به نام خدا

دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر

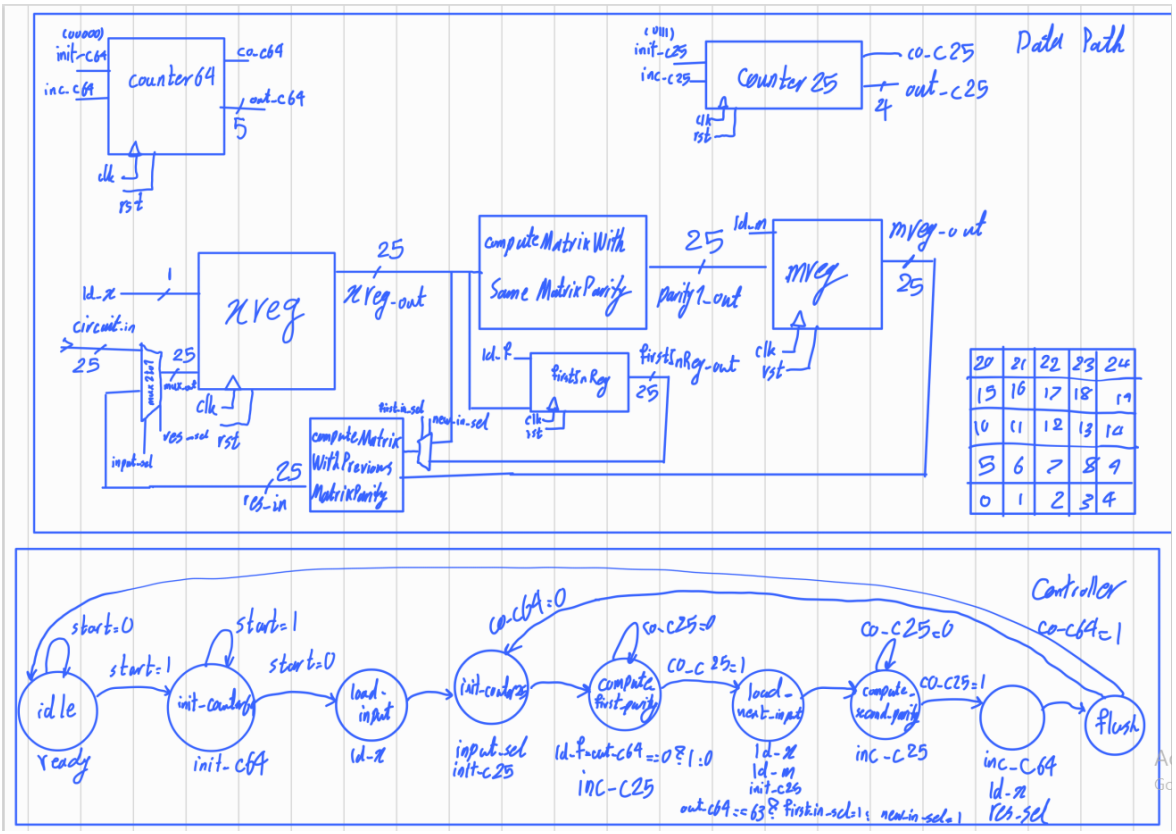
درس طراحی کامپیوتری سیستم های دیجیتال

امتحان میانترم

نام و نام خانوادگی	مهسا همت پناه(810199584) محمد قره حسنلو(810198461)
تاریخ ارسال گزارش	1401/10/12

عنوان ها

فاز اول به همراه تغییرات اعمال شده برای طراحی نهایی



در طراحی جدید نیازی به firstInReg نیست. زیرا میتوانیم با تغییر نحوه ورودی خواندن در تست بنچ، از این رجیستر بی نیاز بشیم.

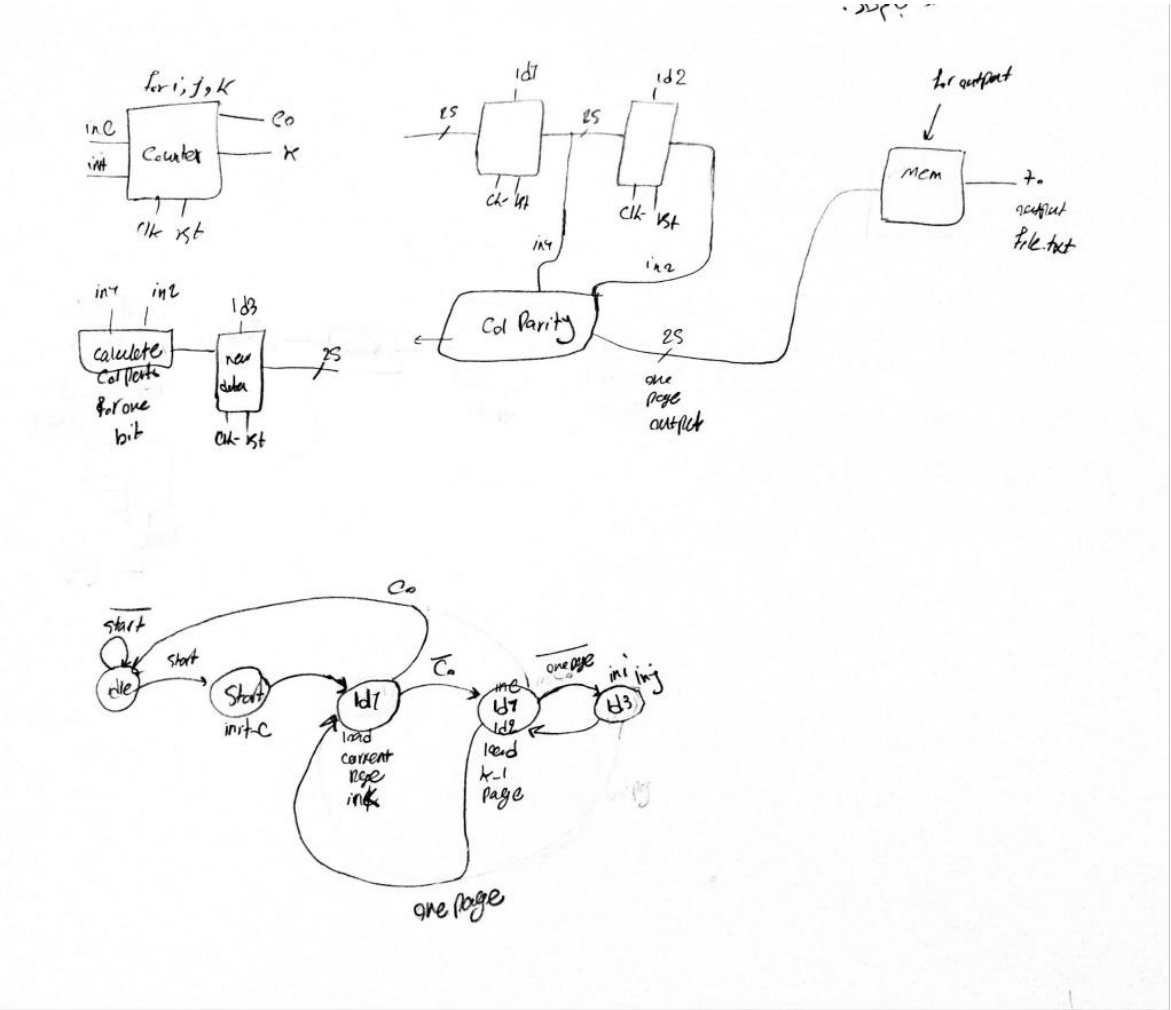
به جای نوشتن خروجی محاسبه شده در xreg و نوشتن آن در همان لحظه در فایل تکست، یک مموری اضافه کردیم که طراحی بهتر، قابل درک و آسان تر باشد و بعد از آنکه همه مقادیر در مموری نوشته شد، شروع به خواندن از مموری میکنیم و مقادیر را در فایل تکست خروجی مینویسیم. برای همین یک متغیر جدید k تعریف کردیم که جایی که از مموری خوانده میشود، از تست بنچ قابل تعیین کردن باشد.

از خروجی های counter25 برای تعیین کردن مقدار i در دو computational component استفاده کردیم تا باعث عدم نیاز استفاده از حلقه for شود.

ورودی ها به صورت جفت به داخل top level design فرستاده میشود و هر بار توسط multiplexer مشخص میشود که کدام یک در رجیستر xreg نوشته شود تا نیازی به استفاده از دو رجیستر برای خواندن ورودی نباشد.

در طراحی جدید نیازی به state آخر طراحی اولیه نبود و با توجه به اضافه شدن مموری، این کار بهتر و با یک state کمتر انجام میشود.

تعداد بیت های خروجی counter یکی کمتر تعیین شده بود.



در این طراحی به این گونه فرض شده است که همان ابتدا بدون اینکه محاسبات داخل ماتریس ابتدایی انجام شود، دو مقدار از ورودی گرفته شده و مقادیر را در رجیسترها ذخیره کرده و سپس در 25 دور با محاسبه i و j انجام شود که خوبی این طراحی نسبت به طراحی نهایی این است که در تعداد state کمتری این عملیات انجام میشود.

در طراحی نهایی دو counter در نظر گرفته شده است که یکی برای جلوگیری از استفاده از for loop است و دیگری برای 64 بار خواندن از ورودی اما در اینجا از یک counter استفاده شده بود؛ چون تمام عملیات در یک computational component انجام میشد و هر دفعه در لوپ پایینی

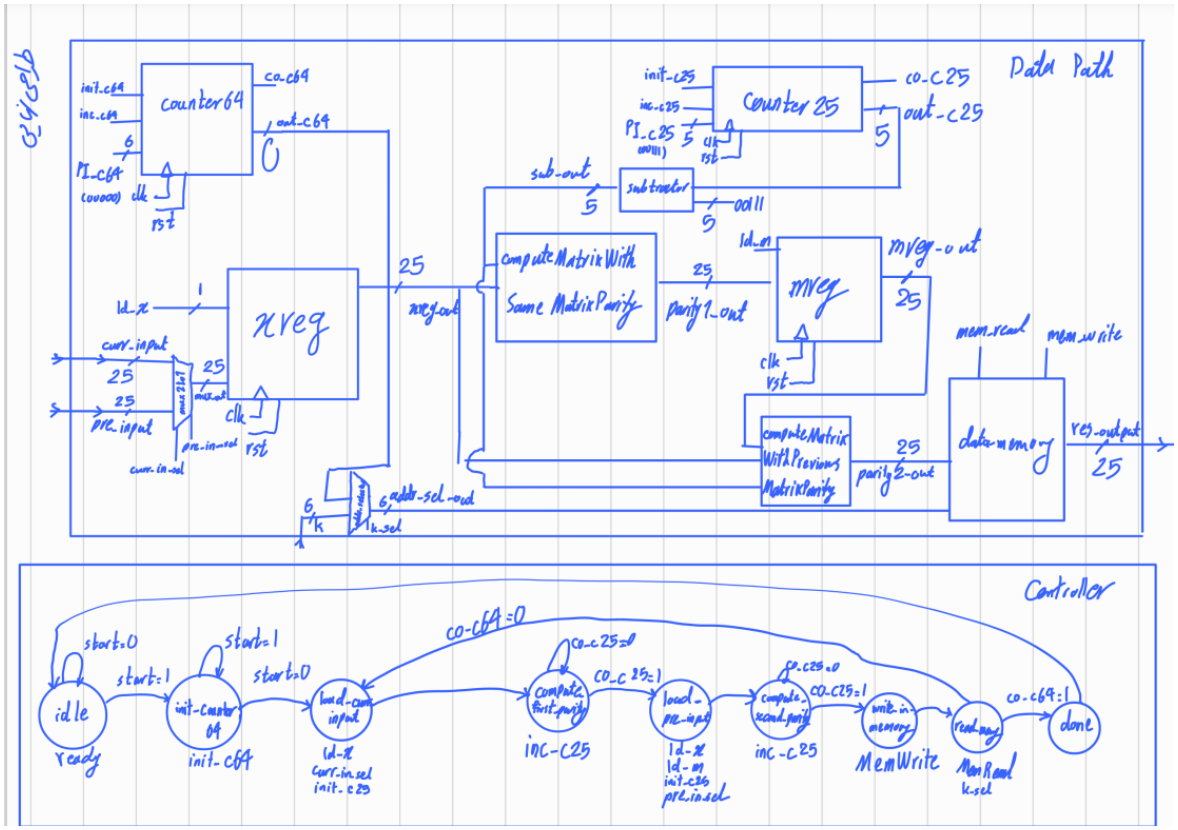
مقدار در 25 تا کلاک حساب میشد و بعد از آنکه مقدار نهایی صفحه آماده میشد، در مموری ذخیره میشد که این ایده نیز در طراحی نهایی استفاده شده است.

ایده استفاده از مموری از این طراحی به دست آمد و ورودی ها به ترتیب از بالا به پایین در این مموری نوشته میشد و در نهایت از طرف تست بنچ خوانده میشد.

در این طراحی استفاده از سه متغیر i، j و k در نظر گرفته شده بود که بعدا در طراحی بعدی، از مقدار j در computational component صرف نظر شد؛ چون مورد استفاده قرار نمیگرفت و تبدیل ورودی به یک ماتریس دو بعدی کافی بود.

در ابتدا فرض شده بود که مانند پروژه از فرمول 5i+j مقدار خانه های هر ماتریس به دست آورده شود اما بعدا تصمیم گرفته شد به همان صورت 25 تایی که در طراحی نهایی هست، عمل شود.

طراحی جدید



توضیح طراحی

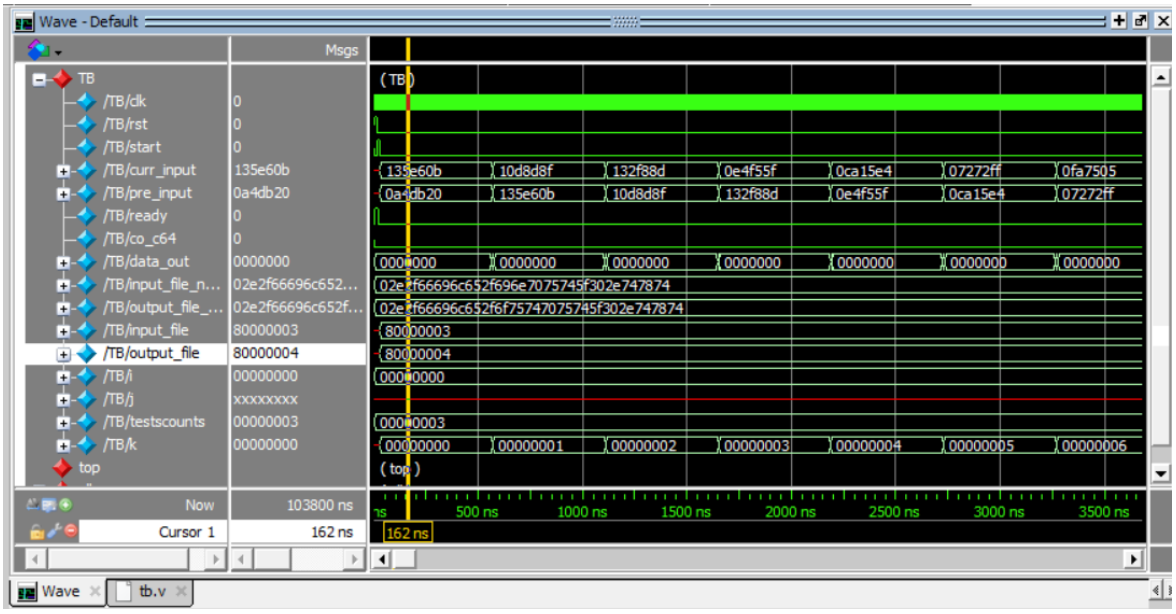
در طراحی جدید، ترکیبی از ایده هر دو دانشجو استفاده شده است و به طور کلی به این صورت است که هر دفعه مقدار سطر i در رجیستر xreg نوشته میشود و parity که در خود این ماتریس است، در computeMatrixWithSameParity انجام میشود و در اینجا خانه ها یکی یکی به ترتیبی که گفته شده، عملیات colParity در آنها انجام میشود و بعد از 25 تا کلاک، مقدار نهایی 25 بیت در mreg نوشته شده و همچنین در این زمان مقدار سطر i-1 در رجیستر xreg نوشته میشود. حال در computeMatrixWithPreviousParity parity که بین دو صفحه هست، انجام شده مقدار نهایی سطر i بعد از 25 کلاک در دسترس بوده و در سطر i در مموری ریخته میشود. این عملیات 64 باز انجام شده و بعد از آن مقدار کامل و حساب شده در مموری در دسترس خواهد بود که در state آخر میتوانیم این مقادیر را از مموری خوانده و در یک فایل تکست بریزیم.

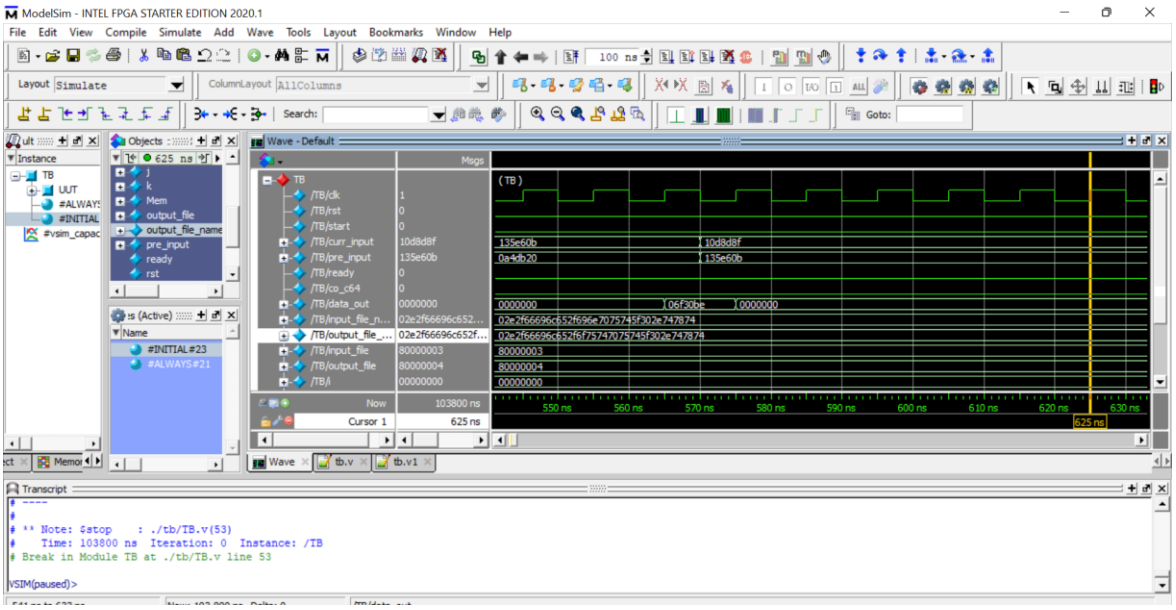
نتایج خروجی

لینک نتیجه خروجی برای تست اول

لینک نتیجه خروجی برای تست دوم

لینک نتیجه خروجی برای تست سوم




ModelSim - INTEL FPGA STARTER EDITION 2020.1
File Edit View Compile Simulate Add Wave Tools Layout Bookmarks Window Help
Wave - Default
Objects
Instance
TB
UUT
Transcript
** Note: $stop : ./tb/TB.v(53)
Time: 103800 ns Iteration: 0 Instance: /TB
Break in Module TB at ./tb/TB.v line 53
VSIM(paused)>