LIGHT OF THINGS COMMUNICATION SYSTEM USING STM BOARDS

Group members:

- MVHASA001
- MKHMAX003
- HDBSTA001
- MPNPHU007

Table of contents

INTRODUCTION

The "Light-of-Things" (LoT) communication project represents a fascinating exploration of innovative ways to transmit data in the field of embedded systems. This project leverages the capabilities of an STM32 development board to design a communication system that transmits data via light signals. In this introduction, we provide an overview of the project, explain our main design choices, and outline how project activities have been allocated among team members as well as report structure.

Project Overview:

The central objective of this project is to develop a communication system that enables data transfer between two STM development boards using light signals. The system comprises a transmitter and a receiver, each equipped with unique functionalities. The transmitter is responsible for sampling data from an onboard POT, converting it into a binary data packet, and transmitting this data via an LED. On the receiver side, the system listens for incoming messages and interprets them by employing light-dependent resistors (LDRs) and ADCs for data reception.

Design Choices:

Our design choices have been guided by the need to balance functionality, complexity, and practical value. We have made the following key design choices:

1. **Communication Protocol:** We have devised a custom LoT message protocol. This protocol includes a defined structure for messages, timing diagrams, and support for both ADC sample transmission and checkpoint messages. By creating a protocol tailored to our project's needs, we can ensure efficient and reliable data transmission.
2. **Light-Based Data Transmission:** We have opted for light-based data transmission for its feasibility and practical application. This approach involves using LEDs to transmit binary data, with the option to adjust transmission speed to cater to different applications.
3. **Checkpoints and Data Tracking:** To enhance the robustness of our system, we have incorporated a checkpoint feature. The transmitter periodically sends a checkpoint message indicating the number of samples sent, allowing the receiver to verify data integrity. This is a crucial feature in ensuring reliable data transmission.

Allocation of Activities:

To efficiently manage our project, we allocated different activities among team members. Our project activities are categorized into several key tasks, including:

- **System Design:** This task involves designing the overall system architecture, communication protocol, and deciding on hardware components.
- **Circuit Design:** We created circuit diagrams for both the transmitter and receiver sides.
- **Software Development:** The development of Python or C programs for the transmitter and receiver, including message encoding and decoding, data conversion, and checkpoint handling.
- **Testing and Debugging:** Rigorous testing and debugging are vital to ensure that a system operates as intended. We met several times over a couple to days to do this task, and some details included tracking samples sent and received, as well as handling checkpoint messages.

- **Documentation:** We collaboratively worked on documenting the project's progress, design choices, and results.

Allocation table

Activity	Group members			
	HDBSTA001	MVHASA001	MPNPHU007	MKHMAX003
System design and diagrams	✓	✓	✓	✓
Circuit design and diagrams	✓	✓	✓	✓
Software development	Transmitter	Transmitter	receiver	receiver
Testing and debugging	✓	✓	✓	✓
Documentation	✓	✓	✓	✓

Project Report Structure:

Our project report follows a structured format that aligns with the project's planning and design phases. It comprises the following sections:

1. **Introduction:** The present section, providing an overview of the project, design choices, and activity allocation.
2. **Specification and Design:** This section includes details on the LoT message protocol, circuit diagrams, design choices, and design diagrams.
3. **Implementation:** This section includes snippets of important code and explanations for this as well as elaborating any parts of the design diagrams.
4. **Validation and performance:** A comprehensive account of the testing process, tracking samples sent and received.
5. **Conclusion:** A summary of the project's achievements, challenges, and future possibilities.

REQUIREMENTS & LOT MESSAGE PROTOCOL

This section outlines the detailed requirements for the implementation of the Light-of-Things (LoT) communication system using STM boards. It contains descriptions of the system components, their functions, and provide diagrams to illustrate the system's architecture and connections. Additionally, any departures or additions made compared to the original project description will be highlighted for clarity.

System Components and Architecture:

The LoT communication system comprises two main components: the Transmitter and the Receiver. The Transmitter is responsible for sampling data from an onboard POT, encoding it into binary data packets, and transmitting this data via an LED. The Receiver listens for incoming messages, decodes them, and uses LDRs and ADCs for signal reception.

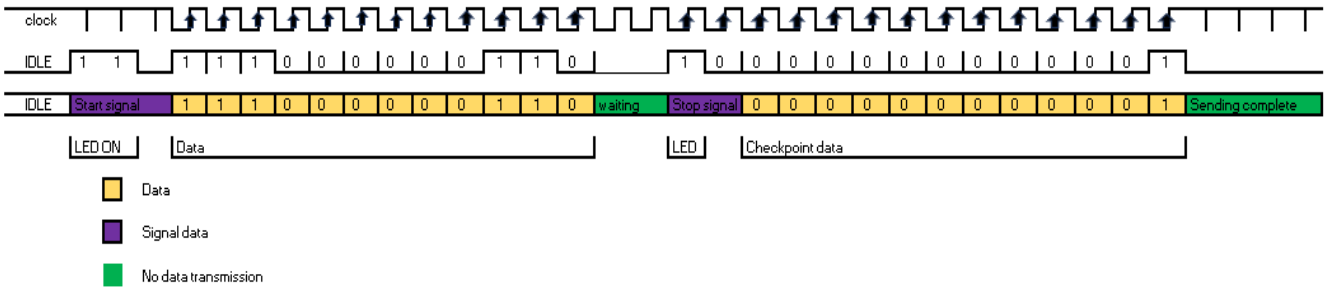
Transmitter:

- **Sampling Module:** The Transmitter features an onboard POT and a push-button. When the push-button is pressed, the system samples data from the POT.
- **Data Encoding:** The sampled data is converted into binary data packets according to the LoT message protocol. The protocol includes start and end indicators and timing specifications.
- **LED Transmission:** Data is transmitted via an LED by turning it on and off based on the binary data.
- **Checkpoint Message:** The transmitter sends a "checkpoint" message indicating the number of samples sent. This is for data integrity and verification

Receiver:

- **Light Reception:** The Receiver employs LDRs and ADCs to receive data transmitted by the LED. It decodes the incoming data packets according to the LoT message protocol.
- **Checkpoint Verification:** The Receiver checks the checkpoint message from the Transmitter to verify data integrity. If the checkpoint count matches the received samples, it indicates an "all received" alert. If not, it displays a "missing samples" alert.

Message protocol:



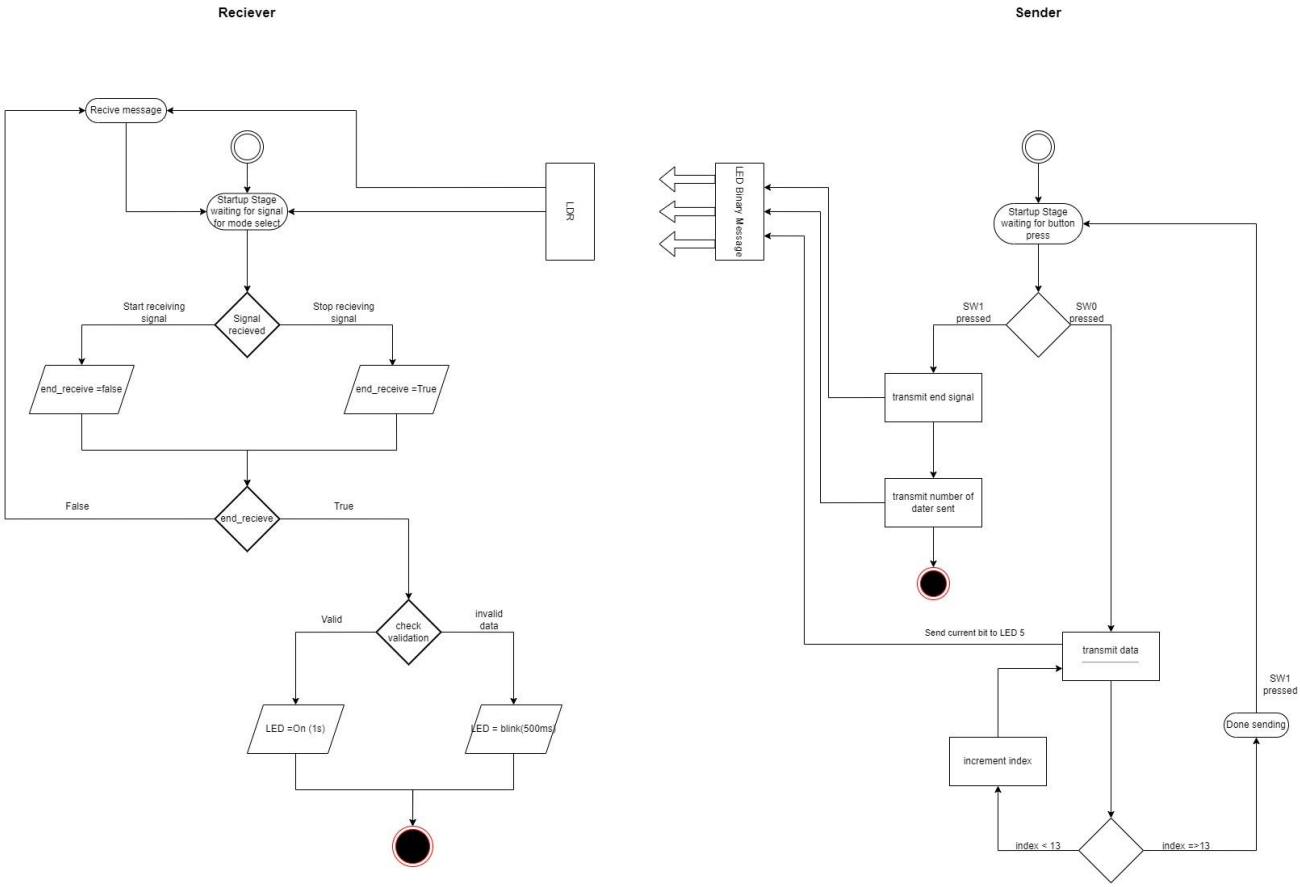
Departures and Additions

- When telling the system to receive a message, we turn the LED on for 1 second
- When telling the system to end all transmission, we turn the LED on for a duration strictly between 0.5s and 1s (exclusive)
- When doing validation of samples sent, we show the actual number received and the actual number the sender is reporting on the LCD instead of blinking the LED if the numbers don't match. This allows us to see the actual difference between the numbers

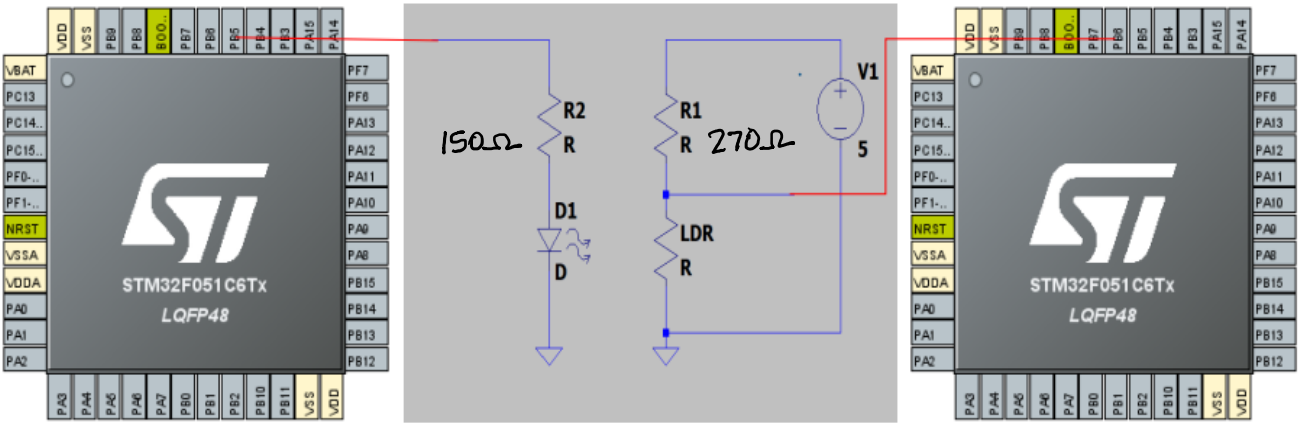
SPECIFICATION AND DESIGN

- We are using an LDR and ADC to transmit data
- The ADC is a combined value of the potentiometer and the LDR reading. This means that the potentiometer needs to be at absolute ZERO for accurate readings of the LDR
- The system was tested in an environment wherein the standard lux reported by LDR was ~10 and the LED increases the lux to 18.
- We use that lux value of 18 to differentiate between and HI signal from a LOW signal

Flow chart



Circuit diagram


VDD
VSS
PB9
PB8
BOO..
PB7
PB6
PB5
PB4
PB3
PA15
PA14
VBAT
PC13
PC14..
PC15..
PF0-..
PF1-..
NRST
VSSA
VDDA
PA0
PA1
PA2
PF7
PF6
PA13
PA12
PA11
PA10
PA9
PA8
PB15
PB14
PB13
PB12
PA3
PA4
PA5
PA6
PA7
PB0
PB1
PB2
PB10
PB11
VSS
VDD
STM32F051C6Tx
LQFP48
R2
R
150Ω
D1
D
R1
R
270Ω
LDR
R
V1
5

IMPLEMENTATION

Code links

The link below is for the github link:

asaMavhungu/Light-of-things (github.com)

Sender:

In the main function we made use of FLAGS for the logic in the while loop

- NO FLAGS: Just poll the and wait for SW0 or SW1 press
- Transmit FLAG: Send “START RECEIVING/ END TRANSMISSION” signal and convert the int adc_val to binary and send out each index through the LED
- Done_sending FLAG: Set adc_value to be ‘counter’, turn on transmission and send that value

We also made use of a function called ‘decimalToBinaryString’:

- This converts an integer decimal value its binary string representation

We made use of funtions setLCD1 and setLCD2:

- These put a string on the first and second row of the LCD respectively

Receiver

Below are snippets of code from the main.c file for the receiver explaining some parts and functionalities that may require more explanation.

Reading ldr implementation

Receive data of each sample by reading ldr, if ldr above threshold then record 1 and turn on the led7,

Else if ldr is below threshold record 0 and turn off the led7.

```
if (recieving_stage)
    {
      if (lux>threshold)
      {
        data = True;
        HAL_GPIO_WritePin(LED7_GPIO_Port,LED7_Pin,GPIO_PIN_SET);
      }
      else
      {
        data = False;
        HAL_GPIO_WritePin(LED7_GPIO_Port,LED7_Pin,GPIO_PIN_RESET);
      }
     if (index!=0)//do not record starting bit
      {
        bin_number2[index-1] = (data+48);
      }
      ++index; //update index of next bit to be received
      writeLCD(bin_number2);
```

Checkpoint implementation

If all samples have been received then validate the number of samples received

If number of samples correponds to checkpoint recieved from sender leave led on for 5sec

Else if they are not equal toggle led on and off with 200ms delay

```
if (actual_checkpoint!=checkpoint)
          {
            uint16_t ledFlash= 0;
          while (ledFlash<5000)
            {
              HAL_GPIO_WritePin(LED7_GPIO_Port,LED7_Pin,GPIO_PIN_SET);
              HAL_Delay(200); //flash led on and off for 1 sec with 200ms
delay inbetween
              HAL_GPIO_WritePin(LED7_GPIO_Port,LED7_Pin,GPIO_PIN_RESET);
              HAL_Delay(200);
              ledFlash += 200;
            }
          }
          //Correct number of samples received, leave led on
          else if(actual_checkpoint==checkpoint)
          {
            HAL_GPIO_WritePin(LED7_GPIO_Port,LED7_Pin,GPIO_PIN_SET);
            HAL_Delay(5000);
          }
```

VALIDATION AND PERFORMANCE

The video shows test cases where we have tested that the system works reliably.

Video links

IMG_2370 - VEED

https://drive.google.com/file/d/1qWlF0G5z1mSOebxp1Baot_OUhQptKiUv/view?usp=sharing

CONCLUSION

Our objective at the start of this project was to create and test a light of things communication system which sends and receives data between two stm32 boards using light signals.

In summary we managed to get the communication system we devised to work as planned. We managed to transmit data signals through an LED by the transmitter and reception by the receiver through an LDR. An import part of the project was to verify and confirm data integrity of the data being transmitted. This was a goal which was achieved by using checkpoints. At the end of transmission, the receiver would verify if it got all the data packets as required by matching its count to that send by the transmitter

Potential future uses.

The LOT communication system works remarkably well at short distances between the transmitter and the receiver. However, this success cannot be guaranteed for long distances of transmission and reception rendering its practical uses to be minimal.

Here are some future uses of the system

Secure data transfer in enclosed spaces like data black boxes where radio frequencies may not be reliable.

Use in limited home automation short distance control e.g smart lighting systems to synchronise lighting patterns

Data collection in agriculture by collecting data from sensors and transmitting it to a central controller

THE END