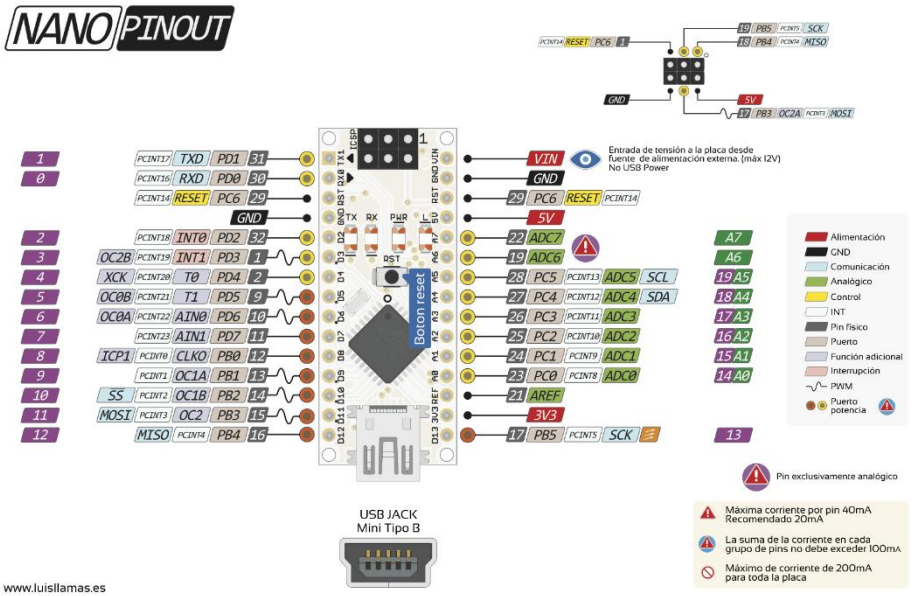
Documentación Hardware

Sistema de Monitoreo Externo

Componentes

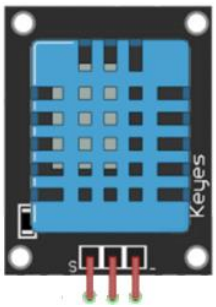
- ***Arduino NANO***



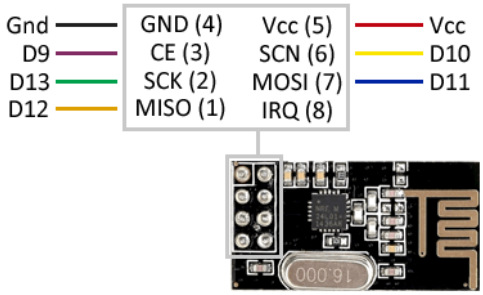
- ***MQ7 Air Co2 Sensor***

- ***MQ135 Air Quality Sensor***

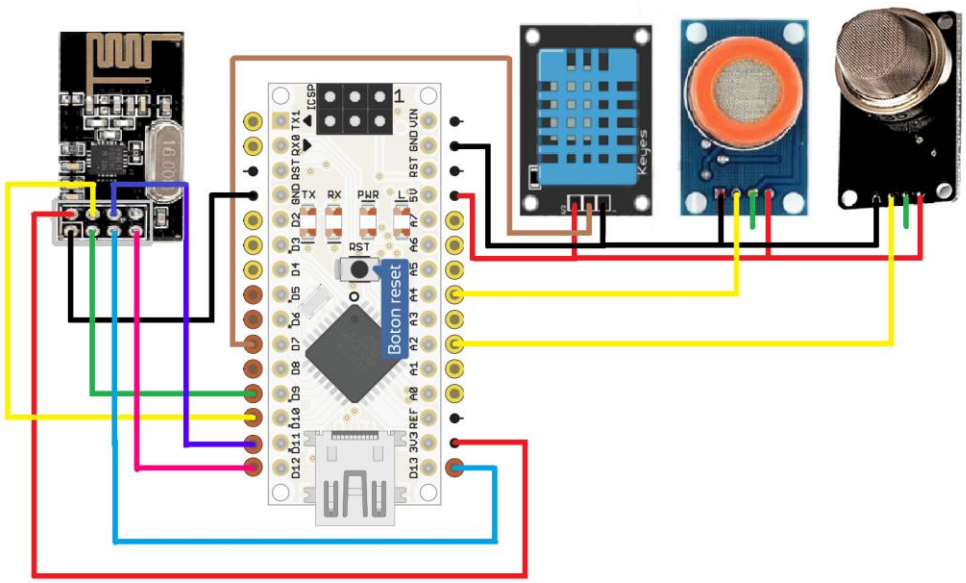
- ***DTH-11 Sensor(Temperature-Humidity)***



- ***NFR2410 2.4Ghz Module TX***



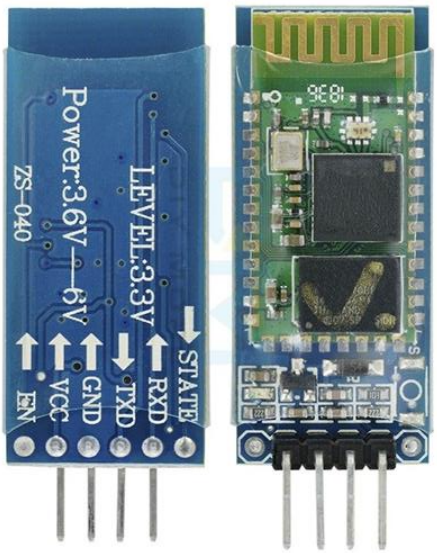
Conecxión


TX
RX
PWR
RST
Boton reset
Keyes

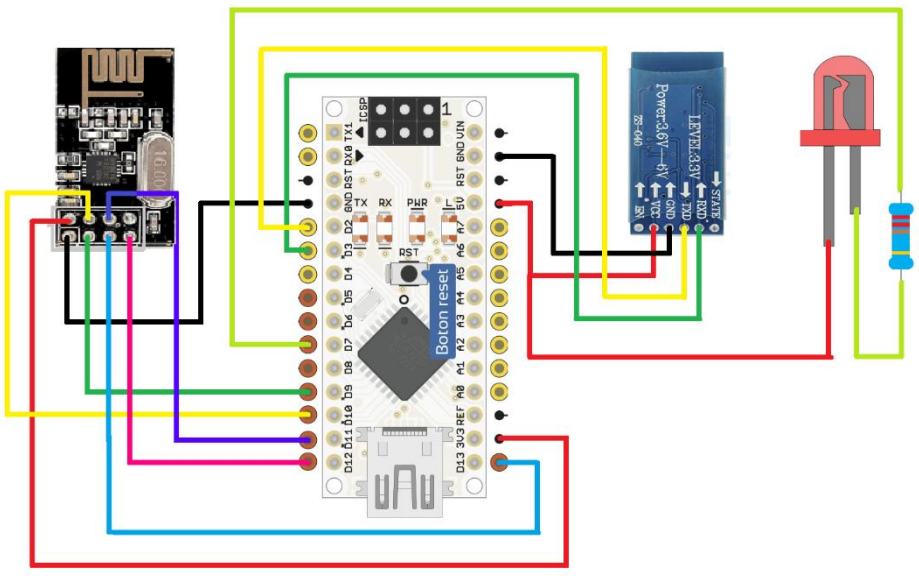
Sistema de alerta Interno

Componentes

- ***Arduino NANO***
- ***NFR2410 2.4Ghz Module RX***
- ***Led Indicator***
- ***BLE HC-05 Wireless Module***



Conexion



Docuemntacion Software

Sistema de Monitoreo Externo

Documentation

El dispositivo número 1, es un dispositivo que tiene como propósito la medición de la calidad del aire en ambientes citadinos, por lo cual está pensado para ser ubicado en un lugar exterior, en donde los sensores tengan contacto directo con el aire circundante. Partiendo de que existe una conexión física entre los sensores y el módulo arduino, es el momento de iniciar el desarrollo del código de proyecto, Para este módulo se tiene, un fragmento de código que permite la lectura del sensor dht11, un fragmento de código para la configuración y lectura de los sensores MQ (mq7 y mq135) y por último el fragmento de código para controlar la comunicación RF (Radio Frecuencia) entre los módulo NFR24L01.

- ***Sensor Dht11***

Las puesta en marcha de este sensor es bastante sencillo, existen diferentes librería que integran funciones y que te permitieran una lectura fácil y correcta del sensor. Para este proyecto se ha utilizado la libreria DX11.zip que integra comandos muy facies de utilizar, y ejemplos de gran ayuda para implementar en arduino IDE, lo único que debes hacer es descargar la librería e importala.

Si deseas un ejemplo más completo de la puesta en funcionamiento de este sensor puedes visitar el link: https://www.prometec.net/sensores-dht11/

- ***Sensor NFR24L01***

El NRF24L01 es un chip de comunicación que integra un transceptor RF a una frecuencia entre 2.4GHz a 2.5GHz, banda libre para uso gratuito. La velocidad de transmisión es configurable entre 250 Kbps, 1Mbps, y 2 Mbps. La librería utilizada en el proyecto para el control de este módulo es nRF24/RF24, la cual presenta funciones y ejemplos que pueden ser de ayuda en el desarrollo de futuros proyectos.

Si deseas mayor informacion del funcionamiento de este modulo puedes visitar la pagina https://www.luisllamas.es/comunicacion-inalambrica-a-2-4ghz-con-arduino-y-nrf24l01/

- ***Sensores MQ***

Los sensores MQ corresponden a una gran variedad de dispositivos para la medición de gases disueltos en el aire, estos sensores son electroquímicos y varían su resistencia cuando se exponen a determinados gases, internamente posee un calentador que aumenta la temperatura interna lo que le permite reaccionar a los gases.

Existen diversos sensores MQ, cada uno con características que lo hacen ideales para la medición de uno o varios gases en particular. En este proyecto se hará uso de los sensores MQ7 para la medición de CO “Monóxido de carbono” y M135 para las mediciones de NH4 “Amonio”. Este tipo de sensores son fáciles de poner en funcionamiento, la mayoría cuentan con una salida digital booleana (1 - 0), o una salida analogica (0v – 5v). El mayor reto y paso importante para las puesta en funcionamiento de este tipo de sensores MQ es la calibracion, proceso con el cual se establece una función que permite que el sensor realice las mediciones con precisión de las condiciones ambientales.

Si deseas una mayor información de toda la gama de sensores MQ, ademas de la documentacion oficial revisa el link: https://playground.arduino.cc/Main/MQGasSensors/

Calibración

El proceso de calibración de este tipo de sensores se divide en 3 pasos:

Paso 1

Leemos los pines analogicos de la arduinoNano, con lo cual obtenemos una valor en voltios que es la respuesta del sensor MQ. Los microcontroladores utilizados por arduino en sus módulos, tienen la incapacidad de interpretar una señal puramente analogica como lo es un voltaje, por lo cual usan un conversor Análogo – Digital interno en sus pines analogicos con una resolucion de 10 bits, con lo cual un el valor de 0 voltios analógico puede ser expresado en digital como 0000000000 (0) y un valor de 5V analógico es expresado en digital como 1111111111 (1023). Por lo tanto todo valor analógico intermedio al final es expresado como un valor digital entre 0 y 1023.

En el codigo primero realizamo la lectura del pin analogico donde conectamos el sensor MQ, con esto se obtiene un valor entre 0 y 1023.

```
adc_MQ = analogRead(analogPin)
```

Por medio de una regla de tres simple convertimos el valor digital obtenido en un valor de voltaje entre 0v y 5v

```
readVolt = adc_MQ * ( powerVolt / 1023.0)
```

Equation 1

Si implementas esto en codigo vas a obtener una lectura digital entre 0 y 1023, y su respectiva conversión a voltios entre 0v y 5v, pero aunque podríamos interpretar esta señal como la cantidad de particulas por millon (ppm) de un determinado gas en el ambiente las cosas no son tan fáciles. La mayoría de sensores de este tipo son sensibles a múltiples gases, así que cualquier gas podría generar cambios de valor en el sensor, y no específicamente el gas que es de tu interés medir, esto genera lecturas erróneas. Por otro lado el comportamiento de los sensores MQ entre la lectura analógica y el valor real no es del todo lineal, por lo cual se debe estimar la curva de comportamiento que nos ofrecen en el dataSheet (paso 2)

Paso 2

En este paso se realiza la estimación de la ecuación para la curva de comportamiento del sensor para un gas en particular. Como se aprecia en la gráfica 1, imagen que se extrajo del datasheet para el sensor MQ 07. Como se aprecia este sensor se comporta diferente ante la presencia de diversos gases como co, h2, lpg, ch4, alcohol y aire.

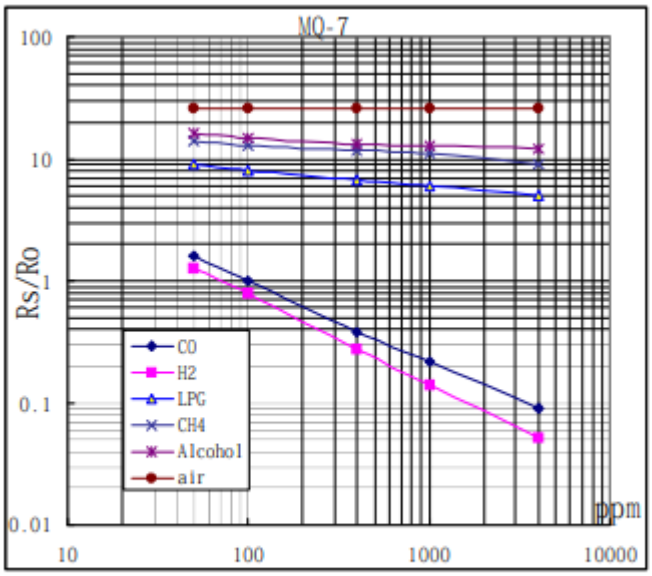


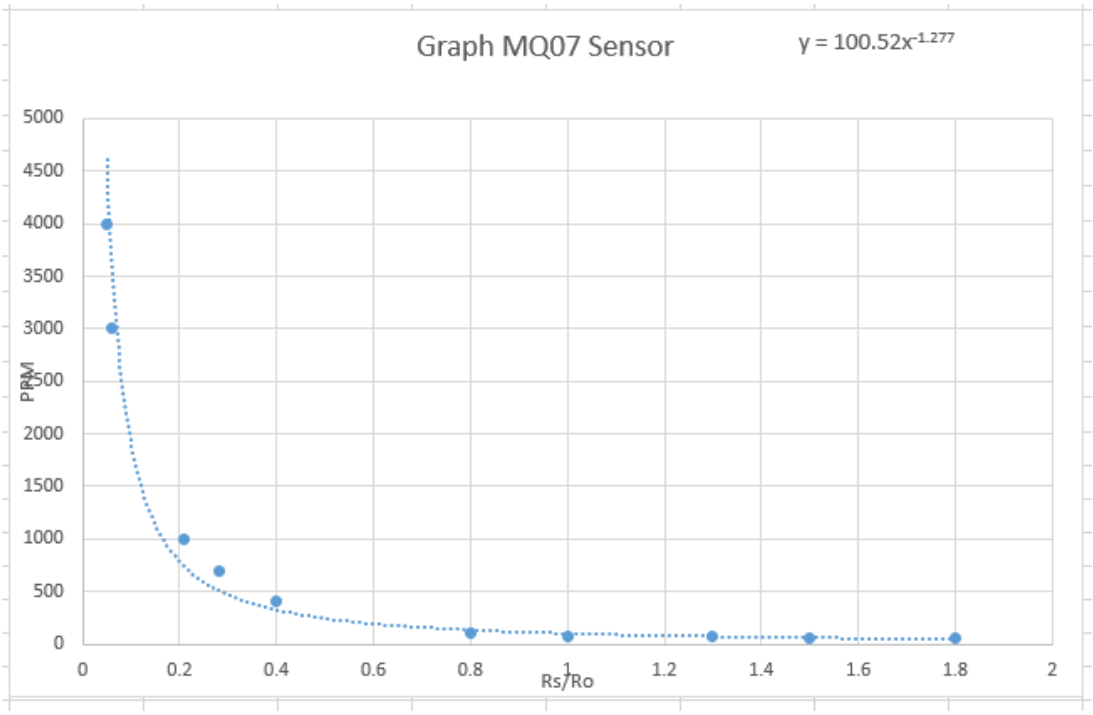
Fig.3 is shows the typical sensitivity characteristics of the MQ-7 for several gases.
in their: Temp: 20℃、
Humidity: 65%、
O_2 concentration 21%
RL=10kΩ
Ro: sensor resistance at 100ppm CO in the clean air.
Rs: sensor resistance at various concentrations of gases.

Fig.3 sensitivity characteristics of the MQ-7

Grafica 1

Para este caso en particular el gas que nos interesa es el monóxido de carbono, por lo cual utilizamos la línea correspondiente para este gas (linea azul con rombos). De forma manual obtenemos los puntos, los graficamos y por medio de la herramienta excel se obtiene la línea de tendencia que más se aproxime a los datos suministrados, al obtener la linea de tendencia también observamos la ecuación correspondiente.

MQ07 Carbon Monoxide Concentration sensor	
Rs/Ro	ppm
1.8	50
1.5	60
1.3	70
1	80
0.8	100
0.4	400
0.28	700
0.21	1000
0.06	3000
0.05	4000



Grafica 2

$$y = 100.52\, x^{-1.277}$$

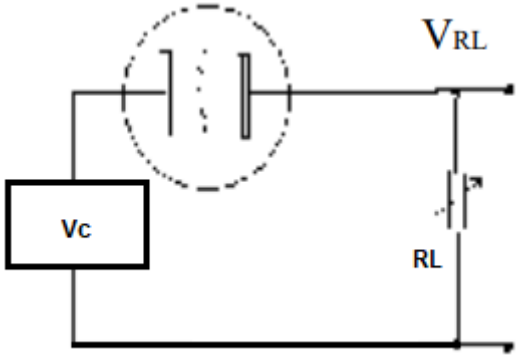
Equation 2

Reemplazando y = ppm y x = Rs/Ro se obtiene

$$ppm = 100.52\, (\frac{Rs}{Ro})^{-1.277}$$

En donde ppm es la cantidad de particulas por millon de monóxido de carbono disueltas en el aire, Ro es la resistencia del sensor a un concentración de 100 ppm de monóxido de carbono y Rs es la resistencia del sensor a varias concentraciones del gas. Esta ecuación presenta 3 incognitas ppm de monóxido de carbono, Rs y Ro.

Para calcular el valor de Rs despejamos la ecuación del divisor de voltaje que forma el sensor con la resistencia de carga RL que se especifica en el datasheet, para el caso de este sensor RL = 10kΩ.



$$V_{RL} = V_c(\frac{R_L}{R_s + R_L})$$

Equation 3

Despejamos Rs de la ecuación 3 se obtiene

$$R_s = R_L\left(\frac{V_c}{V_{RL}} - 1\right)$$

$$donde\ V_c = 5v, \qquad R_L = 10K\Omega, \qquad V_{RL} = readVolt$$

Recordar que el voltaje en RL es el voltaje de salida del sensor, que a su vez es el voltaje que se obtenía con la ecuación 1, cuando leiamos la entrada analogica de la Arduino. Al reemplazar en la formula queda como la ecuación 4. Tener en cuenta que el valor de Rs está variando constantemente y esto depende de la concentracion de monoxido en donde se encuentre el sensor

$$R_s = R_L\left(\frac{V_c}{readVolt} - 1\right)$$

$$R_s = 10K\Omega\left(\frac{5v}{readVolt} - 1\right)$$

Equation 4

Paso 3

A diferencia de Rs, Ro es un valor constante que representa la resistencia del sensor a una única concentración de 100ppm aire limpio, por lo cual para calcular este valor hacemos uso del gráfico 2. En este gráfico se aprecia que a concentraciones muy altas de CO la variación en el valor de Rs/Ro son mínimas, a diferencia que a concentraciones bajas de CO en donde las variaciones de Rs/Ro son significativas. Con el fin de minimizar los errores lo mejor sería elegir Rs/Ro para concentraciones altas de CO, en la gráfica observamos que la mayor concentración es de 4000 ppm de CO y que a esta concentración el valor de Rs/Ro = 0.05 aprox.

Entonces asumiendo que el sensor se satura con 4000ppm al cual según la gráfica le corresponde un Rs/Ro de 0.05. Para estar en este punto de saturacion generamos un ambiente con bastante CO y medimos con el sensor, al realizar esta medicion se obtiene un voltaje de 4.02 v, que al reemplazar en la ecuacion 4 se tiene un valor de Rs=2437.81Ω.

$$R_s = 10K\Omega\left(\frac{5v}{readVolt} - 1\right)$$

$$R_s = 10K\Omega\left(\frac{5v}{4.37v} - 1\right)$$

$$R_s = 1441.64\,\Omega$$

Se sabe que Rs / Ro sera igual a 0.05 a concentraciones muy altas de CO por lo cual

$$Rs/Ro = 0.05$$

Despejamos Ro y reemplazamos Rs

$$Ro = Rs/0.05$$

$$Ro = 1441.64 / 0.05$$

$$Ro = 28832$$

Con Ro calculado, ya tenemos una solución de la ecuación y con esto podemos obtener los valores aproximados de concentración de CO, .

$$ppm = 100.52\ (\frac{Rs}{Ro})^{-1.277}$$

$$ppm = 100.52\ (\frac{Rs}{28832})^{-1.277}$$

Para el módulo MQ135 se realiza exactamente el mismo proceso, entre los dos sensores existen cambios significativos al calcular la línea de tendencia, y esto es normal ya que cada sensor de la serie MQ presenta un comportamiento en particular que debe ser tenido en cuenta, esto se aprecia en el dataSheet. Es importante decir que existen en la web diferentes proceso de calibracion para estos sensores, este proyecto tomo como referenica la informacion y procedimientos que se siguieron en el link: https://naylampmechatronics.com/blog/42_Tutorial-sensores-de-gas-MQ2-MQ3-MQ7-y-MQ13.html

Código completo

Como se aprecia en este código he utilizado los diferentes ejemplos de código individuales para el para el control de los sensores y los he combinado en uno solo. Además de la creación de una funcion que contiene código que se reutiliza tanto para el sensor MQ07 como para el MQ135.

```
#include <DHT11.h> // Incluye librería para manejo DHT
#include <SPI.h>
#include <nRF24L01.h> //Incluye libreria para manejo tranceptor
#include <RF24.h>
#include <SoftwareSerial.h>

int pin=7;
DHT11 dht11(pin);

const int pinCE = 9;
const int pinCSN = 10;
RF24 radio(pinCE, pinCSN);

// Single radio pipe address for the 2 nodes to communicate.
const uint64_t connect1 = 0xE8E8F0F0E1LL;
float data[4];

float temp, hum;
float monOxide, nh4, Rs;


void setup() {
  Serial.begin(9600);
```

```
  radio.begin();
  radio.openWritingPipe(connect1);

}

void loop() {

  //------- Llenamos el arreglo con los datos de los dos sensores MQ y DHT ----------------

    dht11.read(hum, temp); // Hacemos uso del método read para leer y asignar valor a las variables hum y temp

    Rs = functReadMQSensor(A4, 5, 10000);
    monOxide = 100.52 * pow(Rs/5463, -1.277); // Esta ecuación se obtiene por medio de una regresión lineal, exponencial   según sea el caso,
//para el sensor Mq7 es esta, pero varía dependiendo del sensor (revisar DataSheet del sensor)  //calculamos la concentración  de monóxido
de carbono con la ecuación obtenida.

    Rs = functReadMQSensor(A2, 5, 20000);
    nh4 = (30.593 * pow(Rs/5463, 2)) - (167.14 * (Rs/5463)) + 231.55; // Esta ecuación se obtiene por medio de una regresión lineal,
exponencial según sea el caso,  //para el sensor Mq135 es esta, pero varía dependiendo del sensor (revisar DataSheet del sensor)  //calculamos
la concentración  de nh4 con la ecuación obtenida.

   data[0] = monOxide;
   data[1] = nh4;
   data[2] = temp;
   data[3] = hum;

   radio.write(data, sizeof data);
   delay(5000);
}
float functReadMQSensor(int analogPin, float powerVolt, float rL){   // analogPin pin conexion sensor, inVoltage voltaje de alimentación
Sensor (5v o 3.3v)
   int adc_MQ = analogRead(analogPin); //Leemos el puerto analogico para determinar la salida analógica  del sensor MQ
   float readVolt = adc_MQ * ( powerVolt / 1023.0); //Convertimos la lectura en un valor de voltaje
   float rS = rL *((5-readVolt)/ readVolt);  //Ecuación del divisor de voltaje, formada por el sensor y resistencia rL, permite calcular  rS
   return rS;                          // para el sensor MQ7 rL = 10K, para MQ135 rL  = 20K (Segun datasheet)
}
```

Sistema de Alerta Interno

Documentation

El dispositivo número 2 funciona como interfaz que comunica al usuario en tiempo real y por medio de una luz intermitente, cuando los niveles de CO o HN4 están por encima de los parámetros especificados (> 50 ppm) lo que podrían generar desde molestias respiratorias hasta daños severos a la salud. El dispositivo está pensado para ser ubicado en espacios interiores donde el usuario los ubique fácilmente con la vista, este dispositivo cuenta con conexión bluetooth para poder ser vinculado a la aplicación móvil android y de esta forma generar un seguimiento de los niveles de contaminantes en el aire de las vías transitadas.

- ***Sensor NFR24L01***

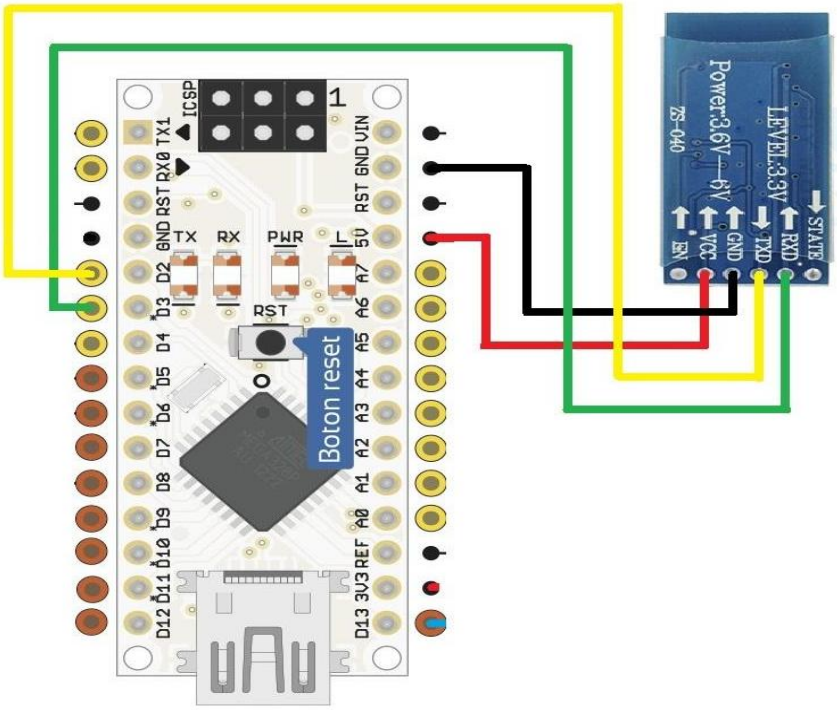
Este dispositivo hace uso de mismo modulo NRF24L01 pero esta vez configurado como receptor, La librería utilizada en el proyecto para el control de este módulo es nRF24/RF24.

Si deseas mayor informacion del funcionamiento de este modulo puedes visitar la pagina https://www.luisllamas.es/comunicacion-inalambrica-a-2-4ghz-con-arduino-y-nrf24l01/

- ***Modulo BLE HC-08 Wireless***

El módulo Bluetooth HC-08 es parte de la nueva generación de dispositivos para la transmisión de datos usando el protocolo Bluetooth Low Energy “BLE” o Bluetooth V4.0, este dispositivo funciona con una frecuencia de trabajo de 2.4Ghz y un alcance máximo de 100m a línea de vista. Esta pensados para brindar la posibilidad de comunicación a tus proyectos con dispositivos que implementen esta tecnología, brindando un bajo consumo de energía y costos de operación considerablemente reducidos.

Para la puesta en funcionamiento de este sensor el primer paso es la configuracion del modulo, características como Modo: Maestro – Esclavo, Nombre: HC-05, Contraseña: 1234, Velocidad (baud rate): 9600 pueden ser modificadas, acción que se realiza por medio de comando AT. Para este proceso debemos conectar el módulo BLE HC-08 a la tarjeta arduino, en este caso se ha seleccionado los pines 2 y 3 como Rx y Tx respectivamente, pero pueden varias según tus preferencias.



Codigo configuracion modulo BLE por comandos AT

```
#include <SoftwareSerial.h>
SoftwareSerial SerialBt(2,3);

void setup()
{
   SerialBt.begin(115200);
   Serial.begin(115200);
   Serial.println("Enter AT commands:");
```

```
}
void loop()
{
   if (SerialBt.available()) {
      Serial.write(SerialBt.read());
   }
   if (Serial.available()) {
      SerialBt.write(Serial.read());
   }
}
```

Nota: Es muy importante que la velocidad en baudios configurada en el módulo HC-08 sea igual a la velocidad con la que configuras tu comunicación serial en la Arduino al momento de programar.

Lista de comandos AT

AT — check operability;

AT+HELP — output all commands;

AT+DEFAULT — reset factory settings;

AT+RESET — soft reset;

AT+ROLE — output current mode;

AT+ROLE0 — set slave mode;

AT+NAME — output module name;

AT+NAMESimon — set module name as Simon;

AT+PIN — output PIN code (password) for pairing;

AT+PIN123456 —set PIN code as 123456;

AT+UUID — output service UUID;

AT+UUID0xFFE0 — set service UUID as 0xFFE0;

AT+CHAR — output characteristic UUD;

AT+CHAR0xFFE1 — set characteristic UUID as 0xFFE1.

Código completo

```
#include <SPI.h>
#include <nRF24L01.h>
#include <RF24.h>
#include <SoftwareSerial.h>

const int pinCE = 9;
const int pinCSN = 10;
RF24 radio(pinCE, pinCSN);
```

```
SoftwareSerial SerialBLE(2,3); //puerto de comunicación BLE

const int ledPIN = 7; // pin de salida del LED indicador

// Single radio pipe address for the 2 nodes to communicate.
const uint64_t connect1 = 0xE8E8F0F0E1LL;

float data[4];  // Arreglo de datos que son recibidos del emisor
uint8_t dataBLE[20]; // Arreglo de datos que serán enviados al Móvil

void setup()
{
  radio.begin();
  Serial.begin(115200);
  SerialBLE.begin(115200);  /// Velocidad de los Baudios para la comunicación BLE
  radio.openReadingPipe(1, connect1);
  radio.startListening();
  pinMode(ledPIN , OUTPUT);  //definir pin como salida
}

void loop()
{
  if (radio.available())
  {
    radio.read(data, sizeof data); // Recibo los datos del modulo Emisor y los alaceno el el arreglo "data"

    if(data[0]>50 || data[1]>50 ){// en caso de que las variables CO o NH4 sean elevadas se enciende el LED
      digitalWrite(ledPIN , HIGH);   // poner el Pin en HIGH
      //delay(1000);  // esperar un segundo
      }
    else{
      digitalWrite(ledPIN , LOW);    // poner el Pin en LOW
      //delay(1000);                 // esperar un segundo
      }

     for(int i=0; i<20; i=i+4){
      dataBLE[i] = data[0]; // colocamos el valor de Co2 que se encuentra en data[0],  y lo almacenamos en dataBLE
      dataBLE[i+1] = data[1];// lo mismo con este y el resto
      dataBLE[i+2] = data[2];
      dataBLE[i+3] = data[3];
      }

    Serial.print("Co2 = " );
    Serial.print(data[0]);
    Serial.print("NH4 = " );
    Serial.print(data[1]);
    Serial.print("--Temp = " );
    Serial.print(data[2]);
    Serial.print("--Hum = " );
    Serial.println(data[3]);

    SerialBLE.write(dataBLE, 20); //Envía el arreglo  al movil
  }
  delay(1000);
  }
```