University of Zurich

Master Thesis

Department of Physics
HS 2022 & FS 2023

Simulation of radioactive sources for quantum applications in dark matter searches

Author:
Noah Brugger

Supervisor:
Dr. Ilya Charaev

Group:
Prof. Dr. Andreas Schilling

7th September 2023

Remember to look up at the stars and not down at your feet. Try to make sense of what you see and wonder about what makes the universe exist. Be curious. However difficult life may seem, there is always something you can do and succeed at.

Stephen Hawking, 2006

Abstract

Understanding the Universe and its component is a herculean task for humanity all over the world and was in its interest since beginning of the attempt to understand the laws of Nature. One of the biggest research fields in recent years is dark matter.

Superconducting Nanowire Single Photon Detectors (SNSPDs) are an exciting new tool that could be used in the search for Dark Matter (DM). These detectors have a broad range of uses, like in quantum computing, quantum optics, bio-science and photon counting. SNSPDs are based on a superconducting film nanofabricated into a meandering wire geometry. The ability to detect light on a single-photon level is attractive for many applications especially where the detection of low light intensity is required. Because of this, SNSPDs are becoming an important technology for counting photons at optical and infrared wavelengths. They have high efficiency, low dark counts and excellent timing resolution. However, there is a problem. Above a certain threshold, the detection efficiency of SNSPDs drops off exponentially. But despite this issue, SNSPDs still hold a lot of promise for future applications.

This work aimed to test and understand how well several SNSPDs work in detecting DM in the low-eV range. We focused on SNSPDs made of superconducting Tungsten Silicide (WSi) films, which offers a high intrinsic quantum efficiency, a high sensitivity for single-photon detection while maintaining a reasonable high critical temperature. Transport measurements were conducted with a Physical Properties Measurement System (PPMS), which is able to cool down to 2K while applying a magnetic field of 9T. To do the actual detection measurements, we used a Oxford dilution Cryostat. This Cryostat cools down the SNSPDs to their superconducting state and keep them stable while we do the tests. By using these tools and methods, we aimed to learn more about how SNSPDs can help in our search for Dark Matter. Due to the challenging fabrication process, these SNSPDs were fabricated over the whole course of this project by Dr. Ilya Charaev at the Clean Room Facility of the ETH Zurich. We successfully achieved a saturated detection efficiency with a WSi-based SNSPD during our photon count rate (PCR) measurements by utilizing a $1550nm$ laser light supplied by a benchtop laser source from Thorlabs. The SNSPD has a width of $1\mu m$ and was measured at a temperature of $100mK$ in the Cryostat. At a bias current of $10\mu A$, the device exhibited a high intrinsic quantum efficiency, resulting in a stable plateau for photon absorption.

Given the low energy spectrum involved in SNSPD measurements, even seemingly negligible sources of particles, such as neutrinos and muons, and radioactive radiation, can influence the measurement. This background was simulated using the Geant4 toolkit, developed and provided by CERN. Geant4 is globally recognized for its efficiency in performing complex simulation related to particle tracking in 3D environments. For the purpose of this study, we used a Caesium-137 radioactive source positioned below the Cryostat, allowing us to simulate the effect of gamma radiation on the SNSPD in a controlled environment. The primary objective was to simulate the deposited energy onto the SNSPD while also recording the number of particles that collided with the surface of the detector. To understand the data produced by the simulation, we used the power of ROOT, a data analysis framework from CERN. This detailed investigation helped illuminate the influence of radiation on our measurements, offering valuable insights that will aid in developing techniques to mitigate such effects and improve the overall reliability and accuracy of SNSPD-based measurements. The simulation indicates that a full 24-hour livetime leads to a cumulative deposited energy of approximately $12keV$ when only gamma particles are accounted for. While this may appear to be a relatively low energy deposition, it's crucial to recognize that even minor energy fluctuations can have a significant influence

on our measurements, especially in the low energy scale that we are working within. The simulated results, therefore, not only substantiate the necessity for a rigorous shielding strategy but also emphasize the critical role that the strategic placement of the detector plays in an experimental setup.

Kurzfassung

Das Verständnis des Universums und seiner Komponenten ist eine grosse Aufgabe für die Menschen weltweit und war seit dem Versuch, die Gesetze der Natur zu verstehen, von Interesse. Eines der größten Forschungsfelder in den letzten Jahren ist die Dunkle Materie.

Superconducting Nanowire Single Photon Detectors (SNSPDs) sind ein aufregendes neues Werkzeug, das in unserer Suche nach Dunkler Materie (DM) entscheidend sein könnte. Diese Detektoren haben eine breite Palette von Anwendungen, wie in der Quanteninformatik, Quantenoptik, Biowissenschaft und Photonenzählung. SNSPDs basieren auf einem supraleitenden Film, der in eine mäandernde Drahtgeometrie nanofabriziert ist. Die Fähigkeit, Licht auf Einzelphotonenebene zu detektieren, ist für viele Anwendungen attraktiv, insbesondere dort, wo die Detektion von schwacher Lichtintensität erforderlich ist. Deshalb werden SNSPDs zu einer wichtigen Technologie zur Zählung von Photonen bei optischen und infraroten Wellenlängen. Sie haben eine hohe Effizienz, niedrige Dunkelzählraten und eine ausgezeichnete Zeitauflösung. Es gibt jedoch ein Problem. Über einem bestimmten Schwellenwert fällt die Detektionseffizienz der SNSPDs exponentiell ab. Aber trotz dieses Problems haben SNSPDs noch viel Potenzial für zukünftige Anwendungen.

Dieses Projekt zielte darauf ab, zu testen und zu verstehen, wie gut verschiedene SNSPDs bei der Detektion von Dunkler Materie (DM) im niedrigen eV-Bereich funktionieren. Unser Fokus lag auf SNSPDs, die aus supraleitenden Wolfram-Silizid (WSi)-Filmen hergestellt sind, die eine hohe intrinsische quantum efficiency und eine hohe Empfindlichkeit für die Detektion einzelner Photonen bieten, während sie gleichzeitig eine angemessen hohe kritische Temperatur aufrechterhalten. Transportmessungen wurden mit einem Physical Properties Measurement System (PPMS) durchgeführt, das bis auf 2K abkühlen und ein Magnetfeld von 9T anlegen kann. Für die eigentlichen Detektionsmessungen verwendeten wir einen Oxford-Dilution-Kryostaten. Dieser Kryostat kühlt die SNSPDs auf ihren supraleitenden Zustand ab und hält sie stabil, während wir die Tests durchführen. Mit diesen Werkzeugen und Methoden wollten wir mehr darüber erfahren, wie SNSPDs bei der Suche nach Dunkler Materie helfen können. Aufgrund des anspruchsvollen Herstellungsprozesses wurden diese SNSPDs im Verlauf dieses Projekts von Dr. Ilya Charaev in der Reinraumanlage der ETH Zürich hergestellt. Wir erreichten erfolgreich eine gesättigte Detektionseffizienz mit einem WSi-basierten SNSPD während unserer Messungen der Photonen-Zählrate (PCR) durch die Verwendung von 1550nm Laserlicht, das von einer Tischlaserquelle von Thorlabs geliefert wurde. Der SNSPD hat eine Breite von $1\mu m$ und wurde bei einer Temperatur von 100mK im Kryostat gemessen. Bei einem Bias-Strom von $10\mu A$ zeigte das Gerät eine hohe intrinsische quantum efficiency, was zu einem stabilen Plateau für die Photon-Absorption führte.

Angesichts des niedrigen Energiespektrums bei SNSPD-Messungen können selbst scheinbar vernachlässigbare Partikelquellen wie Neutrinos und Myonen sowie radioaktive Strahlung die Messung beeinflussen. Dieser Hintergrund wurde mit dem Geant4-Toolkit simuliert, das von CERN entwickelt und bereitgestellt wird. Geant4 wird weltweit für seine Effizienz bei der Durchführung komplexer Simulationen im Zusammenhang mit Partikelverfolgung in 3D-Umgebungen anerkannt. Für die Zwecke dieser Studie verwendeten wir eine radioaktive Caesium-137-Quelle, die unterhalb des Kryostaten positioniert wurde, um die Auswirkungen von Gammastrahlung auf den SNSPD in einer kontrollierten Umgebung zu simulieren. Das Hauptziel bestand darin, die auf den SNSPD deponierte Energie zu simulieren und gleichzeitig die Anzahl der Partikel zu erfassen, die mit der Oberfläche des Detektors kollidierten. Um die durch die Simulation erzeugten Daten zu verstehen, nutzten wir die Leistungsfähigkeit von ROOT, einem Datenanalyse-Framework von CERN.

Diese detaillierte Untersuchung half, den Einfluss der Strahlung auf unsere Messungen zu beleuchten und wertvolle Erkenntnisse zu gewinnen, die bei der Entwicklung von Techniken zur Minderung solcher Effekte und zur Verbesserung der allgemeinen Zuverlässigkeit und Genauigkeit von SNSPD-basierten Messungen hilfreich sein werden. Die Simulation zeigt, dass eine volle 24-stündige Lebensdauer zu einer kumulativen deponierten Energie von etwa 12keV führt, wenn nur Gamma-Partikel berücksichtigt werden. Obwohl dies eine relativ niedrige Energiedeposition zu sein scheint, ist es wichtig zu erkennen, dass selbst geringfügige Energiefluktuationen einen erheblichen Einfluss auf unsere Messungen haben können, insbesondere im niedrigen Energiespektrum, in dem wir arbeiten. Die simulierten Ergebnisse unterstreichen daher nicht nur die Notwendigkeit einer rigorosen Abschirmungsstrategie, sondern betonen auch die entscheidende Rolle, die die strategische Platzierung des Detektors in einem experimentellen Aufbau spielt.

Contents

1 Abbreviations

Dark matter	DM
Weakly Interactive Massive Particle	WIMP
Near- infrared	NIR
Single Photon Detector	SPD
Superconducting nanowire single photon detector	SNSPD
Dark Count Rate	DCR
Photon Count Rate	PCR
Quantum Efficiency	QE
Superconducting quantum interference device	SQUID
Scanning Electron Microscopy	SEM
Quantum Cascade Laser	QCL
Residual- resistivity ratio	RRR
Standard Commands for Programmable Instruments	SCPI
Markov chain Monte Carlo	MCMC
Secure Socket Shell	SSH
Physical Property Measurement System	PPMS

2 Theoretical background

2.1 Dark Matter

This chapter outlines the theory of dark matter and its supporting evidence. While the existence of dark matter has yet to be proven, there exist a reasonable amount of compelling evidence in its favor. In our observable Universe, large- scale structures could not form without DM, given the small amplitude of density fluctuations of the CMB. The subsequent chapters will describe some of the most promising and renowned approaches for understanding dark matter.

2.1.1 Zwicky's approach and the Virial theorem

The first detection of non-luminous matter took place in 1844 by Friedrich Willhelm Bessel, who described the orbit of the two binary stars *Sirius* and *Procyon* as being influenced by an invisible mass comparable to their own.[1] A couple of generations later, the Swiss physicist Zwicky estimated the mass of a galaxy to be significantly greater than what was predicted. To calculate the mass of a galaxy, he used the Mass- Luminosity relation, which takes the mass of the Sun as a reference:

$$\frac{L}{L_\odot} = \left(\frac{M}{M_\odot}\right)^a \tag{1}$$

In this equation, a typically has the value of $a = 3.5$ for main sequence stars. As a reference, Zwicky used the coma cluster, which has an observed luminosity of $L_{coma} \approx 8 \cdot 10^{12} L_\odot$. Compared to the measured luminosity it resulted in the following value:

$$\begin{aligned} M_{virial} &\gtrsim 2 \cdot 10^{15} M_\odot \\ M_{stellar} &\gtrsim 2 \cdot 10^{13} M_\odot \end{aligned} \tag{2}$$

These values were obtained using the Virial theorem, which is described as

$$\frac{1}{2} m v^2 \sim U \sim \frac{G M_{tot} m}{r} \tag{3}$$

From this, we can deduce that $M_{tot} \sim \frac{v^2 r}{G}$. Zwicky's estimate was around 400 times greater than what was observed in the blue band luminosity.[2] Based on this results, one can assume that this missing mass can be explained with dark matter. However, even stronger evidence exists.

2.1.2 Rotation curves

If one assumes the form of a galaxy as a disc, as stated by Kepler, one expects the gravitational force to drop as $1/r^2$. The rotational velocity of the galaxy declines in respect to Kepler:

$$v \propto \frac{1}{\sqrt{R}} \tag{4}$$

with R the radius of the galaxy. This relation is shown in figure 1 as labelled by "disk". If one now approximates the internal structure of halos as a sphere, the velocity dispersion can be assumed constant and independent of the radius: $\sigma(r) = \sigma_0$. Further one can define the parameter ρ_0 as the central density of the halo core and r_0 as the characteristic size of the core, we can relate r_0 to the particle velocity and the dynamical time of the central density:

$$r_0 \simeq \frac{\sigma_0}{2\sqrt{G\rho_0}} \tag{5}$$

with G the gravitational constant. At $r = r_0$ the matter self- gravity becomes significant. At $r > r_0$, the rotation velocities of particles seem to depend on the radius:

$$v_{rot}(r > r_0) = \frac{\sqrt{GM}}{r} = \sqrt{2}\sigma_0 = const. \tag{6}$$

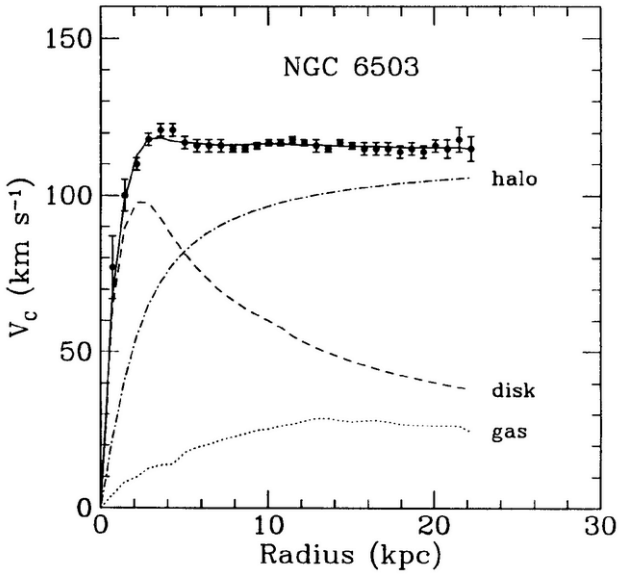


Figure 1: Rotational curve of the spiral galaxy NGC 6503.[3]

This relation can be observed in many galaxies, such as NGC 6503, as illustrated in figure 1 labelled by 'halo'.[3] These observations of flat rotation curves imply that mass M increases linearly with radius r, indicating the presence of a significant amount of unseen mass. This unseen mass, publicly known as dark matter, extends to considerable large distances from the centers of spiral galaxies.[4]

2.1.3 Gravitational lensing

> Space time tells matter how to move;
> matter tells space time how to curve.
>
> -John Wheeler

In 1915, Albert Einstein presented a novel theory of gravity, known as General Relativity, which refines and expands upon Newton's law of universal gravitation. According to this theory, Einstein predicted that the trajectory of a beam of photons that goes through the gravitational field of a massive body would deviate in proportion to the mass M of the body. This deflection can be described using the following angle:

$$\Theta = \frac{4GM}{c^2 b} \tag{7}$$

where c is the speed of light and b is the distance of the trajectory of the photon from the massive body.[5] Many different fields of research exploit the effect of gravitational lensing. It has provided important insights to the DM research:

- The Universe consists of five times more DM than baryonic matter
- DM interacts approximately normally with gravity
- DM is not in the size of planetary bodies
- DM is dynamically cold[6]

With this technique, one can identify potentially invisible mass merely from the deflection of photons in space, as seen in figure 2. This invisible mass could potentially be related to DM.

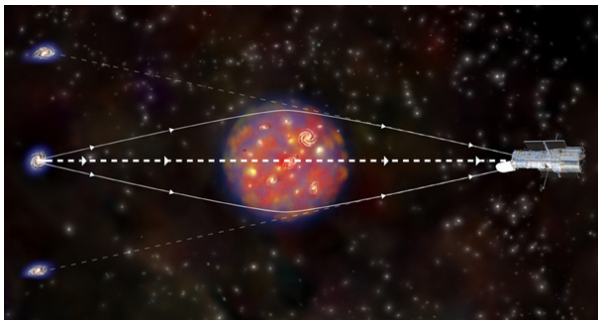


Figure 2: Strong gravitational lensing: Bending of light from background galaxies by a massive galaxy cluster can be used to estimate the mass of the cluster. (Illustration: NASA/CXC/M.Weiss)

2.2 DM candidates

Across many years, the quest for DM particles has yielded many possible solutions, but as of today, a definitive answer remains elusive. DM could be a neutrino, an axion, a WIMP, or it may even be a figment of our own imagination. While there is no scientific consensus, famous experiments such as XENON[7] and ADMX[8] continue to strive for an answer.
Almost all modern models of DM employ the standard concept of the quantum field theory to describe the properties of DM. This means it can be categorized in several groups of constraints:

- **Cold**: All DM today is cold, which means they are non- relativistic. Significant velocities at early times would erase small- scale structures
- **Massive**: Because of observations, DM has to be massive, but it has an extremely large range of viable masses ($10^{-22}eV$ up to $10^{59}eV$!)
- **Long- lived**: Given structure formation, DM could not have decayed away
- **Non- interacting**: DM does not interact with itself and possibly only very weakly with Standard Model particles
- **Electrically neutral**: DM is only detectable gravitationally and it does not appear that it has electromagnetic interactions

Based on these constraints, one could define several potential candidates. One of the most famous particles is the **WIMP**. The WIMP is a hypothetical particle which has no formal definition, but it is considered a new elementary particle which interacts with gravity and any other force, despite not being part of the Standard Model. It has a mass similar to the Higgs boson[1]. The existence of WIMPs would resolve a number of astrophysical problems related to DM, such as the abundance of DM particles. WIMPs align well with the model of a relic DM particle from the early Universe. The most promising experiment is the successor of the XENON Collaboration, the XENONnT experiment. It focuses on the search for WIMPs, by looking for rare nuclear recoil interactions in a liquid xenon target chamber. XENONnT is collecting more data, with improved detector conditions and an even lower background level due to a further improvement of the radon control and online removal system, aiming for an increased WIMP sensitivity over the following years.
One of the earliest and most attractive proposals for a DM particle was the **Axion**. This is not least due to its potential to resolve the strong CP problem in particle physics. The well fitting role as a DM particle comes as a bonus. It was postulated by the Peccei- Quinn theory in 1977.[9] The disadvantage of the Axion would be the mass range, in which it is only viable for a narrow range of masses.[10] This narrow range of masses of axions is located in the low meV range. One of the most promising detection methods is the SNSPD. A property of SNSPD which cannot be stressed enough, is the ability to detect low masses in the meV range.

[1] The Higgs boson has a mass $(125.25 \pm 0.17)GeV$.

This work will focus on the detection of such DM particles and will be described in chapter 2.4. Another influential branch of modern physics is the **Supersymmetry**, as it could resolve the gauge hierarchy problem. The basic idea behind the supersymmetry is that one can duplicate SM particles with particles of the same coupling but opposite spins. The masses of these SUSY particles are not identical to their SM counterparts, which would break the symmetry. The notion that supersymmetric particles could be good DM candidates became attractive when it was realized that R- parity[2] prevents the decay of the particle, given that DM particles need to be stable.[11]

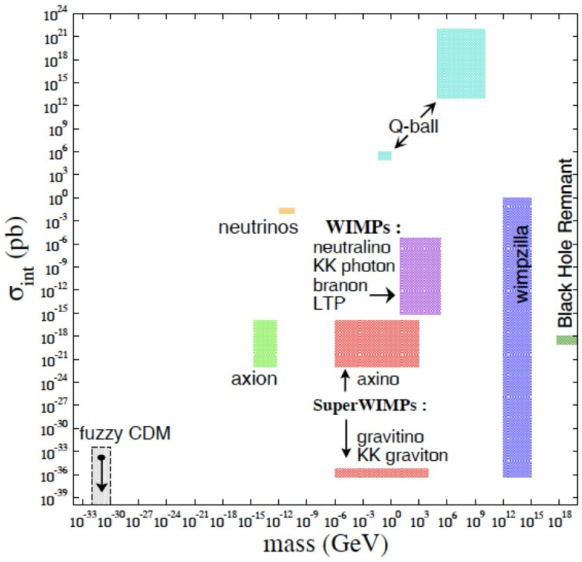


Figure 3: Interaction cross section with SM particles

As one can see in figure 3, there are many more potential candidates, such as sterile neutrinos or dark photons.

2.3 Introduction of Superconductivity

The phenomenon of *superconductivity* was discovered by the Dutch physicist Heike Kamerlingh Onnes in 1911 in the city of Leiden.[12] He observed a gradual decrease in the material dependent resistance of mercury as it was cooled down and liquefied. At a temperature of 4.2K, the resistivity suddenly dropped to a value that was immeasurable, as shown in figure 4[13]. This disappearance of resistance implies a resistivity of $\rho = 0$. This characteristic property of a metal called *superconductivity* induces infinite conductivity:

$$\sigma = 1/\rho \tag{8}$$

Equation 4 only holds for a superconductor and one does not see any dissipation, so the induced currents can persists indefinitely. It does not hold for standard metals, since the current does decrease rapidly due to dissipation. These currents are observable due to the magnetic field they generate.

[2]R-parity is a concept of particle physics, which tells us that in the MSSM the baryonic number and the leptonic number are no longer conserved.

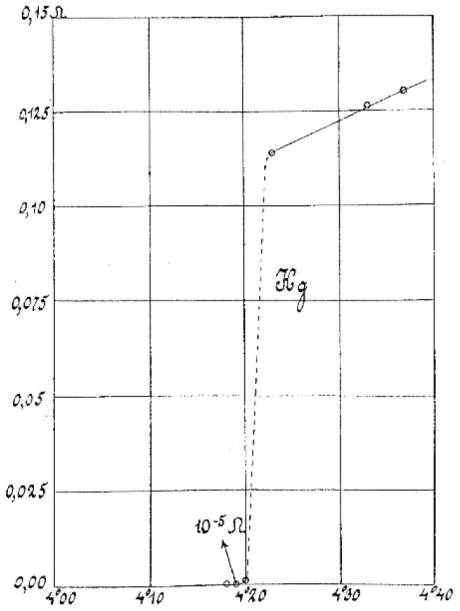


Figure 4: The resistance of Mercury as a function of the temperature as measured by H. Kamerlingh Onnes.[13] The critical temperature is achieved at 4.2K.

2.3.1 Critical temperature

In principle, superconductivity could enable the generation of very large currents and magnetic fields. However, the low temperatures, in which the experiments operate, are highly inconvenient, hence the value of the critical temperature is not only important for the physical aspects of superconductivity, it is also important for the practical application.
More generally, superconductivity appears quite frequently at lower temperatures. The critical temperature T_c is the temperature at which the electrical resistivity of a metal drops to 1% of its initial value. A superconductor operates within the temperature range of $T < T_c$. The transition is so sudden and complete that it seems to represent a transition to a different phase of matter; this superconducting phase is described by the BCS theory. Several materials exhibit superconducting phase transitions at low temperatures, as one can see some examples in table 1.

Superconducting material list		
Name	Chemical formula	Critical temperature T_c [K]
Gallium	Ga	1.1
Aluminum	Al	1.2
Indium	In	3.4
Mercury	Hg	4.2
Niobium	Nb	9.3
Tungsten Silicide	WSi	5.0 - 7.5

Table 1: Example values for T_c for different materials

2.3.2 Meissner- Ochsenfeld effect

There exists a second fundamental property of superconductivity. The two German physicists Walther Meissner and Robert Ochsenfeld cooled down a sample of tin below the superconducting transition temperature in the presence of an applied magnetic field H. They discovered that the magnetic field had strong changes in the vicinity of the tin, similar to a magnetic material. As a result, the field lines are expelled out of the superconductor, as shown in figure 5. This is also known as the Meissner-Ochsenfeld effect.

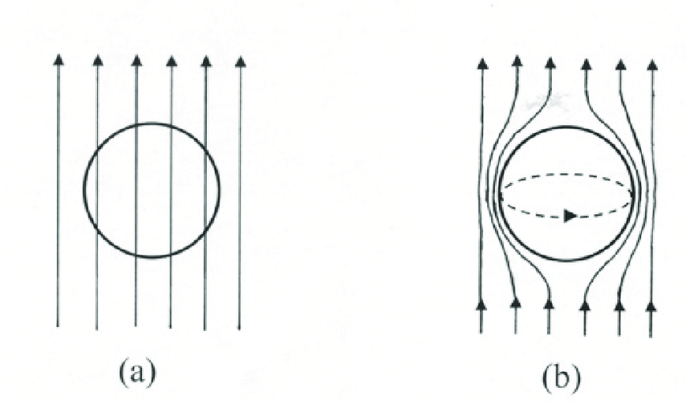


Figure 5: Schematic view of the magnetic field lines. For b) $T < T_c$ the magnetic field lines are pushed away from the superconductor, whereas in a) $T > T_c$.[14]

They detected this effect only indirectly because the magnetic flux is conserved by a superconductor: When the interior field decreases, the external field has to increase.[15]
To summarize the Meissner effect one can say that a magnetic material has a magnetic field of $B = 0$ even with an applied field $H \neq 0$ in the superconductor. One could define it as

$$B = \mu_0(H + M) = \mu_0(1 + \chi)H \text{ since } M = \chi H \tag{9}$$

with μ_0 the vacuum permeability, χ the magnetic susceptibility of the material. Hence, for a superconductor we get

$$\chi = -1 \tag{10}$$

This tells us that a superconductor is a perfect diamagnet. A diamagnet tends to weaken the external magnetic field. A superconductor does it perfectly by reducing the magnetic induction to zero.[16]
The Meissner- Ochsenfeld effect makes it clear that there must exist a critical field, beyond which the superconductor no longer exists at a given temperature. Hence, for a given temperature T, there is a critical field $H_c(T)$ beyond which superconductivity disappears:

$$H_c(T) = H_c(0)\left[1 - \left(\frac{T}{T_c}\right)^2\right] \tag{11}$$

where H_c is the critical value for of the magnetic field, T_c is the critical temperature and T is the current temperature.
The critical field decreases with increasing temperature and goes smoothly to zero at T_c. The threshold denoted as *critical value* of the magnetic field is specific for the material and has been measured for many superconductors as seen in figure 6.

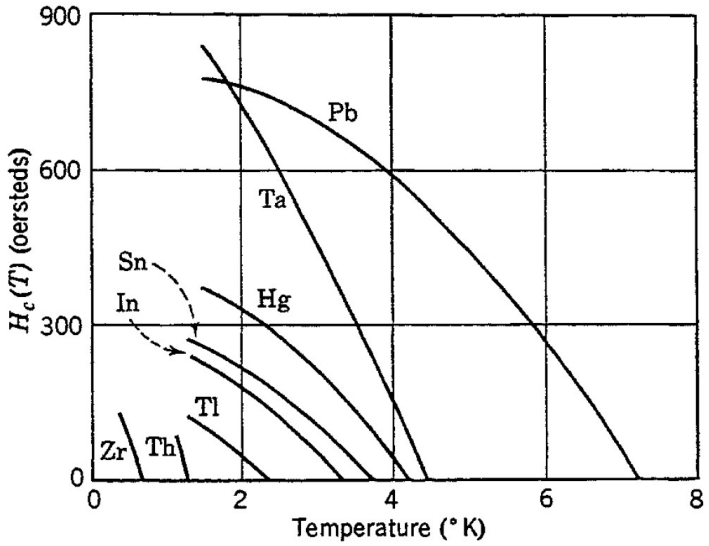


Figure 6: Threshold field curves for different superconductors.[17]

In theory, there exist two types of superconductors. For the sake of completeness, both will be discussed here shortly. In **Type-I** superconductors, superconductivity is abruptly destroyed via a first order phase transition, as shown by equation 11. A **Type-II** superconductor is a superconductor that exhibits an intermediate phase of mixed ordinary and superconducting properties.[18] The different phase diagram is shown in figure 7.

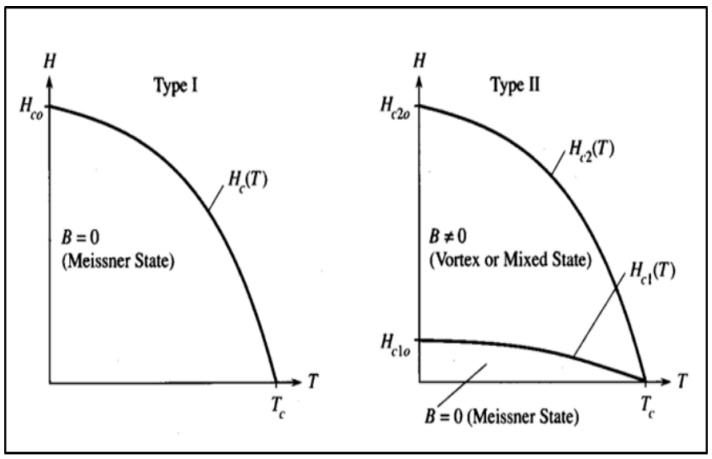


Figure 7: A typical phase diagram for a type I (left) and type II (right) superconductor[19]

2.3.3 BCS theory

The BCS theory, named after the three physicists John Bardeen, Leon Cooper and John Robert Schrieffer, is the first microscopic theory of superconductivity since the discovery of Heike Kamerlingh Onnes in 1911. The theory was rewarded with a Nobel Prize in 1972. A key aspect of superconductivity, as explained by the BCS theory, is the pairing of electrons into Cooper pairs through interaction with the crystal lattice. This coupling is called a phonon interaction. A cooper pair, also known as a BCS pair, is a pair of electrons bound together at low temperatures. This pair of electrons interacts above a Fermi sphere with the type of interaction one might expect due to the phonon and the screened Coulomb fields. Even if their total energy is above zero, if there is a non- zero attraction between the two electrons, they can form a bound state. The properties of a non-interacting system of such bound pairs strongly suggests a state that could lead to superconductivity. At first glance, one might think that the pairing of two electrons violates the Pauli exclusion principle, which states that no two identical fermions may occupy the same quantum state. However, when the electrons pair up, they behave as a bosonic state, therefore the Pauli exclusion principle is still fulfilled.[20] This bosonic state can condense into the ground state, and this pair condensation is the basis for the BCS theory of superconductivity. The pairs should be chosen such that transitions between them are possible, i.e., they all have the same total momentum. To form the ground state, the best choice is $\mathbf{k}\uparrow, -\mathbf{k}\downarrow$.[21]

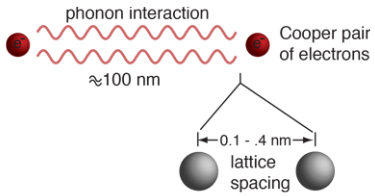


Figure 8: Schematic representation of a Cooper pair. The distance between the two interacting electrons is much higher than the lattice spacing.

Further, the BCS theory derived several important predictions. One of the most important ones for this work is the prediction of the dependence of the value of the energy gap Δ at temperature T. The ratio between the value of Δ at $T = 0$ and at T_c takes a universal value:[22]

$$\Delta(T = 0) = 1.76 k_B T_c \tag{12}$$

2.4 Superconducting Nanowire Single Photon Detectors

The nature of DM remains one of the most important unsolved mysteries of our Universe. As already stated, the problem with identifying DM is its broad energy range. After decades of theoretical and experimental research in the high- energy range in the electroweak scale, the recent focus has shifted towards lighter masses. Nonetheless, the direct detection of sub- GeV DM particles requires new detectors with much lower thresholds than previous experiments.[23] This has motivated scientists all over the world to develop new detection techniques, among which superconductors are the most promising. They are in principle capable of detecting light sub-MeV DM particles with mass as light as 1keV for scattering and even lower in the case of absorption at 1meV.[24] However, for the full application in DM research, more and more advanced technological devices were needed. Technologies from other fields in physics or chemistry were introduced, for example in [25, 26]. In 1971, a laser was used for the first time to disrupt the superconductivity of certain films, forming a resistive state that could not be explained by heating effects.[27]. This experiment showed that the energy of the absorbed photons resulted in a nonequilibrium state, with hot excited quasiparticles at a higher temperature than the Cooper pairs in the superconductor. More than two decades ago, in 2001, G. N. Gol'tsman et al. reported the first demonstration of a superconducting microbridge with a wavelength of $0.81\mu m$. For their achievement, they used Niobium Nitride on a thin sapphire substrate.[28] This development is later known as the SNSPD.

2.4.1 Single photon detection: Basic operation principle

In 1996, it was proposed that patterning the superconducting film into a device with comparable dimensions to the size of a hotspot would result in an infrared sensitive SPD.[29] A SPD can create an electrical signal only if a photon is absorbed. This signal is defined above the level of noise, therefore in the absence of illumination no electrical signal should be returned. But as in every field of research, all devices do not have ideal characteristics, therefore one has to define a performance metric: The detection efficiency η. It depends on optical properties of the film, detector geometry and energy of the incoming photons. It consists of multiple efficiencies, such as the coupling efficiency $\eta_{coupling}$, which describes the loss of photons before reaching the detector, the absorption efficiency $\eta absorption$ and the registering efficiency $\eta_{registering}$, which is the probability that the detector generates an output electrical signal after a photon is absorbed. Taking all these contributions into account, the overall system detection efficiency is given by the product of those three efficiencies:

$$\eta_{SDE} = \eta_{coupling} \cdot \eta_{absorption} \cdot \eta_{registering} \tag{13}$$

Further, the intrinsic device detection efficiency is defined as

$$\eta_{dde} = \eta_{absorption} \cdot \eta_{registering} \tag{14}$$

η_{dde} is often reported as the quantum efficiency. The detection efficiency varies with the energy of incident photons, as shown in figure 9. Further it was found that η_{SDE} changes with the variation of the transport current through the detector. For each current exists a cut- off wavelength λ_c. If the wavelength is smaller than λ_c, η_{SDE} is almost constant, as shown by the colored area in figure 9. For larger wavelengths, the detection efficiency drops rapidly.[30] Other performance metrics quantify limitations in SPD performance. Stray light and electrical noise can also potentially mimic the optical signal. These false detection events are called dark counts, usually quantified in terms of dark count rate (DCR).

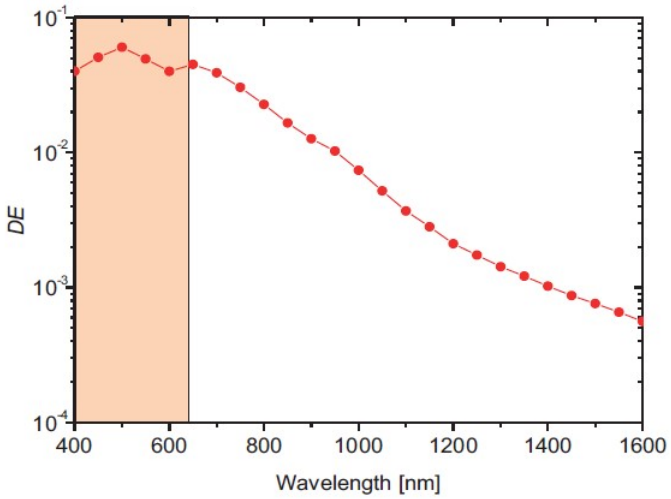


Figure 9: The detection efficiency of SNSPD measured for different wavelengths.[30]

2.4.2 Dark count rate

The DCR is a parameter that denotes the rate of false counts over time in relation to the detection events, representing the noise level of a SNSPD. In the measurement method, DCR is the number of counts in unit time recorded with no illumination. DCR consists mainly of background DCR.[31, 32] Background DCR is difficult to avoid since photons are present wherever thermal electromagnetic radiation is present. According to Planck's law of black- body radiation, matter with a certain temperature will radiate photons due to the thermal movement of the particles within the matter. In the case of fiber- coupled SNSPDs, the thermal radiation from the fiber at room temperature will produce photons that may traverse through the fiber and may be detected as dark count. The wavelength distribution of black- body radiation can be described by the Planck function:[33]

$$B_\lambda(T)d\lambda = \frac{2\pi hc^2}{\lambda^5}\frac{1}{e^{hc/\lambda kT}-1}d\lambda \tag{15}$$

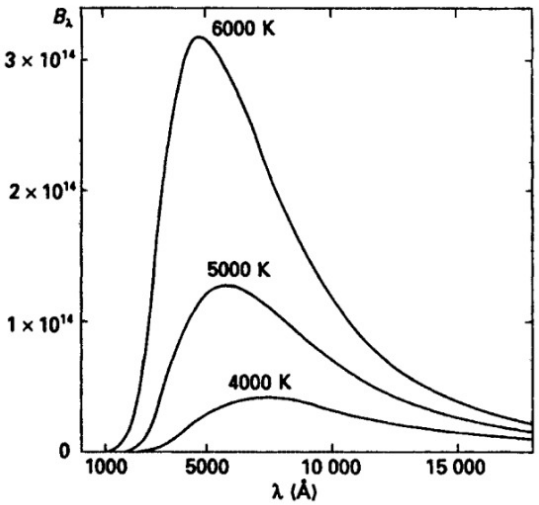


Figure 10: Spectral energy distribution of the Planck's function. The units are radiation energy per unit wavelength $B_\lambda(T)$.

2.4.3 Functionality of SNSPDs

After summarizing the origin of single- photon detection, one has to understand the basic principle of operation of SNSPDs. Regarding the functionality of SNSPDs, one has to keep in mind that photons can contain information using different degrees of freedom such as time, momentum and polarization. SNSPDs are fabricated by using superconducting films of around 50nm thickness. In most cases, as even in our case, they are fabricated in a meander structure covering as much space as possible, within the range of a few hundred square micrometers. The single- photon detection can be roughly separated into six steps, as depicted in figure 11.
In SS), the superconducting nanowire maintained well below the critical temperature is direct current biased just below the critical current. In I), a single photon gets absorbed. A photon is the quantum of an electromagnetic field, including electromagnetic radiation, such as light and radio waves, which is an elementary particle. Further, NIR photons exist. In this thesis we will focus on the detection of traditional NIR photons and visible wavelength.[31][3] This incoming photon, which has a wavelength of $1550nm$ in our case, gets absorbed by the SNSPD, which leads to the creation of a cloud of quasi particles, also known as a hotspot, in II). The supercurrent is forced to flow along the periphery of the hotspot.

[3] $400 - 2000nm$, with a typical energy of $0.6 - 3eV$.

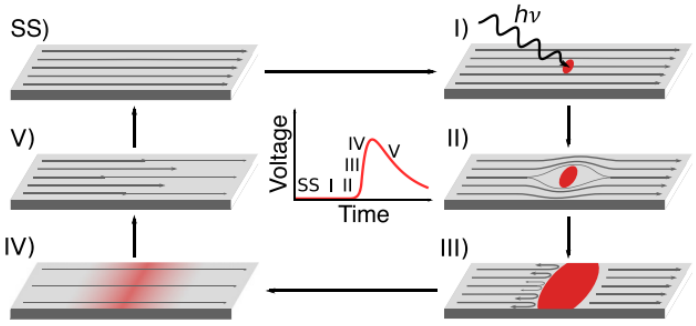


Figure 11: Schematic operation of SNSPD [34]

Since the nanowires are narrow, the local current density around the hotspot increases, exceeding the superconducting critical current density. This is turn leads to the formation of a resistive barrier across the width of the nanowire. The increasing resistance leads to redirection of the bias current from the nanowire to the electronic readout in IV). Once the temperature of the resistive area cooled down to a certain value, superconductivity is restored (V)). The bias current through the nanowire returns to I) again. The described reset dynamics is limited by the kinetic inductance L_k of the device.[34] Proper self-resetting of the device requires that this inductive time constant be slower than the intrinsic cooling time of the nanowire hotspot, also known as latching.[35]

2.4.4 Current state of research of SNSPDs

In recent years, SNSPDs have been proposed for being used in the field of DM research, leading to remarkable progress, as shown in several publications, including those by Hochberg et al.[36, 37]. As one can read in their publication, new prototypes set strongest constraints for DM- electron interactions in the range of sub-MeV masses in the case for absorption. As discussed by Hochberg et al. in [36], their prototype has an extremely low dark count rate, which is the rate of counts in the absence of incoming light, and combined with the extremely low threshold, SNSPDs have the potential to change the way of detecting direct dark matter. Current state of the art is a detection efficiency of over 98 % at a wavelength of 1.55μm.
SNSPDs are becoming a dominant technology in quantum optics and quantum communication, primarily because of their low timing jitter and capability to detect individual low- energy photons with high quantum efficiencies.[38] They are used in a broad range of important application areas: Quantum Key Distribution, which is a secure communication method that implements a cryptography protocol involving components of quantum mechanics,[39] development of quantum computing, space- to- ground communications and fibre sensing applications.[40] As one sees, in the past few years, SNSPD research has advanced apace. The basic device performance, as introduced in chapter 2.4.1, has improved dramatically, offering high η_{SDE}, low DCR at infrared wavelengths and due to those improvements, SNSPDs are used in a wide range of applications.

2.5 Sample production

SNSPDs are intricate devices whose fabrication involves a series of complex, highly precise steps. Initially, a thin superconducting film, typically composed of elements such as NbN or in our case WSi, is deposited onto a substrate, often made of SiO_2 or sapphire. This deposition process is usually carried out using techniques such as magnetron sputtering or atomic layer deposition to ensure a uniform and precisely controlled thickness.[30] The samples used in this work are done externally at the Clean Room Facility at the Swiss Federal Institute of Technology (ETH) in Zurich with the deposition, EBW and Reactive Ion Etching methods. SEM images of our device are shown in figure 12.
The samples were produced by my supervisor Dr. Ilya Charaev over the course of this whole project.

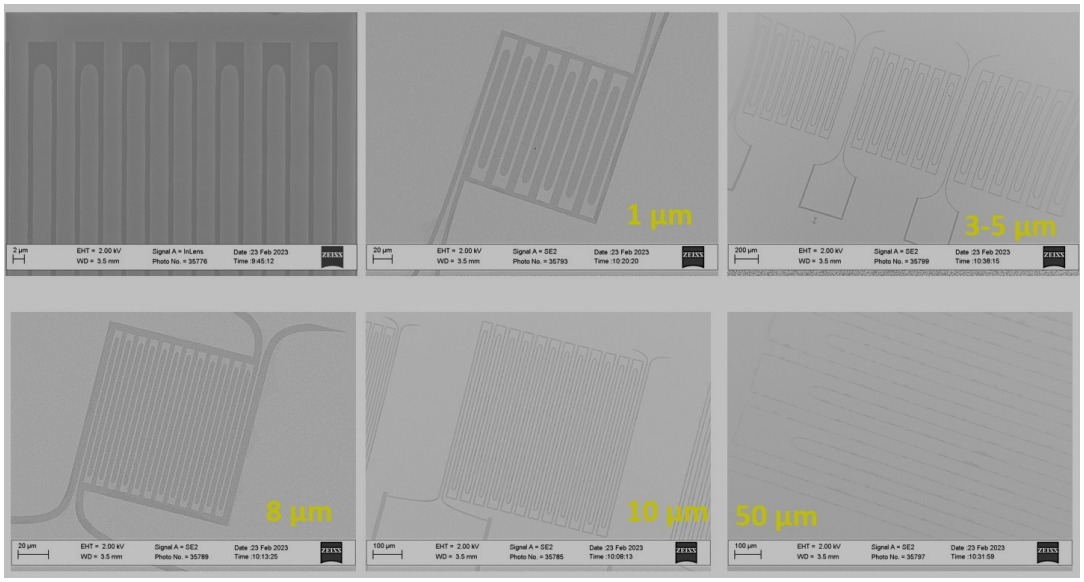


Figure 12: SEM images of WSi SNSPD device taken at different magnifications. One can see the entire device with two contact pads and different widths.

The thin WSi layer is the active detector layer, but the other layers still modify the detector response to deposited energy and momentum.

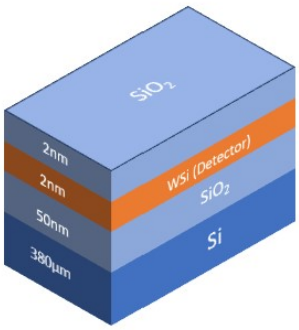


Figure 13: Schematic cross section of a single nanowire.

Following the deposition, lithography is used to define the nanowire pattern. Electron-beam lithography, which employs a focused beam of electrons to draw custom shapes on a surface covered with an electron-sensitive film, is often the method of choice given its high resolution and versatility. This process essentially "writes" the desired nanowire pattern onto the superconducting film.[41]

2.6 Laser theory

Another important device which was not mentioned up to this point in this work, is the laser. The term "Laser" is an acronym for *Light Amplification by Stimulated Emission of Radiation.* We are operating at a wavelength of $1550nm$, which is in the NIR- regime. NIR single photon detection is emerging as an important technology for various applications, such as Earth meteorology[42] or exoplanetary exploration.[43]
SNSPDs fabricated with superconducting films with energy gaps of a few meV have achieved excellent performance in both visible and near-infrared wavelengths with system detection efficiency higher than 95%.[44] Extending the wavelength of SNSPDs into the NIR region is highly attractive, since we are operating with a rather high wavelength, which lets us use low energy photons, since

$$E = \hbar\omega = \frac{\hbar c}{\lambda} \tag{16}$$

The material used, WSi, has a low energy gap of $1.52meV$.[44, 45]

2.6.1 Classical laser theory

A laser is a device that emits light through a process of optical amplification based on the emission of electromagnetic radiation. The basic operation setup of a LASER consists of a laser medium, which contains a collection of atoms, molecules, ions or a semiconducting crystal, a pumping process to excite these particles into higher energy levels and a feedback element, as shown in figure 14.[46]

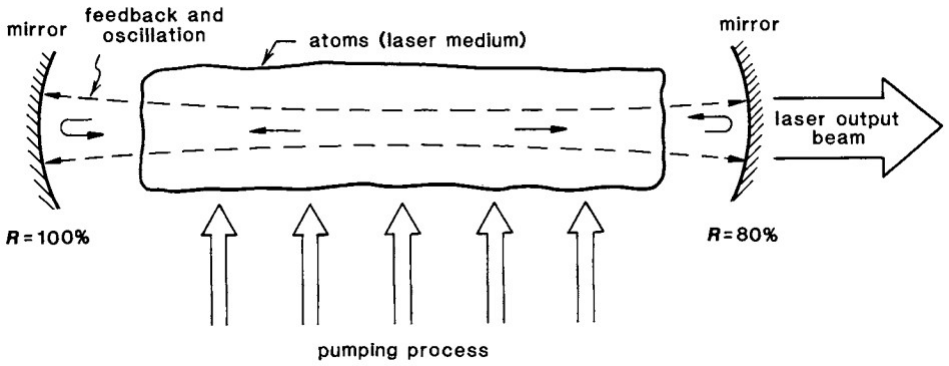


Figure 14: Schematic depiction of a typical laser.[46]

One of the fundamental concepts of a laser is the stimulated emission. In the classical view, electrons with a larger radius around the nucleus have higher energy than closer electrons. But electrons are forced to take on discrete positions due to quantum mechanical effects. An electron can absorb energy from an incoming photon only if there exists energy levels that match the energy of the photon. When an electron is excited from a lower state E_1 to a higher state E_2 with $E_2 > E_1$, it will not remain on this level forever. This interaction with the photon causes the electron to drop to E_1. This liberated energy transfers to the electromagnetic field and creates a second photon with the same characteristics as the incident one, as illustrated in figure 15. On a bigger scale, this can create a beam of light with a certain wavelength and direction.[47]

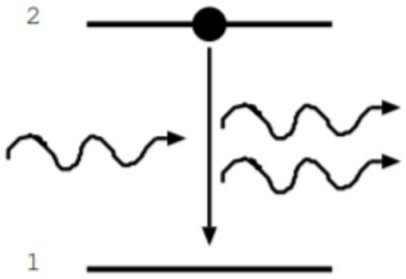


Figure 15: Diagram of the energy levels in the process of stimulated emission.

There exist other types of operating methods of lasers. For the sake of completeness, two other common methods are listed. The first one is the spontaneous emission. As the name implies, it is the same principle as the stimulated emission, the only difference is the missing pumping, as depicted in 14 and shown in figure 16. An electron in E_2 drops down spontaneously to E_1 and emits a quantized amount of energy in the form of a photon.[48]

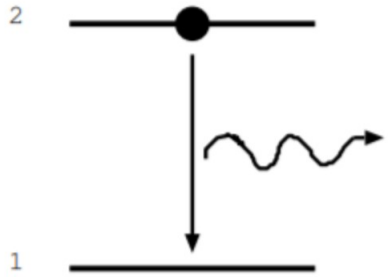


Figure 16: Diagram for the spontaneous emission of a photon

The final type is the induced absorption. In principle, it is the inverse of the spontaneous emission. An electron, which is in E_1 can switch to E_2 by absorbing a photon, thereby altering the internal momentum of the electron and its internal quantum number, as depicted in figure 17.[48]

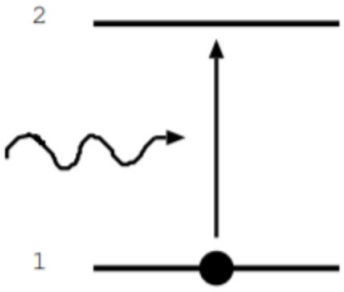


Figure 17: Diagram for the induced absorption of a photon

2.6.2 Quantum Cascade Laser

A new form of laser, which emerged in the mid 1990s, is the Quantum Cascade Laser (QCL). It was first developed by Jérôme Faist, Federico Capasso, Deborah Sivco, Carlo Sirtori, Albert Hutchinson, and Alfred Cho at the Bell Laboratories.[49] Contrary to a classical laser system, semiconductor lasers rely on transitions between energy bands in which conduction band electrons and valence band holes injected into the active layer through a forward bias pn- junction, radiatively recombine across the band gap. The band gap essentially determines the emission wavelength. Once an electron transitioned from the conduction band to the valence band and emitted a photon, it can tunnel into the next period of the structure where another photon can be emitted. Therefore a single electron can emit multiple photons and leading to a *cascade*, producing a higher output power than semiconductor laser diodes. This is a unique property of a QCL.[50] This cascade is schematically shown in figure 18.

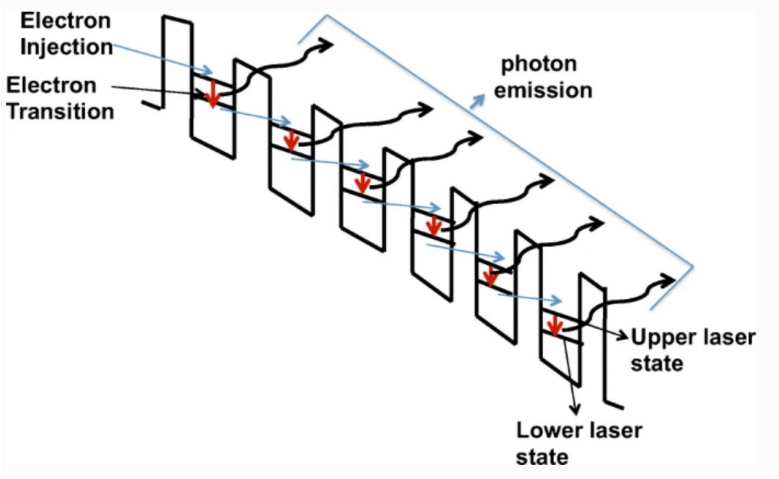


Figure 18: In a cascade structure, the electron can tunnel to the next period and the process repeats.[50]

Such QCLs are mostly based on a three- level- system.[51] The three- level model is shown schematically in figure 19.

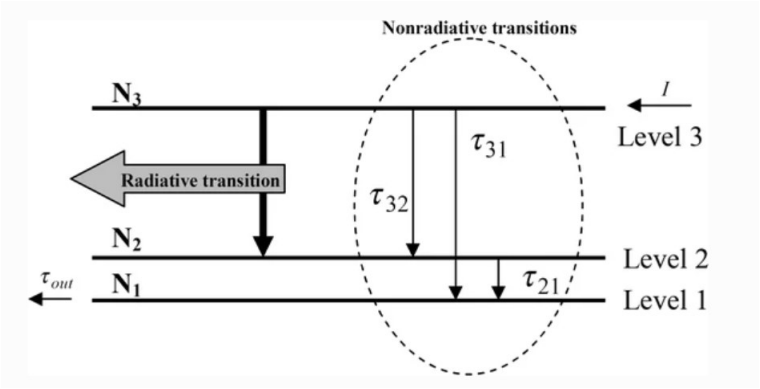


Figure 19: Energy levels of the three- level rate equations.[52]

The dynamics of carrier and photon numbers in a three- level QCL can be described in terms of the following four first- order differential equations, where the number of electrons in the ground state will be denoted as N_1, the number of electrons in level 2 and 3 by N_2 and N_3 respectively, further denotes

N_{ph} the number of photons:

$$\begin{aligned}
\frac{dN_3}{dt} &= \frac{I}{e} - \frac{N_3}{\tau_3} - (N_3 - N_2)Gf(N_{ph})N_{ph} \\
\frac{dN_2}{dt} &= \left(\frac{1}{\tau_{32}} + \frac{1}{\tau_{sp}}\right) N_3 - \frac{N_2}{\tau_{21}} + (N_3 - N_2)Gf(N_{ph})N_{ph} \\
\frac{dN_1}{dt} &= \frac{N_3}{\tau_{31}} + \frac{N_2}{\tau_{21}} - \frac{N_1}{\tau_{out}} \\
\frac{dN_{ph}}{dt} &= N\left(N_3 - N_2\right)Gf(N_{ph})N_{ph} + N\beta\frac{N_3}{\tau_{sp}} - \frac{N_{ph}}{\tau_p}
\end{aligned} \tag{17}$$

N is the number of cascade stages, in this case $N = 3$. G is defined as the gain coefficient. It represents the factor by which an input beam is amplified by a medium.
$f(N_{ph} = (1 + \varepsilon N_{ph})^{-p}$ is the gain- saturation term which includes $0 < p \leq 1$ and ε as the phenomenological gain- saturation factor.[53] β denotes the spontaneous emission coupling coefficient, e is the electric charge, I is the injection current and the three τ are the lifetimes corresponding to the transitions between the levels. τ_{out} represents the electron escape time between two stages and τ_{sp} denotes the radiative spontaneous relaxation time between level 3 and 2. Since further explanations would extend the boundaries of this work too far, please refer to the publication in *Optical and Quantum Electronics* by D. Mohsen and F. Kiazand.[52, 54]

3 Characterization of SNSPDs

3.1 Characteristics of samples

3.1.1 Sheet resistance

Sheet resistance is a term used to describe the resistance of thin films that are uniform in thickness. As for production of SNSPDs, our process is based on thin films with a thickness of a few nanometers. Due to L.J. van der Pauw, the sheet resistance R_S is more convenient to use. The so-called four-point probe Van der Pauw method allows us to accurately measure the sheet resistance of any arbitrary shape, as long as the shape is approximately two-dimensional and the electrodes to measure the voltage are placed on its perimeter.[55]

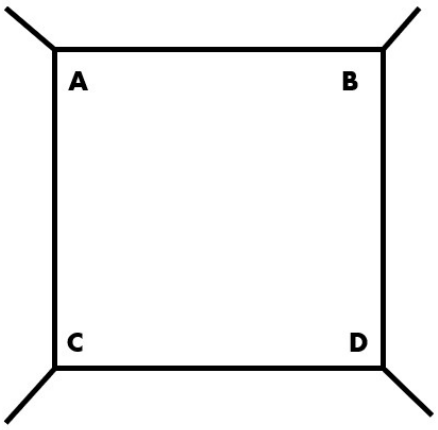


Figure 20: Scheme of a square shaped films with 4 different contact pads to calculate the sheet resistance.

For a bridge structure, we can write the total resistance as

$$R = \rho \frac{L}{dw} \tag{18}$$

with ρ the specific resistivity, which is depending on the material, L is the length, d the thickness and w is the width of the structure. The sheet resistance, which is depending on the geometry, can be written with a dependence on R:

$$R = R_S \frac{L}{w} \text{ with } R_S = \frac{\rho}{d} \tag{19}$$

which can be explained as the resistance per square. In figure 20, one can define a specific resistance $R_{AC,DB}$ as the potential difference between contact pads **D** and **B** per unit current through contact pads **A** and **C**. A relationship between $R_{AC,DB}$ and $R_{CD,BA}$ is known as the Van der Pauw equation[56]:

$$exp(-\pi R_{AC,DB}/R_S) + exp(-\pi R_{CD,BA}/R_S) = 1 \tag{20}$$

R_S can be written in the form:

$$R_S = \frac{\pi}{ln2} \left(\frac{R_{AC,DB} + R_{CD,BA}}{2} \right) \cdot f \left(\frac{R_{AC,DB}}{R_{CD,BA}} \right) \tag{21}$$

where f satisfies

$$\frac{R_{AC,DB} - R_{CD,BA}}{R_{AC,DB} + R_{CD,BA}} = f \cdot arccosh \left(\frac{exp(ln2/f)}{2} \right) \tag{22}$$

This is known as the Standard Van der Pauw method, which can be simplified in our case due to the defective edges of the square. By applying a current I between the contact pads and measuring the voltage V on the other side, we end up with the final formula for the sheet resistance:

$$R_S = \frac{\pi}{8 \cdot ln2} \sum_{n=1}^{4} \left(\frac{V_n + V_{n+1}}{I} \right) \cdot f \left(\frac{V_{n+1}}{V_n} \right) \tag{23}$$

For a homogeneous square film, V_n and V_{n+1} are close to each other, resulting in setting $f \sim 1$.[57, 58] As an example the corresponding values for a $3nm$ thick and $11 \cdot 11mm^2$ $W_{0.59}Si_{0.49}$ film are shown below.
To simplify (and automatize) this calculation, a short script has been written in Python, as one can see in chapter 8.2. The formula 23 has been implemented as:

```
sum = (V_1+V_2)/I + (V_2+V_3)/I + (V_3+V_4)/I + (V_4+V_1)/I

SheetResistance = (pi/ (8*np.log(2)))*sum
```

Where the four different V are taken as an input. The resulting sheet resistance in units of Ω/sq is given below.

```
Use voltage in units of [V]
Enter V1: 0.286
Enter V2: 0.250
Enter V3: 0.254
Enter V4: 0.291

1224.8703283287991
```

The sheet resistance is one of the most important characteristics to describe a SC film. It is also used to determine the total kinetic inductance L_{tot}, the critical temperature T_c and the Residual- resistivity ratio (RRR), which will be discussed in the following sub chapters.

3.1.2 Kinetic inductance

The kinetic inductance is the inertial mass of the current carriers, in our case the electrons. The concept arises from the expression of the energy:

$$E = \int_V \frac{1}{2} \mu H^2 d\tau + \int_{Conductor} \frac{1}{2} m v^2 n d\tau \tag{24}$$

where μ, m, n are constants from the material, V is the whole space over which is integrated. If our conductor is homogeneous with a uniform cross section, we can simplify it to:

$$E = \frac{1}{2} L_M I^2 + \frac{1}{2} \underbrace{[m/ne^2 \cdot (l/\omega)]}_{L_{tot}} I^2 \tag{25}$$

with L_{tot} the total kinetic inductance.[59] For further reference, the width over length l/ω will be defined as the number of squares N. Using the BCS- theory, as depicted in 2.3.3, it is possible to derive L_k of a superconducting nanowire. If one assumes we are in the low frequency limit ($hf \ll k_B T$), one can derive a relation between the imaginary conductivity σ_2 and the normal state conductivity σ_n:

$$\frac{\sigma_2}{\sigma_n} = \frac{\pi \Delta(T)}{hf} tanh \frac{\Delta(T)}{2k_B T} \tag{26}$$

where $\Delta(T)$ was defined in 12 related to the BCS theory. At $T \ll T_c$, this formula can be simplified to:

$$\frac{\sigma_2}{\sigma_n} = \frac{\pi\Delta(T=0)}{hf} = 1.76\pi k_B T_c/hf \tag{27}$$

The imaginary component of the impedance sets the kinetic inductance as $L_k = 1/(2\pi f\sigma_2)$.[60] Solving 27 for σ_2, inserting it in L_k results in:

$$L_k \sim 1.39\frac{R_s}{T_c} \tag{28}$$

If L_k is multiplied by N, one gets the total kinetic inductance[61]:

$$L_{tot} = L_k \cdot N \tag{29}$$

3.1.3 Relaxation time

The relaxation time is defined as the minimum time the SNSPD needs to reinstate the superconducting phase, as shown in figure 11. The relaxation time is given by the following equation:

$$\tau_r = L_{tot}/R_{load} \tag{30}$$

where we insert L_{tot} from 25. τ_r is defined as the relaxation time. In this experiment, R_{load} is set to 50Ω.[59] However, being limited by L_{tot}, the devices have reset times that are several orders of magnitude larger than their intrinsic response time.[62]
As an overview, table 2 shows a compilation of values for different samples. These values are important for characterizing the samples and are necessary for further computations. They are calculated using the equations provided before.

Sample Characterization			
Sample width [μm]	N	$L_{tot}[\mu H]$	$\tau_r[ns]$ expected
1	2485	1.714	34.287
2	2373	1.638	32.754
3	2461	1.698	33.971
1	638	0.440	8.804
3	634	0.437	8.747
5	634	0.437	8.747
40	150	0.104	2.07
30	150	0.104	2.07
50	2566	1.771	35.417

Table 2: This table contains characteristics of a sample used in this experiment. The following parameters are given: $T_c = 2K$, $R_s = 1000\Omega/sq$, $L_k = 690pH/sq$.

3.1.4 Residual- resistivity ratio

The RRR is a common indicator for the level of purity of a metal. In the ideal case, one should achieve $RRR = 1$. The main elements causing this impurities are oxygen, nitrogen, hydrogen and carbon.
In general cases, the electrical resistivity of metals can be described via the Mathiessen's rule:[63, 64]

$$\rho = \rho_{res} + \rho_{ph}(T) + \rho_m \tag{31}$$

where the first term is the residual resistivity at $T \sim 0K$ caused by electron- impurities scattering. The second term represents the temperature dependent electron- phonon interaction and the last term is the resistivity term in a magnetic field.

Resistivity caused by scattering of conduction electrons on defects is proportional to their concentration. Those resistance coefficients $\frac{\Delta\rho_i}{\Delta C_i}$ are individual for different atoms and can be implemented as followed:[65]

$$\rho = \rho_{ph}(T) + \sum \frac{\Delta\rho_i}{\Delta C_i} C_i \tag{32}$$

In pure metals having a lattice without structural defects, the resistivity $\rho \to 0$ for T close to zero.[66] The RRR is defined as the ratio:

$$RRR = \frac{\rho(T = 300K)}{\rho_{max}(T)} \tag{33}$$

where ρ is defined at around room temperature at 300K. ρ_{max} is the maximum value for the resistivity over the entire temperature spectrum. Normally for superconductors, this value is reached just above $T_c \sim 30K$.[67]

3.2 Latching in SNSPDs

Latching is a serious issue in SNSPD technology.[68] Referring to equation 30 and 28, it becomes clear that the relaxation time τ depends on L_{tot}. The factors that determine whether or not a device will latch, are determined by how the dynamics depend on the set bias current I_b, L_{kin} and R. As explained in chapter 2.4.3, figure 11 demonstrates the creation of the hotspot when the photon creates a normal domain. Formation of a hotspot precludes the operation of the SNSPD. After the hotspot is formed, the wire will remain in the normal state even after the photon has been detected. In this state, the wire will not detect any photons, therefore the photon counts will differ from the "real" value. In the worst-case scenario, latching prevents reset and precludes subsequent photon detection. This issue arises when the L_{tot} is too high or low. This malfunction is called latching, if the relaxation time is in the wrong time frame. To prevent latching, the time for the hotspot to cool must be shorter than the inductive time constant, which governs the resetting of the current in the device after hotspot formation.[35, 62]

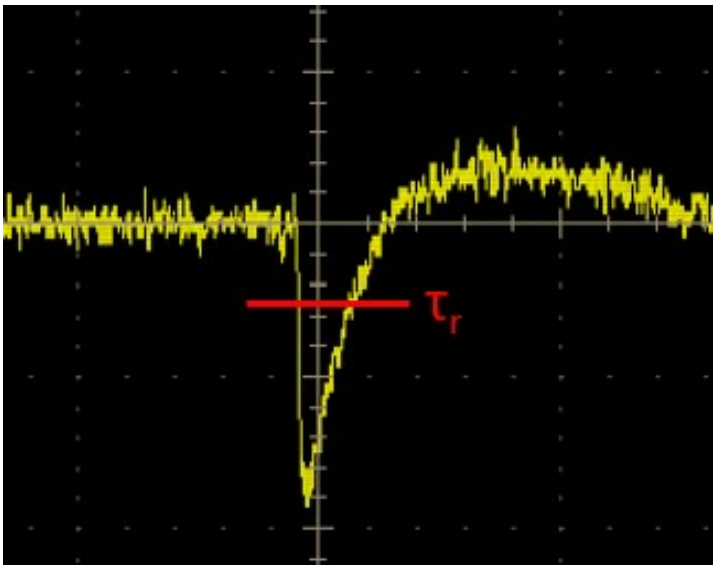


Figure 21: Screenshot from Tektronix Oscilloscope showing single photon pulse with $\tau_r \sim 30ns$. The relaxation time can be readout at 30% of the voltage spike.

3.3 Sample characterization with PPMS

To perform preliminary measurements with the samples, a *Physical Properties Measurement System* (PPMS) is used. These systems were developed by Quantum Design and allow to perform several types of experiments in a controlled environment under external conditions. They consist of a liquid helium based cryostat which enables the regulation of the temperature ranging from $2K$ to $350K$. Superconducting magnets produce a perpendicular magnetic field up to 9 Tesla. For our measurements, the PPMS is mainly used to perform $R(T)$ or $R(H)$ curves at a constant current I to determine the transport properties and the characteristics of the samples.[69]
When considering the resistivity curves, one has to introduce the theoretical model proposed by Aslamazov and Larkin in 1986.[70] It describes the theory of fluctuation conductivity in dependence of T_c. As discussed in chapter 3.1.1, we are dealing with a 2D- film, therefore the theory suggests that the resistance- temperature relation can be expressed as:

$$R(T) = \frac{R_S}{1 + R_S \frac{1}{16} \frac{e^2}{\hbar} \left(\frac{T_c}{T - T_c} \right)} \tag{34}$$

Here, e is the electrical charge and $\hbar$ is the reduced Planck constant.
As a first test, a 4nm WSi sample has been placed into the PPMS with no applied magnetic field. The black dots shown in figure 22 represent the measured values for the resistivity as a function of the temperature. The blue line shows the Aslamazov- Larkin approximation as given in equation 34. We considered T_c and R_S as fitting parameters to map experimental data over a reasonable temperature range.[71] This allows us to read T_c directly from the plot. For the best fit, one can see that $T_c \approx 1.59K$, as an example.

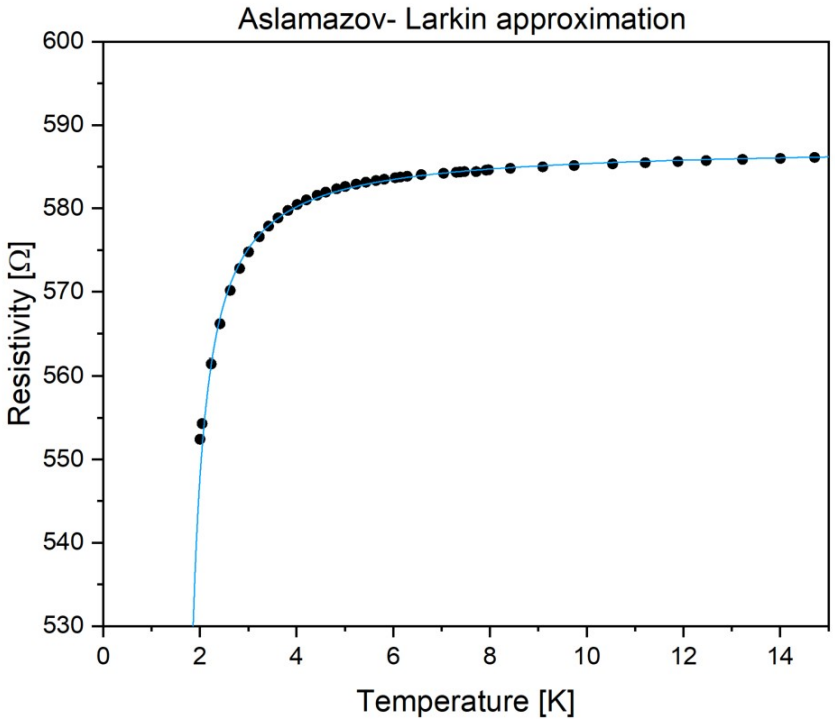


Figure 22: Measurement of a 4nm unpatterned WSi film and the corresponding approximation of Aslamazov- Larkin

Moreover, by applying an external magnetic fields perpendicular to the film's surface, the $R(H)$ dependence for a set of magnetic fields in units of $Oe = \frac{1000}{4\pi} A/m$ can be measured, as shown in figure 23. This figure contains multiple measurements, each taken at a constant temperature ranging between $2K$ to $25K$.

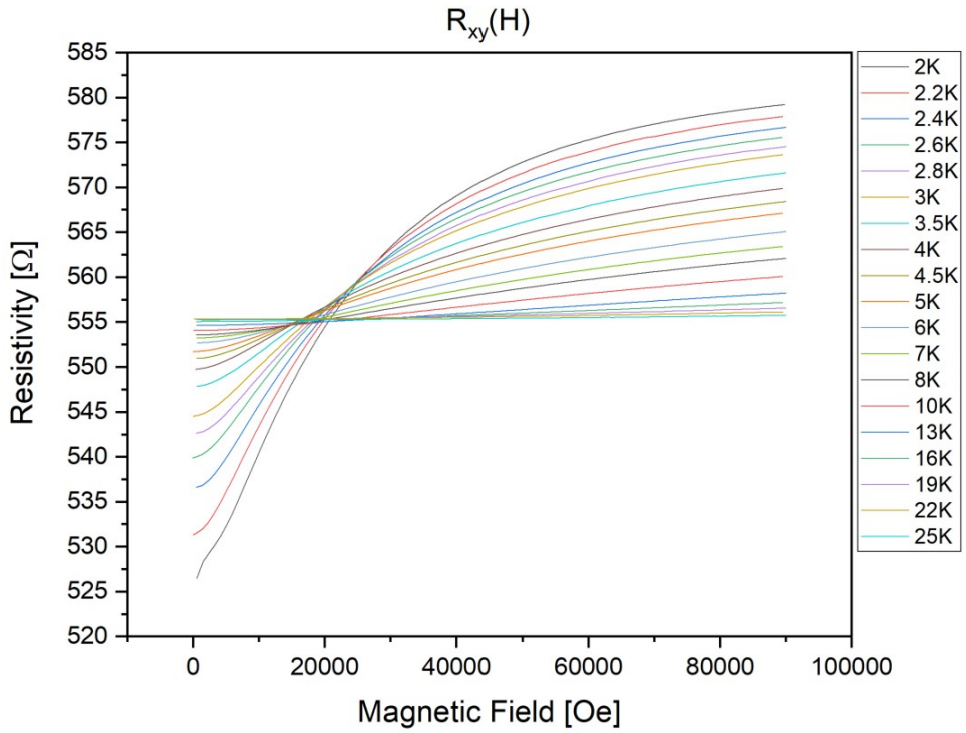
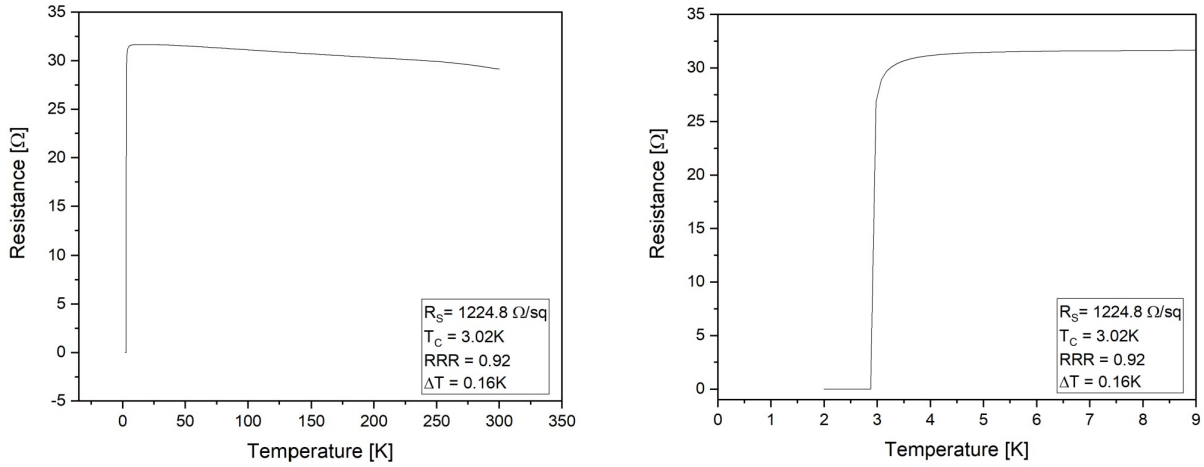


Figure 23: Resistivity as a function of the magnetic field

For even further characterization, one could compare all the parameters defined in chapter 2.3.1 and 3.1.4 in a single plot. From figure 24a, the parameters shown in table 3 can be derived.



(a) $R(T)$ measurement with different parameters. (b) Highlighted transition point in range of $0K$ to $9K$.

Figure 24: $R(T)$ measurement in the PPMS with a MoSi film with $T_c = 3.02K$ and a width of 4nm.

As the samples have been mounted in the PPMS, the resistance as a function of the temperature could be measured as well. One of the significant observations that can be made from data visualizations such as figure 24a is the identification of the transition point. This is the temperature at which the material shifts from its normal conducting state to a superconducting one. This phase transition is marked by an abrupt shift to a flat line on the graph, indicating a sudden onset of constant resistance.

Sample parameters		
Parameter	Calculation	Description
Sheet resistance	Van der Pauw method	Calculated as explained in chapter 3.1.1.
Critical temperature	$T_c = T\left(\frac{1}{2}R_{max}\right)$	Mostly used for comparison with Aslamazov- Larking approximation
Residual- resistivity ratio	$RRR = \frac{R_{300K}}{R_{max}}$	See chapter 3.1.4
Temperature difference	$\Delta T = T(0.9R_{max}) - T(0.1R_{max})$	Width of transition

Table 3: Compilation of parameters needed for characterization

3.4 Measurement of IV- curves

With all important parameters introduced, the first application can be explained. The IV- curve, a diagram which shows the relationship between the bias current and the voltage, is a useful tool to measure the critical current I_c of SNSPDs. I_c is another characteristic property of the samples. To obtain IV- curves, we are using only a part of our final setup, since we do not need any light for this characterization. Our setup consists of the GS200 Voltage/Current source by Yokogawa Test & Measurement Corporation,[72] the Cryostat itself containing the SNSPD and a Bias- TEE that connects the voltage source to the Cryostat, as illustrated in figure 25. To perform reasonable measurements, the temperature in the Cryostat at the cold stage should be around $200mK$.

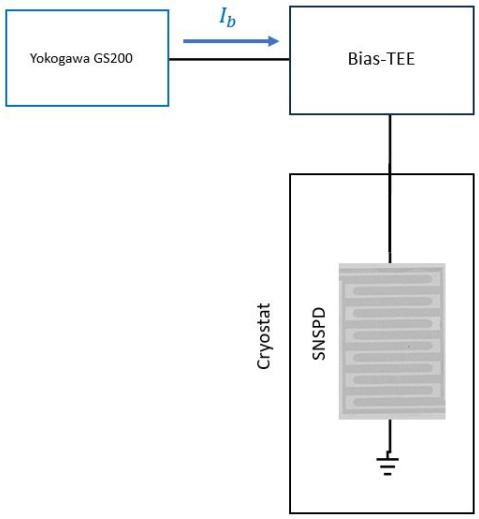


Figure 25: Schematic of the experimental setup for measuring IV- curves. As a remark, the single components do not have values, since we are using different samples.

The GS200 is a low voltage or low current DC source with a high accuracy and a high resolution. It has the advantage of producing extremely low- noise DC signals that are required for such low-energy experiments containing SNSPDs. Further it can be connected via General Purpose Interface Bus (GPIB)[4] to a USB of a computer. This connection enables remote control of the device and automation of the measurement process. Since there are many measurements with different samples included in this work, a short script was developed to automatize the current sweeping and the creation of IV- curves. It is also possible to sweep the voltage and measure the current, but the decision fell to sweep the current and measure the voltage, since the range for the current is more fitting for our

[4]GPIB, also known as IEEE 488, is a short-range digital communications 8-bit parallel multi-master interface bus specification developed by Hewlett-Packard as HP-IB (Hewlett-Packard Interface Bus)[73].

needed purpose with $\pm 200mA$.
The programming language Python 3.9.12 was used within the Anaconda environment to write the script. The first step is to establish the connection between the computer and the current source. To realise this, the Python library PYVISA 1.8 was implemented and the correct bus was allocated. This was achieved with these two lines of code:

```
rm = pyvisa.ResourceManager()
yokogawa = rm.open_resource('GPIB::1::INSTR')
```

If the connection is successful, one can setup the source to measure the voltage and reset the current. As a safety measure, the device is reset after every successful measurement. Communication with the device requires the use of Standard Commands for Programmable Instruments (SCPI), an ASCII-based programming language which can be implemented in Python or C++.
It was developed in the 1990s and is used mainly to control test and measurement devices such as an oscilloscope or in our case a voltage/ current source.

```
yokogawa.write("*RST")
yokogawa.write(":SOUR:FUNC VOLT")
yokogawa.write(":SOUR:PROT:VOLT 30")  # voltage source
yokogawa.write(":SOUR:RANG 1E-1")
yokogawa.write(":SOUR:FUNC CURR")  # current source
yokogawa.write(":SOUR:LEV 0") # set current to 0
yokogawa.write(":SOUR:RANG 100E-5")
yokogawa.write(":OUTP ON")
```

The final line in this code activates the device to start the output. After setting all the parameters to zero, setting the range for the voltage and the current to the desired values, one could define the range for the values one wants to measure. In these measurements, we want to verify if the IV- curve is symmetric, since this information allows us to classify the samples into functional samples and degraded samples. Our loop logic therefore has to measure up to a certain value x, then decreases to $-x$, and goes back up to zero. The program prompts for an absolute input value for x and will divide it into a given number of steps over the entire range. These values for the steps and the value of x are stored in multiple linear spaces (linspaces), which will be used later to sweep the current. The linspaces look as follows:

```
#loop to sweep current
first_range = np.linspace(0,x-x/step, step)
second_range = np.linspace(x,-x+x/step,2*step)
third_range = np.linspace(-x,0,step)
run = np.concatenate((first_range,second_range, third_range))
print(run)
voltage = []
current = []
```

The last two lines are needed to save the measured values for the voltage and the calculated values for the current into two new arrays, which will be used later to plot the curve.
The next step would be the implementation of the measurement itself. For this purpose, we create a for- loop that will sweep over all the values in *run*. The measured data for the voltage is stored in a new array, which will be converted into an array containing float values. These float values will be appended to the newly created *voltage* array. This loop looks like this:

```
for I in run:
    yokogawa.write(":SOUR:FUNC CURR")
    I = I
    yokogawa.write(":SOUR:LEV " + str(I))
    time.sleep(0.4)
    try:
        while not yokogawa.query("*OPC?"):
            pass
    except Exception as e:
        print("Error: ", e)
        break
    data = yokogawa.query(":FETC?").split(",")
    vread = float(data[0])
    #i = float(data[1])
    current.append(I)
    voltage.append(vread)
    print(f"--> Current = {I} A, Voltage = {vread} V")
```

Special interest in this paragraph is on line 5. Since the Yokogawa needs a short time to reset, one has to implement a *sleep* command, which gives the device time to reset. Without this line, it could happen that the measurement before was not finished and the new measurement should start. This could lead to some serious problems or wrong data taking. One has to make sure that this time is not too long, since it will significantly increase the time for the whole program to run. The next line starts an error break mechanism, which will help reduce the amount of incorrect data acquisition. It will stop the measurement if the device fails to receive new input for each measurement, or if it is always the same input value. The final line of code is just for an overview during the run to check if the values are in the desired range.
As the program concludes, it's necessary to deactivate the device's output and virtually sever the connection between the device and the computer. The execution of these steps can be achieved with the following two lines of code:

```
yokogawa.write(":OUTP OFF")
yokogawa.close()
```

The rest of the code contains the saving of the data using Pandas and the plotting of the data. For more information about the code and the full code with all lines, please refer to appendix 8.1. One of those plotted IV- curves is shown in figure 26 as an example. As one can read from the graph (or from the data, which is also saved), The critical current for this sample is $I_c = 28.59\mu A$, the width of this sample is $3\mu m$.

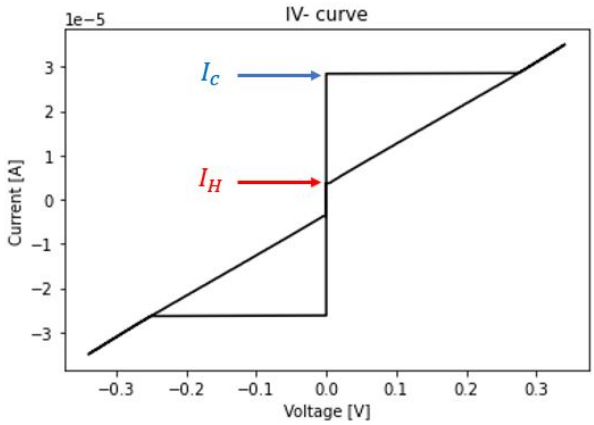


Figure 26: Example for an IV- curve measured in the setup described in figure 25. The width of this nanowire is $3\mu m$.

3.5 Setup for PCR measurement

In this section, the objective was to determine the PCR as a function of the bias current. This assessment aids in identifying the optimal bias current range for conducting experiments, aiming to achieve a high PCR while minimizing the DCR. For this purpose, the setup shown in figure 25 will be extended with multiple devices. The first addition is the SR400, a photon counting device developed by Stanford Research Systems. The second device is an oscilloscope by Techtronix. Initially, the oscilloscope is employed to verify the sample's capability to discern individual photon pulses. If this is successful, the photon counter will count the total number of voltage spikes by the SNSPD. These photons originate from a laser source by Thorlabs. As an addition, a pair of amplifiers were intgrated in the setup to amplify the signal. The final setup can be seen in figure 27.

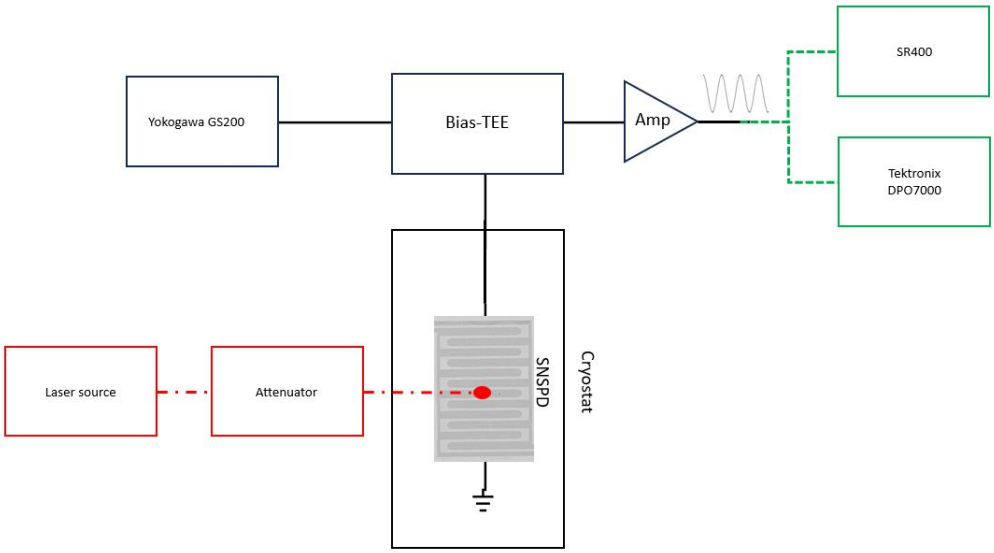


Figure 27: Experimental setup for the PCR measurement

The counted photons are plotted as a function of the bias current with the help of OriginLabs[5]. Ideally, the graph should resemble the representation provided in Figure 28. The plateau shown in figure 28 is created, because the SNSPD has absorbed the maximum number of photons it can accommodate up to the point where the critical current is achieved.

[5]OriginLab Corporation ©2023

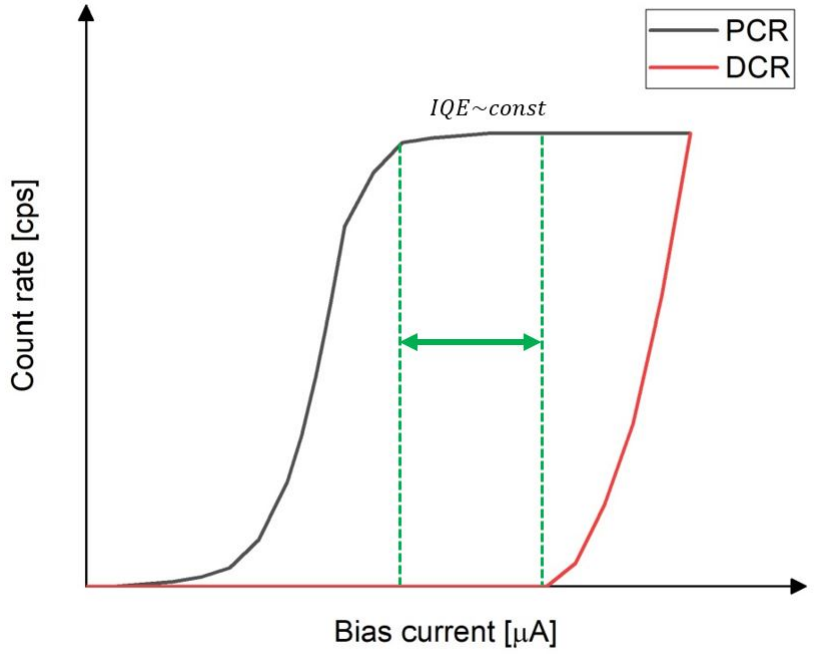


Figure 28: Ideal case for a PCR measurement. At low bias current nearly no photons are detected. While increasing the bias current, the photon count goes up rapidly until reaching a plateau. At this point the intrinsic quantum efficiency is constant and around 100%. The DCR is represented as the red function and is 0 up to a certain point. The green area gives the ideal range of value for the bias current.

SNSPDs possess a high intrinsic QE and therefore it is possible to achieve a high signal-to-noise ratio, especially within a specific bias current range. This QE is influenced by the characteristics of the SNSPD, as explained in the preceding chapters. Although the total intrinsic QE is high, the overall system efficiency depends on other system properties, as explained in chapter 2.4.1. Hence, it's crucial to establish a credible simulation and robust background shielding.

3.6 Introduction to SOLIDWORKS

One part of this project was the construction of some pieces for the inside of the Cryostat, which will need a holder for the sample and the SNSPD. This holder should reduce the background as well. The construction will not be a big part of this thesis here, but since it is a big part of the project, it will be shortly introduced nonetheless. Before the desired object can be constructed and produced it has to be drawn in a CAD application. For this, SOLIDWORKS was used. The SOLIDWORKS CAD software is a mechanical design automation application that is able to design ideas, experiments with features and dimensions, and detailed drawings. This software was developed by Dassault Systèmes SolidWorks Corporation (DS SolidWorks) and offers a variety of design options. A big advantage of SOLIDWORKS is the possibility to save the constructed parts in different file formats, such as STEP, PLY and PDF. An overview over the different formats are shown in figure 29.
The finished drawings were brought to the mechanical workshop at the University of Zurich, which could produce the pieces by using the STEP files created in SOLIDWORKS.[74]
Further, a big part of this project is the simulation of this experiment in a simulation software such as Geant4. For this, the holder and the cryostat itself were created as a STEP file, which could be implemented in Geant4.

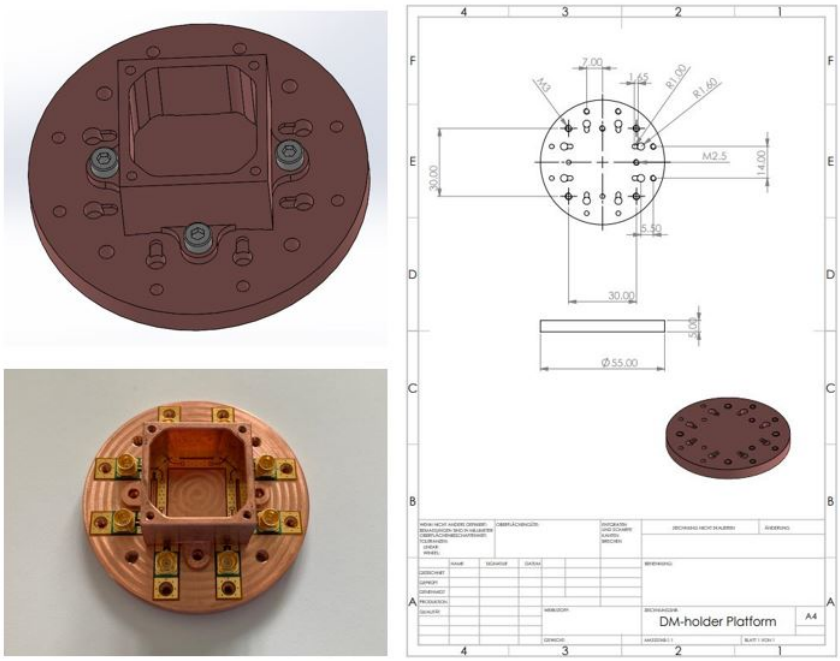


Figure 29: Different file formats for displaying the holder: Upper left: STEP, Bottom left: Picture of the constructed holder including PCB, right: PDF drawing

4 Geant4 Simulation

4.1 Introduction to Geant4

One of the big parts of this project is the simulation of the laboratory environment. The goal was to simulate a reasonable projection of the Cryostat with the surrounding lab. We wanted to check the impact of incoming γ-rays onto our SNSPD inside the constructed holder. This simulation is important, since one wants to compare the simulated data with the measured data.
For this, we are using the software Geant4. It is short for Geometry and Tracking and is a toolkit for the simulation of the passage of particles through matter by using MCMC. It was developed in the late 1990s by the Geant4 Collaboration and it was one of the first to use object oriented programming in C++.[75] A key requirement for Geant4 was the flexibility for different applications, therefore it was designed as a modular toolkit that offers class categories with coherent interfaces such that the user can assemble the program as desired. The software is used by a number of research projects around the world, as for an example XENON[76], the LHCb at the LHC and Borexino at the Gran Sasso Laboratory.[77–79] Geant4 provides predefined modules for the implementation of the detector geometry, user interfaces, particle tracking and visualization. Further, it features a wide variety of libraries containing physical models and processes such as radioactive decays. An advantage of Geant4 is the experimentally determined data of particles and atoms, such as decay chains with their α-, β- and γ- energy spectra.

4.2 Fundamental steps of the simulation

As already mentioned in the previous chapter, the objective is to replicate the laboratory at the University of Zurich, which contains the Cryostat and our SNSPD. In pursuit of the highest fidelity to reality, we aim to simulate the entire lab, including all major furnishings such as tables, the two PPMS systems, computers, etc. In principle, for a simulation. The geometry of the detector has to be implemented in the Geant4 framework. The geometry has to contain information about the materials from which the components are made of. Once this part is set, it can be compiled and does not need to be modified again. Afterwards, one can use macros to still communicate with the code and to specify the type, direction, etc. of the radiation Geant4 should simulate. A simulation run is stored in a ROOT file that contains all the information for every simulated event and its deposited energy. To summarize this procedure, to accomplish this project, the following steps must be undertaken:

1. Set up Geant4 as the primary simulation software on Visual Studio Code.
2. Develop a basic simulation code.
3. Implement CADMesh to directly load Mesh files into Geant4.
4. Create a .STL file of the Cryostat to be used for simulation.
5. Construct the remaining objects in the laboratory using Geant4.
6. Write a macro file to test the simulation with the radioactive source.
7. Write the files for the analysis done in ROOT.

4.3 Geant4 installation

Before embarking on code creation for Geant4, it may be useful to choose a suitable code editor. The choice in this work fell on Visual Studio Code (VS Code) provided by Microsoft Corporation. Launched in 2015, this open- source and powerful code editor offers numerous advantages. It supports a variety of programming languages, such as JavaScript, Python or C++, which is used in this work. Further it contains support for syntax highlighting, seamless integration of GitHub and the direct implementation of GitHub Co- Pilot, which is an AI- driven coding tool used to find the correct

commands and automatically correct spelling errors.
However, the most useful tool in VS Code is its ability to connect directly to a SSH. This feature makes it possible to connect the personal computer to the server of the University of Zurich, which already has a complete version of Geant4 installed. This allows efficient code editing and execution, which significantly enhances the productivity.
For this project, we used Geant4.10.6.2.[6] The initial step involves sourcing all the necessary script files, for which there exists a simple command:

```
alias setG4="source /app/cern/geant_4.10.06.02/bin/geant4.sh
&& source /app/cern/geant_4.10.06.02/share/Geant4-10.6.2/geant4make/geant4make.sh"
```

In order to streamline the startup process, these commands should be included in the .bash file of the corresponding server. This practice reduces the amount of work that needs to be done at each startup. Pertaining to the analysis, the following command should also be sourced:

```
source /app/cern/root_v6.14.00/bin/thisroot.sh
```

This will source ROOT, which is an open-source data analysis framework extensively used in high energy physics and in a wide variety of other topics in the field of physics, as for example the ATLAS COLLABORATION[80] and the Hyper Kamiokande experiment.[81] It is an object- oriented computer program which is written in C++.[7]
The foundation of code used in this simulation was provided by Dr. Jose Javier Cuenca Garcia from the group of Prof. Laura Baudis. This solid basis made it possible to understand the logic structure of the Geant4 toolkit and it could be modified for our purpose.

4.4 Development and modifying base code

The first step involves the creation of the appropriate CMakeList. CMakeLists.txt files contain a set of directives and instructions that describe the project's source files and targets. It contains all the necessary libraries and executables. This code is compiled using CMAKE 3.20.04 and Qt 5.15.2. The complete file is provided in chapter 8.3.1, with only the most signicant parts being highlighted here.
Initially we need to locate the Geant4 package, to verify if it is properly set and build the user interface. This task is achieved with the following few lines of code:

```
option(WITH_GEANT4_UIVIS "Build example with Geant4 UI and Vis drivers" ON)
if(WITH_GEANT4_UIVIS)
  find_package(Geant4 REQUIRED ui_all vis_all)
else()
  find_package(Geant4 REQUIRED)
endif()
```

Subsequently, we need to fetch the source directories for both Geant4 and ROOT. They are located on the server of the University, therefore we simply need to set the path as a global path:

```
add_definitions(${ROOT_CXX_FLAGS})
include(${Geant4_USE_FILE})
include_directories(${PROJECT_SOURCE_DIR}/include
                    ${Geant4_INCLUDE_DIR}
                    ${ROOT_INCLUDE_DIR})
```

[6] The most recent version is Geant4.11.1.2. In principle, this project is runnable on the newest version, but there may be compatibility issues with some imported modules and commands.
[7] Copyright ©ROOT Team 2023

```
6
7   #----------------------------------------------------------------------
8   # Locate sources and headers for this project
9   file(GLOB sources ${PROJECT_SOURCE_DIR}/src/*.cc)
10  file(GLOB headers ${PROJECT_SOURCE_DIR}/include/*.hh)
```

The final two lines in this paragraph simply set the source directory for the header and source files within the corresponding folder.
Since we are using the C++ language, one can write an executable, that can be run independently. This will significantly reduce the compilation time and the final file can function as a standalone program, which can be accessed by others.[8] As previously stated, our experiment is named QROCODILE, which consequently becomes the name of the file as well. The creation of the executable can be accomplished with a single line of code:

```
1   add_executable(QROCODILE_G4 qrocodile.cc ${headers})
```

Once the CMakeList is properly configured and operational, the focus can shift to the main executable. This file is the main file which links all the other files in this project:

```
1   // create the run manager
2   G4RunManager* TheRunManager = new G4RunManager;
3
4   // Detector Construction
5   QrocoDetectorConstruction* DetCon = new QrocoDetectorConstruction();
6   TheRunManager->SetUserInitialization(DetCon);
7
8   // Physics List
9   G4VModularPhysicsList* ThePhysicsList = new Shielding;
10  TheRunManager->SetUserInitialization(ThePhysicsList);
11
12  //Test Radioactive source
13  G4VModularPhysicsList* physicsList = new QBBC;
14  physicsList->RegisterPhysics(new G4RadioactiveDecayPhysics());
15  TheRunManager->SetUserInitialization(physicsList);
16
17  // Primary generator action
18  QrocoPrimaryGeneratorAction* ThePrimaryGeneratorAction = new QrocoPrimaryGeneratorAction();
19
20  // Analysis manager
21  QrocoAnalysisManager* TheAnalysisManager = new QrocoAnalysisManager(ThePrimaryGeneratorAction);
22  TheAnalysisManager->SetOutputFilename(RootFileName);
23
24  // Visualization Manager
25  G4VisManager* TheVisManager = new G4VisExecutive;
26  TheVisManager->SetVerboseLevel(0);
27  TheVisManager->Initialize();
28
29  // Add user-defined action classes to the run manager
30  TheRunManager->SetUserAction(ThePrimaryGeneratorAction);
31  TheRunManager->SetUserAction(new QrocoRunAction(TheAnalysisManager));
32  TheRunManager->SetUserAction(new QrocoEventAction(TheAnalysisManager));
```

[8] As long as they are connected to the same server where the files are located.

```
33
34   // Geometry manager
35   G4UImanager* UImanager = G4UImanager::GetUIpointer();
```

This block of code imports the detector construction, the physics list, which encompasses the physical logic of the code, the radioactive source and the generator action. Furthermore it contains the main visualization manager. The PhysicsList and the DetectorConstruction are two of the main mandatory user classes of the Geant4 toolkit.
To avoid recompiling the entire code base with each minor modification, we use macro files. A macro is a code segment in a program that is replaced by the value of the macro. The creation of these macros will be discussed in a later section. To be able to read macros, one has to implemented following lines:

```
1  if(RunMacro)
2   {
3     Command = "/control/execute " + MacroName;
4     UImanager->ApplyCommand(Command);
5   }
```

The full file can be found in chapter 8.3.2.

4.5 Implementation of CADMesh

In its base form, Geant4 offers limited support for direct loading in pre-existing CAD models as a geometry. However, there can be instances where the geometry is too complex to be defined by basic solid objects. CADMesh is a package for Geant4 that allows the user to load mesh files directly into Geant4. Currently it supports .STL (stereolithography), .OBJ (object file) and .PLY (Polygon file format). Importing CADMesh into the project is quite straightforward and requires only a few lines of code:

```
1  #include "CADMesh.hh"
2  ....
3  auto mesh = CADMesh::TessellatedMesh::FromPLY("mesh.ply");
4  G4VSolid* solid = mesh->GetSolid();
```

where the import line can be replaced by

```
1  auto mesh = CADMesh::TessellatedMesh::FromSTL("mesh.stl");
```

or

```
1  auto mesh = CADMesh::TessellatedMesh::FromOBJ("mesh.obj");
```

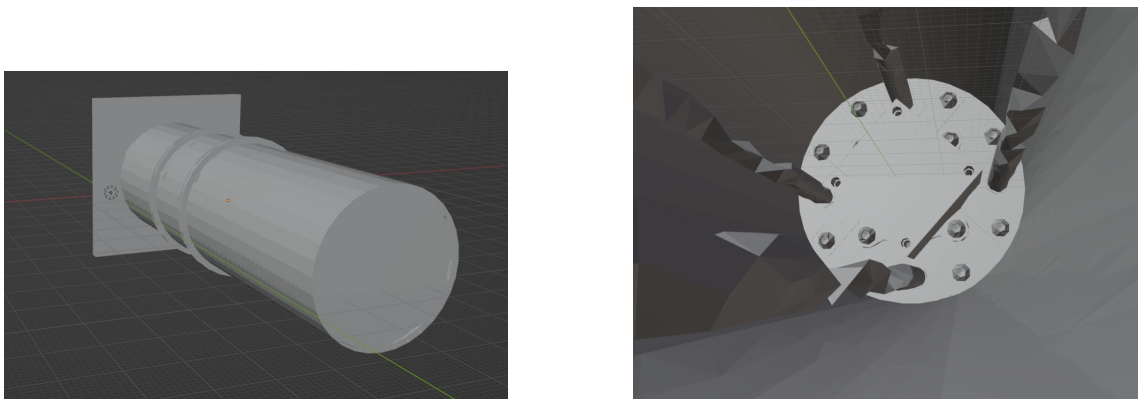
for the respective file format. CADMesh was mainly developed by Christopher Poole.[82]

4.6 Creation of the mesh model of the Cryostat

The centerpiece of the simulation is the Cryostat itself. In order to represent it as accurately as possible, we use a CAD model. To create this model, we used Blender[9], a free and open source 3D computer graphics software tool widely used in animation, video games development and 3D modelling. It can be downloaded from the official website and offers a wide variety of features. A key advantage of Blender is its ability to create 3D models freely and export them in nearly any required file format, such as .STL, .PLY and even .PDF.
With the detailed dimensions and measurements of the Cryostat available from the official manual provided by Oxford Instruments, we were able to model the Cryostat as realistic as possible. In simple terms, the Cryostat comprises the following elements:

- Three distinct layers of outer shells
- Small tubes containing the electronics
- One smaller shell that houses the holder as a whole
- The holder itself

The initial step involved modelling the entire Cryostat as a singular piece. The results of this process can be seen in figure 30.

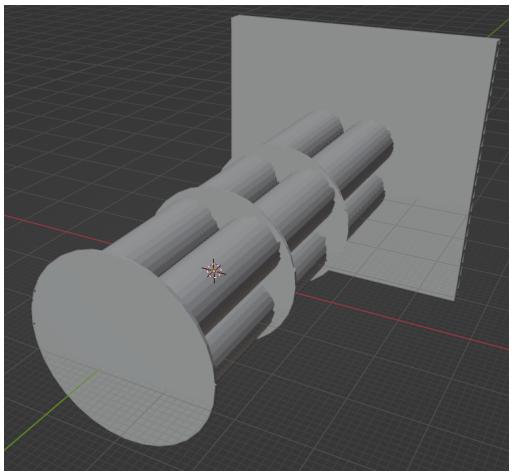


(a) From the outside (b) From the inside, viewed from top on the holder

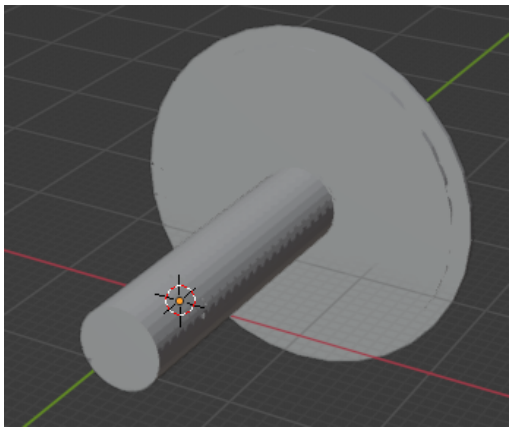
Figure 30: Multiple angles of the Cryostat constructed in Blender

As written more in detail in chapter 6.2.1, this implementation was sadly not successful. Our initial attempt at modelling the Cryostat as a single entity led us to encounter an error about stuck tracks. To get rid of this issue, we modified our approach and segmented the Cryostat into multiple models. Specifically, we divided the Cryostat in two distinct pieces. The first one consists of the outer shells, the tubes for the electronics and the smaller shell. The second model contains the holder, which was constructed from standard solid objects provided directly by Geant4. Each component of the Cryostat was individually modelled and then assembled. As a consequence, two new models have to be created:

[9]Blender 3.6.0 ©GNU GPL 2.0



(a) Tubes containing the electronics



(b) Shell containing the holder, will be placed inside Cryostat

Figure 31: New models created for the separate implementation

Following this modelling process, the only remaining step was to import the piece into Geant4 using CADMesh. This will be done as described in chapter 4.5:

```
auto mesh = CADMesh::TessellatedMesh::FromPLY("./SingleComponents/PLY/ShellsTubesGrob.PLY");
G4VSolid *Cyl_solid = mesh->GetSolid();
auto Cyl_logical = new G4LogicalVolume(Cyl_solid, cryo_mat, "Cyl_logical", 0, 0, 0);
new G4PVPlacement(rotation, G4ThreeVector(-1*m,2.2*m,0), Cyl_logical, "Cyl_physical",
  LabLogicalVolume, false, 0);
```

As our Cryostat is "filled" with vacuum, we need to explicitly define it. Fortunately, vacuum is considered a standard entity and can thus be directly imported from Geant4. To define vacuum in Geant4, you would use the NIST Material Manager to call up the predefined vacuum material, as shown in the following code snippet:

```
G4NistManager* nist = G4NistManager::Instance();
G4Material* vacuum_mat = nist->FindOrBuildMaterial("G4_Galactic");
```

Here, "G4_Galactic" is the predefined vacuum material in Geant4. It's worth noting that while vacuum is usually considered to be completely devoid of matter, in practical terms (such as in this simulation), it's often considered as a very low-density gas. The "G4_Galactic" vacuum is defined to have a density of $10^{-25} g/cm^3$, making it suitable for simulating the conditions inside the Cryostat. With the vacuum material defined, the next step is to create a new volume which matches the dimensions of the Cryostat but is entirely contained within it. This volume will represent the vacuum-filled interior of the Cryostat. Defining this volume involves defining its shape, size, and material properties in the Geant4 environment.

```
G4VSolid* VacBox = new G4Tubs("VacBox", 0.0 * m, 0.134 * m, 0.170 * m, 0.0 * deg, 360.0 * deg);
G4LogicalVolume* VacLogicalVolume = new G4LogicalVolume(VacBox, vacuum_mat, "VacVolume", 0, 0, 0);
new G4PVPlacement(rotation, G4ThreeVector(-0.156*m, 0.355*m,1.17*m), VacLogicalVolume,
  "Vac",Cyl_logical, false, 0,true );
```

As a final piece, the holder itself has to be designed. As previously stated, we use simple geometries for the construction of the holder. It will be composed of a cube, which is defined as a G4Box, and a platform, which is defined as a G4Tubs, which is essentially represented by a cylinder:

```
//Box

G4VSolid* Box_solid = new G4Box("Box", 0.012 * m, 0.012 * m, 0.009 * m);
G4LogicalVolume* Box_logical = new G4LogicalVolume(Box_solid, holder_mat, "Box_logical");
new G4PVPlacement(0, G4ThreeVector(0,0,0), Box_logical, "Box_physical",VacLogicalVolume, false,
  0,true );

//Platform

G4VSolid* Platform_solid = new G4Tubs("Platform", 0.0 * m, 0.0275 * m, 0.002 * m, 0.0 * deg,
  360.0 * deg);
G4LogicalVolume* Platform_logical = new G4LogicalVolume(Platform_solid, holder_mat, "Platform_logical");
new G4PVPlacement(0, G4ThreeVector(0,0, 0.0115 * m), Platform_logical, "Platform_physical",
  VacLogicalVolume, false, 0,true );
```

With the whole Cryostat constructed, one can check with the help of ROOT, if the simulation was successful. Up to this point, the geometry is implemented in code only, it may be very difficult to spot mistakes, since the code does not offer a graphic interface. For this purpose the visualization with geantinos is very useful. Geantinos are a virtual particle species invented by Geant4. They emerge evenly distributed in a defined physical volume and then travel in a straight line to a random direction without any interaction. By running the simulation, plotting all the points of origin of all geantinos, selecting all wanted volumes to be visualized and checking the TBrowser, one can see the following plot:

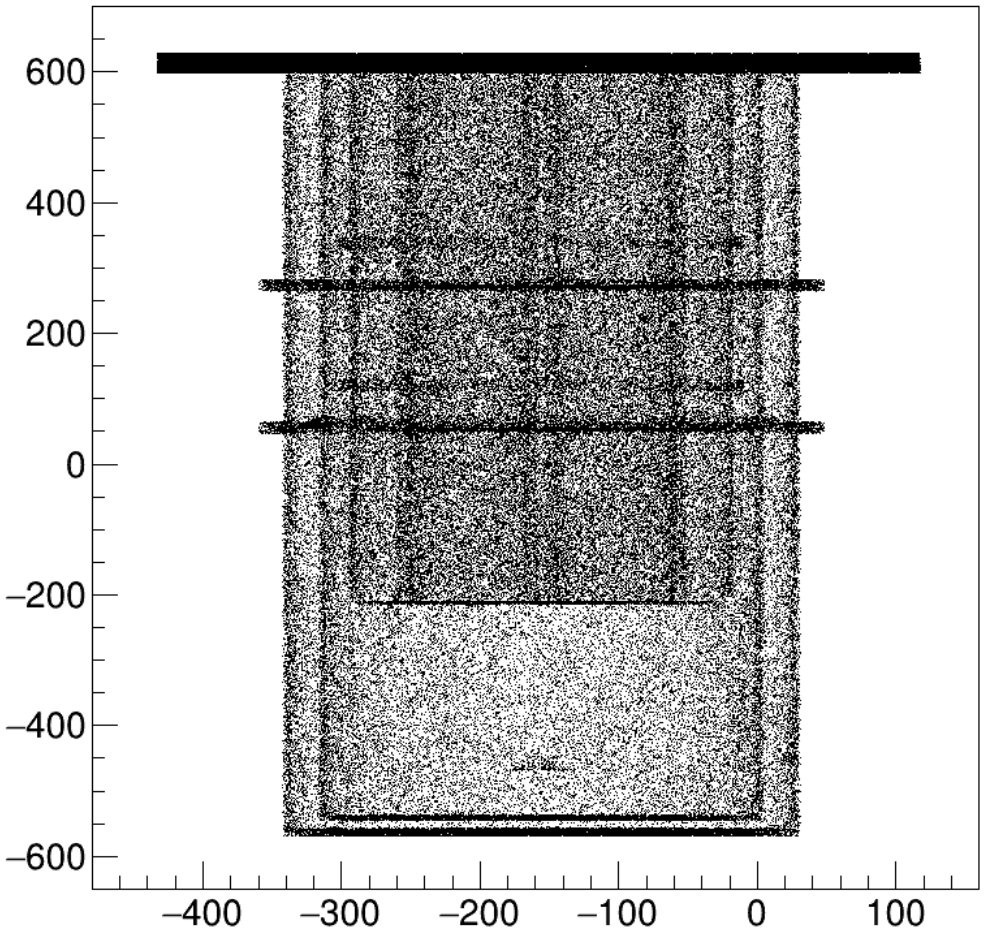


Figure 32: Running the simulation only with the Cryostat. The axes are representing the coordinates inside the laboratory. the x- axis is the x coordinate, while the y- axis shows the z- coordinate. One can see the holder at around (-150,-500). It gets hit by the geantinos.

4.7 Construction of the laboratory

Similar to the holder mentioned in the previous chapter, we will design the rest of the laboratory based on simple solids. The lab consists out of the following few items:

- A computer located next to the Cryostat. Plastic will be used as the material.
- A second Cryostat is situated on the other side of the cryostat, made from stainless steel.
- Two different PPMS system, also made from stainless steel.
- Some basic furniture: A table, a wall and a closet made from wood.

The remaining objects in the lab will be omitted, as they are either too small or too far away to have any significant impact on our simulation. For illustrative purpose, the creation of one object is detailed here:

```
1 //PPMS Physics
2
3 G4VSolid* PPMSP_solid = new G4Tubs("Physics", 0.0 * m, 0.35*m, 1.11*m, 0.0 * deg,
4    360.0 * deg);
5 G4LogicalVolume* PPMSP_logical = new G4LogicalVolume(PPMSP_solid, Steel_mat,
6    "PPMSP_logical");
```

```
new G4PVPlacement(0, G4ThreeVector(-3.49*m, -3.01*m,-0.88*m), PPMSP_logical, "PPMSP_physical",
   LabLogicalVolume, false, 0,true );
```

Up to this point, this object only exists physically in the simulation, but it is not visible. To change that, one has to assign a color. The color code is given by the RGB code of the corresponding color:

```
G4VisAttributes* PPMSPVisAtt = new G4VisAttributes(G4Colour(1,1,0)); //yellow

PPMSP_logical->SetVisAttributes(PPMSPVisAtt);
```

To view the rest of the code, please refer to chapter 8.3.3. Since we are not solely using standard materials, we have to define our own materials, as for example PVC or wood. This challenge is addressed in *QrocoDetectorMaterial.icc*, which includes all the various non- standard materials.
To create a material, one must assign a specific density and a number of components. For instance, wood consists of Carbon, Hydrogen and Oxygen:

```
G4Material* Wood = new G4Material(name="Wood", density=0.9*g/cm3, ncomponents=3);
Wood->AddElement(C, 2);
Wood->AddElement(H, 4);
Wood->AddElement(O, 1);
```

where the ratio for the different elements must be provided.[10]The rest of the material can be found in chapter 8.3.4. For common materials or elements, one can simply use the NistManager of Geant4, which contains most natural elements and some fundamental materials. To load a particular element, one can use the following line of code:

```
pNistManager->FindOrBuildMaterial("G4_Co");
```

This will load Cobalt, as an example.
As a final crucial element, we need to understand the logic behind the LogicalVolume construct in Geant4. Every construction must be contained within a specific LogicalVolume to ensure the simulation correctly compiles it. This concept acts as a spatial and property container for objects within the Geant4 simulation environment.

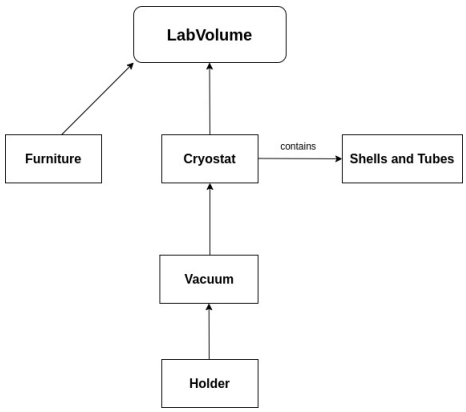


Figure 33: Schematic representation of the different logical volumes

To interpret figure 33, one has to read the arrows as "is contained within". The world volume represents the entire lab, with the Cryostat and the furniture is placed within the world volume. The vacuum,

[10]This has been simplified. It depends on the type of wood, but in general one can say the chemical formula for wood is $C_{18}H_{30}O_{14}$.

the shells and the tubes are contained in the Cryostat, while the holder itself is inside the vacuum. If all the objects up to this point have compiled correctly, one can start the visual representation. The lab will be represented as shown in the figure 34, 35 and 36. The main Cryostat is colored white, the second Cryostat grey, the Computer to the left is blue, the wooden pieces in the lab are brown and the two PPMS systems are yellow.

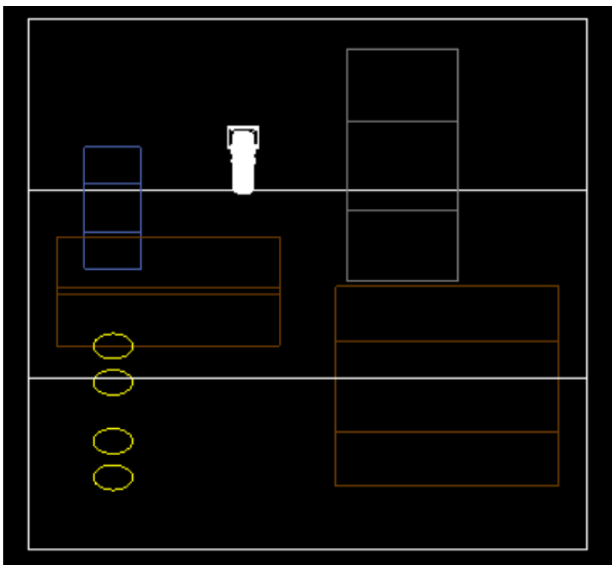


Figure 34: Graphical representation of the laboratory, viewed from the front

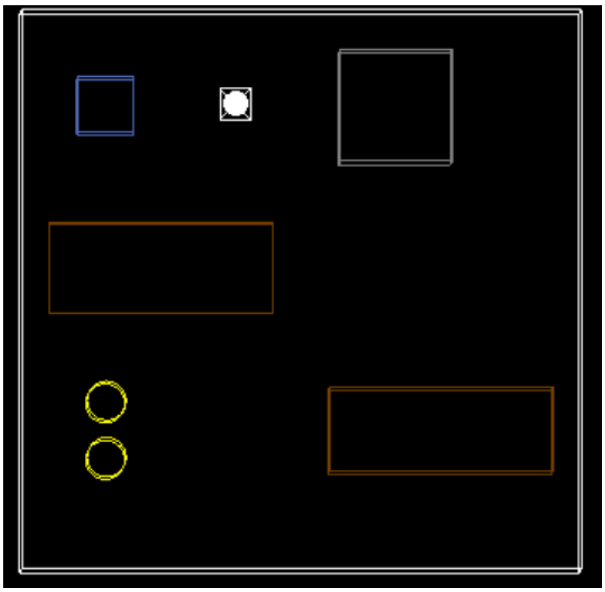


Figure 35: Graphical representation of the laboratory, viewed from the top

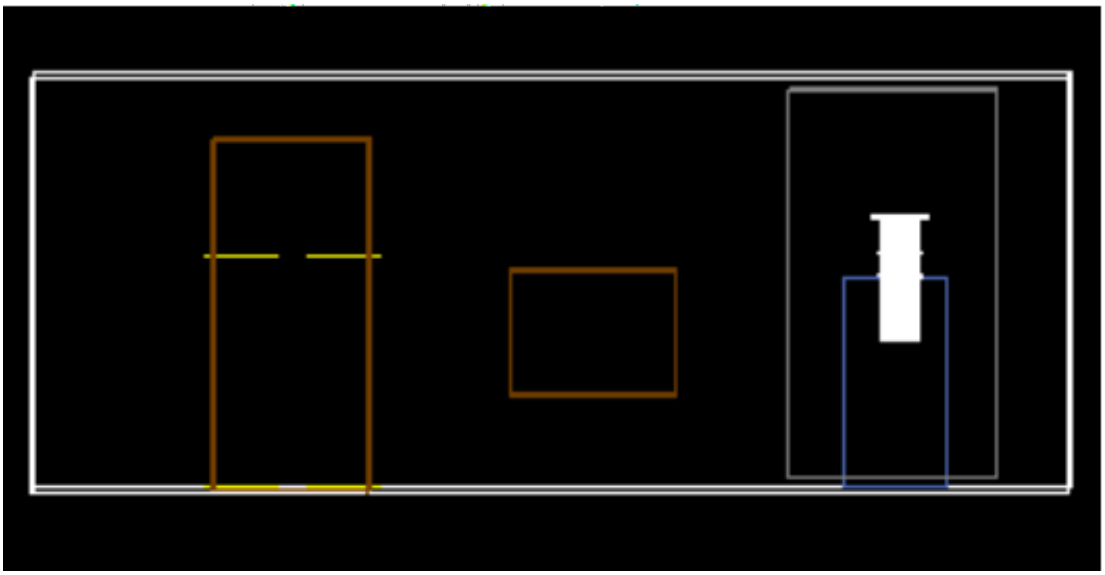


Figure 36: Graphical representation of the laboratory, viewed from the side

Figure 37 represents a close up from the pictures shown in 34, 35 and 36. It shows the mesh format of the Cryostat and the holder is shown in brown.

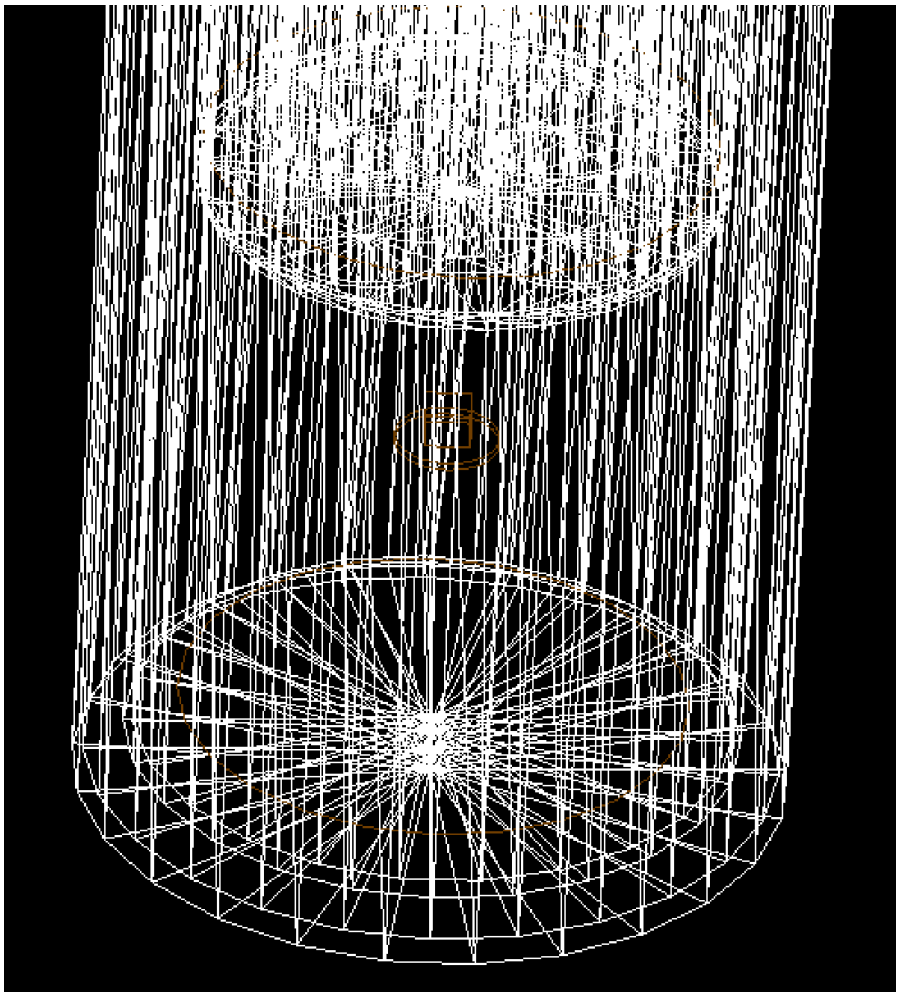


Figure 37: Close Up of the Cryostat. Again, one can clearly see the Holder consisting of the platform and the box.

4.8 Creation of the macro file

To run the simulation with a visual representation and to conveniently modify the setup of our particle, one introduces a macro file. This macro file contains the type of particle we are using, the structure of the "gun" itself, the position of its origin and the volumes it will be confined to.
Each volume that the particle will traverse needs to be defined separately in the macro file, following a structure similar to the following example:

```
/qroco/gun/confine Cyl*
```

For every other volume, the command follows an analogous structure.
In this simulation we are using Caesium-137. Caesium is the 55th element in the periodic table of elements and has only one stable isotope, 133. The β- decay of Cs-137 to Ba-137m results in gamma radiation as the Ba-137m relaxes to its ground state Ba-137, while the emitted photons have an energy of $0.6617 MeV$. Those two isotopes are the most common isotopes in the Chernobyl disaster from 1986.[83] Since we want to measure the impact of those emitted gammas, we simulate gamma particles with the corresponding energy:

```
/qroco/gun/particle gamma

/qroco/gun/energy 662 keV
```

The final step involves defining the position and the type of the source. Since we want to measure the impact of the ions onto our holder, we place it right below to the Cryostat. Given that the position of the Cryostat is fixed, the positioning of the source is straightforward:

```
/qroco/gun/position -1.156 3.37 -0.380 m
```

Given our aim to simulate a point source that emits directly upwards into the Cryostat, we can implement the following line of code to realize this:

```
/qroco/gun/direction 0 0 1
```

The remaining content of the macro file can be found in appendix 8.3.7.
As a final step, it's necessary to compile the entire package as an executable. This can be accomplished through cmake using the following command:

```
cmake -B build && cmake --build build
```

This command generates a build folder, where cmake places the executables and other compilation objects. After the compilation is finished, the executable named QROCODILE is generated within the build directory.
To execute the code, one uses the following line in the main directory:

```
./build/bin/QROCODILE_G4
```

To minimize the additional work, one can implement parsing switches into the main.cc file. The most important switches include:

- -f <exeMacro>: Executes the simulation using a macro file
- -n <number>: Indicates the number of particles to be simulated
- -v <engine>: Enables visualization. Supported engines are *opengl* and *qt*.

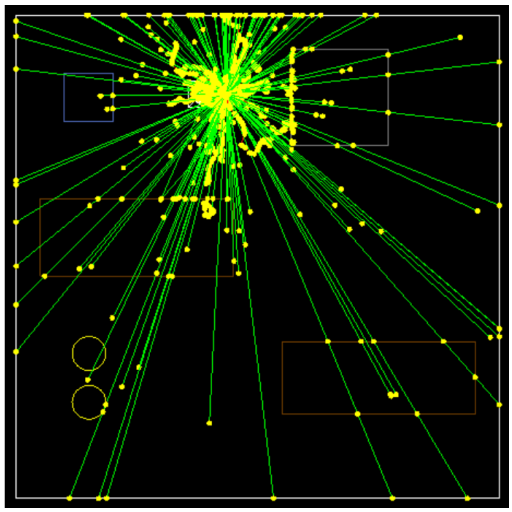
To validate the simulation prior to conducting an analysis, one can use the following example command:

```
./build/bin/QROCODILE_G4 -f macros/run_gamma.mac -n 200 -v qt
```

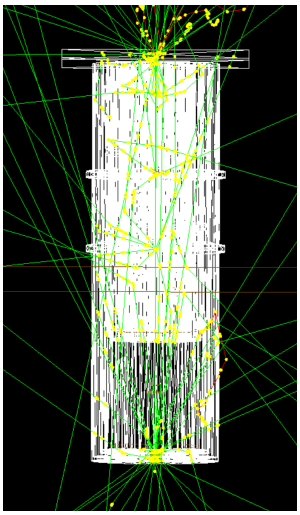
In principle, *-v qt* can be replaced with a command in the macro to always include the visualization feature:

```
/vis/open Qt
/vis/drawVolume
```

This command will run the simulation with 200 gammas of Cs-137 with an energy of $662keV$. If it works properly, it will look like figure 38.



(A) Impact of the decay on the whole lab



(B) Close- Up of the Cryostat and the decay

Figure 38: Visualization of the decay of Cs-137 in the lab

4.9 Background of the analysis

To be able to analyze the number of particles reaching the center of our holder, one has to define a sensitive area, which will be used as the detector for Geant4. This detector tracks the ions, which reach the center and therefore passing all the different layers of protection, such as the shells and the holder itself. To be able to read all the necessary parameters such as energy deposition, particle types, parent particles, etc. one has to set up an *Event*. This can be accomplished within the *QrocoEventAction.cc*:

```
void
QrocoEventAction::BeginOfEventAction(const G4Event *TheEvent)
{
  if(TheEvent->GetEventID() % 1000 == 0)
    G4cerr << "------ Begin event " << TheEvent->GetEventID()<< "------" << G4endl;

  if(AnalysisManager)
    AnalysisManager->BeginOfEvent(TheEvent);
}
```

Moreover, this implementation will provide a timestamp for every thousand events. The complete code can be reviewed in appendix 8.3.6. Now one has to implement all the needed parameters, which will be introduced in the form of vectors:

```
QrocoEventData::QrocoEventData()
{
  TrackId = new vector<int>;
  ParticleType = new vector<string>;
  ParentId = new vector<int>;
        X = new vector<float>;
        Y = new vector<float>;
        Z = new vector<float>;
        EnergyDeposited = new vector<float>;
  Time = new vector<float>;
        KineticEnergy = new vector<float>;

```

```
  PrimaryType = "";
  PrimaryDirX = 0.;
  PrimaryDirY = 0.;
  PrimaryDirZ = 0.;
  PrimaryE = 0.;
}
```

During the simulation, all the data need to be stored in a .ROOT file, which is needed to analyze the data:

```
QrocoAnalysisManager::QrocoAnalysisManager(QrocoPrimaryGeneratorAction* ThePrimaryGeneratorAction)
{
  OutputFileName = "output.root";
  PrimaryGeneratorAction = ThePrimaryGeneratorAction;
  EventData = new QrocoEventData();
}
```

To separate all different parameters from the events into different arrays, EventBranches and Trees are introduced. These crucial constructs in ROOT offer an organized hierarchy for data storage, allowing efficient access and analysis:

```
void QrocoAnalysisManager::BeginOfRun(const G4Run* )
{
  OutputFile = new TFile(OutputFileName.c_str(), "RECREATE");

  G4cout <<"Output file opened."<<G4endl;
  EventTree = new TTree("events", "Simulation data");

  EventTree->Branch("trackId", "vector<int>", &EventData->TrackId);
  EventTree->Branch("type", "vector<string>", &EventData->ParticleType);
  EventTree->Branch("parentid", "vector<int>", &EventData->ParentId);
  EventTree->Branch("xp", "vector<float>", &EventData->X);
  EventTree->Branch("yp", "vector<float>", &EventData->Y);
  EventTree->Branch("zp", "vector<float>", &EventData->Z);
  EventTree->Branch("ed", "vector<float>", &EventData->EnergyDeposited);
  EventTree->Branch("time", "vector<float>", &EventData->Time);
  EventTree->Branch("ekin", "vector<float>", &EventData->KineticEnergy);

  EventTree->Branch("type_pri", "string", &EventData->PrimaryType);
  EventTree->Branch("xPrim", &EventData->PrimaryX, "xPrim/F");
  EventTree->Branch("yPrim", &EventData->PrimaryY, "yPrim/F");
  EventTree->Branch("zPrim", &EventData->PrimaryZ, "zPrim/F");
  EventTree->Branch("ePrim",  &EventData->PrimaryE, "ePrim/F");
  EventTree->Branch("vxPrim", &EventData->PrimaryDirX, "vxPrim/F");
  EventTree->Branch("vyPrim", &EventData->PrimaryDirY, "vyPrim/F");
  EventTree->Branch("vzPrim", &EventData->PrimaryDirZ, "vzPrim/F");

  OutputFile->cd();
}
```

Once the run has ended, one has to store all the data in the output file:

```
void QrocoAnalysisManager::EndOfEvent(const G4Event* )
{

  // Info of the primaries
  EventData->PrimaryType = PrimaryGeneratorAction->GetTypeOfPrimary();
  EventData->PrimaryX = PrimaryGeneratorAction->GetPositionOfPrimary().x();
  EventData->PrimaryY = PrimaryGeneratorAction->GetPositionOfPrimary().y();
  EventData->PrimaryZ = PrimaryGeneratorAction->GetPositionOfPrimary().z();
  EventData->PrimaryDirX = PrimaryGeneratorAction->GetDirectionOfPrimary().x();
  EventData->PrimaryDirY = PrimaryGeneratorAction->GetDirectionOfPrimary().y();
  EventData->PrimaryDirZ = PrimaryGeneratorAction->GetDirectionOfPrimary().z();
  EventData->PrimaryE = PrimaryGeneratorAction->GetEnergyOfPrimary() / keV;

  EventTree->Fill();
  EventData->Clear();
  OutputFile->cd();
}
```

Once the .ROOT file is successfully written, one can start with the analysis in ROOT, as already written in chapter 4.3. There are two possibilities to run an analysis in ROOT. The first is the input via terminal, where the code is inserted line per line. This works well for fast testing, but in our case, a macro file for C++ is used, which can be executed as a whole.
The finalized "*output.root*" is stored in ROOT and has to be loaded into the macro:

```
const char* fileName = "output.root";

// Open the ROOT file
TFile* file = TFile::Open(fileName);
    if (!file || file->IsZombie()) {
        std::cerr << "Error opening file: " << fileName << std::endl;
        return;
    }
```

The conditional if- function simply checks if the output file has been compiled correctly and is not empty. The information within a .ROOT file is stored in trees, which in turn contains branches. A tree, referred to as *TTree*, behaves much like an array of a data structure that resides on storage, except for one entry. That unique entry is accessible in memory, therefore permitting the loading of any tree entry. To load a tree, one uses the *TTree* function:

```
// Access the TTree in the file
    TTree* tree = dynamic_cast<TTree*>(file->Get("events"));
    if (!tree) {
        std::cerr << "Error accessing TTree in file: " << fileName << std::endl;
        file->Close();
        return;
    }
```

Similar as before, the if function checks if the tree is not empty. This is implemented for preventing memory leaking in the case of big data. If the tree is empty, it prevents unnecessary memory allocation and potential crashes that could occur due to attempts to access or manipulate nonexistent data.
A tree consists of a list of independent columns, called branches, called "*TBranch*". When reading a tree, you can select which subset of branches should be read. In this case, you have to tell the tree

about the addresses of these subsets by invoking the method *TTree::SetBranchAddress()*. Our tree contains multiple subsets, the most important are:

- xp, yp, zp: Position around the box in global coordinates
- ed: Energy deposition of the incoming particles in our box
- ekin: The kinetic energy of the particles

To start, one has to define a vector, which will contain the final result. But at the beginning, it will be an empty pointer, serving as a placeholder for future data. This technique provides a flexible and efficient means to handle data sets of unknown size, as the vector will dynamically resize as data is added:

```
std::vector<int> *posx = nullptr;
```

With this vector defined, one has to set the branch address inside the tree, which will be connected to the pointer of the corresponding subset:

```
tree->SetBranchAddress("xp", &posx);
```

In this code snippet, the '&' operator is used to pass the adress of '&pos'.
To check how many of the initial particles hit the holder, one has to find the number of unique particles. In theory, it is possible that a particle hits the holder multiple times, due to reflections or a certain incident impact angle. Such occurrences could lead to inaccurate counting, where one particle could be erroneously counted as two. This will be done with help of the *trackId* value assigned to each of the particles. This parameter will give each particle its unique track. Therefore, if two hits share the same trackId in the past, we count them as one particle. To implement this logic, one has to define four variables first:

```
std::vector<int> uniqueTracks;
int counter = 0;
int currentTrack = -1;
float Totalenergydeposited = 0.0
```

The first variable *uniqueTracks* is the whole track of each particle which has a unique track until now. *counter* simply contains the number of unique tracks and *currentTrack* is the track of the particle that is currently read. *Totalenergydeposited* gives the energy, which is deposited on the holder. Subsequently, we have to loop over each entry in the tree:

```
for (Long64_t i = 0; i < tree->GetEntries(); ++i) {
        tree->GetEntry(i);
        float PartialEnergy = 0.0;
```

Inside this loop, one has to loop over all the *trackId* values:

```
for (int k = 0; k < trackId->size(); ++k) {
            PartialEnergy += ener->at(k);
            Ener->Fill(PartialEnergy);
        }
```

Further in this loop, one has to check if the *uniqueTrack* for the current particle already appeared in a previous particle:

```
if (std::find(uniqueTracks.begin(), uniqueTracks.end(), currentTrack) != uniqueTracks.end()) {
        continue;  // Track already encountered, skip to the next one
            }
```

If the track is new and never appeared before, one can add it to the *currentTrack*:

```
uniqueTracks.push_back(currentTrack);
```

Since we want to visualize the results, one has to put the unique tracks (and therefore the unique particles) into a histogram:

```
TH1F *PosX = new TH1F("Values", "Position X", 40, -1140, -1180);
(...)
PosX->Fill(posx->at(j));
(...)
TCanvas *c1 = new TCanvas();
PosX->Draw();
```

In this code, TH1F is a class provided by ROOT to create a 1-dimensional histogram of integers. The constructor for TH1F takes several arguments. The first is the name of the histogram, the second is the title which will be displayed above the histogram when it is drawn, and the next three parameters define the number of bins, the lower edge and the upper edge of the histogram respectively. After that, we loop over each entry in the tree, and for each unique track found, we fill the histogram with the current size of 'uniqueTracks' vector, representing the count of unique particles till that point. At the end of the analysis, one has to count all the unique tracks and then we can print it:

```
counter += uniqueTracks.size();
uniqueTracks.clear();
(...)
Totalenergydeposited += PartialEnergy;
)...)
std::cout << "The total number of unique particles is: " << counter << std::endl;

std::cout << "The total energy deposited is: " << Totalenergydeposited <<  "keV" << std::endl;
```

For additional information, one can display the time needed to run the analysis.[11] This can be done easily with the *Chrono* inclusion. At the beginning of the analysis, this one line has to be inserted:

```
auto start = high_resolution_clock::now();
```

The 'auto' keyword automatically deduces the type of the variable.
Similarly, at the end of the code:

```
auto stop = high_resolution_clock::now();
```

The resulting time is simply the difference between those two values:

```
auto duration = duration_cast<milliseconds>(stop - start);
```

[11] The time is not really necessary for the simulation, it is just for fun.

For a reasonable simulation, it is essential to choose an appropriate simulation time. As discussed in chapter 4.8, we are simulating gamma particles emitted by a Caesium-137 source with an activity of $1.9kBq = 1900s^{-1}$, which means 1900 decays per second. With the flexibility of our macro file, we can effortlessly adjust the number of simulated particles. As an example, the first run was performed with 100'000 particles, corresponding to around 1 minute of livetime. This is a rather brief simulation, and the whole simulation needed 6 minutes to execute. Hence, to have a meaningful simulation, one should aim for 24 hours of livetime. Achieving 24 hours of livetime would demand approximately 140'000'000 simulated particles, which corresponds to around 164'100'000 Cs-137 ions, assuming 85.1% of Cs-137 decays have a gamma emission.[84] This would need around 140 hours of computational time. As one can see, to have a huge amount of particles and therefore a meaningful amount of hits, one has run a rather long simulation:

Simulation		
Particles	Computational time	livetime
100'000	6 min	1 min
500'000	30 min	5 min
2'000'000	2 h	20 min
10'000'000	10 h	1 h 40 min
25'000'000	25 h	4 h 10 min
140'000'000	140 h	24 h

Table 4: Comparison of different run times for the simulation

5 Results

5.1 Calibration of laser source

One of the first task was to calibrate the power spectrum of SuperK Extreme Laser Source. The setup and connection process is described in the manual, which is attached in appendix 8.6. This specific laser source was employed as a means to confirm the integrity of the connection between the source and the Cryostat's output, and to ensure the transmission of light was functioning properly. For the final experiment, we changed to the setup illustrated in chapter 5.4, figure 27. To check if the connection is fine, one has to use visible light, which is in the range of $400nm$ to $750nm$. However, for experimentation within the lower energy range, the light photons need to reside within the NIR range. As discussed in chapter 2.4.3, photons within this spectrum have a energy in the range of $1.24meV - 1.7eV$[85].

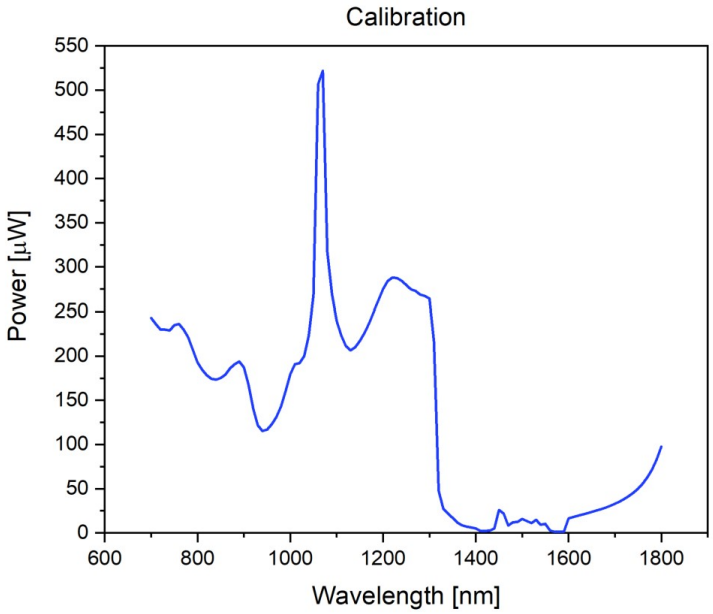


Figure 39: Power spectrum of the SuperK Extreme laser source

Figure 39 reveals that this laser source has a power peak at $1070nm$ which corresponds to an output power of $521\mu W$. Since this setup was too complicated and too inconsistent to be used, due to multiple components such as a monochromator, we switched to the *Fabry-Perot Benchtop Laser Source* from ThorLabs.

5.2 Measurement in PPMS

Chapter 3.3 previously provided a detailed analysis of the resistance- temperature relationship for sample 8.This section extends that examination to additional smples from the first batch, each characterized by varying R_S, T_c and RRR. We wanted to check, if these samples had suitable parameters for our purpose. As outlined in chapter 3.1.2, L_k is depending on T_c and R_S and therefore the measurements in the PPMS are crucial for further experiments regarding the Cryostat. Figure 40 presents the results of these measurement in the PPMS for the functioning samples. All the important parameters are included. By comprehensively studying these properties, we can better tailor our experimental conditions to enhance the potential for meaningful and reliable outcomes.

Sample number	R_S [Ω]	T_c[K]	$L_k[pH]$
5	201.9	5.49	36.61
6	733.2	5.36	136.79
7	273.1	5.82	46.92
8	261.5	8.02	32.61
9	1224.8	3.02	405.56
10	2745.7	-	-
11	244.9	-	-

Table 5: Comparison of L_k for different samples

Table 5 presents additional samples, specifically sample 10 and 11, which have not yet been discussed in depth. The corresponding plots for these samples are shown in figure 41. As one can see, these samples are non- functional for multiple reasons: The films are degraded, the bonds on the sample were not properly connected or the sample cannot transition to the superconducting state. Inspired by those insufficient results, the decision fell to use different samples, described in chapter 3.1.3 and in table 2. With the samples 1A-5A we continued the experiment in the Cryostat, which were hoped to yield more promising and reliable results.

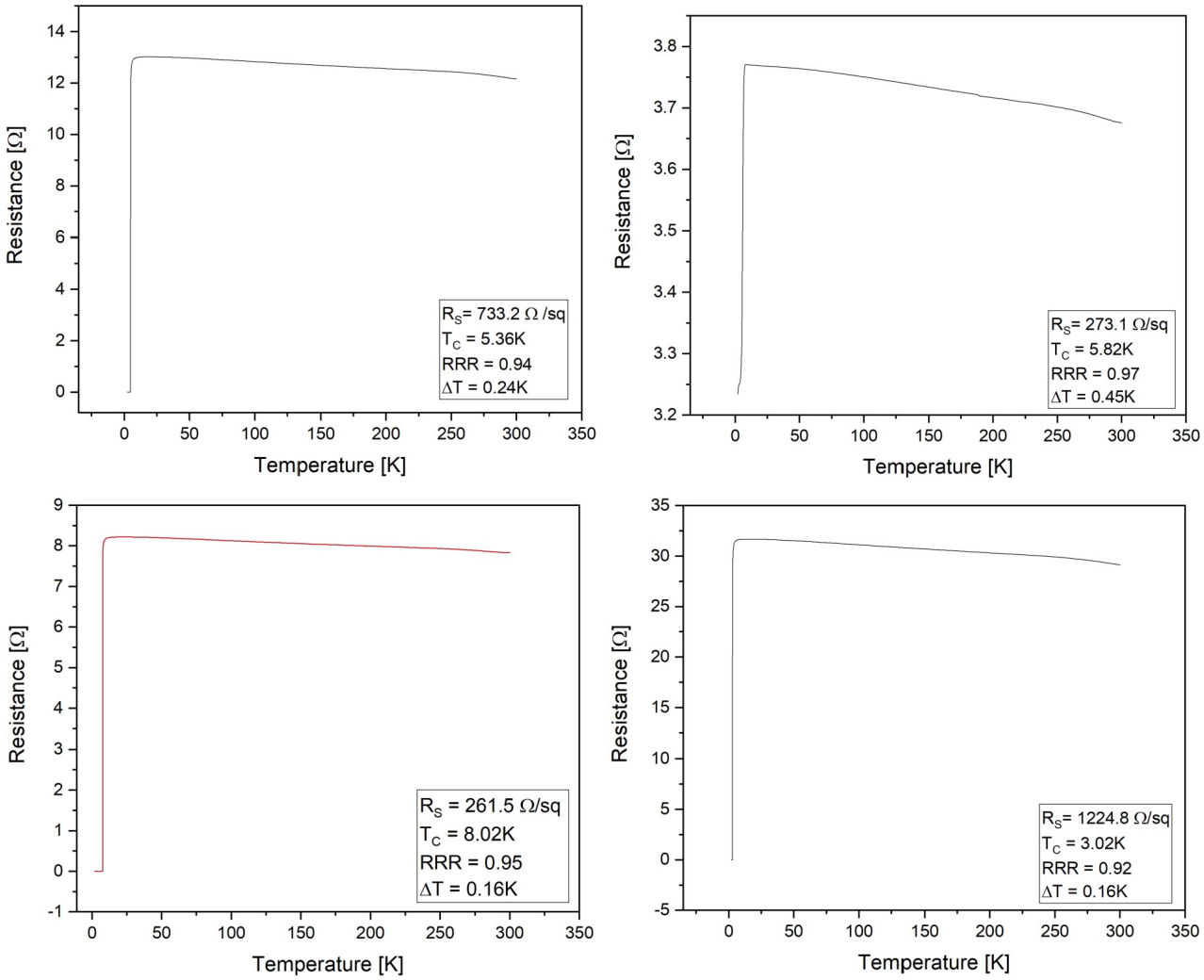


Figure 40: $R(T)$ for films 6-9 with a width of 4nm. Film 6 and 7 (upper two) are made of WSi, film 8 (bottom left) of NbN and film 9 (bottom right) of MoSi.

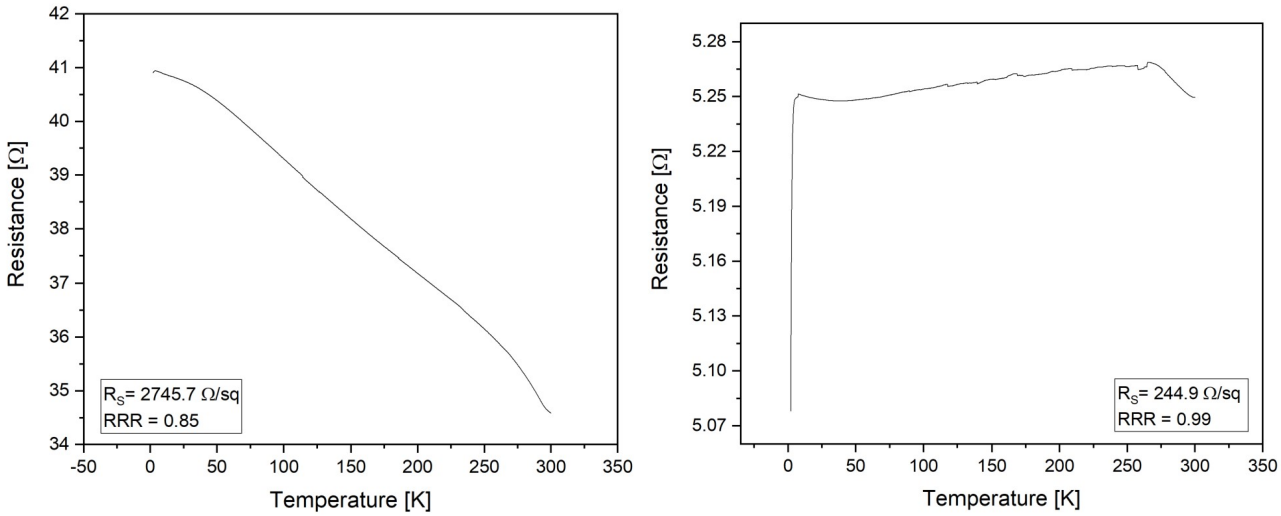


Figure 41: Degraded samples 10 and 11

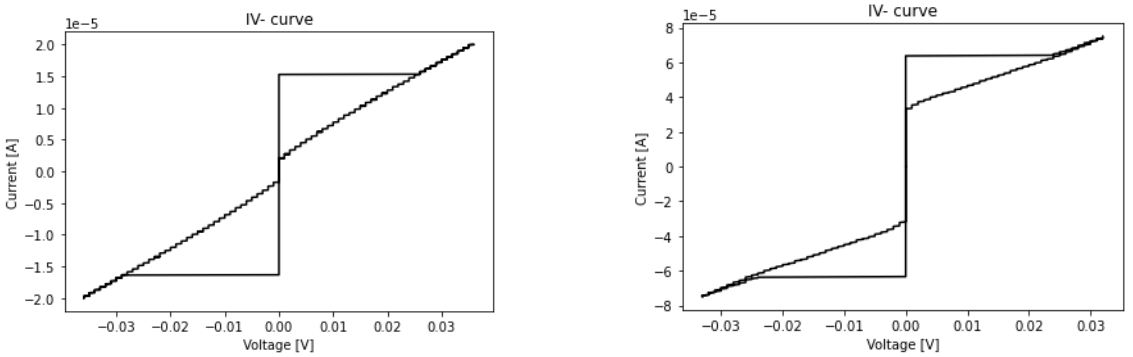
5.3 IV- curves

As explained in chapter 3.4, measurements of the IV curves for the samples as part of their characterization process have been conducted. The collected data, presented in plots, allow us to determine the parameters listed in table 6. One new parameter is introduced in this context, termed hysteresis. Hysteretic IV- curves ensure the switching mechanism of SNSPDs, as depicted in chapter 2.4.3. Hysteresis is typical in nanostructures of this scale and it significantly influences the behavior of our system. The hysteresis signifies the formation and evolution of a self-stabilizing hotspot in the superconducting wire, an essential aspect of the functioning of these detectors.[86, 87]

Width [μm]	$R_{300K}[M\Omega]$	Hysteresis [μA]	Temperature [mK]	I_c [μA]
0.14	11.82	-	-	-
0.5	8.12	-0.68	317	3.4
1	29.0	-1.0	234	6.9
1	5.76	-0.58	318	10
1	2.88	33.2	301	63
2	7.55	2.1	266	16
3	7.48	-	-	-
15	7.32	9.0	291	75

Table 6: Comparison of different samples for IV- curves

Furthermore, the table also includes information on the temperature of the Cryostat at the time of the measurement. This is a crucial factor, as the Cryostat's temperature can significantly impact the performance of the superconducting nanowire and, in turn, the measurements obtained. Thus, noting the Cryostat temperature during the measurements provides valuable context for interpreting the data and understanding the behavior of the samples under different conditions. The hysteresis, in this context, is the characteristic property that defines the transition point where the superconducting nanowire shifts back from a normal conducting state to its superconducting state. Numerically, this transition point is identified when the voltage (V) drops back to zero. In a hysteresis loop on an IV curve, this would be the point where the system returns to the superconducting phase after being in the resistive state, signifying the recovery of superconductivity. This is an essential feature of the SNSPDs, as it indicates the device's ability to reset itself for subsequent detections after the absorption of a photon has driven it into a resistive state.



(a) Sample 1: $I_c = 16.0\mu A$ (b) Sample 7: $I_c = 63.0\mu A$

Figure 42: IV- curves for both samples 1 and 7

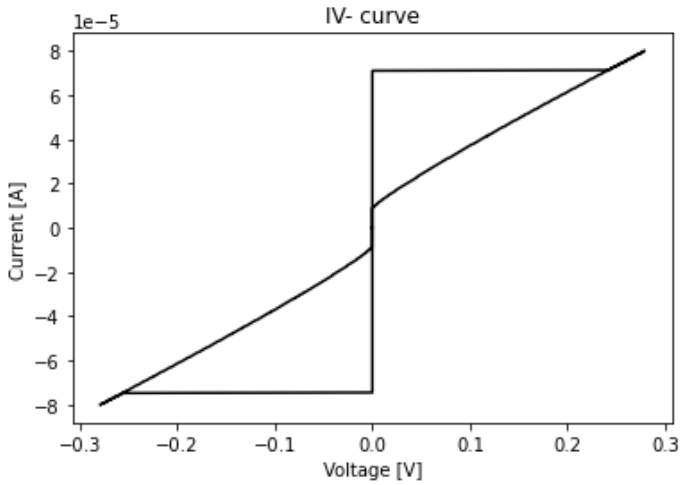


Figure 43: Sample 8: $I_c = 75.0\mu A$

During the testing phase, a total of 8 samples were measured, as presented in table 6.
Unfortunately, samples 5 and 6 were found to be degraded, thereby preventing the collection of IV data. Consequently, the determination of the critical current and hysteresis was not possible for these samples. Sample 5 did not switch until a high critical current of approximately $300\mu A$ was reached, which was beyond the range suitable for these types of measurements. Conversely, sample 6 transitioned too rapidly, essentially at $0mA$, making it impossible to conduct a meaningful measurement. In light of these findings, these two samples were deemed unsuitable for further analysis. Future studies may benefit from investigating the causes of such degradation and finding ways to prevent or minimize it to ensure more consistent and reliable data collection.

5.4 Photon count rate

This section shows one of the most promising approaches for measuring the PCR, aligning with the specific characterization we aimed for. The arrangement, the methodology and the prototype for a PCR measurement are presented in chapter 3.5. Figure 44 shows the PCR as a function of the bias current. The sample used for this measurement has a width of $1\mu m$ and the measurement was conducted at $100mK$ at the cold stage of the Cryostat and the wavelength was set to $1550nm$, as explained in chapter 2.6.

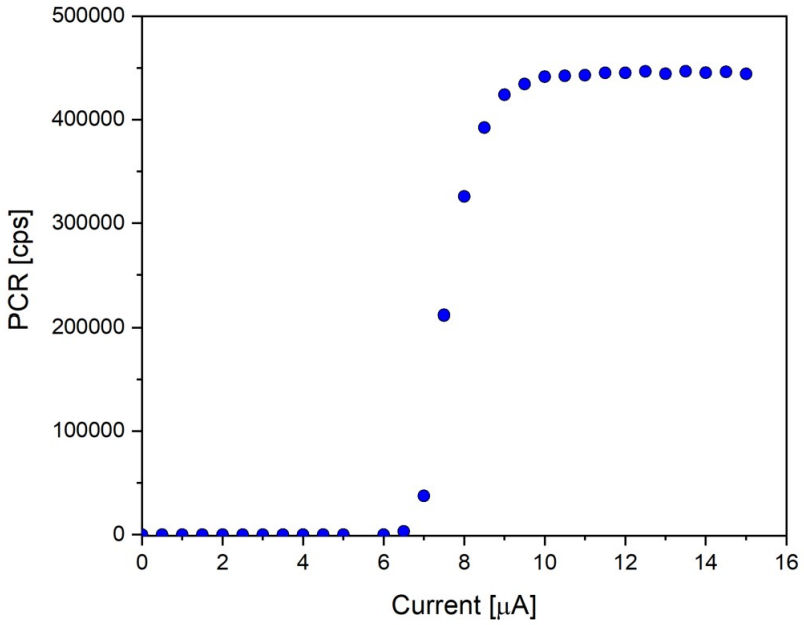


Figure 44: PCR with respect to the bias current. One can clearly see the plateau which corresponds to a quantum efficiency close to 100%. The constant plateau begins at a bias current of $\sim 10\mu A$.

Overall, a lot of experiments have been carried out using a lot of different samples and characterizations, and not every measurements was successful. One of those samples is presented in figure 45. This particular sample doesn't attain saturation, leading to a variable quantum efficiency. This inconsistency renders the sample suboptimal for our objectives.

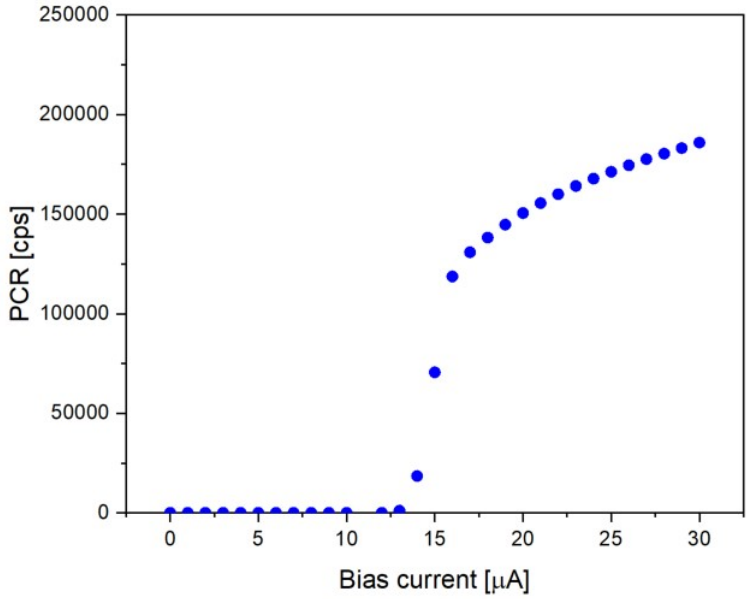


Figure 45: PCR measurement of a sample. The PCR does not reach a plateau.

Another approach was the implementation of a shunt into our setup. This resulted in an even less promising PCR result, as shown in figure 46. A further explanation of a shunt and the resulting challenges can be found in chapter 6.2.3.

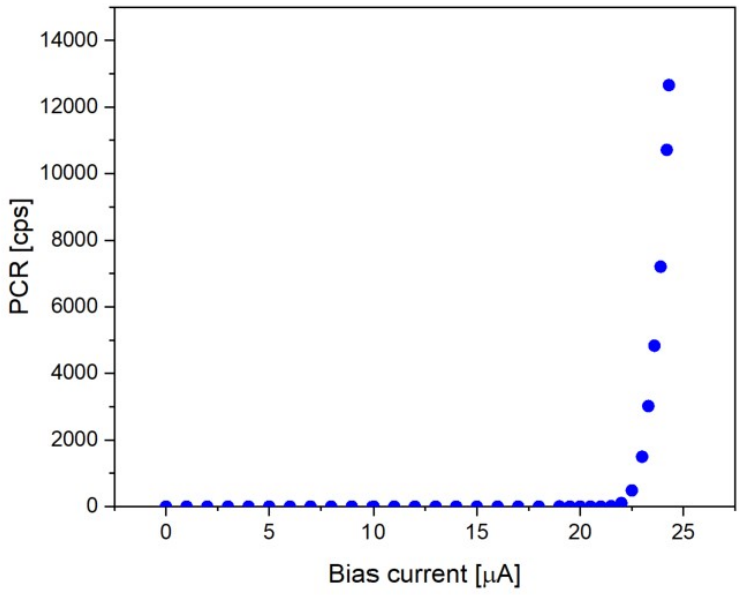


Figure 46: PCR measurement with included shunt. One can see that the PCR has a exponential behaviour.

5.5 Analysis of the Geant4 simulation

The simulation outlined in chapter 4.9 was conducted with six distinct quantities of particles, as already stated. Table 7 showcases the comparison among the different iterations of the simulation. For the shorter runs, specifically those with 100k, 500k and 2Mio particles, the results are appended in appendix 8.4. Table 7 is separated into five columns: "Simulation" gives the number of simulated particles, "Hits" describes the cumulative number of hits, even if a single particle hits multiple times, "Particles" is the number of unique particles that hit the sensitive area, "Ratio" is the percentage of unique particles with respect to the number of simulated particles and "Energy deposit" is the quantity of energy deposited in the sensitive area.[12]

Results				
Simulation	Hits	Particles	Ratio	Energy deposit [keV]
100'000	38	8	0.090 ‰	~ 0
500'000	39	39	0.078 ‰	~ 0
2'000'000	138	120	0.083 ‰	0.001
10'000'000	596	576	0.057 ‰	0.782
25'000'000	1649	1516	0.06 ‰	2.193
140'000'000	≈9230	≈8500	≈0.06 ‰	≈12.3

Table 7: Comparison of results; 140'000'000 particles are an estimation based on the linearity based on the previous results.

As can be observed, the total number of unique particles striking the sensitive area is extremely small in comparison to the simulated number. This could be attributed to the Cryostat itself, which is composed of aluminum and blocks (or deflect) a large number of particles, preventing them from reaching the holder. Despite this, the small number of particles deposits a rather large amount of energy onto our sensitive area.
Out of a total energy of $662keV$ per particle, 25'000'000 particles deposit $2.193keV$. Given that we are operating in the low-eV energy range, this deposited energy has a significant impact on our measurements, thereby emphasizing the importance of robust shielding for experiments involving SNSPDs.
As explained in chapter 4.9, the quantity of simulated particles ideally should have been considerably higher than what was achieved in this study. Nevertheless, a near-linear trend is discernible in the number of unique hits. When comparing the ratios across the various simulations, a slight decrease is observed. Consequently, in a simulation span of 24 hours, an estimated 8500 unique particles would strike the sensitive area depositing an energy of around $12keV$.

5.5.1 Simulation of 10'000'000 particles

The first reasonable amount of simulated particles are 10'000'000 particles, resulting in 576 unique particles. The initial analysis will be performed based on the position of hits on the sensitive area. This analysis can be conducted from all three coordinate axes. In all three directions, a clear inclination towards certain positions is observable. In X- direction, which is the direction from the Computer through the Cryostat as visible in figure 35[13], the simulation reveals one spot that is hit more frequently than the other positions. From a geometrical perspective, the radioactive source is placed directly under the holder. The Cryostat itself could deflect the particles that hit the inner shells towards the center, leading to the outer boundaries of the sensitive area. Consequently, the likelihood of a particle striking this area is increased, leading to a higher number of unique particles at the outer walls of the holder.
A similar picture can be seen along the Y- axis, which is perpendicular to the X- Axis and rotated

[12] As a reminder, the simulated particles have an energy of $1MeV$.
[13] One has to keep in mind that the origin is NOT in the center of the Cryostat, it is in an arbitrary position in the room, which is due to the implementation of the .STL file as discussed in chapter 4.5.

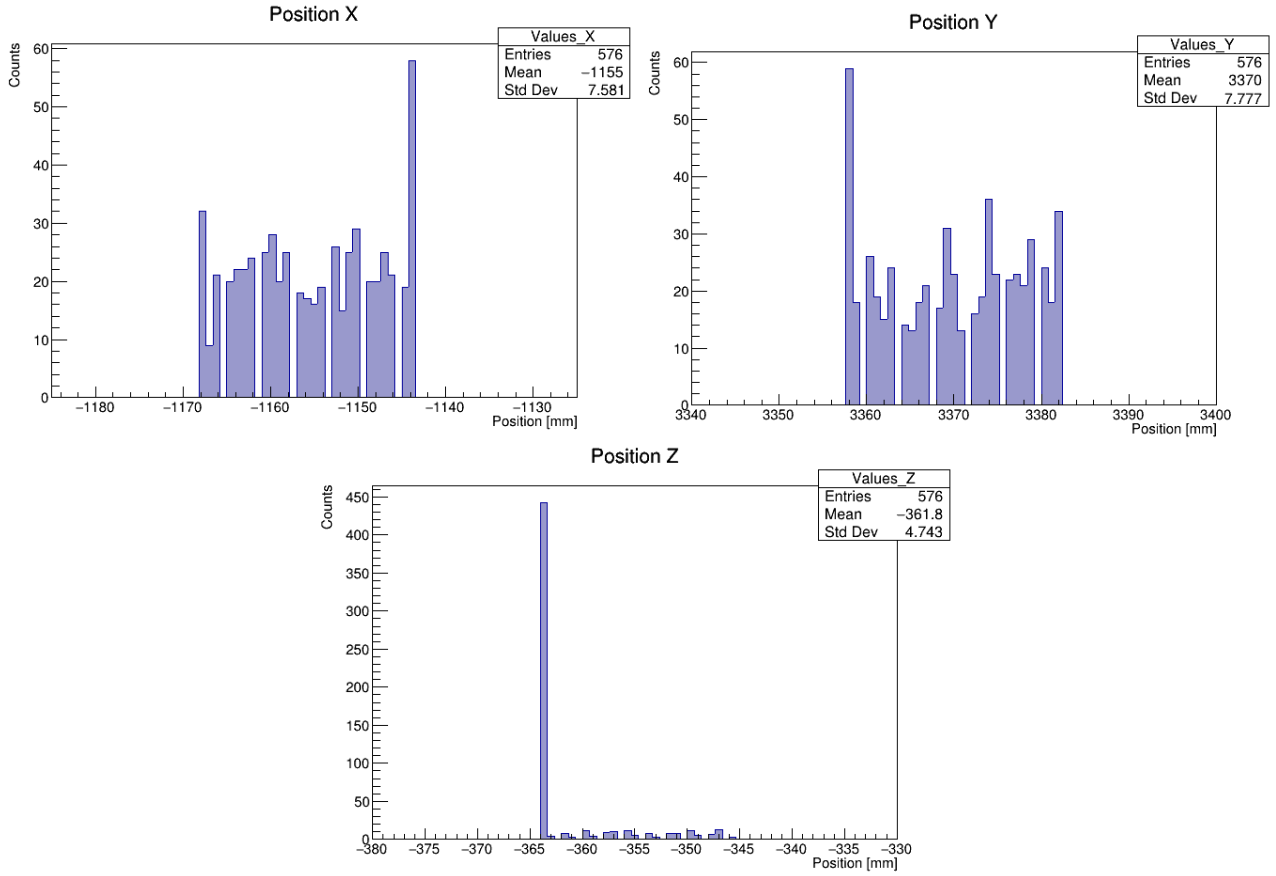


Figure 47: Hits in relation to the position for 10'000'000 particles

90 degrees counterclockwise. Again, the same argument with the deflected particles can be used here, since the Cryostat is in cylindric form and therefore symmetric in all directions. Therefore in Y- direction, the sides of the sensitive area are hit more frequently. For a more comprehensive understanding of the laboratory setup, please refer to figures 36, 35 and 34.
For the Z- axis one have a interesting result. The Z- axis is facing from the bottom of the Cryostat to the top. Since the source is placed directly below the holder on the Z- axis, we see a big increase in hits from the bottom of the holder. This substantial peak can be attributed to particles hitting the holder from directly underneath. Further, some particles could be deflected due to the shells of the Cryostat. This is not the case for the upper part of the holder, since above the holder, there are not many components that could deflect the particle. Further the gamma particles could leave the Cryostat by passing through the upper shells. The ones that are deflected by the shells loose that much energy, that they don't have enough energy left to be detected on the sensitive area.
An additional analysis conducted focused on the kinetic energy of the unique particles. It features one significant peak at $0.18keV$, which hosts most of the gamma particles, since we simulated all particles with the same energy of $662keV$.
Consequently, they behave similarly, sharing identical characteristics.
The final analysis involved the important aspect of energy deposition from the particles. It is essential to remember that this plot contains all the hits in the sensitive area, regardless of how often a single particle hits the holder. This is important because we are interested in knowing the total energy deposited in the sensitive area, and a particle can hit multiple times, thereby depositing energy on each occasion.
A considerable part of the particles deposit energy close to 0. This is due to the fact that the gamma particles loose energy while travelling to the outer shells of the Cryostat and the vacuum inside. In the graph itself, the hits with an energy close to 0 are rounded to 0, due to the nature of the code written in chapter 4.9. As one can see in figure 49, the values are cut out to make the data more presentable.

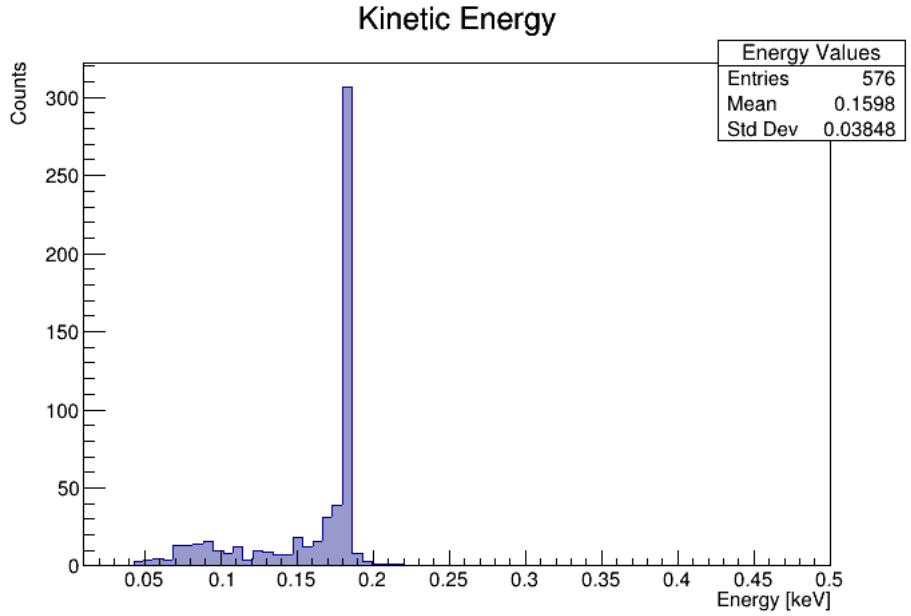


Figure 48: kinetic energy of the unique particles for 10'000'000 particles

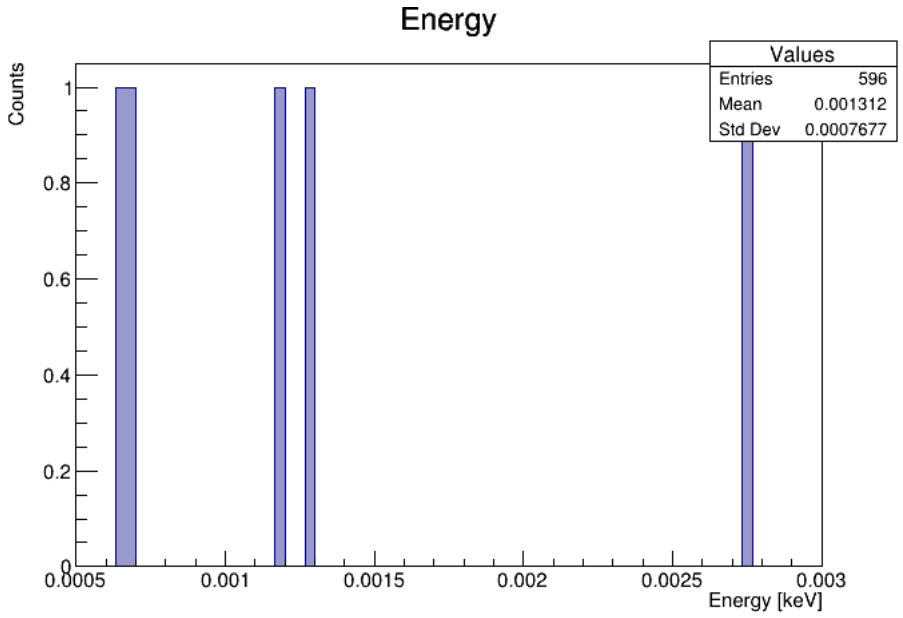


Figure 49: Energy deposit for all hits in 10'000'000 particles

However, the raw data is not affected by this rounding, therefore the mean energy deposit and total deposited energy are not influenced. Furthermore, only a handful of particles deposit the lion's share of the energy.

5.5.2 Simulation of 25'000'000 particles

The same simulation, utilizing identical parameters, was performed with a set of 25'000'000 particles. As shown in figure 50, the results are very similar to those obtained for the 10'000'000-particle sim-

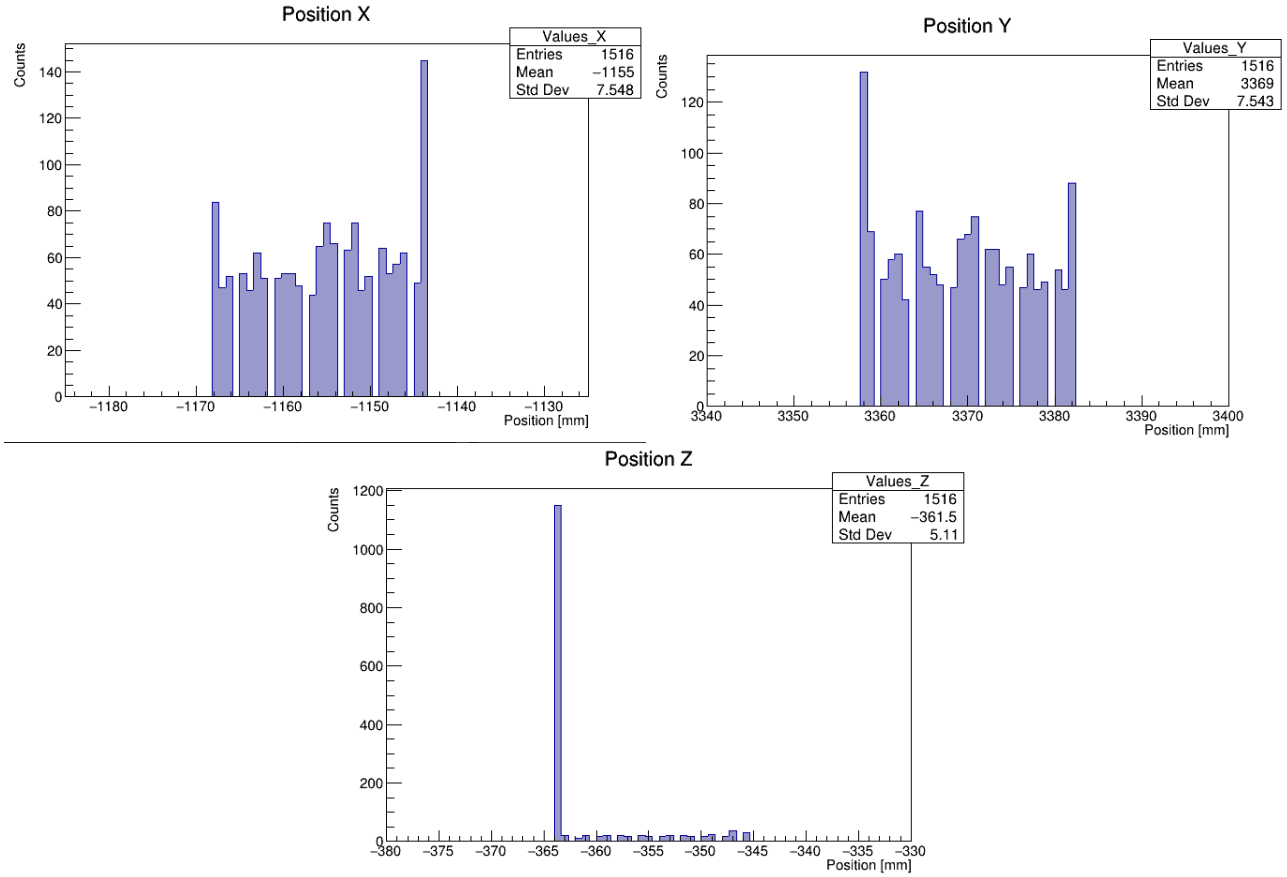


Figure 50: Hits in relation to the position for 25'000'000 particles

ulation, which clearly proves the viability of our simulation. We have a similar peak at the same position. The primary distinction lies in the count volume, which, predictably, is greater given that we're simulating double the number of particles.

A similar pattern can be seen for the kinetic energy in figure 51. Once again, a sizeable peak stands out in relation to the remainder of the dataset.

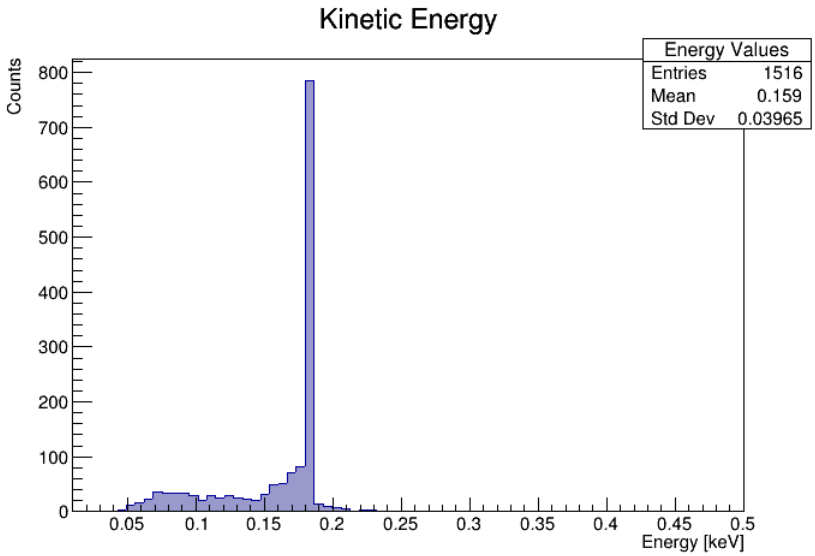


Figure 51: kinetic energy of the unique particles for 10'000'000 particles

For the case of 25'000'000-particle simulation, the energy deposition plot in in figure 52 once more mirrors its earlier counterparts.

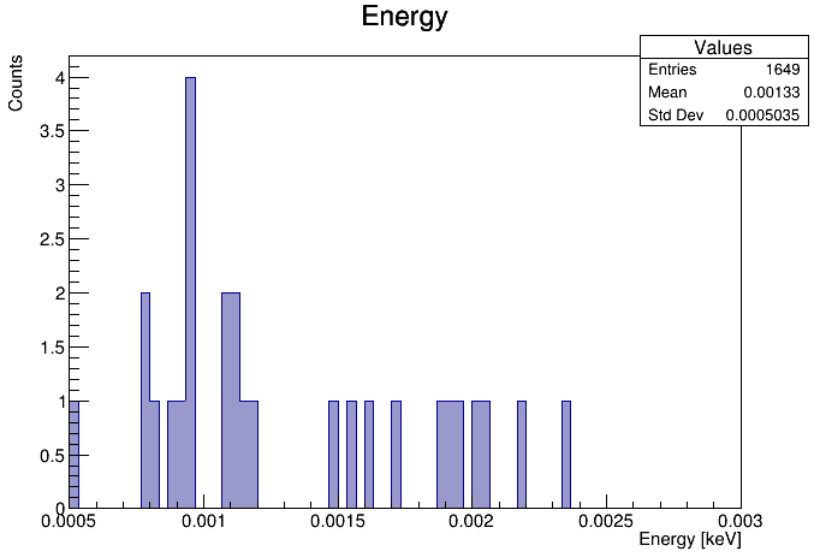


Figure 52: Energy deposit for all hits in 25'000'000 particles

The simulations carried out with 100'000, 500'000 and 2'000'000 particles in appendix 8.4 all yield similar results. Consequently, one can reasonably expect that conducting a simulation with an approximate total of 140'000'000 gamma particles would yield a congruous outcome.

6 Discussion

6.1 Implementation of ChatGPT in automation process

In late 2022, an American company called OpenAI[88], which is specialized on artificial intelligence (AI), introduced their newest Chatbot, called *ChatGPT*. In early 2023, they launched GPT-4 which is the most advanced language model yet. I inserted here the description it gave of itself:

> ChatGPT, developed by OpenAI, is an advanced artificial intelligence model trained using a technique known as transformer-based machine learning, specifically a method called Generative Pre-training Transformer (GPT). The current version at the time of writing, GPT-4, stands at the cutting edge of natural language processing, an AI field that deals with the interaction between computers and human languages. GPT-4 has been trained on a diverse range of internet text up until its knowledge cutoff in September 2021, enabling it to generate human-like text based on the inputs it receives.
>
> This language model functions by predicting the next word in a given sentence, with the ability to generate coherent and contextually accurate passages of text. This capability allows it to assist with a wide array of tasks, from drafting emails and writing articles, to coding help and even creative writing. However, while its responses can be impressively nuanced, it's crucial to note that GPT-4 lacks consciousness and doesn't have beliefs, desires, or subjective experiences. It doesn't understand content in the same way humans do and doesn't have access to personal data about individuals unless explicitly provided in the conversation. Its purpose is to provide information, answer queries, and facilitate a more fluid interaction between humans and machines, promoting the utility of AI in various domains, including education, research, and professional writing.
>
> Despite the remarkable advancements in language understanding and generation represented by models like GPT-4, these tools have their limitations. They can sometimes make mistakes, propagate biases present in the training data, or generate inappropriate content, thus necessitating careful and responsible use. OpenAI continues to strive for improvements in these areas, aiming to create more robust, useful, and ethical AI systems.[14]

This above quote was completely generated by GPT-4 itself. As one can see, the quality of the language is impressively high and it is well written. One could assume, that GPT-4 was used to write large pieces of this work. This is definitely **NOT** the case. The contents, the knowledge, the experience and the insights in this thesis was completely provided by me (and some topic, as described in the abstract and in the Master thesis by Severin) and external sources, as given by the references at the end of this thesis. For this type of information, GPT-4 cannot suffice and does not provide a profound and deep understanding of the topic.
Nevertheless, it was used to improve the level of English in this thesis. It was used to find spelling and typographical errors. Moreover, it was further very useful to give information about Geant4, since this is based on C++, which is a very well known programming language by ChatGPT. It was able to optimize the variable names to reduce the confusion and it helped locating cross- file errors.
In conclusion, while AI language models can be powerful tools for improving writing and providing general information, they should always be used in conjunction with human oversight and expertise.[89]

[14] This paragraph was written using GPT-4 Version 24th of May 2023

6.2 Problems

During the last year, we faced a lot of challenges and problems regarding the lab work and the simulation in Geant4. Some of them could be resolved, for example the problems with Geant4, some others were permanent and are still occurring until this day. In this chapter, some of the problems will be discussed and if possible, solutions will be provided. Those solutions could help to get rid of those problems in the future.

6.2.1 Geant4

As discussed in chapter 4.1, the simulation of the laboratory was carried out successfully. Nevertheless, a few problems occurred. The problems started at the beginning with the installation of Geant4. The combination of the lack of knowledge and the scarce documentation lead to a quite challenging installation. Fortunately, the UZH provided a readily deployable version of Geant4, which is installed on the server, as written in 4.1. Another hurdle was the use of C++, the programming language on which Geant4 is based. This language was unfamiliar to the simulation creator, requiring extra time to learn and become proficient in it.
The problem regarding the code which needed most of the time was the implementation of the Cryostat as a mesh model. As highlighted in chapter 4.6, we used multiple separate files, and only one was successfully imported by CADMesh. At the beginning, we used a .STL file created in Solidworks, but switched then to Blender. The import of one single .STL was not successful, since we ended with the following warning:

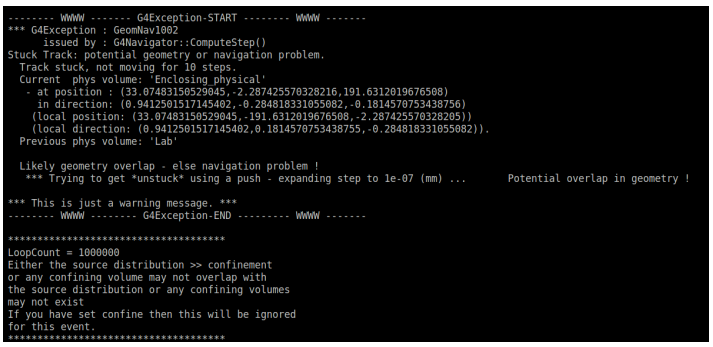


Figure 53: Warning message received by Geant4

Although, technically speaking, it was a warning and not an error, the resultant effect was a non-existing Cryostat in the simulation. Therefore we could not ignore this warning. As stated in 4.6, the strategy shifted towards using multiple smaller files and a majority of the laboratory was hardcoded. Since the geometry was simplified, the sensitive area could be implemented without any difficulties and the simulation could run smoothly. One aspect of the simulation that could be refined was the run time, since it needs a long time to complete the simulation, as explained in chapter 4.9. By optimizing some parts of the process, we could reduce the computational time and potentially complete the simulation with 140'000'000 particles. This improvement would make the simulation results more accurate and reliable for further analysis. Therefore, the next step for the simulation would be the improvement of the code to reduce the computational time. This could be done by reducing the area which is simulated. In the current state, the whole laboratory is simulated for each particle. But the hits in the PPMS systems for example do not have a huge impact on our hits inside the Cryostat due to the energy and the path of the gammas. Reducing the simulated area could significantly reduce the runtime.

6.2.2 Cooling down of the Cryostat

One of the major challenges throughout the whole measurement process was the cooling process of the Cryostat. For performing measurements with SNSPDs, the temperature at the cold stage should be as low as possible at around $100mK$, as discussed in the setup in chapter 5. In theory, the Cryostat is designed for reaching such low temperatures. The cooling process is strictly defined and monitored to prevent damage and ensure proper cooling. For this, the company which manufactured the Cryostat, *Oxford Instruments*, has provided a comprehensive manual detailing the process that one can follow. Regrettably, despite following the guide, as depicted in chapter 8.6, it was not possible to regularly cool down the Cryostat adequately. The problem with this process is the time needed for cooling. From beginning until reaching the temperature, around one week is needed. Given the tight timeframe we had for this project, this ended up as a problem. If the temperature is not reached, we could not execute SNSPD-measurements at around 1K, which was the case for some runs.
Nonetheless, we attempted to get rid of this error by implementing multiple different strategies. Firstly, we contacted *Oxford Instruments* for providing us additional information or manuals. Unfortunately they only referred us to the original manual, a copy of which we already possessed in our laboratory. Therefore we only had one source of information regarding the cool down process and had to rely on that solely. The guide was always followed but resulted in different temperatures.
Another approach we attempted was reducing the mass of the components inserted into the Cryostat. Given that we are operating in the temperature range of $200 - 500mK$, any mass introduced had an impact on the cooling efficiency.

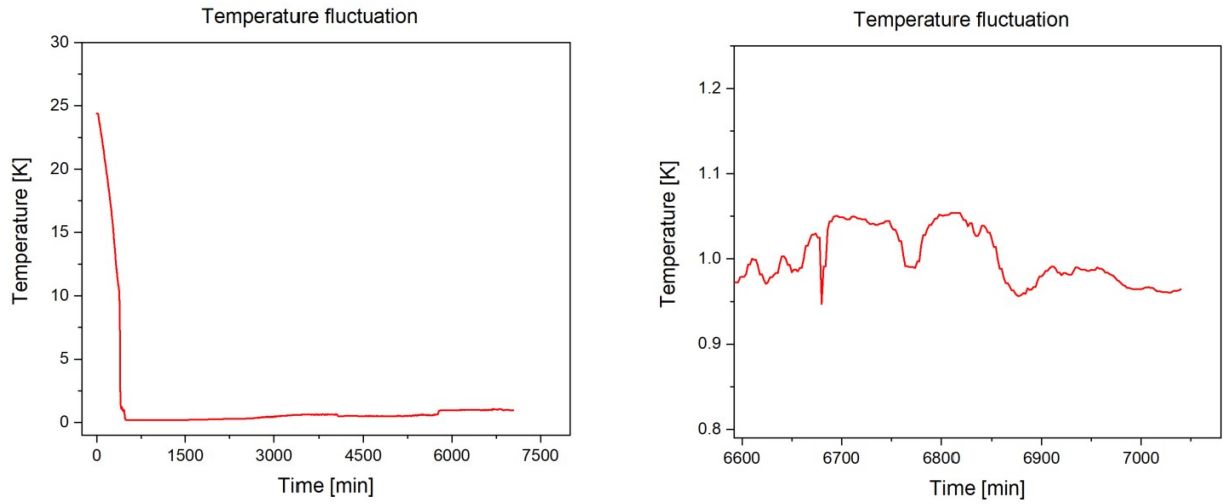


Figure 54: Analysis of the temperature fluctuation. Right plot is showing a zoomed in version of the range from 6800min to 7200min.

As shown in figure 54, the temperature has a large fluctuation especially at the low temperature range. Since the cooling process was not successful, we assumed the additional mass of the holder, which is around 350g, had a too large impact on the cooling efficiency, therefore the mass of the whole holder has to be reduced to around 50% of its original mass by cutting "unnecessary" parts from the pieces. But sadly, those two steps did not help with the cool down process.

Nevertheless, as evident in figure 55, the cooling process did work properly on occasion. However, by the end of this project, we were unable to get completely rid of this issue. As such, identifying the source of this problem and ensuring a consistently efficient cooling process will be part of the future work with this Cryostat.

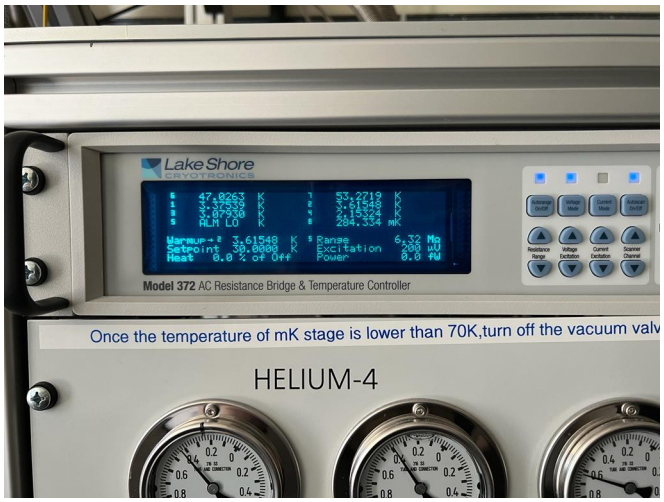


Figure 55: Picture of the Monitor showing the temperature of the inside of the Cryostat

6.2.3 Problems with films and samples

For this experiment, many films and samples have been fabricated and tested. This fabrication resulted in some difficulties and challenges. One of the main challenges was latching, which is described in chapter 3.2. To prevent latching, we started including a shunt resistor into the setup. It can be observed that with a shunt resistor, I_c can be increased. This was necessary for our experiment, otherwise latching would have occurred. With a shunt included, the PCR increases exponentially. But for our purpose, the critical current still had a value, which was too low.[90] This issue can be verified at high current, since relaxation oscillations can occur.
When the bias current surpasses the critical current, a thermal relaxation oscillation occurs, because of the hotspot's periodic appearance, growth and disappearance. Therefore these peaks cannot be counted as single photon counts. Relaxation oscillations are therefore a phenomenon of the chosen setup and the SNSPD, and are not due to the absorption of photons.[91, 92] Another challenge appears

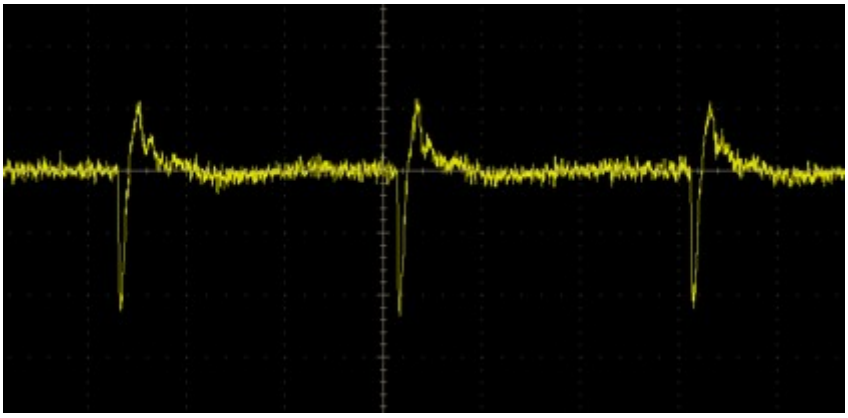


Figure 56: Screenshot of the Oscilloscope during relaxation oscillation

during the fabrication of the films. There are multiple reasons for the film to get degraded. The main causes for degradation are:

- External influences: During fabrication, the WSi films can be degraded due to interaction with the environment
- Natural Oxygen: The films have to be stored in a vacuum environment, since the Oxygen can interact with them causing them to degrade.

- Local defects: The films could have a non- uniform thickness, resulting in an uncertainty on the area of the film.
- Sensitivity for single photon detection caused by the stoichiometry of the film

Those challenges needed a long time to get rid of them, causing a few runs of samples to be degraded and not usable for our purpose. One of the main solutions for those problems was the changing of the stoichiometry of the films, in principle increasing the percentage of the silicon in our film. It was shown by Verma et al. in [93] that tuning the stoichiometry of the film increases the resistivity of the film. Increasing the percentage of silicon improves the sensitivity of single photon detection, but decreases the critical temperature of the film. For our purpose, this compromise was sufficient for measuring PCR.

6.3 Conclusion

This work consisted of two pillars: The first one was the PCR measurement and the characterization of the samples and films. The laboratory work focused on two systems: The film characterization in the PPMS and the sample characterization with the Cryostat. The film characterization described in chapters 3.3 and 5.2 was conducted successfully by using a default recipe, which was already proven to be working. The problems appeared when we switched to the Cryostat to characterize the samples. We faced a lot of problems, as described in chapter 6.2. Due to the nature of SNSPD and physical measurements, this is expected and may lead to useful derivations anyway. Furthermore, the timed boundaries of the master thesis lead to less time to resolve those problems and go further into characterization. Nevertheless, a lot of samples and films have been characterized, which helped set boundaries for different parameters. Finally, we were able to measure PCR with a sufficient plateau and a constant quantum efficiency, which is shown in chapter 5.4.
The second pillar was the simulation of the radioactive source in the laboratory. In conclusion, the simulation was deemed successful. After overcoming the initial challenges with Geant4 as outlined in chapter 6.2.1, the simulation yielded reasonable results. The amount of deposited energy is significant for our purpose, however, it is important to note that the source used in this simulation has a radioactivity level considerably higher than that of typical background radiation. If one assumes this setup is located in a laboratory such as Gran Sasso, known for its exceptionally low background radiation level,[94, 95] the energy deposit from environmental decays would be significantly less than what we observed in our simulation. Nonetheless, within the context of low-eV energy range, it's critical to remember that even minimal energy deposition could potentially trigger false hits or measurements. This reinforces the necessity of implementing a robust shielding system in SNSPD measurements, as it offers a vital line of defense against interference from environmental factors. The importance of reliable shielding cannot be understated. It could mean the difference between a successful, accurate measurement and a flawed one. Hence, future efforts must continue to focus on optimizing shielding systems for the best possible results.

6.4 Outlook

As previously stated, the timeline for this project was rather tight. Coupled with the ongoing challenges concerning the cooldown process, as detailed in chapter 6.2, not all planned tasks could be executed as intended. One such task was the installation of the QCL, as outlined in chapter 2.6.2. Nevertheless, all necessary components are already ordered or available in the laboratory, such as the Laser Diode shown in figure 57 and the Quantum Cascade Laser itself shown in figure 58.
Regarding the installation itself, some preliminary considerations have been made. In appendix 8.5, the setup has been attached as a PDF. This setup presents the method of incorporating the QCL into the holder and its connection to the ITC4020 Laser Diode. While the setup is not in its final form, given that the QCL has not been utilized at this stage of the project, the necessary components have been assembled and the installation process appears to be promising.

Figure 57: Laser source used in potential future setups

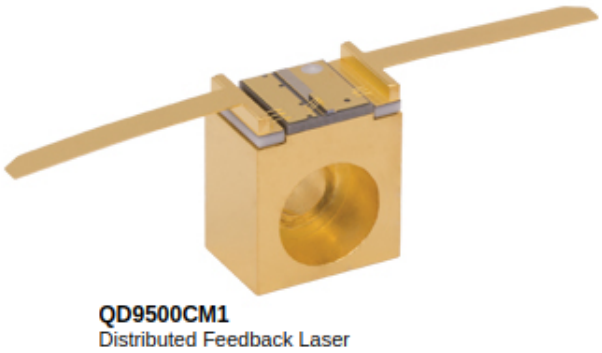


Figure 58: QCL used in future setups, it will contain two equal pieces

Some effort will need to be dedicated to connecting the QCL to the Cryostat, but this could serve as a focal point for a future project in this domain.
The next step for the research with SNSPDs would be the implementation of a nanowire into a big Underground laboratory. Underground laboratories are particular interesting for low- energy dark matter experiments, which require a low background environment. The *Laboratori Nazionali del Gran Sasso* in Italy[15], which hosts experiments such as XENON[96] and BOREXINO[97], is a promising candidate for this implementation.[94, 98–100]. Unlike traditional experiments that require large amounts of material, such as XENON with 3.2 tons of liquid ultra radio-pure Xenon, the implementation of SNSPDs in these experiments does not require high material volumes. Nonetheless, the shield provided by the 1.5km of rock can significantly reduce the background noise, making it beneficial for experiments in the light mass range. For this purpose, we are building our own experiment at Gran Sasso, which is called *Qrocodile*, which stands for *Quantum sensor cryogenic search for dark matter in light range*. The collaboration involves groups from around the world, including the University of Zurich (UZH), the Hebrew University and the Massachusetts Institute of Technology (MIT).

[15]https://www.lngs.infn.it/en, Accessed on 21.06.2023

7 Acknowledgements

This work was made possible with the assistance of the group of Prof. Laura Baudis and the group of Prof. Andreas Schilling from the University of Zurich. I am deeply grateful for the opportunity they provided to gain insight into the scientific research in the field of superconductivity combined with particle physics.
I would like to extend special thanks to Dr. Ilya Charaev, who was my supervisor during this period and offered immense knowledge and an extraordinary amount of patience.
Additionally, I would like to thank Stefan Siegrist, the laboratory assistant for Prof. Schilling's group. He consistently supported us in the laboratory during challenging times when helium was scarce.
I am also grateful to the members of the aforementioned research groups who supported us throughout the entire process. Special mentions go to Huanlong Liu, for his assistance in the lab, and Alexander Bismark, Dr. Diego Ramírez García and Dr. Jose Javier Cuenca García for their guidance in the computational aspects of working with Geant4.
I would like to express my gratitude to Dr. Ben Lehmann from the Massachusetts Institute of Technology, Prof. Yonit Hochberg from the Hebrew University of Jerusalem, Prof. Ben Kilminster and Prof. Titus Neupert from the University of Zurich for their valuable discussions and meetings throughout this year. I am truly appreciative of this experience and always felt welcomed in the research groups. It was a honor to work with everyone involved in this project.
This has certainly been an enriching journey, and it wouldn't have been the same without your collaboration, Severin Nägeli. The support you've provided throughout this project has been indispensable, and it has truly made a difference both in the physical work and in our mental fortitude. It is through our shared efforts and commitment that we've managed to reach the end of this work. It has been an honor to work with you, not only during this year, but during our many years of studying, crying and laughing.
For those interested in a deeper dive into the details of the code or its development, please visit the GitHub Repository maintained by the Qrocodile group at the University of Zurich.

8 Appendix

8.1 Automatization code for Yokogawa GS200

This code has been written using Python 3.9.12 in Spyder IDE 5.3.3 installed on a 64bit machine. Most important packages to make it run are:

- Qt 5.15.2
- PyQt5 5.15.7
- PyVISA 1.8

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import time
import pyvisa

#VISA setup

rm = pyvisa.ResourceManager()
yokogawa = rm.open_resource('GPIB::1::INSTR')
yokogawa.timeout = 10000

yokogawa.write("*IDN?")
print("ID of Yokogawa: ")
print( yokogawa.read())

# Setup yokogawa for  current loop
yokogawa.write("*RST")
yokogawa.write(":SOUR:FUNC VOLT")  # current source
yokogawa.write(":SOUR:RANG 1E-3")
yokogawa.write(":SOUR:FUNC CURR")  # current source
yokogawa.write(":SOUR:LEV 0") # set current to 0
yokogawa.write(":SOUR:RANG 100E-3")
yokogawa.write(":OUTP ON")         # turn on output
print("Yokogawa GS200 connected")

#Limit for the range
x = 25e-6
step = 101

#loop to sweep current
first_range = np.linspace(0,x-x/step, step)
second_range = np.linspace(x,-x+x/step,2*step)
third_range = np.linspace(-x,0,step)
run = np.concatenate((first_range,second_range, third_range))
print(run)
voltage = []
current = []

print("Starting Measurements ...")
for I in run:
    yokogawa.write(":SOUR:FUNC CURR")
```

```
    I = I
    yokogawa.write(":SOUR:LEV " + str(I))
    time.sleep(0.5)
    try:
        while not yokogawa.query("*OPC?"):
            pass
    except Exception as e:
        print("Error: ", e)
        break
    data = yokogawa.query(":FETC?").split(",")
    vread = float(data[0])
    #i = float(data[1])
    current.append(I)
    voltage.append(vread)
    print(f"--> Current = {I} A, Voltage = {vread} V")

yokogawa.write(":OUTP OFF")
yokogawa.close()

#saving

column1 = pd.Series(voltage, name="Voltage")
column2 = pd.Series(current, name="Current")
x = pd.concat([column1,column2],axis = 1)
df = pd.DataFrame(x)
print(x)
df.to_csv('IVMeasurements.txt', header=None, index = None, sep=' ')

#plotting

plt.plot(voltage, current, linewidth = 1.5, color = 'black')
plt.title("IV- curve")
plt.xlabel("Voltage [V]")
plt.ylabel("Current [A]")
plt.savefig("IV-curve.png")
plt.show()
```

8.2 Sheet Resistance

To reduce the calculations for the sheet resistance of multiple samples by hand, this small script has been written. It is written using Python 3.9.12 in Spyder IDE 5.3.3.

```
import numpy as np

print("Use voltage in units of [V]")

pi = np.pi
f = 1           #Modification factor
I = 0.001       #Bias Current

#Input Values for the Voltage

V_1 = float(input("Enter V1: "))
V_2 = float(input("Enter V2: "))
V_3 = float(input("Enter V3: "))
V_4 = float(input("Enter V4: "))

sum = (V_1+V_2)/I + (V_2+V_3)/I + (V_3+V_4)/I + (V_4+V_1)/I

SheetResistance = (pi/ (8*np.log(2)))*sum

print(SheetResistance,"\u03A9")
```

8.3 Geant4 Simulation

In this chapter, I want to highlight the most crucial segments of the source code used in the simulation. For the complete code, please refer to GitHub. This code has been developed using Geant4 Version 10.06.02, with Visual Studio 1.79.2 as a platform. Additional packages required for the simulation were CRY v1.7 Library and RadSrc v1.6 both sourced from the Lawrence Livermore National Laboratory (LLNL).
I would also like to extend my gratitude to Dr. Davide Franco from the *Laboratoire Astroparticule & Cosmologie* at the *Université Paris Cité* for his exceptionally insightful and detailed documenation of Geant4. THe compilation of the code was done via a SSH access to the farm of the University of Zurich.

8.3.1 CMakeLists.txt

```
#----------------------------------------------------------------------------
# Setup the project
cmake_minimum_required(VERSION 2.6 FATAL_ERROR)

project(QROCODILE_G4)
set(CMAKE_CXX_COMPILER g++)
set(CMAKE_CXX_STANDARD 11)
set(CMAKE_CXX_STANDARD_REQUIRED ON)

set(CMAKE_RUNTIME_OUTPUT_DIRECTORY "bin")
set(CMAKE_LIBRARY_OUTPUT_DIRECTORY "lib")
set(INSTALL_BIN_DIR "${CMAKE_INSTALL_PREFIX}/bin" CACHE PATH "Installation directory for executables")
set(INSTALL_LIB_DIR "${CMAKE_INSTALL_PREFIX}/lib" CACHE PATH "Installation directory for libraries")
set(INSTALL_INC_DIR "${CMAKE_INSTALL_PREFIX}/include" CACHE PATH "Installation directory for headers")
set(EXECUTABLE_OUTPUT_PATH "bin")
set(CMAKE_MACOSX_RPATH 1)

#----------------------------------------------------------------------------
# Find Geant4 package, activating all available UI and Vis drivers by default
# You can set WITH_GEANT4_UIVIS to OFF via the command line or ccmake/cmake-gui
# to build a batch mode only executable
#
option(WITH_GEANT4_UIVIS "Build example with Geant4 UI and Vis drivers" ON)
if(WITH_GEANT4_UIVIS)
  find_package(Geant4 REQUIRED ui_all vis_all)
else()
  find_package(Geant4 REQUIRED)
endif()
#-----------------
#Find ROOT (required package)
#
list(APPEND CMAKE_PREFIX_PATH $ENV{ROOTSYS})
find_package(ROOT REQUIRED COMPONENTS MathCore RIO Hist Tree Net)
include(${ROOT_USE_FILE})

include_directories(${CMAKE_SOURCE_DIR} ${ROOT_INCLUDE_DIRS})
#----------------------------------------------------------------------------
# Setup Geant4 include directories and compile definitions
# Setup include directory for this project
add_definitions(${ROOT_CXX_FLAGS})
include(${Geant4_USE_FILE})
include_directories(${PROJECT_SOURCE_DIR}/include
                    ${Geant4_INCLUDE_DIR}
                    ${ROOT_INCLUDE_DIR})

#----------------------------------------------------------------------------
# Locate sources and headers for this project
file(GLOB sources ${PROJECT_SOURCE_DIR}/src/*.cc)
file(GLOB headers ${PROJECT_SOURCE_DIR}/include/*.hh)


```

```
#----------------------------------------------------------------------------
# Create shared library
add_library(QROCODILE SHARED ${sources} ${headers})
target_link_libraries(QROCODILE ${Geant4_LIBRARIES})
target_link_libraries(QROCODILE ${ROOT_LIBRARIES})
#----------------------------------------------------------------------------
# Add the executable, and link it to the Geant4 libraries

add_executable(QROCODILE_G4 qrocodile.cc ${headers})
target_link_libraries(QROCODILE_G4 QROCODILE)
target_link_libraries(QROCODILE_G4 ${Geant4_LIBRARIES})
target_link_libraries(QROCODILE_G4 ${ROOT_LIBRARIES})

#----------------------------------------------------------------------------
# Install the executable to 'bin' directory under CMAKE_INSTALL_PREFIX

install(TARGETS QROCODILE_G4 DESTINATION bin)
```

8.3.2 Main QROCODILE.cc file

```
#include <string>
#include <sstream>
#include <unistd.h>

#include <G4RunManager.hh>
#include <G4UImanager.hh>
#include <G4UIterminal.hh>
#include <G4UItcsh.hh>
#include "G4Event.hh"
#include <G4VisExecutive.hh>
#include <G4UIExecutive.hh>
#include <G4GeneralParticleSource.hh>
#include <G4VModularPhysicsList.hh>
#include "Shielding.hh"
#include "G4RadioactiveDecayPhysics.hh"

#include "QrocoDetectorConstruction.hh"
#include "QrocoPrimaryGeneratorAction.hh"
#include "QrocoAnalysisManager.hh"
#include "QrocoRunAction.hh"
#include "QrocoEventAction.hh"
#include <QBBC.hh>

void usage();

int main(int argc, char **argv){

  // Switch case
  int c = 0;

  std::stringstream Stream;

```

```
33  bool Interactive = false;
34  bool Visualize = false;
35  bool VrmlVisualize = false;
36  bool QtVisualize = false;
37  bool OpenGlVisualize = false;
38  bool HepRepVisualize = false;
39  bool RunMacro = false;
40  bool RunInitMacro = false;
41  std::string MacroName, RootFileName, PreInitMacroName;
42  std::string Command;
43  int EventsToSimulate = 0;
44
45  // parse switches
46  while((c = getopt(argc,argv,"v:f:o:p:n:i")) != -1)
47  {
48    switch(c)
49    {
50      case 'v':
51        Visualize = true;
52        Stream.str(optarg);
53        if(Stream.str() == "vrml")
54          VrmlVisualize = true;
55        else if(Stream.str() == "opengl")
56          OpenGlVisualize = true;
57            else if(Stream.str() == "qt")
58          QtVisualize = true;
59            else if(Stream.str() == "heprep")
60              HepRepVisualize = true;
61        Stream.clear();
62        break;
63
64      case 'f':
65        RunMacro = true;
66        MacroName = optarg;
67        break;
68
69      case 'o':
70        RootFileName = optarg;
71        break;
72
73      case 'p':
74        RunInitMacro = true;
75        PreInitMacroName = optarg;
76        break;
77
78      case 'n':
79        Stream.str(optarg);
80        Stream.clear();
81        Stream >> EventsToSimulate;
82        break;
83
84      case 'i':
85        Interactive = true;
86        break;
```

```
87
88      default:
89        usage();
90    }
91  }
92
93  if(RootFileName.empty()) RootFileName = "output.root";
94
95  // create the run manager
96  G4RunManager* TheRunManager = new G4RunManager;
97
98  // Detector Construction
99  QrocoDetectorConstruction* DetCon = new QrocoDetectorConstruction();
100 TheRunManager->SetUserInitialization(DetCon);
101
102 // Physics List
103 G4VModularPhysicsList* ThePhysicsList = new Shielding;
104 TheRunManager->SetUserInitialization(ThePhysicsList);
105
106 //Test Radioactive source
107 G4VModularPhysicsList* physicsList = new QBBC;
108 physicsList->RegisterPhysics(new G4RadioactiveDecayPhysics());
109 TheRunManager->SetUserInitialization(physicsList);
110
111 // Primary generator action
112 QrocoPrimaryGeneratorAction* ThePrimaryGeneratorAction = new QrocoPrimaryGeneratorAction();
113
114 // Analysis manager
115 QrocoAnalysisManager* TheAnalysisManager = new QrocoAnalysisManager(ThePrimaryGeneratorAction);
116 TheAnalysisManager->SetOutputFilename(RootFileName);
117
118 // Visualization Manager
119 G4VisManager* TheVisManager = new G4VisExecutive;
120 TheVisManager->SetVerboseLevel(0);
121 TheVisManager->Initialize();
122
123 // Add user-defined action classes to the run manager
124 TheRunManager->SetUserAction(ThePrimaryGeneratorAction);
125 TheRunManager->SetUserAction(new QrocoRunAction(TheAnalysisManager));
126 TheRunManager->SetUserAction(new QrocoEventAction(TheAnalysisManager));
127
128 // Geometry manager
129 G4UImanager* UImanager = G4UImanager::GetUIpointer();
130
131 // Preinit macro
132 if(RunInitMacro)
133 {
134   Command = "/control/execute " + PreInitMacroName;
135   UImanager->ApplyCommand(Command);
136 }
137
138 // Initialise
139 TheRunManager->Initialize();
140
```

```
141   // UI session
142   G4UIExecutive* UI = 0;
143   G4UIsession* UIsession = 0;
144   if (QtVisualize)
145   {
146     if(Interactive) UI = new G4UIExecutive(argc, argv, "Qt");
147   } else {
148     if(Interactive) UIsession = new G4UIterminal(new G4UItcsh);
149   }
150
151   if(Visualize)
152   {
153     UImanager->ApplyCommand("/vis/scene/create");
154     if(VrmlVisualize)
155       UImanager->ApplyCommand("/vis/open VRML2FILE");
156     if( (OpenGlVisualize) || (QtVisualize) )
157       UImanager->ApplyCommand("/vis/open OGL");
158     if(HepRepVisualize)
159       UImanager->ApplyCommand("/vis/open HepRepFile");
160
161           UImanager->ApplyCommand("/vis/drawVolume");
162           UImanager->ApplyCommand("/vis/viewer/set/viewpointThetaPhi 90 0 deg");
163     UImanager->ApplyCommand("/vis/viewer/set/upVector 0 0 1");
164     UImanager->ApplyCommand("/vis/viewer/zoom 5.0");
165           UImanager->ApplyCommand("/vis/scene/add/trajectories");
166           UImanager->ApplyCommand("/vis/scene/add/hits");
167     UImanager->ApplyCommand("/tracking/storeTrajectory 1");
168           UImanager->ApplyCommand("/vis/scene/endOfEventAction accumulate");
169           UImanager->ApplyCommand("/vis/modeling/trajectories/create/drawByCharge");
170           UImanager->ApplyCommand("/vis/modeling/trajectories/drawByCharge-0/default/setDrawStepPts true");
171           UImanager->ApplyCommand("/vis/modeling/trajectories/drawByCharge-0/default/setStepPtsSize 5");
172           UImanager->ApplyCommand("/vis/modeling/trajectories/drawByCharge-0/default/setDrawAuxPts true");
173           UImanager->ApplyCommand("/vis/modeling/trajectories/drawByCharge-0/default/setAuxPtsSize 5");
174   }
175
176   // Command macros
177   if(RunMacro)
178   {
179     Command = "/control/execute " + MacroName;
180     UImanager->ApplyCommand(Command);
181   }
182
183   G4cout << "Loaded run macro: " << MacroName << G4endl;
184
185   if(EventsToSimulate)
186   {
187     Stream.str("");
188     Stream.clear();
189     Stream << "/run/beamOn " << EventsToSimulate;
190     UImanager->ApplyCommand(Stream.str());
191
192     G4cout << "Total number of events requested: " << EventsToSimulate << G4endl;
193   }
194
```

```
195   if(Interactive)
196   {
197     if (QtVisualize)
198     {
199       UImanager->ApplyCommand("/control/execute macros/ui.mac");
200       UI->SessionStart();
201       delete UI;
202     } else {
203       UIsession->SessionStart();
204       delete UIsession;
205     }
206   }
207
208   delete TheAnalysisManager;
209   if(Visualize) delete TheVisManager;
210   delete TheRunManager;
211
212   return 0;
213 }
214
215 void usage()
216 {
217   G4cout << " \nUsage: ./build/bin/QROCODILE_G4 [-p <preinitMacro>] "
218   "[-f <exeMacro>] [-n <number>] [-o <file>] [-i] [-v <engine>]\n" << G4endl;
219   G4cout << " The options are: " << G4endl;
220   G4cout << " -f:   command macro.\n"
221          << " -o:   output file.\n"
222          << " -p:   preinit macro.\n"
223          << " -n:   number of events.\n"
224          << " -v:   visualization (qt, opengl, vrml, heprep)\n"
225          << " -i:   interactive." << G4endl;
226   exit(EXIT_FAILURE);
227 }
228
```

8.3.3 Detector Construction

```
1  #include "QrocoDetectorConstruction.hh"
2  #include "QrocoDetectorConstructionMessenger.hh"
3  #include "CADMesh.hh"
4  #include <G4SystemOfUnits.hh>
5  #include <G4VisAttributes.hh>
6  #include <G4Colour.hh>
7  #include <G4Box.hh>
8  #include <G4Tubs.hh>
9  #include "G4RadioactiveDecay.hh"
10 #include "G4DecayTable.hh"
11 #include "G4ParticleDefinition.hh"
12 #include "QrocoDetectorMaterial.ihh"
13 #include "G4NistManager.hh"
14
15
```

```
16
17 QrocoDetectorConstruction::QrocoDetectorConstruction()
18 {
19   Messenger = new QrocoDetectorConstructionMessenger(this);
20 }
21
22 QrocoDetectorConstruction::~QrocoDetectorConstruction()
23 {
24   delete Messenger;
25 }
26
27 void QrocoDetectorConstruction::DefineMaterials()
28 {
29   #include "QrocoDetectorMaterial.icc"
30 }
31
32 G4VPhysicalVolume* QrocoDetectorConstruction::Construct()
33 {
34   DefineMaterials();
35
36   //Vacuum
37   G4NistManager* nist = G4NistManager::Instance();
38   G4Material* vacuum_mat = nist->FindOrBuildMaterial("G4_Galactic");
39
40
41   // Global lab parameters
42   G4double wall = 25.*cm;
43   G4double worldWidth = 1000.0*cm + 2.*wall;
44   G4double worldHeight = 400.0*cm + 2.*wall;
45   G4double worldLength = 1000.0*cm + 2.*wall;
46   G4double LabHalfX = worldWidth - 2.*wall;
47   G4double LabHalfY = worldLength - 2.*wall;
48   G4double LabHalfZ = worldHeight - 2.*wall;
49
50   //Rotation parameters
51   auto rotation = new G4RotationMatrix();
52   rotation->rotateX(90*deg);
53
54   // Room Walls
55   //G4Box* RoomBox = new G4Box("RoomBox", worldWidth/2, worldLength/2, worldHeight/2);
56   //G4LogicalVolume* RoomLogicalVolume = new G4LogicalVolume(RoomBox, world_mat, "RoomVolume", 0, 0, 0);
57   //auto Room_phys = new G4PVPlacement(0, G4ThreeVector(), RoomLogicalVolume, "Room", 0, false, 0);
58
59   // Lab volume
60   G4Box* LabBox = new G4Box("LabBox", LabHalfX/2, LabHalfY/2, LabHalfZ/2);
61   G4LogicalVolume* LabLogicalVolume = new G4LogicalVolume(LabBox, world_mat, "LabVolume", 0, 0, 0);
62   auto world_phys = new G4PVPlacement(0, G4ThreeVector(), LabLogicalVolume, "Lab",0, false, 0);
63
64
65
66   // Cryostat from CAD file
67   auto mesh = CADMesh::TessellatedMesh::FromPLY("./SingleComponents/PLY/ShellsTubesGrob.PLY");
68   G4VSolid *Cyl_solid = mesh->GetSolid();
69   auto Cyl_logical = new G4LogicalVolume(Cyl_solid, cryo_mat, "Cyl_logical", 0, 0, 0);
```

```
  new G4PVPlacement(rotation, G4ThreeVector(-1*m,2.2*m,0), Cyl_logical, "Cyl_physical",
      LabLogicalVolume,false, 0);

  // Vacuum volume
  G4VSolid* VacBox = new G4Tubs("VacBox", 0.0 * m, 0.134 * m, 0.170 * m, 0.0 * deg, 360.0 * deg);
  G4LogicalVolume* VacLogicalVolume = new G4LogicalVolume(VacBox, vacuum_mat, "VacVolume", 0, 0, 0);
  new G4PVPlacement(rotation, G4ThreeVector(-0.156*m, 0.355*m,1.17*m), VacLogicalVolume, "Vac",
      Cyl_logical, false, 0,true );

  //Cryostat primitive construction

  //Box

  G4VSolid* Box_solid = new G4Box("Box", 0.012 * m, 0.012 * m, 0.009 * m);
  G4LogicalVolume* Box_logical = new G4LogicalVolume(Box_solid, holder_mat, "Box_logical");
  new G4PVPlacement(0, G4ThreeVector(0,0,0), Box_logical, "Box_physical",VacLogicalVolume, false,
      0,true );

/*
 // Sensitive detector is Box
  G4SDManager *SDManager = G4SDManager::GetSDMpointer();
  QrocoBoxSensitiveDetector *BoxSD = new QrocoBoxSensitiveDetector("Qroco/BoxSD");
  SDManager->AddNewDetector(BoxSD);
  Box_logical->SetSensitiveDetector(BoxSD);
*/
  //Platform

  G4VSolid* Platform_solid = new G4Tubs("Platform", 0.0 * m, 0.0275 * m, 0.002 * m, 0.0 * deg, 360.0 * deg);
  G4LogicalVolume* Platform_logical = new G4LogicalVolume(Platform_solid, holder_mat, "Platform_logical");
  new G4PVPlacement(0, G4ThreeVector(0,0, 0.0115 * m), Platform_logical, "Platform_physical",
     VacLogicalVolume, false, 0,true );

  //Table

  G4VSolid* Table_solid = new G4Box("Table", 2.0*m, 0.8*m, 0.6*m);
  G4LogicalVolume* Table_logical = new G4LogicalVolume(Table_solid, Wood_mat, "Table_logical");
  new G4PVPlacement(0, G4ThreeVector(-2.5*m, 0.4*m,-0.5*m), Table_logical, "Table_physical",
      LabLogicalVolume, false, 0,true );

  //PC

  G4VSolid* PC_solid = new G4Box("PC", 0.5*m, 0.5*m, 1*m);
  G4LogicalVolume* PC_logical = new G4LogicalVolume(PC_solid, PVC_mat, "PC_logical");
  new G4PVPlacement(0, G4ThreeVector(-3.5*m, 3.3*m,-0.99*m), PC_logical, "PC_physical",
      LabLogicalVolume, false, 0,true );

  //Dingsbums

  G4VSolid* Dings_solid = new G4Box("Dings", 1*m, 1*m, 1.87*m);
  G4LogicalVolume* Dings_logical = new G4LogicalVolume(Dings_solid, Steel_mat, "Dings_logical");
  new G4PVPlacement(0, G4ThreeVector(1.7*m, 3.3*m, 0*m), Dings_logical, "Dings_physical",
      LabLogicalVolume, false, 0,true );

  //PPMS Chemistry
```

```
124
125   G4VSolid* PPMSC_solid = new G4Tubs("Chemistry", 0.0 * m, 0.35*m, 1.11*m, 0.0 * deg, 360.0 * deg);
126   G4LogicalVolume* PPMSC_logical = new G4LogicalVolume(PPMSC_solid, Steel_mat, "PPMSC_logical");
127   new G4PVPlacement(0, G4ThreeVector(-3.49*m, -2*m,-0.88*m), PPMSC_logical, "PPMSC_physical",
128       LabLogicalVolume, false, 0,true );
129
130   //PPMS Physics
131
132   G4VSolid* PPMSP_solid = new G4Tubs("Physics", 0.0 * m, 0.35*m, 1.11*m, 0.0 * deg, 360.0 * deg);
133   G4LogicalVolume* PPMSP_logical = new G4LogicalVolume(PPMSP_solid, Steel_mat, "PPMSP_logical");
134   new G4PVPlacement(0, G4ThreeVector(-3.49*m, -3.01*m,-0.88*m), PPMSP_logical, "PPMSP_physical",
135        LabLogicalVolume, false, 0,true );
136
137   //Wall & Closet
138
139   G4VSolid* Wall_solid = new G4Box("Wall", 2.0*m, 0.75*m, 1.7*m);
140   G4LogicalVolume* Wall_logical = new G4LogicalVolume(Wall_solid, Wood_mat, "Wall_logical");
141   new G4PVPlacement(0, G4ThreeVector(2.5*m, -2.5*m, -0.29*m), Wall_logical, "Wall_physical",
142        LabLogicalVolume, false, 0,true );
143
144   //Visulization attributes
145   G4VisAttributes* CylVisAtt = new G4VisAttributes(G4Colour(1,1,1)); //White
146   G4VisAttributes* BoxVisAtt = new G4VisAttributes(G4Colour(0.45,0.25,0)); //brown
147   G4VisAttributes* PlatformVisAtt = new G4VisAttributes(G4Colour(0.45,0.25,0)); //brown
148   G4VisAttributes* TableVisAtt = new G4VisAttributes(G4Colour(0.45,0.25,0)); //brown
149   G4VisAttributes* PCVisAtt = new G4VisAttributes(G4Colour(0.3,0.4,0.7)); //blue
150   G4VisAttributes* DingsVisAtt = new G4VisAttributes(G4Colour(0.5,0.5,0.5)); //grey
151   G4VisAttributes* PPMSCVisAtt = new G4VisAttributes(G4Colour(1,1,0)); //yellow
152   G4VisAttributes* PPMSPVisAtt = new G4VisAttributes(G4Colour(1,1,0)); //yellow
153   G4VisAttributes* WallVisAtt = new G4VisAttributes(G4Colour(0.45,0.25,0)); //brown
154   G4VisAttributes* VacVisAtt = new G4VisAttributes(G4Colour(0.45,0.25,0)); //brown
155   PC_logical->SetVisAttributes(PCVisAtt);
156   Dings_logical->SetVisAttributes(DingsVisAtt);
157   Table_logical->SetVisAttributes(TableVisAtt);
158   Cyl_logical->SetVisAttributes(CylVisAtt);
159   Box_logical->SetVisAttributes(BoxVisAtt);
160   Platform_logical->SetVisAttributes(PlatformVisAtt);
161   PPMSC_logical->SetVisAttributes(PPMSCVisAtt);
162   PPMSP_logical->SetVisAttributes(PPMSPVisAtt);
163   Wall_logical->SetVisAttributes(WallVisAtt);
164   VacLogicalVolume->SetVisAttributes(VacVisAtt);
165
166   G4cout << G4endl << "Cryostat constructed" << G4endl;
167
168   SecondOverlapCheck();
169   VolumesHierarchy();
170
171   G4cout << G4endl << "Full geometry of the experiment constructed" << G4endl;
172
173   return world_phys;
174 }
175
176 void QrocoDetectorConstruction::SecondOverlapCheck()
177 {
```

```
  G4PhysicalVolumeStore* thePVStore = G4PhysicalVolumeStore::GetInstance();
  G4cout << "\n" << "*****************************" << G4endl;
  G4cout << "\n" << "CHECK FOR OVERLAPS" << G4endl;
  G4cout << "\n" << "*****************************" << G4endl;
  G4cout <<"\n" << G4endl;

  G4cout << thePVStore->size() << " physical volumes are defined" << G4endl;

  G4bool overlapFlag = false;

  for (size_t i=0; i<thePVStore->size();i++)
        {
          overlapFlag = (*thePVStore)[i]->CheckOverlaps(5000) | overlapFlag;
  }
}

void QrocoDetectorConstruction::VolumesHierarchy()
{
  //=== Loop over all volumes and write to file list of: PhysicalVolume, LogicalVolume,
 //MotherLogicalVolume ===

  G4PhysicalVolumeStore* thePVStore = G4PhysicalVolumeStore::GetInstance();

  G4String PhysicalVolumeName;
  G4String LogicalVolumeName;
  G4String MotherVolumeName;

  std::string VolFileName =  "VolumesList.dat";
  std::ofstream Volumeslist(VolFileName);
  G4cout << ">>> Writing list of volumes to file: " << VolFileName << G4endl;
  Volumeslist << "PhysicalVolume LogicalVolume MotherLogicalVolume" << G4endl;

  // loop over all volumes in G4PhysicalVolumeStore and write to file
  for(size_t i = 1; i < thePVStore->size(); i++)
  {
    PhysicalVolumeName = (*thePVStore)[i]->GetName();
    LogicalVolumeName = (*thePVStore)[i]->GetLogicalVolume()->GetName();
    MotherVolumeName = (*thePVStore)[i]->GetMotherLogical()->GetName();
    Volumeslist << PhysicalVolumeName << " " << LogicalVolumeName << " "
                << MotherVolumeName << G4endl;
  }
  Volumeslist.close();
}
```

8.3.4 Detector Material

```
#include <G4Material.hh>


G4double density, a, z;
G4String name, symbol;
G4int ncomponents;
```

```
7
8  G4Element* O  = new G4Element(name="Oxygen", symbol="O", z= 8., a=16.00*g/mole);
9  G4Element* C  = new G4Element(name="Carbon", symbol="C", z= 6., a=12.01*g/mole);
10 G4Element* H  = new G4Element(name="Hydrogen", symbol="H", z= 1., a=1.01*g/mole);
11 G4Element* Cl = new G4Element(name="Chlorine", symbol="Cl", z=17., a=35.45*g/mole);
12 G4Element* Fe = new G4Element(name="Iron", symbol="Fe", z=26., a=55.85*g/mole);
13
14 // Aluminium
15 G4Element* Al = new G4Element(name="Aluminium"  ,symbol="Al" , z= 13., a=26.98*g/mole);
16 G4Material* metalAl = new G4Material(name="MetalAluminium", density=2.700*g/cm3, ncomponents=1);
17 metalAl->AddElement(Al, 1);
18
19 // Air
20 G4Element* N = new G4Element(name="Nitrogen",symbol="N" , z= 7., a=14.00674*g/mole);
21 G4Material* Air = new G4Material("AIR", 1.2929*kg/m3, 2, kStateGas, 300.00*kelvin, 1.0*atmosphere);
22 Air->AddElement(N, 0.8);
23 Air->AddElement(O , 0.2);
24
25 //Copper
26 G4Element* Cu = new G4Element(name="Copper", symbol="Cu", z=29., a= 63.54*g/mole);
27 G4Material* Copper = new G4Material(name= "CopperMaterial", density=  8.92*kg/m3, ncomponents=1);
28 Copper->AddElement(Cu,1);
29
30 //Plutonium
31 G4Element* Pu = new G4Element(name="Plutonium", symbol="Pu", z=94., a= 244.0*g/mole);
32 G4Material* Plutonium = new G4Material("Plutonium", 19.84*kg/m3, ncomponents=1);
33 Plutonium->AddElement(Pu,1);
34
35 //Cobalt
36 G4Isotope* Co59 = new G4Isotope(name="Co59", 27, 59, a=59.93*g/mole);
37 G4Element* elCo = new G4Element(name="Cobalt", symbol="Co", ncomponents=1);
38 elCo->AddIsotope(Co59, 1);
39
40 //Wood
41 G4Material* Wood = new G4Material(name="Wood", density=0.9*g/cm3, ncomponents=3);
42 Wood->AddElement(C, 2);
43 Wood->AddElement(H, 4);
44 Wood->AddElement(O, 1);
45
46 //PVC
47 G4Material* PVC = new G4Material(name="PVC", density=1.38*g/cm3, ncomponents=3);
48 PVC->AddElement(C, 2);
49 PVC->AddElement(H, 3);
50 PVC->AddElement(Cl, 1);
51
52 //BallsOfSteel
53 G4Material* BallsOfSteel = new G4Material(name="BallsOfSteel", density=7.85*g/cm3, ncomponents=2);
54 BallsOfSteel->AddElement(Fe, 0.98);
55 BallsOfSteel->AddElement(C, 0.02);
56
57
58
59 // Materials assigned to each volume
60 cryo_mat = metalAl;
```

```
world_mat = Air;
holder_mat = Copper;
Pu_mat = Plutonium;
Wood_mat = Wood;
PVC_mat = PVC;
Steel_mat = BallsOfSteel;

```

8.3.5 Analysis Manager

```
#include "QrocoPrimaryGeneratorAction.hh"
#include "QrocoEventData.hh"
#include "QrocoAnalysisManager.hh"
#include <TROOT.h>
#include <TFile.h>
#include <TTree.h>
#include <G4Run.hh>
#include <G4Event.hh>
#include <G4Step.hh>
#include <G4SystemOfUnits.hh>

QrocoAnalysisManager::QrocoAnalysisManager(QrocoPrimaryGeneratorAction* ThePrimaryGeneratorAction)
{
  OutputFileName = "output.root";
  PrimaryGeneratorAction = ThePrimaryGeneratorAction;
  EventData = new QrocoEventData();
}

QrocoAnalysisManager::~QrocoAnalysisManager()
{
}

void QrocoAnalysisManager::BeginOfRun(const G4Run* )
{
  OutputFile = new TFile(OutputFileName.c_str(), "RECREATE");

  G4cout <<"Output file opened."<<G4endl;
  EventTree = new TTree("events", "Simulation data");

  EventTree->Branch("trackId", "vector<int>", &EventData->TrackId);
  EventTree->Branch("type", "vector<string>", &EventData->ParticleType);
  EventTree->Branch("parentid", "vector<int>", &EventData->ParentId);
  EventTree->Branch("xp", "vector<float>", &EventData->X);
  EventTree->Branch("yp", "vector<float>", &EventData->Y);
  EventTree->Branch("zp", "vector<float>", &EventData->Z);
  EventTree->Branch("ed", "vector<float>", &EventData->EnergyDeposited);
  EventTree->Branch("time", "vector<float>", &EventData->Time);
  EventTree->Branch("ekin", "vector<float>", &EventData->KineticEnergy);

  EventTree->Branch("type_pri", "string", &EventData->PrimaryType);
  EventTree->Branch("xPrim", &EventData->PrimaryX, "xPrim/F");
  EventTree->Branch("yPrim", &EventData->PrimaryY, "yPrim/F");
  EventTree->Branch("zPrim", &EventData->PrimaryZ, "zPrim/F");
  EventTree->Branch("ePrim",  &EventData->PrimaryE, "ePrim/F");
  EventTree->Branch("vxPrim", &EventData->PrimaryDirX, "vxPrim/F");
  EventTree->Branch("vyPrim", &EventData->PrimaryDirY, "vyPrim/F");
  EventTree->Branch("vzPrim", &EventData->PrimaryDirZ, "vzPrim/F");

  OutputFile->cd();
}

```

```
52 void QrocoAnalysisManager::EndOfRun(const G4Run* )
53 {
54   OutputFile->cd();
55   OutputFile->Write();
56   OutputFile->Close();
57 }
58
59 void QrocoAnalysisManager::BeginOfEvent(const G4Event* )
60 {
61 }
62
63 void QrocoAnalysisManager::EndOfEvent(const G4Event* )
64 {
65
66   // Info of the primaries
67   EventData->PrimaryType = PrimaryGeneratorAction->GetTypeOfPrimary();
68   EventData->PrimaryX = PrimaryGeneratorAction->GetPositionOfPrimary().x();
69   EventData->PrimaryY = PrimaryGeneratorAction->GetPositionOfPrimary().y();
70   EventData->PrimaryZ = PrimaryGeneratorAction->GetPositionOfPrimary().z();
71   EventData->PrimaryDirX = PrimaryGeneratorAction->GetDirectionOfPrimary().x();
72   EventData->PrimaryDirY = PrimaryGeneratorAction->GetDirectionOfPrimary().y();
73   EventData->PrimaryDirZ = PrimaryGeneratorAction->GetDirectionOfPrimary().z();
74   EventData->PrimaryE = PrimaryGeneratorAction->GetEnergyOfPrimary() / keV;
75
76   EventTree->Fill();
77   EventData->Clear();
78   OutputFile->cd();
79 }
80
81 void QrocoAnalysisManager::Step(const G4Step *TheStep)
82 {
83   if(TheStep->GetTrack()->GetVolume()->GetName().find("Box") != string::npos )
84   {
85     EventData->TrackId->push_back(TheStep->GetTrack()->GetTrackID());
86     EventData->ParticleType->push_back(TheStep->GetTrack()->GetDefinition()->GetParticleName());
87     EventData->ParentId->push_back(TheStep->GetTrack()->GetParentID());
88     EventData->EnergyDeposited->push_back(TheStep->GetTotalEnergyDeposit());
89     EventData->X->push_back(TheStep->GetPostStepPoint()->GetPosition().x());
90     EventData->Y->push_back(TheStep->GetPostStepPoint()->GetPosition().y());
91     EventData->Z->push_back(TheStep->GetPostStepPoint()->GetPosition().z());
92     EventData->Time->push_back(TheStep->GetTrack()->GetGlobalTime());
93     EventData->KineticEnergy->push_back(TheStep->GetPostStepPoint()->GetKineticEnergy());
94   }
95 }
```

8.3.6 Event Data

This file is needed to give the executable all the different parameters we need for the analysis.

```
#include "QrocoEventData.hh"

QrocoEventData::QrocoEventData()
{
  TrackId = new vector<int>;
  ParticleType = new vector<string>;
  ParentId = new vector<int>;
        X = new vector<float>;
        Y = new vector<float>;
        Z = new vector<float>;
        EnergyDeposited = new vector<float>;
  Time = new vector<float>;
        KineticEnergy = new vector<float>;

  PrimaryType = "";
  PrimaryDirX = 0.;
  PrimaryDirY = 0.;
  PrimaryDirZ = 0.;
  PrimaryE = 0.;
}

QrocoEventData::~QrocoEventData()
{
  delete TrackId;
  delete ParticleType;
  delete ParentId;
  delete X;
  delete Y;
  delete Z;
  delete EnergyDeposited;
  delete Time;
  delete KineticEnergy;
}

void QrocoEventData::Clear()
{
  TrackId->clear();
  ParticleType->clear();
  ParentId->clear();
  X->clear();
  Y->clear();
  Z->clear();
  EnergyDeposited->clear();
  Time->clear();
  KineticEnergy->clear();
  PrimaryType.clear();
  PrimaryX = 0.;
  PrimaryY = 0.;
  PrimaryZ = 0.;
  PrimaryE = 0.;
```

```
  PrimaryDirX = 0.;
  PrimaryDirY = 0.;
  PrimaryDirZ = 0.;
}
```

8.3.7 Macro file

This file is needed to run the simulation with a visual representation.

```
#################
#VERBOSITY
/control/verbose 0
/run/verbose 0
/event/verbose 0
/tracking/verbose 0
/qroco/gun/verbose 0

#################
#SEED
/run/random/setRandomSeed 0

#################
#Visualization
#/vis/open Qt
#/vis/drawVolume


# Select the volume to confine the source. For cryostat studies choose:

#/qroco/gun/confine Cyl*
#/qroco/gun/confine Platform*
#/qroco/gun/confine Dings*
#/qroco/gun/confine PC*
#/qroco/gun/confine Table*
#/qroco/gun/confine PPMSC*
#/qroco/gun/confine PPMSP*
#/qroco/gun/confine Wall*

##################
/qroco/gun/particle gamma
/qroco/gun/energy 662 keV
/qroco/gun/position -1.156 3.37 -0.60 m
/qroco/gun/direction 0 0 1

#/run/printProgress 1
```

8.3.8 Analysis.C

This file is needed for the analysis of the output.root file created by the simulation.

```
#include <TFile.h>
#include <TTree.h>
#include <TF1.h>
#include <iostream>
#include <chrono>
#include <vector>
#include <algorithm>

using namespace std::chrono;

void analysis() {

    TH1F *TID = new TH1F("Values", "Track ID", 40, 0, 40);
    TID->SetFillColor(kBlue-8);
    TH1F *PosX = new TH1F("Values_X", "Position X", 75, -1185, -1125);
    PosX->SetFillColor(kBlue-8);
    PosX->GetXaxis()->SetTitle("Position [mm]");
    PosX->GetYaxis()->SetTitle("Counts");
    TH1F *Ener = new TH1F("Values", "Energy", 75, 0.0005, 0.003);
    Ener->SetFillColor(kBlue-8);
    Ener->GetXaxis()->SetTitle("Energy [keV]");
    Ener->GetYaxis()->SetTitle("Counts");
    TH1F *PosY = new TH1F("Values_Y", "Position Y", 75, 3340, 3400);
    PosY->SetFillColor(kBlue-8);
    PosY->GetXaxis()->SetTitle("Position [mm]");
    PosY->GetYaxis()->SetTitle("Counts");
    TH1F *PosZ = new TH1F("Values_Z", "Position Z", 75, -380, -330);
    PosZ->SetFillColor(kBlue-8);
    PosZ->GetXaxis()->SetTitle("Position [mm]");
    PosZ->GetYaxis()->SetTitle("Counts");
    TH1F *Ekin = new TH1F("Energy Values", "Kinetic Energy", 75, 0.01, 0.5);
    Ekin->SetFillColor(kBlue-8);
    Ekin->GetXaxis()->SetTitle("Energy [keV]");
    Ekin->GetYaxis()->SetTitle("Counts");

    const char* fileName = "output.root";

    // Open the ROOT file
    TFile* file = TFile::Open(fileName);
    if (!file || file->IsZombie()) {
        std::cerr << "Error opening file: " << fileName << std::endl;
        return;
    }

    // Access the TTree in the file
    TTree* tree = dynamic_cast<TTree*>(file->Get("events"));
    if (!tree) {
        std::cerr << "Error accessing TTree in file: " << fileName << std::endl;
        file->Close();
        return;
```

```
    }
    auto start = high_resolution_clock::now();
    // Set branch addresses
    std::vector<int> *trackId = nullptr;
    std::vector<float> *ener = nullptr;
    std::vector<int> *posx = nullptr;
    std::vector<int> *posy = nullptr;
    std::vector<int> *posz = nullptr;
    std::vector<float> *ekin = nullptr;
    tree->SetBranchAddress("trackId", &trackId);
    tree->SetBranchAddress("ed", &ener);
    tree->SetBranchAddress("xp", &posx);
    tree->SetBranchAddress("yp", &posy);
    tree->SetBranchAddress("zp", &posz);
    tree->SetBranchAddress("ekin", &ekin);

    // Initialize variables
    std::vector<int> uniqueTracks;
    int counter = 0;
    int currentTrack = -1;
    float Totalenergydeposited = 0.0;

    // Loop over entries in the tree
    for (Long64_t i = 0; i < tree->GetEntries(); ++i) {
        tree->GetEntry(i);
        float PartialEnergy = 0.0;


        for (int k = 0; k < trackId->size(); ++k) {
            PartialEnergy += ener->at(k);
            Ener->Fill(PartialEnergy);
        }


        // Loop over the 'trackId' values
        for (int j = 0; j < trackId->size(); ++j) {
            // Check if the track is already in the uniqueTracks vector
            if (std::find(uniqueTracks.begin(), uniqueTracks.end(), currentTrack) != uniqueTracks.end()) {
                continue;  // Track already encountered, skip to the next one
            }

            TID->Fill(trackId->at(j));
            PosX->Fill(posx->at(j));
            PosY->Fill(posy->at(j));
            PosZ->Fill(posz->at(j));
            Ekin->Fill(ekin->at(j));

            // Track not found, add it to the uniqueTracks vector
            uniqueTracks.push_back(currentTrack);
        }

        // Update the counter based on the number of unique tracks in this event
        counter += uniqueTracks.size();

```

```
        uniqueTracks.clear();

        Totalenergydeposited += PartialEnergy;
    }

    auto stop = high_resolution_clock::now();
    auto duration = duration_cast<milliseconds>(stop - start);

    cout << "--------------------------------------------" << endl;
    // Print the time taken
    cout << "Time taken by function: "
         << duration.count() << " milliseconds" << endl;


    // Print the result
    std::cout << "The total number of unique particles is: " << counter << std::endl;

    std::cout << "The total energy deposited is: " << Totalenergydeposited <<  "keV" << std::endl;

    // Close the file
    file->Close();
    TCanvas *c1 = new TCanvas();
    TID->Draw();
    TCanvas *c2 = new TCanvas();
    PosX->Draw();
    TCanvas *c3 = new TCanvas();
    Ener->Draw();
    TCanvas *c4 = new TCanvas();
    PosY->Draw();
    TCanvas *c5 = new TCanvas();
    PosZ->Draw();
    TCanvas *c6 = new TCanvas();
    Ekin->Draw();
}
```

8.4 Simulation

As discussed in chapter 4.9, several simulations have been conducted. Given that a larger number of particles yields a more refined simulation, only the simulations involving 5,000,000 and 10,000,000 particles have been examined in detail in the chapter. The results from the simulations involving fewer particles will be presented in this section.

8.4.1 100'000 particles

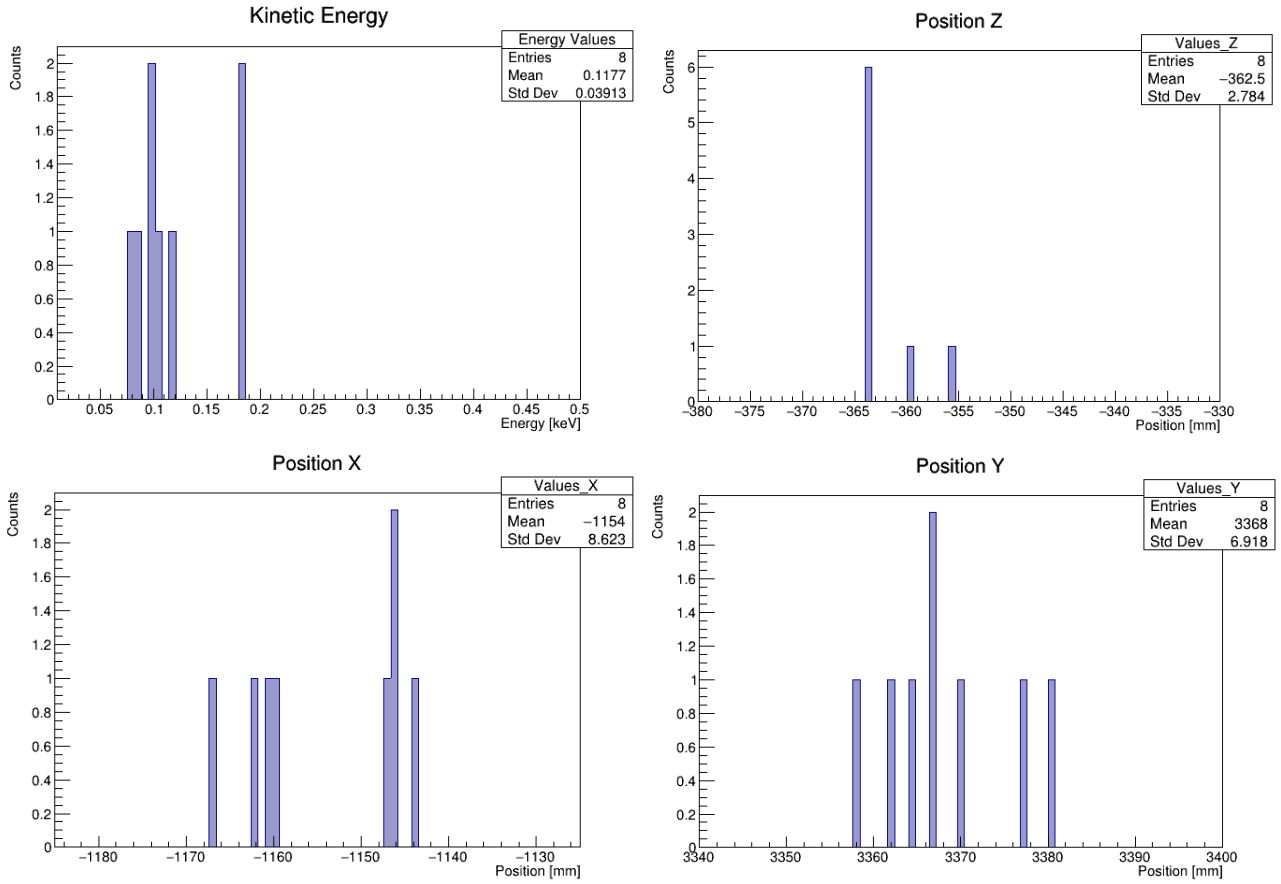


Figure 59: Simulation results for 100'000 particles

8.4.2 500'000 particles

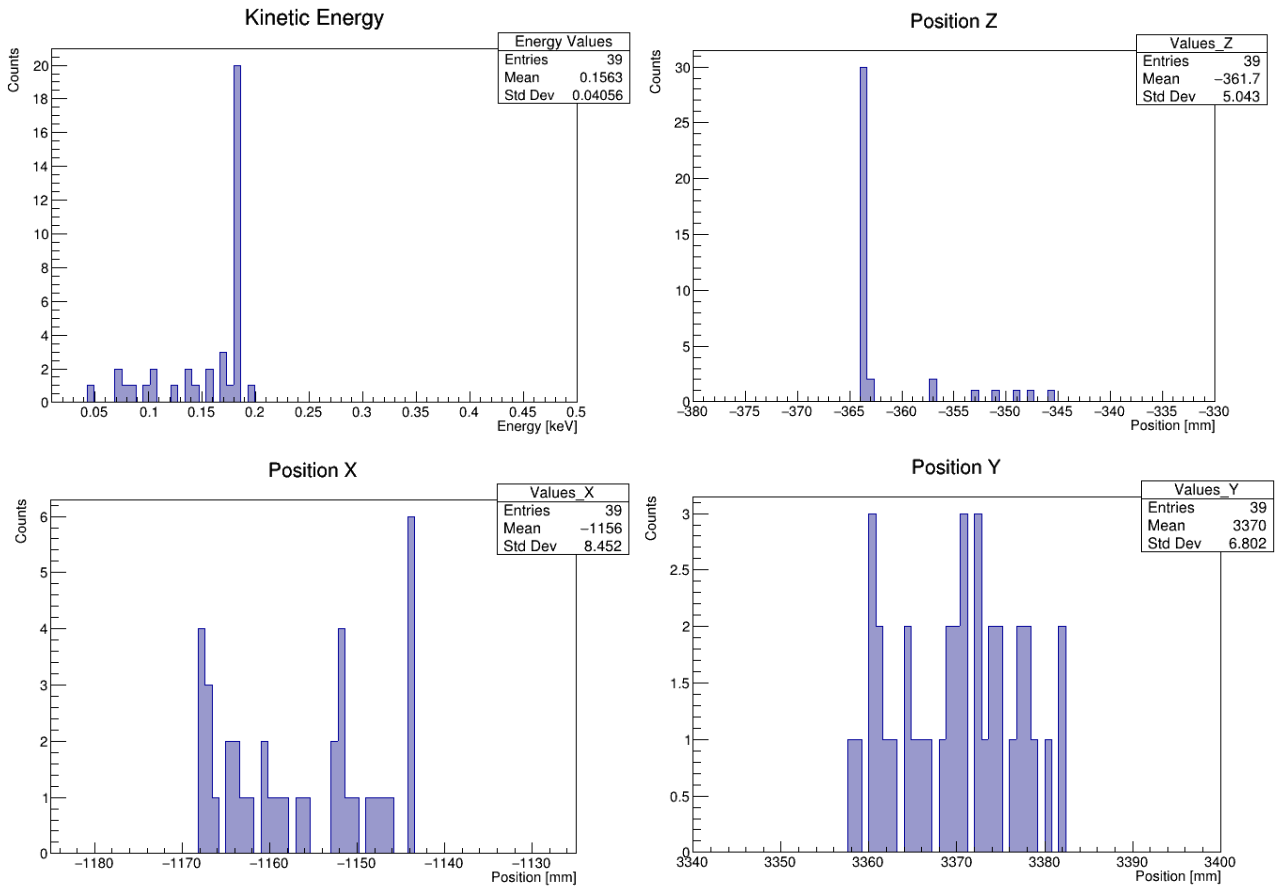


Figure 60: Simulation results for 500'000 particles

8.4.3 2'000'000 particles

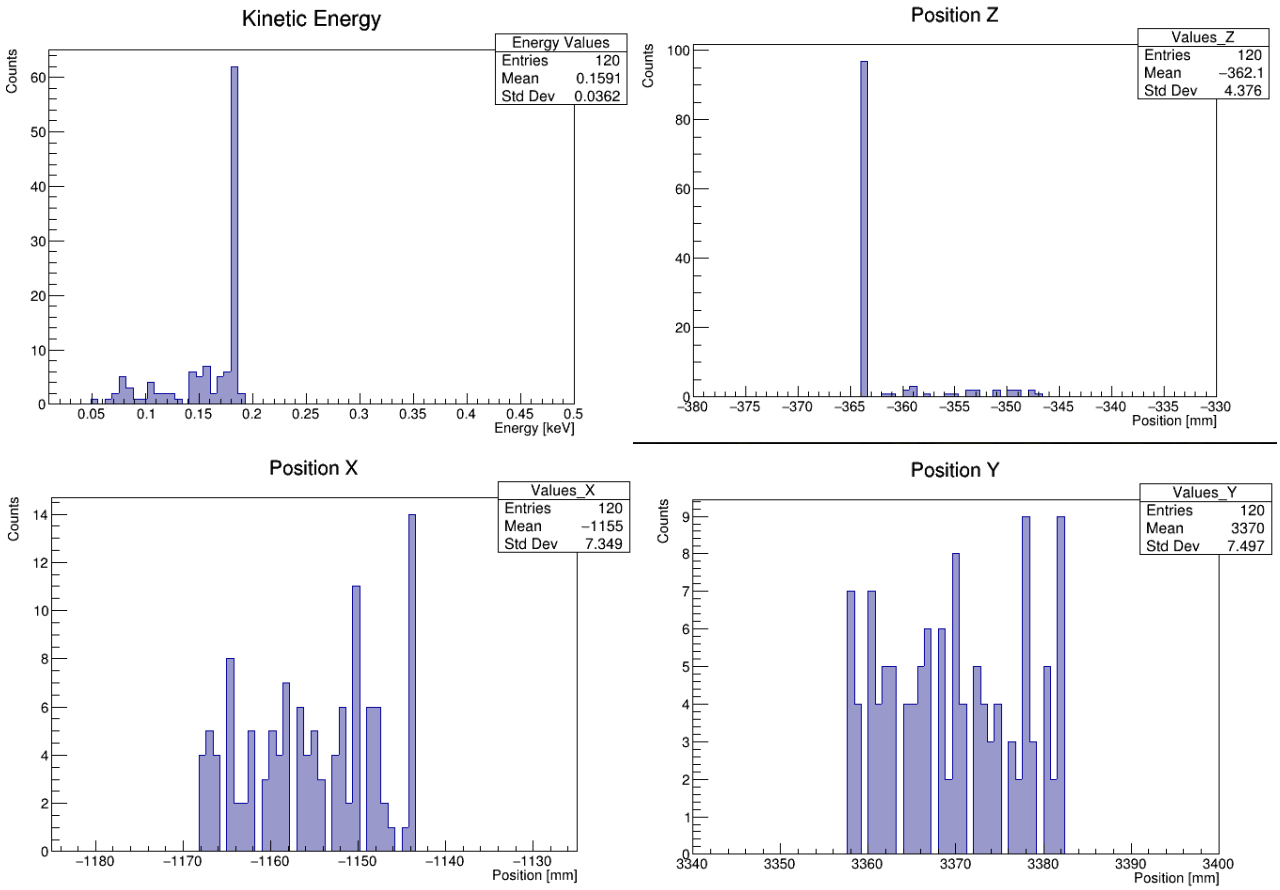


Figure 61: Simulation results for 2'000'000 particles

8.5 Setup for the QCL installation

As stated in chapter 6.4, a setup for the installation was prepared. The setup as a written "manual" is attached here.

Setup for the QCL laser

Noah Brugger and Severin Nägeli

July 24, 2023

1 Needed components

- Quantum Cascade Laser (QCL) Controllers, Current and TEC (Already in lab)
- Quantum Cascade Lasers (QCLs): Distributed Feedback, Two-Tab C-Mount
- C-Mount Laser Mount with Temperature Stabilization
- IR neutral density filter, OD 3.0; 12.5 mm diam

2 Assembling

2.1 Overview

Figure 1: Our mount for the QCL itself, contains the mounting platform and multiple anode / cathode inputs.

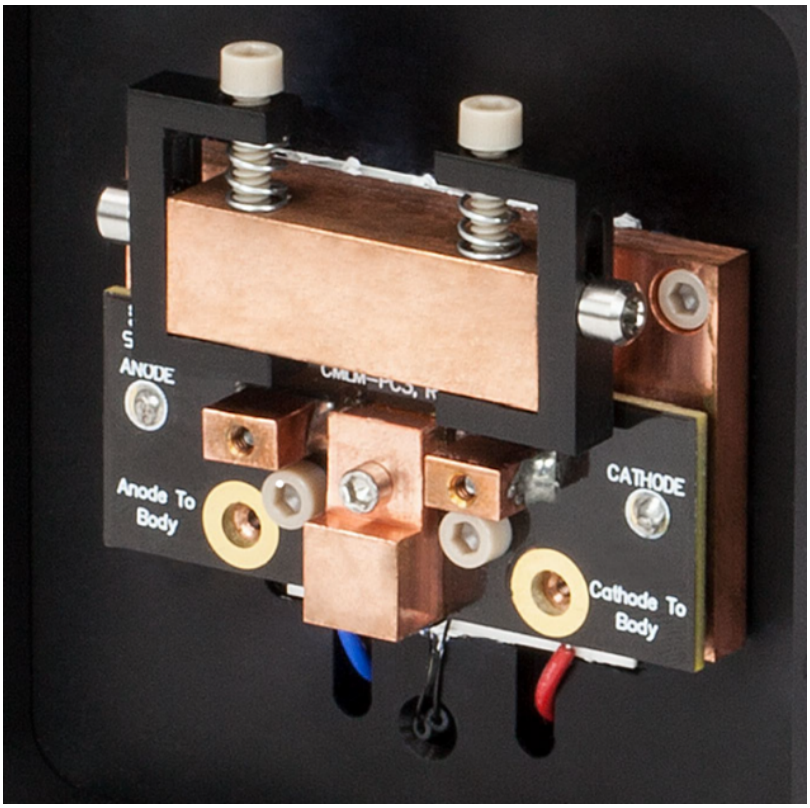


Figure 2: Larger picture of the inside of our mount: By unscrewing the 4 screws seen at the front of the first picture, we unravel this view. In the middle part we mount the QCL (QD4500CM1), as seen on the next picture. The two "legs" of the QCL need to be placed on top of those small pieces of copper.

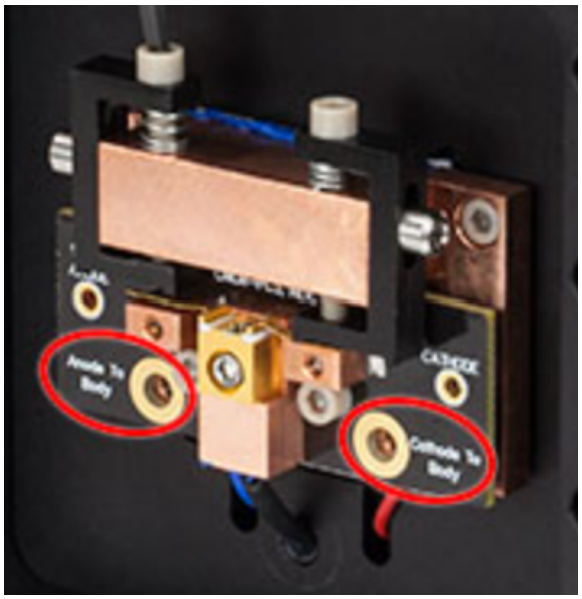


Figure 3: Clearer look at the positioning of the laser itself.

Figure 4: On this picture we see one side of the C-Mount with the different connectors (Included with the 4000 series of our laser controller).

Figure 5: CAB4005 cable included with the laser controller, which connects to the diode mount.

2.2 Assemble the laser

At first, one needs to attach the laser itself to the mount, as described in section 1. Further one needs to connect the mount with the source, in our case the ITC4020 by THORLABS. This needs to be connected to a power outlet, and the other cable is described in 5. Since this results in a problem with the understanding, I will talk about this in section 3.
If everything goes well until now, we should have a laser output now at the one end of the mount, see 1. This laser needs to be collected by a sensor in our black box in the lab, which can transform it into a readable and transmittable signal for the cable, which leads to the cryostat at the top.
At the top of the cryostat, we connect the cable to the inside of the cryostat, where we attached our holder, see the picture below.

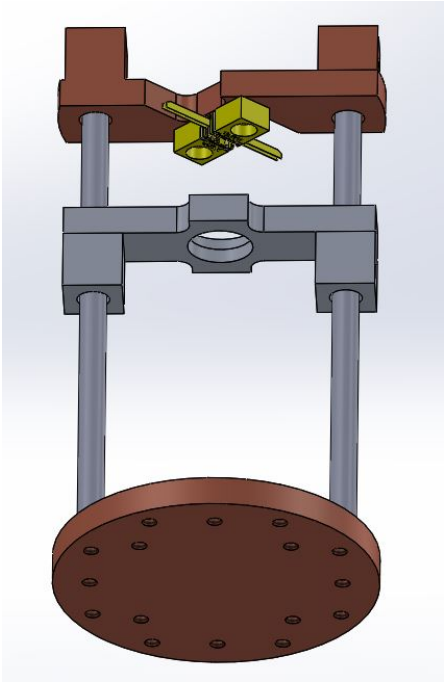


Figure 6: SolidWorks Construction of the holder with two QCLs installed.

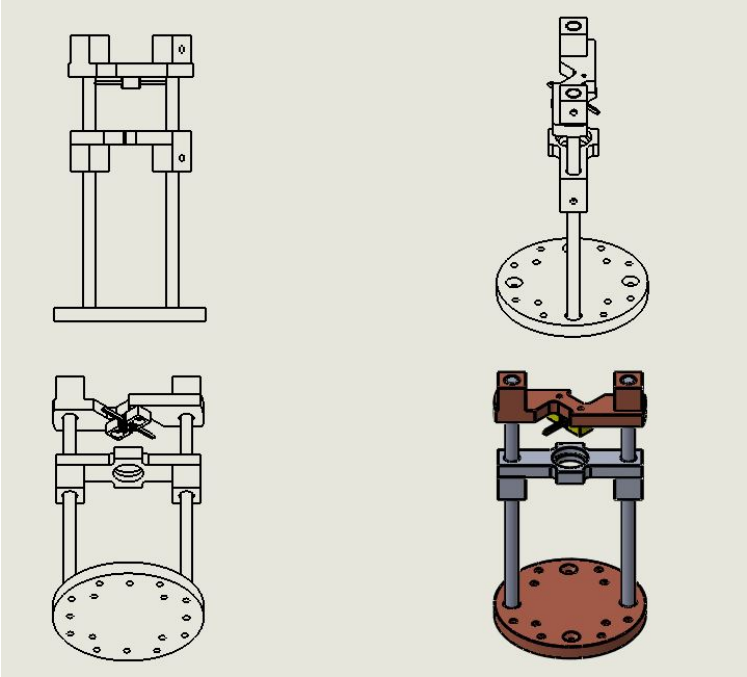
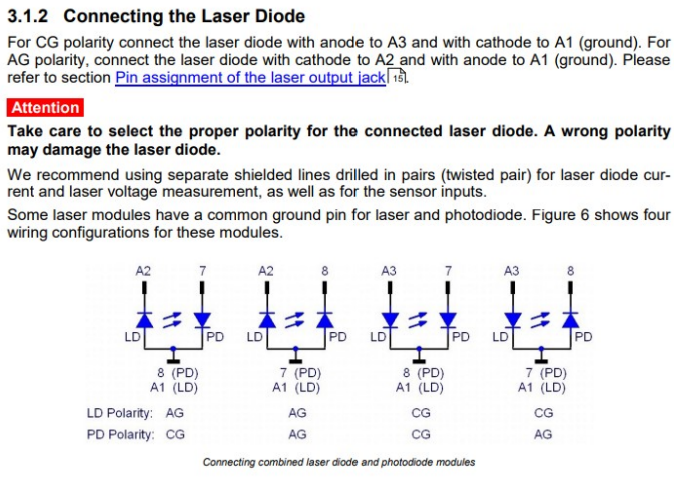


Figure 7: Drawing of the QCL setup shown in figure 6

3 Problems and difficulties

As discussed in section 2, we have a few problems we have to discuss. The first one is regarding the polarity of the connected laser diode.



3.1.2 Connecting the Laser Diode

For CG polarity connect the laser diode with anode to A3 and with cathode to A1 (ground). For AG polarity, connect the laser diode with cathode to A2 and with anode to A1 (ground). Please refer to section Pin assignment of the laser output jack 15.

Attention

Take care to select the proper polarity for the connected laser diode. A wrong polarity may damage the laser diode.

We recommend using separate shielded lines drilled in pairs (twisted pair) for laser diode current and laser voltage measurement, as well as for the sensor inputs.

Some laser modules have a common ground pin for laser and photodiode. Figure 6 shows four wiring configurations for these modules.

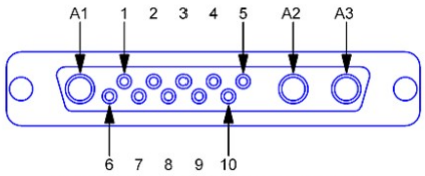
Connecting combined laser diode and photodiode modules

Figure 8: Picture taken from the manual of the ITC4000, found on page 16

For different polarities, we need a different setup and different connections of our pins. We have two types:

- CG polarity: Anode to A3, Cathode to A1. which is the ground
- AG polarity: Cathode to A2, Anode to A1.

To receive further understanding, I include here the pin assignment of the ITC4000, as seen in 9. The



Pin assignment of the 13W3 DSUB laser output jack (rear panel view)

Pin	Connection
	Interlock and Status Indicator:
5	Output for interlock and status indicator "LASER ON/OFF" (+)
6	Ground pin for interlock and status indicator "LASER ON/OFF" (-)
	Monitor Input / Power Feedback Source:
1	(Thermo) voltage sensor input (+)
2	(Thermo) voltage sensor ground (-)
7	Photo current sensor input
8	Photo current sensor ground
	Laser Voltage Measurement:
4	Laser diode anode (+)
10	Laser diode cathode (-)
	Laser Diode:
A1	Laser diode ground
A2	Laser diode cathode (with polarity AG) (-)
A3	Laser diode anode (with polarity CG) (+)
3, 9	Not connected

Figure 9: Picture taken from the manual of the ITC4000, found on page 15

last point which is not clear, is the four connections to the cathode and anode seen in 3. Two of them need to be connected to the Cryostat, and the other two to something different, but it is not clear which pair to where. To end this first sketch, I want to include a short hand drawn sketch of the whole setup, just to give an overview of all the components and some parts which are not clear, see 10.

4 Operating the ITC4000

- Turn on by pressing F2; Wait 30min until accurancy is constant.
- Connect Mount with ITC4000 via Cable CAB4005, which results in a beam of 5A.
- By navigating to the laser diode (LD) output configuration menu by pressing the function keys F4, you can choose the polarity, anode grounded (AG) or cathode grounded (CG). Choosing the right polarity is fundamental and may damage your system.
- Since we use the QCL4002QCL, we cover a range for the modulation frequency of 20 to 130Hz.

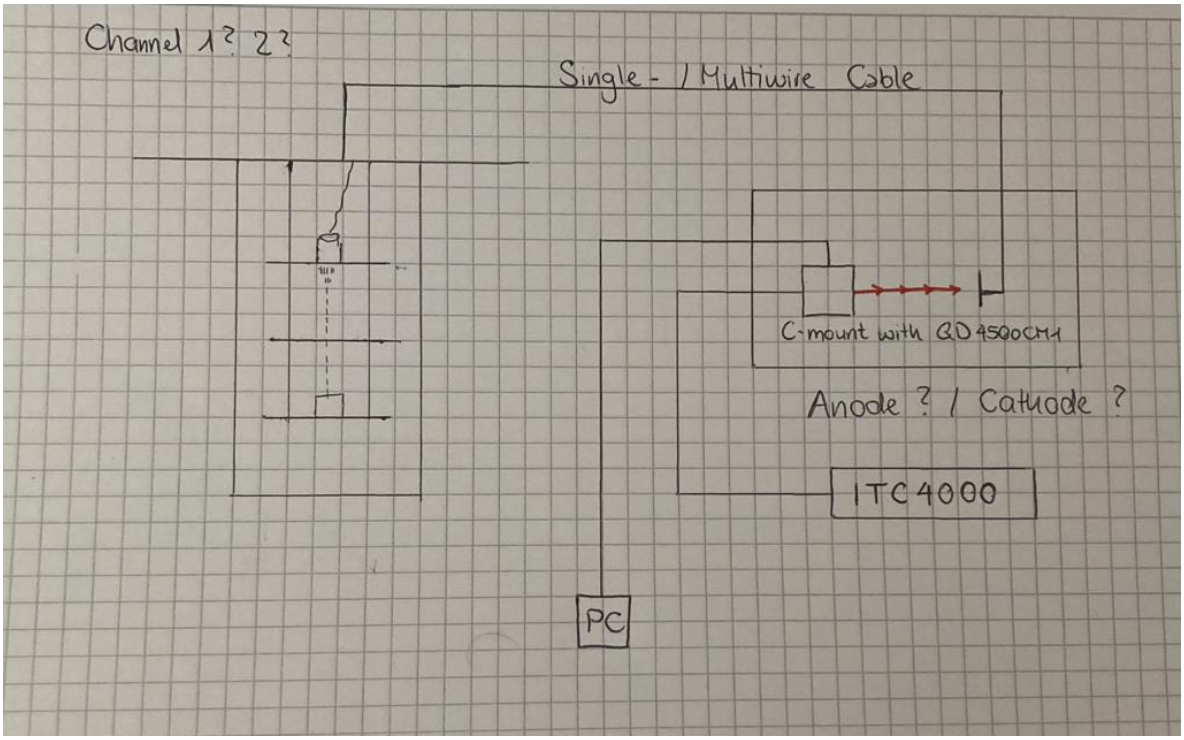


Figure 10: hand drawn sketch

8.6 Weekly tasks in the lab

8.6.1 Manual for the lab work

For this project, we aimed to test multiple samples with varied structures. This required us to frequently open and close the Cryostat in the laboratory at the University of Zurich. Given that the cooling-down process includes several crucial steps to guarantee the success of the cooling, we decided to write a concise yet explanatory manual for operating the Cryostat. While an official, detailed manual exists, our guide focuses on the most important steps. As an additional bonus, this includes the startup process for the NKT Photonics SuperK Extreme laser source. This simplified guide, compiled during the initial weeks of the project, was very helpful for understanding the operating principles of such a laser source.

Universität Zürich[UZH]

Cooling down and warming up the Cryostat

Manual for using the Cryostat: Cooling Down and Warming up

Using the Laser Source

A manual for using the SuperK EXTREME Laser source and its component
and for the use of the monochromator

A manual written for the public use of
the lab members

Department of Physics
University of Zurich
Switzerland
2023
Version 1.21

1 Cryostat

1.1 Warm-up

- **STOP** the program on the OXFORD- IO Computer next to the Cryostat
  → Name of the program: "Cooling_Down_Oxford_dilution_refrigerator_Final.vi"
- **Turn off** PTR pump
- Follow the next few steps regarding the valves:
  1. **Close** Valve #3
  2. **Open** Valve #4
  3. **Close** Valve #6
  4. **Open** Valve #8
  5. **Close** Valve #7
  6. **Open** Valve #9
- **Wait** until Temperature on Channel 3 and 4 reach 5K (Check *Lakeshore Cryogenics* Monitor)
- Once the temperature is reached: **Close** all valves #1-#9.
- **Stop** both pumps, on the left and right of the PTR pump, see figure 1.

Figure 1: Setup of the pumps

- *Open* program on Computer: "Warm_Up_Oxford_dilution_refrigerator.vi"
- **Check** if all settings match with those on fig 2.

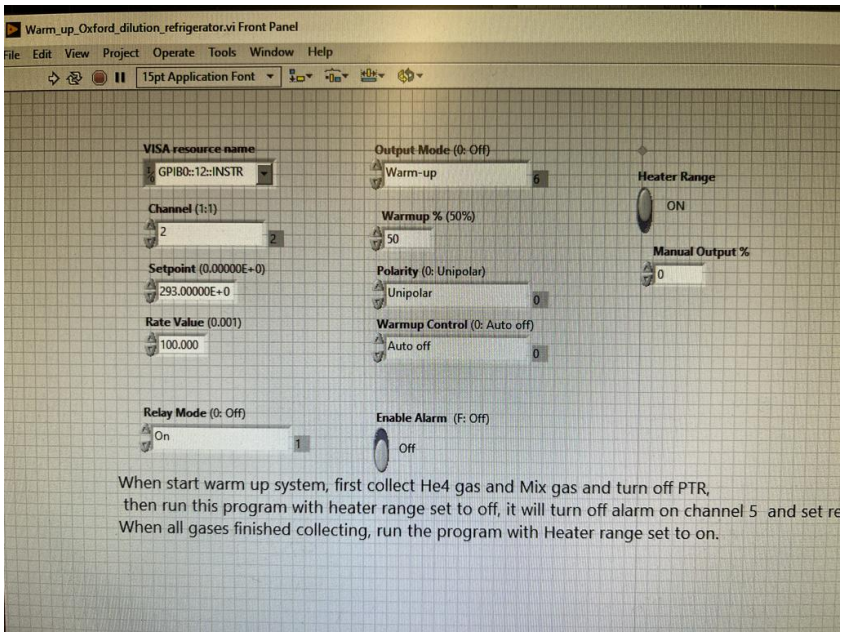


Figure 2: Final state of settings regarding the warm- up

- **Run** the program with the black arrow on the top left. Arrow changes to white if successfully started.
- **Check** if the red light on the *Lakeshore Cryogenics* monitor is on → Heater is on.
- **Check** if set point is set to 293K and if heater is "50% of on".

1.2 Open Cryostat

- Look at the display and **check** if the temperature has around 300K (Room temperature)
- **Open** the black valve at the top of the cryostat until one can see the red mark.
- **Open** slowly the black valve behind the cryostat, above the pump.
- **IMPORTANT**: Wear gloves for all the following steps!
- **Open** the first layer of the cryostat itself, keep care of all the screws.
- **Unscrew** all screws until point of desire.
- If needed, **check** connection of single mode/multi mode fiber and laser.

1.3 Close cryostat

- **Check** if smaller black valve behind Cryostat is closed.
- **Check** if upper black valve is open until the red mark.
- **Mount** all the correct layers of the cryostat in the correct order.
- **Turn on** the pump at the back via the green button.
- **Switch** on display to program 23 and set it to **OFF** by pressing both buttons at the same time.
- **Start** pump by pressing the most right button.
- **Wait** for about one hour.
- **Turn on** motor pump on program 23 by pressing both buttons at the same time and **switch** to **ON**.
- **Check** if the rotation speed is 1500Hz.
- **Wait** over night.
- **Check** if the pressure is around 10^{-3} hPa.

1.4 Cooling down the Cryostat

- **Check** if the vacuum can is evacuated and leak tested.
- The 1K pot circuit should be set to circulate:
  - **Open** valve #1,2,3.
  - **Close** valve #4.

  - **Press** left button labelled with **PUMP**. It should illuminate with a blue light. See figure 1.

- The dilution circuit should be set to circulate:
  - **Open** valve #5,6,7.
  - **Close** valve #8,9.
  - **Press** right button labelled with **PUMP**. It should illuminate with a blue light. See 1.

- The Lakeshore AC 372 should be configured:
  - **Open** Program "Cooling_Down_Oxford_dilution_refrigerator_Final.vi"
  - **Switch** "Heater warm?" to ON/GREEN.

- The pulse tube cooler should be switched on:
  - **Press** button in the middle named PTR. It should illuminate with a blue light. See Figure 1.

2 Laser

2.1 Starting the laser source

- **Start** laser source on the back, see fig 3. Check if the green light in the upper right corner is on, marked with a black circle on fig. 5

Figure 3: Location of on- off switch

- **Switch** the key from OFF to ON, marked with green circle on fig. 5.
- **Press** the RESET INTERLOCK button, marked with a blue circle on fig. 5.

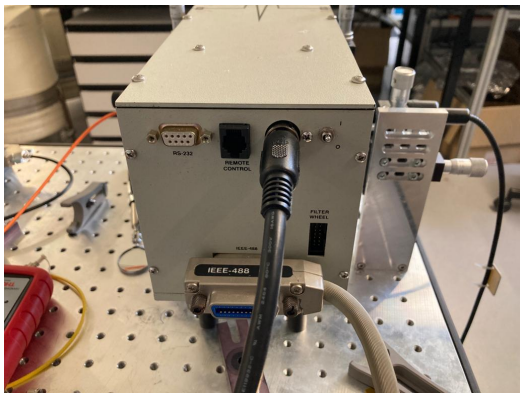


Figure 4: Backside of the monochromator

- **Turn on** the monochromator on the optical table. The switch is on the backside. See fig. 4.
- **Switch** to Computer and **open** the following two programs:
  1. SUPERKONTROL: Program used to control the laser unit and its emission rate. With this program one can change the power of the emitted laser beam, the values are possible from 0 to 100%.
  2. MonoUtility: Program used to control the monochromator and its emitted wavelength. The wavelength can be chosen to be any arbitrary value.
- BEFORE TURNING ON THE LASER EMISSION **check** if all cables are connected to the Cryostat correctly.
- **Start** the laser emission on the Computer by pressing the EMISSION button in SUPERKONTROL.

2.2 Shutting down the laser

- To shut down the laser, **press** first the EMISSION button in SUPERKONTROL.
- **Switch** the key from off to on, see green circle on fig. 5.
- **Turn off** the laser source via the back button, see red circle on fig. 3.
- **Turn off** the monochromator via the switch on the back side, see fig. 4.

Figure 5: Front side of the laser source

8.6.2 Nitrogen refill

Refilling nitrogen in the lab devices is an important task as it helps maintain the low temperatures necessary for certain experiments. Specifically, the Physical Property Measurement System (PPMS) for the chemistry and physics departments, and the dilution refrigerator system used for cooling down the Cryostat.
When refilling nitrogen, it's important to handle the liquid nitrogen with care due to its extremely cold temperature. You should always use appropriate protective equipment such as gloves to prevent injuries caused by the rapid evaporation and extremely cold temperature of liquid nitrogen.
Also, it's crucial to monitor the nitrogen level of these devices regularly and maintain them appropriately. This ensures that the lab devices are always ready for use and that the experiments run smoothly and successfully.

List of Figures

List of Tables

References

[1]V. Trimble, 'Existence and Nature of Dark Matter in the Universe', Annual Review of Astronomy and Astrophysics **25**, 425–472 (1987).

[2]F. Zwicky, 'Republication of: The redshift of extragalactic nebulae', General Relativity and Gravitation **41**, 207–224 (2009).

[3]A. Doroshkevich, V. N. Lukash and E. Mikheeva, 'A solution to the problems of cusps and rotation curves in dark matter halos in the cosmological standard model', Uspekhi Fizicheskih Nauk **182**, 3 (2012).

[4]C. Sivaram and V. M. R. A, 'Dark Matter, Dark Energy and Rotation Curves', (2007).

[5]J. Pinochet and M. V. S. Jan, 'Einstein ring: Weighing a star with light', `10.1088/1361-6552/aac7b9` (2017).

[6]R. Massey, T. Kitching and J. Richard, 'The dark matter of gravitational lensing', `10.1088/0034-4885/73/8/086901` (2010).

[7]The XENON collaboration et al., 'Projected WIMP Sensitivity of the XENONnT Dark Matter Experiment', `10.1088/1475-7516/2020/11/031` (2020).

[8]N. Du et al., 'Search for Invisible Axion Dark Matter with the Axion Dark Matter Experiment', Physical Review Letters **120**, 151301 (2018).

[9]R. D. Peccei and H. R. Quinn, 'Conservation in the Presence of Pseudoparticles', Physical Review Letters **38**, 1440–1443 (1977).

[10]L. Bergstrom, 'Dark Matter Candidates', `10.1088/1367-2630/11/10/105006` (2009).

[11]S. P. MARTIN, 'A SUPERSYMMETRY PRIMER', in (July 1998), pp. 1–98.

[12]R. Combescot, *Superconductivity* (Cambridge University Press, Mar. 2022).

[13]K. Onnes, *Further experiments with liquid helium. C. On the change of electric resistance of pure metals at very low temperatures etc. IV. The resistance of pure mercury at helium temperatures*, tech. rep. (1911).

[14]R. P. Huebener, 'The Path to Type-II Superconductivity', Metals **9**, 682 (2019).

[15]W. Meissner and R. Ochsenfeld, 'Ein neuer Effekt bei Eintritt der Supraleitfähigkeit', Die Naturwissenschaften **21**, 787–788 (1933).

[16]R. Jackson, 'John Tyndall and the Early History of Diamagnetism', Annals of Science **72**, 435–489 (2015).

[17]F. London, *Superfluids*, 2nd ed., Vol. 1 (Dover Publications, Inc., 1961).

[18]S. F. Saipuddin, A. Hashim and N. E. Suhaimi, 'Type I and Type II Superconductivity', in *Superconducting materials* (Springer Nature Singapore, Singapore, 2022), pp. 123–146.

[19]M. M. A. Kechik, *Effect of Yb2O3 Nanoparticle Addition on Superconducting Properties of BSCCO (2223)/Ag Tapes by Acetate Precipitation Method Article in Pertanika*, tech. rep. 1 (2016).

[20]L. N. Cooper, 'Bound Electron Pairs in a Degenerate Fermi Gas', Physical Review **104**, 1189–1190 (1956).

[21]J. Bardeen, L. N. Cooper and J. R. Schrieffer, 'Microscopic Theory of Superconductivity', Physical Review **106**, 162–164 (1957).

[22]M. Tinkham, *Introduction to Superconductivity* (Dover, 1996), p. 63.

[23]Y. Hochberg, Y. Zhao and K. M. Zurek, 'Superconducting Detectors for Super Light Dark Matter', `10.1103/PhysRevLett.116.011301` (2015).

[24]Y. Hochberg, M. Pyle, Y. Zhao and K. M. Zurek, 'Detecting Superlight Dark Matter with Fermi-Degenerate Materials', `10.1007/JHEP08(2016)057` (2015).

[25]M. E. Grein, O. Shatrovoy, D. Murphy, B. S. Robinson and D. M. Boroson, 'A Multimode Fiber-coupled Photon-counting Optical Receiver for the Lunar Laser Communication Demonstration', in Cleo: 2014 (2014), SM4J.4.

[26]J.-i. Yamada et al., 'New Organic Superconductors β-(BDA-TTP) ₂ X [BDA-TTP = 2,5-Bis(1,3-dithian-2-ylidene)-1,3,4,6-tetrathiapentalene; X ⁻ = SbF ₆ ⁻ , AsF ₆ ⁻ , and PF ₆ ⁻]', Journal of the American Chemical Society **123**, 4174–4180 (2001).

[27]L. R. Testardi, 'Destruction of Superconductivity by Laser Light', Physical Review B **4**, 2189–2196 (1971).

[28]G. N. Gol'tsman et al., 'Picosecond superconducting single-photon optical detector', Applied Physics Letters **79**, 705–707 (2001).

[29]A. M. Kadin and M. W. Johnson, 'Nonequilibrium photon-induced hotspot: A new mechanism for photodetection in ultrathin metallic films', Applied Physics Letters **69**, 3938–3940 (1996).

[30]I. Charaev, 'Improving the Spectral Bandwidth of Superconducting Nanowire Single-Photon Detectors (SNSPDs)', PhD thesis (Karlsruher Institut für Technologie, Karlsruhe, 2017).

[31]L. You, 'Superconducting nanowire single-photon detectors for quantum information', Nanophotonics **9**, 2673–2692 (2020).

[32]P. S. Kuo, 'Using silica fiber coupling to extend superconducting nanowire single-photon detectors into the infrared', OSA Continuum **1**, 1260 (2018).

[33]T. Kogure and K. Leung, *Thermodynamic equilibrium and black-body radiation* (Springer, 2007), p. 41.

[34]I. Esmaeil Zadeh et al., 'Superconducting nanowire single-photon detectors: A perspective on evolution, state-of-the-art, future developments, and applications', Applied Physics Letters **118**, 10.1063/5.0045990 (2021).

[35]A. J. Annunziata et al., 'Reset dynamics and latching in niobium superconducting nanowire single-photon detectors', Journal of Applied Physics **108**, 084507 (2010).

[36]Y. Hochberg et al., 'New Constraints on Dark Matter from Superconducting Nanowires', (2021).

[37]Y. Hochberg et al., 'Detecting Dark Matter with Superconducting Nanowires', 10.1103/PhysRevLett.123.151802 (2019).

[38]T. Polakovic, W. Armstrong, G. Karapetrov, Z.-E. Meziani and V. Novosad, 'Unconventional Applications of Superconducting Nanowire Single Photon Detectors', Nanomaterials **10**, 1198 (2020).

[39]C. E. Shannon, 'Communication Theory of Secrecy Systems', Bell System Technical Journal **28**, 656–715 (1949).

[40]C. M. Natarajan, M. G. Tanner and R. H. Hadfield, 'Superconducting nanowire single-photon detectors: physics and applications', Superconductor Science and Technology **25**, 063001 (2012).

[41]M. McCord and J. Rooks, *SPIE Handbook of Microlithography, Micromachining and Microfabrication* (2000).

[42]J. Li, U. Parchatka, R. Königstedt and H. Fischer, 'Real-Time Measurements of Atmospheric CO Using a Continuous-Wave Room Temperature Quantum Cascade Laser Spectrometer', in 2012 symposium on photonics and optoelectronics (May 2012), pp. 1–4.

[43]V. Verma et al., 'Recent advances in superconducting nanowire single-photon detector technology for exoplanet transit spectroscopy in the mid-infrared', Journal of Astronomical Telescopes, Instruments, and Systems (2021).

[44]Y. Pan et al., 'Mid-infrared Nb ₄ N ₃ -based superconducting nanowire single photon detectors for wavelengths up to 10 µm', Optics Express **30**, 40044 (2022).

[45]A. Banerjee et al., 'Characterisation of amorphous molybdenum silicide (MoSi) superconducting thin films and nanowires', Superconductor Science and Technology **30**, 084010 (2017).

[46]A. E. Siegman, *LASERs*, 7th ed. (1986).

[47]A. Einstein, 'Strahlungs-Emission und Absorption nach der Quantentheorie', Deutsche Physikalische Gesellschaft **18**, 318–323 (1916).

[48]P. Dirac, 'The quantum theory of the emission and absorption of radiation', Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character **114**, 243–265 (1927).

[49]J. Faist et al., 'Quantum Cascade Laser', Science **264**, 553–556 (1994).

[50]R. Sharma, H. Kaur and M. Singh, 'Recent Advances of Efficient Design of Terahertz Quantum-Cascade Lasers', Plasmonics **16**, 449–461 (2021).

[51]T. P. Pearsall, *Quantum Photonics* (Springer International Publishing, Cham, 2020).

[52]M. Darman and K. Fasihi, 'Three-level rate-equations-based model of quantum cascade lasers with a single solution regime', Optical and Quantum Electronics **49**, 110 (2017).

[53]P. Mena, Sung-Mo Kang and T. DeTemple, 'Rate-equation-based laser models with a single solution regime', Journal of Lightwave Technology **15**, 717–730 (1997).

[54]A. Hamadou, 'Analytical investigation of the dynamics behaviors of quantum cascade laser', Optics Communications **335**, 271–278 (2015).

[55]D. W. Koon and C. J. Knickerbocker, 'What do you measure when you measure resistivity?', Review of Scientific Instruments **63**, 207–210 (1992).

[56]L. J. van der PAUW, 'A METHOD OF MEASURING SPECIFIC RESISTIVITY AND HALL EFFECT OF DISCS OF ARBITRARY SHAPE', in *Semiconductor devices: pioneering papers* (WORLD SCIENTIFIC, Mar. 1991), pp. 174–182.

[57]Y. Geng, 'Mathematical analysis of Van der Pauw's method for measuring resistivity', Journal of Physics: Conference Series **2321**, 012027 (2022).

[58]X. Zhang, 'Characteristics of Tungsten Silicide and its Application for Single X-ray Photon Detection', PhD thesis (University of Zurich, Zürich, 2018), pp. 31–34.

[59]R. Meservey and P. M. Tedrow, 'Measurements of the Kinetic Inductance of Superconducting Linear Structures', Journal of Applied Physics **40**, 2028–2034 (1969).

[60]A. J. Annunziata et al., 'Tunable superconducting nanoinductors', Nanotechnology **21**, 445202 (2010).

[61]H. Rotzinger et al., 'Aluminium-oxide wires for superconducting high kinetic inductance circuits', Superconductor Science and Technology **30**, 025002 (2017).

[62]L. Zhang et al., 'Hotspot relaxation time of NbN superconducting nanowire single-photon detectors on various substrates', Scientific Reports **8**, 1486 (2018).

[63]K. Taylor, 'Intermetallic rare-earth compounds', Advances in Physics **20**, 551–660 (1971).

[64]P. L. Rossiter, *The Electrical Resistivity of Metals and Alloys* (Cambridge University Press, Apr. 1987).

[65]K. K. Schulze, 'Preparation and Characterization of Ultra-High-Purity Niobium', JOM **33**, 33–41 (1981).

[66]W. Singer, A. Ermakov and X. Singer, *RRR-Measurement Techniques on High Purity Niobium*, tech. rep. (2010), pp. 1–3.

[67]Z. Charifoulline, 'Residual Resistivity Ratio (RRR) Measurements of LHC Superconducting NbTi Cable Strands', IEEE Transactions on Applied Superconductivity **16**, 1188–1191 (2006).

[68]D.-K. Liu et al., 'Nonlatching Superconducting Nanowire Single-Photon Detection with Quasi-Constant-Voltage Bias', Applied Physics Express **5**, 125202 (2012).

[69]Quantum Design, *PPMS Hardware Manual*, Feb. 2008.

[70]L. Aslamasov and A. Larkin, 'The influence of fluctuation pairing of electrons on the conductivity of normal metal', Physics Letters A **26**, 238–239 (1968).

[71]M. Sidorova et al., 'Electron energy relaxation in disordered superconducting NbN films', Physical Review B **102**, 054501 (2020).

[72]Yokogawa Test & Measurement Corporation, *Yokogawa Gs200 User Manual*.

[73]Institute of Electrical and Electronics Engineers, *IEEE Standard Digital Interface for Programmable Instrumentation* (1987).

[74]Dassault Systèmes SolidWorks Corporation, *SolidWorks Manual*, 2015.

[75]J. Allison et al., 'Geant4 developments and applications', IEEE Transactions on Nuclear Science **53**, 270–278 (2006).

[76]E. Aprile et al., 'Analysis of the XENON100 dark matter search data', Astroparticle Physics **54**, 11–24 (2014).

[77]J. Allison, 'The Geant4 Visualization System - A Multi-Driver Graphics System', International Journal of Modeling, Simulation, and Scientific Computing **4** (2013).

[78]J. Allison, 'The Geant4 Visualisation System', Computer Physics Communications **178**, 331–365 (2008).

[79]J. Apostolakis, 'Geometry and Physics of the Geant4 Toolkit for High and Medium Energy Applications', Radiation Physics and Chemistry **78**, 859–873 (2009).

[80]*ATLAS*, Vol. 30 (WORLD SCIENTIFIC, July 2019).

[81]K. Abe et al., 'Hyper-Kamiokande Design Report', (2018).

[82]C. M. Poole, I. Cornelius, J. V. Trapp and C. M. Langton, 'A CAD interface for GEANT4', Australasian Physical & Engineering Sciences in Medicine **35**, 329–334 (2012).

[83]T. KASE, K. KONASHI, H. TAKAHASHI and Y. HIRAO, 'Transmutation of Cesium-137 Using Proton Accelerator.', Journal of Nuclear Science and Technology **30**, 911–918 (1993).

[84]R. Bunting, 'Nuclear data sheets for A = 137', Nuclear Data Sheets **15**, 335–369 (1975).

[85]W. M. Haynes, *CRC Handbook of Chemistry and Physics*, 92nd ed. (CRC Press, 2011), p. 10.233.

[86]A. Stockhausen et al., 'Adjustment of self-heating in long superconducting thin film NbN microbridges', Superconductor Science and Technology **25**, 035012 (2012).

[87]K. Il'in, M. Siegel, A. Semenov, A. Engel and H.-W. Hübers, 'Critical current of Nb and NbN thin-film structures: The cross-section dependence', physica status solidi (c) **2**, 1680–1687 (2005).

[88]J. Markoff, *Artificial-Intelligence Research Center Is Founded by Silicon Valley Investors*, Dec. 2015.

[89]OpenAI, *GPT-4 Technical Report*, tech. rep. (OpenAI, 2023).

[90]G.-Z. Xu et al., 'Superconducting microstrip single-photon detector with system detection efficiency over 90% at 1550nm', Photonics Research **9**, 958 (2021).

[91]D. K. Liu et al., 'Electrical Characteristics of Superconducting Nanowire Single Photon Detector', IEEE Transactions on Applied Superconductivity **23**, 2200804–2200804 (2013).

[92]E. Toomey, Q.-Y. Zhao, A. N. McCaughan and K. K. Berggren, 'Frequency Pulling and Mixing of Relaxation Oscillations in Superconducting Nanowires', Physical Review Applied **9**, 064021 (2018).

[93]V. B. Verma et al., 'Single-photon detection in the mid-infrared up to 10 um wavelength using tungsten silicide superconducting nanowire detectors', APL Photonics **6**, 10.1063/5.0048049 (2021).

[94]M. Haffke et al., 'Background measurements in the Gran Sasso Underground Laboratory', Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **643**, 36–41 (2011).

[95]M. Laubenstein et al., 'Radioactivity measurements for the ERMES project at the STELLA facility', EPJ Web of Conferences **24**, 02002 (2012).

[96]XENON Collaboration et al., 'The XENON1T Dark Matter Experiment', 10.1140/epjc/s10052-017-5326-3 (2017).

[97]G. Alimonti et al., 'The Borexino detector at the Laboratori Nazionali del Gran Sasso', Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **600**, 568–593 (2009).

[98]N. Y. Agafonova et al., 'Characterization of the varying flux of atmospheric muons measured with the Large Volume Detector for 24 years', Physical Review D **100**, 062002 (2019).

[99]L. Baudis et al., 'Gator: a low-background counting facility at the Gran Sasso Underground Laboratory', 10.1088/1748-0221/6/08/P08010 (2011).

[100]D.-M. Mei and A. Hime, 'Muon-induced background study for underground laboratories', Physical Review D **73**, 053004 (2006).