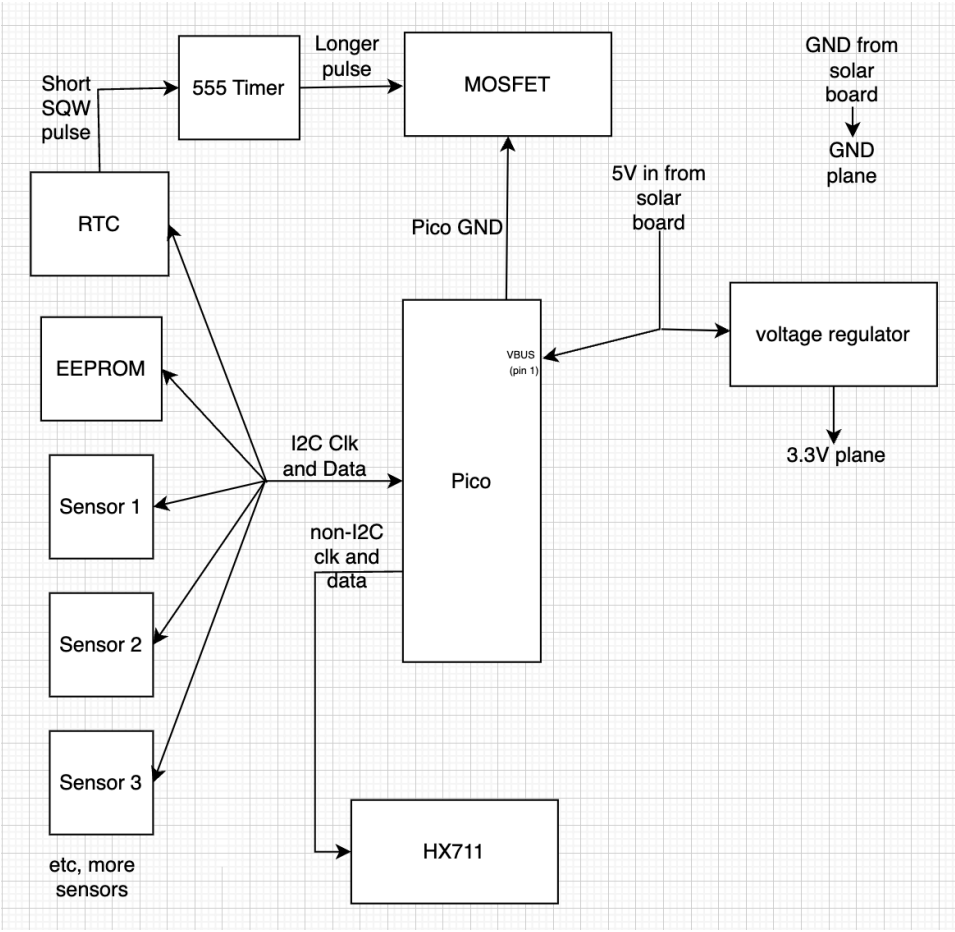
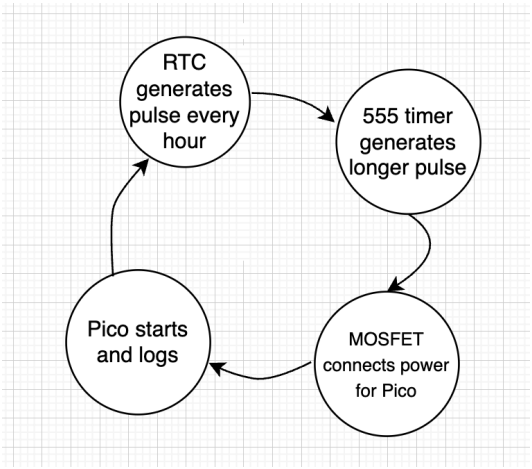
PCB Information
Cameron Fuller 1/25/24

Block diagram (see actual schematic and layout for more detailed view):



The cycle of waiting and power-up:



The pico:

- Reads from each Sensor over I2C
- Reads from HX711 (not I2C, ask weigh scale team for details)
- Reads the time from the RTC over I2C
- Writes data to the EEPROM

Explanation of the 555 Timer and MOSFET:

Maybe read the Plan.pdf that describes this, it's on the website and in the data-collection channel from November 2023. But here is it reiterated:

To save power, we don't want the pico to be on all the time. Rather than put it into a power-saving sleep mode (you can read more about that on the January blog post), we save power by only briefly allowing the pico to power on. This is how we do that:

The pico can be powered by giving the VBUF pin 5V (instead of plugging in the USB cable), which goes to the pico's onboard regulator, effectively powering on the pico whenever 5V is connected to VBUF.

But we want to be able to control when the pico turns on, so we connect 5V to the VBUF pin *but do not connect the pico's gnd to the gnd of the 5V source.* Instead, the pico's gnd goes to the Drain of a MOSFET, which basically acts like a switch, so we can use the Gate pin of the MOSFET to control when the pico's power circuit is complete. If this doesn't make sense, look up a guide on using a MOSFET as a switch.

We want the Gate of the MOSFET to go High periodically to turn on the pico at a set interval. To do this, we use our RTC module's "alarm" functionality, which sends out an active-low signal once per hour (read the plan.pdf for more description on how to do this). The only issue with that is that it's active-low and the actual length of the pulse is too short for the pico to do all the reading and writing it needs to do. So we use a 555 timer in monostable mode to generate a longer pulse, triggered by the SQW output from the alarm on the RTC.

The result of this is that once per hour, the RTC sends out a short active-low pulse to the 555 timer which generates a longer active-high pulse, which goes the the Gate of the MOSFET and allows the pico to turn on. The pico does all its reading and logging, then after it's done the power gets cut so it's back to sleep.

This will use very little power, because the pico is only on for 1-2 seconds per hour. The quiescent currents of the other parts are <1mA total, so this is a really low-power way to do things.

Explanation of the 3.3V regulator:

The EEPROM needs to be able to be read by the reading device at any point, and the reading device does not connect its own power to the EEPROM. So there needs to be 3.3V powering the EEPROM all the time, which cannot come from the pico's 3.3V regulator because it will be powered off except when reading sensors and logging.

So we regulate the 5V down to 3.3V so the EEPROM can be powered all the time. It's also useful to power the RTC from this, so it doesn't use up its own coin cell battery too fast.

Parts chosen:

LMC555 timer chosen because CMOS uses very little power

The **LP2950CZ-3.3** 3.3V regulator was chosen because it has very low drop-out and therefore has very low quiescent current

The **DS3231 module from Parallax** RTC was chosen because it has alarm functionality

The **FQP30N06L** n-channel MOSFET was chosen because its threshold voltage is low enough to be triggered by the 3.3V output of the 555 timer.