Simulador das Camadas Física e Enlace

Teleinformática e Redes 1 - Turma A

Aécio Fernandes Galiza Magalhães
Engenharia de Computação
15/0115121
aeciofgm@gmail.com

Fernando Sobral Nóbrega
Engenharia de Computação
15/0034911
fernando.sobral.unb@gmail.com

Guilherme Fleury Franco
Engenharia de Computação
18/0121472
guilherme.fleury26@gmail.com

Nícolas Machado Schumacher
Ciência da Computação
13/0047660
nicolasmschumacher@gmail.com

Brasília, 18 de novembro de 2019

1. Introdução

Quando um certo pacote de dados é submetido a uma transmissão, está sujeito a vários tipos de falhas. Para garantir um fluxo de dados transmitido com segurança, um dos modelos proposto é o modelo de camadas OSI. Nesse modelo, existem 7 camadas, cada uma com responsabilidades e protocolos bem definidos e que seguem uma hierarquia interdependente.

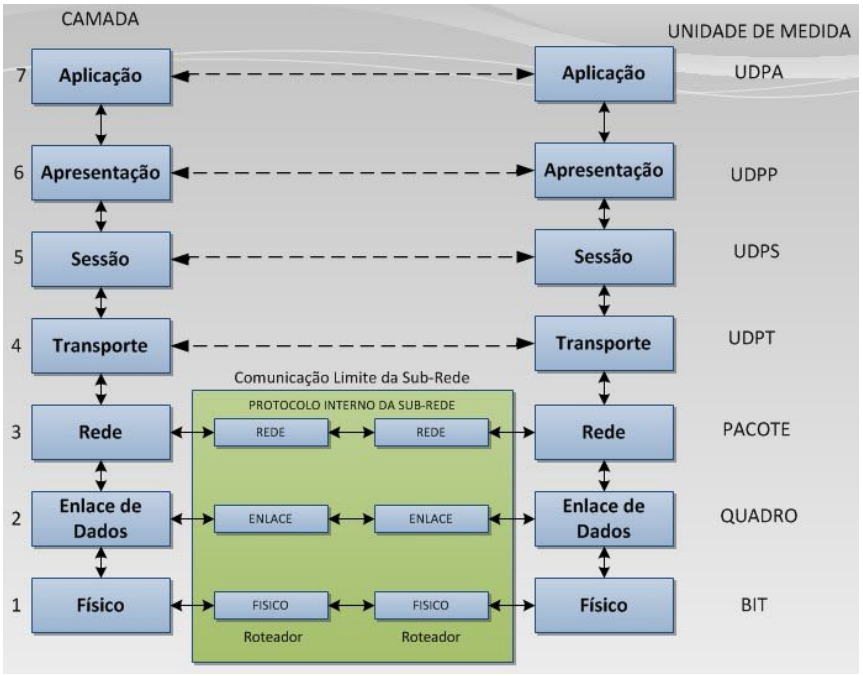

Figura 1. modelo de camadas OSI

O grupo tem como objetivo desenvolver uma aplicação que simule o funcionamento da camada física e de enlace dentro de uma transmissão de dados. Para tal, foram implementados separadamente protocolos referentes a essas duas camadas que tratam os dados tanto no caso de transmissão quanto recepção desses dados. Os protocolos requeridos foram, de acordo com cada etapa:

1.1. Camada Física

A camada física é responsável por codificar, transmitir e receber os bits, decodificando-os. Os tipos de codificação abordadas neste trabalho foram a binária, Manchester e Manchester Diferencial. Essas codificações são utilizadas para melhor adaptar a mensagem ao meio de transmissão, a fim de evitar erros de transmissão, buscando garantir a integridade dos dados por meio de protocolos robustos de comunicação entre o

transmissor e o receptor e utilizar da forma mais eficiente possível o meio de comunicação.

1.2. Camada de Enlace

A camada de enlace é responsável por detectar erros decorrentes da camada física. Pode ocasionalmente, também, ser responsável pela correção desses erros. Neste trabalho, a camada de enlace realiza o enquadramento dos dados e o controle de erros. No caso do enquadramento, foram abordados a contagem de caracteres, a inserção de bytes ou caracteres e a inserção de bits. Esses métodos criam um padrão para o quadro dentro da transmissão dos dados a fim de facilitar a identificação de erros. No caso da detecção de erros, foram abordados o bit de paridade par e ímpar, o CRC e o Código de Hamming. Tendo o dado já enquadrado de maneira correta, um desses métodos é aplicado para que algum tipo de algoritmo verificador possibilite a identificação de algum erro na hora da chegada da mensagem.

2. Implementação

Descrição detalhada do desenvolvimento com diagramas ilustrativos, o funcionamento dos protocolos, procedimentos utilizados, bem como decisões tomadas relativas aos casos e detalhes de especificação que porventura estejam omissos no enunciado.

2.1. Camada Física

2.1.1. Codificação Binária

Na codificação do tipo Binária, a mensagem inserida em ASCII pelo usuário foi traduzida traduzida para um vetor de inteiros com cada elemento do vetor representando um bit do caracter ASCII. Por exemplo, o caracter ‘a’ foi traduzido como 1100001.

2.1.2. Codificação Manchester

Na codificação do tipo Manchester foi utilizada a convenção de G. E. Thomas, assim, o quadro recebido é codificado utilizando o clock, que começa sempre em 0, aplicando as variações do clock para cada bit de

dado recebido, realizando a operação XNOR sobre essas informações de modo que o quadro resultante ficou com o dobro do tamanho anterior à codificação. Na camada receptora, bastou realizar novamente essa mesma operação para cada variação de clock sobre os dados recebidos a fim de decodificar a mensagem.

2.1.3. Codificação Manchester Diferencial

Assim como na codificação Manchester, na codificação Manchester Diferencial o quadro é codificado utilizando o clock, e, além disso, é necessária uma outra entrada referente ao bit anterior para a diferenciação do estado inicial em 0 ou em 1. Com isso, ao ser feita uma transição nos dados para o bit 1, o estado anterior do clock se mantém nesse momento, e posteriormente oscila novamente.

Caso a transição seja feita para o bit 0, o estado anterior é alterado, voltando a oscilar posteriormente.

2.2.Camada de Enlace

2.2.1. Enquadramento

2.2.1.1. Contagem de caracteres

Para a contagem de caracteres, assumimos que, pelo padrão ASCII, um caractere pode ter no máximo 7 bits. Padronizou-se, então, todos os possíveis caracteres para uma representação em 7 bits. Com isso, na contagem de caracteres, foi feita uma leitura da quantidade de bits enviados e, a partir disso, uma divisão por 7 para obter-se a quantidade de caracteres. Com isso, foi adicionado aos 7 primeiros bits do quadro um número de 7 bits em binário referente a essa quantidade.

2.2.1.2. Inserção de bytes ou caracteres

2.2.1.3. Inserção de bits

2.2.2. Controle de erro

2.2.2.1. Bit de paridade par e ímpar

O controle de erro via bit de paridade par foi implementado tendo o quadro como argumento e ele foi percorrido utilizando uma variável bitParidade, de valor inicial zero, que armazena o resultado do XOR entre bitParidade e o bit da mensagem. Assim, bitParidade será alterado toda vez que encontrar o valor 1, de modo que, no final, bitParidade será 1 quando houver um número ímpar de 1's na mensagem e 0 quando for um número par de 1's, segundo a definição do bit de paridade par. O valor retornado é o bitParidade, que deve ser concatenado à mensagem original. O bit de paridade ímpar foi implementado da mesma forma, bastou negar o valor da função de bit de paridade par.

2.2.2.2. Código de Hamming

O controle de erro via código de hamming foi implementado com 8 bits de dados 4 bits de checagem. O módulo recebe um vetor de inteiros, com cada uma das posições desse vetor podendo valer um ou zero. Por meio de um laço que avança de 8 em 8 posições até que o vetor de entrada acabe, um novo vetor vai sendo montado com os bits de checagem sendo calculados a partir de somas binárias sem carry out. Para checar se há erro em uma mensagem recebida, o processo é um pouco diferente. É feita uma comparação da soma de bits de dados com bits de checagem e, apenas se todos corresponderem o código retorna uma mensagem de sucesso. Caso contrário é lançada uma exceção.

2.3. Simulador

O simulador simulada a camada de aplicação realizando a interação com o usuário. Esse último, informa uma mensagem ao simulador que converte essa mensagem em ASCII para um vetor de inteiros e passa esse vetor para a camada física.

O simulador foi compilado usando o seguinte compilador e sistema operacional g++ (Ubuntu 7.4.0-1ubuntu1~18.04.1) 7.4.0 e utilizando os comandos *g++ CamadaEnlace.cpp CamadaFisica.cpp Simulador.cpp -o simulador*. Após isso, para rodar o programa, foi usado o comando *./simulador* e seguidas as instruções em tela para realizar os testes.

3. Membros

A seguir, separadas em tópicos, uma noção superficial das atividades desenvolvidas por cada membro do grupo:

- **Fernando:** Responsável pelo arquivo do simulador e pelos seguindos módulos da camada física: CamadaFisicaTransmissora, CamadaFisicaTransmissoraCodificaçãoBinaria, CamadaFisicaTransmissoraDecodificacaoBinaria, MeioDeComunicacao, CamadaFisicaReceptora, CamadaFisicaReceptoraDecodificacaoBinaria, CamadaAplicacaoReceptora, AplicacaoReceptora e garantir que os módulos da camada físicas se comunicam corretamente entre si.
- **Aécio:** Módulo da camada física para a conversão de ascii para binário, função da codificação e decodificação para Manchester Diferencial, função do meio de comunicação com os alinhamentos subsequentes para a camada física receptora, módulo para enquadramento de contagem de caractere na transmissão e recepção da camada de enlace
- **Guilherme:** Módulo da camada física para a conversão de binário para ascii, função transmissora e receptora do código de hamming, escrita do relatório
- **Nícolas:** Módulo da camada física para transmissão e recepção da codificação Manchester, testes, verificação de paridade ímpar e par do transmissor e receptor da camada de enlace.

4. Conclusão

O trabalho realizado alcançou seu objetivo, de simular o funcionamento das camadas física e de enlace por meio da implementação de protocolos já existentes. Ele auxiliou os integrantes do grupo a se familiarizar mais com este meio de comunicação e a verificar as dificuldades de se implementar e desenvolver tais protocolos, que devem ser pensados para realizar tarefas de forma extremamente ágil e precisa.

Uma das dificuldades foi compreender especificamente qual tipo de algoritmo utilizar em cada caso, pois alguns métodos de codificação, por exemplo, possuem mais de uma versão. Além disso, o código de Hamming foi um dos pontos mais difíceis de serem implementados, apesar de ser, de certa forma, simples de compreender seu funcionamento.