Cosmic Ray Project
Oluwafemi Osidipe, Drew Thomas, and Elizabeth Weber
Advisors: Dr. Julia Thompson and Dave Kraus
June 11, 2001
REUP-FOM/SIUE

1 Background

Cosmic rays are extremely high-energy charged particles which can travel close to the speed of light with energies ranging from 1GeV to beyond 10^{20}eV. The lower energy primary particles are believed to originate in solar flares, and the higher energy particles are suspected to be of extra-galactic origin. Large numbers of particles detected in coincidence at the earth's surface indicate a high energy primary. One interest in this project is to explore possible anomalies at high energies [1].

It is believed that some cosmic rays receive their energy from the explosion of supernovas or from the collision of two supernovas, or two black holes. There are two basic divisions of cosmic rays: 1) primaries, which consist of electrons, protons, helium, and the nuclei of carbon, oxygen, and iron; and 2) secondaries. When high-energy primary particles undergo collisions with atoms in the atmosphere, they produce a cascade of secondary particles that shower down through the atmosphere to the Earth's surface. Secondary particles consist of the nuclei of lithium, beryllium and boron, as well as mesons, muons, neutrinos, photons, antiprotons, and pions, which quickly decay to produce muons, neutrinos, and gamma rays, as well as electrons and positrons produced by muon decay and gamma ray interaction with atmosphere atoms.

Speculation about the existence of cosmic rays arose around the early 1900's. When studying electrical conductance in air and closed vessels, scientists noticed that ionization was present in the closed vessels. This observation was averred when they saw that charged leaves in an electroscope gradually lost their charge. An electroscope works by sending the same charge up both leaves. As shown in Figure 1, because of the effects like charges have on one another, the leaves repel each other. However, as they lose their charge, the leaves droop back down together. In order for the leaves to lose their charge, the particles, for example, a proton, must be attracting the negative charge of the electron from somewhere. From this we can speculate that wherever the protons are attracting electrons, the source must be energetic enough to penetrate the atmosphere and the glass chamber that encloses the leaves.

Figure 1: Electroscope: PDF file on siue.edu/~juthomp/2001docs.

In 1912, Victor F. Hess discovered cosmic rays. He found that as he traveled higher into the sky in a balloon, an electroscope discharged more rapidly. He credited this to some source of radiation entering the atmosphere from above, which caused ionization in the air. In the mid 1900's, before advanced particle accelerators, cosmic rays served as the source for high-energy physics investigation. This led to the discovery of many other sub-atomic particles; such as the muon. Today, the focus of cosmic ray research has been directed toward the origins of cosmic rays, how their makeup can help us understand matter beyond our solar system, and how they get accelerated to very high velocities.

Because cosmic rays are electrically charged, they are deflected by magnetic fields, too complicated to trace exactly, making it extremely difficult to tell where they originated. Cosmic rays are studied with an array of detectors distributed over many square kilometers that sample the particles produced. Detectors used in studying cosmic rays range from Cerenkov detectors, to scintillators. The purpose of this project is to continue the work begun by researchers in finding an economical way to bring cosmic ray detection systems to high schools. In a sense, many small detectors will become one big detector.

2 Individual Contributions

Drew Thomas worked with the photomultipliers and scintillators, learning some basic electricity and magnetism in order to understand the basics of their functioning. He measured geometrical light losses in the light guide attached to the scintillators, tested photomultiplier gains, learned about Hess's experiment, calculated rates including the expected $\cos^2$ dependence of the cosmic ray flux, and assisted in analysis of the data. The high-energy particles were detected by setting up a coincidence procedure which detected particles passing through two scintillators. Light is produced as the particles pass through the scintillators. Photomultipliers will then be used to increase the number of chances one has at detecting a signal from the scintillators. The photons move inside the photomultiplier and hit a photocathode, and, through the photo-electric effect, an electron is produced. The electrons then bounce off specially aligned plates (dynodes), which are designed to create (and multiply) new electrons. Thus, more electrons mean a better chance of getting a clear signal. Roughly, a PMT gain is estimated at 2^n, where n is the number of stages of multiplication, or dynodes. In calibration mode, measuring cosmic ray flux, the scintillators overlap one another; if a specific particle has enough energy when it reaches one detector, it will pass through it and into the next detector. When this occurs, both detectors register in coincidence. A special timing module connected to the computer will initially be used to measure the time differences of particle detection. Eventually, for separated counter stations, a G.P.S. (Global Positioning System) will monitor the detection. Time monitoring, in each case, is important to tell something about the direction (and origin) of the showers. For each sequence, we will compare data received with expected data from previous and other current experiments.

We must understand the response of our detectors, and how uniform that response is. Femi Osidipe worked with Mr. Cliff Parker, visiting high school teacher, on measuring the response of our different detectors and organizing the information obtained into computer data. He also studied possible digitization methods for the data acquisition system, with a particular focus on the waveform digitizer.

Elizabeth Weber took part in some data collection, developed a C program to read the collected data and communicate it with a FORTRAN program in a linux environment, worked on the early stages of the analysis, and vetted a small cosmic ray shower simulation program to estimate single muon flux and overlaps from showers with energies up to $\approx 10^{15}$ eV. The simple program has simple assumptions: simple power law dependence of number of muons on the energy of the primary; no dependence of shower shape on energy except through the total number of muons produced; no correlations between particles; and neglect of fluctuations but simply using means to estimate fluxes. These assumptions are certainly all wrong in detail, but serve to give a "back of the envelope" estimate for overlap rates, dependence on overlaps with detector separation, and which primary particle energy ranges are the most important contributors to overlaps. The program written to analyze the data, and its output, were checked for validity. Various tests were made with the detector system and the data compared to results acquired at University of Pittsburgh and other institutions. The data collected were also compared to the simple simulation results.

3 Goal

This cosmic ray project is intended to detect showers resulting from high energy cosmic rays, using simple detection devices. Similar projects are being set up in other parts of North America /citealta,nalta,walta,chicos,crop. It is our hope, that through the collection of data more can be learned about cosmic rays, specifically the higher energy particles and their potential origins outside our galaxy. The ultimate goal of this project is to create a cost-effective detection system, with analytical capabilities and GPS time stamping, for use in high schools to involve high school students more heavily in hands on scientific research, experimentation, and learning.

4 Apparatus

The basic detector is a combination of 6 full size scintillators. configured in a vertical "stack" for calibration purposes, or spread apart in a "sparse array", for studying showers. Preliminary tests were done in May by Elizabeth Weber and Dave Kraus with fewer and smaller scintillators.

Work by Drew Thomas Scintillator:

A scintillator is a device that is used to detect specific particles, such as those of cosmic rays. The mechanism for production of light in the scintillator is shown in Figure 2. The scintillator is made of material that contains molecules that become excited as a particle enters the scintillator and collides with the molecule. As the molecule is excited as an electron jumps to a higher energy level it quickly decays and produces a photon.

Figure 2: Illustration of how a scintillator works. The PDF file is available on siue.edu/∼juthomp/2001docs.

Photomultiplier:

The photons emitted by the scintillator pass through the second part of the counter, the acrylic light guide, and into the photomultiplier (PMT), whose working is shown in Figure 3. A photomultiplier is an electronic device that converts light, or photons, into electrons through a process called the photoelectric effect. This process takes place in a component of the PMT called the photocathode. The photocathode is a semi-conducting film. As it collides with the photocathode, the photon knocks free some electrons bonded to the surface of the cathode. The freed electrons are guided through the PMT by focusing electrodes. The electrodes create an electric field, which directs the electrons to the first of several dynodes. Dynodes are copper-beryllium metals that are responsible for the multiplication of the electrons. As an electron strikes the dynode, it collides with particles along its surface and frees more electrons. At the end of the multiplication sequence, the number of electrons is approximately 2^n , where (n) is the number of dynodes.

Figure 3: How a photomultiplier works. The PDF file is available on: siue.edu/∼juthomp/2001docs.

Prior to assembling the photomultipliers with the counters, we tested them to see which were the most functional. Of the 14 photomultipliers that were in our possession, only 9 were usable. Figure 4, shows exactly what was measured, and which PMT's had problems.

For the non-functional photomultipliers, as shown in Figure 5 the noise and the signal were indistinguishable.

Figure 4: PMT calibration. The PDF file (pmtest.pdf) is available on siue.edu/∼juthomp/2001docs.

Figure 5: Data produced by DTAKE program. Pulseheight distribution typical of non-functional PMT's. The PDF file (dtakea.pdf) is available on siue.edu/∼juthomp/2001docs.

However, for the tubes we designated as "functional", a clear one photoelectron peak is seen, as shown in Figure 6.

Figure 6: Data produced by DTAKE program. Pulseheight distribution typical of "functional" PMT's. The PDF file (dtakeb.pdf) is available on siue.edu/∼juthomp/2001docs.

Estimate of One Photoelectron Signal from Photomultpiers An estimate of the measured charge corresponding to a one photoelectron signal is shown in Figure 7. The charge found is ≈ 1.2 pico-coulombs.

In the early segments or the project, I (d.t.) took part in the construction of the scintillator platforms. The plan was to construct a platform, using 2x4's, which would fit a scintillator and would allow an additional scintillator to be mounted on top. The initial plans for the platform is in Appendix D However, this configuration was a bit complicated, considering the time that would be needed to build each platform. The current platform being used is also in Appendix D. This platform allows for .25in. on each side and 4in. in front to the attached photomultiplier.

Figure 7: Description of the method used to measure the approximate charge collected from our photomultipliers for one photon incident on the photocathode, liberating one electron ("one photoelectron signal"). The PDF file (onepe.pdf) is available on siue.edu/$\sim$juthomp/2001docs.

Trigger Logic and Data Acquisition A schematic of the electronics is shown in Figure 8.

Figure 8: Schematic of the Trigger Electronics and DAQ for the CosRayHS project in Summer2001 (July4 data). Only two of the photomultipliers are shown, but typically 6, and up to 7, were used. A second set of ADC's, attenuated by a factor approximately 4, were added after July 4. The PDF file (trigdaq.pdf) is available on siue.edu/~juthomp/2001docs.

The signal received from the scintillator is split in two; one of these goes to a discriminator. The discriminator outputs a standard logic signal, but only if the input signal is above a certain threshold, which can be set to a desired value. The other piece of the original signal is sent to the ADC card in the CAMAC, Computer Automated Measurement and Control, crate. The ADC, or analog to digital converter, does just that, converts analog signals to digital signals; however, it only converts the signals that arrive within a specified gate.

After the signal leaves the discriminator, it is again split in two. One of these goes to the TDC, time to digital converter, and the other goes to a voter coincidence card. The TDC simply counts from the time at which it receives a start signal, to the time at which it receives a stop signal. The voter coincidence can receive up to six signals and works just like a logic AND. It can be set to include all or none of the received signal, and sends off a zero or a one, a true or a false, indicating whether the conditions were met.

The signal from the voter coincidence is used as both the start for the TDC, and the gate pulse for the ADC. Before being sent to the ADC, however, the gate pulse passes through a logic card which regulates the pulse to a specified size. This is where a slight problem can be found: it should be obvious that with the system as it has been described, the stop for the TDC will arrive before the start, and similarly, the signal before the gate. To prevent this, the signals running to the ADC and the TDC from the scintillator and discriminator are channeled through an appropriate length of cable to ensure their arrival after their corresponding regulative pulses.

All of the information received by the ADC and the TDC is sent to the crate controller, and then to a PC through a ribbon cable. The PC controls the data collection process and performs some preliminary analysis.

Two of 6 signals were typically required. In the August 7 data, the trigger required a small counter on top of the stack, in coincidence with a counter at the bottom of the stack.

GPS Time Stamp In this summer's running, a GPS time stamp was developed by Dave Kraus, based on a scaler to count the number of GPS 1/100 second signals received from the start of the run, and a scaler (actually set of two) which interpolates between successive GPS signals. In implementation, one interpolating scaler is frozen at the time of the trigger, while the second continues to count. The two interpolating scalers are read twice in each event. The one which corresponds to the time of the real event is frozen at event time. Therefore, one of the re-read scalers should always read the same in the two readings. Figure 9 shows a scatterplot of the time differences betweem the two readings of each of the scalers. One time difference should be zero in each event, and we see that is the case. This time difference can be monitored as an indication of event quality.

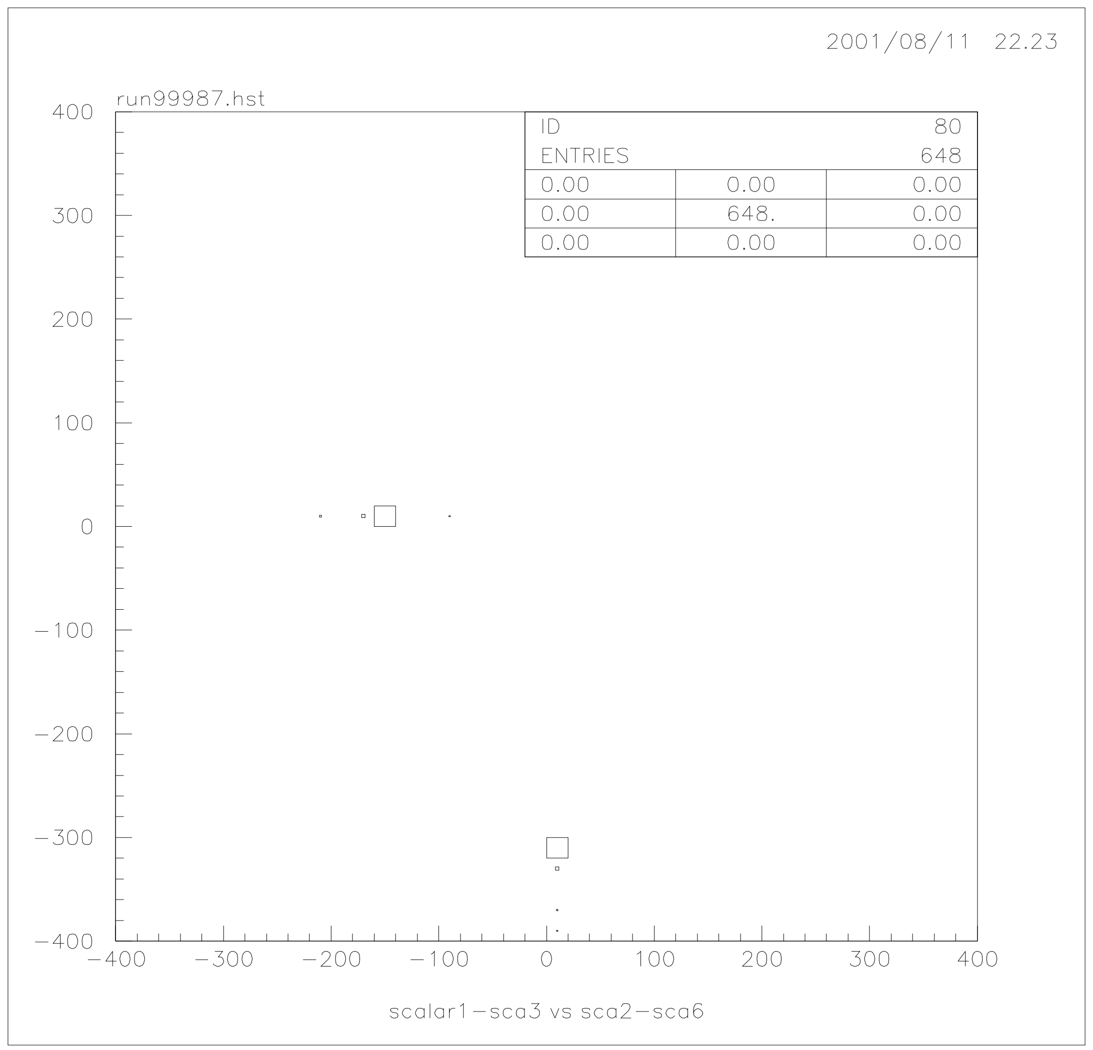

Figure 9: Scatterplot of differences in the re-read interpolation scalers, showing that one of the pairs has been successfully frozen.

5 Rate Estimates

5.1 Estimates of Muon Rates, Including Angular Dependence (dt,jat)

To better understand Hess's experiments, we calculated the expected rate of ionization in his cm^3 ion chamber, taking into account the expected $\cos^2$ dependence of the cosmic ray arrival. The actual calculations are located in Appendix F. Rates were calculated for our large counters of 0.42m^2, a smaller trigger counter used in the Aug. 7 data taking (0.045 m^2), and for Hess's small cm^3 chamber. The rates, respectively, were 31 muons/sec, 3.4 muons/sec, and 0.02 muons/second. Taking into account expected ionization losses of the muons (2 MeV/(g/cm^2)) and a typical ion pair production rate of 30eV per ion pair, we predicted a rate of 3 ion pairs per second while Hess observed an ionization rate of 10 - 3 (background from the earth) , or 7 ion pairs per second due to extra-terrestrial "cosmic rays".

5.2 Shower Development (dt,jt,dek)

In order to perform an analysis of the data gathered with the detectors, an analysis program had been written (by jt and Grace Schoonover, former REU student) based on some crude assumptions on shower development. These assumptions have been re-examined for validity and the program checked for consistency and extensively commented.

A particle shower can start as early as the top of the atmosphere. If we estimate the atmosphere's density to be roughly $1000g/cm^2$, then the change in energy of a charged muon (whcih loses energy primarily through ionization losses) moving through half an atmosphere could be described by

$$dE(1/2atm) \approx (2MeV/g/cm^2)(500g/cm^2) \approx 1GeV$$

This means that in order for us to see the shower at the earth's surface it has to be at least 1GeV.

The interaction length of a charged particle in air is $60 - 90g/cm^2$, therefore we could expect roughly six interactions in half an atmosphere. If the energy is split equally between the particles with each interaction, then

$$E(of\ resulting\ secondary\ at\ 1/2atm) \approx Eo/2^6 \approx E^o/64$$

Therefore, to obtain muons greater than or equal to 1 GeV at the earth's surface, there is a minimum energy for the shower:

$$Eo \approx 100GeV or \approx 10^{11}eV$$

The intensity at which we see primaries of a given energy is described by [2]

$$IN(E) = 1.8E^{-2.7}nucleons/(cm^2 * sec * sr * GeV)$$

The number of charged particles Ne resulting from a primary of a specified energy is (empirically) found to be [2]

$$Eo \approx 3.9 * 10^6(Ne/10^6)^{.9}$$

The number of muons greater than 1 GeV at the earth's surface, from a shower from a primary with Ne total charged particles, is (again, empirically /citePDG2000):

$$Nmu(E > 1GeV atground) \approx 0.95 * 10^5 (Ne/10^6)^{.75}$$

If we invert the initial energy equation, (which Ref. [2] advises against because of shower fluctuations, but we still use for our crude estimation purposes) primary energy, but for our estimation purposes it's ok), we can calculate the expected number of muons in shower from a primary particle of energy Eo:

$$Nmu(E > 1GeV \text{ at ground}) \approx 0.95 * 10^5 [Eo/(3.9 * 10^6)]^{.83}$$

For energies of 10 GeV, we would expect 2 muons; for 100 GeV, we could expect 15 muons. As noted above, most muons from showers with primaries less than 100 GeV would be stopped before reaching the ground, so we expect that this equation has little validity below 100 GeV. If we integrate over the energy we can obtain the rate at which we would expect to see a muon:

$$Nmu \approx (\text{from Emin to } \infty) \int IN(E) * Nm(E) dE$$
$$\approx 6000 \Omega Eo^{-0.87}.$$

The rate given in the Particle Data Book is $\approx 130/m^2/sec$ for muons passing through a flat detector. Taking $\Omega = 6.28$ (2 π, or all solid angle from the upper atmospher), we find Emin $\approx$ 700 GeV to match the PDG expected single muon rate. However, as for the single muon flux, the solid angle may need to be reduced by about a factor of 3. With the factor of 3 reduction, Emin is about 170 GeV.

(Note added in proof: jat continued these investigations after Elizabeth Weber had to leave, and found that including showers up to $\approx 10^{15} eV$ at the upper end in our simple numerical integration program required raising the lower limit for $\Omega = 6.28$ to about 1500 GeV, or about $10^{12} eV$. The reason for this apparent discrepancy has not been checked but may be due to the large step size in our numerical integration routine.)

The muon density at distance r from the center of the shower can be described by

$$\rho_\mu = (1.25 * Nm)/(2 * \pi * \Gamma(1.25)) * (1/320)^{1.25} * r^{-.75} * (1 + r/320)^{-2.5}$$

and normalized

$$\rho_{\mu'} = \rho_\mu / Nm$$

We can then calculate the rate of overlap we would expect in a pair of detectors from a shower of given energy. Figure 10 shows a the configuration we imagine, with the first detector at the origin of our coordinate system and the second detector arbitrarily set on the x-axis, a distance D away from the axis, where D is the separation of the two counters.

Figure 10: Two detectors, of area A1,A2, separated by a distance D. A cosmic ray shower occurs at R,θ. We wish to calculate the rate of overlap (muons) between the two detectors. This PDF file is on: siue.edu/ juthomp/html/2001docs.

Table 1: Overlap rates, in muons/min, for two 1 m^2 detectors, for varying separations. The Low and High integration ranges are described in the text.

Separation (m)	High (Ω 6.28	Low (Ω 6.28)
0.	18.	.055*60*1.5 ≈ 5.0
100.	7.1	0.022*60*1.5 ≈ 2.0
200.	3.9	0.012*60*1.5 ≈ 1.1
300.	2.3	0.007*60*1.5 ≈ 0.63
400.	1.5	0.005*60*1.5 ≈ 0.45
500.	1.0	0.003*60*1.5 ≈ 0.27
600.	.72	0.002*60*1.5 ≈ 0.18
700.		0.0016*60*1.5 ≈ 0.14
800.		0.0012*60*1.5 ≈ 0.11
900.		0.00091*60*1.5 ≈ 0.08

The calculation method is to take the number of muons collected in the first detector ($Nmu* \rho_{\mu^{prime}}$), from a particular shower, mulitplied by the number of muons remaining which might possibly hit in the second detector (Nmu minus the one already detected by the first), multiplied by the probability per mu ($\rho_{\mu^{prime}}$ of seeing a muon from that shower in the second detector, and integrated over the shower position.

$$OV(E) \approx \int Nmu(E) * \rho_{\mu'}(r) * [Nmu(E) - 1] * \rho'_{\mu}(r - ro) * d(showerposition)$$

To obtain the total number of overlaps from all energies we need to include the areas of the detectors as well as the integration over the energy.

$$OV(obs) \approx$$
$$A1 * A2 * \int N(E) * Nmu(E) * [Nmu(E) - 1] * dE * \int \rho'_{\mu}(r) * \rho_{mu'}(r - r0) * d(showerposition)$$

Two programs have been written to perform this series of calculations: one (XNMU) which simply estimates the muon rate integrated over shower energy, (appendix B), and a second (COSGRACE) which estimates the expected number of muons in a shower and overlap rates, (appendix C). The suggestion for these programs came from Prof. Julia Thompson, and the project was begun in summer 2000 by Grace Schoonover, an REUP-FOM participant. The program was modified in some details, extensively commented, and checked for errors by Elizabeth Weber in summer 2001. Further modifications beyond the version included here were made by jat after Weber left, to allow efficient inclusion contributions from higher energy cosmic rays. We note that the integral of ρ_{μ} over all space, as required to get the single muon flux, should be 1.0, but our numerical integration program finds it to be 0.81. We therefore renormalize it to 1.0 by multiplying by 1.2. The source of this error is not yet understood. It may arise from an incomplete r integration, although we integrate out to 1000 km.

In our program XNMU taking Ω as 6.28, and the limits from 1600 GeV ($1.610^{12} eV$ to ≈ 10^{15} eV gives 126 muons/(m^2 * sec), in agreement with the 130 given by the PDG (or 1/min/cm^2) for single muons, and 90458 for the double overlap integral. For the future, we also include the triple overlap integral result (1.74 * 10^9), although we have not yet calculated triple overlaps.

Results are presented in Table 1 for the two integration ranges: Low: 200-1000 GeV, and Hi: 1600 GeV (1.6 10^{12} eV) - 10^{15}eV

Both cases gave ≈ 130 $\mu/(m^2 sec) for single muons. As intuitively expected, overlaps have a relatively larger contribution from higher primary energies than do single muons. Qualitatively, this is tehreason for a sparse, distributed detector system to study high energy cosmic ray showers.$

6 Analysis

6.1 Initial Data, late May

The data taken in May, 2001, by Elizabeth Weber and Dave Kraus was used to introduce students to the data acquisition process and the tests required to verify that the equipment was working. Some results are shown in Figure 11. For this particular set of data, two scintillators were set one on top the other in opposing orientation. One thousand events were collected and sent to the analysis program, yielding the figures below.

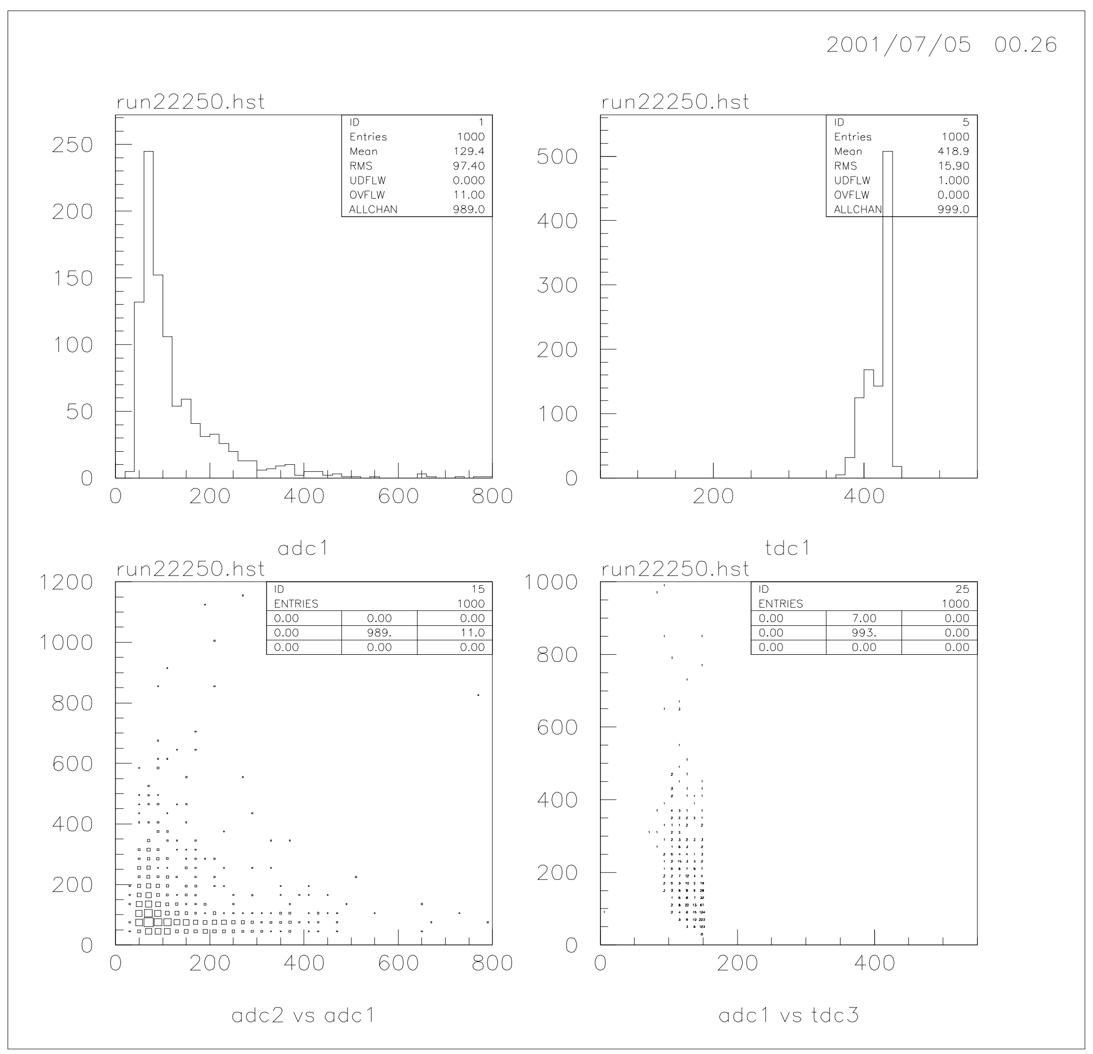

Figure 11: Results from the May, 2001 run

The plot of the adc's are simply a plot of the sizes of the events received for each detector. The plot of the tdc is simply a plot of the times for each detector. Plotting the two adc's against each other shows us about what we expected for one minimum ionizing particle going through the scintillator. The concentration at the origin is where both detectors see a min ion peak. The tails are related to both the expected Landau distribution and the result of the propagation through the scintillator and the imperfect light pipe. The scattered dots in the remainder of the plot could be the result of multiple particles in the scintillator or instances where both detectors experienced an upward fluctuation.

The adc vs tdc correlation shows that the late times (when pulseheight is cut off by the time window, or when pulses arrive after having traversed the length of the counte) correspond to smaller pulseheights.

This run served as a good calibration and learning tool. It allowed us to obtain some initial idea of what the data should look like in tests to come and it gave a good hands-on learning experience to see what the histograms would look like and what the different correlations meant. (Initially, the ADC-TDC correlations were crossed, sending us back to our notebooks to remind ourselves of the actual data taking configuration.)

6.2 July 4, 2001

Early on July 4, 2001, data were taken with the stack of 6 scintillators inside the Science Building at SIUE. Later in the morning, the scintillator stack was moved outside. From this data, rates of ≈ 1/min were found for coincidences between counter pairs separated by 10-20 m. In about 7 of the 1000 events, all 6 counters gave large pulseheights. Only 1 of these 7 events showed both TDC and ADC information for all counters, perhaps due to pre-emption of the TDC channels by earlier events.

A problem in interpretation of the pulseheights was revealed in the study of this data. In calibration data (run 99957), taken to study the apparatus properties on single muons, the stack was arranged vertically in increasing order with 1 on the bottom, 6 on the top, and arranged so that even scintillators were read out on one side, odd on the other. In this way, the difference in time of arrival (TDC) of the PMT pulses of odd/even pairs of PMT's gives an estimate of position. Figure 12 shows the timing correlation from which position is derived, and also pulseheights of counters when the particle position is close to, or far from, the PMT for that particular counter. A striking difference is seen which, as discussed below by Drew Thomas, we attribute to geometrical losses of light rays originating in the scintillator end far from the photomultiplier.

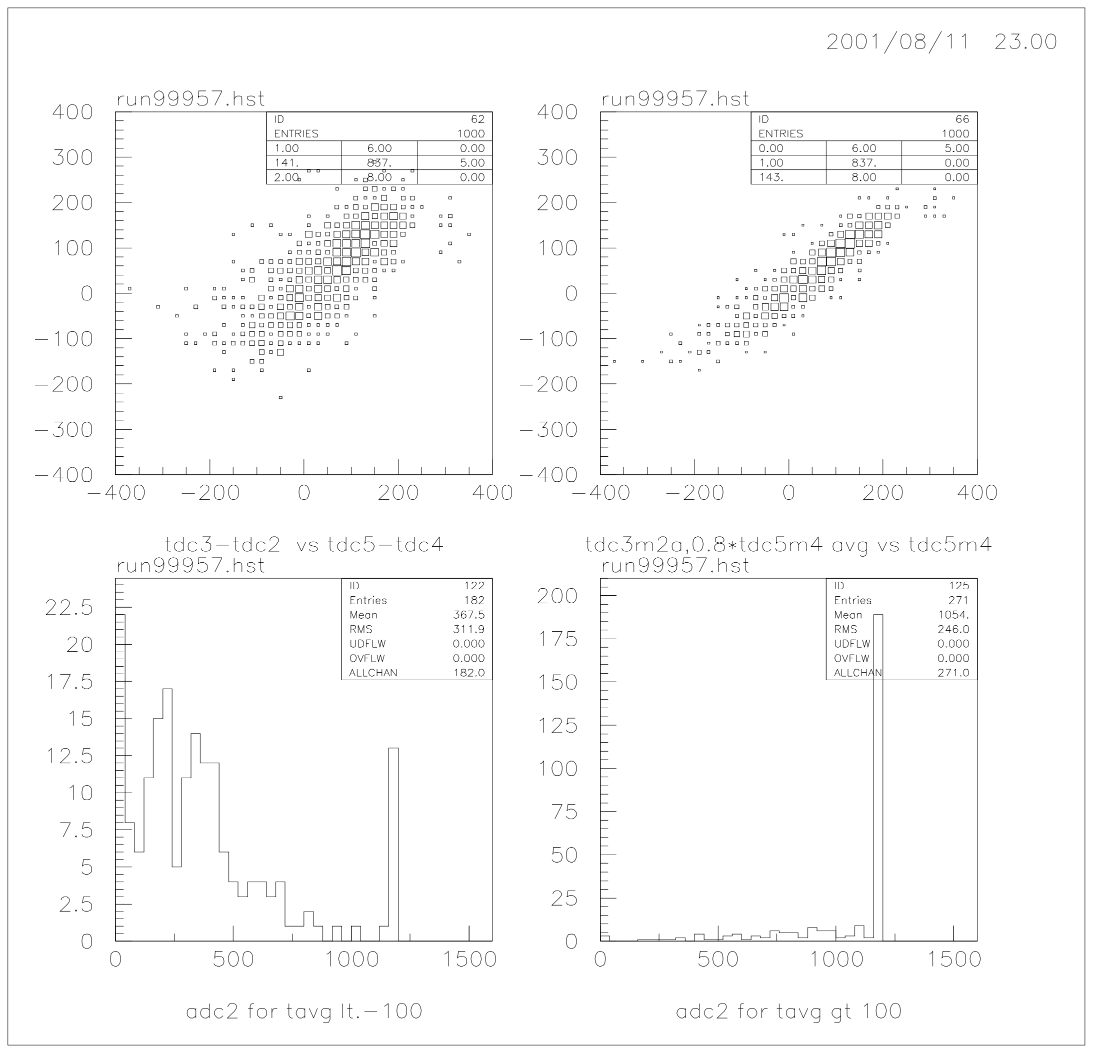

Figure 12: Graph showing how many times the counters fired.

6.3 Analysis Description by Drew Thomas

During the second half of the project, the cosmic ray group focused mainly on collecting data, unlike the first half when we stressed the understanding and calibration of the equipment being used. My part in the second half consisted of understanding the computer languages in analyzing the data to run the programs. Using the operating systems of Unix and Linux, I briefly learned the commands associated with programming languages of Fortran and PAW-Kumac. One of the programs used is the Fortran code, "siuedat.for" (Appendix Ea). Used along with Elizabeth's C program, it is used to read in the data. This program makes an output file called an "ntuple", which the PAW (Physics Analytical Workstation) can read. The other program used is the PAW-Kumac code, "stack.kumac" (Appendix Ec), and its "select function" – the program called in by stack.kumac – Selstack.for (Appendix Eb). The first data collection I took part in analyzing took place on July 26, 2001. Here we were trying to use the data, which was in the form of histograms, to calibrate the signals from the stack of scintillators. We did this by editing the "selstack.for" program. We edited the program to subtract the pedestals from the 'adc' (analog to digital converter) amount to get an amount commensurate to the charge deposit. The pedestals are offsets due to an additional charge that is automatically leaked onto a capacitor that measures the 'adc' amount, in order to put the zero charge point within the range of the ADC.

July26 Results Run99970 was taken on July 26. This is a run with the six counters in a stack. Some electronics improvements were made with respect to the data on July 24. High voltages were adjusted for better overall gains, and a set of adc's reading signals attenuated by a factor of about 4 were added to the data record. From Figure 13, we can see that all six counters fired most of the time, which means that they are working. (Since the counters were configured in a stack for this run, triggering on a coincidence of the top and bottom counters, we would expect that all counters should fire for a valid trigger. Failures would be due to various mismatches of thresholds or other electronic failure.)

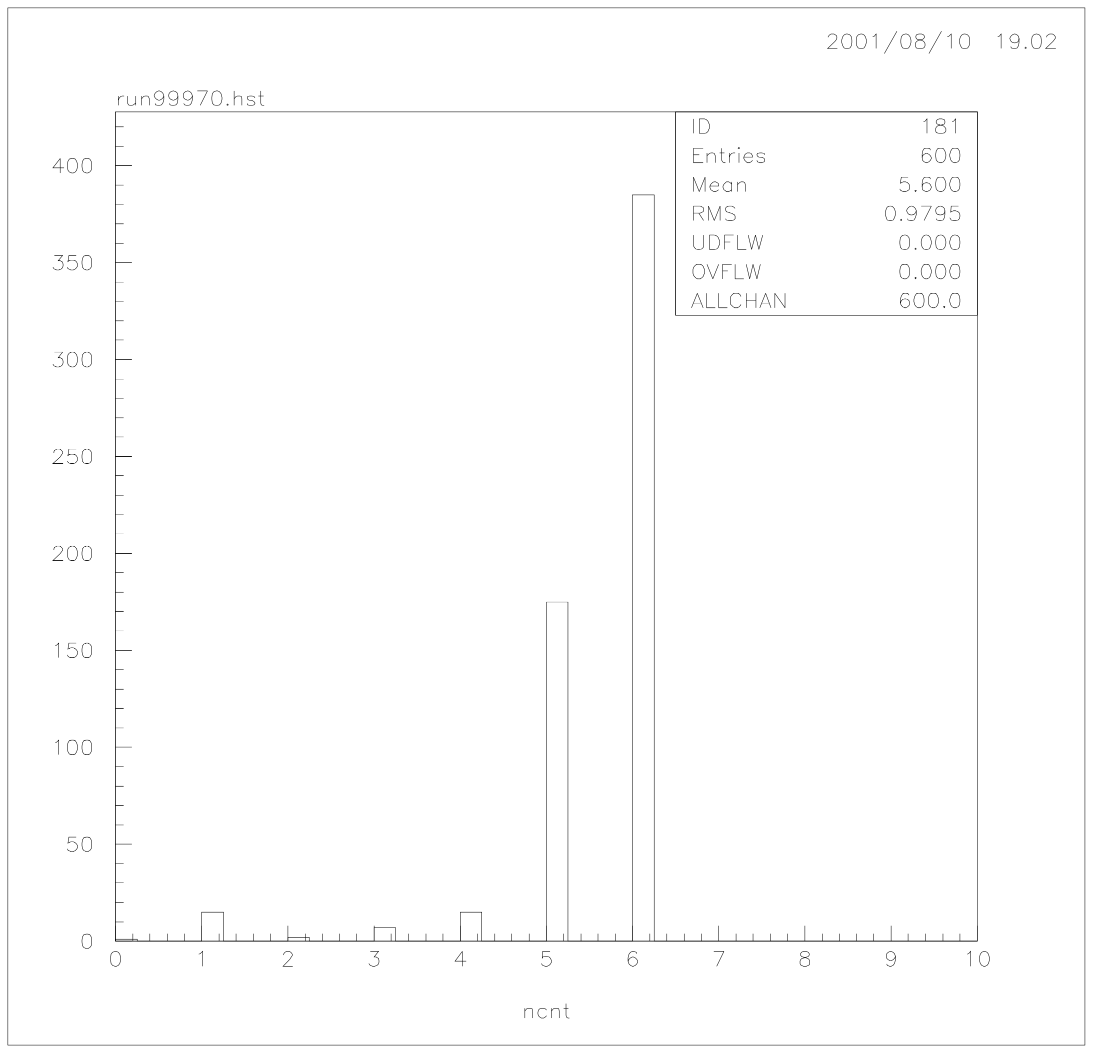

Figure 13: Graph showing how many times the counters fired.

Calibration (Stack) Data taken August 7, 2001 The second set of data I [Drew Thomas] took part in analyzing took place on August 7, 2001. The purpose of this was to get an estimate on the number of muons/sec that the counters were detecting. What we did differently in software for in analysis of this data was to add thresholds. In the hardware, the events were triggered on a small counter (area 0.045 m^2) on top of the stack in coincidence with the counter on the bottom of the stack. By calculating the area of the smaller counter that was recently placed on top of the original six, and integrating over the solid angle of the combination of the small counter at the top of the stack and the the larger counters at the bottom of the stack, we found a rate of 3.4 muons/sec. We compared our rate to the rate of 2.5 muons/sec, which was calculated by David Kraus. They differ slightly. We hypothesize that the difference is because our estimate is as if the counters were triggering outside of the building. We believe that the building may filter many of the muons out, which is why David's estimation of 2.5 muons/sec is lower than 3.4 muons/sec. The discrepancy may be due to the loss of energy and stopping of some muons in the stack. A calculation by Julie Breden indicates that about 1 GeV of energy would be lost by the muons in traversing the building, leading to a small loss in rate, of order 20%. Of course, our calculation is only for muons. In principle, before entering the building, about 50% more particles (in the form of electrons) would be present in the cosmic ray flux. These particles, if low energy, would be filtered out by the approximately 12 radiation lengths in the building.

Figure 14 shows the high efficiency of the counters for these HV settings.

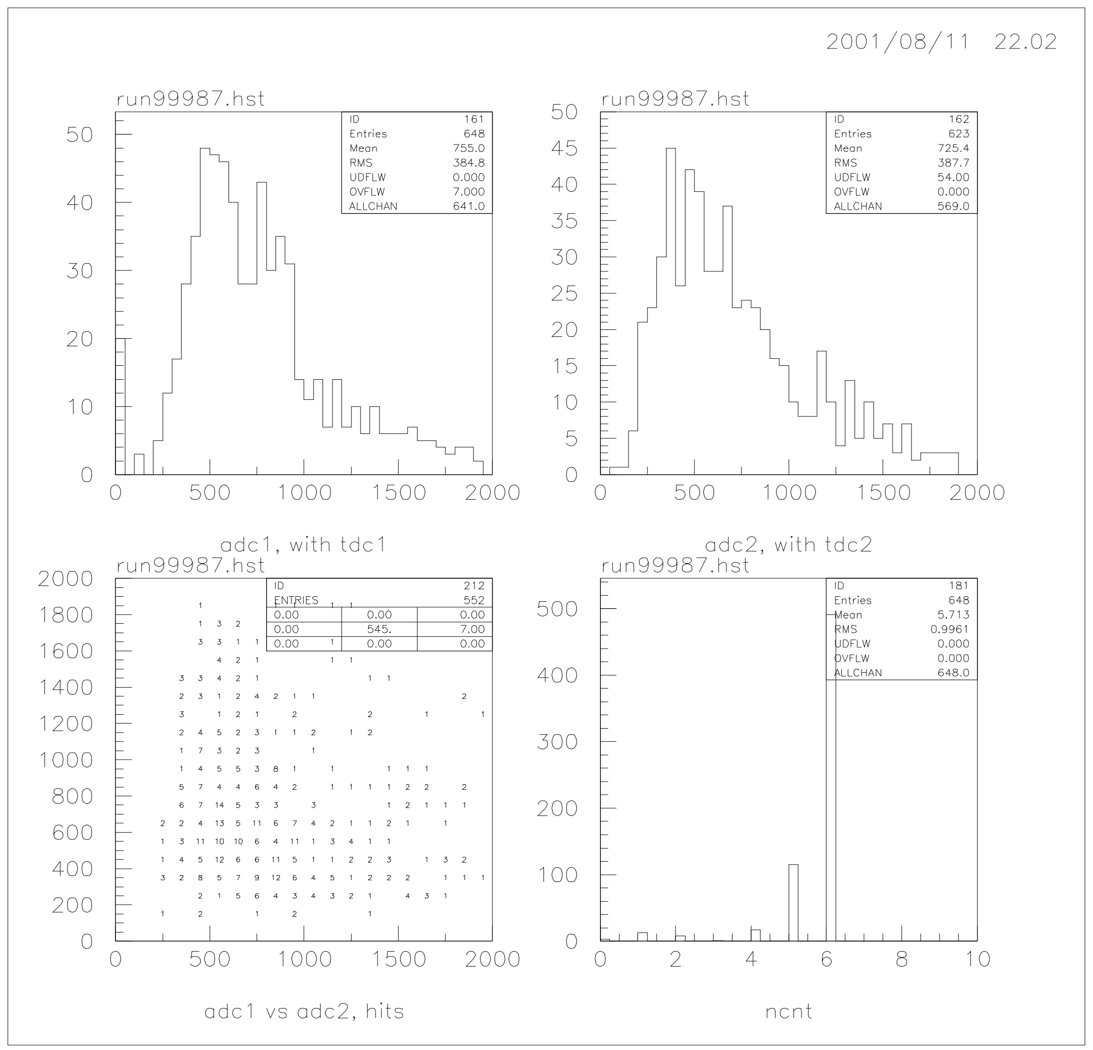

Figure 14: Number of counters firing in individual events. Counters are arranged in a stack, and high efficiency, due to collection of cosmic ray muon singles, is seen.

August 9 Results [discussion by jt] On August 9, more data was taken, this time with a GPS time stamp included in the data. First data was taken with the counters in a stack (Run 99987), and then the stack was spread out in the main lecture room which is in the center of the Science Building.

The trigger for event collection in the spread configuration (Runs 99993-5) was 2 of the 6 counters. To better accomodate the dynamic range of the signals, the high voltages on the PMT's was reduced, and stack calibration data with these lower high voltages (Run 99996) was taken at the end of the spread runs. Figures 15 and 16 show the attenuated and "regular" ADC's, and the number of counters in coincidence in individual events. Because of the lower high voltages, inefficiency is seen in individual counters. These inefficiencies may be used (with some caution because of possible different pulseheight sizes) to correct the counting rates in the spread array.

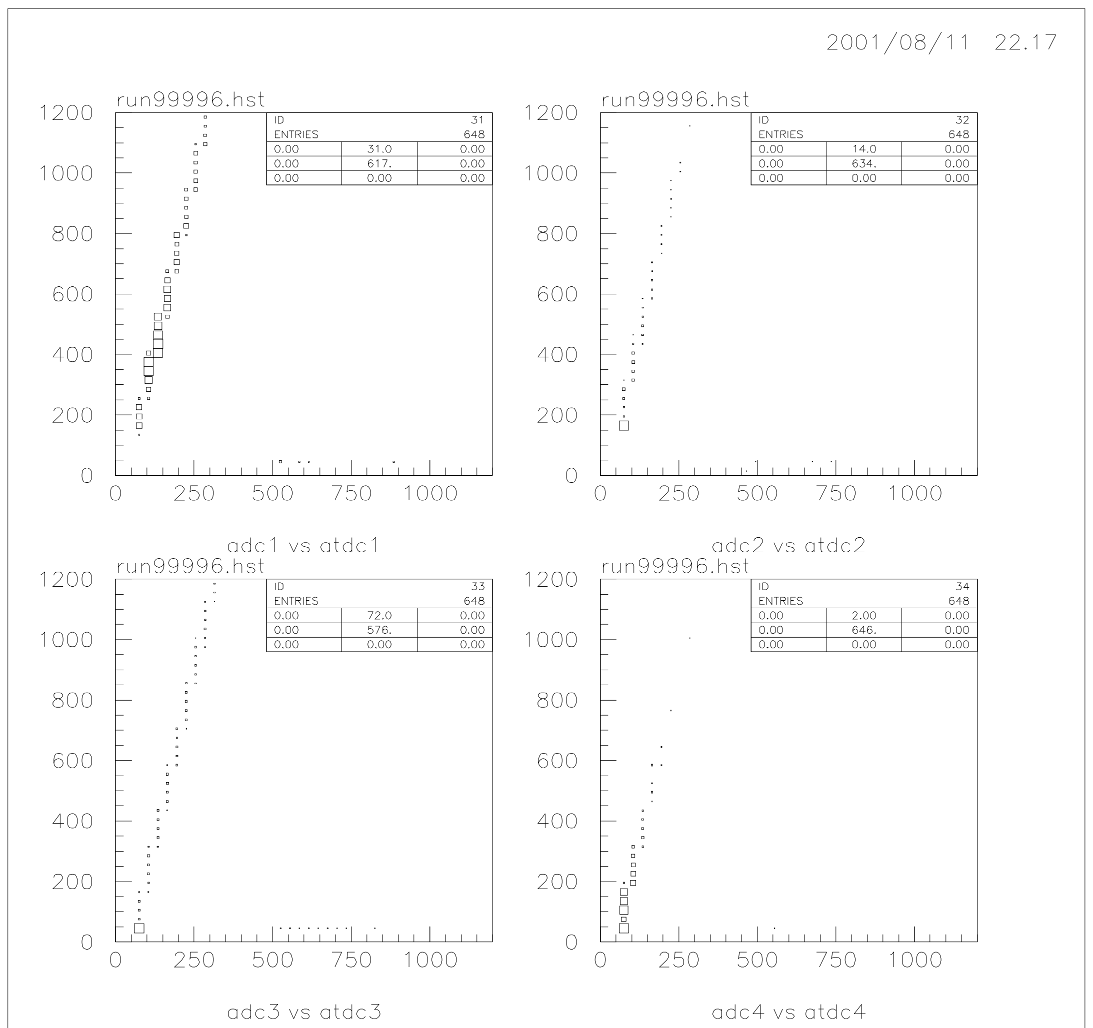

Figure 15: Comparison of "regular" with attenuated ADC distributions.

In the spread array, of course, not all counters contribute to each event. From Figure 17, we can see how many combinations of two counters fired. Figure 17 shows that most of the triggers were a coincidence of a pair of counters. The trigger rate was approximately 650 triggers in 50 min, or approximately 12/min. The total rate of true doubles coincidences was about half of this. Comparison of coincidences of individual counters gave rates ranging from 0.3 coincidences/min to about 4 coincidences/min, again in rough agreement with our rough rate estimates, with in this rather rough measurement.

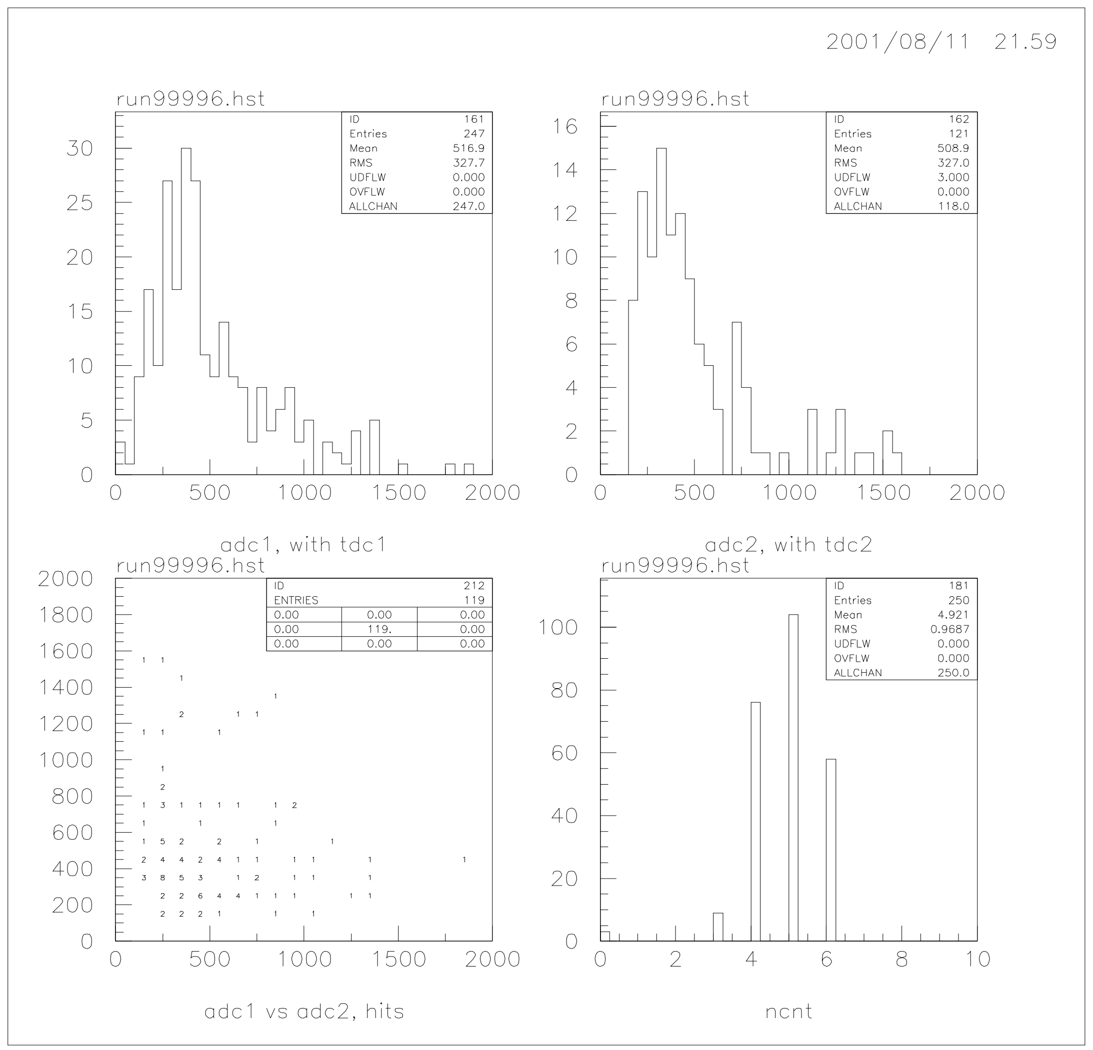

Figure 16: Number of counters firing in individual events. Counters are arranged in a stack, and high efficiency, due to collection of cosmic ray muon singles, is seen.

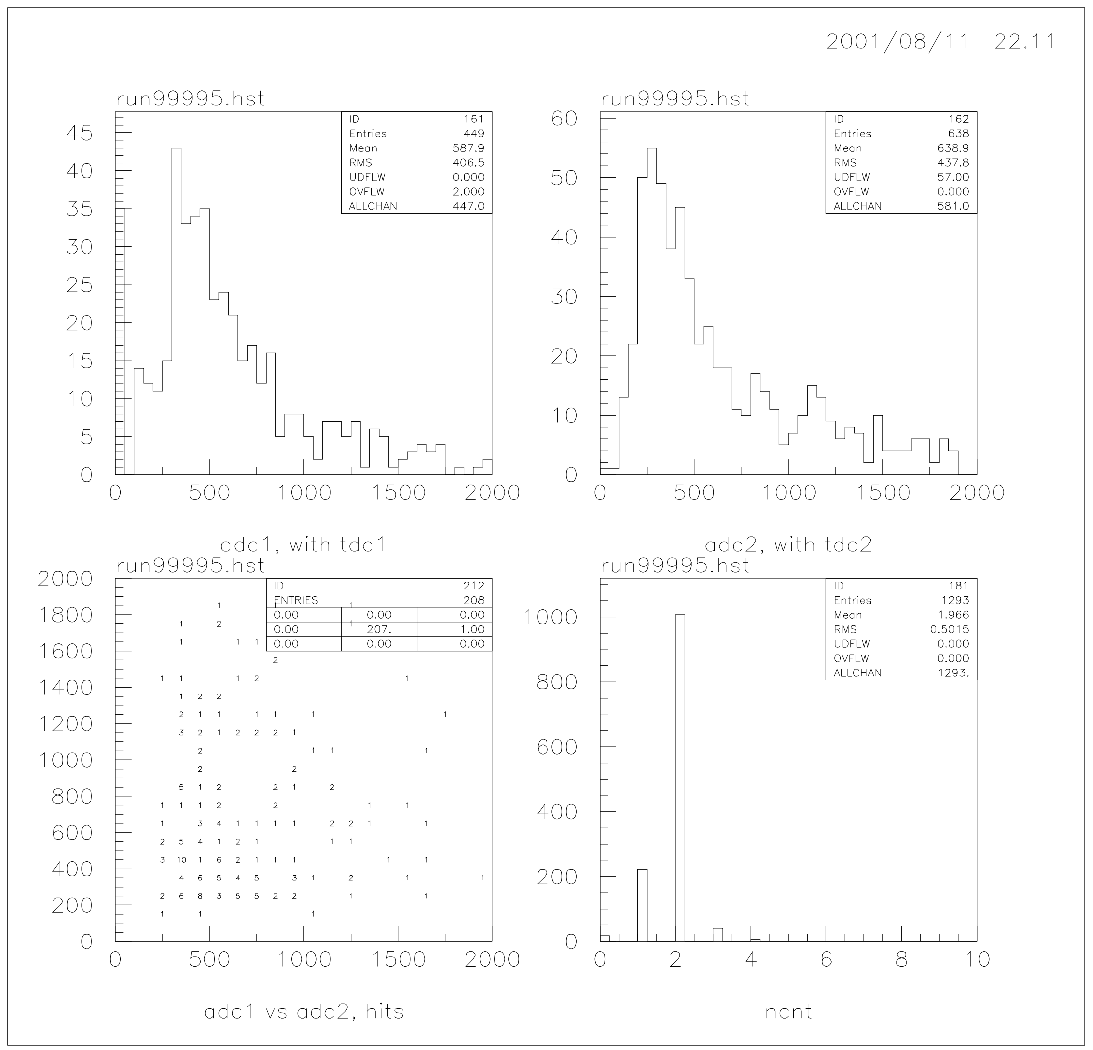

Figure 17: Graph showing how many times the counters fired.

6.4 Possible Improvements to the Equipment

In addition to collecting data with the existing stack of six scintillators, we began to build upon our current equipment by building another set of six scintillators. Each of the six possible scintillators was in two pieces. The plan was to glue them to their approximately triangular light guides together with a special adhesive which transmite the light without compromising the amount of light that we want to collect. First, however, we wanted to improve on the number of photons that enter the photomultiplier. Initially, as the photons left the scintillator portion of the counter and into the acrylic, some of the light would be reflected back into or out of the scintillator, and wouldn't be collected by the photomultiplier. To study this effect, I first measured the range of angles for each entry position x (perpendicular to the scintillator axis) for which rays would be collected. The results are shown as a band in

Figures 18 and 19. For some representative x - points at the middle and rear of the scintillator I then calculated, for a range of entry points to the waveguide, the angle corresponding to those entry points. If the angle is in the measured accepted angular range for the given x entry position, the rays are considered accepted.

Figure 18: Angle range and average light collected. The band gives the range of angles collected for a given entry point to the scintillator. The slanted lines correspond to the set of entry positions and angles for rays coming from half way to the back of the scintillator. Each line is labeled by the x-position at which the rays originate. PDF file on siue.edu/∼juthomp/2001docs

Figure 19: Angle range and average light collected. The band gives the range of angles collected for a given entry point to the scintillator. The slanted lines correspond to the set of entry positions and angles for rays coming from the back of the scintillator, from the x-position shown. The collected part of each trajectory is the part which falls within the measured scintillator entry acceptance band. PDF file on siue.edu/∼juthomp/2001docs.

From these calculations, we affirm the observed result from the July 4 data, that light that arose from the back of the scintillator had less chance of getting to the photomultiplier than light arising from the front. This is illustrated in Figures 18 and 19. by measuring the angles as a function of position along the scintillator.

If we assume that the collection of light is uniform along cos (θ), then we can calculate the average amount of light that would be collected at specific points throughout. The amount of light collected from the front of the scintillator is greater than light from the back about a factor of 4. (Average fraction of light collected was about 0.06 for rays emitted at the front of the scintillator, just at the lightguide joint, 0.03 for rays emitted from half the length of the scintillator away, and 0.015 for rays emitted from the rear of the scintillator. Determining a way to reduce internal refraction and increase light gain was determined to be too difficult while still using the acrylic light guide. Another plan proposed was to use fiber optics instead of the light guide in order to collect the light more uniformly. While we did start to build such equipment, the procedure was never implemented.

7 Possible use of waveform Digitizers

The waveform digitizer was introduced to solve the problem of cost in PC-based data acquisition systems, as described further in Oluwafemi Osidipe's final presentation and paper.

Functioning of Wave Form Digitizer A block diagram of the Wave Form Digitizer is shown in Figure 20.

Figure 20: Block Diagram of Wave Form Digitizer. The PDF file is available on siue.edu/∼juthomp/2001docs.

The incoming signal is digitized using an Analog to Digital Converter (ADC). The functioning of generic ADC's (parallel encoding, half-flash, and successive approximation) were also studied by Oluwafemi Osidipe.

8 Conclusions

We have assembled and tested 6 scintillators, first in a stack, to use observed rates of single muons as a calibration of our equipment, and also in a sparse array, both inside and outside the Science Building at Southern University of Illinois Edwardsville, in order to look for correlations due to particles coming from the same initial primary.

We have compared these results to a simple model of shower development and found rough agreement.

In collaboration with two high school teachers, Mr. Cliff Parker, and Ms. Julie Breden, we have considered ways to introduce such equipment into high school physics classes in a pedagogically interesting fashion. In addition to the 6-counter group which we used in several contexts, another 6-counter setup is partially assembled. We hope to work with high school students to reach a better understanding of cosmic rays. For the high school projects to be successful, we believe it is necessary to make setups of reasonable cost and to define appropriate projects.

9 Appendix A: Designing eread

When moving the project to Southern Illinois University, Edwardsville, the move was also made to using a Unix based system for analysis. However, in order to do this, a new input function had to be written and so, eread was developed.

Eread is written in C; it was originally designed in C++, but as it needs to interface with a fortran main program, translation to C was necessary because there is no direction communication capable between C++ and fortran. In order to commnuicate with the fortran code, some of the fortran conventions had to be observed in the C program. For instance, fortran expects an underscore at the end of the names of functions and subroutines and automatically appends these to functions and subroutines created in fortran; C does not. An underscore has to be physically added if the function or subroutine is to be called in fortran. In addition, we discovered that the names of corresponding argument and parameter arrays, if they are to occupy the same space in memory, should be the same. Through edits and iterations, eread.c was renamed to ereadnjt.c, but the subroutine itself remained nominally eread.

The algorithm utilized by eread is fairly simple. It takes in an array for data storage and a two flags to designate whether the information to be read is a header or record and to indicate success of the subroutine to read in data. Based on the readtype flag, eread attempts to read in the appropriate number of data values, setting the success flag accordingly. As neither the program nor the data is particularly labor intesive, no functionality was implemented to insure that the input file is closed. This can easily be fixed with few lines of code should it be deemed necessary.

```
#include <stdio.h>
#include <stdlib.h>

#define HEADER_SIZE 8
#define RECORD_SIZE 14
#define HEADER 0
#define RECORD 1
#define SUCCESS 1
#define FAILURE 0
```

```
#define FILENAME "RUN99965.DAT"

void eread_(int *readtype, short int ix[], int *success)
{
  static FILE* fin = (void*)-1;
unsigned int num_to_read;

if(fin == (void*)-1)
{
fin = fopen(FILENAME, "r");
}

if(*readtype == HEADER)
num_to_read = HEADER_SIZE;
else
num_to_read = RECORD_SIZE;

if(fread(ix, sizeof(short), num_to_read, fin)<num_to_read)
{
fclose(fin);
*success = FAILURE;
}
else
*success = SUCCESS;
return;
}
```

Subroutine eread

Eread proved more useful than orginally planned. Not only is it used in the siuedat program, but Femi was able to use it in his analysis of the communication of the waveform digitizer with the pc.

The development of eread was helped greatly through the assistance of faculty and students of the computer science department at SIUE. Thanks to Marilynn Livingston, Bill White, Pete Motykowski, and Andrew Lamonica for their help.

10 Appendix B: Program Estimating Muon Rate

```
c PROGRAM: XNMU.FOR

c  en is in Gev
c to make a very rough estimate of rates, we take some formulas from
c PDG 2000 (also PDG 1998) and extrapolate beyond expected limits,
c also inverting where fluctuations make this procedure questionable.

c  differential spectrum is 1.8 En**-2.7 nucleons/cm**2/sr/gev.
c  so approx. 1.8*2 pi* En**-2.7 nucleons/cm**2/gev
c  or       12 10**4 E**-2.7 nucleons/m**2/Gev
c  so, add up "hard" muons (greater than 1 gev) expected at sea leael
c  from these nucleons.
c further, showers with much less energy than 1000 gev (10**11 ev)
c  cannot make muons with energy  greater than 1 gev at the earth's
c surface... the range for a 1 gev muon is 2 mev/g/cm**2, and the
c earth's atmosphere is about 1000 g/cm**2, or half the atmosphere
c would be 500 g/cm**2, or the range for a 1 GeV muon.
c so the muons produced would need to be 2 geV halfway down the earth's
c atmosphere.
c  the interraction length is about 40 g/cm**2, so about 12 interaction
c lengths in half the atmosphere. if one makes a crude model that the
c shower doubles in an interaction length (and energy halves) then
c in 8 interaction lengths a 200 gev shower would drop to 1 gev secondaries
c ( approximately).
c analytic integration shows that integration from 200 gev to infinity
c gives about the right muon flux greater than 1 gev ("hard" component?)
c at the surface of the earth:
c  do 100-1000 gev, in 10 gev steps:

c Summary of Important Variables:
c   en - energy of the primary cosmic ray
c   ien - iterator for the integration over the series of primary energies
c   istep - integration step
c   xnmu - number of muons
c   dxnmu (2,3) - number of muons from a specific primary energy
c   xmutot (2,3) - total number of muons in the shower
c   2's & 3's - used to calculate the probability of overlap in multiple
c              counters (2 or 3 detectors)

c Subroutine
c   nmucal(en,xnmu) - calculates then number of muons resulting from the
c                     shower of a primary with parameterized energy

        xmutot = 0.
        xmutot2 = 0.
```

```
          xmutot3 = 0.
          istep = 10

c          print *,' en xnmu dxnmu dxnmu2 dxnmu3 xmutot xmutot2 xmutot3 '

c        perform the integration over the primary energies (ien) from 100 GeV
c         to 1000 GeV in increments of 10 GeV, choosing the primary energy
c         value (en) in the middle of the bin by multiplying by 0.5

         do ien = 100,1000,istep
          en = float(ien)-0.5*float(istep)

c          calculate the number of muons resulting from the shower

           call nmucal(en,xnmu)

c         modify by the relative probability of seeing a primary of that
c          specific energy I(E) ~= 1.8E^-2.7 (nucleons/cm^2 * s * sr * GeV)
c          modified appropriately for the 2nd and 3rd counters (subtract 1 & 2)

          dxnmu = xnmu*(en**-2.7)
          dxnmu2 = dxnmu*(xnmu-1.)
          dxnmu3 = dxnmu2*(xnmu-2.)

c         add to totals

          xmutot  = xmutot + dxnmu
          xmutot2 = xmutot2 + dxnmu2
          xmutot3 = xmutot3 + dxnmu3

c         intermediate printing step for verification purposes

          print 90, en,xnmu,dxnmu,dxnmu2,dxnmu3,xmutot,xmutot2,xmutot3
 90       format(f6.0,3f6.3,3f8.2)
         enddo

c        modify the totals by the constants in the expression and unit
c         conversions (10^4: cm^2 -> m^2, 6: ~= 4pi sr -> 2pi sr (only looking
c         at the upper half of the atmosphere), 2: ~= 1.8 constant) & modify
c         by the iteration step and print results

         xmutot = xmutot* 12*10**4*istep
         xmutot2 = xmutot2*12*10**4*istep
         xmutot3 = xmutot3*12*10**4*istep
         print *,' xmutot,xmutot2,xmutot3 ' ,xmutot,xmutot2,xmutot3
         call exit
         end
```

```
c SUBROUTINE: NMUCAL(EN,XNMU)

        subroutine nmucal(en,xnmu)

c energy is in Gev
c Computational equivalent of Eo~=3.9*10^6 GeV (Ne/10^6)^0.9 illegally
c  inverted to Ne~=10^6 (Eo/(3.9*10^6))^1.1) to calculate the number
c  of charged particles resulting from a shower of a primary with (en) energy

        real xne,xnmu,en,t1,t2
        xne = (10**6)*(en/(3.9*10**6))**1.1

c       Nmu(>1Gev)~=0.95*10^5 (Ne/10^6)^0.75 in steps to calculate the number
c        of muons from the calculated number of charged particles

        t1 = xne /(10**6)
        t2 = (t1)**.75
        xnmu = (.95*10**5)*t2

c       intermediate printing step for verification purposes

        print 90, en,xne,t1,t2,xnmu
 90     format(' en,xne,t1,t2,xnmu:'2f10.3,2e12.4,f10.3)
        return
        end
```

11 Appendix C: Program Estimating Shower Properties and Overlap Rates

```
c PROGRAM: COSGRACE.FOR

c this program will calculate quantities related to cosmic rays.
c first will recheck the integral for single detector rates,
c   then will do overlaps.

c Summary of Important Variables
c   d - distance between detectors
c   id - iterator for integration
c   r1 - lower integration limit
c   r2 - upper integration limit
c   nr - integration step
c   t - term value: r^-.75*(1+r/320)^-2.5
c   t2 - term value for second detector
c   r - radius of the shower
c   th - theta
c   nth - integration step for theta
c   tint (1,6) - return values from the separate integrations
c   t2int (1,6) - return values from the separate integrations
c   xnsh - number of showers
c   xov (2) - number of overlaps

c Subroutines
c   intt(r1,r2,nr,tint) - integrates within the given limits the radially
c                         dependent portion of the muon density function
c   term(r,t) - calculates the individual term in accordance with the equation
c   intt2(d,t,r1,r2,th,nr,nth,t2int) - integrates over both the radius and
c                                      theta
c   term2(d,r,t,t2,th) - calculates the individual term for the second
c                        integration, taking into account both counters
c   shower(tint, xnsh) - calculates the number of showers per square meter
c                        from the integration of the muon density function
c   overlap(d,t2int,xnsh,xov) - calculates the number of shower overlaps in
c two counters per square meter per second
c   overlap2(xov,xov2) - takes the energy of the primary into consideration
c  in the calculation of the overlap
c   bkhst - sets up the histograms
c   nstore(r,t,th,t2) - stores the information into the ntuple

Program cosgrace
implicit none
        real r,t,r1,r2,dr,tint1,tint2,tint3,tint4,tint5,tint6,tint
   real th,dth,t2int1,t2int2,t2int3,t2int4,t2int5,t2int6,t2int
real xnsh,gam125,pi
```

```
     real xov
real xov2,d
        integer nr,ir,ic,nth,id

        call bkhst

c      integrate over the separations from 0 - 900 meters converting the
c iterator to the desired integrand by subtracting one and multiplying
c by 100

       do 100 id = 1,10
        d = (id-1)*100.
        nr = 40
        r1 = 0.
        r2 = 100.
        call intt(r1,r2,nr,tint1)
call intt2(d,t,r1,r2,th,nr,nth,t2int1)

nr = 40
        r1 = 100.
        r2 = 320.
        call intt(r1,r2,nr,tint2)
call intt2(d,t,r1,r2,th,nr,nth,t2int2)

nr = 60
        r1 = 320.
        r2 = 640.
        call intt(r1,r2,nr,tint3)
call intt2(d,t,r1,r2,th,nr,nth,t2int3)

nr = 70
r1 = 640.
r2 = 1000.
call intt(r1,r2,nr,tint4)
call intt2(d,t,r1,r2,th,nr,nth,t2int4)

nr = 80
r1 = 1000.
r2 = 10000.
call intt(r1,r2,nr,tint5)
call intt2(d,t,r1,r2,th,nr,nth,t2int5)

nr = 80
r1 = 10000.
r2 = 100000.
call intt(r1,r2,nr,tint6)
call intt2(d,t,r1,r2,th,nr,nth,t2int6)
```

```
c add up the resuting terms

        tint = tint1 + tint2 + tint3 + tint4 + tint5 + tint6
t2int= t2int1 + t2int2 + t2int3 + t2int4 + t2int5 + t2int6

c calculate the shower and overlap numbers

call shower(tint,xnsh)
call overlap(d,t2int,xnsh,xov)
call overlap2(xov,xov2)

c print results

c       print 90, tint1,tint2,tint3,tint4,tint5,tint6,tint
c print 91, t2int1,t2int2,t2int3,t2int4,t2int5,t2int6,t2int
print 92,tint,xnsh
print 93,t2int,xov
        print *,'  integ, limits: 0-100, 100-1000, etc. '
c 90     format(' tint1,tint2,tint3,tint4,tint5,tint6,tint: ',/, 7f8.2)
c 91 format(' t2int1,t2int2,t2int3,t2int4,t2int5,t2int6,t2int:',/,7f8.2)
 92     format(' tint,xnsh:',2f8.2)
 93     format(' t2int,xov:',2f8.4)
 100   continue

c make histograms

        call hprint(0)
        call hcdir('//hists',' ')
        call hrout(0,ic,'t')
        call hrend('//hists')
        call hldir(' ',' ')
        call exit
        END

c SUBROUTINE INTT

        subroutine intt(r1,r2,nr,tint)

c integrates term t from r1 to r2 , nr steps, returns tint (integral)

c set the size of a given interval based on the upper and lower limits
c  and the step size

        dr = (r2-r1)/float(nr)

c sets the starting value to half a step below the lower limit
c  in order to use values from the middle of the bins
```

```
        r = r1 -0.5*dr
        tint=0.

c integrates over the radius of the shower to find the value of the
c  r^-.75*(1+r/320)^-2.5 term of the normalized muon density equation
c  Pm' = 1.25/(2*pi*gamma(1.25))*(1/320)^1.25*r^-.75*(1+r/320)^-2.5
c  by calling the term subroutine

        do 100 ir=1,nr
        r = r + dr
        call term(r,t)
        tint=tint + r*t

c        print 90, ir,r,t,tint
 90     format(' ir,r,t,tint: ', i5,3f8.2)
 100    continue

        tint = tint*dr

c intermediate printing step for verification purposes

        print 91, r1,r2,nr,tint
 91     format(' r1,r2,nr,tint: ',2f10.2,i5,f8.2)
        return
        end

c SUBROUTINE TERM

        subroutine term(r,t)

c integrates over the radius of the shower to find the value of the
c r^-.75*(1+r/320)^-2.5 term of the normalized muon density equation
c Pm' = 1.25/(2*pi*gamma(1.25))*(1/320)^1.25*r^-.75*(1+r/320)^-2.5

        real r,t
        t = r**(-.75)*(1.+r/320.)**(-2.5)

c        print*,'t in term ',t

c create histogram

        call hf1(1,r,t)
        return
        end

c SUBROUTINE INTT2

subroutine intt2(d,t,r1,r2,th,nr,nth,t2int)
```

```
c integrates term t2 from r1 to r2, nr steps, and from th1 to th2 in nth steps,
c returns t2int (integral)

implicit none
integer nth,nr,ir,ith
real d,t,r1,r2,r,th,dr,dth,t2,t2int
th = 0.
nth = 15

c set the size of a given interval based on the upper and lower limits
c  (2pi for theta, accounting for half of the atmosphere) and the step
c  sizes: nr and nth

dr =(r2-r1)/float(nr)
dth = 6.28/float(nth)

c sets the starting value of r to half a step below the lower limit
c  in order to use values from the middle of the bins

r =r1 -0.5*dr
t2int = 0.

c  performs the term integration for two counters over the radius:
c  Pm'(r)*Pm'(r-ro)*r*dr*dth

do 100 ir=1,nr
         r = r + dr

c   sets the starting value of th to half a step below the lower limit(0)
c    in order to use values from the middle of the bins

  th = -.5*dth

c    performs the term integration for two counters over theta:
c    Pm'(r)*Pm'(r-ro)*r*dr*dth

          do 50 ith=1,nth
  th = th + dth

c          print *,' before term2: r,th: ',r,th

  call term2(d,r,t,t2,th)
  t2int=t2int + r*t2

c   print 90, ir,r,th,t2,t2int
90      format(' ir,r,th,t2,t2int: ',i3,4f8.2)
50     continue
```

```
100   continue

t2int = t2int*dr*dth

c intermediate printing step for verification purposes

print 91, r1,r2,nr,nth,t2int
91      format(' r1,r2,nr,nth,t2int:',2f10.2,2i5,f8.2)
return
end

c SUBROUTINE TERM2

        subroutine term2(d,r,t,t2,th)

c integrates over the radius of the shower and theta to find the value of the
c Pm'(r)*Pm'(r-ro)*r*dr*dth integral
c **note: this r2 is how far detector 2 is from the shower center

        real r,t,t2,ta,tb,r2,th,d
        integer ntry
        data ntry/0/
c datac /2./

c to keep track of the number of integrations performed

        ntry=ntry+1

        call term(r,ta)

c        print *,' term2: r,ta ',r,ta

c calculate the distance of detector 2 from the shower center

        r2 = sqrt((r*cos(th) - d)**2 + (r*sin(th))**2)

c        print *,' r2: ',r2

        call term(r2,tb)

        t2 = ta*tb

c intermediate printing of a few of the values for verification purposes

if(ntry.lt.10)print 90,r,ta,r2,tb,t2

 90     format(' r,ta,r2,tb,t2', 5f10.3)
```

```
c make histograms

call hf1(2,r,ta)
call hf1(3,r,tb)
call hf1(4,r,t2)
call ntstor(r,t,th,t2)
        return
        end

c SUBROUTINE SHOWER

subroutine shower (tint,xnsh)

c this sub-routine gives xnsh (number of showers/meter square * second) that
c we expect to see in one counter by dividing the total number of particles by
c tint multiplied by some constants from the equation:
c Pm = (1.25*Nm)/(2*pi*gamma(1.25))*(1/320)^1.25*r^-.75*(1+r/320)^-2.5

        implicit none
        real tint,xnsh
integer xnpr,xnmu

c average number of observed muons

data xnpr/180./

c based on the expected rate of muons at the earth's surface

data xnmu/3./

real gam125
        real p1,p2

c gamma function of 1.25

data gam125/.906/

c calculate the first term: (1.25*Nm)/(2*pi*gamma(1.25))
c  (the 2pi in this case is trivial due to no theta dependence
c   for one counter)

        p1 = (1.25*float(xnmu))/gam125

c calculate the constant term

        p2 = (1./320.)**(1.25)

c intermediate printing for verification purposes
```

```
        print*,'p1',p1
        print*,'p2',p2

c put all the terms of the equation together

xnsh = xnpr/(p1*p2 * tint)

c intermediate printing for verification purposes

print *,' tint,xnsh: ',tint,xnsh
return
end

c SUBROUTINE OVERLAP

subroutine overlap(d,t2int,xnsh,xov)

c this sub-routine gives xov (number of showers overlaps per meter square per
c  second) from dA1*dA2*SI(E)*Nm(E)*[Nm(E)-1]*dE*SPm'(r)*Pm'(r-ro)*r*dr*dth
c  where r*dr*dth = d(shower position)

implicit none
real t2int,xnsh,d
real xnpr,xnmu,xov,xov1
real gam125,pi,const,da1,da2

c  da1,da2 are the surface areas of the counters

        data da1,da2/1.,1./

c  average number of muons observed

data xnpr/180./

c based on the expected rate of muons at the earth's surface

data xnmu/3./

c gamma function of 1.25

data gam125/.906/
data pi/3.14161/

c calculates the constant term of Pm' left out of the integration
c  1.25/(2*pi*gamma(1.25))

        const = 1.25/(2.*pi*gam125)
```

```
c factors the constant term into the Pm'(r)*Pm'(r-ro) integration result
c  the constant term is squared because we are looking at 2 detectors

        xov1 = t2int * (const/320.**1.25)**2

c completes the overlap calculation without consideration for the energy
c  integration

        xov = xov1*da1*da2*xnsh*xnmu*(xnmu-1)

c intermediate printing for verification purposes

        print *,' t2int,const,xov1: ',t2int,const,xov1
print *,'xnsh,xnmu,xov: ',xnsh,xnmu,xov

c factor in energy integration where
c 23227 is sum of rate (by energy) of incident particles * xnmu * xnmu-1

        xov1 =xov1*23227

c intermediate printing for verification purposes

        print *,' d, overlaps expected: ',d,xov1

c       make histogram

       call hf1(5,d,xov1)

return
end

c SUBROUTINE OVERLAP2

subroutine overlap2(xov,xov2)

c This routine will calculate the overlap for different energies

implicit none
real xov,xov2
c xov2 = xov * 2 * 10 **(-7) *(96.6428582959/.305611551805)
c print *,'xov,xov2: ',xov,xov2
return
end

c SUBROUTINE BKHST

       subroutine bkhst
```

```
c  this subroutine sets up histograms for later plotting.

       implicit none
       integer nwpawc,ipaw,n,istat,nvar
       data nvar/4/
       character*4 ntname(4)
       data ntname/ 'r','term','th','term2'/
       parameter (nwpawc = 600000)
       common /pawc/ipaw(nwpawc)

       call hlimit(nwpawc)
       call hropen(34,'hists','cosgrace','n',1024,istat)
       call hcdir('//hists',' ' )
       call hbookn(100,'cosrayn',nvar, '//hists',1000,ntname)
       call hbook1(1,' t vs r ',50,0.,100.0,0.)
call hbook1(2,'ta vs r',50,0.,100.0,0.)
 call hbook1(3,'tb vs r',50,0.,100.0,0.)
  call hbook1(4,'t2 vs r', 50,0.,100.0,0.)
        call hbook1(5, ' overlaps expected vs sep. (m)/ m**2 counters ',
     1      10,0.,1000.,0.)
       return
       end

c SUBROUTINE NTSTOR

       subroutine ntstor(r,t,th,t2)

c stores data in the ntupl

       COMMON/NTSTO/ EPSR, thET, TANG, D2W
       real  r,t,xnt(4)
       double precision beta

       xnt(1) =  r
       xnt(2) =  t
xnt(3) = th
xnt(4) = t2
c xnt(5) = xnsh

       call hfn(100,xnt)
       return
       end
```

12 Appendix D: Design of Scintillator Supports

13 Appendix Ea: Siuedat.for

```
     Program SIUEDAT
        integer*2 ix(24)
c this version modified july 4, 2001, jt, to accomodate new data from dek
c this version modified july 26, 2001, ddt, to accomodat new data from dek
c the first 6 adc's are unattenuated, the next 6 are attenuated
c the next 6 are tdc's, and the next 6 are scalars
c  modified aug. 9: scalers read first, then adcs, then tdcs
        integer*4  iwork,nevt,ntot
        data ntot,iwork/1000,0/

        call huinit
        call eread(0,ix,iwork)
        print *,' ix: ',ix

       do  nevt=1,ntot
       call eread (1, ix, iwork)
       if(iwork.eq.0)goto 1000
       if((nevt.lt.200).or.(mod(nevt,100).eq.0))
     1    print 91, nevt,ix
 91   format(' nevt: ',i5, ' scaler: ',6i7,/,' adc: ',6i5,
     1  /,' attenadc: ',/,6i5,' tdc: ',6i5)
        call nstor(ix)
      enddo

 1000   continue
       call hcdir('//pmt',' ')
       call hrout(0,ic,'t')
       call hrend('//pmt')
       call hldir(' ',' ')
       call hprint(0)
       end
       subroutine nstor(ix)
       real x(24)
       integer*2 ix(24)
       integer nevt,i
       nevt=nevt+1
       do i=1,24
        x(i) = ix(i)
       if(i.gt.6) call hf1(i-6,x(i),1.)
       if(i.le.6)call hf1(i+18,x(i),1.)
       enddo
      if(x(1).ne.0.)call hfn(100,x)
       return
       end
       subroutine huinit
       parameter (nwpawc = 600000)
```

```
      common /pawc/ipaw(nwpawc)

       character*5 nampmt(24)
       integer*4 nnam
       data nnam/24/
       data nampmt/
     1      'sca1','sca2','sca3','sca4','sca5','sca6',
     1      'adc1','adc2','adc3','adc4','adc5','adc6',
     1      'atdc1','atdc2','atdc3','atdc4','atdc5','atdc6',
     2      'tdc1','tdc2','tdc3','tdc4','tdc5','tdc6'/

       call hlimit(nwpawc)
c now make new directory for histograms, in order to write out
c  ntuple as we go, not save it in memory.
       call hlimit(nwpawc)
       call hropen(34,'pmt','siuecosray.hst','n',1024,istat)
       print *,' after hropen '
       call hbookn(100,'pmt',nnam,'pmt',1000,nampmt)
       call hbook1(1,' adc1 ',40,0.,1200.,0)
       call hbook1(2,' adc2 ',40,0.,1200.,0)
       call hbook1(3,' adc3 ',40,0.,1200.,0)
       call hbook1(4,' adc4 ',40,0.,1200.,0)
       call hbook1(5,' adc5 ',40,0.,1200.,0)
       call hbook1(6,' adc6 ',40,0.,1200.,0)
       call hbook1(7,' atdc1 ',40,0.,1200.,0)
       call hbook1(8,' atdc2 ',40,0.,1200.,0)
       call hbook1(9,' atdc3 ',40,0.,1200.,0)
       call hbook1(10,' atdc4 ',40,0.,1200.,0)
       call hbook1(11,' atdc5 ',40,0.,1200.,0)
       call hbook1(12,' atdc6 ',40,0.,1200.,0)
       call hbook1(13,' tdc1 ',40,50.,4000.,0)
       call hbook1(14,' tdc2 ',40,50.,4000.,0)
       call hbook1(15,' tdc3 ',40,50.,4000.,0)
       call hbook1(16,' tdc4 ',40,50.,4000.,0)
       call hbook1(17,' tdc5 ',40,50.,4000.,0)
       call hbook1(18,' tdc6 ',40,50.,4000.,0)
       call hbook1(19,' sca1 ',40,50.,4000.,0)
       call hbook1(20,' sca2 ',40,50.,4000.,0)
       call hbook1(21,' sca3 ',40,50.,4000.,0)
       call hbook1(22,' sca4 ',40,50.,4000.,0)
       call hbook1(23,' sca5 ',40,50.,4000.,0)
       call hbook1(24,' sca6 ',40,50.,4000.,0)
       return
       end
```

14 Appendix Eb: Stack.kumac

```
macro stack  run=''
*** Begin of history file: Wed Jun 30 22:38:17 1999
hi/fil 1 run99982.hst
hi/list
nt/print 100
** 31 etc for adc correl.
2dhist 31 ' adc1 vs atdc1 ' 40 0. 1200. 40 0. 1200.
nt/proj 31 100.adc1%atdc1
2dhist 32 ' adc2 vs atdc2 ' 40 0. 1200. 40 0. 1200.
nt/proj 32 100.adc2%atdc2
2dhist 33 ' adc3 vs atdc3 ' 40 0. 1200. 40 0. 1200.
nt/proj 33 100.adc3%atdc3
2dhist 34 ' adc4 vs atdc4 ' 40 0. 1200. 40 0. 1200.
nt/proj 34 100.adc4%atdc4
2dhist 35 ' adc5 vs atdc5 ' 40 0. 1200. 40 0. 1200.
nt/proj 35 100.adc5%atdc5
2dhist 36 ' adc6 vs atdc6 ' 40 0. 1200. 40 0. 1200.
nt/proj 36 100.adc6%atdc6
*2dhist 22 ' adc6 vs adc1 ' 40 0. 1200. 40 0. 1200.
**nt/proj 22 100.adc6%adc1
** 31  etc for tdc correl
*2dhist 31 ' tdc6 vs tdc1 ' 40 1000. 1400. 40 1000. 1400.
*nt/proj 31 100.tdc6%tdc1
*2dhist 32 ' tdc4 vs tdc2 ' 40 0. 2000. 40 1000. 1400.
*nt/proj 32 100.tdc4%tdc2
*2dhist 33 ' tdc5 vs tdc2 ' 40 0. 2000. 40 1000. 1400.
*nt/proj 33 100.tdc5%tdc2
** 41, etc, for adc-tdc correlations
2dhist 41 ' adc1 vs tdc1 ' 40 1000. 1400. 40 0. 1000.
*nt/proj 41 100.adc1%tdc1
2dhist 42 ' adc2 vs tdc2 ' 40 0. 2000. 40 0. 1000.
*nt/proj 42 100.adc2%tdc2
** 51, etc for time  differences
1dhist 51 ' tdc6-tdc1 ' 40 -100. 100.
*nt/proj 51  100.tdc6-tdc1
**  61, etc for time difference correlations
2dhist 61 ' tdc5-tdc2  vs tdc4-tdc2  ' 40 -400. 400. 40 -400. 400.
*nt/proj 61 100.(tdc5-tdc2)%(tdc4-tdc2)
2dhist 62 ' tdc3-tdc2  vs tdc5-tdc4  ' 40 -400. 400. 40 -400. 400.
*nt/proj 62 100.(tdc3-tdc2)%(tdc5-tdc4)
2dhist 63 ' tdc3m2a-tdc5m4 vs tdc5m4 ' 40 -400. 400. 40 -400. 400.
2dhist 64 ' tdc3m2a,tdc5m4 avg vs tdc5m4 ' 40 -400. 400. 40 -400. 400.
2dhist 65 ' tdc3m2a-0.8*tdc5m4 vs tdc5m4 ' 40 -400. 400. 40 -400. 400.
2dhist 66 ' tdc3m2a,0.8*tdc5m4 avg vs tdc5m4 ' 40 -400. 400. 40 -400. 400.
**** now put inselection criteria
cut $1 (tdc3-tdc2).gt.-200..and.(tdc3-tdc2).lt.0.
```

```
cut $2 (tdc5-tdc4).gt.-200..and.(tdc5-tdc4).lt.0.
cut $3 (tdc3-tdc2).gt.0..and.(tdc3-tdc2).lt.200.
cut $4 (tdc5-tdc4).gt.0..and.(tdc5-tdc4).lt.200.
1dhist 72 ' adc2, tdc3-2 and tdc5-4  -200-0' 40 0. 1600.
1dhist 73 ' adc3, tdc3-2 and tdc5-4  -200-0' 40 0. 1600.
1dhist 74 ' adc4, tdc3-2 and tdc5-4  -200-0' 40 0. 1600.
1dhist 75 ' adc5, tdc3-2 and tdc5-4  -200-0' 40 0. 1600.
*nt/proj 72 100.adc2 $1.and.$2
*nt/proj 73 100.adc3 $1.and.$2
*nt/proj 74 100.adc4 $1.and.$2
*nt/proj 75 100.adc5 $1.and.$2
1dhist 82 ' adc2, tdc3-2 and tdc5-4  0-200' 40 0. 1600.
1dhist 83 ' adc3, tdc3-2 and tdc5-4  0-200' 40 0. 1600.
1dhist 84 ' adc4, tdc3-2 and tdc5-4  0-200' 40 0. 1600.
1dhist 85 ' adc5, tdc3-2 and tdc5-4   0-200' 40 0. 1600.
1dhist 92 ' adc2, tdc3-2 lt.0, tdc5-4.gt.0' 40 0. 1600.
1dhist 93 ' adc3, tdc3-2 lt.0, tdc5-4.gt.0' 40 0. 1600.
1dhist 94 ' adc4, tdc3-2 lt.0, tdc5-4.gt.0' 40 0. 1600.
1dhist 95 ' adc5, tdc3-2 lt.0, tdc5-4 .gt.0' 40 0. 1600.
** now use adjusted avg. time diffs (gives rough  position)
prof 102 ' adc2 vs avg tavg' 40 -400. 400.  0. 1600.
prof 103 ' adc2 vs avg tavg' 40 -400. 400.  0. 1600.
nt/proj 103 100.adc2%(0.5*((tdc3-tdc2+125.)+0.8*(tdc5-tdc4)))
1dhist 122 ' adc2 for tavg lt.-100' 40 0. 1600.
1dhist 123 ' adc2 for tavg -100 to 0.' 40 0. 1600.
1dhist 124 ' adc2 for tavg 0-100' 40 0. 1600.
1dhist 125 ' adc2 for tavg gt 100' 40 0. 1600.
*nt/proj 82 100.adc2 $3.and.$4
*nt/proj 83 100.adc3 $3.and.$4
*nt/proj 84 100.adc4 $3.and.$4
*nt/proj 85 100.adc5 $3.and.$4
1dhist 130 ' tgps ' 40 0. 100000000.
1dhist 131 ' scaler 1 ' 40 -10000.  70000.
1dhist 132 ' scaler 2 ' 40 -10000.  70000.
1dhist 133 ' scaler 3 ' 40 -10000.  70000.
1dhist 134 ' scaler 4 ' 40 -10000.  70000.
1dhist 135 ' scaler 5 ' 40 -10000.  70000.
1dhist 136 ' scaler 6 ' 40 -10000.  70000.
1dhist 141 ' adc1, no tdc1 ' 40 0. 400.
1dhist 142 ' adc2, no tdc2 ' 40 0. 400.
1dhist 143 ' adc3, no tdc3 ' 40 0. 400.
1dhist 144 ' adc4, no tdc4 ' 40 0. 400.
1dhist 145 ' adc5, no tdc5 ' 40 0. 400.
1dhist 146 ' adc6, no tdc6 ' 40 0. 400.
1dhist 151 ' atdc1, no tdc1 ' 40 0. 400.
1dhist 152 ' atdc2, no tdc2 ' 40 0. 400.
1dhist 153 ' atdc3, no tdc3 ' 40 0. 400.
1dhist 154 ' atdc4, no tdc4 ' 40 0. 400.
```

```
1dhist 155 ' atdc5, no tdc5 ' 40 0. 400.
1dhist 156 ' atdc6, no tdc6 ' 40 0. 400.
1dhist 161 ' adc1, with tdc1 ' 40 0. 2000.
1dhist 162 ' adc2, with tdc2 ' 40 0. 2000.
1dhist 163 ' adc3, with tdc3 ' 40 0. 2000.
1dhist 164 ' adc4, with tdc4 ' 40 0. 2000.
1dhist 165 ' adc5, with tdc5 ' 40 0. 2000.
1dhist 166 ' adc6, with tdc6 ' 40 0. 2000.
1dhist 171 ' atdc1, with tdc1 ' 40 0. 400.
1dhist 172 ' atdc2, with tdc2 ' 40 0. 400.
1dhist 173 ' atdc3, with tdc3 ' 40 0. 400.
1dhist 174 ' atdc4, with tdc4 ' 40 0. 400.
1dhist 175 ' atdc5, with tdc5 ' 40 0. 400.
1dhist 176 ' atdc6, with tdc6 ' 40 0. 400.
1dhist 181 ' ncnt ' 40 0. 10.
1dhist 19 'dummy' 40 0. 1000.
nt/proj 19 100.adc1 selstack.for
close 1
hi/fil 1 run99983.hst
nt/proj 19 100.adc1 selstack.for
shell rm -f  run99982-3_stack_old.ps
shell mv  run99982-3_stack.ps run99982-3_stack_old.ps
fort/file 2 run99982-3_stack.ps
metafile 2 -111
opt  liny
opt stat
set stat 111111111
opt date
opt file
opt liny
zone 2 2
hi/pl 1
hi/pl 2
hi/pl 3
hi/pl 4
hi/pl 5
hi/pl 6
hi/pl 7
hi/pl 8
hi/pl 9
hi/pl 10
hi/pl 11
hi/pl 12
hi/pl 13
hi/pl 14
hi/pl 15
hi/pl 16
hi/pl 17
```

```
hi/pl 18
hi/pl 19
hi/pl 20
hi/pl 21
hi/pl 22
hi/pl 23
hi/pl 24
zone 2 2
hi/pl 31 box
hi/pl 32 box
hi/pl 33 box
hi/pl 34 box
hi/pl 35 box
hi/pl 36 box
*hi/pl 32 box
*hi/pl 33 box
*igset mtyp=26
*hi/pl 18
*hi/pl 19 ps
*hi/pl 41 box
*hi/pl 42 box
*hi/pl 51 box
*hi/pl 61 box
*hi/pl 62 box
*hi/pl 63 box
*hi/pl 64 box
*hi/pl 65 box
*hi/pl 66 box
zone 2 2
*hi/pl 72
*hi/pl 73
*hi/pl 74
*hi/pl 75
*hi/pl 82
*hi/pl 83
*hi/pl 84
*hi/pl 85
zone 2 2
*hi/pl 92
*hi/pl 93
*hi/pl 94
*hi/pl 95
zone 2 2
*hi/pl 102
*hi/pl 103
*hi/pl 122
*hi/pl 123
*hi/pl 124
```

```
*hi/pl 125
hi/pl 130
hi/pl 131
hi/pl 132
hi/pl 133
hi/pl 134
hi/pl 135
hi/pl 136
hi/pl 141
hi/pl 142
hi/pl 143
hi/pl 144
hi/pl 145
hi/pl 146
hi/pl 151
hi/pl 152
hi/pl 153
hi/pl 154
hi/pl 155
hi/pl 156
hi/pl 161
hi/pl 162
hi/pl 163
hi/pl 164
hi/pl 165
hi/pl 166
hi/pl 171
hi/pl 172
hi/pl 173
hi/pl 174
hi/pl 175
hi/pl 176
hi/pl 181
close 2
return
*** End   of history file: Wed Jun 30 22:55:08 1999
```

15 Appendix Ec: Selstack.for

```
      REAL FUNCTION selstack()
      REAL
     +sca1    ,sca2    ,sca3    ,sca4    ,sca5    ,sca6    ,
     +adc1    ,adc2    ,adc3    ,adc4    ,adc5    ,adc6    ,
     +atdc1   ,atdc2   ,atdc3   ,atdc4   ,atdc5   ,atdc6   ,
     +tdc1    ,tdc2    ,tdc3    ,tdc4    ,tdc5    ,tdc6
*
      LOGICAL         CHAIN
      CHARACTER*128   CFILE
*
      COMMON /PAWCHN/ CHAIN, NCHEVT, ICHEVT
      COMMON /PAWCHC/ CFILE
*
      COMMON/PAWIDN/IDNEVT,OBS(13),
     +sca1    ,sca2    ,sca3    ,sca4    ,sca5    ,sca6    ,
     +adc1    ,adc2    ,adc3    ,adc4    ,adc5    ,adc6    ,
     +atdc1   ,atdc2   ,atdc3   ,atdc4   ,atdc5   ,atdc6   ,
     +tdc1    ,tdc2    ,tdc3    ,tdc4    ,tdc5    ,tdc6
*
      real sca(6),adc(6),atdc(6),tdc(6),tgps
      real thradc(6),thratdc(6)
c sca is the scaler array, and tgps is the gps time inferred
      equivalence (sca1,sca(1)), (adc1,adc(1)), (atdc1,atdc(1))
     1           , (tdc1,tdc(1))
      integer iok,ncnt
      vector flg(2)

*
      selstack=1.
      run=flg(1)
      spread=flg(2)

      if((run.eq.99996).or.(run.eq.99997))
     1    call swtchij(1,5,adc,tdc,atdc)
      if((spread.eq.0).and.
     1     ((tdc(1).gt.3000.).or.(tdc(6).gt.3000.)))return

c now get scaler (time) info
      call tscal(sca,tgps)
c tgps will be returned negative if there is  problem with the
c decoding.
       call pedestal(adc,atdc,tdc,ncnt)
       call evtchk(adc,atdc,tdc,sca,ncnt,iok)
      END
      subroutine evtchk(adc,atdc,tdc,sca,ncnt,iok)
      real adc(6),atdc(6),tdc(6),sca(6),r
```

```
      integer iok,nfunny,nevtchk
      vector flg(2)
      data nevtchk/0/
c this routine checks for consistency between atdc and adc values.
      run= flg(1)
      spread=flg(2)
      nfunny=0
      nevtchk=nevtchk+1
      do 100 i=1,6
      if(tdc(i).gt.3000.) go to 100
      r=adc(i)/atdc(i)
      call hf1(70+i,r,1.)
      if((r.lt.2.8).or.(r.gt.5.2))then
        nfunny=nfunny+1
        go to 100
      endif
       do 50  j=i+1,6
        if(tdc(j).gt.3000.) go to 50
         r=adc(j)/atdc(j)
         ih0=200 + 10*i + j
         if(nevtchk.lt.5) print 90, run,nevtchk,ih0,spread,i,
     1    adc(i),atdc(i),tdc(i),j,adc(j),atdc(j),tdc(j)
 90       format(' run, nevtchk,ih0,spread: ',f9.0,2i5,f5.0
     1    /,' i,adc,atdc,tdc: ',i5,3f6.0,
     1    /,' j,adc,atdc,tdc: ',i5,3f6.0)
         if((r.gt.2.8).and.(r.lt.5.2))
     1     call hfill(ih0,adc(i),adc(j),1.)
  50     enddo
 100  enddo
      call hf1(77,float(nfunny),1.)
      call hfill(78,float(ncnt),float(nfunny),1.)
      ds13 = sca(1)-sca(3)
      ds26 = sca(2)-sca(6)
      call hfill (80,ds13,ds26,1.)
      if((ds13.ne.0.).and.(ds26.ne.0.))then
       print 91, nevtchk,(sca(i),i=1,6)
 91    format(' scaler failure, nevtchk, sca: ',i5,6f7.0)
       call hf1(79,float(nevtchk),1.)
       endif
      return
      end

      subroutine swtchij(i,j,adc,atdc,tdc)
      integer i,j
      real adc(6),atdc(6),tdc(6)
      real swtmp
      integer nswap
      data nswap/0/
```

```
c switches info for channels i,j.

      if(nswap.lt.5) print 90, i,j,adc(i),adc(j),atdc(i),atdc(j)
 90   format(' swapping: i,j: ',2i3, ' adc,atdc: ',4f6.0)
      swtmp=adc(i)
      adc(i)=adc(j)
      adc(j)=swtmp

      swtmp=atdc(i)
      atdc(i)=atdc(j)
      atdc(j)=swtmp

      swtmp=tdc(i)
      tdc(i)=tdc(j)
      tdc(j)=swtmp

      if(nswap.lt.5) print 91, i,j,adc(i),adc(j),atdc(i),atdc(j)
 91   format(' swapped: i,j: ',2i3, ' adc,atdc: ',4f6.0)
      return
      end

      subroutine tscal(sca,tgps)
      real sca(6),tgps
c this routine combines scalers to find an event time. jt. 30 jly 2001
c suppose sca(1) counts resets which effectively counts large scale
c time periods.  sca(2) and sc(3) are one pair (the same scaler
c read at different time s. sca(4) and sca(5) are another pair.
c the pair for which the one read later doesn't change from the
c one read at the time of the event trigger
c is the pair which gives the event time.
c  for the present we approximate the time by multiplying sca(1)
c by 1000000 (to get microseconds) and add whichever of the
c two pairs does not change.  if the scalers are negative
c then a negative (error) time is returned.
          integer nevent,isca(6)

        tgps = -1000000.
        nevent = nevent + 1
       do i=1,6
         isca(i) = sca(i)
         if (sca(i).lt.0.)isca(i)=  65536 + isca(i)
         sca(i) = isca(i)
c 1 aug 2001,dt,jth, after testing on ti2.for in jth.phyast.pitt.edu:
c  jth/tmp
         call hf1(130+i,sca(i),1.)
       enddo
        if(nevent.lt.100)then
```

```
         print 90, sca
 90      format(' scalers: ',6f10.0)
c         if(nevent.lt.100) print 91,isca
 91      format(' scalers: ',6o10,/,6o13,/,' tgps: ', e12.4)
       endif
        if(sca(3).eq.sca(2)) tgps = 1000000.*sca(1) +sca(2)
        if(sca(5).eq.sca(4)) tgps = 1000000.*sca(1) + sca(4)

      return
      end

       subroutine pedestal(adc,atdc,tdc,ncnt)
       real adc(6),atdc(6),tdc(6),aped(6),atped(6),thradc(6),thratdc(6)
c  now look for pedestals
        data atped/0.,62.,65.,61.,15.,125./
        data aped/0.,165.,40.,74.,46.,56./
        data thradc/50.,40.,40.,0.,40.,40./
       integer ncnt
       ncnt = 0
       do i=1,6
         if ( (tdc(i).lt.0).OR.(tdc(i).gt.3000.)) then
           call hf1(140+i,adc(i),1.)
           call hf1(150+i,atdc(i),1.)
         endif
           adc(i) = adc(i)-aped(i)
           atdc(i) = atdc(i)-atped(i)
         if ( (tdc(i).gt.0).AND.(tdc(i).lt.3000.)) then
           call hf1(160+i,adc(i),1.)
           call hf1(170+i,atdc(i),1.)
         if (adc(i).gt.thradc(i)) ncnt = ncnt+1
         endif

        enddo
        call hf1(181,float(ncnt),1.)

       return
       end
```

16 Appendix F: Rate Calculations, Including Cos^2 Dependence

PDF files: rmustack.pdf and rmucalchess.pdf are on siue.edu/∼juthomp/html/2001docs.

17 Appendix G: Scintillator Supports

PDF files (scintsupporta.pdf and scintsupportb.pdf) are on siue.edu/∼juthomp/2001docs.

References

[1] M. Nagano and A.A. Watson, "Observations and implications of the ultrahigh-energy cosmic rays", Reviews of Modern Physics **72**, 689 (2000).

[2] Particle Data Book comilation, 2000.

Contents

List of Figures

List of Tables