Implementation of Decoy State QKD

Michael Auer

München
17. November 2020

Implementierung von Decoy State QKD

Michael Auer

Munich
November 17, 2020

Faculty of Physics
Ludwig-Maximilians-University Munich

Master Thesis

Implementation of Decoy State QKD

Michael Auer

November 17, 2020

Supervised by Prof. Dr. H. Weinfurter

Contents

1. Introduction

In 1999 a time capsule was sealed by the MIT Laboratory for Computer Science (LCS) in honor to its 35 year anniversary. It was meant to be shut for the next 35 years, unless a cryptography puzzle (LCS35 [1]) designed by Ron Rivest, one of the creator of the widely used RSA [2] cryptosystem, is solved earlier. The puzzle, finding $w = 2^{2^t} \pmod{n}$ given t and n, was designed such that the continuous calculation time would estimate to 35 years, whilst considering common chip speeds at that time and taking into account Moore's law. The only known way of finding w without knowledge about the factorization of n is by t successive squarings. In 2015, the self-taught Belgian programmer Bernard Fabrot dedicated a single CPU core of his home desktop computer to solve this intrinsically sequential problem. After three-and-a-half years, 15 years earlier than expected, he finally completed the approximately 80 trillion squaring operations and handed in the solution – "!!! Happy Birthday LCS !!!", just one month before another contestant completed his calculations.
The Cryptophage group approached the problem fundamentally different by using specialized hardware: an FPGA that is configured to run only one specific hardware-implemented algorithm, which was about ten times faster than a high-end consumer CPU, thus solving the puzzle in merely two months.
Rivest admitted that he has overestimated the difficulty of his puzzle, as predicting improvements in technology, let it be small performance gains or breakthroughs like FPGAs, is difficult on timescale this long.
In analogy the RSA cryptosystem, nowadays used by every single one of us in our daily lives, founds its security on a similar problem, where the decryption is only thought to be practically infeasible but not impossible. Rapid advances and breakthroughs in technology severely undermine this assumption as demonstrated by the two different solutions to the LCS35 puzzle. Keep in mind that some data needs to be kept secret for decades, e.g., banking, healthcare or DNA data and a retrospective security breach in ten or twenty years would disclose this confidential data.
The desire for a unconditionally secure cryptosystem leads us to the one-time pad (OTP) [3], where every bit of a message is XORed with at least one bit of a previously exchanged key. The OTP is proven to be information-theoretically secure, given that the communicating parties already share a fully random secret key. Such a key can be exchanged using quantum key dis-

tribution (QKD) [4–6], where in contrary to classical key distribution systems like RSA [2] or Diffie–Hellman [7], the security is not based on mathematical assumptions but only on quantum physical laws.
While the first QKD protocol, proposed by Charles Bennett and Gilles Brassard in 1984 (BB84) [8], was experimentally realized in 1992 over a 32.5 cm free-space link [9], QKD is nowadays demonstrated up to a fiber length of 421 km [10] and a satellite free-space link of 7600 km between China and Austria [11]. To achieve such distances, the *Micius* satellite acted as a trusted relay, i.e., two ground stations exchanged a secret key over approximately 500 km with the satellite, which then publicly announces the XOR of both keys in order to generate a mutual shared key. This 631 kg research satellite did cost around $100 million to build and launch. On the other hand projects using small (3L) and light (4kg) miniature satellites called *CubeSats* exist [12, 13], substantially reducing the mission cost. Commercially available products from ID Quantique [14] or Quantum CTek, able to exchange a secure key using a fiber link up to 75 km, weight more than 10 kg and come assembled in a 4U server rack (36 L). All these devices are not suitable for the usage with CubeSats or even handheld operation, possibly deployed in compact and mobile devices for cardless payment or authentication at ATMs.
Designing a polarization-encoded BB84 QKD sender for just these scenarios is the goal of a current project in our group [15, 16]. The CubeSats, as well as devices for handheld operation require for a light and small device featuring low power consumption and a high mechanical robustness.
This master's thesis is strongly focused on the design and testing of an electronics, extending the currently implemented protocol with decoy states. To be able to electronically switch between signal and decoy states a suitable radio-frequency switch and attenuation circuit has been chosen and incorporated. In addition to that the communication interface was upgraded to USB3.0 enabling real-time key exchange, while at the same time granting a versatile operation on 5V USB power. Finally, the sender electronics was programmed and characterized.

This thesis is organized as follows: Chapter 2 introduces the theoretical foundations quantum mechanics, classical cryptography and QKD. In chapter 3 the working principle of the electronics is explained and the most important design choices are elaborated. Chapter 4 presents the results of the performed characterization measurements followed by chapter 5, which provides a summary and discusses possible future improvements.
Additionally, Appendix A describes the used principles in PCB design and Appendix B gives an in-depth discussion of the used integrated circuits and the software developed for communicating with them.

2. Theoretical principles

In this chapter we take a small historical excourse to recall some principles of quantum mechanics (section 2.1) needed to model our experiment. Then we discuss the basic theory of classical cryptography (section 2.2). This will motivate the introduction of quantum key distribution (QKD) (section 2.3). In the course of this, we get to know some QKD implementations (section 2.3.2) and their vulnerabilities (section 2.3.3).

2.1. Quantum mechanics

Quantum mechanics[1] was developed to describe phenomena that could not be described via classical physics, e.g., the photoelectric effect, black body radiation or observations in the Stern-Gerlach (SG) experiment [18].
In the latter, silver atoms get heated and then shot through a spatially varying magnetic field, which is, without loss of generality, oriented in the z-direction. Due to the magnetic moment of the silver atoms, they get deflected in a corresponding direction. Silver features only one electron in the outmost shell whereas all contributions to the net angular momentum from the inner electrons cancel. This electron is in the s-orbital and hence does not carry any orbital angular momentum. Therefore, only the intrinsic spin of the outer electron contributes to the net angular momentum. From a classical point of view, this intrinsic spin can be seen as a "spinning ball" and as there is no preferred orientation of this angular momentum, one would expect a continuous spatial distribution of the silver atoms after the experiment. But when the experiment is performed, we find two distinct spots on the detector.
This behavior can only be explained by quantum mechanics, treating the silver atoms as neutral spin-$\frac{1}{2}$ particles. The state of the system can be described in Dirac notation by a ket-vector $|\Psi\rangle$ that lies in a complex Hilbert space $\mathcal{H}$ [19]. The dimensionality of this vector space depends on the type of the observable under consideration. In the SG case we consider only the spin described by the operator S_z, featuring two eigenstates $e_1 = |\uparrow\rangle$ and $e_2 = |\downarrow\rangle$ with eigenvalues $\lambda_{1,2} = \pm\frac{\hbar}{2}$. In the case of such a *two-level system* (TLS), the Hilbert space is two-dimensional. All other degrees of freedom, such as the position of the

[1] The section largely follows [17, Ch. 1].

particle, are neglected for the time being. For every ket $|\Psi\rangle$ there exists a corresponding bra $\langle\Psi|$, which lives in a Hilbert space dual to the ket space.
In quantum mechanics, measurable physical quantities such as spin are called observables and can be represented by operators A. In the case of a finite-dimensional Hilbert space, such an operator is represented by a Hermitian matrix. If a system is in an eigenstate $|a\rangle$ of the observable and we apply the operator, we get the same state back but scaled with the corresponding eigenvalue $A\,|a\rangle = a\,|a\rangle$. In the SG case this yields

$$S_z\,|\uparrow\rangle = +\frac{\hbar}{2}\,|\uparrow\rangle\,, \tag{2.1a}$$

$$S_z\,|\downarrow\rangle = -\frac{\hbar}{2}\,|\downarrow\rangle\,. \tag{2.1b}$$

The eigenvalues of a Hermitian operator A are real and the eigenstates of A to different eigenvalues are orthogonal [17, ch 1.3]. When working with normalized kets $\langle a|a\rangle = 1$, it is easy to see that an operator can be spectrally decomposed into its eigenstates and values as $A = \sum_a a|a\rangle\langle a|$. Again, if we look at the SG case

$$S_z = \sum_{i=1}^{2} \lambda_i\,|e_i\rangle\,\langle e_i| = \frac{\hbar}{2}\Big(|\uparrow\rangle\langle\uparrow| - |\downarrow\rangle\langle\downarrow|\Big) \tag{2.2}$$

$$\begin{aligned} S_z\,|\uparrow\rangle &= \frac{\hbar}{2}\Big(|\uparrow\rangle\langle\uparrow| - |\downarrow\rangle\langle\downarrow|\Big)\,|\uparrow\rangle \\ &= \frac{\hbar}{2}\Big(|\uparrow\rangle\langle\uparrow\,|\,\uparrow\rangle - |\downarrow\rangle\langle\downarrow\,|\,\uparrow\rangle\Big) \\ &= \frac{\hbar}{2}\Big(\,|\uparrow\rangle\cdot 1 - |\downarrow\rangle\cdot 0\Big) = +\frac{\hbar}{2}\,|\uparrow\rangle \end{aligned} \tag{2.3}$$

We can see that Equation 2.3 reproduces 2.1a and, analogously 2.1b. The multiplication of operators X and Y is in general non-commutative:

$$XY \neq YX \tag{2.4}$$

and any state $|\psi\rangle$ can always be regarded as a superposition of two or more states. If the states $|\phi_n\rangle$ form a basis of the considered Hilbert space, we can construct any desired state vector $|\psi\rangle$ in that space:

$$|\psi\rangle = \sum_n c_n\,|\phi_n\rangle\,. \tag{2.5}$$

If one considers (non-entangled) multipartite systems, one can express the corresponding state using the tensor product

$$|\psi\rangle = |\uparrow\rangle_1 \otimes |\downarrow\rangle_2 \tag{2.6}$$

where $|\uparrow\rangle_{1,2}$ is the state of system 1 or respectively 2. If a state requires the superposition of these states and cannot be decomposed into a single tensor product of the form Equation 2.6, the system is shown to be entangled.

2.1.1. Quantum measurements

According to P.A.M. Dirac [20, p. 36], a measurement may cause a jump in the state of a quantum system owing to the disturbance introduced into the system by the sole act of measuring an observable. Dirac also states that the possible outcomes of the measurement are limited to the eigenvalues of the measured operator. If we interpret a state $|\psi\rangle = \sum_a c_a |a\rangle$ as a superposition (see Equation 2.5) of all eigenstates of the operator A, the measurement will cause the system to jump to the corresponding eigenstate $|a\rangle$ of the measurement result a. The coefficient c_a of the superposition yields the probability for such a measurement outcome

$$p_a = |c_a|^2 = \left| \sum_{a'} c_{a'} \langle a'|a\rangle \right|^2 = \left| \langle \psi|a\rangle \right|^2. \tag{2.7}$$

We can now think of two consecutive SG devices both measuring S_z. Because the SG experiment only considers a two-dimensional Hilbert space, there are only two distinct results — our two spots — one with atoms jumped into the state $|\uparrow\rangle$ and the other one, where atoms collapsed into $|\downarrow\rangle$. If we then send only the $|\uparrow\rangle$ atoms through another SG with the same orientation, there will only be a single spot because the superposition we started from was destroyed by the first measurement.
If we now replace the second SG measuring S_z with one that measures S_x (or S_y), we get two spots again, in analogy with the previous result. Classically, one could be tempted to interpret it as having atoms with the properties $\left(|\uparrow\rangle_{S_z}, |\uparrow\rangle_{S_x}\right)$ and $\left(|\uparrow\rangle_{S_z}, |\downarrow\rangle_{S_x}\right)$, respectively. But if we now take atoms in one of those spots and send them through a third SG apparatus, again measuring S_z, we find two distinct spots with both $|\uparrow\rangle$ and $|\downarrow\rangle$ atoms. Given that we made sure only $|\uparrow\rangle$ atoms enter the second SG, this seems contradictory. This is a nice example of the *uncertainty principle* introduced by W. Heisenberg in 1927, which asserts a fundamental limit to the precision which certain pairs of observables can be co-measured. Because S_z does not commute with S_x, i.e. $[S_z, S_x] \neq 0$, and we determined the spin in x-direction after the second SG with very high probability, all information about the previous measurement of S_z is destroyed.

2.1.2. Quantum bit

The most fundamental component in digital communication is the *bit*, which can have either the value *0* or *1*. The quantum counterpart is called *qubit* (quantum bit) [21], a TLS which features two orthogonal states $|0\rangle$ and $|1\rangle$. To encode information onto the qubit, one can utilize any two orthogonal states of the system. However, in contrast to the classical case, we are also able to use a superposition $|Q\rangle = c_0 \cdot |0\rangle + c_1 \cdot |1\rangle$ of the two orthogonal states. If we choose the coefficients c_0 and c_1 such that they fulfill $|c_i|^2 = \frac{1}{2}$, we can construct two additional bases. Together they form three mutually conjugated bases resembling the eigenstates of S_x, S_y and S_z, respectively.
A key aspect of qubits comes in handy if we want to communicate securely. Due to the quantum nature of qubits, in general they cannot be duplicated [21]. The simplest approach of measuring and preparing the measured state multiple times will not work due to the effects introduced in section 2.1.1. More sophisticated methods are forbidden by the *no cloning theorem* [19], which states that there is no unitary operation U that clones the arbitrary state $|\psi\rangle$ of a first system onto a second system, i.e., that fulfills

$$U\Big(\,|\psi\rangle \otimes |k\rangle\,\Big) \stackrel{!}{=} |\psi\rangle \otimes |\psi\rangle \tag{2.8}$$

where $|k\rangle$ is the initial state of the second system. As stated above, any TLS system can work as a qubit, so one can use different observables of underlying physical systems. The most popular ones used in quantum computing or quantum cryptography are, following [22],

- the current in a Josephson junction,
- the spin of a quantum dot,
- the spin of an atom,
- the polarization of photons.

The choice of implementation depends on what one wants to achieve. For example, for storing quantum information one needs very long coherence times, for a quantum processor one needs, besides other requirements, tunable coupling between different qubits [22] and for communication one must be able to physically transmit the qubit without high loss or decoherence.

2.1.3. Polarized light

Electromagnetic waves constitute the ideal carrier for quantum information and are hence ideal for quantum communication tasks. In our case, we use the polarization of light. Therefore, we will now introduce the basic properties of light.

Polarization of classical waves With the help of Maxwell's equations it can be shown that an electromagnetic wave fulfills the following relations in optically isotopic media [23].

$$\vec{k} \perp \vec{E}, \tag{2.9a}$$

$$\vec{k} \perp \vec{B}, \tag{2.9b}$$

$$\vec{B} \perp \vec{E}. \tag{2.9c}$$

Using this and assuming, without loss of generality, that $\vec{k}$ is oriented in the z-direction, we can represent every plane wave via

$$\vec{E}(z,t) = \begin{pmatrix} E_{x_0}\cos(kz - wt) \\ E_{y_0}\cos(kz - wt + \varphi) \\ 0 \end{pmatrix}, \tag{2.10}$$

where E_{x_0} and E_{y_0} are the amplitudes of the electric field in x- and y-direction, respectively. If we choose $\varphi = n \cdot \pi$ we get the electric field of linearly polarized light

$$\vec{E}(z,t) = \begin{pmatrix} E_{x_0} \\ \pm E_{y_0} \\ 0 \end{pmatrix} \cos(kz - wt) = \vec{E_0}\cos(kz - wt), \tag{2.11}$$

which is oriented in the $\vec{E_0}$ direction. If we choose $\varphi = \frac{\pi}{2} + n\pi$ and $E_{x_0} = E_{y_0} = E_0$, the resulting electric field is

$$\vec{E}(z,t) = E_0 \begin{pmatrix} \cos(kz - wt) \\ \pm\sin(kz - wt) \\ 0 \end{pmatrix}, \tag{2.12}$$

where the x- and y-coordinates describe an orbit around the z-axis. This special case is called circular polarized light. If we choose $E_{x_0} \neq E_{y_0}$, we get elliptical polarization, which is the most general case but can be decomposed into linear and circular parts.

Polarization of photons A photon is polarized in analogy to the classical electromagnetic wave. In general photons are spin-1 particles and one would expect three different eigenvalues, however as they are massless and moving with the speed of light, the eigenvalue zero is not present and only two eigenvalues are observed, which directly correspond to the polarization eigenvalues. Hence, it is convenient to describe the particle with a spin-$\frac{1}{2}$ formalism.
The basis of the corresponding two-dimensional Hilbert space can be used arbitrarily. Typical choices are $|H\rangle, |V\rangle$ (horizontal, vertical); $|L\rangle, |R\rangle$ (left, right circular) and $|P\rangle, |M\rangle$ (plus, minus 45°). One often uses the Stokes vector to describe the polarization state,

$$\vec{S} = \begin{pmatrix} I \\ Q \\ U \\ V \end{pmatrix} = \begin{pmatrix} I_H + I_V \\ I_H - I_V \\ I_P - I_M \\ I_R - I_L \end{pmatrix}, \tag{2.13}$$

where $I = I_H + I_V = I_P + I_M = I_R + I_L$ and, say, I_H is the measured intensity of H polarization. It is convenient to normalize the vector such that $I = 1$ by dividing every entry by I.
To visualize the states we make use of the *Poincaré sphere* shown in Figure 2.1.

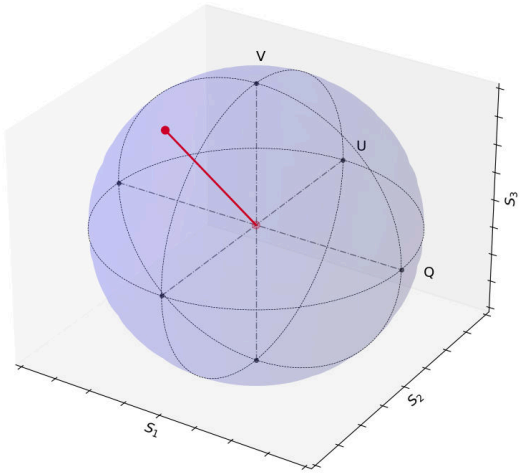

Figure 2.1.: An arbitrary state in the Poincaré sphere with $\vec{S} = (1, -0.3, -0.5, 0.81)$. Q=±1 translates to H/V polarized, U=±1 corresponds to P/M polarized and V=±1 to R/L polarized light.

Since the introduced three bases span the same Hilbert space, one can express one in terms of the others,

$$|P\rangle = \frac{1}{\sqrt{2}}\Big(|H\rangle + |V\rangle\Big) \qquad |M\rangle = \frac{1}{\sqrt{2}}\Big(|H\rangle - |V\rangle\Big) \tag{2.14a}$$

$$|R\rangle = \frac{1}{\sqrt{2}}\Big(|H\rangle + i\,|V\rangle\Big) \qquad |L\rangle = \frac{1}{\sqrt{2}}\Big(|H\rangle - i\,|V\rangle\Big) \tag{2.14b}$$

Degree of polarization If we want to express to what extent an electromagnetic wave is polarized, we utilize

$$p = \frac{\sqrt{Q^2 + U^2 + V^2}}{I} \tag{2.15}$$

where p is the *degree of polarization* (DOP) ranging from zero to one. In the Poincaré picture, the DOP is a measure for the distance from the pictured state to the center of the sphere. The state shown in Figure 2.1 has a DOP of one.

Quantum state tomography To determine the polarization state and its DOP we use *quantum state tomography* (QST). A possible experimental setup for measuring those quantities is illustrated in Figure 2.2.

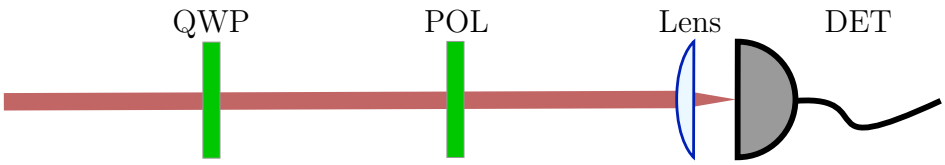

Figure 2.2.: Schematic of an experimental QST setup. A beam of unknown polarization states is sent through a quarter wave plate (QWP) followed by a polarizer (POL) and focused by a lens into any intensity measurement device (DET).

In general, the required wave plate is made of a birefringent material, i.e., a material for which the refractive index differs for polarizations aligned along the different axis of the crystal [23]. A phase shift dependent on the thickness of the material is introduced between those certain perpendicular polarizations. A quarter wave plate introduces a phase shift of $\varphi = \frac{\pi}{2}$ and according to Equation 2.10 this results in elliptical polarization. If the axis of polarization of the incident light is chosen as $\Theta = 45°$ to the optical axis of the material, we get circularly polarized light (see 2.12), for linearly polarized incoming light. A half wave plate, on the other hand, adds a $\varphi = \pi$ phase shift and therefore

rotates linear polarization by 2Θ depending on the angle between the optical axis and the polarization Θ. Elliptically polarized light is inverted in terms of the light's handedness. A polarizer (POL) filters light of a specific, typically linear polarization by only transmitting this polarization and reflecting or absorbing every other.

Projection	H	V	R	L	P	M
QWP	0°	0°	0°	0°	+45°	+45°
Polarizer	0°	90°	+45°	−45°	+45°	−45°

Table 2.1.: Settings of QWP and polarizer angles resulting in a projection to the corresponding polarization. Table taken from [24].

If we set the angles according to Table 2.1 we can project an unknown polarization state onto any of the six basis vectors. From the measured intensities $I_{\mathrm{H}}, I_{\mathrm{V}}, ..., I_{\mathrm{L}}$ we can therefore calculate Q, U, V and p.

Measuring polarization states If we now take a look at the probability of detecting $|H\rangle$ polarized photons, we project this incoming state onto all six different basis states. Using Equation 2.7, 2.14a and 2.14b we get

$$P_H = \left| \langle H|H\rangle \right|^2 = 1 \qquad P_V = \left| \langle V|H\rangle \right|^2 = 0 \tag{2.16a}$$

$$\begin{aligned} P_P &= \left| \langle P|H\rangle \right|^2 \\ &= \left| \frac{1}{\sqrt{2}}(\langle H|H\rangle + \langle V|H\rangle) \right|^2 \\ &= 0.5 \end{aligned} \qquad \begin{aligned} P_M &= \left| \langle M|H\rangle \right|^2 \\ &= \left| \frac{1}{\sqrt{2}}(\langle H|H\rangle - \langle V|H\rangle) \right|^2 \\ &= 0.5 \end{aligned} \tag{2.16b}$$

$$\begin{aligned} P_R &= \left| \langle R|H\rangle \right|^2 \\ &= \left| \frac{1}{\sqrt{2}}(\langle H|H\rangle + i\langle V|H\rangle) \right|^2 \\ &= 0.5 \end{aligned} \qquad \begin{aligned} P_L &= \left| \langle L|H\rangle \right|^2 \\ &= \left| \frac{1}{\sqrt{2}}(\langle H|H\rangle - i\langle V|H\rangle) \right|^2 \\ &= 0.5 \end{aligned} \tag{2.16c}$$

From this we can see that if and only if we measure in the right basis, we can be certain about the outcome of the measurement. This is analogous to the case discussed in section 2.1.1, where we have two successive S_z SG devices. The first one projects an unknown state onto the S_z basis after which the second apparatus is able to measure in the right basis, yielding only the exact result every time.

Photon numbers In later sections we need to be able to estimate the number of photons in a light pulse. A coherent state $|\alpha\rangle$ resembles a state of a quantum harmonic oscillator. The dynamics of such states behave similarly to the classical harmonic oscillator. Such states with $\alpha = |\alpha| \, \mathrm{e}^{i\varphi}$ can be expressed, using second quantization, in the basis of photon number states, where $|n\rangle$ is the state containing n photons,

$$|\alpha\rangle = e^{-\frac{|\alpha|^2}{2}} \sum_{n=0}^{\infty} \frac{\alpha^n}{\sqrt{n!}} |n\rangle \, . \tag{2.17}$$

The probability to detect n photons is given by

$$\begin{aligned} P(n) = |\langle n|\alpha\rangle|^2 &= \left| e^{-\frac{|\alpha|^2}{2}} \sum_{n'=0}^{\infty} \frac{\alpha^{n'}}{\sqrt{n'!}} \langle n|n'\rangle \right|^2 \\ &= \left| e^{-\frac{|\alpha|^2}{2}} \sum_{n'=0}^{\infty} \frac{\alpha^{n'}}{\sqrt{n'!}} \delta_{n,n'} \right|^2 = e^{-|\alpha|^2} \frac{|\alpha|^{2n}}{n!}, \end{aligned} \tag{2.18}$$

which is a Poissonian probability distribution with mean photon number $\mu = |\alpha|^2$. Therefore, we can express the state in terms of the mean photon number by

$$|\alpha\rangle = |\sqrt{\mu} \, \mathrm{e}^{i\varphi}\rangle \, . \tag{2.19}$$

2.2. Classical cryptography

Cryptography (from Ancient Greek: *kryptós* "hidden, secret"; *graphein* "to write") is nowadays used in several different contexts like data integrity, authentication and even anonymous currency. Nevertheless, the initial purpose "to write hidden" from some third party is still its main goal. Two parties, conventionally called Alice and Bob, want to communicate a message m without revealing any information to an eavesdropper Eve.
To achieve this, we either have to prevent Eve from accessing the communication channel, or encrypt our message such that Eve cannot receive any usable information despite listening to the channel. As one can imagine, it is not the best practice to rely on an inherently secure channel, as the courier of the message can always be intercepted by Eve. Therefore one often focuses on the latter approach, where we have encryption $E(m, K_e) = c$ and decryption $D(c, K_d) = m$ algorithms illustrated in Figure 2.3.

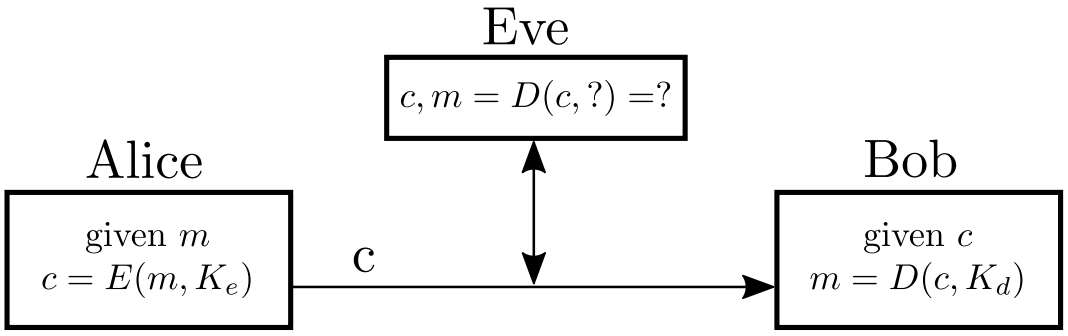

Figure 2.3.: Alice wants to send a message m to Bob. She encrypts m with $E(m, K_e)$ using the encryption key K_e to get the cipher c. She then sends c via a potentially insecure channel to Bob. Bob is able to decrypt the cipher with $D(c, K_d)$ using his decryption key K_d. Eve can intercept the communication, but has no means of decrypting the cipher without knowledge of K_d.

For our current discussion, it is not important which specific algorithms are used, as long as they fulfill

$$m = D\Big(E(m, K_e), K_d\Big). \tag{2.20}$$

Such algorithms utilize a (secret) key K, where one differentiates between symmetric $K_e = K_d$ and asymmetric $K_e \neq K_d$ cryptosystems. Generally, one can argue that symmetrical systems such as the *one-time pad* [3] (OTP), DES [26], AES [27], TwoFish [28] are simpler to implement and faster than asymmetrical ones [29].
The OTP gives an example of providing information-theoretical security: the cipher cannot be broken even if Eve has unlimited computing power. This is achieved by using a one-time key which is at least as long as the sent message. If the key is truly random and kept secret, there is no way an adversary can retrieve the full message, as every message bit is encrypted with at least one uniquely used key bit, therefore exhibiting no correlations to any other key bit. For every possible message (sequence of plain text bits) of the given length of the cipher there exists a key encrypting the message to the given cipher. Therefore, without knowledge about the key, all plain text messages of given length are equally probable. This comes at the cost of a long key length, which other algorithms try to minimize while remaining as safe as possible.
In summary, symmetrical systems have the significant disadvantage of relying on a (potentially long) shared secret key. This key cannot be transmitted to Bob via the insecure channel because, without encryption, Eve has full knowledge about the key and is therefore able to decrypt any sent ciphertext.

Even worse, if Alice wants to communicate with multiple different parties, she needs at least one key per party.

> *"The problem of distributing and managing keys is one of the really difficult parts of cryptography, for which we have only partial solutions."* [30, p. 27].

One of these partial solutions is the widely used RSA [2] algorithm, which is one of the first public key cryptosystems. It utilizes a key pair where $K_{\text{public}} = K_e \neq K_d = K_{\text{private}}$ and is therefore considered as an asymmetrical algorithm. Such cryptosystems are mostly used for key distribution. Alice first transmits a key $K_{\text{sym}} \neq K_{\text{pub/priv}}$ via RSA securely to Bob. With this key K_{sym}, a symmetrically encrypted conversation can be achieved (see TLS-protocol [31]).
If Alice wants to communicate with Bob via RSA, she takes his publicly available key $K^{\text{Bob}}_{\text{public}}$, encrypts her message $m = K_{\text{sym}}$ and sends the ciphertext $c = E(m, K^{\text{Bob}}_{\text{public}})$ to Bob. If the private key $K^{\text{Bob}}_{\text{private}}$ is kept secret, only Bob can decrypt the original message. This statement holds only if we trust the assumptions made in the security proof of RSA. The security of RSA is only based on the infeasibility of solving the RSA problem, which in turn can be reduced to an integer factorization problem [32]. It just takes a long time to break the cipher instead of having information theoretical security.
At the time RSA was invented the authors recommended a key length of $n = 664$bit but considered 265 bits as "moderately secure" [2]. In 1999 a 512 bit number was factorized [33]. A 768 bit number was factorized ten years later [34], which is several thousand times harder than factoring the 512 bit number.
They all reduced the RSA problem to a factorization problem but there is no proof that this is the most efficient way. If some other algorithm is found or if the factorization problem is solved faster, the security of RSA would be compromised. Until now no classical algorithm is known to solve the problem in $\mathcal{O}(n^k)$, i.e., in polynomial time. However, in 1994 Peter Shor discovered a quantum algorithm which solves the problem in $\mathcal{O}(\log_2(n)^3)$ [35].
According to Häner et al. [36], one needs $N_{\text{qubits}} = 2n + 2$ fault-tolerant qubits to implement Shor's algorithm. With the recommended bitlength of $n = 3072$ [37], we would need 6146 fault tolerant qubits available. But today, even one of the most advanced Quantum Computers (e.g. *Sycamore*) by Google features only 54 erroneous qubits [38]. Furthermore to provide fault tolerance, error correction protocols have to be applied with an estimated overhead of at least a factor of 20. Nevertheless this could lead to a retroactive security breach if an attacker is able to store the messages for a long time and simply delay decrypting it until a better algorithm or hardware is invented.

2.3. Quantum key distribution

Quantum key distribution is a possible solution for the issue mentioned in the previous section, i.e., sharing a secure key between sender and receiver. The big advantage of QKD is that an attack from a third party can be detected due to the principles discussed in section 2.1.1. If Eve tampers in any way with the exchanged key, she will introduce errors resulting in a higher *quantum bit error ratio* (QBER). From this quantity we can estimate the information possibly leaked to an attacker.
Another important advantage is that Eve is forced to break the QKD system in real time because after the key is exchanged there exists no record of what was transmitted. The security of QKD does therefore not depend on any mathematical assumptions or infeasible operations, but physical processes.

2.3.1. Basic concepts

For QKD, Alice and Bob have to extend their existing communication link by a quantum channel (Figure 2.4), which can be realized by an optical fiber or using a free-space link. While fiber-based systems have the advantage that the rate of background detections is smaller, the signal is exponentially damped with the distance of transmission. Whereas free-space transmissions undergo a quadratic reduction due to the divergence of the optical beam. However, the latter requires a line of sight, while fibers can still be utilized if the sight is obstructed. Note that a authenticated classical channel is still needed for post processing.

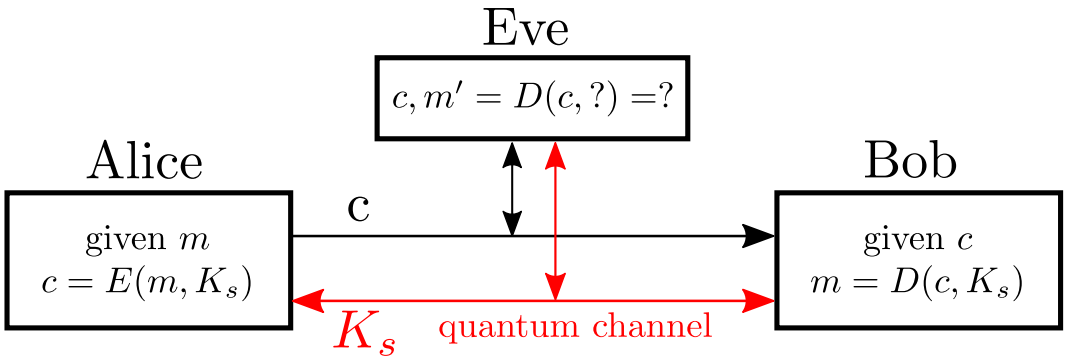

Figure 2.4.: The classical symmetrical cryptographic scheme in Figure 2.3 consisting of a classical channel (black arrow) and an encryption algorithm (E, D) is extended by a quantum channel (red arrow) for the initial distribution of the key (K_s).

The quantum channel is used to establish the raw key, where the received signals can contain some errors, which are introduced by an attacker, by dark counts, by background light or due to other experimental imperfections. These errors can be corrected via some *error correction* protocol (e.g., Cascade [39] or LDPC [40]). After all errors have been detected and corrected, one executes a *privacy amplification* protocol (see, e.g., [41],[42]), which effectively nullifies any information Eve may have gotten into her hands during the quantum signal exchange or the classical communication. After a successful key exchange, one continues with the purely classical steps described in section 2.2.
The biggest challenges of realizing QKD are noise and channel losses, leading to practical QKD setups that achieve distances of approximately 100 km with reasonable key rates through optical fibers. There are some demonstrations of QKD links for distances over 400 km [10], but for a global secure communication one would need several thousand kilometers of range. Classical links have similar problems with losses, though one can solve this problem with repeater nodes, which refresh or amplify the signal. This method, however, cannot be used for quantum communication due to the no cloning theorem.
Since the first QKD protocol was published in 1984 [8], various QKD protocols emerged, each with different strengths and weaknesses. The next section will give a brief overview of the categories those protocols can be assigned to and discuss the protocol implemented in this work in depth.

2.3.2. QKD protocols

As stated in section 2.1.2, the physical implementation of a qubit varies depending on the use case. All QKD protocols use electromagnetic waves as their carrier because they are easy to transport from Alice to Bob and the manipulation of their states is technologically feasible. There exist two main categories for the technical implementation of QKD.

Discrete variable QKD
In DV-QKD, the information is encoded into a discrete variable such as a two-level system. Within this class of protocols, different distinctions can be made. One can differentiate between prepare-and-measure schemes, in which the sender actively prepares the quantum state of the information carrier and sends it to Bob, and entanglement-based schemes, in which Alice and Bob share an entangled state, produced by either one of them or even a third party. If Alice and Bob perform suitable measurements in corresponding bases, they can create a random key. Examples for prepare-and-measure schemes for discrete variable QKD are BB84 [8] and SARG04 [43]. The E91 protocol developed by A. Ekert [44] is an example of an entanglement-based protocol.

Another distinction of DV-QKD protocols can be made from the dimensionality of the system used for encoding. Whereas protocols such as BB84, SARG04 and E91 are typically based on qubits, i.e., two-level systems, one can also resort to higher dimensional systems and encode the information using, e.g., the orbital angular momentum of light. Although the protocol might not depend on the actual choice of used two-level system (polarization of light, phase, etc.), some choices allow for further modifications of the scheme. For example, using an encoding based on the phase of light permits to use a differential encoding in subsequent pulses.

Continuous variable QKD
In CV-QKD, a continuous-variable, i.e., infinite dimensional system is used for information encoding. For example, the quadratures of light can be used here. Similar to DV-QKD protocols, also prepare-and-measure and entanglement-based CV-QKD protocols exist. Possible distinctions for CV-QKD are between coherent [45] and squeezed light [46], between a Gaussian modulation [45] and a discrete modulation [47], between homodyne [45] and heterodyne detection [46] or between direct [48] and reverse [49] reconciliation.

The security proofs of DV-QKD protocols are well understood, but as they are often based on single photons one has to reliably create and measure those quanta of light, which is quite hard to accomplish technically. CV-QKD protocols are able to utilize standard optical communication components used by the industry. However their security proof is more challenging. All those different protocol types are summarized and compared thoroughly in [6].

2.3.2.1. BB84

In this section we will discuss the BB84 [8] protocol in depth because it is used in our experiment. It is the oldest QKD protocol and was proposed by Bennett and Brassard in 1984. In general one could use every DOF of a photon, but similar to the creators we chose the polarization for information encoding. The protocol works as followed

1. For each signal Alice prepares a photon in a random basis and bit selection, illustrated in Figure 2.5. She could use any two of the three bases, here we choose $B_X = \{|H\rangle, |V\rangle\}$ and $B_Y = \{|P\rangle, |M\rangle\}$ where bit value 0 corresponds to $|H\rangle$ and $|P\rangle$ and 1 to $|V\rangle$ and $|M\rangle$, respectively.

2. For each signal, Bob performs a random basis selection and measures in the corresponding basis B_X or B_Y. The probability P_S to measure any state $|S\rangle$ given Alice sent, e.g., $|H\rangle$ is calculated using Equation 2.16a. If

the photon was measured in the same basis as it was sent, the probability to detect the actual polarization is $P_H = 1$. But if the photon is measured in the other basis, one can get both bit values with equal probability.

3. Bob reports whenever a signal (photon) was detected. Using this information, every bit where Bob did not detect anything is discarded by Alice.

4. Alice and Bob publicly announce their choice of basis through an authenticated classical channel and discard the polarization data that have been en-/decoded in the wrong bases. This step is called *basis reconciliation* and the remaining data is referred to the *sifted key*.

5. Both parties now choose a random sample of the sifted key and compare them through the classical channel to compute the QBER $= N_{\text{false}}/N_{\text{total}}$.

6. A high QBER can indicate the presence of an attacker. Thus, if the QBER is above a certain threshold, the key gets discarded and the protocol begins from step 1. Otherwise they proceed with classical post processing.

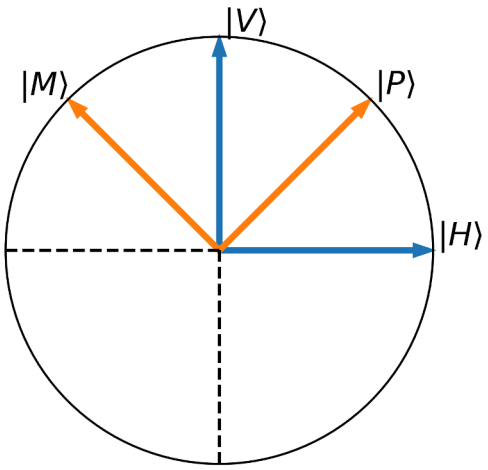

Figure 2.5.: The four different states used in the polarization encoded BB84 protocol. The blue arrows belong to the B_X basis, the red ones respectively to the B_Y basis. As we can see both bases span the same vector space.

To give an example of the promised security of the protocol one can think of the most basic attack: Eve can measure the incoming qubit from Alice and send another qubit prepared in the measured state to Bob. Due to Equation 2.16a, this attack will inevitably introduce additional errors, which will be detected by Alice and Bob in the process of basis reconciliation. Assuming random basis choices of Alice, Bob and Eve, in $P_e = 50\%$ of cases Eve will pick the wrong measurement basis. Because she prepares the qubit in the same basis as she measured in, Bob will detect the right bit value in $P_d = 50\%$ of those cases. The bits on Alice's and Bob's site will therefore differ with a $P_e \cdot P_d = 25\%$ probability. This increased QBER will reveal the presence of an attacker.
The unconditional security, even against more sophisticated attacks, was shown by Shor and Preskill [50] where they take a detour via entanglement purification, as do many QKD security proofs. Renner et al. [51] proves the security of many QKD protocols based on information theoretic results. In those proofs Eve is capable of doing everything physically possible without being limited to current technology. Furthermore all errors in transmission and detection are attributed to Eve. But they assume either a perfect detector or source of the implementation. In real world applications this is impossible to achieve.
Moreover, these proofs are only valid if a single qubit is used to transmit the information. Creating single photons is a technical challenge, as single-photon sources often require cooling to temperatures below 10 Kelvin by a cryostat and feature low repetition rates.
This is the reason why most DV-QKD experiments utilize *weak coherent pulses* (WCP) with mean photon number $\mu < 1$, which is achieved by simple attenuation of a laser. However, a small mean photon number does not guarantee no multi-photon pulses. Even at $\mu = 0.5$ there is a 10% chance of emitting multiple photons (see Figure 2.6), due to the fact that the laser light will follow Poissonian statistics, as discussed in section 2.1.3.

Such pulses would nullify discussed security proofs. Eve could intercept the communication and use the so-called *Photon Number Splitting* (PNS) attack. As the name suggests, Eve takes a multi-photon pulse and splits the photons. One photon is sent to Alice, the other, containing the same information, is stored by Eve. Pulses with only one photon can get blocked to increase Eve's information. After step 4 in the BB84 protocol, where the basis choices are exchanged, Eve can measure the stored photons in the correct basis and obtain full information about the key.
GLLP [5] (**G**ottesman, **L**o, **L**ütkenhaus and **P**reskill) use the same approach as Shor and Preskill [50] but consider the presence of multi-photon pulses. They present two rate formulas, one for a theoretical maximum secure key rate, i.e., fraction of sifted key bits that can be extracted as secure key bits, using single

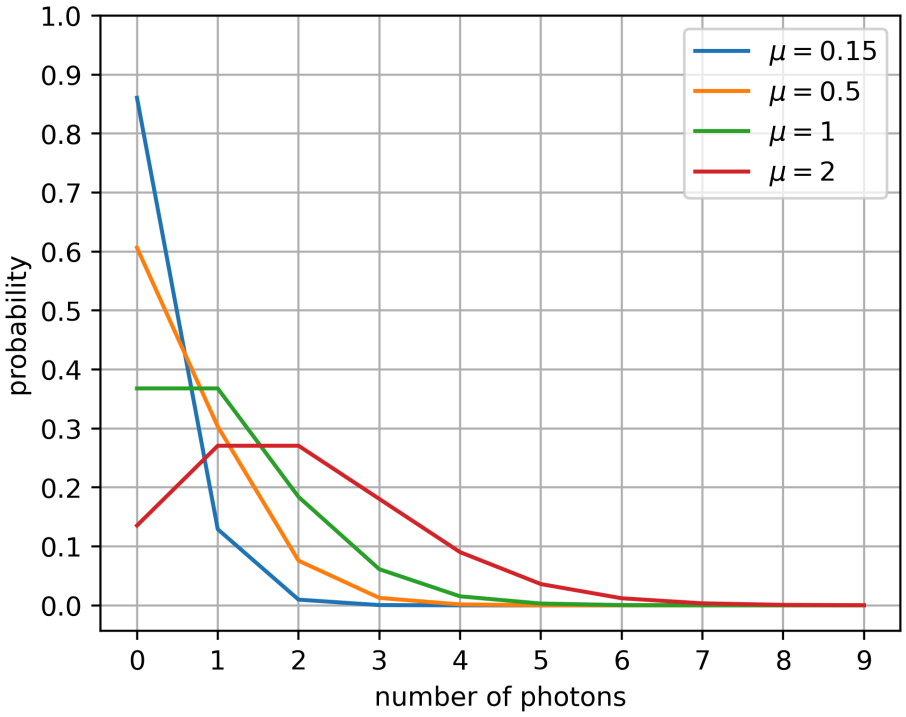

Figure 2.6.: Poissonian distribution plotted for four different mean photon numbers $\mu = 0.1, 0.4, 1, 2$. For $\mu < 1$ most ($\approx 60\,\%$) of the send states would be empty.

photons R_{single} and one for the implementation of WCP R_{GLLP}

$$R_{\mathrm{single}} = R_{\mathrm{sifted}} \cdot \max\Big(1 - 2H_2(E), 0\Big) \tag{2.21a}$$

$$R_{\mathrm{GLLP}} = R_{\mathrm{sifted}} \cdot \max\Big((1-\Delta) - f_{EC}(E)H_2(E) - (1-\Delta)H_2\Big(\frac{E}{1-\Delta}\Big), 0\Big) \tag{2.21b}$$

where $R_{\mathrm{sifted}} = f_{\mathrm{rep}} q \eta$ is the sifted key rate achieved after basis reconciliation, E is the QBER, f_{EC} denotes the efficiency of error correction (typically in the range of 1.22) and Δ is the number of tagged bits (fraction of multi-photon pulses emitted per detected pulse) with overall detection probability η and the factor q accounting for the sift efficiency ($q = 0.5$ for symmetrical basis BB84),

$$\Delta = \frac{P_\mu(n > 1)}{\eta P_\mu(n > 0)}. \tag{2.22}$$

Using Equation 2.21a, we can extract a maximum allowed QBER for secure key exchange $E_{\max} = \mathrm{QBER}_{\max} \approx 11\,\%$. For comparing the performance of the BB84 protocol implemented with single photons to the realization with WCP evaluated with the GLLP method, we plot both Equation 2.21a and 2.21b over the loss $1 - \eta$ in Figure 2.7. There it becomes obvious that the usage of the GLLP evaluation shows a drastically reduced secret key rate, as the mean photon numbers have to be adjusted proportional to the overall transmittance η, in order to guarantee a secure key.

2.3.2.2. Decoy states

GLLP showed how to transmit a secure key despite working with multi-photon pulses. But, as shown in the previous section, it is not very efficient. Based on the idea of Hwang [52], Lo et al. [53] proposed and proved the security of the *decoy state method*. This approach allows unconditional security for most of the experiments, using almost the same hardware.
The basic idea behind this method is that one does not only send pulses with mean photon number μ, but Alice chooses randomly an intensity $\nu < \mu$, with all other characteristics identical. If the intensities of a signal pulse (μ) and of a decoy pulse (ν) chosen such that $\mu, \nu < 1$ typically $\mu = 0.5$ and $\nu = 0.15$, Eve is not able to distinguish signal from decoy pulses and has to treat every pulse equally.
If she executes a PNS attack, removing one photon from every multi-photon pulse, the two different photon statistics of signal and decoy pulses get influenced unequally as illustrated in Figure C.1 and Figure C.2. The two states would undergo a different channel loss, therefore the PNS attack is revealed to Alice and Bob and they can abort the key exchange.
In real-world QKD it is hard to measure those statistics, moreover aborting the whole protocol after an attacker is detected is not efficient. That is why we want to get an estimate of the information available to Eve. With this upper bound we know to what extend we have to perform privacy amplification, to guarantee a secure key.
The original paper of Hwang proposes multiple decoy states with $\nu_i, i \in \mathbb{N}$, but Lo et al. [53] suggested the use of only a few decoy states. In particular, they mention three states: the vacuum $\nu_0 = 0$, a weak decoy state $\nu_1 \ll 1$ and the signal state with $\mu \approx 1$. As shown by Ma et al. [4], this decoy state protocol allows to upper bound the information leaked to Eve. They present a rate formula

$$\begin{aligned} R_{\text{decoy}} &= R_{\text{sifted}} \cdot \Big(- Q_\mu f_{EC}(E_\mu) H_2(E_\mu) + Q_1[1 - H_2(e_1)] \Big) \\ &\geq R_{\text{sifted}} \cdot \Big(- Q_\mu f_{EC}(E_\mu) H_2(E_\mu) + Q_1^{L,\nu,0}[1 - H_2(e_1^{U,\nu,0})] \Big), \end{aligned} \tag{2.23}$$

where $Q_i = Y_i \cdot P_i$ is the gain of an i-photon state, consisting of the probability to send such a state P_i times the yield Y_i, which denotes the probability to detect an i-photon state at Bob's side. The overall gain of a pulse sent with mean photon number μ is

$$Q_\mu = \sum_i Q_i = N_{\text{det}}/N_{\text{sent}} \cdot P_\mu, \tag{2.24}$$

where P_μ denotes the probability to send a state with brightness μ.
Note that Q_μ is quite easy to calculate, but measuring Q_i or the i-photon error rate e_i in a realistic experiment is hard to achieve with current technology, as it would require to resolve the actual photon number in a single pulse. This is why Ma et al. bounded those single photon quantities in Equation 2.23 using

$$Y_1 \geq Y_1^{L,\nu,0} = \frac{\mu}{\mu\nu - \nu^2}\left(Q_\nu e^\nu - Q_\mu e^\mu \frac{\nu^2}{\mu^2} - \frac{\mu^2 - \nu^2}{\mu^2} Y_0\right), \tag{2.25a}$$

$$Q_1 \geq Q_1^{L,\nu,0} = \mu e^\mu \cdot Y_1^{L,\nu,0}, \tag{2.25b}$$

$$e_1 \leq e_1^{U,\nu,0} = \frac{E_\nu Q_\nu e^\nu - e_0 Y_0}{Y_1^{L_1,\upsilon,0}\nu}. \tag{2.25c}$$

It should be stressed that every quantity used in these estimations is experimentally accessible. μ, ν are characterized by Alice before starting the protocol, $E_{\mu,\nu}, Q_{\mu,\nu}$ can be calculated in the basis reconciliation step and Y_0, where only the dark count and background events give a contribution, is available by sending and measuring the vacuum state.

A comparison of the decoy protocol with the previously discussed methods is shown in Figure 2.7. For high losses all graphs exhibits a kink, which can be explained by an increasing QBER, due to the lower signal-to-noise ratio. As the channel losses increase the signal strength is reduced, whereas contributions of background radiation or dark counts stay unchanged.
The data was simulated assuming background noise $f_{\text{background}} = 250\,\text{s}^{-1}$, a pulse repetition rate of $f_{\text{rep}} = 100\,\text{MHz}$, a detector error of $e_{\text{det}} = 2\,\%$, and a fixed error correction efficiency of $f_{EC}(E) = 1.22$. The values for dark counts $f_{\text{dark}} = 50\,\text{s}^{-1}$, detector efficiency $\eta_{\text{det}} = 0.9$ and dead time $t_{\text{dead}} = 10\,\text{ns}$ are modeled after a typical *superconducting nanowire single-photon detector* (SNSPD).
The single photon implementation shows positive key rates up to losses of above $-50\,\text{dB}$ and the unoptimized (in respect to μ and ν) decoy state protocol performs similarly well up to $-45\,\text{dB}$ whereas the rates of the GLLP method exhibit a much steeper decline and drop at approximately $-25\,\text{dB}$.
For example, satellite-based QKD has to expect losses of above $-40\,\text{dB}$ [12], for which the GLLP method does not allow for any secret key at all. On the

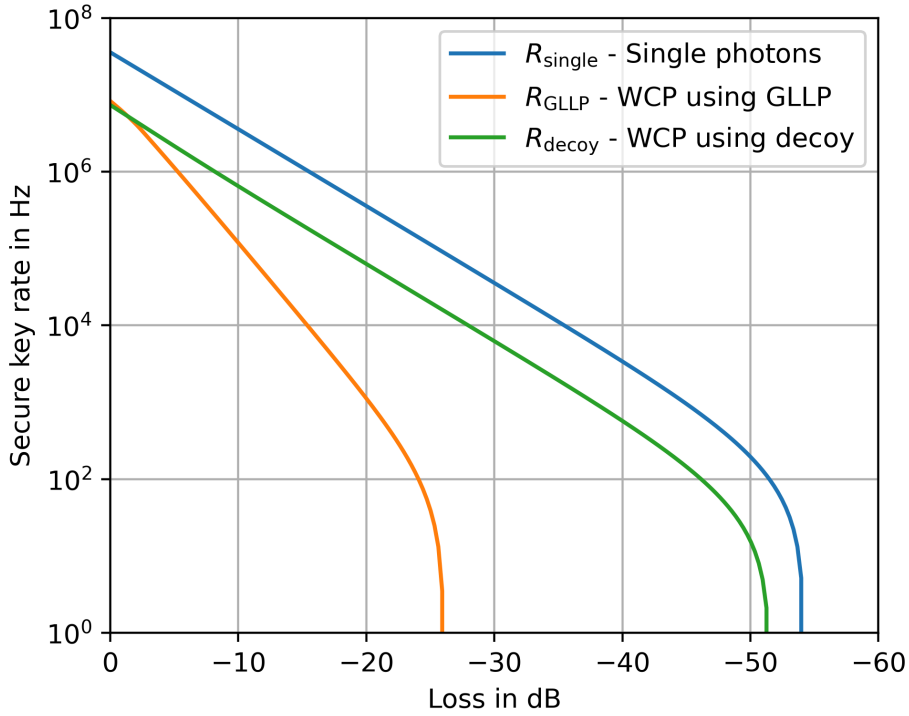

Figure 2.7.: The rate formulas for the BB84 protocol Equation 2.21a (blue), 2.21b (orange) and 2.23 (green) evaluated for different losses. For the GLLP data, the mean photon numbers are optimized in respect to the losses to allow a fair comparison. The decoy figures are derived with a fixed $\mu = 0.5$ and $\nu = 0.15$.

other hand, using the decoy method one is still able to generate a secure key, despite dealing with multi-photon pulses.
Despite proven theoretically unconditional security we must be careful implementing those protocols. The next section gives a brief summary on what technical problems we have to keep an eye on.

2.3.3. QKD vulnerabilities

The previously discussed security proofs always relied on two assumptions:

1. Having an *authenticated* classical channel to work with.

2. Every other *degree of freedom* (DOF) not considered in the security proof has to be identical for every state.

If an implementation of QKD does not fulfill these requirements, the protocol is vulnerable to several attacks and might leak information through a *side channel*. In order to achieve maximal security, one assumes that Eve is not limited in her technological means, but only by fundamental laws of nature. Here we briefly discuss two attacks, a recent review of a variety of additional QKD attacks is given in [6].

Man-in-the-middle attack If the classical channel is not authenticated properly, Eve is able to take the role of Bob, such that Alice generates a secure key between herself and Eve without noticing the presence of an attacker. Using unauthenticated channels, all protocols are insecure. Therefore, Alice and Bob must share a small secret upon executing any classical or QKD protocol. The Carter-Wegman scheme [54] is one of the most used authentication schemes. However, there are developments of quantum authentication protocols [55, 56], which consume a smaller amount of pre-shared secret key.

Trojan-horse attack Eve could be able to retrieve information about the prepared state by sending light pulses into Alice's apparatus and inspecting the back-reflected light, thus uncovering the position of Alice's polarizer. According to Gisin et al. [57], it suffices to equip the sender with wave length and spatial filters and to have the encoding components active only during short times to limit the effectiveness of Trojan-horse attacks.

Side channels are opened if the message is not encoded solely in the intended degree of freedom, but is also correlated to others, i.e., that information on the signal is available by measuring the other DOFs.
Until now we only considered the DOF which are used for QKD. However, real-world QKD devices feature many different DOFs, and if any of them depend on the prepared state, the security of the system is compromised, as Eve would not cause any quantum bit errors by eavesdropping and is therefore not revealed.
Suppose a device prepares the quantum states for QKD with different sources, whose spectral properties may not match, e.g., one source emits light at 870 nm and another one at 840 nm. Then, by simply observing the distinct wavelengths, Eve could gain knowledge about the key. Although this is a rather exaggerated example, it stresses the point that all DOFs of the emitted light that are not an active part of the QKD protocol have to be identical. The available side channels depend heavily on the implementation.
Because our apparatus creates the four polarization states with the use of four different laser diodes, we will now list several examples of possible side channels

important for us.

Temporal If the photons prepared by the individual laser diodes arrive at different times, an attacker is able to measure the arrival times without disturbing the polarization states and thereby obtain full knowledge of the key. By adding a possibility to tweak the pulse duration and arrival time we can match the pulse shapes and hence close this side channel.

Spectral When different lasers shine at distinct wavelengths, the attacker is also able to listen to the transmission without observing the polarization and extract the full key. Ideally, the four different VCSELs of a single array are spectrally indistinguishable. However, manufacturing process variations might introduce deviations of the spectra, which allow an attacker to gain information. This side channel can be closed by either selection of a suitable array with high indistinguishability, a suitable temperature-tuning mechanism or a by using a very narrow spectral filter.

Spatial In the same way, the spatial mode should not allow to obtain information about the sent polarization state. Otherwise, Eve could simply look into the sender and determine the key by observing which of the four laser diodes is lit. Therefore, we use a wave guide to perfectly overlap the spatial modes of all four polarization states, removing all correlation between them and the the spatial mode.

Electric Another side channel emerges from the driving electronics, as the pulse generation might leak some information in form of electromagnetic fields. It is also possible that different laser diodes draw a distinct current and an attacker is able to monitor the overall power consumption. A method to circumvent this is to encase the electronics in a shielded container and let a small current be drawn over a dummy resistor to match the power draw of the lasers.

2.3.4. Summary and motivation

As we have shown, there are multiple crucial pitfalls to avoid while developing a secure QKD device. As we have chosen to use commercially available laser diodes, therefore implementing decoy states to secure the key distribution, we need a design able to quickly modulate the intensity of the pulse. As stray-light and dark counts of the used *avalanche photo diodes* (APD) strongly limit the efficiency of QKD, we also want to make use of time filtering, i.e., only consider detection events within a certain time window. In order to use this

technique, our sender has to generate very short pulses and the clocks of both parties must be well synchronized. Additionally, we also need a way to match all DOFs as discussed previously. The next chapter deals with the development of the Alice sender module and describes the technical challenges, leading to our design decisions.

3. Hardware and software developments for a QKD system

This work is based on a existing sender unit, designed and assembled by Gwenaelle Mélen [58], which is suitable for the BB84 protocol discussed in section 2.3.2.1, utilizing polarization encoded weak coherent pulses. Originating from the electronics used in this sender, C. Sonnleitner [59] designed a mainboard together with modular driver lanes, allowing for rapid prototyping of new pulse generation schemes. As the basic working principle is unmodified by this work, we first introduce the main details of the sender and receiver and continue with the implemented modifications, enabling decoy state generation, and USB3.0 support. The last two sections cover data acquisition and processing at the receiver side as well as synchronization of both sender and receiver.

3.1. Overview and previous implementation

As we want to create a compact integrated sender unit suitable for handheld devices, CubeSats or other small sized network components our most important design aspects are

- small footprint,
- high repetition rate (100 MHz),
- low power consumption($< 10\,\mathrm{W}@5\,\mathrm{V}$),
- portability (no need for external devices),
- low cost.

The first two details are already fulfilled by the previous implementation, while satisfying the third requirement enables the device to be powered by a USB3.0 host. Powering the whole sender unit from this nowadays widely available port

helps meeting the fourth aspect of being portable.
As polarization encoded BB84-like protocols need four different polarization states, four *VCSELs* (Vertical cavity surface emitting laser) are utilized, which come assembled on a 1x4 array[1] by VI Systems. Each diode is responsible for creating one of the four polarization states, by directing each output through a polarizer. For controlling the key exchange, a *FPGA* (Field programmable gate array) is used because they feature high-speed logic operations needed for a 100 MHz repetition rate. We first discuss the electrical parts in depth and later on the corresponding optics.

3.1.1. Electronics

The electronics controlling the micro optical sender assembly consists of three major pieces:

1. Cypress FX – takes care of the USB communication.
2. FPGA – controls the pulse generation, stores the raw key and parameters for the experiment.
3. Pulse Generation – creates short electrical pulses to drive the VCSELs.

For the first two components, the EFM-01 board from CESYS is used, which is plugged into a self-designed mainboard (e.g., Alice Testboard shown in Figure C.3). The EFM-01 embeds a Spartan-3E FPGA[2] from XILINX and a Cypress FX2LP. It comes with two pin headers, the needed voltage regulators and the auxiliary components for both chips. This makes the board versatile and easy to use. Yet, in combination with the Alice mainboard the downsides are limitations of the *input/output* (IO) pins, redundant chips, high price and restriction to USB 2.0.
Via USB/FX2 we can set different operating parameters and transmit the key to the FPGA. The FPGA passes on the desired values to the corresponding chips and later on controls which laser diode sends a pulse according to the stored key.
The Alice mainboard utilizes a 100 MHz differential (see section A.2) clock[3] and distributes this clock signal through a 1:8 clock fanout[4]. As shown in Figure 3.1, we feed one clock signal to each of the four laser driver lanes.
A lane consists of a delay chip[5], a fast AND-gate[6] and a laser driver[7]. The delay chip has two individual ports where every port is able to delay an incoming signal between two and seven nanoseconds. The delay is controlled by a 10-bit

[1] V50-850C4 [2] XC3S500E-4CPG132C [3] LMK61E2-100M00 [4] SY58031U
[5] SY89297U [6] SY58051AU [7] ONET4291VA

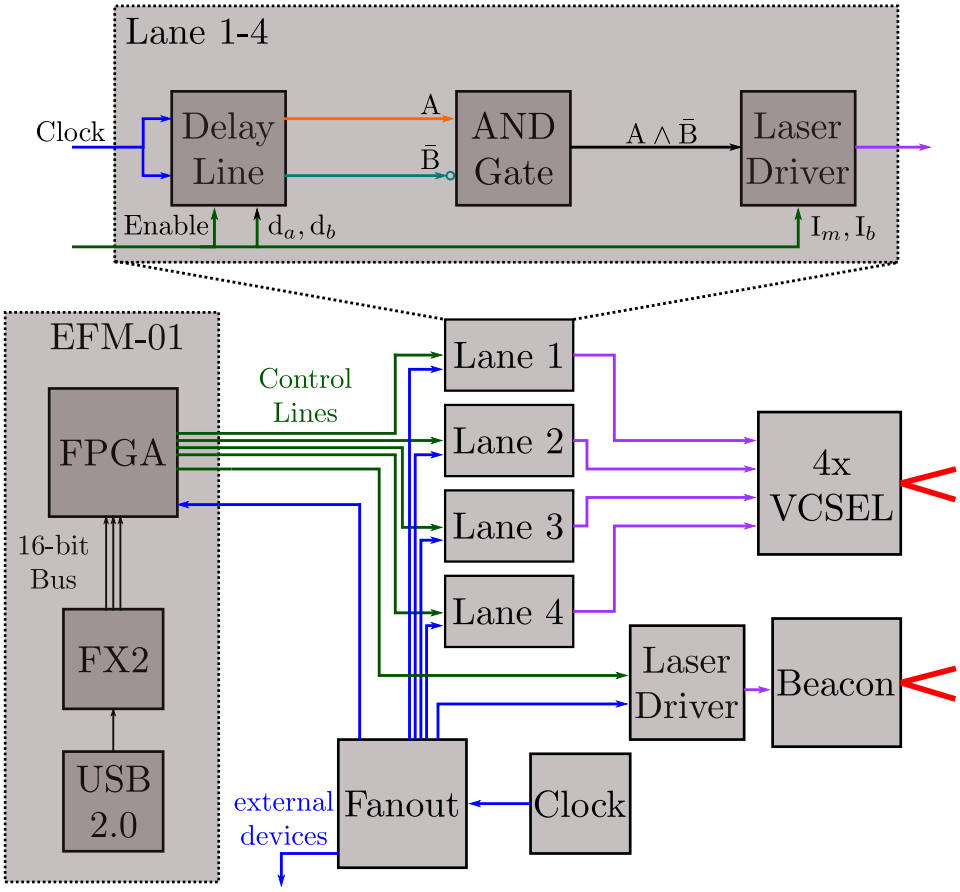

Figure 3.1.: Simplified schematic of existing driver electronics, where the clock signals are shown in blue, control signals in green, electrical pulses in purple and optical outputs in red. The enable signal is a simple logic signal, $d_{a,b}$, $I_{m,b}$ are transmitted via a serial interface.

register d_a (d_b) for channel a (b), providing us $2^{10} = 1024$ steps. Every step delays the incoming signal by $\Delta_d = 5\,\text{ps}$. Both input ports are connected to the clock and the outputs are routed to the fast AND-gate, where one channel is negated. By modifying d_a and d_b we can control the timing when a pulse is sent d_a and simultaneously the pulse width $\Delta = d_b - d_a$ (see Figure C.4).
An additional enable signal controls whether an actual pulse is produced. When this is pulled high, the outputs of the delay line are pulled low and no clock signal is getting through, thus inhibiting the creation of a pulse by the laser driver.
This chip itself is controlled by two 8-bit registers for modulation C_m and bias current C_b. We are therefore limited to $2^8 = 256$ levels. According to the data

sheet, the current values are calculated as

$$I_m = 100\,\mu A + C_m \cdot 68\,\mu A, \tag{3.1a}$$
$$I_b = 100\,\mu A + C_b \cdot 47\,\mu A. \tag{3.1b}$$

Modifying these two values together with d_a and d_b gives us plenty of possibilities to shape our pulses, such that they are overlapping temporally.

3.1.2. Sender optics

The goal of former and still ongoing projects in our group is to develop a miniaturized QKD sender optics. Most of the experimental results of this work are created with a prototype of such an integrated optics. As illustrated in Figure 3.2, the output of each VCSEL is focused using a micro-lens array and polarized using a set of polarizers. Every beam is coupled into a wave guide embedded in a glass structure, written by femto-second laser direct writing [58]. The wave guide circuit is designed such that a quarter of the inserted optical power is coupled evanescently into a single output, respectively. This ensures that the output of the four VCSELs are spatially perfectly overlapping, thus avoiding a corresponding side channel.

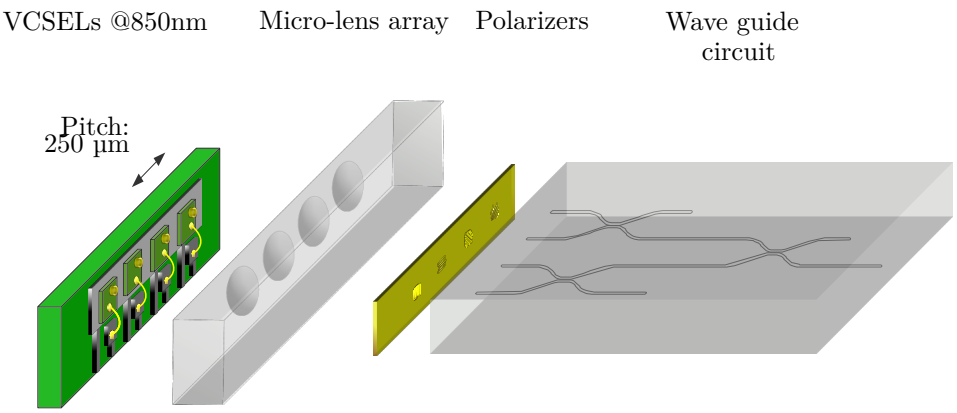

Figure 3.2.: Schematic of the sender optics, where four VCSELs emit light, which is focused by a micro-lens array into four wave guides. A polarizer is mounted between the lens array and the wave guide, such that each diode generates one of the four polarization states. Picture taken from [16].

3.1.3. Receiver optics

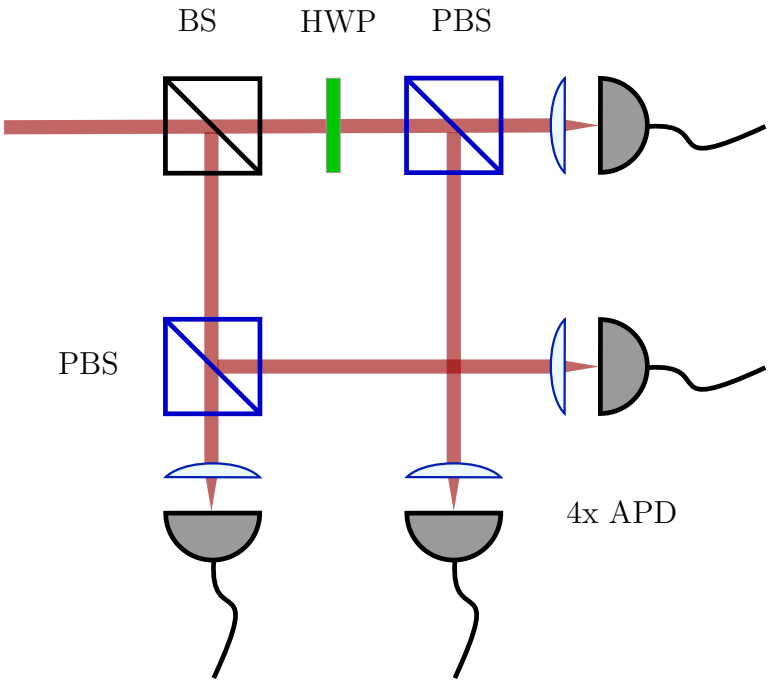

Figure 3.3.: Schematic of the receiver optics, where the first beamsplitter (BS) performs a passive basis choice. A polarizing beamsplitter (PBS) in the reflected arm differentiates between $|H\rangle$ and $|V\rangle$. In the second arm a half wave plate (HWP) rotated by 22.5° rotates the polarization by 45° allowing us to distinguish $|P\rangle$ and $|M\rangle$ after another PBS. Picture taken from [16].

On Bob's side a *polarization analyzer unit* (PAU), as shown in Figure 3.3, is used to distinguish between the four different polarization states. The used 50/50 beamsplitter performs a passive and random basis choice needed for the BB84 protocol. A polarizing beamsplitter reflects light which is polarized perpendicular to the plane spanned by the incoming and reflected beam. In one arm this corresponds to $|V\rangle$ polarized light, whereas the $|H\rangle$ state is transmitted. The other arm features a HWP which rotates the polarization by 45°, this allows us to discriminate between +45° polarized $|P\rangle$ and −45° polarized $|M\rangle$ light. After successfully separating the different polarizations, the photons get sent onto *avalanche photodiodes*[1] (APDs), where the photoelectric effect is used to convert the incoming photon into an electrical signal. The signal events, i.e., the rising edges, are logged by a fast timestamp discussed

[1] PerkinElmer DTS SPCM-AQ4C

in section B.2.

3.2. Decoy state generation

This section describes the development of an electronics for electrically generating decoy states, while making sure they still fulfill our other requirements such as the high repetition rate and short pulse widths. Using the Alice Testboard with the modular approach introduced by [59], different lanes could be connected to the mainboard via a 16-pin flexible flat cable (FFC) connector[1], which allows us to quickly replace and test new driver lanes without replacing the entire electronics.

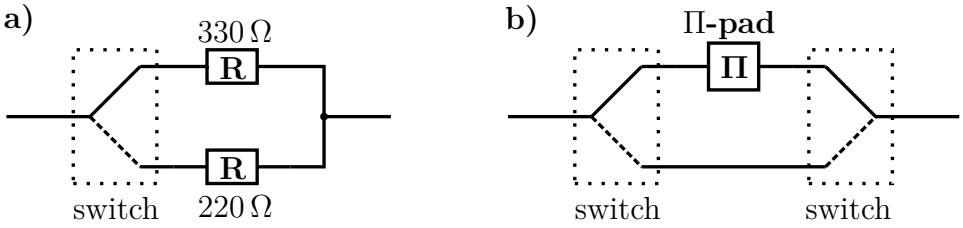

Figure 3.4.: Schematic structure of the driver electronics responsible for modulating the pulse intensity. a) shows the previous implementation with only one switch b) sketches the new improved implementation with two switches and an Π-pad attenuator. The laser driver output is connected to the left side, the VCSELs are attached to the right side.

A previous design of the driver lane connected the output of the laser driver to a fast switch[2]. Via this switch we are able we to run the signal either through a 220 Ω (signal) or a 330 Ω (decoy) resistor, as illustrated in Figure 3.4 a), which results in two different pulse heights. The solution with such a switch entails two major disadvantages along with a high price of about 70€ per chip.

Complementary negative control voltage The used switch requires complementary negative control voltage, which forces us to incorporate a $-5\,\mathrm{V}$ voltage source. Two possibilities to generate such voltage were considered. The first one requires a supply voltage of $V_{\mathrm{sup}} = 10\,\mathrm{V}$, such that we can create a virtual ground at $V_{\mathrm{vg}} = 5\,\mathrm{V}$. This enables us to connect the VCC-pins of all our parts with positive supply voltage to V_{sup} and the GND-pins to our virtual ground V_{vg} resulting in a $V_{\mathrm{sup}} - V_{\mathrm{vg}} = 10\,\mathrm{V} - 5\,\mathrm{V} = +5\,\mathrm{V}$ voltage difference.

[1] FH12_16S-0.5SH [2] HMC347ALP3E

The VSS-pins of parts requiring a negative supply are connected to the real ground of 0 V, whereas the GND-pins are at the virtual ground. This results in a voltage difference of $V_{\mathrm{gnd}} - V_{\mathrm{vg}} = 0\,\mathrm{V} - 5\,\mathrm{V} = -5\,\mathrm{V}$ as needed. Although in principle this method is feasible, it violates our design goal of having only a 5 V power supply.
The other considered method is to use a switching power supply on the board to generate a negative supply voltage. Such power supplies feature low size, high efficiency and any desired output voltage, on the other hand the downsides are a very noisy output voltage and high electromagnetic interference due to fast and sharp current switching, which can be improved using an output ripple filter and differential signaling.
Yet, having a negative supply voltage does not solve the problem of requiring a *complementary* negative *control* voltage. The FPGA outputs positive LVCMOS-signals (low voltage complementary metal oxide semiconductor) ranging between 0 V (logic low) and +3.3 V (logic high). To convert this signal to levels compatible to the switch, we need an auxiliary circuit as shown in Figure 3.5.

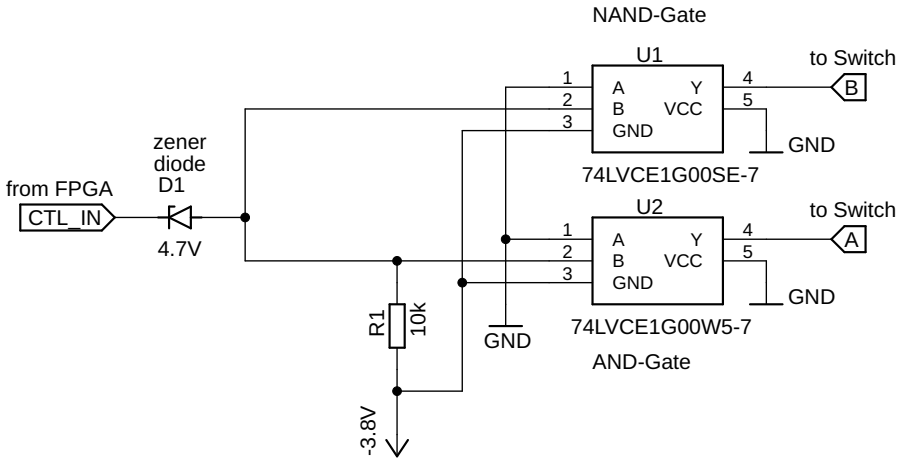

Figure 3.5.: Schematic of the switch interface, where the CTL_IN signal is controlled by the FPGA. The signals A and B form the complementary negative control signal and are connected to the switch. The first input port of both gates (Port A) is tied to GND, which corresponds to logic high.

If the FPGA outputs a low level, the Zener diode D1 blocks because there

are only 3.8V applied. The second input ports of both gates (input B) are therefore at $-3.8\,\mathrm{V}$ corresponding to logic low. U1 outputs a high $0\,\mathrm{V}$, U2 a low $-3.8\,\mathrm{V}$.
When the FPGA sends a high $3.3\,\mathrm{V}$, the voltage across D1 is $7.1\,\mathrm{V} > 4.7\,\mathrm{V}$ and the diode breaks down and becomes conductive. The voltage level on the second input ports of the gates are now at $3.3\,\mathrm{V} - 4.7\,\mathrm{V} = -1.4\,\mathrm{V}$ and therefore at a logical high level. U1 outputs a low and U2 the corresponding high.
This explanation holds for low frequencies where Zener diodes are typically used, but at frequencies $> 100\,\mathrm{kHz}$ the influence of shunt capacitance increases greatly, typically to $C_{shunt} \approx 1\,\mathrm{nF}$, see [60] for further details.
Even "hyperfast" Zener diodes[1] have a reverse recovery time of $t_{\mathrm{rr}} = 16\,\mathrm{ns}$, which is too slow for our $100\,\mathrm{MHz}$ application. Because the capacitance of such diodes is not an intentional feature, one often does not find any information about the value in data sheets. Even worse, two Zener diodes from the same manufacturer may have completely different capacitances due to variations in production batches.
Those problems lead to different rise t_{fr} and fall t_{rr} times of the signal behind the Zener diode, therefore the gates output did not have a 50/50 duty cycle, as seen in Figure 3.6.
According to the data sheet, the switch needs at least $2\,\mathrm{ns}$ to reach a defined state, while we have a time window of $2.6\,\mathrm{ns}$ available. This timing is very strict and hard to optimize. As a result, some pulses of the same type are dimmer than others depending on the preceding state. In order to fix the duty cycle, one needs to use a different Zener diode with a different capacitance. As stated above, the exact value has to be measured, which requires a tedious trial and error procedure.
However there are other level shifter circuits very similar to the former scheme, where the diode is replaced by a capacitor, the pull down resistor is removed and two additional Schottky diodes with auxiliary components are added. See Figure C.5 for a schematic diagram.
The utilized diodes are not crucially necessary because the AND-/NAND-gates exhibit some parasitic diodes, just as any silicone *integrated circuit* (IC) [61]. But the dedicated diodes can handle much more current and therefore support the chip in pulling the voltage level at the working point fast enough. This circuit is preferable because it yields a consistent result after assembling the driver lanes and does not depend on the exact parameters of a single component.
The circuit is designed such that at least every hundredth cycle ($1\,\mu\mathrm{s}$) a transition has to be made or else the capacitor will deplete its charge and needs to be re-pumped for a few cycles again. This is the greatest drawback of this

[1] VS-1EFH02HM3

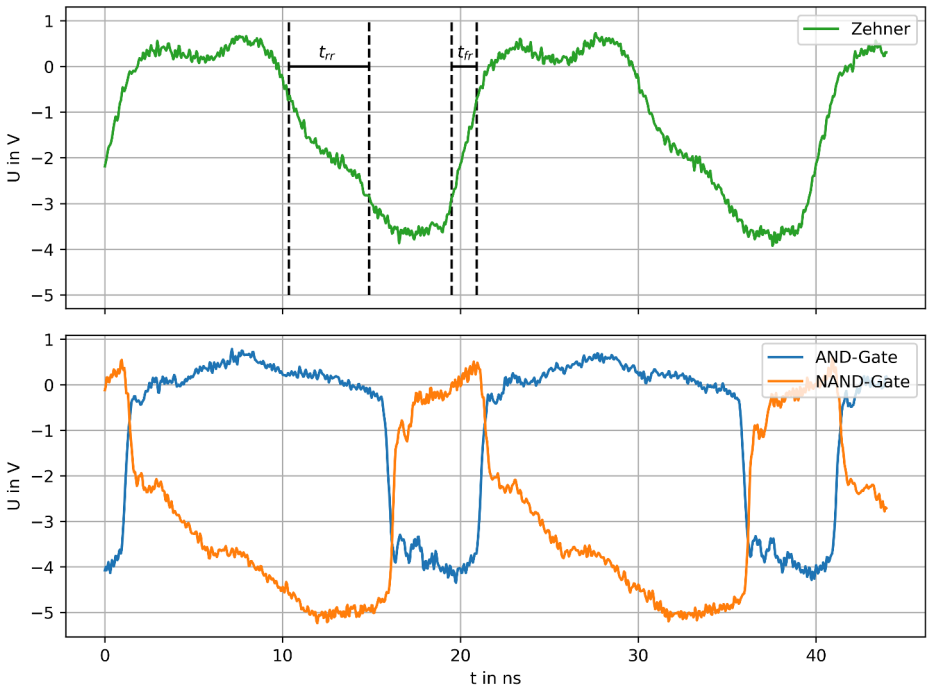

Figure 3.6.: The upper plot shows the voltage behind the Zener diode, which has an larger reverse recovery time $t_{rr} = 4.51\,\text{ns}$ than forward recovery time $t_{fr} = 1.41\,\text{ns}$. Because this mismatch the outputs of the gates, displayed in the lower plot, had a duty cycle of 74/26 instead of the desired 50/50.

method: continuously sending either signal or decoy is not possible, which is an essential feature during the calibration of the device.

Termination of unused port The utilized switch terminates the port which is not active via a $50\,\Omega$ resistor to ground. Although this is useful for most high-frequency applications, it is a big disadvantage for our design. If we want the full brightness when sending signal pulses, we do not want to add any resistor in the signal path. Instead we are forced to include a resistor because, if the signal is running through the decoy path, as illustrated in Figure 3.7 a) and R2 is missing we essentially bypass half of the current destined for the VCSELs via the termination resistor to GND.

In addition to that, as discussed in section A.1, the impedance of the route needs to stay constant and matched to the input/output devices. If we now add a resistor into any of the pulse paths, as illustrated in Figure 3.4 a), we influence this impedance and the signal integrity suffers. To avoid this problem

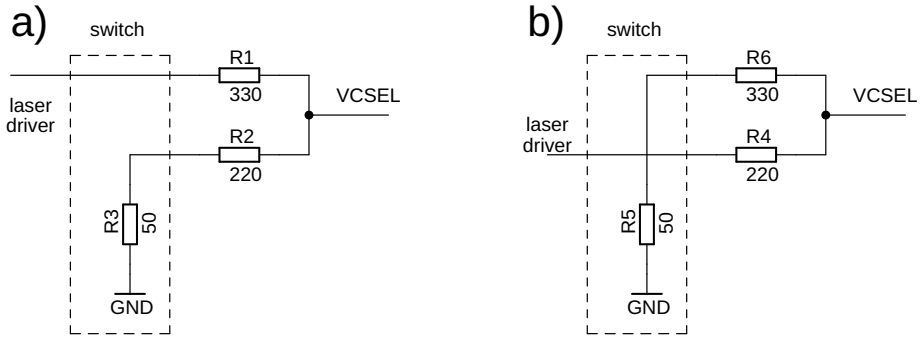

Figure 3.7.: Substitute circuits for the different switch positions. a) shows the switch in decoy mode, where the output of the laser driver is connected to R1, R2 is connected to GND via a 50 Ω resistor. b) pictures the switch in signal position.

in general, one utilizes an attenuator circuit designed to match the impedance of our device. But due to the termination resistor we cannot apply any of those solutions properly.

Summary All those problems forced us to think of a new solution with a different switch and proper attenuation. As shown in Figure 3.4 b) a setup with a Π-pad attenuator and two switches is employed.

3.2.1. Π-pad

As discussed earlier, we want to attenuate a signal while preserving the needed impedance, which can be accomplished by a Π-pad. As the name indicates, this attenuator has a topology similar to the Greek letter Π, pictured in Figure 3.8. There is only one resistor R_1 in the signal line and both the input and output sides are taken to ground via two additional resistors R_2.

If a flat frequency response is desired, only resistors are used, but when they are replaced with inductors and capacitors one is able to filter out specific frequencies [62]. There also exists a T-pad, which performs identically to the Π-pad if resistors are used. Their differences in topology only matter if frequency filtering is needed. If both input and output impedances are the same, $Z_{I1} = Z_{I2} = Z_0$, and we replace the admittance $Y = Z^{-1}$ with the

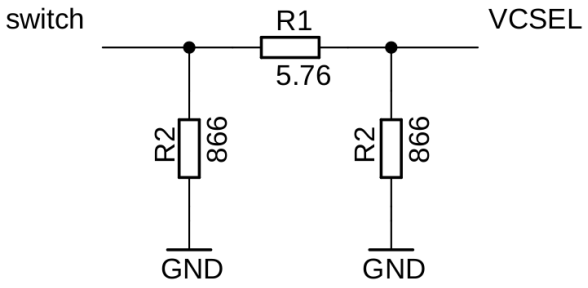

Figure 3.8.: Topology of a Π-pad attenuator, where the resistor values are calculated via Equation 3.4 for a attenuation of $\gamma = 0.115$ and a impedance of $Z_0 = 50\,\Omega$.

impedance Z, we can use (see chapter 3 of [62])

$$\frac{1}{Z_{I1}} = \frac{1}{Z_{I2}} = \sqrt{\frac{1}{Z_1}\left(\frac{1}{Z_1} + \frac{2}{Z_2}\right)}, \tag{3.2}$$

$$\gamma = 2\sinh^{-1}\sqrt{\frac{Z_1}{2Z_2}}, \tag{3.3}$$

where $\gamma = \log\left(\frac{U_{in}}{U_{out}}\right)$ is the attenuation in nepers. We can now solve these equations for Z_1 and Z_2, yielding

$$Z_2 = Z_0 \coth\left(\frac{\gamma}{2}\right), \tag{3.4}$$

$$Z_1 = \frac{2Z_2}{\left(\frac{Z_2}{Z_0}\right)^2 - 1}. \tag{3.5}$$

An attenuation of about 11 % ($\gamma = 0.115$) with an impedance of $Z_0 = 50\,\Omega$ is achieved by setting $Z_1 = 5.76\,\Omega$ and $Z_2 = 866\,\Omega$. Utilizing this kind of attenuator allows for a consistent and easy change of the decoy intensity.

3.2.2. Switches

The previous switch[1] was chosen because it features low rise and fall times ($t_{\text{rise}}, t_{\text{fall}} = 2\,\text{ns}$) and a broadband (from DC to 14 GHz) low insertion loss ($-2\,\text{dB}$). The timings are sketched in Figure 3.9 which illustrates why we need very fast rise and fall times.

[1] HMC347ALP3E

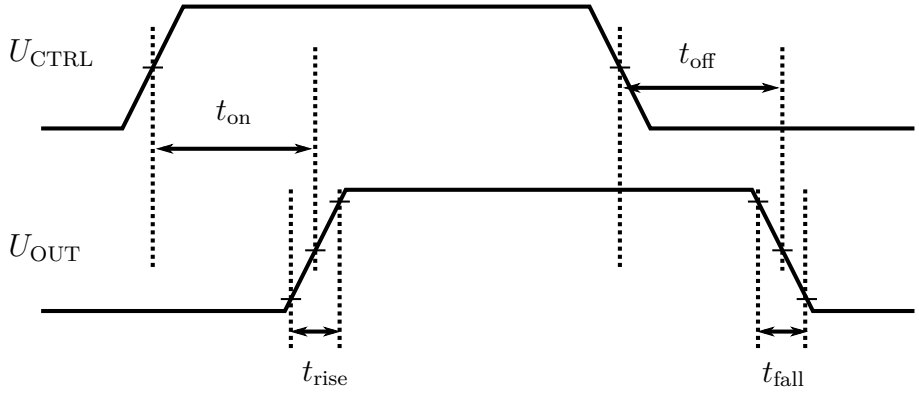

Figure 3.9.: Timing of a switch where U_{CTRL} is the control signal and U_{CTRL} the corresponding output signal. In the diagram there are the marked times $t_{\mathrm{on}}, t_{\mathrm{off}}$ and $t_{\mathrm{rise}}, t_{\mathrm{fall}}$ that are often noted in data sheets.

If those were about 10 ns, the switch would never be in a defined state and our pulsing scheme would not work. If the delay times t_{ON} and t_{OFF} are about 10 ns, while sending with 100 MHz, it does not impact our system since this gives us just a offset which can be compensated by sending the control signals one clock cycle earlier.
The candidates for the switch were chosen with the main focus on $t_{\mathrm{rise}}, t_{\mathrm{fall}} \leq 5\,\mathrm{ns}$. This led to a problem because the majority of fast analog switches are based on a GaAs (Gallium Arsenide) technology, which offers high speed, good linearity and low ON resistance, but most of them are N-channel depletion mode *field effect transistors* (FET) which require a negative gate voltage to turn off.
Having a broadband low insertion loss is also important because we do not want our shaped pulse to be distorted or broadened. In most data sheets of analog switches, the insertion loss is pictured in a frequency-dependent loss diagram, which gives us a sense how the switch will accept different frequencies. If we decompose our short electric pulse into its frequency components via a *fast Fourier transform* (FFT), apply the corresponding loss at this frequency given by the data sheet and then perform an inverse FFT, we get a good appraisal of how the insertion loss of the switch will influence our pulse.
The graphs of this simulation are shown in Figure 3.10, where an electric pulse with $\Delta = 275\,\mathrm{ps}$ is considered. The measured *full width half maximum* (FWHM) of 276.9 ps fits well to the set value. If the losses are constant in all frequencies, one would expect a pulse with the same FWHM, but a reduced maximum. Because none of the tested switches feature a constant insertion loss, they all broaden the pulse depending on their overall irregularity and frequency-dependent losses.

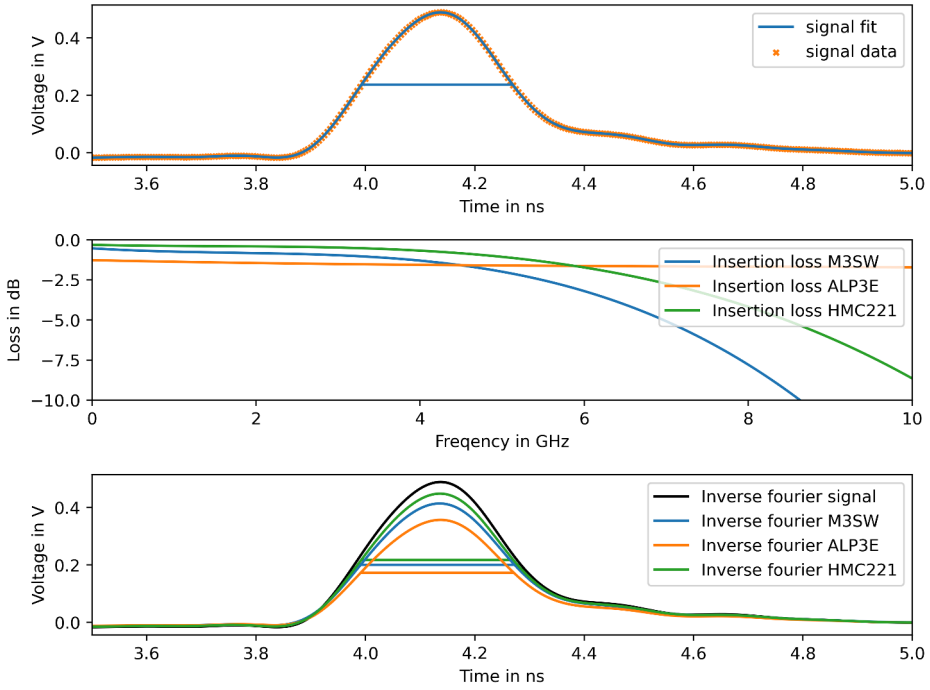

Figure 3.10.: The first plot shows a recorded electrical pulse with $\Delta = 55$ units resulting in a FWHM $= 276.9$ ps marked as the horizontal blue line. The second plot pictures the insertion loss of different switches according to their data sheets. The last subplot shows us the signal pulse after applying the corresponding loss. The M3SW250DR broadens the FWHM to $\text{FWHM}_{\text{M3SW}} = 281.5$ ps, the HMC347ALP3E to $\text{FWHM}_{\text{ALP3}} = 280$ ps and the HMC221B to $\text{FWHM}_{\text{HMC221}} = 278.5$ ps.

According to the FFT, the major part (99%) of the signal is made up by frequencies below 5 GHz. This is the reason why the HMC221B performs equally in terms of FWHM as the HMC347ALP3E despite having a significant drop in insertion loss above 5 GHz. Another value to look at is the maximum voltage of the peaks, which depends mostly on the average insertion loss. Here the HMC221B features the lowest loss and therefore the highest pulse, transmitting 92% of the signal strength.
Considering all those results, the HMC221B from Analog Devices is chosen, exhibiting better performance than the old HMC347ALP3E and no need for a negative control voltage. The relatively low cost of about 2€ made the use

of two switches possible, which at the same time eliminates the problem of termination of the unused port. One drawback of this switch is the need of AC-coupling capacitors on the in- and outputs, because the switch is not able to handle frequencies of $< 10\,\text{kHz}$. This limitation is acceptable because our base frequency is 100 MHz and the DC current is routed around the attenuation circuit anyhow. The schematic of one driver lane is shown in Figure C.6.

3.2.3. Bias current

Tests showed that when routing the whole signal after the laser driver through a switch and attenuator, the VCSELs do not emit any light at all, even when using a switch suitable for DC currents. After injecting a small DC current using a external power supply, the VCSELs showed their typical behavior. It is therefore necessary to only route the modulation signal separated from the constant bias current through the decoy network. The laser driver's data sheet specifies two separated output pins: BIAS, for the constant DC offset and MOD+, the output of the electrical pulse composed of AC currents. Additionally, it suggests a mixing circuit called a Bias-T, consisting of an inductor and a capacitor, shown in Figure 3.11. The inductor is in series with the BIAS, the capacitor in series with the modulation output. The latter stops the DC-current from the bias output interfering with the modulation pin and the inductor blocks any high frequency signals emitted by the MOD+ pin interfering with the bias generation. Treating both the AC and the DC parts of the signals separately, only attenuating the AC modulation current, solved the issue of VCSELs not lighting up.

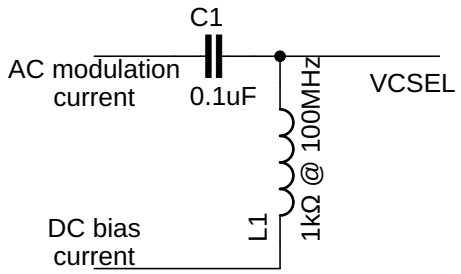

Figure 3.11.: Schematic of a Bias-T consisting of a inductor, blocking the high frequency parts and a capacitor stopping DC current. The values are chosen according to the data sheet of the laser driver (ONET4291VA).

The laser diodes and drivers utilized are typically used in a scenario where

laser light is always present and is only modulated a bit brighter or dimmer. Our usage, as opposed to the typical one, ideally requires no light present at all while only emitting a dim pulse if a state is sent. The desired operation mode with no bias current and therefore no laser emission is not possible, as the VCSELs wont light up even with the highest possible modulation current set. Yet, using the smallest settable bias value of 147 µA, one does get detectable spontaneous emissions and therefore a contribution to the noise level. Switching between zero and some required small bias current between individual modulation peaks (10 ns) is not possible because of the relatively slow serial control interface of the laser driver, requiring a minimum of 100 ns for transmitting each bit.

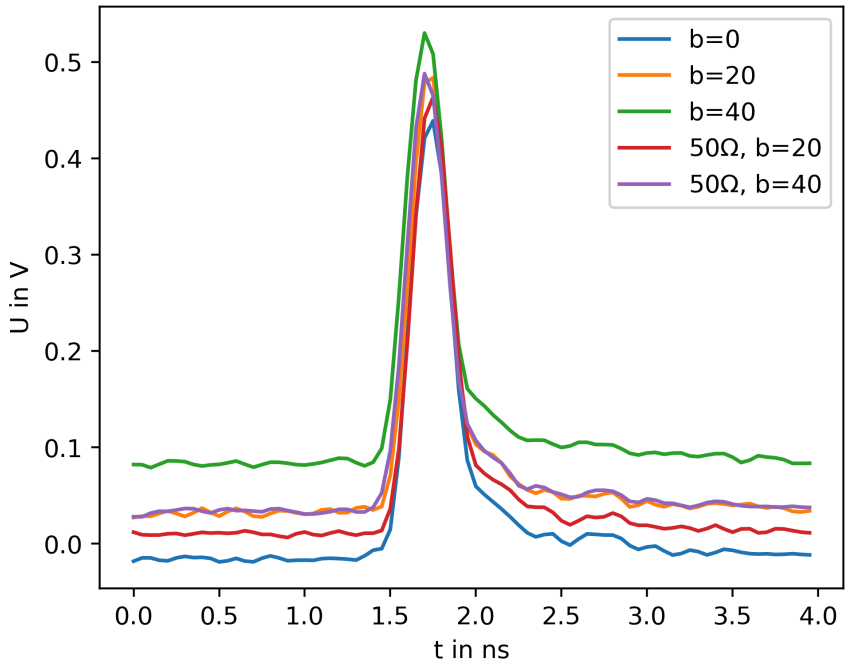

Figure 3.12.: Electrical pulse forms with full modulation (m=255) a set pulse width of 375 ps ($\Delta = 75$) and different bias settings with and without a 50 Ω resistor parallel to the laser diode. The connection to the oscilloscope was terminated with 50 Ω.

As the VCSELs specify a 80 Ω differential resistance, one possible solution is to introduce a resistor into the bias path before the Bias-T, connecting it to ground via 50 Ω effectively dividing the bias current by three. Figure 3.12 shows the electrical pulse with and without such a resistor, seemingly con-

firming this method. Note, as the oscilloscope was terminated with 50 Ω, one would expect a halving of the bias current.
But when connected to such a circuit, the diodes do not produce any light, no matter what modulation current is set. Even at a bias level of 1.45 mA the VCSELs do not emit any light, whereas without the resistor the optical power output is in the range of mW. Additionally a 330 Ω resistor was tested yielding the same results. As Figure 3.13 shows, the VCSELs exhibit a differential resistance of 80 Ω only at currents beyond the lasing threshold. Before the threshold is reached the resistance is close to 3.5 kΩ. This non linear differential resistance leads to the observed difficulties. One possible method to circumvent this is by adding a diode in series to the resistor, blocking at low voltages (before the lasing threshold is reached) and conducting and therefore affecting the current at voltages above 1.8 V. But this method requires further investigation.

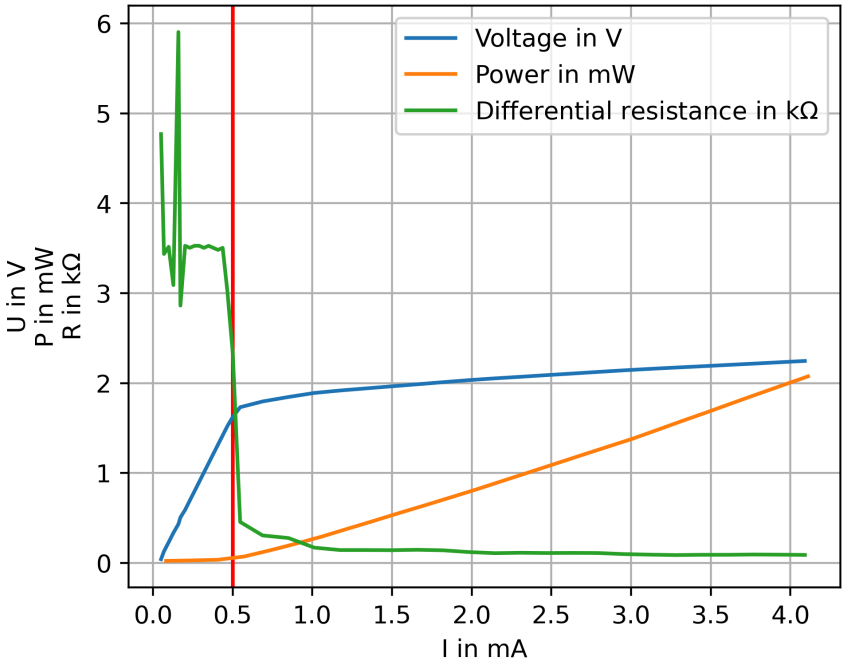

Figure 3.13.: Characteristics of the laser diode with varying operation current. The blue line indicates the voltage in V, the orange line shows the optical output power and the green line calculated with $\Delta R = \frac{\Delta U}{\Delta I}$ illustrates the differential resistance. The vertical red line marks the lasing threshold at 0.5 V. The voltage and power figures are taken from the data sheet of VI Systems' V50-850C.

3.3. Synchronization

Synchronization is a crucial task in QKD, as one has to make sure that Alice and Bob's clock are running at the same pace, or else executing tasks like basis reconciliation is not possible due to timing errors. Furthermore, accurately synchronized clocks enable the parties to time filter, i.e., only consider events in a specific time interval and therefore reducing the impact of background or dark counts.
In an extreme case where the clock frequencies of both parties differ by 10%, every tenth clock cycle the timing error is increased by one whole clock period, resulting in a huge confusion while talking about a specific pulse, e.g., Alice's 42th pulse would correspond to Bob's 38th. Even worse, if the frequency difference is continuously changing we cannot measure and correct for the difference only once at the beginning, but need a way of repeatedly correct for the current difference.

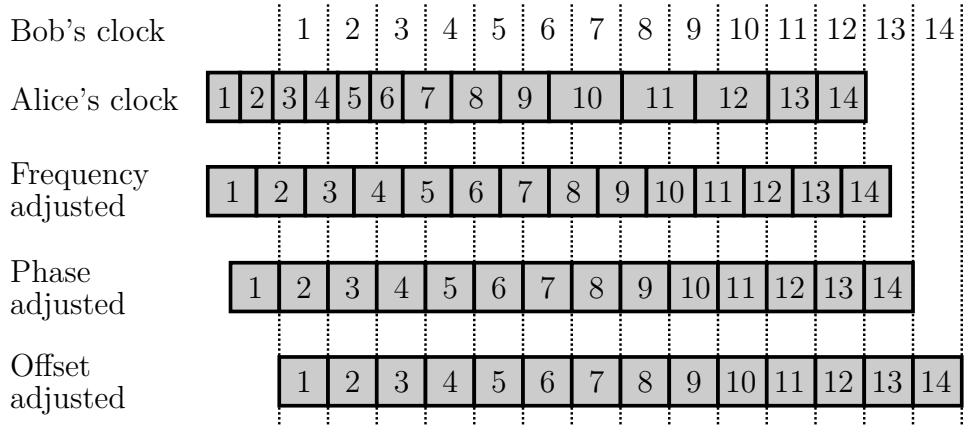

Figure 3.14.: This picture sketches the idea behind synchronization, where the gray boxes represent Alice's clock cycles and the dotted lines show Bob's. The frequency of Alice's clock drifts compared to Bob's clock, which in general varies too. The first step adjusts the frequency of Alice's clock to match Bob's, after which the phase is compensated for, such that cycles of both clocks begin simultaneously. The last step corrects the offset such that both parties can talk about the same clock cycle.

Our used 100 MHz clock[1] has a frequency tolerance of ±50 ppm resulting in a maximum frequency shift of ±5 kHz, where this figure includes temperature

[1] LMK61E2-100M00

and supply voltage variation as well as aging effects over the span of 10 years. The observed frequency shift was about 80 to 140 Hz between Alice's and Bob's (the timestamp units') clock for most of our experiments.
After a successful frequency synchronization there is still a need for detecting when the key transfer has started. Dealing with a high loss environment (−50 dB) only 0.001% of the pulses are detected. The first pulse detected by Bob is therefore most certainly not the first one sent by Alice, correcting for this offset is also part of the QKD synchronization. Both problems can be solved either over the quantum or the classical channel and the next subsections will present some solutions to them.

3.3.1. Clock recovery

A possible strategy for synchronization is via an additional laser at the sender site, modulated with Alice's clock. The receiver can recover the clock signal by detecting the beacon light with a fast photo diode followed by an appropriate clock recovery electronics. Such a technique, has been successfully used in [16]. Here the output of the photo diode was sent to a clock recovery chip[1], which is responsible for locking onto the data stream and extracting a clock and a data signal from it. Both signals are then sent to a FPGA (on an EFM-01 board) where the clock gets divided by 500 resulting in a 200 kHz signal, which is suitable for the timestamp unit discussed in section B.2. The undivided 100 MHz clock would overflow the timestamp unit as it will have problems transmitting all the timestamps via USB2.0 to the PC.
After recording all time dependent events of the experiment, one is now able to interpolate the timestamp of an APD event t_{APD}, lying between two recorded clock events $t_{\mathrm{before}}, t_{\mathrm{after}}$ while assuming a fixed 5 µs period

$$t_{\mathrm{interp}} = t_{\mathrm{before}} + (t_{\mathrm{APD}} - t_{\mathrm{before}}) \cdot \frac{t_{\mathrm{after}} - t_{\mathrm{before}}}{5\,\mu\mathrm{s}}. \tag{3.6}$$

Using this compensation Bob can now evaluate the timestamps as if Alice would have sent them with a steady 100 MHz repetition rate.
In addition to that the data output of the clock recovery chip can be used for detecting a starting condition. If Alice is modulating the beacon with only ones in the 3 preceding clock cycles before any key transmission is started, the FPGA on Bob's site can scan for this specific pattern ("010111") at the data line and output the according start signal. This way both frequency, phase and offset synchronization is achieved.

[1] ADN2814

3.3.2. Software implementation

Other methods have to be utilized if no other dedicated communication channel is available. They have to rely on the signals exchanged via the quantum channel, which are later used for the key. The following method described is adopted from [63] and a offset detection still needs some further research.
For detecting the current frequency of Alice's clock a FFT is executed (see Figure 3.15), this is done by expressing the timestamps t_i as a discrete signal

$$s(t_i) = \sum_j \delta_{t_j, t_i}, \tag{3.7}$$

where the signal is 1 if an event happened at t_i and 0 otherwise. As *discrete*

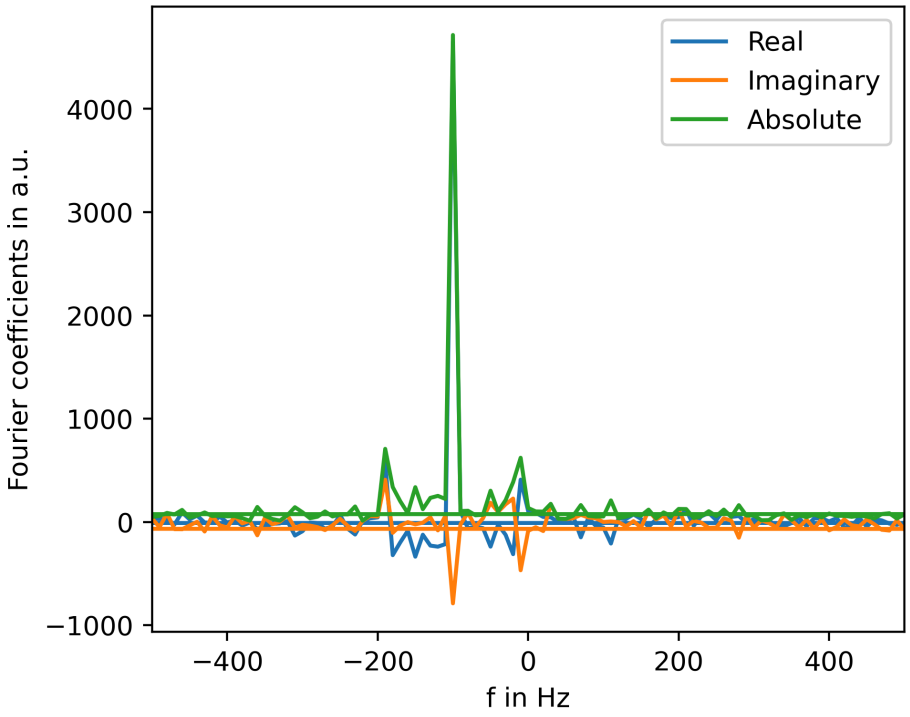

Figure 3.15.: The events of the APD corresponding to the plus polarization, evaluated with Equation 3.9 and Fourier transformed. A sampling time of $T = 0.1\,\mathrm{s}$ and a sampling rate of $f_{\mathrm{sampling}} = 10\,\mathrm{kHz}$ was chosen. One can see the coarse resolution of $\delta f = \frac{1}{T} = 10\,\mathrm{Hz}$ and the peak at $\Delta f_{\mathrm{FFT}} = -100\,\mathrm{Hz}$.

Fourier transforms (DFT), such as FFT, depend on the sampling rate, one has to make sure that the highest sampled frequency, in our case $f_{\mathrm{max}} = f_{\mathrm{base}} =$

100 MHz, is less than the Nyquist frequency $f_{\text{nyquist}} = \frac{f_{\text{sampling}}}{2}$. The resolution of a DFT is limited by

$$\delta f = \frac{f_{\text{sampling}}}{N} = \frac{1}{T}, \tag{3.8}$$

where N is the number of recorded data points and T is the considered time window. As we need to determine the frequency up to $< 1\,\text{Hz}$ and given the resulting sampling frequency of $f_{\text{sampling}} \geq 200\,\text{MHz}$ leads to sampling time of $T \geq 1\,\text{s}$, corresponding to 200 million data points. Executing a FFT on that amount of data would take quite some time, in addition to that the frequency has to be stable in the considered time T.
As we know our frequency is close to 100 MHz we can shift the origin of our FFT by mixing the incoming signal with $\mathrm{e}^{-i\omega_l}$, where $\omega_l = f_{\text{base}} - \frac{f_{\text{nyquist}}}{2}$ resulting in

$$s_{\text{shifted}}(t_i) = \sum_j \delta_{t_j,t_i}\, \mathrm{e}^{-i\omega_l t_i}\,. \tag{3.9}$$

This allows us to only consider a small interval around f_{base}, reducing the needed computing power for the FFT, as we can sample more coarsely. Taking the frequency instability of Alice's clock $\pm 5\,\text{kHz}$ we get a sampling frequency of $f_{\text{sampling}} \geq 10\,\text{kHz}$.
To further improve our frequency resolution and determining the initial phase offset Φ_0 of the Alice's and Bob's clocks, we can make use of the fact that if two clocks run out of sync with a constant frequency difference, their phase will develop linearly in time. The phase of the timestamps, calculated by

$$\Phi(t_i) = \Big(t_i \cdot (f_{\text{base}} + \Delta f_{\text{FFT}})\Big) \mod 1, \tag{3.10}$$

where Δf_{FFT} is the frequency determined by the FFT, give us a measure how much our clock frequency drifts over time (see Figure C.8).

If we reduce our considered time, such that the difference in frequency can be considered constant (see Figure 3.16), we get a very accurate estimation of the real frequency

$$f_{\text{real}} = f_{\text{base}} + \Delta f_{\text{FFT}} - \Delta f_{\text{Fit}} \tag{3.11}$$

where f_{Fit} is the frequency obtained by fitting a linear function onto the phase-time diagram. This method works fine even with very small considered time windows $< 0.1\,\text{s}$, as shown in Figure C.9.
According to [63], a feedback loop can be implemented such that this first, still computational expensive synchronization must only be done once and after suitable initial values are found one can correct for any small frequency drifts by continuously monitoring the arrival time and correcting the values, if the last few photons arrived earlier or later than expected.

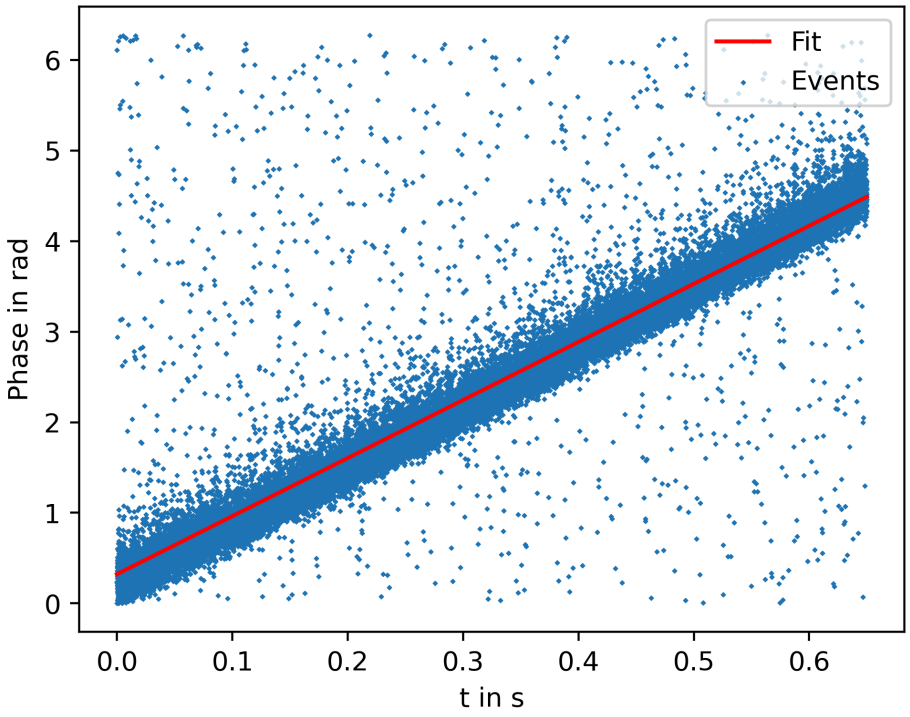

Figure 3.16.: A small sector of Figure C.8, where the phase $\Phi(t_i)$ (see Equation 3.10) of the an event happened at time t_i is shown, considering time frame of 0.65 s. The slope $m = (6.412 \pm 0.007)\,\mathrm{rad\,s^{-1}}$ of the fit corresponds to the difference between the frequency assumed by the FFT and real frequency $\Delta f_{\mathrm{Fit}} = \frac{m}{2\pi} = (1.020 \pm 0.001)\,\mathrm{Hz}$ and the y-intercept marks the phase offset $\Phi_0 = (0.157 \pm 0.002)\,\mathrm{rad}$.

4. Experimental results

This chapter covers the characterization results of our newly designed driver electronics (see Figure 4.1 and C.7) capable of varying single modulation peak intensities with a clock frequency of 100 MHz. This hardware feature allows much higher key generation rates as now the BB84 protocol with decoy extension can be implemented. Our design has the advantage that signal and decoy states are generated by the same driver line and laser diode.

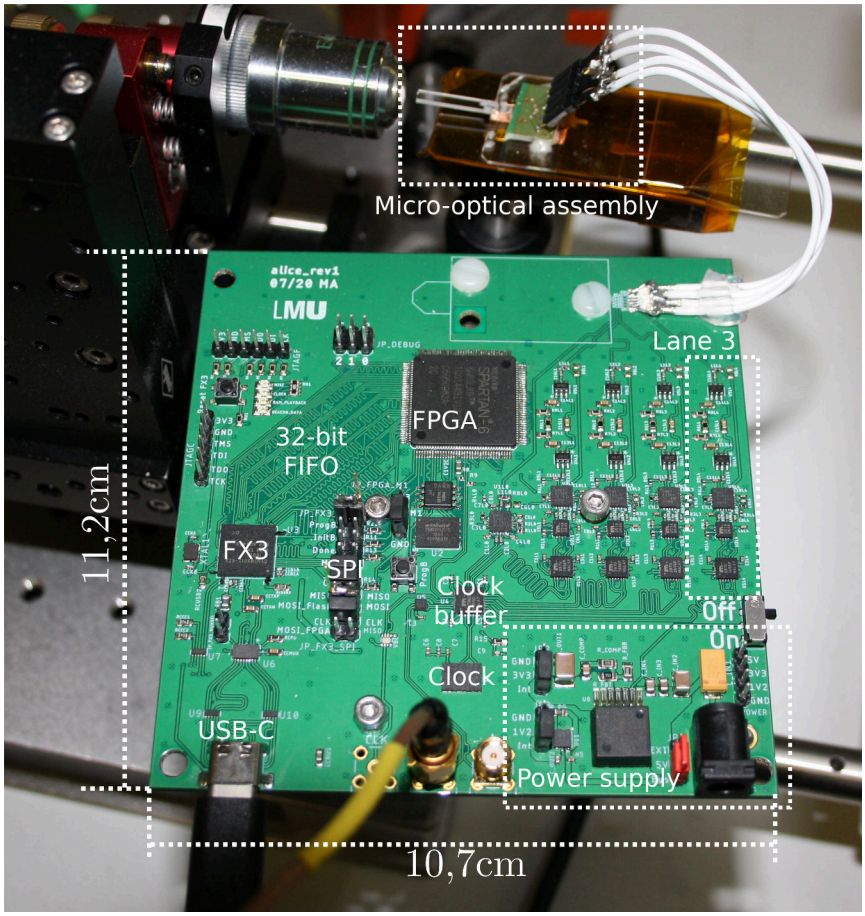

Figure 4.1.: A picture of the assembled electronics, together with the micro-optical assembly located on top, used for the measurements. The most important parts are marked and labeled in white.

In all characterization steps we strongly focused on acquiring data collected in representative measurements, where every driver line of the electronics was connected to a VCSEL and randomly activated, similar as in a QKD modus. Doing so, effects introduced by electrical crosstalk or eventual timing problems in simultaneously controlling all four driver lines are accessible. However, it requires precise time tagging (see section B.2), synchronization (see section 3.3) and sophisticated data processing. In the following we discuss our results on the signal and decoy level discrimination, temporal pulse shapes and the polarization analysis.

4.1. Decoy level analysis

To validate different intensities of signal and decoy pulses, we continuously send a $N_{\text{key}} = 626$ character long key with 20 header bits. The 606 data bits in the key are chosen randomly, with the constraint of observing every permutation of bit orders, e.g., "HP", "PH", "vM", ..., at least six times. In this notation the letter itself stands for the polarization, where a lower case indicates decoy intensity and a capital letter hints a signal intensity. Note that here the probability to send a decoy pulse is 50% while in a typical decoy QKD scenario the probability is reduced.
After analyzing the polarization of every pulse with the PAU and recording the corresponding events with the timestamp unit, every timestamp t_i is sorted into the corresponding time bin j by projecting onto the key's time interval using euclidean division and further dividing it by $t_{\text{bin}} = 10\,\text{ns}$ while rounding to the next lowest integer

$$j = \text{floor}\left(\frac{t_i \pmod{t_{\text{key}}}}{t_{\text{bin}}}\right), \tag{4.1}$$

where $t_{\text{key}} = N_{\text{key}} \cdot 10\,\text{ns}$ is the time one key repetition takes. The evaluation script is now able to determine the offset needed for synchronization by searching for the pattern of the header used ("HHHP"x5), such that we can be sure that every event is sorted into the respective bin. The phase offset is set such that the pulses are centered around the middle of the bins.
If we now divide the number of events in every bin by the number of times the key was repeated $N_{\text{rep}} \approx 6 \times 10^6$ we obtain the probability to observe an event of the respective APD in the corresponding time bin. Those probabilities are illustrated in Figure 4.2, and confirm that the electronics assembled in this work, reliably produces two distinct laser intensity levels for all four polarization states ($|H\rangle, |V\rangle, |P\rangle, |M\rangle$).

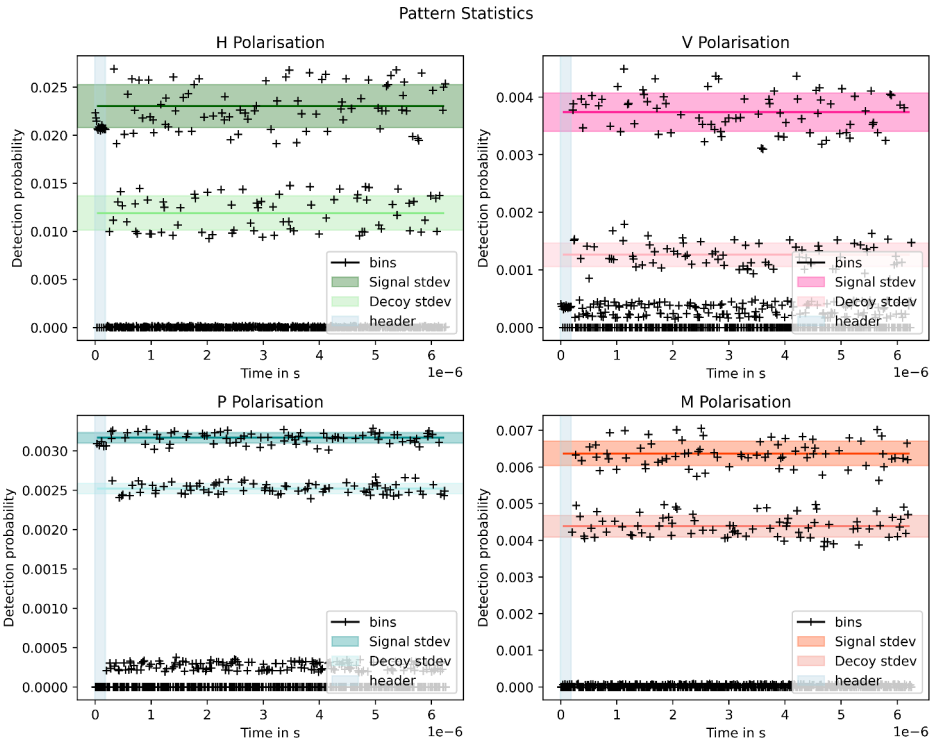

Figure 4.2.: The detection probabilities of the four different polarization states, where the corresponding line marks the mean value and the area indicates the standard deviation. The light blue vertical area indicates the header (”HHHP“x5), which is used to determine the synchronization offset.

The levels shown are distinct but the ratio between signal and decoy detection probabilities of different polarizations do not match, e.g., the ratio of vertical polarized light is lower than the one of plus polarized light. Additionally they do not reassemble the proposed ratio by Ma et al. of 10.4%. However, the relative detection levels would be adjusted by modifying the attenuation of the Π-Pad, thus satisfying the proposed ratio with all four polarizations.
Another property where the different polarization states become distinguishable is the overall intensity. Figure 4.2 shows that the probability to detect horizontally polarized light is seven times higher than plus polarized light. There are several reasons for this intensity mismatch:

- Due to manufacturing, handling and connection differences every laser diode exhibits deviations in output power despite being driven with the same current.

- Even in pulsed mode the DOP of the bare VCSELs is non zero, which leads to different intensities after the corresponding polarizer.
- As VCSEL array, lens array and wave guide circuit have fixed distances between the four spacial modes, it is hard to achieve the same coupling for all four light paths.

In order to get equal optical output power for all four lines, we found following driving parameters:

state	channel	bias	modulation	d_a	d_b	cd
H	0	1	96	200	257	0
V	1	1	148	200	285	0
P	2	1	255	200	260	0
M	3	1	150	200	243	0

Table 4.1.: Laser parameters resulting in equal intensities (counts/s). Bias, modulation, d_a and d_b are explained in section 3.1.1, whereas the clock delay (cd) shifts the whole clock by 25 ps per step, as described in section B.1.2.3. Note that the temporal position is not optimized with this settings.

However, given these settings we could not reach equal ratios for signal and decoy intensities for every channel. Especially for the zeroth channel we observed a very dim pulse if a decoy level was chosen. This behavior can be explained by the highly non-linear optical response of the VCSELs. Even smallest changes in pulse height and duration can completely change the intensity settings. For investigating whether every line can switch between signal and decoy intensity we were forced to increase the modulation intensity for channel H and P, where the pulse parameters previously had to be chosen rather low to match the power of the other channels:

state	channel	bias	modulation	da	db	cd
H	0	1	180	50	115	-10
V	1	1	245	50	107	-10
P	2	1	255	50	125	-10
M	3	1	200	50	207	-10

Table 4.2.: Laser parameters used forFigure 4.2, where signal and decoy levels are nicely separated, even with no individual decoy attenuation for each line.

Using these parameters the overall intensities exhibit a mismatch, but they enabled us, at the very end of this master's thesis, to proof the functionality of the electronics, without modifying the Π-Pad. For a real-world QKD application a better set of laser parameters has to be found and the individual attenuation of each line must be adjusted, such that the intensity levels of the four polarizations, and their decoy ratio match each other.
Ideally the recorded events should follow a binomial statistics as for every sent state, a photon is detected (with the probability p_{det}) or not $(1 - p_{\text{det}})$. As more thoroughly discussed in section 4.3, only the events corresponding to plus polarized states behave as expected.
Nevertheless the electronics reliably produces distinct intensity levels and there are possibilities for tuning every mismatch observed.

4.2. Pulse shape

As we are creating pulses with mean photon numbers $\mu < 1$, according to Figure 2.6, most of our pulses contain only one photon, if any. Therefore talking about the shape of such a pulse is not applicable, as a single photon does not have any temporal shape. Pulse shape, in that sense, is the probability distribution of the arrival time of the photons, measured over many thousand pulses.
For analyzing this temporal shape, we generate a histogram of the pulse arrival times against the reference clock, shown in Figure 4.3. It pictures the distributions of the eight different sent states, and shows that sending a decoy pulse does not affect the timing nor pulse shape, despite the intentional change of the pulse height.
This figure is generated from the same data used in the previous analysis utilizing the same binning method. The only difference is the finer binning time of $t_{\text{bin}} = 100\,\text{ps}$ used in Equation 4.1. As we are synchronized well we can identify bins representing the same state and average over them to get a good impression of the temporal pulse shape.

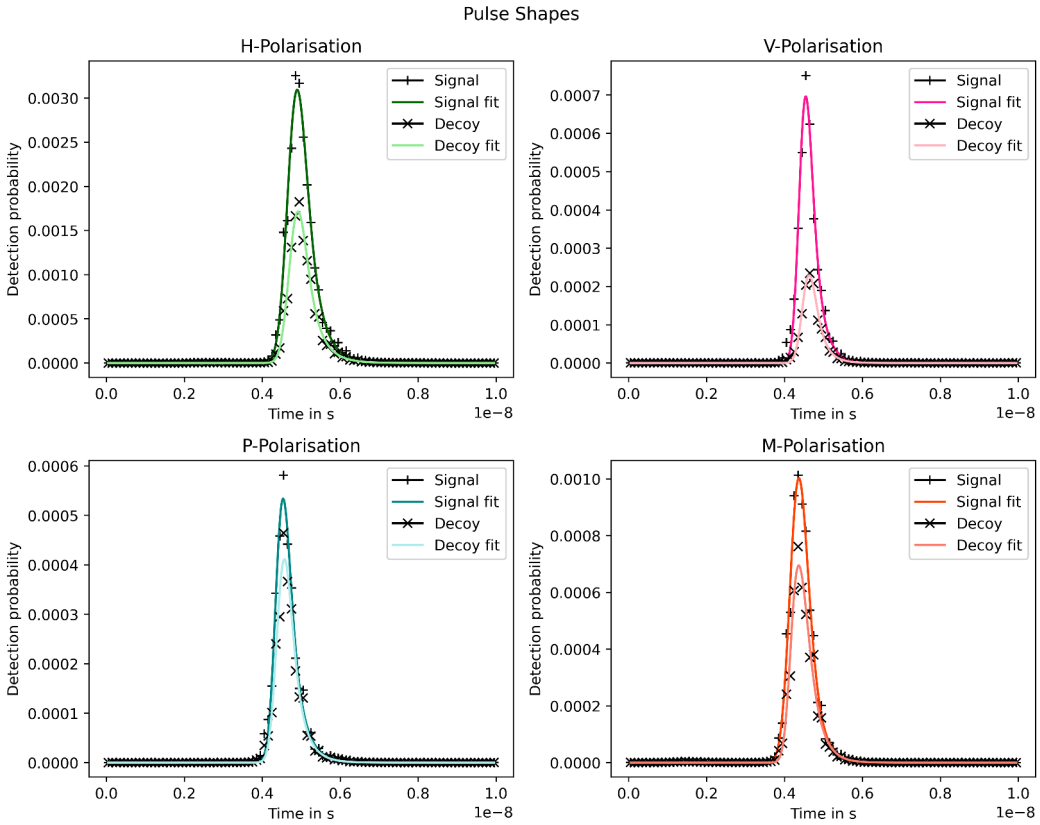

Figure 4.3.: The unnormalized pulse shapes of all respective states. The pulses are recorded with laser parameters listed in Table 4.2 and additional properties are stated in Table 4.3. A binning interval of 100 ps is chosen, limited by the timestamps precision. Note that the H and P polarized light is shifted by 3 ns to coarsely match the arrival time of the other polarizations.

state	$\sum$ det. prob. signal	$\sum$ det. prob. decoy	det. prob. ratio	overlap	fitted overlap
H	2.3%	1.23%	53.48%	99.78%	99.75%
V	0.38%	0.13%	34.21%	99.29%	99.40%
P	0.33%	0.26%	78.79%	99.80%	99.82%
M	0.67%	0.46%	68.66%	99.74%	99.50%

Table 4.3.: Calculated pulse parameters derived from Figure 4.3. The summed detection probabilities of signal and decoy pulses, as well as their ratio is listed together with the overlap of both respective distributions, calculated with Equation 4.3.

The function used to analyze the detection probabilities, composed of the convolution product of two exponential decoys and a Gaussian profile, describes the temporal shape of our laser diodes and is derived in [64], section 5.2:

$$p(t) = a \cdot \operatorname{erfc}\left(-\frac{t-t_0}{\tau_G}\right) \cdot \left[\frac{1}{\tau_{e_1}} \exp\left(-\frac{\tau_G^2}{4\tau_{e_1}^2}\right) \exp\left(-\frac{t-t_0}{\tau_{e_1}}\right) + \frac{1}{\tau_{e_2}} \exp\left(-\frac{\tau_G^2}{4\tau_{e_2}^2}\right) \exp\left(-\frac{t-t_0}{\tau_{e_2}}\right)\right], \tag{4.2}$$

where a is the amplitude, τ_G the standard deviation of a Gaussian profile, τ_{e_1}, τ_{e_2} the two exponential decay constants and t_0 marks the pulse center. The experimentally determined values are listed in Table C.1, additionally the *root mean square error* (RMSE) is calculated, which is smaller than 1.05×10^{-4} for every pulse and therefore indicating a good fit.
The overlap between signal and decoy pulse is calculated according to [64] as

$$o = \int \sqrt{p_{\text{signal}}(t) p_{\text{decoy}}(t)} \mathrm{d}t \approx \sum_t \sqrt{p_{\text{signal}}(t) p_{\text{decoy}}(t)}, \tag{4.3}$$

where the integral is approximated by the sum over discrete values.
Figure 4.3 and Table 4.3 show promising results in terms of timing uniformity as the temporal overlap of signal and decoy pulses is above 99% for every polarization. Nevertheless one needs to calculate the mutual information emerging from the (still differing) pulse shapes, to estimate the information an eavesdropper can harness exploiting this timing side channel. But this is beyond the scope of this work.

4.3. History

As the previous analysis has shown, the detection probabilities for different intensities are distinct, and feature some statistical fluctuations. Especially the channel responsible for creating H polarized light is subjected to such variations (see Figure 4.2) which are larger than one would expect from a binomial statistics. It is possible that the detection probability and therefore the intensity of a pulse is depending on the polarization state that was sent before the current one (predecessor) or on the one sent afterwards (successor), consequently opening a side channel.

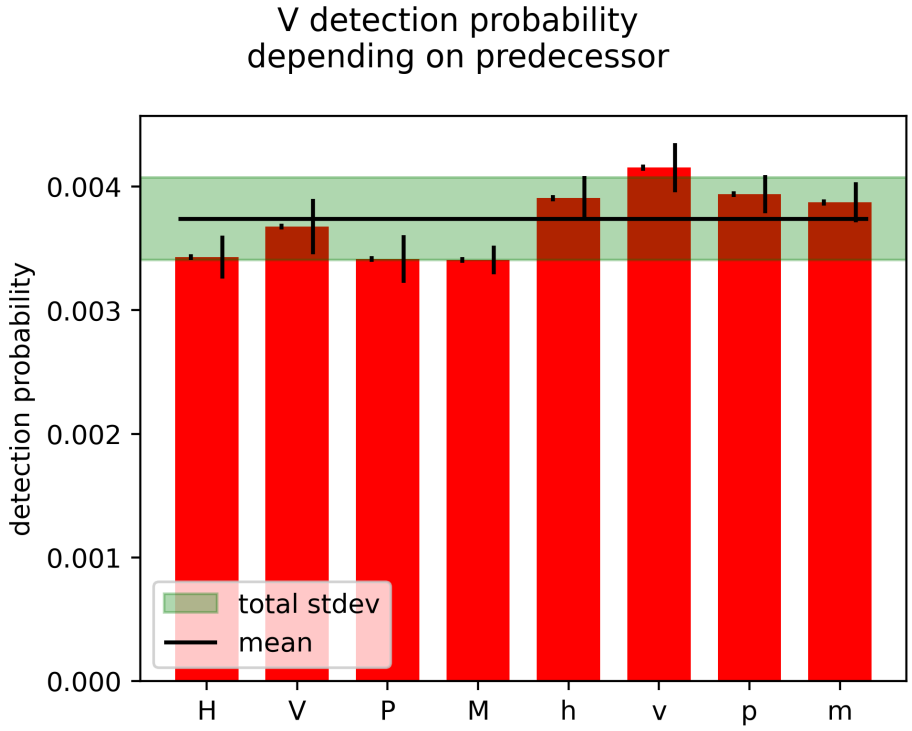

Figure 4.4.: This plot allows to infer whether detection probabilities depend on previously sent polarizations. The bars mark the probability to detect vertically polarized signal states, provided the state named on the x axis was sent beforehand. A capital letter indicates a signal state whereas a lower case corresponds to a decoy state. Every column shows two error bars, the left one corresponds to the expected standard deviation if the events follow a binomial distribution, and the right one shows the actually observed one, calculated by dividing the standard deviation of the counts per bin with given history by the number of times the key was repeated (N_{rep}). The black horizontal line marks the mean detection probability for vertically polarized signal states and the green area represents the standard deviation of all detection events irrespective of the previous state.

The plots, illustrating the detection probabilities of the remaining states considering predecessors are shown in Figure C.10 and C.11 and the ones evaluating the successors in Figure C.12 and C.13. Additionally the probabilities are listed in Table 4.4 and 4.6, using following notation.
The fifth column (h) in Figure 4.4, for example, is the probability to detect a vertically polarized state provided a vertically polarized signal state is sent

and a horizontally polarized decoy state was sent beforehand $P(V|V,h)$. The same notation holds for the analysis of successors, $P(P|p,V)$ corresponds to the detection probability of plus polarized states given a plus polarized decoy state is sent and a vertically polarized signal state is sent afterwards.

$P(d\|c,f)$		f: formerly sent							
d: detected	c: currently sent	H	h	V	v	P	p	M	m
H	H	2.36%	2.15%	2.21%	2.13%	2.44%	2.36%	2.40%	2.38%
	h	1.31%	1.23%	1.28%	1.24%	1.12%	1.12%	1.16%	1.04%
V	V	0.34%	0.37%	0.34%	0.34%	0.39%	0.42%	0.39%	0.39%
	v	0.14%	0.15%	0.13%	0.13%	0.12%	0.13%	0.10%	0.11%
P	P	0.31%	0.31%	0.31%	0.31%	0.32%	0.32%	0.32%	0.32%
	p	0.25%	0.25%	0.26%	0.25%	0.25%	0.25%	0.25%	0.25%
M	M	0.61%	0.61%	0.63%	0.62%	0.64%	0.67%	0.67%	0.67%
	m	0.44%	0.44%	0.45%	0.48%	0.41%	0.41%	0.44%	0.43%

Table 4.4.: The table lists the probabilities $P(d|c,f)$ to detect a specific polarization state d given that a particular state c is sent after the state f has been sent. For example, the element in the first row (H) and fourth column (v) shows that the probability $P(H|H,v)$ to detect a horizontally polarized state if a horizontally polarized signal state has been sent *after* sending a vertically polarized decoy state is 2.13%. Please note that the detection cannot distinguish signal and decoy levels assigning signal and decoy to detected photons is only possible by repeatedly sending the key and inferring the respective intensity level.

As Figure 4.4 shows, the probability to detect a photon indeed depends on the states sent prior and if it happened to be a decoy state the vertically polarized signal photon (V) is more likely to be detected as if the preceding state was a signal state. Such behavior indicates a timing mismatch, i.e., the decoy switch transits a bit too late from decoy to signal and therefore introduces a slight voltage offset which then causes a increase of the detection probability. This voltage offset is influenced by the DC blocking capacitors needed for operating the decoy switches.
Regardless of the chosen intensity, if the predecessor was already vertically polarized, the detection probability is increased. This hints us to a too slow enable signal, turning on the delay line responsible for creating vertically polarized light a bit too late and therefore cutting of some parts of the pulse. In general one can argue if the ratio between decoy and signal state from the same polarization (see Table 4.5 and 4.7) does not change significantly, the decoy switch timing is set right.

probability ratio	predecessor H	V	P	M	h	v	p	m
h/H	55.43%	57.06%	57.81%	58.40%	46.05%	47.43%	48.36%	43.87%
v/V	39.36%	39.93%	39.41%	38.14%	30.40%	31.48%	26.38%	28.64%
p/P	81.82%	82.49%	82.26%	81.27%	76.97%	77.17%	77.92%	77.03%
m/M	72.05%	73.03%	70.87%	77.35%	63.57%	61.43%	65.38%	63.62%

Table 4.5.: Ratio of detection probabilities of decoy and signal level of the different polarizations, listed in the first column, provided the predecessor given in the first row. Probabilities given in Table 4.4.

As Table 4.6 and 4.7 show, the successor of a given pulse does not influence the detection probability as much as its predecessor. This hints us towards shifting the timing of all control signals by tweaking the overall clock delay (cd) of the FPGA (see section B.1.2.3).

$P(d\|c,s)$		s: successively sent							
d: detected	c: currently sent	H	h	V	v	P	p	M	m
H	H	2.14%	2.11%	2.12%	2.12%	2.52%	2.45%	2.53%	2.48%
	h	1.02%	1.00%	1.06%	0.99%	1.34%	1.35%	1.39%	1.31%
V	V	0.35%	0.36%	0.37%	0.36%	0.39%	0.38%	0.39%	0.37%
	v	0.13%	0.12%	0.12%	0.13%	0.13%	0.13%	0.14%	0.13%
P	P	0.32%	0.31%	0.32%	0.32%	0.32%	0.32%	0.32%	0.32%
	p	0.25%	0.25%	0.25%	0.25%	0.25%	0.25%	0.25%	0.25%
M	M	0.64%	0.66%	0.67%	0.65%	0.60%	0.63%	0.63%	0.63%
	m	0.45%	0.45%	0.45%	0.45%	0.43%	0.43%	0.42%	0.42%

Table 4.6.: To verify that detection probabilities are independent of the state which is to be sent next, the conditional probabilities $P(d|c,s)$ for detecting d given c has currently been sent and s will be sent successively are analyzed. For example, the element in the sixth row (P) and last column (m) shows that the probability $P(P|p,m)$ to detect a P state if a P decoy state has been sent *before* sending a M decoy state is 0.25%. As desired, the probabilities do not significantly depend on the state of the successively sent pulse.

probability ratio	successor H	V	P	M	h	v	p	m
h/H	47.50%	47.55%	50.03%	46.55%	53.33%	55.29%	54.96%	52.96%
v/V	35.25%	33.14%	31.43%	35.05%	33.74%	35.70%	34.78%	34.47%
p/P	80.41%	81.11%	79.35%	78.12%	77.88%	79.11%	79.50%	80.17%
m/M	70.65%	68.77%	67.95%	69.07%	71.21%	67.96%	67.90%	66.62%

Table 4.7.: Ratio of detection probabilities of decoy and signal level of the different polarizations, listed in the first column, provided the successor given in the first row. Probabilities given in Table 4.6.

As the FPGA is configured for now, the bit controlling the decoy switch is distributed to every lane equally, i.e., if the state to send is a decoy state, every lane switches to the decoy path. Changing this to only affect the according polarization will help reducing this correlation. However for characterizing the set configuration is preferable, as ideally the preceding and succeeding states should not matter at all and this way one is able to observe more irregularities. While evaluating the error bars pictured in Figure 4.4, it gets obvious that the observed standard deviation is much higher than one would expect from a binomial distribution. This is in contrast to observations made in [65] where the standard deviations matched quite well. One possible reason is the use of a different VCSEL array compared to this work, suggesting that the events created by the photons of the currently used array does not follow a binomial distribution. However the plus polarization, pictured in Figure C.10 indicates a better match between the two standard deviations, therefore hinting non ideal parameters for the other polarizations.

4.4. Polarization analysis

The polarization of the four different states for both signal and decoy levels is illustrated in Figure 4.5 and it shows that the polarization is only changed slightly by selecting a different intensity level through the decoy switch.
Ideally one would expect no change at all, as the polarization is set by the fixed polarizers, which should be not influenced by the chosen intensity. But it is possible that the VCSELs, driven by a altered current, exhibit a change in DOP, which on the other hand can affect the quality of state preparation. Furthermore the possibility of electrical crosstalk due to bad routing is given, which may create additional photons in another polarization state, therefore influencing our tomography.
This measurement is performed with Bob's PAU, which is only able to distinguish two out of three bases, therefore we can only access two components of

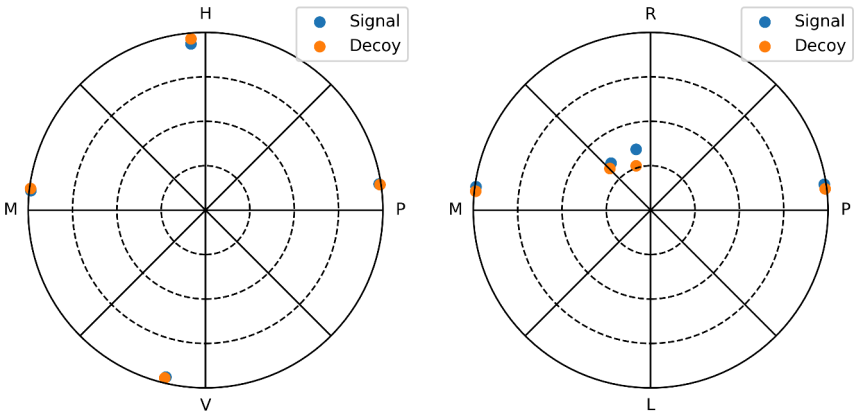

Figure 4.5.: A polarization state tomography through Bob's PAU. The left plot pictures the the state on the Poincaré sphere projected onto the H/V, P/M plane and the right plot shows the states on the R/V, P/M plane perpendicular to the latter. Numerical values are given in Table 4.8.

	$\|H,\mu\rangle$	$\|H,\nu\rangle$	$\|V,\mu\rangle$	$\|V,\nu\rangle$	$\|P,\mu\rangle$	$\|P,\nu\rangle$	$\|M,\mu\rangle$	$\|M,\nu\rangle$
Q	0.936	0.965	-0.938	-0.944	0.148	0.144	0.111	0.122
U	-0.083	-0.083	-0.224	-0.231	0.978	0.982	-0.985	-0.987
V	0.343	0.250	0.265	0.236	0.145	0.121	0.133	0.107

Table 4.8.: Stokes vector components $Q = \frac{I_H - I_V}{I_H + I_V}$ and $U = \frac{I_P - I_M}{I_P + I_M}$, as measured with Bob's PAU and for calculating $V = \sqrt{1 - Q^2 - U^2}$ a DOP of one is assumed. All values exhibit a uncertainty of $< 0.3\,\%$.

the stokes vector (see Equation 2.13). For determining the third component we assume a DOP of one, as previous tests using a full tomography setup, have shown a DOP > 0.99 for the individual polarizations.
Note that, as the measurement was done at the very end of this master's thesis, not much time was invested for calibration. For sure the measurement is subjected to errors due to small misalignments. But as we prioritize the comparison of the polarizations of the two intensity levels to each other, this method suffices to give a good estimation of how the polarization is modified by sending a pulse with decoy intensity instead of a signal level. The figures suggests that hardly any crosstalk or other deviations are present.
For a precise quantum state tomography a well calibrated setup capable of projecting onto all three bases (see section 2.1.3) is required.

5. Conclusion and outlook

This thesis showed the development of a sender electronics suitable for generating decoy states used in a polarization encoded, weak coherent pulse implementation of the BB84 protocol. In contrast to typically employed optical intensity modulation using Pockels cells or variable attenuators, the designed electronics is able to generate distinct optical intensity levels by utilizing attenuated electrical pulses to drive the laser diodes. This design allows for a simple and compact optical hardware.
The previously used USB2.0 interface is upgraded to USB3.0 to allow for real-time key exchange at 100 MHz, by incorporating the Cypress EZ-USB FX3 IC. Besides increasing the data rate, USB3.0 allows our device to be powered by the USB host without the need of an external power supply.
In the course of the redesign the formerly used Spartan-3 FPGA was replaced by a Spartan-6 chip to ensure long-term support and to allow the use of new and improved Xilinx IPs, i.e., prefabricated building blocks for use in the FPGA configuration.
To be able to communicate with the FX3 chip two firmwares were programmed in C++: One is deployed when the memory of the FPGA needs to be flashed with a new configuration file, therefore saving us the otherwise necessary external programming cable. The other firmware is used during normal operation, utilizing a 32-bit, 100 MHz FIFO interface to transmit delay line, laser driver and key data to the FPGA.
The corresponding counterpart at the host side, was extended by the functionality to write a new FPGA configuration file to the flash memory and to support keys including decoy bits. Additionally, the ability to generate a key, using the computer's pseudo-random number generator, and the possibility to play back any key, stored in a key file, was added.
In section 3.3 we discussed methods to synchronize the clocks of Alice and Bob. The presented technique using the time of arrival of the QKD signals shows promising results: The initial frequency and phase difference can be precisely determined up to 1 mHz. To enable real-time key exchange the algorithm needs to be extended by a feedback loop, continuously adjusting the phase and frequency depending on whether the photon arrived earlier or later than expected. Doing so allows to account for frequency drifts due to temperature, vibration, etc. in both clocks. A method to identify the frame offset,

remember, Alice's first sent pulse will most probably not correspond to Bob's first detected pulse, still has to be found.

The performance of the sender was evaluated in chapter 4, while considering real-world conditions to collect representative data. Under such conditions the sender is able to reliable generate distinct intensity levels using all four laser diodes at a repetition rate of 100 MHz. As the four different VCSELs in the used micro optical assembly require individually tuned driving parameters to exhibit equal brightness, the constant attenuation in the decoy path of the four different lanes lead to contrasting results. To still be able to observe notable decoy pulses, without adjusting the individual attenuation, we were forced to increase the brightness of two channels. Note that this measure does not distort the results, as the findings in this work enable us to individually adjust the attenuation for each lane such that the four polarizations would match in signal and decoy intensity.
Moreover the temporal side channel was characterized by calculating the overlap of the temporal shapes of signal and decoy pulses. Even the lowest overlap of 99.29 % indicates a good fit, but nevertheless we still need to analyze the resulting impact on security.
Furthermore the influence of the history of the pulses, i.e., the correlation of detection probability to the previously (or subsequently) sent state, is analyzed. Here we observe a rather strong dependence on the sent pattern, which would be improved by fine tuning the used pulse and laser parameters.
Finally, we checked the effect of selecting different intensities on the polarization, which is, as expected rather low.
To demonstrate an actual key exchange, further work needs to be done on finding suitable parameters for laser driver, delay line and attenuation to minimize any potential side channels. To adjust the attenuation more precisely, conveniently and rapidly one could replace the Π-pad by an digital attenuator IC, again controlled by the FPGA. Future developments need to implement the discussed feedback loop for synchronization, a suitable frame offset detection, the possibility to configure the FPGA directly via the FX3 chip without overwriting the stored configuration on the flash memory and utilize the included burst-mode memory for key storage.

In summary this work presents a small (10.7 x 11.2 x 1.1 cm^3), portable, low power (8 W) electronics, capable of creating very short electrical pulses ($\mathcal{O}$(100 ps)) whilst modulating their intensities at a high repetition rate of 100 MHz, suitable for decoy state QKD.

Appendices

A. Electronics

In this chapter we describe practical electronic design rules and principles needed to create high speed electronics. As we are working with short ($\approx$ 100 ps) pulses featuring rise times in the range of a few pico seconds, we need to pay attention to details that normally wont be notable.
If we think about a digital electric circuit or chip, we often think of voltage levels present on some pins. We pull them high or low with some additional inputs and outputs (I/Os). Those I/Os are connected to the respective pin via a wire or trace. For low speed, this model of just connecting and applying different voltage levels suits well enough. But if we increase the frequency of the signals, the AC characteristics of any building block becomes more and more prominent. Every part introduces some parasitic inductance or capacitance, which can and will change the working principle of any circuit.
Although Howard Johnson calls it "black magic" [61], we do understand the basic principles, which are discussed in the following.

A.1. Transmission lines

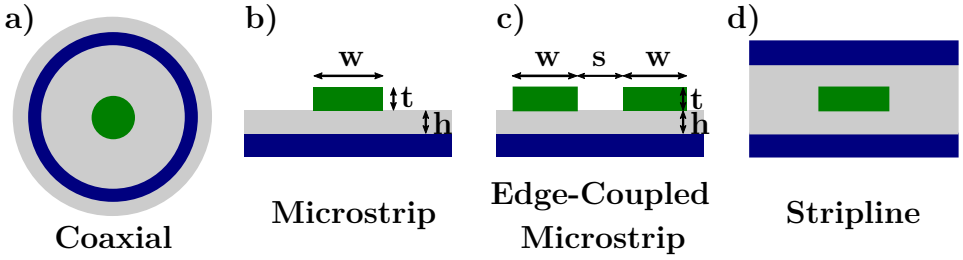

Figure A.1.: Different geometries of transmission lines. The conductor is drawn in green, ground is blue and the dielectric is indicated in gray.

If we want to transmit a high frequency signal without any distortions, we need a proper model for an electric connection. At low frequencies a trace on the *printed circuit board* PCB can be seen as an Ohmic resistor. But a trace

always has a relatively high inductance L [61, p. 139], where the resulting impedance is calculated via

$$X_L = \omega L \tag{A.1}$$

with $\omega = 2\pi f$. This quantity becomes big at high frequencies and inevitably introduces a phase shift between voltage and current. Additionally, different traces might have a so-called cross talk, i.e., they are coupled to each other via induced magnetic fields. A sharp current spike on one line will generate a magnetic field which in turn will induce a current in the other trace.
At such high frequencies we need to model our current as electromagnetic waves which travel through a wave guide. Such wave guides are also called *transmission lines*, as they are also called for antennas. Some important geometries of such lines are illustrated in Figure A.1. For our purposes, *microstrips* and *edge-coupled microstrips* are the most important ones as we are working with plane PCBs.
All transmission lines show a defined distance h between conductor and ground. Using this geometry and the fact that we are dealing with high frequencies and a low-loss line, we get following relations for $\frac{w}{h} \leq 1$:

$$Z = \sqrt{\frac{R + i\omega L}{G + i\omega C}} \approx \sqrt{\frac{L}{C}}, \tag{A.2}$$

$$C = \epsilon_0 \frac{A}{d} = \epsilon_0 \frac{wl}{h} \tag{A.3}$$

$$, L_{\text{micro}} \approx \frac{\mu_0 l}{2\pi^2 \epsilon_{\text{eff}}} \ln\left(8\frac{h}{w} + \frac{w}{4h}\right) \tag{A.4}$$

$$\Rightarrow Z \approx \frac{\mu_0 c}{2\pi\sqrt{\epsilon_{\text{eff}}}} \ln\left(8\frac{h}{w} + \frac{w}{4h}\right). \tag{A.5}$$

Here l is the length of the transmission line, c the speed of light, μ_0, ϵ_0 the magnetic and electric constants and ϵ_{eff} the effective permittivity of our dielectric. As we are using *FR4* (flame retardant 4) for our PCB the dielectric constants of the material are given by $\epsilon_r = \epsilon_{\text{FR4}} = 4.3 \Rightarrow \epsilon_{\text{eff}} \approx 0.64\epsilon_r + 0.36 = 3.11$.
We have to match the impedance of our transmission line to the input and output impedance of our source and sink, respectively. If we have a mismatch, it is possible that our signal integrity is compromised due to reflections. To reach the predefined impedance of $Z_{\text{single}} = 50\,\Omega$ as used by most chips, we chose $w_s = 0.335\,\text{mm}$, $h = 0.2\,\text{mm}$ and $t = 0.035\,\text{mm}$, with which we obtain $Z_s = 50.02\,\Omega$. Additionally we have to make sure that our transmission line is properly terminated.
The impedance of an edge-coupled micro strip does not have a simple approximation and the calculation is rather involved. For calculating the proper

trace width and distance we used the calculator of our PCB manufacturer *Multi Circuit Board*. With the resulting parameters of $w_d = 0.205\,\text{mm}$ and $s_d = 0.15\,\text{mm}$ we get a differential impedance of $Z_d = 100.08\,\Omega$ which matches the standard $Z_{\text{differential}} = 100\,\Omega$.

A.2. Differential signals

The last section explained how to reduce the cross talk of transmission lines and how to avoid unwanted signal reflections. To further improve the signal quality and its robustness against external disturbances, one may resort to *differential signaling*. The basic working principle is illustrated in Figure A.2. Consider a transmission line as in Figure A.1 c). We have two conductors, on one we send our signal $S_s(t)$ the other one carries an inverted copy of the signal $\bar{S}_s(t) = -S_s(t)$. If any disturbance $D(t)$ happens along the line we would receive $S_r(t) = S(t) + D(t)$ and $\bar{S}_r(t) = \bar{S}(t) + D(t)$ at the receiver. If the receiver now subtracts one signal from the other, we obtain

$$\begin{aligned} S(t) &= \frac{1}{2}\Big(S_r(t) - \bar{S}_r(t)\Big) = \frac{1}{2}\Big(S_s(t) + D(t) - \Big(\bar{S}_s(t) + D(t)\Big)\Big) \\ &= \frac{1}{2}\Big(S_s(t) + S_s(t)\Big) = S_s(t), \end{aligned} \tag{A.6}$$

the sent signal $S_s(t)$, where the disturbances cancel out [66]. In addition to that, differential signaling also reduces the electromagnetic disturbances for other signals because a positive current spike in one line is countered by a negative spike in the other line so the magnetic fields cancel each other. Another advantage is that there is close to no return current through the ground plane, so the ground reference becomes less important [66].
All those properties enable us to use a lower voltage for transmitting signals, as we are more resistant against noise, which in turn allows for higher frequencies and reduces power consumption.
One last advantage is the precise timing differential signaling can offer, as a logical high (or low) does not depend on some threshold values or reference voltages. As soon as the signal $S(t)$ is greater than zero we consider a logical high, whereas a signal below zero is considered as a logical low. This also eliminates the need of fully charging and discharging the entire wire, which can take a while depending on the conductor length [66], therefore slowing down the communication.
The downsides of differential signaling are obviously the need of a second conductor and the necessity of matching both wires in length. The reason for the latter point is explained in more detail in the next section.

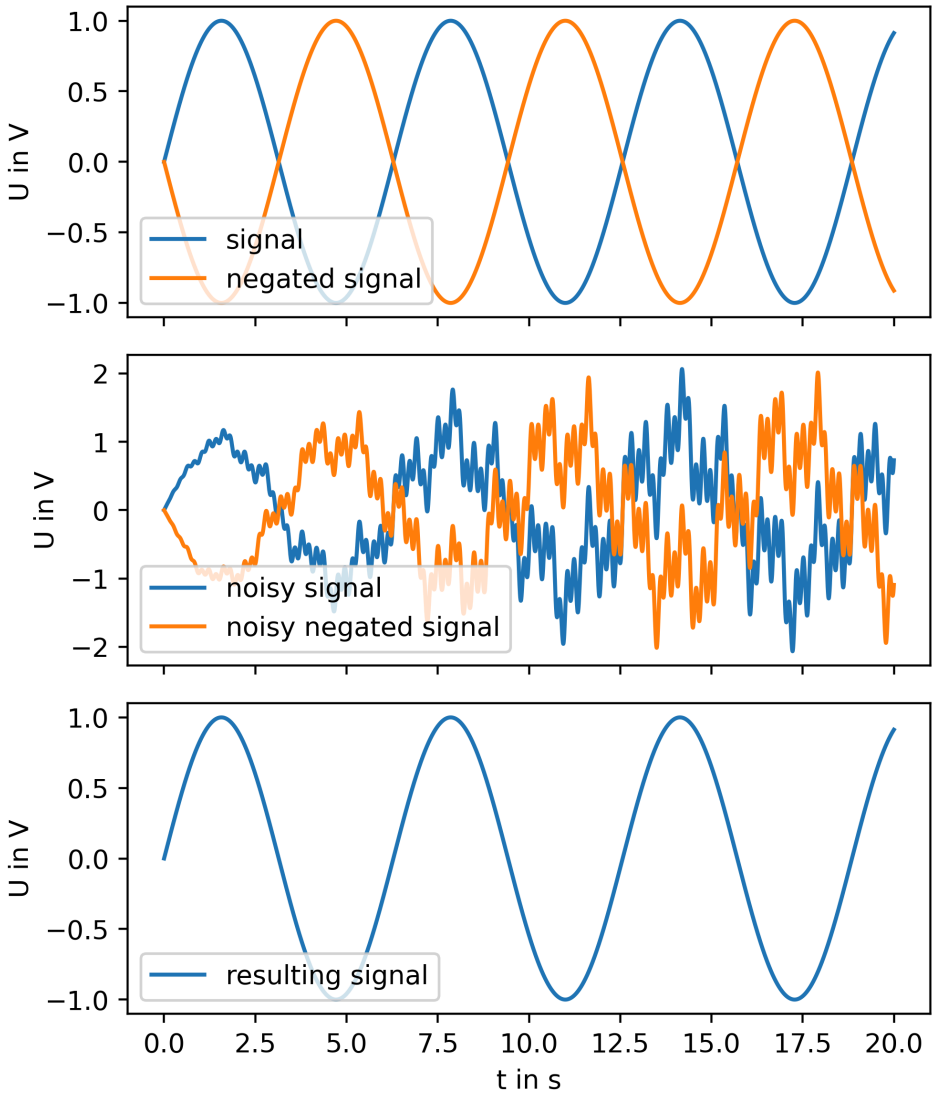

Figure A.2.: Working principle of differential signaling. The upper subplot shows us the signal we want to transmit in blue, a negated copy of it in yellow. The second plot illustrates both signals with a random noise added on top. The lower subplot shows us the resulting signal after both signals are added properly. See Equation A.6.

A.3. Trace speed

Operating at high frequencies has a lot of hidden traps. One of them being sensitive on wire lengths. The speed of a signal in a micro strip is given by [66]:

$$v_{\text{micro}} \approx \frac{c}{\sqrt{\epsilon_{\text{eff}}}}. \tag{A.7}$$

For our used material FR4 we get a value of $v_{\text{FR4}} = 170\,\text{mm}\,\text{ns}^{-1}$. If we want to have a valid timing at the order of pico seconds we need to match our conductors to 170 µm. Keep in mind that we have to deal with pulses shorter than 200 ps, such that a broadening of only 10 ps correspond to a 5 % longer pulse. For matching trace lengths a meander structure is used, where additional trace is added via one or multiple bulges.

B. Technical notes

This chapter provides an in depth discussion of the used electrical components and their communication interfaces.

B.1. Control interfaces

This section will introduce the different interfaces used to control the device. First we will present the FX3 chip and its two different roles, before we discuss the FPGA and its configuration.

B.1.1. FX3

As we want to be able to communicate with our QKD device via USB, we make use of a Cypress EZ-USB chip. USB specifies four different transfer modes. *Control transfers* are intended for configuration and status operations, *Isochronous transfers* for time-dependent information, *Interrupt transfers* are used for infrequent time-dependent data and with *Bulk transfers* large amounts of time-insensitive data gets transmitted.
Previous implementations used the EFM-01 board including an FX2LP chip, featuring only USB2.0 and a 16-bit *first in first out register* (FIFO). If we want to continuously exchange a key with 100 MHz repetition rate, we need a transfer rate of at least $100\,\text{MHz} \cdot 4\text{bit} = 400\text{Mbit/s}$, considering one bit to specify the value, one for basis and the last two for the intensity level (signal, decoy, vacuum). USB2.0 achieves a theoretical transmission speed of 480Mbit/s, but in real world applications a FX2 device is only capable of 350Mbit/s.
Therefore a FX3[1] chip from Cypress is used, utilizing the USB3.0 protocol, which features a theoretical transfer rate of up to 5Gbit/s. In real world benchmarks Cypress states a transfer rate of 3.6Gbit/s which is more than enough, even for faster clock speeds of up to 900 MHz.
Another reason for an USB-C interface is the enhanced power output of USB3.0 or USB-C hosts. While USB2.0 assures only a current of 500 mA at 5 V, USB3.0 delivers a minimum of 900 mA. USB type C hosts must be able to

[1] CYUSB3012-BZXC

source 3 A where USB-PD (power delivery) extends this even more by supplying additional voltages of up to 20 V and 5 A. The values stated above are minimal requirements. Although most manufacturers of USB hosts allow for a higher current, this is not assured.
As our device draws 1.6 A, a USB-C host is required for guaranteed functionality, for using it with older USB-2.0/3.0 hosts an external power supply can be connected. Out of three USB3.0 hosts tested, all were able to power the device with a voltage drop to 4.9 V.
The USB-C plug is symmetric and it is therefore possible to make it reversible, i.e., not dependent on the plug orientation. For the single USB2.0 differential data line, one connects the two positive and the two negative pins to each other, but for the two USB3.0 differential data lines one has to include a 2:1 *multiplexer*[1] (MUX), which selects the correct data lines depending on the orientation of the connector. For detecting the orientation, the FX3 firmware is programmed to try to connect via USB3.0. If this fails, it changes the state of a *general purpose input/output pin* (GPIO) connected to the mux and tries again.
The utilized FX3 chip serves two main roles, where one is used while running an experiment, transmitting laser and pulse parameters as well as raw key data and the other mode is used while programming the FPGA. In general this can be achieved via an external programmer as well but updating the configuration of the FPGA without the use of external cable or installing any additional drivers and software is the preferred way.
Every time the FX3 gets plugged into a USB-host it enumerates itself with a bootloader firmware, which takes care of receiving and booting into one of the two modes mentioned above. It is possible to add a flash memory to load the firmware from but as those firmwares are only around 120Kb the loading times are short and it allows for a more versatile usage of the FX3. The hardware implementation was done according the application note 70707 [67].

B.1.1.1. FIFO

The FX3 features a 32-bit FIFO which can be clocked at 100 MHz [68]. In addition to the 32 data lines, there is a need for eleven control signals, see Figure B.1. One of them is the chip select signal ($\overline{\text{SLCS}}$) which needs to be driven low if any communication with the FIFO is desired. The same is true for the address line (A[1:0]) where the corresponding socket needs to be chosen before any data exchange is started. When running at 100 MHz the FPGA needs to clock PCLK 180° out of phase to its own clock because of tight timing restrictions. The sender must have already prepared the data on the 32-bit

[1] PI3USB302ZBE+DA

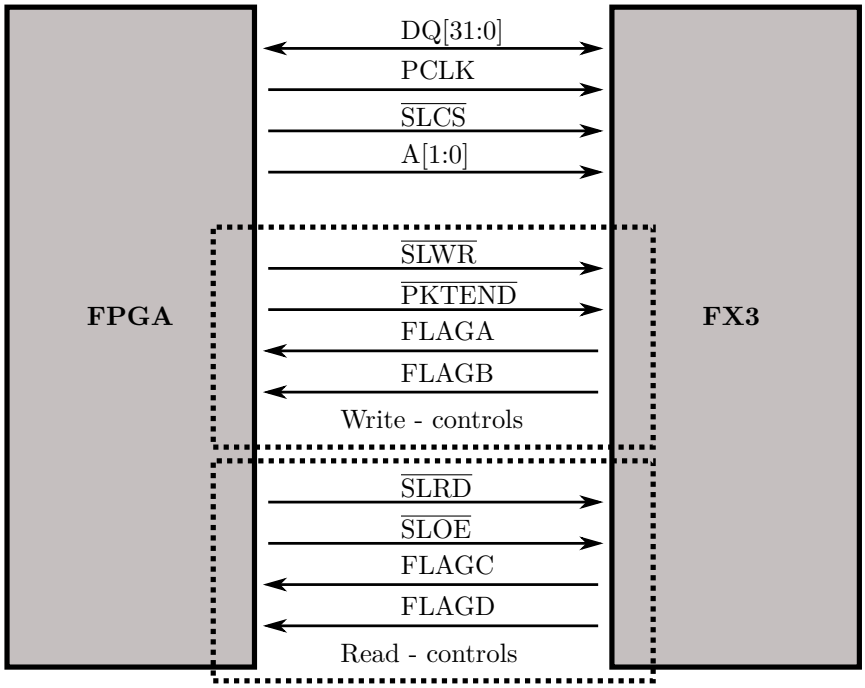

Figure B.1.: A schematic of the FIFO interface with all needed connections between the FX3 and FPGA. $\overline{\text{SLCS}}$ stands for *slave chip select*, the bar on top indicates an active low signal. WR stands for *write*, RD for *read*, OE for *output enable*. The 2 bit address line A[1:0] is used for addressing the correct FIFO if multiple sockets are used.

bus at the time the receiver tries to read it, otherwise, some bits may be not in a stable condition and are therefore transmitted wrongly.
All read or write cycles are realized via a *finite state machine* (FSM) which consists of only a few states where different pins are driven high or low according to the current state. Via transition rules the states change from one to another depending on some flags. As for now the FIFO is solely used for transmitting data from the FX3 to the FPGA, only the read part is described, while the write part is done in a similar manner.
First, data gets transmitted from the USB-host to the FX3 which then drives a flag high, indicating that the FIFO buffer is not empty (FLAGC). After enough data is buffered such that some adjustable watermark (configured to 24 bytes) is reached a flag indicating partial filling (FLAGD) is raised. The FPGA then pulls the lines $\overline{\text{SLRD}}$ and $\overline{\text{SLOE}}$ low, to enable the FX3 to drive the data bus and to signal it is ready to read.
Now the FPGA reads a 32-bit data word in every clock cycle into its memory,

while the FX3 prepares a new data word in between, due to the 180° phase shift. The FPGA stays in this state as long as FLAGD is high.
If this signal gets pulled low, the FX3 indicates that the FIFO buffer is almost empty and the FPGA deasserts $\overline{\text{SLRD}}$ and reads the remaining six data-words, completing a full read cycle with deasserting $\overline{\text{SLOE}}$. In this firmware we are using the Bulk transfer method of the USB specification along with some vendor commands via control transfers to get status information or resetting the FX3 back to bootloader. Note that a minimum of 24 bytes have to be sent such that FLAGD gets asserted and the FPGA begins to read out the FIFO.

B.1.1.2. SPI

The second operating mode of the FX3 is its *serial peripheral interface* (SPI) programming mode where the FX3 obtains configuration data for the FPGA from USB and programs the connected flash memory[1] via SPI. To make sure the memory is not being read by the FPGA while trying to program it, the FX3 drives the PROGRAMB pin of the FPGA low to keep the FPGA in a chip-reset state. After the pin is driven high again, the FPGA tries to reconfigure itself from the connected flash memory.
Here we are using control transfers to transmit the data, split into 4kB blocks consisting of 16 pages of 256 byte each, to the FX3. As a flash write can only switch a bit from 1 to 0, one needs to erase the flash memory beforehand, setting every bit to a 1. Since our used flash has a density of 128Mbit, erasing every 64KB sector at a time would take quite some time. This is why we restrict ourselves to only erase sectors needed for the new configuration file.
The wires are routed for configuring the FPGA directly via the FX3 chip without having the persistent configuration file stored on the flash overwritten, although this function is not implemented yet. According to the manual [69], the FX3 additionally must control the M1 pin and monitor the DONE and INITB pins of the FPGA. We have to reset the FPGA via PROGRAMB and set M1 to logic high, to set the FPGA to SPI slave mode. After driving PROGRAMB high again, we then wait for INITB to be driven low by the FPGA, signaling that the FPGA is ready to be programmed. Receiving this signal we can then write the configuration file via SPI to the FPGA. The DONE pin should be driven high by the FPGA if and only if the configuration was a success.

[1] MT25QL128ABB1ESE0AUT, for use with XILINX indirect programming

B.1.2. FPGA

We decided to use an Xilinx Spartan6 FPGA[1] instead of an Spartan3 as included in the EFM-01 board. This allows the use of newer Xilinx core generator *Intellectual Properties* (IP), which are preconfigured logic functions optimized for Xilinx FPGAs.
The Clocking Wizard IP is used for generating the 180° phase shift used for the PCLK signal of the FIFO, as well as for adjusting the overall phase shift in respect to the input clock. This is necessary to send the control signals to the switches at a suitable time such they are in a defined switch state when the pulse is propagating through them. Another IP, called Block Memory, is utilized for storing the key and can hold up to 64kB of data.
FPGAs are in general programmed in hardware description languages where, as the name suggests, the logic of hardware chips is described. One declares different input and output pins of the virtual chip (entity) and continues with writing the logic leading to the desired behavior. If one is finished creating the different required building blocks (chips), one connects the pins of different entities via virtual cables or signals in the main program.
As there was already a FPGA configuration to begin with and the laserdriver and delayline chips did not change, the entities for controlling those chips namely delay_control and laser_control were only adjusted slighlty. The changes for every other configuration part are listed in the following subsections.

B.1.2.1. USB control

As the FX3 FIFO differs significantly from the FX2LP a whole new communication needed to be written. The FIFO interface is controlled by a FSM, which takes care of all the raw data transfers, described in section B.1.1.1. But for sending delayline or laser parameter to the corresponding chips, we need some sort of protocol and a few registers. The registers are objects consisting of 32 data bits and one update bit. The seven registers used are called

- REG_CONTROL – only bit 0 is used for static decoy control right now.
- REG_DELAY – Bits 31-28 are used for the selection mask where every bit corresponds to one delayline chip. Bits 20-10 are used for setting d_b and 10-0 for d_a.
- REG_LASER – Bits 31-27 are used for the selection mask where every bit corresponds to one laserdriver chip including the beacon. Bits 10-

[1] XILINX_DEVICES_6SLX9TQG144

8 are used for the to set laserdriver register address and 7-0 for the corresponding value.

- REG_RAM_LENGTH – The whole register is used for storing the key length in states[1].
- REG_RAM_DATA – This register is used to store the data which needs to be written to the internal RAM.
- REG_PLAYBACK – Bit 0 is used for the RAM_PLAYBACK, bit 1 for the PLAYBACK_CONTINUOUS The organization of thisand bit 2 for the PLAYBACK_START signal. Those signals control if and how the content of the ram is played, either once or in a continuous loop.
- REG_CLOCK – Bits 31-23 control the target phase shift of the clock where the bits represent a signed integer.

As the FPGA needs to know which data to put in which register, there is a crude protocol utilized via another FSM. Whenever a communication is started, the first 32 data bits received give some information about the following data words. Bits 31-24 represent the register address to write into and bits 15-0 indicate how many data words are following.
This is required because, as stated earlier, we need to send at least six 32-bit data words for the FIFO to function properly, but for most registers one data word is enough. Sending the expected data length enables us to ignore any spare words, or even start a new communication cycle after all expected words are read in. In addition to that, while transmitting huge amounts of data via USB2.0 the FPGA sometimes empties the buffer of the FX3, due to the fast reading of 3.2Gbps. This leads to a deassertion of FLAGD and FLAGC which stops the transmission. After the buffer is filled with key data again, a new communication is started and the FPGA tries to read address and length but there is no such header, as the remaining data of the previous message is still to be transmitted.
This problem can be bypassed by checking if all expected data words are received after a transmission is finished and if not skipping the header check and continue with streaming data into the previous register until the expected length is reached. If a register is written into, the update bit of the corresponding register is set such that the other entities know they need to process the data.

[1] one state $\widehat{=}$ 4bits

B.1.2.2. RAM control

The utilized Block Memory IP can be configured to a dual port RAM, which has dedicated ports for reading and writing. Both ports have their own address and the data width is adjustable and are configured for a 32-bit write width and a 4-bit read width. This allows us to directly write the contents of REG_RAM_DATA into the RAM and increase the write address by one every time the update bit of the register is set high.
The block memory allows for simultaneous reading and writing where the write instructions are executed first. While playing back the key stored in the RAM, the address is incremented at every clock cycle and the four read bits are sent to the playback_control entity which uses the first two bits for determining which laser should be on and the last two for the corresponding intensity.

B.1.2.3. Clock control

As we want to be able to adjust the timing of the signals controlling the decoy-switches and delay line enable pins, we need a method of manipulating the phase shift of the internal FPGA clock. The Clocking Wizard IP provides us with some pins to set a specific phase shift. However, the interface provided needs some additional control circuit as it consists of a clock pin (PSCLK), a phase shift enable pin (PSEN), a phase shift done pin (PSDONE) and the pin for increasing (high) or decreasing (low) the phase shift (PSINCDEC).
The implementation of the Clocking Wizard IP called clock_manager is controlled by the clock_control entity, which implements the protocol described in the user guide of the clocking resources [70]. Again, a FSM is used for managing all the different signals, where as soon as the REG_CLOCK register has its update bit set the FSM jumps into the pre_reset state and the target phase shift is read from the last 9 bits of the corresponding data word stored in REG_CLOCK.
After the FX3 read cycle has finished the FSM shifts to the reset state where the reset pin of the clock_manager is asserted and the variable monitoring the current phase shift is cleared. To trigger the reset circuitry in the clock_manager the reset signal must be asserted for a minimum of five clock cycles, after fulfilling this requirement the FSM jumps into the pre_idle state where it is waiting for the locked pin of the clock_manager becoming high, signaling the reset was successful.
After this condition is true, the FSM is in the idle state where the PSINCDEC is set high if target $>$ current phase shift and low if target $<$ current phase shift, before shifting into the init state. In this state the PSEN pin is driven high, starting a phase shift cycle of the clock_manager, after which the FSM

transits into the shifting state. The FSM stays in this state until the PSDONE pin is pulled high by the clock_manager, signaling a successful phase shift cycle.
The signal keeping track of the current phase shift gets in- or decremented according to PSINCDEC and the FSM shifts into the idle state again, where another cycle starts until the current phase shift equals the targeted value.
Every phase shift step translates to a delay typically around (23 ± 10) ps (see

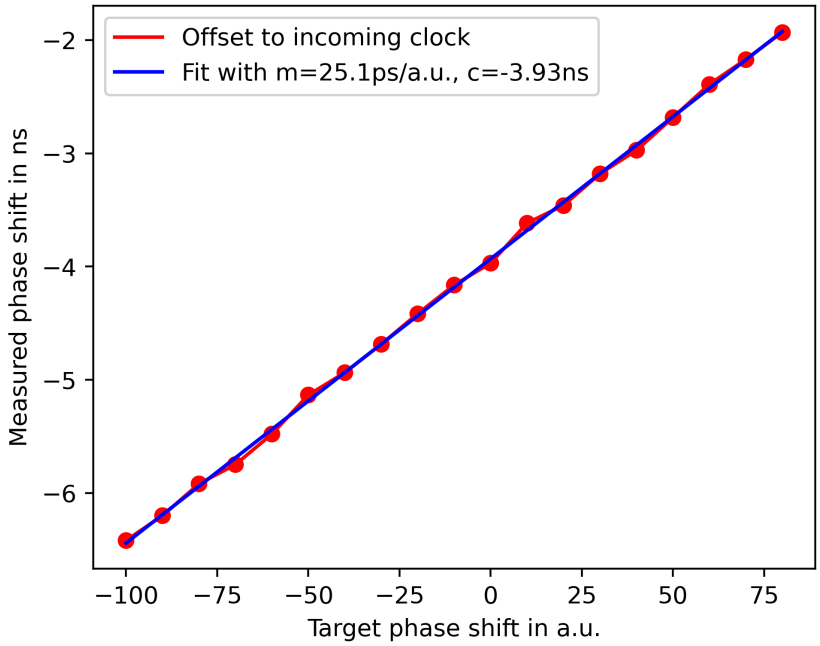

Figure B.2.: The measured phase shift plotted against the target phase shift. The slope of a linear fit 25.1 ps/step matches the value in the data sheet. Even with zero target phase shift a delay of −3.93 ns is introduced by the FPGA, which needs to be accounted for when finding the right timings later on.

Figure B.2) depending on temperature and manufacturing differences. According to the data sheet [70] 105 steps are assured to be available, which result in a phase shift of around ±2.5 ns. But depending on the synthesis and utilization, more steps can be available.
If this dynamical phase shift is in- or decremented to a limit value, the zeroth bit of the STATUS register is set high until another operation shifts away from it. This bit connected to the dedicated clock led of the mainboard to indicate a clocking fault.
In addition to this fine tunable phase shift there is the possibility to output

four different clock signals each shifted by 90° to the last one. The small dynamical phase shift is added on top of this coarse phase shift. It is also possible to shift the pulse generation by 180° by switching the values of d_a and d_b as mentioned in Figure C.4.

B.2. Timestamp unit

To record the clicks of the APD we need a fast and precise timestamp. This unit is designed around a time-to-digital (TDC) converter chip[1] from ScioSense by Markus Rau, a former member of the group. The chip is paired with a FX2LP chip[2], enabling USB functionality and a Spartan-3A[3] which manages the data flow between TDC and FX2.
Altogether the device features eight channels, each with a typical resolution of 81 ps. The possibility to use other modes with different resolution but only two channels is given as well. Due to hardware design each of the eight SMA connectors has a different trace length connecting to the TDC chip. In the worst case there is a difference of $\delta_l = 26.6\,\text{mm}$ and together with the effects discussed in section A.3 this length difference results in a time delay of $\delta_t = \frac{\delta_l}{v_{\text{FR4}}} = 156.6\,\text{ps}$, which need to be compensated for in later measurements.
The timestamp unit features an internal 40 MHz clock[4], as well as an external reference clock input. One can choose between those two clock sources soldering a zero ohm jumper resistor onto the PCB. The usage of the reference clock input is preferred if one needs to be insensitive to frequency drifts of any external clocks.
The timestamps are saved by the dump-read program of the software framework named tdc-apps, where the timestamps are stored in a text file, consisting of two columns. The fist column holds the timestamp in pico-seconds and the second one gives information about the channel at which the event happened. The use of American standard code for information interchange (ASCII) characters results in a human readable data file. Every ASCII character takes up one byte of storage, where smaller timestamps, due to less digits need less characters to be written.
If an experiment is running for more than a second, we need twelve characters for the timestamp, one for the column spacer (in our case a space), one for the channel and another one for the row spacer (new line). This results in fifteen characters and therefore $15 \cdot 8 = 120$ bit of storage per timestamp written. This method yields very large data files, typically around 1.3Gb and 75 million rows for a one minute experiment, depending on the used laser intensities. To parse such kind of files into any program one needs to read the file line by line and

[1] TDC-GPX [2] CY7C68013A-56 [3] XC3S50ATQ144 [4] D75J-040.0M

splitting every line into timestamp and channel data. Because such for loops with file I/Os cannot be parallelized easily, the parsing of the experimental data can take a long time, e.g., the used evaluation program took approximately 140 s to read in 75 million events.
To reduce the file size and simultaneously reducing the parsing time a new data format is introduced. The timestamps are now saved in a binary file, where every timestamp uses 64 bits, where the first 4 bits hold the channel data and the remaining 60 bits store the timestamp itself. Those 60 bits can store timestamps up to $2^{60} \cdot 10^{-12}\,\mathrm{s} = 320.3\,\mathrm{h}$, before an overflow occurs. Using this method we are able to store 75 million events in about 600Mb of storage. As we are reading those events as uint64 (unsigned integer 64 bit), we can make use of highly optimized bit shift operations for splitting up channel and timestamp data, resulting in a read time of 5.2 s for 75 million events.
With this new data format we are therefore able to effectively halve the storage needed, while reducing the read time by 96.4%.
Another approach for reducing file sizes is only keeping track of the differences to the last occurred event. But this would also lead to a increase in the reading time and one needs to consider the numerical error introduced by adding a huge amount of small numbers.
If the reference clock input cannot be used because Alice and Bob do not share any additional channel capable of transmitting the clock signal without distortion, both clocks must be synchronized by other means described in section 3.3.

C. Additional data

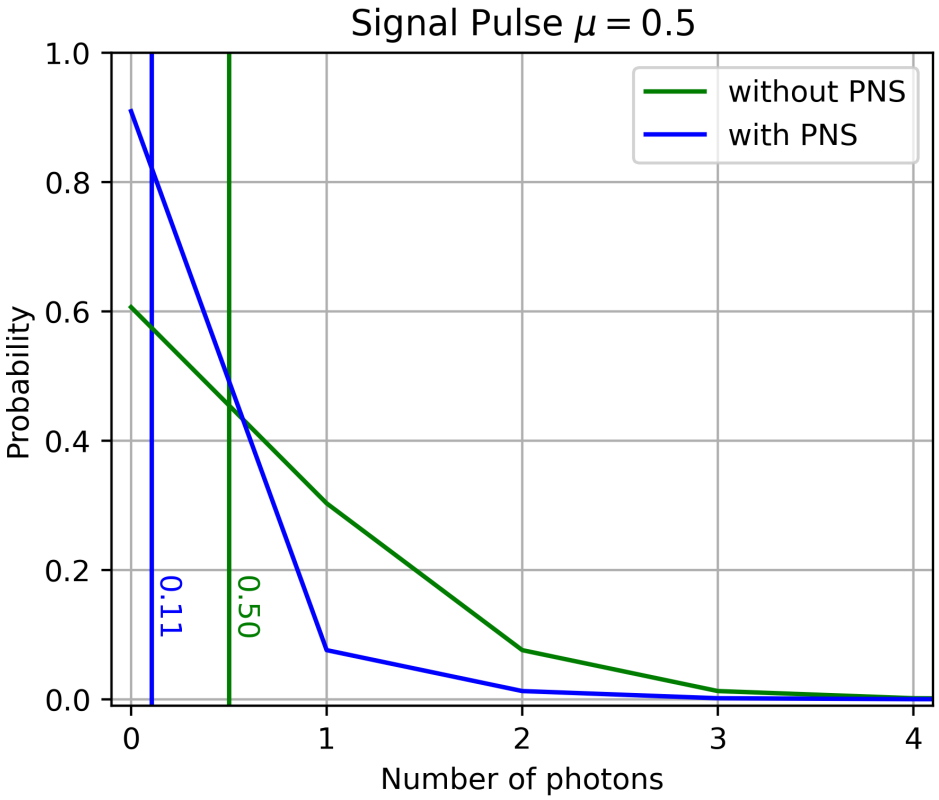

Figure C.1.: Poissonian distribution with $\mu = 0.5$ in green. Probability distribution after a PNS attack in blue. The vertical lines mark the mean photon numbers. With the PNS attack, Eve introduces a channel loss of 78.7%. See Figure C.2 for more explanation.

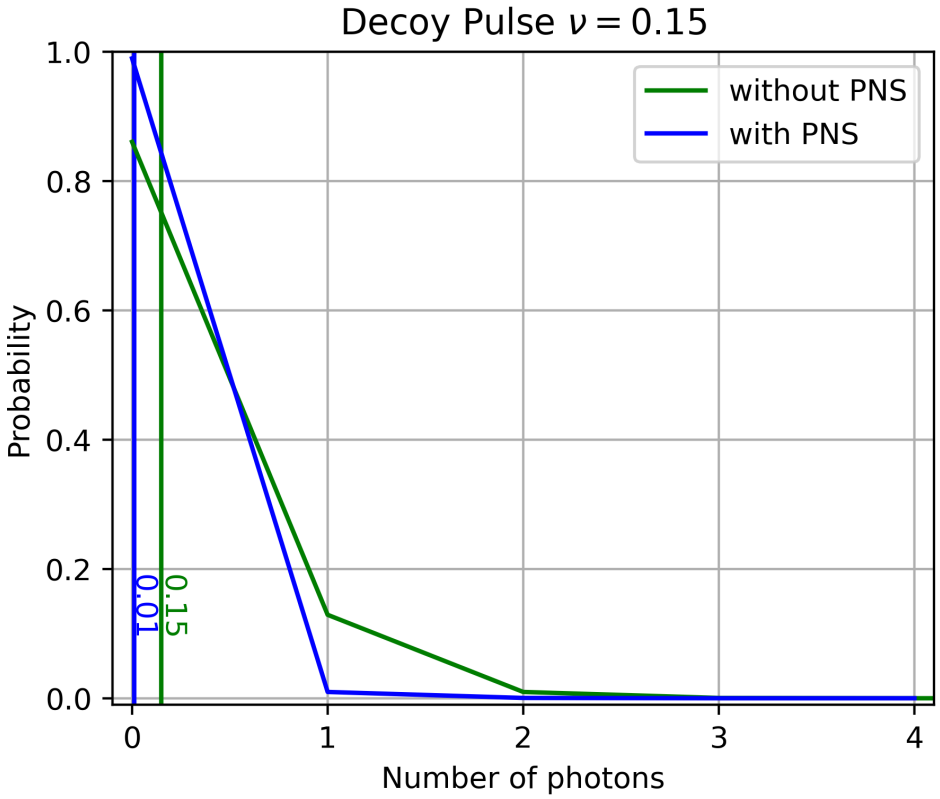

Figure C.2.: Possonian distribution with $\nu = 0.15$ in green. Probability distribution after a PNS attack in blue. The vertical lines mark the mean photon numbers. The resulting channel loss is 92.8%. Figure C.1 shows a channel loss of 78.7% $\neq$ 92.8%. Because normal channel losses do not depend on the underlying statistics, a PNS attack is highly probable and the attacker is revealed.

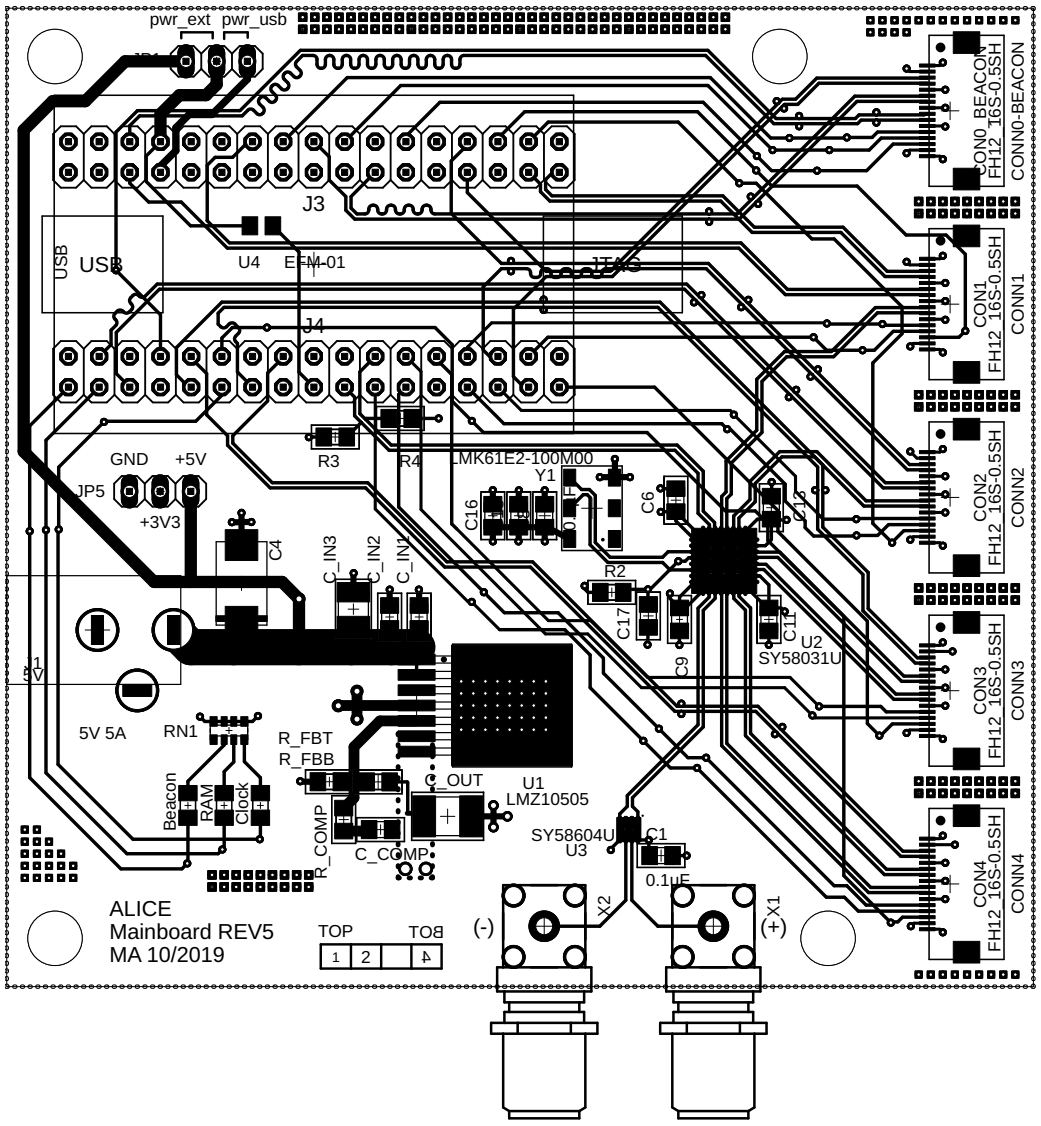

Figure C.3.: Board layout of Alice Testboard, which is only slightly modified to [59] as a new PCB manufacturer was used, supporting different layer buildups. One can see the pin headers (J3, J4) dedicated for the EFM-01 board and the connectors for the driver lanes (CONN0-4). The power supply mainly consisting of a LDO-regulator (U1) is located at the lower left and the clock (Y1) with fanout (U2) is near CONN2 and CONN3. One differential clock signal is routed through a clock buffer (U3) for external use.

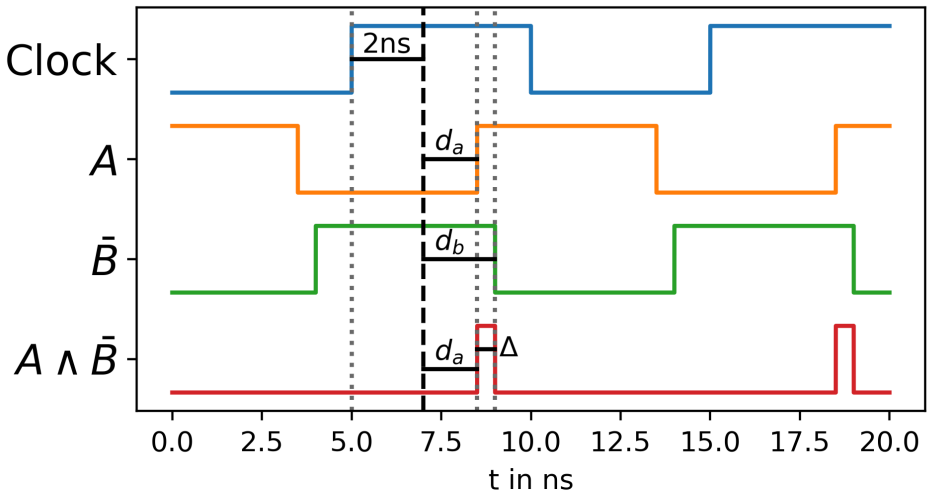

Figure C.4.: Input clock in blue, delay line channel A in orange $d_a = 300 \mathrel{\hat{=}} 2\,\text{ns} + 1.5\,\text{ns} = 3.5\,\text{ns}$, negated channel B in green $d_b = 400 \mathrel{\hat{=}} 2\,\text{ns} + 2\,\text{ns} = 4\,\text{ns}$ and the resulting pulse after the AND gate in red $d_a = 300 \mathrel{\hat{=}} 3.5\,\text{ns}, \Delta = 100 \mathrel{\hat{=}} 0.5\,\text{ns}$. Note by switching d_a and d_b the pulses are shifted by 5 ns thus introducing a 180° phase shift.

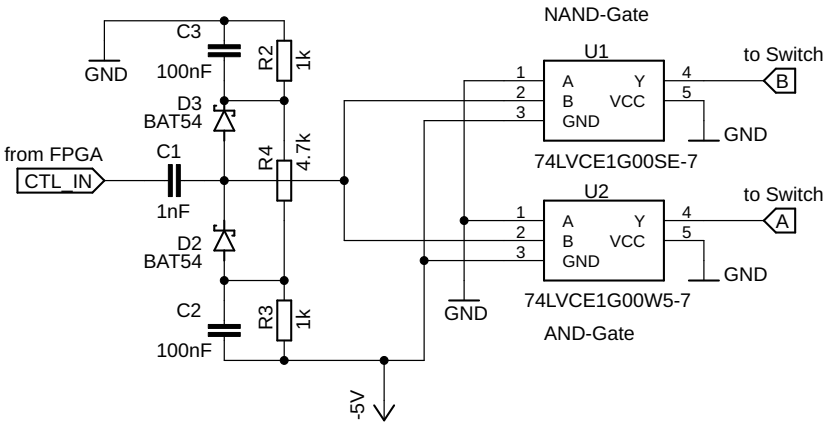

Figure C.5.: Schematic of the switch interface with a capacitor, which is a modification of Figure 3.5, where the Zener diode D1 is replaced by a capacitor C1, Schottky diodes D2, D3 along with resistors R2, R3, R4 and capacitors C2, C3 are added, whereas R1 is removed. All the extra components are used to pull the signal level after the capacitor to the working point.

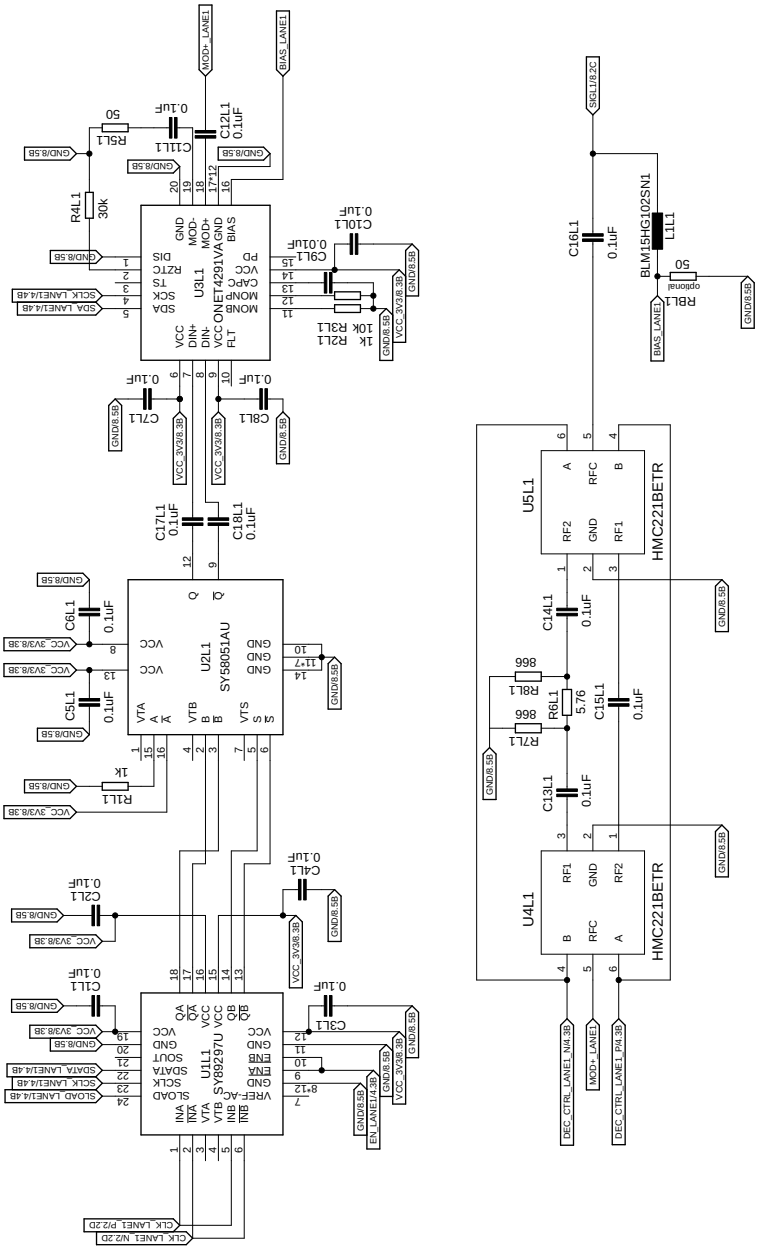

Figure C.6.: The schematic of a driver lane. The clock signal (upper left) gets split and then individually delayed by the delay line chip (U1L1), after which one of the signals is inverted (QA is connected to $\overline{\text{B}}$ and $\overline{\text{QA}}$ to B) and then both signals are combined by the fast and-gate (U2L1). The laser driver (U3L1) converts the resulting short pulse (see Figure C.4) into a signal, suitable for the VCSELs. The modulation output is connected to the decoy switches (U4L1, U5L1), which determine whether the pulse is attenuated by the Π-Pad (R6L1-R8L1) or transmitted unhindered. Finally the modulation signal is mixed with bias output of the laser driver using the Bias-T (L1L1, C16L1).

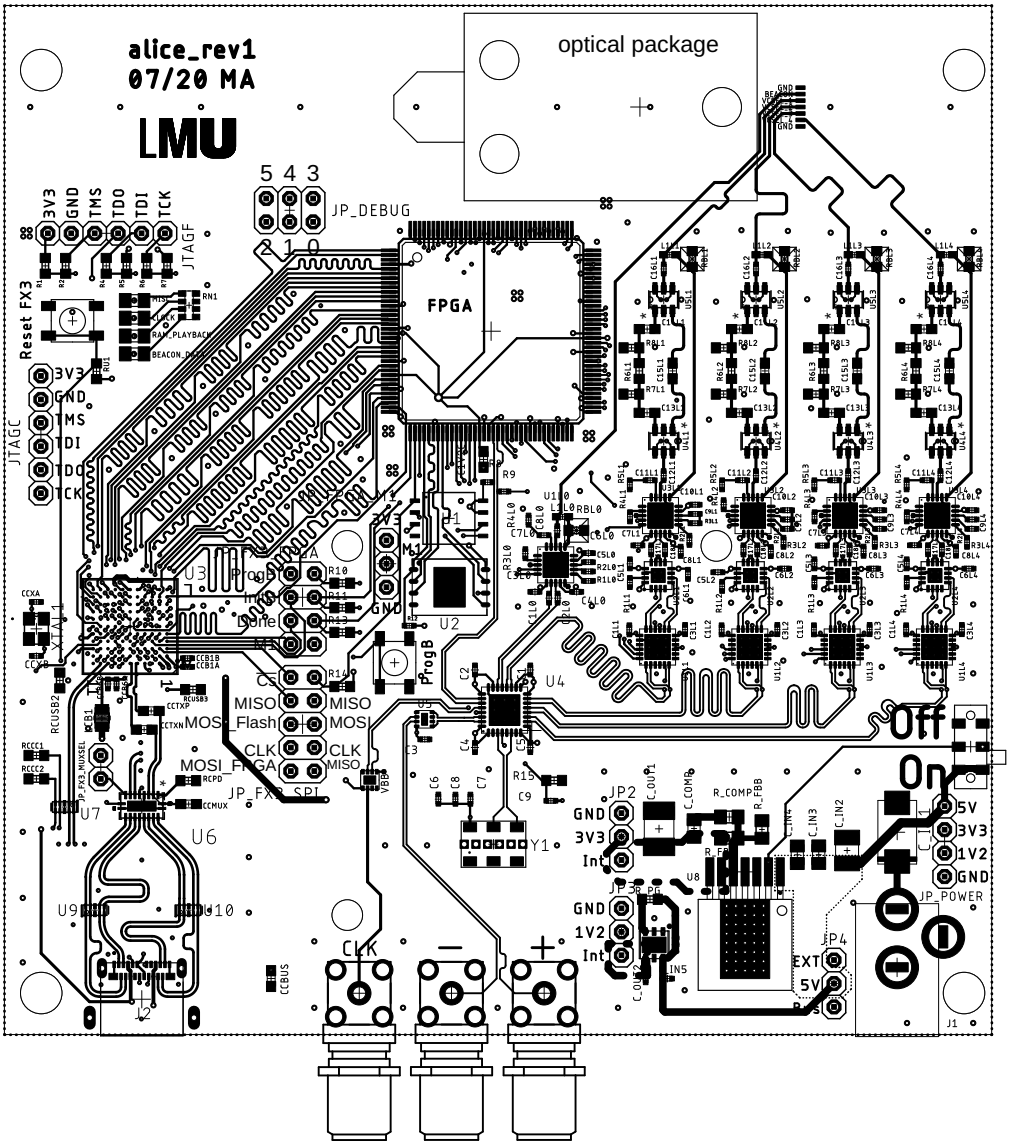

Figure C.7.: Board layout of Alice Mainboard, where at the bottom right the power supply, together with the external power jack (J1) and a switch is located. At the lower left the USB-C receptacle (J2) is mounted, the square above it, after the USB-C MUX (U6), illustrates the FX3 chip (U3) and the 44 length matched FIFO lines exit this chip at its top, connecting to the FPGA almost centered on the board. Slightly below the center of the board the clock buffer (U4) is located, distributing the signal generated by the clock Y1 to the four driver lanes visible on the right side on the board. Those Lanes connect to the optical package on the upper end of the board. Here only the Top-layer of the board, where most components are mounted, is pictured, as the figure would get cluttered with all six utilized layers visible.

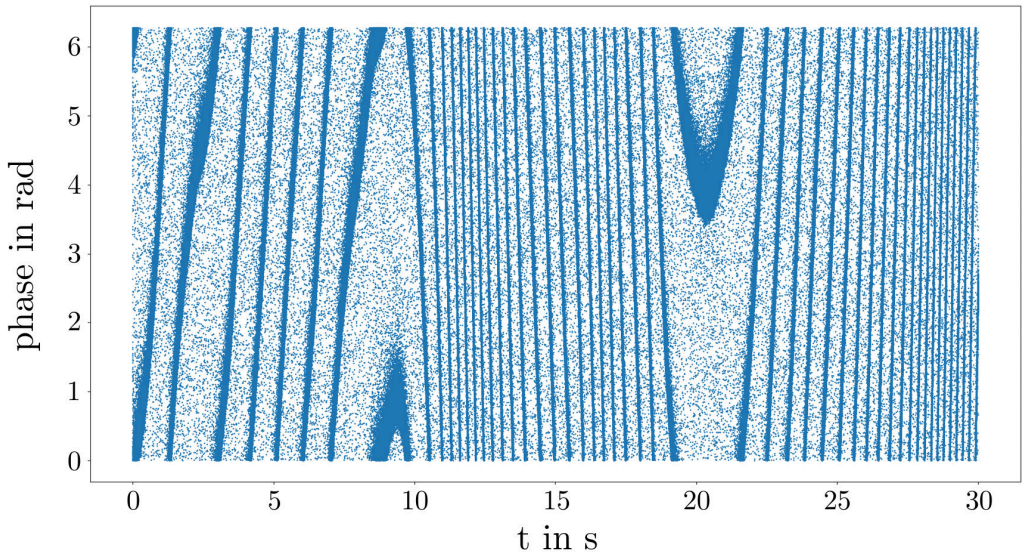

Figure C.8.: The phase $\Phi(t_i)$ of the corresponding event happened at time t_i according to Equation 3.10 with $\Delta f_{\mathrm{FFT}} = -100\,\mathrm{Hz}$ according to Figure 3.15. As most of our events belong to photons sent by Alice (with an instantaneous constant frequency) we can see accumulation along a line. The events between the lines correspond to noise. A constant frequency difference will yield a straight line with a constant slope, no frequency difference would result in a horizontal line. Here, in this picture, we see the phases of a non constant frequency shift, at times close nine and twenty seconds, one can observe a reversal of the frequency change. Figure 3.16 shows the phases between 4.25 s and 4.9 s where the frequency looks almost constant.

state	a	t_0	τ_G	τ_{e_1}	τ_{e_2}	FWHM	$\mathrm{FWHM_{fit}}$	RSME
H	5.71×10^{-13}	$4.93 \times 10^{-9}\,\mathrm{s}$	$2.87 \times 10^{-10}\,\mathrm{s}$	$3.41 \times 10^{-10}\,\mathrm{s}$	$1.74 \times 10^{-10}\,\mathrm{s}$	$5.0 \times 10^{-10}\,\mathrm{s}$	$6.5 \times 10^{-10}\,\mathrm{s}$	1.05×10^{-4}
h	3.02×10^{-13}	$4.92 \times 10^{-9}\,\mathrm{s}$	$2.63 \times 10^{-10}\,\mathrm{s}$	$3.61 \times 10^{-10}\,\mathrm{s}$	$1.77 \times 10^{-10}\,\mathrm{s}$	$6.0 \times 10^{-10}\,\mathrm{s}$	$6.2 \times 10^{-10}\,\mathrm{s}$	5.82×10^{-5}
V	9.07×10^{-14}	$4.57 \times 10^{-9}\,\mathrm{s}$	$1.98 \times 10^{-10}\,\mathrm{s}$	$2.54 \times 10^{-10}\,\mathrm{s}$	$1.24 \times 10^{-10}\,\mathrm{s}$	$4.0 \times 10^{-10}\,\mathrm{s}$	$4.6 \times 10^{-10}\,\mathrm{s}$	2.65×10^{-5}
v	3.09×10^{-14}	$4.65 \times 10^{-9}\,\mathrm{s}$	$2.07 \times 10^{-10}\,\mathrm{s}$	$2.56 \times 10^{-10}\,\mathrm{s}$	$1.31 \times 10^{-10}\,\mathrm{s}$	$4.0 \times 10^{-10}\,\mathrm{s}$	$4.8 \times 10^{-10}\,\mathrm{s}$	8.39×10^{-6}
P	7.94×10^{-14}	$4.55 \times 10^{-9}\,\mathrm{s}$	$2.20 \times 10^{-10}\,\mathrm{s}$	$3.10 \times 10^{-10}\,\mathrm{s}$	$1.43 \times 10^{-10}\,\mathrm{s}$	$5.0 \times 10^{-10}\,\mathrm{s}$	$5.2 \times 10^{-10}\,\mathrm{s}$	1.93×10^{-5}
p	6.22×10^{-14}	$4.58 \times 10^{-9}\,\mathrm{s}$	$2.27 \times 10^{-10}\,\mathrm{s}$	$3.06 \times 10^{-10}\,\mathrm{s}$	$1.48 \times 10^{-10}\,\mathrm{s}$	$5.0 \times 10^{-10}\,\mathrm{s}$	$5.2 \times 10^{-10}\,\mathrm{s}$	1.61×10^{-5}
M	1.65×10^{-13}	$4.41 \times 10^{-9}\,\mathrm{s}$	$2.77 \times 10^{-10}\,\mathrm{s}$	$2.56 \times 10^{-10}\,\mathrm{s}$	$1.79 \times 10^{-10}\,\mathrm{s}$	$6.0 \times 10^{-10}\,\mathrm{s}$	$5.8 \times 10^{-10}\,\mathrm{s}$	3.47×10^{-5}
m	1.11×10^{-13}	$4.50 \times 10^{-9}\,\mathrm{s}$	$2.15 \times 10^{-10}\,\mathrm{s}$	$2.70 \times 10^{-10}\,\mathrm{s}$	$8.29 \times 10^{-11}\,\mathrm{s}$	$4.0 \times 10^{-10}\,\mathrm{s}$	$5.8 \times 10^{-10}\,\mathrm{s}$	2.75×10^{-5}

Table C.1.: Calculated parameters used for fitting Equation 4.2 at Figure 4.3. The FWHM from the raw data and the fit are listed, as well as the RMSE $= \sqrt{\frac{\sum_t (p_{\mathrm{fit}}\,(t) - p_{\mathrm{data}}\,(t))^2}{N}}$ for estimating the fit quality.

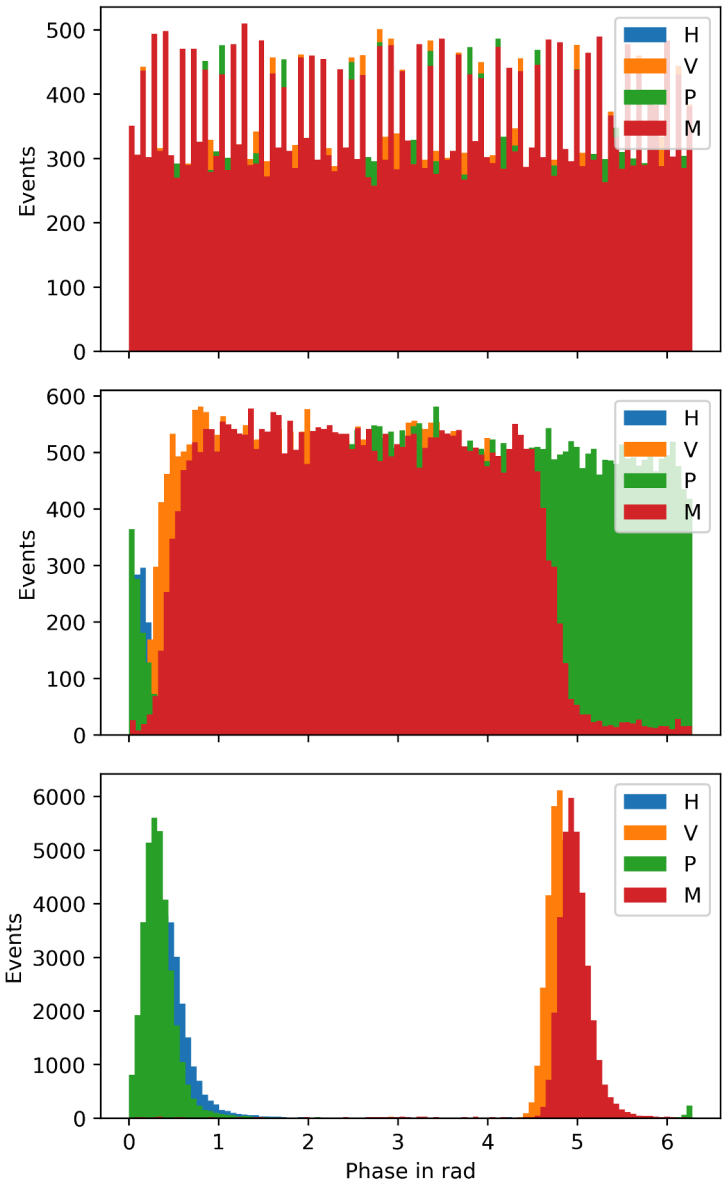

Figure C.9.: Pulse shapes of all different polarizations. The first plot assumes a frequency of 100 MHz, the second is corrected by −100 Hz via the FFT method and the last plot shows the pulses after the correction using the linear fit, again, correcting the frequency by −1.0205 Hz and for a phase of 0.1. The synchronization algorithm was performed on the data of plus polarized light, therefore this pulse is located at the beginning, as its phase was compensated for.

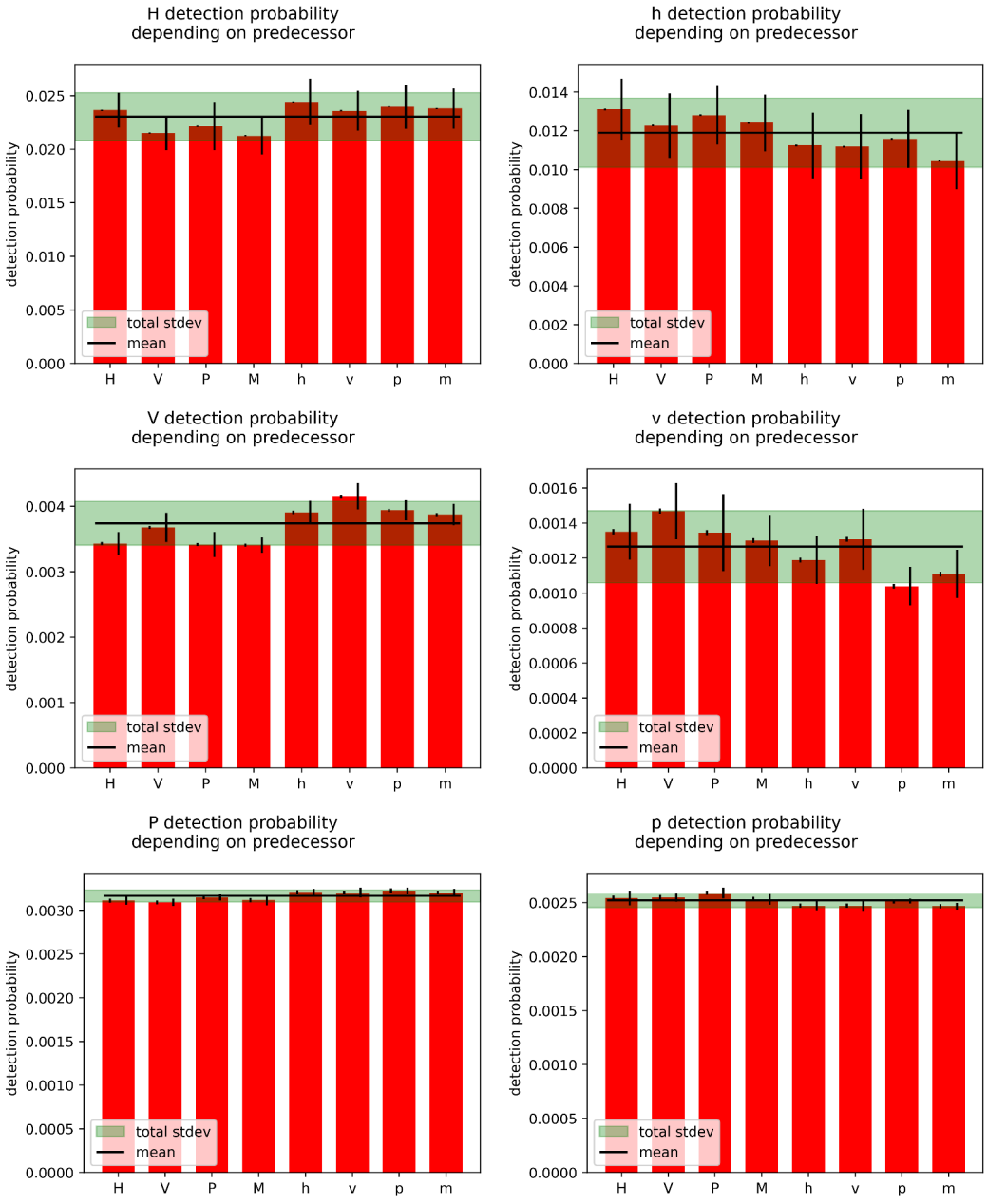

Figure C.10.: The bars mark the mean probability to detect the state mentioned in the title of the corresponding plot, provided the state marked on the x axis was sent beforehand. A capital letter indicates a signal state whereas a lower case corresponds to a decoy state. Every column shows two error bars, the left one corresponds to the expected standard deviation, if the events follow a Poissonian distribution and the right one shows the actual observed one. The black horizontal line marks the mean detection probability for the corresponding state and the green area represents the standard deviation of events, disregarding the history.

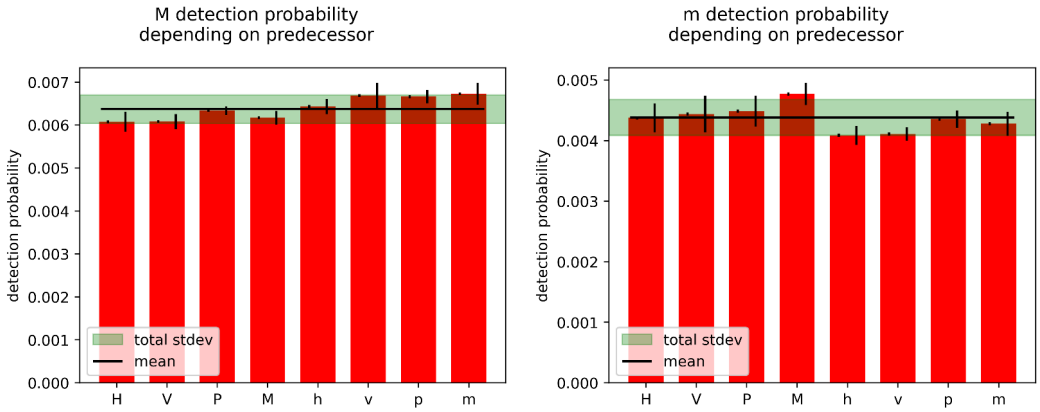

Figure C.11.: See Figure C.10 for explanation.

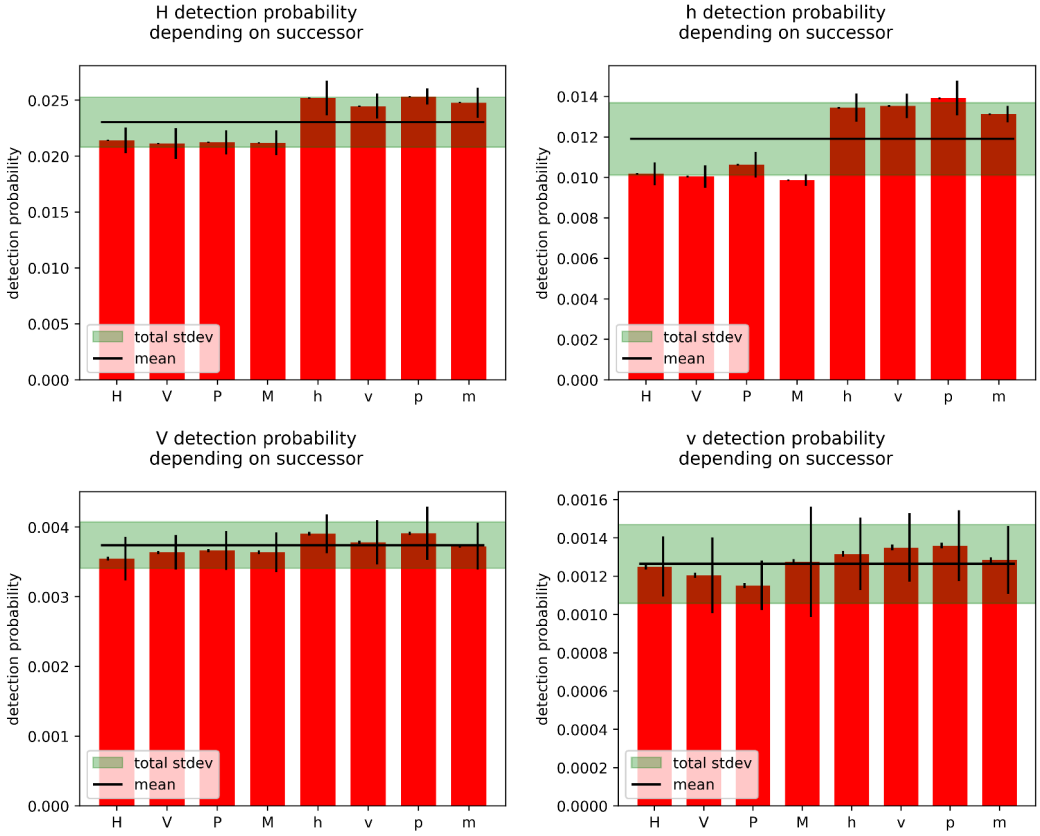

Figure C.12.: The bars mark the mean probability to detect the state mentioned in the title of the corresponding plot, provided the state marked on the x axis is sent afterwards. For further explanation see Figure C.10.

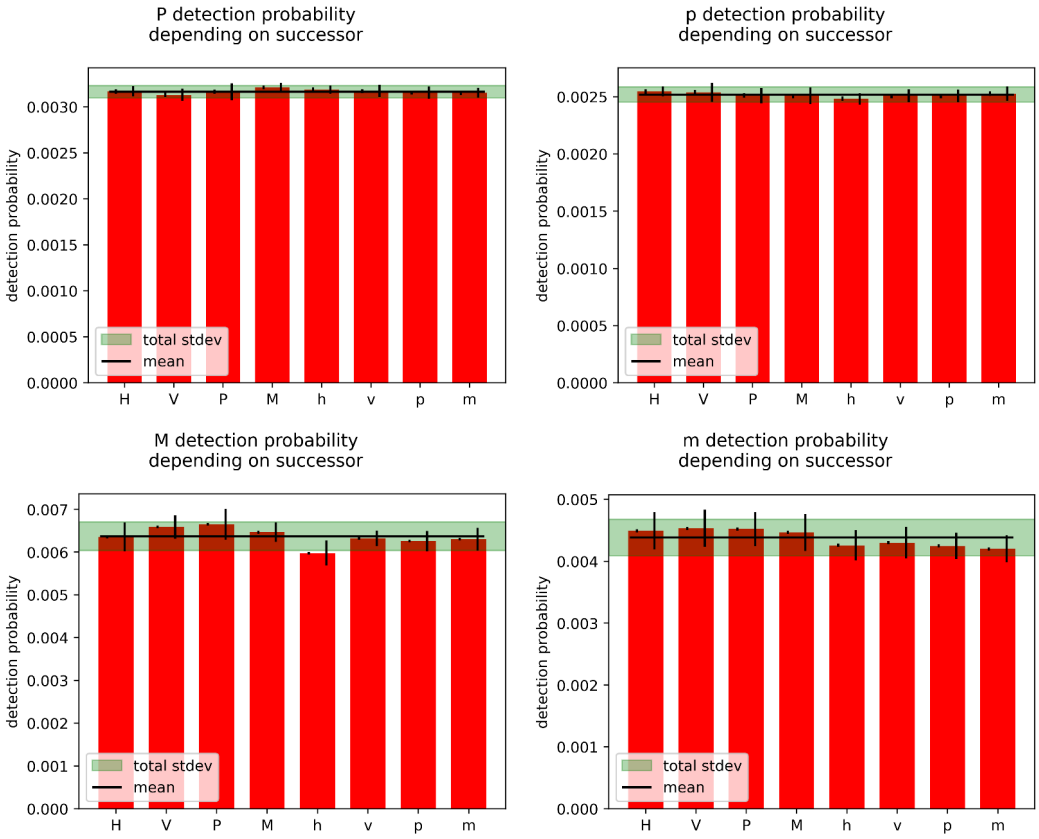

Figure C.13.: See Figure C.12 for explanation.

Bibliography

[1] R. L. Rivest. Description of the LCS35 Time Capsule Crypto-Puzzle. April 1994. URL https://people.csail.mit.edu/rivest/lcs35-puzzle-description.txt.

[2] R. L. Rivest, A. Shamir und L. Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, **21**(2):120–126, February 1978. ISSN 0001-0782. doi: 10.1145/359340.359342. URL https://doi.org/10.1145/359340.359342.

[3] G. S. Vernam. Secret signaling system, 1919. URL https://www.cryptomuseum.com/crypto/files/us1310719.pdf.

[4] X. Ma, B. Qi, Y. Zhao und H.-K. Lo. Practical Decoy State for Quantum Key Distribution. *Physical Review A*, **72**(1):012326, July 2005. ISSN 1050-2947, 1094-1622. doi: 10.1103/PhysRevA.72.012326. URL http://arxiv.org/abs/quant-ph/0503005. arXiv: quant-ph/0503005.

[5] D. Gottesman, H.-K. Lo, N. Lütkenhaus und J. Preskill. Security of quantum key distribution with imperfect devices. *arXiv:quant-ph/0212066*, September 2004. URL http://arxiv.org/abs/quant-ph/0212066. arXiv: quant-ph/0212066.

[6] F. Xu, X. Ma, Q. Zhang, H.-K. Lo und J.-W. Pan. Secure quantum key distribution with realistic devices. *Reviews of Modern Physics*, **92**(2), may 2020. doi: 10.1103/revmodphys.92.025002. URL https://journals.aps.org/rmp/abstract/10.1103/RevModPhys.92.025002.

[7] W. Diffie und M. Hellman. New directions in cryptography. *IEEE Transactions on Information Theory*, **22**(6):644–654, November 1976. ISSN 1557-9654. doi: 10.1109/TIT.1976.1055638. URL https://ieeexplore.ieee.org/document/1055638.

[8] C. H. Bennett und G. Brassard. Quantum cryptography: Public key distribution and coin tossing. *Theoretical Computer Science*, **560**:7–11, dec 2014. ISSN 03043975. doi: 10.1016/j.tcs.2014.05.025. URL http://arxiv.org/abs/2003.06557. arXiv: 2003.06557.

[9] C. H. Bennett, F. Bessette, G. Brassard, L. Salvail und J. Smolin. Experimental quantum cryptography. *Journal of Cryptology*, **5**(1):3–28, January 1992. ISSN 1432-1378. doi: 10.1007/BF00191318. URL `https://doi.org/10.1007/BF00191318`.

[10] A. Boaron, G. Boso, D. Rusca, C. Vulliez, C. Autebert, M. Caloz, M. Perrenoud, G. Gras, F. Bussières, M.-J. Li, D. Nolan, A. Martin und H. Zbinden. Secure Quantum Key Distribution over 421 km of Optical Fiber. *Physical Review Letters*, **121**(19), nov 2018. doi: 10.1103/physrevlett.121.190502. URL `https://journals.aps.org/prl/abstract/10.1103/PhysRevLett.121.190502`.

[11] S.-K. Liao, W.-Q. Cai, J. Handsteiner, B. Liu, J. Yin, L. Zhang, D. Rauch, M. Fink, J.-G. Ren, W.-Y. Liu, Y. Li, Q. Shen, Y. Cao, F.-Z. Li, J.-F. Wang, Y.-M. Huang, L. Deng, T. Xi, L. Ma, T. Hu, L. Li, N.-L. Liu, F. Koidl, P. Wang, Y.-A. Chen, X.-B. Wang, M. Steindorfer, G. Kirchner, C.-Y. Lu, R. Shu, R. Ursin, T. Scheidl, C.-Z. Peng, J.-Y. Wang, A. Zeilinger und J.-W. Pan. Satellite-Relayed Intercontinental Quantum Network. *Physical Review Letters*, **120**(3):030501, January 2018. doi: 10.1103/PhysRevLett.120.030501. URL `https://link.aps.org/doi/10.1103/PhysRevLett.120.030501`.

[12] L. Mazzarella, C. Lowe, D. Lowndes, S. K. Joshi, S. Greenland, D. McNeil, C. Mercury, M. Macdonald, J. Rarity und D. K. L. Oi. QUARC: Quantum Research Cubesat—A Constellation for Quantum Communication. *Cryptography*, **4**(1):7, March 2020. doi: 10.3390/cryptography4010007. URL `https://www.mdpi.com/2410-387X/4/1/7`.

[13] J. A. Grieve, R. Bedington, Z. Tang, R. C. M. R. B. Chandrasekara und A. Ling. SpooQySats: CubeSats to demonstrate quantum key distribution technologies. *Acta Astronautica*, **151**:103–106, October 2018. ISSN 0094-5765. doi: 10.1016/j.actaastro.2018.06.005. URL `http://www.sciencedirect.com/science/article/pii/S0094576518304405`.

[14] *Clavis3 QKD Platform*. ID Quantiqe, 2020. URL `https://marketing.idquantique.com/acton/attachment/11868/f-0216/1/-/-/-/-/Clavis3QKDPlatform_Brochure.pdf`.

[15] D. K. Oi, A. Ling, G. Vallone, P. Villoresi, S. Greenland, E. Kerr, M. Macdonald, H. Weinfurter, H. Kuiper, E. Charbon und R. Ursin. CubeSat quantum communications mission. *EPJ Quantum Technology*, **4**(1):6, December 2017. ISSN 2196-0763. doi: 10.1140/epjqt/

s40507-017-0060-1. URL https://epjqt.epj.org/articles/epjqt/abs/2017/01/40507_2017_Article_60/40507_2017_Article_60.html.

[16] G. Vest, M. Rau, L. Fuchs, G. Corrielli, H. Weier, S. Nauerth, A. Crespi, R. Osellame und H. Weinfurter. Design and Evaluation of a Handheld Quantum Key Distribution Sender module. *IEEE Journal of Selected Topics in Quantum Electronics*, **21**(3):131–137, may 2015. doi: 10.1109/jstqe.2014.2364131. URL https://xqp.physik.uni-muenchen.de/publications/files/articles_2014/ieee-jstqe_21_6600607.pdf.

[17] J. J. Sakurai. *Modern quantum mechanics*. Addison-Wesley Pub. Co, Reading, Mass, 1994. ISBN 9780201539295.

[18] W. Gerlach und O. Stern. Der experimentelle Nachweis der Richtungsquantelung im Magnetfeld. *Zeitschrift für Physik*, **9**(1):349–352, dec 1922. doi: 10.1007/bf01326983. URL https://link.springer.com/article/10.1007/BF01326983.

[19] G. Benenti. *Principles of quantum computation and information*. World Scientific Pub, Singapore River Edge, NJ, 2004. ISBN 9812388303.

[20] P. A. M. Dirac. *The Principles of Quantum Mechanics*. Oxford University Press, 1981. ISBN 0198520115. URL https://www.ebook.de/de/product/3237757/p_a_m_dirac_the_principles_of_quantum_mechanics.html.

[21] B. Schumacher. Quantum coding. *Physical Review A*, **51**(4):2738–2747, apr 1995. doi: 10.1103/physreva.51.2738. URL https://journals.aps.org/pra/abstract/10.1103/PhysRevA.51.2738.

[22] D. P. DiVincenzo. The Physical Implementation of Quantum Computation. *Fortschritte der Physik*, **48**(9-11):771–783, 2000. ISSN 1521-3978. doi: 10.1002/1521-3978(200009)48:9/11<771::AID-PROP771>3.0.CO;2-E. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/1521-3978%28200009%2948%3A9/11%3C771%3A%3AAID-PROP771%3E3.0.CO%3B2-E.

[23] W. Zinth. *Optik Lichtstrahlen - Wellen - Photonen*. Oldenbourg, Muenchen, 2009. ISBN 9783486588019.

[24] P. Freiwang. Towards Hand-held Quantum Key Distribution. Masters thesis, 2017. URL https://xqp.physik.uni-muenchen.de/publications/theses_master/master_freiwang.html.

[25] C. Gerry und P. Knight. *Introductory Quantum Optics*. Cambridge University Press, oct 2004. doi: 10.1017/cbo9780511791239. URL https://www.cambridge.org/core/books/introductory-quantum-optics/B9866F1F40C45936A81D03AF7617CF44.

[26] W. Diffie und M. Hellman. Special Feature Exhaustive Cryptanalysis of the NBS Data Encryption Standard. *Computer*, **10**(6):74–84, jun 1977. doi: 10.1109/c-m.1977.217750. URL https://ieeexplore.ieee.org/document/1646525.

[27] Specification for the Advanced Encryption Standard (AES). Federal Information Processing Standards Publication 197, 2001. URL http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf.

[28] B. Schneier, J. Kelsey, D. Whiting, D. Wagner, C. Hall und N. Ferguson. *The twofish encryption algorithm : a 128-bit block cipher*. J. Wiley, New York, 1999. ISBN 0471353817.

[29] P. Patil, P. Narayankar, Narayan D.G. und Meena S.M. A Comprehensive Evaluation of Cryptographic Algorithms: DES, 3DES, AES, RSA and Blowfish. *Procedia Computer Science*, **78**:617–624, January 2016. ISSN 1877-0509. doi: 10.1016/j.procs.2016.02.108. URL http://www.sciencedirect.com/science/article/pii/S1877050916001101.

[30] N. Ferguson, B. Schneier und T. Kohno. *Cryptography Engineering*. John Wiley & Sons Inc, 2010. ISBN 0470474246. URL https://www.ebook.de/de/product/9395494/niels_ferguson_bruce_schneier_tadayoshi_kohno_cryptography_engineering.html.

[31] T. Dierks und C. Allen. RFC2246. Forschungsbericht, Network Working Group, 1999. URL https://tools.ietf.org/html/rfc2246.

[32] E. Machie. *Network security traceback attack and react in the United States Department of Defense network*. Trafford Publishing, Bloomington, Indiana, 2013. ISBN 1466985747.

[33] S. Cavallar, B. Dodson, A. K. Lenstra, W. Lioen, P. L. Montgomery, B. Murphy, H. te Riele, K. Aardal, J. Gilchrist, G. Guillerm, P. Leyland, J. Marchand, F. Morain, A. Muffett, C. a. C. Putnam und P. Zimmermann. Factorization of a 512-Bit RSA Modulus. In B. Preneel, editor, *Advances in Cryptology — EUROCRYPT 2000*, Lecture Notes in Computer Science, pages 1–18, Berlin, Heidelberg, 2000. Springer. ISBN 9783540455394. doi: 10.1007/3-540-45539-6_1. URL https://link.springer.com/chapter/10.1007/3-540-45539-6_1.

[34] T. Kleinjung, K. Aoki, J. Franke, A. K. Lenstra, E. Thomé, J. W. Bos, P. Gaudry, A. Kruppa, P. L. Montgomery, D. A. Osvik, H. te Riele, A. Timofeev und P. Zimmermann. Factorization of a 768-Bit RSA Modulus. In *Advances in Cryptology – CRYPTO 2010*, pages 333–350. Springer Berlin Heidelberg, 2010. doi: 10.1007/978-3-642-14623-7_18. URL https://link.springer.com/chapter/10.1007/978-3-642-14623-7_18.

[35] P. Shor. Algorithms for quantum computation: discrete logarithms and factoring. In *Proceedings 35th Annual Symposium on Foundations of Computer Science*. IEEE Comput. Soc. Press, 1994. doi: 10.1109/sfcs.1994.365700. URL https://ieeexplore.ieee.org/document/365700.

[36] T. Häner, M. Roetteler und K. M. Svore. Factoring using 2n+2 qubits with Toffoli based modular multiplication. *arXiv:1611.07995 [quant-ph]*, June 2017. URL http://arxiv.org/abs/1611.07995. arXiv: 1611.07995.

[37] E. B. Barker und Q. H. Dang. Recommendation for Key Management Part 3: Application-Specific Key Management Guidance. Forschungsbericht, NIST, jan 2015. URL https://csrc.nist.gov/publications/detail/sp/800-57-part-3/rev-1/final.

[38] F. Arute, K. Arya, R. Babbush, D. Bacon, J. C. Bardin, R. Barends, R. Biswas, S. Boixo, F. G. S. L. Brandao, D. A. Buell, B. Burkett, Y. Chen, Z. Chen, B. Chiaro, R. Collins, W. Courtney, A. Dunsworth, E. Farhi, B. Foxen, A. Fowler, C. Gidney, M. Giustina, R. Graff, K. Guerin, S. Habegger, M. P. Harrigan, M. J. Hartmann, A. Ho, M. Hoffmann, T. Huang, T. S. Humble, S. V. Isakov, E. Jeffrey, Z. Jiang, D. Kafri, K. Kechedzhi, J. Kelly, P. V. Klimov, S. Knysh, A. Korotkov, F. Kostritsa, D. Landhuis, M. Lindmark, E. Lucero, D. Lyakh, S. Mandrà, J. R. McClean, M. McEwen, A. Megrant, X. Mi, K. Michielsen, M. Mohseni, J. Mutus, O. Naaman, M. Neeley, C. Neill, M. Y. Niu, E. Ostby, A. Petukhov, J. C. Platt, C. Quintana, E. G. Rieffel, P. Roushan, N. C. Rubin, D. Sank, K. J. Satzinger, V. Smelyanskiy, K. J. Sung, M. D. Trevithick, A. Vainsencher, B. Villalonga, T. White, Z. J. Yao, P. Yeh, A. Zalcman, H. Neven und J. M. Martinis. Quantum supremacy using a programmable superconducting processor. *Nature*, **574**(7779):505–510, oct 2019. doi: 10.1038/s41586-019-1666-5. URL https://www.nature.com/articles/s41586-019-1666-5.

[39] G. Brassard und L. Salvail. Secret-Key Reconciliation by Public Dis-

cussion. pages 410–423. Springer-Verlag, 1994. URL https://link.springer.com/chapter/10.1007/3-540-48285-7_35.

[40] R. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, **8**(1):21–28, January 1962. ISSN 2168-2712. doi: 10.1109/TIT.1962.1057683. URL https://ieeexplore.ieee.org/document/1057683.

[41] C. H. Bennett, G. Brassard und J.-M. Robert. Privacy Amplification by Public Discussion. *SIAM Journal on Computing*, **17**(2):210–229, April 1988. ISSN 0097-5397. doi: 10.1137/0217014. URL https://epubs.siam.org/doi/10.1137/0217014.

[42] M. Hayashi und T. Tsurumaru. More Efficient Privacy Amplification With Less Random Seeds via Dual Universal Hash Function. *IEEE Transactions on Information Theory*, **62**(4):2213–2232, apr 2016. doi: 10.1109/tit.2016.2526018. URL https://arxiv.org/abs/1311.5322.

[43] V. Scarani, A. Acin, G. Ribordy und N. Gisin. Quantum cryptography protocols robust against photon number splitting attacks for weak laser pulses implementations. *Physical Review Letters*, **92**(5):057901, February 2004. ISSN 0031-9007, 1079-7114. doi: 10.1103/PhysRevLett.92.057901. URL http://arxiv.org/abs/quant-ph/0211131. arXiv: quant-ph/0211131.

[44] A. K. Ekert. Quantum cryptography based on Bell's theorem. *Physical Review Letters*, **67**(6):661–663, August 1991. doi: 10.1103/PhysRevLett.67.661. URL https://link.aps.org/doi/10.1103/PhysRevLett.67.661.

[45] F. Grosshans und P. Grangier. Continuous Variable Quantum Cryptography Using Coherent States. *Physical Review Letters*, **88**(5):057902, January 2002. doi: 10.1103/PhysRevLett.88.057902. URL https://link.aps.org/doi/10.1103/PhysRevLett.88.057902.

[46] C. Weedbrook, A. M. Lance, W. P. Bowen, T. Symul, T. C. Ralph und P. K. Lam. Quantum Cryptography Without Switching. *Physical Review Letters*, **93**(17):170504, October 2004. doi: 10.1103/PhysRevLett.93.170504. URL https://link.aps.org/doi/10.1103/PhysRevLett.93.170504.

[47] T. C. Ralph. Continuous variable quantum cryptography. *Physical Review A*, **61**(1), dec 1999. doi: 10.1103/physreva.61.010303. URL https://journals.aps.org/pra/abstract/10.1103/PhysRevA.61.010303.

[48] N. J. Cerf, M. Lévy und G. V. Assche. Quantum distribution of Gaussian keys using squeezed states. *Physical Review A*, **63**(5):052311, April 2001. doi: 10.1103/PhysRevA.63.052311. URL `https://link.aps.org/doi/10.1103/PhysRevA.63.052311`.

[49] F. Grosshans, G. V. Assche, J. Wenger, R. Brouri, N. J. Cerf und P. Grangier. Quantum key distribution using gaussian-modulated coherent states. *Nature*, **421**(6920):238–241, jan 2003. doi: 10.1038/nature01289. URL `https://arxiv.org/abs/quant-ph/0312016`.

[50] P. W. Shor und J. Preskill. Simple Proof of Security of the BB84 Quantum Key Distribution Protocol. *Physical Review Letters*, **85**(2):441–444, July 2000. doi: 10.1103/PhysRevLett.85.441. URL `https://link.aps.org/doi/10.1103/PhysRevLett.85.441`.

[51] R. Renner, N. Gisin und B. Kraus. Information-theoretic security proof for quantum-key-distribution protocols. *Physical Review A*, **72**(1), jul 2005. doi: 10.1103/physreva.72.012332. URL `https://journals.aps.org/pra/abstract/10.1103/PhysRevA.72.012332`.

[52] W.-Y. Hwang. Quantum Key Distribution with High Loss: Toward Global Secure Communication. *Physical Review Letters*, **91**(5):057901, August 2003. ISSN 0031-9007, 1079-7114. doi: 10.1103/PhysRevLett.91.057901. URL `http://arxiv.org/abs/quant-ph/0211153`. arXiv: quant-ph/0211153.

[53] H.-K. Lo, X. Ma und K. Chen. Decoy State Quantum Key Distribution. *Physical Review Letters*, **94**(23):230504, June 2005. doi: 10.1103/PhysRevLett.94.230504. URL `https://link.aps.org/doi/10.1103/PhysRevLett.94.230504`.

[54] M. N. Wegman und J. Carter. New hash functions and their use in authentication and set equality. *Journal of Computer and System Sciences*, **22**(3):265–279, jun 1981. doi: 10.1016/0022-0000(81)90033-7. URL `https://www.sciencedirect.com/science/article/pii/0022000081900337`.

[55] G. M. Nikolopoulos und E. Diamanti. Continuous-variable quantum authentication of physical unclonable keys. *Scientific Reports*, **7**(1), apr 2017. doi: 10.1038/srep46047. URL `https://pubmed.ncbi.nlm.nih.gov/28393853/`.

[56] E. O. Kiktenko, A. O. Malyshev, M. A. Gavreev, A. A. Bozhedarov, N. O. Pozhar, M. N. Anufriev und A. K. Fedorov. Lightweight authentication

for quantum key distribution. *arXiv:1903.10237 [quant-ph]*, March 2019. URL `http://arxiv.org/abs/1903.10237`. arXiv: 1903.10237.

[57] N. Gisin, S. Fasel, B. Kraus, H. Zbinden und G. Ribordy. Trojan-horse attacks on quantum-key-distribution systems. *Physical Review A*, **73**(2), feb 2006. doi: 10.1103/physreva.73.022320. URL `https://journals.aps.org/pra/abstract/10.1103/PhysRevA.73.022320`.

[58] G. Mélen. *Integrated Quantum Key DistributionSender Unit for Hand-Held Platforms*. phdthesis, LMU, January 2016. URL `https://xqp.physik.uni-muenchen.de/publications/files/theses_phd/phd_melen.pdf`.

[59] C. Sonnleitner. Towards a practical integrated QKD sender. Masters thesis, 2018. URL `https://xqp.physik.uni-muenchen.de/publications/files/theses_master/master_sonnleitner.pdf`.

[60] ON Semiconductor. *Zener Theory and Design Considerations*, hbd854/d edition, 2017. URL `https://www.onsemi.com/pub/Collateral/HBD854-D.PDF`.

[61] H. W. Johnson, M. Graham und G. Martin. *High Speed Digital Design*. Pearson Education (US), 1993. ISBN 0133957241. URL `https://www.ebook.de/de/product/3239643/howard_w_johnson_martin_graham_graham_martin_high_speed_digital_design.html`.

[62] G. Matthaei, L. Young und E. M. T. Jones. *Microwave Filters, Impedance-Matching Networks, and Coupling Structures*. ARTECH HOUSE INC, 1908. ISBN 0890060991. URL `https://www.ebook.de/de/product/3734423/g_matthaei_l_young_e_m_t_jones_microwave_filters_impedance_matching_networks_and_coupling_structures.html`.

[63] H. Weier. *Experimental Quantum Cryptography*. PhD thesis, TUM, 2003. URL `https://xqp.physik.uni-muenchen.de/publications/files/theses_diplom/diplom_weier.pdf`.

[64] D. Bell. Security analysis of the timing side-channel of a freespace QKD system. Bachelors Thesis, 2018.

[65] T. M. P. Schattauer. Generating Decoy Statesfor Quantum Key Distribution. Bachelors Thesis, June 2019.

[66] U. Tietze. *Electronic circuits : handbook for design and application*. Springer-Verlag Berlin Heidelberg, New York, 2008. ISBN 9783540004295.

[67] H. Osman. *EZ-USBFX3/FX3S Hardware Design Guidelines and Schematic Checklist*. Cypress, AN70707, October 2019. URL `https://www.cypress.com/file/139936`.

[68] R. S. K. Vakkantula. *Designing with the EZ-USB FX3 Slave FIFO Interface*. Cypress, AN65974, March 2019. URL `https://www.cypress.com/file/136056`.

[69] R. S. K. Vakkantula. *Configuring a Xilinx FPGA over USB using Cypress EZ-USB FX3*. Cypress, AN84868, April 2018. URL `https://www.cypress.com/file/179146`.

[70] Xilinx. *Spartan-6 FPGA Clocking Resources*, 2015. URL `https://www.xilinx.com/support/documentation/user_guides/ug382.pdf`. UG382.

Acknowledgements

Finally, I want to thank everyone who contributed to my study and this work.

- Prof. Dr. Harald Weinfurter for the opportunity to work on this very interesting project, in a comfortable and independent working environment.
- Peter Freiwang and Dr. Lukas Knips for guiding me through this thesis and their patient but enthusiastic responses to any questions or ideas – "Das bekommen wir schon hin!".
- All members of the XQP group for creating a fun time, especially at lunch in the Mensa.
- My friends for creating a way to relax and enjoy my leisure.
- My girlfriend Eva for her love, the support on my way through life and the encouragement on bad days.
- My parents Heidi and Hubert as well as my brothers Sebastian und Matthias for the ongoing financial and emotional support, without you I would have never been in this fortunate situation.

Erklärung

Mit der Abgabe dieser Masterarbeit versichere ich, dass ich die Arbeit selbstständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

München, 17. November 2020

Michael Auer