# Design Specs

Power Supply: Isolated 5VDC

Balance Current: 200mA max (out of the cell simulator)

Output Voltage: 2.5-4.5V +/- 5%

Balance Current can be measured with tolerance of +/-10%

Microcontroller: STM32G431CBT6

UART interface for RX / TX.

# Initial Ideas

## Linear Regulator Based Circuit

Idea: use a 1117 series linear regulator as the output regulator and output stage, since it can source high current (up to 1A). Use an ADJ variant and control the feedback pin with an op-amp in order to set the output voltage. Add current sense in the output path, but close the loop around the output of the current sense resistor in order to avoid an offset voltage on the output when current is being drawn.

Scrapped because not enough headroom. 1117 requires >1V of headroom and cell simulator needs to output up to 4.5V with a 5V supply. Don’t want to deal with adding a boost converter etc.

## Lots of Op-Amps in Parallel

Idea: use a number of op-amps in parallel as the control and output stage.

Scrapped because would need to current sense after the closed loop, (induces an offset voltage on the output), or only current sense one op-amp channel and assume that all op-amps are sourcing the same current.

## Get a Big Op Amp and Call It a Day

Idea: just get a big op-amp that can source 200mA.

Not very fun. Also don’t have any lying around, and higher current op-amps tend to have worse characteristics for input offset voltage / bias current / etc.

## Op Amp with Discrete BJT +Rail Output Stage

Idea: beef up an op-amp’s output stage with discrete BJT’s in a manner that allows the output to get as close to the +5V rail as possible. Close the loop around a set of current sense resistors on the high side of the output.

Haven’t found anything wrong with this yet!

# Design Iterations

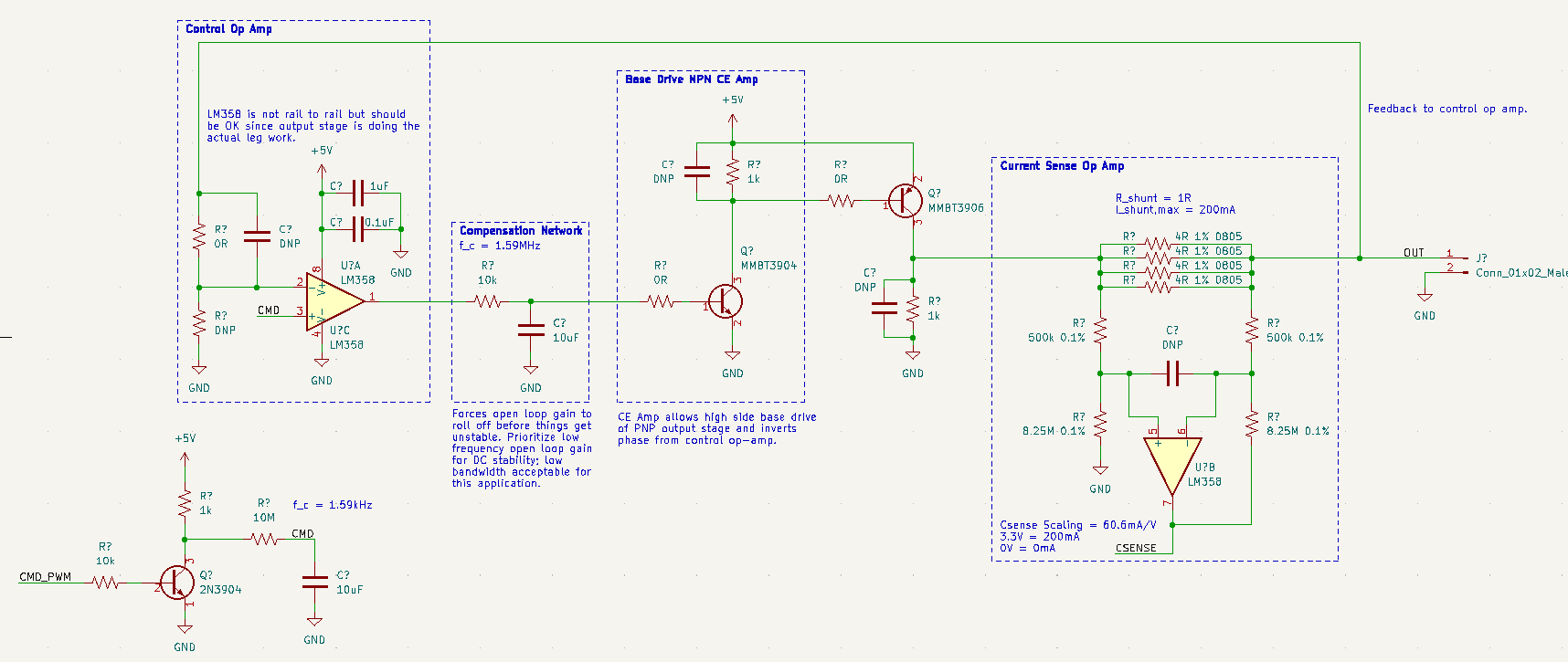
## Demo Board Rev 0.1

### Design Overview

Was hoping to use LM358’s since neither op-amp needs a rail-to-rail output, but got snagged on the fact that I needed a rail-to-rail input. The input of the control op-amp was clipping and preventing the output stage from being properly commanded to a high enough output voltage.

Encountered lots of issues with stability in simulation until I added a compensator network on the output of the control op-amp. Needed to roll off the open-loop gain before the phase got to 0°. Practically the issue with stability / oscillation can be seen when the output voltage gets really close to the commanded voltage; the output of the op-amp goes nuts and swings high/low pretty aggressively. Knocking this down with a filter prevents this oscillation from making it to the output stage.

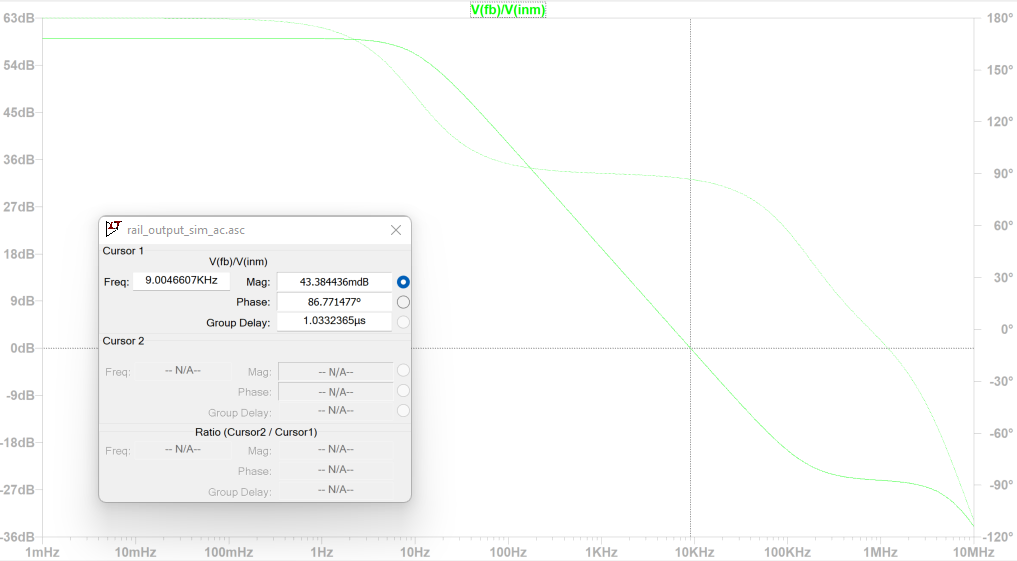
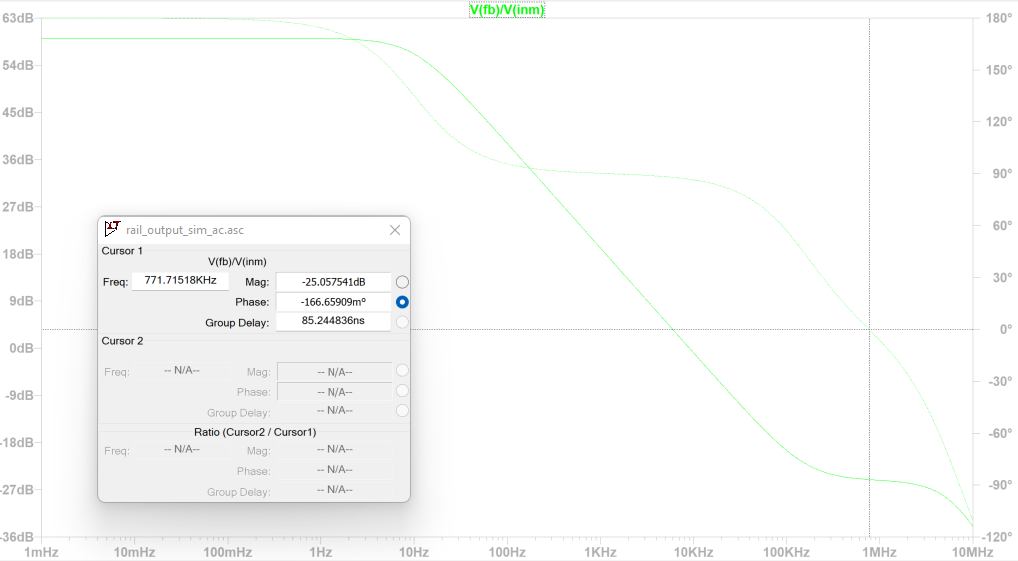
Originally the control op-amp loop was also providing gain in a non-inverting amplifier configuration so that the CMD input could be from 0-3.3V while the commanded output voltage could range from 0-4.5V.



### AC Compensation

Oscillations are a bit of a mess (the op amp is going nuts since it’s trying to drive a BJT in voltage mode—ends up kind of like an integrator and oscillates all over the desired signal). Oscillations are either well behaved (+/- a few mV about the output waveform) or really bad (rail to rail). Rail to rail oscillations occur when the output can’t reach the target voltage due to headroom issues etc. Not desirable behavior.

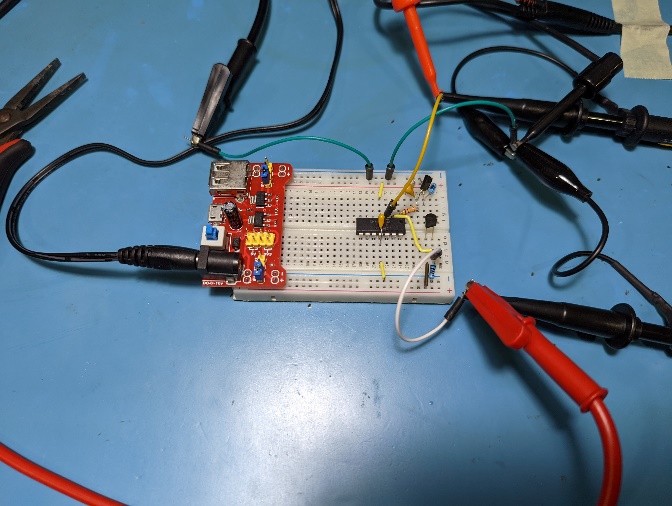
An analysis of the open loop gain showed that it was more or less OK at higher frequencies. Slapping a big RC filter on the output of the op-amp tones the oscillations down a bit, but is not an elegant solution.



Left: Open loop uncompensated gain margin. Right: Open loop uncompensated phase margin.

### Breadboard Test

Test of this circuit on a breadboard worked great (used a rail to rail op amp instead of the LM358 due to the input range issues discovered in simulation), for the most part. Output was able to source 200mA into a DC load quite easily (the main issue was the power regulator on the breadboard power supply giving out). I did notice that when the breadboard regulator would shut off, the output of the simulated BMS would have a high frequency wiggle as it faded out. Although this circuit works, I don’t really like it as it’s basically PWMing the base of the BJT after the control op-amp and then using a big RC filter to try slowing the loop down enough that the output doesn’t oscillate. The result is a lot of ripple on the output (pretty low magnitude but still feels wrong on a circuit that shouldn’t really be switching constantly). Need to find a way to drive the output stage via the op-amp without the need for aggressive rail-to-rail pusles.



Demo board Rev 0.1 on breadboard. DC load + is attached on bottom right. Drove 4.5V square wave pulses into CMD input (yellow jumper, top) via a signal generator, and ran power into a DC load set to 200mA.

## Demo Board Rev 0.2 (scbs\_pico REV A)

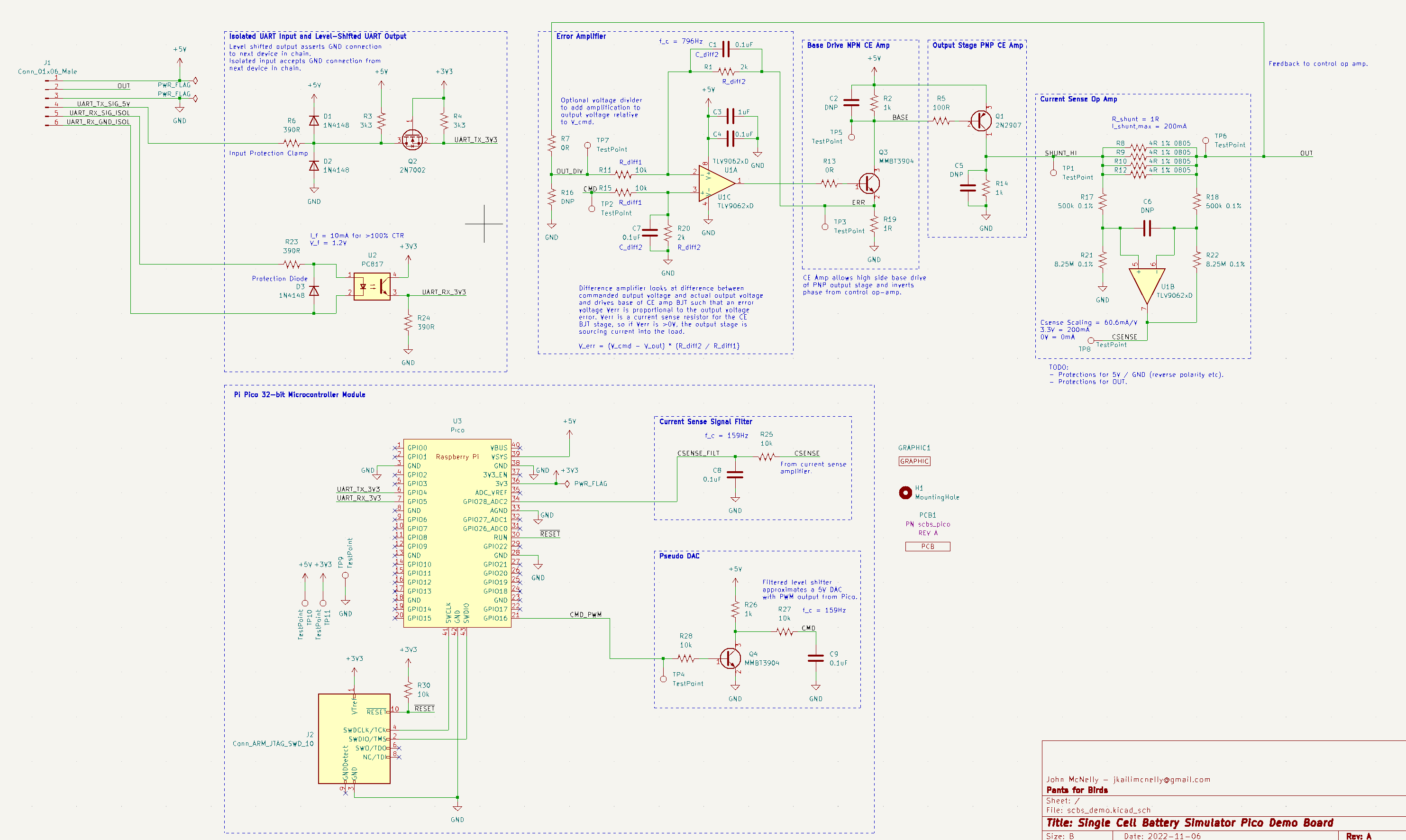
Primary motivation: remove control loop instability by reducing the gain through the control amplifier and creating a linear transfer function from error voltage to current through the base drive BJT.

The Demo Board Rev 0.1 circuit was redesigned in order to use the difference amplifier topology for its error amplification. Additional features were added to accommodate an onboard microcontroller with an isolated / level-shifted UART interface, PWM-based DAC, and SWD programming interface. A Pi Pico was chosen as the microcontroller in order to test a physical prototype, with plans to make a schematic for the STM32G4 specified in the project spec after testing.

Changing to the Pi Pico from the STM32G4 required adding a PWM filter, since it lacks the STM32’s DAC output. Tweaks were made to the error amplifier circuit such that it could be configured to add gain to the CMD channel if it was hooked to a DAC instead of PWM (gain is not necessary with PWM since it is fed into a MOSFET based level shifter and filter).

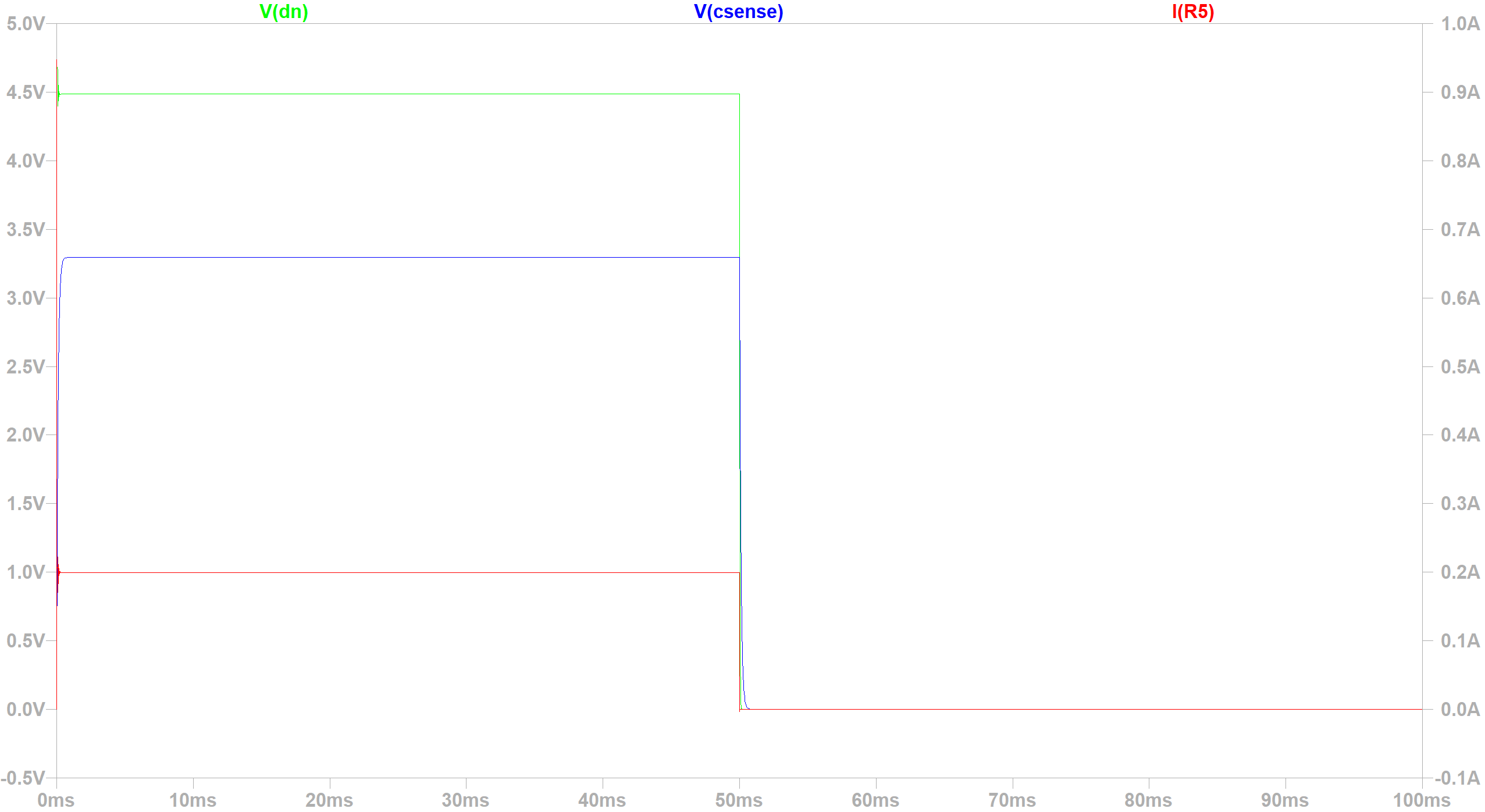
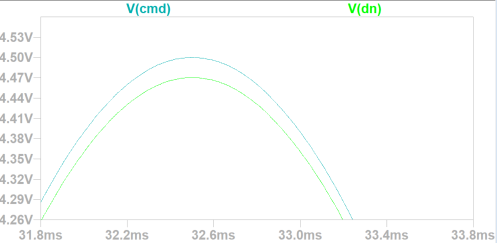
When designing the UART interface, it was decided to isolate the RX channel in order to enable it to be daisy-chained with another cell (presumably in series) with a different GND and logic HI voltage. The UART TX channel also supports daisy chaining, but assumes that the next cell in line will have an isolated RX channel, so it does not bother with isolation. UART inputs / outputs utilize 5V logic levels for improved robustness (less catastrophic if a wire gets confused with the +5V supply; this way, all input wires are +5V compatible).

No reverse polarity protection is designed for the +5V rail, but the pinout of the pin header is designed such that plugging it in backwards or offset should not cause reverse polarity on the +5V rail.

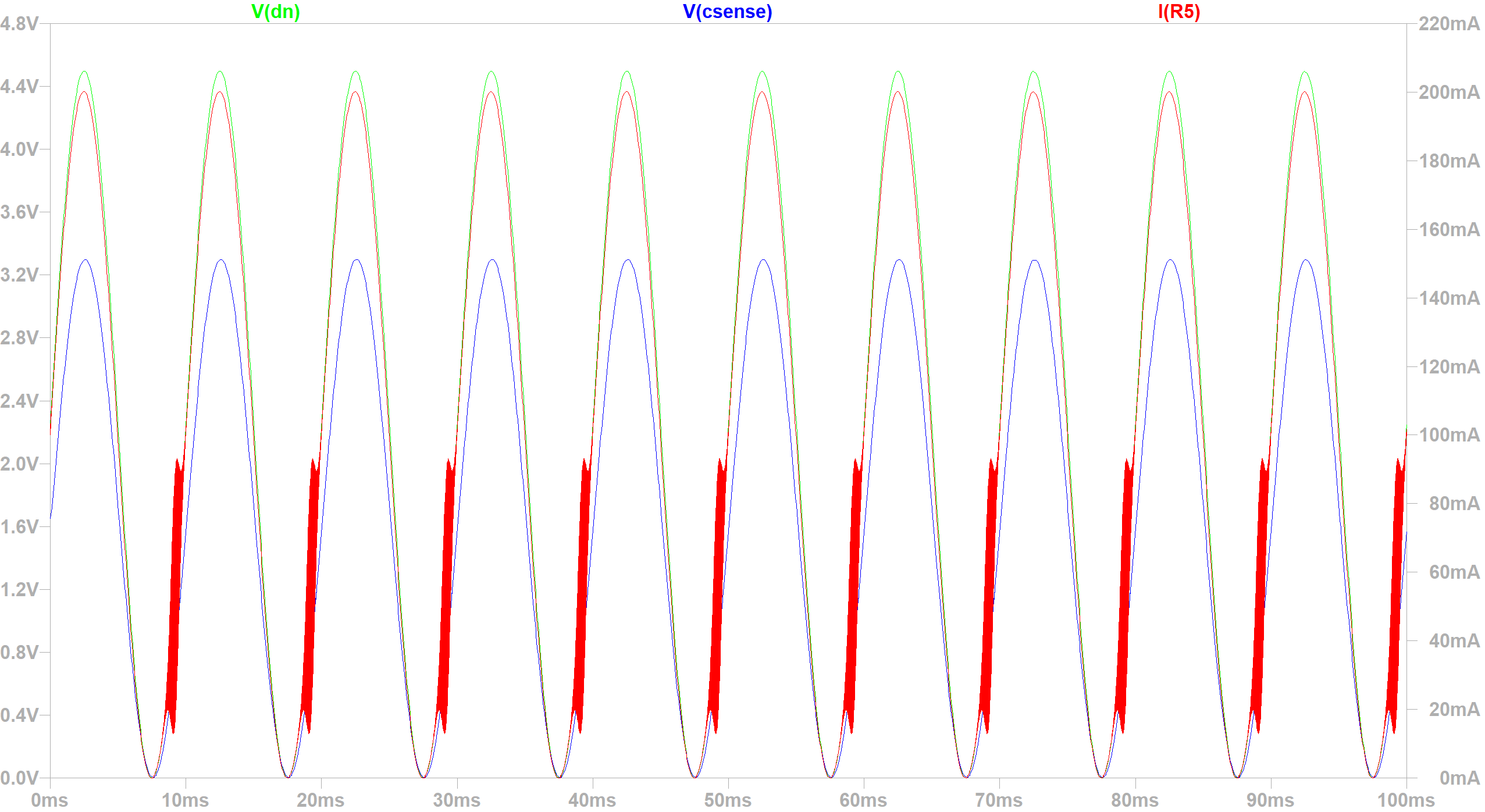
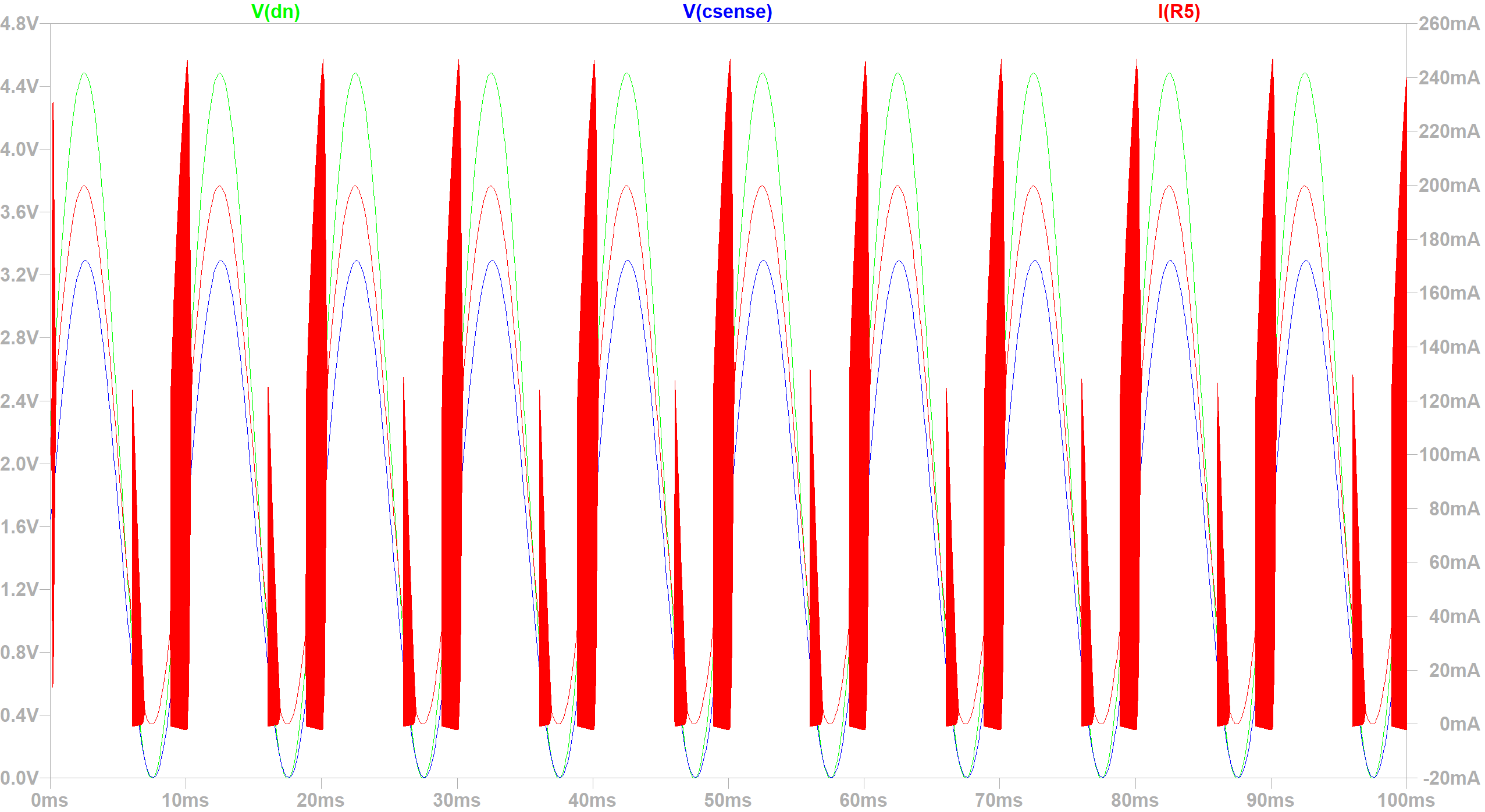


Error amplifier based schematic, with microcontroller added.

### Transient Simulation



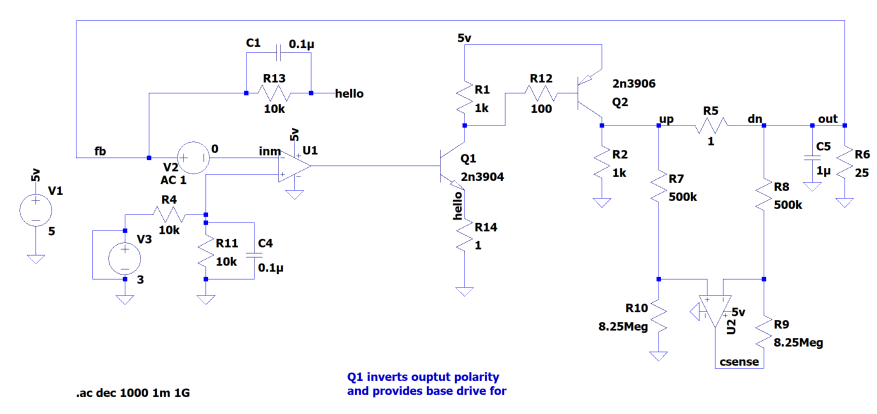
Left: Detailed view of undershoot when tracking a commanded sine wave, with an error amplifier voltage gain of 1/10. Larger gain results in less undershoot but less phase margin. Right: Well-behaved output in response to commanded square wave with an error amplifier voltage gain of 1/5.



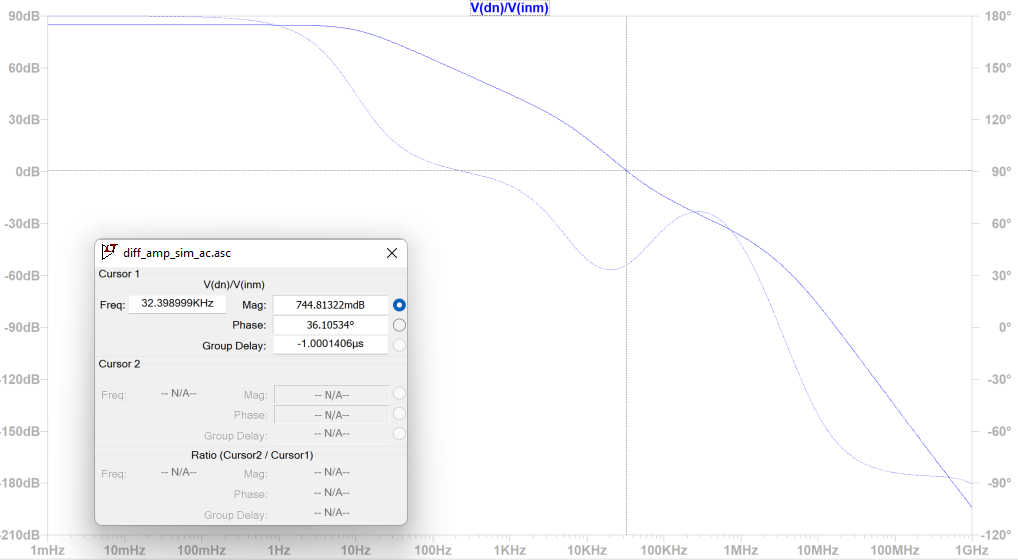
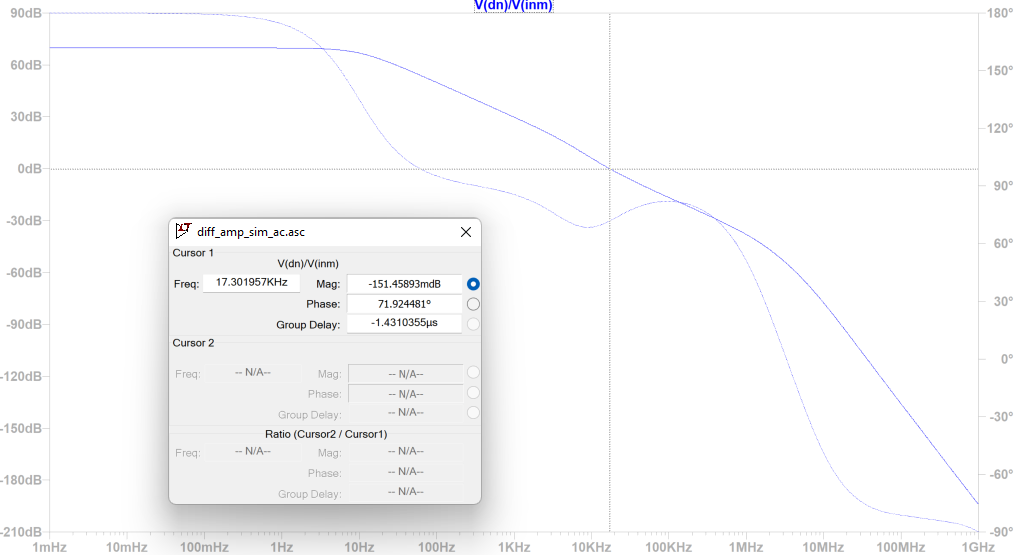
Left: Oscillations induced by removing bypass caps on gain setting resistors in the error amplifier circuit. Right: Oscillations induced by setting error amplifier voltage gain to 1.

### AC Simulation

Using a low error amplifier gain is important for creating enough phase margin to avoid oscillations on the output stage. Increasing from a gain of 1/5 to a gain of 1 creates instability in AC and transient analysis. Adding bypassing to the error amplifier circuit is also important for avoiding oscillations. A corner frequency slightly above 1.5kHz was chosen for convenience and seems to work pretty well.



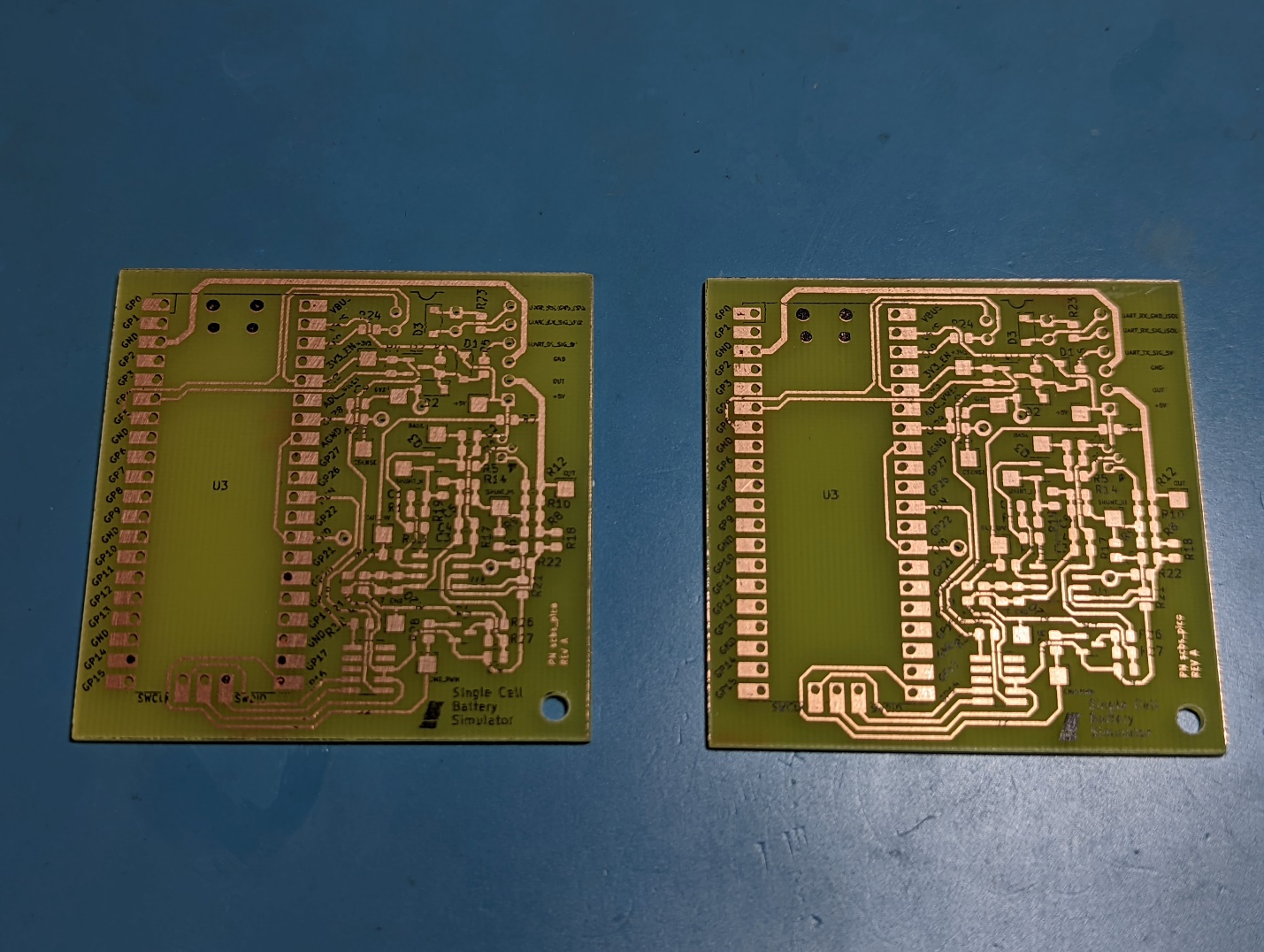
LTSpice schematic used for AC analysis.



Left: Phase margin with error amp gain of 1/5. Right: Phase margin with error amp gain of 1.

### Homebrew PCB Prototype

2x pico\_scbs boards were etched using a toner transfer process. Additional toner was applied to the top layer as legend. Boards are single sided. Note that Pi Pico needs to be populated for testing since it is used for sharing ground connections due to routing constraints.



Notable changes when populating the board:

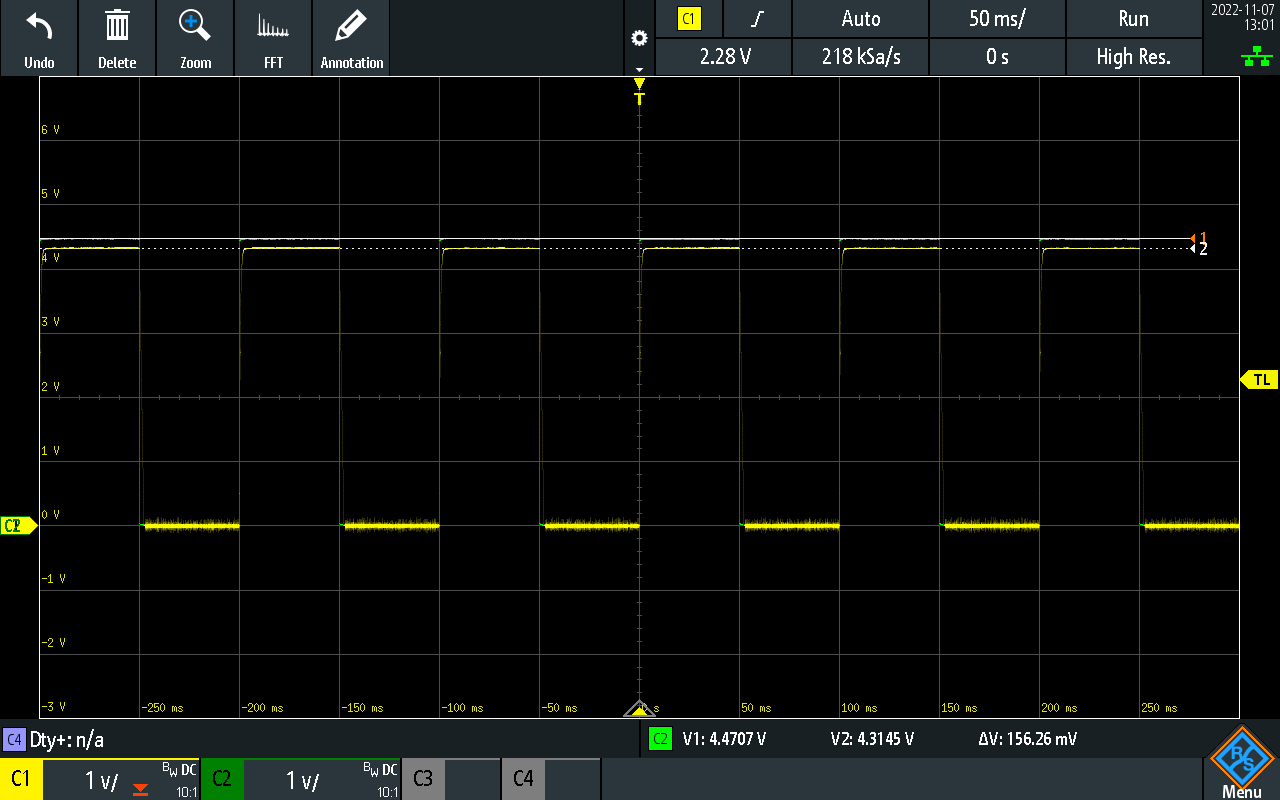
* Didn’t have 4 Ohm resistors for the shunt resistor, approximated with 3x 3.9 Ohm resistors in parallel with 1x 4.3 Ohm resistor.
* Didn’t have 500 kOhm resistors, swapped with 510 kOhm resistors.

Notable changes during bringup:

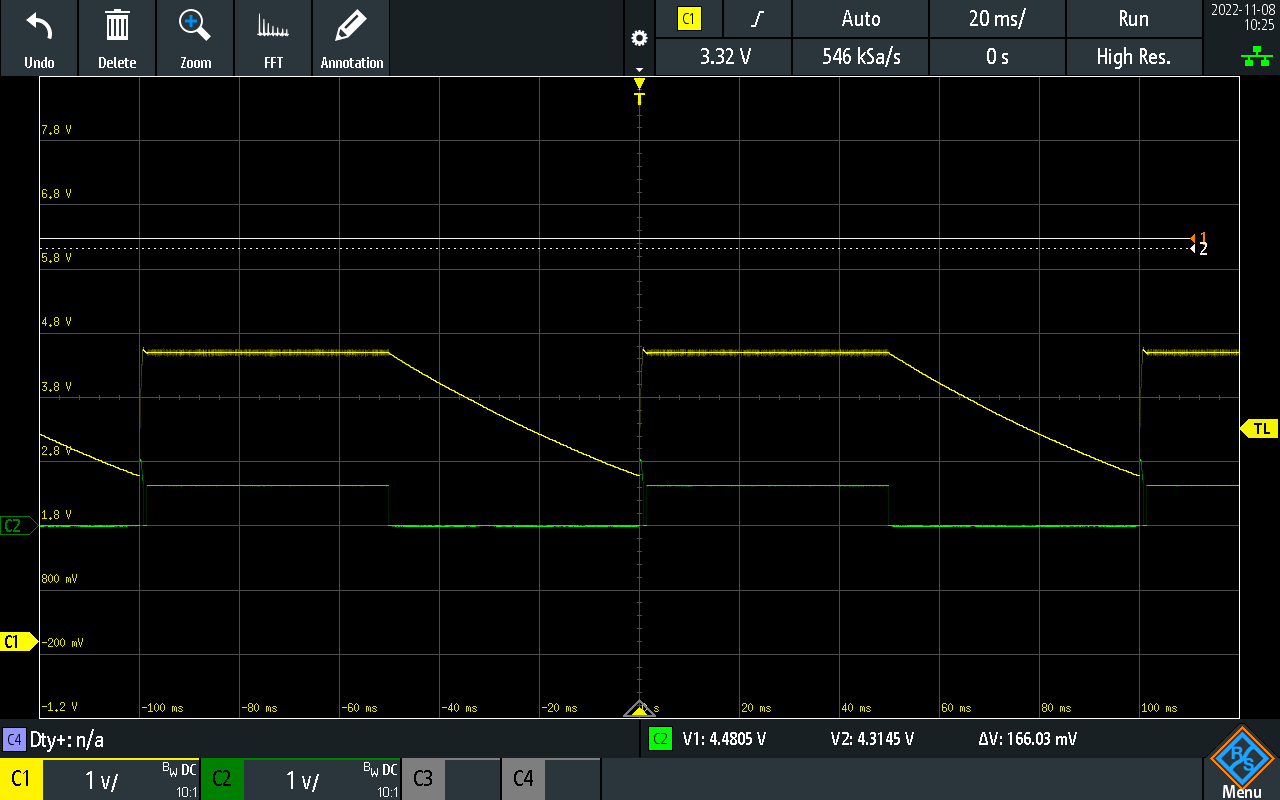
* Chasing my tail while the output stage PNP was backwards…
  + Changed R5 to 0 Ohms to try increasing base current to output stage PNP transistor. Later changed back to 100 Ohms.
  + TLV9062 has minimum open loop gain of around 100dB (10k), whereas the op-amp used in sim had an open-loop gain of 10M. There might be an issue with not enough gain for properly closing the loop.
  + As load increases, output voltage decreases until it suddenly drops off a cliff. Turns out my generic Amazon 2N2907 PNP BJT on the output was completely flipped! The generic device I had is EBC not CBE like the legit manufacturer device (pins in ascending order). Transistor tester saved the day.
* Testpoint ERR is mistakenly labeled SHUNT\_HI on legend.



Homebrew PCB was tested with 5VDC power supply, DC load, signal generator, and 2x oscilloscope channels. Analog portion of the board was brought up first with commanded output voltage signal injection via the white wire hooked to the CMD net.



CMD Input (CH2, green) and OUT output (CH1, yellow) waveforms showing tracking for a 10Hz 0-4.5V square wave. Output is 3.3% below commanded voltage (within spec).



Output not responding to 1Hz square wave (slew rate super low), I think Q3 might be fried? CH1 (Yellow) is OUT. Ch2 (Green) is base of Q3. Haha I’m dumb, DC load wasn’t on.