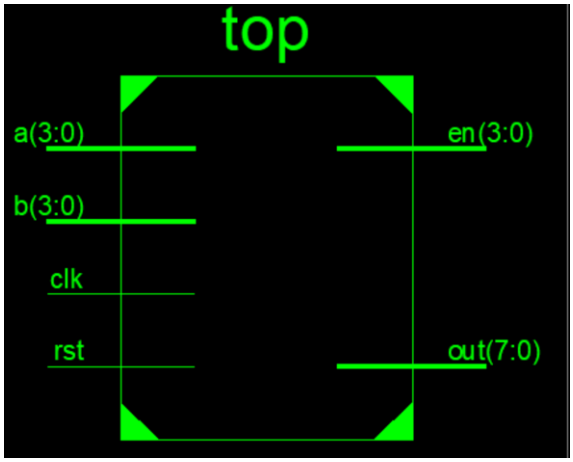
به نام خدا

گزارش آزمایش شماره 2

محدثه غفوری(9632133)

گروه چهارشنبه عصر



مطابق تصویر بالا ماژول اصلی بنام top ورودی های 4 بیتی a,b را از کاربر میگیرد و خروجی های out_sub , out_sum که معادل کد سون سگمنت برای خروجی های تفریق و جمع دو عدد است ، را میدهد

برای انجام عمل جمع و تفریق توسط پیکوبلیز باید کد انجام این کار را به زبان اسمبلی در یک فایل تکست نوشته و انرا به فایل psm. تبدیل کنیم که کد ان به صورت زیر است (دستور add جمع و دستور sub تفریق دو متغیر را انجام داده و حاصل را در متغیر اول میریزد) همچنین اعداد 81تا84 port_id هستند که در مبنای 16 نوشته شده اند

```
Loop:
INPUT S0,81
INPUT S1,82
LOAD S2,S0
LOAD S3,S0
ADD S2,S1
SUB S3,S1
OUTPUT S2,83
OUTPUT S3,84
JUMP Loop
```

پس از تبدیل به فایل psm. باید بوسیله اسمبلر داده شده این فایل را به یک ماژولv. تبدیل کنیم

پس از تبدیل به فایل v. پروژه دو ماژول دیگر دارد یکی ADDORSUB و یکی kcpsm6

ماژول kcpsm6 هسته ی پیکوبلیز است که به زبان وریلاگ در امده است و ماژول ADDORSUB همان عملیات جمع و تفریق را انجام میدهد که توسط اسمبلر تولید شده است

در این دو ماژول باید پایه های address و enable و clk هر دو بهم متصل باشد . همچنین باید پایه ی sleep ماژول kcpsm6 صفر باشد

درصورتی که ریست خارجی نداشته باشیم پایه rdl ماژول ADDORSUB به پایه ریست ماژول kcpsm6 متصل میشود اما اینجا بعلت وجود یک ورودی بنام rst باید این ریست خارجی هم اعمال شود به این منظور rdl, rst با هم or میشود و خروجی حاصل را به ریست ماژول kcpsm6 میدهیم تا اگر هر کدام از این پایه ها فعال شد ماژول ریست شود و تحت اثر هر دو باشد

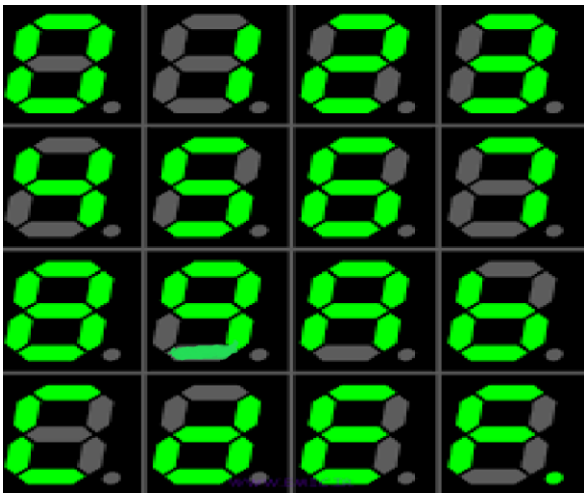
از انجایی که سوال خواسته است فرکانس ورودی (MHz24) به یک هرتز کاهش یابد متغیری از جنس حافظه بنام cnt را تعریف میکنیم که در هر لبه بالارونده کلاک ورودی یک واحد به ان اضافه شود و تا 12,000,000را شمارش کند و هر دوازده میلیون یک بار متغیر clk1 که نشان دهنده کلاک یک هرتزی است را تاگل کند ، تا این کلاک جدید ساخته شود (توجه شود که این عملیات و ساخت کلاک یک هرتزی زمانی انجام میشود که ریست ورودی غیر فعال باشد در غیر این صورت کلاک یک هرتز که به ماژول های ADDORSUB و kcpsm6 اعمال شده است همواره صفر خواهد بود و خروجی تولید نخواهد شد)

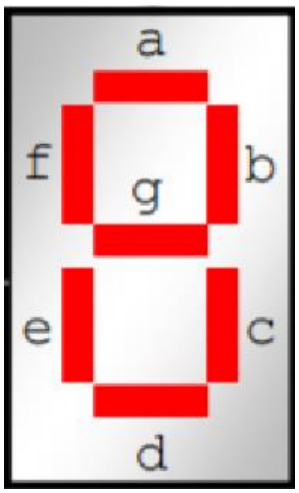
در این مرحله که کلاک یک هرتزی ساخته شد باید بر مبنای port_id ها ورودی ها را به پیکوبلیز بدهیم سپس در طی مراحلی که انجام میشود هنگامی که خروجی جمع یا تفریق اماده شد و port_id برابر 83 شد خروجی جمع دو عدد را در یک حافظه بنام add بریزیم و وقتی port_id برابر 84 شد خروجی تفریق را در حافظه ای دیگر بنام sub بریزیم

حال میدانیم که حاصل جمع و تفریق قرار است بر روی دو سون سگمنت نمایش داده شود پس باید این حاصل را به کد سون سگمنت تبدیل کنیم

با توجه به جدول زیر ماژول sevenseg_decoder را چنان طراحی میکنیم که به ازای هر ورودی از جدول بالا بیت های متناظر خروجی را یک کند تا عدد مبنا شانزده آن روی سون سگمنت نمایش داده شود اما باید دقت داشت تنها زمانی که پایه فعال ساز یعنی ENفعال و صفر باشد خروجی خواهیم داشت در غیر این صورت خروجی ماژول دیکودر صفر و هیچ LED سون سگمنت روشن نخواهد بود که به وسیله ی یک conditional operator در صورت برقرار بودن شرط یک بودن پایه EN خروجی را غیر صفر نمایش خواهد داد

در شکل پایین led مربوط به دات نمایش داده نشده است که در این ازمایش بعلت نیاز نداشتن به ان بعنوان msb در تمامی خروجی ها صفر قرار داده شد





	g	f	e	d	c	b	a
0	0	1	1	1	1	1	1
1	0	1	1	0	0	0	0
2	1	0	1	1	0	1	1
3	1	0	0	1	1	1	1
4	1	1	0	0	1	1	0
5	1	1	0	1	1	0	1
6	1	1	1	1	1	0	1
7	0	0	0	0	1	1	1
8	1	1	1	1	1	1	1
9	1	1	0	1	1	1	1
10=A	1	1	1	0	1	1	1
11=b	1	1	1	1	1	0	0
12=c	0	1	1	1	0	0	1
13=d	1	0	1	1	1	1	0
14=E	1	1	1	1	0	0	1
15=F	1	1	1	0	0	0	1

از انجایی که سون سگمنت ها بصورت مالتی پلکس هستند باید در هر کلاک یکی از 4 بیت پایه ی enable فعال باشد بنابراین متغیر counter را تعریف میکنیم و در هر کلاک 24 مگاهرتزی برای کاهش فرکانس 24 مگاهرتز به چند کلیو هرتز بوسیله ی متغیر counter خروجی را مالتی پلکس میکنیم بدین صورت که بعد از گذشت هر 1024 کلاک 24 مگاهرتز یکی از دو مقدار تفریق یا جمع را روی سون سگمنت ها نمایش میدهیم بنابراین سرعت مالتی پلکس شدن سون سگمنت ها کمی کاهش میابد تا با چشم دیده شود و برای فعال کردن سون سگمنت ها در متغیر en یکی از دو بیت کم ارزش را صفر و یک میکنیم تا بوسیله ان یکی از sum یا sub که مربوط به نمایش جمع یا تفریق روی سون سگمنت است ، فعال شود ودیگری خاموش میماند و چون تنها یک پورت خروجی داریم با توجه به اینکه در هر کلاک enable هرسون سگمنت که فعال شده است (سون سگمنت حاصل جمع یا حاصل تفریق) مقدار حاصل جمع یا حاصل تفریق را روی پایه خروجی ماژول قرار میدهیم

در اینجا چون دو سون سگمنت دیگر بورد بی استفاده اند دو بیت پر ارزش enable همواره یک و غیرفعال میماند

خروجی جمع بعد از 18 کلاک یک هرتزی و خروجی تفریق هم بعد از 20 کلاک یک هرتزی یعنی پس از 18 و 20 ثانیه تاخیر اماده شد

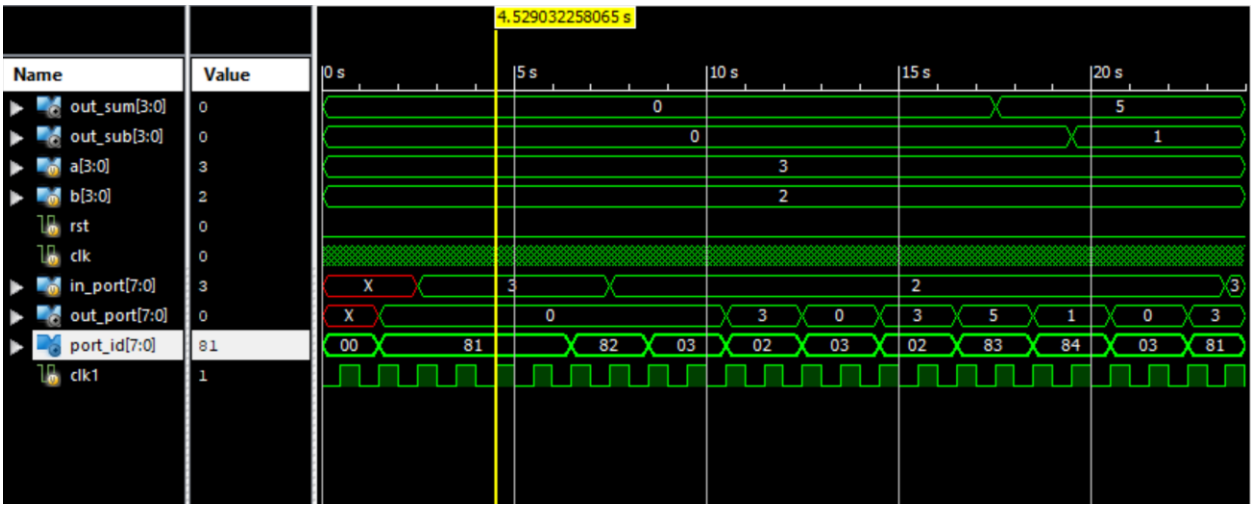
در کد تست بنچ از انجایی که میخواهیم دو سری ورودی a=3,b=2 و a=4,b=2 بدهیم و نمیدانیم خروجی جمع و تفریق در سری اول ورودی ها چه زمان اماده میشود تا بتوانیم ورودی سری دوم را اعمال کنیم از یک شرط استفاده میکنیم ، چون میدانیم خروجی تفریق برای سری داده اول برابر یک است و خروجی ماژول که کد سون سگمنت است برای عدد یک کد 8’b00110000 را میدهد از یک شرط استفاده میکنیم بدین صورت که اگر خروجی ماژول برابر کد مربوط به عدد یک شد سری داده جدید به صورت زیر اعمال شوند . با این روش ورودی ها را زودتر اعمال نمیکنیم و باعث اشتباه در محاسبات نمیشویم

```
always @(posedge clk) begin
if (  out==8'b00110000 ) begin
a=4;
b=2;
end
end
```

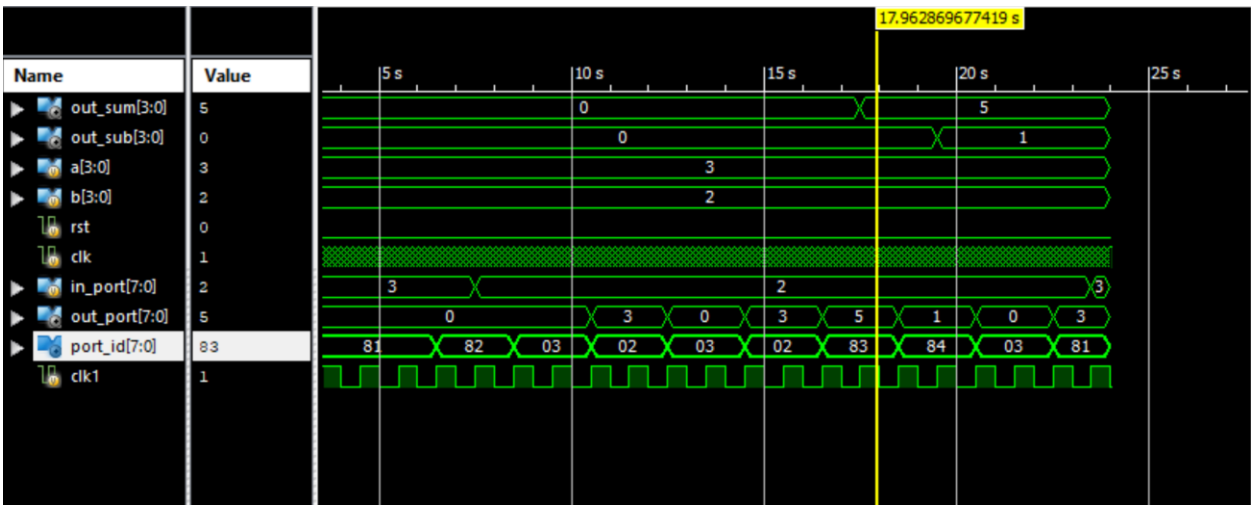
نتایج شبیه سازی و waveform ها

در این قسمت نتایج شبیه سازی قسمت 1 یعنی فقط خروجی پیکوبلیز را برای چند ورودی نشان میدهیم

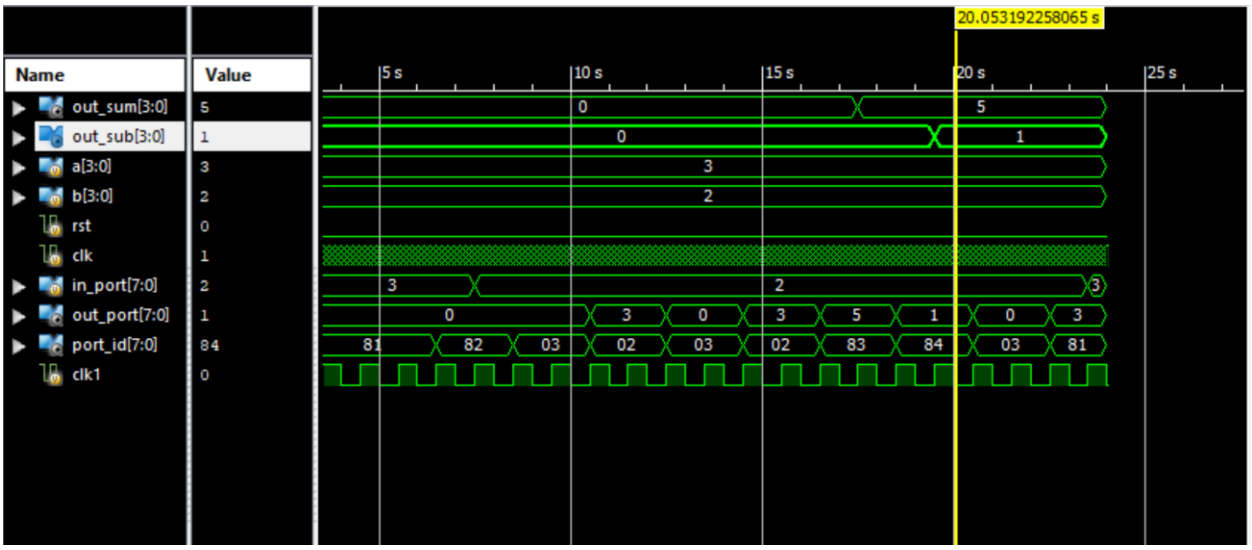
در port_id برابر 81 اولین ورودی یعنی a و در 82 دومین ورودی یعنی b بدرستی دریافت شده اند



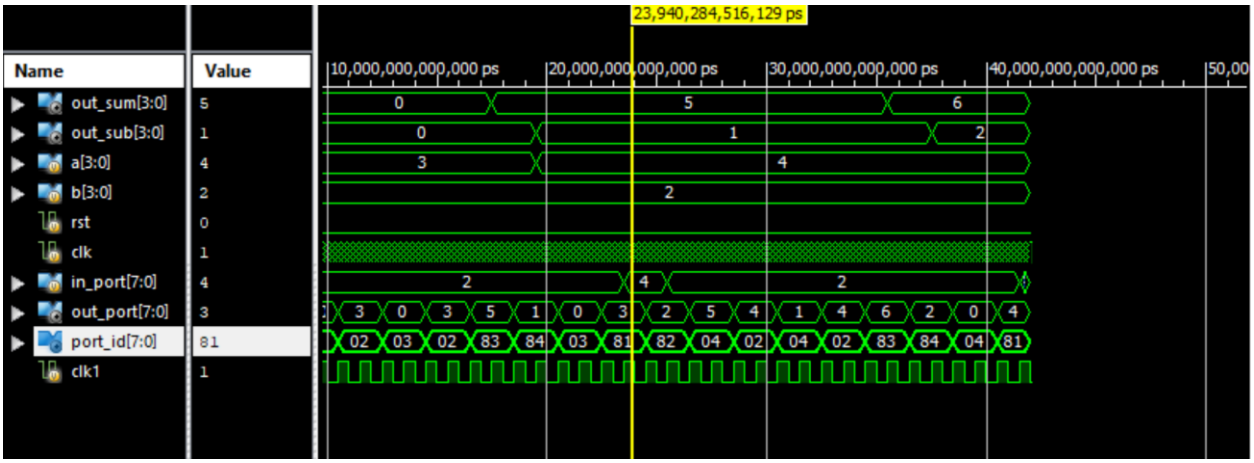
در port_id برابر 83 خروجی جمع بدرستی در متغیر out_sum قرار داده شده است



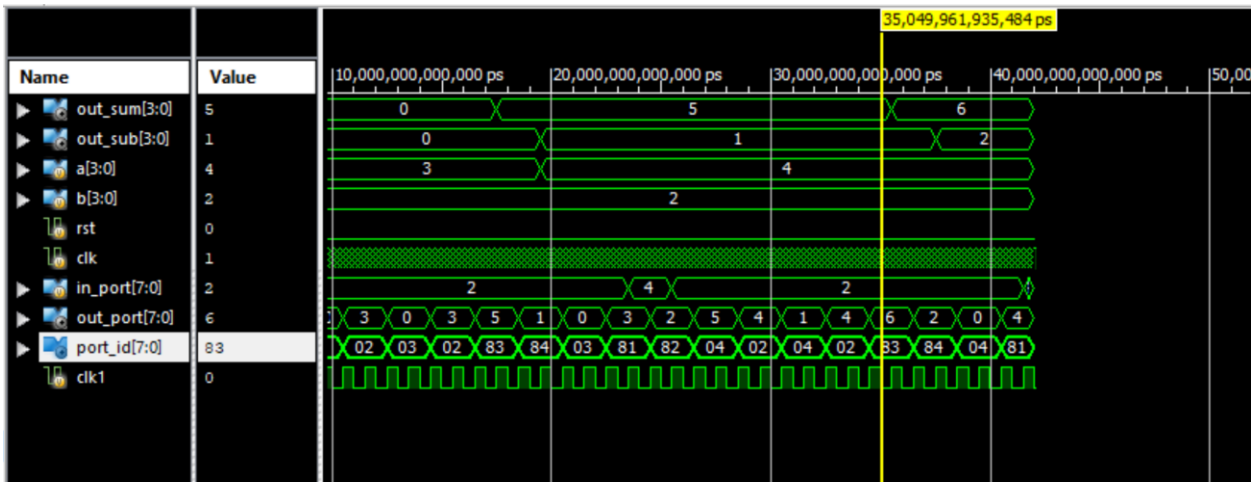
در port_id برابر 84 خروجی تفریق بدرستی در متغیر out_sub قرار داده شده است



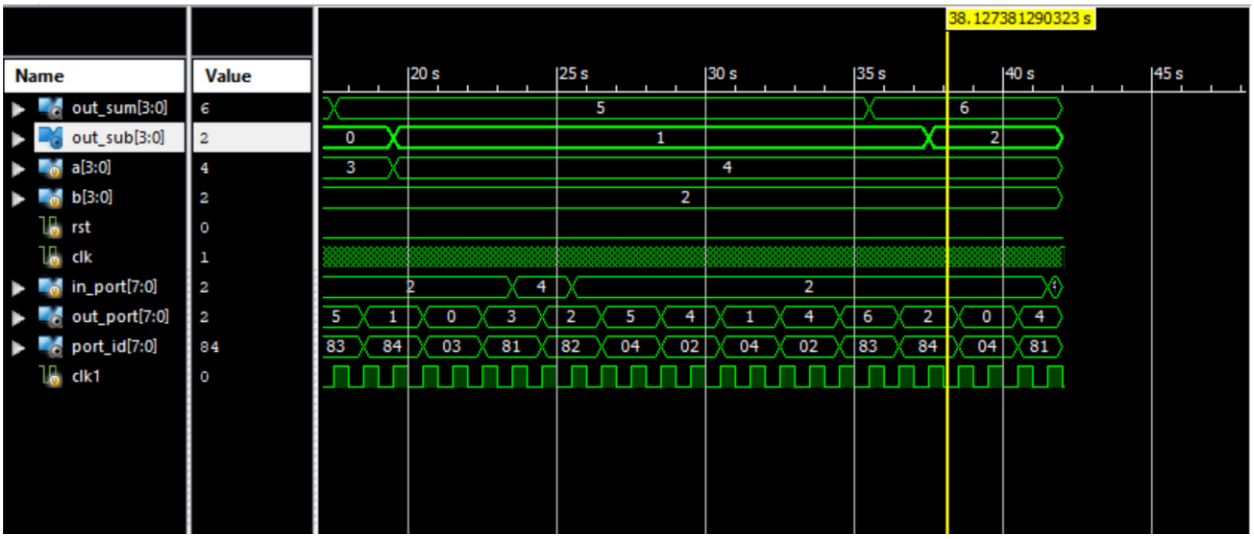
در port_id برابر 81 ورودی جدید a و در 82 ورودی جدید b بدرستی دریافت شده اند a=4,b=2



در port_id برابر 83 خروجی جمع بدرستی در متغیر out_sum قرار داده شده است sum=6

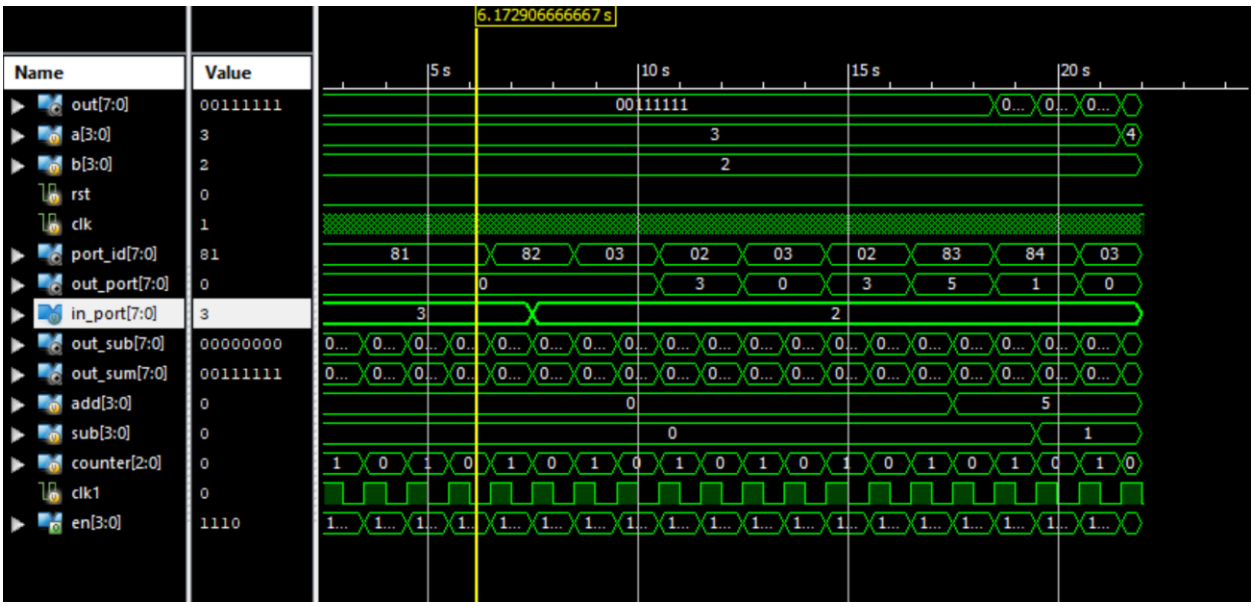


در port_id برابر 84 خروجی تفریق بدرستی در متغیر out_sub قرار داده شده است sub=2



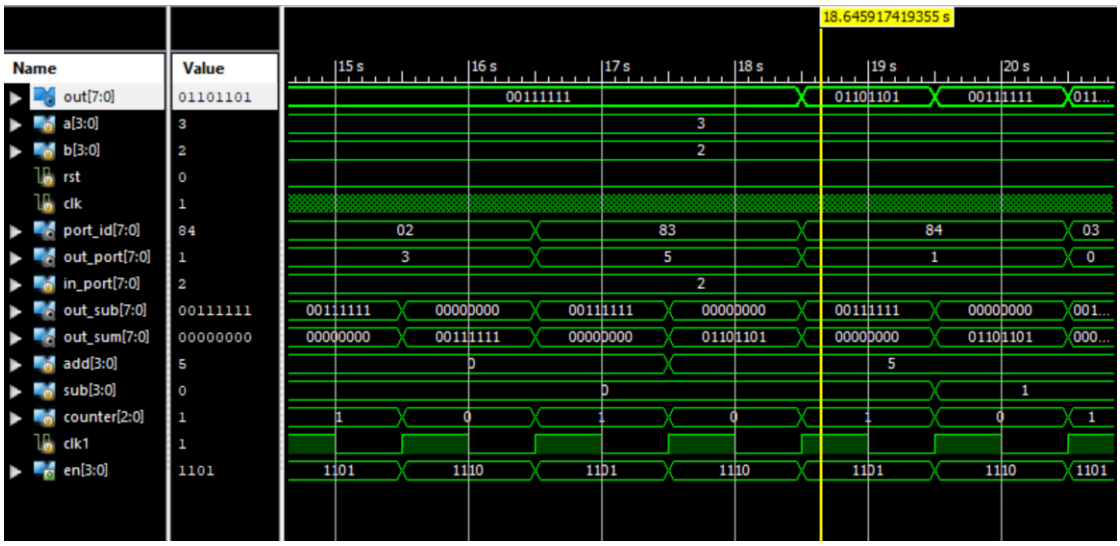
در این قسمت به برسی سوال دو ازمایش میپردازیم که خروجی ماژول یک عدد 8 بیتی و پایه ی 4 بیتی فعال ساز سون سگمنت هاست که 4 بیت است و در ابتدای کار از انجایی که مقدار اولیه متغیر ها 0 بوده است خروجی کد سون سگمنت مربوط به عدد 0 را نمایش میدهد

در port_id برابر 81 اولین ورودی یعنی a و در 82 دومین ورودی یعنی b بدرستی دریافت شده اند



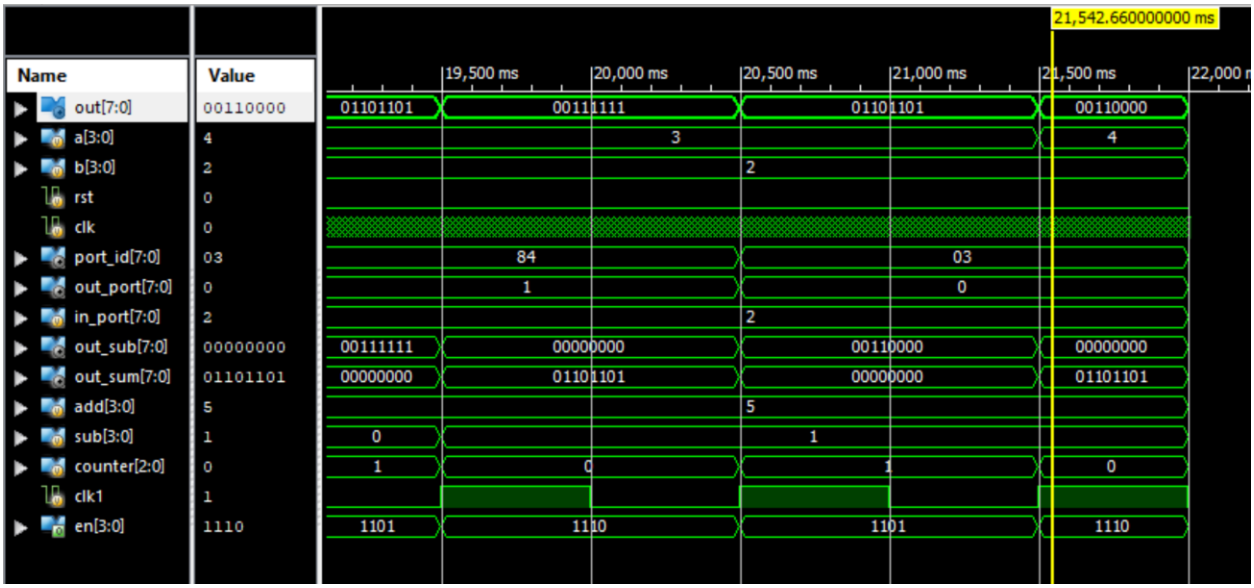
در port_id برابر 83 خروجی جمع بدرستی در متغیر out_sum قرار داده شده است

در این کلاک با اماده شدن حاصل جمع که برابر 5 است و فعال شدن یکی از بیت های en خروجی کد سون سگمنت عدد 5 را نشان میدهد01101101

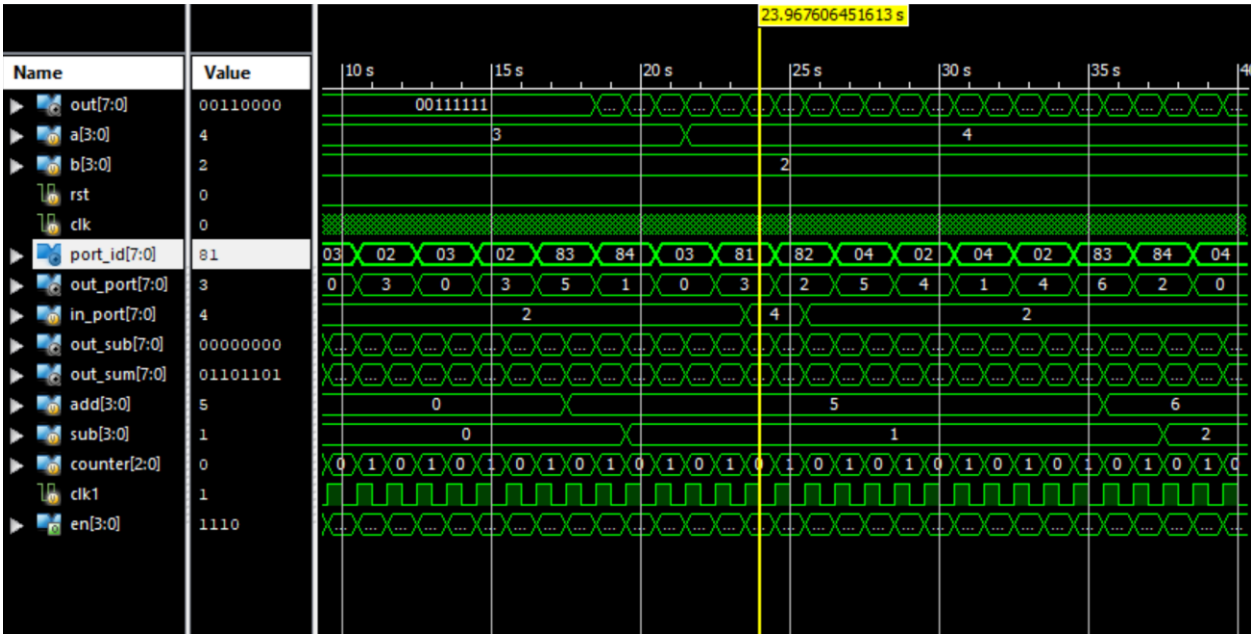


در port_id برابر 84 خروجی تفریق بدرستی در متغیر out_sub قرار داده شده است

حاصل تفریق عدد یک است که معادل سون سگمنت ان 00110000 است و حال بخاطر مالتی پلکسر بودن در یک کلاک مقدار حاصل جمع یعنی 01101101 و در کلاک عدی حاصل تفریق یعنی 00110000 نشان داده میشود

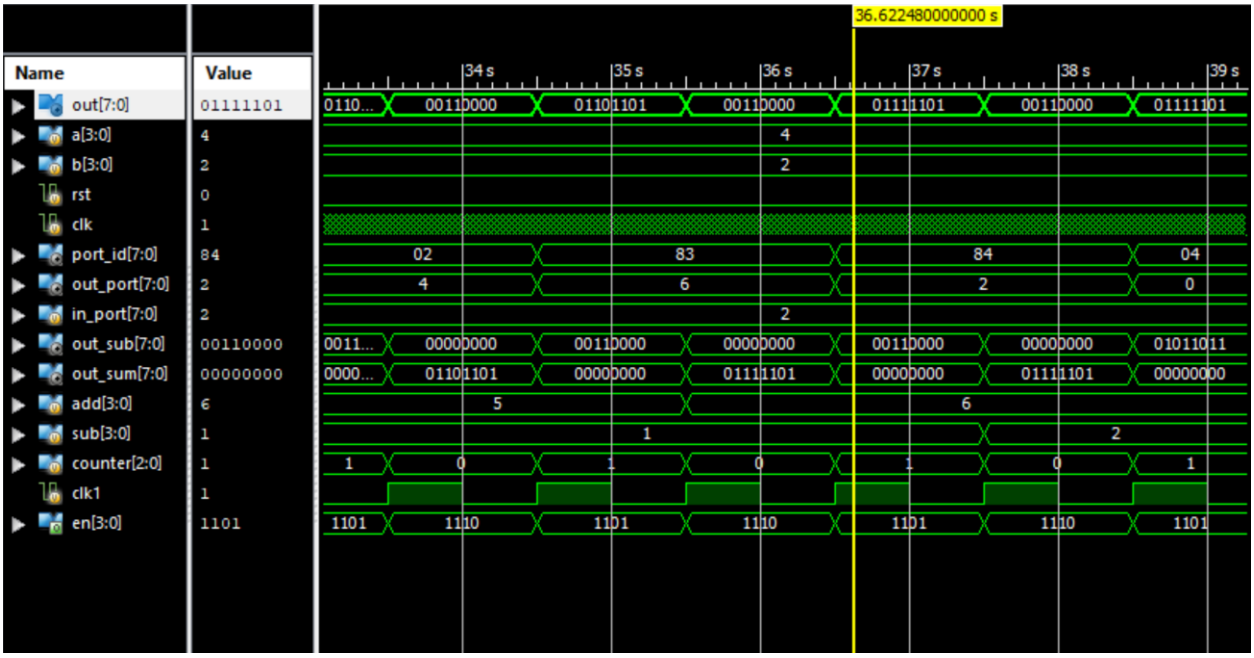


در port_id برابر 81 ورودی جدید a و در 82 ورودی جدید b بدرستی دریافت شده اند a=4,b=2



در port_id برابر 83 خروجی جمع بدرستی در متغیر out_sum قرار داده شده استsum=6

در این کلاک با اماده شدن حاصل جمع که برابر6است و فعال شدن یکی از بیت های en خروجی کد سون سگمنت عدد 6 را نشان میدهد01111101



در port_id برابر 84 خروجی تفریق بدرستی در متغیر out_sub قرار داده شده است sub=2

حاصل تفریق عدد دو است که معادل سون سگمنت ان 01011011 است و حال بخاطر مالتی پلکسر بودن در یک کلاک مقدار حاصل جمع یعنی 01111101 و در کلاک عدی حاصل تفریق یعنی01011011 نشان داده میشود