MUON WIRELESS DATA Tx/Rx

Rehaan Ahmad, Muhammad Shafique and Muhammad Sabieh Anwar

July 29, 2021

1 MUON Counting Sytem

The MUON counting system is based on a transmitter that senses the MUON particles, counts the pulses and transmits the count data using Wi-Fi technology. On the other end wi-fi receiver receives the pulse count and transmits the data over serial port towards the PC.

1.1 Wireless Data Transmission Unit and Data Logger

MUON data transmitter stores pulse count, GPS data and weather parameters in its internal memory. The data could be obtained anytime by connecting the transmitter to the PC through USB at 115200 bps. MUON data transmitter is a dual channel system, consists of C-series silicon photomultiplier sensors. C- Series sensors are used for ultra-fast timing applications with rise time of 300 ps and pulse width of 600 ps and output pulse amplitude of around 25 mV that need further amplification to integrate the system with digital circuit. There are two main PCB each with electronic components. One PCB contains amplifier and monostable vibrator to shape the signal for SiPM sensor that can be detect by microcontroller. The signal is amplified using `AD8055` with a gain of 100 and monostable multi-vibrator (`74LS123`) converts it to TTL signal between 0 to 5V. the second PCB is pulse counter using `Atmega328p` controller, counting the pulses coming in every second and transmits the total counts towards ESP32 module after every minute.

ESP32 is a wi-fi (802.11n) module used in the system as wireless transmitter and could save 3MB of data in its flash using SPIFFS (SPI Flash File System). Pulse count data is sent by Arduino NANO and GPS module transmits Latitude, Longitude, Altitude and time through serial port at 9600 bps. `BMP-180` is a weather sensor used in a system as a slave device and sends pressure (hPa) and temperature (°C) to ESP32 upon receiving command from ESP32 using I2C. The range of data transmission is around 1Km in the line of sight.

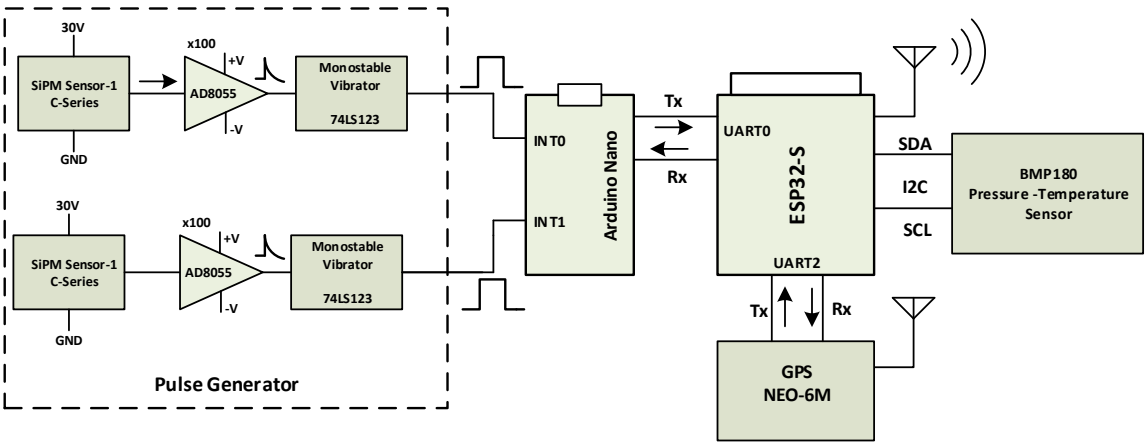

Figure 1: Block diagram of MUON wi-fi data transmission system. Pulse generator is integrated with Atmega controller and ESP32 is for data transmission using wi-fi.

1.2 Silicon Photomultiplier (SiPM) Enclosure

It is a light-tight enclosure to keep the light inside and it also protects against photons entering from outside to avoid environmental noise. The power of SiPM sensor is provided by an external connector and the same connector is used to send the signal from sensor to the amplifier circuit.

1.3 Amplifier

Positive pulses are produced by SiPM after photo-avalanche due to muons is amplified through a non- inverting amplifier with a gain of 100 times as compared to the pulse voltage of around 10-50 mV.

1.4 Monostable Vibrator

The output of the amplifier (trigger input, V_p) is given to the input of monostable vibrator to generate a smooth pulse of constant time period and amplitude (V_o) so it could be detected by a microcontroller for counting the pulses.

1.5 Microcontroller

Arduino NANO board having ATmega328 microcontroller receives pulses from the dual channel MUON pulse generator through the 'INT0' and 'INT1' pins. Pulse counter could be configured (in the firmware) to count the pulses/sec or pulses/min according to the requirement in the field. Arduino board is connected with the ESP32 wireless transmission module through TxD and RxD pins. The rate of data communication between controller and ESP32 is 9600 bps.

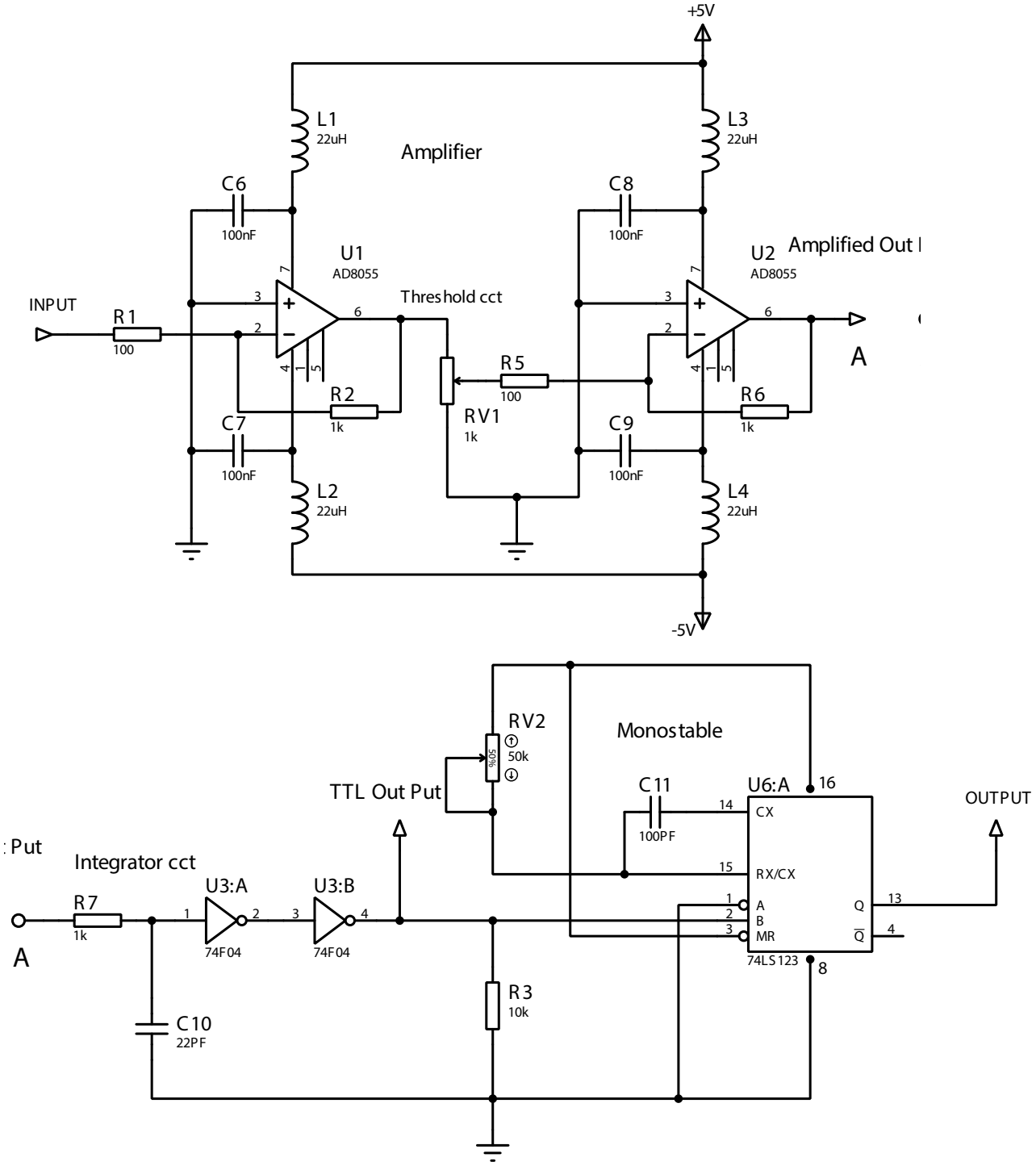

Figure 2: MUON pulse generator circuit, output signal from scintillator is amplified by low noise AD8055, amplified signal is digitized by 74LS123.

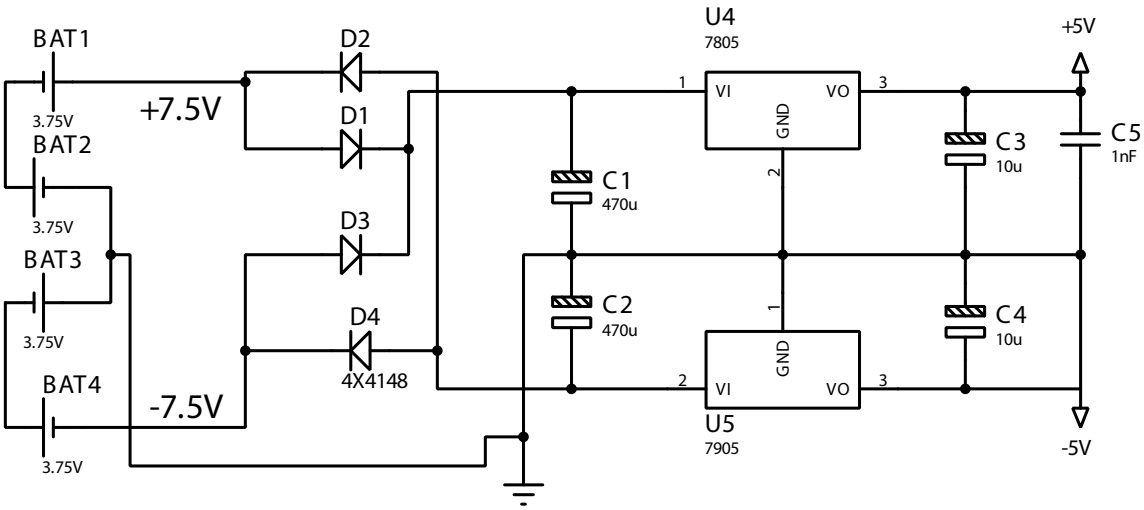

Figure 3: Power supply section of pulse generator circuit.

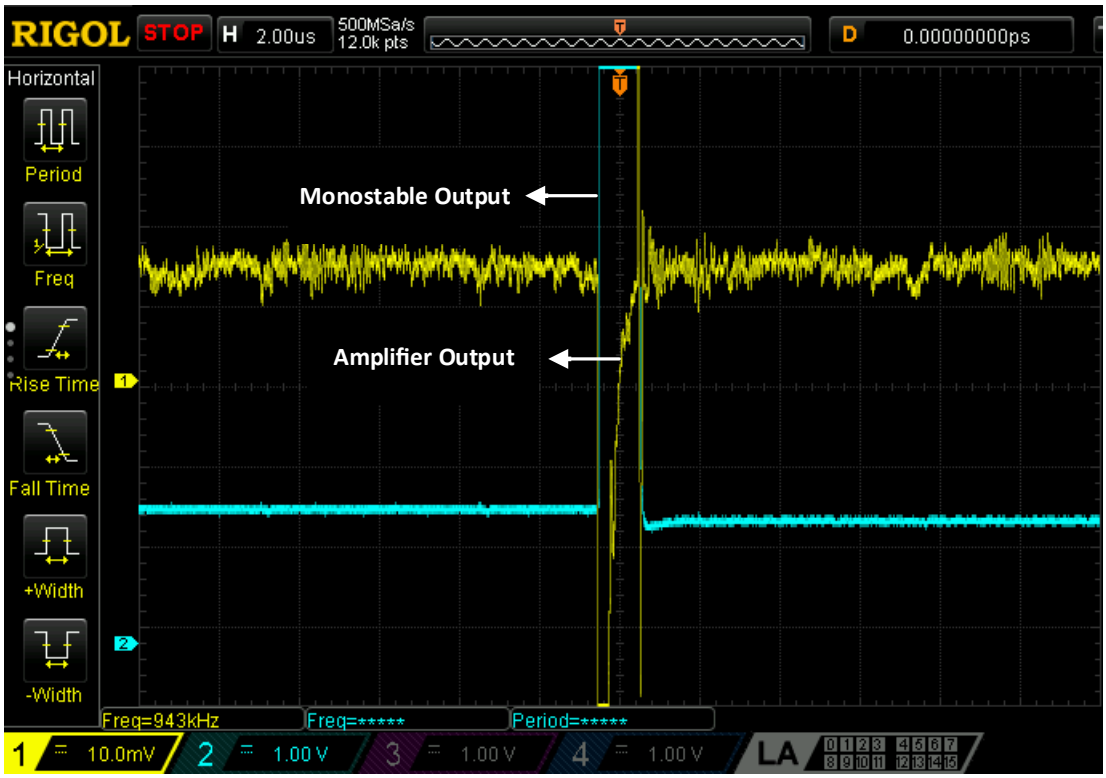

Figure 4: Low Noise amplifier and monostable circuit Outputs

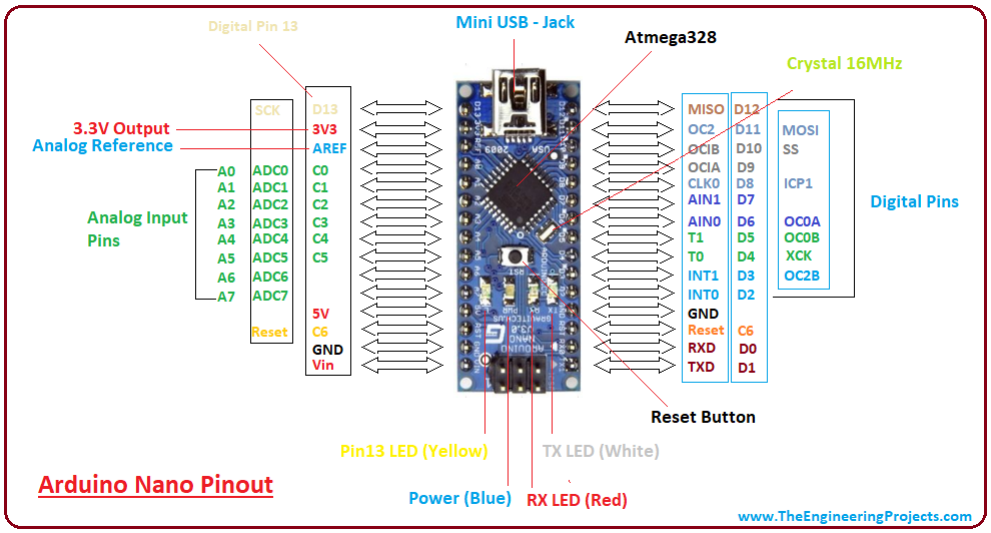

Figure 5: Pinouts for Arduino NANO. [arduino.cc]

Sr NO.	Arduino NANO	ESP32
1	TxD	UART 0 RxD
2	RxD	UART 0 TxD

Table 1: Arduino NANO serial connections with ESP32 Module.

Simplex communication technique is used between Arduino and ESP32 but Duplex connections are made in the circuit between Tx and Rx for the future advancement in the application.

1.6 Wi-fi Transceiver

Node MCU ESP32 V1.2 is the wireless data transmission module used in the circuit to send the following data to the remote receiver.

- Pulse count / min
- Altitude (m)
- Latitude
- Longitude
- Pressure (hPa)
- Temperature (°C)
- Time

Operating frequency range of ESP32 is 2.4 GHz to 2.5 GHz. The pinout of ESP32 is shown in Figure 6.

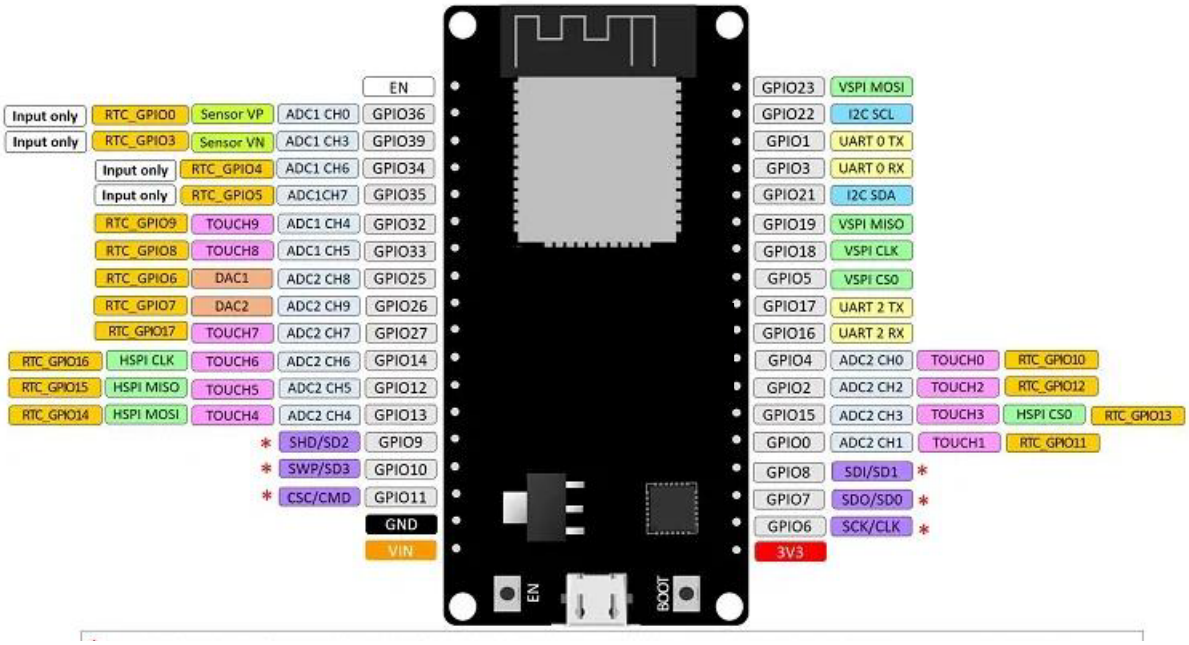

Figure 6: MODE MCU ESP32 pinouts.

Categories	Items	Value
WiFi	Protocols	802.11n up to 150 Mbps
	Frequency Range	2.4-2.5 GHz
Bluetooth	Protocols	Bluetooth v4.2 Br/EDR & BLE
	Radio	NZIF receiver with -98 dbm sensitivity Class-1, class-2 & class-3 transmitter AFH
	Audio	CVSD and SBC
Hardware	Module Interface	SD Card, SPI, UART, SDIO, I2C, LED PWM, Motor PWM, I2S, IR
	On-chip Sensor	GPIO, ADC, DAC, LNA, Capacitive Touch Sensor
	Operating voltage	3.0 – 5.0 V
	Operating current	Average Value: 80 mA
	Operating temperature range	-40 – 125 C
	Ambient temperature range	Normal temperature
Software	Wi-Fi Mode	Station / SoftAP / P2P
	Security	WPA / WPA2 / WPA2-Enterprise / WPS
	Encryption	AES / RSA / ECC / SHA
	Firmware upgrade	UART Download / OTA /
	Software Development	Support cloud server development SDK for custom firmware development
	Network Protocols	IPv4, IPv6, SSL, TCP, UDP, HTTP, FTP
	User Configuration	AT instruction set, cloud server, Android

Table 2: Properties of ESP32 module.

Sr NO.	GPS	ESP32
1	TxD	UART 2 RxD
2	RxD	UART 2 TxD

Table 3: ESP32 to GPS serial connections.

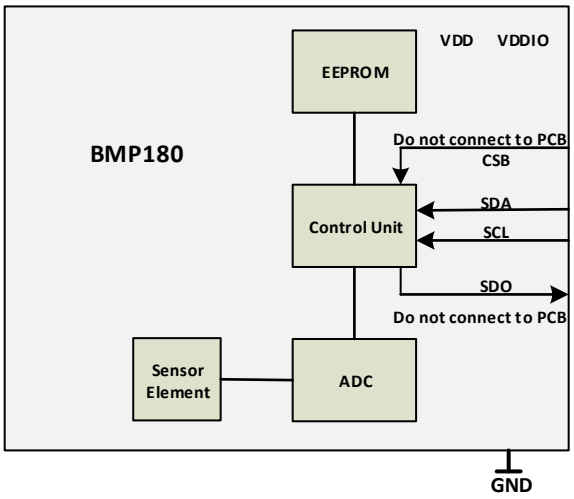

Figure 7: Internal block diagram of temperature and pressure sensor `BMP180`.

1.7 GPS Module

GPS module "NEO–6M–0–001" is connected with ESP32 module through UART. GPS module continuously sends the data that contains the information regarding altitude, latitude, longitude, time etc. over serial port. The firmware inside the ESP32 module will extract the desired data from GPS data, inserts the delimiters and makes a packet and transmits this data to ESP32 receiver unit.

1.8 GPS and ESP32 Module Connection

ESP32 UART2 is used for communication with GPS module, although communication is one way but both serial transmit and receiver are connected in the circuit. GPS module is only transmitting the data and ESP32 is receiving the GPS data through UART_2 RxD.

1.9 Sensor Module

`BMP-180` is directly connected to the microcontroller through I2C. Pressure and temperature data is stored in the EEPROM and this is obtained through I2C interface. Microcontroller gets the data from the sensor module after a pre-set time of pulse counter.

State	Voltage (V)	Current (mA)	Power (W)
Start-up	5	200	1
Data Transmission	5	500	2.5

Table 4: Voltage and current rating of complete MUON unit.

1.10 Power Source

A standalone power source is required for the system and is made by connecting four (1800 mAh, 3.7 V) cells, because the same power source is also used to supply the power to the pulse generator circuit at 7.4 V, 0 V, −7.4 V. Power requirement of wireless data transmitter board is as follows

1.11 Wireless Data Receiver Unit

The same EPS32 v1.2 is used as wireless data receiver for data reception. It is connected to the computer through serial port and sends the data to PC upon reception from transmitter. MUON data receiver is with GUI (graphical user interface) having all the fields related to receive incoming data and visual description along with data rate.

GUI contains separate fields for Time, Latitude, Longitude, Altitude, Pressure and Temperature. GUI also saves complete data present in 'Serial Data Monitor' in a '.txt' file upon pressing 'SAVE DATA' button.

MUON data receiver GUI application could also be installed in PC using .exe file.

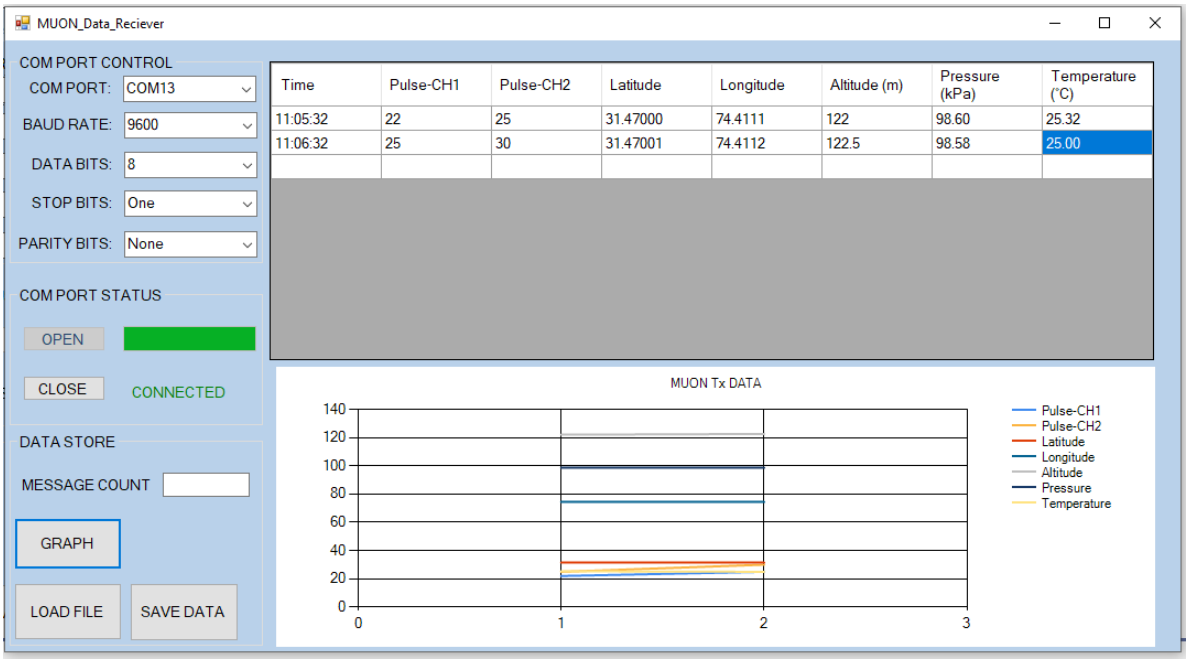

Figure 8: Graphical user interface of Muon data receiver.

```
Time        Pulse-CH1    Pulse-CH2      Latitude      Longitude     Altitude    Pressure    Temperature
15:30:58  ,   63      ,      72    ,    31.47000  ,   74.4111  ,    122.51  ,   98.066   ,    27.14
15:31:58  ,   50      ,      61    ,    31.47000  ,   74.4111  ,    122.51  ,   98.055   ,    27.07
15:32:58  ,   45      ,      62    ,    31.47000  ,   74.4111  ,    122.51  ,   98.055   ,    27.15
15:33:58  ,   50      ,      68    ,    31.47000  ,   74.4111  ,    122.51  ,   98.044   ,    27.22
15:34:58  ,   48      ,      65    ,    31.47000  ,   74.4111  ,    122.51  ,   98.044   ,    27.23
15:35:58  ,   47      ,      63    ,    31.47000  ,   74.4111  ,    122.51  ,   98.044   ,    27.38
15:36:58  ,   63      ,      63    ,    31.47000  ,   74.4111  ,    122.51  ,   98.066   ,    27.14
15:37:58  ,   50      ,      61    ,    31.47000  ,   74.4111  ,    122.51  ,   98.055   ,    27.07
15:38:58  ,   45      ,      62    ,    31.47000  ,   74.4111  ,    122.51  ,   98.055   ,    27.15
15:39:58  ,   50      ,      68    ,    31.47000  ,   74.4111  ,    122.51  ,   98.044   ,    27.22
15:40:58  ,   48      ,      65    ,    31.47000  ,   74.4111  ,    122.51  ,   98.044   ,    27.23
15:41:58  ,   47      ,      63    ,    31.47000  ,   74.4111  ,    122.51  ,   98.044   ,    27.38
```

Figure 9: Data format in .txt file.

The data will be saved in the text file in the following format

References

[1] https://www.theengineeringprojects.com/2018/06/introduction-to-arduino-nano.html

[2] https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

[3] https://acoptex.com/project/258/basics-project-053b-neo-6m-gy-gps6mv2-gps-module-at-lex-c/