تمرین دوم مدارهای منطقی برنامه‌پذیر

محاسبه گر جذر (ریشه‌ی دوم)

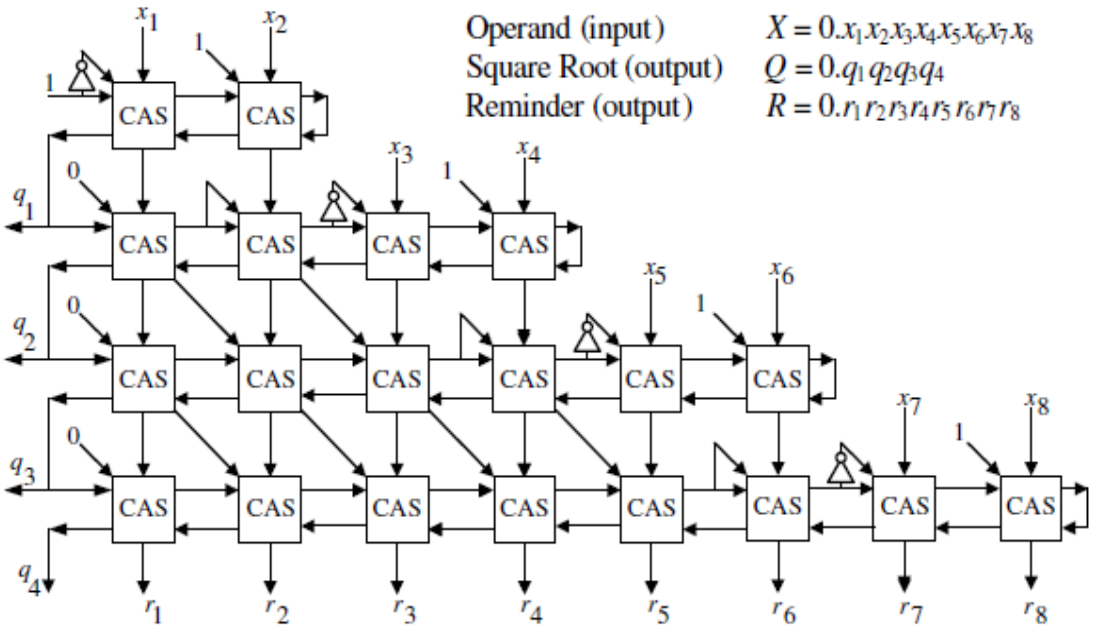
محاسبه‌ی ریشه‌ی دوم یک عدد، یکی از محاسبات مهم و حیاتی پردازش سیگنال‌های دیجیتال و سیستم‌های محاسباتی است. در گذشته، به خصوص در ریزپردازنده‌ها، محاسبات ریاضی پیچیده (تقسیم، جذر، لوگاریتم و ...) به صورت نرم‌افزاری پیاده‌سازی می‌شدند که تاخیر بسیار زیادی داشتند. اما امروزه به دلیل امکان استفاده از تعداد ترانزیستور بیشتر بر روی تراشه و تقاضا برای سرعت محاسباتی بیشتر، پیاده‌سازی سخت‌افزاری آن‌ها نیز مورد توجه قرار گرفته است. با توجه به این موضوع، طراحی سخت‌افزاری بهینه برای محاسبه‌ی جذر، به عنوان یکی از محاسباتی که سابقا به طور نرم‌افزاری انجام می‌شده، به یکی از موضوعات مورد بحث طراحی VLSI تبدیل گشته است. روش‌های بسیاری برای محسابه‌ی جذر یک عدد پیشنهاد شده که به دو دسته‌ی روش‌های تخمینی (Estimation) و روش‌های رقم به رقم (digit-by-digit) تقسیم می‌شوند.

یکی از طرح‌های پیشنهادی که در این تمرین قصد بررسی آن را داریم، پیاده‌سازی آرایه‌ای و ماژولار الگوریتم non-restoring ریشه‌ی دوم (جذر) است. در شکل (۱)، شبه کد الگوریتم non-resotring جذر قابل مشاهده است.

```
Q0 := 0; R0 := X;
R1 := R0 - 2^2(n-1);
for i in 1 to n loop
  if Ri ≥ 0 then Qi := 2·Qi-1 + 1; Ri+1 = Ri-(1 + 4·Qi)·2^2(n-i-1);
  else Qi := 2·Qi-1; Ri+1 = Ri + (3 + 4·Qi)·2^2(n-i-1);
  end if;
end loop;
Q := Qn;
```

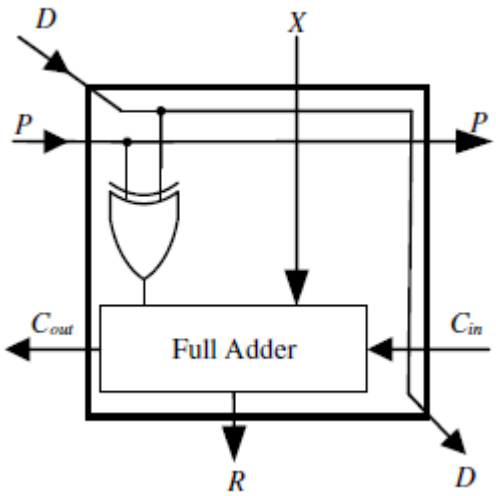
شکل (۱) الگوریتم non-restoring محاسبه‌ی جذر

مداری که به طور کلاسیک برای پیاده‌سازی این الگوریتم به صورت آرایه‌ای ارائه می‌شود، در شکل (۲) قابل مشاهده است. بلوک‌های سازنده‌ی این ماژول نیز که به عنوان جمع–تفریق کنترل‌شده (Controlled Add/Substract) شناخته می‌شوند، در شکل (۳) قابل مشاهده است.



شکل (۲) مدار کلاسیک محاسبه‌ی non-restoring ریشه‌ی دوم برای یک عدد ۸ بیتی با استفاده از بلوک‌های CAS

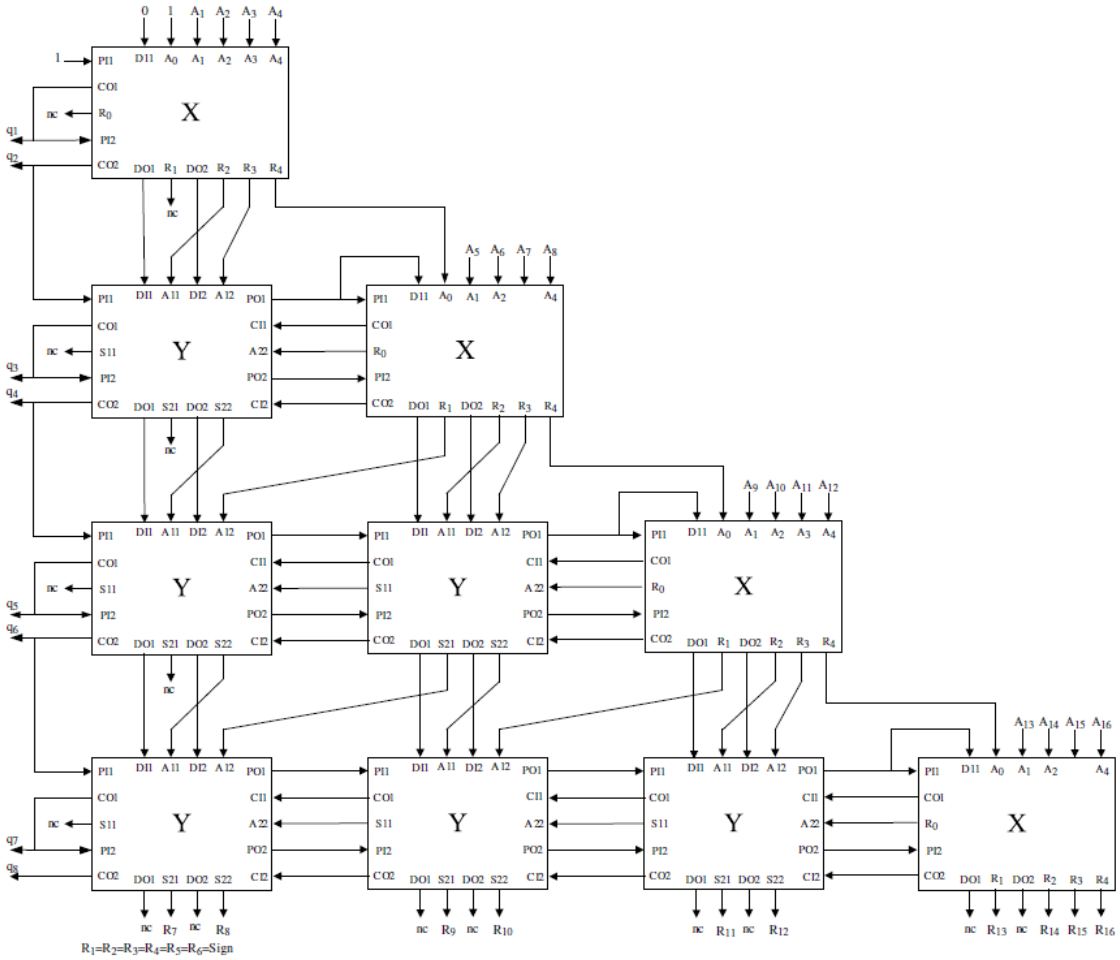
(در این شکل x1، q1 و r1 به ترتیب با ارزش‌ترین بیت‌های X، Q و R هستند)



شکل (۳) ساختار داخلی یک بلوک CAS

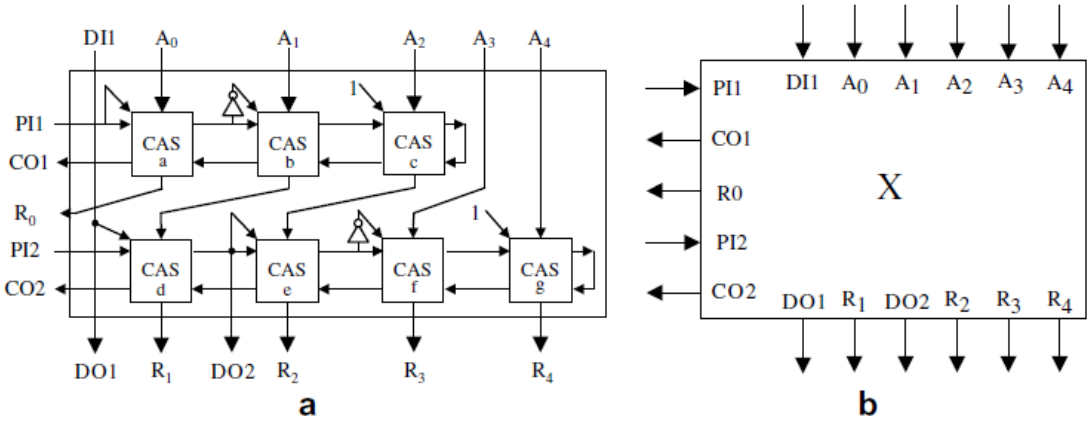
در شکل (۴) مدار ریشه‌ی دوم به صورت ماژولار پیاده‌سازی شده است. به این معنی که با بزرگ‌تر شدن ورودی کافی است تعداد ماژول‌های مورد استفاده را بیشتر کنیم و نیازی به طراحی دوباره مدار نیست. البته تنها در صورتی که تغییر عرض بیت ضریبی از عدد ۴ باشد، می‌توان از این آرایش استفاده کرد. علاوه بر این، این مدار مشابه مدار کلاسیک و مدار ضرب‌کننده‌ی آرایه‌ای (تمرین ۱)، به صورت آرایه‌ای و ترکیبی

پیاده‌سازی شده است. همانطور که قبلا نیز اشاره شد، هدف از این نوع پیاده‌سازی افزایش سرعت محاسبات به قیمت استفاده بیشتر از منابع سخت‌افزاری است.

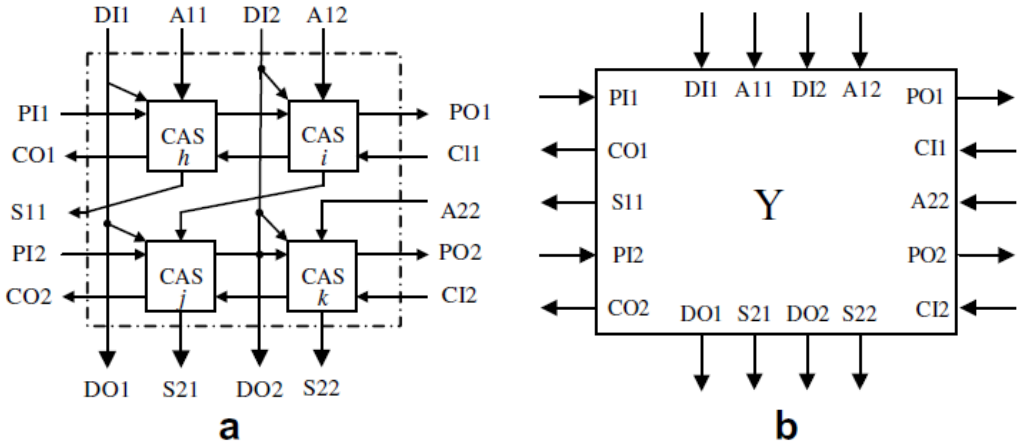


شکل (۴) مدار جذر ۱۶ بیتی با استفاده از ماژول‌های *X* و *Y* (*nc* به معنی عدم اتصال یا *not connected* است) – با ارزش ترین بیت ورودی، بیت A_1 و کم ارزش‌ترین بیت ورودی بیت A_{16} است. با ارزش‌ترین بیت خروجی q_1 و کم‌ارزش‌ترین بیت q_8 است.

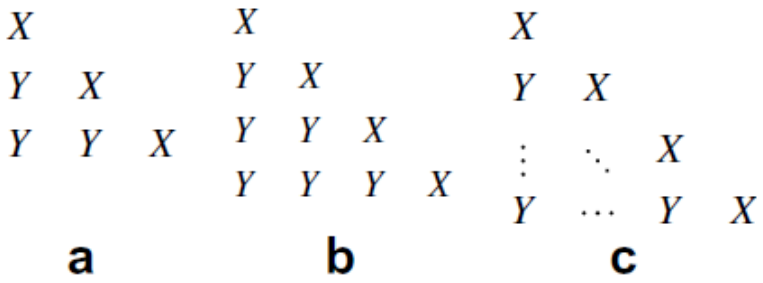
ساختار داخلی ماژول‌های *X* و *Y* در شکل (۵ و ۶) قابل مشاهده است. علاوه بر این، آرایش ماژول‌ها نیز برای چند عرض بیت مختلف در شکل (۷) ارائه شده است.



شکل (۵) ماژول *X* – *a*: ساختار داخلی ماژول *b*: بلوک دیاگرام



شکل (۶) ماژول *Y* – *a*: ساختار داخلی ماژول *b*: بلوک دیاگرام



شکل (۷) آرایش مدار محاسبه‌ی جذر برای *a*: ۱۲ بیت – *b*: ۱۶ بیت – *c*: *4n* بیت

شرح تمرین

هدف از این تمرین، پیاده‌سازی یک مدار محاسبه‌گر جذر ماژولار و آرایه‌ای برای یک ورودی ۱۲ بیتی به زبان *VHDL* است (مشابه مدار شکل ۴ اما با تعداد بیت ورودی ۱۲). پیاده‌سازی این محاسبه‌گر باید به صورت سلسله مراتبی و با استفاده از بلوک‌های شکل (۳، ۵ و ۶) انجام شود. به این صورت که ابتدا در یک فایل *VHDL* جداگانه مدار *CAS* (شکل ۳) طراحی شود. در ادامه با استفاده از مدار *CAS*، دو ماژول *X* و *Y*

(شکل (۵ و ۶)) در دوفایل *VHDL* جداگانه طراحی شوند. در انتهای کار، با استفاده از آرایش معرفی شده برای ورودی‌های ۱۲ بیتی در شکل (۷) و ماژول‌های *X* و *Y* ای که طراحی کردید، پیاده‌سازی محاسبه‌گر جذر ۱۲ بیتی را انجام دهید.

- ✓ در صورتی که علاقه داشتید، می‌توانید مدار بالا را با استفاده از تعریف *Generic*، برای ورودی‌های *4n* بیتی طراحی کنید.

نکات تکمیلی

- گزارش ارسالی باید شامل موارد زیر باشد:
  - سطح مصرفی *FPGA* (منابع مصرفی: تعداد *Slice*، *FF* و ...) هر ماژول
  - 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.
- در صورت امکان، سعی کنید که از نرم‌افزار *Vivado* برای انجام این تمرین استفاده کنید.
- نیازی به ارسال پروژه‌ی کامل نیست و فقط فایل‌های *VHDL* را آپلود نمایید.
- 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.