Universidad Nacional de Quilmes

Trabajo Final

Batería Electrónica
ARDRUMONE

Materia: Participación y Gestión en Proyectos de Software Libre

Profesor: Di Biasse, José Luis

Alumnos:

- Marchionne, Cristian
- Palazzo, Leonardo

Índice de contenido

Proyecto Ardrumone....3
Objetivo:....3
Introduccion:....4
MIDI....4
Desarrollo....5
SOFTWARE UTILIZADO....5
Hydrogen....5
Configuracion MIDI....5
Jack....6
Qjackctl....6
TTYMIDI....7
INSTALACIÓN....7
COMPILACION 7
DEPENDENCIAS....7
Libreria ArduinoMIDI 7
EJECUCIÓN....8
HARDWARE UTILIZADO....9
Arduino....9
Instalación....9
Características....9
Sensores....10
Programa....11
Ejemplo 1:....11
Ejemplo 2:....12
Ejemplo 3:....13
Refrencias....14
Hydrogen....14
Documentación....14
Tutorial....14
Videos de Hydrogen....14
Arduino....14
Instalacion de Plataforma....14
Lenguaje....14
Ejemplos....14
Arduino ONE 14
Esquema de Pines....14
Distribuidores....14
DealExtreme....14
LadyAda....14
Trabajos Futuros....15

Proyecto Ardrumone

Objetivo:

Objetivo Mezclar el mundo del Software y Hardware.

El proyecto consta de construir una batería electrónica a partir de información y material libre de copyrigth. Esto se documenta a modo de tutorial para que lo puedan abordar principiantes.

Introduccion:

MIDI

MIDI es un protocolo de comunicación serial estándar para dispositivos musicales electrónicos. Nacio en el año 1983.

Ventajas:

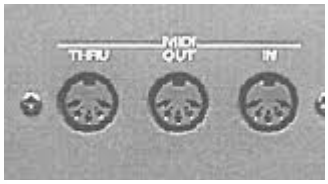
- Espacio de almacenamiento.
- Editar con facilidad
- Manejar a los sonidos independientemente

El protocolo establece distintos tipos de mensaje.

- Activar/Desactivar una Nota
- Selección de canción
- Reloj de temporización
- Inicio
- Continuación
- Parada
- Espera activa
- Reseteo del sistema
- Volumen

Ejemplo de un mensaje MIDI:

CODIGO DEL MENSAJE	CANAL	NOTA	VOLUMEN
Note on/off	0-15	0-127	0-127



Desarrollo

SOFTWARE UTILIZADO

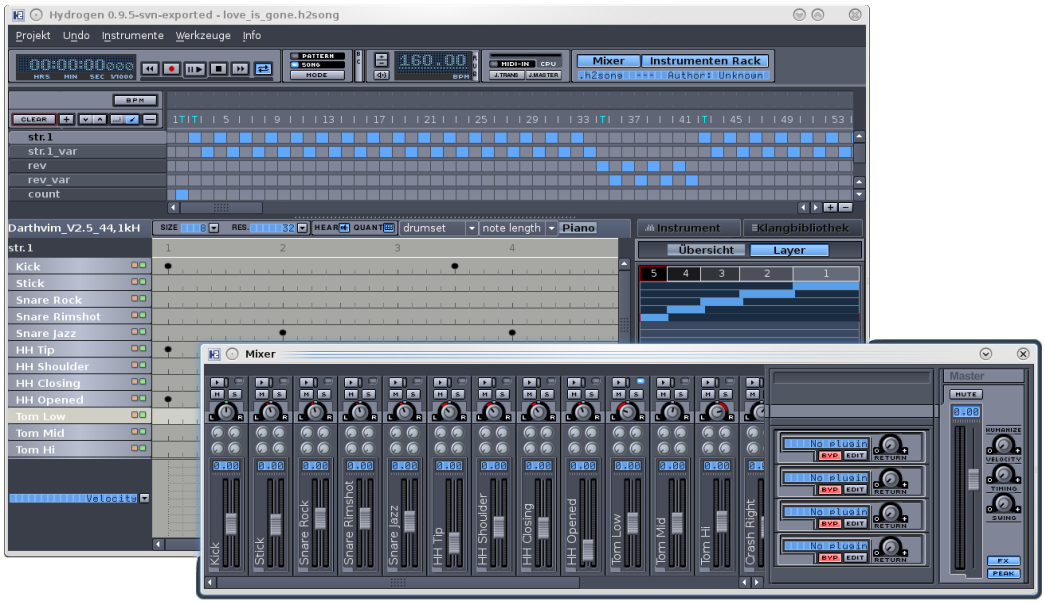
El objetivo es reproducir la nota recibida en este caso por el puerto usb. Para esto necesitaremos programas para emular al puerto como una entrada midi , que reproduzca la nota y que enlace ambos.

Hydrogen

Para la reproducción de sonidos de batería.

Lo único que debemos configurar es en Preferencias->Sistema de audio elegimos Jack. (asegurarse de ternerlo corriendo).

También tenemos la forma de configura el disparo de acciones propias (play, pause) mediante algún mensaje midi.



Configuracion MIDI

Se pueden cargar distintos tipos de librerías (drumkit) para simular los sonidos de la batería. Por defecto esta configurado el GmKit.

Las librerías relacionan una posición de un instrumento con una nota midi y una tecla del teclado de la pc. Hay que tener cuidado y entender bien que la posición no esta relacionado

con en nombre del instrumento en si.

Posición de Instrumentos	Nombre del Instrumento	Nota MIDI
1	Kick	36
2 / 3	Snare Jazz/Rock	37 / 38
4 / 5 / 6	Tom Low/Mid/High	39 / 40 / 41
17 / 18 / 19	HH Closed/Pedal/Pppen	52 / 53 / 54
20 / 21	Crash / Crash Jazz	55 / 56

Para descarga y más detalles: http://www.hydrogen-music.org/hcms/

Jack

Servidor de sonido que provee conexión de baja latencia entre aplicaciones. Es decir que la salida de un programa puede ser la entrada de otro. Por ejemplo si tenemos un reproductor de musica que por defecto sale por los parlantes, podemos configurarlo que sea la entrada de un programa que es por ejemplo una consola y luego a los parlantes.

Para descarga y más detalles : http://jackaudio.org/download

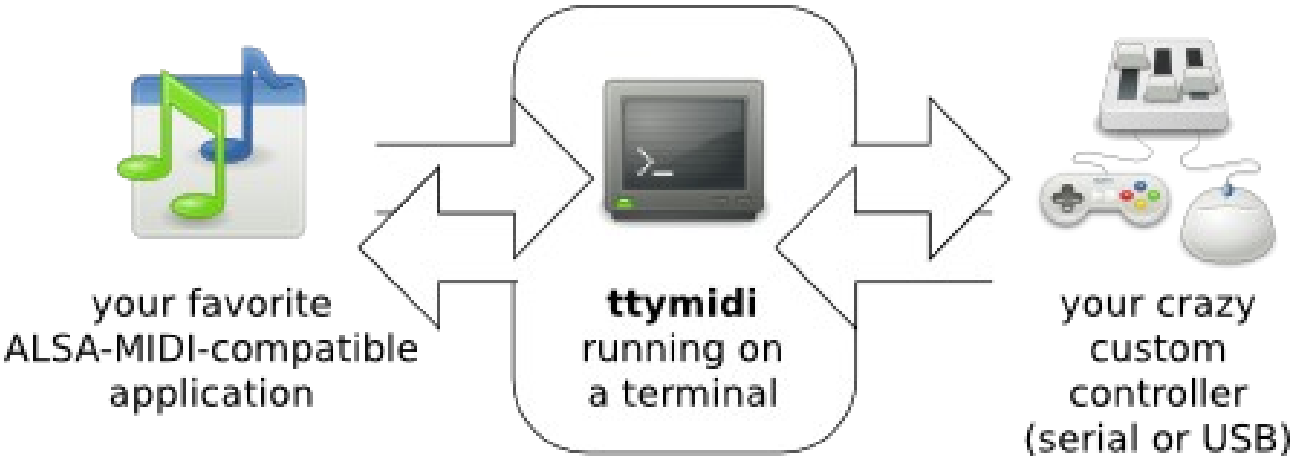
Como este programa no tiene interfaz gráfica podemos agregarla con

Qjackctl

Si bien tenemos muchas opciones es bastante intuitivo lograr nuestro objetivo que es conectar la salida del TTYMIDI a la entrada del Hydrogen y su salida a los parlantes.

Para descarga y más detalles : http://qjackctl.sourceforge.net/

TTYMIDI



ttyMIDI es un programa que nos permite interconectar un dispositivo serial externo con el secuenciador ALSA.

INSTALACIÓN

COMPILACION

El código del ttyMIDI esta compuesto por un único archivo escrito en C. Para compilarlo hay que ejecutar el siguiente comando: ***make***

DEPENDENCIAS

Este programa tiene una dependencia con: libasound2, si no esta instalada hay que hacerlo ejecutando: ***apt-get install libasound2-dev***

Después de haber compilado ttyMIDI hay que copiarla al directorio **/usr/bin** para acceder facilmente. Esto se puede hacer así: **sudo make install**

Libreria ArduinoMIDI

Para instalar en el Arduino la librería ttyMidi (conocida como ardumidi) solo hay que copiar la carpeta ttymidi a la carpeta de librerias del Arduino

cp -r ardumidi /path/to/arduino/sketchbook/libraries/

Esta librería trae algunos ejemplos básicos. Conviene mirarlos porque contienen información de como se maneja la librería.

EJECUCIÓN

Para correr este programa y escuchar un puerto usb como si fuera una entrada del tipo midi. Ejecutamos:

- ttymidi -s /dev/ttyS0 -b 2400

donde /dev/ttyS0 es el puerto serie que queremos leer y 2400 es la velocidad.

- ttymidi -s /dev/ttyUSB0 -vvv

donde la que la velocidad por defecto es 115200bps y la opción -v es para que muestre las entradas recibidas por consola.

El nombre del puerto lo podemos averiguar desde la IDE de arduino yendo a Tools → serialPort .

Para descarga y más detalles : http://www.varal.org/ttymidi/

HARDWARE UTILIZADO

Arduino

Arduino es una herramienta para hacer que las computadoras puedan sentir y controlar el mundo físico. Es una plataforma de electrónica abierta para la creación de prototipos basada en software y hardware flexibles y fáciles de usar. Posee un comunidad muy amplia que esta en constante crecimiento.

El microcontrolador en la placa Arduino se programa mediante el periférico USB utilizando el lenguaje de programación Arduino (basado en Wiring) con su propio entorno de desarrollo (basado en Processing).

Instalación

Para realizar la instalación de la plataforma ver: http://arduino.cc/es/Main/Software
Si usas linux y sos fanático de la consola:

```
$ apt-get install arduino arduino -core
$ apt-get install gcc-avr avr-libc
$ apt-get install sun-java6-jre
```

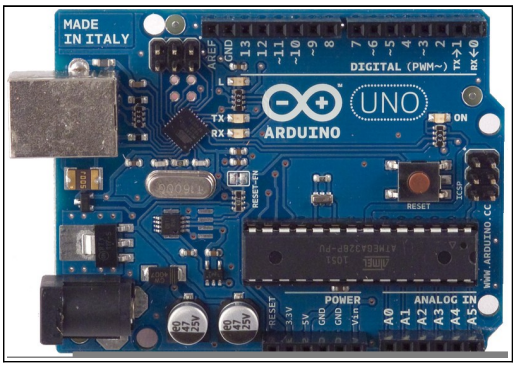
Características

Placa Utilizada: Arduino UNO

En términos generales esta placa posee:

- 14 entradas i/o salidas digitales (de las cuales 6 pueden ser salidas PWM).
- 6 entradas analógicas con una resolución de 10 bits.

Se alimentación con 5V cc, aprovechando el USB o con una fuente externa.

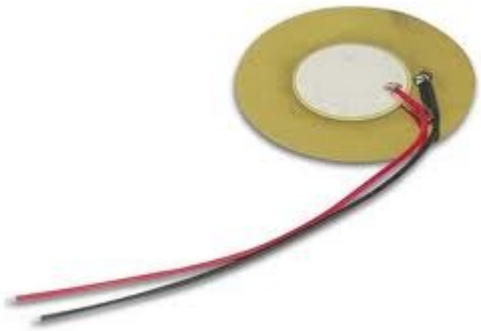


Para más detalles se puede ver: http://es.wikipedia.org/wiki/Arduino#Esquema_de_pines

Sensores

Como la placa que utilizamos tiene varias entradas analógicos, tenemos que decidir cuantos cuerpos queremos utilizar. Por cada cuerpo colocamos un sensor que se conectara a una entrada analógica.

Utilizamos un Buffer Piezoeléctrico como sensor para detectar la accion (el golpe).
Para mejorar la adquisición de datos se coloco un circuito adicional que nos permite obtener mayor sensibilidad al golpe.



Programa

A continuación vemos a ver tres ejemplos de programas que van a cargarse en el Arduino:

1. Básico — Para comprobar el funcionamiento del protocolo MIDI
2. Simple — Para relacionar el sensor con el nivel del volumen de una Nota.
3. Completo — Para armarte una batería con cuerpos analógicos y digitales.

Ejemplo 1:

Programa para verificar el funcionamiento del Protocolo MIDI y comprobar que la configuración de la PC.

```
/*
  Reprodusco una Nota mediante el protocolo MIDI por USB
*/

#include <ardumidi.h>

const int channel= 0;   //Canal que para identificar el instrumento
const int NOTE_KICK= 36;//Nota MIDI
const int volume= 127;  //Variable donde almaceno el valor del Volumen MIDI

void setup()
{
  Serial.begin(115200);
}

void loop()
{
    // Resprodusco una nota en un canal con un volumen determinado cada 500mseg
    midi_note_on(channel, NOTE_KICK, volume);
    delay(500);
}
```

Ejemplo 2:

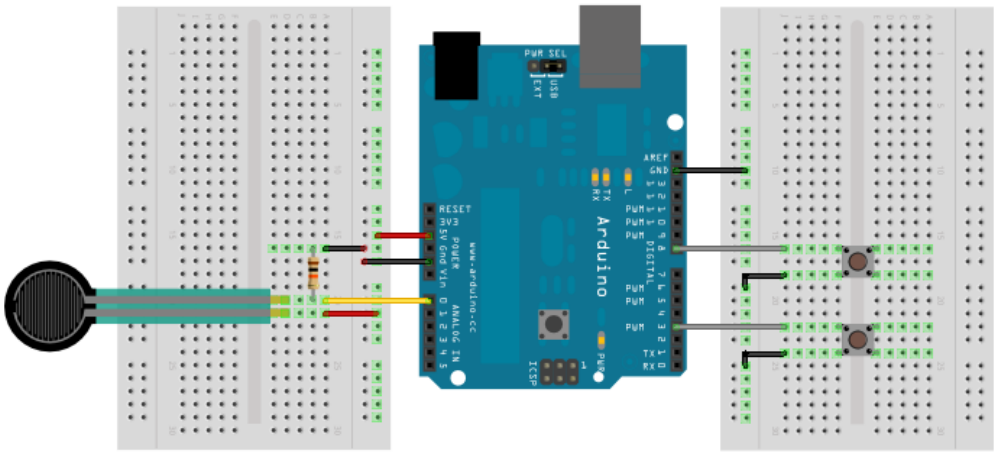
Programa para verificar la interaccionan del sensor con la variación del volumen MIDI de una nota.

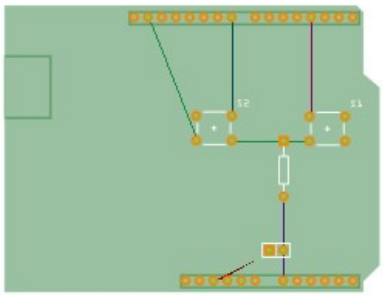
```
/*
  Leo el valor del canal Analogico cero (A0) y
  reprodusco el sonido MIDI con su correspondiente volumen
*/

#include <ardumidi.h>

const int ANALOG_TIME_OF_CONVERSION = 50;  // Tiempo necesario para la
conversion

const int channel = 0;         //Canal que voy a utilizar para identificar el
instrumento
const int NOTE_KICK = 36;      //Nota MIDI

const int sensorPin = 0;       //Canal Analogico en el que conecto el Sensor
const int threshold = 50;      //Nivel del Sensor. Los Valores inferiores se
consideran Ruido

int volume = 0;                //Variable donde almaceno el valor del Volumen
MIDI

void setup()
{
  Serial.begin(115200);
}

void loop()
{
    //Leo el valor analogico de algun Pin
    volume = readVolumeMidi(sensorPin);

    //Comparo el nivel de Volumen contra un nivel minimo
    if( volume > threshold)
    {
      // Resprodusco una nota en un canal con un volumen determinado
      midi_note_on(channel, NOTE_KICK, volume);
    }
    delay(ANALOG_TIME_OF_CONVERSION);
}

int readVolumeMidi(const int analogPin){
  //Leo el valor analogico de algun Pin y lo ajusto a los niveles de Volumen
MIDI [0-127]
  return map(analogRead(analogPin), 0, 1023, 0, 127);
}
```

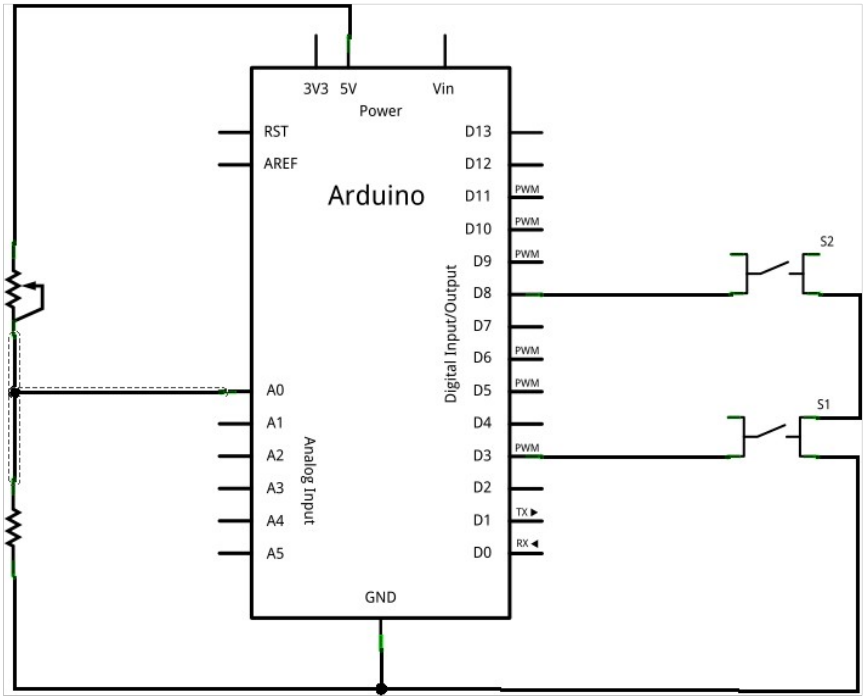
Ejemplo 3:

Programa para comprobar la adquisición de datos analógicos.

```
/*
  Leo el valor del canal Analogico cero (A0) y visualizo el valor obtenido
*/

#include <ardumidi.h>

const int ANALOG_TIME_OF_CONVERSION = 50;  // Tiempo necesario para la
conversion

const int sensorPin = 0;      //Canal Analogico en el que conecto el Sensor
const int threshold = 50;     //Nivel del Sensor. Los Valores inferiores se
consideran Ruido

int volume = 0;               //Variable donde almaceno el valor del Volumen
MIDI

void setup()
{
  Serial.begin(115200);
}

void loop()
{
    //Leo el valor analogico de algun Pin
    volume = analogRead(sensorPin);

    //Comparo el nivel de Volumen contra un nivel minimo
    if( volume > threshold )
    {
      Serial.print("\nSensorValue = ");
      // Resprodusco una nota en un canal con un volumen determinado
      Serial.print(volume);

      Serial.print("\tMidiValue = ");
      // Resprodusco una nota en un canal con un volumen determinado
      Serial.print(map(volume,0,1023,0,127));
    }
    delay(ANALOG_TIME_OF_CONVERSION);
}

int readVolumeMidi(const int analogPin){
  //Leo el valor analogico de algun Pin y lo ajusto a los niveles de Volumen
MIDI [0-127]
  return map(analogRead(analogPin), 0, 1023, 0, 127);
}
```


PWR SEL
Arduino
RESET
3V3
5V
GND
Vin
ANALOG IN
DIGITAL
PWM
TX
RX
ICSP
www.arduino.cc


S2
S1


3V3
5V
Vin
Power
RST
AREF
Arduino
D13
D12
D11
D10
D9
D8
D7
D6
D5
D4
D3
D2
D1
D0
PWM
TX
RX
Digital Input/Output
A0
A1
A2
A3
A4
A5
Analog Input
GND
S2
S1

Refrencias

Hydrogen

http://www.hydrogen-music.org/hcms/

Documentación

http://www.hydrogen-music.org/hcms/node/5

Tutorial

http://www.hydrogen-music.org/hcms/node/7

Videos de Hydrogen

http://www.hydrogen-music.org/hcms/node/44

Arduino

http://www.arduino.cc/es/

Instalacion de Plataforma

http://arduino.cc/es/Main/Software

Lenguaje

http://arduino.cc/es/Reference/HomePage

Ejemplos

http://arduino.cc/es/Tutorial/HomePage

Arduino ONE

http://arduino.cc/en/Main/ArduinoBoardUno

Esquema de Pines

http://es.wikipedia.org/wiki/Arduino#Esquema_de_pines

Distribuidores

http://arduino.cc/es/Main/Buy

DealExtreme

http://s.dealextreme.com/search/arduino

LadyAda

http://www.ladyada.net http://www.adafruit.com

Trabajos Futuros

- Investigar por la implementación de distintos sensores.

- Implementar el diseño de distintos Instrumentos (teclado, guitarra, flauta, etc).

- Investigar si existe o realizar un programa para la pc que sea intermediario entre la entrada midi y el controlador final.
  Que sea Flexible permitiendo configurar distintas salidas midi de a grupos de canales y configurar el nivel de disparo de cada uno.
  De esta manera el programa del Arduino estaría trabajando de forma transparente, escaneando todos los sensores y enviando los datos a su máxima velocidad sin ningún control lógico.