به نام خدا

طراحی کامپیوتری سیستم‌های دیجیتال - پاییز 1404

تمرین کامپیوتری پنجم

پیاده‌سازی الگوریتم زمان‌بندی و تولید کد وریلاگ

طراحان: علی دارابی و محمدمهدی صمدی

هدف پروژه

در این پروژه ابتدا دو الگوریتم زمان‌بندی Minimum-Latency, Resource-Constrained و Minimum-Resource, Latency-Constrained را پیاده‌سازی می‌کنید. سپس کد Controller و Datapath متناظر را در زبان Verilog تولید می‌کنید و با نوشتن برنامه تستی، خروجی مدار طراحی‌شده برای یک ورودی را محاسبه می‌کنید.

نحوه‌ی اجرای برنامه

به عنوان ورودی به کد `main.py` آدرس فولدری که شامل فایل input.json است را آرگومان دهید. برای مثال:

```
py main.py ./samples/sample1/
```

فایل ورودی شامل عبارت ریاضی، عنوان الگوریتم و محدودیت‌های زمانی و منابع است. برای دیدن ساختار آن به نمونه‌های موجود مراجعه کنید. در انتها دو تصویر از DFG اولیه و بعد از زمان‌بندی، فایل json اطلاعات خروجی زمان‌بند و یک فولدر شامل دو کد verilog در فولدر داده شده خواهید داشت.

توجه کنید که جهت سنجش برنامه هنگام تحویل، فرمت ورودی و خروجی json نباید تغییر کنند. همچنین در فایل `main.py` باید تابع `generate_verilog` را کامل کنید اما برای سنجش خروجی، بقیه‌ی آن را عوض نکنید.

شرح تمرین

تمرین شامل بخش‌های زیر است:

1. تبدیل عبارت ریاضی به Data Flow Graph
2. پیاده‌سازی الگوریتم‌های Scheduling

3. تولید Verilog Code

تبدیل عبارت ریاضی به Data Flow Graph

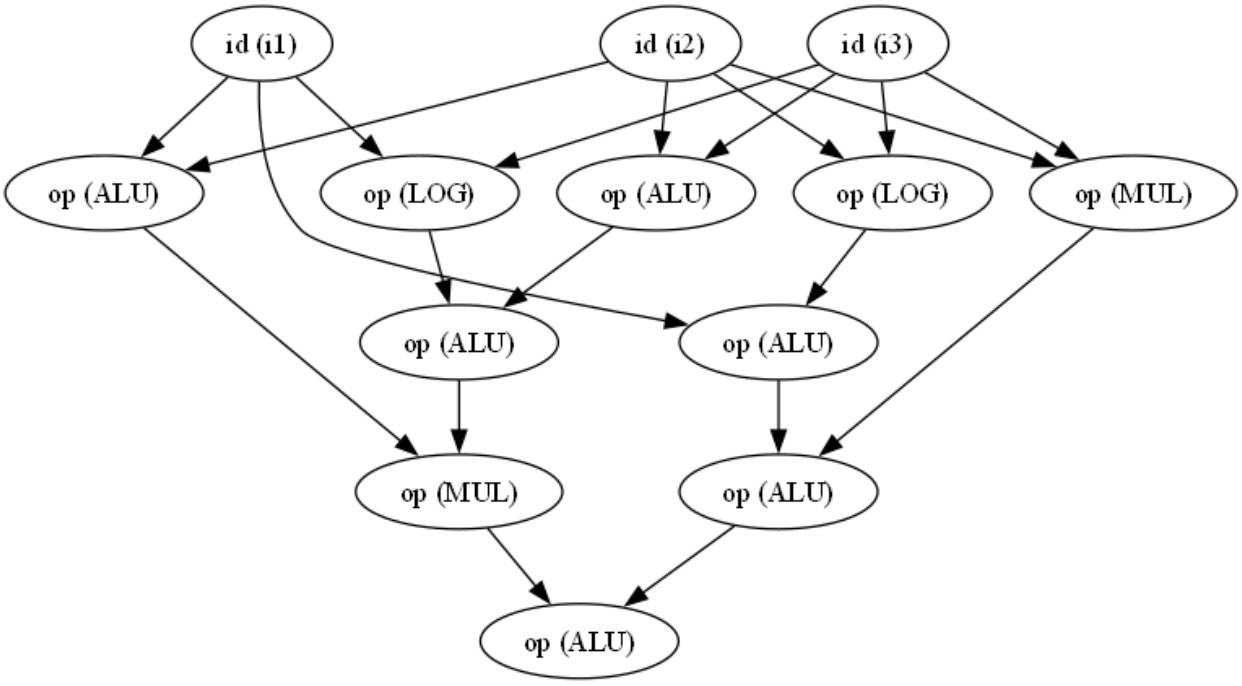
وقتی برنامه شروع شود، این عبارت به operator و operand هایش parse شده و گراف متناظر آن ساخته می‌شود. در این تمرین سه نوع operator زیر تعریف شده‌اند:

1. عملیات ALU: جمع (+) یا تفریق (-)
2. عملیات MUL: ضرب (*)، تقسیم (/)
3. عملیات LOG (منطقی): and (&) یا or (|)

این بخش پیاده‌سازی شده است. کدهای استفاده شده برای این بخش به شرح زیر هستند:

- کد `graph_visualizer` مسئولیت پارس کردن و مصورسازی گراف را بر عهده دارد.
- کد `dfg_builder` گراف را با دو نوع node تعریف شده و پوینتر های بین‌شان برای استفاده در `scheduler` می‌سازد.

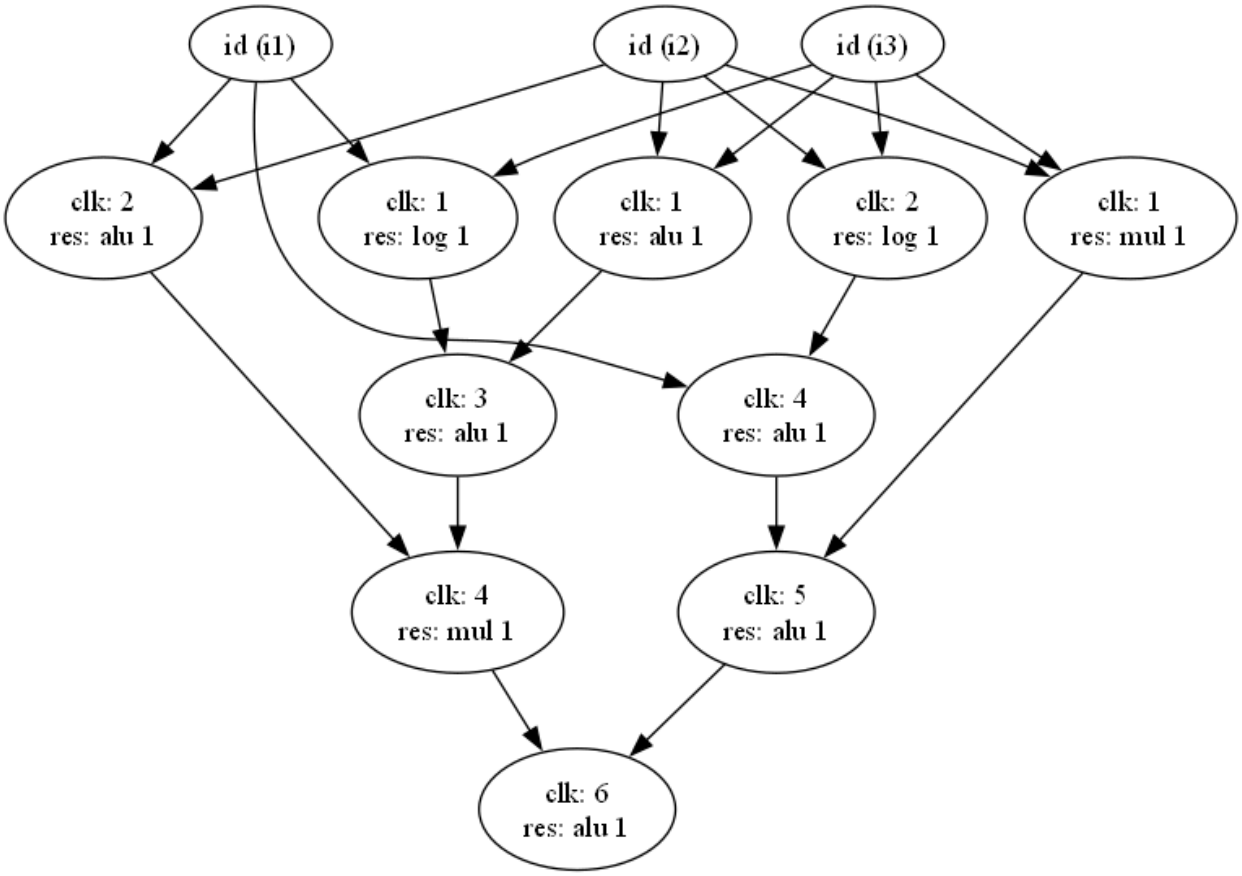
در زیر مثالی را برای یک عبارت ریاضی نمونه و گراف معادل آن می‌بینید:



پیاده‌سازی الگوریتم‌های Scheduling

در این بخش Template در اختیارتان قرار گرفته. شما باید فایل `scheduler` را با نوشتن بخش‌های to do کامل کنید تا دو الگوریتم خواسته‌شده به درستی کار کنند. در صورت نیاز کلاس‌های آن را تغییر دهید. وظیفه هر تابع داخل فایل کد توضیح داده شده. جهت سهولت تست اتومات در زمان تحویل، اسم توابع را عوض نکنید و فقط هر کدام را کامل کنید. در صورت هرگونه ابهام، با دستیار آموزشی این بخش در ارتباط باشید.

بعد از schedule کردن، با فراخوانی `get_scheduling_info` اطلاعات ذخیره شده به منظور استفاده در مرحله بعدی گرفته می‌شوند. یک method برای این بخش در فایل `graph_visualizer` نوشته شده که DFG زمان‌بندی شده را نمایش می‌دهد. برای درک بهتر خروجی برنامه می‌توانید از آن استفاده کنید. مثلا برای همان گراف بالا با محدودیت‌های داده شده بدین صورت خواهد بود:



در هر راس operator گراف بالا شماره clock cycle و شماره resource ای که برای آن زمان‌بندی شده قرار دارند.

تولید Verilog Code

در این بخش لازم است برای خروجی Scheduling که در بخش قبلی تولید شده، Datapath و Controller طراحی و تولید کنید. توجه داشته باشید که در این مرحله **Template آماده در اختیار شما قرار داده نمی‌شود** و باید خودتان یک کلاس جدید با نام `verilog_generator` تعریف کرده و کدهای مربوط به این بخش را در آن بنویسید.
(پیشنهاد می‌شود برای افزایش خوانایی، کدهای مربوط به تولید Datapath و Controller را در بخش‌های جداگانه پیاده‌سازی کنید.)

نکات مهم در پیاده‌سازی:

1. معیار اصلی بررسی، **صحیح بودن کد Verilog و نتایج آن در Simulation** است، نه جزئیات کد تولید شده. به این معنا که لزومی ندارد همه دانشجویان یک کد یکسان تولید کنند.
2. جزئیات کد Verilog (مانند نام یا تعداد متغیرهای ورودی/خروجی یا سیگنال‌های محلی) بررسی نمی‌شود؛ تنها صحت عملکرد کد مورد توجه است.
3. Datapath باید دارای یک سیگنال `done` باشد تا اتمام عملیات مشخص شود، و یک خروجی `result` که نتیجه‌ی نهایی روی آن قرار گیرد.
4. فرض کنید تمامی متغیرهای موجود در Expression ورودی، ۳۲ بیتی هستند.
5. Controller باید به تعداد سیکل‌های مورد نیاز برای انجام عملیات، State داشته باشد.
6. برای تست خروجی تولید شده، می‌توانید یک Testbench بنویسید و در ModelSim اجرا کنید. (این کار **اجباری نیست**، اما ما برای بررسی صحت خروجی شما از همین روش استفاده خواهیم کرد.)
7. بعد از تعریف Datapath و Controller باید یک Top Module تعریف کنید که ورودی‌های آن clk و rst و start و متغیرهای ورودی هستند، و خروجی‌های آن result و done هستند. (برای تست پیاده‌سازی شما از Top Module استفاده می‌شود.)

در صورت وجود هرگونه ابهام، با دستیار آموزشی این بخش در ارتباط باشید.

موارد تحویل

- الگوریتم‌های scheduling به درستی کار کنند.
- کدهای verilog تولید شده در برنامه ModelSim، به درستی کامپایل شوند.
- در صورت تست کدهای تولید شده، خروجی کدهای تولید شده شما باید مطابق با خروجی باشد که پس از اتمام مراحل scheduler به آن می‌رسیم.

سایر نکات

- انجام این تمرین به صورت گروه های دو نفره خواهد بود.
- فایل ها و گزارش خود را تا قبل از موعد تحویل، با نام CAD_HW5_<SID>.zip در محل مربوطه در صفحه درس آپلود کنید.
- در صورت هرگونه ابهام می توانید از طریق ایمیل یا تلگرام با طراحان در ارتباط باشید.
- هدف این تمرین یادگیری شماست! در صورت کشف تقلب، مطابق با قوانین درس برخورد خواهد شد.

موفق باشید.