

CONTENTS

1. Project Description.....	2
2. Components	3
3. Key concepts	4
4. Objectives	5
5. Experimental Setup and Performance Evaluation.....	6
i. Test 1: Transmitter and Receiver Capacity Assessment (Successful).....	6
ii. Test 2: Single Transceiver Operation via Router (Successful)	6
iii. Test 3: Clock Synchronization and TDM with 3 Transmitters and 1 Router (Successful)	7
iv. Test 4: Router Data Broadcasting Post-Interval Reception (Failed).....	8
6. Discussion.....	9
Figure 1. Assembly of experiment components	4
Figure 2. Illustration of the star topology.....	5
Figure 3. Illustration of TDM mechanism in the project configuration.....	5
Figure 4. Received data of test 3 BEFORE separating according to addresses	7
Figure 5. Received data of test 3 AFTER separating according to addresses	8

1. Project Description

The project focuses on the development and implementation of a data transmission system using Light Fidelity (LiFi) technology. LiFi employs LED lights to transmit data wirelessly, providing a secure and efficient alternative to traditional communication methods. The system utilizes a star topology with Time Division Multiplexing (TDM) to optimize data exchange among four transceivers placed in a closed box.

2. Components

- **Transceivers**

The experiment employs four sets of transceivers, with a single master transceiver functioning as a router. Each transceiver consists of three key components:

- 1) A TX (transmit) module, utilizing either a 650nm laser module (for users) or a 940nm LED diode (for the router).
- 2) An RX (receive) module, capable of receiving either laser light (for users) or 940nm infrared (IR) light (for the router).
- 3) An Arduino Uno board responsible for processing the TX/RX modules.

These transceivers have been strategically positioned within an enclosed box to optimize data transmission.

- **Computers**

External computers are linked to each Arduino Uno through USB cables, tasked with processing data input and output according to the assigned timeslot of the transceiver.

- Illustrations for experimental setup

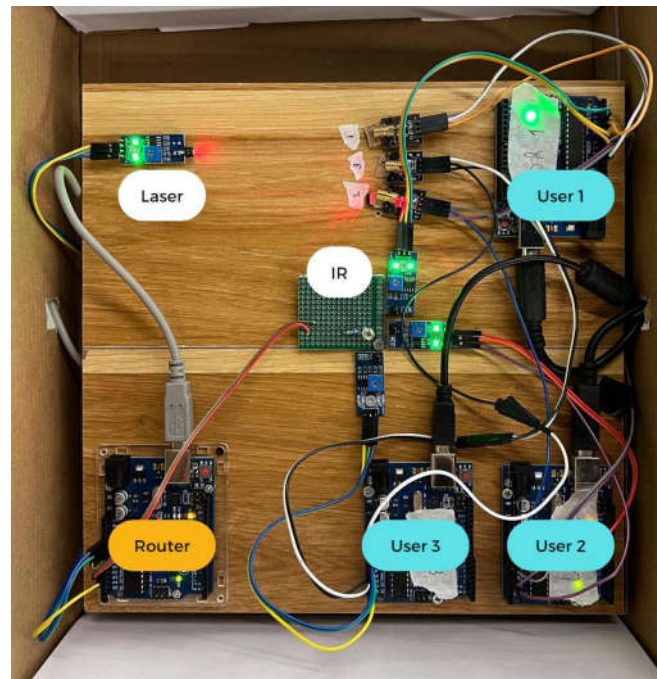


Figure 1. Assembly of experiment components

3. Key concepts

- LiFi Technology: The system leverages LiFi technology for wireless data transmission through modulated LED light. This method offers advantages such as increased security, reduced interference, and high-speed data transfer.
- Star Topology: The star topology ensures a centralized communication structure, with each transceiver directly connected to a central point, which is one master transceiver playing the role of a router for three other transceivers. This design simplifies data routing and management within the closed box.

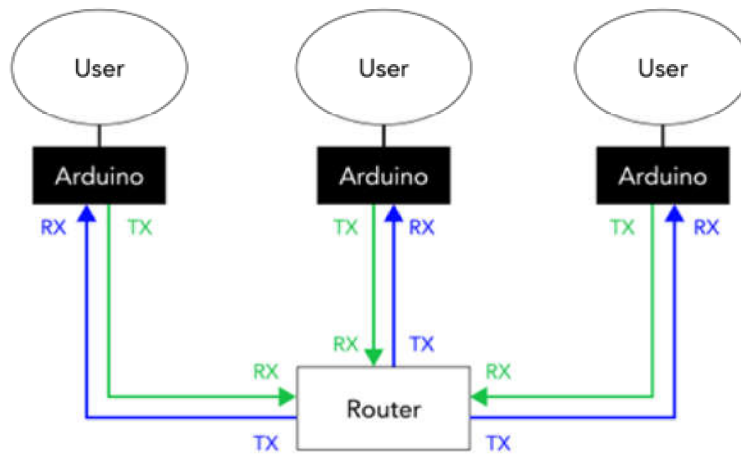


Figure 2. Illustration of the star topology

- Time Division Multiplexing (TDM): TDM is employed to optimize the utilization of the available time slots for data transmission. Each transceiver is assigned a specific time slot, enabling synchronized communication without interference.

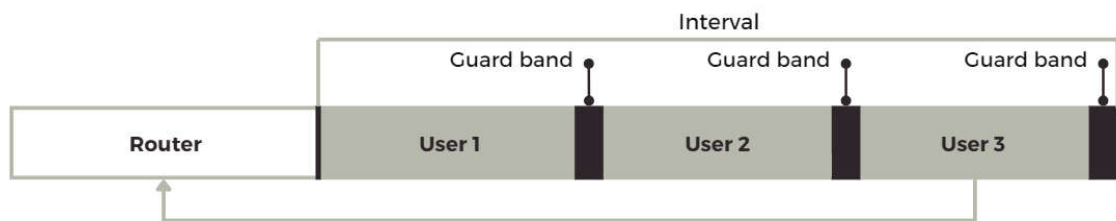


Figure 3. Illustration of TDM mechanism in the project configuration

- Clock Synchronization: Clock synchronization is vital in our LiFi-based data transmission system. The router, acting as the master transceiver, regularly sends timestamps to all connected transceivers. This synchronization aligns their internal clocks, ensuring precise data transmission during their assigned time slots

4. Objectives

- Demonstrate the feasibility and effectiveness of LiFi technology in a closed environment.
- Highlight the advantages of a star topology in simplifying communication within the network.
- Implement TDM for optimized data transmission among multiple transceivers.

5. Experimental Setup and Performance Evaluation

Upon the installation of the four transceivers inside the enclosed box, we executed a series of rigorous evaluations to determine the efficacy and functionality of the LiFi data transmission system.

i. Test 1: Transmitter and Receiver Capacity Assessment (Successful)

Objective: To evaluate the maximum and minimum baud rates for both laser module and IR 940nm LED diode receivers, using only serial communication of Arduino.

- **Laser Module to Receiver**

- File Size: 500 bytes
- Maximum Baud Rate: 19,800 (Bit Error Rate: 0.001)
- Minimum Baud Rate: 700 (Bit Error Rate: 0.00)

- **IR 940nm LED Diode to Receiver**

- File Size: 500 bytes
- Maximum Baud Rate: 1,200 (Bit Error Rate: 0.00)
- Minimum Baud Rate: 300 (Bit Error Rate: 0.00)

ii. Test 2: Single Transceiver Operation via Router (Successful)

Objective: To test data transmission from one user to another through the router.

Process: User 1's Arduino reads and encodes a text file into binary, transmitting it to the router's Arduino. The router temporarily stores the data and, upon completion, forwards it to User 2's Arduino. User 2's Arduino relays the received data to the PC for decoding and storage.

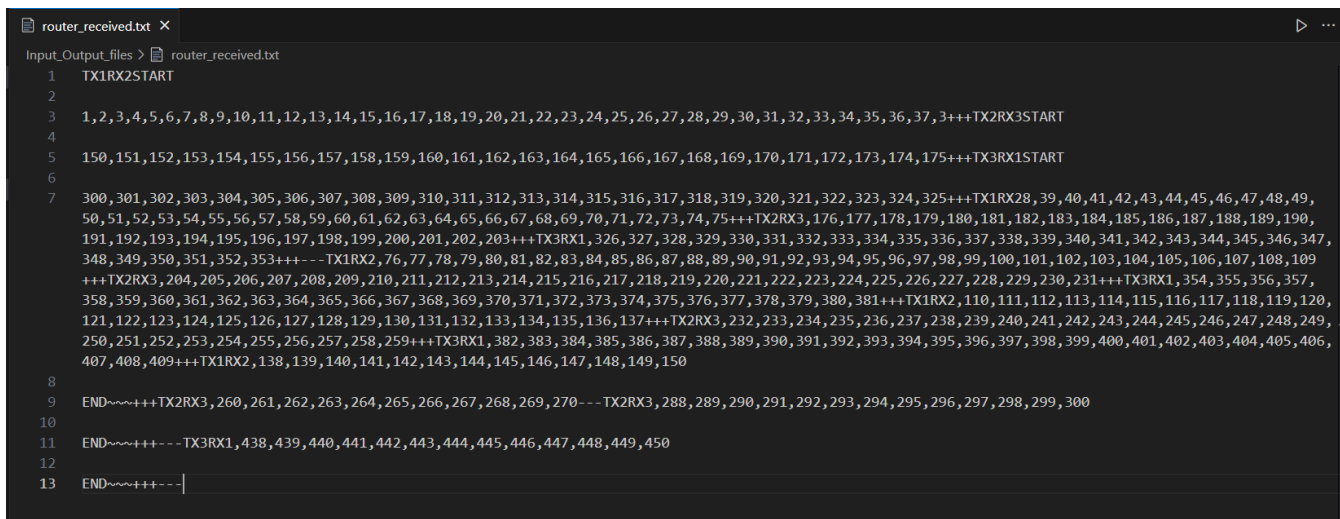
- File Size: 500 bytes
- Baud Rate: 1,200

- Timeout: 4 seconds
- Total Runtime: 9 seconds (Arduino sketches uploading and Timeout are not included)
- Bit Error Rate (BER): 0.00

iii. Test 3: Clock Synchronization and TDM with 3 Transmitters and 1 Router (Successful)

Objective: To assess the system's efficiency in handling multiple transmitters using TDM and synchronized clocks.

Process: The router dispatches timestamps to align internal clocks of users. During their allocated timeslots, users transmit data to the router. The router compiles received data into a buffer, transmitting it to the PC post-interval for processing and storage.



```

router_received.txt
Input_Output_files > router_received.txt
1 TX1RX2START
2
3 1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20,21,22,23,24,25,26,27,28,29,30,31,32,33,34,35,36,37,3+++TX2RX3START
4
5 150,151,152,153,154,155,156,157,158,159,160,161,162,163,164,165,166,167,168,169,170,171,172,173,174,175+++TX3RX1START
6
7 300,301,302,303,304,305,306,307,308,309,310,311,312,313,314,315,316,317,318,319,320,321,322,323,324,325+++TX1RX2,39,40,41,42,43,44,45,46,47,48,49,
50,51,52,53,54,55,56,57,58,59,60,61,62,63,64,65,66,67,68,69,70,71,72,73,74,75+++TX2RX3,176,177,178,179,180,181,182,183,184,185,186,187,188,189,190,
191,192,193,194,195,196,197,198,199,200,201,202,203+++TX3RX1,326,327,328,329,330,331,332,333,334,335,336,337,338,339,340,341,342,343,344,345,346,347,
348,349,350,351,352,353+++---TX1RX2,76,77,78,79,80,81,82,83,84,85,86,87,88,89,90,91,92,93,94,95,96,97,98,99,100,101,102,103,104,105,106,107,108,109
+++TX2RX3,204,205,206,207,208,209,210,211,212,213,214,215,216,217,218,219,220,221,222,223,224,225,226,227,228,229,230,231+++TX3RX1,354,355,356,357,
358,359,360,361,362,363,364,365,366,367,368,369,370,371,372,373,374,375,376,377,378,379,380,381+++TX1RX2,110,111,112,113,114,115,116,117,118,119,120,
121,122,123,124,125,126,127,128,129,130,131,132,133,134,135,136,137+++TX2RX3,232,233,234,235,236,237,238,239,240,241,242,243,244,245,246,247,248,249,
250,251,252,253,254,255,256,257,258,259+++TX3RX1,382,383,384,385,386,387,388,389,390,391,392,393,394,395,396,397,398,399,400,401,402,403,404,405,406,
407,408,409+++TX1RX2,138,139,140,141,142,143,144,145,146,147,148,149,150
8
9 END~+++TX2RX3,260,261,262,263,264,265,266,267,268,269,270---TX2RX3,288,289,290,291,292,293,294,295,296,297,298,299,300
10
11 END~+++---TX3RX1,438,439,440,441,442,443,444,445,446,447,448,449,450
12
13 END~+++---

```

Figure 4. Received data of test 3 BEFORE separating according to addresses

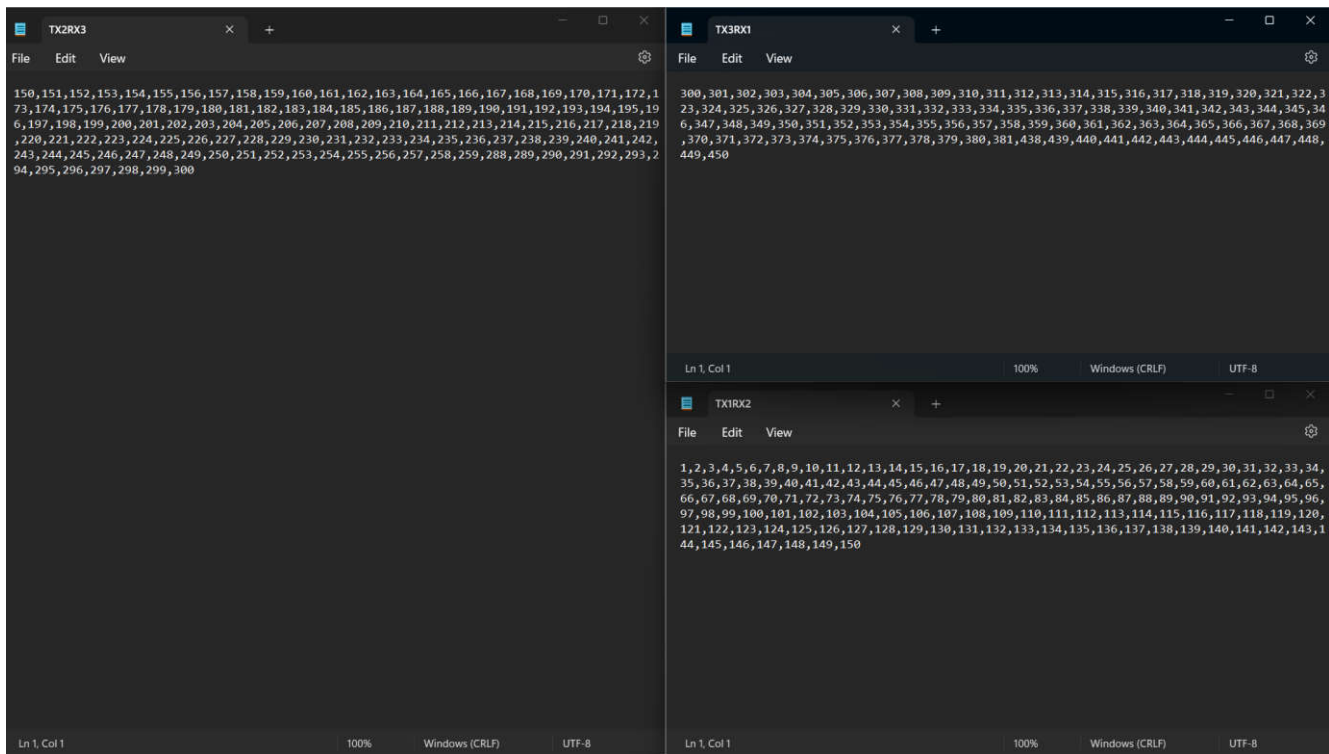


Figure 5. Received data of test 3 AFTER separating according to addresses

- File Size: 500 bytes
- Baud Rate: 1,200
- Timeslot Duration: 0.7 second
- Guard Band: 1 second
- Interval: (Timeslot Duration + Guard Band) * Number of Users
- Timeout: Interval * 3
- Address Format: TX{transmitter_id}RX{receiver_id}
- Timestamp: 0.00
- Total Runtime: 31 seconds (Arduino sketches uploading and Timeout are not included)
- Bit Error Rate of each transceiver (BER): 0.00

iv. Test 4: Router Data Broadcasting Post-Interval Reception (Failed)

Objective: To validate the router's capability to broadcast received data to all users post-interval.

Issue: The test was unsuccessful due to challenges in implementing multiple or sequential serial communications on the Arduino boards.

Observations: The failure highlights a crucial area for further research and development, specifically in managing simultaneous serial communications in Arduino-based systems.

6. Discussion

- **Hardware limitations**

- IR LEDs and sensors: Following numerous tests, we achieved a peak of 4800 bps. Unfortunately, the PCB of the IR LED was damaged. Despite constructing a new PCB circuit, we were only able to attain a speed of 1200 bps.
- Inequal in different lines' speed: Theoretically, the speed of the lines transmitted from router to user must have 3 times the capacity of user to router, as the data from the router contains the information of 3 users and broadcasts their data. However, the real case test shows that the capacity of the line from users to router (laser) is up to 38400. While the speed from router to users (IR led) is only 1200. As a result, we wasted the capacity of laser.
- Arduino: The GPIO pins of Arduino can only adapt only 1 TX line only or 1 RX line only. If both are transmitting and receiving, the data will be broken. Thus, we cannot transmit and receive at the same time. The speed of Additionally, the default value of the software serial, when not transmitting data, is 1. This implies that the laser remains ON in the absence of data transmission. Thus, once a user completes their data transmission, the laser needs to be deactivated, and the subsequent user will initiate their software serial. The transition time between two devices extends the guard band time, resulting in slower transmission.

- **Software limitations:**

- Data Storage: The Arduino is incapable of storing substantial data for transmission. Consequently, if data transmission is required, the Arduino can only serve as an intermediary device between one laptop and another. This setup leads to an increase in transmission time as the file is stored on the laptop and programmed to be sent to the Arduino before it is transmitted through the optical link.
- Debugging and Configuration: Upon executing the program, it monopolizes the Arduino port for its exclusive use. As a result, we are unable to monitor the data directly during execution and only become aware of the outcome once the code has finished running. This makes troubleshooting challenging, as it is difficult to determine whether the issue lies within the software, hardware, or libraries if an error arises.
- Forward Error Correction: Our attempts to implement forward error correction or checksum, or to execute the SYN-ACK as in TCP topology, were unsuccessful. We have opted to use UDP for data transmission. This approach exposes us to the risk of erroneous data appearing in our data frame. In the most severe cases, errors could occur in the add-in data, including the END_SIGNAL and USR_IP. Such errors would render the data irretrievable, leading to a significant loss of data.
- Data Broadcasting: We managed to successfully test all the individual functions of the system, which included reading data from users, allocating timeslots for data transmission, reading data from the router, and broadcasting data. We were also successful in integrating these tasks. However, we encountered difficulties with the final task of data broadcasting into the whole system.