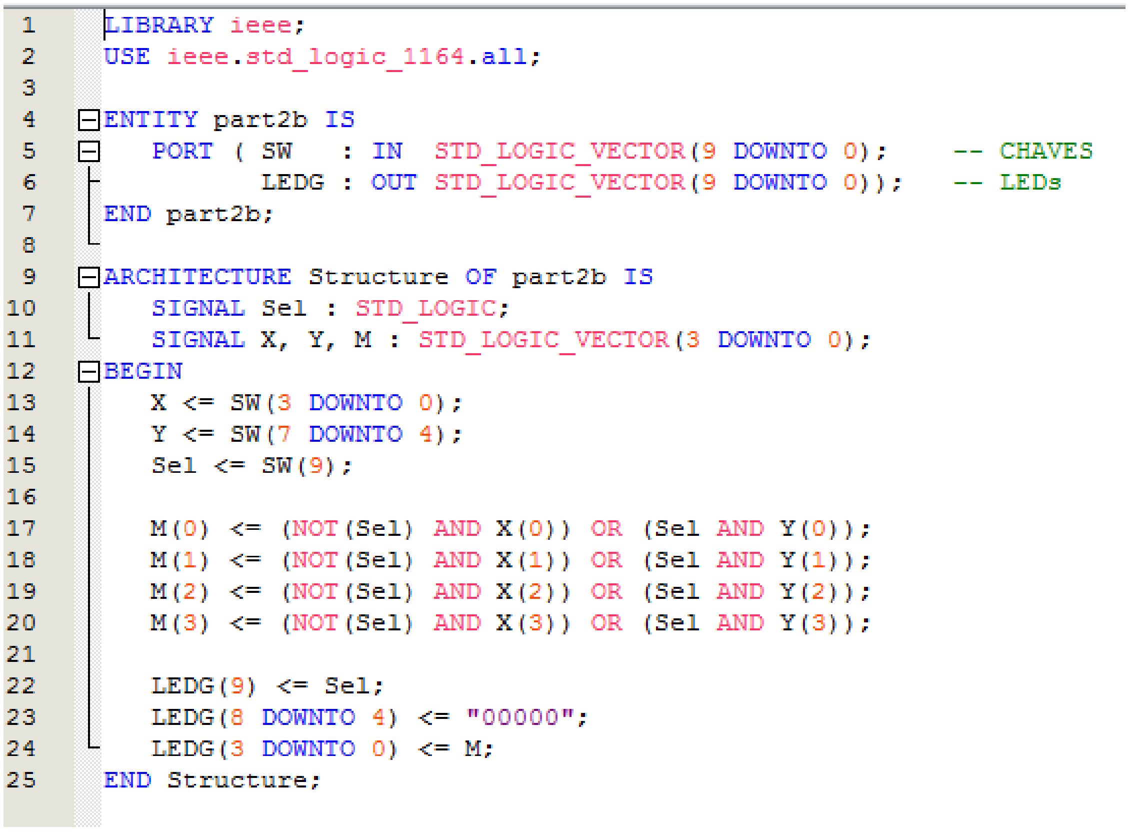
FUNDAÇÃO EDSON QUEIROZ
UNIVERSIDADE DE FORTALEZA
ENSINANDO E APRENDENDO

Sistemas Digitais Avançados

Prof. Imbiriba

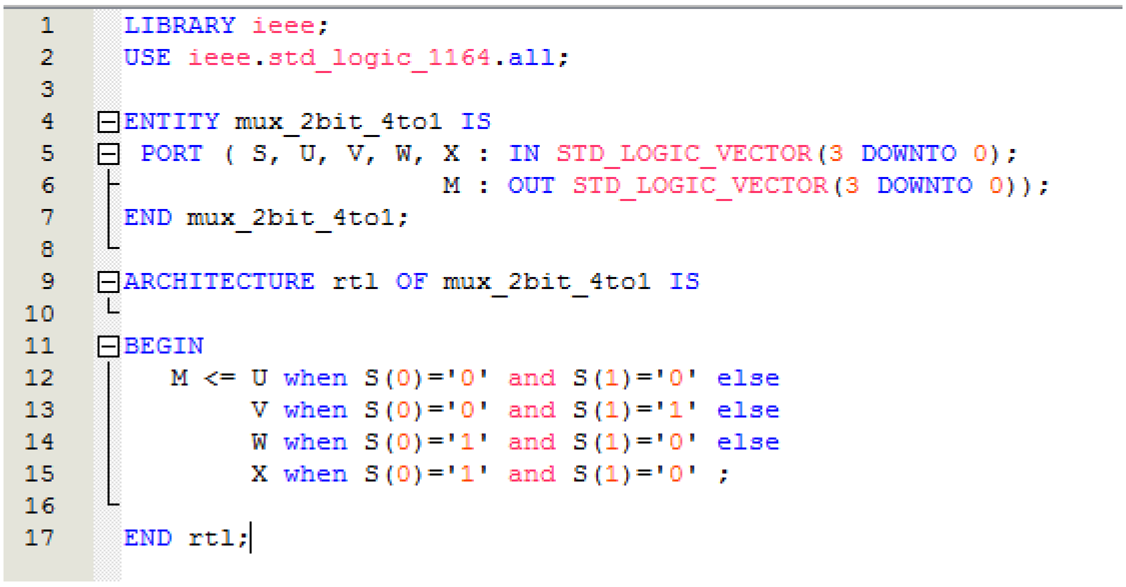
Solução - Trabalho 1

Parte 2a: Usando SOP



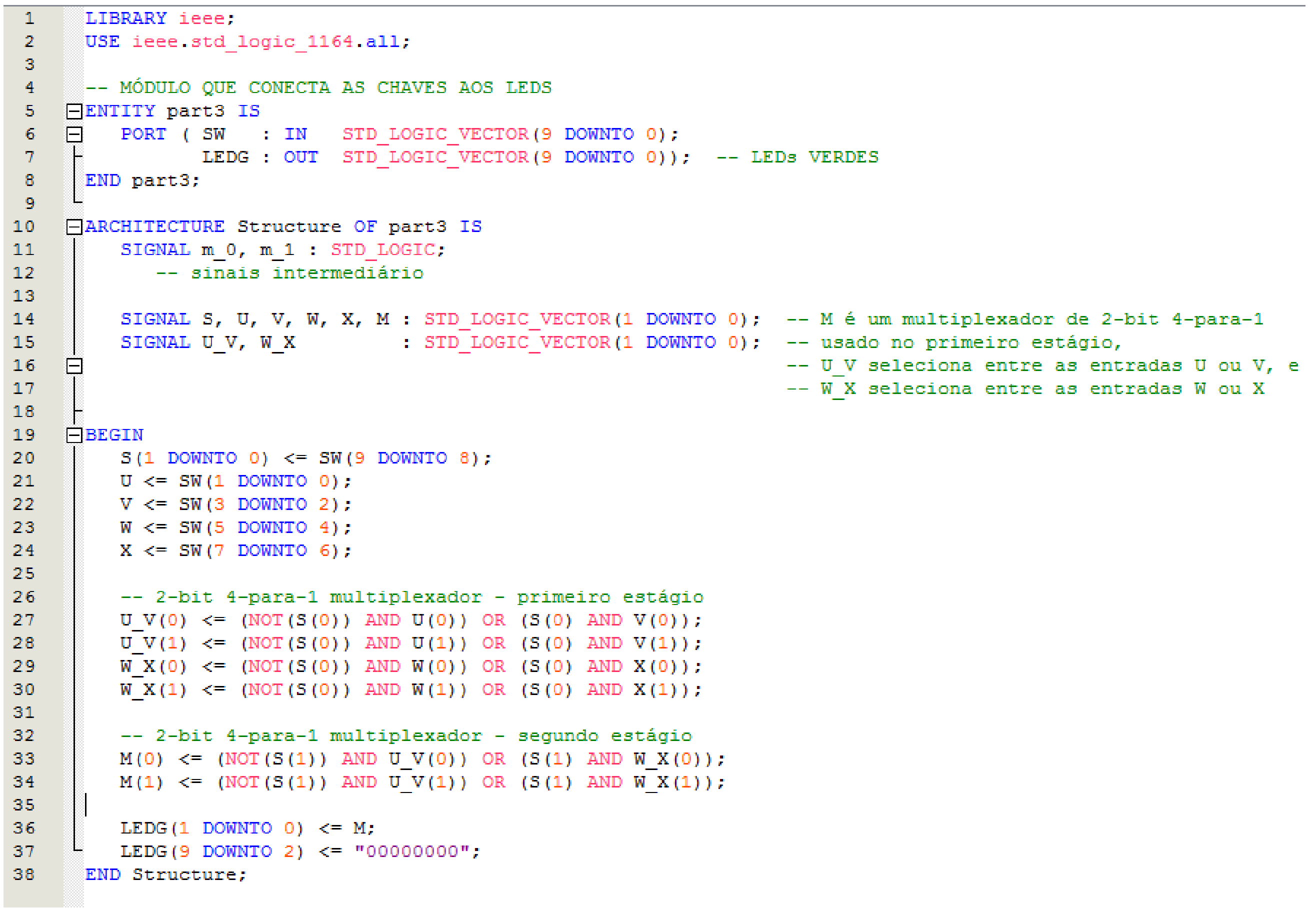
```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY part2b IS
    PORT ( SW   : IN  STD_LOGIC_VECTOR(9 DOWNTO 0);      -- CHAVES
           LEDG : OUT STD_LOGIC_VECTOR(9 DOWNTO 0));     -- LEDs
END part2b;

ARCHITECTURE Structure OF part2b IS
    SIGNAL Sel : STD_LOGIC;
    SIGNAL X, Y, M : STD_LOGIC_VECTOR(3 DOWNTO 0);
BEGIN
    X <= SW(3 DOWNTO 0);
    Y <= SW(7 DOWNTO 4);
    Sel <= SW(9);

    M(0) <= (NOT(Sel) AND X(0)) OR (Sel AND Y(0));
    M(1) <= (NOT(Sel) AND X(1)) OR (Sel AND Y(1));
    M(2) <= (NOT(Sel) AND X(2)) OR (Sel AND Y(2));
    M(3) <= (NOT(Sel) AND X(3)) OR (Sel AND Y(3));

    LEDG(9) <= Sel;
    LEDG(8 DOWNTO 4) <= "00000";
    LEDG(3 DOWNTO 0) <= M;
END Structure;
```

Parte 2b: Usando comando WHEN ELSE



```
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY mux_2bit_4to1 IS
 PORT ( S, U, V, W, X : IN STD_LOGIC_VECTOR(3 DOWNTO 0);
                    M : OUT STD_LOGIC_VECTOR(3 DOWNTO 0));
END mux_2bit_4to1;

ARCHITECTURE rtl OF mux_2bit_4to1 IS

BEGIN
   M <= U when S(0)='0' and S(1)='0' else
        V when S(0)='0' and S(1)='1' else
        W when S(0)='1' and S(1)='0' else
        X when S(0)='1' and S(1)='0' ;

END rtl;
```

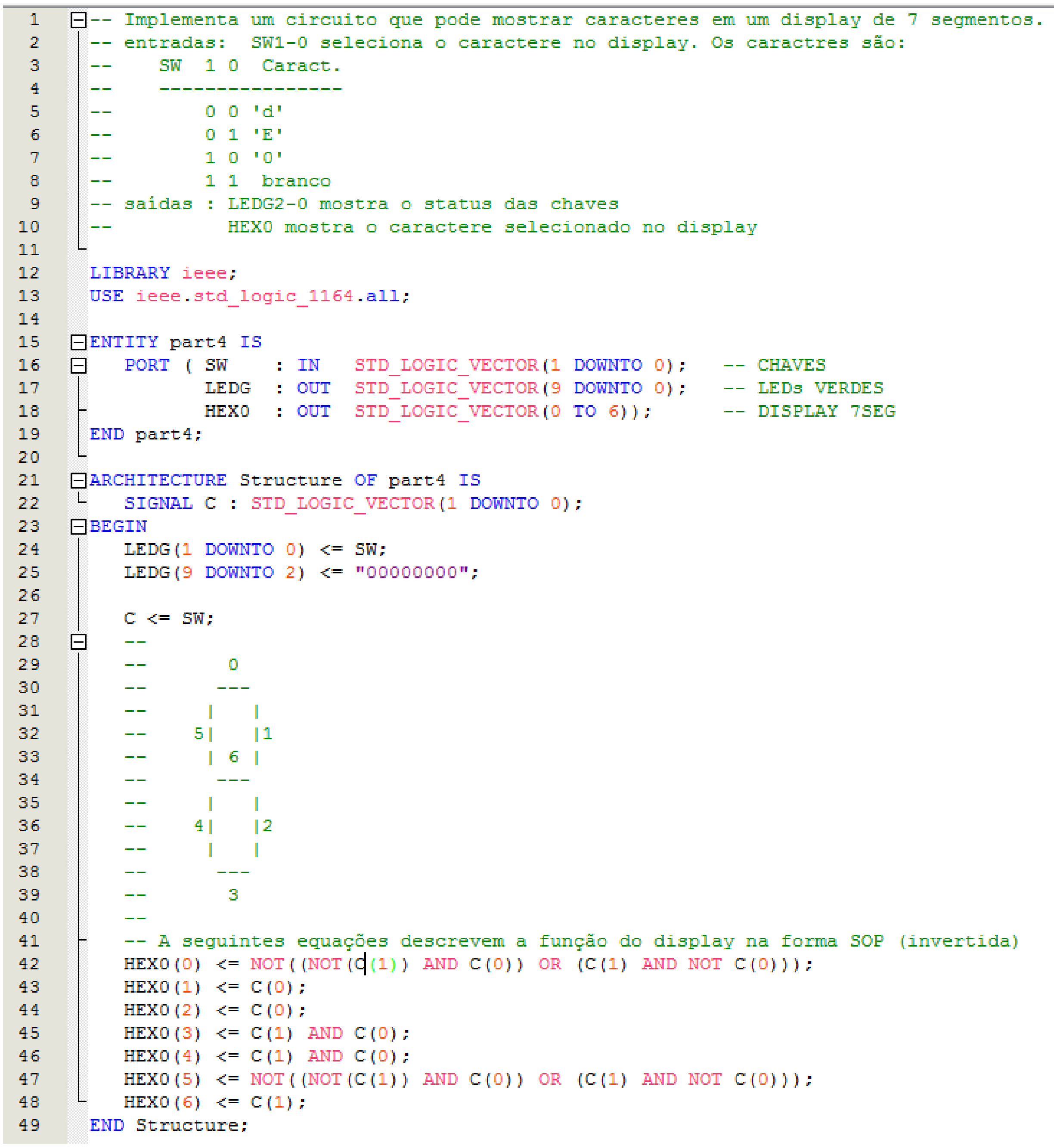
Parte3:



```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.all;

-- MÓDULO QUE CONECTA AS CHAVES AOS LEDS
ENTITY part3 IS
   PORT ( SW    : IN   STD_LOGIC_VECTOR(9 DOWNTO 0);
          LEDG : OUT  STD_LOGIC_VECTOR(9 DOWNTO 0));  -- LEDs VERDES
END part3;

ARCHITECTURE Structure OF part3 IS
   SIGNAL m_0, m_1 : STD_LOGIC;
      -- sinais intermediário

   SIGNAL S, U, V, W, X, M : STD_LOGIC_VECTOR(1 DOWNTO 0);  -- M é um multiplexador de 2-bit 4-para-1
   SIGNAL U_V, W_X         : STD_LOGIC_VECTOR(1 DOWNTO 0);  -- usado no primeiro estágio,
                                                            -- U_V seleciona entre as entradas U ou V, e
                                                            -- W_X seleciona entre as entradas W ou X

BEGIN
   S(1 DOWNTO 0) <= SW(9 DOWNTO 8);
   U <= SW(1 DOWNTO 0);
   V <= SW(3 DOWNTO 2);
   W <= SW(5 DOWNTO 4);
   X <= SW(7 DOWNTO 6);

   -- 2-bit 4-para-1 multiplexador - primeiro estágio
   U_V(0) <= (NOT(S(0)) AND U(0)) OR (S(0) AND V(0));
   U_V(1) <= (NOT(S(0)) AND U(1)) OR (S(0) AND V(1));
   W_X(0) <= (NOT(S(0)) AND W(0)) OR (S(0) AND X(0));
   W_X(1) <= (NOT(S(0)) AND W(1)) OR (S(0) AND X(1));

   -- 2-bit 4-para-1 multiplexador - segundo estágio
   M(0) <= (NOT(S(1)) AND U_V(0)) OR (S(1) AND W_X(0));
   M(1) <= (NOT(S(1)) AND U_V(1)) OR (S(1) AND W_X(1));

   LEDG(1 DOWNTO 0) <= M;
   LEDG(9 DOWNTO 2) <= "00000000";
END Structure;
```

Continua …

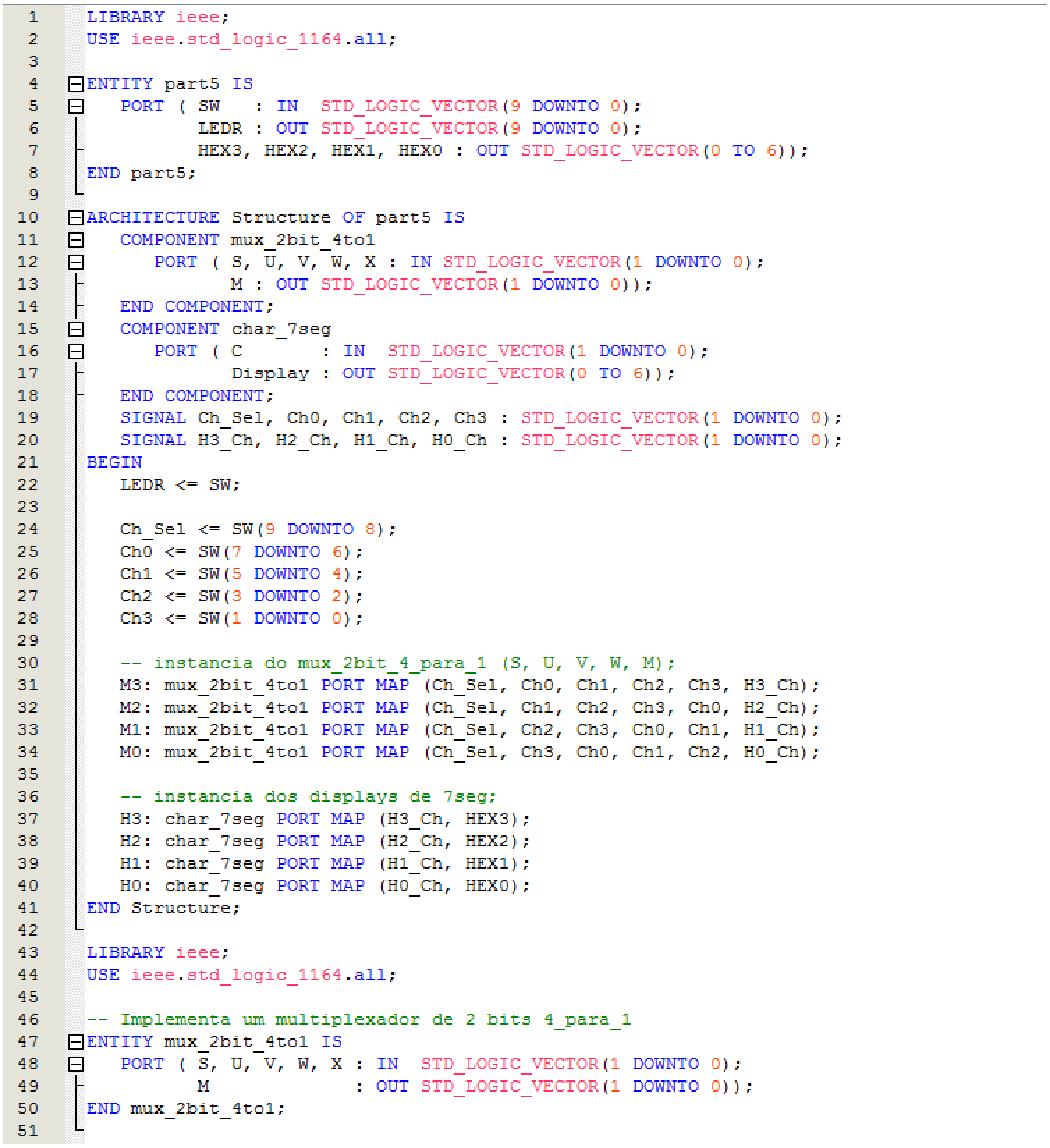
Parte4:



```vhdl
-- Implementa um circuito que pode mostrar caracteres em um display de 7 segmentos.
-- entradas:  SW1-0 seleciona o caractere no display. Os caractres são:
--    SW  1 0  Caract.
--    ----------------
--        0 0 'd'
--        0 1 'E'
--        1 0 '0'
--        1 1  branco
-- saídas : LEDG2-0 mostra o status das chaves
--          HEX0 mostra o caractere selecionado no display

LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY part4 IS
   PORT ( SW    : IN    STD_LOGIC_VECTOR(1 DOWNTO 0);   -- CHAVES
          LEDG  : OUT   STD_LOGIC_VECTOR(9 DOWNTO 0);   -- LEDs VERDES
          HEX0  : OUT   STD_LOGIC_VECTOR(0 TO 6));      -- DISPLAY 7SEG
END part4;

ARCHITECTURE Structure OF part4 IS
   SIGNAL C : STD_LOGIC_VECTOR(1 DOWNTO 0);
BEGIN
   LEDG(1 DOWNTO 0) <= SW;
   LEDG(9 DOWNTO 2) <= "00000000";

   C <= SW;
   --
   --       0
   --      ---
   --     |   |
   --    5|   |1
   --     | 6 |
   --      ---
   --     |   |
   --    4|   |2
   --     |   |
   --      ---
   --       3
   --
   -- A seguintes equações descrevem a função do display na forma SOP (invertida)
   HEX0(0) <= NOT((NOT(C(1)) AND C(0)) OR (C(1) AND NOT C(0)));
   HEX0(1) <= C(0);
   HEX0(2) <= C(0);
   HEX0(3) <= C(1) AND C(0);
   HEX0(4) <= C(1) AND C(0);
   HEX0(5) <= NOT((NOT(C(1)) AND C(0)) OR (C(1) AND NOT C(0)));
   HEX0(6) <= C(1);
END Structure;
```

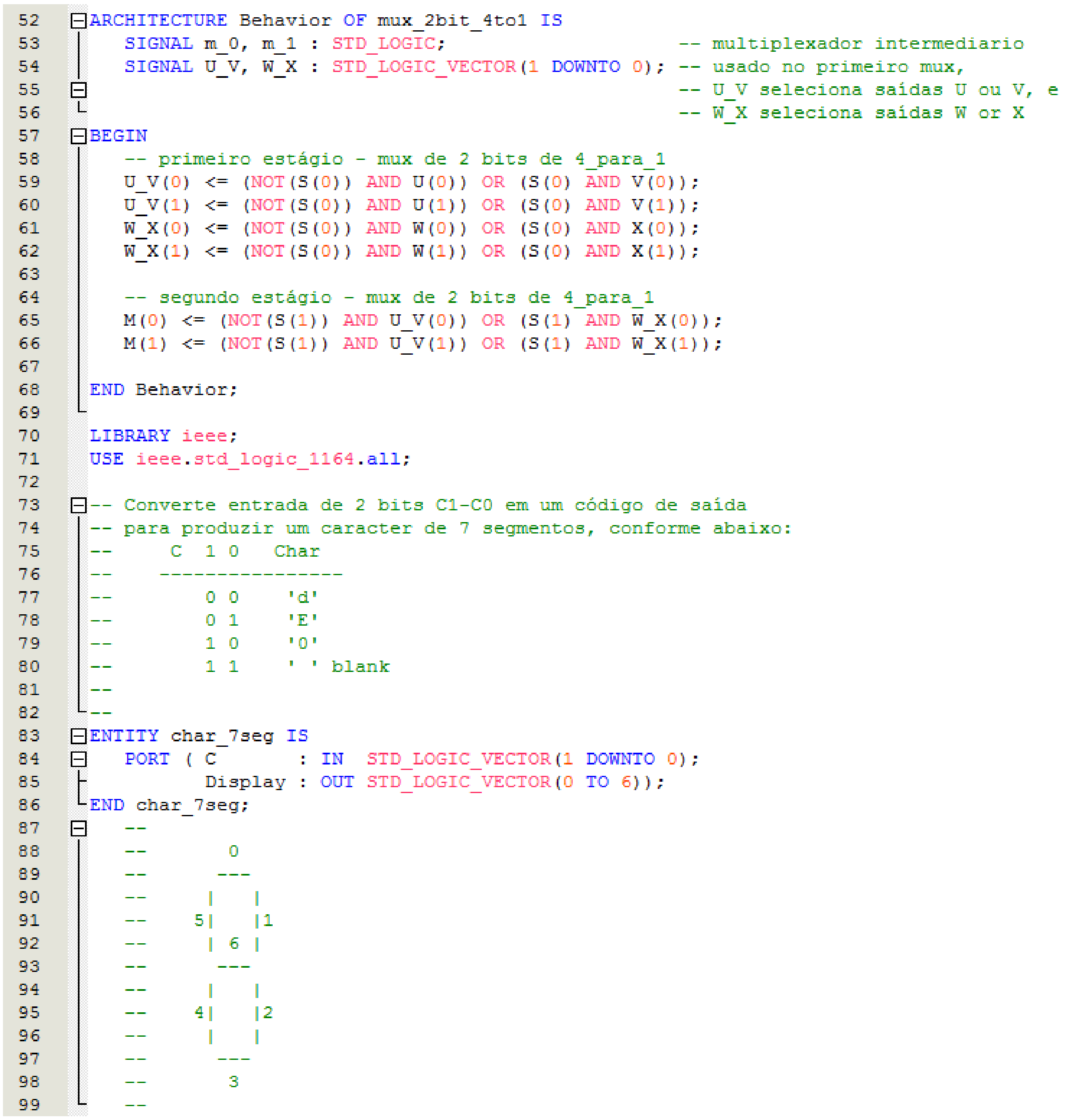
Continua...

Parte5



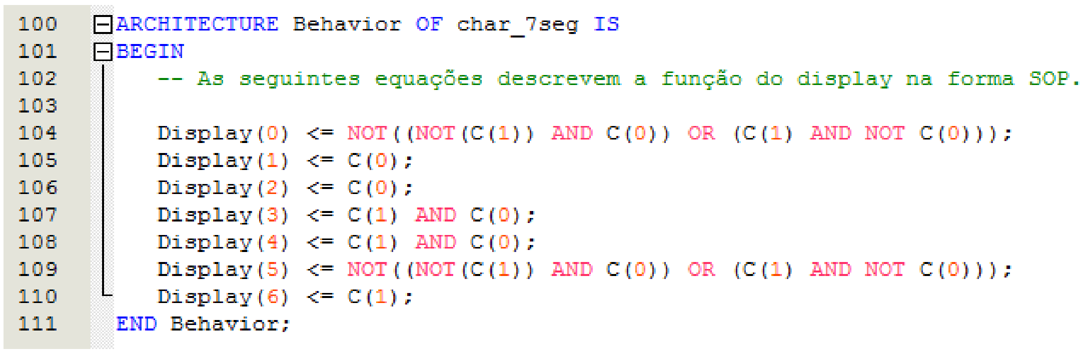
```vhdl
LIBRARY ieee;
USE ieee.std_logic_1164.all;

ENTITY part5 IS
   PORT ( SW    : IN  STD_LOGIC_VECTOR(9 DOWNTO 0);
          LEDR : OUT STD_LOGIC_VECTOR(9 DOWNTO 0);
          HEX3, HEX2, HEX1, HEX0 : OUT STD_LOGIC_VECTOR(0 TO 6));
END part5;

ARCHITECTURE Structure OF part5 IS
   COMPONENT mux_2bit_4to1
      PORT ( S, U, V, W, X : IN STD_LOGIC_VECTOR(1 DOWNTO 0);
             M : OUT STD_LOGIC_VECTOR(1 DOWNTO 0));
   END COMPONENT;
   COMPONENT char_7seg
      PORT ( C       : IN  STD_LOGIC_VECTOR(1 DOWNTO 0);
             Display : OUT STD_LOGIC_VECTOR(0 TO 6));
   END COMPONENT;
   SIGNAL Ch_Sel, Ch0, Ch1, Ch2, Ch3 : STD_LOGIC_VECTOR(1 DOWNTO 0);
   SIGNAL H3_Ch, H2_Ch, H1_Ch, H0_Ch : STD_LOGIC_VECTOR(1 DOWNTO 0);
BEGIN
   LEDR <= SW;

   Ch_Sel <= SW(9 DOWNTO 8);
   Ch0 <= SW(7 DOWNTO 6);
   Ch1 <= SW(5 DOWNTO 4);
   Ch2 <= SW(3 DOWNTO 2);
   Ch3 <= SW(1 DOWNTO 0);

   -- instancia do mux_2bit_4_para_1 (S, U, V, W, M);
   M3: mux_2bit_4to1 PORT MAP (Ch_Sel, Ch0, Ch1, Ch2, Ch3, H3_Ch);
   M2: mux_2bit_4to1 PORT MAP (Ch_Sel, Ch1, Ch2, Ch3, Ch0, H2_Ch);
   M1: mux_2bit_4to1 PORT MAP (Ch_Sel, Ch2, Ch3, Ch0, Ch1, H1_Ch);
   M0: mux_2bit_4to1 PORT MAP (Ch_Sel, Ch3, Ch0, Ch1, Ch2, H0_Ch);

   -- instancia dos displays de 7seg;
   H3: char_7seg PORT MAP (H3_Ch, HEX3);
   H2: char_7seg PORT MAP (H2_Ch, HEX2);
   H1: char_7seg PORT MAP (H1_Ch, HEX1);
   H0: char_7seg PORT MAP (H0_Ch, HEX0);
END Structure;

LIBRARY ieee;
USE ieee.std_logic_1164.all;

-- Implementa um multiplexador de 2 bits 4_para_1
ENTITY mux_2bit_4to1 IS
   PORT ( S, U, V, W, X : IN  STD_LOGIC_VECTOR(1 DOWNTO 0);
          M             : OUT STD_LOGIC_VECTOR(1 DOWNTO 0));
END mux_2bit_4to1;

```

Continua ...



```
ARCHITECTURE Behavior OF mux_2bit_4to1 IS
   SIGNAL m_0, m_1 : STD_LOGIC;                     -- multiplexador intermediario
   SIGNAL U_V, W_X : STD_LOGIC_VECTOR(1 DOWNTO 0); -- usado no primeiro mux,
                                                    -- U_V seleciona saídas U ou V, e
                                                    -- W_X seleciona saídas W or X
BEGIN
   -- primeiro estágio - mux de 2 bits de 4_para_1
   U_V(0) <= (NOT(S(0)) AND U(0)) OR (S(0) AND V(0));
   U_V(1) <= (NOT(S(0)) AND U(1)) OR (S(0) AND V(1));
   W_X(0) <= (NOT(S(0)) AND W(0)) OR (S(0) AND X(0));
   W_X(1) <= (NOT(S(0)) AND W(1)) OR (S(0) AND X(1));

   -- segundo estágio - mux de 2 bits de 4_para_1
   M(0) <= (NOT(S(1)) AND U_V(0)) OR (S(1) AND W_X(0));
   M(1) <= (NOT(S(1)) AND U_V(1)) OR (S(1) AND W_X(1));

END Behavior;

LIBRARY ieee;
USE ieee.std_logic_1164.all;

-- Converte entrada de 2 bits C1-C0 em um código de saída
-- para produzir um caracter de 7 segmentos, conforme abaixo:
--      C  1 0   Char
--     ----------------
--         0 0    'd'
--         0 1    'E'
--         1 0    '0'
--         1 1    ' ' blank
--
--
ENTITY char_7seg IS
   PORT ( C       : IN  STD_LOGIC_VECTOR(1 DOWNTO 0);
          Display : OUT STD_LOGIC_VECTOR(0 TO 6));
END char_7seg;
   --
   --        0
   --       ---
   --      |   |
   --     5|   |1
   --      | 6 |
   --       ---
   --      |   |
   --     4|   |2
   --      |   |
   --       ---
   --        3
   --
```

Continua ...



```vhdl
ARCHITECTURE Behavior OF char_7seg IS
BEGIN
   -- As seguintes equações descrevem a função do display na forma SOP.

   Display(0) <= NOT((NOT(C(1)) AND C(0)) OR (C(1) AND NOT C(0)));
   Display(1) <= C(0);
   Display(2) <= C(0);
   Display(3) <= C(1) AND C(0);
   Display(4) <= C(1) AND C(0);
   Display(5) <= NOT((NOT(C(1)) AND C(0)) OR (C(1) AND NOT C(0)));
   Display(6) <= C(1);
END Behavior;
```

FIM