CIRCUITOS CODIFICADORES

Júlia Pessoa Souza

Teoria de Eletrônica Digital-Turma C
Faculdade Gama - Universidade de Brasília
15/0133294
Laboratório 5

João Pedro Vergara Coneglian

Teoria de Eletrônica Digital-Turma C
Faculdade Gama - Universidade de Brasília
14/0145974
Laboratório 5

RESUMO

No primeiro projeto, foi feita a implementação na base Basys 3 de um circuito que transformava código binário em hexadecimal até o número 9. No segundo projeto, foi feita a implementação de um circuito que transformava binário em hexadecimal de 0 a F. Nos dois casos, foi mostrado no display de 7 segmentos o número para o qual o binário era convertido.

1. INTRODUÇÃO

Circuito codificador é um circuito lógico que executa passagem de um código conhecido para um desconhecido. Possui um número certo de linhas de entrada, mas é ativada apenas uma por vez, produzindo uma saída de n bits dependendo de qual entrada está ativada, enquanto um decodificador faz o contrário.
O display de 7 segmentos possibilita escrever os números hexadecimais de 0 a F. Cada segmento é composto por um led e representado por uma letra de *a* a *g*. Circuitos codificadores são utilizados para manusear o display.

2. EXPERIMENTO

2.1. Projeto 1

2.1.1. Vivado

Para o primeiro projeto foi necessário construir uma tabela verdade com um número binário de 4 bits como entrada e 7 bits de saída, um para cada led do display. A tabela foi feita com base no led que se acenderia a partir do número de entrada. Foi feito um mapa de Karnaugh e uma expressão booleana para cada segmento e, em seguida, as expressões foram aplicadas no código. Foi utilizado o programa Vivado para escrever o algoritmo em VHDL. Criou-se um novo projeto com a opção RTL project selecionada. Foi escolhida a board Basys 3 para fazer a implementação posteriormente, então criou-se o arquivo fonte em VHDL. A entrada foi um vetor A com 4 bits e a saída foi um vetor S com 7 bits. No código, para cada bit de S foi adicionada a expressão booleana encontrada anteriormente a partir da tabela verdade. Para verificar a funcionalidade do código, foi feita a simulação.

2.2. Projeto 2

2.2.1. Vivado

Para o projeto 2, a mesma tabela verdade foi utilizada. Foi criado um novo RTL project no Vivado com a board Basys 3 selecionada. Porém, para esse projeto, utilizou-se uma abordagem diferente no código. Para cada combinação de entrada A, foi fixada qual seria a combinação de saída S. Utilizando o comando WITH...SELECT foram feitas todas as saídas para que o display apresentasse os números corretos. Posteriormente foi feita a simulação.

2.3. Implementação

Para a implementação, nos dois casos foi adicionado o arquivo Basys 3 Master com o código para a placa. Na opção de switch, foram colocadas as entradas A0, A1, A2 e A3 para as chaves V17, V16, W16 e W17, e as saídas foram colocadas para o display. Foi rodada a síntese e a implementação e gerado o bitstream. E então, depois de conectada a placa, já estava pronta para os testes.

2.4. Figuras e Tabelas

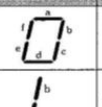
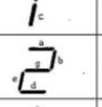
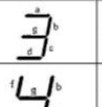
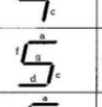
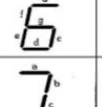
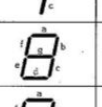
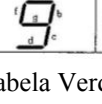
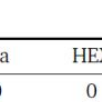
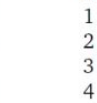
Characteres	Display	BCD 8421	Código para 7 Segmentos
		A B C D	a b c d e f g
0		0 0 0 0	1 1 1 1 1 1 0
1		0 0 0 1	0 1 1 0 0 0 0
2		0 0 1 0	1 1 0 1 1 0 1
3		0 0 1 1	1 1 1 1 0 0 1
4		0 1 0 0	0 1 1 0 0 1 1
5		0 1 0 1	1 0 1 1 0 1 1
6		0 1 1 0	1 0 1 1 1 1 1
7		0 1 1 1	1 1 1 0 0 0 0
8		1 0 0 0	1 1 1 1 1 1 1
9		1 0 0 1	1 1 1 1 0 1 1

Figura 1: Tabela Verdade de 0 a 9.

Entrada	HEX	ABCDEFG
0000	0	1111110
0001	1	0110000
0010	2	1101101
0011	3	1111001
0100	4	0110011
0101	5	1011011
0110	6	1011111
0111	7	1110000
1000	8	1111111
1001	9	1111011
1010	A	1110111
1011	B	0011111
1100	C	1001110
1101	D	0111101
1110	E	1001111
1111	F	1000111

Figura 2: Tabela Verdade de 0 a F.

```
entity LAB52 is
    Port ( A : in STD_LOGIC_VECTOR (3 DOWNTO 0);
           S : out STD_LOGIC_VECTOR (6 DOWNTO 0));
end LAB52;
```

Figura 3: Declaração dos vetores de entrada e de saída.

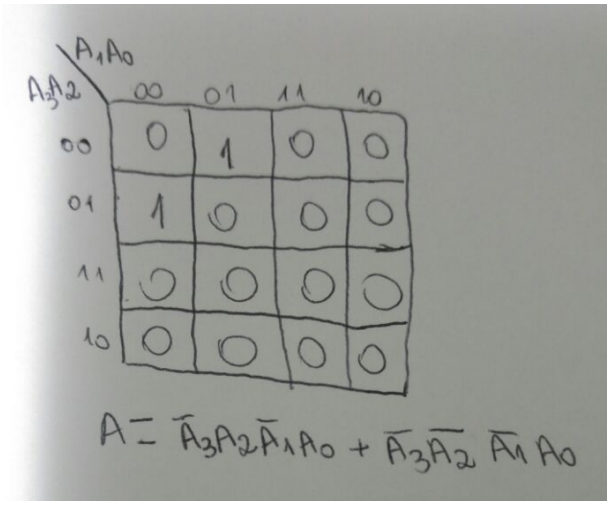


Figura 4: Mapa de Karnaugh de a.

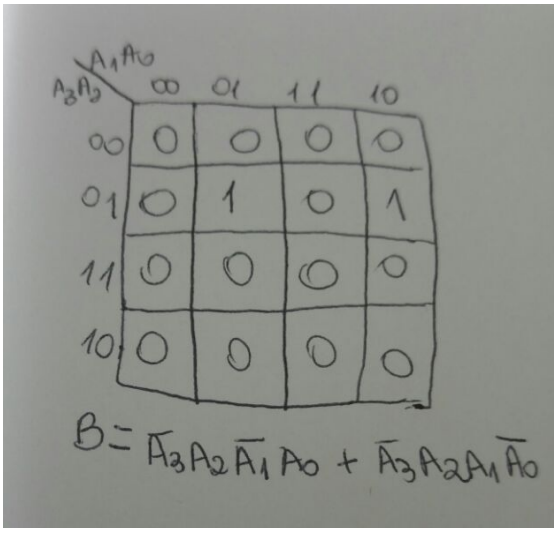


Figura 5: Mapa de Karnaugh de B.

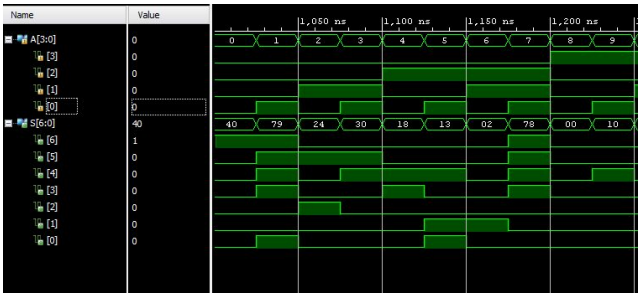


Figura 6: Simulação do Projeto 1.

```
begin
WITH A SELECT
S <= "1000000" WHEN "0000",
     "1111001" WHEN "0001",
     "0100100" WHEN "0010",
     "0110000" WHEN "0011",
     "0011001" WHEN "0100",
     "0010010" WHEN "0101",
     "0000010" WHEN "0110",
     "1111000" WHEN "0111",
     "0000000" WHEN "1000",
     "0010000" WHEN "1001",
     "0001000" WHEN "1010",
     "0000011" WHEN "1011",
     "1000110" WHEN "1100",
     "0100001" WHEN "1101",
     "0000110" WHEN "1110",
     "0001110" WHEN "1111",
     "1111111" WHEN OTHERS;

end Behavioral;
```

Figura 7: Arquitetura do Projeto 2.

Figura 8: Simulação do Projeto 2.

```
##7 segment display
set_property PACKAGE_PIN W7 [get_ports {S[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {S[0]}]
set_property PACKAGE_PIN W6 [get_ports {S[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {S[1]}]
set_property PACKAGE_PIN U8 [get_ports {S[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {S[2]}]
set_property PACKAGE_PIN V8 [get_ports {S[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {S[3]}]
set_property PACKAGE_PIN U5 [get_ports {S[4]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {S[4]}]
set_property PACKAGE_PIN V5 [get_ports {S[5]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {S[5]}]
set_property PACKAGE_PIN U7 [get_ports {S[6]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {S[6]}]

```

Figura 9: Código Basys 3 Master Display.

```
## Switches
set_property PACKAGE_PIN V17 [get_ports {A[0]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {A[0]}]
set_property PACKAGE_PIN V16 [get_ports {A[1]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {A[1]}]
set_property PACKAGE_PIN W16 [get_ports {A[2]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {A[2]}]
set_property PACKAGE_PIN W17 [get_ports {A[3]}]
    set_property IOSTANDARD LVCMOS33 [get_ports {A[3]}]
```

Figura 10: Código Basys 3 Master Switches.

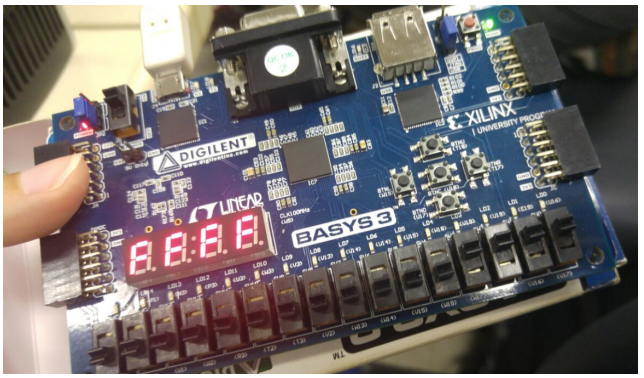


Figura 11: Projeto 2 implementado.

3. RESULTADOS

As tabelas da figura 1 e 2 foram necessárias para montar as expressões no código dos projetos. Com essas tabelas, os mapas de Karnaugh foram montados, assim como mostram as figuras 4 e 5. Os códigos de ambos os projetos foram montados utilizando as mesmas entradas e saídas, mostradas na figura 3. As simulações foram feitas para que se garantisse a implementação correta do projeto. Como pode ser visto nas figuras 6 e 8, quando todas as entradas são zero, apenas S6 é 1, o que representa que o G é o único led desativado, formando o número zero no display. Os dois projetos foram implementados da mesma maneira, como mostram as figuras 9 e 10. A figura 11 mostra que quando a entrada é 1111, o display mostrará a letra F.

4. DISCUSSÃO E CONCLUSÕES

A implementação dos códigos foi um sucesso, com a implementação ocorrendo de forma esperada de acordo com as tabelas verdade. Percebe-se que as tabelas verdade, a simulação e a placa estão de acordo, o que demonstra que o experimento cumpriu seu objetivo.

5. REFERÊNCIAS

[1] I. Idoeta e I. Valeije, “Elementos de Eletrônica Digital”, Érica Ltda, São Paulo, 2008.

[2] R. Tocci, N.Widmer e G. Moss, “Sistemas Digitais: Princípios e aplicações” Pearson.

[3] G. Bezerra, “Apostila Prática de Eletrônica Digital I”, 2017

[4] G. Bezerra, “Tutorial de Simulação e Implementação no Vivado”, 2017