

Relatório 7 VHDL - Turma 05
Yan Tavares de Oliveira
202014323



Introdução

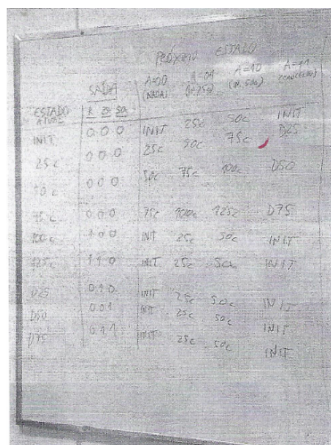
Este experimento consiste na criação de uma máquina de estados síncrona do tipo Moore para descrever o funcionamento de uma máquina de refrigerantes que aceita moedas como pagamento. Para descrever melhor o funcionamento da máquina, irei usar a descrição do experimento presente no relatório da disciplina.

“Implementar em VHDL e simular no ModelSim uma máquina de estado síncrona do tipo Moore para controlar uma máquina de refrigerantes que aceita moedas de R\$ 0,25 e R\$ 0,50. A cada transição do clock, a máquina deve contar o dinheiro inserido e liberar o refrigerante (e o troco) assim que a soma totalizar ou exceder R\$ 1,00. A máquina deve aceitar qualquer combinação de moedas de R\$ 0,25 e R\$ 0,50, independentemente da ordem em que as moedas foram inseridas. A qualquer momento (desde que a contagem ainda não tenha alcançado R\$ 1,00) o usuário poderá cancelar a compra e a máquina deve, também na transição do clock, devolver a quantia inserida.

Considere que a máquina só dispõe de um sabor de refrigerante (ou que a escolha do refrigerante é feita antes da máquina de estados iniciar). Logo, o refrigerante é liberado automaticamente (mas na transição do clock) após a inserção do valor de R\$ 1,00 com ou sem troco, não sendo necessário pressionar nenhum botão após a inserção do montante para receber o refrigerante. Isto impede a possibilidade, por exemplo, da inserção do valor de R\$ 1,50.”

Teoria

Podemos escrever a tabela de próximos estados da máquina de refrigerantes, que depende somente dos estados anteriores, visto que estamos tratando de uma máquina de Moore.



The image shows a handwritten state transition table for a vending machine. The table is titled 'PRÓXIMO ESTADO' and has columns for 'ESTADO ATUAL' and 'PRÓXIMO ESTADO'. The rows represent the current state, and the columns represent the next state based on the input (0.25 or 0.50). The table is as follows:

ESTADO ATUAL	0.25	0.50
INIT	0.00	0.25
0.25	0.00	0.50
0.50	0.00	1.00
1.00	0.00	0.75
0.75	0.00	1.00
1.25	0.00	0.75
1.50	0.00	1.00
0.00	0.00	0.25
0.25	0.00	0.50
0.50	0.00	1.00
1.00	0.00	0.75
0.75	0.00	1.00
1.25	0.00	0.75
1.50	0.00	1.00

Imagem 1: tabela de próximos estados da máquina de refrigerante

Códigos

Na **questão 1**, utilizamos as bibliotecas IEEE e IEEE.std_logic_1164.all para descrever o funcionamento da máquina.

```

library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity q1 is
    port ( A: in std_logic_vector (1 downto 0);
          clock, rst: in std_logic;
          B, C, D : out std_logic);
end q1;

architecture q1_arch of q1 is
    type State is (INIT, v_25, v_50, v_75,
                  liberou, liberou_25, cancela_25, cancela_50, cancela_75);

    signal currentstate, nextstate: state;

begin
    sync_proc: process(clock)
    begin
        if rising_edge(clock) then
            currentstate <= nextstate;
        end if;
    end process sync_proc;

    combinacao: process(currentstate, A)
    begin
        case currentstate is
            when INIT =>
                B <= '0';
                C <= '0';
                D <= '0';
                if (A = "00" or A = "11") then
                    nextstate <= INIT;
                elsif (A = "01") then
                    nextstate <= v_25;
                elsif (A = "10") then
                    nextstate <= v_50;
                end if;

            when v_25 =>
                B <= '0';
                C <= '0';
                D <= '0';
                if (A = "00") then
                    nextstate <= v_25;
                elsif (A = "01") then
                    nextstate <= v_50;
                elsif (A = "10") then
                    nextstate <= v_75;
                elsif (A = "11") then
                    nextstate <= cancela_25;
                end if;

            when v_50 =>
                B <= '0';
                C <= '0';
                D <= '0';
                if (A = "00") then
                    nextstate <= v_50;
                elsif (A = "01") then
                    nextstate <= v_75;
                elsif (A = "10") then
                    nextstate <= liberou;
                elsif (A = "11") then
                    nextstate <= cancela_50;
                end if;

            when v_75 =>
                B <= '0';
                C <= '0';
                D <= '0';
                if (A = "00") then
                    nextstate <= v_75;
                elsif (A = "01") then
                    nextstate <= liberou;
                elsif (A = "10") then
                    nextstate <= liberou_25;
                elsif (A = "11") then
                    nextstate <= cancela_75;
                end if;

            when liberou =>
                B <= '1';
                C <= '0';
                D <= '0';
                nextstate <= INIT;

            when liberou_25 =>
                B <= '1';
                C <= '1';
                D <= '0';
                nextstate <= INIT;

            when cancela_25 =>
                B <= '0';
                C <= '1';
                D <= '0';
                nextstate <= INIT;

            when cancela_50 =>
                B <= '0';
                C <= '0';
                D <= '1';
                nextstate <= INIT;

            when cancela_75 =>
                B <= '0';
                C <= '1';
                D <= '1';
                nextstate <= INIT;

            when others =>
                B <= '0';
                C <= '0';
                D <= '0';
                nextstate <= INIT;
        end case;
    end process combinacao;
end q1_arch;

```

Imagem 2: código principal da questão 1

Foi também criado um código auxiliar de testbench para gerarmos os estímulos necessários para abranger todas as combinações possíveis de entradas. Para ajustar o tempo do clock, foi criada a variável “Tempo”.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;

entity tb_q1 is
end tb_q1;

architecture tb_q1arch of tb_q1 is
  component q1 is
    PORT ( A: in std_logic_vector (1 downto 0);
           clock, rst: in std_logic;
           B, C, D : out std_logic);
  end component;

  constant Tempo:time:= 5 ns;
  signal clock: std_logic;
  signal rst: std_logic;
  signal moeda: std_logic_vector (1 downto 0);

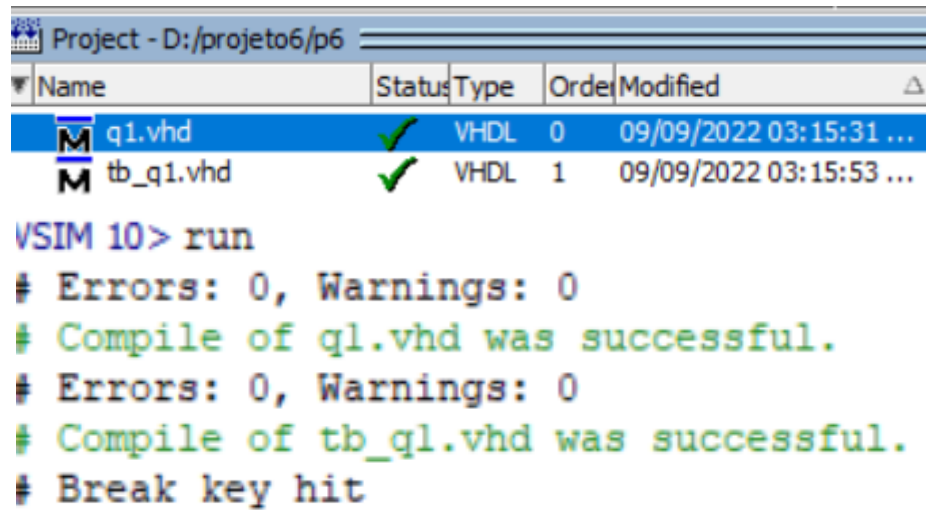
  begin
    machine: q1 PORT map(moeda, clock, rst, open, open, open);
    mudar:process
    begin
      clock <= '1', '0' after Tempo/2, '0' after Tempo;
      wait for Tempo;
    end process mudar;

    captura:process
    begin
      rst <= '0';
      wait for 5 ns;
      moeda <= "00";
      wait for 5 ns;
      moeda <= "10";
      wait for 5 ns;
      moeda <= "01";
      wait for 5 ns;
      moeda <= "10";
      wait for 5 ns;
      moeda <= "00";
      wait for 5 ns;
      moeda <= "01";
      wait for 5 ns;
      moeda <= "10";
      wait for 5 ns;
      moeda <= "00";
      wait for 5 ns;
      moeda <= "01";
      wait for 5 ns;
      moeda <= "10";
      wait for 5 ns;
      moeda <= "11";
      wait for 5 ns;
      moeda <= "00";
      wait for 5 ns;
      moeda <= "01";
      wait for 5 ns;
      moeda <= "01";
      wait for 5 ns;
      moeda <= "11";
      wait for 5 ns;
      moeda <= "00";
      wait for 5 ns;
      moeda <= "01";
      wait for 5 ns;
      moeda <= "11";
      wait for 5 ns;
      moeda <= "00";
      wait for 5 ns;
      moeda <= "01";
      wait for 5 ns;
      moeda <= "11";
      wait for 5 ns;
    end process captura;
  end tb_q1arch;
```

Imagem 3. Código auxiliar da questão 1

Compilação

Abaixo estão as mensagens de compilação do projeto, com nenhum erro sendo apresentado em nenhum dos casos



The image shows a screenshot of a project window and a terminal window. The project window, titled 'Project - D:/projeto6/p6', displays a table of files with their status, type, order, and modification date. Below the table, the terminal window shows the output of the 'run' command, indicating successful compilation of both 'q1.vhd' and 'tb_q1.vhd' with no errors or warnings.

Name	Status	Type	Order	Modified
q1.vhd	✓	VHDL	0	09/09/2022 03:15:31 ...
tb_q1.vhd	✓	VHDL	1	09/09/2022 03:15:53 ...

```
VSIM 10> run
# Errors: 0, Warnings: 0
# Compile of q1.vhd was successful.
# Errors: 0, Warnings: 0
# Compile of tb_q1.vhd was successful.
# Break key hit
```

Imagem 4. Mensagem de compilação

Simulação

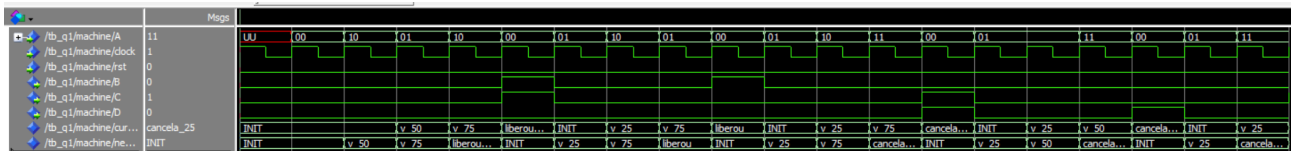


Figura 5. Simulação em forma de onda binária da máquina.

Análise

Para determinar a ordem dos bits utilizados na simulação e para abranger todas as combinações possíveis, foram utilizados clocks para alterar o valor bit a bit. Pudemos perceber um resultado condizente com a tabela verdade proposta na seção Teoria do relatório.

Conclusão

Como não houve divergência alguma nos dois casos, podemos então afirmar que implementamos uma máquina de refrigerantes funcional por meio de uma simulação no software ModelSim.