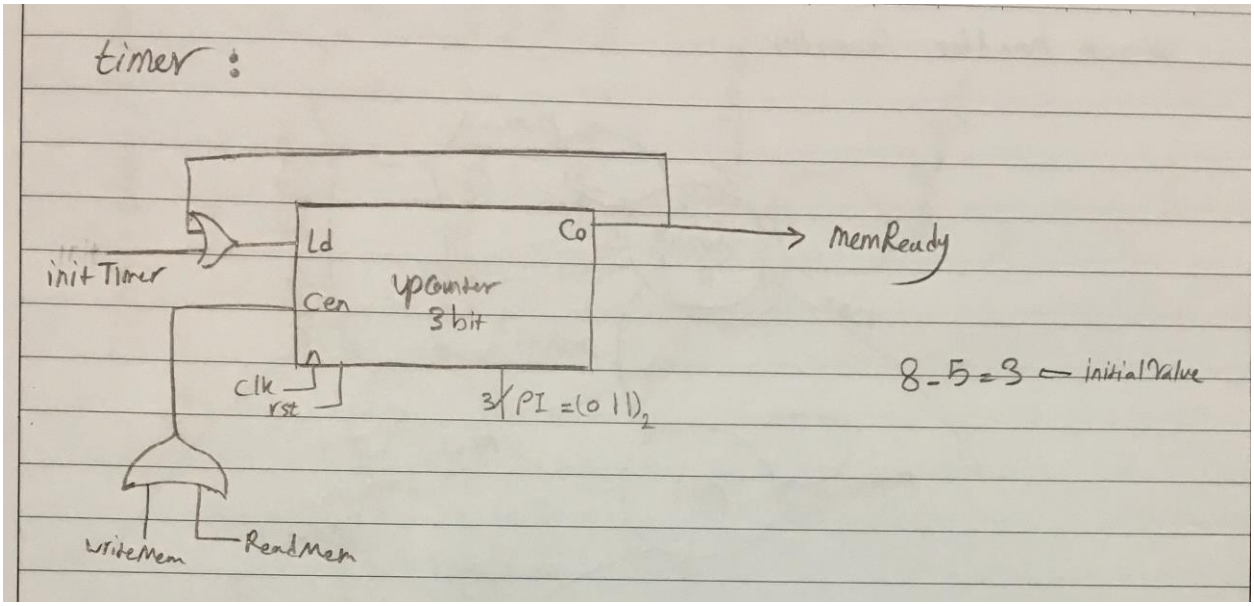
نگین سفاری ۸۱۰۱۹۷۵۲۵
گزارش تمرین کامپیوتری ۴

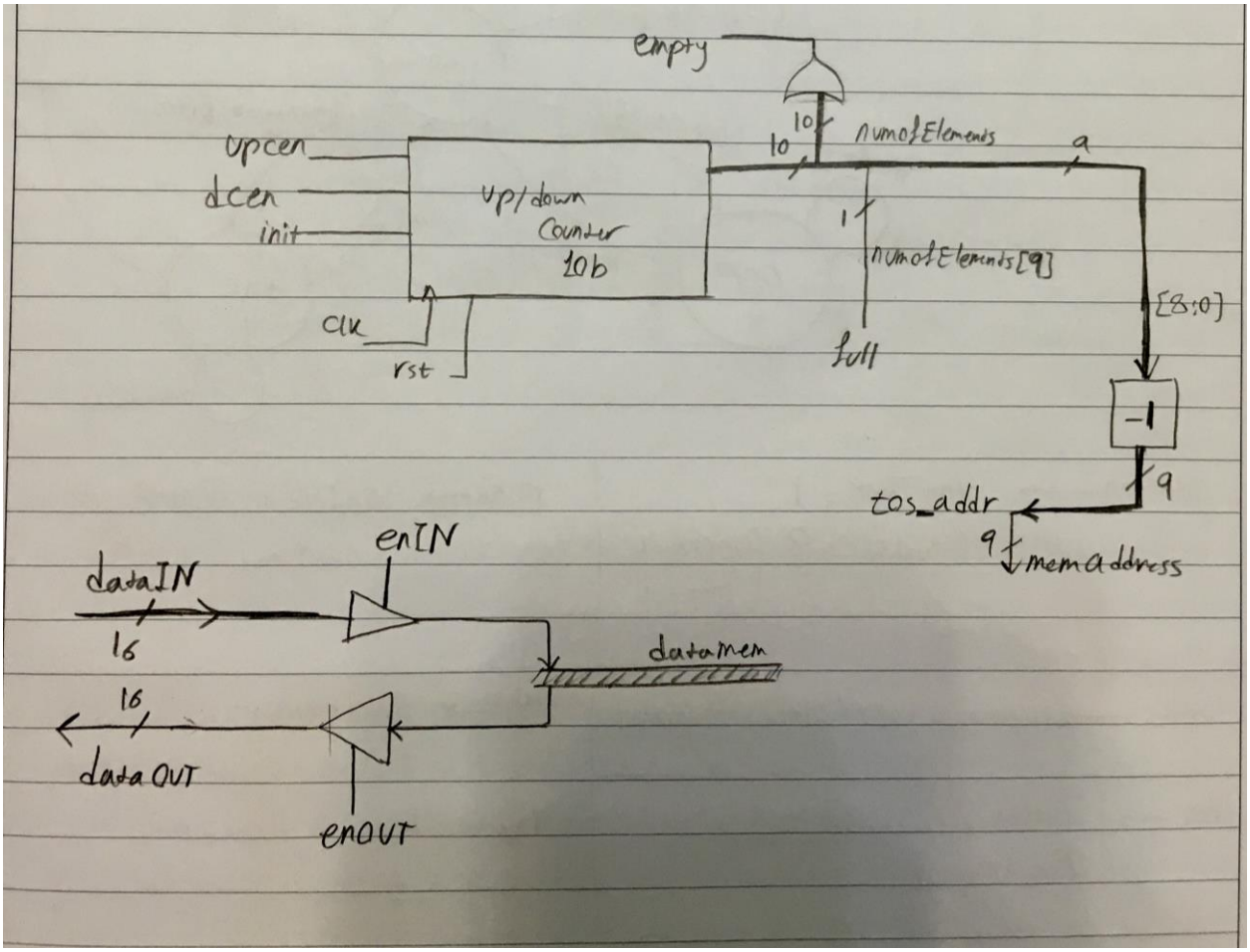
برای قسمت timer از آنجا که می خواهیم در صورت یک شدن سیگنال writeMem و یا سیگنال readMem به اندازه پنج سیکل صبر کنیم، از یک بالا شمار ۳ بیتی استفاده می کنیم. این شمارنده دارای یک ورودی Ld و یک ورودی cen و یک ورودی PI به منظور قرار دادن مقدار شروع شمارش است. اگر سیگنال initTimer و یا سیگنال Co در این شمارنده ۱ شود مقدار PI در شمارنده لود می شود. بخش cen هم در صورتی فعال می شود که یکی از سیگنال های readMem و یا writeMem مقدار یک گیرند.مقدار PI عدد 5-8 که همان 3 است می باشد. خروجی Co در این شمارنده همان memReady است.

در شکل زیر می توان طراحی این timer را دید.



نگین سفاری ۸۱۰۱۹۷۵۲۵
گزارش تمرین کامپیوتری ۴

برای انجام این تمرین Data path استک هندلر را طراحی کردم.

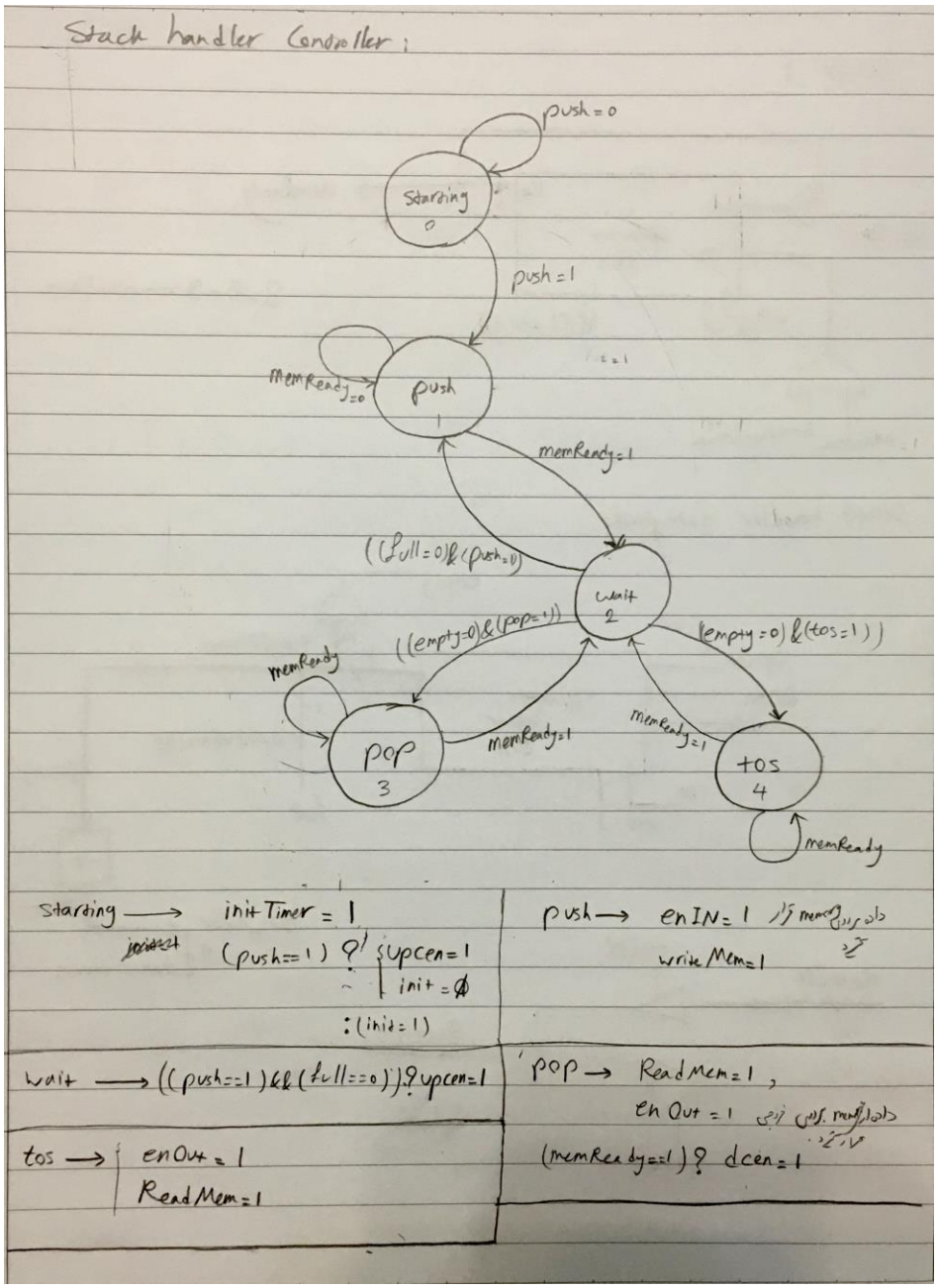


این قسمت شامل یک counter ده بیتی است که هم توانایی up count دارد و هم توانایی down count. این شمارنده برای شمارش تعداد داده های ذخیره شده در حافظه است. اگر عدد خروجی این شمارنده برابر 1000000000 شود یعنی ۵۱۲ داده در حافظه است و کاملا پر می باشد، پس سیگنال empty باید ۱ شود که همان پر ارزش ترین بیت این عدد ۱۰ بیتی است.در صورتی که خروجی تماما صفر باشد به معنای خالی بودن حافظه است پس سیگنال full حاصل or شدن تمامی بیت ها است.از آنجا که آدرس top of stack یکی کمتر از تعداد محتویات حافظه است، پس به کمک یک decrementer ۹ بیتی می توان تعداد المان ها را گرفت و در خروجی آدرس بالای استک را داد که به قسمت address در حافظه متصل می شود. به دلیل آنکه حافظه یک مسیر برای دریافت داده ورودی و قرار دادن داده

نگین سفاری ۸۱۰۱۹۷۵۲۵
گزارش تمرین کامپیوتری ۴

خروجی دارد(inout) به کمک دو عدد بافر می توان data in و data out را به پورت دو طرفه حافظه متصل کرد. به این شکل که هر گاه enIN برابر ۱ شد داده ورودی بر روی inout قرار می گیرد. هر گاه enOUT برابر ۱ شد داده inout بر روی خروجی قرار می گیرد.

کنترل کننده استک هندلر به صورت زیر است.

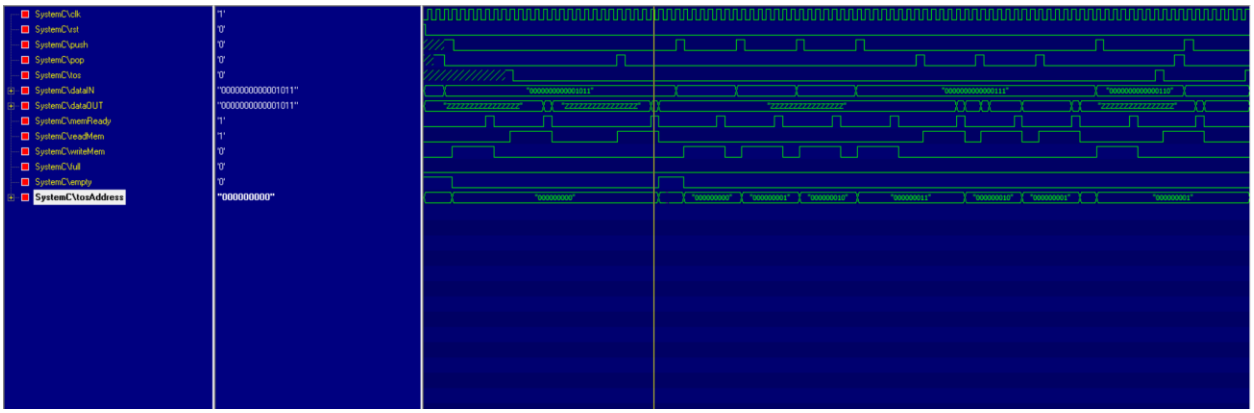


نگین سفاری ۸۱۰۱۹۷۵۲۵
گزارش تمرین کامپیوتری ۴

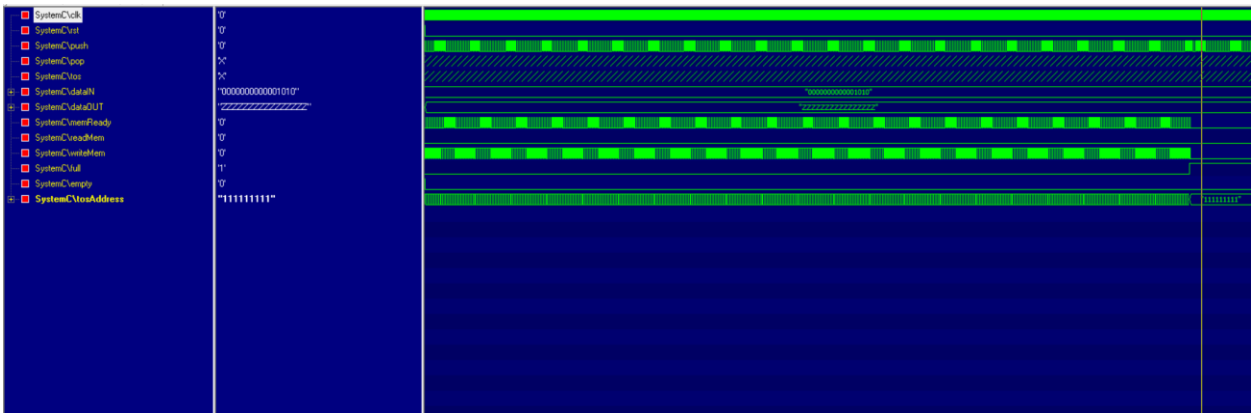
در این کنترل کننده می دانیم که در شروع باید init در تایمر صورت گیرد در همین استیت باید سیگنال init مربوط به شمارنده تعداد محتویات حافظه باید ۱ شود . تنها در صورتی که سیگنال push برابر ۱ شود باید عملیاتی صورت گیرد زیرا در ابتدای کار حافظه کاملا خالی است و تنها کار ممکن push است. به منظور آنکه در آدرس صفر حافظه داده را بنویسیم در همین استیت باید سیگنال upcen در شمارنده ۱۰ بیتی را فعال کنیم. در استیت push باید مسیر enIN یک شود و writeMem یک می شود. پس از آنکه سیگنال memReady مقدار یک گرفت یعنی کار تمام شده و به استیت wait می رویم و منتظر دستور بعدی می مانیم. اگر دستور tos دیده شود و حافظه خالی نباشد، به استیت tos می رویم و مسیر enOUT را فعال میکنیم و ReadMem را یک میکنیم پس از آنکه سیگنال memReady مقدار یک گرفت به استیت wait باز می گردیم و منتظر دستور بعدی می مانیم. در صورتی که حافظه خالی نباشد و دستور pop داشته باشیم، به استیت pop می رویم ودر آن سیگنال readMem یک می شود و مسیر enOUT را فعال می کنیم، در صورتی که memReady یک شود در همین استیت down count در شمارنده ی آدرس را فعال میکنیم و پس از دیدن کلاک هم به استیت wait می رویم و هم از تعداد محتویات ثبت شده برای حافظه و آدرس top of stack یک عدد کم می شود. در استیت wait اگر دستور push دیده شود و حافظه پر نباشد به استیت push رفته می شود و همان کار هایی که در ابتدا گفته شد انجام میگیرد.

قسمت Memory دقیقا به شکل گفته شده در کلاس پیاده سازی شده و ورودی های readMemory و writeMemory دارد و آدرس را در یک پورت ورودی دریافت می کند و یک پورت inout به عنوان گرفتن و دادن داده دارد.

نگین سفاری ۸۱۰۱۹۷۵۲۵
گزارش تمرین کامپیوتری ۴
در تصویر زیر می توان کار کرد این مدار را مشاهده کرد.

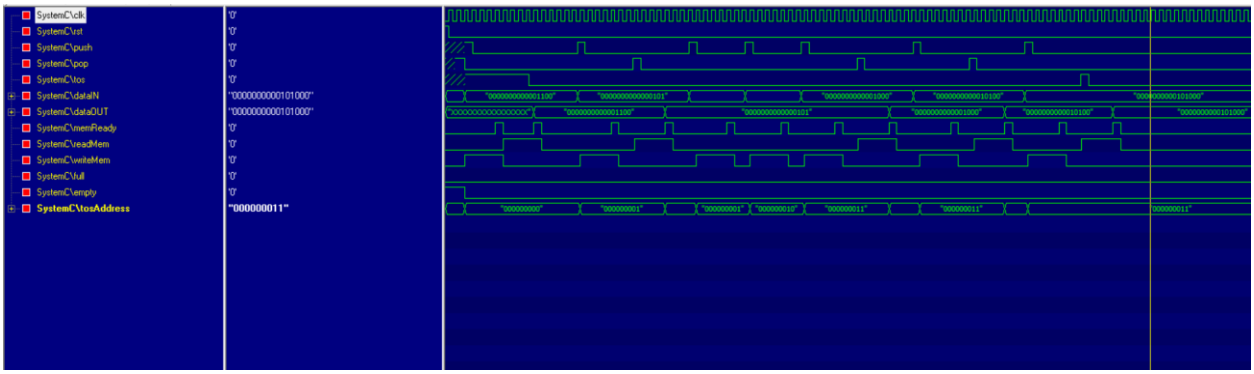


برای آن که فعال شدن سیگنال full در صورت پر شدن حافظه را نشان دهیم از حلقه for استفاده شد. همانطور که در شکل زیر مشخص است، در صورت پر بودن حافظه هر چه قدر که باز دستور pop داده شود به آن توجهی نمی شود.

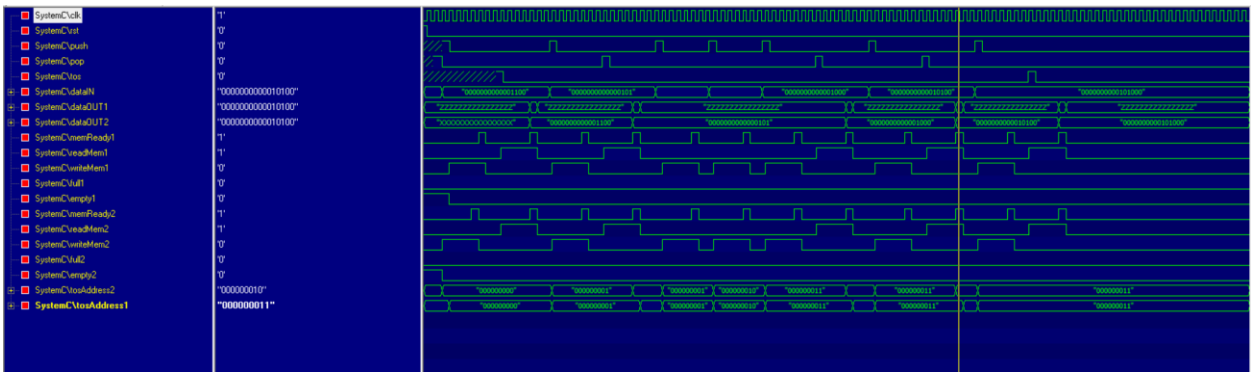


نگین سفاری ۸۱۰۱۹۷۵۲۵
گزارش تمرین کامپیوتری ۴

برای قسمت دوم تمامی کار های قسمت قبل به صورت functional پیاده سازی شد و تنها به اندازه پنج کلاک صبر می کنیم و داده را در حافظه قرار می دهیم و یا بر روی خروجی می گذاریم. و کار بسیار ساده تری بود که در سیگنال های نهایی فرقی با حالت قبل نداشت.



در آخر برای مقایسه دقیق دو حالت با هم تست بنچ دیگری نوشته شد و این دو مدار همزمان با هم تست شدند و نتیجه به صورت زیر است. تمامی سیگنال ها با اندیس ۱ مربوط به حالت RTL و تمامی سیگنال ها با اندیس ۲ مربوط به حالت BFM اند.



می توان دید که در حالت دوم بر خلاف حالت اول از همان زمان اعمال دستور pop یا tos داده خواسته شده بر خروجی است و وقتی memReady ۱ شود قابل برداشت است، اما در حالت اول تنها در بازه ی memReady =1 می توان خروجی را دید. و در کل کارایی یکسانی دارند.