

EDITOR GRÁFICO DE CIRCUITOS DIGITAIS EM MODO *PROTOBOARD*

José dos Santos Filho; Sidnei Araújo Júnior; Leonardo Rodrigues de Lima Teixeira.
IFRN – Câmpus Currais Novos
josecrrn@gmail.com; sidnei.ifrn@gmail.com; leonardo.teixeira@ifrn.edu.br

RESUMO

Este trabalho visa o desenvolvimento de um editor gráfico de circuitos digitais em modo *protoboard*, com o objetivo de tornar muito mais próximo a prática da teoria. O editor irá possibilitar a montagem de circuitos no computador de forma mais

realista que os editores convencionais, devido sua forma gráfica com o desenho mais realista possível de uma *protoboard* tal como seus recursos que adicionam e removem componentes com grande praticidade.

PALAVRAS-CHAVE: Editor de circuitos digitais, *protoboard*, *software*, eletrônica.

GRAPHICAL EDITOR OF DIGITAL CIRCUITS IN BREADBOARD MODE

ABSTRACT

This work aims to develop a graphical editor of digital circuits in breadboard mode, aiming to become much closer to the practice of theory. The editor will allow for the assembly of circuits on the computer more

realistically than conventional editors due to its graphic form with drawing realistic as possible a breadboard as resources that add and remove components with great practicality.

KEY-WORDS: Digital circuits editor, breadboard, software, electronic.

1 INTRODUÇÃO

Na área educativa de eletrônica existem três pontos de muita importância para formação do aluno:

1. Conteúdo passado em sala para os alunos;
2. Simulação em sala de circuito eletrônico;
3. Prática no laboratório.

Para a simulação temos o auxílio de simuladores de circuito eletrônico que se encontram no mercado, uns comerciais, e outros como *softwares* livres. Entre esses simuladores temos alguns principais: *CircuitMaker*, *Multisim*, *EWB*, *Proteus* entre outros. Todos esses simuladores simulam de forma abstrata as portas lógicas trazendo apenas para a prática a semelhança em relação ao circuito lógico. Observando a Figura 1 veremos um exemplo de circuito (circuito multiplexador) feito no simulador *CircuitMaker*.

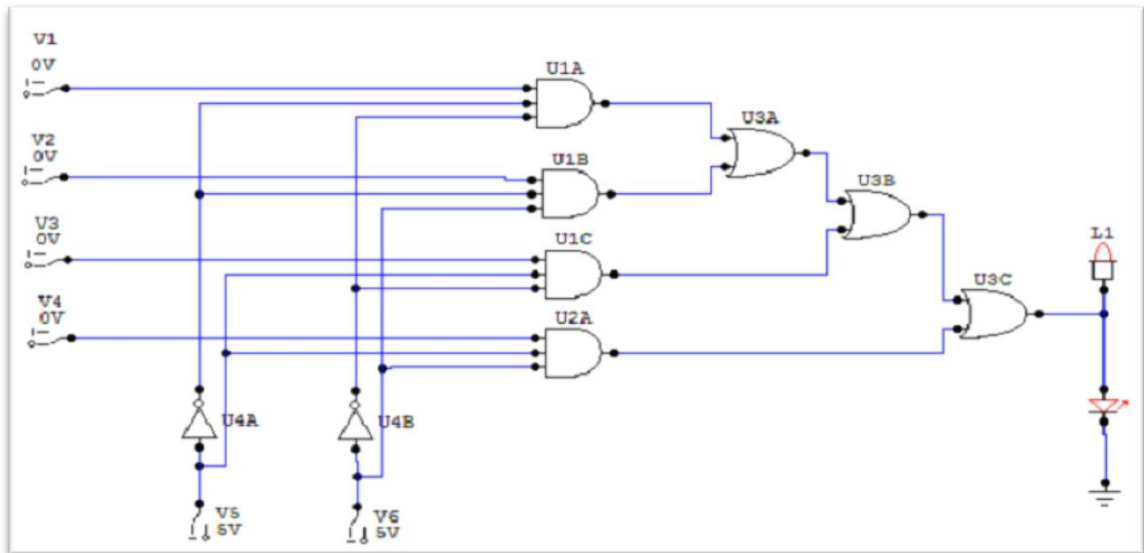


Figura 1 - Circuito multiplexador simulado pelo *CircuitMaker*

Inicialmente essa forma é muito boa e de fácil entendimento, mas o problema é quando vamos passar da simulação na sala e for para a prática em laboratório. Na prática utilizamos os CI's (Circuitos Integrados) e esses têm duas ou mais portas lógicas enquanto que na simulação tratávamos de uma porta lógica de cada vez.

Quando falamos de Circuitos Integrados devemos compreender sua estrutura interior para que assim possamos utilizá-los para montar nossos circuitos eletrônicos. Podemos encontrar essa informação nos *datasheet* que são manuais que indicam as portas lógicas que temos em cada CI. A seguir teremos na Figura 2 o *datasheet* do CI 7408, na Figura 3 o do CI 7408 e na Figura 4 o do CI 7432.

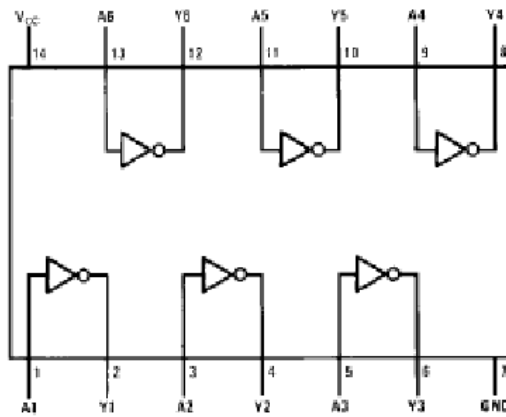


Figura 2 - Circuito integrado 7404

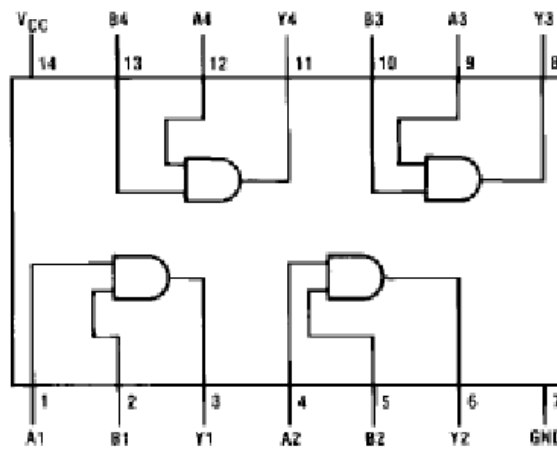


Figura 3 - Circuito integrado 7408

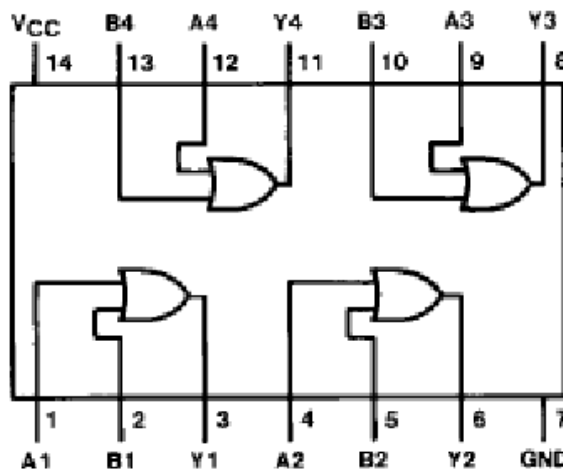


Figura 4 - Circuito integrado 7432

Observe que no 7404 temos 6 portas NOT, no 7408 temos 4 portas AND e no 7432 temos 4 portas OR. Se fossemos fazer o multiplexador (Figura 1) na prática teríamos que utilizar apenas um CI 7404, já que necessitamos de duas portas NOT (quatro ficariam sem utilização), um CI 7408, do qual utilizaríamos as suas 4 portas AND, além de um CI 7432, pois necessitaríamos de três portas

OR (uma ficaria sem utilização). Ainda existiria uma complicação para o aluno. (Sabendo quais CI's devem ser utilizados, falta entender como fazer as “ligações” entre os componentes, pois na *protoboard*, ou matriz de contatos, temos algumas regras específicas para que não ocorram curto circuitos e nem circuitos abertos e os alunos ainda não estão habituado com isso, o que deixa as aulas práticas muito extensas.

Observando a Figura 5 a seguir, que representa um circuito somador, e lembrando-se da nossa Figura 1 podemos perceber a diferença entre a simulação e a prática.

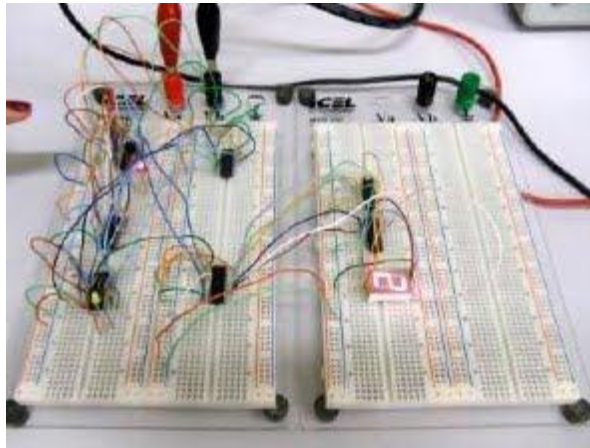


Figura 5 - Circuito somador desenvolvido em *protoboard*

Seria relevante se os alunos pudessem realizar as simulações em sala de uma forma semelhante ao que será encontrado no laboratório. Isso traria muitos benefícios e vantagens, como por exemplo:

1. Segurança na montagem dos circuitos eletrônico.
2. Queda no tempo gasto na prática com as montagens dos circuitos
3. Diminuições em danos nos componentes.

Nosso trabalho tem como objetivo a continuação e melhora do *software* OSDAM, que visa possibilitar, de forma mais simples e de melhor entendimento, o desenho de circuitos digitais, e porque não dizer um programa com uma facilidade de manuseio muito maior.

2 TECNOLOGIAS ENVOLVIDAS

2.1 XML

O XML é uma tecnologia versátil, pois pode se torna tudo aquilo que um documento possa apresentar como necessário para que haja a distribuição de informações pela rede de internet e em aplicativos (*softwares*). Sendo assim podemos usar essa tecnologia para difundir pela web as melhoras de nosso trabalho (sistema). Tittel (2003) diz que o XML permite de forma simples que interagjamos com essas estruturas e suas trocas de dados.

Podemos ler no Almeida (2002) que o XML permite pelo editor do arquivo fazer suas próprias marcas, facilitando a disseminação da informação e também uma melhor “qualidade” semântica, sendo essa uma importante característica.

Para a utilização (leitura e escrita) do XML optou-se pelo uso do DOM (DocumentObjectModel). DOM é um modelo baseado em árvores que armazena os dados do documento em uma hierarquia de nodo, assim todos os dados ficam na memória, conseqüentemente eles serão mais rapidamente acessados, assim foi descrito pela Deitel (2003).

2.2 API Gráfica Piccolo

A ferramenta Piccolo serve para o desenvolvimento de programas com estruturas gráficas em 2D e que possui uma interface Zoomable (ZUIs), isto é, aplicações gráficas com um desempenho ótimo e com uma qualidade excepcional, podendo ser reduzida, ampliada e até rapidamente refeita sem perder qualidade, fazendo dessa forma com que o usuário se sinta confortável ao utilizar o software.

Piccolo atualmente suporta duas de programação Java e C#. Além da ótima qualidade das imagens sem perder em desempenho, Piccolo possibilita ao programador desenvolver em código desenhos geométricos como elipses, círculos e retas, e até não geométricos, através da iteratividade com o usuário possibilitando até o desenvolvimento dos desenhos com o mouse. Esta API facilita em muito a vida dos programadores, através dos seus recursos avançados que possibilitam a manipulação dos desenhos, uma vez em que cada um pode ser tratado como um objeto. E fora que Piccolo tem seu código fonte aberto, ou seja, programadores do mundo inteiro podem melhorar ainda mais seus recursos.

2.3 Inkscape

Inkscape é um aplicativo, onde é possível desenvolver graficamente imagens SVG (Scaleble Vector Graphics), com recursos bastante avançados como marcadores, clones, etc. O inkscape possibilita o desenvolvimento detalhado de imagens SVG, dessa forma fazendo com que o programador faça as imagens personalizadas da forma que ele realmente deseja e com uma qualidade excelente que satisfaz os usuários mais exigentes.

3 DESENVOLVIMENTO

O programa é desenvolvido em Java, utilizando também recursos do XML. Para leitura de arquivos dentro das classes foi utilizado DOM, da mesma forma as de adição, remoção e edição de nodos em um arquivo. Cada componente novo é representado no formato XML.

Os recursos gráficos da biblioteca swing em conjunto com a API Piccolo foi desenvolvida a parte de menus e de manipulação dos componentes adicionados CIs, LED, Resistores e fios, tal

como a classe “*Protoboard*” que irá receber os componentes adicionados pela classe “*Mesa*”(que representa a bancada do laboratório) e também possui o desenho real de uma *protoboard* padrão do laboratório de eletricidade e eletrônica do IFRN - câmpus Currais Novos, essa classe é primordial, por ser detalhada e projetada para ser realmente uma *protoboard* virtual.

A API Piccolo que é uma extensão do Java2D, ou seja, possui os recursos de Java2D e ainda outros mais. Trabalha de forma a deixar mais eficiente a adição e remoção de componentes, pois pelo fato dela tratar cada imagem (tanto as SVG como as desenvolvidas por ele próprio) como objetos, facilitando no desempenho do programa, já não necessitando a cada edição da *protoboard* refazer todos os desenhos (isso aconteceria caso o programa estivesse sendo desenvolvido utilizando apenas Java2D em vez de Piccolo), mas editar ela sem perda de tempo atualizando os componentes sem necessitar atualizar o programa como um todo. Assim o programa ganha em desempenho e qualidade.

No momento estamos trabalhando no desenvolvimento de uma ferramenta de barra de rolagem para se tornar mais padronizado o programa em relação a tantos outros no mercado, onde essa ferramenta é padrão para o deslize sobre a página, atualmente o deslize é possível apenas pressionando o botão direito do mouse e arrastando. Ao mesmo tempo estamos aperfeiçoando a ferramenta de deletar componentes, que atualmente só permite a exclusão de LEDs, CIs e resistores, e que pretendemos adicionar a possibilidade de exclusão de fios. A exclusão é realizada ativando o botão “*remove componente*” e assim basta apenas sair clicando com o botão direito do mouse em cima do componente a ser deletado.

Podemos observar na Figura 7 como o programa está atualmente.

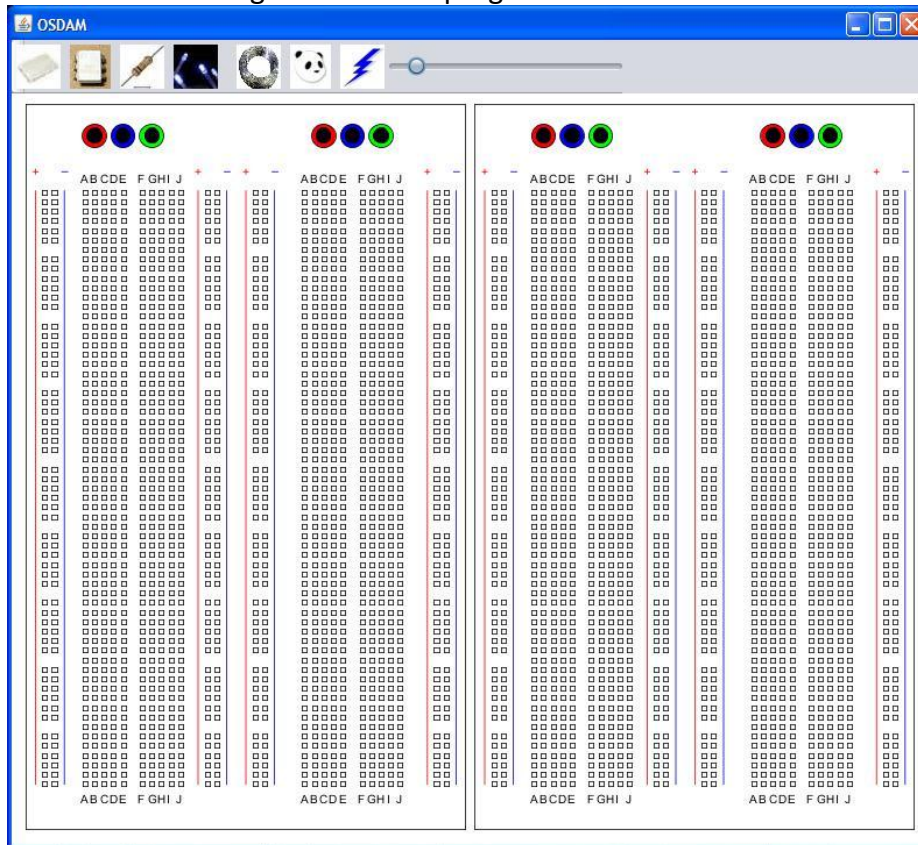


Figura 7: Editor gráfico de circuitos digitais em execução.

Na Figura 7, podemos observar a existência de duas *protoboards* alinhadas horizontalmente, a barra de ferramentas (que podemos observar mais detalhadamente na Figura 8) um pouco acima junto com a ferramenta de zoom.



Figura 8: Barra de ferramentas.

Na Figura 8 temos os botões para adicionar componentes com as imagens respectivas. Além deles, temos os botões para remover (com um raio) e o botão para adicionar resistores. Por ultimo, temos a barra de zoom.

Se quisermos adicionar um componente como um CI, por exemplo, só necessitamos clicar no botão “Adicionar CI” (com um CI na frente) logo após aparecerá uma janela e só basta selecionar o CI desejado feito isso só é necessário clicar em “Adicionar”. Figura 9.

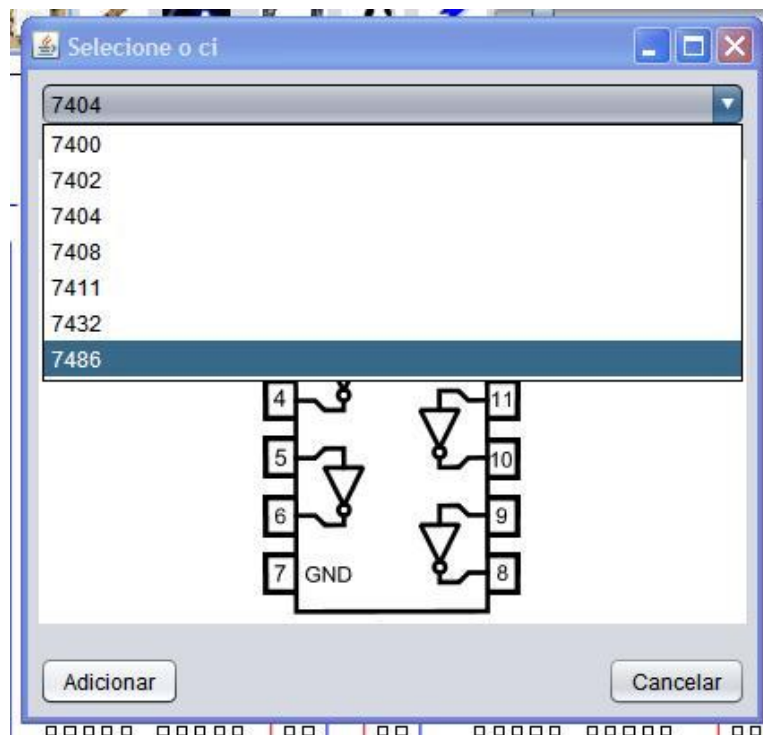


Figura 9: Janela para adicionar CI a *protoboard*.

Da mesma forma da janela de seleção para adicionar CIs (Figura 9) foram feita as demais para adicionar LEDs e Resistores. A parte de visualização de CI na janela foi desenvolvida usando a tecnologia de Piccolo e Swing. Dessa forma também é possível ao usuário manipular a imagem (arrastar ou ampliar).

4 CONCLUSÃO

Concluimos que é possível continuar o desenvolvimento deste simulador, pois o grupo já vem buscando o conhecimento das tecnologias e se habituando ao ambiente. Tais tecnologias são: qual API gráfica utilizar, como representar o circuito através de grafos e o armazenamento baseado em XML e a linguagem Java e edição de imagem no formato SVG.

Acreditamos que será muito útil a conclusão desse sistema tanto para alunos como para profissionais da área, pois com isso poderão armazenar em seus computadores seus projetos principalmente para os alunos que poderão ir guardando para estudos futuros e análises dos circuitos eletrônicos até de uma forma mais segura e preservada.

Será um projeto em longo prazo, pois queremos aperfeiçoar cada vez mais o sistema e torná-lo cada vez mais viável para a utilização acadêmica.

5 REFERÊNCIAS BIBLIOGRÁFICAS

ALMEIDA, M. B. Uma Introdução ao XML, sua Utilização na Internet e Alguns Conceitos Complementares. *Ciência da Informação*, v. 31, n. 2, 2002.

Feofiloff P., Kohayakawa Y., Wakabayashi Y. Uma introdução sucinta à Teoria dos Grafos. Minicurso na II Bienal da SBM Salvador, 2004.

Flagship Institution of the University System of Maryland, College Park, USA. Disponível em: <<http://www.cs.umd.edu/hcil/jazz/>>. Acessado em: 10 maio de 2013.