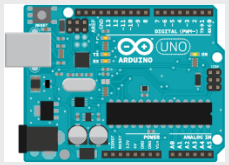
ΕΛΛΗΝΙΚΟ ΜΕΣΟΓΕΙΑΚΟ ΠΑΝΕΠΙΣΤΗΜΙΟ
ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ
Τμήμα Μηχανολόγων Μηχανικών

ΜΗΧΑΤΡΟΝΙΚΑ ΣΥΣΤΗΜΑΤΑ I & II

«Ο Μικροελεγκτής Arduino»

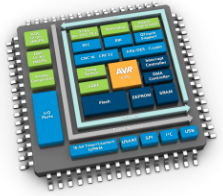
Δρ. Φασουλάς Ιωάννης
jfasoulas@hmu.gr



Εργαστήριο Συστημάτων Ελέγχου & Ρομποτικής (CSRL)
Σχολή Μηχανικών
Ελληνικό Μεσογειακό Πανεπιστήμιο
Ηράκλειο Κρήτης, Ελλάς

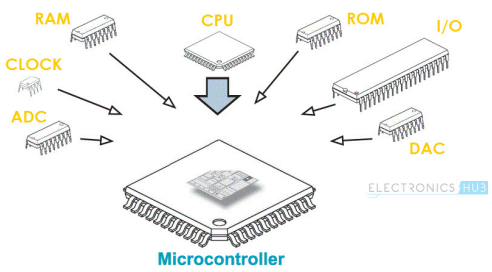
Γενικές πληροφορίες για τους μικροελεγκτές

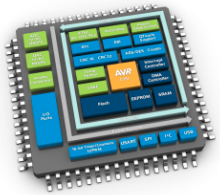
Ο μικροελεγκτής (αγγλικά, microcontroller) μπορεί να λειτουργήσει με ελάχιστα εξωτερικά εξαρτήματα, λόγω των πολλών ενσωματωμένων υποσυστημάτων που διαθέτει. Χρησιμοποιείται ευρύτατα σε όλα τα ενσωματωμένα συστήματα (embedded systems) ελέγχου χαμηλού και μεσαίου κόστους, όπως αυτά που χρησιμοποιούνται σε αυτοματισμούς, ηλεκτρονικά καταναλωτικά προϊόντα (από ψηφιακές φωτογραφικές μηχανές έως παιχνίδια), ηλεκτρικές συσκευές και κάθε είδους αυτοκινούμενα τροχοφόρα οχήματα.

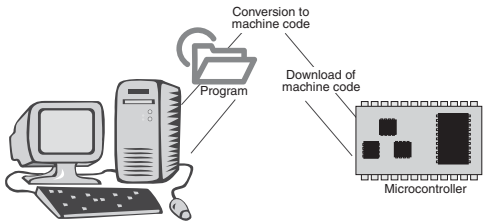


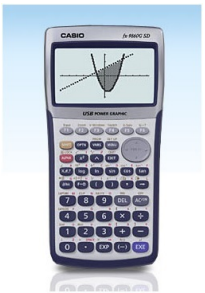
Το Arduino είναι μια υπολογιστική πλατφόρμα βασισμένη σε μια απλή μητρική πλακέτα με ενσωματωμένο μικροελεγκτή και εισόδους/εξόδους, και η οποία μπορεί να προγραμματιστεί με τη γλώσσα Wiring (ουσιαστικά πρόκειται για τη C++ με κάποιες μετατροπές). Το Arduino μπορεί να χρησιμοποιηθεί για την ανάπτυξη ανεξάρτητων διαδραστικών αντικειμένων αλλά και να συνδεθεί με υπολογιστή μέσω προγραμμάτων σε Processing, Max/MSP, Pure Data, SuperCollider.

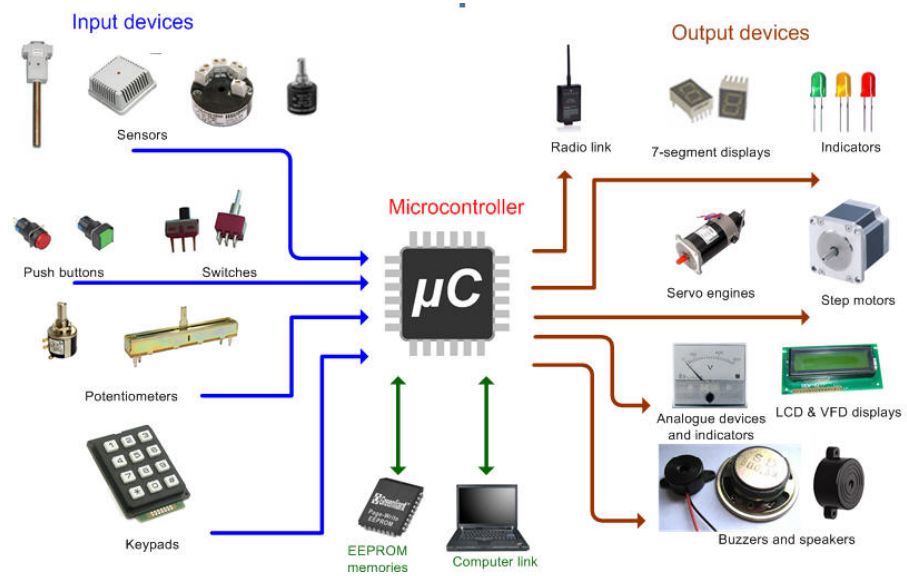
Γενικές πληροφορίες για τους μικροελεγκτές



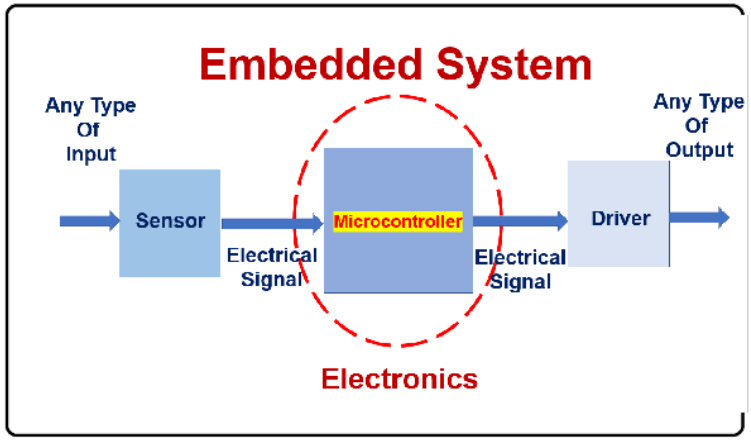




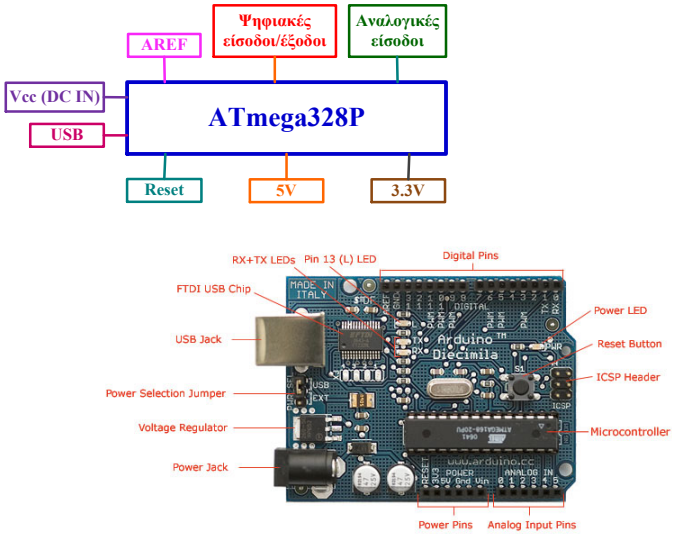




Μικροελεγκτής και ενσωματωμένα συστήματα

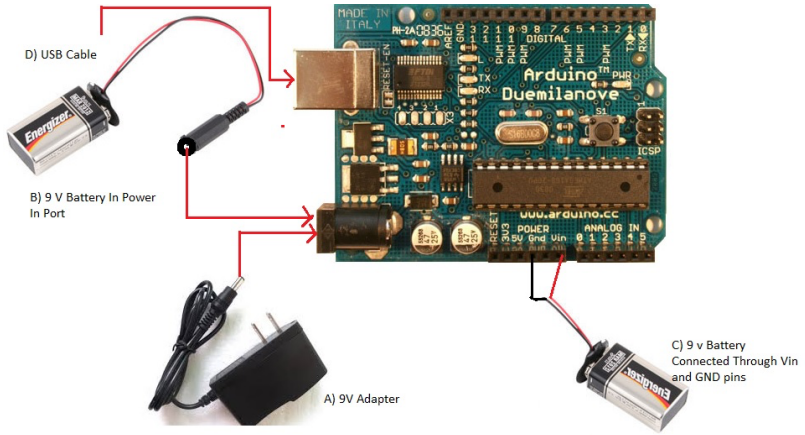


Η πλατφόρμα ARDUINO



Διάφοροι τρόποι τροφοδοσία του μικροελεγκτή

- Vcc (DC IN) Εξωτερική τροφοδοσία για αυτόνομη λειτουργία χωρίς υπολογιστή. Η τροφοδοσία αυτή μπορεί να προέρχεται από τροφοδοτικό, μπαταρία ή άλλο παρόμοιο στοιχείο.



Η πλατφόρμα ARDUINO

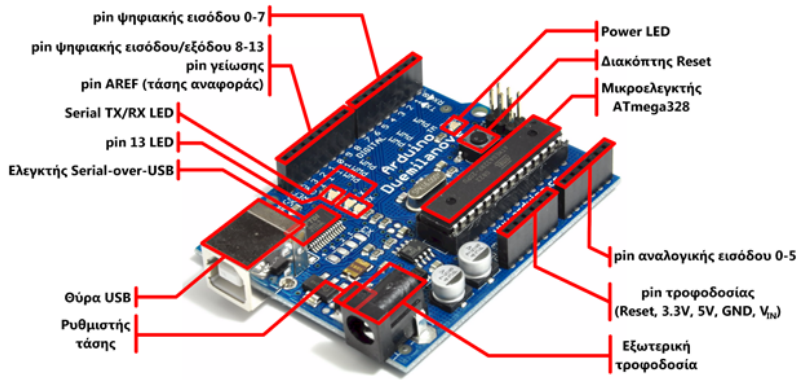
Αποτελείται από ένα μικροελεγκτή **Atmel AVR**
(ATmega328 και ATmega168 στις νεότερες εκδόσεις, ATmega8 στις παλαιότερες)
Όλες οι πλακέτες περιλαμβάνουν:

- ένα σταθεροποιητή τάσης 5V
- ένα κρυσταλλικό ταλαντωτή 16MHz

Ο προγραμματισμός γίνεται μέσω της σειριακής θύρας **RS-232**. Όμως, οι πλακέτες Arduino που κυκλοφορούν σήμερα στην αγορά, συμπεριλαμβανόμενης και της Diecimila, προγραμματίζονται μέσω USB, χρησιμοποιώντας ένα τσιπ προσαρμογέα USB-to-serial όπως το FTDI FT232

Η γλώσσα προγραμματισμού που χρησιμοποιείται ονομάζεται **Wiring C**, μια γλώσσα που **μοιάζει με την C** και παρέχει παρόμοια λειτουργικότητα

Ανάλυση της πλατφόρμας ARDUINO



Σύντομη περιγραφή της γλώσσας Wiring

Κάθε πρόγραμμα στον Arduino ονομάζεται ***Sketch***

ΒΑΣΙΚΗ ΔΟΜΗ ΤΟΥ ΠΡΟΓΡΑΜΜΑΤΟΣ
Κάθε πρόγραμμα αποτελείται από δύο βασικές συναρτήσεις:

void setup ( ) { }
ότι κώδικας υπάρχει ανάμεσα στις αγκύλες θα εκτελεστεί μία φορά κατά την εκτέλεση του προγράμματος. Περιλαμβάνει βασικές εντολές αρχικοποίησης του προγράμματος

void loop ( ) { }
Αυτή η συνάρτηση «τρέχει» μετά την **void setup ()**. Ότι κώδικας υπάρχει ανάμεσα στις αγκύλες θα εκτελείται ασταμάτητα και μέχρι την διακοπή της παροχής ηλεκτρικού ρεύματος στον ARDUINO .

Συντακτικές Παρατηρήσεις

//
(σχόλιο μίας γραμμής)
πολύ χρήσιμο για την καταγραφή σημειώσεων

/* */
(σχόλιο περισσότερων γραμμών)
πολύ χρήσιμο όταν έχεις πολλά να γράψεις.....

{ } (άγκιστρα)
Χρησιμοποιούνται για να ορίσουμε μία ομάδα (block) εντολών

; (Ερωτηματικό)
Κάθε γραμμή κώδικα απαιτείται να τελειώνει με το σύμβολο του ελληνικού ερωτηματικού

```
/*
 Fading
 This example shows how to fade an LED using the analogWrite() function
*/

int ledPin = 9;

void setup()  { // nothing happens in setup  }

void loop(){
                // fade in from min to max in increments of 5 points:
  for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5)
    { analogWrite(ledPin, fadeValue);
      delay(30);
    }

                // fade out from max to min in increments of 5 points:
  for(int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5)
    { analogWrite(ledPin, fadeValue);
      delay(30);
    }
          }
```


Τύποι μεταβλητών που χρησιμοποιεί ο ARDUINO

int
(Integer-Ακέραιος αριθμός)
Δεσμεύει μνήμη **2 bytes** (16 bits), μπορεί να αποθηκεύσει ένα αριθμό από -32768 μέχρι 32767

long
(long - Ακέραιος αριθμός)
Χρησιμοποιείται όταν ο τύπος **int** δεν είναι αρκετά μεγάλος. Δεσμεύει μνήμη **4 bytes** (32 bits), μπορεί να αποθηκεύσει ένα αριθμό από -2147483648 μέχρι 214748367

boolean
(boolean-δυαδική μεταβλητή)
Είναι μία απλή μεταβλητή True ή False. Δεσμεύει μνήμη **1 bit**

float
(float - Πραγματικός αριθμός)
Αριθμός κινητής υποδιαστολής. Δεσμεύει μνήμη **4 bytes** (32 bits), μπορεί να αποθηκεύσει ένα αριθμό από
$-3.4028235\ 10^{38}$ μέχρι $3.4028235\ 10^{38}$

char
(character - χαρακτήρας)
Αποθηκεύει ένα χαρακτήρα με την χρήση κώδικα ASCII (π.χ 'A' = 65). Δεσμεύει μνήμη **1 bytes** (8 bits). Ο Arduino διαχειρίζεται τα string (συμβολοσειρά) σαν ένα πίνακα από χαρακτήρες

15

Έλεγχος δομής του προγράμματος

Τελεστές σύγκρισης

== (ίσο με)
!= (όχι ίσο με)
< (μικρότερο από)
> (μεγαλύτερο από)

for(int i = 0; i < αριθμός_επαναλήψεων ; i++) { }
Χρησιμοποιείται όταν θέλουμε τμήμα του κώδικα, που βρίσκεται ανάμεσα στις αγκύλες, να επαναληφθεί συγκεκριμένο αριθμό επαναλήψεων (μέτρηση του i προς τα πάνω i++ ή κάτω i--)

Μαθηματικοί Τελεστές

= (αντιστοίχιση)
% (modulo) (επιστρέφει το υπόλοιπο της διαίρεσης)
+ (Τελεστής πρόσθεσης)
- (Τελεστής αφαίρεσης)
***** (Τελεστή πολλαπλασιασμού)
/ (Τελεστής διαίρεσης)

if (συνθήκη 1) { }
else if (συνθήκη 2) { }
else { }
Αν η συνθήκη 1 είναι αληθής, θα εκτελεστεί το σώμα εντολών ακριβώς μετά την συνθήκη 1. Αν η συνθήκη 2 είναι αληθής, θα εκτελεστεί το σώμα εντολών ακριβώς μετά την συνθήκη 2. Αν δεν αληθεύει καμιά συνθήκη θα εκτελεστεί το σώμα εντολών μετά την εντολή else

Λογικοί Τελεστές

&& (logical AND)
|| (logical OR)
! (logical NOT)

Bitwise Τελεστές

& (AND)
| (OR)
~ (NOT)

Διαχείριση Εισόδων/Εξόδων

Ψηφιακών εισόδων

pinMode (pin, mode);
Ορίζει το pin (0~19) σε συγκεκριμένο mode λειτουργίας δηλαδή, σαν είσοδο (INPUT) ή έξοδο (OUTPUT) .

digitalRead (pin);
Όταν το pin έχει οριστεί σαν είσοδο μπορούμε να διαβάσουμε την ψηφιακή τιμή στην είσοδο.

digitalWrite(pin, value);
Όταν το pin έχει οριστεί σαν έξοδο μπορούμε να ορίσουμε την κατάστασή του pin σε HIGH ή LOW.

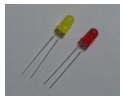
Διαχείριση αναλογικών εισόδων και έξοδοι PWM

analogRead (pin);
Διαβάζει την αναλογική τιμή της τάσης που υπάρχει στο pin. Η τιμή που θα διαβάσουμε είναι ένας ακέραιος αριθμός μεταξύ του 0 (για τα 0 volts) και 1024 (για τα 5 volts).

analogWrite (pin,value);
Μερικά από τα pin του ARDUINO (3, 5, 6, 9, 10, 11) υποστηρίζουν την δημιουργία σήματος PWM (pulse width modulation). Το duty cycle του PWM καθορίζεται με ένα ακέραιο αριθμό από 0 (για 0% duty cycle) και 255 (για 100% duty cycle).

Εξαρτήματα που θα χρησιμοποιήσουμε στο εργαστήριο

Led

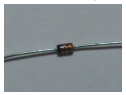


Τρανζίστορ

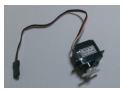
Ποτενσιόμετρο

Ολοκληρωμένο κύκλωμα (IC)

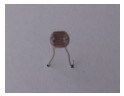
Δίοδος



RC-Servo

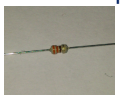


Φωτο-αντίσταση (LDR)



Μπουτόν

Αντίσταση



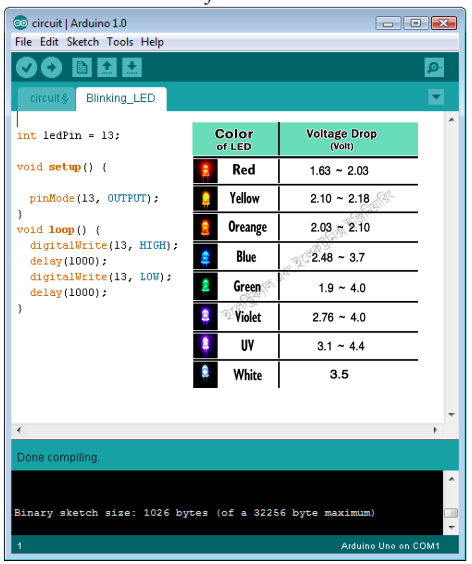
DC-κινητήρας

Πιεζο-στοιχείο

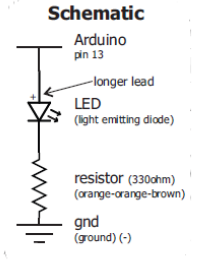
Ρελέ

Ένα απλό παράδειγμα

•Ο κώδικας:



•Το κύκλωμα



- **Η χρήση της αντίστασης περιορισμού ρεύματος R είναι απαραίτητη προκειμένου να προφυλαχθεί η δίοδος LED από τη διαρροή υψηλού ρεύματος το οποίο θα μπορούσε να την καταστρέψει.**
- $I_{LED}=(5-2)/330 = 9.09$ mA
- Η πτώση τάσης στο LED είναι τυπικά 2 volt

Βασικές συναρτήσεις σχετικά με το χρόνο

delay()

- https://www.arduino.cc/reference/en/

Description

Pauses the program for the amount of time (in miliseconds) specified as parameter. (There are 1000 milliseconds in a second.)

Syntax

delay(ms)

Parameters

ms: the number of milliseconds to pause (*unsigned long*)

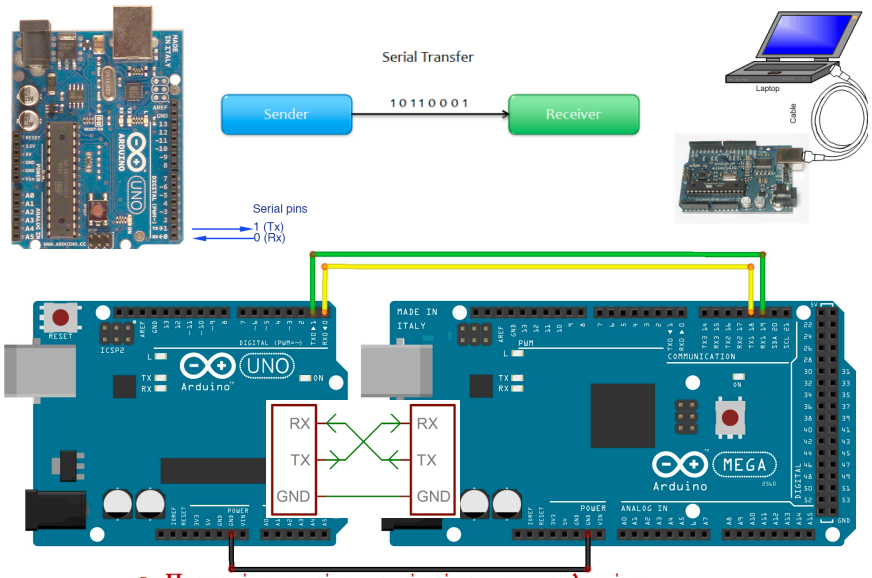
Returns

nothing

Example

```
int ledPin = 13;

void setup(){ pinMode(ledPin, OUTPUT); }
void loop() {
                digitalWrite(ledPin, HIGH);
              delay(1000);
              digitalWrite(ledPin, LOW);   delay(1000);
            }
```


Σειριακή επικοινωνία συστημάτων



- **Προσοχή: απαιτείται κοινή γείωση των κυκλωμάτων**

Σειριακή επικοινωνία με την οθόνη του υπολογιστή

millis()

Description

Returns the number of milliseconds since the Arduino board began running the current program. This number will overflow (go back to zero), after approximately 50 days.

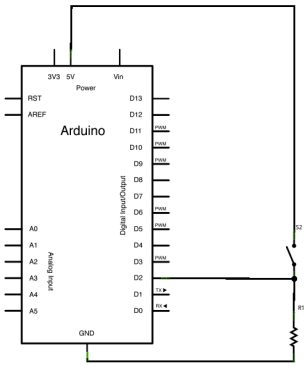
Returns

Number of milliseconds since the program started (*unsigned long*)

Example

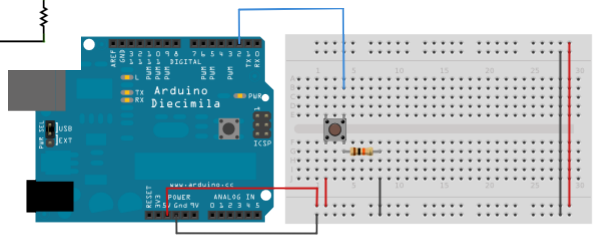
```
unsigned long time;

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

void loop(){
              Serial.print("Time: ");
                           time = millis();
              Serial.println(time); //prints time since program
started
              delay(1000);
             }
```


Παράδειγμα 1: (χρήση διακοπτών push button)



Παράδειγμα 1: Ενεργοποίηση και απενεργοποίηση του led 13 μέσω διακόπτη push button

Παράδειγμα 2: Απεικόνιση της κατάστασης του διακόπτη push button, μέσω της σειριακής θύρας, στην οθόνη του υπολογιστή



Απάντηση στο παράδειγμα 1

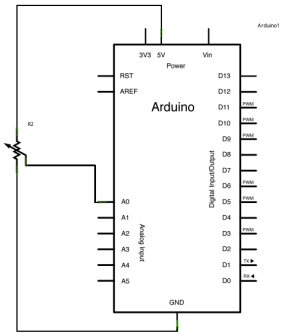
```
const int buttonPin = 2;
const int ledPin =  13;

int buttonState = 0;

void setup() {
           pinMode(ledPin, OUTPUT);
           pinMode(buttonPin, INPUT);
             }

void loop(){
          buttonState = digitalRead(buttonPin);

              if (buttonState == HIGH) {digitalWrite(ledPin, HIGH);}
            else                       {digitalWrite(ledPin, LOW);}
           }
```

Πλήθος παραδειγμάτων μπορείτε να βρείτε στην διεύθυνση
http://www.arduino.cc/en/Tutorial/HomePage

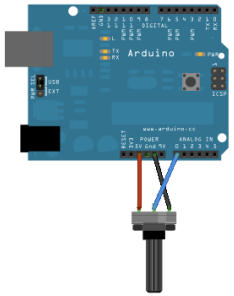
Απάντηση στο παράδειγμα 2

```
/*
  DigitalReadSerial
 Reads a digital input on pin 2, prints the result to the serial monitor


int pushButton = 2;

void setup() {
              Serial.begin(9600);
              pinMode(pushButton, INPUT);
             }

void loop() {
    int buttonState = digitalRead(pushButton);

  Serial.println(buttonState);      // print out the state of the button:
  delay(1);                        // delay in between reads for stability
            }
```

Παράδειγμα 3 (Μετατροπή αναλογικού σήματος σε ψηφιακό)

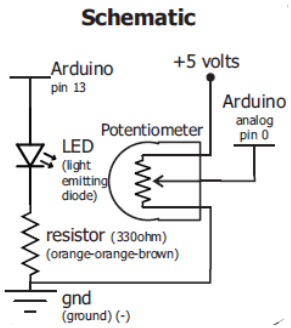


Παράδειγμα 3: Καταγραφή αναλογικού σήματος και εκτύπωση του αποτελέσματος στο σειριακό μόνιτορ

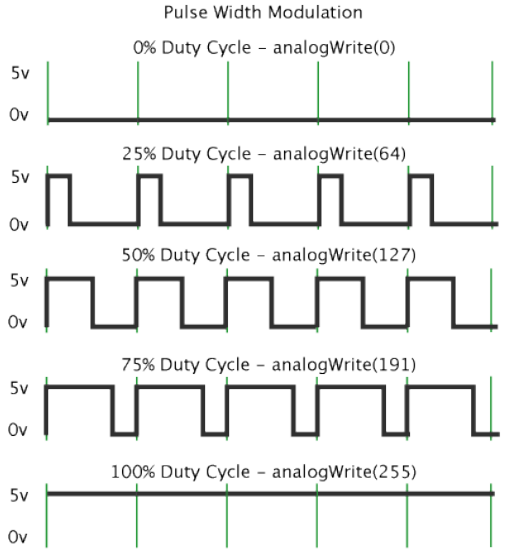


```
/*
  ReadAnalogVoltage
  Reads an analog input on pin 0, converts it to voltage, and prints the result
to the serial monitor */

void setup() { Serial.begin(9600); }
void loop()  {  int sensorValue = analogRead(A0);

  float voltage = sensorValue * (5.0 / 1023.0);/* Convert the analog reading
                                                  (which goes from 0 - 1023) to a
                                                    voltage (0 - 5V):*/
  Serial.println(voltage);
          }
```

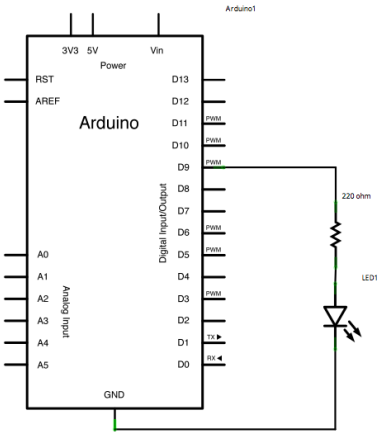

Ρύθμιση φωτεινότητας με ποτενσιόμετρο και τη συνάρτηση delay

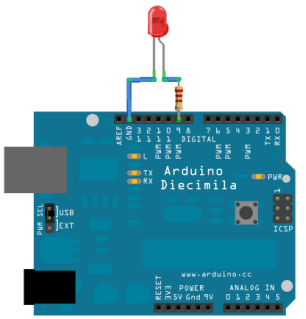


Η έννοια του PWM



Ρύθμιση φωτεινότητας LED με την βοήθεια σήματος PWM





Ρύθμιση φωτεινότητας LED με την βοήθεια σήματος PWM

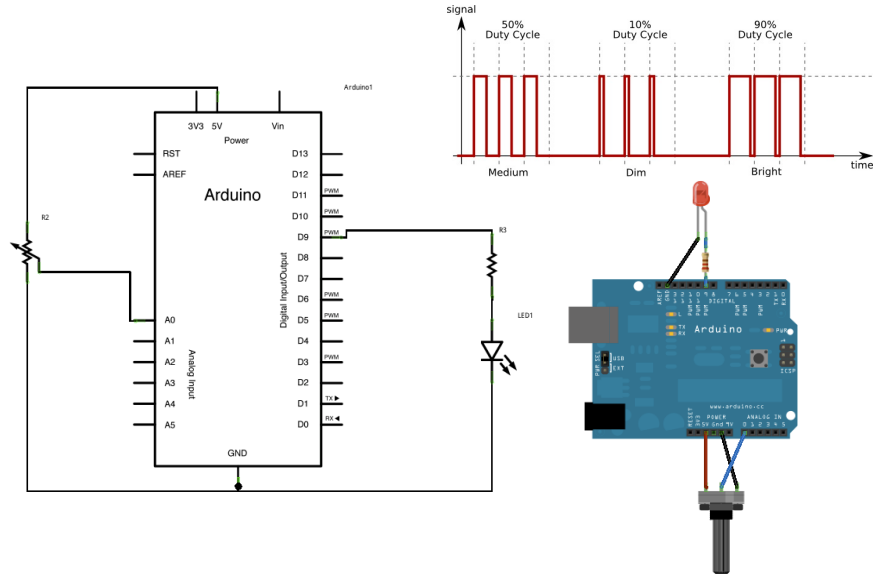
Fading

Demonstrates the use of analog output (Pulse Width Modulation (PWM)) to fade an LED. PWM is a technique for getting an analog-like behavior from a digital output by switching it off and on very fast.

Code

```
/*
 Fading
 This example shows how to fade an LED using the analogWrite()
function
*/


int ledPin = 9;

void setup()  { // nothing happens in setup  }

void loop(){
                // fade in from min to max in increments of 5 points:
  for(int fadeValue = 0 ; fadeValue <= 255; fadeValue +=5)
    { analogWrite(ledPin, fadeValue);
      delay(30);
    }

                // fade out from max to min in increments of 5 points:
  for(int fadeValue = 255 ; fadeValue >= 0; fadeValue -=5)
    { analogWrite(ledPin, fadeValue);
      delay(30);
    }
          }
```

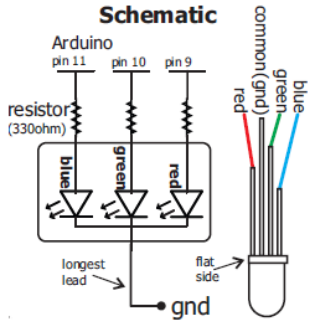

Ρύθμιση φωτεινότητας LED με PWM και ποτενσιόμετρο



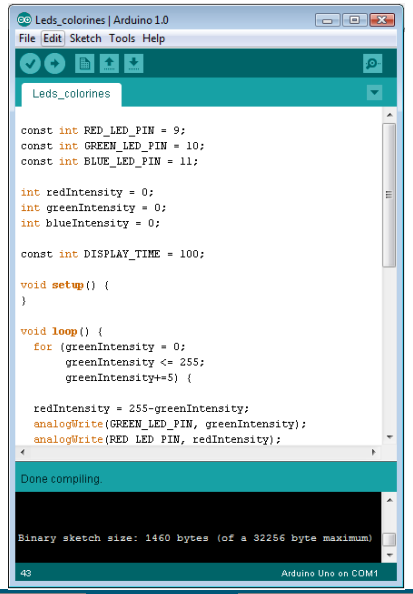
Ρύθμιση φωτεινότητας LED με PWM και ποτενσιόμετρο

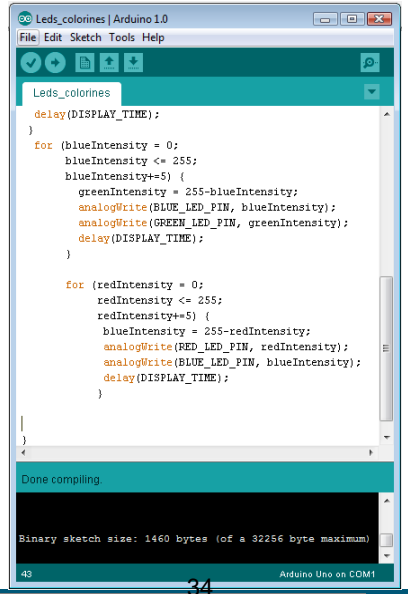
```
/*
  Analog input, analog output, serial output
 Reads an analog input pin, maps the result to a range from 0 to 255 and uses
the result to set the pulsewidth modulation (PWM) of an output pin. Also prints
the results to the serial monitor.
 */

const int analogInPin = A0;
const int analogOutPin = 9;//Analog output pin that the LED is attachedto

int sensorValue = 0;        // value read from the pot
int outputValue = 0;        // value output to the PWM (analog out)

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

void loop() {
  sensorValue = analogRead(analogInPin);
                        // map it to the range of the analog out:
  outputValue = map(sensorValue, 0, 1023, 0, 255);

analogWrite(analogOutPin, outputValue);//change the analog out value:


  Serial.print("sensor = " );// print the results to the serial monitor:
   Serial.print(sensorValue);
  Serial.print("\t output = ");
  Serial.println(outputValue);
                        // wait 2 milliseconds before the next loop
                        // for the analog-to-digital converter to settle
                        // after the last reading:
  delay(2);
            }
```


Χρήση LED τριπλού χρώματος

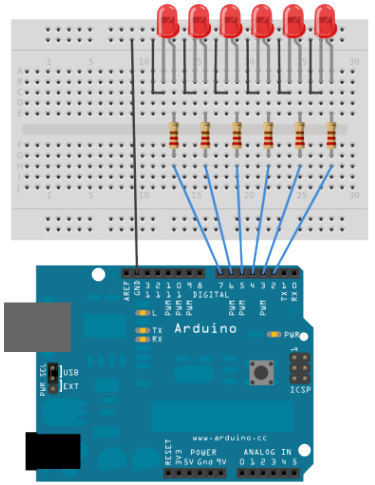


Χρήση LED τριπλού χρώματος





Παράδειγμα: For Loop (The Knight Rider)




```
/*
  For Loop Iteration
 Demonstrates the use of a for() loop.
 Lights multiple LEDs in sequence, then in reverse.
 */

int timer = 100; // The higher the number, the slower the timing.

void setup() {
                // use a for loop to initialize each pin as an output:
  for (int thisPin = 2; thisPin < 8; thisPin++)
      { pinMode(thisPin, OUTPUT); }
            }

void loop()    {
  // loop from the lowest pin to the highest:
  for (int thisPin = 2; thisPin < 8; thisPin++)
          {digitalWrite(thisPin, HIGH); // turn the pin on:
           delay(timer);                // turn the pin off:
           digitalWrite(thisPin, LOW);
           }

  // loop from the highest pin to the lowest:
  for (int thisPin = 7; thisPin >= 2; thisPin--)
    {digitalWrite(thisPin, HIGH);// turn the pin on:
     delay(timer); // turn the pin off:
     digitalWrite(thisPin, LOW);
    }
            }
```