Digital & Analog Implementations of Visible Light Communication

Submitted to University College Dublin in part fulfilment of the requirements for the degree of Master of Electronic and Computer Engineering

April 2018

Author: Kevin Keegan
Thesis Supervisor: Dr Nam Tran

Acknowledgements

I would like to thank my parents and Nam Tran, my supervisor. This thesis would not have been possible without their support.

I would also like to thank my friends in UCD engineering who made pursuing this masters enjoyable; in particular Conor Power, Chris Hayes, Sam Luby & Darren Coughlan.

Declaration of Authorship

I certify that ALL of the following are true:

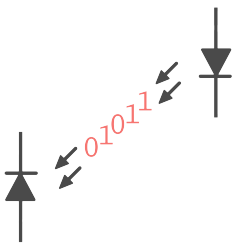
- I have read and fully understand the consequences of plagiarism as discussed in the *College of Engineering and Architecture Plagiarism Protocol* and the *UCD Plagiarism Policy.* These documents were available to me to properly consider.
- I fully understand the definition of plagiarism.
- I recongnise that plagiarised work (in whole or in part) may be subject to penalties as outlined in the *College of Engineering and Architecture Plagiarism Protocol* and the *UCD Plagiarism Policy.*
- I have not previously submitted this work, or any version of it, for assessment in any other subject or any other institution.
- All of the information I have provided in this Declaration is, to the best of my knowledge, correct.
- I have not plagiarised any part of this work; it is original and my own.

Approved: _____

Date: _____

Abstract

Driven partially by the Internet Of Things (IOT) the number of wireless devices is growing rapidly. This increase in wireless traffic has congested the usable RF bandwidth creating a critical need for a new technology in wireless communications that can operate in a different frequency spectrum. Visible light communication(VLC) is one such form of optical wireless communication (OWC). It can address the shortfalls and limitations of existing RF communication by utilising the large modulation bandwidth of LEDs. In this thesis, An analog system is implemented to transmit and receive mono audio via an amplitude modulated visible light signal. Audio is perfectly recognizable at a distance of upto 1.5m. Following this, digital VLC, the main goal of the thesis is achieved. Two real-time digital VLC systems are successfully implemented employing a form of On Off Keying (OOK) modulation. The first implementation is an Arduino UNO based LED-Photodiode communication link where all sample processing is performed on the inexpensive Arduino micro-controller. The second, transfers files from one computer to another. It performs processing on an external computer connected to the Arduino via USB. This configuration, although more costly, has the vastly greater processing power available on a typical computer enabling higher data rates and complex processing of incoming samples. The maximum achieved performance is a data rate of 98 bits/s over a distance of 1.8m.



Contents

1 Introduction **4**

2 Background & Literature Review **8**

2.1 Visible Light Channel Model 9

2.2 Intersymbol Interference 12

2.3 VLC Digital Modulation Schemes 13

2.3.1 Single Carrier Schemes 13

2.3.2 OFDM 14

2.3.3 Typical OFDM System using Cyclic Prefix 16

2.4 Visible Light Transmitter 18

2.4.1 Adapting OFDM for VLC 18

2.4.2 LED Types 19

2.5 Visible Light Receiver 19

2.5.1 Photodiodes 19

2.5.2 Synchronisation 20

2.6 Limitations of VLC 21

3 Simulation of OFDM on a Visible Light Channel **22**

3.1 Simulation Model 22

3.2 Simulation of OFDM Performance in Different Lighting Conditions . 24

4 Prototype Implementing Analog VLC **26**

4.1 Analog Transmitter 27

4.2 Analog Receiver 28

4.3 Results 28

5 Proposed Arduino Based Digital VLC **30**

5.1 Digital Modulation Scheme 30

5.1.1 Proposed On-Off Keying Based Scheme 30

5.1.2 Proposed System for Synchronisation 32

5.2 Real-Time Implementations of the Proposed System 33

5.2.1 Overview of Contributions 33

5.2.2 Arduino UNO 34

5.2.3 Proposed Implementation of a 5V Single-Supply Linear Transimpedance Photodiode Amplifier 36

5.3 Proposed System 1, (Arduino Based ASCII Transmitter) 41

5.3.1 Transmitter 41

5.3.2 Example: Recovery from Temporary Blockage 43

5.3.3 Receiver 46

5.3.4 Data Rate 49

5.4 Proposed System 2, (Arduino & Computer Based File Transmitter) . 50

5.4.1 Assembling Frames from the Outgoing File 51

5.4.2 Transmitting Frames 54

5.4.3 Sampling the Amplified Photodiode 55

5.4.4 Software Demodulation, Assembling the Received File 55

5.4.5 Data Rate 58

6 Results of the Digital System **59**

6.1 Range 59

6.1.1 The Effect of Ambient Light on Range 62

6.2 Data Rate 63
6.3 Blockage Recovery 64

7 Conclusions & Future Work **65**

Chapter 1

Introduction

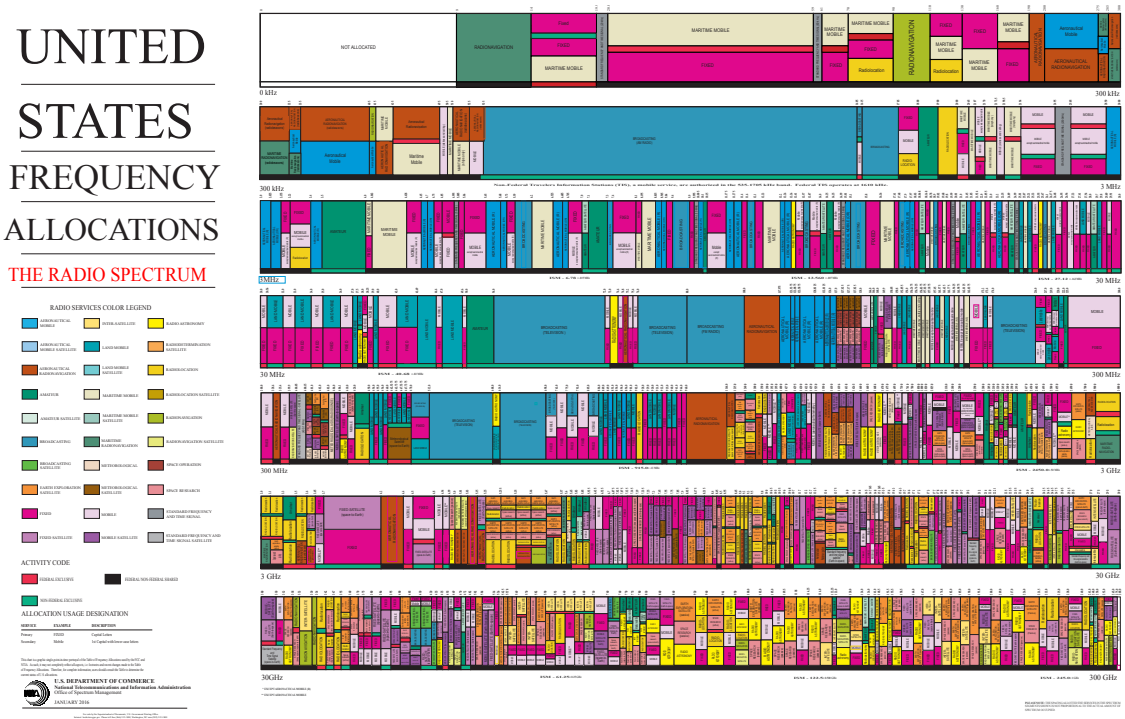


Figure 1.1: Official United States allocation chart for the RF spectrum 0 - 300GHz [1].

The majority of wireless communication today transmits in the Radio Frequency spectrum. This spectrum was originally chosen due to its low noise and large area coverage [2]. It consists of the electromagnetic (EM) frequencies from 20 kHz to 300 GHz . It is considered a country's natural resource and is consequently highly

regulated [3]. Domestic wireless systems can only transmit in the Amateur bandwidths of the RF spectrum. As an example, the official allocation of the United States RF spectrum is illustrated in Figure 1.1. In this chart Amateur bandwidths are labeled and coloured green. These bandwidths are the only unlicensed bandwidths in the spectrum and make up a relatively tiny portion of it [2]. This lack of usable bandwidth has led to overcrowding; for instance, Bluetooth, most WIFI, handsets, wireless mice and keyboards are all forced to share the bandwidth centered at 2.4GHz. As more devices share this bandwidth the speed of the connections decreases due to interference [4].

In light of IOT, the quantity of wireless devices is inevitably growing along with the need for higher and higher data rates. The bandwidth available to us in the RF spectrum will no longer suffice [5]. VLC, the most promising solution to this problem, transmits data via the visible portion of the electromagnetic spectrum, 400 THz to 800 THz. This whole spectrum is unlicensed and 1,000 times larger than the entire RF spectrum [2].

LEDs are the ideal VLC transmitter. They have a large bandwidth meaning their light intensity can be modulated very quickly. This translates to high data rates [6]. The LED is also becoming a widespread form of lighting due to its durability and energy efficiency [7]. By integrating VLC into existing lighting infrastructure, LED lights can provide lighting while simultaneously acting as VLC transmitters. Wherever we have lighting, we can also have strong wireless connectivity. Since the LED lighting would be paid for regardless of VLC, the cost of the VLC transmitter is arguably only that of the electronics needed to modulate the LEDs, making it less costly than alternatives.

VLC is intrinsically more secure than RF communication. As it is a form of OWC, it is highly directional and does not travel through opaque objects. Also, the light carrying the transmission is visible to the human eye by definition, making it easy to see and control where the signal can be accessed.

It is the main goal of this research to design and implement a Digital VLC system capable of transmitting at a distance of approximately one meter in an open plan office environment. The goal will then be to optimize the system, improving data throughput, range and robustness to changing ambient light. Such a system will have countless applications including file transfer and communication with IOT devices.

A typical implementation of a VLC system uses LEDs as transmitters and photodiodes(PD) (especially PIN photodiodes) as receivers [2]. In this configuration the intensity of the LED's light is directly varied by the data signal. This signal will then be present as a component of the current produced by the receiving PD.

The data signal to be transmitted is generated from binary data using digital modulation schemes. Single carrier schemes such as OOK or Pulse Position Modulation(PPM) are popular in VLC systems. These were carried over from infra-red communication, another form of OWC [8]. In research such as [6] and [9] where the goal was to maximize VLC data rates, researchers opted for the more complex Orthogonal Frequency Division Multiplexing (OFDM) achieving the highest data rates thus far at 7.36 Gb/s in lab conditions.

The significant advantage OFDM has over those previously mentioned is its inherent resilience against intersymbol interference (ISI) (i.e. multi-path effects). This form of interference is a significant throughput limiting factor of typical VLC systems [10]. However, as a rule of thumb the effects of ISI are negligible if the delay spread is less than $\frac{1}{10}$ of the symbol duration. Delay spread is a measure of the delay between the arrival of a symbol propagating through the shortest path (typically the line-of-sight path) and the arrival of the same symbol propagating through the longest significant path (typically a reflection path).

In [11], VLC transmission was modeled for medium-size room taking into account multiple propagation paths. They found that the delay spread of such a scenario would be $\tau \leq 3.47ns$. Meaning the effects of ISI can be ignored in similar scenarios as long as the modulation scheme being used has a symbol duration $\geq 34.7ns$.

Therefore ISI can be ignored in an OOK based system transmitting at speeds up to $\frac{1}{34.7ns} \Rightarrow 29Mbps$.

The field of VLC is still a new one, there has not yet been a great deal of research published on it and commercial applications such as *Indoor Positioning by Philips* have only begun to appear.

Part of this work was also presented in a conference paper entitled *Arduino Based Implementations of Digital Visible Light Communication*

Chapter 2

Background & Literature Review

A common approach to designing a digital VLC link is to use a micro-controller at the receiving and transmitting ends. The transmitter's micro-controller handles the process of taking binary data and producing from it a transmittable baseband signal. This signal is sent by directly varying the instantaneous power output of an LED. The LED's light becomes an intensity modulated optical signal [2].

The transmitted signal is distorted as it passes through the visible light channel and shines on the receiver. As a result the received signal is expected to differ from the transmitted one. The received signal is generally demodulated using analog electronics for speed. The output of the demodulating circuitry is often a binary baseband signal that is read by the micro-controller. The micro-controller processes this baseband signal and from it recovers original binary data. Opting for more complex modulation schemes can achieve higher data throughputs but be more difficult and expensive to implement. The cost and complexity of the receiver circuitry can be substantially reduced if signal demodulation is performed in software. To implement this, the received waveform is sampled directly by the micro-controller using an analog to digital converter (ADC). The samples can then be processed in software to perform the demodulation using IIR or FIR filters. It is substantially easier and more flexible to implement demodulation in software, however it requires

much higher sampling rates and processing power, making it much slower in terms of throughput.

2.1 Visible Light Channel Model

The robustness of our visible light channel model determines the validity of our simulations. The digital system is designed to be used in an indoor open-plan office environment. The channel is modeled as the impulse response $h(t)$. The received signal $r(t)$ is taken to be the linear convolution of the transmitted signal $x(t)$ and $h(t)$ plus additive white Gaussian noise $n(t)$. $h(t)$ will consist of sequence of scaled, shifted impulses taking into account the delay of multiple propagation paths and the decrease in LED light intensity over distance due to geometric spreading. The additive noise models the thermal noise at the receiver along with any interference from the office environment.

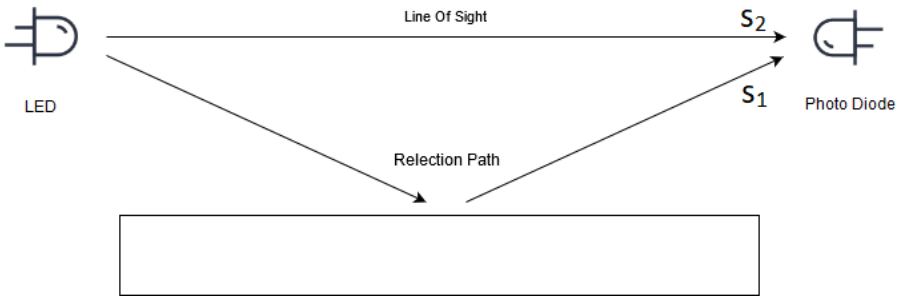


Figure 2.1: Illustration of two different propagation paths through the VLC channel from the transmitting LED to the receiving Photodiode.

$$r(t) = x(t) \star h(t) + n(t) \tag{2.1}$$

The impulse response $h(t)$ can be broken down into two components:

1. Line of Sight link $h_{los}(t)$

2. Link due to reflections $h_{ref}(t)$

$h_{ref}(t)$ is the sum of the impulse responses of every individual reflection path that we wish to include in our model. Most reflections are typically diffuse in nature, and most are well modeled as Lambertian [12]. In [11] only two reflections are modeled, from a wall and a floor. Only single bounce reflections are considered, this is where light only reflects off a surface once between transmitter and receiver. In other works such as [8] they deemed it sufficient to only model the LOS link ignoring reflections paths altogether. This could be done because they found that the symbol duration of their modulation scheme was greater than ten times the delay spread of the channel rendering the effect of multi-path propagation negligible. When simulating high data rates however, it is important not to neglect reflection paths as they allow us to model ISI.

In [11] they employ equation 2.2 as a mathematical model for the line of sight (LOS) channel impulse response. It is derived from a model of the infra-red (IR) optical channel. This can be done because the visible light channel is similar to the IR channel as they are both optical; however it is worth noting that due to the difference in wavelengths, there will be a difference in the reflectivity of surfaces and this will have some impact on channel characteristics [2].

$$h_{los}(t) = \frac{A_{PD}(m+1)\,cos^{m}\left(\phi\right)cos\left(\theta\right)}{2\pi d^{\,2}}\,rect\left(\frac{\theta}{FOV}\right) \times\delta\left(t-\frac{d}{c}\right) \tag{2.2}$$

- A_{PD} - Active area of the PD
- $m = -\frac{1}{log_2 cos(\phi_{1/2})}$ The directivity of the light beam
- $\phi_{1/2}$ - Half-angle of the LED
- ϕ - Angle of Irradiance - The angle between a line connecting the PD and the LED and the normal to the LED
- θ - Angle of Incidence - The angle between a line connecting the PD and the LED and the normal to the PD
- FOV - Field of View of the PD
- d - Distance between the LED and the PD
- c - Speed of light

The reflection path impulse response modeled in equation 2.3 depends mostly on variables similar to those in the LOS model. Variables denoted 1 pertain to the segment of the path from the LED to the reflection surface. Variables denoted with 2 pertain to the second segment of the path that is from the surface to the PD. Γ_{surf} is the reflection coefficient of the surface and A_{surf} is the active area of the surface.

$$L_1 = \frac{A_{surf}(m+1)\, cos^m \phi_{surf,1}\, cos\theta_{surf,1}}{2\pi\, {d_{surf,1}}^2}$$

$$L_2 = \frac{A_{PD}(m+1)\, cos^m \phi_{surf,2}\, cos\theta_{surf,2}}{2\pi\, {d_{surf,2}}^2}$$

$$h_{ref}(t) = L_1 L_2\, \Gamma_{surf}\, rect\left(\frac{\theta_{surf,2}}{FOV}\right) \sigma\left(t - \frac{d_{surf,1} + d_{surf,2}}{c}\right) \tag{2.3}$$

These equations specifically model the impulse response of the channel between one LED and one PD. To handle cases where there are multiple LEDs and PDs, the model must be applied to all links from each LED to each PD individually.

In [8], an optical channel model is used that is quite similar to the one previously described. In that study the model proved accurate as the simulation results closely resembled experimental results.

2.2 Intersymbol Interference

In digitally modulated signals data is sent as a sequence of symbols. These symbols each represent one or more bits. A symbol is a waveform with a unique characteristic that is sent for a duration known as the symbol duration. ISI is a form of signal distortion that arises when multiple symbols arrive at the receiver at similar times and thus interfere each other. This occurs because symbols can propagate though many different paths. A symbol propagating through a long reflection path can arrive at the receiver at the same time as a subsequent symbol propagating through a shorter one. This can also be described as a channel with a non-flat frequency response.

To increase the throughput of a particular system we often increase the symbol rate, thus decreasing the symbol duration. ISI becomes a limiting factor on how much we can decrease the symbol duration. As symbol duration decreases the delay

spread of the channel becomes more significant. However, if the delay spread is less than $\frac{1}{10}$ of symbol duration the effects of ISI can be ignored [11]. The effects of ISI can undone or reduced applying an equalizer at the receiver.

2.3 VLC Digital Modulation Schemes

Digital modulation schemes dictate how we form a baseband signal from an input bit stream. This baseband signal can be used to modulate a carrier signal. However, in the case of VLC implementations, the baseband signal is often sent directly by modulating the transmitter's instantaneous power output. Single carrier modulations such as OOK and PPM have been widely used in IR communications [13]. Unsurprisingly this trend followed to VLC research such as the in-flight entertainment application in [14]. In this study Differential Pulse Position Modulation, a variation of PPM, was applied. Their system achieved a baud rate of 1 Mb/s capable of streaming MPEG -TS encoded video.

2.3.1 Single Carrier Schemes

Figure 2.2: Illustration of the 4-ary Pulse Position modulation symbol set. Symbols are all of equal duration and contain identical pulses. They difference between them is the time at which the pulse occurs

In PPM the symbol is a time frame of fixed duration. For fraction of the time frame a pulse of fixed width and amplitude occurs. It is the position in time where the pulse occurs that determines the symbol's value.

Baseband OOK is the simplest form of amplitude shift keying. Bits are sent at a fixed rate. As illustrated in Figure 2.3, to send a 1 the LED is turned fully on for a symbol duration; to send a 0 the LED is off for a symbol duration. PPM

is more power efficient but less bandwidth efficient than OOK [8]. These single carrier modulation techniques are worth considering because of their simplicity and consequent low cost. In general, these schemes are suitable for VLC systems when low to moderate data rates are required. "However in general, the performance of single carrier modulation techniques deteriorate as the data rates increase due to the increased ISI" [5].

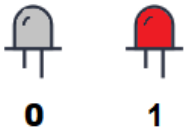


Figure 2.3: LED state for each baseband OOK symbol.

2.3.2 OFDM

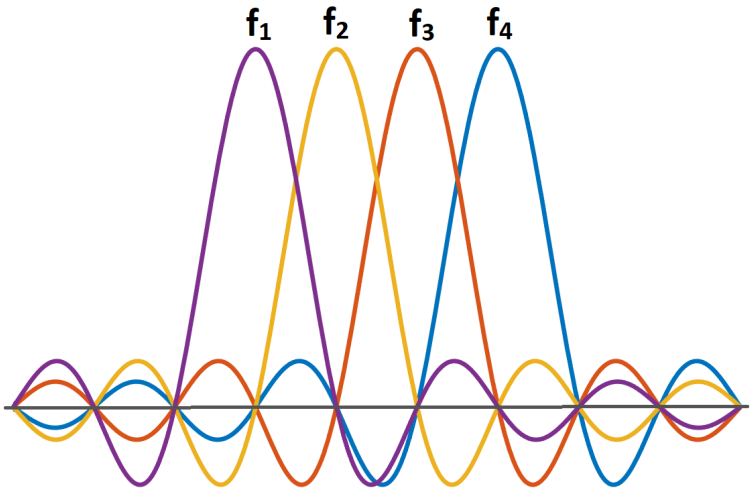


Figure 2.4: Frequency spectrum of four orthogonally spaced OFDM sub-carriers. This shows how adjacent sub-carriers to not interfere as the peaks of one sub-carriers spectrum align with the nodes of the others.

When high data rates are of interest, OFDM has been proven to be superior to single carriers. It counters speed limiting ISI in two very effective ways. Firstly, it can utilize a large bandwidth while maintaining long symbol durations. This gives it

the high data rate associated with utilizing a large bandwidth while keeping the ISI resistance of long symbols. Second, the OFDM setup makes it very easy to insert a frequency domain equalizer. This consists of a simple division but it can undo the effects of ISI. Its effectiveness depends on the accuracy of your channel model. Taking the OFDM approach for high data rate VLC was successfully chosen by [10], [8], [9] & [6] to name a few. With it they achieved the highest VLC data rates at the time of their publishing.

In OFDM the entire available bandwidth is split up in to many sub-bands, for example in the IEEE 802.11 WIFI standard 64 are used. Each sub-band has its own sinusoidal sub-carrier. These sub-carriers are all orthogonal to each other. Each sub-carrier is independently modulated with data, typically using PSK or QAM modulation. The data to be sent is sent in parallel streams. If 64 sub-carriers are used then each sub-carrier can only use a 64^{th} of the channel's bandwidth. In theory for QPSK this would mean that the sub-carrier will have a 64^{th} of the data rate of a single carrier using the entire bandwidth. However, as there are 64 of these sub-carriers OFDM will theoretically achieve the same speed. The sub-carriers each have symbols that would be 64 times longer than that of a single carrier scheme. In practice not all sub-bands are used to send data. Some are reserved for pilot carriers used for estimating the channel. Others named guard bands are left unused. The technique of bit loading is used to further optimise an OFDM system. Higher order modulation such as 64-QAM is applied to the sub-bands with least noise. Sub-bands that are highly noisy can be left unused or given less complex modulation to minimise error rates. This technique was employed by [9]. The approach proved successful allowing them to achieve a data rate of 3 Gb/s. Another method to improve OFDM's performance is channel coding. Coding schemes such as convolution code should bring a substantial improvement to the Bit Error Rate (BER) by introducing redundant bits that allow certain bit errors to be corrected without retransmission.

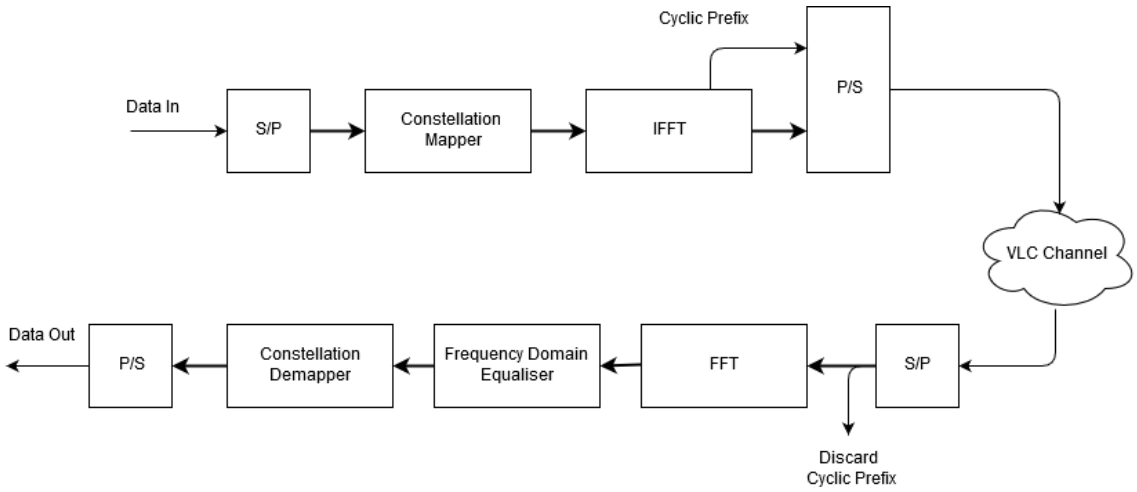


Figure 2.5: System diagram of an OFDM system using a cyclic prefix to allow for single tap frequency domain equalisation.

2.3.3 Typical OFDM System using Cyclic Prefix

The diagram in Figure 2.5 illustrates the OFDM system. It functions as follows.

1. A serial stream of bits is converted into parallel streams. One for each sub-carrier.

2. A constellation mapper converts each parallel stream of bits in to a stream of symbols corresponding to the required modulation format (PSK or QAM).

3. These symbols must be multiplied by their corresponding sub-carrier in order to make the OFDM symbol. This is efficiently done by passing the symbols as frequency domain coefficients to an inverse Fourier transform. The output of this IFFT is a time domain sequence that is a sum of all the modulated sub-carriers.

4. A cyclic prefix is added to the start of the IFFT output. It is a copy of the last X samples of the OFDM symbol.

5. This new OFDM symbol is sent sample by sample through a digital to analog converter and then through the VLC channel. The sequence is linearly convolved with the channel's impulse response.

6. The addition of the cyclic prefix prior to sending is essential because with it the received signal contains the circular convolution of the original OFDM symbol and the channel. We retrieve this by simply discarding the cyclic prefix at the receiver.

7. The received sequence (without prefix) is passed through an FFT.

8. Ensuring that what we receive is the result of circular convolution is important for the next stage. This stage is the frequency domain equaliser. Here we divide the FFT of the received symbol by the Fourier transform of our channel model's impulse response. This frequency domain division will perform a circular deconvolution in the time domain, reversing distortion caused by the channel. This would not work if we did not have the cyclic prefix.

9. The output of the equaliser is passed through a constellation demapper.

10. We have now recovered the bits originally sent.

The cyclic prefix should ideally be as long as the longest significant delay in the channel.

2.4 Visible Light Transmitter

2.4.1 Adapting OFDM for VLC

OFDM VLC systems such as those implemented in [10], [8], [6] & [9], were designed to directly modulate the intensity of the LED transmitter with the samples of the OFDM symbols. This is done via a digital to analog converter. However, if one is to take this approach, modifications must be made to the OFDM system. A standard OFDM implementation will produce a signal that is both complex valued and non-unipolar. This is incompatible with the implementation described above. LEDs will clip the negative portion of the signal and complex values cannot be sent using intensity modulation (IM) without adding further complexity to the design. Both of these issues are overcome by manipulating the inputs to the IFFT stage. If the input has Hermitian symmetry, the output of the IFFT is guaranteed to be real. Hermitian symmetry is applied by ensuring the negative frequency inputs to the IFFT are conjugates of their corresponding positive frequency inputs. With this approach we sacrifice half the bandwidth of the IFFT hardware.

The first IFFT input corresponds the 0Hz, the DC value. Giving this input a value greater than 0 will apply a DC shift on the output signal. Ideally this value is chosen so that the input to the LED is unipolar and also so that the LED operates in its most linear region throughout the entire transmission.
This modulation scheme is known as DC offset OFDM, *DCO-OFDM* [5]. An example of the IFFT input for this scheme is given in Table 2.1.

$$\text{DC} \mid X_1 \mid X_2 \mid X_3 \mid 0 \mid \overline{X_3} \mid \overline{X_2} \mid \overline{X_1}$$

Table 2.1: IFFT inputs in a DCO-OFDM system. DC input ensures output samples are unipolar. Negative frequency inputs $\overline{X_i}$ are conjugates of their corresponding positive frequency inputs X_i. This Hermitian symmetry guarantees real valued output samples.

2.4.2 LED Types

It is misleading to dwell on the visible light spectrum's bandwidth when discussing the potential data rates of the medium. This is because the vast majority of VLC systems transmit by intensity modulating LEDs as opposed to modulating the frequency(colour) of the light transmitted. The bandwidth of the medium is in reality only the bandwidth of the LED not the spectrum. This is why an LED's bandwidth, i.e. the range of input frequencies for which attenuation is < 3dB, is an important property to consider. A variety of LEDs have been used in VLC studies. In [10], the ubiquitous PC-LEDs are used. These LEDs are the most popular white LED, often used for lighting. The white colour is achieved by a phosphor coating on a blue, Indium Gallium Nitride LED. The phosphor coverts a portion of the blue light to green, yellow and red. These mix with the rest of the blue light, producing white light [15]. This phosphor coating limits the modulation bandwidth of the LED. However, the phosphor's slow response can be removed using a blue light filter at the receiver through which only the blue light from underlying Indium Gallium Nitride LED passes [6].

Micro LEDs (μ-LEDs) achieve the highest bandwidths. This property comes from their small active area and low capacitance [2]. Their modulation bandwidths are generally in the hundreds of megahertz. This LED is used when the highest data-rates have been achieved [6], [9].

2.5 Visible Light Receiver

2.5.1 Photodiodes

At the receiver a PD is used to detect the transmitted light signal. A photodiode is often modeled as a current source whose current is linearly controlled by the intensity of incident light. PIN photodiodes are particularly popular receivers as can operate

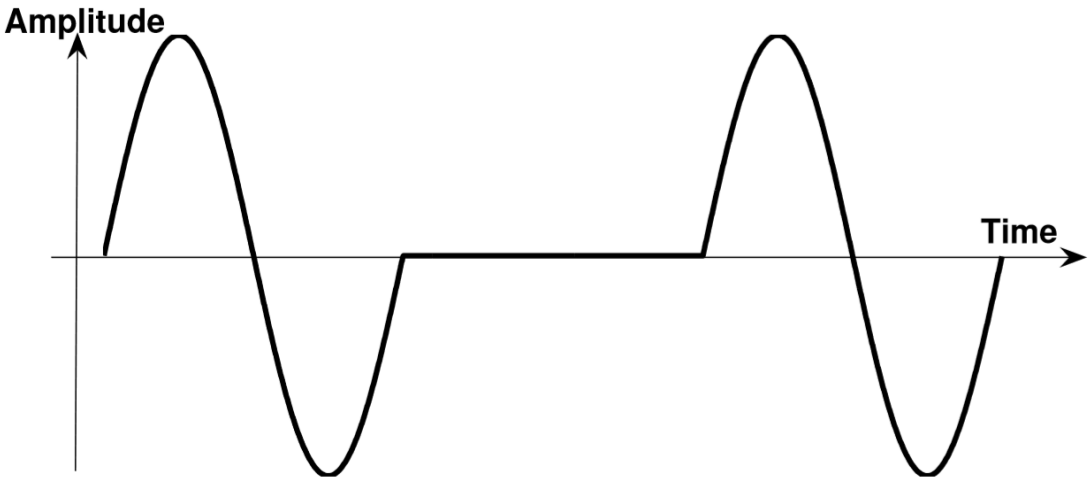


Figure 2.6: Plot of pilot waveform used in [8].

at very high frequencies. The general purpose SFH 203 PIN diode can detected IM signals at frequencies exceeding 100 MHz.

2.5.2 Synchronisation

The issue of synchronisation between transmitter and receiver is one that arises when a system is being physically implemented. In [8] the pilot symbol presented in Figure 2.6 is sent directly before OFDM symbols. This pilot symbol is known to both transmitter and receiver. Its purpose is to facilitate identifying the start of the OFDM symbols. The receiver performs a sum of the absolute values of a window of past samples. The length of this window is the same as duration of the "silent" period in the center of the pilot. The sum will be non-zero when the sinusoidal sections at either end of the pilot fall within the sum window. The synchronisation algorithm notes the position at which the minimum value of the sum is observed. This will occur at the end of the silent period, where neither sinusoidal section falls within the sum window. From this the starting position of the OFDM symbol can be easily determined. There are many different algorithms to achieve synchronisation and most operate on a similar principle to this.

2.6 Limitations of VLC

Interference from light sources can be a limitation of VLC. In [8], the effect of different lighting conditions on a VLC OFDM link was investigated. There was not a significant difference between its performance in the dark vs in the presence of diffused sunlight. This is unsurprising as sunlight varies slowly and does contain any frequency components that could interfere with those of the transmission. However, when the fluorescent lighting was introduced, the system's BER deteriorated dramatically. A BER of 10^{-1} was achieved at a distance of 200mm with fluorescent lighting. In the sunlit room a BER of 10^{-1} was achieved at roughly 1400mm. The fluorescent lighting added a great deal of interfering noise due to its flickering nature. Clearly fluorescent lighting could be a significant issue for VLC links. As it is a popular lighting technology, its adverse effects should be kept in mind when designing a VLC system. Typical VLC links are designed expecting a LOS path between both ends [2]. This will not be the case if an object obstructs the LOS path. Although the signal can still reach the receiver through a reflection path, it will be significantly attenuated and more susceptible to ISI from other reflection paths as they will be of similar power.

Chapter 3

Simulation of OFDM on a Visible Light Channel

A simulation for an OFDM VLC system was built because, for reasons discussed in the Background Reading, OFDM is a very compelling modulation scheme for high-speed digital VLC systems.

3.1 Simulation Model

In the simulation the channel is based on the line of sight and reflection path equations discussed earlier in section 2.1. In MATLAB, functions were implemented to generate a channel's LOS and reflection path impulse responses. The functions take as parameters various LED, PD and reflection surface properties, as well as the angles and distances of the desired scenario. The functions return two values, the path's gain and its propagation delay. To build a model for channels made of multiple propagation paths, you first use the LOS and reflection path functions to generate the path gains and propagation delays of each path. With these values, the channel's complete impulse response can be assembled by combining the different paths gains together.

Paths that have different delays will appear as separate, scaled impulses in the channels impulse response. The gains of paths with identical delays contribute to the same scaled impulse in the impulse response. If the propagation delays of the paths differ, the impulse response will consist of multiple scaled and shifted impulses modeling the different delays. This allows us to model intersymbol interference. When running simulations parameters were taken from papers such as [11] & [8]. The digital symbols to be sent through the channel are linearly convolved with the impulse response generated. Additive white Gaussian noise is generated and added to the result of the convolution.

To demodulate, the simulation takes the samples corresponding to the OFDM symbols and discards the cyclic prefix. The cyclic prefix is discarded. The remaining noisy, attenuated samples are parallelized and passed as inputs to an FFT. Channel equalisation is now performed on the frequency domain outputs of the FFT. This is performed by dividing the frequency domain symbol by the FFT of the channel model. This aims to reverse any distortion the symbol acquired when it was convolved with the channel by passing through it.

The symbol received on each sub-carrier is determined by mapping each of the equalized frequency domain samples to the symbol in the QPSK constellation that they are closest to. The symbol is then mapped to the bit pattern it represents.

This is performed for many OFDM symbols at a range of distances. Received bits are compared to those originally transmitted, to count the amount of errors in each transmission.

3.2 Simulation of OFDM Performance in Different Lighting Conditions

OFDM modulation is used to transmit data though the simulated VLC channel. The implemented OFDM system uses 64 orthogonal sub-carriers each modulated with QPSK modulation. The VLC channel is setup to replicate the scenario tested in [8]. Following their assumptions, only a LOS channel is modeled. This results in a channel impulse response consisting of a single scaled impulse. This simulation tests BER performance in two lighting conditions, a dark room and a room with fluorescent lighting. In [8], ambient noise levels of both scenarios were measured. The dark room measured -71dBW. The fluorescent lighting measured -41dBW. White Gaussian noise is generated at these power levels and added to the convolved signal to model the environments. This simulation produces the BER plot in Figure 3.1, showing how error rates increase as the distance between transmitter and receiver grows.

The results of the simulation clearly show that system's range is greatly hindered by the presence of fluorescent lighting. The same bit error rates are achieved at greater distances, when transmission is in a dark room. The plot produced by this simulation follows a similar trend to the simulation produced in [8], the study upon which our parameters are based. However there is a major shift in the distances. In their simulation a BER of 10^{-1} was achieved at 20cm for fluorescent lighting and 170 cm in a dark room. In this simulation, the distances 5cm and 28cm achieved that same BER respectively. It is likely that the discrepancy in results is due to the channel model in this simulation being too pessimistic.

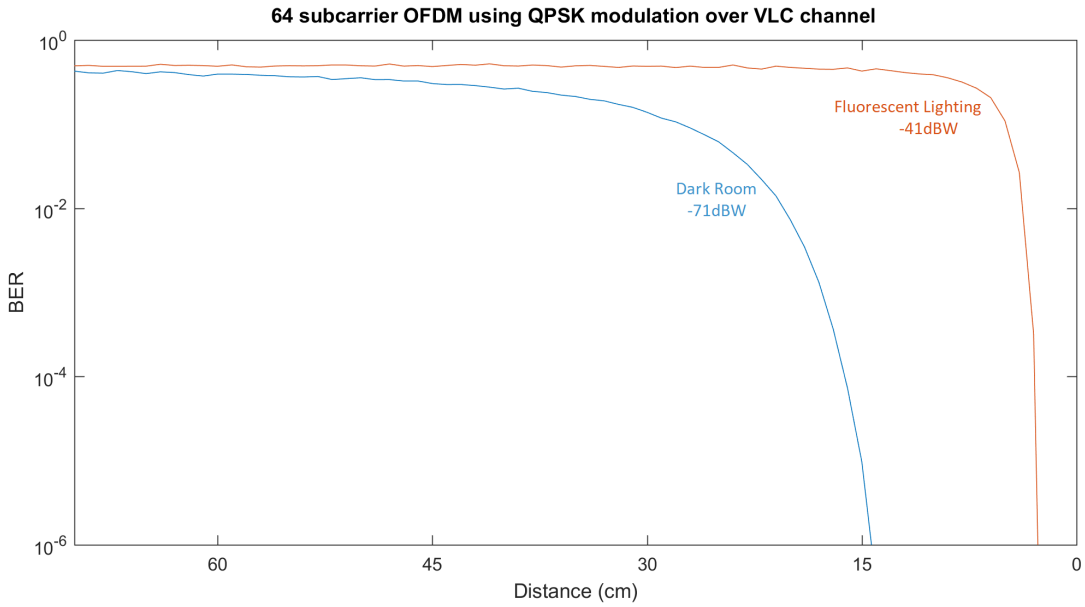


Figure 3.1: A plot of Bit error rate vs Transmission distance. Produced by a simulation of an OFDM system using 64 QPSK modulated sub-carriers, communicating over a modeled Visible Light channel. Performance is plotted for two different scenarios, a Dark Room and a Room illuminated by Fluorescent Lighting. The lighting scenarios are modeled as in [8], by adding noise to the received signal, where the power of that noise is based on values measured in similar lighting scenarios.

Chapter 4

Prototype Implementing Analog VLC

The goal of the project is to implement a digital VLC system. However, in order to develop an intuitive and practical understanding of communication through visible light, a basic analog VLC transmitter and receiver were designed and implemented. This analog system is capable of transmitting mono-audio. It was implemented on breadboards using off-shelf components.

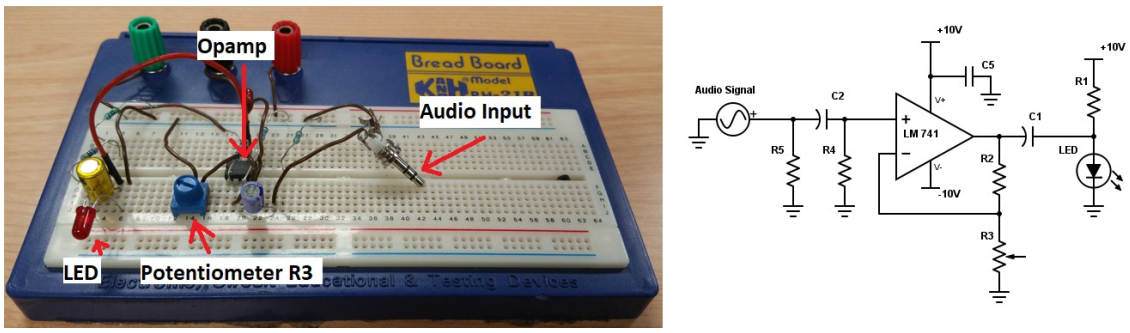


Figure 4.1: Breadboard implementation and corresponding schematic of the analog VLC mono-audio transmitter.

4.1 Analog Transmitter

The transmitter circuit can be plugged into a headphone jack. An input analog audio signal can be played in through this connection. This AC signal passes through capacitor $C2$ and is amplified by the LM741 op-amp. The purpose of C2 is to filter out any DC components of the input. It will also prevent any DC voltages present at the op-amp input from affecting the audio source. Resistors R5 and R4 act as pull-down resistors, keeping the DC voltage at either end of C2 at 0V and filtering out low frequencies such as the signal component due to ambient sunlight. If a DC voltage were allowed to form at the op-amp input it would be amplified along with the AC signal, producing a large voltage at its output. This could cause the op-amp to saturate and distort the audio signal.

The LM741 op-amp in conjunction with resistors R2 and R3 form a standard linear non-inverting amplifier. The amplifier gain is given by, $A_v = \frac{\mathrm{R2}}{\mathrm{R3}} + 1$. R3 is a potentiometer allowing the user to adjust the gain. The LED is biased using R1 so that it is turned on but not at peak brightness allowing room for the LED to be brightened and darkened by the signal. The amplified audio signal is superimposed on the LED's DC bias current. C1 prevents the LED bias from affecting the op-amp feedback loop. The Audio signal's modulates the LED's light intensity. As the amplifier gain is increased the signal on the LED becomes stronger. If the gain is increased too much the LED will begin to distort the audio waveform by flattening its peaks. Capacitor C5 is approximately 100nF. Its purpose is to decouple the op-amp supply to ground. Without it internal oscillation can degrade the op-amp's linearity. In this application the absence of C5 caused crackling in the received audio signal.

4.2 Analog Receiver

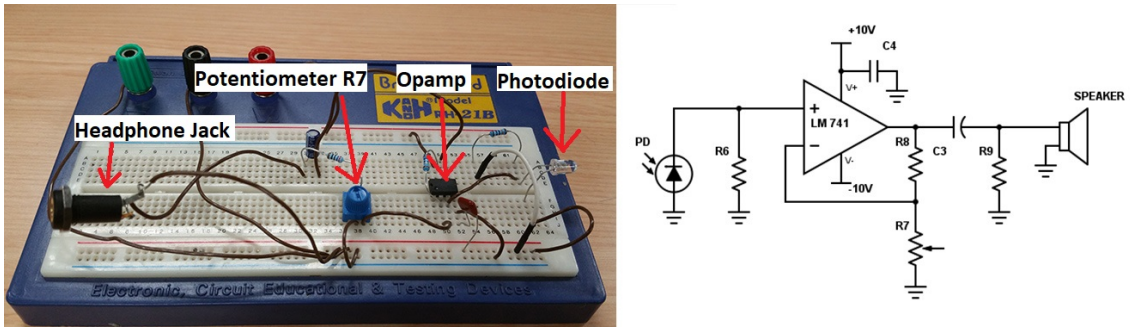


Figure 4.2: Breadboard implementation and corresponding schematic of the analog VLC mono-audio receiver.

The receiver is similar to the transmitter. Its input is the small photocurrent signal produced by the SFH203 PD. If the transmitter's light is visible to PD the transmitter audio signal will be present in this photocurrent along with noise. This signal is amplified by the same configuration as before. The amplified signal is played through speakers. In the circuit there is a headphone jack for headphones to be connected. Resistor R9 is used to pull-down the voltage at the headphone jack. This prevents charge build up and the consequent sound crack when a speaker is connected.

4.3 Results

The design worked quite well. It was tested by playing music into the transmitter's input and listening with headphones at the receiver. When LED and PD where facing each other, you could hear the music in the headphones. Sound quality had notably degraded in particular in the low frequency end. Increasing the distance between both ends reduced the volume of the received audio. This could be compensated for, by adjusting the gain on either end. Audio was perfectly recognisable at a distance of over 1m. These are very promising results for such a simple design. They show the potential VLC has. The performance did not noticeably change in a dark vs

a sunlight room. However, performance was degraded by noise when LED based lighting was turned on. Using an oscilloscope to measure the PD output directly, noise produced by the lighting was identified as a large 100Hz component and less significant harmonics of it. The oscilloscope output is shown in Figure 4.3. This noise is caused by the flickering nature of the overhead LED lighting which runs off a fully rectified 50Hz mains supply.

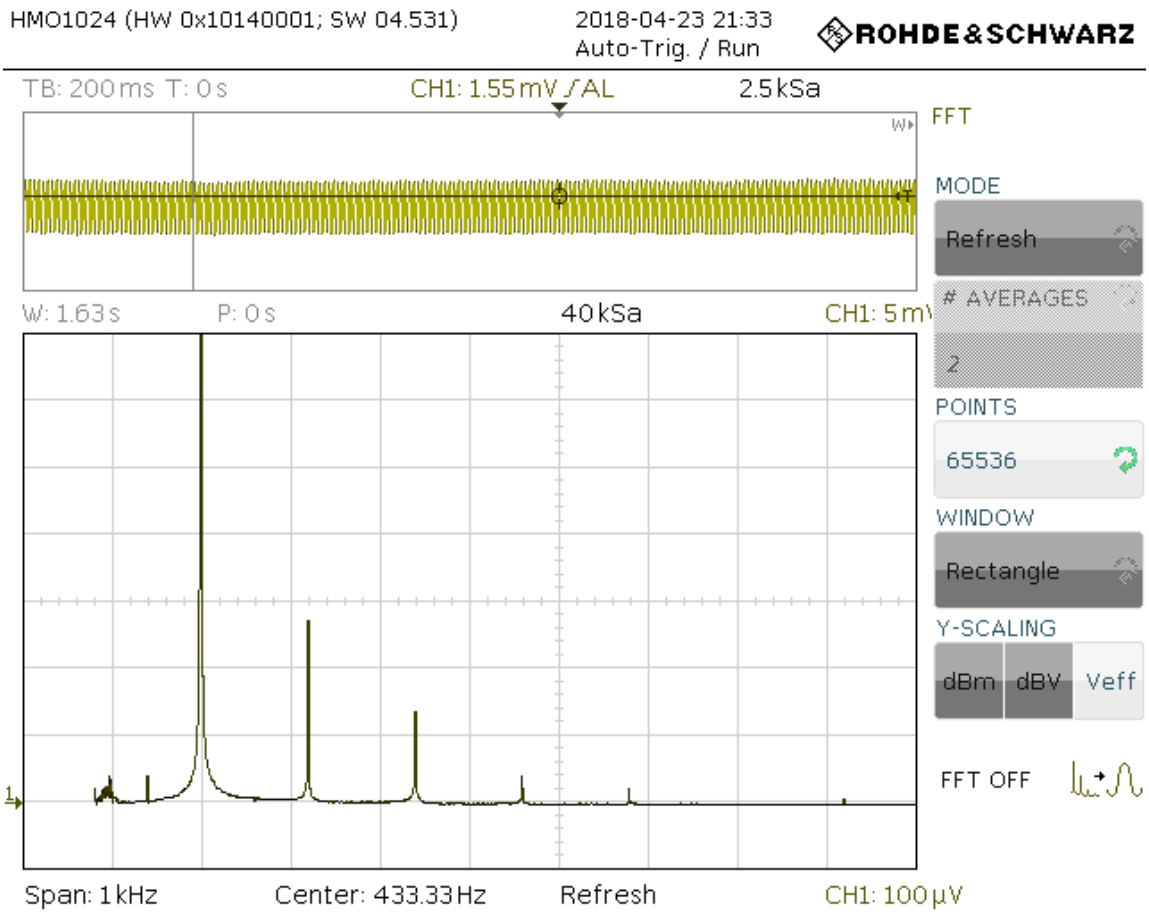


Figure 4.3: Oscilloscope Screenshot showing the frequency spectrum of the signal produced by photodiode in our testing environment. It shows significant noise at 100 Hz and at some of its harmonics, caused by overhead lighting. Measurement was taken while no VLC transmission was occurring.

Chapter 5

Proposed Arduino Based Digital VLC

5.1 Digital Modulation Scheme

5.1.1 Proposed On-Off Keying Based Scheme

The designs employ a variation of OOK as their modulation scheme. OOK was chosen due to it simplicity and consequent low cost. In this scheme two symbols can be sent the first representing a 1, the other a 0. To send a 1 in this variant, a square wave is transmitted at a chosen carrier frequency for one symbol duration. To send a 0 nothing is transmitted for one symbol duration. The symbol duration and carrier frequency are fixed for a given scheme. Shorter symbol durations directly translate to higher bit rates, as each symbol represents one bit. However, schemes with shorter symbols are observed to be more sensitive to synchronisation inaccuracies and require resynchronisation more frequently, which introduces an overhead. Shorter symbols will create a baseband signal that uses a larger bandwidth. In order to transmit this a larger bandwidth must be available around the carrier frequency. In these designs the symbol duration is chosen to span 10 cycles of the carrier frequency.

Many communication systems in particular wired ones use similar variants of OOK however they chose a carrier frequency of 0 Hz. In other words a 1 is transmitted simply by turning on the LED (or applying the supply voltage in the case of wired communication) for one symbol duration, without a utilizing a changing waveform. Although this is significantly simpler, it was opted against. These designs utilize carrier frequencies such as 245Hz and 980Hz. It would be otherwise very difficult to distinguish between variations due to the transmission, and those due to ambient light. There is a great deal of DC noise in a the visible light spectrum. This is caused by ambient light. The intensity of this slowly varying light ($< 1Hz$) is unpredictable and it would be difficult to determine whether a change in light intensity seen by the PD occurred due to a transmitted symbol or due a change in the environment such as a cloud moving. By modulating a carrier frequency with symbols, a system is created that is impressively robust. An ideal carrier frequency is one around which there is a bandwidth with low noise.

At the receiver a bandpass filter is applied that greatly attenuates all frequencies outside the bandwidth used by the transmission. In this project the filter was implemented as a digital filter in software, for its easy of implementation and flexibility. It can however be implemented as analog circuit. With this filtering, the receiver's performance should not be affected by slow changing ambient light as it only looks for the energy in a specific bandwidth around the carrier frequency. The ubiquitous IR remote NEC protocol operates on a similar principle using a carrier frequency of 38 kHz [16].

5.1.2 Proposed System for Synchronisation

The sequence of binary data to be sent is used to generate a sequence of symbols. These symbols are sent in frames. Synchronisation is performed at the receiver at the start of each frame. Without it, the receiver would not be able to determine when a symbol begins and ends. The synchronisation technique designed for this implementation is influenced by the method used in [8].

Synchronisation is achieved by prefixing each frame of data with a pilot sequence known to the sender and receiver. The pilot sequence chosen in the implementations is 0, 1, 0. The receiver keeps track of the energy received over the last symbol duration. The energy in this window is recalculated as each new sample arrives. It will be low when the initial 0 symbol is fully received, as all samples in the window will be close to zero. The energy of the window will then grow at a fixed rate as the samples corresponding to the 1 symbol arrive. The energy of window will then decrease at a fixed rate as the second 0 arrives. A plot of the window's energy value changing as the pilot sequence is received during an experimental transmission is shown in Figure.5.11.

When the energy of the window has decreased by a significant amount, the receiver will look back on past windows to identify the position of peak energy. This peak occurs when window was perfectly aligned with the 1 symbol. From this it can determine when the first data symbol of the frame will occur. Thus synchronisation is achieved.

There will always be inaccuracy in synchronisation. Overtime this inaccuracy can grow to the point where symbols are misinterpreted as we no longer predict symbol boundaries correctly. This is desynchronisation. To remedy this the frame size is set to an amount of symbols for which synchronisation can be reliably held. After a frame is received the receiver resynchronises with the next frame's pilot sequence. With this setup, data can be transmitted continuously and indefinitely while maintaining consistent performance.

5.2 Real-Time Implementations of the Proposed System

5.2.1 Overview of Contributions

In this section two digital system implementations are discussed in detail.
In both implementations an Arduino UNO micro-controller is used to modulate the transmitting LED and another is used to sample the receiving PD.
A basic 5mm Red LED and the SFH203 PIN Photodiode were used. The photocurrent produced by a PD is amplified using the circuit in Figure 5.1 prior to sampling.

1. The first implementation sends ASCII strings from one Arduino to the other using a carrier frequency of 245.1 Hz. All processing of samples is performed on the receiving Arduino which samples at 1 kHz. This includes filtering, synchronisation and symbol decision. This system is independent of external computers and is therefore a low cost system.

2. The second implementation sends generic files through visible light using a carrier frequency of 980.1 Hz. In this implementation, the receiving Arduino was only used as an ADC. Samples are forwarded to a laptop via serial port in real-time where all other signal processing is done. The Arduino thus had little processing to perform allowing a much higher sample rate of 9.61 kHz. The purpose of this implementation is to overcome limitations of the first. Limitations due to the Arduino's lack of processing power and memory greatly hindered efforts to improve the system using signal processing techniques. In this second implementation however, signal processing is delegated to an external computer creating an environment in which complex calculations can be performed on incoming samples in real-time. This creates an environment ideal for future research.

5.2.2 Arduino UNO

The Arduino UNO is a very popular low cost micro-controller. It was chosen as the receiver and transmitter micro-controller for these implementations for the following reasons;

- Arduino UNO boards are relatively inexpensive priced at €20.
- Arduinos are well known and widely used.
- The board has outputs supporting PWM at various frequencies, ideal for generating the carrier frequency.
- The board has a 10-bit analog to digital converter whos sampling can be controlled by hardware timers at a desired frequency. This produces samples that are taken at very precise intervals which is essential for digital filters to perform as intended.
- The board supports serial port communication with an external computer via USB at speeds upto 115200 baud.
- The Arduino IDE facilitates developing and debugging C code for the Arduino. It also allows you to plot values reported by the Arduino in real time.

There are limitations with the Arduino however. Its ATmega328P processor is slow. This becomes a particularly significant issue when processing is done on samples in real time. At one point during the development of the first implementation, the processing performed on each sample was made more complex, this resulted in the sampling intervals becoming very inconsistent. This occurred because the processor unable to perform the increased number of instructions in the interval between samples. To fix this it was necessary to reduce the sampling rate which ultimately limited the data rate.

The Arduino also suffers from a limited SRAM of only 2 kB. This greatly limits the amount of variables one can use. This could potentially be an issue as digital

signal processing often uses many past values to perform calculations. 2kB of memory can only hold 1024 integers or 512 floating point numbers. Flash memory is used to store the machine code of the program. It is limited to 32 kB meaning there is a notable limit on the size of an Arduino program. This limitation never became relevant in this project.

Code developed in this project for the Arduino was specifically implemented with efficiency in mind. Variable types were carefully chosen. Bytes were used instead of integers where 256 values was sufficient. Floats were avoided whenever possible as the ATmega328P does not have a floating point unit. Floating point operations on the Arduino are performed slowly, using a floating point software library.

Since a prototype was being developed, high data rates were not deemed essential for the project making the limitations of Arduino acceptable. The advantages of the Arduino in cost and ease of use are very compelling over alternative micro-controllers.

5.2.3 Proposed Implementation of a 5V Single-Supply Linear Transimpedance Photodiode Amplifier

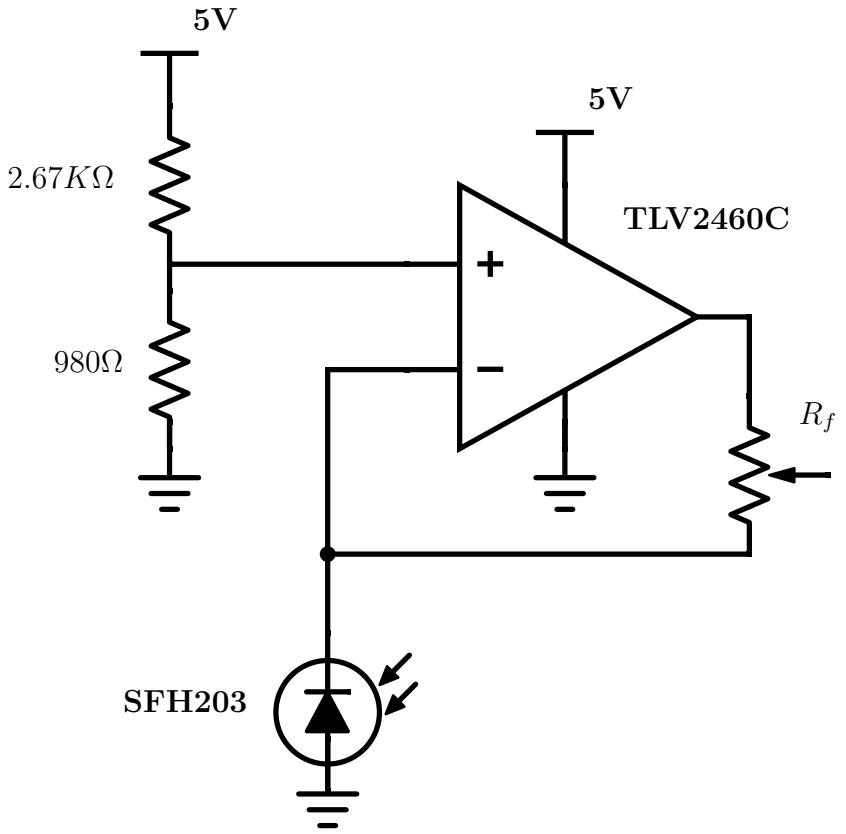


Figure 5.1: Single supply transimpedance photodiode amplifier powered entirely by an Arduino UNO's 5V pin. Transimpedance gain = R_f. Output voltage is taken at the voltage output of the op-amp.

The purpose of this amplifier is to linearly amplify the photocurrent produced by the PIN photodiode used at the receiver. The photocurrent produced in scenarios relevant to this project were observed to typically be around $2\mu A$. The component of this photocurrent that corresponds to the carrier wave of the transmission is also expected to be small, growing weaker as the distance between receiver and transmitter grows. The output of the amplifier is passed as an input to the Arduino's analog to digital converter, who's full range is 0 - 5V. As the signal produced by the unamplified photodiode is very small it is crucial to amplify it prior to sampling. As shown in our results, the transmission range achievable using amplification is up to

900 times larger than that achieved with no amplification.

A photodiode can be modeled as a light intensity controlled voltage source, or a light intensity controlled current source. The latter model was considered when designing this amplifying circuit. This choice was made because the current response of photodiodes is typically very linear over a wide range of light intensity [17], a trait not equally present in its voltage response.

Opting for a conventional approach, the amplifier class was chosen to be transimpedance, allowing the input photocurrent to be amplified and converted to a voltage output, ideal for the Arduino ADC. The next important design choice was the decision to ensure the amplifier could be powered entirely by the Arduino's 5V supply. This is the alternative to being powered by separate power supplies, achieving positive and negative voltage rails, as used in Chapter 4 to create +10V and -10V. Although this constraint made the design more difficult, as it is not particularly common, the system would greatly benefit from it. With an amplifier solely powered by the Arduino, the final product would be very flexible and easy to setup. The entire receiver and transmitter systems would be powered by simply connecting the Arduino to a computer's USB port. The design would also be more efficient in terms of power usage.

The main component in the transimpedance amplifier is an op-amp. The modern TLV2460C op-amp was chosen, primarily because unlike many op-amps, it can operate using a single 5V supply. Another reason for choosing this device is its decent Gain Bandwidth Product of 6.4MHz. Allowing 6.4 times more gain for a particular input bandwidth than the LM741 used in this project's Analog implementation. Having a large gain bandwidth product prevents the amplifier from limiting the modulation scheme's carrier frequency and symbol rate.

A schematic of the final amplifier design is shown in Figure 5.1. In the circuit the photodiode is operating in photo-conductive mode, meaning there is a reverse bias voltage established across the photodiode. When operating in photo-conductive

mode the photodiode's response time improves because the reverse bias increases the width of the depletion region in the semiconductor, which decreases the junction capacitance. Photodiode sensitivity is also increased by the reverse bias. Incident photons form a electron-hole pairs, but only those formed in the depletion region tend to contribute to the photocurrent, as they can be separated by the strong electric field present there. The increase in width of the depletion region, caused by the reverse bias, increases the volume of photons contributing to the photocurrent [18].

Feedback is implemented in the circuit by the current flowing through R_f. With this negative feedback, the circuit tends to keep the voltage at the negative input equal to that at the positive. The positive input is fixed at $\frac{5\times980}{980+2760} \approx 1.34V$, meaning the feedback converges toward keeping the negative input at that voltage, establishing the reverse bias across the photodiode.

The photocurrent flows towards ground, and increases in this direction with light intensity. The op-amp output voltage is such that the current flowing through R_f approximately matches the photocurrent, thus keeping the negative input at the same voltage as the positive. The output voltage is simply the product of the photocurrent and R_f. Thus the transimpedance gain is simply $\frac{V_0}{I_{photo}} = R_f$. The photocurrent signal that we are amplifying is always DC, meaning it never changes polarity, which is why we have no need for a negative supply rail. The voltage output will never want to go negative.

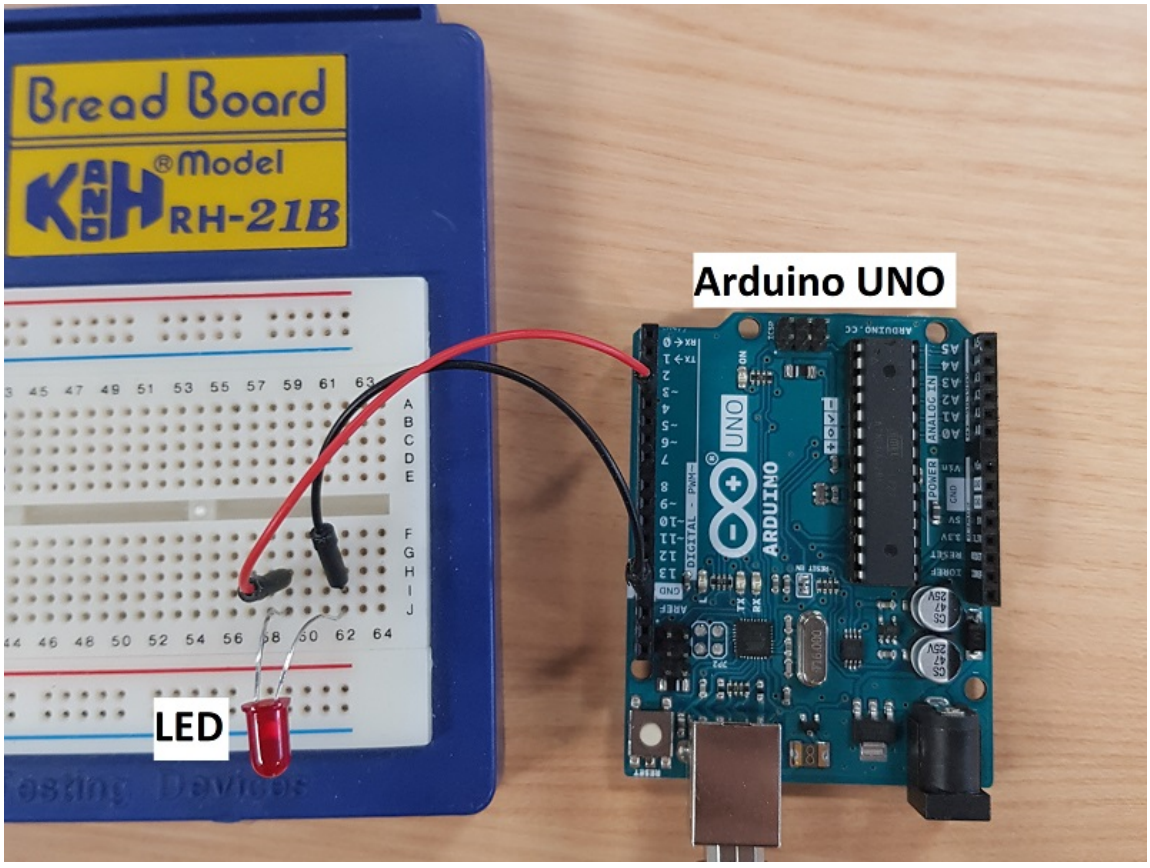


Figure 5.2: Photo of the Arduino UNO in the implementation of the Digital VLC transmitter used in Systems 1 & 2.

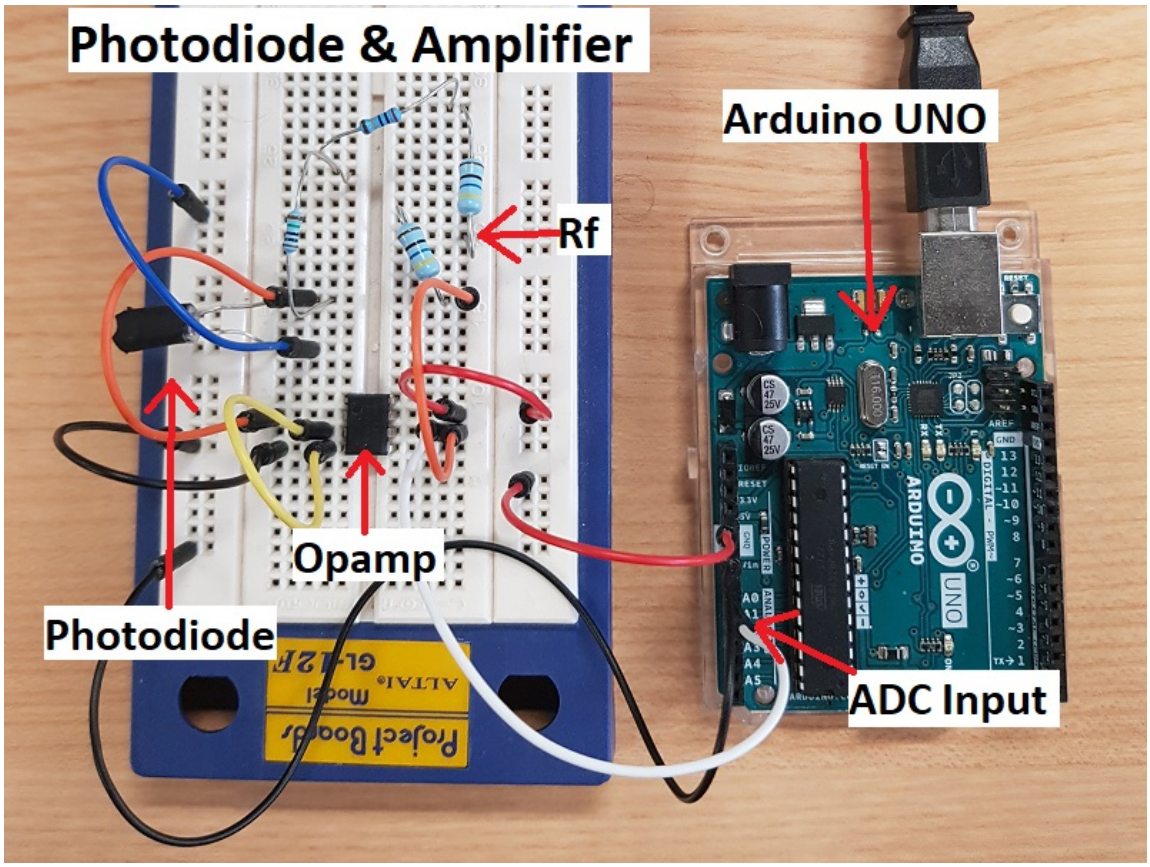


Figure 5.3: Breadboard implementation of the photodiode & amplifier from Figure 5.1. The amplified signal of the photodiode is connected to the ADC input of the Arduino UNO. This circuit is used as the Digital VLC receiver in Systems 1 & 2.

5.3 Proposed System 1, (Arduino Based ASCII Transmitter)

5.3.1 Transmitter

Transmitting a 1 symbol requires a square wave at the chosen carrier frequency. To do this accurately, the Arduino's pulse width modulation(PWM) hardware on pin 3 is used with a 50% duty cycle. The frequency of the square wave produced is adjusted to 245.1 Hz by modifying the Timer 2 pre-scaler in register TCCR2B. To send a 0 symbol the duty cycle is simply set to 0% effectively fixing the output at 0V.

After the duty cycle of the next symbol is set a delay of 41ms (the symbol duration) is called. This delay only affects the software loop, the PWM output driving the LED continues in the background. When the delay ends the symbol has been sent. The program immediately repeats this process for the next bit. An example of the waveform modulating the LED is illustrated in Figure 5.4.

Strings to be sent are split into frames of 39 characters/bytes. The transmission of each begins by sending the pilot sequence 0,1,0. The software then iterates through each bit in every character in the frame. These bits are mapped to symbols and

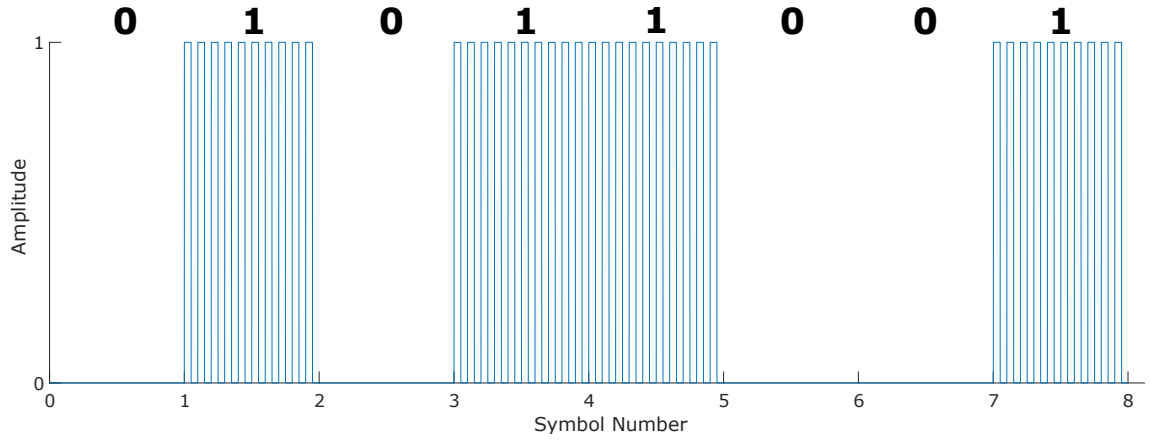


Figure 5.4: Illustration of OOK waveform with 254.1Hz carrier frequency. The data being sent consists of the pilot sequence, (0, 1, 0) followed by the data bits, (1, 1, 0, 0, 1).

transmitted. The end of the frame is marked by transmitting a null character which is a byte of zeros.

This feature of the protocol benefits the whole system. The receiver does not need to know in advance the length of a frame allowing it to be varied at any time. Another advantage of this design is that it allows the system to recover from the transmission errors that occur when the LOS between transmitter and receiver is momentarily blocked.

If the block occurs the receiver will interpret what it sees as zeros due to the absence of signal. As a result the byte that was in mid transmission at the time of the blockage will be corrupted. The latter part of its bits will be received as zeros. Then, every bit of the following byte will be received as zeros, which is interpreted as an end of the frame. The receiver now ceases interpreting samples as data. Instead it looks for a pilot symbol of the next frame. Assuming a different frame is being transmitted when the block is removed, the receiver will interpret the first 1 symbol received as the 1 in the pilot sequence. This will likely lead to a few misinterpreted frames because the receiver will be misaligned with the bytes of the following frames, incorrectly assuming where they start and end. However the pilot symbols at the start of the of each frame cause the extent of the misalignment to change on each new frame. It eventually realigns and communication is restored. This is described in detail in the following example.

5.3.2 Example: Recovery from Temporary Blockage

This example serves to illustrate how the design choice described in the previous section allows the system to recover from temporary communication loss.

Suppose a frame of size of N. Where each frame is prefixed with the pilot sequence.

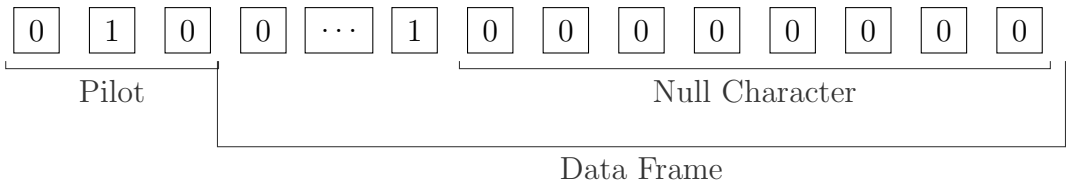


Figure 5.5: Structure of Transmitted Frame.

The end of the frame is null terminated in the same fashion as a C string. It follows that when the receiver sees low energy in the transmission bandwidth for the 8 symbol durations of a byte, it deduces that the frame has ended. This scenario will also occur if the visible light channel between sender and receiver becomes blocked.

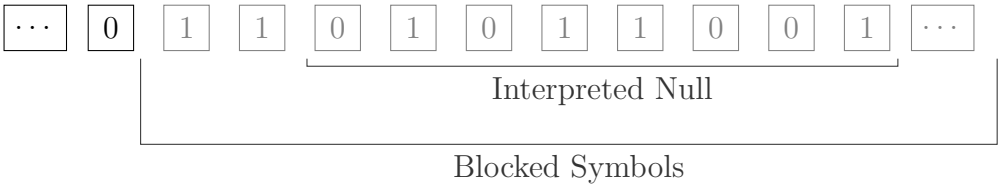


Figure 5.6: Segment of a data frame at the moment where a LOS blockage occurs.

In Figure 5.6, *Blocked Symbols* are the symbols that were transmitted at the time of the blockage. They are received as 0s. The last two bits 1,1 of a byte were being transmitted when the blockage occurred. That byte was corrupted since they were received as 0,0. The receiver continues on to receive the next byte. It is received entirely as 0s, forming a null character. To the receiver, this byte signifies the end of frame. It now ceases to interpret what it receives as data and looks for a pilot sequence marking to start of the next frame. This continues until the end of the blockage.

Figure 5.7 shows how the first unblocked 1 symbol seen by the receiver is incorrectly interpreted as the 1 of a pilot sequence. In this example the start of the interpreted

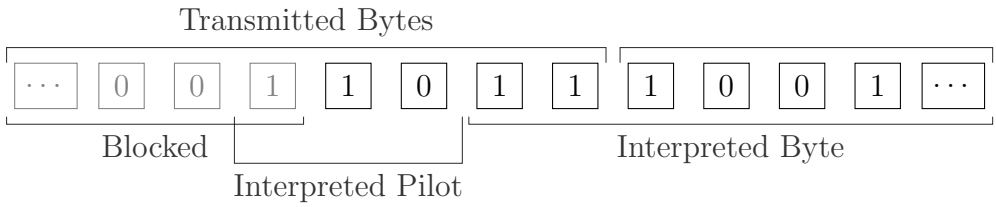


Figure 5.7: Segment of a data frame at the moment where a LOS blockage is removed. Misalignment of receiver is illustrated as byte boundaries do not match interpreted boundaries.

data frame, i.e. what the receiver interprets as the the start of a byte, is misaligned by 2 bits with one of the transmitted bytes. This 2 bit misalignment will persist for every byte in this frame corrupting it entirely. We will refer to this as a misalignment of -2.

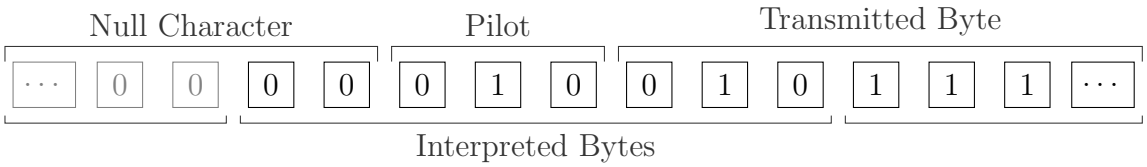


Figure 5.8: Illustration of how an end of a frame is missed when the receiver is misaligned with the transmission. This section shows how the misalignment of -2 becomes a misalignment of -5.

In Figure 5.8 we see the effect the -2 misalignment has on the end of frame. As shown, the null character is not seen by the receiver because its eight 0s are received across two separate bytes. Consequently the receiver, not realising the frame ended, interprets the pilot as bits of the next byte. However, because the pilot sequence is 3 bits long, the misalignment is now -5.

At the end of this frame, the next pilot will cause the misalignment to shift by -3 again. Our new misalignment will be of -8. Now the start of the next byte will coincide with where the receiver thinks a byte starts. Realignment has occurred. From this point on, the receiver will remain in alignment with the transmission, unless another block occurs.

In practice it is likely that the first frame after alignment will also be corrupted

Initial Misalignment	Frames Corrupted
-1	5
-2	2
-3	7
-4	4
-5	1
-6	6
-7	3

Figure 5.9: List of all possible receiver misalignments, and the frames that will be corrupted while the system realigns.

as the receiver may have desynchronized by that stage, since it will not have had an opportunity to resynchronise for several frames.

In conclusion; when a temporary block in the line of sight between transmitter and receiver has ended, the receiver will likely be misaligned with the transmitted bytes. The extent of this misalignment changes by -3 at every new frame. Regardless of frame size, the requirement for realignment is that the misalignment increases to an integer multiple of -8. All frames that are sent during this phase of misalignment are corrupted. Table 5.9 lists all possible misalignments values and the amount of frames that will be corrupted in each case.

5.3.3 Receiver

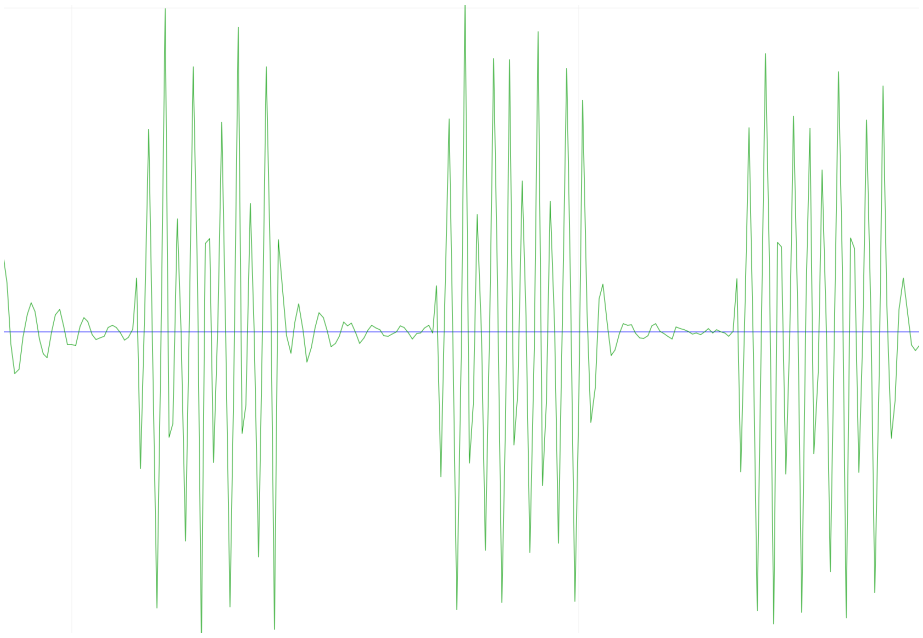


Figure 5.10: Plot of ADC samples after passing through a 5^{th} order high-pass IIR filter implemented in software running on an Arduino UNO. The Receiver is seeing an OOK modulation of a 245.1Hz carrier frequency, transmitting the symbol sequence (0, 1, 0, 1, 0, 1) in the presence of normal ambient light. This plot demonstrates how effective the filtering is at removing interference from ambient light.

The receiver samples at a rate of 1 kHz. Sampling is performed in an interrupt service routine that is invoked by hardware Timer 1 at the desired frequency. These samples are passed through a 5^{th} order high-pass IIR filter designed in MATLAB to have a cut off frequency of 150Hz. The performance of this filter in practice is demonstrated in Figure 5.10.

The output of this filter sits around 0 regardless of changes in ambient light. The output is only seen to change significantly when the sender is transmitting a 1 symbol. An IIR filter is chosen over FIR as it achieves good filter specifications with far fewer coefficients [19]. A high-pass filter was chosen over a bandpass as it would work sufficiently well and requires fewer coefficients. This is significant as the Arduino is quite slow at floating point calculations and has limited memory. The lack of processing power on the Arduino is also why a higher sampling rate was not chosen.

5.3.3.1 Implementation of Proposed Synchronisation

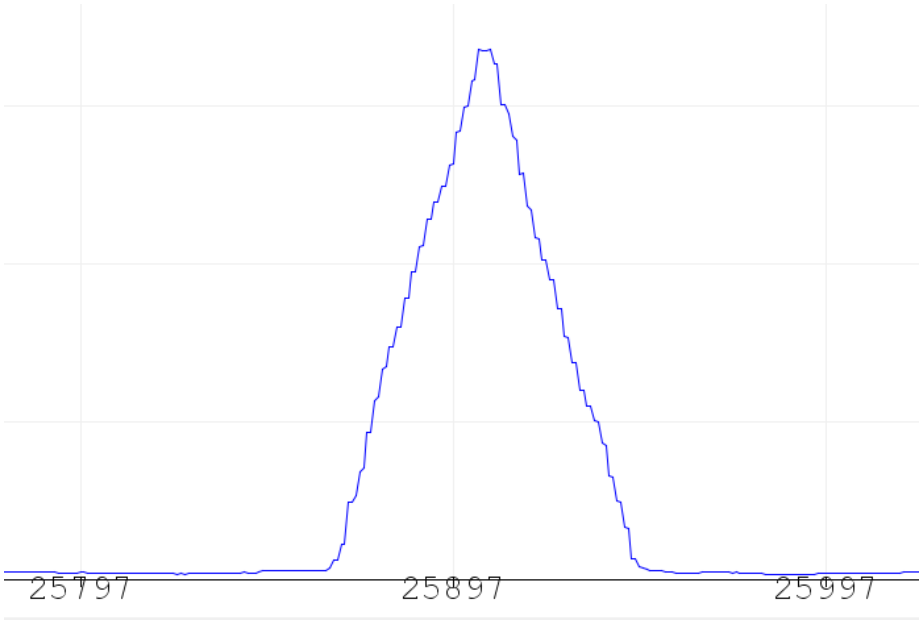


Figure 5.11: A plot showing the change in window energy as a pilot symbol arrives at the receiver. The window energy is the energy of the last 41 filtered samples. The window energy grows as samples of the 1 symbol of the pilot arrive and falls as the subsequent 0 symbol arrives.

After each new sample is filtered, the receiver calculates the energy of the last symbol window (41 samples).

$$E_{window} = \sum_{n=0}^{40} x[n]^2 \tag{5.1}$$

Where x vector of past samples.

The Arduino tracks the energy of the 41 most recent windows (where each window is 41 samples). The receiver uses this to identify the arrival of the pilot as described in section 5.1.2. Figure 5.11 plots the window energy observed as a pilot symbol is received. When the most recent window energy falls above a hard-coded threshold, we assume the 1 symbol of the pilot is arriving. The window energy will continue to grow until it peaks at the last sample of this symbol. After the peak, the window energy decreases at a steady rate due to the arrival of the second 0 symbol. The

software waits until the latest window energy falls below 90% of the original threshold to ensure the peak has definitely been received. At this point we searches through the array of past window energies to locate when the peak occurred.

The sample directly after the window of peak energy corresponds to the first sample of the second 0 symbol in the pilot. Knowing this the receiver determines start of the first data symbol. After discarding the remaining pilot samples, all subsequent samples are divided up into groups of 41. The system then interprets each group as one symbol. If the energy of a symbol exceeds the hard-coded threshold, the symbol is mapped to a 1 and otherwise to a 0.

The value of the threshold may need to be tweaked if the signal to noise ratio changes substantially though it tends to be flexible.

The synchronisation achieved by this scheme is not perfect. Slight discrepancies in timing tend to grow as time passes. The current implementation cannot reliably send more than 40 bytes of data without needing to resynchronise. In order to overcome this, data can be sent in frames consisting of 39 bytes prefixed with the 3 bit pilot sequence. The receiver can then resynchronise at the start of each frame.

5.3.4 Data Rate

This current implementation cannot reliably send more than 40 bytes of data without needing to resynchronise. In order to overcome this the frame size was chosen to be 39 bytes.

The bit rate of this system is;

$$\frac{8 \times 39_{bytes}}{(3_{\,pilot} + 39_{bytes} \times 8) \times 41\text{ms}} \approx 24\,\text{bit/s}$$

Higher bit rates are possible with shorter symbol durations. However in order to significantly reduce the symbol duration it is necessary to increase the carrier frequency, making a larger bandwidth available.

The Arduino does not possess great deal of processing power. It has an 8-bit, 16MHz ATmega328P processor with no floating point unit. Since the software implementation of the IIR filter involved many floating point multiplications per sample, the highest rate the Arduino could filter incoming samples at was approximately 1 kHz. When measuring a signal it is good to sample at at least four times the highest frequency of the desired signal. With the sampling rate limited to 1 kHz, 245.1 Hz was chosen as the carrier frequency.

5.4 Proposed System 2, (Arduino & Computer Based File Transmitter)

This second system allows one computer to transfer any file to another computer via visible light communication. The modulation scheme is similar to that implemented for System 1 with some modifications. In System 2 there is a computer wanting to send a file, it is connected via serial port to the transmitting Arduino. This Arduino receives a stream of data via this serial port. It handles mapping these received bits to their corresponding symbols, modulating a 980.39 Hz carrier square wave with them, and transmitting the resultant waveform by intensity modulating an LED.

The waveform transmitted by the LED's light is sensed by a PD at the receiver along with unwanted signals from ambient light and noise. This is then amplified by the transimpedance amplifier in Figure 5.1. The amplifier's voltage output is connected to the receiving Arduino's ADC input. Software implemented to run on the receiving Arduino handles sampling its ADC at 9.615 kHz and forwarding these samples to the receiving computer via serial port in real-time. The receiving computer collects ADC samples from the Arduino and processes them in real time, synchronising with, and demodulating the transmission to recover the data sent, assembling it into a file.

5.4.1 Assembling Frames from the Outgoing File

A script named "python_binFile_sender.py", implemented in Python, runs on the transmitting computer. It takes the path of the file to be sent, as its only input. It establishes a connection with the transmitting Arduino via serial port, using the pySerial library.

In system 1, the frame size was chosen to be 39 bytes in length. In this system it is reduced to 15 bytes because desynchronisation tends to occur more quickly. This is likely caused by the higher frequency of the carrier wave and the shorter symbol duration of 10ms (vs 41ms in System 1).

The frame structure is similar to that in System 1. Each frame is prefixed with a pilot sequence, however that is inserted by the Arduino, so the python script does not worry about it. The end of the frame is denoted by a null character, a byte of zeros. This protocol was sufficient for System 1, as the data to be transmitted consisted only of ASCII characters, thus a null character always marked the end of frame, it would never be sent as data. Since System 2 must be capable of sending any type of file, the data bytes are not limited to ASCII values. This means zero bytes may be contained in the data we are transmitting, and the receiver must not misinterpret them as the end of the frame. There must also be a way for the transmitter to indicate to the receiver that the transmission has ended, that they have received the file in its entirety. To address these issue the following additions were made to the protocol.

Three control bytes were defined:

- End of Frame(EOF), 0x00
- End of Transmission(EOT), 0x04
- Escape(ESC), 0x5C

EOF, is the null character marking the end of the current frame, this is carried over from the protocol in System 1. EOT, is a byte that marks the end of transmission.

When the receiver sees this byte, it knows the entire file has been received, it can finish writing to the output file. The file to be transmitted may contain bytes that have the same value as the control bytes. This is an issue when we are transmitting them. We do not want bytes from the file to be interpreted as control bytes. The purpose of ESC is to solve this issue. This script inserts an ESC byte immediately before any data byte that shares its value with one of the control bytes. When the receiver sees an ESC, it first discards it as it is not part of the data. It then knows to treat the next byte as a data byte even if it happens to have the same value as a control byte. When the sender wishes to send a data byte with the same value as ESC, it is also escapes it by sending another ESC byte before it. The byte after the second ESC is not escaped as the that second ESC is merely interpreted an ordinary data byte. For further clarity see the example 5.4.1.1 below.

5.4.1.1 Example: Converting sequences of data into frames

In this example we consider 9 data bytes that have been read from the outgoing file. For simplicity we take the desired frame length to be 6. From the 9 data bytes we will build frames, following the algorithm implemented in the python script. Note that the EOF byte marking the end of a frame is not counted towards frame length.

0	1	2	3	4	5	6	7	8
32	AC	3A	*00*	24	90	92	*5C*	22

Figure 5.12: Hex values of data bytes from the outgoing file. Bytes 3 and 7 have the values of the EOF and ESC respectively, and must be escaped to prevent misinterpretation at the receiver.

0	1	2		3	4	
32	AC	3A	**ESC**	*00*	24	**EOF**

Figure 5.13: Hex values of the first frame, formed using the bytes 0 - 4 from Figure 5.12. ESC bytes are inserted in front of data bytes that have special control values.

5	6		7	8	
90	92	**ESC**	*5C*	22	**EOT**

Figure 5.14: Hex values of the second frame, formed using the bytes 5 - 8 from Figure 5.12. ESC bytes are inserted in front of data bytes that have special control values.

The algorithm builds the first frame by iterating through the data bytes shown in Figure 5.12. It encounters byte 3 and finds that it has the value 0x00, which is reserved for the EOF control byte. To prevent misinterpretation at the receiver, the value is escaped by inserting an ESC before it. The algorithm continues to iterate until the frame length is 6. At this point the EOF control byte is appended and the frame is ready to be transfered to the transmitting Arduino via serial port.

The second frame is built is the same fashion as the first. Byte 7 has the same value as ESC. When it is encountered it is escaped, leaving two consecutive ESC bytes. The second is only interpreted as data at the receiver. The algorithm reaches the end of the data before the second frame has reached the standard frame size of 6. As this is the end of the data to be sent, an EOT byte is appended to the end of the frame. When an EOT is being used it is no longer necessary to mark the end of the frame with an EOF.

5.4.1.2 Forwarding Frames to the Transmitting Arduino

Frames are sent to the Arduino via serial port. They are written byte by byte using the write function in the pySerial library. When a frame has been fully written to the Arduino the python script waits, reading from the serial port until the Arduino replies with the character "E", indicating that it has finished transmitting the previous frame through visible light. It is now ready to receive the next frame from the python script.

The first frame of the transmitter consists only of the name of the file being sent. The receiver knows to use this first frame to name the file it creates.

5.4.2 Transmitting Frames

The transmitting Arduino receives frames and transmits them prefixing them with the pilot. It is not concerned with the significance of the data it sends. It is not concerned with which transmission a frame belongs to.
Software implemented on the transmitting Arduino reads bytes from the serial port and stores them in an array. When an unescaped EOF or EOT byte is read, the program stops reading from the serial port, and proceeds to transmit the received frame through the visible light channel.

A 980.39 Hz square wave is applied to the pin connected to the LED using the Arduino's PWM hardware in the same fashion as in System 1. This square wave is switched on or off by setting the duty cycle of the PWM to 50% or 0% respectively. First, the pilot sequence is transmitted, then the bits in the frame. Symbols are produced from the bits in the frame and transmitted exactly as in System 1, but with a shorter symbol duration.

When the frame is transmitted, the Arduino writes the character "E" to the serial port indicating to the computer that it is ready for the next frame. The values of the next frame overwrite the old frame in memory.

5.4.3 Sampling the Amplified Photodiode

The voltage output of the receiving photodiode is connected to the Arduino's ADC input.

The receiving Arduino samples this ADC at a precise rate of 9.61538 kHz. This is achieved by configuring registers ADCSRA, ADCSRB and ADMUX to cause the ADC to continuously sample at a timer controlled interval. When a ADC sample is ready an interrupt service routine (ISR) is invoked. The instructions implemented in the ISR write the sample to the receiving computer via serial port. The ADC has 10 bit accuracy however we only relay only the most significant byte (register ADCH) to lighten the load on the serial connection that has a limited data-rate of 115200 bits/s, losing 2 bits of accuracy in each sample.

5.4.4 Software Demodulation, Assembling the Received File

A script named "python_receiver_binFiles.py", implemented in Python, runs on the receiving computer. It reads ADC samples from the receiving Arduino via the serial port. It then processes these samples aiming to output the file that was transmitted to it.

An 80^{th} order bandpass FIR filter is implemented in the script. The filter coefficients were generated using designfilt in MATLAB with the following parameters.

```
h = designfilt('bandpassfir', 'FilterOrder', 80, 'CutoffFrequency1',
980-500, 'CutoffFrequency2', 980+500, 'SampleRate', 9615);
```

This produces filter with a passband of 1 kHz centered at the carrier frequency 980 hz, designed for the sampling rate of the ADC. The magnitude plot response of this filter is presented in Figure 5.15. The ambient light in our open plan office environment had no significant noise at frequencies above 150 hz. Knowing this, the bandwidth of the filter was chosen to be large, while ensuring frequencies below 150 hz are significantly attenuated. The issue with a narrow bandwidth is that it limits

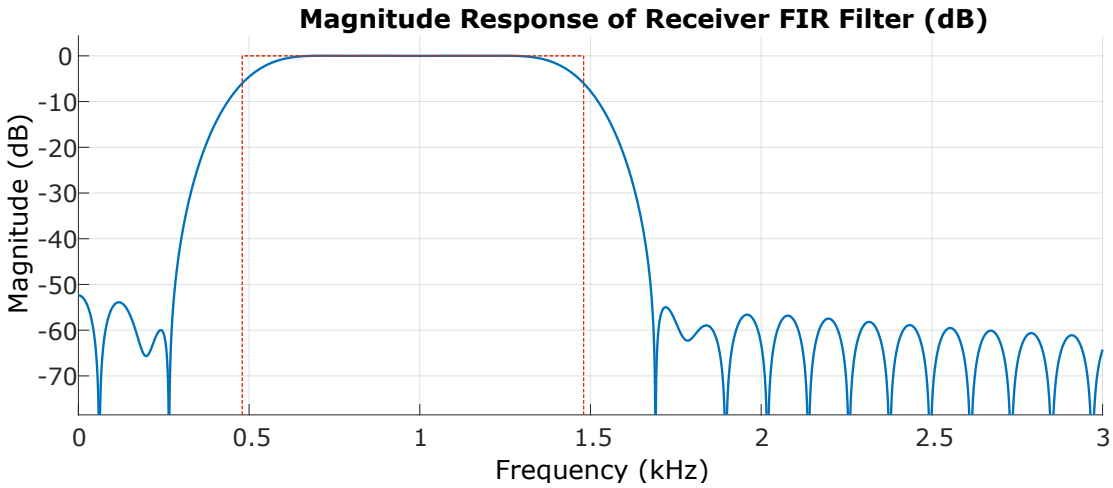


Figure 5.15: Magnitude response of the 80^{th} order FIR bandpass filter cenetered at 980 hz. It is implemented at the receiver in a python script. The purpose of this filter is to remove unwanted components from the signal sampled from the photodiode, leaving the bandwidth associated with the transmission intact.

the symbol rate. The bandwidth of the baseband signal (i.e the signal produced by the symbols before it is multiplied by the carrier frequency) is shifted to be centered around our carrier frequency. If the symbol rate is increased the baseband signal changes at a higher frequencies causing the bandwidth it occupies to grow. A bandwidth of this size, around our carrier frequency, must not be distorted by our filter.

The main part of the script is a loop that starts before the transmitter begins to transmit, and is ended by the arrival of the EOT byte, which marks the end of transmission. Each iteration of the loop starts by reading the latest ADC sample from the receiving Arduino. This sample is appended to the start of a `dqueue` data structure. This data structure holds a enough samples for each coefficient in the FIR filter, i.e. it holds the last 81 samples. The script then evaluates the output of the FIR filter for this sample by multiplying the coefficient of the filter by the corresponding ADC samples from the aforementioned `dqueue`. This filter output is likewise appended to another `dqueue` that holds the 96 latest FIR filter outputs. Note that given the symbol duration is 10ms there are approximately $10\text{ms} \times 9.615\text{ kHz} \approx 96$ samples per symbol duration. A third `dqueue` is updated

at each new sample. The new value, appended to the start of this queue on each iteration, is the energy of the 96 past filter output samples. Energy is calculated like in System 1 following equation (5.1). The values in this queue store the energy of the last 96 symbol durations. Consecutive windows are calculated with a lot of the same samples. The oldest sample of the older window is replaced with the latest sample. The first few outputs of the window energy calculation are based on incomplete information and are useless. When the script commences, 500 samples are allowed to pass through the System to *prime* the window energy queue, before we begin looking for the pilot sequence. This make the system considerably more reliable, and only wastes 52ms.

Once the window energy values have been primed, the script begins analyzing them following essentially the same methods as described in detail for System 1. There are three states the script operates in as it is receiving.

1. Looking for Pilot
2. Synchronizing
3. Receiving

It begins in state 1, tracking the values of the latest window energy. When the hard-coded energy threshold is exceeded, it assumes the pilot has been seen and switches to state 2. It identifies the sample at which the peak energy occurred. Knowing this it achieves synchronisation, and moves to state 3. In the receiving state the window energy of each symbol mapped to the bit 1 or a 0 depending on whether the threshold is exceeded or not. From every set of 8 bits received a bytes is assembled. There is a boolean that tracks whether the next byte is escaped or not. This boolean is set to True when an unescaped ESC byte is received, and set to False after the escaped byte is received. If an unescaped EOF is received the system returns to state 1. If an unescaped EOT arrives the output file is closed and the program ends.

The first frame of data bytes are interpreted as ASCII characters and stored in a variable `fname`. When the frame has ended the script opens a file named with the value in `fname`. The remaining data bytes are written to this file as they arrive.
For more detail on the implementation of states 1 and 2, see subsection 5.3.1 under System 1.

5.4.5 Data Rate

The bit rate achieved with this system is;

$$\frac{8 \times 15_{bytes}}{(3_{\,pilot} + 15_{bytes} \times 8) \times 10\text{ms}} \approx 98\text{bit/s}$$

Chapter 6

Results of the Digital System

The following discusses tests performed on Digital System 2.

6.1 Range

Figure 6.1: Photograph of Digital System 2, operating at a range of 1.7m. The based Arduino VLC receiver is on the left. It includes photodiode amplifier implemented on a breadboard. The Arduino based VLC transmitter is on the right. Both are powered and controlled by the laptop via USB.

The receiver was initially implemented with an unamplified photodiode directly connected to the ADC input. With this configuration consistently successful communication was achieved through a distance $\approx$ 2cm. By linearly amplifying the photodiode using a TLV2460C op-amp in the trans-impedance configuration shown in Figure 5.1, the range was extended substantially. Achieving very low bit error rates at distances of up to 1.7m in a typical office environment.

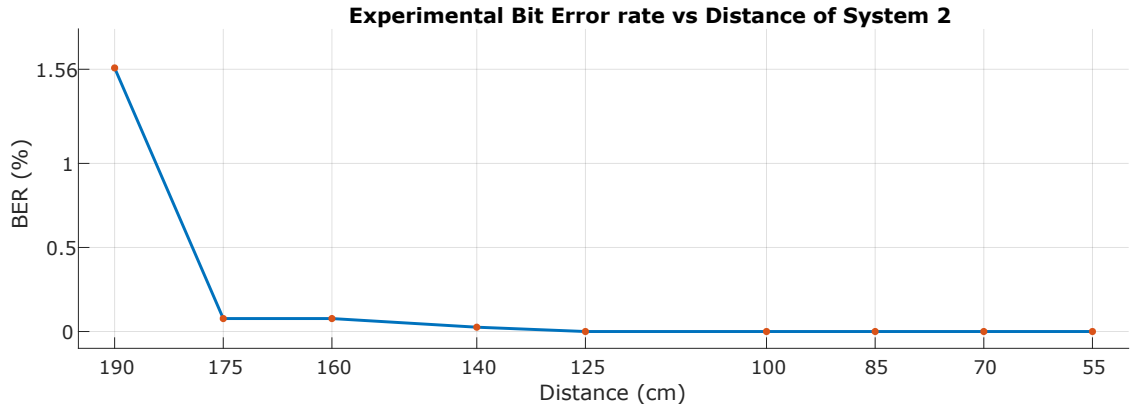


Figure 6.2: Plot of the percentage of bits received in error vs Distance between transmitter and receiver. Plotted using experimental data retrieved by sending a file, 3890 bytes long, at many different distances. Transmission took place in a typical office environment with overhead LED lighting and sunlight.

To test the performance of file transfer through Digital System 2, a text file, 3890 bytes in length was transmitted through several distances. The distance between transmitter and receiver was increased in increments of about 15cm, in the range of 55cm to 190cm. The amount bits received in error was determined by comparing received and transmitted copies of the file. A file comparison tool was used to aid with this. A section of its output is shown in Figure 6.3. The file transmitted consisted of the numbers 0 - 799 each on their own line. The numbering helps the comparison tool.

The findings of this experiment are plotted in Figure 6.2. The system's error rate performance is excellent at distances up to and including 175cm. When the distance was $\leq$ 125cm, the file was received with 0 errors. At a distance of 140cm, only 1 byte was received in error. At both 160cm and 175cm, 3 bytes out of the 3890 were received in error. Beyond this the performance begins to plummet. 61 bytes were received in error at 190m.

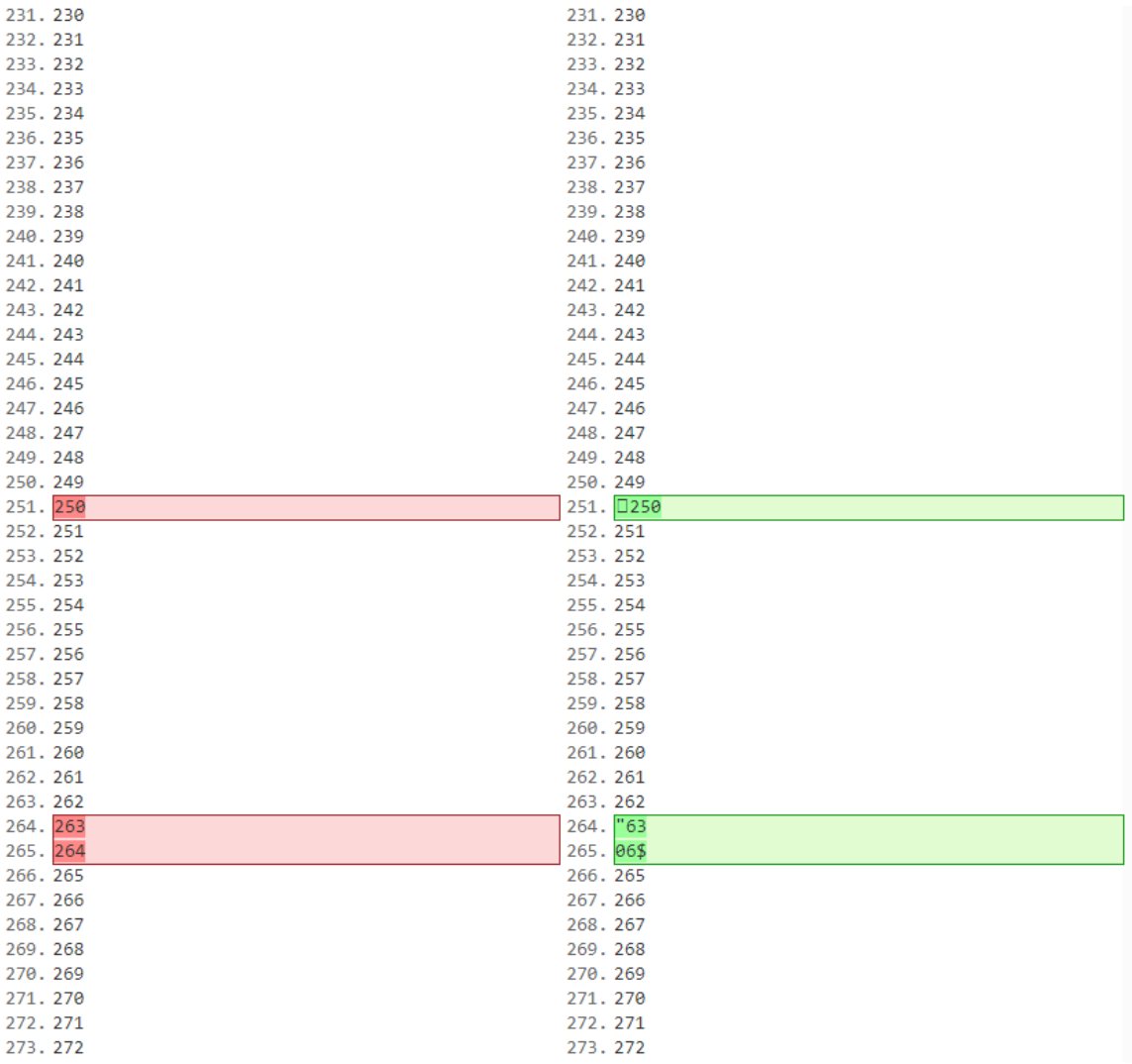


Figure 6.3: A section of the Output of www.diffchecker.com. This tool was comparing the original file (left) with the received file (right). In red and green are the lines in which discrepancies were found.

This performance is excellent overall. Since its error rate performance is so strong at distances $\leq$ 125cm, there are may application to which this system can be applied. Error rates of the larger ranges can be further reduced with the inclusion of channel coding, though the data rate will be consequently reduced as more bits will be included in the transmission. To obtain more accurate BER values for the shorter distances, it would be good to transmit much larger files. Low BER values can not be evaluated without transmitting a considerable amount of bytes, several orders of magnitude more than was done in this experiment. Though it will take a considerable amount of time given the data rate is 98 bit/s.

6.1.1 The Effect of Ambient Light on Range

As the ambient light increases in brightness, the photodiode's DC output increases. The gain of the amplifier must be consequently reduced, so as not to saturate the op-amp. With this reduction in gain the amplitude of the data signal is also reduced. This limits the range of the digital system, as it is difficult to distinguish a weaker signal from noise. In other words the range of the system improves in darker environments substantially.

This limitation could be mitigated by implementing a 2 stage amplifier. The first stage would apply a minimal amount of gain, so that the signal is then large enough to pass through an active filter, that will attenuate the the low unwanted frequency components of the signal caused by ambient lighting, while amplifying only components corresponding to our transmission bandwidth.

A more practical modification that improves range in bright scenarios, is to attach a shade to the PD similar to that of a camera's lens hood. The shade could physically block a substantial amount of unwanted light in particular from the overhead lighting without limiting the photodiodes field of view to a significant extent. This modification was implemented in the final design of both implementations and can be seen in Figure 6.4. The improvement observed was significant. It is one of

the main reasons the system was capable of transmitting at distances greater than 70cm in well lit environments.

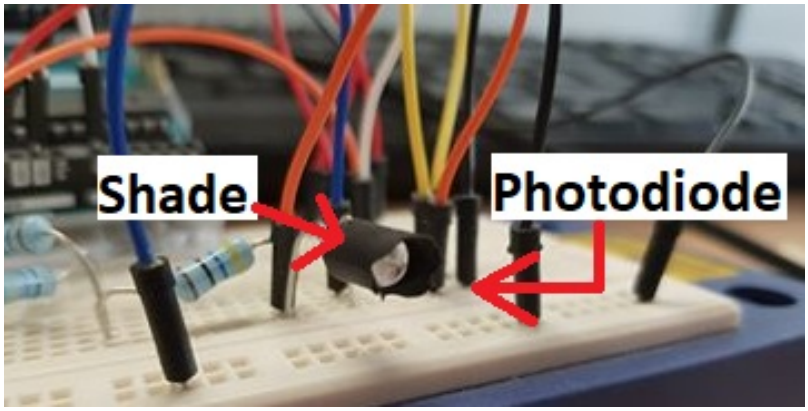


Figure 6.4: Close up of the receivers photodiode. A considerable amount of unwanted ambient light is blocked by the plastic covering placed around the photodiode. This does not significantly reduce the FOV of the receiver. Reducing unwanted light allows us to amplify the data signal by a greater extent, consequently increasing our range.

6.2 Data Rate

The data rate of System 1, the entirely Arduino UNO based implementation, is 24 bits/s. The data rate of the second implementation, where processing is delegated to an external computer, is 98 bits/s. The first implementation is significantly cheaper as Arduinos are inexpensive and no external computer is required to accomplish communication. There is a trade of between cost and performance here.

As a proof of concept they are great results when the limitations of having implementing all signal demodulation in software are taken into account.
The data rate of this design can be vastly increased if signal demodulation is implemented in analog circuitry. We would then only need to sample at the the symbol rate, as opposed to our current implementation where we need to sample at at least

four times the carrier frequency. In addition to this, we would not need to use the ADC as the demodulator would provide digital output.

6.3 Blockage Recovery

The following tests were performed on System 2.

A test was performed to observe the system's ability to recover from blockages as analysed in section 5.3.2. While transmitting over a distance of 1m, we tested the system's robustness by temporarily blocking the LOS path between receiver and transmitter. The LOS was blocked at two separate times for approximately 2 seconds during the transmission of a large text file (The file transmission takes 81 seconds).

The receiver recovered from both communication interrupts. In the first, 31 characters were either not seen or corrupted. Approximately 24 of those characters could not have been received as they were sent during the 2 second block. The remaining 7 were seen by the receiver, but were received in error due to misalignment, as discussed in section 5.3.2. In the second block, 32 characters were lost.

The system's ability to recover adds greatly to the system's usefulness. It would not be particularly usable if the system became permanently misaligned due to a brief blockage. The results of this test agree with the conclusions from the analysis in section 5.3.2.

Chapter 7

Conclusions & Future Work

In this thesis three visible light communication systems are successfully designed and implemented. The first is an Analog mono-audio transmitter and receiver capable of transmitting audio via the light of an amplitude modulated LED. This system performs well at distances up to 1m. Developing the analog system brought to light the issue of interference caused by certain types of ambient lighting. When LED lighting is used, it should be expected that a very large signal is present at double the mains supply frequency, and can interfere with VLC.

The second and third communication systems are real-time digital VLC systems. Both systems communicate through the medium by using a variant of OOK modulation presented in this thesis. Both systems achieve distances of up to 1.7 m, while keeping bit errors rates below 0.08%. These systems are unique due to the fact that all signal demodulation is performed in software running at the receiver. This is as opposed to demodulating using analog electronics prior to sampling. This configuration allows many factors of the modulation scheme to be altered easily and at any time. For instance, the location of the digital filter's pass-band and be changed at any time, allowing us to change the carrier frequency of the protocol. This might be desirable if the default bandwidth becomes exceedingly noisy. The trade off for this flexibility is limited data rates.

The first digital system can transmit ASCII strings from one Arduino to another, with out the need for any external computer. It can be entirely powered by a 5V supply or battery. This system is low in cost and has many application in the field of IOT. For instance, this system could be used to send commands, to IOT devices. This kind of communication does not require great data rates.

The second digital system transmits any type of file, from one computer to another through the visible light channel. An Arduino is used at the receiving end to relay photodiode samples to a script running on the receiving computer. All signal demodulation performed on these samples run on the computer. The environment is easily extendable, facilitating future research. Due to the processing power of an external computer, complex digital signal processing can be added to the current system and performed on received samples in real-time.

Future work on this project should include incorporating channel coding to improve error rates. It should also aim to improve the range of the system by employing a multistage photodiode amplifier including active filtering stages. This allowing the transmission bandwidth to be isolated from ambient light and then amplified. This would make the system less sensitive to the intensity of ambient light. Finally speed can be improved using a faster micro-controller and or by moving signal demodulation to analog circuitry.

The performance of the implemented systems show that LED based VLC is a promising solution to the problem posed by the dwindling RF spectrum. We find that communication can be achieved reliably in an open-plan office environment at distances exceeding 1.7 m. From our testing we found compelling evidence that this range can even be significantly extended by incorporating several straightforward design additions which we discussed. The digital systems we presented are built with common electronic components and are ready to be used as low cost, low power solutions for indoor IOT applications that that favor flexibility over high data rates.

Abbreviations

ADC Analog to Digital Converter

BER Bit Error Ratio/Rate

EM Electromagnetic

IM Intensity Modulation

IOT Internet Of Things

IR Infrared

ISI Inter-Symbol Interference

ISR Interrupt Service Routine

LOS Line Of Sight

OFDM Orthogonal Frequency Division Multiplexing

OOK On OFF Keying

OWC Optical Wireless Communication

PD Photodiode

PPM Pulse Position Modulation

RF Radio Frequency

VLC Visible Light Communcation

Bibliography

[1] N. Telecommunications and I. Administration, "U.S. Frequency Allocation Chart as of January 2016."

[2] D. Karunatilaka, F. Zafar, V. Kalavally, and R. Parthiban, "Led based indoor visible light communications: State of the art.," *IEEE communications surveys and tutorials*, vol. 17, no. 3, pp. 1649–1678, 2015.

[3] P. S. Ryan, "Application of the public-trust doctrine and principles of natural resource management to electromagnetic spectrum," *Mich. Telecomm. & Tech. L. Rev.*, vol. 10, p. 285, 2003.

[4] N. Golmie and F. Mouveaux, "Interference in the 2.4 ghz ism band: Impact on the bluetooth access control performance," in *Communications, 2001. ICC 2001. IEEE International Conference on*, vol. 8, pp. 2540–2545, IEEE, 2001.

[5] M. S. Islim and H. Haas, "Modulation techniques for li-fi," *ZTE communications*, vol. 14, no. 2, pp. 29–40, 2016.

[6] M. S. Islim, R. X. Ferreira, X. He, E. Xie, S. Videv, S. Viola, S. Watson, N. Bamiedakis, R. V. Penty, I. H. White, *et al.*, "Towards 10 gb/s orthogonal frequency division multiplexing-based visible light communication using a gan violet micro-led," *Photonics Research*, vol. 5, no. 2, pp. A35–A43, 2017.

[7] H. Haas, L. Yin, Y. Wang, and C. Chen, "What is lifi?," *Journal of Lightwave Technology*, vol. 34, no. 6, pp. 1533–1544, 2016.

[8] M. Z. Afgani, H. Haas, H. Elgala, and D. 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*, pp. 6–pp, IEEE, 2006.

[9] D. Tsonev, H. Chun, S. Rajbhandari, J. J. McKendry, S. Videv, E. Gu, M. Haji, S. Watson, A. E. Kelly, G. Faulkner, *et al.*, "A 3-gb/s single-led ofdm-based wireless vlc link using a gallium nitride μled," *IEEE Photon. Technol. Lett.*, vol. 26, no. 7, pp. 637–640, 2014.

[10] H. Elgala, R. Mesleh, H. Haas, and B. Pricope, "Ofdm visible light wireless communication based on white leds," in *Vehicular Technology Conference, 2007. VTC2007-Spring. IEEE 65th*, pp. 2185–2189, IEEE, 2007.

[11] R. Wang, Q. Gao, J. You, E. Liu, P. Wang, Z. Xu, and Y. Hua, "Linear transceiver designs for mimo indoor visible light communications under lighting constraints," *IEEE Transactions on Communications*, vol. 65, no. 6, pp. 2494–2508, 2017.

[12] K. Lee, H. Park, and J. R. Barry, "Indoor channel characteristics for visible light communications," *IEEE communications letters*, vol. 15, no. 2, pp. 217–219, 2011.

[13] J. M. Kahn and J. R. Barry, "Wireless infrared communications," *Proceedings of the IEEE*, vol. 85, no. 2, pp. 265–298, 1997.

[14] C. Quintana, V. Guerra, J. Rufo, J. Rabadan, and R. Perez-Jimenez, "Reading lamp-based visible light communication system for in-flight entertainment," *IEEE Transactions on Consumer Electronics*, vol. 59, no. 1, pp. 31–37, 2013.

[15] D. A. Steigerwald, J. C. Bhat, D. Collins, R. M. Fletcher, M. O. Holcomb, M. J. Ludowise, P. S. Martin, and S. L. Rudaz, "Illumination with solid state lighting

technology," *IEEE journal of selected topics in quantum electronics*, vol. 8, no. 2, pp. 310–320, 2002.

[16] Altium, "NEC Infrared Transmission Protocol."

[17] D. of Physics and G. S. U. Astronomy, "Photodiode Light Detector."

[18] O. Electronics, "Photodiode Characteristics and Applications."

[19] J. G. Proakis and D. G. Manolakis, *Digital signal processing*. Pearson Education, 2013.