How to build a Smart Boombox

Mike MacIsaac – mike99mac@gmail.com, 862-308-5089

ACM

Poughkeepsie, NY

May 15, 2022

Abstract

- This presentation will describe how to create a "smart boombox" that can play music by voice command. Open source also includes hardware. There will be discussion on woodworking, CNC machines, electronics, sound theory, Systems on Chip, GNU/Linux, distros, desktops, free and open-source media players, voice assistants, and other software. All the hardware and a small software package created by the speaker are free and open. There is also a new section on how ChatGPT might change the nature of personal voice assistants

Outline

- Introduction
- History
- Goals & principles
- Sound design
- Woodworking design
- CNC machines
- Electronics & components
- Computer hardware & software
- Mycroft, OVOS and Minimy
- How to build one
- Demo
- ChatGPT
- Summary and resources

Introduction

- Introduction
  - Mike MacIsaac — mike99mac@gmail.com
  - US Marines — radio repair
  - BS Comp Sci — Northeastern U, Boston MA
  - IT 35+ years — Linux on IBM mainframes, NY
  - Philosophy — “Less is more”
- Based on LinuxCon presentation – Austin TX, June 2022
- Not a sales pitch

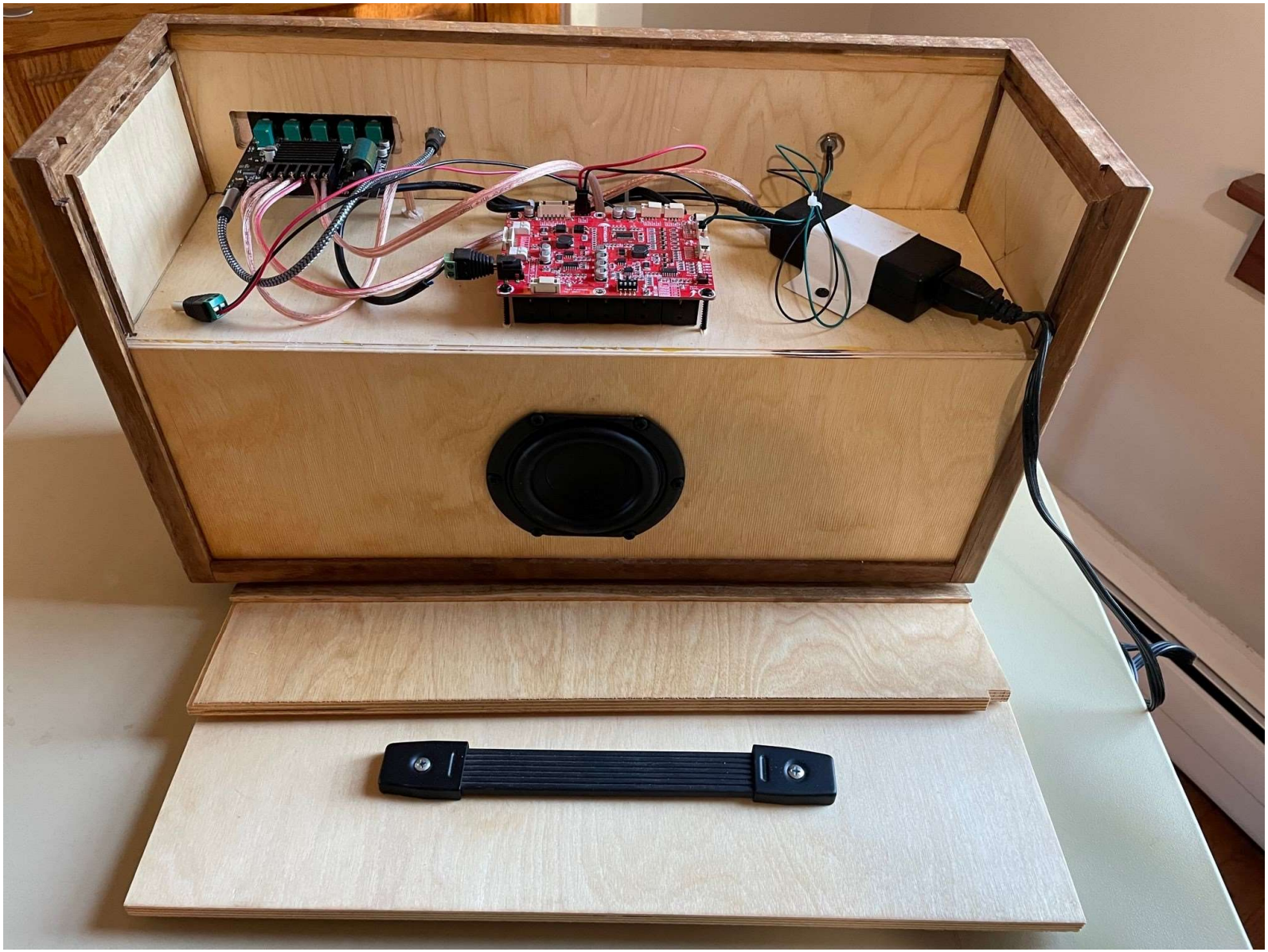
History

- History of the smart boombox
  - Cell phone conference call
  - Cardboard cell phone base with gramophone/Victrola speaker
  - Experiments with larger enclosures
  - 1st boombox Oct 2016
  - 1st smart boombox Jun 2019
  - 1st music skill Dec 2021
  - 17th boombox Mar 2023
  - 1st day Music + OOBs 5/7/2023

Prototype #1

Prototype #15

rear view



3 models

- 1000 Lithium-ion batteries – no computer
- 1250 Model 1000 + Raspberry Pi 400
- 1500 Rasberry Pi on board – no batteries

Goals & principles

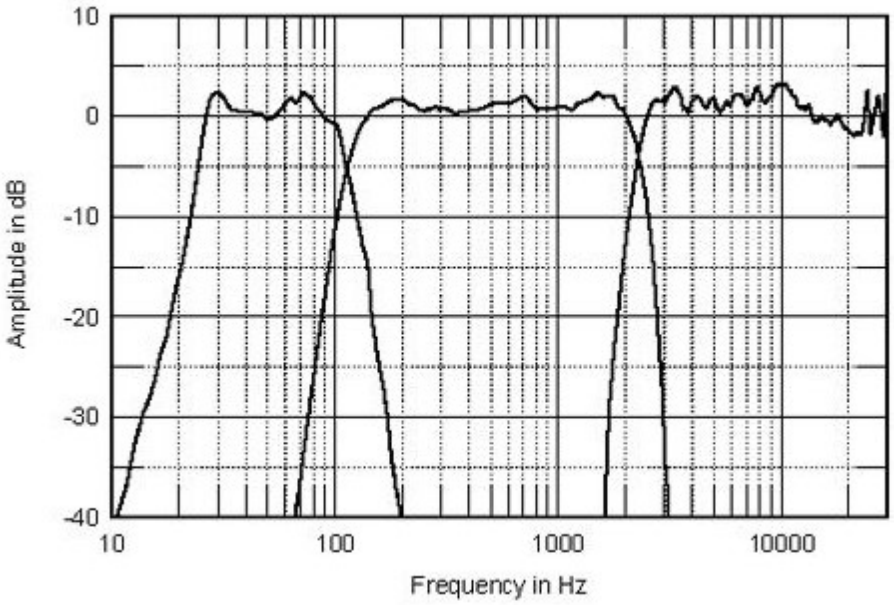
- Goals
  - Great sound (music)
  - Retro look + quality craftsmanship
  - Many input options – Line in, Bluetooth, USB drive, Streaming service
  - Ease of use (UX) => voice input => requires SoC => bonuses:
    - Personal voice assistant – much more than music
    - General purpose (GP) computer

So, is it a boombox that is smart, or a GP computer w/great sound?

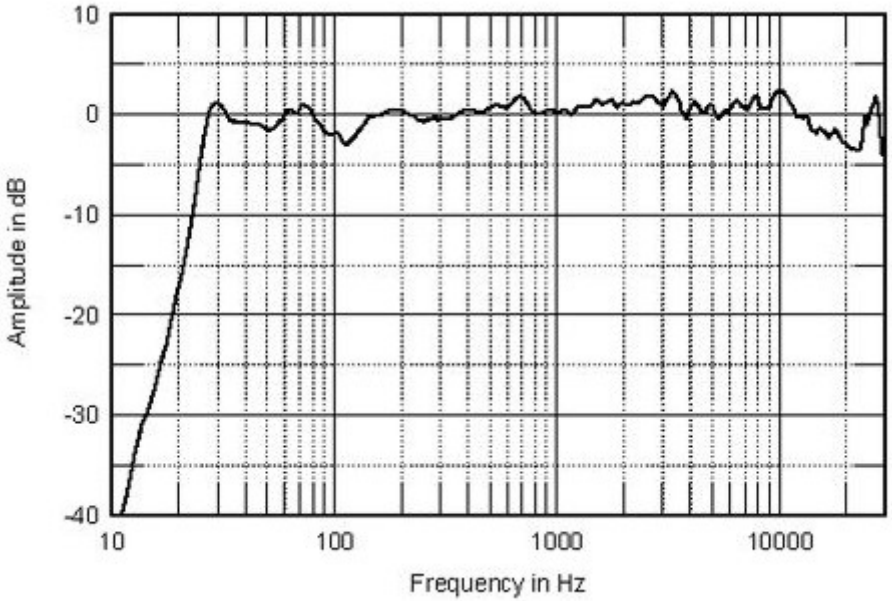
- Principles
  - FOS! – Free and open software - and hardware (FOSH?)
  - Fair compensation for musical artists
  - Privacy

Sound design

- “2.1” system – left, right, subwoofer
- Left and right enclosures
  - Are ported
  - Do mid-woofers with tweeters sound better than *full-range drivers*?
- Subwoofer enclosure
  - *Down-firing* subwoofer
  - Sealed with passive radiator(s)
- Enclosure material: 1/2“ (12mm) Baltic birch plywood
- Use Thiele-Small numbers from manufacturers
  - For enclosure and port sizes
- Sound: the more you know, the more you know you don’t know ☺



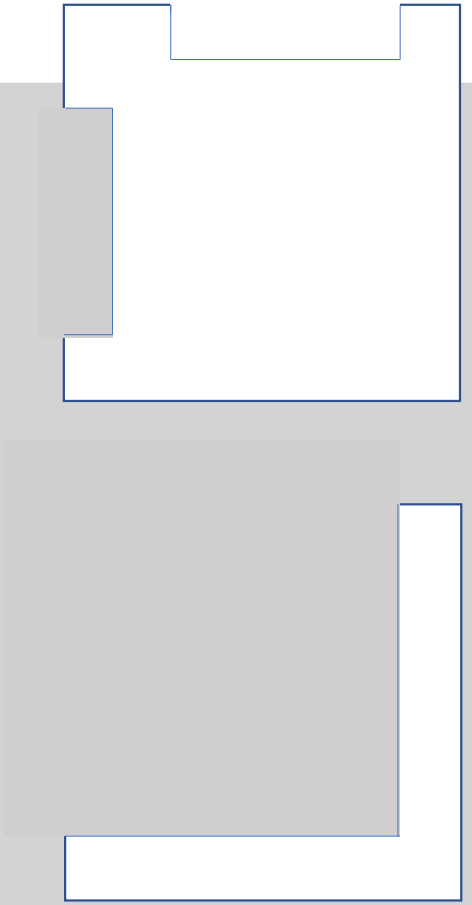
Subwoofer, mid-woofer, tweeter frequency response



Combined frequency response

Woodworking design

- Four enclosures
  - Left and right speakers - ported
  - Subwoofer – sealed w/passive radiators
  - Components - dual removable panels
- Speaker holes and dado joints cut w/CNC machine
- Solid moldings of hardwoods – cut w/table saw
  - Side moldings - dual dadoes
  - Top/bottom moldings – L shaped
  - Inlay moldings to add character
- Wood is glued
  - No screws or nails
  - Only two “glue-ups”

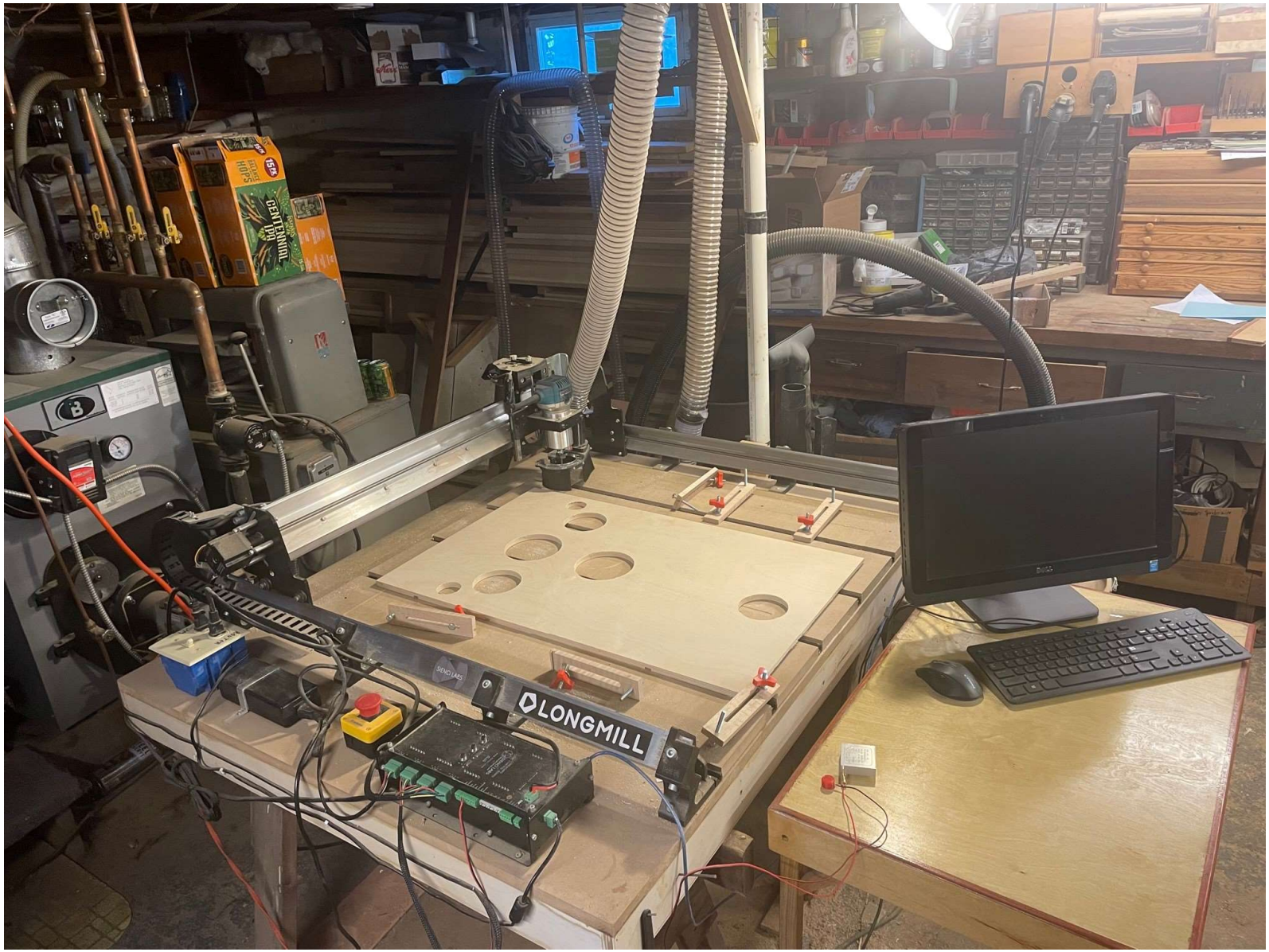


CNC machines

- Computer controlled router cutting in 3 dimensions
- Driven by computer hardware and software
- Input is “G code” (~ computer’s machine code)
- Recommend Sienci Longmill – sienci.com
  - Authors of FOSS gsender
  - Excellent community

CNC Machine

Longmill by Sienci



Dust collection system

Electronics & components

- 115V AC – Electrical box with sockets & cover plate (model 1500)
- Speakers – use best connectors possible
- Amplifier – Use built-in Bluetooth – old and new amps
- Power supplies - AC to DC
  - Both 12-24V and 5V are needed
  - Amp supply needs to power the amp
- Microphone
  - Don’t buy a cheap one
  - Need experiments with “beam forming” microphones
- Line/Aux in – old and new connectors
- Lessons learned
  - Avoid soldering
  - Use quality connectors
  - Use amplifier’s built-in Bluetooth
  - Use tweeters that come with crossover networks
  - Keep power on one side and signals on the other

Connectors and crimping pliers

By Wirefy

Computer hardware

- System on a Chip (SoC)
  - Raspberry Pi – Model 4 w/4GB
    - Best history
    - Best community
    - Due to global chip shortage ☹ they are "unobtanium"
  - Alternatives – Asus, Libre, Odroid, Pine64, etc.
- Power supply
  - For SoC: Use official Raspberry Pi 5V USB-C
  - For Amp: 24V * 5A = 120W
- Digital to Audio Converter (DAC)
  - Hifiberry DAC+ - sweet spot of price: performance
  - Provides a headphone amp - "Audiophile quality"?

Computer software

- OS — Ubuntu Linux Desktop 22.04 LTS
- Desktop — GNU based
- Audio — Pulseaudio and ALSA
- PVA — Minimy (alternatives: Mycroft, OVOS, Neon)
- Music skill — Homegrown
- Bluetooth — Built into Amp (turn off on Linux)

How to build a smart boombox

1. Design
2. Get wood
3. Get components
4. Make drawings
5. Create G-code for CNC
6. Cut main piece on CNC using G-code
7. Cut out all panels and moldings
8. Label panels
9. Glue sides then enclosures
10. Finish
11. Install electronic and computer components
12. Install Linux onto SoC + DAC hat
13. Install and configure Mycroft
14. Test & debug

2 - Get wood

- ½” Baltic birch plywood
  - Local lumberyard
- ¾” Hardwood
  - Local lumberyard
  - Stevewall.com
- Drawings and Easel zip file – email mike99mac@gmail.com
- Might be a kit available? at smartboomboxes.com

Components – model 1000

- 2 Eminance 4” mid-woofers $50 PE: 290-4012
- Tang Band 5-1/4” subwoofer $50 PE: 264-917
- 2 Skar 1” tweeters $43 Amazon.com
- 6” subwoofer speaker grill $3 PE: 260-371
- Dayton 5-1/4” passive radiator $17 PE: 290-217
- Peerless 3-1/2” passive radiator $8 PE: 264-1060

Subtotal (without wood): $171

Components

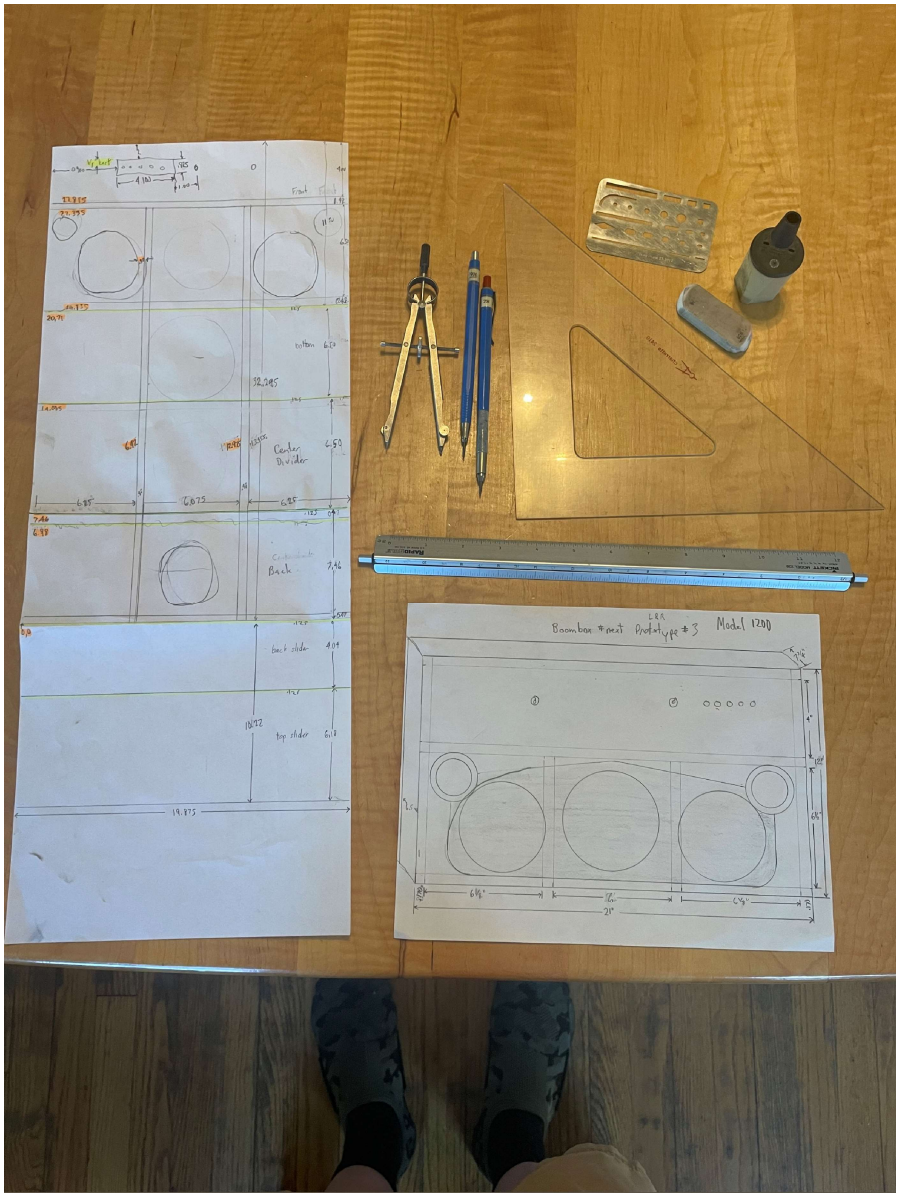
• Damgoo 2.1 amp	$33	amazon.com/gp/product/B089KT3FG9
• 24v x 5A power supply	$21	Amazon
• Raspberry Pi 4 w/4 GB	$70	**BLOCKER**: "Unobtanium" due to global chip shortage
• 5v power supply & heat sinks		Comes with above kit
• RasPi inline switch	$7	canakit.com/raspberry-pi-4-on-off-power-switch.html
• Samsung PRO 64G SD card	$13	https://tinyurl.com/ymat2es4
• HiFiBerry DAC+	$45	hifiberry.com/shop/boards/hifiberry-dac2-pro/
• 3.5mm Y cable	$2	PE: 240-1026
• RCA female/3.5mm male	$6	amazon.com/gp/product/B000I963XQ
• USB microphone	$14	amazon.com/gp/product/B0779PKLV9
• Electrical box/outlet/cover	$8?	Hardware store
• 4 rubber feet	$10	PE: 260-7508
• Handle	$6	PE: 262-314
• 2 3.5mm x 1 ft jacks	$7	Amazon
Subtotal:	$**254**	

Total costs

- **Total cost** (without wood)
  - **$375 - $450**

Components for two
Boomboxes
Model 1500

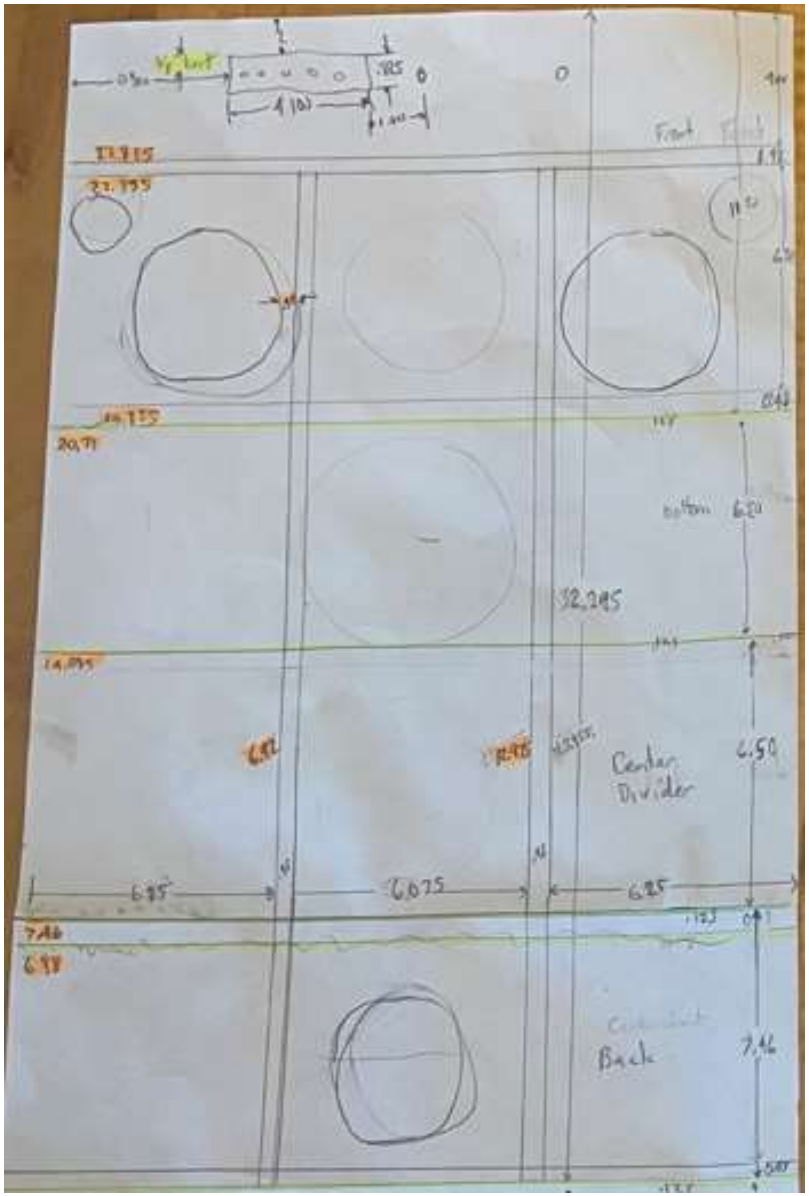
4 - Make drawings



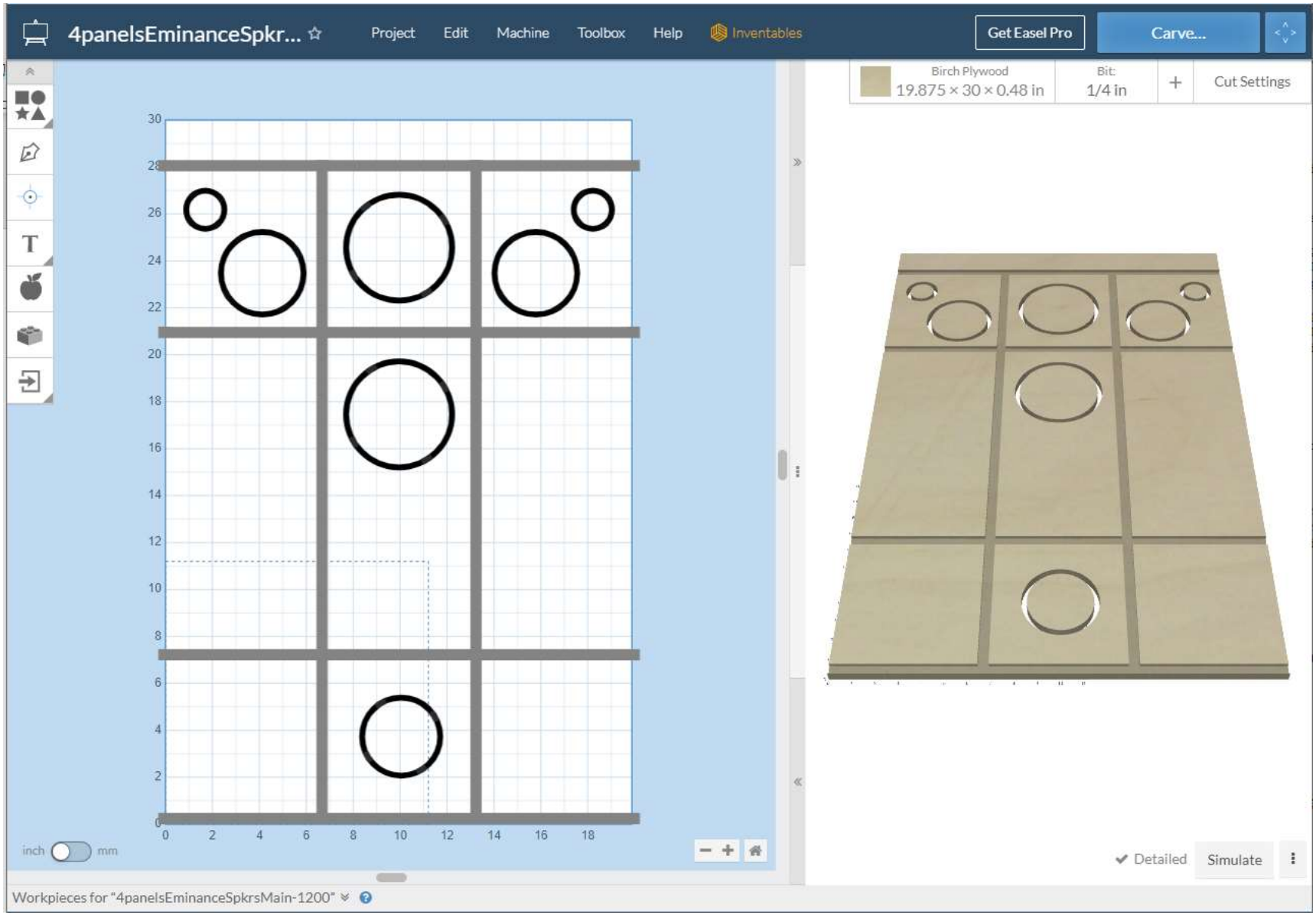
5 - Create G-code for CNC

- Easel.com is recommended (free, but priced for some function)
- Browser-based, projects “in the cloud”
- Download G-code/zip file

Drawing with X & Y values



Easel Gcode for Model 1200

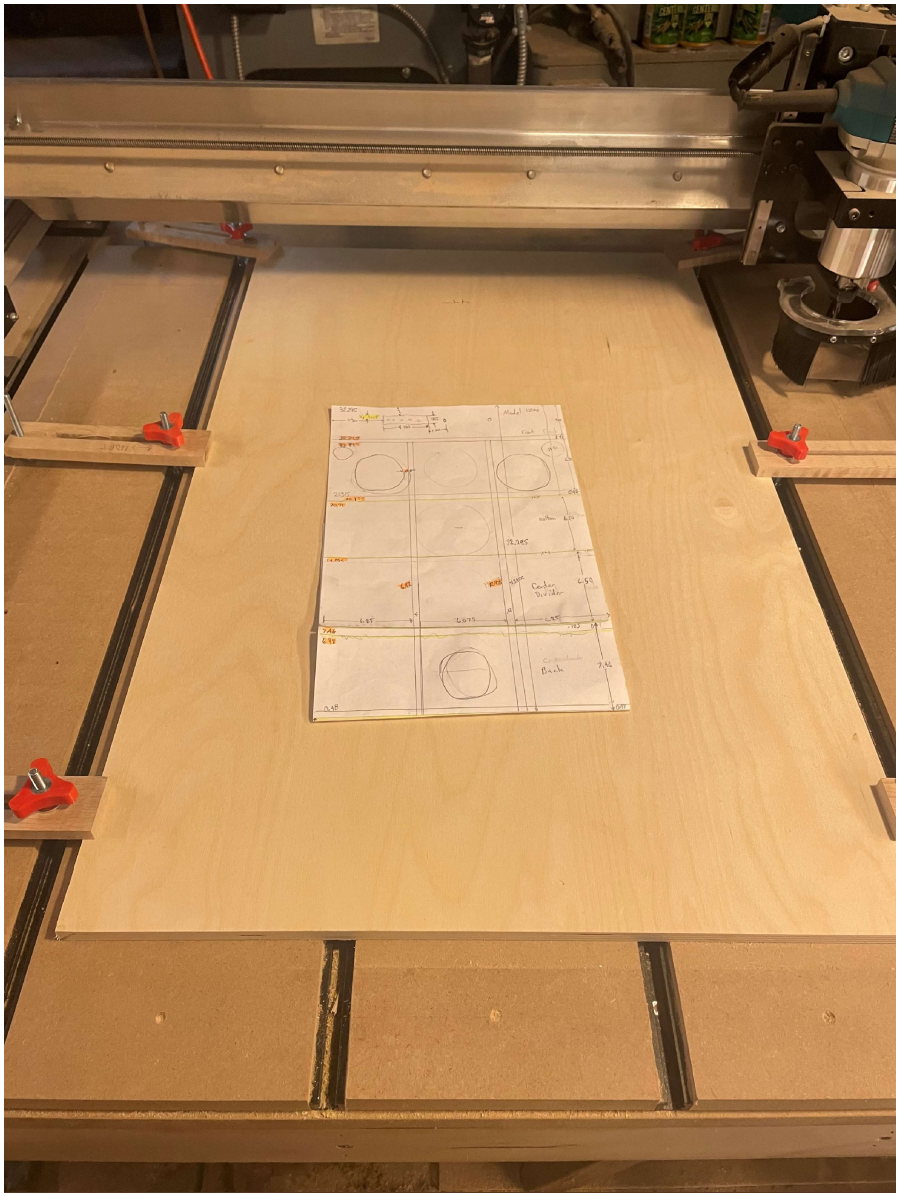


6 - Cut main piece on CNC

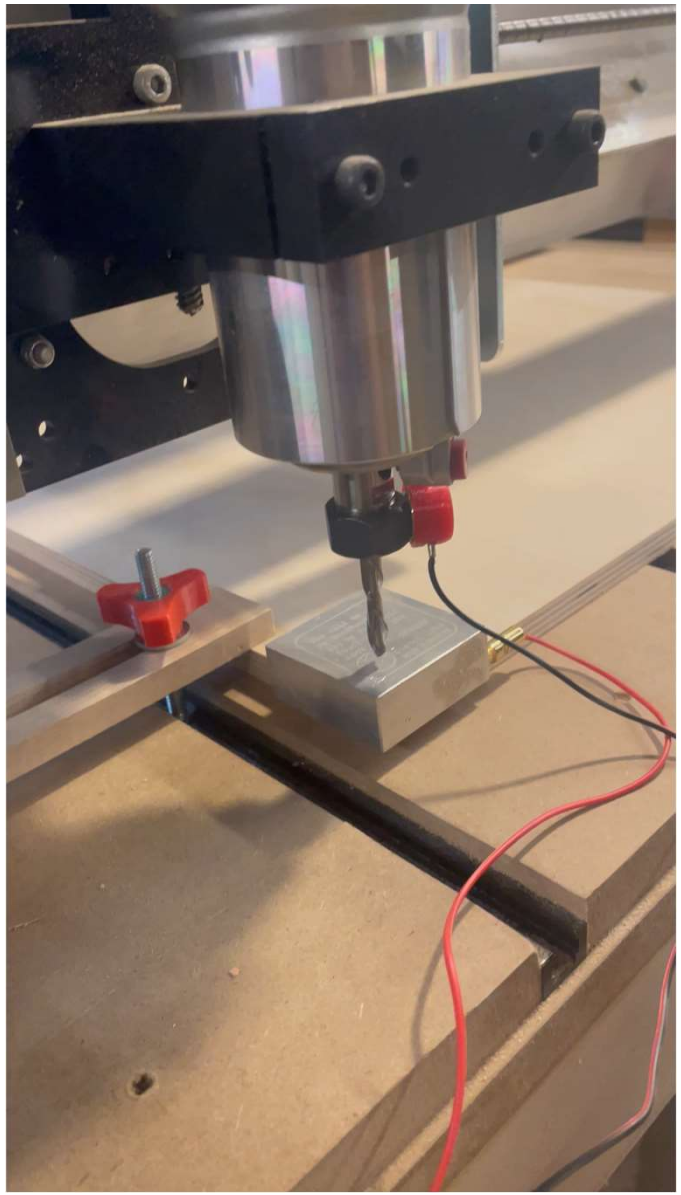
- Main piece is front, bottom, lower back and inside divider panels
- Cut Baltic birch (~ 60” x 60”) into thirds (60” x 20”)
- Cut main piece for four panels to
  - 19.875” x 32.295” (model 1200)
- Square it
- Clamp it
- Zero X, Y and Z
- Load gcode into gsender
- Cut it! (and stand by the big red button ☺)
- Sand the good side

Square the workpiece

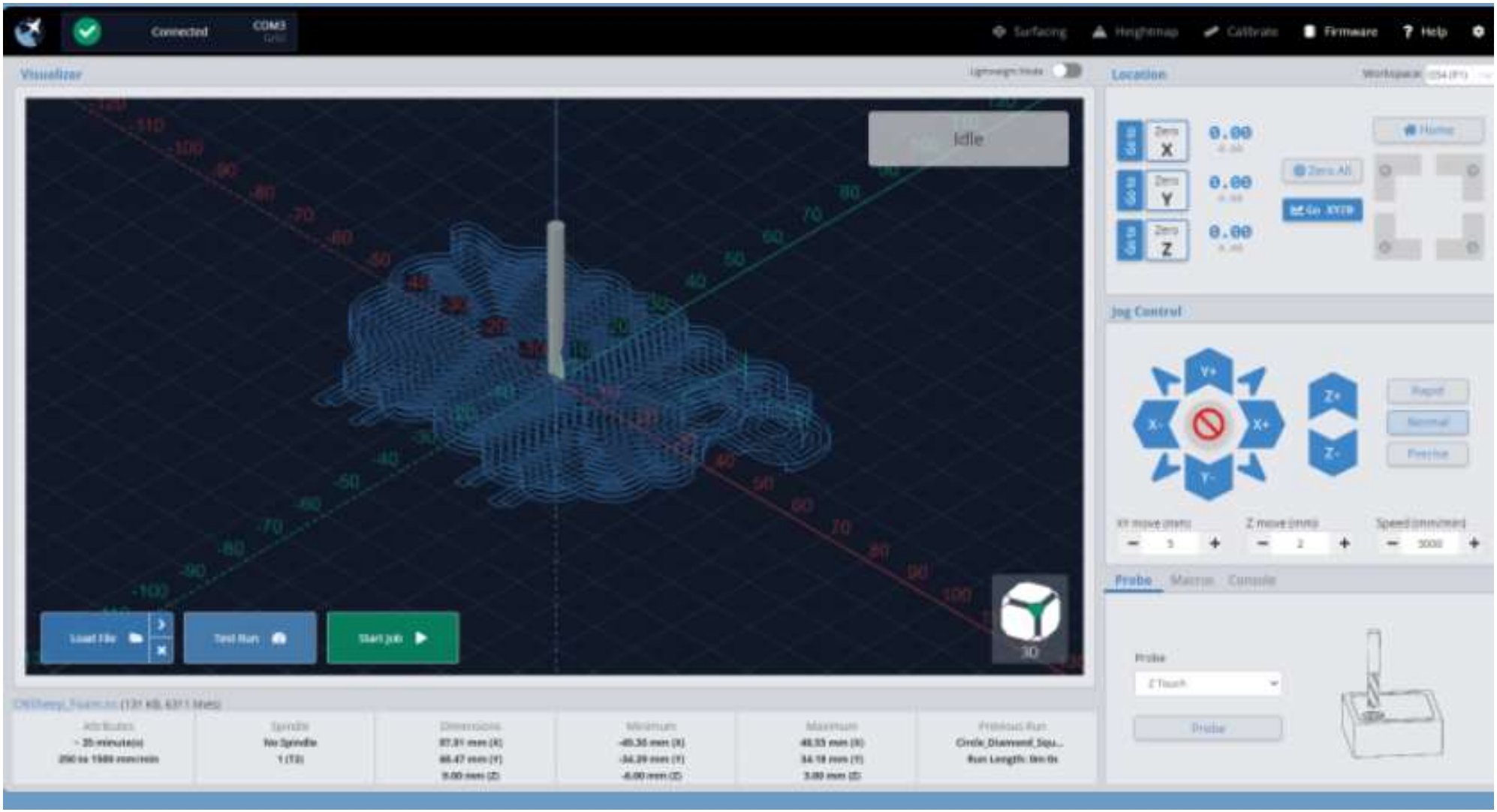
Clamp the workpiece



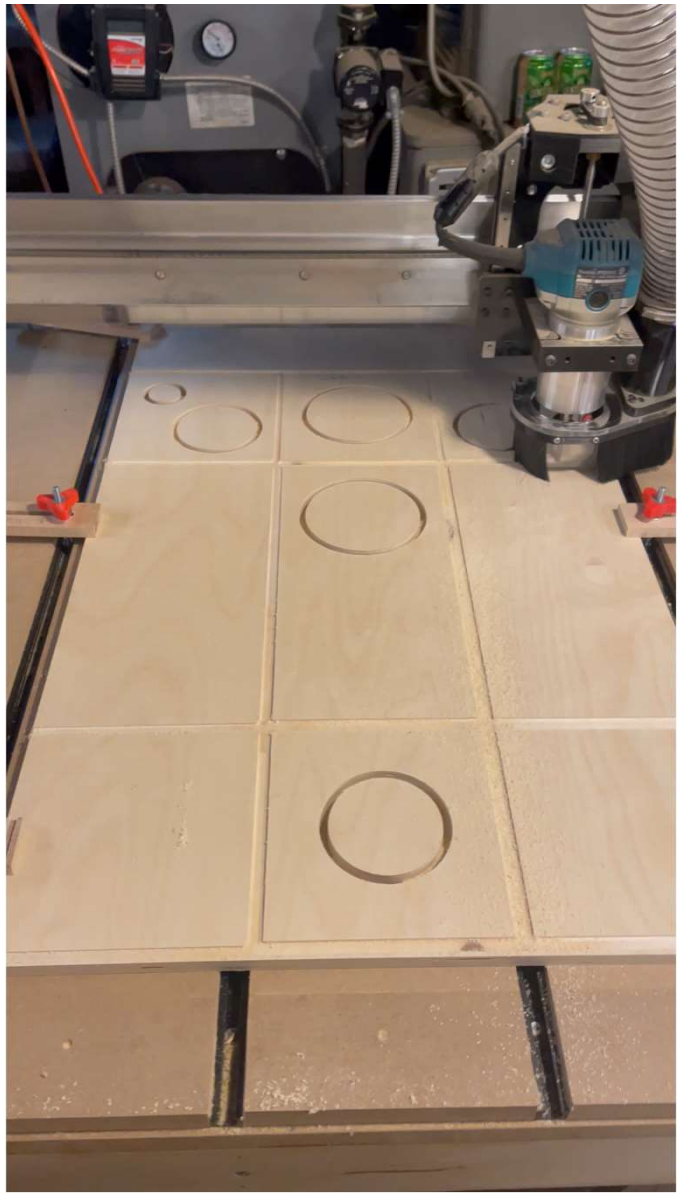
Zero X, Y and Z function



Load gcode into gsender

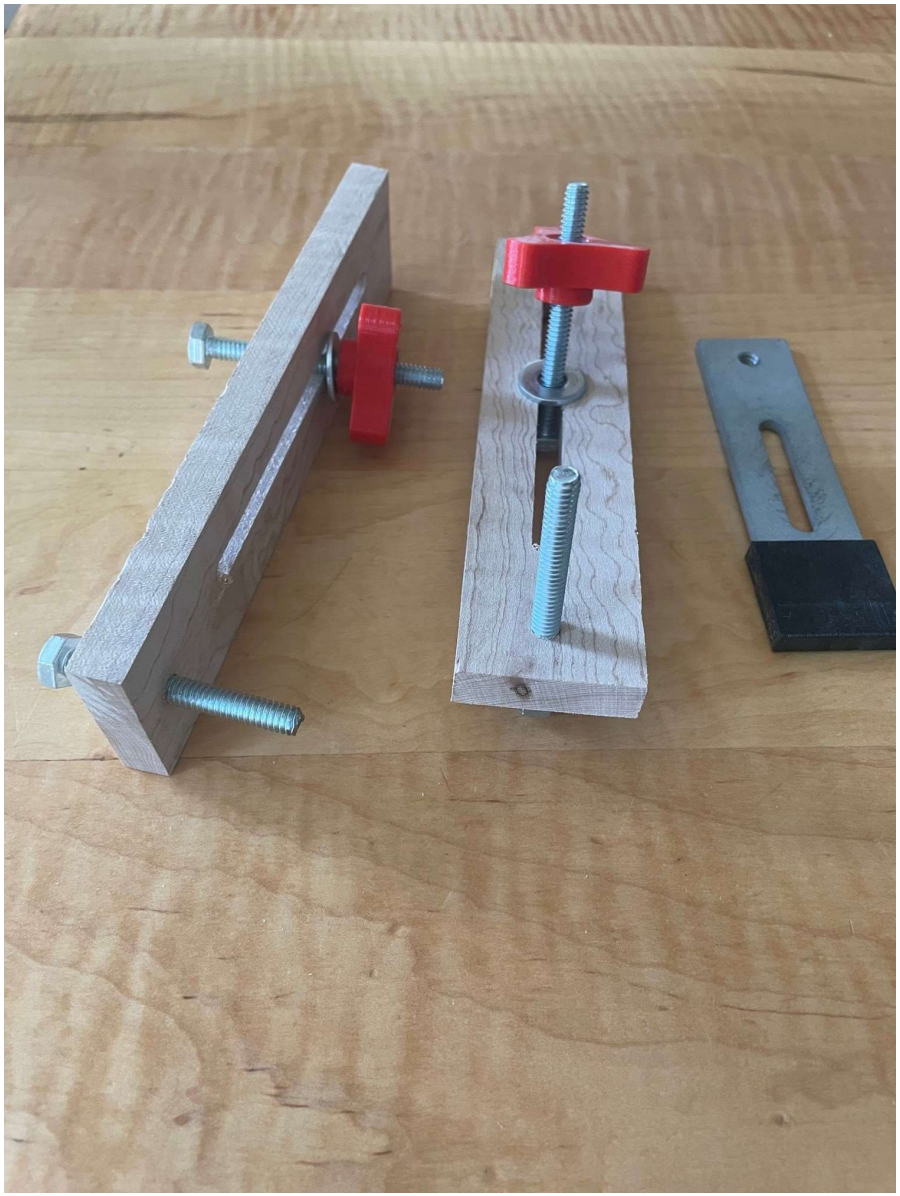


CNC machine in action



Metal and wood Hold-downs

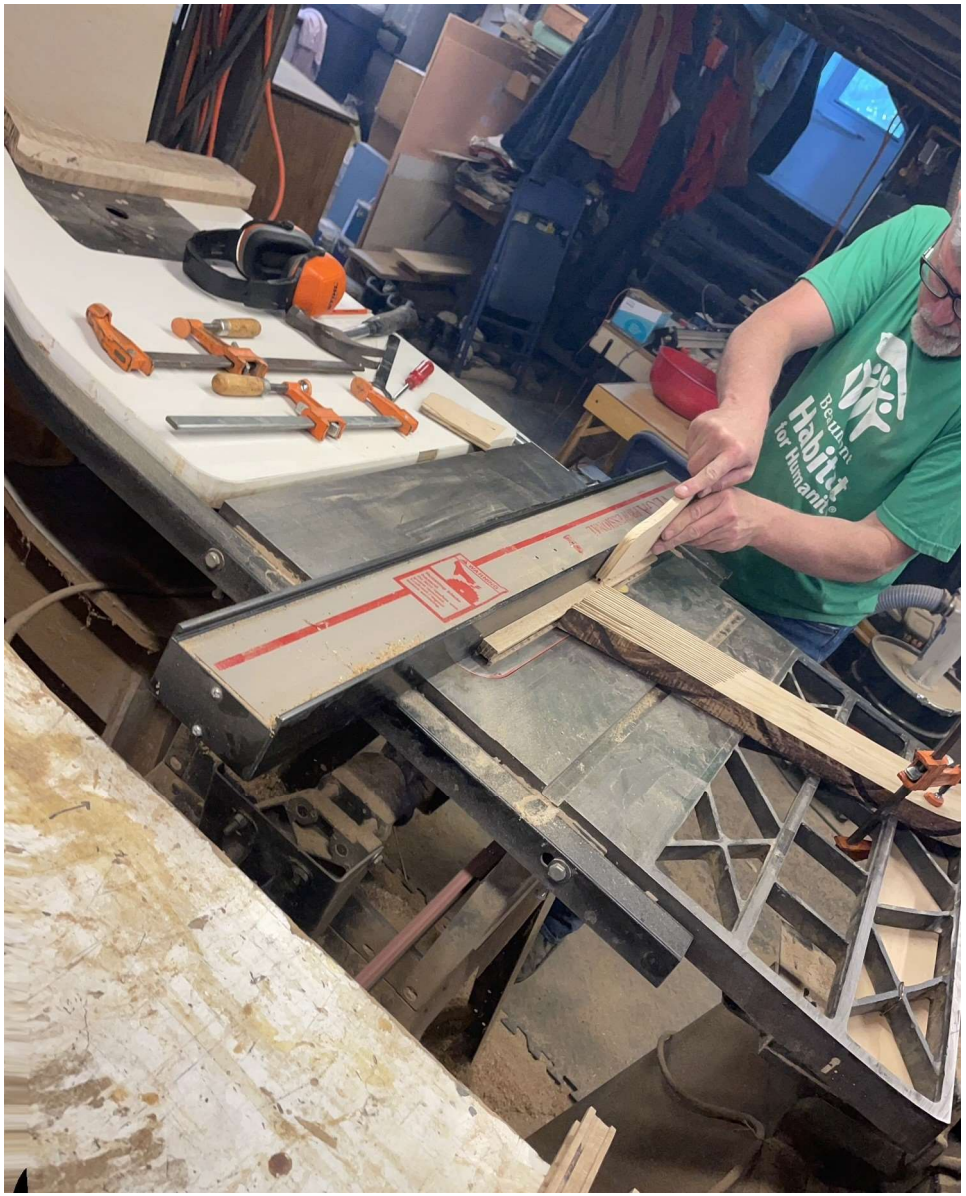
with some near misses



Good side after CNC cut

7 - Cut out all panels and moldings

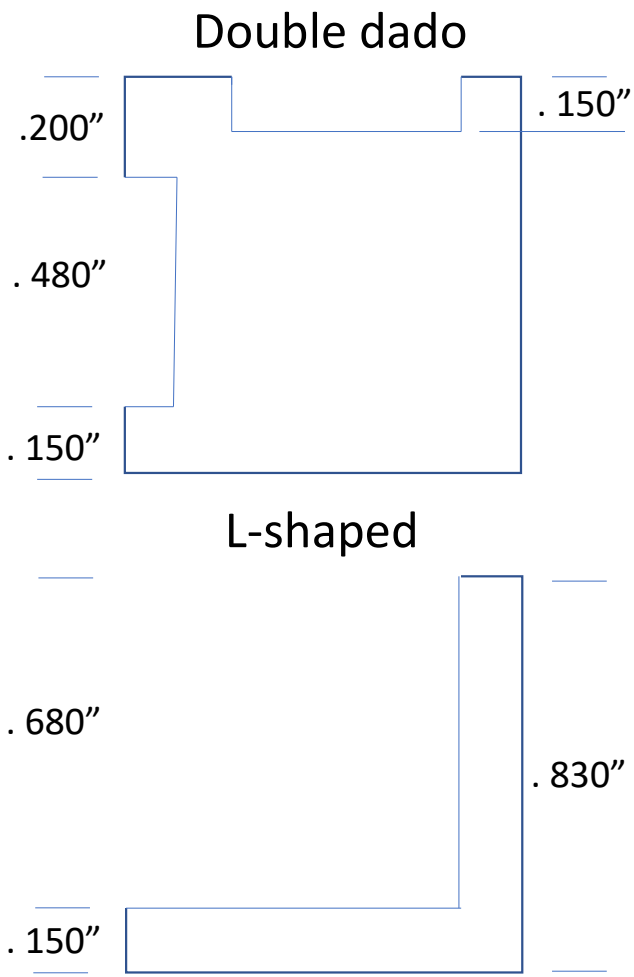
- Cut panels on table saw
- Tried using the CNC machine to cut moldings
- Cut moldings on table saw
  - Dado blade for double dados
  - Saw blade for L-shaped moldings
- Sand inside two sides on all moldings


Habitat
for Humanity


INDUSTRIAL
MAX
RPM
4500

Cut list

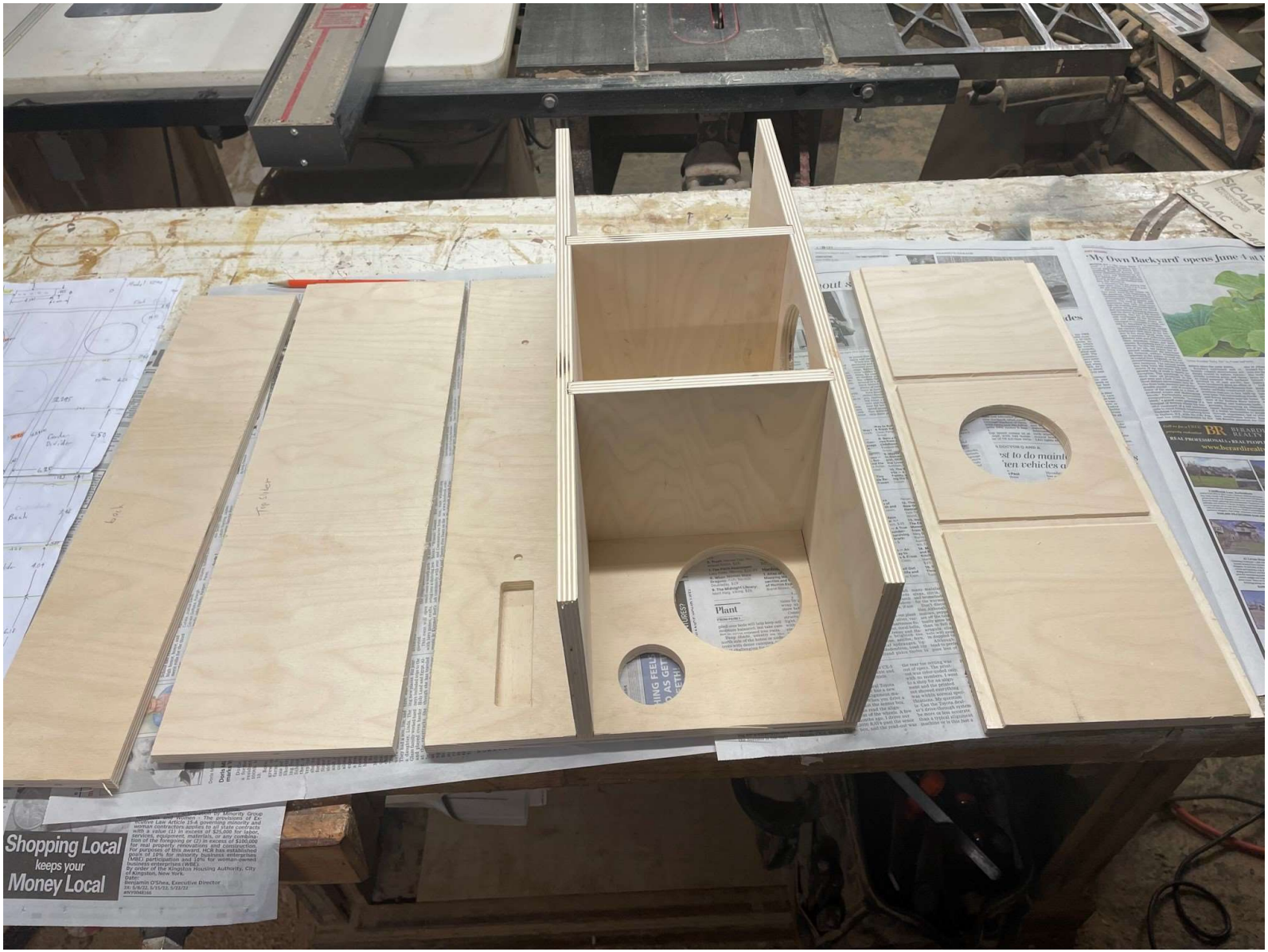
- .480” Baltic birch panels (10)
  - 1: 11.500” x 19.875” Front
  - 2: 6.500” x 19.875” Bottom, inside top divider
  - 1: 7.375” x 19.875” Back bottom
  - 1: 6.250” x 19.875” Top sliding panel
  - 1: 4.125” x 19.875” Back sliding panel
  - 2: 6.500” x 6.875” Small bottom dividers
  - 2: 6.200” x 10.500” Sides
- .830” x .830” hardwood moldings (12)
  - 4: 11.875” Dbl dado Sides, uprights
  - 4: 7.500” Dbl dado Sides, top and bottom
  - 4: 20.000” L-shaped Width, top and bottom



left side


STIHL

8 panels cut
& dry-fitted



Top and bottom moldings

Side moldings
and cam clamps

8 - Label panels

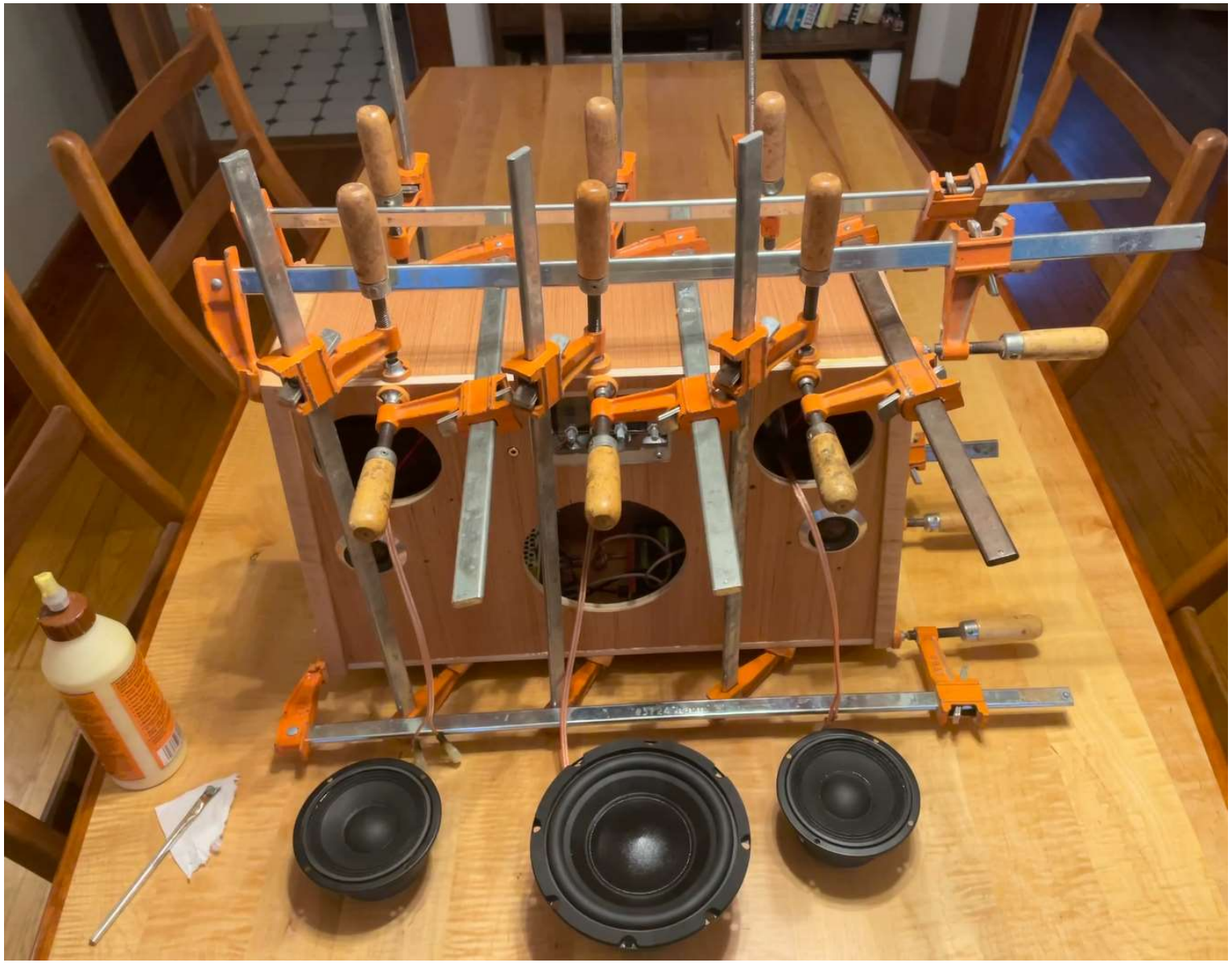
- How to label the box (on, off, bass, treble, etc.) ?
- Bought a FoxAlien LE4040 laser engraver
  - Cannot get it to work from a Linux PC ☹
  - Would not recommend FoxAlien
- Still using Dymo labels (as few as possible)

Prototype #13

Labels

9 - Glue sides then enclosure

- Glue sides w/picture frame clamps and let dry
- Run CNC job on two sides
- Glue and clamp bottom and three inside dividers loosely
- Glue everything together

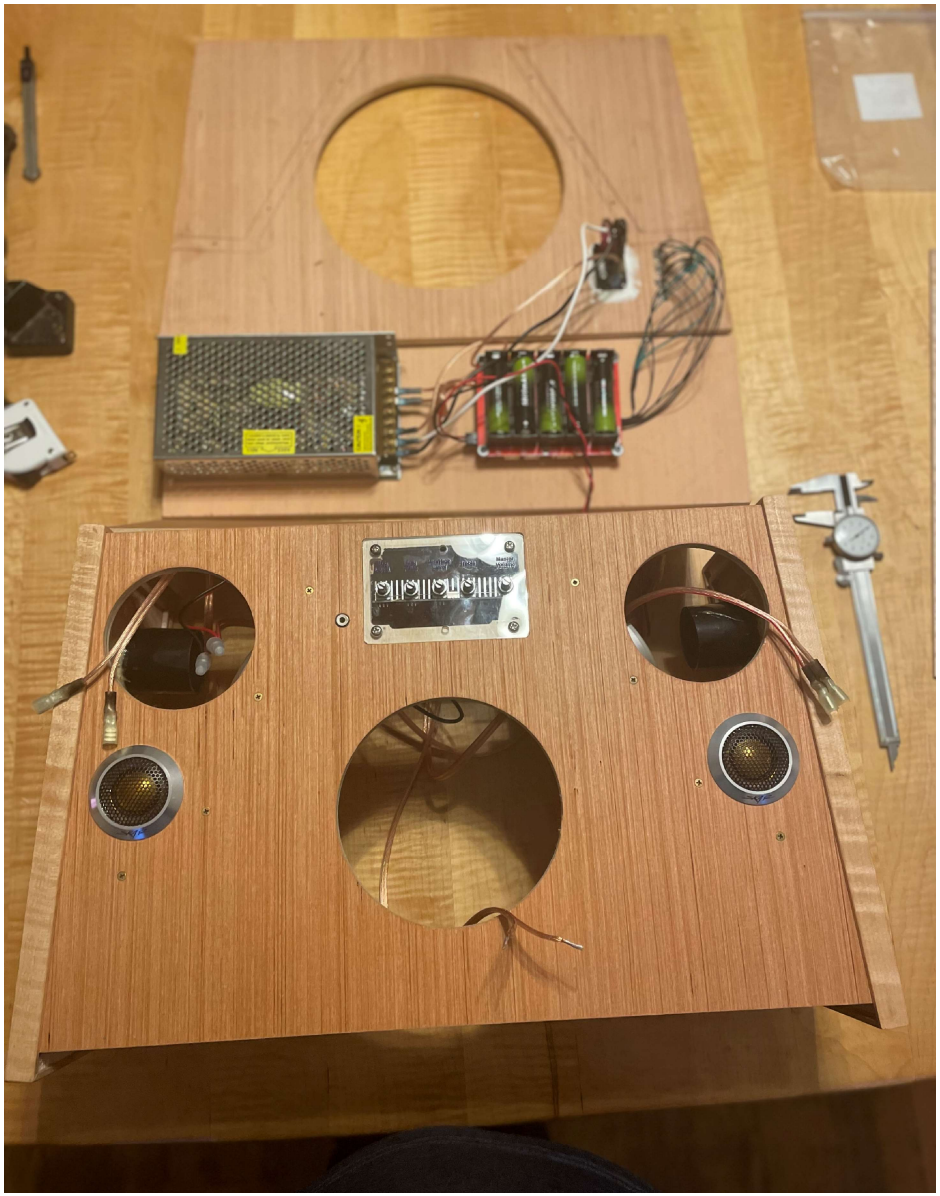


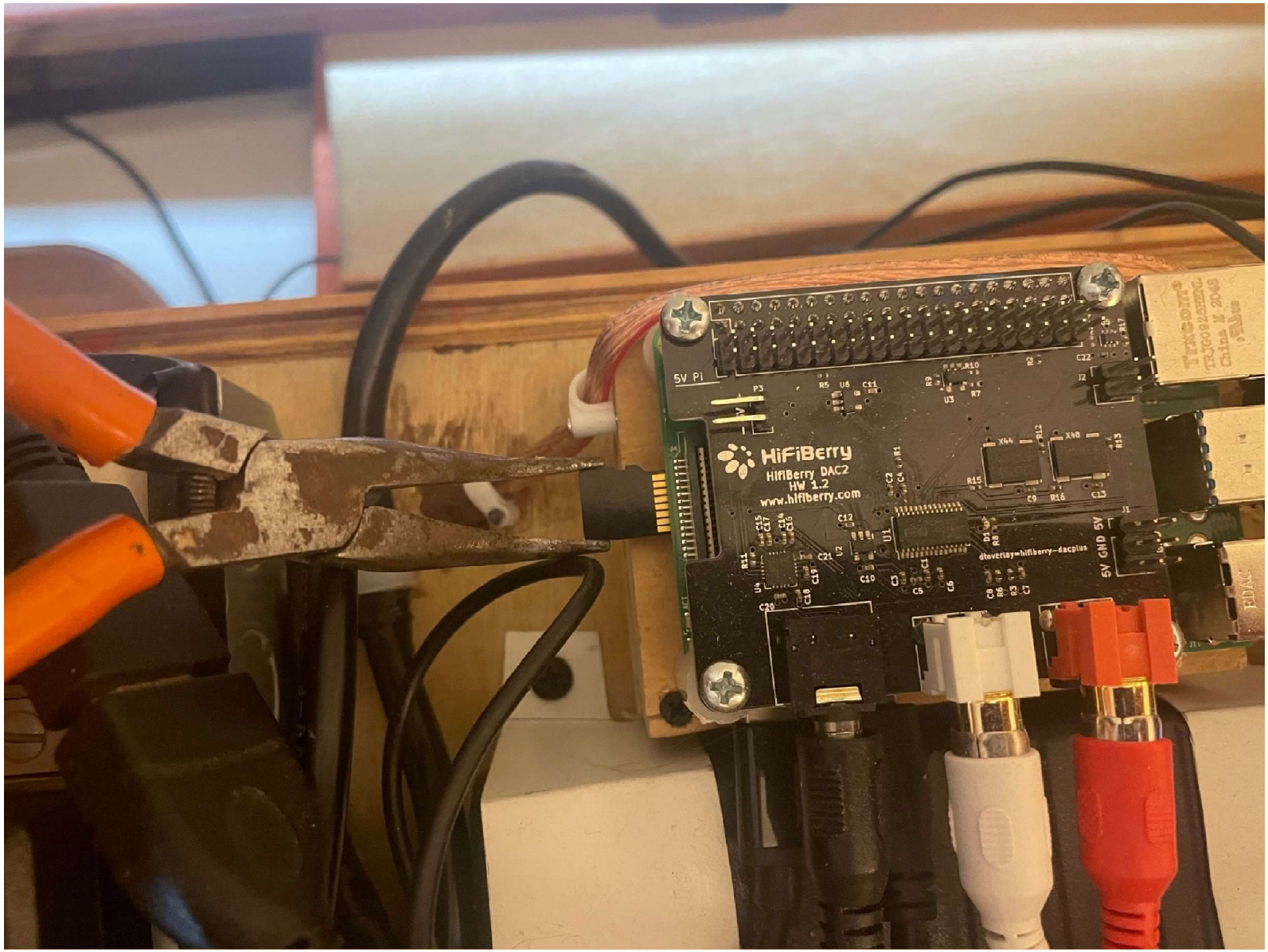
10 - Finish

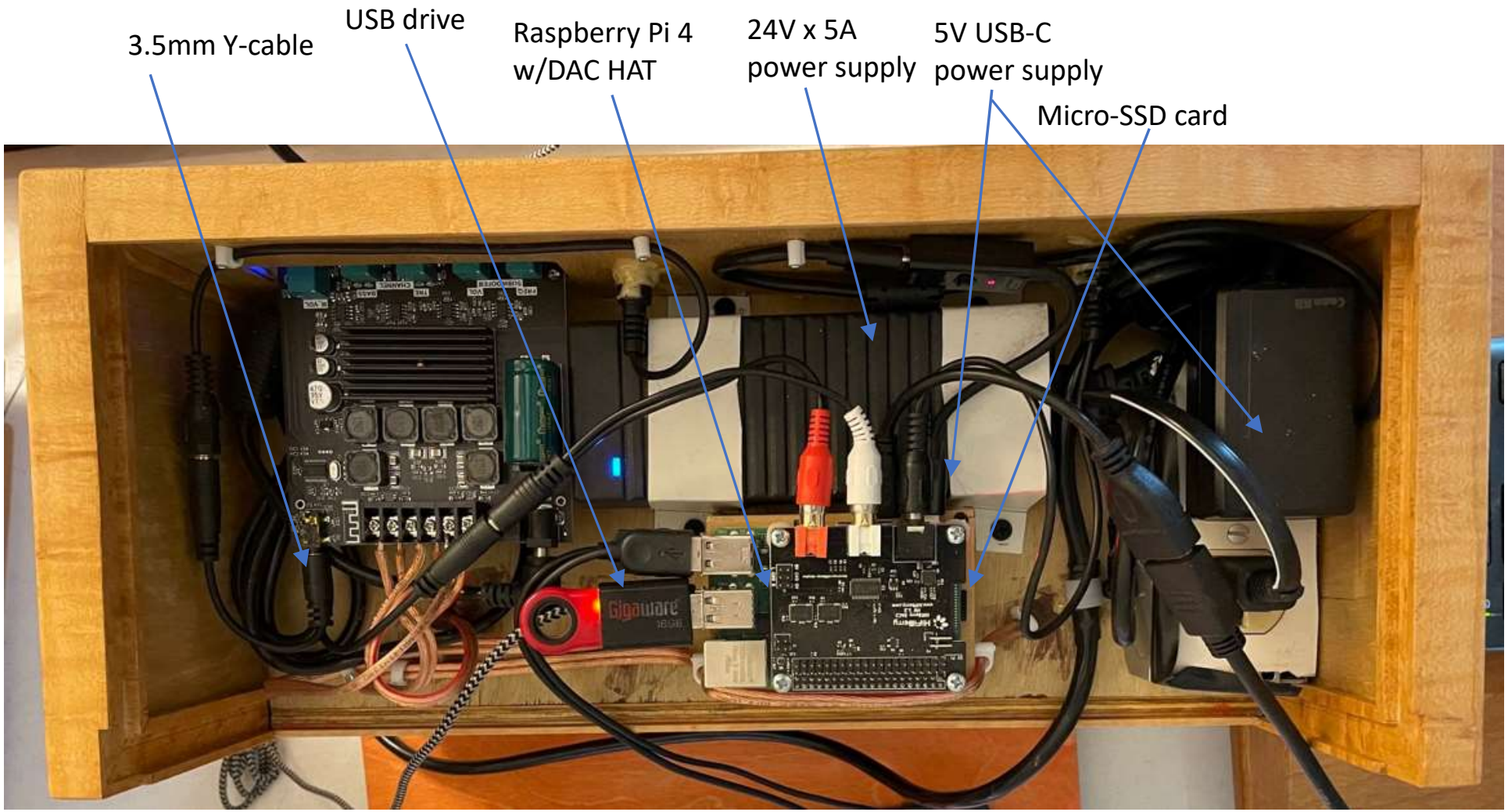
- First coat is natural (no pigment) or colored stain
  - Brush on, wipe off
  - Recommend Minwax
- Second coat is “the mixture”
  - Equal parts polyurethane/boiled linseed oil/turpentine
  - Brush on, wipe off
  - Can be done right after first coat
- After assembly, last coat is “Feed-n-Wax” (by Howard)
  - Bees wax and orange oil
  - Apply with rag
  - Avoid components

11 - Install components and computer

- Leave enough wire
- Allow access to components
- Attach wiring to walls of enclosure where possible




5V Pi
HiFiBerry
HiFiBerry DAC2
HW 1.2
www.hifiberry.com
dtoverlay=hifiberry-dacplus
5V GND 5V


3.5mm Y-cable
USB drive
Raspberry Pi 4
w/DAC HAT
24V x 5A
power supply
5V USB-C
power supply
Micro-SSD card
Gigaware

12 - Install Linux onto SoC

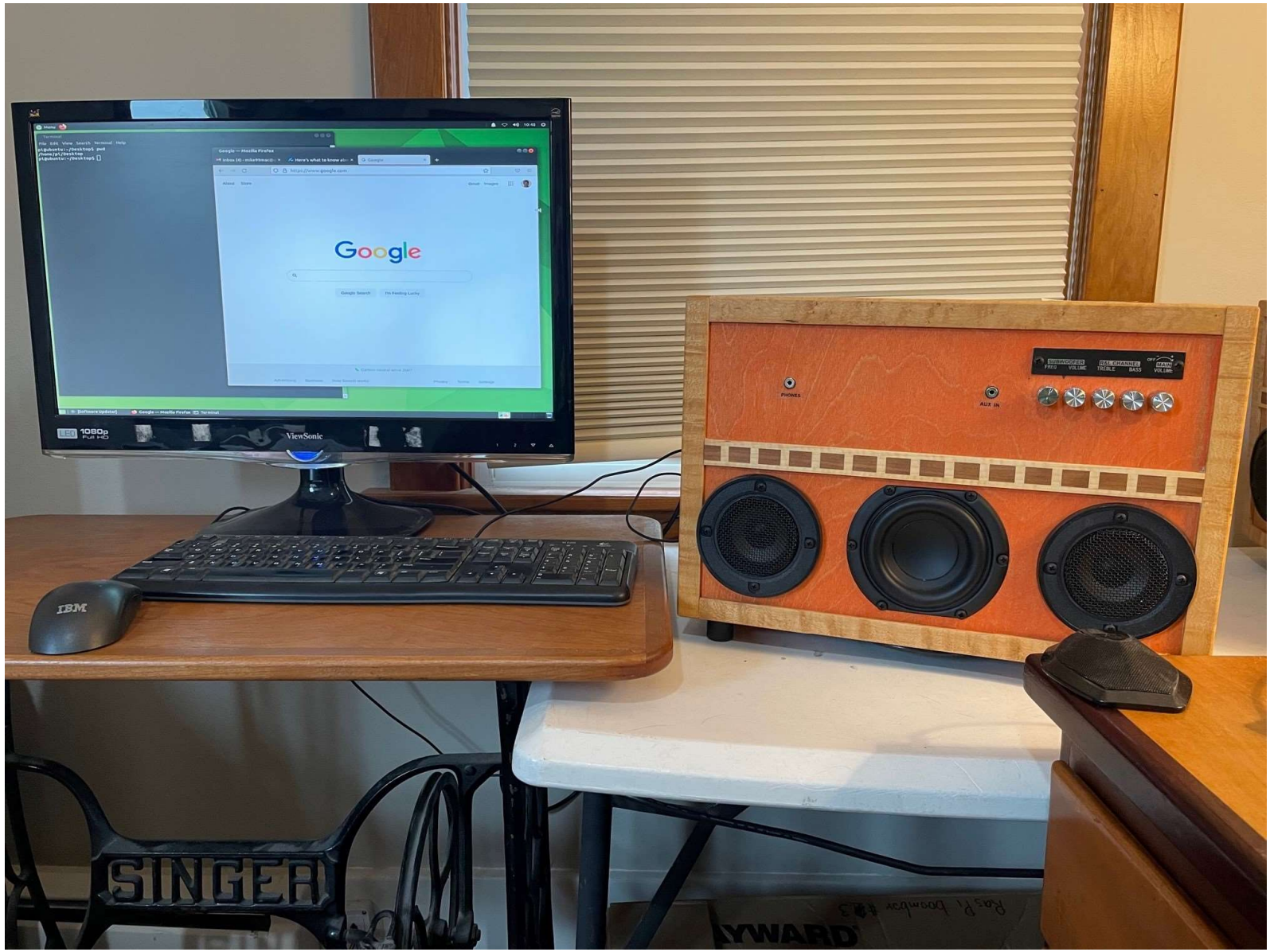
NOTE: Details are in “The Smart Boombox Cookbook”:
https://smartboomboxes.com/wp-content/uploads/2022/01/smartBoombox.pdf

- Download Ubuntu 22.04 LTS Desktop (or another distro)
  - `https://ubuntu.com/download/raspberry-pi`
- Copy to micro-SSD card: Linux rpi-imager
- Plug micro-SSD card in RasPi
- Connect monitor, keyboard and mouse
- Boot the RasPi
- Login as ubuntu/ubuntu and change password
- Get onto Wi-Fi
- Install openssh-server & start it
- Either SSH in or start a terminal session
- Create user and group pi

Install Linux onto SoC (cont'd)

- Prevent automatic upgrades, then upgrade system
- Set time zone
- Turn off default sound card, turn on DAC HAT
- Turn off Linux Bluetooth
- Test microphone and speakers
- Set up music on USB/network drive (if you have one)

Smart Boombox desktop



13 - Install and configure Minimy

- Download Minimy

```
$ git clone https://github.com/mike99mac/minimy-mike99mac
```

- Install

```
$ ./install/linux_install.sh
```

- Configure

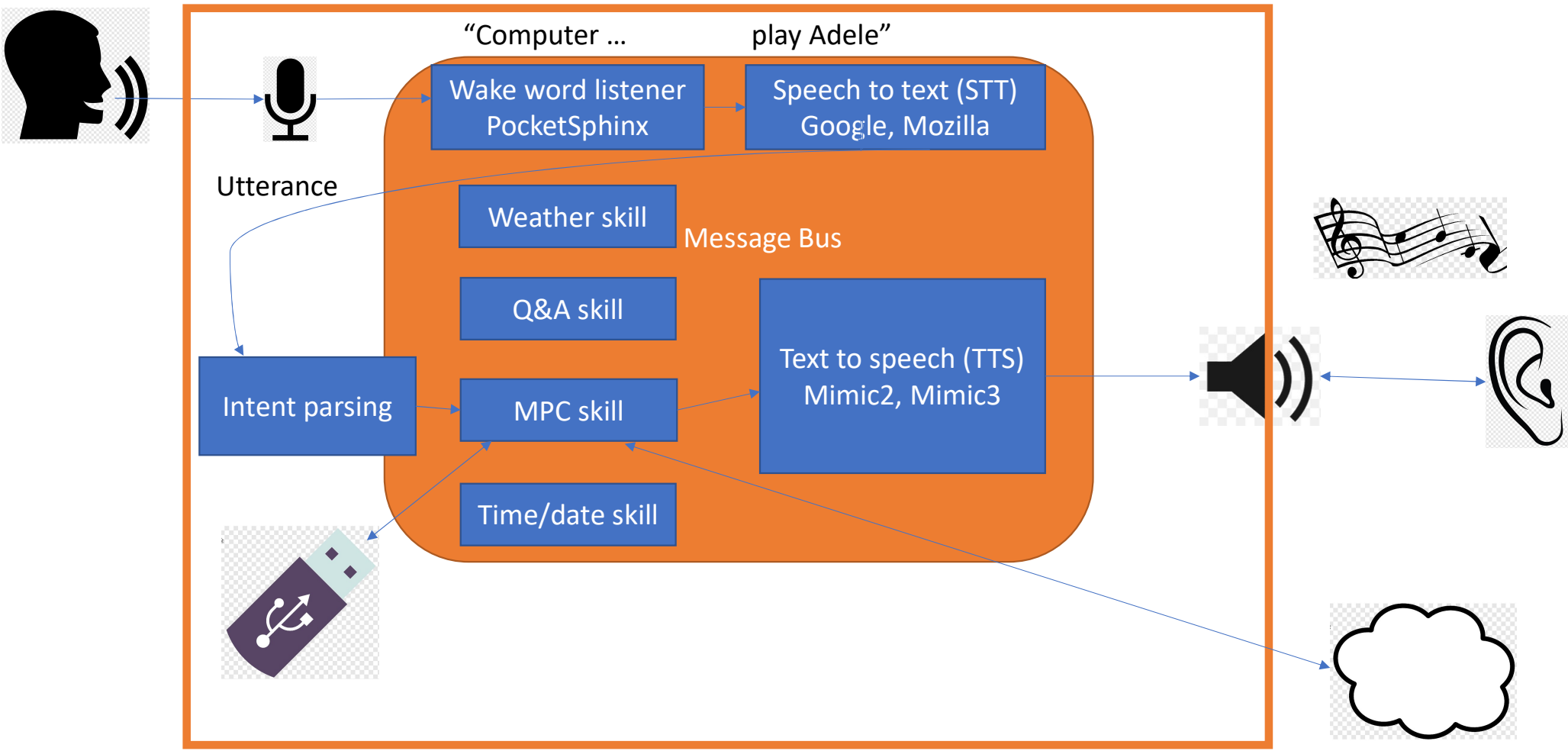
```
$ ./mmconfig.py sa
```

- Set to boot with system

```
$ systemctl enable minimy
```

- Start using your Smart Boombox!!! ☺

Mycroft/Minimy block diagram



Coding philosophy

- Mix of OO & Old-fashioned “modular top down” is OK
- 3 P’s: Patience, Persistence, Pit-bull instinct
- Comment heavily – column 44
- At least 1 debug line per function w/`ClassName.Function`()
- 2-space indents, no tabs, avoid blank lines
- Write new function if:
  - Beyond a page
  - 3 levels of indentation
- Build regression tests
- Do what the f*** you want to

Music_info object

```
#
#            DO WHAT THE FUCK YOU WANT TO PUBLIC LICENSE
#   TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION
#
#  0. You just DO WHAT THE FUCK YOU WANT TO.
#
class Music_info:
  match_type = ""                    # album, artist, song, next or prev
  mesg_file = ""                     # if mycroft has to speak first
  mesg_info = {}                     # values to plug in
  tracks_or_urls = []                # list of music files or radio URLs to play
  def __init__(self, match_type, mesg_file, mesg_info, tracks_or_urls):
    self.match_type = match_type
    self.mesg_file = mesg_file
    self.mesg_info = mesg_info
    self.tracks_or_urls = tracks_or_urls
```

Intent parsing in mpc_skill

```
song_words = ["track", "song", "title", "album", "record", "artist", "band" ]
if "internet radio" in sentence:
  request_type = "radio"
elif "internet" in sentence:
  request_type = "internet"
elif any([x in sentence for x in song_words]):
  request_type = "music"
elif "radio" in sentence:
  request_type = "radio"
elif "n p r" in sentence or "news" in sentence or "m p r" in sentence: # 'npr
  request_type = "news"
else:
  request_type = "music"
```

```
match request_type:
  case "music":                         # if not found in library, search internet
    self.music_info = self.mpc_client.search_library(sentence)
    self.log.debug(f"MpcSkill.get_media_confidence() match_type = {self.music_info.match_type}")
    if self.music_info.match_type == "none": # music not found in library
      self.sentence = sentence
      self.log.debug(f"MpcSkill.get_media_confidence(): not found in library - searching Internet")
      mesg_file = "searching_internet"
      mesg_info = {"sentence": sentence}
      self.speak_lang(self.skill_base_dir, mesg_file, mesg_info, self.fallback_internet) # tell user "searching internet"
  case "radio":
    self.music_info = self.mpc_client.parse_radio(sentence)
  case "internet":
    self.music_info = self.mpc_client.search_internet(sentence)
  case "news":
    self.music_info = self.mpc_client.search_news(sentence)
if self.music_info.tracks_or_urls != None: # no error
  self.log.debug("MpcSkill.get_media_confidence(): found tracks or URLs")
else:                                   # error encountered
  self.log.debug(f"MpcSkill.get_media_confidence() did not find music: mesg_file = {self.music_info.mesg_file} mesg_info
confidence = 100                        # always return 100%
return {'confidence':confidence, 'correlator':0, 'sentence':sentence, 'url':self.url}
```

Demo

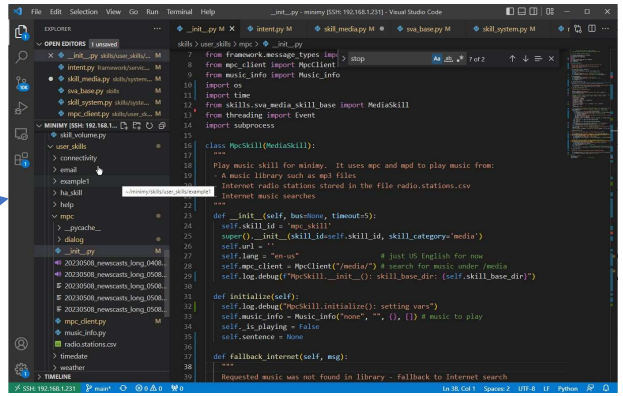
Writing music skills

- Mycroft Emby/Jellyfin
  - Difficult API
- Mycroft mpc/mpd
  - Easy to call mpc line command
  - ~38k LoC
- OVOS mpc/mpd
  - Could not get running ~3 weeks
  - ~18k Loc
- Minimy mpc/mpd
  - Got running in one day
  - “Less is more”
  - ~10k LoC

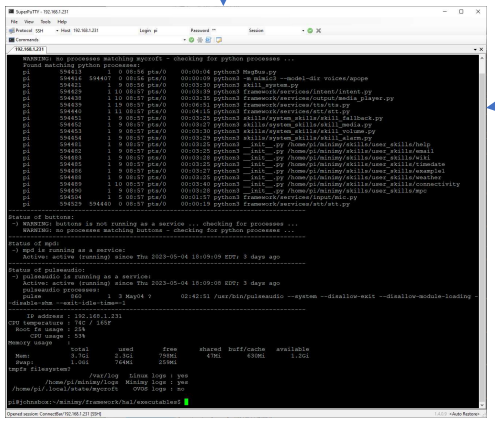
Development environment

VS code on Windows laptop

ssh



ssh



/home/pi/minimy/

diff, cp

git commit/push

/home/pi/minimy-mike99mac/

ChatGPT resources

- https://github.com/NeonGeckoCom/skill-fallback_llm

Converse with an LLM and enable LLM responses when Neon doesn't have a better response.

To send a single query to an LLM, you can ask Neon to "ask Chat GPT ". To start conversing with an LLM, ask to "talk to Chat GPT" and have all of your input sent to an LLM until you say goodbye or stop talking for awhile.

Enable fallback behavior by asking to "enable LLM fallback skill" or disable it by asking to "disable LLM fallback".

ChatGPT – trying to enable

```
2023-05-15 05:24:22.236 - skills - neon_core.skills.intent:handle_utterance:222 - INFO - ['enable llm fall back skill', 'enable llm fallback skill', 'enable ll.m fallback skill']

2023-05-15 05:24:22.332 - skills - mycroft.skills.intent_services.commonqa_service:handle_question:119 - INFO - Searching for enable llm fall back skill

2023-05-15 05:24:22.377 - skills - skill_caffeinewiz:CQS_match_query_phrase:206 - INFO - <mycroft_bus_client.message.Message object at 0x7f042d7c50>

2023-05-15 05:24:22.401 - skills - skill_fallback_wolfram_alpha:CQS_match_query_phrase:163 - INFO - enable llm fall back skill

2023-05-15 05:24:22.404 - skills - skill_caffeinewiz:_clean_drink_name:502 - INFO - enable llm fall back skill

2023-05-15 05:24:23.319 - skills - mycroft.skills.intent_services.commonqa_service:_query_timeout:173 - INFO - Timeout occurred check responses

2023-05-15 05:24:23.586 - skills - skill_fallback_unknown:handle_fallback:79 - INFO - Unknown Fallback Checking for Neon!!!

2023-05-15 05:24:23.589 - skills - ovos_workshop.skills.fallback:handler:141 - ERROR - Exception in fallback.

Traceback (most recent call last):

  File "/home/neon/venv/lib/python3.7/site-packages/ovos_workshop/skills/fallback.py", line 131, in handler

    if handler(message):

  File "/home/neon/venv/lib/python3.7/site-packages/ovos_workshop/skills/fallback.py", line 219, in wrapper

    if handler(*args, **kwargs):

  File "/home/neon/venv/lib/python3.7/site-packages/skill_fallback_unknown/__init__.py", line 83, in handle_fallback

    if not (self.neon_in_request(message) or

AttributeError: 'UnknownSkill' object has no attribute 'neon_in_request'
```

ChatGPT – trying to use

```
2023-05-15 05:16:50.089 - skills - skill_caffeinewiz:_clean_drink_name:502 - INFO - connect to chat g p t

2023-05-15 05:16:50.287 - skills - mycroft.skills.intent_services.commonqa_service:_query_timeout:173 - INFO - Timeout occurred check responses

2023-05-15 05:16:50.713 - skills - skill_fallback_unknown:handle_fallback:79 - INFO - Unknown Fallback Checking for Neon!!!

2023-05-15 05:16:50.720 - skills - ovos_workshop.skills.fallback:handler:141 - ERROR - Exception in fallback.

Traceback (most recent call last):

  File "/home/neon/venv/lib/python3.7/site-packages/ovos_workshop/skills/fallback.py", line 131, in handler

    if handler(message):

  File "/home/neon/venv/lib/python3.7/site-packages/ovos_workshop/skills/fallback.py", line 219, in wrapper

    if handler(*args, **kwargs):

  File "/home/neon/venv/lib/python3.7/site-packages/skill_fallback_unknown/__init__.py", line 83, in handle_fallback

    if not (self.neon_in_request(message) or

AttributeError: 'UnknownSkill' object has no attribute 'neon_in_request'
```

Fair compensation to artists

Platform	Royalties/stream	Streams to earn $1
Napster	$0.019	53 (17x < Deezer)
Tidal Music	$0.01284	78
Apple Music	$0.008	125
Amazon Music	$0.00402	249
Spotify	$0.00318	314
YouTube Music	$0.002	500
Pandora	$0.00133	752
Deezer	$0.0011	909

Summary and resources

- The code: https://github.com/mike99mac/minimy-mike99mac
- Tools: https://github.com/mike99mac/mycroft-tools
- smartboomboxes.com – “A stake in the ground”
- The Smart Boombox cookbook
- My email: mike99mac@gmail.com
- Lines of Python code:

```
mycroft-core     : 38074
ovos-core        : 18067
minimy-mike99mac: 10107
```

Questions

- ???
- Does anybody want partner?
  - Mike MacIsaac
  - mike99mac@gmail.com
  - 862-308-5089