**Variable digital**

Se caracterizan por tener dos estados diferenciados. Un ejemplo de una señal digital es el interruptor del timbre de tu casa, por que este interruptor tiene dos estados pulsado y sin pulsar.

**Variable análoga**

Son aquellas que pueden tomar un número infinito de valores comprendidos entre dos límites. Un ejemplo de sistema electrónico analógico es un parlante, que se emplea para amplificar el sonido de forma que éste sea oído por una gran audiencia. Las ondas de sonido que son analógicas en su origen, son capturadas por un micrófono y convertidas en una pequeña varia-ción analógica de tensión denominada señal de audio.

**Conversor análogo-digital CAD ó ADC**

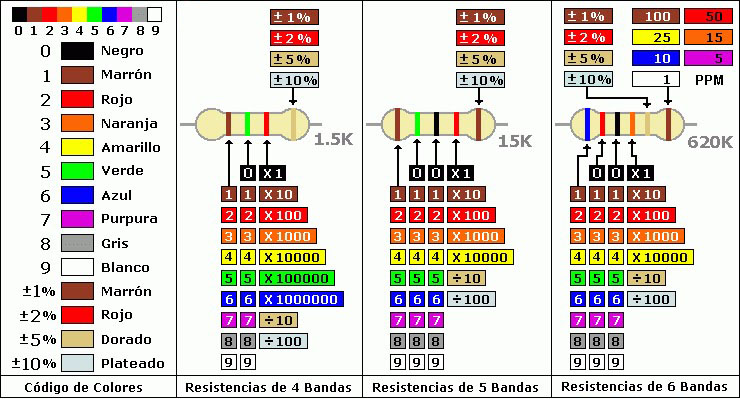
Un conversor (o convertidor) analógico-digital (CAD), (o tam-bién ADC del inglés "Analog-to-Digital Converter") es un dispo-sitivo electrónico capaz de convertir una entrada analógica de voltaje en un valor binario.

**Modulación por ancho del pulso PWM**

La modulación por ancho de pulsos (también conocida como PWM, siglas en inglés de pulse-width modulation) de una señal o fuente de energía es una técnica en la que se modifica el ciclo de trabajo de una señal periódica, ya sea para transmitir información a través de un canal de comunica-ciones o para controlar la cantidad de energía que se envía a una carga.

**Resistencias**

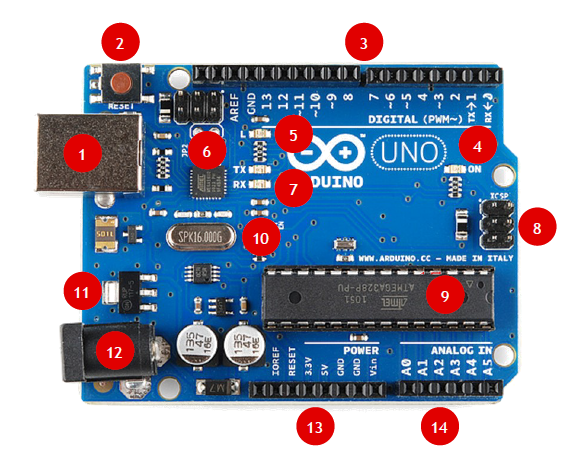
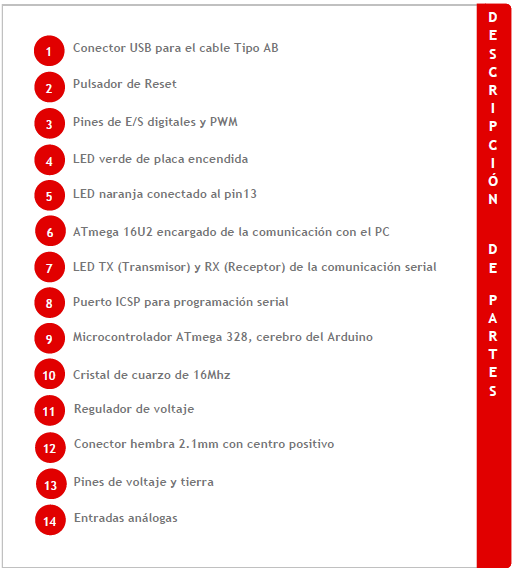
Tabla de colores de resistencias



**Programación arduino**

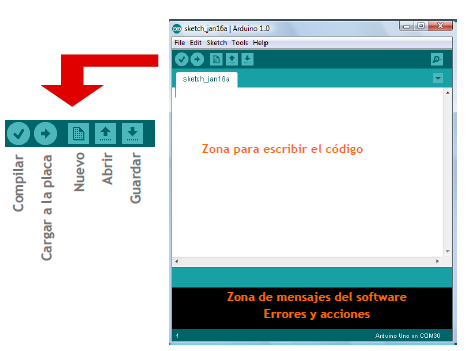
La programación es un gran recurso que nos permite crear diversas secuen-cias de pasos lógicos que van a satisfacer nuestras necesidades y las de nuestros sistemas. Programar involucra áreas como el análisis y diseño de la aplicación.

Placa Arduino:



Estructuras

Arduino se programa en el lenguaje de alto nivel C/C++. El programa de Arduino se puede dividir en tres partes principales: la estructura, las variables (valores y constantes) y funciones.



Son dos funciones principales que debe tener todo programa en Arduino:

setup(){

}

Código de configuración inicial, solo se ejecuta una vez.

loop(){

}

Esta función se ejecuta luego del setup(), se mantiene ejecutándose hasta que se des- energice o desconecte el Arduino.

**Funciones digitales**

pinMode() Permite configurar un pin

pinMode(pin,modo) // Inicializa el pin como una salida

pinMode (13,OUTPUT);

pinMode (a,INPUT);

digitalRead()

**Leer un pin digital (0 ó 1)**

digitalRead(pin)

int a = digitalRead (13);

digitalWrite()

digitalWrite(pin,estado)

digitalWrite (13,HIGH);

digitalWrite (13,LOW);

**Funciones análogas**

Ideales para la lectura y escritura de valores análogos

analogRead()

Leer un valor análogo 0 a 1023

analogRead(pin)

int a = analogRead (A0);

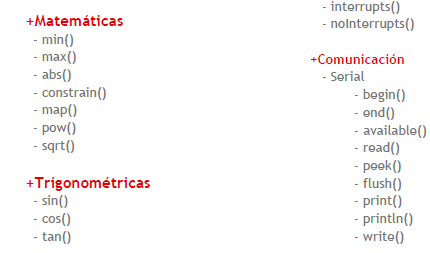
analogWrite() —> PWM

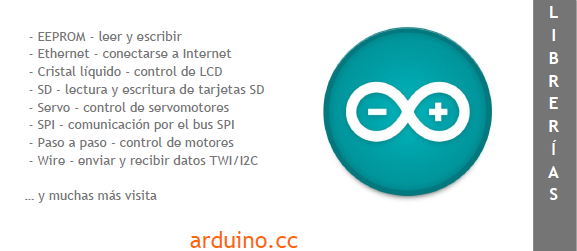
**Escribir un valor análogo 0 a 255**

analogWrite(pin,valor de PWM)

analogWrite (9, 134);







**Lectura serial de una entrada digital**

//------------------------------------

//Declara puertos de entradas y salidas

//------------------------------------

int boton=2; //Pin donde se encuentra el pulsador, entrada

//------------------------------------

//Funcion principal

//------------------------------------

void setup() // Se ejecuta cada vez que el Arduino se inicia

{

//Configuración

pinMode(boton,INPUT); //Configurar el boton como una entrada

Serial.begin(9600); //Inicia comunicación serial

}

//------------------------------------

//Funcion ciclicla

//------------------------------------

void loop() // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

//Guardar en una variable entera el valor del boton 0 ó 1

int estado = digitalRead(boton);

//Condicional para saber estado del pulsador

if (estado==1)

{

// Pulsado

Serial.println("Pulsado"); //Imprime en la consola serial

} // "Pulsado"

else

{

// No esta pulsado

Serial.println("NO Pulsado"); //Imprime en la consola serial

} // "NO Pulsado"

delay(100); //Retardo para la visualización de datos en la consola

}

//Fin programa

**Lectura serial de una entrada análoga**

void setup() // Se ejecuta cada vez que el Arduino se inicia

{

Serial.begin(9600); //Inicia comunicación serial

}

void loop() // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

//Guardar en una variable entera el valor del potenciómetro 0 a 1024

int valor= analogRead(A0);

//Imprime en la consola serial el valor de la variable

Serial.println(valor);

//Retardo para la visualización de datos en la consola

delay(100);

}

//Fin programa

**Escritura de una salida análoga (Enciende/Apaga un LED de forma proporcional)**

//Declara puertos de entradas y salidas y variables

//--------------------------------------------------

int brillo = 0; //Variable de brillo inicia en 0

int variacion = 5; //Variable de incremento configurada de 5 en 5

int led = 9; //Pin donde se encuentra el LED, salida

//------------------------------------

//Funcion principal

//------------------------------------

void setup () // Se ejecuta cada vez que el Arduino se inicia

{

pinMode(led, OUTPUT); //Configurar el LED como una salida

}

//------------------------------------

//Funcion ciclicla

//------------------------------------

void loop () // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

**// Escritura analoga (PWM) en el LED escribo el valor de brillo**

analogWrite(led, brillo);

// Incremento la variable brillo de 5 en 5

brillo = brillo + variacion;

// Nota: PWM ----> 0 - 255

// Si el brillo es 0 o 255

if (brillo == 0 || brillo == 255)

variacion = -variacion; //La variación se vuelve negativa

delay (30); //Tiempo de incremento en el brillo

}

//Fin programa

**Control ON/OFF con potenciómetro**

void setup() // Se ejecuta cada vez que el Arduino se inicia

{

Serial.begin(9600); //Inicia comunicación serial

pinMode(13,OUTPUT); //Configurar el pin 13 como una salida

}

//------------------------------------

//Funcion ciclicla

//------------------------------------

void loop() // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

//Guardar en una variable el valor de la lectura análoga

int valor = analogRead(A0);

Serial.println(valor); //Imprime el valor por la consola

//Si el valor es mayor o igual a 500

if (valor >= 500)

{

digitalWrite(13,HIGH); //Enciende el LED en el pin 13

}

//Si el valor es menor a 500

else

{

digitalWrite(13,LOW); //Apaga el LED en el pin 13

}

delay(100); //Retardo de 100ms para ver los datos de la consola

}

//Fin programa

**Control de intensidad de un LED**

//Funcion principal

//------------------------------------

void setup() // Se ejecuta cada vez que el Arduino se inicia

{

Serial.begin(9600); //Inicia comunicación serial

pinMode(9,OUTPUT); //Configurar el pin 9 como una salida de PWM

}

//------------------------------------

//Funcion ciclicla

//------------------------------------

void loop() // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

//Guardar en una variable el valor de la lectura análoga de la

// fotocelda

int foto = analogRead(A0);

//Verifica el valor máximo y realizar una conversión

int conversion = 780 - foto;

//Condicional para establecer un valor absoluto

if ( conversion < 0)

conversion = conversion \* -1; //Multiplicar por -1 porque es negativo

//Imprimir datos del valor de la lectura análoga de la fotocelda

Serial.print("Foto : ");

Serial.print(foto);

Serial.println("");

//Imprimir datos del valor de la conversión

Serial.print("Conv : ");

Serial.print(conversion);

Serial.println("");

//Escritura análoga de PWM en el LED de acuerdo a la conversión

analogWrite(9, conversion);

delay(100); //Retardo para datos en la consola

}

//Fin programa

**Contador de pulsos**

//Declara puertos de entradas y salidas y variables

//--------------------------------------------------

int conta = 0; //Variable para guardar el conteo de los pulsos

//------------------------------------

//Funcion principal

//------------------------------------

void setup() // Se ejecuta cada vez que el Arduino se inicia

{

Serial.begin(9600); //Inicia comunicación serial

pinMode(2,INPUT); //Configura el pin 2 como una entrada, pulsador

pinMode(13,OUTPUT); //Configura el pin 13 como una salida, LED

}

//------------------------------------

//Funcion ciclicla

//------------------------------------

void loop() // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

// Si el pulsador esta oprimido

if ( digitalRead(2) == HIGH )

{

// Si el pulsador no esta oprimido, flanco de bajada

if ( digitalRead(2) == LOW )

{

conta++; //Incrementa el contador

Serial.println(conta); //Imprime el valor por consola

delay (100); // Retardo

}

}

// Si el valor del contador es 5

if (conta==5)

{

digitalWrite(13,HIGH); //Enciende el LED

}

// Si el valor del contador es 8

if (conta==8)

{

digitalWrite(13,LOW); // Apaga el LED

}

}

//Fin programa

**Generar tonos con un buzzer**

//Declara puertos de entradas y salidas y variables

//--------------------------------------------------

int buzzer = 9; //Declara Pin del buzzer

int tono = 0; //Declara Pin del potenciómetro

//------------------------------------

//Funcion principal

//------------------------------------

void setup() // Se ejecuta cada vez que el Arduino se inicia

{

// No se configuran parámetros inciales, pero se debe

// colocar el encabezado de la función setup()

}

//------------------------------------

//Funcion ciclicla

//------------------------------------

void loop() // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

//Variable entera donde se almacena el valor del potenciómetro

int sensor = analogRead(tono);

//Variable donde se escala la frecuencia de 100 a 5000Hz

int frecuencia = map(sensor,0,1023,100,5000);

//Variable entera para guardar el tiempo deseado en ms

int duracion = 250;

//Funcion tone(), que recibe:

// 1ra posición: Pin del elemento sonoro

// 2da posición: Frecuencia deseada en Hz

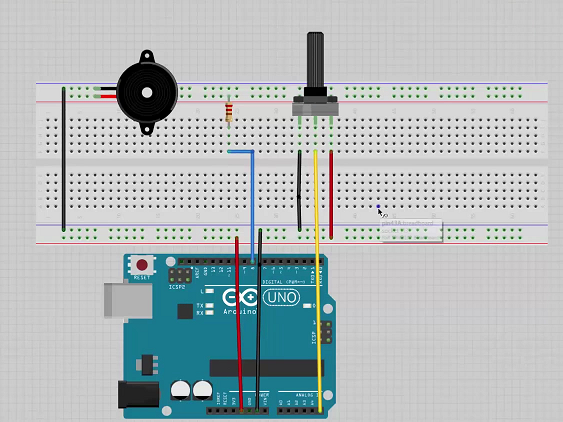
// 3ra posición: Duración del tono

tone(buzzer, frecuencia, duracion);

//Retardo

delay(100); }

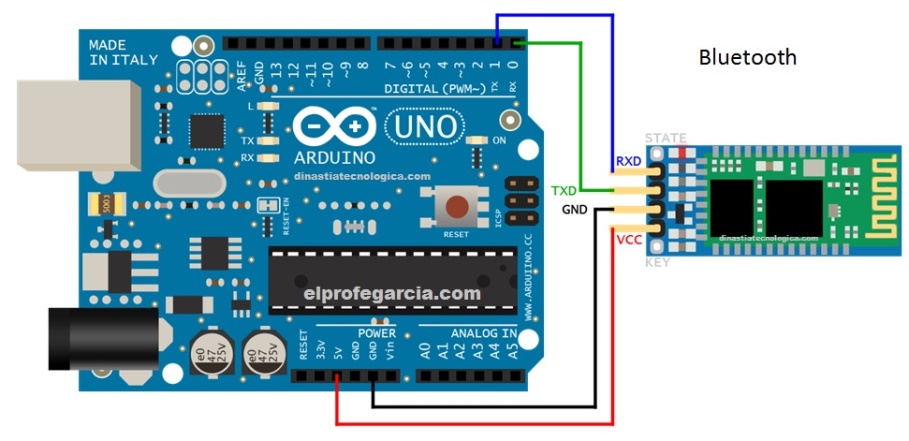
//Fin programa





La variable tono es de tipo analógica. Con la instrucción map mapeamos la variable tono desde 0 a 1023 y la asociamos con los valores de salida de 600 a 3000 respectivamente, por lo tanto el tono se corresponde con la frecuencia de salida. lo mismo con la variable altavoz y duración. Entonces el tono sale por el altavoz con una frecuencia y una determinada duración establecida en la función tone. Con la instrucción delay (200) cambiamos la frecuencia (velocidad) de reproducción.

**Conectar modulo Bluetooth**



Desconectar el modulo Bluetooth mientras carga este programa al Arduino, luego conecte:

ARDUINO BLUETOOTH

Pin 0 (RX) TXD

Pin 1 (Tx) RXD

5V VCC

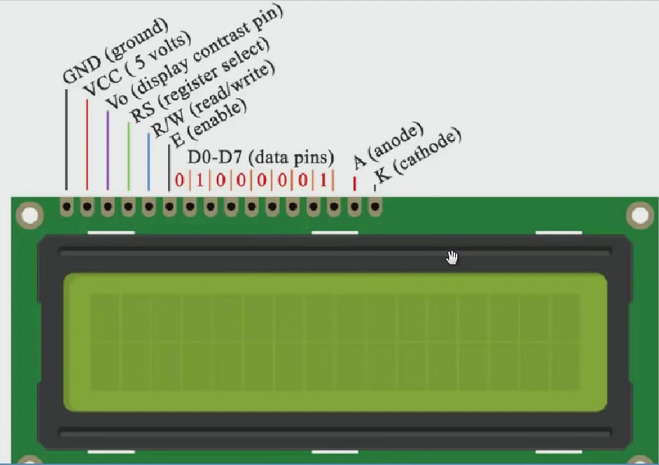
GND GND

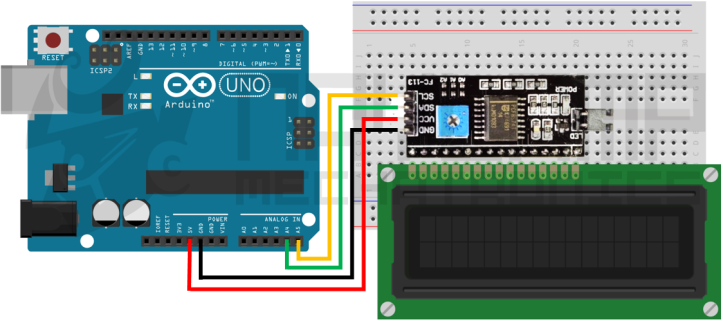
Con las capacidades de Arduino solo debemos tener nociones básicas de electrónica y programación, eso es suficiente para comenzar a desarrollar nuestros proyectos. Un Shield o escudo es una placa que permite expandir funcionalidades a tu Arduino, con lo cual puedes conectar motores, o a la red celular, a una red WiFi, a una red Ethernet o tener un MP3 en el Arduino,

**Caracteres especiales LCD**

Estos LCDs tienen normalmente un controlador **HD44780** o compatible y se componen de una matriz de 5 píxeles horizontales por 8 verticales por carácter. En su memoria **CGRAM** puede almacenar hasta 8 nuevos caracteres







Otra forma:

