Arduino + XBee - Primeros Pasos

Introducción

Los módulos XBee de MaxStream permiten enlaces seriales de señales TTL en distancias de 30 metros en interiores, 100 metros en exteriores con línea de vista y hasta 1.5 Km con los módulos Pro.

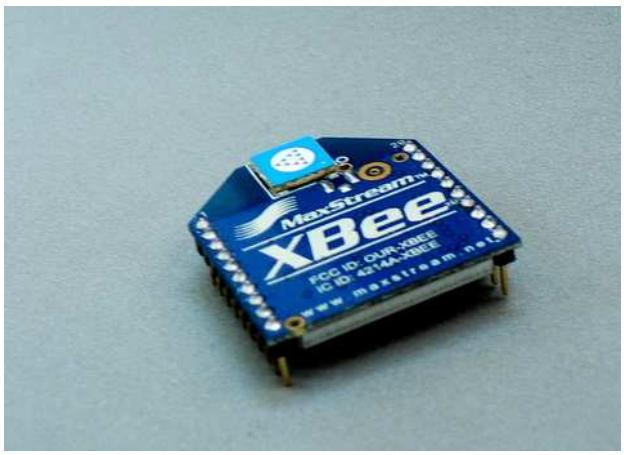

Fig. 1. Módulo XBee de MaxStream

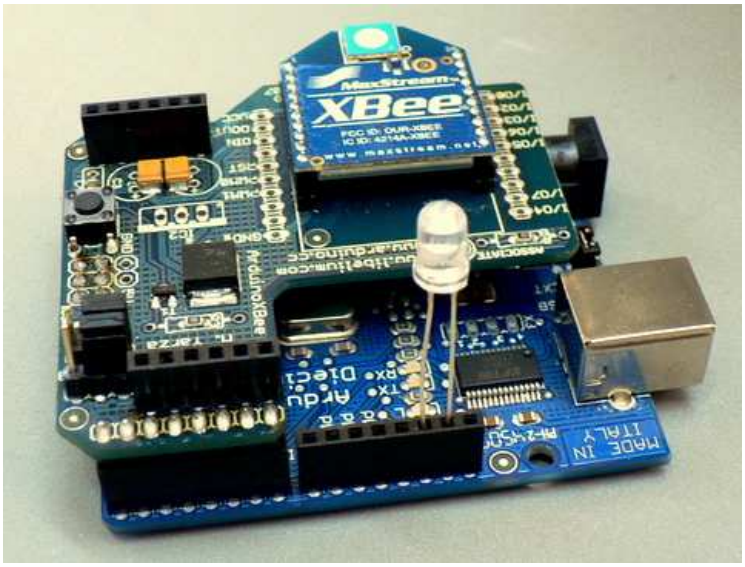

Fig. 2. XBee Shield montado sobre la placa Arduino

Lós módulos XBee utilizan el protocolo IEEE 802.15.4 mejor conocido como ZigBee. Este protocolo se creó pensando en implementar redes de sensores. El objetivo es crear redes tipo mesh que tengan las propiedades de auto-recuperación y bajo consumo de energía.

Tomado de Wikipedia (http://en.wikipedia.org/wiki/Zigbee), las áreas de aplicación son:

- **Entretenimiento en casa y control** – iluminación inteligente, control avanzado de temperatura, seguridad, películas y música.
- **Hogar Prevenido** – sensores de agua, sensores de potencia, electrodomésticos inteligentes y sensores de acceso.
- **Servicios Móviles** – pagos móviles, monitoreo y control móvil, seguridad y control de acceso móvil, cuidados de salud móviles y teleasistencia.
- **Edificios comerciales** – monitoreo de energía, HVAC, iluminación y control de acceso.
- **Plantas industriales** – control de procesos, gestión de ventajas, gestión ambiental, gestión de energía y control de dispositivos industriales.

En nuestro caso particular vamos a utilizar los módulos de XBee para crear una comunicación serial inalámbrica entre una computadora y un Arduino.

Materiales necesarios

- 2 módulos XBee Shield
- 2 placas Arduino, una de ellas SIN el microcontrolador (hay que tener cuidado especial al retirarlo para no doblarle los pines)
- Un LED
- Un eliminador de 9V para alimentar el Arduino remoto
- Un cable USB para conectar el Arduino local a la computadora

Configuración inicial

De fábrica cada módulo XBee viene configurado con un PAN ID (el identificador de la red personal) de 3332 y configurados con una tasa de transferencia de 9600 baudios, con datos de 8 bits, sin paridad y 1 bit de paro.

Cada XBee Shield tiene un par de jumpers para definir si la comunicación serial se realiza hacia el puerto USB o hacia el módulo XBee. Como primera prueba vamos a configurarlos para interactuar con el puerto USB, esto quiere decir poner ambos jumpers en la posición externa de los pines, en la figura 3 puede verse la ubicación de tales jumpers.

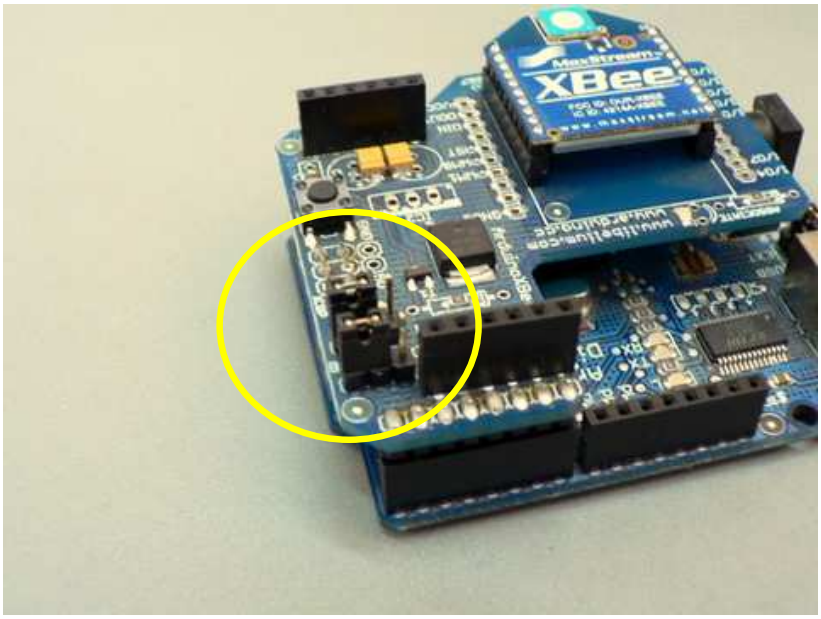

Fig. 3. Ubicación de los jumpers selectores

Podemos utilizar el programa Hyperterminal en Windows para comenzar a configurar el módulo XBee. Otras alternativas son el Serial Monitor que viene integrado en el entorno de programación de Arduino o el programa Bray++ Terminal (http://hubbard.engr.scu.edu/embedded/avr/software/Terminal.exe).

El comando necesario para comenzar la interación es +++, esto es, tres signos de suma consecutivos. Tecleándolos debemos de recibir como respuesta un OK. En la figura 4 podemos ver la entrada en la parte inferior y la respuesta en la parte superior.

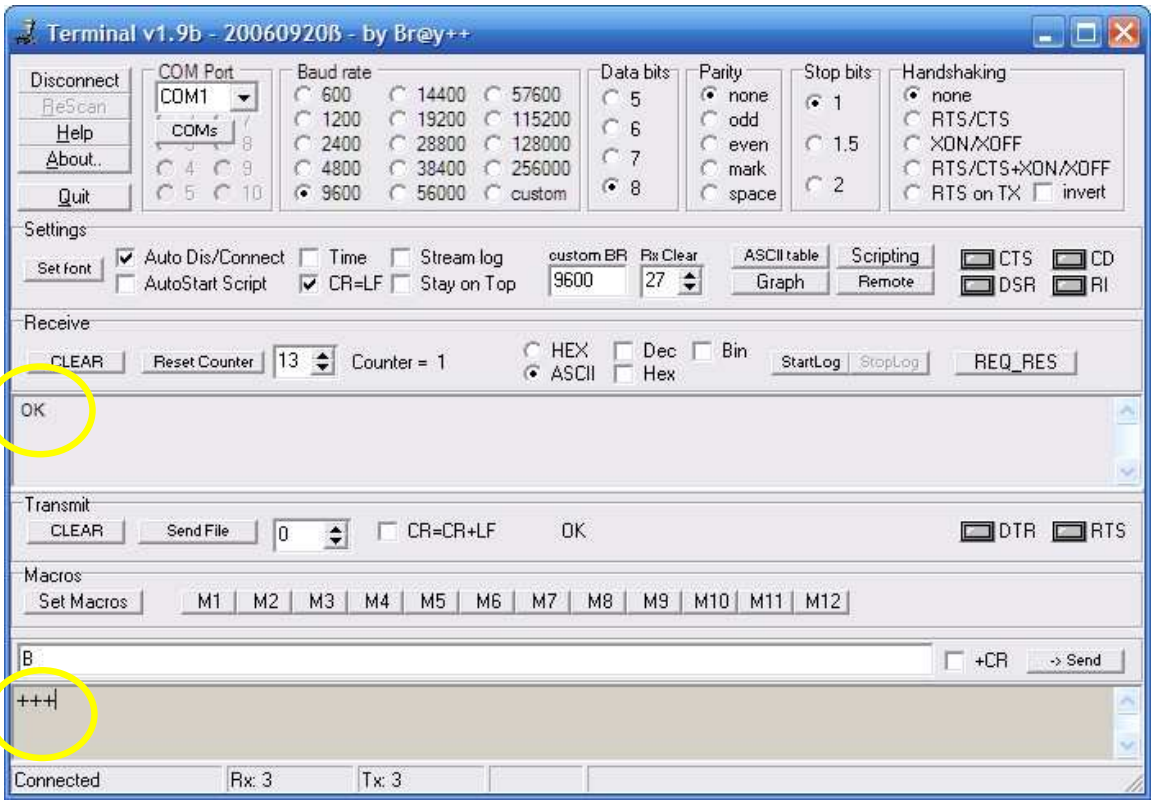

Fig. 4. Estableciendo conexión con el módulo XBee

Una vez establecida la conexión sólo tenemos 5 segundos para interactuar con el módulo. Después de ese tiempo, el módulo regresa a su estado nativo y para volver a interactuar tenemos que teclear +++ nuevamente.

Cada comando que le ingresemos debe ir precedido por las letras AT. Este es el típico modo de operación de módems.

La tabla de comandos más usuales la podemos obtener en
http://www.arduino.cc/en/Main/ArduinoXbeeShield

Usaremos para la conexión a la computadora la placa Arduino SIN el microcontrolador ATMEGA168. La razón de esto es que de esa manera podemos directamente interactuar via USB con el módulo XBee.

Para este ejemplo proponemos la siguiente configuración: un PAN ID de 3332, un módulo con dirección 0 y otro módulo con dirección 1. El módulo con dirección 0 será el que estará conectado a la computadora y el módulo con dirección 1 será el módulo remoto.

Para configurar el primero con la dirección 0, velocidad 9600 8-n-1 y comunicación con el módulo de ID1 usamos el siguiente comando:

ATID3332,DH0,DL1,MY0,BD4,WR,CN

Las letras AT le indican al XBee que le vamos a enviar comandos. ID3332 le define un PAN ID de 3332, DH0 y DL1 definen la direción 01 como el XBee con el que se estará comunicando, el comando MY0 define la dirección propia como 0, BD4 define la velocidad en 9600, el comando WR escribe la configuración a la memoria y el CN cierra la configuración.

Paso siguiente es conectar el segundo XBee shield al Arduino sin micro. Configuramos de la misma manera. ¿Qué cambios se realizaron en el siguiente comando?:

ATID3332,DH0,DL0,MY1,BD4,WR,CN

Después de terminada la configuración ya estamos en condiciones de establecer una comunicación entre los dos módulos XBee.

Programando el Arduino remoto

Vamos a colocar el módulo XBee con dirección 1 en el Arduino que SÍ tiene microcontrolador y lo vamos a programar con un código que envía vía serial una cuenta numérica. También vamos a incluir una lectura del puerto serial para saber si se ha recibido un caracter en particular y en caso afirmativo conmutar el estado del LED conectado al pin 13.

Es importante remover el módulo XBee de este Arduino mientras lo programamos para que podamos establecer la comunicación serial.

```
/* Prueba Serial
   =============

   Este programa se utiliza para enviar una cuenta por
   la conexión serial del Arduino y leer si se recibe
   un caracter para cambiar el estado del LED conectado al pin 13

   Félix E. Guerrero
   Verano 2008

 */

int cuenta = 0;
char recepcion;
int estado = 1;

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

void loop() {

  Serial.print(cuenta);
  Serial.println();
  delay(1000);
  cuenta++;
```

```
  // leer del serial

  recepcion = Serial.read();
  if (recepcion == 'x')
{
  estado = !estado;
  digitalWrite(13,estado);
}
}
```

Después de cargar este código podemos probarlo abriendo una conexión serial hacia el Arduino. Debemos de observar una cuenta ascendente y cuando enviamos el caracter x debemos observar un cambio en el estado del LED conectado al pin 13. NOTA: parece que la primera vez que recibe la x no la reconoce como válida. Hasta la 2da ocasión es cuando comienza a conmutar el LED.

Probando la comunicación inalámbrica

Una vez programado el Arduino lo vamos a desconectar del USB y a colocar en otro sitio, donde podamos alimentarlo con el eliminador de 9V. También vamos a conectarle el módulo de XBee Shield para que pueda comenzar a transmitir inalámbricamente la cuenta ascendente. Debemos asegurarnos que los jumpers estén hacia el interior del Arduino, es decir, en la posición XBee.

En la computadora vamos a conectar al USB el Arduino SIN microcontrolador con el XBee Shield de dirección 0 y vamos a abrir el Hyperterminal. Después de unos segundos debemos de empezar a observar la cuenta ascendente y cuando tecleamos la x debemos de lograr que el estado del LED cambie.

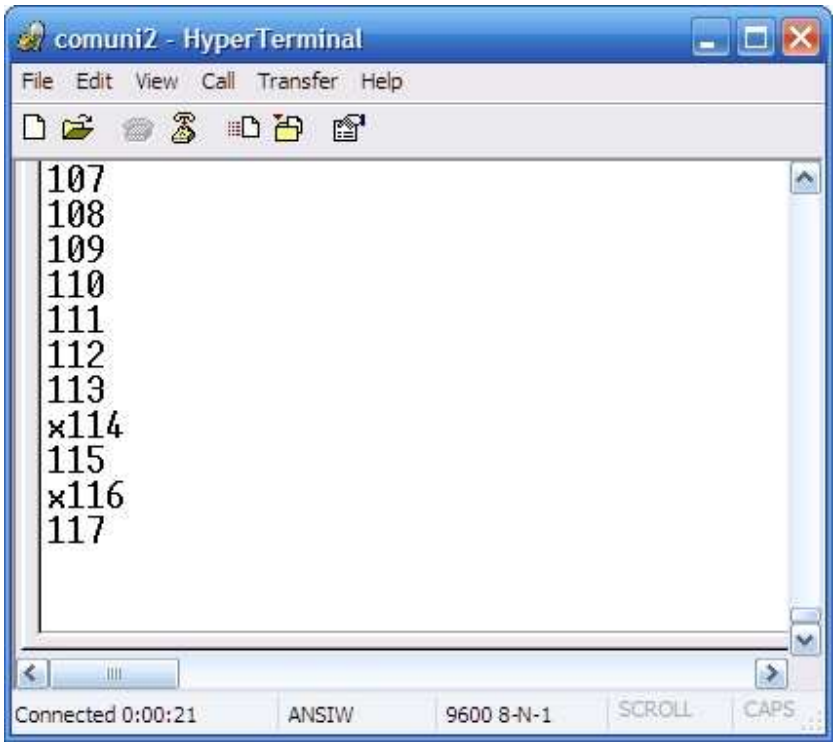

Fig. 5. Recibiendo datos inalámbricamente

Actividades Sugeridas

- Conectar un Actuador Universal (el seguro eléctrico de los coches, menos de 50 pesos) y accionarlo remotamente para asegurar o liberar una puerta o ventana
- Leer y registrar temperatura remotamente
- Montar una red completa donde la información se transmita por todos los nodos (hay que leer cómo implementar una red mesh)
- Controlar el encendido y apagado de un motor para abrir y cerrar persianas
- Transmisión y recepción inalámbrica de datos MIDI
- Control de iluminación para escenarios via MIDI sin cables
- Monitoreo remoto de nivel de un tinaco
- Monitorear tinacos de un conjunto habitacional completo
- Conteo de accesos a través de una puerta
- Actualización inalámbrica de las imágenes para un sistema POV en rines de automóvil
- Control inalámbrico de un brazo de robot utilizando un guante

Referencias:

Guía rápida - http://www.arduino.cc/en/Guide/ArduinoXbeeShield

Descripción detallada - http://www.arduino.cc/en/Main/ArduinoXbeeShield

Versión 2 del XBee Shield - http://www.arduino.cc/cgi-bin/yabb2/YaBB.pl?num=1208291234

Entrada en la Wikipedia para el protocolo ZigBee - http://en.wikipedia.org/wiki/Zigbee

Página del fabricante (Digi, antes MaxStream) - http://www.digi.com/products/wireless/point-multipoint/xbee-series1-module.jsp

Procedimiento para actualizar el firmware - http://itp.nyu.edu/~raf275/meshnetworking/XBee/XBee_firmware_upgrade.html

Ejemplo con PIC 18F452 - http://itp.nyu.edu/~raf275/meshnetworking/XBee/XBee_example.html

Cómo programar un Arduino de manera inalámbrica - http://itp.nyu.edu/~raf275/meshnetworking/XBee/XBee_program_Arduino_wireless.html
Proyectos de Rob Faludi que involucran XBee - http://itp.nyu.edu/~raf275/cgi-bin/mt/mt-search.cgi?IncludeBlogs=3&search=xbee

El libro de Tom Igoe *Making Things Talk* - http://itp.nyu.edu/~raf275/cgi-bin/mt/mt-search.cgi?IncludeBlogs=3&search=xbee

El manual del usuario de MaxStream - http://www.makingthings.com/resources/datasheets/manual_xb_oem-rf-modules_802.15.4.pdf

Un artículo muy completo por Fred Eady para Circuit Cellar - http://www.circuitcellar.com/library/print/0906/Eady194/Index.htm

Tutorial para módulos XBee con el Make Controller - http://www.makingthings.com/documentation/tutorial/xbee-wireless-interface/tutorial-all-pages

Incluye un excelente comentario en direccionamiento:

Addressing

How to make sure your messages go where you want.
There are 3 values that allow you to organize your XBee module address space:

1. Individual module addresses
2. PAN (Presonal Area Network) IDs
3. Channels

For two modules to communicate, they must be on the same channel, have the same PAN ID and the destination address of the sender must match the address of the receiver.

Each XBee module has its **own unique address**, as well as a **destination address** that it sends its messages to. The destination address can specify a single destination or it can be a broadcast address, which will be received by all XBee modules within range. The broadcast address is **65535** (0xFFFF in hex).

If you only have a couple modules, you'll probably never need to change the PAN ID or channel. But if you're in an evironment with a bit more traffic, you might want to be sure that nobody else's messages are getting mixed up with yours, so this offers a nice option for that.

Unique and Broadcast Addresses

XBee modules can be used with 16-bit addresses or 64-bit addresses. 16 bits already gives you 65536 possible unique values, but hey you never know when you might need number 65537, right? Any time a module's address is less than 0xFFFE, it will use 16-bit addresses. When the module's address is set to 0xFFFF or 0xFFFE, it will use a 64-bit address derived from its internal serial number.

There are 2 values that configure the destination address - **DH** (Destination High) and **DL** (Destination Low). If DH is set to 0, as it is by default, the board will send to a 16-bit address. To send broadcast messages, which will be received by all modules on that channel and PAN, set DH 0x0000FFFF to and DL to 0x00000000 (default value).