دانشگاه صنعتی امیرکبیر
(پلی‌تکنیک تهران)
دانشکده برق

گزارشکار آزمایشگاه مدار منطقی

ساخت یک دیکدر bcd به 7segment برای عداد چهار بیتی

نگارش
پارسا محمّدی ۹۹۲۳۱۲۱

استاد درس
مهندس مهرابادی

اردیبهشت ۱۴۰۲

در این پروژه یک نمایش دهنده اعداد که با 7segment کار میکند طراحی شده است برای ساخت این ماژول از یک دیکدر bcd به 7segment استفاده شده است. این دیکدر در جلسات قبلی ساخته شده است. با توجه به اینکه نمی‌توان تمام دیجیت های صفحه در آن روشن نگه داشت؛ با آن ها را یکی یکی روشن کنیم. برای این پروژه از سیگنال کلاک ۱۰ نانو ثانیه استفاده شده است و بعد در یک پروسس آن را به یک سیگانل ۴۰ نانو ثانیه کردیم که هر دیجیت ۴۰ نانو ثانیه روشن باشد و بعد از آن دیجیت بعدی مقدار دهی می‌شود و نمایش داده می‌شود.

```
44  -- Makking clock number 2
45  signal clk2 : std_logic := '0';
46
47  -- Creating list of Cases for digits
48  type Digit is (start,digit_one, digit_two, digit_three, digit_four);
49  signal CurrentDigit : Digit;
50
51  -- Input of decoders
52  signal decoder_input : std_logic_vector (3 downto 0);
53
54  -- Output of decoders
55  signal A_dec, B_dec, C_dec, D_dec, E_dec, F_dec, G_dec : std_logic;
56
```

در این پروژه ابتدا سیگنال کلاک۲ ساخته شده است که برای تعیین زمان روشن و خاموش شدن ال ای دی هاست.

در مرحله بعد چند کیس ساخته شده اند یک کیس ها مشخص میکنند که در هر لحظه کدام دیجیت ها مقدار دهی شوند.

بقیه سیگنال برای ادامه جزئیات کد می‌باشد.

در اینجا دیکدر اضافه شده است.

```
60  -- Inserting BCD to Seven Segment component
61  digits : Decoder
62     port map(
63           B0 => decoder_input(3),
64           B1 => decoder_input(2),
65           B2 => decoder_input(1),
66           B3 => decoder_input(0),
67           A => A_dec,
68           B => B_dec,
69           C => C_dec,
70           D => D_dec,
71           E => E_dec,
72           F => F_dec,
73           G => G_dec
74  );
75
```

در این پروسس سیگنال کلاک به یک کلاک با پریود بیشتر تبدیل می‌شود که تا زمان مورد نظر ما که می‌خواهیم هر دیجیت روشن باشد را ایجاد کند. بعدا از این سیگنال برای یک پروسس دیگر استفاده خواهیم کرد که روشن شدن دیجیت ها را مشخص می‌کند.

```
clock_process: process(clk)
   variable counter : std_logic_vector (3 downto 0) := "0000";
   begin
      if (rising_edge(clk)) then
         if(counter < "0011") then
            counter := counter + 1;

         elsif (counter = "0011") then
            counter := "0000";

            if (clk2 = '0') then
               clk2 <= '1';
            elsif (clk2 = '1') then
               clk2 <= '0';
            end if;

         end if;
         clk_out   <= clk2;
      end if;
end process clock_process;
```

در اینجا پروسس اصلی انجام می‌شود که در هر کیس فقط به یکی از دیجیت ها مقدار دهی می‌شود و بقیه رها می‌شوند. عوض شدن هر کیس وابسته به عوض شدن سیگنال clk2 می‌باشد.

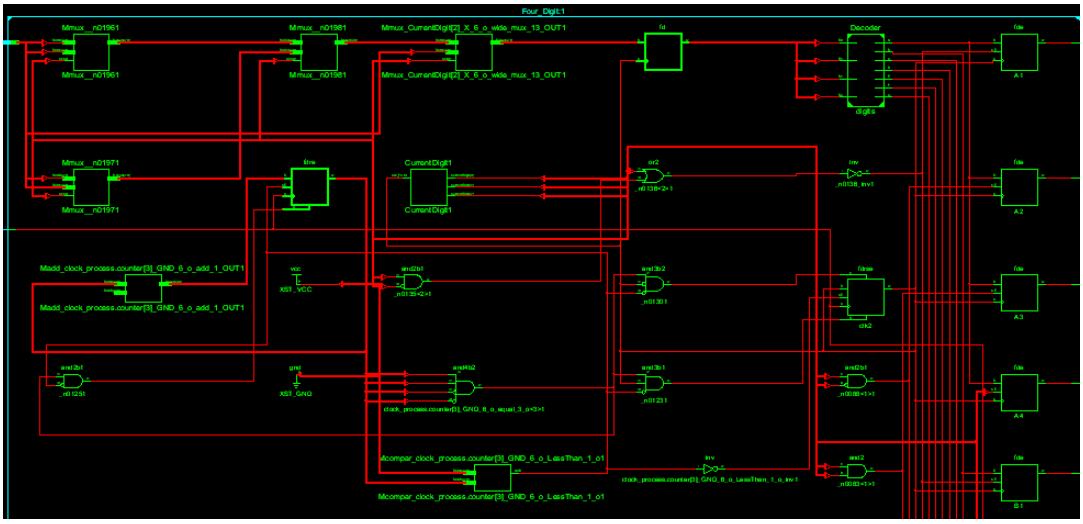
```
main_process : process(clk2)
begin
      if (rising_edge(clk2)) then
         case CurrentDigit is

               when start =>  -- start stage
                     A1 <= A_dec;
                     B1 <= B_dec;
                     C1 <= C_dec;
                     D1 <= D_dec;
                     E1 <= E_dec;
                     F1 <= F_dec;
                     G1 <= G_dec;
                     decoder_input <= input(3 DOWNTO 0);

                     CurrentDigit <= digit_one;

               when digit_one => -- Assiging value for first digit
                     A1 <= A_dec;
                     B1 <= B_dec;
                     C1 <= C_dec;
                     D1 <= D_dec;
                     E1 <= E_dec;
                     F1 <= F_dec;
                     G1 <= G_dec;
                     decoder_input <= input(7 DOWNTO 4);
```

تست بنچ

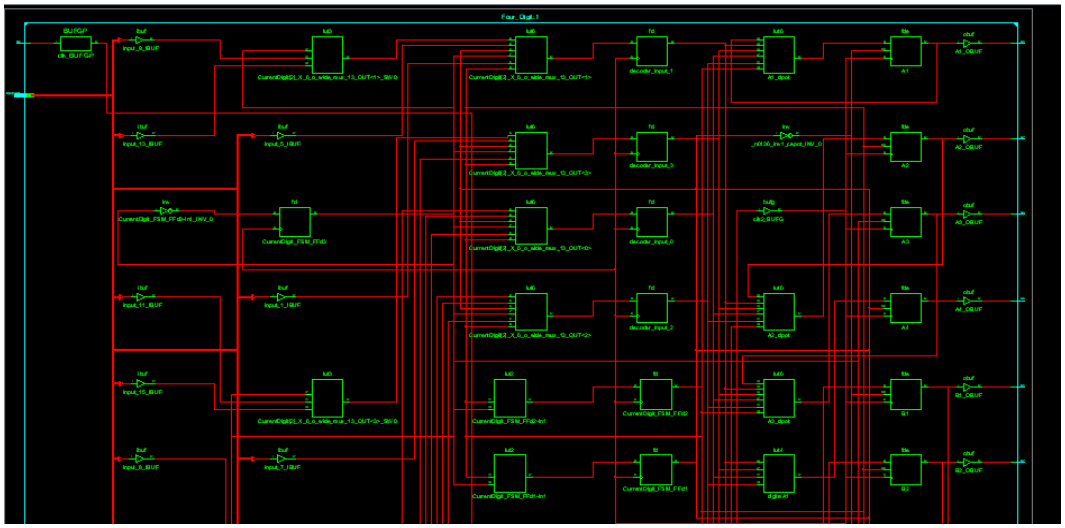
در تست بنچی که برای این پروژه در نظر گرفته شده است تمام اعداد از ۰ تا ۹۹۹۹ تولید شده و به bcd تبدیل شده است و بعد از گذر از دیکدر به 7segment تبدیل میشود و در دیجیت ها نمایش داده می‌شود. برای این امر یک لوپ طراحی شده است که تا ۹۹۹۹ می شمرد و عدد شمارنده را به یک تابع با ورودی اینتیجر می دهد و سپس این اینتیجر به bcd تبدیل می‌شود و بعد به کد اصلی داده می‌شود و اعداد نمایش داده می‌شوند.

```vhdl
-- Function to convert decimal to BCD
function decimal_to_bcd(decimal_num : integer) return std_logic_vector is
   variable bcd_num : std_logic_vector(15 downto 0);
   variable temp : integer := decimal_num;
begin
   bcd_num := (others => '0');
   for i in 0 to 3 loop
      bcd_num(i*4+3 downto i*4) := std_logic_vector(to_unsigned(temp mod 10, 4));
      temp := temp / 10;
   end loop;
   return bcd_num;
end function;
```

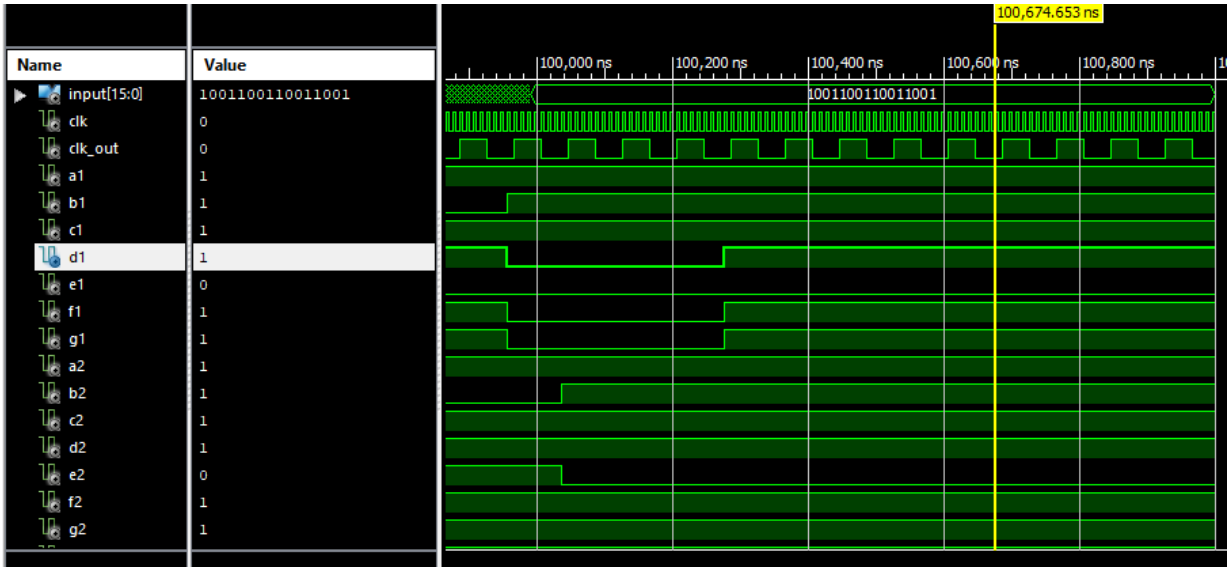
تابع بالا عمل تبدیل عدد اینجیر به bcd را انجام می‌دهد.



تصویر بالا نمایش دهنده بخشی از RTL ماژول می باشد.



تصویر بخشی از نقشه تکنوژی ماژول طراحی شده



تصویر تست بنچ ماژول می‌باشد که عدد ۹۹۹۹ را در حالت bcd نمایش میدهد برای تمام دیجیت ها فقط ای ای دی e خاموش می باشد که نمایش دهنده عدد ۹ است.

در طراحی این ماژول نام گذاری خروجی ها به صورت تصویر زیر انجام شده است.