Trabajo Fin de Master

Centro Superior de Enseñanza Musical
" *Katarina Gurska*"
Madrid

Master en Composición Electroacústica

SENSOPHONE

El saxofón Ampliado

Realizado por

Fco Javier Rodríguez García

bajo la direccion de Sergio Luque

2014

1 INDICE

2 AGRADECIMIENTOS

Antes de comenzar con el contenido de este trabajo cabría hacer una mención especial para agradecer a todas aquellas personas que han colaborado en el desarrollo de este proyecto y que han supuesto en muchos casos un pilar fundamental en la construcción de algunos de sus apartados:

- A Sergio Luque, por su labor tutorial impecable. Exigente, constante y motivador, ha sido el elemento fundamental desde el momento de asentar las bases y perfilar las actuaciones a llevar a cabo a lo largo de todo el proceso hasta la revisión final, así como en la adaptación del sistema a Supercollider.

- A Alberto Bernal. Por su innumerable aporte de ideas en muchos de los apartados del trabajo así como su eficacia y predisposición en la propuesta, desarrollo y solución de problemas para la comunicación del dispositivo con Max/MSP.

- A José Antonio Rodríguez. Por su impagable colaboración en el diseño, asesoramiento y construcción de todas las piezas necesarias para la instalación de los sensores, pulsadores, placa etc. en el saxofón.

- A Honorino Fernández. Por su conocimiento del medio, predisposición y profesionalidad en el asesoramiento de la parte más técnica del proyecto. Manejo de los sensores, programación de la placa, materiales etc.

- A Miguel Ángel Fernández. Por transmitir su pasión y conocimientos por este medio de manera tan positiva y motivadora y por su generosidad y naturalidad en el aporte de ideas y soluciones.

- A Manuel Fernández (Mafer-Música). Por su atenta colaboración en la obtención de información acerca de los antecedentes del saxofón como controlador interactivo.

3 INTRODUCCIÓN

El avance de la tecnología en los últimos tiempos ha sido inherente al desarrollo del mundo digital, si entendemos el término como aquellos sistemas que utilizan el lenguaje binario.

Atrás quedó ya aquella idea en el que todo lo relacionado con la computación, programación, control de sistemas etc. era tarea de unas pocas personas altamente cualificadas. La tecnología ha ido encaminada a hacer que ésta sea cada vez más accesible, comprensible y útil para cualquier usuario. El mundo del arte y por extensión el de la música, no ha sido ajeno al cambio hacia la era digital. Mundo real y digital comparten cada vez más espacios, siendo difícil en ocasiones distinguir donde acaba uno y empieza otro.

Este punto en el que nos encontramos es el que ha inspirado este trabajo, cuyo objetivo principal era encontrar ese espacio común en el que el sonido acústico y sonido digital se diesen la mano de una manera desconocida hasta el momento. Para ello aprovecharíamos las excelentes cualidades sonoras que nos ofrece un instrumento acústico como es el Saxofón y por otro, la apertura del mercado de los microcontroladores y sensores que derivan del proyecto Arduino.

La relación entre los instrumentos acústicos y el mundo del sonido electrónico y digital no es nada nuevo, pero hasta ahora la combinación de ambos requería la implicación de varias personas (el instrumentista y el encargado de controlar la parte electrónica-digital, máxime si hablamos de cambios de procesos en tiempo real). Imaginamos que fuesen necesarias dos personas para tocar un violín, una persona encargada de frotar el arco y otra para la digitación. Quizás resulte exagerado aplicar esta disociación a la relación entre el intérprete y el encargado de manejar los procesos electroacústicos, pues se puede entender igual que la relación entre dos músicos interpretando música de cámara, no obstante, es difícil pensar que el nivel de sincronía y sintonía que alcanza una sola persona al interpretar su instrumento pueda ser igualado si éste fuese tocado por dos personas. Por esto nos planteamos la posibilidad de diseñar un dispositivo que ofreciese a

una misma persona la capacidad de tocar su instrumento y manejar los procesos de transformación del sonido en tiempo real sin renunciar a las cualidades sonoras de éste ni al modo de ejecución.

Hablaremos en los siguientes apartados de los elementos que configuran nuestro proyecto, su funcionamiento y el camino que hemos seguido para la construcción y puesta en marcha de los mismos.

4 ANTECEDENTES DEL SAXOFÓN COMO CONTROLADOR ELECTROACÚSTICO

No habrá que esperar mucho para encontrar un ejemplo en el que las cualidades acústicas del saxofón se vean alteradas por dispositivos y procedimientos electrónicos. La popular marca de saxofones *Selmer*, en 1967 lanza al mercado su proyecto "Varitone". El "Varitone" es un dispositivo formado por un micrófono pick-up instalado en el tudel, una caja multiefectos ubicada a través de una plataforma sobre las protecciones de las llaves *B* y *Bb* y un amplificador. La caja de efectos procesaba la señal procedente del micrófono y podía producir trémolos, aplicar una ecualización básica (claro y oscuro), octavas simultáneas y efectos de eco. La señal procesada era enviada al amplificador. Las posibilidades sonoras de este sistema diseñado para un saxofón tenor Selmer Mark VI fueron demostradas a un gran nivel por el saxofonista Eddie Harris. Proyecto ambicioso por los medios técnicos disponibles en la época se desconoce la verdadera razón por la que este sistema no ha trascendido hasta nuestros días. Los problemas derivados de la ubicación del micrófono, el elevado coste añadido a la compra del propio instrumento o la reticencia de los propios saxofonistas de alterar la sonoridad de uno de los saxofones más laureados de la historia (modelo Mark V) parecen ser algunas de las razones.

La asociación de la palabra saxofón con el mundo de la electrónica tiempo más tarde fue por otros derroteros. En los años 80, con el desarrollo del protocolo MIDI, aparecen varios controladores de marcas como Yamaha (WX5), Akai (EWI) o Casio (Dh-100). A pesar de que poco tienen que ver con un saxofón convencional al margen de la forma del modelo Dh-100 de Casio,

adquieren el sobrenombre de *Saxofón electrónico* por su similitud en la disposición de botones controladores y modo de ejecución.

De la mayoría de proyectos que vamos a citar en este capítulo hemos tenido conocimiento después de plantearnos los objetivos y líneas básicas del proyecto que ocupa este trabajo, pero han sido una referencia a tener en cuenta para poder perfilar con más detalle nuestro proyecto, superando sus inconvenientes e intentando no caer en las deficiencias que desde nuestro punto de vista éstos pudiesen tener.

El primer ejemplo del que hablaremos en el que un saxofón convencional es transformado electrónicamente para enviar datos MIDI a un sintetizador o un pc es el *Synthophone*[1]. Proyecto cuyo primer prototipo, creado por Martin Hurni, se remonta al origen del lenguaje MIDI en 1981. Consiste en un avanzado controlador de aire compuesto por una boquilla especial combinada con un sensor adherido a una caña convencional(tratada) que recogen el flujo de aire del intérprete. El sistema mecánico del saxofón Yamaha-280 al que se le ha instalado un conjunto de sensores adheridos a cada llave, servirá de controlador. En realidad, en ningún momento escucharemos el sonido acústico del saxofón. Su funcionamiento está mucho más cerca al de los controladores Yamaha (WX5), Akai (EWI) etc. La idea de incorporarlo sobre un saxofón real hace que su manejo resulte ergonómicamente mucho más familiar al intérprete. A través de combinaciones y posiciones especiales en la digitación se puede cambiar la fuente de sonidos y efectos del sintetizador desde el propio saxofón sin necesidad de tener el verdadero generador de sonidos (sintentizador) cerca del intérprete.

Metasax[2] es un proyecto llevado a cabo en 1999 por el norteamericano Matthew Burtner en el Centro para la Investigación Computacional en Música y Acústica (CCRMA). Un conjunto de sensores ubicados a lo largo de todo el instrumento son procesados por una placa dotada de un microcontrolador que envía los datos recogidos por los diferentes sensores vía MIDI al ordenador. El sistema cuenta con tres micrófonos ubicados tanto dentro como fuera del tubo, encaminados a recoger la resonancia en diferentes puntos del instrumento.

---

[1] [en línea]. Disponible en http://www.synthophone.info/[consulta el 28/11/14]
[2] [en línea]. Disponible en https://ccrma.stanford.edu/~mburtner/metasax.html [consulado 25/11/14

La información de los sensores y valores de los micrófonos relativos a la intensidad, altura y otras cualidades del sonido son enviados al pc para llevar a cabo la programación de distintos procesos electroacústicos.

5 EL PROYECTO ARDUINO

Arduino es un proyecto electrónico de hardware y software libre que nace en el año 2005 en el Instituto de Diseño de Interacción IVREA, Italia, con la idea de diseñar una interface que permitiese a los estudiantes de éste llevar a cabo sus proyectos mediante una placa controladora mucho más económica que la que utilizaban hasta el momento. El proyecto no solo sobrevive al propio cierre del instituto IVREA sino que se convierte con el tiempo en el sistema con el que usuarios de todo el mundo diseñan, programan y ejecutan sus proyectos interactivos basados en sensores, luces, motores, etc.

El equipo de trabajo dirigido por Massimo Banzi, entre los que se encuentra el español David Cuartielles, diseñan una placa integrada dotada de entradas y salidas digitales y entradas analógicas, controladas por un microcontrolador “Atmega”. La programación del controlador se lleva a cabo a través de un lenguaje propio de programación Arduino basado en Processing/Wiring, que a su vez parten de los lenguajes de programación C y C++.

La filosofía del proyecto Arduino pretende que cualquier usuario pueda llevar a cabo sus proyectos sin necesidad de tener una formación avanzada en ingeniería de sistemas, electrónica o programación. Tanto el lenguaje de programación, simplificado en relación a los lenguajes de los que parte (C/C++), como el modo de funcionamiento de la placa para la conexión y obtención de datos, hacen que un usuario con una formación básica pueda interactuar con el mundo que le rodea ,con un sistema de codificación de datos con los que poder operar. Esto supone una nueva herramienta al mundo del arte y del diseño con la que poder llevar a cabo proyectos interactivos de todo tipo.

Manteniendo la filosofía *hardware/software* libre, han sido muchas las personas, empresas y entidades las que han colaborado en el desarrollo del proyecto en los últimos años. Desde sus inicios, el desarrollo de éste ha dado

como fruto diversas versiones de la placa original *Arduino Uno,* con el fin de satisfacer las necesidades de los diferentes usuarios en función de sus proyectos.

A partir de *Arduino Uno* se han desarrollado un catálogo de placas basadas en los mismos principios pero con características diferentes, con el fin de adaptarse a las necesidades de los usuarios. En este momento podemos hablar de una variedad de 20 placas Arduino diferentes encaminadas a ofrecer:

- un mayor número de conexiones de pines tanto analógicos como digitales.
- diferentes modelos del microprocesador Atmega con mayor capacidad de proceso.
- diferentes tipos de conexiones – serie, USB, miniUSB, etc.
- implementación a la placa de un sistema de comunicación de red – ethernet, wifi o bluetooth.
- placas de mayor o menor tamaño, con alimentación independiente, más ligeras con posibilidad de adherir a la ropa, etc.

Toda una variedad de placas cuyas características podemos consultar en la propia página de Arduino[3].

Hablaremos mas detenidamente de la placa que nos va a servir de vehículo para gestionar y enviar al pc la lectura de los sensores y pulsadores de nuestro proyecto, *Arduino Fio*.

5.1 ARDUINO FIO. CARACTERÍSTICAS Y PROGRAMACIÓN

La elección de la placa ha sido uno de los puntos pilares en la selección del material para el desarrollo del proyecto. Ésta condiciona el número de conexiones analógicas y digitales de las que podemos disponer, el tamaño, la conectividad, etc.

Su reducido tamaño, la posibilidad de insertar un sistema de comunicación inalámbrico preinstalado, el aumento de dos pines analógicos con respecto a *Arduino Uno* (8 en lugar de 6), y la posibilidad de tener un sistema de alimentación autónomo, hacen que sea éste el modelo que mejor se

[3] [En línea]: Disponible en: http://arduino.cc/en/Main/Products [consultada 20/06/14]

ajusta a las necesidades del proyecto. Nos permite sobre todo, poder insertar el módulo que procesa los datos en el propio instrumento.

Arduino Fio está dotada de un microcontrolador ATMEGA328P que funciona a 3,3 V y a 8 Mhz. Dispone de un total de 14 pines de entrada/salida digitales (6 de los cuales se pueden usar como salidas PWM[4]) y 8 pines de entrada analógicos.

Puede alimentarse a través de una bateria LiPo de dos pines que se puede cargar a través del puerto USB mini-b. También está dotada de un interruptor *on/off*, (no todas lo incorporan) y un botón de *reset*.

Para llevar a cabo la comunicación serial que nos permite programar la memoria flash del microcontrolador, necesitamos utilizar un adaptador conectado a los puertos *GND, AREF, 3V3, RXI, TXO Y DTR* que vienen identificados en la fila de pines a continuación de los puertos analógicos. Nosotros hemos optado por el adaptador "FTPI Basic Breakout" como alternativa al USB-serie oficial.

A diferencia de gran parte de la gama, funciona solamente a 3.3 V. La mayoría ofrece la posibilidad de trabajar a 5 V o 3.3 V.

La programación se llevará a cabo a través de un entorno de programación propio escrito en Java y Processing, avr-gcc y otro programas de código abierto. Software que podemos descargar de manera gratuita a través de la propia página Arduino[5].

Instalado el software, procederemos a abrir y cargar en la placa el código que se facilita como anexo en este trabajo.

Explicamos brevemente los pasos que debemos de seguir para grabar nuestro código en la memoria flash de la placa.

[4] Salidas PWM "Pulse With Modulation" Modulación de Ancho de pulso. Pines digitales que se comportan como puertos analógicos hembra. Los puertos PWM, en lugar de una señal continua, pueden emitir una señal cuadrada formada por pulsos de frecuencia constante en la que al variar la duración de dichos pulsos, varía la tensión promedio de salida del puerto. Genera así valores de voltaje de salida variables, pudiendo variar la intensidad de luz de un led, las revoluciones de un motor u otros elementos que lo requieran. Tiene una resolución de 8 bits, por lo que pueden manejar valores de 0 a 255.

[5] [En linea]: Disponible en: http://arduino.cc/es/Reference/HomePage[consultada 20/06/14]

Ejecutamos el software Arduino y se abrirá el entorno de programación.

Imagen 1

Desde esta consola podemos escribir nuestro nuevo código, verificarlo, cargarlo en nuestra placa Arduino, salvarlo o recuperar un código que hayamos guardado o descargado previamente y subirlo. Para cargar el código en la placa vamos a seguir los siguientes pasos.

1. En la parte superior del entorno de programación Sketch pinchamos sobre el icono que dibuja una flecha hacia arriba, y de las opciones disponibles, seleccionamos "*Abrir*".

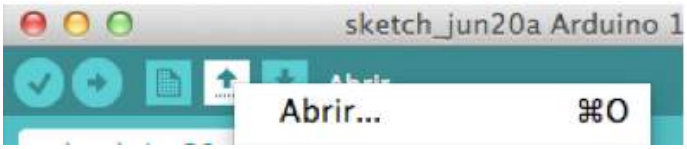


Imagen 2

También podemos ir a la barra de tareas situada en la parte superior de la pantalla y seleccionamos, *Archivo/Abrir*.

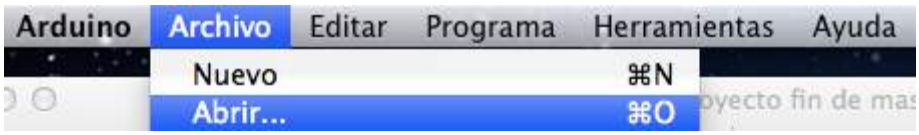


Imagen 3

Abrimos el árbol de archivos de nuestro disco duro y nos dirigiremos a la ruta en la que se encuentra el archivo con nuestro código. Una nueva ventana Sketch aparecerá en la que veremos el texto que contiene nuestro código. Una vez abierto el texto procedemos a cargarlo en la memoria flash de nuestra placa.

Imagen 4

2. Antes de cargar el código, deberemos de haber conectado nuestra *Arduino Fio* al equipo a través del adaptador FTDI, conectado tal y como habíamos citado en el apartado anterior, a los puertos (GND, AREF, 3V3, RXI, TXO Y DTR), marcados de manera clara con estas siglas en la placa, tal como se muestra en la imagen.

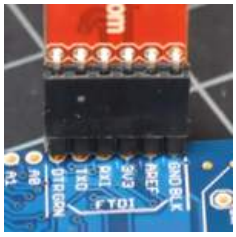


Imagen 5

El adaptador FTDI[6] se conectará por un lado a los seis pines que se corresponden con los pines marcados en la placa y por el otro lado, a través de un cable USB a microUSB, a nuestro equipo. Una vez conectada al pc, pasamos a configurar el software.

Imagen 8.Adaptador FTDI

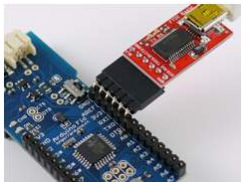


Imagen 9. Conexión FTDI al PC

Imagen 10
Cable de conexión

[6] Debemos de conectar nuestra placa a través del adaptador FTDI al pc sin el módulo XBEE insertado en la placa. De lo contraría el proceso de grabado del código en la placa dará error.

Existen diferentes tipos de adaptadores FTDI. Nosotros hemos utilizado el fabricado por *Sparkfun Electronic.*

Más información acerca de la programación de la placa y materiales necesario en la página web oficial del proyecto Arduino[7].

Antes de cargar nuestro código a la placa debemos de realizar varias configuraciones dentro del software de Arduino en el que le indicaremos que estamos trabajando con una placa *Arduino Fio* y el puerto por el que nos vamos a comunicar. Para ello, iremos al menú "*Herramientas\Placa*" del software Arduino y del listado de placas disponibles seleccionamos aquella con la que queramos trabajar.

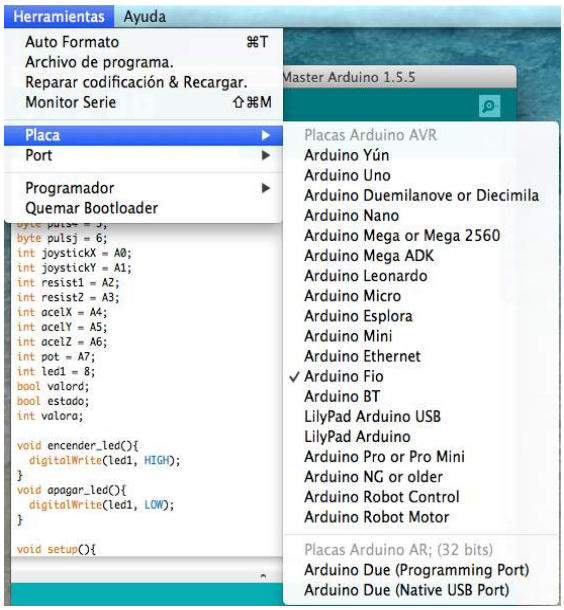


Imagen 11

Una vez seleccionada la placa y de nuevo en el menú *Herramientas*, seleccionamos el submenú *"Puerto".* En este caso, deberemos de saber el nombre del puerto serie por el que nuestro equipo se comunica con la placa. Esto variará dependiendo de si trabajamos en un sistema operativo Windows o Mac OS. En ambos casos, la comunicación por el puerto serie con nuestra placa, necesitará de la instalación de los drivers para la comunicación serial por USB y del convertidor FTDI. Deberemos de tener en cuenta la versión de

[7] [En línea]: Disponible en: http://arduino.cc/en/Main/ArduinoBoardFioProgramming [Consulta 06/08/]

nuestro sistema operativo. Podemos descargar los drivers que se ajusten a nuestro sistema operativo en la web FTDI Chip[8].

Las imágenes capturadas como ejemplo en este trabajo está hechas en Mac OS X.

En el menú "Herramientas" del software Arduino en Windows, el listado *Placa* está bajo el submenú "*Tarjeta",* así como la selección de puerto que como podemos ver en la imagen se selecciona bajo el submenú "*Port*" y en Windows aparece con el término "*Puerto Serial".*

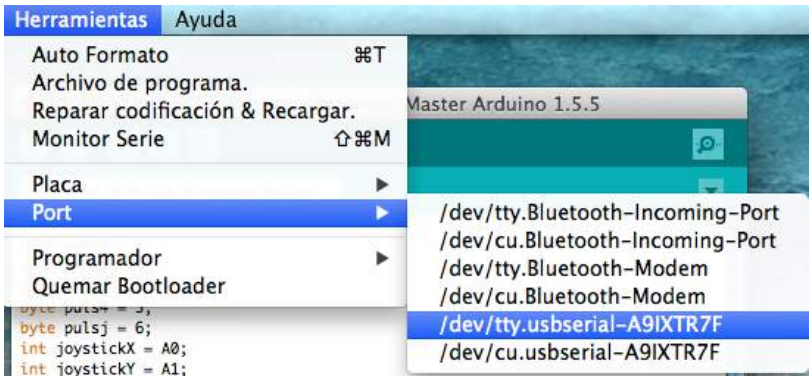


Imagen 12

Si no sabemos con seguridad, cual es el nombre del puerto que se corresponde con el dispositivo que nos interesa, podemos hacer una simple comprobación. En Windows, si no tenemos ningún periférico que utilice el puerto serie antes de conectar nuestra placa al pc, en el submenú "*Puerto Serial"* no nos saldrá ninguna opción. Una vez conectado nuestro dispositivo, comprobaremos como en el submenú nos aparece el nombre que el sistema ha asignado a nuestro dispositivo y que será del tipo "COM1, COM2, COM3 etc.".

En Mac, aun no habiendo ningún dispositivo conectado, podremos comprobar como en el submenú "*Port*" del menú "Herramientas", nos aparecerán varios puertos relacionados con el Bluetooth del equipo.

Cuando conectemos nuestra placa veremos como aparecen varias opciones más, */dev/tty.usbserial-A9IXTR7F* y */dev/tty.usbserial-A9IXTR7F*. Para trabajar seleccionaremos la primera de ellas.

En caso de tener algún dispositivo más conectado, podemos proceder de igual forma. Comprobamos el listado del submenú "Port", antes de conectar

[8] [En línea] Disponible en la url: http://www.ftdichip.com/FTDrivers.htm [Consulta 06/08/2014]

nuestra placa. Una vez hagamos la conexión, nos aparecerá en el listado el nombre del puerto que se corresponde con nuestro dispositivo.

Configurado el tipo de placa y puerto en el software Arduino, estaremos preparados para subir el código a la placa.

Pulsamos el icono *Subir.*

Imagen 13

En la barra de estado de la parte inferior de la ventana y si todo ha ido correctamente veremos la secuencia. “Compilando programa- Subiendo - Subido”.

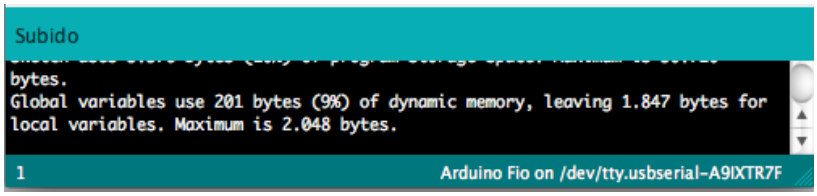


Imagen 14

Si vemos Subido y las letras que aparecen en la franja negra está en blanco y no en naranja o rojo, significa que el proceso de subida se ha realizado correctamente.

Imagen 15

Ya tendremos nuestro código cargado en la placa y por tanto configurada para funcionar conforme al proyecto. Una vez cargado el código, la placa comienza a funcionar automáticamente (emitir, leer, enviar) según se haya programado.

5.2 SISTEMA DE COMUNICACIÓN INALÁMBRICA XBEE

La comunicación inalámbrica XBEE está basada en la conexión de módulos dotados de un emisor-receptor Zigbee[9].

Hay dos tipos de módulos. Serie 1 y Serie 2[10]. Utilizan chips y protocolos de comunicación distintos y lo importante que debemos de saber es que no podemos comunicar módulos de la serie 1 con módulos de la serie 2.

Imagen 16

2. Ambos soportan lo que en comunicación de redes se denomina comunicación punto a punto o punto a multipunto y la configuración en ambos casos va a ser igual para estos modos, por lo que la elección para nuestro trabajo es indiferente.

5.2.1 MATERIALES

Los materiales necesarios para la comunicación inalámbrica de nuestra Arduino Fio con nuestro equipo serán:

- Dos módulos XBEE serie 1 o serie 2. Uno conectado en el zócalo que la Fio lleva por defecto para este tipo de módulos, que servirá de emisor y otra conectada a través del adaptador XBEE Xplorer USB a nuestro pc, que hará de receptor.

Imagen 17

- Un adaptador XBEE Xplorer USB. Nos servirá tanto para la programación de los módulos XBEE como para establecer la conexión del segundo módulo al pc, que servirá de receptor.

Imagen 18

[9] Zigbee es un protocolo de comunicaciones inalámbrico basado en el estándar de comunicaciones para redes inalámbricas IEEE_802.15.4. Creado por Zigbee Alliance, una organización, teóricamente sin ánimo de lucro, de más de 200 grandes empresas (destacan Mitsubishi, Honeywell, Philips, Motorola, Invensys, ...) , muchas de ellas fabricantes de semiconductores. “Arduino + XBee”
Implementación de Sistemas de Trasmisión de Datos y Sensores en Redes Inalámbricas con XBee integrado en la “Plataforma Open Hardware” Arduino. José Manuel Ruiz Gutiérrez. Pag. 4.

[10] La serie 1 está basada en el chipset Freescale y está pensado para ser utilizado en redes punto a punto y punto a multipunto. Los módulos de la serie 2 están basados en el chipset de Ember y están diseñados para ser utilizados en aplicaciones que requieren repetidores o una red mesh(malla).

Imagen 19

- Cable USB a miniUBS. Para conectar el adaptador XBEE Xplorer USB con el módulo XBEE acoplado a nuestro equipo.

5.2.2 PROGRAMACIÓN XBEE

Para la programación o configuración de los módulos XBEE vamos a utilizar el programa X-CTU que podremos descargar de la web http://www.xbee.cl en el apartado *"Descargas"*[11].

Con X-CTU podremos hacer una lectura de los parámetros de configuración del módulo XBEE, modificarlos y grabarlos en su memoria. Para ello, el programa nos da la posibilidad de realizar tales tareas desde un entorno gráfico o a través de consola de comandos.

La programación por comando a menudo resulta más rápida y efectiva. De todos los parámetros susceptibles de ser modificados en función del tipo de comunicación que queramos llevar a cabo, nos centraremos en los que describimos a continuación, el resto de valores los dejaremos por defecto.

Estos parámetros serán los necesarios para establecer nuestra conexión serial punto a punto.

- ID. Identificador de la red que vamos a crear. Los dos módulos deben de tener el mismo ID que se expresará en el formato *0 - 0xFFFF.*

- MY. Dirección del módulo. Con posibilidad de expresarlo hasta en 16 bit, *0 - 0xFFFF*[12]*.*

- DH (High Direction). Dirección de destino para comunicación inalámbrica Alta. Hasta 32 bit, 0 - 0xFFFFFFFF.

- DL (Low Direction). Dirección de destino para comunicación inalámbrica baja. Hasta 32 bit, 0 - 0xFFFFFFFF.

Estos son los parámetros básicos para nuestra configuración punto a punto y funcionan siguiendo la siguiente lógica.

[11] [En línea] Disponible en la url: http://www.xbee.cl/descargas.html (consulta 07/08/2014)
[12] El número de bit y su representación en lenguaje hexadecimal. Determina el rango de valores que podremos utilizar en cada parámetro.

En ambos módulos debemos de poner el mismo ID con el que indicamos que ambos pertenecen a la misma red. Por ejemplo *1234*. Asignamos una dirección (*MY*) diferente a cada módulo dentro de un mismo rango. Por ejemplo *0* para el modulo 1 y *1* para el módulo 2. Ahora debemos de indicar a cada módulo, con que dirección se va a comunicar. Para ello, a través de la dirección de comunicación inalámbrica *DH* y *DL*, en cada módulo pondremos como *DL* la dirección MY del otro. Vemos en la tabla de la imagen 20, como quedarían configurados estos valores en cada terminal XBEE.

Modulo XBEE 1		Modulo XBEE 2	
Parámetro	Valor	Parámetro	Valor
ID	1234	ID	1234
MY	0	MY	1
DH	0	DH	0
DL	1	DL	0

Imagen 20

Éstos son valores a modo de ejemplo. Pueden ser sustituidos por aquellos que cualquier usuario estime oportuno siguiendo la lógica descrita anteriormente.

Pasamos ahora a hablar de otros parámetros que también debemos de tener en cuenta.

- DB. Velocidad de transmisión expresada en *baudios*. Los valores estándar que se suelen utilizar para la comunicación serial entre las placas Arduino y el pc son los siguientes. El valor más común es 9600 bps. En nuestro proyecto, y realizadas las pruebas pertinentes optaremos por la opción de 115200 bps, velocidad estándar para el protocolo Bluetooth.

Opción	Baudios por segundo
0	1200 bps
1	2400 bps
2	4800 bps
3	9600 bps

4	19200 bps
5	38400 bps
6	57600 bps
7	115200 bps

Imagen 21

- RE. Restaura el módulo a sus valores por defecto.

- WR. (Write). Importantísimo este comando. Debemos de utilizarlo después de realizar los cambios que hayamos realizado para grabar los datos en la menoría del módulo. Sin este comando, todas las modificaciones realizadas no surtirán efecto.

Para leer, modificar y grabar los datos expuestos anteriormente en nuestros módulos XBEE a través del entorno gráfico del programa X-CTU.

1. Conectamos el módulo que queramos programar al adaptador Explorer XBEE USB y éste a su vez al equipo.
2. Abrimos el programa X-CTU. Con el dispositivo conectado pulsamos el icono para buscar dispositivo. Nos saldrán varios menús para filtrar la búsqueda.

“Discover Radio Devices” para marcar el tipo de dispositivos inalámbricos que queremos buscar. Marcaremos la opción userial-A603…… , pulsamos *next*. En la siguiente pantalla marcamos otros parámetros de filtrado como los Baud Rate (velocidad de comunicación en baudios), Data bit (número de bit de datos, *Stop Bits*, *Parity* y *flow controls*). Estos últimos no demasiado relevantes.

3. Pulsamos Finish y comenzará a buscar dispositivos. Pasados unos segundos aparecerán en la siguiente pantalla el resultado de la búsqueda con nuestro modulo conectado, en el que podremos ver el nombre de puerto serie y la dirección *mac* del dispositivo. Marcamos el dispositivo y pinchamos sobre “Add selected devices”.
4. En la venta principal del programa, parte izquierda veremos nuestro módulo en la sección “Radio Modules”. Si hacemos doble clic sobre éste, comenzará la lectura de parámetros de configuración cuyo resultado nos aparecerá en la parte derecha de la pantalla. “Radio Configuración.

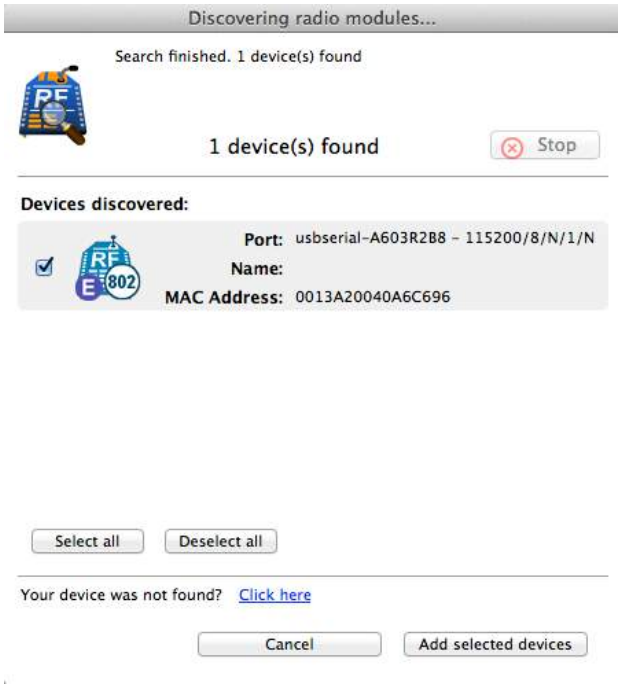


Imagen 22

Imagen 23

Como podemos ver en la pantalla. Los primeros parámetros que nos muestra son: CH (canal), ID (identificador de la red), DH (dirección de destino Alta), DL (dirección de destino baja), MY (Mi dirección). Si nos movemos por la barra de navegación hacia abajo podremos ver todos los parámetros susceptibles de ser modificados y configurados. Por ahora nos quedamos con los que hemos tratado. Nos faltaría por modificar el parámetro DB (velocidad en baudios) que se encuentra un poco más abajo.

Una vez realizados los cambios en los parámetros citados con los valores que hayamos programado, sirva de ejemplo la tabla que presentamos a continuación, debemos de grabarlos en el *firmware* de la placa. Para ello

tenemos dos posibilidades. Junto a la línea de configuración de cada parámetro tenemos dos iconos.

Imagen24

Con el primero desde la derecha (lápiz marrón), grabamos en los valores introducidos en ese parámetro. Con el segundo (flechas verdes) actualizamos el estado de dicho parámetro.

Esto es útil cuando solamente hemos cambiado uno o dos parámetros puntualmente.

Cuando hemos hecho una configuración que implica un número importante de cambio de parámetros, podemos

Imagen 25

grabarlos todos a la vez, pulsando el icono lápiz situado en la parte superior del apartado “Radio configuración”.

Esta acción graba los valores que hemos modificado en el *firmware* del módulo XBEE y comenzará a emitir en base a los mismos. El resto de parámetros que no hemos modificado, mantendrán su valor por defecto.

Haremos el proceso de modificación y grabado de los dos módulos XBEE atendiendo a los siguientes valores.

Modulo XBEE 1		Modulo XBEE 2	
Parámetro	Valor	Parámetro	Valor
ID	1234	ID	1234
MY	0	MY	1
DH	0	DH	0
DL	1	DL	0
BD	7 [115200]	BD	7 [115200]

Imagen 26

Con estos valores se establecerá la comunicación entre ambos dispositivos a una velocidad de 115200 bps (baudios por segundo), dentro de la red 1234, con dos dispositivos cuya identificación será 0 (para el módulo 1, receptor) y 1 (para el módulo 2, emisor). El módulo 1 tendrá una *DL* de 1 (dirección de comunicación inalámbrica baja), pues éste apunta al módulo 2. El módulo 2 por tanto tendrá un valor *DL* de 0, pues apunta al módulo 1.

5.3 LA COMUNICACIÓN SERIAL

Los valores de los diferentes sensores y pulsadores son recogidos y procesados por el microcontrolador Atmega328p. Estos valores en bruto no nos serían útiles por sí solos, necesitamos enviarlos a algún dispositivo externo capaz de comprenderlos y llevar a cabo las transformaciones que estimemos oportunas a través de la programación. En nuestro caso, el objetivo es enviar dichos datos a nuestro PC y es aquí donde haremos uso de la comunicación serial.

La comunicación en serie o comunicación serial desde los orígenes de la computación es un estándar que se desarrolla para la comunicación de éstas con el exterior. Periféricos de entrada y salida de datos (ratón, teclado, modem, monitores etc.) utilizan este tipo de comunicación.

El concepto de comunicación en serie es sencillo y consiste básicamente en el envío y recepción de datos en cadena, es decir, uno detrás de otro . En los lenguajes de computación se trataría del envío y recepción de bits. Difiere en este sentido con otro de los protocolos más comunes en los sistemas de comunicación de dispositivos, la comunicación en paralelo, que sí que permite el envío simultáneo de bits[13], llegando a enviar hasta un byte[14] o más de manera simultánea.

[13] Bit. es el acrónimo Binary digit. Es la unidad básica que puede representar un dispositivo digital a través del en sistema binario. Un bit representaría un dígito del sistema binario cuyo valor únicamente podría ser 0 o 1. La combinación de estos valores es la base del lenguaje binario con el que se comunica cualquier dispositivo digital como las computadoras.

[14] Byte. Unidad de medida del sistema binario que equivale a 8 bits. 16 bits es igual a 2 byte. 3 bytes sería igual a 24 bits etc.

A pesar de que la comunicación en paralelo a priori parece más rápida y eficiente, presenta algunos inconvenientes que tienen que ver con la limitación de la distancia y volumen y coste del tipo de cableado y conexiones.

La comunicación en serie sigue siendo muy útil en la actualidad por cuestiones de infraestructura y sobre todo económicas. Mientras que una comunicación serial básica se podría llevar a cabo con un cable de cuatro hilos, en la comunicación en paralelo los cables deberán de tener tantos hilos como bits simultáneos pretendamos enviar. Por otra parte uno de los inconvenientes de la comunicación serial con respecto a la comunicación en paralelo que tenía que ver con la velocidad de transmisión, ha sido subsanada notablemente con los nuevos sistemas comunicación serial actuales, USB, FireWire o serial ATA.

Tanto la *Arduino Fio* como el resto de placas de la gama utilizan la comunicación serial para el envío y recepción de datos.

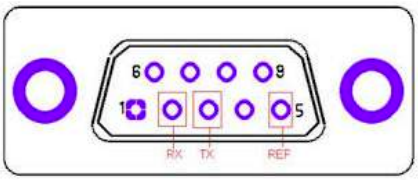


Imagen 27


Imagen 28

Los valores que se envía y reciben a través del puerto serie corresponden al estándar de comunicación creado en el año 1963 ASCII (American Standard Code for Information Interchange). Es importante tener esto en cuenta pues, a la hora de manejar los datos que nos llega de nuestra placa, debemos de saber que no serán los valores absolutos útiles para manejar en nuestra programación. Deberemos por tanto hacer la conversión de ASCII a Símbolo (números, letras, símbolos etc. comprensibles para nosotros), teniendo en cuenta la tabla de equivalencia de los valores ASCII a símbolos estandarizado por dicho lenguaje tal y como refleja la siguiente tabla.

Imagen 29

Si al recibir los datos de nuestra placa desde cualquier programa capaz de mostrarnos lo que llega por el puerto serie, vemos impreso el valor 49, debemos de tener en cuenta que este valor corresponde al carácter numérico *1*. En cambio si vemos impreso el número 97, correponderá a la letra “a” minúscula (pues “A” mayúscula corresponde al valor ASCII 65).

Es por esto que al filtrar y usar los valores procedentes de la placa en nuestro patch de Max/MSP o códigos de Supercollider, hemos tenido que hacer la conversión de ASCII a caracteres, numéricos o letras comprensibles para nuestra programación.

En las figuras correspondientes a las imágenes 27 y 28 podemos ver un conector serial hembra y macho de 9 pines. Nos vamos a centrar en tres de ellos que debemos de entender para la comunicación de la placa con el pc o con otros dispositivos que entiendan la comunicación en serie.

Como vemos en la figura 27, los pines 2, 3 y 5 están marcados como RX, TX Y REF, respectivamente.

- RX es la vía por la que se reciben los datos de otro dispositivo.
- TX es la vía por la que se transmiten los datos a otro dispositivo
- REF O GND. Toma de tierra o nivel de tierra del sistema.

Estos serán los pines que debemos de identificar y tener en cuenta para la transmisión de datos entre dispositivos.

En la placa Arduino Fio, tenemos dos puertos de transmisión y recepción TX Y RX. A través del puerto FTDI y por otra lado, el módulo de comunicación inalámbrica XBEE, también posee entre tantos otros, pines TX, RX y GND para la comunicación con el otro módulo XBEE conectado al pc. No implica que podamos utilizar ambos de manera simultánea. Si conectamos nuestra placa al

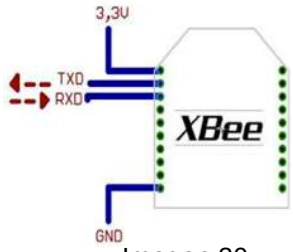


Imagen 30

pc para programarla por medio del adaptador FTDI y tenemos insertado el módulo XBEE inalámbrico en el zócalo habilitado para ello en nuestra *Arduino Fio*, el proceso de escritura dará error. Como indicábamos en el pié de página número 6, cuando queramos llevar a cabo la programación de nuestra placa, el módulo XBEE debe de estar quitado.

El esquema básico de conexión de una comunicación serial sería.

Dispositivo1	***Dispositivo2***
TXD (transmisión de datos)	TXD (transmisión de datos)
RXD (recepción de datos)	RXD (recepción de datos)
GND (nivel de tierra)	GND (nivel de tierra)

6 SENSORES Y PULSADORES. CARACTERÍSTICAS Y MONTAJE

En este apartado hablaremos de los elementos físicos que se encargan de generar los datos: sensores y pulsadores.

Vamos a manejar dos tipos de valores según los elemento que utilicemos:

- Valores continuos: sensores.
- Valores binarios. 0 y 1. High/Low.

6.1 SENSORES GENERADORES DE VALORES CONTINUOS

Los sensores funcionan ejerciendo una resistencia que hace variar el voltaje de entrada al puerto analógico al que lo hemos conectado. Éste produce una variación de tensión que va de los 3.3v - 5v a los 0v (según la placa). Un microprocesador solo entiende valores numéricos binarios (ceros y unos) por lo que todas las placas llevan incorporado un conversor que traduce los valores eléctricos en digitales. Realizada la conversión, los valores que vamos a manejar procedentes de los diferentes sensores estará comprendido en un rango que va de 0 a 1023. El conversor analógico/digital tiene una resolución de 10 bits (2^{10} = 1024. 1024 posibilidades de combinar ceros y unos). Así pues, el conversor traducirá escalando las posibilidades de voltaje entre 3.3v a 0v, a valores que irán de 1023 a 0:

- Ausencia de resistencia = voltaje más alto (3.3v) = mayor valor (1023)
- Resistencia total a la corriente = voltaje más bajo (0v) menor valor (0).
-

Según el voltaje que entra en cada pin analógico, determinado por la resistencia a la corriente que está ejercicio en cada momento nuestro

sensor, al realizar la conversión a digital, nos dará un valor u otro (entre 0 y 1023).

Con el fin de eliminar valores inestables de voltaje en el circuito, algunos de nuestros sensores llevan incorporado en el montaje una resistencia pull-down de 10KΩ. Así pues, además de la resistencia ejercida por el sensor, existe una resistencia fija que reduce de manera constante el valor de entrada de voltaje al pin correspondiente. Los valores que manejamos en alguno de nuestros sensores serán inferiores a 1023 (entre 900 y 1000).

6.1.1 Sensor de fuerza FSR circular - FSR

Imagen 31

Utilizamos tres de ellos situados en las llaves, 3, 4 y 5 del saxofón. Se podrían ubicar (con cinta adhesiva de doble cara) indistintamente en cualquiera de las llaves del registro natural. 1, 2, 3, 4, 5 y 6. Se quita con facilidad y no deja restos de ningún tipo.

Imagen 33

Imagen 34

Sus valores van de 0 a 950-970 según la fuerza que ejerzamos. La fuerza ejercida por cada instrumentista en su ejecución normal puede

variar. En nuestro caso hemos comprobado que la resistencia ejercida en una ejecución normal, da valores entorno a 650-750. Si queremos hacer uso de los valores, solamente cuando ejerzamos una fuerza especial mayor a la ejercida en la ejecución normal, debemos de marcar nuestro rango útil a partir del valor 800.

6.1.2 Potenciómetro de Membrana SOftPot 50mm

Imagen 35

Ejerce resistencia y varia la corriente del circuito del que forma parte desplazando nuestro dedo índice de la mano derecha sobre la banda central del sensor. Hemos añadido una resistencia física de 10KΩ para evitar ruido. La capacidad de resistencia va desde los 100 ohms a los 10000 ohms. Generará valores que van de 0 a 1000 aproximadamente.

Imagen 36

Está ubicado en el instrumento en el espacio que hay entre la protección trasera de las llaves 4, 5 y 6 y la pieza que hace de soporte para el dedo pulgar de la mano derecha. El diseño está pensado para ser controlado con este dedo.

Éste está adherido a una pieza de chapa de 0.5mm de color dorado que aprovecha los tornillos que sujetan la pieza que protege la parte trasera de las llaves 4, 5 y 6 tal y como se muestra en la imagen.

6.1.3 Joystick analógico

Imagen37

Este modelo está dotado de dos potenciómetros cuyos valores (0 a 1023) variarán según lo movamos en el eje X o Y. Genera valores muy precisos sin apenas ruido. La posición central del joystick sitúa cada potenciómetro a la mitad de su rango de valores, por tanto los valores de cada uno de ellos cuando está en reposo será de 509-510. Según desplacemos el mando hacia un extremo u otro de cualquiera de los ejes, iremos hacia el valor 0 o hacia el valor 1023.

Imagen 38

Situado un poco más abajo y a la derecha de la pieza donde apoyamos el dedo pulgar derecho y montado sobre una superficie de chapa de 0.5mm anclado a las sujeciones de la pieza que sirve de soporte para el dedo pulgar y la llave D#, está pensado para ser accionado también con dicho dedo.

Lleva incorporado además un pulsador que se acciona cuando pulsamos el mando de pasta negro hacia adentro.

6.1.4 Acelerómetro de 3 ejes – Montado en placa ± 3g

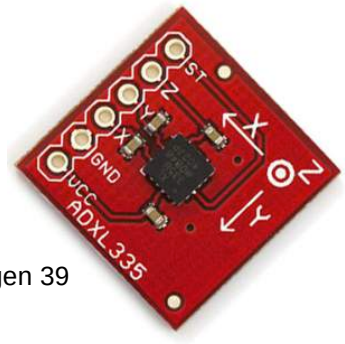


Imagen 39

Un acelerómetro es un es un tipo de sensor analógico que detecta el movimiento o el giro, es decir, es capaz de responder con una señal eléctrica ante una perturbación inducida por la aplicación de una fuerza o la gravedad. Este dispositivo es capaz de detectar si está en posición horizontal o vertical o si lo agitamos en el aire.

Este modelo da la posibilidad de transmitir a nuestra placa tres voltajes correspondientes a cada uno de los ejes X, Y y Z según el giro, movimiento o inclinación correspondiente a cada eje. Mide la aceleración con una escala de ± 3G y utiliza un nivel de tensión de 3.3 V. Debemos de conectarlo a la alimentación 3.3V, de lo contrario se dañaría el componente

Debido a la limitación de pines analógicos de la placa (8 analógicos) hemos prescindido del eje Z en favor de un FSR más.

El valor dado por cada uno de los ejes oscila entre los 400 y los 600 aproximadamente en función de la inclinación o fuerza dada al movimiento.

Hemos colocado el acelerómetro en la pieza metálica sobre la que montamos el joystick.

6.2 Pulsadores

El funcionamiento y lógica de los pulsadores es sencillo.

Serán los encargados de activar o desactivar procesos o tareas. Compuesto por tres microrruptores (finales de carrera) de tres contactos y el pulsador del modulo joystick, en total serán cuatro las posibilidades que tengamos para utilizar en nuestra programación.

Están conectados a los pines digitales de nuestra placa, y nos ofrecen dos valores posibles, 0 o 1. A diferencia de los pines analógicos, los digitales no tienen la capacidad de dar valores en función del voltaje de entrada. Solo distinguen dos estados de corriente. Valor 0 cuando el nivel de entrada de corriente es nula o valor 1 cuando hay corriente, independientemente del nivel de corriente que llegue.

Si conectamos nuestro sistema y leemos los datos que nos llegan de los diferentes pulsadores sin accionar ninguno de ellos, veremos que tres de ellos nos dan valor 0 y uno de ellos envía el valor 1. Los pines a los que hemos conectado los microrruptores (D2, D3, D4) nos darán valor 0 sin haber accionado éstos y 1 cuando los accionemos, sin embargo, el pulsador del joystick (conectado al pin D5) nos da un valor constante de 1 cuando está en reposo y valor 0 cuando lo pulsamos. Es importante tener esto en cuenta a la hora de llevar a cabo nuestra programación y no pensar que se trata de un error.

Para entender la razón de este hecho debemos de conocer los conceptos pull-up y pull-down. Pull-up y pull-down hacen referencia a una resistencia colocada a la entrada del circuito, la cual, en función de su ubicación hará que el voltaje entre en el circuito o por el contrario vaya a tierra.

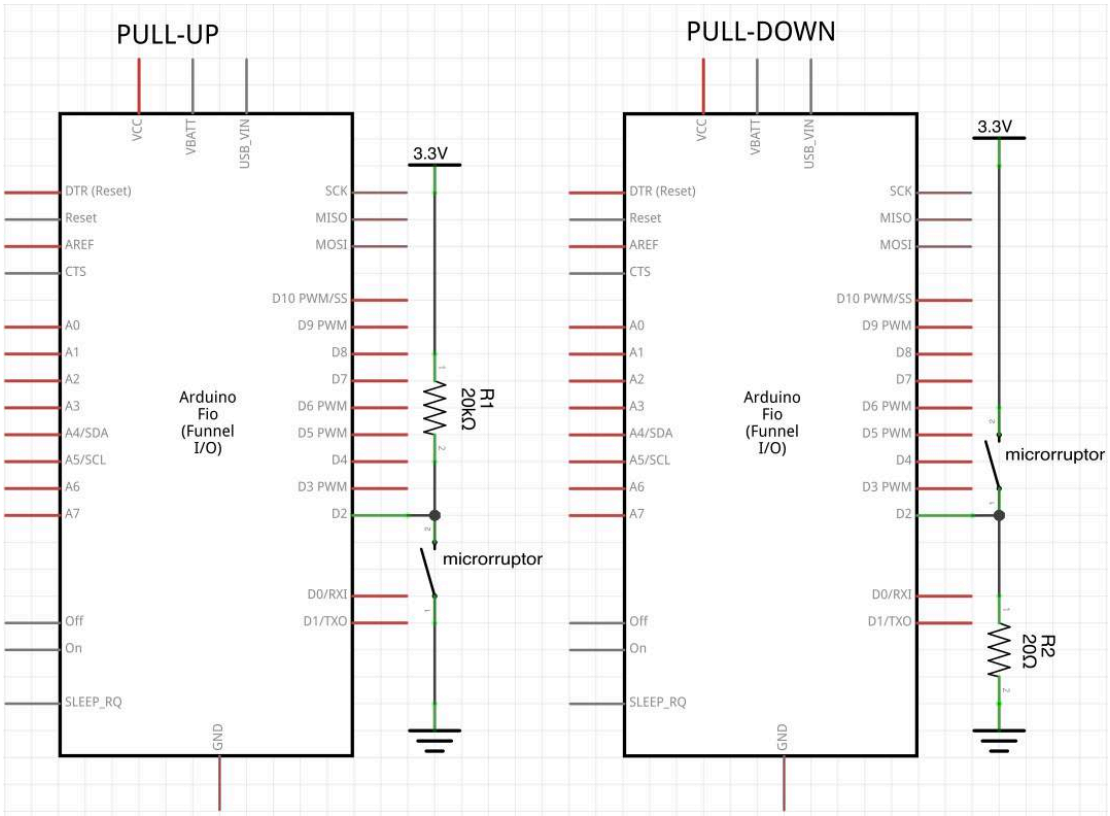


Imagen 40

La resistencia aplicada a los pulsadores no está en el circuito integrado, hemos aprovechado una posibilidad que ofrece el chip Atmega el cual lleva incorporada una resistencia de 20KΩ que se puede aplicar a cualquier pin a través de la programación. Se trata de una resistencia pull-up. (esquema de la izquierda imagen 40).

El comportamiento de los microrruptores en estado de reposo (abierto o cerrado) lo podemos modificar según hagamos la conexión 1-3 o 1-2. Como el comportamiento del pulsador del joystick no lo podemos modificar, la resistencia tipo pull-up hace que en estado de reposo su valor sea 1(en reposo el paso a tierra está abierto).

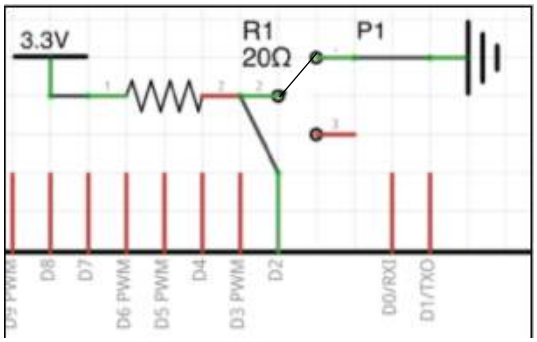


Imagen 41

Pulsador microrruptor en reposo

Pulsador joystick pulsado

Paso a tierra cerrado

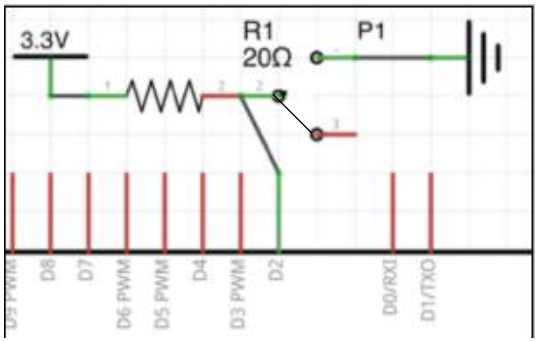


Imagen 42

Pulsador microrruptor accionado

Pulsador joystick en reposo

Paso a tierra abierto

Los tres pulsadores final de carrera están montados en el saxofón por medio de una estructura metálica adherida a las protecciones de los platos de las llaves *B* y *Bb* tal y como muestran las imágenes.

Imagen 43

Imagen 44

La estructura está formada por dos piezas de aluminio cuadradas de 5x5mm, curvadas a las que se le han realizado orificios utilizando brocas de 3mm y 2.5mm, tanto para el anclaje de la estructura a las protecciones de las llaves, como para instalar entre ambas piezas dos varillas redondas calibradas de 2.5mm. Sobre estas varillas se colocan los microrruptores (aprovechando los orificios que éstos traen por defecto).

Imagen 45

Para la sujeción de las varillas redondas a las piezas de aluminio se han utilizado piezas extraídas de fichas de empalme de cable de 2.5mm partidas en dos.

Imagen 46

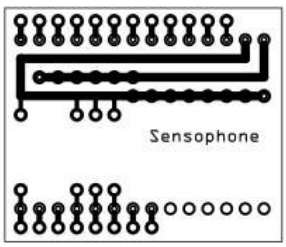
El montaje de estos pulsadores está diseñado para ser accionados con comodidad por los dedos índice, medio y anular de la mano derecha.

La posibilidad de desplazamiento de la estructura hacia delante y hacia atrás, junto con la del desplazamiento lateral de los pulsadores a los que se le ha insertado un recubrimiento (macarrón trasparente) en la palanca que acciona el pulsador, está destinado a ofrecer la mayor ergonomía posible al ejecutante.

Imagen 47

6.3 El circuito impreso

El circuito que conecta los sensores y pulsadores a nuestra placa ha sido creada a partir de una placa de prototipado virgen con unas dimensiones de 3.3 X 3.7 cm sobre la que se han impreso las pistas de cobre y puntos de soldadura necesarios para nuestro proyecto, siguiendo el procedimiento que describimos a continuación:



- En primer lugar diseñamos el circuito con el programa Fritzing.

Imagen 48. Esquema circuito impreso para impresión[15]

[15] Las dimensiones de este esquema puede que no se correspondan con precisión a las reales. Para la confección del circuito sería conveniente utilizar el esquema anexo a este trabajo.

- Imprimimos el diseño en papel couché con una impresora láser.
- Por otro lado, preparamos la placa, limpiando y puliendo muy bien la superficie de cobre con lana de acero.
- El siguiente paso consiste en transferir el diseño de la placa impresa en papel couché en la superficie de cobre de la placa de prototipado. Para ello, ponemos la cara de papel impresa sobre la superficie de cobre de la placa y con una plancha convencional, aplicamos calor al papel intentando no moverlo durante 10-15 minutos. Pasado este tiempo la tinta habrá quedado impresa en la superficie de cobre. Quitamos el papel cuidadosamente después de haber metido la placa en un recipiente con agua durante 10-15 minutos. Retiramos el papel, pero la tinta queda en la placa.
- Introducimos la placa en una solución a partes iguales de agua fuerte, agua oxigenada de 110 volúmenes y agua.
- La solución anterior disuelve el cobre de la placa de prototipado excepto el cobre que se encuentra justo debajo de la tinta impresa. Cuando la superficie de cobre ha desaparecido, sacamos la placa sin tocar la solución de ácido, lavamos con agua la placa y por último eliminamos la tinta que habíamos impreso previamente con calor. Esto se puede hacer con cetona, aguarrás u otra solución similar.
- Eliminada la tinta observaremos que han quedado sobre la placa las pistas de cobre conforme al circuito que habíamos diseñado.

Ya tendríamos la placa lista para taladrar y soldar los elementos necesarios.

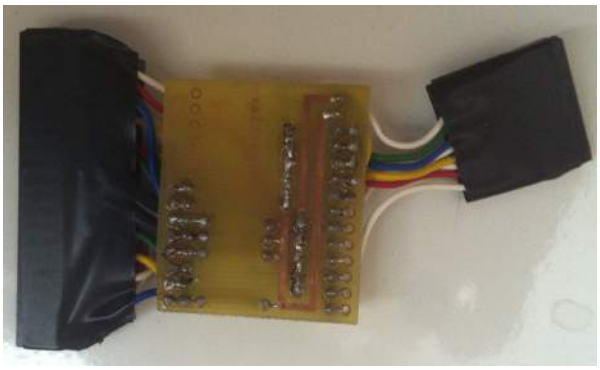


Imagen 48b

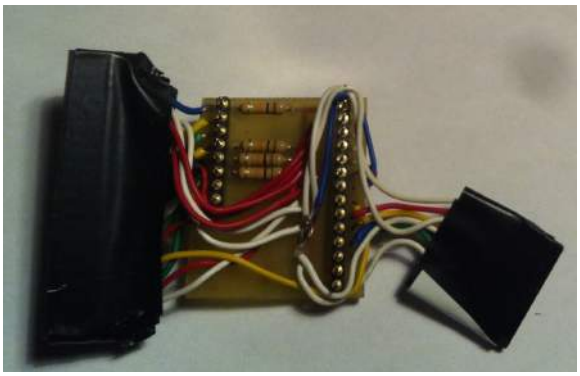
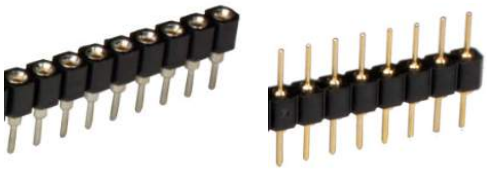


Imagen 48c

Para conectar la placa Arduino con la placa de prototipado hemos utilizado regletas de conexión de tipo “regleta de pin torneado de 2,54mm hembra”[16] para la Arduino y “regleta de pin torneado de 2,54mm macho” para nuestro circuito impreso.

Imagen 49



6.4 Cables y conexiones

Para la comunicación entre los sensores y pulsadores con la placa, hemos fabricado diferentes cables utilizando cable plano y conectores “regleta de pin de 2.54mm” macho y hembra

Imagen 59.Cable plano

Imagen 60. Conectores regleta de pin 2.54mm hembra y macho

A través de dos regletas de pines de 2.54mm hembra se han distribuido los puntos de conexión de los sensores y pulsadores de forma independiente de la siguiente manera:

Pulsadores (pines digitales)

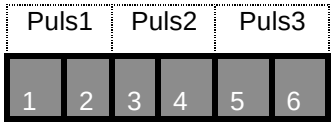


Imagen 61

[16] [en línea]. Disponible en. http://www.electronicaembajadores.com/Productos/Detalle/11/CTO1HT40/regleta-pin-torneado-2-54mm-hembra-recta-40-contactos [consultada 24/11/14]

1. GND. Toma a tierra pulsador 1
2. Conexión pulsador 1 pin D2
3. GND. Toma a tierra pulsador 2
4. Conexión pulsador 2 pin D3
5. GND. Toma a tierra pulsador 3
6. Conexión pulsador 3 pin D4

Sensores (pines analógicos)

Imagen 62

1. GND. Toma a tierra joystick
2. Vcc. Alimentación joystick
3. VRX. Eje X joystick pin A1
4. VRY. Eje Y joystick pin A2
5. SW. Pulsador joystic pin D5
6. Vcc. Alimentación resistivo 1
7. Conexión resistivo 1 pin A3
8. Vcc. Alimentación resistivo 2
9. Conexión resistivo 2 pin A4
10. Vcc. Alimentación resistivo 3
11. Conexión resistivo 3 pin A5
12. ST. Self Test acelerómetro. No conectado
13. Eje Z acelerómetro. No conectado
14. EJeY acelerómetro, pin A6
15. EjeX acelerómetro, pin A7
16. GND. Toma a tierra acelerómetro
17. VCC. Alimentación acelerómetro
18. GND. Toma a tierra potenciómetro
19. Conexión potenciómetro pin A8
20. Vcc. Alimentación potenciómetro

Hemos construido diferentes cables agrupados siguiendo el siguiente esquema.

1. Cable plano de 5 hilos para potenciómetro, con conexión hembra en un extremo y macho en el otro extremo. (20cm)

2. Tres cables planos de 2 hilos una para cada sensor resistivo (FSR) con conexión hembra (utilizando regleta de pin torneado de 2.54mm) en un extremo y macho (regleta de pin normal de 2.54mm) para la conexión con la placa. (2 cables de 11cm y 1 de 23cm)


Imagen 63

3. Cable plano de 6 hilos para acelerómetro conexión hembra en un extremo y conexión macho en el otro extremo. (24cm)

Imagen 64

4. Cable plano de 6 hilos para pulsadores final de carrera con conexión hembra en un extremo y conexión macho en el otro extremo. (22cm)

Imagen 65

5. Cable plano de 3 hilos para potenciómetro con conexión hembra (regleta de pin torneado de 2.54mm) en un extremo y conexión macho en el otro extremo. (25cm)

Imagen 66

6.5 La Caja

El conjunto formado por la placa *Arduino Fio*, el circuito integrado, la batería, los pines de conexión, el led y el interruptor on/ff externo, está instalado en la sujeción que mantiene la estabilidad entre la campana y el cuerpo del saxofón a través de una caja fabricada en chapa dorada de 0.5mm que responde al diseño de las siguientes imágenes.

Imagen 67

Imagen 68

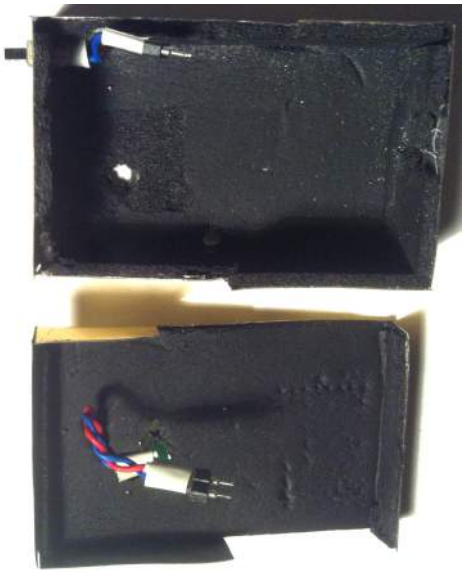


Imagen 69

7 PROCESOS CON MAX/MSP Y SUPERCOLLIDER

Como parte de este trabajo, se adjuntan los patch necesarios para la recepción y procesado de datos, tanto para Max/MSP como para Supercollider.

- Desde el patch de Max/MSP podremos realizar los envíos inalámbricos a través los objetos *send y receibe* de cada uno de los sensores y pulsadores y aplicarlos a aquellos procesos que hayamos programado. Están claramente identificados en el patch.
- El código de Supercollider envía el valor de cada uno de los sensores y pulsadores a los Bus de control que asignaremos evaluando las líneas:
    - a = {Bus.control(s, 1)} ! 6;
    - d = {Bus.control(s, 1)} ! 5

Evaluar las líneas según están ordenador en el patch.

Para usar los valores insertaremos en nuestro código a través de In.kr con los argumentos según la siguiente tabla

SENSORES	
In.kr(a[0])	Eje X joystick
In.kr(a[1])	Eje Y joystick
In.kr(a[2])	FSR 1
In.kr(a[3])	FSR 2
In.kr(a[4])	FSR3
In.kr(a[5])	Acelerómetro eje X
In.kr(a[6])	Acelerómetro eje Y
In.kr(a[7])	Potenciómetro
PULSADORES	

In.kr(d[0])	Pulsador 1
In.kr(d[1])	Pulsador 2
In.kr(d[2])	Pulsador 3
In.kr(d[3])	Pulsador joystick

Imagen 70

8 CONCLUSIONES

El desarrollo del proyecto *Sensophone* ha resultado una experiencia enriquecedora, apasionante y sobre todo reveladora. En el camino recorrido para adquirir los conocimiento necesarios para llevar a cabo el trabajo, nos hemos encontrado con otros muchos proyectos en los que si bien los materiales, procesos y medios utilizados son muy parecidos, hemos visto cómo los resultados son de lo más diverso, habiendo resultado especialmente llamativos aquellos relacionados con el mundo del arte. Esto nos ha dado una perspectiva muy amplia sobre las posibilidades creativas que el mundo de los sensores y el manejo de sus datos pueden ofrecer al mundo del sonido. Sin ser algo excepcional, pues se lleva muchos años experimentando en esta línea, lo que si parece haber surgido en los últimos tiempos es una democratización y apertura de los recursos, a los que tiempo atrás solo empresas y marcas comerciales podían acceder debido al elevado costo de este tipo de proyectos. Costo que se traducía en un elevado coste del producto final.

El desarrollo del proyecto Arduino supone un cambio en la manera de entender y llevar a cabo ciertos proyecto en el que al igual que ocurriese con la aparición de la filosofía de software libre instaurada por la creación de Linux, aquello que parecía reservado a una decena o centena de personas con información privilegiada y recursos exclusivos que trabajan al amparo de una determinada empresa, ahora queda abierto al resto del mundo. Cualquier persona cuya inquietud, talento y esfuerzo le permita llevarlo a cabo podrá desarrollar trabajos a un alto nivel con un coste asumible.

El resultado final del trabajo responde a los objetivos marcados inicialmente. Hemos conseguido un sistema de transmisión de datos en tiempo real instalado en el saxofón que podría ser equivalente al *Metasax* de Burtne o el Synthophone de Hurni pero a diferencia de éstos, el funcionamiento acústico

del saxofón no ha sufrido la más mínima transformación, manteniendo por tanto sus cualidades sonoras. Tampoco ha habido cambios en la digitación. La posibilidad de emitir a través de un sistema inalámbrico y a una velocidad mayor de proceso a la utilizada por el sistema MIDI (11520 baudios frente a los 31250 del lenguaje MIDI) mejora las prestaciones de los anteriormente citados.

La continua evolución del mercado de los microcontroladores y sensores hace del nuestro un proyecto que no está cerrado. Las actuaciones llevadas a cabo para la mejora y evolución del *Sensophone* podrían ir en la siguiente línea:

- Mejorar la estructura sobre la que están instalados los pulsadores final de carrera pudiendo ser sustituidos por pulsadores táctiles. Esto junto a un cambio en el diseño del cableado estaría encaminado a agilizar al máximo la labor de montaje y desmontaje del sistema del saxofón.
- Aumentar el número de sensores FSR de 3 a 6, cubriendo así todas las notas del registro natural excepto el Do grave. Para esto, debería de salir al mercado una placa que mantuviese o mejorase las características de la placa Arduino Fio en cuanto a tamaño, posibilidad de incorporar un sistema inalámbrico, alimentación etc., y que además ofrezca la posibilidad de más pines analógicos(hasta 11).
- Adaptar el sistema al resto de saxofones de la familia(Tenor y Barítono). Su inclusión en el saxofón Soprano supondría un cambio significativo en el diseño original.
- La investigación e instalación de un sistema de amplificación eficiente a través de varios micrófonos adaptados al saxofón. Éste debiera de ser uno de los objetivos a alcanzar a corto o medio plazo. La posibilidad de disponer de un buen sistema de amplificación dotaría al trabajo actual del complemento perfecto para redondear el proyecto.

Independientemente de las mejoras que se puedan llevar a cabo, algo que queda pendiente finalizado el proyecto inicial es el conocimiento de éste por parte de los diferentes círculos musicales. Llamar la atención tanto del colectivo saxofonístico como del mundo de la composición resultaría indispensable para la continuidad del proyecto en tiempos venideros

9 BIBLIOGRAFÍA

9.1 Libros

- **Alfonseca Moreno, Manuel y Alejandro Sierra Urrecho**: *"Programación en C/C++, Edición revisada y ampliada 2005"* (Madrid: Anaya, 2005
- **Collins, Nicolas:***"Handmade Electronic Music, The Art of Hardware Hacking" (*New York: Routledge Taylor & Francis Group, 2006)
- **Colosanto, Francisco**: "*Max/MSP, Guia de programación para artísticas*"(Morelia, Michoacán: Centro Mexicano para la Música y las Artes Sonoras Casa de la Cultura, 2010), Vol. 1
- **H & A. Selmer, inc:** *"News About Varitone, Varitone Players and their Music"*(s. l. : Chuck Slinkard, 1968)
- **Nussey, John**: *"Arduino for dummies"* (Chichester: John Wiley & Sons, Ltd, 2013)
- **Purdum, Jack:***"Beginning C for Arduino, Learn C Programming for the Arduinoand Compatible Microcontrollers"* (New York: Apress, 2012)
- **Ruiz Gutiérrez, José Manuel**: *"Arduino + Pure Data, Conexión de la Plataforma Open Hardware con Pure Data* (s. l. : Creative Commons Attribution 3.0, 2013), Versión 1.0
- **Ruiz Gutiérrez, José Manuel**: *"Arduino + Xbee, Implementación de sistemas de Datos y Sensores en Redes Inalámbricas con Xbee integrado en la plataforma Open Hardware Arduino"* (s. l. : Creative Commons Attribution 3.0, 2012), Versión 1
- **Torrente Altero, Oscar: "***Arduino, Curso práctico de formación"* (Madrid: RC Libros, 2013)

9.2 Recursos Web

- www.woodwindcourse.co.uk/user/image/dave_gelly_grafton.doc
- http://arduino.cc/en/Main/Products (consulta:
- http://bildr.org/
- http://matthewburtner.com/metasax/
- http://nerdclub-uk.blogspot.com.es/2012/06/midi-sax.html
- http://robosax.com/kleinbio.html
- http://savageelectronics.blogspot.com.es/2011/09/configurando-radios-xbee.html
- http://stores.ebay.es/Chip-Partner-Store
- http://web.media.mit.edu/~tristan/
- http://www.academia.edu/1351972/What_Happened_to_the_Electronic_Saxophone
- http://www.digi.com/support/productdetail?pid=3352
- http://www.electronicaembajadores.com/?gclid=CKSvr6LXpMICFWfKtAod2x8AmQ
- http://www.ftdichip.com/FTDrivers.htm
- http://www.madtracker.org/plugins.php?author=custoMade
- http://www.synthophone.info/
- https://www.youtube.com/watch?v=Klj3Zqn3NzE “como fabricar placas de circuito impreso en casa. (proceso completo)
- https://www.youtube.com/watch?v=oQE56FVNqgs “selmer saxophone mark VI playing with Varitone amplifier