

## Internet de las cosas y redes eléctricas inteligentes.

### Proyecto Final

Universidad de Guanajuato – Campus Irapuato – Salamanca

Ramos Soto Michel Abraham [[michel.ramos@ugto.mx](mailto:michel.ramos@ugto.mx)]

NUA: 768936

Licenciatura en Ingeniería Comunicaciones y Electrónica, DICIS UG.

Carretera Salamanca – Valle de Santiago Km, 3.5 + 1.8, Comunidad de Pablo Blanco, 36885  
Salamanca, Gto.

6 - Feb -21

### 1. Introducción

Se puede obtener el control de tu hogar a través de dispositivos inteligentes, como tu celular o computadora, que te permiten conectarte y programar tus dispositivos para que cubran tus necesidades.

**Seguridad:** Se podra monitorear tu casa desde donde estés desde la app o tu computadora.

**Vigilancia:** Se puede programar para detectar intrusos o movimiento en un areas específica.

**Iluminación:** Se puede controlar de manera remota el prendido o apagado de las luces de tu hogar, o se puede monitorear el estado de la luces.

#### 1.1 Especificación del proyecto.

El dueño de una casa donde vive con su esposa e hijos quiere automatizar y controlar de forma remota algunos dispositivos eléctricos presentes en su hogar, debido a que normalmente se encuentra fuera de su hogar y quiere estar pendiente de algunas situaciones cotidianas.

Un plano simplificado de la casa se presenta en la siguiente figura.

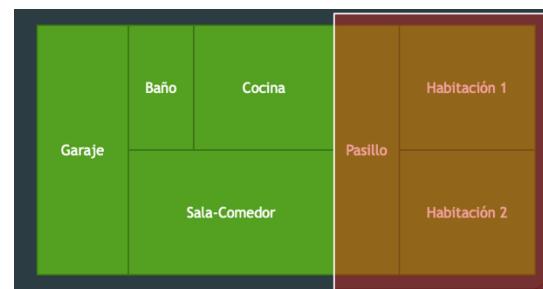


Fig. 1. Ejemplo de casa

El interés principal es poder controlar las áreas del pasillo y habitaciones.

### 2. Objetivo

Desarrollar un ciclo de especificación, diseño, implementación y validación de un proyecto IoT de alcance limitado.

### 3. Diseño e implementación del proyecto

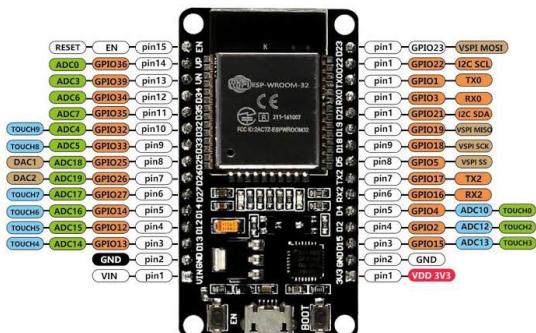
#### 3.1 Esquema de hardware



Fig. 2. Hardware



## Tarjeta de desarrollo ESP32



Tarjeta1. ESP32

La plataforma ESP32 [1] ofrece a bajo costo capacidades wifi, bluetooth y BLE. Tiene un CPU de dos núcleos de hasta 240Mhz que se pueden controlar independientemente. Ademas integra internamente una gran cantidad de periféricos incluyendo: sensores táctiles capacitivos, sensor de efecto Hall, amplificadores de bajo ruido, interfaz para tarjeta SD, Ethernet, SPI de alta velocidad, UART, I2S e I2C.

El punto fuerte de esta plataforma es su gran comunidad en internet que le brinda soporte y desarolla constantemente nuevas herramientas para su uso.

Para su desarrollo cuenta con gran variedad de software, lenguajes de programación, frameworks, librerías, códigos, y otros recursos. Los más comunes a elegir son: Esp-idf (Espressif IoT Development Framework) desarrollado por el fabricante del chip, Arduino (en lenguaje C++), Simba Embedded Programming Platform (en lenguaje Python), RTOS's (como Zephyr Project, Mongoose OS, NuttX RTOS), MicroPython, LUA, Javascript (Espruino, Duktape, Mongoose JS), Basic.

### Especificaciones:

- Voltaje de Alimentación (USB): 5V DC
- Voltaje de Entradas/Salidas: 3.3V DC
- SoC: ESP32
- CPU principal: Tensilica Xtensa 32-bit LX6

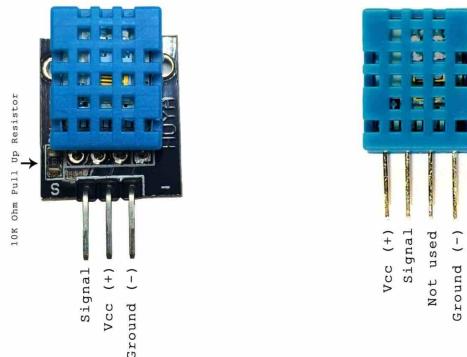
- Frecuencia de Reloj: hasta 240Mhz
- Desempeño: Hasta 600 DMIPS
- Procesador secundario: Permite hacer operaciones básicas en modo de ultra bajo consumo
- Wifi: 802.11 b/g/n/e/i (802.11n @ 2.4 GHz hasta 150 Mbit/s)
- Bluetooth:v4.2 BR/EDR and Bluetooth Low Energy (BLE)
- Xtensa® Dual-Core 32-bit LX6 microprocessors, up to 600 DMIPS

### Memoria:

- 448 KByte ROM
- 520 KByte SRAM
- 16 KByte SRAM in RTC
- QSPI Flash/SRAM, 4 MBytes
- Pines Digitales GPIO: 24 (Algunos pines solo como entrada)
- Conversor Analogico Digital: Dos ADC de 12bits tipo SAR, soporta mediciones en hasta 18 canales, algunos pines soporta un amplificador con ganancia programable
- USART: 2
- Chip USB-Serial: CP2102
- Antena en PCB
- Seguridad:
- Estandares IEEE 802.11 incluyendo WFA, WPA/WPA2 and WAPI
- 1024-bit OTP, up to 768-bit for customers
- Aceleración criptográfica por hardware: AES, HASH (SHA-2), RSA, ECC, RNG



## Sensor de temperatura y humedad DHT11



**Sensor1.** DHT11

El DHT11 [2] es un sensor digital de temperatura y humedad relativa de bajo costo y fácil uso. Integra un sensor capacitivo de humedad y un termistor para medir el aire circundante, y muestra los datos mediante una señal digital en el pin de datos (no posee salida analógica).

Utilizado en aplicaciones académicas relacionadas al control automático de temperatura, aire acondicionado, monitoreo ambiental en agricultura y más.

### Especificaciones:

- Voltaje de Operación: 3V - 5V DC
- Rango de medición de temperatura: 0 a 50 °C
- Precisión de medición de temperatura: ±2.0 °C
- Resolución Temperatura: 0.1°C
- Rango de medición de humedad: 20% a 90% RH.
- Precisión de medición de humedad: 5% RH.
- Resolución Humedad: 1% RH
- Tiempo de sensado: 1 seg.
- Interface digital: Single-bus (bidireccional)
- Modelo: DHT11
- Dimensiones: 16\*12\*5 mm
- Peso: 1 gr.
- Carcasa de plástico celeste

## Sensor de movimiento PIR-HC-SR501



**Sensor2.** PIR-HC-SR501

Este sensor de movimiento PIR [3] es compatible con las placas de desarrollo o microcontroladores, es de gran utilidad para realizar proyectos de automatización, como encendido de motores, actuadores o luces. Su tamaño es compacto (2.45 cm x 3.26 x 2.4 cm) para una instalación sencilla, puede detectar movimiento hasta una distancia de 4 metros, tiene controles (potenciómetros) para ajuste de temporizador e intensidad de luz y un jumper que te permite configurarlo para pulso simple o continuo.

### Especificaciones:

- Sensor de movimiento
- 4 m de distancia de detección
- Pulso simple o continuo
- Retardo ajustable
- Alimentación: 5 Vcc 70 mA
- Tamaño compacto: 24.5 x 32.6 x 24 mm

## Fotoresistencia LDR



**Sensor3.** Fotoresistencia LDR

Una fotoresistencia o LDR [4] (Light Dependent Resistor, o resistencia dependiente de la luz) es un componente foto electrónico cuya resistencia varía en función



de la luz que incide en él. Esta resistencia es muy baja, de unos pocos  $\Omega$ s con una luz intensa incide en él y va creciendo fuertemente a medida que esa luz decrece.

Se les suele utilizar como sensores de luz, para arrancar luces automáticamente cuando la oscuridad sobrepasa un cierto umbral, o como detectores de movimiento próximo (Cuando algo se interpone). Para poder utilizarlo solo debes agregar una resistencia de 10K en serie y conectarlo a una de las entradas analógicas de tu Arduino (A0-A5) como un divisor de tensión.

#### Especificaciones:

- Sensor: LDR GL5528
- Resistencia en luz (10 lux): 8K-20K Ohm
- Resistencia en oscuridad: 1M Ohm
- Voltaje máx: 150V
- Potencia máx: 100mW
- Material fotosensible: CdS (Sulfato de Sodio)
- Frecuencia de luz pico: 540 nm
- Tamaño de 5mm.

### 3.2 Jerarquía de tópicos y sistema IoT MQTT

MQTT [5] es el protocolo de comunicación Machine to Machine (M2M) que tanta popularidad está alcanzando para la comunicación entre dispositivos de IoT.

Es parte de los protocolos de comunicación para IoT. Ahí vimos los conceptos de pub-sub, message queue, y message service. Usaremos estos conceptos así que, si tenéis dudas, deberías pasáros y echarle un vistazo.

El protocolo MQTT se ha convertido en uno de los principales pilares del IoT por su sencillez y ligereza. Ambos son condicionantes importantes dado que los dispositivos de IoT, a menudo, tienen limitaciones de potencia, consumo, y ancho de banda. Esta es la creación de sistemas IoT

con dispositivos como el ESP8266 o Raspberry Pi.

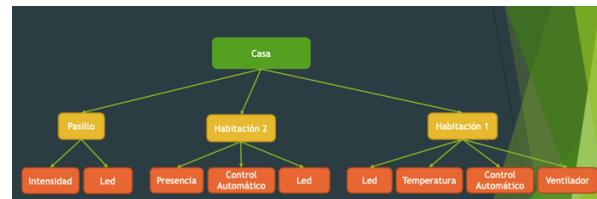


Fig. 3. Jerarquía de MQTT

```
// Credenciales MQTT
const char *mqtt_server = "ioticos.org";
const char *Raiz_Sus = "c0dVYg5FaZ2hHtj";
const char *Raiz_Hab1_Sus = "c0dVYg5FaZ2hHtj/Hab1";
const char *Raiz_Hab1_Led_Sus = "c0dVYg5FaZ2hHtj/Hab1/Led";
const char *Raiz_Hab1_Tem_Sus = "c0dVYg5FaZ2hHtj/Hab1/Tem";
const char *Raiz_Hab1_Vent_Sus = "c0dVYg5FaZ2hHtj/Hab1/Vent";
const char *Raiz_Hab2_Mov_Sus = "c0dVYg5FaZ2hHtj/Hab2/Mov";
const char *Raiz_Pas_Luz_Sus = "c0dVYg5FaZ2hHtj/Pas/Luz";
const char *Raiz_Pas_Bom_Sus = "c0dVYg5FaZ2hHtj/Pas/Bom";
```

Fig. 4 Jerarquía en Script

```
Tópico => c0dVYg5FaZ2hHtj/Hab1/Led Mensaje => 0
Tópico => c0dVYg5FaZ2hHtj/Hab1/Vent Mensaje => 0
Tópico => c0dVYg5FaZ2hHtj/Hab1/Tem Mensaje => 22.80
Tópico => c0dVYg5FaZ2hHtj/Hab2/Mov Mensaje => 1
```

Fig. 5 Jerarquía en IoTicos.org



Fig. 6. Sistema de comunicación

### 4 Implementación

Se pretende implementar un sistema IoT que permita automatizar algunas tareas en el hogar, donde se busca lograr los siguientes objetivos:

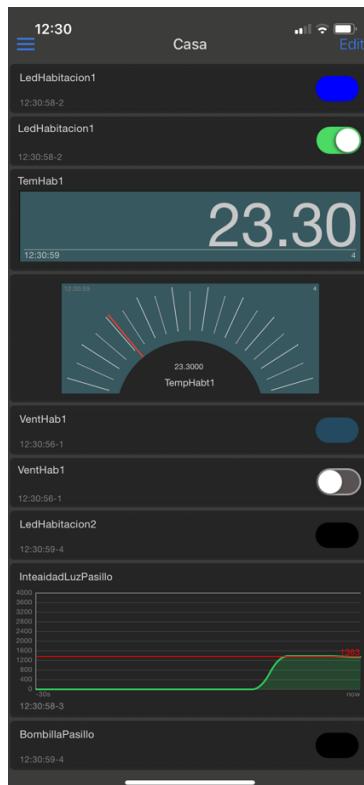
#### 4.1 Controlar remotamente una bombilla led en la habitación 1.

Se programa en la aplicación *IoT OnOff* para poder controlarlo a distancia desde nuestro celular.



En el tópico:  
cOdVYg5FaZ2hHtj/Hab1/Led

Cada vez que se presione el botón “Ledhabitacion1” manda 0 o 1 y con esto prende o apaga el Led de la habitación 1.



App. 1. LedHabitacion1 esta prendido

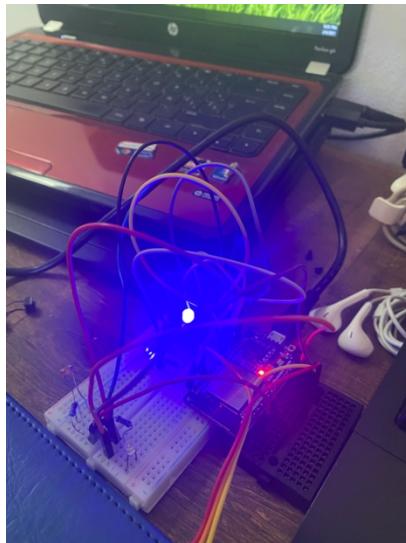
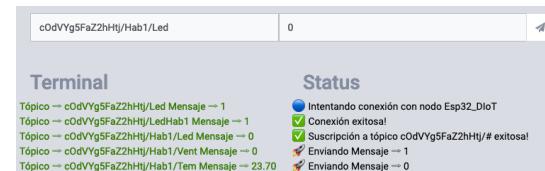


Foto. 1. Led habitación 1 prendido

Se manda directamente 0 o 1 desde [IoTicos.org](http://IoTicos.org) para controlarlo desde la computadora, así prende y apaga el Led.



Desde el monito serial del IDE de Arduino

Mensaje recibido desde -> cOdVYg5FaZ2hHtj/Hab1/Led  
Mensaje -> 0  
Mensaje recibido desde -> cOdVYg5FaZ2hHtj/Hab1/Led  
Mensaje -> 1

#### 4.2 Medir la temperatura de la habitación 1.

Se programa en la aplicación y se suscribe al tópico para poder ver la lectura del sensor de temperatura de la habitación 1.

En el tópico:  
cOdVYg5FaZ2hHtj/Hab1/Tem



App. 2. Lectura de la temperatura



Desde la aplicación se puede apreciar que la temperatura es de 23.30ºC.

Desde *IoTicos.org* se puede apreciar la lectura.

Tópico → cOdVYg5FaZ2hHtj/Hab1/Tem Mensaje → 23.70

Desde el monito serial del IDE de Arduino

Temperatura:  
23.70

#### 4.3 Controlar automáticamente en función de la temperatura o manualmente de forma remota un ventilador de la habitación 1.

Se programa en la aplicación *IoT OnOff* para poder controlarlo a distancia desde nuestro celular. Cada vez que se presione el botón “VentHab1” manda 0 o 1 y con esto prende o apaga el ventilador de la habitación 1.

En el tópico:

cOdVYg5FaZ2hHtj/Hab1/Vent



App. 3. VentHab1 esta prendido

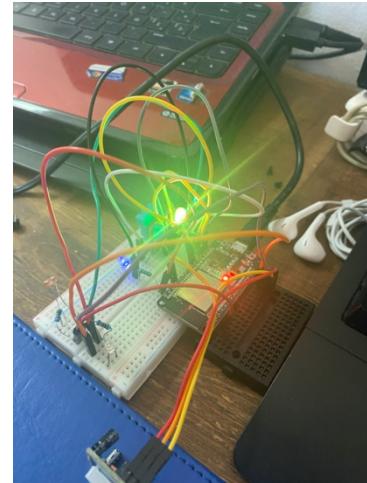


Foto. 3. Ventilador Prendido

Se manda directamente 0 o 1 desde *IoTicos.org* para controlarlo desde la computadora, así prende y apaga el ventilador.



Desde el monito serial del IDE de Arduino

Mensaje recibido desde → cOdVYg5FaZ2hHtj/Hab1/Vent  
Mensaje → 1  
Mensaje recibido desde → cOdVYg5FaZ2hHtj/Hab1/Vent  
Mensaje → 0

#### 4.4 Controlar automáticamente ante la presencia de personas o manualmente de forma remota el encendido y apagado de una bombilla led de la habitación 2.

Se programa en la aplicación y se suscribe al tópico para poder ver la lectura del sensor si detecta movimiento y así identificar si el LedHabitacion2 esta prendido o apagado.

En el tópico:

cOdVYg5FaZ2hHtj/Hab2/Mov

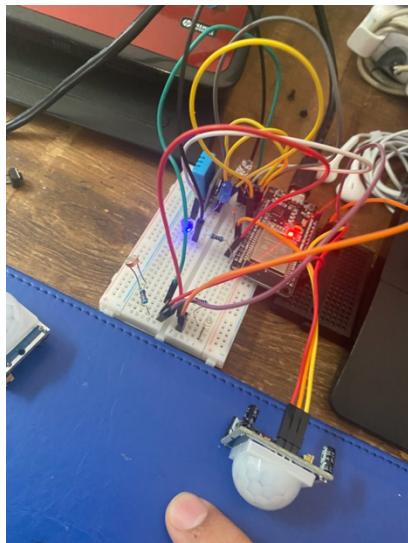


Foto. 4. Led Azul prendido por el movimiento del dedo

Lectura de [IoTicos.org](#) para ver si prende y apaga el Led.

Tópico → cOdVYg5FaZ2hHtj/Hab2/Mov Mensaje → 1  
Tópico → cOdVYg5FaZ2hHtj/Hab2/Mov Mensaje → 0

Desde el monito serial del IDE de Arduino

Hay movimiento  
No hay movimiento

#### 4.5 Medir la intensidad de luz del pasillo.

Se programa en la aplicación y se suscribe al tópico para poder ver la lectura del sensor.

En el tópico:

COdVYg5FaZ2hHtj/Pas/Luz

En las foto 5 y 6 se aprecia la medición de luz de 5000 cuando le pongo luz directamente del flash de mi celular y de 300 cuando es luz ambiente de medio día.

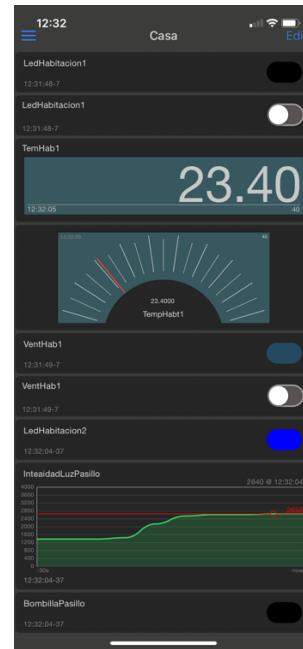


Foto. 5. Medición de luz con luz flash del celular



Foto. 6. Medición de luz con luz ambiente

Se lee desde [IoTicos.org](#) el valor del sensor

Tópico → cOdVYg5FaZ2hHtj/Pas/Luz Mensaje → 1613

Desde el monito serial del IDE de Arduino

Medición Luz:  
1630



#### 4.6 Controlar automáticamente en función de la intensidad de luz una bombilla ubicada en el pasillo.

Se programa en la aplicación y se suscribe al tópico para poder ver la lectura del sensor, si la lectura es menor de 200 este prende la bombilla y si es mayor este apaga la bombilla.

En el tópico:

cOdVYg5FaZ2hHtj/Pas/Bom

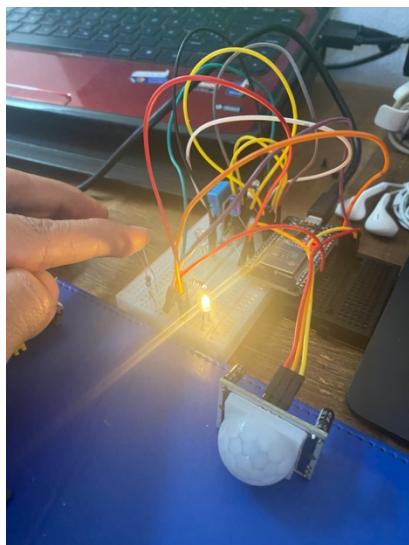


Foto. 7. Bombilla prendida cuando tapo el sensor de luz

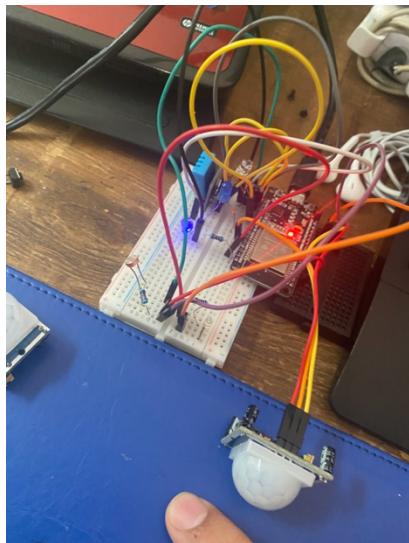


Foto. 8. Bombilla apaga cuando detecta luz ambiente

#### 4. Conclusiones

Es importante entender la programación para poder manipular el código y así darle las funciones adecuadas a los sensores poder publicarlos en el servidor y suscribirnos a este desde nuestro celular para la manipulación y lectura de datos. Este proyecto se concluyó con éxito, usando todos los conocimientos adquiridos en el diplomado.

Se puede adaptar la tarjeta para incorporar nuevos sensores, a si mismo la creación de una aplicación y la adaptación de un servicio MQTT de pago para mejorar la seguridad de nuestra red inteligente, este proyecto es el esqueleto para poder conformar al cuerpo de una red inteligente con muchas mejoras.

#### 5. Bibliografía

##### [1]Data Sheet ESP32

###### ESP32 Series Datasheet

Including:  
ESP32-C0WD-V3  
ESP32-C0WDQ-V3  
ESP32-C0WD  
ESP32-C0WDQ  
ESP32-C2WD  
ESP32-S0WD  
ESP32-U0WDH

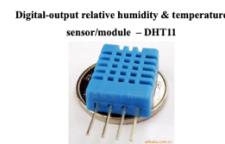


www.espressif.com

Enlace:

[https://www.espressif.com/sites/default/files/documentation/esp32\\_datasheet\\_en.pdf](https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf)

## [2] Data Sheet DHT11



Resistive-type humidity and temperature module/sensor

### 1. Feature & Application:

- Full range temperature compensated
- Resistive type sensor
- Low power consumption
- Outstanding long term stability
- Very compact and low cost
- Long transmission distance
- High resolution
- Low cost
- No external components required
- Interchangeable

### 2. Description:

DHT11 output calibrated digital signal. It utilizes exclusive digital-signal-collecting-technology and humidity sensing technology, ensuring its reliability and stability. Its sensing elements is composed of a single-chip computer.

Every sensor of the model is tested, compensated and calibrated in accurate calibration chamber and the calibration coefficient is stored in OTP memory.

Small size & low consumption & long transmission distance(20m) enable DHT11 to be suited in all kinds of harsh application occasions. Single row package with four pins, making the connection very convenient.

Enlace:

[http://image.dfrobot.com/image/data/DFR0067/DFR0067\\_DS\\_10\\_en.pdf](http://image.dfrobot.com/image/data/DFR0067/DFR0067_DS_10_en.pdf)

## [3] Data Sheet PIR-HC-SR501

Sensor infrarrojo de movimiento PIR  
HC-SR501

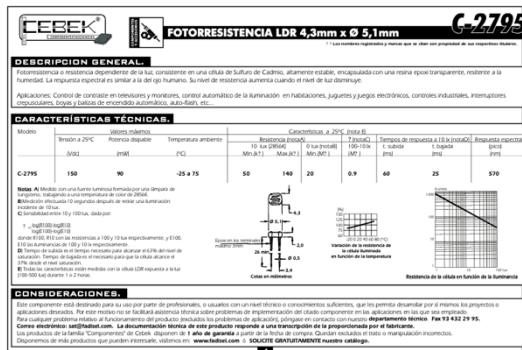


Indice:  
1. Propósito de Instrumentación.  
2. Descripción del HC-SR501.  
3. Ajustes y configuración del sensor.  
4. Pruebas preliminares del módulo HC-SR501.

Enlace:

<https://www.puntoflotante.net/MANUAL-DEL-USUARIO-SENSOR-DE-MOVIMIENTO-PIR-HC-SR501.pdf>

## [4] Data sheet Fotoresistencia LDR



Enlace:

<https://www.electan.com/datasheets/cebek/CE-C2795.pdf>

## [5] MQTT

<https://www.luisllamas.es/que-es-mqtt-su-importancia-como-protocolo-iot/>

## [6] Script

//Proyecto Final

//Michel A Ramos Soto

//Diplomado 2021 //192.168.1.78

```
#include <DHT.h>
#include <WiFi.h>
#include <PubSubClient.h>
const int mqtt_port = 1883;
const char *mqtt_user = "FnJ939HUTm6Hr2F";
const char *mqtt_pass = "2EzPNpWjedWT8P";
```

## // Credenciales MQTT

```
const char *mqtt_server = "ioticos.org";
const char *Raiz_Sus = "cOdVYg5FaZ2hHtj";
const char *Raiz_Hab1_Sus = "cOdVYg5FaZ2hHtj/Hab1";
const char *Raiz_Hab1_Led_Sus = "cOdVYg5FaZ2hHtj/Hab1/Led";
const char *Raiz_Hab1_Tem_Sus = "cOdVYg5FaZ2hHtj/Hab1/Tem";
const char *Raiz_Hab1_Vent_Sus = "cOdVYg5FaZ2hHtj/Hab1/Vent";
const char *Raiz_Hab2_Mov_Sus = "cOdVYg5FaZ2hHtj/Hab2/Mov";
const char *Raiz_Pas_Luz_Sus = "cOdVYg5FaZ2hHtj/Pas/Luz";
const char *Raiz_Pas_Bom_Sus = "cOdVYg5FaZ2hHtj/Pas/Bom";
```

## // Credenciales de WiFi

```
const char* ssid = "INFINITUM3992";
const char* password = "zDP5aBLn7J";
```

## // Variables Globales



```
WiFiClient espClient;
PubSubClient client(espClient);
char msg[25];
char msg2[25];
char msg3[25];
char msg4[25];
long count=0;
int contconexion = 0;

// Funciones
void callback(char* topic, byte* payload,
unsigned int length);
void reconnect();
void setup_wifi();

//Sensor de temperatura
const int P_Sensor=4;
DHT DHT1(P_Sensor, DHT11); // Crea
dht1, que es un sensor DHT11 conectado

//Sensor de movimiento
const int M_Sensor = 21; // Asigna el pulsador al pin 4
const int MLed = 23;
bool Mov = LOW; // Variable para guardar el estado del pin 2

//Sensor Luz
const int P_Ldr = 39;
int AdcLuz=0;

void setup() {
  Serial.begin(115200);
  DHT1.begin(); // Inicializa el sensor
  DHT11
  pinMode(M_Sensor, INPUT);
  pinMode(MLed, OUTPUT);
  pinMode(18, OUTPUT); //Led Habitacion 1
  digitalWrite(18, LOW);
  pinMode(15, OUTPUT); //Ventilador Habitacion 1
  digitalWrite(15, LOW);

pinMode(22, OUTPUT); //Bombilla Pasillo
digitalWrite(22, LOW);
// Configuración conexión WIFI
setup_wifi();
// Configuración conexión MQTT
client.setServer(mqtt_server,
mqtt_port);
client.setCallback(callback);
}

void loop() {
  LeerDHT1(); // Ejecuta la función LeerDTH1
  // Se pregunta si el cliente está conectado
  if (!client.connected()) {
    // Si no está conectado se realiza la reconexión
    reconnect();
  }

  if (client.connected()){

    //Temperatura
    float t1=DHT1.readTemperature(); // Lee la temperatura en la variable t1
    float h1=DHT1.readHumidity(); // Lee la humedad en la variable h1
    while (isnan(t1) || isnan(h1)){ // Verifica que la lectura no sea fallida
      Serial.println("Lectura fallida");// Mensaje de lectura fallida
      delay(1000); // Retardo de 1000 msegundos
      t1=DHT1.readTemperature(); // Lee nuevamente la temperatura
      h1=DHT1.readHumidity(); // Lee nuevamente la humedad
    }
    Serial.println("Temperatura:"); // Imprime Mensaje
    Serial.println(t1); // Imprimera valor de temperatura
  }
}
```

```

Serial.println("Humedad:");           // BombillaPrendida = String(BP);
Imprime mensaje
Serial.println(h1);

String Tem = String(count);
Tem = String(t1);
Tem.toCharArray(msg,25);

client.publish(Raiz_Hab1_Tem_Sus,msg);

count++;

}

// Movimiento
Mov = digitalRead(M_Sensor);
if (Mov==HIGH){
Serial.println("Hay movimiento");
digitalWrite(MLed, HIGH);
}
else{
Serial.println("No hay movimiento");
digitalWrite(MLed, LOW);
}
String Movimiento = String(count);
Movimiento = String(Mov);
Movimiento.toCharArray(msg2,25);

client.publish(Raiz_Hab2_Mov_Sus,msg2
);

//Medicion de Luz
AdcLuz = analogRead(P_Ldr); // Mide el Adc y lo guarda AdcLuz
Serial.println("Medición      Luz:");// Imprime mensaje
Serial.println(AdcLuz); // Imprime valor
//Bombilla del pasillo
if (AdcLuz<300){
Serial.println("Prende Bombilla");
digitalWrite(22, HIGH);
int BP = 0;
String BombillaPrendida = String(count);
}

client.publish(Raiz_Pas_Bom_Sus,msg4);

client.publish(Raiz_Pas_Bom_Sus,msg4);
}

else{
Serial.println("Apaga Bombilla");
digitalWrite(22, LOW);
int BA = 1;
String BombillaApagada = String(count);
BombillaApagada = String(BA);

BombillaApagada.toCharArray(msg4,25);

client.publish(Raiz_Pas_Bom_Sus,msg4);
}

//Publish
String luz = String(count);
luz = String(AdcLuz);
luz.toCharArray(msg3,25);
client.publish(Raiz_Pas_Luz_Sus,msg3);

client.loop();
delay(1000);
}

void LeerDHT1(){
float t1=DHT1.readTemperature(); // Lee la temperatura en la variable t1
float h1=DHT1.readHumidity(); // Lee la humedad en la variable h1
while (isnan(t1) || isnan(h1)){ // Verifica que la lectura no sea fallida
Serial.println("Lectura      fallida");// Mensaje de lectura fallida
delay(1000); // Retardo de 1000 msegundos
t1=DHT1.readTemperature(); // Lee nuevamente la temperatura
h1=DHT1.readHumidity(); // Lee nuevamente la humedad
}
}

```

```

Serial.println("Temperatura:");           // Se crea un cliente ID
Imprime Mensaje                         String clientId = "DIoT_Cliente";
                                         clientId += String(random(0xffff), HEX);
                                         // Se intenta realizar la conexión
                                         conectar
                                         if
                                         (client.connect(clientId.c_str(),mqtt_user
                                         ,mqtt_pass)) {
                                         // Si hay conexión se imprime el
                                         mensaje
                                         Serial.println("Conectado con el
                                         Broker!");
                                         // Nos suscribimos al tópico Raíz/Piso3
                                         if(client.subscribe(Raiz_Sus)){
                                         Serial.print("Suscripcion --");
                                         Serial.println(Raiz_Sus);
                                         Serial.println(" -- ok");
                                         }else{
                                         Serial.println("fallo Suscripción a --");
                                         Serial.println(Raiz_Sus);
                                         }
                                         // Nos suscribimos al tópico
                                         Raíz/Habitacion1/Vent

                                         if(client.subscribe(Raiz_Hab1_Vent_Sus))
                                         {
                                         Serial.print("Suscripcion --");
                                         Serial.println(Raiz_Hab1_Vent_Sus);
                                         Serial.println(" -- ok");
                                         }else{
                                         Serial.println("fallo Suscripción a --");
                                         Serial.println(Raiz_Hab1_Vent_Sus);
                                         }
                                         // Nos suscribimos al tópico
                                         Raíz/Habitacion1/Led

                                         if(client.subscribe(Raiz_Hab1_Led_Sus)){
                                         Serial.print("Suscripcion --");
                                         Serial.println(Raiz_Hab1_Led_Sus);
                                         Serial.println(" -- ok");
                                         }else{
                                         Serial.println("fallo Suscripción a --");
                                         Serial.println(Raiz_Hab1_Led_Sus);
                                         }

Serial.println("Temperatura:");           // Se crea un cliente ID
Imprime Mensaje                         String clientId = "DIoT_Cliente";
                                         clientId += String(random(0xffff), HEX);
                                         // Se intenta realizar la conexión
                                         conectar
                                         if
                                         (client.connect(clientId.c_str(),mqtt_user
                                         ,mqtt_pass)) {
                                         // Si hay conexión se imprime el
                                         mensaje
                                         Serial.println("Conectado con el
                                         Broker!");
                                         // Nos suscribimos al tópico Raíz/Piso3
                                         if(client.subscribe(Raiz_Sus)){
                                         Serial.print("Suscripcion --");
                                         Serial.println(Raiz_Sus);
                                         Serial.println(" -- ok");
                                         }
                                         // Nos suscribimos al tópico
                                         Raíz/Habitacion1/Vent

                                         if(client.subscribe(Raiz_Hab1_Vent_Sus))
                                         {
                                         Serial.print("Suscripcion --");
                                         Serial.println(Raiz_Hab1_Vent_Sus);
                                         Serial.println(" -- ok");
                                         }else{
                                         Serial.println("fallo Suscripción a --");
                                         Serial.println(Raiz_Hab1_Vent_Sus);
                                         }
                                         // Nos suscribimos al tópico
                                         Raíz/Habitacion1/Led

                                         if(client.subscribe(Raiz_Hab1_Led_Sus)){
                                         Serial.print("Suscripcion --");
                                         Serial.println(Raiz_Hab1_Led_Sus);
                                         Serial.println(" -- ok");
                                         }else{
                                         Serial.println("fallo Suscripción a --");
                                         Serial.println(Raiz_Hab1_Led_Sus);
                                         }

//Conexión WiFi
void setup_wifi(){
  delay(10);
  delay(10);
  // Conexión WIFI
  Serial.println();
  Serial.print("Conectando a ssid: ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED
        and contconexion <50) {
    ++contconexion;
    delay(500);
    Serial.print(".");
  }
  if (contconexion <50) {
    Serial.println("");
    Serial.println("WiFi conectado");
    Serial.println(WiFi.localIP());
  }
  else {
    Serial.println("");
    Serial.println("Error de conexión");
  }
}

// Conexión MQTT
void reconnect() {
  // Si no hay conexión se entra dentro del
  ciclo
  while (!client.connected()) {
    Serial.println("Intentando conexión
Mqtt...");
  }
}

```



```
    } else {
        // Si no hay conexión se imprime el
        // mensaje, el estado de conexión
        // y se intenta nuevamente dentro de
        5 segundo
        Serial.print("falló :( con error -> ");
        Serial.print(client.state());
        Serial.println(" Intentamos de nuevo
en 5 segundos");
        delay(5000);
    }
}

// Función para cuando se recibe un
// mensaje por parte del broker de un tópico
// al
// cual se está suscrito
void callback(char* topic, byte* payload,
unsigned int length){
    String incoming = "";
    Serial.print("Mensaje recibido desde ->
");
    Serial.print(topic);
    Serial.println("");
    for (int i = 0; i < length; i++) {
        incoming += (char)payload[i];
    }
    incoming.trim();
    Serial.println("Mensaje -> " + incoming);

    if
(strcmp(topic,Raiz_Hab1_Vent_Sus)==0)
{
    int VentHab1 = incoming.toInt();
    if(VentHab1==1)
    {
        digitalWrite(15, HIGH);
    }
    else
    {
        digitalWrite(15, LOW);
    }
}

/*Serial.print("Mensaje recibido desde -> ");
Serial.print(topic);
Serial.println("");
for (int i = 0; i < length; i++) {
    incoming += (char)payload[i];
}
incoming.trim();
Serial.println("Mensaje -> " + incoming);*/
```

```
(strcmp(topic,Raiz_Hab1_Led_Sus)==0)
{
    int LedHab1 = incoming.toInt();
    if(LedHab1==1)
    {
        digitalWrite(18, HIGH);
    }
    else
    {
        digitalWrite(18, LOW);
    }
}

/*if
(strcmp(topic,Raiz_Hab2_Mov_Sus)==0){
    Movimiento = digitalRead(P_Sensor);
    if (Movimiento==HIGH){
        Serial.println("Hay movimiento");
        digitalWrite(Led, HIGH);
    }
    else{
        Serial.println("No hay movimiento");
        digitalWrite(Led, LOW);
    }
}*/
```