DESIGN AND CONSTRUCTION OF A TIME-RESOLVED OPTICALLY STIMULATED LUMINESCENCE MEASUREMENT SYSTEM

A THESIS SUBMITTED TO
THE GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES
OF
MIDDLE EAST TECHNICAL UNIVERSITY

BY

DENİZ ORHUN BOZTEMUR

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR
THE DEGREE OF MASTER OF SCIENCE
IN
PHYSICS

AUGUST 2024

Approval of the thesis:

DESIGN AND CONSTRUCTION OF A TIME-RESOLVED OPTICALLY STIMULATED LUMINESCENCE MEASUREMENT SYSTEM

submitted by **DENİZ ORHUN BOZTEMUR** in partial fulfillment of the requirements for the degree of **Master of Science in Physics, Middle East Technical University** by,

Prof. Dr. Naci Emre Altun
Dean, **Graduate School of Natural and Applied Sciences**

Prof. Dr. Seçkin Kürkçüoğlu
Head of the Department, **Physics**

Prof. Dr. Enver Bulur
Supervisor, **Physics Dept., METU**

Examining Committee Members:

Prof. Dr. Mehmet Parlak
Physics Dept., METU

Prof. Dr. Enver Bulur
Physics Dept., METU

Assoc. Prof. Dr. Burak Yedierler
Physics Dept., METU

Assoc. Prof. Dr. Ali Kiracı
Inter-Curricular Courses Dept., Çankaya University

Asst. Prof. Dr. Ihor Pavlov
Physics Dept., METU

Date: 27.08.2024

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name Last Name: Deniz Orhun Boztemur

Signature :

ABSTRACT

DESIGN AND CONSTRUCTION OF A TIME-RESOLVED OPTICALLY STIMULATED LUMINESCENCE MEASUREMENT SYSTEM

Boztemur, Deniz Orhun
Master of Science, Physics
Supervisor: Prof. Dr. Enver Bulur

August 2024, 221 pages

Pulsed Optically Stimulated Luminescence (POSL) has become one of the dominant measurement modalities in recent decades for its ability to measure the absorbed radiation dose in wide bandgap solids. Time-Resolved OSL is a variant of POSL in which the luminescence decay of an irradiated sample, after a stimulation pulse, is recorded and analyzed and can be used for the separation of luminescence signals from recombination centers with different decay constants. In multi-mineral samples, analysis can help discriminate phases using decay data. In some insulators, different radiation types may induce distinct types of defects with dissimilar decays; in such a case, discrimination of the radiation type becomes feasible. In radiation dosimetry, conventionally microcrystalline materials have been employed; synthesis of dosimeter materials based on nanoparticles has proven that these materials offer high sensitivity and reproducibility that may overtake conventional materials. Synthesis and usage of new materials require an understanding of the electron storage and luminescence recombination centers. Combined with spectral information, TR-OSL gives valuable information about the recombination processes. This thesis will cover the process of designing, constructing, and testing an open-

sourced TR-OSL measurement system for the study of luminescence mechanisms in nanoparticle and microcrystalline materials. The device will be based on two stimulation wavelengths (λ_p~470 and λ_p~850 nm) and could be used for the analysis of luminescence decays with lifetimes of around 100 ns to seconds. Using the measurement system materials including Al_2O_3:C and BeO and similar ones will be investigated.

Keywords: Micro and Nanotechnology, Condensed Matter Physics, Radiation Physics, Scientific Instrumentation, Optically Stimulated Luminescence

ÖZ

ZAMAN TANIMLI OPTİK UYARMALI LÜMİNESANS ÖLÇÜM SİSTEMİNİN TASARIMI VE YAPIMI

Boztemur, Deniz Orhun
Yüksek Lisans, Fizik
Tez Yöneticisi: Prof. Dr. Enver Bulur

Ağustos 2024, 221 sayfa

Atımlı Optik Uyarmalı Lüminesans (POSL), soğurulan radyasyon dozunu ölçme yeteneği (ve kuvars ve feldspat içeren örneklerin yaşını belirleme) nedeniyle son yıllarda önde gelen ölçüm yöntemlerinden biri haline gelmiştir. Atımlı OSL tekniğinin bir alt dalı olan Zaman Tanımlı OSL (TR-OSL), kısa bir uyarım darbesinden sonra önceden radyasyona maruz kalmış bir örneğin lüminesans sönüm eğrisinin kaydedilip analiz edildiği ve farklı sönüm sabitlerine sahip rekombinasyon merkezlerinden gelen lüminesans sinyallerinin ayrılmasında kullanılabilir. Bu nedenle, çok mineralli örnekler analiz edilirken lüminesans sönüm verilerini kullanarak çeşitli fazları ayırt etmeye yardımcı olabilir. Ayrıca, farklı radyasyon türleri, bazı yalıtkanlarda farklı lüminesans sönümlerinden kaynaklanan farklı türde kusurlar oluşturabilir; böyle durumlarda radyasyon türünün ayrımı mümkün hale gelebilir. Radyasyon dozimetrisi alanında, geleneksel olarak mikrokristalli (toz veya seramik formunda) malzemeler kullanılmıştır. Son birkaç on yılda nanopartikül bazlı dozimetrik malzemelerin sentezi, bu malzemelerin yüksek hassasiyet ve tekrarlanabilirlik sunduğunu ve geleneksel malzemelerle rekabet edebileceğini ve hatta onları geçebileceğini kanıtlamıştır. Yeni malzemelerin sentezi ve kullanımı, elektron depolama ve lüminesans rekombinasyon merkezlerinin anlaşılmasını

gerektirir. Spektral veri ile birleştirildiğinde, TR-OSL, lüminesans rekombinasyon süreçleri hakkında değerli bilgiler sunar. Bu tez, nanopartikül bazlı ve mikrokristalli malzemelerde lüminesans üretim mekanizmalarının incelenmesi için açık kaynaklı bir TR-OSL ölçüm sisteminin tasarım, üretim ve test edilme sürecini kapsayacaktır. Cihaz, iki yaygın uyarım dalga boyuna (λ_p~470 nm ve λ_p~850 nm) dayanacak ve ömürleri yaklaşık 100 ns ile saniyeler arasında olan lüminesans sönümlerinin analizinde kullanılabilecektir. Geliştirilen ölçüm sistemi kullanılarak kuvars, feldspatlar, karbon katkılı alümina, BeO ve benzeri birçok OSL malzemesi incelenecektir.

Anahtar Kelimeler: Mikro ve Nanoteknoloji, Katı Hâl Fiziği, Radyasyon Fiziği, Bilimsel Enstrümantasyon, Optik Uyarmalı Lüminesans

to a lifetime of learning

and to those who believe they will survive beyond Chapter

ACKNOWLEDGMENTS

I would like to thank my parents, Prof. Dr. Recep Boztemur and Muteber Boztemur for their unending, unwavering support in all aspects of life and, more important than all, nurturing my love for learning.

I would also like to express my gratitude to my supervisor, Prof. Dr. Enver Bulur, for his guidance, advice, insight and belief.

I am also grateful for all my friends and loved ones, who never stopped believing in me and kept their trust constant.

Finally, I would like to appreciate all the cups of coffee I consumed whilst doing this research.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS

OSL: Optically Stimulated Luminescence

CW-OSL: Continuous Wave Optically Stimulated Luminescence

LM-OSL: Linearly Modulated Optically Stimulated Luminescence

POSL: Pulsed Optically Stimulated Luminescence

TR-OSL: Time-Resolved Optically Stimulated Luminescence

TL: Thermoluminescence

TL/OSL: Thermoluminescence/Optically Stimulated Luminescence

TLD: Thermoluminescent Dosimeter

OSLD: Optically Stimulated Luminescent Dosimeters

TOSL: Thermally and Optically Stimulated Luminescence

LED: Light Emitting Diode

PMT: Photomultiplier Tube

DAQ: Data Acquisition Unit

IDE: Integrated Development Environment

OSF: Open Science Foundation

VFD: Vacuum Fluorescent Display

PCB: Printed Circuit Board

λ_p: Peak Wavelength

NASA: National Aeronautics and Space Administration

ESA: European Space Agency

CHAPTER 1

INTRODUCTION

As a scientific term, “Luminescence” was introduced to the scientific community by Eilhard Wiedemann in 1888 to distinguish between “cold light” from radioactivity and “hot light” from incandescent sources. A new term was required to save the second law of thermodynamics, where heat cannot flow from a colder body to a hotter body. It explains the physical phenomenon where a relatively cold material emits photons. When an atom’s (or molecule’s) electrons are excited to an energy state, these electrons attempt to return to their ground states with a process called transitions, either non-radiative or radiative. In the former type, a photon is not emitted as the result, but in the latter, light is emitted. Types of luminescence will be explained in Chapter 2, Luminescence; however, it is essential to follow a chronological timeline for optically stimulated luminescence studies first. Optically Stimulated Luminescence is a process where are material, previously exposed to ionizing radiation, is subjected to a stimulation light signal, and in turn, emits a light signal proportional to radiation dose. In 1956, Antonov-Romanovskii et al. (1956) proposed the idea of radiation dose measurement using OSL, but the suggested method was not used for nearly a decade. It was Bräunlich et al. (1967) and, Sanborn and Beard (1967) who studied this method near-simultaneously. However, the materials used, with phosphorescent characteristics, were not appropriate for dosimetry. Strontium Selenide, Magnesium Sulfide, and Cadmium Sulfide doped with Cerium, Samarium, Europium, and other rare earth elements, however classified as phosphors, did not work well as dosimeters. However, materials like BaFBr: Eu^{+2} have been used in X-ray detection or autoradiography in the 1980’s, despite their inadequacy for dosimetry (Kenji, 1989). Huntley et al. (1985) were the first to use an argon-ion laser to stimulate natural materials such as quartz and feldspar to determine the age of archeological artifacts. Widespread use and ease of access to stimulation sources have allowed OSL to be studied widely and enabled it to be used in dosimetry applications. In this area, Markey et al. (1995) first used Al_2O_3:C as dosimetry, taking advantage of this material's enhanced luminescent properties. A great and detailed review of these studies and OSL in general including applications and physical mechanisms has been published by Bøtter-Jensen et al. (2003), Yukihara and McKeever (2011), Chithambo (2018), McKeever, (2019), Chen and Pagonis, (2019), Yuan et al. (2020), Nando et al, (2022), Bajaj (2022) and Yukihara (2023).

Currently, Al_2O_3:C and BeO OSL dosimeters are the only two materials within the OSL dosimetry industry, that have been studied by Akselrod et al. (1996), Markey et al. (1995), McKeever (1996b), Bulur and Göksu. (1998a, b), Sommer et al (2006), Jahn et al. (2013), and recently reviewed by Yukihara (2014). Widespread use and application flexibility of these dosimeters have allowed millions of radiation workers to be safely monitored by passive devices. Medical professionals have also made use of these dosimeters (Lovelock et al., 2012) as well as Space Agencies such as NASA (Zhou et al., 2009) and ESA (Berger et al., 2016). Usage of Al_2O_3:C and BeO in dosimetry is steadily increasing, yet there is a need for new types of luminescent materials, which is where nanoparticles might make an appearance. Nanomaterials are preferred in luminescence applications for their optical properties and tunability of size and/or wavelength of doping material. The small size of particles leads to quantum confinement effects that enhance the efficiency of luminescence and allow precise control over trap depths, in other words, emission wavelengths. The versatile nature of these nanoparticles, along with their excellent performance, make them an ideal choice for luminescent experiments and technologies. In recent years, nanoparticles in OSL have been used extensively, $Li_2B_4O_7$:Ag, Cu (Hemam et al., 2016), Al_2O_3 (Bossin et al., 2018), CaB_4O_7:Dy^{3+} (Hemam et al., 2022), MgB_4O_7:Ce, Li (Bossin, 2023) just to name a few. Along with these, Altunal et al. have been developing and testing OSL-based dosimeters using BeO: Na, Dy, Er (Altunal et al.,2020), along with determining luminescence properties of SrB_6O: Ce, Gd (Altunal et al., 2024b) and $CaAl_{12}O_{19}$:Ce, Be (Altunal et al., 2024c) for use in dosimetry, luminescence characterization of LiAlO (Altunal et al., 2024a) and OSL properties of alkali and rare earth doped BeO based novel dosimeters (Altunal et al., 2023). An increase in nanoparticle use in OSL systems will enable more precise and efficient solutions for dosimetry applications.

Measurement of the maximum number of items in the minimum amount of time without sacrificing efficiency and accuracy is a key idea for any instrumentation system. To achieve this, for dosimetry and dating studies, automated OSL readers have been designed and constructed, and are in widespread use (see e.g., Yukihara and McKeever, 2011a) and are both commercially available, industry-grade and homemade, open-sourced. Many scientists and engineers have worked on these systems in different configurations, often employing a sample changer, a measurement unit, and a personal computer to go with it.

The first commercial automated reader was the Elsec 9010 OSL by Littlemore Scientific (Spooner et al., 1990). Along with this, Risø TL/OSL reader (Bøtter-Jensen, 1988 (TL only), The Risø TL/OSL Reader, 2024), Daybreak TL/OSL readers (Daybreak, 2008) and the lexsyg series of combined TL/OSL readers (Risø TL/OSL Reader Vs. Lexsyg Smart & Lexsyg Research, 2022), along with LANDAUER (Landauer Optically Stimulated Luminescence (OSL), 2023) and Nucleonix (Nucleonix Systems, 2023) OSL readers have been used, where the latter two are mainly commercialized and made for dosimetry applications only. However, these listed products are end-products, therefore are not available for a scientist to fully modify and study. Even though these off-the-shelf products are ideal for use in the industry and are commonly used in scientific research, the need for different measurement modes for novel experiments is still dominant. This scientific equipment is highly specialized and thus is expensive and hard to obtain. To conduct experiments and research, open-sourced measurement systems have made a prominent entry into the scientific domain. Previously, Kulkarni (2008) designed an OSL system to measure dosimeters, which included all CW-OSL, LM-OSL and

POSL. Choi et al. (2014) designed a compact and cheap OSL reader with a 12-sample tray, an X-ray source to irradiate and 2 LEDs to illuminate samples. In the same year, Guérin et al. (2014) introduced a device that can measure both thermoluminescence and optically stimulated luminescence. A year later, Kearfott (2015) designed a low-cost OSL reader able to use several illumination wavelengths. Lapp et al (2015) realized a new head for the Risø TL/OSL reader, improving the precision and sensitivity, as well as laying the foundation of EMCCD (electron-multiplying charge-coupled device) usage in OSL readers. Maraba (2017) has also reported a low-cost OSL system with a blue/green LED for illumination. In recent years, Mittani et al (2019) devised a TL/OSL reader system using commercial components using a 532nm LED for illumination source for further dosimetry research. Kim et al. (2024) prototyped a TL/OSL reader to be used in a possible CBRNE (Chemical, Biological, Radiological, Nuclear, and Explosives) disaster, which could be used on-site, wherever needed.

As in any open-sourced instrumentation system, both hardware and software should be accessible by the general public. This adds an extra degree of freedom above the commercially available OSL systems. Using open-sourced hardware and software also decreases the cost of scientific research, making it possible for researchers everywhere to be able to further science. The most common open-sourced software framework currently is Python, which is free to use, available for improvement, and has a very large community for support. It is widely used in programming, especially wherever high-level software is needed (Rossum, 1995). An open-sourced software is, of course, available to be used in any research field, but this framework is also widely employed within Physics for research (Imreh, 2014, Koenka et al, 2014, Sang, 2021, Leitão, 2024).

In any contemporary instrumentation and measurement system, a controller is essential. In the past few decades, microcontrollers have made an entrance to scientific research, by being accessible to everyone, easy to use, affordable, expendable, and mostly open-sourced. These open-sourced versions are known as free and open-source hardware and are both publicly and widely available (Pearce, 2014). In the market, the most prominent microcontroller carrying hardware is currently the Arduino^{TM}, which is both a hardware and software framework for use and has adopted an important role in replacing expensive scientific instrumentation controllers (Fisher et al., 2012).

This thesis aims to design, construct, and implement a low-cost, nearly completely open-sourced TR-OSL measurement system. The expected outcome can be considered as a usable proof of concept design and is expected to serve as a modifiable research basis for various needs. It will facilitate an easily changeable stimulation source and detection filters, an 8-sample tray for multi-sample studies. For further research, this device will be used to run different experiments, to determine the emission lifetimes of various kinds of samples relevant for dosimetry and dating, and hopefully further knowledge on luminescence production mechanisms in dosimeter materials.

In Chapter 2, the basics of luminescence will be explained. Types of luminescence, fluorescence, and phosphorescence will be discussed. Following this, the luminescence of nanophosphors will be mentioned.

In Chapter 3, the basic principles of OSL will be discussed. Types of OSL, Continuous Wave OSL, Linearly Modulated OSL, Pulsed OSL and finally Time-Resolved OSL will be explained, along with the importance of luminescence in OSL and nanophosphors and their use in dosimetry.

In Chapter 4, the basics of hardware and software in scientific instrumentation will be mentioned. The specifics of an OSL instrumentation system will be written and its parts will be explained briefly. Open-source devices, both hardware and software will also be mentioned.

In Chapter 5, the design considerations of the TR-OSL measurement system will be explained in detail. Hardware and software used will be clarified and their reasons to be chosen will be shared. The methods of how these have been used will be shown and expressed elaborately. The hardware and the associated software will be explained.

In Chapter 6, the results of the test experiments carried out with the constructed device will be mentioned. Starting from the characteristics of the illumination system, and the response of the system to test measurements, the chapter will then progress into the OSL measurement results of experiments performed using Al2O3:C, BeO, quartz, feldspars, and similar materials relevant to dosimetry and dating.

In Chapter 7, a brief discussion of the results and the conclusion and finalization of the research will be shared with the reader.

CHAPTER 2

LUMINESCENCE

Luminescence is the physical mechanism where a material emits light without external thermal energy applied. Within the luminescence concept, the emission of a photon is a result of several different phenomena, which will be listed in the table below, where energy from the excitation is absorbed by electrons. These energetic electrons move to a higher energy state. When they return to their stable, lower energy position, the energy difference is then released as photons. The main difference between luminescence and incandescence is heat, whereas in incandescence, heat is applied to a substance until the material “glows” (Lakowicz, 2006).

The main concept behind luminescence includes excitation and relaxation of electrons. When energy is given to a luminescent substance, electrons are excited from their ground states to an excited state. This energy can be provided by light, chemical reactions, electricity, mechanical energy, or heat (however, unlike incandescence, the primary light source is not the applied heat, but rather the release of electrons). The excited state of an electron is not stable; thus, the electron will try to return to its relaxed, ground state by losing energy. This energy loss from the excited state to the ground state releases the stored energy in the form of photons (see, for example, Valeur, 2002). A simple diagram representing the excitation and relaxation mechanisms is shown in Figure 2.1.

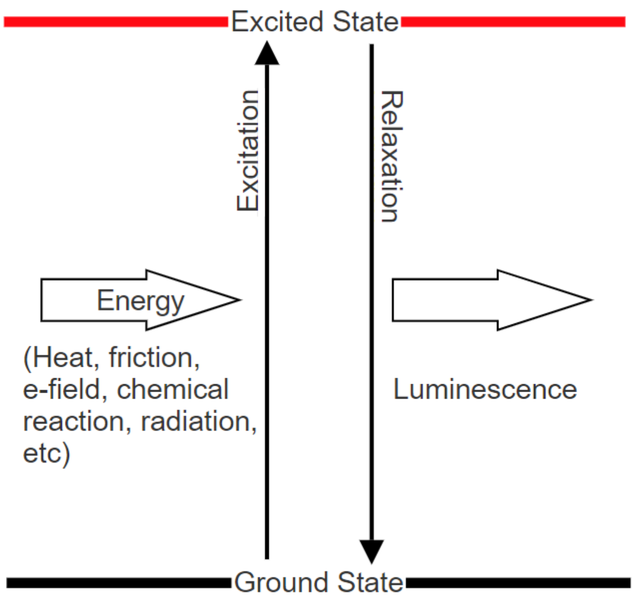

Figure 2.1. Representation of the Luminescence Mechanism between two energy levels

Wavelength, thus the color of the emitted photons depends on the energy gap between the two states. Different substances have different energy gaps, so the emission of various colors is possible. This emission can be delayed emission or instantaneous, depending on the phenomena behind the electron transition. As an example, fluorescence has an instantaneous emission, but phosphorescence can have a delayed emission due to electrons being trapped in meta-stable states (Ronda, 2007). There are many different types of luminescence. These types and the reasons for electron excitations are listed below in Table 2.1 (Rodriguez et al, 2007).

Having an understanding of luminescence mechanisms is important for both research and potential applications in new technology. These concepts are extremely common, for example, photoluminescence is commonly used in white LEDs (however the principle illumination from LED is low-field electroluminescence) and fluorescent lamps, and chemiluminescence is used in forensic science (to find blood stains using flashlights), bioluminescence is used as a genetic marker in biological studies, electroluminescence is the driving mechanism behind displays (basically LEDs), triboluminescence is used in material characterization, especially, stress analysis. Thermoluminescence is widely used in both radiation dosimetry and archeological dating studies (Rodriguez et al, 2007).

Table 2.1 Types of Luminescence and Their Respective Excitation Sources

Photoluminescence Optically Stimulated Luminescence	Photons, light (After radiation)
Chemiluminescence Bioluminescence	Chemical reactions
Electroluminescence Cathodoluminescence	Electrons
Thermoluminescence	Heat (After radiation)
Radioluminescence	Radioactivity
Rontgenoluminescence	X-rays
Mechanoluminesence Triboluminescence	Mechanical effects
Ionoluminescence	Ions
Sonoluminescence	Sound
Crystalloluminescence	Crystallization

2.1 Fluorescence

Fluorescence is characterized by a material absorbing light of a specific wavelength and emitting a photon with a longer wavelength. This mechanism starts with a photon being absorbed by an electron, jumping to an excited state. This electron cannot occupy the excited state for a long period of time, deexciting in only a few nanoseconds. Deexcitation releases photons, which is called fluorescence (Lakowicz, 2006).

The efficiency of fluorescence is explained in two parameters, quantum yield and lifetime. Quantum yield is the ratio of the emitted number of photons to the absorbed number of photons. Lifetime is the average time an excited electron stays excited before relaxation and emission of a photon. These parameters result from the molecular structure and the solvent parity of the material under testing as well as the ambient temperature and the existence of quenching agents (Valeur, 2002).

Fluorescence is used widely across many scientific fields, mainly for its sensitivity. In biology and medicinal science, it’s commonly used in fluorescence microscopy, fluorescence resonance energy transfer (FRET), and flow cytometry. In forensic science, it’s used to detect biological fluids, in chemistry, it’s used to analyze molecular interactions. Also, in environmental science, it is mainly used in the form of a dye, and is used to trace flowing water and pollution detection. These many and diverse uses make fluorescence an invaluable tool in modern science, for both qualitative and quantitative analysis (Rost, 1992).

2.2 Phosphorescence

In phosphorescence, the electrons relax into a meta-stable state known as a triplet state, as opposed to fluorescence, where the excited states are singlet. Electrons can occupy this state longer than their excited state, thus delaying the emission of light. This delay ranges from microseconds (10^{-6} s) to days (more than 10^{5} s) (Lakowicz, 2006). A representation of the singlet and triplet states is shown in Figure 2.2 (Mobley, 2021).

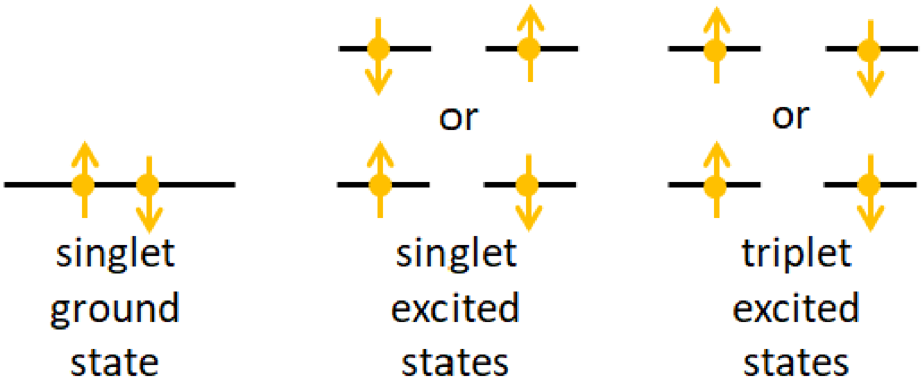

Figure 2.2. Illustration of singlet and triplet states. Arrows represent up and down angular momentum states.

This delayed emission is due to the "forbidden" status of a triplet state to singlet state transition, in accordance with the Pauli Exclusion Principle. In the scope of quantum mechanics, states with different spin multiplicities have a much lower probability of having a transition occur between them, but if it does, the transition is characterized to be slow. Generally, the excited state lifetime is inversely proportional to the probability of the transition back to the ground state. Since a forbidden transition probability is low, electrons stay in their excited state longer, in the order of 10^{-4} to 10's of seconds or even more, and finally emit a photon to return to their stable state. Emitted photon generally has a longer wavelength than its absorbed counterpart due to this energy loss (Valeur, 2002).

A simplified Jablonski diagram indicating the states and the transition between states resulting in phosphorescence and fluorescence can be found in Figure 2.3. (Reproduced from Schweizer et al, 2021).

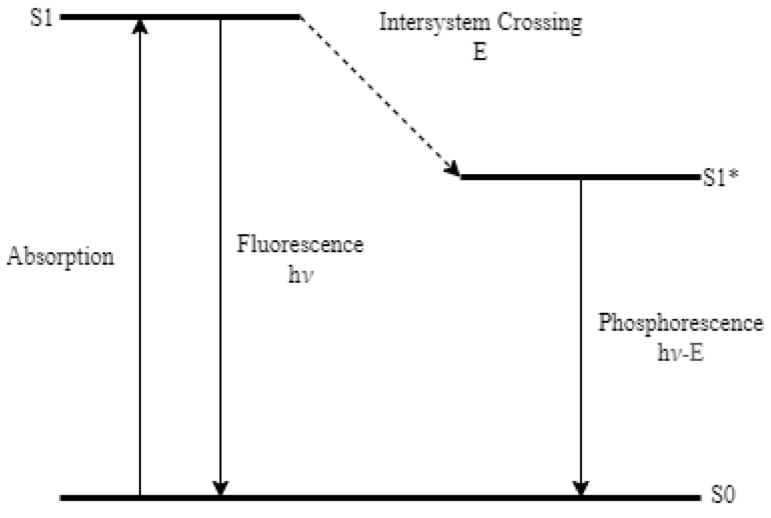

Figure 2.3. Jablonski diagram displaying the fluorescent and phosphorescent transitions with the forbidden process

Phosphorescent materials are generally used with their slow emissions exploited, such as glow-in-the-dark products like safety signs, wristwatches, and novelty items, where the item absorbs energy from light and emits it during the dark. However, even while the difference between phosphorescence and long-afterglow is slim, it should be noted that they are different concepts.

Figure 2.4. Long Afterglow Emission or Room Temperature Phosphorescent RPG dice

As an addition, phosphorescent materials are used in specialized imaging technologies, where prolonged luminescence is useful. Their ability to emit light without continued exposure makes them useful in areas requiring non-stop visibility (Ronda, 2007). However, for the scope of this research, millisecond-fast processes like Cr^{+3} in ruby lasers or Mn^{+2} and Eu^{+3} that are used in thermometry are more relevant (Luchechko et al., 2015, Malíčková et al., 2018).

2.3 Differences Between Phosphorescence and Fluorescence

In recombination, the luminescence phenomenon can be studied in two main topics, fluorescence and phosphorescence. These topics are split due to the difference in allowed electron spin states in the de-excitation of an electron. In fluorescence, a ground-state electron is excited with its spin-state conserved, which means there will be two electrons with opposite spin states. This excitation is an allowed energy transition. In phosphorescence, one of the two same-spin electrons has its spin direction reversed, thus causing two electrons with the same spin to be excited in the same state at once. This is called a forbidden process. In phosphorescence, the time between excitation and emission of a photon is longer than in fluorescence (Chen & McKeever, 1997) caused by the "forbidden" transition requiring the low probability of triplet to singlet state transition, thus lengthening the afterglow (Lakowicz, 2006). In summary, the most important difference between phosphorescence and fluorescence is the mechanism and duration of their emission process. Both employ the absorption and emission of photons, but fluorescence happens instantaneously but phosphorescence happens much slower.

2.4 Luminescence in Nanophosphors

Nanophosphors are luminescent materials made up of nanoparticles, exhibiting exceptional optical properties. These nanomaterials are typically made up of a host doped with luminescent ions (however not necessarily). Their size, typically from 1 to 100 nanometers, provides unique advantages over conventional phosphorescent materials, such as brightness, stability, wavelength tunability, and manufacturing/application ease (Chen et al, 2014).

The luminescence mechanism remains the same as normal materials, but since nanophosphors can be fine-tuned by changing the size of the dopant or host, their emission spectrum can be altered, changing the wavelength and, thus the energy of emitted light. The nanoscale also increases the surface-to-volume ratio, which increases the luminescent efficiency and brightness of nanomaterials with respect to the bulk material. (Li et al, 2010)

One of the possible application areas of nanomaterials or nanophosphors can be the field of ionizing radiation detection and dosimetry. Nanophosphors are commonly used in TLDs (Thermoluminescent Dosimeters) due to their high sensitivity and precision. These materials store energy when exposed to ionizing radiation and release it in the form of photons when subjected to heat. The intensity of emitted luminescence is directly correlated to the amount of radiation absorbed, allowing for very accurate dose measurement (Nagy et al, 2013).

Other important applications of nanophosphors include LEDs, where they're utilized in producing white light. Blue or near-ultraviolet LEDs are combined with red, green, and blue light-emitting nanophosphors. This technique allows for the production of a wide LED color spectrum, thus widening the LED application range. Also, nanophosphors are used in flexible displays and various optoelectronic devices for their ability to be in thin films (Wang et al, 2011).

As the main scope of this research is optically stimulated luminescence, this chapter can be considered a preliminary text on a more general approach to luminescence to inform the reader of the underlying principles. The following chapter will explain OSL and its techniques in depth.

CHAPTER 3

BASIC CONCEPTS OF OSL

To understand the basic concept of Optically Stimulated Luminescence (OSL), one first needs to have basic knowledge of the properties of materials. The band gap in semiconductors and insulators is the energy difference between the valence energy band and the conduction energy band. In insulators, this band gap is generally large, normally in the order of 10 eV, where the electrons require high energy to make a transition into the conduction band from the valence band. The valence band in insulators is the energy level where electrons are fixed and are not free. In an insulator, the valence band is fully occupied by electrons in absolute zero temperatures (or relatively low temperatures), which makes conduction extremely difficult. The conduction band, however, is the range of energy levels above the valence band. Electrons in this band are generally free and contribute to electrical conductivity. In insulators, however, this band is generally empty and excitation of an electron to the conduction band requires high energy. The energy area, or gap, between the valence and conduction bands can be called the “forbidden energy gap”. In any material, defects may exist in the gap, sometimes caused by impurities of the material, sometimes originating from the native lattice defects. These defects can capture electrons or holes, creating trapping states (generally named as traps). A general image of the band diagram can be found in Figure 3.1, which was reproduced from (Ciftja, 2013).

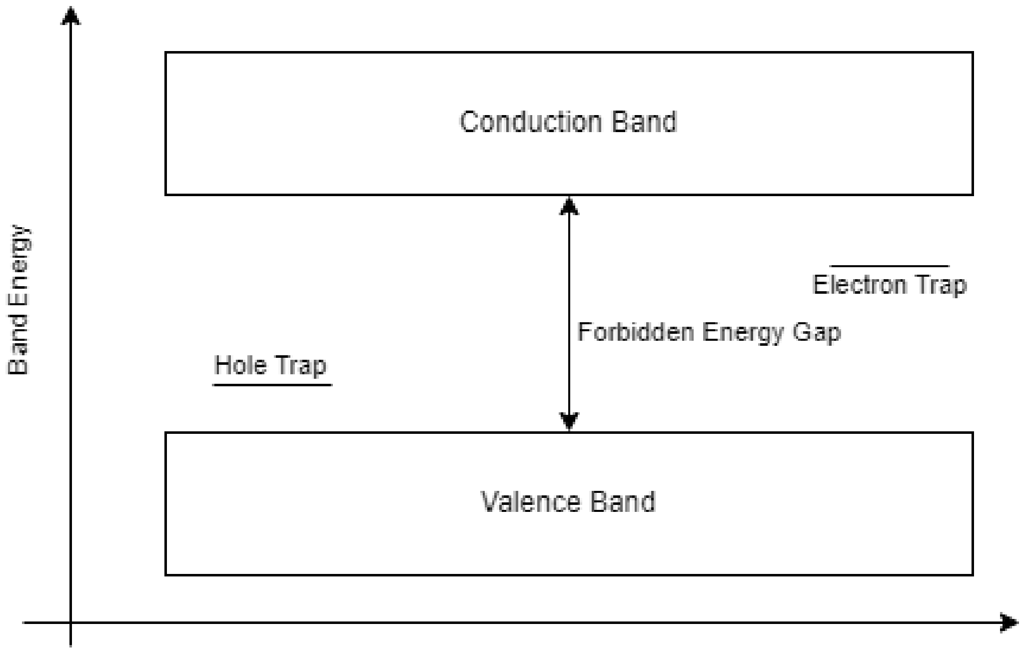

Figure 3.1. Flat Band Energy Diagram Indicating the Forbidden Energy Gap

After an electron gets enough energy to move into the conduction band, the electron will return to its equilibrium state. Sometimes, during this transfer, electrons get “trapped” in electron traps, caused by defects or impurities in the crystal lattice structure of an insulator. Traps that are close to the conduction band and valence band are called shallow electron traps and shallow hole traps respectively. These traps are not stable and it is easy for electrons or holes to escape from these traps using thermal energy. However, traps that are far from either band are called deep traps. If an electron or hole is trapped in a deep trap, it will be there for an extended period, unless heated or illuminated with light of a proper wavelength.

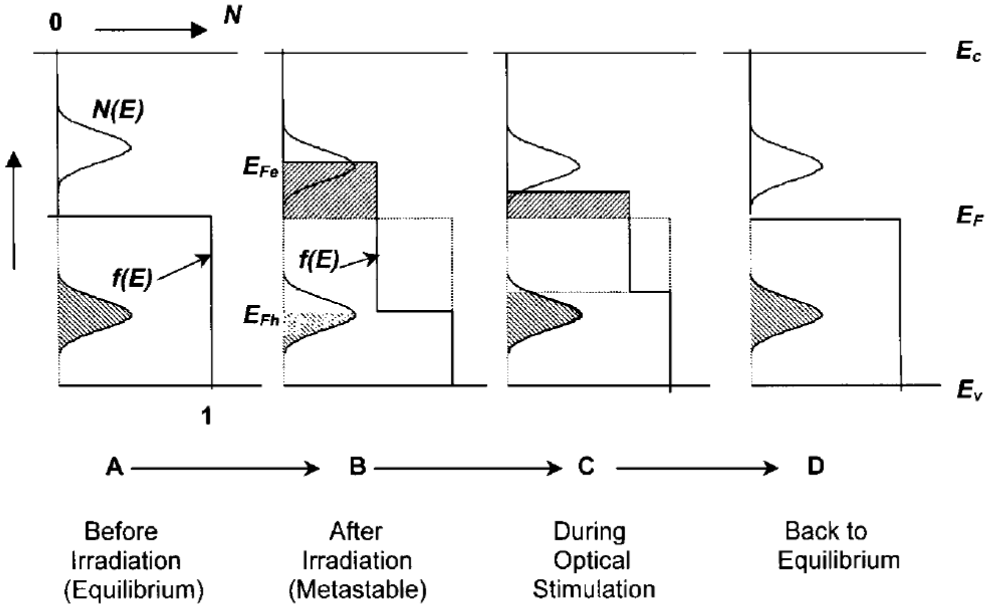

Figure 3.2. Occupancy of forbidden band gap states at 0°K

Figure 3.2 represents the occupancy of forbidden gap states at absolute zero temperature (Bøtter-Jensen et al, 2003). Here, f(E) is the occupancy function of said state, E_C is the conduction band energy, E_F is the Fermi energy, and E_V is the valence band energy. Figure 3.2(A) shows the separate valence bands and occupancies before irradiation. All states below E_F are shown to be full. In the metastable (B) state, all states up to the quasi-Fermi level E_{Fe} are filled and several states below the fermi level are empty, until the quasi-Fermi level for holes, E_{Fh}. During optical stimulation (C), full states above the Fermi levels are slowly emptied and recombination takes place until all states return to their original equilibrium states (D). Radiative transition includes the recombination of electrons and holes that result in the emission of a photon. Since this transition mainly takes place in deep traps, it requires significant energy. This energy must be externally supplied by, for example, light or heat.

3.1 Thermoluminescence

If the energy supplied externally is in the form of heat, the mechanism is called Thermoluminescence. Thermoluminescence is the type of luminescence where the mechanism is heat, applied on certain crystalline materials. This phenomenon occurs when certain crystalline materials that have been exposed to ionizing radiation are subjected to heat. Electrons trapped in lattice defects due to ionizing radiation are then released from their traps and return to their ground states. This transition causes electrons to release energy in the form of a photon (McKeever, 1985).

This mechanism can be divided into three parts, trapping, heating, and emission. Initially, trapping happens when a material is exposed to ionizing radiation and electrons are excited and then subsequently trapped in the defects. Trapped states are stable unless there is an external energy applied. With heating, this thermal energy provides the electrons the means to escape from their trapped states. When these electrons recombine with holes, absorbed energy is released in the form of photons, which can be observed as thermoluminescence (Aitken, 1985).

Thermoluminescence is an important tool in archeology, geology, and dosimetry. One of its most common uses is TL Dating, which is used to estimate the age of archaeological artefacts such as pottery and geological sediments. By measuring the emitted light, it is possible to collect information about the last time the material under testing has been exposed to high temperatures (or sometimes light), providing information about ancient artefacts and geological events. As an addition, TL Dosimeters have been the most common type of personal dosimeters up until recent times, where the trend is now shifting to OSL Dosimeters. In TLD, the amount of radiation the sample has been exposed to can be measured, a concept very useful in radiology and environmental monitoring (Bailiff, 2010). A thermoluminescence glow curve of a Dy-doped $BaSi_2O_5$ is given in Figure 3.3 (Alajlani et al, 2022).

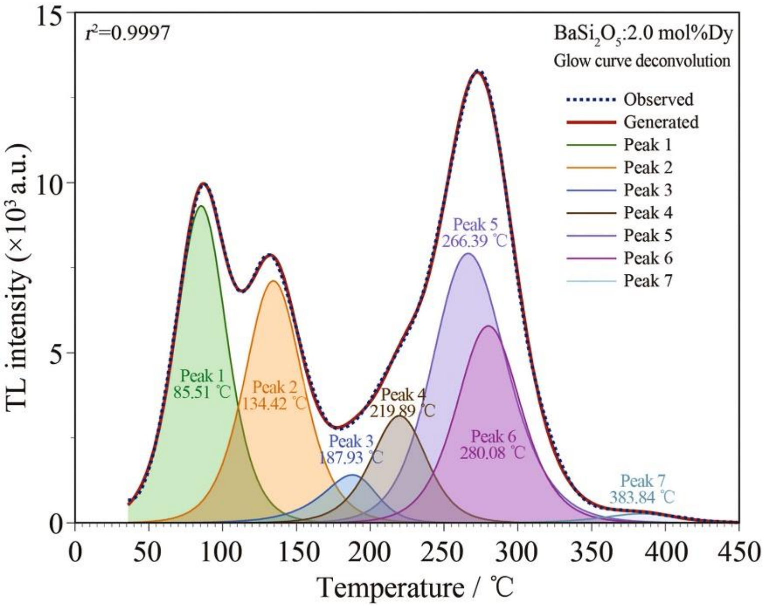

Figure 3.3. Thermoluminescence Glow Curves of $BaSi_2O_5$:Dy Phosphor

3.2 Introduction to Optically Stimulated Luminescence

Optically Stimulated Luminescence (OSL) is a technique used in radiation dosimetry, geochronology, and archaeology to measure various parameters generally found in minerals such as quartz and feldspar. This technique makes use of the property of these minerals' emission when they are optically stimulated, in other words, exposed to light. The emitted light, or luminescence, is directly linked to the radiation dose received for dosimetry and related to time since the sample has been exposed to heat or sunlight, allowing researchers to date samples and study past radiation exposure with good accuracy and precision.

Understanding the fundamentals of OSL techniques is important for making use of their full potential in scientific and practical areas. This chapter will explore different stimulation modalities of OSL, focusing on their mechanisms, applications, advantages, and limitations. In the end, emphasis will be put on the advanced capabilities of Time-Resolved OSL.

3.2.1 Importance of Luminescence in OSL

Optically Stimulated Luminescence (OSL) is essential in the age determination of geological and archaeological materials as it enables the estimation of the last time mineral grains were exposed to light or heat.

The age determination process works by stimulating the material with light, causing trapped electrons—accumulated over time due to natural background radiation (due to the elements within the earth and/or cosmic background radiation)—to be released. As these electrons return to their ground state, they emit photons (luminescence). The amount of luminescence emitted is proportional to the accumulated radiation dose, which, in turn, is proportional to the time since the last exposure to sunlight or heat. This makes OSL an effective tool for dating sediments, pottery, and other archaeological artifacts.

Moreover, OSL's ability to date events up to several hundred thousand years old makes it invaluable in studying human history, geological processes, and the Earth's past status. Its non-destructive nature preserves the integrity of archaeological samples, allowing for repeated testing and verification. The versatility and accuracy of OSL have thus made it a keystone in the fields of geochronology and archaeology, offering a reliable method to reconstruct historical timelines and environmental changes (Rapp, 1998).

OSL's importance extends beyond dating. It provides an understanding of past environmental conditions, helping researchers figure out changes in climate and landscape over time. This technique can also be used in radiation dosimetry, where it measures radiation exposure for health and safety monitoring.

Similar to age determination, radiation dosimetry also employs the concept of trapped electrons, however, these electrons are accumulated by exposure to ionizing radiation from radioactivity. The rest of the mechanism is the same, therefore luminescence is proportional to the radiation dose the sample has absorbed. A basic band gap model of a dosimetry OSL system can be found in Figure 3.4 (Musa et al, 2017). OSL dosimeters are generally more sensitive to low levels of radiation than other dosimeter types and, therefore are convenient to use as passive radiation monitors that can be quickly and easily measured (Bøtter-Jensen, 2003).

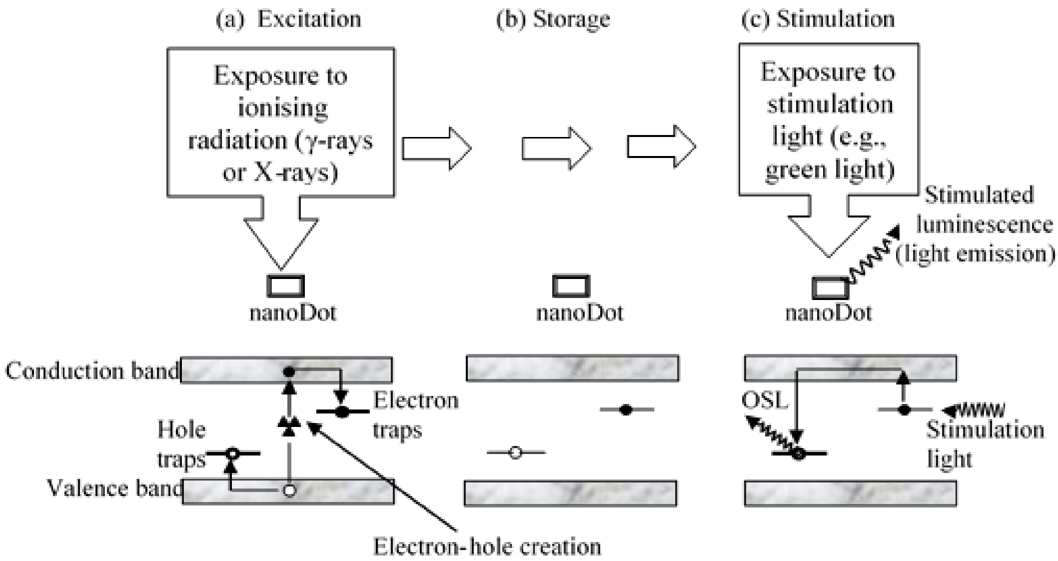

Figure 3.4. Basic Band Gap Model illustrating OSL process.

In recent years, OSL films (or imaging plates) have been considered in medical dosimetry applications. These films are reusable, have a sub-mm spatial resolution, and a linear dose-response. These films are generally used in radiotherapy, where the location of the absorbed dose is of importance, thus the resolution of the films makes them a suitable replacement for Gafchromic foils (Wouter et al, 2017).

3.2.2 Types of Optically-Stimulated Luminescence

OSL includes three main measurement modalities. Every one of them has distinct stimulation/measurement profiles and applications with different levels of sensitivity and detail. These measurement modalities are Continuous Wave OSL, Linearly Modulated OSL and Pulsed OSL. There also is a subtopic of POSL, which this thesis will focus on, and that is Time-Resolved OSL. TR-OSL differs from the others due to the possibility of providing detailed information about the luminescence process, making it invaluable for advanced research and measurement. The stimulation modes of the three main OSL types are illustrated in Figure 3.5 (Bøtter-Jensen et al, 2003).

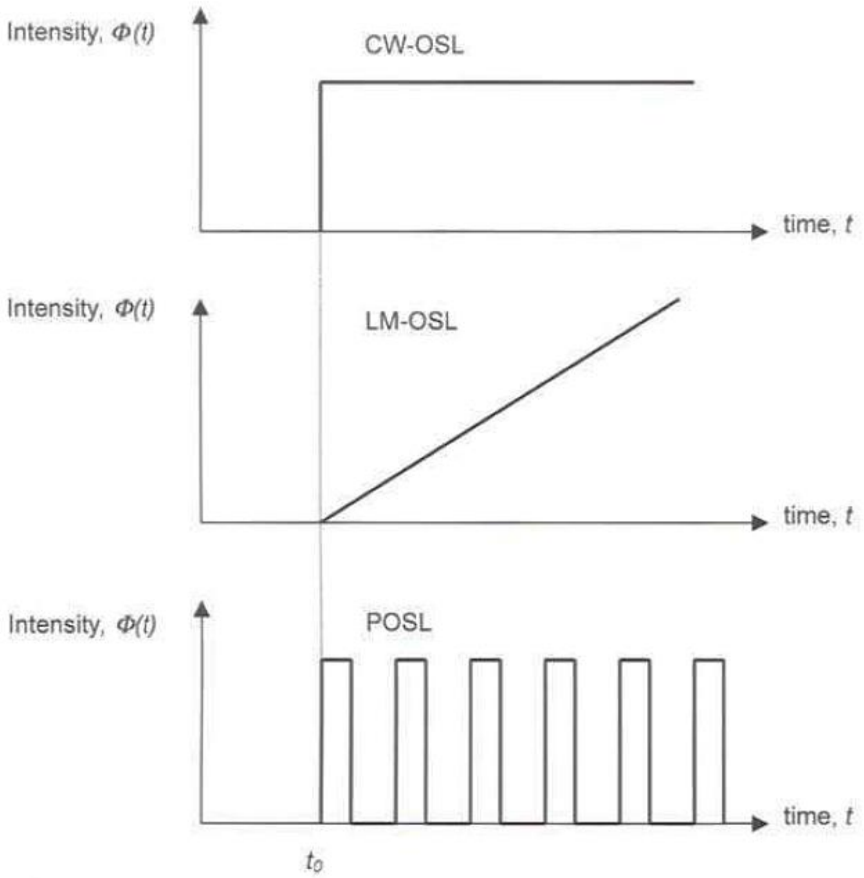

Figure 3.5. Light intensity representation for the main OSL modes.

3.2.2.1 Continuous Wave OSL (CW-OSL)

Continuous Wave Optically Stimulated Luminescence (CW-OSL) is the most common method, used widely for dating and dosimetry.

To measure the luminescence, the sample is illuminated continuously, generally in the blue (Bøtter-Jensen et al, 1999) or green (Galloway et al, 1997) areas of the color spectrum. Illumination provides the energy needed to release the trapped electrons. As the electrons return to their original positions, they recombine with positively charged vacancies, or holes. This recombination process releases a photon. This process is known as optically stimulated luminescence.

The intensity of the luminescence during CW-OSL decreases monotonically (nearly exponentially) with respect to time. Initially, the luminescence is strong because many electrons are being freed from traps. However, as more electrons recombine and fewer remain trapped, the intensity decreases. This decay pattern is crucial for analyzing luminescence and understanding data to determine the radiation dose the sample has absorbed.

CW-OSL is valued for its simplicity and efficiency, making it a preferred method for common applications. In archaeological dating, CW-OSL can determine the last time a sediment or artifact was exposed to sunlight. Exposure to sunlight "resets" luminescence. In radiation dosimetry, CW-OSL helps measure accumulated radiation doses in materials, allowing the researcher to use said material in dosimetry applications, thus contributing to human safety in various environments, such as space.

Despite its advantages, CW-OSL has some limitations. Continuous stimulation does not allow the researcher to distinguish between different types of luminescence signals from various trapping states within the material. Also, while CW-OSL provides reliable data for many applications, it lacks the resolution required to separate luminescence signals with different decay constants.

In summary, CW-OSL is a robust and useful technique for researchers working in the fields OSL is commonly used. Its straightforward approach and effectiveness in measuring radiation doses and dating samples are what make it common, and reliable and these qualities ensure its future use in both scientific research and applications.

3.2.2.1.1 Mathematical Model of CW-OSL

The mathematical model has been explained in several sources. In this text, a simplified model (Chithambo, 2018) will be followed. Only the first-order (or monomolecular) kinetics will be explained for the sake of simplicity.

While illuminated, electrons are freed from their trapped states by the following equation;

$$p = \phi\sigma \tag{1}$$

Where ϕ is the stimulation light intensity (number of photons per cm^2 per second) of the stimulation light, $\sigma(cm^2)$ is the photoionization cross section, $p(s^{-1})$ is the de-trapping probability. Taking n_{trap}, n_{holes} and $n_{conduction}$ as the concentrations of trapped electrons (number per cm^3), holes available at the recombination center, and electrons in the conduction band respectively, one can easily deduce;

$$\frac{dn_{conduction}}{dt} = -\frac{dn_{trap}}{dt} + \frac{dn_{holes}}{dt} \tag{2}$$

Rate equations for re-trapped electrons and luminescence can be written as;

$$\frac{dn_{trap}}{dt} = n_{trap}p - n_{conduction}A(N - n_{trap}) \tag{3}$$

$$L(t) = -\frac{dn_{holes}}{dt} = n_{conduction}A_m n_{holes} \tag{4}$$

In which $A(m^3s^{-1})$ is the probability of electron trapping, $A_m(m^3s^{-1})$ is the probability of recombination and $N(m^{-3})$ is the total concentration of traps. Then, one needs to assume the quasi-equilibrium equation, which claims that the conduction band free electron concentration is stable. Assuming the quasi-equilibrium conditions is essential in obtaining an analytical expression for the luminescence decay as the above set of equations does not have an analytical solution without assuming quasi-equilibrium. A simple one trap, one recombination center model is shown in Figure 3.6 (reproduced from Chithambo, 2018), where 1 represents optical stimulation, 2 represents retrapping, 3 represents recombination, n, and m are electron concentrations of the trap, and the hole concentration of the hole, respectively.

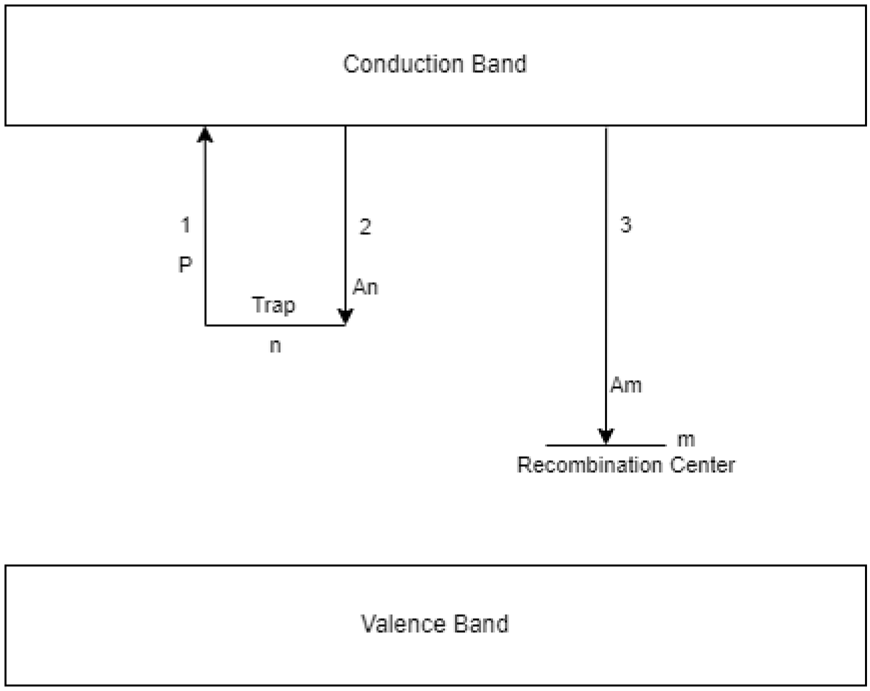

Figure 3.6. A model of a one-trap, one-recombination center band structure.

$$\left|\frac{dn_{conduction}}{dt}\right| \ll \left|\frac{dn_{trap}}{dt}\right|, \left|\frac{dn_{holes}}{dt}\right| \tag{5}$$

When re-trapping and $n_{conduction} \ll n_{trap}, n_{holes}$ are both assumed, the OSL intensity can be found as;

$$L = -\frac{dn_{trap}}{dt} = n_{trap}p \tag{6}$$

Time dependence of the OSL can then be derived by, using n_0 as the initial trapped electron concentration and L_0 as the initial intensity;

$$L = n_0 p e^{-pt} = L_0 e^{-pt} \tag{7}$$

In equation 7, L_0 is dependent on the absorbed radiation dose, which defines the CW-OSL decay curve.

An experimental CW-OSL measurement graph of SiO_2 tested with a 480nm laser is shown in Figure 3.7 (Espinosa, 2011).

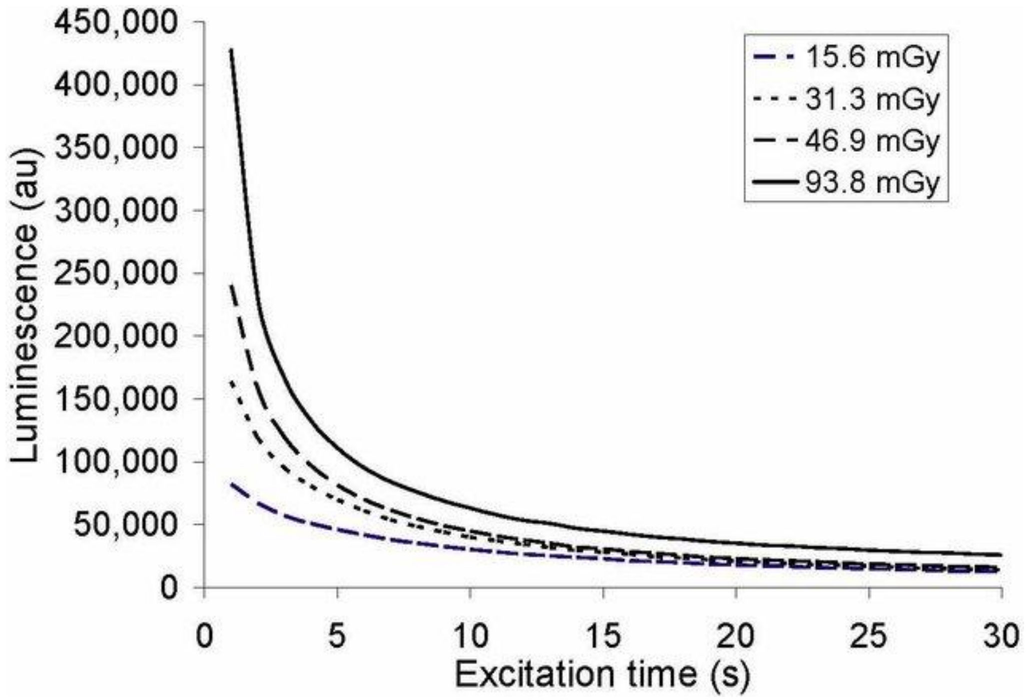

Figure 3.7. CW-OSL Exponential decay plots, for four different gamma doses (after Espinosa, 2011)

As can be understood from the figure, the peak point of the luminescence decay signal (L_0) is proportional to the absorbed radiation dose. The rest of equation 7 defines an exponential decay, which corresponds to the decreasing number of de-trapped electrons.

3.2.2.2 Linearly Modulated OSL (LM-OSL)

Linearly Modulated Optically Stimulated Luminescence (LM-OSL) (also conventionally named the Bulur method) is a measurement modality promising enhanced capabilities to analyze the luminescence properties of minerals. Unlike CW-OSL, where the sample is continuously illuminated with constant light intensity, LM-OSL involves a stimulation light with its intensity linearly increasing over time. This increase in light intensity allows for the detailed examination of the material's luminescence process.

While ionizing radiation exposure, electrons in the mineral get trapped at defects or imperfections within the crystal lattice. In LM-OSL, as the light intensity increases, it provides different amounts of energy with photons of different wavelengths that can free these trapped electrons at different rates. This "linear modulation" helps in distinguishing between electrons trapped at different sites, in other words, with different activation energies. Electrons at shallow traps need less energy to be released and, thus will emit luminescence earlier in the stimulation compared to those in deeper traps.

The luminescence emitted is recorded as a function of increasing light intensity. This results in a luminescence curve that can include multiple peaks, each corresponding to different types of traps within the mineral. Analyzing these peaks allows researchers to distinguish between overlapping luminescence signals, providing a clearer understanding of the different trapping sites and their properties. A measurement plot from the LM-OSL introduction paper from E.Bulur (1996) is given in Figure 3.8.

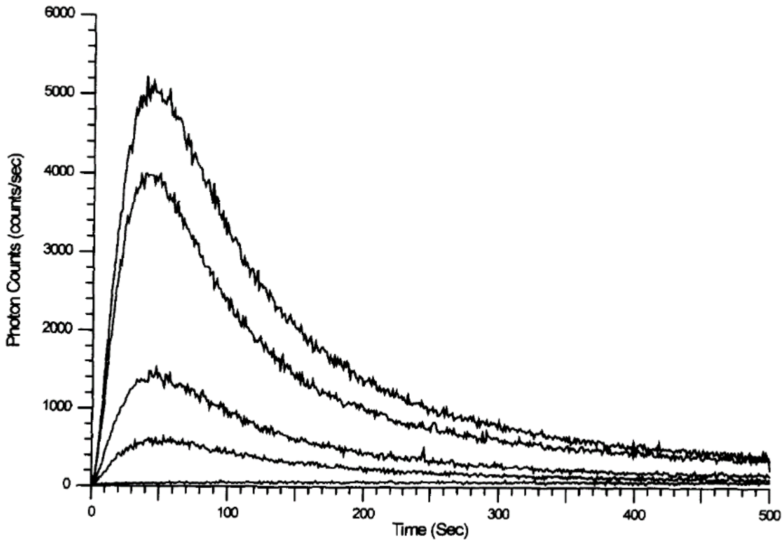

Figure 3.8. LM-OSL Peaks for different samples

LM-OSL is useful in the characterization of mineral samples. It offers information on the overlapping luminescence signals, which is important for understanding the thermal and radiation history of the sample. In archaeological dating, LM-OSL can help identify and separate different luminescence signals, improving the accuracy of age determinations. In radiation dosimetry, it helps in accurately measuring radiation doses by distinguishing between signals from different types of traps.

One of the key advantages of LM-OSL is its ability to provide detailed information on the luminescence properties of a sample. However, this method requires more sophisticated instrumentation and analytical techniques compared to CW-OSL. The linear modulation of light intensity and the analysis of the resulting luminescence curves require precise control.

In summary, LM-OSL is a powerful technique for understanding different luminescence signals and detailed properties of trapped electrons in materials. The ability to differentiate between various trapping sites based on their activation energies makes it invaluable for advanced research in archaeology, and radiation dosimetry. Information gained from LM-OSL contributes significantly to our understanding of the luminescence processes and the history of radiation exposure in natural samples.

3.2.2.2.1 Mathematical Model of LM-OSL

In this type of OSL, the genesis of this model (Bulur, 1996) will be followed. Again, only first-order kinetics will be explained for the sake of simplicity.

The rate of change of trapped electrons can be explained as, where, $n(t)$ is the number of trapped electrons, ϕ is the probability of electrons escaping from traps at initial intensity;

$$\frac{dn(t)}{dt} = -\sigma\phi n(t) \tag{8}$$

If the number of trapped electrons is assumed to be, n_0, intensity can be shown as the same as in equation 7. If the intensity is increased from 0 to ϕ in a measurement period of T, then the rate of change of trapped electrons becomes, where T is the total measurement time

$$\frac{dn(t)}{dt} = -\sigma\frac{\phi}{T}\, t\, n(t) \tag{9}$$

With this expression, the trapped electron concentration can be expressed and lead to

$$n(t) = n_0 e^{-\frac{\sigma\phi}{2T}t^2} \tag{10}$$

And thus, the luminescence intensity would be;

$$L(t) = -\frac{dn(t)}{dt} = n_0\frac{\phi}{T}te^{-\frac{\sigma\phi}{2T}t^2} = L_0\frac{t}{T}e^{-\frac{\sigma\phi}{2T}t^2} \tag{11}$$

The position of the OSL peak, t_{max}, can easily be deduced from Eq. 12;

$$t_{max} = \sqrt{\frac{T}{\sigma\phi}} \tag{12}$$

The maximum value of the new OSL peak can then be calculated by;

$$L_{max} = n_0 \sqrt{\frac{\sigma\phi}{T}} e^{-\frac{1}{2}} = \frac{n_0}{t_{max}} e^{-\frac{1}{2}} \tag{13}$$

As the luminescence intensity function, the maximum value of luminescence, and the position of the said maximum are known, it is possible to simulate, understand, and analyze data taken by applying the LM-OSL method. L_{max} is, as can be understood by looking at the right-hand side of Eq. 14, is dependent on n_0, the number of initially trapped electrons, which is proportional to the radiation dose absorbed, which is the crucial parameter for age-determination of dosimetry. The most important advantage of using LM-OSL, however, is the ability to discern between different traps.

3.2.2.3 Pulsed OSL (POSL)

Pulsed Optically Stimulated Luminescence (POSL) is another mode of OSL measurement, distinguished by its use of light pulses rather than continuous or linearly increasing light. This method involves illuminating the sample with a series of brief, controlled light pulses and measuring the resulting luminescence during or after each pulse. This pulsing approach provides several advantages, particularly in separating and analyzing different luminescence components.

When a mineral like quartz or feldspar is exposed to ionizing radiation, electrons become trapped at defects or imperfections within the crystal lattice. To release these trapped electrons, the sample is subjected to short pulses of light. Each pulse provides a burst of energy that can free some of the trapped electrons, causing them to recombine with holes in the crystal lattice and emit light (Sanderson, 1994).

After each light pulse, the luminescence is measured during the “dark” time before the next pulse. A visualization of the mechanism is shown in Figure 3.9. In the figure, black lines represent the pulses, and red lines represent the luminescence signal.

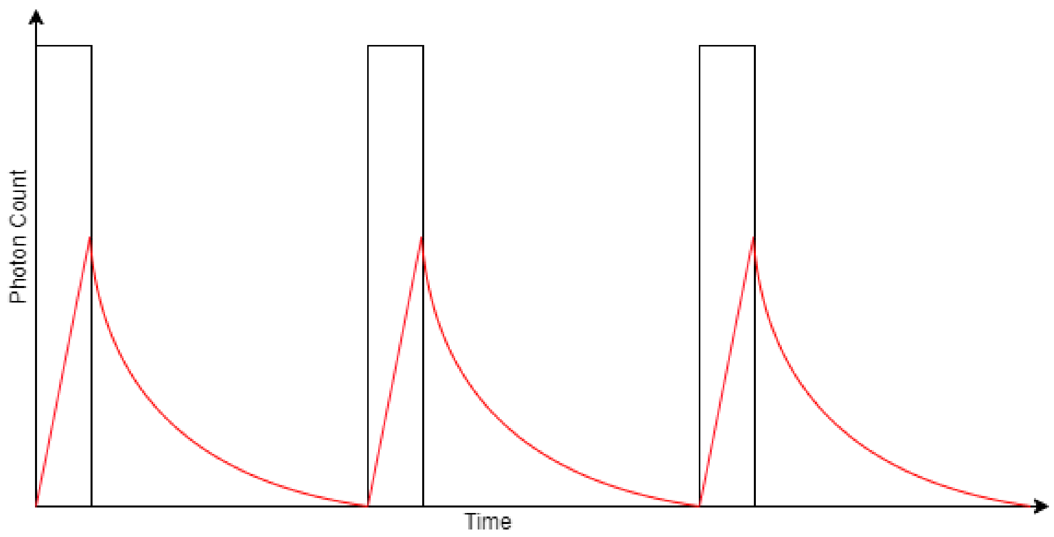

Figure 3.9. Example POSL Stimulation and Measurement Curves where the black lines indicate stimulation and red lines indicate stimulated luminescence and decay

This interval allows the luminescence signal to be observed without the interference of the pulsed illumination. By analyzing the decay of the luminescence signal between pulses, researchers can gain detailed insight into the luminescence process. This pulsed measurement method enhances the ability to distinguish between fast and slow components of the luminescence signal, which corresponds to different types of trapping sites with varying energy levels. An example of a POSL decay curve can be found in Figure 3.10 (Tsukamoto et al, 2013).

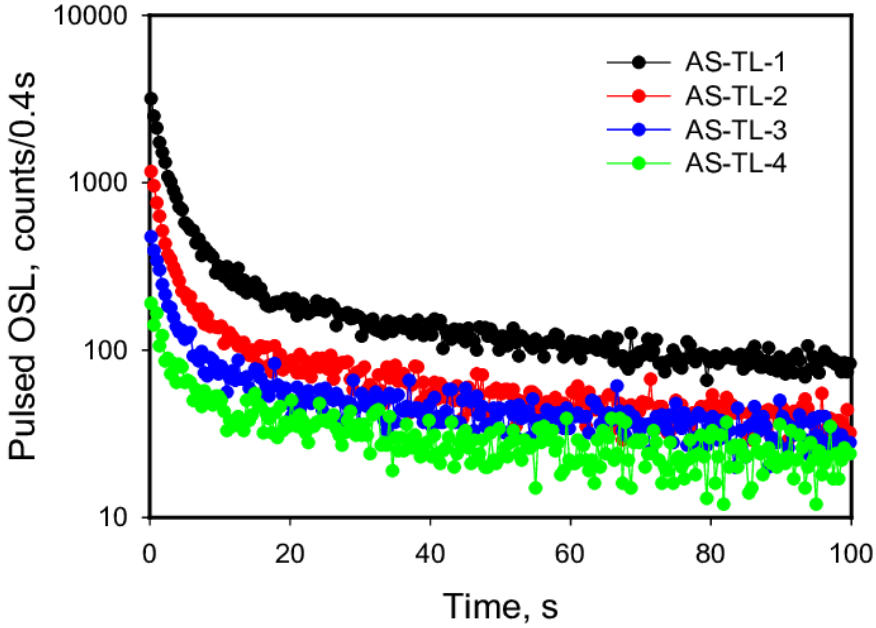

Figure 3.10. POSL Decay Curves

POSL is particularly useful in scenarios where high sensitivity and precision are required. It improves the signal-to-noise ratio by allowing the separation of luminescence signals from background noise more effectively than the two previous methods. This makes POSL highly effective for low-dose radiation measurements and for analyzing samples with complex luminescence behavior.

The detailed resolution provided by POSL is advantageous in several applications. In archaeology, POSL can be used to date samples with greater accuracy by clearly separating different luminescence components. In radiation dosimetry, it helps in accurately assessing radiation exposure, especially in environments where radiation levels are low or changing. POSL also contributes to the study of luminescent materials, aiding in the development of materials better suited for dosimetry and improving our understanding of the luminescence mechanisms.

While POSL offers significant advantages in terms of sensitivity and detail, it requires more sophisticated instrumentation to generate and control the light pulses accurately and to measure the luminescence precisely. The data analysis process is also more complex, as it involves interpreting the decay curves of luminescence signals between pulses.

In summary, Pulsed Optically Stimulated Luminescence (POSL) is a powerful and precise technique for studying luminescence properties and measuring radiation doses. By using short light pulses and recording the luminescence output after the pulse, this method leads to an enhanced sensitivity and better signal-to-noise ratio. These capabilities make POSL important for applications in radiation dosimetry (for example, Landauer OSL Readers use this method), archaeology, and material science.

3.2.2.3.1 Mathematical Model of POSL

Chithambo and Galloway introduced a simplified mathematical model for POSL (Chithambo & Galloway, 2000). Taking the initial trapped electron number as n_0, probability of stimulation with respect to time as s, the probability of a stimulated electron to emit a photon to be P and stimulated electron number at a time t as $n(t)$, one can say;

$$dn(t) = sn_0dt - Pn(t)dt \tag{14}$$

It can be found that the time dependence of the equation can be shown as;

$$n(t) = \frac{sn_0}{P}\left(1 - e^{-Pt}\right) \tag{15}$$

$$Pn(t) = sn_0\left(1 - e^{-Pt}\right) \tag{16}$$

Which is an exponential increase from time $t = 0$. At time $t = t_{end}$, which would indicate the time stimulation ends, luminescence can be expressed as;

$$Pn(t) = Pn(t_{end})\left(1 - e^{-P(t-t_{end})}\right) \tag{17}$$

If one takes $pn(t)$ as the luminescence signal variable $L(t)$, the equation then becomes;

$$L(t) = \mathrm{L}(t_{end})\left(1 - e^{-P(t-t_{end})}\right) \tag{18}$$

It can be seen from Eq. 16 that the luminescence signal under illumination is an exponential increase, dependent on the initial trapped electron concentration and the probability of stimulation, therefore proportional to the age or radiation dose absorbed by the sample. The exponential decay, however, is a function of how long the sample is illuminated, and the remaining concentration of electrons at the end of the illumination process, which can then be linked with the electrons and holes remaining in the recombination center, providing valuable data on the centers and their concentrations.

3.2.2.4 Time-Resolved OSL (TR-OSL)

Time-Resolved Optically Stimulated Luminescence (TR-OSL) is an advanced submethod of the POSL technique, divided for its ability to provide high-resolution time-related data on the luminescence process. This technique involves measuring the emitted luminescence at specific time intervals following short, strong, controlled pulses of stimulating light, rather than using continuous or linearly modulated light sources or measuring in between pulse intervals. By analyzing how the luminescence decays over time after each pulse, researchers may develop a better understanding of the mechanisms of recombination processes within the material.

Similar to POSL, stimulation pulses release the trapped electrons and subsequent recombination results in the emission of light. Unlike POSL, where the luminescence intensity during or after pulses is recorded, in TR-OSL luminescence decay after the pulse is recorded in time.

As the light is being pulsed, released electrons recombine with holes in the crystal lattice. In TR-OSL, the critical aspect is the timing of the luminescence measurement. The emitted light measured after the end of the stimulation pulse– recorded as a function of time- is generally an exponential decay with a time constant ranging from microseconds to seconds. Such a slow decay may give information about the forbidden transitions in the recombination centers. A time-resolved OSL decay example is given in Figure 3.11, where Maraba (2017) calculated the lifetime of the F-centers in Al_2O_3:C which act as recombination centers. The inset within Fig. 3.11 also shows the proportionality relation between the absorbed dose and the luminescence signal intensity.

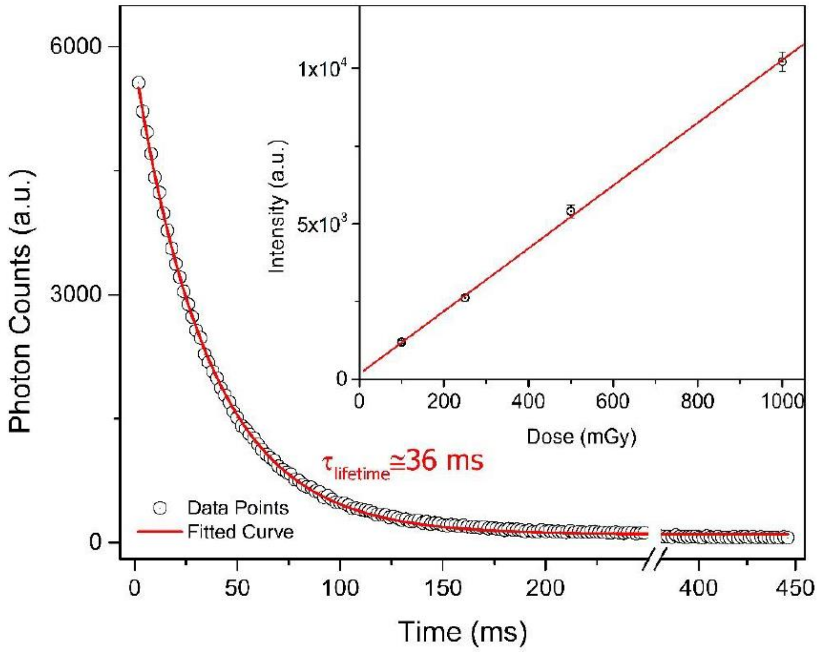

Figure 3.11. Time-Resolved OSL curve example

By examining the luminescence decay curves, researchers may gain knowledge of the recombination mechanism. The measurement yields information on the luminescence centers' properties, for example, whether the transition is allowed or forbidden. Combining this information with the emission spectrum allows the researcher to characterize the luminescent centers (Bailiff, 1998).

TR-OSL is valuable for applications requiring high sensitivity and precise time resolution. In radiation dosimetry, this method enables accurate measurement of radiation doses, even in environments with low radiation levels or mixed radiation exposure. This makes it useful in medical physics, where precise dosimetry is important for patient safety. In archaeology, TR-OSL helps date geological and archaeological samples by providing detailed information about the sample's luminescence characteristics, leading to more accurate age determinations.

One advantage of TR-OSL can be stated as the possibility of resolving luminescence emissions from different centers (provided that they have distinct recombination lifetimes). On some occasions, these centers could be different recombination centers created by different types of radiation; for example heavy ion radiation in space (see e.g. Yukihara and McKeever, 2011b). Another possibility is that different mineral contributions may exist in a sample such as feldspar and quartz may coexist in a sediment. Due to the great differences in lifetimes, it is possible to discriminate the feldspar signal (in the order of 10 nanoseconds) from the quartz signal (around 40 microseconds), as can be seen from the figure (Denby et al., 2006).

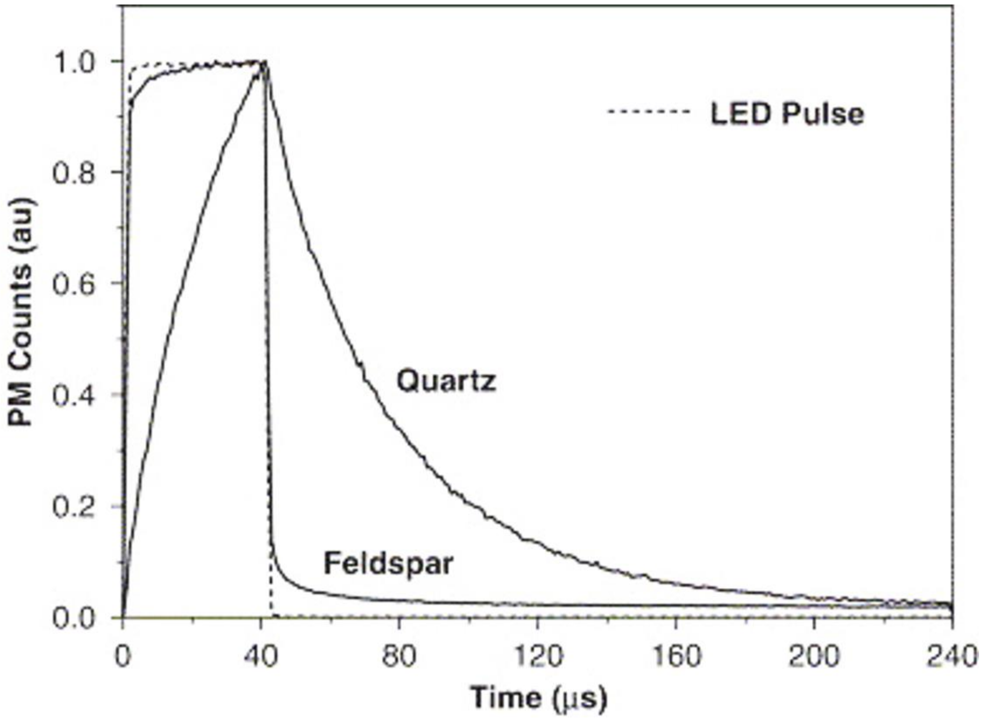

Figure 3.12. Lifetime Separation of Quartz and Feldspar

Implementing a TR-OSL device requires complicated instrumentation capable of generating exact light pulses and capturing high-resolution time-resolved data. The analysis of TR-OSL data is also more complex than other OSL methods, as it involves fitting the luminescence decay curves to models that describe recombination.

3.2.2.4.1 Mathematical Model of TR-OSL

A model of TR-OSL was proposed by Chithambo (2007). A team of researchers recently published an analytical model of TR-OSL beautifully, thus this text will follow their expressions (Pagonis et al, 2011). These two models have different approaches but converge on the same result.

Assuming direct radiative transition probability to be P and n_2 as excited electron number in recombination centers, instantaneous luminescence can be defined as;

$$I(t) = Pn_2 \tag{19}$$

In temperature-dependent measurements, P can be expressed as $P + A_{NR}e^{\frac{-W}{kT}}$ (where A_{NR} is the nonradiative transition probability, k is the Boltzmann constant, T is the temperature and W is the activation energy), but, in the scope of this research, since T is approximately constant, around room temperature, and materials under testing will not be heated, $e^{\frac{-W}{kT}} \cong 0$, so that temperature dependence can be ignored. The thermal component of this mathematical model will be considered negligible and will not be written. Following n_2, one would have to express its time dependence as;

$$n_2(t) = \frac{f}{P}\left(1 - e^{-[P]t}\right) \qquad 0 < t < t_0 \tag{20}$$

Where f is a steady-state constant dependent on the transition probabilities and the initial concentration of electrons. t_0 is the pulse width. At the end of the pulse sequence, when $f = 0$ is set into the equation, $n_2(t)$ can be expressed as;

$$\frac{dn_2}{dt} = -n_2[P] \tag{21}$$

The solution of this equation is then;

$$n_2(t) = n_2(t_0)e^{-P(t-t_0)}, \;\; t > t_0 \tag{22}$$

Combining the above equations, one can express the luminescence intensity for both the duration of the pulses and after the pulses;

$$L(t) = f(1 - e^{-Pt}), \;\; 0 < t < t_0 \tag{23}$$

$$L(t) = Pn_2(t_0)e^{-P(t-t_0)}, \;\; t > t_0 \tag{24}$$

Since luminescence intensity was expressed as $t > t_0$, if one sets $t \to \infty$, L_{max} , the luminescence intensity during the optical stimulations can be calculated;

$$L_{max} = f \tag{25}$$

The relaxation time of electrons (from their excited states to the ground state of the recombination center) can be expressed as;

$$\tau = \frac{1}{P} \tag{26}$$

Equation 25 tells the reader that the maximum luminescent intensity is directly dependent on the steady-state constant f (in constant, negligible temperature), which is a parameter dependent on the initial electron concentration and the transition probabilities, therefore is directly proportional to the radiation dose absorbed by the sample. The right-hand side of Eq. 24 also yields the same conclusion to the reader. The exponential term of Eq. 24 explains the decrease in the luminescence with respect to the remaining electron concentration, thus yielding information on the recombination centers. Equation 26 describes the lifetime of the decay.

3.3 Introduction to Nanophosphors

Nanophosphors are materials that show important optical properties e.g. luminescence. They are typically made up of rare earth or transition metal doped materials, nanophosphors show good luminescence efficiency and are tunable due to quantum effects (Ryu et al., 2021). Since their shape, size, and composition can all be fine-tuned, they are important in many applications, especially in dosimetry. Nanophosphors can be explained as hosts (generally a metal oxide nanoparticle) doped with activators (normally an inorganic salt). This doping process generates energy levels between the conduction band and the valence band of the host, making absorption and efficient emission possible. At the nanoscale, quantum confinement and large surface area increase their luminescent properties. Different nanophosphors with several different emission wavelengths can be seen in the following figure (Nanomaterial Store, 2009).

Figure 3.13. Several PhosphorDots^{TM} under UV light

3.3.1 Advantages and Applications of Nanophosphors in Dosimetry

Conventional methods of dosimetry make use of thermoluminescence (TLD) or optically stimulated luminescent dosimeters (OSLD) which are in crystalline, microcrystalline, or ceramic forms. Nanophosphors are advantageous over conventional materials since they exhibit better luminescent properties such as higher sensitivity, faster response times, and a possibility of size-advantages (like fine-tunability of size). Since they put forward better properties, they are excellent contenders for OSL studies.

Nanophosphors may offer several advantages in TR-OSL applications. Their high surface area provides more trapping sites, increasing the dosimeter's sensitivity (Gupta et al., 2021). The ability to tailor their size and composition allows for optimization of their luminescent properties for specific radiation types and doses. Additionally, the fast response times of nanophosphors may enable real-time dosimetry, valuable in dynamic environments where radiation levels change rapidly. These properties make nanophosphors highly effective for precise and accurate radiation measurement. However, the advantages of nanophosphors are still not proven, thus these advantages are mostly hypothetical. Further research is needed in this area in order to explore these advantages in reality.

Since nanophosphors can be fine-tuned in their shape, size, and even activator (dopant) concentrations, their luminescent properties can be changed to the exacting standards of specific luminescence applications (Binnemans, 2020). These fine-tuned nanophosphors are essential to developing specific dosimeters with desirable sensitivity, precision, and accuracy for use in radiation safety applications (Kumar et al., 2018). These specialized dosimeters provide real-time data with extremely high sensitivity and accuracy. They also are able to detect lower-level radiation exposure than conventional methods, making them suitable for application in most zones with ionizing radiation. Since real-time monitoring is possible, this allows one to detect potential risks quickly and take preemptive measures instantaneously. (Lim et al., 2020).

3.3.2 Future of Nanophosphor Dosimetry

Future research of nanophosphors in TR-OSL includes the detection of multiple types of radiation by developing nanophosphors and provides additional information on the ionizing radiation received. These multifunctional nanophosphors will probably yield better sensitivity, more accuracy, and even more versatile materials for dosimetry, aiding in the possibility and safety of radiation use or exposure in the future (Xu et al., 2022).

Radiation dosimetry using OSL is a widely employed technique and a continuously developing field of study. New dosimeter materials should be developed/searched for as there are only two materials in use commercially. In this context, TR-OSL may help these development efforts as it supplies valuable information about the luminescence recombination mechanisms. It provides high sensitivity and resolution in applications, along with the ability to discriminate fast and slow luminescence signal components. Nanophosphor application in TR-OSL may enable accurate and precise measurement of radiation dose. In addition, OSL’s non-destructive nature and extensive dating range make OSL an invaluable tool for reconstructing historical timelines and understanding geological and archaeological processes, therefore contributing significantly to our knowledge of the Earth's history and human civilization.

CHAPTER 4

INTRODUCTION TO SCIENTIFIC INSTRUMENTATION

A tool or device used, in a scientific context, to better understand nature's many phenomena or to improve research methods further than currently known is called a scientific instrument (Hessenbruch, 2013). Physicists in particular are interested in obtaining results on how the universe works both qualitatively and quantitatively. Gaining information about a state, a value, or an amount of a quantity is called a "measurement" (Bhuyan, 2011). A complete system with both measurement and control elements is called "instrumentation" in general (Placko, 2007). An instrumentation system should be considered as two systems combined, a control system and a measurement system. The control system enables one to control both the measurement system and the complete instrumentation system. The measurement system, on the other hand, is made up of devices for gathering/collecting information, quantifying, and processing the information to make understanding a quantity possible. Instrumentation systems can be divided into two main groups: closed-loop and open-loop. In a closed-loop system, the measurement system quantitatively measures a physical variable and returns an output value to the control system (called feedback) depending on the state, value, or amount measured. In an open loop system, there's no feedback to the controlling element but only gives a dependent output. Flowcharts of open and closed-loop instrumentation are presented in Figure 4.1 (Alleman, 2002)

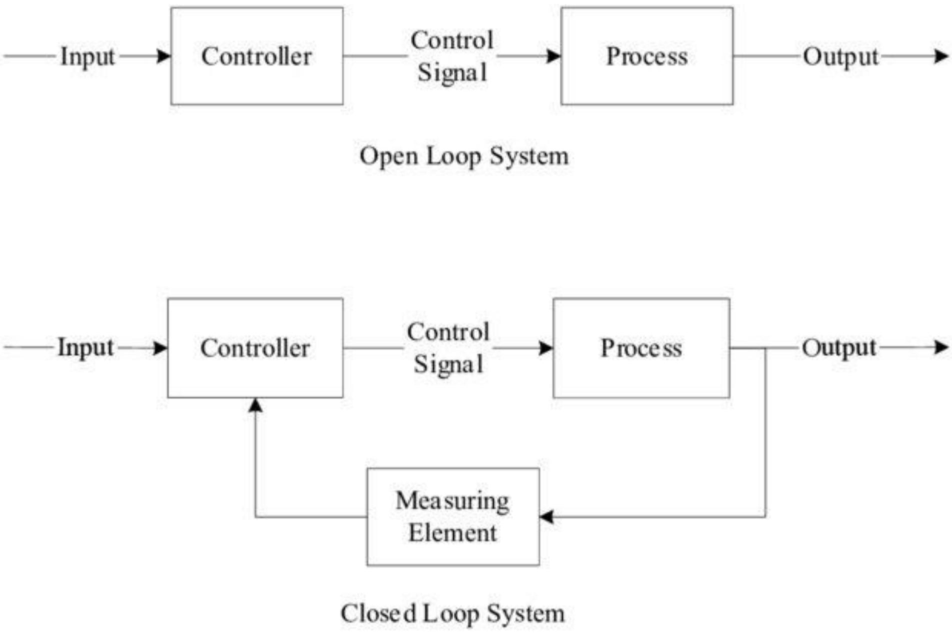

Figure 4.1. Open and Closed-loop instrumentation systems

To obtain repeatable, verifiable, and comparable results, an instrumentation system must have reliability, for the experiment to be repeatable, precision, for two or more measurements to be close to each other, and accuracy, in order to measure the true value of said quantity (McMahon, 2008). To better achieve these qualities, measurement systems/elements are constantly under research and improvement.

A measurement system, generally, is composed of a sensing element, a signal conditioning element, a signal processing element, and a data presentation element. A flowchart of the relation between the measurement system elements is shown in Figure 4.2 (Bentley, 2008).

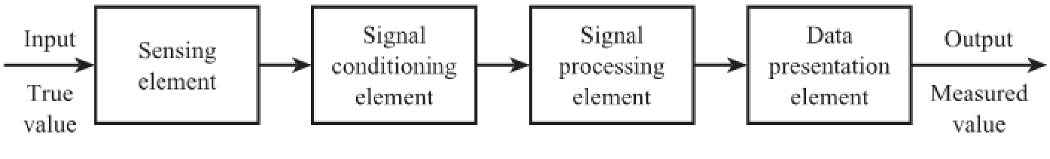

Figure 4.2. An example of the flowchart of a measurement system.

A sensing element is a device that is able to measure physical phenomena by converting some type of energy to a measurable quantity. These devices are called transducers or sensors. Transducers take an input from a physical quantity and provide an output depending on the variable measured. Signal conditioning elements are some forms of devices that convert the transducer output to a more convenient quantity to be processed. Signal processing elements take the output of the conditioning element as an input and process the signal to a more suitable form for further processing. These devices generally apply some form of calculation or correction to their input signal. Lastly, the data presentation element uses the processed data and formats it for human use.

The control system is generally composed of a microcontroller or microprocessor, with more advanced applications requiring a higher degree of sophistication using FPGAs, or even central processing units (CPU). However, most up-to-date systems use microcontrollers, for their many input/output ports, signal conditioners and processors as integrated circuits, on-board memory, and ability to control many sensors and other instruments simultaneously.

4.1 Basic Concepts of an Instrumentation System

As mentioned above, a measurement system is required to have accuracy, precision, and reliability. Extending these parameters to an instrumentation system, one has to add error, resolution, range, and repeatability to these requirements. These values define the quality and performance of the overall system. Accuracy defines the closeness of a set of measurements to their true value. One can say as the closer the measurement is to its true value, as better accuracy the system exhibits. Precision on the other hand exhibits the spread of data points. If the data are sufficiently close to each other, an instrument is called a high-precision instrument. Reliability is the ruggedness of the system overall. If a system does not break down easily under normal operating conditions, the system can be regarded as reliable. Error is the difference between the true value of a parameter and the measured value. Normally, for scientific purposes, one will have to deal with percent error. Percent Error or in other words Percent Uncertainty (Giancoli, 2009) can be defined as;

$$Percent\ Error = \left|\frac{Measured\ Value - True\ Value}{True\ Value}\right| \times 100\% \qquad (27)$$

The range can be considered as the absolute maximum and the absolute minimum values a system is available to measure. The repeatability aspect considers precision. If a system can consistently measure a similar output in repeated measurements, the system is called repeatable. Resolution is the smallest quantity the system can separate from each other and single out successfully. Final quality an instrumentation system should have can be defined as the proportionality of the input value and the output value and can be named "sensitivity". These parameters can be classified as static characteristics of an instrumentation system.

There are also dynamic characteristics, and dynamic characteristics relate the system as a whole to time. When considering an instrumentation system, one should be aware of its rise time, fall time, response time, and propagation delay. Rise time is the time duration between a signal's value to go up from 10% to 90% of its static value. Fall time is the opposite, it can be defined as the time it takes to go down from 90% to 10% of the steady-state value. Response time is the time the system takes to give a change in output following a change in input. Lastly, propagation delay is the time the signal loses whilst traveling within the system. For more details on these parameters, see Bentley (2008).

4.2 Instrumentation System Overview for OSL

In designing an OSL instrumentation system, one would have to consider all the parameters mentioned in the previous subtopic. An OSL measurement system is composed of a previously irradiated sample (acting as a transducer, with memory, to sense and store the energy of the radiation in the form of trapped charges), a stimulation light source with proper wavelength and power, and a sensitive photodetector for measuring the weak luminescence emission from the sample. The stimulation light source and the detector should have appropriate filters for wavelength selection. Input to the system is the ionizing radiation and after is a weak luminescence emission where the intensity is dependent on the radiation dose. The measurement is achieved by illuminating the sample with a stimulating light.

To achieve the OSL measurement, one may use a measurement system similar to the one shown in Figure 4.3. Stimulation and detection systems/processes are generally operated by controller hardware which is capable of managing the stimulation light source, reading and storing (acquiring) the measured luminescence data, and directing the movement systems (if available). Generally, the controller is also responsible for communicating with a client computer where a human-machine interface is running. The measured data is transferred to the client and stored for further analysis.

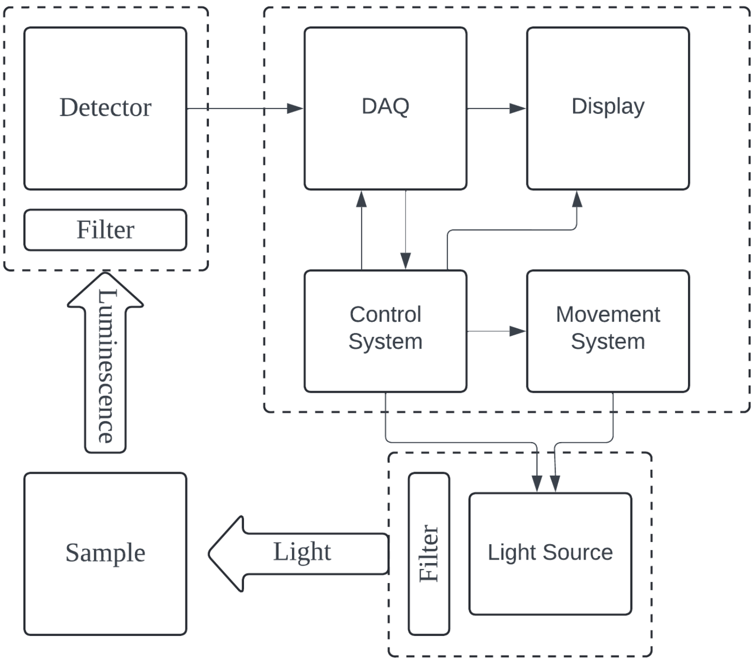

Figure 4.3. Structure of a basic OSL measurement system

4.2.1 Photodetectors

A photodetector is an essential part of the OSL measurement system, which is responsible for converting the weak luminescence emission into an electrical signal. Depending on the selected working mode, the electrical output can be in the form of electric current (continuous) or discrete electrical pulses (for absorbed photons).

Depending on the luminescence signal intensity, either of the approaches could be used. To achieve low-level light detection, there are different types of detectors are available. A few of the detectors are summarized below.

4.2.1.1 Photomultiplier Tube (PMT)

PMTs are extremely sensitive light detectors that normally work in ultraviolet, visible, or near-infrared regions of the electromagnetic spectrum. PMTs are composed of a photocathode and an electron multiplier composed of several dynodes and an anode, encompassed in a vacuum glass tube (see Figure 4.4 (reproduced from Hamamatsu, 2017)). The dynodes and the anode are held at successively higher potentials respective to the photocathode so that the generated electrons can be accelerated towards the anode. The photocathode is a light-sensitive material (generally made using alkali metals), which emits an electron (called a photoelectron) when a photon (with proper energy) is absorbed in accordance with the photoelectric effect. A photo-generated electron is accelerated toward the first dynode and the impact of the energetic electron on the dynode results in the creation of new electrons. These electrons are then accelerated to the next dynode and new electrons are generated. The travel of electrons between the dynodes increases the number of electrons and at the end, a great number of electrons are available at the anode giving out a tiny electrical pulse resembling the absorption of the photon by the photocathode. If the light intensity is relatively high, these pulses may overlap and produce a continuous current (Hamamatsu, 2017).

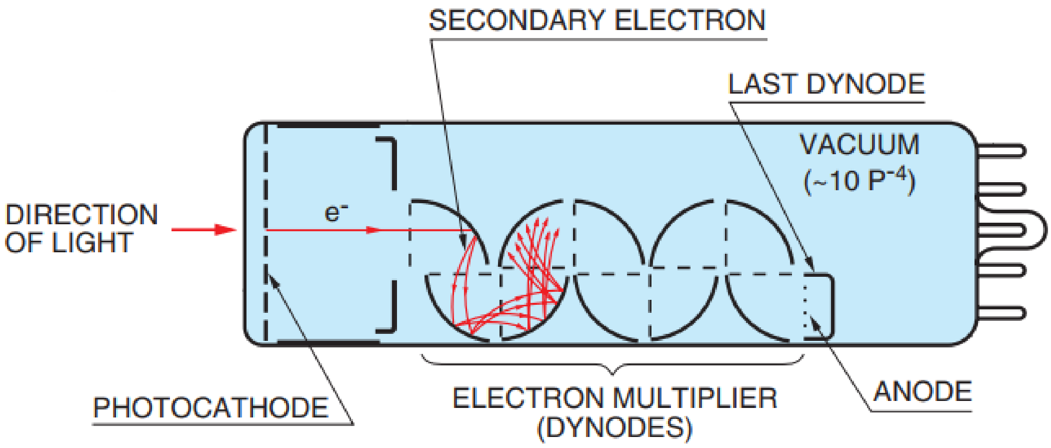

Figure 4.4. Basic Schematic of a PMT

PMTs are advantageous in scientific applications involving low-level light detection for their high sensitivity and fast response times. They are able to detect even single photons and generate a response signal within tens of nanoseconds. Their amplification factor is in the order of 10^6-10^7, although this gain can be fine-tuned using the forward bias voltage, which is normally in the kilovolt range. Their wavelength range is also an advantage; manufacturers offer a large range of photomultiplier tubes with working wavelengths from far ultraviolet to near-infrared. PMT's signal-to-noise ratio is also extremely high, making them ideal to be used in low-signal, high-noise environments. These characteristics make them ideal for applications with a precise time resolution requirement, and/or in low-light applications.

PMTs can be used in two different modes. The first mode is the *current mode* which requires the measurement of current generated on the anode. The second mode is the *photon counting* mode. In this mode, incident light is read out using a fast pulse amplifier and a discriminator to distinguish between signal heights. (see, e.g., Bøtter-Jensen et al., 2003).

In choosing a photomultiplier tube, an OSL researcher has to consider its spectral range, electron transit time, and dark current. The spectral range is the region of the electromagnetic spectrum where the photocathode material is able to eject electrons, thus it is dependent on the work function of the photocathode. Electron transit time, similar to response time, specifies the time it takes from the electron ejecting from the photocathode to the said electron arriving at the anode. Dark current is the current noise the PMT generates with no incident light. The most common cause of dark current is the thermionic emission of electrons from the photocathode and the dynodes.

4.2.1.2 Avalanche Photodiode

APDs are highly sensitive devices made up of a semiconductor structure similar to PIN photodiodes. They are designed to operate at higher (than photodiodes or PIN diodes) reverse bias voltages, thus generating a stronger electric field within the depletion region. This strong electric field causes the generated electron-hole pair to be accelerated and ionize surrounding atoms, generating a cascade effect called avalanche breakdown. Due to this cascade effect, initial charge carriers are multiplied and a much higher gain and sensitivity is achieved. This high internal gain makes APDs suitable for several scientific applications and researchers are recently working on their feasibility in OSL applications (Lim et al, 2021).

4.2.1.3 Silicon Photomultipliers

SiPMs are advanced devices composed of an array of avalanche photodiodes. These sensors are preferred for their superior sensitivity and precision compared to other semiconductor detectors in low-light-level applications. In their interior design, a number of APDs are placed as an array, and those APDs operate in their Geiger mode, generating pulses depending on incident light. This allows SiPMs to be able to detect single photons. Key characteristics of SiPMs are their photon detection efficiency, their high gain (10^5 to 10^6), and their excellent time resolution, in the order of nanoseconds. They also require a low voltage to be operated and are not sensitive to magnetic field fluctuations (Hamamatsu Photonics, 2023).

In the sense of OSL, SiPMs are an up-and-coming type of detector to be expected in the future, since their high sensitivity and fast response times will be up to par with requirements. Their current disadvantages are their size, high background noise, and their cost. Small-sized detectors require complex optical solutions to focus luminescence, and these optical devices induce losses and noise into the system.

4.2.1.4 Multichannel Detectors

Multichannel detectors, like charge-coupled devices, complementary metal-oxide semiconductor transducers, or EMCCDs (electron-multiplied charge-coupled devices), make use of multiple pixel sensors to measure light. These pixels are normally arranged in an array formation, inducing an ability to measure direction from the incident light and the possibility of spectral analysis along with imaging. (Bestvater et al., 2010)

4.2.2 Light Sources

In modern applications of OSL, one needs to consider LEDs and Lasers as stimulation light sources. LEDs are semiconductor devices that emit light when current passes through a semiconductor junction. Emitted photons are produced from the recombination of electrons and holes. LEDs (4.5 B) are preferred for their efficiency, lifespan, spectral range, availability, and cost. Lasers (Figure 4.5 A), however, emit light using the process of stimulated emission. This process generates a coherent, monochromatic, and highly directional beam of light. This is made possible by amplifying the light using a gain medium within a resonant cavity. This emitted light can be found to have precise wavelength and phase properties (Kasap, 2017).

In OSL, the light source is used to stimulate luminescence from materials. LEDs are advantageous for their low cost, spectral range, and uniform illumination within their given angle. This makes stimulating several materials simultaneously possible.

Lasers on the other hand provide better power and precision (in the sense of directionality and speed). Coherence and the monochromatic nature of lasers allow for point-by-point stimulation of a sample, leading to better sensitivity (Bøtter-Jensen et al., 2003).

4.2.3 Filters

Optical filters are devices that either transmit or block a specified wavelength or a range of wavelengths. By achieving this, controlling the light passing through a point becomes possible. They can be categorized as bandpass, longpass, or shortpass filters. Generally, they are made from colored glass or glass coated with thin film layers to filter or pass specific wavelengths of light. Filters are essential in systems where precise control over wavelength is required. In an OSL system, they are mainly used to fine-tune the detector to the luminescent light wavelength and to separate between the incident illumination and luminescence.

4.3 Open-Source Instrumentation Devices

Open-Source is a hardware and software development model in which the design or the source code respectively are publicly available and not under copyright for non-commercial use. These designs and software are viewable, modifiable, and distributable by any person. Open-source software ideals can be traced back to the very beginning of computing, but it has gained popularity and momentum with the formation of the Free Software Foundation (FSF) in 1985. The vision behind FSF was to encourage the freedom to share, modify, use, and distribute software (Stallman, R., 2002). This ideal was the key behind GNU (a mass collaboration free-software project) General Public License (GPL). The term was first keyed in 1998 when a group of individuals attempted to widen the idea of open-source into the corporate world. This movement is the main actor behind widely accepted and used open-source software like Linux, Apache, Firefox, and many more (DiBona et al, 1999). In modern instrumentation, open-source is a fundamental tool to be used, enabling scientists to homebrew their experimental setups to reduce expenses and also collaborate easily on research.

4.3.1 Data Acquisition Unit

DAQ units include many devices that are able to interface with transducers to collect, process, and analyze data. The most common DAQ types can be listed as analog-to-digital converters, digital-to-analog converters, and multiplexers. MUX units allow multiple sensors to be used by a single data acquisition device (Lynn, 2013). DAQ systems mostly have signal processing modules, which prepare the signal from a transducer and amplify, filter, and/or isolate them to be used. In this thesis, a multichannel scaler will be used as a DAQ unit, more information can be found in Chapter 5 **S**ubchapter 5.1.1. In an OSL system, as in all measurement systems, a DAQ unit is a requirement.

4.3.2 Displays and Outputs

For a researcher or user to make sense of the collected data, displays or outputs are essential in an instrumentation system. They can be called the human interface device of any instrumentation system. Their main job is to provide a user-friendly interface to watch or analyze incoming data from an instrumentation system. Many forms exist, from simple light indicators to complicated graphical user interfaces that are capable of even graphing data. Modern instrumentation often uses a combination of screens and interfaces to allow users to connect, control, and improve usability and functionality (Bentley, 2008). Open-source displays or outputs can be considered as a spectrum following light indicators, LCDs to interfaces written with programming languages, for example, python.

4.3.3 Movement System

In instrumentation, a movement system is not a requirement. However, in any modern instrumentation system or mass-fabricated instrument, there exist moving parts. These parts generally consist of motors, with rotary and linear stages. Rotary stages allow stepper motors to be used in angular positioning, thus allowing the user to rotate the component on the stage. Linear stages, however, allow one to use a stepper motor to convert their discrete rotational steps to a linear motion, granting exact displacement (Jones, 2019). These systems enable OSL measurement systems to be loaded with samples externally, and for the samples to be placed under the stimulation light/detector using software, preventing further exposure to unwanted light. Open-source stepper motors and stages allow for the sharing of design and schematics. The open-source community provides firmware, drivers, and 3d printable accessories for these systems to be freely modified, used, and shared. This universalizes the use of high-precision motion systems (Brown, 2021).

4.3.4 Microcontrollers

Microcontrollers are tiny integrated circuits made to execute predefined commands within an instrumentation system. They include a processor, memory, and both analog and digital I/O (input/output) pins combined on either a single chip or a single board. The most notable among them in current times are Arduino (Arduino - Home, 2006), Raspberry (Raspberry Pi Foundation, 2011), and ESP32 (Espressif Systems, 2016). These brands work in an open-sourced ideal, they provide publicly accessible schematics, source codes, and even development software. This enables the widespread use of microcontrollers, among the industry, scientific research, and even hobbyists (Smith, 2021).

4.3.4.1 Arduino™

Arduino is an open-sourced microcontroller board that is currently being widely used due to its simplicity, flexibility, and large community of users/developers. The most common microcontroller Arduino makes use of is the ATMEL's integrated circuit line. As an additional advantage, Arduino comes along with an integrated development environment (IDE), helping with both writing code for it and uploading said code onto the board. Arduino boards are designed to have input/output ports and read inputs from sensors and control outputs for motors, screens, and many more output devices, making them a very versatile option for applications (Banzi et al, 2014).

In scientific instrumentation, Arduino is invaluable for their ability to interface with many sensors and devices to use in larger systems. This ability allows a user to collect and process data post-measurement or even in real-time. Arduino also has an extensive library collection for sensor integration, calculation, and control. The versatility, flexibility, availability, and simplicity of Arduino and its programming make it a strong instrument for scientists to build custom instrumentation systems for specialized use (Monk, 2017).

As was mentioned above, Arduino comprises a large community. This has been possible because Arduino, the company, follows an open-source development model. The community provides a large collection of resources, libraries, tutorials, forums, designs, and schematics, therefore enabling experienced scientists, experimenters, or even beginners to be able to build an instrumentation system from scratch. The open-source ideal encourages use and collaboration, thus allowing users to adapt their ideas to existing designs, enabling them to quickly prototype their ideas for specific needs (Banzi et al, 2014). For example, Maraba incorporated an Arduino in his thesis to construct a CW/LM-OSL system (Maraba, 2017), Shakarji and Gölcük (2022) designed a system to measure Heart Rate and Body Temperature et cetera.

Additionally, in the modern world, Arduinos are affordable and modular, thus having two critical elements for scientific research, in which budget and prototyping are a concern. Compared to privately owned systems, they offer a cost-effective solution and depending on the model of Arduino used, might not even lose functionality. More information and details on the full range of Arduino microcontrollers information on their technical details can be found in the Arduino Store (2024).

4.3.5 Open-Sourced Software in Instrumentation

Since the rise of open-source ideals, many fields, scientific instrumentation included, have been highly improved. Open-source software provides flexibility, customization, and cost-efficiency for new and improved data collection, data analysis, and automation. The most common open-source programming languages are Python and the C family, each having unique qualities that can be tailored to the precise needs of research.

Python, especially, is preferred in scientific studies for its simple and readable syntax, which makes it usable by both experienced and inexperienced programmers. It has a large collection of libraries and frameworks, like NumPy, SciPy, and Matplotlib, that make numerical computations, data analysis and manipulation, and visualization efficient. The versatile nature of Python allows the programmer to integrate it with other languages and software easily, thus making the development of complex instrumentation systems an easy task. For example, PyVISA and PySerial allow the user to integrate Python with National Instruments tools and other COM-Port-based tools such as Arduino, and Raspberry Pi-based instruments (Boztemur, 2018). This allows researchers to automate the DAQ process easily (Millman & Aivazis, 2011).

On the other hand, C and its subsidiaries, C#, and C++, are renowned for their high performance and efficiency in system resource control, which are important for data processing and real-time analysis. C family is generally used in embedded systems, developing the integrated firmware and supporting software where direct interaction with hardware and memory might be crucial. These languages let the user control timing and execution in a very precise manner, like synchronizing instruments to each other or via NTP, which is essential for scientific instrumentation. They also are extremely usable for their support for Windows and Linux-based applications and their integration for .NET Framework, making it a strong candidate for developing interfaces and managing complicated workflows in scientific research environments (Stroustrup, 2013).

The open-source nature of these languages helps create a collaborative environment where researchers and developers can share and build alongside each other, sharing knowledge gained. This helps reduce redundant code, enabling improvements and fixes to be shared across the community quickly. As an addition, the large documentation library, forums, and community support for the said languages help researchers and developers overcome technical difficulties and improve their software tools. This also ensures that scientific software remains transparent, thus helping scientists keep the highest integrity standards intact.

Open-source software, firmware, and libraries like GNU Scientific Library for C and the IPython/Jupyter framework for Python are great examples of how open-source can improve scientific research, both in productivity and in reproducibility (Oliphant, 2007).

4.3.6 Open Science Foundation

OSF is a non-profit organization aiming to promote open science practices and transparency in research. It provides an open-access platform for registering, sharing, and archiving research projects and their data, OSF helps with collaboration and reproducibility of research. OSF can be used to pre-register studies, making sure the methodology and analysis plans used can be independently verified, thus reducing the risk of publication bias (Nosek et al., 2015). As an addition, OSF encourages material and data sharing, allowing other researchers to replicate the experiments, conduct their analysis, and build upon the shared work, accelerating discovery. OSF's existence helps scientific knowledge be accessible, credible, and cumulative, benefiting the scientific community and the society overall (Foster et al, 2017).

By providing an open-access platform for the registration, sharing, and archiving of research projects and data, the OSF facilitates greater collaboration and reproducibility in the scientific community. Researchers can use the OSF to pre-register their studies, ensuring that their methodologies and analysis plans are transparent and can be independently verified, thus reducing the risk of selective reporting and publication bias (Nosek et al., 2015). Additionally, the OSF supports the sharing of data and materials, enabling other scientists to replicate experiments, conduct meta-analyses, and build upon existing work, which accelerates scientific discovery and innovation. The foundation's commitment to fostering an open research culture helps ensure that scientific knowledge is accessible, credible, and cumulative, ultimately benefiting the global scientific community and society at large (Foster & Deardorff, 2017).

CHAPTER 5

DESIGN AND CONSTRUCTION OF THE EXPERIMENTAL SETUP

In the scope of this thesis, a multi-sample TR-OSL measurement system was designed and constructed. The measurement device is constructed around a multi-channel scaler (Easy MCS by ORTEC Inc.) for the measurement of time-resolved photon counting, a photomultiplier tube for light detection, and a pulsed LED-based stimulation unit. The wiring diagrams of the device will be shared in Appendix A. A simplified block diagram of the TR-OSL system can be seen in Figure 5.1.

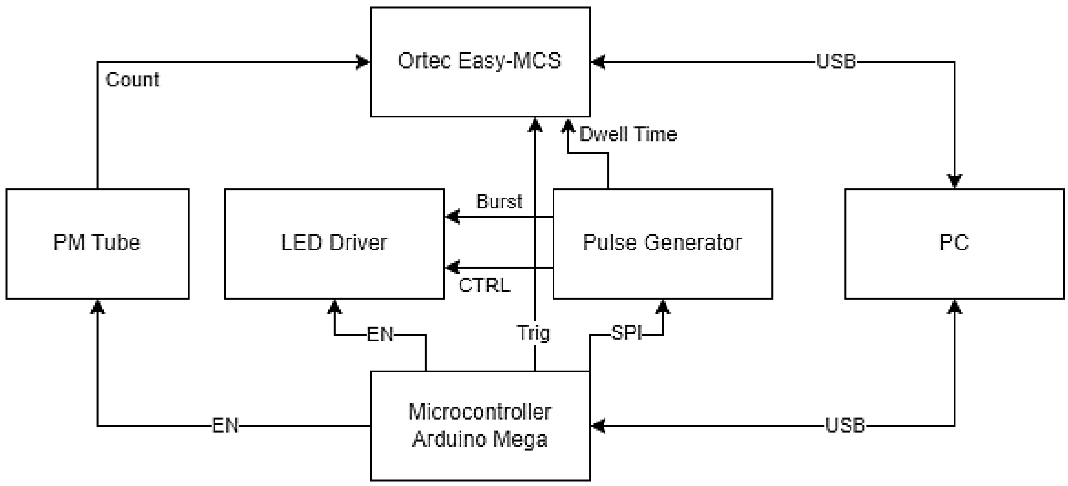

Figure 5.1. Simple Block Diagram of the Experimental Setup

As mentioned in previous chapters, a simple representation of the time-resolved OSL measurement is to stimulate the irradiated sample with a brief pulse and simultaneously record the luminescence output where the luminescence decay following the cease of the pulse is to be used for analysis. The system is able to measure TR-OSL signals from the materials with the help of the multi-channel scaler

(MCS) which can be directly interfaced to the PMT and can measure 65535 data points with the selected resolution and can accumulate signals 2^{32} times. The measured data can be saved in various formats for further analysis. Pulsed stimulation is achieved by using Blue (λ_p~470 nm) and IR (λ_p~850 nm) LEDs and controlled by a pulse generator/LED driver combination. The data collection and stimulation timing are achieved using a pulse generator/microcontroller combination. The microcontroller can take orders from the client PC and acts as a control unit enabling the synchronous working of the pulse generator electronics and MCS. Both the MCS and the microcontroller are connected to a client PC via the USB interface. The device contains a motorized sample changer unit and a motorized tray carrying it for changing the samples and carrying the samples to the measurement positions. All the measurement system is contained in a light-tight box constructed using sigma profiles. The light-tight box is 470 mm in length, 190 mm in width, and 260 mm in height in exterior dimensions. A 3-D model of the system can be seen in Figure 5.2.

Figure 5.2. 3D model of the measurement system and the constructed system

After a basic description of the measurement device, the components of the system can be summarized below. In this system, open-source hardware and software have been used wherever possible to make it as replicable and available to the public as much as possible. All the pieces were modeled using SolidWorks 2023 and design files will be available in the GitHub repository shared in the appendix.

5.1 Essential Components

5.1.1 The Multi-Channel Scaler: Easy-MCS

For the main data acquisition unit, ORTEC's Easy-MCS, USB-connected multichannel scaler was used. A multichannel scaler is a device that records counts with respect to time. At the end of a preset dwell time, it advances a channel, saving data (counts) into a new memory block for the duration of that dwell time. This process is repeated until a pre-determined number of channels have been reached (called a pass). This procedure can be repeated many times to accumulate a statistically meaningful count on the desired channels are reached. The data collection timing can be supplied by either using an internal clock or an external one. In our experiment, external timing is selected where dwell times can be between 100 ns and 125μs. There exists no dead time between channel passes and at the end of a pass. The number of maximum accumulations is 2^{32} and a channel can hold counts as high as 2^{30}. The acquired data is saved into an internal memory that can be transferred to a computer after the measurement, thus making the data transfer seamless (Ortec, 2013).

As mentioned above, the photomultiplier tube can be directly interfaced to the MCS. Photon counting is achieved with the help of an integrated constant fraction discriminator. The threshold levels for the discriminator can be adjusted by software in the range -1.6V to 3V with a resolution of 1.5mV.

The 25-pin I/O connector in the front of the Easy-MCS carries several pins for controlling the device. Inputs placed in the I/O connector are the 1-MHz window discriminator for positive pulse counting, two external TTL inputs for triggering experiment start and stop, a channel advance input in order to increment the bins with respect to an external pulse, a gate in order to start/stop counting independent of the experiment being run or not and a TTL input to enable/disable the start trigger pin. Outputs are a start-out signal, which triggers depending on the experiment start/stop, a channel advance out signal, which generates a pulse for each channel advance, a discriminator pulse output which generates a pulse with every discriminator threshold cross, a midpass out in order to signal when half the channels are complete and a ramp out that provides a voltage ramp to drive external devices.

In the scope of this experiment, the dwell time, pass length, and trigger signals to the Easy-MCS are supplied by a pulse generator circuitry associated with Arduino, which synchronizes data acquisition and illumination. From the I/O connector, the experiment start TTL signal and the channel advance input signal was used. In Figure 5.3, a snapshot of the MCS software (MCS-32) and the front panel of the device can be seen.

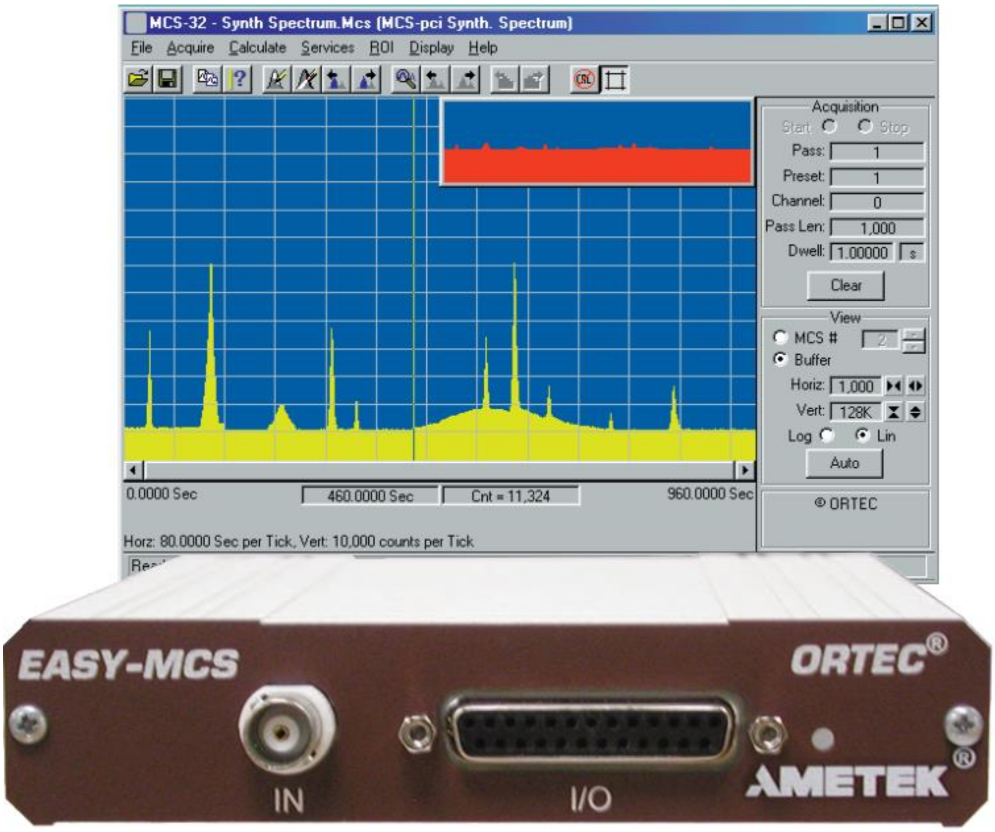

Figure 5.3. Snapshot of the MCS-32 software and the Easy-MCS

5.1.2 The Microcontroller, Arduino Mega 2560

For the electronics, an Arduino (MEGA) microcontroller board (carrying an ATMEGA 2560 chip) was used as the main control system. It controls the drivers of the movement subsystem, a data acquisition unit (MCS-32 dwell times and other control signals), generates the pulses for the illumination system, and a relay to control the power signals for the PMT and the LEDs. The status of the device and messages are monitored on a 24-character 2-line vacuum fluorescent (VFD) display (Noritake CU24025ECPB-U1J VFD).

Using this controller, an instrumentation system was achieved, where the user controls or sets all the experiment parameters. The control board was connected to the client PC via the USB interface taking and sending messages from/to the human interface running on a personal computer. The software running on the microcontroller and the PC will be explained later.

5.1.3 Light Detector (PMT)

Hamamatsu's H12056P-110 type photomultiplier tube module was selected as a detector for light measurement. This PMT module was chosen basically for its quick response time (0.57ns rise time) and the spectral response -ranging between 220 nm to 720 nm with a peak sensitivity at 400 nm- (one should keep in mind that most of the OSL materials have emission between 300 and 400 nm). Besides this, it has a small form factor with dimensions (51x25x30 mm) with an active window area of 8 mm^2. The module operates using a 5V DC supply and contains an internal HV circuitry which can be turned on or off using a TTL signal (Hamamatsu, 2022). HV setting (therefore the gain) can be adjusted using a potentiometer in the range of 0 V to 1100 V.

5.1.4 Stimulation Light Sources

The system constructed employs two types of LEDs emitting one in the visible (Foryard Optoelectronics' FYL-5014UBC1C-1MA-TR13) and the other in the near-infrared (CREE XP-E clone) regions of the spectrum. Almost all of the OSL materials have a response to blue stimulation, therefore blue light is the most common light for stimulation. Feldspars and some materials like ZnS can be stimulated using near-infrared light. Spectral responses of two common minerals (quartz and feldspar) used for dating and retrospective dosimetry studies can be found in Barnett and Bøtter-Jensen's papers (Barnett, 1997, Bøtter-Jensen et al., 1994). These LEDs were specifically chosen for two of their properties: their peak emission wavelength and viewing angle. Their peak wavelengths are λ_p~850 nm, (~ 1.46 eV) (SZGMLED, 2021), and λ_p~470 nm (~2.63 eV) (Foryard, 2019).

Time response studies carried out with these LEDs have shown that they can be successfully driven with pulses of ~150 ns with rise and fall times of ~30 ns. To drive LEDs a two-channel ultrafast MOSFET gate driver (IX4340) was used. These gate drivers have very short (7 ns) rise and fall times and able to supply a peak current of 5A (IXYS Integrated Circuit Division, 2019). To achieve precise timing of the pulse width and match the possible delays two monostable multivibrators for each channel are employed. In Figure 5.4, a simplified structure of the pulse shaping and driver stages is shown. A trigger pulse is recognized by the first monostable multivibrator where a delay Δ can be inserted; the output triggers another monostable which adjusts the pulse width ω.

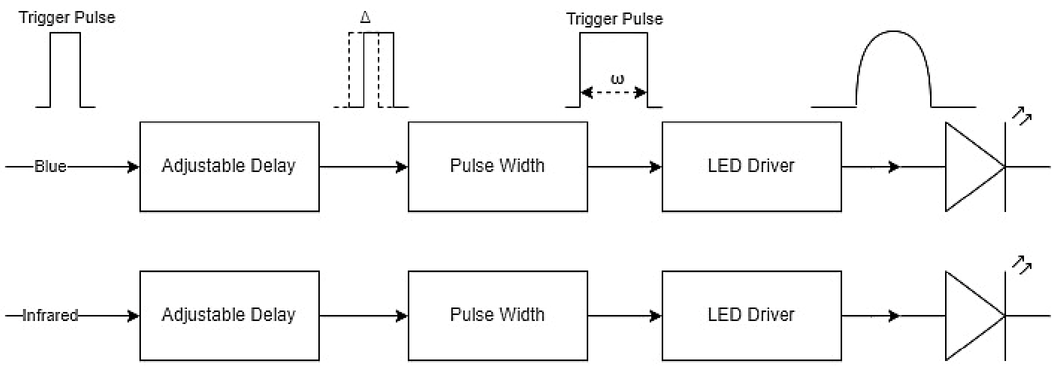

Figure 5.4. Block diagram and expected waveforms of the light source driver PCB

The LED driver had to be calibrated to provide the required sharp electrical pulses, which will be shown in the next chapter. In order to get the correct capacitive load, the Blue LED required a 4.7 nF capacitor parallel to its output, along with a series 15 Ω resistor with a 10 nF capacitor parallel to it. The IR LED also required a 6.8 nF capacitor parallel to its output, as well as a 5 Ω resistor, but with a 2.2 nF capacitor parallel to the resistors. A circuit schematic is given in the figure below.

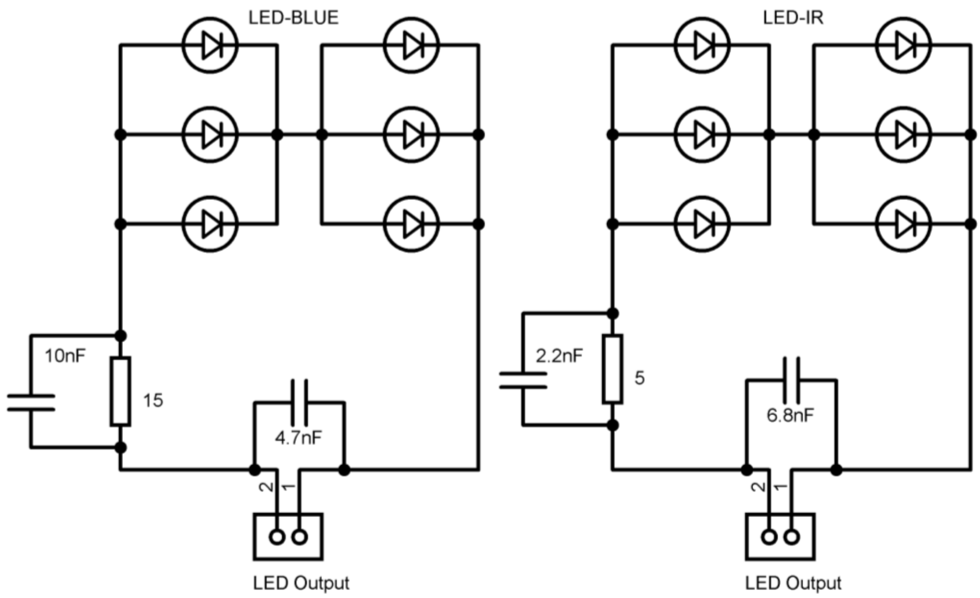

Figure 5.5. Schematics of the LED's

The above-mentioned pulse shaping and driving process is realized by designing a dedicated printed circuit board. The board carries two 74HC123 chips (dual monostable multivibrators) and the dual channel gate driver IX4340. In addition, a 5V fixed (78L05) and an adjustable voltage regulator (LM338T) supply the necessary voltages for the chips placed on the board. The LEDs are driven with electrical pulses of 150 ns width; thus, they can stand much higher currents than the quoted values for increasing the light output. The trigger pulse used for driving the LEDs has periods of 250 ns, resulting in a driving current profile of 150 ns ON and 100 ns OFF allowing the diodes to cool down. The stimulation light is then carried to the sample using a fiber-optical cable.

In the setup, there are 12 LEDs in total, 6 of which are blue and 6 are infrared. These were connected in bundles of 3 each to be parallel, and each parallel bundle was connected in series. With the 15 V output of the gate driver, each blue LED has approximately 165 mA running through it and each IR LED has approximately 600 mA each. This 3-parallel configuration was chosen to maximize illumination while being within the safe limits of the gate driver and to protect each LED from burning out. TR-OSL technique (stimulation and decay expectations shown in Figure 5.6) was also used to overdrive the LEDs with short, strong pulses to protect them from burning out.

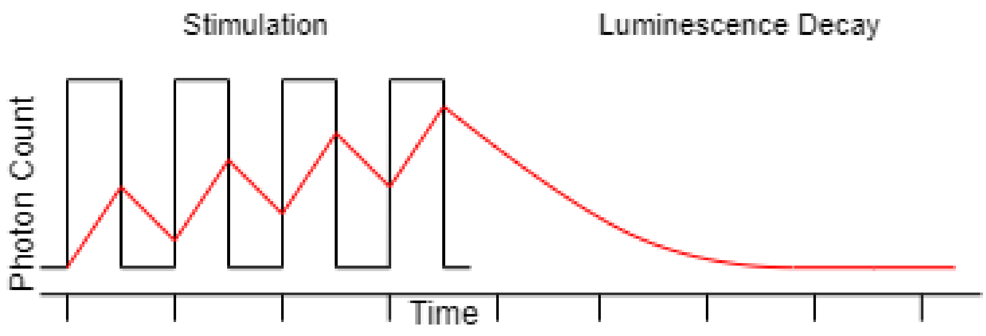

Figure 5.6. Process of the TR-OSL illumination and measurement system

5.1.5 Detection and Stimulation Filtering

Although not shown in the block diagram above, filtering of the detector and also the stimulation light source is very important as the intensity difference between them is very large. Scattered light through the filters should be kept at a minimum.

It was previously mentioned that this study employs two different peak wavelengths of LEDs, λ_p~470 nm for the visible and ~850 nm for the infrared. Although it is not possible to record easily, using conventional spectrometers, emissions from these LEDs do have tails extending several tens of nanometers from the peak wavelength position. These emissions (especially the emission from the blue one) may leak through the detection filters and increase the background signal in the photomultiplier tube. In the worst case, it may mask the weak luminescence emission. To prevent this, a long pass filter (Schott GG420) (SCHOTT Advanced Optics | GG420, 2022) was placed in front of the LEDs; Figure 5.7 shows the emission spectra of the LEDs (taken from the datasheets) and the transmission spectrum (measured using Shimadzu 1201UV spectrophotometer).

For the detection of the luminescence emission, a combination of a UV bandpass filter (Hoya U-340, passband 280-380 nm, 2.5 mm thick (Hoya Optics, 2020)) and a visible bandpass filter (Schott BG 39, passband 320-620 nm (SCHOTT Advanced Optics | BG39, 2022)) was used. The transmission spectra of both filers and their combinations are given in Figure 5.7.

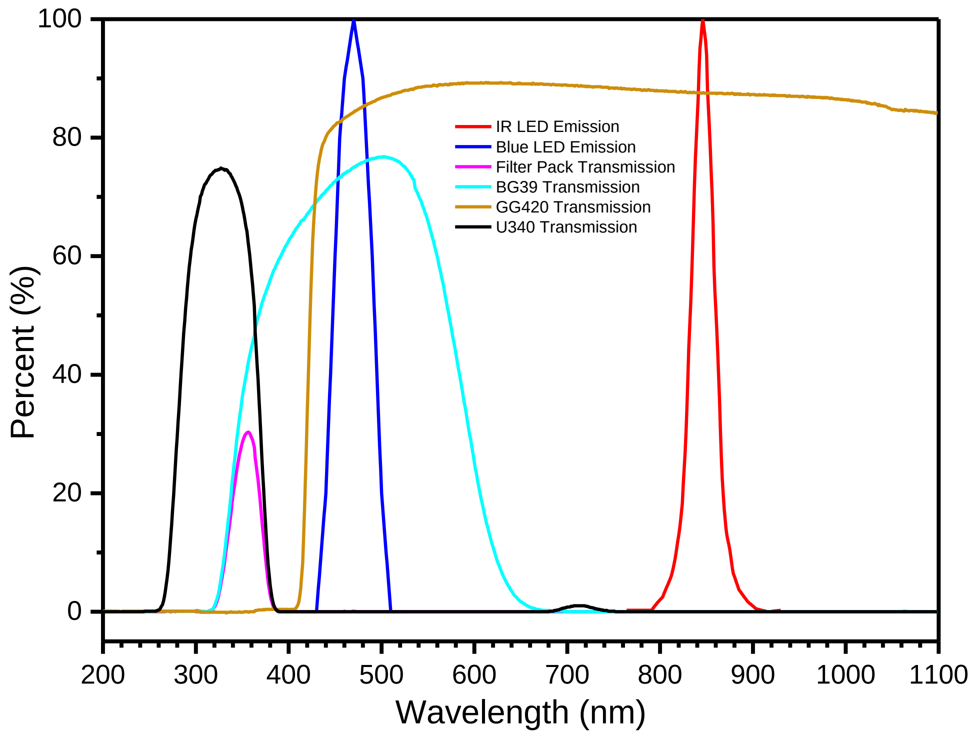

Figure 5.7. Percent Transmission of the filter pack and Spectral Response of the LEDs

5.1.6 Burst Generation

To achieve burst generation, an Arduino shield was designed in the laboratory to carry MCP23017 (16-bit I/O Expander with Serial Interface), 74F525 (Programmable Counter), and Si5351 (8 kHz – 160 MHz Clock generator) ICs and module, respectively. As the user inputs a pulse count and dwell time into the software, the command is sent from the PC to the microcontroller. The microcontroller then relays both commands and sends the number count into the I/O port expander. The port expander converts this number into a 16-bit parallel form and sends the corresponding number to the programmable pulse generator. The pulse generator then takes a trigger signal from Arduino to start the countdown, with respect to the dwell time it receives from the clock generator. More details on how each of these components works can be found below.

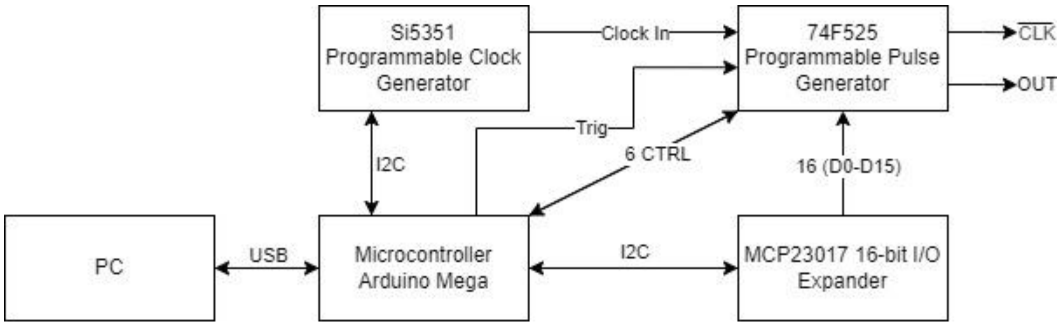

Figure 5.8. Flowchart of the Burst Generator PCB

5.1.6.1 74F525

74F525 is a programmable counter. It contains a 16-bit up/down counter and logic to control both the counter and the outputs. 74F525 can be loaded with a 16-bit number, and supplied with a clock to perform a predefined task depending on which mode it is being used in. It has 8 different modes to be chosen from. In this experiment, this integrated circuit is used in Mode 6, the re-triggerable synchronous one-shot counter mode. In this mode, after the 16-bit number is loaded into the IC, a start signal triggers the counting of clock cycles. Once the trigger signal is received, the output of this IC, normally LOW, stays HIGH until the number is reached. Once the number is reached, the output is brought LOW. This output creates an envelope signal depending on the number of pulses input by the user (National Semiconductor, 1995). This envelope signal is then processed through a logic AND gate with the clock signal received, thus making a predefined shot of pulses possible. This process's logic diagram is shown in Figure 5.9.

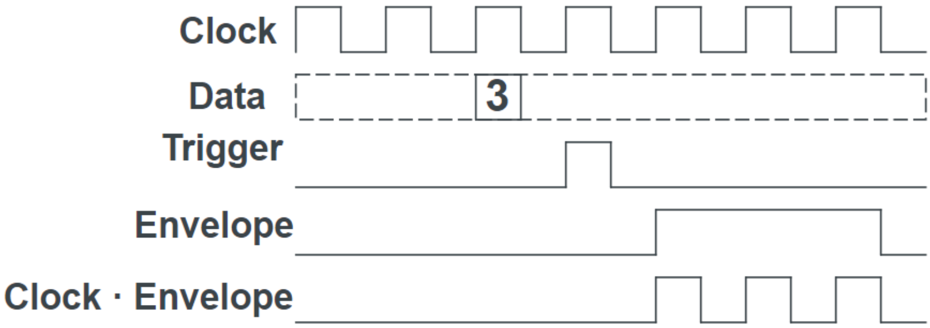

Figure 5.9. Timing Diagram of the pulses

5.1.6.2 MCP23017

MCP23017 is a 16-bit I/O port expander with an I^2C interface (Microchip, 2005). Since it can be configured over I^2C, the pin requirement for a 16-bit number decreases drastically, down from 16 hardware pins on an Arduino to a mere 4, 2 of which are power pins, while the other two are data and clock pins.

In the scope of this experiment, the MCP23017 is used to take a 16-bit number input from the user interface and relay that number to the 16 1-bit input pins of the 74F525. This allows for pin efficiency and ease of use.

5.1.6.3 Si5351

Si5351 module is an I^2C interfaced clock generator. It can generate a pulsed clock signal ranging from 8kHz to 160MHz. Its frequency is very precise; its claimed error is 0 ppm (Adafruit Industries, 2011). The module used in this experiment has the capability of generating 3 clock signals at once, of which all are used.

This clock generator was used in this experiment to synchronize every other device to a single clock signal. 74F525 uses the clock generated by this module in order to create the burst pulses for the light source and trigger signal for the data acquisition unit. It is programmed within the user interface and is used as the main clock signal of the experiment.

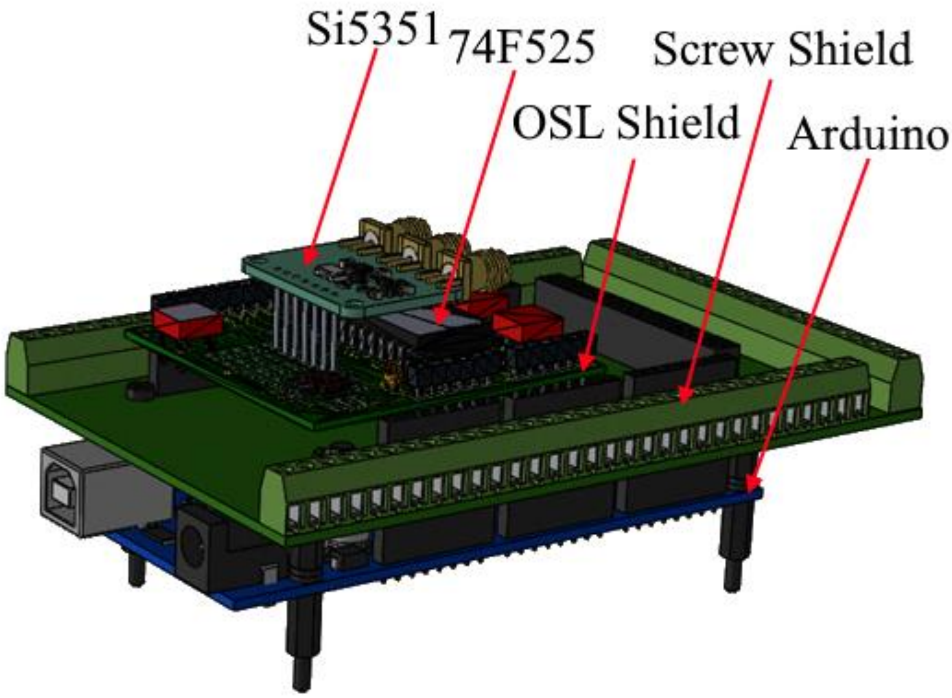

Figure 5.10. Arduino Mega, Screw Shield, OSL Shield with 74F525 on top, Si5351

5.2 Auxiliary Components

5.2.1 Fiber Optic, Lens and Mirror Holder

To hold the sample side of the fiber-optic cable, as well as the focusing lens and the parabolic mirror, a 3D printed part was designed. This part fits on the filter holder on top of the PMT and holds the fiber optic cable header in place and the focusing lens for the light output. It was designed so that internal reflections and the interference of mechanical parts were kept to a minimum. To the other end of the holder, the focusing mirror is placed to collect the emitted light from the OSL and direct it into the PMT module. The Interior of the holder was painted with liquid chrome paint, further improving reflectance and increasing the chance of a photon arriving at the PMT. A wireframe and isometric view of the holder can be seen in Figure 5.11.

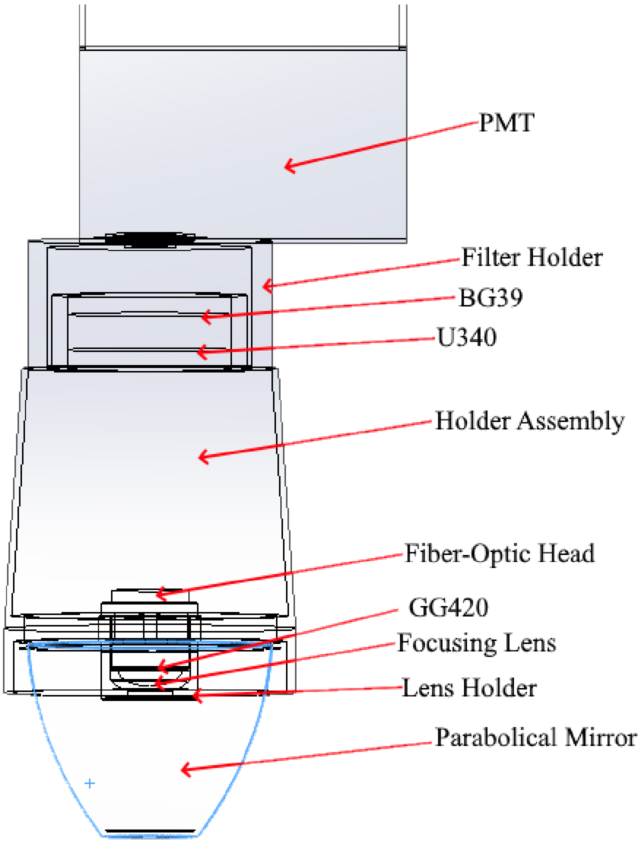

Figure 5.11. PMT, Filters, and Fiber Optic Cable Holder

5.2.2 Linear Stage

The linear stage has been hand and machine-crafted to fit in the experimental setup. The stepper motor used within the linear stage is a Soyo SY35ST28-0504A, a commercially available NEMA14 bipolar stepper motor with a 1.8° step angle (200 steps per revolution), consuming 500mA per coil at 10V (Pololu, 2013). A commercially available end-stop optocoupler is used to mark the endpoint of the system. At the very front of the linear stage, the light-tight door is connected to the stage with L-shaped aluminum pieces. The technical 3D drawing of the linear stage can be found below.

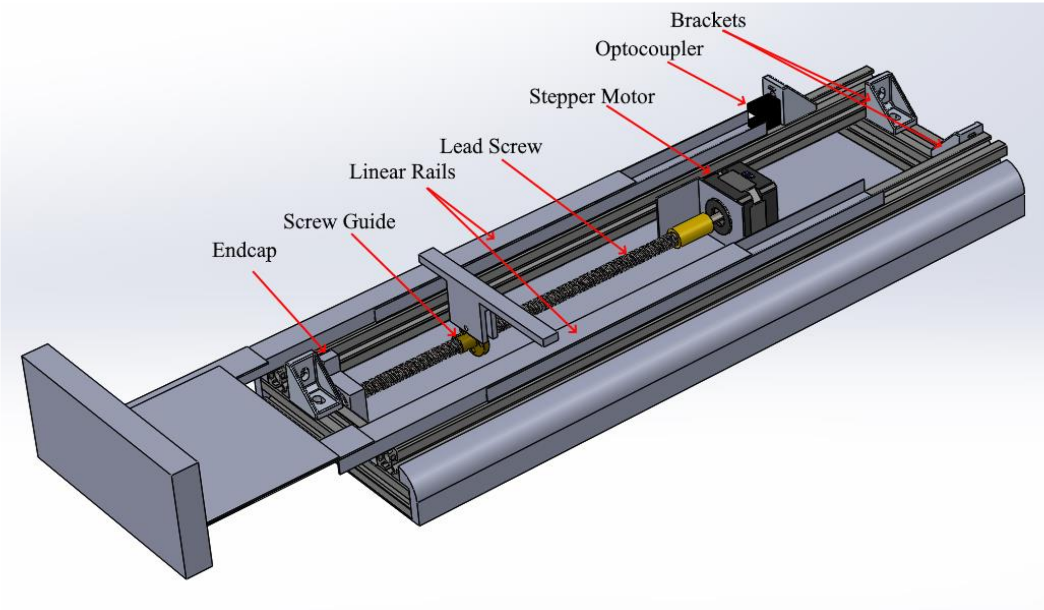

Figure 5.12. Linear Stage

5.2.3 Sample Changer (Rotary Stage)

The rotary stage is fixed onto the linear stage for it to leave the chamber for sample loading. Its main purpose is to carry and rotate samples under testing. The stage includes 2 brackets for rail mounting, a stepper motor (Sanyo Denki 103-4902-0740), a main body, a rotating wheel with 8 sample holders, and a 3D-printed sample separator. The Stepper motor is a NEMA23 package, 0.9° per step motor (Sanyo Denki, 2001). The sample separator is a 2 mm-thick piece that may prevent/reduce cross-illumination of neighboring samples.

A 3D visualization can be found in Figure 5.13.

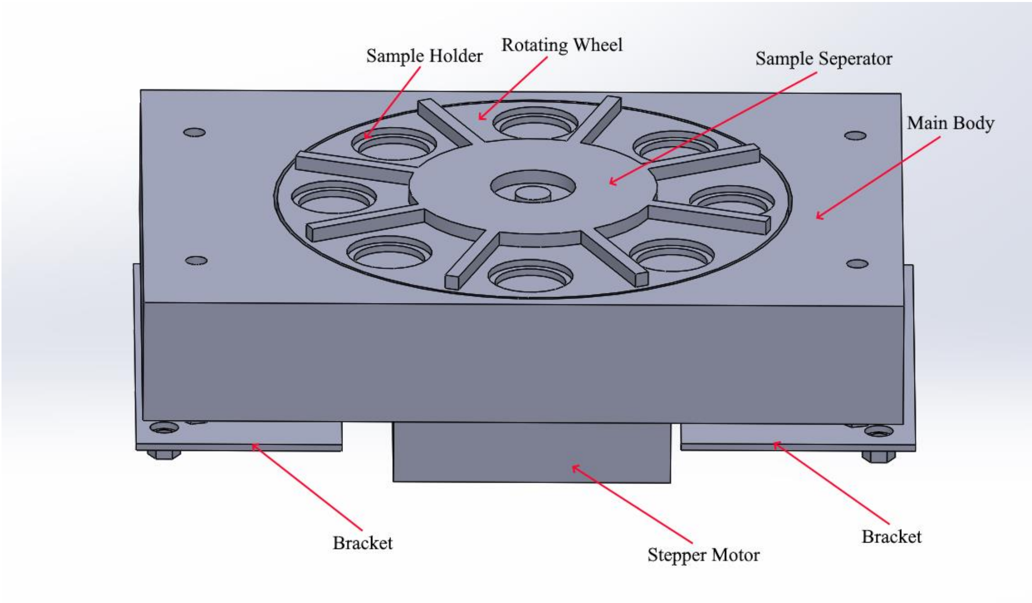

Figure 5.13. 3-D model of the Rotary Stage

5.2.4 Light Source Holder

5 mm Blue LEDs (λ_p~470 nm) and 3.5x3.5 mm surface-mounted-device (SMD) infrared LEDs (λ_p~850 nm) mounted on a 12 mm aluminum PCB were selected as light sources as will be explained in the light source subchapter. To carry light from the LEDs and focus properly on the sample, a commercially available plastic fiber optic cable and its assembly were used. To properly couple the LEDs to the fiber optic cable headers, a 12-hole assembly was used to hold all the LEDs and headers securely in place. The light source holder can be seen in Figure 5.14.

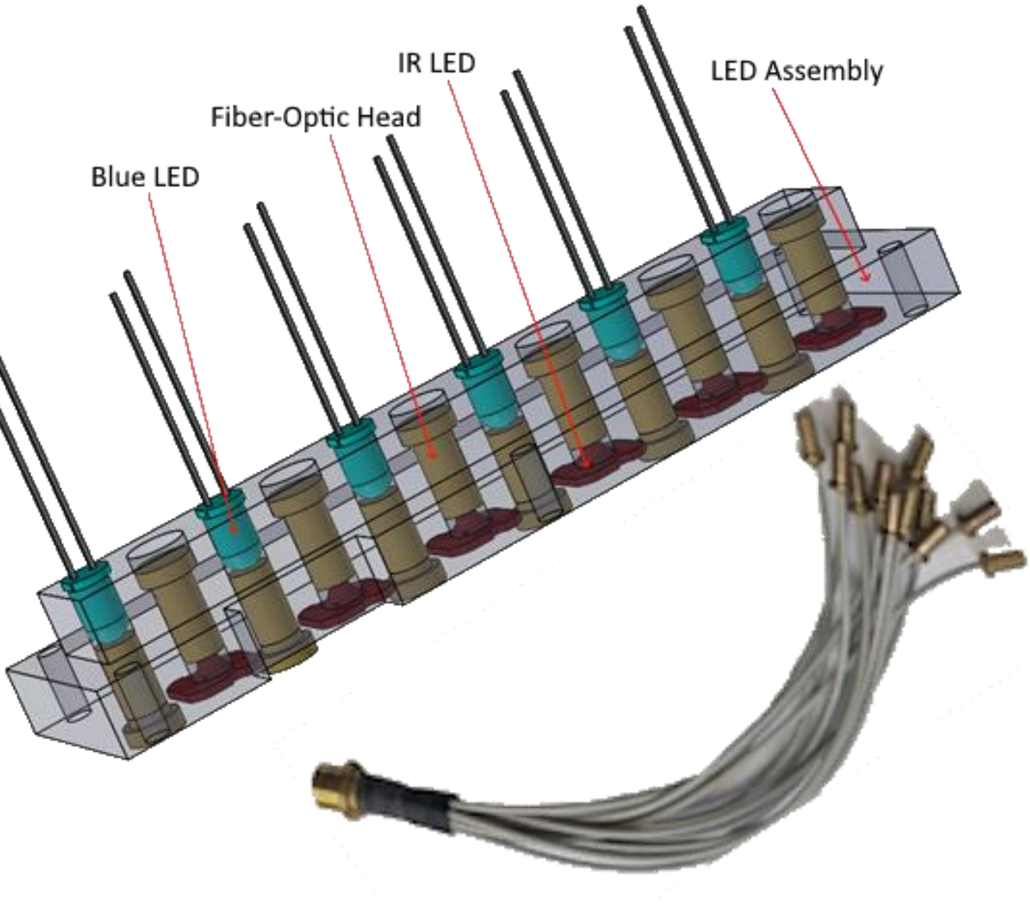

Figure 5.14. Light source and fiber-optic cable holder with Fiber-Optic Cable

5.2.5 Motor Drivers

As the stepper motor driver, Makeblock's Me 2H Microstep driver was found to be suitable. It is a 2-phase hybrid stepper motor driver. It has 8 options for both microstepping and output phase current (Makeblock, 2016). In this experiment, the driver is controlled at half-current mode outputs 1.415A phase current and is configured to be used as 25600 pulses per revolution. This configuration lets the controlled stepper motors, both in the linear and rotary stages, be used without vibration or noise, and minimizes the chance of skipping a step.

5.2.6 Power Supply

For the main power supply, XP Power's SDS120PS24B 220V-to-24V switching power supply was used. Its small footprint, the existence of mounting holes, ±1% output noise, and 120W power rating were the reasons this component was chosen (XP Power 2019). This power supply also has two commercially available LM2596 DC-DC (LM2596 Data Sheet, TI.com, 2012) step-down converters mounted on top to supply the system with 5V, 12V, and 24V simultaneously.

5.2.7 Power Distribution PCB

As 5V, 12V, and 24V were all needed in many connections inside the experimental setup, a power distribution PCB was designed. This PCB is made up of a perfboard and terminal blocks and has 5V and 12V inputs and outputs as well as a common ground input. This PCB allows for at least 6 of each 5V and 12V devices to be connected to power.

5.3 Software

The software of this research has been completely written by open-source programming languages, using free IDE's and under the Creative Commons CC4 License. Therefore, any reader can replicate & improve upon this research, as is the ideal of open-source software. To achieve this, an open-sourced hardware platform with again, an open-sourced software framework was needed. The codes used within this research will be shared in Appendix C in proper syntax copyable form as well as an online storage service link.

As mentioned above, the TR-OSL measurement system consists of a commercial multichannel scaler (Easy-MCS) to acquire luminescence data and a microcontroller to control the hardware required for motion and luminescence stimulation, as well as timing and synchronization. Arduino was chosen for this research as the microcontroller. Amount of pin requirements urged the use of an Arduino Mega. The multichannel scaler has its software, which is to be initialized using the TR-OSL human-interface software running on a personal computer. The interface software also communicates with the microcontroller for the synchronization of the MCS with the rest of the hardware.

5.3.1 Python

For the human interfacing software, Python was used. The main purpose of the interface is to be a translator between the human using the software and the machine processing the commands. There are two tabs in the main interface, one named "Guidelines" (Figure 5.15) and one "Controls" (Figure 5.16). The user can swap between these tabs at any time during an experiment as it does not interfere with the experiment commands. The guidelines tab carries basic information and reminders for the user on how to run the experiment and how the parameters should be entered. Controls tab is the main experiment software and the buttons follow a certain predefined flow. This flow is visualized in Figure 5.17.

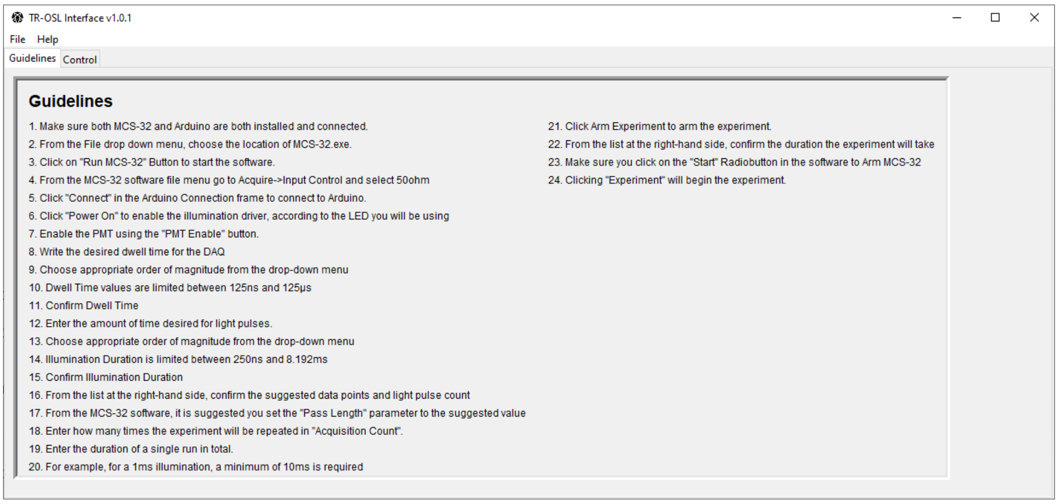

Figure 5.15. Guidelines Tab of the User Interface

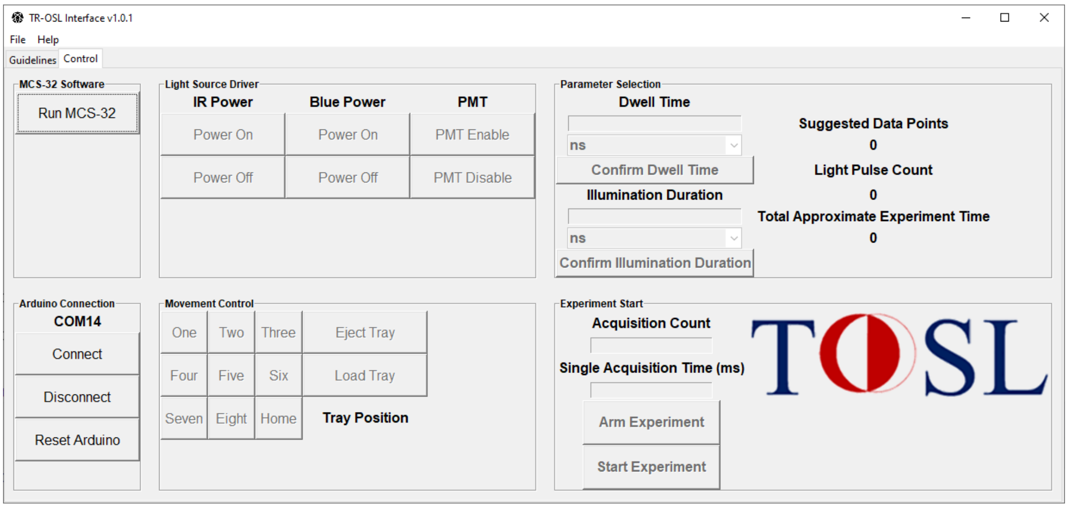

Figure 5.16. Controls Tab of the User Interface

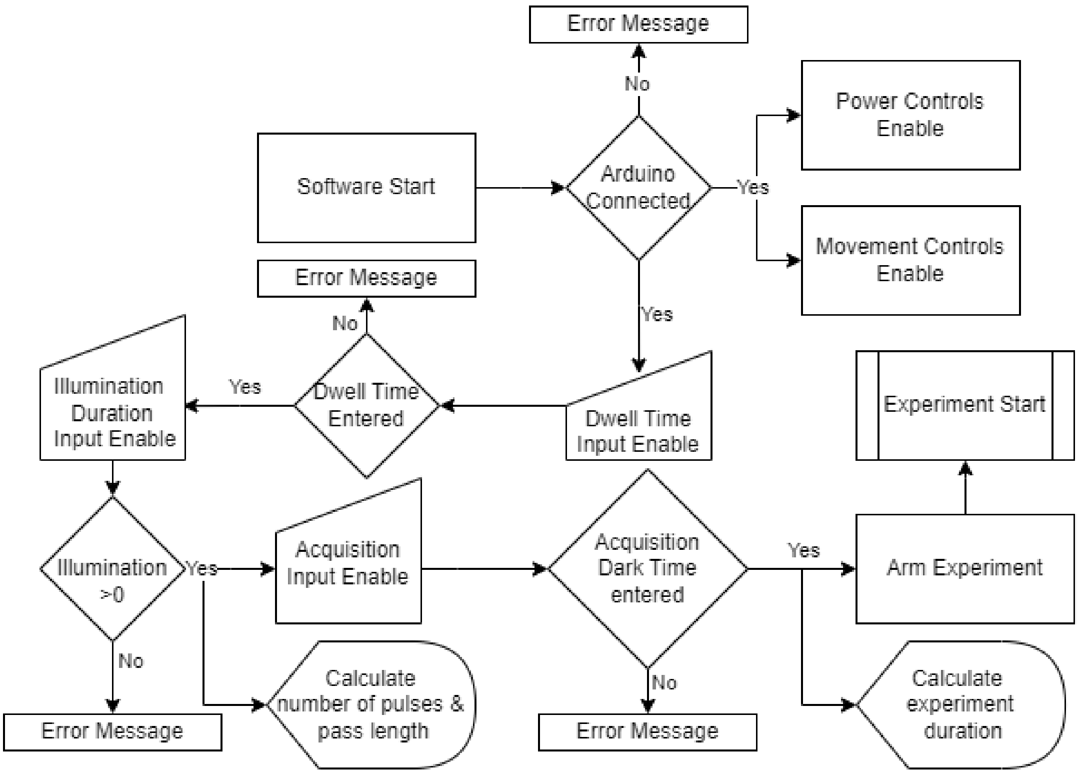

Figure 5.17. Main Flowchart of the Experiment Human Interface Software

As the software is started, most of the normally clickable buttons are set to be grayed and not clickable. Once the computer is connected to an Arduino, power controls, movement controls, and dwell time controls become available for use. After the initialization of the software, only the dwell time textbox and the dwell time drop-down menu will be available for use. If a correct dwell time is entered (between 125 ns and 125 µs) and Confirm Dwell Time has been clicked, Illumination Duration controls will be enabled. If the illumination time input yields a calculable result (between 1 and 32768) the interface will display the number of pulses and the maximum number of bins the DAQ software requires for data. As illumination time is selected, acquisition count and the experiment single acquisition time textbox will be enabled. The acquisition count textbox controls the number of passes, in other words, the amount of repeated acquisition runs to be done. Single acquisition time specifies the time required until another acquisition pass begins, thus is directly correlated with the duration of a single pass and the dark time within a single pass. If, for example, one needs to measure an exponential decay from the illumination of 1 ms, the user should enter at least 10 ms (at least 10 times the illumination duration). The Arm Experiment function will set the selected parameters as the experiment parameters. As the Arm Experiment button is pressed, the interface will update and show the expected experiment runtime. Start Experiment will initialize the experiment with the selected parameters. Once the experiment is done, the VFD Display in front of the setup will notify the user (by writing “Experiment Done” and showing the PMT Bias HV), and other parameters can be loaded or another sample can be tested. Unless Arduino is disconnected or reset using the button from the interface, the experiment will run until the parameters are satisfied, ignoring any new commands issued. There are also two buttons not visualized in the figure “Disconnect” and “Reset Arduino”, which disconnect the Arduino from the interface and reset the Arduino, respectively.

Once any of the power controls are turned on, the respective button is disabled and cannot be clicked again, as well as the same case for power being turned off. In the movement controls, once a movement command has been selected, the software will process that command, show the current position within the interface, and disable that button from being re-clicked. Once the command is completed, another movement command can be given.

There is a minimal number of functions unrelated to Arduino within the software. The “File Menu” (within the interface, see top left) selection “MCS-32 Location” asks the user to direct and point to the location of MCS-32.exe, the main software of the ORTEC Easy-MCS. Once this location within the operating system has been defined, the Run MCS-32 button in the interface will start MCS-32, thus helping this software to combine both controlling the experiment and acquiring data. Developer Options is some sort of a debug system and can be used to see if the burst generator subsystem works as intended. A rough function diagram of the software menu can be found below.

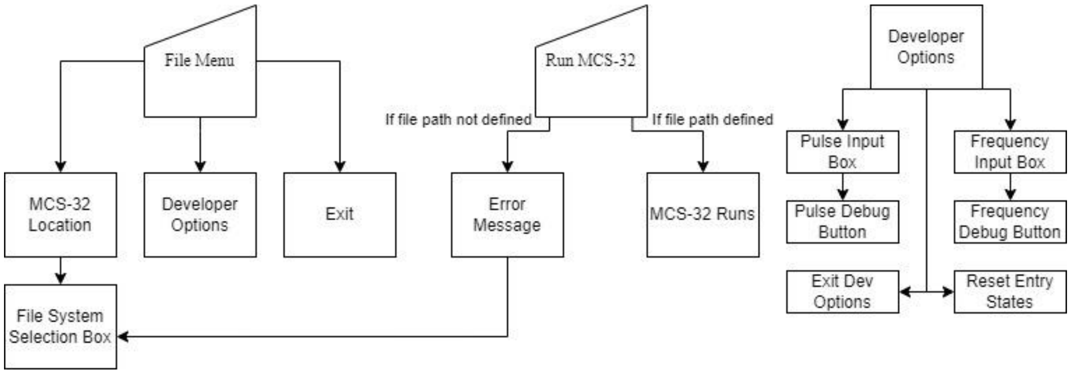

Figure 5.18. Interface Function Diagram for the scripts on MCS-32 control and Developer Options Scripts for Debug Purposes

For a microcontroller to be used from a computer without having to use an IDE, an underlying framework must be used. Python’s PySerial package is a useful tool to interface devices connected from the serial port with a personal computer (PySerial, 2013). Another vital package used is the Tkinter library. This library allows the user to create a user interface as well as to make a flexible background for it (Tkinter, 2012).

In the interface, buttons are connected to the commands Arduino can parse. All buttons except the MCS-32-related ones and Arduino connection buttons issue the commands given in Table 5.1. to the Arduino. In the following figure, Arduino-based Python functions can be found. After the microcontroller is connected, the blocks shown in the figure will send commands to the Arduino according to either predefined parameters or variables entered. In the table, the first 4 commands are visualized in the far right of the figure. The following 6 commands are shown in the top left, and the remaining commands are shown in the bottom right.

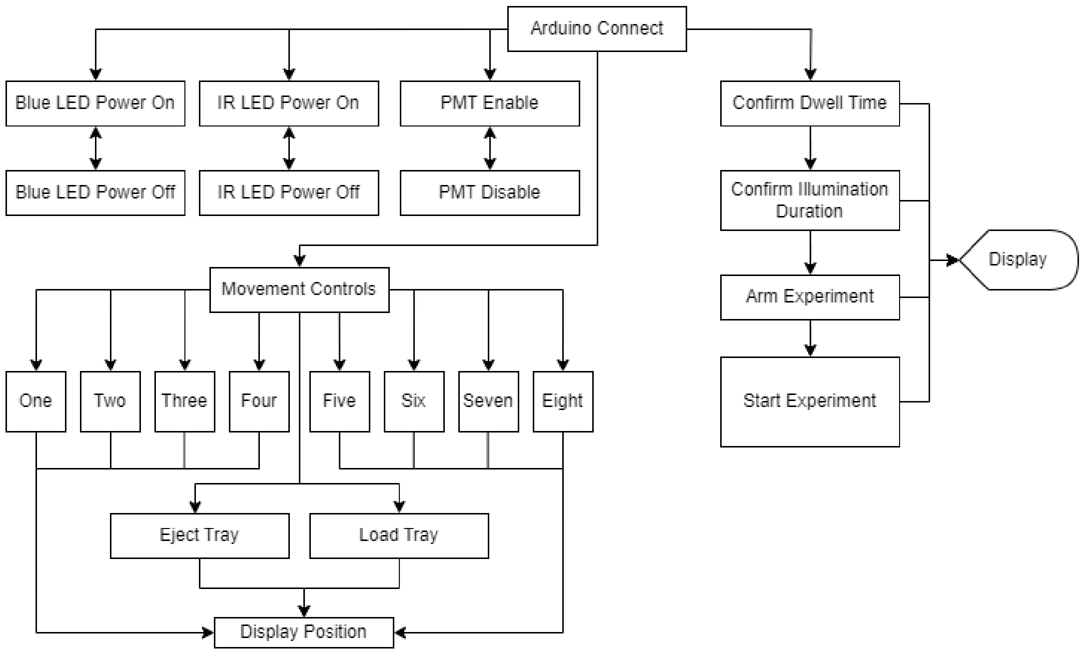

Figure 5.19. Function Diagram for the Python Scripts to control Arduino

5.3.2 Arduino

As mentioned above, an Arduino microcontroller board (MEGA) was utilized for controlling almost everything except luminescence data acquisition. Therefore, the software required for the other tasks such as mechanical motion, sample changing, preparation of timing signals (both for the MCS and the illumination system), driving the LEDs, displaying device status, etc. is embedded within the Arduino. Thus, the microcontroller parses serial inputs in an infinite loop, and once a required command is received from the serial port, runs the corresponding task for that command. A function diagram of the Arduino functions is given in Figure 5.20 and a list of the commands and their respective meanings which Arduino can parse are given in the following table.

Table 5.1 Commands Arduino can parse and their respective functions

“P X^*”	Sets the number of pulses
“F X*”	Sets Dwell Time integer multiple
“L”	Arms Experiment
“START X^* Y^*”	Begins Experiment
“IRPOWER”	Enables IR stimulation
“IRPOWERO”	Disables IR stimulation
“BLPOWER”	Enables Blue stimulation
“BLPOWERO”	Disables Blue stimulation
“PMTEN”	Enables the PMT Gate
“PMTDIS”	Disables the PMT Gate
“HOME”	Homes the Rotary Stage
“ONE”	Sets the Rotary Stage to Sample One
“TWO”	Sets the Rotary Stage to Sample Two
“THREE”	Sets the Rotary Stage to Sample Three
“FOUR”	Sets the Rotary Stage to Sample Four
“FIVE”	Sets the Rotary Stage to Sample Five
“SIX”	Sets the Rotary Stage to Sample Six
“SEVEN”	Sets the Rotary Stage to Sample Seven
“EIGHT”	Sets the Rotary Stage to Sample Eight
“LHOME”	Homes the Linear Stage
“EJECT”	Ejects the Linear Stage

*X and Y are integer numbers. For P, $X>1$ is required. For F, a frequency should be entered with $8,000<X<160,000,000$. Dwell time will be calculated as

1/X. For START, first parameter (X) should be sent as the acquisition count, and the second parameter (Y) should be sent as the number of data points times dwell time (in ms).

As Table 5.1 indicates, 21 functions were written in Arduino, with 11 of those functions (Sample 1-8, Load Tray, Eject Tray, Sample Tray Home) for controlling the movement of the linear and rotary stages. 6 of the functions (Blue LED Enable/Disable, IR LED Enable/Disable, PMT Enable/Disable control the power of the illumination system and the high voltage of the detector. From the 4 other functions, Receive Illumination Pulse Number ("P X") controls the number of pulses from the LED, Frequency Control ("F X") sets the dwell time for the DAQ, Latch Number ("L") arms the experiment and Start ("START") begins the process. There's also an extra function called whilst the Arduino is waiting for a Serial Input in an infinite loop, a moving average function, which continuously reads the PMT's HV monitoring output, takes a moving average of these values to reduce noise and displays on the VFD Screen. This function does not need to be triggered. These functions are shared in Appendix C.

To parse the commands and run corresponding functions, several Arduino libraries have been utilized. Most importantly, the SerialCommand library was used. This library allows the Arduino to run all functions in parallel, as opposed to the Setup to Infinite Loop characteristic (Rado, 2013). With this library, it is possible to predefine several serial command inputs to call matching functions, and once that command is received, Arduino runs that function once and starts waiting for another command. With this library, a parallel function tree was possible to implement in this thesis. The Wire library was used to make communication between the microcontroller and I^2C devices possible (Wire - Arduino Reference, 2022). This is a built-in library for the Arduino IDE and can be accessed by including its header file in the software. Likewise, the LiquidCrystal library has been employed in this software, another built-in library (LiquidCrystal - Arduino Reference, 2021). This library makes implementing the VFD Screen into the experiment system possible by providing a parallel data connection as required.

The movingavg library (which takes the moving average of an analog input pin) was also included to reduce noise level display on the PMT High Voltage feedback loop (MovingAvg - Arduino Reference, 2020). To generate the frequency required, the Si5351 breakout board has been used. This board is paired with a library that can be simply downloaded and allows for this board to be used at the researcher's will (Adafruit Si5351 Library - Arduino Reference, 2021). To control the movement system of this experiment, the AccelStepper library was used. The main advantage of this library is the fact that it has built-in acceleration control and is very easy to configure, therefore eliminating most of the issues of using other libraries, such as acceleration control and jerk (AccelStepper - Arduino Reference, 2021).

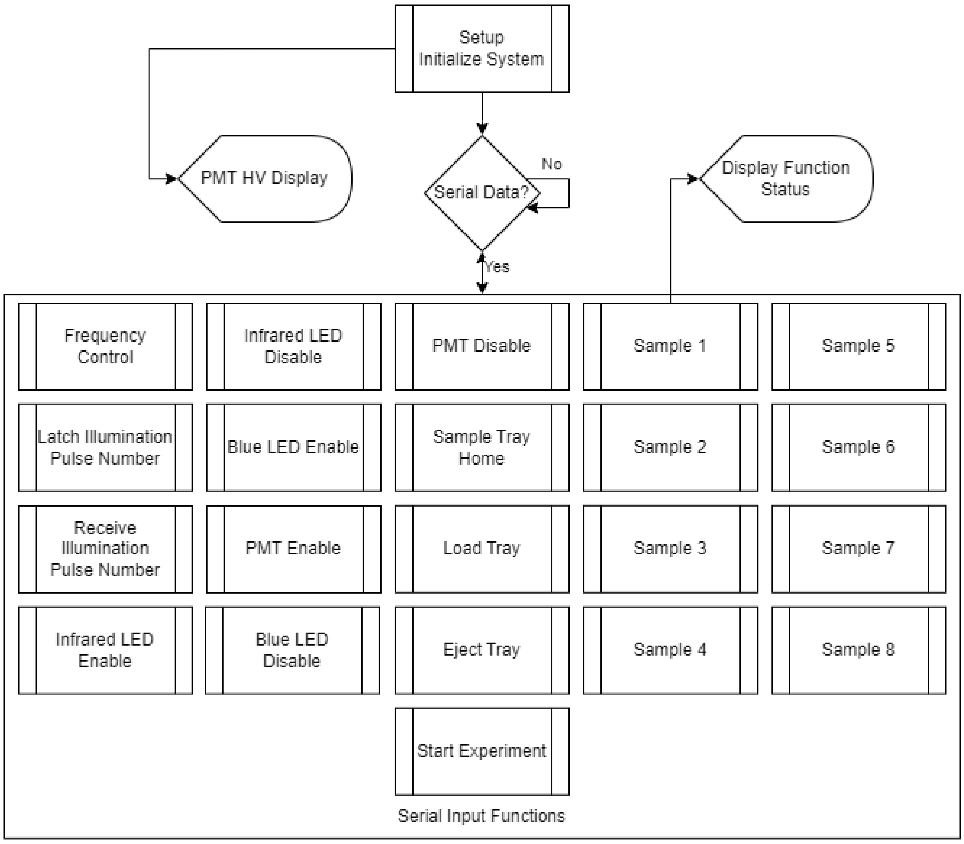

Figure 5.20. A representation of the Arduino Function Diagram

CHAPTER 6

RESULTS

As summarized in the previous chapter, a TR-OSL measurement system was constructed. Before experimenting with the device individual parts of the device should be thoroughly tested. Here, starting with the multichannel scaler (which is the only component that we have not full control) testing of the parts of the measurement system is summarized. The Easy-MCS comes with (and should be used) its software; however start/stop of the acquisition, dwell timing, and accumulation abilities are controlled externally. These timing signals and the burst signal used for driving the LEDs are supplied using a burst generator which is governed by the Arduino MEGA microcontroller board. All tests were performed using the Tektronix TBS 1052B-EDU 50MHz, 1GS/s oscilloscope (TBS1000B and TBS1000B-EDU Series Digital Storage Oscilloscopes User Manual, 2012).

6.1 Tests concerning the Timing/ Burst Generation System

In the previous chapter, the design criteria and approach for the timing signals and burst generator were mentioned. The burst generator is designed around a precise/accurate clock generator (Si5351 and a precise programmable counter chip (74F525) working in the monostable multivibrator mode. The clock generator Si5351 has three output channels, and one of them was used for dwell timing for the MCS, where the width of the dwell/integration times are adjustable between 100 ns and 250 μs. The lower limit was set by the MCS and the upper limit was determined by the clock/burst generator. Examples of dwell clocks can be seen in Figure 6.1

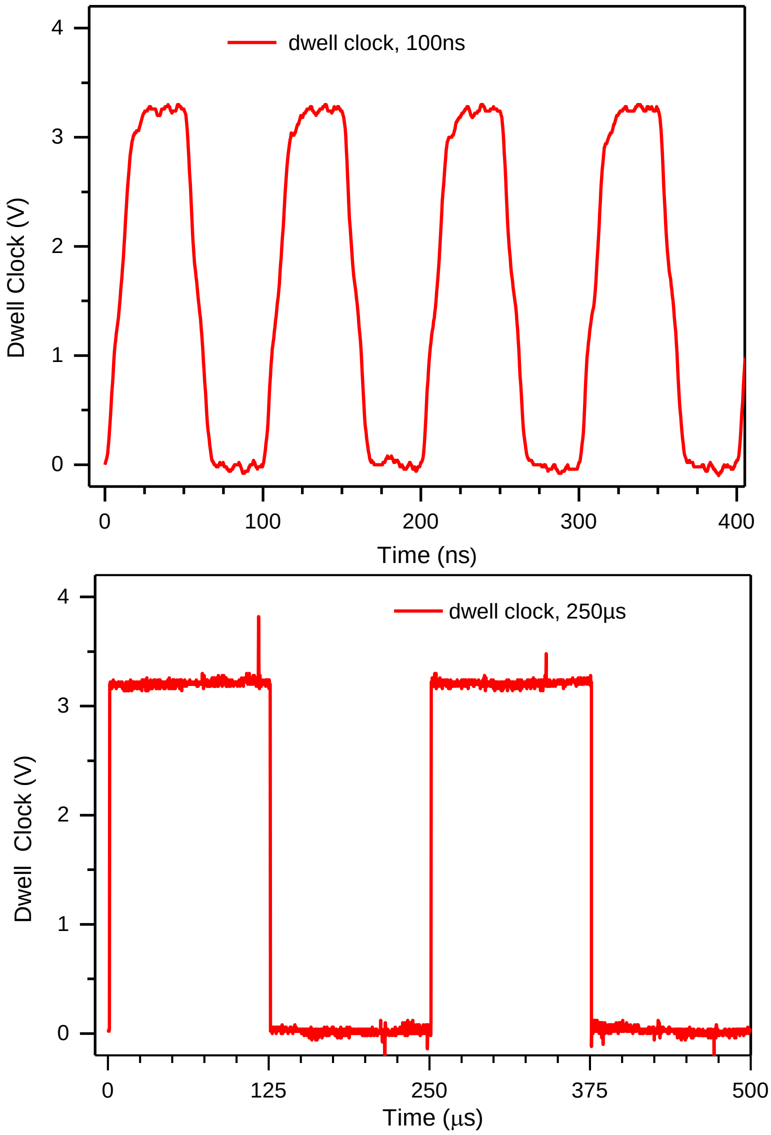

Figure 6.1. Examples of dwell clocks generated (upper plot:100ns, lower plot: 250µs)

As can be seen from the figure, the burst generator circuit is able to generate a dwell clock signal ranging from 100ns to 250µs with no discernible noise.

6.2 Light Source Burst Generator

The stimulation light source also uses the clock pulses generated by Si5311 (another channel). Due to the restrictions in the response times of the LEDs and driving circuitry the period of pulses driving the LED was fixed and limited to a period of 250 ns (150 ns ON, 100ns OFF). The reason for choosing these numbers was the current limitations of the LEDs in use, the LEDs were overdriven to increase the light output without damaging them. An example of this clock signal is shown in Figure 6.2.

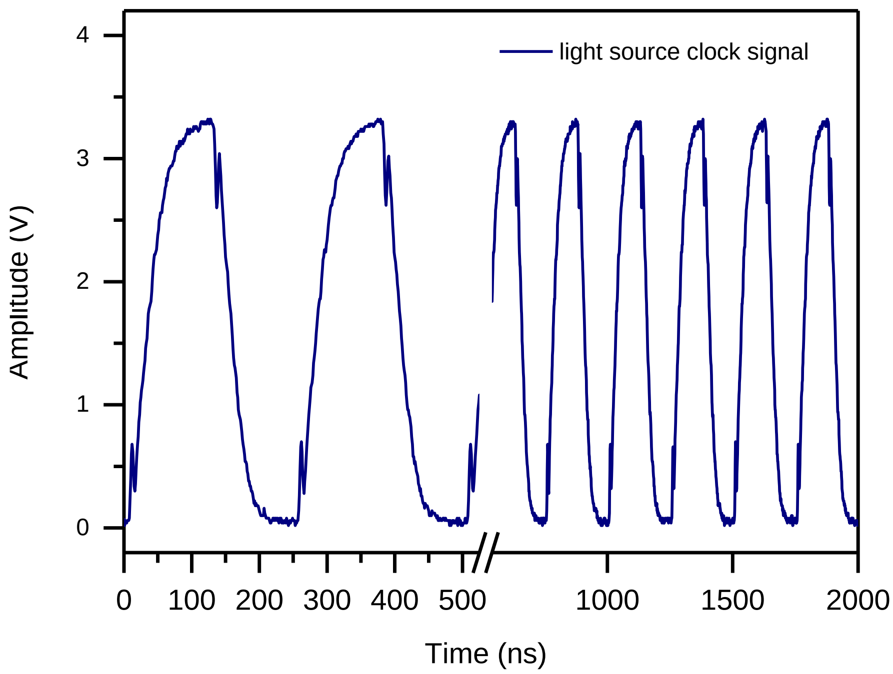

Figure 6.2. 250ns light source clock signal

Desired pulse widths were obtained by increasing the number of pulses (250 ns wide), thereby creating a pulse train for stimulation. The length of the pulse train was adjusted by creating an envelope signal using a programmable monostable multivibrator (one-shot) based on 74F525. This envelope signal is logically ANDed with a light source clock signal to generate an exact number of clock pulses. This output is used by the LED driver circuitry to generate the desired stimulation light pulse train.

As mentioned in the previous chapter in the LED driver circuitry the pulse train generated is first reshaped (150ns ON 100 ns OFF) using monostable multivibrators and conveyed to a gate driver for driving the LEDs. The setup is capable of generating pulses up to 32768 (2^{15}). Each illumination pulse has a period of 250ns; summing up to a total of 8.192 milliseconds of illumination at maximum. The electrical outputs of the burst generator for the infrared LEDs have rise and fall times of 3.6ns and 5.8ns, and the blue LED has rise and fall times of 4.12 ns and 6.23 ns, respectively. The output of the gate for the two wavelengths (λ_p~470 nm and λ_p~850 nm) are given below for single and multiple pulses are shown in Figure 6.3.

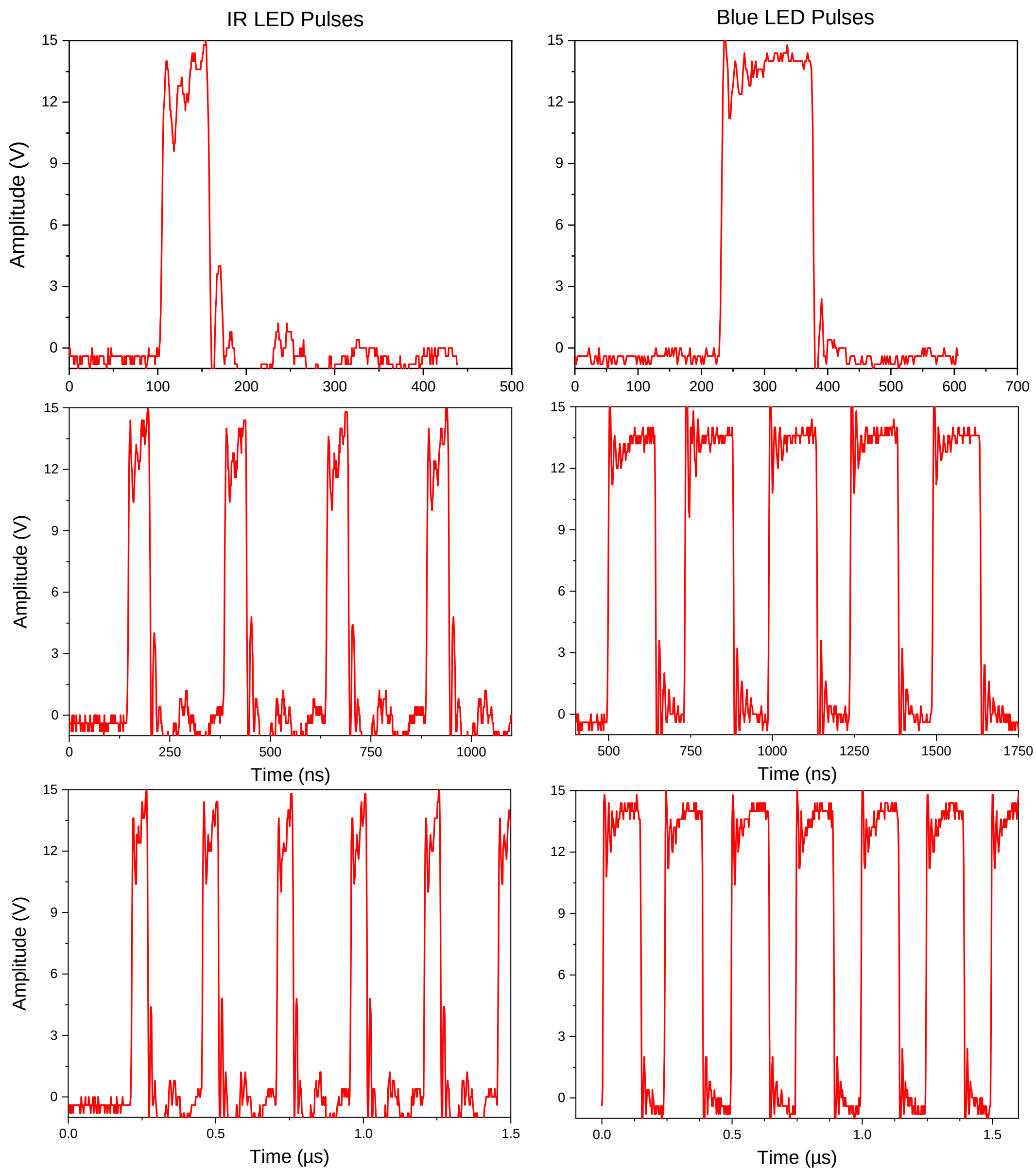

Figure 6.3. Sample Electrical Pulse Signals for both LEDs

6.3 Stimulation System (LED Output)

After the burst generation was achieved, the actual LED outputs were (rise and fall times and illumination profiles) characterized. This included the LEDs themselves, and the LED Power drivers. As mentioned in the previous section, the LED driver contains two cascaded monostable multivibrators (in two channels) and a fast two-channel MOSFET gate driver IC for driving the LEDs. Experiments have shown that the gate driver/LED combination for each channel is able to respond to a pulse width of 150ns. Keeping in mind that the LEDs are overdriven, an OFF time of 100ns had to be inserted to let a cool-down period. Thus, in driving LEDs with multiple pulses (to achieve longer illumination periods) a clock period of 250 ns was chosen. All stimulation profiles are to be integer multiples of 250ns. A 150ns pulse is obtained by using a fast/precision monostable multivibrator (namely 74HC123). Before this multivibrator, another multivibrator is used for inserting an adjustable delay before the pulse. It was found best that both LEDs were to be set on a 150ns fixed pulse-width with a 100ns delay.

Single voltage pulses applied to the blue and IR LEDs are given in Figure 6.4. As seen from the figure, they do have different time responses as the LED technologies are different resulting in different junction capacitances. The illumination profiles of the LEDs were measured by using a fast photodiode, which has a spectral sensitivity of 200-1100 nm and a rise time of 1 ns (Thorlabs - DET10A/M SI Detector, 2015).

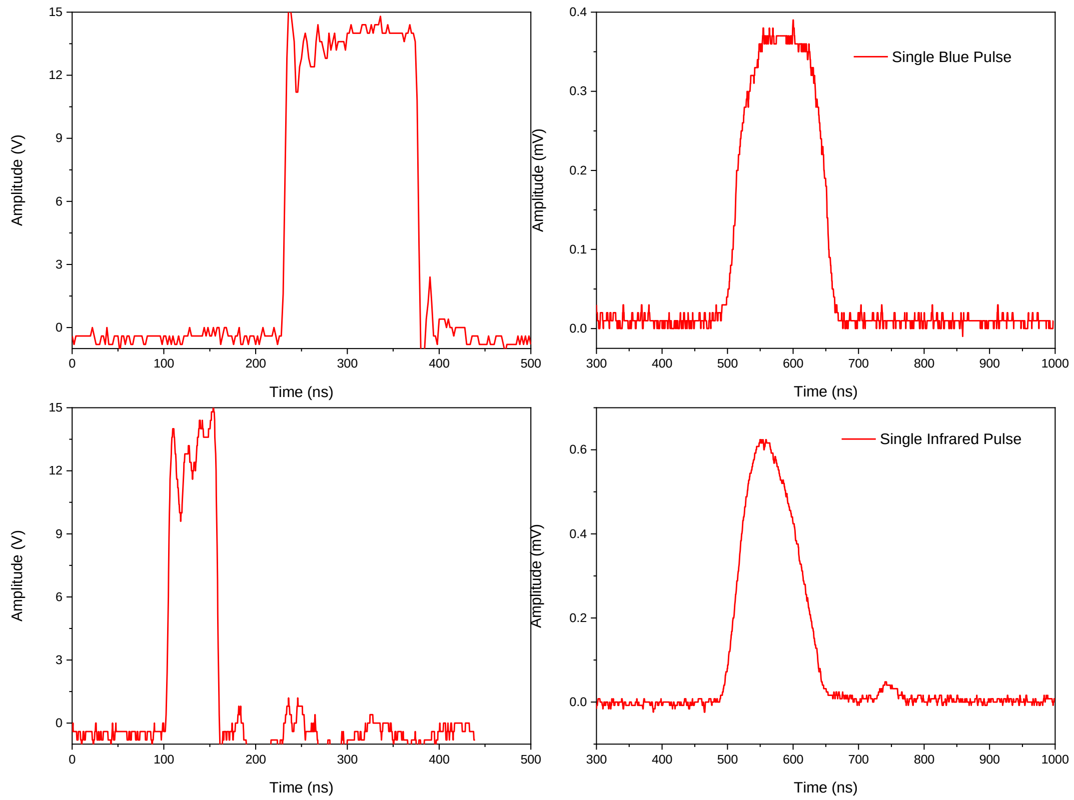

Figure 6.4. Electrical pulses and their respective stimulation pulses for blue and IR LEDs

This will also be fine-tuned in the future, but as for this research, the rise-time and fall-time of both profiles and the oscillatory ringing of the burst signal are acceptable, the rise and fall times are all under 10ns, which is a requirement for a sharp pulse, and the oscillatory ringing does not exceed TTL high level, thus can be ignored.

Two LEDs used, a 5 mm Blue LED from Foryard and a 3.5x3.5 mm to Cree clone LED mounted on a 12 mm PCB, output very different profiles, however, this is caused by the fact that the infrared LED is a power-LED, capable of continuously be run on 1A, whilst the blue LED is a 5 mm, standard package LED. Further studies will be done on this aspect of the experiment before any publications are made.

6.4 Tests of the Detection System

The detection system, which comprises a compact PMT, also needs to be thoroughly tested essentially in terms of its time response. Before mounting and integrating the tube into the system its response to low light levels was tested. In Figure 6.5 an example of a current pulse produced in response to an absorption of a photon is shown. The current pulse was recorded with the help of an oscilloscope by measuring the voltage drop on a 50-ohm terminating resistor. As seen from the figure, the negative pulse has an amplitude of 80 mV and it is 20 ns wide (FWHM). The PMT was biased with 900V during this test.

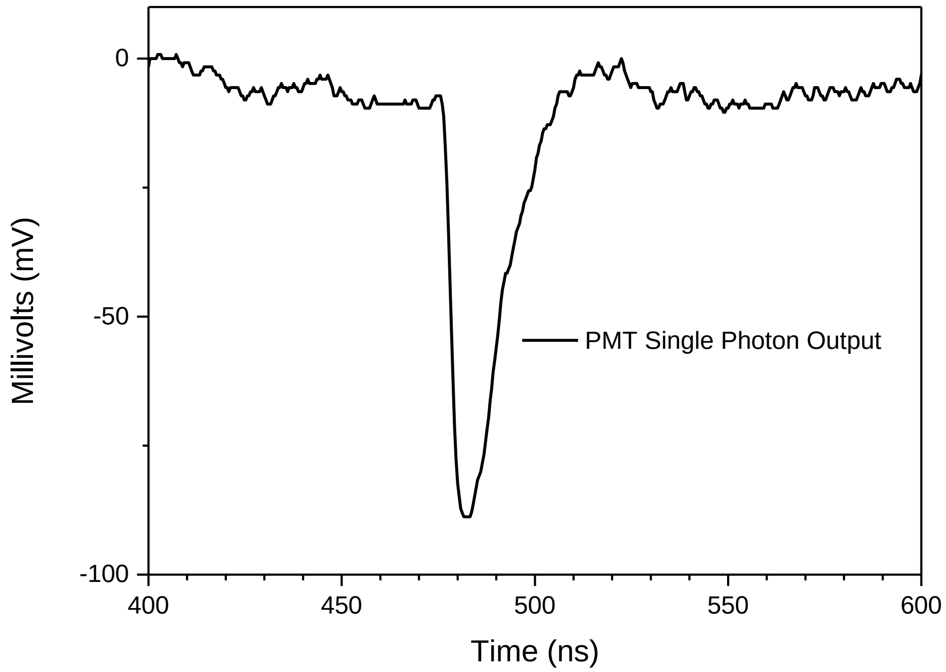

Figure 6.5. PMT Photon Response under External Test

After the verification of the PMT's single photon pulse, its high bias voltage was characterized. As the datasheet states, the bias voltage should be between 0.9kV and 1.1kV whilst using a potentiometer to control (Hamamatsu, 2022). The test was done for dark current and blue light illumination since the PMT has an excellent radiant sensitivity (around 130mA/W) for around 400 nm, yet is not able to detect any photon above approximately 720 nm. 11 steps of 20 Volts between 0.9kV and 1.1kV was measured, while the dwell time was set to 1μs and total illumination time (in the case of stimulation HV scan) was chosen to be 1μs as well. The threshold for the data acquisition system was set to -50.5mV during this test. In setting the PMT to be biased at 980V, the signal-to-noise ratio was found to be 49.07, which can be classified as an excellent SNR value. Thus, the high voltage bias of the PMT was decided. The plot of the HV scan performed can be found in Figure 6.6.

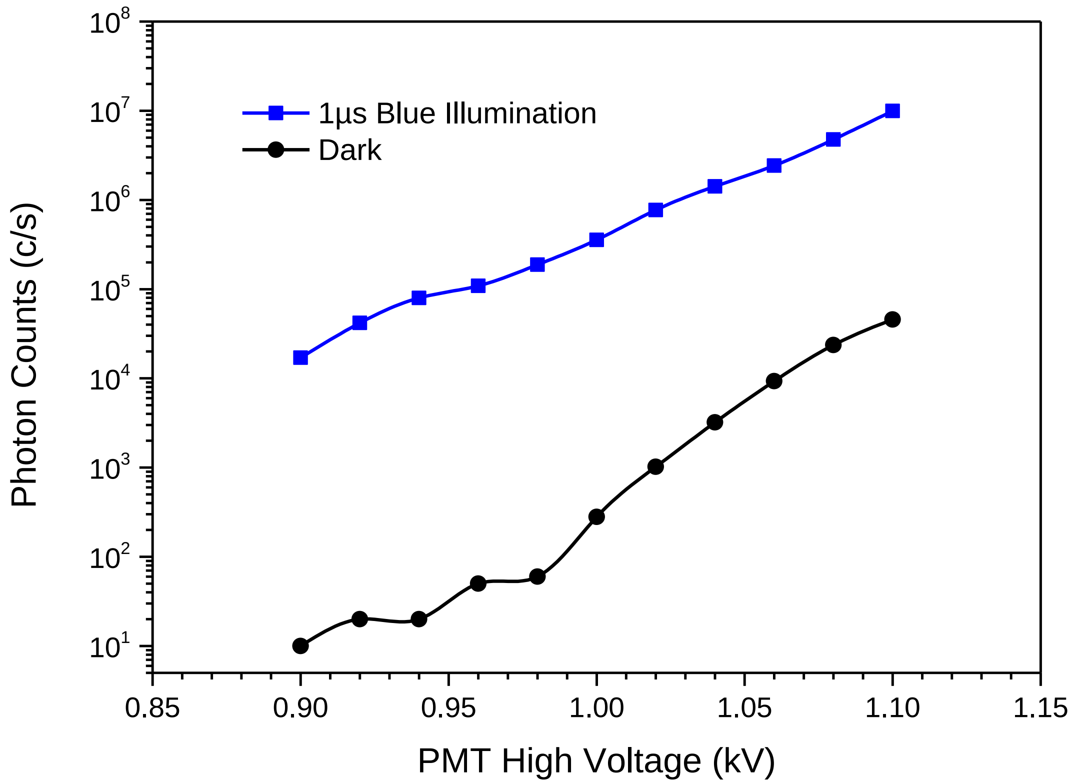

Figure 6.6. PMT High Voltage Scan between 900 and 1000V

Easy-MCS allows the user to set a predefined threshold level for any analog signal input. This threshold is adjusted from its software and is between -1.6 V and +3 V with 1.5mV steps, along with a choice between 50Ω and 1kΩ load and the type of edge detection, rising or falling (Ortec, 2013). To match impedance, 50Ω was selected, and as photon pulses would be negative, falling edge was chosen. The measurement was done between -100mV and -1mV, at approximately steps of 10mV each, as best as the 1.5mV resolution allowed. Similar to the HV scan, dwell time and total illumination duration were set to be 1µs, as a different parameter, bias voltage was set as 1kV. The PMT plateau was found to be between -61mV and -31mV, as can be seen from Figure 6.7, and the highest signal-to-noise ratio was found to be 1304 at -61mV.

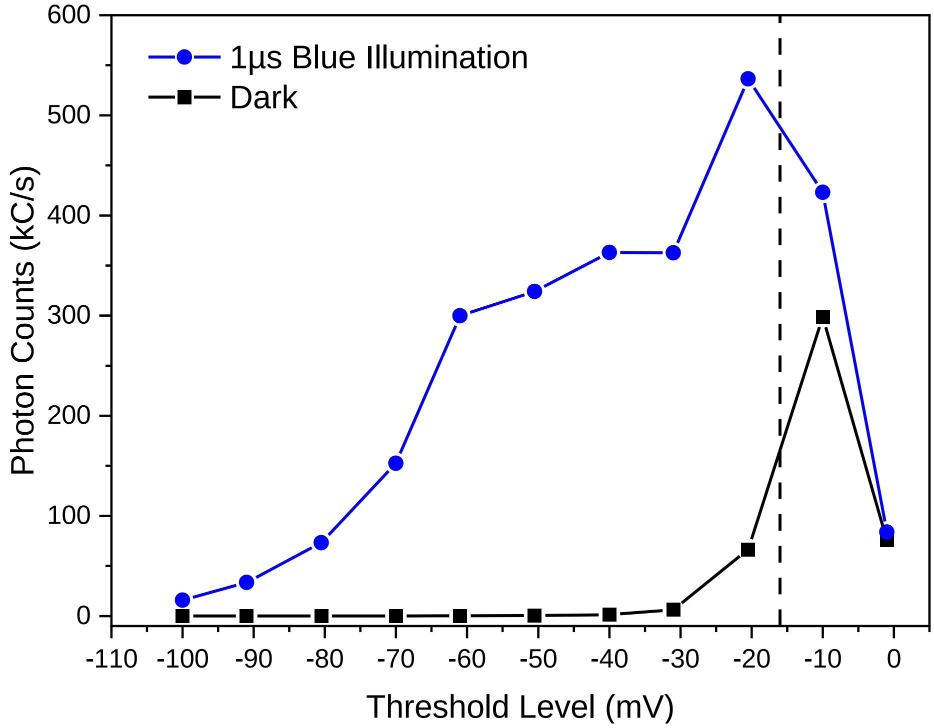

Figure 6.7. PMT Threshold Scan and Plateau between -61 and -31mV

Afterward, the total illumination profiles were tested for the LEDs to verify their synchronization with the dwell time pulses. Sample pulses of 1, 4, 20 and 40 pulses were given to the LED's. This test was done by using a 250ns dwell time, -40 mV threshold, and 1.1 kV PMT bias. The expected outcome was for the number of illumination pulses (restricted to a total of 250ns for each pulse) to be equal to the number of channels the pulses were detected. This was achieved by adjusting the delay using the LED Power PCB. The resulting channel vs. counts graph can be seen in Figure 6.8.

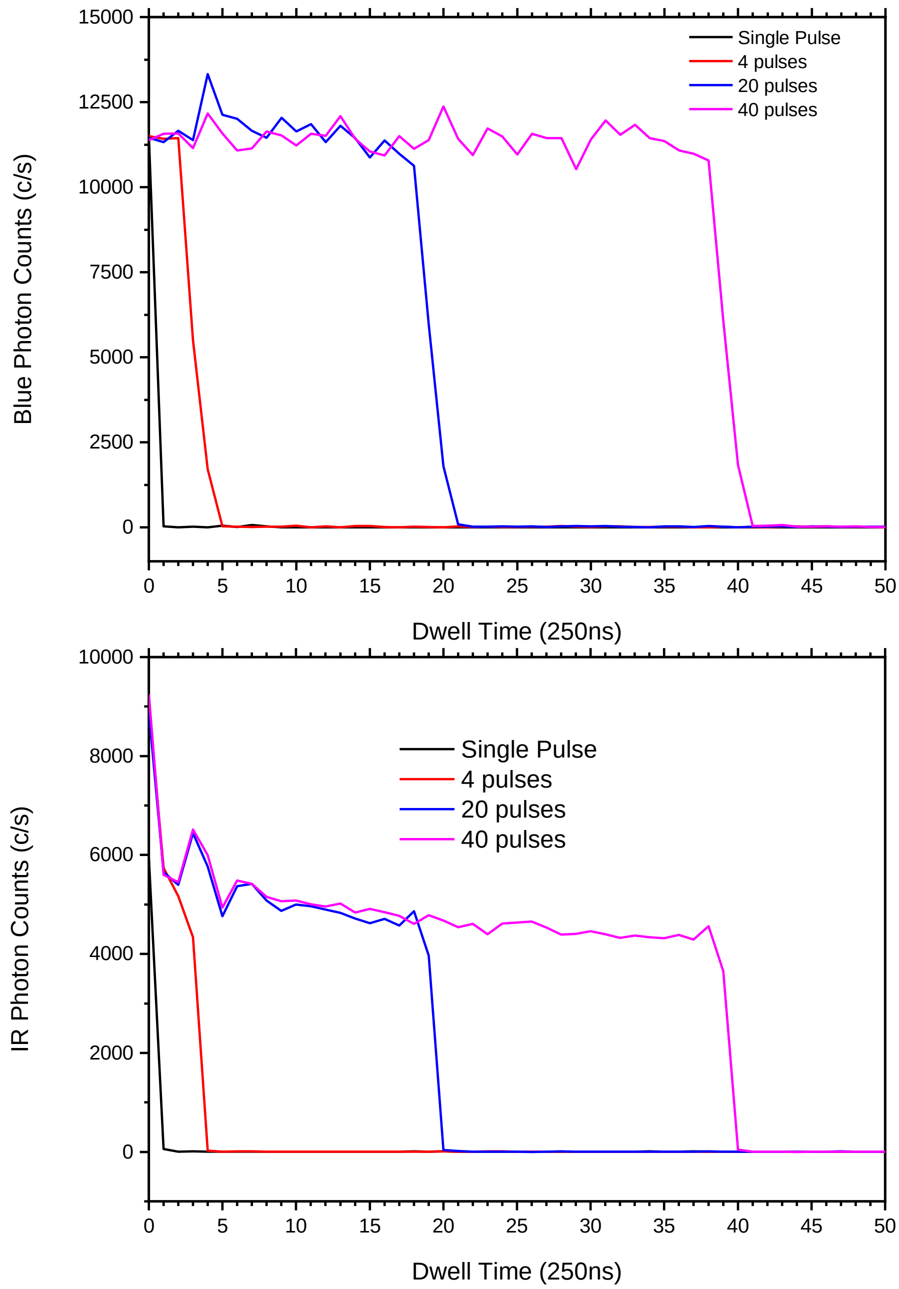

Figure 6.8. Counts vs Dwell Time graph for several pulses from each LED

After the stimulation system was verified to be within experimental parameters, filters were implemented into the system and tested to observe their effect on photon counts. It was found that even though the result of the high voltage scan, which showed that the ideal PMT bias voltage was between 980 and 1000V was accurate, more adjustment was needed on the threshold. -16mV was selected as the threshold after the implementation of filters and tests were conducted, where the dwell time was set as 1µs and a total illumination of 50µs was given to the system, acquisition of 10^5 was run. Even though 3 different filters were used, some photons were observed to have wavelengths within the detection region of the spectrum. The resulting photon counts can be seen in Figure 6.9.

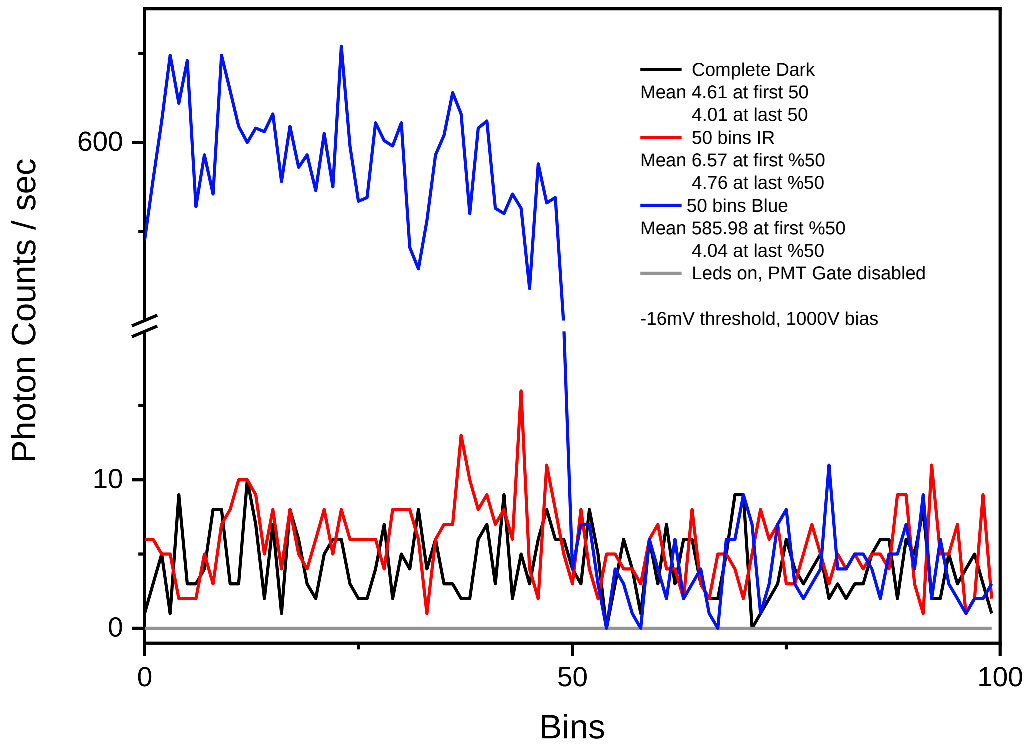

Figure 6.9. Light illumination profiles after filter installation

6.5 Experimental Results

In this part of the thesis, experiments conducted on materials relevant to dosimetry and dating will be presented. These measurements aim to test the abilities of the constructed instrument by comparing the outcomes of it with the previously published data, thereby confirming the reliability of the device.

For this purpose, TR-OSL from BeO ceramics (Thermalox 995) and Al_2O_3:C (OSL dosimetry materials used in commercial dosimetry systems) and minerals relevant for age determination such as quartz, zircon, and feldspar were studied. In the following paragraphs, the results of these experiments are summarized.

6.5.1 BeO Experiments

BeO (Beryllium oxide) is one of the materials used in commercial dosimetry systems therefore has been studied in detail previously and, therefore can be used for testing the system. Five BeO ceramic discs (Thermalox 995), with dimensions of 4 mm diameter and ~1 mm thickness (samples used by Bulur and Saraç, 2013), thermally depleted (450°C, 10 hours) samples were irradiated with 1 Gy using a $^{90}Sr/^{90}Y$ β-source (dose rate 0.025Gys^{-1}). After the irradiation, samples were preheated at 100 °C for 15 minutes to remove the charges trapped by shallow traps. TR-OSL measurements were carried out at room temperature using blue LEDs, with 50 μs stimulation and 150 μs measurement (dark) time. Easy-MCS dwell time was set to 1 μs and a total of 10^6 acquisitions were performed per sample.

An example of a TR-OSL curve from a BeO sample (showing both the stimulation and decay parts of the curve) can be seen in Figure 6.10 below. The dashed line indicates where the stimulation ends and the decay begins. As the stimulation data yield no usable data in the scope of this research, in further graphs only the decay part of the TR-OSL curve will be plotted. An example plot from the 1 Gy irradiated sample can be seen in Figure 6.11.

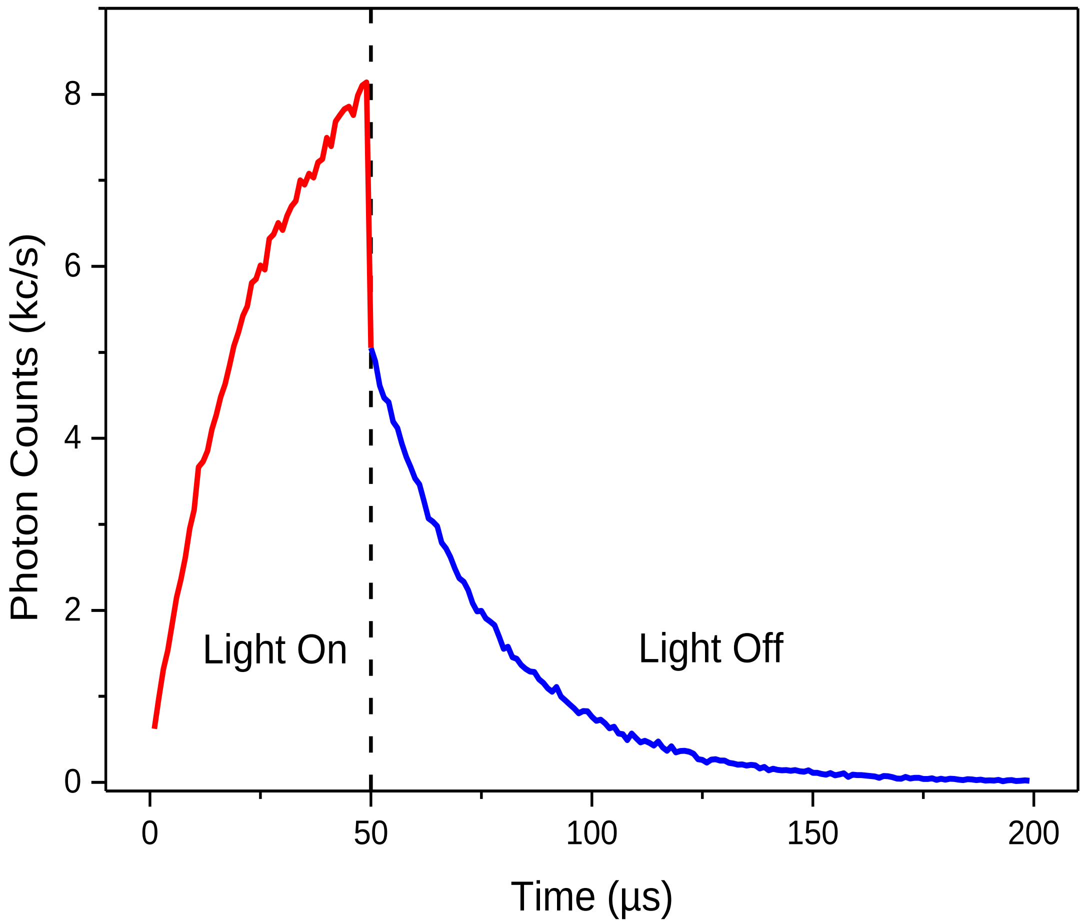

Figure 6.10. TR-OSL from BeO ceramics, red line shows stimulation, blue line denotes decay

The decay part of the TR-OSL curve can be analyzed by curve-fitting using an exponential decay function of the form;

$$y = L_{max}\, e^{\frac{-x}{\tau}} + \eta \tag{27}$$

where L_{max} is the maximum luminescence intensity, τ is the time constant of the decay, and η denotes the background counts. This curve fit is the same as the equation in Chapter 3, equation 24, with equation 26 substituted in place. In order to curve fit, OriginPro 9 was used.

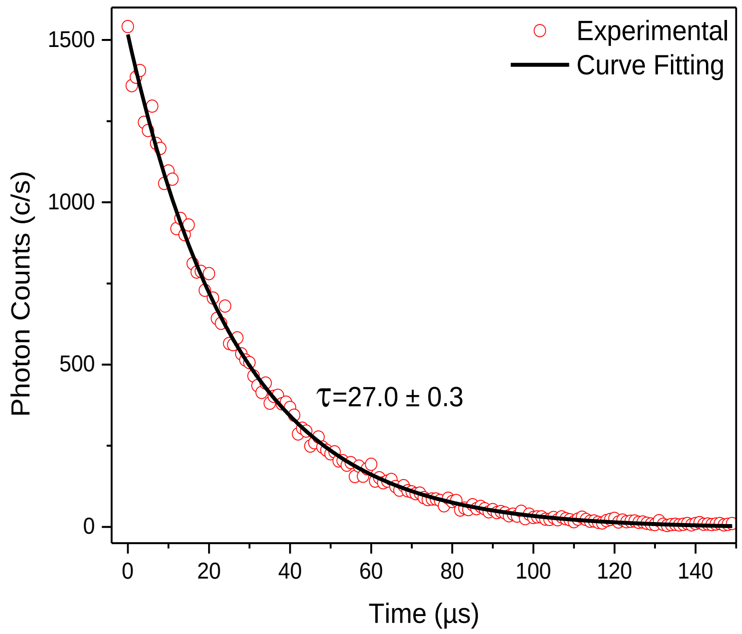

Figure 6.11. TR-OSL decay curve of a BeO sample after 1 Gy irradiation

To verify the dose-response of the material, the above-mentioned measurement was repeated for 5 and 10 Gy irradiated samples (see Figures 6.12 and 6.13). Using the L_{max} values obtained from all these measurements a dose-response curve can be plotted as shown in Figure 6.14.

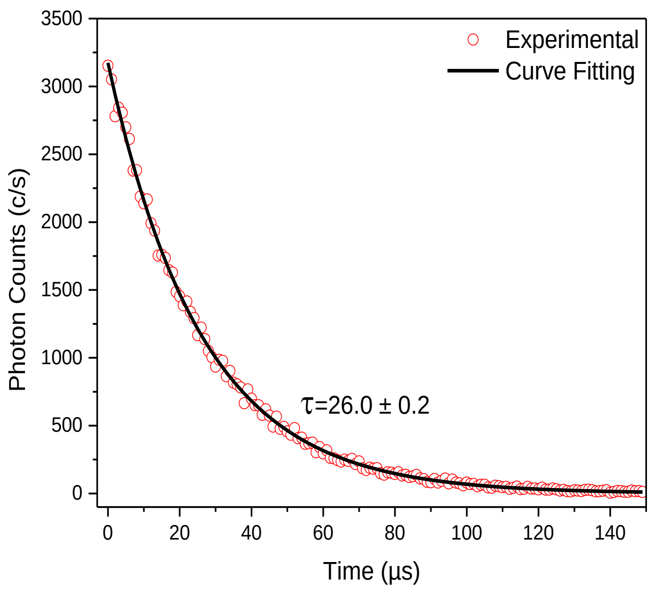

Figure 6.12. TR-OSL decay curve of a BeO sample after 5 Gy irradiation

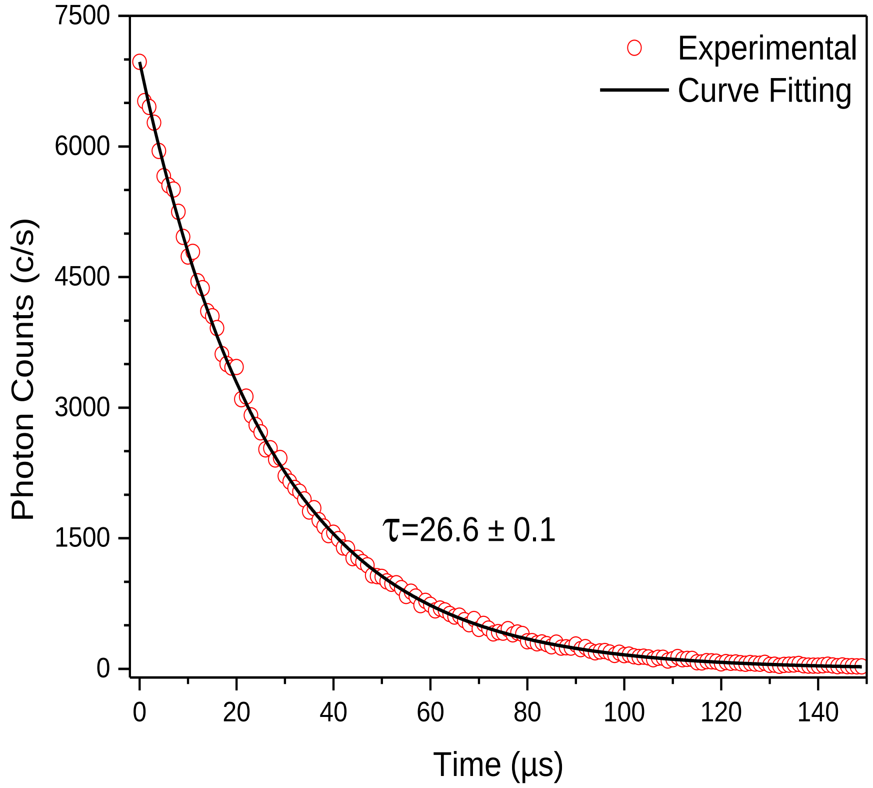

Figure 6.13. TR-OSL decay curve of a BeO sample after 10 Gy irradiation

The calculated time constants of all samples with different dose levels (total average of 26.0 ± 0.2) are in agreement with Bulur and Saraç's (2013) results of ~26 µs.

As mentioned in Chapter 3, it is expected for maximum luminescence intensity to be linearly increasing with the absorbed dose, as shown by Yukihara et al. (2016). The intensity (L_{max}) of the TR-OSL curves of five samples were normalized with respect to sample 5. The normalized L_{max} values were then used to plot dose vs. time. As can be seen from the plot in Figure 6.14, luminescence intensity is linearly increasing with respect to dose, as was shown in past research (for example, Yukihara et al. 2016).

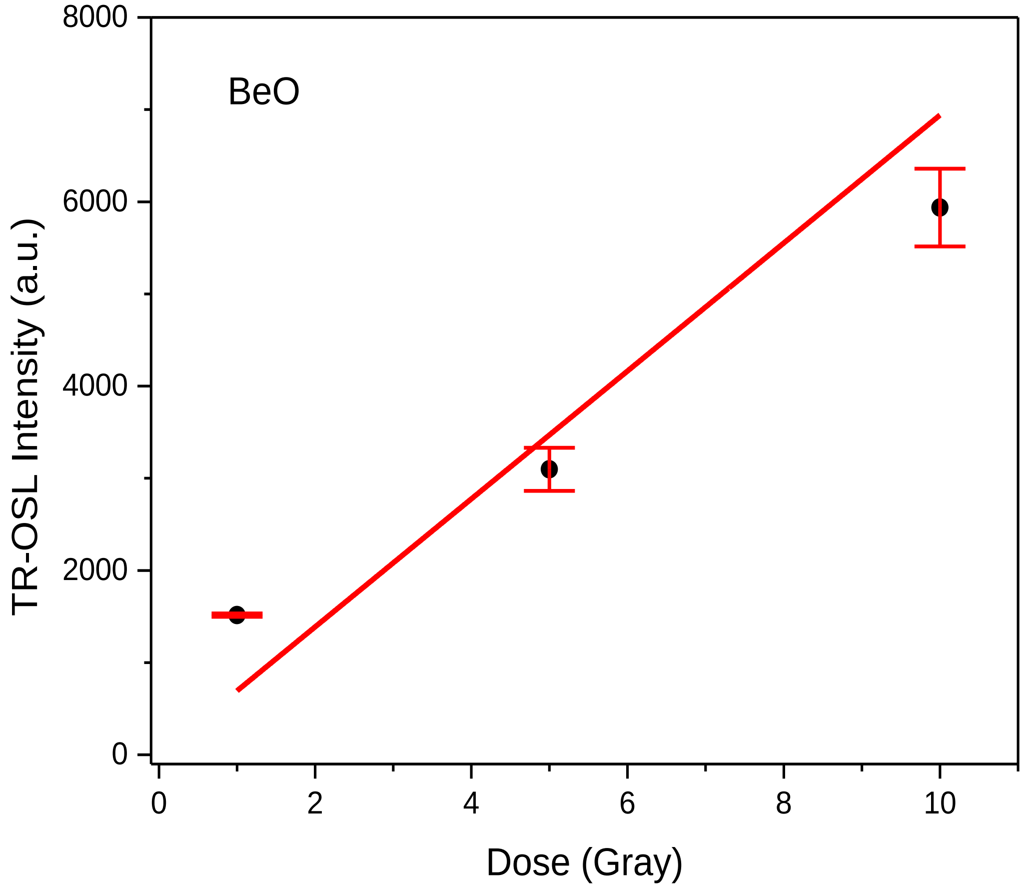

Figure 6.14. Dose-response of the TR-OSL signal from BeO

6.5.2 Al_2O_3:C Experiments

Al_2O_3:C, similar to BeO, is considered an industry standard for OSLDs. They are commonly used as passive dosimeters and have been extensively tested and researched in the past therefore can also be used to test the capabilities of the instrumentation system.

Before the irradiations, all samples were optically bleached (exposing to bright blue light to remove trapped electrons, for 10 hours) and then irradiated. The samples were then heated to 100°C for 15 minutes in order to remove shallow traps. A dwell time of 125 μs and a stimulation duration of 8 ms was set and kept constant for all Al_2O_3:C measurements. Dark (measurement) duration was set to 242 ms.

6.5.2.1 Experiments using Blue LEDs

This part of the experiment was conducted with blue LEDs. 5 identical, ground & compressed Al_2O_3:C samples were irradiated with 5 Gy. A total of 5×10^3 acquisitions were performed per sample. A plot of the 5 Gy irradiated samples' luminescence signal is given in Figure 6.15.

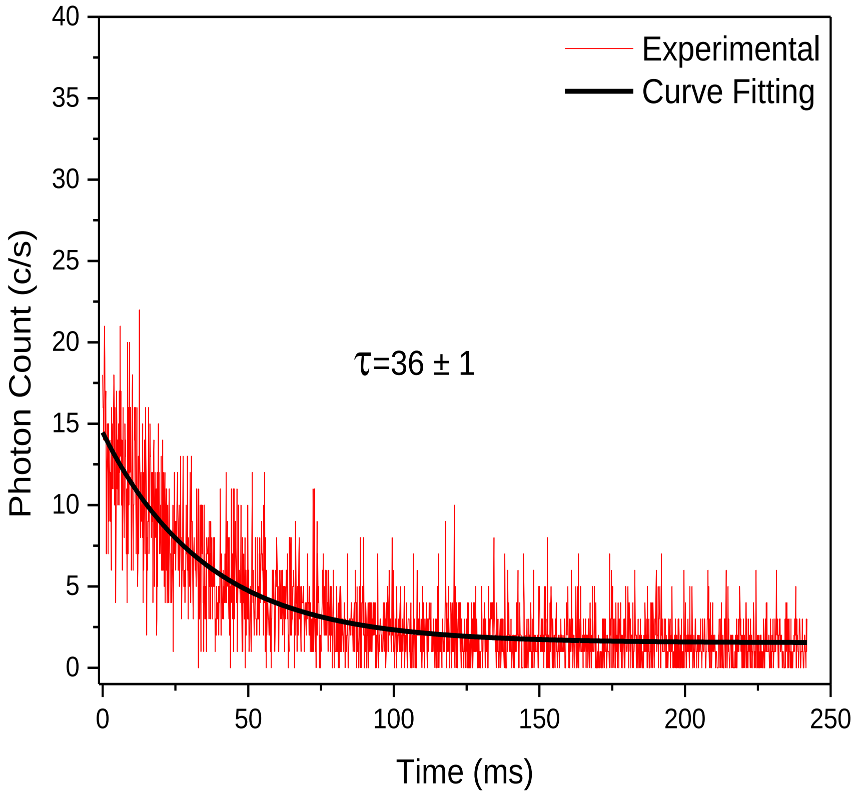

Figure 6.15. TR-OSL decay curve of an Al_2O_3:C sample after 5 Gy irradiation

The average of the decay time constants τ (~36 ms) from the 5 Gy measurements is in good agreement with past research results (Markey et al, 1995) of the Al_2O_3:C OSL decay, ~35 ms.

Following this measurement, the samples were exposed to an additional 5 Gy of dose and tested under the same parameters. An example plot can be found in Figure 6.16.

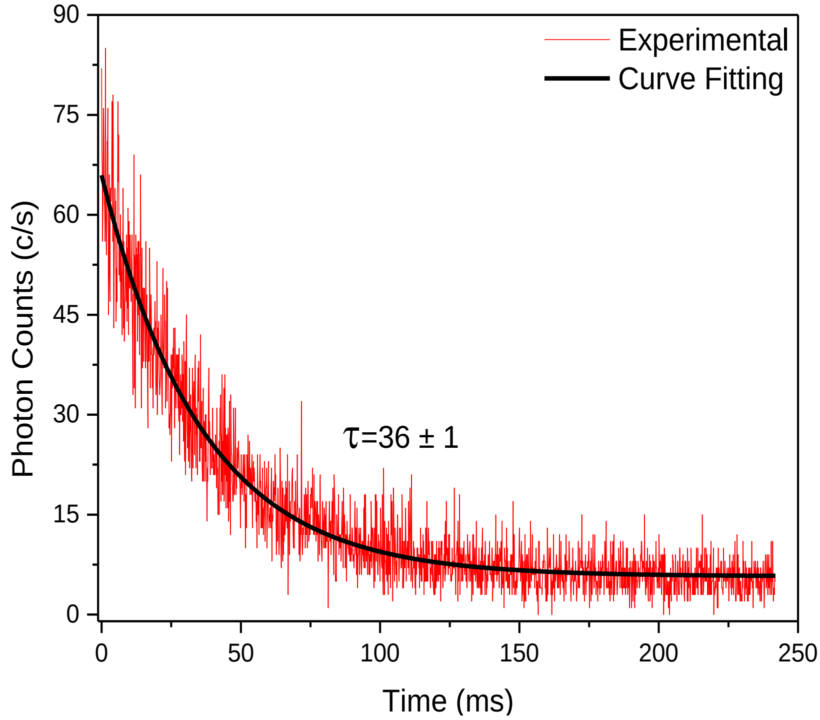

Figure 6.16. TR-OSL decay curve of an Al_2O_3:C sample after 10 Gy irradiation

Luminescence from Al_2O_3:C is also expected to be increasing linearly with dose as Markey et al. showed in their 1995 research. TR-OSL intensities were normalized to Sample 3 and then plotted with respect to the absorbed dose. The plot can be seen in Figure 6.17.

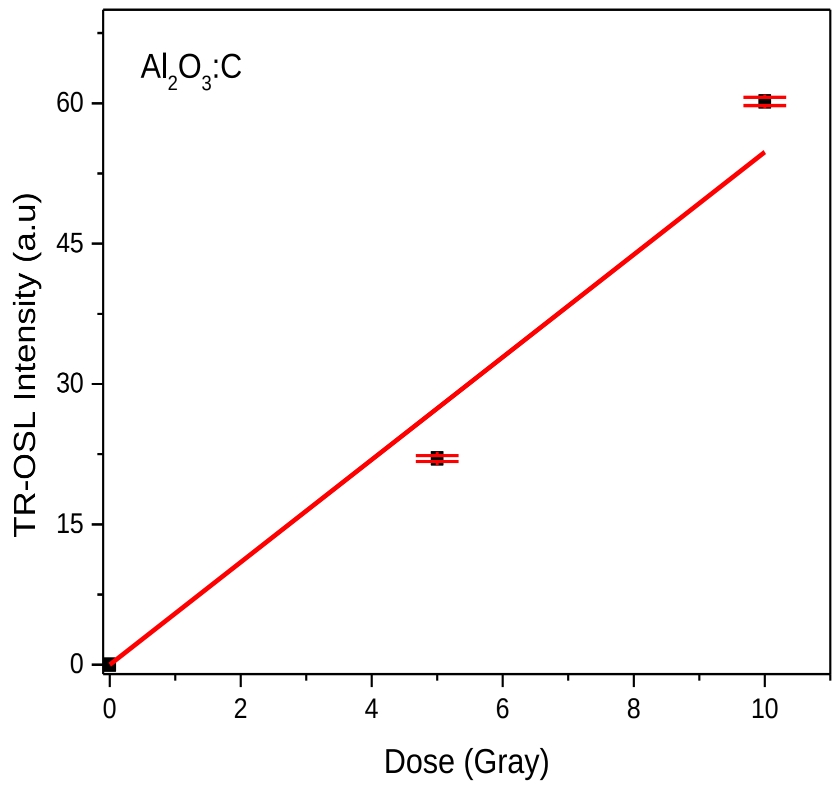

Figure 6.17. Dose-response of the TR-OSL signal from Al_2O_3:C

As can be understood by comparing figures 6.14 and 6.17, both follow a linear trend increasing with the dose, however, as the dose gets larger the linearity relation breaks down, and the probability of error increases.

6.5.2.2 Experiments using IR LEDs

As reported by Bulur and Göksu (1998b), Al_2O_3:C has a TR-OSL decay component with τ~37ms under infrared stimulation. This part of the experiment was conducted with infrared LEDs (λ_p~850 nm). A total of 1×10^4 acquisitions was performed. One crystalline Al_2O_3:C sample was irradiated with 10 Gy. For the measurement of the luminescence signal to be possible, the HOYA U340 filter was removed from the system, since the luminescence from IR stimulation is around 420nm, outside the detection area seen in Figure 5.7. The result of 39.4 ms decay constant is in agreement with the ~37ms constant that can be found in past research (Bulur and Göksu, 1998b). The resulting plot is presented in Figure 6.18.

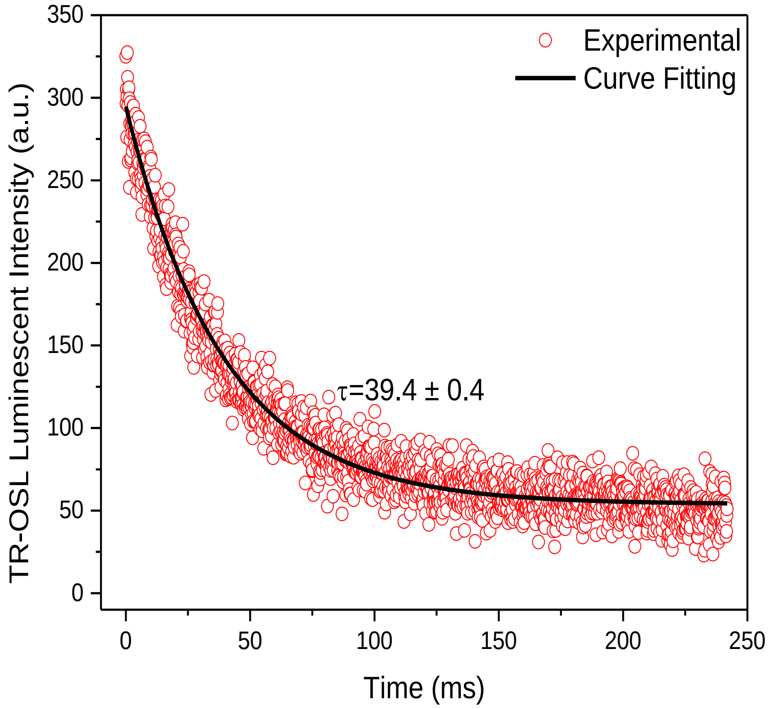

Figure 6.18. TR-OSL decay curve of an Al_2O_3:C sample after 10 Gy irradiation with IR Stimulation

6.5.3 CaF_2 Experiments

Natural Fluorites such as CaF_2 (fluorspar) are also candidates for dosimetry, however, these materials have not been extensively studied. Kuşoğlu Sarıkaya, in her 2011 research, tested 7 different fluorspars. In her results, she found that every fluorspar exhibited a 50 ns fast decay, as well as a slower decay, ranging from ~58 ms to ~250 ms depending on the sample. The measurement system does not have the time resolution required to measure the ~50 ns signal, thus the longer decay was attempted to be measured. The CaF_2 used was in its natural crystal form and was irradiated to 5 Gy. The dwell time of the Easy-MCS was set to 125 μs and the sample was stimulated for 1 ms using the blue LED. 5×10^4 acquisitions were performed. An example plot of the measurement is shown in Figure 6.19.

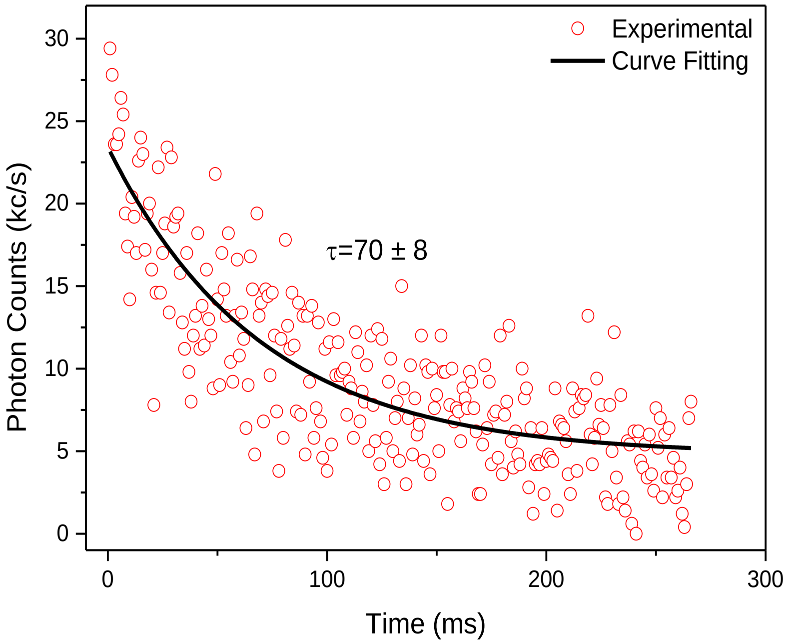

Figure 6.19. TR-OSL decay of CaF_2 after 5 Gy irradiation

The measured time constant of 70 ± 8 ms is within the reported range of ~58 and ~250 ms as shown by Kuşoğlu Sarıkaya (2011). More information on the CaF_2 luminescence mechanism can be obtained by measuring the sample with a faster system as well as implementing spectral resolution.

6.5.4 $ZnSiO_4$ Experiments

Zircon is a common mineral in the Earth's crust and has been shown to be an ideal candidate for age-determination applications. Smith et al. (1988), Godfrey-Smith et al. (1989), Turkin et al. (2006) and Bulur et al. (2014) have published research on the OSL properties of $ZrSiO_4$. More recently, Shewan et al. (2021) have conducted TR-OSL experiments on $ZrSiO_4$.

In this research, $ZrSiO_4$ powder was irradiated to 5 Gy. The samples were then heated up to 100°C for 10 minutes. The sample was then stimulated with blue light (λ_p~470 nm) for 100 µs and was measured over 1 ms. The dwell time of the Easy-MCS was set to 1 µs. The exponential decay of $ZrSiO_4$ can be seen in Figure 6.20.

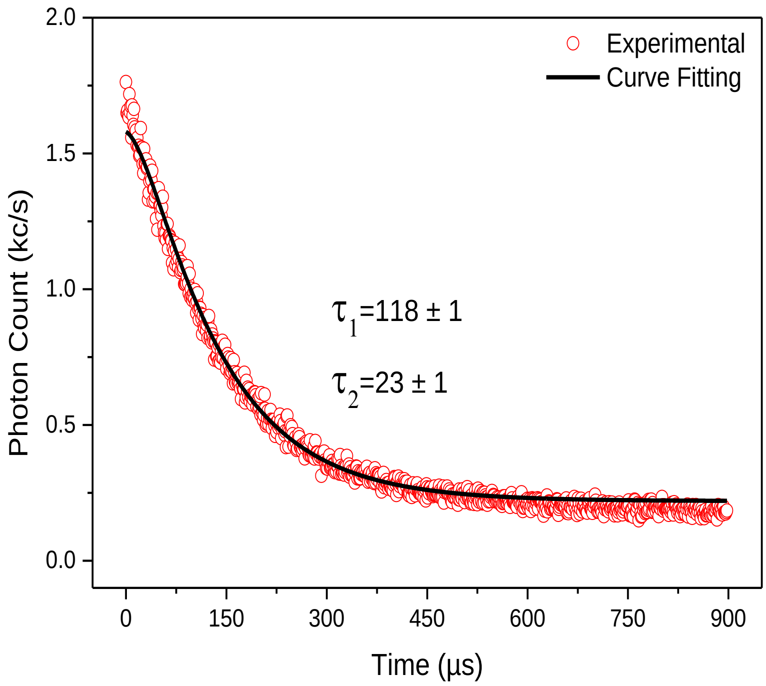

Figure 6.20. TR-OSL decay of $ZrSiO_4$ after 5 Gy irradiation

In their study, Bulur et al. (2014) found that zircon has several decay components. In this research, two decay components were verified with good approximation. In their research, Bulur et al. estimated the fast component of the Zirconium decay to be ~17 µs and the slow component to be ~110 µs. Curve fitting the luminescence data taken from the measurement system yielded an approximation of ~118 µs for the slow component and ~23 µs for the fast component of the Zirconium TR-OSL decay.

6.5.5 Quartz Experiments

Quartz crystals was historically a prime candidate for age determination. It has been studied extensively in the past (Murray et al., 2021). In more recent decades, Bailiff

(2000), Chithambo and Galloway (2000a), and Yeltik (2009) have shown the TR-OSL properties of Quartz crystals. Yeltik, in her research, has used Merck quartz, a similar sample of quartz used in this research. In her work, she determined the lifetime of quartz as 42 ± 2 µs. In this research, a similar Merck quartz powder was irradiated to 10 Gy. The sample was then stimulated using blue LEDs (λ_p~470 nm) for 50 µs and the decay was measured for 350 µs. The dwell time of the Easy-MCS was set to 1 µs. 10^6 acquisitions were performed. The TR-OSL decay curve can be found in Figure 6.21.

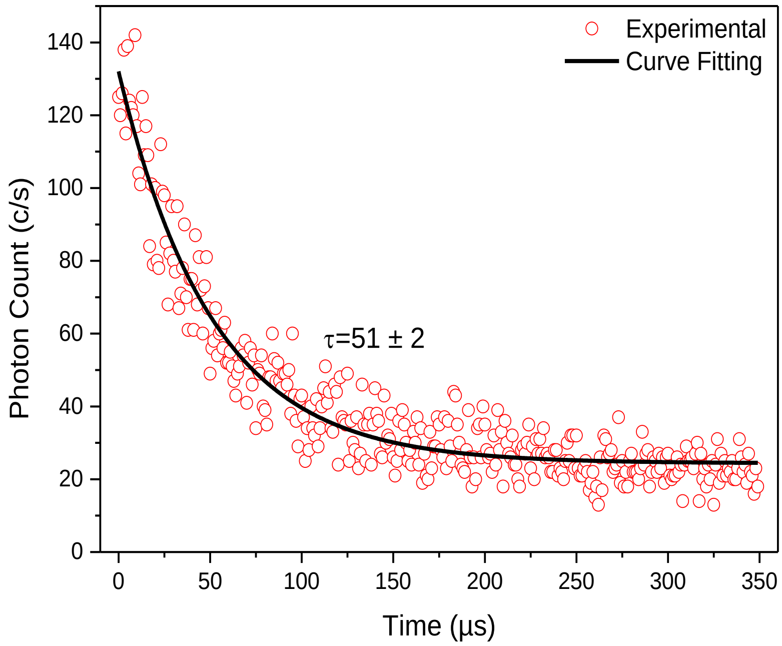

Figure 6.21. TR-OSL decay of quartz after 10 Gy irradiation

The decay time constant of quartz was found to be 51 ± 2 µs. This is in the range of previous literature, which states that the time constant is between ~35 and ~55 µs (Schmidt et al., 2019).

6.5.6 Feldspar Experiments

Feldspar is another material that is widely used for age determination and dating. Ankjaergaard et al. (2009) researched TR-OSL properties of feldspar using a similar

470nm LED and estimated the lifetime of the decay to be in the order of 10 ns. In the past, this number was also calculated by Sanderson and Clark (1994) and Bailiff and Poolton (1991). The system used in the latter two was able to measure the decay of the nanosecond domain, thus their results are a reliable point of reference.

This system does not possess the temporal resolution to measure that fast of a time constant, yet it can be shown that the system does indeed stimulate feldspar and measure luminescence during stimulation. The feldspar sample used is a sedimentary fine-grained sample that has been irradiated to 50 and 500 Gy sequentially. The data acquisition dwell time was set to 1 μs and the sample was stimulated with 100 μs of light. 10^6 acquisitions were performed. The expectation was for the total count during stimulation to be approximately 10 times between 50 and 500 Gy measurements. The 100 μs stimulation and the following 100 μs dark times can be seen in the plot below. The plot shows that whilst the sample was irradiated to 50 Gy, a total of 594 counts were detected, in comparison, while the sample was irradiated to 500 Gy, a total of 5527 counts were found. A factor of 9.31 was found between the two measurements, which is in agreement with the expectation of approximately 10 times. Both luminescence signals can be seen in Figure 6.22.

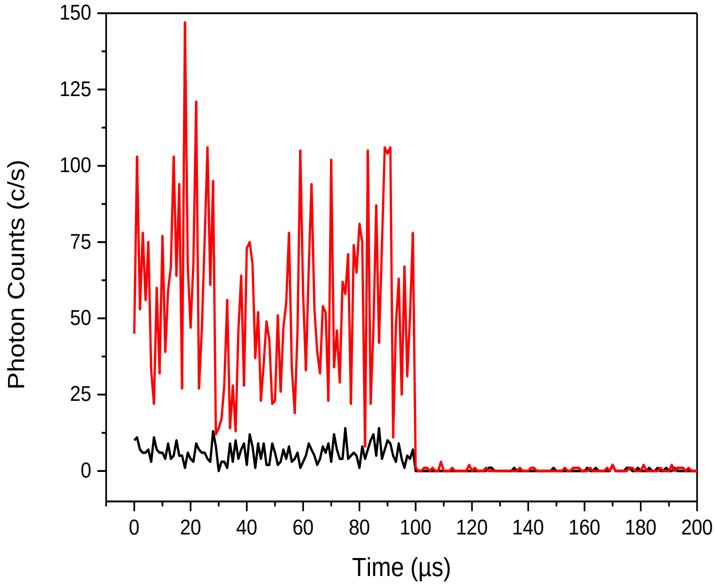

Figure 6.22. Feldspar stimulation and background after 50 Gy and 500 Gy,

6.5.7 BeO Nanophosphor Experiments

To show that the system is also able to make use of nanophosphor luminescence, BeO: Al, Ca ceramic nanoparticles were used. The BeO: Al, Ca nanoparticles tested in this research are of the ceramics Altunal synthesized for his 2018 study (Altunal et al., 2018). As these are BeO ceramics, the time response is expected to be ~26 µs as mentioned above. Four samples were first irradiated with 5 Gy, then stimulated using Blue LEDs (λ_p~470 nm) for 50 µs. Easy-MCS dwell time was set to 1 µs and 10^6 acquisitions were performed. Dark time was set to 350 µs. A sample plot can be seen in Figure 6.23.

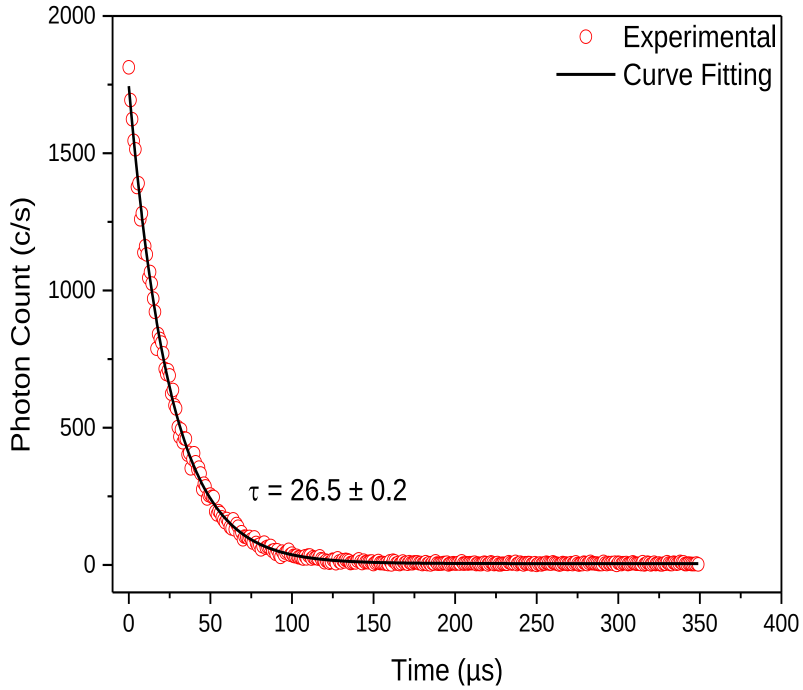

Figure 6.23. Decay Graph of BeO: Al, Ca Sample 3 after 5 Gy irradiation

The average decay time constant (26.5 ± 0.2) is in agreement with the value of ~26 µs of Bulur and Saraç's (2013) research and past literature (Yukihara et al, 2011a).

CHAPTER 7

CONCLUSION

In this thesis, an automated multi-sample Time-Resolved Optically Stimulated Luminescence Measurement System has been realized using an open-source approach (at least partially). The device employs LEDs (blue, λ_p~470 nm and IR, λ_p~850 nm) as stimulation sources and a PMT module (Hamamatsu H12056P-110 working in photon counting mode) for detecting the emitted luminescence signal with appropriate filters to block stimulation light reaching to PMT.

Time-resolved recording of the luminescence signal was achieved by a multi-channel scaler (Ortec Easy-MCS). This device is able to collect data with a resolution adjustable from 100 ns to 125 us and can be triggered/stopped using external signals. The system supplies a stimulation light of adjustable width and simultaneously records the time development of luminescence with the possibility of accumulating the luminescence to increase the signal-to-noise ratio. The stimulation pulse width is adjusted by applying a sequence of pulses (burst) of 250 ns period (with a 3/5 mark to space ratio: 150 ns ON, 100 ns OFF). Such a driving regime allows the application of large current pulses to the LEDs, thereby increasing stimulating light pulse intensity. Stimulation light bursts and timing signals for the MCS were created using a clock generator/pulse generator circuitry governed by an Arduino-based microcontroller, which was interfaced with a client computer. To achieve easy control, the whole experiment a human interface software was written (using Python).

After the device was constructed, test experiments were conducted to verify whether the experimental setup was able to satisfy the desired specifications. These included testing of dwell times and timing, testing of the light source, testing of the stimulation system, and testing the detection system.

The response of the system with and without stimulation light was also noted to quantify background noise from stimulation. After ensuring the system is running properly, through a series of test experiments, the TR-OSL measurement system was tested using materials relevant to dosimetry and dating. TR-OSL signals from BeO (commercial ceramics and synthesized nanoparticles) have shown that BeO has an emission with a lifetime of around ~26 µs and Al2O3:C samples (of crystalline and powder form) exhibited well-known ~35 ms decay constants. The decay constants of both samples were found to be in good agreement with the previously reported values. Although not considered a standard material for radiation dosimetry with OSL, experiments carried out with natural CaF2 have shown a TR-OSL signal with a lifetime of around 70 ms conforming to the previous reports on the same material.

Quartz, feldspar, and zircon minerals, which are relevant for age determination, have also been analyzed. TR-OSL from quartz exhibited a decay of around 50 µs, which is in agreement with past research. The decay time constant of Zircon was found to be in the order of 118 µs and 23 µs, which is an acceptable result compared to the known values (~110 µs and ~17 µs) from the literature. Lastly, the decay time constant of feldspar could not be measured, due to the temporal resolution of the system not being enough and possibly the feldspar sample not being a pure, standardized sample. However, the system was used to stimulate feldspar with two different dose levels, and the luminescence signal intensity during stimulation increased as expected, approximately tenfold as the dose was multiplied by 10.

All these test experiments have shown that the system is reliable in terms of timing accuracy and reproducibility. However, it was also observed that the setup seems to not have enough sensitivity which is basically due to problems associated with the light-collecting optics which is to be enhanced in the near future by redesigning the optical/optomechanical system.

In summary, the constructed TR-OSL system is able to measure the TR-OSL decay curves of various samples relevant for dosimetry and dating. The information obtained from the TR-OSL, combined with spectral studies, may help in identifying/characterizing luminescence recombination centers. Thus, may help in increasing the knowledge of luminescence production mechanisms.

The open-sourced approach to this system enables any researcher to be able to improve and modify this system for their specific needs, with a budget of ~ 5000$ and may inspire other researchers planning to construct similar home-made research systems, as well as ensure that this system can, and will be, modified in the future for further research, with improved parameters.

REFERENCES

AccelStepper - Arduino reference. (2021).

https://www.arduino.cc/reference/en/libraries/accelstepper/ (Accessed: 21 July 2024).

Adafruit Industries, Si5351A/B/C Data Sheet, 2011.

https://cdn-shop.adafruit.com/datasheets/Si5351.pdf (Accessed: 7 August 2024).

Adafruit Si5351 Library - Arduino Reference. (2021).

https://www.arduino.cc/reference/en/libraries/adafruit-si5351-library/ (Accessed: 21 July 2024).

Aitken, M. J. (1985). Thermoluminescence Dating. London: Academic Press, 153.

Aitken, M. J. (1999). An introduction to optical dating: the dating of Quaternary sediments by the use of photon-stimulated luminescence. *Choice Reviews Online*, *36*(11), 36–6294. https://doi.org/10.5860/choice.36-6294

Alajlani, Y., & Can, N. (2022). Thermoluminescence glow curve analysis and kinetic parameters of Dy-doped BaSi2O5 phosphor. *Journal of Rare Earths*, *40*(2), 234–242. https://doi.org/10.1016/j.jre.2020.10.020

Alleman, Glen. (2002). Is There an Underlying Theory of Software Project Management? (A critique of the transformational and normative views of project management).

Altunal, V., Güçkan, V., Özdemir, A., Bulur, E., & Yeğingil, Z., (2018). Optically Stimulated Luminescence (OSL) from BeO:Al,Ca Ceramics Synthesized via Sol-Gel Method . 11st Internatinal Conferences on Luminescence and ESR Dosimetry (LUMIDOZ11), Gaziantep, Turkey

Altunal, V., Abusaid, W., Akdeniz, Y., Guckan, V., Ceyran, G., Khandaker, M., & Yegingil, Z. (2023). Studies of OSL properties of alkali- and rare earth-doped BeO based novel dosimeters for applications in external beam radiotherapy. *Radiation Physics and Chemistry*, *212*, 111136. https://doi.org/10.1016/j.radphyschem.2023.111136

Altunal, V. (2024a). Structural and luminescence characterization of $LiAlO_2$ ceramics synthesized by Sol-Gel technique. *Adıyaman Üniversitesi Fen Bilimleri Dergisi*. https://doi.org/10.37094/adyujsci.1489596

Altunal, V., Ozdemir, A., Guckan, V., Kurt, K., Kavgaci, M., Takci, K., Yigit, O.(2024b). Luminescence properties of SrB_6O_{10}:Ce,Gd phosphor for applications in OSL dosimetry and lighting. Journal of Luminescence, vol.273.

Altunal, V., Ozdemir, A., Kurt, K., Yigit, O., Guckan, V., Isik, B., & Yegingil, Z. (2024c). Luminescence properties of Ce and Be-doped $CaAl_{12}O_{19}$ for dosimetric and light-emitting applications. *Journal of Alloys and Compounds*, *985*, 174081. https://doi.org/10.1016/j.jallcom.2024.174081

Ankjaergaard C., Jain M., Kalchgruber R., Lapp T., Klein D., McKeever S. W. S., Murray A. S. and Morthekai P. (2009), Further investigations into pulsed optically stimulated luminescence from feldspars using blue and green light, *Radiat. Meas*. Article in Press.

Antonov-Romanovskii, V. V., Keirum-Marcus, I. F., Poroshina, M. S., & Trapeznikova, Z. A. (1955). Conference of the Academy of Sciences of the USSR on the Peaceful Uses of Atomic Energy. *USAEC Report AEC-tr-2435, Part*, *1*, 239-250.

Arduino - home. (2006). https://www.arduino.cc/ (Accessed: 7 August 2024).

Arduino Mega 2560 REV3 (2021) Arduino Official Store.

https://store.arduino.cc/products/arduino-mega-2560-rev3 (Accessed: 29 June 2024).

Arduino Store (2024). Boards shields kits accessories. Arduino Official Store. https://store.arduino.cc/

Bajaj, N. S., & Palan, C. B. (2022). Understanding OSL radiation dosimetry and its application. In *Elsevier eBooks* (pp. 99–128). https://doi.org/10.1016/b978-0-323-85471-9.00007-5

Bailiff, I., & Poolton, N. (1991). Studies of charge transfer mechanisms in feldspars. International Journal of Radiation Applications and Instrumentation Part D Nuclear Tracks and Radiation Measurements, 18(1–2), 111–118. https://doi.org/10.1016/1359-0189(91)90101-m

Bailiff I. K., Characteristics of time-resolved luminescence in quartz, Radiat. Meas. 32, 401-405 (2000).

Bailiff, I.K. (2010). Thermoluminescence: Its Use in Archaeology. In Radiation Measurements in Archaeology and Art (pp. 93-117). Elsevier.

Banzi, M., & Shiloh, M. (2014). Getting Started with Arduino: The Open Source Electronics Prototyping Platform. Maker Media, Inc.

Barnett, S., & Bailiff, I. (1997). Infrared stimulation spectra of sediments containing feldspars. *Radiation Measurements,27*(2), 237–242.https://doi.org/10.1016/s1350-4487(96)00116-3

Bentley, J. (2008). Principles of Measurement Systems John P. Bentley. Pearson Prentice Hall.

Bestvater, F., Seghiri, Z., Kang, M. S., Gröner, N., Lee, J. Y., Im, K., & Wachsmuth, M. (2010). EMCCD-based spectrally resolved fluorescence correlation spectroscopy. *Optics Express, 18*(23), 23818. https://doi.org/10.1364/oe.18.023818

Bhuyan, M., (2011). Intelligent instrumentation: principles and applications. CRC Press, Boca Raton, FL.

Binnemans, K., (2020). Rare-earth ions as luminescent probes in life sciences. Encyclopedia of Spectroscopy and Spectrometry, pp. 209-218.

Bossin, L., Plokhikh, I., Christensen, J. B., Gawryluk, D. J., Kitagawa, Y., Leblans, P., Tanabe, S., Vandenbroucke, D., & Yukihara, E. G. (2023). Addressing Current challenges in OSL Dosimetry using MgB_4O_7:Ce,Li: State of the Art, Limitations and Avenues of Research. *Materials*, *16*(8), 3051. https://doi.org/10.3390/ma16083051

Bossin, L., Bailiff, I. K., & Terry, I. (2018). Phototransferred TL properties of alumina substrates. *Radiation Measurements*, *120*, 41–46. https://doi.org/10.1016/j.radmeas.2018.07.018

Brown, T. (2021). Open-Source Robotics and Motion Control. MIT Press.

Braeunlich, P, Schaefer, D, & Scharmann, A. A Simple Model for Thermoluminescence and Thermally Stimulated Conductivity of Inorganic Photoconducting Phosphors and Experiments Pertaining to Infrared-Stimulated Luminescence.

Bulur, E. (1996). An alternative technique for optically stimulated luminescence (OSL) experiment. *Radiation Measurements*, *26*(5), 701–709. https://doi.org/10.1016/s1350-4487(97)82884-3

Bulur, E., & Göksu, H. (1998a). OSL from BeO ceramics: new observations from an old material. *Radiation Measurements*, *29*(6), 639–650. https://doi.org/10.1016/s1350-4487(98)00084-5

Bulur, E., Göksu, H., & Wahl, W. (1998b). Infrared (IR) stimulated luminescence from α-Al_2O_3:C. *Radiation Measurements*, *29*(6), 625–638. https://doi.org/10.1016/s1350-4487(98)00076-6

Bulur, E., & Saraç, B. (2013). Time-resolved OSL studies on BeO ceramics. *Radiation Measurements*, *59*, 129–138. https://doi.org/10.1016/j.radmeas.2013.04.009

Bøtter-Jensen, L. (1988). The automated Risø TL dating reader system. *International Journal of Radiation Applications and Instrumentation Part D Nuclear Tracks and Radiation Measurements*, *14*(1–2), 177–180. https://doi.org/10.1016/1359-0189(88)90060-x

Bøtter-Jensen, L., Duller, G., & Poolton, N. (1994). Excitation and emission spectrometry of stimulated luminescence from quartz and feldspars. *Radiation Measurements*, *23*(2–3), 613–616. https://doi.org/10.1016/1350-4487(94)90108-2

Bøtter-Jensen, L., Larsen, N. A., Markey, B., & McKeever, S. (1997). Al2O3:C as a sensitive OSL dosemeter for rapid assessment of environmental photon dose rates. Radiation Measurements, 27(2), 295–298. https://doi.org/10.1016/s1350-4487(96)00124-2

Bøtter-Jensen, L. B., Duller, G., Murray, A., & Banerjee, D. (1999). Blue light emitting diodes for optical stimulation of quartz in retrospective dosimetry and dating. *Radiation Protection Dosimetry*, *84*(1), 335–340. https://doi.org/10.1093/oxfordjournals.rpd.a032750

Bøtter-Jensen, L., Bulur, E., Duller, G., & Murray, A. (2000). Advances in luminescence instrument systems. *Radiation Measurements*, *32*(5–6), 523–528. https://doi.org/10.1016/s1350-4487(00)00039-1

Bøtter-Jensen, L., McKeever, S. W. S., & Wintle, A. G. (2003). *Optically stimulated luminescence dosimetry* (1st ed). Elsevier. http://site.ebrary.com/id/10190134

Chen G, Qiu H, Prasad PN, Chen X (2014). Upconversion nanoparticles: design, nanochemistry, and applications in theranostics. Chem Rev.;114(10):5161-214. doi: 10.1021/cr400425h

Chen, R., & McKeever, S. W. S. (1997). *Theory of thermoluminescence and related phenomena*. https://doi.org/10.1142/2781

Chen, R., Pagonis, V. (2019) Advances in Physics and Applications of Optically and Thermally Stimulated Luminescence. Japan: World Scientific Publishing Company.

Choi, J., Kim, M., Cheong, C., & Hong, D. (2014). Development of OSL system using two high-density blue LEDs equipped with liquid light guides. *Nuclear Instruments and Methods in Physics Research Section B Beam Interactions with Materials and Atoms*, *323*, 19–24. https://doi.org/10.1016/j.nimb.2014.01.016

Chithambo, M. L. (2007). The analysis of time-resolved optically stimulated luminescence: I. Theoretical considerations. *Journal of Physics D Applied Physics*, *40*(7), 1874–1879. https://doi.org/10.1088/0022-3727/40/7/006

Chithambo, M. L., & Galloway, R. B. (2000). A pulsed light-emitting-diode system for stimulation of luminescence. *Measurement Science and Technology*, *11*(4), 418–424. https://doi.org/10. Chithambo M. L. and Galloway R. B., On luminescence lifetimes in quartz, Radiat. Meas. 32, 621-626 (2000a).1088/0957-0233/11/4/312

Chithambo M. L. and Galloway R. B., On luminescence lifetimes in quartz, Radiat. Meas. 32, 621-626 (2000a).

Chithambo, M. L. (2018). *An introduction to Time-Resolved optically stimulated luminescence*. https://doi.org/10.1088/2053-2571/aae5da

Ciftja, O. (2013). Understanding electronic systems in semiconductor quantum dots. *Physica Scripta*, *88*(5), 058302. https://doi.org/10.1088/0031-8949/88/05/058302

Clark, R., & Bailiff, I. (1998). Fast time-resolved luminescence emission spectroscopy in some feldspars. *Radiation Measurements*, *29*(5), 553–560. https://doi.org/10.1016/s1350-4487(98)00068-7

Daybreak Nuclear and Medical Systems, Inc. (2008). OSL Adapters. Guilford, Connecticut; Daybreak.

Denby, P., Bøtter-Jensen, L., Murray, A., Thomsen, K., & Moska, P. (2006). Application of pulsed OSL to the separation of the luminescence components from a mixed quartz/feldspar sample. *Radiation Measurements*, *41*(7–8), 774–779. https://doi.org/10.1016/j.radmeas.2006.05.017

DiBona, C., Ockman, S., and Stone, M., (1999). Open Sources: Voices from the Open Source Revolution.

Espinosa, G. (2011). A study and characterization of the optically stimulated luminescence response of commercial SiO2 optical fiber to gamma radiation. *Revista Mexicana De Física*, *57*(1), 30–33. http://www.scielo.org.mx/pdf/rmf/v57s1/v57s1a7.pdf

Espressif Systems. (2016). Wireless SOCs, Software, Cloud and AIoT Solutions | Espressif Systems. https://www.espressif.com/en (Accessed: 7 August 2024).

Fisher, D. K., & Gould, P. J. (2012). Open-Source hardware is a Low-Cost alternative for scientific instrumentation and research. *Modern Instrumentation*, *01*(02), 8–20. https://doi.org/10.4236/mi.2012.12002

Foryard (2019) Foryard Optoelectronics Φ5mm LED FYL-5014UBC1C, Φ5mm_NINGBO foryard Optoelectronics Co.,ltd.. https://www.foryard.com/product/91.html (Accessed: 02 July 2024).

Foster, E. D., & Deardorff, A. (2017). Open Science Framework (OSF). *Journal of the Medical Library Association JMLA*, *105*(2). https://doi.org/10.5195/jmla.2017.88

Galloway, R. B., Hong, D. G., & Napier, H. J. (1997). A substantially improved green-light-emitting diode system for luminescence stimulation. *Measurement Science and Technology, 8*(3), 267–271. https://doi.org/10.1088/0957-0233/8/3/008

Giancoli, C, D... (2009). Physics for Scientists & Engineers with Modern Physics (4th Ed). Upper Saddle River, New Jersey: Pearson Education.

Godfrey-Smith, D., McMullan, W., Huntley, D., & Thewalt, M. (1989). Time-dependent recombination luminescence spectra arising from optical ejection of trapped charges in zircons. Journal of Luminescence, 44(1–2), 47–57. https://doi.org/10.1016/0022-2313(89)90020-3

Gupta, I., Singh, S., Bhagwan, S., & Singh, D. (2021). Rare earth (RE) doped phosphors and their emerging applications: A review. *Ceramics International*, *47*(14), 19282–19303. https://doi.org/10.1016/j.ceramint.2021.03.308

Guérin, G., & Lefèvre, J. (2014). A low cost TL–OSL reader dedicated to high temperature studies. *Measurement*, *49*, 26–33. https://doi.org/10.1016/j.measurement.2013.11.035

Hamamatsu, (2017) Photomultiplier Tubes: Basics and Application. 4th Edition, Hamamatsu.

Hamamatsu Photonics, 2023. SiPM Overview. https://www.hamamatsu.com/resources/pdf/ssd/sipm_technical_information.pdf [Accessed 6 June 2024].

Hemam, R., Singh, L. R., & Singh, S. D. (2022). Investigation of thermoluminescence (TL) and optically stimulated luminescence (OSL) properties of DY3+ doped CAB4O7 nanoparticles. *Journal of Fluorescence*, *32*(6), 2107–2117. https://doi.org/10.1007/s10895-022-03009-x

Hessenbruch, A. (2013). Reader’s Guide to the history of Science. Taylor and Francis.

Hoya Optics - Ultraviolet Transmitting, Visible Absorbing Filters (U) (2020). https://hoyaoptics.com/colored-glass-filters/ultraviolet-transmitting-visible-absorbing-filters-u/ (Accessed: 20 August 2024).

Huntley, D. J., Godfrey-Smith, D. I., & Thewalt, M. L. W. (1985). Optical dating of sediments. *Nature*, *313*(5998), 105–107. https://doi.org/10.1038/313105a0

Hamamatsu Photonics: Photosensor module with gate function H12056P-110 (2022) Photosensor module with gate function H12056P-110 | Hamamatsu Photonics. https://www.hamamatsu.com/eu/en/product/optical-sensors/pmt/pmt-module/current-output-type/H12056P-110.html (Accessed: 02 July 2024).

IXYS Integrated Circuit Division (2019). IX4340 5-Ampere, Dual Low-Side MOSFET Driver. https://www.mouser.com/datasheet/2/240/IX4340-1546052.pdf?srsltid=AfmBOoo5QlHV0PLFygaljYRHQSeAR66MC8w1BLvwm06kydTOlrHykZoZ (Accessed: 17 August).

Jahn, A., Sommer, M., Ullrich, W., Wickert, M., & Henniger, J. (2013). The BeOmax system – Dosimetry using OSL of BeO for several applications. *Radiation Measurements*, *56*, 324–327. https://doi.org/10.1016/j.radmeas.2013.01.069

Jones, L. (2019). Precision Engineering and Motion Control Systems. Springer.

Kasap, S.O. (2017) Principles of Electronic Materials and Devices. 4th ed. New York: McGraw-Hill Education.

Kearfott, K. J., & West, W. G. (2015). An affordable optically stimulated luminescent dosimeter reader utilizing multiple excitation wavelengths. *Applied Radiation and Isotopes*, *104*, 87–99. https://doi.org/10.1016/j.apradiso.2015.06.002

Kenji, T. (1989). Radiation image recording and reproducing method and radiation image storage panel employed for the same.

Kim, H., Park, C., Kim, S. I., Kim, M. C., & Lee, J. (2024). Development of a prototype TL/OSL reader for on-site use in a large-scale radiological accident. *Nuclear Engineering and Technology*. https://doi.org/10.1016/j.net.2024.01.019

Knoll, G.F. (2010) Radiation Detection and Measurement (4th ed.). Hoboken, NJ: John Wiley & Sons.

Koenka, I. J., Sáiz, J., & Hauser, P. C. (2014). Instrumentino: An open-source modular Python framework for controlling Arduino based experimental instruments. *Computer Physics Communications*, *185*(10), 2724–2729. https://doi.org/10.1016/j.cpc.2014.06.007

Kulkarni, M.S. (2008). Development of optically stimulated luminescence reader systems in BARC. India.

Kumar, V., Prasad, S., & Sharma, A., (2018). Surface modification of nanophosphors for enhanced luminescence and stability. Journal of Nanomaterials, 2018, pp. 1-10.

Kuşoğlu Sarıkaya, C. (2011) Optically stimulated luminescence studies on natural fluorites, OpenMETU, MSc Thesis

Lakowicz, J.R. (2006). Principles of Fluorescence Spectroscopy. 3rd ed. Springer.

Landauer Optically Stimulated Luminescence (OSL). (2023). Landauer. https://www.landauer-fr.com/en/knowledge/go-further/osl-technology/ (Accessed: 22 July 2024).

Lapp, T., Kook, M. H., Murray, A. S., Thomsen, K. J., Buylaert, J., & Jain, M. (2015). A new luminescence detection and stimulation head for the Risø TL/OSL reader. *Radiation Measurements*, *81*, 178–184. https://doi.org/10.1016/j.radmeas.2015.02.001

Leitão, U. A. (2024). RLab Calorimetry: an internet-enabled calorimeter using Arduino for local or remote lab activities. *Physics Education*, *59*(4), 045018. https://doi.org/10.1088/1361-6552/ad4539

Li, Y., Wang, Y., & El-Sayed, M. A. (2010). Surface enhanced Raman scattering in nanophosphors: A new sensing platform. Journal of the American Chemical Society, 132(41), 14733-14739.

Lim, C.S., Jeon, D.Y., & Park, H.D., (2020). Development of highly sensitive nanophosphor dosimeters for radiation detection. Radiation Measurements, 135, pp. 106349.

Lim, K. T., & Kim, H. (2021). Feasibility assessment of an SiPM for implementation in an OSL reader. *Nuclear Instruments and Methods in Physics Research Section a Accelerators Spectrometers Detectors and Associated Equipment*, *986*, 164746. https://doi.org/10.1016/j.nima.2020.164746

LiquidCrystal - Arduino reference. (2021). https://www.arduino.cc/reference/en/libraries/liquidcrystal/ (Accessed: 21 Julu 2024).

LM2596 data sheet, product information and support | TI.com. (2012). https://www.ti.com/product/LM2596 (Accessed: 20 August 2024).

Luchechko, A., Kravets, O., Kostyk, L., & Tsvetkova, O. (2016). Luminescence spectroscopy of Eu3+ and Mn2+ ions in $MgGa_2O_4$ spinel. *Radiation Measurements*, *90*, 47–50. https://doi.org/10.1016/j.radmeas.2015.12.003

Lynn, P.A., and Fuerst, W. (2013) Introductory Digital Signal Processing with Computer Applications. 2nd ed. Hoboken, NJ: Wiley.

Makeblock (2016) 2H microstep driver, Me 2h Microsterp Driver. https://www.makeblock.com.cn/en/project/2h-microstep-driver (Accessed: 29 June 2024).

Malickova, I., Fridrichova, J., Bacik, P., Milovska, S., Skoda, R., Illasova, L., & Stubna, J. (2018). Laser effect in the optical luminescence of oxides containing Cr. *Acta Geologica Slovaca*, *10*(1), 27–34. http://geopaleo.fns.uniba.sk/ageos/archive/2018_01/malickova_et_al_2018.pdf

Maraba, D. (2017). Design and implementation of an open-source optically stimulated luminescence measurement system [M.S. - Master of Science]. Middle East Technical University.

Maraba, D., & Bulur, E. (2017). Design and construction of an automated OSL reader with open source software and hardware. *Radiation Measurements*, *106*, 632–637. https://doi.org/10.1016/j.radmeas.2017.04.011

Markey, B., Colyott, L., & Mckeever, S. (1995). Time-resolved optically stimulated luminescence from α-Al_2O_3:C. *Radiation Measurements*, *24*(4), 457–463. https://doi.org/10.1016/1350-4487(94)00119-l

McKeever, S., Akselrod, & Markey, B. (1996b). Pulsed optically stimulated luminescence dosimetry using Alpha-AL_2O_3:C. *Radiation Protection Dosimetry*, *65*(1), 267–272. https://doi.org/10.1093/oxfordjournals.rpd.a031639

McKeever, S. W. S. (2019). An Introduction to Time-Resolved Optically Stimulated Luminescence. *Radiation Protection Dosimetry*. https://doi.org/10.1093/rpd/ncz042

McKeever, S.W. S. (1985). Thermoluminescence of Solids. Cambridge University Press.

McMahon, G., 2008. Analytical instrumentation: a guide to laboratory, portable and miniaturized instruments. J. Wiley & Sons, Chichester.

Microchip. (2005). MCP23017/MCP23S17. Microchip.

https://ww1.microchip.com/downloads/en/DeviceDoc/20001952C.pdf (Accessed: 29 June 2024).

Millman, K.J., & Aivazis, M. (2011). Python for Scientists and Engineers. Computing in Science & Engineering, 13(2), 9-12.

Mittani, J. C. R., Tudela, D. R. G., & Ramirez, A. M. (2019). Construction of a simple thermoluminescence and optically stimulated luminescence reader for luminescence dosimetry research. *Review of Scientific Instruments*, *90*(5). https://doi.org/10.1063/1.5082762

Mobley, C. (2021) Theory of fluorescence and phosphorescence, Theory of Fluorescence and Phosphorescence: Ocean Optics Web Book.

https://oceanopticsbook.info/view/scattering/level-2/theory-fluorescence-and-phosphorescence (Accessed: 27 June 2024).

Monk, S. (2017). Programming Arduino: Getting Started with Sketches. McGraw-Hill Education.

MovingAvg - Arduino reference. (2020). https://www.arduino.cc/reference/en/libraries/movingavg/ (Accessed: 21 July 2024).

Murray, A., Arnold, L. J., Buylaert, J., Guérin, G., Qin, J., Singhvi, A. K., Smedley, R., & Thomsen, K. J. (2021). Optically stimulated luminescence dating using quartz. *Nature Reviews Methods Primers, 1*(1). https://doi.org/10.1038/s43586-021-00068-5

Musa, Y., Hashim, S., Karim, M. K. A., Bakar, K. A., Ang, W. C., & Salehhon, N. (2017). Response of optically stimulated luminescence dosimeters subjected to X-rays in diagnostic energy range. *Journal of Physics Conference Series*, *851*, 012001. https://doi.org/10.1088/1742-6596/851/1/012001

Nagy, V., Horváth, Z., & Wojnárovits, L. (2013). Luminescent properties of nanophosphors and their application in radiation dosimetry. Radiation Physics and Chemistry, 83, 18-23.

Nanto, H., & Okada, G. (2022). Optically stimulated luminescence dosimeters: principles, phosphors and applications. *Japanese Journal of Applied Physics*, *62*(1), 010505. https://doi.org/10.35848/1347-4065/ac9106

National Semiconductor (1995). 74F525 Programmable Counter Datasheet. National Semiconductor.

Nosek, B. A., Ebersole, C. R., DeHaven, A. C., & Mellor, D. T. (2018). The preregistration revolution. *Proceedings of the National Academy of Sciences*, *115*(11), 2600–2606. https://doi.org/10.1073/pnas.1708274114

Nucleonix Systems (2023). TL / OSL RESEARCH READER - Nucleonix Systems. Nucleonix Systems - Manufactures and Suppliers of Nuclear Instruments and Systems. https://www.nucleonix.com/product/tl-osl/ (Accessed: 22 July 2024).

Oliphant, T.E. (2007). Python for Scientific Computing. Computing in Science & Engineering, 9(3), 10-20.

ORTEC (2013) Easy-MCS Hardware and Software User's Manual. Ortec. https://www.ortec-online.com/-/media/ametekortec/manuals/e/easy-mcs-mnl.pdf?la=en&revision=0177c45a-04b7-4858-a493-7d9333cc18e0 (Accessed 23 June 2024)

Pagonis, V., Lawless, J., Chen, R., & Chithambo, M. (2011). Analytical expressions for time-resolved optically stimulated luminescence experiments in quartz. *Journal of Luminescence*, *131*(9), 1827–1835. https://doi.org/10.1016/j.jlumin.2011.04.040

Pearce, J., 2014. Open-source lab: how to build your own hardware and reduce research costs. Elsevier, Amsterdam.

Placko, D., 2007. Fundamentals of instrumentation and measurement. ISTE Ltd., London.

Pololu (2013) SY35ST28-0504A Stepper Motor Datasheet - Pololu. https://www.pololu.com/file/0J688/SY35ST28-0504A.pdf (Accessed: 29 June 2024).

PySerial. (2013). Welcome to pySerial's Documentation — pySerial 3.0 Documentation. https://pythonhosted.org/pyserial/ (Accessed: 21 July 2024).

Rado, S.R. (2013). GitHub - Kroimon/Arduino-SerialCommand: A Wiring/Arduino library to tokenize and parse commands received over a serial port. GitHub. https://github.com/kroimon/Arduino-SerialCommand (Accessed: 21 July 2024).

Raspberry Pi Foundation (2011). Teach, learn, and make with the Raspberry Pi Foundation. https://www.raspberrypi.org/ (Accessed: 7 August 2024).

Risø TL/OSL Reader vs. Lexsyg Smart & Lexsyg Research. (2022). Rotunda Scientific Technologies LLC. https://www.rotundascitech.com/blogs/lf/riso-tl-osl-reader-vs-lexysg-smart-lexysg-research (Accessed: 22 July 2024).

Rodriguez, V., & Bassani, D. M. (2007). Handbook of Photochemistry and Photobiology: Luminescence in Chemistry and Biology. CRC Press.

Ronda, C.R. (2007). Luminescence: From Theory to Applications. Wiley-VCH.

Rossum, G.van, 1995. Python tutorial, Technical Report CS-R9526, Centrum voor Wiskunde en Informatica (CWI), Amsterdam

Rost, F.W. D. (1992). Fluorescence Microscopy. Cambridge University Press.

Ryu, B., Cho, Y.J., & Lee, K.H., (2021). Recent advances in nanophosphor-based optoelectronic devices. Nano Energy, 82, pp. 105738.

Sanborn, E. N. and Beard, E. L. (1967) Sulfides of strontium, calcium and magnesium in infrared-stimulated luminescence dosimetry. In Luminescence Dosimetry, ed. F. H. Attix, pp. 183-193. U.S. Atomic Energy Comission.

Sanderson, D., & Clark, R. (1994). Pulsed photostimulated luminescence of alkali feldspars. *Radiation Measurements*, *23*(2–3), 633–639. https://doi.org/10.1016/1350-4487(94)90112-0

Sang, N. D. (2021). Estimate half-life of thermoluminescence signals according to the different models by using Python. *Journal of Taibah University for Science*, *15*(1), 599–608. https://doi.org/10.1080/16583655.2021.1996113

Sanyo Denki (2001) Stepping Systems, 2-Phase Stepping Systems. https://db.sanyodenki.co.jp/stepping_e/two/pdf/2step_e.pdf (Accessed: 29 June 2024).

Schmidt, C., Simmank, O., & Kreutzer, S. (2019). Time-resolved optically stimulated luminescence of quartz in the nanosecond time domain. *Journal of Luminescence*, *213*, 376–387. https://doi.org/10.1016/j.jlumin.2019.05.042

SCHOTT Advanced Optics | GG420. (2022). https://www.schott.com/shop/advanced-optics/en/Matt-Filter-Plates/GG420/c/glass-GG420 (Accessed: 20 August 2024).

SCHOTT Advanced Optics | BG39. (2022). https://www.schott.com/shop/advanced-optics/en/Matt-Filter-Plates/BG39/c/glass-BG39 (Accessed: 20 August 2024).

Schweizer, T., Kubach, H., & Koch, T. (2021). Investigations to characterize the interactions of light radiation, engine operating media and fluorescence tracers for the use of qualitative light-induced fluorescence in engine systems. *Automotive and Engine Technology*, *6*(3–4), 275–287. https://doi.org/10.1007/s41104-021-00092-3

Shakarji̇, A. M., & Gölcük, A. (2022). Arduino based system design for measuring heart rate and body temperature. *International Journal of Applied Mathematics Electronics and Computers*, *10*(1), 15–19. https://doi.org/10.18100/ijamec.1029843

Shewan, L., O'Reilly, D., Armstrong, R., Toms, P., Webb, J., Beavan, N., Luangkhoth, T., Wood, J., Halcrow, S., Domett, K., Van Den Bergh, J., & Chang, N. (2021). Dating the megalithic culture of laos: Radiocarbon, optically stimulated luminescence and U/Pb zircon results. PLoS ONE, 16(3), e0247167. https://doi.org/10.1371/journal.pone.0247167

Smith, B. (1988): Zircon from sediments: a combined OSL and TL auto-regenerative dating technique, Quaternary Science Reviews, 7, 401-406,

Smith, A. (2020). Principles of Light Control in Scientific Equipment. Oxford University Press.

Smith, J. (2021). Open-Source Hardware and Software in Embedded Systems. McGraw-Hill.

Sommer, M., & Henniger, J. (2006). Investigation of a BeO-based optically stimulated luminescence dosemeter. *Radiation protection dosimetry*, *119*(1-4), 394–397. https://doi.org/10.1093/rpd/nci626

Spooner, N., Aitken, M., Smith, B., Franks, M., & McElroy, C. (1990). Archaeological dating by Infrared-Stimulated luminescence using a diode array. *Radiation Protection Dosimetry*, *34*(1–4), 83–86. https://doi.org/10.1093/oxfordjournals.rpd.a080853

Stallman, R., 2002. Free Software, Free Society: Selected Essays of Richard M. Stallman. Boston: GNU Press.

Stroustrup, B. (2013). The C++ Programming Language. Addison-Wesley.

SZGMLED Co, Ltd [SGZMLED]. (2021). Ceramic (1W) 3535 840-850 GH42. https://www.szgm-led.com/product-21.html

TBS1000B and TBS1000B-EDU Series Digital Storage Oscilloscopes User Manual. (2012, August 3). Tektronix. https://download.tek.com/manual/TBS1000B-User-Manual-077088601_2.pdf (Accessed: 22 July 2024).

The Risø TL/OSL reader. (2024). https://www.fysik.dtu.dk. https://www.fysik.dtu.dk/english/research-sections/lumphys/research/radiation-instruments/tl_osl_reader (Accessed: 22 July 2024).

Thorlabs - DET10A/M SI Detector. (2015). Thorlabs - DET10A/M SI Detector, 200 - 1100 Nm, Ns Rise Time, 0.8 Mm2, M4 Taps. https://www.thorlabs.com/thorProduct.cfm?partNumber=DET10A/M (Accessed: 22 July 2024).

Tsukamoto, S., Kataoka, K., & Miyabuchi, Y. (2013). Luminescence dating of volcanogenic outburst flood sediments from Aso volcano and tephric loess deposits, southwest Japan. *Geochronometria*, *40*(4), 294–303. https://doi.org/10.2478/s13386-013-0135-4

Tkinter. (2012). Tkinter — Python Interface to Tcl/Tk. https://docs.python.org/3/library/tkinter.html (Accessed: 21 July 2024).

Turkin, A., Vainshtein, D., & Hartog, H. D. (2006). Modelling of optically stimulated luminescence of zircon: assessment of the suitability for dating. Journal of Physics D Applied Physics, 39(7), 1367–1377. https://doi.org/10.1088/0022-3727/39/7/006

Valeur, B. (2002). Molecular Fluorescence: Principles and Applications. Wiley-VCH.

Wang, F., Liu, X., & Zhuang, J. (2011). Phosphorescent nanomaterials for high-resolution bioimaging. Chemical Society Reviews, 40(9), 4615-4636.

Wire - Arduino reference. (2022).

https://www.arduino.cc/reference/en/language/functions/communication/wire/ (Accessed: 21 July 2024).

Wouter, C., Dirk, V., Paul, L., & Tom, D. (2017). A reusable OSL-film for 2D radiotherapy dosimetry. *Physics in Medicine and Biology*, *62*(21), 8441–8454. https://doi.org/10.1088/1361-6560/aa8de6

XP Power (2019) AC - DC 120-180 watts SDS series - XP power, AC - DC 120-180 Watts SDS Series - XP Power. https://www.xppower.com/portals/0/pdfs/SF_SDS120-180.pdf (Accessed: 29 June 2024).

Xu, Z., Chen, L., & Wang, J., 2022. Multifunctional nanophosphors for advanced radiation dosimetry. Advanced Functional Materials, 32(8), pp. 2108765.

Yeltik A. (2009) Time-Resolved Optically Stimulated Luminescence (OSL) Studies on Samples Containing Quartz and Feldspar, OpenMETU, MSc Thesis

Yuan, L., Jin, Y., Su, Y., Wu, H., Hu, Y., & Yang, S. (2020). Optically Stimulated Luminescence Phosphors: Principles, applications, and Prospects. *Laser & Photonics Review*, *14*(12). https://doi.org/10.1002/lpor.202000123

Yukihara, E.G., McKeever, S.W.S., (2011a). Optically stimulated luminescence: fundamentals and applications. Wiley, Chichester, West Sussex.

Yukihara, E. G. (2011b). Luminescence properties of BeO optically stimulated luminescence (OSL) detectors. *Radiation Measurements*, *46*(6–7), 580–587. https://doi.org/10.1016/j.radmeas.2011.04.013

Yukihara, E.G., Mckeever, S.W.S., Akselrod, M.S., (2014). State of art: Optically stimulated luminescence dosimetry – Frontiers of future research. Radiat. Meas. 71, 15–24.

Yukihara, E., Andrade, A., & Eller, S. (2016). BeO optically stimulated luminescence dosimetry using automated research readers. Radiation Measurements, 94, 27–34. https://doi.org/10.1016/j.radmeas.2016.08.008

Yukihara, E. G. (2023). TL and OSL as research tools in luminescence: Possibilities and limitations. *Ceramics International*, *49*(14), 24356–24369. https://doi.org/10.1016/j.ceramint.2022.10.199

APPENDICES

A. Wiring Diagrams

This appendix contains the wiring diagrams of this research. For reference, Figure 9.1 contains the essential components, and said components will be linked to the wiring diagrams in the following figures.

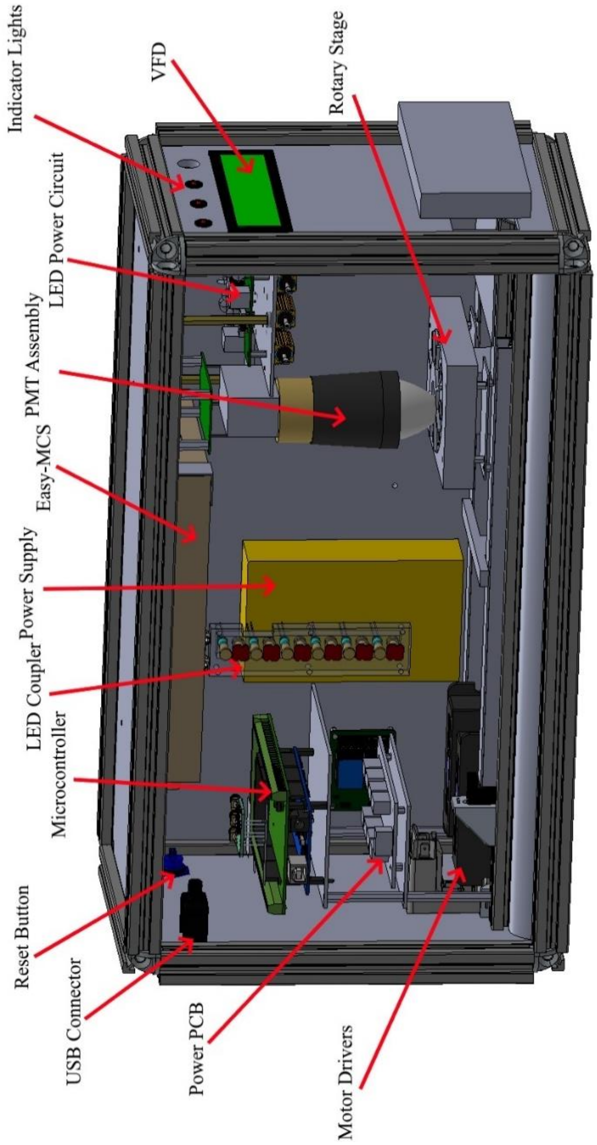

Figure 9.1. Overview of the System and its key components

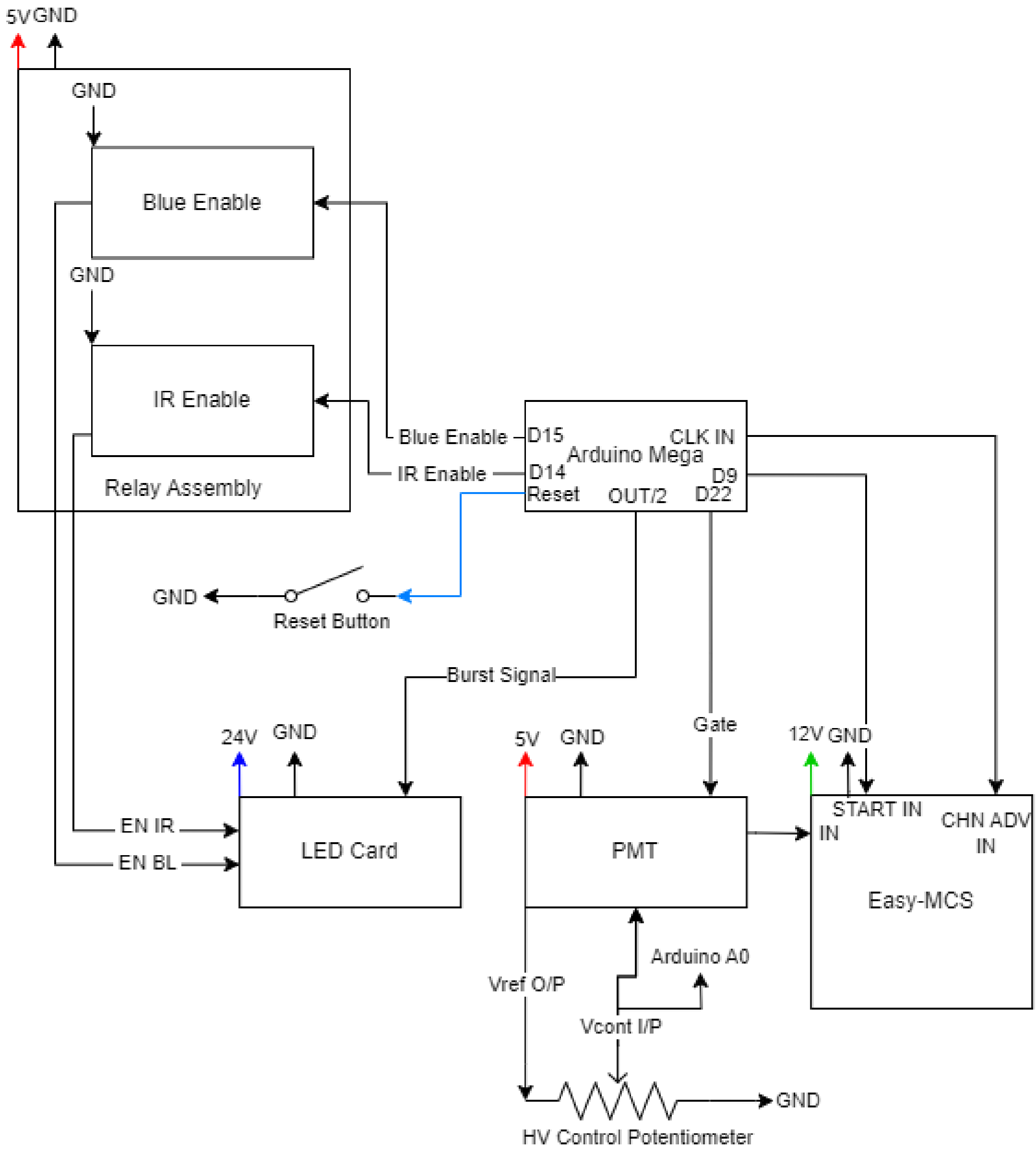

Figure 9.2. Wiring Diagram of DAQ System

The data acquisition system is centered around the Arduino. The microcontroller triggers the Easy-MCS' acquisition passes and channels within the acquisition pass using two signals. The microcontroller also controls the stimulation burst generation and enable signals for the blue and IR LEDs, as well as the gate signal for the PMT. An emergency reset button was placed at the back of the experimental setup to be able to hardware reset the microcontroller in case software and/or hardware fails. The connections for said functions of the system can be found in Figure 9.2.

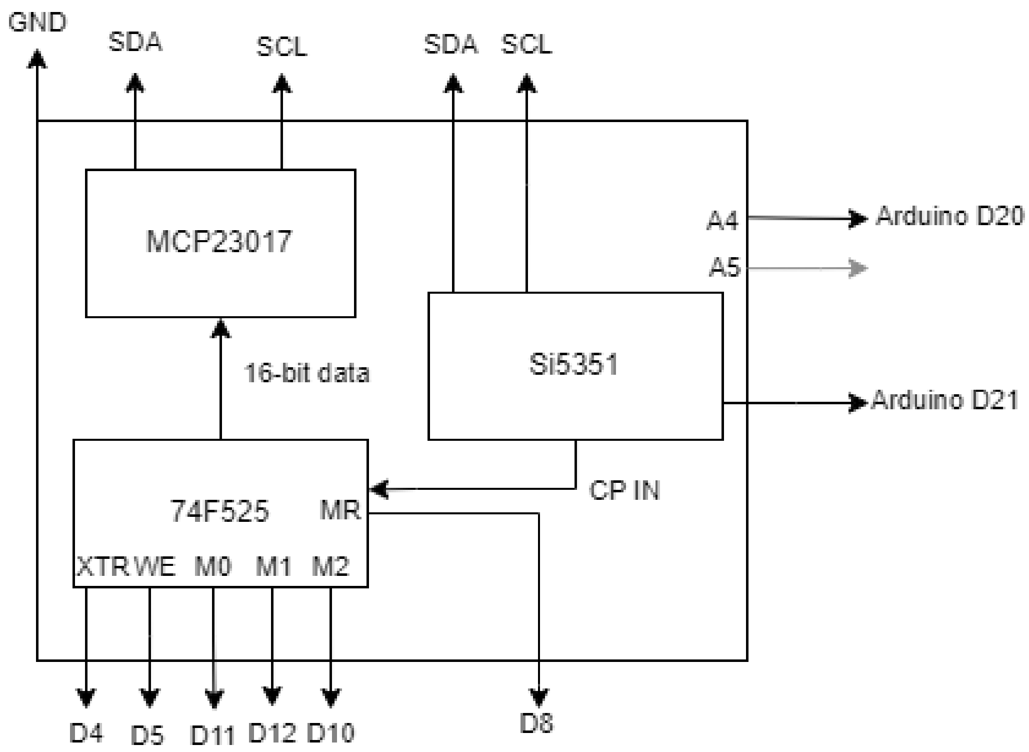

Figure 9.3. Wiring Diagram of the Burst Generator PCB

As can be followed in Figure 8.3, the microcontroller has a burst generator shield assembly to generate the pulses. This shield is connected to the microcontroller with I^2C and is compatible with any board that supports the protocol; however, its original configuration is to use this protocol over A4/A5 pins. If one wants to use another type of Arduino, one should bypass the correct I^2C pins. The circuit schematic of the clock generator is given in Figure 9.4.

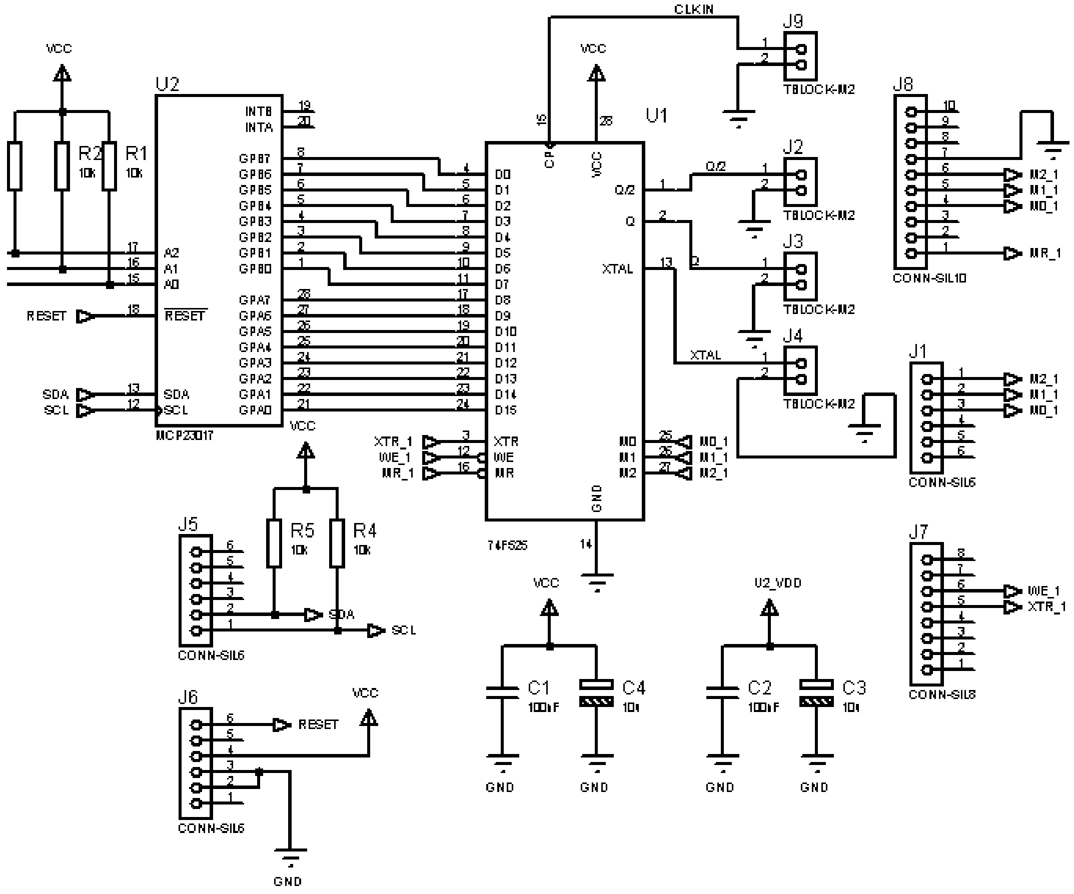

Figure 9.4. Circuit Schematic of the Burst Generator PCB

Wiring of the movement system was done in a straightforward fashion, for each stage (linear and rotary), a driver and an end-stop sensor were used. For the rotary stage, a hall effect sensor was used in the "home" position, which also corresponds to sample one. In the linear stage, the exit location was predefined by the software, the zero position was defined by a commercially available optocoupler. The drivers, therefore the motors are being controlled by the microcontroller. The connections can be followed in Figure 9.5.

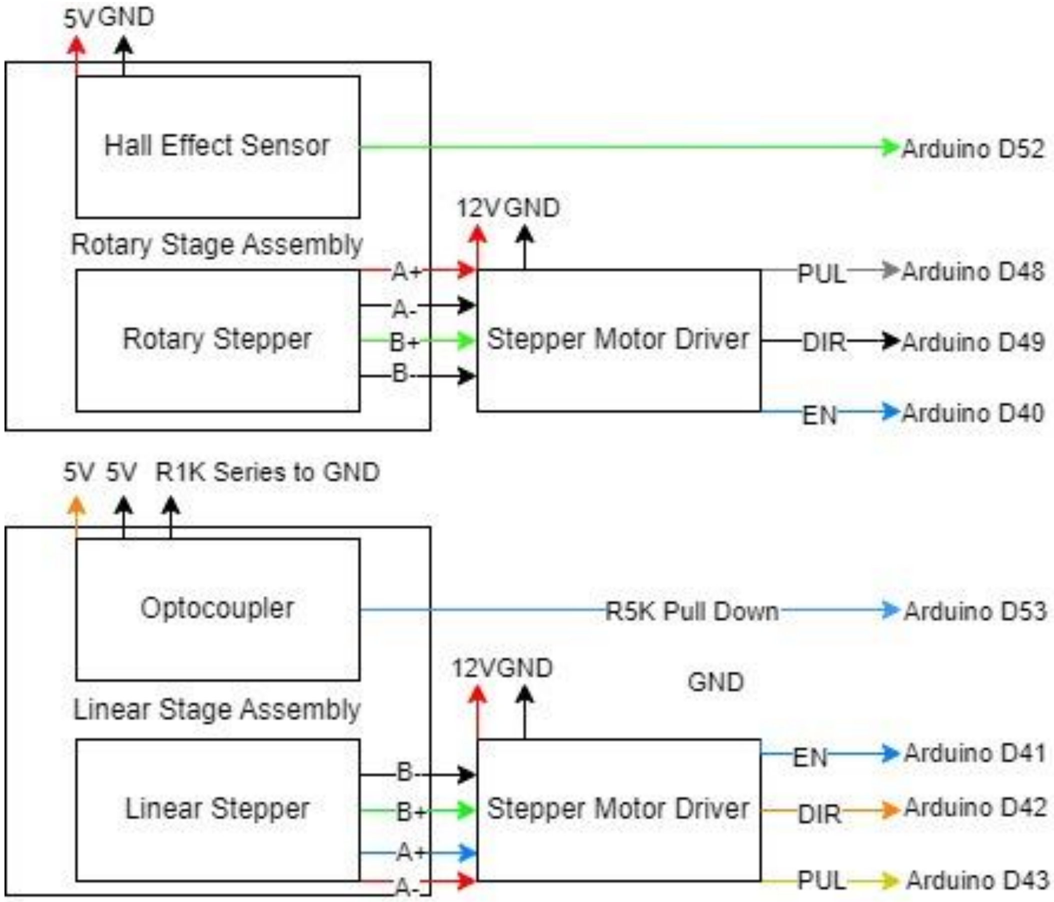

Figure 9.5. Wiring Diagram for the Movement Subsystem

The LED Power circuitry handles the propagation delay, the pulse width, and the amplification of the burst signal to be fed into the LEDs. This is made possible by using two 74F123 monostable multivibrators and a MOSFET gate driver in a daisy chain. The potentiometers change the delay and the width, while the gate driver's V_{cc} handles the amplification. The output of the circuit should be calibrated with appropriate resistors (for current limiting) and with proper capacitors (for the load requirements of the driver). The circuit diagram is given below.

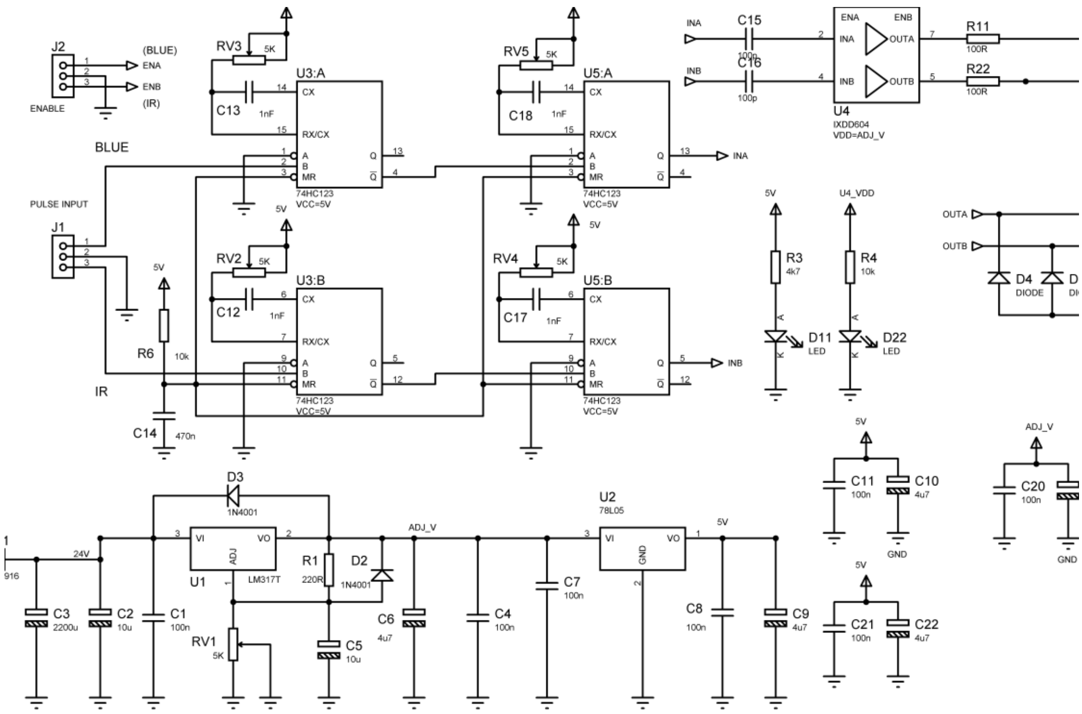

Figure 9.6. Circuit Schematic of the LED Driver PCB

The system is powered by the 220/240VAC mains. This is then converted into 24VDC by an off-the-shelf power supply. This 24VDC is further converted into 5VDC and 12VDC using buck converters. The outputs of these 3 power supplies and their respective grounds are carried to a power distribution PCB where there are two 24VDC outputs, six of each 12VDC and 5VDC outputs, and twelve ground connectors via terminal blocks. The main wiring can be seen in Figure 9.7.

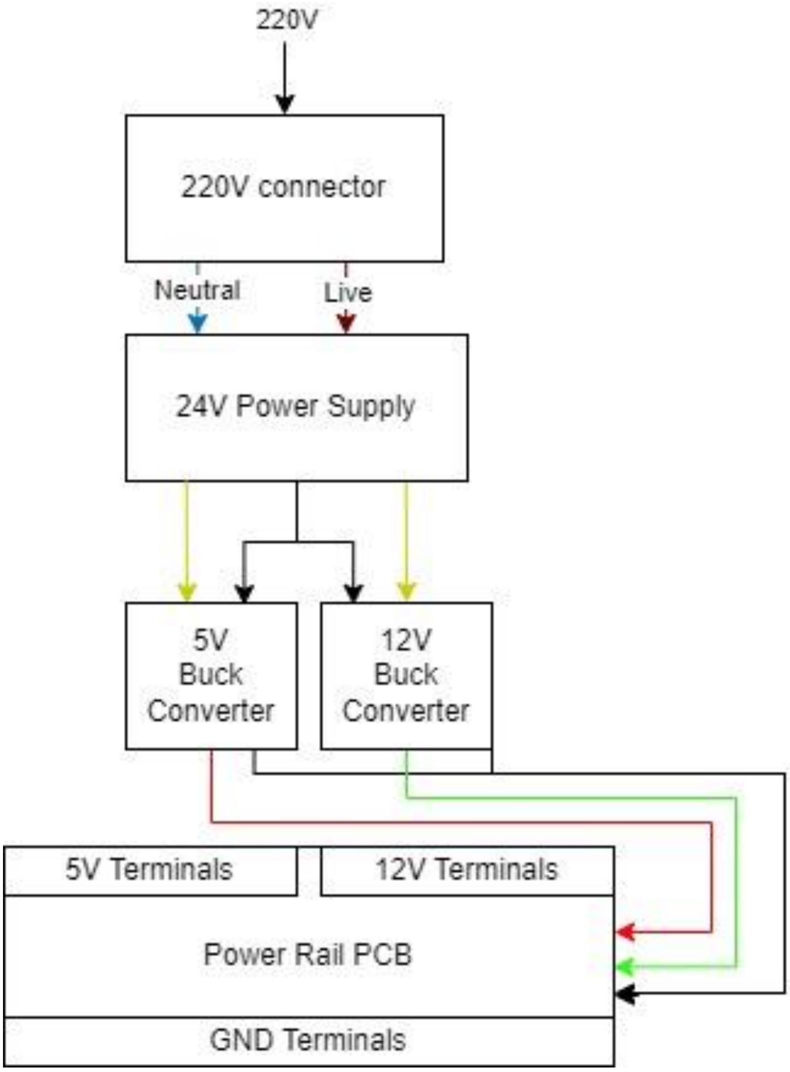

Figure 9.7. Wiring Diagram for the Power Distribution System

The front panel of the instrument carries the VFD, the 100k multi-turn wire wound potentiometer for HV adjustment, and three LEDs to indicate power supply status. The VFD Screen is directly controlled by the microcontroller, whilst the indicator lights are directly connected to the respective power supplies. The PMT HV Dial is connected to the PMT to take a reference voltage and be used as a voltage divider. The output of the potentiometer is then fed back into the PMT and the same output is monitored by the microcontroller to be written onto the VFD Screen. These connections are given in Figure 9.8.

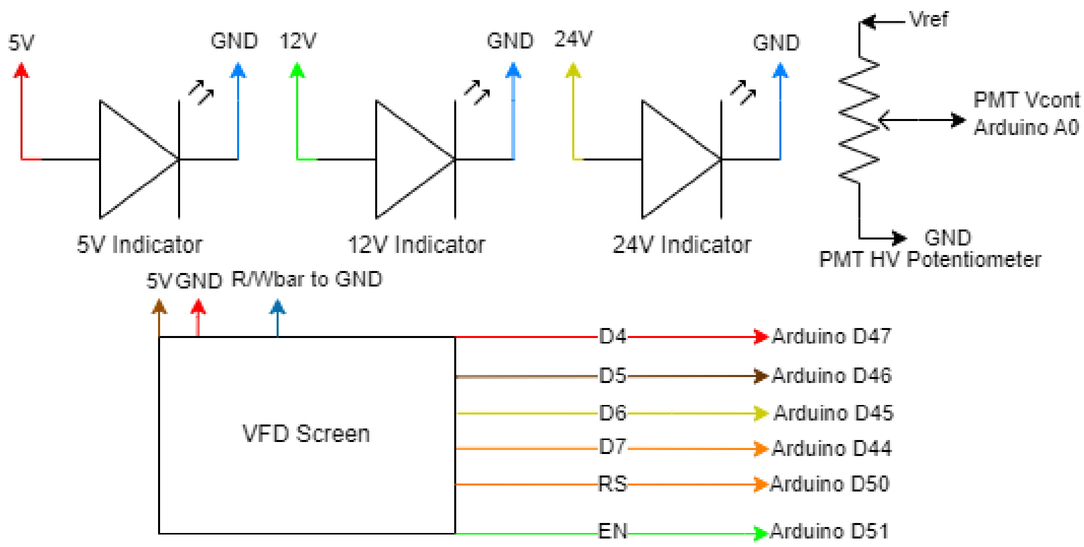

Figure 9.8. Wiring Diagram of the Front Panel

B. Images of the Experimental Setup

In this appendix, various images of the experimental setup will be shown. The enclosed system and the front panel of the setup can be seen in Figure 9.9.

Figure 9.9. Setup Enclosure and Front Panel

The Easy-MCS was placed on the top of the experiment system. An image of how the MCS is mounted can be seen in Figure 9.10, and its location in the interior of the system can be seen in Figure 9.11.

Figure 9.10. Setup Interior and Components

This enclosure houses every component used in this experiment. The interior of the enclosure and components can be seen in Figure 9.11.

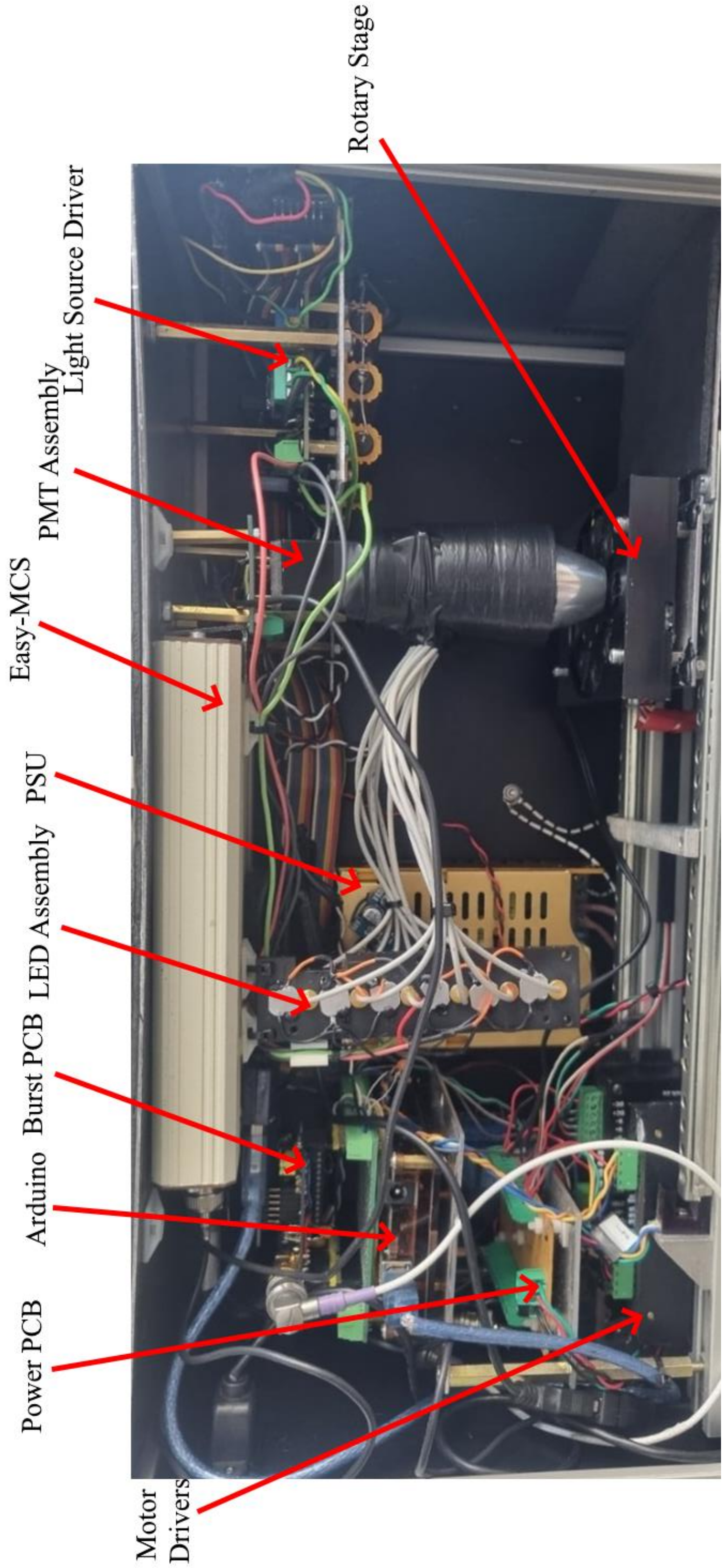

Figure 9.11. Setup Interior and Components

In figure 9.12, the light source driver PCB can be seen. This driver was placed on a platform attached to the top side of the setup.

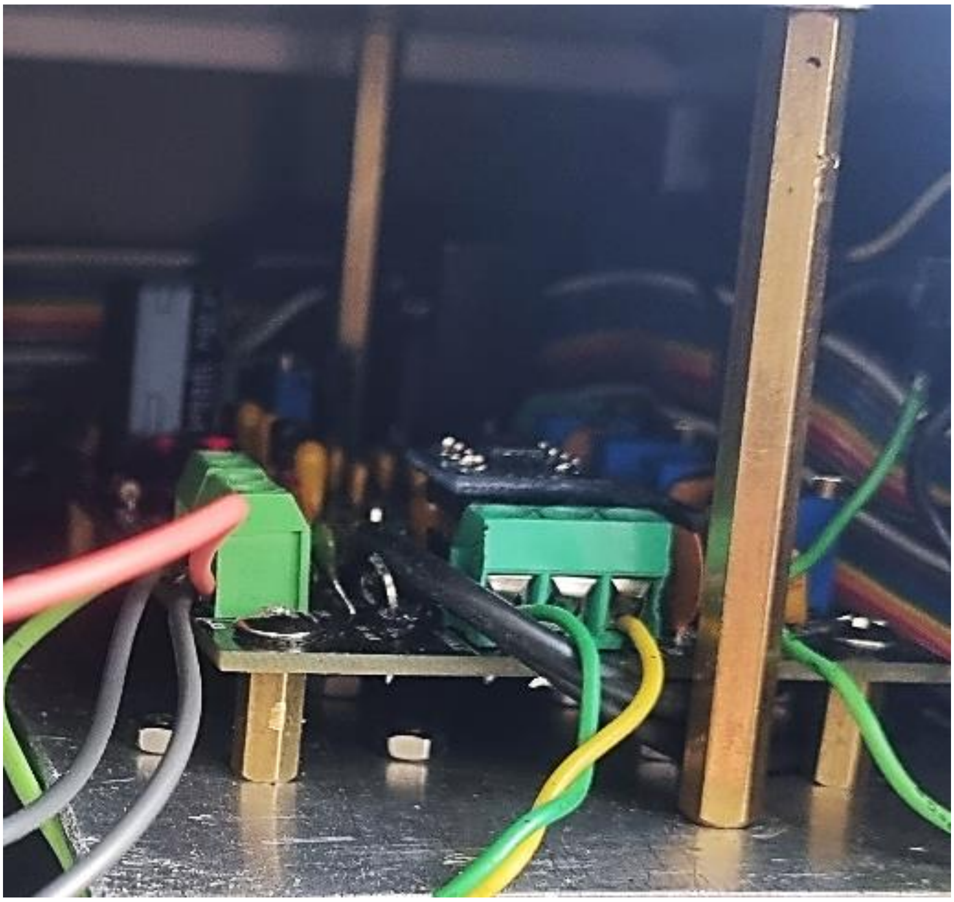

Figure 9.12. Light Source Driver

In Figure 9.13, at the bottom, the Arduino enclosed in a plastic box can be seen. A screw shield was placed on the microcontroller to simplify and strengthen the cable connections. On the screw shield, the burst generator PCB, with the Si5351 module at the top, MCP23017 behind the Si5351, and 74F525 at the bottom side of the PCB can be seen. The BNC connector used to carry the DAQ dwell time signal to the Easy-MCS can be also seen here.

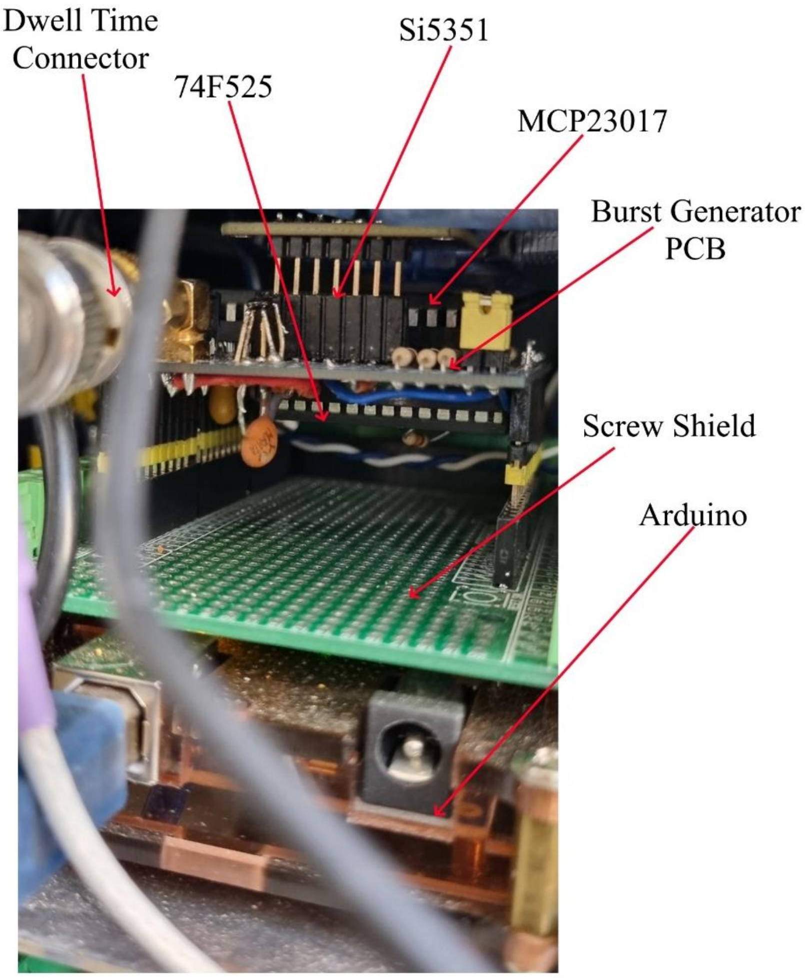

Figure 9.13. Arduino, Burst Generator PCB, and the Si5351

In the following figure, the fiber head/lens/mirror holder and the parabolical mirror can be seen. Underneath the mirror, the rotary stage with its sample trays and sample separator can also be noticed. The distance between the opening of the mirror and the sample tray was set to 2.5 mm after the system was photographed. To the right of the same figure, the PMT can be seen. In Figure 9.14, a sample stimulation can be found. Instead of samples, circular sheets of paper with a radius of 4.6 mm were used. It can be seen from the photograph that the stimulation light does not affect the adjacent samples.

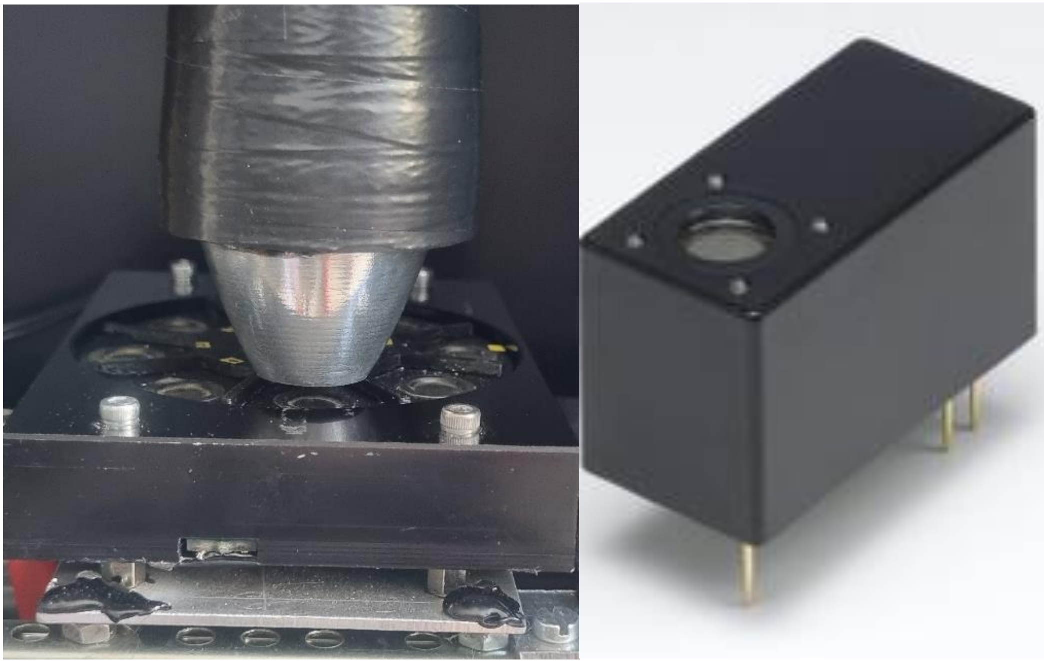

Figure 9.14. Left Side-Rotary Stage and the Mirror Assembly, Right Side-PMT

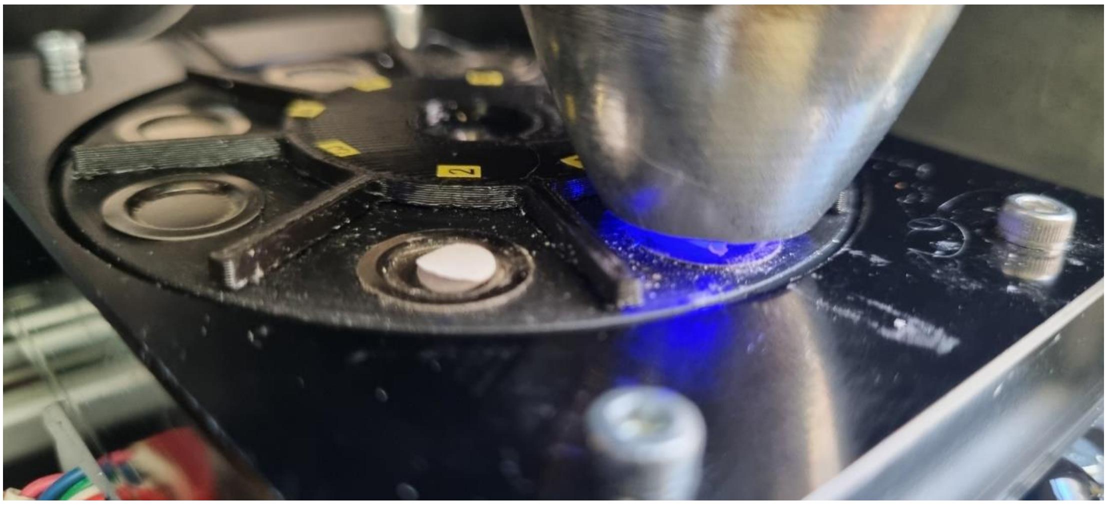

Figure 9.15. Illumination of the Sample Tray

C. Software and Codes

All codes and design files can be accessed via the following GitHub link.

https://github.com/metalli-guy/thesis-work

9.1.1 Arduino

```
/***************************************************
  License:CC4, 2024, Deniz Boztemur
  File Name: burstgenerator.ino
  Description: Arduino Code for the August 2024 Thesis of Deniz Boztemur.
  Revision History: v1.1
***************************************************/
//libraries used
#include <SerialCommand.h>
#include <MCP23017.h>
#include "si5351.h"
#include <Wire.h>
#include <AccelStepper.h>
#include <LiquidCrystal.h>
#define MCP23017_I2C_ADDRESS 0x20
#include <movingAvg.h>
```

```
MCP23017 mcp23017 = MCP23017(MCP23017_I2C_ADDRESS);

SerialCommand sCmd;

Si5351 si5351;

//rotary stage stepper pins

const int ROTARYENABLE = 40;

int dirPin = 49;

int pulPin = 48;

long initial_homing = -1;

int interruptPin = 52;

AccelStepper stepper(AccelStepper::DRIVER, pulPin, dirPin);

//linear stage stepper pins

const int LINEARENABLE = 41;

int LdirPin = 42;

int LpulPin = 43;

int interruptPin2 = 53;

AccelStepper Lstepper(AccelStepper::DRIVER, LpulPin, LdirPin);

//vfd screen pins

//vfd RS pin to digital pin 12

//vfd Enable pin to digital pin 11

//vfd D4 pin to digital pin 5

//vfd D5 pin to digital pin 4

//vfd D6 pin to digital pin 3
```

```
//vfd D7 pin to digital pin 2

//vfd R/W pin to ground

const int rs = 50, en = 51, d4 = 47, d5 = 46, d6 = 45, d7 = 44;

LiquidCrystal vfd(rs, en, d4, d5, d6, d7);

//burst generator PCB pins, all connected via hardware

const int XTR = 4;

const int WE = 5;

const int MR = 8;

const int M2 = 10;

const int M1 = 12;

const int M0 = 11;

//blue and infrared power pins

const int blLP = 15;

const int irLP = 14;

//PMT enable pin

const int pmtena = 22;

//acquisition signal pin

const int STARTIN = 9;

//PMT high voltage readout pin

const int pmtv = A0;

//create empty string for Si5351

String inString = "";
```

```
//create empty string for acquisition

String controlString = "";

//number of pulses

uint16_t aNumberrec;

//pulse number to be written in 74F525

uint16_t aNumber;

//dwell time frequency for Si5351

uint32_t freqmult;

uint16_t invertA;

uint8_t bNumber;

uint8_t invertB;

uint8_t cNumber;

uint8_t invertC;

//moving average number of points

movingAvg pmtbias(50);

void setup() {

	pinMode(WE, OUTPUT);

	pinMode(XTR, OUTPUT);

	pinMode(MR, OUTPUT);

	pinMode(M2, OUTPUT);

	pinMode(M1, OUTPUT);

	pinMode(M0, OUTPUT);
```

```
	pinMode(irLP, OUTPUT);
	pinMode(blLP, OUTPUT);
	pinMode(pmtena, OUTPUT);
	pinMode(STARTIN, OUTPUT);
	pinMode(LINEARENABLE, OUTPUT);
	pinMode(ROTARYENABLE, OUTPUT);
//set stage sensor pins and PMT HV pin as input
	pinMode(interruptPin, INPUT);
	pinMode(interruptPin2, INPUT);
	pinMode(pmtv, INPUT);
//sets 74f525 to mode 6
	digitalWrite(M2, HIGH);
	digitalWrite(M1, HIGH);
	digitalWrite(M0, LOW);
	digitalWrite(STARTIN, LOW);
//high disables all 3 power switches
	digitalWrite(pmtena, HIGH);
	digitalWrite(irLP, HIGH);
	digitalWrite(blLP, HIGH);
	Serial.begin(9600);
//disable stepper motors
	digitalWrite(LINEARENABLE, LOW);
```

```
digitalWrite(ROTARYENABLE, LOW);

//function calls for serial command
        sCmd.addCommand("P",     countNumber);
        sCmd.addCommand("F",     frequencyCommand);
        sCmd.addCommand("L", latch);
        sCmd.addCommand("START", writepulse);
        sCmd.addCommand("IRPOWER", irledPoweron);
        sCmd.addCommand("IRPOWERO", irledPoweroff);
        sCmd.addCommand("BLPOWER", blPoweron);
        sCmd.addCommand("BLPOWERO", blPoweroff);
        sCmd.addCommand("PMTEN", pmten);
        sCmd.addCommand("PMTDIS", pmtdis);
        sCmd.addCommand("HOME",    homing);
        sCmd.addCommand("ONE",   sampleone);
        sCmd.addCommand("TWO",   sampletwo);
        sCmd.addCommand("THREE",   samplethree);
        sCmd.addCommand("FOUR",   samplefour);
        sCmd.addCommand("FIVE",   samplefive);
        sCmd.addCommand("SIX",   samplesix);
        sCmd.addCommand("SEVEN",   sampleseven);
        sCmd.addCommand("EIGHT",   sampleeight);
```

```
sCmd.addCommand("LHOME", lhoming);
sCmd.addCommand("EJECT", eject);
Wire.begin();
```

//initialize mcp23017, set all pins as outputs, define both to be used and set 0 on both

```
mcp23017.init();
mcp23017.portMode(MCP23017Port::A, 0b00000000);
mcp23017.portMode(MCP23017Port::B, 0b00000000);
mcp23017.writeRegister(MCP23017Register::GPIO_A, 0x00);
mcp23017.writeRegister(MCP23017Register::GPIO_B, 0x00);
```

//start si5351 and set 8mA drive on both channels

```
si5351.init(0, 0, 0);
si5351.drive_strength(SI5351_CLK0, SI5351_DRIVE_8MA);
si5351.drive_strength(SI5351_CLK1, SI5351_DRIVE_8MA);
si5351.drive_strength(SI5351_CLK2, SI5351_DRIVE_8MA);
```

//begin vfd and pmt HV movingavg function

```
vfd.begin(24, 2);
pmtbias.begin();
```

//set stepper parameters, call both stages to initial position

```
stepper.setMaxSpeed(100);
stepper.setAcceleration(3000);
Lstepper.setMaxSpeed(1000);
```

```
            Lstepper.setAcceleration(20000);
            initialhoming();
  //set AREF to 2.56V
            analogReference(INTERNAL2V56);
            Serial.println("hazırım hacı");
            vfd.clear();
            vfd.write("System Ready");}
void countNumber() {
  //this function takes a number from the serial port in the form of "P X"
  //this number is split into 2-8 bit numbers, reversed by reverse_bits function
  //then added to each other and written into MCP23017 ports
            char *arg;
            arg = sCmd.next();
            if (arg != NULL) {
              aNumberrec = atoi(arg);
              aNumber = (aNumberrec * 2) + 1;
              if (aNumber == 0) {
                countNumber;    }
              invertA = reverse_bits(aNumber);
              bNumber = aNumber >> 8;
              invertB = reverse_bits(bNumber);
              cNumber = aNumber;
```

```
    invertC = reverse_bits(cNumber);
    mcp23017.writePort(MCP23017Port::B, invertC);
    mcp23017.writePort(MCP23017Port::A, invertB);  }
   vfd.clear();
   Serial.println("Pulse Count Received");
   vfd.write("Pulse Count Received");}
void latch() {
 //this function latches the number of pulses into 74F525
   digitalWrite(MR, LOW);
   delay(100);
   digitalWrite(MR, HIGH);
   delay(100);
   digitalWrite(WE, LOW);
   delay(100);
   digitalWrite(WE, HIGH);
   vfd.clear();
   Serial.println("System Armed");
   vfd.write("System Armed");}
void writepulse() {
 //this is the main functin of the experiment
 //takes two parameters, number of acquisitions and time between acquisitions
 //first parameter is the number of acquisitions, second is the time
```

//in the for loop, triggers the counting process of 7F525, then sends the start signal to MCS-32

//repeats this for the number of acquisitions

//waits for the delay between acquisitions

//first two nop's are for stability

//portg | b0010000 sets XTR high

//following 8 nops is for the XTR to be settled

//portg & b11011111 sets XTR low

//porth | b01000000 sets startin high

//following nops are a requirement for easymcs to register an external trigger signal

//porth & b10111111 sets startin low

//rest of the nops are for stability

```
uint64_t passnumber;
uint32_t delaytime;
char *arg;
vfd.clear();
vfd.write("Experiment Running");
arg = sCmd.next();
controlString += arg;
if (controlString != NULL) {
  passnumber = controlString.toInt();  }
arg = sCmd.next();
```

```
        if (arg != NULL) {
          delaytime = atoi(arg);  }
        for (uint64_t i = 0; i <= (passnumber - 1); i++) {
          _NOP();    _NOP();
          PORTG = PORTG | B00100000;  //this is digitalhigh
          _NOP(); _NOP(); _NOP(); _NOP(); _NOP(); _NOP(); _NOP(); _NOP();
          PORTG = PORTG & B11011111; //this is digitallow
          PORTH = PORTH | B01000000; //this is digitalhigh
          _NOP(); _NOP(); _NOP(); _NOP();
          PORTH = PORTH & B10111111; //this is digitallow
          _NOP(); _NOP();
          delay(delaytime);  }
        controlString = "";
        vfd.clear();
        vfd.write("Experiment Done");
        Serial.println("Experiment Done");}
byte reverse_bits(byte inverting) {
  //this function reverses the order of bits of an 8-bit number
        byte  rtn = 0;
        for (byte i = 0; i < 8; i++) {
          bitWrite(rtn, 7 - i, bitRead(inverting, i));  }
        return rtn;}
```

```
void movingavg() {
  //this function takes a moving average of the specified analogread
        int raw = analogRead(pmtv);
        float biasvoltage = raw * 2.5;
        int avg = pmtbias.reading(biasvoltage);
        vfd.setCursor(20, 2);
        vfd.println(avg);
        delay(10);}
void initialhoming() {
  //this is the initial homing function
  //where both stages are called to their home position
  //when sensors are not triggered, turns the steppers until they are
  //sets absolute position for both home positions
        vfd.clear();
        vfd.write("Homing");
        digitalWrite(LINEARENABLE, HIGH);
        while (digitalRead(interruptPin2)) {
          Lstepper.move(10850);
          Lstepper.run();  }
        Lstepper.setCurrentPosition(0);
        digitalWrite(LINEARENABLE, LOW);
```

digitalWrite(ROTARYENABLE, HIGH);

while (digitalRead(interruptPin)) {

stepper.moveTo(11000);

stepper.run();

delay(5); }

stepper.setCurrentPosition(0);

digitalWrite(ROTARYENABLE, LOW);}

void **pmten()** {

//when LOW, enables PMT HV

digitalWrite(pmtena, LOW);

Serial.println("pmt power enabled");}

void **pmtdis()** {

//when HIGH, disables PMT HV

digitalWrite(pmtena, HIGH);

Serial.println("pmt power disabled");}

void **irledPoweron()** {

//when LOW, enables IR LED power

digitalWrite(irLP, LOW);

Serial.println("ir power enabled");}

void **irledPoweroff()** {

//when HIGH, disables IR LED power

digitalWrite(irLP, HIGH);

```
            Serial.println("ir power disabled");}
void blPoweron() {
  //when LOW, enables Blue LED power
            digitalWrite(blLP, LOW);
            Serial.println("blue power enabled");}
void blPoweroff() {
  //when HIGH, disables Blue LED power
            digitalWrite(blLP, HIGH);
            Serial.println("blue power disabled");}
void frequencyCommand() {
  //this function sets the main dwell times
  //74f525 count rate is set to 125ns in this function by basefreq
  //led pulse width is set to 250ns using ledfreq
  //daq dwell time is calculated in python, then written here
  //Si5351 expects desired frequency x100 as an input
  //function takes desired frequency from serial port and multiplies
  //then writes into Si5351
            char *argu;
            uint32_t ledfreq = 400000000;
            uint32_t basefreq = 800000000;
            argu = sCmd.next();
            inString += argu;
```

```
        if (inString != NULL) {
          freqmult = (inString.toInt());
          freqmult = freqmult * 100;
          inString = "";
          si5351.set_freq(basefreq, SI5351_CLK0);
          si5351.set_freq(ledfreq, SI5351_CLK1);
          si5351.set_freq(freqmult, SI5351_CLK2);
          si5351.pll_reset(SI5351_PLLA);
          si5351.update_status();  }
        vfd.clear();
        vfd.write("Dwell Time Received");
        Serial.println("Dwell Time Received");}
void lhoming() {
  //function to load tray into the experiment.
  //spins the motor until sensor is triggered
  //also sets an absolute position for the home position
        digitalWrite(LINEARENABLE, HIGH);
        vfd.clear();
        vfd.write("Loading Tray");
        while (digitalRead(interruptPin2)) {
          Lstepper.move(10850);
          Lstepper.run();  }
```

```
            vfd.clear();
            vfd.write("Tray Loaded");
            Serial.println("Tray Loaded");
            Lstepper.setCurrentPosition(0);
            digitalWrite(LINEARENABLE, LOW);}
void eject() {
  //function to eject tray from the experiment.
  //spins the motor for a fixed number of steps.
            digitalWrite(LINEARENABLE, HIGH);
            vfd.clear();
            vfd.write("Ejecting Tray");
            Lstepper.moveTo(-9500);
            Lstepper.runToPosition();
            Lstepper.run();
            delay(5);
            vfd.clear();
            vfd.write("Tray Ejected");
            Serial.println("Tray Ejected");
            digitalWrite(LINEARENABLE, LOW);}
void homing() {
  //function to home the rotary stage.
  //home = sample1
```

```
  //spins the motor until sensor is triggered
        digitalWrite(ROTARYENABLE, HIGH);
        vfd.clear();
        vfd.write("Rotary Homing");
        while (digitalRead(interruptPin)) {
          stepper.moveTo(11000);
          stepper.run();
          delay(5);  }
        stepper.setCurrentPosition(0);
        vfd.clear();
        vfd.write("Rotary Home");
        Serial.println("Rotary Home");
        digitalWrite(ROTARYENABLE, LOW);}
void sampleone() {
  //function to first home, then turn the stage to sample position 1
        homing();
        digitalWrite(ROTARYENABLE, HIGH);
        delay(5);
        vfd.clear();
        vfd.write("Sample One");
        Serial.println("1");
        digitalWrite(ROTARYENABLE, LOW);}
```

```
void sampletwo() {
 //function to first home, then turn the stage to sample position 2
        homing();
        digitalWrite(ROTARYENABLE, HIGH);
        stepper.moveTo(-200);
        stepper.runToPosition();
        delay(5);
        vfd.clear();
        vfd.write("Sample Two");
        Serial.println("2");
        digitalWrite(ROTARYENABLE, LOW);}
void samplethree() {
 //function to first home, then turn the stage to sample position 3
        homing();
        digitalWrite(ROTARYENABLE, HIGH);
        stepper.moveTo(-420);
        stepper.runToPosition();
        delay(5);
        vfd.clear();
        vfd.write("Sample Three");
        Serial.println("3");
```

```
        digitalWrite(ROTARYENABLE, LOW);}
void samplefour() {
  //function to first home, then turn the stage to sample position 4
        homing();
        digitalWrite(ROTARYENABLE, HIGH);
        stepper.moveTo(-600);
        stepper.runToPosition();
        delay(5);
        vfd.clear();
        vfd.write("Sample Four");
        Serial.println("4");
        digitalWrite(ROTARYENABLE, LOW);}
void samplefive() {
  //function to first home, then turn the stage to sample position 5
        homing();
        digitalWrite(ROTARYENABLE, HIGH);
        stepper.moveTo(-820);
        stepper.runToPosition();
        delay(5);
        vfd.clear();
        vfd.write("Sample Five");
        Serial.println("5");
```

```
            digitalWrite(ROTARYENABLE, LOW);}
void samplesix() {
  //function to first home, then turn the stage to sample position 6
            homing();
            digitalWrite(ROTARYENABLE, HIGH);
            stepper.moveTo(-1000);
            stepper.runToPosition();
            delay(5);
            vfd.clear();
            vfd.write("Sample Six");
            Serial.println("6");
            digitalWrite(ROTARYENABLE, LOW);}
void sampleseven() {
  //function to first home, then turn the stage to sample position 7
            homing();
            digitalWrite(ROTARYENABLE, HIGH);
            stepper.moveTo(-1210);
            stepper.runToPosition();
            delay(5);
            vfd.clear();
            vfd.write("Sample Seven");
            Serial.println("7");
```

```
        digitalWrite(ROTARYENABLE, LOW);}
void sampleeight() {
 //function to first home, then turn the stage to sample position 8
        homing();
        digitalWrite(ROTARYENABLE, HIGH);
        stepper.moveTo(-1400);
        stepper.runToPosition();
        delay(5);
        vfd.clear();
        vfd.write("Sample Eight");
        Serial.println("8");
        digitalWrite(ROTARYENABLE, LOW);}
void loop() {
 //infinite loop reads the serial port
 //on initialization, calls the moving average function
 //as soon as another function is called, moving average is stopped
        sCmd.readSerial();
        movingavg();}
```

9.1.2 Python

```
/******************************************************
*License:CC4, 2024, Deniz Boztemur
*File Name: interface.py
*Description: Python Code for the August 2024 Thesis of Deniz Boztemur.
*Revision History: v1.1
************************************************/
#importing required packages
import tkinter
from tkinter import ttk
from tkinter import *
from tkinter import messagebox
import time
import serial.tools.list_ports
import os
from tkinter import filedialog as fd
import math
import psutil
#initializing and setting global variables
frequency = 0
illuminate = 0
binNumber = 0
```

```
dwelltime = 0
totaltime = 0
pos = ""
powerstatus = ""
global ser

#automatic connect function for Arduino. Finds and connects to an original Arduino plugged into the USB
arduino_ports = [
    p.device
    for p in serial.tools.list_ports.comports()
    if 'Arduino' in p.description]
if not arduino_ports:
    messagebox.showerror("Error", "No Arduino Found")
if len(arduino_ports) > 1:
    messagebox.showwarning("Warning", "Multiple Arduino found - using the first")
ports = serial.tools.list_ports.comports()
time.sleep(1)
def run():
    #function to define the file path of MCS-32.exe
    try:
        os.startfile(file_path)
```

```
        def check_if_process_running(process_name):
            for process in psutil.process_iter(['name']):
                if process.info['name'] == process_name:
                    softwarecanvas.itemconfig(softwareindicator, fill="green",
outline="green")
        check_if_process_running("Mcs32.exe")
    except NameError:
        messagebox.showerror("Error", "Please select the file path from the menu
bar")
    except Exception:
        messagebox.showerror('Error', "System Cannot Find the File Specified")
def connect():
    #this function connects to Arduino and sets clickable buttons
    global ser
    try:
        ser = serial.Serial(arduino_ports[0])
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        lport.config(fg="green", font=("Arial", 12, "bold"))
        b3.config(state=tkinter.DISABLED)
        b10.config(state=tkinter.NORMAL)
        db.config(state=tkinter.NORMAL)
        b4.config(state=tkinter.NORMAL)
```

```
b6.config(state=tkinter.NORMAL)
b7.config(state=tkinter.NORMAL)
bl8.config(state=tkinter.NORMAL)
bl9.config(state=tkinter.NORMAL)
pmtb.config(state=tkinter.NORMAL)
pmtb2.config(state=tkinter.NORMAL)
dwellcombo.config(state=tkinter.NORMAL)
mb1.config(state=tkinter.NORMAL)
mb2.config(state=tkinter.NORMAL)
mb3.config(state=tkinter.NORMAL)
mb4.config(state=tkinter.NORMAL)
mb5.config(state=tkinter.NORMAL)
mb6.config(state=tkinter.NORMAL)
mb7.config(state=tkinter.NORMAL)
mb8.config(state=tkinter.NORMAL)
mb9.config(state=tkinter.NORMAL)
mb10.config(state=tkinter.NORMAL)
mb11.config(state=tkinter.NORMAL)
IRpower.config(text="IR Power", fg="black", font=("Arial", 12, "bold"))
BLpower.config(text="Blue Power", fg="black", font=("Arial", 12, "bold"))
time.sleep(0.5)
if ser.readline() != 0:
```

```
            connectioncanvas.itemconfig(connectindicator, fill="green", outline="green")
            connectioncanvas.itemconfig(disconnectindicator, fill="red", outline="red")
            connectioncanvas.itemconfig(resetindicator, fill="red", outline="red")
    except serial.SerialException:
        messagebox.showerror("Error", "Cannot connect, please restart")
def disconnect():
    #this function first resets, then disconnects the Arduino
    #and sets buttons as not clickable
    global ser
    try:
        if ser is not None and ser.is_open:
            resetting()
            ser.reset_input_buffer()
            ser.reset_output_buffer()
            ser.close()
            lport.config(fg="red", font=("Arial", 12, "bold"))
            b3.config(state=tkinter.NORMAL)
            b4.config(state=tkinter.DISABLED)
            b6.config(state=tkinter.DISABLED)
            b7.config(state=tkinter.DISABLED)
            plb.config(state=tkinter.DISABLED)
```

```
b8.config(state=tkinter.DISABLED)

bl8.config(state=tkinter.DISABLED)

bl9.config(state=tkinter.DISABLED)

delayb.config(state=tkinter.DISABLED)

b9.config(state=tkinter.DISABLED)

b10.config(state=tkinter.DISABLED)

db.config(state=tkinter.DISABLED)

dwellcombo.config(state=tkinter.DISABLED)

tb.config(state=tkinter.DISABLED)

lightcombo.config(state=tkinter.DISABLED)

b11.config(state=tkinter.DISABLED)

pmtb.config(state=tkinter.DISABLED)

pmtb2.config(state=tkinter.DISABLED)

mb1.config(state=tkinter.DISABLED)

mb2.config(state=tkinter.DISABLED)

mb3.config(state=tkinter.DISABLED)

mb4.config(state=tkinter.DISABLED)

mb5.config(state=tkinter.DISABLED)

mb6.config(state=tkinter.DISABLED)

mb7.config(state=tkinter.DISABLED)

mb8.config(state=tkinter.DISABLED)

mb9.config(state=tkinter.DISABLED)
```

```
        mb10.config(state=tkinter.DISABLED)
        mb11.config(state=tkinter.DISABLED)
        poslabel2.config(text="")
        IRpower.config(text="IR Power", fg="black", font=("Arial", 12, "bold"))
        BLpower.config(text="Blue Power", fg="black", font=("Arial", 12, "bold"))
        pmtpower.config(text="PMT Power", fg="black", font=("Arial", 12,
"bold"))
        connectioncanvas.itemconfig(connectindicator, fill="black",
outline="black")
        connectioncanvas.itemconfig(disconnectindicator, fill="green",
outline="green")
        connectioncanvas.itemconfig(resetindicator, fill="black", outline="black")
        ledcanvas.itemconfig(irindicator, fill="black", outline="black")
        ledcanvas.itemconfig(pmtindicator, fill="black", outline="black")
        ledcanvas.itemconfig(blueindicator, fill="black", outline="black")
        selectioncanvas.itemconfig(dwellindicator, fill="black", outline="black")
        selectioncanvas.itemconfig(stimulationindicator, fill="black",
outline="black")
        datacanvas.itemconfig(armindicator, fill="black", outline="black")
        datacanvas.itemconfig(startindicator, fill="black", outline="black")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
```

```
        messagebox.showerror("Error", "Connect to an Arduino")
def resetting():
    # resets the power status of LED's and PMT, stops experiment, resets arduino
    try:
        IRpower.config(text="IR Power", fg="black", font=("Arial", 12, "bold"))
        BLpower.config(text="Blue Power", fg="black", font=("Arial", 12, "bold"))
        pmtpower.config(text="PMT", fg="black", font=("Arial", 12, "bold"))
        ledcanvas.itemconfig(irindicator, fill="black", outline="black")
        ledcanvas.itemconfig(pmtindicator, fill="black", outline="black")
        ledcanvas.itemconfig(blueindicator, fill="black", outline="black")
        selectioncanvas.itemconfig(dwellindicator, fill="black", outline="black")
        selectioncanvas.itemconfig(stimulationindicator, fill="black",
outline="black")
        datacanvas.itemconfig(armindicator, fill="black", outline="black")
        datacanvas.itemconfig(startindicator, fill="black", outline="black")
        poslabel2.config(text="")
        b7.config(state=tkinter.NORMAL)
        b6.config(state=tkinter.NORMAL)
        bl8.config(state=tkinter.NORMAL)
        bl9.config(state=tkinter.NORMAL)
        pmtb.config(state=tkinter.NORMAL)
        pmtb2.config(state=tkinter.NORMAL)
```

```
    mb1.config(state=tkinter.NORMAL)
    mb2.config(state=tkinter.NORMAL)
    mb3.config(state=tkinter.NORMAL)
    mb4.config(state=tkinter.NORMAL)
    mb5.config(state=tkinter.NORMAL)
    mb6.config(state=tkinter.NORMAL)
    mb7.config(state=tkinter.NORMAL)
    mb8.config(state=tkinter.NORMAL)
    mb9.config(state=tkinter.NORMAL)
    mb10.config(state=tkinter.NORMAL)
    mb11.config(state=tkinter.NORMAL)
    ser.reset_input_buffer()
    ser.reset_output_buffer()
    ser.close()
    time.sleep(1)
    ser.open()
    started = ser.readline()
    if started != 0:
        connectioncanvas.itemconfig(connectindicator, fill="green",
outline="green")
        connectioncanvas.itemconfig(disconnectindicator, fill="red", outline="red")
        connectioncanvas.itemconfig(resetindicator, fill="red", outline="red")
```

```
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def poweron():
    #function to power the IR LED on. Enables IR pulses
    try:
        selection = 'IRPOWER\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        IRpower.config(text="IR Power On", fg="green", font=("Arial", 12, "bold"))
        b6.config(state=tkinter.DISABLED)
        b7.config(state=tkinter.NORMAL)
        if ser.readline() != 0:
            ledcanvas.itemconfig(irindicator, fill="green", outline="green")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def poweroff():
    # function to power the IR LED off. Disables IR pulses
```

```
    try:
        selection = 'IRPOWERO\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        IRpower.config(text="IR Power Off", fg="red", font=("Arial", 12, "bold"))
        b7.config(state=tkinter.DISABLED)
        b6.config(state=tkinter.NORMAL)
        if ser.readline() != 0:
            ledcanvas.itemconfig(irindicator, fill="red", outline="red")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def pmtenable():
    # function to enable the gate of the PMT
    try:
        selection = 'PMTEN\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        time.sleep(2)
```

```
        pmtpower.config(text="PMT Enabled", fg="green", font=("Arial", 12,
"bold"))
        if ser.readline() != 0:
            ledcanvas.itemconfig(pmtindicator, fill="green", outline="green")
        pmtb.config(state=tkinter.DISABLED)
        pmtb2.config(state=tkinter.NORMAL)
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def pmtdisable():
    #function to disable the gate of the PMT
    try:
        selection = 'PMTDIS\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        pmtpower.config(text="PMT Disabled", fg="red", font=("Arial", 12, "bold"))
        if ser.readline() != 0:
            ledcanvas.itemconfig(pmtindicator, fill="red", outline="red")
        pmtb2.config(state=tkinter.DISABLED)
        pmtb.config(state=tkinter.NORMAL)
```

```
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def bluepoweron():
    # function to power the Blue LED on. Enables Blue pulses
    try:
        selection = 'BLPOWER\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        BLpower.config(text="BL Power On", fg="green", font=("Arial", 12, "bold"))
        if ser.readline() != 0:
            ledcanvas.itemconfig(blueindicator, fill="green", outline="green")
        bl8.config(state=tkinter.DISABLED)
        bl9.config(state=tkinter.NORMAL)
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def bluepoweroff():
    # function to power the Blue LED off. Disables Blue pulses
```

```
    try:
        selection = 'BLPOWERO\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        BLpower.config(text="BL Power Off", fg="red", font=("Arial", 12, "bold"))
        if ser.readline() != 0:
            ledcanvas.itemconfig(blueindicator, fill="red", outline="red")
        bl9.config(state=tkinter.DISABLED)
        bl8.config(state=tkinter.NORMAL)
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def latch():
    #takes set variables and sends  them to the Arduino to arm 74f525 for counting
    try:
        global passlength
        global delaytime
        passlength = int(plb.get())
        delaytime = float(delayb.get())
        selection3 = dwellcombo.get()
```

```
    if illuminate != 0:
        selection = 'L ' + '\n\r'
        if selection3 == "ns":
            exptime = (((dwelltime * 1000000 / 250) + delaytime * 1000000) *
passlength)
            exptime = int(exptime)
        if selection3 == "µs":
            exptime = (((dwelltime * 1000) + delaytime * 1000000) * passlength)
            exptime = int(exptime)
        nstous = exptime / 1000
        ustoms = nstous / 1000
        mstos = ustoms / 1000
        stomin = mstos / 60
        mintohour = stomin / 60
        stomin = float(stomin)
        mintohour = float(mintohour)
        if 1 < exptime < 1000:
            expnumber = str(exptime) + "ms"
            lexperimentnumber.config(text=expnumber)
        if 1 < mstos < 119:
            mstos = format(mstos, '.2f')
            expnumber = str(mstos) + "s"
```

```
            lexperimentnumber.config(text=expnumber)
        if 1 < stomin < 119:
            stomin = format(stomin, '.2f')
            expnumber = str(stomin) + "min"
            lexperimentnumber.config(text=expnumber)
        if 1 < mintohour:
            mintohour = format(mintohour, '.2f')
            expnumber = str(mintohour) + " hour"
            lexperimentnumber.config(text=expnumber)
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        if ser.readline() != 0:
            datacanvas.itemconfig(armindicator, fill="green", outline="green")
    else:
        messagebox.showerror("Error", "No number to latch")
except ValueError:
    messagebox.showerror("Error", "Enter both variables")
except NameError:
    messagebox.showerror("Error", "Connect to an Arduino")
except serial.SerialException:
    messagebox.showerror("Error", "Connect to an Arduino")
```

```
def start():
    # sends a command to initiate the start experiment function of Arduino
    # also sets the delay time between acquisitions automatically
    passlength = int(plb.get())
    delaytime = int(delayb.get())
    try:
        if delaytime > 0.99:
            selection6 = 'START ' + str(passlength) + " " + str(delaytime) + '\n\r'
            ser.reset_input_buffer()
            ser.reset_output_buffer()
            ser.write(selection6.encode('utf-8'))
            if ser.readline() != 0:
                datacanvas.itemconfig(startindicator, fill="green", outline="green")
        else:
            messagebox.showerror("Error", "Minimum Delay is 1ms")
    except ValueError:
        messagebox.showerror("Error", "Please enter an integer number")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
```

```
def frequencyselection():
    # sets the dwelltime as the integer multiple of LED stimulation pulses
    # sends this information to Arduino
    try:
        tb.config(state=tkinter.NORMAL)
        lightcombo.config(state=tkinter.NORMAL)
        b11.config(state=tkinter.NORMAL)
        global dwelltime
        dwelltime = float(db.get())
        selection3 = dwellcombo.get()
        if selection3 == "ns":
            if 124 < dwelltime < 125001:
                value = 0.000000001
                frequencydwell = 1 / (dwelltime * value)
                frequencydwell = round(frequencydwell, 6)
                frequencydwell = int(frequencydwell)
                seconds = frequencydwell
                selection4 = 'F ' + str(seconds) + '\n\r'
                ser.reset_input_buffer()
                ser.reset_output_buffer()
                ser.write(selection4.encode('utf-8'))
                if ser.readline() != 0:
```

```
                selectioncanvas.itemconfig(dwellindicator, fill="green",
outline="green")
            else:
                messagebox.showerror("Error", "Dwell time should be between 125ns
and 125µs")
        elif selection3 == "µs":
            if 0.249 < dwelltime < 126:
                value = 0.000001
                frequencydwell = 1 / (dwelltime * value)
                frequencydwell = round(frequencydwell, 6)
                frequencydwell = int(frequencydwell)
                seconds = frequencydwell
                selection4 = 'F ' + str(seconds) + '\n\r'
                ser.reset_input_buffer()
                ser.reset_output_buffer()
                ser.write(selection4.encode('utf-8'))
                if ser.readline() != 0:
                    selectioncanvas.itemconfig(dwellindicator, fill="green",
outline="green")
            else:
                messagebox.showerror("Error", "Dwell time should be between 250ns
and 125µs")
    except ValueError:
```

```
        messagebox.showerror("Error", "Please enter a number")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
def timeselection():
    # calculates the amount of pulses 74f525 will count down from
    # using the dwell time and selection boxes
    try:
        plb.config(state=tkinter.NORMAL)
        b8.config(state=tkinter.NORMAL)
        b9.config(state=tkinter.NORMAL)
        delayb.config(state=tkinter.NORMAL)
        global illuminate
        global multiplier
        global selected
        global bins
        selection2 = lightcombo.get()
        time = float(tb.get())
        binmultiplier = float(db.get())
        if selection2 == "ns":
            if time < 1000:
                multiplier = 1 / 4000000
                time = time * 0.000000001
```

```
            multiplier = float(multiplier)
            illuminate = time / multiplier
            illuminate = round(illuminate)
            selected = str(illuminate)
            binmultiplier = binmultiplier / 250
            bins = illuminate * 10 / binmultiplier
            bins = math.ceil(bins)
            l9.config(text=bins)
            l6.config(text=selected)
            selection3 = 'P ' + str(selected) + '\n\r'
            ser.reset_input_buffer()
            ser.reset_output_buffer()
            ser.write(selection3.encode('utf-8'))
            if ser.readline() != 0:
                selectioncanvas.itemconfig(stimulationindicator, fill="green",
outline="green")
        else:
            messagebox.showwarning("Warning", "Please choose µs")
    elif selection2 == "µs":
        if time < 1000:
            multiplier = 1 / 4000000
            time = time * 0.000001
```

```
            multiplier = float(multiplier)

            illuminate = time / multiplier

            illuminate = round(illuminate)

            selected = str(illuminate)

            bins = illuminate * 10 / binmultiplier

            bins = math.ceil(bins)

            l9.config(text=bins)

            l6.config(text=selected)

            selection3 = 'P ' + str(selected) + '\n\r'

            ser.reset_input_buffer()

            ser.reset_output_buffer()

            ser.write(selection3.encode('utf-8'))

            if ser.readline() != 0:

                selectioncanvas.itemconfig(stimulationindicator, fill="green",
outline="green")

        else:

            messagebox.showwarning("Warning", "Please choose ms")

    elif selection2 == "ms":

        if time < 8.193:

            multiplier = 1 / 4000000

            time = time * 0.001

            multiplier = float(multiplier)
```

```
            illuminate = time / multiplier

            illuminate = round(illuminate)

            selected = str(illuminate)

            bins = illuminate * 10 / binmultiplier

            bins = math.ceil(bins)

            l9.config(text=bins)

            l6.config(text=selected)

            selection3 = 'P ' + str(selected) + '\n\r'

            ser.reset_input_buffer()

            ser.reset_output_buffer()

            ser.write(selection3.encode('utf-8'))

            if ser.readline() != 0:

                selectioncanvas.itemconfig(stimulationindicator, fill="green",
outline="green")

        else:

            messagebox.showwarning("Warning", "Cannot provide more than 32768
pulses")

    except serial.SerialException:

        messagebox.showerror("Error", "Connect to an Arduino")

    except NameError:

        messagebox.showerror("Error", "Please enter a number in Frequency or
Dwell")

    except ValueError:
```

```
        messagebox.showerror("Error", "Please enter a number")
def devoptions():
    #developer options, allows for sending raw numbers of pulses and frequency
    #allows for resetting the entry boxes
    def burstgenerator():
        global ser
        try:
            pulsecount = int(ib.get())
            if pulsecount == int(ib.get()):
                selection = 'P ' + str(pulsecount) + '\n\r'
                ser.reset_input_buffer()
                ser.reset_output_buffer()
                ser.write(selection.encode('utf-8'))
        except ValueError:
            messagebox.showerror("Error", "Please enter an integer from 1 to 65,535")
        except (serial.SerialException, NameError):
            messagebox.showerror("Error", "Connect to an Arduino")
    def frequencygenerator():
        global ser
        try:
            frequency = int(fb.get())
            if frequency == int(fb.get()):
```

```
            selection = 'F ' + str(frequency) + '\n\r'
            ser.reset_input_buffer()
            ser.reset_output_buffer()
            ser.write(selection.encode('utf-8'))
        except ValueError:
            messagebox.showerror("Error", "Please enter an integer from 8000 to 
4000000")
        except (serial.SerialException, NameError):
            messagebox.showerror("Error", "Connect to an Arduino")
    def resetstate():
        db.config(state=tkinter.NORMAL)
        db.delete(0, END)
        tb.delete(0, END)
        fb.delete(0, END)
        ib.delete(0, END)
        plb.delete(0, END)
    #tkinter's main dropdown menu selections and developer options window
    devwindow = Toplevel()
    devwindow.iconbitmap("myIcon.ico")
    devwindow.title("Developer Options")
    devwindow.geometry()
    ib = Entry(devwindow, justify=LEFT)
```

```
    fb = Entry(devwindow, justify=LEFT)
    b1 = Button(devwindow, text='Send Burst Number', command=burstgenerator,
width=17)
    b9 = Button(devwindow, text='Send Frequency', command=frequencygenerator,
width=17)
    b17 = Button(devwindow, text="Exit Dev Options",
command=devwindow.destroy, width=17)
    l6 = Label(devwindow, text="Frequency")
    l7 = Label(devwindow, text="Pulses")
    b13 = Button(devwindow, text="Reset Entry States", command=resetstate,
width=17)
    l6.grid(row=0, column=2)
    fb.grid(row=1, column=2)
    b9.grid(row=2, column=2)
    b13.grid(row=3, column=2)
    l7.grid(row=0, column=3)
    ib.grid(row=1, column=3)
    b1.grid(row=2, column=3)
    b17.grid(row=3, column=3)
    Label(devwindow, text="Tips", font=("Arial", 14, "bold")).grid(row=0,
column=0, padx=10, pady=(10, 5), sticky="w")
    guidelines = [
        "Send frequency directly sends it to Arduino without limitations",
```

```
        "Send Burst Number directly sends the number of light pulses",
        'Reset Entry States resets all the boxes in the interface',    ]
    for i, guideline in enumerate(guidelines, start=1):
        Label(devwindow, text=f"{i}. {guideline}", font=("Arial", 9),
justify="left").grid(row=i, column=0, padx=10, sticky="w")
def mcs32location():
    #file path function for the file dropdown menu
    global file_path
    file_path = fd.askopenfilename()
def aboutmenu():
    #an about part of the software, accessible from help->about
    aboutwindow = Toplevel()
    aboutwindow.iconbitmap("myIcon.ico")
    aboutwindow.title("Developer Options")
    aboutwindow.geometry()
    authorlabel = Label(aboutwindow, text="Author")
    namelabel = Label(aboutwindow, text="Deniz Boztemur")
    datelabel = Label(aboutwindow, text="2024")
    maillabel = Label(aboutwindow, text="dboz@metu.edu.tr")
    infolabel = Label(aboutwindow, text="Thesis work in TOSL Lab")
    infolabel2 = Label(aboutwindow, text="Supervised by Enver Bulur")
    authorlabel.grid(row=0, column=0)
```

```
    namelabel.grid(row=1, column=0)
    datelabel.grid(row=2, column=0)
    maillabel.grid(row=3, column=0)
    infolabel.grid(row=4, column=0)
    infolabel2.grid(row=5, column=0)
def rotaryhome():
    #homes the rotary stage
    #sets all positions available
    try:
        mb1.config(state=tkinter.NORMAL)
        mb2.config(state=tkinter.NORMAL)
        mb3.config(state=tkinter.NORMAL)
        mb4.config(state=tkinter.NORMAL)
        mb5.config(state=tkinter.NORMAL)
        mb6.config(state=tkinter.NORMAL)
        mb7.config(state=tkinter.NORMAL)
        mb8.config(state=tkinter.NORMAL)
        mb9.config(state=tkinter.DISABLED)
        selection = 'HOME\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
```

```
        pos = "Rotary Home"
        poslabel2.config(text=pos)
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def sampleone():
    #sets the rotary stage to sample one
    #sample one button is disabled
    try:
        poslabel2.config(fg="black")
        mb1.config(state=tkinter.DISABLED)
        mb2.config(state=tkinter.NORMAL)
        mb3.config(state=tkinter.NORMAL)
        mb4.config(state=tkinter.NORMAL)
        mb5.config(state=tkinter.NORMAL)
        mb6.config(state=tkinter.NORMAL)
        mb7.config(state=tkinter.NORMAL)
        mb8.config(state=tkinter.NORMAL)
        mb9.config(state=tkinter.NORMAL)
        selection = 'ONE\n\r'
        ser.reset_input_buffer()
```

```
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        pos = "Sample One"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def sampletwo():
    # sets the rotary stage to sample two
    # sample two button is disabled
    try:
        poslabel2.config(fg="black")
        mb2.config(state=tkinter.DISABLED)
        mb1.config(state=tkinter.NORMAL)
        mb3.config(state=tkinter.NORMAL)
        mb4.config(state=tkinter.NORMAL)
        mb5.config(state=tkinter.NORMAL)
        mb6.config(state=tkinter.NORMAL)
        mb7.config(state=tkinter.NORMAL)
```

```
        mb8.config(state=tkinter.NORMAL)
        mb9.config(state=tkinter.NORMAL)
        selection = 'TWO\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        pos = "Sample Two"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def samplethree():
    # sets the rotary stage to sample three
    # sample three button is disabled
    try:
        poslabel2.config(fg="black")
        mb3.config(state=tkinter.DISABLED)
        mb1.config(state=tkinter.NORMAL)
        mb2.config(state=tkinter.NORMAL)
```

```
        mb4.config(state=tkinter.NORMAL)
        mb5.config(state=tkinter.NORMAL)
        mb6.config(state=tkinter.NORMAL)
        mb7.config(state=tkinter.NORMAL)
        mb8.config(state=tkinter.NORMAL)
        mb9.config(state=tkinter.NORMAL)
        selection = 'THREE\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        pos = "Sample Three"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def samplefour():
    # sets the rotary stage to sample four
    # sample four button is disabled
    try:
```

```
    poslabel2.config(fg="black")
    mb4.config(state=tkinter.DISABLED)
    mb1.config(state=tkinter.NORMAL)
    mb2.config(state=tkinter.NORMAL)
    mb3.config(state=tkinter.NORMAL)
    mb5.config(state=tkinter.NORMAL)
    mb6.config(state=tkinter.NORMAL)
    mb7.config(state=tkinter.NORMAL)
    mb8.config(state=tkinter.NORMAL)
    mb9.config(state=tkinter.NORMAL)
    selection = 'FOUR\n\r'
    ser.reset_input_buffer()
    ser.reset_output_buffer()
    ser.write(selection.encode('utf-8'))
    pos = "Sample Four"
    poslabel2.config(text=pos)
    if ser.readline() != 0:
        poslabel2.config(fg="green")
except NameError:
    messagebox.showerror("Error", "Connect to an Arduino")
except serial.SerialException:
    messagebox.showerror("Error", "Connect to an Arduino")
```

```
def samplefive():
    # sets the rotary stage to sample five
    # sample five button is disabled
    try:
        poslabel2.config(fg="black")
        mb5.config(state=tkinter.DISABLED)
        mb1.config(state=tkinter.NORMAL)
        mb2.config(state=tkinter.NORMAL)
        mb3.config(state=tkinter.NORMAL)
        mb4.config(state=tkinter.NORMAL)
        mb6.config(state=tkinter.NORMAL)
        mb7.config(state=tkinter.NORMAL)
        mb8.config(state=tkinter.NORMAL)
        mb9.config(state=tkinter.NORMAL)
        selection = 'FIVE\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        pos = "Sample Five"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
```

```
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def samplesix():
    # sets the rotary stage to sample six
    # sample six button is disabled
    try:
        poslabel2.config(fg="black")
        mb6.config(state=tkinter.DISABLED)
        mb1.config(state=tkinter.NORMAL)
        mb2.config(state=tkinter.NORMAL)
        mb3.config(state=tkinter.NORMAL)
        mb4.config(state=tkinter.NORMAL)
        mb5.config(state=tkinter.NORMAL)
        mb7.config(state=tkinter.NORMAL)
        mb8.config(state=tkinter.NORMAL)
        mb9.config(state=tkinter.NORMAL)
        selection = 'SIX\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
```

```
        pos = "Sample Six"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def sampleseven():
    # sets the rotary stage to sample seven
    # sample seven button is disabled
    try:
        poslabel2.config(fg="black")
        mb7.config(state=tkinter.DISABLED)
        mb1.config(state=tkinter.NORMAL)
        mb2.config(state=tkinter.NORMAL)
        mb3.config(state=tkinter.NORMAL)
        mb4.config(state=tkinter.NORMAL)
        mb5.config(state=tkinter.NORMAL)
        mb6.config(state=tkinter.NORMAL)
        mb8.config(state=tkinter.NORMAL)
        mb9.config(state=tkinter.NORMAL)
```

```
        selection = 'SEVEN\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        pos = "Sample Seven"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def sampleeight():
    # sets the rotary stage to sample eight
    # sample eight button is disabled
    try:
        poslabel2.config(fg="black")
        mb8.config(state=tkinter.DISABLED)
        mb1.config(state=tkinter.NORMAL)
        mb2.config(state=tkinter.NORMAL)
        mb3.config(state=tkinter.NORMAL)
        mb4.config(state=tkinter.NORMAL)
```

```
        mb5.config(state=tkinter.NORMAL)
        mb6.config(state=tkinter.NORMAL)
        mb7.config(state=tkinter.NORMAL)
        mb9.config(state=tkinter.NORMAL)
        selection = 'EIGHT\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        pos = "Sample Eight"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def linearhome():
    # sets the linear stage to its home position
    # disables the home button
    try:
        poslabel2.config(fg="black")
        selection = 'LHOME\n\r'
```

```
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
        pos = "Linear Home"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
        mb11.config(state=tkinter.DISABLED)
        mb10.config(state=tkinter.NORMAL)
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
def lineareject():
    #ejects the linear tray for sample loading
    #disables the eject button
    try:
        poslabel2.config(fg="black")
        selection = 'EJECT\n\r'
        ser.reset_input_buffer()
        ser.reset_output_buffer()
        ser.write(selection.encode('utf-8'))
```

```
        pos = "Tray Ejected"
        poslabel2.config(text=pos)
        if ser.readline() != 0:
            poslabel2.config(fg="green")
        mb10.config(state=tkinter.DISABLED)
        mb11.config(state=tkinter.NORMAL)
    except NameError:
        messagebox.showerror("Error", "Connect to an Arduino")
    except serial.SerialException:
        messagebox.showerror("Error", "Connect to an Arduino")
#from now on, tkinter
#creates menu bar, menus, several frames, creates buttons, entryboxes and drop-down menus
#then places them accordingly into the frames
#creates two tabs, places frames into one and guidelines into the other
root = Tk()
root.iconbitmap("myIcon.ico")
root.geometry("1024x600")
root.title("TR-OSL Interface v1.0.2")
tabControl = ttk.Notebook(root)
tab1 = ttk.Frame(tabControl)
tab2 = ttk.Frame(tabControl)
```

```
tabControl.add(tab1, text='Guidelines')

tabControl.add(tab2, text='Control')

tabControl.grid(row=0, column=0)


defaultbg = root.cget('bg')

menubar = Menu(root)

filemenu = Menu(menubar, tearoff=0)

helpmenu = Menu(menubar, tearoff=0)

filemenu.add_command(label="MCS-32 Location", command=mcs32location)

filemenu.add_separator()

filemenu.add_command(label="Developer Options", command=devoptions)

filemenu.add_separator()

filemenu.add_command(label="Exit", command=root.quit)

helpmenu.add_command(label="About", command=aboutmenu)

menubar.add_cascade(label="File", menu=filemenu)

menubar.add_cascade(label="Help", menu=helpmenu)

infoframe = Frame(tab1, borderwidth=5, relief="sunken")

softwareframe = LabelFrame(tab2, text="MCS-32 Software", font=("Arial", 9,
"bold"))

connection_frame = LabelFrame(tab2, text="Arduino Connection", font=("Arial",
9, "bold"))

ledframe = LabelFrame(tab2, text="Light Source Driver", font=("Arial", 9,
"bold"))
```

```
dataframe = LabelFrame(tab2, text="Experiment Start", font=("Arial", 9, "bold"))

selectionframe = LabelFrame(tab2, text="Parameter Selection", font=("Arial", 9, "bold"))

movementframe = LabelFrame(tab2, text="Movement Control", font=("Arial", 9, "bold"))

infoframe.grid(row=1, column=0, rowspan=10, padx=5, pady=5, sticky="nsew")

softwareframe.grid(row=1, column=1, padx=5, pady=5, sticky="nsew")

connection_frame.grid(row=2, column=1, padx=5, pady=5, sticky="nsew")

ledframe.grid(row=1, column=2, padx=5, pady=5, sticky="nsew")

dataframe.grid(row=2, column=3, padx=5, pady=5, sticky="nsew")

selectionframe.grid(row=1, column=3, padx=5, pady=5, sticky="nsew")

movementframe.grid(row=2, column=2, padx=5, pady=5, sticky="nsew")

mb1 = Button(movementframe, text="One", command=sampleone, height=2, width=5, font="Arial")

mb2 = Button(movementframe, text="Two", command=sampletwo, height=2, width=5, font="Arial")

mb3 = Button(movementframe, text="Three", command=samplethree, height=2, width=5, font="Arial")

mb4 = Button(movementframe, text="Four", command=samplefour, height=2, width=5, font="Arial")

mb5 = Button(movementframe, text="Five", command=samplefive, height=2, width=5, font="Arial")

mb6 = Button(movementframe, text="Six", command=samplesix, height=2, width=5, font="Arial")
```

```
mb7 = Button(movementframe, text="Seven", command=sampleseven, height=2, width=5, font="Arial")

mb8 = Button(movementframe, text="Eight", command=sampleeight, height=2, width=5, font="Arial")

mb9 = Button(movementframe, text="Home", command=rotaryhome, height=2, width=5, font="Arial")

mb10 = Button(movementframe, text="Eject Tray", command=lineareject, height=2, width=10, font="Arial")

mb11 = Button(movementframe, text="Load Tray", command=linearhome, height=2, width=10, font="Arial")

poslabel = Label(movementframe, text="Tray Position", font=("Arial", 12, "bold"))

poslabel2 = Label(movementframe, text=pos, font=("Arial", 12, "bold"))

mb1.grid(row=0, column=0)

mb2.grid(row=0, column=1)

mb3.grid(row=0, column=2)

mb4.grid(row=1, column=0)

mb5.grid(row=1, column=1)

mb6.grid(row=1, column=2)

mb7.grid(row=2, column=0)

mb8.grid(row=2, column=1)

mb9.grid(row=2, column=2)

mb10.grid(row=0, column=3)

mb11.grid(row=1, column=3)
```

```
poslabel.grid(row=2, column=3)

poslabel2.grid(row=3, column=3)

mb1.config(state=tkinter.DISABLED)

mb2.config(state=tkinter.DISABLED)

mb3.config(state=tkinter.DISABLED)

mb4.config(state=tkinter.DISABLED)

mb5.config(state=tkinter.DISABLED)

mb6.config(state=tkinter.DISABLED)

mb7.config(state=tkinter.DISABLED)

mb8.config(state=tkinter.DISABLED)

mb9.config(state=tkinter.DISABLED)

mb10.config(state=tkinter.DISABLED)

mb11.config(state=tkinter.DISABLED)

b1 = Button(softwareframe, text='Run MCS-32', command=run, height=2,
width=11, font="Arial")

b1.grid(row=0, column=0, sticky="NEWS")

softwarecanvas = Canvas(softwareframe, height=40, width=10)

softwarecanvas.grid(row=1, column=0, sticky="NEWS")

softwareindicator = softwarecanvas.create_oval(60, 10, 40, 30, fill="black",
outline="black")

lport = Label(connection_frame, text=arduino_ports[0], font=("Arial", 12, "bold"))

b3 = Button(connection_frame, text="Connect", command=connect, height=2,
width=11, font="Arial")
```

```
b4 = Button(connection_frame, text="Disconnect", command=disconnect, height=2, width=11, font="Arial")

b5 = Button(connection_frame, text="Reset Arduino", command=resetting, height=2, width=11, font="Arial")

lport.grid(row=0, column=0)

b3.grid(row=1, column=0)

b4.grid(row=2, column=0)

b5.grid(row=3, column=0)

connectioncanvas = Canvas(connection_frame, height=100, width=40)

connectioncanvas.grid(row=1, column=1, rowspan=3, sticky="news")

connectindicator = connectioncanvas.create_oval(10, 15, 30, 35, fill="black", outline="black")

disconnectindicator = connectioncanvas.create_oval(10, 65, 30, 85, fill="black", outline="black")

resetindicator = connectioncanvas.create_oval(10, 115, 30, 135, fill="black", outline="black")


IRpower = Label(ledframe, text="IR Power", font=("Arial", 12, "bold"))

BLpower = Label(ledframe, text="Blue Power", font=("Arial", 12, "bold"))

b6 = Button(ledframe, text="Power On", command=poweron, height=2, width=10, font="Arial")

b7 = Button(ledframe, text="Power Off", command=poweroff, height=2, width=10, font="Arial")
```

```
bl8 = Button(ledframe, text="Power On", command=bluepoweron, height=2, width=10, font="Arial")
bl9 = Button(ledframe, text="Power Off", command=bluepoweroff, height=2, width=10, font="Arial")
pmtb = Button(ledframe, text="PMT Enable", command=pmtenable, height=2, width=10, font="Arial")
pmtb2 = Button(ledframe, text="PMT Disable", command=pmtdisable, height=2, width=10, font="Arial")
pmtpower = Label(ledframe, text="PMT", font=("Arial", 12, "bold"))
IRpower.grid(row=0, column=0)
BLpower.grid(row=0, column=1)
b6.grid(row=1, column=0)
b7.grid(row=2, column=0)
bl8.grid(row=1, column=1)
bl9.grid(row=2, column=1)
pmtpower.grid(row=0, column=2)
pmtb.grid(row=1, column=2)
pmtb2.grid(row=2, column=2)
pmtb.config(state=tkinter.DISABLED)
pmtb2.config(state=tkinter.DISABLED)
b6.config(state=tkinter.DISABLED)
b7.config(state=tkinter.DISABLED)
bl8.config(state=tkinter.DISABLED)
```

```
bl9.config(state=tkinter.DISABLED)

ledcanvas = Canvas(ledframe, height=40, width=100)

ledcanvas.grid(row=3, column=0, columnspan=3, sticky="news")

irindicator = ledcanvas.create_oval(40, 10, 60, 30, fill="black", outline="black")

blueindicator = ledcanvas.create_oval(140, 10, 160, 30, fill="black",
outline="black")

pmtindicator = ledcanvas.create_oval(240, 10, 260, 30, fill="black",
outline="black")

l3 = Label(dataframe, text="Acquisition Count", font=("Arial", 12, "bold"))

ldelay = Label(dataframe, text="Single Pass Time (ms)", font=("Arial", 12,
"bold"))

b8 = Button(dataframe, text="Arm Experiment", command=latch, height=1,
width=15, font=("Arial", 12, "bold"))

plb = Entry(dataframe, justify=LEFT, width=23)

delayb = Entry(dataframe, justify=LEFT, width=23)

b9 = Button(dataframe, text="Start Experiment", command=start, height=1,
width=15, font=("Arial", 12, "bold"))

image = PhotoImage(file="tosl.png")

label = Label(dataframe, image=image)

l3.grid(row=0, column=0)

plb.grid(row=1, column=0)

ldelay.grid(row=2, column=0)

delayb.grid(row=3, column=0)
```

```
b8.grid(row=4, column=0)
b9.grid(row=5, column=0)
label.grid(row=0, column=1, rowspan=2)
delayb.config(state=tkinter.DISABLED)
plb.config(state=tkinter.DISABLED)
b8.config(state=tkinter.DISABLED)
b9.config(state=tkinter.DISABLED)
datacanvas = Canvas(dataframe, height=10, width=40)
datacanvas.grid(row=4, column=1, rowspan=6, sticky="news")
armindicator = datacanvas.create_oval(5, 7, 25, 27, fill="black", outline="black")
startindicator = datacanvas.create_oval(5, 38, 25, 58, fill="black", outline="black")
l3 = Label(selectionframe, text="Dwell Time", font=("Arial", 12, "bold"))
db = Entry(selectionframe, justify=LEFT, width=33)
dwellcombo = ttk.Combobox(selectionframe, state="readonly", values=["ns",
"µs"], font=("Arial", 12, "bold"))
dwellcombo.current(0)
b10 = Button(selectionframe, text="Confirm Dwell Time",
command=frequencyselection, height=1, width=22, font=("Arial", 12, "bold"))
l4 = Label(selectionframe, text="Stimulation Duration", font=("Arial", 12, "bold"))
tb = Entry(selectionframe, justify=LEFT, width=33)
lightcombo = ttk.Combobox(selectionframe, state="readonly", values=["ns", "µs",
"ms"], font=("Arial", 12, "bold"))
lightcombo.current(0)
```

```
b11 = Button(selectionframe, text="Confirm Stimulation Duration",
command=timeselection, height=1, width=22, font=("Arial", 12, "bold"))

l8 = Label(selectionframe, text="Suggested \n Data Points", font=("Arial", 12,
"bold"))

l9 = Label(selectionframe, text=binNumber, font=("Arial", 12, "bold"))

l5 = Label(selectionframe, text="Light Pulse Count", font=("Arial", 12, "bold"))

l6 = Label(selectionframe, text=illuminate, font=("Arial", 12, "bold"))

lexperiment = Label(selectionframe, text="Total Approximate \n Experiment
Time", font=("Arial", 12, "bold"))

lexperimentnumber = Label(selectionframe, text=totaltime, font=("Arial", 12,
"bold"))

l3.grid(row=0, column=0)

db.grid(row=1, column=0)

dwellcombo.grid(row=2, column=0)

b10.grid(row=3, column=0)

l4.grid(row=4, column=0)

tb.grid(row=5, column=0)

lightcombo.grid(row=6, column=0)

b11.grid(row=7, column=0)

l8.grid(row=1, column=2)

l9.grid(row=2, column=2)

lexperiment.grid(row=5, column=2)

lexperimentnumber.grid(row=6, column=2)
```

```
l5.grid(row=3, column=2)
l6.grid(row=4, column=2)
b10.config(state=tkinter.DISABLED)
db.config(state=tkinter.DISABLED)
dwellcombo.config(state=tkinter.DISABLED)
tb.config(state=tkinter.DISABLED)
lightcombo.config(state=tkinter.DISABLED)
b11.config(state=tkinter.DISABLED)
selectioncanvas = Canvas(selectionframe, height=10, width=40)
selectioncanvas.grid(row=0, column=1, rowspan=8, sticky="news")
dwellindicator = selectioncanvas.create_oval(10, 100, 30, 120, fill="black",
outline="black")
stimulationindicator = selectioncanvas.create_oval(10, 225, 30, 245, fill="black",
outline="black")
Label(infoframe, text="Guidelines", font=("Arial", 14, "bold")).grid(row=0,
column=0, padx=10, pady=(10, 5), sticky="w")
guidelines = [
    "Make sure both MCS-32 and Arduino are both installed and connected.",
    "From the File drop down menu, choose the location of MCS-32.exe.",
    'Click on "Run MCS-32" Button to start the software.',
    'From the MCS-32 software file menu go to Acquire->Input Control and select
50ohm',
    'Click "Connect" in the Arduino Connection frame to connect to Arduino.',
```

'Click "Power On" to enable the stimulation driver, according to the LED you will be using',

'Enable the PMT using the "PMT Enable" button.',

'Write the desired dwell time for the DAQ',

'Choose appropriate order of magnitude from the drop-down menu',

'Dwell Time values are limited between 125ns and 125μs',

'Confirm Dwell Time',

'Enter the amount of time desired for light pulses.',

'Choose appropriate order of magnitude from the drop-down menu',

'Stimulation Duration is limited between 250ns and 8.192ms',

'Confirm Stimulation Duration',

'From the list at the right-hand side, confirm the suggested data points and light pulse count',

'From the MCS-32 software, it is suggested you set the "Pass Length" parameter to the suggested value',

'Enter how many times the experiment will be repeated in "Acquisition Count".',

'Enter the duration of a single run in total.',

'For example, for a 1ms stimulation, a minimum of 10ms is required',]

guidelines2 = [

'Click Arm Experiment to arm the experiment.',

'From the list at the right-hand side, confirm the duration the experiment will take', 'Make sure you click on the "Start" Radiobutton in the software to Arm MCS-32',

'Clicking "Experiment" will begin the experiment.']

for i, guideline in enumerate(guidelines, start=1):

Label(infoframe, text=f"{i}. {guideline}", font=("Arial", 8), justify="left").grid(row=i, column=0, padx=10, sticky="w")

for i, guideline in enumerate(guidelines2, start=21):

Label(infoframe, text=f"{i}. {guideline}", font=("Arial", 8), justify="left").grid(row=i - 20, column=1, padx=10, sticky="w")

#main loop for the interface

root.config(menu=menubar)

root.mainloop()