PALACKÝ UNIVERSITY OLOMOUC
FACULTY OF SCIENCE

Department of Optics

Quantum key distribution using polarization encoded weak coherent states

BACHELOR THESIS

Jiří Fadrný

2018

PALACKÝ UNIVERSITY OLOMOUC
FACULTY OF SCIENCE

Department of Optics

Quantum key distribution using polarization encoded weak coherent states

BACHELOR THESIS

Author:	Jiří Fadrný
Programe of study:	B1701 Physics
Field of study:	Optics and optoelectronics, 3rd year
Study mode:	full-time
Supervisor:	Mgr. Martina Nováková, Ph.D.
Cosupervisor:	RNDr. Miroslav Ježek, Ph.D.

Thesis done on: .....

UNIVERZITA PALACKÉHO V OLOMOUCI
PŘÍRODOVĚDECKÁ FAKULTA

Katedra optiky

Kvantová distribuce klíče s polarizačně kódovaným slabým signálem

BAKALÁŘSKÁ PRÁCE

Vypracoval:	Jiří Fadrný
Studijní program:	B1701 Fyzika
Studijní obor:	Optika a optoelektronika, 3. ročník
Forma studia:	prezenční
Vedoucí bakalářské práce:	Mgr. Martina Nováková, Ph.D.
Konzultant bakalářské práce:	RNDr. Miroslav Ježek, Ph.D.

Abstract

The goal of the Thesis is to develop a core of quantum key distribution system. Quantum key distribution allows generation and transmission of a secret key, where any eavesdropping attempt is revealed before the key is used to transmit an actual message. I have designed and developed an experimental implementation of two-state B92 quantum key distribution protocol, which can readily be extended to more advanced multi-state protocols even including decoy states. The implementation is based on polarization encoded weak coherent states generated by polarization filtered luminescent diodes controlled by a few hundred long nanosecond electronic pulses produced by a microcontroller. I have also developed software for controlling the experiment, extraction of raw key, and evaluation of error rate. I have demonstrated quantum key distribution over short free-space channel with clock frequency of 2.3 MHz, key rate of 170 kHz, and error probability of 0.8%. The further work will be focused on improvement of speed and security of quantum key distribution implementation and its application to realistic quantum channels.

Key words

Quantum key distribution, protocol, key transmission, cryptography, B92.

Abstrakt

Cílem této práce je vyvinout základ systému pro kvantovou distribuci klíče. Kvantová distribuce klíče nám umožňuje generovat a přenášet tajný klíč, přičemž je schopna odhalit jakýkoli pokus o jeho odposlech dříve, než je použit pro šifrování zpráv. Navrhnul jsem a postavil experimentální realizaci dvoustavového B92 protokolu pro kvantový přenos klíče, kterou lze snadno rozšířit pro potřeby pokročilejších vícestavových protokolů včetně těch používající takzvané decoy stavy. Moje realizace je založena na polarizačním kódování zeslabených koherentních pulzů produkovaných polarizačně filtrovanou luminiscenční diodou, která je řízena několika stovkami nanosekund dlouhými elektronickými pulzy produkovanými mikrokontrolerem. Dále jsem vyvinul program pro řízení experimentu, extrakci klíče a vyhodnocování chybovosti přenosu. Demonstroval jsem kvantový přenos klíče ve volném prostoru na krátkou vzdálenost s opakovací frekvencí hodin 2,3 MHz, rychlostí generace klíče 170 kHz při pravděpodobnosti chyby 0,8%. Další práce bude zaměřena na zlepšení rychlosti a bezpečnosti přenosu klíče a jeho aplikaci v realistických kvantových kanálech.

Klíčová slova

Kvantová distribuce klíče, protokol, přenos klíče, kryptografie, B92.

Acknowledgement

I would like to express deep gratitude to Mgr. Martina Nováková, Ph.D. and RNDr. Miroslav Ježek, Ph.D. for knowledge they passed on me, guidance through my experimental work and a lot of patience. I am grateful for the assistance in my software development provided by Mgr. Ivo Straka and Mgr. Robert Stárek. Moreover, I wish to acknowledge the particular help granted by my friends Jan Grygar, Radek Přívara and Voltěch Kala which was also valuable. Last but not least, I appreciate all the support and love given by my family and friends.

Declaration

I declare that this thesis is my own personal effort and I cite all the resources used. The work was done under the guidance of Mgr. Martina Nováková, Ph.D. and RNDr. Miroslav Ježek, Ph.D.

In Olomouc on .....

Contents

1 Introduction **1**

2 The B92 protocol implementation **7**

3 Experimental setup **9**

4 True random number generator **13**

5 Data analysis **17**

6 Discussion **21**

7 Conclusion **25**

References **26**

1 Introduction

Cryptography studies provide a secure communication between two (or more) parties. Traditionally, they are called Alice (the transmitter) and Bob (the receiver). The third party that might want to eavesdrop the communication is called Eve. A message, plain text respectively, is encrypted with an algorithm called a cipher and a secret sequence called a key [1]. The encrypted message is called ciphertext. In the case of symmetric encryption, we need the same secret key to read the ciphertext that was used to encrypt the message as shown in Figure 1 below. There is also an asymmetric encryption that uses a public key to encrypt and a private key to decrypt the message.

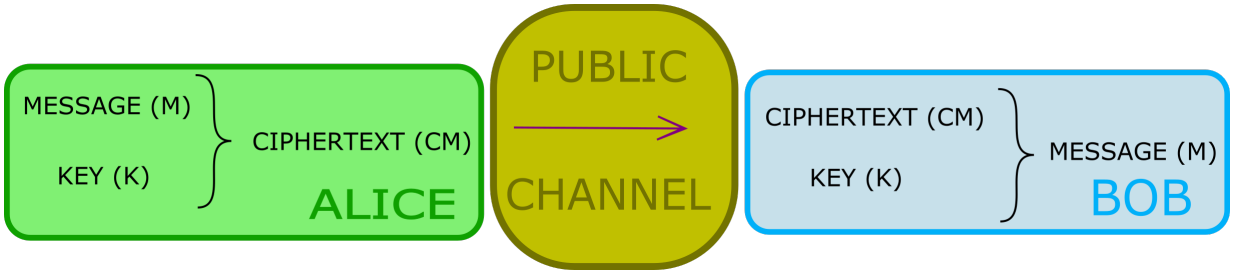

Figure 1: An example of symmetric key cipher.

First of all, let us take a look at some classical ciphers. For example, the Caesar cipher (also called shift cipher) dates back to the time of the Roman Empire. A message is encrypted by shifting each of the letters down three positions in the alphabet [1]. If the shift reaches Z, it goes on with A. Obviously, to decrypt the ciphertext, one need to shift each of the letters three positions back in the alphabet. Such a cipher can be easily implemented. However, it can be easily broken. The Romans relied on illiterateness of their enemies.

A little bit advanced cipher was invented in the 16th century by Giovan Battista Bellaso. It is called the Vigenere cipher and it was used during the American Civil War or World War I [1]. The letters are also shifted in the same way as in the Caesar cipher. But the number of places one letter is moved is defined by a key. The key here is often way shorter than the message and is composed of letters as well. Each letter in the key represents a number given by its position in the alphabet. As the key is usually shorter than the message, the pattern of the key repeats until we encode the whole message. For example, the key BENG shifts the letters in a message by 2, 5, 14, and 7 positions. Those are just simple ciphers that can be easily broken with the current computing power available. For example, a method called frequency analysis might be used to break the Vigenere Cipher. This method is based on fact that some letters in a message are used more often than others. In English, the most common letter is E for instance. With the BENG key, every

4th letter of the message is shifted by the same number of positions. We can make the key as long as the message to solve this. However, there is another problem. We cannot use the same key several times as it would be easy to guess how the key looks. Consequently, to achieve perfect secrecy we have to use long secret key just once for each message. Another requirement is that the key has to be truly random. This method is called one-time pad and it is being implemented into the modern quantum cryptography. Here I would like to make a simple thought experiment to highlight that one-time pad offers an absolute security. Think of a variant of the Vigenere cipher. But let us use a random key that is as long as the message to encrypt it. Every letter of the message is shifted by a random number of positions. Note, that it cannot be more than 25 as there are just 26 letters of the English alphabet. The resulting ciphertext then looks also random. The chance for Eve to guess first character is $\frac{1}{26}$. To correctly guess the second character Eve has also probability of $\frac{1}{26}$. For a message of n letters, the possibility to guess the correct meaning is $\frac{1}{26^n}$. Such a probability is negligible for common key lengths and we are almost certain that Eve will no get the correct message.

So far we have considered the symmetric key ciphers. However, today's cryptosystems often use asymmetric encryption. Unlike symmetric encryption, there are two keys. One of them is utilized for encryption and it is called a public key. It is available to everybody. The second one is used to decrypt the message. It is called a private key. The ciphertext can be read only with help of the private key. The public key is generated from the private one. An example of asymmetric key cipher is the Rivest–Shamir–Adleman (RSA) cipher. The security here is based on a principle that some operation cannot be inverted. It uses so-called trapdoor permutation which is a function that turns a message into a ciphertext of the same length while opposite transformation is practically impossible or, at least, very difficult. RSA algorithms use large numbers $N = p * q$, where p and q are large prime numbers. The security here relies on the inability of today's computer systems to effectively factorize large numbers. The time needed to factorize N grows exponentially with the number of bits N has [2]. However, it is believed that there might be a way how to solve the factoring problem in the polynomial time. Although, it has been not discovered yet. Consequently, we would like to develop even more reliable cryptosystem. That brings us back to the one-time pad and its usage in quantum cryptography.

In classical cryptography one-time pad is hardly ever being used. The reason is that you have to generate a new key for every message. Both Alice and Bob have to share the key before Alice transmits the message. Moreover, they have to be sure that nobody else has knowledge about the key. They can meet in person but that is really impractical. The potential of one-time pad rise with an idea of quantum key distribution (QKD). The goal of QKD is to safely distribute a secret key between Alice and Bob. There is a whole range

of QKD protocols to this purpose. All of them use binary logic (note, that one-time pad using binary logic is called the Vernam cipher). QKD protocols use either non-orthogonal states or a pair of quantum entangled states to transfer a bit. To highlight the strength of the Vernam cipher I have used it to encode the logo of Palacký University as shown in Figure 2. Note, that the initial pattern is lost in the ciphertext.

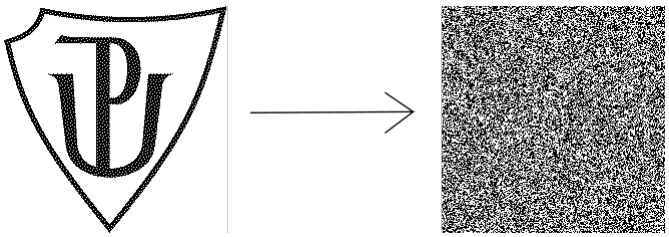

Figure 2: The logo of Palacký University encoded with a secret key using the Vernam cipher. The key was generated with the QKD system described in this thesis.

Let us take a look at QKD protocols using non-orthogonal quantum states. Bits can be encoded into polarization or phase states of photons [3]. Security is guaranteed by the Heisenberg uncertainty principle which says that quantum measurement disturbs the system [4]. Unlike classical physics, quantum measurement can give reliable results only in so-called measurement basis. Measurement basis is formed by orthogonal space vectors. Think of a single photon being measured with two detectors behind a balanced polarizing beam splitter. Assume that a horizontally polarized photon goes through the beam splitter and is detected by a detector 1. Whereas, a vertically polarized photon is reflected on the beam splitter at an angle of 45 degrees and is detected by a detector 2. We can be absolutely sure about the result, if and only if the incoming photon is horizontally (detector 1 clicks) or vertically (detector 2 clicks) polarized. These two polarization states of the photon (the orthogonal space vectors) form the measurement basis. However, if we send a photon in a balanced linear superposition of the measurement basis states, let us say the diagonally polarized photon, we will get a random result.

Now let us suppose that Eve would like to attack the quantum channel to learn the secret key. To do so, she has to perform a measurement on the information carrier. As we said, bits are being transferred as non-orthogonal quantum states. In other words, the states have non-zero overlap. If Eve takes a measurement on them she inevitably introduces disturbances and these errors can be detected. Quantum cryptography does not avert eavesdropping but allows us to reveal it. Alice and Bob distribute the key only via the quantum channel. If they detect the presence of Eve they can discard the key and try again. Once they obtain the secret key and they are sure nobody was listening, Alice can

send a ciphertext via public channel.

All those protocols suffer from the photon-number splitting attack (PNS). Classical light sources, such as the attenuated laser or light emitting diodes, are not perfect single photon generators as they obey a Poisson photon statistics (so-called coherent state). Even though the mean number of photons in a single pulse is lower than one, some pulses include multiple photons. Eve can split the channel and steal some of the photons to get particular information about the key. Of course, there are some protocols that aim to prevent PNS attack. As an example, see decoy-state protocols [5] or SARG [6].

Another class of QKD transfers the secret key with help of entangled photon pairs. Entangled states are correlated in such a way that a measurement made on one photon from the pair influences the state of the second one. The correlation is the same in all measurement bases. Alice and Bob perform the measurement on her, his photon respectively. Alice takes a measurement on her photon first which decides the measurement basis for that one bit. The measurement basis is often chosen from two or more basis while Alice and Bob choose their basis randomly and independently of each other. Some entanglement-based protocols, as well as those using non-orthogonal states, use a so-called passive setup where the measurement basis is chosen without modulators by a beam splitter [4]. These protocols have a lower quantum bit error rate and are resistant to PNS attack [3].

The aim of my experimental work is to build a simple system capable of a quantum key distribution over a short free-space channel. For that purpose, I use the B92 protocol which was suggested by C. H. Bennett in 1992 [7]. It is a simple superposition-based protocol which utilizes only two non-orthogonal states. I remark that the B92 is considered not to be secure. Apart from the PNS attack, there are other Eve's attacks, that can break the protocol. However, the system I have developed can be easily expanded into more advanced N–state protocols, $N \geq 3$ and decoy state protocols as it is shown in Figure 3 below. Such protocols basically differ only in the number of Alice's outputs and Bob's detecting channels. Additionally, various real channels may be tested with the proposed setup.

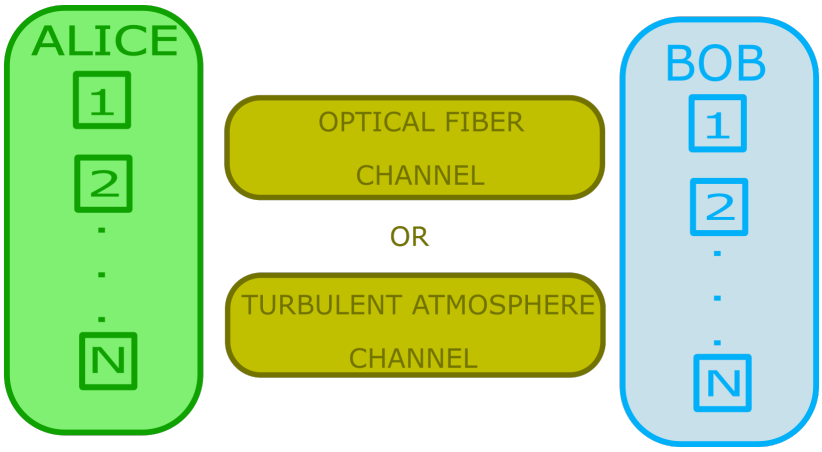

Figure 3: An outlook of the possible expansion of the developed setup.

Even though I use only two state protocol, I have solved all the technical challenges related to the quantum key distribution. It includes constructing both Alice and Bob devices and ensuring indistinguishability of Alice's outputs in their degrees of freedom that do not carry information. Synchronization of the signals with the clock has also been solved. Moreover, the whole system has to be controlled and the data from detectors analyzed with a suitable software, and one need to be equipped with a true random number generator, which is also handled in this Thesis.

Further text is structured as followed. First of all, I will present the B92 protocol and its experimental implementation. Secondly, I will describe the experimental setup. Next, I will explain how the experiment is controlled and the key is generated. In the end, I will discuss the limitations of my cryptosystem and further plans and improvements.

2 The B92 protocol implementation

Let us take a look at B92 protocol that I use for testing of the QKD. It was presented in 1992 by C. H. Bennett [7]. The security of this protocol relies on the inability of Eve to distinguish unambiguously and without perturbation between the two non-orthogonal states that Alice sends to Bob [3]. Two non-orthogonal states are sufficient for the purpose of quantum key distribution. In my case, the polarization encoding is utilized. Alice encodes the bits into horizontal (H) and diagonal (D) linear polarization states. Bits are generated randomly and sent to Bob. Each bit is sent and detected in a single time window. Moreover, there are on average bellow 0.1 photons in every pulse. I assume, that number of photons generated by such an attenuated coherent state follows the Poisson statistics. A lot of pulses are empty in that case. They contain zero photons. However, this also reduces the number of multi-photon instances bellow 1 %. On the other side of the channel, Bob measures the incoming photons as a projection into antidiagonal (A) and vertical (V) linear polarization states. For each bit, he randomly chooses his projection used. In other words, Bob detects the states orthogonal to Alice's states. The reason is to minimalize the error rate of the protocol. Obviously, Bob does not get a detection, if Alice sents (H) state while Bob is expecting a (V) state. This leads to a transmission rate decrease. In 50% of cases, Bob does not get the detection even though he chooses right projection. We can express that as follows:

$$\begin{aligned} |\langle A|H\rangle|^2 &= \frac{1}{2}, \\ |\langle A|D\rangle|^2 &= 0, \\ |\langle V|H\rangle|^2 &= 0, \\ |\langle V|D\rangle|^2 &= \frac{1}{2}. \end{aligned} \tag{1}$$

Now we will take a look at my implementation of the B92 protocol in detail. Following lines will demonstrate how I distribute the secret key. The scheme of the protocol is shown in Table 1.

1a	1	0	1	1	0	0	0	1	0	0	1	1	0	1	0	1	0	1	1	0	0	1	0	1
1b	H	D	H	H	D	D	D	H	D	D	H	H	D	H	D	H	D	H	H	D	D	H	D	H
2a	1	1	0	1	0	0	1	1	1	0	1	0	0	1	1	0	1	1	0	0	1	1	0	0
2b	A	A	V	A	V	V	A	A	A	V	A	V	V	A	A	V	A	A	V	V	A	A	V	V
3a	yes	no	no	yes	yes	yes	no	yes	no	yes	yes	no	yes	yes	no	no	no	yes	no	yes	no	yes	yes	no
3b	yes	-	-	-	yes	-	-	yes	-	yes	-	-	yes	-	-	-	-	yes	-	-	-	yes	-	-
Bob calls Alice																								
4	1	-	-	-	0	-	-	1	-	0	-	-	0	-	-	-	-	1	-	-	-	1	-	-
5a	1	-	-	-	0	-	-	0	-	1	-	-	0	-	-	-	-	1	-	-	-	0	-	-
5b	yes	-	-	-	-	-	-	-	-	yes	-	-	-	-	-	-	-	yes	-	-	-	-	-	-
6	-	-	-	-	0	-	-	1	-	-	-	-	0	-	-	-	-	-	-	-	-	1	-	-

Table 1: Quantum key distribution, 1a (1b) are bits (polarization states) Alice sends, 2a (2b) are bits (projection) Bob expects, 3a shows instances of the possible detection, 3b are instances of detection, 4 stands for a raw key, 5a is a verification mask, 5b presents a result of verification, and 6 represents a secret key.

As mentioned above, in my quantum key distribution process, each bit is sent and detected in a single time window. Alice generates a sequence of random numbers, logic levels 1 and 0 (line 1a). Each bit is encoded into a polarization state. Let us encode the logic 1 into horizontal polarization (H) and logic 0 into diagonal polarization (D) as it is shown in line 1b. Bob generates a sequence of random bits on his own (line 2a). The random binary number generated by Bob determines the projection in which Bob detects (line 2b). Bob detects incoming states as a projection into antidiagonal (A) and vertical (V) states. Obviously, Bob can obtain the bit if and only if he expects the same bit as Alice sends. This is represented by line 3a where *yes* stands for a possible detection in the time window which can occur, if and only if the 1a and 2a are the same. However, Bob gets detection only in about half of the instances (line 3b) because of the equation (1). Bob publicly informs Alice in which instances (time windows) he detected a photon (but he does not tell, which result he got) and other bits are discarded. Line 4 presents a raw key obtained. Both parties now have to verify that Eve did not attack the channel. Next, Bob generates a random mask that chooses a subset of the raw key to be checked (line 5a). In the line 5b, there is the result of the comparison between 1a and 2a according to the mask 5a. The subset used in verification process cannot be further used as the key. Bob and Alice finally get their secret key as it is presented in line 6.

3 Experimental setup

Scheme of my experimental setup (see Figure 4) consists of the transmitter (Alice), a channel, the receiver (Bob), and the control electronics. In this section, we will discuss all the parts and the components used.

Let us take a look at Alice's part of the setup. As a light sources, I use OSRAM 850 nm light emitting diodes (LEDs). Spectral bandwidth at 50% of the maximal intensity (FWHM) is 30 nm which suits very well for the purpose of QKD. The spectral emission of any two light emitters of the same type typically slightly differs. Consequently, laser diodes with narrow spectral bandwidth may be partially distinguishable. Rise and fall time of the intensity of the LEDs is 12 ns. That allows us to produce short pulses down to 24 ns. Light emitted by LEDs is coupled into a multi-mode optical fibers (MMF) via fiber launch systems (FLS). At the end of MMF the light is outcoupled into free space and collimated via another FLS. The light then goes through a linear polarizer (LP) to obtain desired polarization state. Two non-orthogonal states used are horizontal (H) and diagonal (D). The two beams are then superimposed using balanced non-polarizing beam splitter (BS). Finally light goes through a set of neutral-density filters (ND filter) to be attenuated and, optionally, spectral filter is added to makes LED's spectra indistinguishable.

There is a mirror in my experimental setup to steer the beams which is not shown in Figure 4. It is caused by the composition of the components on the laboratory breadboard table. Unfortunately, the mirror induces a phase shift between vertical and horizontal polarization components of the light. Therefore, before the light is transmitted to Bob it goes through a tilted waveplate (WP) rotated to 0 degrees to recover initial polarization state of light that is changed during propagation in Alice's module. Bits are then sent through 10 cm long free-space channel.

Bob's stage looks very similar. The beam is split by BS and analyzed by two LPs. The polarizers used in Bob's module are set to project onto antidiagonal (A) and vertical (V) linear polarization states. Note, that apart from losses caused by the protocol, there are additional 50% losses caused by the BS. Consequently, only every fourth photon would reach the detector if there were no another inserted losses and just one photon in every pulse while the quantum efficiency is not taken into account. Each beam is then coupled into a multi-mode optical fiber via FLS. At the end of MMF, photons are detected by single photons avalanche diodes (SPADs) from Excelitas. Detection of a photon is followed by an emission of an electron and a hole in. These free carriers are accelerated by an electric field to achieve impact ionization. An avalanche current is generated and measured. After the detection, the avalanche has to be quenched which takes some time (so-called

dead time). During the dead time any other detection cannot occur. Dead time of SPADs employed is about 23 ns and photon detection efficiency is about 56 %. Dark count rates are $DC_1 = (53 \pm 1)$ Hz and $DC_2 = (77 \pm 1)$ Hz when a statistical set of 100 s of dark count measurement was evaluated. Dark counts are random detection while there are no photons present. They are caused by the thermal noise and increase the bit error rate. The scheme of the setup including all the electronics is pictured below.

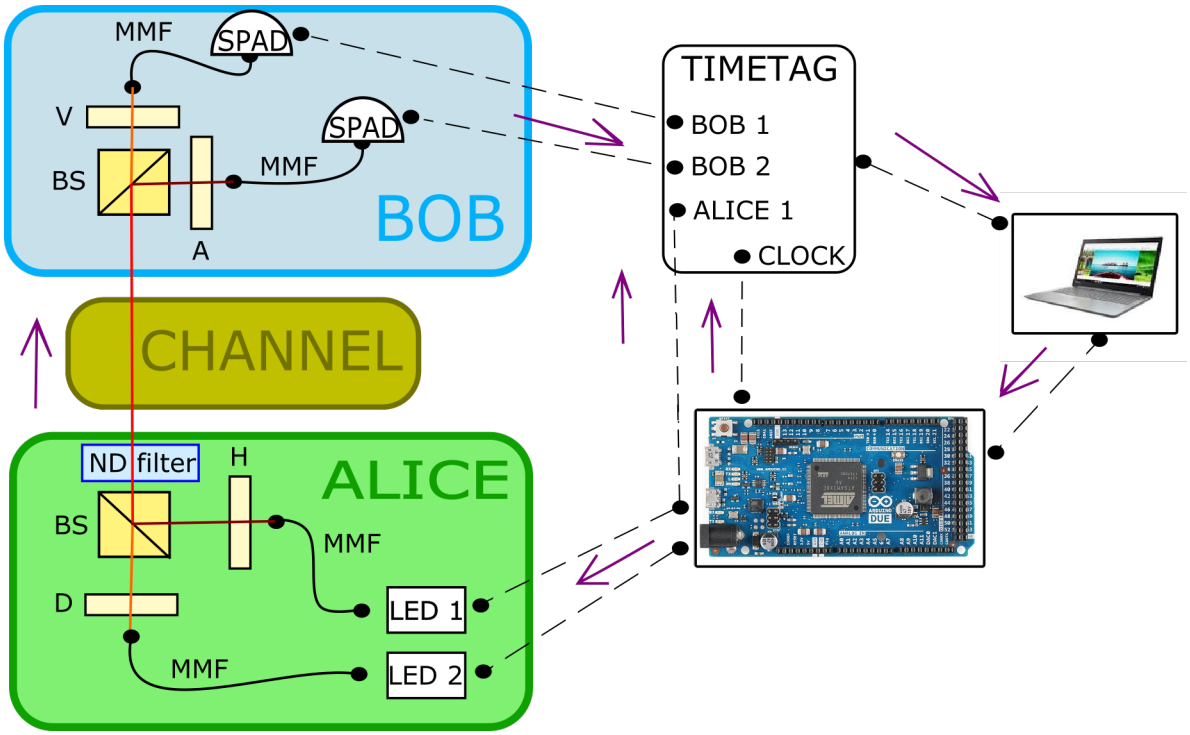

Figure 4: The scheme of the QKD setup. LED – light emitting diode, MMF – multi-mode optical fiber, BS – beam splitter, ND filter – set of neutral-density filters, SPAD – single photons avalanche diode detector, H(D) – horizontal (diagonal) polarization state preparation, A(V) – antidiagonal (vertical) polarization state measurement.

The whole experiment is controlled by Arduino Due, which is a microcontroller board based on the Atmel SAM3X8E ARM Cortex-M3 central processing unit (CPU) [8]. In what follows I will denote the control system simply as Arduino. The first task of this system is to produce a clock for both Alice and Bob. The clock is an electric square wave signal. A single period of the wave defines a time window in which Alice sends a single bit and Bob measures it. The CPU of the Arduino also provides a hardware random number generator (HRNG). It is a crucial feature that will be discussed later in a separate section.

Another necessary device is a time-to-digital converter (timetag) from UQDevices which allows us to keep record of up to 16 channels with resolution 156.25 ps [9]. Although, I utilize just 4 channels.

Now let us take a look at how exactly Arduino controls the experiment. HRNG of the Arduino generates a 32-bit long sequence of logical numbers 0 and 1. According to the number generated the LEDs are switched on or off. If I get a logic 1, LED 1 is switched on, LED 2 is off, and H bit is sent to Bob. Alice transmits a D bit if logic 0 is obtained. For every bit generated, Arduino sends the clock to the timetag, which carries the information that the period began. Moreover, every time a H bit is sent, Arduino also emits another electronic signal to the timetag. That allows us to monitor which bits are actually transmitted because H and D are complementary. Whenever one of the SPADs at Bob's side detects a photon, it sends an electronic pulse to timetag. In total, there are 4 channels of timetag to be analyzed. All of them are synchronized so the signals arrive at the same time. It is realized by coaxial cables of different lengths to keep the same propagation time of all the signals. Note that quantum key distribution over long distances has to utilize different time synchronization. Alice and Bob cannot share the same clock in general. Each of them uses its own clock and they are synchronized via global positioning system (GPS) or Alice periodically sends a strong synchronization sequence. The frequency of my clock is 2.3 MHz which makes a single time window for both the generation and the detection of a bit about 390 ns wide. However, the actual optical pulse transmitted has a duration of 180 ns. Less than a half of the time window is occupied by the signal to make the windows easily distinguishable during analysis as will be shown later in section 5.

The synchronization of all the signals is a crucial feature that makes quantum key distribution possible. Alice and Bob have to somehow synchronize the clock. Otherwise, each of them obtains a random raw key. To demonstrate that, let us take a look what would happen if the propagation time was not synchronized. Assume that the bits Alice sends are shifted by one clock cycle and Bob receives a bit while he is expecting the previous one. Desynchronized quantum key distribution is demonstrated in Table 2 below.

N	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
1a	1	0	1	1	0	0	0	1	0	0	1	1	0	1	0	1	0	1	1	0	0	1	0	1
2a	-	1	1	0	1	0	0	1	1	1	0	1	0	0	1	1	0	1	1	0	0	1	1	0
3a	-	no	yes	no	no	yes	yes	yes	no	no	no	yes	yes	no	no	yes	yes	yes	yes	yes	yes	yes	no	no
3b	-	-	yes	-	-	-	yes	-	-	-	-	yes	-	-	-	-	yes	-	yes	-	-	yes	-	-
Bob calls Alice																								
4a	0	0	1	1	1	0																		
4b	1	0	1	0	1	1																		
5a	1	0	0	1	1	0																		
5b	no	-	-	no	yes	-																		

Table 2: An example of desynchronized quantum key distribution (a demonstration).

The upper line in Table 2 presents positions of the bits N as Bob receives them. QKD begins same as in section 2. Alice generates a random sequence of binary numbers (line 1a). Of course, she encodes them into appropriate polarization states. In line 2a, I see what bit Bob expects according to the projections which he randomly chooses to detect in. Line 3a represents in which instances a detection can occur. Positive detection is shown in line 3b. Now Bob calls Alice and tells her in which time windows he measured a bit. From his point of view, they are 2nd, 6th, 11th, 16th, 18th, and 21st windows. The raw key Bob obtains is in line 4a. However, Alice's bits in those instances are different. Hence, Alice obtains a different raw key (line 4b). The pattern of obtaining lines 4a and 4b is depicted in the figure below.

N	0	1	2	3	4	5	6
1	1	0	1	1	0	0	0
2	-	1	1	0	1	0	0
3a	-	no	yes	no	no	yes	yes
3b	-	-	yes	-	-	-	yes

Figure 5: The shift between Alice's and Bob's raw key during the desynchronized QKD (ademonstration).

Now Alice and Bob verify a subset of the key. Bob randomly generated a mask to choose a part of the raw key to be checked (line 5a). In line 5b the raw keys of Bob and Alice (4a and 4b) are compared according to the mask 5a. Because of the desynchronization, Alice and Bob find out that quantum bit error rate has increased even though there was no evesdropper.

4 True random number generator

In this section, we will discuss how important the randomness of the key in the quantum key distribution is. For that purpose, we should mention the Vernam cipher. It is an one-time pad utilizing the binary logic. It requires a secret key as long as the message is. To combine both the key and the message the exclusive-or logical operation (XOR) is used. This outputs logical 1 if and only if the inputs differ to each other as we can see in Table 3.

Input		Output
XOR		-
0	0	0
0	1	1
1	0	1
1	1	0

Table 3: XOR logical operation truth table

Obviously, combining a truly random key with a message we obtain a truly random ciphertext. Such a ciphertext no longer carries any information and cannot be read without the key. However, it works only if the key does not possess any regularities or correlations. Which makes the true random number generator a necessary condition for any secure QKD protocol.

As mentioned before, I use a CPU of Arduino for direct digital pin addressing and as HRNG. However, it is not a trivial task to address the CPU directly. Therefore, in the rest of this section, I will discuss advantages and methods of the direct addressing.

Basically, there are two ways how to use the Arduino. One can use standard Arduino library functions to control the LEDs or the clock. There is also a function called *random* to generate pseudo-random numbers. However, one cannot address multiple pins of the Arduino board at once. That can be compensated with a suitable software though. But still, it is an unpleasant feature. Also, the library functions are very slow. I achieved only 2700 Hz transmission rate using Arduino library functionality. Addressing the CPU directly solves all these problems. Let us look at direct pin addressing. Hence, one needs to know the SAM3X pin name to directly address the pin [10]. Below, there is an example of different names of the same pins for both board and CPU in Arduino programming language.

```
#define NOP __asm__ __volatile__("nop\n\t")

int led1 = 51;
int led2 = 50;
int Clock = 49;
int Alice = 48;

int pin1 = 12;  // Direct addressing of 51-board-position pin.
int pin2 = 13;  // Direct addressing of 50-board-position pin.
int pinC = 14;  // Direct addressing of 49-board-position pin.
int pinA = 15;  // Direct addressing of 48-board-position pin.
int i;
uint8_t bl,bQ; // Defines unsigned byte - 8 bits.
uint32_t r; // Defines unsigned 32-bit-long string.
```

I also use *no operation instruction* (NOP) as you can see above. That makes CPU to do nothing for one cycle, long about 12 ns. Now we will describe the true random number generator `trng` function. It is a built-in function of the CPU. And, it is significantly faster than the Arduino library `random` function. Additionally, the numbers generated by the `trng` passes the American NIST Special Publication 800-22 and Diehard Random Tests suites [11]. The `trng` function has to be enabled first.

```
void setup() {
  pmc_enable_periph_clk(ID_TRNG);
  trng_enable(TRNG);
  pinMode(led1, OUTPUT);
  pinMode(led2, OUTPUT);
  pinMode(Clock, OUTPUT);
  pinMode(Alice, OUTPUT);
```

Finally, let us look at the direct pin addressing. First, I use `trng` to generate a 32-bit-long binary number. Each bit represents the state Alice will send to Bob. Next, I use several binary operations to achieve direct pin addressing. Essentially I deliver a voltage to the pins which turn on the appropriate LED and the clock. For example, to switch on the LED 1 (CPU pin position 14) and the clock (CPU pin position 13) at the same time, I need to generate 110000000000000 binary number. Each position of such a number represents one pin from 0 to 14 (taking from the right). Obviously, 1 means there is the voltage on the pin. For an interested reader, there is the rest of the code demonstrated below.

```
hile(1)
  {
  r = trng_read_output_data(TRNG); //Random generator.
  for (i=0;i<32;i++)
  {
    bl = r & 1; // Least significant bits.
    r = r>>1;   // Shifting the binary number "r" to right.
// That allows us to use second bit in the next cycle.
    if (bl == 1)
    {
      REG_PIOC_ODSR = ((1<<3)+(1<<2)+1) << pin1;
// Here I directly address pins 15, 14, and 12.
      NOP;NOP;NOP;NOP;NOP;
// This makes about 60 ns long pause to ensure
// that both loops take the same amount of time.
    }
    if (bl == 0)
    {
      REG_PIOC_ODSR = ((1<<1)+1) << pin2;
// Here I directly address pins 13 and 14.
    }
    REG_PIOC_ODSR = 0;
// Every pin is turned down.
  } } }
```

5 Data analysis

In this section, we will closely look at the process of obtaining a secret key from all the data captured by the timetag. The time resolution of the timetag is 156.25 ps (a bin). All the signals are synchronized and they arrive within a single time window 390 ns (about 2500 bins) long. As mentioned before, timetag keeps a record of 4 signals (clock, LED 1, SPAD 1, and SPAD 2). The light from LEDs propagates through the experiment 0.8 m of free space and 4.5 m of the optical fiber before it reaches one of the SPADs. It takes about 25 ns. Also, it takes another 15 ns for SPAD to generate an electronic signal after the detection. To roughly synchronize all 4 signals at the input of the timetag, the propagation time of clock and LED 1 is delayed by additional 8 m of coaxial cable which makes about 40 ns delay.

The signals from the clock and LED 2 arrive at the same time at the beginning of each time window. The signals from the SPADs are delayed. I keep the record of detections which arrive up to 1200 (187.5 ns) bins after the signal from the clock is detected which corresponds to a duration of optical pulse generated by Alice. Notice that the FWHM of the light pulse produced by the LEDs is about 180 nm. The photon can be generated whenever during this time which causes the 1200 bins long gap when the photon detection is expected with high probability. Additionally, SPADs also take a variable time to generate the electric pulse after a photon detection (jitter). This feature is negligible as the jitter is below 0.5 ns. If any SPADs clicks outside the 1200 bins long gap the detection is considered as a dark count and is not analysed.

To clarify the process of detection, see Figure 6 where the illustration of three cycles of the clock is pictured. The clock signal is depicted as a green line. The red line presents a relative probability of detection of a photon and the blue points represent real click of any detector.

The data are recorded as binary tags. Each tag presents a single occurrence on any channel such as a beginning of the time window, any detection, etc. Data are analyzed in smaller batches to fasten the process. Each of them is 1000 tags long. Clearly, there are multiple tags in every time window. There is always a tag representing a beginning of the clock cycle. For every tag of the clock, I look whether there are some tags in the other channels in the time window. In half of the cases, there is a signal from the LED 1 which means that Alice transmits the logical 1 in this window. Otherwise, I know the logical 0 was transmitted. The probability of detecting a bit is only $P = \frac{N_d}{N_t} = 0.0748$. N_t stands for the total number of bits sent by Alice and N_d stands for the number of bits detected by Bob. Taking into account the clock repetition rate of 2.3 MHz, the key rate is about 170 kHz. I

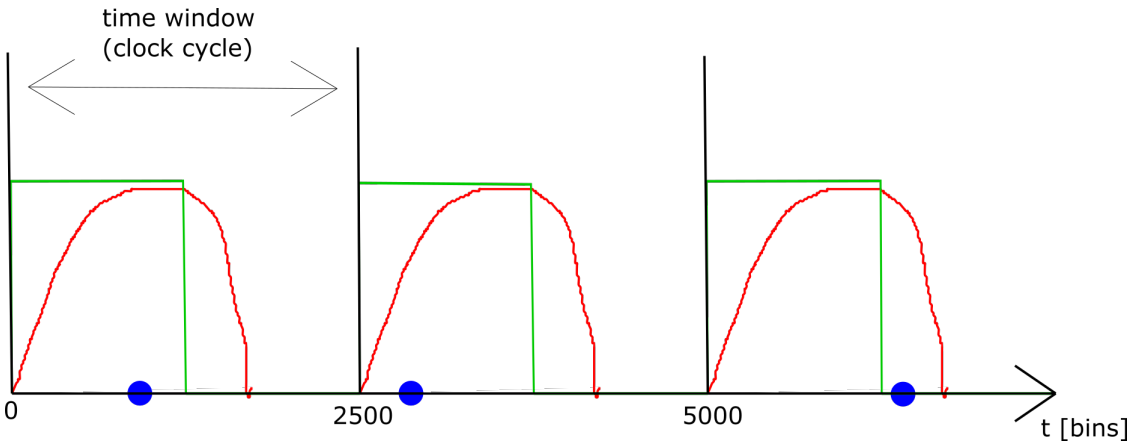

Figure 6: An illustration of three cycles of the clock. The green line is the clock signal, the red line is the relative probability of detecting a photon and the blue points are actual detection.

evaluate a positive detection if and only if there are tags just from one SPAD while multiple detections from the SPAD are also counted. If the time window does not contain any tags from channels of the SPADs or both of the SPADs clicks, the time window is discarded

If the bit is detected, I note its position and the detector which the detection occurred on. Once the whole batch of 1000 tags is analyzed, the software moves to the next one. The last time window of the batch and the first window of the following batch are discarded to prevent processing of incomplete clock cycle. After all of the bits are analyzed, I have a complete knowledge about bits Alice sent and I know in which instances there has been a detection.

However, to follow the B92 protocol, Bob has to randomly alternate his measurement projection. It is solved by postselection. Bob generates a random number to determine whether he choose V or A projection. Thus half of the detected bits is discarded, the rest of the bits represents the raw key for Bob. Classical communication between Alice and Bob is ensured in a computer in my case. As I know what positions in the transmission have all the bits in Bob's key, Alice obtains her raw key just by taking the corresponding bits in her record of bits sent. Positions of the detected bits are converted into a logic 0 or 1 according to the detector which the detection occurred at. The click of SPAD 1 presents logic 0 and the click of SPAD 2 presents logic 1. Security verification, tests of bit error rate, and tests of randomness follow next in every QKD protocol. However, I skip the verification process it is not the goal of this Thesis. I have performed only bit error rate evaluation.

Note that errors caused by dark counts are decreased as I expect the detection only

in a small part of the clock cycle. I have developed a software to test the bit error rate. In my case, I compare Alice's and Bob's key and evaluate their difference. The bit error rate is then given as $QEBR = \frac{n}{N_d} = 0.83\%$, where n stands for the number of instances in which Alice's and Bob's key differs.

6 Discussion

In this section, we will discuss further some difficulties related to the implementation of QKD. We will mention other requirements on Alice which allow her to resist certain Eve's attack. Next, we will talk about the difference between QKD utilizing polarization encoding in free space and optical fibers. After that an imperfection of Bob's detectors on key transmission will be discussed. At the end, basic ideal of authorization proeess will be shown.

First, let us take a look at Alice's transmitter again. To ensure secure QKD, Alice has to meet certain standards for the outcoming light pulses. The spectral and the spatial indistinguishability of the beams, the same intensity of them, and, last but not least Alice has to prepare correct non-orthogonal states. If the balanced intensity is not guaranteed, Eve could use PNS attack on the stronger beam to gain the knowledge of the key. Moreover, higher intensity of one LED would lead to a higher probability that Bob detects a 1 bit or 0 bit as there is a higher probability that Alice transmits 1 bit or 0 bit, respectively.

The spatial indistinguishability, which is an issue in QKD protocols utilizing free-space channel, prevents specific Eve's attack. Assume that the two beams would be spatially distinguishable, i.e. the beams would not overlap. In that case, Eve might be able to eavesdrop a fraction of the states Alice sends. Or if she splits the beams entirely, she distinguishes all of the states. Additionally, free-space quantum cryptography faces some hindrances. Any beam diverges which introduces additional technical difficulties.For long distance transfer, Bob has to collect the incoming light from a large solid angle and use spectral filters to eliminate stray light. Air turbulence should be also mentioned as it affects the signal and can lead to fluctuating loses, even though polarization encoding in general suits very well for the purposes of free-space QKD as the polarization states do not suffer from decoherence during the propagation. A polarization encoded B92 QKD cryptosystem was recently implemented over 143 km long free-space channel [12].

Obviously, some of these problems do not occur if an optical fiber would be used as the channel. However, there is another obstacle resulting from utilizing the optical fiber as the channel. Polarization encoded states suffer from deformations and ultimately a decoherence during the propagation in the optical fiber which gets woese over long distances. QKD in the optical fiber is limited up to dozens of km. It is possible to actively compensate the polarization changes induced in the fiber over a short distance. Alternatively, another encoding (e.g. time-bin encoding) can be used.

We see that it is not an easy task to maintain the polarization state during the propagation. However, the states can be also damaged while being prepared by Alice. In my experimental setup (see Figure 4), there is BS in Alice's module which combines the two optical signals from LEDs into a single beam. The transmittance of the BS is different for horizontal and vertical polarization components. Consequently, D polarization is changed after the BS. For this reason, the polarizer which prepares D state is not set to 45°but to 37°. Moreover, the BS is not perfectly balanced. Hence, it has to be also precompensated by a different intensity of the LEDs.

Let us discuss the spectral indistinguishability now. Once more, spectral indistinguishability averts Eve to identify the bits, Alice sends. Different spectra of the LEDs would allow Eve to get information about the state by measuring the wavelength of the photon.
In Section 3 we mentioned that LEDs suit well to the purpose of QKD due to its wide spectrum. The spectrum of LEDs is plotted in the picture below.

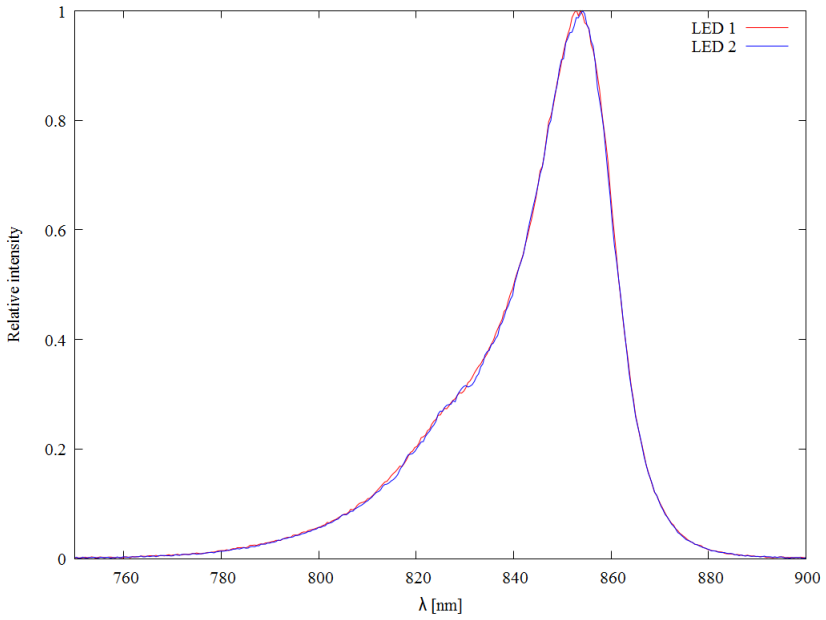

Figure 7: The spectra of Alice's LEDs representing H or D state.

A spectral filter can be used if the spectra do not perfectly overlap. It is an analogous problem to spatial overlap and spatial filtering. Suitable spectral filter at the end of Alice's module selects the part of spectra that both LEDs share. The spectra of my LEDs are practically the same thus I do not have to use the spectral filter. We can consider a spectral filtering when a dispersion in the issue.

Let us take a look at Bob's module. A crucial part here are the detectors. The quan-

tum efficiency of them and the dead time affects the key transmission rate. Obviously, a time between two light pulses should not be shorter than the dead time of SPAD as there cannot be any other detection before the SPAD is recovered. The dark counts increase the bit error rate. Effect of the dark counts can be decreased by gating the SPADs which means that the SPAD is activated (it is under voltage) if and only if the incoming photons are expected. In my setup, the SPADs are free-running. However, to decrease the error caused by dark counts, I do not analyze all the occurrences in time window as mention in the previous section.

Public discussion between Alice and Bob is an important part of QKD protocol, which ensures a verification of security. The discussion can be realized via the internet, but Bob has to be certain that he communicates with Alice and vice versa. There is always danger that Eve pretends to be Alice in front of Bob and at the same time act as Bob and chat with Alice (termed man-in-the-middle attack). As soon as Alice and Bob start the public conversation they have to go through authorization process which is implemented by a short ciphered message. The message is sent by one party before the public communication begins so that the recipient (e.g. Alice) knows it comes from a proper sender. It requires a secret key known only to Alice and Bob which they have to share before the first QKD. They have to inevitably meet each other at least once to do so. After each transmission, they can keep a short key for future QKD. Therefore, quantum cryptography does not really provide a secure tool of communication. It is more accurate to say that quantum cryptography provides an unlimited expansion of secret information [4].

7 Conclusion

Classical cryptography faces a threat of being no longer secure in the upcoming decades. The quantum mechanics provides a solution to this problem. Quantum key distribution offers unconditional security for symmetric-key cryptosystems which allows us to transmit a secret key over long distances. Such a key is then used to encrypt the message with the Vernam cipher.

In this Thesis, I have demonstrated a device capable of quantum key distribution over a short free-space channel. After a brief historical review and motivation, I have described a simple implementation of B92 protocol using polarization encoded weak coherent states. The B92 is a two-state protocol which does not provide a satisfactory security. However, we showed that the developed core of the system solves most of the challenges that we face in any real QKD tilizing weak coherent states and can be expanded into N–state protocols, $N \geq 3$, to ensure secure key transmission.

The experimental setup controlled by Arduino Due has been introduced. Arduino also operates as a shared clock between Alice and Bob with the frequency of 2.3 MHz. To achieve the frequency of the clock, the software has to address the Arduino pins directly. Two LEDs with identical spectra and FWHM of 30 nm are used as the light source. We have highlighted the importance of the time synchronization. The CPU of the Arduino also provides a true random number generator which is a necessary condition of the security.

For the purpose of key extraction, I have developed a software toolbox. It includes the analysis of the signals from the detector, the clock, and the reference signal captured by a time-to-digital converter. The final key distribution rate is 0.17 MHz. Furthermore, the software evaluates the bit error rate to be 0.8%.

Finally, the difficulties related to QKD utilizing polarization encoding are discussed. Further work should lead to the improvement of the quantum key distribution parameters over realistic quantum channel such as long optical fiber or turbulent atmosphere.

References

[1] J.P. Aumasson, *Serious cryptography: A Practical Introduction to Modern Encryption*, San Francisco: W. Pollock, 2018.

[2] M. Dušek, *Koncepční otázky kvantové teorie*, Olomouc: Univerzita Palackého v Olomouci, 2002.

[3] N.Gisin, G. Ribordy, W. Tittel and H. Zbinden, *Quantum cryptography*, Rev. Mod. Phys. 74, p. 145-195, 2002.

[4] M. Dušek, N. Lütkenhaus, M. Hendrych, *Quantum cryptography*, Elsevier, Progress in Optics, 381-454, 2006.

[5] Y. Liu, et al. *Decoy-state quantum key distribution with polarized photons over 200 km*, Optics Express 18, 85878594, 12 April, 2010.

[6] V. Scarani et al., *Quantum Cryptography Protocols Robust against Photon Number Splitting Attacks for Weak Laser Pulse Implementations*, Phys. Rev. Lett. 92, 057901, 2004.

[7] C. H. Bennett,*Quantum cryptography using any two nonorthogonal states*, Phys. Rev. Lett. 68, 3121, 1992.

[8] Getting started with the Arduino Due, 2018, March 25, retrieved from *https://www.arduino.cc/en/Guide/ArduinoDue*.

[9] ID900 Time Controller, 2018, March 25, retrieved from *https://www.idquantique.com/single-photon-systems/products/id900-time-controller/*.

[10] SAM3X-Arduino Pin Mapping, 2018, March 30, retrieved from *https://www.arduino.cc/en/Hacking/PinMappingSAM3X*

[11] Atmel SAM3X8E ARM Cortex-M3 CPU data sheet, 2018, March 30, retrieved from *http://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-11057-32-bit-Cortex-M3-Microcontroller-SAM3X-SAM3A_Datasheet.pdf*

[12] G. Vallone, et al. *Adaptive real time selection for quantum key distribution in lossy and turbulent free-space channels*, Phys. Rev. A 91, 042320, 2015.