# DSRmarimba: low-cost, open source actuated acoustic marimba framework

**Shawn Trail**
Computer Science, Music
University of Victoria
trl77@uvic.ca

**Leonardo Jenkins**
Electrical Engineering
University of Victoria
leonardo.jenkins@gmail.com

**Peter Driessen**
Electrical Engineering
University of Victoria
peter@ece.uvic.ca

## ABSTRACT

This paper outlines the motivation, design and development of the *DSRmarimba (DSRm.)*. This work presents a portable, autonomous mechanically actuated, digitally controlled musical mallet instrument intended for creative and pedagogical use. Detailed tests performed to optimize technical aspects of the DSRmarimba are described to highlight usability and performance specifications for artists and educators. The DSRm. (Figure 1) is intended to be open-source so that the results are reproducible and expandable using common components with minimal financial constraints and little engineering overhead. Because the field of musical robotics is so new, standardized systems need to be designed from existing paradigms. Such paradigms are typically singular in nature, solely reflecting the idiosyncrasies of the artist and often difficult to reproduce. The DSRm. is an attempt to standardize certain existing and novel robotic pitched-percussion techniques in order to establish a formal system for experimentation.

## 1. INTRODUCTION

In the past few years there has been a growing interest in music robotics. Robotic instruments that generate sound acoustically using actuators have been increasingly developed and used in performances and compositions over the past 10 years. There is a long history of mechanical devices that generate acoustic sounds without direct human interaction- starting from mechanical birds in antiquity to sophisticated player pianos in the early 19th century that could perform arbitrary scores written in piano roll notation. Using computers to control such devices has opened up new possibilities, while retaining the richness of the acoustic sound associated with actual musical instruments. This facet of the natural sound radiation patterns occurring from acoustic instruments, coupled with the sophisticated control paradigms that computers allow make musical robotics a dynamic platform. When observing the electronic music performance convention of the performer using a computer that might have many layers of software synths, samplers, audio loops, and effects processing occurring simultaneously one gets a better perspective on the motivational factors that musical robotics might offer. Simply put, the new instruments being introduced in the research and artistic community are opening up the possibility for a world of acoustic sounds that have yet to be heard, matched with a level of digital interactivity that has never before been possible. Similarly, these instruments inarguably present an unprecedented potential for novel multi-media performance contexts.

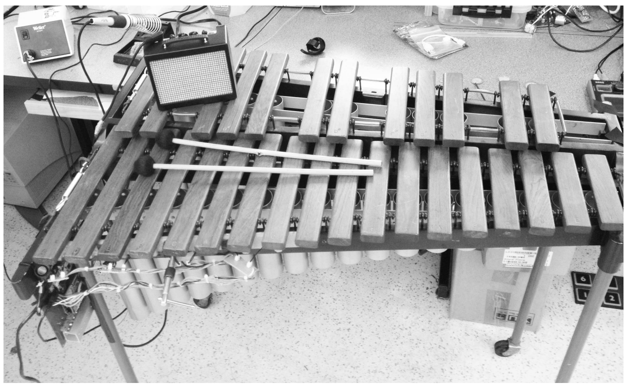

**Figure 1**. DSRm.

Aside from modifying existing acoustic instruments, a designer can invent entirely new instruments considering the physics of available actuators so as to optimize their potential. Adapting acoustic instruments designed specifically for the gestures a human produces using conventional techniques is often difficult to emulate mechanically on any level of musical sophistication comparable to a human's nuances on the same instrument. Typically electronic music requires speakers which flatten all the layered voices that might be occurring in the performance into one focal point radiating outward in the direction the speaker is facing, usually situated in a stereo configuration. In an otherwise all acoustic setting these sounds would have have a spatialized context radiating in a 360 degree pattern while modulating each other's character as well, all the while. Musical robots retain the acoustic, physical vibrations occurring in a truly spatial context while allowing for (at times) unprecedented musical results that humans cannot execute physically as the instrumentalist. The DSRm. has set out to develop, compile, and embed certain sensing and actuation techniques in order to minimize the often cumbersome footprint of actuated pitched-percussion instruments while preserving conventional playing techniques so a performer can play in tandem with the actuated capacity, essentially becoming an all-in-one framework.

The terms music robots or musical robotic instruments have been used to describe such devices. We believe these new robotic instruments have a legitimate place with potential to become part of a conventional musical practice, not just a research curiosity. While musical-robotics might seem niche and esoteric at this point, historic innovations such as monophonic to polyphonic music, electrical amplication of the guitar, or computers in the recording studio were controversial, but eventually became mainstay practices. The recent proliferation of physical computing and the support of user communities dedicated to specific platforms have made working with sound and computers, along with building custom hardware and software for such tasks more accessible than ever to the general public.

Although such music robots have been used in performance of both composed and improvised music, as well as with or without human performers sharing the stage, they are essentially passive output devices that receive control messages and in response actuate mechanisms that produce sound. Their control is typically handled by software written specifically for each piece by the composer/performer. Musicians acquire a body of musical concepts commonly known as musicianship through formalized or self-directed training. Machine musicianship refers to the technology of implementing musical processes such as segmentation, pattern processing and interactive improvisation in computer programs. The majority of existing work in this area has focused on symbolic digital representations of music, typically MIDI. Our system is embedded and as a result uses it's own custom protocol. The typical architecture of interactive music robots is that the control software receives symbolic messages based on what the other performers (robotic or human) are playing as well as messages from some kind of score for the piece. It then sends control messages to the robot in order to trigger the actuators generating the acoustic sound. In our case we interface the solenoids two ways via software. One that is controlled from the tines of a Likembe (a type of lamellophone) via a custom designed interface and another that is controlled from triggers generated by the hits on the xylophone bars by the human performer using piezos.

## 2. DESIGN CONSIDERATIONS

The DSRm. is meant to be portable and compact, yet robust and flexible as a platform for testing various pitched-percussion actuation and interfacing techniques. Design problems that DSRm. addresses for improved functionality and increased flexibility are as follows: low-cost, easy to implement, ready cased solenoids mounted underneath the bars so that the instrument could still be played conventionally and direct signal acquisition for filtering and amplification. Our goals were that the DSRm. be autonomous, eschewing the PC dependancy. We chose the software platform Pure Data (Pd), because it is free and ubiquitous within the computer music community. Pd runs on the Beaglebone [1] , being Linux compatible. This affords embedding the computing environment on the DSRm., while taking advantage of Pd′s existing DSP libraries. This way, we can develop new custom interfaces for the DSRm. that are compatible with our existing framework.

[1] http://beagleboard.org/Products/BeagleBone

### 2.1 Hardware Configuration

The DSRm.′s central computer is the Beaglebone. Pure Data is the software used to communicate between interfaces and the solenoids, as well as being the DSP environment. Both the DSRm. and the Likembe have piezo pickups on them that plug directly into the Beaglebone′s audiocape, which provides stereo audio ADC/DAC via two 1/8th" jacks. Their signal′s are routed and processed in Pure Data then output directly to their own speakers if selected or into the xylophone bars, as described later. The Likembe has a custom Arduino Nano interface that plugs into the Beaglebone′s USB port. This sends the serial data from the sensors into Pure Data for mapping. The Beaglebone communicates to an Arduino UNO, which has a shield and breakout circuit that drives the solenoids via control messages from Pd (see Figure 2), where a custom sequencer 3 has been designed to accept messages from the bar hits via triggers detected by the piezos, or from the capacitive touch interface 12 on the Likembe.

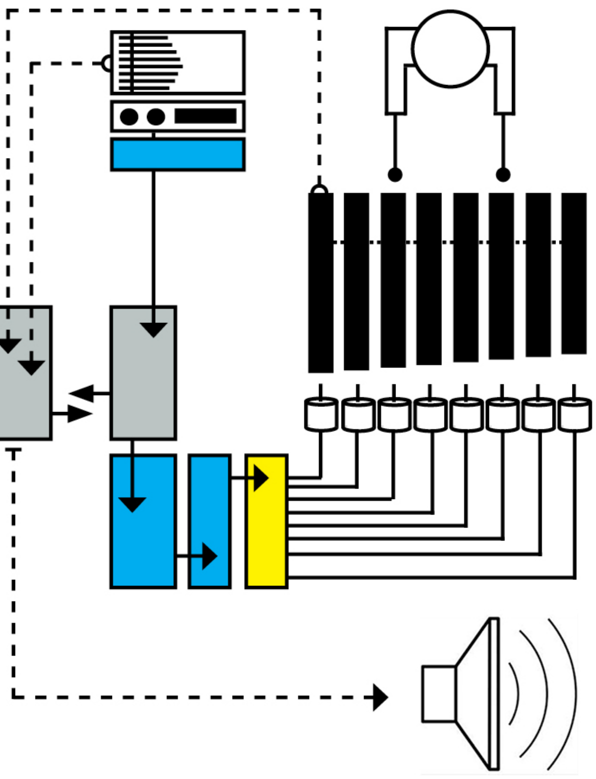

**Figure 2**. Hardware Configuration

### 2.2 Mechanical Design

The current version of the DSRm. is a rapid prototype. Solenoids were spaced appropriately and taped to a ruler (Figure 6) which is "C" clamped to the frame of the marimba. The solenoids patch into a protoboard that has the Arduino Uno and Beaglebone mounted to it 4. This board setup is

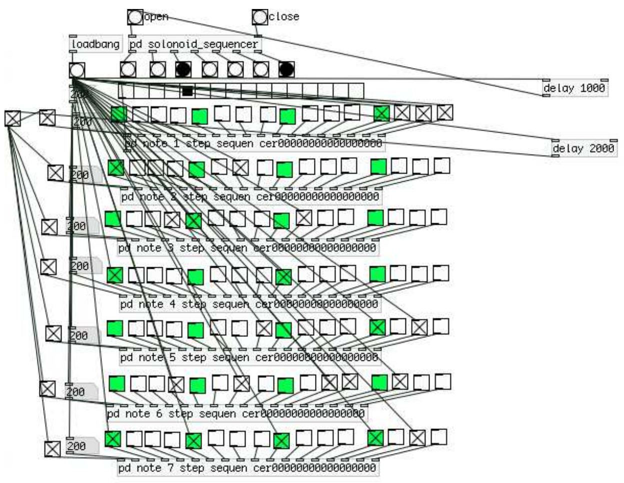
open
close
loadbang
pd solonoid_sequencer
delay 1000
delay 2000
pd note 1 step sequen cer
pd note 2 step sequen cer
pd note 3 step sequen cer
pd note 4 step sequen cer
pd note 5 step sequen cer
pd note 6 step sequen cer
pd note 7 step sequen cer
200

**Figure 3**. Sequencer

designed to be easily removable and reconfigurable. The board is velcro′d to the frame of the marimba for easy removal. Despite it's primitive nature the prototype is quite robust and streamlined for the marimba′s frame, while allowing for easy access to all the I/O of both the Beaglebone and Arduino for maintance. The audio mixer is also built on a protoboard and velcro′d to the frame making it easily removable, as well. The Piezo′s are taped to the underside of the bars and patched into the mixer respectively.

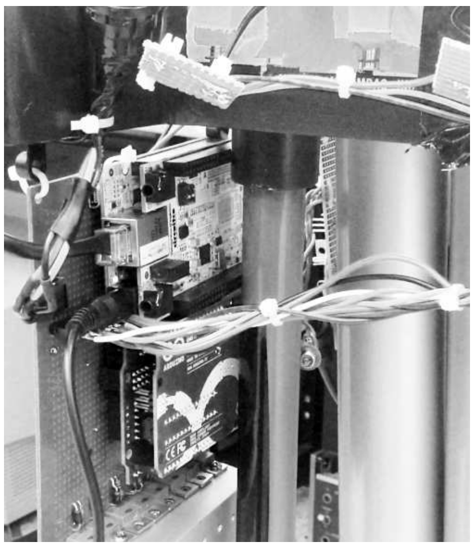

**Figure 4**. Beaglebone/Arduino/Heatsink

### 2.2.1 *Solenoid*

The main goal was to implement an actuation system that did not interfere much with the natural configuration and playing technique of the marimba. For this purpose a set of 5V solenoids were found. These solenoids have a 4.5 mm throw, and where placed underneath the marimba bars that provided strong enough hits (65 gf @ 4 mm throw) to the bars to produce audible signals and permit amplification with a good signal-to-noise ratio. The unit is the *Sparkfun ROB-11015* and only cost 4.95US each (Figure 5).

**Figure 5**. Solonoid

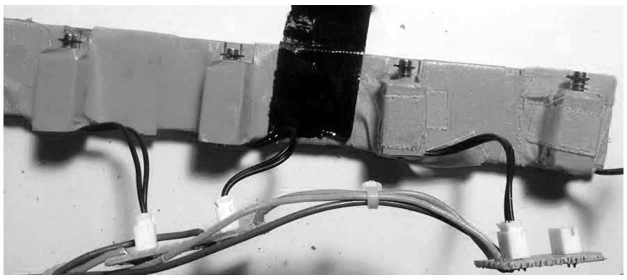

**Figure 6**. Ruler and Solenoids

## 2.3 Electrical Design

### 2.3.1 *Solenoid Actuation*

A circuit based on Darlington-transistors was designed to control the solenoids. An Arduino UNO connected to a sensor shield, drives the solenoids. The schematic shown in Figure 7 represents the system's circuit. A TIP102 Darlington transistor was used, with the Base connected to a digital pin of the Arduino, the collector connected to one of the terminals of the solenoid, and the emitter connected to ground. The other terminal of the solenoid was connected to power. A 1N4004 Diode was connected in parallel to the solenoid, with its anode connected to the collector of the transistor, to dissipate any remaining energy when the solenoid is turned off. A 1K ohm resistor was connected between the Arduino's digital pin and the transistor's base to limit the current, protecting the pin.

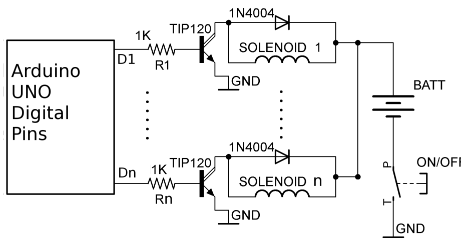

**Figure 7**. Schematic of Solenoid actuation

Initially we tried to achieve a more streamlined and enclosed system, initial testing was done with the Arduino's 5V pin used to power the circuit for the solenoids. Unfortunately this resulted in some unreliable behaviour, with some solenoids not being triggered at all. It was decided then to incorporate a dedicated battery pack to power the solenoids. A power switch was incorporated to disconnect the battery pack from the circuit when not in use.

The Arduino's firmware consists mainly of a routine that

```
int numSol = 7; // number of connected solenoids
int sol; //solenoid number
int led = 13;
// Initialization
void setup() {
  Serial.begin(57600);
  pinMode(led,OUTPUT);
  digitalWrite(led,LOW);
  // initialize the digital pins as an output.
  for(int s=2;s<2+numSol;s++){
    digitalWrite(led,HIGH);
    pinMode(s, OUTPUT); delay(200);
    digitalWrite(s,HIGH); delay(10);
    digitalWrite(s,LOW);  delay(1);
    digitalWrite(led,LOW); delay(200);
  } }
void loop() {
  while (Serial.available()>0){
    sol = Serial.read();
    // trigger the corresponding solenoid
    digitalWrite(sol,HIGH); delay(1);
    digitalWrite(sol,LOW); delay(1);
  } }
```

**Figure 8**. Arduino Code

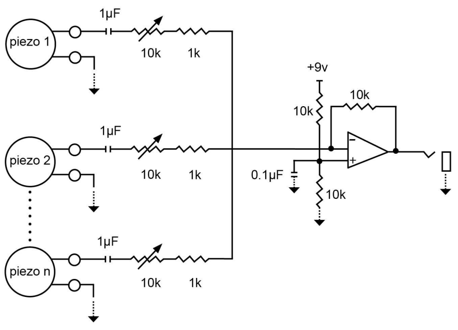

**Figure 9**. piezo summing mixer

reads incoming data from the serial port that communicates when to trigger the corresponding solenoids. When the system is turned on, the microcontroller configures the digital pins connected to the solenoids to Output mode and triggers each solenoid once to make sure that the circuit is functioning correctly. The internal LED (digital pin 13) on the Arduino lights up when every pin is being set up to provide visual feedback (Figure 8).

### 2.3.2 *Audio PIckup and Mixer*

Piezos are used for direct audio acquisition and a summing mixer has been designed to accept all the signals of each bar, outputting a summed mono signal (Figure 9).

### 2.3.3 *DSP*

This project draws from our Open Music Computing Framework expanding previous research [1], which continues to be extended as new musical applications arise. The framework provides analog-to-digital conversion (ADC) by way of a TLV320AIC3106 codec provided by a Beaglebone expansion board, known as the ***Audio Cape***. This expansion also provides digital-to-analog conversion (DAC) for output. Since this framework provides audio interfacing, the DSRm.'s mixer is connected directly into the 1/8" audio input jack in the Beaglebone, which can recieve both the Likembe's and the DSRm.'s signals independently using a standard dual-mono to stereo adaptor.

The framework uses the object oriented, audio programming language **Pure Data (PD)** [2]. Pd is open-source and free. Programming is done in a graphical environment called a *patch* by way of interconnecting processing *objects*, resembling the patching paradigm in analog audio signal chains (or even more simply- a flowchart). Being oriented for audio applications and given its relative ease of use (as compared to low-level C programming typical to embedded DSP), PD facilitates the capture of audio, as well as custom digital signal processing (DSP) techniques.

Both the marimba and Likembe have piezo pickups that input directly into the audiocape. Within Pd their signals are routed independently into various filters for creative processing. For testing we have implemented a multi-channel audio looper, a multi-tap delay with low-pass frequency filtering and a sinusoidal synthesizer instrument. The various parameters of each of the modules is controlled via the Likembe and marimba interfaces.

### 2.3.4 *Bar Triggers*

The piezo signal from each bar that has been summed for audio routing can be repurposed for control signal data providing the amplitude envelope (ADSR: Figure 10) of each hit independently by being split and routed into the analog inputs of the of the Beaglebone (Figure 11 - image courtesy of Todbot [3]). This works the same way as conventional drum triggers. In pd the data is scaled and used the same as any FSR would be for percussion triggering. This offers the possibility of looping musical phrases played by the human that can be continued to be played by the solenoids.

### 2.3.5 *DSR componant - bars as speakers*

DSR stands for decay, sustain, release. When you strike the marimba you naturally have an attack and that's nearly all. With this system you can shape the envelope the same way you would with a synthesizer, only it would all be driven from an acoustic audio waveform triggered by the bar hit. A link to a video demo below shows where the bar hit triggers an oscillator to be played back through the bar. During the video, filters are applied to the signal from the digital oscillator in which case the parameters are modified via a MIDI interface. It can be imagined that the author's previous work with idiomatic and non-invasive gesture sensing for the marimba could be employed to have a fully dynamic system tailored to the DSRm. [2]. The DSRm. builds on the work of C. Britt [3]. However, the

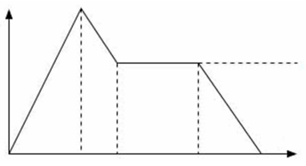

**Figure 10**. Attack Decay Sustain Release

[2] http://puredata.info/
[3] http://todbot.com/blog/2006/10/29/spooky-arduino-projects-4-and-musical-arduino/

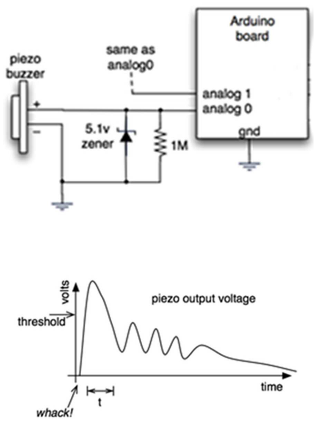

**Figure 11**. Piezo schematic and reponse

DSRm. has the potential for broader extended acoustic sound design capabilities, as well as the option to acquire a direct signal for filtering possibilities and/or amplification using the piezos as both sensors and actuators. The EmVibe only is able to produce long sustaining tones with minimal dynamic range. This is a result of driving the bars to reach their self-resonating frequency and offers little control of the dynamics and no possibility of direct signal acquisition of the bar's acoustic sound. Also, in contrast to the EmVibe, the DSRm.'s digitally extended sounds are not interfered by traditional striking of the bars by a human, which allows 3 diminsions of playabitlty: 1. with the mallets; 2. with the solonoids; and 3. with the audio signals. Given that there are 30+ bars on the marimba, theoretically, one could perceive there to be that many speakers and/or solonoids allowing for a wide range of sonic possibilities. The EmVibe's major drawback, besides the lack of dynamic control over the actuated sound, is that the extended sound is dampened and compromised when the bar is struck by the mallet, this is not the case with the DSRm..

### 2.4 Interface

A hyperinstrument [4] was developed to interface the DSRm. A likembe with a custom sensor interface has been designed and built to interface the sequencer of the DSRm. A Likembe is a type of Lamellophone. Lamellophones have a series of tines that are fixed at one end and free on the other. The musical tone is produced by depressing and releasing the free end with the thumb allowing the tine to freely vibrate. A capacitive touch interface has been created to allow for communication directly with the DSRm. via Pd on the Beaglebone via the tines. A press of the tine will result in the corresponding marimba bar to be struck by the solenoid without interrupting play on the Likembe- creating a symbiotic duo instrumental relationship. This instrument is called El-Lamellophone (Ella) [5] and is a framework for lamellophone hyperinstruments. Ella, combined with the sensing paradigms developed specifically for mallet instruments in the author's previous work ([6], [7], [8]) offer a complete sensing and DSP paradigm for pitched percussion providing unprecedented and seamless integration with the computer. This level of custom multi-model interfacing techniques coupled with the various novel sound production methods create a complete performance framework [9]. The DSRm. eschews the laptop and need for speakers, while retaining the power of DSP filters and synthesis capabilities which are designed to be interfaced by the traditional marimbist′s conventional techniques. For a detailed look at the overview of how the interfacing integrates view Figure 12.

## 3. CONCLUSIONS AND FUTURE WORK

The DSRm. is a platform for physically actuated/digitally controlled mallet instruments. It presents specific building blocks as techniques that can be expanded and serves as a foundation to explore various aspects of musical robotics [10]. However, room for improvement exists. Further work needs to be done to improve the digital audio signal fidelity. We will explore optical sensors for audio pickups as introduced by the *Guitarbot* [11] and the *Mechbass* [12], which eliminate unwanted system noise interference. We have initiated this work in our EROSS [13] system for control data and intend to expand it for audio acquisition. We also intend to develop a physical fader interface on the bar using conductive marimba mallets. Other interfacing methods will also be explored. The robustness of the audio actuation system will be further developed as we attempt to implement a multi-channel version where many bars can be actuated independently with different types of signals on the various respective bars. We plan to introduce wireless capabilities in order to minimize the footprint and cabling requirements of the system. In conclusion, the DSRm is an inexpensive, relatively easy to implement/use, modular hyper-marimba framework prototype that presents vast potential for previously unattainable sounds and performance based interfacing solutions for digital pitched percussion electro-acoustic music. Using a small micro-computer that eliminates the need for speakers and the laptop, all the DSP capabilities that a conventional music computing platform would afford are retained while introducing an array of entirely new sound production facets to the instrument. The instrument itself (the bars) become the speakers and the instrument remains totally acoustic, able to be performed in it's most simple, traditional capacity, as well as with the new digitally extended features. The author intends on developing a solar powered battery system so that the instrument can be played in a remote setting where no electricity is available. The whole system, including computer has minimal power requirements and could conceivably be battery powered. Since the bars would otherwise be the speakers, the DSRm presents a framework for a computer instrument that could be used in a folk capacity. A user study with results featuring the feedback of experienced musicians is planned to gather feedback about how to improve the instrument so that artists can begin realize repertoire for the system.

### 3.1 System Demo:

Solonoids- https://vimeo.com/72316324
Audio Actuators- https://vimeo.com/46789706

## 4. REFERENCES

[1] D. MacConnell, S. Trail, G. Tzanetakis, P. Driessen, and W. Page, "Reconfigurable autonomous novel guitar effects (range)," in *Proc. Int. Conf. on Sound and Music Computing (SMC)*, 2013.

[2] S. Trail and et al., "Non-invasive sensing and gesture control for pitched percussion hyper-instruments using the kinect," in *Proc. Int. Conf. New Interfaces for Musical Expression (NIME)*, 2012.

[3] C. Britt and et al, "The emvibe: An electromagnetically actuated vibraphone," in *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2012.

[4] T. Machover, "Hyperinstruments - a progress report 1987 - 1991," MIT, Tech. Rep., 1992.

[5] S. Trail, "El-lamellophone - a low-cost diy open framework for acoustic lemellophone based hyperinstruments," in *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2014.

[6] G. Odowichuk, "Sensor fusion: Towards a fully expressive 3d music control interface," in *IEEE Pacific Rim Conf. on Communications, Computers and Signal Processing (PacRim)*, 2011.

[7] S. Trail, T. Fernandes, D. Godlovitch, and G. Tzanetakis, "Direct and surrogate sensing for the gyil african xylophone," in *Proc. Int. Conf. on Sound and Music Computing (SMC)*, 2012.

[8] S. Ness and et al, "Music information robotics: Coping strategies for musically challenged robots," in *Proc. International Society for Music Information Retrieval Conf. (ISMIR)*, 2011.

[9] M. Bretan, M. Cicconet, R. Nikolaidis, and G. Weinberg, "Developing and composing for a robotic musician using different modes of interaction," in *Proceedings of the International Computer Music Conference (ICMC)*, 2012.

[10] A. Kapur and et al, "Ensembles of digitally controlled acoustic instruments," *Computer Music Journal*, 2011.

[11] E. Singer, "Lemur guitarbot: Midi robotic string instrument," in *Proc. Int. Conf. New Interfaces for Musical Expression (NIME)*, 2003.

[12] J. McVay, "Mechbass: A systems overview of a new four-stringed robotic bass guitar," School of Engineering Victoria University, Tech. Rep., 2012.

[13] L. Jenkins and et al, "An easily removable, wireless optical sensing system (eross) for the trumpet," in *Proc. Int. Conf. on New Interfaces for Musical Expression (NIME)*, 2013.

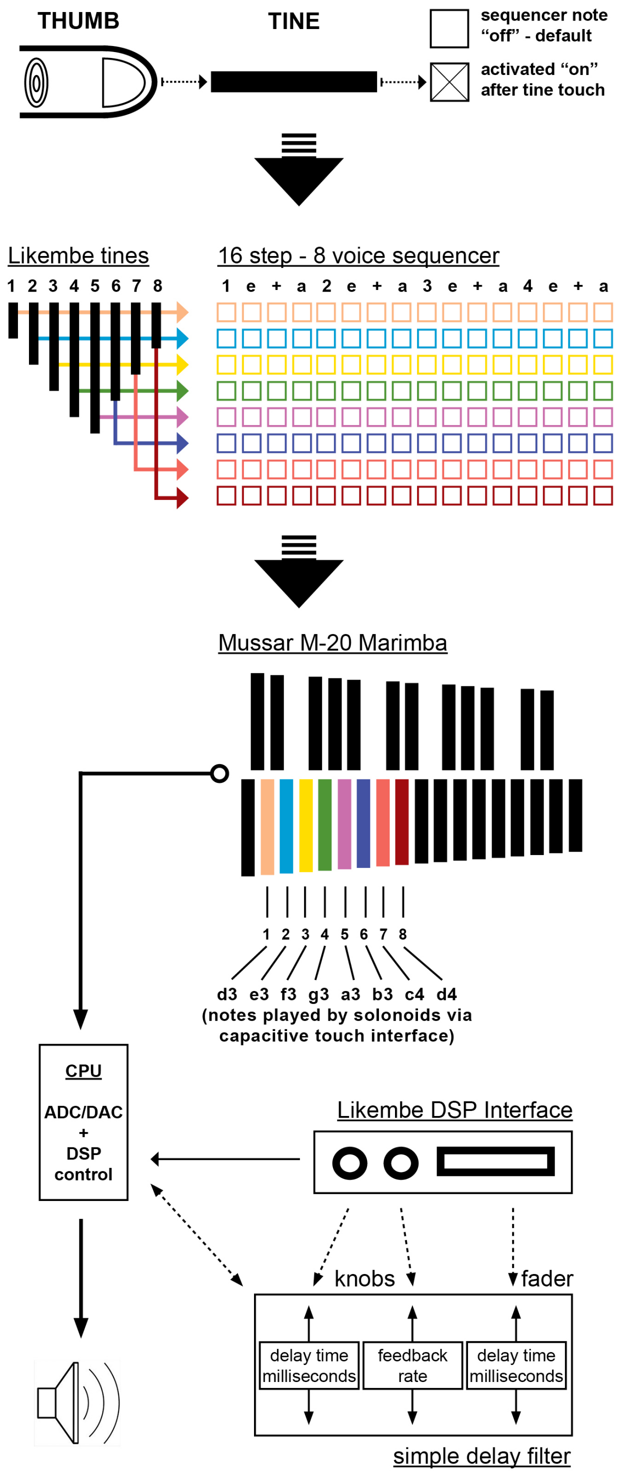

**Figure 12**. System flow