Experimento 7
Implementação de Circuitos Combinacionais com Multiplexadores

Lucas Mafra Chagas, 12/0126443
Marcelo Giordano Martins Costa de Oliveira, 12/0037301

[1]Dep. Ciência da Computação – Universidade de Brasília (UnB)
CiC 116351 - Circuistos Digitais - Turma C

{giordano.marcelo, chagas.lucas.mafra}@gmail.com

Abstract. *In this experiment, we focus on building combinational circuits using Multiplexers.*

Resumo. *O foco desse experimento é construir Circuitos Combinacionais utilizando Multiplexadores.*

1. Objetivos

O objetivo deste experimento é adquirir conhecimento na implementação e também utilização de multiplexadores, que são circuitos combinacionais.

2. Materiais

- Painel Digital
- *protoboard*
- Fios conectores
- Portas Lógicas NAND e NOT
- 2 x MUX-4
- DECOD-16

3. Introdução

3.1. Multiplexadores e Demultiplexadores

Um multiplexador é um circuito combinacional lógico projetado que tem como saída única e compartilhada um e somente um de seus inputs de dados, a partir da aplicação de um sinal de controle. O demultiplexador, por sua vez, realiza a operação inversa.

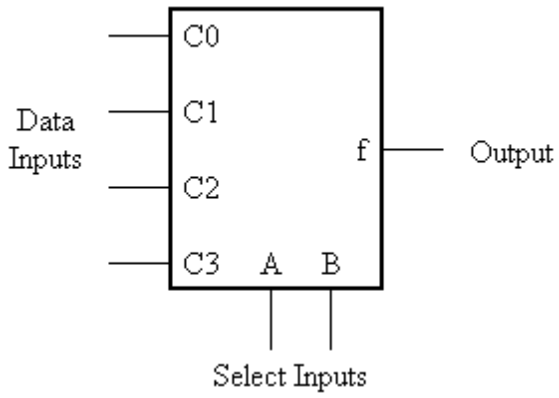

Figure 1. Estrutura de um Multiplexador

Sob o ponto de vista da implementação de seus circuitos, os multiplexadores e demultiplexadores são simplesmente circuitos combinacionais com diversas entradas e somente uma saída, ou vice-versa.

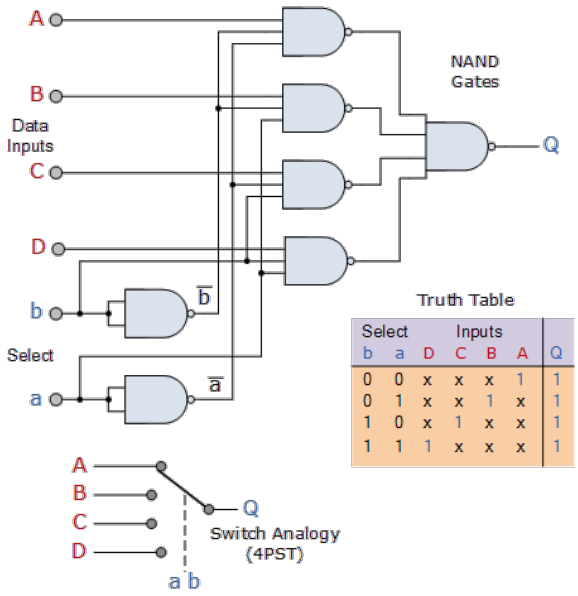

Figure 2. Implementação de um multiplexador com 7 portas NAND's

3.2. Aplicações

A eficiência dos sistemas de comunicação pode ser consideravelmente aumentada ao utilizar-se multiplexadores, por permitir a transmissão de diferentes tipos de dados (como áudio e vídeo) ao mesmo tempo, usando uma única linha de transmissão. Eles também são utilizados para implementar grandes quantidades de memória no computador, reduzindo drasticamente a quantidade de fios de cobre necessários para ligar conectar a memória a outras partes do circuito. Isso se deve ao fato de um multiplexador ser capaz de implementar qualquer função booleana genérica. Quando temos uma função em que obter a tabela verdade não é tão simples, a utilização de multiplexadores e demultiplexadores pode simplificar o problema da implementação dessa função.

4. Procedimentos

4.1. Usar um MUX - 4 duplo em MSI para implementar um somador completo

Para um somador completo, temos que considerar as entradas A,B e C, com A e B sendo os bits na qual se quer somar e C sendo o carry vindo da soma anterior. Como saídas temos que considerar T e S, com S sendo o valor da soma dos 3 bits de entrada e T sendo o carry gerado por essa soma. Temos portanto, que a tabela verdade para esse circuito seria:

Entradas			Saídas	
A 1º bit	B 2º bit	C Vem Um	T Vai Um	S Soma
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Figure 3. Tabela verdade de um Somador Completo

Utilizando a tabela na Figura 5 como refrência e tendo em mente que o circuito deveria ser montado a partir de um MUX - 4 duplo, o seguinte esquema para o circuito a ser implementado foi projetado:

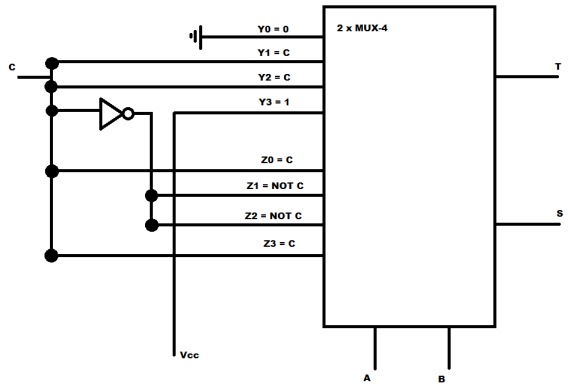

Figure 4. Esquema de um somador completo com 3 entradas e 2 saídas montado com um Multiplexador

O esquema apresentado acima foi montado com base na seguinte ideia: como A e B são as chaves seletoras, precisamos com que as entradas de dados, que terão seus dados selecionados, apresentem os resultados corretos em suas respectivas saídas. Portanto, para a saída T, temos:

Entradas			Saídas	
A 1º bit	B 2º bit	C Vem Um	T Vai Um	S Soma
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Figure 5. Entradas de dados para a saída T

Vemos que quando AB selecionar a entrada Y_0, ou seja, quando AB = 00, que T possui o valor 0 independente da entrada C. Portanto, temos que a entrada de dados Y_0 precisa apresentar esse valor para que quando AB for igual a 00 a saída apresente o valor correto de T. O mesmo ocorre para o caso em que AB = 11, porém, ao invés de T possuir o valor 0, ele possui o valor 1. Assim, conectando Y_0 na terra e Y_1 na fonte, conseguimos estes valores. Já para os casos em que AB = 01 e AB =10, vemos que a saída T tem que ser igual ao valor de C. Portanto, quando AB selecionar a entrada de dados Y_1 ou Y_2, a saída tem que possuir o valor de C. Portanto, conectamos C à essas entradas de dados. Já para o caso da saída S temos a seguinte análise:

Entradas			Saídas	
A 1º bit	B 2º bit	C Vem Um	T Vai Um	S Soma
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Figure 6. Entradas de dados para a saída S

Quando AB = 00 e AB = 11 temos que a saída S é a mesma que o valor da variável S. Portanto, ao selecionar as entradas de dados Z_0 e Z_3 precisamos que elas apresentem o valor de C, explicando a conexão feita. Quando AB = 01 e AB = 10, ocorre o oposto: a saída C possui o valor de NOT C. Portanto, temos que as entradas Z_1 e Z_2 precisam conter esse valor. Feito o esquema apresentado, o circuito foi montado e implementado com a ajuda dos materiais apresentados acima. É possível ver o resultado no seguinte link: Vídeo no Youtube

Erramos na hora de falar a saída de A=0, B=1 e C=0, mas o resultado revisado vendo o vídeo esta correto, no caso T=0 e S=0.

4.2. Implementar uma função de 7 variáveis usando um multiplexador com 8 entradas de dados (MUX-8) construído com 2 multiplexadores usando 4 entradas de dados (MUX-4 duplo)

A função que foi implementada nesta parte do experimento é a seguinte:

$$f(A,B,C,D,E,F,G) = FG + ABCD\overline{EF}G + \overline{ABCDEF}G + A\overline{B}CDEF\overline{G} + \overline{A}BCD\overline{EF}\overline{G} + ABCDE\overline{FG} + A\overline{BC}DEF\overline{G}$$

Para implementar essa função com um decodificador e um multiplexador de 4 entradas de dados duplo, foi preciso usar o decodificador para gerar minitermos que serviriam como entrada de dados para o multiplexador. Como o decodificador possuía 4 entradas, as quatro variáveis mais significativas da função foram escolhidas para gerar os minitermos. Observando a função principal, temos que os minitermos gerados pelas quatro primeiras variáveis são:

$$m_0 = \overline{ABCD}$$

$$m_7 = \overline{A}BCD$$

$$m_9 = A\overline{BC}D$$

$$m_{11} = A\overline{B}CD$$

$$m_{15} = ABCD$$

Portanto, temos que no esquema, o decodificador será da seguinte forma:

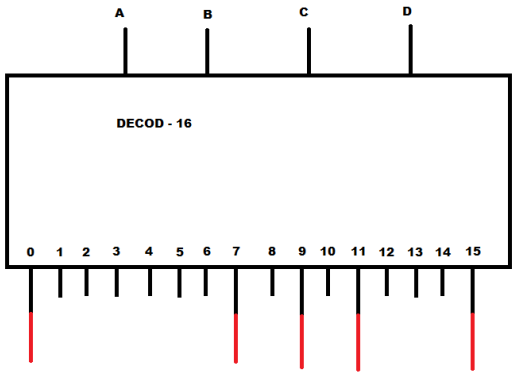

Figure 7. Esquema do decodificador a ser implementado. Os fios em vermelhos são aqueles que apresentam os minitermos formados pelas variáveis mais significativas que são válidos na função

Antes de definir o que fazer com as saídas do decodificador é necessário primeiramente analisar as variáveis restantes. Fazendo esta análise na forma de tabela verdade, temos:

E	F	G	S	Entrada de dados ativada
0	0	0	0	Y_0
0	0	1	1	Y_1
0	1	0	1	Y_2
0	1	1	1	Y_3
1	0	0	1	Z_0
1	0	1	0	Z_1
1	1	0	1	Z_2
1	1	1	1	Z_3

Table 1. Tabela verdade para as 3 variáveis menos significativas da função f

Cada elemento da saída nessa tabela de dados corresponderá à uma entrada de dados no MUX - 4 duplo. Porém, temos que o MUX - 4 duplo tem apenas 2 entradas de seleção, e aqui temos 3 variáveis. Observando bem, podemos chegar à uma conclusão: quando E = 0, nenhuma das entradas de dados Z estará ativada. O oposto ocorre quando E = 1. Portanto, para este circuito, a variável E será incrementada na entrada de ativação de cada MUX - 4 dentro do MUX - 8. Essa ativação estabelece que quando E = 0, o MUX funciona normalmente. Quando a entrada está em 1, não importa os valores nos seletores e nem nas entradas de dados, a saída será 0. Essa configuração nos permite inserir as 3 variáveis e obter o resultado desejado. Temos, portanto, que o esquema do multiplexador duplo para esse circuito será da seguinte forma:

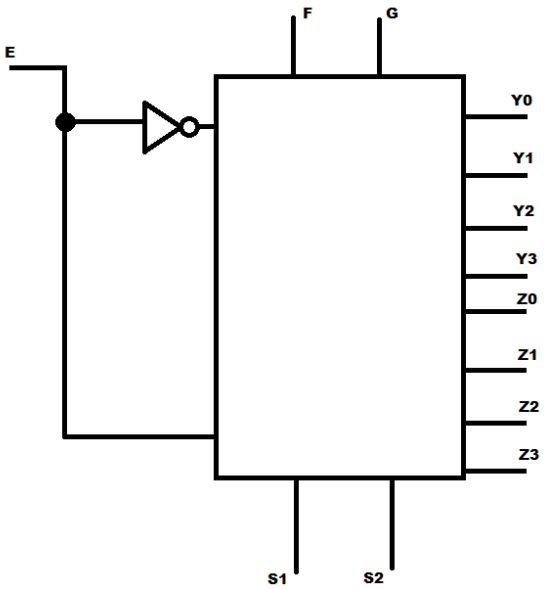

Figure 8. Esquema do multiplexador a ser implementado

Agora, juntando os dois esquemas é possível implementar definir as entradas de dados e implementar as funções. Para EFG = 000 temos que a saída será 0 independente do valor das variáveis mais significativas Quando EFG = 001, temos, olhando a função que as variáveis mais significativas precisam ser correspondentes à e . Portanto, para que a saída seja correta, precisamos que a entrada de dados Y_1 precisa ter como entrada OU .

Quando EFG = 010, as quatro variáveis mais significativas formam . Portanto, a entrada de dados de Y_2 é justamente a saída 7 do decodificador. Vemos que quando FG = 11, que a saída será 1 independente dos outros termos. Portanto, as entradas de dados Y_3 e Z_3 podem estar sempre conectadas à fonte. Quando EFG = 100 temos que ABCD são os minitermos 9 e 15. Isso implica que Z_0 tem que estar conectado à união dos dois. Quando EFG = 101 temos o mesmo caso que o de EFG = 000. Finalmente, quando EFG = 110 temos que ABCD corresponde ao minitermo 11. Portanto, Z_2 precisa ter como entrada a saída desse minitermo no decodificador. A partir dessas informações foi possível montar o esquema do circuito completo.

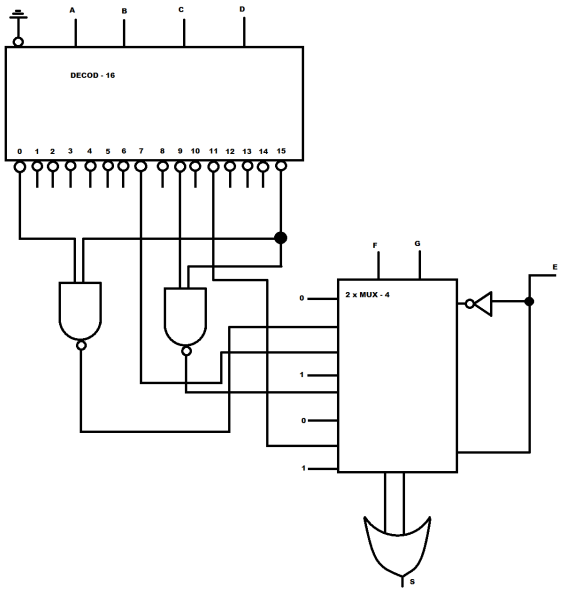

Figure 9. Esquema da função f utilizando um DECOD -16 e um MUX - 4 duplo

Obs.: Como a saída do decodificador era invertida, a porta NAND foi utilizada para substituir a porta OR. Além disso, como o multiplexador possui 2 saídas, temos que a saída será válida se uma delas for verdadeira. Por isso a porta OR foi utilizada ao final: para transformar as duas saídas em uma.

5. Análise dos Resultados

Não foi possível realizar a implementação do segundo circuito, graças a queda do moodle no dia do experimento, mas a análise da segunda parte foi feita e a primeira parte foi um sucesso, portanto o experimento foi executado de acordo como pedido no roteiro. Então, o experimento foi um sucesso.

6. Conclusão

A realização do experimento resultou num excelente aprendizado a respeito dos multiplexadores e demultiplexadores. Na experiência, os multiplexadores são tomados como circuitos combinacionais e a relação deles com os demultiplexadores também foi estudada. Foi possível concluir que um multiplexador é uma ótima ferramenta para gerar

funções booleanas. A simplificação de funções booleanas, como sempre, ajuda a simplificar o experimento, realçando novamente sua importância. A complexidade do experimento e o curto espaço de tempo que se tem para realiza-lo não permite atrasos de nenhum tipo e a simplificação das funções é de extrema ajuda nesse sentido.

Auto-Avaliação

1. C
2. C
3. E
4. C
5. E
6. C
7. C
8. C
9. E
10. E
11. C