



Arduino Inicial

Ing. Juan C. Abdala

Clase 3

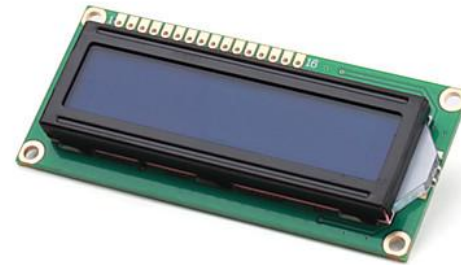
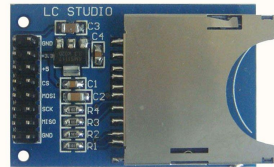
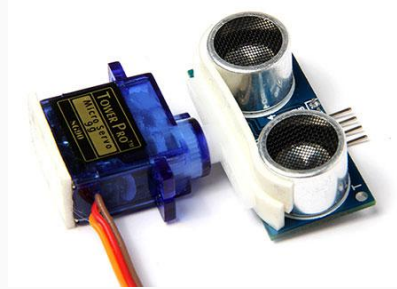
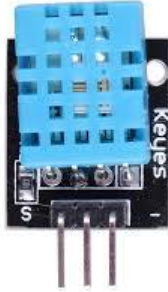
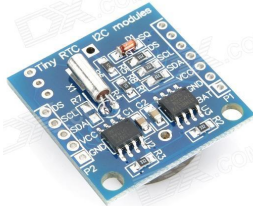
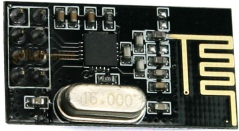


Objetivo de esta clase

- Aprender sobre los periféricos de Arduino : Relé, LCD, DHT y RTC.
- Aprender qué son y cómo se usan los buses de comunicación.
- Aprender qué es y como se descarga una librería.
- Aprender cómo se crea una librería.



Periféricos



SD Card Reader Module



Periféricos

¿Como se usan?



Librerías.

- Son junto a los periféricos la parte más importante de arduino.
- Existen las oficiales (página de arduino) y miles como Software Libre.
- Todas traen uno o más ejemplos de cómo usarlas.



¿Como se instalan?.

1. Descargar
2. Descomprimir
3. Colocar en arduino/libraries/ (dist. ubic. segun S.O.)
4. Agregarla con `#include <"Nombre de Librería">`



Tipo de Periféricos.

Analogicos: Estos reciben alimentacion y devuelven valores en mV(mili Volt) que deben ser traducidos.

Digitales: Nos devuelven el valor procesado solo debemos tomarlo.

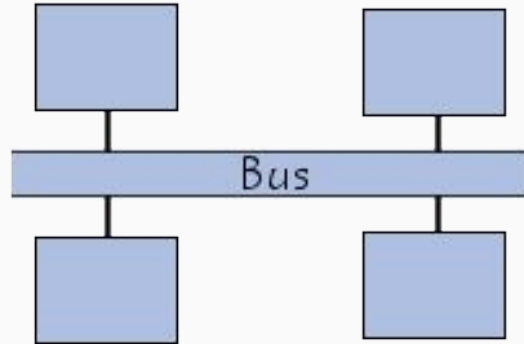
Periféricos

¿Pero cómo se
conectan?



Bus de comunicación

El bus (o canal) es un sistema digital que transfiere datos entre los componentes de una computadora o entre varias computadoras.



Buses de Arduino UNO

1. **SPI**
2. **I2C**
3. **Serial.**
4. **Onewire**



SPI

- SPI (Serial Peripheral Interface)
- Soportado por hardware.
- Modo Maestros esclavos.



SPI

MOSI: Corresponde a las siglas “Master Output – Slave Input”, es decir, el maestro enviará los datos a través de esta línea y el esclavo los recibirá.

MISO: Corresponde a las siglas “Master Input – Slave Output”, y es la línea por la que los esclavos enviarán datos al dispositivo maestro.

SS: Es la señal de “Slave Select”, es decir, la línea que el maestro activará para indicar al esclavo que se va a establecer la comunicación con él.

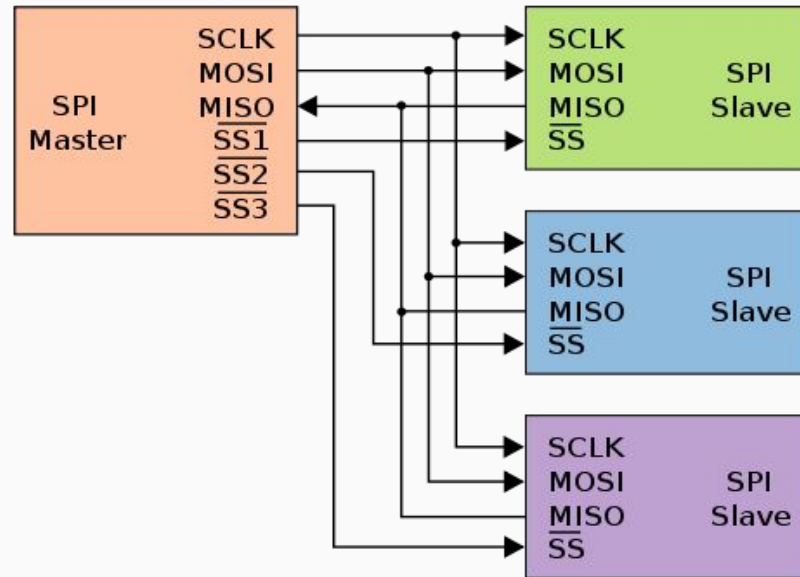
SCLK: Es la señal de reloj, impuesta por el dispositivo maestro.



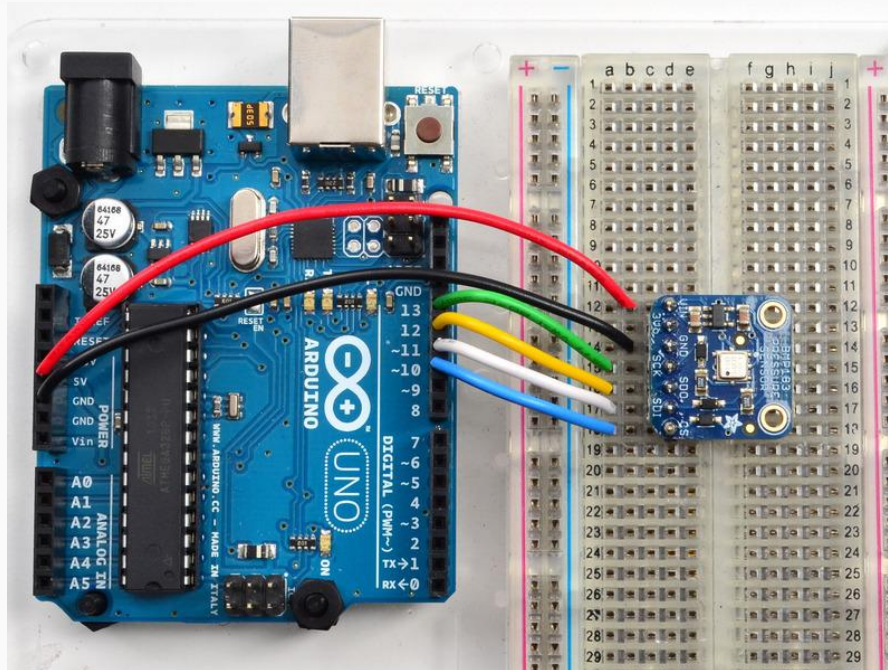
SPI pines Arduino



SPI



Ejemplo Bmp183 Presión



Ejemplo Bmp183 Presión

```
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP183.h>

#define BMP183_CS 10
Adafruit_BMP183 bmp = Adafruit_BMP183(BMP183_CS);

void setup(void)
{
  Serial.begin(9600);
  if(!bmp.begin()) { Serial.print("No se encuentra el sensor");}
}
```

```
void loop()
{
  Serial.print("Presion = ");
  Serial.print(bmp.readPressure());
  Serial.println(" Pa");
  delay(500);
}
```



I2C

- Inter-Integrated Circuit (Inter-Circuitos Integrados).
- Soportado por hardwares.
- Tipo serial, solo 3 conexiones, pero mas complejo a nivel software.



I2C

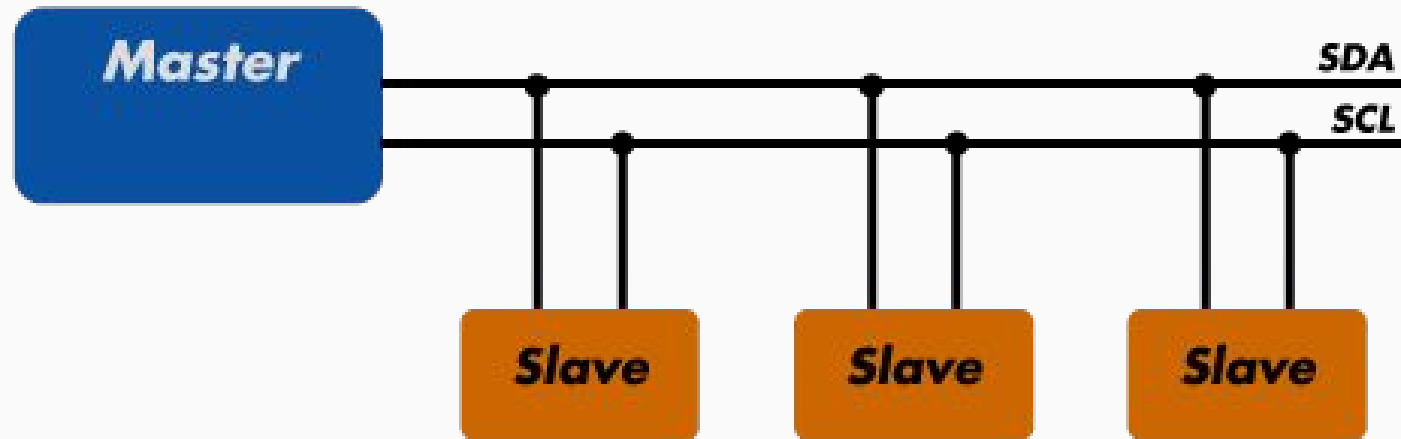
SDA: Datos

SCL: Reloj

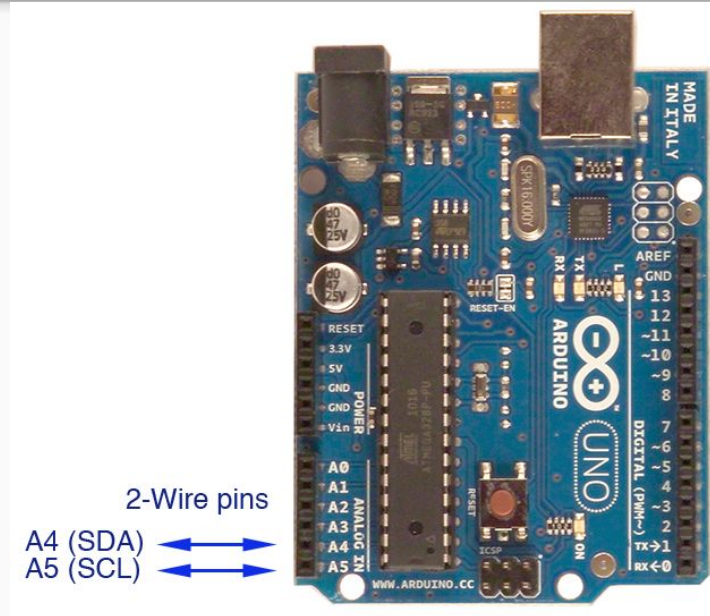
GND: Tierra



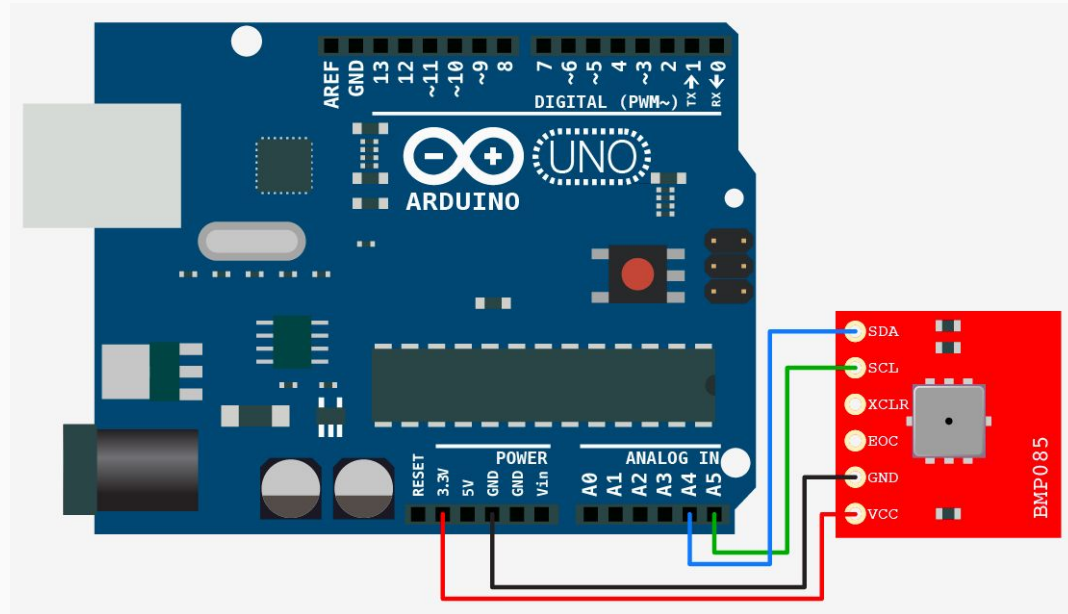
I2C



I2C



Ejemplo Bmp85 - Presión



Ejemplo Bmp85 - Presión

```
#include <Wire.h>
#include <Adafruit_BMP085.h>
```

```
Adafruit_BMP085 bmp;
```

```
void setup() {
  Serial.begin(9600);
  if (!bmp.begin()) {
    Serial.println("No se encuentra sensor");
    while (1) {}
  }
}
```

```
void loop()
{
  Serial.print("Presion = ");
  Serial.print(bmp.readPressure());
  Serial.println(" Pa");
  delay(500);
}
```



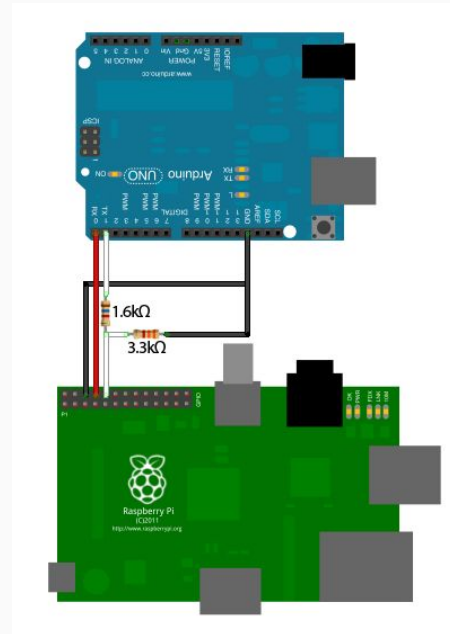
Serial-RS232

- Soportado por hardware y también por software, algunos arduinos poseen varios seriales.
- A nivel hardware se transforma el serial rs232 a USB.



Serial

No solo como debugger,
puede ser el nexa a otra placa.



Serial Funciones

available()

availableForWrite()

begin()

end()

find()

findUntil()

flush()

parseFloat()

parseInt()

peek()

print()

println()

read()

readBytes()

readBytesUntil()

readString()

readStringUntil()

setTimeout()

write()

serialEvent()

<https://www.arduino.cc/en/Reference/Serial>

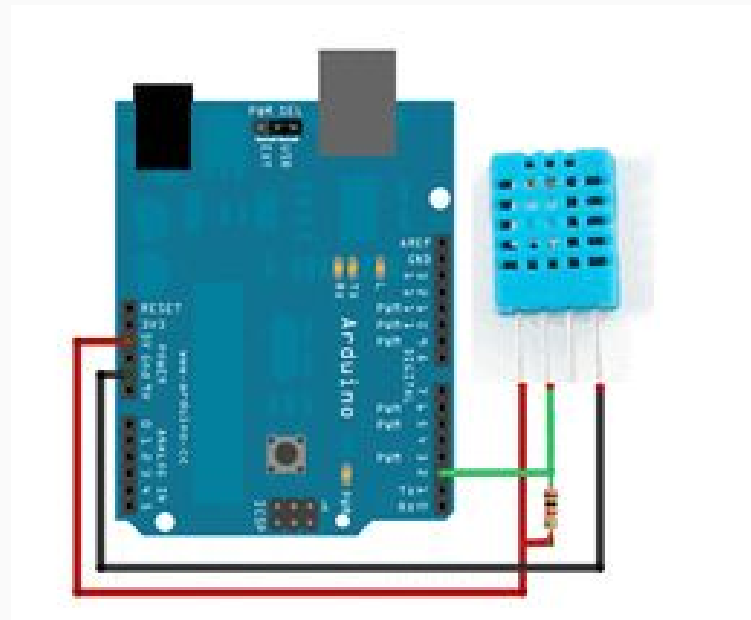


Onewire

- Una sólo conexión, bidireccional.
- Soportados por software, se descarga una librería.
- Muchos periféricos arduinos lo soportan. Ej DHT



Conexión



Ejemplo

```
#include <dht.h>
```

Libreria

```
dht DHT;
```

```
#define DHT11_PIN 5
```

**Pin de
configuracion**

```
void setup()
```

```
{  
  Serial.begin(115200);  
  Serial.println("DHT TEST PROGRAM ");
```

```
}
```

```
void loop()
```

```
{
```

```
  // READ DATA
```

```
  Serial.print("DHT11, \t");
```

```
  DHT.read11(DHT11_PIN);
```

```
  Serial.print(DHT.humidity, 1);
```

```
  Serial.print(",\t");
```

```
  Serial.println(DHT.temperature, 1);
```

```
  delay(2000);
```

```
}
```



Nuestra Librería

Como podemos escalar

Debemos crear nuestras librerías.

En la misma carpeta que el sketch creamos **2 archivos**.

Un HEADER y el Código Fuente



Simple

Carpeta Librería

- Libreria.h
- Libreria.cpp
- Carpeta examples



HEADER libreria .h

- Este archivo es donde vamos a declarar los métodos y atributos necesarios para la librería.
- extern evita que llame instancias en el código.

En el h debe tener.

```
#ifndef __UTILIDADES__  
#define __UTILIDADES__  
class UtilidadesClass  
{  
    public:  
        static void inicio();  
        static int funcion(int parametro);  
  
};  
extern UtilidadesClass utilidades;  
#endif
```



FUENTE librería.cpp

En este archivo está la lógica de la librería aquí escribiremos el código.

Aquí deben estar declarados todos los métodos del archivo *Libreria.h* .

```
#include "utilidades.h"

void UtilidadesClass::inicio(void)
{
    //Algo que debe poner en el setup
}

int UtilidadesClass::funcion(int parametro)
{
    //La funcion que quiera realizar
    return algunParametro;
}
```



En mi proyecto

En mi proyecto solo debo llamar:

```
#include "utilidades.h"
```

y usarla con:

```
util.inicio();
```

```
dato = util.funcion(10);
```



Práctico



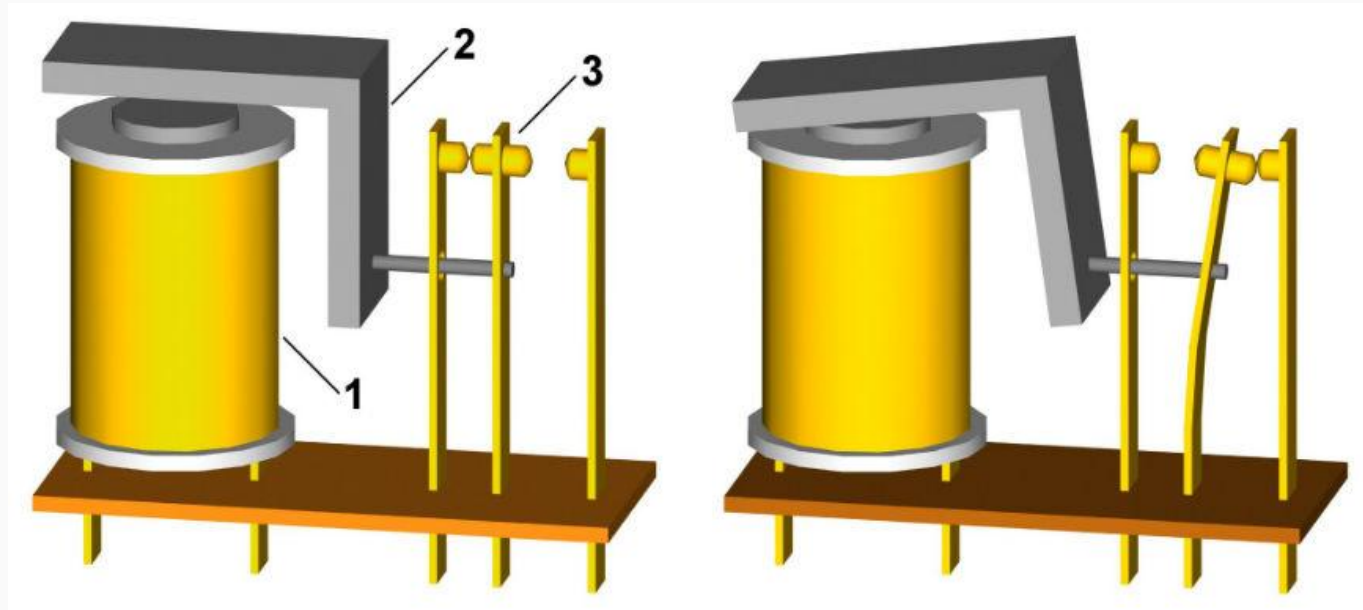
Relé

Relé

- Un relé es un dispositivo electromecánico, que funciona como interruptor controlado por un circuito eléctrico
- Por medio de una bobina o electroimán, se acciona un juego de contactos que permiten abrir o cerrar otros circuitos eléctricos.
- Es capaz de controlar un circuito de salida de mayor potencia que el de entrada.



Rele por dentro



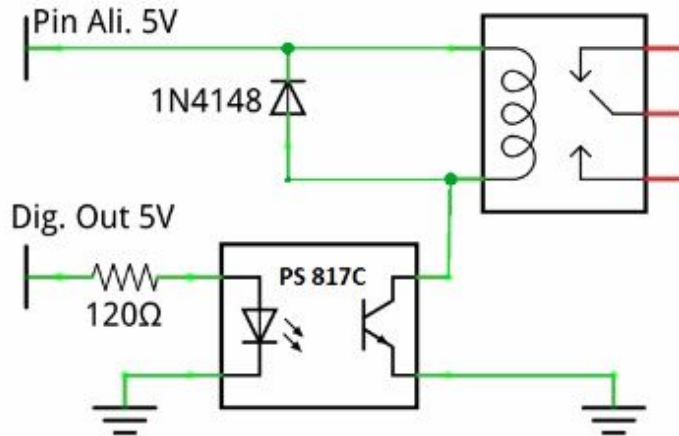
Rele Arduino



- Tan simple como un led, pero se activa por bajo y apaga por alto.
- Podemos controlar Luces, o electrodomésticos.
- Nuestro periférico tiene 2 Relés.
- Soportan hasta 220v a 10 A
- OJO tenemos que saber que corriente necesita nuestra CARGA.



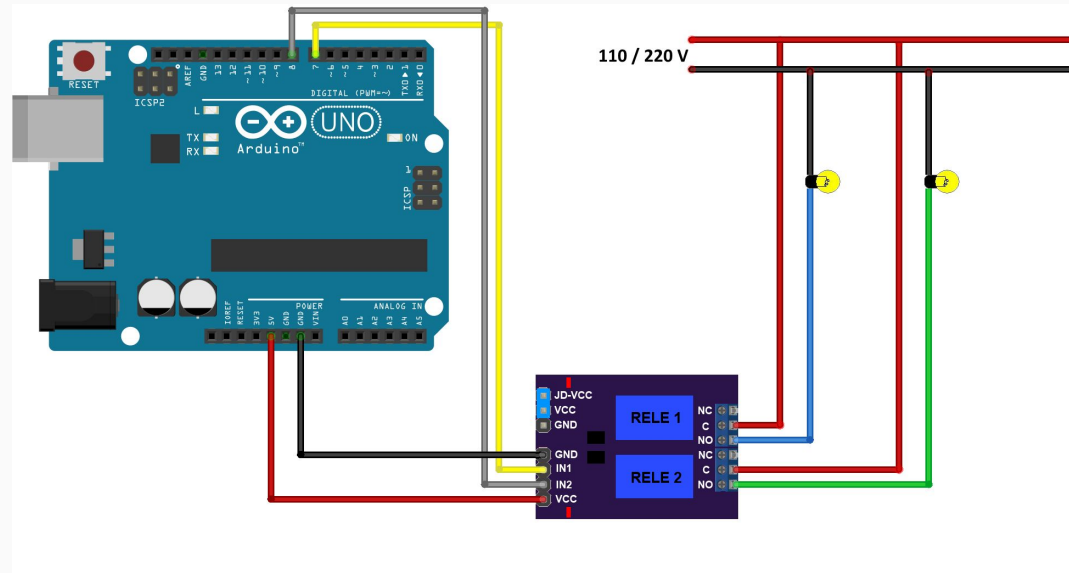
Periferico por Dentro



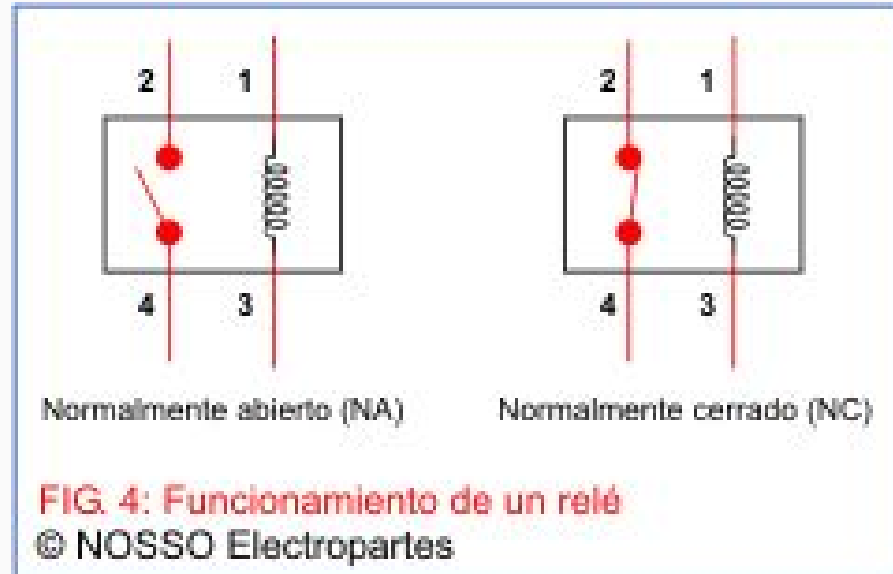
- Nuestros Relés necesitan unos 80 ma para que las bobinas muevan la placa.
- Nuestro arduino no entrega mas de 40 ma en el pin digital.
- Se adapta con un optoacoplador o un Transistor.
- NO NOS IMPORTA VIENE TODO ARMADO!!



Esquemático Rele



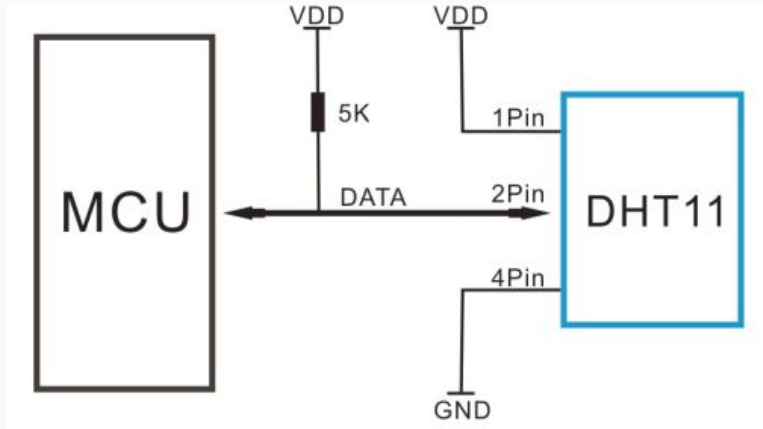
Salidas Relé.



DHT

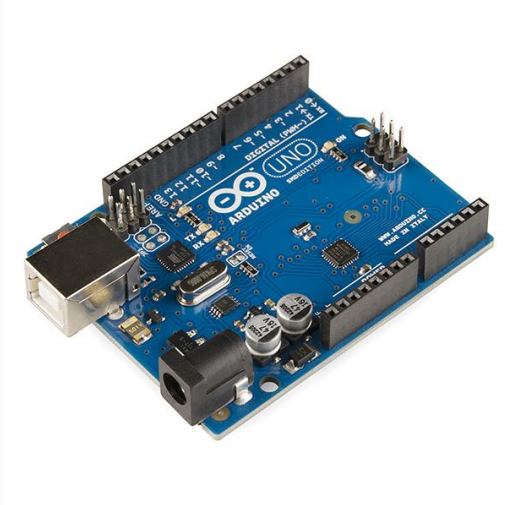
DHT ¿que es?

Item	Measurement Range	Humidity Accuracy	Temperature Accuracy	Resolution	Package
DHT11	20-90%RH 0-50 °C	$\pm 5\%$ RH	$\pm 2^\circ\text{C}$	1	4 Pin Single Row

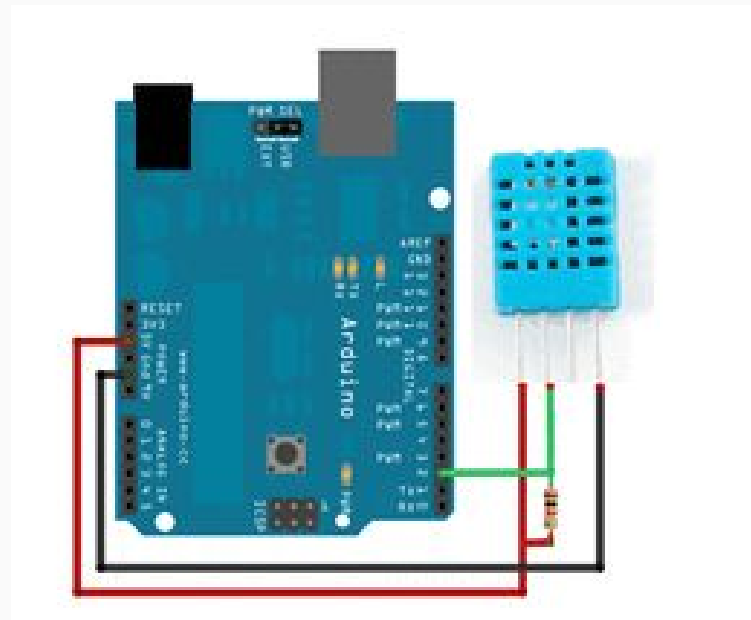


DHT

Arduino + Sensor de Temp. y Humedad



Conexion



DHT LIB

Libreria:

<https://github.com/RobTillaart/Arduino/tree/master/libraries/DHTlib>

Se incluye:

#include <dht.h>



Funciones

Inicio:

dht DHT;

Se define pin:

#define DHT11_PIN 5



Funciones

Leer:

```
DHT.read11(DHT11_PIN);
```

Los datos:

```
DHT.humidity
```

```
DHT.temperature
```

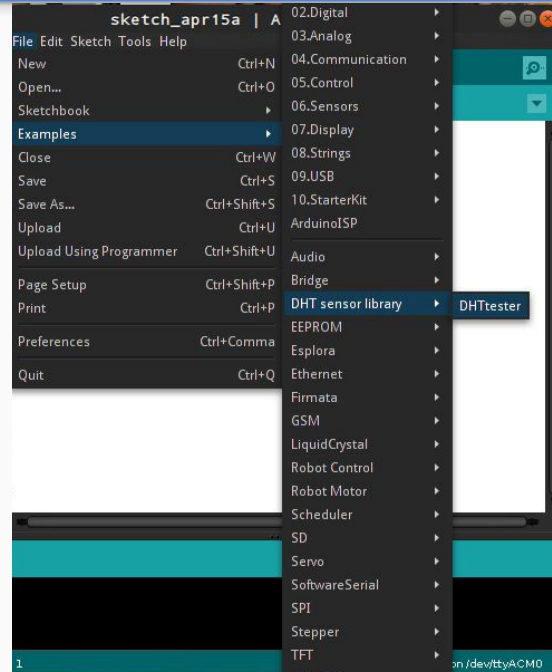


Otros DHT

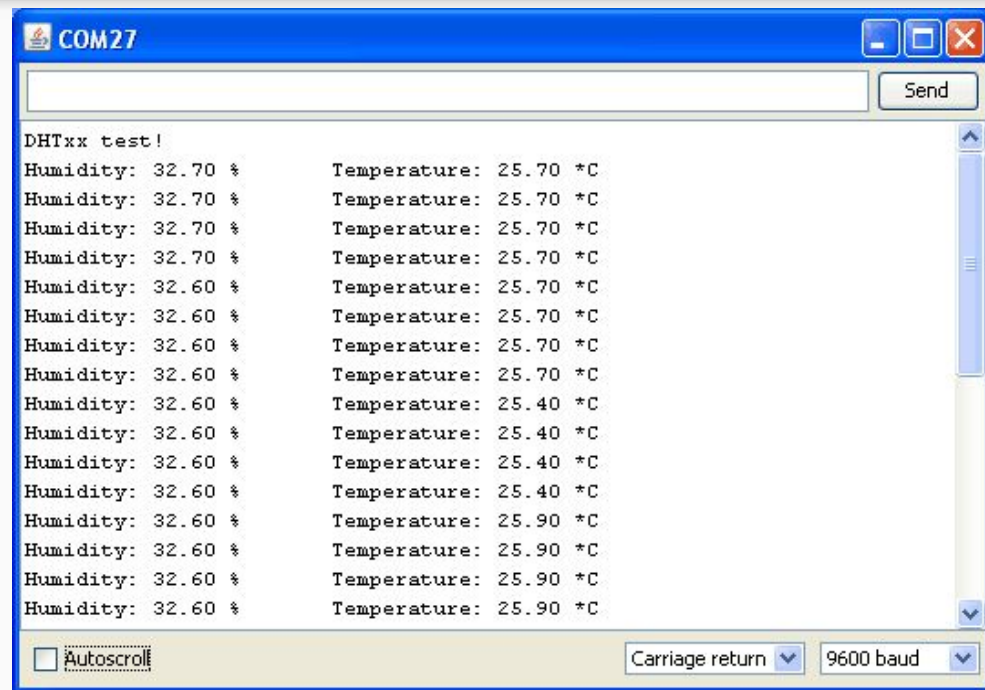
Modelo	DHT11	DHT22
<i>Rango de medición de humedad</i>	20-90 % HR	0-100 % HR
<i>Rango de medición de temperatura</i>	0 hasta 50 °C	-40 hasta 80 °C
<i>Precisión de temperatura</i>	±2 °C	±0.5 °C
<i>Precisión de humedad</i>	±5 % HR	±2 % HR



Aprovechemos Examples



Salida

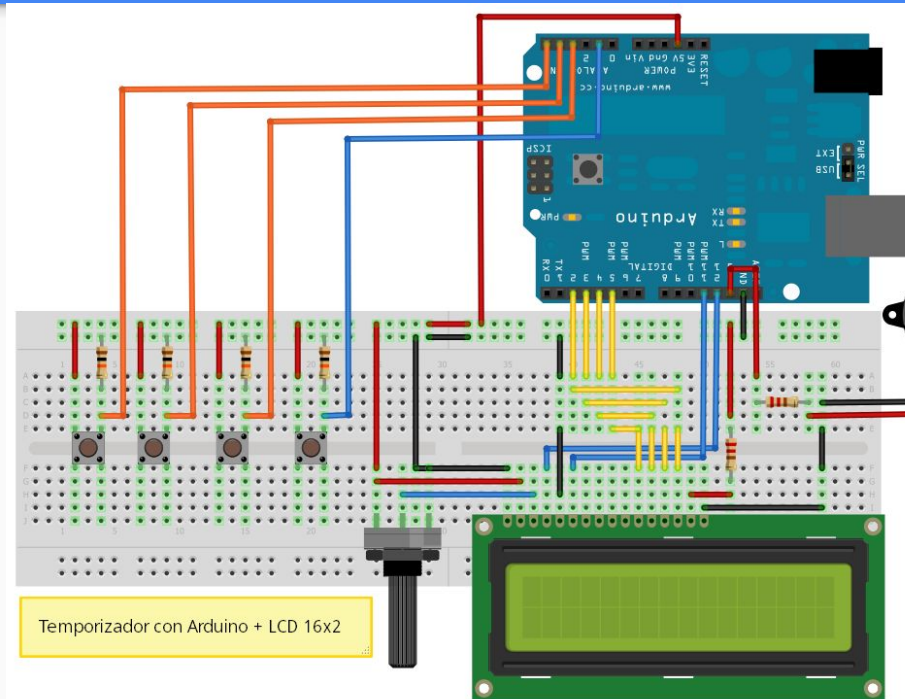


LCD SHIELD

SHIELD LCD DFR_LCD_Keypad



Sería parecido a:

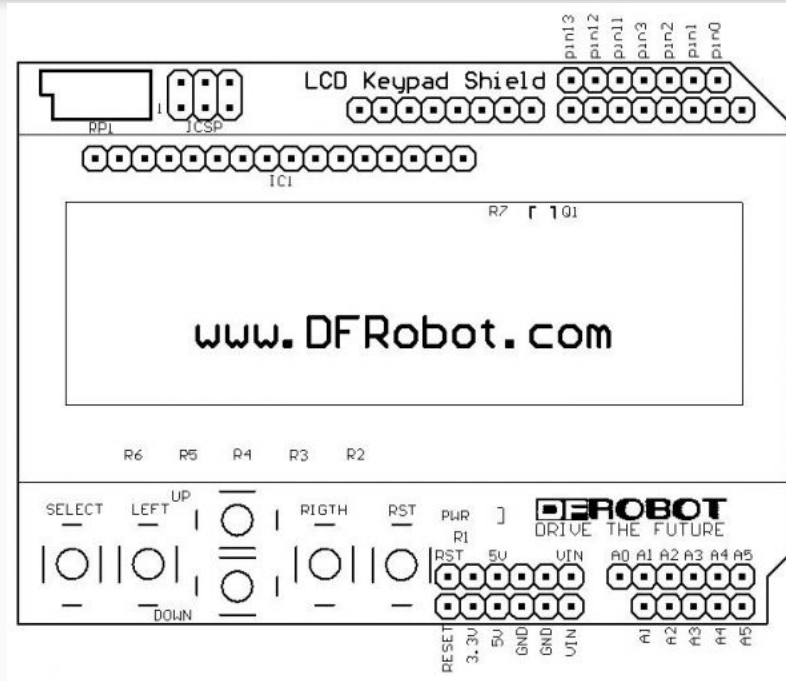


Pines ocupados

Pin	Function
Analog 0	Button (select, up, right, down and left)
Digital 4	DB4
Digital 5	DB5
Digital 6	DB6
Digital 7	DB7
Digital 8	RS (Data or Signal Display Selection)
Digital 9	Enable
Digital 10	Backlit Control



Pinout del SHIELD



inicialización.

Creamos el objeto:

```
DFR_Keypad keypad(16, 2, keyPin, 3);
```

Después solo lo usamos.



Funciones

//define dónde va a empezar a escribir

keypad.setCursor(columna, fila);

//imprime en pantalla

keypad.print("palabra");

//limpia la pantalla

keypad.clear()



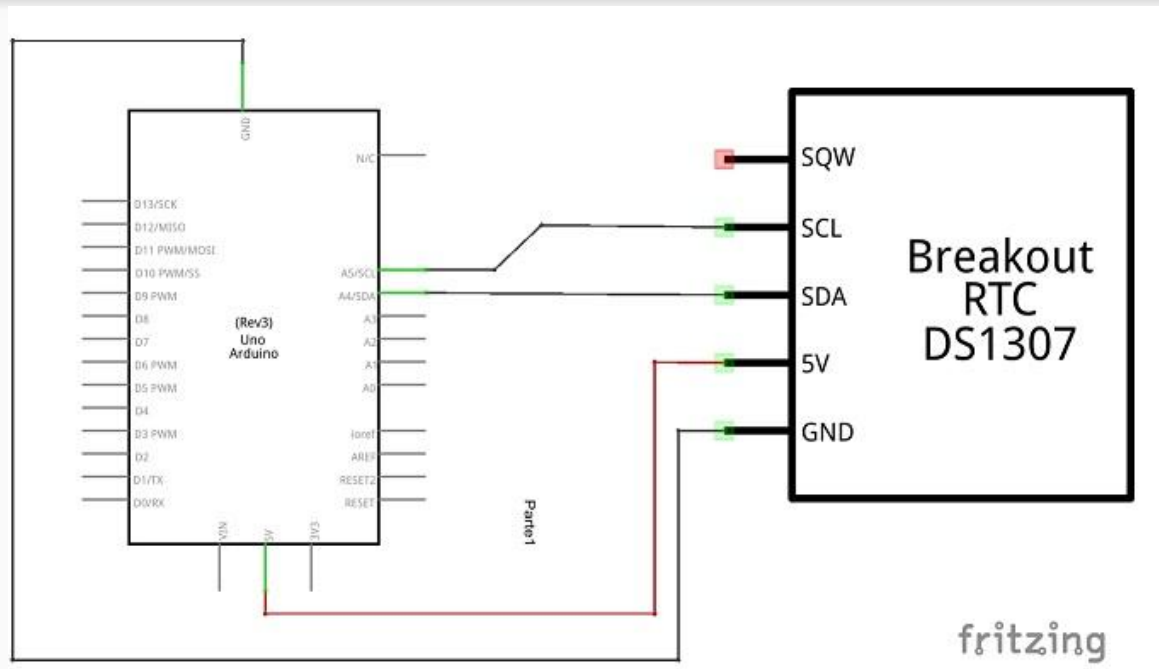
RTC

RTC

- RTC: Real Time Clock, o reloj de tiempo real. Que lleva la cuenta de segundos minutos y horas además de día mes y año automáticamente, válido hasta el año 2100.
- Una batería exterior, que mantiene la fecha y hora cuando no hay corriente.
- Detección automática de corte de corriente y cambio a modo batería.
- Muy bajo consumo, lo que ayuda a que la batería dure entre 5 y 10 años.
- I2C integrado en el mismo chip.



Esquematico RTC DS1307



RTC Lib

Hay muchas Librerías pero vamos a usar:

<http://www.prometec.net/wp-content/uploads/2014/11/DS1307RTC.zip>

Después la incluimos:

```
#include <DS1307RTC.h>
```



Funciones

setSyncProvider(RTC.get); //podemos definir el modelo de reloj que vamos a usar

setTime(21,46,00,8,11,2014); //para ajustar el reloj ej 21:45:00 del dia 8 de Nov de 2014



Funciones

Se accede por:

hour();

minute();

second();

day()

month()

year();



Variables de Ayuda.

El compilador de Arduino, dispone de un par de argumentos que nos informan de la fecha y hora de la última compilación:

`__DATE__` y `__TIME__`

Ojo SetTime no lo soporta directamente en nuestra librería.



Consultas:



abdalajc@gmail.com



@toniabdala

