به نام خدا

گزارش کار پروژه

نام ونام خانوادگی:فروغ افخمی

شماره دانشجویی:9823006

استاد:زهرا زارع

بخش اول:

ابتدا یک کلاک و یک up_down به صورت ورودی تعریف میکنیم.

چون counter باید با فرکانس یک پنجم فرکانس کلاک تغییر کند در نتیجه با استفاده از تکنیک clkdivider یک کلاک دیگر(clk_5) که پریود ان 1/5 کلاک اولیه(clk) و duty cycle ان 50% است ایجاد میکنیم.

با توجه به اینکه برای مقادیر فرد هستند دو counter_1 تعریف میکنیم یکی لبه های بالا روند کلاک را میشمارد یک counter_2 لبه های پایین رونده را میشمارد .مثلا برای تقسیم بر 5 به ازای 2 لبه clk ورودی ds_1را یک میگذاریم و به ازای 2 clk مقدارش را صفر میگذاریم.ds_1 با لبه بالارونده میشمارد یعنی هر لبه بالارونده ای بیاید counter_1 یکی اضافه میشود.و ds_1 کلاک تغییراتش 1/5 کلاک ورودی است. ولی دیوتی سایکل 50٪ نیست.درنتیجه یک ds_2 نیز تعریف میکنیم که با لبه های پایین رونده‌clk را بشمارد و همانند ds_1 3تا لبه 1 باشد2 تا 0 .حال وقتی این دو را باهم or کنیم خروجی اصلی ایجاد میشود که فرکانسش 1/5 کلاک ورودی است و duty cycle اش هم 50٪ است.

برای اینکه Dff ها به لبه بالا رونده یا پایین رونده حساسند پس لازم است دو process تعریف کنیم.

در کد برای هر clk divider باید دو سیگنال ds و دو سیگنال counter تعریف کنیم.counter با یک شروع میشود.

Process اول حساس به لبه بالا رونده clk است.counter_1 از یک شروع میکند وقتی 1 است در else اخر هستیم و مقدار ds_1را 1 میگذاریم. تا زمانی که مقدار counter_1به2 برسد ds_1 را صفر میکنیم. Counter_1 میرود تا برسد به5 وقتی به 5رسید ds_1 را تغییر میدهیم به 1 و counter_1 را یک میکنیم.

برای ds_2 نیز به همین شکل است فقط حساس به لبه پایین رونده کلاک است.یعنی اگر clk اتفاق افتاد و مقدارش 0 بود.

سپس ds_2 , ds_1 را با هم orمیکنیم و در سیگنال clk_5 قرار میدهیم.

Counter:

برای counter از clk_5 استفاده میکنیم و process ما به clk_5 و up_down وابسته است. به توجه به مقدار up_down قرار است شمارش به سمت بالا یا پایین باشد در نتیجه اینگونه عمل میکنیم. اگر up_down=0 باشد باید شمارش به سمت بالا باشد در نتیجه چک میکنیم اگر counter برابر مقدار باینری عدد99 بود ان را صفر میکنیم در غیر ان صورت یکی به ان اضافه میکنیم.

و در صورتی که up_down=1 باشد باید شمارش پایین رونده باشد چک میکنیم اگر counterبرابر 0 بود که ان را برابر مقدار باینری 99 میکنیم در غیر این صورت یکی از ان کم میکنیم.
Counter را به صورت سیگنال unsigned 7 بیتی با مقدار اولیه صفر تعریف کرده ایم.
Convert to decimal:
حال باید مقدار counter را به decimalتبدیل کنیم.
برای اینکار باید عدد 7 بیتی را به 8 بیت تبدیل کنیم در نتیجه با یک صفر سمت چپ اش کان کتنیت میکنیم.

و در binary_s ریخته شده است.

حال باید رقم های باینری این عدد باینری 8 بیتی را جدا کنیم.یک decimal 8 بیتی تعریف میکنیم.برای وقتی که باینری ورودی ما بین 0 تا 9 است یعنی کوچکتر از to unsigned(9,8) باشد خود bianary_s را به decimalمیدهیم. اگر بین 10 تا19 باشد باینری را باعدد 6 جمع میکنیم و به decimalمیدهیم.

همینطور ادامه میدهیم و با مضارب 6 جمع میکنیم . و ما عدد6 بیتی داریم در نتیجه تا 63 بیشتر نمیتوان شمرد. این عدد decimal ایجاد شده در 4 بیت پایینش رقم یکان عدد باینری ورودی قرار دارد و در 4 بیت بالا دهگان قرار دارد.

**to unsigned(9,8): یک عدد unsigned هشت بیتی با مقدار 9 تولید میکند.

set 2 MHz for enables and sevensegment:

حال باید بین دو seven segment تقسیم زمان کنیم با توجه به صورت سوال باید با فرکانس تغییرات 0 و 1 شدن en ها 2مگا هرتز باشد. یک counter تعریف میکنیم که در صورتی که کمتر از 50/2 لبه کلاک اتفاق بیفتد حالت را تغییر میدهد سون سگمنت را برابر 4 بیت بالای دسیمال قرار میدهیم و en2 رو فعال میکنیم و کانتر را اضافه میکنیم تا برسد به 50/2 وقتی به 50/2 رسید این بار 4 بیت پایین decimal را به سون سگمنت میدهیم و en1 را فعال میکنیم (2 en غیر فعال) و کانتر را اضافه میکنیم. وقتی counter به 50 رسید ان را ریست میکنیم(1 میکنیم.) و همین روند ادامه میابد.

عدد 50 از تقسیم فرکانس clk_5 بر فرکانس enableها به دست امده است یعنی:

حال باید عدد باینری خود را به ورودی سون سگمنت تبدیل کنیم در واقع دیکدر باینری به sevensegment باید بسازیم.این دیکدر ورودی اش seven_s است که seven_s به ازای 50/2 کلاک مقدارش برابر 4 بیت پایین decimalاست و به ازای 5000 کلاک دیگر برابر 4 بیت بالای دسیمال است.

که همانند تمرینات با توجه به اینکه که مقدار seven_s چه باشد اعداد باینری متفاوت به seven میدهیم.

Flag:

حال برای مقدار دهی به flag اینگونه عمل کرده ایم که processرا به clk_5و up_down حساس کرده ایم. با هر لبه بالا رونده clk_5 چک میکنیم اگر up_down=0 بود چک میکنیم اگر counterبه 25 رسیده بود f_25را 1 میکنیم و برای اینکه با کلاک بعدی 0 شود میتوان از شمارنده یا counter استفاده کرد چون counter باclk_5 تغییر میکند.در نتیجه وقتی counter=26 شد f_25 را صفر میکنیم.برای f_50هم از وقتی کانتر50 میشود ان را یک میکنیم و بعد از ان وقتی 51 شد ان را 0 میکنیم همینطور برای f_75.

اگر up_down=1 بود چک میکنیم اگر counterبه 25 رسیده بود f_25را 1 میکنیم و برای اینکه با کلاک بعدی 0 شود میتوان از شمارنده یا counter استفاده کرد چون counter باclk_5 تغییر میکند.در نتیجه وقتی counter=24 شد f_25 را صفر میکنیم.برای f_50هم از وقتی کانتر50 میشود ان را یک میکنیم و بعد از ان وقتی 74 شد ان را 0 میکنیم همینطور برای f_75.

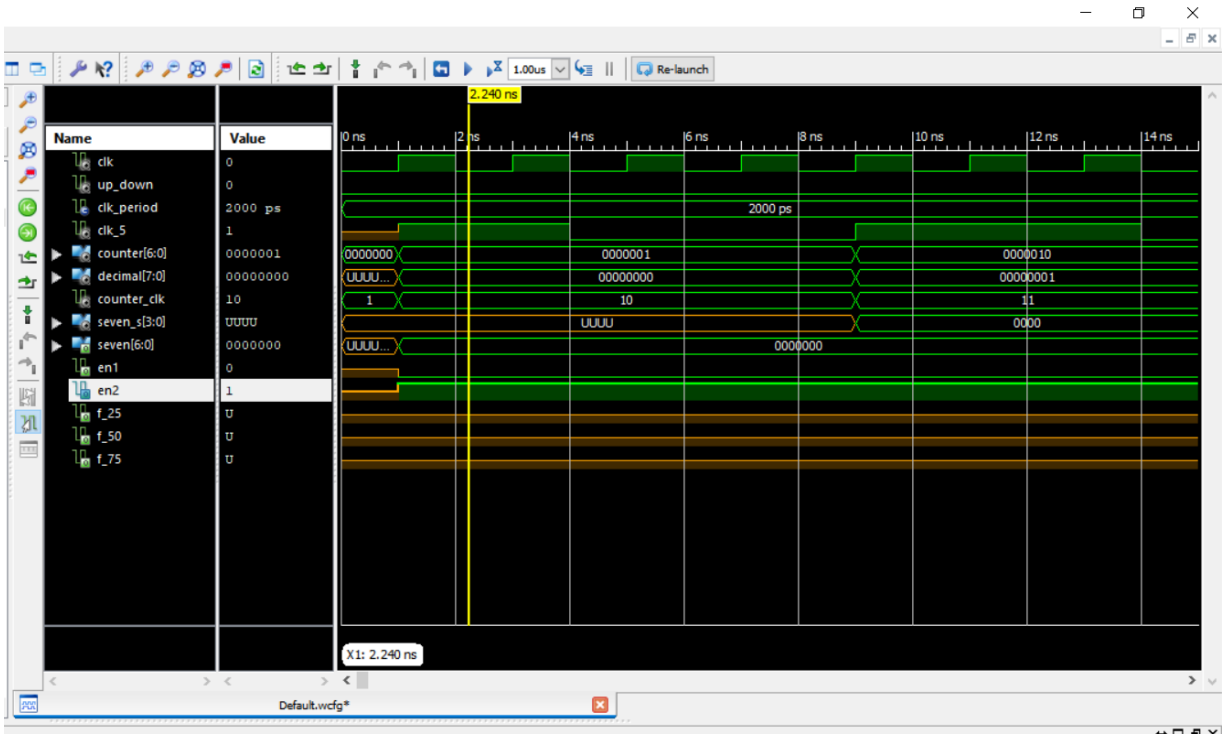
در تست بنچ نیز دوره تناوب کلاک را 2ns میگذاریم.

```
constant clk_period : time := 2 ns;
```

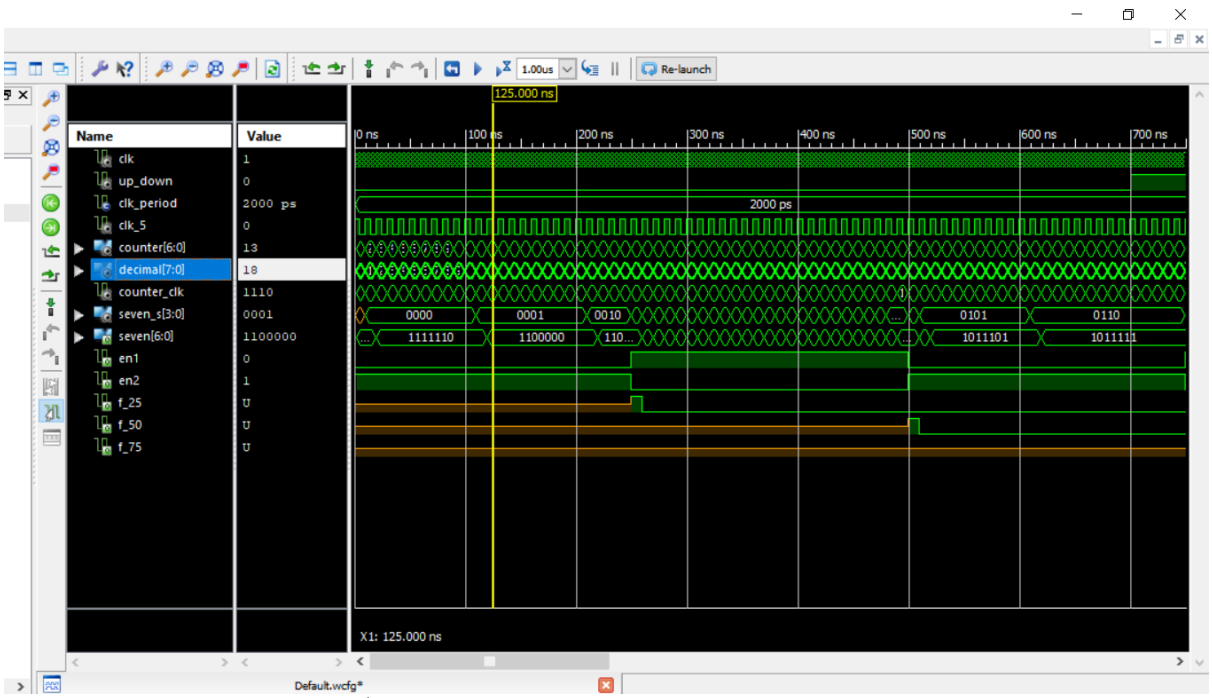
و با توجه به صورت سوال زمان تغییر از up_down 0 به 1 هم 700ns تنظیم میکنیم.

**فایل تست بنچ کل برنامه نیز به اسم testcounter میباشد.

Test bench:

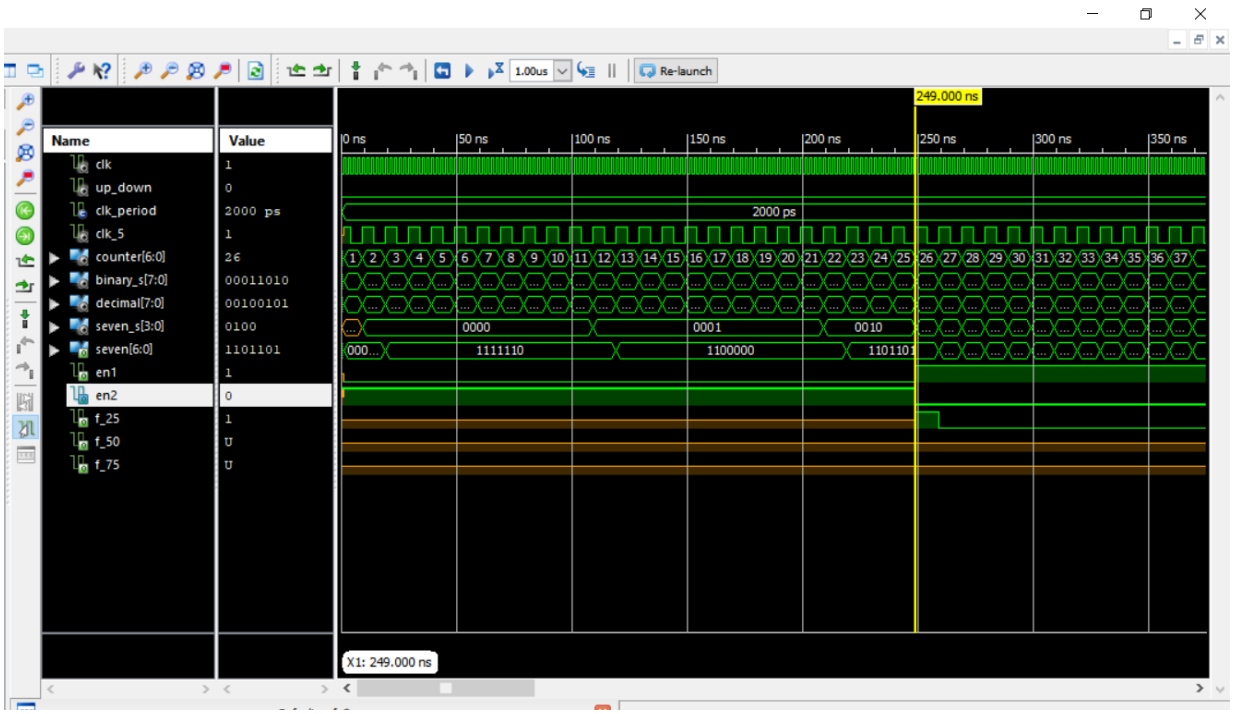


همانطور که مشاهده میکنیم clk_5 فرکانسش 1/5 clk ورودی است.
زمانی که up_down صفر است کلاک در حال شمارش به سمت بالاست.
Decimal به دلیل اینکه process اش با کلاک همراه هست یک کلاک دیرتر از counter است.



هر پریود en ها 500ns میباشد که برابر است با فرکانس تغییرات 2مگاهرتز و خواسته مسیله براورده شده است.

Seven سگمنت نیز در حال شمارش است و عدد های دو رقمی به دوتا چهار بیتی تقسیم شده اند با زمان مشخص نمایش داده میشوند.



برای flag ها هم مثلا f_25 زمانی که کانتر25 شده است منتظر یک لبه ی کلاک مانده است و 1 شده است و یک کلاک بعد صفر شده است.

سوال 2:

الف:ساختار اول

ب:

چک کردن a و مقدار دهی out put را نباید حساس به لبه کلاک بنویسیم وباید کامبینیشنال تعریف کنیم.

چون این سیگنال های تعریف شده dsها هم حساس به لبه ی بالا رونده کلاک هست و هم در لبه های پایین رونده تغییرات دارد . اگر a را در لبه های بالا رونده کلاک مقدار دهی کنیم وقتی ds15 در لبه ی پایین رونده تغییر کرده سریع خروجی تغییر نمیکند و منتظر لبه بالارونده کلاک میماند بعد خروجی را تغییر میدهد درنتیجه خروجی کلاک دیوتی سایکلش 50 درصد نمیشود.یعنی انگار فقط یکی از ds15_1یا ds15_2 را به خروجی داده ایم.پس برای مقدار دهی به خروجی حساس به لبه کلاک نباید باشد.dsها خودشان سنکرون با کلاک هستندو اینجوری تولید میشوند.

ج:مشکل: a ممکن هست با کلاک سنکرون نباشد و تا این ورودی بیاید این سیگنال ها ds به خروجی متصل میشوند و منتظر کلاک نمی مانند.برای حل این مسئله a را یک لول رجیستر میکنیم.

تغییرات a را باید کند تر تغییرات باشد وگرنه خروجی را نمیتوان خوب مشاهده کرد.

اگر در تست بنچ تغییرات a با کلاک نباشد باعث میشود output با کلاک تغییر نکند و وسط کلاک مقدارش تغییر کند و کلاک قبلی و بعدی ما خراب شود.

درنتیجه یک a_S تعریف میکنیم و یک process مینویسیم و a را به a_S متصل میکنیم.

```
Process(clk)
Begin
If rising_edge(clk) then
a_s<=a;
end if;
end process;
```

که در صورت سوال هم وجود دارد.
حال مشکل برای a_s زمانی است که ما یک بخش کامبینیشنال داریم که نسبتا کند است هر بخش کد ما در یک clb پیاده سازی میشود وقتی این ها را با هم Or میکنیم در یک قسمت باهم or میشوند و خروجی به یه قسمت دیگر میرود که Mux را در ان پیاده سازی کرده ایم.خروجی ها ممکن است فاصله فیزیکی زیادی داشته باشند و وقتی فرکانس کاری ما بالاست تاخیر مسیر ها روی مدار تاثیر زیاد میگذارد.اگر مدار سنکرون با کلاک باشد ما مطمینیم قبل از اینکه لبه کلاک برسد خروجی ما رسیده است.اما در ساختار کامبینیشنال یک لبه کلاک می اید ds_1,2 مقدار میگیرند قبل از اینکه لبه کلاک بعدی بیاید باید مقدار دهی برای خروجی انجام شود در غیر این صورت مقدار ds ها ممکن است عوض شود و ما اطلاعات از دست میدهیم. درنتیجه در همان کلاکی که ds15_1,2 تولید میشوند باید به output هم منتقل شوند. و محدودیت زمانی برای ما ایجاد میشود و باید فرکانس کلاک را پایین بیاوریم.