Somador Binário com Decodificador Decimal

Fabíola A. Pessoa[1], Paulo C. Oliveira[1], Zander P. Souza[1] & André L.B. Cavalcante[2]

[1]Aluno, Sistemas de Informação, UPIS Faculdades Integradas
[2]Professor D.Sc, Sistemas de Informação, UPIS Faculdades Integradas

fabiola33096@upis.br, paulo33113@upis.br, zander33132@upis.br, andre02592@upis.br

Resumo: O artigo apresenta a aplicação dos conceitos teóricos aprendidos na disciplina Lógica Matemática e a interligação com os princípios de Arquitetura de Computadores. De fato, o artigo apresenta os passos e conceitos utilizados na construção de um somador com decodificador de base binária para decimal. A calculadora permite que seja realizada a soma de dois números binários de três bits e exibe o resultado na base decimal.

Abstract: The paper presents the application of theorical concepts learned in Logical Math and the connection with the principles of Computer Arquitecture. In fact, the paper presents the steps and concepts used in the construction of an adder with decoder binary-decimal. The calculator allows the sum of two binary numbers with three bits and it displays the total in decimal notation.

1. INTRODUÇÃO

O artigo descreve o procedimento da criação de uma somadora com decodificador de base binária para decimal. Fazendo-se uso das informações adquiridas na disciplina de Lógica Matemática, aplica-se o conhecimento teórico na prática, estendendo-o até os limites de Arquitetura de Computadores. O trabalho possibilita ao aluno entender de forma mais clara a respeito de assuntos, tais como, portas lógicas, álgebra booleana e, finalmente, circuitos lógicos e digitais, objetivo principal.

O grupo de alunos, após algumas reuniões, decide montar um somador que faça operações entre números binários cujo resultado não ultrapasse quatorze e, ao final, converta-o em decimal.

O somador possui um comportamento próximo ao de uma calculadora, porém com um diferencial principal, a calculadora comum faz a operação em decimal, já a calculadora montada pelo grupo de alunos faz a operação em binário. Uma calculadora faz diversas operações como divisão, multiplicação, subtração, soma, já o somador produzido, realiza somente a operação de soma.

O procedimento consiste no recebimento de dois números de base binária, cada número possuindo o valor entre um a sete. Estes dois valores recebidos são somados, dando origem a um resultado. Como o resultado obtido também é um número binário, o somador possui um decodificador que faz essa reversão, da base binária para a decimal. Assim sendo, o somador realiza a soma de dois números binários, gerando um resultado e converte-o para a base decimal, por meio de um decodificador.

2. CONCEITOS TEÓRICOS

Segundo Salmon (1993), a história da Lógica, ciência que trata dos princípios válidos do raciocínio e da demonstração, tem início com o filósofo grego Aristóteles (384–322 a.C.) na Macedônia. Aristóteles criou a ciência da Lógica cuja essência fundamentava-se na teoria do silogismo, isto é, uma regra de inferência para se demonstrar um argumento válido.

De acordo com Aristóteles, um argumento é uma série de afirmações, divididas em premissas, afirmações de evidências, e conclusão, que pode ser extraída das premissas. Aristóteles construiu uma sofisticada teoria dos argumentos, cujo núcleo é a caracterização e análise dos assim chamados silogismos, os típicos raciocínios desse filósofo. O famoso argumento:

(p) Todo homem é mortal.
(q) Sócrates é homem.
(r) Logo, Sócrates é mortal.

é um exemplo típico do silogismo perfeito.

Entre as características mais importantes da silogística aristotélica está a de se ter pensado pela primeira vez na história da lógica em fazer uso de letras (p, q, r, ...) que poderiam ser usadas para representar uma expressão substantiva qualquer, fundamental para desenvolvimentos posteriores. É também com Aristóteles que se encontra uma das primeiras tentativas de se estabelecer um rigor nas demonstrações matemáticas.

Tentativas de se formalizar uma análise matemática da Lógica vieram por meio de trabalhos de Gottfried Wilhelm Leibniz (1646-1716), que só foram reconhecidos no século XIX. Mas foi com Augustus De Morgan (1806-1871), George Boole (1815-1864), Gotlob Frege (1848-1925) e Giuseppe Peano (1858-1932) que se desenvolveu a Análise Matemática da Lógica e a Lógica Formal (Salmon, 1993).

George Boole, matemático inglês, lançou os primeiros passos à álgebra booleana, ramo da matemática com propriedades e regras semelhantes às da álgebra comum, embora com algumas diferenças.

Segundo Daghlian (1986), a álgebra de boole é útil para a lógica matemática e para a teoria dos conjuntos, pois estuda proposições e os valores lógicos associados a ela, ao invés de estudarem variáveis e os respectivos valores numéricos.

A lógica matemática (ou lógica simbólica) trata do estudo das sentenças declarativas também conhecidas como proposições. As proposições são todos conjuntos de palavras ou símbolos que exprimem um pensamento de sentido completo, seja verdadeiro ou falso e, nas quais, devem satisfazer aos dois princípios fundamentais seguintes (Alencar Filho, 1990):

- Princípio do terceiro excluído: uma proposição só pode ser verdadeira ou falsa, não havendo outra alternativa.
- Princípio da não contradição: uma proposição não pode ser ao mesmo tempo verdadeira e falsa.

Diz-se então que uma proposição verdadeira possui valor lógico V (verdade) e uma proposição falsa possui valor lógico F (falso). Os valores lógicos também costumam ser representados por 0 (zero) para proposições falsas (0 ou F) e 1 (um) para proposições verdadeiras (1 ou V).

As proposições podem ser classificadas em simples ou compostas. As proposições simples são aquelas que não contém nenhuma outra proposição como parte integrante de si mesma. Exemplo: Carlos é careca. As compostas são aquelas formadas pela combinação de mais de uma proposição simples. Exemplo: Carlos é careca e Pedro é estudante.

Note, por meio do exemplo anterior, que a combinação é feita por palavras denominadas conectivos. Os conectivos, também chamados de operadores lógicos, ligam duas proposições gerando um único resultado, uma única saída. Os principais operadores lógicos usuais em Lógica Matemática são as palavras “e”, “ou”, “ou-exclusivo” e “não” (Tabela 1).

Tabela 1: Principais operadores lógicos

OPERADORES	**SIMBOLOGIA**
Negação (não)	~p, p’, NOT
Conjunção (e)	^ , **.** , AND
Disjunção (ou)	v , + , OR
Ou exclusivo	$\underline{v}$, ⊕, XOR

As proposições lógicas podem ser combinadas por meio de operadores lógicos como (**.**) e (+), assim, sendo A e B duas proposições simples, pode-se formar proposições compostas, por exemplo: A**.**B + (A⊕B). Conhecendo valores lógicos de duas proposições simples A e B, se determina o valor lógico das proposições compostas por meio do uso da Tabela 2, também conhecida pelo nome sugestivo Tabela-Verdade.

Tabela 2: Tabela verdade

A	**B**	**A’**	**B’**	**A . B**	**A + B**	**A⊕B**
0	0	1	1	0	0	0
0	1	1	0	0	1	1
1	0	0	1	0	1	1
1	1	0	0	1	1	0

A Tabela-Verdade, nada mais é do que todas as possíveis entradas de uma proposição composta e as respectivas saídas, falsas ou verdadeiras. Da Tabela 2, infere-se que (Morris & Kime, 2000):

- A negação, como o próprio nome diz, nega a proposição dada. Além disso, a negação da proposição A é representada por A'. Lembre-se que o símbolo nada mais é que uma simples representação da negação. O que é relevante é que o significado do símbolo seja explicitamente declarado.
- A conjunção entre duas proposições só é verdadeira quando ambas são verdadeiras. O símbolo mais utilizado para a conjunção (^), em eletrônica digital, é o ponto (.).
- A disjunção entre duas proposições é verdadeira quando pelo menos uma entre as duas proposições for verdadeira. O símbolo mais utilizado para a disjunção (v), em eletrônica digital, é o sinal "+".
- A disjunção exclusiva entre duas proposições é verdadeira apenas quando as proposições possuem valores lógicos diferentes. O símbolo mais utilizado para a disjunção exclusiva ($\underline{v}$), em eletrônica digita, é o sinal “⊕”.

Formalmente, a álgebra de Boole é um sistema matemático composto por um conjunto de elementos, chamado normalmente de B, munido de duas operações binárias, que podem ser descritas com os símbolos **.** e + . Estas operações estão definidas no conjunto B e satisfazem os seguintes axiomas (Daghlian, 1986; Morris & Kime, 2000):

- As operações $x + y$ e $x . y$ são fechadas dentro do conjunto B. Ou seja, para qualquer par de elementos *x*, *y* pertencentes ao conjunto B, tem-se que:

$$x + y \in B \quad (1)$$

$$x \,.\, y \in B \quad (2)$$

- As operações **.** e +, são comutativas. Ou seja, para qualquer par de elementos *x*, *y* pertencentes ao conjunto B, tem-se que:

$$x.y = y.x \quad (3)$$

$$x+y = y+x \quad (4)$$

- Cada uma das operações **.** e + é distributiva uma em relação à outra. Isto é, para três elementos quaisquer *x*, *y*, *z* pertencentes ao B, cumpre-se que:

$$x.(y+z) = x.y+x.z \quad (5)$$

$$x+(y.z) = (x+y).(x+z) \quad (6)$$

- No conjunto B existe um elemento neutro bem definido para cada uma das operações . e +. Estes elementos são representados normalmente com os símbolos 0 e 1 $\in$ B, e possuem a seguinte propriedade:

$$0+x = x \quad (7)$$

$$1.x = x \quad (8)$$

- A cada elemento *x* pertencente ao conjunto B corresponde outro elemento chamado complementar de *x*, que normalmente representa-se pelo símbolo *x'*. O elemento *x'* cumpre a propriedade:

$$x.x' = 0 \quad (9)$$

$$x+x' = 1 \quad (10)$$

Uma álgebra de Boole pode ter um conjunto de axiomas diferente do anterior, mas sempre é possível se pode demonstrar que são equivalentes.

A partir desses princípios básicos, Boole sugeriu então que a álgebra booleana poderia ser usada na resolução de problemas que envolvessem a construção de circuitos digitais.

3. DESENVOLVIMENTO PRÁTICO

Definiu-se a elaboração de uma calculadora que faça somente soma. Uma somadora na qual a entrada seja de dois números (digitados pelo usuário), de três bits quaisquer, de base binária (Figura 1).

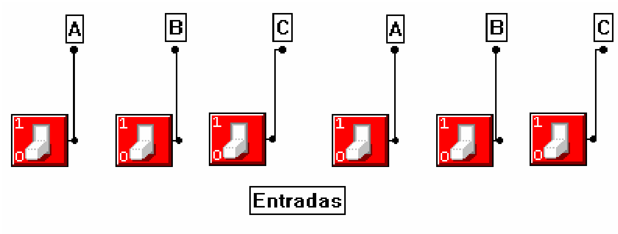

Figura 1: Número de entrada

Esses dois números de entrada passam por um processo de adição, no qual, após serem somados, o resultado da operação em base binária é decodificado, pela própria somadora, sofrendo uma conversão, para a base decimal. O resultado da soma deve ser mostrado por meio de led's representativos, espécies de lâmpadas sinalizadoras, que estão ligados a cada um dos quatorze números decimais, representando todas as saídas, resultados, possíveis da somadora (Figura 2).

Figura 2: Somadora

3.1 Processo de Soma dos Números

Segundo Idoeta & Capuano (1984), para efetuar a adição no sistema binário, deve-se agir como uma adição convencional no sistema decimal, lembrando que, no sistema binário têm-se apenas dois algarismos (Figura 3).

Convém observar que no sistema decimal 1+1 = 2 e no sistema binário representa-se o número 2_{10} por 10_2. Assim sendo, $1+1 = 10_2$. Portanto, tem-se a primeira regra de transporte para a próxima coluna: 1+1 = 0 e transporta 1 (vai um). A Figura 4 apresenta um exemplo de cálculo.

$0 + 0 = 0$
$0 + 1 = 1$
$1 + 0 = 1$
$1 + 1 = 10$
$1 + 1 + 1 = 11$

Figura 3: Regras de adição em Binário

Sabendo que o excedente (vai um) será um dos valores de saída e, ao mesmo tempo, um dos valores de entrada da próxima coluna, este, é chamado de Ts (termo de saída) na saída, e de Te (termo de entrada) na entrada.

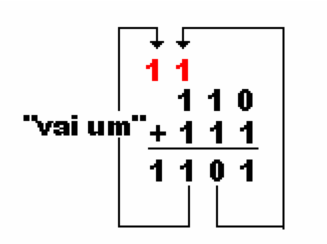

Figura 4: Operação $110_2 + 111_2$

Para somar números com vários bits é necessário somar também o bit de transporte (excedente) vindo do estágio anterior. Portanto, têm-se três bits a serem somados linha a linha: as parcelas A e B e o transporte *Te*, gerado pelo estágio anterior. O somador gera o bit de saída *S* e o bit de transporte *Ts* para o próximo estágio. A Figura 5 apresenta as possíveis soluções para soma de três bits:

$(0 + 0 + 0 = 0 \rightarrow Ts = 0)$
$(0 + 0 + 1 = 1 \rightarrow Ts = 0)$
$(0 + 1 + 0 = 1 \rightarrow Ts = 0)$
$(0 + 1 + 1 = 0 \rightarrow Ts = 1)$
$(1 + 0 + 0 = 1 \rightarrow Ts = 0)$
$(1 + 0 + 1 = 0 \rightarrow Ts = 1)$
$(1 + 1 + 0 = 0 \rightarrow Ts = 1)$
$(1 + 1 + 1 = 1 \rightarrow Ts = 1)$

Figura 5: Representação da soma de três bitas

Deste modo, o próximo passo é a construção da tabela verdade correspondente às constatações da Figura 5. Após a obtenção da mesma, destacam-se as linhas na qual o campo de saída S gera valores iguais a um (Tabela 3). Posteriormente, destacam-se as linhas cujo campo *Ts* gerá valores iguais a um. Ao final, os valores destacados dão origem às expressões *S* e *Ts*, respectivamente, conhecidas por funções normais disjuntivas (FND's).

As FND's obtidas para S (saída) e Ts (termo de saída) estão apresentadas nas Equações 1 e 2, respectivamente:

$$S = A'.B'.Te + A'.B.Te' + A.B'.Te' + A.B.Te \qquad (11)$$

$$Ts = A'.B.Te + A.B'.Te + A.B.Te' + A.B.Te \qquad (12)$$

Tabela 3: Tabela Verdade

A	***B***	***Te***	***S***	***Ts***
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Após a aquisição das FND's a partir da tabela verdade (Tabela 3), é necessária a realização da simplificação da expressão utilizando Álgebra de Boole, para facilitar a montagem da somadora. As expressões simplificadas obtidas para S (saída) e Ts (termo de saída) encontram-se apresentadas nas Equações 3 e 4, respectivamente:

$$S = A \oplus B \oplus Te \qquad (13)$$

$$Ts = B.Te + A.Te + A.B \qquad (14)$$

Após obter os resultados da soma, estes foram colocados na ordem em que foram adquiridos e, posteriormente, é utilizado o decodificador.

3.2 Decodificador

Primeiramente, faz-se a análise do significado das palavras: codificador e decodificador. Para tanto se utiliza o exemplo de uma pessoa de nacionalidade francesa conversando com outra pessoa de nacionalidade brasileira, por meio de um tradutor. O tradutor faz o papel de um codificador e decodificador, ao mesmo tempo.

Exemplificando, se uma pessoa de nacionalidade francesa está conversando com outra de nacionalidade brasileira, por meio de um tradutor, este faz a função de um codificador ao receber as informações em francês, e logo, de decodificador para pessoa que fala português, pois a informação passa de um código desconhecido (o francês) para um código conhecido (o português). Isto é, o tradutor faz o papel de um codificador porque transforma uma linguagem desconhecida para uma outra conhecida.

O somador binário possui um decodificador decimal porque passará o resultado da soma, de uma linguagem de difícil compreensão que é a linguagem binária para a linguagem decimal facilmente compreendida.

Depois de obtidos os resultados das somas, o decodificador tem como entrada esses números, que possuem exatamente quatro bits. Isto porque a soma de dois números de três bits poderá ter como resultado, um outro número de no máximo quatro bits na base binária (Figura 6). Dessa forma, tem-se um decodificador com quatro linhas de entrada e quinze linhas de saída (4 x 15).

$$\begin{array}{r} 111 \\ \underline{111} \\ 1110 \end{array}$$

Figura 6: Soma de números de 3 bits gerando um número de 4 bits

Com o intuito de apresentar o conteúdo de forma didática, a Figura 7, apresenta um decodificador com duas linhas de entrada e quatro linhas de saída (2 x 4).

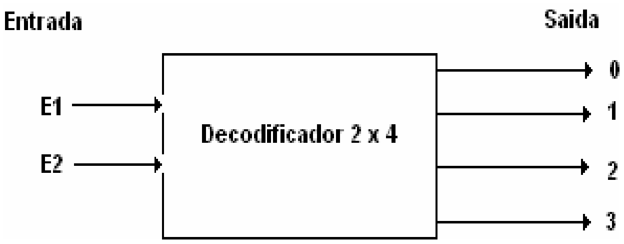

Figura 7: Diagrama em bloco - decodificador (2 x 4)

Monta-se a tabela verdade com as possíveis entradas onde, para cada configuração de bits que aparece na entrada, haverá uma e somente uma linha de saída ativa. Esta linha de saída, representada pelo número um, está assinalando a única saída ativa. Assim quando a linha for acionada, o decodificador verifica os valores dos termos e, conseqüentemente, aciona a linha correspondente ao valor decimal. Por exemplo, quando aparecer na entrada o valor 10_2, é ativada (bit 1) a terceira linha de saída, que equivale ao valor 2 em decimal (Tabela 4).

Tabela 4: Tabela Verdade decodificador (2 x 4)

		Representação em decimal			
A	**B**	**0**	**1**	**2**	**3**
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

A idéia para que se possa conseguir uma expressão onde o valor de saída seja verdadeiro é que todos os números operados sejam verdadeiros. Esta expressão é relativa para cada linha da tabela verdade e, assim, o operador lógico de conjunção é obrigatoriamente utilizado para se obter todos os valores iguais a um. Conhecendo essa necessidade do operador, é realizada a negação de cada valor igual a 0, para que se possa obter o valor condicionado (Tabela 5).

Tabela 5: Negação das entradas

			*RD
0	0	A'.B' = 0'.0' = 1.1 = 1	0
0	1	A'.B = 0'.1 = 1.1 = 1	1
1	0	A.B' = 1.0' = 1.1 = 1	2
1	1	A.B = 1.1 = 1.1 = 1	3
*RD: Representação em decimal			

Finalmente, os resultados da decodificação são ligados a led's, e estes estão representando números decimais.

4. PROTÓTIPO

Para a implementação prática do protótipo, são necessários 21 led's, 10 CI's, sendo eles distribuídos entre portas AND (7408), OR (7432), XOR (7486), sete interruptores, 2 pilhas, fios, fita isolante e um protoboard. Todos esses produtos são adquiridos em lojas de eletrônica. Além disso, são necessários isopor, caixa e cola, materiais encontrados em papelarias.

Para a montagem da somadora, procura-se utilizar todas as portas presentes no CI. Como cada CI possui quatro portas, sempre será utilizada a porta restante combinada com o próximo CI.

O somador é composto por 3 portas AND, 2 portas XOR e 2 portas OR (Equações 13 e 14). A combinação destas ligações possibilita uma economia na quantidade de CI's utilizados. A Figura 8 apresenta a representação das portas lógicas utilizadas:

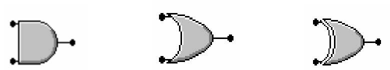

Figura 8: Representação das portas AND, OR, XOR, respectivamente

O somador completo é representado na Figura 9, onde a1 e b1 são as entradas, Te_1 é o termo excedente da operação anterior (vai um), e Saída é o resultado final da operação:

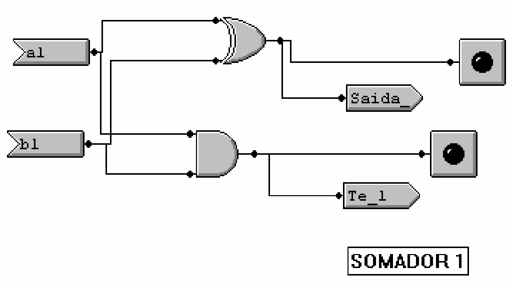

Figura 9: Somador completo

Posteriormente, passa-se para a parte da construção do decodificador, cujas etapas são apresentadas a seguir.

Primeiramente, adicionam-se negações às entradas do decodificador (Figura 10).

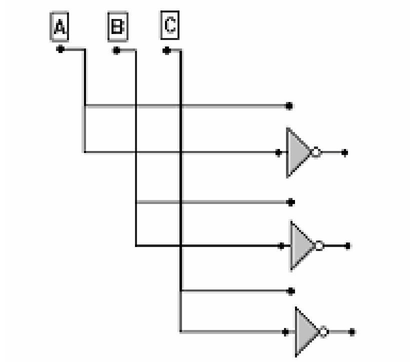

Figura 10: Negação das entradas

Montam-se todas as possíveis expressões lógicas, utilizadas pelo decodificador, a partir da tabela verdade (Tabela 6).

Tabela 6: Tabela verdade do decodificador

A	B	C	D	S0	S1	S2	S3	S4	S5	S6	S7	S8	S8	S10	S11	S12	S13	S14	S15
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
1	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0
1	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
1	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
Ultima linha não é valida para a somadora																			

Após a confecção da tabela verdade, todas a saídas iguais a 1 são, respectivamente, uma saída no decodificador. Por exemplo, ao obter a resposta S3 no somador, a representação na expressão lógica é A’B’CD.

Na Tabela 7 estão representadas todas as expressões que compõem o decodificador.

Tabela 7: Todas as expressões do somador

S0	A’B’C’D’	S8	AB’C’D’
S1	A’B’C’D	S9	AB’C’D
S2	A’B’CD’	S10	AB’CD’
S3	A’B’CD	S11	AB’CD
S4	A’BC’D’	S12	ABC’D’
S5	A’BC’D	S13	ABC’D
S6	A’BCD’	S14	ABCD’
S7	A’BCD		

A seguir apresenta-se a seqüência de etapas para a construção de uma expressão lógica.

A Figura 11 é um CI de porta AND. No qual, 1 e 2, 4 e 5, 10 e 9, 13 e 12, são as entradas do circuito integrado e 3, 6, 11, 8 são, respectivamente, as saídas.

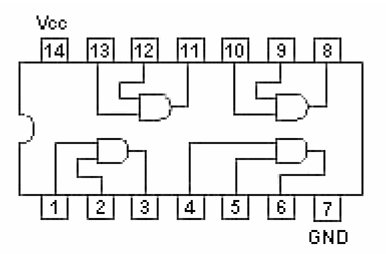

Figura 11: CI de porta AND

Utiliza-se, novamente, a expressão lógica A’B’CD como exemplo (Figura 12):

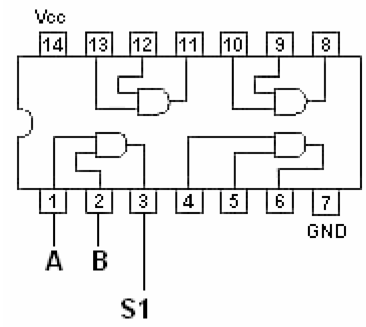

Figura 12: Expressão lógica A’B’

A’ e B’ são as primeiras entradas da expressão. S1 é o resultado A’B’. Assim, S1 = 0’.0’ = 1.1 = 1.

Posteriormente, multiplicam-se as outras entradas C e D obtendo-se o resultado S2 (Figura 13). Assim, S2 = 1.1 = 1.

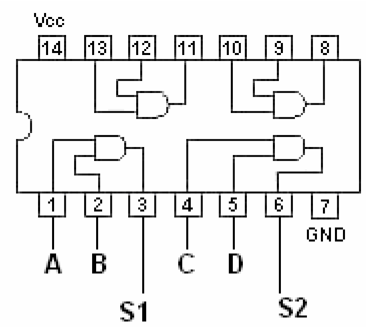

Figura 13: Expressão lógica A’B’ e CD

S1 e S2 agora são as entradas da próxima multiplicação (Figura 14). Assim, R = 1.1 = 1. Para cada expressão lógica montada, são utilizadas três saídas de um CI, como mostrado nas etapas acima, desta maneira se consegue também reduzir a quantidade de CI’s utilizados. Cada saída do decodificador, R, está ligada a um led por meio de um fio. Esse led, por sua vez, é representado por um número de base decimal e, deste modo, o resultado é mostrado ao usuário.

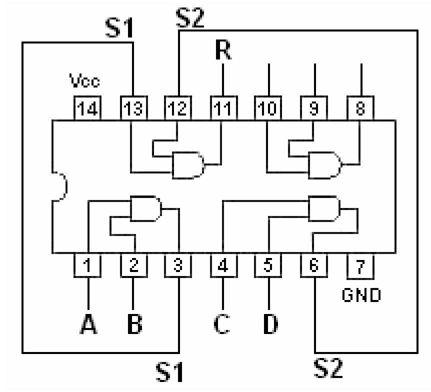

Figura 14: Etapas de uma expressão lógica

A Figura 15 mostra o estado final do decodificador.

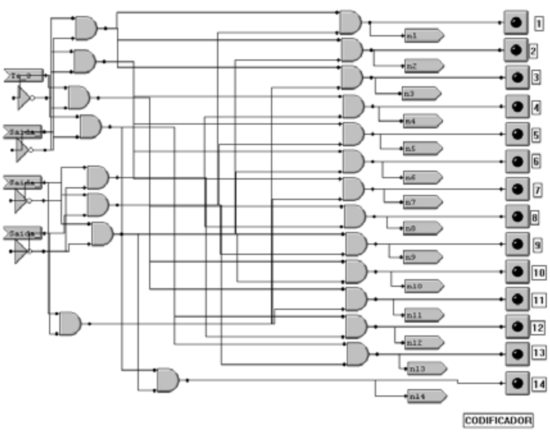

Figura 15: Decodificador da Somadora

O isopor e a caixa servirão como base de proteção e também como base de fixação para o protoboard. Os interruptores e as pilhas utilizados na elaboração da somadora servirão de alimentação para a própria. Os led's utilizados representarão os números decimais, o resultado em binário da soma dos dois números somados, e no canto da somadora, o led utilizado representará se a mesma está ligada ou desligada.

A Figura 16 apresenta a seqüência das etapas construtivas da somadora.

5. CONCLUSÕES E SUGESTÕES

O trabalho apresentado neste artigo é um instrumento que possibilita a oportunidade de mostrar como se faz o processo de soma de dois números binários, cujo resultado não pode passar de quatorze e, assim, converte-o para um número de base decimal por meio de um decodificador sugerido.

Unir conceitos históricos de informática como Álgebra Booleana junto com a eletrônica que permanece tão atual torna esse trabalho gratificante, criando uma perspectiva para o futuro.

Deste modo, ficam para futuro as possibilidades de ao invés de se usar os led's para a representação em decimal, serão utilizados dois painéis digitais para a representação da soma, outra possibilidade será o aumento dos números de bits de entrada, possibilitando uma soma ainda maior.

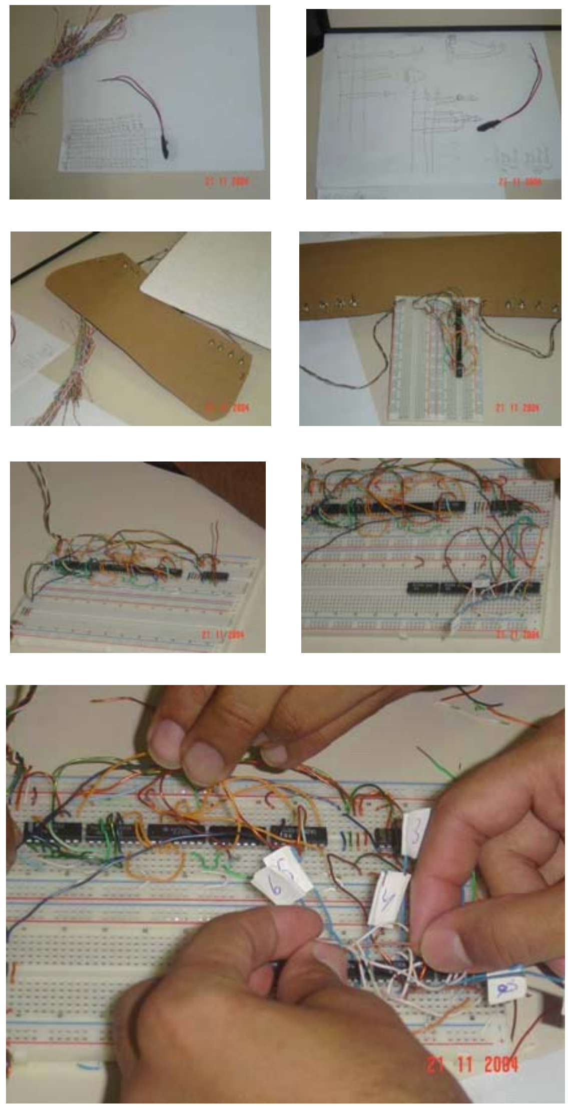

Figura 16: Etapas construtivas

6. REFERÊNCIAS BIBLIOGRÁFICAS

Daghlian, J. Lógica e Álgebra de Boole. São Paulo: Atlas, 1986.

Idoeta, I.V. & Capuano, F.G. Elementos de Eletrônica Digital. São Paulo: Erica, 1984.

Lipschtz S. & Lipson M. Teoria e Problemas de Matemática Discreta. Porto Alegre: Bookman, 2004.

Morris, M.M & Kime, C.R. Logic and Computer Design Fundamentals. Englewood Cliffs:Prentice Hall, 2000.

Salmon C.W. Lógica. São Paulo: Prentice-Hall do Brasil, 1993.

Tanenbaum, S.A. Organização Estruturada de Computadores. São Paulo: Prentice-Hall do Brasil, 1993.

Tocci, R.J. Sistemas Digitais: Princípios e Aplicações. Rio de Janeiro: LTC, 1999.