Universidade de Brasília – UNB

Curso: Engenharia de Redes de Comunicação
Disciplina: Laboratório de Sistemas Digitais
Turma: 08

Relatório da Disciplina Laboratório de Sistemas Digitais
Tema: Experimento 03 – Introdução à Linguagem VHDL

Aluno: Pedro Henrique Dias Avelar
Matrícula: 241037112
Professor: Eduardo Paiva

Lista de Referências

Introdução

O presente experimento tem os seguintes objetivos:

- Implementar circuitos combinacionais simples utilizando atribuições condicionais seletivas da linguagem VHDL
- Desenvolver módulos básicos (decodificador e multiplexador) que podem ser utilizados futuramente na implantação de circuitos de maior complexidade
- Realizar a simulação dos circuitos arquitetados no ModelSim

Experimento 01 – Multiplexador 8 para 1

O multiplexador 8 para 1 pode ser arquitetado utilizando dois vetores de entrada (S com 3 bits e D com 8 bits) e um bit de saída (Y), com a seguinte tabela-verdade:

S	Y
000	D_0
001	D_1
010	D_2
011	D_3
100	D_4
101	D_5
110	D_6
111	D_7

Tabela 1-Tabela-Verdade do Multiplexador 8x1

O multiplexador pode ser implantado facilmente no ModelSim com o uso de atribuições condicionais **WHEN-ELSE**.

```
-- Experimento 03 - Questão 01
-- Aluno: Pedro Henrique Dias Avelar 241037112
-- Turma 08
-- Data: 13/11/2024

-- Multiplexador 8x1
-- Entrada: Vetor S (3 bits) e Vetor D (8 bits)
-- Saída: Y (1 bit)

-- Usar atribuições condicionais WHEN-ELSE

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY CIRCUITO_E03Q01 IS
    PORT (S: IN STD_LOGIC_VECTOR (2 DOWNTO 0);        -- SELETOR 3 BITS
          D: IN STD_LOGIC_VECTOR (7 DOWNTO 0);        -- ENTRADA 8 BITS
          Y: OUT STD_LOGIC);                          -- SAÍDA
END CIRCUITO_E03Q01;

ARCHITECTURE ARC_CIRCUITO_E03Q01 OF CIRCUITO_E03Q01 IS
BEGIN
    Y <= '1' WHEN (S = "000" AND D(0) = '1') ELSE     -- S=000 -> Y=D0
         '1' WHEN (S = "001" AND D(1) = '1') ELSE     -- S=001 -> Y=D1
         '1' WHEN (S = "010" AND D(2) = '1') ELSE     -- S=010 -> Y=D2
         '1' WHEN (S = "011" AND D(3) = '1') ELSE     -- S=011 -> Y=D3
         '1' WHEN (S = "100" AND D(4) = '1') ELSE     -- S=100 -> Y=D4
         '1' WHEN (S = "101" AND D(5) = '1') ELSE     -- S=101 -> Y=D5
         '1' WHEN (S = "110" AND D(6) = '1') ELSE     -- S=110 -> Y=D6
         '1' WHEN (S = "111" AND D(7) = '1') ELSE     -- S=111 -> Y=D7
         '0';                                         -- Demais casos - Y=0

END ARC_CIRCUITO_E03Q01;
```

Código 1:Multiplexador 8x1

Com as atribuições **WHEN-ELSE**, é possível atribuir a saída Y o valor do vetor D de acordo com o valor do vetor S. Para cada um dos possíveis valores do vetor S, Y será igual a um determinado bit do vetor D; para as demais situações, Y será zero.

Este foi o código executado durante a aula. Embora funcional, há uma maneira ainda mais simples de implantar o multiplexador. Poderíamos atribuir diretamente o valor do vetor D a Y da seguinte maneira:

```
Y <=      D(0) WHEN (S = "000") ELSE     -- S=000 -> Y=D0
          D(1) WHEN (S = "001") ELSE     -- S=001 -> Y=D1
          D(2) WHEN (S = "010") ELSE     -- S=010 -> Y=D2
          D(3) WHEN (S = "011") ELSE     -- S=011 -> Y=D3
          D(4) WHEN (S = "100") ELSE     -- S=100 -> Y=D4
          D(5) WHEN (S = "101") ELSE     -- S=101 -> Y=D5
          D(6) WHEN (S = "110") ELSE     -- S=110 -> Y=D6
          D(7) WHEN (S = "111") ELSE     -- S=111 -> Y=D7
          '0';                           -- Demais casos - Y=0
```

Código 2:Outra maneira de implantar o Multiplexador 8x1

Embora os dois códigos sejam funcionalmente iguais, a segunda forma é muito mais simples e elegante, e será a maneira adotada futuramente em caso de reutilização de código para o multiplexador.

Para a simulação no ModelSim, foi utilizado um tempo total de simulação de 800ps. O vetor S foi configurado na opção Clock com os seguintes períodos:

S(2):100ps
S(1):50ps
S(0):25ps

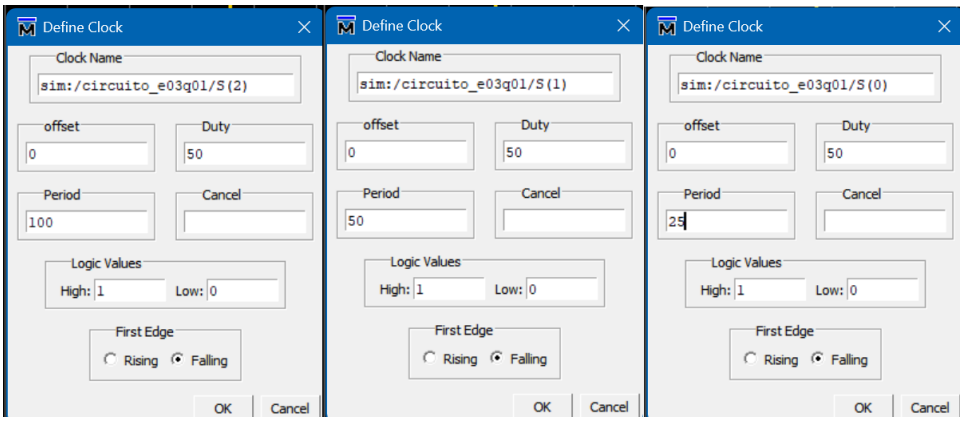

Figura 1:Ajuste de Clock para o vetor S

Já o vetor D foi configurado na opção Force, com seu valor alterando bit a bit a cada 100 ps.

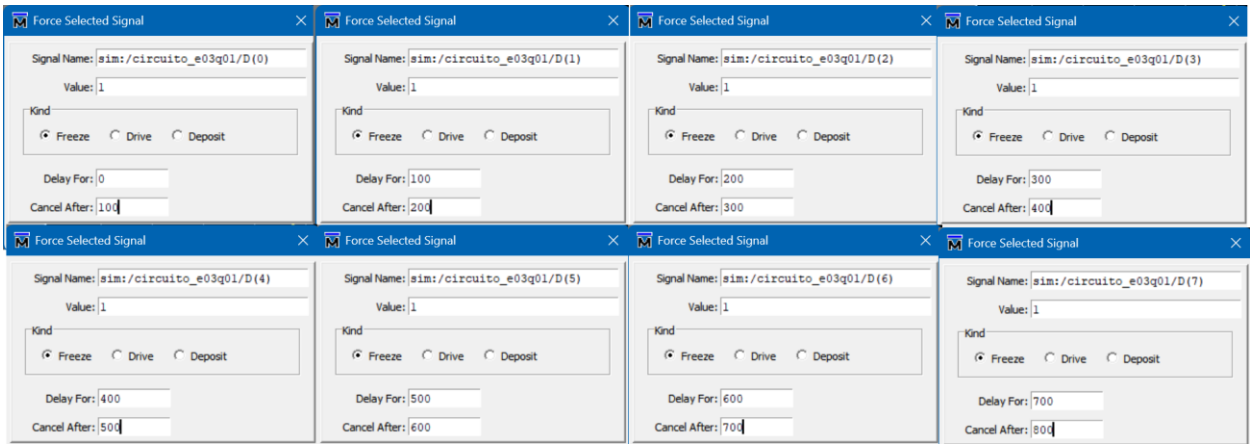

Figura 2:Ajuste de Force para o vetor D

E assim foi obtido o seguinte gráfico para a simulação:

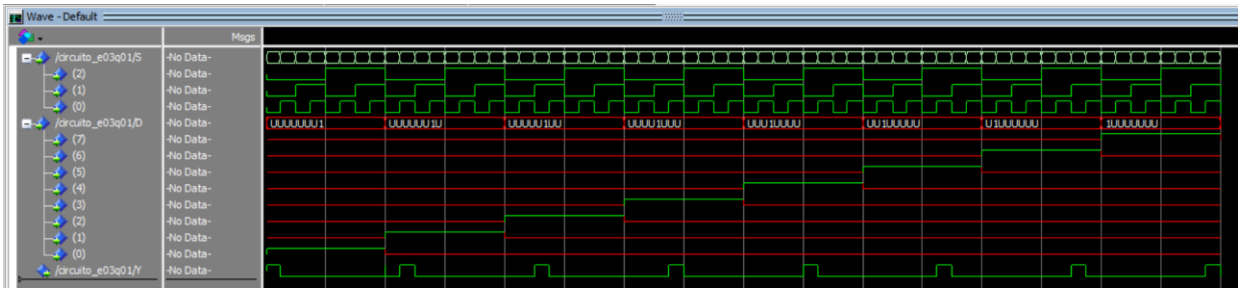

Figura 3:Resultado da Simulação para o Multiplexador 8x1

E com isso podemos a partir do gráfico obter os valores de Y para os 8 valores possíveis de S, e compara com o valor de D para cada um de seus 8 bits ativados; pela tabela verdade não se faz necessário testar todos os possíveis valores de D:

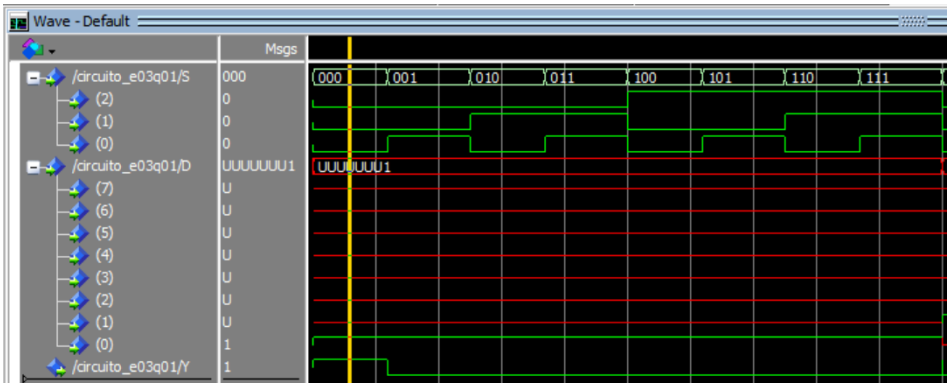

Figura 4:Y=D0 para S=000

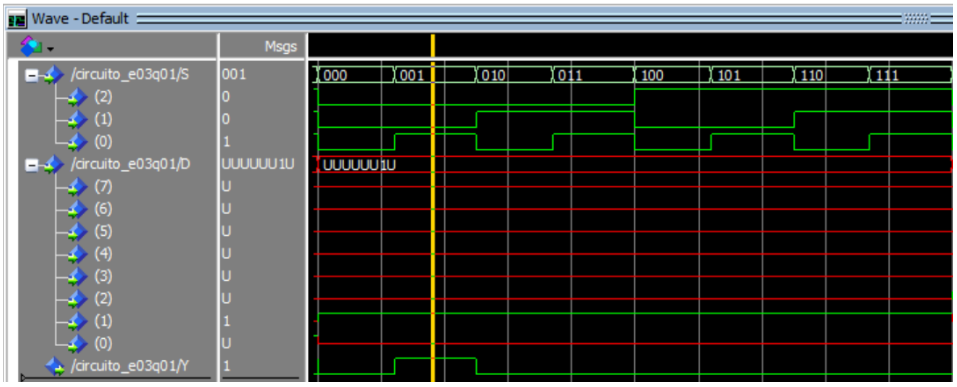

Figura 5:Y=D1 para S=001

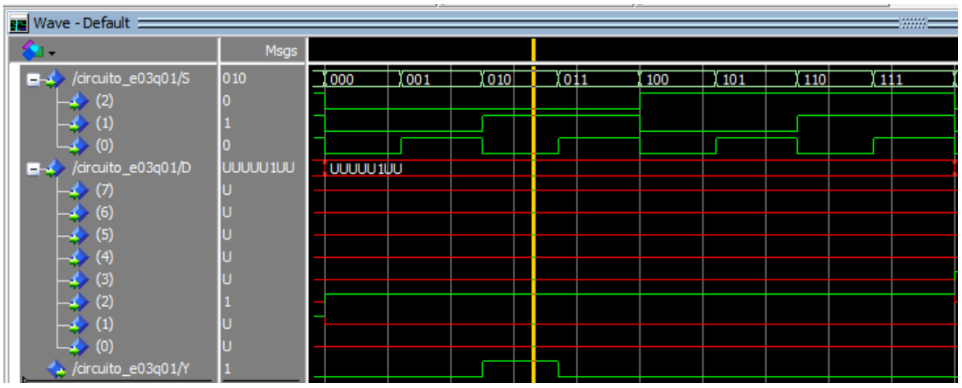

Figura 6:Y=D2 para S=010

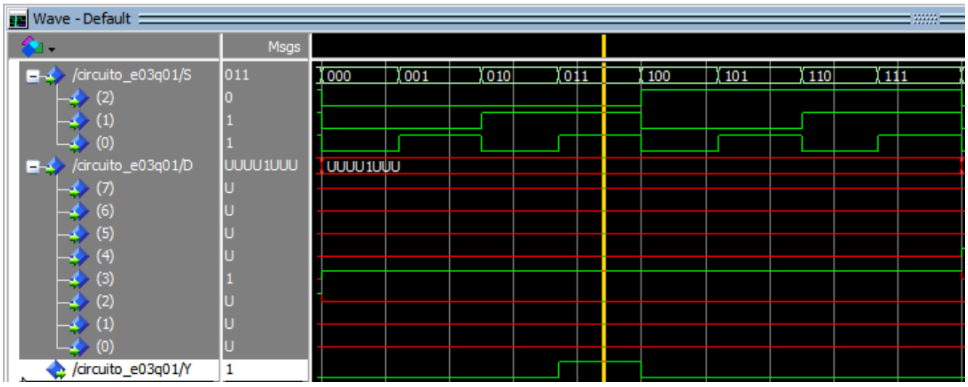

Figura 7:Y=D3 para S=011

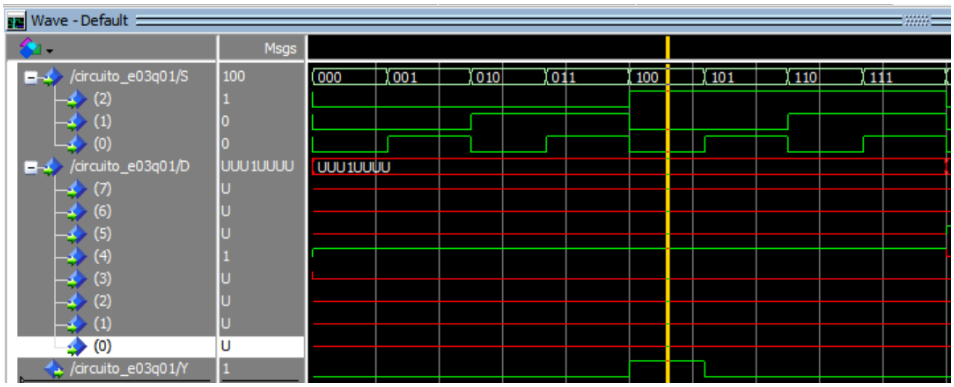

Figura 8:Y=D4 para S=100

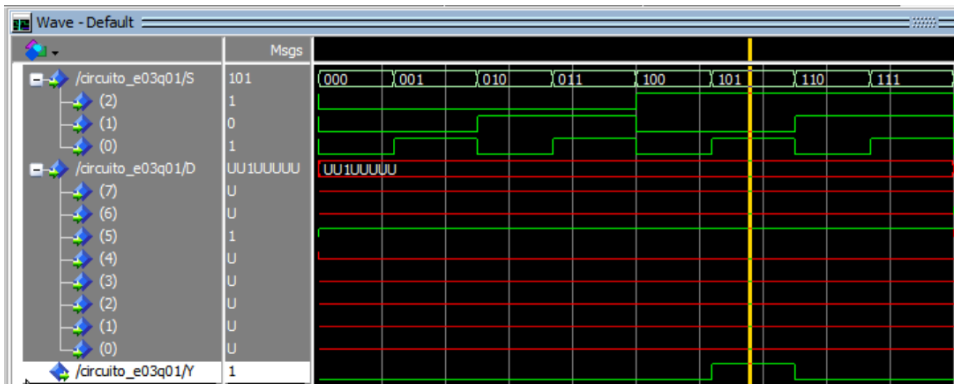

Figura 9:Y=D5 para S=101

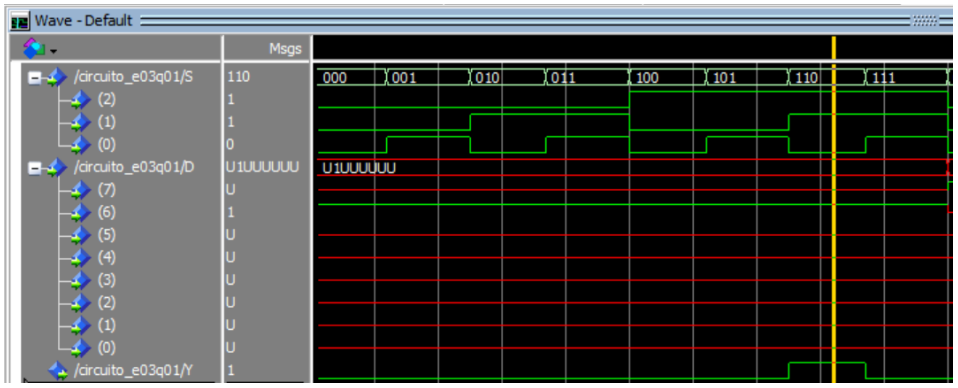

Figura 10:Y=D6 para S=110

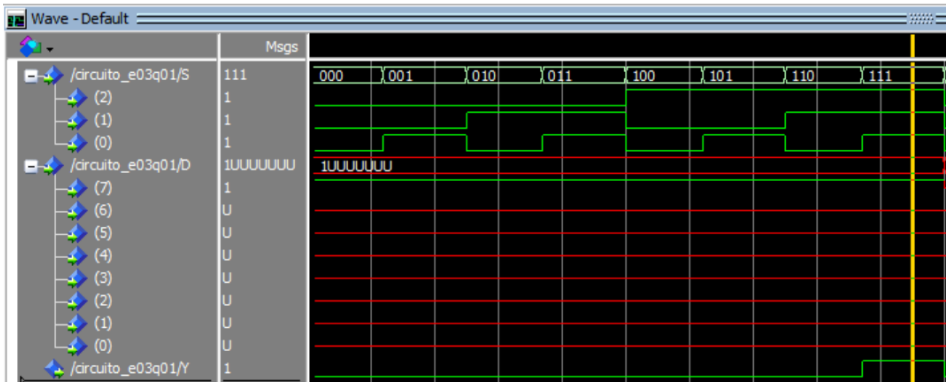

Figura 11:Y=D7 para S=111

Podemos ver pelos gráficos acima que a tabela-verdade foi atendida – isto é, para cada um dos possíveis valores do vetor S, o sinal do vetor Y foi igual ao respectivo sinal do vetor D. A partir dos gráficos acima, montando a tabela verdade completa, temos que:

Bit Ativo	D7	D6	D5	D4	D3	D2	D1	D0	S2	S1	S0	Y	Figura de referência
D0	0	0	0	0	0	0	0	1	0	0	0	1	Figura 4:Y=D0 para S=000
D0	0	0	0	0	0	0	0	1	0	0	1	0	
D0	0	0	0	0	0	0	0	1	0	1	0	0	
D0	0	0	0	0	0	0	0	1	0	1	1	0	
D0	0	0	0	0	0	0	0	1	1	0	0	0	
D0	0	0	0	0	0	0	0	1	1	0	1	0	
D0	0	0	0	0	0	0	0	1	1	1	0	0	
D0	0	0	0	0	0	0	0	1	1	1	1	0	
D1	0	0	0	0	0	0	1	0	0	0	0	0	
D1	0	0	0	0	0	0	1	0	0	0	1	1	Figura 5:Y=D1 para S=001
D1	0	0	0	0	0	0	1	0	0	1	0	0	
D1	0	0	0	0	0	0	1	0	0	1	1	0	
D1	0	0	0	0	0	0	1	0	1	0	0	0	
D1	0	0	0	0	0	0	1	0	1	0	1	0	
D1	0	0	0	0	0	0	1	0	1	1	0	0	
D1	0	0	0	0	0	0	1	0	1	1	1	0	
D2	0	0	0	0	0	1	0	0	0	0	0	0	
D2	0	0	0	0	0	1	0	0	0	0	1	0	
D2	0	0	0	0	0	1	0	0	0	1	0	1	Figura 6:Y=D2 para S=010
D2	0	0	0	0	0	1	0	0	0	1	1	0	
D2	0	0	0	0	0	1	0	0	1	0	0	0	
D2	0	0	0	0	0	1	0	0	1	0	1	0	
D2	0	0	0	0	0	1	0	0	1	1	0	0	
D2	0	0	0	0	0	1	0	0	1	1	1	0	
D3	0	0	0	0	1	0	0	0	0	0	0	0	
D3	0	0	0	0	1	0	0	0	0	0	1	0	
D3	0	0	0	0	1	0	0	0	0	1	0	0	
D3	0	0	0	0	1	0	0	0	0	1	1	1	Figura 7:Y=D3 para S=011
D3	0	0	0	0	1	0	0	0	1	0	0	0	
D3	0	0	0	0	1	0	0	0	1	0	1	0	
D3	0	0	0	0	1	0	0	0	1	1	0	0	
D3	0	0	0	0	1	0	0	0	1	1	1	0	
D4	0	0	0	1	0	0	0	0	0	0	0	0	
D4	0	0	0	1	0	0	0	0	0	0	1	0	
D4	0	0	0	1	0	0	0	0	0	1	0	0	
D4	0	0	0	1	0	0	0	0	0	1	1	0	
D4	0	0	0	1	0	0	0	0	1	0	0	1	Figura 8:Y=D4 para S=100
D4	0	0	0	1	0	0	0	0	1	0	1	0	
D4	0	0	0	1	0	0	0	0	1	1	0	0	
D4	0	0	0	1	0	0	0	0	1	1	1	0	
D5	0	0	1	0	0	0	0	0	0	0	0	0	
D5	0	0	1	0	0	0	0	0	0	0	1	0	
D5	0	0	1	0	0	0	0	0	0	1	0	0	
D5	0	0	1	0	0	0	0	0	0	1	1	0	
D5	0	0	1	0	0	0	0	0	1	0	0	0	
D5	0	0	1	0	0	0	0	0	1	0	1	1	Figura 9:Y=D5 para S=101
D5	0	0	1	0	0	0	0	0	1	1	0	0	
D5	0	0	1	0	0	0	0	0	1	1	1	0	
D6	0	1	0	0	0	0	0	0	0	0	0	0	
D6	0	1	0	0	0	0	0	0	0	0	1	0	
D6	0	1	0	0	0	0	0	0	0	1	0	0	
D6	0	1	0	0	0	0	0	0	0	1	1	0	
D6	0	1	0	0	0	0	0	0	1	0	0	0	
D6	0	1	0	0	0	0	0	0	1	0	1	0	
D6	0	1	0	0	0	0	0	0	1	1	0	1	Figura 10:Y=D6 para S=110
D6	0	1	0	0	0	0	0	0	1	1	1	0	
D7	1	0	0	0	0	0	0	0	0	0	0	0	
D7	1	0	0	0	0	0	0	0	0	0	1	0	
D7	1	0	0	0	0	0	0	0	0	1	0	0	
D7	1	0	0	0	0	0	0	0	0	1	1	0	
D7	1	0	0	0	0	0	0	0	1	0	0	0	
D7	1	0	0	0	0	0	0	0	1	0	1	0	
D7	1	0	0	0	0	0	0	0	1	1	0	0	
D7	1	0	0	0	0	0	0	0	1	1	1	1	Figura 11:Y=D7 para S=111

Tabela 2:Tabela-Verdade montada a partir da simulação do multiplexador 8x1

Comparando a Tabela 2 com a Tabela 1 temos que os resultados da simulação condizem com a tabela verdade para o multiplexador 8 para 1.

Experimento 02- Decodificador 4 para 16

O decodificador 4 para 16 pode ser arquitetado utilizando como entrada um vetor A de 4 bits e como saída um vetor Y de 16 bits, possuindo a seguinte tabela-verdade:

A	Y
0000	0000 0000 0000 0001
0001	0000 0000 0000 0010
0010	0000 0000 0000 0100
0011	0000 0000 0000 1000
0100	0000 0000 0001 0000
0101	0000 0000 0010 0000
0110	0000 0000 0100 0000
0111	0000 0000 1000 0000
1000	0000 0001 0000 0000
1001	0000 0010 0000 0000
1010	0000 0100 0000 0000
1011	0000 1000 0000 0000
1100	0001 0000 0000 0000
1101	0010 0000 0000 0000
1110	0100 0000 0000 0000
1111	1000 0000 0000 0000

Tabela 3:Tabela-Verdade para o Decodificador 4 para 16

Com as atribuições **WITH-SELECT** podemos facilmente modelar o comportamento da tabela-verdade acima no ModelSim:

```
-- Experimento 03 - Questão 02
-- Aluno: Pedro Henrique Dias Avelar 241037112
-- Turma 08
-- Data: 13/11/2024
-- Decodificador 4x16
-- Entrada: Vetor A (4bits)
-- Saída: Vetor Y (16 bits)
--Tabela Verdade:
--  A    |          Y
-- 0000    0000 0000 0000 0001  -- 1
-- 0001    0000 0000 0000 0010  -- 2
-- 0010    0000 0000 0000 0100  -- 3
-- 0011    0000 0000 0000 1000  -- 4
-- 0100    0000 0000 0001 0000  -- 5
-- 0101    0000 0000 0010 0000  -- 6
-- 0110    0000 0000 0100 0000  -- 7
-- 0111    0000 0000 1000 0000  -- 8
-- 1000    0000 0001 0000 0000  -- 9
-- 1001    0000 0010 0000 0000  -- 10
-- 1010    0000 0100 0000 0000  -- 11
-- 1011    0000 1000 0000 0000  -- 12
-- 1100    0001 0000 0000 0000  -- 13
-- 1101    0010 0000 0000 0000  -- 14
-- 1110    0100 0000 0000 0000  -- 15
-- 1111    1000 0000 0000 0000  -- 16
-- Usar atribuições seletivas WITH-SELECT

LIBRARY IEEE;
USE IEEE.STD_LOGIC_1164.ALL;

ENTITY CIRCUITO_E03Q02 IS
    PORT (A: IN STD_LOGIC_VECTOR   (3 DOWNTO 0);       -- ENTRADA 4 BITS
          Y: OUT STD_LOGIC_VECTOR (15 DOWNTO 0));          -- SAÍDA 16 BITS
END CIRCUITO_E03Q02;

ARCHITECTURE ARC_CIRCUITO_E03Q02 OF CIRCUITO_E03Q02 IS
BEGIN
    WITH A SELECT
    Y <= "0000000000000001" WHEN "0000",  -- 1
         "0000000000000010" WHEN "0001",  -- 2
         "0000000000000100" WHEN "0010",  -- 3
         "0000000000001000" WHEN "0011",  -- 4
         "0000000000010000" WHEN "0100",  -- 5
         "0000000000100000" WHEN "0101",  -- 6
         "0000000001000000" WHEN "0110",  -- 7
         "0000000010000000" WHEN "0111",  -- 8
         "0000000100000000" WHEN "1000",  -- 9
         "0000001000000000" WHEN "1001",  -- 10
         "0000010000000000" WHEN "1010",  -- 11
         "0000100000000000" WHEN "1011",  -- 12
         "0001000000000000" WHEN "1100",  -- 13
         "0010000000000000" WHEN "1101",  -- 14
         "0100000000000000" WHEN "1110",  -- 15
         "1000000000000000" WHEN "1111",  -- 16
         "0000000000000000" WHEN OTHERS;
END ARC_CIRCUITO_E03Q02;
```

Código 3:Decodificador de 4 para 16

Com o comando “WITH A SELECT” podemos então atribuir os valores desejados para o vetor Y com base nos valores do vetor A.

Para a realização da simulação, o vetor A foi configurado na opção clock com os períodos A(0)=25ps, A(1)=50ps, A(2)=100ps e A(3)=200ps:

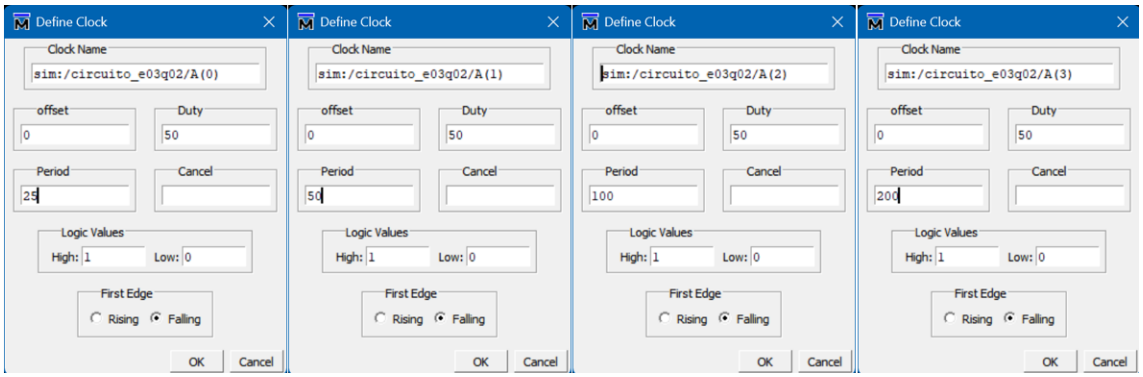

Figura 12:Ajuste de clock para o vetor A

O tempo de simulação foi ajustado para 200ps. Após rodar a simulação foi obtido o seguinte gráfico:

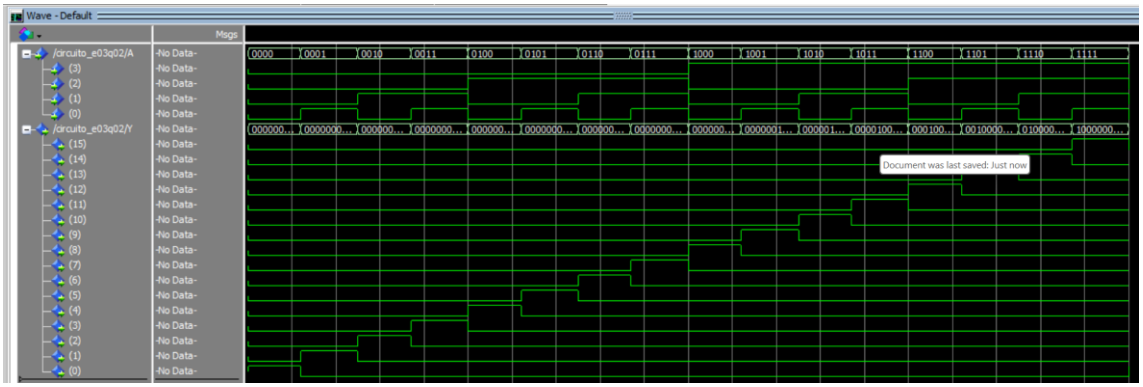

Figura 13:Resultado da simulação para o decodificador 4x16

Destacando para cada um dos 16 valores do vetor A, temos então os seguintes resultados:

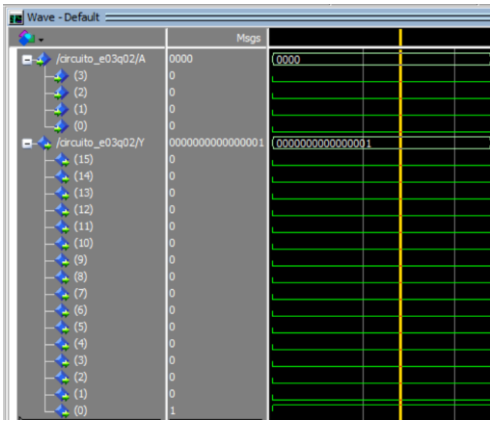

Figura 14:Y=0000 0000 0000 0001 para A=0000

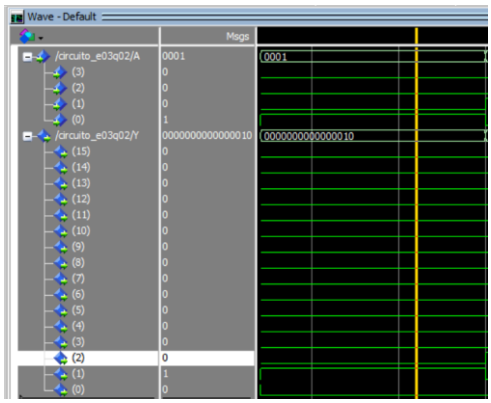

Figura 15:Y=0000 0000 0000 0010 para A=0001

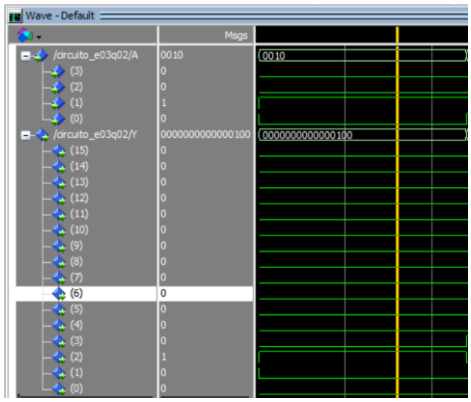

Figura 16:Y=0000 0000 0000 0100 para A=0010

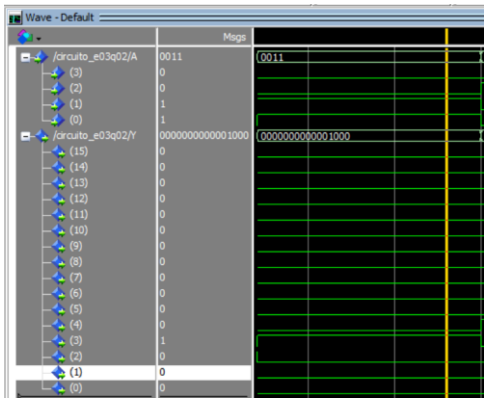

Figura 17:Y=0000 0000 0000 1000 para A=0011

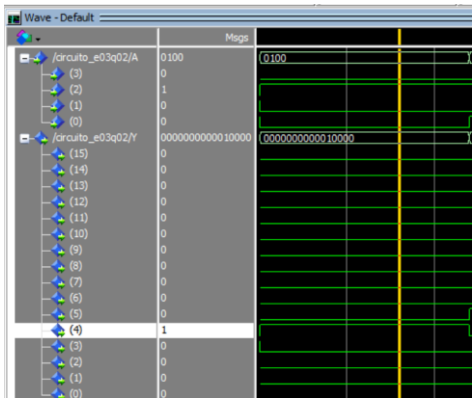

Figura 18:Y=0000 0000 0001 0000 para A=0100

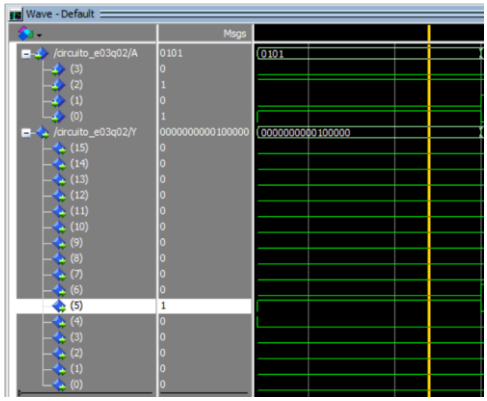

Figura 19:Y=0000 0000 0010 0000 para A=0101

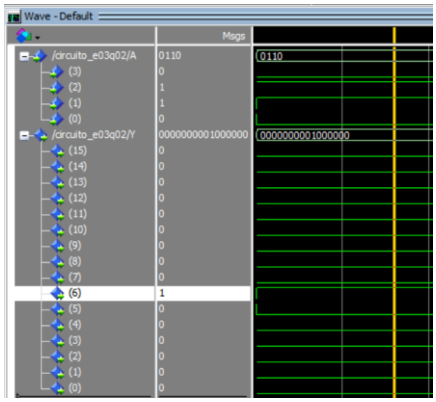

Figura 20:Y=0000 0000 0100 0000 para A=0110

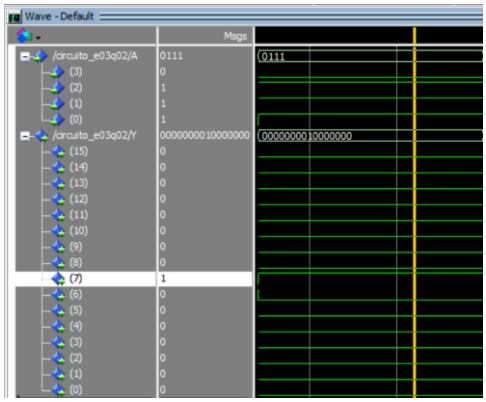

Figura 21:Y=0000 0000 1000 0000 para A=0111

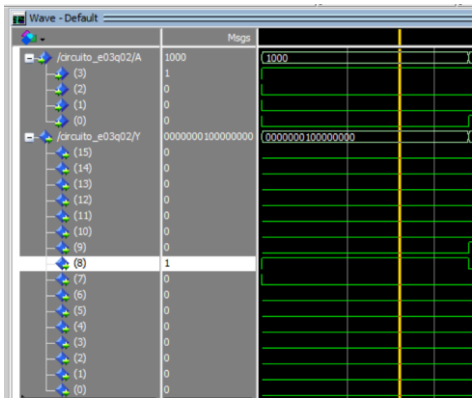

Figura 22:Y=0000 0001 0000 0000 para A=1000

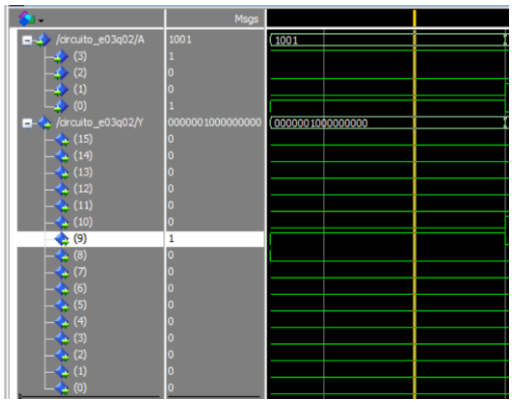

Figura 23:Y=0000 0010 0000 0000 para A=1001

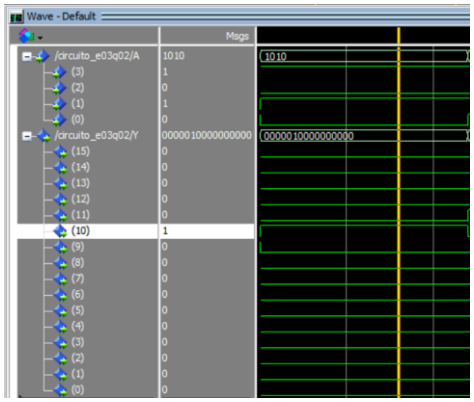

Figura 24:Y=0000 0100 0000 0000 para A=1010

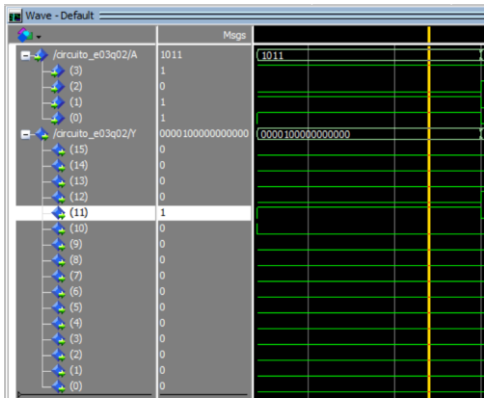

Figura 25:Y=0000 1000 0000 0000 para A=1011

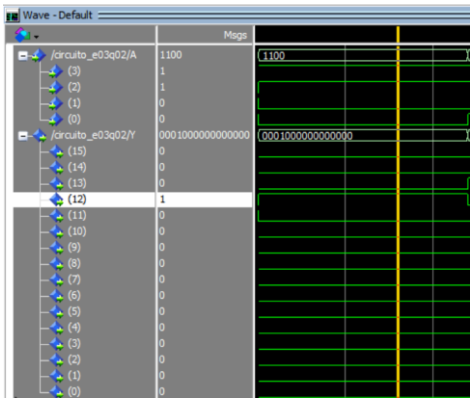

Figura 26:Y=0001 0000 0000 0000 para A=1100

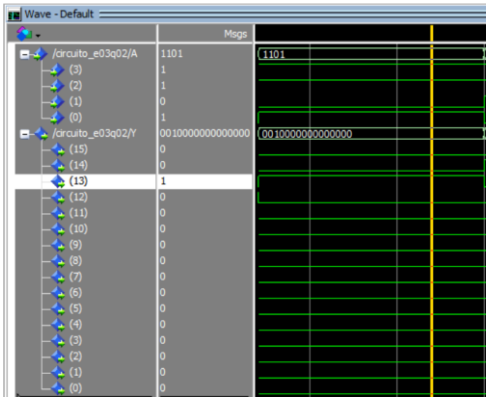

Figura 27:Y=0010 0000 0000 0000 para A=1101

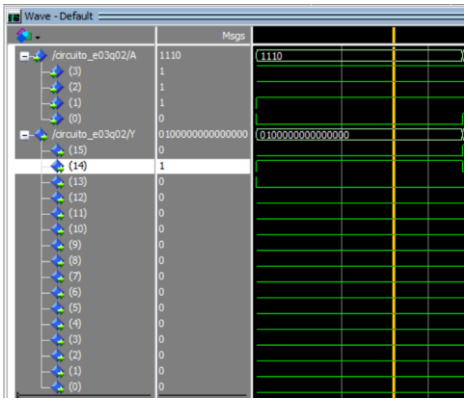

Figura 28:Y=0100 0000 0000 0000 para A=1110

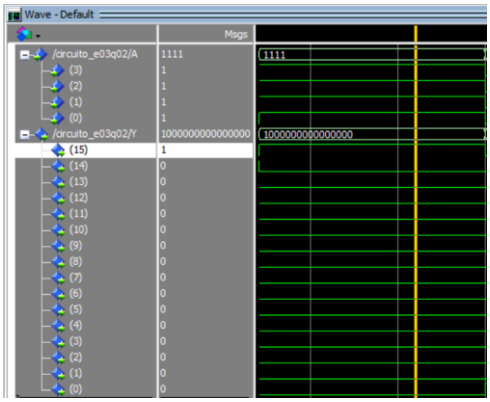

Figura 29:Y=1000 0000 0000 0000 para A=1111

A partir dos resultados obtidos da simulação podemos então montar a tabela verdade:

A	Y	Figura de Referência
0000	0000 0000 0000 0001	Figura 14
0001	0000 0000 0000 0010	Figura 15
0010	0000 0000 0000 0100	Figura 16
0011	0000 0000 0000 1000	Figura 17
0100	0000 0000 0001 0000	Figura 18
0101	0000 0000 0010 0000	Figura 19
0110	0000 0000 0100 0000	Figura 20
0111	0000 0000 1000 0000	Figura 21
1000	0000 0001 0000 0000	Figura 22
1001	0000 0010 0000 0000	Figura 23
1010	0000 0100 0000 0000	Figura 24
1011	0000 1000 0000 0000	Figura 25
1100	0001 0000 0000 0000	Figura 26
1101	0010 0000 0000 0000	Figura 27
1110	0100 0000 0000 0000	Figura 28
1111	1000 0000 0000 0000	Figura 29

Tabela 4:Tabela-verdade montada a partir da simulação do decodificador 4x16

Comparando as tabelas 3 e 4 temos que os resultados da simulação condizem com a tabela-verdade do decodificador 4x16.