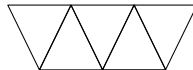
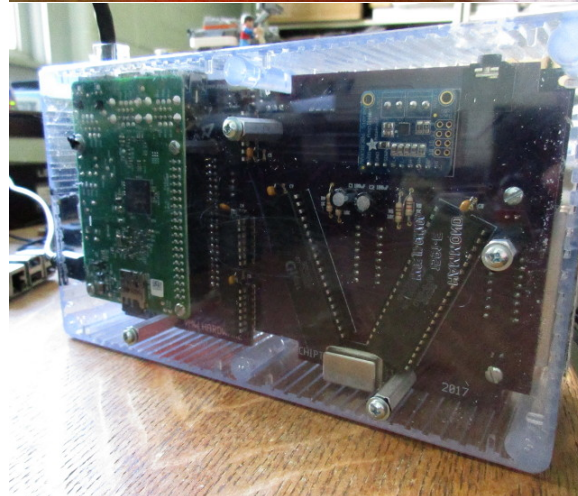


# Building the VMW Raspberry Pi / AY-3-8910 Chiptune Player

by Vincent M. Weaver

<http://deater.net/weave/vmwprod/hardware/ay-3-8910/>

8 September 2017



A VMW Hardware Production

# 1 Introduction

This document describes a custom Raspberry-Pi Chiptune Player with LED visualization. The design contains an EEPROM and can be considered a Pi “hat”, although I have not gotten around to testing or programming the EEPROM interface.

The sound board contains two AY-3-8910 [1] sound chips which were common in the 1980s. These were popular chips, found in the ZX Spectrum and Atari ST. I know of them from their appearance in the Apple II Mockingboard soundcard.

Features found on this device:

- Dual AY-3-8910 for 6-channel stereo sound
- i2c display with 8x16 matrix, 6 bargraphs, 12 characters, and 8 buttons
- i2c real-time-clock so it can be used as an alarm clock
- 1-wire DS18B20 temperature sensor
- Dual 4 Ohm / 3.7W speakers, can also output to stereo 3.5mm jack
- “hat” EEPROM

## 2 Pi Interface

The Pi plugs directly into the board and a large number of GPIOs are used to control things, as seen in Table 1. The AY-3-8910 side runs at 5V so those GPIOs have to be level converted.

### 2.1 AY-3-8910

The AY-3-8910 circuit is more or less the one found as a reference implementation in the datasheet [1].

To limit the number of GPIOs needed, two 74HC595 shift registers are used to provide the parallel inputs to the AY-3-8910. The Pi drives these via the SPI interface (level shifted up to 5V by 74AHCT125N chips).

To actually drive the bus signals, three GPIOs are used, hooked to !RESET, BDIR1 and BC1 on the AY-3-8910s.

The clocks on the AY-3-8910s are driven by a 1MHz crystal, which seems to do a reasonable job. The various 1980s computers used different clock

Table 1: Raspberry Pi GPIO assignments.

Signal	Pi GPIO
74HC595 Shift Register <i>DS</i>	SPI MOSI
74HC595 Shift Register <i>SHCP</i>	SPI CLK
74HC595 Shift Register <i>STCP</i>	SPI CE0
AY-3-8910 <i>BC1</i>	GPIO18
AY-3-8910 <i>B DIR</i>	GPIO23
AY-3-8910 $\overline{RESET}$	GPIO24
DS18B20 1-Wire	GPIO4
Display i2c <i>SDA</i>	SDA1
Display i2c <i>SCL</i>	SCL1
EEPROM i2c <i>SDA</i>	ID_SD
EEPROM i2c <i>SCL</i>	ID_SC
MAX98306 <i>GAIN</i>	GPIO19
MAX98306 <i>GAIN</i> tristate	GPIO26
MAX98306 <i>GAIN'</i>	GPIO16
MAX98306 <i>GAIN'</i> tristate	GPIO20
MAX98306 $\overline{SHDN}$	GPIO21
Speaker Disable Switch	GPIO13

crystals ranging from 1MHz to 2MHz, but with proper scaling of the audio files 1MHz seems to suffice.

There is a rough hand-drawn schematic shown in Figure 1.

## 2.2 Display

The LED displays are driven by the i2c bus. The 14-segment displays and the 8x16 display are standard breakout boards from Adafruit. They have been configured so each has its own i2c address.

The bargraphs are hooked up manually to a ht16k33 breakout board, which is an i2c LED adapter.

The buttons are connected to the ht16k33 breakout via the interface described in the datasheet [2].

No schematic is available for the Display board, I more or less took the breadboard design directly into gEDA/pcb with a long hand-written netlist.

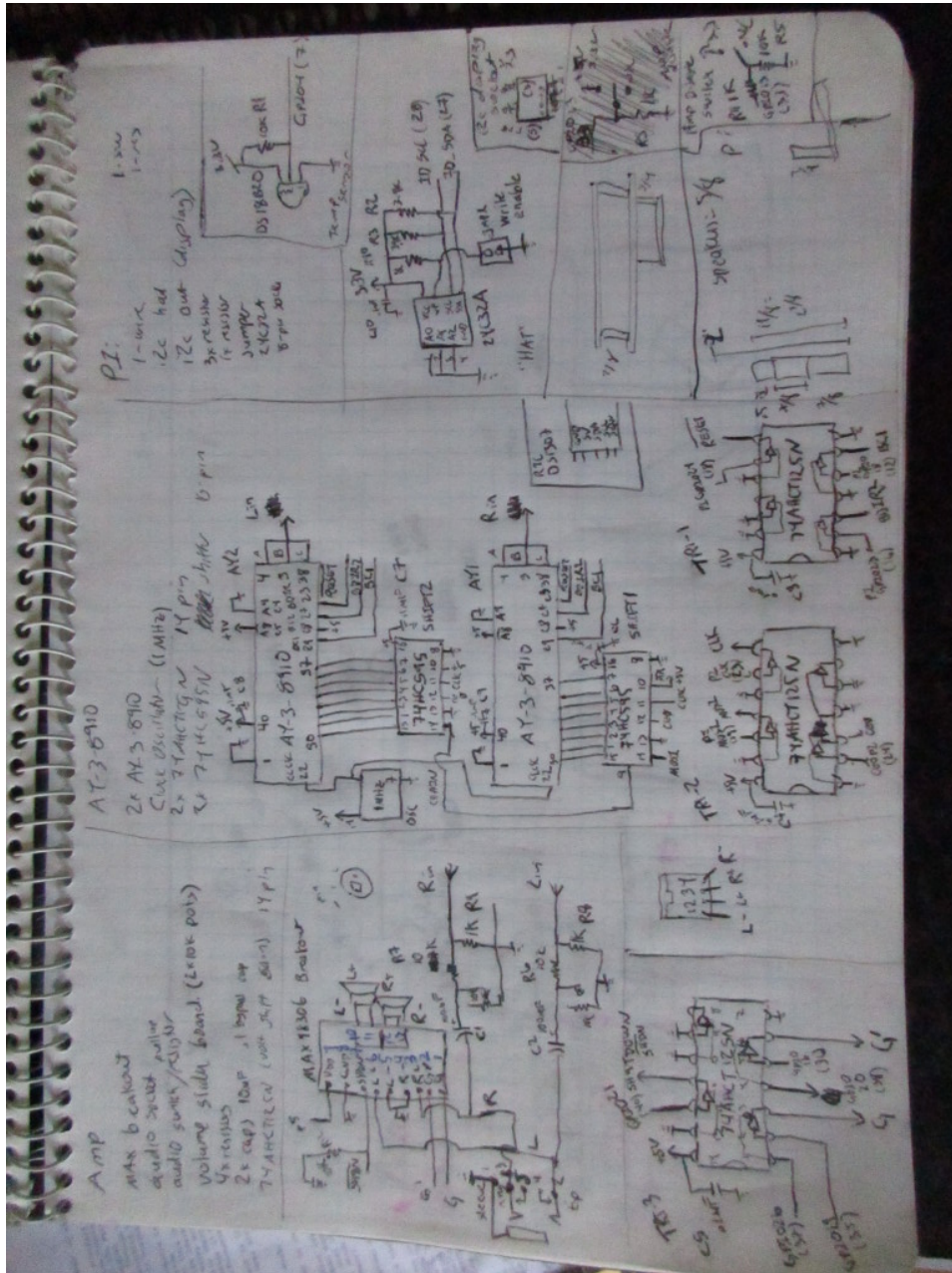


Figure 1: Schematic for the sound board.

## 2.3 Audio

The audio output from the AY-3-8910 is fed into both a 3.5" stereo jack, as well as a MAX98306 class-D amplifier breakout board. The MAX98306 feeds into dual 4 Ohm 3.7W speakers.

The MAX98306 gain can be controlled via input pins, which are hooked to GPIO pins on the raspberry pi (through a 74AHCT125 for 3.3V to 5V level shifting).

This works, but the output on the speakers is often too loud even with the lowest gain setting. Because of this two 10k sliding potentiometers were used to act as analog volume controls.

There is a SPDT slide switch to indicate to the Pi that you want to disable the speakers. This is all in software; it is up to the Pi to monitor the GPIO and turn off the speakers in hardware.

I spent a lot of time trying to design a circuit that could detect the impedance of the jack and automatically cut off the speakers when a plug is inserted. I could not get this to work.

The Pi can shutdown the amplifier using a GPIO line connected to the !SHDN signal. A pull-down resistor is in place to make sure this signal is pulled low until the Pi gets around to driving it at boot.

## 2.4 Peripherals

While designing the board, various extra features were added.

A DS18B20 1-wire temperature sensor was added to monitor for overheating. If thermal issues become a problem then extra ventilation holes can be drilled in the case.

An EEPROM was connected to the Pi i2c-configuration bus. This could be used to autoconfigure the device as a "hat" although I have not tested this functionality.

A DS1307 i2c Real Time Clock was added, so that the Pi can remember the time when disconnected from the internet. This could allow use in alarm-clock type situations.

## 3 Performance

The device functions relatively well when being driven by a Raspberry Pi-2B or 3B. I tried using a Pi-B+ but it was not able to run things smoothly. It is likely Linux was interrupting and messing up the real time handling

Table 2: Original bitbang GPIO timing results. This could not reliably meet the 20ms deadline. Moving to SPI dropped the `ym_play_frame` delay to 8ms which allowed full 50Hz operation.

Routine	Time (ms)
<code>ym_play_frame (gpio)</code>	14ms
<code>display_update (i2c)</code>	1-3ms
<code>display_keypad_read:</code>	0.5ms
<code>display_string</code>	5ms

(especially when I am printing debug messages over Ethernet while playing music).

To play a `ym5` file you need to update the 14 registers every 50Hz (20ms). The traditional `/sys` GPIO interface for Linux is much too slow for this. Instead I use the `bcm2835` library [3].

My original implementation did not use SPI, but bitbanged the shift register GPIOs. As you can see in Table 2 playing music with visualizations could sometimes overflow the 20ms window. Displaying a string on the 14seg displays takes an especially long time as the 3 subdisplays are each on a separate i2c address. By moving to SPI the `ym_play_frame` time reduced to 8ms, allowing full audio and LED effects at 50Hz.

There is still some occasional jitter in the output. The solution might be to move to bare-metal coding (or a real time operating system) as Linux by default is not configured for real-time use.

## 4 Software

I provide software that can drive this board on github:

```
https://github.com/deater/vmw-meter.git
```

You can download it with:

```
git clone https://github.com/deater/vmw-meter.git
```

and look in the `ay-3-8910` directory.

The software includes a YM5 player, conversion software, a rough tracker for creating your own chiptunes, and more.

## 5 Building

Note that you can build just the sound board, you don't need to have the i2c display. Also the sound board has lots of extras that you do not need to populate if you don't want to.

Optional parts:

1. The “hat” circuitry (the EEPROM and resistors)
2. The 1-wire temp sensor
3. The real-time clock
4. One half of the sound circuitry (AY-3-8910, etc) if you only want a mono board.

### 5.1 The Primary Sound Board “hat”

Build directions for the sound board. Full list of parts can be found in Table 3.

1. Order the Printed Circuit Boards (PCBs). They are shared at OSH Park, but I provide the gerber files under **hardware** if you want to get them somewhere else. Often you are forced to buy three of each even if you only want one.
2. Solder parts on the AY side (the one with the picture of a man on it) first.
  - (a) Solder the two 40-pin sockets.
  - (b) Solder on the oscillator socket
  - (c) Solder on the three 14-pin sockets. Annoyingly the ones I had were taller than the 40-pin.
  - (d) Solder on the audio jack
  - (e) Solder on the two 1k resistors (brown black red)
  - (f) Solder on the two 10k resistors (brown black orange)
3. Flip the board over to the spaceship side.
  - (a) Solder on the two 16-pin sockets
  - (b) Solder on the 8-pin socket

Table 3: Sound Board Parts.

NAME	WHERE	PART#	QTY	PRICE	TOTAL
PCB	OSH Park	VMW-CHIPTUNE-SOUND-MK1	1	\$45.08	\$45.08
PCB	OSH Park	VMW-CHIPTUNE-SLIDER-MK1	1	\$2.80	\$2.80
AY-3-8910	eBay	AY-3-8910	2	\$3.00	\$6.00
40-pin sockets	Jameco	112311, 40-pin 0.60"	2	\$0.49	\$0.98
16 pin socket	Jameco	37402, 16-pin 0.3"	2	\$0.75	\$1.50
14-pin osc sock	Jameco	133006, 4-pin osc socket	1	\$0.65	\$0.65
14-pin socket	Jameco	37197, 14-pin 0.3"	3	\$0.65	\$1.95
8 pin socket	Jameco	51626, 8-pin 0.3"	1	\$0.45	\$0.45
10k resistors	Jameco	691104, 10k 1/4 Watt 5%	5	\$0.10	\$0.50
3.9k resistors	Jameco	691008, 3.9k 1/4 Watt 5%	2	\$0.10	\$0.20
1k resistors	Jameco	690865, 1k 1/4 Watt 5%	4	\$0.10	\$0.40
0.1uF capacitors	Jameco	544921 ceramic 0.1uF, 50V, 10%	8	\$0.15	\$1.20
100uF capacitors	Jameco	1946228 radial 100uF, 10V, 20%	2	\$0.25	\$0.50
SPDT Switch	Jameco	588220, SPDT 12V	1	\$0.49	\$0.49
Slider pot	Jameco	2237757, 10k 1/5W	2	\$1.89	\$3.78
4-pin header	Jameco	152726 rt-angle 4-pin shroud	2	\$0.49	\$0.98
Rt-angle Bracket	Jameco	1581530 Key 621, 0.25" 4-40	2	\$0.35	\$0.70
4-40 screws	Jameco	2094389, 0.25" 4-40	4	\$0.06	\$0.24
1MHz Oscillator	Jameco	27861	1	\$1.95	\$1.95
Audio Jack	DigiKey	CP1-3525NG-ND 3.5mm stereo	1	\$0.82	\$0.82
i2c EEPROM	Digikey	CAT24C32LI-G-ND	1	\$0.37	\$0.37
40-pin header	Adafruit	2223, 40-pin 10mm stacking	1	\$2.50	\$2.50
1-wire DS18B20	Adafruit	374, DS18B20 temp sensor	1	\$3.95	\$3.95
DS1307 Breakout	Adafruit	3296, DS1307 i2c RTC	1	\$7.50	\$7.50
MAX98306 Break	Adafruit	987, 3.7W Class D Amp	1	\$8.95	\$8.95
74AHCT125N	Adafruit	1787	3	\$1.50	\$4.50
74HC595	Adafruit	450	2	—	\$2.75
					\$106.69



- (c) Solder on three 10k resistors (brown black orange)
  - (d) Solder on two 1k resistors (brown black red)
  - (e) Solder on two 3.9k resistors (orange white red)
4. Flip back to the man side.
- (a) Solder on the 40-pin PI header. To get it the right size, first push the header onto the Pi's pins. Then attach the 18mm stand-offs and push the header through the holes. Then solder, which ensures the pins are the right distance.
  - (b) Solder on five 0.1uF bypass capacitors. The board only has 0.1" spacing which makes this hard.
  - (c) Solder on two 100uF capacitors. Be sure the minus pin goes in the minus hole.
  - (d) Solder on the SPDT switch (if we did this earlier it would made soldering the resistors harder).
5. Flip back to the rocket side
- (a) Solder on three more 0.1uF bypass caps
  - (b) Solder on the DS18B20 1-wire temperature sensor (looks like a transistor). Be sure to line the pins up properly.
  - (c) Solder on the two 4-pin connectors, one for i2c out, one for speaker out.
  - (d) Solder on a 2-pin piece of 0.1" header for the EEPROM write protect. You probably have some extra from the various Adafruit parts.
  - (e) Put together the Adafruit DS1307 RTC board. Solder it in place. Note, you'll have to put a watch battery in or it won't appear on i2c scans.
6. Flip back to the man side.
- (a) Attach the MAX98306 amplifier breakout. Place the 9-pin header into the board and put 4 leftover pins into the speaker outputs. Put the MAX board on top and solder these on. Then flip and solder this to the board. You don't need to solder on the screw terminals or the gain jumper blocks.
7. Slide potentiometer sub-assembly

- (a) Get the slider PCB
- (b) Solder the two slider potentiometers in place.
- (c) Screw in place with the right-angle hardware and four screws.
- (d) Cut small 1/2" or so pieces of wire and solder in place. The two toward top side of main board are ground, bottom two are signal. This can be a tricky bit of soldering.

#### 8. Testing

- (a) Check 5V, 3.3V, and GND continuity
- (b) Hook up a pi. Check for smoke.
- (c) Run `i2cdetect -y 1` and see if RTC found at 0x68 (note a battery must be on the RTC or this won't work)
- (d) Check that the 1-wire DS1820 shows up under `/sys`

#### 9. Populate the sockets

- (a) Put the AY-3-8910 chips in place (you might only want to do one first to make sure you don't fry both)
- (b) Put in the 74AHCT125N chips
- (c) Put in the 74HC595 chips
- (d) Put in the 1MHz oscillator. It's a tight fit next to the AY-3-8910, you might need to snip off the pin-1 indicator tag so it will fit.
- (e) Put in the EEPROM

## 5.2 The Display

These are the build instructions for the i2c display board. This is optional, but looks nice. Full list of parts can be found in Table 4.

1. Get the purple PCB from OSH Park or your preferred PCB provider. You may need to file down the little nubs around the edges, especially in the speaker areas, for it to fit into the case.
2. Solder things on the back
  - (a) Solder on the 4-pin i2c connector.
  - (b) Next the ht16k33 breakout. (Build it first, putting the pin headers on each side) We leave it at the default i2c address (0x70)

Table 4: Display Parts.

NAME	WHERE	PART#	QTY	PRICE	TOTAL
PCB	OSH Park	VMW-CHIPTUNE-DISPLAY-MK1	1	\$45.00	\$45.00
4-pin header	Jameco	152726 rt-angle 4-pin shroud	1	\$0.49	\$0.49
Diode 1N4148	Jameco	36038	1	\$0.04	\$0.04
39k resistors	Jameco	691243 39k, 1/4 Watt, 5%	8	\$0.10	\$0.80
20-pin sockets	Jameco	112248 20-pin 0.3"	6	\$0.17	\$1.02
RGY 10-seg LED	Jameco	2217596	6	\$1.49	\$8.94
ht16k33 breakout	Adafruit	1427	1	\$5.95	\$5.95
Buttons	Adafruit	1010 Omron b3f with tops	8	—	\$5.95
14seg display	Adafruit	1911 red, i2c	3	\$9.95	\$29.85
8x16 LED matrix	Adafruit	2041	1	\$15.95	\$15.95
1/8" spacers	Mouser	749-908-125 0.125" nylon	50	\$0.04	\$2.00
#2 Screws	Mouser	5721-256-2/8-SS 2-56 0.375"	100	—	\$8.26
#2 nuts	Mouser	5721-256-SS 2-56	100	—	\$5.83
					\$130.08

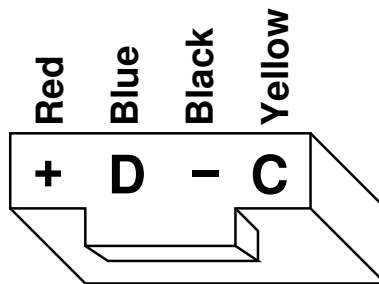


Figure 2: Order of wires on the i2c cable

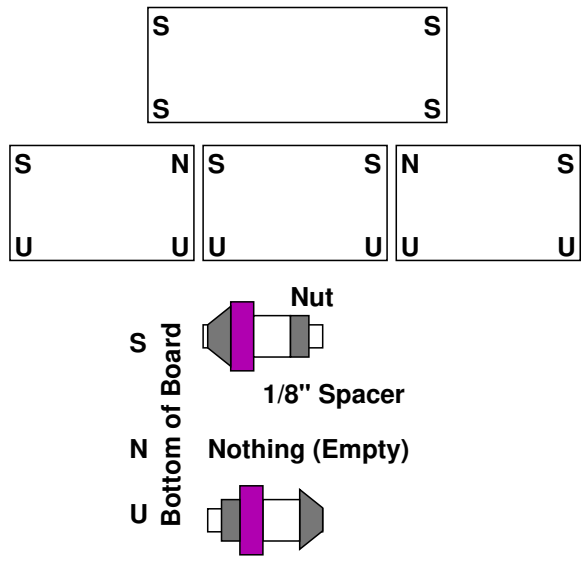


Figure 3: Diagram of screw/nut placement.

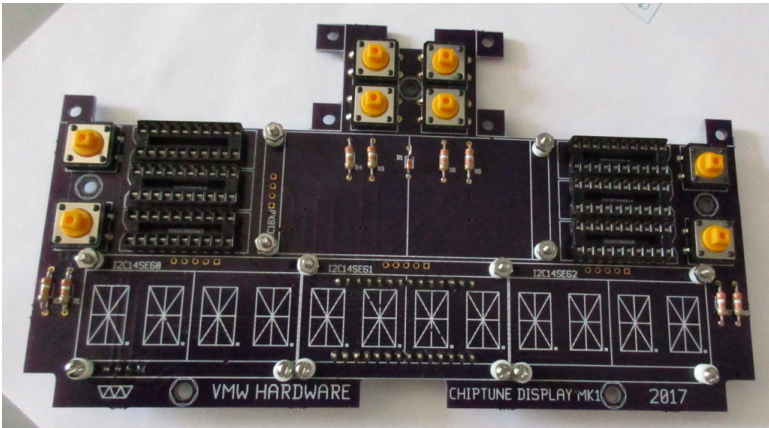


Figure 4: Picture of display board showing screw/nut placement.

- (c) If you already have an i2c cable, now might be a good time to hook things up and test with i2c-scan to see if it is detecting properly. The pinout for the cable is shown in Figure 2, the signals are not marked on the board.
3. Solder things on the front
- (a) Solder on the diode (band direction is indicated, be sure to get it in the right way)
  - (b) Solder on the eight 39k resistors (orange white orange)
  - (c) Next solder in the six 20-pin sockets
  - (d) Next the 8 buttons. There are bumps on the bottom of the switches that slot into holes on the board, which should ensure correct orientation.
4. 14-segment LED boards
- (a) We will be using three of these. Solder the LEDs to the carrier board (follow the instructions from Adafruit, or else by aligning based on the picture on the board). Also solder on the header.
  - (b) Next, assuming you are using #2 screws, we need to enlarge the drill holes on the Adafruit boards as they are just slightly too small. I used a file, a drill can maybe be used.
  - (c) Solder the three boards to have different i2c addresses. Left is 101 (A0=1, A1=0, A2=1), middle 110 (A0=1, A1=1, A2=0), right 111 (A0=1, A1=1, A2=1).
  - (d) Now set up the spacers and screws.
    - i. This didn't really go according to plan (mostly due to nut size and the lower holes being misaligned on the Mark1 board).
    - ii. Figures 3 and 4 have possibly helpful diagrams.
    - iii. Each board will be floating a bit in the air to avoid shorting out against component leads from the other side. The height is equal to one 1/8" spacer plus a nut length.
    - iv. On the top row, put the screw from the bottom, with a spacer and a nut. The board should slip onto this. There is not enough clearance to put an additional nut on top. On the lower ones, put the screw in from \*the top\* going through a spacer with a nut on the other side. The bottom holes on the middle display are too close to the ones on either side, so leave those off.

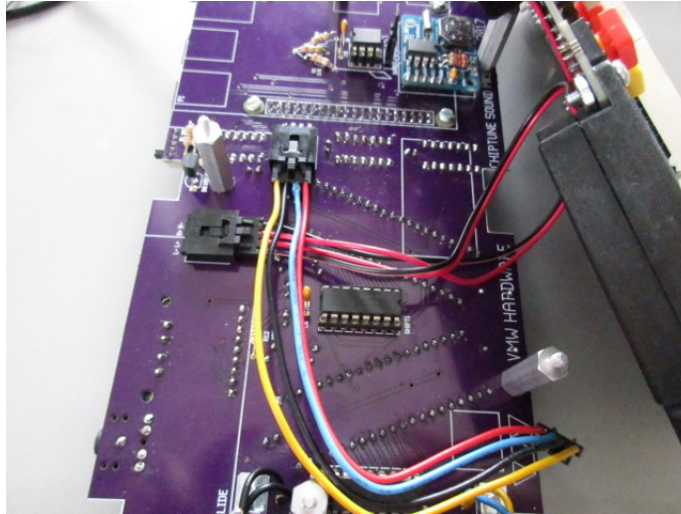


Figure 5: Assembled i2c and speaker wires

#### 5. 8x16 LED matrix board

- (a) Attach the header.
- (b) Configure solderpads for i2c address 010 ( $A_0=0$   $A_1=1$   $A_2=0$ ).
- (c) Expand all four holes on the board with a file
- (d) The holes all work for this one, so put the screw from the bottom, with a spacer and nut on top.

#### 6. Bargraph displays

- (a) Put the 10-segment bargraph displays into the sockets. On the left side of the board the lettering on the displays should face down, on the right side they should face up.

#### 7. Finishing

- (a) Put button tops onto the buttons (colors are up to you)
- (b) Plug in the i2c wire and test (if you haven't been testing all along)

### 5.3 Building the Enclosure / Final Assembly

Putting it all together, list of parts in Table 5.

1. Attaching the Raspberry Pi to the sound board.

Table 5: Case Parts.

NAME	WHERE	PART#	QTY	PRICE	TOTAL
Case	Allied	1591ETCL Hammond clear 7.5x4.3x2.2	1	\$15.71	\$15.71
Raspberry Pi2		?	1	\$35.00	\$35.00
8GB SD-Card	Adafruit	2767	1	\$11.95	\$11.95
Power Supply	Adafruit	1995 5V 2.4A microUSB	1	\$7.50	\$7.50
4 Ohm Speakers	Adafruit	1669 rectangle, 4 Ohm, 3Watt	1	\$7.50	\$7.50
#4 Screws	Jameco	2094397 #4 0.375"	20	\$0.06	\$1.20
#4 nuts	Jameco	40943	20	\$0.06	\$1.20
#4 washers	Jameco	106826	20	\$0.06	\$1.20
Header pins	Jameco	181673	100	\$0.07	\$7.00
Header connector	Jameco	152734 2.54mm connector	3	\$0.59	\$1.77
7/8" standoff	Mouser	534-8404 4-40	10	\$1.00	\$10.00
Standoffs 18mm	Mouser	534-24426 Hex 2.5mm/4.5mm/18mm	4	\$0.52	\$2.08
Nuts 2.5mm	Digikey	36-4707-ND	4	\$0.14	\$0.56
Washers 2.5mm	Digikey	RPC2276-ND	10	\$0.10	\$1.00
Screws 2.5mm	Mouser	534-29300, 2.5mm x 4mm	4	\$0.34	\$1.36
					\$90.03

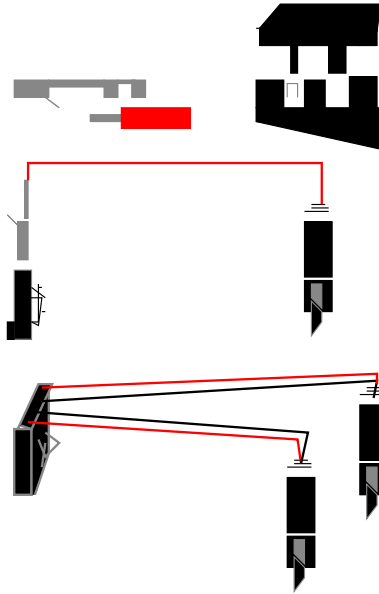


Figure 6: Not very helpful crimping example.

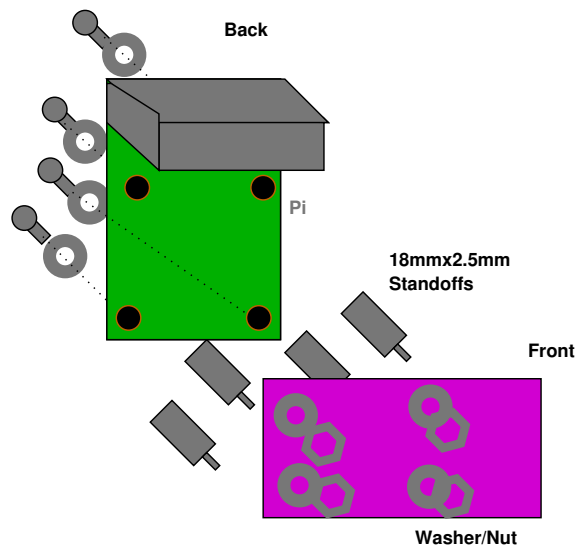


Figure 7: Hastily thrown together pi standoff diagram.



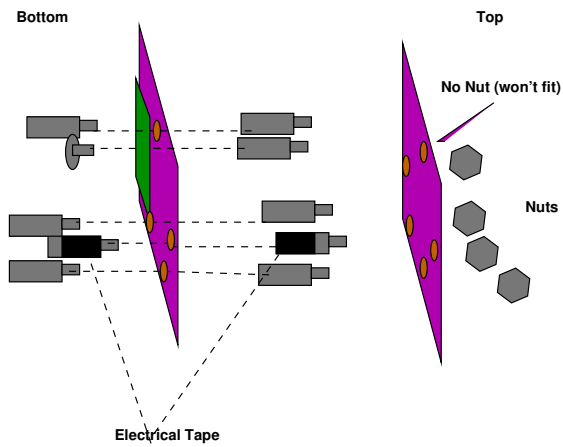


Figure 8: Rough standoff diagram.

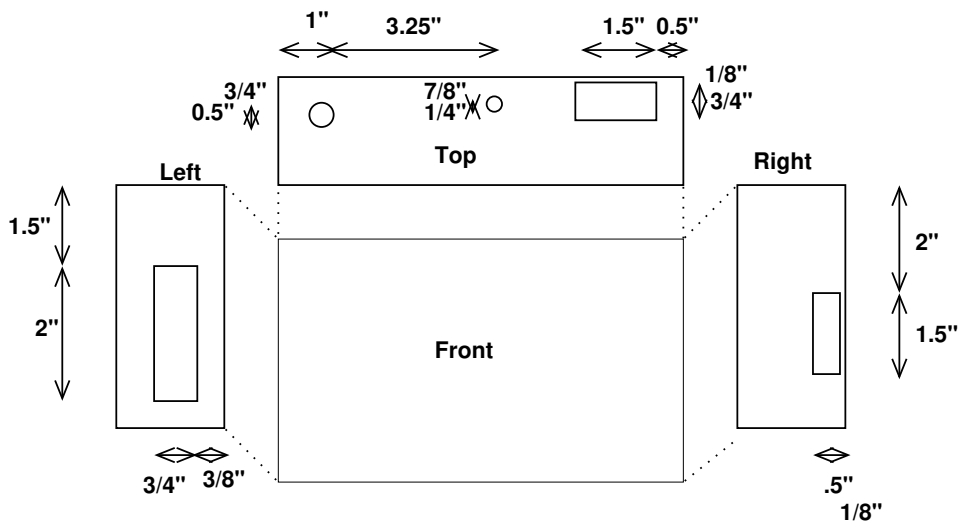


Figure 9: Holes to make in the case.

- (a) I use a pi2. A pi-B+ fits but unfortunately goes a bit too slow with Linux running. A pi-3B seems to work. Not sure about the long term heat solution (be sure to use a heatsink on the CPU).
- (b) If you haven't already, attach the Pi to the sound board using the 18mm standoffs, 2.5mm screws, and washers. See Figure 7. Be careful as once the header is attached it is hard to get off again.
- (c) If in the end the pi doesn't lie flat with the case standoffs, you can add an extra washer before the screw to make the screwhead flush.

## 2. Setting up the Standoffs

- (a) There is a rough diagram in Figure 8.
- (b) Put four of the 7/8" standoffs through the bottom of the sound board (with the AY-3-8910 pointing down). You can't fit one where the pi is blocking things. Instead put a #4 screw and washer there instead.
- (c) Screw five of the 7/8" standoffs on top of the sound board. Put electrical tape around the standoffs with close clearance (the one near the sliders) to avoid shorts.

## 3. Attach the Speakers to the Display board

- (a) The speakers should fit into the upper cutout space on the display board. The wires should come in toward the center of the board, not the edges.
- (b) Use three #4 screws and nuts to attach each to board. The screwheads should be on the top (on the same side as the LEDs).
- (c) You will need to cut the outer (unscrewed) mounting holes off of the outside corners of the speakers so they fit into the case.
- (d) Create the proper connector for the speaker, which is the 4-pin CDROM connector. Trim the wires to about 6" Crimp the connectors to the four wires (this is tricky, see Figure 6 with a crimping tool. Insert the wires into the connector. L-,L+,R-,R+

## 4. Create the proper i2c connector

- (a) Use the same crimper as for the speaker, but this time on both ends. Four wires, each around 6". Make a right angle bend at 2".

- (b) Order is Vdd, SDA, GND, SCL (note this is not marked on the display board). Figure 5 shows the assembled cables.
5. Attach the Display and Sound boards
    - (a) Attach the i2c and speaker connectors (now would be a good time to test)
    - (b) Place the display board on top of the sound board.
    - (c) Use #4 washer and #4 nuts to attach. The center top connector, the nut won't fit due to clearance issues with the switches, so leave that one without a nut on.
  6. Prepare the Case
    - (a) Drill the holes and openings as indicated in Figure 9. I used a hand drill and coping saw to do this, but you probably have better tools at your disposal.
    - (b) Place the assembly inside and be sure it fits. You might need to put the sliding pots through the hole first.
    - (c) Mark on the bottom where to drill mounting holes.
    - (d) Drill then with 7/64th bit.
    - (e) Attach screws through the bottom.
  7. In theory you are all done! Try it out!

## 6 ERRATA!

There are various things to fix if designing a new version of the hardware:

1. Display Board
  - (a) The top mounting hole on the display is too close to the buttons.
  - (b) The i2c signals on the i2c connector are not labeled.
  - (c) The bottom screw holes for the Alphanumeric displays are not in the right place.
2. Music Board
  - (a) The standoff behind the slider is too close to various components

- (b) The standoff on the bottom left completely blocks the Pi memory card slot
- (c) The capacitors should be at least 300 mil not 100 mil
- (d) The oscillator socket is too close to the 40-pin sockets
- (e) Use curves not so many right angles on PCB layout
- (f) The DS1307 board has changed shape from adafruit?

## References

- [1] General Instruments Corporation. *AY-3-8910/8912 Programmable Sound Generator Data Manual*, Feb. 1979.
- [2] Holtek. *HT16K33: RAM Mapping 16\*8 LED Controller Driver with keyscan*, May 2011.
- [3] M. McCauley. C library for Broadcom BCM 2835 as used in Raspberry Pi. <http://www.airspayce.com/mikem/bcm2835/>.