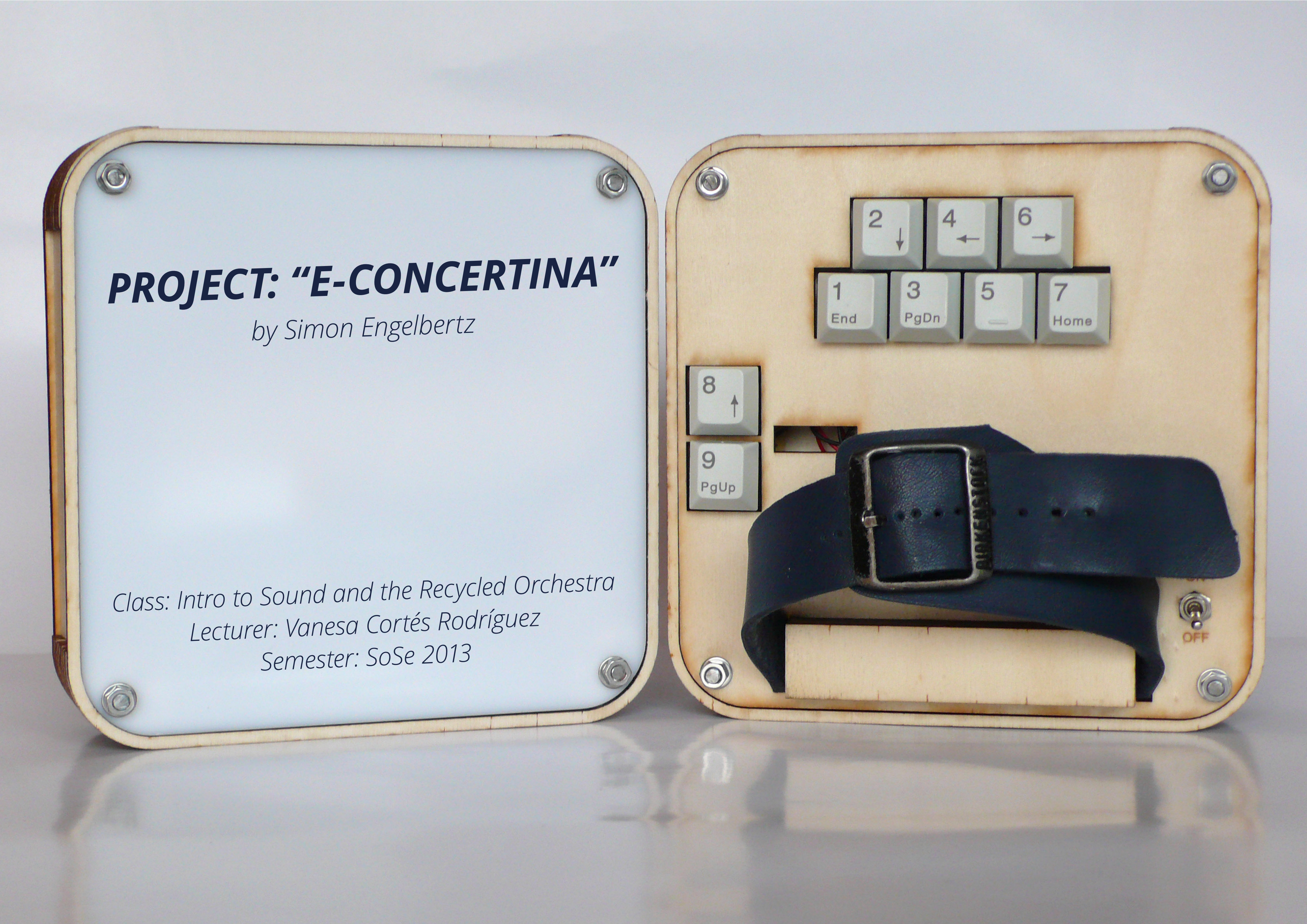


PROJECT: "E-CONCERTINA"

by Simon Engelbertz

Class: Intro to Sound and the Recycled Orchestra
Lecturer: Vanesa Cortés Rodríguez
Semester: SoSe 2013

ABSTRACT

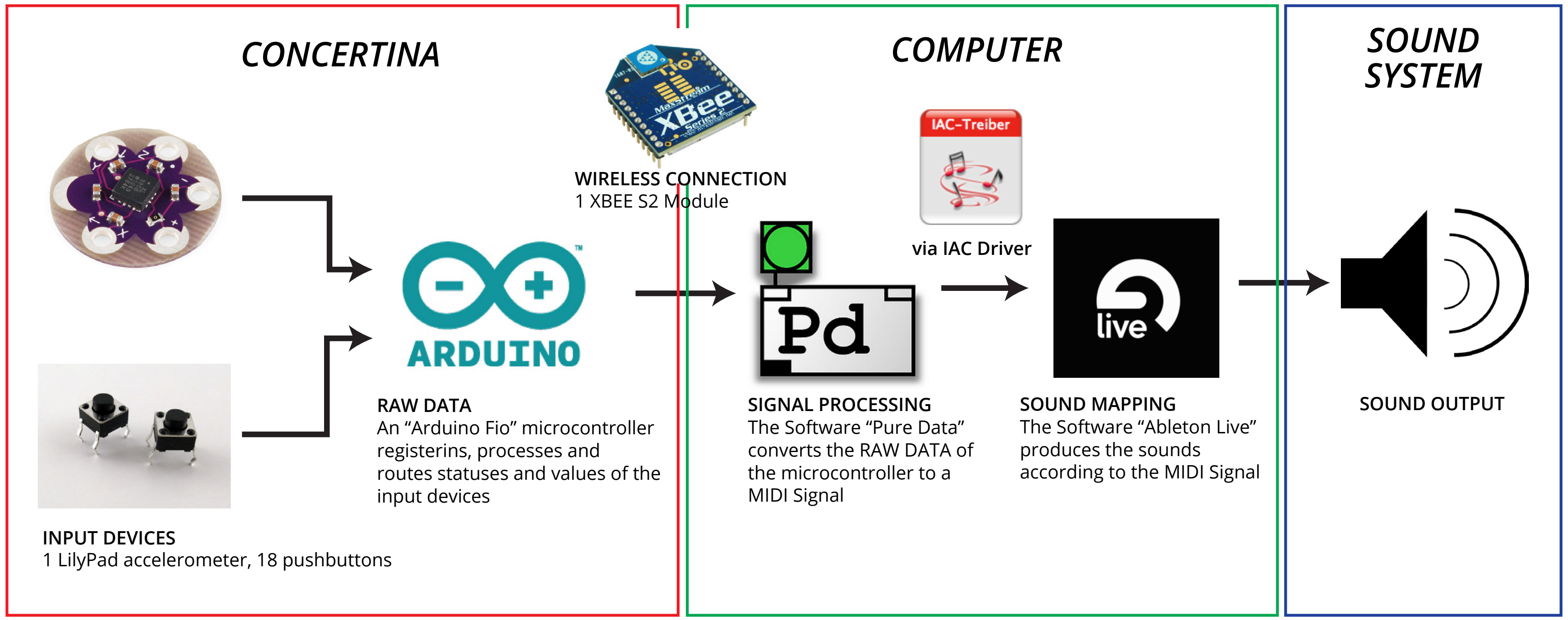
In this documentation I will show the most important working proccesses during my work on building an E-Concertina MIDI Controller. I will start with an introduction in which I´ll refer to the initial idea, my minimum goal and the tasks I set out to do. In the next step I'll show the setup and the signalflow chain of the E-Concertina system. After that I'll picture and describe shortly how I built the case as a basic structure for the E-Concertina. This is followed by a wiring diagram for the electronics which I used in order to make the system interactive. Then I´ll go into the program and will shortly explain what each part of the program does and what it´s good for. Next I'll show a graphic about the MIDI settings in the audio program "Ableton Live" and "Pure Data" which is a real-time graphical programming environment for audio, video and graphical processing. Last but not least I'll show the functionalities of the E-Concertina. The documentation ends with a short summary and a conclusion.

INTRODUCTION

Inspired by the artist Onyx Ashanti and his self-built “Beatjazz Controller” I wanted to create my own E-Accordion MIDI Controller within in the class “Intro to Sound and the Recycled Orchestra”. Since an E-Accordion would have been to comlpex and big for the relatively short amount of time of one semester I decided to cut my idea down to a smaller version of an E-Accordion - an E-Concertina. Therefore my minimum goal within this project was to have a stabilized E-Concertina Midi Controller in the end, which can be linked to some musical software (like Ableton Live) and makes sounds according to the values of the buttons and sensors. Thus my task was to design a case that looks familiar to a Concertina and equip this case with: a microcontroller for the signalprocessing, a sensor(s) to simulate the bellow function of a Concertina, transmitting devices and last but not least some buttons to make the different tones. As for the “Recycled Orchestra” part of this project I was intending to use materials of old things in my household which I don´t need or use anymore. I´ll explain in the following pages what I came up with.

Accordion without Bellows;
Image: http://www.rolandus.com/products/details/1072

Onyx Ashanti;
Image: http://www.kulturzentrum-klausenburg.ro/typo3temp/pics/8747991e96.jpg

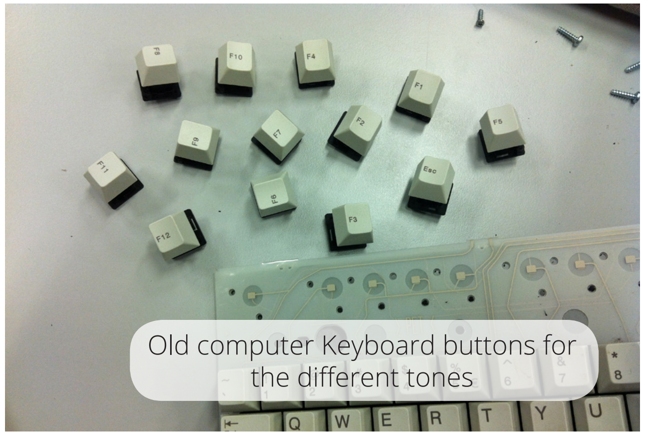

CONCERTINA
COMPUTER
SOUND SYSTEM
WIRELESS CONNECTION
1 XBEE S2 Module
IAC-Treiber
via IAC Driver
ARDUINO
Pd
live
RAW DATA
An “Arduino Fio” microcontroller registerins, processes and routes statuses and values of the input devices
SIGNAL PROCESSING
The Software “Pure Data” converts the RAW DATA of the microcontroller to a MIDI Signal
SOUND MAPPING
The Software “Ableton Live” produces the sounds according to the MIDI Signal
SOUND OUTPUT
INPUT DEVICES
1 LilyPad accelerometer, 18 pushbuttons

SETUP & SIGNALFLOW OF THE E-CONCERTINA

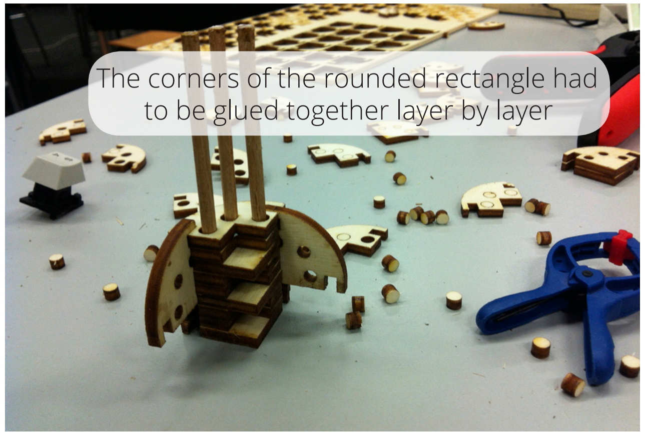
In a simplified form the signal flow of the E-Concertina happens as shown in the figure above. It starts with the Input devices such as the 18 buttons and the LilyPad accelerometer. These parts are connected to the Arduino Fio microcontroller, which registers, processes and routes the “RAW Data” of the input devices. Due to a generic communicative protocol called “Firmata”, which is uploaded on to the Arduino Fio Microcontroller, the “RAW Data” is sent from the Arduino Fio microcontroller to a computer via a XBEE wireless module. The programm Pure Data receives the “RAW Data” and converts it to a MIDI Signal. The Program Ableton Live picks up the MIDI Signal, maps the signal to Concertina soundfiles or any other soundfiles and sends the signal further to the sound output, which can be speakers or any other sound system. In the following I will elaborate more closely how each stage of the signalflow works.

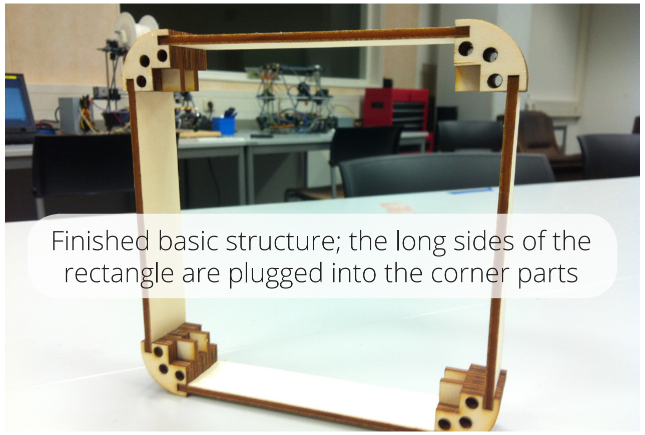
BUILDING THE OF CASE

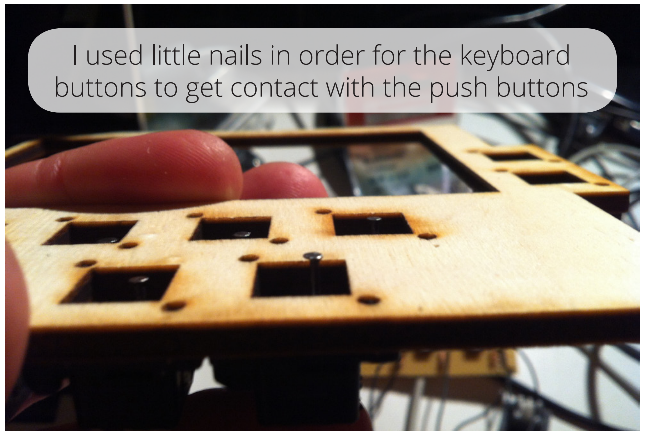
For the case the plan was to build a simple rounded rectangle which is modular and plugged together since I wanted to use as less screws as possible. I used Plywood of 3mm thickness for the basic structure. The template files for each part I designed with the vector graphic program “Inkscape” and I used a Laser Cutter to cut each piece into matching sizes. For the buttons of the E-Concertina I used buttons from an old computer keyboard. The hand straps of the E-Conertina are made out of old “Birkenstock” slippers of mine which were totally worn out.

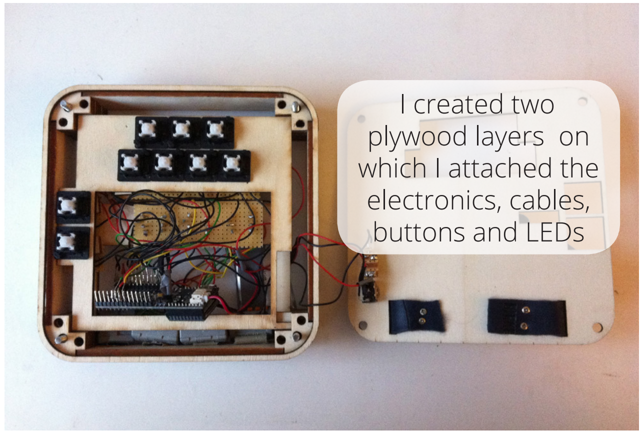

Old computer Keyboard buttons for the different tones

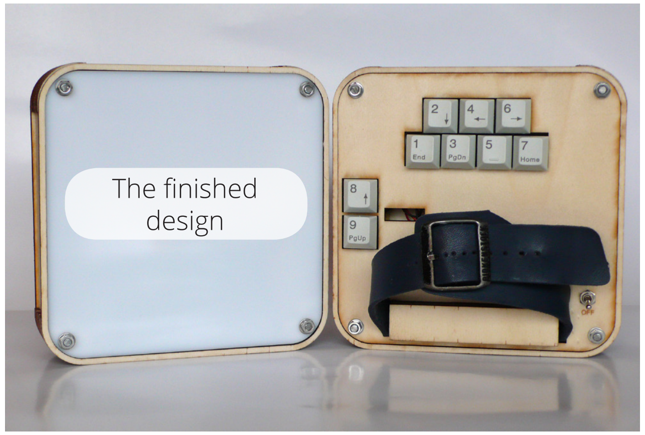
My old Birkenstock slippers used as hand straps


The corners of the rounded rectangle had to be glued together layer by layer

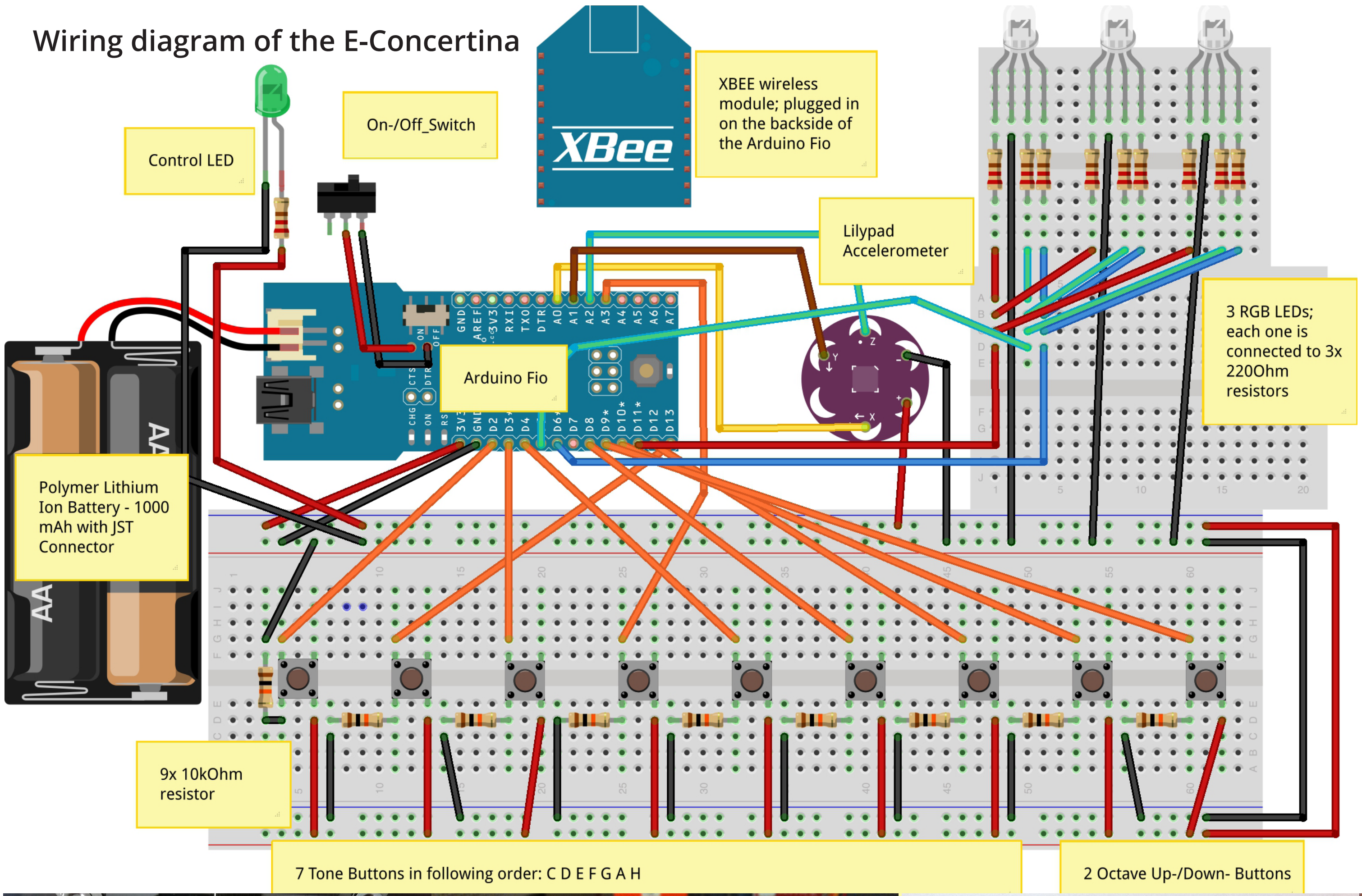

Finished basic structure; the long sides of the rectangle are plugged into the corner parts


I used little nails in order for the keyboard buttons to get contact with the push buttons


I created two plywood layers on which I attached the electronics, cables, buttons and LEDs

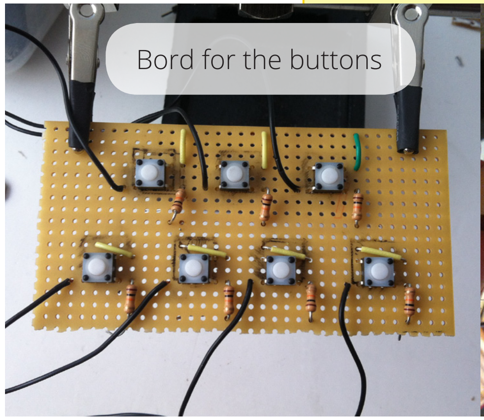

The finished design

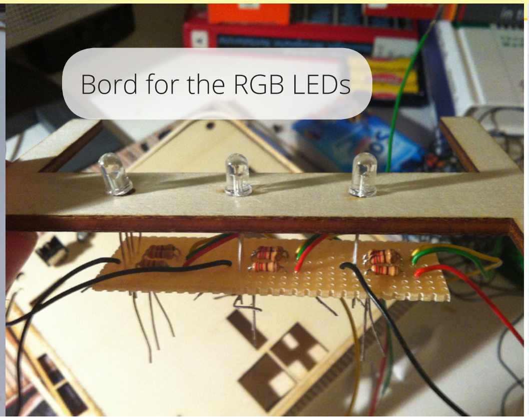
Wiring diagram of the E-Concertina

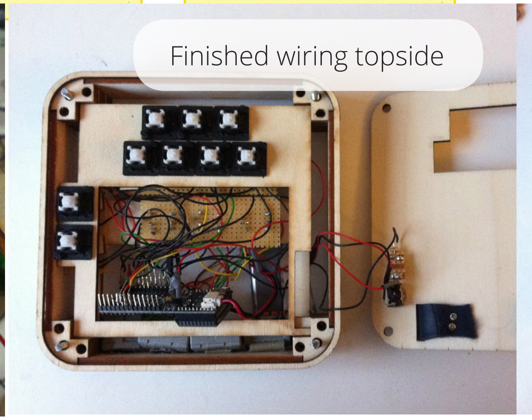

Control LED
On-/Off_Switch
XBee
XBEE wireless module; plugged in on the backside of the Arduino Fio
Lilypad Accelerometer
Arduino Fio
3 RGB LEDs; each one is connected to 3x 220Ohm resistors
Polymer Lithium Ion Battery - 1000 mAh with JST Connector
9x 10kOhm resistor
7 Tone Buttons in following order: C D E F G A H
2 Octave Up-/Down- Buttons

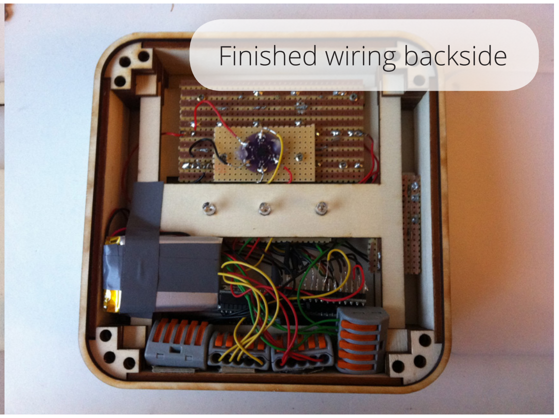
INSIDE THE E-CONCERTINA

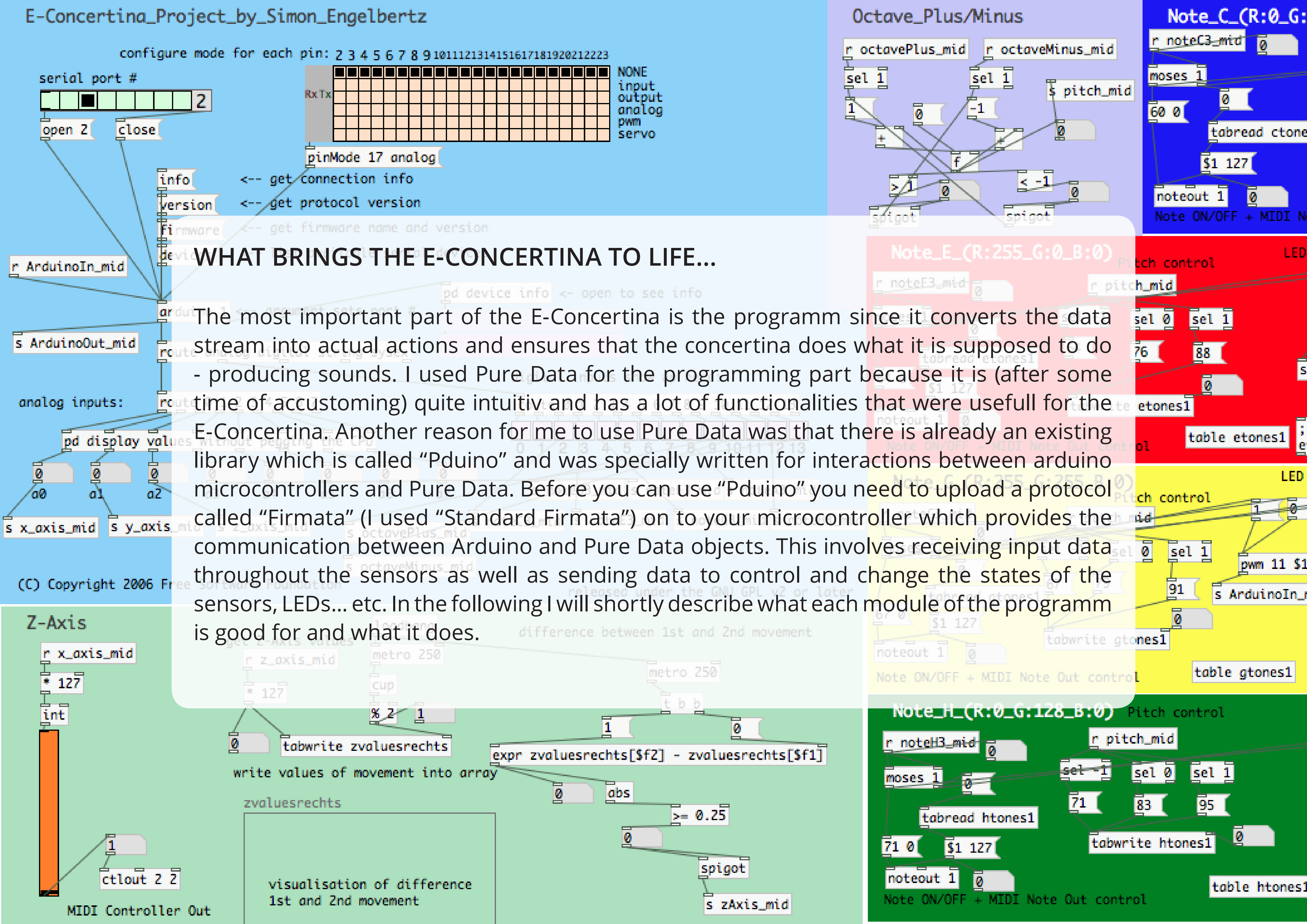
The wiring for the electronical parts of the E-Concertina are as shown in the figure on the left side. For one device I used: an Arduino Fio microcontroller, a XBEE wireless module which I plugged in on the backside of the microcontroller, a LilyPad accelerometer, 3 RGB LEDs with a cathode, 9 push buttons, 1 green control LED, 1 On-/Off-Switch, 10x 220 Ohm resistors, 9x 10kOhm resistors and 1 Polymer Lithium Ion Battery.





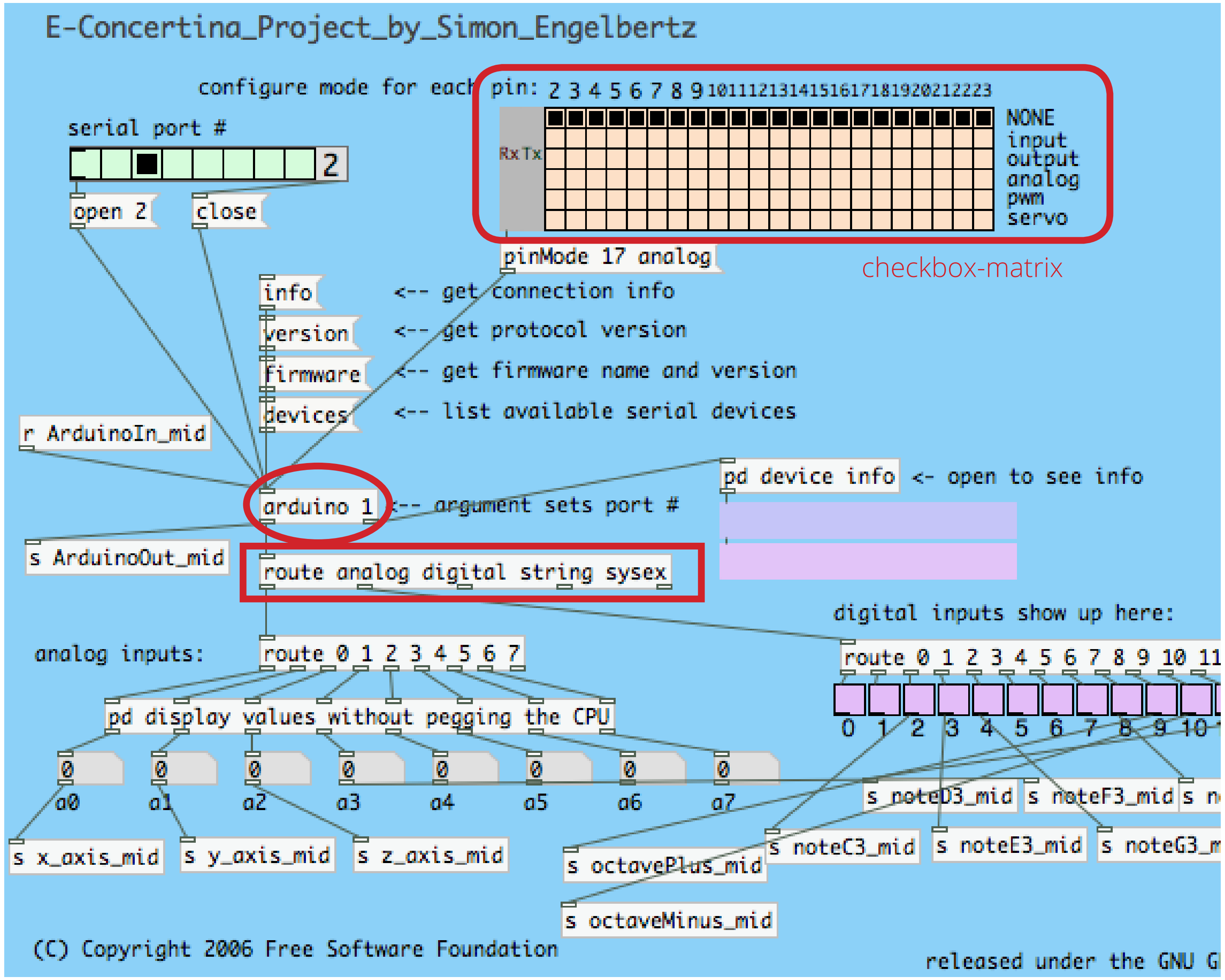






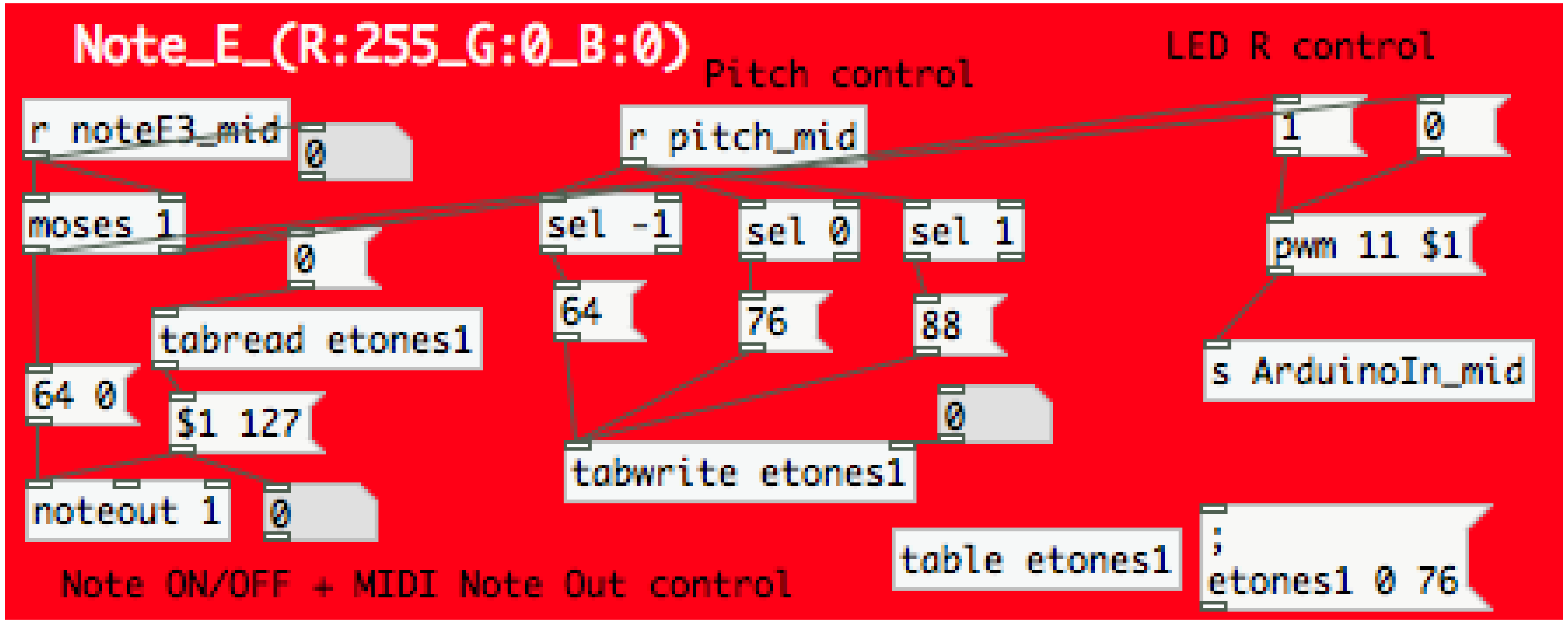
WHAT BRINGS THE E-CONCERTINA TO LIFE...

The most important part of the E-Concertina is the programm since it converts the data stream into actual actions and ensures that the concertina does what it is supposed to do - producing sounds. I used Pure Data for the programming part because it is (after some time of accustoming) quite intuitiv and has a lot of functionalities that were usefull for the E-Concertina. Another reason for me to use Pure Data was that there is already an existing library which is called “Pduino” and was specially written for interactions between arduino microcontrollers and Pure Data. Before you can use “Pduino” you need to upload a protocol called “Firmata” (I used “Standard Firmata”) on to your microcontroller which provides the communication between Arduino and Pure Data objects. This involves receiving input data throughout the sensors as well as sending data to control and change the states of the sensors, LEDs... etc. In the following I will shortly describe what each module of the programm is good for and what it does.



CORE PART

This part is the heart of the programm. Most important is the “arduino 1” object. It has an inlet and an outlet. Everything above this object are commands and messages which are responsible for the general setup of the “Arduino Fio”. With the checkbox-matrix on the upper right for example, you can set the pin-Mode for each Pin of the Arduino...etc. Basically the inlet of the “Arduino 1” object takes care of the data that comes out of the Arduino Fio board. Beneath the “Arduino 1” Object the incoming data is split into it´s different components. The “route” object for example seperates the “analog” and the “digital” values from the Arduino Fio board. The outlet of the “Arduino 1” object also sents out data that goes right to the Arduino Fio board.

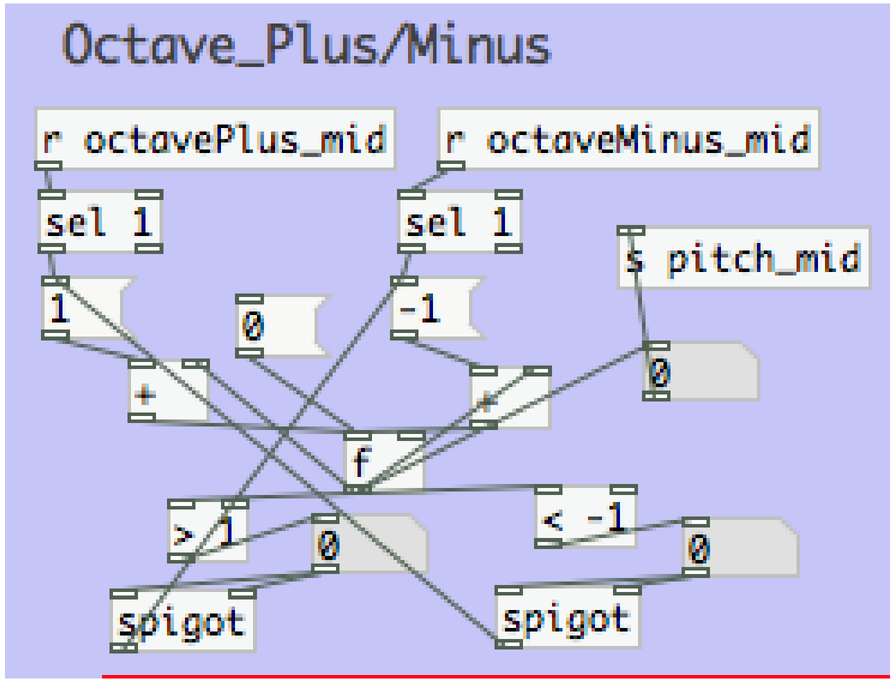


CREATION OF MIDI NOTES

This block takes care of sending out the note “E”. There is a block for each note. So in total there are 7 blocks. It converts “the pushing a button” action into a MIDI Note which is then send out to a music programm like “Ableton Live” via the “IAC Driver” which is a virtual MIDI device. Each MIDI Note consists of the 3 attributes “Pitch, Velocity and Duration”. The “Pitch” value is set according to the “Octave Plus/Minus” Block which I will explain later. Part of this block is also the “LED Control”. This part determines in which color the RGB LEDs should shine. I did the tone-to-color mapping after the color scheme of the chemist George Field who created this scheme in 1841.

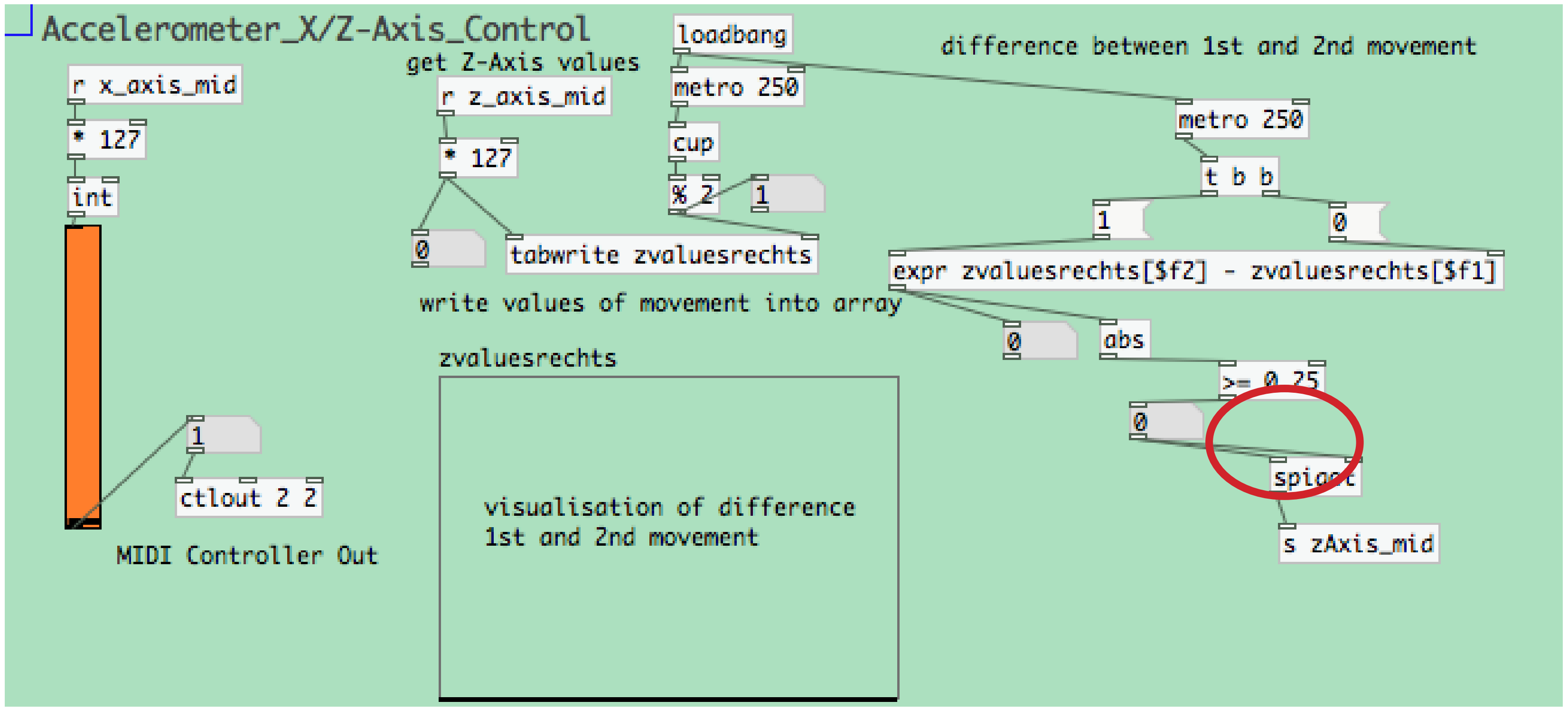
THE OCTAVE JUMPER

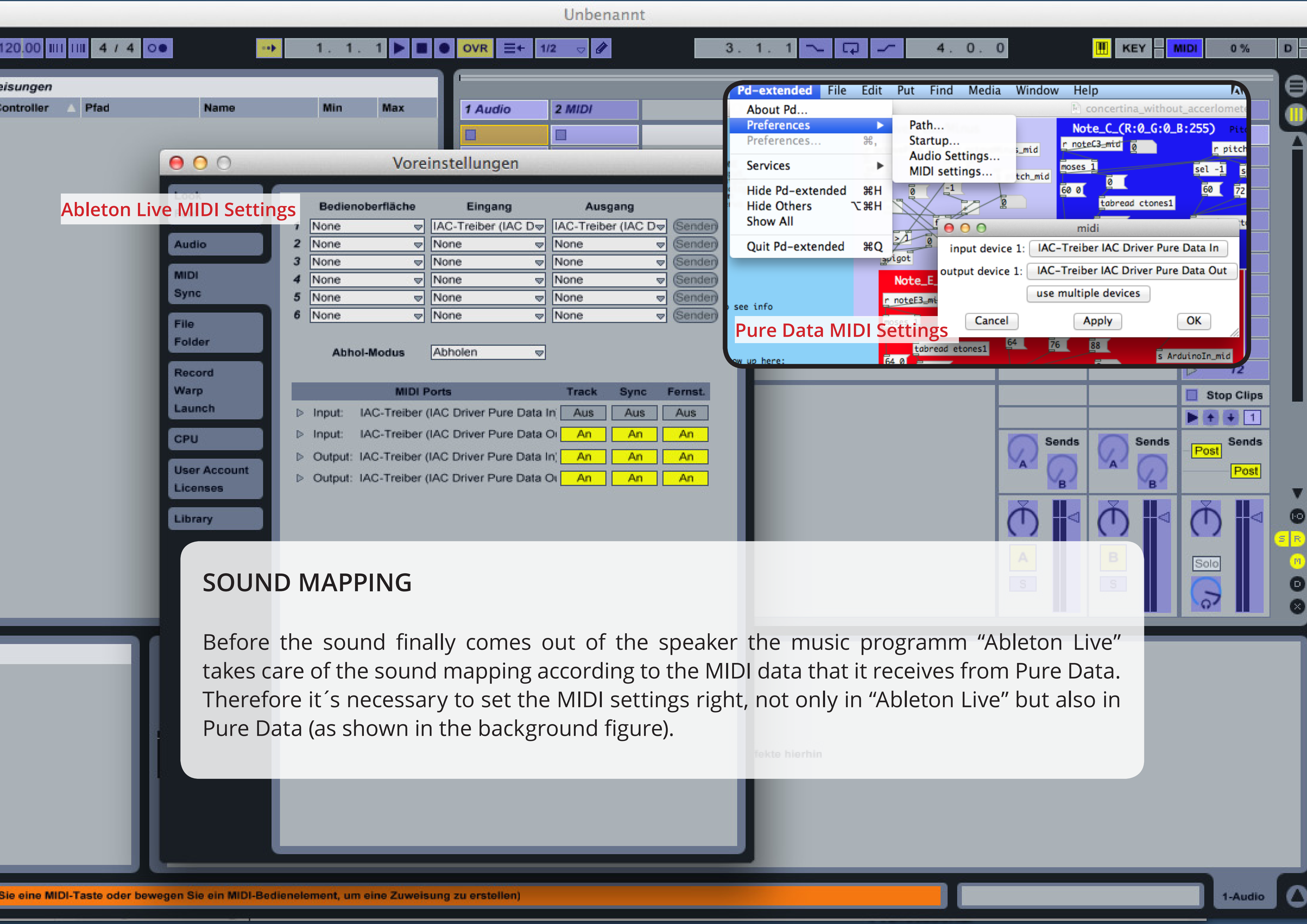
as the name already indicates this block is in charge of providing the information for the octave according to the pushing of the up- and down- buttons on the concertina



THE BELLOW SIMULATION

This block takes care of the bellow simulation, so that the E-Concertina only produces a sounds when it is in movement. It should also control the volume of the sound but this part is still in progress. In order to know if the two ends of the E-Concertina are moving, the programm needs to recall the values of the z-axis of the accelerometer. It always compares two following values and checks if the difference is bigger or equal the amount of 0,25. If it´s less than 0,25 the programm knows that the ends of the E-Concertina are not moving.





	Bedienoberfläche	Eingang	Ausgang	
1	None	IAC-Treiber (IAC D	IAC-Treiber (IAC D	Senden
2	None	None	None	Senden
3	None	None	None	Senden
4	None	None	None	Senden
5	None	None	None	Senden
6	None	None	None	Senden

Abhol-Modus: Abholen

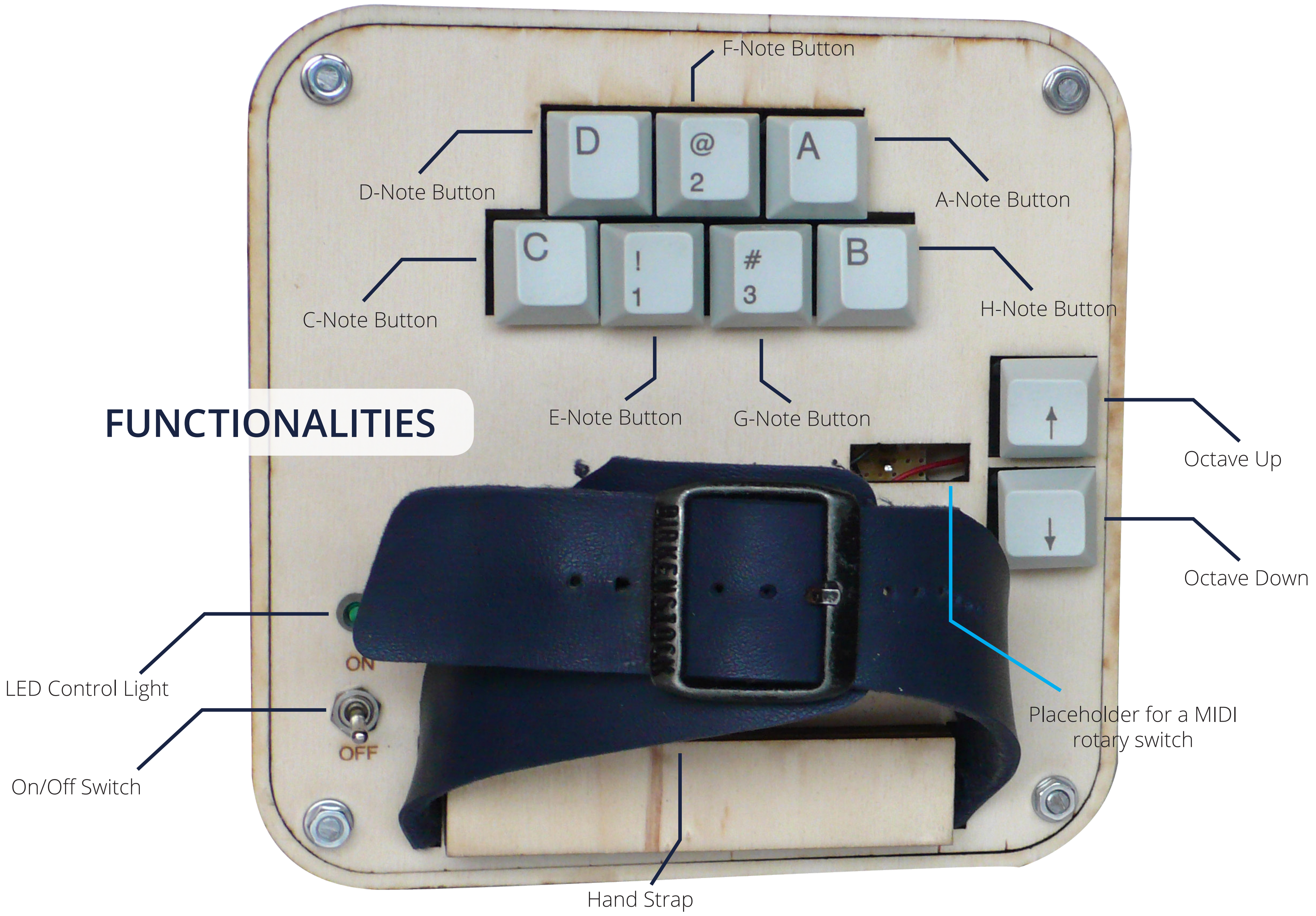
MIDI Ports		Track	Sync	Fernst.
Input:	IAC-Treiber (IAC Driver Pure Data In	Aus	Aus	Aus
Input:	IAC-Treiber (IAC Driver Pure Data O	An	An	An
Output:	IAC-Treiber (IAC Driver Pure Data In	An	An	An
Output:	IAC-Treiber (IAC Driver Pure Data O	An	An	An

input device 1: IAC-Treiber IAC Driver Pure Data In
output device 1: IAC-Treiber IAC Driver Pure Data Out

SOUND MAPPING

Before the sound finally comes out of the speaker the music programm “Ableton Live” takes care of the sound mapping according to the MIDI data that it receives from Pure Data. Therefore it´s necessary to set the MIDI settings right, not only in “Ableton Live” but also in Pure Data (as shown in the background figure).

FUNCTIONALITIES



SUMMARY AND CONCLUSION

Building the E-Concertina MIDI Controller was fun, instructively and ambitious - maybe a little too ambitious for a one semester project. Even though the final result still doesn´t work as I wanted it to, I´m content with it. The future improvements will be regarding the wireless connection with the XBEE modules, the bellow simulation and the haptic of the E-Concertina. Instead of working with the XBEE wireless serial connection, I will try to setup a WIFI network in which both devices of the E-Concertina will have their own IP-addresses. That should make the connection faster and more stable. As for the bellow simulation I´ll want to figure out a better way how to calibrate the accelerometer in order to get more reliable data. Last but not least I would like to improve the haptic of the E-Concertina, since some of the buttons - especially the octave up/down buttons - are hard to reach.