρισμα, και ανάλογα με την πίεση και την ταχύτητα πέδησης, εντοπιστεί μια κρίσιμη κατάσταση πέδησης

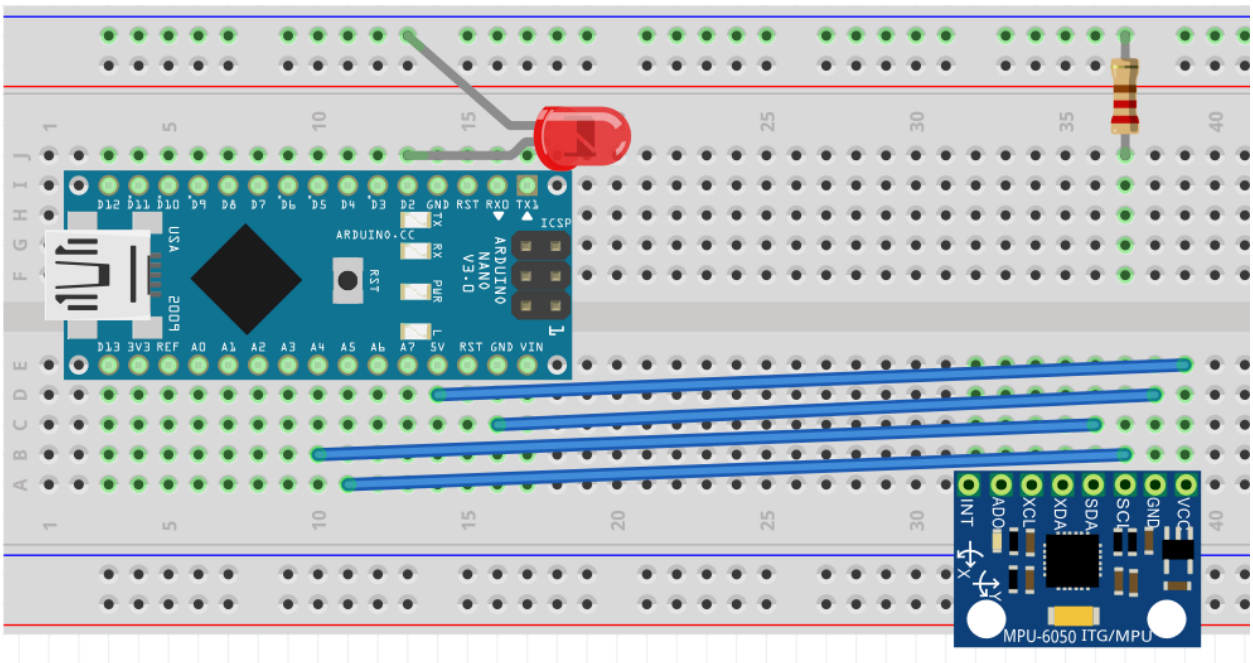
Μεθοδολογία του Project:

Αρχικά, ετοιμάζεται ο αλγόριθμος του έργου. Σύμφωνα με τον αλγόριθμο, γίνεται ο λογικός σχεδιασμός και γράφεται ο κώδικας Arduino. Το Arduino χρησιμοποιείται στο σχεδιασμό του κυκλώματος. Κατά τη δημιουργία του σχεδίου, οι μαθητές μας επωφελούνται από όλα τα αποτελέσματα και τα προϊόντα του έργου Erasmus+ KA-202 VET «ArduinVET».

Το τελικό κύκλωμα μας ρυθμίζεται αφού παρέχουμε τα απαραίτητα υλικά και προγραμματίσουμε το Arduino. Το τελικό μας κύκλωμα τοποθετείται σε αυτοκίνητο.

Αυτό το έργο εκτίθεται σε εκθέσεις και υπενθυμίζει στους επισκέπτες πόσο σημαντική είναι η κυκλοφοριακή ασφάλεια..

Κύκλωμα του Project.



Λειτουργία:

Τα φώτα φρένων έκτακτης ανάγκης ενεργοποιούνται **για να ειδοποιούν τα έποντα οχήματα για ισχυρό φρενάρισμα**. Η λειτουργία συνίσταται ότι το φως των φρένων αναβοσβήνει αντί -όπως στο κανονικό φρενάρισμα- να λάμπει συνεχόμενα. Τα φώτα φρένων έκτακτης ανάγκης ενεργοποιούνται σε ταχύτητες άνω των 50 km/h σε περίπτωση ισχυρού φρεναρίσματος.

Βασίζεται σε ένα ηλεκτρονικό επιταχυνσιόμετρο (MPU6050) υποβοηθούμενο από μικροελεγκτή και ανιχνεύει το ξαφνικό φρενάρισμα με βάση τη δύναμη της αδράνειας και αναβοσβήνει γρήγορα το 3ο στοπ του φρένου, κάνοντας το φρενάρισμα πολύ πιο εύκολο να εντοπιστεί.

Το MPU6050 διαθέτει γυροσκόπιο 3 αξόνων, επιταχυνσιόμετρο 3 αξόνων και ψηφιακό επεξεργαστή κίνησης ενσωματωμένο σε ένα μόνο τσιπ. Λειτουργεί με τροφοδοτικό 3V-5V. Το MPU6050 χρησιμοποιεί το **πρωτόκολλο I2C για επικοινωνία και μεταφορά δεδομένων**. Αυτή η μονάδα διαθέτει ενσωματωμένο ADC 16-bit που παρέχει μεγάλη ακρίβεια.

Το φίλτρο Kalman είναι ένα αποτελεσματικό αναδραστικό φίλτρο που αξιολογεί την κατάσταση ενός δυναμικού συστήματος με βάση μια σειρά μετρήσεων ευαίσθητων στο θόρυβο. Λόγω των εγγενών χαρακτηριστικών του, είναι ένα εξαιρετικό φίλτρο για θόρυβο και διαταραχές που δρουν σε συστήματα Gaussian μηδενικού μέσου όρου.

Κατάλογος υλικών:

Ενα Arduino Nano, ένα MPU6050, μια αντίσταση 180 Ohm, ένα κόκκινο LED.

Κώδικας Arduino του Project:

//Project Name: EMERGENCY BRAKE BLINKER - ADAPTIVE BRAKE LIGHT

```
#include<Wire.h>;
//https://www.codetd.com/en/article/11749279 this is an articol who explain very well how to comunicate more fast with MPU6050
//in automotive conditions is many noise from the road conditions, vibrations etc. and I'll use a very good software filter (Kalman)
const int MPU_addr = 0x68;
float kalman_old = 0;
float cov_old = 1;
float val1, val2, val3, val4, val5;
int med1_sort, med2_sort, med3_sort, med4_sort;
```

```
float avg;
int med;
int m;
int med_sort[5];
int c_avg = 1;
int c_med = 1;
int d = 0;
float old_x = 0;
float real_angle = 0;
float prev_angle = 0;
unsigned long previousMillis = 0;
const long interval = 50;
void setup() {
  Serial.begin(9600); // for USB monitor
  Serial.println ("Hit a key to start");     // signal initalization done
  while (Serial.available() == 1);
  //Connect to G-Sensor
  Wire.begin();
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x6B); // PWR_MGMT_1 register
  Wire.write(0x00); // set to zero (wakes up the MPU-6050)
  Wire.endTransmission(true);
  delay(50);
  Wire.beginTransmission(MPU_addr);
  Wire.write(0x1C);
  Wire.write(0x00);
  Wire.endTransmission(true);
  pinMode(2, OUTPUT); //output signal for FIRST and SECOND stage.
digitalWrite (2, LOW);
//pinMode(7, OUTPUT);//output signal for SECOND stage. Enable this and next line if you want more expressive functionality.
//digitalWrite (7, LOW); If enable this option, you need enable digital output pin 7 in all sketch!!
//this signal might be use with one PNP transistor or P-Gate MOSFET for complete the circuit
```

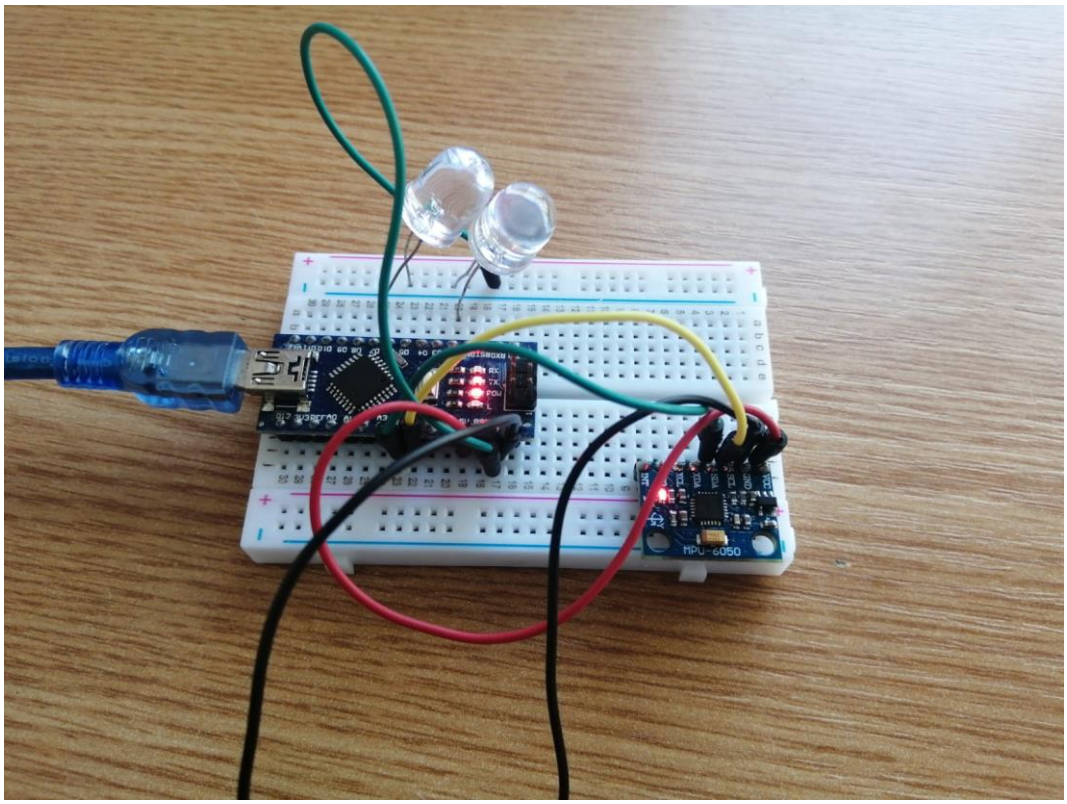
```
//in real circuit I use a 5v blue led. In fritzing circuit use a regular 1.8 red led with aditional
resistor - 180 ohm}
void loop() {
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= interval) {
    previousMillis = currentMillis;
    Wire.beginTransmission(MPU_addr);
    Wire.write(0x3D);
// 0x3B (ACCEL_XOUT_H) & 0x3C (ACCEL_XOUT_L)
// 0x3D (ACCEL_YOUT_H) & 0x3E (ACCEL_YOUT_L)
// 0x3F (ACCEL_ZOUT_H) & 0x40 (ACCEL_ZOUT_L)
    Wire.endTransmission(false);
    Wire.requestFrom(MPU_addr, 2, true);
    int16_t YAxisFull =  (Wire.read() << 8 | Wire.read());
    float YAxisFinal = (float) YAxisFull / 16384.0;
    int YAxis_Filtered = general_filter(c_avg, c_med, YAxisFull);
    int YAxis_kalman = kalman_filter(YAxisFull);
    int YAxis_movavg = mov_avg(c_avg, YAxisFull);
    int YAxis_median = median(c_med, YAxisFull);
    Serial.print("YAxis_Filtered");Serial.print(YAxis_Filtered/16384.0);
    Serial.print("\t");
   // Serial.print("YAxis_kalman");Serial.print(YAxis_kalman);
  //  Serial.print("\t");
  //  Serial.print("YAxis_movavg");Serial.print(YAxis_movavg);
   // Serial.print("\t");
  //  Serial.print("YAxis_median");Serial.print(YAxis_median);
  //  Serial.print("\t");
    Serial.print("YAxisFull");Serial.println(YAxisFull/16384.0);
   // Serial.print("\t");
   // Serial.print("YAxis_Final");Serial.println(YAxis_Final);
//FIRST stage
if (YAxis_Filtered/16384.0 > 0.4 and YAxis_Filtered/16384.0 <= 0.7){
  for (int i = 1; i <= 4; i++) {
```

```
      digitalWrite (2, HIGH);
     // digitalWrite (7, HIGH);
      delay (75);
      digitalWrite (2, LOW);
     // digitalWrite (7, LOW);
      delay (50);}}
else {
  //SECOND stage - very hard brake
if (YAxis_Filtered/16384.0 > 0.7){
   for (int i = 1; i <= 7; i++) {
      digitalWrite (2, HIGH);
     // digitalWrite (7, HIGH);
      delay (75);
      digitalWrite (2, LOW);
      //digitalWrite (7, LOW);
      delay (50);}}}
    c_avg = c_avg + 1;
    c_med = c_med + 1;
    if (c_avg > 5 && c_med > 5)
    {      c_avg = 6;
      c_med = 6;    }  }}
float kalman_filter (float input)
{  float kalman_new = kalman_old;
  float cov_new = cov_old + 0.50;
  float kalman_gain = cov_new / (cov_new + 0.9);
  float kalman_calculated = kalman_new + (kalman_gain * (input - kalman_new));
  cov_new = (1 - kalman_gain) * cov_old;
  cov_old = cov_new;
  kalman_old = kalman_calculated;
  return kalman_calculated;}
float mov_avg (int counter, float input)
{  avg = 0;
  m = 0;
```

```
if (counter == 1) {
  val1 = input;
  avg = val1;  }
else if (counter == 2) {
  val2 = input;
  avg = val2;  }
else if (counter == 3) {
  val3 = input;
  avg = val3;  }
else if (counter == 4) {
  val4 = input;
  avg = val4;  }
else if (counter == 5) {
  val5 = input;
  avg = val5;  }
else if (counter > 5 ) {
  counter = 6;
  if (val1 == 0) {
    m = m + 1;    }
  if (val2 == 0) {
    m = m + 1;    }
  if (val3 == 0) {
    m = m + 1;    }
  if (val4 == 0) {
    m = m + 1;    }
  if (val5 == 0) {
    m = m + 1;    }
  if (input == 0) {
    m = m + 1;    }
  d = 6 - m;
  if (d == 0)
  {      avg = input;
    counter = 1;    }
```

```
    else
    {      avg = (val1 + val2 + val3 + val4 + val5 + input) / d;    }
    val1 = val2;
    val2 = val3;
    val3 = val4;
    val4 = val5;
    val5 = input;  }
  return avg;}
float median (int counter, int input)
{  if (counter == 1) {
    med1_sort = input;
    med_sort[0] = med1_sort;  }
  else if (counter == 2) {
    med2_sort = input;
    med_sort[1] = med2_sort;  }
  else if (counter == 3) {
    med3_sort = input;
    med_sort[2] = med3_sort;  }
  else if (counter == 4) {
    med4_sort = input;
    med_sort[3] = med4_sort;  }
  else if (counter >= 5) {
    counter = 6;
    med_sort[4] = input;
    sort(med_sort, 5);
    med = med_sort[2];
    med1_sort = med2_sort;
    med2_sort = med3_sort;
    med3_sort = med4_sort;
    med4_sort = input;
    med_sort[0] = med1_sort;
    med_sort[1] = med2_sort;
    med_sort[2] = med3_sort;
```

```
    med_sort[3] = med4_sort;  }
  return med;}
void sort(int a[], int size) {
  for (int i = 0; i < (size - 1); i++) {
    for (int o = 0; o < (size - (i + 1)); o++) {
      if (a[o] > a[o + 1]) {
        int t = a[o];
        a[o] = a[o + 1];
        a[o + 1] = t;      }    }  }  }
//function with all filters applied
float general_filter (int counter_avg, int counter_med, float input) {
float input_movavg = mov_avg(counter_avg, input);    //Moving Avg application
float input_med = median(counter_med, input_movavg); //Median application
float input_filtered = kalman_filter(input_med);     //Kalman application
  return input_filtered;}
```



Εικόνα: Τελική μορφή του κυκλώματος

4ο PROJECT: LINEFOLLOWER (ΑΥΣΤΡΙΑ)

Τι είναι ένα ρομπότ Linefollower:

Το ρομπότ που ακολουθεί τη γραμμή είναι ένα μηχάνημα που μπορεί να εντοπίσει και να ακολουθήσει τη γραμμή που χαράσσεται στο πάτωμα. Γενικά, η διαδρομή είναι προκαθορισμένη και μπορεί να είναι ορατή ως μαύρη γραμμή σε λευκή επιφάνεια με χρώμα υψηλής αντίθεσης. Σίγουρα, αυτό το είδος ρομπότ θα πρέπει να 'αισθάνεται' τη γραμμή με τους αισθητήρες υπέρυθρων ακτίνων (IR) που είναι εγκατεστημένοι στο κάτω μέρος του ρομπότ. Μετά, τα δεδομένα μεταδίδονται στον επεξεργαστή. Έτσι, ο επεξεργαστής θα αποφασίσει τους κατάλληλες κινήσεις και στη συνέχεια τις στέλνει στον οδηγό και έτσι τη διαδρομή που θα ακολουθήσει το ρομπότ ακολουθώντας τη γραμμή.

Το σημαντικό σημείο της κατασκευής ενός ρομπότ που ακολουθεί γραμμή είναι ένας καλός έλεγχος που επαρκεί για να ακολουθήσει τη διαδρομή όσο το δυνατόν γρηγορότερα.

Λειτουργία ενός ρομπότ Line Follower

Η έννοια του ρομπότ Line Lollower σχετίζεται με το φως. Εδώ, χρησιμοποιούμε την αντίδραση του φωτός στην ασπρόμαυρη επιφάνεια. Το λευκό χρώμα αντανακλά όλο το φως που πέφτει πάνω του, ενώ το μαύρο απορροφά το φως.

Σε αυτό το ρομπότ Line Lollower, χρησιμοποιούμε πομπούς και δέκτες υπερύθρων (φωτοδίοδοι). Χρησιμοποιούνται για την αποστολή και λήψη των φώτων. Όταν οι ακτίνες IR πέφτουν σε μια λευκή επιφάνεια, αντανακλώνται προς τον δέκτη IR, δημιουργώντας κάποιες αλλαγές τάσης.

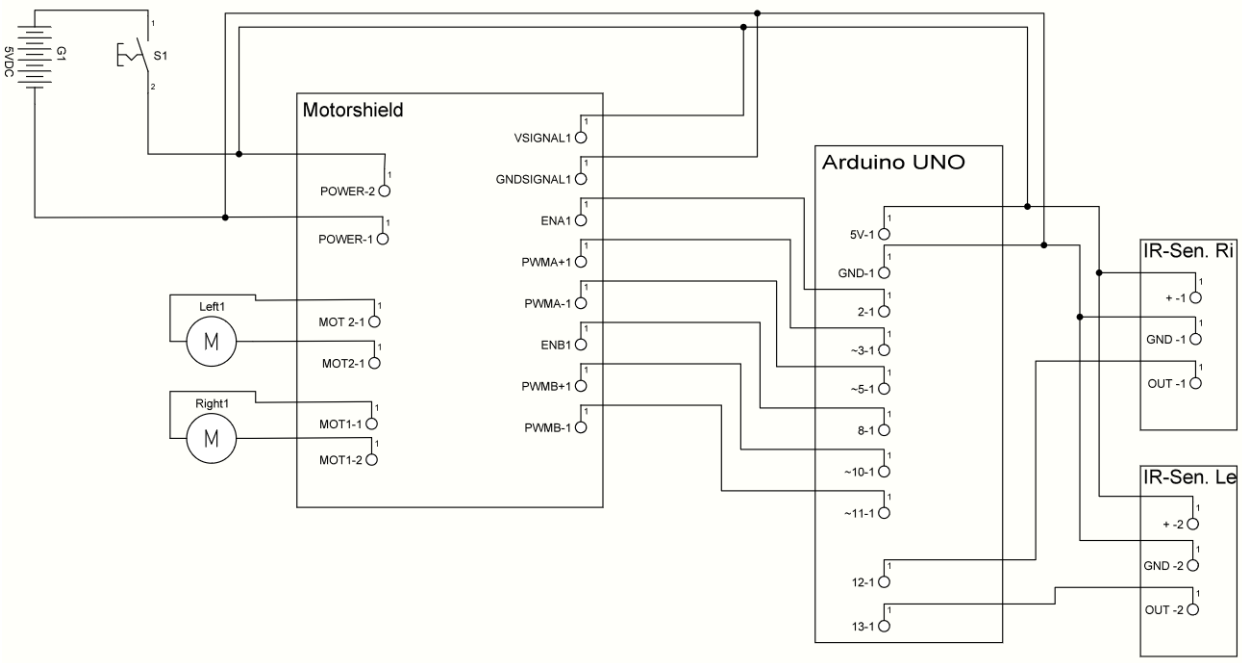
Όταν οι ακτίνες IR πέφτουν σε μια μαύρη επιφάνεια, απορροφάται από τη μαύρη επιφάνεια και δεν αντανακλώνται ακτίνες. Έτσι, ο δέκτης υπερύθρων δεν λαμβάνει ακτίνες.

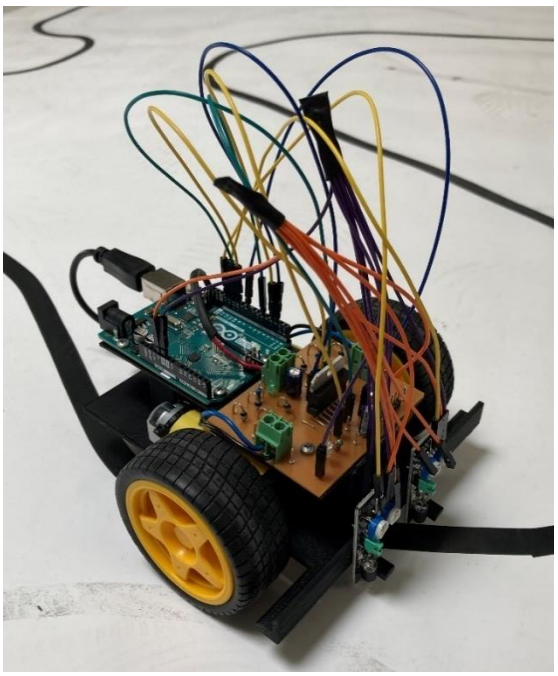
Σε αυτό το έργο, όταν ο αισθητήρας υπερύθρων ανιχνεύει μια λευκή επιφάνεια, ένα Arduino παίρνει 1 (HIGH) ως είσοδο και όταν ανιχνεύει μια μαύρη γραμμή, ένα Arduino παίρνει 0 (LOW) ως είσοδο. Με βάση αυτές τις εισόδους, ένα Arduino Uno παρέχει την κατάλληλη έξοδο για τον έλεγχο του robot.

Κατάλογος υλικών:

υλικά	**τύπος**	**τεμάχια**
Arduino	Uno	1
Motorshield	by HTL Wolfsberg	1
Κινητήρας DC	COM-Motor01, 3-9Volt DC, joy-it	2
PowerBank	Flip12 PhonePowerBank, 3350mAh	1
Αισθητήρες IR	SEN-KY032IR, joy-it	2
Κεντρικός διακόπτης		1
Καλώδια	RB-CB3-025, joy-it	1
Καλώδιο USB	Digitus,AK-300102-010-S, Typ A-B	s

Σχηματικό διάγραμμα του κυκλώματος:





Εικόνα: Τελική μορφή του κυκλώματος

Κώδικας Arduino του Project:

```
// Linefollower
const byte sr = 12;
const byte sl = 13;
const byte enr = 2;
const byte enl = 8;
const byte mrf = 3;
const byte mrb = 5;
const byte mlf = 10;
const byte mlb = 11;

void setup()
{
  Serial.begin(9600);

  pinMode(sr,INPUT);
  pinMode(sl,INPUT);

  pinMode(enr,OUTPUT);
  pinMode(enl,OUTPUT);
  digitalWrite(enr,HIGH);
  digitalWrite(enl,HIGH);

  pinMode(mrf,OUTPUT);
  pinMode(mrb,OUTPUT);
  pinMode(mlf,OUTPUT);
  pinMode(mlb,OUTPUT);

  analogWrite(mrf,0);
  analogWrite(mrb,0);
  analogWrite(mlf,0);
  analogWrite(mlb,0);
}
  bool senR;
  bool senL;

void loop()
{
  senR = digitalRead(sr);
  senL = digitalRead(sl);

  if(!senR && !senL) //staight
  {
    analogWrite(mrf,220);
    analogWrite(mrb,0);
    analogWrite(mlf,200);
    analogWrite(mlb,0);
```

```
  }

  //right sensor on line
  //drive to right
  else if(senR && !senL)
  {
    analogWrite(mrf,0);
    analogWrite(mrb,100);
    analogWrite(mlf,255);//255
    analogWrite(mlb,0);
  }

  //left sensor on line
  //drive to left
  else if(!senR && senL)
  {
    analogWrite(mrf,255);//255
    analogWrite(mrb,0);
    analogWrite(mlf,0);
    analogWrite(mlb,100);
  }

  // both sensors on black
  // stop
  else if(senR && senL)
  {
    analogWrite(mrf,0);
    analogWrite(mrb,0);
    analogWrite(mlf,0);
    analogWrite(mlb,0);
    delay(3000);
  }
}
```

5ο PROJECT: ΑΥΤΟΜΑΤΟΠΟΙΗΜΕΝΟ ΘΕΡΜΟΚΗΠΙΟ (ΙΤΑΛΙΑ)

Στόχος του Project: Γιατί ένα project αυτοματοποιημένου θερμοκηπίου στο σχολείο;

Οι μαθητές στο σχολείο ακολούθησαν διάφορες διαδρομές σχετικά με την ενεργό συμμετοχή του πολίτη και την Ατζέντα του ΟΗΕ 2030 για τη Βιώσιμη Ανάπτυξη, σχετικά με την ενεργειακή βιωσιμότητα, τη σημασία της αποφυγής της σπατάλης νερού και την ανάγκη ανάπτυξης καινοτόμων και φιλικών προς το περιβάλλον μεθόδων καλλιέργειας, λαμβάνοντας υπόψη τις βαθιές κλιματικές αλλαγές. Η οικολογική συνείδηση των νέων εκφράζεται στην αναζήτηση τεχνικών λύσεων και καινοτόμων ιδεών, ξεκινώντας από τις θεωρητικές και εργαστηριακές γνώσεις που αποκτήθηκαν στο σχολείο που είναι σύμφυτες με τις ευκαιρίες εφαρμογής του Arduino.

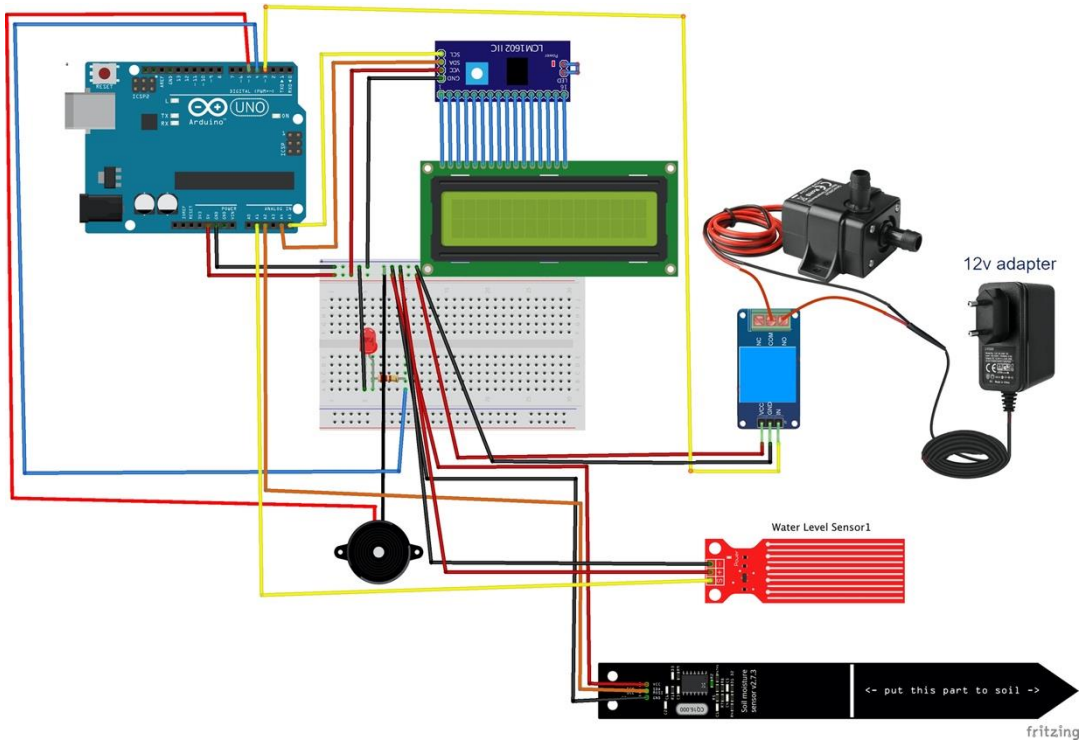
Μερικοί μαθητές, με την υποστήριξη των καθηγητών, σχεδίασαν ένα σύστημα ανίχνευσης της θερμοκρασίας του εδάφους και του αέρα, της υγρασίας του εδάφους, για τη διαχείριση της άρδευσης και του αερισμού ενός πρωτότυπου θερμοκηπίου κατάλληλου για την καλλιέργεια τοπικών φρούτων και λαχανικών.

Ο πειραματισμός έδωσε τα αναμενόμενα αποτελέσματα και θα συμβάλει στην πραγματική υλοποίηση στην περιοχή που βρίσκεται κοντά στο σχολείο που προορίζεται να φιλοξενήσει τον μελλοντικό λαχανόκηπο του σχολείου.

Μεθοδολογία του Project:

Καταχρήν, ετοιμάστηκε μια λίστα με όλες τις παραμέτρους που θέλαμε να χρησιμοποιήσουμε για τη λειτουργία του θερμοκηπίου. Ξεκινώντας από αυτό ετοιμάστηκε μια πρώτη έκδοση του αλγορίθμου του έργου. Σύμφωνα με αυτόν τον αλγόριθμο, αγοράστηκαν οι απαραίτητοι αισθητήρες και δημιουργήθηκε το λογικό διάγραμμα. Το Arduino χρησιμοποιήθηκε στη σχεδίαση του κυκλώματος. Τελικά υλοποιήθηκε το τελικό σενάριο. Αφού ελέγξαμε τη σωστή ρύθμιση των τιμών κατωφλίου των διαφόρων αισθητήρων, τοποθετήθηκαν μέσα στο θερμοκήπιο.

Κύκλωμα του Project.



Λειτουργία:

Μια οθόνη LCD παρέχει τις τιμές της θερμοκρασίας και της υγρασίας του αέρα, της υγρασίας του εδάφους και της ποσότητας νερού που υπάρχει στο δοχείο άρδευσης. Όταν οι τιμές είναι κάτω από ένα συγκεκριμένο όριο, μια μηχανοκίνητη αντλία που ελέγχεται από ένα ρελέ ξεκινά την άρδευση του εδάφους. Όταν η στάθμη του νερού είναι κάτω από ένα συγκεκριμένο όριο, στην οθόνη εμφανίζεται μια προειδοποίηση αναπλήρωσης συνοδευόμενη από ηχητικό συναγερμό.

Κατάλογος υλικών:

Ένα θερμοκήπιο, ένα Arduino Uno R3, ένας αισθητήρας θερμοκρασίας/υγρασίας DHT11, ένας βομβητής, ένας αισθητήρας υγρασίας εδάφους, ένας αισθητήρας στάθμης νερού, ένα κόκκινο LED, ένας προσαρμογέας ρεύματος 12 V, μια αντλία 12 V, μια αντίσταση 220 ohm, ένα ρελέ, μια Οθόνη LCD.

Κώδικας Arduino του Project:

//Project Name: AUTOMATED GREENHOUSE:

```
#include <LiquidCrystal_I2C.h>

#include <Wire.h>

#include <DHT.h>

#define DHTPIN 11

#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE);

LiquidCrystal_I2C lcd(0x27,20,4);  // set the LCD address to 0x27 for a 16 chars and 2 line display

int t, h, Lumen, lumin, water, igro, umdtr, wlevel;


const int pin_water = A1;

const int pin_igro = A2;

const int pin_pump = 3;

const int pin_led = 4;

const int pin_buzzer = 5;


void setup() {

   Serial.begin(9600);


   dht.begin();

   lcd.init();

   lcd.backlight();

   lcd.setCursor(5,0);

   lcd.print("School");

   lcd.setCursor(3,1);

   lcd.print("Greenhouse");

   delay (5000);
```

```
    lcd.clear();
    lcd.setCursor(2,0);
    lcd.print("IIS Einstein");
    lcd.setCursor(3,1);
    lcd.print("De Lorenzo");
    delay (5000);
    lcd.clear();

    pinMode(pin_pump,OUTPUT);
    pinMode(pin_led, OUTPUT);
    pinMode(pin_buzzer, OUTPUT);
    digitalWrite(pin_pump, HIGH);
}

void loop() {
    // water level
    water = analogRead (pin_water);
    wlevel = map (water,0, 1023, 0, 100);
    if(wlevel < 35){
        lcd.clear();
        tone(pin_buzzer, 800);
        digitalWrite(pin_led, HIGH);
        delay(300);
        noTone(pin_buzzer);
        digitalWrite(pin_led, LOW);
        delay(300);
        lcd.clear();
        //lcd.begin(16, 2);
```

```
    lcd.setCursor(2,0);
    lcd.print("Refill Water");
    delay (1000);
    lcd.clear();
} else {
    noTone(pin_buzzer);
    digitalWrite(pin_led, LOW);
}

// Igrometer
igro = analogRead(pin_igro);
umdtr = map (igro, 0, 1023, 0, 100);

if(umdtr < 45){
    digitalWrite(pin_pump, LOW);
    delay(10000);
    digitalWrite(pin_pump, HIGH);
}

// Lettura umidità e temperatura del sensore DHT11
h = dht.readHumidity();
t = dht.readTemperature();

// posiziona il cursore in colonna 0 e linea 1
// (nota: la linea 1 e la seconda linea, poichè si conta incominciando da 0):
lcd.setCursor(0, 0);
lcd.print("T:");
lcd.print(t);
```

```
  lcd.print( (char) 223 );
  lcd.print("C");

  lcd.setCursor(10, 0);
  lcd.print("H:");
  lcd.print(h);
  lcd.print("%");

  lcd.setCursor(0, 1);
  lcd.print("SH:");
  lcd.print(umdtr);
  lcd.print("%");

  lcd.setCursor(10, 1);
  lcd.print("WL:");
  lcd.print(wlevel);
  lcd.print("%");
}
```

Εικόνα: Τελική μορφή του Project

Παραρτήματα

Η αποστολή του Arduino είναι να επιτρέψει σε οποιονδήποτε να βελτιώσει τη ζωή του μέσω προσβάσιμων ηλεκτρονικών και ψηφιακών τεχνολογιών. Κάποτε υπήρχε ένα εμπόδιο μεταξύ του κόσμου των ηλεκτρονικών, του σχεδιασμού και του προγραμματισμού και του υπόλοιπου κόσμου. Ο μικροελεγκτής Arduino έχει σπάσει αυτό το φράγμα. Για περισσότερες πληροφορίες σχετικά με Arduinos, μπορείτε να επισκεφτείτε τον ιστότοπο, παρακάτω…

www.arduino.cc/

Για περισσότερες πληροφορίες σχετικά με το έργο μας ARDinVET, μπορείτε να επισκεφτείτε τον ιστότοπο, παρακάτω…

www.arduinvet.com

«Το έργο αυτό χρηματοδοτείται από το Πρόγραμμα Erasmus+ της Ευρωπαϊκής Ένωσης. Ωστόσο, η Ευρωπαϊκή Επιτροπή και η Οργανωτικός Φορέας της Τουρκίας δεν φέρουν ευθύνη για οποιαδήποτε χρήση των πληροφοριών που περιέχονται σε αυτές».