**Welding Machine Musical Instrument**

**Rationale**

To the outsider, welding is a process of merely practical application. Although it is an effective, necessary task, it is something often confined to industry, construction zones, and engineering labs. This project attempts to bridge the gap between science and art through the creation of a welding machine musical instrument.

The tungsten inert gas welding (TIG) process was chosen for its low noise arc and easier ability to control welding parameters. TIG welding uses a torch with adjustable height, angles, and travel speed, with a foot pedal to control supply current. With TIG, the user has greater control over parameters than other welding processes, giving it the potential to be a more functional musical instrument.

**Design**

*Note: many improvements can and will be made on this imperfect design*

The setup involves using the welding machine as a producer of signals which can be measured and processed by a computer. After processing, these signals will be converted into musical parameters, and played through a speaker. In the future, we may use the welding arc as a speaker, creating a pseudo-feedback loop eliminating the need for an external speaker. However, for now, we will focus on creating the setup with an external speaker.

The first step is to measure welding parameters, such as (but not limited to):

* Voltage
* Current
* Torch angle
* Temperature
* Torch travel speed

As of the end of summer, we focused on using voltage and current as the main parameters.

Next, these parameters from the welding machine must be converted to musical parameters, such as:

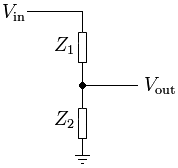
* Pitch
* Volume
* Vibrato
* Timbre

The following sections will outline the methods used for obtaining parameters and converting into sound.

**Part One: Measuring Welding Machine Signals**

Effectively, we will need to create a data acquisition device for the welding machine. Firstly, we must choose a sensor device. For industrial applications, a transducer type sensor using inductance/ Hall Effect would be used for welding machine data acquisition. We strongly considered this idea; however, we would have the possibility of a “black box”, or extraneous software, and not be fully in control of the measurement.

After speaking to a fellow CCWJ member (Mitch), we found it was possible to take readings from the welding machine using simple electrical circuits. The cables would run from the welder to a microprocessor to analyze the data.

To read voltage, cables are connected in parallel from the power supply. The magnitude must be scaled down for readability by electronics. The welding machine operates at ~0-20V, while measurement devices generally have a maximum of ~5V. Therefore, a voltage divider must be used.


V_\mathrm{out} = \frac{Z_2}{Z_1+Z_2} \cdot V_\mathrm{in} 


To measure current, ammeter-type devices are insufficient to handle the large (~200A) currents of a welding machine. A current shunt is the better option; it operates by principle of Ohm’s Law. The shunt, connected in series, has a known resistance that will reduce the voltage of the circuit by a certain amount. The voltage change, which is proportional to the current, can be measured and converted back to current.

Once these signals are gathered, they must be processed by some electronic device. We settled on a full micro-computer, the Raspberry Pi. The Pi is significantly more powerful than a microcontroller such as the Arduino, making it the best choice for our task, especially since data processing and sound output are necessary.

Measuring the signals by the Raspberry Pi requires an analog to digital converter (ADC). Many different factors made choosing a suitable analog to digital converter difficult. For example, the Raspberry Pi can connect using I2C, SPI, and Serial interfaces, which all have different speed and coding implications. Furthermore, each ADC has its own factors, such as sampling rate, resolution, number of channels, input voltage range, and price. In the end, we chose the ADS1015 ADC from Adafruit, a 4-channel 10 bit, 3300 sps converter.

Although we encountered many difficulties setting up the system due to the large learning curve, we were eventually able to create a working measurement device that could measure analog voltage.

The Instrumentation Amplifier - Tutorial

Since the signals received from the current shunt are very low voltages, ~50mV, it is necessary to implement an amplifier to make these signals readable by the ADC.

Wiring: with the little hole facing away, numbering up to down, left column then right, 1 to 6

1 and 5: Resistor. Changes the gain of the amp. Refer to datasheet for exact values.

2: V-in (ground)

3: V+in

4: V- (ground)

6: V+ (from power supply). Note that the amp cannot amplify to greater than input voltage. In fact, maximum is a certain amount lower. (See data sheet)

7: Vout. To ADC

8: Vref. To ground

Note: The values are not smooth without a capacitor. Not sure if correct usage, but placing one from V-in to V- and another from Vref to V+in will smooth values.

Potentiometer - Tutorial

The potentiometer is a useful tool for testing the ADC and as a mini pseudo-instrument later on.

Wiring: with knob facing towards user and pins facing away, number left to right the pins from 1-6.

2 to ground

3 is V in

4 is V out

5 to ground

Note: Potentiometer has logarithmic scaling. Therefore, hard to get precise values, not linear transitions with turning of the knob.

**Part Two - MIDI and Sound Output**

To convert the welding machine signals (e.g. voltage, current) into musical parameters, we decided to use the MIDI protocol. MIDI is essentially a language that gives information to a synthesizer. It is powerful enough to dictate many parameters, such as pitch, volume, etc.

On the Raspebrry Pi, we used the pygame library to create MIDI support. Coding is done using Python. (The script currently used is under /Adafruit-Raspberry-Pi-Python-Code/Adafruit\_ADS1x15/midivol.py) This script uses voltage measurements to change the pitch, and current measurements (which are truly voltage measurements, but from the current shunt) to modify volume. For now, we used a software synthesizer to play the MIDI commands outputted by the python script. This method is simplest (as opposed to using a dedicated hardware synth), but is not the most powerful.

Tutorial for Running FluidSynth Software Synthsizer

Link to Raspberry Pi on Terminal (password: welding)

Start up software synth - type these commands:

/usr/share/sounds/sf2

fluidsynth FluidR3\_GM.sf2 -a ‘alsa’ -m ‘alsa seq’ -p ‘FluidSynth’ -g 1 -s

Start up new Terminal to run commands

cd /Adafruit-Raspberry-Pi-Python-Code/Adafruit\_ADS1x15

aconnect 14 128 (link the Linux alsa input to the fluidsynth output. can do aconnect -io to see input/output options)

run script: sudo python midivol.py

Part 3 - Hook up to Welding Machine

To be continued…

**Extra Notes**

Torch angle – can perhaps introduce vibrato?

* accelerometer: ADXL345, MMA 7361

ADCs

MX7582- although high quality ADC chip, discovered was difficult to interface with Raspberry Pi. Parallel port device, not easy to link to Raspberry Pi. More suited for application through direct link to a processor. To make this chip work, would need an I2C to Parallel converter like this http://www.nxp.com/documents/leaflet/75016079.pdf

Perhaps it is still possible to use this chip. Finding a MagPi article, “Building an oscilloscope with a Raspberry Pi”, a parallel chip was able to create a very fast data acquisition system with the raspi. Running the chips in series increases resolution, running in parallel increases sampling rate.

ADS1015 - set up. some code problems, also must connect input circuit to Vgnd. Code to run ADS1015 from Adafruit sample. Must follow tutorial for installing github library of Adafruit code, or else run into problems. Then, adjust the reading code to suit needs. Create loop to read ADC at certain sampling rate. Soldering necessary, or must hold down chip. Main problem is limitations to sampling rate. This will apply to all chips. Python inherent limitations, possibly improved by “bit-banging”, working with C, using WiringPi. smbus limitations. Also, I2C interface. When there are two inputs (e.g. current and voltage), the sampling speed halves. Maximum sampling rate is divided over the number of channels.

Further notes on Data Acquisition: The Raspberry Pi has the potential to be a powerful data acquisition tool. A MagPi article has shown the Raspberry Pi is powerful enough to make a oscilloscope (10s of MHz range sampling rate). Reaching high sample speeds requires creating a realtime operating environment, with a kernel module. Perhaps more reliable solution is to link a micro controller to provide the ADC to link to, then output data to raspberry pi. Also, using parallel interface is probably more complicated than necessary. Simple bit-banging SPI in a clean manner would give good 10s KHz sampling rates.

MIDI

Running midi using pygame library. Coding using python Somewhat big learning curve, since this kind of coding not completely familiar. seems generally user friendly, can create a framework script then just adjust parameters like volume, instrument, how long to hold note, etc. Took model of a theremin script to get a start

Found out midi commands do not directly equal sound. A soft synth will be required to play the sound. Conversely, a hardware synth can be attached with the midisport interface, and is arguably easier, but don’t have at the moment.

Getting soft synth working on the Pi proves to be a huge pain in the ass. To run, will have to use ALSA or JACK, but running headless creates problems with compatibility. Many soft synths tried give errors. Updating to newest Raspbian very creates a very large problem with the sound architecture. But “FluidSynth" working in the end. Run a server, then execute commands in another terminal window. Seeing large decrease in sampling rate due to increased stress on processor to compute sound commands. Possibly hardware synth is better option, simply output midi commands so that soft synth not needed. More control over realtime sound changes.

Attach potentiometer, it works. Working now to improve tone quality, make it more instrumental. Noticing bugs, for example lag, script not closing on Ctrl+C. Maybe due to power supply not adequate, or overheating. Output over 3.5mm jack is staticky and not good quality. Will add USB sound card

Added another definition to change volume. Frustrating bug in which only one channel was read, but being outputted as both channels. Fixed by copying code from working ADC measurement python script. Volume change is unlike theremin, because midi commands work such that volume cannot be changed after a note is played, unless the note is replayed. Therefore, there can be no crescendo/decrescendo of a single note, only proceeding notes can get quieter/louder.

Introduce vibrato: perhaps started by tilting of torch. Need to use LFO

Hooking up to Welding Machine

Calculations voltage divider, current shunt. Maximum allowed current on ADC is 100 mA. On Syncrowave 200 TIG machine, connect two wires to voltage divider from the power cables (one to torch, one to ground) - this is in parallel. Measure voltage difference. Using 47KOhm and 200KOhm voltage divider, for 0.19x multiplier. Total resistance should be enough to prevent damage to Pi due to current. However, will have to be aware of high frequency start’s damaging potential.