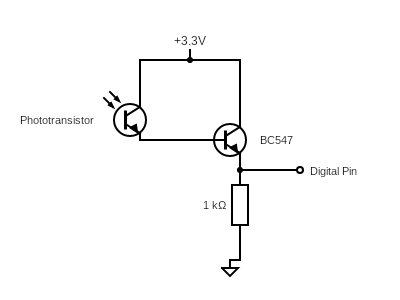
# Sensing IR Pulses

## Developing a solution and reviewing the results

Detecting infrared pulses can be accomplished using phototransistors, which we have used before in the Spring skills lab. Phototransistors [1] can detect visible and infrared light; a current is produced through it when photons of IR and visible are incident on the exposed base and can be measured by placing a load resistor in series with it so the voltage across it can be measured, forming a similar circuit configuration shown in the skills lab instructions [2]. We did this since the Adafruit board cannot measure current inputs; hence, voltage inputs were used instead. To improve upon that circuit, we coupled the phototransistor with a class A amplifier to form a photodarlington pair [3], as shown in fig. 1.

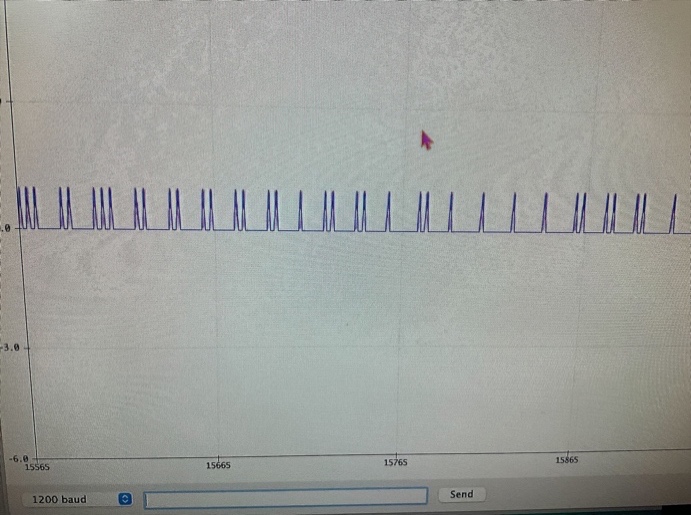
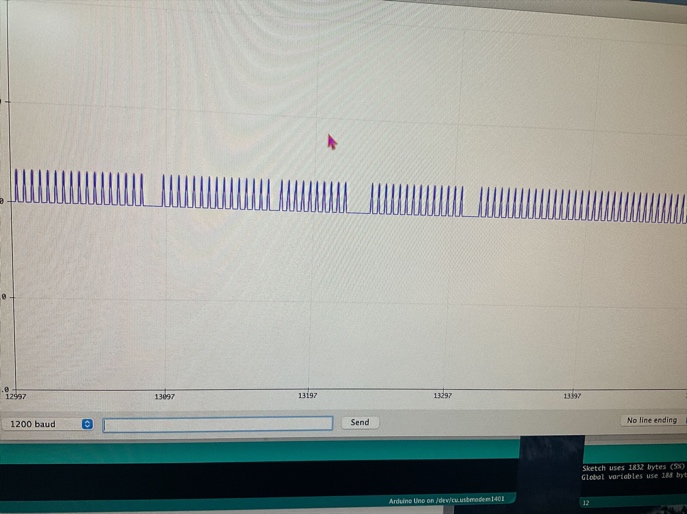


*Fig. 1: Photodarlington pair coupled with the emitter resistor to measure the voltage produced by IR pulses incident on the phototransistor.*

Coupling the phototransistor to a transistor to form a photodarlington increases the current gain at the emitter, resulting in a greater voltage being measured compared to a configuration without the class A amplifier.

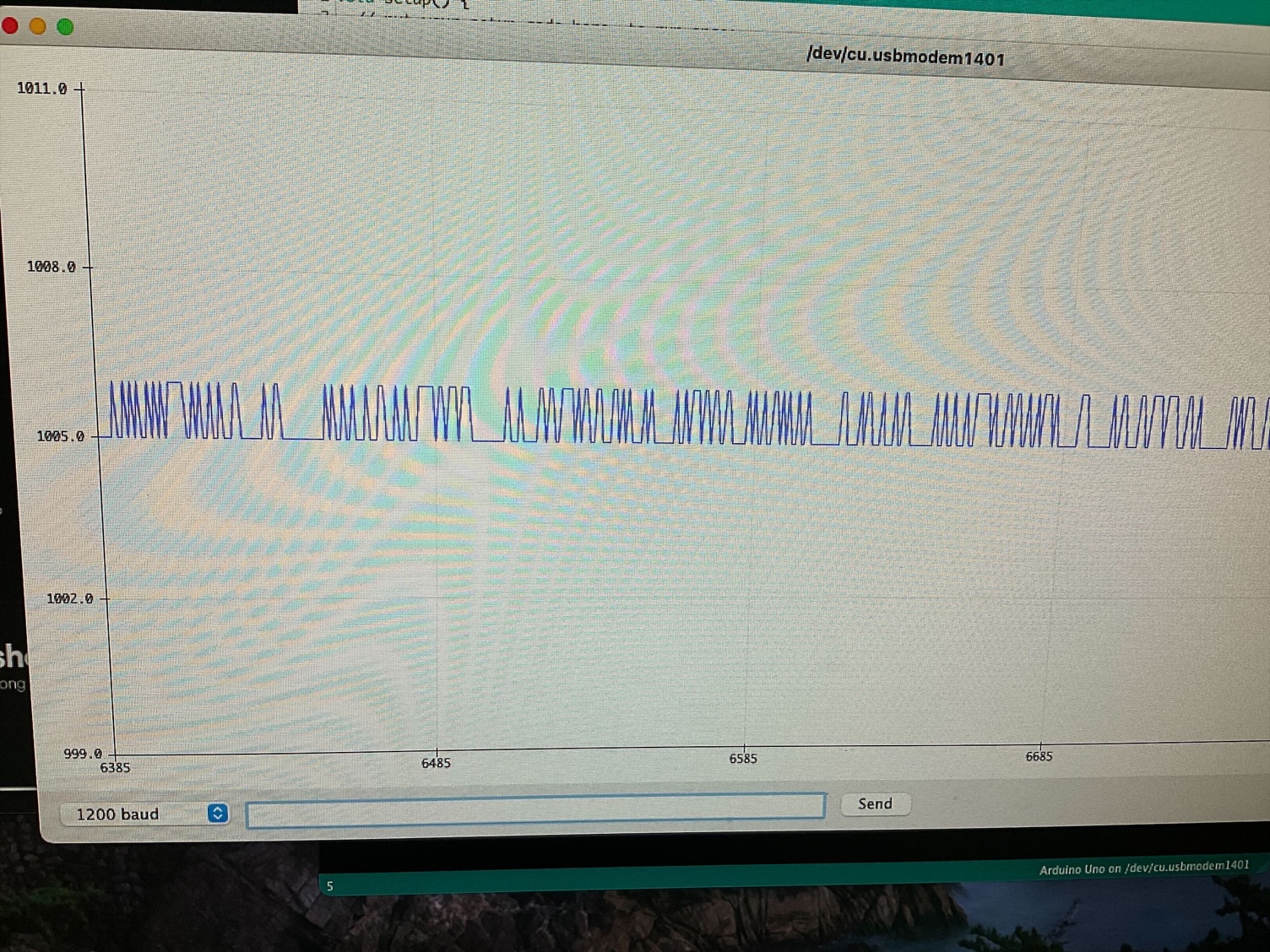
We set the supply voltage at 3.3V as this reduces the chance of causing damage to Adafruit voltage regulators when an external circuit is used as an input into the board. The signal was a square wave therefore, we can analyse the frequency of the signal using digital pins on the board.

The signal produced by the circuit in fig. 1 was processed using by plotting the digital input on the serial plotter software (for the code used to achieve this, refer to appendix 1). Initially, the signals were measured with the exorock shell removed to determine whether the code was correct for detecting the signal and the circuit arrangement. Detection of the pulses was successful for both pulse frequencies (see fig. 2). However, there was some unexpected behaviour reported by the serial plotter software: the detection of false pulses and no pulses were produced when the exorock shell was installed.



*Fig. 2: Arduino serial plotter readings from pulses of infrared of frequency 571 Hz (left) and 353 Hz (right).*

A combination of signal noise and ambient light incident on the phototransistor caused the presence of false pulses to be detected in the serial plotter (see fig. 3). These pulses could interfere with the actual pulses we are detecting; thus, we need to determine a method to remove these false pulses.



*Fig. 3: Arduino serial plotter readings with no infrared pulses directed at the phototransistor. Visible light and noise caused pulses in the signal.*

Since we detected no pulses when the exorock shell was installed, we had to analyse the signals from the circuit using more sophisticated software such as picoscope to understand the problem. When the exorock was placed directly next to the phototransistor, the signal was extremely weak, with a small amplitude among the noise in the signal. For any other distance, the picoscope would not detect the pulses, so we had to design an amplifier to amplify the signal amplitude.

## Refinements and Implementation

To prevent the interference of ambient light, we need to cover the phototransistor so visible light would reach the exposed base. We could implement this either using an IR phototransistor or 3D printing a cover for the phototransistor. Both methods would work but since we are conscious of the rover's weight, we used the IR phototransistor since the 3D printed cover would contribute to additional weight, which was undesirable. The phototransistor would block visible light but allow infrared light to pass through. This was how we prevented visible light from interfering with our signal.

We can remove noise by implementing filters with corner frequencies near the pulse frequencies (353 and 571 Hz). High-frequency noise was removed using a low-pass filter with a corner frequency of 589 Hz and low-frequency noise was removed using a high-pass filter with a corner frequency of 339 Hz.

We amplified the signal from the exorock using a non-inverting amplifier with a gain of 20,001; gain and resistor values were purely experimental. The output of this amplifier was coupled with another non-inverting amplifier with a more conservative gain of 11–depending on the distance between the exorock and phototransistor, the signal wasn’t amplified adequately, thus this amplifier was added. We set the supply rail voltage for both amplifiers to 3.3V because the amplitude limit of the amplified signal would be 3.3V, the logic level of the Adafruit board and there wouldn’t be any risk of damaging the voltage regulators on the board. If a 5V supply rail was used, the amplified signal may damage the board as it exceeded the board’s limit.

Diagram, schematic

Description automatically generatedThe order of components would affect the output waveform when all parts of the circuit are combined (for the different waveform shapes and circuit arrangements, refer to appendix 2). The final circuit configuration is shown in fig. 4.

*Fig. 4: The final circuit used to detect infrared pulses, featuring both amplification and filtering stages to the signal.*

## Signal analysis

With our processed signal, we need to determine the frequency of the infrared pulse to correctly identify the mineral sample. We determined the frequency of the signal by developing code that ran on the Adafruit (see appendix 3). The pulse frequency was outputted to the serial monitor window. We imported all readings to an excel spreadsheet, formatted the data, and then performed calculations to determine the 10th and 90th percentiles. The set of frequency value readings between these two values would trigger a line of code displaying which pulse frequency was detected. If the frequency reading was outside of the region, then no output would be displayed.

To test our circuit and code, first, we shined visible light at the circuit and didn’t detect any signal hence our circuit blocks visible light. This was expected since we used an IR phototransistor, which offered this property. Next, we assessed the effectiveness of our final circuit by placing a picoscope probe out of the output node and analysing the waveform shape displayed. This was a distinct pulse (see appendix 2), meaning the circuit configuration provided the expected result. Finally, running the code that identifies the pulse frequency within a tolerance range determined using percentiles (refer to appendix 4). Our code can detect pulse frequencies of 353 and 571 Hz (refer to appendix 5).

## Evaluation:

The main difficulty in detecting this signal was the circuit used to detect the pulses. Despite using a Darlington pair configuration with a phototransistor, we didn’t expect the need to amplify the signal further and didn’t expect to use an amplifier with a gain greater than 1000. Countless hours were spent attempting to amplify the signal with a more reasonable gain such as 20 but we didn’t obtain our desired result until doing something ridiculous and forming an amplifier with a gain of 20001 and only then did we obtain our expected result. We considered using second-order filters instead of first-order filters due to their drop-off per decade being double that of first’s but the additional space they took up to essentially do the same wasn’t worth the trade-off. Developing the code to determine the frequency of the pulse was the easiest part of this signal.

# Reference:

1. Circuit Globe. “Phototransistor”. circuitglobe.com. <https://circuitglobe.com/phototransistor.html> (accessed May 30, 2022)

2. E. Stott. (2022). EEE/EIE Lab Handbook: Lab Skills Part 2 [pdf document]. Available: <https://edstem.org/us/courses/15387/resources>

3. Electronics Notes. “Photodarlington”. electronics-notes.com. <https://www.electronics-notes.com/articles/electronic_components/transistor/what-is-a-photodarlington.php> (accessed May 17, 2022)

# Appendix

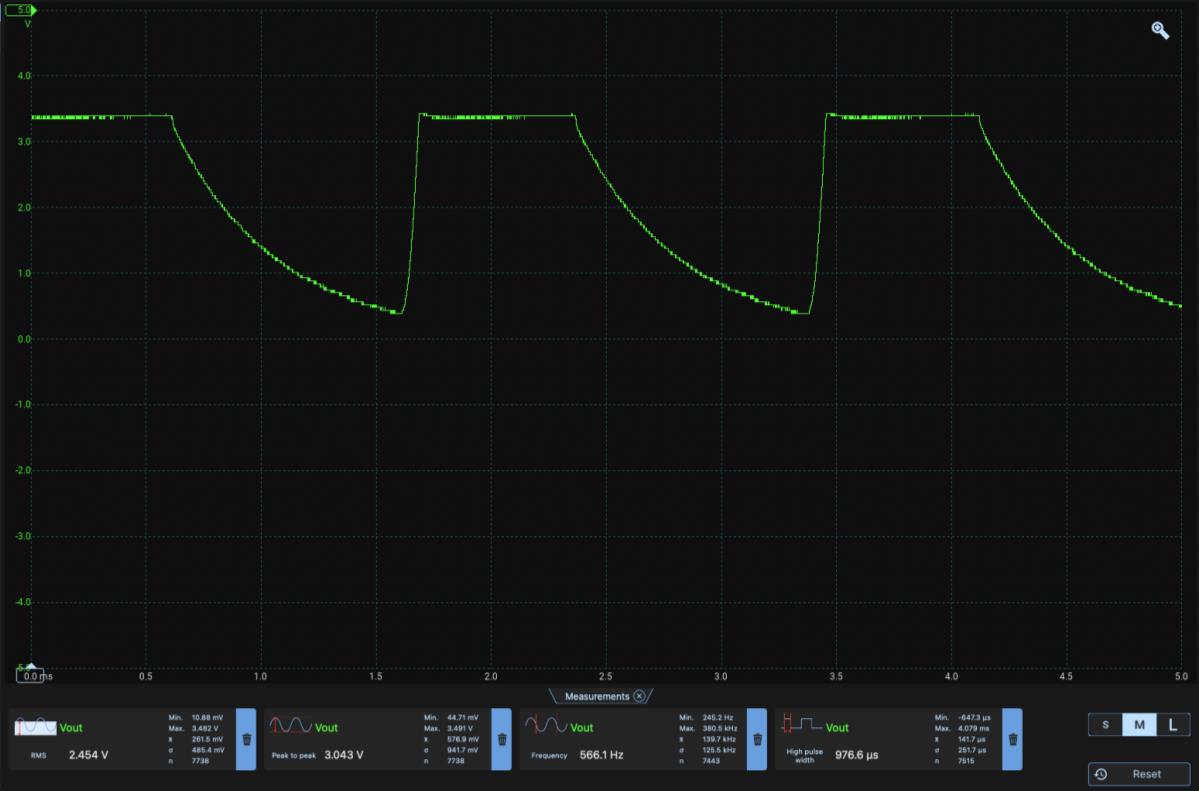
1. Code used to convert the digital input signal into a waveform using the serial plotter software. Code can be found here: <https://github.com/shekratul10/EEProject/blob/main/Sensor/digital_signal_tests.ino>

2. Different combinations of circuits that follow the photodarlington result in various output waveforms. Conducting amplification stages before filtering the signal resulted in the following waveform for infrared pulses of 571 Hz.



Having both filtering stages (low-pass and high-pass) before the amplification stages result in the following waveform:

And if the signal is filtered once, amplified once, filtered again, and amplified, then the following waveform is produced at the output.

We can see that the infrared pulses are more apparent in the last circuit configuration. Although the picoscope software detects these pulses at the correct frequency, the Adafruit may have difficulty determining the pulses from input signals, such as the second circuit arrangement (band-pass followed with amplifiers). For our sensor circuit to be reliable, the pulses need to be distinct for the Adafruit to detect it reliably hence the final circuit configuration is the last one.

3. Code used to determine the frequency of the pulse signal and output the frequency of the signal. The code can be found here: <https://github.com/shekratul10/EEProject/blob/main/Sensor/IR%20Pulse/code/IR_digital_test1.ino>

4. Code used to determine whether the signal corresponds to either frequency emitted by the mineral samples. The code can be found here: <https://github.com/shekratul10/EEProject/blob/main/Sensor/IR%20Pulse/code/IR_digital_analysis.ino>

Graphical user interface, application

Description automatically generatedGraphical user interface, application

Description automatically generated5. The serial monitor window showing both infrared pulse frequencies (353 and 571 Hz) detected using the code in appendix 4 and circuit in fig. 4. Pulses with a frequency of 571 Hz are shown on the left and pulses with a frequency of 353 Hz are shown on the right.