MC613

Laboratório de Circuitos Lógicos

2010

Profs.:

Guido Araújo (guido @ ic….)
Mário Lúcio Cortes (cortes @ ic….)

MC613

Revisão de Circuitos Lógicos

"Fundamentals of Digital Logic with VHDL Design"

Conteúdo

- Mux – Comandos Concorrentes
- Decodificador
- Codificador
- Comparador
- Generate Statement
- Instanciação Condicional

Conteúdo

- MUX – Comandos Seqüenciais
- Codificador
- Comparador
- BCD para 7 Segmentos

MUX - Comandos Concorrentes

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

ENTITY mux2to1 IS
   PORT (w0, w1, s  : IN    STD_LOGIC ;
         f          : OUT STD_LOGIC ) ;
END mux2to1 ;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
   WITH s SELECT
      f <=   w0 WHEN '0',
             w1 WHEN OTHERS ;
END Behavior ;
```

MUX – Comandos Concorrentes

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY mux4to1 IS
   PORT (w0, w1, w2, w3  : IN    STD_LOGIC ;
          s     : IN    STD_LOGIC_VECTOR(1 DOWNTO 0);
          f     : OUT   STD_LOGIC ) ;
END mux4to1 ;

ARCHITECTURE Behavior OF mux4to1 IS
BEGIN
   WITH s SELECT
       f <= w0 WHEN "00",
            w1 WHEN "01",
            w2 WHEN "10",
            w3 WHEN OTHERS ;
END Behavior ;
```

MUX

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

PACKAGE mux4to1_package IS
   COMPONENT mux4to1
        PORT (w0, w1, w2, w3 : IN       STD_LOGIC ;
                s : IN    STD_LOGIC_VECTOR(1 DOWNTO 0);
                f : OUT STD_LOGIC ) ;
   END COMPONENT ;
END mux4to1_package ;
```

MUX

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
LIBRARY work ;
USE work.mux4to1_package.all ;

ENTITY mux16to1 IS
   PORT (w : IN  STD_LOGIC_VECTOR(0 TO 15) ;
         s : IN   STD_LOGIC_VECTOR(3 DOWNTO 0) ;
         f : OUT STD_LOGIC ) ;
END mux16to1 ;
```

MUX

```
ARCHITECTURE Structure OF mux16to1 IS
   SIGNAL m : STD_LOGIC_VECTOR(0 TO 3);

BEGIN
    Mux1: mux4to1 PORT MAP (w(0), w(1), w(2), w(3),
                              s(1 DOWNTO 0), m(0) );
    Mux2: mux4to1 PORT MAP (w(4), w(5), w(6), w(7),
                             s(1 DOWNTO 0), m(1) );
    Mux3: mux4to1 PORT MAP (w(8), w(9), w(10), w(11),
                             s(1 DOWNTO 0), m(2) );
    Mux4: mux4to1 PORT MAP (w(12), w(13), w(14), w(15),
                             s(1 DOWNTO 0), m(3) );
    Mux5: mux4to1 PORT MAP (m(0), m(1), m(2), m(3),
                             s(3 DOWNTO 2), f ) ;
END Structure ;
```

MUX

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

ENTITY mux2to1 IS
   PORT (w0, w1, s : IN  STD_LOGIC;
            f : OUT   STD_LOGIC);
END mux2to1 ;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
   f <= w0 WHEN s = '0' ELSE w1 ;
END Behavior ;
```

Decodificador

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

ENTITY dec2to4 IS
   PORT (w    : IN    STD_LOGIC_VECTOR(1 DOWNTO 0);
         En   : IN    STD_LOGIC;
         y    : OUT STD_LOGIC_VECTOR(0 TO 3));
END dec2to4 ;
```

Decodificador

```
ARCHITECTURE Behavior OF dec2to4 IS
   SIGNAL Enw : STD_LOGIC_VECTOR(2 DOWNTO 0);

BEGIN
   Enw <= En & w ;
   WITH Enw SELECT
        y <= "1000" WHEN "100",
             "0100" WHEN "101",
             "0010" WHEN "110",
             "0001" WHEN "111",
             "0000" WHEN OTHERS ;
END Behavior ;
```

Codificador

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY priority IS
   PORT (w    : IN    STD_LOGIC_VECTOR(3 DOWNTO 0);
         y    : OUT STD_LOGIC_VECTOR(1 DOWNTO 0);
         z    : OUT STD_LOGIC );
END priority;

ARCHITECTURE Behavior OF priority IS
BEGIN
   y <= "11" WHEN w(3) = '1' ELSE
        "10" WHEN w(2) = '1' ELSE
        "01" WHEN w(1) = '1' ELSE  "00";
   z <= '0' WHEN w = "0000" ELSE '1';
END Behavior ;
```

Codificador

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

ENTITY priority IS
   PORT (w  : IN    STD_LOGIC_VECTOR(3 DOWNTO 0);
         y  : OUT   STD_LOGIC_VECTOR(1 DOWNTO 0);
         z  : OUT   STD_LOGIC );
END priority;
```

Codificador

```
ARCHITECTURE Behavior OF priority IS
BEGIN
   WITH w SELECT
      y <=   "00" WHEN "0001",
             "01" WHEN "0010",
             "01" WHEN "0011",
             "10" WHEN "0100",
             "10" WHEN "0101",
             "10" WHEN "0110",
             "10" WHEN "0111",
             "11" WHEN OTHERS;
   WITH w SELECT
      z <=   '0' WHEN "0000",
             '1' WHEN OTHERS;
END Behavior ;
```

Comparador – Sem Sinal

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_unsigned.all;
ENTITY compare IS
   PORT (A, B : IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
           AeqB, AgtB, AltB  : OUT STD_LOGIC );
END compare ;

ARCHITECTURE Behavior OF compare IS
BEGIN
   AeqB <= '1' WHEN A = B ELSE '0';
   AgtB <= '1' WHEN A > B ELSE '0';
   AltB  <= '1' WHEN A < B ELSE '0';
END Behavior ;
```

Comparador – Com Sinal

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all;
USE ieee.std_logic_arith.all;
ENTITY compare IS
   PORT (A, B      : IN SIGNED(3 DOWNTO 0) ;
            AeqB, AgtB, AltB : OUT   STD_LOGIC ) ;
END compare ;

ARCHITECTURE Behavior OF compare IS
BEGIN
   AeqB <=  '1' WHEN A = B ELSE '0' ;
   AgtB <=  '1' WHEN A > B ELSE '0' ;
   AltB  <=  '1' WHEN A < B ELSE '0' ;
END Behavior ;
```

Generate Statement

Mux 16-para-1 usando Generate Statement

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
USE work.mux4to1_package.all ;

ENTITY mux16to1 IS
   PORT (w   : IN  STD_LOGIC_VECTOR(0 TO 15) ;
         s   : IN  STD_LOGIC_VECTOR(3 DOWNTO 0);
         f   : OUT STD_LOGIC ) ;
END mux16to1 ;
```

Generate Statement

Mux 16-para-1 usando Generate Statement

```
ARCHITECTURE Structure OF mux16to1 IS
     SIGNAL m : STD_LOGIC_VECTOR(0 TO 3) ;
BEGIN
  G1: FOR i IN 0 TO 3 GENERATE
    Muxes: mux4to1 PORT MAP (w(4*i), w(4*i+1), w(4*i+2),
                                  w(4*i+3), s(1 DOWNTO 0), m(i));
     END GENERATE ;
     Mux5: mux4to1 PORT MAP (m(0), m(1), m(2), m(3),
                                   s(3 DOWNTO 2), f ) ;
END Structure ;
```

MUX – Comandos Seqüenciais

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

ENTITY mux2to1 IS
   PORT (w0, w1, s  : IN          STD_LOGIC;
           f        : OUT  STD_LOGIC );
END mux2to1 ;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
   PROCESS ( w0, w1, s )
   BEGIN
        IF s = '0'
           THEN  f <= w0 ;
           ELSE   f <= w1 ;
        END IF ;
   END PROCESS ;
END Behavior ;
```

MUX – Comandos Seqüenciais

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

ENTITY mux2to1 IS
   PORT (w0, w1, s  : IN    STD_LOGIC;
         f          : OUT   STD_LOGIC);
END mux2to1 ;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
   PROCESS (w0, w1, s)
   BEGIN
        f <= w0;
        IF s = '1'
          THEN  f <= w1;
        END IF ;
   END PROCESS;
END Behavior ;
```

Codificador

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

ENTITY priority IS
   PORT (w  : IN     STD_LOGIC_VECTOR(3 DOWNTO 0);
         y  : OUT    STD_LOGIC_VECTOR(1 DOWNTO 0);
         z  : OUT    STD_LOGIC );
END priority ;
```

Codificador

```
ARCHITECTURE Behavior OF priority IS
BEGIN
   PROCESS ( w )
   BEGIN
        IF w(3) = '1' THEN
                y <= "11";
        ELSIF w(2) = '1' THEN
                y <= "10";
        ELSIF w(1) = '1' THEN
                y <= "01";
        ELSE
                y <= "00";
        END IF ;
   END PROCESS ;
   z <= '0' WHEN w = "0000" ELSE '1' ;
END Behavior ;
```

Codificador

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

ENTITY priority IS
   PORT (w : IN      STD_LOGIC_VECTOR(3 DOWNTO 0);
             y : OUT   STD_LOGIC_VECTOR(1 DOWNTO 0);
             z : OUT   STD_LOGIC ) ;
END priority ;
ARCHITECTURE Behavior OF priority IS
BEGIN
   PROCESS ( w )
   BEGIN
        y <= "00";
        IF w(1) = '1' THEN y <= "01" ; END IF ;
        IF w(2) = '1' THEN y <= "10" ; END IF ;
        IF w(3) = '1' THEN y <= "11" ; END IF ;
        z <= '1' ;
        IF w = "0000" THEN z <= '0' ; END IF ;
   END PROCESS ;
END Behavior ;
```

Comparador

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

ENTITY compare1 IS
   PORT (A, B :  IN    STD_LOGIC ;
          AeqB :  OUT   STD_LOGIC ) ;
END compare1 ;

ARCHITECTURE Behavior OF compare1 IS
BEGIN
   PROCESS ( A, B )
   BEGIN
        AeqB <= '0' ;
        IF A = B THEN
                AeqB <= '1' ;
        END IF ;
   END PROCESS ;
END Behavior;
```

Comparador

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

ENTITY implied IS
   PORT ( A, B   : IN     STD_LOGIC ;
          AeqB   : OUT   STD_LOGIC ) ;
END implied ;

ARCHITECTURE Behavior OF implied IS
BEGIN
   PROCESS ( A, B )
   BEGIN
      IF A = B THEN
           AeqB <= '1' ;
      END IF ;
   END PROCESS ;
END Behavior ;
```

Este código INFERE memória (latch)

MUX

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

ENTITY mux2to1 IS
   PORT (w0, w1, s  : IN     STD_LOGIC ;
         f          : OUT    STD_LOGIC ) ;
END mux2to1;

ARCHITECTURE Behavior OF mux2to1 IS
BEGIN
   PROCESS ( w0, w1, s )
   BEGIN
        CASE s IS
            WHEN '0'  =>  f <= w0 ;
            WHEN OTHERS => f <= w1 ;
        END CASE ;
   END PROCESS ;
END Behavior ;
```

Decoder 2→4

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY dec2to4 IS
      PORT (w   : IN    STD_LOGIC_VECTOR(1 DOWNTO 0);
             En  : IN    STD_LOGIC;
             y   : OUT  STD_LOGIC_VECTOR(0 TO 3) );
END dec2to4 ;
ARCHITECTURE Behavior OF dec2to4 IS
BEGIN
      PROCESS ( w, En )
      BEGIN
         IF En = '1' THEN
               CASE w IS
                  WHEN "00"          =>   y <= "1000" ;
                  WHEN "01"          =>   y <= "0100" ;
                  WHEN "10"          =>   y <= "0010" ;
                  WHEN OTHERS  =>   y <= "0001" ;
               END CASE ;
         ELSE                                y <= "0000" ;
         END IF ;
      END PROCESS ;
END Behavior;
```

BCD para 7 Segmentos

```
LIBRARY ieee ;
USE ieee.std_logic_1164.all ;
ENTITY seg7 IS
    PORT (bcd     : IN    STD_LOGIC_VECTOR(3 DOWNTO 0) ;
          leds    : OUT   STD_LOGIC_VECTOR(1 TO 7) ) ;
END seg7 ;
ARCHITECTURE Behavior OF seg7 IS
BEGIN
    PROCESS ( bcd )
    BEGIN
          CASE bcd IS          --                   abcdefg
              WHEN "0000"     => leds    <=      "1111110";
              WHEN "0001"     => leds    <=      "0110000";
              WHEN "0010"     => leds    <=      "1101101";
              WHEN "0011"     => leds    <=      "1111001";
              WHEN "0100"     => leds    <=      "0110011";
              WHEN "0101"     => leds    <=      "1011011";
              WHEN "0110"     => leds    <=      "1011111";
              WHEN "0111"     => leds    <=      "1110000";
              WHEN "1000"     => leds    <=      "1111111";
              WHEN "1001"     => leds    <=      "1110011";
              WHEN OTHERS     => leds    <=      "-------";
          END CASE ;
    END PROCESS ;
END Behavior ;
```