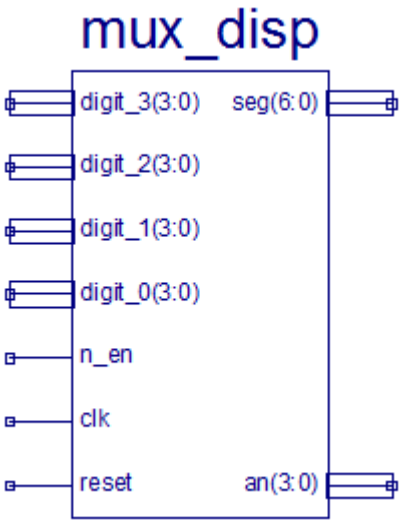
Pilotaggio multiplexato di 4 display a 7 segmenti

Descrizione dell'attività:

Realizzare un componente di libreria per il pilotaggio multiplexato dei 4 display a 7 segmenti presenti sulla scheda nexys3.

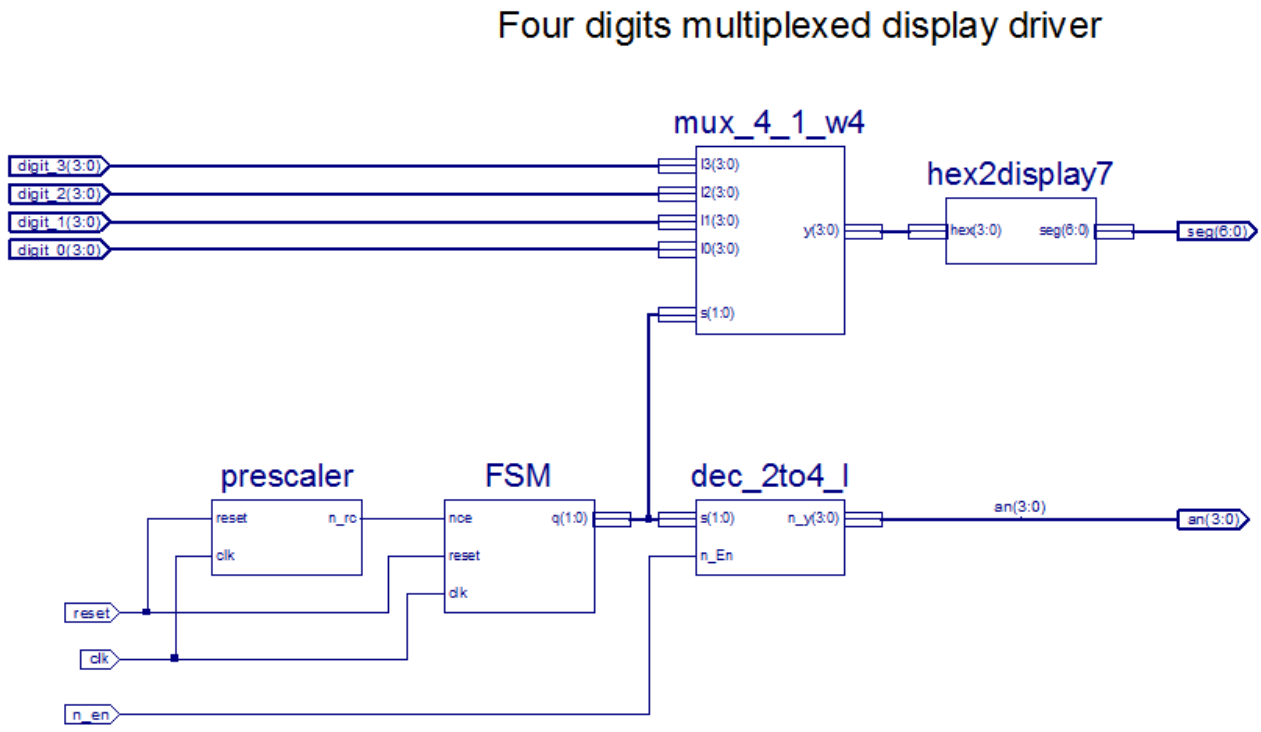


digit_3(3:0) : (I) valore esadecimale IV cifra;
digit_2(3:0) : (I) valore esadecimale III cifra;
digit_1(3:0) : (I) valore esadecimale II cifra;
digit_0(3:0) : (I) valore esadecimale I cifra;

n_en : (I) spegnimento display;
clk : (I) clock;
reset : (I) reset;

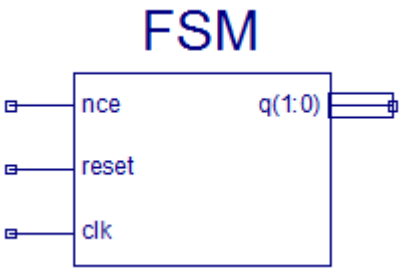
seg(6:0) : (o) segmenti, 6 → g, … 0 → a;
an(3:0) : (o) abilitazione digit (attivi bassi);

Schema logico del display driver



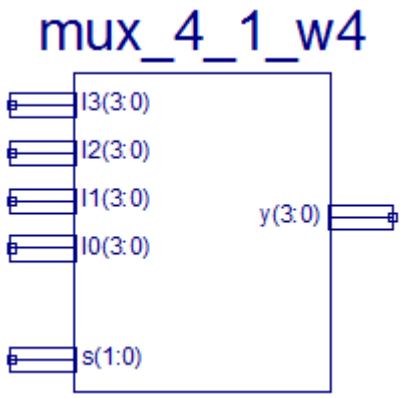

```
entity prescaler is
        generic(CLK_FREQ: natural   := 100;         -- Main frequency     MHz (system clock)
                OUT_FREQ: natural   := 200);        -- Ouput frequency    Hz

        Port (  reset   : in    STD_LOGIC;
                clk     : in    STD_LOGIC;
                n_rc    : out   STD_LOGIC);
end prescaler;

architecture Behavioral of prescaler is
signal cnt_clk: integer range 0 to (CLK_FREQ*1_000_000)/OUT_FREQ-1;

begin
cnt:process(clk, reset)
     begin
        if(clk'event and clk ='1')then
                if      (reset = '1')   then    cnt_clk <= (CLK_FREQ*1_000_000)/OUT_FREQ-1;
                elsif   (cnt_clk = 0)   then    cnt_clk <= (CLK_FREQ*1_000_000)/OUT_FREQ-1;
                else                            cnt_clk <= cnt_clk - 1;
                end if;
        end if;
end process;

rlc:process(cnt_clk, clk)
     begin
        n_rc <= '1';
        if( (cnt_clk = 0) and (clk = '0') ) then
                n_rc <= '0';
        end if;
end process;
end Behavioral;
```

Contatore binario a 2 bit



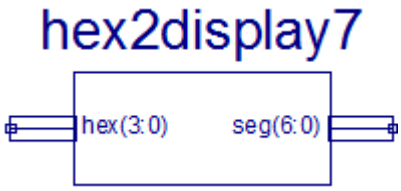

```
entity FSM is
port (  clk     : in    STD_LOGIC;
        reset   : in    STD_LOGIC;
        nce     : in    STD_LOGIC;
       q        : out   STD_LOGIC_VECTOR (1 downto 0));
end FSM;

architecture Behavioral of FSM is
signal q_int: std_logic_vector(1 downto 0);
begin
q <= q_int;
process(clk, reset)
begin
        if(clk'event and clk ='1') then

                if      (reset = '1')   then
                        q_int <= (others => '0');

                elsif   (nce = '0')     then
                        q_int <= q_int+"01";

                end if;

        end if;
end process;
end Behavioral;
```

Multiplexer 4:1, parallelismo dati 4 bit




```
entity mux_4_1_w4 is
port (  I0      : in    STD_LOGIC_VECTOR (3 downto 0);
        I1      : in    STD_LOGIC_VECTOR (3 downto 0);
        I2      : in    STD_LOGIC_VECTOR (3 downto 0);
        I3      : in    STD_LOGIC_VECTOR (3 downto 0);
        s       : in    STD_LOGIC_VECTOR (1 downto 0);
        y       : out   STD_LOGIC_VECTOR (3 downto 0));
end mux_4_1_w4;

architecture Behavioral of mux_4_1_w4 is

begin
        y <=    I0 when s = "00"        else
                I1 when s =  "01"       else
                I2 when s = "10"        else
                I3 when s = "11"        else "0000";
end Behavioral;
```

Decodifica binario → 7 segmenti




```
entity hex2disp7 is
port(hex   : in       std_logic_vector(3 downto 0);
     seg   : out      std_logic_vector(6 downto 0));
end hex2disp7;

architecture behavior of hex2disp7 is
begin

with hex select
     seg <=     "1000000" when "0000", -- 0
                "1111001" when "0001", -- 1
                "0100100" when "0010", -- 2
                "0110000" when "0011", -- 3
                "0011001" when "0100", -- 4
                "0010010" when "0101", -- 5
                "0000010" when "0110", -- 6
                "1111000" when "0111", -- 7
                "0000000" when "1000", -- 8
                "0010000" when "1001", -- 9
                "0001000" when "1010", -- a
                "0000011" when "1011", -- b
                "1000110" when "1100", -- c
                "0100001" when "1101", -- d
                "0000110" when "1110", -- e
                "0001110" when "1111", -- f
                "1111111" when others;

end behavior;
```

Uscita ad 1 → segmento spento; Uscita ad 0 → segmento acceso;