Wrocław University of Science and Technology

Wroclaw University of Science and Technology

Faculty of Electronics, Photonics and Microsystems

Portable Li-Fi Data Transmission for Embedded Applications

Author: Ezeh Stanley Okechukwu
Department: Embedded Robotics
Class: Intermediate Project
Tutor: Dr. Witold Paluszyński
Date: 10/01/2022

Abstract

In recent years, tremendous research on the standardization and implementation of visible light communication has been done to utilize the infrastructure of the already existing lighting system for data transmission. This project demonstrates the principle of data transmission using standard electronic components and evaluates the performance characteristics of using Li-Fi in the transmission of data in embedded systems. Data are transmitted asynchronously in a half-duplex mode using the Universal Asynchronous Receive Transmit protocol between two nodes at different baud rates. The UART protocol presents some inherent limitations to the speed of data transfer, but was chosen for simplicity and speed up prototyping due to the time constraints of this project. The results obtained were satisfactory and acceptable when considered as a proof of concept despite performing less compared to the benchmark speeds already obtained by the researchers. Further recommendations have been made on how to boost the speed to 100Mbps from the 4.8Kbs obtained for this project.

Introduction

Light Fidelity, also known as Li-Fi, is a visible-light communication technology that allows for the wireless transmission of data using light as a medium. This technology was introduced by a German professor Harald Hass in 2010 after it was publicly announced during a TEDGlobal Talk in Edinburgh [1-Li-Fi - Wikipedia]. In October 2011, his company, PureLiFi, founded the LiFi Consortium, which is a body responsible for the standardization and development of high-speed Li-Fi network protocol in order to overcome the inherent challenges associated with radio wave transmission.

Operation

The standard implementation of the Li-Fi protocol is based on the IEEE 802.11 protocol used in the implementation of the Wi-Fi protocol. In contrast to the latter, the Li-Fi technology solves a major challenge faced in Radio Frequency communication, limited frequencies with finite bandwidth. This is because the Li-Fi protocol uses the visible light spectrum which is approximately ten thousand times larger than the radio wave frequency band.

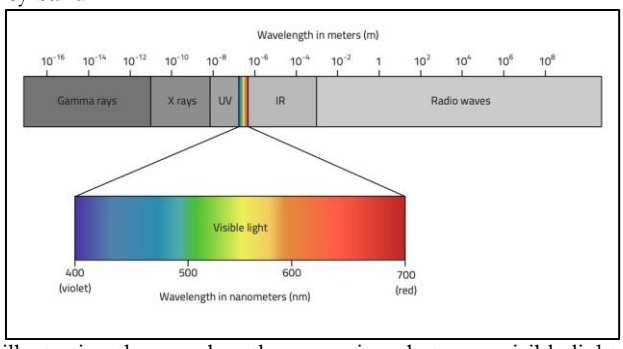


Figure 1: Electromagnetic spectrum illustrating the wavelength comparison between visible light and radio waves. Source: radio2spac

Aim and scope of project

The goal of this project is to implement a working prototype which serves as a proof-of-concept of the visible light communication over free space. The scope is limited to the research, implementation, and testing of a prototype system that is capable of modulating LED pulses to carry data from the transmitter to the receiver without minimal loss and high integrity. Goals achieved include the following.

1. Research on Visible Light Communication and undertake all necessary analyses.
2. Evaluate and compare alternative approaches to project implementation and choose the best fit that satisfies the project constraints.
3. Build, test, and evaluate the performance of the system for transmission of basic data (text-based) files.
4. Build a desktop application to send and receive files between two nodes.
5. Propose improvements and areas of further research and application.

Preliminary Design:

The part list for design has been made using computer-aided design software and simulated. The selected parts include the following.

1. Potentiometer - 220k
2. Operational amplifier – LM358N
3. Phototransistor (L-53P3C)
4. KY-008 laser diode module (650nM)
5. Resistors
6. USB to UART Converters
7. Light-emitting diodes
8. Solderless breadboards.

UART Transmission Protocol

The Universal Asynchronous Receive Transmit protocol was chosen for this project because it allows data to be sent in both full- and half-duplex modes using a minimum number of data channels. For this project, data are sent in a half-duplex mode, thus requiring only one channel of transfer between the receive and the transmit ports of the two communicating devices. Although this protocol does not guarantee the highest speeds available today, it provides reasonable and acceptable maximum speeds of up to 1.5Mbps per second depending on the hardware. For most dedicated UART chips, data speeds average around 1Mbps can be achieved, and this is the reason I chose to use dedicated chips instead of microcontrollers.

How Data is Transmitted

The first step in data transmission involves serialization and encapsulation of the packets. UART can only send serialized data packets, and this makes it suitable for this application because Li-Fi transmits data using modulated 1 and 0's in a serial fashion and uses a single wire for transmission. A typical byte frame is shown below.

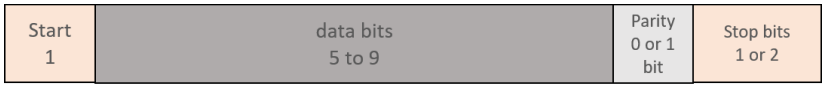


Figure 2: UART Data Frame

The data to be transmitted are encapsulated and serialized into the data bits of the UART packet frame before being sent from the transmitter to the receiver.

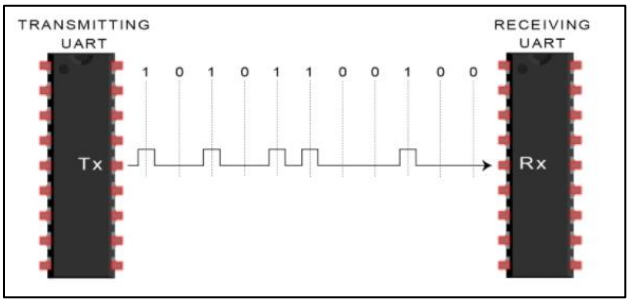


Figure 2: UART transmission (Circuit Labs 2021)

UART was chosen for this project because it possesses the following properties.

1. It does not require a clock signal to be transmitted along with the data, as in most other communication protocols. This simplifies the circuitry and makes it much easier to transmit data using a simple On-Off Keying (OOK) technique.
2. Data can be transmitted seamlessly without the need for modulation or channel encoding.
3. The protocol has an inbuilt error detection system that ensures that the transmitted data are reasonably accurate.

It also possesses some inherent shortcomings that were neglected because the primary objective was to develop a proof-of-concept within the project time constraints. These include the following.

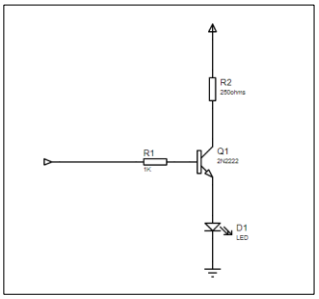
1. The speed of data transmission is limited to hardware. Most embedded microcontrollers can communicate only at baud rates of up to 115200 without loss of data integrity.
2. The size of the data frame is limited to a maximum of 9 bits, which places a huge constraint on how much data can be sent at once.
3. It does not support multiple masters/slaves.

Transmission circuitry

The transmission circuit was designed to send data through the transmit pin of the sending UART chip in the form of light pulses from two different light sources. The design specifications of the LED and Laser Diodes are shown below.

LED	Laser module
• Operating Voltage: 3.2V • Current: 20 mA • Diameter: 3mm • Color: White • Brightness: 2500 mcd • Beam angle: 20 °	• Range – Unlimited in vacuum • Operating voltage – 5V • Current - 30mA • Angle – N/A

In order to switch the LEDs, the circuit is designed thus.



According to the transistor bias specifications, a maximum of 200 mA is required to saturate the transistors. The maximum voltage at the transmit Tx and receive Rx pins of the UART chips is around 5V. Therefore, the bias resistor, R1, as shown in the image above, can be obtained from the ohms law as follows:

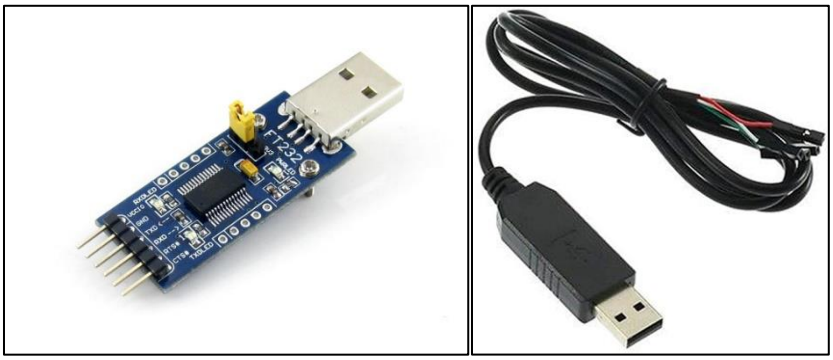
$$R_B = \frac{5}{0.006} = 833\Omega$$

The standard resistor value closest to this value is the 1 kilo ohm resistor. Similarly, to ensure that the maximum current through the LED is clipped at 20mA, a 220 ohm resistor is used.

USB-to-UART Chips

For simplicity and ease of monitoring communication from a personal computer, I have chosen to use USB to Serial Converters with dedicated UART chips. For this project, two chips were used:

1. FT232 UART to USB converter
2. PL2303HX UART Controller



Both chips also have bidirectional buffers, which allow data to be queued before being transmitted or received, especially when large amounts of data are to be transmitted or received when compared to microcontrollers. They possess configurable transmission properties such as baud rates, FIFO buffer size, parity, stop bits, and flow control, and these can be configured through the device drivers.

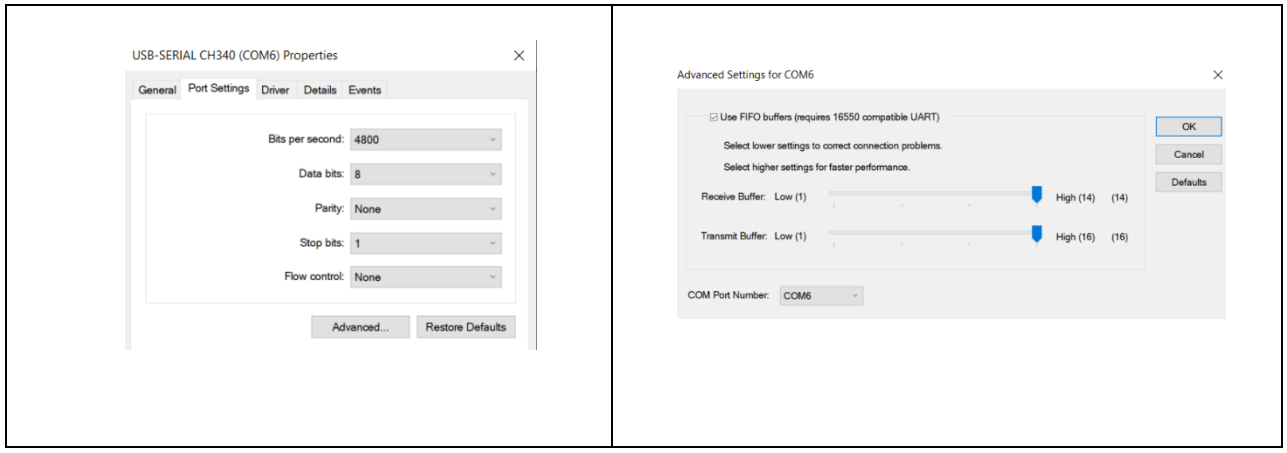


Figure 3: Driver Configuration for USB-UART Controllers

Reception Circuit

The design of the receiver circuit is simple and intuitive. The SFH300-3/4 phototransistor is connected to act as a voltage source and connected in series with a 220 kilo ohms resistor. The voltage across the terminals of the phototransistor depends on the intensity of the incident light. According to the sensor characteristics, it is able to detect incident light with wavelengths within the range of 450 and 1100 nanometers, which cuts across the infrared to visible light region of the electromagnetic spectrum. It possesses a half-angle sensitivity of 25 degrees, with rise and fall times summing up to 20 microseconds. This places huge limitations on the switching applications and the maximum number of bits that can be transmitted per second. At 20 µs, the expected switching frequency at room temperature (25° C) is approximately 50 Kbps, which is quite disappointing but would suffice for the purpose of this project.

The voltage output is then compared with a variable voltage output from a 10k precision potentiometer, and the operational amplifier is configured to operate as a comparator in non-inverting mode as shown below.

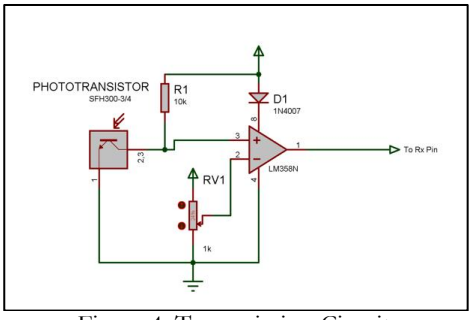
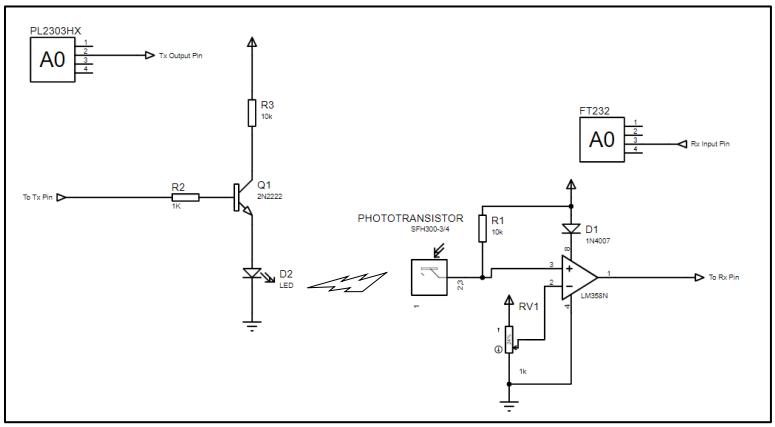


Figure 4: Transmission Circuit

The op-amp switches to high state (logical HIGH) when the voltage in the non-inverting input exceeds the voltage in the inverting input and vice versa. This translates into logical pulses received and amplified from the phototransistor with output voltage.

The complete circuit diagram is shown below:



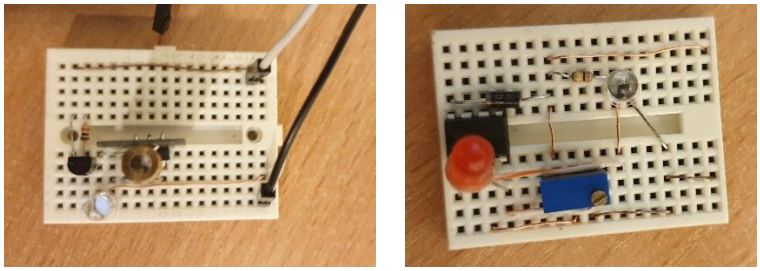


Figure 5: Transmitter and Receiver Circuit

Software Programming

The software aspect of the project focuses on controlling the transmission and reception of data using both chips. In order to allow the PC to transmit data via UART, I have written a Python script that uses the win32 API for serial communication to send and receive data through designated ports with configuration parameters using the PySerial Module. A desktop application capable of transmitting both image and text files was also created to facilitate data transfer and configuration.

Implementation

The GUI is created using the PyQT5 GUI Library, a Python version of the Qt Framework designed to work specifically in C++. It basically provides wrapper functions that call the native C++ APIs contained in the Qt framework. The first version of the created application was only able to transmit 1 Kilobyte of data at a maximum speed of 300 bits per second and crashes when this limit is exceeded. After weeks of debugging, it was discovered that the application crashed because the GUI framework uses a main thread that handles events and tasks and can only respond to one activity at a time. So, when large files are transmitted, the event loop is trapped in the transmission and/or reception activity, and the GUI freezes. To fix this, I created threads to handle these tasks. This allowed me to transmit larger files at faster speeds without crashing. A screenshot of the application is shown below.

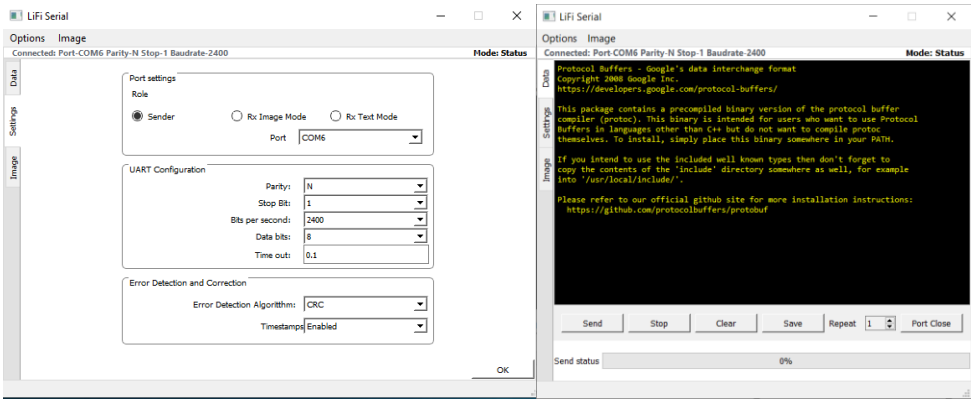


Figure 6: GUI Screenshots

To transmit images, the image is read as bytes and converted to Python byte arrays, which are stored in a buffer before being transmitted serially by the transmitting application. The receiver reassembles the byte array into an image using the Python Image Library and then displays it with the GUI application. Data files like text files are transmitted, which require a little more processing. Since additional information like the time when the image was sent and the number of bytes sent was to be transmitted from the transmitter to the receiver, I decided to use the Binary JavaScript Object Notation, which allows for encapsulation and serialization of structured data. Another option which was explored was the Google protocol buffers, but the BSON format was chosen due to its ease of implementation and light weight. A typical BSON object is created using the native Python dictionary before being encoded. On the receiver side, it is decoded and the original dictionary is obtained. This is shown below.

```python
timeValue = time.time()
msgSize = sys.getsizeof(self.message)
output = {"message": self.message,
          "timestamp": timeValue,
          "messageSize": msgSize
          }
buffer = bson.encode(output)
```

```python
tReceived = time.time()
self.done.emit(True)
val = bson.decode(output)
tSent = val['timestamp']
msg = val['message']
msgSize = val['messageSize']
transmitTime = tReceived - tSent
```

Figure 7: BSON Encoding and Decoding

Tools Used

The following is a list of tools used for this project.

1. Proteus Circuit Designer and Simulator
2. PyCharm IDE Community Edition

Results

Speed: The system was tested at different speeds and for increasing file sizes and file types. The latest version of the application is capable of sending images (.PNG and JPG) files seamlessly and the maximum speed attained in ambient light is 4800 bps bits per second and text files at a maximum of 19200 bps. The results obtained for the laser beam are close to those obtained from the LED. Below is a table showing the transmission speeds for different file sizes and at different baud rates.

S/N	Baud rate (b/s)	File Type	File Size (Bytes)	Time (s)	Distance (mm)
1	2400	Text	101	0.66	80
2	2400	Text	10299	43.18	80
3	2400	Text	20551	85.91	80
4	4800	Text	178	0.61	80
5	4800	Text	4109	8.81	80
6	4800	Text	6138	13.03	80
7	4800	Image	21800	46.03	80
8	9600	Text	1200	1.2	50
9	19200	Text	1000	0.4	50

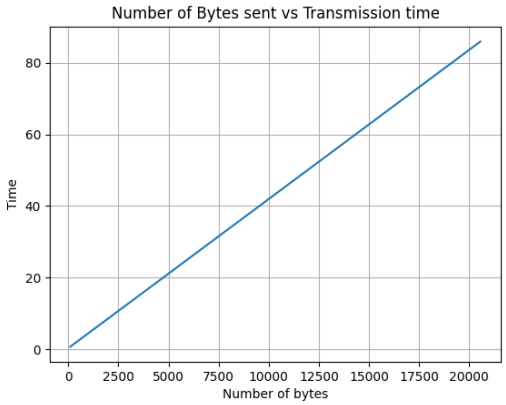


Figure 8: Transfer characteristics for 2400 b/s

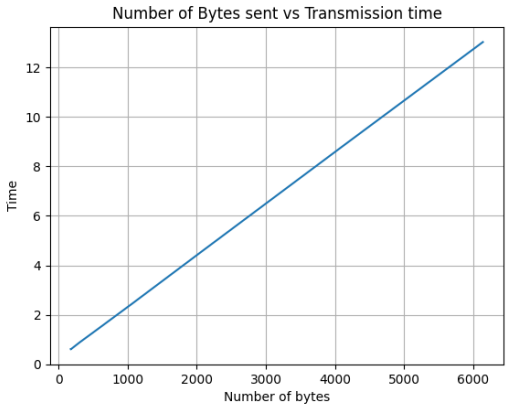


Figure 10: Transfer characteristics for 4800 b/s

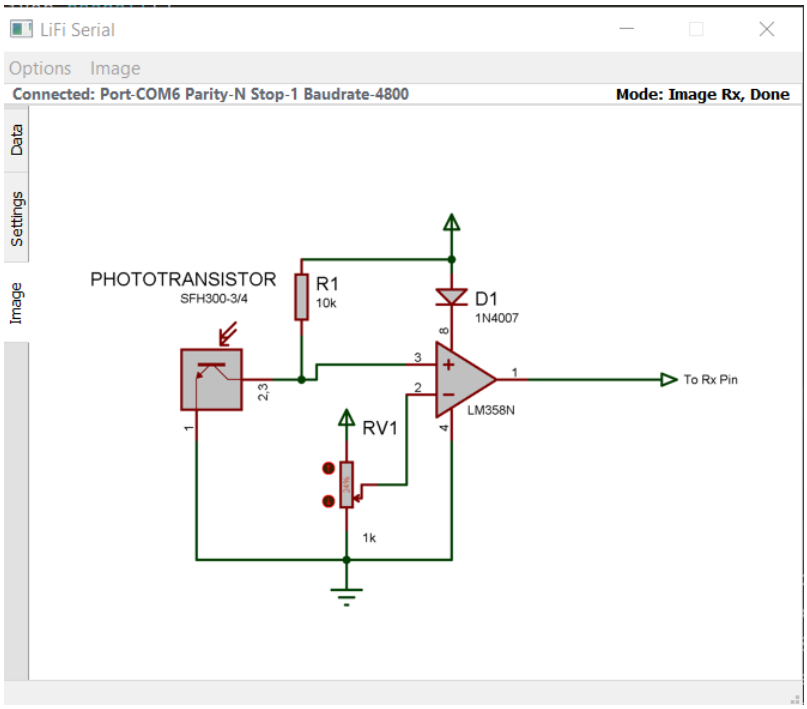


Figure 9: Received Image

Challenges

Some of the challenges faced during the course of this project are

1. Ambient Light: Depending on the time of day, I had to repeatedly adjust the sensitivity of the phototransistor using the precision potentiometer. The initial circuit design was implemented using an ordinary potentiometer, and this presented great difficulties in sensitivity adjustments.
2. Positioning: Since the light sensor has a detection range of about 25 °, it is challenging to position the reception circuit to fall within the line of sight of the laser module. Hence, most measurements were made at a fixed point using the LED, which creates a dispersed beam with higher illuminous intensity.
3. Power: The laser module requires a large amount of current which cannot be sourced from the chips. Thus, a dedicated power supply unit can be added to this to solve this.

Conclusions and recommendations

Although very high data transfer speeds were not achieved, the project can be considered successful. For small data transfer purposes, typically used in embedded systems, this system works perfectly. After researching possible ways to speed up data transmission, we recommend the following recommendations.

1. Instead of using the SH300-3/4 transistor, we use a high-speed photodiode like S5973 Hamamatsu with a very low thermal capacitance of 1.6pF and switching speeds of up to 1Gbps.
2. The comparator circuit can be replaced with a Transimpedance Amplifier with high sensitivity and switching characteristics such as the OPA380

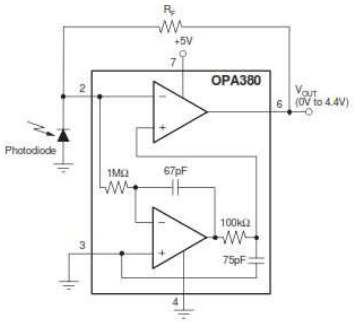


Figure 10: Internal circuitry of the transimpedance amplifier

3. The SPI I2C communication protocol could be explored for this application, as it allows higher transmission speeds. To support speeds of up to 32 Mbps, a Raspberry Pi controller could be used, as it processes data faster than most available microcontrollers.
4. An alternative programming language can be used, since Python is usually outperformed by other hardware-oriented languages like C++ and C.

Applications

1. Vehicle-to-Vehicle Communication using Car Head and Rare Lamps
2. Mobile Robot Localization
3. Large file transfer by incorporating an SMTP server
4. Internet of Things

Resources

1. https://en.wikipedia.org/wiki/Li-Fi
2. Light-Fidelity (Li-Fi) prototype with Raspberry Pi, Paul Fergusson, University of Southern Queensland, October 2016
3. Shekhar Pandey, Purnendu. (2016). Laser as a Medium for Data Transmission, Academic Research In Engineering Management And Information Technology (Icare Mit-2016)
4. Manas Ranjan Mallick, A Comparative Study of Wireless Protocols with Li-Fi Technology: A Survey, IRF International Conference, 29 May 2016.
5. Qian Huang , Yuanzhi Zhang, Zhenhao Ge , Chao Lu. Refining Wi-Fi Based Indoor Location with Li-Fi Assisted Model Calibration in Smart Buildings, International Conference on Computing in Civil and Building Engineering (ICCCBE), 2016.

Appendix

1. The code link can be found on GitHub: https://github.com/CodeyBank/Intermediate-Project.git