**MIEIC 3º Ano**

**Novembro 2019**

**Redes de Computadores**

**1º Trabalho Laboratorial – Ligação de Dados**

Prof. Manuel Pereira Ricardo

Prof. Rui Lopes Campos

Turma 7

João Praça - up201704748

Leonor Sousa - up201705377

Sílvia Rocha - up201704684

## Sumário

(dois parágrafos: um sobre o contexto do trabalho; outro sobre as principais conclusões do relatório)

O 1º Projeto Laboratorial foi elaborado no contexto da unidade curricular Redes de Computadores e teve como tema ligação de dados. Deste modo o objetivo do projeto passava em elaborar um programa que simulava um protocolo de ligação de dados e um protocolo de uma aplicação que tinham como objetivo a transferência de dados entre dois computadores via porta de série.

## Introdução

(indicação dos objectivos do trabalho e do relatório; descrição da lógica do relatório com indicações sobre o tipo de informação que poderá ser encontrada em cada uma secções seguintes)

Este projeto tinha como **objetivos**:

* Elaborar um protocolo de ligação de dados;
* Elaborar uma aplicação simples que permite a transferência de ficheiros, utilizando o protocolo de ligação de dados previamente elaborado.

O relatório adjacente ao projeto tem como principal objetivo esclarecer a arquitetura, estrutura e a aplicação dos protocolos utilizados no projeto, assim como fazer a validação e a análise da eficiência do código desenvolvido.

Seguidamente, apresentam-se, neste relatório, informações sobre a implementação, casos de uso e resultados deste projeto. Deste modo, o relatório encontra-se estruturada da seguinte forma:

* **Arquitetura e Estrutura do Código** – descrição da arquitetura utilizada e da estruturação do código (APIs, estruturas de dados utilizadas e funções), assim como a relação entre ambas.
* **Casos de Uso Principais** – identificação dos casos de uso mais importantes, assim como da sequência de chamada de funções relacionada com cada um.
* **Protocolos** - descrição dos principais aspetos funcionais e implementação respetiva dos seguintes protocolos:
  + Protocolo de Ligação de Dados
  + Protocolo de Aplicação
* **Validação e Eficiência do Protocolo de Ligação de Dados** – descrição dos testes utilizados e seus respetivos resultados, caracterização estatística da eficiência do protocolo e comparação da mesma com os resultados esperados.

## Arquitetura e Estrutura do Código

(blocos funcionais e interfaces)

(APIs, principais estruturas de dados, principais funções e sua relação com a arquitetura)

A arquitetura deste projeto consiste abrange 3 blocos funcionais:

**protocol** (protocol.h e protocol.c)

Bloco responsável por fazer a criação e processamento de tramas e por fazer a transmissão em si. É também neste bloco que são efetuados os mecanismos de stuffing e destuffing, assim como eventuais retransmissões e verificação da integridade das tramas.

A API deste bloco é constituída pelas seguintes principais funções:

|  |  |
| --- | --- |
| int open\_port(char\* porta, struct termios \*oldtio) | void close\_port(int fd, struct termios \*oldtio) |
| void send\_set(int fd); | void send\_ua\_rcv(int fd); |
| void send\_ua\_snd(int fd); | void send\_disc\_rcv(int fd); |
| void send\_disc\_snd(int fd); | void send\_resp(int fd, char c, char a); |
| int send\_msg(int fd,unsigned char\* msg, int length); | |
| int receive\_msg(int fd, unsigned char c, unsigned char a, bool data, unsigned char data\_buf[], bool data\_resp); | |
| void receive\_set(int fd); | void receive\_disc\_rcv(int fd); |
| void receive\_disc\_snd(int fd); | void receive\_ua\_rcv(int fd); |
| void receive\_ua\_snd(int fd); | bool receive\_data\_rsp(int fd); |
| int receive\_data(int fd, unsigned char data\_buf[]); | |

**application** (application.h e application.c)

Bloco responsável por fazer a divisão do ficheiro em diversas partes e por construir e processar os pacotes (de controlo e de dados).

A API deste bloco é constituída pelas seguintes principais funções:

|  |  |
| --- | --- |
| int llopen(unsigned char \*porta, bool transmitter); | int llclose(int fd); |
| int llwrite(int fd, unsigned char \* buffer, int length); | int llreadFile(int fd); |
| void llopen\_image(unsigned char \*path, int fd); | |

**nserial** (nserial.c)

Bloco responsável por fazer a interface entre o utilizador e o bloco application. É constituído pela função main e pode ser chamado, via consola, através da instrução:

.\nserial <portaDeSerie> <TRANSMITTER(1)|RECEIVER(0)> <ficheiro>

* <portaDeSerie> é uma string do tipo "/dev/ttySX", em que X é o número da porta de série
* O campo <TRANSMITTER(1)|RECEIVER(0)> deverá ser 1 se se tratar do transmissor e 0 se se tratar do receptor
* O campo <ficheiro> só deverá existir caso se trato do transmissor e representa o ficheiro que deve ser enviado.

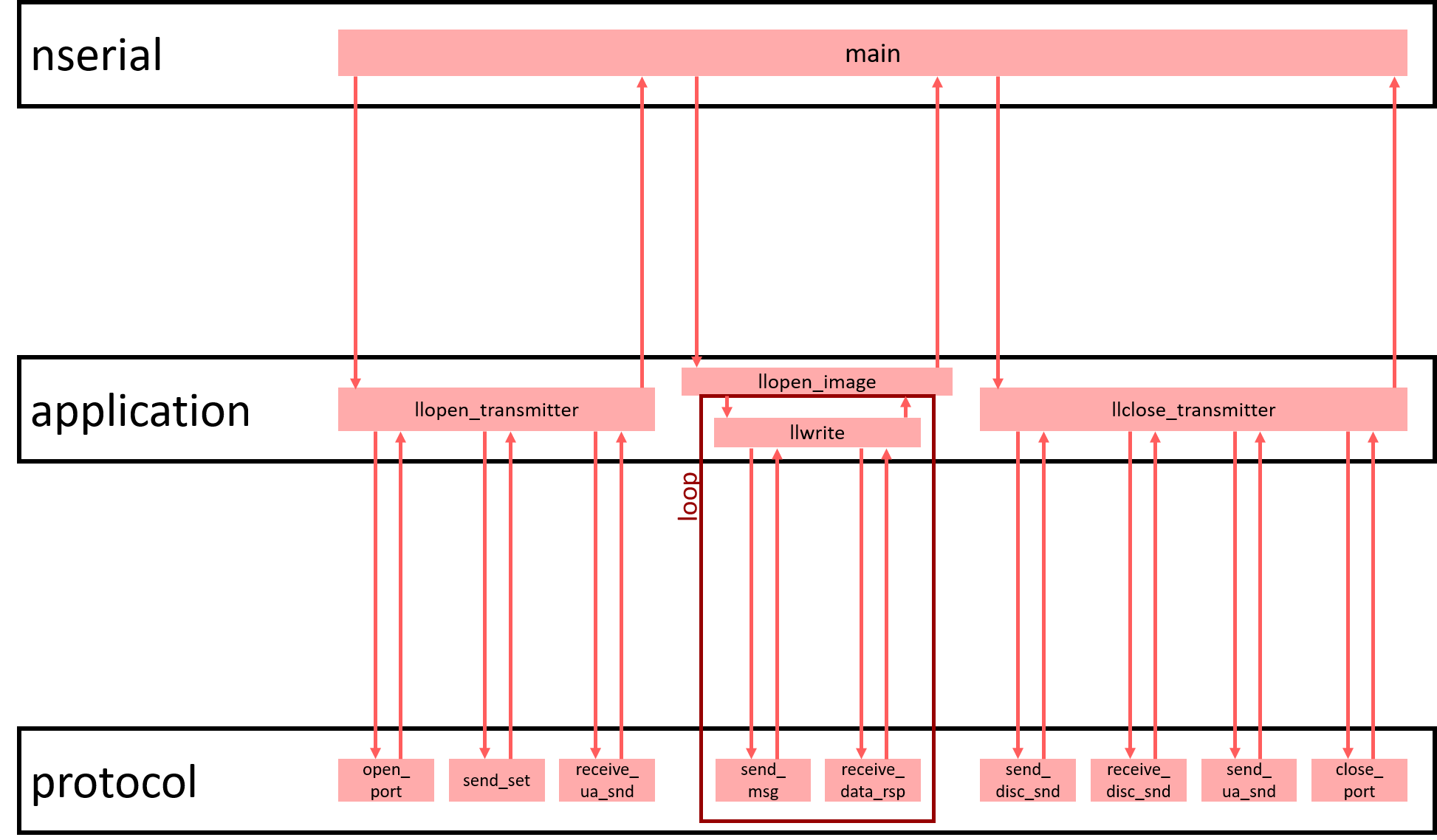
## Casos de Uso Principais

(identificação; sequências de chamada de funções)

O principal caso de uso deste programa é a transmissão de ficheiros via porta de série. Desta forma, podemos subdividir este caso em dois subcasos:

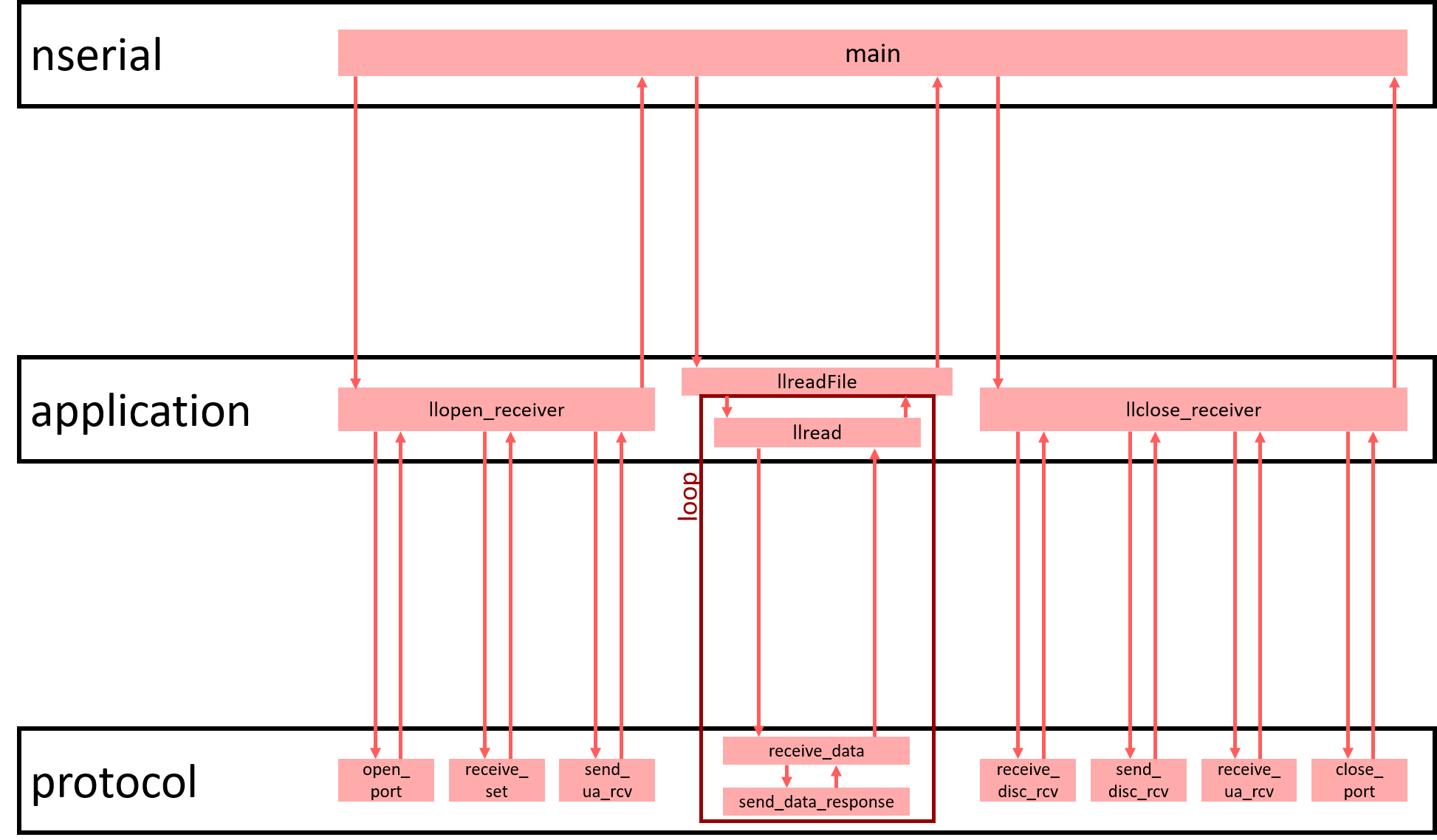
**Envio de um Ficheiro**

O envio de um ficheiro pode ser feito correndo, na linha de comandos, a instrução: .\nserial <portaDeSerie> 1 <ficheiro>. Esta instrução aciona a seguinte sequência de chamada de funções:



**Receção de um Ficheiro**

O envio de um ficheiro pode ser feito correndo, na linha de comandos, a instrução: .\nserial <portaDeSerie> 0. Esta instrução aciona a seguinte sequência de chamada de funções:



## Protocolos

Protocolo de Ligação Lógica

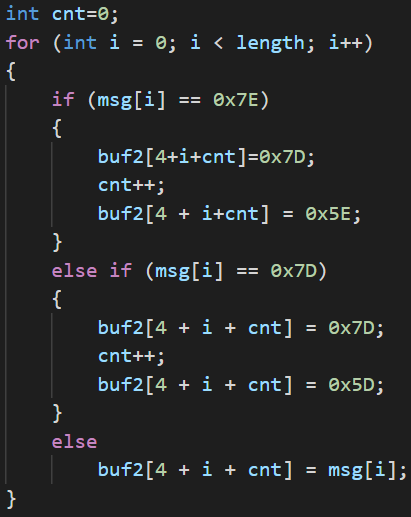
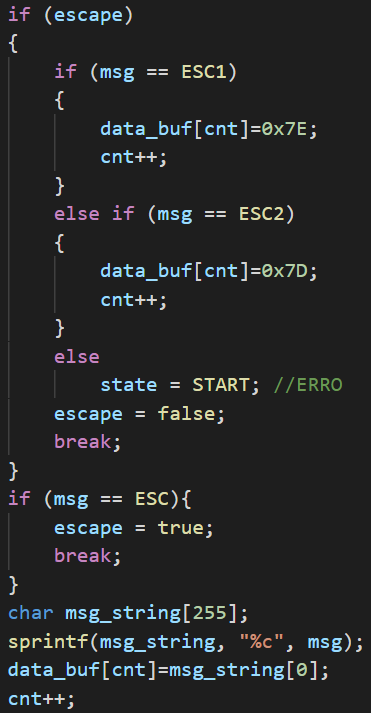
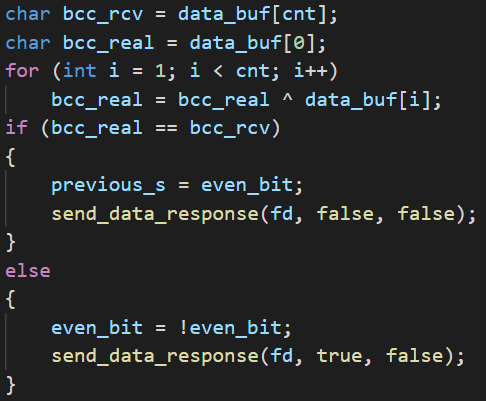
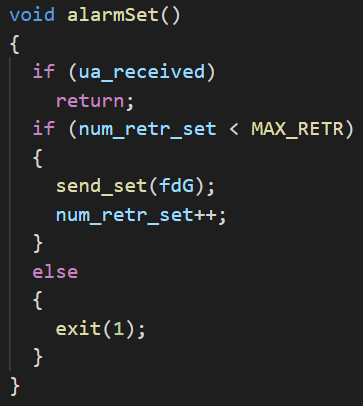
(identificação dos principais aspetos funcionais; descrição da estratégia de implementação destes aspetos com apresentação de extratos de código)

Uma das principais funcionalidades do protocolo de ligação lógico é a criação e o processamento de tramas do tipo I e de tramas de dados. Estas tramas têm um formato já especificado à priori. O seu processamento é efetuado recorrendo à função receive\_msg(), onde é implementada uma máquina de estados. A sua criação é efetuada nas funções send\_msg() (tramas de dados) e send\_resp (tramas do tipo I).

A estas tramas é aplicado byte stuffing (no envio de dados) e byte destuffing (na receção de dados). Este mecanismo é aplicado recorrendo ao octeto escape (0x7D). Desta forma, sempre que se quer enviar o byte 0x7E, são enviados os bytes 0x7D e 0x5E e, sempre que se quer enviar o byte 0x7D, são enviados os bytes 0x7D e 0x5D. Na leitura faz-se o processamento inverso.

O protocolo de ligação de dados faz também a verificação da integridade das tramas, a partir da análise do BCC e do BCC2, no caso de tramas de dados.

No caso de ocorrer um timeout na receção da resposta ao envio da trama, a trama é reenviada. Este mecanismo é efetuado recorrendo a sinais do tipo ALARM.

1. Retransmissão
2. Destuffing
3. Stuffing
4. Verificação da Integridade

Protocolo de Aplicação

(identificação dos principais aspetos funcionais; descrição da estratégia de implementação destes aspetos com apresentação de extratos de código)

O protocolo de aplicação é responsável por fazer a construção e o processamento dos pacotes (de dados e de controlo), não tendo qualquer conhecimento da estrutura das tramas e de todos os outros pormenores que fazem parte do protocolo de ligação de lógica. Na construção e desconstrução dos pacotes deve ser efetuada uma numeração correta dos pacotes.

## Validação e Eficiência do protocolo de ligação de dados

(descrição dos testes efectuados com apresentação quantificada dos resultados, se possível)

(caraterização estatística da eficiência do protocolo, feita com recurso a medidas sobre o código desenvolvido. A caracterização teórica de um protocolo Stop&Wait, que deverá ser usada como termo de comparação, encontra-se descrita nos slides de Ligação Lógica das aulas teóricas).

## Conclusões

(síntese da informação apresentada nas secções anteriores; reflexão sobre os objectivos de aprendizagem alcançados)