ازمایش دوم

موضوع: شمارنده ۴ رقمی

تاریخ آزمایش: ۱۴۰۲/۷/۲۶

استاد: دکتر جوادی

جواد فرجی (۹۹۵۲۲۰۰۵)

محمد رحمانی (۹۷۵۲۱۲۸۸)

ورودی و خروجی:

```
gclk : in std_logic;
reset : in std_logic;
seg_sel : out std_logic_vector (4 downto 0);
seg_data : out std_logic_vector (6 downto 0));
```

ورودی ساعت، یک بیت برای ریست، ۵ بیت برای این که کدام یک از 7seg ها روشن شود. ۷ بیت برای دیتایی که قرار است در 7seg نشان داده شود.

سیگنال های موجود در مدار:

```
signal CLK10MS, TIMER1S: std_logic;
signal seg_data_reg1,seg_data_reg2,seg_data_reg3,seg_data_reg4:
std_logic_vector(6 downto 0);
signal counterseg1sig,counterseg2sig,counterseg3sig,counterseg4sig :
std_logic_vector(3 downto 0);
```

دو سیگنال برای ساعت، یکی برای رفرش کردن 7seg و یکی برای شمارنده ثانیه. ۴ سیگنال دیتا برای هر کدام از 7seg ها. ۴ سیگنال برای ذخیره عدد مورد نظر هر 7seg

شرط های آپدیت کردن 7seg: (در بدنه بلاک قرار دارد و پروسه نیست)

۴ شرط برای ۴ تا 7seg داریم. که در هر لحظه و به صورت موازی اجرا می شوند و خروجی هر کدام را تعیین میکنند. این ۴ شرط شبیه هم بوده و به این صورت نوشته می شوند. به ازای هر رقمی که در شمارنده ها ذخیره شده باشد. این اطلاعات تغییر میکنند.

```
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;
```

نمایش 7seg ها: (process)

باید با استفاده از gclck کاری کنیم که هر کدام از 7seg به صورت دوره ای روشن خاموش شوند، به طوری که fps آنها مقداری باشد که چشمک زدن چراغ ها با چشم دیده نشود. برای این کار هر ۱۰ میلی ثانیه یک بار (تقریبا باید صبر کنیم تا کلاک اصلی ۸۰ هزار بار تریگر شود) یکی از چراغ ها را روشن کرده و خاموش میکنیم.

نمایش اطلاعات هر 7seg:(process)

برای این کار نیاز است هر ۱ ثانیه یک بار اطلاعات را اپدیت کنیم. برای این کار هر ده میلیون بار که کلاک تریگر شود یک بار شروط لازم برای اپدیت شدن اعداد را به صورت if های تو در تو کنترل میکنیم. برای مثال اگر 7seg ها را به صورت زیر نشان دهیم.

```
0000 0000 0000 0000
```

که بیانگر عدد صفر هستند. بعد از یک ثانیه باید به صورت زیر تغییر کنند:

```
0000 0000 0000 0001
```

- نکته: برای دو پروسه بالا، نیاز است که یک شمارنده داشته باشیم که وقتی به عدد مورد نظر در ساعت رسیدیم. کد اجرایی برنامه را اجرا کنیم. در غیر این صورت، شمارنده را یکی افزایش میدهیم. این کار را میتوانیم یا در همین پروسه انجام دهیم. یا به صورت جدا یک متغیر داخلی تعریف کنیم و در پروسه دیگری انجام دهیم در این صورت باید dependency های پروسه های بالا را، برابر شمارنده بگذاریم. مانند روبرو

```
Process (counter)
Begin
…
End
```

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 "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;
```

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.