

RELATÓRIO- PROJETO 8

Questão 1

Arquivo Contador10.vhd:

```
11 -----
12 library IEEE;
13 use IEEE.std_logic_1164.ALL;
14 -----
15 entity contador10 is port(
16     clock, reset, low_enable, low_RCI, load :in std_logic;
17     D :in std_logic_vector(3 DOWNTO 0);
18     Q :out std_logic_vector(3 DOWNTO 0);
19     low_RCO :out std_logic;
20 end contador10;
21 -----
22
23 architecture contador10_arch of contador10 is
24     type estado is (ST0, ST1, ST2, ST3, ST4, ST5, ST6, ST7, ST8, ST9);
25     signal currentState, nextState, loadState : estado;
26
27 begin
28     with D select
29         loadState <= ST0 when "0000",
30                    ST1 when "0001",
31                    ST2 when "0010",
32                    ST3 when "0011",
33                    ST4 when "0100",
34                    ST5 when "0101",
35                    ST6 when "0110",
36                    ST7 when "0111",
37                    ST8 when "1000",
38                    ST9 when "1001",
39                    ST0 when others;
40
41     sync_proc: process (clock)
42     begin
43         if rising_edge(clock) then
44             currentState <= nextState;
45         end if;
46     end process sync_proc;
47
48     comb_proc: process (currentState, reset, low_enable, low_RCI, load, loadState)
49     begin
50         case currentState is
51             when ST0 =>
52                 Q <= "0000";
53                 low_RCO <= '1';
54                 if (reset = '1') then nextState <= ST0;
55                 elsif (load = '1') then nextState <= loadState;
56                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST1;
57                 else nextState <= ST0;
58             end if;
59
60             when ST1 =>
61                 Q <= "0001";
62                 low_RCO <= '1';
63                 if (reset = '1') then nextState <= ST0;
64                 elsif (load = '1') then nextState <= loadState;
65                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST2;
66                 else nextState <= ST1;
67             end if;
68
69             when ST2 =>
70                 Q <= "0010";
71                 low_RCO <= '1';
72                 if (reset = '1') then nextState <= ST0;
73                 elsif (load = '1') then nextState <= loadState;
74                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST3;
75                 else nextState <= ST2;
76             end if;
77
78             when ST3 =>
79                 Q <= "0011";
80                 low_RCO <= '1';
81                 if (reset = '1') then nextState <= ST0;
82                 elsif (load = '1') then nextState <= loadState;
83                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST4;
84                 else nextState <= ST3;
85             end if;
86
87             when ST4 =>
88                 Q <= "0100";
89                 low_RCO <= '1';
90                 if (reset = '1') then nextState <= ST0;
91                 elsif (load = '1') then nextState <= loadState;
92                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST5;
93                 else nextState <= ST4;
94             end if;
95
96             when ST5 =>
97                 Q <= "0101";
98                 low_RCO <= '1';
99                 if (reset = '1') then nextState <= ST0;
100                elsif (load = '1') then nextState <= loadState;
101                elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST6;
102                else nextState <= ST5;
103            end if;
104
105             when ST6 =>
106                 Q <= "0110";
107                 low_RCO <= '1';
108                 if (reset = '1') then nextState <= ST0;
109                 elsif (load = '1') then nextState <= loadState;
110                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST7;
111                 else nextState <= ST6;
112             end if;
113
114             when ST7 =>
115                 Q <= "0111";
116                 low_RCO <= '1';
117                 if (reset = '1') then nextState <= ST0;
118                 elsif (load = '1') then nextState <= loadState;
119                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST8;
120                 else nextState <= ST7;
121             end if;
122
123             when ST8 =>
124                 Q <= "1000";
125                 low_RCO <= '1';
126                 if (reset = '1') then nextState <= ST0;
127                 elsif (load = '1') then nextState <= loadState;
128                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST9;
129                 else nextState <= ST8;
130             end if;
131
132             when ST9 =>
133                 Q <= "1001";
134                 low_RCO <= '0';
135                 if (reset = '1') then nextState <= ST0;
136                 elsif (load = '1') then nextState <= loadState;
137                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST0;
138                 else nextState <= ST9;
139             end if;
140
141             when others =>
142                 Q <= "0000";
143                 low_RCO <= '1';
144                 if (reset = '1') then nextState <= ST0;
145                 elsif (load = '1') then nextState <= loadState;
146                 elsif ((low_enable = '0') and (low_RCI = '0')) then nextState <= ST1;
147                 else nextState <= ST0;
148             end if;
149         end case;
150     end process comb_proc;
151 end contador10_arch;
```

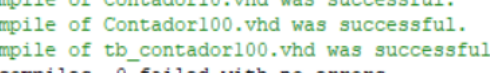
Arquivo Contador100.vhd:

```
1  -- *****
2  -- Circuito: Contador módulo 100 construído
3  -- com 2 contadores módulo 10 cascadeados
4  --      CLOCK, RESET,
5  --      LOW_ENABLE, LOAD - Entradas
6  --      dezload, unload - Vetores de Entrada
7  --      da dezena e da unidade
8  --      cont_dezena, cont_unidade - Vetores
9  --      de saída(dígitos da contagem)
10 --      da dezena e da unidade
11 -- *****
12 -----
13 library IEEE;
14 use IEEE.std_logic_1164.ALL;
15 -----
16 entity contador100 is port(
17     CLOCK, RESET, LOW_ENABLE, LOAD :in std_logic;
18     dezload, unload :in std_logic_vector(3 DOWNTO 0);
19     cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0));
20 end contador100;
21 -----
22 architecture ex1 of contador100 is
23     component contador10 is port(
24         clock, reset, low_enable, low_RCI, load :in std_logic;
25         D :in std_logic_vector(3 DOWNTO 0);
26         Q :out std_logic_vector(3 DOWNTO 0);
27         low_RCO :out std_logic);
28     end component;
29
30     signal rci_aux : std_logic;
31 begin
32     contU: contador10 PORT MAP (clock => CLOCK, reset => RESET, low_enable => LOW_ENABLE, low_RCI => '0',
33                               load => LOAD, D => unload, Q => cont_unidade, low_RCO => rci_aux);
34     contD: contador10 PORT MAP (clock => CLOCK, reset => RESET, low_enable => LOW_ENABLE, low_RCI => rci_aux,
35                               load => LOAD, D => dezload, Q => cont_dezena, low_RCO => open);
36 end ex1;
```

Arquivo tb_contador100.vhd:

```
12 -- *****
13 LIBRARY ieee;
14 USE ieee.std_logic_1164.ALL;
15 USE std.textio.ALL;
16 USE ieee.numeric_std.ALL;
17 USE ieee.std_logic_signed.ALL;
18
19 ENTITY testbench_contador100 IS END;
20
21 architecture tb_contador100 of testbench_contador100 is
22     component contador100 is port(
23         CLOCK, RESET, LOW_ENABLE, LOAD :in std_logic;
24         dezload, unload :in std_logic_vector(3 DOWNTO 0);
25         cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0));
26     end component;
27
28     signal clk : std_logic := '0';
29     signal rst, en, ld : std_logic;
30     signal un, dz : std_logic_vector(3 DOWNTO 0);
31
32 begin
33     cont: contador100 PORT MAP (CLOCK => clk, RESET => rst, LOW_ENABLE => en, LOAD => ld, unload => un, dezload => dz, cont_dezena => open, cont_unidade => open);
34
35     clk <= not clk after 5 ns;
36
37     estimulo: PROCESS
38     begin
39         rst <= '0'; en <= '0'; ld <= '0'; un <= "0111"; dz <= "0101"; wait for 10 ns;
40         ld <= '1'; wait for 10 ns; -- Carregando
41         ld <= '0'; un <= "0000"; dz <= "0000"; rst <= '1'; wait for 10 ns; -- Resetando
42         rst <= '0';
43
44         for i in 0 to 8 loop
45             un <= "0000";
46             for j in 0 to 8 loop
47                 un <= un + 1;
48                 wait for 10 ns;
49             end loop;
50             if (i = 5) then en <= '1'; -- Desativando o enable em um
51             else en <= '0'; -- intervalo arbitrário para
52             end if; -- mostrar que a contagem parou
53             un <= "0000";
54             dz <= dz + 1;
55             wait for 10 ns;
56         end loop;
57     end PROCESS estimulo;
58 end tb_contador100;
```

Compilação:

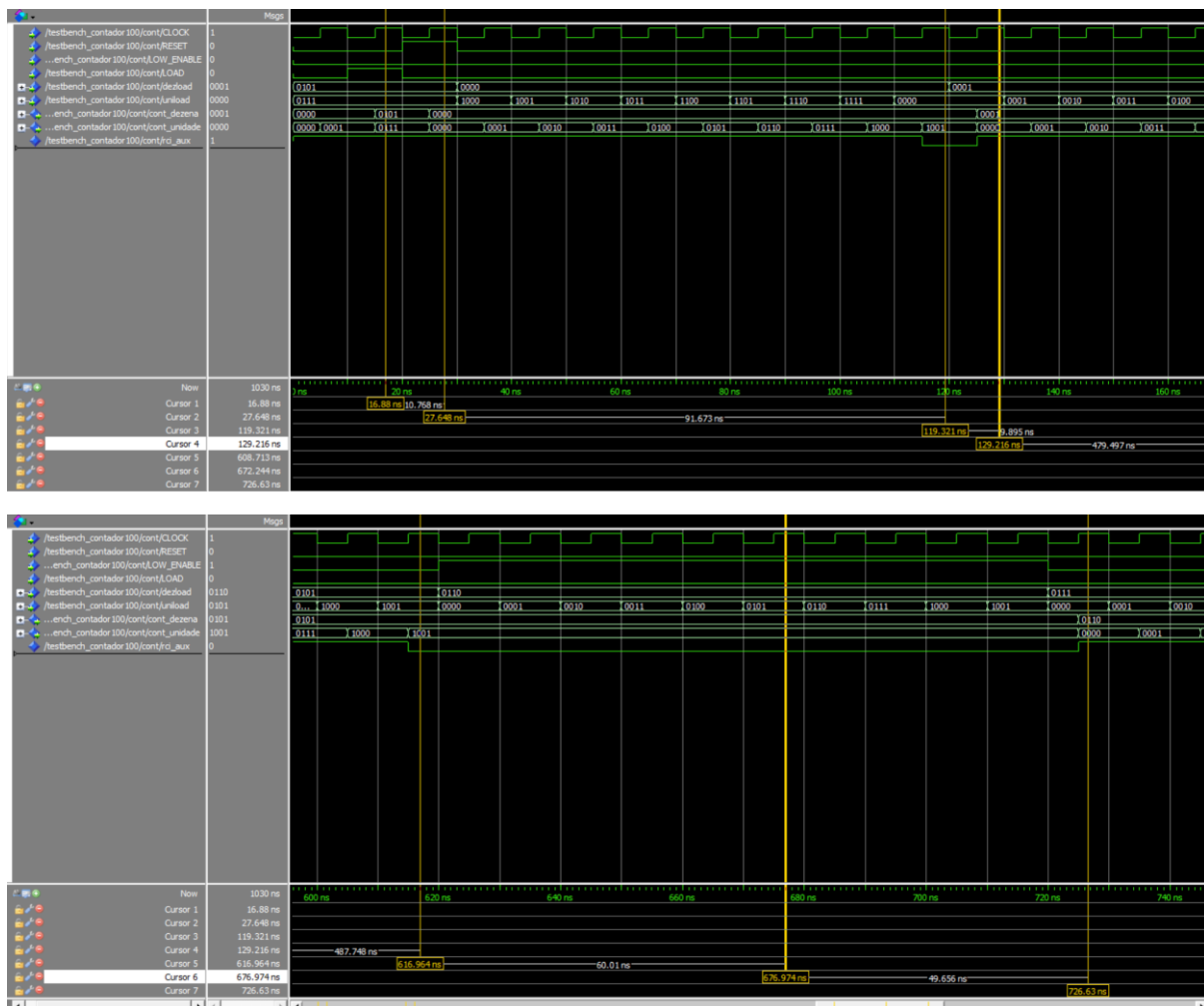


The screenshot shows the Quartus II IDE interface. The top window is the 'Project' window, displaying a list of files in the project hierarchy. The files are 'Contador10.vhd', 'Contador100.vhd', and 'tb_contador100.v...'. Each file has a status icon (a green checkmark) and a type of 'VHDL'. The 'Modified' column shows the date and time of the last modification: '04/23/2021 03:50:26 pm' for 'Contador10.vhd', '04/23/2021 04:04:22 pm' for 'Contador100.vhd', and '04/23/2021 04:05:40 pm' for 'tb_contador100.v...'. Below the Project window is the 'Transcript' window, which displays the output of the compilation process. The transcript shows three successful compilation messages for the three VHD files, followed by a summary line: '# 3 compiles, 0 failed with no errors.'

Name	Status	Type	Orig	Modified
Contador10.vhd	✓	VHDL	0	04/23/2021 03:50:26 pm
Contador100.vhd	✓	VHDL	1	04/23/2021 04:04:22 pm
tb_contador100.v...	✓	VHDL	2	04/23/2021 04:05:40 pm

```
# Compile of Contador10.vhd was successful.
# Compile of Contador100.vhd was successful.
# Compile of tb_contador100.vhd was successful.
# 3 compiles, 0 failed with no errors.
```

Simulação (não mostrei toda a simulação, apenas os pontos mais relevantes):



	CLOCK	RESET	ENABLE	LOAD	dezload	unload	cont_dez	cont_un
Cursor 1	SUBIDA	0	0	1	0101	0111	0101	0111
Cursor 2	SUBIDA	1	0	0	0101	0111	0000	0000
Cursor 3	SUBIDA	0	0	0	0000	0000	0000	1001
Cursor 4	SUBIDA	0	0	0	0001	0000	0001	0000
Cursor 5	SUBIDA	0	0	0	0101	1001	0101	1001
Cursor 6	SUBIDA	0	1	0	0110	0101	0101	1001
Cursor 7	SUBIDA	0	0	0	0111	0000	0110	0000

Questão 2

Arquivo time_flags.vhd:

```

1  -- *****
2  --     Circuito: Sistema de temporização com
3  --     flags indicando quando o tempo atinge 5,
4  --     6, 20 e 60 segundos. Construido
5  --     utilizando um contador módulo 100
6  --     clock, reset - Entradas
7  --     T5, T6, T20, T60 - Saídas
8  -- *****
9  library IEEE;
10 use IEEE.std_logic_1164.ALL;
11 -----
12 entity timeFlags is port(
13     clock, reset :in std_logic;
14     T5, T6, T20, T60 :out std_logic);
15 end timeFlags;
16 -----
17
18 architecture timeFlags_arch of timeFlags is
19     component contador100 is port(
20         CLOCK, RESET, LOW_ENABLE, LOAD :in std_logic;
21         dezload, unload :in std_logic_vector(3 DOWNTO 0);
22         cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0));
23     end component;
24
25     signal dz_aux, un_aux : std_logic_vector(3 DOWNTO 0);
26     signal A : std_logic_vector(7 DOWNTO 0);
27 begin
28     tflag: contador100 PORT MAP (CLOCK => clock, RESET => reset, LOW_ENABLE => '0',
29         LOAD => '0', dezload => "0000", unload => "0000",
30         cont_dezena => dz_aux, cont_unidade => un_aux);
31
32     A <= dz_aux & un_aux;
33     T5 <= '1' when (A >= x"05") else '0';
34     T6 <= '1' when (A >= x"06") else '0';
35     T20 <= '1' when (A >= x"20") else '0';
36     T60 <= '1' when (A >= x"60") else '0';
37 end timeFlags_arch;

```

Arquivo tb_time_flags.vhd:

```
2  --      Testbench para simulacao Funcional do
3  --      Circuito: Sistema de temporização com
4  --      flags indicando quando o tempo atinge 5,
5  --      6, 20 e 60 segundos. Construido
6  --      utilizando um contador módulo 100
7  --      clock, reset - Entradas
8  --      T5, T6, T20, T60 - Saídas
9  --      *****
10 LIBRARY ieee;
11 USE ieee.std_logic_1164.ALL;
12 USE std.textio.ALL;
13 USE ieee.numeric_std.ALL;
14 USE ieee.std_logic_signed.ALL;
15
16 ENTITY testbench_time_flags IS END;
17
18 architecture tb_time_flags of testbench_time_flags is
19     component timeFlags is port(
20         clock, reset :in std_logic;
21         T5, T6, T20, T60 :out std_logic);
22     end component;
23
24     signal clk : std_logic := '0';
25     signal rst : std_logic;
26
27 begin
28     tf: timeFlags PORT MAP (clock => clk, reset => rst, T5 => open,
29         T6 => open, T20 => open, T60 => open);
30
31     clk <= not clk after 0.5 sec;
32
33     estimulo: PROCESS
34     begin
35         rst <= '0'; wait for 80 sec;
36         rst <= '1'; wait;
37     end PROCESS estimulo;
38 end tb_time_flags;
```

Compilação:

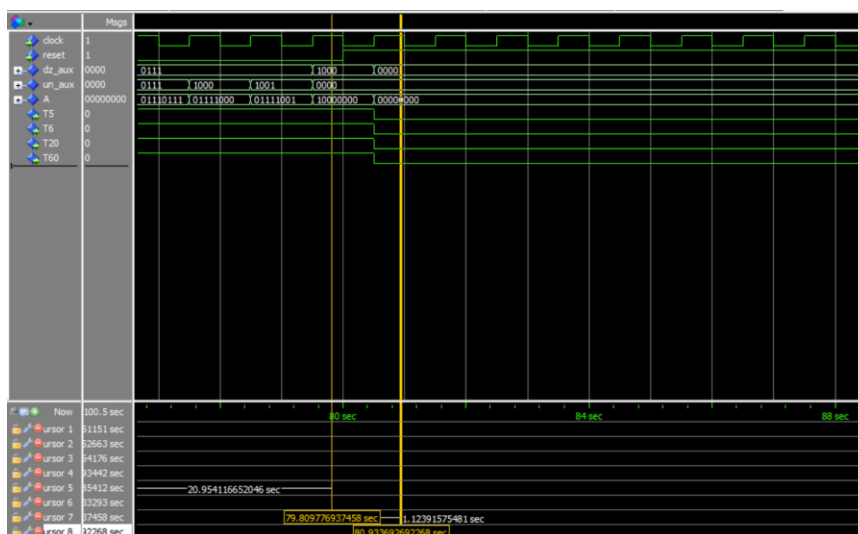
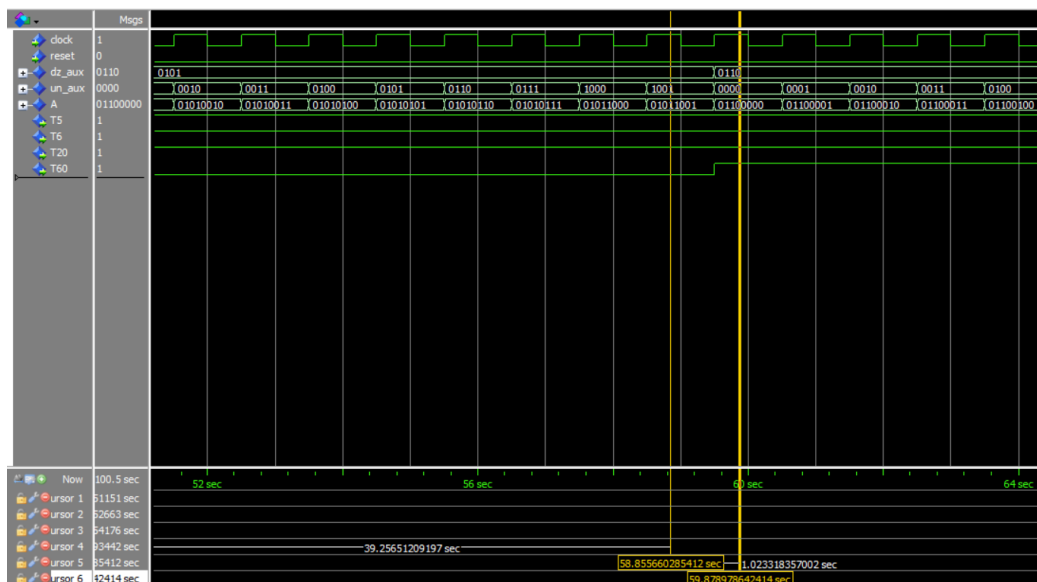
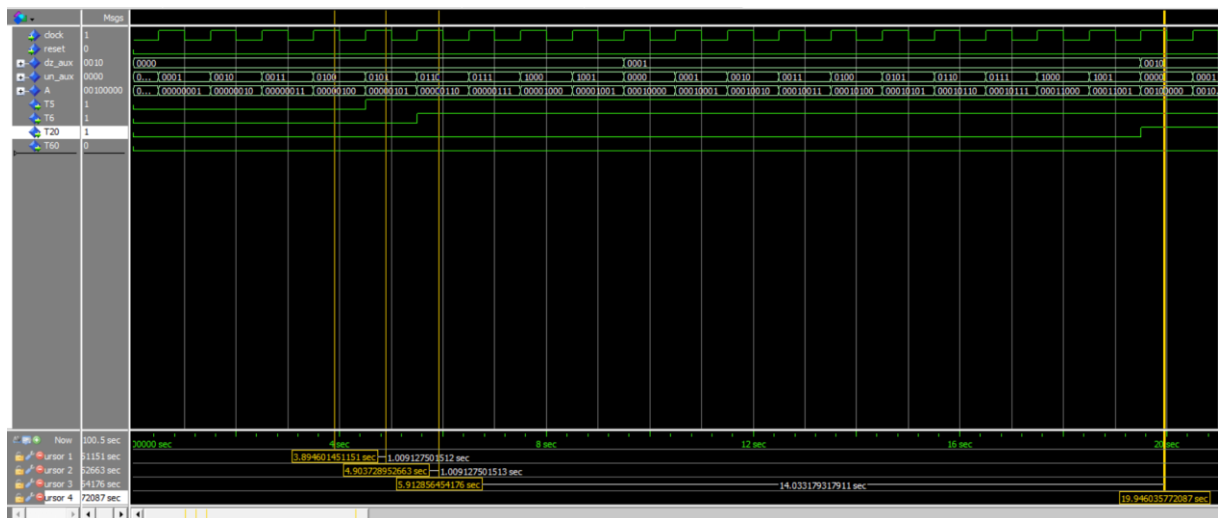
Name	Status	Type	Orig	Modified
tb_time_flags.vhd	✓	VHDL	0	04/23/2021 10:06:58 pm
time_flags.vhd	✓	VHDL	1	04/23/2021 10:05:29 pm
Contador10.vhd	✓	VHDL	2	04/23/2021 03:50:26 pm
Contador100.vhd	✓	VHDL	3	04/23/2021 04:04:22 pm

Library x Project x sim x

Transcript

```
# Compile of tb_time_flags.vhd was successful.
# Compile of time_flags.vhd was successful.
# Compile of Contador10.vhd was successful.
# Compile of Contador100.vhd was successful.
# 4 compiles, 0 failed with no errors.
```

Simulação:



	clock	reset	dezena	unidade	T5	T6	T20	T60
Cursor 1	SUBIDA	0	0000 (0)	0100 (4)	0	0	0	0
Cursor 2	SUBIDA	0	0000 (0)	0101 (5)	1	0	0	0
Cursor 3	SUBIDA	0	0000 (0)	0110 (6)	1	1	0	0
Cursor 4	SUBIDA	0	0010 (2)	0000 (0)	1	1	1	0
Cursor 5	SUBIDA	0	0101 (5)	1001 (9)	1	1	1	0
Cursor 6	SUBIDA	0	0110 (6)	0000 (0)	1	1	1	1
Cursor 7	SUBIDA	0	1000 (8)	0000 (0)	1	1	1	1
Cursor 8	SUBIDA	1	0000 (0)	0000 (0)	0	0	0	0

Questão 3

Arquivo time_flags_v2.vhd (modificado em relação à questão anterior para tirar o atraso de 1s):

```

1  -- *****
2  --     Circuito: Sistema de temporização com
3  --     flags indicando quando o tempo atinge 5,
4  --     6, 20 e 60 segundos. Construído
5  --     utilizando um contador módulo 100
6  --     clock, reset - Entradas
7  --     T5, T6, T20, T60 - Saídas
8  -- *****
9  library IEEE;
10 use IEEE.std_logic_1164.ALL;
11 -----
12 entity timeFlags is port(
13     clock, reset :in std_logic;
14     cont_dezena, cont_unidade :out std_logic_vector(3 downto 0);
15     T5, T6, T20, T60 :out std_logic);
16 end timeFlags;
17 -----
18
19 architecture timeFlags_arch of timeFlags is
20     component contador100 is port(
21         CLOCK, RESET, LOW_ENABLE, LOAD :in std_logic;
22         dezload, unload :in std_logic_vector(3 DOWNTO 0);
23         cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0));
24     end component;
25
26     signal dz_aux, un_aux : std_logic_vector(3 DOWNTO 0);
27     signal A : std_logic_vector(7 DOWNTO 0);
28 begin
29     tflag: contador100 PORT MAP (CLOCK => clock, RESET => reset, LOW_ENABLE => '0',
30                                LOAD => '0', dezload => "0000", unload => "0000",
31                                cont_dezena => dz_aux, cont_unidade => un_aux);
32     cont_dezena <= dz_aux; cont_unidade <= un_aux;
33
34     A <= dz_aux & un_aux;
35     T5 <= '1' when (A >= x"04") else '0';
36     T6 <= '1' when (A >= x"05") else '0';
37     T20 <= '1' when (A >= x"19") else '0';
38     T60 <= '1' when (A >= x"59") else '0';
39 end timeFlags_arch;

```

Arquivo maquestados.vhd

```
1  -- *****
2  --      Circuito: Máquina de estados feita para
3  --      implementar um cruzamento com dois sinais de trânsito
4  --      construído utilizando um componente timeFlags que
5  --      indica quando passaram 5, 6, 20 e 60 segundos
6  --      sensorA, sensorB - Entradas que indicam
7  --      se está passando carro ou não em cada sinal
8  --      chave - Entrada que indica se os
9  --      semáforos devem estar ativos ou não
10 --      clock - Sinal de clock
11 --      semaforoA, semaforoB - Saídas de 3 bits
12 --      indicando o estado atual dos semáforos, sendo
13 --      100, 010 e 001 para vermelho, amarelo e verde,
14 --      respectivamente
15 --      cont_dezena, cont_unidade - Saída
16 --      indicando a contagem do tempo
17 --      resetcounter - Saída indicando quando
18 --      a contagem do tempo foi resetada
19 -- *****
20
21 library IEEE;
22 use IEEE.std_logic_1164.ALL;
23
24 entity maquestados is port(
25     sensorA, sensorB, chave, clock :in std_logic;
26     semaforoA, semaforoB :out std_logic_vector(2 DOWNTO 0);
27     cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0);
28     resetcounter :out std_logic);
29 end maquestados;
30
31
32 architecture maquestados_arch of maquestados is
33     component timeFlags is port(
34         clock, reset :in std_logic;
35         T5, T6, T20, T60 :out std_logic;
36         cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0));
37     end component;
38
39     type estado is (RR1, RG, RY, RR2, GR, YR, YY, NN);
40     signal currentState, nextState : estado;
41     signal rst, T5_signal, T6_signal, T20_signal, T60_signal : std_logic;
42
43     type estado is (RR1, RG, RY, RR2, GR, YR, YY, NN);
44     signal currentState, nextState : estado;
45     signal rst, T5_signal, T6_signal, T20_signal, T60_signal : std_logic;
46
47 begin
48     tflag: timeFlags PORT MAP (clock => clock, reset => rst, cont_dezena => cont_dezena, cont_unidade => cont_unidade,
49                               T5 => T5_signal, T6 => T6_signal, T20 => T20_signal, T60 => T60_signal);
50
51     resetcounter <= rst;
52
53     sync_proc: process(clock)
54     begin
55         if rising_edge(clock) then
56             currentState <= nextState;
57         end if;
58     end process sync_proc;
59
60     comb_proc: process(currentState, T5_signal, T6_signal, T20_signal, T60_signal, sensorA, sensorB, chave)
61     begin
62         case currentState is
63             when RR1 =>
64                 semaforoA <= "100"; semaforoB <= "100";
65                 if chave = '0' then nextState <= YY; rst <= '1';
66                 elsif (T5_signal = '1') then nextState <= RG; rst <= '1';
67                 else nextState <= RR1; rst <= '0';
68                 end if;
69             when RG =>
70                 semaforoA <= "100"; semaforoB <= "001";
71                 if chave = '0' then nextState <= YY; rst <= '1';
72                 elsif (T20_signal = '1' and sensorA = '1' and sensorB = '0') or (T60_signal = '1') then
73                     nextState <= RY; rst <= '1';
74                 else
75                     nextState <= RG; rst <= '0';
76                 end if;
77             when RY =>
78                 semaforoA <= "100"; semaforoB <= "010";
79                 if chave = '0' then nextState <= YY; rst <= '1';
80                 elsif (T6_signal = '1') then nextState <= RR2; rst <= '1';
81                 else nextState <= RY; rst <= '0';
82                 end if;
```



```

81         when RR2 =>
82             semaforoA <= "100"; semaforoB <= "100";
83             if chave = '0' then nextState <= YY; rst <= '1';
84             elsif (T5_signal = '1') then nextState <= GR; rst <= '1';
85             else nextState <= RR2; rst <= '0';
86             end if;
87
88         when GR =>
89             semaforoA <= "001"; semaforoB <= "100";
90             if chave = '0' then nextState <= YY; rst <= '1';
91             elsif (T20_signal = '1' and sensorA = '0' and sensorB = '1') or (T60_signal = '1') then
92                 nextState <= YR; rst <= '1';
93             else
94                 nextState <= GR; rst <= '0';
95             end if;
96
97         when YR =>
98             semaforoA <= "010"; semaforoB <= "100";
99             if chave = '0' then nextState <= YY; rst <= '1';
100             elsif (T6_signal = '1') then nextState <= RR1; rst <= '1';
101             else nextState <= YR; rst <= '0';
102             end if;
103
104         when YY =>
105             if chave = '1' then nextState <= GR; rst <= '1';
106             else
107                 semaforoA <= "010"; semaforoB <= "010";
108                 nextState <= NN; rst <= '1';
109             end if;
110
111         when NN =>
112             if chave = '1' then nextState <= GR; rst <= '1';
113             else
114                 semaforoA <= "000"; semaforoB <= "000";
115                 nextState <= YY; rst <= '1';
116             end if;
117
118     end case;
119 end process comb_proc;
120 end architecture maqestados_arch;
121

```

Arquivo cruzamento.vhd:

```

1  -- *****
2  -- Circuito: Cruzamento com dois sinais de trânsito,
3  -- construido utilizando uma máquina de estados com
4  -- os estados possíveis combinados desses sinais
5  -- sensorA, sensorB - Entradas que indicam
6  -- se está passando carro ou não em cada sinal
7  -- chave - Entrada que indica se os
8  -- semáforos devem estar ativos ou não
9  -- clock - Sinal de clock
10 -- semaforoA, semaforoB - Saídas de 3 bits
11 -- indicando o estado atual dos semáforos, sendo
12 -- 100, 010 e 001 para vermelho, amarelo e verde,
13 -- respectivamente
14 -- cont_dezena, cont_unidade - Saída
15 -- indicando a contagem do tempo
16 -- *****
17 library IEEE;
18 use IEEE.std_logic_1164.ALL;
19
20 entity cruzamento is port(
21     sensorA, sensorB, chave, clock :in std_logic;
22     semaforoA, semaforoB :out std_logic_vector(2 DOWNTO 0);
23     cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0));
24 end cruzamento;
25
26 architecture cruzamento_arch of cruzamento is
27     component maqestados is port(
28         sensorA, sensorB, chave, clock :in std_logic;
29         semaforoA, semaforoB :out std_logic_vector(2 DOWNTO 0);
30         cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0);
31         resetcounter :out std_logic);
32     end component;
33
34     signal rst : std_logic;
35
36 begin
37
38     mquest: maqestados PORT MAP (sensorA, sensorB, chave, clock, semaforoA,
39                                 semaforoB, cont_dezena, cont_unidade, rst);
40
41 end cruzamento_arch;

```

Arquivo tb_cruzamento.vhd:

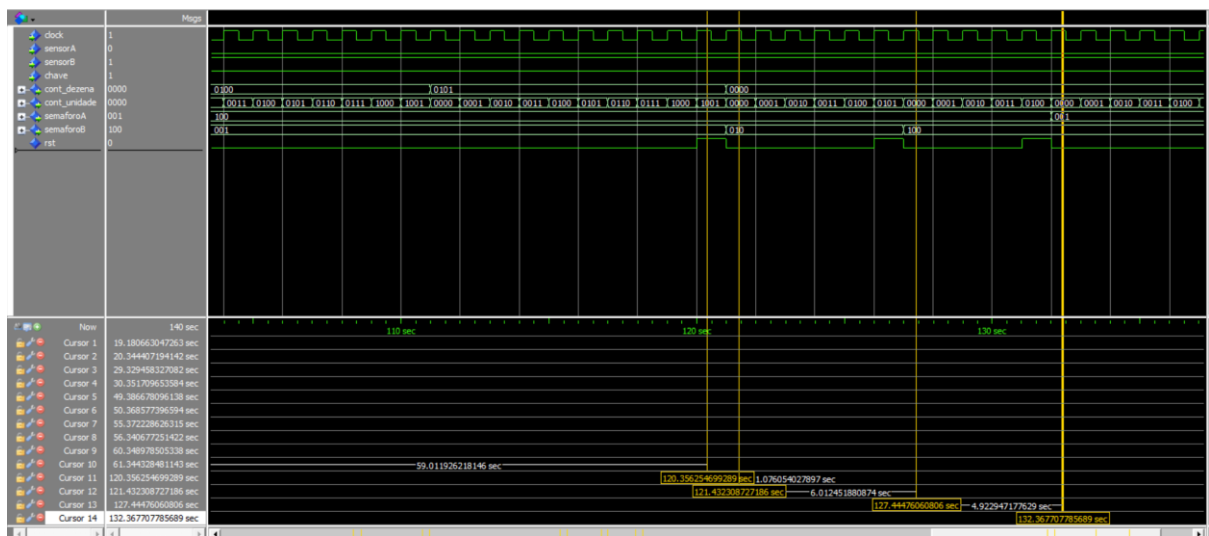
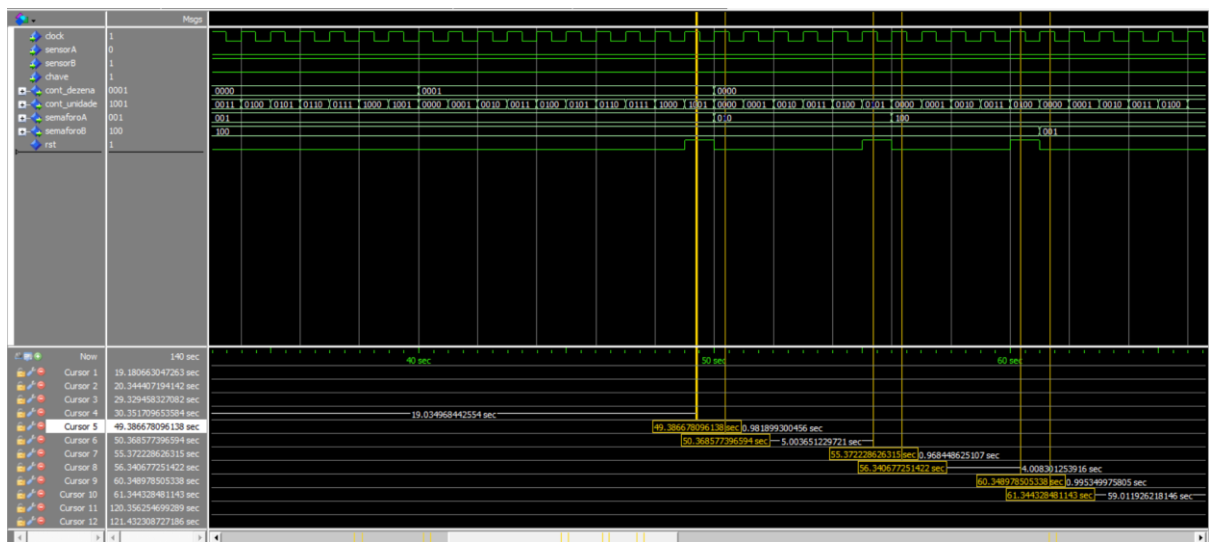
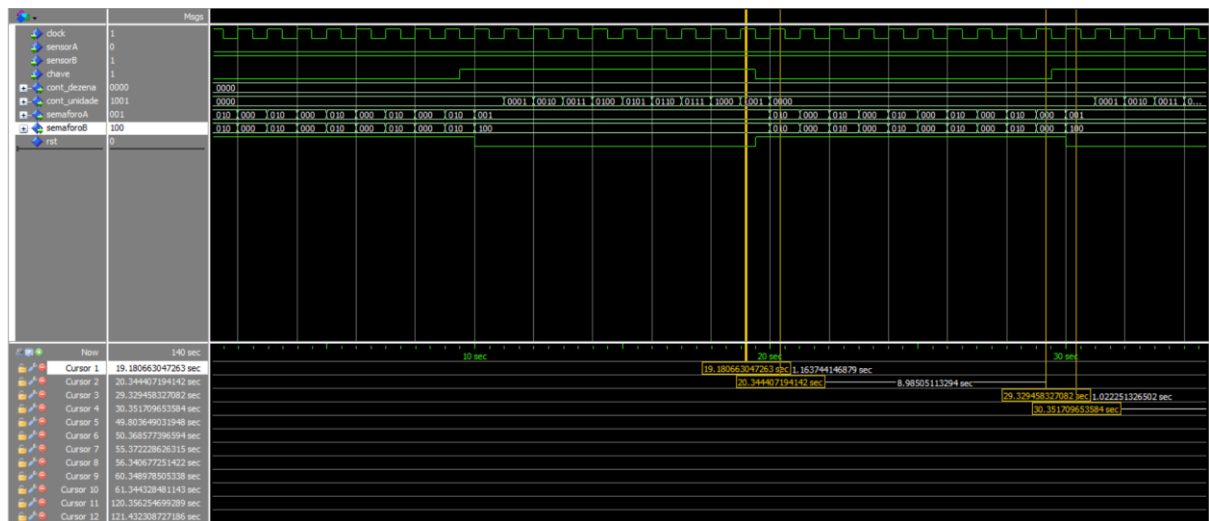
```
16  -- *****
17  LIBRARY ieee;
18  USE ieee.std_logic_1164.ALL;
19  USE std.textio.ALL;
20  USE ieee.numeric_std.ALL;
21  USE ieee.std_logic_signed.ALL;
22
23  ENTITY testbench_cruzamento IS END;
24
25  architecture tb_cruzamento of testbench_cruzamento is
26      component cruzamento is port(
27          sensorA, sensorB, chave, clock :in std_logic;
28          semaforoA, semaforoB :out std_logic_vector(2 DOWNTO 0);
29          cont_dezena, cont_unidade :out std_logic_vector(3 DOWNTO 0));
30      end component;
31
32      signal clk : std_logic := '1';
33      signal key, sensor_A, sensor_B : std_logic;
34
35  begin
36      cruz: cruzamento PORT MAP (clock => clk, chave => key, sensorA => sensor_A,
37          sensorB => sensor_B, semaforoA => open, semaforoB => open,
38          cont_dezena => open, cont_unidade => open);
39
40      clk <= not clk after 0.5 sec;
41
42      estimulo: PROCESS
43      begin
44          key <= '0'; sensor_A <= '0'; sensor_B <= '1'; wait for 9.5 sec; -- Setando o sensor B para 1 e o A para 0
45          key <= '1'; wait for 10 sec; -- para mostrar tanto a mudança com 20s e com 60s.
46          key <= '0'; wait for 10 sec; -- Ligando os semáforos.
47          key <= '1'; wait; -- Desligando os semáforos temporariamente.
48          -- Religando os semáforos.
49      end PROCESS estimulo;
50  end tb_cruzamento;
```

Compilação:

Name	Status	Type	Order	Modified
tb_cruzamento.vh...	✓	VHDL	4	04/30/2021 03:05:25 pm
time_flags_v2.vhd	✓	VHDL	5	04/30/2021 01:24:43 pm
maquestados.vhd	✓	VHDL	3	04/30/2021 02:56:22 pm
Contador100.vhd	✓	VHDL	1	04/30/2021 08:25:54 am
Contador10.vhd	✓	VHDL	0	04/30/2021 08:58:28 am
cruzamento.vhd	✓	VHDL	2	04/30/2021 03:03:57 pm

```
Library x Project x sim x
Transcript
# Compile of Contador10.vhd was successful.
# Compile of Contador100.vhd was successful.
# Compile of cruzamento.vhd was successful.
# Compile of maquestados.vhd was successful.
# Compile of tb_cruzamento.vhd was successful.
# Compile of time_flags_v2.vhd was successful.
# 6 compiles, 0 failed with no errors.
```

Simulação:



Simulação foi feita com o sensor B fixo em 1 e o A fixo em 0, todos os valores da tabela seguinte foram computados após a borda de subida do clock:

	chave	dezena	unidade	Sinal A	Sinal B
Cursor 1	1	0000	1001	001	100
Cursor 2	0	0000	0000	010	010
Cursor 3	0	0000	0000	000	000
Cursor 4	1	0000	0000	001	100
Cursor 5	1	0001	1001	001	100
Cursor 6	1	0000	0000	010	100
Cursor 7	1	0000	0101	010	100
Cursor 8	1	0000	0000	100	100
Cursor 9	1	0000	0100	100	100
Cursor 10	1	0000	0000	100	001
Cursor 11	1	0101	1001	100	001
Cursor 12	1	0000	0000	100	010
Cursor 13	1	0000	0000	100	100
Cursor 14	1	0000	0000	001	100

Legenda:

- Sinais:

001 – Verde

010 – Amarelo

100 – Vermelho

- Chave:

1 – Sinais Ligados

0 – Sinais Desligados