تمرین اول مدارهای منطقی برنامه‌پذیر

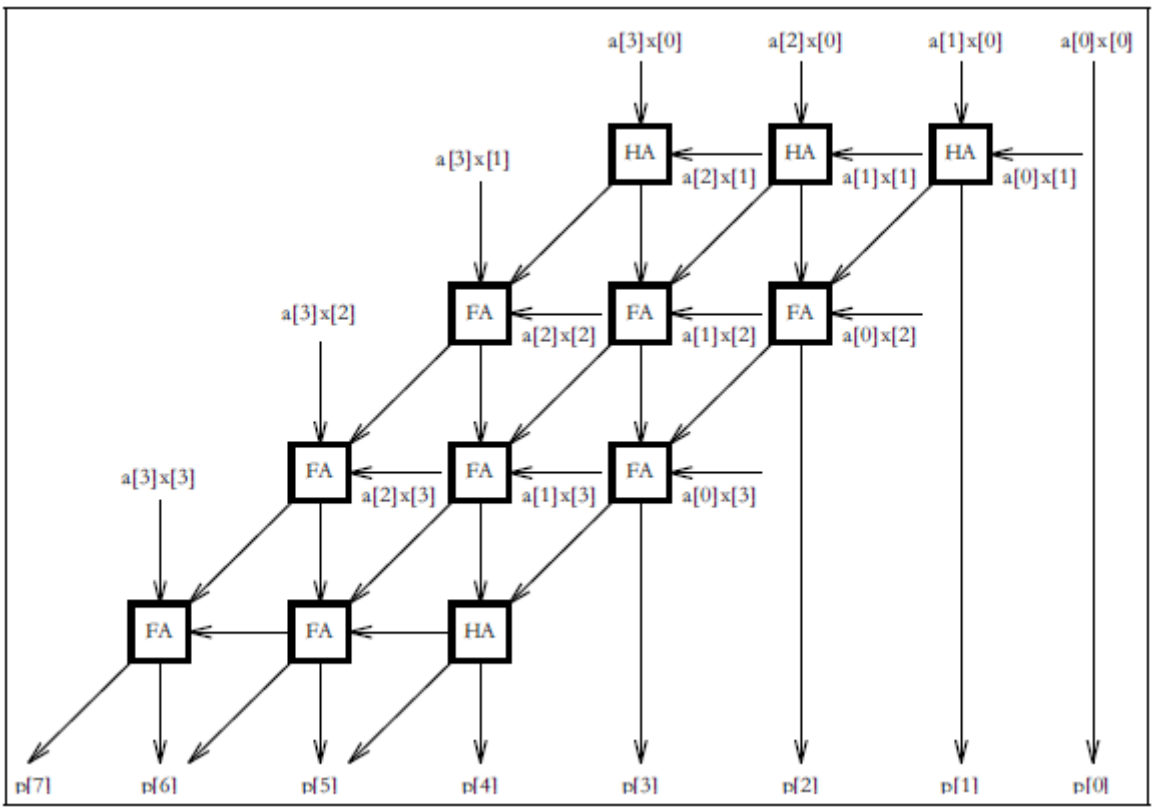
ضرب کننده‌ی آرایه‌ای

ضرب دو عدد باینری N بیتی، مثلا X و $A = \sum_{k=0}^{N-1} a_k 2^k$، می تواند به صورت زیر نوشته شود:

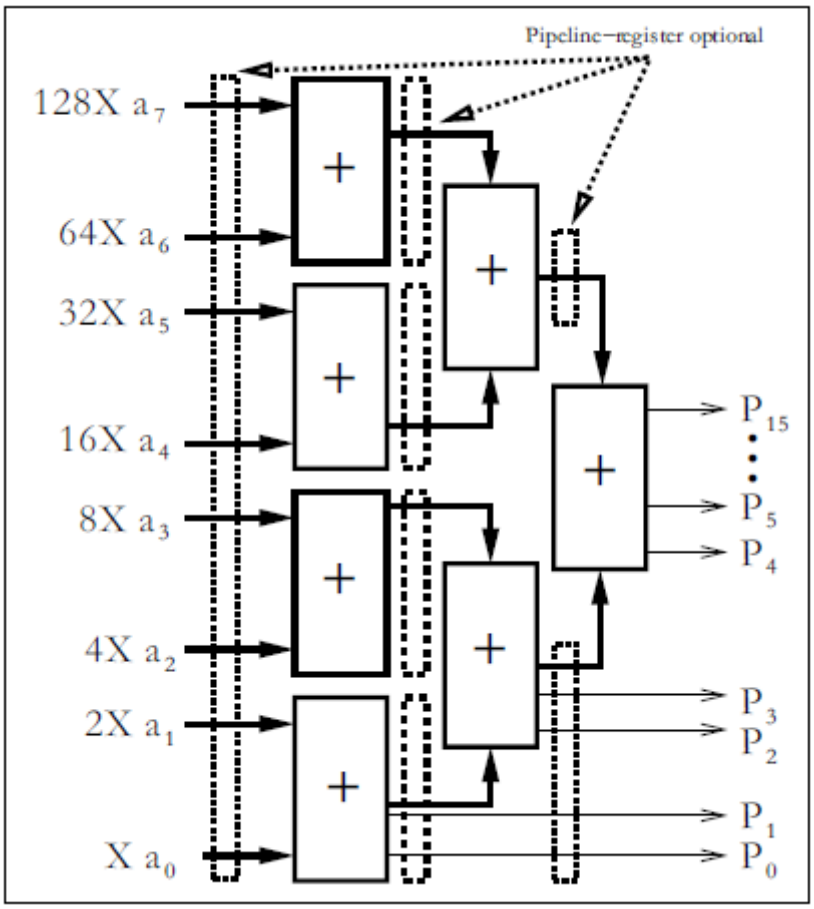
$$P = A \times X = \sum_{k=0}^{N-1} a_k 2^k X$$

همانطور که مشاهده می‌شود، ورودی X هر دفعه K بار شیفت داده شده و هر جا $a_k \neq 0$، $X2^k$ جمع شده و هر جا $a_k = 0$، عمل شیفت و جمع انجام نمی‌شود. به دلیل این که X موازی است و عملوند دوم یعنی A به صورت بیتی عمل می‌کند، به ضرب‌کننده‌ی بالا، ضرب‌کننده‌ی سری–موازی گفته می‌شود. ضرب‌کننده‌ای که در آن هر دو عملوند به صورت سری استفاده می‌شود، به نام ضرب کننده سری–سری شناخته می‌شود. ضرب‌کننده‌ی سری–سری تنها به یک تمام‌جمع کننده نیاز دارد، ولی تاخیر آن زیاد است $O(N^2)$. به همین دلیل، برای انجام عملیات ضرب به N^2 سیکل ماشین نیاز خواهیم داشت.

روش دیگری که می‌توان به کار برد، مصالحه‌ای بین سرعت و پیچیدگی است که به آن ضرب‌کننده‌ی آرایه‌ای (ضرب‌کننده‌ی موازی–موازی) گفته می‌شود. دیاگرام یک ضرب‌کننده‌ی آرایه‌ای 4×4 بیتی در شکل زیر قابل مشاهده است. همانطور که قابل مشاهده است، هر دو عملوند به صورت موازی به آرایه‌ای متشکل از N^2 سلول جمع‌کننده اعمال می‌شوند.



این آرایش، زمانی قابل اتکا است که زمان لازم برای محاسبه‌ی Carry و مجموع یکسان باشد. در FPGAهای مدرن، محاسبات Carry سریع‌تر از محاسبات مجموع انجام می‌شود. به همین دلیل، آرایش دیگری برای ضرب‌کننده‌ی آرایه‌ای ارائه می‌شود که به آن ضرب‌کننده‌ی آرایه‌ای سریع گفته می‌شود. دیاگرام یک نمونه ضرب‌کننده‌ی 8×8 بیتی در شکل زیر قابل مشاهده است. روشی که در این‌جا استفاده شده، ترکیب دو مقدار همسایه در Stage اول آرایه است. یعنی در Stage اول جدید، مقادیر $a_n X2^n$ و $a_{n+1} X2^{n+1}$ با یکدیگر جمع شده و عملیات تا رسیدن به نتیجه‌ی نهایی ادامه می‌یابد.



با توجه به شکل، تعداد Stageهای این ضرب کننده برابر $log_2(N)$ است. با مقایسه‌ی دیاگرام‌های دو ضرب‌کننده‌ی آرایه‌ای، درمی‌یابیم که روش جایگزین، علاوه بر تعداد Stage کمتر، امکان استفاده از Pipelining را هم آسان‌تر می‌سازد.

شرح تمرین

هدف از این تمرین، آشنایی با پیاده‌سازی ضرب‌کننده‌ی آرایه‌ای معمولی 8×8 بیتی به زبان *VHDL* است. توجه کنید که در ضرب‌کننده‌ی آرایه‌ای معمولی، *Stage* اول با استفاده از نیم‌جمع کننده پیاده‌سازی می‌شود. پیاده‌سازی شما باید به صورت ساختاری و ماژول‌وار باشد. به این صورت که ابتدا نیم جمع‌کننده و تمام جمع کننده تک بیتی را تعریف کرده و در ادامه با استفاده از آن‌ها ضرب‌کننده‌ی آرایه‌ای ۸*۸ بیتی را پیاده‌سازی کنید.

نکات تکمیلی

- گزارش ارسالی باید شامل موارد زیر باشد:
  - سطح مصرفی *FPGA* (تعداد *Slice*، *FF* و ...) در گزارش سنتز
  - تصویری از نتایج شبیه‌سازی مجزای هر ماژول (ترجیحا از شبیه‌ساز *Vivado* استفاده کنید)
- کلیه کدهای نوشته شده در آخر گزارش به صورت تک ستونی آورده شود.
- سعی کنید که کدهای *VHDL* تا حد ممکن دارای *Comment* بوده و به صورت مرتب نوشته شود. (در متن اصلی گزارش نیازی به توضیح کد نیست)
- کلیه‌ی کدهای *VHDL* به همراه گزارش با فرمت‌های *Word* و *PDF* به صورت یک فایل *zip*. آرشیو شده و در سامانه‌ی *Courses* آپلود گردد.
- از کپی کردن به شدت بپرهیزید. در صورتی که دو گزارش دقیقا مشابه یکدیگر باشند، به هر دو گزارش نمره صفر داده خواهد شد.

موفق باشید.