Faculdade de Engenharia da Universidade do Porto

Data Connection Protocol

1st Project

Redes de Computadores
Class 2

Gonçalo Miranda - up202108773
Sophie Large - up202303141

Introduction

The goal of this work is to develop and test a data link protocol in accord with the specifications in the guide, to transfer a file through the serial port. This report is divided in several parts:

- **Code Structure**: Showing functional blocks and interfaces.
- **Data Link Protocol**: The functioning of the data link protocol.
- **Application Protocol**: The functioning of the application protocol.
- **Data Link Protocol Efficiency**: Measuring the efficiency of our protocol in the data link layer.
- **Conclusion**: Summary of the report's information and what this work brought us

Code Structure

Functional Blocks:
The project is made up of two main parts: The Link Layer and the Application Layer.

The Link Layer, also called the data link layer, involves the implementation of the protocol. It's in charge of setting up and ending the connection, creating and sending data frames through the serial port, and checking received frames for errors, sending error messages if needed.

The Application Layer uses the Link Layer's tools to send and receive data packets that make up a file. This layer is closer to the user and allows you to adjust things like frame size, transfer speed, and the number of retries. The program runs on two computers, one in transmitter mode and the other in receiver mode, with each using a separate terminal.

Interfaces:
The program runs on two computers, each using its own terminal. On one computer, it is set to send data (transmitter mode), and on the other, it is set to receive data (receiver mode).

Data Link Protocol

The data link layer is the layer that interacts directly with the Serial Port, responsible for communication between the sender and the receiver. We use the Stop-and-wait protocol for establishing and terminating the connection, and for sending supervision and information frames.

The connection establishment is carried out by the function llopen(). After opening and configuring the serial port, the sender sends a SET supervision frame and waits for the receiver to respond with a UA supervision frame. When the receiver receives the SET frame, it responds with UA. If the sender receives the UA frame, the connection has been successfully established. After the connection is established, the sender begins to send information that will be read by the receiver.

Information transmission is done by the function llwrite(). This function takes a control or data packet and applies byte stuffing to it to avoid conflicts with data bytes that are the same

as the frame flags. Subsequently, this packet is transformed into an information frame, sent to the receiver, and a response is awaited. If the frame is rejected, the transmission is repeated until it is accepted or the maximum number of attempts is reached. Each transmission attempt has a time limit, and if it exceeds this limit, a timeout occurs.

The reading of information is performed by the function llread(). This function reads the information received through the serial port and checks its validity. It "destuffs" the data field of the frame and validates BCC1 and BCC2 to verify if any errors occurred during transmission.

The termination of the connection is carried out by the function llclose(). It is invoked by the sender when the number of failed attempts exceeds or when the transfer of data packets is completed. The sender sends a DISC supervision frame and waits for the receiver to respond with the same frame, effectively ending its operation. When the sender receives DISC again, it responds with UA and terminates the connection.

Application Protocol

The Application Layer directly interacts with both the file to be transferred and the user. In this layer, you can set parameters like the file to transfer, the serial port to use, transfer speed, the number of data bytes in each packet, maximum retry attempts, and the maximum waiting time for the receiver's response. The actual file transfer is facilitated by the Link Layer, which translates data packets into information frames.

Once the sender and receiver perform a handshake, the entire file content is copied to a local buffer using the sendFile() function. A packet is sent by the sender contains data in a TLV format (Type, Length, Value), this format is created with the sendDataPacket() function. And it send a packet to control the frame, with the function sendControlPacket().

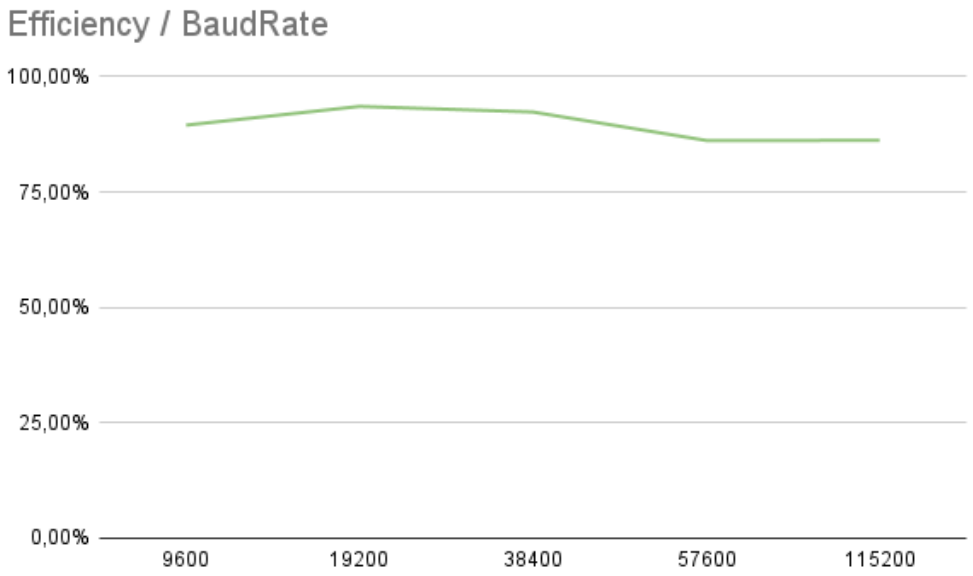
After each transmission, a response is expected from the receiver, indicating whether it accepts or rejects the packet. If it's accepted, the sender sends the next fragment; if rejected, the same fragment is resent.

Each packet is individually assessed by the receiver with receiveFile() and receiveDataPacket() function. After that, the receiveControlPacket() function extracts the control frame to know if there is any errors.

The connection between the two machines concludes when the llclose() function is invoked, either after completing the transfer of data packets or when the maximum number of failed attempts is reached.

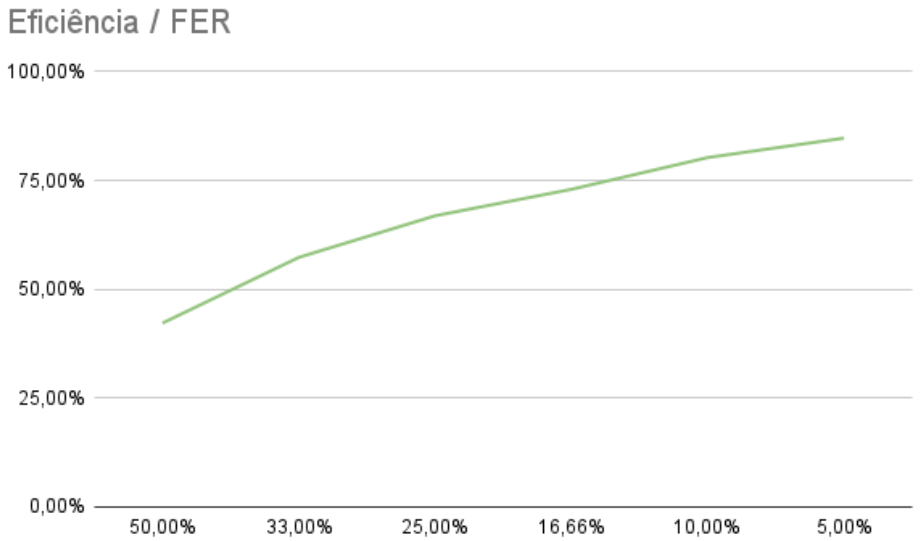
Data Link Protocol Efficiency

- **Baud Rate Variation:** With constant file and information frame sizes, varying the Baud Rate resulted in the following data:



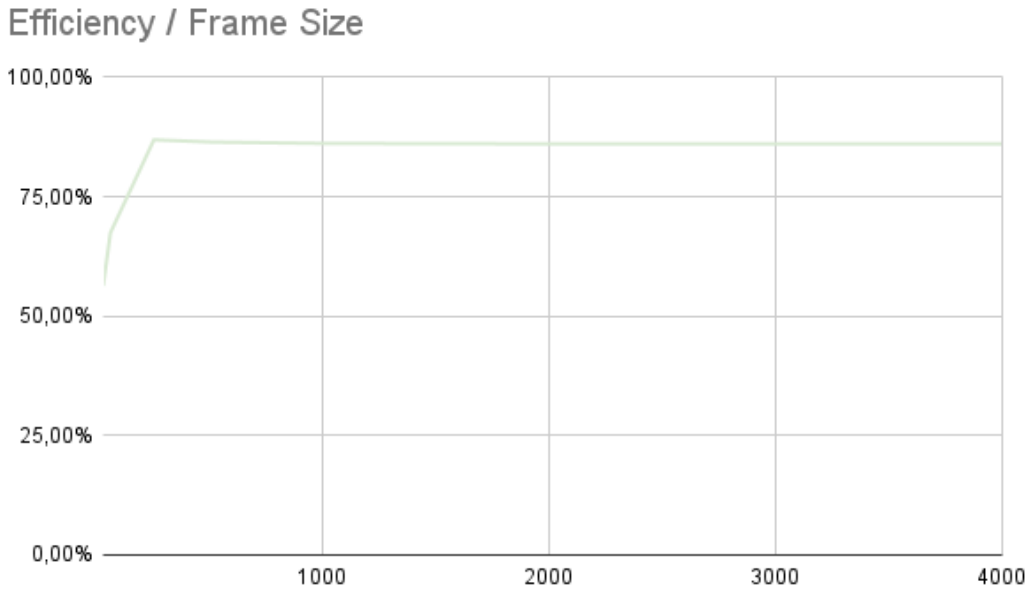
As stated in the previous graph, the protocol's efficiency gradually decreased with the increase of the baud rate, because, even if the frame propagation speed increases, the transmission time for that information also increases.

- **Frame Error Ratio (FER) Variation:** With constant file and information frame sizes and constant baud rate, varying the error ratio in the frames resulted in the following data:



As observed in the previous figure, the efficiency is inversely proportional to the FER induced during the transmission, since the Stop-And-Wait protocol triggers a retransmission once an error has been detected on the frame, which leads to an increased transfer time - which leads to more inefficient transfers.

- **Frame Size Variation:** With constant file size and constant baud rate, varying the frame size resulted in the following data:



As observed in the previous figure, the increase in the protocol's efficiency is more notable when the frame size is smaller, becoming more stable when the frame size increases.

Conclusion

The elaboration of the data link protocol, constituted by the LinkLayer (the layer that allows the communication between the serial port and the information frame manager) and by the ApplicationLayer (the layer that directly interacted with the file that was being transferred) was of extreme importance in the consolidation of the class' content. By elaborating this project, we had the chance to better understand and assimilate the concepts of byte stuffing, framing and how the Stop-And-Wait protocol works and deals with errors.