Arduino Inicial

Ing. Juan C. Abdala

Clase 5

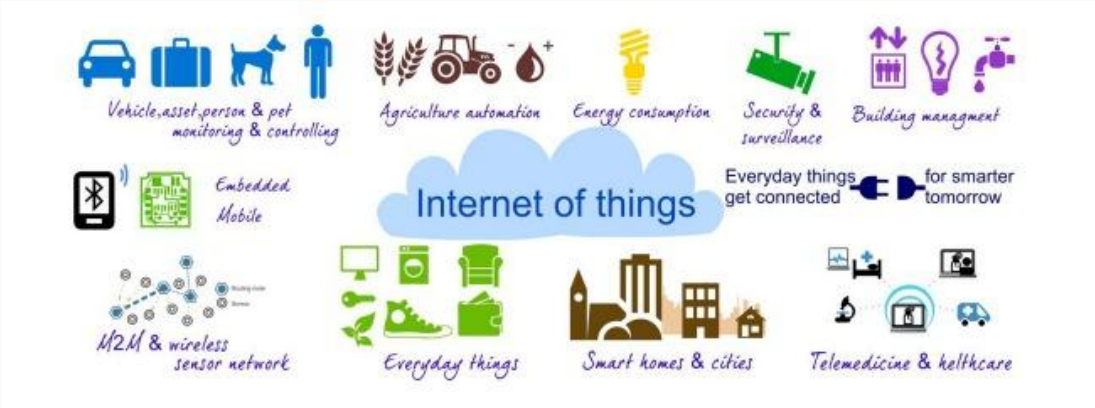
Objetivo de esta clase

- Aprender sobre que es WSN o IOT.
- Aprender sobre las diferentes Tipos de conexiones inalámbricas para Arduino.
- Cómo armar mi proyecto inalambrico.
- Aprender cómo se usa la Radio NRF24 con arduino.

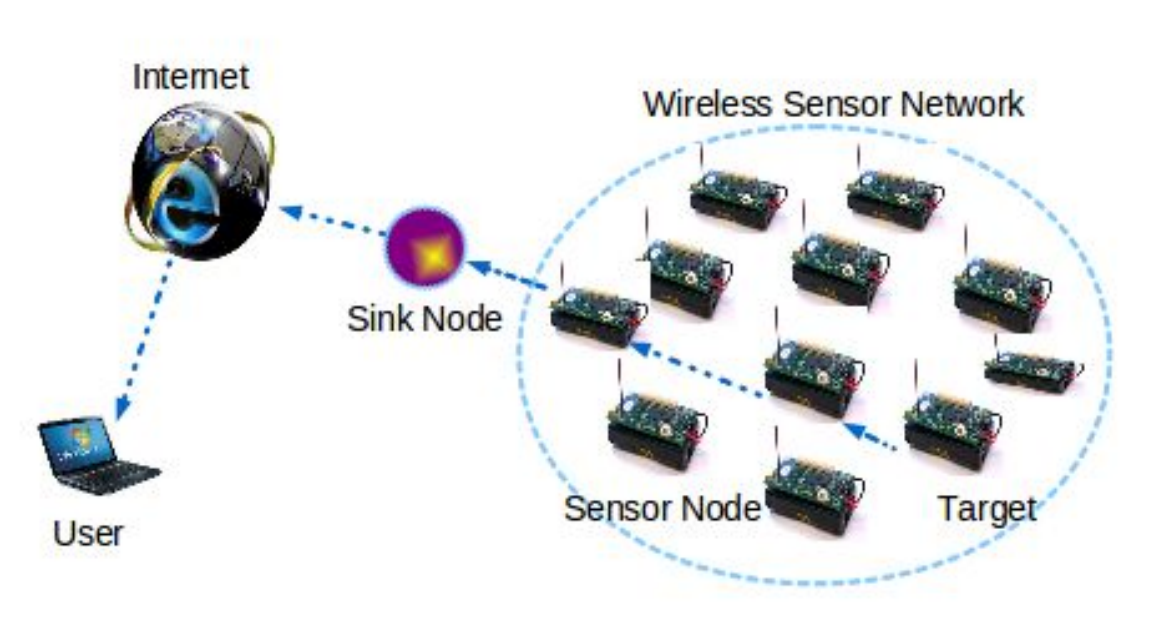
IOT / WSN

Internet de las Cosas (IOT internet of things)

- Es un concepto que se refiere a la interconexión digital de objetos cotidianos con internet.

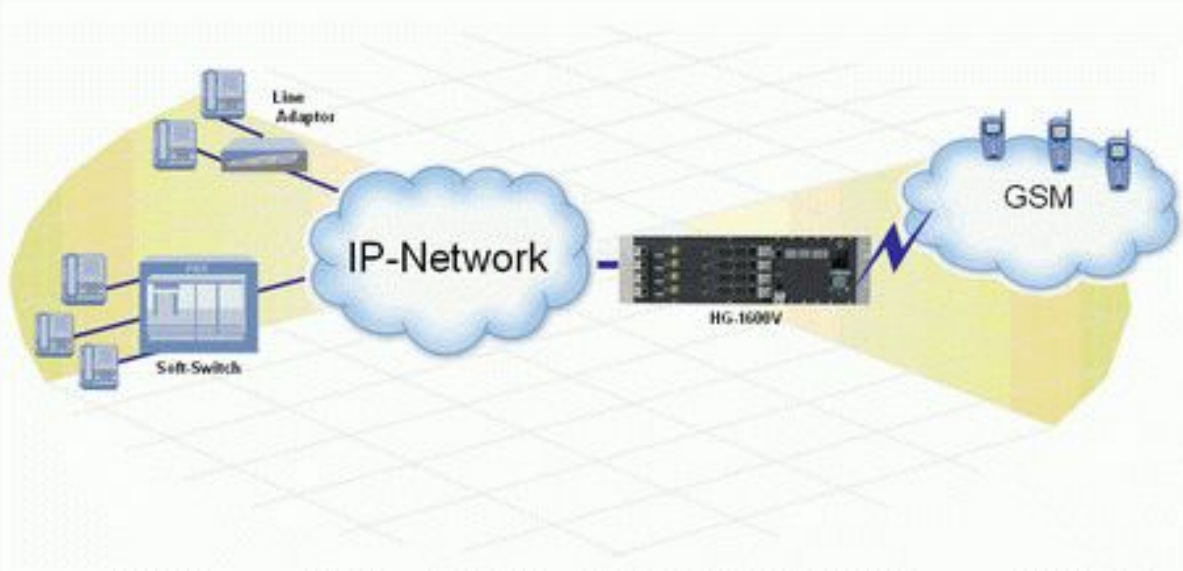


WSN - A traves de un Sink



IOT todos a internet

IOT- Directo a GSM



Protocolo de Comunicación

- Es un **sistema de reglas** que permiten que dos o más entidades que se comuniquen entre ellas para transmitir información por algún medio físico.

- Se trata de las reglas o el estándar que define la sintaxis, semántica y sincronización de la comunicación

IOT - Protocolo de Comunicación

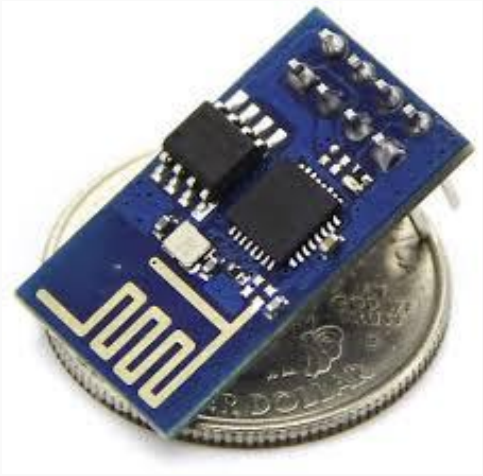
- IPV4/6 + 802.11 (El de todas las computadoras).
- IPV4/6 + 802.15.4 (Radios especiales para IOT).
- Zigbee + 802.15.4 (No se usa mas, propietario).
- Particular del Hardware (nrf24).
- GSM/GPRS.

Conexiones Inalámbricas Arduino

GSM/GPRS

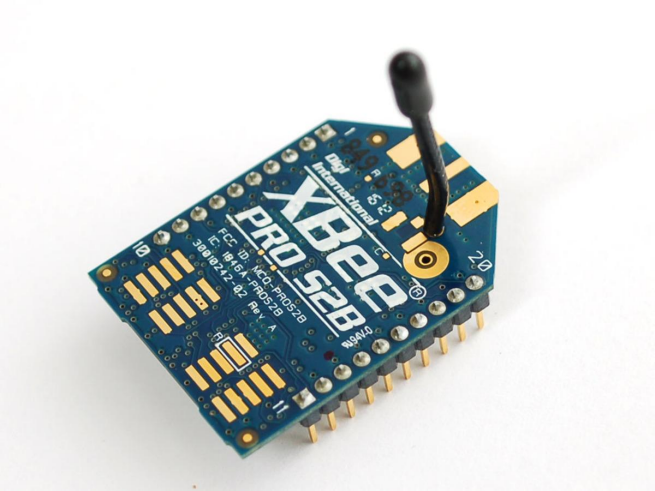
- Usan la red celular (conexión en cualquier parte).
- Alto consumo energetico (~1 A).
- Se paga por lo que se usa.
- Costoso.
- Simple programacion.

WIFI



- 802.11
- Se conectan a red hogareña
- Consumo energetico Medio (~300 mA).
- Precio Medio.
- Programación media, orientada a web.
- ESP8266
- Tambien vienen shields.

Xbee



- 802.15.4
- Bajo consumo energético.
- Costoso.
- Programación más compleja, siempre debemos tener en cuenta la red.
- Diferentes modelos que varían en alcance y frecuencia.

Radios Arduino

Radios Arduino

- Economicas.
- Diversas Frecuencias 433mhz, 900 mhz, 2.4 Ghz.
- Lentas en general.
- Usan sus propios protocolos.
- Bajo consumo.
- Diferentes alcance según la necesidad.

Nuestro Proyecto Inalambrico

A tener en cuenta

- Consumo (que tanta autonomía necesito).
- Alcance.
- Escalabilidad.
- Interferencia (Alteración del canal).
- Velocidad.

Alcance.

- Influye directamente el consumo.
- La frecuencia influye.
- Las antenas mejoran nuestro alcance.

Escalabilidad.

- Conexiones Punto a Punto.
- Gran Tamaño, influye el direccionamiento y enrutamiento .
- Gran Densidad, debemos controlar el acceso al medio.
- Redes Dinamicas on ruteo Mesh.

Interferencia.

- A mayor frecuencia más intolerable obstáculos, como árboles o muros, en comparación .
- Frecuencias muy usadas.
- Jaulas de Faraday.

Velocidad

- Depende de la necesidad.
- Mayor velocidad mayor frecuencia.
- La Frecuencia limita la velocidad.

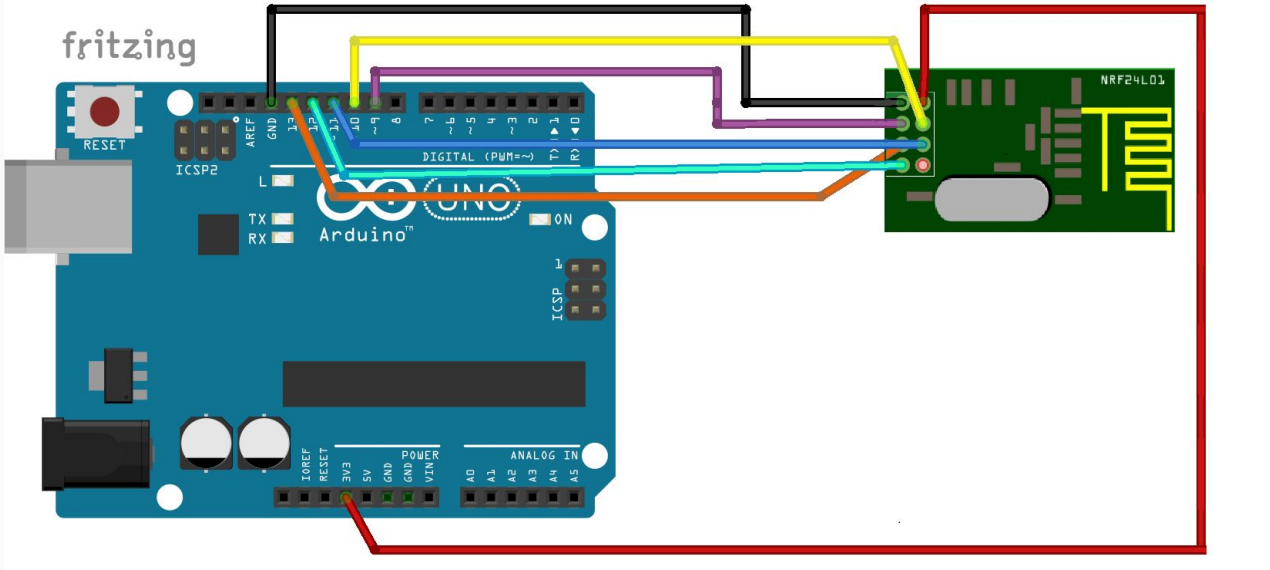
NRF24

Características Tecnicas

- Operan en la banda de 2.4Ghz.
- Velocidad configurable de 250kb a 1Mb por segundo.
- Muy bajo consumo en Stand By (Dormido).
- El alcance entre 20 m y 900m depende modelo.
- Funciona a 3.3v.
- Conexion a traves de SPI.

NRF24 PINOUT - SPI

Esquematico



Libreria

- Buscamos y descargamos:
  Clase5/Codigos/RF24-master

Inicio

```
#include <SPI.h>

#include "RF24.h"

// Iniciamos Radios, con pin 9 CE y pin 10 CS

RF24 radio(9,10);

// Definimos la dirección propia y la de recepción.

byte addresses[][6] = {"1Node","2Node"};
```

Configuracion

```
radio.begin();                    // Inicio el objeto
radio.setAutoAck(1);              //Activa el auto envío de recepción
radio.setPayloadSize(24);         //Tamaño de dato a enviar
radio.setDataRate(RF24_250KBPS); // Velocidad
radio.setPALevel(RF24_PA_MAX);    //MAxima potencia de transmision
radio.setChannel(0x55);           //Canal para transmission
radio.setRetries(15,15);          //Cantidad de tiempo y veces de reenvío
radio.setCRCLength( RF24_CRC_8 ) ;        // Si uso Correccion de Errores
radio.openWritingPipe(addresses[0]);              // Buffer a quien envio
radio.openReadingPipe(1,addresses[1]);            //Buffer de quien recibo
radio.startListening();           // Comienza a escuchar
```

Bien conectado??

Después de nuestra configuración, va:

radio.printDetails();

```
STATUS                  = 0x0e RX_DR=0 TX_DS=0
MAX_RT=0 RX_P_NO=7 TX_FULL=0
RX_ADDR_P0-1            = 0x544d52687c 0xabcdabcd71
RX_ADDR_P2-5            = 0xc3 0xc4 0xc5 0xc6
TX_ADDR                 = 0x544d52687c
RX_PW_P0-6  = 0x01 0x01 0x00 0x00 0x00 0x00
EN_AA                   = 0x3f
EN_RXADDR   = 0x02
RF_CH                   = 0x4c
RF_SETUP    = 0x07
CONFIG                  = 0x0f
DYNPD/FEATURE           = 0x03 0x06
Data Rate       = 1MBPS
Model           = nRF24L01+
CRC Length      = 16 bits
PA Power        = PA_MAX
```

Envio

```
radio.stopListening();
byte dato;
if ( radio.write(&dato,1) )
{
        Serial.println("Sending OK.");
}
else
{
       Serial.println("Sending failed.");
}
```

Recepcion

```
byte recvByte;

while(radio.available())
{
        radio.read( &recvByte, 1 );          //Buffer de donde leo y tamaño
        Serial.print("recibido ");
        Serial.print(recvByte);
 }
```

Estructuras

Para enviar varios datos:

```
struct payload_t                // 4 bytes
{
  int temp;                     // 2 bytes
  int hum;                      // 2 bytes
};
```

Envio Structs

```
radio.stopListening();
payload_t payload = {23, 60 };
if ( radio.write(&payload,4) )
{
        Serial.println("Sending OK.");
}
else
{
       Serial.println("Sending failed.");
}
```

Recepcion

```
while(radio.available())
{
        radio.read( &payload, 4);          //Buffer de donde leo y tamaño
        Serial.print("Temperatura ");
        Serial.println(payload.temp);
        Serial.print("Humedad ");
        Serial.println(payload.hum);
 }
```

Descansemos 15 min

Practico

Vamos al Practico 5:

Material: https://github.com/jcabdala/ArduinoInicialUNC

Consultas:

abdalajc@gmail.com

@toniabdala