بسم الله الرحمن الرحیم

گزارش تمرین سوم FPGA

مهیار عنصری ۹۶۳۲۰۹۳

توضیحات سوال اول بخش الف:

برای به دست آوردن تعداد در هر دقیقه (یا ثانیه) باید ابتدا زمان هر دور را بتوانیم به دست بیاوریم.

برای اینکه موتور یک دور بزند باید دو پالس تولید شود. از طرفی میدانیم مدت زمان دو پالس ۴ برابر مدت زمان یک بودن (یا صفر بودن) آن است چون dutycycle برابر ۵۰ درصد است. پس اگر بتوانیم به طریقی زمان یک بودن در یک پالس را به دست آوریم می توانیم زمان یک دور را از ۴ برابر کردن آن به دست بیاوریم.

برای این منظور در کد شرط می نویسیم که اگر در لبه مثبت clk پالس برابر یک بود به شمارنده یک واحد اضافه شود.

در پایان عدد شمارنده نشان می دهد چند برابر زمان یک کلاک پالس برابر با یک بوده است.

چون فرکانس کاری روی ۴۰ مگاهرتز است پس زمان هر کلاک ۲۵ نانوثانیه است و بنابراین زمان یک دور برابر با ۱۰۰=۴×۲۵ برابر عدد شمارنده با مقیاس نانوثانیه می شود. یعنی اگر شمارنده عدد ۱۵۰.۰۰۰ را نشان دهد زمان یک دور برابر با ۱۵.۰۰۰.۰۰۰ نانوثانیه یا ۱۵ میلی ثانیه بوده است.

Time of a round=۱۰۰ × $counter\ (in\ ns)$

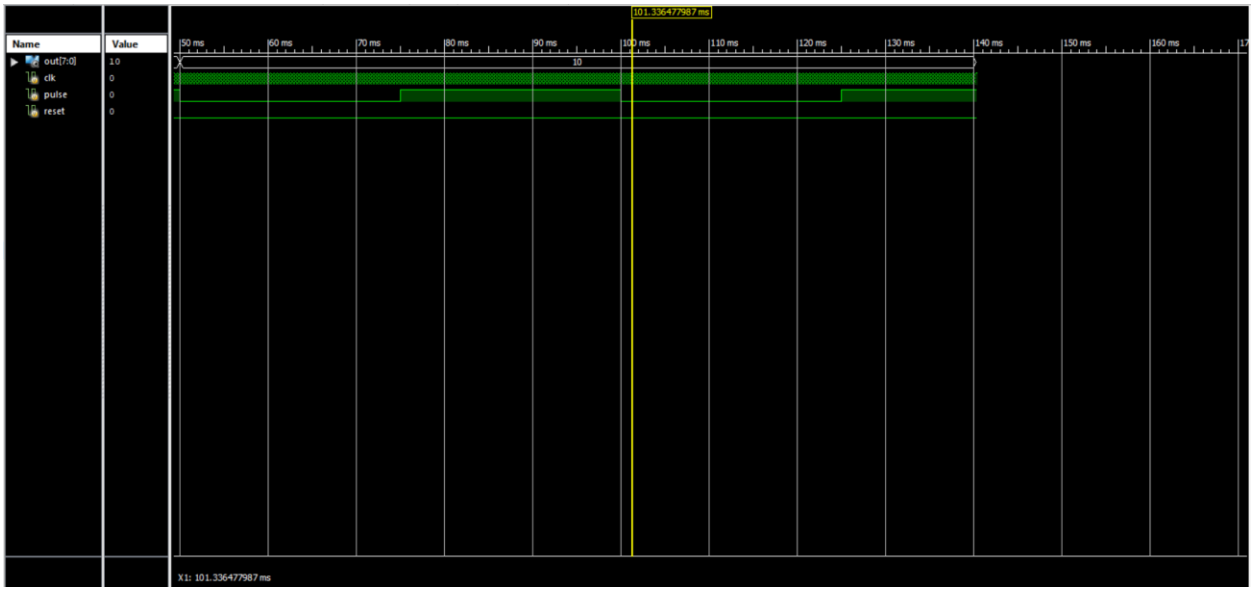
Number of rounds=۱s/Time of a round

=$۱۰^{۹}$/($۱۰^{۲}$ × Time of a round)

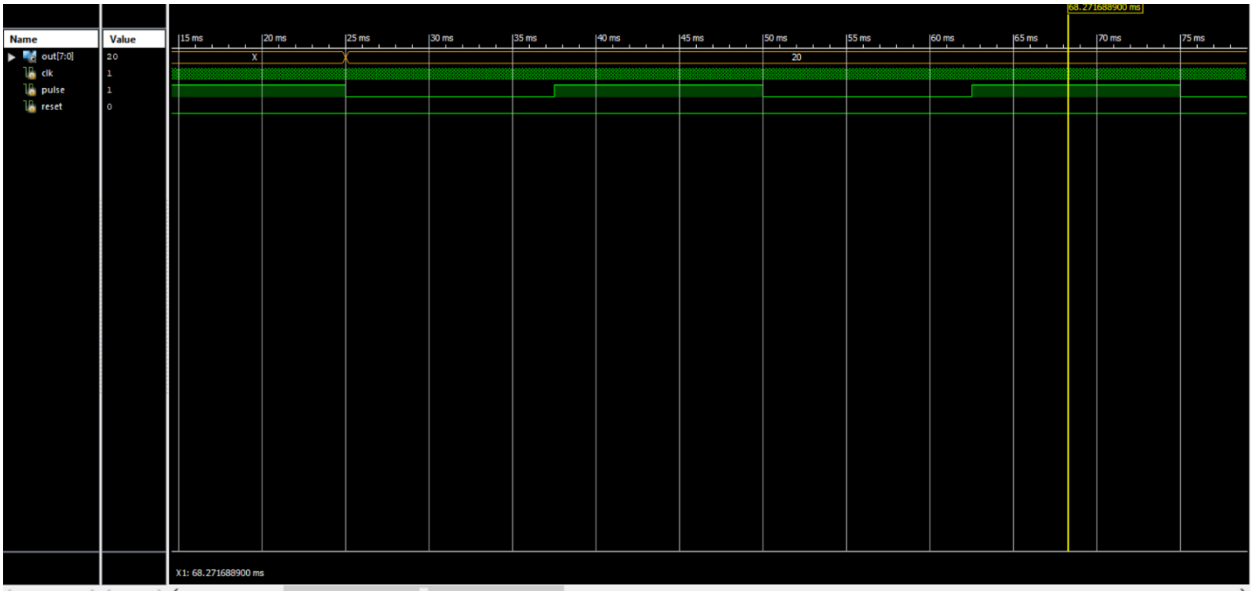
=۱۰۰۰۰۰۰۰/Time of a round

همچنین اگر پایه reset برابر یک شود همه چیز صفر می شود.

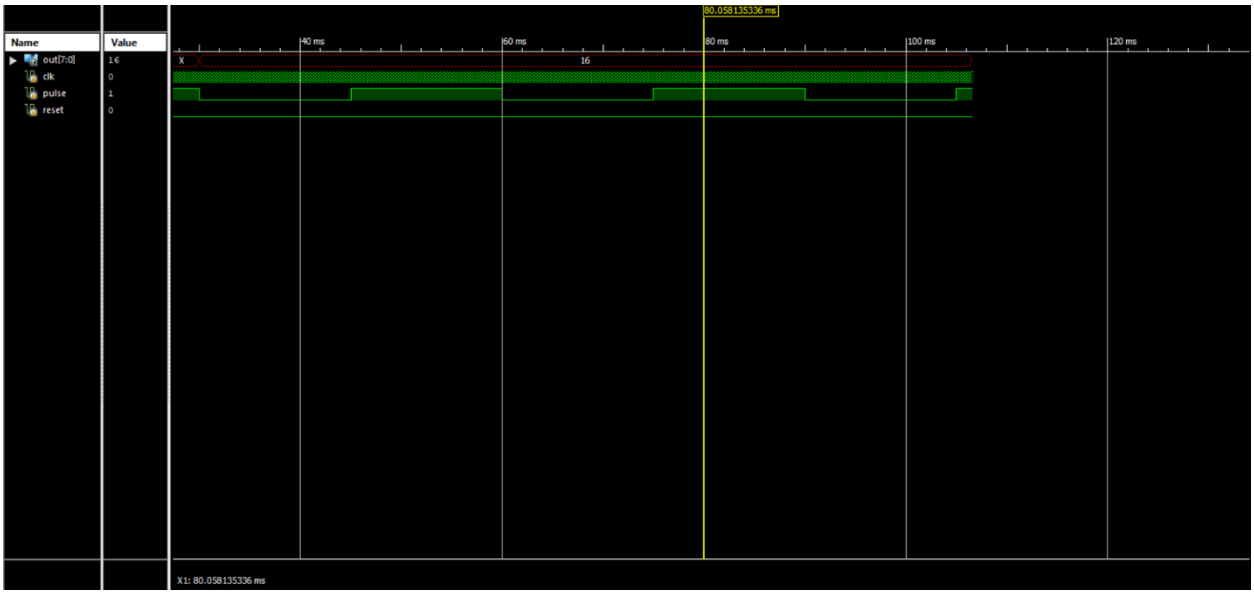
عکس های سوال اول بخش الف:



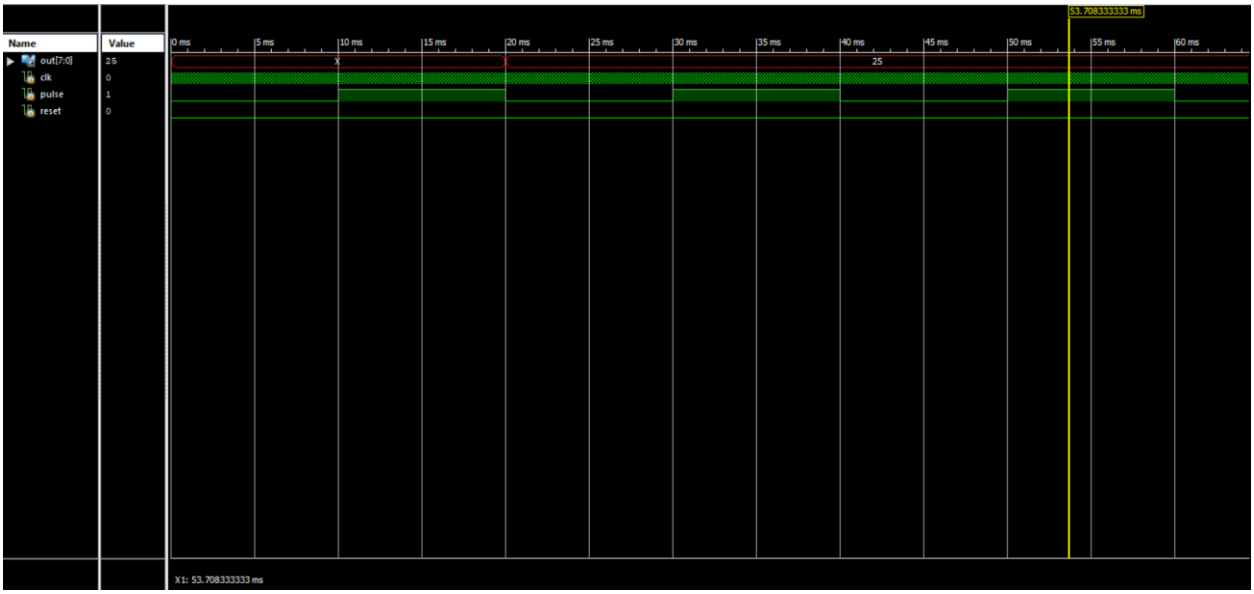
برای اینکه ۱۰ دور در ثانیه یا ۶۰۰ دور بر دقیقه داشته باشیم باید زمان هر دور ۰.۱ ثانیه یا ۱۰۰ میلی ثانیه باشد و مدت زمان یک بودن پالس ۲۵ میلی ثانیه است.



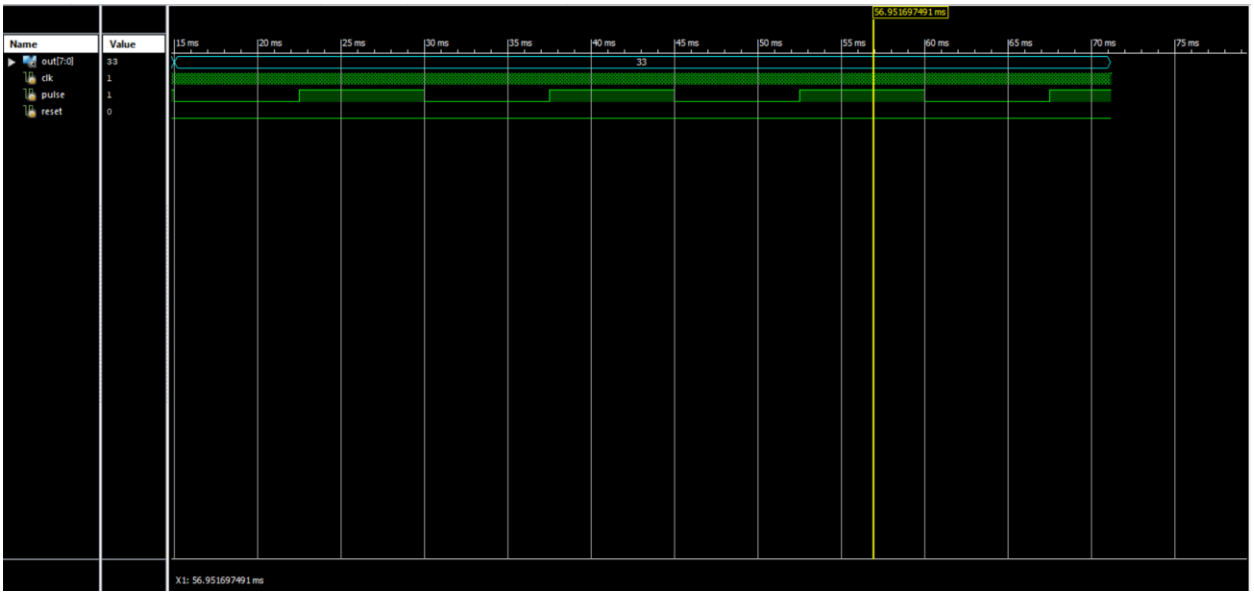
برای اینکه ۲۰ دور در ثانیه یا ۱۲۰۰ دور بر دقیقه داشته باشیم باید زمان هر دور ۰.۰۵ ثانیه یا ۵۰ میلی ثانیه باشد و مدت زمان یک بودن پالس ۱۲.۵ میلی ثانیه است.



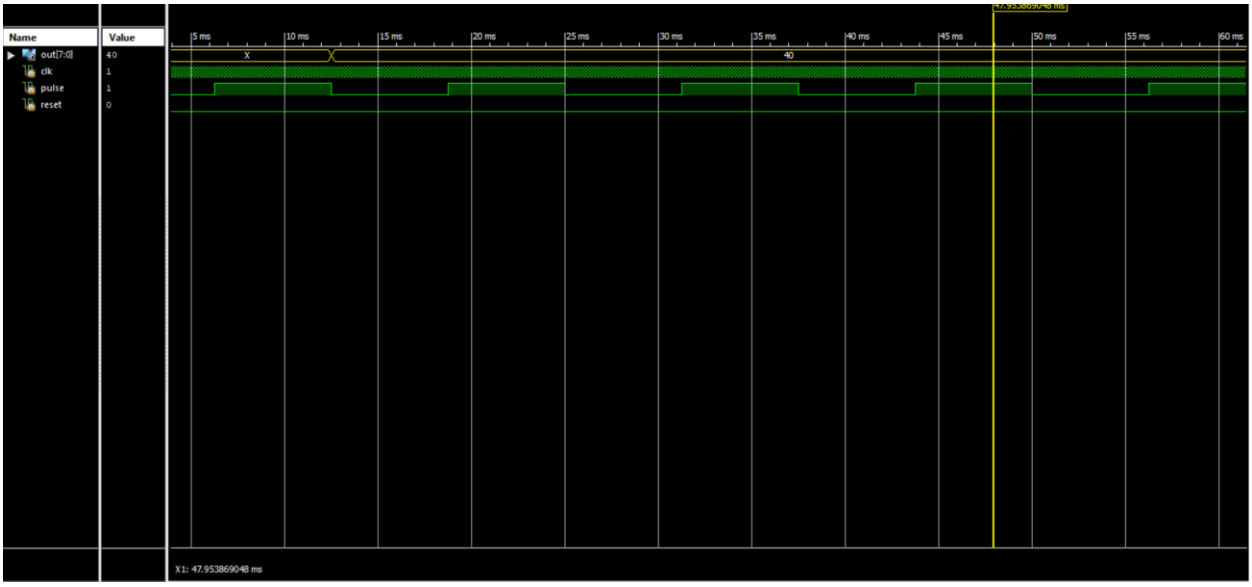
برای اینکه ۱۶.۶۷ دور در ثانیه یا ۱۰۰۰ دور بر دقیقه داشته باشیم باید زمان هر دور ۰.۰۶ ثانیه یا ۶۰ میلی ثانیه باشد و مدت زمان یک بودن پالس ۱۵ میلی ثانیه است.



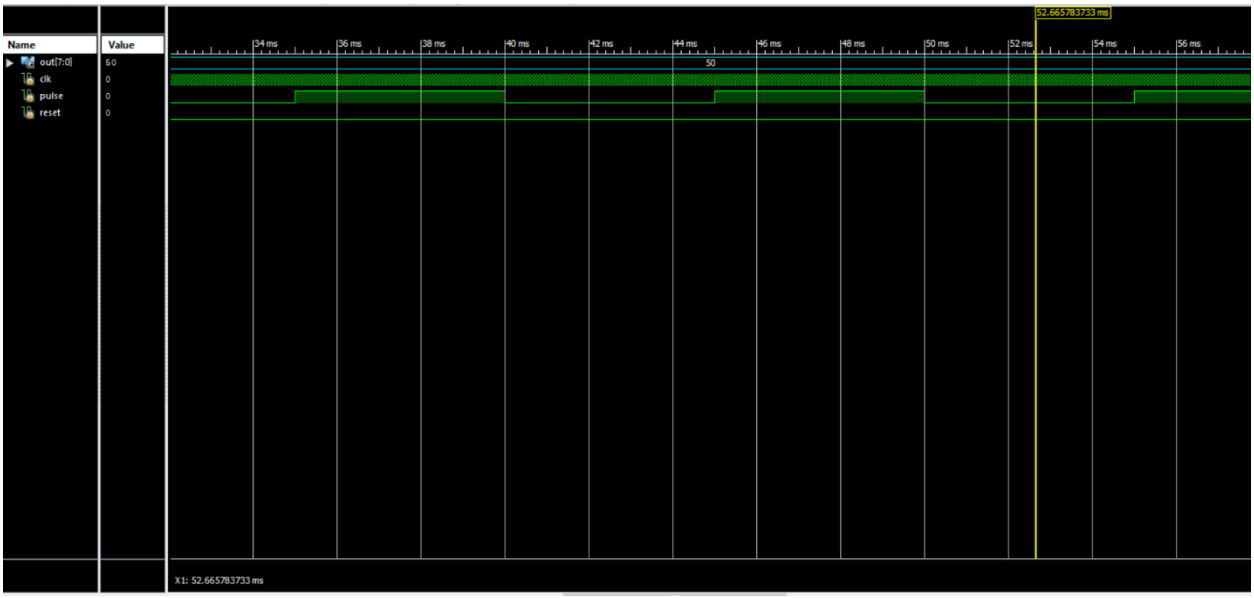
برای اینکه ۲۵ دور در ثانیه یا ۱۵۰۰ دور بر دقیقه داشته باشیم باید زمان هر دور ۰.۰۴ ثانیه یا ۴۰ میلی ثانیه باشد و مدت زمان یک بودن پالس ۱۰ میلی ثانیه است.



برای اینکه ۳۳.۳۴ دور در ثانیه یا ۲۰۰۰ دور بر دقیقه داشته باشیم باید زمان هر دور ۰.۰۳ ثانیه یا ۳۰ میلی ثانیه باشد و مدت زمان یک بودن پالس ۷.۵ میلی ثانیه است.



برای اینکه ۴۰ دور در ثانیه یا ۲۴۰۰ دور بر دقیقه داشته باشیم باید زمان هر دور ۰.۰۲۵ ثانیه یا ۲۵ میلی ثانیه باشد و مدت زمان یک بودن پالس ۶.۲۵ میلی ثانیه است.



برای اینکه ۵۰ دور در ثانیه یا ۳۰۰۰ دور بر دقیقه داشته باشیم باید زمان هر دور ۰.۰۲ ثانیه یا ۲۰ میلی ثانیه باشد و مدت زمان یک بودن پالس ۵ میلی ثانیه است.

برای اینکه ۱۰۰دور در ثانیه یا ۶۰۰۰ دور بر دقیقه داشته باشیم باید زمان هر دور ۰.۰۱ ثانیه یا ۱۰ میلی ثانیه باشد و مدت زمان یک بودن پالس ۲.۵ میلی ثانیه است.

می بینیم که با یک شدن reset همه چیز برابر با صفر شده و دوباره از ابتدا دور محاسبه گشته است.

توضیحات سوال اول بخش ب:

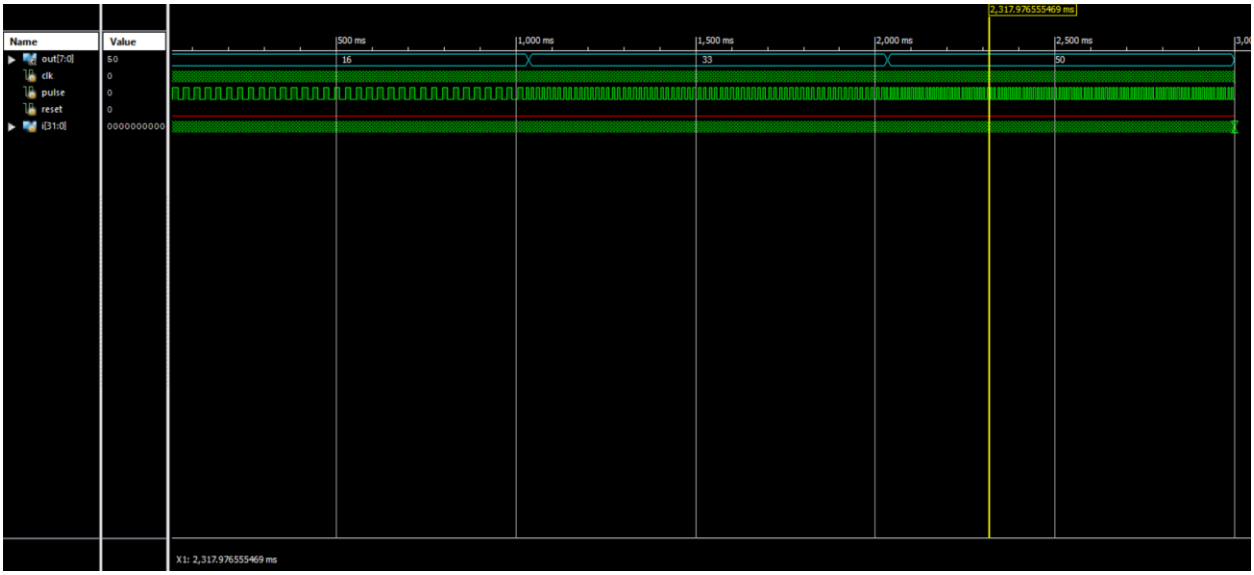
ابتدا محاسبه میکنیم چند بار باید هر پالس یک شود تا یک ثانیه از آن پالس داشته باشیم.

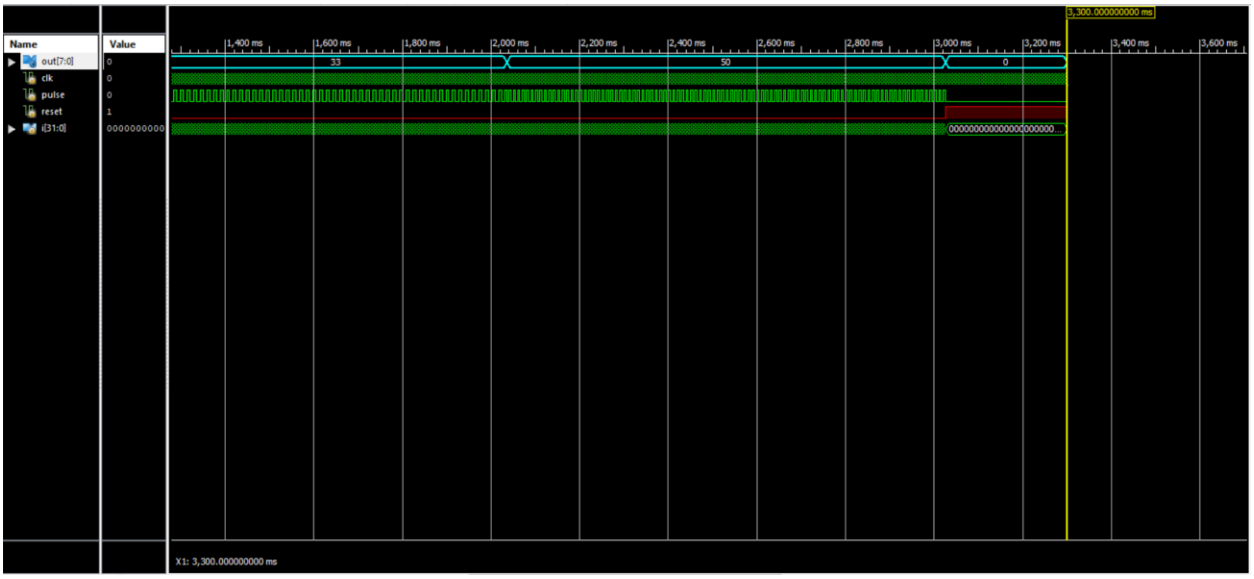
برای ۱۰۰۰ دور در دقیقه که هر دور برابر ۱۵ میلی ثانیه است باید ۶۷=۶۶+۱=۱۵÷۱۰۰۰ بار از این پالس داشته باشیم تا تقریبا یک ثانیه (یک ثانیه و پنج میلی ثانیه به طور دقیق) از این پالس داشته باشیم.

به طریق مشابه برای ۲۰۰۰ دور در دقیقه که هر دور برابر ۷.۵ میلی ثانیه است باید ۱۳۴=۱+۱۳۳=۷.۵÷۱۰۰۰ بار از این پالس داشته باشیم و برای ۳۰۰۰ دور در دقیقه که هر دور برابر ۵ میلی ثانیه است باید ۲۰۰=۵÷۱۰۰۰ بار از این پالس داشته باشیم.

پس یک شمارنده در حلقه for می گذاریم که از ۱ تا مجموع این سه پالس یعنی ۴۰۱ بشمرد و در ۶۷ شمارش اول هر ۱۵ میلی ثانیه یک بار پالس صفر و یک شود و در ۱۳۴ شمارش بعدی هر ۷.۵ میلی ثانیه یک بار پالس صفر و یک شود و ۲۰۰

شمارش آخر هم هر ۵ میلی ثانیه یک بار پالس صفر و یک شود. همچنین هر ۱۲.۵ میلی ثانیه یک بار clk را صفر و یک میکنیم تا فرکانس ۴۰ مگاهرتز را داشته باشیم.





عکس اول خواسته سوال و عکس دوم نشان دهنده درست بودن پایه reset است.

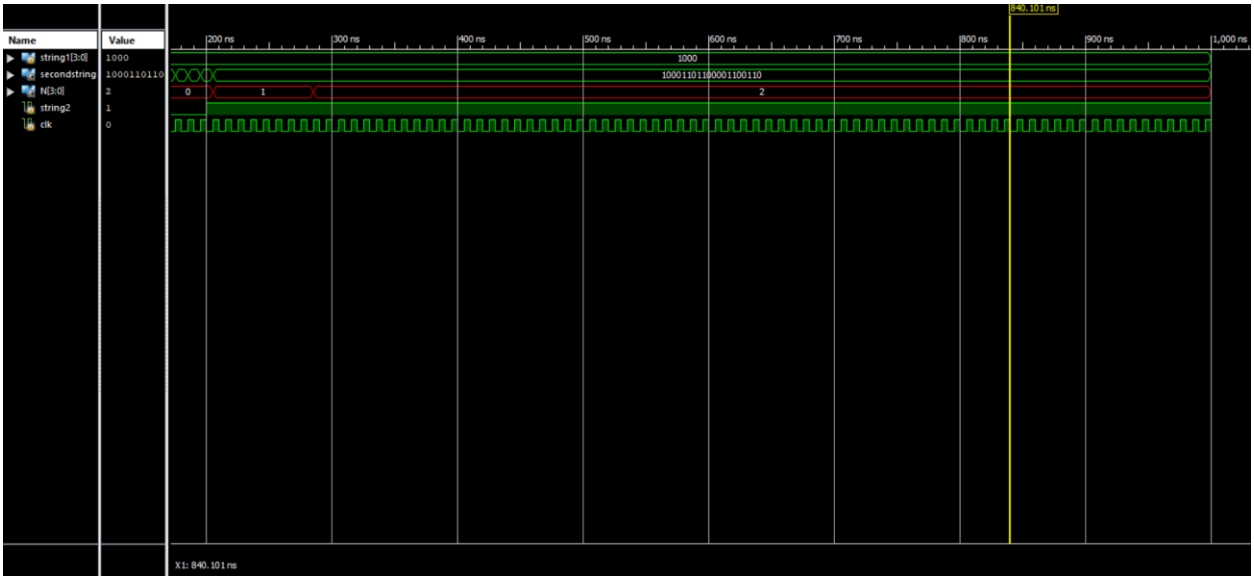
توضیحات سوال دوم:

برای هر دو بخش من ابتدا رشته دوم را با کمک یک شمارنده در یک register ۲۰ بیتی ذخیره می کنم.

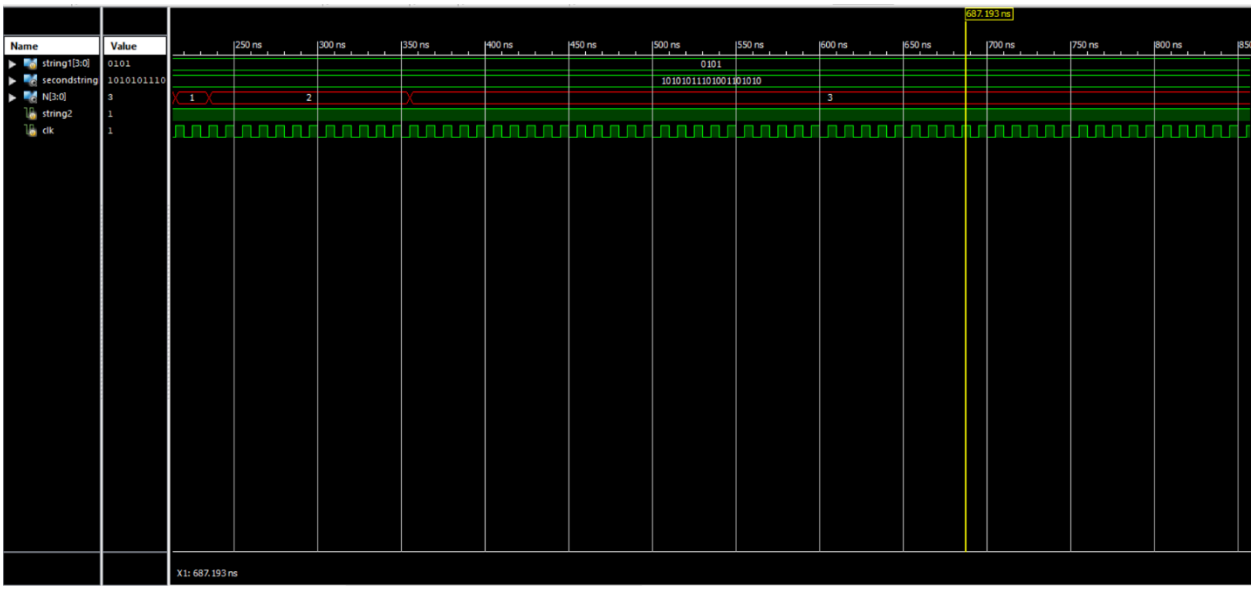
یک register کمکی ۴ بیتی تعریف میکنم و در ابتدا ۴ بیت سمت چپ register ۲۰ بیتی را در آن می ریزم.

در بخش الف یعنی با همپوشانی:

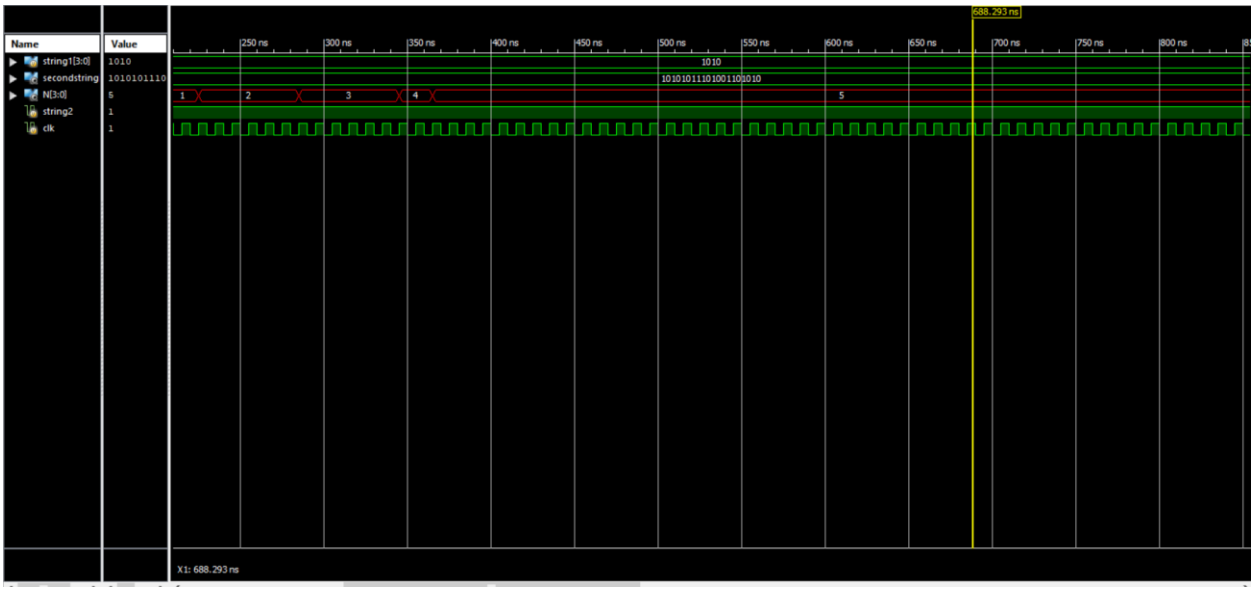
سپس با کمک یک شمارنده معکوس هر بار با رشته اول بررسی می شود و یکی از شمارنده کم می شود و در صورت تساوی یک واحد به جواب اضافه می شود.



رشته ۱۰۰۰ در ۱۰۰۰۱۱۰۰۱۱۰۱۰۰۰۱ تنها دو بار تکرار شده.



رشته ۰۱۰۱ در ۱۰۱۰۱۰۱۱۱۰۱۰۰۱۱۰۱۰۱۰۱ تنها سه بار تکرار شده.



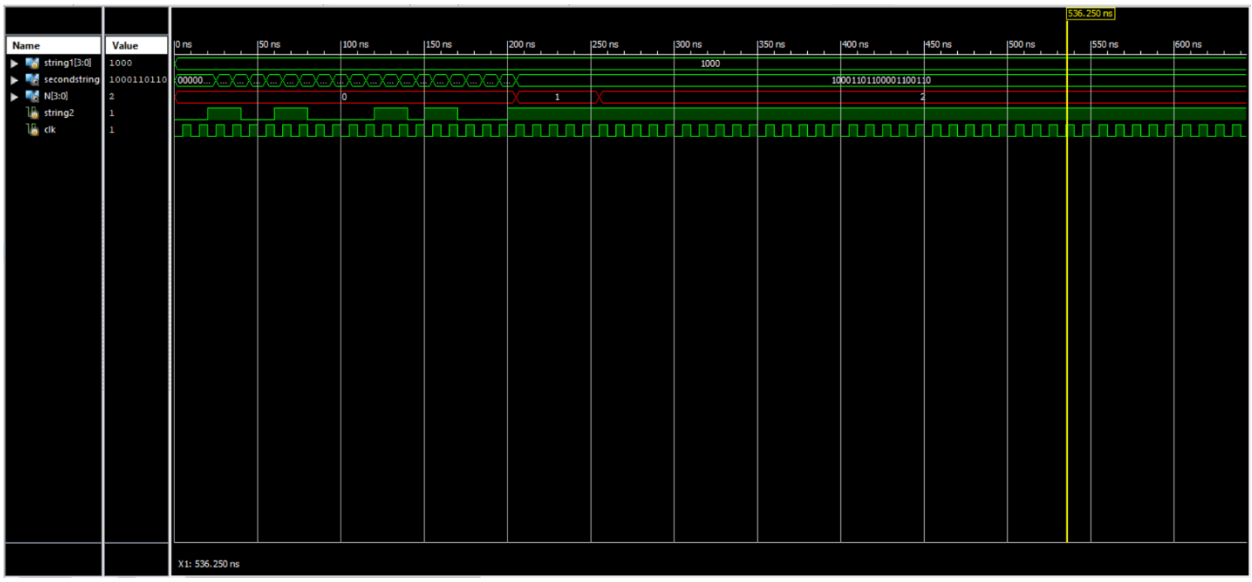
رشته ۱۰۱۰ در ۱۰۱۰۱۰۱۱۱۰۱۰۰۱۱۰۱۰۱۰۱ پنج بار تکرار شده است.

این هم داده صورت سوال است که به درستی ۴ بار تکرار رشته ۰۱۱۰ در رشته مدنظر را تشخیص داده است.

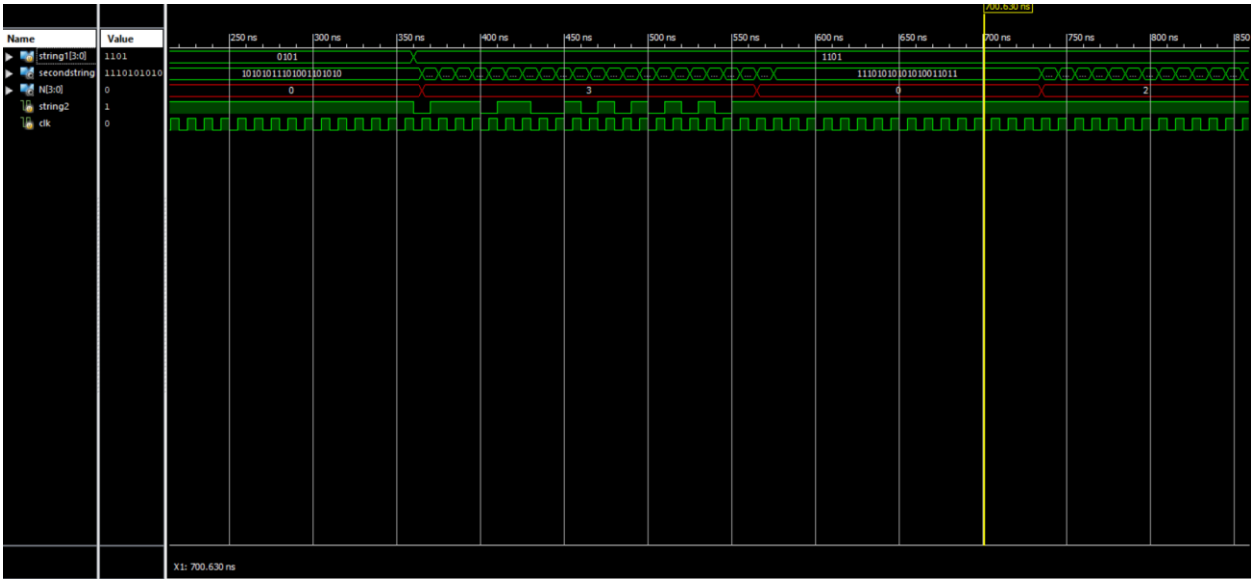
برای بخش ب یعنی بدون همپوشانی:

مراحل تقریبا شبیه حالت قبل است فقط اگر ۴ بیت درون register کمکی با رشته اول برابر بود از شمارنده معکوس ۴ واحد کم می شود (در حالت اول یک واحد کم می شود) اما اگر برابر نبودند مانند بخش قبل فقط یک واحد از شمارنده معکوس کم می شود.

علت کم شدن ۴ واحد این است که بین ۴ بیت کنونی که دارند مقایسه می شوند با ۴ بیت بعدی نباید هیچ اشتراکی باشد (چون همپوشانی نداریم) و باید ۴ بیت از شمارنده کم شود چون مثلا شمارنده به خانه ۱۵ تا ۱۸ اشاره کرده و اکنون باید به ۱۱ تا ۱۴ اشاره کند پس ۴ واحد باید از شمارنده معکوس کم شود.

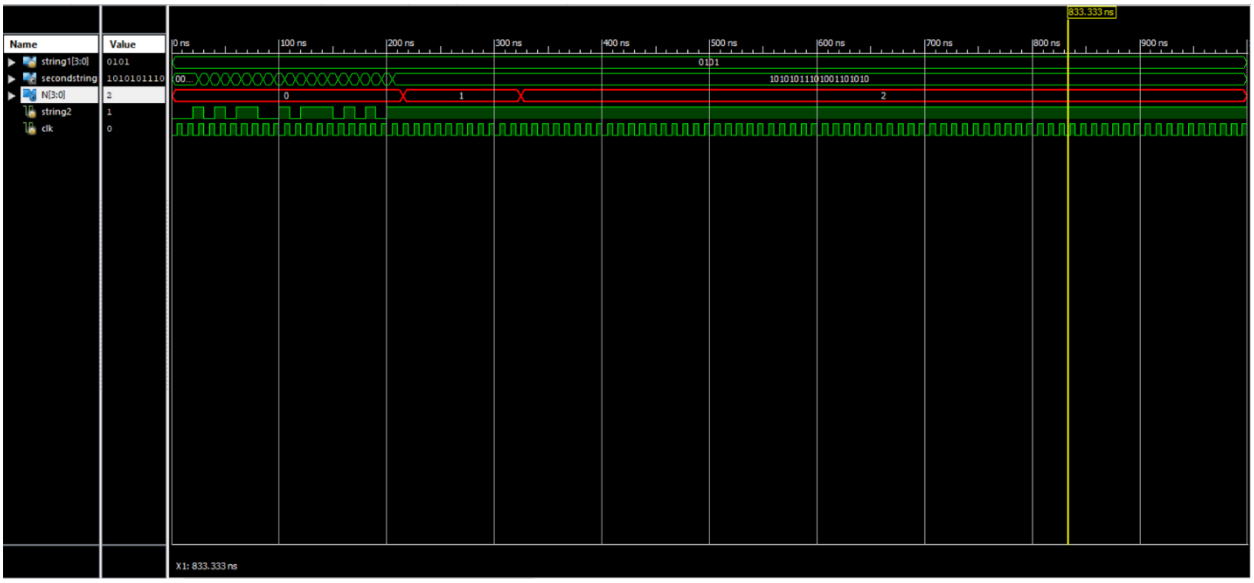


رشته ۱۰۰۰ در ۱۰۰۰۱۱۰۰۱۱۰۱۰۰۰۱۱۰۰۱۱۰ تنها دو بار تکرار شده.

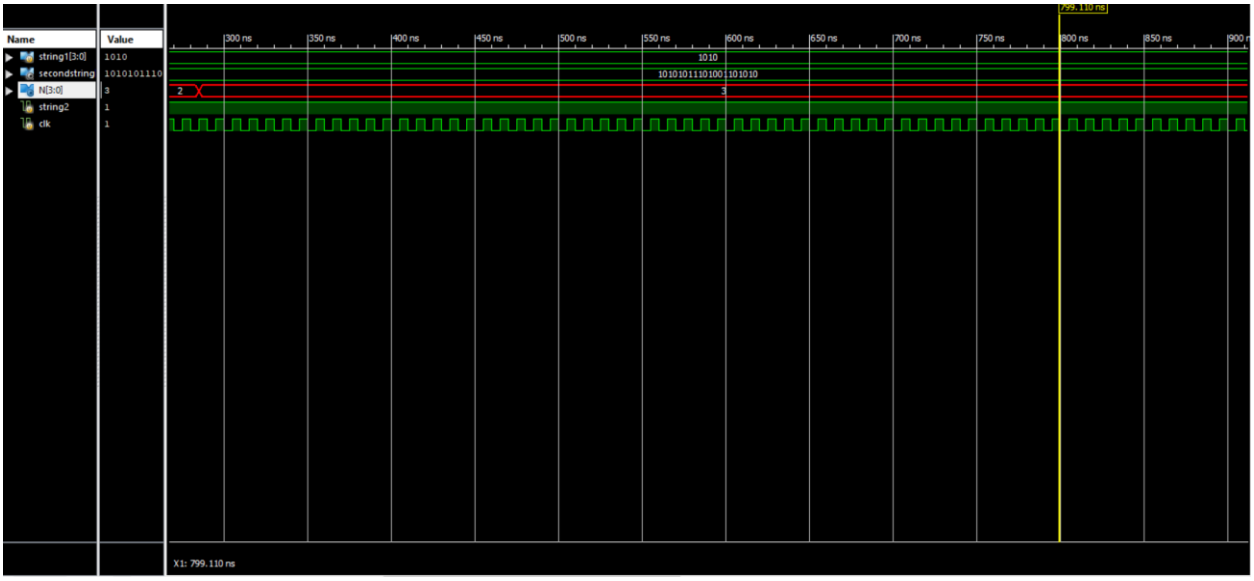


دادن دو ورودی پشت هم به ماژول در حالت الف و درست شمردن آن.

در حالت قبلی تفاوتی بین دو بخش نبود اما از این داده تفاوت ها را میبینیم.



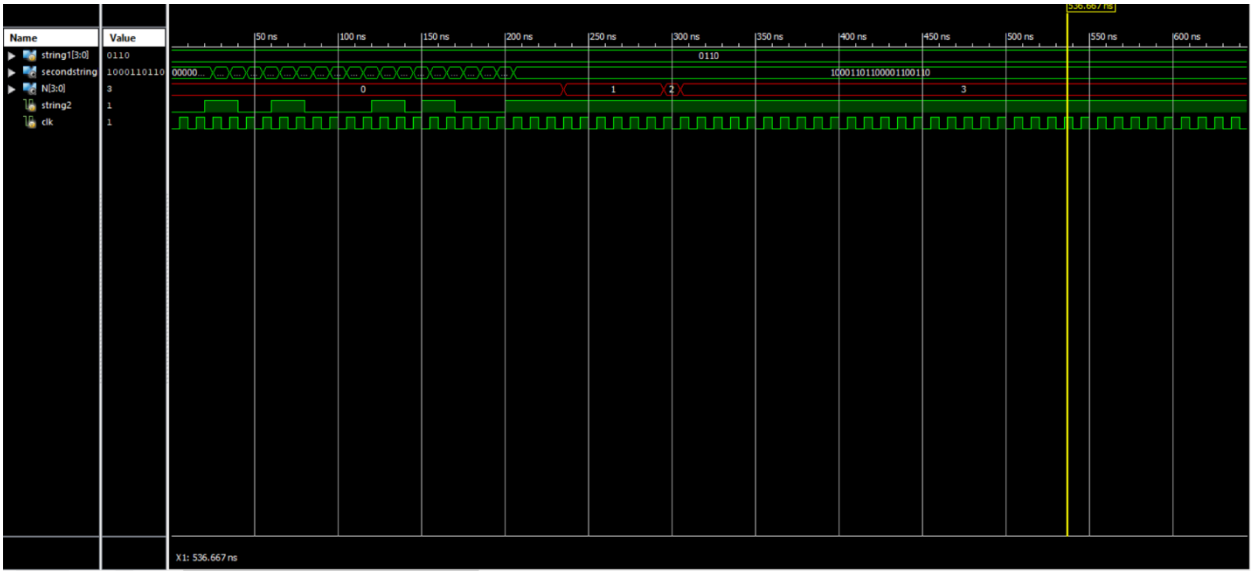
رشته ۰۱۰۱ در ۱۰۱۰۱۰۱۱۱۰۱۰۰۱۱۰۱۰۱۰۱۰ تنها دو بار تکرار شده و آن دو بیت آبی رنگ به دلیل عدم در نظرگرفتن همپوشانی حساب نمی شوند.



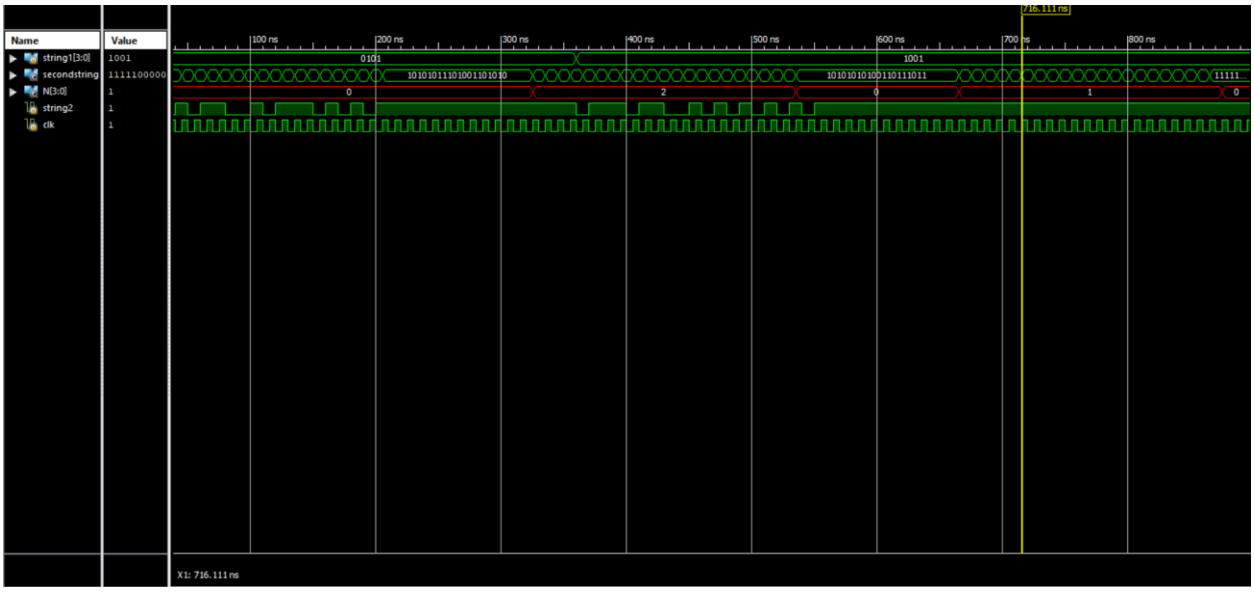
رشته ۱۰۱۰ در ۱۰۱۰۱۰۱۱۱۰۱۰۰۱۱۰۱۰۱۰ سه بار تکرار شده است.

دو بار تکرار ۱۰۱۰ که با رنگ آبی رنگ مشخص شده است به دلیل عدم در نظر گرفتن همپوشانی حساب نشده است و با حالت الف تفاوت دارد.

در صفحه بعد به سراغ داده صورت سوال می رویم:



مشاهده می شود که به مانند توضیحات صورت سوال رشته ۰۱۱۰ سه بار بدون در نظر گرفتن همپوشانی در رشته داده شده دیده می شود.



دادن دو ورودی پشت هم به ماژول در حالت ب و درست شمردن آن.

توضیحات سوال سوم:

در این سوال ابتدا باید از روی شکل متوجه روابط بین بیت های شیفت رجیستر شویم. بیت اول از سمت چپ از xor کردن بیت اول سمت راست و بیت دوم از سمت چپ از شیفت بیت اول به دست می آید. بیت سوم از سمت چپ از xor کردن بیت دوم و پنجم از سمت چپ و ورودی به دست می آید و دو بیت چهارم و پنجم از سمت چپ هم از شیفت دادن بیت های سمت راست خود به دست می آیند.

همچنین این ماژول یک پایه reset دارد که اگر برابر یک شود خروجی صفر می شود.

در ساخت تست بنچ متوجه شدم اگر فاصله تغییرات و دادن ورودی کم باشد خروجی خراب می شود و به تعداد ورودی خروجی نخواهیم داشت. اما وقتی که زمان

تغییرات و دادن ورودی های جدید به اندازه دو کلاک طول کشید مشکل حل شد و دیگر خروجی ها قاطی نشدند در بعضی از فیلم های مشاهده شده در اینترنت بیت ها از راست به چپ ارزش داشتند (پر ارزش ترین در راست) ولی در صورت سوال ظاهرا برعکس می باشد بنابراین من با هر دو حالت خروجی را مشاهده کردم و در دو پوشه جدا همراه گزارش قرار دادم. در حالت اول ابتدا باید بیت پرارزش و در حالت دوم بیت کم ارزش وارد شود.

کد هر دو حالت را قرار دادم و برحسب مدل که MSB در چپ یا راست باشد نام گذاری کردم.

حالتی که بیت سمت راست پر ارزش ترین بیت باشد.

حالتی که بیت سمت چپ پر ارزش ترین بیت باشد.

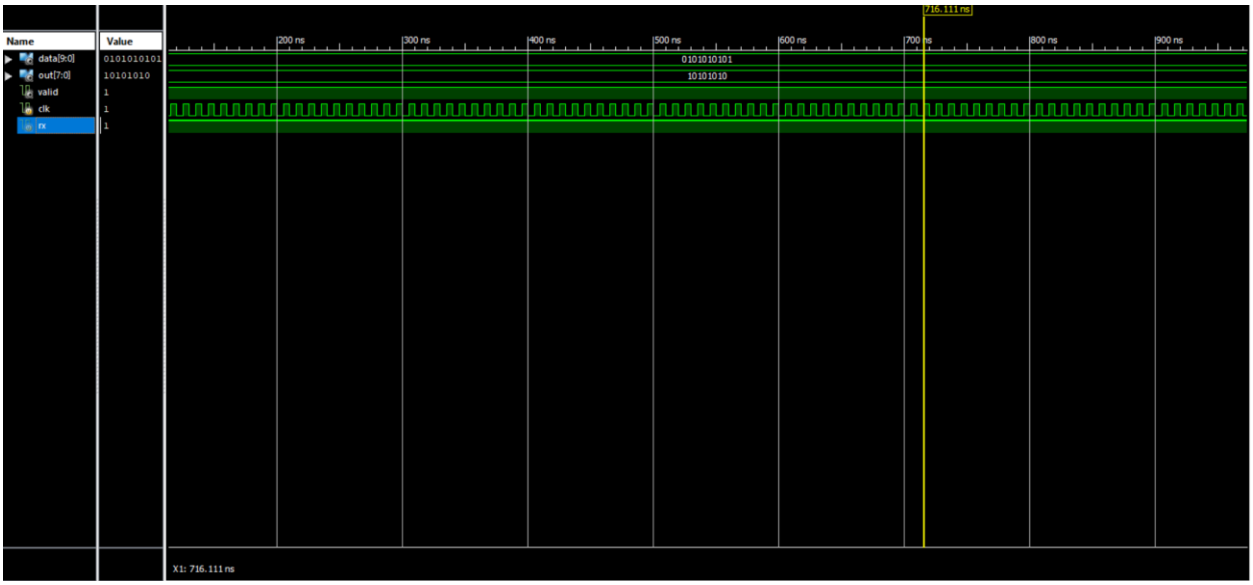
توضیحات سوال چهارم:

وقتی توضیحات و استاندارد ارتباط uart را خواندم و به این نتیجه رسیدم که در ارتباط uart به صورت پیش فرض بین ارسال کننده و گیرنده بیت یک در حال ارسال می باشد و قبل از ارسال هر بسته یک بیت صفر ارسال می شود تا گیرنده متوجه شود فرستنده قصد ارسال داده دارد. همچنین یک بیت به آخر ورودی اضافه می شود به عنوان بیت stop که همیشه برابر یک است.

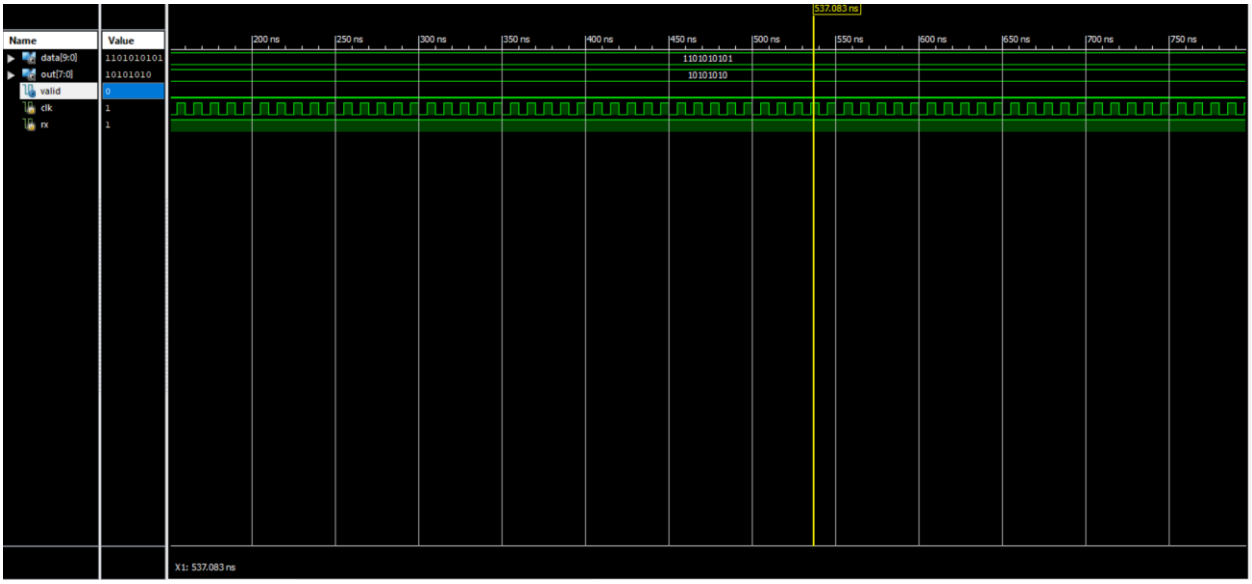
پس اگر قرار است ۸ بیت داده ارسال شود ۱۰ بیت ارسال می شود که بیت اول باید برابر صفر و بیت آخر باید برابر یک باشد.

در این ماژول ما ابتدا بیت به بیت داده را دریافت می کنیم و در یک register ۱۰ بیتی ذخیره می کنیم. سپس بیت اول و آخر را به صورت concatenate با ۰۱ مقایسه

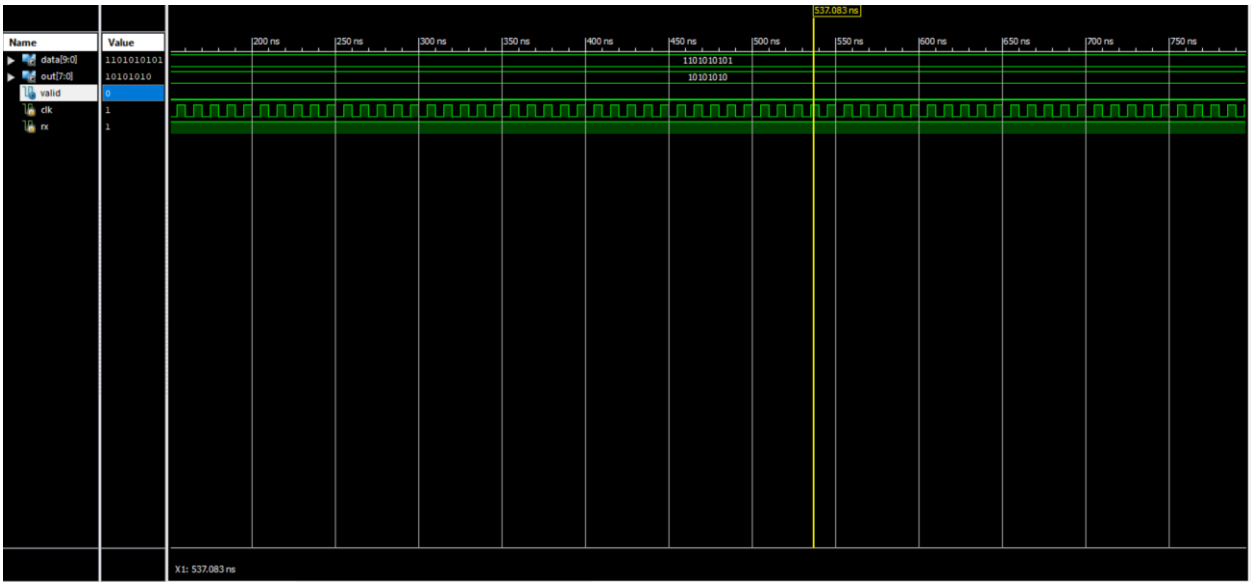
می کنیم اگر برابر بود آنگاه خروجی valid را یک میکنیم و ۸ بیت میانی را روی خروجی ۸ بیتی out قرار می دهیم.



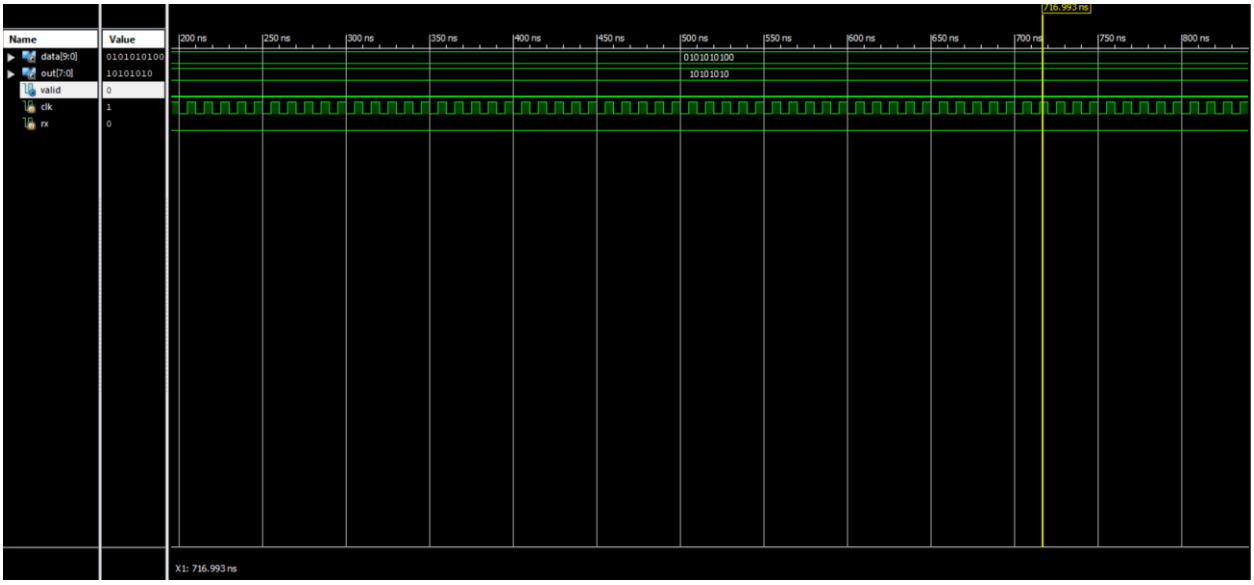
قواعد رعایت شده و valid برابر یک است و out هم برابر ۸ بیت میانی رشته دریافت شده است.



قواعد رعایت شده و valid برابر یک است و out هم برابر ۸ بیت میانی رشته دریافت شده است.



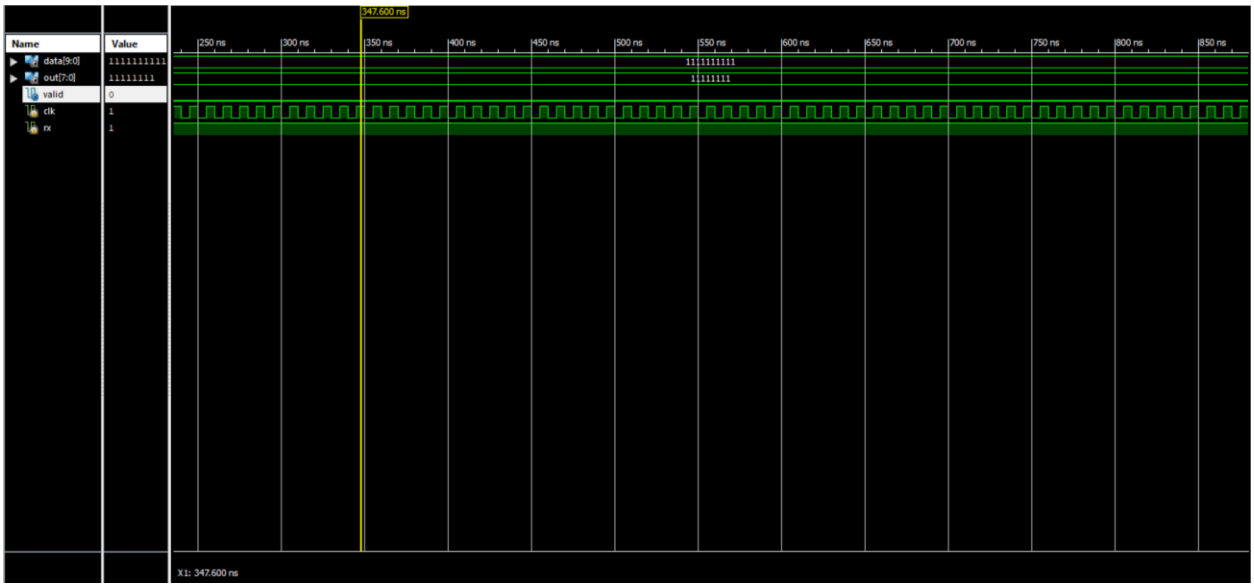
قواعد رعایت نشده و valid برابر صفر است و out هم برابر ۸ بیت میانی رشته دریافت شده است.



قواعد رعایت نشده و valid برابر صفر است و out هم برابر ۸ بیت میانی رشته دریافت شده است.

قواعد رعایت نشده و valid برابر صفر است و out هم برابر ۸ بیت میانی رشته دریافت شده است.

قواعد رعایت شده و valid برابر یک است و out هم برابر ۸ بیت میانی رشته دریافت شده است.



قواعد رعایت نشده و valid برابر صفر است و out هم برابر ۸ بیت میانی رشته دریافت شده است.

قواعد رعایت نشده و valid برابر صفر است و out هم برابر ۸ بیت میانی رشته دریافت شده است.