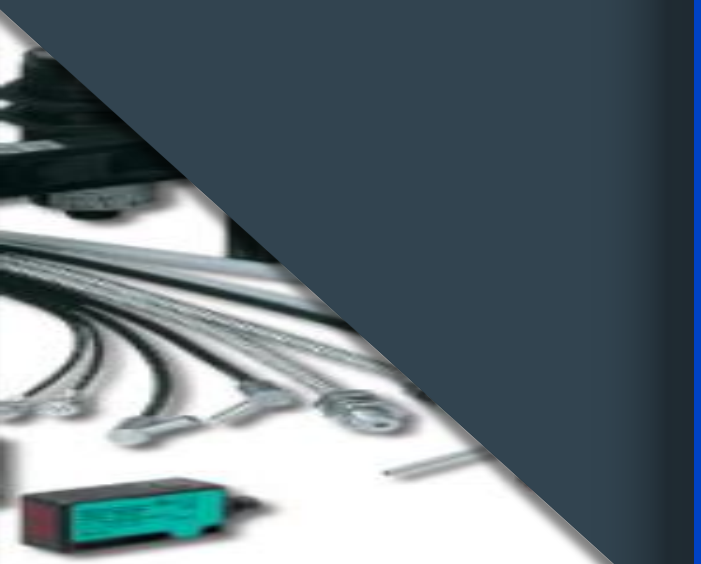
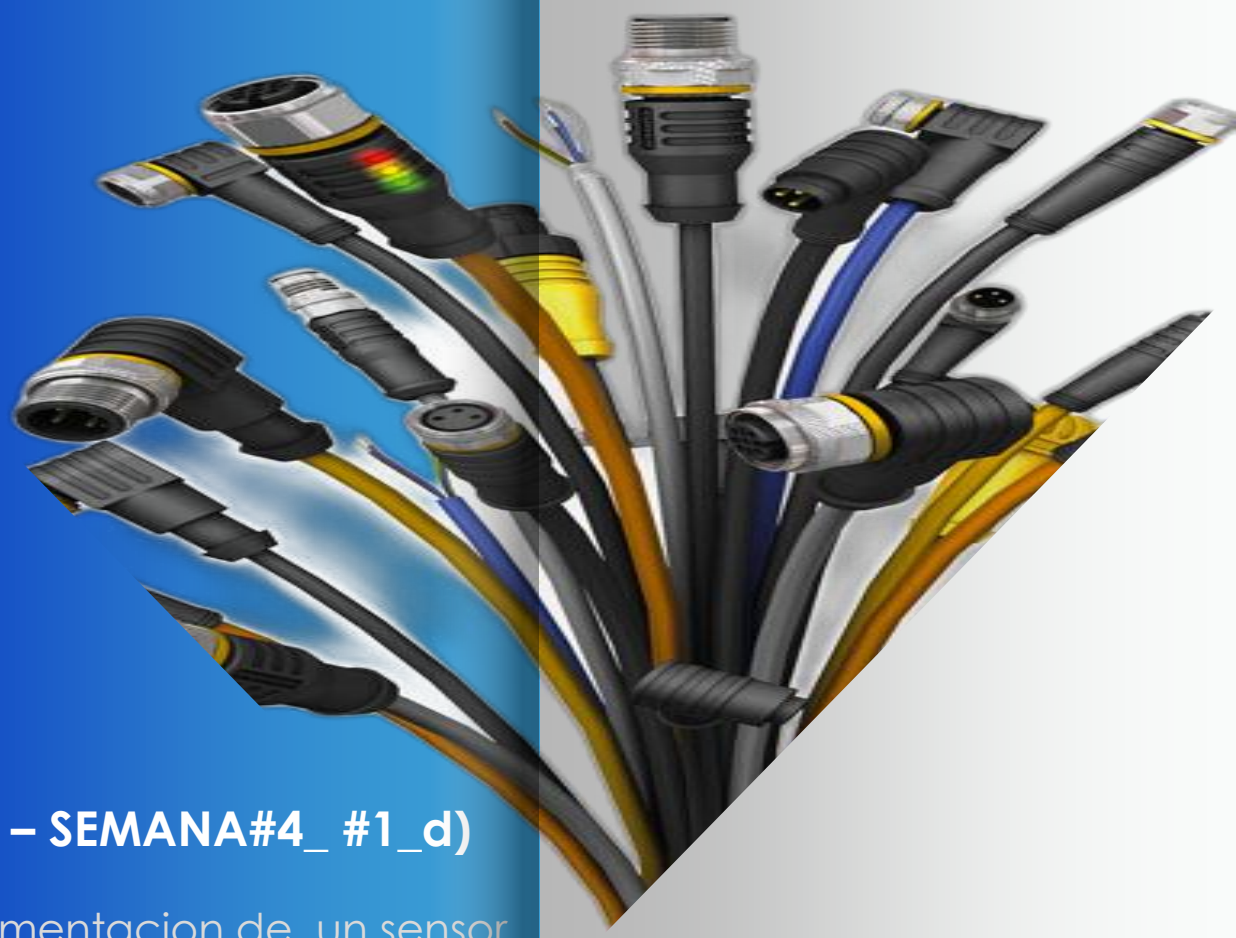
GRUPO 1

Sensores y Actuadores





TP #4 – SEMANA#4_ #1_d)

Implementacion de un sensor inteligente de temperatura

Alumno:

- Miguel Angel Segnana

ISPC Cordoba

INDICE

Alumno: Miguel A. Segnana - TST - SyA : TP#4

Sensor Inteligente:

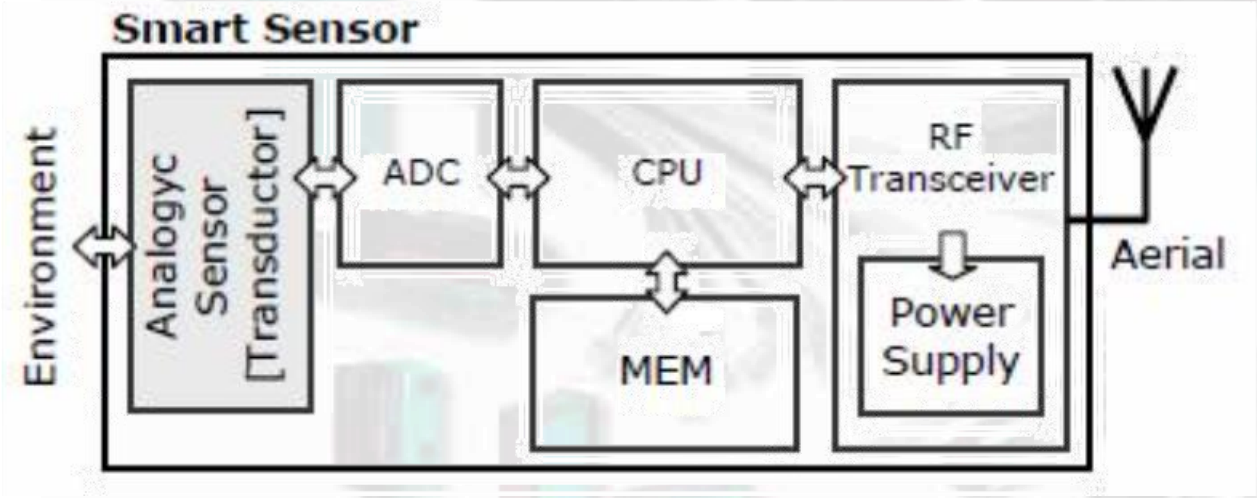
Un sensor inteligente es un dispositivo que toma datos del entorno físico y utiliza recursos informáticos integrados para realizar funciones predefinidas al detectar una entrada específica y luego procesar los datos antes de transmitirlos.
Los sensores inteligentes permiten una recopilación más precisa y automatizada de datos ambientales con menos ruido erróneo entre la información registrada con precisión. Estos dispositivos se utilizan para monitorear y controlar los mecanismos en una amplia variedad de entornos que incluyen redes inteligentes, reconocimiento del campo de batalla, exploración y una gran cantidad de aplicaciones científicas.

El sensor inteligente también es un elemento crucial e integral en internet de las cosas (IoT), el entorno cada vez más prevalente en el que casi cualquier cosa imaginable puede equiparse con un identificador único (UID) y la capacidad de transmitir datos a través de internet o una red similar.
Una implementación de los sensores inteligentes es como componentes de una red inalámbrica de sensores y actuadores (WSAN) cuyos nodos pueden contarse por miles, cada uno de los cuales está conectado con uno o más sensores y concentradores de sensores, así como con actuadores individuales.
Los recursos informáticos suelen ser proporcionados por microprocesadores móviles de baja potencia. Como mínimo, un sensor inteligente está hecho de un sensor, un microprocesador y tecnología de comunicación de algún tipo. Los recursos informáticos deben ser una parte integral del diseño físico: un sensor que simplemente envía sus datos para su procesamiento remoto no se considera un sensor inteligente.

Alumno: Miguel A. Segnana - TST - SyA : TP#4

Estructura:

Un sensor inteligente también puede incluir otros componentes además del sensor primario. Estos componentes pueden incluir transductores, amplificadores, control de excitación, filtros analógicos y compensación. Un sensor inteligente también incorpora elementos definidos por software que proporcionan funciones como conversión de datos, procesamiento digital y comunicación a dispositivos externos.
Un esquema básico seria como el de la figura:



Alumno: Miguel A. Segnana - TST - SyA : TP#4

Beneficios del uso de sensores inteligentes :

La industria de manufactura ha adoptado paulatinamente las nuevas tecnologías relacionadas con la conectividad y la recopilación de datos, pero esto ha empezado a cambiar con la introducción de la industria 4.0. Las empresas tienen que prepararse para la digitalización de las fábricas y el uso del análisis de datos, la inteligencia artificial y el Internet de las cosas.

Un caso de esta transición son los sensores inteligentes, que pueden apoyar la optimización del rendimiento de la maquinaria de manufactura para convertirla en dispositivos inteligentes, capaces de conectarse a redes inteligentes a lo largo de toda la cadena de valor.

Los sensores inteligentes pueden mejorar las operaciones y aumentar la eficiencia de la producción, permitiendo a los fabricantes la oportunidad de mejorar sus operaciones, pero ¿cuáles son los beneficios que pueden aportar los sensores inteligentes a la industria de manufactura?

1. Operaciones respaldadas con datos

Los sensores inteligentes generan datos mediante la conexión de diferentes dispositivos y sistemas, lo que permite generar una conectividad perfecta en todas las instalaciones de una planta, lo que a su vez ofrece a los fabricantes:

- Supervisar el rendimiento de los equipos y sistemas.
- Agrupar todos los datos generados.
- Crear indicadores, comparar y análisis de conjuntos de datos.

Alumno: Miguel A. Segnana - TST - SyA : TP#4

2. Prevención de fallas y protocolos de mantenimiento

Los sensores inteligentes permiten que los fabricantes reduzcan el Valor de Reposición de Activos (RAV) a través de la reducción del mantenimiento programado innecesario, costos de reemplazos de piezas y el tiempo potencial de inactividad. La tecnología inteligente facilita la transición del mantenimiento programado a mantenimiento predictivo, lo que se traduce en ahorros y aumento de la eficiencia general de la empresa.

Los datos pueden detectar patrones, lo que pronostica la necesidad de realizar mantenimiento a los equipos. Los sensores inteligentes pueden utilizar estos datos para enviar alertas a los usuarios para informarles acerca de problemas potenciales, de modo que puedan prevenirse antes de convertirse en puntos de fallas.

3. Registros de rendimiento y cumplimiento de normativas

Los fabricantes deben cumplir con reglamentaciones medioambientales, como las normas y reglamentaciones OSHA, ISO y Energy Star que acatan las plantas manufactureras. Por lo que muchos fabricantes deben generar informes para demostrar su cumplimiento. Estos informes pueden incluir datos y registros históricos, implicando en ocasiones mucho tiempo para cotejarlos de sistemas fragmentados o incluso inexactos.

La instalación de sensores inteligentes en equipos de manufactura puede mejorar la eficiencia y precisión de la información, ya que registran automáticamente datos como el consumo de energía, la temperatura, la humedad, las horas de funcionamiento, el mantenimiento y los resultados de las líneas de producción.

4. Supervisión del proceso de producción

Los sensores inteligentes no solo ayudan al usuario con el cumplimiento de normativas, también mejoran los procesos de producción a través de la identificación de anomalías en el sistema que podrían afectar la producción o la calidad de los productos y proporcionan notificaciones en tiempo real de estos problemas.

Los fabricantes pueden ser proactivos en lugar de reactivos a la hora de resolver problemas, evitando periodos de inactividad en la cadena de producción y consiguiendo ventajas competitivas, como mejorar los servicios, aumentar la disponibilidad y mejorar la satisfacción del cliente.

5. Capacidad de respuesta a las condiciones de mercado

Con el conocimiento de las demandas de los clientes, los fabricantes pueden ser más receptivos y ampliar sus empresas más fácilmente para garantiza que la productividad conduzca siempre a la rentabilidad.

En este sentido, los sensores inteligentes ofrecen la oportunidad de adoptar metodologías ágiles para realizar cambios en tiempo real a los procesos que pueden aumentar la producción y con base en los datos que generan pueden incrementar la transparencia de rendimiento a través de representaciones visuales del desempeño de las instalaciones.

Los sensores inteligentes y la digitalización de la manufactura permitirán que los negocios produzcan de una manera más transparente, eficiente y de mayor calidad. Los fabricantes tendrán mayor cumplimiento y serán más rentables como resultado de una mayor precisión en los procesos y operaciones en las instalaciones.

Implementación de un Sensor Inteligente de Temperatura

Componentes principales:

- Sensor de Temperatura: DS18B20
- Microcontrolador: ESP8266
- Plataforma IoT : ThingSpeak

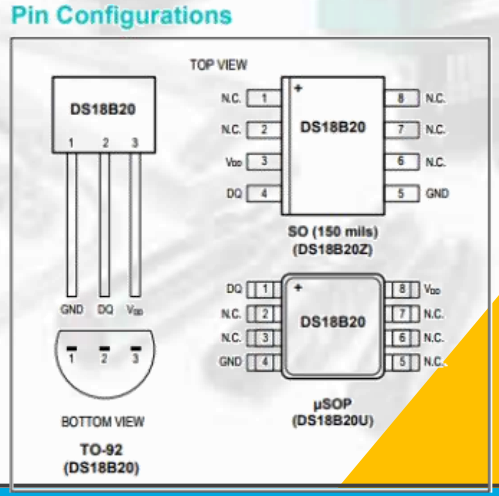
Descripción del proyecto:

- Componentes
- Diagrama en bloques
- Funcionalidades

Alumno: Miguel A. Segnana - TST - SyA : TP#4

Sensor de Temperatura: DS18B20

El DS18B20 es un sensor digital de temperatura que utiliza el protocolo 1-Wire para comunicarse, este protocolo necesita solo un pin de datos para comunicarse y permite conectar más de un sensor en el mismo bus. El rango de alimentación va de 3.3 V a 5 V, por lo que se puede alimentar directamente a través de cualquier placa de desarrollo.
El sensor DS18B20 es fabricado por Maxim Integrated, el encapsulado de fabrica es tipo TO-92 similar al empleado en transistores pequeños. La presentación comercial más utilizada por conveniencia y robustez es la del sensor dentro de un tubo de acero inoxidable resistente al agua, con el que trabajemos este tutorial.
Con este sensor podemos medir temperatura desde los -55°C hasta los 125°C y con una resolución programable desde 9 bits hasta 12 bits, logrando una precisión de +-0,5ºC
Cada sensor tiene una dirección única de 64bits establecida de fábrica, esta dirección sirve para identificar al dispositivo con el que se está comunicando, puesto que en un bus 1-wire pueden existir más de un dispositivo.

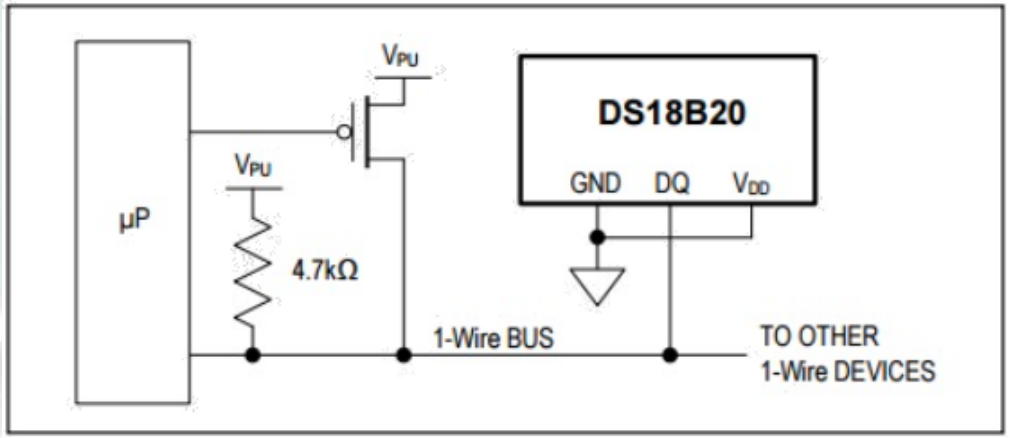


Alumno: Miguel A. Segnana - TST - SyA : TP#4

Métodos de alimentación:

<u>Alimentación a través del pin de datos</u>:

De esto forma, el sensor internamente obtiene energía del pin de datos cuando este se encuentra en un estado alto y almacena carga en un condensador para cuando la línea de datos esté en una estado bajo, a esta forma de obtener energía se le llama “Parasite Power” y se usa cuando el sensor debe conectarse a grandes distancias o en donde el espacio es limitado, puesto que de esta forma no se necesita la línea de VDD. El diagrama para su conexión debe ser de la siguiente forma:

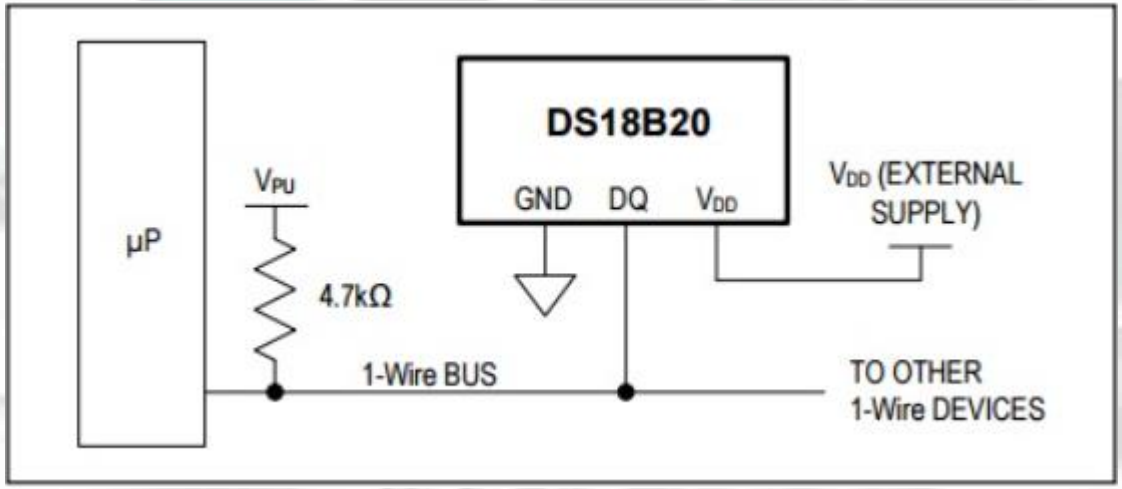


Notar que el pin GND y VDD están ambos conectados a GND, esto es indispensable para que se active el Parasite Power . EL MOSFET en la imagen es necesario para cuando se realicen conversiones de temperatura o copiar datos desde la memoria de circuito de la EEPROM, en estas operaciones la corriente de operación aumenta y si solo se suministra energía a través de la resistencia pueden causar caídas de voltaje en el condensador interno.

Alimentación usando una fuente externa:

De esta forma el sensor se alimenta a través del pin VDD, de esta forma el voltaje es estable e independiente del tráfico del bus 1-wire.

El diagrama de conexión es de la siguiente forma:



Esta forma de alimentación es la más recomendada

Para temperaturas entre -10ºC y 85ºC podemos tener ±0,5ºC. Para el resto de temperaturas entre -55ºC y 125ºC el error es de ±2ºC
Esto equivale a decir que si el sensor DS18B20 suministra una temperatura de 23ºC el valor real estará entre 22,5ºC y 23,5ºC. Si por el contrario suministra un valor de 90ºC el valor real estará entre 88ºC y 92ºC.

Resolución del sensor de temperatura DS18B20

Una de las características más interesantes de este sensor es que podemos trabajar con diferentes resoluciones. Cuando se menciona resolución se refiere a cual es la variación mínima que podemos medir entre dos temperaturas.
El DS18B20 admite resoluciones de 9-bit, 10-bit, 11-bit y 12-bit. Por defecto utiliza la resolución de 12-bit. Las variaciones para cada resolución las puedes consultar en la siguiente tabla:

RESOLUCIÓN	TEMPERATURA
9-bit	0,5ºC
10-bit	0,25ºC
11-bit	0,125ºC
12-bit	0,0625ºC

Alumno: Miguel A. Segnana - TST - SyA : TP#4

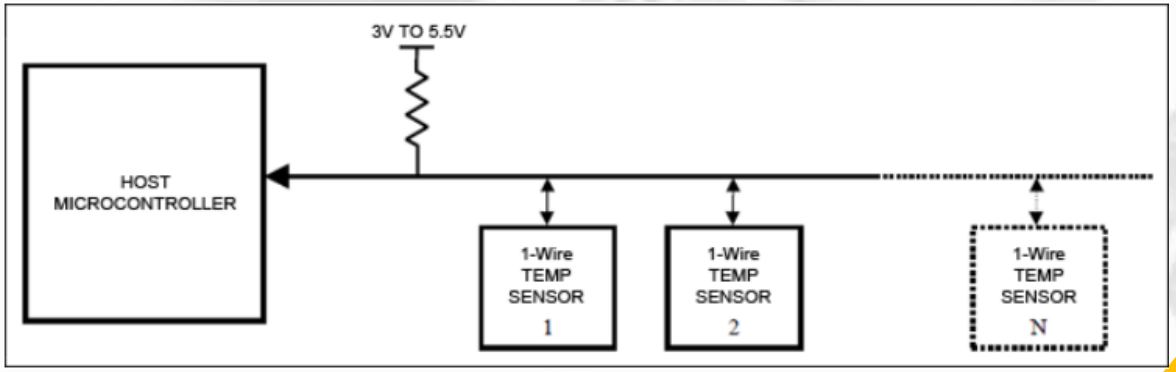
Otras características del sensor de temperatura DS18B20

Además de medir la temperatura, el DS18B20 incorpora una memoria de 64-bit (equivalente a 8 bytes) para almacenar el identificador o dirección única de cada sensor.
El primer byte identifica el tipo de componente. Por ejemplo para los DS18B20 es el número 28 en hexadecimal.

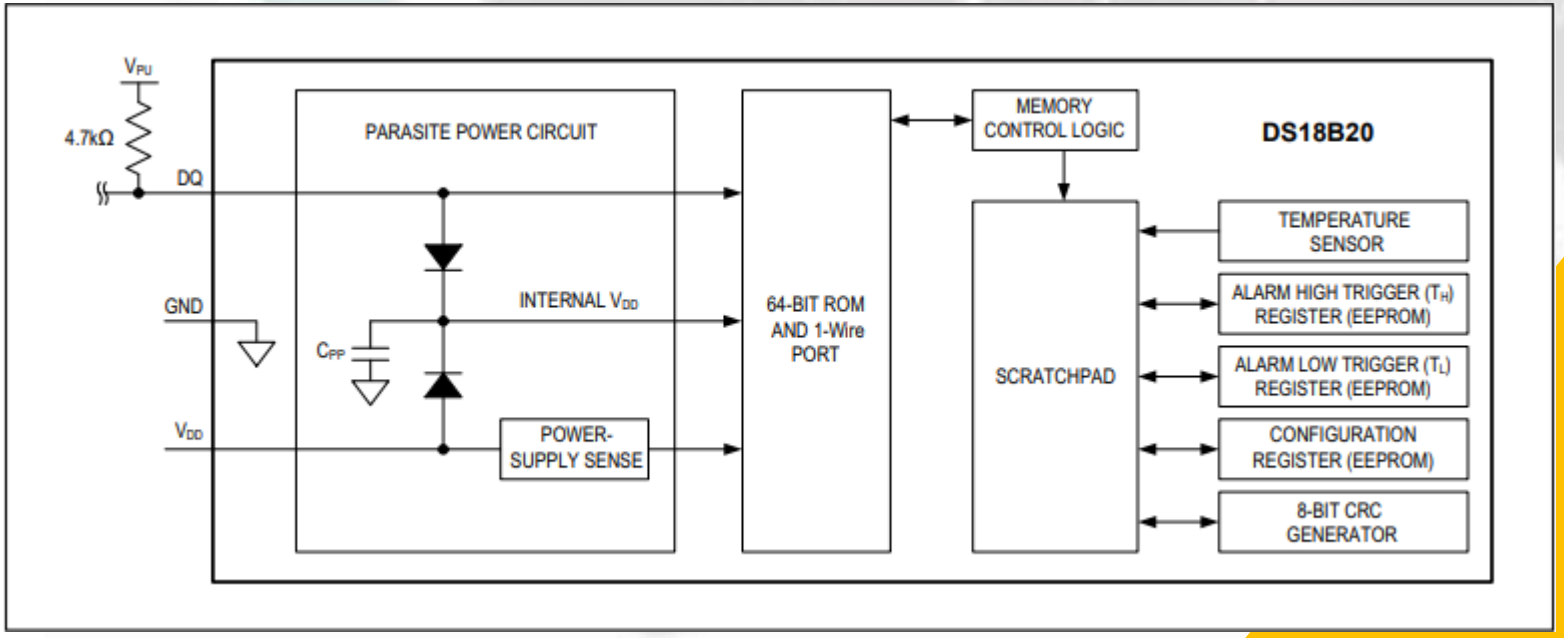
Esta dirección única es necesaria dentro del bus 1-Wire para identificar cada uno de los sensores de temperatura DS18B20 conectados al bus de comunicación.

Gracias a que utiliza este tipo de comunicaciones, se consiguen dos cosas:

- Por un lado robustez en la transmisión de los datos ya que trabaja con datos digitales, mucho menos sensibles a los efectos adversos del ruido que las señales analógicas.
- Por otro lado permite conectar muchos sensores de temperatura con un único pin digital.



Internamente tiene otro tipo de memoria que sirve para diferentes cosas. Utiliza el sistema de verificación de redundancia cíclica CRC para la detección de errores en los datos. El código CRC se almacena en la memoria.
También almacena la temperatura obtenida y dispone de dos alarmas que se disparan si la temperatura es mayor o menor que un umbral de temperatura máxima o temperatura mínima.
Con todas estas características, el DS18B20 se convierte en un sensor muy potente con unas capacidades superiores a otros en el mismo rango de precios.



Vemos en el esquema que hay que conectar Vcc y el pin de datos mediante una resistencia (Pull-up) de 4,7 kΩ. Esto es necesario debido a que, si no se reciben datos, se necesita tener Vcc (HIGH) en el pin de señal.

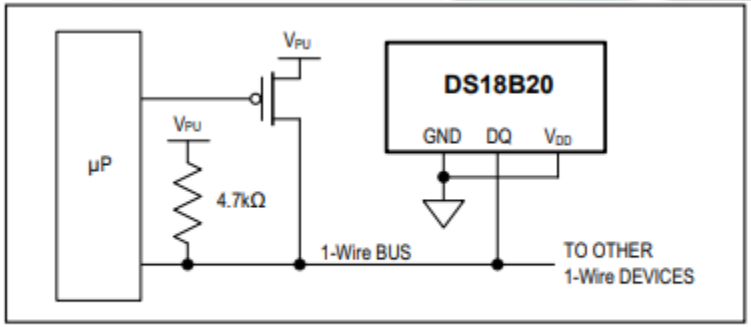


Figure 6. Supplying the Parasite-Powered DS18B20 During Temperature Conversions

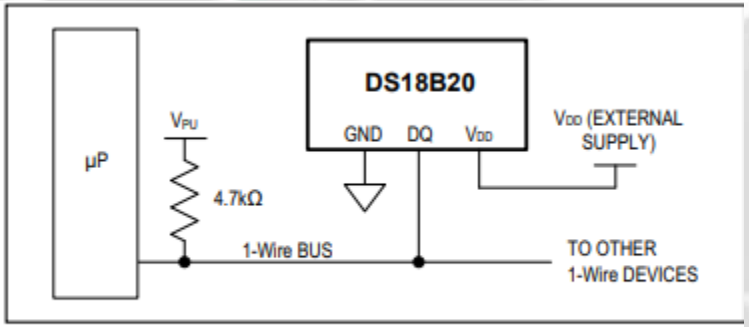


Figure 7. Powering the DS18B20 with an External Supply

ESP8266

NodeMCU incorpora un circuito CH340G, con lo cual es posible comunicarse durante la fase de desarrollo con el puerto serial del microcontrolador desde el puerto USB de la PC y usando un cable MicroUSB.
El NodeMCU integra el protocolo TCP-IP y tiene conectividad Wifi,, con antena de radiofrecuencia integrada, lo cual permite su conexión a internet a través de un router cercano al módulo (digamos hasta unos 3 metros), con la posibilidad de control remoto del tipo "Anywhere", desde cualquier parte del mundo. Esto significa que, ya sea a través de un teléfono celular ó una computadora conectados a internet, se pueden enviar comandos al NodeMCU para activar dispositivos y recibir información de sensores ó alarmas instaladas en casas y oficinas.
El NodeMCU ESP8266, en su versión ESP-12E, V3, es una plataforma completa de desarrollo, módulo SoC (system on chip), basado en el microcontrolador Tensílica L106 de 32 bits, funcionando a 80 Mhz, con una memoria Flash de 4MB, y 32 KB de SRAM. Tiene conectividad Wifi, estándar 802.11 b/g/n, para la implementación de dispositivos operando como servidores de internet (web server).

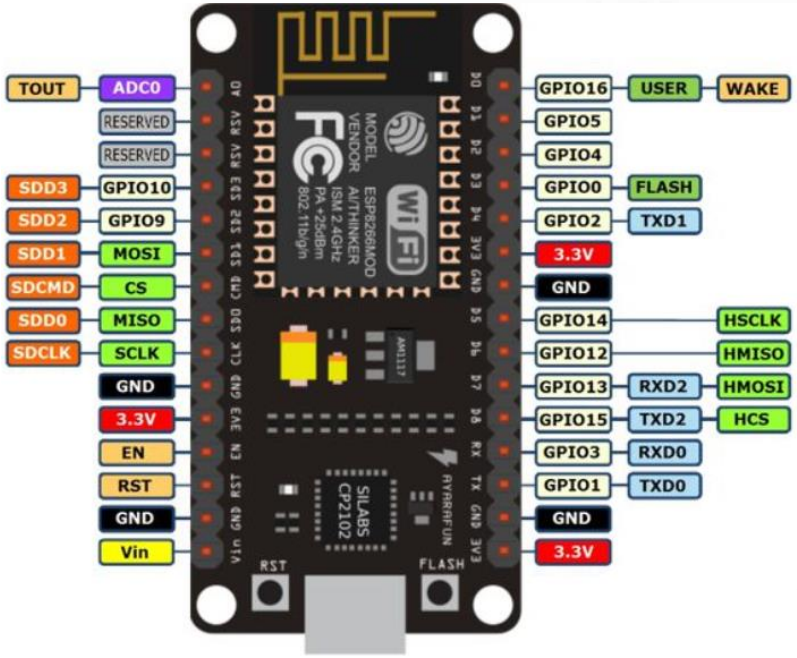
Durante la fase de desarrollo, NodeMCU se conecta directamente a la computadora con un cable microUSB, el cual permite la carga de nuevos programas a una velocidad de hasta 921,600 bps, y proporciona la alimentación de 5v al módulo. Integra una interfaz USB-Serial CH340G (cuyo driver generalmente se carga en modo plug and play en Windows 10).

NodeMCU cuenta con 13 pines de entrada/salida, llamados GPIO (General Port Input Output), y puede funcionar de manera totalmente autónoma como servidor web y recibir comandos remotos para operar los pines mencionados, ya sea para activar dispositivos ó enviar status de sensores digitales. Incluye también pines con las funciones PWM, I2C, SPI, UART, y un convertidor A/D de 10 bits.

Su voltaje de alimentación es de 5v, el cual es reducido a 3.3v por medio de un regulador on board. El consumo promedio es de 100 ma. La corriente aumenta durante la trasmisión y recepción Wifi, de tal manera que, si se alimenta con fuente externa, se recomienda una capacidad de 300 ma. También puede alimentarse directamente a través del conector microUSB.

Los lenguajes de programación más usados para NodeMCU, son MicroPython, Lua y Arduino. Todos son de código abierto y cuentan con herramientas de desarrollo gratuitas.

ESP8266



Pinout del ESP8266

Especificaciones técnicas:

- Modelo: ESP8266, versión ESP-12E, V3
- Función: plataforma de conectividad Wifi, TCP-IP, IoT.
- Procesador: Tensílica L106 de 32 bits, bajo consumo.
- Memoria: 4 MB Flash, 32 KB SRAM
- Radiofrecuencia: WiFi, 802.11 b/g/n, 2.4 Ghz, antena integrada.
- Protocolo: TCP/IP integrado, conectividad a internet.
- Comunicación con computadora: a través de interfaz USB-Serial CH340G
- Velocidad de comunicación por USB: 9600 hasta 961,600 bps
- Alimentación: 5v, a través del puerto microUSB ó con fuente externa.
- Corriente máxima consumida: 250 ma.
- 11 pines GPIO disponibles para entrada/salida.
- Pines con funciones de PWM, 1wire, I2C, SPI, UART, ADC.
- Dimensiones: 5.7 x 3.3 x 1.5 cms
- Distancia entre pines: estándar 2.54 mm

ThingSpeak

ThingSpeak es una plataforma de Análisis de datos para Internet de las Cosas (IoT), pero podría analizar cualquier tipo de dato numérico, está en la nube y es posible de una manera muy intuitiva agregar y visualizar datos.

Algunas características:

- Configura fácilmente los dispositivos para enviar datos usando los protocolos populares de IoT.
- Visualiza los datos de tu sensor en tiempo real.
- Datos agregados bajo demanda de fuentes de terceros.
- Utiliza la potencia de MATLAB para darle sentido a tus datos IoT.
- Ejecuta tus análisis de IoT automáticamente en función de los horarios o eventos.
- Prototipado y creación de sistemas IoT sin configurar servidores o desarrollar software web.
- Actúa automáticamente sobre tus datos y se comunica contigo empleando servicios de terceros como Twilio® o Twitter®.

¿Cómo funciona ThingSpeak?

Primero se debe registrar en la web de ThingSpeak.
Configurar un canal (channel). Esta parte es muy importante, porque define los datos que se van a recibir. Cada canal admite 8 campos de datos, 8 posibles datos que vas a poder subir a través de este canal.
Como por ejemplo se pueden subir parámetros de:

- Temperatura
- Humedad
- Presión barométrica
- Velocidad del viento
- Dirección del viento
- Lluvia
- Nivel de luz
- Nivel de batería

Una vez creado el canal, los datos que se necesita para transferir información desde nuestro dispositivo, son el Channel ID y la APIKeys, los mismos son proporcionados por la plataforma.

Como sucede en la mayor parte de servicios de nube, ThingSpeak, también dispone de una aplicación para smartphone.
ThingView, es gratuita y permite acceder a los canales ya creados para visualizar la información. Asimismo, debemos considerar que no tenemos el poder analítico que nos ofrece ThingSpeak en la plataforma web. Sin embargo, es de gran utilidad para conocer el estado de nuestros datos y/o dispositivos conectados a la nube. Cabe señalar que permite acceder a canales tanto públicos, como privados. Si bien, la aplicación no tiene tantas herramientas para la interacción con dispositivos, como sucede con Blynk, es de gran utilidad para proyectos IoT y visualizar la información que fluye.

Descripción del proyecto.

El sensor inteligente consta de los siguientes elementos:

- Sensor de temperatura DS18B20
- Microprocesador ESP8266
- Resistencia de 4,7K
- Batería 6V

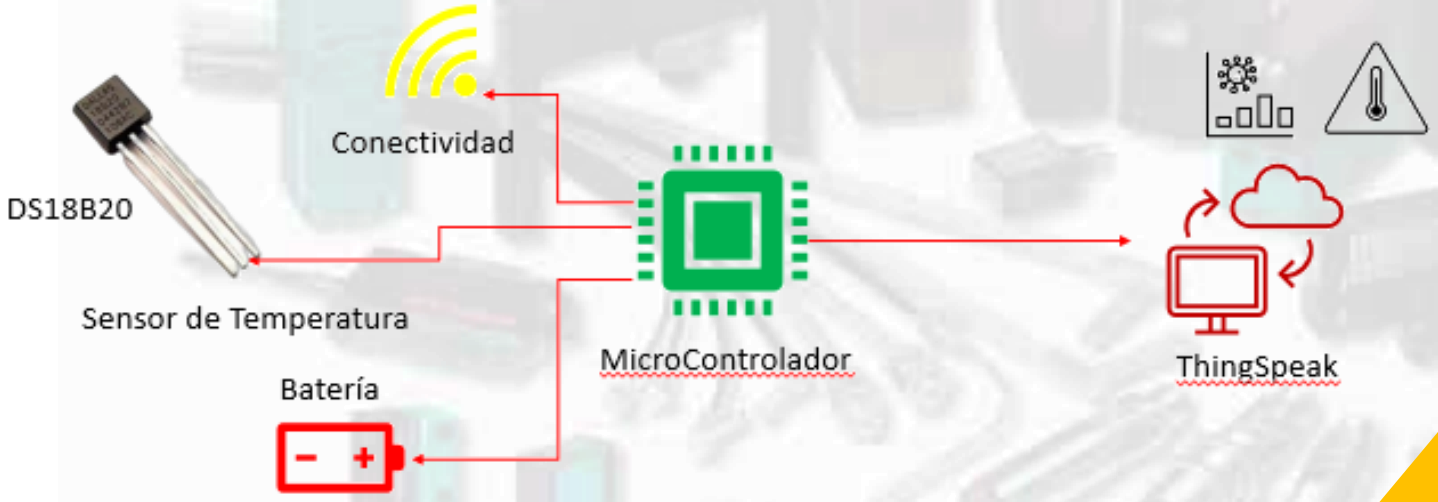
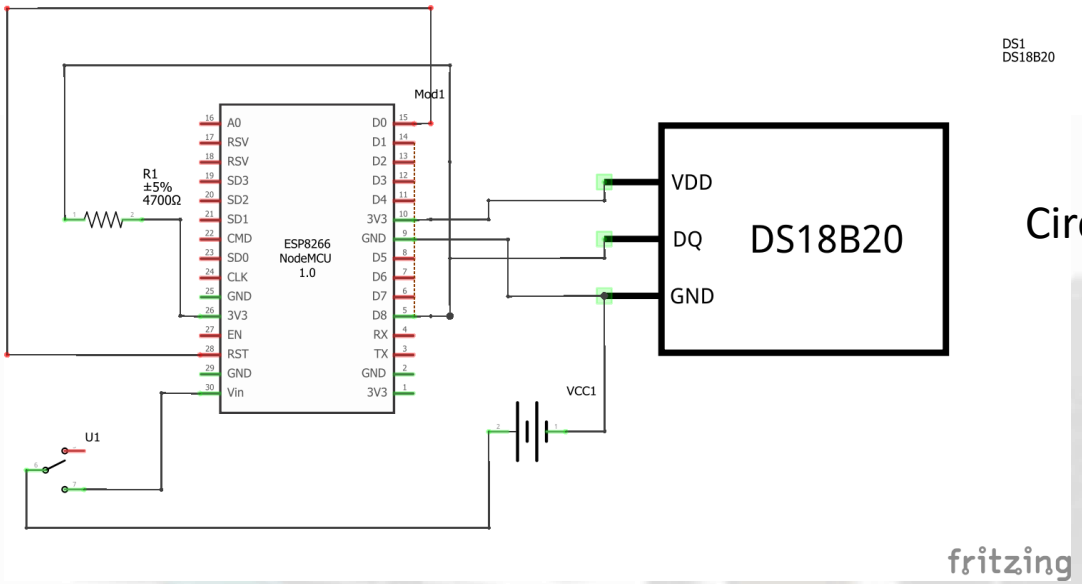


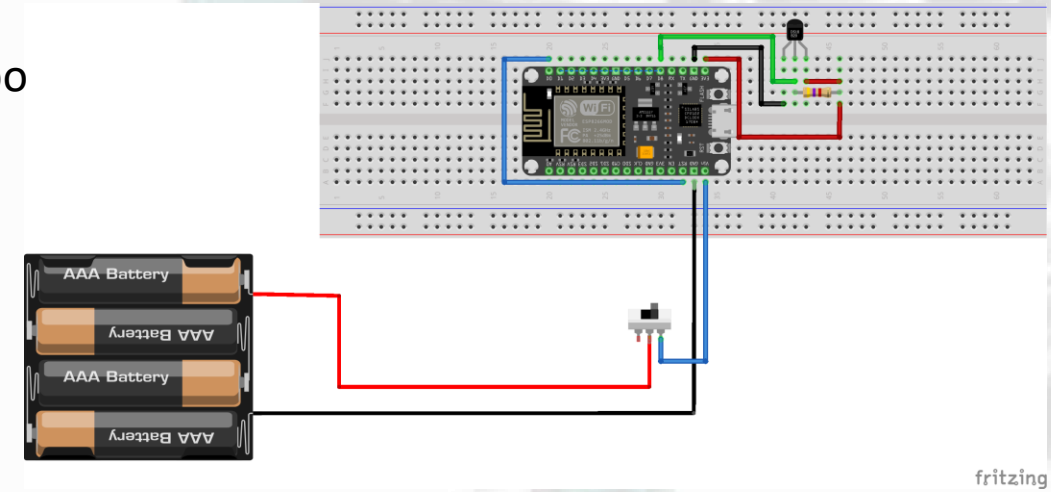
Diagrama en bloques

Alumno: Miguel A. Segnana - TST - SyA : TP#4

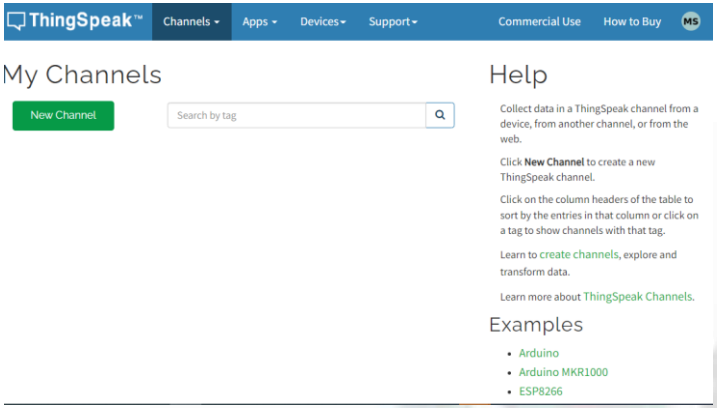
Circuito Electrico

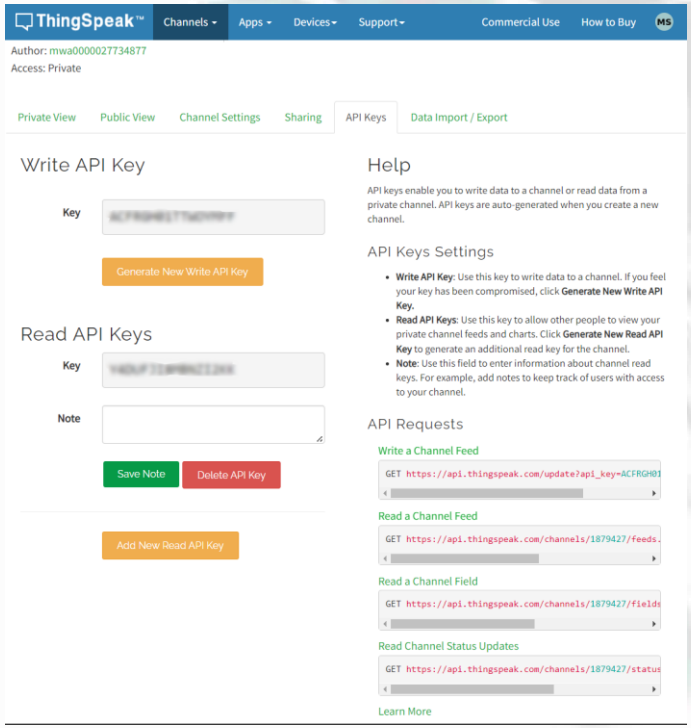


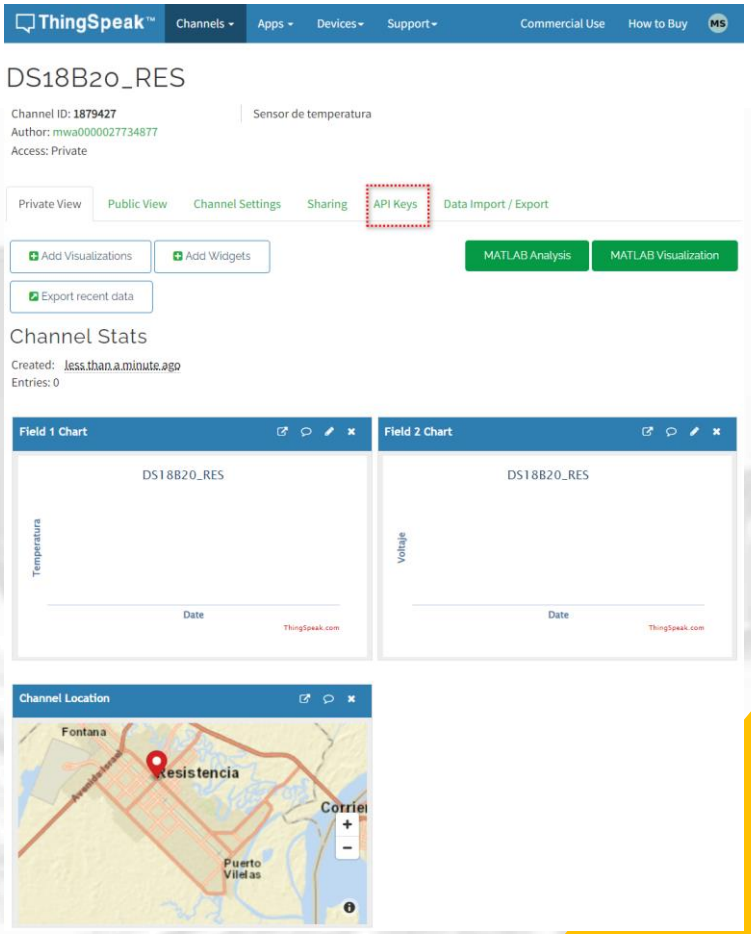
Prototipo



El objetivo es construir un sensor de temperatura inteligente y ver estos datos a través de internet, ya sea por una PC como un SmartPhone.
Además del dato de temperatura, se enviara el dato de voltaje que tiene la batería, esto a los efectos de monitorear también el posible reemplazo de la misma.
Para economizar consumo de energía, el sensor se pondrá en estado de reposo (Sleep) entre intervalos de lecturas y envío de datos, a los efectos del ahorro de energía. Para esta función se utiliza la salida digital D0 y el pin RST, de esta manera es posible "despertar" (wake up) al modulo.
Es importante que el puente entre D0<>RST este desconectado al momento de la carga del firmware del modulo, una vez realizado eso, se conecta nuevamente, se alimenta con la batería y se presiona el botón de RST de la placa para que el sistema funcione correctamente.
Con el firmware instalado y funcionando, el dispositivo envía los datos de temperatura y voltaje a la nube de ThingSpeak, dentro de la misma se configuran los campos necesarios de visualización en el dashboard del canal creado.
Para la vinculación entre el dispositivo y la nube, se utiliza la API Key que entrega en este caso ThingSpeak. Esta API Key puede ser de Lectura y Escritura, esta ultima para poder interactuar con el dispositivo desde Internet, en este caso solo utilizaremos la de lectura para obtener datos.
El mismo se agrega en el código previo a la carga al dispositivo. De esta manera queda vinculado el sensor de temperatura con la nube.


ThingSpeak™
Channels
Apps
Devices
Support
Commercial Use
How to Buy
MS
My Channels
New Channel
Search by tag
Help
Collect data in a ThingSpeak channel from a device, from another channel, or from the web.
Click New Channel to create a new ThingSpeak channel.
Click on the column headers of the table to sort by the entries in that column or click on a tag to show channels with that tag.
Learn to create channels, explore and transform data.
Learn more about ThingSpeak Channels.
Examples
Arduino
Arduino MKR1000
ESP8266


ThingSpeak™
Channels
Apps
Devices
Support
Commercial Use
How to Buy
MS
Author: mwa0000027734877
Access: Private
Private View
Public View
Channel Settings
Sharing
API Keys
Data Import / Export
Write API Key
Key
Generate New Write API Key
Read API Keys
Key
Note
Save Note
Delete API Key
Add New Read API Key
Help
API keys enable you to write data to a channel or read data from a private channel. API keys are auto-generated when you create a new channel.
API Keys Settings
Write API Key: Use this key to write data to a channel. If you feel your key has been compromised, click Generate New Write API Key.
Read API Keys: Use this key to allow other people to view your private channel feeds and charts. Click Generate New Read API Key to generate an additional read key for the channel.
Note: Use this field to enter information about channel read keys. For example, add notes to keep track of users with access to your channel.
API Requests
Write a Channel Feed
GET https://api.thingspeak.com/update?api_key=
Read a Channel Feed
GET https://api.thingspeak.com/channels/1879427/feeds.
Read a Channel Field
GET https://api.thingspeak.com/channels/1879427/fields
Read Channel Status Updates
GET https://api.thingspeak.com/channels/1879427/status
Learn More


ThingSpeak™
Channels
Apps
Devices
Support
Commercial Use
How to Buy
MS
DS18B20_RES
Channel ID: 1879427
Author: mwa0000027734877
Access: Private
Sensor de temperatura
Private View
Public View
Channel Settings
Sharing
API Keys
Data Import / Export
Add Visualizations
Add Widgets
Export recent data
MATLAB Analysis
MATLAB Visualization
Channel Stats
Created: less than a minute ago
Entries: 0
Field 1 Chart
DS18B20_RES
Temperatura
Date
ThingSpeak.com
Field 2 Chart
DS18B20_RES
Voltaje
Date
ThingSpeak.com
Channel Location
Fontana
Resistencia
Puerto Vilelas

```
ADC_MODE(ADC_VCC);

// Librerias necesarias
#include <OneWire.h>
#include <DallasTemperature.h>
#include <ESP8266WiFi.h>

#define DS18B20 D5 //DS18B20 esta conectado al pin GPIO D5 del NodeMCU

String apiKey = "xxxxxx";     //  Poner la apikey que te da ThingSpeak
const char* ssid =  "yyyyyy";  // Poner el nombre de tu red wi-fi
const char* pass =  "zzzzzz"; // Poner la contraseña de tu red wi-fi
const char* server = "api.thingspeak.com";// El servidor de Thingspeak, esto NO se
modifica

// Defino variables numericas
float temp;// Variable para la temperatura leida
int veces;// Variable para los intentos de subir el dato de temperatura a la nube; para que
en caso de apagón, no se gaste toda la batería intentando e intentando subir el dato.
```

```
 OneWire ourWire(DS18B20);// Se declara un objeto para la libreria
DallasTemperature sensor(&ourWire);// Se declara un objeto para la otra libreria
WiFiClient client;// Se define el cliente wi-fi

void setup() {
  pinMode(D0, WAKEUP_PULLUP);//
  delay(1000);// Esperamos 1 segundo
  sensor.begin();// Inicializa el sensor de T
  WiFi.begin(ssid, pass);// Inicializa la comunicación por wi-fi
  while (WiFi.status() != WL_CONNECTED) // Si no se logra conectar, repetira el intento de
conectarse tantas "veces" como se puso en dicha variable
    {
        delay(100);
        ++veces;
        if (veces>50) ESP.deepSleep(300e6);// Luego de 50 intentos se va a dormir para
ahorrar energia, esto sirve para prevenir que si hay un apagón y el router se apago el no
siga gastando energia intentando conectarse
    }}
```

```
 void loop() {
   sensor.requestTemperatures();    // Le pide el valor de temperatura al sensor
   temp = sensor.getTempCByIndex(0);// Lee el valor de temperatura del sensor
   int vcc = ESP.getVcc();          // Mide el voltaje a la entrada de la alimentacion del ESP8266
para evaluar cuanta carga le queda a la batería
   float vccVolt=1.105*vcc/1000;    // Ese factor empírico.
   if (client.connect(server,80))   // "184.106.153.149" or api.thingspeak.com
      {
       String sendData =
apiKey+"&field1="+String(temp)+"&field2="+String(vccVolt)+"\r\n\r\n"; // En field 1 va la
temperatura y en field 2 va el voltaje
```

```
// Estableciendo conexion con Thingspeak
     client.print("POST /update HTTP/1.1\n");
     client.print("Host: api.thingspeak.com\n");
     client.print("Connection: close\n");
     client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
     client.print("Content-Type: application/x-www-form-urlencoded\n");
     client.print("Content-Length: ");
     client.print(sendData.length());
     client.print("\n\n");
     client.print(sendData);
    }
    client.stop();
    ESP.deepSleep(60e6);// microsegundos hasta la proxima medida el modulo se pone a dormir para ahorrar energia
    }
```