Foghorn Log:

Apr 4:

* Did some initial research, I have some components that seem promising
  + I need to finish component selection before I can order
* Do I need an amplifier?
  + If there are relatively easy, cheap options, I’ll do this
* I want input from either ESP32 or microphone, is it easier to do this before or after the DAC?
  + How does I2S work? Can I share lines, and will having 3 devices on the line cause problems for the unused device?
  + It seems like I probably want to do this before DAC if possible, to avoid issues with analog levels/amplification

Apr 7:

* I ended up buying some parts, the Adafruit FM transmitter, a DAC, and a microphone
  + ESP32 can handle I2S audio, but I can’t share a line because it would have to be master and slave at once (only one channel)
* No amplifier at the moment, I may have to look at amplifying options later, but that will be a stretch goal
* Going to get through finals and wait for parts to arrive, then we roll!

May 8:

* Parts are all here, and I’m free from school, time to get to work
* First step is reviewing hardware
  + I’ll try out just the FM transmitter first with an audio source
* Oopsie, first hardware oversight, I’ll need an aux cable to test this out
  + Will buy
* Initial setup:
  + Arduino UNO on breadboard
  + Wires soldered to relevant FM transmitter pins
  + Phone connected to aux input
* I have all hardware set up
* When running the example code, I seem to have an issue with the I2C connecting
  + Make sure connections are all good
    - Seems all good, but reset and ground are connected when powered, why?
  + Try authentic Arduino
    - Same result
  + Investigate voltages
    - I2C pins seem to be high
    - Reset is 3.3V, why?
  + Very strange that reset (pin 12) and GND are both connected and separated by 3.3V when the thing’s running
  + Review code
    - Seems like a normal connection, I might revisit this, but ok for now
  + Pullup resistors
    - Added 10k resistors, no luck
* Fixed!
  + Connecting the reset pin and ground seems to solve the problem, I’ll have to figure out a software solution to this
    - Added a manual reset with the reset pin
* Radio works well
  + Very clear
  + 10m range with no problems
  + 50m absolute range
* May play with antenna a bit
* Done for today, next step is to integrate more components!

May 10:

* Damn it, I drained a pair of AAAs by leaving the radio receiver on
  + Gonna make it a habit to take batteries out
* Next steps:
  + Test to see if the device works without pullup resistors
  + Figure out why it seems to work intermittently
  + Figure out how it works
  + Set up github project (?)
  + Begin porting to ESP32
* Works fine without pullups, although same issues with inconsistency
* Ok, I have ideas for a solution to fix consistency, but first let’s set up the git so I don’t mess this up
* The antenna capacitance could be an issue
  + I’ve shortened the antenna to about a foot, still same behaviour
  + Tried a few different frequencies, no change
  + Tried adding a delay after reset, this may have improved things slightly, but problem is not solved
  + Aha, adding this delay after the begin function (I2C init?) solves the problem
    - This was because of a ramp-up time that the crystal oscillator needs, which explains the close but not quite freqs I was getting sometimes

May 11:

* Focus for today is transferring to ESP32 hardware
* First step is to see if I can get it working using raw I2C commands, that way I can transfer easily without worrying as much about hardware
  + Will start this by reading the FM transmitter programming datasheet
* Looks like I might not need a DAC after all, it has digital audio input options including I2S
* Wait a second, what if I can just use the existing library?
  + I have decided to not do that
  + Although possible, it’s going to be tricky to port, and I’d rather deal with low level I2C command sending trouble than library trouble
* Switched to using VS Code and PlatformIO for this
  + Open serial monitor using command palette
* Yay, got it set up so that I can power it up using a few of my own commands (but mostly still library)
* I think this serves as a good proof of concept, I’m going to make the leap to ESP32
* First step is setting up the build environment, this will require a lot of transfer from the old code
* Build env set up
* Added I2C and GPIO init functions
* Next step is to begin the long and arduous process of initializing using I2C

May 13:

* I’ve just tried getting the part number off the FM transmitter with I2C and Arduino, this will be a good first step for ESP32
* Now to do the hardware wiring
* No FUCKING way it worked first try
* Now to reorganize/format my code so that it’s a bit cleaner before moving forwards
* Alright, on to the grindy part of adding all the commands I want
* We’re balling
  + FM transmission working from ESP32
* Now just gonna chuck it into a task

May 14:

* Sike, I2C error
  + May be hardware, will investigate
  + Fixed, was just a mistake with passing i2c handle
* Decided not to put init in task for now, since it only needs to run once
* Going to look into I2S, ideally I’d like to try I2S transmission from ESP to FM without involving mic first
* Sike, the mic is not I2S, that’s actually nice
* Means I’ll do ADC on ESP32, but always output I2S
* Sike, I can’t use digital with the FM radio, must transmit to DAC first
  + Just took me quite a while to figure out stuff I already knew a month ago
* Note that setting the mute pin high on the DAC mutes the output, this could be a good way to implement push-to-talk
* Just wired up DAC, now computer doesn’t recognize ESP, very concerning, will debug
  + Cause by a solder bridge
  + Yikes, looks like this put the esp into bootloader mode
  + Hmm, seems like the issue is that my new firmware was so aids that it bricked the MCU
* Sequence to unbrick:
  + While holding boot, press reset (GPIO9 floating)
  + Flash the code (will fail)
  + While holding boot, press reset (GPIO9 pulled to high)
  + Flash code
* So it seems pins 12 and 13 are used for the JTAG interface, this is almost certainly causing the issue
* Will resolder to pins 19, 20
* Ok, it kinda works now, but there’s some kinda issue that causes distortion
  + In any case, enough for today

May 15:

* I’m not going to bother fixing the I2S sine wave, since it way be a problem with the wave, not the I2S
* Instead, let’s get the microphone set up
  + After soldering microphone, I get an issue with too much power, gonna try restarting mac
* Beginning to think I’ve fried my esp32, I’m gonna start removing connections, if it is broken I’ll have to do this anyways
* All desoldered and same result
  + I hate this board
* Since board is broken, I need to reevaluate my hardware
* Ok I’m gonna pivot and use ESP32 with SDD1306 LCD display
  + I have two ESP32s
* Gonna put everything together on a breadboard
* Soldering time

May 16:

* Got the FM working and maybe the DAC, but gonna try mic first
* First I’ll connect and just read a stream to see what’s up
  + Mic seems to be reactive at least, now to convert to I2S
* Not sure what the best way of passing data is
* Set up a queue, but should get DAQ done with a timer for consistent sampling freq

May 19:

* First step is to get acquisition on a timer for consistency
* Then I’ll work on using that data (converting and sending on I2S)
* Ok, DAQ is on timer, using 44.1kHz for now, may change
* Now to set up my receiving task that sends to I2S
  + Data consumption done, now just need to send
* I get audio from the DAC, but it’s incomprehensible at low sample rates and my queue overflows at high sample rates
* I have a feeling high data rates are gonna pose some issues

May 20:

* Round two!
  + Got to figure out high freq sampling
  + First, let’s determine exactly who is blocking
* Looks like I can’t sample the ADC in the kHz range (or at least 16kHz)
* What if I connected the mic’s analog out directly to the FM transmitter?
  + Would mean I’d need to control the FM transmitter to control output
  + May run into the same problem later with I2S comms
    - But Bluetooth (the only thing that needs I2S) is a stretch goal anyways
* Fuck yeah baby she’s going!
* Ok so massive simplification
  + DAC is completely dropped
  + Mic output goes directly into FM
* Next steps:
  + Figure out how to turn FM on and off
* I think it’s acceptable if the mic has a button on the output line
  + Means the radio will hear silence even when voice is not being transmitted
  + But this is ok, simplifies things, and may even be a feature
* Wow, now I just need to sort out tuning and the power cycle!
  + Huge progress by cutting fat
* I2S should be kept in, as it could be implemented later
  + Will be saved in a commit
  + Will have to remember to wire it up for I2S (or Bluetooth output of some sort)
* I should try the other gain settings on the mic before wiring everything up
  + Tried high gain, I prefer just regular, it sounds fine
* Got FM tuning IO going, as well as push to talk
* Now for the most feared part, the screen
* I likely have the following screen:
  + SSD1306 controller
  + 128x64 size screen
* Ok, after a lot of poking around, I have a demo going
* Now I just need to integrate this into my main project
* Raaahhhhhhhhhhhhh
  + Got the frequency display working
  + May see if I can get an indicator going for the voice
  + That may be tricky given the way I’ve implemented it (button on line)
* Should add some safeties for the FM limits
* Wow, I think I’m pretty much done!
* Should test plugging a phone on while the mic is running
  + No problem, phone audio overwhelms mic audio
* Let’s check out how the battery board needs to work, then I’ll start CAD!
  + Seems it doesn’t really talk with smart chargers like my laptop or its charger
  + Does seem to work otherwise
* CAD time!
* I’m gonna need some kinda glue to hold all my parts in place

May 21:

* First test print done, gonna disassemble board to test parts
* Rev two is printing now, made some good changes, this rev is technically fully functional
  + Although it probably will have mistakes
  + And is unpolished
* Print is 80% done, seems something got messed up on 2 layers
  + Maybe filament got stuck, not sure
  + Should still be enough to try out the next rev

May 22:

* After some adjustments, I’m now printing what I hope to be the final rev
* Also have ordered some stuff for assembly, which I’ll likely do Saturday
* All looking good, now we just wait for prints and parts
* Docs are gonna be a bit of a pain, but I can do that in my own time later

May 23:

* Only part left to print in antenna
* TPU is annoying, the height of it causes a lot of wobbling, and supports are unusable
* Gonna try printing real slow
* All parts printed, they look pretty good
* Now to wait for parts to arrive (today) and we can start the final build

May 25:

* I’ve made it most of the way through the build
* Unfortunately, I need to order parts
* The LCD screen seems to be broken, which is likely given what I was doing to it, and I’ve run out of leaded solder (and soon unleaded)
* New parts are ordered, but build is on hold till they arrive
* Final steps:
  + Replace LCD
  + Test LCD
  + Glue in buttons and switch
  + Wire switch
  + Wire button positive leads
  + Test system
  + Done!

May 26:

* Just replaced the LCD but am having the same issue
* Will look into a few things, but if this thing came broken or was immediately broken by soldering I’ll piss myself
* To check:
  + Getting 3V3 power?
    - Yes, but maybe adding a cap would help?
  + Pins are correct (in firmware)?
    - Fuuuuuuuuuuuuck
* Ok so I really should have checked that before getting a new screen, old one is likely not broken
  + Damn it
* New screen is not quite the right size, but it’s close enough and also blue so I’ll keep it
  + Also was planning on gluing the screen anyways cause the mounts are bad
  + The holes are 2mm (my screws are 2.5mm)
* All installed, and to nobody’s surprise it works
* Glued parts, just waiting for them to dry now
* Call me god king Duncan