FERRAMENTA DE AUXÍLIO DIDÁTICO: CONVERSÃO DE EXPRESSÕES DE FUNÇÕES LÓGICAS EM LUA

Bruno de S. Santos – brunodess@gmail.com
Universidade Federal Fluminense – Escola de Engenharia
Departamento de Engenharia de Telecomunicações
Rua Passo da Pátria, 156 / Bloco D / Sala 504
24.210-240 – Niterói – RJ
Alexandre S. de la Vega – alexandre.delavega@gmail.com
Universidade Federal Fluminense – Escola de Engenharia
Departamento de Engenharia de Telecomunicações
Rua Passo da Pátria, 156 / Bloco D / Sala 504
24.210-240 – Niterói – RJ

***Resumo**: O objetivo principal do presente trabalho foi desenvolver um algoritmo para conversão de expressões de funções lógicas, e codificá-lo usando a linguagem de programação Lua. A motivação inicial para o desenvolvimento desse aplicativo foi a sua integração em um conjunto de ferramentas de auxílio didático que o grupo PET-Tele está desenvolvendo para o ensino de Circuitos Digitais. Além disso, uma vez de posse do aplicativo de conversão, pretende-se desenvolver diferentes tipos de interfaces com usuário, a fim de distribuí-lo de diferentes formas.*

***Palavras-chave**: Ferramenta didática, Ensino à distância, Circuitos Digitais, Expressões de Funções Lógicas.*

1 INTRODUÇÃO

O Programa de Educação Tutorial (PET) exige que os bolsistas dos grupos PET, ao serem submetidos a uma formação complementar, desenvolvam atividades que possuam, cada uma delas, itens relativos às áreas de Pesquisa, Ensino e Extensão, bem como consigam algum tipo de penetração no curso ao qual o seu grupo pertence.

Trabalhando na linha de ferramentas de auxílio didático, o grupo PET do Curso de Engenharia de Telecomunicações da Universidade Federal Fluminense (PET-Tele) aproveitou o surgimento da tecnologia de Televisão Digital (TV Digital) e desenvolveu um aplicativo interativo multimídia (MATTOS et al., 2009), codificado na linguagem NCL (SOARES & BARBOSA, 2009) (Ginga - NCL). O aplicativo é um manual que, utilizando recursos de áudio, texto e imagem, descreve o algoritmo de minimização de funções booleanas conhecido por Algoritmo de Quine-McCluskey (HILL & PETERSON, 1981) (RHYNE, 1973) (McCLUSKEY, 1956).

A partir do manual desenvolvido, visualizando a possibilidade de implementar um pacote de aplicativos com as opções de interação com o usuário nas formas local ou via rede de computadores, o grupo implementou o Algoritmo de Quine-McCluskey (BEPPU et al., 2010), utilizando a linguagem de programação Lua (IERUSALIMSCHY, 2006) (The programming language Lua).

Dando prosseguimento aos trabalhos desenvolvidos, o aplicativo aqui apresentado realiza conversões entre as expressões de funções lógicas mais comumente utilizadas. Uma vez que a

expressão mínima também é uma forma de expressão para uma função lógica, o aplicativo de minimização anteriormente desenvolvido foi incorporado ao atual aplicativo de conversão.

Objetivando-se uma ampla e gratuita divulgação do material produzido, todo o material produzido pelo grupo é colocado à disposição para *download* no *website* do grupo, após devida publicação.

A ferramenta desenvolvida é apresentada a seguir. A Seção 2 comenta o pacote de aplicativos que o grupo está desenvolvendo. As formas comuns para a expressão de funções lógicas são discutidas na Seção 3. A Seção 4 justifica a escolha da linguagem. O aplicativo desenvolvido é apresentado na Seção 5. Na Seção 6 encontra-se o resumo de um exemplo de conversões. Finalmente, a Seção 7 apresenta as conclusões e os trabalhos futuros.

2 PACOTE DE APLICATIVOS

Um pacote de aplicativos didáticos, e interfaces com o usuário para tais aplicativos, encontra-se em desenvolvimento pelo grupo PET-Tele.

O passo inicial (MATTOS et al., 2009) foi um manual interativo para o algoritmo de minimização de funções booleanas conhecido por Algoritmo de Quine-McCluskey (HILL & PETERSON, 1981) (RHYNE, 1973) (McCLUSKEY, 1956), implementado usando a linguagem NCL (SOARES & BARBOSA, 2009) (Ginga - NCL), para TV Digital.

Em seguida, a fim de possibilitar uma maior interação com o usuário, o algoritmo de minimização foi implementado (BEPPU et al., 2010). Dado que a linguagem NCL não possui mecanismos procedurais de processamento, foi utilizada a linguagem de programação Lua (IERUSALIMSCHY, 2006) (The programming language Lua). Ela foi escolhida devido à facilidade de interação com a linguagem NCL e pela sua adoção para os dispositivos móveis no Sistema Brasileiro de Televisão Digital (SBTVD) (Ginga - NCL).

Uma vez que a linguagem Lua também possibilita uma fácil integração com outras formas de interface com o usuário (local ou via internet), além de NCL para TV Digital, o grupo decidiu desenvolver outros aplicativos, cada um com três possibilidades de distribuição e interfaces com o usuário: local, rede via TV Digital e rede via Internet.

O trabalho atual é um aplicativo para realizar conversões entre os formatos mais comumente utilizados para expressar funções lógicas.

3 FORMAS COMUNS PARA A EXPRESSÃO DE FUNÇÕES LÓGICAS

No ramo de Telecomunicações, circuitos digitais são largamente utilizados, nas mais diversas aplicações. Os circuitos digitais podem ser divididos em dois grandes grupos: circuitos combinacionais e circuitos sequenciais. Circuitos digitais combinacionais podem ser interpretados como implementações de funções lógicas booleanas. Uma mesma função pode ser descrita por diversas representações: tabela verdade, mapa de Karnaugh, lista de mintermos, lista de maxtermos, lista de valores indeterminados, equação na forma SOP (*Sum-Of-Products*), equação na forma POS (*Product-Of-Sums*) (HILL & PETERSON, 1981) (RHYNE, 1973).

A tabela verdade de uma função lógica booleana é uma simples tabela que lista todas as possíveis combinações de valores para as variáveis das quais a função é dependente, quando elas assumem os valores lógicos “1” ou “0”, bem como o resultado da função para cada uma dessas combinações, que deverá assumir os valores “1”, “0” ou “X” (indeterminado ou *don´t care* ou *can´t happen*). Cada linha da tabela verdade pode ser numerada, interpretando-se cada combinação de valores das variáveis como um número na base numérica binária.

O mapa de Karnaugh é uma reorganização estrutural da tabela verdade em uma forma mais adequada à busca de uma expressão mínima para a função.

Uma combinação que empregue a operação lógica AND de todas as variáveis envolvidas na função, e que sintetize um de seus valores lógicos "1", é denominado mintermo. Uma combinação que empregue a operação lógica OR de todas as variáveis envolvidas na função, e que sintetize um de seus valores lógicos "0", é denominado maxtermo.

Referenciando-se os mintermos e os maxtermos através da numeração das linhas da tabela verdade, uma função pode ser expressa por uma associação da sua lista de valores indeterminados com a sua lista de mintermos ou com a sua lista de maxtermos.

Utilizando-se as equações que definem os mintermos e os maxtermos, as funções podem ser descritas por equações mais complexas. Reunindo-se mintermos com a operação lógica OR, obtém-se uma equação na forma SOP (*Sum-Of-Products*). Juntando-se maxtermos com a operação lógica AND, obtém-se uma equação na forma POS (*Product-Of-Sums*). Uma forma onde cada termo contém todas as variáveis, sem repetição das mesmas, é denominada forma normal expandida ou forma padrão. Partindo-se de uma forma padrão, os termos podem ser combinados sistematicamente, gerando uma forma mínima. Uma mesma função pode apresentar mais de uma forma (SOP ou POS) mínima.

Quanto menor for a expressão lógica associada à função, menor será o circuito digital implementado. Assim, deve ser feito um esforço para minimizar as expressões lógicas envolvidas.

4 ESCOLHA DA LINGUAGEM

A linguagem de programação Lua (IERUSALIMSCHY, 2006) (The programming language Lua) foi naturalmente escolhida por uma série de motivos.

Inicialmente, ela já havia sido utilizada na implementação do algoritmo de minimização realizada pelo grupo (BEPPU et al., 2010), devido à sua compatibilidade com a linguagem NCL e à sua adoção no SBTVD. O objetivo foi incorporar tal implementação no manual sobre o algoritmo, também implementado pelo grupo (MATTOS et al., 2009), a fim de aumentar o grau de interatividade do mesmo.

Em seguida, foi percebido que podem ser implementados diferentes tipos de interfaces com o usuário, que são facilmente compatíveis com Lua. O primeiro deles é a própria linguagem NCL, para aplicações em TV Digital. Para uso local (*stand alone*), pode-se associar o código Lua com uma interface gráfica local (Tecgraf). Um terceiro tipo, para uso via Internet, pode ser obtido ao se utilizar a linguagem de ligação CGILua (CGILua), que permite associar código Lua a formulários e a páginas de Internet (*webpages*).

5 APLICATIVO DESENVOLVIDO

A versão atual do aplicativo implementa as diversas conversões entre as formas de expressões para funções lógicas.

O código está separado em arquivos que contêm funções específicas: funções básicas, funções para manipulação de listas, funções do algoritmo, código principal, dados de entrada, funções para exibição dos dados de saída. A união de todo o código é realizada sem qualquer esforço, através da facilidade de inclusão de código em Lua (função *dofile*).

Basicamente, a estrutura de dados de entrada é uma tabela Lua, contendo informação sobre uma das seguintes representações: tabela verdade, mapa de Karnaugh, número de variáveis + lista de mintermos + lista de valores indeterminados, número de variáveis + lista

de maxtermos + lista de valores indeterminados, equação na forma SOP, equação na forma POS.

Nas listas de mintermos, de maxtermos e de valores indeterminados, são armazenados valores numéricos decimais não negativos, que representam as posições de tais termos na tabela verdade. Para as demais representações, são realizados os seguinte mapeamentos: i) variáveis não complementadas e valores lógicos "1" são representados pelo valor decimal 1, ii) variáveis complementadas e valores lógicos "0" são representados pelo valor decimal 0 e iii) valores indeterminados "X" são representados pelo valor decimal –1.

Após realizar as conversões adequadas, o aplicativo fornece uma tabela Lua contendo informação sobre todas as seguintes representações: tabela verdade, mapa de Karnaugh, lista de mintermos, lista de maxtermos, lista de valores indeterminados, equações nas formas SOP padrão e mínima, equações nas formas POS padrão e mínima.

Na apresentação dos resultados, os valores lógicos são convertidos das representações internas –1, 0 e 1 para os caracteres textuais "X", "0" e "1", respectivamente. No caso das expressões lógicas, os valores internos –1, 0 e 1 são convertidos, respectivamente, em: ausência da variável, variável complementada e variável não complementada.

Caso a função possua valores indeterminados, não é possível escrever uma única forma padrão (SOP ou POS). Nesse caso, as informações referentes às equações padrões assumem a forma de uma mensagem textual indicando tal ocorrência.

Caso haja mais de uma forma mínima para a função desejada, as informações referentes às equações mínimas assumem a forma de uma mensagem textual indicando tal ocorrência.

A saída de dados é realizada durante toda a execução do aplicativo. A cada etapa do algoritmo, as informações geradas são apresentadas, de forma que o usuário possa acompanhar a evolução do processo de conversão.

As operações básicas do aplicativo podem ser brevemente resumidas, como a seguir.

Caso a entrada seja uma forma SOP (ou POS), ela é transformada na sua forma padrão, seguida da geração da forma POS (ou SOP) padrão, da formação das listas de mintermos e de maxtermos, bem como da montagem da tabela verdade.

Se a entrada for uma tabela verdade, são formadas as listas de mintermos, de maxtermos e de valores indeterminados. Caso seja possível, são geradas as formas SOP e POS padrões.

Dado um mapa de Karnaugh, é montada uma tabela verdade, seguida das formação das listas de mintermos, de maxtermos e de valores indeterminados. Caso seja possível, são geradas as formas SOP e POS padrões.

Sendo fornecidos o número de literais, a lista de mintermos (ou maxtermos) e a lista de valores indeterminados, é gerada a lista de maxtermos (ou mintermos) e é montada a tabela verdade.

Uma vez que um mapa de Karnaugh é apenas uma alteração estrutural de uma tabela verdade, ele é gerado durante a apresentação dos resultados.

As formas SOP e POS mínimas são obtidas pela invocação do aplicativo de minimização anteriormente desenvolvido pelo grupo, que utiliza as listas de mintermos, de maxtermos e de valores indeterminados.

6 EXEMPLO DE CONVERSÕES

O número de padrões diferentes a serem testados e a quantidade de dados produzidos pelo aplicativo inviabilizam sua exposição completa no corrente texto. Assim, é apresentado aqui apenas o resumo de um exemplo de conversões, o qual ilustra tanto as expressões de funções lógicas mais comuns quanto a representação utilizada no aplicativo desenvolvido. Nas expressões que se seguem, a negação de uma variável lógica é indicada pelo símbolo "~".

A Figura 1 apresenta a tabela verdade para uma função de três variáveis F(A,B,C). O mapa de Karnaugh correspondente pode ser visto na Figura 2. A função ainda pode ser descrita por seus mintermos:

$$F(A,B,C) = \Sigma\, m\,(0,2,5,7)$$
$$= m_0 + m_2 + m_5 + m_7$$
$$= (\sim A \cdot \sim B \cdot \sim C) + (\sim A \cdot B \cdot \sim C) + (A \cdot \sim B \cdot C) + (A \cdot B \cdot C)$$
$$= (\sim A \cdot \sim C) + (A \cdot C)$$

ou por seus maxtermos:

$$F(A,B,C) = \prod M\,(1,3,4,6)$$
$$= M_1 \cdot M_3 \cdot M_4 \cdot M_6$$
$$= (A+B+\sim C) \cdot (A+\sim B+\sim C) \cdot (\sim A+B+C) \cdot (\sim A+\sim B+C)$$
$$= (A+\sim C) \cdot (\sim A+C)\ .$$

Linha	A	B	C	F(A,B,C)
0	0	0	0	1
1	0	0	1	0
2	0	1	0	1
3	0	1	1	0
4	1	0	0	0
5	1	0	1	1
6	1	1	0	0
7	1	1	1	1

Figura 1 – Exemplo de tabela verdade.

		AB			
		00	01	11	10
C	0	1	1	0	0
	1	0	0	1	1

Figura 2 – Exemplo de mapa de Karnaugh.

As estruturas de dados, de entrada e de saída, usadas pelo aplicativo para armazenar as informações das diversas representações da função lógica são tabelas Lua. Para esse exemplo, as tabelas são:

```
Tabela_Verdade = {1,0,1,0,0,1,0,1,}
Mapa_Karnaugh = { {1,1,0,0,} , {0,0,1,1,} , }

Lista_mintermos = {0,2,5,7,}
Lista_maxtermos = {1,3,4,6,}

SOP_padrão = { {0,0,0,} , {0,1,0,} , {1,0,1,} , {1,1,1,} , }
POS_padrão = { {1,1,0,} , {1,0,0,} , {0,1,1,} , {0,0,1,} , }

SOP_mínima = { {0,-1,0,} , {1,-1,1,} , }
POS_mínima = { {1,-1,0,} , {0,-1,1,} , }
```

7 CONCLUSÃO E TRABALHOS FUTUROS

O grupo PET-Tele desenvolveu anteriormente um manual interativo multimídia para o Algoritmo de Quine-McCluskey, codificado em NCL e empregando diferentes mídias.

A partir do manual desenvolvido, visualizando a possibilidade de implementar um pacote de aplicativos, com as opções de interação com o usuário nas formas local ou via rede de computadores, o grupo implementou o algoritmo básico de minimização de funções booleanas, utilizando a linguagem de programação Lua.

Dando prosseguimento aos trabalhos desenvolvidos, o aplicativo aqui apresentado realiza conversões entre as expressões de funções lógicas mais comumente utilizadas.

Uma vez que a expressão mínima também é uma forma de expressão para uma função lógica, o aplicativo de minimização anteriormente desenvolvido foi incorporado no atual aplicativo de conversão.

Além disso, o grupo PET-Tele já iniciou o desenvolvimento de outras interfaces com o usuário, a fim de oferecer uma versão NCL (para TV Digital), uma versão para uso local e uma versão Internet para o aplicativo aqui apresentado.

Da mesma forma, o grupo já começou a desenvolver outros aplicativos a serem utilizados como ferramentas de auxílio didático para o ensino de Circuitos Digitais.

8 AGRADECIMENTOS

O grupo PET-Tele faz parte do Programa de Educação Tutorial (PET), financiado pelo Ministério da Educação (MEC).

O grupo agradece aos alunos das disciplinas “Técnicas Digitais” e “Circuitos Digitais” que ajudaram a testar o aplicativo desenvolvido.

REFERÊNCIAS BIBLIOGRÁFICAS

BEPPU, M.M.; AMARAL, V.R.L.; DE LA VEGA, A.S. Ferramenta de Auxílio Didático: Algoritmo de Quine-McCluskey em Lua. **Anais: XXXVIII - Congresso Brasileiro de Educação em Engenharia – COBENGE**. Fortaleza, v.1, n.1, p. 1-7, 2010.

COMUNIDADE GINGA. **Ginga-NCL**. Disponível em: <http://www.gingancl.org.br> Acesso em: 01 jul. 2011.

HILL, F. J.; PETERSON, G. R. **Introduction to Switching Theory and Logical Design**. 3. ed. New York: John Wiley, 1981. 636 p, il.

IERUSALIMSCHY, R. **Programming in Lua**.2. ed. Rio de Janeiro: Lightning Source Inc, 2006. 328 p, il.

IERUSALIMSCHY, R. **The Programming Language Lua**. Disponível em: <http://www.lua.org> Acesso em: 01 jul. 2011.

KEPLER PROJECT. **CGILua: Building Web Scripts with Lua**. Disponível em: <http://keplerproject.github.com/cgilua> Acesso em: 01 jul. 2011.

MATTOS, H.; SOUZA, T.; DE LA VEGA, A.S.; SAADE, D.M. Ferramenta Didática Interativa Utilizando a Linguagem NCL: Algoritmo de Quine-McCluskey. **8th International Information and Telecommunication Technologies Symposium**, Florianópolis, v.1, n.1, p. 1-4, 2009.

MCCLUSKEY, E. J. Minimization of Boolean Functions. **Bell System Tech. J.**, New Jersey, v.35, n.5., p. 1417-1444, 1956.

MINISTÉRIO DA EDUCAÇÃO. **Apresentação – PET**. Disponível em: <http://portal.mec.gov.br/index.php?option=com_content&view=article&id=12223&ativo=481&Itemid=480> Acesso em: 01 jul. 2011.

PET-TELE. **PET – Engenharia de Telecomunicações da UFF.** Disponível em: <http://www.telecom.uff.br/pet> Acesso em: 01 jul. 2011.

PUC-RIO. **Tecgraf – Tecnologia em Computação Gráfica**. Disponível em: <http://www.tecgraf.puc-rio.br> Acesso em: 01 jul. 2011.

RHYNE, V. T. **Fundamentals of Digital Systems Design**. New York: Prentice-Hall, 1973. 560 p, il.

SOARES, L. F. G.; BARBOSA S. D. J. **Programando em NCL 3.0: Desenvolvimento de aplicações para o middleware Ginga**.1. ed. Rio de Janeiro: Campus, 2009. 341 p, il.

WIKIPEDIA. **ISDB-T International – Wikipedia, the free encyclopedia**. Disponível em: <http://en.wikipedia.org/wiki/ISDB-T_International> Acesso em: 01 jul. 2011.

LEARNING AID TOOL: CONVERSION OF LOGIC FUNCTION EXPRESSIONS CODED IN LUA

***Abstract**: The aim of this job was to develop an algorithm to convert different logic function expressions one into another as well as to code it by using Lua programming language. The initial motivation to develop such a software tool was to include it into a learning aid tools package that is under construction by the PET-Tele group. The package is intended to be used on Digital Circuits learning. As a next step the group is now working on building some different kinds of user interfaces for this software in order to offer different ways of usage.*

***Key-words**: Learning aid tool, Distance education, Digital Circuits, Logic Function Expression.*