SIMULADOR DE CODIFICAÇÃO DE LINHA E MODULAÇÕES PARA COMUNICAÇÃO DE DADOS

Guilherme de Campos Lemes[1]
Pedro Henrique Serafim Raimundo[2]
Juliano Coelho Miranda[3]

RESUMO

O trabalho apresenta um simulador das modulações e codificações de linha mais utilizadas na área de comunicação de dados. O objetivo consiste em facilitar o aprendizado de comunicação de dados e redes de computadores, pois verifica-se certa ausência de exemplos práticos nessas áreas. O simulador exibe a formação de ondas analógicas moduladas ou ondas digitais codificadas, com dados obtidos a partir de números digitais binários fornecidos individualmente, caracteres dados pelo usuário ou até mesmo dados transmitidos de outro computador por meios de comunicação. A solução está em torno da implementação de um sistema didático que permita exibir os resultados das modulações ou codificações por meios gráficos de fácil compreensão. Como principal usabilidade destaca-se a área da docência em comunicação de dados, redes de computadores e em cursos a distância.

Palavras-chave: Comunicação de Dados. Codificação de Linha. Modulação.

1 INTRODUÇÃO

O forte apelo por equipamentos digitais proporcionou uma grande evolução nas redes de comunicação, visando à obtenção de informação de forma rápida e segura no formato digital.

Os sistemas de cabeamento para transmissão analógica ainda são os meios mais utilizados, em decorrência do seu baixo custo e facilidade de instalação. Esse baixo custo facilitou na comunicação direta entre equipamentos a grandes distâncias e a nível global.

Como esses meios de comunicação analógica já estão presentes em todas as áreas, foi necessária a criação de equipamentos que permitissem a transmissão de informação digital

[1] Aluno do Curso de Ciência da Computação do Centro Universitário do Sul de Minas UNIS-MG. Email: guincecarter@hotmail.com.
[2] Aluno do Curso de Ciência da Computação do Centro Universitário do Sul de Minas UNIS-MG. Email: pedro_vga@hotmail.com.
[3] Doutorando da Escola de Engenharia Elétrica de São Carlos (EESC). Email: coelhojm@unis.edu.br.

entre dois pontos por um meio de transmissão analógico. Assim foi criado o modem, objetivando a economia dos custos na transmissão de dados.

Os modems ou ECDs (Equipamento de Comunicação de dados) têm como objetivo principal a modulação dos dados de forma digital para analógico, ou seja, modulando e demodulando respectivamente por meio de métodos de modulação que podem ser por freqüência, fase, amplitude ou a combinação entre elas. Assim, a onda modulada pode ser enviada por meios de transmissão analógicos como a linha telefônica.

O objetivo deste artigo é apresentar um *software* simulador de codificação de linha e modulações de dados, que possibilita um meio didático para explicação desses sistemas e facilita o aprendizado de comunicação de dados e redes de computadores.

2 COMUNICAÇÃO DE DADOS

Comunicação de dados é a troca de dados entre dois dispositivos (computadores, telefones, dentre outros) através de algum meio (canal) de comunicação como, por exemplo, um par de fios. São transmitidos sinais que correspondem a códigos (0,1) que representam a informação digital (FOROUZAN, 2006).

Quando nos referimos às redes de comunicação, lembramos de rede telefônica. As redes de telefonia foram construídas inicialmente para serem exclusivamente responsáveis pela transmissão da voz. Essas "redes de comunicação" existem para que dados sejam enviados de um lugar para outro, são a resposta para a troca de qualquer tipo de informação nos dias atuais (DANTAS, 2002).

Os princípios de comunicação de dados podem ser explicitados de acordo com a visualização da Figura 1:

Figura 1: Modelo Genérico de Comunicação. Fonte: (DANTAS, 2002, p. 8).

O primeiro componente da Figura 1, a fonte, segundo Dantas (2002, p. 8) "uma fonte é caracterizada pela geração da informação que se deseja transmitir no sistema de comunicação", essa informação pode ser voz, sinais de rádio, dados digitais e etc. A

informação precisa ser modificada/adaptada para ser transmitida pela rede de comunicação. O transmissor é o responsável por modificar estes dados gerados pela fonte para serem transmitidos pela rede de comunicação (esta pode ser digital ou analógica). A rede de comunicação é o meio no qual o sinal transmitido irá trafegar para chegar ao receptor remoto e só então ao destino solicitado (DANTAS, 2002).

3 CODIFICAÇÃO DE LINHA

É a transformação de dados digitais ou sequência de bits (0,1) em sinais digitais utilizando algum tipo de codificação. Como explicitado anteriormente, qualquer dado em um computador está em formato digital (Imagens, vídeos, texto, entre outros), a codificação transforma esses dados digitais em sinais digitais, conforme mostra a figura 2.

Figura 2: Codificação de linha

As codificações podem ser divididas em Unipolar, Polar e Bipolar.

3.1 Unipolar, polar e bipolar

Segundo Forouzan (2006, p. 107), o método unipolar é o "método mais simples e primitivo, utiliza somente um nível de tensão onde um pulso representa 1, chama-se unipolar pois utiliza uma polaridade apenas."

O método Polar, segundo Forouzan (2006, p. 107) "utiliza dois níveis de tensão, um positivo e outro negativo, para representar os dados", este método engloba algumas codificações, conforme figura 3.

Figura 3. Codificação Polar e suas divisões. Fonte: (FOROUZAN, 2006, p. 108).

Já o método Bipolar, segundo Forouzan (2006, p. 110) “utiliza três níveis de tensão: positivo, negativo e zero”. O nível zero representa um bit 0, os bits 1s são representados através de pulsos alternados de tensão positiva e negativa. Um desses métodos mais utilizados é o *Alternate Mark Inversion* (AMI).

4 MODULAÇÃO

Trata-se da modificação das características de uma onda digital (portadora) segundo uma função modulante para esta ser transmitida em um meio analógico. (DANTAS, 2002) Segundo Forouzan (2006, p. 129) “os dados originais estão na forma digital dentro do computador, mas os meios metálicos das linhas telefônicas podem transportar apenas sinais analógicos”. Os dados digitais devem ser modulados em um sinal analógico de acordo com regras que permitam diferenciar claramente entre os valores 0s e 1s do sinal original, como pode ser visualizado na Figura 4 (FOROUZAN, 2006).

Figura 4. Modulação digital-analógico. Fonte: (FOROUZAN, 2006, p. 130).

A modificação de algum aspecto (amplitude, freqüência ou fase) de um sinal elétrico permite representar convenientemente os dados binários. Qualquer um dos aspectos supracitados pode ser alterado, isso possibilita pelo menos três mecanismos para modulação de dados em sinais analógicos, que são: *Amplitude Shift Keying* (ASK), *Frequence Shift Keying* (FSK) e *Phase Shift Keying* (PSK), fluxograma dessas modulações conforme Figura 5. (FOROUZAN, 2006)

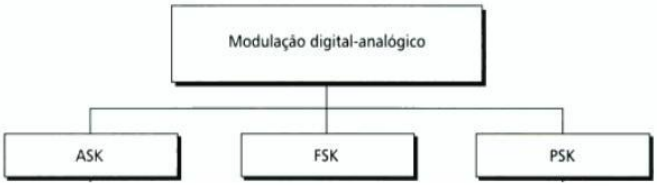

Figura 5: Modulações digital-analógico. Fonte: (FOROUZAN, 2006, p. 130).

Numa transmissão analógica, o dispositivo transmissor produz um sinal de alta freqüência que funciona como suporte para o sinal de informação. Este sinal suporte é denominado onda portadora, o dispositivo transmissor emite a onda para o receptor de forma que o receptor configure sua onda portadora para a mesma freqüência do transmissor (FOROUZAN, 2006).

Depois da troca de informação da portadora, o transmissor modifica informações nessa onda (amplitude, freqüência ou fase) e envia a nova onda modulada para o receptor, que terá que decifrar ou demodular esta onda (comparando-a com a portadora inicialmente recebida) para poder "entender" a informação.

4.1 ASK, OOK, FSK e PSK

Na técnica ASK (codificação por deslocamento de amplitude), a intensidade ou a amplitude do sinal da portadora varia de modo a representar a informação binária 0 ou 1 ou segundo Stallings (2002, p. 339) "os dois valores binários são representados por duas amplitudes diferentes". Tanto a freqüência como a fase permanecem constantes enquanto apenas a amplitude sofre variações (FOROUZAN, 2006).

Outra técnica de modulação por amplitude é a OOK, nesse tipo de modulação o nível 0 ou o nível 1 são representados por um nível de tensão 0v. A vantagem desta modulação é que não há transmissão quando a informação é 0, possibilitando uma economia de energia (FOROUZAN, 2006).

Na técnica FSK (codificação por deslocamento de freqüência) a freqüência do sinal da portadora varia de modo a representar os níveis binários 0 ou 1. O valor da freqüência em cada intervalo depende do bit representado, tanto o valor de amplitude quanto a fase permanecem inalterados (FOROUZAN, 2006).

Já na técnica PSK (Codificação por deslocamento de fase), a fase da portadora é variada de modo a representar os níveis 0 ou 1. Além disso, tanto a amplitude quanto a freqüência permanecem constantes enquanto a fase estiver variando. Segundo Forouzan (2006, p. 135) "se partirmos do pressuposto que uma fase igual a 0º representa o binário 0, então podemos variar a fase para 180º de modo a enviar o binário 1".

5 OUTROS SIMULADORES

Podem ser encontrados outros simuladores na área de comunicação de dados e redes de computadores, a maioria envolvendo o uso de outros *softwares* para geração das ondas como o *Matlab* e o *LabView*, a desvantagem do uso desses sistemas é que além de exigência de maior conhecimento técnico pela necessidade de conhecer a plataforma há a necessidade de possuir a licença desses ambientes. Na figura 6 pode ser visualizado um breve histórico dos simuladores na área de comunicação de dados.

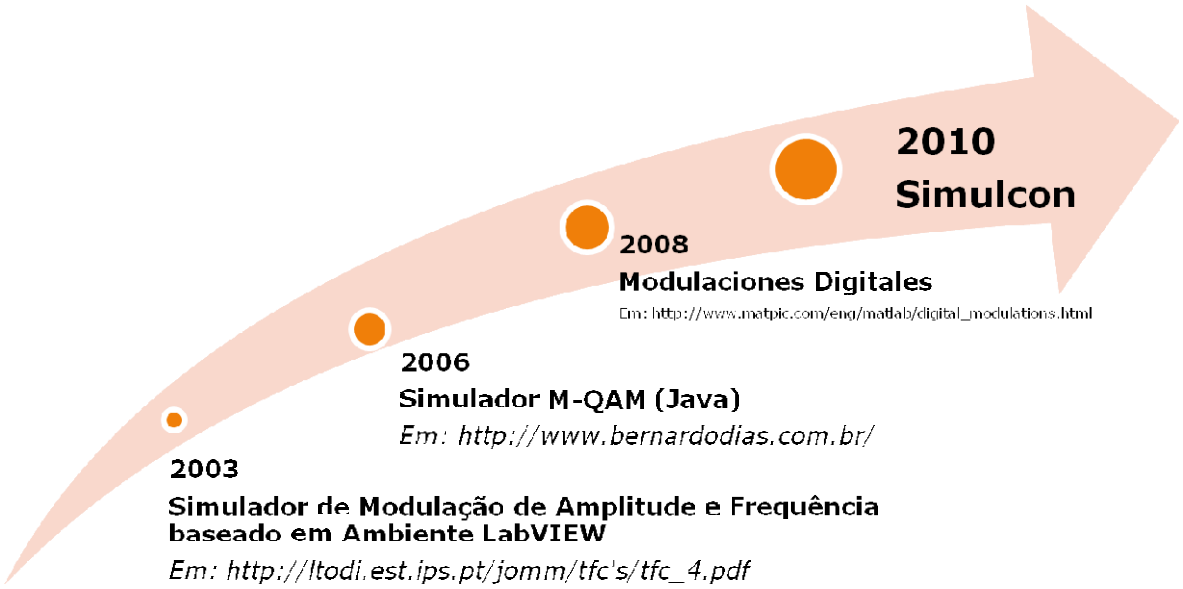

Figura 6: Histórico de simuladores

A proposta é visualizar as vantagens desses *softwares* simuladores que já são amplamente utilizados no mercado e desenvolver um novo sistema buscando atender o maior número de modulações e codificações de linha, além do adicional de permitir a comunicação de dados entre dois computadores.

6 MÓDULOS DO SIMULADOR

O simulador foi todo dividido em módulos para mais fácil programação e melhor organização de código, sendo estes módulos de codificação, comunicação e modulação, a figura 7 mostra as subdivisões do simulador, bem como as codificações de linha e modulações trabalhadas.

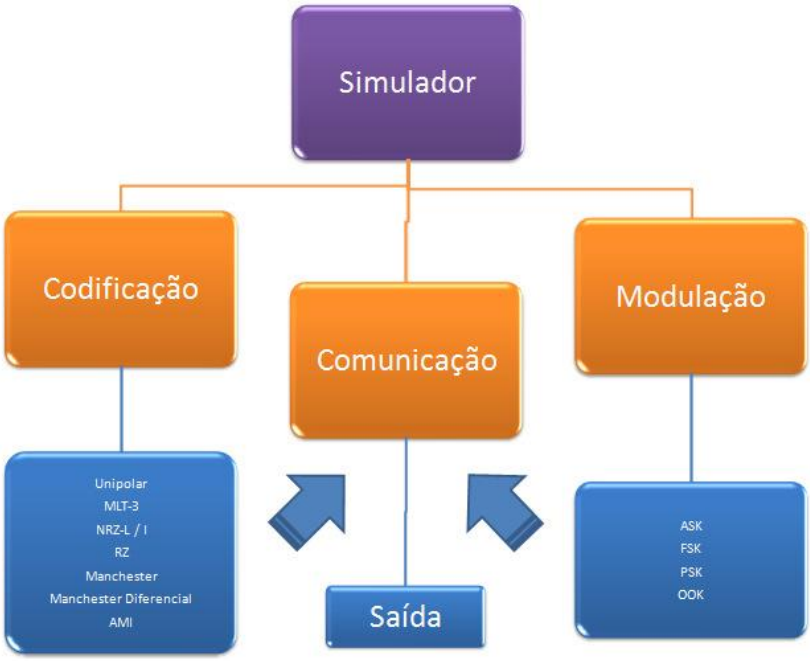

Figura 7: O simulador e seus módulos.

O desenvolvimento do simulador foi na linguagem C# e na plataforma de desenvolvimento *.Net*, que é desenvolvida pela Microsoft e engloba ferramentas e linguagens de programação, além de diversas tecnologias para o desenvolvimento de *softwares* (GALLUPO; MATHEUS; SANTOS, 2004).

Foi optado pela linguagem C# e a plataforma *.Net*, pois o usuário não precisará adquirir nenhum outro *software* para execução do simulador, além da *.Net* propiciar melhores ferramentas para facilitarem a comunicação com o usuário (utilizando botões, caixas de texto e imagens).

6.1 Codificação, modulação e comunicação

A codificação envolve toda a parte de codificação de linha, como identificação das entradas e desenho das ondas, como a codificação de linha gera ondas digitais, as ondas são formadas por conjuntos de retas.

A modulação envolve toda a parte de modulação, como identificação das entradas e desenho das ondas, como a modulação resulta uma onda analógica, as ondas são formadas por conjuntos de pontos utilizando o método *DrawCurve* em C#.

A comunicação é responsável por garantir a conexão entre duas máquinas, este módulo está dividido em três submódulos, de acordo com o tipo de comunicação, sejam elas RS232, *Bluetooth* ou *Wi-Fi*.

7 PROGRAMAÇÃO

Tendo como base outros sistemas que desenvolveram informações de ondas gráficas na tela de forma dinâmica, foram propostos classes e métodos para utilização dessas informações. Partindo deste principio o sistema então foi classificado de forma a facilitar sua manutenção e suporte futuro. Sua classificação depende exclusivamente do estado em que o programa se encontra e suas rotinas.

A primeira classificação foi baseada nas rotinas de declaração de classes e variáveis globais, eventos dinâmicos e construtores de imagens de acordo com a figura 8. Na declaração de classes, foram criados os objetos de forma privada o *Server*, *Client_A* e *Client_B*, que herdam informações da classe *Csocket*. Em destaque nesta rotina existem os *bits*, que foram declarados como *byte* que por sua vez são os principais manipuladores das informações geradas na tela. É através desta declaração que se transforma os valores que foram convertidos de Hexadecimal, ASCII e decimal para um único bit. Os eventos dinâmicos por si, já se definem como eventos que são instanciados de acordo com o que é chamado. Neste caso há o que será chamado quando um objeto *Server* precisa acessar um objeto *Client_X*.

```
CServerSocket Server = null;
CClientSocket Client_A;
CClientSocket Client_B;

delegate void SetTextCallback(string text);

public Bitmap bmp1;//Construtor de Bitmap
public Bitmap bmp2;//Construtor de Bitmap
public Bitmap bmp3;//Construtor de Bitmap RESERVA

public byte b1 = 0;
public byte b2 = 0;
public byte b3 = 0;
public byte b4 = 0;
public byte b5 = 0;
public byte b6 = 0;
public byte b7 = 0;
public byte b8 = 0;
public byte cima = 0;
```

Figura 8: Rotinas de construção

A segunda classificação foi baseada em informações e arquivos de entrada, como números e letras, os tipos de entradas que irão corresponder a uma saída de vários tipos (Codificação e Modulação). Estas entradas podem ser do tipo *Random*, *Decimal*, Hexadecimal ou *String*. De acordo com as escolhas tomadas pelo usuário, o sistema executa exceções para que retorne um valor possível de ser transmitido. Um exemplo é quando um usuário tenta escrever uma letra no campo de *bit* é então disparado uma exceção de entrada.

As *strings* são transformadas para códigos ASCII, para que se execute de forma correta esta formatação foi criada uma rotina que de conversão através dos comando de *foreach* e *Convert.ToString*(*char*(convertido para *int*), 2(base binária). Para o caso do randômico como já citado é feito da mesma forma porém o que muda é a base fornecida passando para 16(*hex*). Já o decimal é convertido de forma diferente pois utiliza o *Convert.ToByte*(*String*).

A terceira classificação foi desenvolvida para que se validem todos os erros possíveis de entrada, tanto os já citados como os possíveis erros de conversão. Um exemplo é que a cadeia de *bits* tem que ser validada. O sistema deve colocar barreiras para não permitir erros nas entradas.

A quarta parte do sistema é dada com o termo *Kernel*. É atraves deste motor que foi desenvolvida toda a rotina de escrita na tela de acordo com suas entradas validadas e objetos que foram instanciados. A principal função do *Kernel* é trabalhar com os bits de entrada e o objeto *Graphics*. Os *bits* são as informações que serão validas em cada *switch* para a escrita de um *bit* 0 ou 1 na tela, levando em consideração o tipo de saída escolhido.

Esta classificação é importante pois ela coordena *pixel* a *pixel* na tela. Para ser escrita uma linha na forma digital no caso do bit 1, é utilizada a seguinte função, *digital.DrawLine*(c(*bit*),150,102,150,23). Percebe-se que cada número corresponde a uma coordenada cartesiana.

Ao fim, todo o conteúdo é exibido a onda modulada/codificada na tela. Porém para que seja feito da forma correta, utilizando o efeito de velocidade de escrita e criando a onda de acordo com que vai escrevendo o bit, foi utilizado um “truque” de programação. Este “truque” nada mais é do que uma utilização de atualização de *Threads*, ou seja, utiliza-se atualização de eventos no instante onde a escrita é executada, então, de acordo com o que foi criado na tela, dinamicamente é disparada uma atualização de imediato na tela, fazendo com que o usuario tenha a sensação que esta onda está sendo criada de forma dinâmica. Levando em consideração um componente que controla a taxa de atualização, foi criado uma *TrackBar*. Este componente controla o fluxo de velocidade de exibição das imagens na tela. Para completar a escrita foi criado também um acionamento exclusivo para exibir no exato momento qual *bit* esta sendo exibido na tela, conforme figura 9.

```
for (int tam = 0; tam <= 600; tam++)
{
    ligando os led para ver qual bit esta modulando

    picanalogica.Size = new System.Drawing.Size(tam, 200);              //incremento de tamanho da pb
    picanalogica.Image = bmp2;                                          //escrita da imagem
    if (cmbmodulacao.Enabled)
    {
        lblModulando.Text = "MODULANDO " + (porcentagem).ToString("##,##0.00") + "%"; //formatação de porcentager
    }
    else
    {
        lblModulando.Text = "CODIFICANDO " + (porcentagem).ToString("##,##0.00") + "%"; //formatação de porcentag
    }
    porcentagem += 0.1665;                                              //incremento de porcentamge 1/6
    Application.DoEvents();                                             //executa no exato momento o evento
    Thread.Sleep(Math.Abs(velocidadeModulcao.Value-25));                //pausa para escrita
}
```

Figura 9: Controle de velocidade

7.1 Comunicação

Foram desenvolvidas classes separadas para o módulo de comunicação. RS232 e *Bluetooth* utilizam a mesma classe (serial) por utilizarem o mesmo padrão de comunicação, para a *Wifi* e *Eth0*, foi implementado a classe *CClientSocket* e *CServerSocket*. Na classe serial foram implementados atributos como Porta, Velocidade, *Open*, *Close*, *Bits*/Dados, Paridade e *Bit*/Paridade. A troca de informações com o outro computador, é feita instanciando um componente chamado *SerialPort*, que possibilita a troca de mensagens entre as máquinas, conforme *sPort.ReadLine()* e *private void sPort_DataReceived()* para recebimento e *sPort.WriteLine()* para envio. Antes de qualquer envio é feito um processo de verificação de status de todos os atributos e se o outro computador esta síncrono com a comunicação. No caso do *Bluetooth*, é necessária a criação de uma Topologia PAN, esta topologia utiliza o mesmo componente de hardware que o RS232, facilitando assim a configuração por parte da programação. Nas outras duas classes foi desenvolvido um algoritmo baseado no conceito de *Socket*, sendo utilizado em Cliente/Servidor, conforme figura 10. Suas rotinas funcionam como escuta e transmissão, ou seja, recebimento e envio.

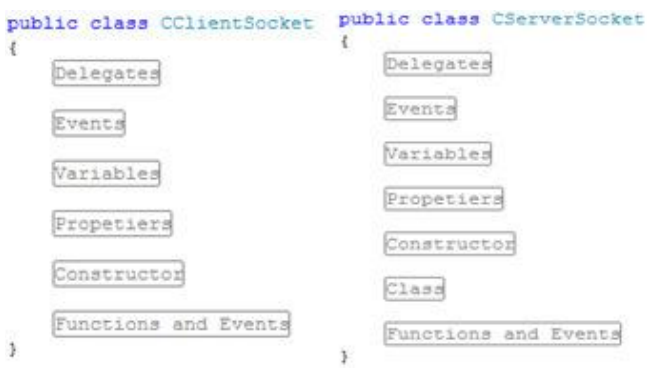

Figura 10: Cliente *Socket*

8 O SIMULADOR

Inicialmente a entrada possível para geração das ondas era uma seqüência de 8 *bits*. Com o desenvolvimento de novas versões, foi adicionada comunicação de dados e novos tipos de entrada (texto e hexadecimal) e por fim a parte de codificação de linha.

O desenho das ondas é todo feito de forma manual, utilizando o método *Draw*, esse método permite o desenho de qualquer formato de figura em um espaço delimitado, além de facilitar o desenho de retas e curvas informando-se apenas dois pontos

O êxito obtido com a parte de simulação permitiu o desenvolvimento de outras funcionalidades ao programa como por exemplo:

- A criação de bits randômicos.
- A possibilidade de se salvar uma modulação ou codificação resultante.
- A entrada de dados em hexadecimal e texto.
- Comunicação de dados.

Uma imagem da tela principal do programa e suas funções principais podem ser visualizadas na Figura 11.

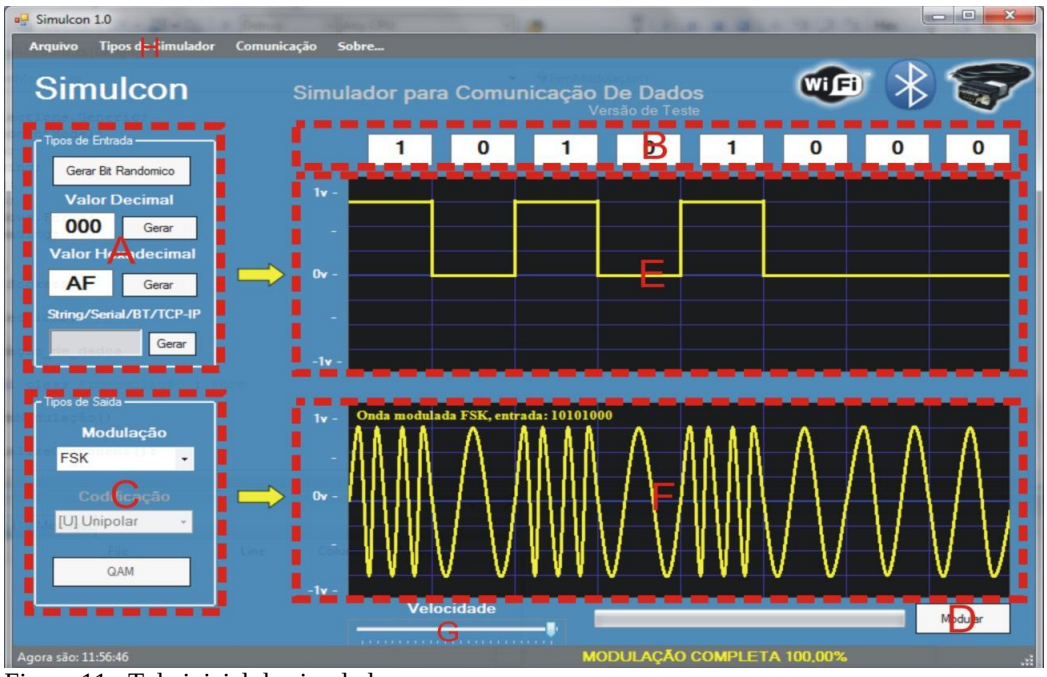

Figura 11: Tela inicial do simulador

Divisões do sistema:

A- Entrada de dados:

A entrada de dados pode ser feita das seguintes formas:

- Colocando-se os bits individualmente (Conforme B).
- Colocando-se um valor decimal até 255 (1 *byte*).
- Colocando-se um valor hexadecimal.
- Colocando-se uma string.
- Recebendo uma string por meio de comunicação porta serial.
- Recebendo uma string por meio de comunicação Bluetooth.
- Recebendo uma string por meio de comunicação por sockets.

De acordo com a entrada, os dados são calculados e convertidos para bits para assim serem apresentados nas duas caixas de imagem (picturebox) do sistema.

B- Entrada bit-a-bit.

C- Local para escolha da modulação ou codificação de linha que será feita pelo sistema.

D- Botão que modula ou codifica o sinal da entrada e gera as imagens em E e F.

E- Exibição da onda digital unipolar.

F- Exibição da onda analógica modulada ou da onda digital codificada (de acordo com a entrada). Isso representa a comunicação entre um ETD e outro ETD, ou seja, a transferência de dados por um meio analógico de comunicação de dados como, por exemplo, a linha telefônica. Ou, se selecionado o módulo de codificação de linha, é desenhada a onda digital codificada de acordo com uma das codificações selecionadas.

G- Seleciona a velocidade em que as ondas serão desenhadas em E e F.

H- Campos para seleção de codificação de linha ou modulação de dados.

9 TOPOLOGIA DE ENSAIO

A topologia utilizada para os testes no simulador pode ser visualizada na figura 12.

Figura 12: Topologia de Ensaio

Os requisitos utilizados em cada teste podem ser verificados na figura 13.

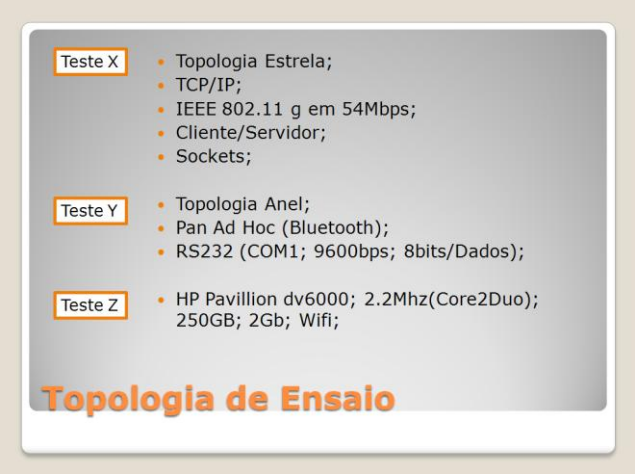

Figura 13: Requisitos de cada teste

Os resultados dos testes podem ser verificados na tabela 1.

TABELA 1: RESULTADO DOS TESTES

Critério/Teste	X	Y	Z
Comunicação	OK	OK	OK
Modulação	OK	OK	OK
Codificação	OK	OK	OK
Saídas	OK	OK	OK

As imagens de saída geradas pelas simulações podem ser visualizadas na figura 14 e na figura 15.

Figura 14: Saída gerada no teste de codificação

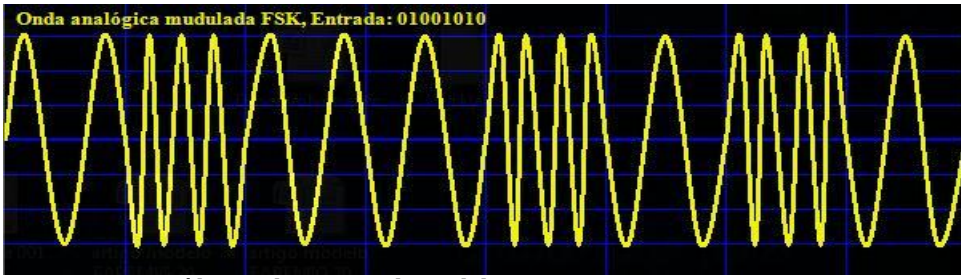

Figura 15: Saída gerada no teste de modulação

10 PROJETOS FUTUROS

Esse projeto pode ser ampliado de diversas formas, desde a adição de novas modulações ou codificações de linha até a implementação de novas áreas de comunicação de dados ou redes de computadores.

Para um futuro próximo, a adição da modulação QAM (Junção da PSK e da ASK) é o maior desafio, pois exige a utilização de novas bibliotecas gráficas como o *MsChart*.

Outro projeto futuro é o uso efetivo do sistema como simulador, facilitando a explanação de modulações. Também a ampla distribuição desta ferramenta pela Web, já que não há nenhum simulador desse porte em formato executável e que tenha acesso a porta serial e sockets. Também vale ressaltar a distribuição do sistema para outras instituições de ensino.

11 CONCLUSÃO

Todos os testes de comunicação e de geração das ondas realizados obtiveram êxito, as ondas foram geradas perfeitamente e nenhum problema foi detectado, seja na comunicação ou no desenho das ondas.

Comunicação de dados via RS232, *Bluetooth* e *Wi-FI*, possibilidade de simulação de codificações de linha, desenho dinâmico das ondas geradas e a possibilidade de se salvar alguma onda gerada foram vantagens desse simulador que não foram encontradas em nenhum outro *software* pesquisado, vale ressaltar também que não é necessária a instalação de nenhum *software* adicional para a execução do sistema. Como desvantagem vale ressaltar a ausência da modulação QAM, pois necessita do envolvimento de novas bibliotecas.

Os objetivos definidos no inicio do projeto foram cumpridos com êxito e o simulador está pronto para ser utilizado em grande escala e em cursos de EAD ou presenciais na área de computação, sistemas de informação e telecomunicações.

12 AGRADECIMENTOS

À FAPEMIG pela bolsa de iniciação científica, ao Unis-MG pelo material fornecido e aos professores que auxiliaram no desenvolvimento do simulador e do artigo.

SIMULATOR OF LINE CODIFICATION AND MODULATION FOR DATA COMMUNICATIONS

ABSTRACT

This work presents a simulator for most used modulation and line codification in data communication area. The objetive is to facilitate the learning of data communications and computer networks, because there is a certain lack of practical examples in these area. The simulator show the formation of modulated analogic waves or codificated digital waves , with data obtained from binary digital numbers supplied individually, character given by the user or even another computer data transmitted.The solution is around the implementation of an educational system that allows viewing the results of modulations or codifications with easy understanded graphic pictures. The main usability is the area of data communications,computer network and distance larning courses.

Keywords: *Data Communication. Line Codification. Modulation*

REFERÊNCIAS

COMER, Douglas E. **Internet Working with TCP/IP**. 4. ed. [S. l.]: Prentice Hall, 2000.

DANTAS, Mário. **Tecnologias de Redes de Comunicação e Computadores**. Rio de Janeiro: Axcel Books, 2002

FOROUZAN, B. A. **Comunicação de Dados e Redes de Computadores**. 3. ed. São Paulo: Bookman, 2006.

GALLUPO, F.; MATHEUS, V.; SANTOS, W. **Desenvolvendo com C#**. Porto Alegre: Bookman, 2004.

MALBURG, M. M. **Modulação**, 2004. Disponível em: <http://www.gta.ufrj.br/grad/04_2/Modulacao/ >. Acesso em: 30 nov. 2010.

Modulação ASK, FSK, PSK e QAM, 2008. Disponível em: <http://www.oficinadanet.com.br/artigo/1017/modulacao_ask_fsk_psk_e_qam>. Acesso em: 29 nov. 2010.

STALLINGS, William. **Redes e sistemas de comunicação de dados**. 5. ed. São Paulo: Campus, 2005.

STALLINGS, William. **Data and computer communications**. 8. ed. Upper Saddle River/ NJ: Prentice Hall, 2006.