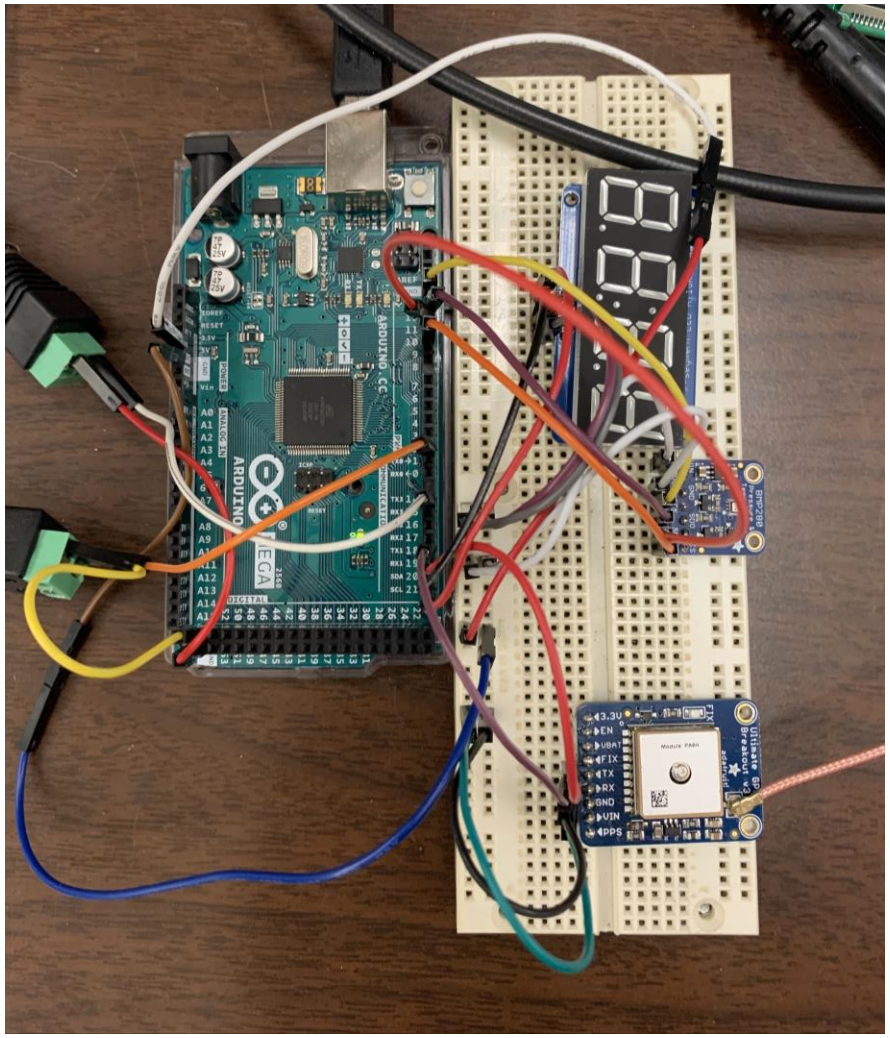
# Redefining data acquisition programs used to collect synthetic cosmic rays generated by pulse generators

Presenter: Gabe (Jun ha) Kim

Instructors: Professor Armendariz & Professor Stalerman

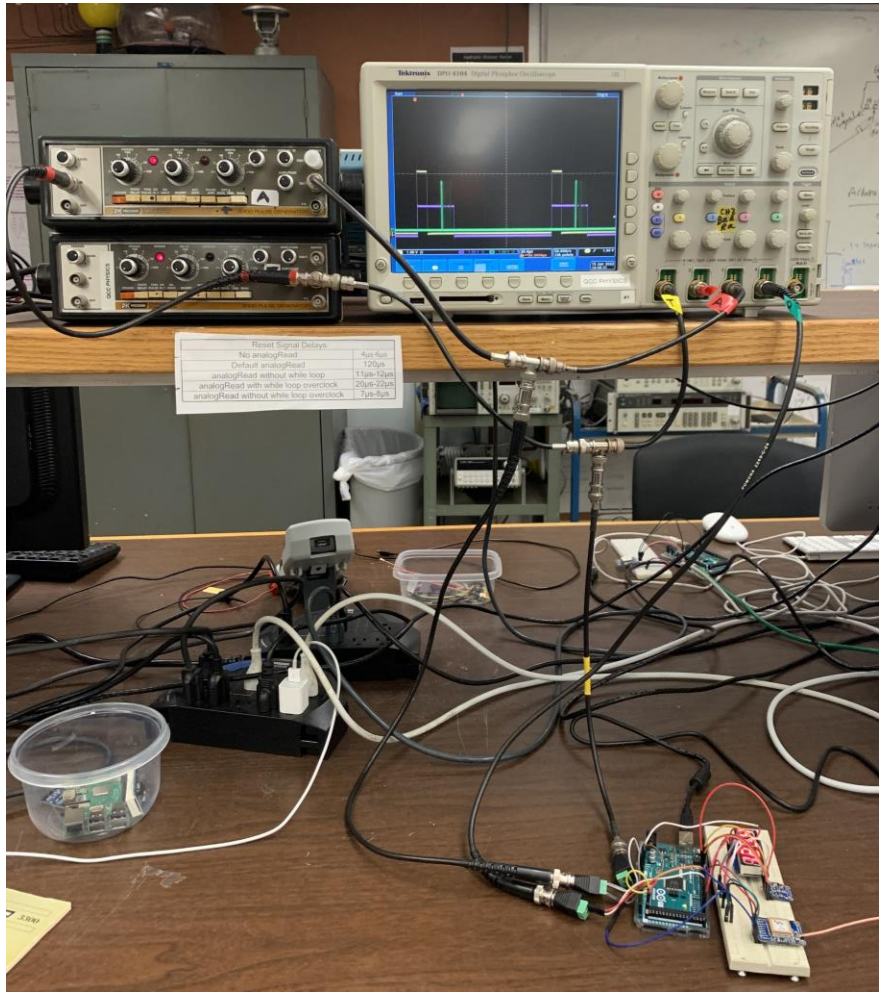
# Arduino Board Setup

- Composed of x1 Arduino Mega, x1 Breadboard with sensors
- Placed on Breadboard:
  - GPS
  - Temp Sensor
  - LED light
- Arduino Board:
  - Connected to pulse generators
  - Code runs on this setup



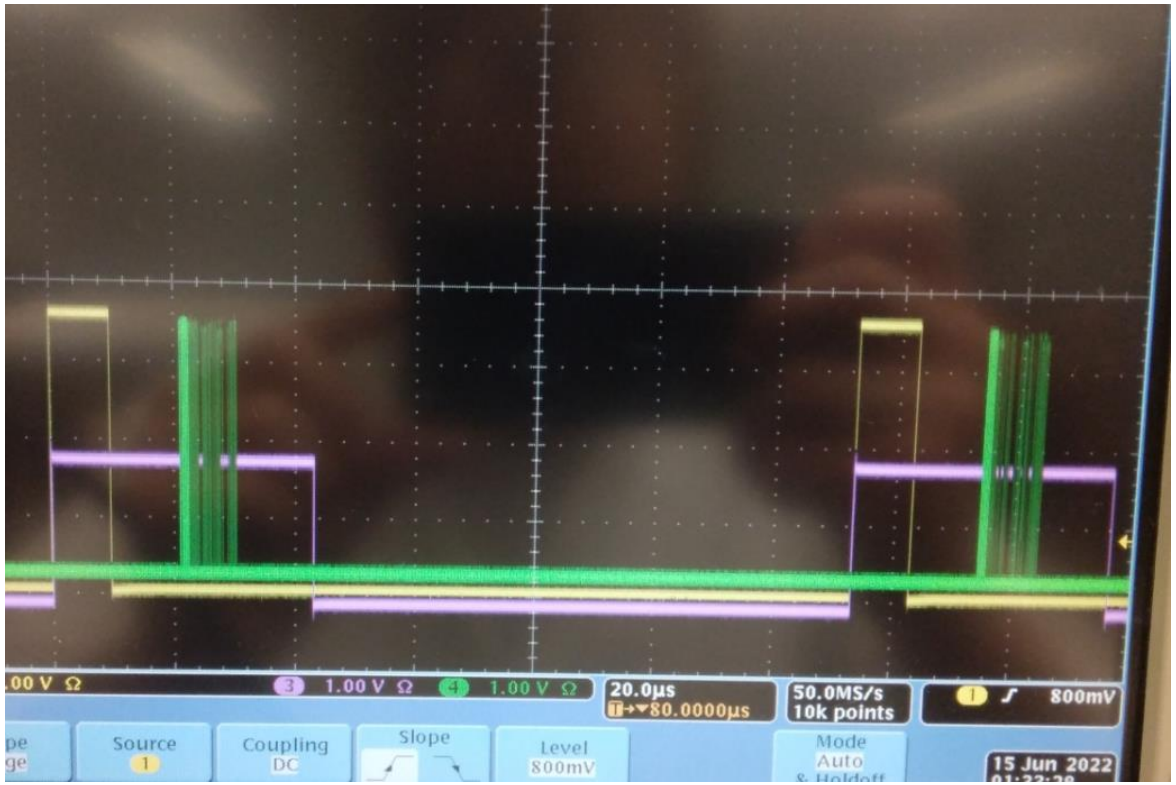
# Setting up Pulse Generators

- Top pulse generator
  - Sends out a “trigger” signal
- Bottom pulse generator
  - Sends out an “analog” signal i.e. Voltage
- Oscilloscope
  - Displays information visually
- Arduino Setup



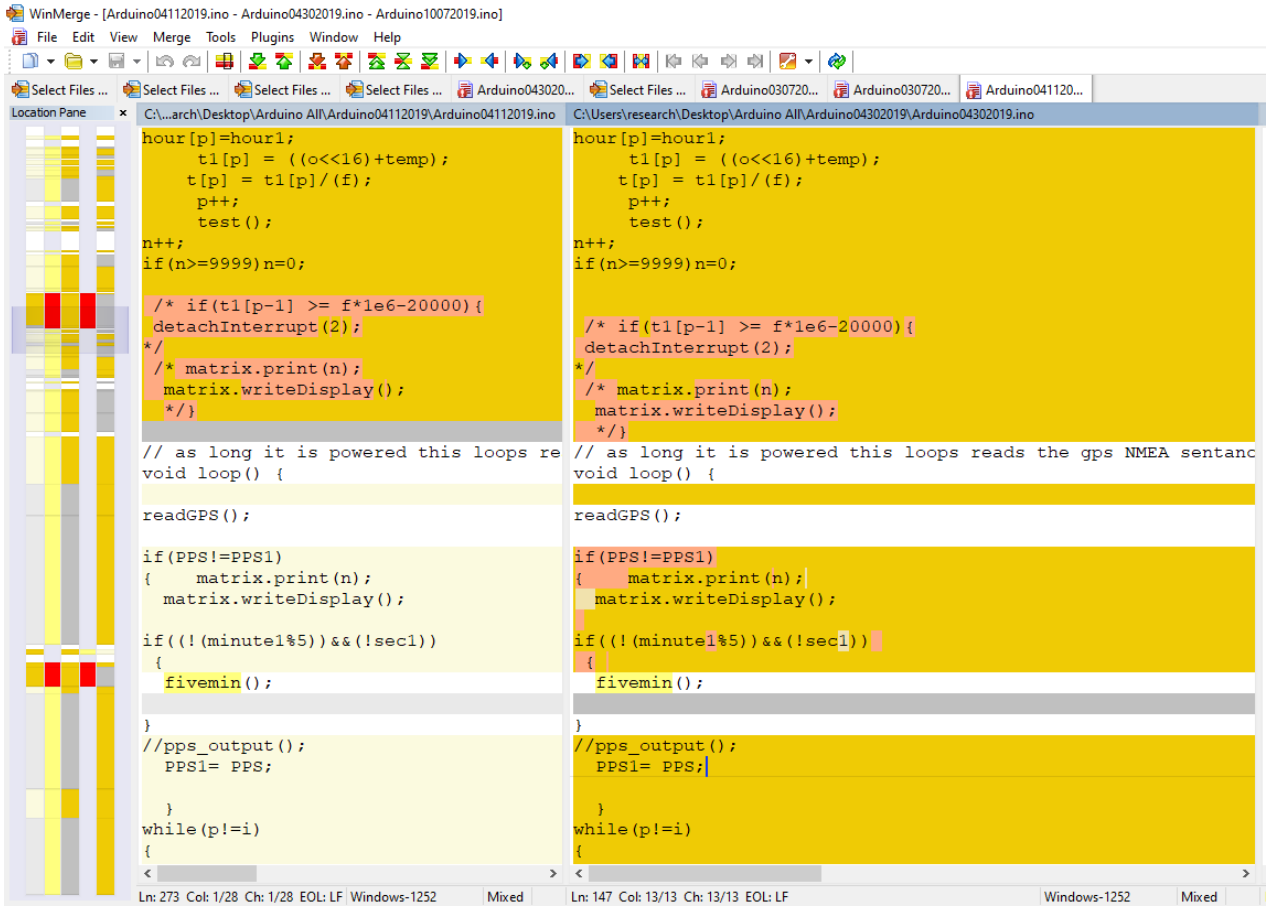
# Oscilloscope Output

- Yellow pulse
  - Trigger pulse
  - Pulse generator emits a signal (ranges from 1 to ~200,000) every 1 second
  - Period and Width knobs
- Purple pulse
  - Analog pulse
  - Pulse generator emits a signal that describes the amplitude of the wave
- Green pulse
  - Reset signal disseminated from the Arduino board, after the Trigger pulse



# Using WinMerge

- WinMerge: Third party software used to see differences in distinct versions of code
- Coloring scheme:
  - White = No difference
  - Yellow = Difference detected in block of code
  - Red = Line or character differences



# Working Versions of Code Selected

- Confirmed the file works in various period and width settings
  - Trigger Rate: 1 Hz, Width: 100 ms
  - Trigger Rate: 10 Hz, Width: 10 ms
  - Trigger Rate: 100 Hz, Width: 1 ms
  - Trigger Rate: 1 kHz, Width: 100 µs
  - Trigger Rate 10 kHz, Width: 10 µs
- Rate of occurrence (sentence output) is adequate, as seen on the right

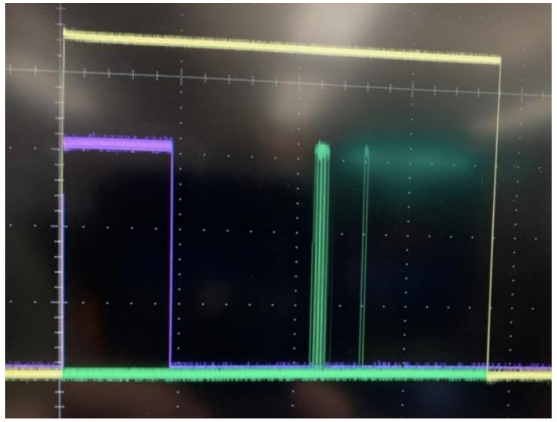
| N/A | Analog Pulse Voltage (V) | Coordinated Universal Time (UTC) HH MM SS | | | Amount of Time after Pulse Per Second Signal (PPS) (µs) | Cycle Count Difference |
|---|---|---|---|---|---|---|
| 1 | 0.62 | 18 | 10 | 0 | 972234 | 15554315 |
| 1 | 0.62 | 18 | 10 | 0 | 978283 | 15651090 |
| 1 | 0.62 | 18 | 10 | 0 | 984331 | 15747853 |
| 1 | 0.62 | 18 | 10 | 0 | 990381 | 15844649 |
| 1 | 0.62 | 18 | 10 | 0 | 996429 | 15941395 |

# Edits made to code

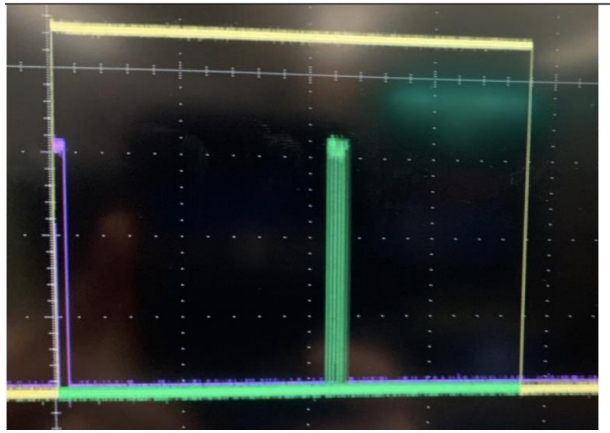
- Comments & Naming Conventions
  - *// Description: This code captures trigger pulses and the analog voltage from the trigger and analog pulse generators respectively*
- int n=0 → unsigned long int totalTriggers=0;
  - totalTriggers counts how many trigger signals are received over a 1 minute period
  - Variable type changed to capture more numbers (up to 4 billion values)

# User Manual

- For those unacquainted with the program
- I discussed:
  - Items and Libraries required for the program
  - Objectives
  - Conditions that prevent the program from working
  - How to understand the output



- In this scenario the analog pulse is approximately 120 milliseconds away from the reset pulse
- However, the analog pulse does not overlap with the reset pulse, preventing precise data collection (i.e. Voltage information will fluctuate)



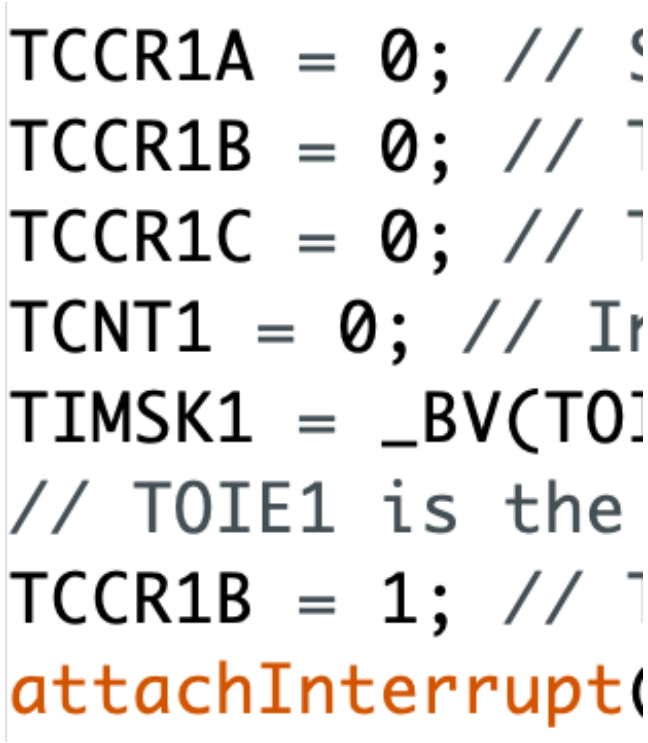
- The trigger pulse overlaps with the reset pulse
- The analog pulse is placed further away from the reset pulse (about 200 milliseconds)
- In this scenario, no Voltage data will be collected using serial monitor

# Analog to Digital Converter (ADC) Limitations

- However, I discovered scenarios that prevent the code from working:
- Limitations exist that cause voltage fluctuation on Serial Monitor (if trigger period < 100 µs)
- Per official documentation, it takes ~ 100 µs to read an analog input
  - The maximum reading rate is about 10,000 times a second
  - Analog to Digital Converter (ADC) is at fault
  - Hardware limitation

# Importance of Timer Interrupts

- Timer Interrupts enable the program to run a new set of commands
- It pauses the execution of the loop() function for a predefined number of seconds
- Timer1 is a 16-bit timer, so the timer will increase its value to 65,535 before reverting to 0
- Once executed, the program resumes at the same position (i.e. the loop)



# Conclusion

- Developed familiarity with pulse generators and oscilloscopes
- Learned how to use and prepare Arduino and breadboards for data collection purposes
- Discovered the importance of good naming conventions and comments while writing code
- Gained insight into hardware limitations that prevent data collection in higher frequencies
- Need to familiarize with Timer Interrupts and Registers