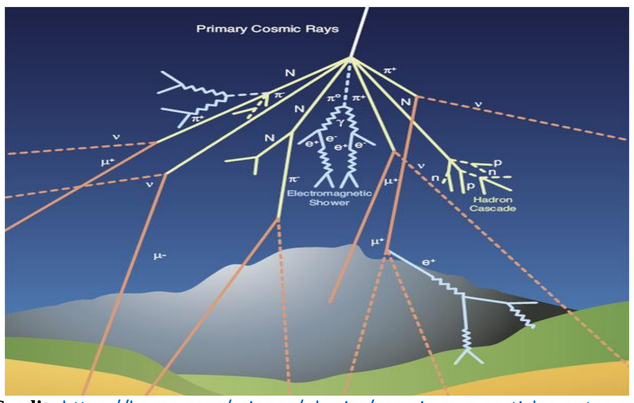
Developing a wireless GPS data collection system for cosmic ray timing

David Jaffe, Ph.D., Physics, Brookhaven National Laboratory, Upton, NY, 11973
David Buitrago, Physics, York College, Jamaica, NY, 11451
Raul Armendariz Ph.D., Physics, Queensborough Community College, Bayside, NY, 11364

7/29/19

Abstract

The cosmic ray detector experiment aims to measure and study muons that reach the earth's surface from incoming cosmic rays when they collide with atoms in the atmosphere (mostly nitrogen) and create a "shower" of particles, including muons. The energy of these cosmic rays can be determined by measuring the area of the shower created by the collision. Through a collective effort from cosmic ray detectors operated at various schools across the country, cosmic ray shower data is being gathered to investigate high energy cosmic rays. Precise time and position information from each detector are needed to measure the cosmic ray showers. One inconveniences of our current system is the long cable between the GPS receiver and our data acquisition (DAQ) board to obtain timing data.



Credit: https://home.cern/science/physics/cosmic-rays-particles-outer-space

Figure 1: Cosmic Showers

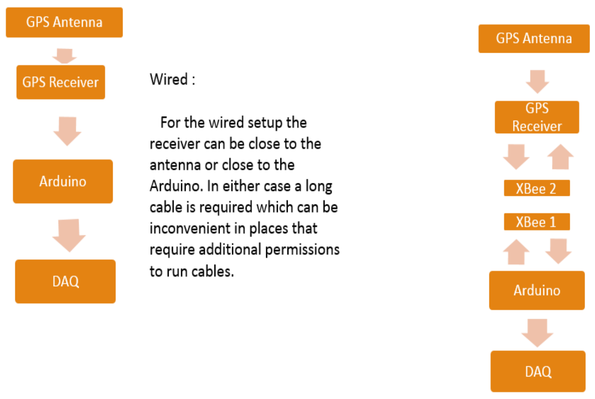
One of the solutions to this problem and the main goal of this summer project is to demonstrate the capability of wireless data communication to replace the long cable. We wish to measure the accuracy of wireless to wired GPS data and determine if our hardware is also limiting the speed at which we receive data.

Introduction

We are building a data acquisition circuit (DAQ) for cosmic ray muon detectors made from plastic scintillators and photomultiplier tubes. Our DAQ will consist of a two-channel DAQ front end circuit (amplifiers, discriminators, coincidence logic, and signal peak detectors) connected to an Arduino microcontroller which digitizes the DAQ FE data and time stamps cosmic ray signals with GPS UTC time.

The GPS antenna signal is sent to a GPS receiver which communicates to our Arduino; the antenna has to be near a window to see satellites however at some detector sites the Arduino will be located far from the window and wireless is desired to avoid having to run long cables which attenuate the signal, and which building rules sometimes do not allow.

We want to compare the GPS timing resolutions for the wired and wireless methods; to do this we measured the number of Arduino clock cycles between successive GPS 1 PPS pulses.



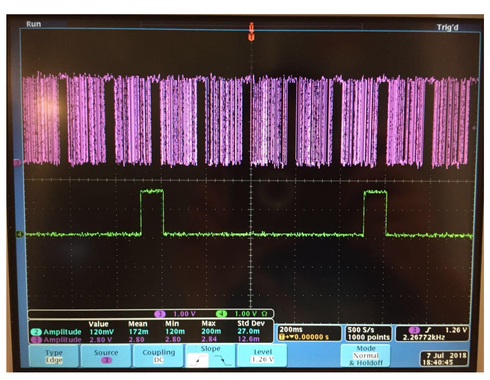


Figure 2: NMEA pulses and 1PPS

Methods

The first part of my project was understanding Arduinos and the code a previous student (Junjie) had already created to operate our Arduino for data acquisition.
Arduinos are powerful microcontrollers that can grab and print serial data, send and receive digital data and convert analog data to digital data.

Our DAQ board will consist of a two channel DAQ front end board connected to the Arduino microcontroller and GPS.

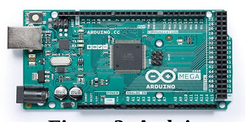


Figure 3: Arduino

Figure 4: X-Bee Series 3

The required specifications for the wireless controller are:

- Have a baud rate of 9600 (matching our GPS receiver).
- Have the correct data transfer protocol to handle fast data transfer.
- Have available resource materials.

The X-Bee fits all of these requirements; other wireless technologies exist but there was not much information on their use with an Arduino; X-Bees seem to be the standard when it comes to wireless communication with Arduino.
X-Bee series 3 has the added benefit of itself being a microcontroller capable of sending messages to a serial port.

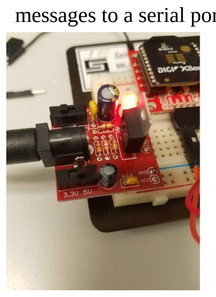


Figure 5: Barrel Head Connector

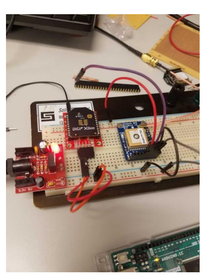


Figure 6: Remote Setup

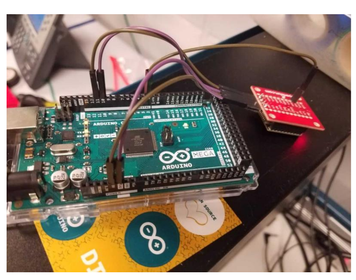


Figure 7: Receiver Setup

Results

Here are previous measurements of the number of clock cycles between successive PPS pulses for a wired connection; the wire from antenna to receiver was 5m, and the network cable from receiver to Arduino was 100 ft. The results show oscillator drift or jitter of about of +/- 20 clock cycles which corresponds to +/- 1.25 microsecond resolution:

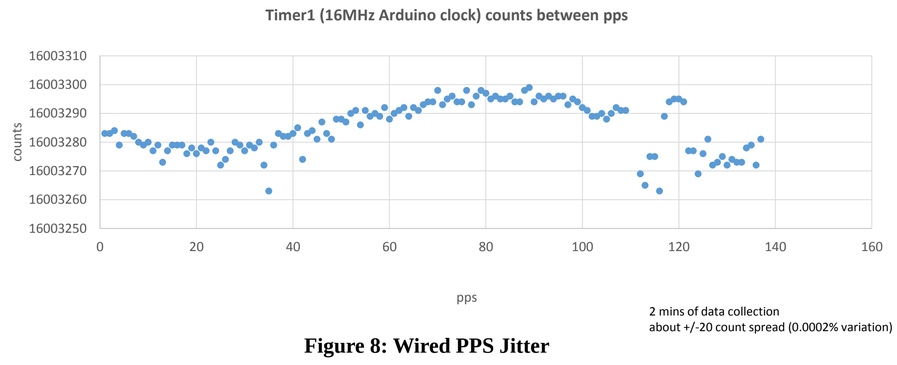


Figure 8: Wired PPS Jitter

The next step was to observe the 1PPS jitter through the wireless XBee; code was written for the Arduino to record the number of clock cycles between successive PPS pulses. The results show inconsistent jitter of about of +/- 10^6 clock cycles or +/- 62.5 ms, which seems to increase over time. However the majority of time the number of cycles land around the expected 16 million.

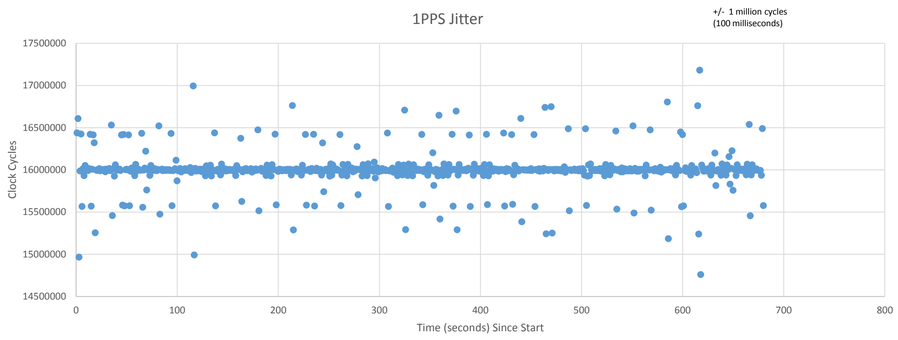


Figure 9: Wireless Jitter

The NMEA data required a different strategy, for the wired setup we had a way of communicating directly with the GPS with the help of libraries written for this specific purpose. For our wireless connection however because the Arduino and the GPS are now separated by X-Bees we cannot configure the receiver and must parse the serial data manually. Of the available NMEA sentences we require only the GPRMC sentence as it holds all of the useful information for timing events, however the script was written to parse any sentence necessary or multiple ones. The figure below shows both the 1PPS clock cycle difference and parsed NMEA data.

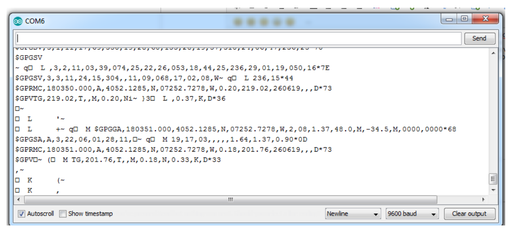


Figure 10: Corrupted Data

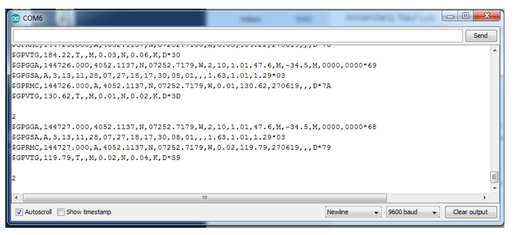


Figure 11: Fixed Data

```
GPRMC,160930.000,A,4052.1312,N,07252.7247,W,0.45,89.66,230719,,,D
16000138.00
GPRMC,160931.000,A,4052.1311,N,07252.7246,W,0.44,52.27,230719,,,D
16001616.00
GPRMC,160932.000,A,4052.1310,N,07252.7246,W,0.36,29.69,230719,,,D
15931655.00
GPRMC,160933.000,A,4052.1311,N,07252.7245,W,0.30,36.34,230719,,,D
16054857.00
GPRMC,160934.000,A,4052.1312,N,07252.7245,W,0.44,48.16,230719,,,D
16000033.00
GPRMC,160935.000,A,4052.1312,N,07252.7243,W,0.19,29.11,230719,,,D
16006212.00
GPRMC,160936.000,A,4052.1313,N,07252.7243,W,0.14,5.73,230719,,,D
15988905.00
```

Figure 12: Complete Data Output

Conclusion

While unsure at the beginning of this project we have shown conclusively that you can add wireless communication between a GPS and Arduino and receive real time data. However, the jitter in the 1PPS shows that while a wireless setup can provide convenience when having to setup the detectors for data taking the accuracy of these readings is above the maximum allowed inaccuracy to time events.
For the serial NMEA sentences they are now properly working and being sent on time, the tolerance for these sentences is much higher, as long as they come in after the first PPS and before the second the can define the time up to the most recent second. In the future, if the rate at which these sentences are sent increases, the jitter will once again become a problem for the sentences as well.

Acknowledgements

This project was supported in part by the U.S. Department of Energy, Office of Science, Office of Workforce Development for Teachers and Scientists (WDTS) under the Science Undergraduate Laboratory Internships Program (SULI).