

# SD132 – Circuitos Digitais II

## A-207 Exemplos de Máquinas de Estado

---

Autores	
Gilmar Silva Beserra (UnB)	Daniel Muñoz Arboleda (UnB)
Nome 3 (INSTITUIÇÃO)	Nome 4 (INSTITUIÇÃO)

Histórico de revisões		
10/02/2025	V1.0	Versão inicial

# Tópicos

---

- Exemplos de Máquinas de Estado
- Exercícios

---

Página em branco.

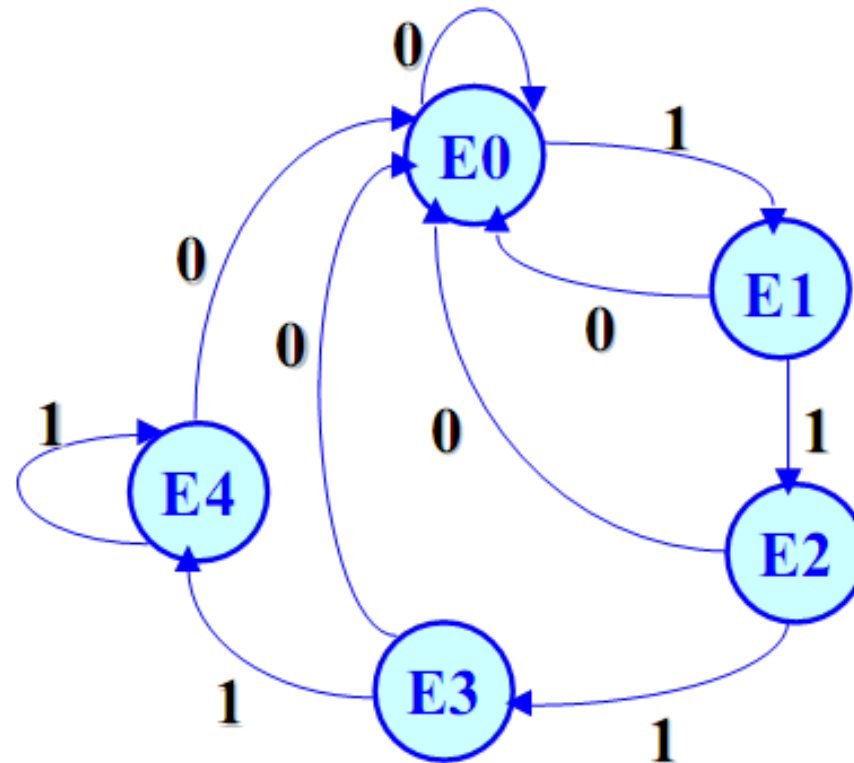
# Aula 22

## Exemplos de Máquinas de Estado

# Exemplo 1 – VHDL

---

- Detector de sequência de 4 ou mais 1's seguidos



# Exemplo 1 – VHDL

---

```
21 library IEEE;
22 use IEEE.std_logic_1164.all;
23 use IEEE.std_logic_arith.all;
24
25 entity ones_detector is
26 port (
27     clk      : in  std_logic;
28     reset    : in  std_logic;
29     input     : in  std_logic;
30     output    : out std_logic
31 );
32 end ones_detector;
33
34
35 architecture behavioral of ones_detector is
36
37     type t_state is (waiting,s1,s2,s3,s4);
38     signal state : t_state;
39
40
41 begin
```

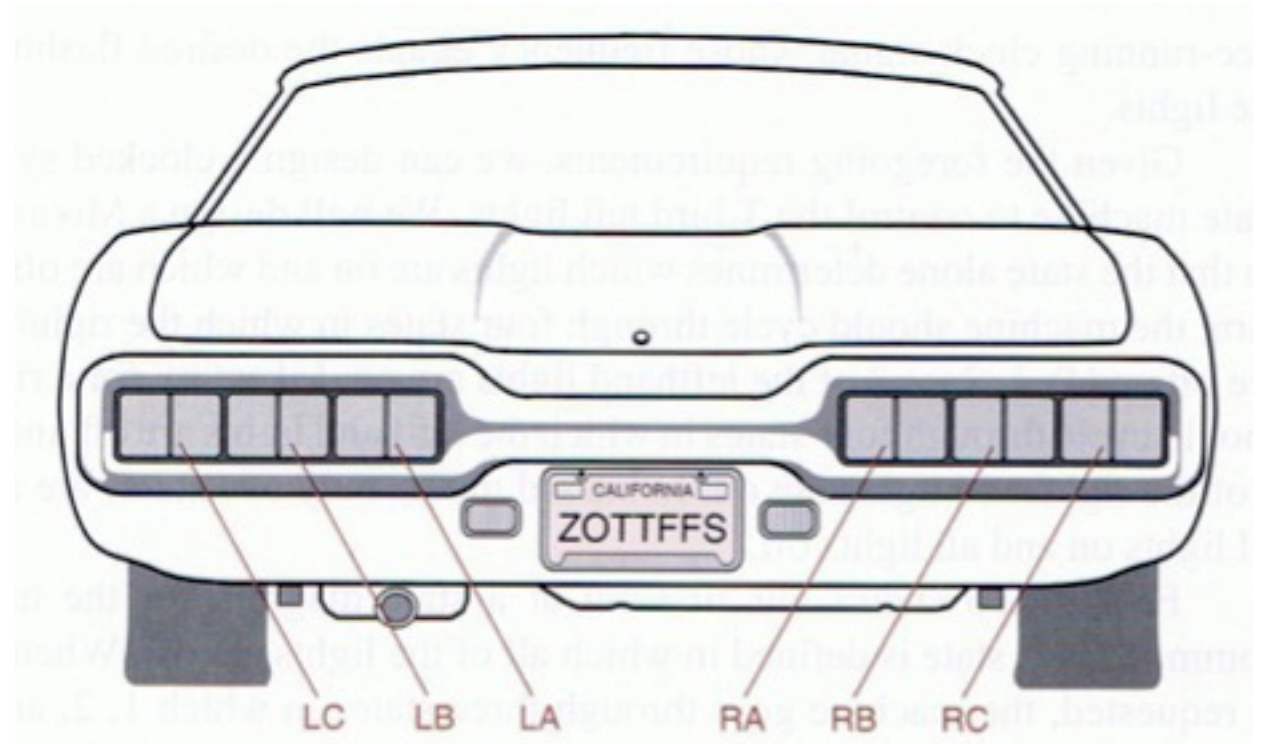
# Exemplo 1 – VHDL

```
44 process(reset,clk,input)
45 begin
46     if reset='1' then
47         output <= '0';
48         state <= waiting;
49     elsif rising_edge(clk) then
50         case state is
51             when waiting =>
52                 output <= '0';
53                 if input = '1' then
54                     state <= s1;
55                 else state <= waiting;
56                 end if;
57             when s1 =>
58                 output <= '0';
59                 if input = '1' then
60                     state <= s2;
61                 else state <= waiting;
62                 end if;
63             when s2 =>
64                 output <= '0';
65                 if input = '1' then
66                     state <= s3;
67                 else state <= waiting;
68                 end if;
69             when s3 =>
70                 output <= '0';
71                 if input = '1' then
72                     state <= s4;
73                 else state <= waiting;
74                 end if;
75             when s4 =>
76                 output <= '1';
77                 if input = '0' then
78                     state <= waiting;
79                 else state <= s4;
80                 end if;
81             when others => state <= waiting;
82         end case;
83     end if;
84 end process;
85
86 end behavioral;
```



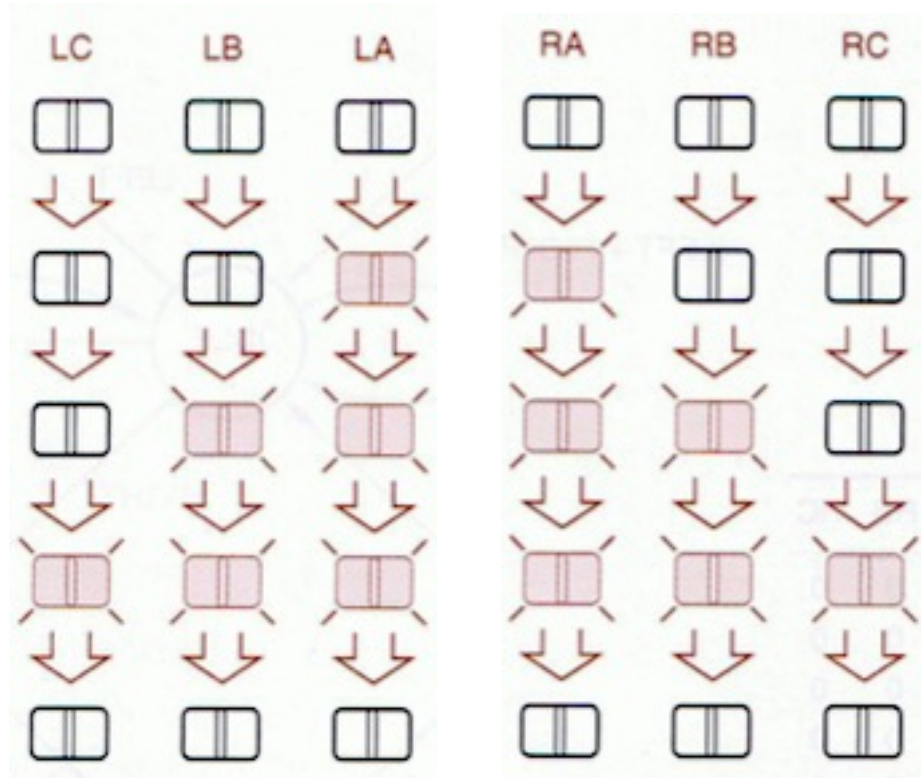
## Exemplo 2 – VHDL

- Controlador de lâmpadas do farol traseiro de um Ford Thunderbird 1965
- **Entradas:** **LEFT** (seta para esquerda), **RIGHT** (seta para direita), **HAZ** (pisca-alerta), **CLK** e **RESET**
- **Saídas:** **LA, LB, LC** (lâmpadas do farol esquerdo), **RA, RB e RC** (lâmpadas do farol direito)



## Exemplo 2 – VHDL

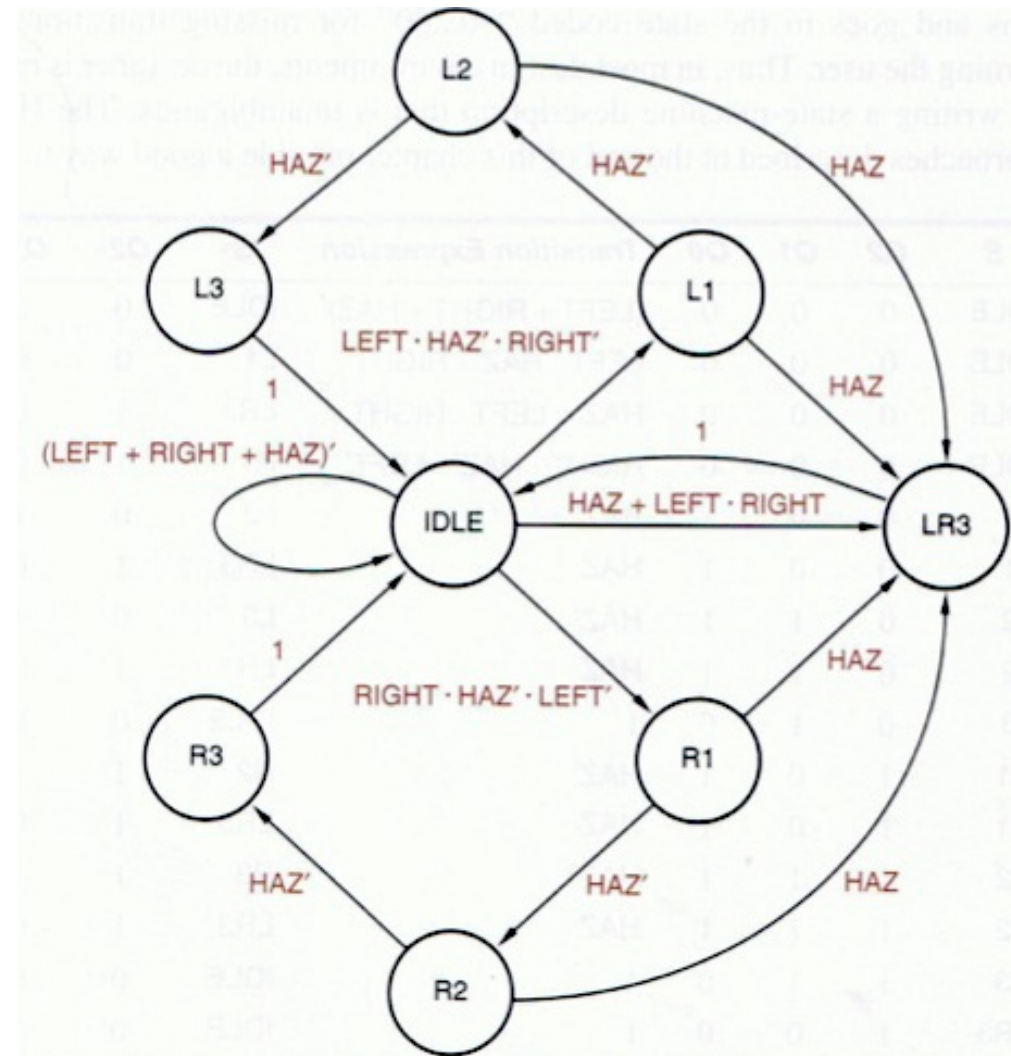
- Sequência de acendimento das lâmpadas e tabela de saídas:



Output Table						
State	LC	LB	LA	RA	RB	RC
IDLE	0	0	0	0	0	0
L1	0	0	1	0	0	0
L2	0	1	1	0	0	0
L3	1	1	1	0	0	0
R1	0	0	0	1	0	0
R2	0	0	0	1	1	0
R3	0	0	0	1	1	1
LR3	1	1	1	1	1	1

## Exemplo 2 – VHDL

- Diagrama de estados
  - IDLE: estado inicial
  - Sequência quando **LEFT** é acionada: IDLE – L1 – L2 – L3 – IDLE
  - Sequência quando **RIGHT** é acionada: : IDLE – R1 – R2 – R3 – IDLE
  - Sequência quando **HAZ** é acionada: : IDLE – LR3 – IDLE
  - Caso **HAZ** seja acionada durante a sequência para a direita ou esquerda, a FSM fica alternando entre os estados IDLE e LR3



# Exemplo 2 – VHDL

---

```
34 entity fsm is
35     Port ( LEFT, RIGHT, HAZ, RST, CLK : in STD_LOGIC;
36           LA, LB, LC, RA, RB, RC : out STD_LOGIC);
37 end fsm;
38
39 architecture Behavioral of fsm is
40
41     type state is (IDLE, L1, L2, L3, R1, R2, R3, LR3);
42     signal current_state, next_state: state;
43
44 begin
45     -- Processo de estado atual
46     process (CLK, RST)
47     begin
48         if rising_edge(CLK) then
49             if RST = '1' then
50                 current_state <= IDLE;
51             else
52                 current_state <= next_state;
53             end if;
54         end if;
55     end process;
```

## Exemplo 2 – VHDL

```
56      -- Processo de próximo estado e saída
57      process (current_state, LEFT, RIGHT, HAZ)
58      begin
59
60          case current_state is
61              when IDLE =>
62                  LA <= '0'; LB <= '0'; LC <= '0';
63                  RA <= '0'; RB <= '0'; RC <= '0';
64                  if LEFT = '1' and RIGHT = '0' and HAZ = '0' then
65                      next_state <= L1;
66                  elsif LEFT = '0' and RIGHT = '1' and HAZ = '0' then
67                      next_state <= R1;
68                  elsif (LEFT = '1' and RIGHT = '1') or HAZ = '1' then
69                      next_state <= LR3;
70                  else
71                      next_state <= IDLE;
72                  end if;
73              when L1 =>
74                  LA <= '1'; LB <= '0'; LC <= '0';
75                  RA <= '0'; RB <= '0'; RC <= '0';
76                  if HAZ = '1' then
77                      next_state <= LR3;
78                  else
79                      next_state <= L2;
80                  end if;
```



# Exemplo 2 – VHDL

```
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
```

```
when L2 =>
    LA <= '1'; LB <= '1'; LC <= '0';
    RA <= '0'; RB <= '0'; RC <= '0';
    if HAZ = '1' then
        next_state <= LR3;
    else
        next_state <= L3;
    end if;
when L3 =>
    LA <= '1'; LB <= '1'; LC <= '1';
    RA <= '0'; RB <= '0'; RC <= '0';
    next_state <= IDLE;

when R1 =>
    LA <= '0'; LB <= '0'; LC <= '0';
    RA <= '1'; RB <= '0'; RC <= '0';
    if HAZ = '1' then
        next_state <= LR3;
    else
        next_state <= R2;
    end if;
```

## Exemplo 2 – VHDL

```
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
```

```
when L2 =>
    LA <= '1'; LB <= '1'; LC <= '0';
    RA <= '0'; RB <= '0'; RC <= '0';
    if HAZ = '1' then
        next_state <= LR3;
    else
        next_state <= L3;
    end if;
when L3 =>
    LA <= '1'; LB <= '1'; LC <= '1';
    RA <= '0'; RB <= '0'; RC <= '0';
    next_state <= IDLE;

when R1 =>
    LA <= '0'; LB <= '0'; LC <= '0';
    RA <= '1'; RB <= '0'; RC <= '0';
    if HAZ = '1' then
        next_state <= LR3;
    else
        next_state <= R2;
    end if;
```

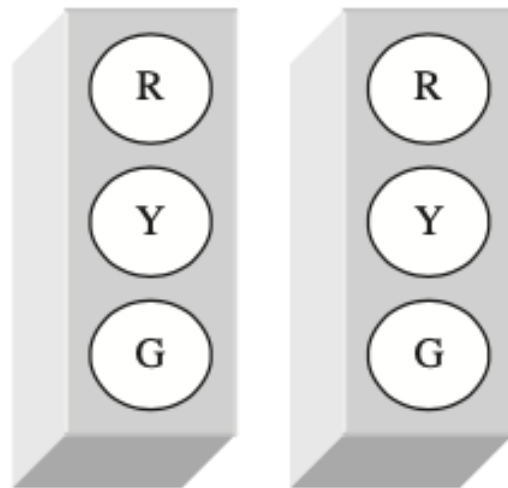
## Exemplo 2 – VHDL

```
102      when R2 =>
103          LA <= '0'; LB <= '0'; LC <= '0';
104          RA <= '1'; RB <= '1'; RC <= '0';
105          if HAZ = '1' then
106              next_state <= LR3;
107          else
108              next_state <= R3;
109          end if;
110      when R3 =>
111          LA <= '0'; LB <= '0'; LC <= '0';
112          RA <= '1'; RB <= '1'; RC <= '1';
113          next_state <= IDLE;
114      when LR3 =>
115          LA <= '1'; LB <= '1'; LC <= '1';
116          RA <= '1'; RB <= '1'; RC <= '1';
117          next_state <= IDLE;
118      end case;
119
120  end process;
121
122
123  end Behavioral;
```



# Exemplo 3 – VHDL

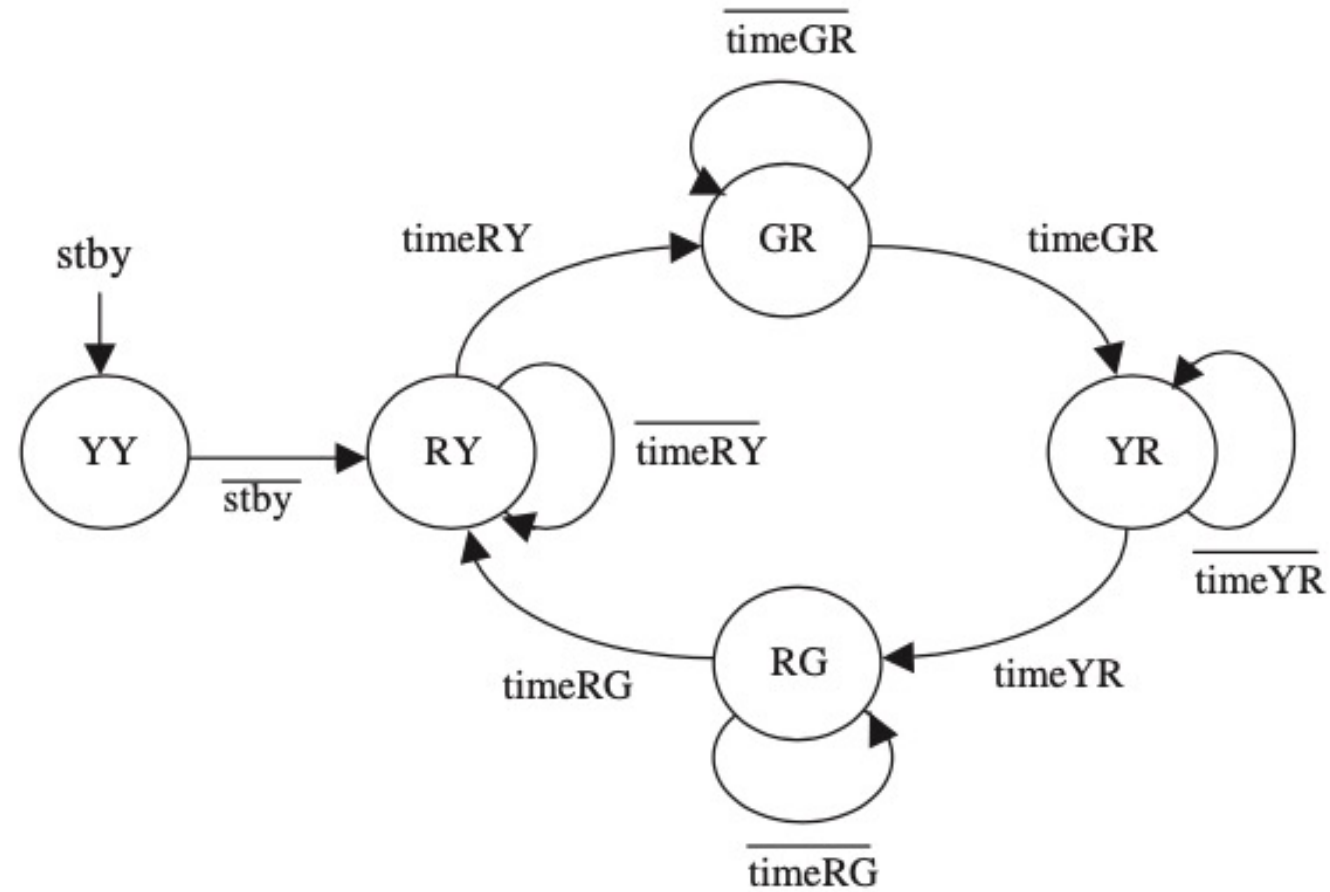
- Controlador de semáforo
- 3 modos de operação:
  - Regular: 4 estados, cada um com um temporizador independente, programável através de uma constante
  - Test: permite reprogramar os timers
  - Standby: sistema ativa as luzes amarelas em ambos os semáforos



State	Operation Mode		
	REGULAR	TEST	STANDBY
	Time	Time	Time
RG	timeRG (30s)	timeTEST (1s)	---
RY	timeRY (5s)	timeTEST (1s)	---
GR	timeGR (45s)	timeTEST (1s)	---
YR	timeYR (5s)	timeTEST (1s)	---
YY	---	---	Indefinite

## Exemplo 3 – VHDL

- Diagrama de estados (modo regular):



# Exemplo 3 – VHDL

---

```
1  -----
2  LIBRARY ieee;
3  USE ieee.std_logic_1164.all;
4  -----
5  ENTITY tlc IS
6      PORT ( clk, stby, test: IN STD_LOGIC;
7              r1, r2, y1, y2, g1, g2: OUT STD_LOGIC);
8  END tlc;
9  -----
10 ARCHITECTURE behavior OF tlc IS
11     CONSTANT timeMAX : INTEGER := 2700;
12     CONSTANT timeRG : INTEGER := 1800;
13     CONSTANT timeRY : INTEGER := 300;
14     CONSTANT timeGR : INTEGER := 2700;
15     CONSTANT timeYR : INTEGER := 300;
16     CONSTANT timeTEST : INTEGER := 60;
17     TYPE state IS (RG, RY, GR, YR, YY);
18     SIGNAL pr_state, nx_state: state;
19     SIGNAL time : INTEGER RANGE 0 TO timeMAX;
```

# Exemplo 3 – VHDL

---

```
20 BEGIN
21     ----- Lower section of state machine: -----
22     PROCESS (clk, stby)
23         VARIABLE count : INTEGER RANGE 0 TO timeMAX;
24     BEGIN
25         IF (stby='1') THEN
26             pr_state <= YY;
27             count := 0;
28         ELSIF (clk'EVENT AND clk='1') THEN
29             count := count + 1;
30             IF (count = time) THEN
31                 pr_state <= nx_state;
32                 count := 0;
33             END IF;
34         END IF;
35     END PROCESS;
```

# Exemplo 3 – VHDL

---

```
36  ----- Upper section of state machine: ----
37  PROCESS (pr_state, test)
38  BEGIN
39      CASE pr_state IS
40          WHEN RG =>
41              r1<='1'; r2<='0'; y1<='0'; y2<='0'; g1<='0'; g2<='1';
42              nx_state <= RY;
43              IF (test='0') THEN time <= timeRG;
44              ELSE time <= timeTEST;
45              END IF;
46          WHEN RY =>
47              r1<='1'; r2<='0'; y1<='0'; y2<='1'; g1<='0'; g2<='0';
48              nx_state <= GR;
49              IF (test='0') THEN time <= timeRY;
50              ELSE time <= timeTEST;
51              END IF;
```

# Exemplo 3 – VHDL

---

```
52         WHEN GR =>
53             r1<='0'; r2<='1'; y1<='0'; y2<='0'; g1<='1'; g2<='0';
54             nx_state <= YR;
55             IF (test='0') THEN time <= timeGR;
56             ELSE time <= timeTEST;
57             END IF;
58         WHEN YR =>
59             r1<='0'; r2<='1'; y1<='1'; y2<='0'; g1<='0'; g2<='0';
60             nx_state <= RG;
61             IF (test='0') THEN time <= timeYR;
62             ELSE time <= timeTEST;
63             END IF;
64         WHEN YY =>
65             r1<='0'; r2<='0'; y1<='1'; y2<='1'; g1<='0'; g2<='0';
66             nx_state <= RY;
67     END CASE;
68 END PROCESS;
69 END behavior;
70 -----
```

---

Página em branco.

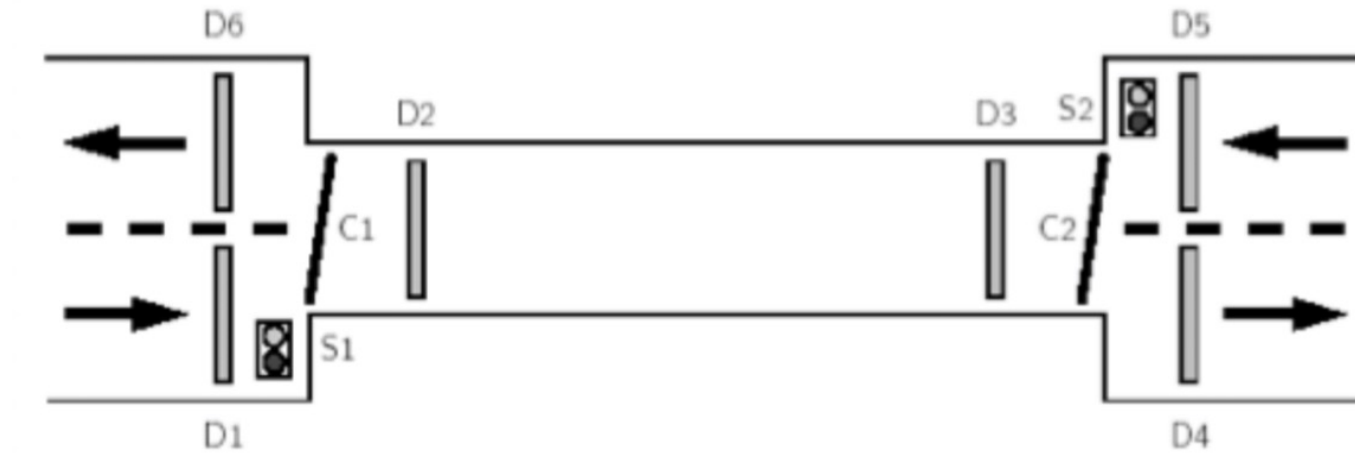
# Aula 23

## Exercícios



# Exercício 1

- Considere uma estrada estreita de montanha, na qual só é possível passar um carro de cada vez, conforme a mostra a figura abaixo:



- Existem duas cancelas (saídas **C1** e **C2**) nas duas extremidades da estrada, seis de-tectores de presença (entradas **D1** a **D6**) e dois semáforos (**S1** e **S2**)

# Exercício 1

---

- Considere que:
  - os detectores recebem o valor 1 quando uma viatura passa por eles;
  - as cancelas **C1** e **C2** são abertas quando recebem o valor 1;
  - os semáforos **S1** e **S2** acendem a luz verde quando recebem o valor 1, e a luz vermelha, quando recebem o valor 0;
  - em situações normais (ou seja, quando não há passagem de viaturas), as cancelas ficam fechadas e os semáforos ficam em vermelho.
- Como exemplo, considere que surge uma viatura no lado esquerdo da estrada. Se não houver nenhum carro deslocando-se na estrada estreita, o semáforo **S1** fica verde, a cancela **C1** abre e a viatura entra na estrada estreita. Assim que a viatura passa no detector **D2**, a cancela **C1** é fechada, o semáforo volta a ficar vermelho e a situação fica estável nesse estado até a viatura sair da estrada estreita. Quando a viatura passar pelo detector **D3**, a cancela **C2** abre e permanece aberta até a viatura passar pelo detector **D4**. Então, a cancela **C2** fecha

# Exercício 1

---

- Se, entretanto, chegar uma segunda viatura em qualquer dos lados enquanto a primeira viatura estiver na estrada estreita, a segunda deve esperar a primeira sair e então o processo é iniciado no mesmo sentido, ou no sentido inverso, conforme o sentido de chegada da segunda viatura. Caso cheguem duas viaturas ao mesmo tempo, a prioridade é do sentido da esquerda para a direita
- Projete uma máquina de estados de **Mealy** para um controlador de acesso a essa estrada estreita considerando as seguintes alterações:
  - no estado em que a cancela **C1** é aberta, o semáforo **S1** fica verde apenas até o veículo abandonar o detector **D1**
  - assim que isso acontecer, o semáforo **S1** volta a ficar vermelho para impedir que um segundo veículo possa seguir o primeiro
  - o mesmo acontece no estado em que a cancela **C2** é aberta, com a saída **S2** e a entrada **D5**

## Exercício 2

---

- Projete uma FSM que gera sinal PWM com ciclo de trabalho programável
- **Entradas:** *duty\_cycle (0–255), clk, reset*
- **Saída:** *pwm\_out*

## Exercício 3

---

- Projete uma FSM que calcula a média aritmética de quatro valores de 8 bits
- **Entradas:** *clk, reset, start, data\_in (8 bits), valid*
- **Saída:** *media (8 bits), done*
- **Estados:** IDLE, READ\_1, READ\_2, READ\_3, READ\_4, CALC, DONE

---

Página em branco.