# AN ABSTRACT OF THE THESIS OF

Kacey D. McGee for the degree of Master of Science in Radiation Health Physics presented on August 14, 2019.

Title: Silicon Photomultiplier Modeling of CsI(Tl) with Front End Electronics using a Monte Carlo Model

Abstract approved:

Abi T. Farsoni

Silicon photomultipliers (SiPM) have become increasingly popular in radiation detection design due to smaller profiles, lower operating voltage, and magnetic insensitivity. However, there are nonlinear effects that make modeling SiPM challenging. This study looks at using a Monte Carlo approach to modeling the output of an SiPM and front end electronics for a slow scintillation pulse (CsI(Tl)) and then verifying the results with measurements. The SiPM circuit and front-end electronics were designed in LT Spice for the simulation. In order to define the current source within the SiPM model, a python script was written using a Monte Carlo approach simulating microcells firing throughout the scintillation process. Measurements were taken from a physical circuit and compared. It was found that the rising edge agreed well with measurements. However, the falling edge showed an over-estimate on the SiPM output and an underestimate on the amplifier output. The SiPM output amplitude was likely due to a difference in the simulated light yield and actual light yield from the scintillation crystal. The difference in the falling edge was

likely due to errors in the modeling of the CsI(Tl) scintillation decay. Overall this model produced a good estimate for the output from front end electronics.

# Silicon Photomultiplier Modeling of CsI(Tl) with Front End Electronics using a Monte Carlo Model

by

Kacey D. McGee

A THESIS

submitted to

Oregon State University

in partial fulfillment of
the requirements for the
degree of

Master of Science

Presented August 14, 2019
Commencement June 2020

Master of Science thesis of Kacey D. McGee presented on August 14, 2019

APPROVED:

Major Professor, representing Radiation Health Physics

Head of the School of Nuclear Science and Engineering

Dean of the Graduate School

I understand that my thesis will become part of the permanent collection of Oregon State University libraries. My signature below authorizes release of my thesis to any reader upon request.

Kacey D. McGee, Author

# ACKNOWLEDGEMENTS

I would like to thank the professors of Oregon State University's School of Nuclear Science and Engineering for the guidance and knowledge they have provided during my time here.

I would like to thank Dr. Abi Farsoni, Steven Czyz, Harish Gadey, and Mitch Mannino for their patience, time, and assistance throughout this work.

Finally, I would like to thank my wife, Sarah McGee, and my four children for their love and understanding as I worked through this program.

# TABLE OF CONTENTS

Page

TABLE OF CONTENTS (Continued)

# LIST OF FIGURES

# LIST OF FIGURES (Contiued)

# LIST OF TABLES

# LIST OF APPENDICES

# Silicon Photomultiplier Modeling of CsI(Tl) with Front End Electronics using a Monte Carlo Model

## 1. INTRODUCTION

### 1.1 Motivation

Silicon Photomultipliers (SiPM) continue to become a more popular choice in radiation detection design. This is due to the advantages SiPM have over traditional photocathode/ photomultiplier tube setups. These advantages include lower operating voltage, smaller profile, and magnetic field insensitivity [1]. The drawbacks of SiPM include nonlinearity effects such as cross talk, after pulsing, dark counts, and dynamic range [2]. These nonlinearity effects can alter the expected signal response and should be taken into account when designing the front-end electronics for a radiation detector. One method of doing this would be using a SPICE modeling program, such as LT Spice or OrCAD.

When modeling the SiPM output from a scintillation pulse, there are two portions to be modeled. The first is the electrical circuit of the SiPM. This portion models the physical characteristics of the SiPM circuit. A couple of generic SiPM models are shown in Figure 1. The second portion involves modeling the source signal produced from the scintillation pulse. The source signal drives the output signal of the SiPM and can be modeled with either as a current source or a voltage source.

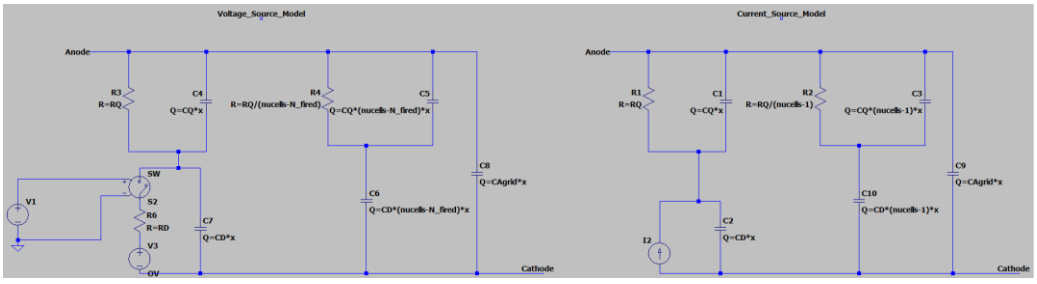


**Figure 1. Generic Electrical Models of an SiPM. The voltage model on the left is driven by a voltage switch that opens when a microcell is firing, allowing the overvoltage to flow through the circuit. The current model on the right is driven by the current source only** *[1] [2]*.

The SiPM consists of many photon-activated microcells connected in parallel. As a microcell is activated, a current is produced proportional to the voltage the microcell is biased at. This leads to the two different SiPM models shown in Figure 1. Variations of these models have been well established in recent years [1] [2] [3] [4].

There are multiple methods available for creating the source signal. Among them, the two most popular include a pulse dynamic range method and a Monte Carlo method. The pulse dynamic range method is an equation that converts incident photons to number of microcells as an estimation for dynamic range effects. This method is used when the light pulse is short in comparison to the recovery time of the SiPM. The Monte Carlo method uses the probabilities of the nonlinear effects (after pulsing, cross talk, etc.) to generate a probable driving source. The monte carlo method has been used in the past to obtain accurate output for fast scintillators, such as LYSO or $LaBr_3$, and then extended to predict theoretical output response for slower scintillators, such as CsI(Tl) [1] [5] [6].

SiPM signal output is nonlinear and output will vary depending on the scintillation crystal being used. Being able to accurately predict SiPM output during the design phase can save money and time in selecting the front-end electronics. This work aims to generate a monte carlo model of the SiPM response to CsI(Tl), similar to those done for fast scintillators LYSO and $LaBr_3$, and compare simulated results to measured results using a circuit built at OSU.

## 1.2 Research Objectives

This research uses an electrical model of the MicroJ 60035 SiPM provided by the manufacturer. This work attempts to use this model and do the following:

1. Implement the SiPM model provided by SensL in LT Spice and validate with the provided testbench circuit/ output signals.
2. Alter the SensL model from being a voltage source driven model to a current source driven model.
3. Implement a python script to generate a current source signal of CsI(Tl) exposed to Cs-137 for the monte carlo model.
4. Compare output from the Monte Carlo model to a measured signal output.

## 1.3 SensL NDA

In an attempt to obtain the most accurate results, this research uses an SiPM electrical model provided by SensL of their Micro J-series 60035 SiPM under a Non-Disclosure Agreement (NDA). The model provided by SensL was a specialized version of the generic voltage model shown in Figure 1. Component values for the model were also provided along with a testbench circuit and signal output.

While effort was placed in maintaining the model as provided, small alterations were necessary (such as changing the model from a voltage driven model to a current driven model). Whenever an alteration was made to the SiPM circuit, accuracy was verified using the testbench circuit and output provided with the model. While the SensL circuit can not be shown in this work, a generic model of an SiPM without component values will be shown whenever visualization is necessary.

# 2. LITERATURE REVIEW

## 2.1 Scintillators

### 2.1.1 Introduction

Scintillation materials are used to convert absorbed radiation into light photons. These photons, when coupled with a photocathode and PMT or a Silicon Photomultiplier generate a current signal to the rest of the circuit.

Scintillation occurs as a process when radiation interacts with material capable of scintillating. First, energy from incident radiation is transferred to the scintillation material. This energy is passed to electrons in the scintillation material. With the increase in energy, the electrons are raised to an excited energy state. As excited electrons transition back to their ground state, the excess energy is released as a photon [7]. Due to the quantum orbital states of electrons, this wavelength should be one of a finite number of possible transitions as shown in Figure 2.

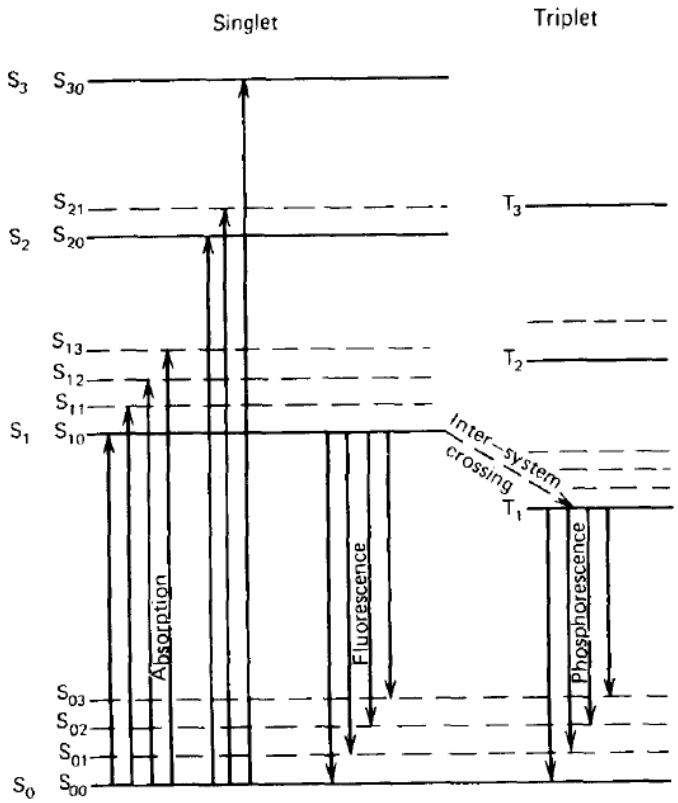


**Figure 2. Electron states of a scintillator. From left to right, the arrows show absorption, fluorescence, and phosphorescence** *[7]*.

Another consequence of Figure 2 is the formation of a scintillation spectrum. The spectrum contains all energies between the maximum possible excited state, and the first excited state, approximately. Figure 2 would indicate a finite number of single wavelengths, in a perfect detection system. The reason the spectrum is continuous deals with the material being in a crystalline structure. The electron interactions between atoms cause each of the energy states to broaden. As a whole, the lattice forms energy bands around the original energy states. This, along with detector noise and thermal vibrations, causes the scintillation output to look like a full spectrum as in Figure 3. This leads to the term valence band instead of ground state, conduction band instead of excited state, and band gap instead of energy gap when referring to a congregate of material instead of a single molecule.

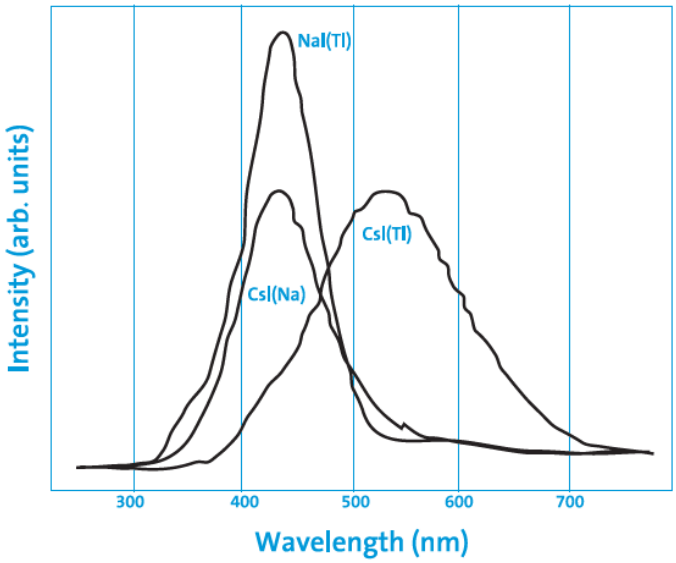


**Figure 3. Scintillation Spectrum of common scintillators** *[8]*.

Looking back at Figure 2, the figure indicates singlet and triple states. These states depend on the electron spin configuration. Singlet states (spin 0) are indicated by an S prefix, and triplet states (spin 1) are indicated by a T prefix. Singlet states are at a lower energy and so the ground state may only be in the singlet state. Figure 4 provides more details on triplet vs. singlet states, with each arrow indicating an electron with spin up or spin down characteristic [7].

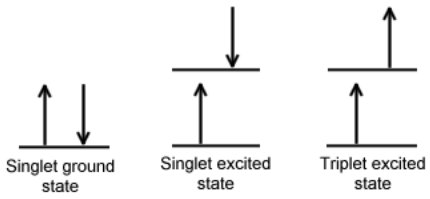


**Figure 4. Singlet and Triplet states** *[9]*.

### 2.1.2 Inorganic Scintillators

There are two main categories of scintillators based on the material being used. They are classified as inorganic and organic scintillators due to slightly different characteristics. Alkali halide crystals such as NaI, CsI, and LYSO are inorganic scintillators. Organic Scintillators include stilbene, anthracene, and plastics [7].

While the introduction was generalized to cover the basics of all scintillation materials. Inorganics are slightly different due to the generally larger band gap between the valence band and the conduction band. The energy of photon produced is proportional to the band gap and for many inorganic scintillators will cause the energy to fall outside the spectrum detectable by a photocathode or SiPM. This leads to the use of activators to produce energy states within the band gap, producing a lower energy photon with a larger wavelength. This is shown in Figure 5. The activators also act to greatly reduce any self-absorption present in the original crystal.

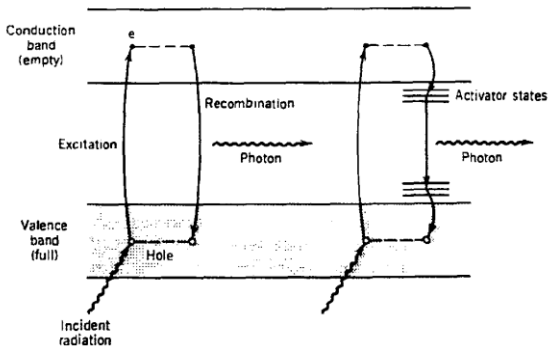


**Figure 5. Inorganic Scintillator model. (Left) inorganic scintillator without added activator. (Right) scintillator with activator** *[10]*.

Scintillation with an activated (or doped) material is very similar to the scintillation described earlier. The electron is first excited to the conduction band. It quickly moves from the conduction band to an energy state generated by the activator, as this is at a lower energy state. The electron then decays to the ground state of the activator, followed by transition to the valence band.

For inorganic scintillators, scintillation can occur as either fluorescence or phosphorescence. Fluorescence occurs when the excited electron in a singlet state transitions to the ground state in the singlet form. This transition occurs quickly and is typical scintillation behavior. Phosphorescence occurs when the originally excited electron in a singlet state moves to a lower energy state of opposite spin, placing the system into a triplet state. This triplet state then slowly transitions to the ground singlet state. This occurs with a slower rate than Fluorescence. A triplet state to singlet state is forbidden during transition, but may occur between excited states where vibrational states overlap [7] [11].

These transition types each have characteristic transition times, (sometimes multiple transition times within one type). Together, the transition times generate an estimated decay constant ($\tau$) for the scintillation material. $\tau$ represents the time it takes for $[1 - e^{-1}]$ (~63%) of the photons to transition from the conduction band to the valence band. There may also be a $\tau$ associated with the promotion of the electrons into the conduction band. When this is the case, the light pulse may be modeled as Equation (1) [7].

$$I(t) = I_0 \left( e^{t/\tau_{fall}} - e^{t/\tau_{rise}} \right) \quad (1)$$

### 2.1.3 CsI(Tl) Properties

CsI(Tl) is an inorganic scintillation crystal. It is comprised of a CsI matrix doped with Thallium activator. It is slightly hygroscopic and must be sealed from the environment to prevent degradation to the crystal over time. Some advantages of CsI(Tl) are the exceptionally high light yield, high efficiency for absorbing gammas (due to the large Z material), and ability to differentiate different types of radiation through variable decay time [7] [12].

Light yield of CsI(Tl) has been reported with variability in the past. In 1988, a study completed by Holl et. al. reported a light yield of ~51,500 photons/MeV [13]. In 1993, a study completed by Valentine et. al. reported a light yield of ~65,000 photons/MeV, indicating an increase of ~25% [12] [7]. The crystal manufacturer Saint-Gobain has also listed 54,000 photons/MeV for its product [14]. This work uses the results reported by Valentine as the latest scientifically published results. However, an argument could be made for using the manufacturer's data in future work.

Just as the light yield has variability in reporting, the scintillation constants reported also have some variability. In 1990, a study completed by Schotanus et. al. reported $\tau_{rise}$ as ~30 ns, and two decay constants $\tau_{fall1}$ ~ 600 ns (54%) and $\tau_{fall2}$ ~ 3400 ns (46%) [15]. In 1993, Scintillation constants were also been reported as $\tau_{rise}$ = 20ns, $\tau_{fall1}$ = 680ns (64%), $\tau_{fall2}$ = 3340 ns (36%) by Valentine et. al. [16]. The manufacturer Saint-Gobain has also listed a $\tau_{fall}$ of 1000 ns with no rise time listed [14]. In 2007, a study completed by Syntfeld-Kazuch et. al. reported three decays constants of $\tau_{fall1}$ ~ 730ns (44%), $\tau_{fall2}$ ~ 3.2 μs (31%), and $\tau_{fall3}$ ~ 16μs (25%). Decay constants $\tau_{fall1}$ and $\tau_{fall3}$ were reported to vary in probability with incident radiation energy [17]. Figure 6 shows the differences between the 4 reported models of the CsI(Tl) scintillation for each set of constants listed, using Equation (1).

While the models proposed by Valentine and Schotanus follow very closely throughout, the Saint-Gobain and Syntfeld-Kazuch models tend to the extremes. The Saint-Gobain model shows the scintillation pulse occurring very quickly and the Syntfeld-Kazuch shows the scintillation pulse lasting much longer. This research uses

the results from Valentine's work as they tend to match the middle of the models and match closely with the Schotanus model.

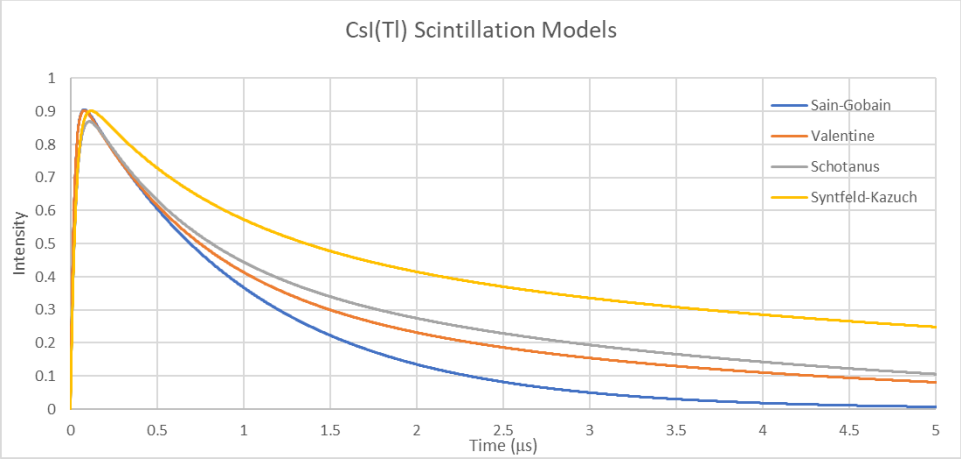


**Figure 6. CsI(Tl) Scintillation Models using Equation (1), $I_0 = 1$, for various reported scintillation constants. It is assumed the rising constant as 20ns for those models where it was not reported.**

The scintillation spectrum for CsI(Tl) and CsI(Na) are provided below from the Saint-Gobain information sheet [14]. The CsI(Tl) scintillation spectrum begins at ~700nm, rises to a peak at ~550nm, and begins to decay until ~350 nm. The scintillation spectrum must match well with the absorption spectrum of the photocathode or SiPM in order to generate a large signal. For comparison, the absorption spectrum for the SensL MicroJ Series SiPM peak at ~420 nm with up to 50% photon detection, but only up to about ~27% at 550 nm. CsI(Tl) peaks at a longer wavelength compared to the absorption peak for most photon detecting devices, and may lead to a smaller signal output than from other scintillators with shorter wavelength peaks [7].

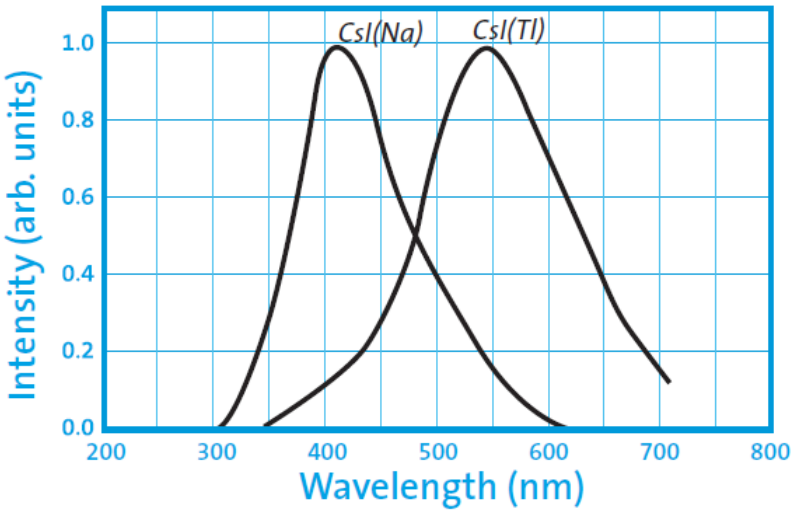


**Figure 7. CsI(Tl) and CsI(Na) scintillation spectrums** *[14]*.

It has been reported that CsI(Tl) has nonlinear properties for temperature effects as well as energy absorption. Temperature has been reported to effect the light intensity. As temperature decreases, light yield increases to a maximum at ~-35 $^{o}$C. Room temperature light yield was estimated at ~65,000 photons / MeV [12]. For energy absorption, it has been reported that the decay constants have a dependence on the energy of the incident radiation [17]. It has also been reported that light yield is dependent on the energy as shown in Figure 8. Lower energy radiation produces more photons per MeV than higher energy radiation.

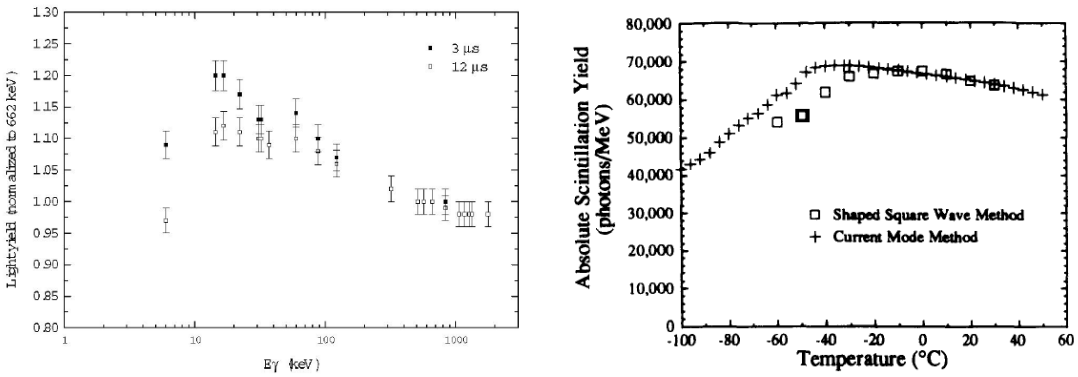


**Figure 8 (Left) Reported Energy Dependence of CsI(Tl) at two shaping times, normalized to 662keV. (Right) Reported Temperature Dependence** *[18] [16]*.

## 2.2 Photon Detectors

### 2.2.1 Introduction

Photon detectors, such as a photocathode combined with a photomultiplier tube (PMT) or silicon photomultipliers (SiPM), convert scintillation photons into current pulses.

For photocathodes combine with a PMT, the scintillation photons ionize the photocathode. These electrons forward scattered through the photocathode to the interface where the photocathode and photomultiplier tube meet. The photomultiplier is held at vacuum and so the electrons must overcome the work function energy to enter the PMT. Once in the PMT, the electron is multiplied via dynode stages at very high voltage. This signal is then sent out to the anode of the PMT as a current pulse.

A picture of the Hamatsu PMT H3178-51 is shown in Figure 9. For reference this PMT has a 4.7 cm diameter and 16.2 cm long profile with an operating voltage of 1.5 kV.

**Figure 9 Hamatsu PMT H3178-51** *[19]*.

The other category of devices used to convert scintillation photons into current pulses are SiPM. These devices are composed of many single photon avalanche diodes (SPAD). Each SPAD, or microcell when incorporated into an SiPM device, operates as a Geiger detector. As a photon hits a microcell there is an avalanche of current that occurs. As a larger number of microcells are triggered, a

larger signal is produced. However, there are limits. SiPM only have a limited number of microcells and if two photons are absorbed in the same microcell, the same avalanche current signal is produced, leading to dynamic range effects (saturation). Other nonlinear effects are also present, but will be discussed later.

A picture of the SensL MicroJ Series 60035 SiPM mounted to a pin adapter board is pictured in Figure 10. For reference, this SiPM without the board has a ~ (6mm x 6mm x 0.37mm) profile and operates at less than 32V. However, there are the nonlinear effects that must be accounted for in designing with the SiPM.

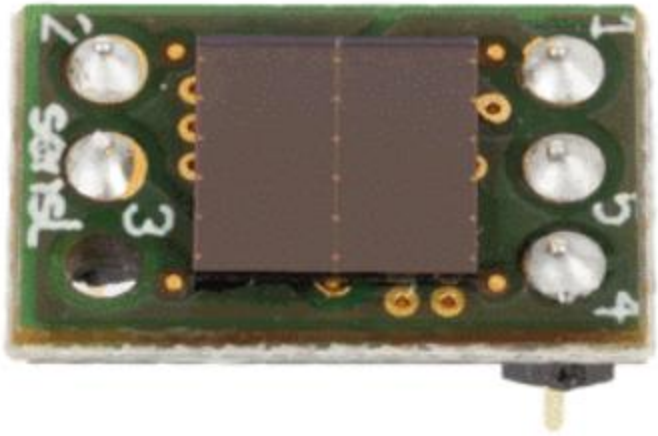


**Figure 10 SensL MicroJ Series 60035 SiPM mounted to a Pin board** *[20]*.

### 2.2.2 Single Photon Avalanche Diodes (SPAD)

Silicon Photomultiplier (SiPM) circuits can be described as an array of Single Photon Avalanche Diodes (SPAD). In modeling the SiPM, it is important to be able to accurately model the SPAD.

Single Photon Avalanche Diodes (SPAD) behave as a P-N junction diode that operates at a reverse bias above the breakdown voltage [4]. While operating above the breakdown voltage, the SPAD acts in a similar fashion to a Geiger detector. The characteristic I-V curve is shown in Figure 11. There is a large voltage across the SPAD and a single photon can generate an avalanche signal. This is shown as the avalanche phase on the I-V curve with a sudden increase in current. As the avalanche phase continues, the voltage across the SPAD drops from the biased voltage towards the breakdown voltage until the avalanche can no longer support itself and stops. This is shown as the quenching phase on the I-V curve with the current and voltage dropping towards the reset point. The voltage then slowly resets to the bias voltage [2]. The leakage current indicates the small current always present for the SPAD when reverse biased. This leakage current resets the SPAD after an avalanche.

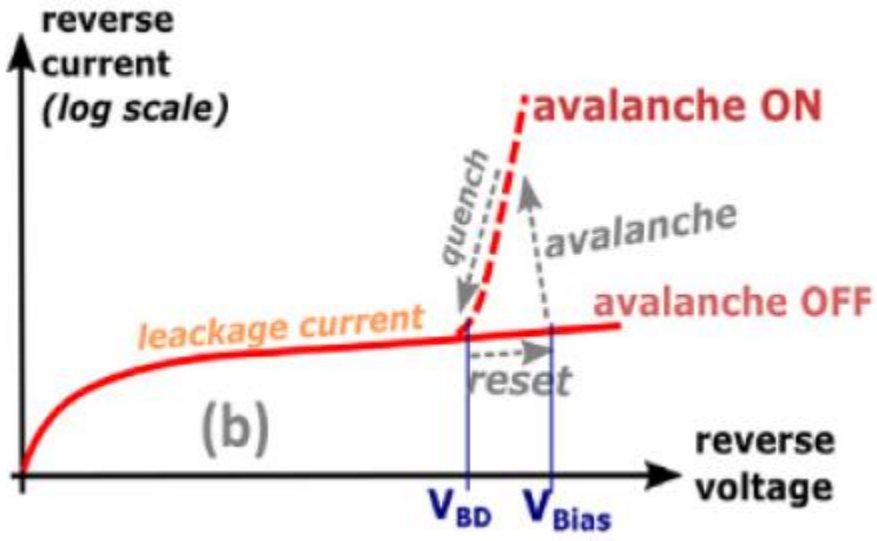


**Figure 11. SPAD characteristic I-V curve. Current flowing through the SPAD is on the Y axis, while Voltage across the SPAD is on the X axis** *[2]*.

The SPAD electrical circuit is well modeled with Figure 12. The model includes an integrated quenching resistor. It is driven by the overvoltage (V_Bias – V_breakdown) controlled by a switch, to model the SPAD firing. While the switch is open, an avalanche current flows through the integrated quench resistor (R_q) and the SPAD's internal discharge resistor (R_d). There is a small parasitic capacitance that is modeled across the quench resistor (C_q). The internal capacitance of the SPAD is modeled with the discharge capacitance (C_d) [4] [21]. R_sense has been added to obtain a voltage signal from the model.

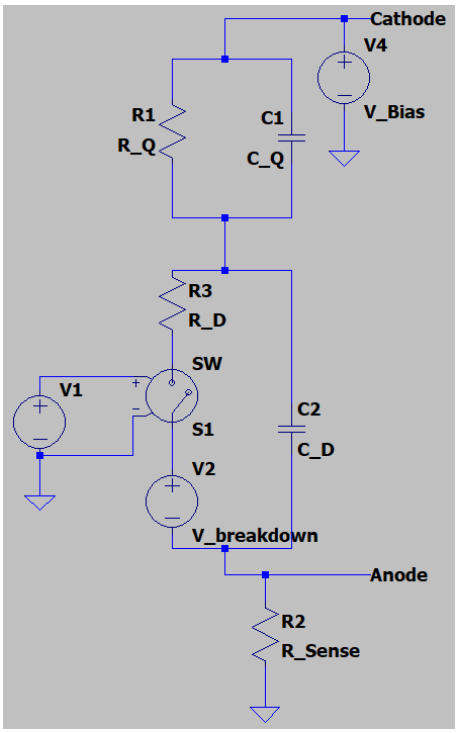


**Figure 12. SPAD Electrical Model implemented in LT Spice VII. Voltage Source Driven.**

With the model, the following observations can be made. The output will have a fast component (avalanche phase) and a slow component (reset phase). The total amount of charge dissipated during avalanche and reset is given by Equation (2). This is a function of the overvoltage and the capacitance within the circuit. This means the gain of the circuit can be controlled by selecting the bias voltage. This can be broken down into the fast avalanche component given by Equation (3) and the slow reset component given by Equation (4) [21].

$$Q_{total} = V\_overvoltage * (C\_d + C\_q\ ) \quad (2)$$

$$Q_{avalanche} = V\_overvoltage * C\_q \quad (3)$$

$$Q_{reset} = V\_overvoltage * C\_q \quad (4)$$

The timing constants can also be determined. Equation (5) gives the avalanche time constant, and Equation (6) gives the reset time constant. The peak current of the avalanche can be determined with Equation (7). Combining Equation (5) with Equation (7), a model of the avalanche can be determined. This model is given with Equation (8).

$$\tau_{av} = R\_d * (C\_d + C\_q) \quad (5)$$

$$\tau_{reset} = R\_q * (C\_d + C\_q) \quad (6)$$

$$I_{peak} = \frac{V_{ov}}{R\_d} = \frac{Q_{tot}}{\tau_{av}} \quad (7)$$

$$I(t) = I_{peak} * \ e^{-t/\tau_{av}} \quad (8)$$

While Equation (8) provides a good estimation for the avalanche current pulse, it does not take into account the abrupt cutoff when the avalanche is no longer able to sustain itself. This cutoff has been reported at ~ 20μA, but has also been characterized as a current less than 100μA in other studies [2] [4] [21]. In order to model the cutoff, the voltage switch is generally left on for < 5 $\tau_{av}$.

With the avalanche pulse accurately defined, the SPAD model can be altered to use the avalanche current pulse to generate the same output. The altered SPAD model is shown in Figure 13. The breakdown voltage, voltage switch, and discharge resistor have all been removed from the circuit. The current flows towards the anode, just as in the voltage driven model. While it was mentioned earlier that the output of the SPAD has fast and slow components, the current source only models the fast avalanche component. The slow component occurs after the avalanche phase and occurs as the voltage is reestablished to the bias voltage.

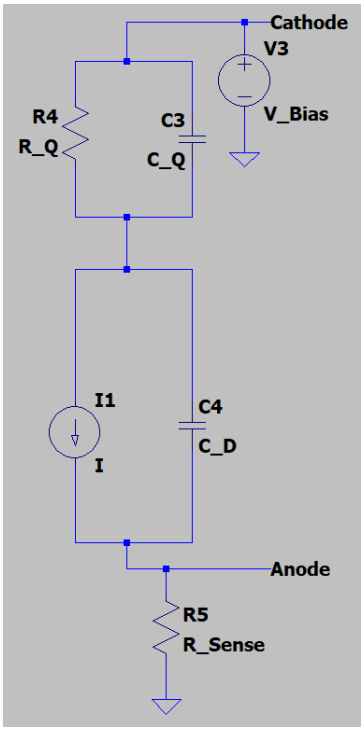


**Figure 13. SPAD Electrical Model implemented in LT Spice VII. Current Source driven.**

Using both models with estimated component values, the outputs can be seen in Figure 14. This is the modeled current across $R_{\_sense}$ at the anode. There is a very fast rise in current as the voltage within the SPAD drops from the bias voltage to the breakdown voltage due to an avalanche. As the difference in voltage becomes smaller, there comes a point where the avalanche is no longer sustained and stops. This is followed by a recovery period where a much smaller current flow returns the

SPAD voltage back to the bias voltage. This leads to a fast component (avalanche phase) and slow component that occurs after the avalanche (recovery phase) [21].

Comparing the two models, the current model (green) used Equation (8) to estimate the avalanche current without a cutoff. The voltage model (blue) cut off after ~$2\tau_{av}$. The models agree closely until the cutoff point (reset phase begins). If the SPAD were to be initiated with a photon prior to a full reset, the avalanche signal would be smaller based on the overvoltage being less.

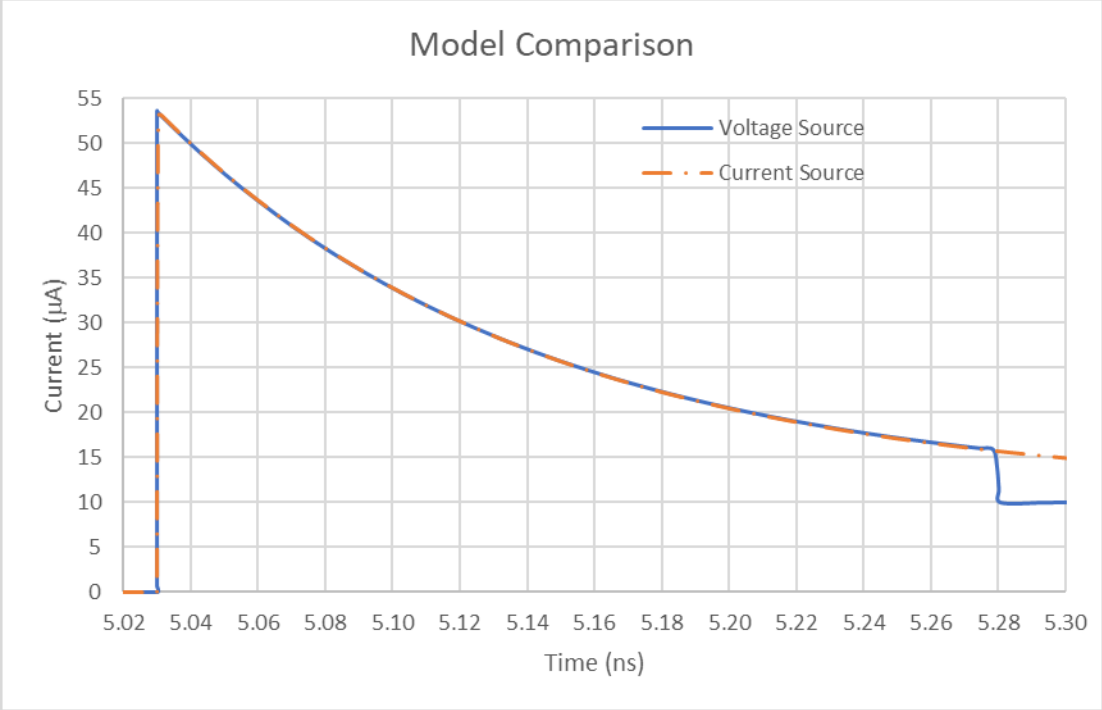


**Figure 14. Comparison of SPAD anode output from the voltage driven model (blue) and the current driven model (green).**

The advantage of using the voltage driven model is the ability to model an avalanche event with or without quenching [2]. However, when using these models to simulate an SiPM circuit, the current model will have the advantage of superimposing the current of multiple SPAD firing simultaneously.

### 2.2.3 Silicon Photomultipliers (SiPM)

#### Introduction

Silicon Photomultipliers (SiPM) are comprised of an array of SPAD connected in parallel. Each SPAD, or microcell, acts independently (with the exception of crosstalk events). Where a single SPAD could only detect presence / absence of a photon event, a SiPM can also estimate how many photons are present.

Silicon Photomultipliers (SiPM) continue to become a more popular choice over traditional Photomultiplier Tubes (PMT) in radiation detection design. This is due in part to lower operating voltages, a smaller profile, and magnetic field insensitivity. However, SiPM are not without drawbacks.

#### Inherent Non-Linear Effects

The most important drawbacks being their non-linearity due to inherent properties such as dynamic range, after-pulsing, dark counts, and crosstalk. Of the nonlinear effects, after-pulsing, dark counts, and crosstalk can be altered by the choice of the biasing voltage. As the overvoltage increases, so does the probability of these nonlinear effects.

Dynamic Range effects occur due to the limited number of microcells available on an SiPM. As more photons are introduced there is a greater probability that two photons will hit the same microcell, or that a microcell will be triggered prior to a full recovery, leading to a smaller than normal signal and nonlinear output [21]. This effect is most prominent when the scintillation pulse is short in comparison to the SiPM recharge constant given in Equation (6). For long scintillation pulses, such as those seen with a CsI(Tl), this effect should be less pronounced [5].

After-pulsing occurs shortly after an initial pulse. This pulse is not a photon induced pulse, and adds to the nonlinearity of the system. This is thought to be caused by a trapped carrier. When the carrier is released during the recovery phase, another avalanche is created [1]. Occurrence is usually expressed as a percentage of fired microcells. After-pulsing produces a larger than expected signal.

Optical Crosstalk occurs when avalanche carriers generate photons that trigger neighboring microcells. Occurrence is usually expressed as a percentage of fired microcells. This also produces a larger than expected signal.

Dark Counts are considered one of the main sources of noise in a SiPM. These are thermally generated electrons that cause an avalanche. [1] These are not caused by light and thus called dark counts. These are usually calculated as an average frequency and listed in Hz. This will have minor effects on short pulses, but will impact longer pulses, such as from CsI(Tl).

Nonlinear effects are taken into account with the driving source modeling. The driving source is generally modeled in one of two ways. These are the pulse dynamic range method and the Monte Carlo method.

The pulse dynamic range method is shown in Equation (9). The number of fired microcells is estimated based on the total number of microcell (M), the photon detection efficiency (PDE), and number of incident photons ($N_{ph}$). This method assumes no optical crosstalk, no after pulsing, and a large amount of saturation. This method gives a good estimate for very short light pulses when compared to the recharge constant of the SiPM, such as those from laser pulses, or those from fast scintillators such as LYSO.

$$N_{fired} = M * \left[1 - \exp\left(-PDE * \frac{N_{ph}}{M}\right)\right] \quad (9)$$

The Monte Carlo method uses the probabilities of each nonlinear effect, and applies them to the incident photons. This method is a more accurate estimate for scintillators with longer scintillation pulses, such as CsI(Tl).

## Temperature Non-Linear Effects

In addition to inherent non-linear effects, there is a temperature dependent non-linear effect that occurs as well. The breakdown voltage is shifted by changes in temperature. It is increased by a rate of 21.5 mV/$^{o}$C. If the bias voltage is maintained

which the temperature is increased, there is a smaller over voltage value, leading to a smaller SiPM output as temperature is increased [22].

It is also seen that the quench resistor is affected by changes in temperature, being reduced as temperature is increased [22]. Because the avalanche is shaped by $\tau_{av}$, as temperature is increased, the decay constant will become smaller. The avalanche will take a longer period of time to complete.

While temperature decreases the overvoltage, and this would produce an expected decrease in dark counts, the opposite is reported [22]. When held at a constant overvoltage, a larger number of dark counts can be determined. It was also reported that cross talk is affected very slightly by temperature changes.

## Electrical Circuit Models

Due to the nature of the SiPM, the electrical circuit model is very similar to the SPAD. Figure 15 shows the generic voltage source model for the SiPM [2]. Figure 16 shows the generic current source model for the SiPM.

Looking at Figure 15 and Figure 16, the model can be broken up into three regions. The first region models the physical circuit of a single microcell. This is the same as the SPAD model for the current driven model, but slightly different in the voltage driven model. For the voltage driven model, the components are modified by the number of microcells firing. This is because the modeled voltage does not change as more microcells fire, and so the component values must change. For quick pulses, the number fired may be estimated to be constant. In the current driven model, the current source takes into account the changes in the number of fired cells, and so the circuit components may have static values.

The second region models the load effects from the rest of the microcells. The SPAD model was not connected to any other SPAD and so would not have this region. For a scintillation pulse, the components in the voltage driven model will be dynamic for this region as well. They will remain static for the current driven model.

The final region models the parasitic portion the grid has on the signal. These values are static for both models.

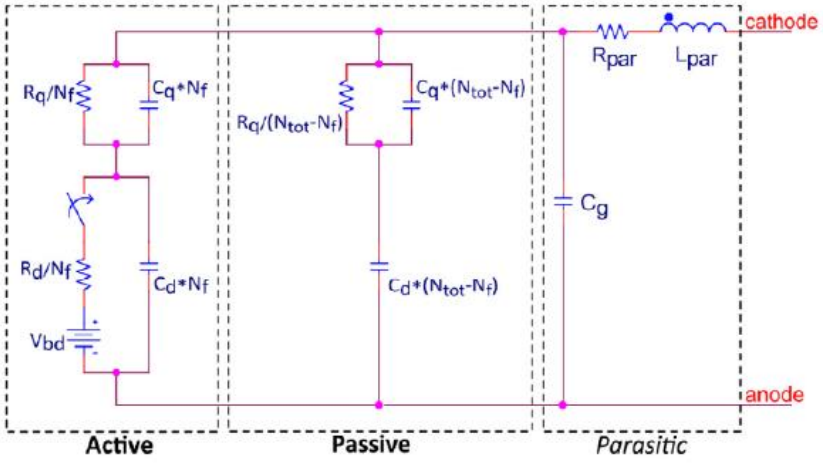


**Figure 15. Voltage Source Driven SiPM Model** *[2]*.

This model is best used for short duration pulses with a constant number of photons. For scintillation pulses, the number of fired cells would need to be adjusted throughout the duration of the scintillation process. This in turn would cause the capacitor / resistor values to change throughout the simulation. This could lead to unintended effects, such as a capacitor changing to a value less than the charge it is currently holding.

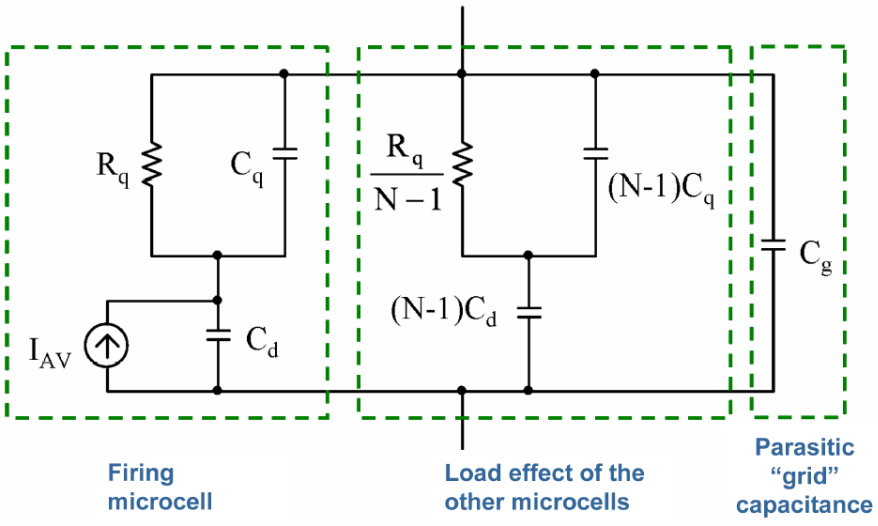


**Figure 16. Current Source SiPM Model** *[23]*.

The current model contains the same sections as the voltage source model. However, this model does not require any changes to component values. Instead, the current source should include all the SPAD avalanche signals placed with correct timing.

The current source model is better suited for scintillation as no component values change, despite the number of fired cells changing. The challenge is ensuring the current source is well defined to fit scintillation pulses and handle non-linear effects. When generating the piecewise linear current source, this will be done by keeping the step size small enough to accurately describe the current.

## Characterizing Model Component Values

This research uses values provided by SensL for their product. If these are not available or need to be verified, there are a series of tests that can obtain these values [24]. The values that need to be defined include the discharge capacitance (Cd), quench capacitance (Cq), quench resistor (Rq), discharge resistor (Rd), and grid capacitance (Cg).

Rq may be determined by placing the SiPM in forward bias and using Equation (10). In this equation, k (in Amps/V) is the slope of the linear portion of the IV curve. Rs is the sense resistor, and N is the number of microcells in one pixel of the SiPM [3] [24].

$$R_q = \left(\frac{1}{k} - R_s\right) * N \tag{10}$$

Biasing the SiPM at the breakdown voltage and measuring the capacitance / conductance of the SiPM will estimate the lumped capacitance and conductance ($C_m$ and $G_m$) [3].

The total capacitance of a microcell ($C_d + C_q$) may be determined as a function of overvoltage, given by Equation 6. The charge from dark counts can be tracked as overvoltage is increased to determine the total Q and breakdown voltage [3] [23].

$$Q_{total} = V_{overvoltage} * (C_d + C_q) \quad (11)$$

Using previously obtained values, the values of quench capacitor, discharge capacitor, and grid capacitor may be determined with Equations (12), (13), (14) [2] [3].

$$C_d = \left[\frac{1 + \omega^2(C_d + C_q)R_q^2}{\omega^2 N_{tot} R_q^2} G_m\right]^{\frac{1}{2}} \quad (12)$$

$$C_{qTOT} = C_{TOT} - C_{dTOT} \quad (13)$$

$$C_g = C_m - N_{tot}C_d - \frac{\omega^2 C_d^2 (C_d + C_q) R_q^2 N_{TOT}}{1 + \omega^2 (C_d + C_q)^2 R_q^2} \quad (14)$$

In Equations 12-14, $N_{tot}$ denotes the total number of microcells in the SiPM, $C_{qtot}$ denotes total quench capacitance for the SiPM ($C_{qtot} = C_q * N_{tot}$). Similarly, $C_{dtot}$ denotes total discharge capacitance for the SiPM ($C_{dtot} = C_d * N_{tot}$). $C_{tot}$ denotes the total capacitance of the SiPM $(C_q + C_d) * N_{tot}$, ω indicates the angular frequency at which the forward bias is applied. $C_m$ indicates the lump sum capacitance of the SiPM $(Cg + (C_q + C_d)(N_{tot}))$, measured in forward bias. $G_m$ denotes the conductance of the SiPM in the forward bias.

These tests have defined 4 out of 5 values needed for the model. The final value, Rd, may be estimated from single SPAD rise time. However, this method is not reliable. It can be assumed that Rd in the range of ~1k [2].

## SensL Micro J Series Properties

There are multiple brands and models of SiPM. A couple of the major brands include Hamatsu and SensL. This work uses the SensL Micro J Series 60035 SiPM. The relevant properties of the Micro J 60035 are listed in Table 2.1.

| Micro J Series 60035 | | | |
|---|---|---|---|
| Overvoltage | 2.5 | 5 | V |
| Vbreakdown | ~24.5 | | V |
| Number of Microcell | 22292 | | |
| Recharge Time | 50 | | ns |
| PDE @ 420nm | 38 | 50 | % |
| Gain | 2.90E+06 | 6.20E+06 | |
| Dark Count | 50 | 150 | $kHz/mm^2$ |
| Cross Talk | 8 | 25 | % |
| After pulsing | 0.75 | 5 | % |

**Table 2.1 Micro J Series Properties** *[25]*.

The photon detection efficiency (PDE) spectrum for the Micro J series is shown in Figure 17. The overlap between the Micro J series PDE for absorbing photons, and the CsI(Tl) scintillation emission spectrum provide the PDE to be used for the scintillation crystal. This is shown in Figure 18. Taking the ratio of the areas under the curves for the CsI(Tl) Intesity and SiPM Intesity provides an estimated ~22.3% for the PDE.

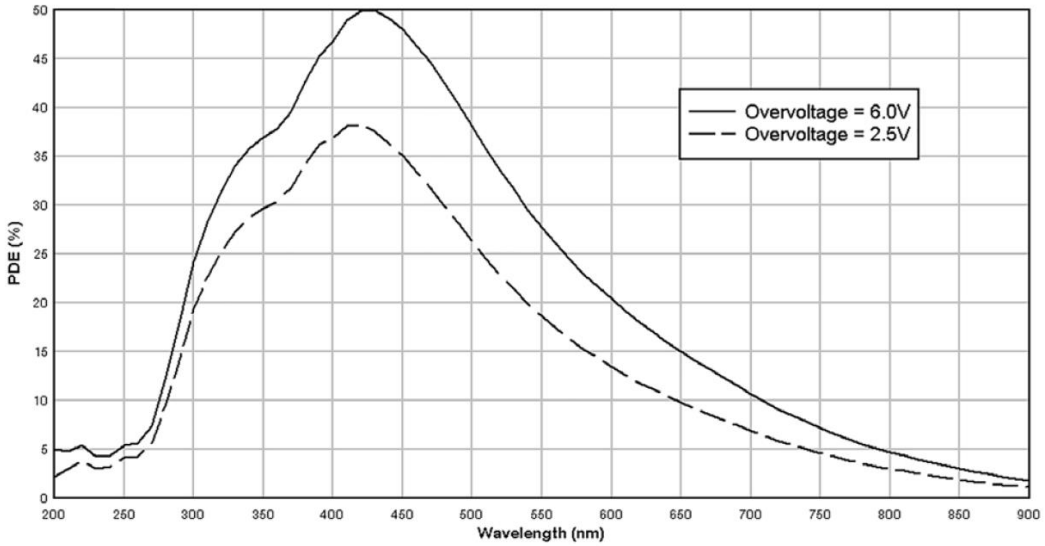


**Figure 17 SensL MicroJ Series PDE spectrum** *[25]*.

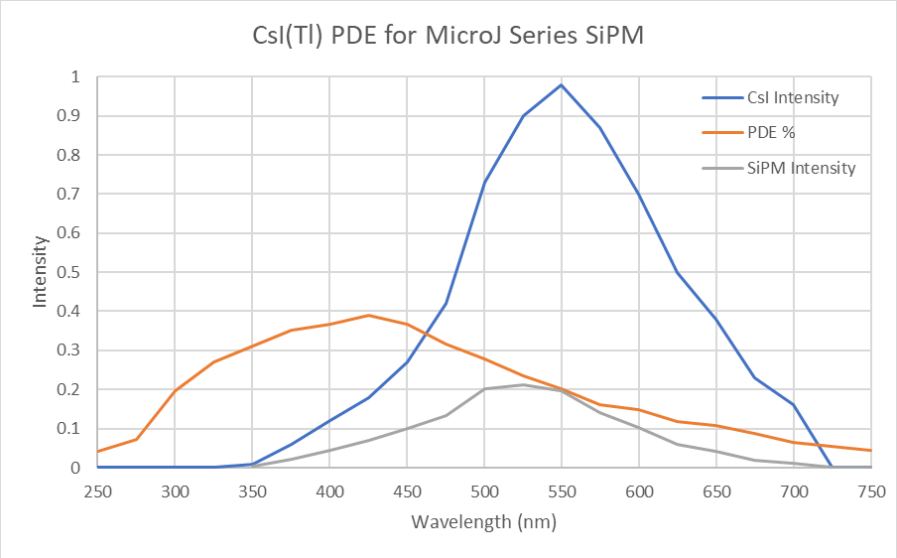


**Figure 18 Photon Detection efficiency for SiPM coupled with CsI(Tl). The difference in area between the CsI(Tl) Intensity curve, and the SiPM intensity curve should provide the PDE.**

## Modeling Scintillators

Depending on the decay constant of the scintillator and the recharge rate of the SiPM, there are a few different models that could be used. Figure 19 provides estimated responses for the extreme cases where the scintillation decay constant is very small ($\tau_d = 0$) and when it is very large ($\tau_d = 10$ μs). When very small, saturation effects are the dominant, leading to smaller than expected signals. When very large, after pulsing and crosstalk effects are the dominant, leading to a larger than expected signal.

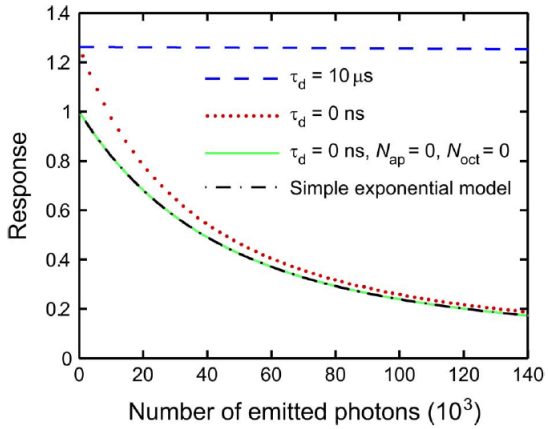


**Figure 19. Number of Photons vs. Output Response for SiPM** *[5]*.

The first model involves a light pulse with a decay constant much shorter than the recharge rate of the SiPM, and an assumption that the photon count stays constant. In this situation, the voltage model should work well and the number of fired microcells can be estimated with Equation (9). The photons all arrive before microcells recover, leading to a reduced response at higher photon counts [5]. This leads to a non-linear energy output that is smaller than expected. For convenience, Equation (8) has been transposed again below. Again, M is the total number of microcells, PDE is the Photon Detection Efficiency, and $N_{ph}$ is the number of incident photons. The switch should be opened on the timescale of the photon exposure.

$$N_{fired} = M * \left[1 - \exp\left(-PDE * \frac{N_{ph}}{M}\right)\right] \tag{9}$$

The second situation involves the opposite case, where the scintillation decay constant is much larger than the recharge rate. The photons are separated enough that the chance of two photons hitting the same microcell prior to full recharge (saturation) is low. After-pulsing and dark counts are still generated and the overall output is larger than expected, but the energy spectrum is still linear. This situation should work well with the current model, using a Monte Carlo style approach to generate a current source file.

The Final situation involves a middle case in which the scintillation decay constant is comparable to the recharge rate of the SiPM. In this situation, there is a higher probability of saturation, as well as cross-talk and after-pulsing. This situation should also work well with the current source model. A rigorous Monte Carlo Style approach should be used in order to produce an accurate current source file. A careful analysis should be done to ensure energy linearity if used for spectroscopy.

# 3. MATERIALS AND METHODS

## 3.1 Generating the Electrical Circuit Model

### 3.1.1 Selecting the Appropriate Electrical Circuit Model

This work uses a CsI(Tl) scintillation crystal coupled with a SensL MicroJ Series 60035 SiPM. In order to choose the appropriate model, a comparison between the scintillation decay constants and the SiPM recovery rate was made. The scintillation constants used were $\tau_{fall1}$ = 680ns (64%), $\tau_{fall2}$ = 3340 ns (36%) reported by Valentine et. al. [16]. The MicroJ 60035 has a recovery constant of 50ns. Since the recovery constant is much smaller than the scintillation decay constant, the current driven electrical circuit model was selected for this work.

### 3.1.2 Modeling the Electrical Circuit in SPICE

In an attempt to obtain the most accurate results, this research uses an SiPM electrical model provided by SensL of their Micro J-series 60035 SiPM under a Non-Disclosure Agreement (NDA). The model provided by SensL was a specialized version of the generic voltage model. Component values for the model were also provided along with a testbench circuit and signal output.

The circuit provided by SensL was altered into a current source driven configuration. This was done by replacing the voltage source with a current source and removing the voltage-controlled switch and discharge resistor. Finally, each circuit component reliant on the number of fired cells was set statically to model a single fired cell. The final circuit is similar to the generic circuit shown in Figure 20. After alterations, the circuit was compared to the original output for the anode signal. This is shown in Figure 21. For simplicity, the current truncation was removed from both models for this comparison.

All circuit modeling was done in LT Spice VII. The SiPM circuit in Figure 20 is designed to have the bias voltage applied to the cathode, and have the anode attached to the front-end electronics. The current source will drive the output signal by modeling all the microcell avalanche currents as a piecewise linear (PWL) function which will be written with a python script.

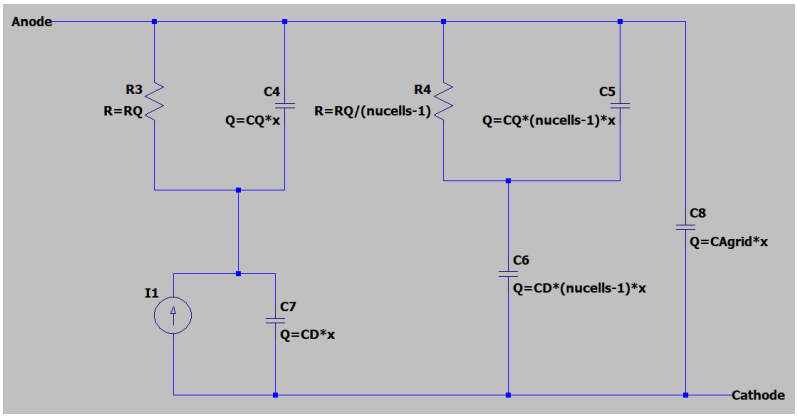


**Figure 20 Generic SiPM Circuit Model used for this research.**

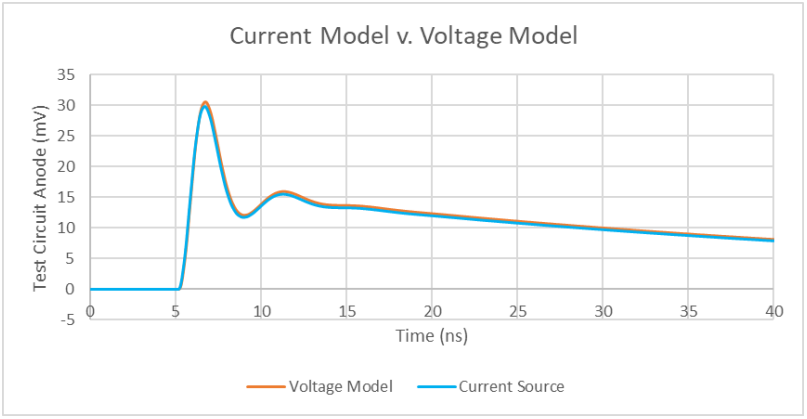


**Figure 21 Comparison of Original (Voltage) model and Altered (Current) model with the SensL test bench circuit. Current truncation was removed for both models.**

To reduce complexity on the top-level circuit diagram, the SiPM circuit will be represented with the symbol shown in Figure 22. The fast output node on the diagram represents a deviation between the SensL model and the generic model. In an effort to maintain SensL's model, the fast output components were left in the circuit, but the fast output was not used for this research.

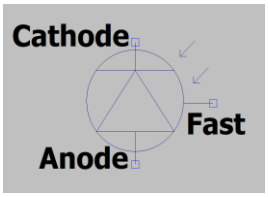


**Figure 22 SiPM schematic symbol.**

### 3.1.3 Modeling with Front-End electronics

The top level circuit diagram includes all front-end electronics included on the OSU board used for measurements. These include (4) SensL MicroJ Series 60035 SiPM, a generic high pass filter, a CR-113-R2.1 charge sensitive preamplifier from Cremat, and the biasing voltages. The Cremat charge sensitive preamplifier circuit model was obtained through the Cremat website [26]. The top level circuit used for signal simulation is shown in Figure 23. The nodes SiPM_Out and Amplifier_Out are used as signal output points.

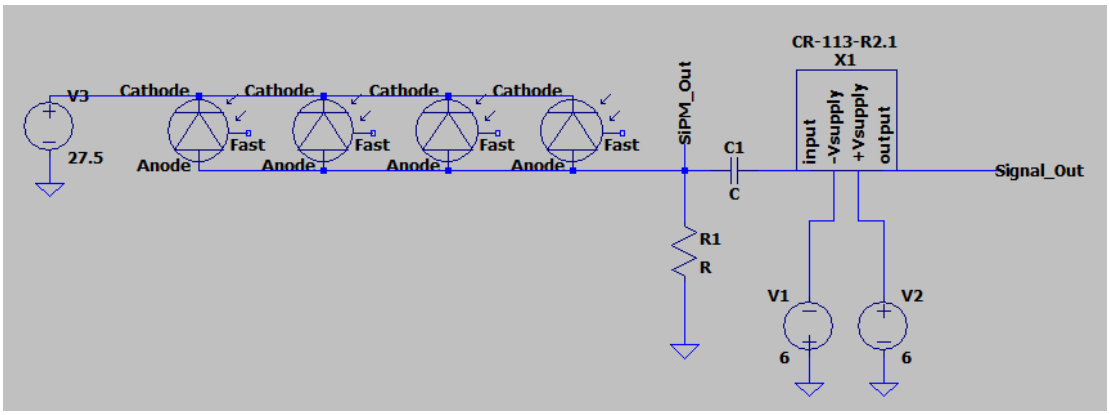


**Figure 23 Top level circuit model.**

### 3.1.4 Reducing Error within SPICE modeling

LT Spice VII was used for the circuit simulation portion of this work. LT Spice is a circuit simulator based on Berkeley SPICE and produced by Linear Technology.

Within LT Spice, there are many option variables that can be set to adjust the accuracy of the simulation, usually at the expense of a longer run time. For this research the alternative solver along with double precision options for measdgt and numdgt were used. All other constraints were left at the default settings.

The alternative solver was used over the normal solver to reduce round off error. The numdgt and measdgt variable options are used to set double precision for dependent variable data and .measure statement output, respectively. LT Spice will use double precision for these instances when the numdgt/measdgt are set to values large than 6. Double precision was used to reduce round off error.

Other relevant option variables include vntol, reltol, and abstol are used to set absolute voltage error tolerance, relative error tolerance, and absolute current error tolerance, respectively. These values are initially set to 1μV, 0.1%, and 1pA, respectively.

The gmin option variable is used to set the conductance added to every PN junction to aid in convergence. This value has the default value of 1pA. The itl1, itl2, and itl4 option variables are used to set the DC iteration count limit, DC transfer curve iteration count limit, and transient analysis time point iteration count limit, respectively. These values are initially set to 100, 50, and 10 respectively. These may be increased.

The maxstep variable option may also be used to restrict the maximum time step used by LT Spice. LT Spice uses a variable timestep that is reduced when certain criteria aren't met between time steps. While this may be useful, it was not used for this work. While other variable options are available, these were the most commonly used to reduce errors within the simulation.

## 3.2 Generating the Current Source Model

### 3.2.1 Time Step Analysis

In the previous section, the SiPM current driven electrical circuit was tested against the original circuit with an exponential decay modeled with Equation (8). This is an available source behavior in LT Spice and a custom source wasn't necessary. However, when generating a custom current source, a Piecewise Linear (PWL) source file must be used. This file lists coordinate points (time, current) at selected time steps. LT Spice then linearly interpolates between the points. Depending on the step size between defined points, the accuracy of the results can be impacted.

In order to check the accuracy dependence on time step, a python script was used to generate a PWL using Equation (8) at different time steps. The python script was very basic, using an array to hold time steps, and an array to hold the exponential value at the time step. These were then written to file as coordinates. The script is listed under Appendix A.

The anode output is compared to the original model provided by SensL. This time the model also uses the appropriate truncation as defined by the original circuit. The results are shown in Figure 24. A quick analysis evaluating the position of the peaks is provided in Table 3.1.

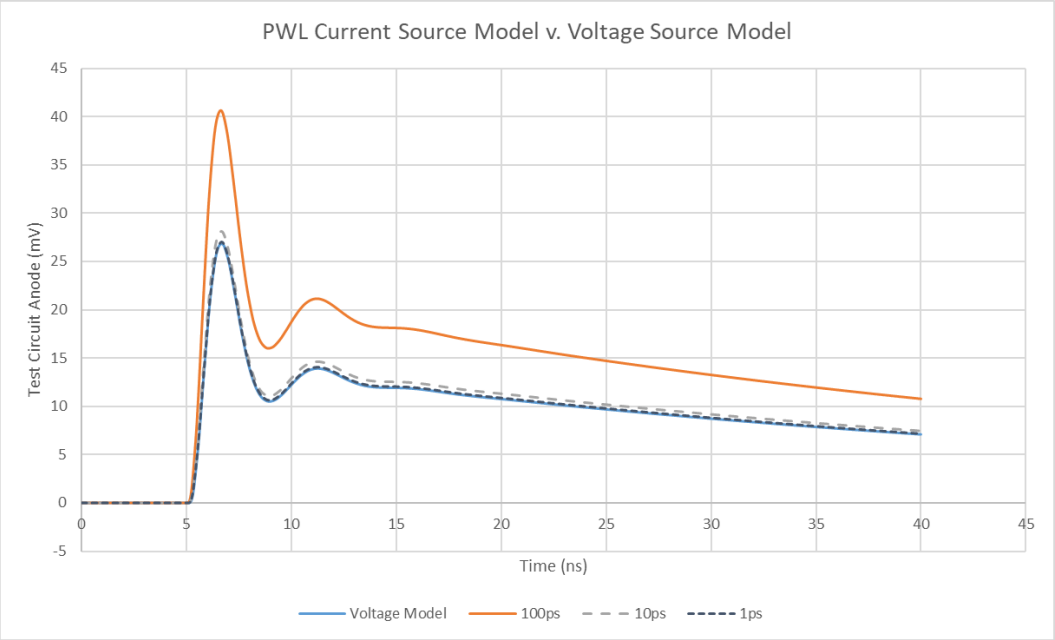


**Figure 24. Time Step Analysis**

| | Amplitude (mV) | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | **Original** | **100ps** | % Difference | **10ps** | % Difference | **1ps** | % Difference | **100fs** | % Difference |
| Anode Peak 1 | 26.837 | 40.636 | 51.418 | 28.117 | 1.130 | 27.020 | 0.326 | 26.913 | 0.281 |
| Anode Peak 2 | 13.914 | 21.146 | 51.978 | 14.625 | 1.460 | 14.055 | 0.654 | 13.999 | 0.610 |
| | **Timing (ns)** | | | | | | | | |
| | **Original** | **100ps** | % Difference | **10ps** | % Difference | **1ps** | % Difference | **100fs** | % Difference |
| Anode Peak 1 | 6.679 | 6.636 | -0.647 | 6.665 | -0.213 | 6.680 | 0.032 | 6.680 | 0.013 |
| Anode Peak 2 | 11.260 | 11.225 | -0.319 | 11.253 | -0.065 | 11.271 | 0.078 | 11.269 | 0.078 |

**Table 3.1 Time Step Analysis**

Looking at Table , 1ps was chosen as the time step used going forward. While runtime was not listed, 1ps offered a good tradeoff between accuracy and simulation runtime. It was seen that simulations run with 10ps time steps would run on the timescale of < 1minute, with 1ps time steps would run on the timescale of minutes, and with 100fs, simulations would run on timescale of hours. This was done with a laptop using an intel i7 processor.

### 3.2.2 Generating the Monte Carlo Model

The Monte Carlo based current source model was written in Python, using Jupyter Notebook. The program depends on numpy and matplotlib packages in addition to the built in math, os.path, and random libraries (random is the name of one of python's libraries).

The method used with this program is to generate a small array describing the avalanche signal for one micro cell firing and then superimpose it into the current signal array whenever a microcell is fired.

#### Setup

The Python script takes in the circuit model component values, the scintillation model values, as well as some user defined options, such as timestep and run time (in the form of number of tau).

#### Generating the Avalanche Signal and Recharge Array

The program creates two arrays to keep track of the time and amplitude of an avalanche signal. Equation (8) was used to define the amplitude of the avalanche signal at each time step.

The recovery of the microcell is estimated with the recovery rate constant given in Equation (6). This array estimates the percent of full charge the microcell has at each time step. This will be used when the same microcell is fired multiple times before full recovery, leading to a subsequent smaller than expected avalanche.

The array is stopped at the point where truncation would occur. Figure 25 shows the avalanche signal and microcell recharge array.

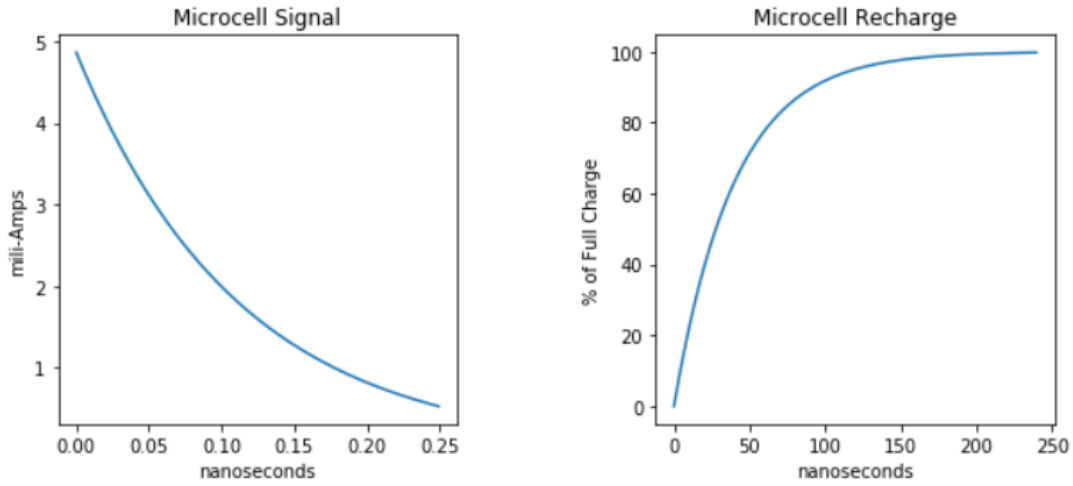


**Figure 25 Microcell arrays defining the avalanche signal and recovery.**

## Generating the Scintillation Signal

Just as the avalanche signal was designed with two arrays, the scintillation signal was also designed this way. The amplitude array was generated using an integrated version of Equation (1) to determine the photons per time step. This is shown in Equation (15), where a and b represent subsequent time steps. $I_0$ is determined with Equation (16) and (17).

$$N_{photons} = \int_a^b I_0 * \left( e^{\frac{t}{\tau_{fall}}} - e^{\frac{t}{\tau_{rise}}} \right) \tag{15}$$

$$N_{total\ photons} = \int_0^\infty I_0 * \left( e^{\frac{t}{\tau_{fall}}} - e^{\frac{t}{\tau_{rise}}} \right) \tag{16}$$

$$I_0 = \frac{N_{total\ photons}}{\tau_{fall} - \tau_{rise}} \tag{17}$$

The scintillation model uses 54,000 photons/MeV with decay constants $\tau_{fall1}$ = 680ns (64%) and $\tau_{fall2}$ = 3340 ns (36%). This provides a scintillation signal as

shown in Figure 26. When the step size is significantly small compared to the number of photons being generated, each time step generates a fraction of a photon. This can not occur, and so a separate array is generated that adds up each fraction of a photon until it reaches an integer value. The photon is assumed to fire at the point the value is an integer. This leads to the quantized photon array in Figure 26. The quantized photon array may only have values of 1 or 0, and when the photons fire very close together, it looks like a block as in Figure 26.

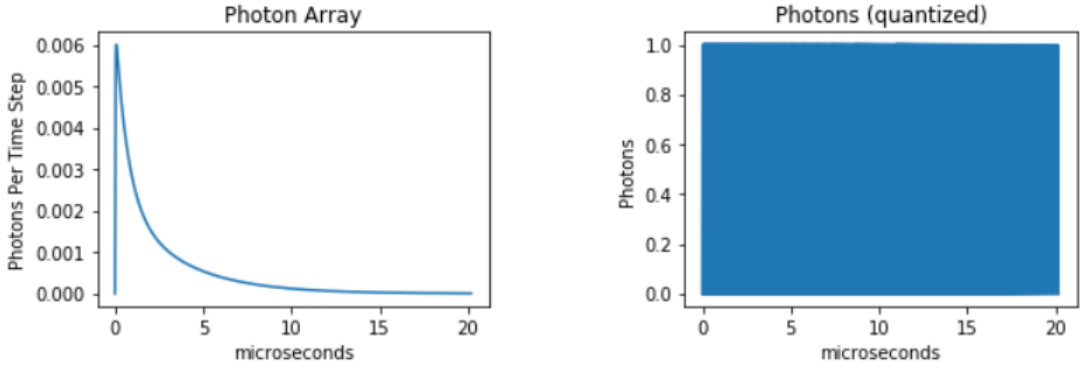


**Figure 26 Photon Arrays Per time step and quantized.**

## Generating the Fired Microcells Array

Now that there is an array statistically dictating when photons hit the silicon photomultiplier, a Monte Carlo model is used to determine whether the photon generates a fired cell and if so, which microcell is being fired. The model then goes on to determine whether after pulsing or cross talk has occurred. In addition, there is a chance for a dark pulse based on the frequency of dark pulses, and the time step selected.

In determining the probability that a photon will create a fired microcell, a random number generator is used with values 1-1000. If the number generated is less than the (PDE*Circuit Fill Factor)*1000, then the microcell is assumed to fire. The fill factor is the reduction factor produced when the crystal isn't fully covered by SiPM, in the case of this work, the fill factor was ~71%. Another random number with values between 1 and the total number of microcells is then generated to track

which microcell is firing. This assumes a uniform distribution of photons across the SiPM.

If a microcell has fired, the model goes on to check if a cross talk event, or an after-pulse event has occurred. This is only done for a scintillation photon induced microcell firing and is not considered after the firing of a cross talk event. The model then goes on to check is a dark pulse has occurred. The probability for this is determined as a per time step basis. This event is checked for regardless of whether a photon has induced a microcell firing.

At the end generating the Fire Cell microarray, the data can be viewed as a table of the microcell fired and the time at which the event took place that is originally sorted by time. In order to determine whether microcells have fired prior to a full recovery, the data is sorted by microcell number and firing times are compared. For any duplicates that fired within the recovery time window, they are then logged into a separate array. The recovery time window was set to $6\tau$ which would charge the microcell to ~99.75% of the max value. This provides two arrays, one of normally fired microcells, and one of microcells fired prior to full recovery that can be used for generating the current array.

## Generating the Current Array

At this point an empty current array is created that will eventually turn into the PWL source file. There are two fired microcell arrays that must be processed into the current array. The first fired microcell array is the full signal arrays and is used to superimpose the avalanche signal at each point where a microcell was determined to have fired. This is done by adding the amplitude array of the avalanche to each corresponding timestep in the current array.

The second microcell array includes the duplicates and requires further processing before the avalanche signal can be added. The first check is whether the duplicate microcell fired prior to the end of the avalanche, in which they would be considered the same avalanche. Subsequently, the first avalanche is treated as a normal avalanche, and the second is reduced based on the recovery of the SiPM at the

point of the avalanche. A possible current array generated with this monte carlo method is shown in Figure 27. The current array is then written to file to be used with the electrical model. Many of the coordinates lie between microcells firing. In order to shorten the size of the file, any points that were the same as the previous and the next point were excluded from the text file. The size of the current source text files depends on the step size and the scintillation pulse duration. Typical file sizes for 1ps time steps coupled with a CsI(Tl) scintillation pulse are in the megabyte range.

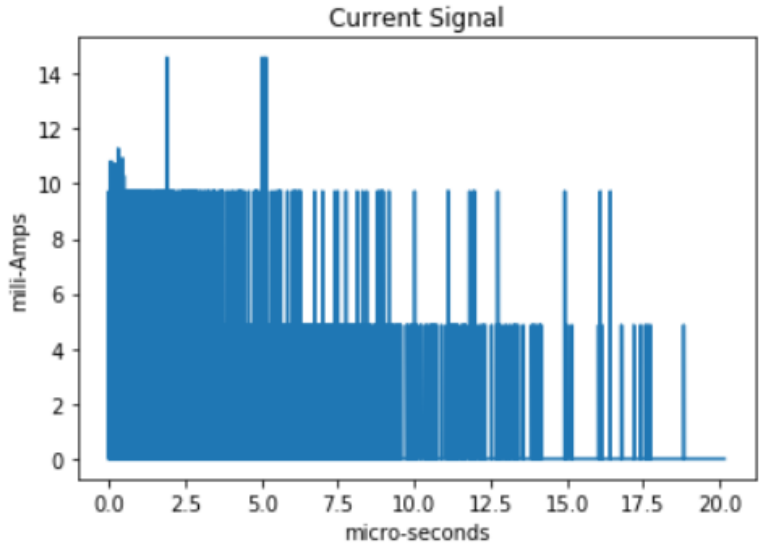


**Figure 27 Current Signal produced from the Model**

## 3.3 Physical Circuit Measurements

In order to compare the accuracy of the simulated results, measurements were taken from the physical circuit modeled in SPICE. The physical circuit was printed on PCB and contained the same elements as the SPICE model. The crystal used was measured at 1” x 0.5” x 0.5”. The SiPM measuring 6mm x 6mm were placed at the four corners with a 2mm gap between each. This produced a fill factor of ~0.71. The crystal was covered with reflective teflon tape on five faces and covered with a plastic casing to prevent light. The final face was coupled to the SiPM with optical grease to reduce the incidence of refraction.

Measurements were taken with a Tektronix oscilloscope. Probes were latched to the circuit at the SiPM output and amplifier output. A Cs-137 source was placed in close proximity to the CsI(Tl) scintillation crystal in order to obtain the output signal from a 662 keV photon. The trigger level on the oscilloscope was adjusted in an attempt to obtain the maximum output. These measurements were saved to a flash drive and were analyzed using Microsoft excel.

# 4. Results and Analysis

## 4.1 Simulated Results

There were two nodes within the front-end electronics where the simulation was chosen for comparison. They were the SiPM output node and the Amplifier output node.

Due to the random behavior of the simulation, the output from 5 runs were plotted together for both outputs. The SiPM output is shown in Figure 28 and the amplifier output is shown in Figure 29.

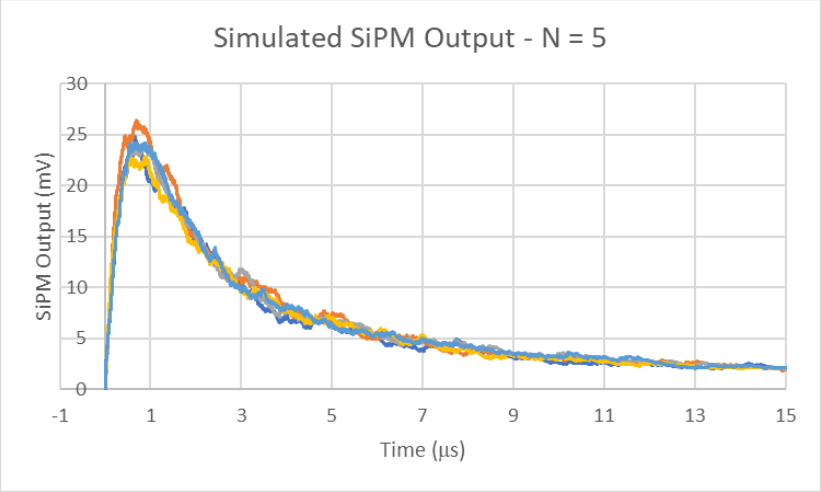


**Figure 28. Simulated SiPM Output. 5 Output are plotted.**

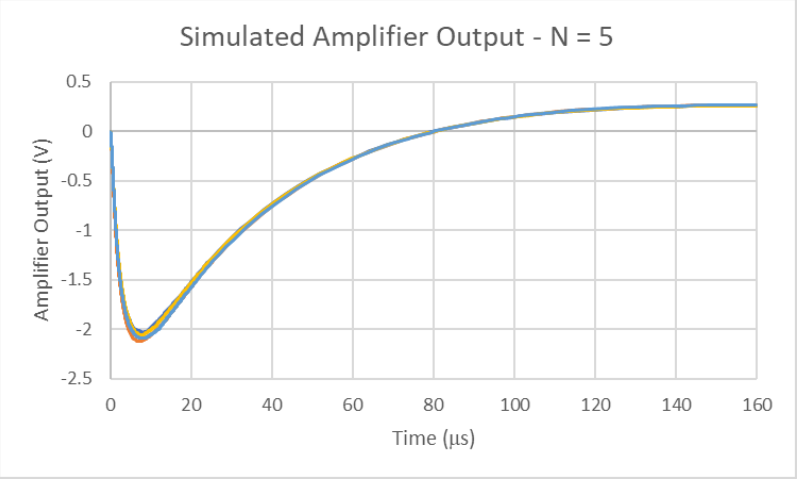


**Figure 29. Simulated Amplifier Output. 5 Output are plotted.**

The results from the simulation showed much greater variability in the SiPM output than in the amplifier output. This is most likely due to the RC circuit in the transimpedance amplifier. The charge is collected on the capacitor and released across the resistor at the rate dictated by the RC time constant, smoothing out the SiPM output. Table 4.2 provides the average peak values and curve characteristics for the simulated output. The ranges all indicate $2\sigma$.

The SiPM output had a 10.4% range of values in peak amplitude and a 33% range of values in the peak timing. This large range is expected. Each photon throughout the scintillation is probability based. Analyzing a much larger number of models may reduce the range. The amplifier output had a 3.5% range in peak amplitude and a 12.4% range in peak timing. The amplifier output was seen to have a much smaller range.

The signals can also be described by their rise and fall components. Table 4.1 lists the average rise and fall seen for the simulated output. An estimate is made for a decay and rise constant, these are used for exponential signals and may not be suitable. The 10% to 90% rise/fall times are a more generalized approach.

| **Simulation** | | |
|---|---|---|
| **SiPM Output** | Average | Error (+/-) |
| Peak Amplitude (mV) | 24.49 | 2.55 |
| Peak Time (μs) | 0.76 | 0.26 |
| Rise Time (μs) | 0.43 | 0.11 |
| Fall Time (μs) | 11.56 | 1.54 |
| Tau_rise (μs) | 0.22 | 0.05 |
| Tau_fall (μs) | 2.79 | 0.37 |
| | | |
| **Amplifier Output** | Average | Error (+/-) |
| Peak Amplitude (V) | -2.08 | 0.07 |
| Peak Time (μs) | 7.81 | 0.97 |
| Rise Time (μs) | 3.70 | 0.28 |
| Fall Time (μs) | 50.47 | 1.19 |
| Tau_rise (μs) | 2.03 | 0.15 |
| Tau_fall (μs) | 31.10 | 1.68 |

**Table 4.1 Top) Simulated SiPM Timing Analysis**
**Bottom) Simulated Amplifier Timing Analysis**

The simulated SiPM output signal was seen to have a rise time of 0.431 +/- 0.11 μs, a fall time of 11.56 +/- 1.54 μs, a tau_rise of 0.23 +/- 0.05 μs and a tau_fall of 2.79 +/- 0.37 μs. The simulated amplifier output signal was seen to have a rise time of 3.70 +/- 0.28 μs, a fall time of 50.47 +/- 1.19 μs, a tau_rise of 2.03 +/- 0.15 μs and a tau_fall of 31.10 +/- 1.68 μs.

## 4.2 Measured Results

The following signals were measured from the physical circuit with the use of an oscilloscope. Each represents the signal from a CsI(Tl) crystal being exposed to a Cs-137 source. Multiple measurements were taken from each node. The SiPM output node was measured five times and is shown in Figure 30. The amplifier output was measured four times and is shown in Figure 31.

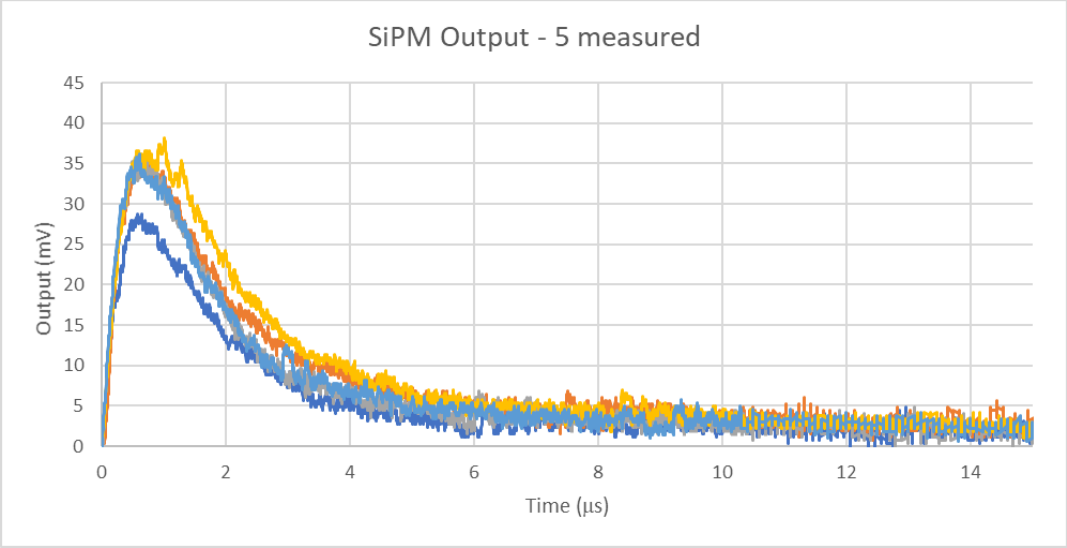


**Figure 30 Measured SiPM signal Output.**

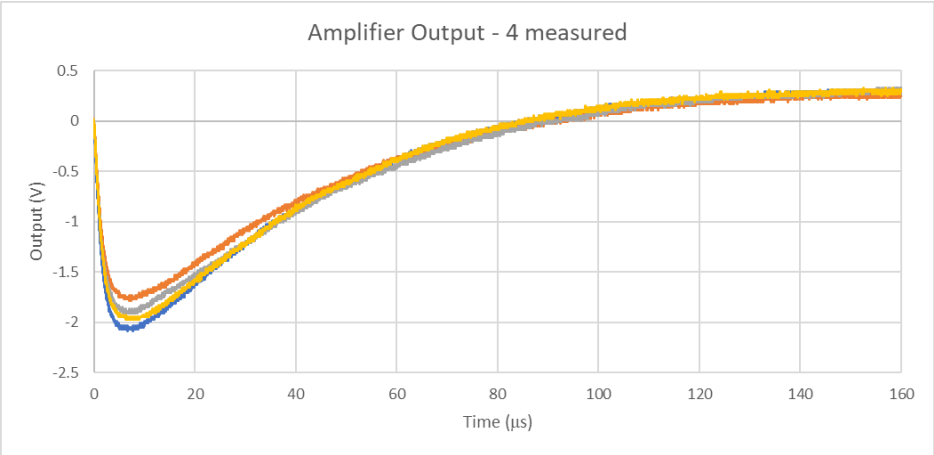


**Figure 31 Measured Amplifier signal output.**

The measurements again show much greater variability in the SiPM output than in the amplifier output. Table 4.2 provides the average peak values and curve characteristics for the simulated output. The ranges all indicate 2σ.

The SiPM output had a 20.5% range of values in peak amplitude and a 48.2% range of values in the peak timing. The amplifier output had a 12.9% range in peak amplitude and a 16.2% range in peak timing. The amplifier output was seen to have a much smaller range than the SiPM output. The large range could indicate the trigger level of the oscilloscope needed to be increased.

| **Measured** | | |
|---|---|---|
| **SiPM Output** | Average | Error (+/-) |
| Peak Amplitude (mV) | 34.96 | 7.17 |
| Peak Time (µs) | 0.71 | 0.34 |
| Rise Time (µs) | 0.40 | 0.05 |
| Fall Time (µs) | 5.24 | 1.41 |
| Tau_rise (µs) | 0.23 | 0.03 |
| Tau_fall (µs) | 1.83 | 0.31 |
| | | |
| **Amplifier Output** | Average | Error (+/-) |
| Peak Amplitude (V) | -1.94 | 0.25 |
| Peak Time (µs) | 6.36 | 1.03 |
| Rise Time (µs) | 3.02 | 0.21 |
| Fall Time (µs) | 57.23 | 5.88 |
| Tau_rise (µs) | 1.64 | 0.46 |
| Tau_fall (µs) | 38.80 | 3.43 |

**Table 4.2 Top) measured SiPM timing analysis**
**Bottom) measured amplifier timing analysis**

The measured SiPM output signal was seen to have a rise time of 0.40 +/- 0.05 μs, a fall time of 5.24 +/- 1.41 μs, a tau_rise of 0.23 +/- 0.03 μs and a tau_fall of 1.83 +/- 0.31 μs. The simulated amplifier output signal was seen to have a rise time of 3.02 +/- 0.21 μs, a fall time of 57.23 +/- 5.88 μs, a tau_rise of 1.64 +/- 0.46 μs and a tau_fall of 38.80 +/- 3.43 μs. These measurements will be used to measure the accuracy of the model.

## 4.3 Comparison Between Measurements and Simulation

Both the measured and simulated output signals share a similar shape. A comparison of the outputs is shown in table 4.3. The simulated SiPM output showed a greater discrepancy than the amplifier output. This is expected as the charge is integrated with the amplifier, smoothing the curve.

These discrepancies can be better visualized with overlapping plots as in Figure 32 and Figure 33. Figure 32 provides the standard plots and can be used to compare magnitude as well as shape. Figure 33 has each plot normalized to its peak and is used to compare the shapes of the curves.

| **Simulation and Measurements Comparison** | | | |
|---|---|---|---|
| **SiPM Output** | Simulated | Measured | % Difference |
| Peak Amplitude (mV) | 24.49 | 34.96 | 29.9 |
| Peak Time (μs) | 0.76 | 0.71 | 6.9 |
| Rise Time (μs) | 0.43 | 0.40 | 8.1 |
| Fall Time (μs) | 11.56 | 5.24 | 120.4 |
| Tau_rise (μs) | 0.22 | 0.23 | 5.5 |
| Tau_fall (μs) | 2.79 | 1.83 | 52.7 |
| | | | |
| **Amplifier Output** | Simulated | Measured | % Difference |
| Peak Amplitude (V) | -2.08 | -1.94 | 7.2 |
| Peak Time (μs) | 7.81 | 6.36 | 22.9 |
| Rise Time (μs) | 3.70 | 3.02 | 22.7 |
| Fall Time (μs) | 50.47 | 57.23 | 11.8 |
| Tau_rise (μs) | 2.03 | 1.64 | 23.8 |
| Tau_fall (μs) | 31.10 | 38.80 | 19.8 |

**Figure 4.3 Comparison of measured and simulated output**

Figure 32 shows a close match to the measured pulse. The peak is seen to be smaller than the measure peak. Also indicated in Figure 32, the rising edge looks very similar, but the falling edge is very different.

Evaluating the amplifier output signal again indicates a similar rising edge, and a slightly differing falling edge. The differences noted in the amplifier output are smaller in comparison to the SiPM output due to the shaping from the charge sensitive preamplifier. The charge sensitive preamplifier smooths the signals out into similar output. This is shown in Figure 34 and Figure 35.

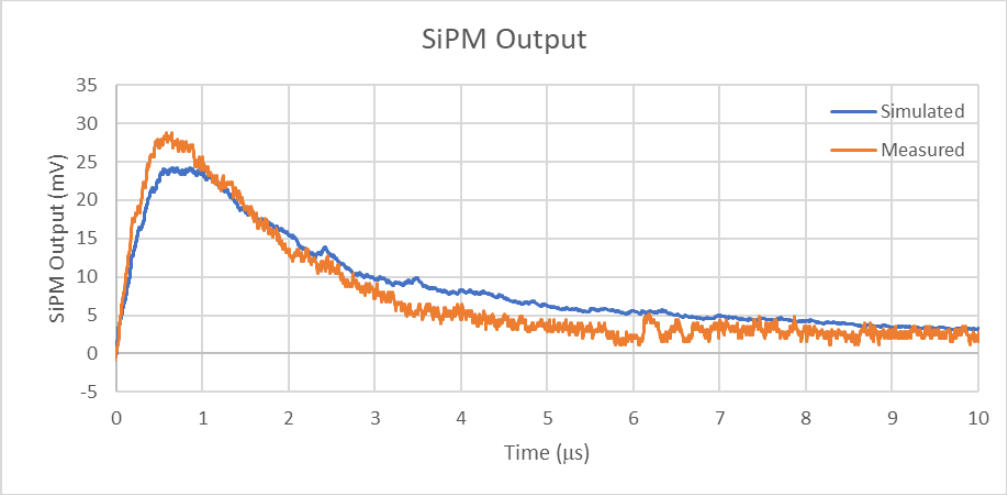


**Figure 32. Measured vs Simulated SiPM Output.**

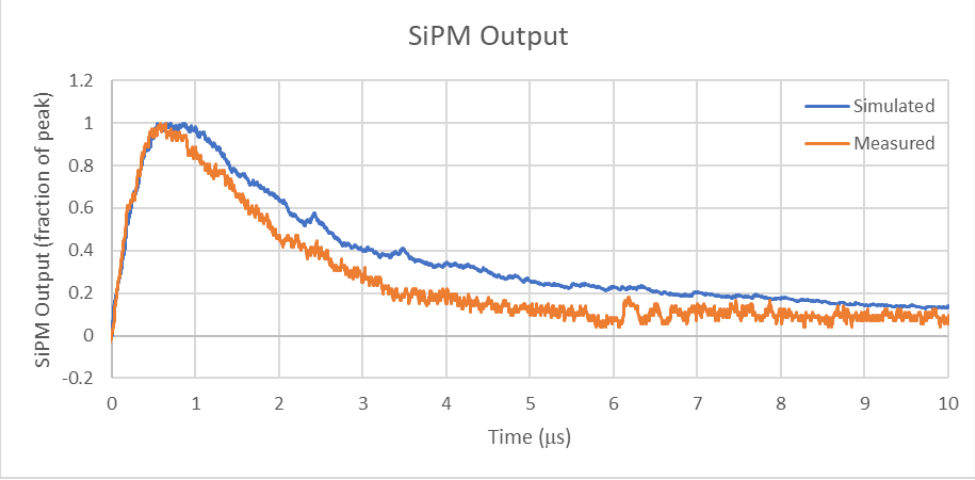


**Figure 33. Measured vs Simulated SiPM Output (Normalized) plots. Each plot was normalized to its peak.**

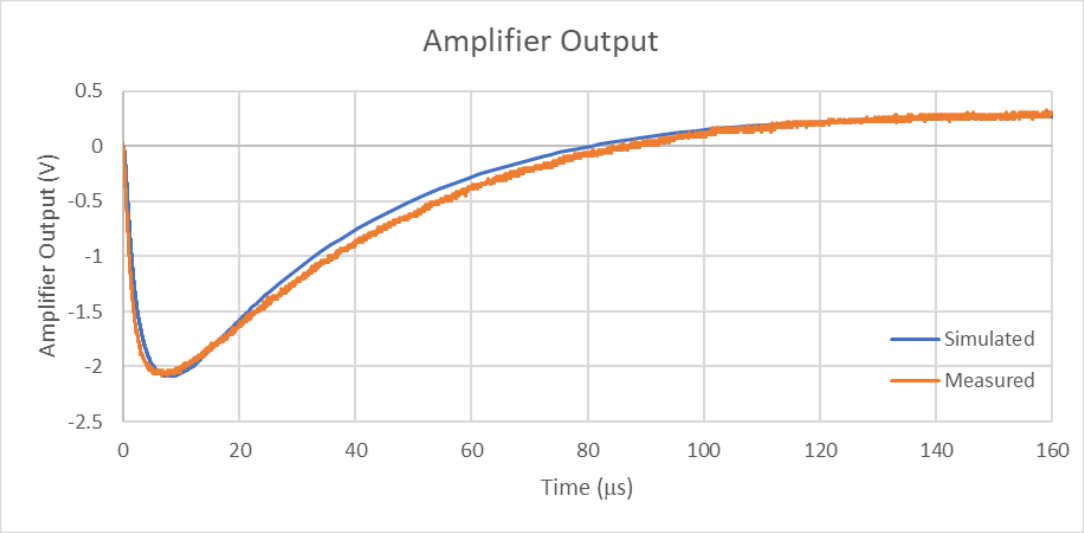


**Figure 34 Simulation vs Measured output from the amplifier.**

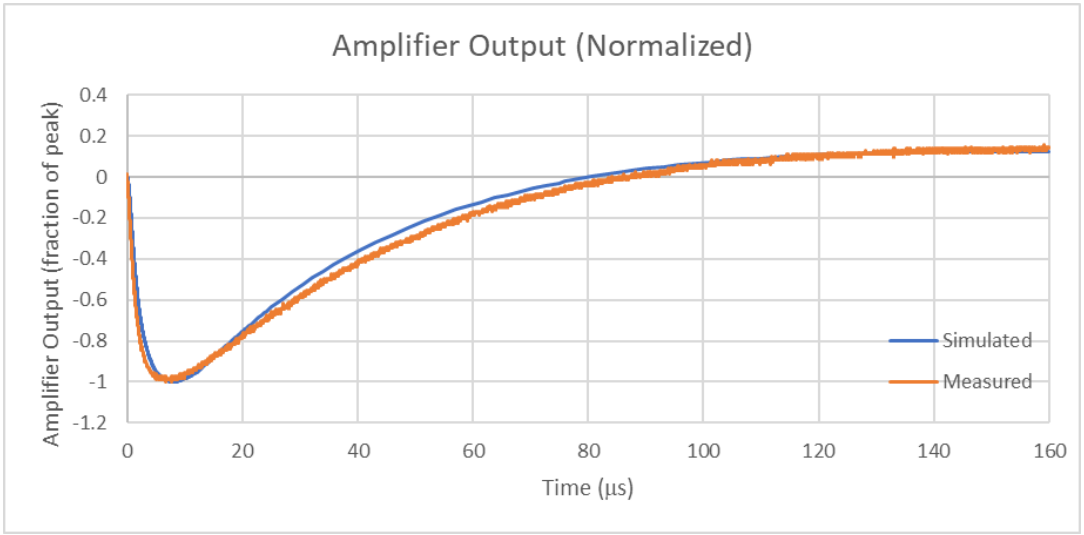


**Figure 35 Simulation vs Measured output from the amplifier. Normalized to peak values.**

Overall the fit matches well for estimating the scintillation pulse from a 662 keV photon. The falling edge of the simulation does not quite fit the falling edge of the measured SiPM output. This could be due to inaccuracy in the scintillation model chosen. The decay constants for CsI(Tl) have been reported to change with the magnitude of incident energy and this may play a role [17].

## 4.4 Comparison of CsI(Tl) Scintillation Models

There was large variance between the four scintillation models proposed for CsI(Tl). Figures 37-40 evaluate the differences between the output of each model. Each was simulated at 54k ph/MeV. It was seen that with the front-end electronics, as the scintillator decay time increased the SiPM output decreased, this is due to the same amount of charge being distributed over a larger time.

There was seen a very large range in the amplitudes between the models, but the models matched well with the shape of the measured SiPM output. Of the models looked at, the model proposed by Sentfeld-Kazuch showed a much smaller output than the measured result [17]. The models proposed by Schotanus et. al. and Valentine et. al. underestimated the SiPM output but showed a closer shape to the amplifier output. Overall, the model proposed by Valentine et al. worked best in matching the measured shape of both the SiPM output and amplifier output.

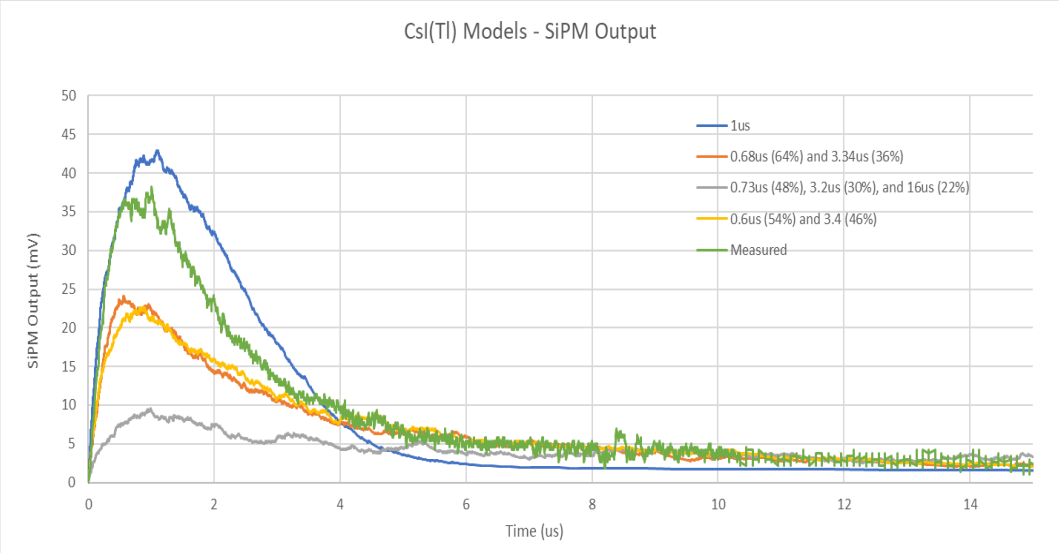


**Figure 36 CsI(Tl) Models - SiPM Output**

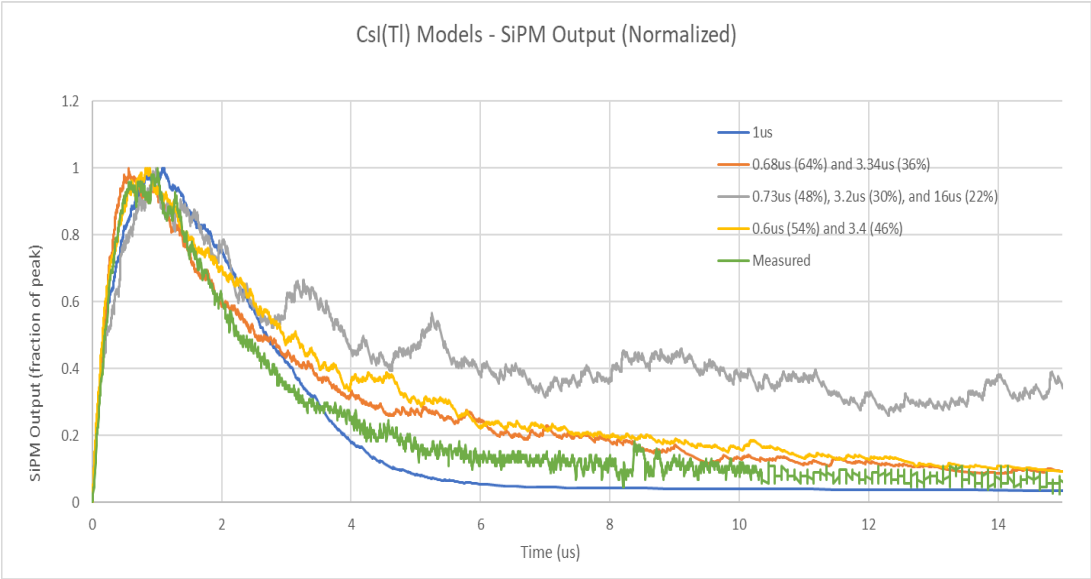


**Figure 37 CsI(Tl) Models - SiPM Output (Normalized)**

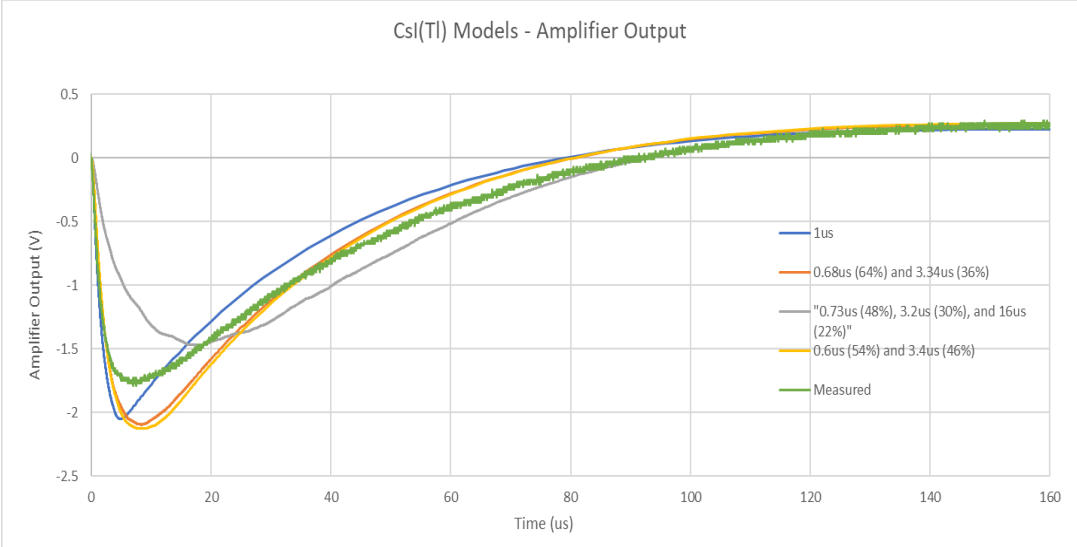


**Figure 38 CsI(Tl) Models - Amplifier Output**

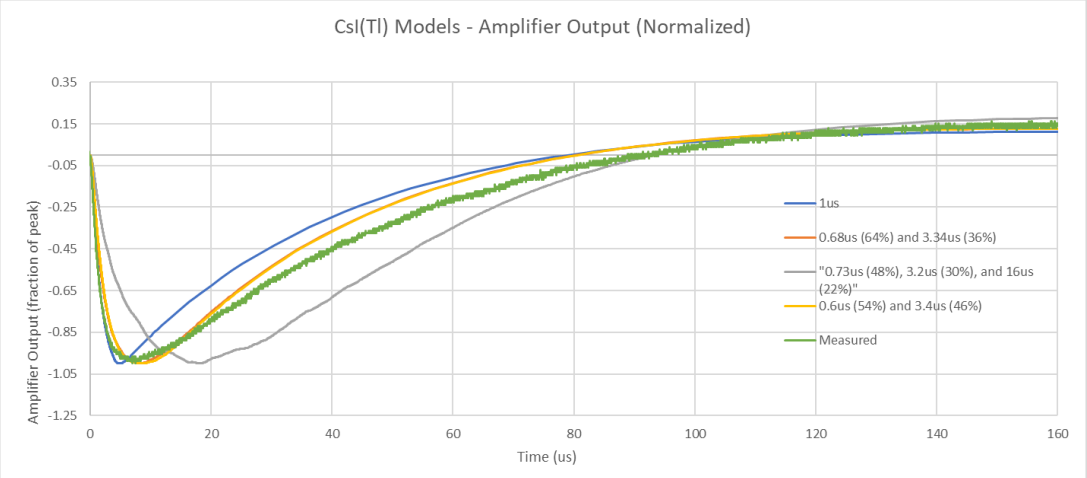


**Figure 39 CsI(Tl) Models - Amplifier Output (Normalized)**

## 4.5 Predicted output for Common Scintillators

Based on the front-end electronic used in the circuit, the Figure 40-41 are the predicted outputs for the SiPM and Amplifier nodes, respectively. $SrI_2(Eu)$ was estimated to have a decay constant of 1μs, a scintillation output of 100,000 photons/ MeV, and a PDE of 32%. NaI(Tl) was estimated to have a decay constant of 0.23 μs, a scintillation output of 38,000 photons/MeV, and a PDE of 35%. BGO was estimated to have a decay constant of 0.3 μs, a scintillation output of 8,200 photons/MeV, and a PDE of 30%. The fill factor for the circuit was assumed to be 71%.

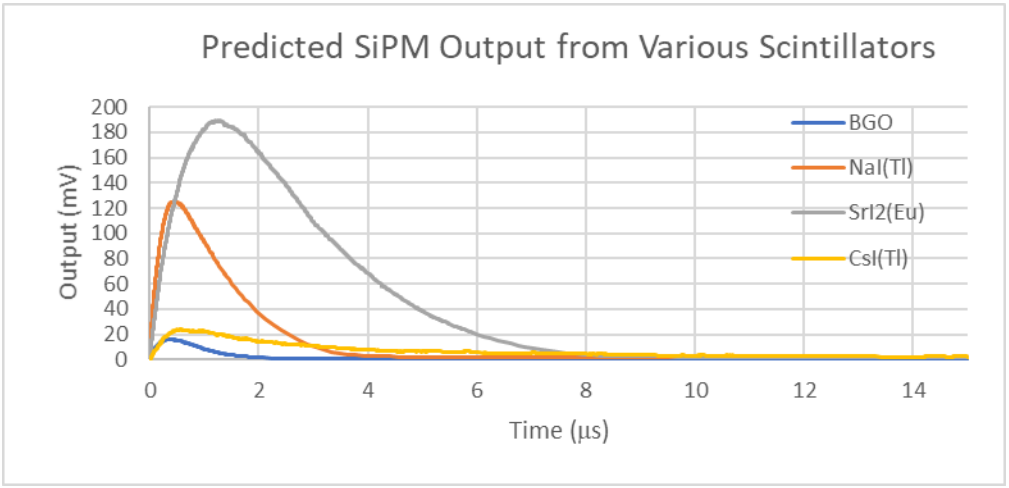


**Figure 40 Predicted SiPM Output for Various Scintillators**

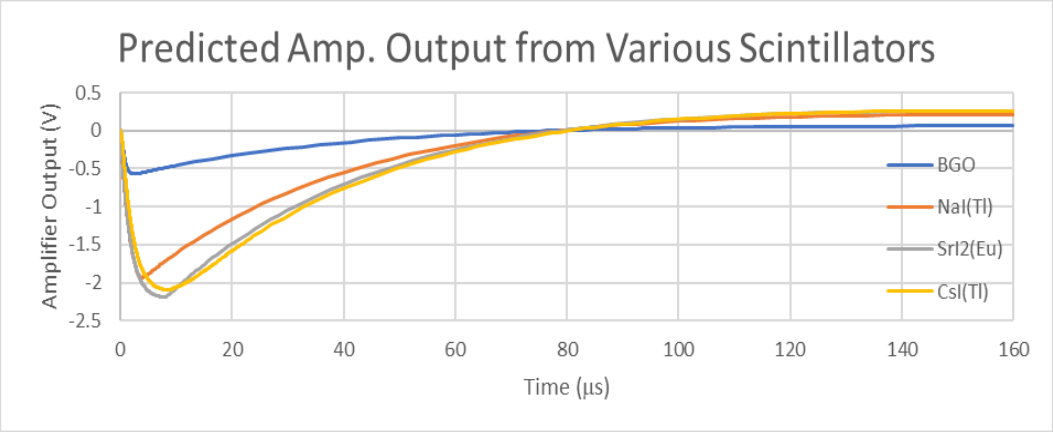


**Figure 41 Predicted Amplifier Output for Various Scintillators**

It is seen that the SiPM output for the $SrI_2(Eu)$ had the largest amplitude which was expected. BGO output was also expected to be small. The output from the amplifier showed unexpected results. The output from the Amplifier had the same output for $SrI_2(Eu)$ as for CsI(Tl).

It is suspected that saturation is occurring in the amplifier. Using the gain of the SiPM, it is expected that each fired microcell would produce 0.54 pC. The maximum charge collection supported by the amplifier is 2.08 nC, leading to the maximum number of fired cells that could be collected without saturating the amplifier to be ~3800.

Evaluating whether saturation is occurring in the Cremat preamplifier, an estimated number of fired cells for 662 keV was determined. The $SrI_2(Eu)$ output had an estimated 9.2 nC (~16,860 fired cells), and the CsI(Tl) had an estimated output of 3.09 nC (~5,700 fired cells). NaI(Tl) also showed an estimated output of 3.9nC (~7,200 fired cells) and possibly has saturation effects as well. BGO is not expected to have shown amplifier saturation with 0.73 nC (~1348 fired cells). The simulation is estimating that the input to the preamplifier should be reduced for higher energy radiation use, such as for detection of 662 keV as is.

# 5. CONCLUSIONS

## 5.1 Summary

Overall this research attempts to take a 662 keV photon through the process of hitting a CsI(Tl) crystal and accurately provide the voltage output of the front-end electronics within the detector.

In order to achieve this, the following steps were taken:

1. The CsI(Tl) scintillation process was modeled using the decay constants and photon yields of published reports.
2. The interactions between the scintillation photons and the SiPM was modeled in Python with Monte Carlo approach to include nonlinear effects such as after pulsing, crosstalk, dark counts, and saturation.
3. The microcells fired were modeled as avalanche current pulses, placed into an array, and written to file. This was used as the source current for the SiPM.
4. The SiPM circuit came from SensL and was used to model the SiPM component. The Cremat amplifier model came from the manufacturer as well. The rest of the front-end electronic components were modeled with ideal components.
5. Measurements from a physical circuit were taken and compared to simulated results.

The SiPM output had a smaller amplitude and slower decay time when compared to measurements, with other factors such as rise time matching well. The amplitude difference is likely due to manufacturing variability between crystals. The falling edge differences may be produced as a result of error in the CsI(Tl) scintillation model used, or possibly as error produced within the SPICE simulation.

As a simulation tool, and without adjusting for variability in light yield between crystals, the output was seen to match well with the output expected from a physical circuit.

## 5.2 Sources of Error

During simulation, there are multiple models used in serial. The output from each model is used as the input to the next. This leads to a compounding effect as the signal moves through the system.

It is suspected that the scintillation model used for CsI(Tl) may have introduced some error in the model. There have been multiple studies with multiple results discussing the decay constants of CsI(Tl). There have also been varying reports on light output as was discussed in the literature review sections.

It is expected that the difference in amplitude between the simulated and measured output was the result of a difference between the simulated light yield of 54,000 photons/MeV and the actual light yield of the CsI(Tl) crystal used. This is most likely due to variability within manufacturing of the crystal.

There may also be small error introduced from the measurements taken. The measurement may have been slightly less than the full 662 keV.

While it was not investigated, there may also be error in the SiPM circuit model provided by SensL and the amplifier model provided by Cremat. However, these were provided by theier respective companies and are assumed to be very accurate.

There may also be a small amount of error introduced through the assumption of a uniform photon distribution. Geant4, The C++ particle toolkit, was used in a previous study by Pulko et. al. for modeling a fast scintillator. This would better detect saturation effects, though it is not expected to have a large impact on modeling a slow scintillator such as CsI(Tl).

A small amount of error could have been introduced by neglecting the loss of photons from the reflective surfaces of the crystal, but losses are expected to be < 1%.

While noise is present in the actual system, it was neglected in this work and may influence the output variability of the simulated current file. It is also known that using a piecewise linear text file as the current input introduces a small amount of error based on the step size selected. Rounding error introduced from the Spice simulation is also present, though steps were taken to reduce this.

## 5.3 Future Work

Possible improvements could be made on the modeling of the photon interactions with the SiPM. This could be done through the use of a particle physics application to track the production of photons within the crystal and track the locations in which they interact with the SiPM. This would more accurately predict the saturation effects that occur.

Noise could be introduced in the system to more accurate predict the output variance within the system. This would allow for a pulse height spectrum with FWMH calculations to estimated on the scintillation detector.

Within the LT Spice simulation, modeling error may be able to be reduce through the use of the option variables mentioned earlier in this work. This could also lead to a more accurate output signal.

# BIBLIOGRAPHY


[1] A. K. Jha, H. T. v. D. Dam, M. A. Kupinski and E. Clarkson, "Simulating Silicon Photomultiplier Response to Scintillation Light," *IEEE Transactions on Nucler Science,* vol. 60, no. 1, pp. 336-351, 2013.

[2] F. Acerbi and S. Gundacker, "Understanding and simulating SiPM," *Nuclear Instruments and Methods in Physics Research A,* vol. 926, pp. 16-35, 2019.

[3] F. Corsi, A. Dragone, C. Marzocca, A. D. Guerra, P. Delizia, N. Dinu, C. Piemonte, M. Boscardin and G. F. D. Betta, "Modelling a silicon photomultiplier (SiPM) as a signal source for optimum front-end design," *Nuclear Instruments and Methods in Physics Research A,* vol. 572, pp. 416-418, 2007.

[4] S. Cova, M. Ghioni, A. Lacaita, C. Samori and F. Zappa, "Avalanche photodiodes and quenching circuits for single-photon detection," *Applied Optics,* vol. 35, no. 12, pp. 1956-1976, 1996.

[5] H. T. van Dam, S. Seifert, R. Vinke, P. Dendooven, H. Lohner, F. J. Beekman and D. R. Schaart, "A Comprehensive Model of the Response of Silicon Photomultipliers," *IEEE Transactions on Nuclear Science,* vol. 57, no. 4, pp. 2254-2266, August 2010.

[6] D. Liksonov, B. Barbier and J. Chavanelle, "Development of a Simulation Tool to Predict the Behavior of a SiPM Detector Coupled to a Scintillation Crystal," in *IEEE Nuclear Science Symposium conference record. Nuclear Science Symposium*, 2010.

[7] G. F. Knoll, Radiation Detection and Measurement, John Wiley & Sons, 2000.

[8] Saint-Gobain, "NaI(Tl) and Polyscin NaI(Tl) Sodium Iodide," [Online]. Available: https://www.crystals.saint-gobain.com/products/nai-sodium-iodide. [Accessed 2 August 2019].

[9] T. Matthews, "Wikipedia," 5 April 2009. [Online]. Available: https://en.wikipedia.org/wiki/Intersystem_crossing#/media/File:ISC_excited_states.png. [Accessed 30 July 2019].

[10] K. S. Krane, Introductory Nuclear Physics, John Wiley & Sons, 1988.

[11] P. Atkins and J. d. Paula, Atkins' Physical Chemistry, New York: W. H. Freeman and Company, 2006.

[12] J. D. Valentine, D. K. Webe, G. F. Knoll and C. E. Moss, "Temperature Dependence of CsI(Tl) Absolute Scintillation Yield," *IEEE Transactions on Nuclear Science,* vol. 40, no. 4, pp. 1267-1274, 1993.

[13] I. Holl, E. Lorenz and G. Mageras, "A Measurement of the Light Yield of Common Inorganic Scintillators," *IEEE Transactions on Nuclear Science,* vol. 35, no. 1, pp. 105-109, 1988.

[14] Saint-Gobain, "CsI(Tl) Thallium activated Cesium Iodide," [Online]. Available: https://www.crystals.saint-gobain.com/products/csitl-cesium-iodide-thallium. [Accessed 31 July 2019].

[15] P. Schotanus, R. Kamermans and P. Dorenbus, "Scintillation characteristics of pure and Tl-doped CsI crystals," *IEEE Transactions on Nuclear Science,* vol. 37, no. 2, pp. 177-182, 1990.

[16] J. D. Valentine, W. W. Moses, S. E. Derenzo, D. K. Wehe and G. F. Knoll, "Temperature dependence of CsI(tl) gamma-ray excited scintillation characteristics," *Nuclear Instruments and Methods in Physics Research,* vol. 325, no. 1-2, pp. 147-157, 1993.

[17] A. Syntfeld-Kazuch, M. Moszynski, L. Swiderski, W. Klamra and A. Nassalski, "Light Pulse Shape Dependence on gamma-ray energy in CsI(Tl)," *IEEE Transactions on Nuclear Science,* vol. 55, no. 3, pp. 1246-1250, 2008.

[18] A. Syntfeld-Kazuch, L. Swiderski, W. Czarnacki, M. Gierlik, W. Klamra and P. Shotanus, "Non-Proportionality and Energy Resolution of CsI(Tl)," *IEEE Transactions on Nuclear Science,* vol. 54, no. 5, pp. 1836-1841, 2007.

[19] Hamatsu, "Photomultiplier Tube Assembly H3178-51," [Online]. Available: https://www.hamamatsu.com/us/en/product/type/H3178-51/index.html. [Accessed 3 August 2019].

[20] On Semiconductor, "MICROFJ-SMTPA-60035-GEVB: MicroFJ-60035-TSV mounted on a pin adapter board," [Online]. Available: https://www.onsemi.com/support/evaluation-board/microfj-smtpa-60035-gevb. [Accessed 3 August 2019].

[21] E. b. R. Turchetta, Analog Electronics for Radiation Detection, Boca Raton: Taylo & Francis, CRC Press, 2016.

[22] Hamatsu, "How does temperature affect the performance of an SiPM?," [Online]. Available: https://hub.hamamatsu.com/us/en/technical-note/sipm-temperature-performance/index.html. [Accessed 15 August 2019].

[23] F. Corsi, M. Foresta, C. Marzocca, G. Matarrese and A. D. Guerra, "ASIC development for SiPM readout," *Journal of Instrumentation,* vol. 4, 2009.

[24] P. Peng, Y. Qiang, S. Ross and K. Burr, "Characterization of silicon photomultipliers and validation of the electrical model," *Nuclear Instruments and Methods in Physics Research A,* vol. 887, pp. 144-149, 2018.

[25] On Semi, "J-SERIES SIPM: Silicon Photomultiplier Sensors, J-Series (SiPM)," [Online]. Available: https://www.onsemi.com/products/sensors/silicon-photomultipliers-sipm/j-series-sipm. [Accessed 4 August 2019].

[26] Cremat, "CR-11X charge senstitive preamplifier modules," Cremat, [Online]. Available: https://www.cremat.com/home/charge-sensitive-preamplifiers/. [Accessed 14 August 2019].

[27] J. Pulko, F. R. Schneider, A. Velroyen, D. Renker and S. Ziegler, "A Monte-Carlo model of a SiPM coupled to a scintillating crystal," *Journal of Instrumentation,* vol. 7, 2012.

[28] A. Syntfeld-Kazuch, L. Swiderski, W. Czarnacki, M. Gierlik, W. Klamra, M. Moszynski and P. Schotanus, "Non-proportionality and Energy Resolution of CsI(Tl)," in *IEEE Nuclear Science Symposium Conference Record*, 2006.

# APPENDIX A

# Appendix A: Python Code

This is a copy of the Jupyter notebook used to generate the PWL current file for the time step analysis.

In [1]:
```
#Program dependencies
import matplotlib.pyplot as plt
import numpy as np
import math
import os.path
```

In [2]:
```
# ************** Constants Used ********************
num_of_tau_spad     = ###
num_fired           = ###


# ******************** SiPM Properties *************************
# Microcell Properties
r_discharge   = ###                                          # In ohms
r_quench      = ###                                          # In ohms
c_discharge   = ###                                          # In Farads (microcell
capacitance)
c_quench      = ###                                          # In Farags
v_overvoltage = ###                                          # In Volts

tau_discharge = r_discharge*(c_discharge + c_quench)*10**6      # In micro-seconds
q_discharge   = v_overvoltage*(c_discharge+c_quench)*num_fired  # In Coulombs
i_initial     = (q_discharge)/(tau_discharge*10**-6)            # In Amps (C/s)


# ********************* Time Steps (in microseconds) ***************
time_step = 0.000001
zero_Offset = 0.00503

# Number of Time Steps
microcell_time_steps = math.floor(tau_discharge*num_of_tau_spad/time_step) or 1
# microcell steps per time step
offset_time_steps = int(zero_Offset/time_step) or 1


# ******************** Print Values *********************************
#print (microcell_time_steps)
#print (Time_step)

#print (tau_discharge)
#print (tau_recharge)
```

In [3]:
```
# ************** SPAD Signals ********************
micro_array_size = microcell_time_steps + offset_time_steps

# Setting up Time Array - SPAD Discharge
micro_time_arr = np.zeros(micro_array_size)
for i in range(0, micro_array_size):
    micro_time_arr[i] = i*time_step
```

In [4]:
```
# Setting up [Current Signal] Array, Super imposing SPAD signal for each fired cell
current_signal_arr = np.zeros(micro_array_size)
for i in range (microcell_time_steps):
    j = offset_time_steps + i
    current_signal_arr[j] = i_initial*math.exp(-micro_time_arr[i]/tau_discharge)
current_signal_arr[-1] = 0
```

In [5]:
```
current_file = open('D:\\School\\Research\\Analog Circuit Analysis\\Current Files\\
current_file_100fs.txt', "w")
for i in range(micro_array_size-1):
    if (current_signal_arr[i] == current_signal_arr[i+1] and i > 0 and current_sign
al_arr[i] == current_signal_arr[i-1] ):
        continue
    else:
        current_file.write(str('{:10.8f}'.format(micro_time_arr[i])) + "u\t" + str
('{:7.4f}'.format(current_signal_arr[i])) +"\n")
current_file.write(str('{:10.8f}'.format(micro_time_arr[-1]+time_step)) + "u\t" + s
tr('{:7.4f}'.format(0)) +"\n")
current_file.close()
```

In [6]:
```
# SPAD Signal (Amp vs time)
plt.title("SPAD Signal")
plt.xlabel("nanoseconds")
plt.ylabel("Amps")
plt.plot(micro_time_arr, current_signal_arr)

#print (current_signal_arr[0:200])
```

# APPENDIX B

# Appendix B: Python Code: MC

This is a copy of the Jupyter notebook used to generate the PWL current file for the MC based model.

In [1]:
```
#Program dependencies
import matplotlib.pyplot as plt
import numpy as np
import math
import os.path
import random
#np.set_printoptions(threshold=np.inf)
```

In [21]:
```
# *************** Constants Used *********************
num_of_tau_scint    = 6
num_of_tau_spad     = 2.235
num_of_tau_recharge = 6


# ******************** SiPM Properties *************************
sipm_cells = int(22292)         #Total * fraction accessable to the Scintillation C
rystal
num_sipm = 4

# Microcell Properties
r_discharge   = ###                                     # In ohms
r_quench      = ###                                     # In ohms
c_discharge   = ###                                     # In Farads (microcell ca
pacitance)
c_quench      = ###                                     # In Farags
v_overvoltage = ###                                     # In Volts
cross_talk    = 0.104
afterpulse    = 0.013
dark_pulse_rate = 2.880                                    #1/(micro-sec)


tau_discharge = r_discharge*(c_discharge + c_quench)*10**6   # In micro-seconds
tau_recharge  = r_quench*(c_discharge + c_quench)*10**6      # In micro-seconds
q_discharge   = v_overvoltage*(c_discharge+c_quench)         # In Coulombs
i_initial     = v_overvoltage/r_discharge #(q_discharge)/(tau_discharge*10**-6)
# In Amps (C/s)


# ******************** Scintillation Material ******************
tau_rise = 0.02

#Single Tau
#tau_sc1 = 1
#tau_dif = tau_sc1 - tau_rise
#tau_timestep = tau_sc1 + tau_rise

# In Case of Two Tau
tau_sc1 = 0.6                                           # In microseconds
tau_sc1_fraction = 0.54
tau_sc2 = 3.4
tau_sc2_fraction = 0.46
tau_dif = tau_sc1*tau_sc1_fraction + tau_sc2*tau_sc2_fraction - tau_rise
tau_timestep = (tau_sc1 + tau_rise) if (tau_sc1 > tau_sc2) else (tau_sc2 + tau_ris
e)

# In Case of Three Tau
#tau_sc1 = 0.73                                          # In microseconds
#tau_sc1_fraction = 0.48
#tau_sc2 = 3.2
#tau_sc2_fraction = 0.30
#tau_sc3 = 16
#tau_sc3_fraction = 0.22
#tau_dif = tau_sc1*tau_sc1_fraction + tau_sc2*tau_sc2_fraction + tau_sc3*tau_sc3_fr
action - tau_rise
#tau_timestep = (tau_sc3 + tau_rise) if (tau_sc3 > tau_sc2) else (tau_sc2 + tau_ris
e)

photon_total = (65000*0.662/num_sipm)
photon_initial = photon_total / (tau_dif)

pde = 0.223                                             # For SiPM
```

In [ ]:
```
# ********************** Time Steps (in microseconds) ***************
time_step = 0.000001
zero_Offset = 0.0

# Number of Time Steps
microcell_time_steps = math.floor(tau_discharge*num_of_tau_spad/time_step) or 1
# microcell steps per time step
microcell_recharge_steps = math.floor(tau_recharge*num_of_tau_recharge/time_step) o
r 1  # microcell recharge steps per time step
n_time_steps = int((num_of_tau_scint*tau_timestep)/time_step) or 1
#time steps per 3 tau scintillator decay
offset_time_steps = int(zero_Offset/time_step) or 1



# ******************** Print Values *********************************
#print (microcell_time_steps)
#print (microcell_recharge_steps)
#print (n_time_steps)

#print(sipm_cells)

#print (tau_discharge)
#print (tau_recharge)
```

In [ ]:
```
# *************** SPAD Signals *********************

micro_array_size = microcell_time_steps
micro_recharge_size = microcell_recharge_steps


# Setting up Time Array - SPAD Discharge
micro_time_arr = np.zeros(micro_array_size)
for i in range(0, micro_array_size):
    micro_time_arr[i] = i*time_step

# Setting up SPAD Signal Array (single microcell firing)
spad_arr = np.zeros(micro_array_size)
for i in range (0, microcell_time_steps):
    spad_arr[i] = i_initial*math.exp(-micro_time_arr[i]/tau_discharge)


# Setting up Time Array - SPAD Recharge
micro_recharge_time_arr = np.zeros(micro_recharge_size)
for i in range(0, micro_recharge_size):
    micro_recharge_time_arr[i] = i*time_step

# Setting up SPAD Recharge Signal Array (Used to determine if microcell has fully r
echarged)
spad_recharge_arr = np.zeros(micro_recharge_size)
for i in range (0, micro_recharge_size):
    spad_recharge_arr[i] = math.exp(-micro_recharge_time_arr[i]/tau_recharge)
```

In [ ]:
```
# *************** Visualization ********************
plt.subplots(1,2, figsize=(10,4))
plt.subplots_adjust(wspace = 0.5, hspace = 0.8)

# SPAD Signal (Amp vs time)
plt.subplot(1,2,1)
plt.title("SPAD Signal")
plt.xlabel("nanoseconds")
plt.ylabel("mili-Amps")
plt.plot(micro_time_arr*1000, spad_arr*1000)

# SPAD Recharge
plt.subplot(1,2,2)
plt.title("Microcell Recharge")
plt.xlabel("nanoseconds")
plt.ylabel("% from Full Charge")
plt.plot(micro_recharge_time_arr*1000, spad_recharge_arr*100)
```

In [ ]:
```
# ************** Scintillator Signals **************

array_size = n_time_steps + offset_time_steps

# Setting up [Time] Array
time_arr = np.zeros(array_size)
for i in range(array_size):
    time_arr[i] = i*time_step

#Setting up [Photons] Array, Using N-Excited photons at each timestep,
#difference between two points = photons incident upon SiPM.
photon_arr = np.zeros(array_size)
total_counter = 0
for i in range (0, n_time_steps):
    j = offset_time_steps + i

    # Single Tau
    #point1 = tau_sc1*math.exp(-time_arr[i]/tau_sc1) - tau_rise*math.exp(-time_arr
[i]/tau_rise)
    #point2 = tau_sc1*math.exp(-time_arr[i+1]/tau_sc1) - tau_rise*math.exp(-time_ar
r[i+1]/tau_rise)
    # Multiple Tau
    point1 = tau_sc1*tau_sc1_fraction*math.exp(-time_arr[i]/tau_sc1) + tau_sc2*tau_
sc2_fraction*math.exp(-time_arr[i]/tau_sc2) - tau_rise*math.exp(-time_arr[i]/tau_ri
se)
    point2 = tau_sc1*tau_sc1_fraction*math.exp(-time_arr[i+1]/tau_sc1) + tau_sc2*ta
u_sc2_fraction*math.exp(-time_arr[i+1]/tau_sc2) - tau_rise*math.exp(-time_arr[i+1]/
tau_rise)

    photon_arr[j] = photon_initial*(point1-point2)              # Difference between
2 points for excited photons
    total_counter += photon_arr[j]                          # Use if interested in t
otal n-fired summed across time array



# Setting up [Photons quantized] Array, summing signals between timesteps into inte
gers.
photon_counter = 0
total_counter_2 = 0
photon_quantized_arr = np.zeros(array_size)
for i in range (n_time_steps):
    j = offset_time_steps + i
    photon_counter += photon_arr[j]
    if photon_counter > 1:
        photon_quantized_arr[j] = math.floor(photon_counter)
        total_counter_2 += math.floor(photon_counter)
        photon_counter -= math.floor(photon_counter)
```

In [ ]:
```
# *************** Visualisation ********************
plt.subplots(1,2, figsize=(10,8))
plt.subplots_adjust(wspace = 0.5, hspace = 0.8)

# Photons Per Time Step
plt.subplot(2,2,3)
plt.title("Photon Array")
plt.xlabel("microseconds")
plt.ylabel("Photons Per Time Step")
plt.plot(time_arr, photon_arr)

# Photon Quantized (either 1 or 0) (should look like a block)
plt.subplot(2,2,4)
plt.title("Photons (quantized)")
plt.xlabel("microseconds")
plt.ylabel("Photons")
plt.plot(time_arr, photon_quantized_arr)



# ******************** Print Values *********************************
#print(photon_total)
#print(total_counter)
#print(total_counter_2)
```

In [ ]:
```
# ************** Fired Cells **************

# N_Fired Estimate from SENSL (Overestimated for Long exposures (such as scintillat
ors))
n_fired_cell_tot = int(sipm_cells * (1 - math.exp(-(photon_total*pde) / sipm_cell
s)))     # Over-Estimate


# Setting up [Microcell Fired, Time Fired] Array
random.seed()
fired_arr = np.zeros((total_counter_2, 2))
fired_arr_counter = 0

for i in range(array_size):
    if photon_quantized_arr[i] > 0:
        if (random.randint(1,1000) < int((pde)*1000)):
            fired_arr[fired_arr_counter, 0] = random.randint(1,sipm_cells)
            fired_arr[fired_arr_counter, 1] = i
            fired_arr_counter += 1
            if (random.randint(1,1000) < int((cross_talk)*1000)):
                fired_arr[fired_arr_counter, 0] = random.randint(1,sipm_cells)
                fired_arr[fired_arr_counter, 1] = i
                fired_arr_counter += 1
                #print("cross")
            if (random.randint(1,1000) < int((afterpulse)*1000)):
                fired_arr[fired_arr_counter, 0] = random.randint(1,sipm_cells)
                fired_arr[fired_arr_counter, 1] = i
                fired_arr_counter += 1
                #print("after")


fired_arr.resize(fired_arr_counter, 2)              # Shrink array to area actually
used.
fired_arr.view('u8,u8').sort(order=['f0'], axis=0)  # Sort array by SiPM Cell fire
d.
```

In [ ]:
```
duplicates_arr = np.zeros((fired_arr_counter, 5))
duplicates_arr_counter = 0
for i in range(fired_arr_counter-1):
    if (fired_arr[i,0] == fired_arr[i+1,0] and (fired_arr[i+1,1]-fired_arr[i,1]) <
microcell_recharge_steps):
        duplicates_arr[duplicates_arr_counter,0] = fired_arr[i,0]
        duplicates_arr[duplicates_arr_counter,1] = fired_arr[i,1]
        duplicates_arr[duplicates_arr_counter,2] = fired_arr[i+1,1]
        duplicates_arr[duplicates_arr_counter,3] = fired_arr[i+1,1] - fired_arr[i,
1]
        duplicates_arr[duplicates_arr_counter,4] = 1 - spad_recharge_arr[int(fired_
arr[i+1,1] - fired_arr[i,1])]

        duplicates_arr_counter += 1
        fired_arr[i,0] = 0
        fired_arr[i,1] = 0
        fired_arr[i+1,0] = 0
        fired_arr[i+1,1] = 0

duplicates_arr.resize(duplicates_arr_counter, 5)


# ******************** Print Values *********************************
#np.set_printoptions(threshold=np.inf)
print(fired_arr_counter)
print(total_counter_2*pde)
print(n_fired_cell_tot)
#print(fired_arr)
#print(fired_arr[:,1])
print(duplicates_arr_counter)
#print(duplicates_arr)
```

In [ ]:
```
# ************** Current Signal **************

# Setting up [Current Signal] Array, Super imposing SPAD signal for each fired cell
current_signal_arr = np.zeros(array_size)

# Adding Normal Fired Cells
for i in range (fired_arr_counter):
    if fired_arr[i,0] > 0:
        for j in range (micro_array_size):
            current_signal_arr[int(fired_arr[i,1]+j)] += spad_arr[j]

# Adding Cells fired prior to recharge
for i in range (duplicates_arr_counter):
    #First of the Duplicate
    if (duplicates_arr[i,3] > micro_array_size):
        for j in range (micro_array_size):
            current_signal_arr[int(duplicates_arr[i,1]+j)] += spad_arr[j]
    else:
        for j in range (duplicates_arr[i,3]):
            current_signal_arr[int(duplicates_arr[i,1]+j)] += spad_arr[j]
    #Second of the Duplicate
    for k in range (micro_array_size):
        current_signal_arr[int(duplicates_arr[i,2])+k] += spad_arr[k]*duplicates_ar
r[i,4]

plt.title("Current Signal")
plt.xlabel("micro-seconds")
plt.ylabel("mili-Amps")
plt.plot(time_arr, current_signal_arr*1000)
```

In [ ]:
```
current_file = open('D:\\School\\Research\\Analog Circuit Analysis\\Current Files\\
current_file-knoll-MC-1ps(1Aug-3).txt', "w")
for i in range(array_size-1):
    if (current_signal_arr[i] == current_signal_arr[i+1] and i > 0 and current_sign
al_arr[i] == current_signal_arr[i-1] ):
        continue
    else:
        current_file.write(str('{:10.8f}'.format(time_arr[i])) + "u\t" + str('{:7.4
f}'.format(current_signal_arr[i]*1000)) +"m\n")
current_file.write(str('{:10.8f}'.format(time_arr[-1]+time_step)) + "u\t" + str('{:
7.4f}'.format(0)) +"m\n")
current_file.close()
```

In [ ]:
```
plt.subplots(2,2, figsize=(15,5))
plt.subplots_adjust(wspace = 2.0, hspace = 1.0)

# SPAD Signal (Amp vs time)
plt.subplot(2,2,1)
plt.title("SPAD Signal")
plt.xlabel("nanoseconds")
plt.ylabel("Amps")
plt.plot(micro_time_arr*1000, spad_arr)

# Photons Per Time Step
plt.subplot(2,2,2)
plt.title("Photon Array")
plt.xlabel("microseconds")
plt.ylabel("Photons Per Time Step")
plt.plot(time_arr, photon_arr)

# Photon Quantized (either 1 or 0) (should look like a block)
plt.subplot(2,2,3)
plt.title("Photons (quantized)")
plt.xlabel("microseconds")
plt.ylabel("Photons")
plt.plot(time_arr, photon_quantized_arr)


# Current (Amp)
plt.subplot(2,2,4)
plt.title("Current Signal")
plt.xlabel("micro-seconds")
plt.ylabel("mili-Amps")
plt.plot(time_arr, current_signal_arr*1000)

plt.savefig('C:\\Users\\McGee\\Desktop\\Current-MC-knoll-100fs.pdf')

"""
# Current Signal
plt.subplot(2,2,4)
plt.title("Current Signal")
plt.xlabel("micro-seconds")
plt.ylabel("mili-amps")
plt.plot(time_arr, current_signal_arr)
"""
```