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 07 – Máquinas de Estado de Moore

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

Introdução

O presente experimento tem o seguinte objetivo:

- Implementar uma máquina de estados síncrona do tipo Moore em VHDL e a simular no ModelSim

Atividade

Implemente em VHDL e simule no ModelSim uma máquina de estados síncrona do tipo Moore para controlar uma máquina de vendas que aceita moedas de 25 centavos e 50 centavos. A cada transição do clock, a máquina deve contar o dinheiro inserido e liberar o produto e o troco assim que a soma totalizar ou exceder 1 real. A máquina deve aceitar qualquer combinação de moedas de 25 centavos e 50 centavos, em qualquer ordem. O usuário pode cancelar a compra a qualquer momento, desde que a soma das moedas ainda esteja abaixo de 1 real.

Considere que a máquina vende apenas um produto e que ele é liberado automaticamente quando a soma das moedas inseridas atingir ou exceder 1 real, com ou sem troco. Considere também que o usuário faz, no máximo, uma ação a cada período do clock. Isso impede, por exemplo, a possibilidade de o usuário inserir R$ 1,50.

A entidade VHDL deve ter duas entradas: um clock (de um bit) e um vetor A de dois bits. Se A = 01, foi inserida uma moeda de 25 centavos, se A = 10, foi inserida uma moeda de 50 centavos, se A = 11, o usuário solicitou o cancelamento da compra e, se A = 00, não houve ação do usuário. As saídas serão três, todas de um bit: uma para indicar se a máquina liberou o produto, outra para indicar se a máquina devolveu uma moeda de 25 centavos e outra para indicar se a máquina devolveu uma moeda de 50 centavos. Após o fim da venda (pela liberação do produto ou cancelamento), a máquina deve voltar ao estado inicial. Uma vez que a soma das moedas atinja ou exceda 1 real ou que o usuário cancele a compra, a máquina só aceitará novas moedas após voltar ao estado inicial.

ATENÇÃO! Considere os seguintes pontos:

- Antes de começar a escrever o código, desenhe o diagrama de transição de estados, você deve incluí-lo no relatório.
- Lembre-se que a variável A tem quatro valores possíveis, logo, para cada estado, pode haver até quatro transições possíveis.
- Em alguns casos, não serão possíveis todos os valores de A . Por exemplo, se a máquina já acumulou 1 real ou mais, não é mais possível inserir moedas nem cancelar a compra, logo, o único valor de entrada possível neste estado é A=00 (seu código deve simplesmente ignorar outros valores de entrada, eles seriam fisicamente impossíveis em uma implementação real).

Uma máquina de estados de Moore é um tipo de máquina de estados finita onde as saídas são determinadas apenas pelo estado atual da máquina. Em outras palavras, para cada estado, há uma saída específica associada a ele. No caso de uma máquina de Moore síncrona, as transições de estado irão ocorrer no evento de sincronia, sendo este geralmente a borda de subida do clock.

O processo de arquitetura de uma máquina de estados geralmente envolve as seguintes etapas:

- Preparo do diagrama de transição de estados
- Preparo da tabela de transição de estados
- Atribuição de códigos aos estados da tabela de transição
- Construção das equações do próximo estado a partir da tabela de transição de estados
- Construção das equações das saídas a partir das equações de próximo estado
- Montagem do circuito de forma a atender as equações de próximo estado e das saídas

Por meio da linguagem VHDL podemos modelar e simular uma máquina que atenda os requisitos propostos sem necessariamente descrever a arquitetura física do circuito – no caso, pensando no conceito de caixa preta, precisamos que a máquina modelada apenas atenda os requisitos de entrada e saída. Embora dessa forma o modelo não seja possível de ser implantado fisicamente, ainda assim ele terá utilidade como *"golden module"* para a realização de testes.

Mesmo para este modelo ideal, ainda assim se faz necessário preparar o diagrama de transição e a tabela de transição de estados. Estes dois componentes irão definir qual o comportamento esperado para a máquina.

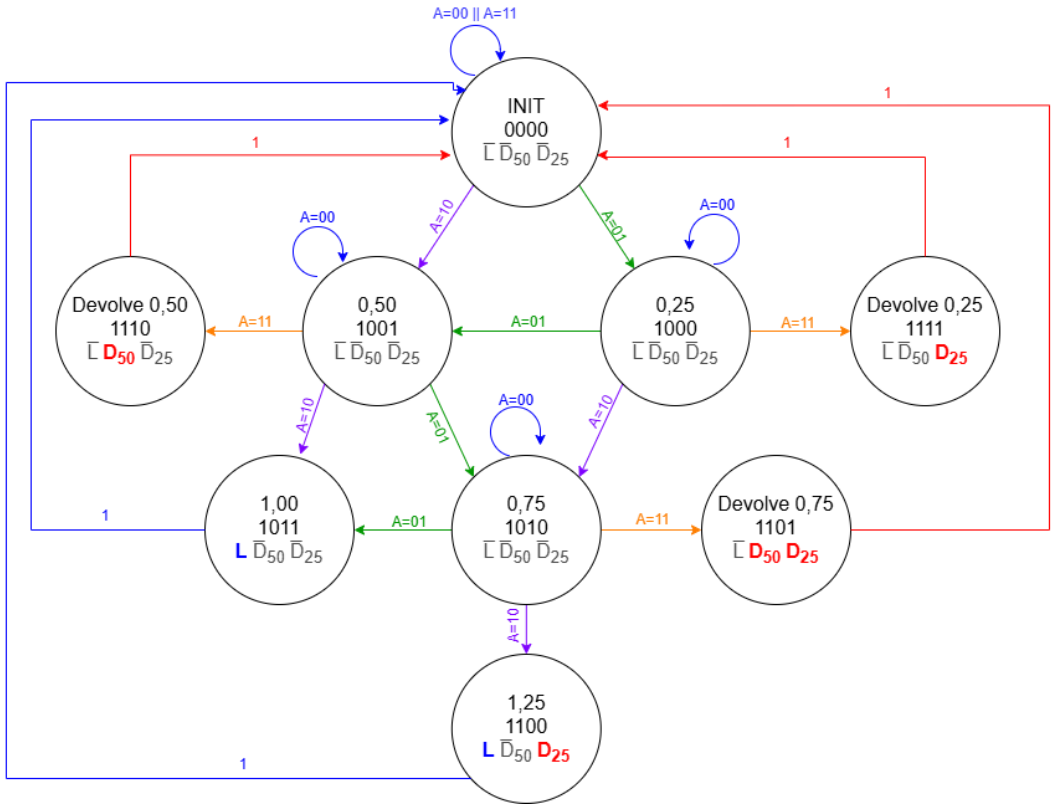

Figura 1:Diagrama de Transição de Estados para a máquina de refrigerante proposta – Imagem do autor feita com a ferramenta online draw.io

Para a máquina proposta foram definidos 9 estados: INIT; 0,25; 0,50; 0,75; 1,00; 1,25; Devolve 0,75; Devolve 0,50; Devolve 0,25. De acordo com o comportamento definido no enunciado, os estados INIT, 0,25, 0,50 e 0,75 devem aceitar todas as combinações possíveis de entrada. A partir do momento em que a máquina atinge 1 real, situação representada pelos estados 1,00 e 1,25, ou que seja solicitada a devolução, situação representada pelos estados Devolve 0,75, Devolve 0,50 e Devolve 0,25, a máquina deve então impedir a inserção de novas moedas e voltar ao estado inicial INIT, momento em que voltará a aceitar novas moedas.

Nome	Estado				Entrada - A1A0				Saídas		
	S4	S3	S2	S1	00 (fazer nada)	01 (moeda de 0,25)	11 pedir devoluçao	10 (moeda de 0,50)	L	D50	D25
INIT	0	0	0	0	0000 (INIT)	1000 (0,25)	0000 (INIT)	1001 (0,50)	0	0	0
Estados não utilizados (Dont Care)	0	0	0	1	xxxx	xxxx	xxxx	xxxx	0	0	0
	0	0	1	0	xxxx	xxxx	xxxx	xxxx	0	0	0
	0	0	1	1	xxxx	xxxx	xxxx	xxxx	0	0	0
	0	1	0	0	xxxx	xxxx	xxxx	xxxx	0	0	0
	0	1	0	1	xxxx	xxxx	xxxx	xxxx	0	0	0
	0	1	1	0	xxxx	xxxx	xxxx	xxxx	0	0	0
	0	1	1	1	xxxx	xxxx	xxxx	xxxx	0	0	0
0,25	1	0	0	0	1000 (0,25)	1001 (0,50)	1111 (Devolve 0,25)	1010 (0,75)	0	0	0
0,50	1	0	0	1	1001 (0,50)	1010 (0,75)	1110 (Devolve 0,25)	1011 (1,00)	0	0	0
0,75	1	0	1	0	1010 (0,75)	1011 (1,00)	1101 (Devolve 0,75)	1100 (1,25)	0	0	0
1,00	1	0	1	1	0000 (INIT)	0000 (INIT)	0000 (INIT)	0000 (INIT)	1	0	0
1,25	1	1	0	0	0000 (INIT)	0000 (INIT)	0000 (INIT)	0000 (INIT)	1	0	1
Devolve 0,75	1	1	0	1	0000 (INIT)	0000 (INIT)	0000 (INIT)	0000 (INIT)	0	1	1
Devolve 0,50	1	1	1	0	0000 (INIT)	0000 (INIT)	0000 (INIT)	0000 (INIT)	0	1	0
Devolve 0,25	1	1	1	1	0000 (INIT)	0000 (INIT)	0000 (INIT)	0000 (INIT)	0	0	1

Tabela 1: Tabela de Transição de estados para a máquina de refrigerantes proposta

Em uma máquina real, o estado da máquina é registrado com o uso de flip-flops. Para n flip-flops, é possível registrar 2^n estados. Como foi necessário criar 9 estados, em tese com 3 flip-flops seria possível representar apenas 2^3=8 estados, sendo assim um número insuficiente de flip-flops. No entanto, com 4 flip-flops temos então 2^4=16 estados possíveis; como apenas 9 estados foram designados, temos 7 estados não utilizados.

Há duas abordagens para lidar com esses estados, a abordagem de custo mínimo e a abordagem de risco mínimo. No custo mínimo, designamos *don't cares* para os estados não utilizados. Esta abordagem, embora mais barata, apresenta o risco de, por alguma situação inesperada, a máquina acabar transitando para um desses estados e, a partir daí, tendo então um comportamento não definido. A outra alternativa, risco mínimo, envolve planejar para os estados não utilizados a transição para um determinado estado, de forma a assegurar o conhecimento do comportamento da máquina independente do estado atual e da próxima transição. Para esta máquina, por exemplo, uma boa alternativa seria regredir dos estados não

utilizado para o estado INIT. No entanto, como o experimento está sendo modelado como *"golden module"*, iremos então assumir que a máquina é perfeita e incapaz de transitar para um dos estados indefinidos, e assim então será modelada sob a abordagem de custo mínimo. A máquina foi modelada através do código a seguir:

```
0   -- Experimento 07
1   -- Aluno: Pedro Henrique Dias Avelar 241037112
2   -- Turma 08
3   -- Data: 25/01/2025
4
5   -- Maquina de Estado Síncrona de Moore - Máquina de Refrigerante
6
7   LIBRARY IEEE;
8   USE IEEE.STD_LOGIC_1164.ALL;
9   USE WORK.type_estado.ALL;
10
11
12  ENTITY MAQ_MOORE_REFRI is
13      PORT (
14            CLK         : IN  STD_LOGIC;
15            A           : IN  STD_LOGIC_VECTOR (1 DOWNTO 0);
16            L, D50, D25: OUT STD_LOGIC;
17            CURRENT_STATE: OUT ESTADO
18            );
19  END MAQ_MOORE_REFRI;
20
21  ARCHITECTURE ARC_MAQ_MOORE_REFRI OF MAQ_MOORE_REFRI IS
22
23      SIGNAL ESTADO_ATUAL, PROXIMO_ESTADO: ESTADO;
24
25      BEGIN
26          CURRENT_STATE <= ESTADO_ATUAL;
27          PROCESSO_SINCRONO: PROCESS (CLK)
28          BEGIN
29              IF RISING_EDGE(CLK) THEN
30                  ESTADO_ATUAL <= PROXIMO_ESTADO;
31              END IF;
32          END PROCESS PROCESSO_SINCRONO;
33
34          PROCESSO_COMBINACIONAL: PROCESS (ESTADO_ATUAL, A)
35          BEGIN
36              L   <= '0';
37              D50 <= '0';
38              D25 <= '0';
39              CASE ESTADO_ATUAL IS
40                  WHEN INIT =>
41                      CASE A IS
42                          WHEN "00" =>   PROXIMO_ESTADO <= INIT;
43                          WHEN "01" =>   PROXIMO_ESTADO <= VLR_25;
44                          WHEN "10" =>   PROXIMO_ESTADO <= VLR_50;
45                          WHEN "11" =>   PROXIMO_ESTADO <= INIT;
46                          WHEN OTHERS => PROXIMO_ESTADO <= INIT;
47                      END CASE;
48                  WHEN VLR_25 =>
49                      CASE A IS
50                          WHEN "00" =>   PROXIMO_ESTADO <= VLR_25;
51                          WHEN "01" =>   PROXIMO_ESTADO <= VLR_50;
52                          WHEN "10" =>   PROXIMO_ESTADO <= VLR_75;
53                          WHEN "11" =>   PROXIMO_ESTADO <= DEVOLVE_25;
54    -- EM CASO DE ALGUM DEFEITO, A MÁQUINA IRÁ DEVOLVER OS 25 CENTAVOS
55                          WHEN OTHERS => PROXIMO_ESTADO <= DEVOLVE_25;
56                      END CASE;
```

```
57                  WHEN VLR_50 =>
58                      CASE A IS
59                          WHEN "00" =>   PROXIMO_ESTADO <= VLR_50;
60                          WHEN "01" =>   PROXIMO_ESTADO <= VLR_75;
61                          WHEN "10" =>   PROXIMO_ESTADO <= VLR_100;
62                          WHEN "11" =>   PROXIMO_ESTADO <= DEVOLVE_50;
63      -- EM CASO DE ALGUM DEFEITO, A MÁQUINA IRÁ DEVOLVER OS 50 CENTAVOS
64                          WHEN OTHERS => PROXIMO_ESTADO <= DEVOLVE_50;
65                      END CASE;                                    --
66                  WHEN VLR_75 =>
67                      CASE A is
68                          WHEN "00" =>   PROXIMO_ESTADO <= VLR_75;
69                          WHEN "01" =>   PROXIMO_ESTADO <= VLR_100;
70                          WHEN "10" =>   PROXIMO_ESTADO <= VLR_125;
71                          WHEN "11" =>   PROXIMO_ESTADO <= DEVOLVE_75;
72      -- EM CASO DE ALGUM DEFEITO, A MÁQUINA IRÁ DEVOLVER OS 75 CENTAVOS
73                          WHEN OTHERS => PROXIMO_ESTADO <= DEVOLVE_75;
74                      END CASE;
75      --PARA TODOS OS DEMAIS CASOS, O PROXIMO ESTADO SEMPRE SERA
76      --INIT. ASSIM QUE A MAQUINA CONSIGA 1 REAL OU SEJA SOLICITADO
77      --A DEVOLUCAO, SÓ SERÁ ACEITO NOVOS DEPÓSITOS APÓS A MÁQUINA VOLTAR
78      --AO ESTADO INICIAL
79                  WHEN VLR_100 =>
80                      PROXIMO_ESTADO <= INIT;
81                      L   <= '1'; -- LIBERA O REFRI
82                      WHEN VLR_125 =>
83                      PROXIMO_ESTADO <= INIT;
84                      L   <= '1';   -- LIBERA O REFRI
85                      D25 <= '1';   -- DEVOLVE 25 CENTAVOS
86                  WHEN DEVOLVE_75 =>
87                      PROXIMO_ESTADO <= INIT;
88                      D50 <= '1'; -- DEVOLVE 50 CENTAVOS
89                      D25 <= '1'; -- DEVOLVE 25 CENTAVOS
90                  WHEN DEVOLVE_50 =>
91                      PROXIMO_ESTADO <= INIT;
92                      D50 <= '1'; -- DEVOLVE 50 CENTAVOS
93                  WHEN DEVOLVE_25 =>
94                      PROXIMO_ESTADO <= INIT;
95                      D25 <= '1'; -- DEVOLVE 25 CENTAVOS
96              END CASE;
97          END PROCESS PROCESSO_COMBINACIONAL;
98  END ARC_MAQ_MOORE_REFRI;
```

Código 1:Modelagem da máquina de refrigerante

A arquitetura da máquina de refrigerantes é divida em dois processos. No processo síncrono (linha 27) temos a atualização do estado da máquina a cada borda de subida do clock. No processo combinacional, temos a definição do próximo estado da máquina de acordo com a entrada A e o estado atual da máquina, replicando o comportamento definido na tabela 1. Além disso, temos a ativação das saídas nos respectivos estados informados na tabela 1. Na linha 13 do código há a instrução USE WORK.type_estado.ALL; o tipo estado foi definido no arquivo type_estado.vhd, apresentado a seguir:

```
1   library IEEE;
2   use IEEE.STD_LOGIC_1164.ALL;
3
4   package type_estado is
5   TYPE ESTADO IS (INIT, VLR_25, VLR_50, VLR_75, VLR_100, VLR_125,
DEVOLVE_75, DEVOLVE_50, DEVOLVE_25);
6   end package type_estado;
```

Código 2: Definição do tipo customizado ESTADO

O tipo de dado estado é uma simples lista enumerada. Poderíamos utilizar a representação binária definida na tabela 1 para representar os estados, mas o uso da lista enumerada auxilia na escrita e na visualização do código. Além disso, na entidade MAQ_MOORE_REFRI definida na linha 12 do código 1, foi adicionado uma saída do tipo estado. Embora não possua significado físico, mais a frente será possível ver que, com essa saída, podemos representar o estado da máquina na simulação do testbench, facilitando muito na visualização dos resultados. Segue o código do testbench:

```
0  -- Experimento 07 - TESTBENCH
1   -- Aluno: Pedro Henrique Dias Avelar 241037112
2   -- Turma 08
3   -- Data: 25/01/2025
4
5   -- Testbench - Tempo de simulação: ?
6
7   LIBRARY IEEE;
8   USE IEEE.STD_LOGIC_1164.ALL;
9   USE IEEE.STD_LOGIC_UNSIGNED.ALL;
10  USE IEEE.NUMERIC_STD.ALL;
11  USE WORK.type_estado.ALL;
12
13  ENTITY TESTBENCH_E07_MAQ_REFRI IS
14  END TESTBENCH_E07_MAQ_REFRI;
15
16  ARCHITECTURE ARC_TESTBENCH_E07_MAQ_REFRI OF TESTBENCH_E07_MAQ_REFRI IS
17  COMPONENT MAQ_MOORE_REFRI IS
18      PORT (
19            CLK         : IN  STD_LOGIC;
20            A           : IN  STD_LOGIC_VECTOR (1 DOWNTO 0);
21            L, D50, D25: OUT STD_LOGIC;
22            CURRENT_STATE: OUT ESTADO
23            );
24  END COMPONENT;
25
26      SIGNAL CLK_TB: STD_LOGIC := '0';
27      SIGNAL A_TB: STD_LOGIC_VECTOR(1 DOWNTO 0);
28      SIGNAL L_TB, D50_TB, D25_TB: STD_LOGIC;
29      SIGNAL CURRENT_STATE_TB: ESTADO;
30
31      BEGIN
32          DUT: MAQ_MOORE_REFRI PORT MAP (CLK_TB, A_TB, L_TB, D50_TB,
D25_TB,CURRENT_STATE_TB);
33          CLK_TB <= NOT CLK_TB AFTER 5 NS;
34          PROCESS
35          BEGIN
36          REPORT "INICIANDO TESTE..." SEVERITY NOTE;
37  -- CENARIO 1: INIT -> VLR_25 -> DEVOLVE_25 -> INIT
38  --A:               01         11            XX
39          A_TB <= "01"; WAIT FOR 10 NS;
40          A_TB <= "11"; WAIT FOR 10 NS;
41          A_TB <= "00"; WAIT FOR 10 NS;
42  -- CENARIO 2: INIT -> VLR_25 -> VLR_50 -> DEVOLVE_50 -> INIT
43  --A:               01         01         11            XX
44          A_TB <= "01"; WAIT FOR 10 NS;
45          A_TB <= "01"; WAIT FOR 10 NS;
46          A_TB <= "11"; WAIT FOR 10 NS;
47          A_TB <= "00"; WAIT FOR 10 NS;
48  -- CENARIO 3: INIT -> VLR_50 -> DEVOLVE_50 -> INIT
49  --A:               10         11            XX
50          A_TB <= "10"; WAIT FOR 10 NS;
51          A_TB <= "11"; WAIT FOR 10 NS;
52          A_TB <= "00"; WAIT FOR 10 NS;
```

```
53  -- CENARIO 4: INIT -> VLR_25 -> VLR_50 -> VLR_75 -> DEVOLVE_75 -> INIT
54  --A:               01        01        01        11            XX
55          A_TB <= "01"; WAIT FOR 10 NS;
56          A_TB <= "01"; WAIT FOR 10 NS;
57          A_TB <= "01"; WAIT FOR 10 NS;
58          A_TB <= "11"; WAIT FOR 10 NS;
59          A_TB <= "00"; WAIT FOR 10 NS;
60  -- CENARIO 5: INIT -> VLR_25 -> VLR_75 -> DEVOLVE_75 -> INIT
61  --A:               01        10        11            XX
62          A_TB <= "01"; WAIT FOR 10 NS;
63          A_TB <= "10"; WAIT FOR 10 NS;
64          A_TB <= "11"; WAIT FOR 10 NS;
65          A_TB <= "00"; WAIT FOR 10 NS;
66  -- CENARIO 6: INIT -> VLR_50 -> VLR_75 -> DEVOLVE_75 -> INIT
67  --A:               10        01        11            XX
68          A_TB <= "10"; WAIT FOR 10 NS;
69          A_TB <= "01"; WAIT FOR 10 NS;
70          A_TB <= "11"; WAIT FOR 10 NS;
71          A_TB <= "00"; WAIT FOR 10 NS;
72  -- CENARIO 7: INIT -> VLR_25 -> VLR_50 -> VLR_75 -> VLR_100 -> INIT
73  --A:               01        01        01        01          XX
74          A_TB <= "01"; WAIT FOR 10 NS;
75          A_TB <= "01"; WAIT FOR 10 NS;
76          A_TB <= "01"; WAIT FOR 10 NS;
77          A_TB <= "01"; WAIT FOR 10 NS;
78          A_TB <= "00"; WAIT FOR 10 NS;
79  -- CENARIO 8: INIT -> VLR_25 -> VLR_50 -> VLR_100 -> INIT
80  --A:               01        01        10        XX
81          A_TB <= "01"; WAIT FOR 10 NS;
82          A_TB <= "01"; WAIT FOR 10 NS;
83          A_TB <= "10"; WAIT FOR 10 NS;
84          A_TB <= "00"; WAIT FOR 10 NS;
85  -- CENARIO 9: INIT -> VLR_25 -> VLR_75 -> VLR_100 -> INIT
86  --A:               01        10        01        XX
87          A_TB <= "01"; WAIT FOR 10 NS;
88          A_TB <= "10"; WAIT FOR 10 NS;
89          A_TB <= "01"; WAIT FOR 10 NS;
90          A_TB <= "00"; WAIT FOR 10 NS;
91  -- CENARIO 10: INIT -> VLR_50 -> VLR_100 -> INIT
92  --A:                10        10        XX
93          A_TB <= "10"; WAIT FOR 10 NS;
94          A_TB <= "10"; WAIT FOR 10 NS;
95          A_TB <= "00"; WAIT FOR 10 NS;
96  -- CENARIO 11: INIT -> VLR_25 -> VLR_50 -> VLR_75 -> VLR_125 -> INIT
97  --A:                01        01        01        10        XX
98          A_TB <= "01"; WAIT FOR 10 NS;
99          A_TB <= "01"; WAIT FOR 10 NS;
100         A_TB <= "01"; WAIT FOR 10 NS;
101         A_TB <= "10"; WAIT FOR 10 NS;
102         A_TB <= "00"; WAIT FOR 10 NS;
103 -- CENARIO 12: INIT -> VLR_25 -> VLR_75 -> VLR_125 -> INIT
104 --A:                01        10        10        XX
105         A_TB <= "01"; WAIT FOR 10 NS;
106         A_TB <= "10"; WAIT FOR 10 NS;
107         A_TB <= "10"; WAIT FOR 10 NS;
108         A_TB <= "00"; WAIT FOR 10 NS;
109 -- CENARIO 13: INIT -> VLR_50 -> VLR_75 -> VLR_125 -> INIT
110 --A:                10        01        10        XX
111         A_TB <= "10"; WAIT FOR 10 NS;
112         A_TB <= "01"; WAIT FOR 10 NS;
113         A_TB <= "10"; WAIT FOR 10 NS;
114         A_TB <= "00"; WAIT FOR 10 NS;
```

```
115 -- CENARIO 14: INIT -> INIT -> VLR_25 -> VLR_25 -> VLR_50 ->
116 --               00       01         00         01         00
117 --           VLR_50 -> VLR_75 -> VLR_75 -> VLR_100 -> INIT
118 --                  01         00         01          00
119         A_TB <= "00"; WAIT FOR 10 NS;
120         A_TB <= "01"; WAIT FOR 10 NS;
121         A_TB <= "00"; WAIT FOR 10 NS;
122         A_TB <= "01"; WAIT FOR 10 NS;
123         A_TB <= "00"; WAIT FOR 10 NS;
124         A_TB <= "01"; WAIT FOR 10 NS;
125         A_TB <= "00"; WAIT FOR 10 NS;
126         A_TB <= "01"; WAIT FOR 10 NS;
127         A_TB <= "00"; WAIT FOR 10 NS;
128         REPORT "TESTE FINALIZADO!" SEVERITY NOTE;
129         WAIT;
130         END PROCESS;
131 END ARC_TESTBENCH_E07_MAQ_REFRI;
```

Código 3:Testbench para a máquina de refrigerante

O testbench consistiu na realização de 14 diferentes cenários buscando passar pelas possíveis transições de estado esperadas no funcionamento da máquina. A simulação teve uma duração de 610 ns e apresentou o seguinte resultado:

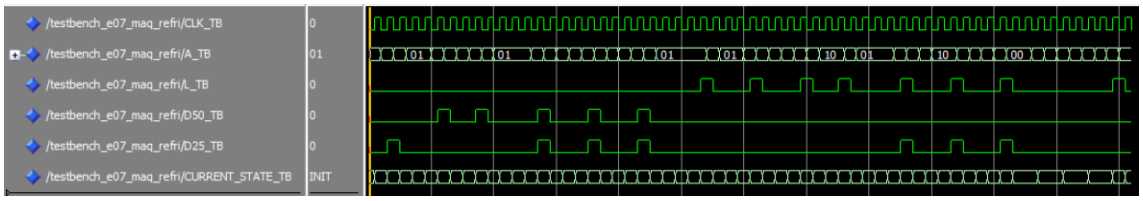

Figura 2:Resultado da simulação para a máquina de refrigerante

Como mencionado anteriormente, o uso da variável de saída CURRENT_STATE, presente na linha 22 do código 3, junto com sua atribuição ao sinal CURRENT_STATE_TB na linha 29 do mesmo código nos permite facilmente visualizar em qual estado a máquina se encontra no diagrama de ondas do Modelsim. Visualizando agora cenário a cenário, temos os seguintes resultados:

```
37  -- CENARIO 1: INIT -> VLR_25 -> DEVOLVE_25 -> INIT
38  --A:               01         11             XX
39          A_TB <= "01"; WAIT FOR 10 NS;
40          A_TB <= "11"; WAIT FOR 10 NS;
41          A_TB <= "00"; WAIT FOR 10 NS;
```

Código 4:Código do primeiro cenário de teste realizado

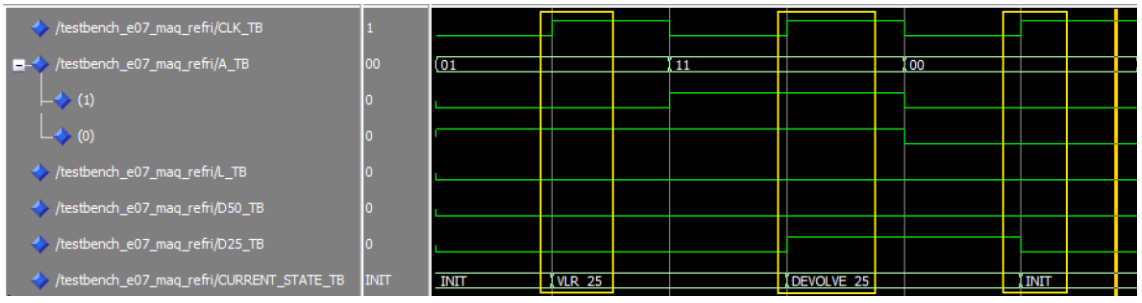

Figura 3:Resultado do primeiro cenário de teste

Na primeira borda de subida de clock com a entrada 01 (moeda de 25 centavos) a máquina sai do estado INIT para o estado VLR_25; em seguida na próxima borda de subida de clock com a entrada 11 (cancelamento da compra) a máquina passa para o estado DEVOLVE_25 e ativa a saída D25. Por fim, a máquina retorna ao estado INIT.

```
42  -- CENARIO 2: INIT -> VLR_25 -> VLR_50 -> DEVOLVE_50 -> INIT
43  --A:                01         01          11             XX
44          A_TB <= "01"; WAIT FOR 10 NS;
45          A_TB <= "01"; WAIT FOR 10 NS;
46          A_TB <= "11"; WAIT FOR 10 NS;
47          A_TB <= "00"; WAIT FOR 10 NS;
```

Código 5:Código do segundo cenário de teste realizado

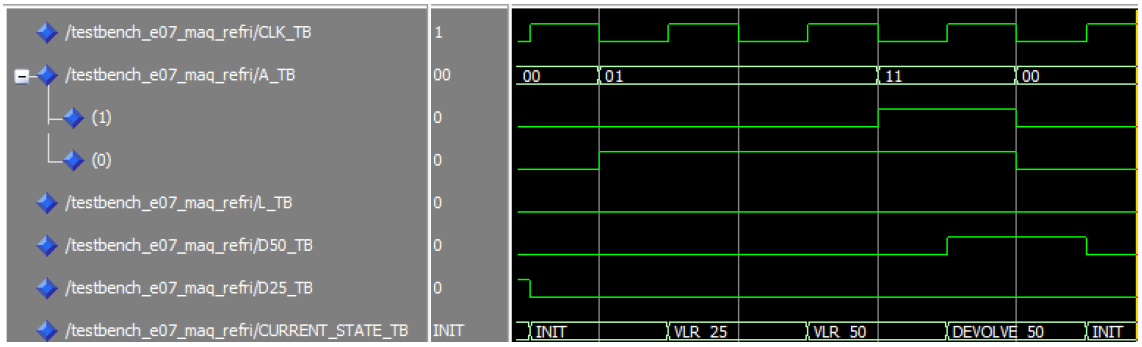

Figura 4:Resultado do segundo cenário de teste

A máquina passa por duas bordas de subida de clock com a entrada 01 (moeda de 25 centavos), transitando pelos estados VLR_25 e VLR_50. Em seguida é solicitada a devolução, o que faz a máquina ir para o estado DEVOLVE_50, o que simultaneamente ativa a saída D50. Por fim a máquina volta para o estado INIT.

```
48  -- CENARIO 3: INIT -> VLR_50 -> DEVOLVE_50 -> INIT
49  --A:                10         11             XX
50          A_TB <= "10"; WAIT FOR 10 NS;
51          A_TB <= "11"; WAIT FOR 10 NS;
52          A_TB <= "00"; WAIT FOR 10 NS;
```

Código 6:Código do terceiro cenário de teste realizado

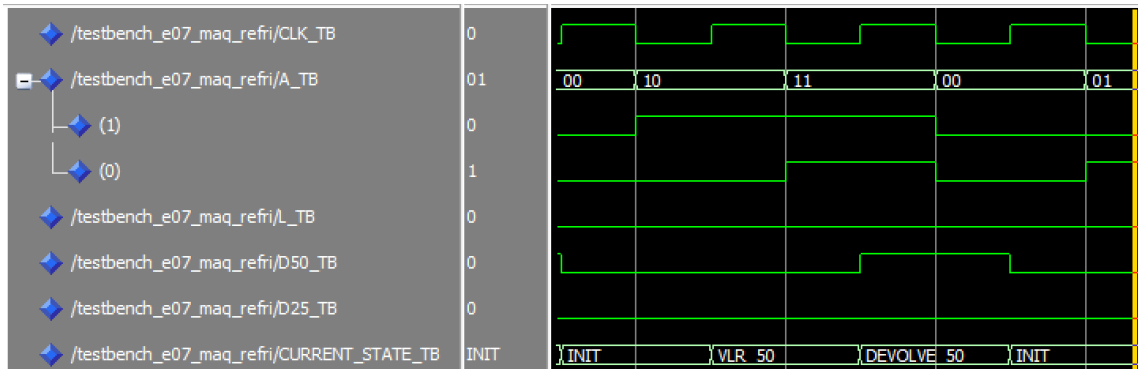

Figura 5:Resultado do terceiro cenário de teste

Após a inserção de uma moeda de 50 centavos (entrada 10), é solicitada a devolução, fazendo com que a máquina transite pelos estados VLR_50 e DEVOLVE_50, antes de voltar a INIT. No estado DEVOLVE_50 a saída D50 é ativada.

```
53  -- CENARIO 4: INIT -> VLR_25 -> VLR_50 -> VLR_75 -> DEVOLVE_75 -> INIT
54  --A:               01        01        01        11             XX
55          A_TB <= "01"; WAIT FOR 10 NS;
56          A_TB <= "01"; WAIT FOR 10 NS;
57          A_TB <= "01"; WAIT FOR 10 NS;
58          A_TB <= "11"; WAIT FOR 10 NS;
59          A_TB <= "00"; WAIT FOR 10 NS;
60  -- CENARIO 5: INIT -> VLR_25 -> VLR_75 -> DEVOLVE_75 -> INIT
61  --A:               01        10        11             XX
62          A_TB <= "01"; WAIT FOR 10 NS;
63          A_TB <= "10"; WAIT FOR 10 NS;
64          A_TB <= "11"; WAIT FOR 10 NS;
65          A_TB <= "00"; WAIT FOR 10 NS;
66  -- CENARIO 6: INIT -> VLR_50 -> VLR_75 -> DEVOLVE_75 -> INIT
67  --A:               10        01        11             XX
68          A_TB <= "10"; WAIT FOR 10 NS;
69           A_TB <= "01"; WAIT FOR 10 NS;
70           A_TB <= "11"; WAIT FOR 10 NS;
71           A_TB <= "00"; WAIT FOR 10 NS;
```

Código 7:Código do quarto, quinto e sexto cenário de teste realizado

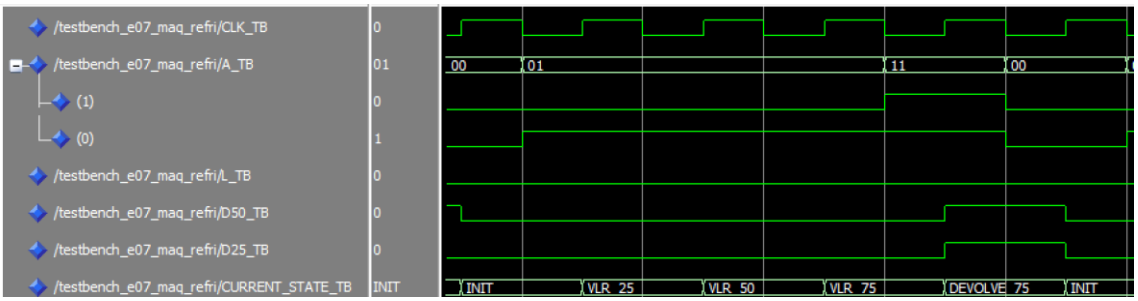

Figura 6:Resultado do quarto cenário de teste

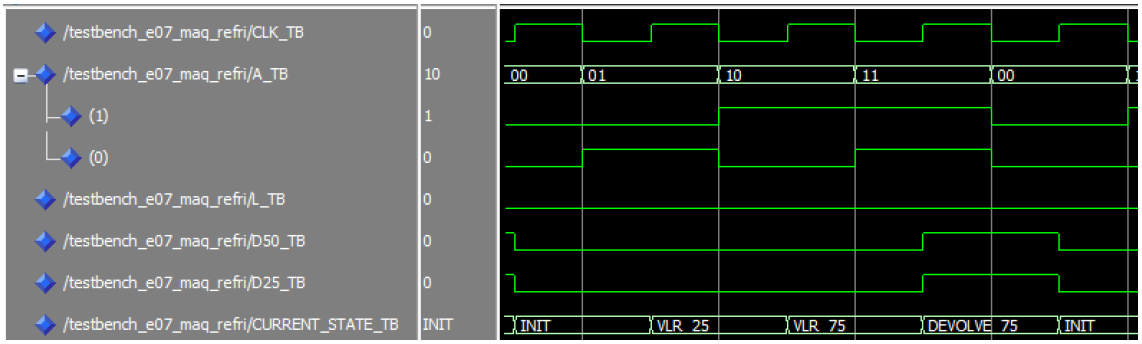

Figura 7:Resultado do quinto cenário de teste

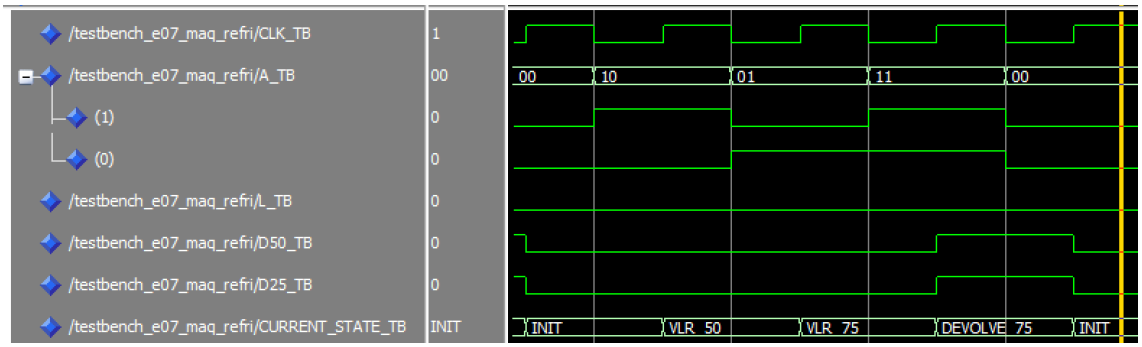

Figura 8:Resultado do sexto cenário de teste

O quarto, quinto e sexto cenário de teste envolve as possíveis permutações para que a máquina chegue ao estado VLR_75, seguido de um pedido de devolução. Ao chegar ao estado DEVOLVE_75, as saídas D50 e D25 são ativadas.

```
72  -- CENARIO 7: INIT -> VLR_25 -> VLR_50 -> VLR_75 -> VLR_100 -> INIT
73  --A:                01         01         01         01          XX
74          A_TB <= "01"; WAIT FOR 10 NS;
75          A_TB <= "01"; WAIT FOR 10 NS;
76          A_TB <= "01"; WAIT FOR 10 NS;
77          A_TB <= "01"; WAIT FOR 10 NS;
78          A_TB <= "00"; WAIT FOR 10 NS;
79  -- CENARIO 8: INIT -> VLR_25 -> VLR_50 -> VLR_100 -> INIT
80  --A:                01         01         10         XX
81          A_TB <= "01"; WAIT FOR 10 NS;
82          A_TB <= "01"; WAIT FOR 10 NS;
83          A_TB <= "10"; WAIT FOR 10 NS;
84          A_TB <= "00"; WAIT FOR 10 NS;
85  -- CENARIO 9: INIT -> VLR_25 -> VLR_75 -> VLR_100 -> INIT
86  --A:                01         10         01         XX
87          A_TB <= "01"; WAIT FOR 10 NS;
88          A_TB <= "10"; WAIT FOR 10 NS;
89          A_TB <= "01"; WAIT FOR 10 NS;
90          A_TB <= "00"; WAIT FOR 10 NS;
91  -- CENARIO 10: INIT -> VLR_50 -> VLR_100 -> INIT
92  --A:                 10         10         XX
93          A_TB <= "10"; WAIT FOR 10 NS;
94          A_TB <= "10"; WAIT FOR 10 NS;
95          A_TB <= "00"; WAIT FOR 10 NS;
```

Código 8:Código do sétimo, oitavo, nono e décimo cenário de teste realizado

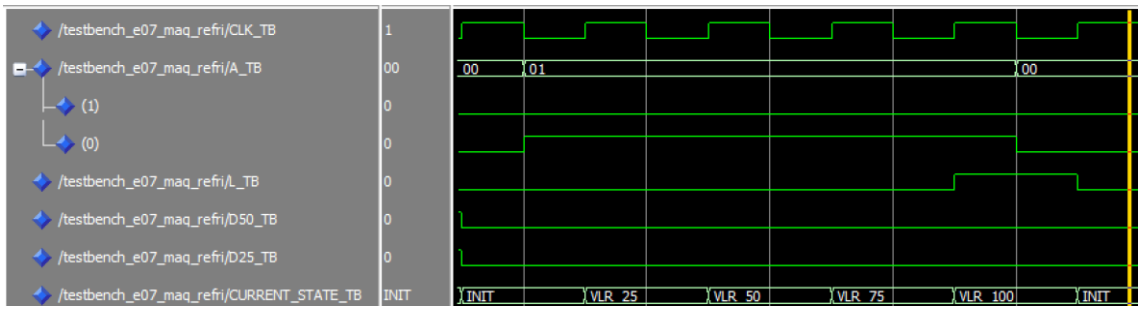

Figura 9:Resultado do sétimo cenário de teste

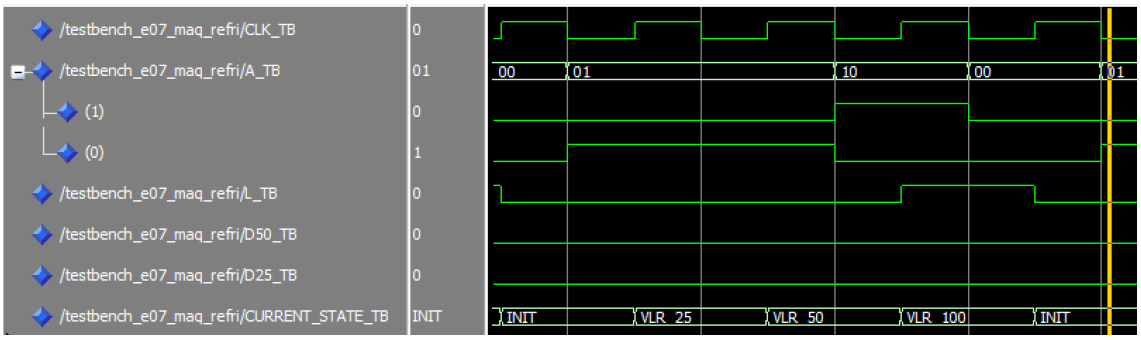

Figura 10:Resultado do oitavo cenário de teste

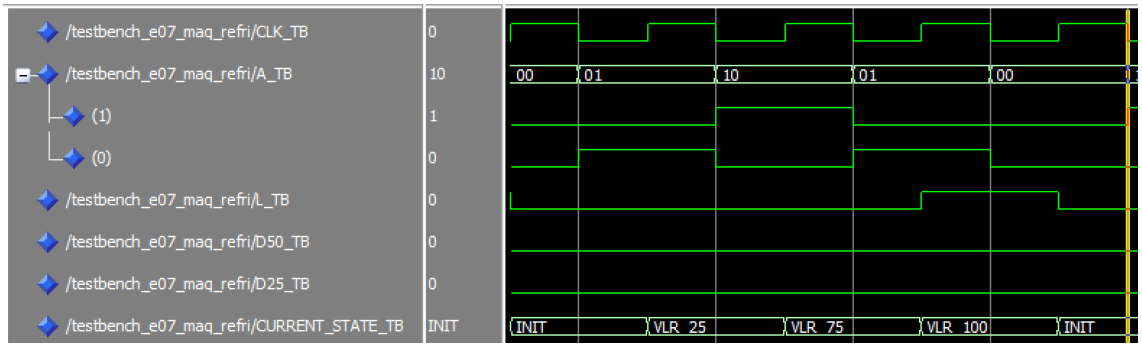

Figura 11:Resultado do nono cenário de teste

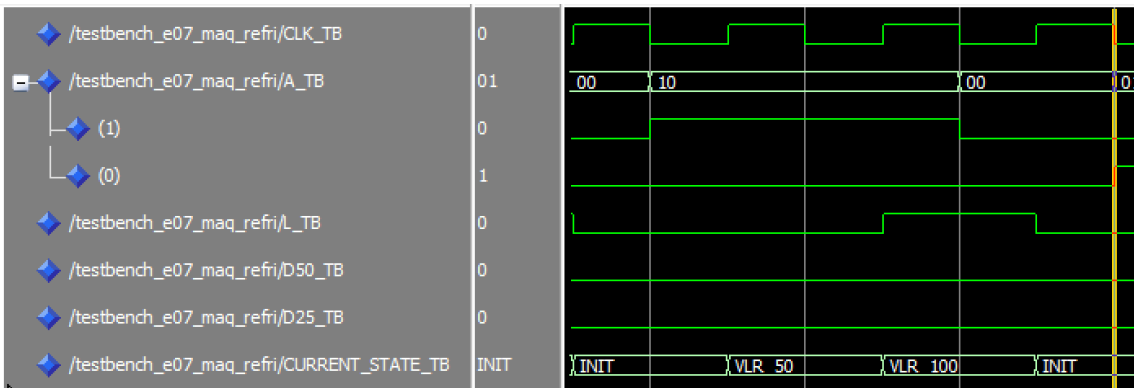

Figura 12:Resultado do décimo cenário de teste

Os cenários 7 a 10 representam 4 possíveis maneiras de chegar ao estado VLR_100. Evidencia-se que, ao chegar ao estado VLR_100, apenas a saída L é ativada.

```
96  -- CENARIO 11: INIT -> VLR_25 -> VLR_50 -> VLR_75 -> VLR_125 -> INIT
97  --A:                 01         01         01         10         XX
98          A_TB <= "01"; WAIT FOR 10 NS;
99          A_TB <= "01"; WAIT FOR 10 NS;
100         A_TB <= "01"; WAIT FOR 10 NS;
101         A_TB <= "10"; WAIT FOR 10 NS;
102         A_TB <= "00"; WAIT FOR 10 NS;
103 -- CENARIO 12: INIT -> VLR_25 -> VLR_75 -> VLR_125 -> INIT
104 --A:                 01         10         10          XX
105         A_TB <= "01"; WAIT FOR 10 NS;
106         A_TB <= "10"; WAIT FOR 10 NS;
107         A_TB <= "10"; WAIT FOR 10 NS;
108         A_TB <= "00"; WAIT FOR 10 NS;
109 -- CENARIO 13: INIT -> VLR_50 -> VLR_75 -> VLR_125 -> INIT
110 --A:                 10         01         10          XX
111         A_TB <= "10"; WAIT FOR 10 NS;
112         A_TB <= "01"; WAIT FOR 10 NS;
113         A_TB <= "10"; WAIT FOR 10 NS;
114         A_TB <= "00"; WAIT FOR 10 NS;
```

Código 9:Código do décimo primeiro, décimo segundo e décimo terceiro cenário de teste realizado

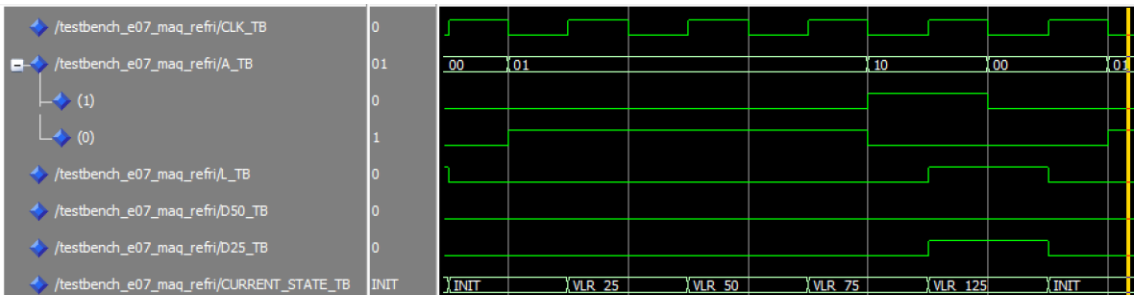

Figura 13:Resultado do décimo primeiro cenário de teste

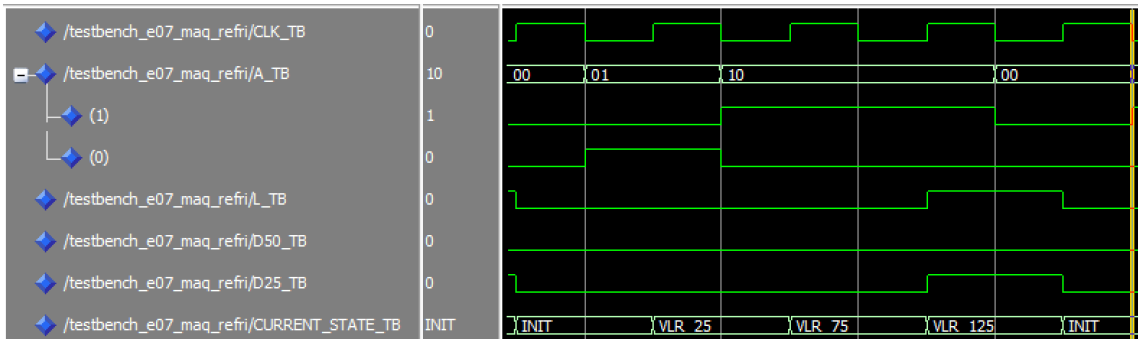

Figura 14:Resultado do décimo segundo cenário de teste

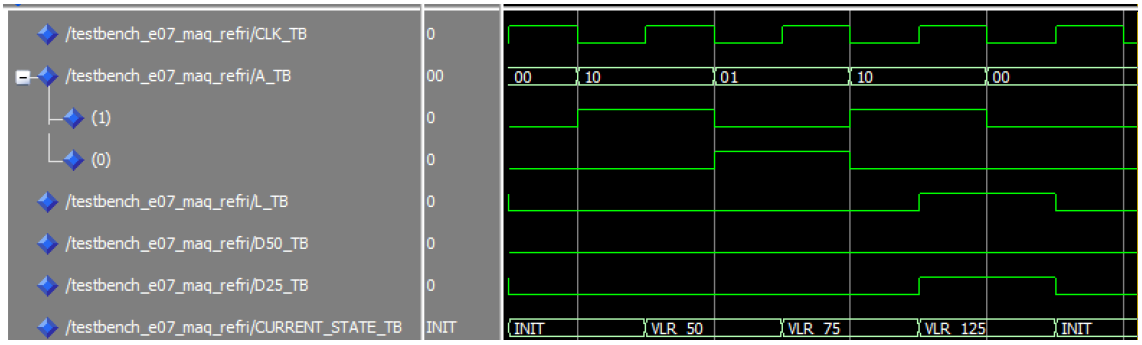

Figura 15:Resultado do décimo terceiro cenário de teste

Os cenários 11,12 e 13 envolvem diferentes percursos para chegar ao estado VLR_125. Neste estado, as saídas L e D25 são ativadas.

```
115 -- CENARIO 14: INIT -> INIT -> VLR_25 -> VLR_25 -> VLR_50 ->
116 --               00      01        00        01        00
117 --           VLR_50 -> VLR_75 -> VLR_75 -> VLR_100 -> INIT
118 --                  01        00        01          00
119         A_TB <= "00"; WAIT FOR 10 NS;
120         A_TB <= "01"; WAIT FOR 10 NS;
121         A_TB <= "00"; WAIT FOR 10 NS;
122         A_TB <= "01"; WAIT FOR 10 NS;
123         A_TB <= "00"; WAIT FOR 10 NS;
124         A_TB <= "01"; WAIT FOR 10 NS;
125         A_TB <= "00"; WAIT FOR 10 NS;
126         A_TB <= "01"; WAIT FOR 10 NS;
127         A_TB <= "00"; WAIT FOR 10 NS;
```

Figura 16:Código do décimo quarto cenário de teste realizado

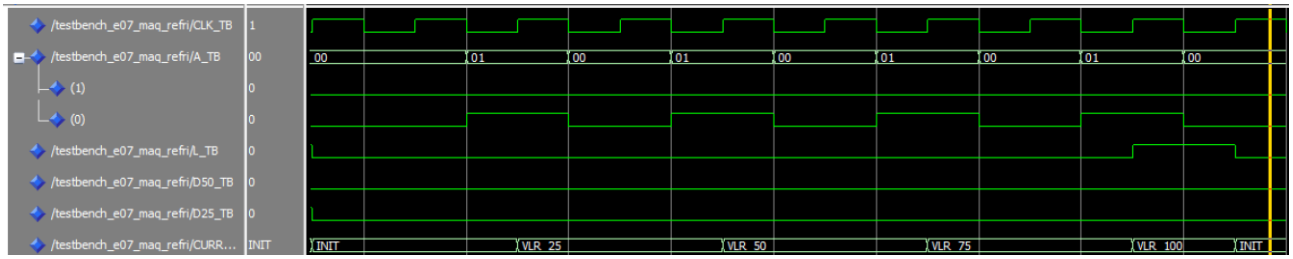

Figura 17:Resultado do décimo quarto cenário de teste

O último teste realizado visou avaliar o comportamento da máquina nos casos de inatividade do usuário (entrada 00). Podemos ver que nestes casos, o aparentemente estado se manteve; na prática o que ocorre é uma transição para o mesmo estado.