بسمه تعالی

درس مدارهای منطقی و سیستم‌های دیجیتال

پروژه امتیازی پایانی

پیاده‌سازی واحد تقسیم‌کننده

دانشکدگان فنی دانشگاه تهران
دانشکده مهندسی برق و کامپیوتر
دکتر بیژن علیزاده
نیم‌سال اول سال تحصیلی ۰۴-۱۴۰۳
دستیار آموزشی: محمد حسین نیکخواه قمی، سید صدرا قوامی
mhr.nikkhah@gmail.com , sadra.ghavami@gmail.com

مقدمه:

هدف از این تمرین، طراحی سیستمی یک تقسیم‌کننده تقریبی ممیز ثابت به کمک ضرب‌کننده می‌باشد. در بخش اول شما یک ضرب‌کننده add and shift طراحی و پیاده‌سازی می‌کنید. در ادامه، به کمک این ضرب‌کننده و روش نیوتون-رافسون[۱] ماژول محاسبه معکوس یک عدد را پیاده می‌کنید.

اعداد ممیز ثابت[۲] نحوه‌ای از نمایش اعداد اعشاری به صورت دودویی است. در این نوع نمایش، عدد به دو قسمت صحیح[۳] و اعشاری[۴] تشکیل می‌شود که به کمک (.) از یکدیگر تفکیک می‌شوند. ارزش عدد n ام پس از اعشار از مرتبه 2^{-i} است. برای مثال عدد 0.1 برابر 0.5، عدد 0.01 برابر 0.25 و عدد 10.11 برابر 2.75 می‌باشد. به مثال زیر دقت کنید:

$$\begin{matrix} 1 & 0 & 0 & . & 1 & 0 & 1 \\ 2^2 & 2^1 & 2^0 & & 2^{-1} & 2^{-2} & 2^{-3} \end{matrix}$$

$$1 * 2^2 + 1 * 2^{-1} + 1 * 2^{-3} = 4.625$$

[۱] Newton-Raphson
[۲] Fixed-Point Numbers
[۳] Integer
[۴] Fractional

تعیین تعداد اعشار در سخت افزار قراردادی است، زیرا عملا اعداد باینری بدون در نظر گرفتن اعشار با تقسیم به مقدار ((2^(-f)) که در آن f تعداد بیت اعشار است، به مقدار حقیقی تبدیل می‌شوند. در مثال قبلی، در صورتی که ممیز را حذف کنیم به عدد 100101 یا همان 37 می‌رسیم که اگر بر 8 تقسیم شود، همان 4.625 خواهد بود. لذا وجود بخش اعشاری در محاسبات و همچنین در سخت افزار محسوس نیست، اما در هنگام طراحی باید به آن توجه شود تا مرتبه درستی از اعداد در نظر گرفته شوند.

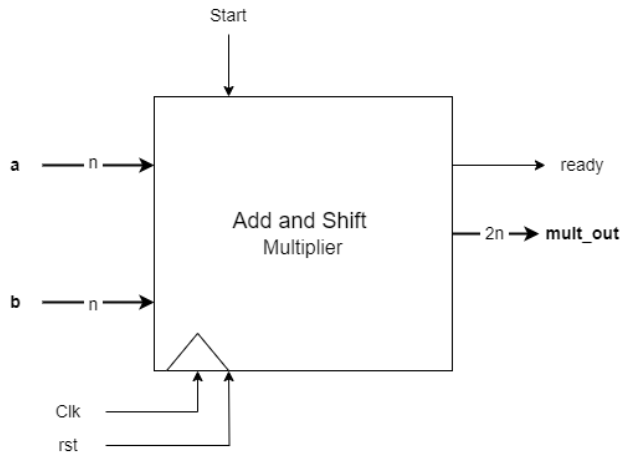
رابطه نیوتون-رافسون برای محاسبه معکوس یک عدد: فرمول زیر رابطه ارائه شده برای محاسبه معکوس یک عدد به روش نیوتون رافسون می‌باشد:

Initialize y_0 with an initial guess
for i = [0:n-1] do {
$y_{i+1} <= y_i * (2.0 - x_i * y_i)$
}

ممیز برای دقت بهتر نتیجه نهایی، لازم است تا حدس اولیه (y_0) به مقدار واقعی $\frac{1}{x}$ نزدیک باشد. از آنجایی که در این تمرین، هدف محاسبه معکوس اعداد بین ۱ و ۲ است، مقدار حدس اولیه را ثابت و برابر ۱ در نظر می‌گیریم.

Add and Shift Multiplier

در این قسمت شما می‌بایست سخت افزار ضرب‌کننده add and shift را طراحی و پیاده‌سازی کنید. دیاگرام این ماژول به صورت شکل ۱ است.



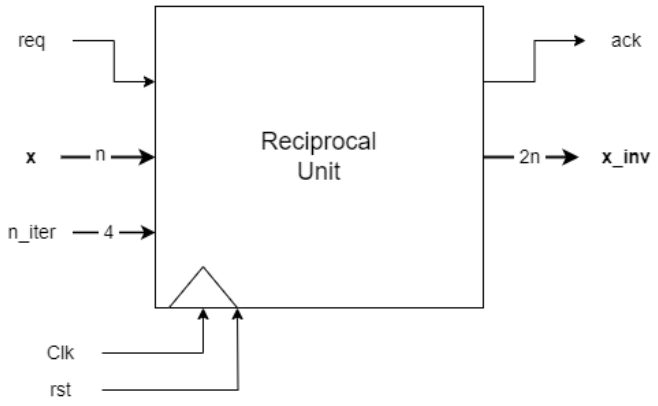
شکل ۱- دیاگرام ماژول Add and Shift

این واحد با دریافت سیگنال کنترلی start شروع به انجام ضرب کرده و با آماده شدن مقدار خروجی، سیگنال ready را فعال می‌کند و تا وقتی مقدار start صفر نشود خروجی را نگه داشته و منتظر می‌ماند. عرض بیت ورودی‌های a و b پارامتری هستند.

۱. مسیرداده و واحد کنترل مربوط به ضرب‌کننده add and shift با ورودی های ۸ بیتی و خروجی ۱۶ بیتی را طراحی کنید.
۲. ساختار ارائه شده برای این ضرب‌کننده را در Verilog بنویسید. توجه شود که طول بیت ورودی، باید پارامتر در نظر گرفته شود و ثابت نیست (از parameter در Verilog استفاده کنید). برای عدم افت دقت، عرض بیت خروجی را همواره دو برابر عرض بیت ورودی در نظر بگیرید.
۳. برای ماژول طراحی شده تست بنچ بنویسید و نتایج ضرب را با مقدار واقعی آن مقایسه و گزارش کنید. برای عرض بیت ۸ و ۱۶ بیت، هر کدام ۵ نمونه ضرب را تست کرده و گزارش کنید.
۴. توضیح دهید چطور این ساختار توانایی محاسبات اعداد fixed-point را دارد. دو نمونه ضرب اعداد اعشاری را مثال زده و نتیجه ضرب‌کننده را با مقدار واقعی مقایسه کنید.

Reciprocal:

در این قسمت به کمک ماژول ضرب‌کننده قسمت قبل، واحدی طراحی می‌کنید که با استفاده از رابطه نیوتون-رافسون، معکوس اعداد fixed-point بین ۱ تا ۲ را محاسبه کند. از آنجایی که این اعداد حداکثر ۲ هستند، تنها یک بیت برای بخش صحیح و سایر بیت ها برای بخش اعشاری هستند. شکل ۲ دیاگرام مربوط به این ماژول را نشان می‌دهد.



شکل ۲- دیاگرام ماژول Reciprocal

این ماژول با دریافت سیگنال req و گرفتن مقدار عدد اعشاری x که بین ۱ تا ۲ است، شروع به محاسبه می‌کند و به تعداد n بار، رابطه نیوتون-رافسون را محاسبه می‌کند و پس از اتمام کار، سیگنال ack را فعال کرده و مقدار محاسبه شده را روی خروجی x_inv قرار می‌دهد. تعداد بیت ورودی پارامتری بوده و عرض بیت خروجی، دو برابر ورودی است.

۱. مسیرداده و واحد کنترل این ساختار را تنها با داشتن یک ضرب‌کننده و برای ورودی ۸ بیتی ارائه دهید. حداکثر تعداد دفعات تکرار محاسبه را برابر n_iter=16 بگیرید.
۲. ساختار طراحی‌شده را در Verilog پیاده‌سازی کنید. تعداد بیت ورودی به صورت پارامتری و تعداد بیت خروجی دو برابر تعداد بیت ورودی در نظر گرفته شود. همچنین تعداد دفعات تکرار را نیز به صورت ورودی در نظر بگیرید.
۳. برای تست ماژول خود تست بنچ بنویسید و مقدار نتیجه شده از سیستم را با مقدار دقیق معکوس به ازای ۵ عدد اعشاری متفاوت مقایسه و گزارش کنید.

Synthesis and Post-Synthesis Simulation

۱. در نرم افزار Quartus، یک پروژه بسازید. Device هدف برای سنتز، از خانواده Cyclone II انتخاب شود. همچنین در هنگام ساخت پروژه در صفحه EDA Tools Settings و بخش Simulation، گزینه Modelsim-Altera را انتخاب کنید.
۲. ماژول تقسیم‌کننده خود را به کمک نرم‌افزار Quartus، برای ساختار ۸ بیتی،۱۶ بیتی و ۳۲ بیتی سنتز کرده و نتایج سنتز را گزارش کنید. تعداد رجیسترهای استفاده شده چقدر است؟ صحت تعداد رجیسترها را با طراحی خود بسنجید و توجیه کنید. هم چنین فرکانس clock را برای هر کدام از بخش زیر به‌دست آورید:

TimeQuest Timing Analyzer > Slow Model > Fmax Summary

۳. پس از compile در پوشه پروژه Quartus، یک پوشه به نام Simulation تولید شده است که حاوی دو فایل با پسوند vo. و sdo. می‌باشد. فایل vo. دارای کد Verilog پس از سنتز برای Post-Synthesis Simulation است. فایل sdo. کتابخانه مربوط برای اجرای این کد Verilog می‌باشد. برای شبیه‌سازی نتیجه پس از سنتز، در Modelsim یک پروژه بسازید و برای این ماژول یک تست بنچ بنویسید. برای شبیه‌سازی از پنجره simulation >> start simulation، سربرگ Libraries را انتخاب کرده و در بخش

Search Libraries دو کتابخانه cycloneii_ver و altera_ver را انتخاب کنید. سپس به سربرگ SDF رفته و در بخش SDF Files، فایل sdo. دریافتی از Quartus را اضافه کنید. نتایج شبیه سازی تقسیم کننده را با شبیه‌سازی پیش از سنتز مقایسه کرده و گزارش کنید.

بارم‌بندی سوالات

- Add and Shift Multiplier : ۱۰ نمره
- Reciprocal Unit : ۵۰ نمره
- Synthesis and Post Synthesis Sim. : ۳۰ نمره
- فرمت صحیح گزارش و پوشه‌بندی فایل‌ها : ۱۰ نمره

با آرزوی بهترین‌ها برای شما