Delft University of Technology
Master's Thesis in Embedded Systems

Design of a network stack for directional visible light communication

Lennart Pieter Klaver

Design of a network stack for directional visible light communication

Master's Thesis in Embedded Systems

Embedded Software Section
Faculty of Electrical Engineering, Mathematics and Computer Science
Delft University of Technology
Mekelweg 4, 2628 CD Delft, The Netherlands

Lennart Pieter Klaver
l.p.klaver@student.tudelft.nl

Monday 6th October, 2014

Author

Lennart Pieter Klaver (l.p.klaver@student.tudelft.nl)

Title

Design of a network stack for directional visible light communication

MSc presentation

13 October 2014

Graduation Committee

Chair: Prof. dr. K.G. Langendoen, Faculty EEMCS, TU Delft
Committee member: dr. ir. Z. Al-Ars, Faculty EEMCS, TU Delft
Committee member: dr. ir. M.A. Zúñiga Zamalloa, Faculty EEMCS, TU Delft

Abstract

During the last years, Visible light communication (VLC), a novel technology that enables standard Light-Emitting-Diodes (LEDs) to transmit data, is gaining significant attention. In the near future, this technology could enable devices containing LEDs –such as car lights, city lights, screens and home appliances– to form their own networks. VLC, however, is currently limited to point-to-point communication. To unleash the full potential of VLC, more advance network capabilities are required. This Master Thesis presents the design and implementation of a novel research platform aimed at distributed multi-hop visible light communication. Compared to state-of-the-art platforms, our platform provides similar data rates and coverage, but adds two unique characteristics: (i) 360° coverage, which is necessary to investigate an important property of LED communication: directionality, and (ii) a flexible design, which allows our platform to be connected to many experimental boards such as Arduino, Beaglebone, Raspberry Pi and sensor nodes. We evaluate the communication properties of our board (link quality, neighbor discovery and packet forwarding), and hope that our work will lower the entry barrier for members of the pervasive and networking communities to investigate and exploit future LED-based networks.

Preface

This Master thesis is a report on the work I did during the past year. I have always been interested in Embedded Systems. As a child I wondered what was going on inside these black squares I saw when opening electronic devices in and around the house. These things, microprocessors, apparently did magic tricks that made the most amazing things possible. Planes, rockets, TV, machines on Mars..

During my Master I became more and more conscious about the crucial part that Embedded Systems play in our society. You can even ask yourself if we could have become such a society without computer technology. Today everything around us is communicating with us and a big field of research is still open. The Internet of Things is a starting field in which embedded systems will solve problems together without the users' inference. In the next decades the borders between humans and technology will fade, and communication will play a central role in this. My interest in the topic of this thesis is sparked by this kind of developments, and I hope the reader finds inspiration for new ideas in this thesis.

I am using this opportunity to thank everyone that supported me during my thesis work. Special thanks for the guidance of my thesis go to Marco Zuniga, our talks about the thesis work sparked new ideas that hopefully will find their way into new possibilities. I would also thank the Embedded Software department for listening to my ideas and the support during the development of the hardware. Thanks go to Domenico Giustiniano, Daniele Puccinelli and Qing Wang for feedback and ideas on this project. Writing a thesis takes a lot of energy and time, therefore I want to thank my colleagues for their time, patience and reflective ideas. Family is the backbone of everything, my deepest gratitude goes to them.

Lennart Klaver

Delft, The Netherlands
Monday 6th October, 2014

Contents

Chapter 1

Introduction

For the last century, radio frequency (RF) has dominated wireless communications. The early work of James Clerk Maxwells on electromagnetic theory, followed by Heinrich Hertz inventions on radio systems and the miniaturization of transceivers achieved with semiconductors, have taken RF to the central role it plays on today's communications systems.

But RF has been a victim of its own success. The number of mobile devices and embedded systems has increased tremendously as technology becomes a primary need in peoples lives. This explosion of mobile devices comes at a cost: the frequency spectrum is a scarce resource. The massive increase in RF communicating devices is leading to a saturation of the available bandwidth, which will result in a drop of quality-of-service.

To ameliorate the bandwidth saturation problem, the research community has been exploring other wireless technologies. One of the most promising alternatives is visible light communication. Light is electromagnetic radiation just like RF, but the difference lies in frequency. Because of this frequency, the interaction between light and matter is different on a fundamental level which gives light unique properties.

With the advent of Visible Light Communication (VLC), the widespread exploitation of the visible light spectrum is becoming a reality. VLC enables standard Light Emitting Diodes (LEDs) to transmit data wirelessly, and this is an important step because LEDs are permeating our daily environments at a very fast pace. After intensive research into energy consumption, in 2009 the European Union and other countries started measures to phase out incandescent light bulbs in favor of high efficient LEDs. But it is not only residential and commercial lighting that is being replaced with LEDs, a number of other objects such as car lights, city lights, billboards, smartphone and laptop screens, price tags, toys and home appliances, are also using LEDs to reduce their energy consumption.

A nice VLC example can be found in stores: for many years stores use paper labels to show pricing and product information at the front of the shelf. As pricing becomes more dynamic, stores had to replace price labels more often. A solution is now offered in the form of an electronic label which uses a solar panel not only to harvest energy but also to receive the updated price via VLC.

Thus, considering that VLC can potentially transform any LED device into a wireless transmitter; in the near future, there may be a new generation of objects waiting to be networked in a distributed and multi-hop manner

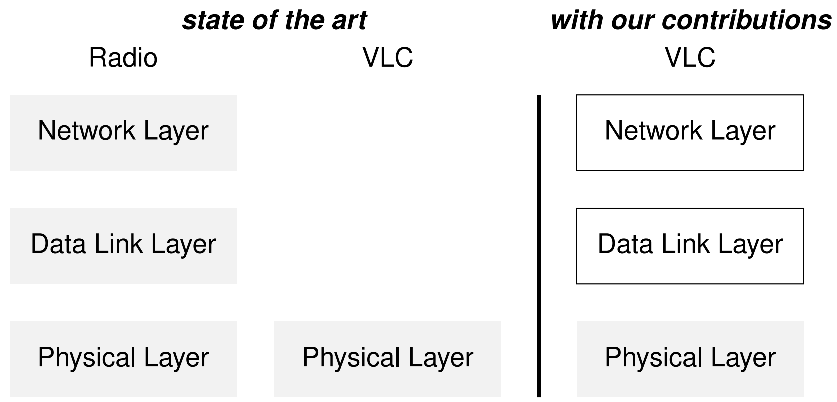
1.1 Problem definition

Visible light communication has gained significant attention in the last five years, but most of this interest is limited to point-to-point communication. A high quality data link is the first step for VLC, so most research topics focus on the improvement of this link in terms of distance coverage, capacity and stability. The fact that VLC can provide communication between nodes is a great leap forward, but the communication still remains pairwise. From our experience in networking we know that the value of a network is proportional to the square of the number of connected nodes (Metcalfe's Law), so we need to look beyond pairwise communication to further increase the value of (future) VLC networks.

To unleash the full potential of VLC, we need distributed multi-hop communication. In the future there will be scenarios that could exploit such networking capabilities, and current technology lacks support for this. Take the example of Smart Traffic, where the LED lights of cars could be used to exchange information about the vehicle's mass, speed, size and driving patterns to achieve cooperative tasks such as collision warning, adaptive cruise control and trajectory determination. Combining the capabilities of VLC for point-to-point communication with distributed multi-hop methods can allow us to fully exploit (future) LED networks.

1.2 Thesis contributions

The aim of this thesis is to propose an embedded platform that facilitates research on *distributed multi-hop* VLC, in both hardware and software. As the figure below shows, the network stack of VLC is not at par with radio. This thesis is a first step into filling this gap.



More concretely, the specific contributions of this thesis are:

- We delve into what has been mainly Physical Layer work in visible light communication, and take a ubiquitous computing perspective to identify the key elements and methods to design a board that is amenable for the pervasive and wireless networking community.

- We implement a platform that hides the low level complexities of visible light communication while exposing two unique features compared to the state-of-the-art: the ability to investigate the directional coverage of LEDs, and a flexible interface to connect to various platforms. The hardware schematics and accompanying software will be made freely available to the research community.

- We evaluate three basic communication characteristics of Shine at the Data Link and Network Layers: link quality, neighbor discovery process and packet forwarding.

It is important to remark that this thesis is only a preliminary step into an area that has not been investigated yet, and hence, there are many opportunities for improvement.

1.3 Dissemination and Collaboration

Results from this thesis has been used as part of lectures for a PhD course at Uppsala University, Sweden: Visible Light Communication, November 2013, `http://www.it.uu.se/grad/courses/gc1014/VLC13`, and for an invited talk at the EU COST Action IC1101 OPTICWISE: “Open Light: Software Defined VLC”, held in Graz, Austria on July 7-9, 2014.

1.4 Thesis organization

Now that the reader is introduced to the problem and the main topic, we will further explain VLC and give more information about our solution to the problem. First, in Chapter 2, we will explain the way visible light communication takes place. Then, in Chapter 3, we will state ideas for creating VLC networking, and those ideas are implemented in Chapter 4. Following this implementation chapter, in Chapter 5 we will evaluate the solution we created. The conclusion and future work can be found at the end of this thesis, in Chapter 6. Additional information that can be of interest to the reader, can be found in the Appendix.

Chapter 2

Background

VLC as we know it today would not be possible without LEDs. Before we introduce our implementation of networked VLC, we provide background information about LEDs as wireless transmitter, then we describe the many ways in which light intensity can be modulated and received, and the SoA on networked VLC.

2.1 Using LEDs as wireless transmitters

Modulation of light has been used for centuries. In 1792 the French inventor Claude Chappe invented the optical telegraph, a system of towers with signaling devices, which allowed Napoleon to pass messages throughout his empire. This idea was reused on ships in the form of a heliograph that used mirrors for telegraphy and used on-off-keying to modulate letters and numbers in light. Although simple and practical, these systems are the ancestors of VLC. For some decades after these events, light communication in the form of morse was used, but further development of light communication stood still because with incandescent lamps the data rates pale in comparison to what could be achieved with radio.

It is only until recently that we have a pervasive infrastructure that can modulate light to achieve data rates that are meaningful for today's communication needs. Due to recent advancements in LED technology we can transform light bulbs into high speed wireless transmitters. Data can be modulated in every light source using changes in intensity, but only certain light sources possess the necessary properties to transmit high data rates.

Fundamentally, modulating light requires changes of light intensity. For the last century incandescent lamps have been the primary source of light, but incandescent light cannot comply with high speed modulation because of the mechanism it uses to generate light. Incandescence is the effect of emitting thermal radiation from matter as a result of its temperature. In incandescent light bulbs a wire is heated by running a current through it, and the resistance of this wire forms kinetic energy which is released in the form of light. This means that intensity control of incandescent lamps takes place through two steps, resulting in indirect control of the signal. This would not be a problem if the thermal inertia would not make the system too slow for high speed modulation, but it does.

In the case of LEDs, the direct relation between intensity and electricity permit high speed modulation. LEDs consist of a semiconductor material that contains excited electrons. A fundamental property of electrons is that when they are forced into a lower energy state they release their energy in the form of the emission of photons. This effect is called electroluminescense and gives direct control over light intensity through the control of voltage and current. A simple circuitry with transistors can deliver the necessary control of current, and this makes the changes in light intensity fast enough to transfer information at a high data rate.

An alternative to LEDs is laser. Lasers can also be controlled at high speed, and have the additional capability of strengthening the electroluminescense effect by amplifying and focusing the generated light [7]. This makes laser a good candidate for long range communication, but for short distances its transmission angle is too narrow (and hence it can be easily obstructed) without using lenses. Another important disadvantage of laser is that poses health hazards to human beings.

2.2 Transmitting and Receiving data with Visible Light Communication

The light emitted by LEDs can be modulated in different forms, and there are also several types of receivers that can be used to decode the modulated light. Next, we describe all these options.

2.2.1 Transmitters

As mentioned before, the ability of LEDs to modulate light at high speeds make them the obvious choice for VLC transmitters in terms of speed and light intensity, but potentially good transmitters are not the only thing that make a communication system work. Data needs to be encoded (or modulated) into a signal before transmission.

In radio communication, two common parameters used in signal modulation are *amplitude* and *frequency*. Although radio and light are both electromagnetic waves, the effect of modulation is not the same. We will briefly describe this point to make the reader aware of the fundamental difference in modulation between radio and VLC.

In radio, frequency modulation changes the frequency of the waves in the electromagnetic field, and amplitude modulation changes the height of this waves.

In VLC, amplitude modulation works the same way as in radio, and it is often called intensity modulation. Frequency modulation however, is very hard to apply with visible light. It can be done but requires advanced laser setups to change the signal's properties, like phase and frequency, through interference. Instead, when people talk about frequency modulation in VLC, they mean that the *intensity* is shaped into a high frequency signal. By varying the frequency of this intensity pulses, we get frequency modulation.

So, frequency modulation in VLC is actually *frequency modulation through amplitude modulation*. This is the main difference between radio and light modulation. In this thesis, when we refer to frequency modulation, we mean *frequency modulation through amplitude modulation*. The reader must be aware of this fact.

There are many different modulation schemes in use for VLC, and we will now discuss the most commonly used.

i On-Off-Keying (OOK), or *amplitude shift keying*, uses keying (switching) to turn a carrier signal on and off. OOK has a low processing burden but is proven to be very sensitive to noise. Enhanced schemes like On-Off-Keying Non-Return-to-Zero have shown data rates of 1.5 Gigabits per seconds, using a Integrated Circuit with LEDs bonded into the chip [27].

ii Pulse Time Modulation (PTM) is a technique in which data is modulated in the ratio between the on and off time of the carrier signal. Pulse Position Modulation (PPM) and Pulse Width Modulation (PWM) fall into this category. The advantage of PTM is that it does not require digital-to-analog converters to generate a smooth output signal, and does not require an analog-to-digital converter either but only a comparator circuit. On the other hand, PTM requires accurate timing for both the receiver and transmitter because the ratio between on and off periods needs to be well synchronized. Another advantage of PTM is that it provides flickering and dimming support without additional techniques, which is good for use in illumination devices, but speeds are lower than with other modulation systems [13].

iii Pulse Amplitude Modulation (PAM) uses brightness levels to realize multilevel signals to encode symbols. This modulation is sensitive to external light sources as they influence the intensity, and has relative low speeds compared to other techniques. Dimming control can easily be implemented in PAM by changing the probability of the constellation points [19].

iv Frequency Shift Keying (FSK) looks a lot like On-Off-Keying, but instead of switching a carrier on and off, the system switches between two frequencies. This switch in frequency can be in terms of color i.e. a one is red and a zero is blue, or in pulse frequency [3][12]. The color technique uses a red and a blue colored LED, and toggles between them to signal symbols. Both techniques have the advantage of filtering noise by implementing band pass filters. With the pulse frequency variant this can be done using a signal processing algorithm, and with the color variant it can be done by placing color filters over the photodiodes.

v Phase Shift Keying (PSK) encodes symbols by changing the phase of the light intensity that has been shaped into a sinusoidal form. Many variants of PSK exist, depending on the number of constellation points used (Binary PSK uses 2 points, $0°$ and $180°$, Quadrature PSK uses 4 points). PSK can be used as base modulation [10], but can also be used in combination with other techniques like OFDM to improve certain weaknesses like inter-symbol interference [2].

vi Orthogonal Frequency Division Multiplexing (OFDM) is a technique that uses a large number of modulated carriers with sufficient frequency spacing so that they are orthogonal. Again, this frequency is modulated through intensity. The strongest advantage of OFDM is that it provides resistance to multipath effects, which result in long distances and high speed data transfers. Data rates beyond 3 Gigabit per seconds have been reported at short distances ($\approx$ 10cm) [1][4]. Unfortunately such high speeds demand require high-speed processing units.

vii Quadrature Amplitude Modulation (QAM) is a modulation scheme that conveys bit streams by changing the amplitudes of two or more independent carrier signals, which results in a spectral efficient scheme [9]. A form of QAM is the Carrier-less Amplitude and Phase (CAP) modulation, a promising high speed VLC modulation scheme capable of transmitting at 3.22 gigabits per second using an RGB-type led [23]. QAM and CAP can both be combined with OFDM to improve its performance even further [22].

To select a good modulation scheme, people have designed models that can be used to compare different schemes by simulating environmental conditions. The Matlab-based platform of De Lausnay *et al.* evaluates different modulation techniques and can also help to select the right modulation type [6]. To asses the performance of a modulation type, models have been developed to describe the signal properties of VLC in a free medium like air, for example the model by Komine and Nakagawa [11] which describes multipath effects. The influence of ambient light sources on light communication is modelled in the system of Virma *et al.* [18]. These models show that On-Off-Keying, although it is the most simple modulation type, has good performace for VLC.

Based on the information above, we decided to use On-Off-Keying modulation for our platform because of the good average performance. The use of this technique keeps component count low so that multiple transmitters and receivers fit on the board.

2.2.2 Receivers

While on the transmitter side the most widely used element is the LED, on the receiver side we have several options: cameras, photodiodes and phototransistors, and LEDs themselves. The common name for these types of sensors is photodetectors and they use the same fundamental principle, the *photoelectric effect*: many metals release electrons when light shines upon it[1]. Although these sensors share the same effect, they have subtle differences that result in unique characteristics.

i Cameras can detect light, its intensity and its color, using an array of semiconductor junctions or capacitors. As the miniaturization of cameras continues they can be found in embedded systems like mobile phones, allowing them to receive vlc signals without additional hardware [12]. Cameras have the advantage of focusing on the transmission source by looking at specific pixels, so it becomes possible to select a specific VLC source or to ignore a noise source. As a receiver for VLC, cameras have the disadvantage of requiring more processing to retrieve the signal from the image sensor. Secondly, a bigger disadvantage is the frame rate which is limited by the camera's (electronic or mechanic) shutter speed, which limits the signal sample rate and thus the data transfer rate [8][14]. There are high speed cameras which offer a solution to this problem [24], but currently they are too big and power hungry to use in embedded devices.

ii Photodiodes and phototransistors work based on the same principle but their assembly is different: photodiodes have only one metal junction and transistors have two. This semiconductor junction is exposed through a transparent casing, so photons can reach the junction and, when having the right frequency, excite electron-hole pairs which results in a current and voltage. Because of the big surface of the semiconductor junction, photodiodes and phototransistors are very sensitive to light. When comparing photodiodes to phototransistors, diodes are faster due to the single junction giving it a fast response time. On the other hand, phototransistors have a bigger signal gain which results in a stronger electric signal. A disadvantage of photodiodes is the dedicated circuitry needed for amplification, filtering and sampling to be able to receive a clear electronic signal. Advances in technology are capable of integrating these into a single chip which can alleviate this disadvantage.

[1] This effect led to the proof of Albert Einstein that light under some conditions can be observed discrete and quantized: the photon.

iii LEDs can be used as light sensors. An attentive reader may have noticed that photodiodes and LEDs exploit an oppose effect: while in photodiodes photons that strike the material generate a flow of electrons, in LEDs a flow of electrons release photons (light). This reversed and complementary effect can be used to make an LED both a receiver and a transmitter of VLC. The opposite is not true: a photodiode cannot be used as light source, as their type of material prevent the release of light.

    The circuit around the LED contains a driver for the transmitter part and an amplifier with analog-to-digital converter for the receiver, and using a switching component the LED can operate in one of the two modes [16]. This dual operation reduces the area needed for a transmitter and receiver module, but requires a more complicated circuitry. Another disadvantage is that LEDs are not as light sensitive as photodiodes. Additionally the use of a combined 'antenna' hinders full-duplex communication because receiving and transmitting at the same time is not possible. However, when an application only needs half-duplex communication LEDs are a very elegant solution.

Considering the various options for receivers, we opted for photodiodes because they have the higher potential to achieve high data rates, are more sensitive than other sensors, and can provide full duplex communication.

2.3 Networking for Visible Light Communication

As stated earlier, the main goal of this thesis is to provide a tool that fosters the development of *networked*-VLC, in particular, multi-hop VLC networking. The need to move VLC beyond point-to-point communication has already been spotted by the community and there are some important on-going efforts in this direction. Below we describe the state-of-the-art for networked VLC, and highlight the novelty of our platform.

The first studies in light communication networking were for infrared. These studies led to the IrDA standard allowing communication at short range between two nodes, and later on in the form of a star network. Infrared was the technology of choice because fast high intensity leds were not easily available. After the breakthrough of high speed point-to-point connections with LEDs, the massive interest in VLC was summarized in the IEEE 802.15.7 task group. This group aims at standardizing the point-to-point communication protocols, medium access schemes, modulation types and speed, and the first official release of the VLC standard was made in 2011 [21]. The 2011 standard summarizes the research in VLC networking in two categories: contention-based and contention-free channel access mechanisms. The first one uses collision avoidance and backoff algorithms, the second uses slots to access the medium. Although the IEEE standard is a major step to foster the use of VLC systems, it lacks the use of directional multihop communication.

At Disney Research Labs, a new network technology for smart toys is developed based on VLC. Using a single led for transmission and reception [5][16], they design a CSMA/CA MAC with 16 slots. When a node wants to send data, it contends for one of the 16 slots available. With this simple MAC, six nodes in a clique topology can efficiently share the medium. The solution that Disney offers is a novel way of low-complexity software for networking VLC, and just like they point out, the single led system is a good candidate for the Internet-of-Things because of its simplicity .

On a different line of work, Wang *et al.* focus on connecting VLC point-to-point technology to the existing network stack in Linux. Their VLC hardware is similar to Disney's, and use a single LED transceiver. The transceiver is connected to a Beaglebone Linux platform with a full Ethernet stack on board. The VLC NIC (Network Interface Card) driver that was developed, connects to the TCP/IP stack which makes it possible to use existing network technology with VLC, like the Internet Layer and Transport Layer [20]. As Wang *et al.* point out in their publication, there are no open-source hardware platforms available, and publications lack detailed schematics so results cannot be reproduced. Hence, open source platforms are a great contribution to the research community.

One of the properties of light is that it can be aimed at a specific point. Yang and Pandharipande connected multiple luminary devices mounted on a ceiling to each other using the light that shines on the floor. When a destination luminaire can not be reached, a luminaire in between is used in relay mode to propagate the data [26]. An important detail of this research is that they used full duplex communication by substracting the self-interference through a light propagation model [25]. This is a multi-hop network system but does not benefit from the directional properties of light.

The current state of networking systems is summarized in Table 2.3. To make it easy for the reader to compare our work with sophisticated VLC devices, the column of *Tsonev, Haas et al.* is added to indicate the speeds that are achievable with resource rich devices, although this kind of work focuses solely at the Physical Layer and provides no networking capabilities.

Overall, the main novelty of our platform is that it allows the investigation of directionality for vlc networking. The use of directionality in communications is not new, but it has not been applied to VLC nodes. The strong point of aiming data is, although trivially, that it keeps parts of the medium free. As we will describe later, this simple feature open various challenges and opportunities to be invest-

Name	Schmid *et al.* [16]	Wang *et al.* [20]	Yang *et al.* [26]	Tsonev *et al.* [17]	*This thesis*
Max. Rate	800bps	2.2Kbps	Unknown[a]	3.7Gbps	1kbps
Max. Range	∼2m	∼1m	∼6m[b]	∼5cm	∼0.5m
Addressing	Yes	Yes	Yes	No	Yes
Multihop routing	No	Existing TCP/IP	Yes	No	Yes
Directional Routing	No	No	No	No	Yes
Modulation	OOK	OOK+RLL	Manchester	OFDM	OOK
Network focus	Star topology	TCP/IP	Linear relay	No	Multi-hop
Platform	Arduino	Beaglebone	Special	None	Generic/(Arduino)
Transmitter	Single LED	Single LED	Luminaires	Single LED	Multiple LEDs
Receiver	in transmitter	in transmitter	Photodiode	Photodiode	Multiple photodiodes
Publication date	July 2014	September 2014	May 2014	April 2014	-

Table 2.1: *Comparison of state-of-the-art VLC network technology.*

[a] The authors do claim a data rate specificly, but it is expected to be in the kilobits per second range.

[b] This is the distance between two luminaires.

igated. Another important feature of our platform is that it is general, so research experiments do not need to be bounded to a specific processing platform. Our board can be connected to Rasberry Pis, Beaglebones, Arduinos, sensor nodes or laptops.

Because networking is the focus of this thesis, little attention is put into the link speed. The transmitters that can reach state-of-the-art speeds require special hardware and circuitry, which at this point are too complicated to use in an omnidirectional hardware platform with multiple embedded transmitters and receivers.

Chapter 3

Design Motivation & Requirements

Our goal is to design a VLC Platform that has multihop networking capabilities, directional coverage and that is generic. Next, we describe the importance of these three properties, and the elements that need to be considered in our design to implement them.

3.1 Multihop communication

Multi-hop communication with VLC will make it possible to create a new generation of distributed systems, such as the Smart Traffic example presented in Section 1.1. To achieve this goal we need a Network stack consisting not only of the Physical Layer, which is the current focus of VLC, but also of Data Link and Network Layers.

The first step in networking is a reliable link between two nodes, formed by the transmitter and receiver hardware. Establishing a link requires many components including modulation, synchronization and decoding. These components which for the basic *Physical Layer* of our Platform are described and in Section 4.3.1.

Since nodes can start their transmission at a random moment, a medium access control (MAC) scheme is required to avoid collisions. A MAC requires at least two basic primitives to operate: clear channel assessment, to avoid sending a packet into a busy channel. and addressing, to target the desired destination. The implementation of our MAC is presented in Section 4.3.2.

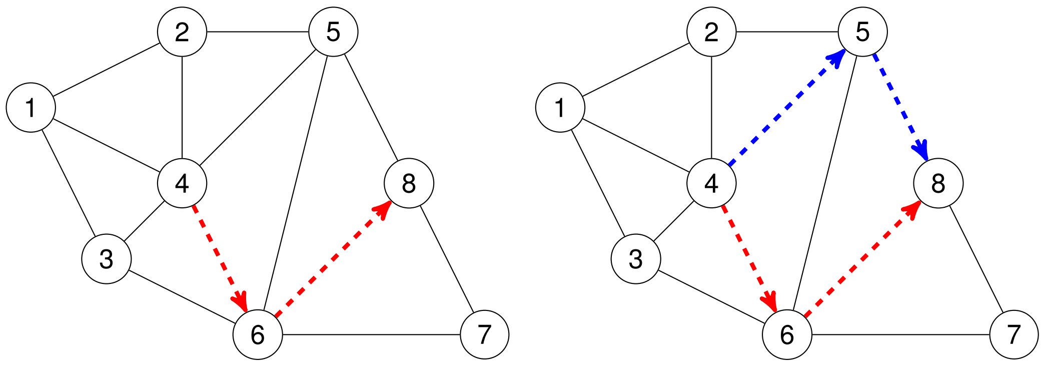
Once a basic access control mechanism is in place, the next important step for multi-hop communication is forwarding. Upon receiving a packet, a node that is an intermediate relay, needs to check its neighborhood table and forward the packet. This simple process requires a different level of addressing. The modules dedicated to forwarding tasks, referred to as the *Network Layer*, are described in Section 4.3.3.

3.2 Directionality

A network stack with directionality support has a number of advantages: i) increased in throughput due to concurrent multi-paths, ii) reduction of energy consumption, and iii) increased in throughput due to pipelining. We will first describe these three advantages, and then discuss the disadvantages of directionality.

3.2.1 Increased throughput due to concurrent multipaths

Directional communication can increase the overall network throughput by applying concurrent multipath routing. In most radio systems, a transmitter can only send information to one receiver at a time. With directional sources, two (or more) transmissions can occur concurrently, as shown in Figure 3.1. Compared to other VLC platforms which have only one LED, or many LEDs pointing on the same direction, our platform has the (potential) ability to support concurrent paths because of the multi-LED and multi-photodiode design. Achieving this concurrency is however not a simple matter, since self-interference effects have to be taken into account. In this thesis, the topic of concurrent multi-path is not covered, and it is presented in this section only to highlight the potential benefits of our platform.



Single link in omnidirectional systems (red)

Dual link in directional systems (red and green)

Figure 3.1: *Increased data throughput by utilizing multiple paths towards the destination. This is only possible with directional transceivers.*

3.2.2 Reduced power consumption

Transmitting over a limited angle, instead of transmitting in an omnidirectional manner, has the advantage of not only reducing interference but also the energy consumption (because fewer LEDs are used). Directional communication however requires a neighbor discovery process, and there is a tradeoff that needs to be considered regarding the width of the 'searching' angle. Thinner angles consume less energy per step but needs several steps to find a neighbor, while wider angles have the opposite behavior. In Section 5.3 we analyze this tradeoff.

3.2.3 Increased throughput due to pipelining

Another important advantage that directionality provides is concurrent transmission and reception of data, which we call pipelining. Similar to concurrent multipath transmissions (Section 3.2.1), pipelining also increases the throughput of the network.

Figure 3.2 depicts the basic idea of pipelining. In radio systems, most nodes have a sort of circular radiation pattern that prevents neighboring nodes from transmitting. In the case of VLC, a node can receive data via one direction and forward information concurrently via a different direction. Therefore, if a message needs to be routed through the network and the hops do not fall inside each others field-of-view, the next data packet can be transmitted right after the first one finished. Thus, faster throughputs can be obtained. We took some initial steps to test the ability of our platform to provide pipelining, but we were not able to conclude our implementation.

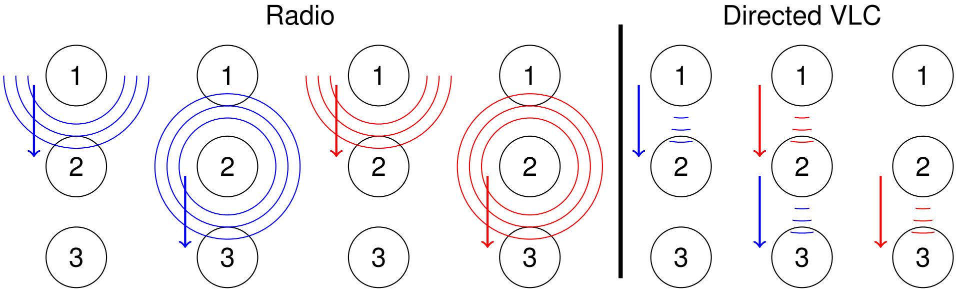


Figure 3.2: *Boardcast of two data packets (blue and red) from node 1 to node 3. Radio systems have to wait for all nodes in range to clear the channel due to the hidden terminal problem. VLC with directionality makes data pipelining possible and thus reduces transfer time.*

3.2.4 Cost of directionality

Directionality comes at a cost. Because the field-of-view of the system is limited, it misses information that could otherwise be used to gain situational awareness. In omnidirectional systems the position of neighbors is not important, but for directional systems it is required to search different sectors one at a time. Therefore a directional networking system requires position discovery and registration. To satisfy these requirements, we developed a software abstraction to control LEDs and Photodiodes based on ‘angles of communication’.

It is important to note that the neighbor discovery problem is further exacerbated if nodes are mobile, because ‘angle’ information can rapidly become stale. In this thesis, we focus only on the static case.

3.3 Genericity

Due to the proliferation of embedded systems, researchers are currently using many different platforms to investigate pervasive computing applications: sensor nodes, smartphones, Raspberry Pi, Beaglebone, Arduino, to name a few. Each one of these platforms has unique characteristics in terms of hardware (e.g. processor, ADC, GPIO pins) and software (operating system and real-time capabilities). To avoid developing a platform that is tied to a unique host, we abstracted the necessary hardware and software components into a self-contained platform that provides Physical Layer services to *any* host via a simple serial connection.

Chapter 4

Implementation

The previous chapters make clear that there is not yet an off-the-shelf solution which fully supports VLC with directed multihop networking, so a dedicated platform is build to make this possible. This platform consists of software and a circular hardware board with LEDs, photodiodes, processing and communication on board. In this chapter the details of this platform will be discussed.

Before manufacturing the final board (PCB), we tested the various components for transmission and reception in prototype circuits on a Beaglebone Black, a processing platform running Linux. The Beaglebone has a breadboard on top of it, which allows circuits to be easily changed. An example of the Beaglebone Black with one of our initial prototypes is depicted in Figure 4.1, and the final platform is shown in 4.2

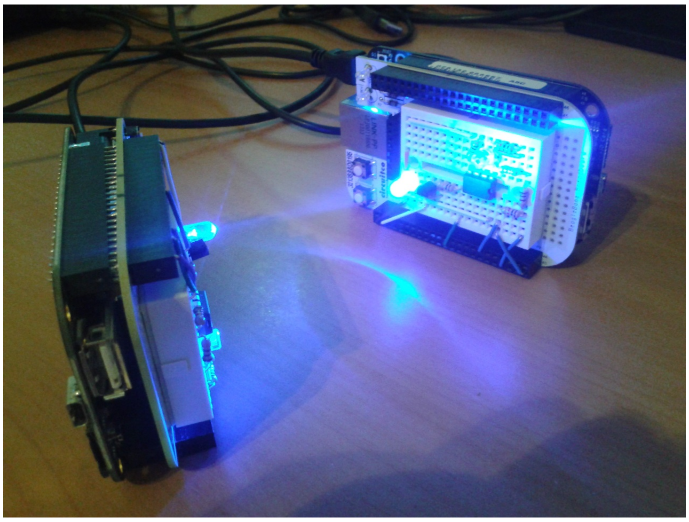


Figure 4.1: *Two Beaglebone prototype boards exchanging data using VLC.*

The VLC networking platform has three main components: i) A hardware platform with VLC transmitters and receivers, ii) Firmware that allow signal synchronization and easy access to the hardware, and iii) A network stack to accomodate *VLC multihop networking*. Compared to the state-of-the-art, the platform provides similar data rates and coverage, but adds two unique characteristics: (i) 360° coverage, which is necessary to investigate an important property of LED communication: directionality, and (ii) a flexible design, which allows our platform to be connected to many experimental boards such as Arduino, Beaglebone, Raspberry Pi and sensor nodes. The reminder of this Chapter goes first into the issue of directionality (Chapter 4.1), then into the flexible design (Chapter 4.2) and finalizes with the description of the network stack (Chapter 4.3).

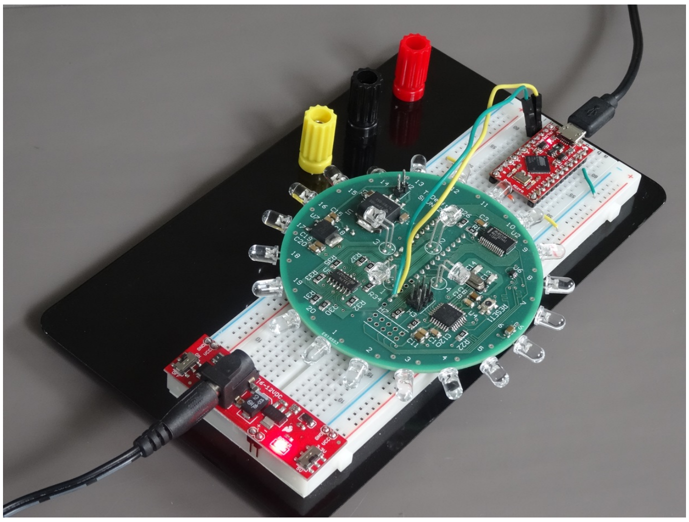


Figure 4.2: *The multihop directional research platform embedded on a breadboard.*

4.1 Directionality

As stated in the previous chapter, network communication can benefit from directional signals because they reduce interference problems and can improve the network throughput.

In this section we will first discuss how the number of LEDs for the transmitter are selected. The selection of the number of LEDs is important because it determines the coverage of the transmitter. After discussing the transmitter, we will talk about the design of the receiver, which involves many independent photodiodes to provide a good reception. The reader must know that, as discussed before, this platform is only a first step into the area. The system has not been optimized

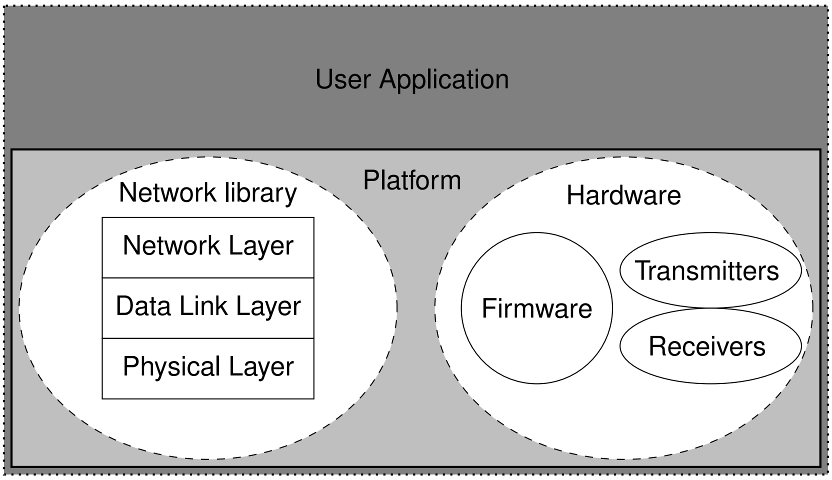


Figure 4.3: *An overview of the various platform components.*

for speed and signal reach, but the aim is rather to capture all the required elements in their basic form to make it possible to investigate them. Therefore this section ends with recommendations to further improve the hardware and bring the platform to a higher level.

4.1.1 Transmitter Coverage

In VLC, the signal is mostly modulated using intensity, so it is crucial that the transmitter can deliver a strong intensity that stands out of surrounding noise signals.

For this platform a green LED is selected with a high intensity to current ratio, the Avago HLMP-CM1A-450DD. This LED has an intensity of 47 Candela at a forward voltage of 3.2V and current of 20mA, and is packaged in a radial T-1 3/4 5mm casing.

An LED needs a current source to operate, and most microcontrollers are not capable of delivering this current for a long period, so a current driver circuit is required, whose input is controlled by the microcontroller to convert the digital signal to an electronic one. The choice of the Avago LED determines that the driver should deliver a current of 20mA, and a transistor circuit is used to accomplish this. The gate of the transistor is connected to the microcontroller, and the LED is connected to the transistor's output. At the output of the transistor a resistor is added to limit the maximum current, so the LED cannot be damaged by a large current. An example driver circuit can be found on the left side of Figure 4.4.

When putting multiple LEDs together to obtain a 360° coverage, the LEDs need to be distributed equally around the board. This distribution is based on a number

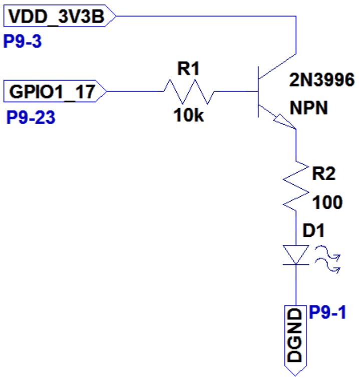


Figure 4.4: *Transmitter circuit on board the Beaglebone prototype.*

of details: i) the width of the LEDs' main and side lobes, ii) the beam angle, and iii) the radius of the board. Each LED has a spatial pattern based on the casing. In this pattern, a main lobe can be observed which has the highest intensity, and side lobes which spread a more parabolic pattern. Between the main lobe and the side lobes there is an area with low intensity.

From the Avago HLMP datasheet the shape of the intensity lobes are known, as well as the beam angle. Based on the area of normal component sizes, the radius of the board ended up being 40mm. This data was fed into a 3d simulation software to see the result of the LED placement. From the simulation it is shown that 50 centimeters of contiguous coverage is achievable when 20 LEDs (18° per LED) are placed around the board edge (Figure 4.5).

Although it does not sound like much, twenty transistors take quite a lot of space on a PCB, as every transistor also needs three wires to operate. This area can be reduced by using transistor arrays. In transistor arrays, transistors are joined in a single chip to reduce the number of components required in a platform. These transistor array chips are indicated with Q1, Q2 and Q3 in Figure 4.6.

4.1.2 Receiver Coverage

The receiver's task is to convert the analog light signal to digital values, and it does this using three modules: a light sensor (photodiode) to measure the intensity, an amplifier to strengthen the sensor output, and a analog-to-digital converter to obtain digital values.

VLC requires a sensor that is sensitive to visible light. The Osram SFH203P photodiode is chosen for this platform because wit has (i) a 90° angle of incidence with a relative sensitivity above 80% (see Figure 4.1.2), which implies that only four photodiodes are required to get a good 360° coverage, and (ii) a high

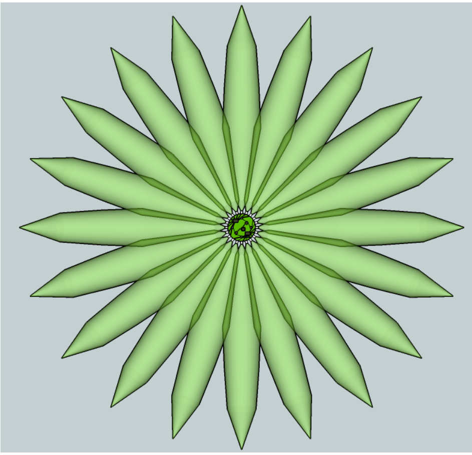


Figure 4.5: *3D Simulation of the board with 20 LEDs placed at a radius of 40mm. The overlap between two bundles ends at 50 centimeter.*

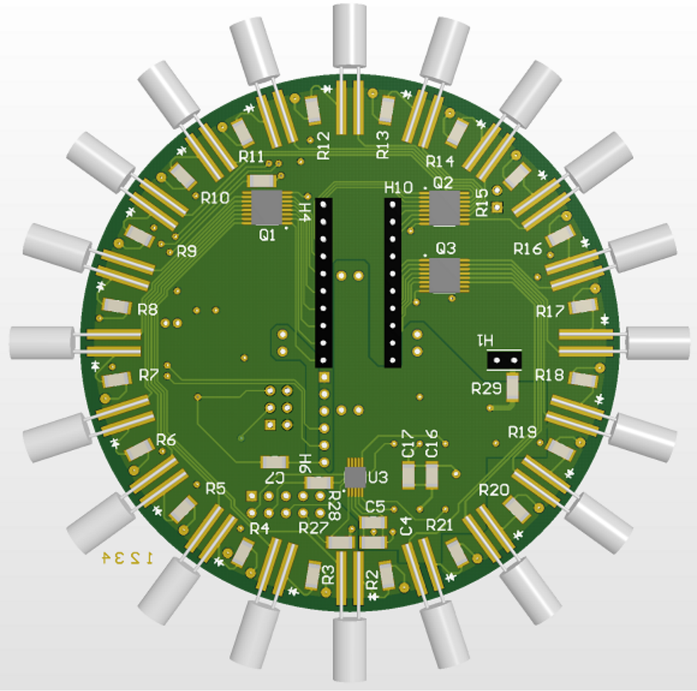


Figure 4.6: *Bottom view of the hardware used in the platform.*

sensitivity for visible light and a reduced sensitivity for infrared light, which reduces the amount of noise.

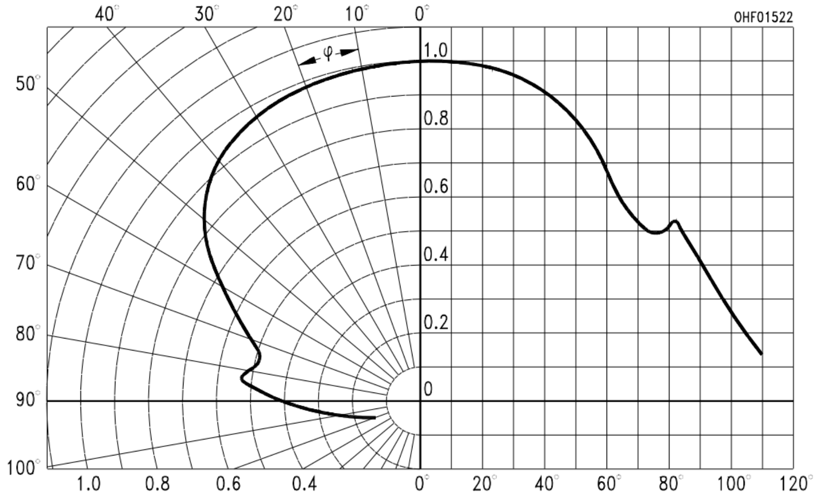


Figure 4.7: *Directional characteristics of the Osram SFH203P photodiode [15].*

For a photodiode two modes exist to represent the intensity: the photovoltaic and photoconductive mode. The photovoltaic mode has a voltage over its pins that is related to the light that reaches the junction. The photoconductive mode (or current mode) is faster than the photovoltaic mode because of the reduced capacitance, but noise effects like dark current need to be taken into account which do not occur in photovoltaic mode. In this thesis we use the photoconductive mode due to its higher speed. The circuitry for the four photodiodes is shown in Figure 4.8.

To accomplish full coverage the platform uses four photodiodes, each with its own amplifier circuit connected to a 4 channel analog-to-digital converter, the Texas Instruments ADC124S101. This 12-bits ADC has a resolution of 1.2 milli-Volts over a range of 5 Volt and can sample at 1.10^6 samples per second and is accessed through a Serial Peripheral Interface bus. .

4.1.3 Basic Achievable Rate

We now analyze the achievable rate of our design. This analysis not only elucidates the current data rate, but also exposes opportunities for improvement. There are three macro components that determine the data rate: the sampling rate of the receiver, the communication speed with the processor, and the modulation speed of the transmitter.

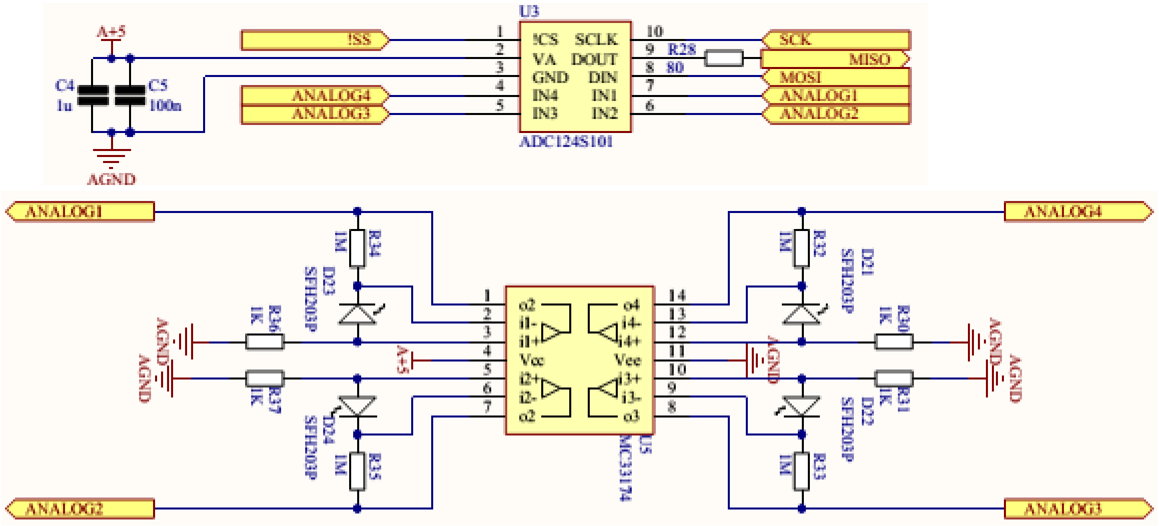


Figure 4.8: *The receiver circuit with four photodiodes (bottom circuit) and the Analog-to-Digital converter (top).*

The receiver's ADC has a sampling rate of 1MHz. Considering that we have four photodiodes, the sampling rate is reduced to 250KHz (1MHz / 4 channels) thus influencing the maximum data rate possible

The communication with the processor poses a more stringent constraint than the ADC itself. Let's say that we run the ADC at maximum speed. For the Texas Instrument ADC124S101, this is 1 megasamples per second in total. To transfer this data to the processor, a SPI bus is used. SPI is a serial bus where at each clock pulse a new data bit is put onto the data line. Each sample of the ADC is represented by 12 bits but in embedded systems this is an odd number, as the ADC SPI module can only communicate through 16 data bits. This means that a padding of 4 zeros is needed in front of the sample bits to fill this gap. Now let's look into how this 16-bit number relates to the data rate. The processor used on the platform has a maximum SPI clock speed of 8 MHz, which implies that a complete transfer of a sample (16 bits) is done at a rate of $\frac{8MHz}{16}$, which in turn results in a sample rate of 500 kilosamples per second. However, we need to sample four channels, so this sample rate is divided by four, resulting in a maximum sample speed of 125 kilosamples per second.

The main bottleneck of our design is the transmitter. The transmitter circuit (including its driving transistor) was tested using a function generator, and the output was measured using a photodiode connected to an oscilloscope. Our measurements showed that at 1.0 kilobits per second, there is no signal deformation. However, Figure 4.9 shows that for block wave signals above 10.0 KHz, a deformation starts to form around the signal edges, which indicates problems with the LED's rise time. At 1.0 MHz the block wave has entirely disappeared, the signal has become a triangular wave function as Figure 4.10 displays. The combination of the LED capacity and the driver strength limit the transmitter data rate to 10.0 KHz.

Overall, while in theory the platform can achieve 10kbps (due to the maximum rate allowed by the transmitter), in practice, clock drifts and external noise affect the efficiency of the system. We found that a robust performance could be guaranteed for a data rate of 1Kbps, which validates the data rates achieved by comparable platforms (Table 2.3).

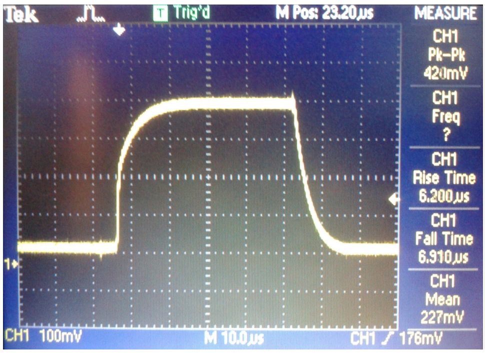


Figure 4.9: *Transmitter at 10 KHz.*

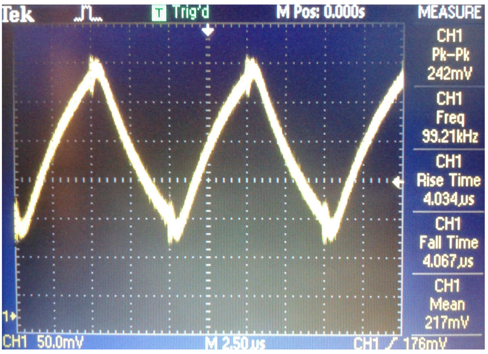


Figure 4.10: *Transmitter at 100 KHz.*

4.1.4 Improvements

Our goal achieves the basic goal of directional VLC communication, but there are still points for improvement. The data rate of the platform can be improved by using a better transistor array that is capable of driving the system faster at transitions. There are high speed LED drivers available, but this needs to be investigated to fit such drivers onto the board while keeping the board small.

Although the receiver is capable of running at a high speed it can be made faster. As stated before, the SPI bus speed dictates the sample speed, so a faster processor will increase the receiver sample rate. But this has a limit because the analog-to-digital converter has a maximum sample rate of 1 megasample per second. So when the SPI bus is clocked at 16 MHz (from the current 8 MHz) the maximum rate will be reached, resulting in a sample speed of 250 kilo samples per second for each channel.

Another opportunity for improvement is to use more sophisticated amplifiers, either to adjust dynamically the gain to amplify weak signals or to track fast changes in the signal in a more accurate manner. Furthermore, these amplifying methods can be enhanced with circuitry to filter out the background illumination so the only value at the output is the data signal. Combining these three alternatives can improve the data rate and transmission range, and facilitate the processing of the signal.

4.2 Genericity

The goal of the platform is to facilitate research. We designed a very flexible hardware interface, where physical headers are installed to expose all systems: the drivers of the LEDs, the GPIO expander, the analog-to-digital converter and the on board processor. This makes it possible to reuse the hardware at every level: the board can do all communication tasks or just be used for the basic circuitry related to transmission and reception.

4.2.1 Timing problem

Accurate and fast timing is arguably the most important aspect for modulation and demodulation. Without proper timing, the receiving system does not know when to look at the symbol values. This makes symbols indistinguishable from each other, which renders the entire signal unreadable. If cost and complexity would not be an issue, the best way to tackle this problem is certainly via hardware (FPGAs or ASICs). To reduce cost and complexity, we follow a software solution, as other low-end devices do.

When an external platform is used to control LEDs and photodiodes, e.g. BeagleBone, timing needs to be tackled in kernel-space because in user-space VLC communication would need to compete with many other tasks, making timing very unstable. Kernel solutions however are usually tied to the specific platform and OS. To remove timing problems arising from a particular setup, we use an on-board processor exclusively dedicated to timing tasks related to VLC communication. In this way, the dedicated platform can relieve any processing platform from the time crucial tasks, so even a light weight processor like the Atmel Tiny microcontrollers can use this platform.

4.2.2 Solutions for a Self contained board

To facilitate the communication between our board and the host platform, we use the *Universal Asynchronous Receiver/Transmitter* (UART) bus. UART has several advantages: it is available on most microcontrollers, it can be connected to a PC directly or through a USB-to-UART converter and it does not require complicated hardware to use it. Even if a processor does not support UART it is possible to use the UART connection through software (so called bit banging).

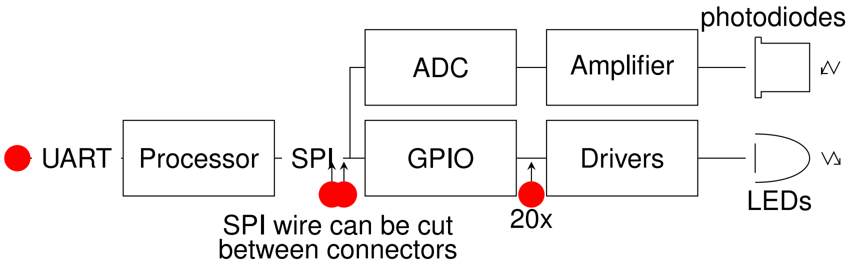


Figure 4.11: *Points where the platform can be interfaced by external systems are indicated in red.*

As stated before all submodules on board the platform can be interfaced independent of the processor on board. The 20 gate pins for the LED driver are available at a pin header, and the SPI bus that transfers data between the ADC, the GPIO expander and the on board processor is also available through a pin header. This specific SPI header can be used to control the transmitter and receiver.

To power all the circuits the VLC board has two on board power supplies, one high current supply for the general power and transmitters, and one noise reduced power supply for the receiver circuit. Each component is sufficiently decoupled using capacitors so that any rapid changes in current are dampened and do not cause voltage drops around the board. This is especially important with rapid changing current during transmissions. As an input supply a 12 volt supply is recommended.

4.2.3 Arduino compatibility

The processor used on the board is the Atmel Mega 328p, and is supported by the Arduino open-source platform so new firmware can be build using these libraries. Support of the Arduino platform is added to lower the threshold into VLC networking because Arduino is a platform for embedded systems that is easier to use in comparison with custom firmware, because it hides complicated functions inside a library.

One thing in the design that is influenced by Arduino is the UART connector, which exposes the reset pin and a power pin so a USB-to-UART converter can remotely reset the board to trigger the Arduino bootloader. This can be done from the Arduino Integrated Development Environment if the bootloader has been programmed into the processor. This is not the case by default as our solution uses a bare-metal firmware application without Arduino.

4.3 Multihop communication

We have argued that visible light communication can benefit from multihop networking because it fills a gap in the existing communication techniques. To facilitate this networking, we need to combine hardware, firmware and software into a network stack. This section will discuss the implementation of the firmware and software to accommodate VLC directional multihop networking.

In this section we start by discussing the so called Physical layer. In this part we discuss the firmware for the board, how signals are modulated and how data is transferred from and to the board. The next part, Data Link Layer, is part of the network stack. We will discuss how we can create a simple two-node communication link using the hardware and software from the Physical Layer. When this link is established we can look into multiple links, the Network Layer. In this part we will discuss how we can exchange data using multiple nodes when a direct link to the destination node is not available.

4.3.1 Physical Layer

The physical layer takes care of bit exchange and synchronization. The goal of the physical layer is to connect network processing and real world signals together. In this platform the physical layer consists of firmware in the microcontroller and the software in the networking stack. Most of the Physical Layer is implemented in firmware, except for a module that connects the hardware with the network stack through the UART connection.

Firmware: Data structure

Transmission at the Physical Layer requires defining the*Physical layer Protocol Data Unit* (PPDU). The PPDU contains all information required for a receiver to synchronize to the signal and to decode it, and not more. All other information is inside the payload section of the PPDU and is meaningless for the receiver. The PPDU is shown in Table 4.1.

Preamble			**Length field**		**Payload**
Sync		SFD	Check	Size	Data
3 Byte	4 Bits	4 Bits	1 Bit	7 Bits	1..128 Bytes

Table 4.1: *Physical Layer Protocol Data Unit.*

As shown in Table 4.1 the length field specifies the payload length. This length field is protected by a parity check bit to reduce the possibility of errors. The software decodes the length field and performs a parity check, then sets a counter that determines when the entire payload is received.

Firmware: Carrier Sensing

The first step required for packet reception is to sense the channel for a valid transmission. Our board achieves this by looking for sharp transitions in the signal strength. Let us denote x_i and x_{i+1} as two consecutive samples captured by one of the four photodiodes (each photodiode is an independent reception channel). A *phase transition* is observed if $|x_i$ - $x_{i+1}| > \Delta_{sense}$. This transition (could) denote the presence of data; for example, in Figure 4.12 a transition occurs at 3ms with a sudden jump of 100mV. The value of Δ_{sense} determines the sensitivity of the receiver, and it should be high enough to distinguish between noise and data. As an example, Figure 4.13 shows the changes in the noise floor observed during daylight when the noise floor is sampled at 10Khz. Based on the distributions observed in different scenarios and the resolution of the ADC (1.2mV), we found that Δ_{sense}=24mV is a good value to distinguish most valid signals from noise.

There is a tradeoff in this type of carrier sensing. When the difference threshold is set too high, weak incoming signals will not be detected. On the other hand, if it is set too low the system becomes susceptible to noise signals that have some repeating flickering in them. In general, it is better to set the threshold lower, because the synchronization system can later eliminate false patterns.

The transmitter uses the carrier sensing information from the receiver to do a clear channel assessment. The moment the receiver has detected a signal, it will change state. Therefore when a clear channel assessment needs to be done, the software can check the current state of the receiver and determine if the medium is occupied.

Firmware: Synchronization

The system has no dedicated hardware for synchronization, so software based synchronization and detection is used to convert incoming signals to proper digital values.

When a receiver detects a new incoming signal, it has to synchronize to the signal. To make this possible, a repeating signal is added in front of the data (*synch* pattern). The known pattern of this signal makes it possible to adjust the sample interval so the signal after the pattern can be decoded.

After sensing carrier signals, the system can start an attempt to synchronize to the incoming signal so the timing matches that of the transmitter. The way this is implemented is by using an interval counter. This counter is started at the detection of a signal edge, and counts the number of samples between two edges. For real data signals this sample count will be regular as the signal has

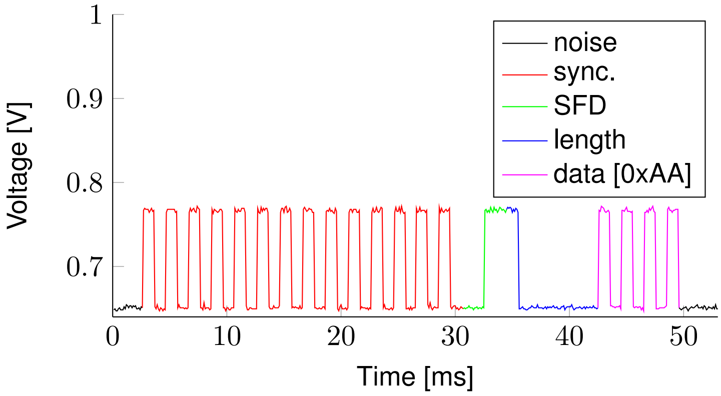


Figure 4.12: *Raw VLC signal with colored data fields.*

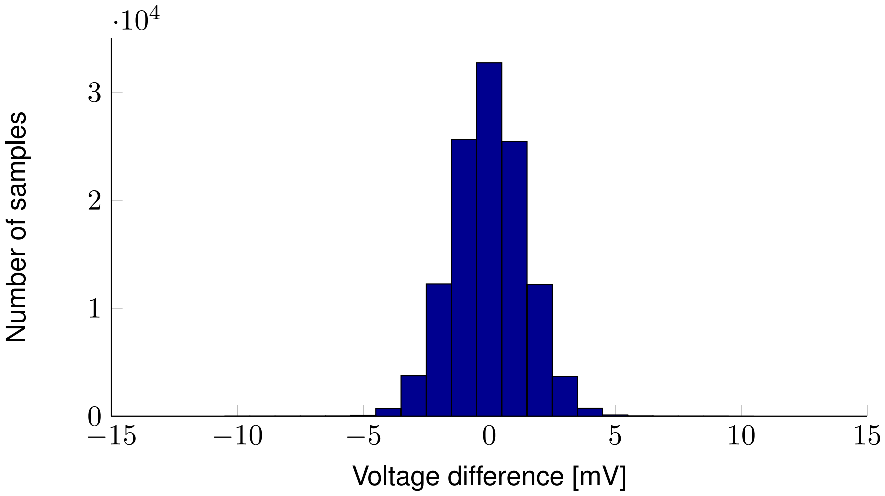


Figure 4.13: *Histogram of the difference between two consecutive samples,* Δ_{sense}.

repeating edges in the synchronization fields, in comparison with random noise. This sample interval is stored in a queue, and the mean and standard deviation is calculated calculated from a moving window, by using Equations 4.1 and 4.2 where n is the sample interval: the number of samples between two edges for a specific moment, N is the number of intervals in the queue, $\mu_{interval}$ is the mean and $\sigma_{interval}$ is the deviation of these N intervals.

$$\mu_{interval} = \frac{1}{N}\sum_{i=0}^{N} n_{samples,i} \tag{4.1}$$

$$\sigma_{interval} = \sqrt{\frac{\sum_{i=0}^{N}(n_{samples,i} - \mu_{interval})^2}{N}} \tag{4.2}$$

Based on the deviation we can determine if a signal is repetitive and if it is stable enough for synchronization. The top figure of Figure 4.14 shows an incoming data signal, with the synchronization pattern in front of the data part (which starts around 1 second). As this figure shows, the edges in the noise are much smaller than those in the data signal, so the carrier sense threshold can distinguish signal edges with ease. From 0.4 seconds onward we can see an increase in the sample interval mean from 0 to 10, and that the deviation (lower figure) remains stable at 0 from 0.5 seconds onward. A threshold checks if the sample interval deviation (lower figure) is below a certain value, at which the system decides that the incoming signal is stable enough to be used for data decoding. This threshold will be referenced to as Δ_{sync}. In the given figure, the threshold was set to $\Delta_{sync} = 1$.

One disadvantage of this type of synchronization is that the averaging window for the synchronization symbols should be small enough to eliminate rapidly the *noisy* samples that were present before the start of the synchronization. A solution is to make the averaging window smaller than half the number of synchronization symbols. Making the window too small will result in a false synchronization when the background noise contains some periodicity.

```

// Check if the current sample is part of an edge.
  if (is_edge(currentsample, previoussample, EDGETRESHOLD)){
    // It is a transition.
    // Add the time between last edge and now to the interval array so that it is included in the calculation.
    RingBuffer_Insert(&intervalbuffer, samplecounter++);
    symbolcounter++;

    // Calculate the standard deviation.
    uint16_t mu  =      mean(&intervalbuffer);

```

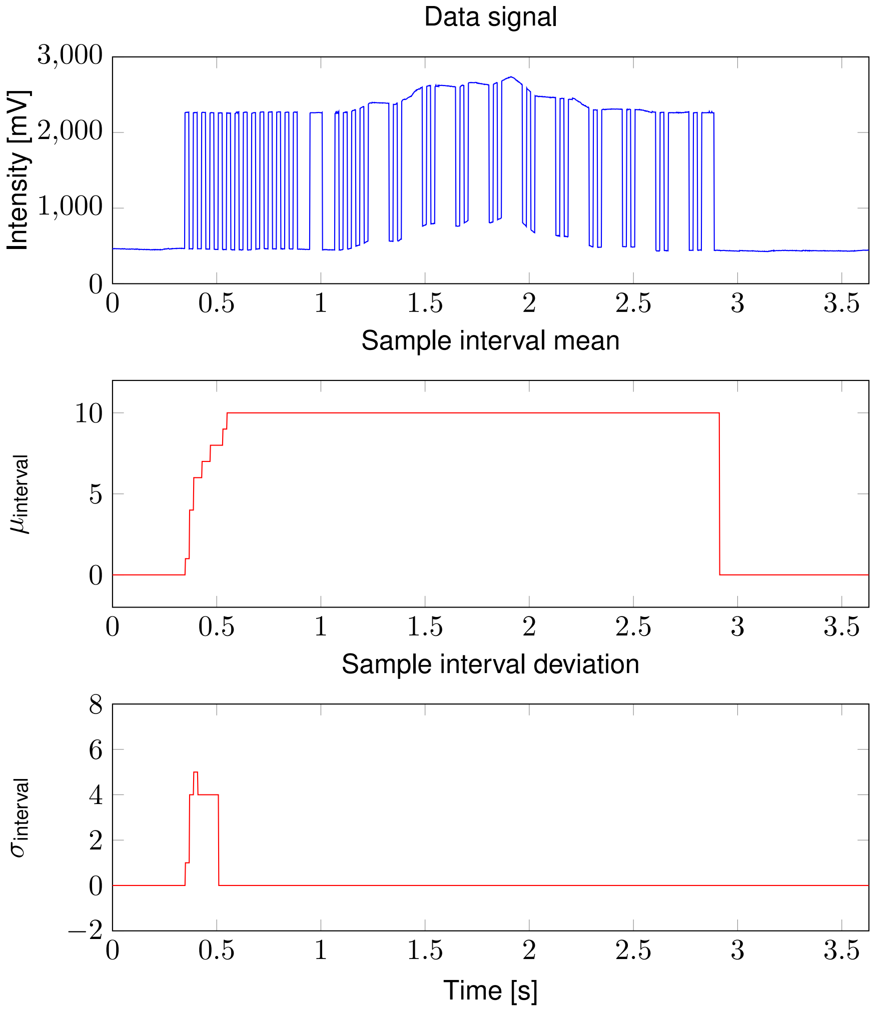


Figure 4.14: *An incoming data signal with the corresponding mean and deviation of the number of samples per symbol, in this case 10 samples per symbol.*

```
    double sigma = deviation(&intervalbuffer, mu);

    // Check if we have received all of the sync symbols.
    if (symbolcounter == SYNCSYMBOLTHRESHOLD) {
      ..Go to locked state

    }

    // Check if the interval does not vary too much.
    if (sigma < SIGMATHRESHOLD) {
      // Variation falls within limits, so we have a stable repeating signal.

      // Check if we have a rising edge.
      if (previoussample < currentsample) {
        // Previous samples collected in the sample_array were low symbols.
        thresholdlow = array_mean(samplearray, samplecounter);
      }
      else {
        // Previous samples collected in the sample_array were high symbols.
        thresholdhigh = array_mean(samplearray, samplecounter);
      }
      // Update the threshold.
      threshold = (thresholdhigh + thresholdlow) / 2;
    }
    else {
      // Variation is too large.
      // Reset the sync symbol counter.
      symbolcounter = SYMBOL_COUNTER_INIT;
    }

    // Reset the sample counter because of edge transition between samples.
    samplecounter = SAMPLE_COUNTER_INIT;
  }
  else {
    // It isn't a transition.

  }

  previoussample = currentsample;

  // Add the sample to the array.
  samplearray[samplecounter] = currentsample;

  // Increase the sample counter (number of samples received).
  samplecounter++;
```

Listing 4.1: *Simplified synchronization code.*

Firmware: Adaptive Symbol Thresholding

After the synchronization process is concluded, the board needs to start decoding each incoming bit. To achieve this, we use an adaptive threshold Δ_{decode} to discern 1s and 0s. If the average signal strength of s continuous samples is above Δ_{decode}, the received symbol is deemed to be a 1, else it is deemed to be a 0. Considering that the intensity of the background light(s) can change while receiving a packet, we use a dynamic threshold Δ_{decode} to decode bits.

Let us denote μ_0^i and μ_0^{i-1} as the average signal strength of the last two symbols deemed to be 0, similarly let us denote μ_1^i and μ_1^{i-1} for the last two symbols deemed to be 1. Then Δ_{decode} is defined as:

$$\Delta_{decode}(i) = \frac{(\mu_1^{i-1} + \mu_0^{i-1}) + (\mu_1^i + \mu_0^i)}{4} \tag{4.3}$$

Figure 4.15 showcases the importance of having an *adaptive* threshold to cope with the dynamics in the environment. Around 50ms, the background light changes its intensity, increasing the perceived voltage at the photodiode, but the adaptive threshold is able to follow the trend.

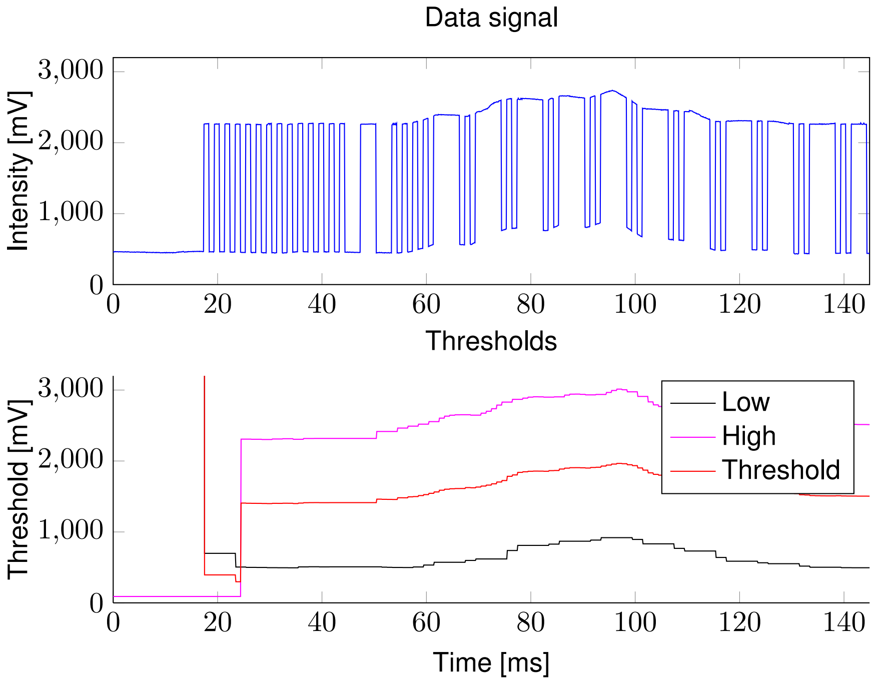


Figure 4.15: *A data signal with varying intensity and the adaptive thresholds below.*

Stack interface

The stack software runs on a different platform (host) than the firmware of the board. Therefore a link is required between the hardware and the network stack. This connection consists of messages send over the UART link in two formats: Requests and Answers. When a request is send, it is always replied with an Answer message to confirm or deny the request. These requests can be found in Appendix B.

As stated before, an Universal Asynchronous Receiver/Transmitter bus is used to communicate with the hardware. From the stack side (host platfrom), this connection is abstracted by the *IConnector* interface. This interface defines abstract functions like *read()* and *write()*, so the underlying communication with the driver becomes abstract and thus invisible to the stack. In other words, the stack can use the connection without bothering how the *Connector* is implemented. Currently the *Connector* class implementing the *IConnector* interface is build for Windows, but can easily be converted to Linux with little adjustments.

4.3.2 Data Link Layer

The Data Link Layer is responsible for the link between nodes, and provides services to set up a high quality link with a specific node. For this, it uses medium access control to avoid collisions and addressing to select a neighboring node. Because we work with directionality, the data link layer needs to track neighbors so it can select the right angle for transmission.

Frame construction

The functionality of the data link layer requires additional data appended to the transmission message to operate, in the form of two data fields. The first data field contains the address of the destination neighbor, the second field contains the address of the sending neighbor. The address fields are 2 bytes long, so a form of network partitioning can take place using the first byte as network designator and the second as node designator.

Length	Destination	Source	Payload
1 Byte	2 Byte	2 Byte	1..124 Bytes

Table 4.2: *Datalink Layer Protocol Data Unit.*

Medium Access Control

The Medium Access Control mechanism has two functions: i) control the transmitter to avoid collisions, and ii) filter incoming data.

For shared networks without a central control mechanism, collision avoidance is required. When multiple nodes are transmitting at the same time in the same medium space, the performance can decrease severely. Two forms of collision mechanisms are available, Collision Avoidance and Collision Detection, but for our platform only Collision Avoidance is possible as the used modulation and encoding scheme does not result in a distinctive signal state when a collision occurs.

The used Collision Avoidance system activates when the user wants to transmit data. It then requests medium information from the hardware. Transmitters can start sending at any random moment, so active channel assessment methods are required. The hardware measures the noise floor and checks if the receiver has locked to a synchronized signal. Based on thresholds the hardware then decides to return *MEDIUM_IDLE* or *MEDIUM_BUSY*. If the medium is idle the transmission can begin and the MAC system will instruct the transmitter to start, or if busy the system waits until it is free again. This wait time increases with every try following Equation 4.4, until the medium becomes free or a retry counter reaches its limit.

$$T_{backoff}(c) = \sum_{i=0}^{c} T_{delay}.i \tag{4.4}$$

The Medium Access Control mechanism supports directionality. From the neighbor table we discuss in section 4.3.2 the MAC can determine the transmission direction, and it uses the receiver in that direction to determine the medium state.

Incoming data needs to be filtered to remove data not intended for a node. This filtering happens through addressing. Each node has an unique address to listen to, and there is one broadcast address to which every nodes listens. The frame described in Frame Construction contains enough information to apply this filtering.

Neighbor discovery

The board uses directional transmitters and receivers, therefore it must know the position of its neighbors to be able to communication with them. This neighbor discovery process has multiple purposes, i) to see which boards are within reach, ii) obtain information about what their position is, iii) to obtain the link quality to a neighbor.

The neighbor discovery procedure uses an active and passive form of discovery. The passive neighbor discovery system uses intercepted transmissions to discover new nodes. The active discovery algorithm is a request based system with the request transmitted in a certain direction or as broadcast. If a board then receives the discovery request it replies back with a confirmation. Because this

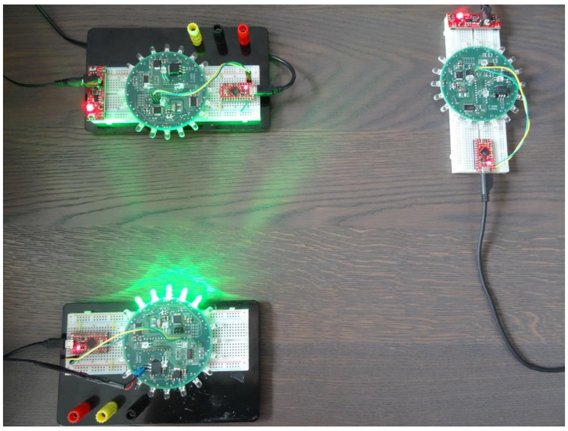


Figure 4.16: *Photo of the setup used to test the neighbor discovery algorithm.*

confirmation is transmitted in a Data Link Protocol Data Unit, the reply already contains the address fields. The discovery algorithm can be compared with the *ping* command on a computer.

Knowledge about the position of the neighbor is obtained from the combination of the LED used for transmission and the photodiode active during reception. If the discovery request is transmitted using all the LEDs at the same time (a broadcast transmission) the position will be known from the direction the reply is received. Because this depends on the receiver angle, the accuracy will be roughly $90°$. On the other hand if a single LED is used and we get a reply back, the neighbor is within the transmission angle of the LED which is $18°$ in our case.

However, this precision comes at a cost. There is a tradeoff between precision, discovery time and energy consumption. Gaining the position of a neighbor with a high precision takes more time than a rough position, because every single LED must be used after the other to transmit the discovery request. This takes much more time than a broadcast, but a broadcast will result in a $90°$ precision and the single LED approach $18°$. So a broadcast can speed up the discovery time, but comes at the cost of energy. This tradeoff is further evaluated in Chapter 5.3.

Information gained from the neighbor discovery is stored in a table together with two angles that determine the field of transmission to reach the neighbor. This table is shared with the network layer which uses it for routing, this is discussed in the next section.

The platform currently uses four zones for network discovery corresponding to the four receiving angles of the photodiodes. This is done so the discovery time is not too long and the precision is the same as for passive neighbor discovery.

```

// Begin of a pseudo code directional discovery algorithm.
#define stepsize 4
#define anglestep  = 360 / stepsize
#define stopafternewdiscovery 1

// Repeat the discovery multiple times until we find something.
for(int i = 0; i < stepsize; i++) {

  Angle startangle = anglestep * i;
  Angle endangle   = startangle + anglestep;

  // Send the discovery request.
  sendDiscoveryRequest(startangle, endangle);

  // Have we found something?
  Message msg = Datalinklayer.read();

  ..  //process the message.

  // Get the address.
  Neighbor newneighbor = parseHopSource(msg);

  // Do we already know this neighbor?
  if(!isRegistered(newneighbor)) {
    // Add it.
    registerNeighbor(newneighbor, startangle, endangle);

    // Do we stop the discovery or continue?
    if(stopafternewdiscovery) {
      // We stop discovering.
      break;
    }
  }
}
```

Figure 4.17: *A neighbor discovery algorithm in pseudo C code.*

However, it can easily be changed by adjusting a predefined step value.

4.3.3 Network Layer

The task of communicating through multiple hops lies with the Network Layer. To achieve this, the Network Layer needs to keep track of nodes in the network, and how to reach them via its direct neighbors.

Frame construction

As we discussed before, we already have a data unit containing local source and destination fields which is called the *Datalink Protocol Data Unit*. However, these fields are meant for the connection between two boards so the message is passed to the right neighbor. To exchange data over multiple hops, we need two additional fields: the final destination field and the original source field.

A packet needs to contain information about its final destination in the network, so it can be passed on until it has reached this destination. Next to this destination field this this final node also needs to know which node was the source of the data, so it can reply back to it if needed. To indicate the difference in addressing at different layers, for the Data Link Layer we use the terms *hop source* and *hop destination* fields, for the network layer we use the terms *source* and *destination*.

The new fields are added to the *Payload* section of the *Datalink Protocol Data Unit*, resulting in a new message which we will call the *Network Protocol Data Unit* or NPDU. The complete NPDU message is shown in Table 4.3, with the fields of the *Datalink Protocol Data Unit* added. The NDPU contains one other field, the *service type* field. The Network Layer offers a number of services which this field identifies. After receiving a message, the Network Layer decodes the service type field from the payload and then checks what to do with the payload.

$$\eta_{NPDU,max} = \frac{l_{payload,max}}{\sum l_{fields}} = \frac{119}{129} = 92.2\% \tag{4.5}$$

$$\eta_{NPDU,min} = \frac{l_{payload,min}}{\sum l_{fields}} = \frac{1}{11} = 9.1\% \tag{4.6}$$

When adding header data like the addressing fields, the overhead of a message increases. The header fields do not add information for the user application but are required to pass data correctly. Therefore we can express the efficiency of the added header data versus the payload size. As Equation 4.5 shows, for a small payload the system is not very efficient as only 9.0% is used to contain the data.

When the entire payload is added, the efficiency rises to 92.2% with only 7.8% overhead.

Length	**Hop Destination**	**Hop Source**	**Destination**	**Source**	**Service type**	**Payload**
1 Byte	2 Byte	2 Byte	2 Byte	2 Byte	1 Byte	1..119 Bytes

Table 4.3: *Network Layer Protocol Data Unit.*

Forwarding mechanism

At the Network Layer, packets that are not destined to the node itself may need to be forwarded. For this, the Network Layer module has a function TimerTick that needs to be triggered regularly so that the system can process these packets. The software is designed in such a way that this trigger is not implemented yet. The designer can use various techniques to trigger the function, like timer interrupts, task scheduling in a RTOS or calling the function in the main loop. The TimerTick function places data that is intended for the own platform in a queue, which is read every time the Read function is called.

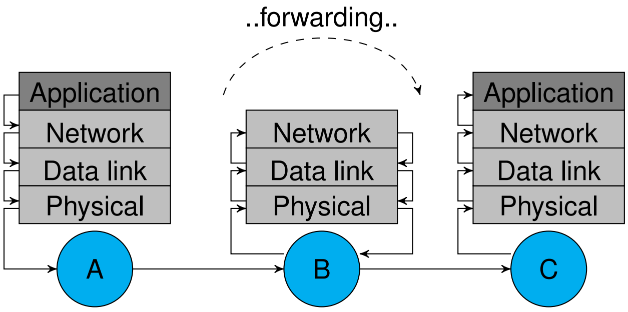


Figure 4.18: *Node A communicates with node C through forwarding by node B.*

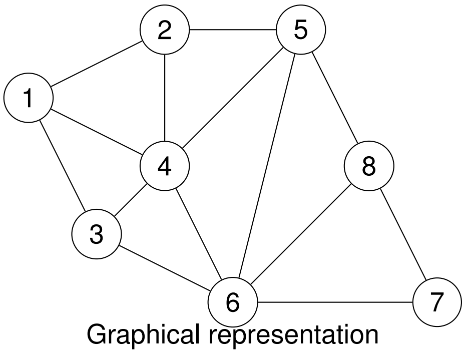
End-to-End Routing

We did not have time to implement end-to-end routing, but we made some progress. End-to-end routing requires three main components, (i) the data structures to maintain information about the network, (ii) routing algorithm that uses the previous structures to find the shortest paths, and (iii) a dissemination protocols that enables gathering the necessary for the data structures. We made some initial progress on the first two points.

Data Structures. Nodes keep track of the network's state in two lists. The first list contains the direct neighbors of the platform, plus the addresses and angles. The second list represents the detected part of the network.

The way this information is stored is shown in Figure 4.19. Once the information is disseminated (step iii above and not done in this thesis), an entry point is created for each newly discovered node. This entry point contains the id of the node and the list of its neighbors. As more and more information is disseminated, all nodes start having a more complete view of the graph.

Routing algorithm. Given that data structures mentioned below, the Dijkstra algorithm is run to discover the shortest path to a node. In our case, given that we don't have a method to discover the network's state automatically, the information needs to be inserted manually before running the Dijkstra algorithm.



Graphical representation

1	→	2	3	4		
2	→	1	4	5		
3	→	1	4	6		
4	→	1	2	3	5	6
5	→	2	4	6	8	
6	→	3	4	5	7	8
7	→	6	8			
8	→	5	6	7		

Table representation

Figure 4.19: *Graph representation of a random node network with the corresponding link table.*

Chapter 5

Evaluation

In this chapter we discuss the four experiments we ran with the directional VLC boards. The first experiment shows the LED intensity at various distances to give an indication of the transmission range. The second experiment analyzes the Signal-to-Noise ratio (SNR) of links when different numbers of LEDs are used. The third experiment evaluates the energy consumption of the neighbor discovery process. And the last experiment showcases a preliminary evaluation of multi-hop forwarding.

5.1 Hardware performance and coverage

Introduction

A measurement of the light intensity is an important performance metric because it shows the relation of the transmitted signal strength over a certain distance. These measurements are of use when developing a receiver, because it allows us to fine tuning the circuitry according to the observed signal strengths.

Methodology

To measure the light intensity a calibrated fluxmeter is used which measures intensity in lumen [lm]. Lumen is a derived SI unit, it measures candela across a solid angle of one steradian [1 cd.sr]. Therefore lumen expresses light intensity at a specific point. Fluxmeters contain a light sensor that measures lumen using a specially designed lens and casing. The measurement was done in a dark room with a constant background light intensity around 5.0±0.8 lumen. The board was placed on a platform without reflecting surfaces around it.

The used fluxmeter has a recording function that can log data at specific points. When the sensor is placed in a given position, a button can be pressed to trigger the meter. It then takes a measurement until the signal is stable. This measurement is done at predetermined points, 1 centimeter from each other, and then logged. All the LEDs are switched on so a full coverage can be measured.

Results

The raw data points can be found in Appendix A. Using Matlab this data is interpolated which results in Figure 5.2. A peak was measured around 18000 lumen, but the difference between the lowest and highest value was so large it was impossible to show in a plot. Because of this, in the plot of Figures 5.2, 5.3 and 5.4 the data is clipped to 1000 lumen.

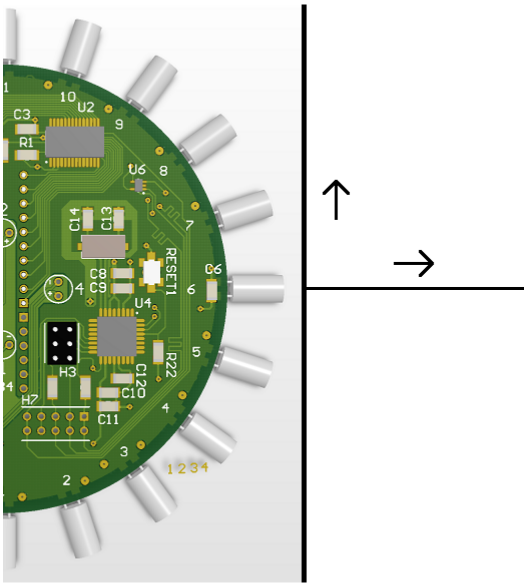


Figure 5.1: *Measurement grid indicating positive distances*

Figure 5.2 shows the data from a top view. At the point 0,0 the edge of LED 6 is positioned. As can be seen in the plot, the lobes of the other LEDs can be spotted easily. At a distance lower than 20 centimeter the intensity is above 1000 lumen which indicates a strong separation between background and signal. At a distance of 30 centimeters the intensity drops to 500 lumen. At 60 centimeters the difference between the background and signal is 220 lumen, which is not very strong. When comparing this data to the simulated data used in the design, the results overlap pretty good. In the simulation it was expected to have overlap until 30 centimer, which is also shown in Figure 5.2.

Figure 5.4 shows a cross section from the center of the board. In the figure, the left curve shows the intensity drop from the 6th LED, which follows the typical quadratic loss expected from electromagnetic signals (Equation 5.1).

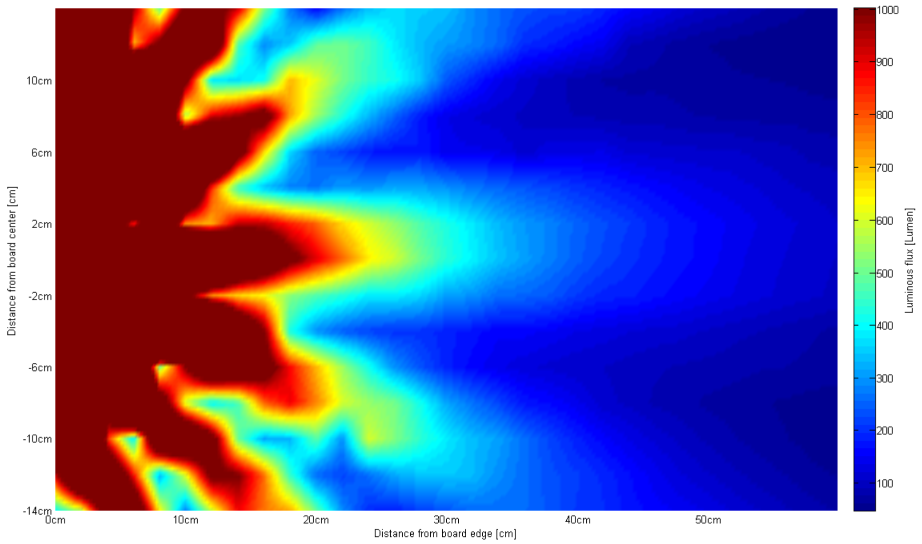


Figure 5.2: Plot of measured intensity around the board, measured in lumen (candela per arcsecond).

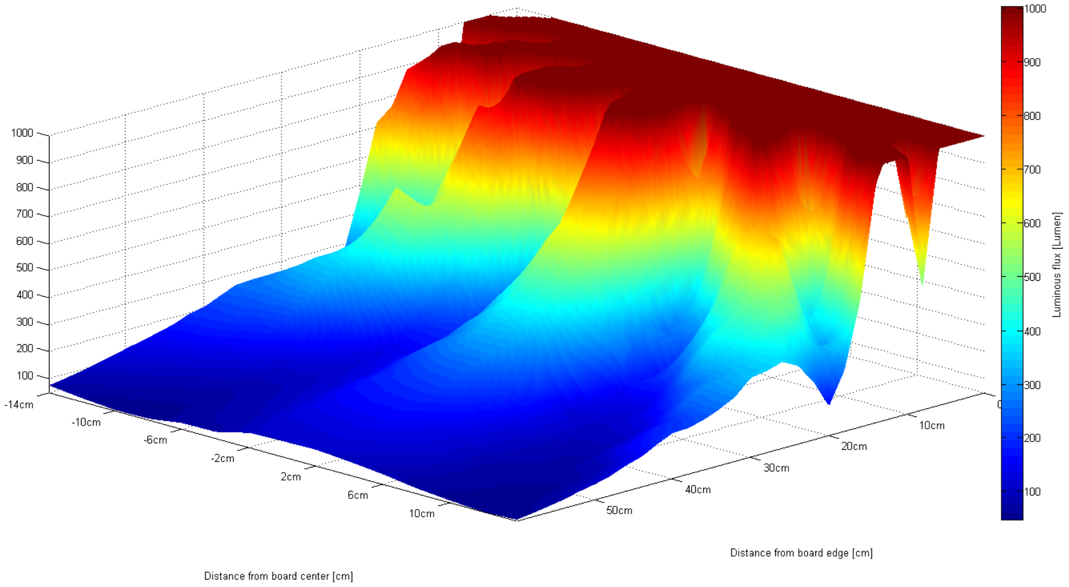


Figure 5.3: *Plot of measured intensity around the board, measured in lumen (candela per arcsecond).*

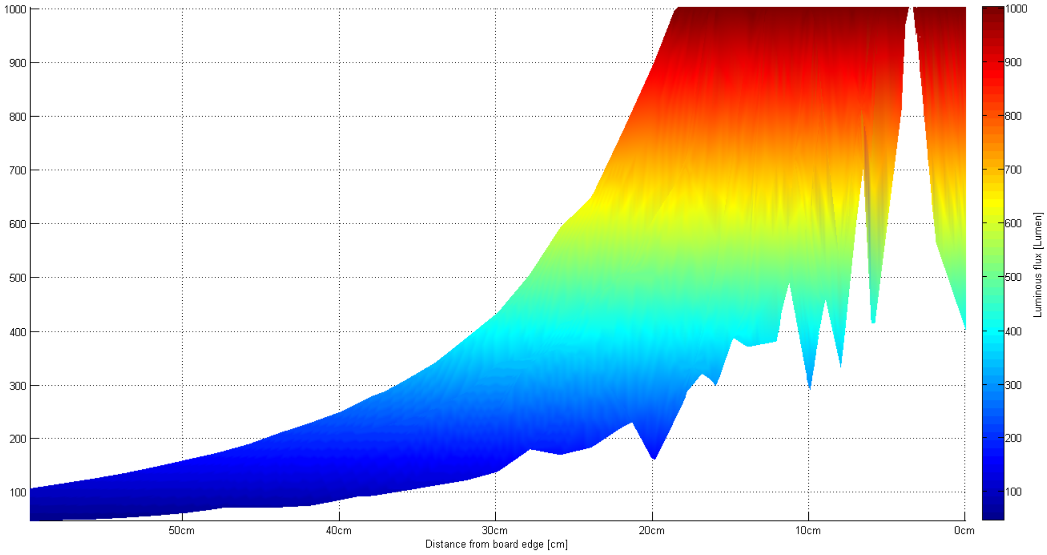


Figure 5.4: *Size view of the main lobe (center of the board) displaying the quadratic drop of intensity, measured in lumen (candela per arcsecond).*

$$Intensity \propto \frac{1}{distance^2} \tag{5.1}$$

5.2 Analysis of the beamforming technique

Introduction

An important metric for any wireless link is the analysis of the signal-to-noise ratio (SNR) as a function of the transmitter-receiver distance. For our board, this analysis is of particular interest because, as we will see, the SNR depends on the number of LEDs used during transmissions.

Methodology

For these measurements, we use the same room and setup as in the first experiment. On each measurement, a data packet is sent from board A to board B, and the received packet is checked for errors. This measurement is repeated at increasing distances until the signals are lost, either by bit errors or by loss of synchronization.

Just like radio, VLC suffers from multipath effects like interference and reflection. Because the platform consists of multiple transmitters, it is important to take these effects into account. Therefore we repeat the distance versus quality measurement for instances where various number of LEDs are *on* simultaneously.

A difficulty in this experiment is that the raw signal samples are not available directly for evaluation because the board decodes data signals internally. Therefore the firmware is modified to include the low and high threshold values μ_1 and μ_0 at the end of each reception. These thresholds allow us to calculate the SNR. A binary zero has the same energy as when the transmitting LED is off, so μ_0 is a measure of the background noise. Both μs are expressed in Volt.

The general expression of the signal-to-noise ratio in terms of power is:

$$SNR = \frac{P_{signal}}{P_{noise}} \tag{5.2}$$

Using Ohm's Law we can describe the ratio in terms of voltages:

$$SNR = \frac{U^2_{signal,rms}}{U^2_{noise,rms}} \tag{5.3}$$

This equation helps in expressing the signal-to-noise ratio in terms of high and low threshold voltages.

To evaluate the SNR, we need to calculate the RMS values of the mean threshold values μ_1 and μ_0. The OOK block switches between an energized and a zero state as shown in Figure 5.5. Assuming that on average the occurrence of ones and zeroes are distributed equally, and denoting T as the period of the signal and D as the ratio between on and off periods, we can mathematically describe the signal-to-noise ratio as follows:

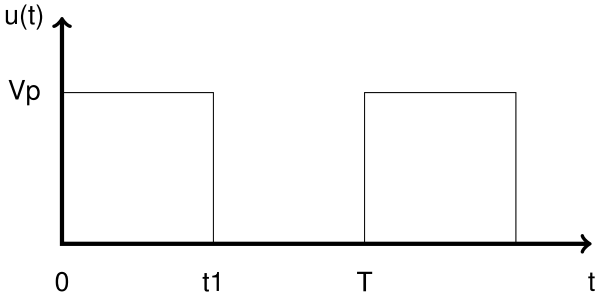


Figure 5.5: *Block wave signal used in the On Off Keying modulation.*

$$U_{rms} = \sqrt{\frac{1}{T}\int_0^T u(t)^2.dt} \tag{5.4}$$

$$U_{rms}^2 = \frac{1}{T}\int_0^{t_1} V_p^2.dt = \frac{1}{T}.t_1$$

$$U_{rms} = V_p\sqrt{\frac{t1}{T}} = V_p\sqrt{D} = V_p\sqrt{\frac{1}{2}}$$

$$U_{rms} = \frac{V_p}{\sqrt{2}} \tag{5.5}$$

The U_{noise} is the lower value of the signal and originates from the background illuminance, which we assume constant during the short transmission time. Therefore $U_{noise,rms} = U_{noise} = V_n$ for short periods. Combining Equation 5.3 and Equation 5.5 results in the Signal-to-noise ratio given in Equation 5.6.

$$\begin{aligned} SNR &= \frac{U_{signal,rms}^2}{U_{noise,rms}^2} \\ &= \frac{V_p^2}{2Vn^2} \end{aligned} \tag{5.6}$$

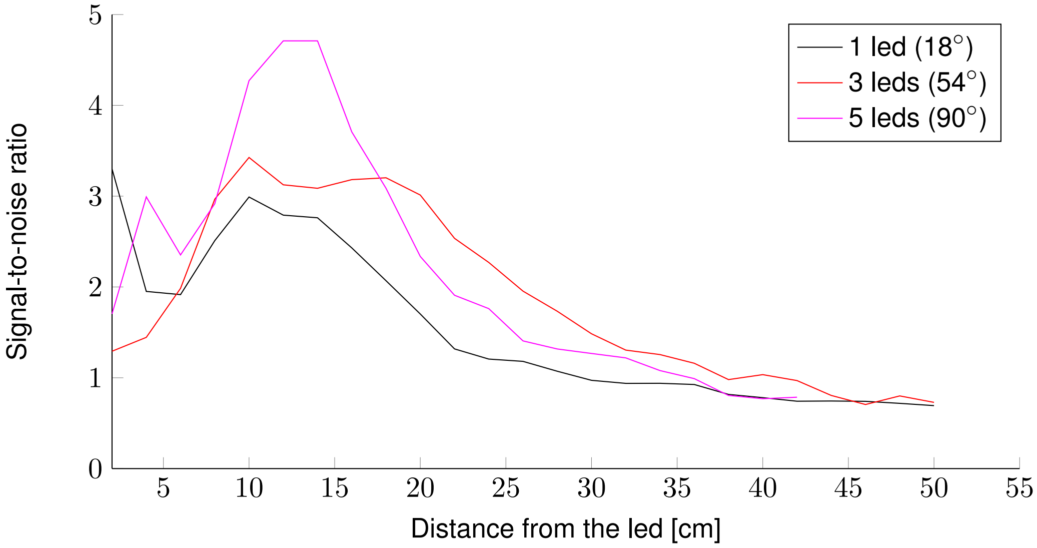


Figure 5.6: *Signal-to-noise ratio for various number of LEDs.*

Results

Figure 5.6 has two trends that are important to explain. First, while beyond a distance of 15cm the expected decay of electromagnetic signals is observed;

between 0 and 15cm, the signal strength is lower than expected. This is due to the saturation at the operational amplifiers, which are calibrated to amplify weak signals. The second, and more important, observation is that adding more LEDs to the same communication channel can have detrimental effects. We observe that when three LEDs are used instead of one, we obtain a higher SNR, but when we further increase the number of active LEDs to five, the SNR decays. Packets are still received, but there is no real gain in SNR. We hyphotesize that this occurs due to destructive interference. At close distances LEDs pointing at different radial directions are likely to increase the SNR due to constructive interference, but as the distance increases, the signals get progressively out of synch causing destructive interference. Contrary to most VLC platforms which consist of a single LED or LEDs facing the same direction, the 360° coverage of our board allow us to investigate the problem of directionality.

5.3 Neighbor discovery

5.3.1 Introduction

The hardware is capable of transmitting data in a directional manner, but first we need to know who is within reach of communication, and for that a neighbor discovery process is necessary. Neighbors around the board are discovered using a request/reply system, just like the *ping* command works on a pc. There are multiple LED combinations that can be used in neighbor discovery: all LEDs turned on at once, one LED at a time, or any combination in between these two extremes. This discovery process takes energy and time, and in this section we analyze it.

5.3.2 Methodology

When placing one neighbor at a random position, the probability that we find this neighbor when sending from one single LED is $p_{discovery} = \frac{1}{20}$, because there are 20 positions where the neighbor can be located. When using a different number of LEDs ℓ, this probability will change to $p_{discovery} = \frac{\ell}{20}$. Denoting L to indicate the total number of LEDs in the board and l as the number of LEDs in one discovery step, the expected number of steps until discovery is given by Equation 5.7.

To measure the power consumption at each step, we transmit a discovery request and log the voltage and current at the board's power supply. The start of the transmission is clearly indicated by the increase in current consumption, but the end of the transmission is difficult to determine because the message ends with transmitting a 0x00. Therefore a current pulse is added to indicate the end of the transmission, this current pulse is activated by enabling a specific LED from source code. The measurements are repeated multiple times so that an average current and voltage value can be calculated, which then results in average power. For this measurement a calibrated lab multimeter from Hewlett Packard is used which is capable of data logging to a usb memory stick.

$$\begin{aligned} E[T] &= \sum_{i}^{\infty} t_i.p_i \quad \textit{definition} \\ &= \sum_{i=1}^{steps} i.p_{discovery} \\ &= \sum_{i=1}^{\frac{L}{l}} i.\frac{1}{\frac{L}{l}} \\ &= \frac{\frac{L}{l}(\frac{L}{l}+1)}{2}.\frac{l}{L} \\ &= \frac{(\frac{L}{l}+1)}{2} \end{aligned} \tag{5.7}$$

5.3.3 Results

The power measurement shows a clear linear line with barely any deviation (Figure 5.7), indicating that the board's power supply is strong enough to provide a stable current and voltage even under heavy load. When no LEDs are used we get the idle power consumption of the microcontroller, ADC and GPIO, which is on average 652mW.

Figure 5.8 depicts the number of steps required to discover a neighbor in one of the 20 available sectors. The sectors are numbered from 0 to 19. We observe that, as expected, using all 20 LEDs requires a single step, while using a single LED may require up to 20 steps.

We will now present results that combine the number of steps taken with the power taken at each step. While it may be tempting to say that discovery using one LED at a time uses less power than using all LEDs at once, we will show that this is not the case. In fact the optimal (minimal) energy consumption occurs for a value in between these two extremes.

Figure 5.9 shows that, in expectation, the discovery of a randomly positioned neighbor costs less energy when using all the LEDs at the same time then using a single LED. This is very unfortunate, because a discovery with a single LED will give the highest precision (regarding the radial location of the neighbor) and would also minimize interference. This figure also shows that the expected energy consumption of other combinations (5 or 10 LEDs) may be lower than the 20-LED configuration.

We will now derive a mathematical expression for the expected energy consumption of the discovery process when ℓ LEDs are used. Denoting C_o as the default energy cost of turning on the platform and C_ℓ as the cost of turning one

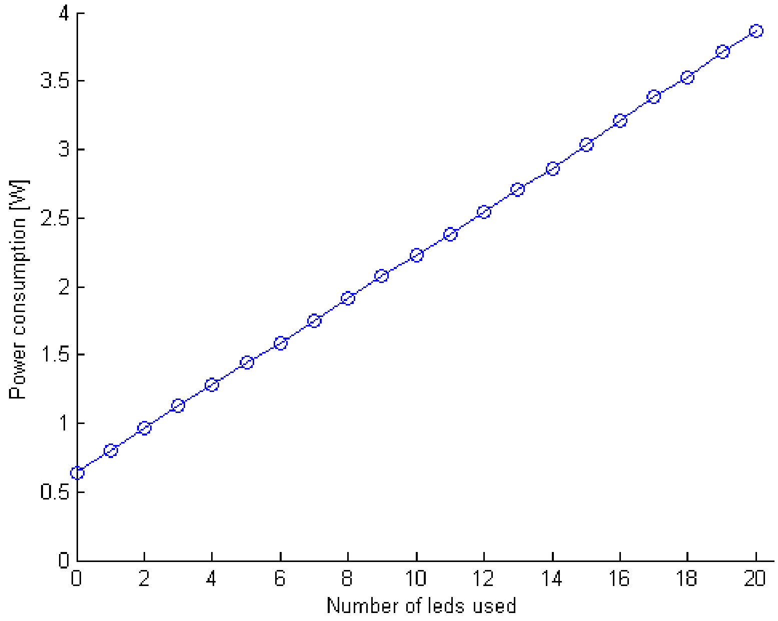


Figure 5.7: *Power consumption during transmission of neighbor discovery requests using different number of LEDs.*

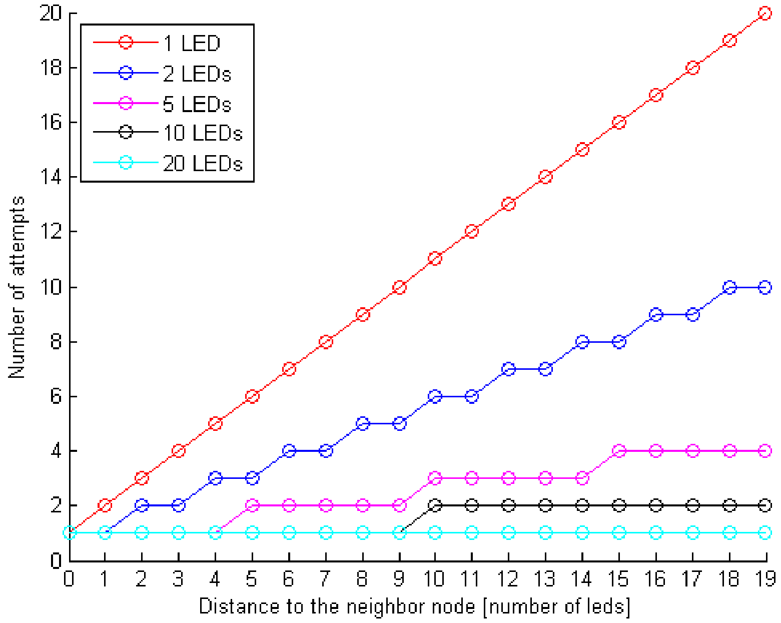


Figure 5.8: *Number of discovery attempts versus the distance to a neighbor.*

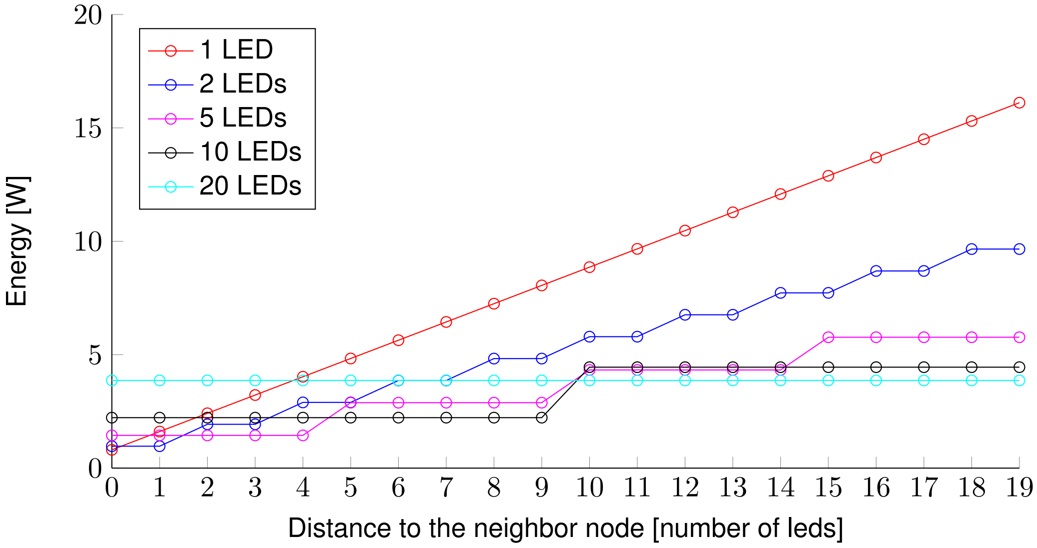


Figure 5.9: *Measured power versus the distance of the undiscovered neighbor.*

LED, the cost of one discovery step using ℓ LEDs is given by $Cost_i = (C_o + \ell.C_\ell)$. Recalling that L is the number of LEDs available in the board; the expected cost of discovering a node randomly placed between 0 and 360° using ℓ LEDs at each step is given by::

$$
\begin{aligned}
E_l[cost] &= \sum_{i=1}^{\frac{L}{l}} P[discovering\ the\ node\ at\ step\ i].Cost_i \\
&= \sum_{i=1}^{\frac{L}{l}} \frac{l}{L}.(i(C_0 + l.C_l)) \\
&= \frac{l}{L}(C_0 + l.C_l)(\frac{L}{l}+1)(\frac{L}{2l})
\end{aligned}
$$

From this equation we can determine the minimal energy cost for neighbor discovery, by taking the derivative of E_l with respect to l and making it equal to zero:

$$\frac{\partial E_l[cost]}{\partial l} = 0 \rightarrow l_{optimal} = \sqrt{\frac{C_0}{C_l}L} \tag{5.8}$$

From our experiments, we obtained that the ratio $\frac{C_0}{C_l}$ is equal to 4, which leads to an $l_{optimal}$ of approximately 9 LEDs.

If we take the average power consumption for each LED configuration (from Figure 5.9), we obtain Figure 5.10. This figure shows that using 10 LEDs will be the optimal selection in terms of energy consumption (similar to the value derived by our analytical model). However, this comes at the cost of lower precision in the location of the neighbor (only two sectors) and higher interference.

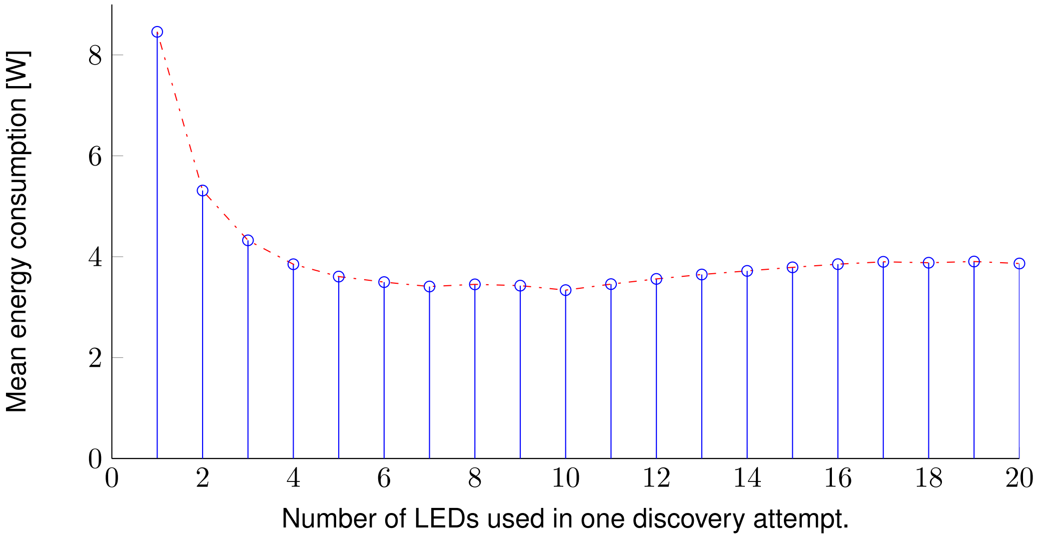


Figure 5.10: *Mean power consumption at a different number of LEDs used during discovery for a random placed neighbor.*

5.4 Network throughput

5.4.1 introduction

In this experiment, the goal is to see if the system is capable of transferring a message M_1 from node A, through node B, to node C. To accomplish this data transfer, the system needs to use all components within the network stack, and thus, this experiment is a full system evaluation. Important features involved in this experiment are: (i) the neighbor discovery algorithm, (ii) the data link mechanism (clear channel assessment) (iii) the addressing mechanism, and (iv) the forwarding mechanism.

5.4.2 Methodology

To test multihop networking we place three boards in a configuration such that node A and node C are not within reach of each other, but node B is within reach of both. This configuration is shown in Figure 5.11. We place the boards in such a way that it will take A two discovery attempts before discovery B. The step size for discovery is five LEDs, so it is the same as the reception angle of the receivers.

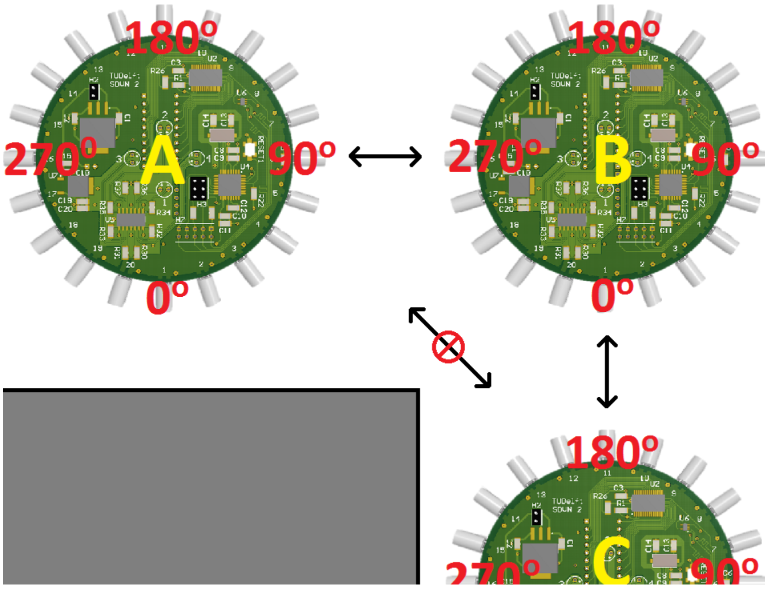


Figure 5.11: *Visual representation of routing.*

To showcase the events involved during multihop communication, we use a timeline. To log the necessary information, the network stack is modified to write specific events to the terminal. Because we use three boards, we have three instances of the network stack running on a PC, so we can use the system's time as a common baseline.

In the experiments, boards B and C already know about each other's existence and have a valid link. Board A does not know about the position of board B. Hence, to send a message to node C, it first needs to discover B and use it as intermediate relay.

5.4.3 Results

As figure 5.12 shows, we have two discovery attempts from board A, as expected. Between the two discovery attempts the system waits for some time to allow board B to reply. When this reply is not received, board A retransmits the request in the next direction. We can see that board B takes some time to process the transmission and then sends a reply to inform A that they are direct neighbors. Node A also records the direction of B. After the discovery process is completed , A sends a packet to B, approximately at time $t = 1000ms$. B receives this message, decodes it, sees that C is the destination, and then forwards the information to C. The entire transmission including discovery took 1.5 seconds.

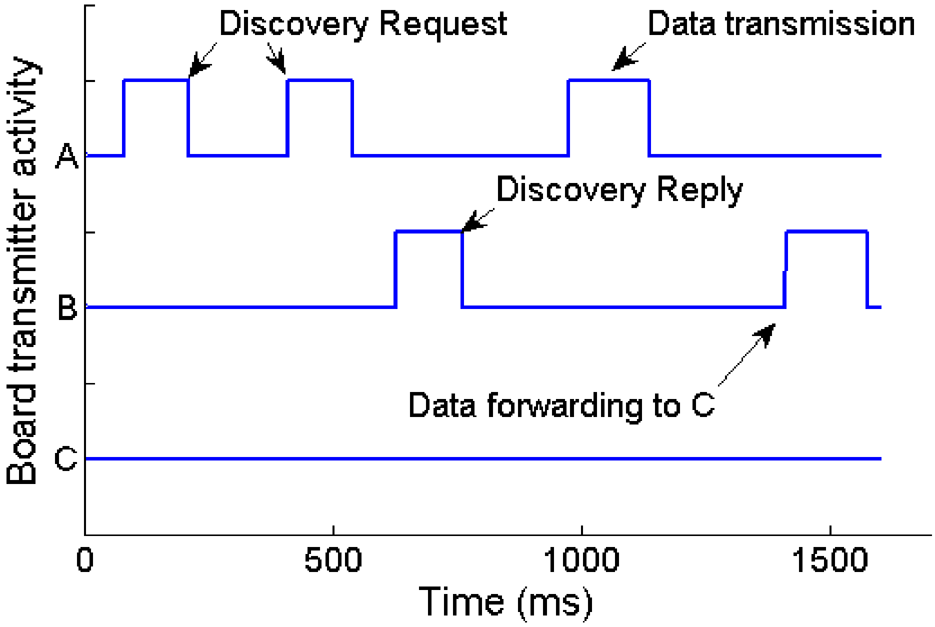


Figure 5.12: *Timeline of events during the experiment.*

Chapter 6

Conclusions and Future Work

6.1 Conclusions

Motivated by the significant attention that visible light communication has gained in recent years, we wanted to develop a platform that is amenable to the embedded systems community. To realize these ideas, an experimental platform is designed with distributed multi-hop communication capabilities. This platform has performance comparable with current state-of-the-art alternatives with similar resources, but is unique in two important aspects: (i) it exposes the directionality of VLC, and (ii) it is generic enough to be connected to various embedded systems and processing platforms. As our platform is experimental, there are areas of improvement in both hardware and software. We hope that this platform will lower the threshold for research into VLC and its unique features, so future LED-networks can be investigated and exploited.

6.2 Future Work

Our work opened up many more research problems than the ones we solved, which is a good thing. While developing the platform, we identified many ideas for future work. Besides the ideas presented in Chapter 3, that we did not have time to evaluate, below we present some other open opportunities for future work.

6.2.1 Improvements to the processing unit

The board is designed to support directional transmissions, and it is mostly focused on the transmitters and receivers. Improvement steps in terms of processing can be made so that more advanced (and computationally intensive) modulation can be used. An FPGA is very suitable for this line of work, so a logical step is to outfit the hardware with a FPGA. If more advanced modulation methods are applied, the hardware will become more robust to noise, have better reception reach, and will improve in terms of signal stability.

6.2.2 Improvements to the receiver

Sensitivity is an important issue in receiver design. The current receiver has an operational amplifier that is designed to focus on weaker signals. Signal reception can be improved by applying a dynamic gain to this amplifier, so that the hardware can zoom into a weak signal by increasing the amplifier gain. This will benefit range and signal strength, both good improvements to a communication system.

6.2.3 Extending the routing software

There are multiple opportunities to extend the network stack, as only basic networking services are currently available. Extensions to the router can be made in the form of motion support, by developing more advanced methods for neighbor discovery and tracking.

6.2.4 Improving the discovery algorithm

In Section 5.3 we noticed the relation between the discovery step size, energy consumption and precision of the location. When discovery attempts are repeated with a certain time interval, it is possible to increase precision without the consumption penalty of a small discovery step size, by shifting the start position by one LED at each new attempt. The overlap of repeated discovery attempts can increase the precision of the locational information. This technique needs to be evaluated further.

Bibliography

[1] Mostafa Z Afgani, Harald Haas, Hany Elgala, and Dietmar Knipp. Visible light communication using ofdm. In *Testbeds and Research Infrastructures for the Development of Networks and Communities, 2006. TRIDENTCOM 2006. 2nd International Conference on*, pages 6–pp. IEEE, 2006.

[2] S.E. Alavi, H. Rezaie, and AS.M. Supa'at. Application of ofdm on integrated system of visible free space optic with plc. In *Applied Electromagnetics (APACE), 2010 IEEE Asia-Pacific Conference on*, pages 1–5, Nov 2010.

[3] Y. Ashida, K. Okuda, and W. Uemura. A vlc receiving devise using audio jacks with a folding noise. In *Information Theory and its Applications (ISITA), 2012 International Symposium on*, pages 475–479, Oct 2012.

[4] Ahmad Helmi Azhar, T Tran, and Dominic O'Brien. A gigabit ps indoor wireless transmission using mimo-ofdm visible-light communications. *Photonics Technology Letters, IEEE*, 25(2):171–174, 2013.

[5] G. Corbellini, K. Aksit, S. Schmid, S. Mangold, and T. Gross. Connecting networks of toys and smartphones with visible light communication. *Communications Magazine, IEEE*, 52(7):72–78, July 2014.

[6] S. De Lausnay, L. De Strycker, J.-P. Goemaere, N. Stevens, and B. Nauwelaers. Matlab based platform for the evaluation of modulation techniques used in vlc. In *Development and Application Systems (DAS), 2014 International Conference on*, pages 57–61, May 2014.

[7] Gábor Feher, Eszter Udvary, C Fuzy, Tamás Cseh, and Tibor Berceli. Pulsed mode red vcsel for high speed vlc communication. In *Transparent Optical Networks (ICTON), 2012 14th International Conference on*, pages 1–4. IEEE, 2012.

[8] Mariam M Galal, Heba A Fayed, Ahmed Abd El Aziz, and Moustafa H Aly. Smartphones for payments and withdrawals utilizing embedded led flashlight for high speed data transmission. In *Computational Intelligence, Communication Systems and Networks (CICSyN), 2013 Fifth International Conference on*, pages 63–66. IEEE, 2013.

[9] P.A Haigh, Z. Ghassemlooy, and I Papakonstantinou. 1.4-mb/s white organic led transmission system using discrete multitone modulation. *Photonics Technology Letters, IEEE*, 25(6):615–618, March 2013.

[10] Hyun-Seung Kim, Deok-Rae Kim, Se-Hoon Yang, Yong-Hwan Son, and Sang-Kook Han. Inter-cell interference mitigation and indoor positioning system based on carrier allocation visible light communication. In *Signal Processing and Communication Systems (ICSPCS), 2011 5th International Conference on*, pages 1–7, Dec 2011.

[11] Toshihiko Komine and Masao Nakagawa. Fundamental analysis for visible-light communication system using led lights. *Consumer Electronics, IEEE Transactions on*, 50(1):100–107, 2004.

[12] Nam-Tuan Le, Trang Nguyen, and Yeong Min Jang. Frequency shift on-off keying for optical camera communication. In *Ubiquitous and Future Networks (ICUFN), 2014 Sixth International Conf on*, pages 22–25, July 2014.

[13] Xiaoxue Ma, Kyujin Lee, and Kyesan Lee. Appropriate modulation scheme for visible light communication systems considering illumination. *Electronics Letters*, 48(18):1137–1139, August 2012.

[14] Ratan Kumar Mondal, Nirzhar Saha, and Yeong Min Jang. Performance enhancement of mimo based visible light communication. In *Electrical Information and Communication Technology (EICT), 2013 International Conference on*, pages 1–5. IEEE, 2014.

[15] OSRAM Opto Semiconductors GmbH. *SFH203P datasheet*, rev.1 edition, 10 2003. http://www.osram-os.com/Graphics/XPic8/00029214_0.pdf/SFH%20203%20P,%20SFH%20203PFA.pdf.

[16] Stefan Schmid, Giorgio Corbellini, Stefan Mangold, and Thomas R Gross. Led-to-led visible light communication networks. In *Proceedings of the fourteenth ACM international symposium on Mobile ad hoc networking and computing*, pages 1–10. ACM, 2013.

[17] D. Tsonev, Hyunchae Chun, S. Rajbhandari, J.J.D. McKendry, S. Videv, E. Gu, M. Haji, S. Watson, AE. Kelly, G. Faulkner, M.D. Dawson, H. Haas, and D. O'Brien. A 3-gbps single-led ofdm-based wireless vlc link using a gallium nitride $\mu\mathrm{led}$. *Photonics Technology Letters, IEEE*, 26(7):637–640, April 2014.

[18] Santosh Verma, Abhinav Shandilya, and Ankit Singh. A model for reducing the effect of ambient light source in vlc system. In *Advance Computing Conference (IACC), 2014 IEEE International*, pages 186–188. IEEE, 2014.

[19] Jin-Yuan Wang, Jun-Bo Wang, Ming Chen, and Xiaoyu Song. Dimming scheme analysis for pulse amplitude modulated visible light communications. In *Wireless Communications Signal Processing (WCSP), 2013 International Conference on*, pages 1–6, Oct 2013.

[20] Qing Wang, Domenico Giustiniano, and Daniele Puccinelli. Openvlc: Software-defined visible light embedded networks. *ACM MobiCom 2014*, 2014.

[21] WG802.15. *Short-Range Wireless Optical Communication Using Visible Light*. IEEE, ieee 802.15.7-2011 edition, 2011.

[22] F.M. Wu, C.T. Lin, C.C. Wei, C.W. Chen, Z.Y. Chen, H.T. Huang, and Sien Chi. Performance comparison of ofdm signal and cap signal over high capacity rgb-led-based wdm visible light communication. *Photonics Journal, IEEE*, 5(4):7901507–7901507, Aug 2013.

[23] Fang-Ming Wu, Chun-Ting Lin, Chia-Chien Wei, Cheng-Wei Chen, Zhen-Yu Chen, and Hou-Tzu Huang. 3.22-gb/s wdm visible light communication of a single rgb led employing carrier-less amplitude and phase modulation. In *Optical Fiber Communication Conference and Exposition and the National Fiber Optic Engineers Conference (OFC/NFOEC), 2013*, pages 1–3, March 2013.

[24] Takaya Yamazato, Isamu Takai, Hiraku Okada, Toshiaki Fujii, Tomohiro Yendo, Shintaro Arai, Michinori Andoh, Tomohisa Harada, Keita Yasutomi, Keiichiro Kagawa, et al. Image-sensor-based visible light communication for automotive applications. *Communications Magazine, IEEE*, 52(7):88–97, 2014.

[25] Hongming Yang and A Pandharipande. Full-duplex relay vlc in led lighting linear system topology. In *Industrial Electronics Society, IECON 2013 - 39th Annual Conference of the IEEE*, pages 6075–6080, Nov 2013.

[26] Hongming Yang and Ashish Pandharipande. Full-duplex relay vlc in led lighting triangular system topology. In *Communications, Control and Signal Processing (ISCCSP), 2014 6th International Symposium on*, pages 85–88, May 2014.

[27] Shuailong Zhang, Scott Watson, Jonathan JD McKendry, David Massoubre, Andrew Cogman, Erdan Gu, Robert K Henderson, Anthony E Kelly, and Martin D Dawson. 1.5 gbit/s multi-channel visible light communications using cmos-controlled gan-based leds. *Journal of Lightwave Technology*, 31(8):1211–1216, 2013.

Appendix A

Measurement results

Table A.1: Intensity values measured from the edge of the middle led, measured in Lumen [cd.sr]

Distance of center	-14	-12	-10	-8	-6	-4	-2	0	2	4	6	8	10	12	14
0	400	930	2930	4000	6780	11580	18820	25000	18500	13160	4470	6810	2079	1507	2804
2	570	1600	3079	3239	2780	4580	15570	14800	9980	2283	7260	1003	966	3112	1031
4	1200	1400	840	1165	6330	3540	2150	9220	1500	3985	5300	4430	2488	2403	2270
6	1280	864	374	2630	2024	5110	1262	4360	845	4490	9770	3600	2255	682	980
8	640	309	1490	1218	518	3740	1005	3658	1620	3032	1370	1740	1940	1054	497
10	276	609	1445	590	1003	2624	1093	2876	741	1570	1840	530	1060	1460	1053
12	588	1068	1105	371	1441	2322	752	2132	707	824	1906	899	381	1378	1078
14	888	1002	496	455	1315	1380	675	1558	936	440	1128	984	370	470	929
16	763	771	314	785	1100	1016	635	1308	934	333	671	1115	407	290	514
18	715	442	329	888	872	412	529	1057	847	278	361	726	700	345	274
20	490	251	480	756	700	292	473	904	783	265	250	545	610	486	156
22	290	220	293	602	546	243	431	776	666	306	222	425	508	501	263
24	198	262	595	540	417	218	384	651	597	294	184	330	443	461	338
26	190	317	518	508	298	203	385	595	538	318	170	254	412	382	368
28	215	345	439	412	235	203	361	505	470	312	181	179	349	333	350
30	237	349	380	317	188	187	312	435	422	295	140	139	241	305	341
32	262	335	335	264	160	174	296	388	378	272	140	122	191	286	287
34	261	291	310	221	140	155	261	342	333	254	137	112	143	255	257
36	258	261	261	178	127	157	253	308	300	237	115	103	117	198	234
38	234	232	220	163	117	149	239	276	280	222	133	94	99	185	218
40	209	214	187	143	110	137	211	247	251	204	136	92	87	160	216
42	198	193	160	113	107	130	191	222	230	190	137	88	76	144	190
44	175	168	135	103	100	122	186	202	211	175	119	84	73	94	160
46	160	144	118	89	95	116	172	185	190	161	115	85	73	82	143
48	146	123	102	82	90	113	158	172	174	149	112	79	72	70	120
50	134	109	89	74	87	105	146	159	161	137	110	76	71	62	105
52	119	96	79	69	84	97	136	142	148	129	110	73	65	57	94
54	107	85	70	64	79	92	126	130	136	121	101	72	63	53	80
56	94	77	63	60	77	86	117	121	126	110	97	72	61	50	70
58	84	66	57	56	73	80	110	113	117	104	87	69	58	49	61
60	75	61	51	53	71	72	103	106	108	98	85	67	56	47	53

Table A.2: Power versus number of leds used during transmission of a neighbor discovery message.

Number of leds	Average current [A]	Average voltage [V]	Power [W]
0	0,054	12,00	0,642
1	0,067	12,00	0,806
2	0,080	12,00	0,966
3	0,094	12,00	1,123
4	0,107	12,00	1,284
5	0,120	12,00	1,443
6	0,132	12,00	1,589
7	0,146	12,00	1,748
8	0,160	12,00	1,918
9	0,173	12,00	2,077
10	0,185	12,00	2,225
11	0,199	12,00	2,384
12	0,212	12,00	2,543
13	0,225	12,00	2,702
14	0,238	12,00	2,861
15	0,253	12,00	3,032
16	0,268	12,00	3,210
17	0,283	12,00	3,390
18	0,294	12,00	3,528
19	0,310	12,00	3,719
20	0,322	12,00	3,866

Appendix B

Overview of UART commands.

Start-of-Header	Message Type	Message	End-of-Transmission
0x01	1 Byte	.. Bytes	0x04

Table B.1: Generic message format.

1	Channel
1 Byte	2 Byte

Table B.2: Data read request message.

2	Starting Angle	Ending Angle	Length	Payload	RSS
1 Byte	2 Byte	2 Byte	1 Byte	1..128 Byte	2 Byte

Table B.3: Data read request message.

3
1 Byte

Table B.4: No data available answer message.

4	Channel number
1 Byte	2 Byte

Table B.5: Medium information request message.

5	Channel state	Channel noise floor
1 Byte	1 Byte	2 Byte

Table B.6: Medium information answer message.

State field value	Meaning
0	Channel is disabled.
1	Channel medium is idle.
2	Channel is receiving incoming data.
>2	Not used.

Table B.7: Definition of the medium state field.

6	Channel number
1 Byte	2 Byte

Table B.8: Get receiver state request message.

7	Channel number	Channel state
1 Byte	2 Byte	1 Byte

Table B.9: Set receiver state request message.

8	Channel state
1 Byte	1 Byte

Table B.10: Receiver state answer message.

9	Starting Angle	Ending Angle	Length	Payload
1 Byte	2 Byte	2 Byte	1 Byte	1..128 Byte

Table B.11: Data transmission request message.

The starting angle field indicates the angle in which the hardware should start sending data, this area is closed by the ending angle. For example: suppose the starting angle is 30° and the ending angle is 20°, then the signal will be send in all directions from 20° to 359°, and from 0° till 20°. The signal will not be transmitted in the region between 20° and 30°. As shown, the angles are thus overlapping. The definition of the angles are clockwise.

10
1 Byte

Table B.12: Data transmission success answer message.

11
1 Byte

Table B.13: Data transmission failed answer message.