ازمایش هشتم

موضوع: تغییر عدد 7seg با استفاده از کلید ها و تغییر نورled با توجه به آن

تاریخ آزمایش: ۱۴۰۲/۹/۲۲

استاد: مهندس جوادی

جواد فرجی (۹۹۵۲۲۰۰۵)

محمد رحمانی (۹۷۵۲۱۲۸۸)

ورودی و خروجی:

کلاک که همیشه هست! دو ورودی برای تغییر عدد و دو خروجی یکی برای نمایش عدد 7seg و دیگری برای انتخاب این که کدام 7seg روشن باشد. یک خروجی برای led

```
entity ex8 is
    port( gclk : in std_logic;
                up : in std_logic;
                down : in std_logic;
                seg_sel : out std_logic_vector (4 downto 0);
                seg_data : out std_logic_vector (6 downto 0);
                LED : out std_logic_vector (7 downto 0));
end ex8;
```

پروسس ها:

- **پروسس برای تغییر عدد با استفاده از کلید ها:**

برای این ازمایش نیاز است که دو پروسس جدید داشته باشیم که با تغییر کلید ها صدا زده میشوند. و با توجه به مقدار متغیری که داریم عدد را تغییر دهند. برای مثال عدد نباید از ۲۵۶ بالاتر برود و همچنین با هر بار فشردن کلید، عدد ده واحد کم یا زیاد میشود. این کار با استفاده از چند شرطی که در تصویر زیر است انجام شده.

```
if (previous_up = '0') then
    previous_up := '1';
    if (counterseg3 = "0010") then
            if (counterseg2 = "0101") then
                counterseg3 := "0000";
                counterseg2 := "0000";
                number <= "00000000";
            else
                counterseg2 := counterseg2 + '1';
                number <= number + "00001010";
            end if;
    else
            if (counterseg2 < "1001") then
                counterseg2 := counterseg2 + '1';
            else
                counterseg2 := "0000";
                counterseg3 := counterseg3 + '1';
            end if;

        number <= number + "00001010";

    end if;
end if;
```

- **پروسس برای این که کدام یک از 7seg ها روشن باشد و نمایش داده شود، تا بتوانیم هر ۴ 7seg را با چشم به صورت روشن ببینیم**

برای این کار نیاز است که طی هر ۱۰ میلی ثانیه یک بار یکی از اعداد را روشن کنیم و این کار با اتسفاده از کلاک ۱۰ میلی ثانیه انجام میشود. مانند تمارینی که در ابتدای ترم داشتیم:

```
    variable RefreshSEG : integer range 0 to 4 := 0;
    begin
        if(rising_edge(CLK10MS)) then
            if RefreshSEG < 4 then
                RefreshSEG := RefreshSEG + 1;
            else
                RefreshSEG := 0;
            end if;
            case RefreshSEG is
                when 0 =>
                    SEG_SEL(4) <='0';
                    SEG_SEL(0) <='1';
                    SEG_DATA <= SEG_DATA_reg1;
                when 1 =>
                    SEG_SEL(0) <='0';
                    SEG_SEL(1) <='1';
                    SEG_DATA <= SEG_DATA_reg2;
                when 2 =>
                    SEG_SEL(1) <='0';
                    SEG_SEL(2) <='1';
                    SEG_DATA <= SEG_DATA_reg3;
                when 3 =>
                    SEG_SEL(2) <='0';
                    SEG_SEL(3) <='1';
                    SEG_DATA <= SEG_DATA_reg4;
                when 4 =>
                    SEG_SEL(3) <='0';
                    SEG_SEL(4) <='1';
                    SEG_DATA <= "0000000";
                when others => null;
            end case;
        end if;
    end process;
```

- **پروسه ای برای تغییر نور ال ای دی ها:**

دقیقا مانند آزمایش قبلی است فقط این بار متغیر توسط کلید ها تغییر میکند.

```
process(GCLK)
variable duty: std_logic_vector(7 downto 0) := "00000000";
begin
    if (rising_edge(GCLK))  then
        if (duty = "11111111") then
            duty := "00000000";
            --LED <= "00000000";
        end if;
        if (number > duty) then
            duty := duty + 1;
            LED <= "11111111";
        elsif(number = duty) then
            LED <= "00000000";
            duty := duty + 1;
        elsif(number < duty) then
            LED <= "00000000";
            duty := duty + 1;
        end if;
    end if;
end process;
```

- **انتقال اطلاعات به 7seg:**

این کار در پروسه انجام نمیشود و در بلاک اصلی برنامه است. برای هر ۴ عددی که داریم باید این کار را انجام دهیم:

```
end process;

with counterseg1sig select
    seg_data_reg1 <= "0111111" when "0000",
    "0000110" when "0001",
    "1011011" when "0010",
    "1001111" when "0011",
    "1100110" when "0100",
    "1101101" when "0101",
    "1111101" when "0110",
    "0000111" when "0111",
    "1111111" when "1000",
    "1101111" when "1001",
    "0000000" when others;
```

مپ کردن خروجی ها به روی fpga:

برای مپ کردن روی برد های fpga، با استفاده از داکیومنت موجود، این خطوط را داخل فایل ucf قرار میدهیم:

```
NET "gclk" CLOCK_DEDICATED_ROUTE = FALSE;
NET "up" CLOCK_DEDICATED_ROUTE = FALSE;
NET "down" CLOCK_DEDICATED_ROUTE = FALSE;
NET "gclk" LOC = P184;
NET "seg_data[0]" LOC = P10;
NET "seg_data[1]" LOC = P7;
NET "seg_data[2]" LOC = P11;
NET "seg_data[3]" LOC = P5;
NET "seg_data[4]" LOC = P4;
NET "seg_data[5]" LOC = P12;
NET "seg_data[6]" LOC = P9;

NET "seg_sel[0]" LOC = P15;
NET "seg_sel[1]" LOC = P20;
NET "seg_sel[2]" LOC = P19;
NET "seg_sel[3]" LOC = P18;
```

```
NET "seg_sel[4]" LOC = P16;

NET "up" LOC = P187;
NET "down" LOC = P185;

NET "LED[0]" LOC = P61;
NET "LED[1]" LOC = P62;
NET "LED[2]" LOC = P63;
NET "LED[3]" LOC = P64;
NET "LED[4]" LOC = P65;
NET "LED[5]" LOC = P67;
NET "LED[6]" LOC = P68;
NET "LED[7]" LOC = P71;
```

اجرای برنامه بر روی برد fpga:

1. Synthesize
2. Implement design
3. Generate programming

در این سه مرحله گزینه run را میزنیم و در صورتی که مشکل خاصی در برنامه وجود نداشته باشد و به باگ نخوریم به مرحله بعد می‌رویم.

4. Impact

با استفاده از این برنامه، فایل باینری ساخته شده را به programmer انتقال میدهیم و programmer این برنامه را روی بردهای fpga اجرا میکند.