**Effects of a Total Solar Eclipse on High Altitude Muon Production**

J. Fender, J. Morse, and J. Ringler, Faculty Advisor: M. Semak

*Department of Physics and Astronomy, University of Northern Colorado, Greeley, CO 80639*

April 10, 2018

Abstract

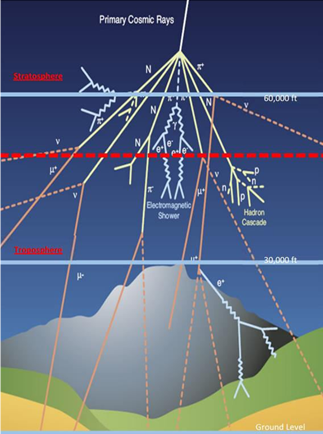
We have developed a method for the detection of cosmic ray muons as a function of altitude in an effort to determine the effect of a total solar eclipse on muon production. The detector is part of a self-contained autonomous payload that is carried to altitude via a weather balloon. The muon detector consisted of three Geiger-Muller tubes connected in a coincidence circuit. This system, along with various other environmental sensors and a GPS unit, is controlled by an onboard Arduino Mega microcontroller. It was previously theorized that the majority of the cosmic rays that reach the atmosphere originate from extrasolar sources. Because of this, it was predicted that a total solar eclipse would have a negligible effect on cosmic ray muon production in the atmosphere. The data collected from this detector suggest this prediction to be true, collected from a flight launched during the total solar eclipse on August 21, 2017, near Guernsey, WY.

*I. Introduction*

**Muons**

From 1911-1913, the experiments performed by Victor Hess led to the discovery of cosmic rays, for which he received the Nobel Prize in 1936 [1]. Originally thought to be some form of electromagnetic radiation, it was later revealed that cosmic rays are almost exclusively high-energy charged particles that originate outside of Earth’s atmosphere and bombard the planet from all directions in space [2]. Though the sun is a source of some of these cosmic rays, it’s believed that most of these particles originate from extrasolar sources such as mechanisms observed in the center of active galactic nuclei, as well as in the catastrophic events following supernovae [3]. Cosmic rays are composed of particles in similar proportion to that found throughout the galaxy. Approximately 89% of the particles that make up the rays are hydrogen nuclei, protons, 10% are helium nuclei, and 1% are nuclei from heavier elements [4].

When very high energy cosmic rays enter Earth’s atmosphere, they interact and collide with atmospheric particles, through which large “showers” of secondary particles are generated [5]. These collisions occur primarily in the stratosphere, which is a layer of the Earth’s atmosphere that extends radially from approximately 9 to 31 miles above the planet’s surface [6]. The predominant mechanism through which atmospheric ionizing radiation is produced is the collision between cosmic rays and atmospheric molecules. As the cosmic ray penetrate more deeply into Earth’s atmosphere, the increasing density of molecules in the atmosphere greatly increases the probability of additional collisions. These later collisions then cause product particles to disperse and create larger showers [see Figure 1].



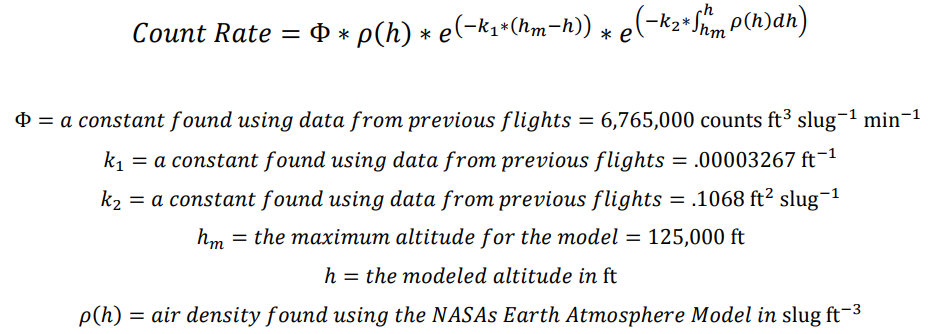
*Figure 1: A diagram depicting an idealized shower of particles generated by collisions of cosmic rays with atmospheric nuclei.*

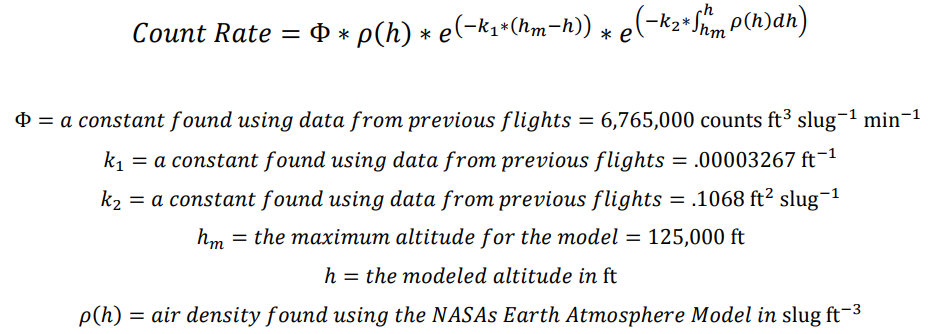
*Source: https://wipac.wisc.edu/deco/project*

When these showers occur, most products are pions which decay rapidly and subsequently form even larger showers of muons, neutrinos, and gamma rays [7]. Of these particles, the muon is most likely to reach the planet’s surface before decaying into other particles. The muon travels at large speeds approaching that of light, which means that relativistic effects on its trajectory are significant. Given these effects, some of the muons generated in very high-altitude collisions can still reach the Earth’s surface despite its short lifespan of 2.20 microseconds [8].

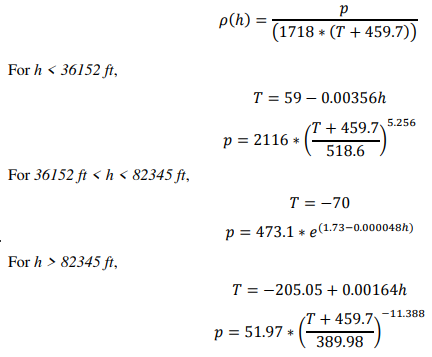
**The Palmer Model**

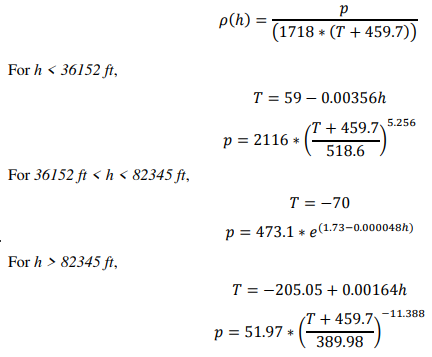
Dr. G. Michael Palmer, a professor at West Virginia University, developed a mathematical model that describes the muon count rate as being proportional to the atmospheric density at a given altitude [9]. The Palmer Function is given below.

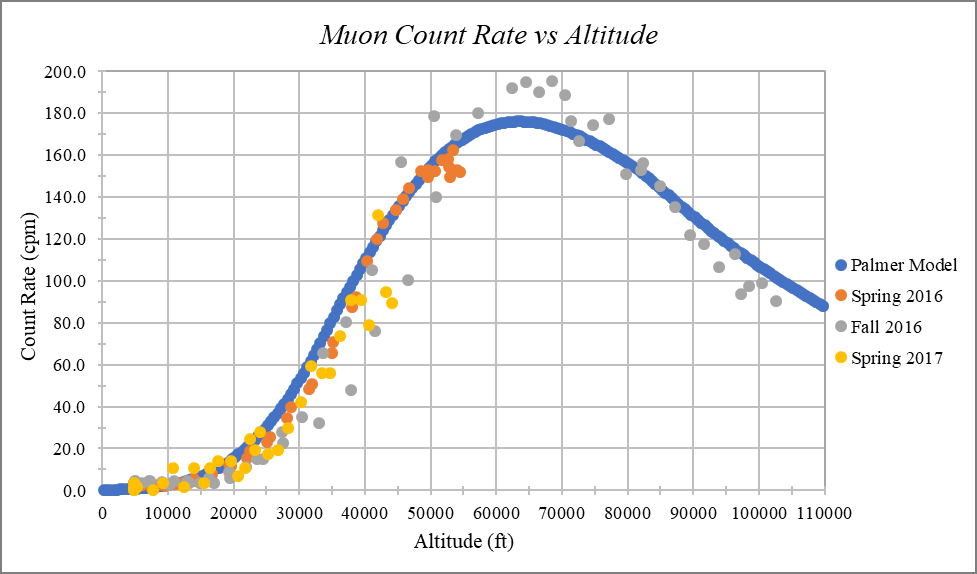


Where:

NASA’s Earth Atmosphere Model [10]:







*Figure 2: A graph displaying the data from the Fall 2016 DemoSat launch with the predicted count rate from the Palmer model.*

**Pfotzer Maximum**

While many of the muons can reach the Earth’s surface, a larger portion of them decays before traveling all the way to the ground. This can be seen in an increase in the muon count rate as the payload increases in altitude and can be shown with the Palmer Model (Fig. 2). The count rate reaches a maximum when the payload reaches an altitude between 60,000 and 70,000 feet. This maximum count rate is known as the Pfotzer Maximum [11]. This threshold is caused by the decrease in the atmospheric density which consequently decreases the probability of interactions between cosmic rays and atmospheric particles.

**Solar Eclipses**

Solar eclipses occur when the Moon lines up directly between the Earth and the Sun, casting a showdown on a portion of the Earth. The shadow is made up of two parts, the umbra where all of the Suns light is blocked and the penumbra where only a portion is blocked [12]. The short period of time, two to three minutes, when the Sun is blocked is known as totality. In addition to blocking the light from the Sun, the Moon also any cosmic rays created by the Sun from reaching the part of the Earth cast in shadow.

**Expected Results**

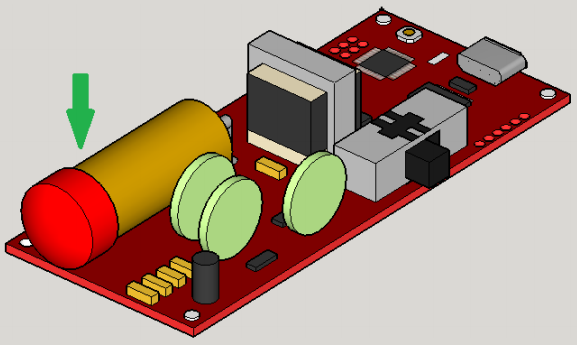
It is theorized that the majority of the cosmic rays incident on the earth originate from sources outside of the solar system [13]. These sources include such objects as active galactic nuclei, remnants of supernovae, and many others. While the sun certainly produces these cosmic rays in addition to the extrasolar contributions, but at a much lower frequency. Therefore, it is expected that there will be no statistical change in the muon count rate during the eclipse totality.

*II. Materials and Methods*

For this experiment, we used a series of Geiger-Muller tubes in a coincidence circuit, where each tube was covered with a thin sheet of lead. The lead was added to assist in filtering false positive detections that may be caused by charged particles other than a muon. Additional sensors were used to maintain a steady temperature inside of the payload, collect external temperature data, internal pressure data, and GPS coordinates.

**Geiger-Müller Tubes**

The primary component of a Geiger counter is the Geiger-Müller tube [see Figure 3]. The operation of Geiger-Müller tubes is based on phenomena observed during the photoelectric effect; that being the ejection of valence electrons in metals due to interactions with incident light [14]. The casing of the tube functions as a cathode, while a thin wire through the center of the tube functions as an anode [15]. The potential difference between the casing and the wire is very large to be able to adequately respond to small charges.

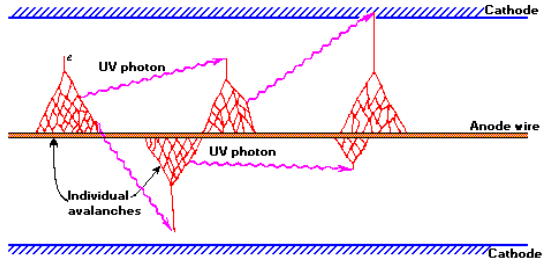


*Figure 3: A 3D rendering of a Geiger counter. The green arrow points to the Geiger-Müller tube.*

The tube is filled with an inert gas to avoid electronic interference between the atoms in the gas and the device’s anode and cathode. When high-energy radiation enters the chamber, it ionizes the inert gas, releasing an electron with high kinetic energy. The freed electron then collides with other nearby atoms, ionizing them thus freeing additional electrons to participate in further collisions. This free electron cascade is called a Townsend avalanche (Fig. 4). This “avalanche” of electrons is then attracted to the positively charged anode, which simultaneously repels the positive ions of the inert gas. The avalanche is stopped by removing the collected excess electrons, process known as a Geiger-Müller discharge. This discharge creates a strong electronic pulse to a detector, which then produces a count.

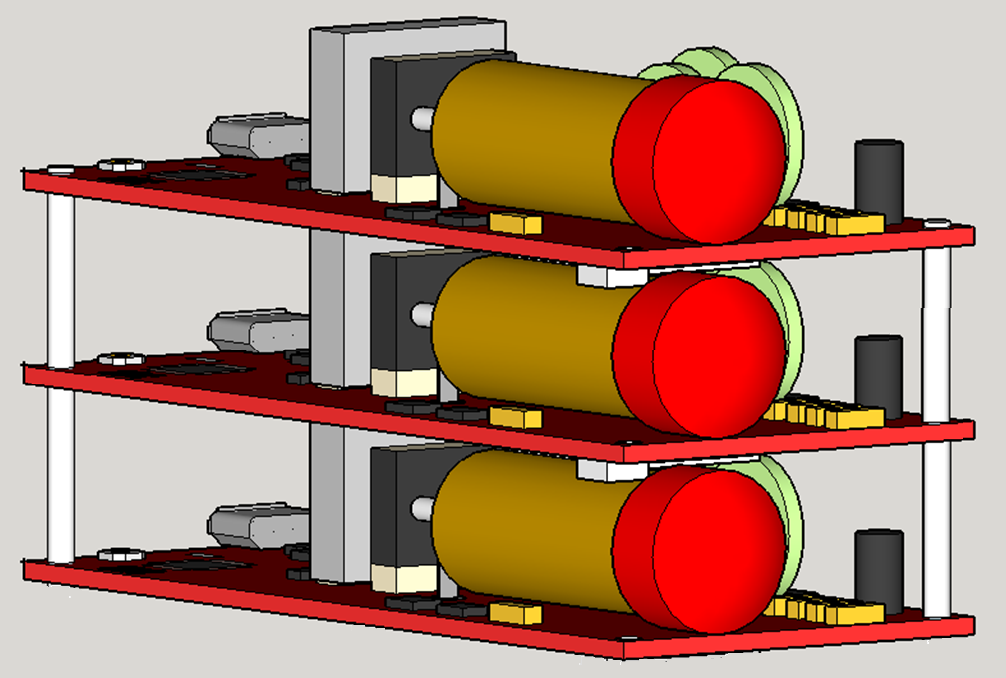
A major disadvantage of the use of Geiger–Müller tubes in the detection of particular particles is that any charged particle will trigger a “count” on the device. To eliminate as many of the false counts as possible, thin sheets of lead, 1/8 inch-thick, were added above each of the tubes. The sheets of lead should act as filtering agents by attenuating the radiation caused by particles other than the particle of interest, in this case muons. Muons have much higher energy than other particles and are not as easily deflected by lead.

*Figure 4: A diagram of a detection event inside a Geiger-Muller tube. The purple lines are incoming ionizing radiation, and the red web-like lines are the resulting Townsend Avalanches. Their collection on the anode is discharged out of the chamber to be registered as a “count.”*



**Coincidence Detection**

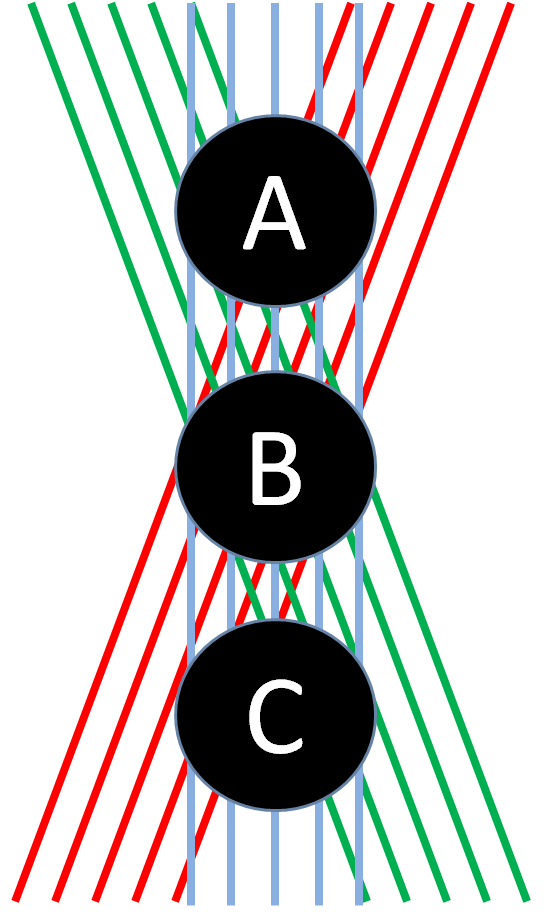
Even with the use of lead sheets and the scintillation detector, there is still a possibility for the misidentification of other particles as cosmic ray muons. As a further method to filter out as much of the noise as possible, a coincidence detection circuit is used. A coincidence detection circuit will only register a count when at least two of the detectors are triggered virtually simultaneously. For this payload, the coincidence circuit consists of an “AND” statement built into the payload code for the Arduino Mega [see Appendix 4]. This method of coincidence counting is dependent on the time resolution of the Arduino Mega which has an accuracy down to 1.50 microseconds.



*Figure 4: A 3D rendering of the complete integrated detection device*

For the coincidence circuit to work properly, the three Geiger counters must be stacked vertically [see Figure 4]. In this configuration, a coincidence count will consist of a positive detection signal from any two of the Geiger–Müller tubes simultaneously. When two counters are used, only particles that are generated in a small window directly above the payload will pass through both detectors. This causes a large number of particles to not be detected. The use of a third Geiger–Müller tube increases this window slightly while still filtering out noise [see Figure 5]. This method of detection should provide greater certainty in distinguishing the muons from other forms of ionizing radiation.

Figure 5. A diagram illustrating some of the possible paths for muons through at least one of the Geiger– Müller tubes indicated by circles A, B, and C.



**Arduino Mega and Other Sensors**

The rest of the payload consists an Arduino Mega microcontroller, a barometer, a thermometer, a GPS unit, and an SD card data logger. The Arduino Mega is an integrated microcontroller unit that will control the sensors and pass the collected information to the data logger. The data logger is a micro SD card “shield”, a preassembled board that can be stacked directly onto the Mega while still providing access to all the microcontroller’s input and output pins. The barometer and thermometer are built onto a single chip and provide data on the integrity of the Geiger counter pressure vessel. A second temperature sensor is used in conjunction with a transistor to deliver power to a resistor, which is used as the onboard heater. When the temperature inside of the payload drops below 0° C, the heater is switched on. A third temperature sensor collects data on the external temperature of the payload for the duration of the flight. Finally, a GPS unit is utilized to confirm the location of the payload, and to verify timestamps on the data collected. This is necessary to identify exactly when totality occurs in the data.

**Payload Construction**

The basic skeletal structure of the payload was designed several semesters ago by the UNC DemoSat students. The primary goals of the design were that it had to be modular, easy to modify, easy to manufacture, and would prove to be reliable. During the last several high- altitude balloon launches, this design has been used and has met these goals. Because of this, the original design was to be utilized again. Therefore, the major focus of the project, from a structural stand point, was the development of a more advanced payload system for the sensitive instrumentation.

During testing of the electronics, it was found that the high voltage required by the Geiger–Müller tubes would cause electrical arcing when exposed to low pressures, like those that occur at high altitude. After several failed attempts to shield the electronics from arcing, the Geiger–Müller tubes had to be sealed inside a large PVC pipe to maintain ground level pressure during the flight.

The PVC pipe, while fulfilling its intended purpose, was extremely heavy and a poor use of space inside the frame. From the onset of this flight, the development of a properly sealed vessel for the instruments was a top priority. Utilizing computer aided drafting, and 3D printed ABS plastic, several prototypes were produced and tested. While sounding like a simple and straight forward project, an extensive list of obstacles was encountered. These included; mitigating layer separation of the 3D print, determining the best type of sealants to be used, and identifying assembly techniques that would reduce the possibility of construction errors. A box was designed that was believed to be able to maintain ground pressure. During testing, the design was placed in a vacuum chamber that produced the similar pressures to those encountered at an altitude of approximately 250,000 feet above sea level. During this testing, the box was able to retain an internal pressure to those found at about 6,000 feet above sea level. The box was easily integrated into the existing frame, and the resulting payload mass was significantly reduced over using PVC pipe.

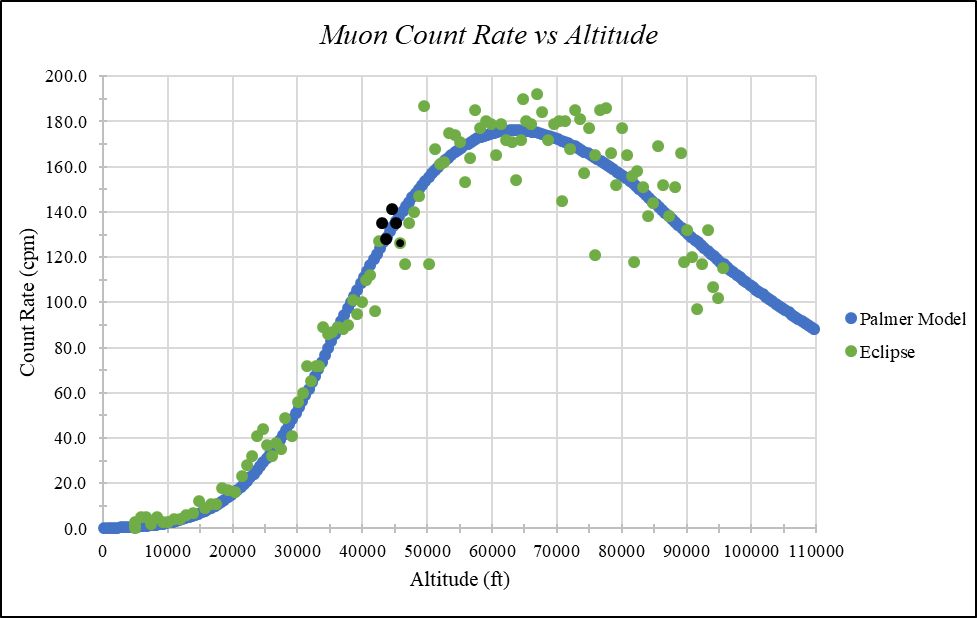
**Balloon Provider**

The payload was launched on a high-altitude helium balloon provided by Edge of Space Sciences (EOSS). In addition to providing the balloon, they attach their own small payload that transmitted live GPS data including latitude, longitude, and altitude. The balloon was launched near Guernsey. WY. Approximately 15 minutes before the eclipse began.

*III. Results*

For this flight the balloon was launched approximately 15 minutes before the start of the eclipse and approximately 45 minutes before totality was expected to occur at the launch location. During the flight prevailing winds pushed the payload along the path of totality giving us slightly more time in the Sun’s shadow to collect data. Unfortunately, the payload did not reach the 60,000-70,000 foot during totality.

*Figure 6. A graph displaying the data from the 2017 Eclipse flight in green with the predicted count rate from the Palmer model in blue. The black points indicate when the payload was in totality.*



IV. Analysis

V. Acknowledgments

As a team, we would like to thank several individuals whose guidance was irreplaceable throughout the course of this experiment. Dr. C Galovich, Dr. C. Kuehn, and Dr. M. Semak, professors at the University of Northern Colorado’s Department of Physics and Astronomy, whose supervision and expertise were invaluable. The volunteers at Edge of Space Sciences, Inc. that facilitated the actual launch of our payload by providing the balloon and continuously tracking its position during the flight for successful recovery. Finally, we would like to thank Chris Koehler, Bernadette Garcia, and the rest of the staff at the Colorado Space Grant Consortium for their guidance and support throughout this project.

VI. Works Cited

1. Hess, V. F., & Anderson, C. D. (1965). Victor F. Hess - Biographical. In *Nobel lectures physics 1922-1941: including presentation speeches and laureates' biographies*. Amsterdam: Elsevier. Retrieved from http://www.nobelprize.org/nobel\_prizes/physics/laureates/1936/hess-bio.html
2. Mewaldt, R. A. (1996). Cosmic Rays. Retrieved from http://www.srl.caltech.edu/personnel/rmewaldt/cos\_encyc.html
3. Ibid.
4. Ibid.
5. Ibid.
6. Zell, H. (Ed.). (2015, July 30). Earth's Atmospheric Layers. Retrieved from https://www.nasa.gov/mission\_pages/sunearth/science/atmosphere-layers2.html
7. Mewaldt, R. A. (1996). Cosmic Rays. Retrieved from http://www.srl.caltech.edu/personnel/rmewaldt/cos\_encyc.html
8. Nave, C. R. (n.d.). Atmospheric Muons. Retrieved from http://hyperphysics.phy- astr.gsu.edu/hbase/Particles/muonatm.html
9. Brewer, J. H., III, Hansford, N. J., Hill, J. C., Palmer, M. G., & Kuhlman, J. M. (2007). *High Altitude Cosmic Radiation (HACR) Detector Payload Summary and Flight Science Report* [Scholarly project]. In *The High Altitude Student Platform*. Retrieved from http://laspace.lsu.edu/hasp/groups/2007/science\_report/Payload\_08/Flight\_Report.pdf
10. Benson, T. (Ed.). (2014, July 12). Earth Atmosphere Model. Retrieved from https://www.grc.nasa.gov/www/k-12/rocket/atmos.html
11. Brewer, J. H., III, Hansford, N. J., Hill, J. C., Palmer, M. G., & Kuhlman, J. M. (2007). *High Altitude Cosmic Radiation (HACR) Detector Payload Summary and Flight Science Report* [Scholarly project]. In *The High Altitude Student Platform.* Retrieved from http://laspace.lsu.edu/hasp/groups/2007/science\_report/Payload\_08/Flight\_Report.pdf
12. Pogge, R. (2010, June 3). Lecture 9: Eclipses of the Sun & Moon. Retrieved April 1, 2018, from http://www.astronomy.ohio-state.edu/~pogge/Ast161/Unit2/eclipses.html
13. Mewaldt, R. A. (1996). Cosmic Rays. Retrieved from http://www.srl.caltech.edu/personnel/rmewaldt/cos\_encyc.html
14. The Geiger Counter and Counting Statistics. (1994). Retrieved from http://www.its.caltech.edu/~derose/labs/exp2.html
15. Rex, A. and Thornton, S. (2013). *Modern Physics for Scientists and Engineers, 4th Edition*. Cengage Learning

VII. Appendices

Appendix 1: Parts and Price List

Appendix 2: Arduino Code

//Libraries

#include <SPI.h>

#include <Wire.h>

#include <SD.h>

#include <TinyGPS++.h>

#include <SoftwareSerial.h>

#include <OneWire.h>

#include <DallasTemperature.h>

#include <SparkFunMPL3115A2.h>

//Setup OneWire temperature sensors

OneWire e\_one\_wire(3);

OneWire i\_one\_wire(2);

DallasTemperature external(&e\_one\_wire);

DallasTemperature internal(&i\_one\_wire);

float external\_temp;

float internal\_temp;

//Setup GPS unit

TinyGPSPlus GPSdata;

float LAT;

float LNG;

float ALT;

int SAT;

int HRS;

int MIN;

int SEC;

//Setup pressure sensor

MPL3115A2 myPressure;

float pressure;

float temperature;

//Setup GPS serial

SoftwareSerial gpsSerial(69, 68); //RX: 69=A15, TX: 68=A14

//Setup SD card file namer

char filename[24];

//Keep track of elapsed time

unsigned long timeStamp;

//Counter variables

int timer;

int start;

int GC1\_value;

int GC2\_value;

int GC3\_value;

int GC\_max = 0;

unsigned long count\_GC1;

unsigned long count\_GC2;

unsigned long count\_GC3;

unsigned long count\_Muon;

//Store all of the data in a string and print all at once

String dataString;

void setup() {

//Set the heater pin

pinMode(4, OUTPUT);

//Start RX & TX ports

Serial.begin(38400); //Serial Monitor

Serial1.begin(9600); //Giger Counter 1

Serial2.begin(9600); //Giger Counter 2

Serial3.begin(9600); //Giger Counter 3

gpsSerial.begin(4800); //GPS

//Start OneWire ports

external.begin(); //External temperature sensor

internal.begin(); //Internal temperature sensor

//Start SDA & SCL ports

Wire.begin();

//Configure pressure sensor

myPressure.begin();

myPressure.setModeBarometer();

myPressure.setOversampleRate(128);

myPressure.enableEventFlags();

//Setup the SD sheild

pinMode(8, OUTPUT);

Serial.print("Initializing SD card...");

delay(1000);

if (!SD.begin(10, 11, 12, 13)) {

Serial.println("Card failed or not present");

return;

}

//Initialize the SD card and find the name of the last file, then create a new sequential file

Serial.println("card initialized.");

delay(500);

int n = 0;

snprintf(filename, sizeof(filename), "log\_%03d.txt", n);

while(SD.exists(filename)) {

n++;

snprintf(filename, sizeof(filename), "log\_%03d.txt", n);

}

//Begin a new data run

File dataFile = SD.open(filename,FILE\_WRITE);

if (dataFile) {

Serial.print("Begin New Data Run: #");

Serial.println(n);

dataFile.close();

}

else {

Serial.println("error opening datalog.txt");

}

delay(500);

Serial.println("Timer, Hr:Min:Sec, Sat, Latitude, Longitude, Altitude, Pressure, Internal, External, GC-1, GC-2, GC-3, Muon");

}

void loop() {

String dataString = "";

GPSdelay(1000);

{//Timestamp

timeStamp = millis();

timeStamp = timeStamp/1000;

dataString += String(timeStamp) += ", ";

}

{//GPS

SAT = GPSdata.satellites.value();

LAT = GPSdata.location.lat();

LNG = GPSdata.location.lng();

ALT = GPSdata.altitude.feet();

HRS = GPSdata.time.hour();

MIN = GPSdata.time.minute();

SEC = GPSdata.time.second();

dataString += String(HRS) += ":";

dataString += String(MIN) += ":";

dataString += String(SEC) += ", ";

dataString += String(SAT) += ", ";

dataString += String(LAT, 6) += ", ";

dataString += String(LNG, 6) += ", ";

dataString += String(ALT, 2) += ", ";

}

{//Pressure inside of gieger counter box

pressure = myPressure.readPressure();

dataString += String(pressure, 2)+= ", ";

}

{//Temperature inside of gieger counter box

temperature = myPressure.readTemp();

dataString += String(temperature, 2)+= ", ";

}

{//Temperature inside of the payload

internal.requestTemperatures();

float internal\_temp = internal.getTempCByIndex(0);

dataString += String(internal\_temp, 2)+= ", ";

//Heaters

if (internal\_temp < 0) {

digitalWrite(4, HIGH);

}

else {

digitalWrite(4, LOW);

}

}

{//Temperature outside of the payload

external.requestTemperatures();

float external\_temp = external.getTempCByIndex(0);

dataString += String(external\_temp, 2)+= ", ";

}

{//Muon Counter

//Setup counters

start = millis();

timer = start;

count\_GC1 = 0;

count\_GC2 = 0;

count\_GC3 = 0;

count\_Muon = 0;

while ((timer-start) <= 30000) {

//Geiger Counter readers

GC1\_value = Serial1.read();

GC2\_value = Serial2.read();

GC3\_value = Serial3.read();

if (GC1\_value > GC\_max) {

count\_GC1++;

}

if (GC2\_value > GC\_max) {

count\_GC2++;

}

if (GC3\_value > GC\_max) {

count\_GC3++;

}

if ((GC1\_value > GC\_max && GC2\_value > GC\_max) || (GC2\_value > GC\_max && GC3\_value > GC\_max)) {

count\_Muon++;

}

timer = millis();

}

dataString += String(count\_GC1 \* 2) += ", ";

dataString += String(count\_GC2 \* 2) += ", ";

dataString += String(count\_GC3 \* 2) += ", ";

dataString += String(count\_Muon \* 2);

}

{//Print data to Serial and SD card

File dataFile = SD.open(filename, FILE\_WRITE);

if (dataFile) {

Serial.println(dataString);

dataFile.println(dataString);

dataFile.close();

}

else {

Serial.println("error opening datalog.txt");

Serial.println(dataString);

}

}

}

//Extra functions

//Feeds the gps data from the receiver to the arduino, refreshes at 1Hz which adds 1 second to the cycle time

static void GPSdelay(unsigned long ms) {

unsigned long delay\_start = millis();

do {

while (gpsSerial.available())

GPSdata.encode(gpsSerial.read());

} while (millis() - delay\_start < ms);

}