بسم الله الرحمن الرحیم

گزارش تمرین دوم FPGA

مهیار عنصری ۹۶۳۲۰۹۳

سوال اول:

نحوه کارکرد barrel shift register به این صورت است که بیت های خارج شده از یک سمت را به سمت دیگر وارد می کند یعنی مثلا با یک شیفت به راست LSB خارج شده و به جای MSB قرار می گیرد.

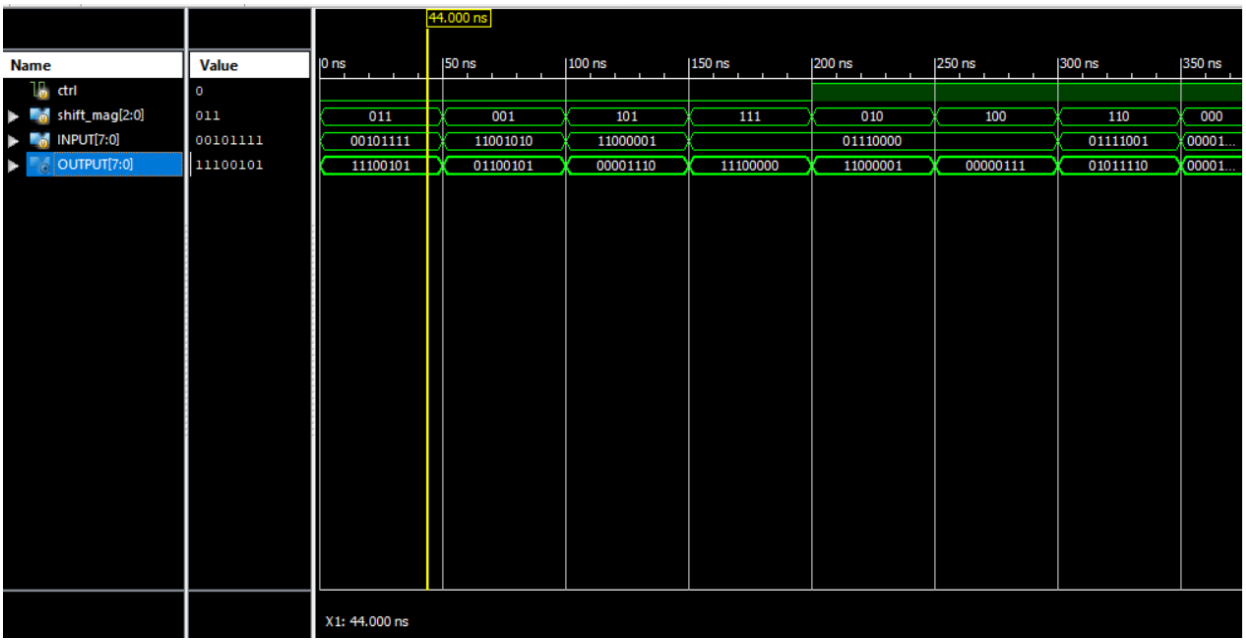
ما در اینجا ۳ بیت برای مشخص کردن میزان شیفت به چپ یا راست داریم و بنابراین در مجموع ۱۶ حالت شیفت وجود دارد که البته می توان آن را در ۸ حالت خلاصه کرد چون در یک عدد ۸ بیتی m بیت شیفت به راست با ۸-m بیت شیفت به سمت چپ برابر است و می توان شرط ها را در کد خلاصه کرد که من بدون خلاصه نویسی شرط ها را نوشتم اما در اینجا حالت خلاصه شده را می نویسم.

برای نوشتن شرط ها من از concatenate کردن یک بیت جهت شیفت و سه بیت اندازه شیفت کمک گرفتم و برای مجموع ۴ بیت آن شرط را نوشتم.

می توانیم در کد شرط های ۷ بیت به راست و ۱ بیت به چپ و همچنین ۶ بیت به راست و ۲ بیت به چپ و ... را با کمک or یکی کنیم و از تعداد شرط های خود کم کنیم.

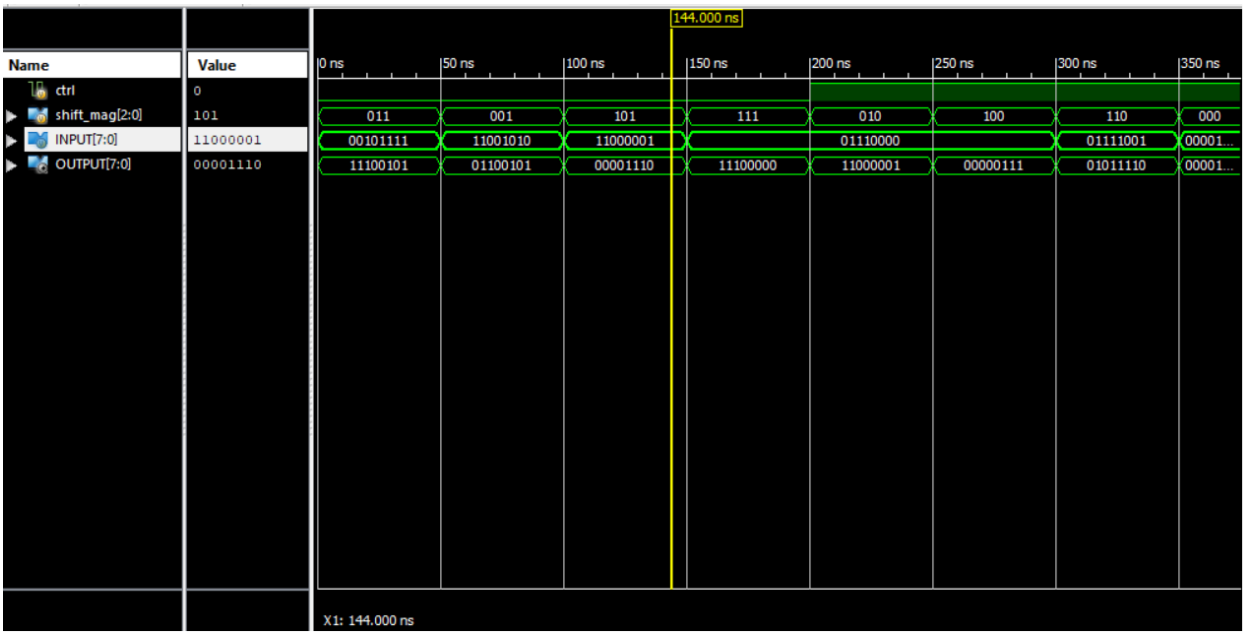
من در کد اگر کنترل ۰ بود شیفت به راست و اگر ۱ بود شیفت به چپ در نظر گرفتم.

عکس های سوال اول:

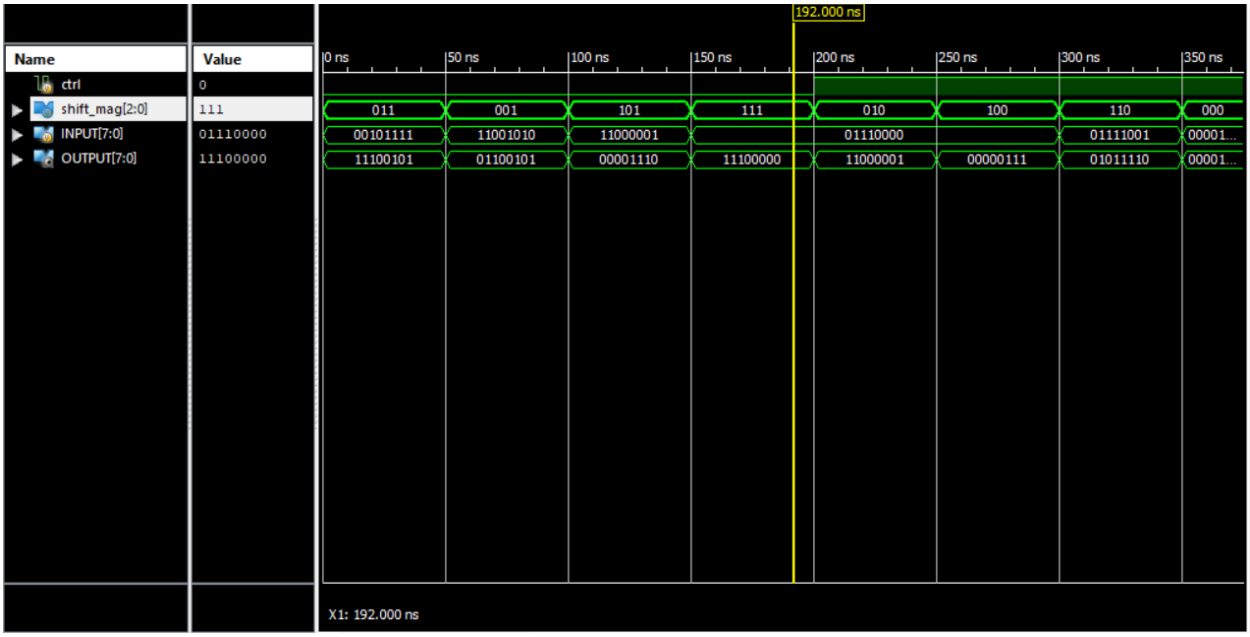


عدد ۰۰۱۰۱۱۱۱ با سه شیفت به راست به ۱۱۱۰۰۱۰۱ تبدیل شد.

عدد ۱۱۰۰۱۰۱۰ با یک شیفت به راست به ۰۱۱۰۰۱۰۱ تبدیل شد.

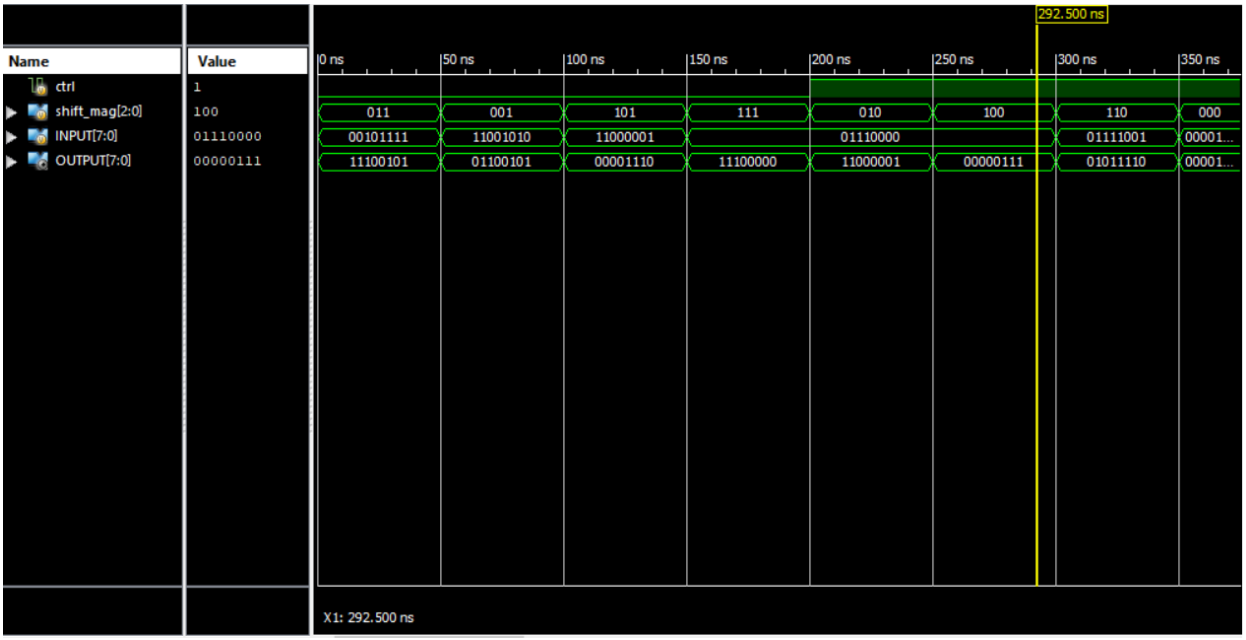


عدد ۱۱۰۰۰۰۰۱ با پنج شیفت به راست به ۰۰۰۰۱۱۱۰ تبدیل شد.



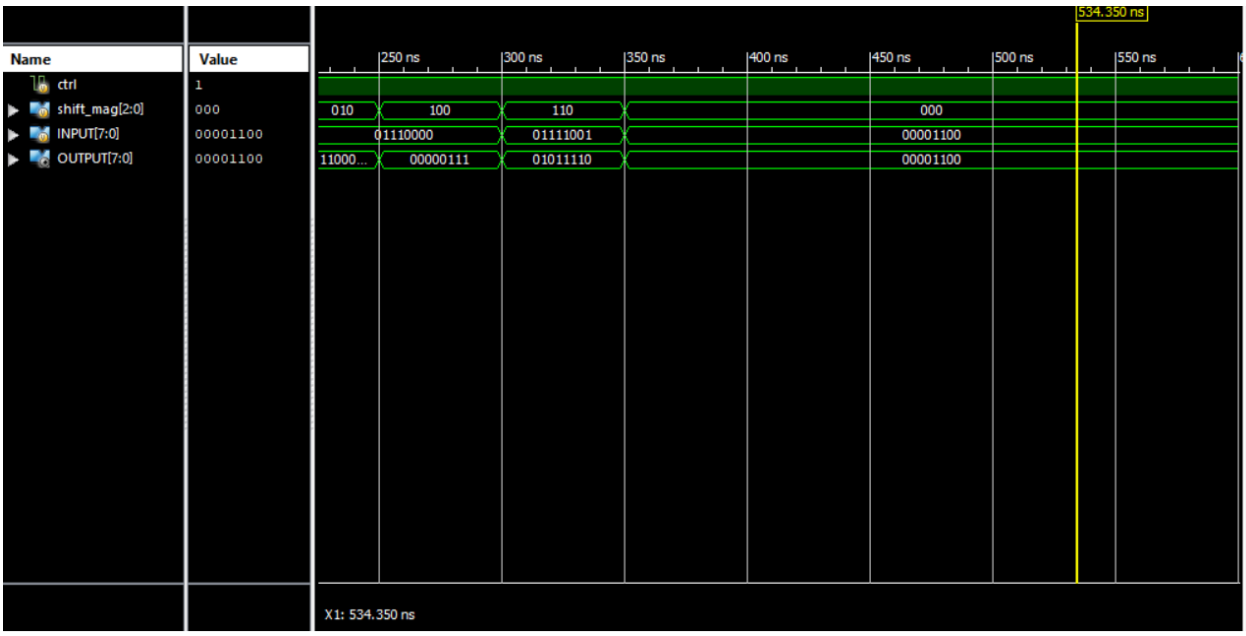
عدد ۰۱۱۱۰۰۰۰ به هفت شیفت به راست به ۱۱۱۰۰۰۰۰ تبدیل شد. در این عدد به خوبی برابر بودن هفت شیفت به راست با یک شیفت به چپ مشخص است.

عدد ۰۱۱۱۰۰۰۰ با دو شیفت به چپ به ۱۱۰۰۰۰۰۱ تبدیل شد.

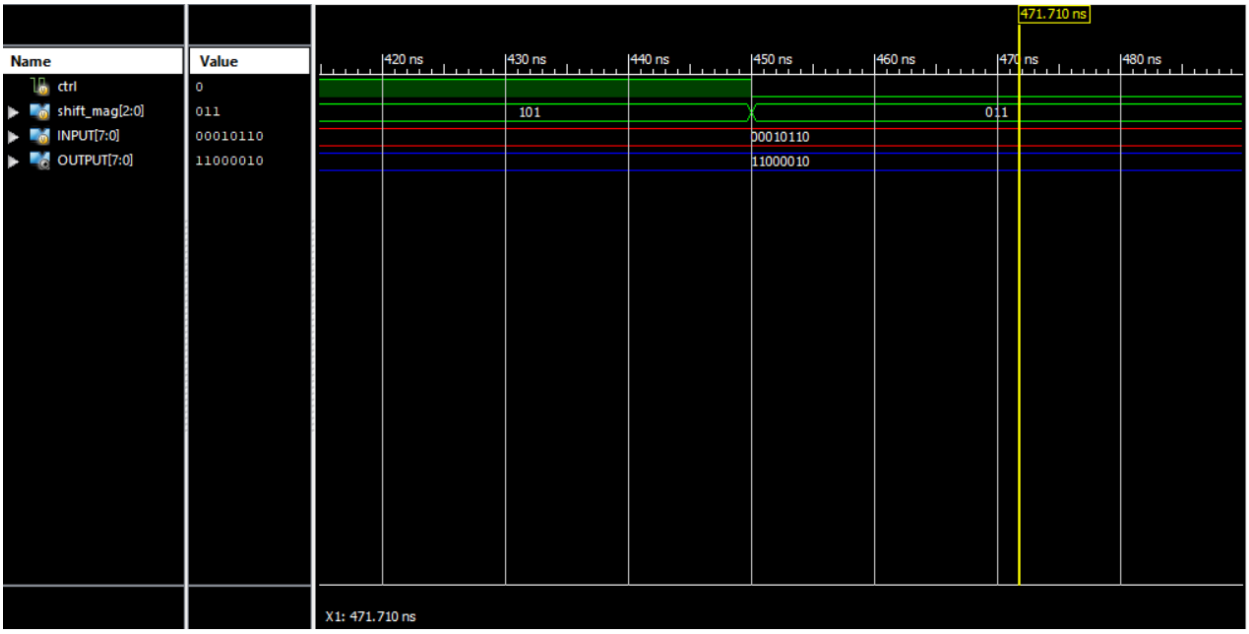


عدد ۰۱۱۱۰۰۰۰ با چهار شیفت به چپ به ۰۰۰۰۰۱۱۱ تبدیل شد.

عدد ۰۱۱۱۱۰۰۱ با شش شیفت به چپ به ۰۱۰۱۱۱۱۰ تبدیل شد.



و در نهایت عدد ۰۰۰۰۱۱۰۰ که صفر شیفت داده شده و طبیعتا تغییری نکرده است.



در اینجا میبینیم که عدد ۰۰۰۱۰۱۱۰ با سه شیفت به راست و پنج شیفت به چپ در هر دو حالت خروجی ۱۱۰۰۰۰۱۰ را داده است.

سوال دوم:

در ابتدا فاصله همینگ را تعریف میکنیم. برای عدد یا رشته فاصله همینگ برابر با تفاوت در میزان بیت های نظیر آن دو است. برای بهتر فهمیدن موضوع دو مثال میزنیم:

مثال اول:

First string=Damad

Second string=Madam

در اینجا فاصله همینگ برابر سه می باشد چون تنها کاراکتر های دوم و چهارم یکسان است و سایر کاراکتر ها با وجود شباهت باعث فاصله همینگ می شود.

مثال دوم:

First number=۱۰۱۰۱۰۱۰

Second number=۰۰۱۱۰۰۱۱

در اینجا فاصله همینگ برابر ۴ می باشد. کاراکتر ها و بیت های هرمز مشترک بوده و مشکی ها فاصله همینگ را ایجاد می کنند.

برای یافتن فاصله همینگ بین دو عبارت عددی دو راه زیر وجود دارد:

راه اول:

بیت به بیت دو عدد را باهم xor کنیم(bit wise xor) چون عملگر xor اگر دو ورودی یکسان داشته باشد خروجی آن صفر و اگر غیر یکسان داشته باشد خروجی آن یک است.پس با xor کردن بیت به بیت دو عدد می توانیم از روی تعداد یک های به دست آمده فاصله همینگ را محاسبه کنیم.

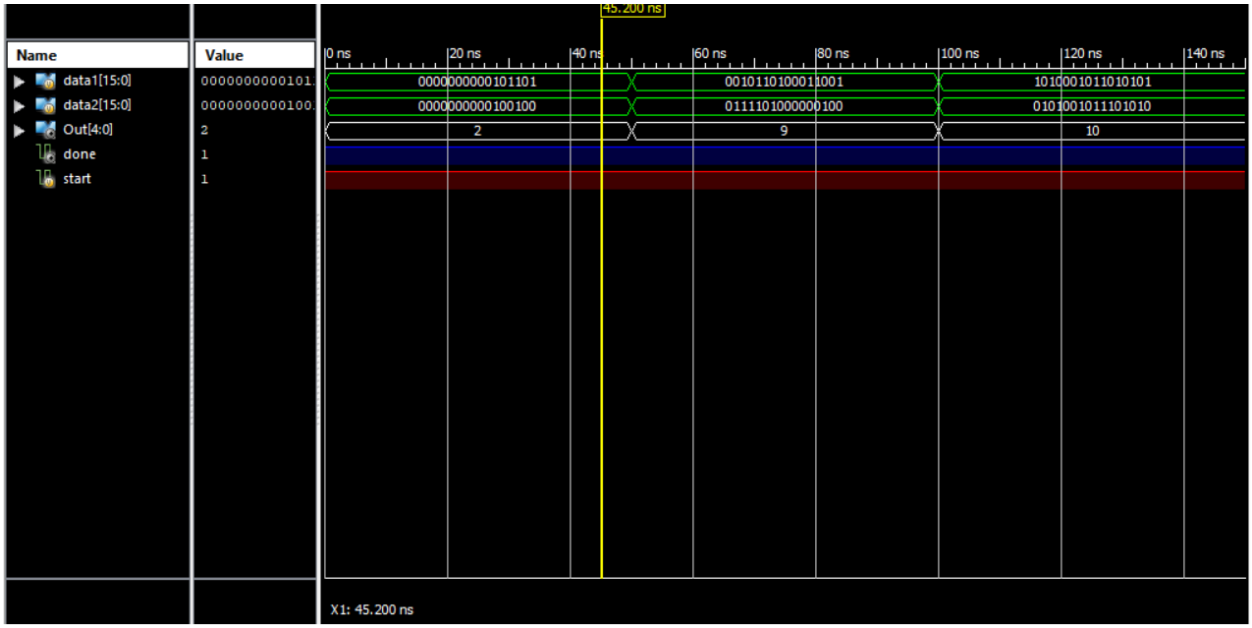
راه دوم:

بیت به بیت با هم مقایسه کنیم و در صورت تساوی به یک شمارنده یک واحد اضافه کنیم که البته سخت افزار بیشتری لازم دارد و در این سوال از روش اول استفاده شده است.

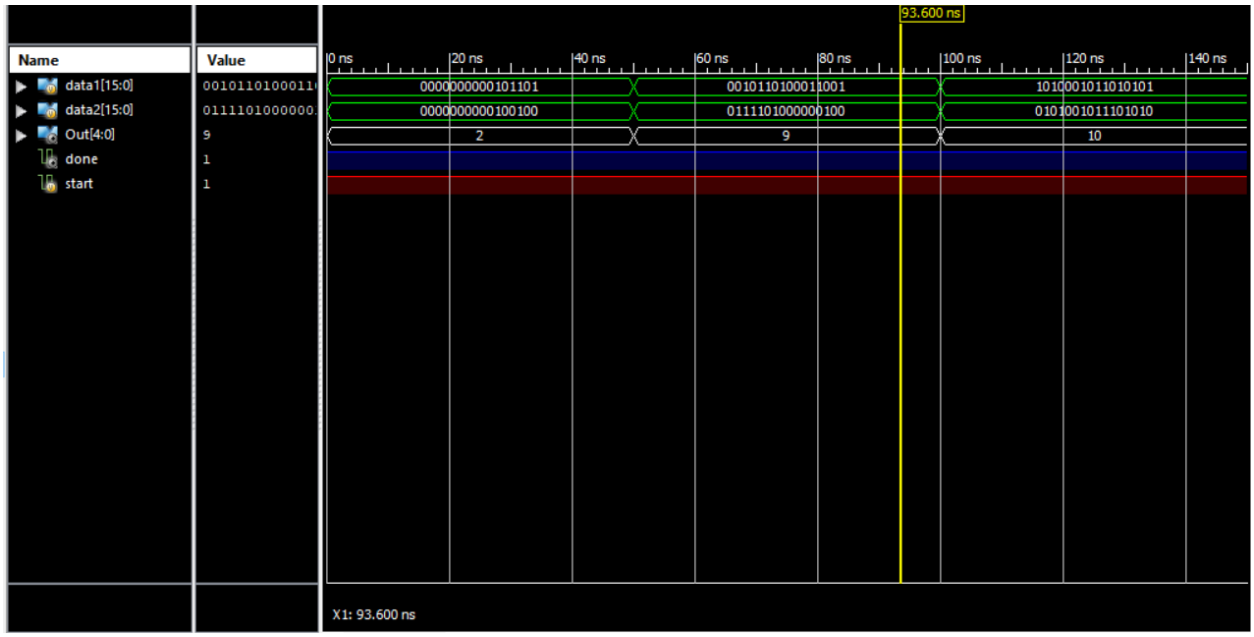
همچنین برای اینکه به شرط یک بودن پایه start عملیات انجام شود برای انجام هر xor یک شرط بررسی ۱ بودن پایه start قرار دادم و اگر پایه استارت برابر ۱ نباشد خروجی برابر z یا high impedance است.

برای پایه done هم شرط ۱ بودن پایه استارت را قرار دادم و مانند خروجی out اگر پایه استارت برابر صفر باشد در خروجی done هم high impedance خواهیم داشت.

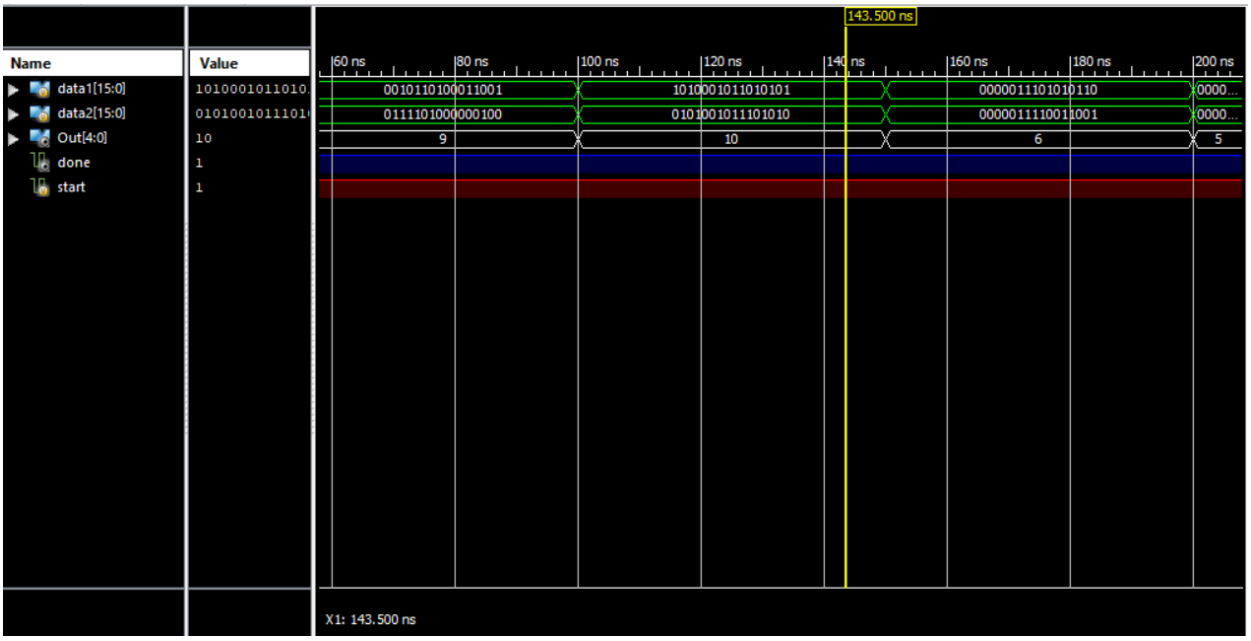
عکس های سوال دوم:



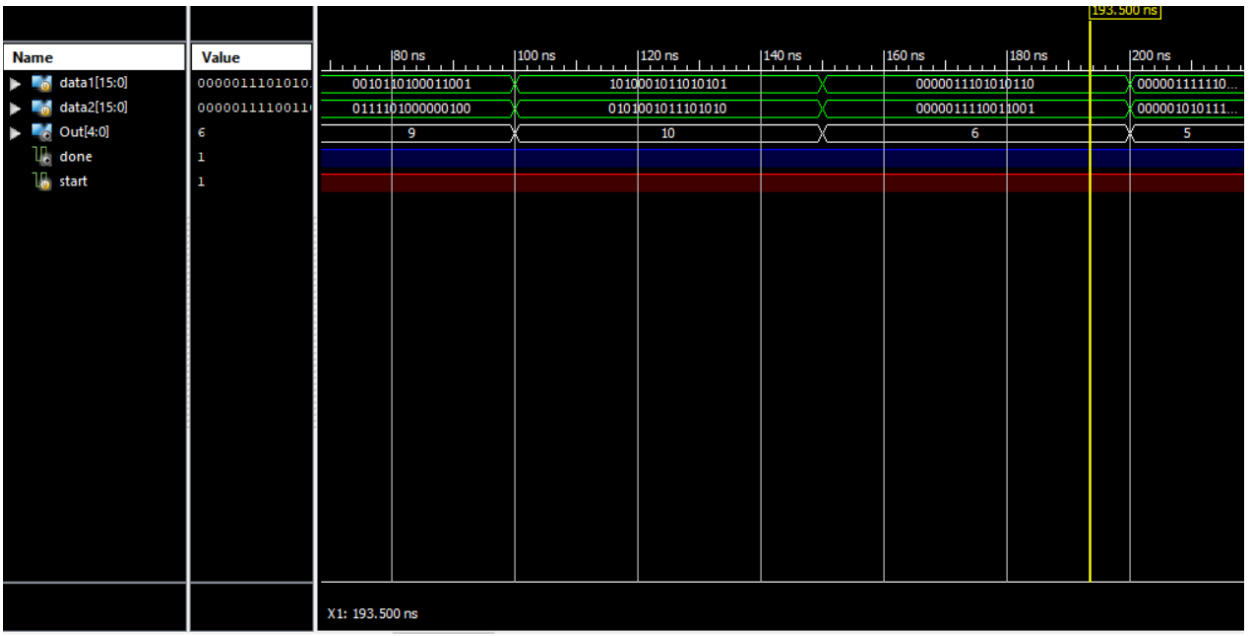
اعداد خوق معادل باینری ۱۶ بیتی اعداد ۴۵ و ۳۶ هستند.



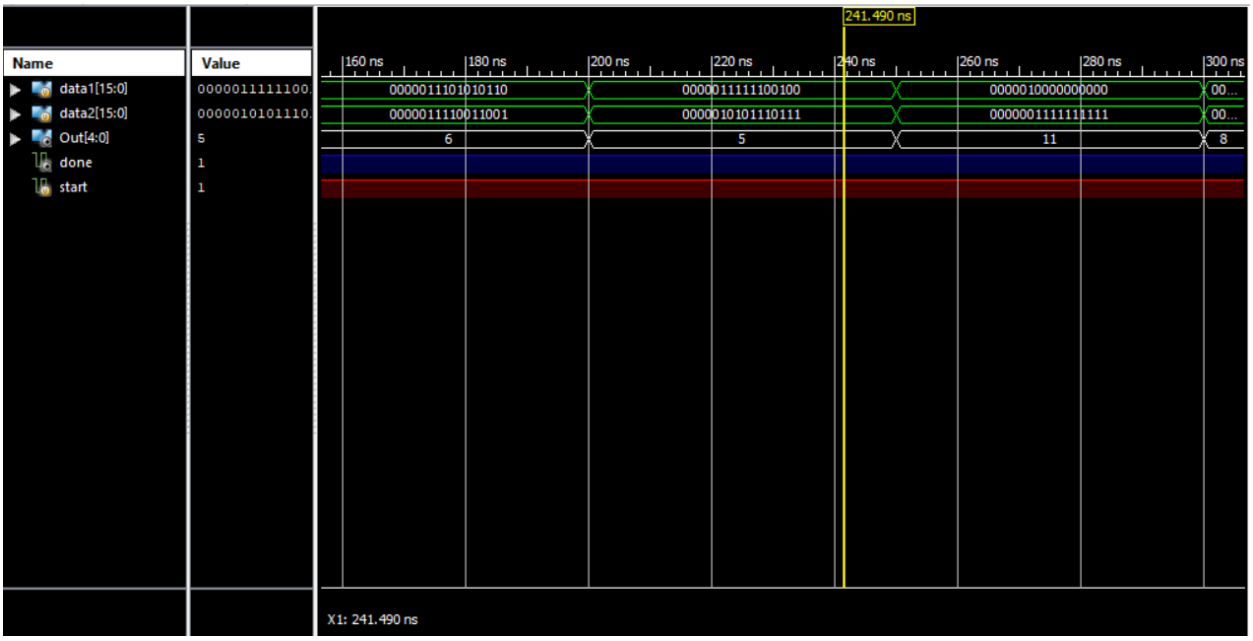
اعداد خوق معادل باینری ۱۶ بیتی اعداد ۱۱۵۴۵ و ۳۱۲۳۶ هستند.



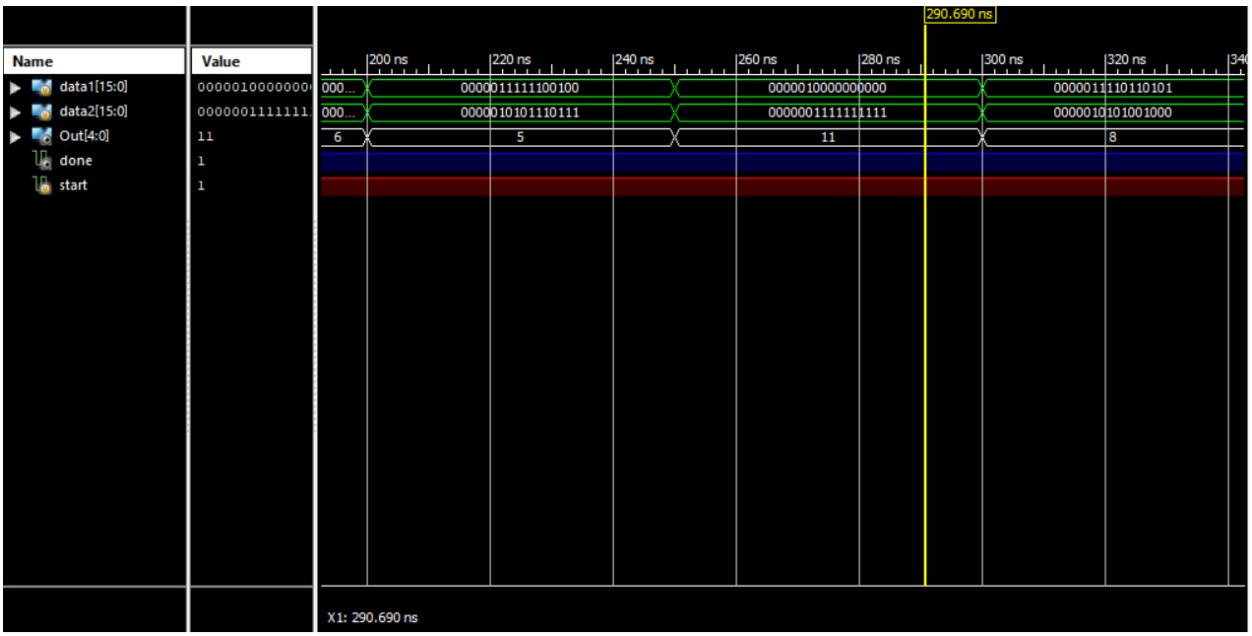
اعداد فوق معادل باینری ۱۶ بیتی اعداد ۴۱۶۸۵ و ۲۱۲۲۶ هستند.



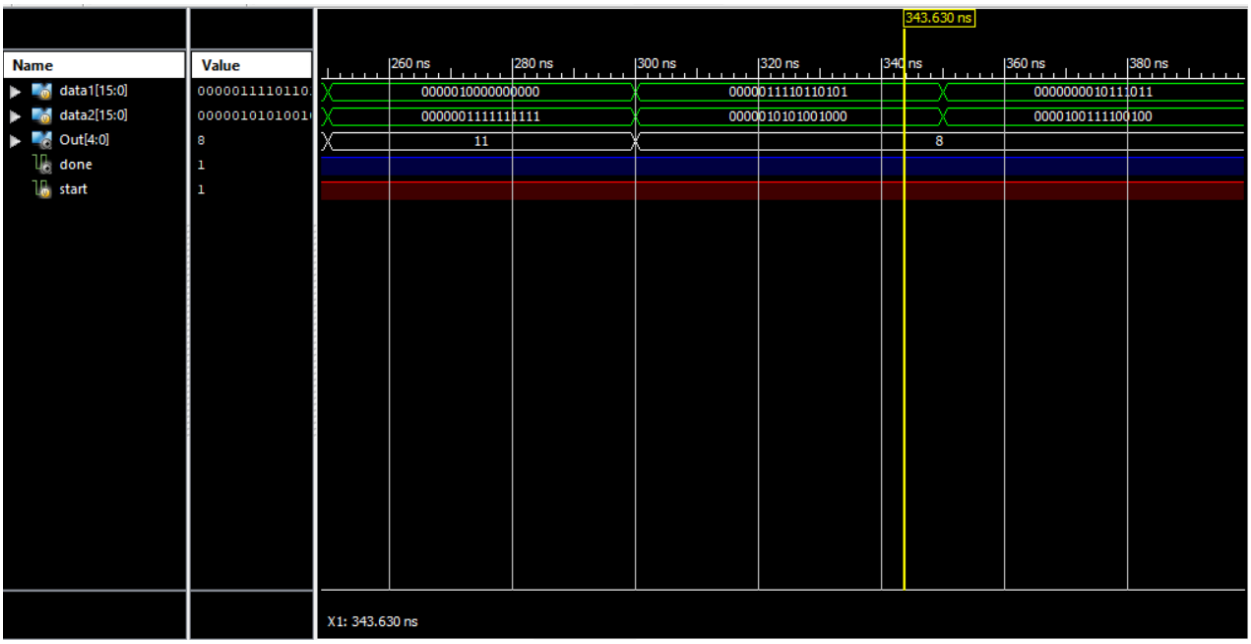
اعداد فوق معادل باینری ۱۶ بیتی اعداد ۱۹۴۵ و ۱۸۷۸ هستند.



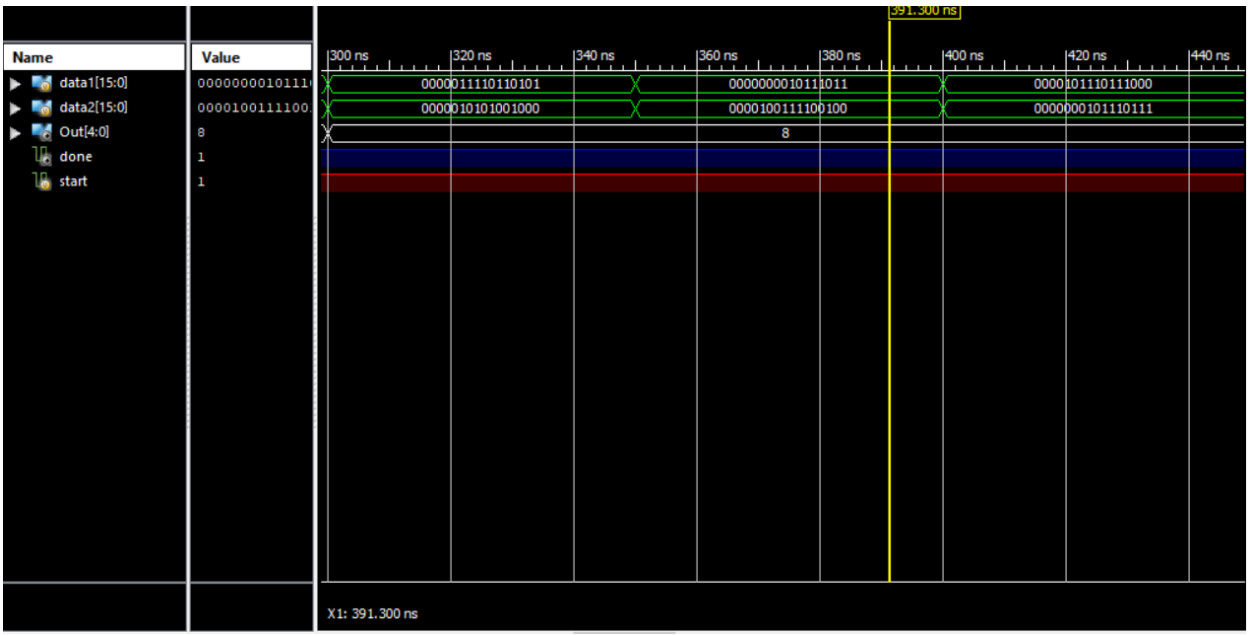
اعداد فوق معادل باینری ۱۶ بیتی اعداد ۲۰۲۰ و ۱۳۹۹ هستند.



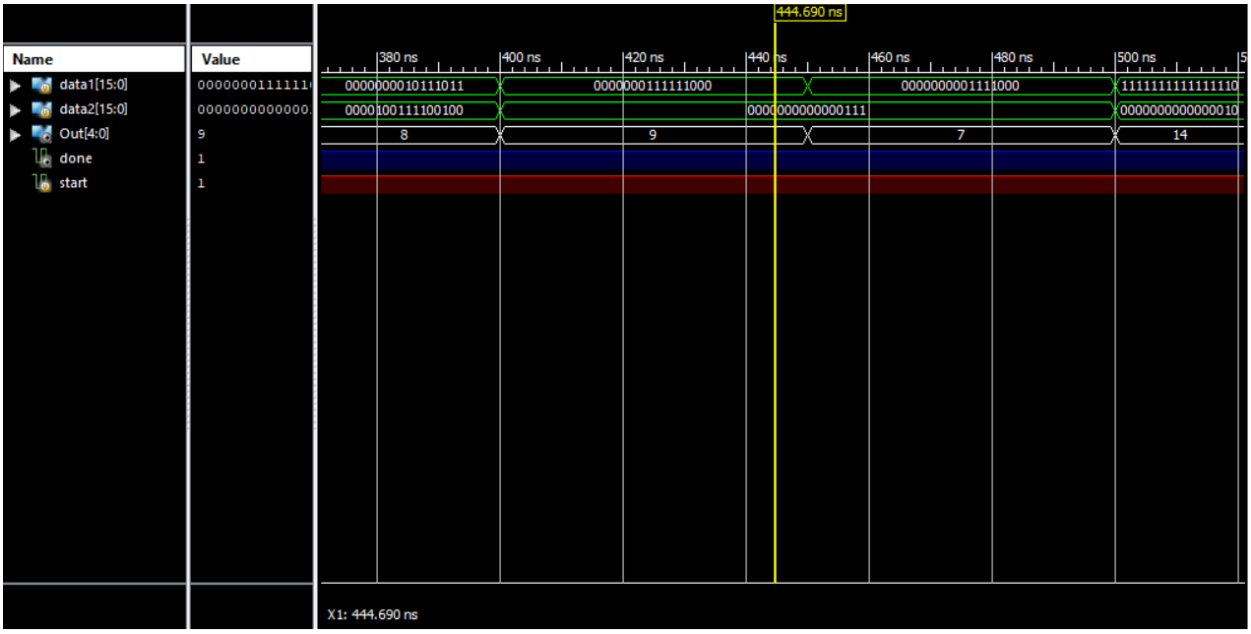
اعداد فوق معادل باینری ۱۶ بیتی اعداد ۱۰۲۴ و ۱۰۲۳ هستند.



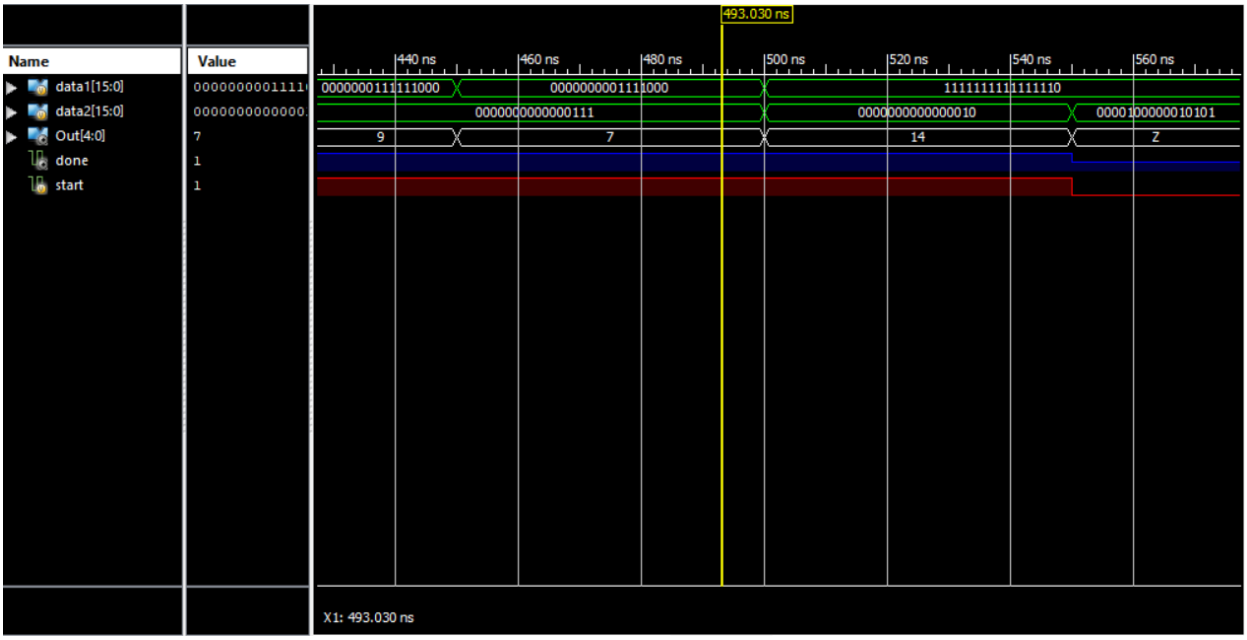
اعداد فوق معادل باینری ۱۶ بیتی اعداد ۱۹۷۳ و ۱۳۵۲ هستند.



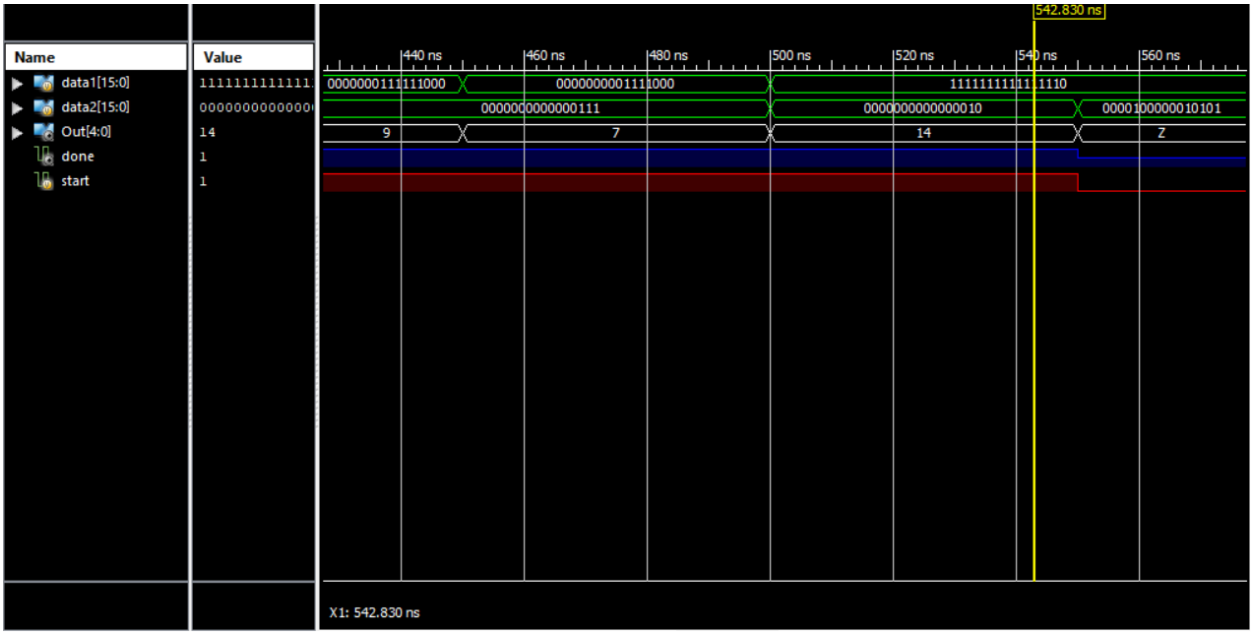
اعداد فوق معادل باینری ۱۶ بیتی اعداد ۱۸۷ و ۲۵۳۲ هستند.

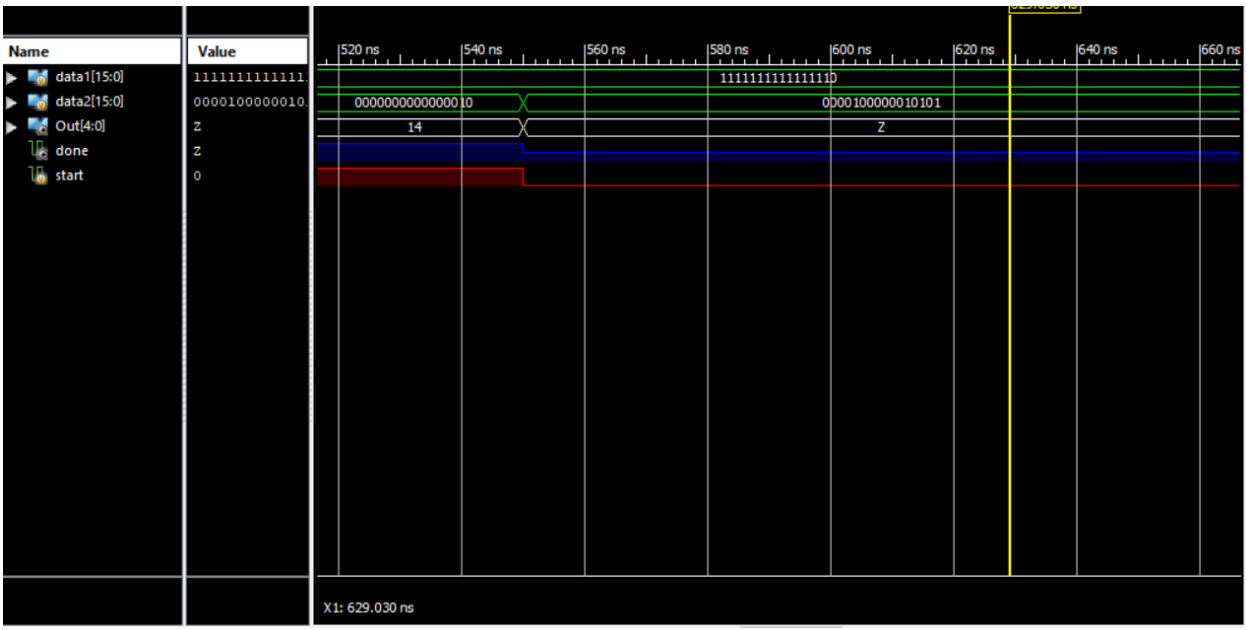


اعداد خوق معادل باینری ۱۶ بیتی اعداد ۵۰۴ و ۷ هستند.



اعداد خوق معادل باینری ۱۶ بیتی اعداد ۱۲۰ و ۷ هستند.





اعداد فوق معادل باینری ۱۶ بیتی اعداد ۶۵۵۳۴ و ۲۰۶۹ هستند. در عکس اول پایه start مانند عکس های قبل برابر ۱ است اما در عکس دوم پایه start صفر شده و بنابراین out و done هر دو z می شوند.

نکته مهم:کد سوال آخر بعد از ران کردن در ابتدا با خطای fatal error:not able to add errors مواجه بود و برای رفع مشکل باید گزینه run as administrator زده شود.

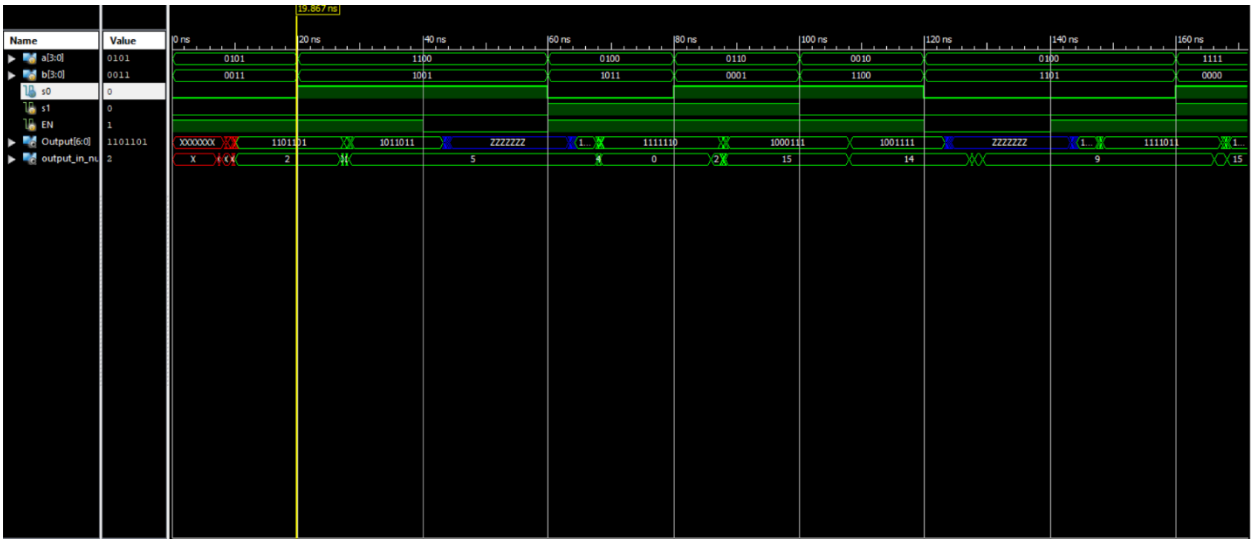
سوال سوم:

در ابتدا با کمک دستور assign و کمک گرفتن از عملگر های منطقی خروجی های مدنظر را میسازیم که البته من خروجی nand را با کمک not کردن and نوشتم و مستقیما دستوری برای آن نبود. سپس خط انتخاب را با کمک ترکیب s۰ و s۱ و not های آن ها می سازیم. من برای ساختن خروجی مورد نظر چهار خروجی خط انتخاب را که یک بیت بودند(حاصل and کردن s۰ و s۱ و not های آن ها)را concatenate کردم تا به ۴ بیت تبدیل شود و با ۴ بیت خروجی های محاسبه کننده بتوان آن را بیت به بیت and کرد.

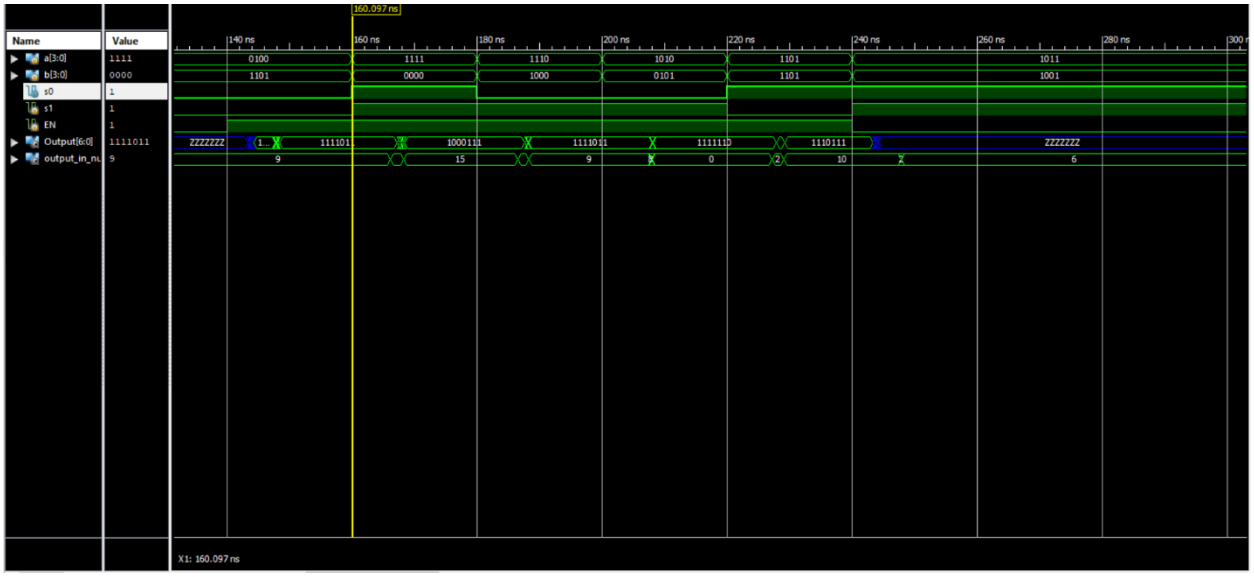
در ماژول hexto۷seg هم ۱۶ شرط گذاشتم که هر شرط خود از دو شرط تشکیل شده یکی مقدار عدد ۴ بیتی و یکی هم ۱ بودن پایه EN. این ماژول به این صورت کار میکند که با کمک این ۱۶ شرط عدد ۴ بیتی ورودی را پیدا می کند و به عنوان خروجی یک عدد ۷ بیتی به سون سگمنت می دهد. اگر هم معادل هیچکدام نبود خروجی را Z در نظر می گیرد.

البته من ۷ بیت خروجی را به ترتیب abcdefg در نظر گرفتم و پر ارزش ترین بیت برای روشن یا خاموش کردن a است و کم ارزش ترین برای g. همچنین سون سگمنت را کاتد مشترک در نظر گرفتم.

عکس های سوال سوم:



می بینیم که حین صفر شدن EN خروجی هم z یا High Impedance شده است.



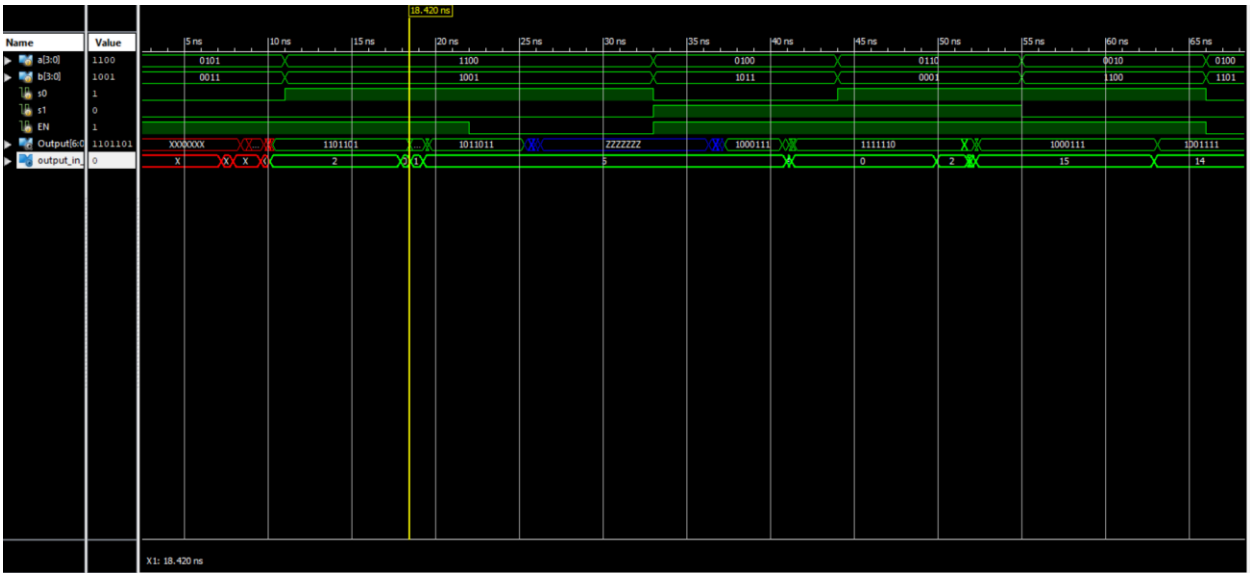
حدودا جمع ۸/۸ نانو ثانیه، تفریق ۱۰/۳ نانوثانیه، xnor ۸/۶ ثانیه و nand ۸/۴ ثانیه طول کشید

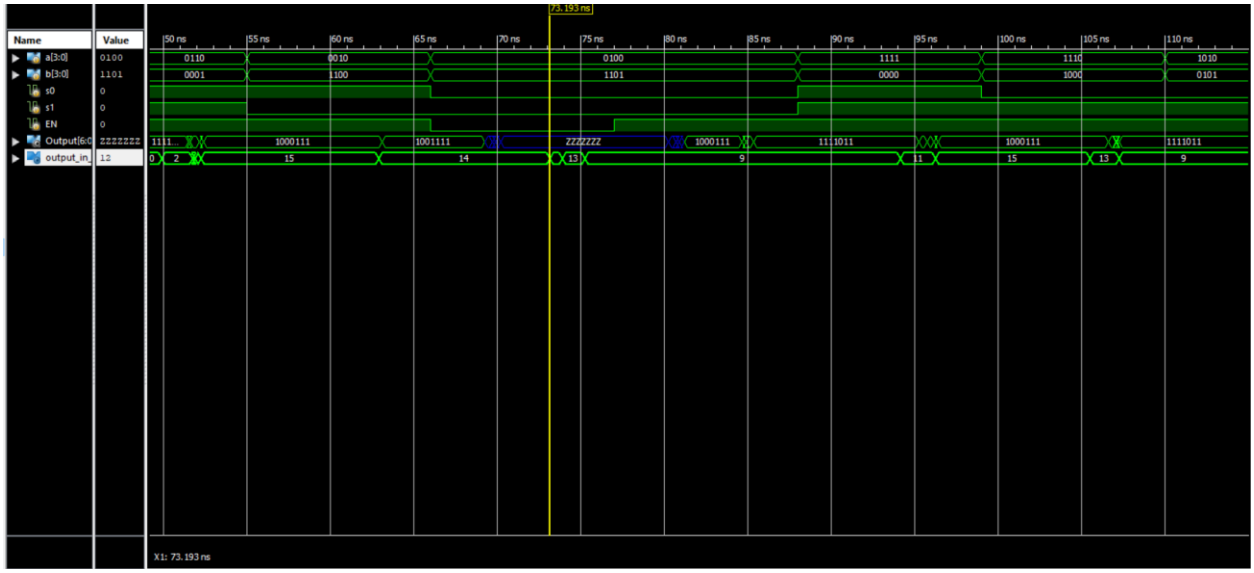
نکته قابل توجه این است که در بعضی لحظات مانند لحظه ۶۴ یا ۸۵ نانوثانیه به مدت حدودا دو نانوثانیه خروجی مدار با خروجی تئوری تفاوت دارد و این بخاطر اختلاف زمانی بین بیت های مختلف برای انجام عملیات است.

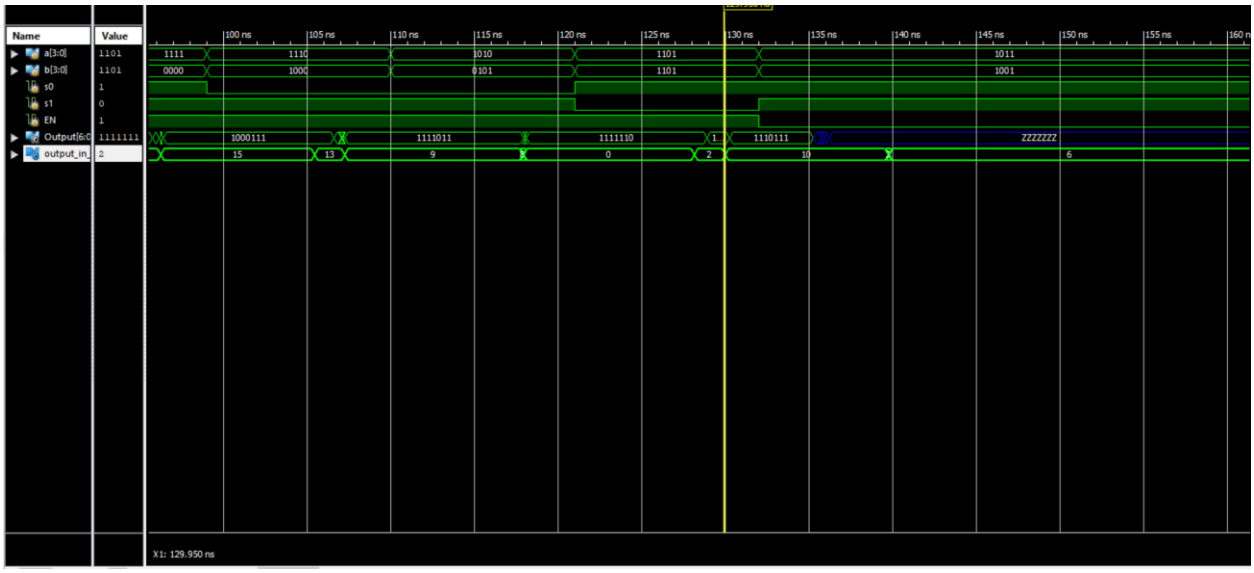
عکس های فوق با تاخیر ۲۰ نانوثانیه ای می باشد.

بعد از تغییر تاخیرها به ۱۷ و ۱۴ نانوثانیه هم اتفاق خاصی نیفتاد.

اما بعد از کم کردن تاخیر به ۱۱ نانوثانیه خروجی رفته رفته دچار مشکل شد.





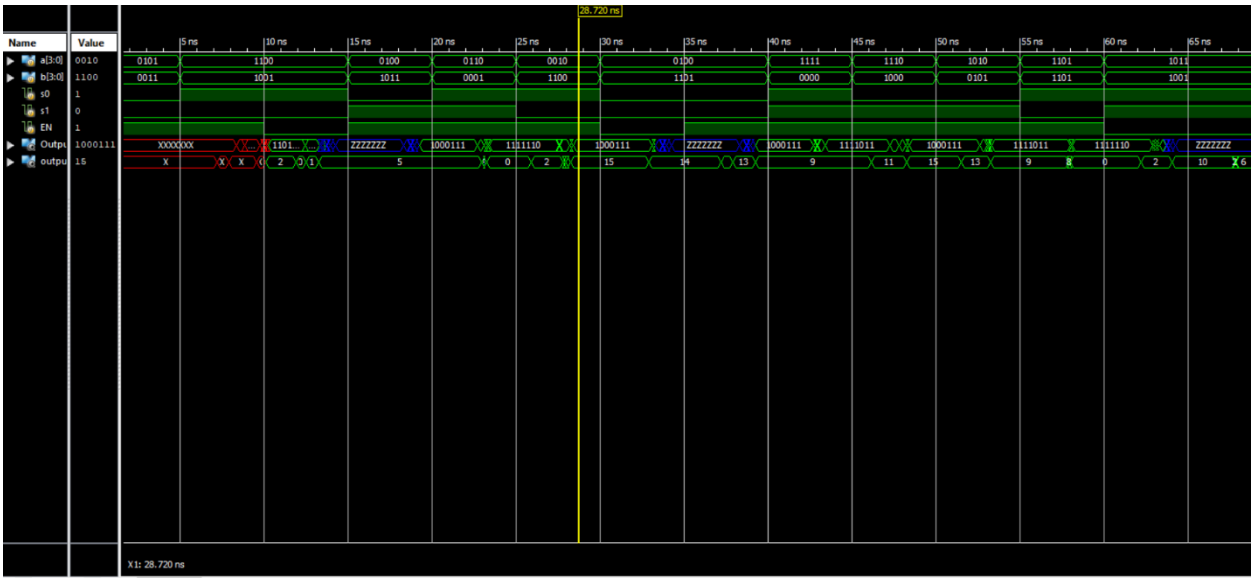


در جایی که خط زرد قرار داده شده است خروجی دچار مشکل های دو نانو ثانیه ای شده و به دلیل تفاوت در سرعت انجام عملیات روی بیت های مختلف و البته تغییرات سریع می باشد.

مثلا در همین شکل آخر ALU روی مد جمع قرار دارد و هر دو ورودی برابر ۵ می باشند اما می بینیم به خاطر دلایل ذکر شده به مدت حدود دو ثانیه خروجی به اشتباه ۲ ذکر شده و نتیجتا کد نظیر ۲ به سون سگمنت ارسال گردیده است.

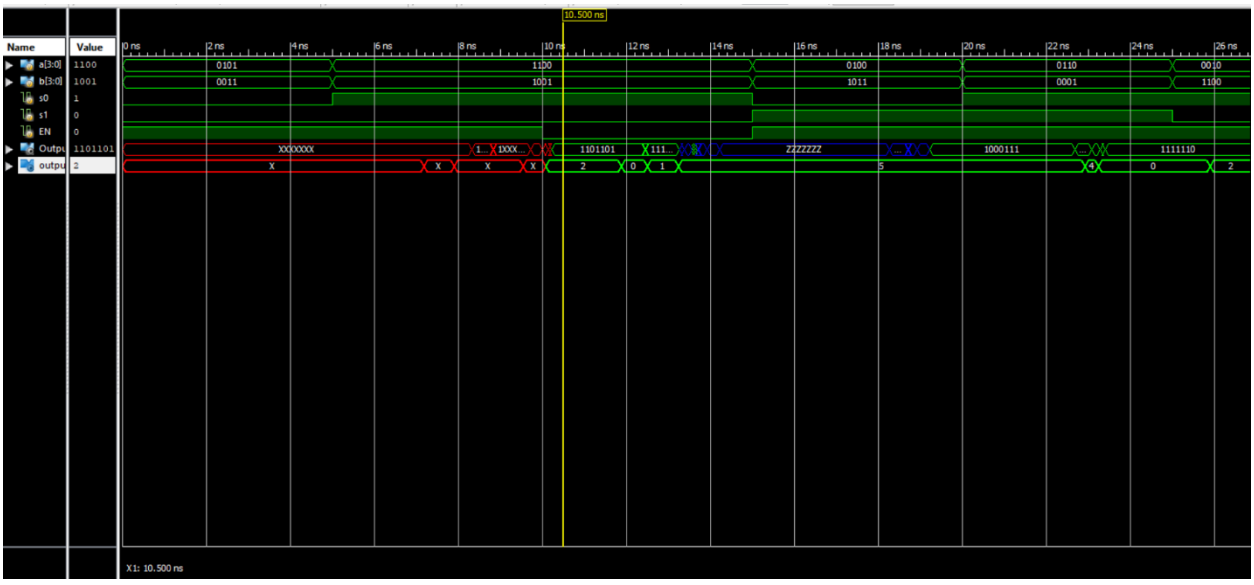
سپس میزان تاخیر را به ۷ نانوثانیه کاهش دادم و دیدم دنبال کردن خروجی به شدت سخت شده است و تداخل ها بیشتر شده است.

سپس میزان تاخیر را به ۵ نانوثانیه کاهش دادم:

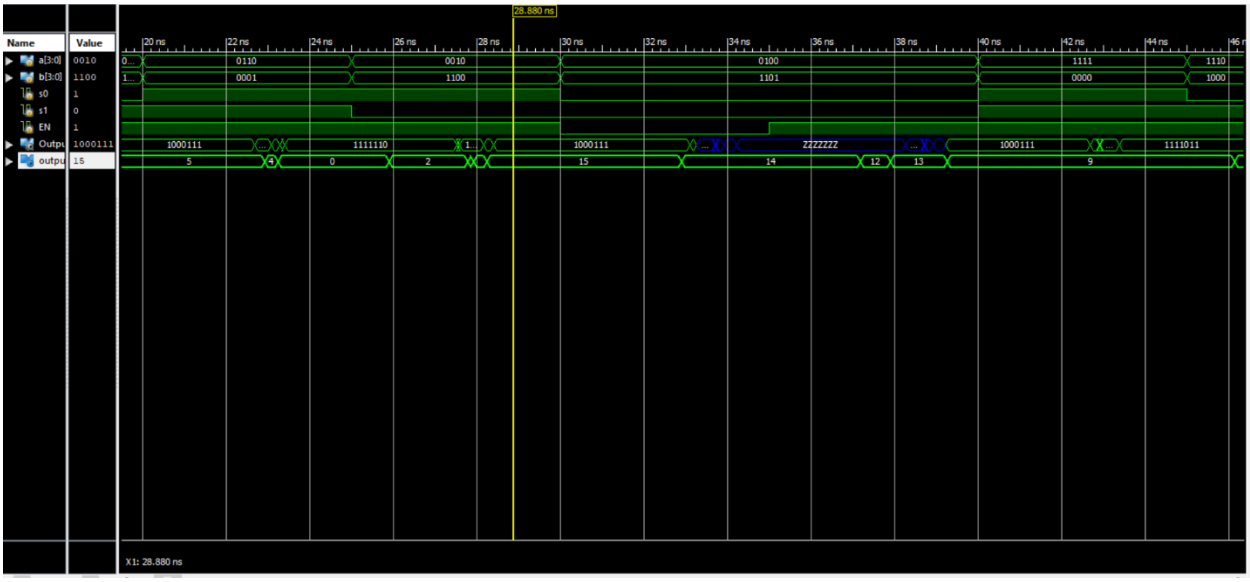


می بینیم که دنبال کردن خروجی از روی ورودی غیر ممکن است و سرعت تغییرات ورودی از سرعت انجام عملیات و محاسبات بسیار بیشتر است و بسیاری از خروجی ها بی معنی هستند.

برای مثال دو عکس زیر را قرار دادم:



در عکس بالا میبینیم که خروجی ۲ که خروجی صحیح است مدتها بعد از اعمال ورودی نظیر آن ساخته شده و نمیتوانیم به درستی خروجی را دنبال کنیم.



در اینجا هم حدود لحظه ۲۸ نانو ثانیه شاهد تغییرات بسیار سریع و پی در پی خروجی هستیم که به دلیل سرعت بسیار زیاد تغییرات ورودی نسبت به سرعت انجام عملیات و تفاوت سرعت اجرای عملیات روی بیت های مختلف است.