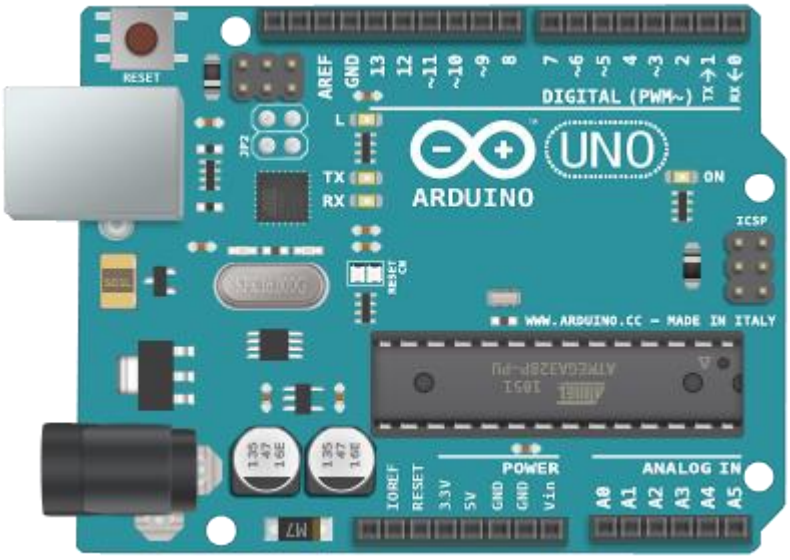
Universidad del Valle de México.

ING.COMUNICACIONES Y ELECTRÓNICA

MATERIA: ELECTRONICA DE COMUNICACIONES



AREA: INGENIERÍA.
PROFESOR: GERSON VILLA GONZÁLEZ

PRACTICA No. 6 **Módulos de Radio Frecuencia RF de 433 MHz**	Fecha de Elaboración_____ Fecha de revisión.:_____ Responsable:_____

OBJETIVO: Como trabajar con módulos de Radio Frecuencia RF de 433 MHz

INVESTIGACIÓN PREVIA

A) Manejo del Arduino UNO

B) Programacion de un módulo RF de 433 Mhz

MATERIAL

- Arduino Uno
- Arduino Nano (Puede ser otro Arduino Uno)
- Módulos RF a 433 MHz (Emisor y Receptor)
- Cables macho-macho

MARCO TEORICO:

vamos a trabajar con los módulos de Radio Frecuencia RF de 433 MHz con los que podremos realizar una comunicación inalámbrica entre dos Arduinos de una manera muy económica, pero a la vez fácil. Si estás buscando como enviar información wireless o

simplemente por aprender cosas nuevas, seguro que este tutorial te va a encantar.

¿QUÉ ES UN MÓDULO DE RF 433 MHZ?.

Un módulo de radio frecuencia RF es aquel que tiene la capacidad de establecer una comunicación inalámbrica con otro módulo de RF a 433 MHz. Al ser necesaria la existencia de dos módulos, uno debe de actuar como receptor y el otro como emisor (comunicación unidireccional o simplex – lo que sucede con estos módulos) o ambas a la vez (comunicación bidireccional o duplex – con otros módulos). Estos dispositivos se han convertido en elementos muy comunes en la programación en Arduino debido a su precio, ya que son muy baratos.

Estándares de comunicaciones inalámbricas

Estos módulos trabajan concretamente con una frecuencia de 433 MHz, que se trata de una banda libre con la que se puede trabajar sin problema alguno. Si quieres consultar más sobre las bandas de frecuencias disponibles te dejamos un pequeño

Bluetooth Tecnología de radio de corto alcance que permite las comunicaciones y la sincronización de datos entre ordenadores personales, PDAs, teléfonos móviles y dispositivos periféricos así como también entre los dispositivos e Internet.

ZigBee Zigbee es un estándar de transmisión de datos de baja velocidad, en dos direcciones, para redes de automatización industriales y domésticas. Utiliza dispositivos pequeños y de muy poca potencia para su conexión conjunta con el fin de formar una web de control inalámbrica. Este estándar soporta bandas de radio sin licencia de 2,4 GHz.

GPRS Abreviatura de General Packet Radio Service. Estándar para las comunicaciones inalámbricas que funciona con velocidades de hasta 115 kbps, comparado con el actual GSM (Sistema Global para Comunicaciones Móviles) de 9,6 kbps.

RFID Contactless tags: Abreviatura de Radio Frequency Identification. La energía emitida por el lector de la etiqueta de la RFID es absorbida por la etiqueta y luego se utiliza para escribir nuevos datos en la etiqueta o para descargar los datos de identificación.

433MHz: Estándar europeo de-frecuencia ancha. Se utiliza para telemetría por radio, por control remoto. Distancia hasta 100 m. Exento de licencia.

434MHz: Estándar europeo de frecuencia ancha. Se utiliza para telemetría por radio, por control remoto Distancia hasta 1 kilómetro. Exento de licencia.

458MHz: Estándar europeo de frecuencia ancha. Se utiliza para telemetría por radio, por control remoto Distancia hasta 10 kilómetros. Exento de licencia.

868MHz: Estándar europeo de frecuencia ancha, menos difundido que el de 433 MHz. Distancia hasta 500 m.

Banda estrecha: Radio de banda estrecha que opera en una banda de frecuencia o difusión pequeña. Por consiguiente, el receptor está menos expuesto a interferencias externas RF y por la naturaleza de su diseño tiene una alta sensibilidad.

2.4GHz: Actualmente es el único estándar de frecuencia mundial. Distancia típica 100 m. Los diseños de radio de esta frecuencia

utilizan el "spread spectrum" (espectro extendido). Esto significa que pueden coexistir muchas radios en el mismo entorno al mismo tiempo sin que se interfieran las unas con las otras.

W-I.AN: Abreviatura de Wireles Local Area Network. Se basa en IEEE 802.11. Incluye los protocolos WIFI y WiMAX.

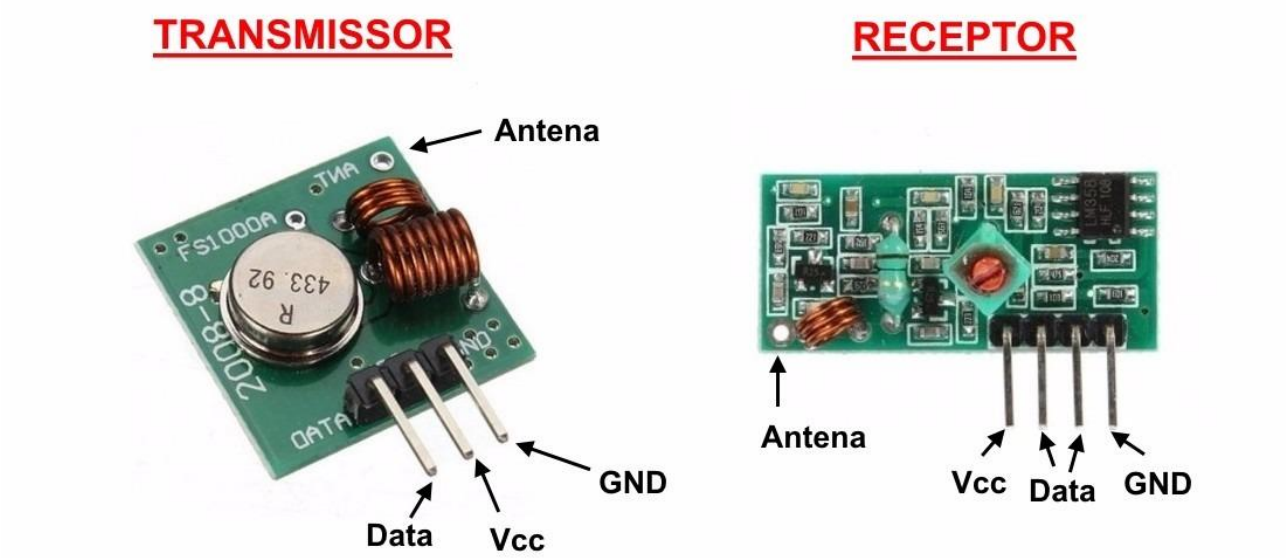
EDGE: Enhanced Data GSM Environment. Se trata de una versión más rápida del GSM. Permite velocidades para datos hasta 384 kbps.

GPS: Sistema de Posicionamiento Global. Es un sistema mundial MEO de navegación por satélite formado por 24 satélites que están en órbita alrededor de la tierra, emitiendo una señal durante las 24 horas del día, y por sus correspondientes receptores en la tierra.

Como ya mencionamos anteriormente, la comunicación es simplex, es decir, unidireccional (uno es el emisor y el otro el receptor) y se realiza a una velocidad de transmisión muy baja, aproximadamente 2400 bps. Una de las características más destacadas de estos módulos es que el alcance depende del voltaje con el que los alimentemos y la antena empleada.

Por ejemplo, tenemos los siguientes casos:
Con 5V y con antena conectada al módulo, la distancia de transmisión no superará los 2 metros.
Con 12V y haciendo uso de una antena de cobre de 16.5cm de longitud, la distancia que puede llegar a alcanzar en un espacio abierto es de unos 300 metros.
Por lo general, estos elementos son empleados en pequeños proyectos donde no se requiera una exigencia elevada en cuanto a distancia, ya que el alcance se puede ver perfectamente afectado por obstáculos reduciéndolo de manera drástica. Algunas de las

aplicaciones de estos módulos son, por ejemplo, la comunicación a distancia con sensores de humedad, presión, temperatura, luminosidad,… Activar y desactivar de manera inalámbrica dispositivos como una alarma,… controlar un led o un motor,…



Algunas de las alternativas que hay en el mercado y que tienen un mayor alcance que los módulos tratados en este post son:

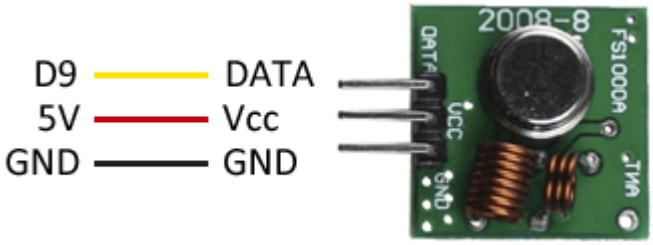
NRF24L01 (Muy económicos y una muy buena alternativa)
XBee (Más caros que los anteriores, pero con mayor alcance. Hay varios modelos con y sin antena y en función de ello variará el precio)
APC220 (Probablemente los que más alcance tengan, pero no son baratos)
Decantarte por uno o por otro depende de tus necesidades para el proyecto en cuestión y de tu presupuesto disponible.

DESARROLLO DE LA PRACTICA.

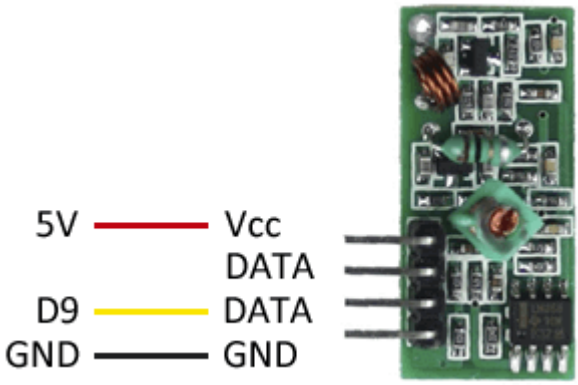
ESQUEMA

En este caso haremos uso de dos módulos de RF a 433 MHz. Cada uno de ellos irá conectado a una placa de Arduino (mediante el Pin 9) de la siguiente manera para poder lograr así una comunicación inalámbrica:

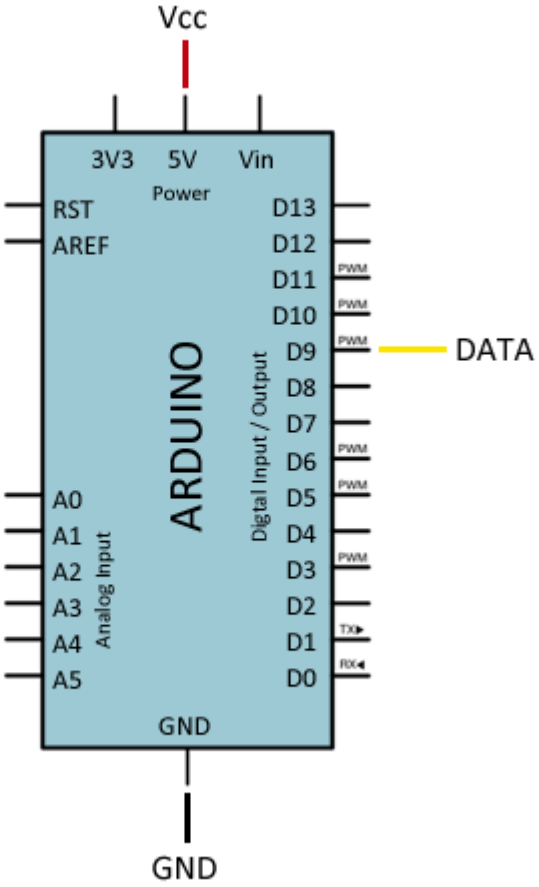
Por tanto, la conexión del emisor FS1000A sería la siguiente,



Y la del receptor XY-MK-5V la que se ve a continuación.



Finalmente, la conexión en ambos casos de los Arduino que controlan cada uno de los módulos emisor/receptor es la misma,



Opcionalmente, podemos alimentar el módulo a tensión superior para aumentar el alcance. Si estáis alimentando Arduino a través de su regulador de voltaje (por ejemplo, desde una batería de litio de 7.4V), podéis emplear esta fuente antes del regulador para alimentar el módulo.

EJEMPLOS DE CÓDIGO

Para realizar la comunicación usaremos la librería Virtual Wire desarrollada por Mike McCauley, disponible en este enlace.
http://www.airspayce.com/mikem/arduino/VirtualWire/index.html

¿Por qué usar una librería en vez de emplear directamente el UART? Para mejorar la robustez de la comunicación. Las transmisiones ASK requieren una serie de impulsos de

“entrenamiento” para sincronizar el receptor y el transmisor. También necesitan de un buen balance entre 1s y 0s para mantener el balance DC del mensaje. La UART no realizan estas funciones.

Con la librería Virtual Wire cada transmisión consta de un código de entrenamiento, el mensaje, la longitud del mensaje, y el checksum. Los mensajes son enviados con codificación 4-a-6 bit para mejorar el balance DC.

Por supuesto, estas mejoras contienen su coste, que en este caso es el espacio que ocupa la librería, la alta carga que supone al procesador, y el uso intensivo que hace de interrupciones. Por otro lado, no se dispone de acuse de señal de recibo (ACK) por lo que no sabemos si los mensajes han llegado correctamente.

La librería proporciona ejemplos de código, que resulta aconsejable revisar. Los siguientes ejemplos son modificaciones a partir de los disponibles en la librería.

ENCENDER UN LED A DISTANCIA

En el primer ejemplo, basado en el que vimos al ver el puerto serie, enciende de forma remota el LED integrado de un Arduino desde otro conectado a un ordenador. El Arduino emisor recibe un byte desde puerto serie y lo reenvía al Arduino receptor. Si se transmite ‘a’ el receptor enciende el LED, y si transmite ‘b’ lo apaga.

CÓDIGO EMISOR

```
#include <VirtualWire.h>

const int dataPin = 9;

void setup()
{
    Serial.begin(9600);
    vw_setup(2000);
    vw_set_tx_pin(dataPin);
}

void loop()
{
    while (Serial.available() > 0)
    {
      char data[1];
      data[0] = Serial.read();
      vw_send((uint8_t*)data,sizeof(data));
      vw_wait_tx();
    }
    delay(200);
}
```

CODIGO RECEPTOR

```
#include <VirtualWire.h>

const int dataPin = 9;
const int ledPin = 13;

void setup()
{
    vw_setup(2000);
    vw_set_rx_pin(dataPin);
    vw_rx_start();

    pinMode(ledPin, OUTPUT);
    digitalWrite(ledPin, false);
}

void loop()
{
    uint8_t data;
    uint8_t dataLength=1;

    if (vw_get_message(&data,&dataLength))
    {
        if((char)data=='a')
        {
            digitalWrite(ledPin, true);
        }
        else if((char)data=='b')
        {
            digitalWrite(ledPin, false);
        }
    }
}
```

ENVIAR UN STRING

El siguiente ejemplo muestra el envío de una cadena de texto desde un Arduino emisor a un Arduino receptor, que al recibir el texto lo muestra por puerto serie.

CÓDIGO EMISOR

```
#include <VirtualWire.h>

const int dataPin = 9;
const int ledPin = 13;

void setup()
{
    vw_setup(2000);
    vw_set_tx_pin(dataPin);
}

void loop()
{
    const char *msg = "Hola mundo";

    digitalWrite(ledPin, true);
    vw_send((uint8_t *)msg, strlen(msg));
    vw_wait_tx();
    digitalWrite(ledPin, false);
    delay(200);
}
```

CÓDIGO RECEPTOR

```
#include <VirtualWire.h>

const int dataPin = 9;
const int ledPin = 13;

void setup()
{
    Serial.begin(9600);
    vw_setup(2000);
    vw_set_rx_pin(dataPin);
    vw_rx_start();
}

void loop()
{
    uint8_t buf[VW_MAX_MESSAGE_LEN];
    uint8_t buflen = VW_MAX_MESSAGE_LEN;

    if (vw_get_message(buf, &buflen))
    {
      digitalWrite(ledPin, true);
      Serial.print("Mensaje: ");
      for (int i = 0; i < buflen; i++)
      {
         Serial.print((char)buf[i]);
      }
      Serial.println("");
        digitalWrite(ledPin, false);
    }
}
```

ENVIAR UNA VARIABLE INTEGER Y FLOAT

El siguiente ejemplo muestra el envío de variables integer y float desde un Arduino emisor a otro receptor, que muestra los datos por puerto serie. El Arduino emisor envía dos datos cualquiera, en el ejemplo millis()/1000 como integer y 3.14 como float, para lo cual tiene que convertirlos en un array de char. Además, añade un identificador ‘i’ o ‘f’ para distinguir el tipo de variable enviado.

Por su parte, el receptor recibe los datos, y en función del identificador convierte los datos recibidos a integer o float, y muestra el resultado por el puerto serie.

CÓDIGO EMISOR

```
#include <VirtualWire.h>

const int dataPin = 9;

void setup()
{
    vw_setup(2000);
    vw_set_tx_pin(dataPin);
}

void loop()
{
   String str;
    char buf[VW_MAX_MESSAGE_LEN];

   // Ejemplo de envío int
   int dataInt = millis() / 1000;
    str = "i" + String(dataInt); /// Convertir a string
    str.toCharArray(buf,sizeof(buf)); // Convertir a char array
    vw_send((uint8_t *)buf, strlen(buf)); // Enviar array
    vw_wait_tx(); // Esperar envio

   // Ejemplo de envío float
   float dataFloat = 3.14;
    str = "f" + String(dataFloat); // Convertir a string
    str.toCharArray(buf,sizeof(buf)); // Convertir a char array
    vw_send((uint8_t *)buf, strlen(buf)); // Enviar array
    vw_wait_tx(); // Esperar envio

    delay(200);
}
```

CÓDIGO RECEPTOR

```
#include <VirtualWire.h>

const int dataPin = 9;

void setup()
{
    Serial.begin(9600);
    vw_setup(2000);
    vw_set_rx_pin(dataPin);
    vw_rx_start();
}

void loop()
{
    uint8_t buf[VW_MAX_MESSAGE_LEN];
    uint8_t buflen = VW_MAX_MESSAGE_LEN;

   // Recibir dato
    if (vw_get_message((uint8_t *)buf,&buflen))
    {
      String dataString;
        if((char)buf[0]=='i')
        {
            for (int i = 1; i < buflen; i++)
            {
            dataString.concat((char)buf[i]);
            }
            int dataInt = dataString.toInt();  // Convertir a int
            Serial.print("Int: ");
            Serial.println(dataInt);
        }
        else if((char)buf[0]=='f')
        {
            for (int i = 1; i < buflen; i++)
            {
            dataString.concat((char)buf[i]);
            }
            float dataFloat = dataString.toFloat();  // Convertir a float
            Serial.print("Float: ");
            Serial.println(dataFloat);
        }
    }
}
```

ANÁLISIS Y PRESENTACIÓN DE RESULTADOS

NOTAS PARA LOS ALUMNOS.

1. El reporte final de la práctica deberá ser entregado a máquina de escribir o en procesador de textos (PC) sin excepción.
2. Las prácticas impresas sólo sirven de guía y referencia.
3. No se aceptan copias fotostáticas del reporte final.
4. La entrega del reporte de práctica es por alumno.

CONCLUSIONES DE APRENDIZAJE.

RECURSOS BIBLIOGRÁFICOS.

http://arduino.cl/que-es-arduino/

http://arduino.cl/descargas/

http://arduino.cl/programacion/