بسمه تعالی

درس طراحی در سطح سیستم

تکلیف کامپیوتری ۱: پیاده‌سازی ضرب ماتریسی با استفاده از ابزار سنتز سطح بالای **Catapult**

دانشکدگان فنی دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر
دکتر بیژن علیزاده
نیم‌سال دوم سال تحصیلی ۱۴۰۴-۱۴۰۳
نگارش: معین فرهادی (moeenfarhadi@ut.ac.ir)

مقدمه

هدف از این تمرین آشنایی با یک ابزار سنتز سطح بالا و بررسی تکنیک‌ها و مراحل سنتز در چنین ابزاری است. برای این منظور، ابزار سنتز سطح بالای شرکت Mentor Graphics با نام Catapult انتخاب شده است. این ابزار یک توصیف سطح بالا به زبان‌های C/C++/SystemC را دریافت کرده و یک توصیف در سطح انتقال ثبات (RTL) به زبان‌های VHDL/Verilog ارائه می‌دهد.

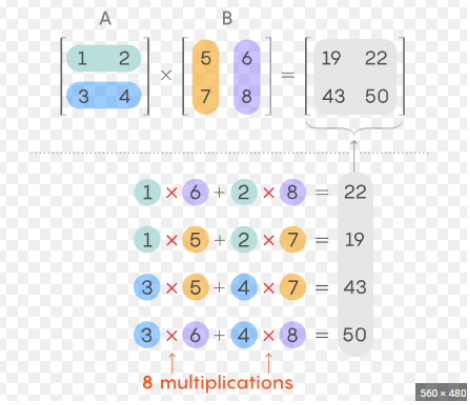
آشنایی با شبکه‌های عصبی RNN و معماری LSTM

شبکه‌های عصبی مکرر (RNN) کلاسی از شبکه‌های عصبی مصنوعی هستند که برای پردازش داده‌های ترتیبی مانند توالی‌های زمانی، متن و گفتار طراحی شده‌اند. برخلاف شبکه‌های عصبی رو به جلو، شبکه‌های عصبی مکرر می‌توانند از وضعیت درونی خود، برای پردازش دنباله‌ی ورودی‌ها استفاده کنند که آن‌ها را برای مواردی نظیر تشخیص صوت، یا تشخیص دست‌نوشته‌های غیر بخش‌بندی شده‌ی متصل مناسب می‌کند.

شبکه‌های عصبی LSTM یک معماری و پیاده‌سازی از شبکه‌های RNN هستند. شبکه عصبی LSTM یک شبکه بازگشتی است که اگرچه جدید نیست، اما یکی از بهترین شبکه‌های بازگشتی محسوب می‌شود. کارکرد این شبکه عصبی مبتنی بر ضرب‌های ماتریسی است و با اینکه موفقیت‌های زیادی در زمینه پردازش صوت و گفتار و غیره بدست آورده است. اما یکی از مشکلات آن حجم بالای محاسبات و طولانی شدن زمان یادگیری است. لذا در سالیان اخیر پژوهش‌های زیادی برای پیاده‌سازی محاسبات این نوع شبکه‌های عصبی روی پلتفرم‌های Multi Core انجام شده است. برای آشنایی بیشتر با این نوع شبکه عصبی به این لینک مراجعه کنید.

همانطور که در شکل ۱ مشاهده می‌کنید، برای انجام ضرب ماتریسی دو ماتریس با ابعاد N * K و K * M به M * N * K ضرب نیاز است. با توجه به این نکته، حجم محاسبات ضرب ماتریسی در شبکه‌های عصبی با ابعاد بزرگ

بسیار زیاد خواهد شد. به همین دلیل پیاده‌سازی سخت‌افزاری با سرعت بالا و توان کمتر مورد توجه قرار گرفته است.



شکل ۱: نمونه‌ای از ضرب ماتریسی

توصیف مساله

هدف از این تمرین پیاده‌سازی ضرب ماتریسی با استفاده از Catapult است. در واقع می‌خواهیم با استفاده از ماتریس‌های ورودی، حاصل خروجی را تولید کنیم. برای این کار با استفاده از چندین حلقه با ابعاد مناسب، عملیات مورد نظر را انجام می‌دهیم.

در این پروژه فرض شده است که ماتریس‌های ورودی در حافظه خارجی قرار دارد و با استفاده از آرایه‌های input_matrix1 و input_matrix2 در دسترس قرار می‌گیرند. ابعاد ماتریس‌های موجود در فایل MatMul.cpp به صورت پارامتری تعریف شده‌اند. شما باید برای انجام پروژه متغیرهای M و N و K را به طور مناسب با توجه به شماره دانشجویی خود تغییر دهید.

سه رقم پایانی شماره‌دانشجویی شما: ABC

$N = 5 + (ABC \bmod 10), M = 5 + (ABC \bmod 9), K = 5 + (ABC \bmod 8)$

فرض می‌شود که هر خانه از ماتریس‌های ورودی ۱۶بیتی است. لذا با توجه به این موضوع و همچنین مقدار K می‌توان تعداد بیت‌های خروجی را مشخص نمود که در کد داده شده ، به صورت پارامتری توسط متغیر OUTPUT_BITS مشخص شده است. درستی این رابطه پارامتری را بررسی نمایید.

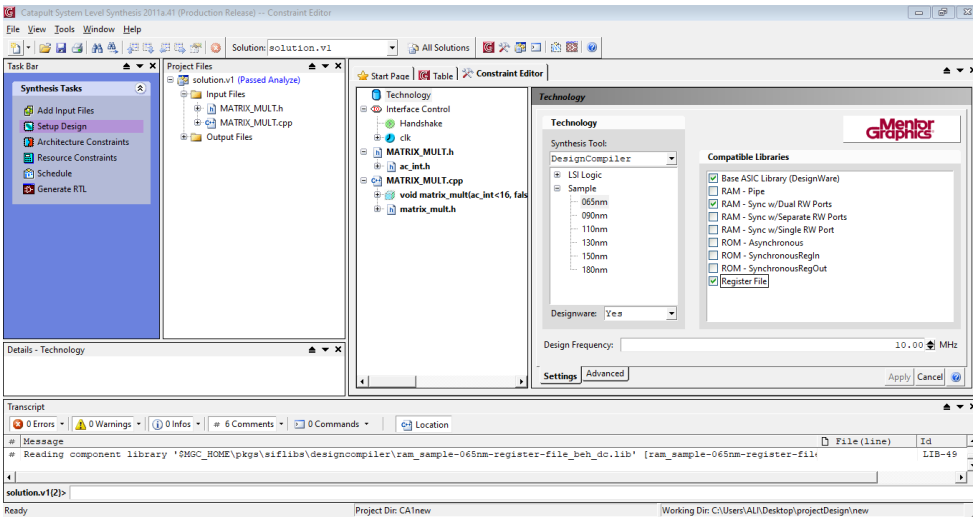
نتیجه محاسبه شده به عنوان خروجی روی سیگنال output قرار می‌گیرد. همچنین آدرس هر خانه از ماتریس خروجی که نتیجه‌اش آماده شده است، بر روی سیگنال addr قرار می‌گیرد. یک بودن سیگنال output_valid نیز به معنای آماده بودن داده است. هدف این برنامه این است که پس از خواندن ورودی‌ها از حافظه، عملیات ضرب ماتریسی بر روی آن‌ها انجام شده و نتایج در حافظه ذخیره شوند.

جهت آشنایی بیشتر با این مسئله توصیف سطح بالایی به نام MATRIX_MULT در اختیار شما قرار گرفته است. MATRIX_MULT از حلقه‌های تو در تو تشکیل شده که عملیات ضرب ماتریسی را انجام می‌دهد. این توصیف سطح بالا باید سنتز شده و نتایج خواسته شده گزارش شوند. در این راستا مراحل زیر را انجام داده و نتایج و تحلیل‌های خود را گزارش کنید .

خواسته‌های مساله

گام اول

یک پروژه جدید بسازید و فایل MATRIX_MULT را به آن اضافه کنید. سپس از زبانه‌ی Tools به بخش Set Options رفته و در بخش Output گزینه های SystemC و Verilog را فعال کنید و بر روی OK کلیک کنید. حال از قسمت Synthesis Tasks مرحله‌ی Setup Design را انتخاب کرده و با انتخاب Technology تنظیمات را مطابق شکل ۲ انجام دهید.



شکل ۲: تنظیمات Technology

گام دوم

فرکانس کاری مدار را روی ۱۰MHz تنظیم کنید. با انتخاب Generate RTL از قسمت Synthesis Tasks، یک بار پروژه را سنتز کنید. حال مقادیر پارامترهای خروجی طراحی مانند Latency Cycle، Latency Time، Throughput Cycle، Total Area و Slack را از Table نتایج در Catapult گزارش کنید. (۵ نمره)

این پارامترها را توصیف کنید. (۵ نمره)

گام سوم

با استفاده از نتایج گام دوم مشخص کنید که فرکانس کاری مدار بدون ایجاد تغییر در تنظیمات، تا چه مقدار قابل افزایش است؟ دلیل خود را ارائه کنید و پس از تغییر فرکانس کاری مدار پارامترهای خروجی طراحی را گزارش کنید. تغییرات نتایج نسبت به گام دوم را تحلیل کنید. (۱۰ نمره)

حال با استفاده از نرم‌افزار Quartus ، فایل سنتز شده را برای Cyclone II EP2C70F896C6 سنتز کنید و F_{max} حاصل را گزارش کرده و با جواب بخش قبل مقایسه کنید. (۵ نمره)

گام چهارم

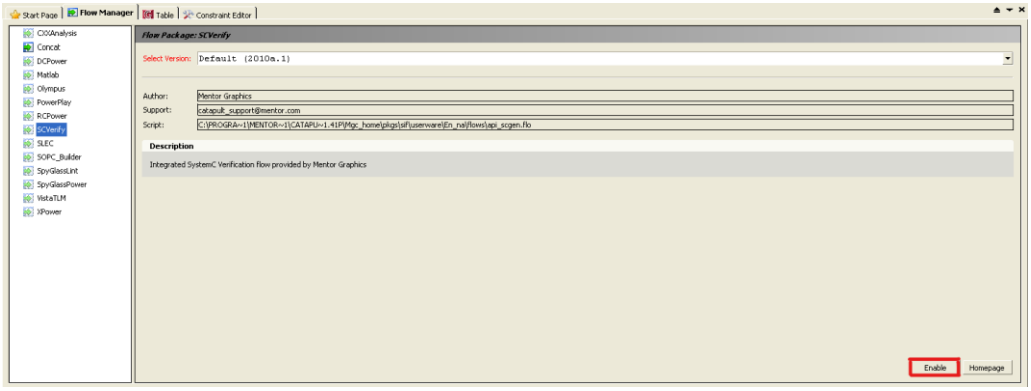
در این گام قصد داریم برای کد Verilog و SystemC حاصل از سنتز، درستی‌سنجی (Verification)انجام دهیم.

الف) ابتدا از بخش Flow طبق شکل ۳، این قابلیت را فعال کنید. مانند شکل ۴ بخش Verification به پروژه‌ی شما اضافه خواهد شد. برای درستی‌سنجی ابتدا باید یک Testbench به زبان C++ بنویسید به طوری که خروجی ماژول را به ازای چند ورودی متفاوت چاپ کند. برای این کار، می‌توانید از کدی که در اختیار شما قرار گرفته است، کمک بگیرید. (۱۵ نمره)

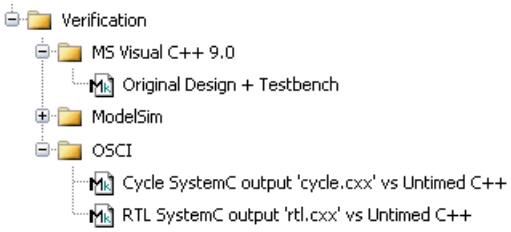
ب) بعد از نوشتن Testbench، بار دیگر پروژه را سنتز کنید تا تغییرات اعمال شده و Testbench شما توسط Catapult شناخته شود.

ج) سپس ابتدا با استفاده از گزینه‌ی Original Design + Testbench مطابق شکل ۴ ، کد اولیه را درستی‌سنجی کنید. در مرحله‌ی بعد، با استفاده از آیتم‌های بخش OSCI کد SystemC سنتز شده را درستی‌سنجی و نتیجه آن را گزارش کنید. (۱۰ نمره)

د) حال با نوشتن یک Testbench در Verilog، فایل‌های سنتز شده را درستی‌سنجی کنید. نتایج را گزارش کرده و با نتایج بخش (ج) مقایسه کنید. (۱۵ نمره امتیازی)



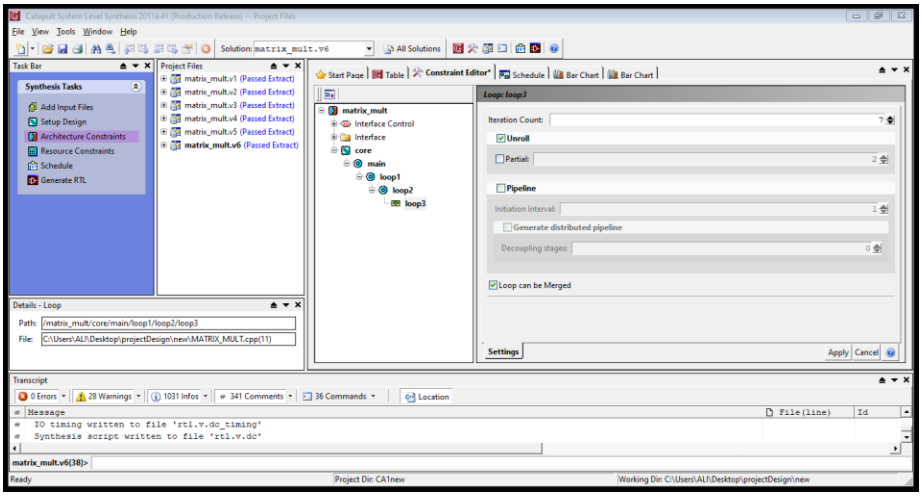
شکل ۳: فعال سازی SCVerify



شکل ۴: گزینه‌های درستی‌سنجی

گام پنجم

مجددا فرکانس کاری مدار را روی ۱۰ MHz تنظیم نمایید. مطابق شکل ۵، حلقه داخلی (loop3) را Unroll کنید. تغییرات را Apply کرده و پارامترهای خروجی طراحی را گزارش کنید و تغییرات نتایج نسبت به گام دوم را تحلیل کنید. برای تحلیل دقیق‌تر می‌توانید از گزارش‌های موجود در پوشه Reports از پوشه Output Files استفاده کنید. (۵ نمره)



شکل ۵: تنظیمات unroll و pipeline

حال تنظیمات را به حالت قبل برگردانده و این بار سعی کنید به صورت دستی و با اعمال تغییرات در کد سطح بالا، حلقه داخلی (loop3) را Unroll کنید. نتایج را با قسمت قبل مقایسه کنید و در صورت وجود تفاوت، علت را بیان کنید. (۱۰ نمره)

گام ششم

تنظیمات را به گام دوم برگردانید و این بار با هدف کاهش تاخیر و افزایش Throughput از امکان Loop Pipelining و Loop Unrolling در بخش تنظیمات Architecture Constraints برای هر حلقه به صورت جداگانه استفاده کنید. نتایج را گزارش و تحلیل خود بیان کنید. برای این بخش باید ۶ نتیجه و تحلیل جداگانه ارائه دهید.

با استفاده از مسیر View -> Other Windows -> Bar Chart نمودار Timing و Area Score را برای این ۶ مورد رسم کنید و بیان کنید که در کل کدام حالت از نظر شما برای طراحی مناسب‌تر است؟ چرا؟ (۲۰ نمره)

گام هفتم

تنظیمات مربوط به حلقه‌ها را همانند گام پنجم اما با هدف کمینه‌کردن مساحت تغییر دهید. نتایج را همراه با تحلیل خود گزارش دهید. (۱۵ نمره)

راهنمایی: با بررسی اثر Loop Pipelining و Loop Unrolling بر روی مساحت در گام قبل تنظیمات را به گونه‌ای انجام دهید که مساحت کمینه شود.

نکات تحویل

- بارم‌بندی سوالات
  - گام دوم: ۱۰ نمره
  - گام سوم: ۱۵ نمره
  - گام چهارم: ۴۰ نمره
  - گام پنجم: ۱۵ نمره
  - گام ششم: ۲۰ نمره
  - گام هفتم: ۱۵ نمره
  - نگارش خوانا و مرتب بودن فایل‌های ارسالی: ۵ نمره
- توجه شود که نمره تمرین از ۱۲۰ بوده و ۲۰ نمره امتیازی در نظر گرفته شده است.
- انجام این تمرین کامپیوتری به صورت **انفرادی** میباشد.
- گزارش و فایل‌های شبیه‌سازی و سایر ضمائم را با ترتیب نام‌گذاری زیر در سامانه یادگیری الکترونیکی ایلرن بارگذاری نمایید.

<CA1>_<Student ID>_<Last Name>.zip

- در صورت وجود هرگونه ابهام یا سوال، می‌توانید از طریق ایمیل یا سامانه ایلرن سوالات خود را مطرح نمایید.

با آرزوی بهترین‌ها برای شما