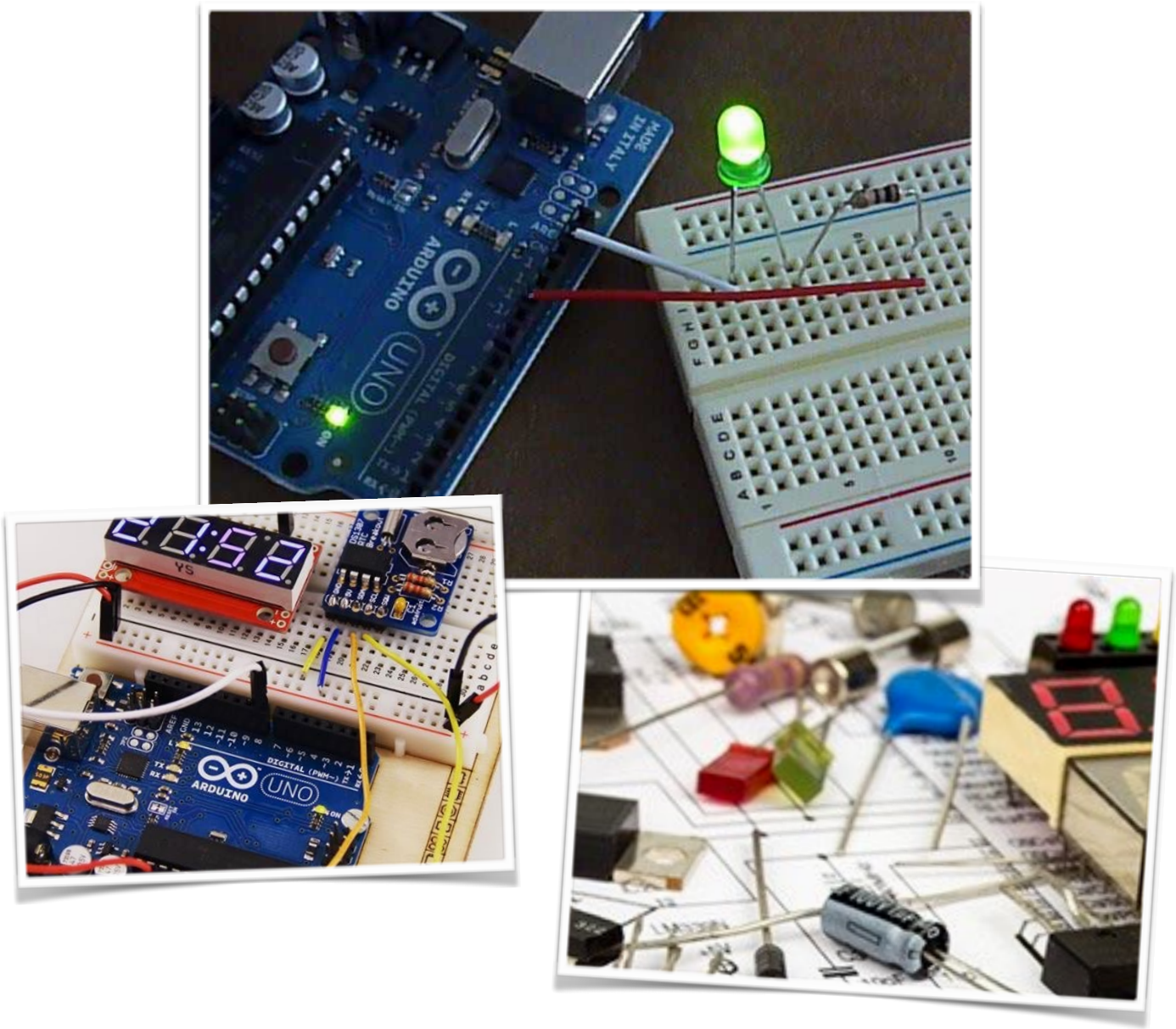
Σετ Μικροελεγκτή

Οδηγίες χρήσης & Ασκήσεις



Πρόλογος

Μάθε μέσα από την άσκηση

Σε αυτό το βιβλίο θα διαβάσεις λίγα λόγια οδηγιών για τα εξαρτήματα που συνοδεύουν το σετ Μικροελεγκτή καθώς και μια σειρά ασκήσεων που μπορείς να εκτελέσεις με τα υλικά του σετ ώστε να κατανοήσεις καλύτερα την λειτουργία των μικροελεγκτών και των δυνατοτήτων αυτών σε αυτοματισμούς και ηλεκτρονικά κυκλώματα.

Οι ασκήσεις που ακολουθούν είναι ενδεικτικές και δείχνουν μερικές από τις πολλές δυνατότητες που έχεις για να πειραματιστής με τα υλικά του σετ και πιστεύουμε ότι θα χρησιμεύσουν σαν αφετηρία για την εξερεύνηση του κόσμου της ηλεκτρονικής.

Εκτός από τις ασκήσεις αυτές στο διαδίκτυο μπορείς να βρεις πολλά ακόμα παραδείγματα στις διευθύνσεις:

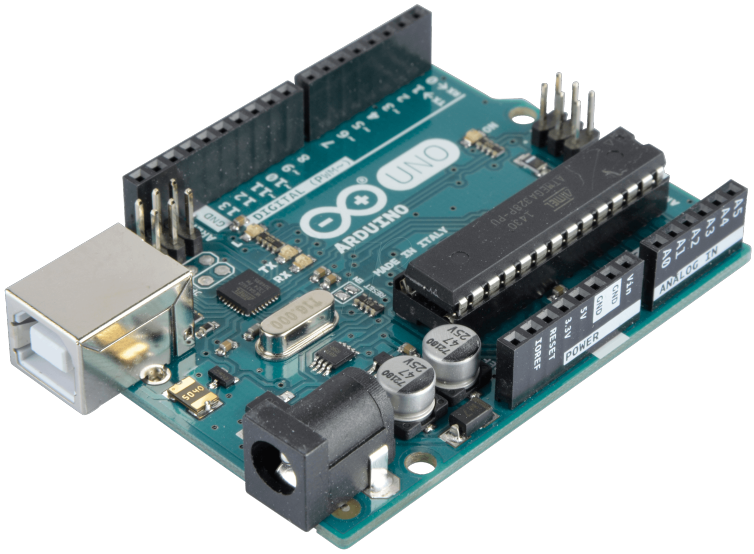
https://www.arduino.cc/en/Tutorial/HomePage

https://learn.adafruit.com/category/learn-arduino

Επίσης στην ιστοσελίδα της εταιρίας μας http://www.scienter.gr/web/el/education/ μπορείτε να βρείτε την τελευταία έκδοση αυτού του βιβλίου καθώς και άλλο εκπαιδευτικό υλικό.

Για την δημιουργία των διαγραμμάτων και των εικόνων συνδεσμολογίας χρησιμοποιήθηκε το λογισμικό ανοιχτού κώδικα Fritzing (http://fritzing.org).

Επίσης δανειστήκαμε και εμπνευστήκαμε για το περιεχόμενο αυτού του βιβλίου από πηγές στο internet όπως το arduino.cc, adafruit.com, forum, blog κλπ.



Εισαγωγή

Το σετ των υλικών και εξαρτημάτων που έχετε μπροστά σας έχει σχεδιαστεί για να σας εισάγει σ' αυτό που λέμε ψηφιακή τεχνολογία

Όλοι μας, κάθε μέρα, χρησιμοποιούμε την ψηφιακή τεχνολογία. Οι περισσότεροι όμως αφήνουμε τον προγραμματισμό στους μηχανικούς και στους προγραμματιστές γιατί πιστεύουμε ότι η δημιουργία κώδικα εντολών και η ηλεκτρονική είναι κάτι το περίπλοκο και δύσκολο. Στην πραγματικότητα όμως η ψηφιακή τεχνολογία μπορεί να γίνει μια διασκεδαστική και συναρπαστική δραστηριότητα.

Με τα εξαρτήματα του σετ θα μάθετε να κατασκευάζετε διατάξεις και ηλεκτρονικά κυκλώματα που χρησιμοποιούμε καθημερινά στη ζωή μας, χωρίς να το πολυσκεφτόμαστε. Η καρδιά του σετ είναι μία μικρή ηλεκτρονική πλακέτα που λειτουργεί σαν ένας μικρός ηλεκτρονικός υπολογιστής ή αλλιώς μικροελεγκτής. Ο μικροελεγκτής αυτός, που είναι γνωστός με διάφορα ονόματα (Genuino Uno ή Arduino Uno ή Funduino Uno), έχει εισόδους για τη σύνδεση εξωτερικών

αισθητήρων, εξόδους για τη σύνδεση εξαρτημάτων ή συσκευών που θέλουμε να ελέγξουμε και μνήμη για την εισαγωγή και αποθήκευση των προγραμμάτων με τα οποία πετυχαίνεται η επιθυμητή διαδικασία. Οι μικροελεγκτές είναι αυτοί που κάνουν τα αντικείμενα να είναι διαδραστικά.

Για χρόνια ο μικροελεγκτής Uno είναι ο «εγκέφαλος» χιλιάδων εργασιών, κάθε μια και περισσότερο δημιουργική από την προηγούμενη. Μια παγκόσμια κοινότητα δημιουργών υιοθέτησε αυτήν την πλατφόρμα ανοικτού κώδικα, περνώντας από την χρήση του προσωπικού υπολογιστή στη διάσταση της προσωπικής κατασκευής, συνεισφέροντας σε έναν ολόκληρο νέο κόσμο συμμετοχής, συνεργασίας και κοινής χρήσης.

Κατασκευή ρομπότ με την χρήση του Genuino

Το Genuino/Arduino/Funduino Uno είναι ανοικτό και απλό. Δημιουργήθηκε ξεκινώντας από την παραδοχή ότι είναι εύκολο και προσβάσιμο να μάθουμε να κάνουμε πράγματα μέσω της ψηφιακής τεχνολογίας. Τελικά η ηλεκτρονική και η δημιουργία κώδικα εντολών έγιναν δημιουργικά εργαλεία τα οποία μπορούν να χρησιμοποιηθούν από όλους. Το εγχειρίδιο αυτό σας καθοδηγεί στις βασικές αρχές με πρακτικό τρόπο ώστε να μαθαίνετε μέσα από την κατασκευή δημιουργικών εργασιών. Από τη στιγμή που θα κατακτήσετε τα βασικά, θα αποκτήσετε μια παλέτα λογισμικού και κυκλωμάτων που μπορείτε να χρησιμοποιήσετε για να φτιάξετε εφαρμογές χρήσιμες για σας και για τους άλλους

Πλήθος μικροελεγκτών μας περιστοιχίζουν καθημερινά: είναι ενσωματωμένοι σε χρονιστές, θερμοστάτες, παιχνίδια, τηλεχειριστήρια, φούρνους μικροκυμάτων, ακόμη και σε μερικές οδοντόβουρτσες. Κάνουν συγκεκριμένο έργο χωρίς να τους ξεχωρίζουμε και το κάνουν καλά. Έχουν προγραμματιστεί να αντιλαμβάνονται και να ελέγχουν την κίνηση χρησιμοποιώντας αισθητήρες και ενεργοποιητές.

Οι αισθητήρες αντιλαμβάνονται το φυσικό κόσμο.
Μετατρέπουν την ενέργεια που εκπέμπουμε πατώντας ένα μπουτόν ή όταν κουνάμε τα χέρια μας ή όταν φωνάζουμε, σε ηλεκτρικά σήματα. Μπουτόν και στρεφόμενα κουμπιά είναι αισθητήρες που ενεργοποιούμε με τα δάχτυλά μας αλλά υπάρχουν και πολλά άλλα είδη αισθητήρων.

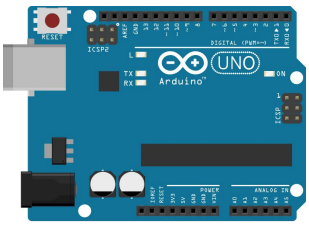
Οι ενεργοποιητές δρουν στον φυσικό κόσμο. Μετατρέπουν την ηλεκτρική ενέργεια σε φυσική ενέργεια, όπως το φως, η θερμότητα και η κίνηση.

Οι μικροελεγκτές "ακούν" τους αισθητήρες και "μιλούν" στους ενεργοποιητές. Αποφασίζουν και δρουν βασιζόμενοι πάνω στο πρόγραμμα που γράφετε.

Οι μικροελεγκτές και τα ηλεκτρονικά που συνδέετε σ' αυτούς είναι απλά ο σκελετός της εργασίας (άσκησης) σας. Θα πρέπει να χρησιμοποιήσετε και τις γνώσεις που έχετε ώστε να δώσετε σάρκα σ' αυτόν το σκελετό.

Το Uno σχεδιάστηκε ώστε να σας βοηθήσει να κάνετε πράγματα. Για να επιτευχθεί αυτός ο στόχος, έχει διατηρηθεί ένα ελάχιστο υπόβαθρο στον προγραμματισμό και τα ηλεκτρονικά.

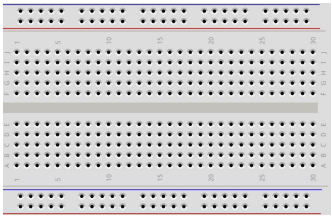
Εξαρτήματα του σετ σας



Μικροελεγκτής Uno – Ο μικροελεγκτής που θα είναι η καρδιά των εργασιών σας. Είναι ένας απλός υπολογιστής, όμως ακόμη δεν έχει μέσα αλληλεπίδρασης μ' εσάς. Θα κάνετε κυκλώματα και διεπαφές για να επιτύχετε την αλληλεπίδραση και θα υποδείξετε στον μικροελεγκτή πώς να επικοινωνεί με άλλα εξαρτήματα..

Καλώδιο USB – Με το καλώδιο αυτό συνδέεται ο μικροελεγκτής με τον προσωπικό υπολογιστή. Επίσης παρέχει και την τροφοδοσία του μικροελεγκτή UNO για τις περισσότερες από τις εργασίες αυτού του σετ.

Συνδετήρας μπαταρίας – Κουμπώνει πάνω σε μια μπαταρία 9V και καταλήγει σε βύσμα το οποίο συνδέεται πάνω στην πλακέτα του μικροελεγκτή σας.



Πλακέτα κυκλωμάτων (Breadboard) – Μια πλακέτα πάνω στην οποία θα κατασκευάζετε ηλεκτρονικά κυκλώματα. Είναι μία διάτρητη πλαστική πλάκα με σειρές από οπές που σας επιτρέπουν να συνδέετε καλώδια και εξαρτήματα μαζί. Η πλακέτα έχει κεντρικά δύο στήλες με ομάδες 5 οριζόντιων οπών, και σε καθένα από τα δύο άκρα της δύο στήλες από μονές οπές. Οι οπές της κάθε πεντάδας οδηγούν σε ένα ηλεκτρικό κόμβο, είναι δηλαδή βραχυκυκλωμένες μεταξύ τους. Το ίδιο συμβαίνει και στις οπές των μονών στηλών,

όπου όμως είναι βραχυκυκλωμένες μεταξύ τους όλες οι οπές της κάθε στήλης (και όχι μόνον ανά πεντάδες). Τοποθετώντας τα άκρα δύο εξαρτημάτων σε δύο οπές της ίδιας πεντάδας δημιουργείται ηλεκτρική σύνδεση μεταξύ των εξαρτημάτων αυτών. Οι μονές στήλες χρησιμοποιούνται κυρίως για τη σύνδεση της τάσης τροφοδοσίας και τη γείωση των κυκλωμάτων που θα κατασκευάζετε.

Βραχυκυκλωτήρες – Χρησιμοποιούνται για την σύνδεση των εξαρτημάτων μεταξύ τους πάνω στην πλακέτα συνδεσμολογιών και στο Genuino.

Πυκνωτής (Capacitor) – Αυτά τα εξαρτήματα αποθηκεύουν και αποδίδουν ηλεκτρική ενέργεια σε ένα κύκλωμα. Όταν η τάση του κυκλώματος έχει υψηλότερη τιμή από αυτή που είναι αποθηκευμένη στον πυκνωτή, επιτρέπει την ροή ρεύματος, φορτίζοντάς τον. Ενώ όταν η τιμή της τάσης του κυκλώματος είναι χαμηλότερη, το αποθηκευμένο φορτίο διοχετεύεται προς τη χαμηλή τάση. Συχνά τοποθετείται μεταξύ της τροφοδοσίας και της γείωσης και κοντά σε ένα αισθητήρα ή ένα κινητήρα ώστε να κρατά ομαλές τις διακυμάνσεις τάσεως.

Κινητήρας συνεχούς ρεύματος (DC motor) – Μετατρέπει την ηλεκτρική ενέργεια σε μηχανική όταν εφαρμοστεί συνεχής τάση στους ακροδέκτες του. Σπείρες καλωδίων εντός του κινητήρα δημιουργούν μαγνητικά πεδία όταν περνά ρεύμα από μέσα τους. Τα μαγνητικά πεδία έλκουν και απωθούν μαγνήτες που είναι ενσωματωμένοι στον κινητήρα, περιστρέφοντας τον άξονά του. Όταν η πολικότητα της τάσης αντιστραφεί, ο κινητήρας θα λειτουργήσει προς την αντίθετη φορά.

Δίοδος (Diode) – Εξάρτημα με δύο άκρα, την άνοδο και την κάθοδο. Επιτρέπει την ροή ρεύματος προς μία κατεύθυνση μόνον. Χρήσιμη όταν στο κύκλωμα υπάρχει κινητήρας ή άλλα φορτία υψηλής τάσης / ρεύματος. Οι δίοδοι έχουν πολικότητα, που σημαίνει ότι παίζει ρόλο ο τρόπος σύνδεσής τους στο κύκλωμα. Τοποθετημένες με τη μια φορά επιτρέπουν την ροή ρεύματος. Τοποθετημένες διαφορετικά το εμποδίζουν. Για να υπάρχει ροή ρεύματος η άνοδος συνδέεται σε σημείο υψηλότερης τάσης στο κύκλωμα απ' ότι η κάθοδος. Γενικά, η κάθοδος συνδέεται σε σημείο χαμηλής τάσης ή στη γείωση. Υπάρχουν πολλοί τύποι

διόδων, ανάλογα με το ρεύμα που μπορούν να περάσουν μέσα τους. Η πλευρά της καθόδου σημειώνεται καθαρά πάνω στο εξάρτημα με μια περιφερειακή γραμμή (άσπρη ή μαύρη, ανάλογα με τον τύπο της διόδου)

Φωτοδίοδος (photodiode) – Ένας τύπος διόδου που παράγει τάση ανάλογη με την ένταση του φωτός που πέφτει επάνω της.

Φωτοδίοδος LED (λαμπάκια LED) – Ένας τύπος διόδου που ακτινοβολεί όταν την διαπερνά ρεύμα. Όπως σε όλες τις διόδους έτσι και στις φωτοδιόδους, το ηλεκτρικό ρεύμα τις διαπερνά και αυτές κατά μια μόνο κατεύθυνση. Είσαστε πιθανώς εξοικειωμένοι μαζί τους αφού χρησιμοποιούνται σε ενδείκτες σε ποικιλία ηλεκτρονικών συσκευών. Η άνοδος, η οποία τις περισσότερες φορές συνδέεται με την τροφοδοσία, είναι συνήθως το πιο μακρύ από τα δύο άκρα της φωτοδιόδου και η κάθοδος το πιο κοντό.

Φωτοδίοδος RGB (RGB LED) – Είναι μία φωτοδίοδος LED με τέσσερα άκρα, της οποίας το χρώμα εξαρτάται από τον συνδυασμό των τάσεων που εφαρμόζονται στα άκρα της

Οθόνη υγρών κρυστάλλων LCD – Ένας τύπος αλφαριθμητικής ή γραφικής οθόνης που απαρτίζεται από υγρούς κρυστάλλους. Υπάρχουν διαθέσιμες σε διάφορα μεγέθη, σχήματα και στυλ. Η οθόνη του σετ σας διαθέτει δυο σειρές των δεκαέξι χαρακτήρων η κάθε μια.

Βομβητής (buzzer) – Είναι ένα ηλεκτρονικό εξάρτημα (πιεζοηλεκτρικό) το οποίο μπορεί να χρησιμοποιηθεί για να ανιχνεύει δονήσεις και να δημιουργεί θορύβους.

Αντιστάσεις (Resistors) – Αντιστέκονται στην ροή ηλεκτρικής ενέργειας σε ένα κύκλωμα, και επηρεάζουν τις τιμές του ρεύματος και της τάσης. Οι τιμές των αντιστάσεων μετρούνται σε Ωμ και αποδίδονται με χρωματιστές περιφερειακές ζώνες, βάσει ενός χρωματικού κώδικα

Φωτοαντίσταση – Επίσης γνωστή σαν φωτοκύτταρο. Είναι μία μεταβλητή αντίσταση η οποία αλλάζει την τιμή της ανάλογα με την ένταση του φωτός που πέφτει επάνω της.

Ποτενσιόμετρο (Potentiometer) – Είναι μια μεταβλητή αντίσταση με τρία άκρα. Δυο από αυτά είναι συνδεδεμένα στα άκρα μιας αντίστασης σταθερής τιμής. Το μεσαίο άκρο

κινείται κατά μήκος της αντίστασης, χωρίζοντάς στην σε δύο μέρη. Όταν τα άκρα του ποτενσιόμετρου είναι συνδεδεμένα με κάποια τάση και μια γείωση, το μεσαίο άκρο δίνει τη διαφορά τάσης σε σχέση με τη γείωση, καθώς γυρίζουμε το μηχανισμό που κινεί το μεσαίο άκρο.

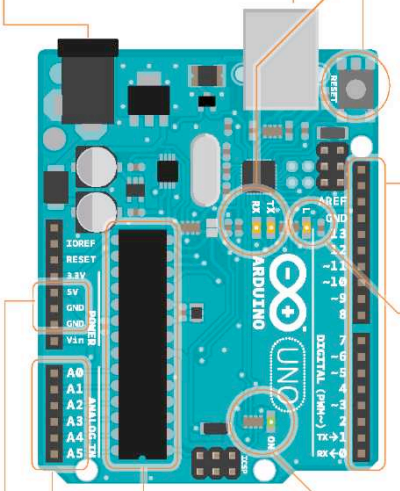
Κουμπί πίεσης (μπουτόν) – Στιγμιαίος διακόπτης ο οποίος κλείνει την επαφή του όταν πιεστεί. Τοποθετείται στην πλακέτα συνδεσμολογιών εύκολα. Είναι κατάλληλος για την δημιουργία σημάτων on/off.

Αισθητήρας θερμοκρασίας – Διαθέτει έξοδο τάσης η οποία εξαρτάται από την θερμοκρασία του αισθητήρα. Τα δύο άκρα συνδέονται στην τροφοδοσία και την γείωση. Η τιμή της τάσης στο μεσαίο ακροδέκτη μεταβάλλεται καθώς θερμαίνεται ή ψύχεται ο αισθητήρας.

Τρανζίστορ – Είναι ένα εξάρτημα με τρεις ακροδέκτες το οποία μπορεί να λειτουργήσει και σαν ηλεκτρονικός διακόπτης. Χρήσιμο για τον έλεγχο συσκευών υψηλής τάσης ή υψηλού ρεύματος όπως οι κινητήρες. Το ένα άκρο συνδέεται στη γείωση, ένα άλλο με τη συσκευή που θέλουμε να ελέγξουμε και το τρίτο με τον μικροελεγκτή UNO. Όταν το εξάρτημα δεχθεί τάση στον ακροδέκτη που είναι συνδεδεμένος με τον μικροελεγκτή, κλείνει το κύκλωμα μεταξύ της γείωσης και της ελεγχόμενης συσκευής.

Αισθητήρας ανίχνευσης κίνησης (PIR Sensor) – Ανιχνεύει την κίνησης μέσω της μεταβολής της υπέρυθρης ακτινοβολίας που εκπέμπουν όλα τα σώματα. Βρίσκει ευρεία χρήση σε συναγερμούς

Η ΠΛΑΚΕΤΑ



Θύρα τροφοδοσίας
Μέσω αυτής μπορούμε να τροφοδοτήσουμε το Arduino όταν δεν το έχουμε συνδέσει σε USB θύρα για τροφοδοσία. Δέχεται από 7-12V.

Θύρα USB
Χρησιμοποιείται για την τροφοδοσία του Arduino , ανέβασμα των προγραμμάτων (sketch) στο Arduino, και για επικοινωνία με το πρόγραμμα (sketch) που τρέχει το Arduino (μέσω εντολών: Serial. println()

Κουμπί Reset
Κάνει resets στον ATmega microcontroller.

TX & RX φωτάκια (LEDs)
Αυτά τα LEDs δείχνουν την επικοινωνία μεταξύ του Arduino και του υπολογιστή. Θα πρέπει να αναβοσβήνουν γρήγορα κατά την "φόρτωση" του προγράμματος, όπως και κατά την σειριακή επικοινωνία. Είναι πολύ χρήσιμα για έλεγχο (debugging).

Ψηφιακά pins
Χρησιμοποιούμε αυτά τα pins με τις digital-Read(), digitalWrite() εντολές. Οι analogWrite(). analogWrite() λειτουργούν μόνο με τα pins που φέρουν το σύμβολο "PWM".

Pin 13 φωτάκι (LED)
Το μόνο ενσωματωμένο ενδεικτικό ενεργοποίησης στο Arduino. Επίσης μπορεί να χρησιμοποιηθεί για το πρώτο σας πρόγραμμα με led. Είναι πολύ χρήσιμο και για έλεγχο (debugging).

GND & 5V pins
Για την τροφοδοσία με +5V & GND των κυκλωμάτων.

Μικρο-ελεγκτής ATmega
Η καρδιά του Arduino.

Αναλογικές είσοδοι
Χρησιμοποιούμε αυτά τα pin με την analogRead() εντολή.

LED τροφοδοσίας
Παρέχει την ένδειξη τροφοδοσίας ή όχι του Arudino. Χρήσιμο για έλεγχο

ΕΓΚΑΤΑΣΤΑΣΗ

ΚΑΛΩΣ ΗΡΘΑΤΕ ΣΤΟΝ ΚΟΣΜΟ ΤΟΥ ARDUINO! ΠΡΙΝ ΑΡΧΙΣΟΥΜΕ ΝΑ ΕΛΕΓΧΟΥΜΕ ΤΟΝ ΚΟΣΜΟ ΓΥΡΩ ΜΑΣ, ΘΑ ΠΡΕΠΕΙ ΝΑ ΚΑΤΕΒΑΣΟΥΜΕ (DOWNLOAD) ΤΟ ΛΟΓΙΣΜΙΚΟ ΓΙΑ ARDUINO (IDE) ΓΙΑ ΝΑ ΠΡΟΓΡΑΜΜΑΤΙΣΟΥΜΕ ΤΗΝ ΠΛΑΚΕΤΑ.

Το Arduino IDE μας επιτρέπει να φτιάχνουμε προγράμματα και να τα φορτώνουμε (upload) στο Arduino

Μπορούμε να κατεβάσουμε την τελευταία έκδοση του IDE από:
Arduino.cc/download

Να έχεις την πλακέτα του Arduino και το καλώδιο USB δίπλα στον υπολογιστή σου. Μην τα συνδέσεις ακόμα.

Ακολούθησε την σωστή διαδικασία όπως περιγράφεται στις επόμενες σελίδες.

ΕΓΚΑΤΑΣΤΑΣΗ LINUX

Έκδοση Online
Arduino.cc/linux

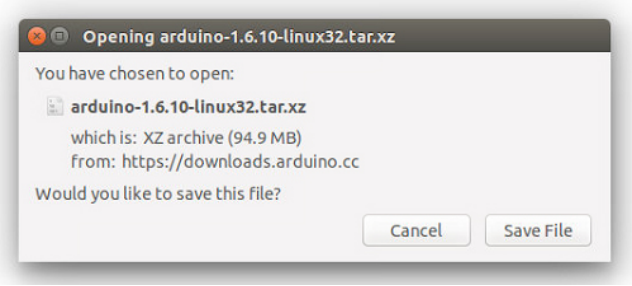
Γενικά

Το λογισμικό του Arduino (IDE) για Linux, είναι ένα πακέτο που δεν απιτεί ξεχωριστή διαδικασία για κάθε έκδοση του Linux. Η μόνη σχετική διαφοροποίηση αφορά την έκδοση του λειτουργικού συστήματος (OS), αν είναι 32 ή 64 bit.

Λογισμικό για Arduino (IDE)

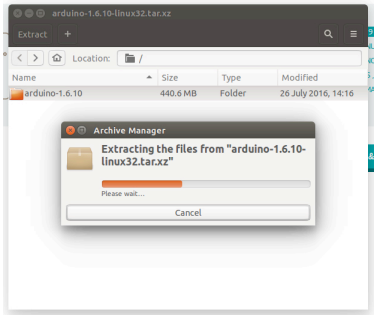
Μπορούμε να βρούμε την τελευταία έκδοση στο:
https://www.arduino.cc/en/Main/Software

Μπορούμε να διαλέξουμε μεταξύ των 32, 64 & ARM εκδόσεων. Είναι σημαντικό να επιλέξουμε την σωστή. Επιλέγοντας την σωστή έκδοση, εμφανίζεται η σελίδα για "δωρεά" και εμείς επιλέγουμε να σώσουμε το αρχειο στον υπολογιστή μας.



Αποσυμπίεση

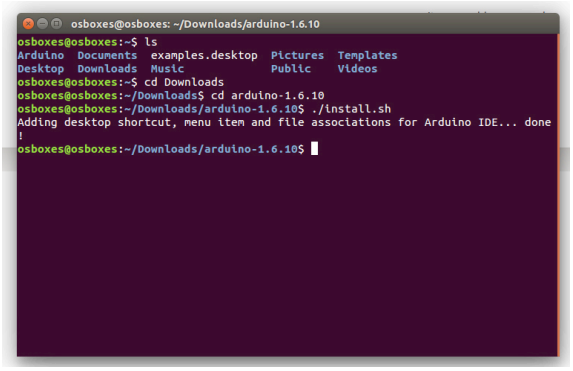
Το αρχείο είναι συμπιεσμένο και θα πρέπει να το αποσυμπιέσουμε στον φάκελο από τον οποίο θα το εκτελέσουμε κιόλας.



Αρχείο εγκατάστασης

Στον φάκελο "arduino-1.6.x" που δημιουργήθηκε μετά την αποσυμπίεση, βρίσκουμε το αρχείο install.sh . Κάνουμε δεξί κλίκ σε αυτό και επιλέγουμε Run in Terminal. Η διαδικασία εγκατάστασης θα τελειώσει σύντομα και ένα νέο εικονίδιο θα δημιουργηθεί στην επιφάνεια εργασίας.

Αν δεν βρίσκουμε την επιλογή Run in Terminal, θα πρέπει να ανοίξουμε ένα παράθυρο (Terminal window) και να πάμε στον φάκελο arduino-1.6.x . Πληκτρολογούμε την εντολή ./install.sh και περιμένουμε να ολοκληρωθεί η εγκατάσταση.



Προγραμματισμός πλακέτας

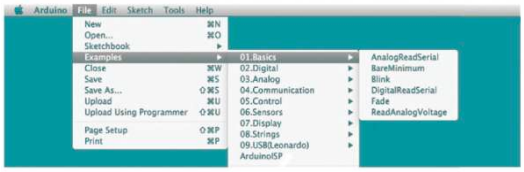
Όταν εγκατασταθεί σωστά το λογισμικό για Arduino (IDE), μπορούμε να ξεκινήσουμε τον προγραμματισμό του.

Επικοινωνία με το Arduino

Μετά την εγκατάσταση του λογισμικού Arduino (IDE), ελέγχουμε την επικοινωνία μεταξύ της πλακέτας και του Η/Υ.

1 Κάνουμε διπλό κλικ στην εφαρμογή του Arduino για να ανοίξει. Αν το λογισμικό είναι σε λάθος γλώσσα , μπορούμε να την αλλάξουμε.
Περισσότερες πληροφορίες θα βρούμε στην ενότητα "Language Support" στην ιστοσελίδα: *arduino.cc/ide*

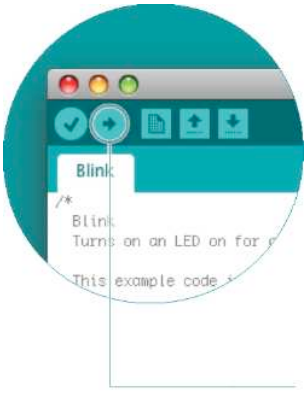
2 Εργαζόμαστε στο παράδειγμα προγραμματισμού (sketch) με τίτλο: "LED Blink".
Θα το βρούμε στο: *FILE>EXAMPLES>01.BASICS>BLINK*



3 Θα ανοίξει ένα παράθυρο με κείμενο. Το αφήνουμε ως έχει προς το παρόν, και επιλέγουμε τον τύπο της πλακέτας που διαθέτουμε από το: *TOOLS>BOARD* menu

4 Επιλέγουμε την σειριακή πόρτα του Η/Υ που έχουμε συνδέσει το Arduino από το: *TOOLS>SERIAL PORT* menu.

Σε Windows
Θα είναι η σειριακή θύρα (COM port) με το μεγαλύτερο νούμερο. Δεν πειράζει αν κάνουμε λάθος. Σε περίπτωση λάθους δοκιμάζουμε την επόμενη θύρα κ.ο.κ.
Για να την βρούμε, μπορούμε να αποσυνδέσουμε το Arduino, και να ξανα-ανοίξουμε το μενού με τις θύρες. Η θύρα που δεν εμφανίζεται πλέον (ενώ με το Arduino εμφανιζόταν), είναι η σωστή. Επανασυνδέουμε την πλακέτα και επιλέγουμε την σωστή θύρα.

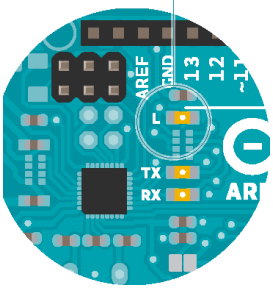


Σε Mac
Θα είναι κάτι με το */dev/tty.usbmodem* μέσα.Συνήθως υπάρχουν 2 από αυτές. Διάλεξε την μία.

5 Για να φορτώσουμε το πρόγραμμα (Blink sketch) στο Arduino, επιλέγουμε το κουμπί *UPLOAD* στην πάνω αριστερή γωνία του παραθύρου.

6 Θα πρέπει να εμφανιστεί μία μπάρα ένδειξης της προόδου φόρτωσης (upload progress), κοντά στην αριστερή κάτω γωνία του λογισμικού (IDE), ενώ τα φωτάκια (leds) TX & RX πάνω στην πλακέτα, θα αναβοσβήνουν γρήγορα. Αν η φόρτωση ολοκληρωθεί επιτυχημένα, στο λογισμικό θα εμφανιστεί το μήνυμα: *DONE UPLOADING*

7 Λίγα δευτερόλεπτα μετά την ολοκλήρωση, θα δούμε το κίτρινο led με ένα *L* δίπλα του, να αναβοσβήνει.
Σε αυτήν την περίπτωση έχουμε επιτυχημένα φορτώσει το πρόγραμμα για να αναβοσβήνει το ενσωματωμένο led του Arduino.



Μερικές φορές μπορεί ένα καινούριο Arduino να είναι προγραμματισμένο ήδη με το "Blink Led" πρόγραμμα. Σε αυτήν την περίπτωση για να ελέγξουμε ότι το Arduino έχει το δικό μας πρόγραμμα, μπορούμε να αλλάξουμε το *delay* (καθυστέρηση), αλλάζοντας το νούμερο στην παρένθεση σε 100, και να ξανα-φορτώσουμε το πρόγραμμα. Τώρα το led θα πρέπει να αναβοσβήνει πολύ πιο γρήγορα.

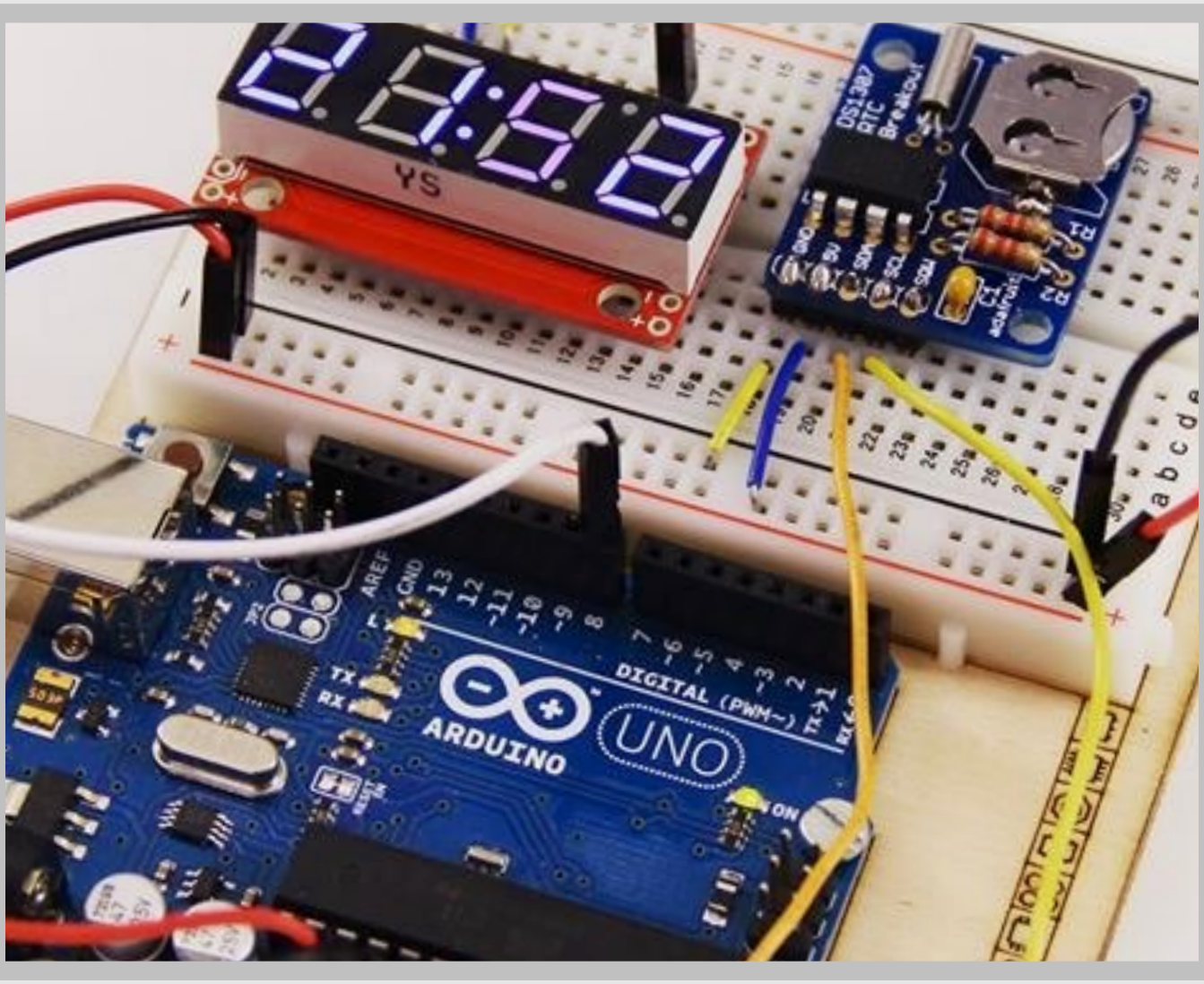
Πληροφορίες

Αν έχουμε προβλήματα με την παραπάνω διαδικασία, μπορούμε να δούμε τον οδηγό επίλυσης προβλημάτων στο:
Arduino.cc/trouble

Πριν ξεκινήσουμε να φτιάχνουμε το δικό μας πρόγραμμα, μπορούμε να βρούμε περισσότερες πληροφορίες σχετικά με το περιβάλλον προγραμματισμού στο:
Arduino.cc/ide

Για παραδείγματα προγραμμάτων με την χρήση διαφόρων αισθητήρων, μπορούμε να δούμε στο:
Arduino.cc/tutorial

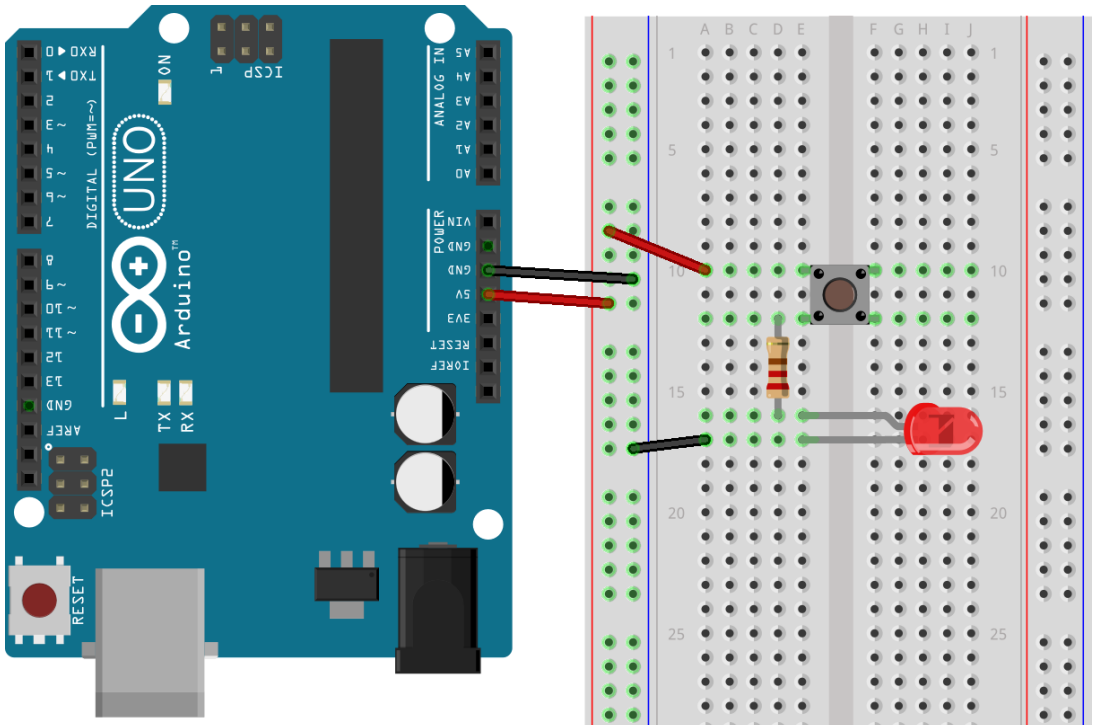
ΑΣΚΗΣΕΙΣ



ΤΟ ΤΑΞΙΔΙ ΑΡΧΙΖΕΙ

Πριν αρχίσεις να συνδέεις τα υλικά των ασκήσεων στο breadboard θα πρέπει ο μικροελεγκτής UNO να μην είναι συνδεδεμένο σε υπολογιστή ή σε ρεύμα. Έλεγξε σχολαστικά την συνδεσμολογία πριν το συνδέσεις σε υπολογιστή ή ρεύμα. Ποτέ μην κάνεις αλλαγές στη συνδεσμολογία των υλικών με το UNO σε ρεύμα ή συνδεδεμένο σε υπολογιστή.

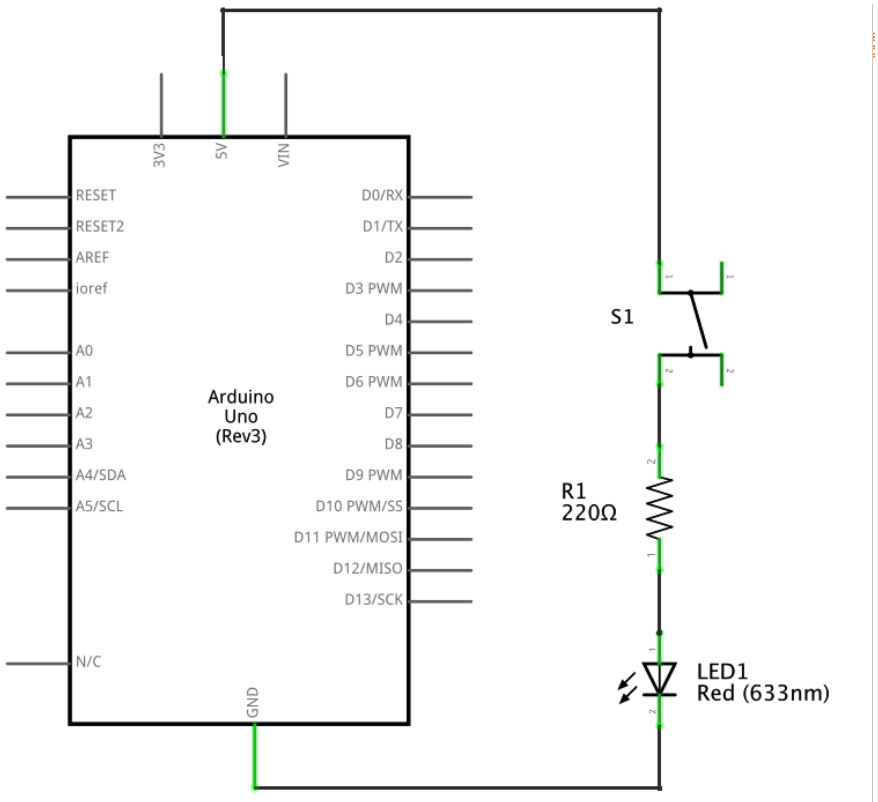
⚠ **ΠΡΟΣΟΧΗ**



ΟΙ

Άναψε το φως

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Εφαρμογή

Σ' αυτό το μάθημα θα μάθετε πώς να ανάβετε ένα λαμπάκι LED με τη χρήση ενός διακόπτη, χρησιμοποιώντας την πλακέτα του μικροελεγκτή UNO. Όταν ο διακόπτης είναι πατημένος κλείνει το ηλεκτρικό κύκλωμα και τροφοδοτεί το λαμπάκι LED με τάση κάνοντάς το να ανάβει.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελλεγκτής UNO
2. Πειραματική πλακέτα
3. Ένας Διακόπτης μπουτόν (push ON)
4. Ένα λαμπάκι LED κόκκινο (ή κάποιο άλλο χρώμα)
5. Μία Αντίσταση 220Ω
6. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Θα παρατηρήσετε ότι υπάρχει μία αντίσταση συνδεδεμένη σε σειρά με ένα λαμπάκι (φωτοδίοδο) LED. Το άλλο άκρο της αντίστασης συνδέεται στο ένα άκρο του διακόπτη push ON , το άλλο άκρο του οποίου συνδέεται στην έξοδο 5V του μικροελεγκτή UNO. Η κάθοδος της φωτοδιόδου συνδέεται στη γείωση (GND) του μικροελεγκτή που είναι και η γείωση του κυκλώματος. Ο ρόλος της αντίστασης (R1) είναι να περιορίζει το ρεύμα που περνάει μέσα από τη φωτοδίοδο προστατεύοντάς την από υπερθέρμανση και καταστροφή της.

Κώδικας

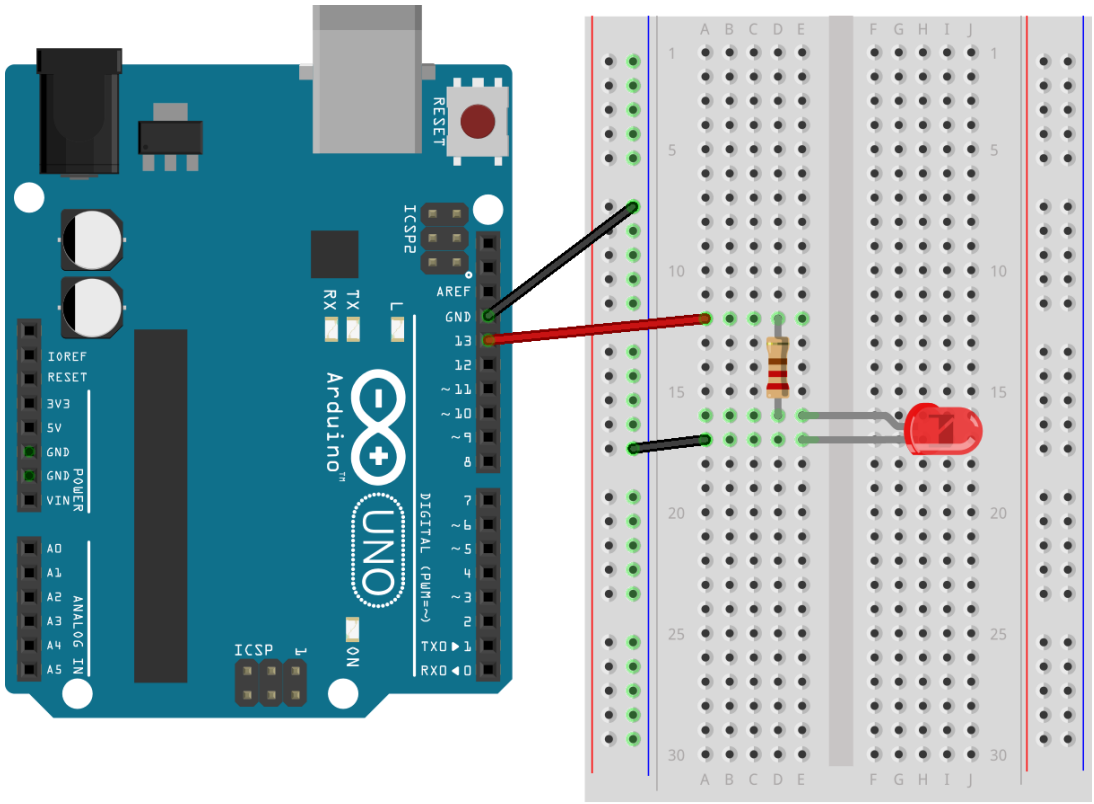
Για αυτή την άσκηση δεν χρειάζεται να φτιάξεις ένα πρόγραμμα και να το φορτώσεις στο Genuimo γιατί το κύκλωμα τροφοδοτείται από την έξοδο 5V του UNO, που

δίνει πάντα 5V, ανεξάρτητα από το πρόγραμμα που έχει φορτωμένο.

Εργασία

1) Τροποποίησε το παραπάνω κύκλωμα προσθέτοντας ένα ακόμη μπουτόν στην πειραματική πλακέτα ώστε να απαιτείται το πάτημα δύο μπουτόν ταυτόχρονα προκειμένου να ανάψει το λαμπάκι.

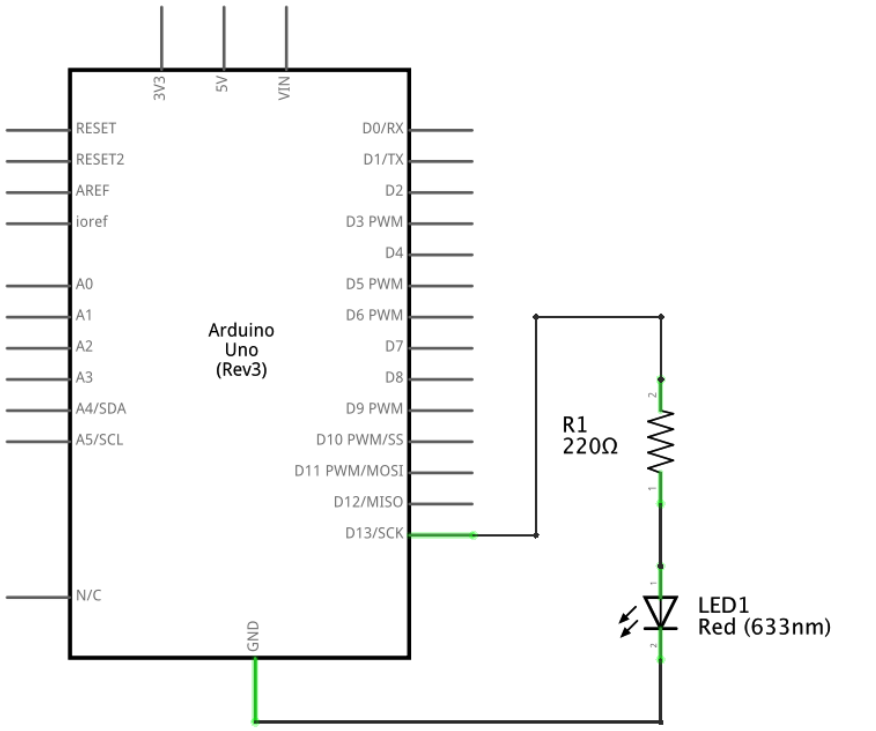
2) Τροποποίησε το παραπάνω κύκλωμα προσθέτοντας ένα ακόμη μπουτόν στην πειραματική πλακέτα ώστε να απαιτείται το πάτημα δύο μπουτόν ταυτόχρονα προκειμένου να ανάψει το λαμπάκι.



02

Φάρος

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Εφαρμογή

Σ' αυτό το μάθημα θα μάθετε πώς να προγραμματίζετε τον Μικροελεγκτη UNO ώστε να κάνετε ένα λαμπάκι LED τοποθετημένο στην πειραματική πλακέτα να αναβοσβήνει, προσομοιώνοντας τη λειτουργία ενός φάρου.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. ΄Ενας μικροελλεγκτής UNO
2. Το καλώδιο USB διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Πειραματική πλακέτα
4. ΄Ενα λαμπάκι LED κόκκινο (ή κάποιο άλλο χρώμα)
5. Μία Αντίσταση 220Ω
6. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Θα παρατηρήσετε ότι υπάρχει μία αντίσταση, της οποίας το ένα άκρο συνδέεται στον ακροδέκτη 13 του μικροελεγκτη UNO και το άλλο στην άνοδο της λυχνίας LED της οποίας η κάθοδος συνδέεται στη γείωση (GND) του κυκλώματος. Ο ρόλος της αντίστασης (R1) είναι να περιορίζει το ρεύμα που περνάει μέσα από τη λυχνία LED προστατεύοντάς την από υπερθέρμανση και καταστροφή της.

Κώδικας

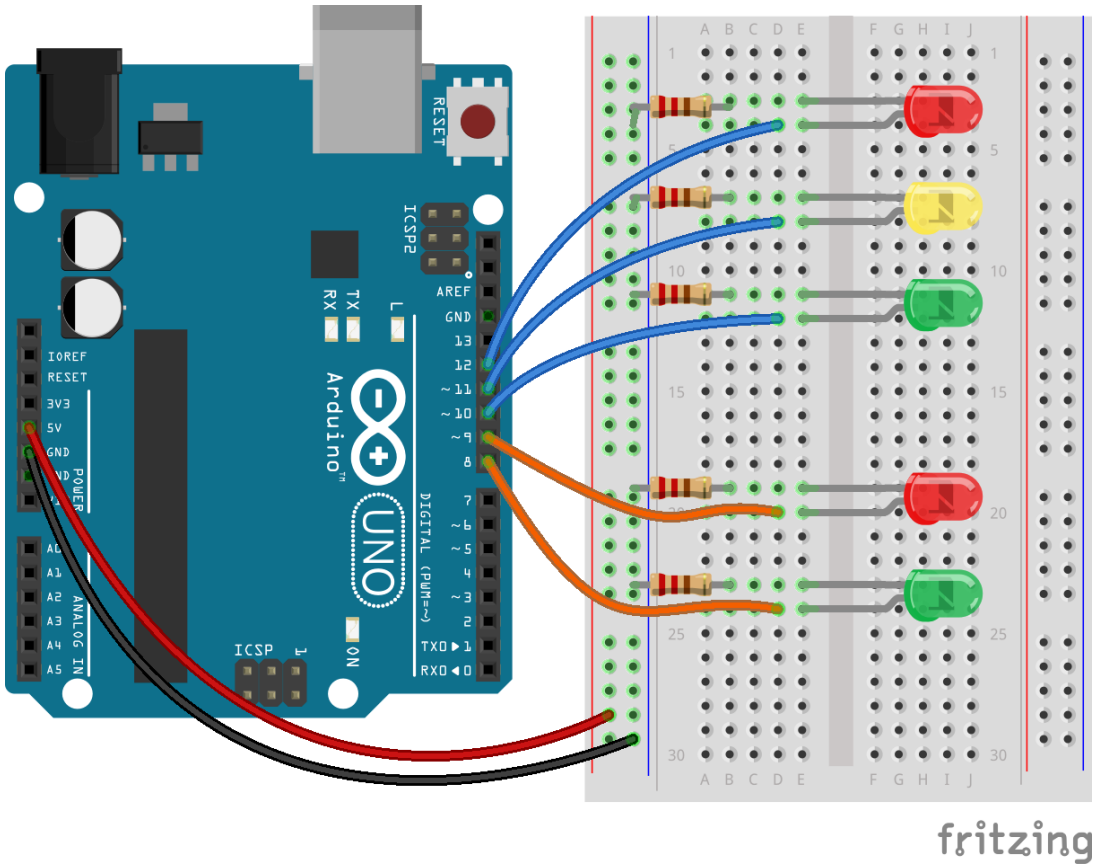
Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

```
/*
Turns on a LED on for one second, then off for
one second, repeatedly.
```

```
*/

// constants won't change. They're used here to
// set pin numbers:
// in Pin 13 we will connect a LED.
// Give it a name.
const int led = 13;

// the setup routine runs once when you press
// reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again
//forever:
void loop() {
  // turn the LED on (HIGH is the voltage level)
  digitalWrite(led, HIGH);
  // wait for 3 seconds
  delay(3000);
  // turn the LED off by making the voltage LOW
  digitalWrite(led, LOW);
  // wait for a second
  delay(1000);
}
```

Εργασία

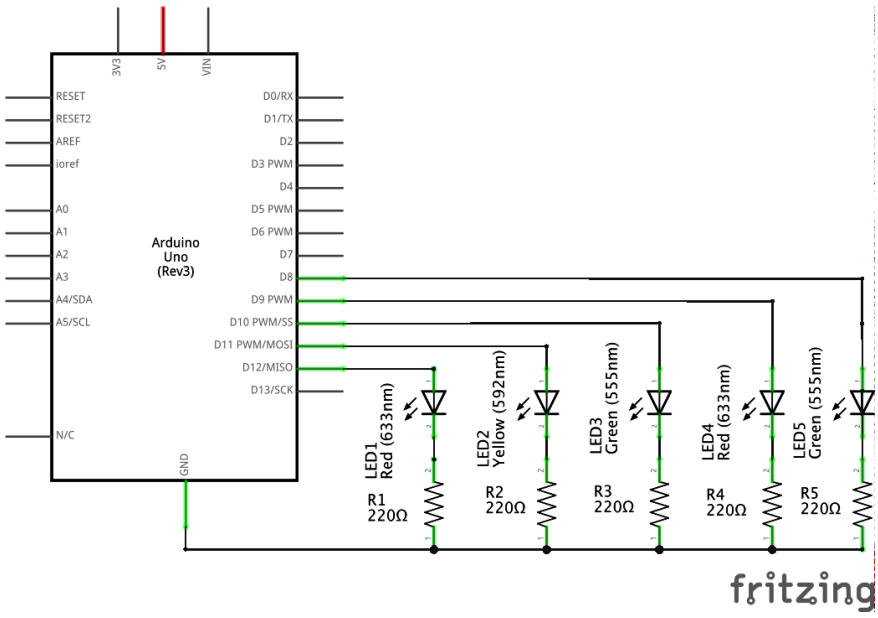
Τροποποίησε την άσκηση ώστε ο φάρος να εκπέμπει το σήμα SOS σε κώδικα μορς.



03

Φανάρι κυκλοφορίας

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Εφαρμογή

Με την άσκηση αυτή θα μάθεις πως να κάνεις την λειτουργία ενός φαναριού κυκλοφορίας προγραμματιστικά.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Πέντε λαμπάκια LED (2 κόκκινα, 2 πράσινα και ένα κίτρινο)
5. Πέντε Αντιστάσεις 220Ω
6. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Θα παρατηρήσετε ότι και πάλι σε σειρά με κάθε λαμπάκι LED υπάρχει μία αντίσταση, ενώ το άλλο άκρο του κάθε LED συνδέεται σε διαφορετικό ακροδέκτη της πλακέτας του μικροελεγκτή (ακροδέκτες 8 – 12). Οι ακροδέκτες αυτοί είναι προγραμματιζόμενοι και στη συγκεκριμένη περίπτωση έχουν οριστεί σαν έξοδοι. Τα λαμπάκια LED έχουν χωριστεί σε δύο ομάδες. Η πρώτη ομάδα αποτελείται από τρία LED (κόκκινο, κίτρινο, πράσινο) και αντιστοιχεί στο φανάρι για τα αυτοκίνητα και η δεύτερη αποτελείται από δύο LED (κόκκινο και πράσινο) και αντιστοιχεί στο φανάρι των πεζών. Το πρόγραμμα ανάβει διαδοχικά το πράσινο, το κίτρινο και το κόκκινο LED της πρώτης ομάδας για χρονικά διαστήματα που ορίζονται προγραμματιστικά και προσομοιώνει τη λειτουργία του φαναριού για τα αυτοκίνητα. Όταν το φανάρι των αυτοκινήτων γίνει κόκκινο, και μετά από κάποια δευτερόλεπτα, ανάβει το πράσινο των πεζών. Μετά από κάποιο χρονικό διάστημα

ανάβει το κόκκινο LED των πεζών και μετά από λίγο το πράσινο LED για τα αυτοκίνητα και ούτω καθ' εξής.

Κώδικας

Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

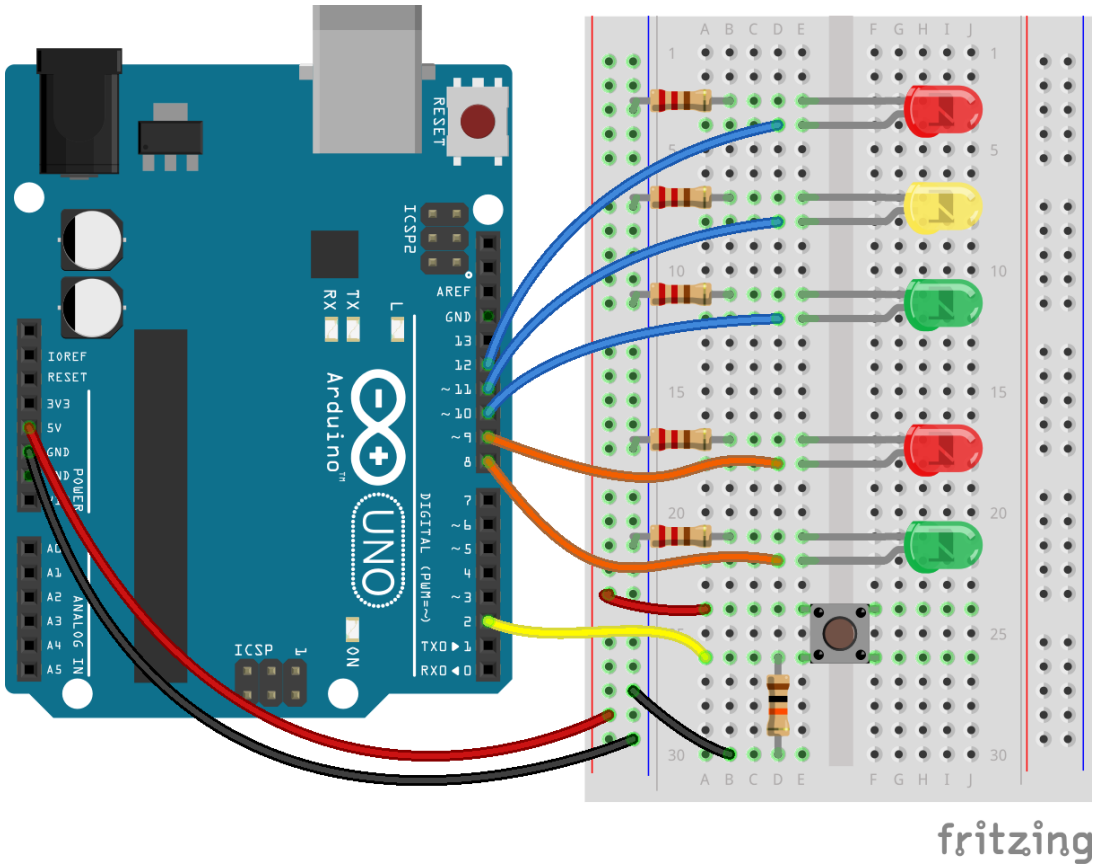
```
/*
Traffic light
*/

// constants won't change. They're used here to
// set pin numbers:
// in Pin 12 we will connect the car red LED.
// Give it a name.
int carRed = 12;
// in Pin 11 we will connect the car yellow LED.
// Give it a name.
int carYellow = 11;
// in Pin 10 we will connect the car green LED.
// Give it a name.
int carGreen = 10;
// in Pin 9 we will connect the pedestrian red
// LED. Give it a name.
int pedRed = 9; //ped lights being assigned
// in Pin 8 we will connect the pedestrian green
// LED. Give it a name.
int pedGreen = 8;
//crossing time given to pedestrians
int crossTime = 5000;

// the setup routine runs once when you press
// reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(carRed, OUTPUT);
  pinMode(carYellow, OUTPUT);
  pinMode(carGreen, OUTPUT);
  pinMode(pedRed, OUTPUT);
  pinMode(pedGreen, OUTPUT);
  //start with green car light on
  digitalWrite(carGreen,HIGH);
  //start with red ped light on
  digitalWrite(pedRed, HIGH);
}
```

Εδώ βλέπουμε την χρήση μιας function στο πρόγραμμά μας για να μας διευκολύνει στον προγραμματισμό και να κάνει πιο κατανοητό τον κώδικα

```
// the loop routine runs over and over again
// forever:
void loop() {
  changeLights();
  delay(15000);
}

void changeLights() {
  //green car light off
  digitalWrite(carGreen,LOW);
  // yellow car light on
  digitalWrite(carYellow,HIGH);
  //wait 2 seconds
  delay(2000);
  // yellow car light off
  digitalWrite(carYellow,LOW);
  //red car light on
  digitalWrite(carRed,HIGH);
  //wait 1 second to turn on ped light
  delay(1000);
  //red ped light off
  digitalWrite(pedRed,LOW);
  //green ped light on. allow crossing
  digitalWrite(pedGreen,HIGH);
  //delay preset time of 5 seconds
  delay(crossTime);
  //flashing of ped green light
  for (int x=0; x<10; x++) {
    digitalWrite(pedGreen,HIGH);
    delay(250);
    digitalWrite(pedGreen,LOW);
    delay(250);
  }

  //turn red ped light on
  digitalWrite(pedRed, HIGH);
  delay(100);
  //car green light on
  digitalWrite(carGreen,HIGH);
  //car red light off
  digitalWrite(carRed,LOW);
}
```

Εργασία

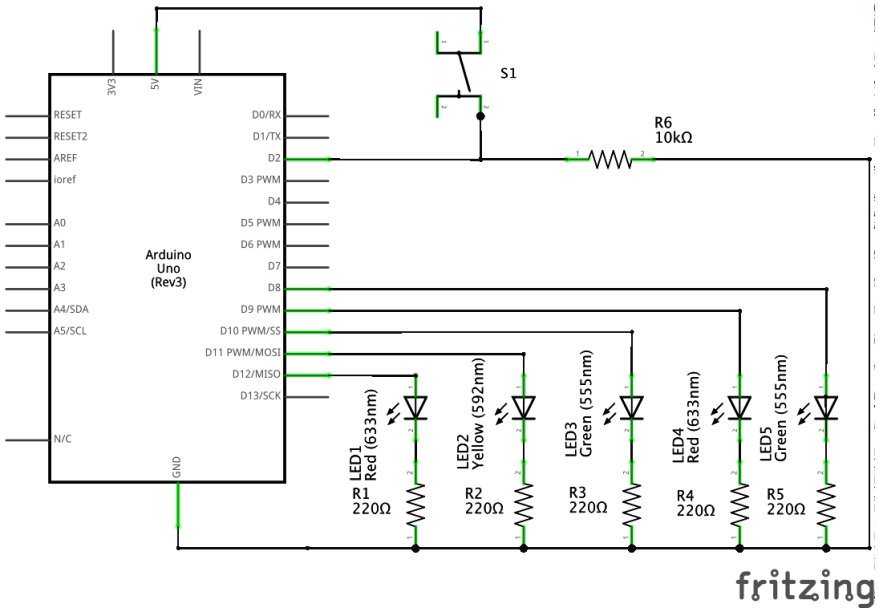
Τροποποίησε την άσκηση ώστε να περιλαμβάνει τα φανάρια για διασταύρωση δύο δρόμων.



04

Φανάρι κυκλοφορίας με κομβίο πεζών

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Εφαρμογή

Με την άσκηση αυτή μάθετε πώς να προγραμματίζετε τον Μικροελεγκτή UNO ώστε να προσομοιώσετε προγραμματιστικά τη λειτουργία ενός φαναριού κυκλοφορίας με μπουτόν για πεζούς.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Πέντε λαμπάκια LED (2 κόκκινα, 2 πράσινα και ένα κίτρινο)
5. Πέντε Αντιστάσεις 220Ω
6. Μια Αντίσταση 10kΩ
7. Ένα διακόπτη μπουτόν
8. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Θα παρατηρήσετε ότι το βασικό κύκλωμα είναι ίδιο με αυτό της προηγούμενης άσκησης, στο οποίο έχει προστεθεί ένας διακόπτης μπουτόν και μία αντίσταση.

Θα παρατηρήσετε ότι και πάλι σε σειρά με κάθε λαμπάκι LED υπάρχει μία αντίσταση, ενώ το άλλο άκρο του κάθε LED συνδέεται σε διαφορετικό ακροδέκτη της πλακέτας του μικροελεγκτή (ακροδέκτες 8 – 12). Οι ακροδέκτες αυτοί είναι προγραμματιζόμενοι και στη συγκεκριμένη περίπτωση έχουν οριστεί σαν έξοδοι. Ο διακόπτης μπουτόν συνδέεται στον ακροδέκτη 2, που προγραμματίζεται σαν είσοδος. Όπως και στην προηγούμενη άσκηση τα λαμπάκια LED έχουν

χωριστεί σε δύο ομάδες. Η πρώτη ομάδα αποτελείται από τρία LED (κόκκινο, κίτρινο, πράσινο) και αντιστοιχεί στο φανάρι για τα αυτοκίνητα και η δεύτερη αποτελείται από δύο LED (κόκκινο και πράσινο) και αντιστοιχεί στο φανάρι των πεζών. Το πρόγραμμα ανάβει διαδοχικά το πράσινο, το κίτρινο και το κόκκινο LED της πρώτης ομάδας για χρονικά διαστήματα που ορίζονται προγραμματιστικά και προσομοιώνει τη λειτουργία του φαναριού για τα αυτοκίνητα. Η προτεραιότητα είναι στην κυκλοφορία των αυτοκινήτων. Αν πιεστεί το μπουτόν των πεζών το φανάρι των αυτοκινήτων ανάβει αρχικά κίτρινο και ακολούθως το κόκκινο λαμπάκι για τα αυτοκίνητα και μετά από λίγο το πράσινο λαμπάκι για τους πεζούς. Μετά από κάποιο χρονικό διάστημα ανάβει το κόκκινο LED των πεζών και μετά από λίγο το πράσινο LED για τα αυτοκίνητα. Για να ληφθεί υπόψιν το πάτημα του μπουτόν των πεζών θα πρέπει να έχει περάσει ένα καθορισμένο χρονικό διάστημα από την προηγούμενη φορά που είχε ενεργοποιηθεί η εναλλαγή της κυκλοφορίας οχημάτων - πεζών. Όλα τα χρονικά διαστήματα και οι καθυστερήσεις που εμπλέκονται στη διαδικασία είναι προγραμματιζόμενα.

Κώδικας

Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

```
/*
Traffic light with a pedestrian button
*/

// constants won't change. They're used here to
// set pin numbers:
// in Pin 12 we will connect the car red LED.
// Give it a name.
int carRed = 12;
// in Pin 11 we will connect the car yellow LED.
// Give it a name.
int carYellow = 11;
// in Pin 10 we will connect the car green LED.
// Give it a name.
int carGreen = 10;
// in Pin 9 we will connect the pedestrian red
// LED. Give it a name.
int pedRed = 9; //ped lights being assigned
```

```
// in Pin 8 we will connect the pedestrian green
// LED. Give it a name.
int pedGreen = 8;
// in Pin 12 we will connect the pedestrian
// button. Give it a name.
int button = 2;
//crossing time given to pedestrians
int crossTime = 5000;

// Variables
// collects the time since the button was last
// pressed
unsigned long changeTime;
// store pedestrian press button
int pedestrian;

// the setup routine runs once when you press
// reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(carRed, OUTPUT);
  pinMode(carYellow, OUTPUT);
  pinMode(carGreen, OUTPUT);
  pinMode(pedRed, OUTPUT);
  pinMode(pedGreen, OUTPUT);
  // initialize the digital pin as an input
  pinMode(button, INPUT);
  // start with green car light on
  digitalWrite(carGreen,HIGH);
  // start with red ped light on
  digitalWrite(pedRed, HIGH);
  // initialize changeTime
  changeTime = millis();
  // initialize preState
  pedestrian = LOW;
}

// the loop routine runs over and over again
// forever:
void loop() {
  // check if button is pressed
  // and it is over 5 sec since last button
  // press
  int state = digitalRead(button);
  if (state == HIGH) {
    pedestrian = HIGH;
  }
```

```
  if (pedestrian == HIGH &&
      (millis() - changeTime) > 15000) {
    //function to change lights
    changeLights();
    changeTime = millis();
    pedestrian = LOW;
  }
}

void changeLights() {
  //green car light off
  digitalWrite(carGreen,LOW);
  // yellow car light on
  digitalWrite(carYellow,HIGH);
  //wait 2 seconds
  delay(2000);
  // yellow car light off
  digitalWrite(carYellow,LOW);
  //red car light on
  digitalWrite(carRed,HIGH);
  //wait 1 second to turn on ped light
  delay(1000);
  //red ped light off
  digitalWrite(pedRed,LOW);
  //green ped light on. allow crossing
  digitalWrite(pedGreen,HIGH);
  //delay preset time of 5 seconds
  delay(crossTime);
  //flashing of ped green light
  for (int x=0; x<10; x++) {
    digitalWrite(pedGreen,HIGH);
    delay(250);
    digitalWrite(pedGreen,LOW);
    delay(250);
  }

  //turn red ped light on
  digitalWrite(pedRed, HIGH);
  delay(100);
  //car green light on
  digitalWrite(carGreen,HIGH);
  //car red light off
  digitalWrite(carRed,LOW);
}

// the loop routine runs over and over again
forever:
```

```
void loop() {
  // check if button is pressed
  // and it is over 5 sec since last button
  // press
  int state = digitalRead(button);
  if (state == HIGH &&
     (millis() - changeTime) > 5000)) {
    //function to change lights
    changeLights();
  }
}
```

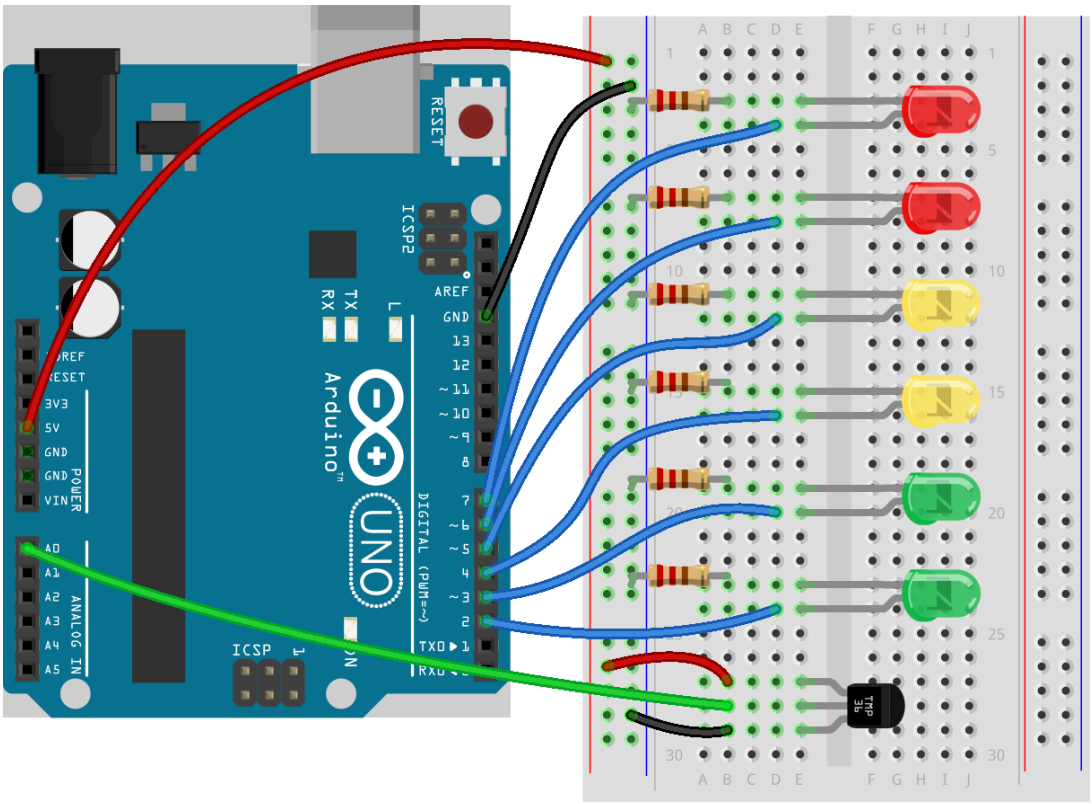
Εδώ βλέπουμε την χρήση μιας function στο πρόγραμμά μας για να μας διευκολύνει στον προγραμματισμό και να κάνει πιο κατανοητό τον κώδικα

```
void changeLights() {
  //green car light off
  digitalWrite(carGreen,LOW);

  // yellow car light on
  digitalWrite(carYellow,HIGH);

  //wait 2 seconds
  delay(2000);

  // yellow car light off
  digitalWrite(carYellow,LOW);

  //red car light on
  digitalWrite(carRed,HIGH);

  //wait 1 second to turn on ped light
  delay(1000);

  //red ped light off
  digitalWrite(pedRed,LOW);

  //green ped light on. allow crossing
  digitalWrite(pedGreen,HIGH);

  //delay preset time of 5 seconds
  delay(crossTime);

  //flashing of ped green light
  for (int x=0; x<10; x++){
    digitalWrite(pedGreen,HIGH);
    delay(250);
    digitalWrite(pedGreen,LOW);
    delay(250);
  }
```

```
  //turn red ped light on
  digitalWrite(pedRed, HIGH);
  delay(100);

  //car green light on
  digitalWrite(carGreen,HIGH);

  //car red light off
  digitalWrite(carRed,LOW);
}
```

Εργασία

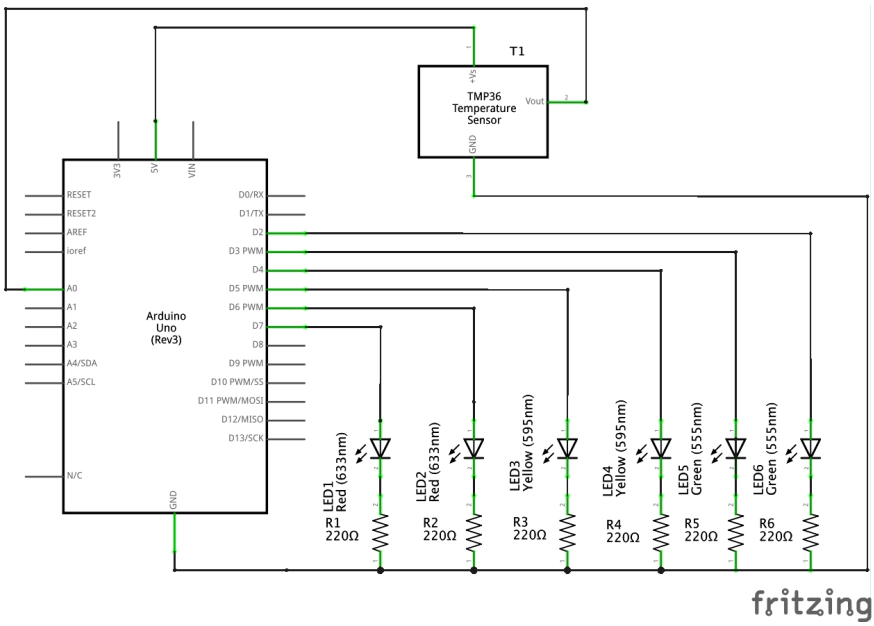
Τροποποιήστε το πρόγραμμα ώστε υπάρχει αυτόματη εναλλαγή μεταξύ αυτοκινήτων - πεζών και το φανάρι των πεζών να έχει προτεραιότητα στη λειτουργία της εναλλαγής μεταξύ αυτοκινήτων / πεζών με βάση κάποιο χρόνο που θα καθορίσετε.



05

Χρωματικό θερμόμετρο

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Εφαρμογή

Με την άσκηση αυτή μάθετε πώς να προγραμματίζετε τον μικροελεγκτή UNO ώστε να κατασκευάσετε ένα χρωματικό θερμόμετρου, παρόμοιο με αυτά που χρησιμοποιούνται σε ψυγεία, π.χ., για παρακολούθηση των θερμοκρασιακών περιοχών λειτουργίας τους.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελλεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Έξι λαμπάκια LED (2 κόκκινα, 2 κίτρινα, 2 πράσινα)
5. Έξι Αντιστάσεις 220Ω
6. Έναν Αισθητήρα θερμοκρασίας
7. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Στο κύκλωμα αυτό χρησιμοποιούνται 6 λαμπάκια LED, το καθένα από τα οποία αντιστοιχεί σε μια περιοχή (ζώνη) θερμοκρασίας. Όταν η θερμοκρασία κυμαίνεται στην χαμηλότερη ζώνη, ανάβει το κάτω πράσινο LED. Αν η θερμοκρασία ανεβεί και πέσει στην αμέσως ψηλότερη ζώνη ανάβουν και τα δύο πράσινα LED. Αν ανεβεί ακόμη περισσότερο και πέσει στην επόμενη ζώνη θα ανάψουν τα δύο πράσινα και το πρώτο κίτρινο LED, και ούτω καθ' εξής.

Τα λαμπάκια LED συνδέονται στις ψηφιακές θύρες του μικροελεγκτή που προγραμματίζονται μέσω του κώδικα σαν έξοδοι. Η ανίχνευση της θερμοκρασίας γίνεται με τον αισθητήρα θερμοκρασίας, το μεσαίο άκρο του οποίου συνδέεται στην είσοδο Α0 του μικροελεγκτή. Μέσω του

κώδικα ορίζονται οι έξι ζώνες θερμοκρασιών και με βάση την τάση που παράγει ο αισθητήρας και μετράει ο μικροελεγκτής ο κώδικας ενεργοποιεί τις ψηφιακές εξόδους ώστε να ανάψουν τα λαμπάκια που πρέπει για κάθε περίπτωση

Κώδικας

Για αυτή την άσκηση θα φτιάξετε το παρακάτω πρόγραμμα και θα το φορτώσετε στο UNO.

```
/*
Display temperature as a color bar
*/

// constants won't change. They're used here to
// set pin numbers:
// in Pin 0 we will connect the temperature
// sensor. Give it a name.
const int tempPin = 0;
// in Pin 2 we will connect the low green
// LED. Give it a name.
const int lowGreenPin = 2;
// in Pin 3 we will connect the hi green
// LED. Give it a name.
const int hiGreenPin = 3;
// in Pin 4 we will connect the low yellow
// LED. Give it a name.
const int lowYellowPin = 4;
// in Pin 5 we will connect the hi yellow
// LED. Give it a name.
const int hiYellowPin = 5;
// in Pin 6 we will connect the low red
// LED. Give it a name.
const int lowRedPin = 6;
// in Pin 7 we will connect the hi red
// LED. Give it a name.
const int hiRedPin = 7;

//VARIABLES
float tempReading;
float tempVolts;
float tempC;

// the setup routine runs once when you press
// reset:
void setup() {
```

Εδώ βλέπουμε την χρήση μιας function στο πρόγραμμά μας για να μας διευκολύνει στον προγραμματισμό και να κάνει πιο κατανοητό τον κώδικα

```
  // initialize the digital pin as an output.
  pinMode(lowGreenPin, OUTPUT);
  pinMode(hiGreenPin, OUTPUT);
  pinMode(lowYellowPin, OUTPUT);
  pinMode(hiYellowPin, OUTPUT);
  pinMode(lowRedPin, OUTPUT);
  pinMode(hiRedPin, OUTPUT);
  // turn off all leds at start
  turnOffLeds();
}

// the loop routine runs over and over again
// forever:
void loop() {
  // read the temperature
  tempReading = analogRead(tempPin);
  // convert reading to C
  tempVolts = tempReading * 5.0 / 1024.0;
  tempC = (tempVolts - 0.5) * 100.0;
  // turn off all leds
  turnOffLeds();
  // compare temperature with color scale
  if (tempC >= 10.00) {
    digitalWrite(lowGreenPin,HIGH);
  }
  if (tempC >= 15.00) {
    digitalWrite(hiGreenPin,HIGH);
  }
  if (tempC >= 20.00) {
    digitalWrite(lowYellowPin,HIGH);
  }
  if (tempC >= 25.00) {
    digitalWrite(hiYellowPin,HIGH);
  }
  if (tempC >= 30.00) {
    digitalWrite(lowRedPin,HIGH);
  }
  if (tempC >= 35.00) {
    digitalWrite(hiRedPin,HIGH);
  }
  delay(99);
}
```

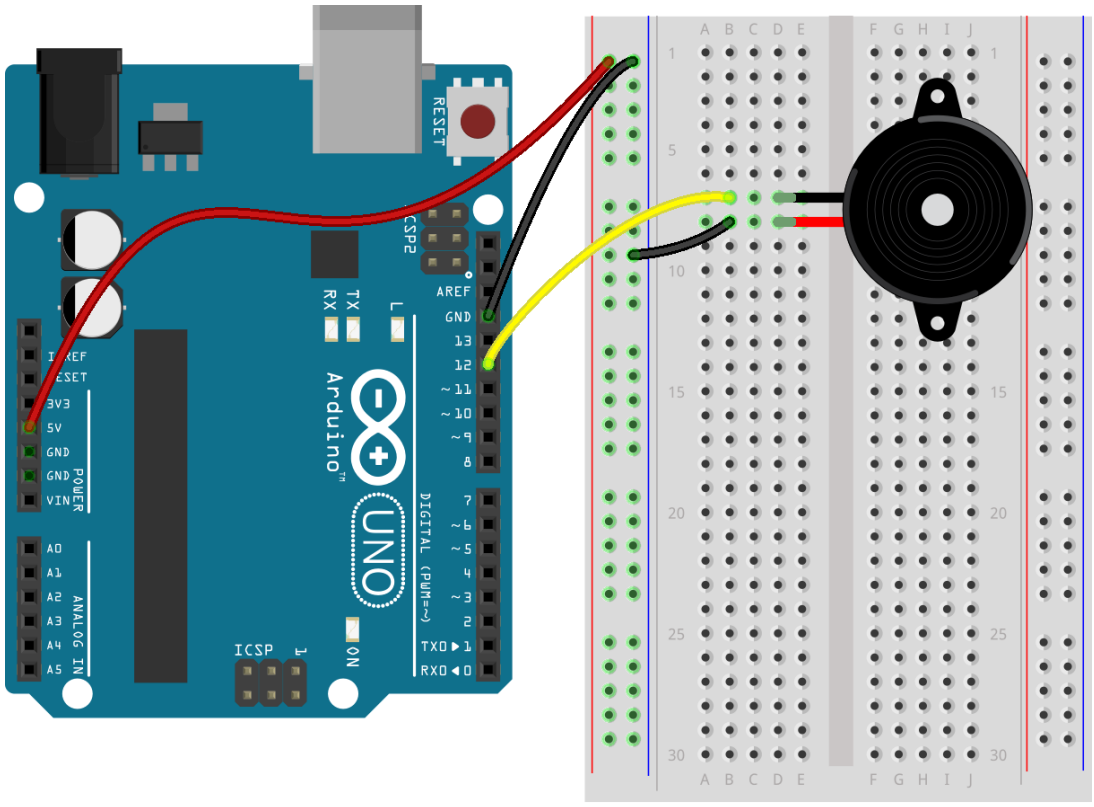
Ορισμός της function turnOffLeds

```
void turnOffLeds() {
  digitalWrite(lowGreenPin, LOW);
  digitalWrite(hiGreenPin, LOW);
  digitalWrite(lowYellowPin, LOW);
```

```
  digitalWrite(hiYellowPin, LOW);
  digitalWrite(lowRedPin, LOW);
  digitalWrite(hiRedPin, LOW);
}
```

Εργασία

Τροποποιήστε το πρόγραμμα ώστε όταν η θερμοκρασία είναι κάτω από την ελάχιστη που ελέγχουμε στο πρόγραμμα να αναβοσβήνουν τα δύο πράσινα LED και τα υπόλοιπα να είναι σβηστά.

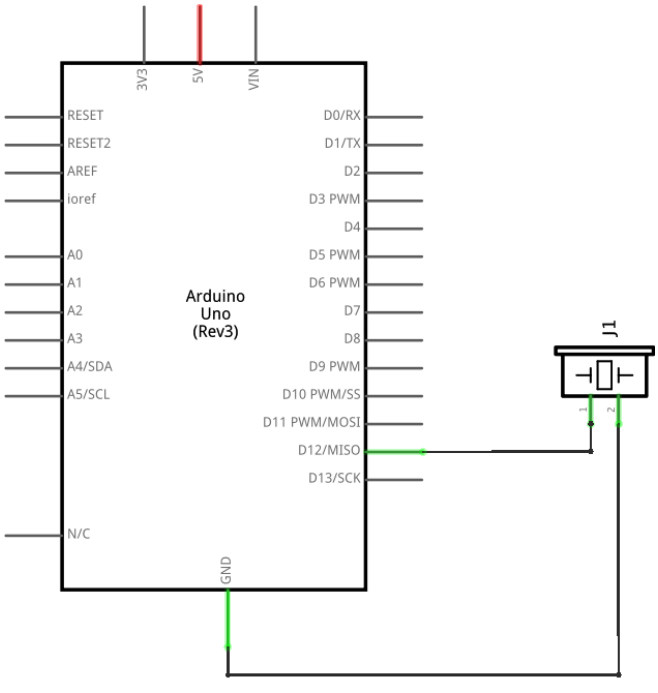


fritzing

06

Μουσικοί ήχοι

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



fritzing

Εφαρμογή

Με την άσκηση αυτή μάθετε πώς να προγραμματίζετε τον μικροελεγκτή UNO ώστε να παράγετε μουσικούς ήχους χρησιμοποιώντας τον πιεζοηλεκτρικό βομβητή.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Τον πιεζοηλεκτρικό βομβητή
5. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Όπως φαίνεται από τη συνδεσμολογία, ο πιεζοηλεκτρικός βομβητής οδηγείται από τη την ψηφιακή θύρα 12 του μικροελεγκτή που ορίζεται μέσω του προγράμματος σαν έξοδος. Το πρόγραμμα περιλαμβάνει εντολές οι οποίες ορίζουν τον αριθμό των τόνων που θα παραχθούν, τις συχνότητές τους και τη διάρκειά τους.

Κώδικας

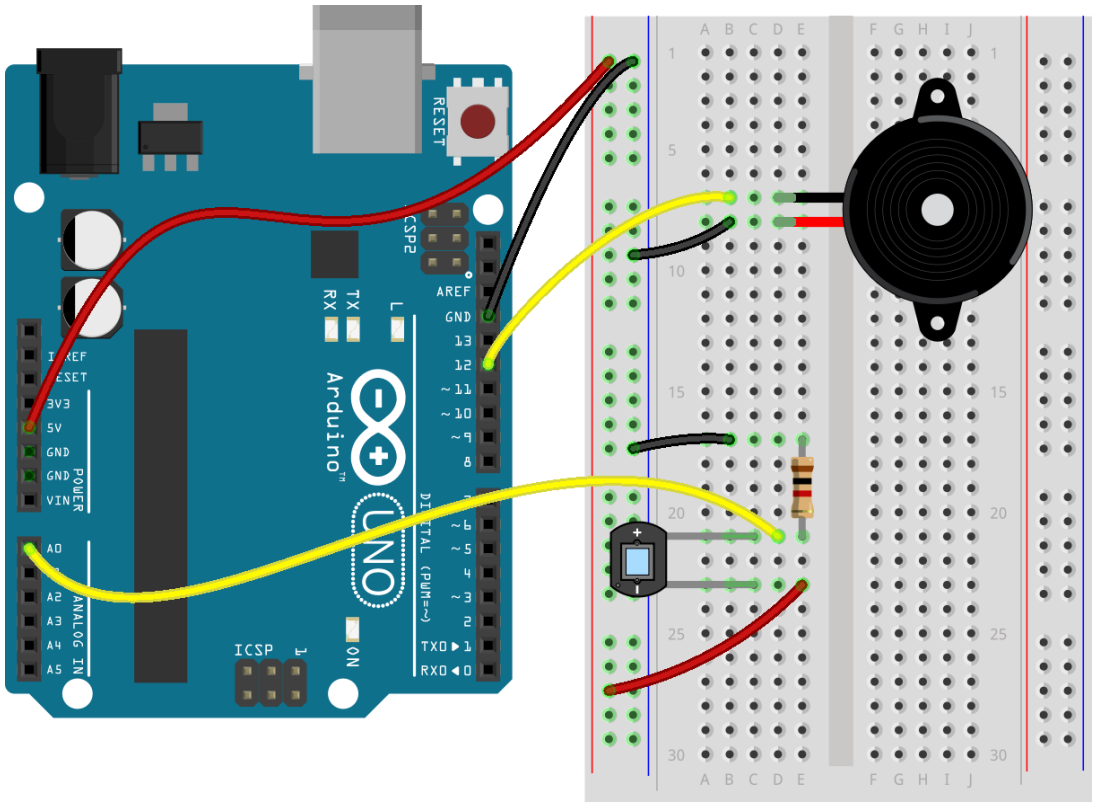
Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

```
/*
Make simple sounds
*/

// constants won't change. They're used here to
// set pin numbers:
// in Pin 12 we will connect the piezo speaker.
```

```
// Give it a name.
const int speakerPin = 12;
// the number of tones to play
const int numTones = 10;
// the tones to play
// mid C C# D D# E F F# G G# A
int tones[] = {261, 277, 294, 311, 330, 349, 370,
392, 415, 440};

// the setup routine runs once when you press
// reset:
void setup() {
  for (int i = 0; i < numTones; i++) {
    tone(speakerPin, tones[i]);
    delay(500);
  }

  noTone(speakerPin);
}

// the loop routine runs over and over again
// forever:
void loop() {
}
```

Εργασία

Τροποποιήστε το πρόγραμμα ώστε ο βομβητής να παράγει με ήχους το σήμα SOS σε κώδικα Μορς, με παύση ενός λεπτού μεταξύ δύο διαδοχικών σημάτων SOS.

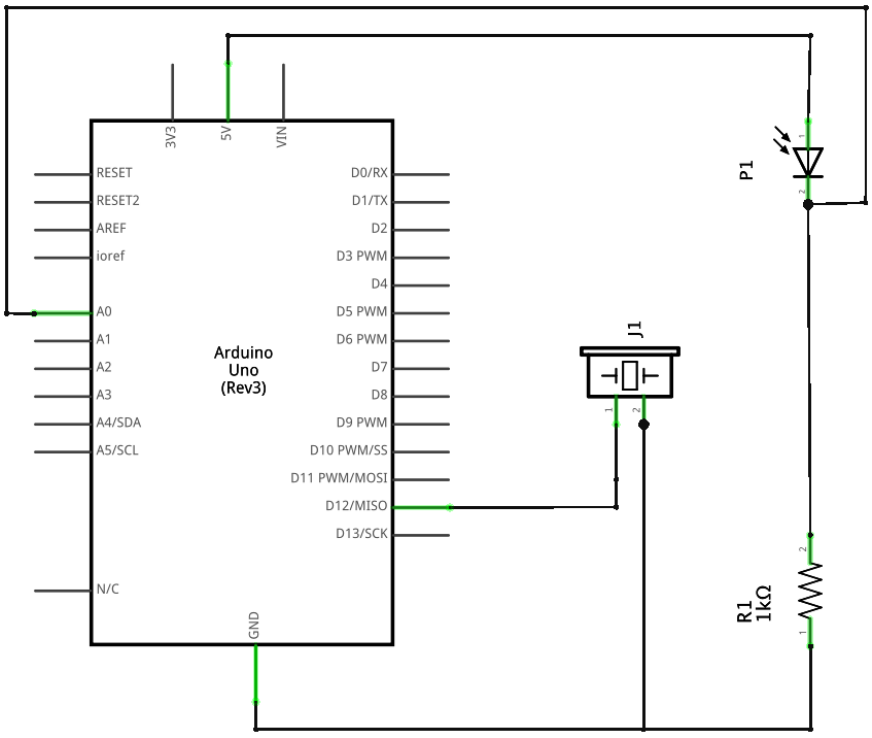


fritzing

07

Μουσικό όργανο Theremin

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



fritzing

Σκοπός

Με την άσκηση αυτή μάθετε πώς να παράγετε μουσικούς ήχους προγραμματιστικά όπως συμβαίνει με το μουσικό όργανο Theremin.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Τον πιεζοηλεκτρικό βομβητή
5. Μια Αντίσατση 1kΩ
6. Μια Φωτοδίοδο
7. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Όπως φαίνεται από τη συνδεσμολογία, ο πιεζοηλεκτρικός βομβητής οδηγείται από τη την ψηφιακή θύρα 12 του μικροελεγκτή που ορίζεται μέσω του προγράμματος σαν έξοδος. Στην προηγούμενη άσκηση οι μουσικοί τόνοι παράγονταν από τον ίδιο τον μικροελεγκτή βάσει του προγραμματισμού του. Στην παρούσα άσκηση, οι τόνοι που παράγονται από τον μικροελεγκτή καθορίζονται από το σήμα που δέχεται στην αναλογική είσοδο Α0. Το σήμα αυτό παράγεται από το κυκλωματάκι που αποτελείται από μία φωτοδίοδο και μία αντίσταση 1ΚΩ σε σειρά. Το κύκλωμα αυτό τροφοδοτείται με τάση 5V που την παίρνουμε από την αντίστοιχη έξοδο του μικροελεγκτή και στο σημείο σύνδεσης αντίστασης και φωτοδιόδου δημιουργείται μία τάση που εξαρτάται από την φωτεινότητα του περιβάλλοντος. Σκιάζοντας ή φωτίζοντας προοδευτικά τη φωτοδίοδο,

μεταβάλλεται η συχνότητα του τόνου που παράγει ο πιεζοηλεκτρικός βομβητής.

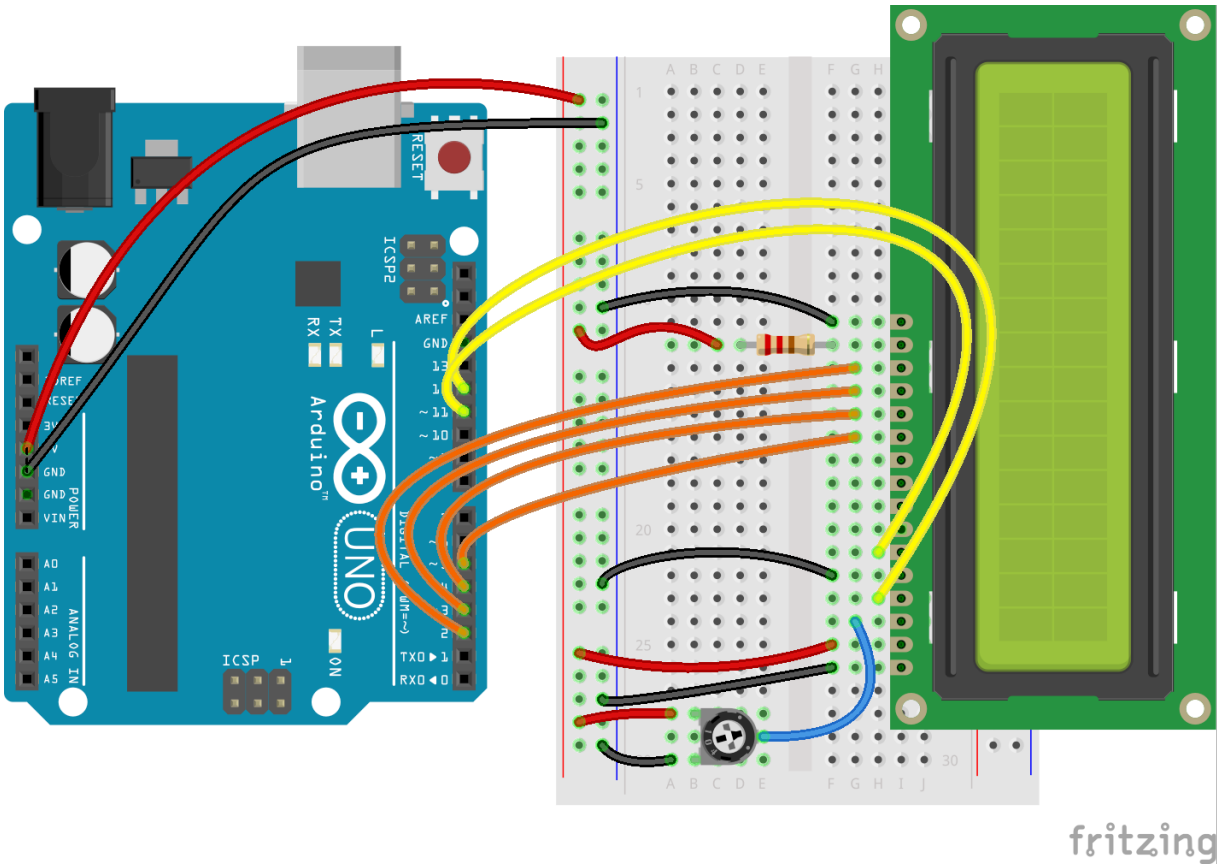
Κώδικας

Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

```
/*
Theremin like music organ to play sounds it
will change the pitch of the note as you wave
your hand in front of it
*/

// constants won't change. They're used here to
// set pin numbers:
// in Pin 12 we will connect the piezo speaker.
// Give it a name.
const int speakerPin = 12;
// in Pin 0 we will connect the photocell.
// Give it a name.
const int photocellPin = 0;

// the setup routine runs once when you press
// reset:
void setup() {
}

// the loop routine runs over and over again
// forever:
void loop() {
  int reading = analogRead(photocellPin);
  int pitch = 200 + reading / 4;
  tone(speakerPin, pitch);
}
```

Εργασία

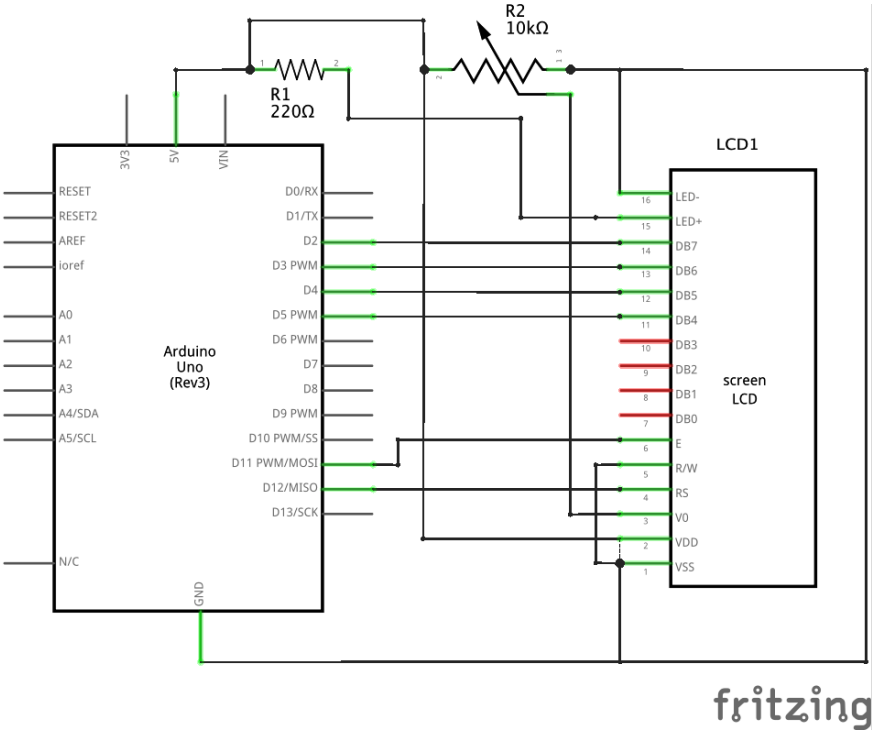
Τροποποιήστε την συνδεσμολογία και το πρόγραμμα ώστε ο βομβητής να παράγει συχνότητες που εξαρτώνται όχι μόνον από τη φωτεινότητα αλλά και από τη θερμοκρασία του περιβάλλοντος.



08

Οθόνη μηνυμάτων

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Σκοπός

Με την άσκηση αυτή θα μάθετε πώς να εμφανίζετε μηνύματα / ανακοινώσεις προγραμματιστικά σε μία οθόνη υγρών κρυστάλλων (LCD Display).

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. ΄Ενας μικροελεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Μια οθόνη χαρακτήρων LCD
5. Μία Αντίσταση 220Ω
6. Μία μεταβλητή Αντίσταση (ποτενσιόμετρο) 10kΩ
7. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Η οθόνη υγρών κρυστάλλων (LCD Display) χρησιμοποιείται για να εμφανίζει αλφαριθμητκούς χαρακτήρες. Η οθόνη του σετ σας μπορεί να εμφανίσει δύο σειρές των 16 χαρακτήρων η κάθε μία, δηλαδή μπορεί να εμφανίσει 32 χαρακτήρες συνολικά. Η οθόνη διαθέτει μία σειρά από εισόδους για την τροφοδοσία της και για την επικοινωνία της με τον μικροελεγκτή UNO. Σ' αυτήν την άσκηση, χρησιμοποιούνται οι είσοδοι που φαίνονται στο σχηματικό διάγραμμα. Η οθόνη παίρνει την τροφοδοσία της από την έξοδο των 5V του μικροελεγκτή και η φωτεινότητά της ρυθμίζεται με το ποτενσιόμετρο των 10KΩ.

Η οθόνη διαθέτει στο κάτω μέρος της 16 ποδαράκια (pins), η αρίθμηση των οποίων αρχίζει από αριστερά προς τα δεξιά. Στο κάθε ποδαράκι αντιστοιχεί μία επισήμανση η οποία προσδιορίζει τη λειτουργία του. Η αντιστοίχιση αυτή

εμφανίζεται στο πίσω μέρος της οθόνης σε μορφή πίνακα. Από την συνδεσμολογία και το ηλεκτρονικό σχέδιο θα παρατηρήσετε ότι η τάση τροφοδοσίας της οθόνης εφαρμόζεται στο ποδαράκι 2 (VDD) και η γείωση του κυκλώματος (GND) είναι το ποδαράκι 1 (VSS). Στο ποδαράκι 3 (Vo) οδηγείται μία τάση από τη μεσαία λήψη του ποτενσιόμετρου που ελέγχει τη φωτεινότητα της οθόνης. Το σήμα που πηγαίνει στο ποδαράκι 4 (RS) καθορίζει σε ποιο σημείο της οθόνης θα εμφανιστούν χαρακτήρες και από το ποδαράκι 5 (R/W) καθορίζεται αν η οθόνη θα διαβάζει ή αν θα γράφει χαρακτήρες. Στο ποδαράκι 6 (EN) μεταβιβάζεται η εντολή που ειδοποιεί την οθόνη ότι θα λάβει κάποια εντολή. Τα ποδαράκια 7 – 10 δεν χρησιμοποιούνται σ' αυτήν την εφαρμογή, ενώ στα ποδαράκια 11 – 14 μεταφέρονται οι πληροφορίες που στέλνει ο μικροελεγκτής, με το κείμενο που θα εμφανίσει η οθόνη. Τέλος, στα ποδαράκια 15 και 16 εφαρμόζεται η τάση τροφοδοσίας των χαρακτήρων.

Με τον κώδικα που παρατίθεται πιο κάτω, ο μικροελεγκτής θα εμφανίσει στην οθόνη το μήνυμα "Hello World".

Κώδικας

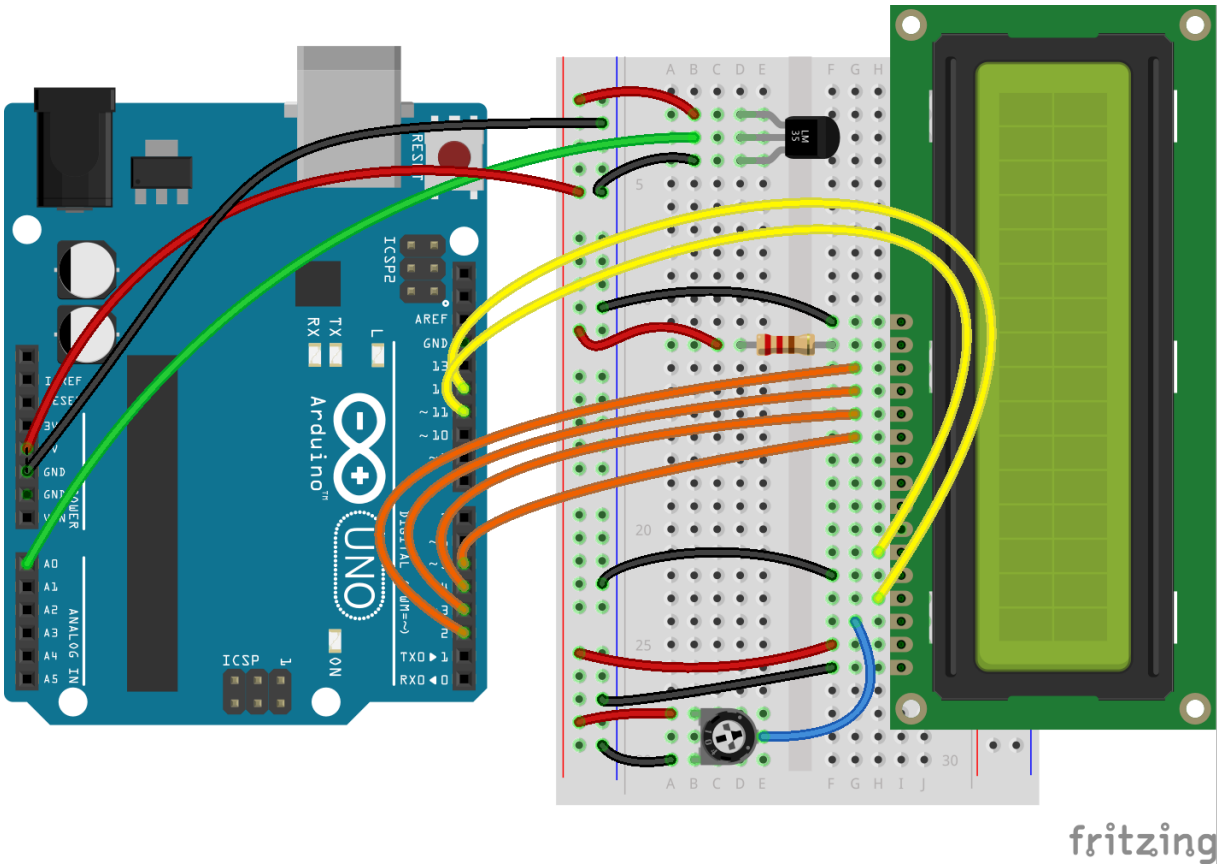
Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

```
/*
Display messages in an LCD display
*/

// include LCD library headers
#include <LiquidCrystal.h>
// setup LCD
// RS E D4 D5 D6 D7
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// the setup routine runs once when you press
// reset:
void setup() {
  // initialize the LCD
  lcd.begin(16, 2);
  // print message
  lcd.print("hello, World!");
}
```

```
// the loop routine runs over and over again
// forever:
void loop() {
  // print the number of milliseconds since the
  // Arduino was reset
  lcd.setCursor(0, 1);
  lcd.print(millis()/1000);
}
```

Εργασία

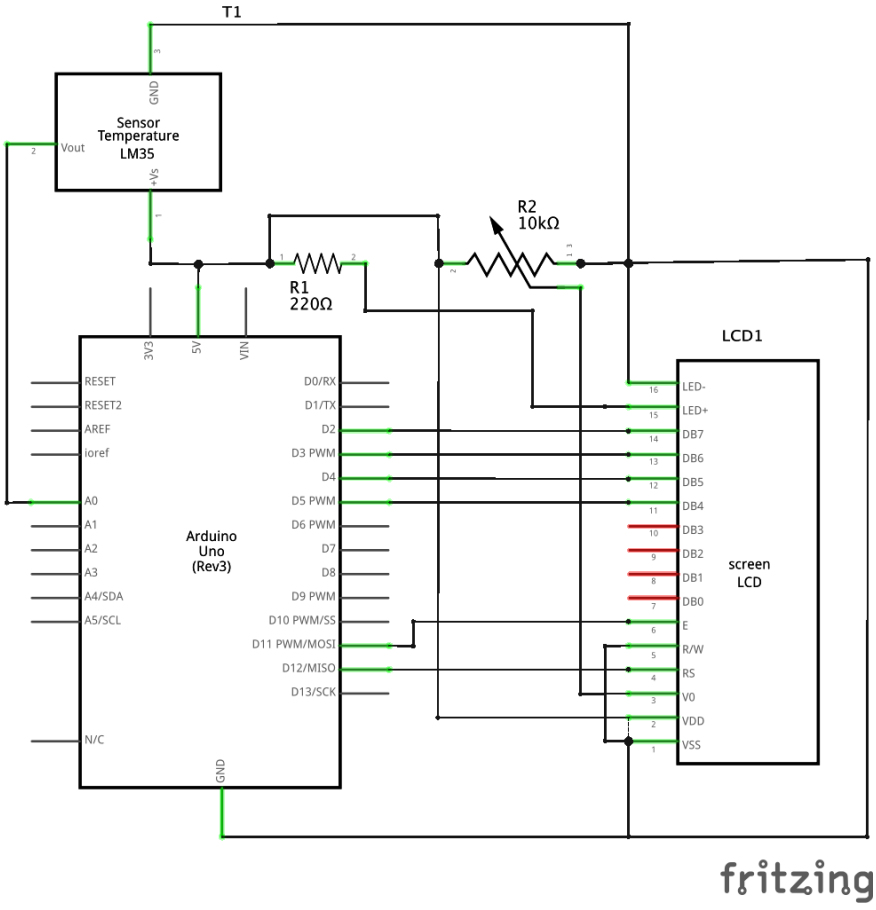
Τροποποιήστε το πρόγραμμα ώστε να εμφανίσετε το μήνυμα στην οθόνη με κύλιση από αριστερά προς τα δεξιά, με πλήρη σάρωση της οθόνης.



09

Ψηφιακό θερμόμετρο χώρου

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Εφαρμογή

Με την άσκηση αυτή θα μάθετε πώς να εμφανίζετε τη θερμοκρασία του περιβάλλοντος προγραμματιστικά σε μία οθόνη υγρών κρυστάλλων (LCD Display).

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Μια οθόνη χαρακτήρων LCD
5. Μία Αντίσταση 220Ω
6. Μία μεταβλητή Αντίσταση (ποτενσιόμετρο) 10kΩ
7. Έναν αισθητήρα θερμοκρασίας
8. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Θα παρατηρήσετε ότι η συνδεσμολογία της οθόνης είναι όπως στην προηγούμενη άσκηση. Γι' αυτή την εφαρμογή έχει προστεθεί και ο αισθητήρας θερμοκρασίας που χρησιμοποιήσατε στην άσκηση 5. Η έξοδος του αισθητήρα θερμοκρασίας (μεσαίο ποδαράκι) συνδέεται στην αναλογική είσοδο Α0 του μικροελεγκτή. Η τάση που παράγει ο αισθητήρας, η οποία είναι ανάλογη της θερμοκρασίας του περιβάλλοντος, μεταφράζεται με κατάλληλο αλγόριθμο του προγράμματος σε αριθμητικούς χαρακτήρες που εμφανίζονται στην οθόνη.

Κώδικας

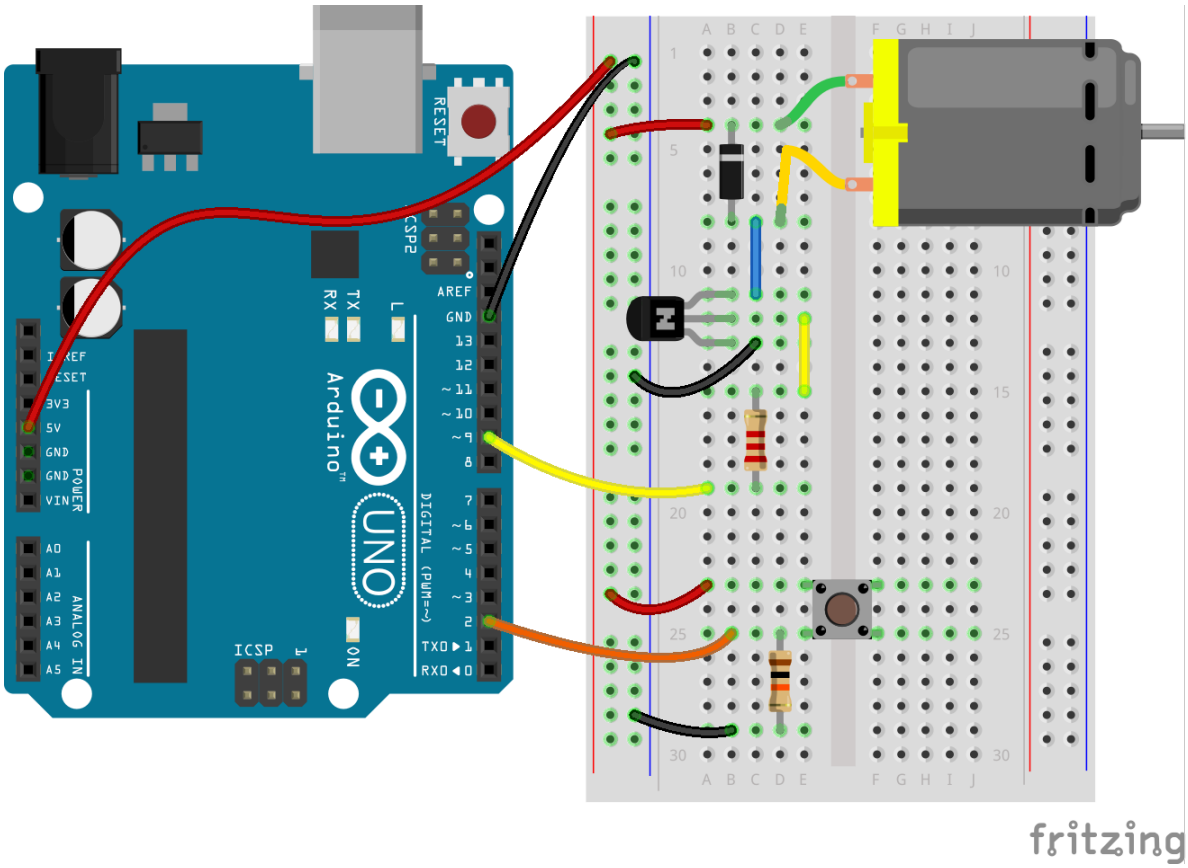
Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

```
/*
Display temperature in an LCD display
in celsius
*/

// include LCD library headers
#include <LiquidCrystal.h>
// constants won't change. They're used here to
// set pin numbers:
// in Pin 0 we will connect the temperature
// sensor. Give it a name.
const int tempPin = 0;
// variables will change:
// variable for reading the temperature from
// sensor
int tempReading;
// variables to convert sensor reading to C
float tempVolts, tempC;
// setup LCD
// RS E D4 D5 D6 D7
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

// the setup routine runs once when you press
// reset:
void setup() {
  // initialize the LCD
  lcd.begin(16, 2);
  lcd.setCursor(0, 1);
  lcd.print("Room condition");
  lcd.setCursor(0, 0);
  lcd.print("Temp C: ");
}

// the loop routine runs over and over again
// forever:
void loop() {
  // read the temperature
  tempReading = analogRead(tempPin);
  // convert reading to C
  tempVolts = tempReading * 5.0 / 1024.0;
  tempC = (tempVolts - 0.5) * 100.0;
  // print it to LCD
```

```
  lcd.setCursor(8, 0);
  lcd.print(tempC);
  delay(500);
}
```

Εργασία

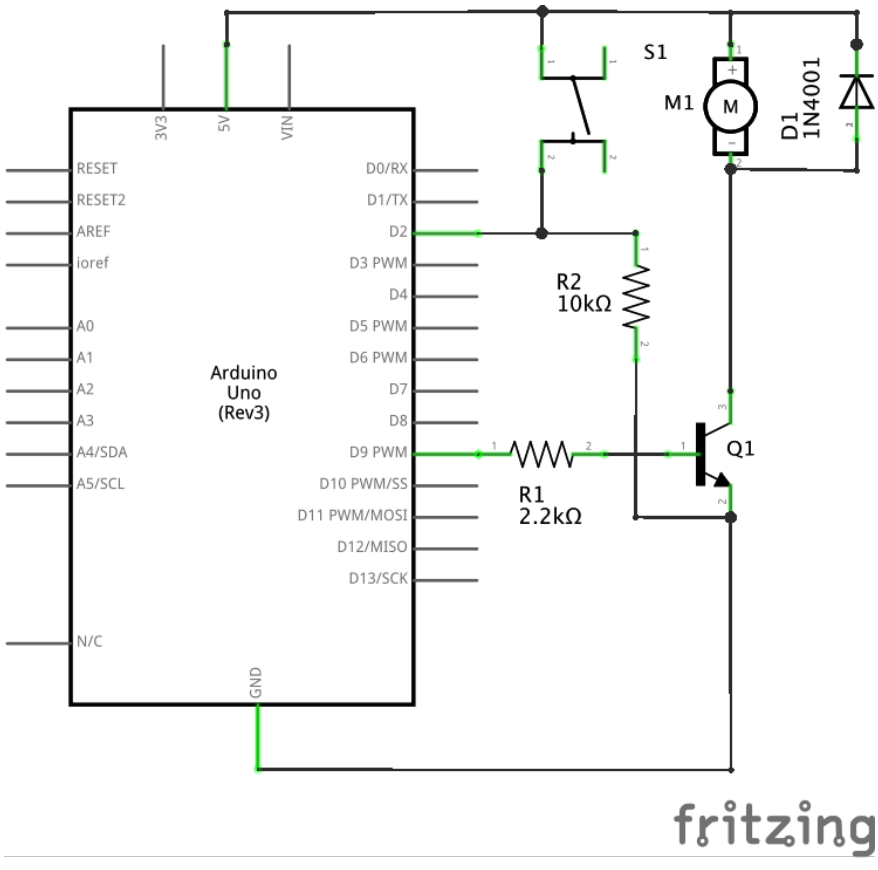
Τροποποιήστε το κύκλωμα και το πρόγραμμα ώστε με τη χρήση μιας φωτοαντίστασης να εμφανίσετε στη δεύτερη γραμμή της οθόνης τη φωτεινότητα του χώρου.



IO

Ανεμιστήρας

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Εφαρμογή

Με την άσκηση αυτή θα μάθετε πώς να κάνετε έναν ανεμιστήρα συνεχούς ρεύματος (DC Motor) να περιστρέφεται όταν πιέζετε έναν διακόπτη πίεσης (μπουτόν). Όσο κρατάτε τον διακόπτη πιεσμένο, ο κινητήρας θα περιστρέφεται. Μόλις ελευθερώσετε τον διακόπτη ο κινητήρας σταματάει.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Ένας κινητήρας DC
5. Ένα διακόπτη πίεσης
6. Μία Αντίσταση 2,2 ΚΩ
7. Μία Αντίσταση 10 ΚΩ
8. Μία Δίοδο
9. Ένα Τρανζίστορ
10. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

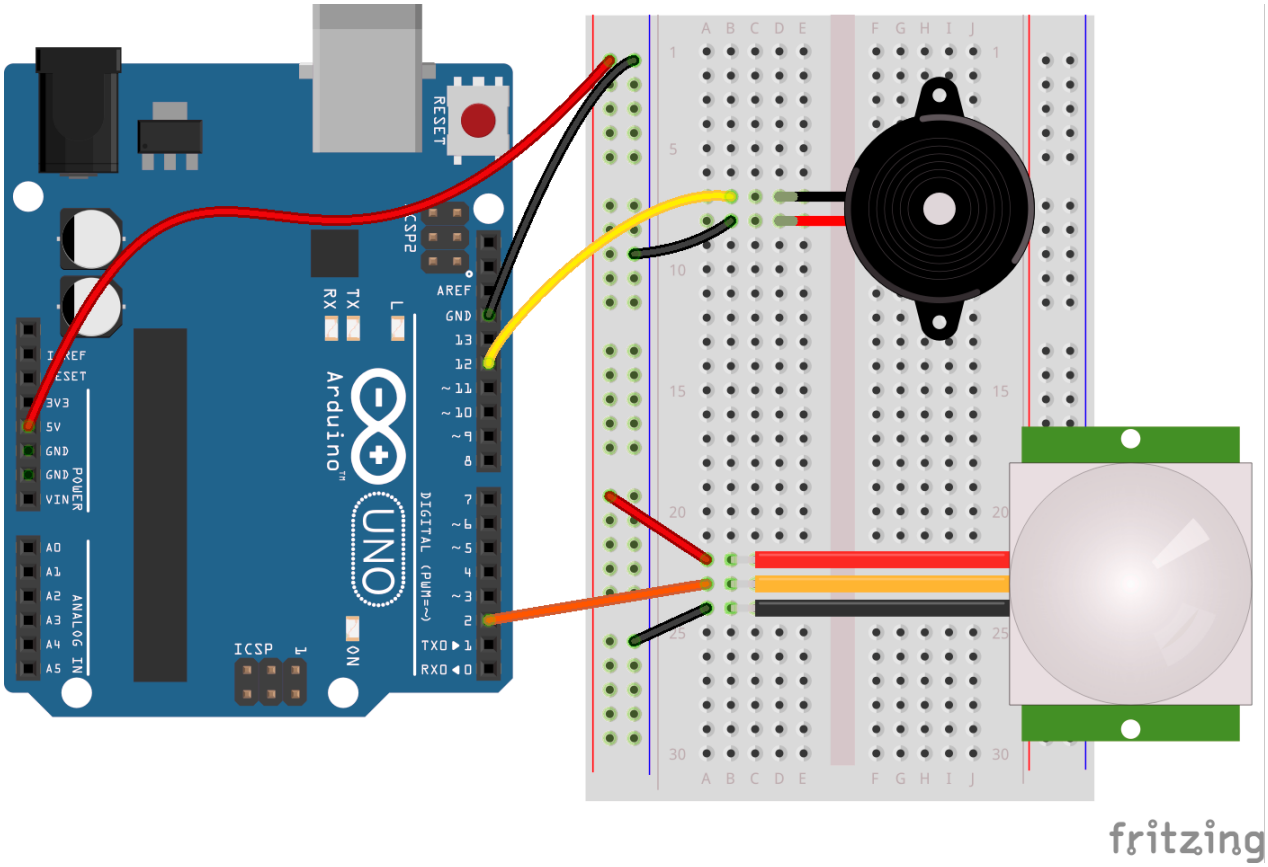
Κώδικας

Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

```
/*
Make the motor spin while the pushbutton is
pressed and stop it when the pushbutton
is released.
*/

// constants won't change. They're used here to
// set pin numbers:
// in Pin 9 we will connect the motor control
// signal. Give it a name.
const int motorPin = 9;
// in Pin 2 we will connect the button control
// signal. Give it a name.
const int buttonPin = 2;
// variables will change:
// variable for reading the pushbutton status
int buttonState = 0;

// the setup routine runs once when you press
// reset:
void setup() {
  // initialize the motor pin as an output
  pinMode(motorPin, OUTPUT);
  // initialize the pushbutton pin as an input
  pinMode(buttonPin, INPUT);
}

// the loop routine runs over and over again
// forever:
void loop() {
  // read the state of the pushbutton value
  buttonState = digitalRead(buttonPin);
  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH.
  if (buttonState == HIGH) {
    // spin motor
    digitalWrite(motorPin, HIGH);
  } else {
    // stop motor
    digitalWrite(motorPin, LOW);
  }
}
```

Εργασία

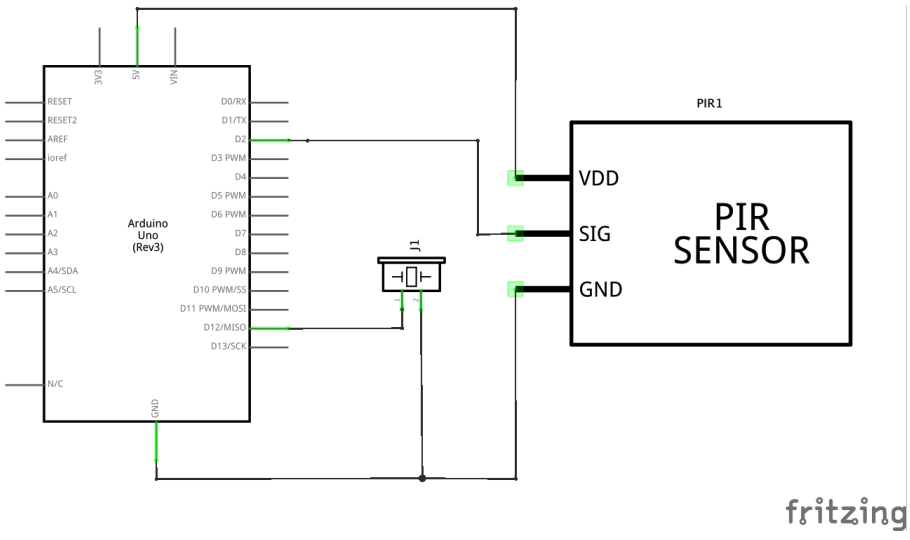
Τροποποίησε την άσκηση ώστε πατώντας μια φορά το κομβίο ο κινητήρας να αρχίζει να περιστρέφεται και μετά την απελευθέρωση του κομβίου. Για να σταματήσει ο κινητήρας θα πρέπει να πιέσουμε ξανά το κομβίο.



II

Συναγερμός

Ηλεκτρικό διάγραμμα άσκησης και συνδεσμολογία των υλικών στο breadboard στην πάνω εικόνα



Εφαρμογή

Με την άσκηση αυτή θα μάθετε πώς να φτιάξετε ένα σύστημα συναγερμού που όταν ανιχνεύει κίνηση σε ένα χώρο θέτει σε λειτουργία ηχητική ειδοποίηση.

Υλικά

Για την εκτέλεση της άσκησης χρειάζονται τα παρακάτω υλικά:

1. Ένας μικροελεγκτής UNO
2. Το καλώδιο διασύνδεσης του μικροελεγκτή UNO με τον Η/Υ
3. Την πλακέτα κυκλωμάτων (Breadboard)
4. Ένας Ανιχνευτής κίνησης PIR
5. Ένα πιεζοηλεκτρικό βομβητή
6. Καλώδια συνδεσμολογίας

Φτιάξε το κύκλωμα

Συναρμολόγησε το κύκλωμα συνδέοντας τα παραπάνω υλικά σύμφωνα με το σχέδιο.

Ο αισθητήρας κίνησης έχει τρία ποδαράκια. Τα δύο ακραία χρησιμοποιούνται για την τροφοδοσία του αισθητήρα (με πολικότητα) και το μεσαίο παρέχει το σήμα εξόδου που οδηγείται στην ψηφιακή θύρα 2 που ορίζεται σαν είσοδος. Το πρόγραμμα διαβάζει συνεχώς την είσοδο 2 και αν διαπιστώσει αλλαγή δίνει εντολή στην ψηφιακή θύρα 12, που έχει οριστεί σαν έξοδος, να δημιουργήσει το σήμα οδήγησης του πιεζοηλεκτρικού βομβητή.

Κώδικας

Για αυτή την άσκηση θα φτιάξεις το παρακάτω πρόγραμμα και θα το φορτώσεις στο UNO.

```
/*
PIR sensor alarm
*/
```

```
// constants won't change. They're used here to
// set pin numbers:
// in Pin 2 we will connect the PIR sensor
// signal. Give it a name.
const int pirPin = 2;
// in Pin 12 we will connect the piezoelectric
// speaker. Give it a name.
const int speakerPin = 12;
// variables will change:
// variable for reading the PIR status
// we start, assuming no motion detected
int pirState = LOW;

// the setup routine runs once when you press
// reset:
void setup() {
  // initialize the speaker pin as an output
  pinMode(speakerPin, OUTPUT);
  // initialize the PIR pin as an input
  pinMode(pirPin, INPUT);
}

// the loop routine runs over and over again
// forever:
void loop() {
  // read the state of the PIR sensor
  pirState = digitalRead(pirPin);
  // check if the PIR detected motion.
  // if it is, the pinState is HIGH.
  if (pirState == HIGH) {
    // play siren tones
    noTone(speakerPin);
    tone(speakerPin, 40, 200);
    tone(speakerPin, 494, 500);
    tone(speakerPin, 53, 300);
  } else {
    // stop speaker
    noTone(speakerPin);
  }
}
```

Εργασία

Τροποποιήστε το κύκλωμα με την προσθήκη ενός κόκκινου LED που θα αναβοσβήνει σαν φάρος και ο πιεζοηλεκτρικός βομβητής θα ηχεί ταυτόχρονα όταν έχουμε ανίχνευση κίνησης.