LGB Time of Flight Facility

Nicole Quist

A senior thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Bachelor of Science

Dr. Lawrence Rees, Advisor

Department of Physics and Astronomy

Brigham Young University

April 2013

ABSTRACT

LGB Time of Flight Facility

Nicole Quist
Department of Physics and Astronomy
Bachelor of Science

The double pulse response of the gadolinium lithium borate cerium detector suggested its effectiveness for applications in Laboratory Nuclear Astrophysics, specifically in sparse neutron spectroscopy amidst the competition of background radiation. Using Cf-252 as a neutron source, a time of flight facility was built to determine the efficiency of the LGB detector. The detector has a 10% efficiency in neutron detection, however, the energy detected from pulse area correlated with the calculated time of flight energy only 2% of the time. Thus, the total efficiency for neutron spectroscopy is 0.2%. Therefore, the LGB detector is not a good choice for neutron spectroscopy.

Keywords: LGB, Scintillator, Neutron Detection, Spectroscopy

ACKNOWLEDGMENTS

I would like to thank all those who have helped me in this endeavor. I would like to thank John Ellsworth for his mentoring and guidance. I would also like to thank Dr Lawrence Rees for serving as my advisor. I also appreciate Dr J Bart Czirr for answer technical questions. Finally, I would like to thank the BYU Physics and Astronomy department for the funding and support provided.

Contents

Chapter 1

Introduction

1.1 Motivation

Stellar nuclear reactions result from low energy reactants and have low reaction rates. Laboratory Nuclear Astrophysics research focuses on reproducing these reactions in the laboratory at energy characteristic of their reactants. Measuring these reactions in the lab is made difficult by competition from background radiation interference and the low reaction rates consequential of low energy (Rolfs et al. 1987). Much of the empirical knowledge about stellar nuclear reactions is extrapolated from higher energy reactions. A more accurate understanding of these reactions requires a detector with the ability to distinguish between background radiation and experimental neutrons.

The lithium gadolinium borate cerium (LGB) detector promised to be useful for neutron spectroscopy in this energy range. It also has potential applications in other areas of nuclear research as well as national security, and nuclear proliferation prevention.

1.2 Background

1.2.1 Neutron Spectroscopy

The energy of a neutron is difficult to measure, but very important to nuclear research and industry. Its charge neutrality precludes using charged particle detection techniques. It is, therefore, necessary to detect neutrons as secondary interactions with other atoms and molecules. The lithium gadolinium borate cerium (LGB) detector contains two materials with which neutrons interact and produce light.

The first material is the plastic scintillator made of polyvinyl-toluene ($C_6H_4(CH_3)_2$) from Eljen product number EJ290 (Technology 2010). This is a plastic which is composed primarily of hydrogen and carbon. Hydrogen, the element with the smallest nucleus, can transfer up to 100% of a neutron's energy, the highest of any element (Wallace 2011). When a neutron collides with a hydrogen atom, energy is transferred to the proton, which in turn releases the energy in the form of light. However, the proportionality of the light to the energy decreases in the low energy range (Fig. 1.1). Therefore, if there are multiple collisions between the neutron and hydrogen atoms, a portion of the neutron energy is lost to the system. The pulses from the neutron-hydrogen interactions are collected by photomultiplier tubes (PMT) as a light pulse. The light from these interactions has a short decay time, so the resulting thermal pulse is narrow and somewhat proportional to the original energy of the neutron.

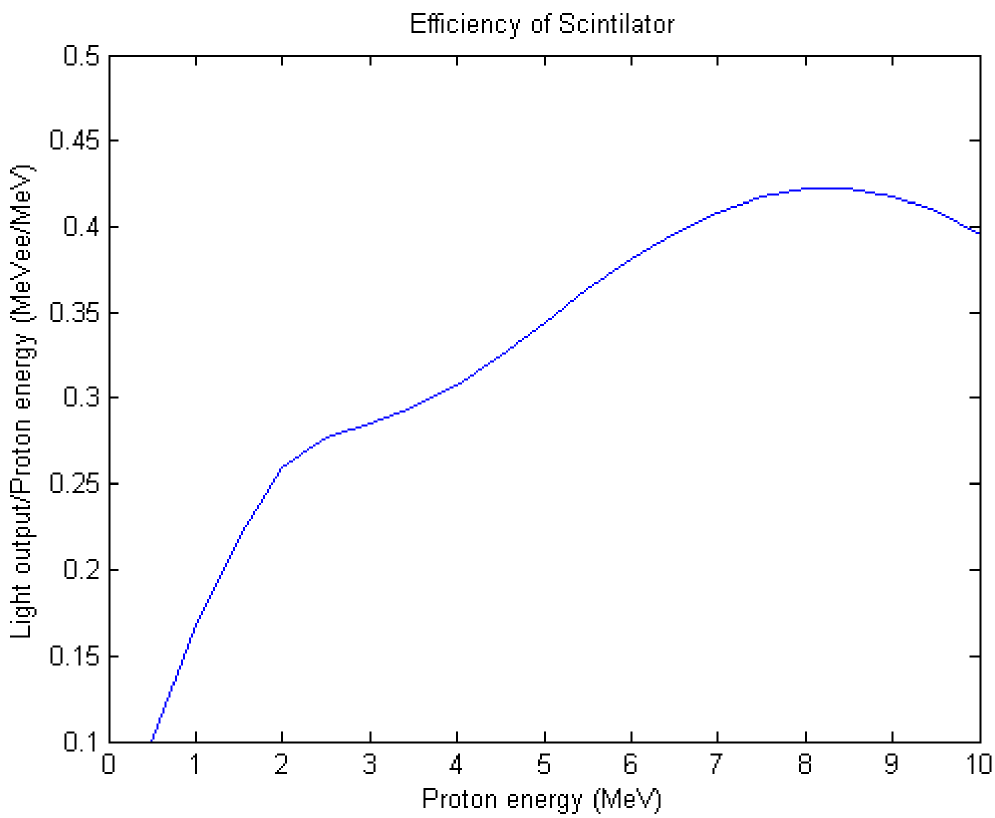

Figure 1.1 The plastic efficiency of as a function of energy. As the energy of the proton decreases, the light output drops. (Czirr 2013)

The decrease of neutron energy increases the cross section for the lithium and the boron in the crystal (Rolfs et al. 1987). This crystal is made of lithium gadolinium borate and doped with cerium, $Li_6Gd(BO_3)_3 : Ce$, (Fig. 1.2). It was grown at Institut de Chimie de la Matière Condensée de Bordeaux, France and the crystals and was specifically tested at BYU in a light tight box. The LGB crystal is unique. When it interacts with neutrons, it produces seven times more light than lithium-6 glass, a material traditionally used in neutron detection (Fig. 1.3).

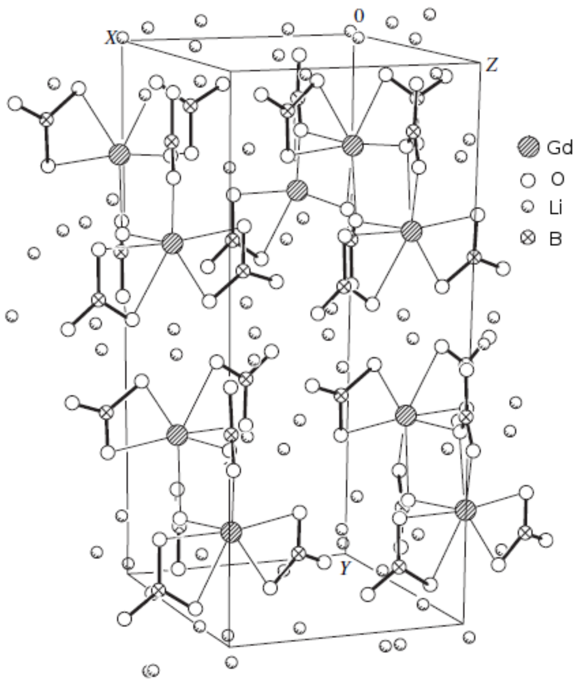

Figure 1.2 The atomic structure of the $Li_6Gd(BO_3)_3$: Ce crystal (Dolzhenkova et al. 2003)

LiGdBCe Crystals grown at Institut de Chimie de la Matière Condensée de Bordeaux, France for Photogenics, Inc.						
Tested at BYU LNAR Lab Oct 8, 2007						
Lithium glass and then LGB crystals where each in-turn affixed with optical grease to the horizontal face of a PMT place in a dark box and exposed to a Cf252 source. Data was take with an ORTEC Trump MCA card and spectroscopy software.						
Sample	Amplifier Gain	Li Pulse Height	Li gain	B Pulse Height	B gain	Relative Brightness to Li Glass Cz/GS-20
GS-20 Li Glass	.40X300	259	2.158			
Cz272	.6X30	271	15.056	67	3.72	6.98
Cz274	.6X30	248	13.778	63	3.50	6.38
Cz275	.6X30	252	14.000	64	3.56	6.49
Cz283	.6X30	251	13.944	65	3.61	6.46
Cz277	.6X30	259	14.389	66	3.67	6.67
Cz278	.6X30	115	6.389	too low		2.96
Cz279	.6X30	249	13.833	63	3.5	6.41
Cz276A	.6X30	181	10.056	52	2.88889	4.66
Cz276B	.6X30	170	9.444	too low		4.38
NorthrupGrumman small	.6X30	63	3.500	too low		1.62

Figure 1.3 Data on LGB crystal tests

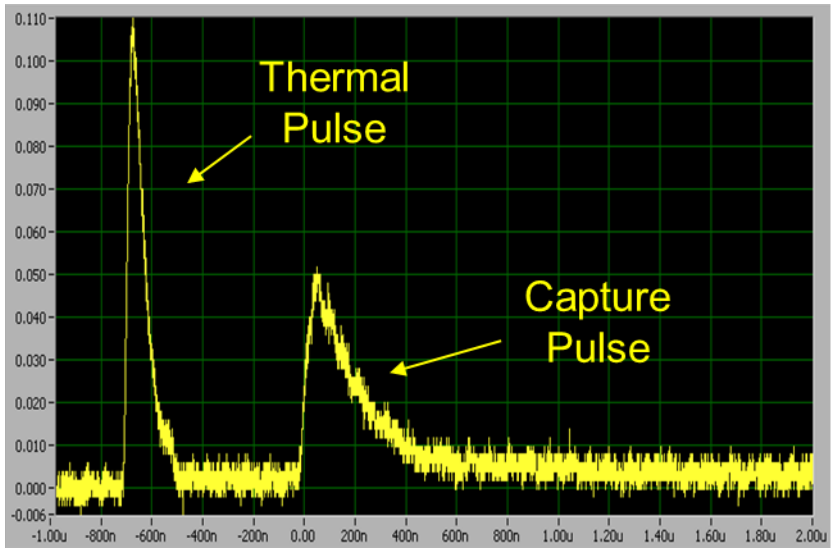

Figure 1.4 The first pulse shape is the thermal pulse which results from the neutron-plastic interaction. The second pulse, capture pulse, is the neutron-LGB crystal interaction which has a longer decay which makes the signature shape.

When the Li-6 or the B-10 in the crystal absorbs a neutron, the atoms fission and produce energetic charged particles which excite the surrounding electrons in the crystal (Ellsworth 2013). This reaction produces a light pulse with decay time much longer in comparison to the thermal pulse and constitutes a capture pulse with a long tail as its signature shape. The presence of this signature pulse is the indication of neutron capture. The correlation in time of the thermal pulse and the signature pulse provides a high probability of neutron identification and the height or area of the first pulse providing energy information (Fig. 1.4).

1.2.2 Energy Detection

An important part of nuclear reaction identification is determining the neutron energy. The specific neutrons we are looking for fall in the 2 MeV range. Once a neutron is detected, the energy information needs to be extracted to determine if it is significant to the experiment.

The primary method for energy detection is through the light output from the scintillator. Since the neutron loses energy in the detection process and the energy is then turned into light, the amount of light detected is somewhat proportional to the energy of the incident neutron. The accuracy of this detection method is dependent on the energy efficiency of the detector which is derived from the efficiency of the polyvinyl-toluene. This efficiency is the ratio of the light output to the light output from a single collision.

In order to determine the accuracy of the detector, a secondary approach was used simultaneously. This approach is to measure the energy of a neutron by its time of flight. This setup requires two triggers, a start and a stop value. The neutron triggers the start value with a detector at a known distance. The neutron then travels a specified distance before detection by the stop detector. The timing between pulses and the distance traveled allows for the calculation of the neutron's energy. This is more accurate for measuring energy and will be used as a comparison to the energy measured with our detector.

Chapter 2

Experimental Method

2.1 Experimental Setup

2.1.1 Configuration

In addition to the LGB detector, we used a sodium iodide (NaI) detector in our time of flight facility. The NaI detector responds to gamma rays and produced our start pulse for the time of flight calculations. The two detectors were placed a meter apart on a rack facing each other. The source was placed at the face of the NaI detector (Fig. 2.1). When a neutron event occurs, the NaI detector will detect an accompanying gamma and start the timing equipment. The neutron will then travel the distance between the detectors (1 meter) before detection by the LGB detector.

Integral to each detector is a PMT. PMTs converted the collected light from the scintillator interactions into analog pulses. The analog signals were then processed by nuclear instrumentation module (NIM) equipment before being digitized. First, both NaI and LGB signals were amplified and then each signal was then split into two signals, with one of the signals going into the Caen Digitizer. The two other signals are sent through constant fraction discriminators. The constant fraction discriminator (CFD) produces a narrow timing pulse at a constant fraction of the peak

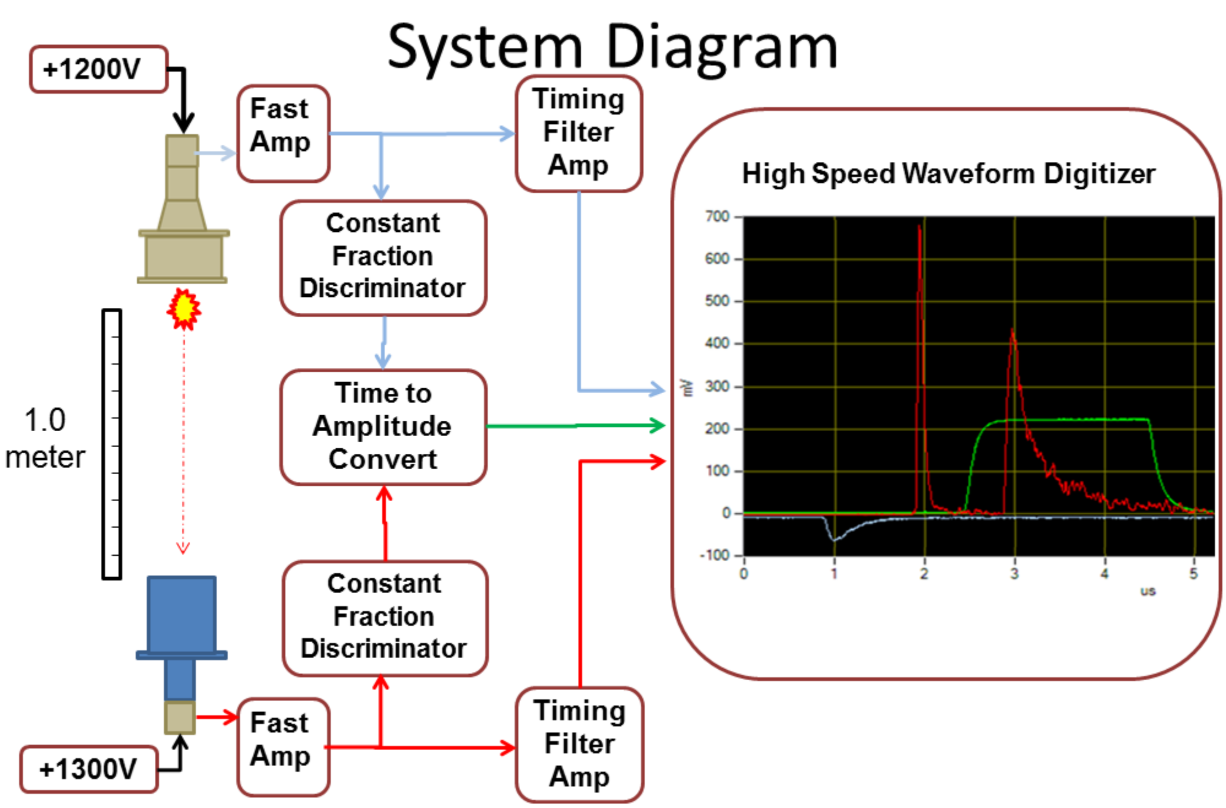

Figure 2.1 Basic Setup for the BYU Time of Flight Facility (see appendix A)

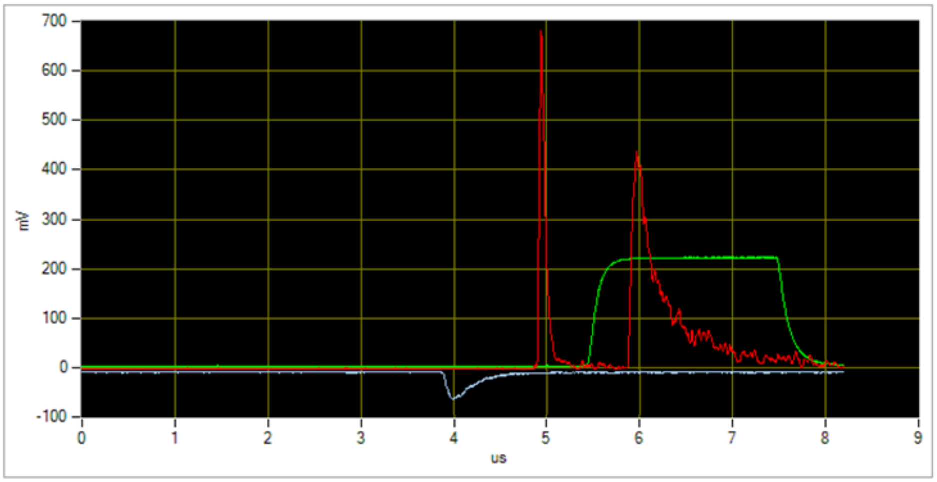

Figure 2.2 The three pulses from our system. Red is LGB detector, Green is the TAC, white is the NaI detector.

height. These time pulses were then fed into a time to amplitude converter (TAC). The TAC takes in two pulses within a specified time window and produces a pulse with amplitude proportional to the time between pulses. This signal was then sent to the digitizer (Fig. 2.2).

A 12 bit, 4 channel, 250 Ms/s digitizer converts the analog signal into a digital signal. We used three inputs; one input collected the signal from the LGB detector, one collected the signal from the NaI detector, and the third input was for the TAC pulse. We set it to trigger when there was a pulse from the LGB detector. Neil Turley wrote the program that runs the digitizer and collects the data.

2.1.2 Neutron Source

Californium 252 was our neutron source. Californium 252 is an unstable isotope with a half-life of about 2.73 years. About 97% of Cf-252 decays are alpha decays, but the other 3% of the time are spontaneous nuclear fission. This produces an average of 3.76 neutrons and 9 gammas per fission in the energy range we are looking for with our detectors. The neutrons produced by Cf-252 have a wide energy spectrum as seen in in figure 2.3. The majority of the neutrons fall in the energy

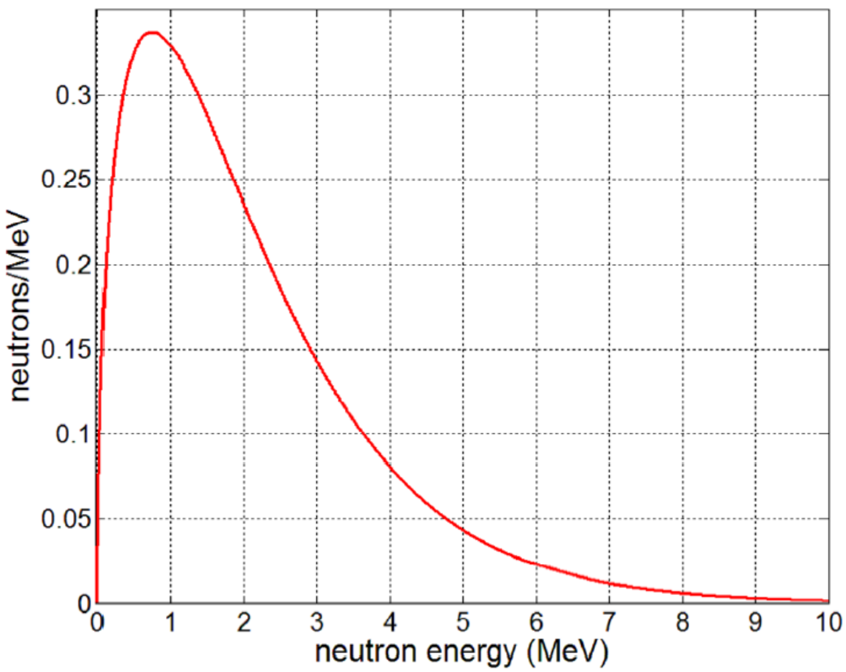

Figure 2.3 Energy of neutrons resulting from Cf-252 fission decays (Wallace 2011)

range of 1-10 MeV with an average of 0.7 MeV.

With the information on the activity rate at a specific day as well as the half-life, the decay rate at the time of the experiment was determined using

$$N = N_0 e^{-\lambda t} \qquad R = \lambda N$$

At the time of our experiment, our source was decaying at a rate of 3.92×10^5 nuclei per second. Only 3.09% of the time is the decay mode fission, so the rate is 1.21×10^4 fissions per second. Since the neutrons are isotropic, only a portion of those will travel in the direction of the LGB detector. The number of neutrons traveling in the direction of the LGB detector can be found by using the following solid angle formula:

$$\Omega = 2\pi \int sin\theta d\theta \quad or \quad \Omega = 2\pi(1 - cos\theta)$$

This gives a rate of 12.24 neutrons per second. With a run of 17982 s we predict 2.2×10^5 neutrons to be incident on the LGB detector.

Chapter 3

Results

3.1 Data

3.1.1 Computational Results

Data were collected for 5 hours (17982 s) and in that time 5.4×10^5 events were captured and recorded. Not all of these events were valid, because some lacked pulses on at least one of the three channels. The valid events were processed using Matlab code (see Appendix A). The program looked for a valid TAC pulse, NaI pulse and LGB pulse. The peak of the TAC pulse is found and converted from amplitude to time. The time was then converted into energy using the following relativistic energy formula:

$$\left(\frac{1}{\sqrt{1 - \frac{d^2}{t^2 c^2}}} - 1 \right) mc^2$$

The calculated time of flight energy was then compared to the energy calculated from the height of the first LGB pulse.

It was expected, that there would be a correlation between the LGB pulse height and the time of flight, such that with only a few data points, a trend could be found. However, it was not until a large sample was used that the trend was apparent. This is an indication that the efficiency is low for energy detection with the LGB detector.

3.1.2 Efficiency

The decay rate of the californium suggests 2.2×10^5 neutrons should have hit the detector during the 5 hours (17982 s) of data collection. 5.4×10^5 events where collected, however, only 2.4×10^4 of the events were valid neutron events, which shows a 10.9% percent efficiency. This was expected for this type of neutron detector. However, when the energy graphs were compared with the expected energy charts for Californium, there was only a 0.2% correlation(Fig. 3.1). For systems or reactions with low neutron counts, this is not an effective method.

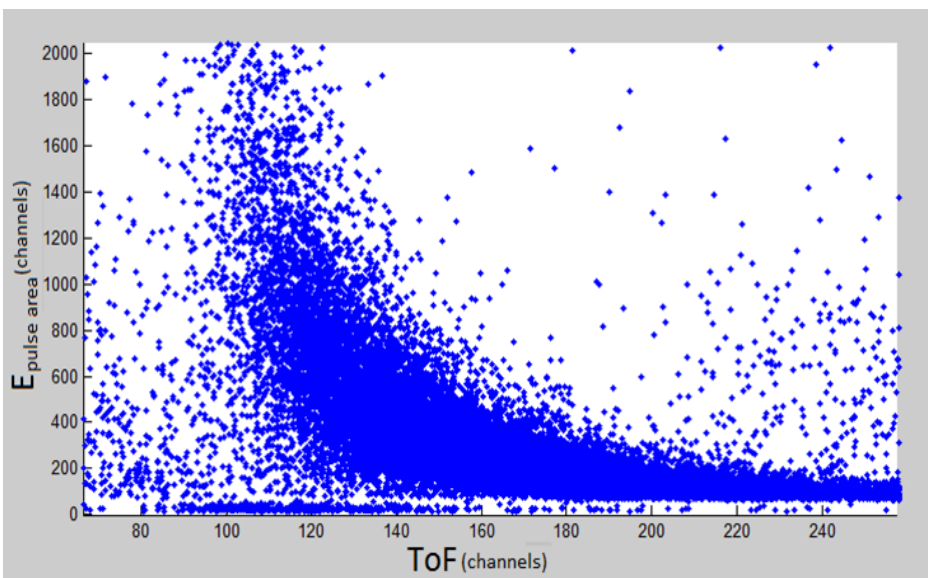

Figure 3.1 An estimated plot of the pulse area to the time of flight.

3.1.3 Conclusion

The characteristics of the LGB detector make it valuable for neutron detection. The detector has many practical applications because of the distinct double pulse signal with the signature neutron capture pulse. However, it is inadequate as a spectrometer for sparse neutrons. The expected correlation between detector pulse height or area and the energy indicated by the time of flight facility, was not seen. Instead, the correlation can only be seen using large amounts of data. Only 2% of the neutrons are correlated with the trend, and when combined with the efficiency of the detector, 10%, shows an efficiency of about 0.2% for sparse neutron spectroscopy.

One source of inefficiency is due to the plastic scintillation material in the LGB detector. Energy is lost and therefore not detected when the neutrons collide with multiple hydrogen atoms and carbon atoms. The hydrogen releases less light from a lower energy collision, and with multiple collisions, much of the light doesn't get released. The neutron can only transfer 28% of its energy the carbon. This makes the polyvinyl-toluene very inefficient (Wallace 2011).

In the future, the use of an inorganic hydrogenous scintillator material would create higher efficiency. An inorganic hydrogenous scintillator would eliminate energy loss from carbon collisions and maximize the neutron-proton collisions. Inefficiency also comes from the multiple neutron-proton collisions. Changing the scintillator design to minimize multiple collisions would maximize the light output and improve accuracy in neutron spectroscopy.

Bibliography

Czirr, J. B. 2013, Private communication, Brigham Young University Department of Physics and Astronomy

Dolzhenkova, E. F., Baumer, V. N., & Gordeev, S. I. 2003, Crystallography Reports, 48, 563

Ellsworth, J. 2013, Private communication, Brigham Young University Department of Physics and Astronomy

Rolfs, C., Trautvetter, H. P., & Rodney, W. S. 1987, Reports on Progress in Physics, 50, 233

Technology, E. 2010, Material Safety Data Sheet, Sweetwater, T.X. 79556

Wallace, A. 2011, Bachelors Thesis, Brigham Young Univeristy, Provo, U.T.

Appendix A

Full System Setup

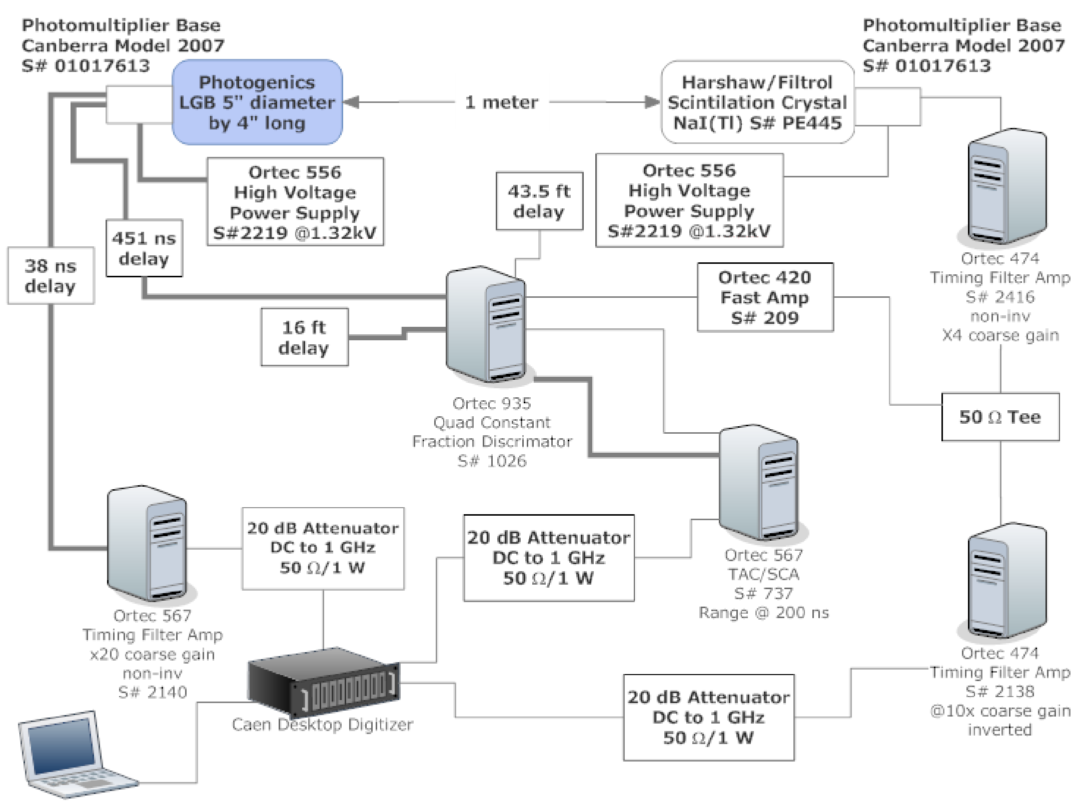

Figure A.1

Appendix B

Data Processing Code

B.1 Raw Peek

```
%  RawPeek.m   used for peaking at the data before processing.
%   channel number: 0 contains TAC waveform data
%                   1 contains the LGBPMT waveforms
%                   2 contains NaI PMT waveform
%   event recorded when LGB pusle exceeded 150 samples above 5 mv.
%   Events are 4096 samples long (16.384 ns)
%   jee@byu.edu 9-1-2011

clear; clc;

%**********************************************************************
% SETTINGS
%define a pulse by an amplitude that exceeds
startapmthreshold = 8;
%for a minimum duration of
durationthreshold = 8;
%minimum pulsewidth of
pulsewidththreshold = 15;
```

```
%and an ending amplitude that

%front and back porch lengths to retain
frontp =32;
backp = 16;
%*********************************************************************

%init
% wfmcnt = 0;   %wfmcnt and pulsenum used for data struc array dimensions
pulsecnt =0;
maxmultiple =0;
tracecnt = 0;
stopampthreshold = 4;
%CONSTANTS SPECIFIC TO THIS ACQUISITION
% filename = 'neutron_TOF_040811';
samplesPerEventPerChannel = 4096;
channelsEnabled = 2;
% eventsToRead = 420540;
maxEventsPerFile = 10000;

%get the waveform file names
wfm_files = dir('*).dat');
filecnt = size(wfm_files,1)
if filecnt == 0, break, end

for j = 1: filecnt
    wfilename =  [wfm_files(j).name]
    fid=fopen(wfilename,'r');
    wfmcnt = 0;   %wfmcnt and pulsenum used for data struc array dimensions
%      while isstr(wfmtext);  %end of file is -1
    for i = 1 : maxEventsPerFile
       %read waveform
        %We don't need the headers for this specific analysis, but it's
        %still useful for deadtime analysis
%missing        header = fread(fid,3,'ubit32');
        [TACwfm,samples] = fread(fid, samplesPerEventPerChannel,'uint16');
```

```
        [LGBwfm,samples] = fread(fid, samplesPerEventPerChannel,'uint16');
        [NaIwfm,samples] = fread(fid, samplesPerEventPerChannel,'uint16');
        TACwfm = TACwfm - 2047;
        LGBwfm = LGBwfm - 2047;
        NaIwfm = NaIwfm - 2047;

        subplot(3,1,1); plot(TACwfm)
        subplot(3,1,2); plot(LGBwfm)
        subplot(3,1,3); plot(NaIwfm)
        pause
    end
end%
```

B.2 Data Reduction

```
%  B_DataReduction.m   used for finding valid neutrons in data.
%                      using CAEN DT????
%   channel number: 0 contains TAC waveform data
%                   1 contains the LGBPMT waveforms
%                   2 contains NaI PMT waveform
%   An event was recorded when LGB pusle exceeded 150 samples above 5 mv.
%   Therefore the need to discriminate for start vs stop is not needed.
%   Events are 4096 samples long (16.384 ns)
%   jee@byu.edu 9-1-2011

clear; clc;

%************************************************************************
% SETTINGS
%channel for the lead edge of the LGB stop (capture) pulse
LGBStart = 3120;
%TAC processing delay in number of channels (time from starting edge of
% stop pulse to TAC output
```

```
TACdelay = 155;
%define a pulse by an amplitude that exceeds
startapmthreshold = 8;
%for a minimum duration of
durationthreshold = 8;
%minimum pulsewidth of
pulsewidththreshold = 15;
%and an ending amplitude that

%front and back porch lengths to retain
frontp =32;
backp = 16;
%**********************************************************************

%init
% wfmcnt = 0;   %wfmcnt and pulsenum used for data struc array dimensions
pulsecnt =0;
maxmultiple =0;
tracecnt = 0;
ncnt = 0;
stopampthreshold = 4;
%CONSTANTS SPECIFIC TO THIS ACQUISITION
% filename = 'neutron_TOF_040811';
samplesPerEventPerChannel = 4096;
channelsEnabled = 2;
% eventsToRead = 420540;
maxEventsPerFile = 10000;

%get the waveform file names
wfm_files = dir('*).dat');
filecnt = size(wfm_files,1)
if filecnt == 0, break, end

for j = 1: filecnt
    wfilename =  [wfm_files(j).name];
    fid=fopen(wfilename,'r');
```

```
    wfmcnt = 0;   %wfmcnt and pulsenum used for data struc array dimensions
    for i = 1 : maxEventsPerFile
        [TACwfm,samples] = fread(fid, samplesPerEventPerChannel,'uint16');
        if samples < samplesPerEventPerChannel; break; end
        [LGBwfm,samples] = fread(fid, samplesPerEventPerChannel,'uint16');
        [NaIwfm,samples] = fread(fid, samplesPerEventPerChannel,'uint16');

        TACwfm = TACwfm - mode(TACwfm);  %2047;
        LGBwfm = LGBwfm - 2047;
        NaIwfm = NaIwfm - 2047;
      %search for valid TAC pulse
%         tracecnt = tracecnt+1
        pulsenum =1;
        length = size(TACwfm,1);
        wfm = TACwfm;
        state = 'searchinit';  %init pulse finding state machine
        durationcnt = 0;
       %scan through pmt trace looking for a pulse,
        for s=1:length
            switch state;
                case 'searchinit'  %if trace starts mid pulse, discard it
                    if wfm(s) > stopampthreshold; %startapmthreshold    %if ...
                        above then start over
                        durationcnt = 1;
                        else                           %if below enough samples
                        durationcnt = durationcnt + 1;
                        if durationcnt > durationthreshold
                            state = 'search';   %start searching for pulse
                        end
                    end
                case 'search'  %search for min amplitude
                    if wfm(s) ≥ startapmthreshold
                        durationcnt = 1;
                        pstart = s;
                        state = 'minamplitude';
                    end
```

```

                case 'minamplitude' %search for min width
                    if wfm(s) < startapmthreshold;
                        state = 'search';
                    else
                        durationcnt = durationcnt+1;
                        if durationcnt > durationthreshold
                            state = 'findend';
                        end
                    end

                case 'findend'  %look for falling edge
                    if wfm(s) ≤ stopampthreshold  %%startapmthreshold
                        durationcnt = 1;
                        state = 'minend';
                    end

                case 'minend'  %ensure falling edge stays low
                    if wfm(s) ≥ stopampthreshold  %%startapmthreshold;
                        state = 'findend';
                    else
                        durationcnt = durationcnt+1;
                        if durationcnt > durationthreshold ...
                           && (s-pstart) > pulsewidththreshold
                                %if low long enough, record the pulse

                            %pulse found
                            pulsecnt = pulsecnt +1;
%     %                             pulsenum = pulsenum+1;
                            if pulsenum == 1;
                                wfmcnt = wfmcnt +1;
%                                     %save event identity
%     %                                 pdata(wfmcnt,pulsenum).dir = dirname;
% %                                     pdata(wfmcnt,pulsenum).file = wfilename;
% %                                     pdata(wfmcnt,pulsenum).trace = wfm(1);
%                                     %pdata(wfmcnt,pulsenum).data = fswfm;
```

```
%                                     pdata(wfmcnt,pulsenum).length = length;
%                                     %pulsenum = pulsenum+1;
                             end;
%                                 if maxmultiple < pulsenum
%                                     maxmultiple =  pulsenum;
%                                 end
%                             if pstart-frontp < 1;       %increase lead in samples
%                                 s1 = pstart-durationthreshold+1;
%                                 else
%                                 s1 = pstart - frontp;
%                             end
%                             if length-s > backp
%                                 s2 = s + backp;
%                                 else
%                                 s2 = length;
%                             end
%                                 %wfml = [wfm(s1:s); zeros(1023-s+s1,1)];
%                                 wfml = (wfm(s1:s2));
%                                 pdata(wfmcnt,pulsenum).start = s1;
%                                 pdata(wfmcnt,pulsenum).pstart = pstart;
%                                 pdata(wfmcnt,pulsenum).pend = s;
%                                 pdata(wfmcnt,pulsenum).data = wfml;
                                pulseend = s;
                                pulsestart = pstart;
                            pulsenum = pulsenum+1;
                            if (pulsenum ) > 2  %if more than one pulse, ignore it
                                break;  %s = length;
                            else
                               state = 'search';
                            end
                        end
                    end
                %end of case
            end
        end
        if pulsenum -1 ≠ 1  %if only one waveform in TAC conitnue on,
```

```
            continue  % if not found quit this waveform and go to next one
        end
        %TAC pulse found
        TACStart = pulsestart;
        TACPulse = wfm(pulsestart:pulseend);

       %recover neurton thermal given TAC and capture pulses locations and
       % only one pulse
%          tracecnt = tracecnt+1
        pulsenum =1;
        length = size(TACwfm,1);
        wfm = LGBwfm;
        state = 'searchinit';  %init pulse finding state machine
        durationcnt = 0;
       %scan through pmt trace looking for a pulse, in valid range
        rangestart = TACStart − 160;
        if rangestart < 1; continue; end  %if no pulse skip this one
        rangestop = 3130;
        for s=  rangestart:rangestop
            switch state;
                case 'searchinit'  %if trace starts mid pulse, discard it
                    if wfm(s) > stopampthreshold; %startapmthreshold    %if ...
                        above then start over
                        durationcnt = 1;
                    else                            %if below enough samples
                        durationcnt = durationcnt + 1;
                        if durationcnt > durationthreshold
                            state = 'search';   %start searching for pulse
                        end
                    end
                case 'search'  %search for min amplitude
                    if wfm(s) ≥ startapmthreshold
                        durationcnt = 1;
                        pstart = s;
                        state = 'minamplitude';
                    end
```

```
                case 'minamplitude' %search for min width
                    if wfm(s) < startapmthreshold;
                        state = 'search';
                   else
                        durationcnt = durationcnt+1;
                        if durationcnt > durationthreshold
                            state = 'findend';
                        end
                    end

                case 'findend'  %look for falling edge
                    if wfm(s) ≤ stopampthreshold  %%startapmthreshold
                        durationcnt = 1;
                        state = 'minend';
                    end

                case 'minend'  %ensure falling edge stays low
                    if wfm(s) ≥ stopampthreshold  %%startapmthreshold;
                        state = 'findend';
                    else
                        durationcnt = durationcnt+1;
                        if durationcnt > durationthreshold ...
                           && (s-pstart) > pulsewidththreshold
                                %if low long enough, record the pulse

                            %pulse found
                            pulsecnt = pulsecnt +1;
%     %                             pulsenum = pulsenum+1;
                            if pulsenum == 1;
                                wfmcnt = wfmcnt +1;
%                                     %save event identity
%     %                                   pdata(wfmcnt,pulsenum).dir = dirname;
% %                                       pdata(wfmcnt,pulsenum).file = wfilename;
% %                                       pdata(wfmcnt,pulsenum).trace = wfm(1);
%                                     %pdata(wfmcnt,pulsenum).data = fswfm;
```

```
%                                     pdata(wfmcnt,pulsenum).length = length;
%                                     %pulsenum = pulsenum+1;
                             end;
%                                 if maxmultiple < pulsenum
%                                     maxmultiple =  pulsenum;
%                                 end
%                             if pstart-frontp < 1;       %increase lead in samples
%                                 s1 = pstart-durationthreshold+1;
%                                 else
%                                 s1 = pstart - frontp;
%                             end
%                             if length-s > backp
%                                 s2 = s + backp;
%                                 else
%                                 s2 = length;
%                             end
%                                 %wfml = [wfm(s1:s); zeros(1023-s+s1,1)];
%                                 wfml = (wfm(s1:s2));
%                                 pdata(wfmcnt,pulsenum).start = s1;
%                                 pdata(wfmcnt,pulsenum).pstart = pstart;
%                                 pdata(wfmcnt,pulsenum).pend = s;
%                                 pdata(wfmcnt,pulsenum).data = wfml;
                                pulseend = s;
                                pulsestart = pstart;
                            pulsenum = pulsenum+1;
                            if (pulsenum ) > 2  %if more than one pulse, ignore it
                                break;  %s = length;
                            else
                               state = 'search';
                            end
                        end
                    end
                %end of case
            end
        end

```

```
        if pulsenum -1 ≠ 1  %if only one waveform in TAC,
            continue  % not found
        end
        j;
        ncnt = ncnt+1;

        ndata(ncnt).TACPulse = TACPulse;
                [height, place]=max(TACPulse);
        if TACPulse(place+50,1)>height-10 && TACPulse(place-200,1)>height-10
            ndata(ncnt).TAC = sum(TACPulse(place-200:place+50,1))/250;
        else
            if TACPulse(place+50,1)>height-10 && TACPulse(place-100,1)>height-10
            ndata(ncnt).TAC = sum(TACPulse(place-100:place+50,1))/150;
            else
                if TACPulse(place+25,1)>height-10 && ...
                    TACPulse(place-200,1)>height-10
                    ndata(ncnt).TAC = sum(TACPulse(place-200:place+25,1))/225;
                else
                    ndata(ncnt).TAC = sum(TACPulse(place-100:place+25,1))/125;
                end
            end
        end

        ndata(ncnt).LGBPulse = wfm(pulsestart:pulseend);
        ndata(ncnt).LGBArea = sum(wfm(pulsestart:pulseend));

        ndata(ncnt).Capture = LGBwfm(LGBStart:4095);

%          subplot(3,1,1); plot(TACwfm)
%          subplot(3,1,2); plot(LGBwfm)
%          subplot(3,1,3); plot(NaIwfm)
%          pause

    end
end
save('B_Neutrons2','ndata');
```

```
%
```

B.3 Plot Energy and Time of Flight

```
% D_EnToF.m used to find area and time of flight of LGB thermal pulses
%  Reads C*.mat file
%
% jee 8-22-2011

clear; clc

%load calibration curves
Co = load('CoCurves.mat');
Cs = load('CsCurves.mat');
%get the waveform file names
wfm_files = dir('B*.mat');
filecnt = size(wfm_files,1)
if filecnt == 0, break, end

ncnt = 0;



for j = 1: filecnt
    pdatafile = load(wfm_files(j).name);
    ndata = pdatafile.ndata;

    [pcnt, mcnt] = size(ndata);
%     ncnt = 0;
    for i = 1 : mcnt
        En(i) = ndata(i).LGBArea;
        Ev(i) = max(ndata(i).LGBPulse);
```

```
        ToF(i) = ndata(i).TAC;
        if size(ndata(i).LGBPulse,1) < 35
            SRatio(i) = 1;
        else
            SRatio(i) = sum(ndata(i).LGBPulse(1:35)) / En(i);
        end
        %SArea is already done and is En
        if size(ndata(i).Capture,1) < 35
            CRatio(i) = 1;
        else
            CArea(i) = sum(ndata(i).Capture);
            CRatio(i) = sum(ndata(i).Capture(1:35))/CArea(i);
        end
    end
end
% pdata = ndata;
% save('C_PreenedNeutrons','pdata');
subplot(2,2,1)
hist(Ev,256)
xlabel('Pulse Height')
ylabel('Count')
subplot(2,2,2)
hist(ToF,256)
xlabel('Time of Flight')
ylabel('Count')
subplot(2,2,4)
scatter(ToF,Ev, '.')
xlabel('Total Pulse Area')
ylabel('Pulse Area less than 140 ns')
subplot(2,2,3)
scatter(En,SRatio,'.r')
hold on
scatter(CArea,CRatio,'.b')
xlabel('Time of Flight')
ylabel('Area of Pulse')
hold off
```

```
%overlay cal curves
% Overlay pulse heights histogram
cdata = Co.cdata;
n = cdata.Evn;
xout = cdata.Evxout;
subplot(2,2,1)
hold on
plot(xout(1:255),n(1:255),'r')

cdata = Cs.cdata;
n = cdata.Evn;
xout = cdata.Evxout;
subplot(2,2,1)
hold on
plot(xout(1:255),n(1:255),'r')

% Overlay times on TAC  histogram
n = cdata.ToFn;
scale = max(ToF) / max(n);
n = n * scale;
xout = cdata.ToFxout;
subplot(2,2,2)
hold on
plot(xout,n,'r')

% Overlay energy vs time
n = cdata.PhVsTE;
scale = max(ToF) / max(n);
n = n * scale;
xout = cdata.PhVsTt;
subplot(2,2,4)
hold on
plot(xout,n,'r')
```

B.4 Convert Energy to Time of Flight

```
% This program loads the data from the TAC_Distribution file, converts
% it to energy through v=distance/time, gamma=1/squareroot(1-(v/c)^2),
% KE=(gamma-1)*m*c^2. The resluting energies are histogramed. This is
% the precourser to the TtoE Function.
%
% N. Quist 6-08-2012

clear;

format long;

Cf= xlsread('Source Data.xlsx');
off=51.44; %The offset value to reset the start time of the pulses
% to zero.

dist(1,1)=Cf(1,1)/2;
for n=2:length(Cf)
    dist(n,1)=(Cf(n,1)+Cf(n-1,1))/2;
end
c=3.00e8;% m/s
Mc2=939.56537821;%MeV/c^2 * c^2
tau=(129-9.44)/1185;% The time step determined by using a pulse
% traveling different known cable lengths and findng the difference.
Cs = load('CsCurves2.mat');
Time = load('TAC_Distribution.mat');

ToF = Time.TacDist;
% ToF = ToF-off;
i=1;

for n=1:length(ToF)
    if ToF(n)>0
```

```
            vel(i,1)=1./(((ToF(n).*tau)+5).*(10^-9));
            if vel(i,1)>3e8
                i=i-1;
            else
                i=i+1;
            end
        end
end

gamma=real(1./sqrt(1-(vel./c).^2));
E=((gamma-1).*Mc2);
i=1;

 for n=1:length(E)
     if E(n)<11
         E2(i)=E(n);
         i=i+1;
     end
 end
 hist(E2, 256, dist)
 xlim([0 2])

% [counts, Energy]=hist(E2,128,dist);
% [A, B]=max(counts);
% sprintf('Mean - %g, Offset - %g, Peak - %g',mean(E),off,Energy(B))
% bar(Energy, counts/length(E2))
% xlim([0 2])
% title('Energy vs. Time Histogram')
% xlabel('Energy')
% ylabel('Count')
```