دانشگاه صنعتی شریف
دانشکده مهندسی کامپیوتر

عنوان:

گزارش تمرین عملی هفتم

اعضای گروه:

پارسا ملکیان_۴۰۲۱۷۱۰۷۵
پارسا حاجی قاسمی_۴۰۲۱۰۵۸۷۹

نام درس

معماری کامپیوتر

نیم‌سال دوم ۱۴۰۴-۱۴۰۳

نام استاد درس

دکتر اسدی

۱ گزارش پیاده‌سازی pipeline در پردازنده mips

۱-۱ نحوه پیاده‌سازی

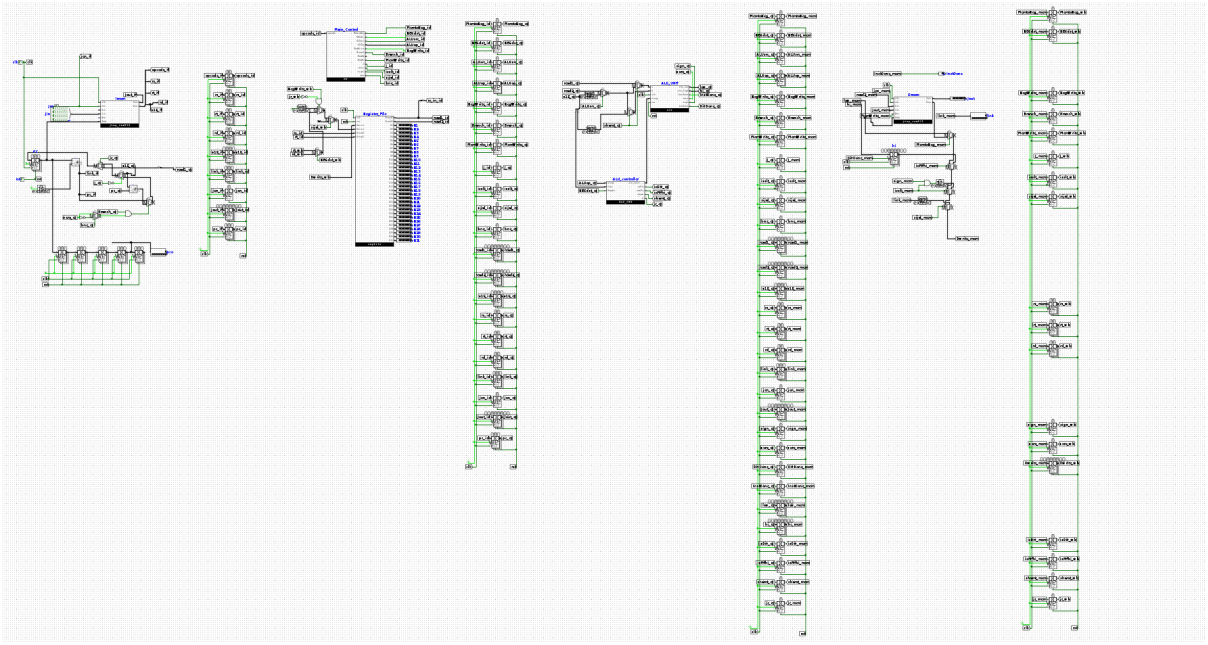
ابتدا معماری پردازنده را به ۵ بخش تقسیم میکنیم به طوری که تمام دستورات باید این ۵ مرحله را طی کنند.
مراحل عبارتند از:

- IF
  مراحل خواندن دستور از حافظه و تشخیص مقادیر ،rs ،...rt
- ID
  مرحله decode دستور که با گرفتن opcode دستور، سیگنال های کنترلی مناسب را تولید میکند و همچنین به صورت موازی، مقادیر رجیستر های مورد نیاز یعنی ،rt rs و... از حافظه خوانده میشوند
- Ex
  مرحله عملیات محاسباتی که محاسبات ALU در این بخش انجام میشود و func بیت نیز در این بخش تحلیل میشود و باقی سیگنال‌های کنترلی ساخته میشوند
- Mem مرحله نوشتن یا خواندن در حافظه در این بخش انجام میشود همچنین تصمیم گیری برای دیتایی که باید درصورت نیاز داخل رجیستر ها ریخته شود نیز در این مرحله است
- WB مرحله نهایی که درصورت نیاز، دیتای از پیش تعیین شده را داخل رجیستر مقصد که آن نیز از مرحله IF تعیین شده، میریزد.

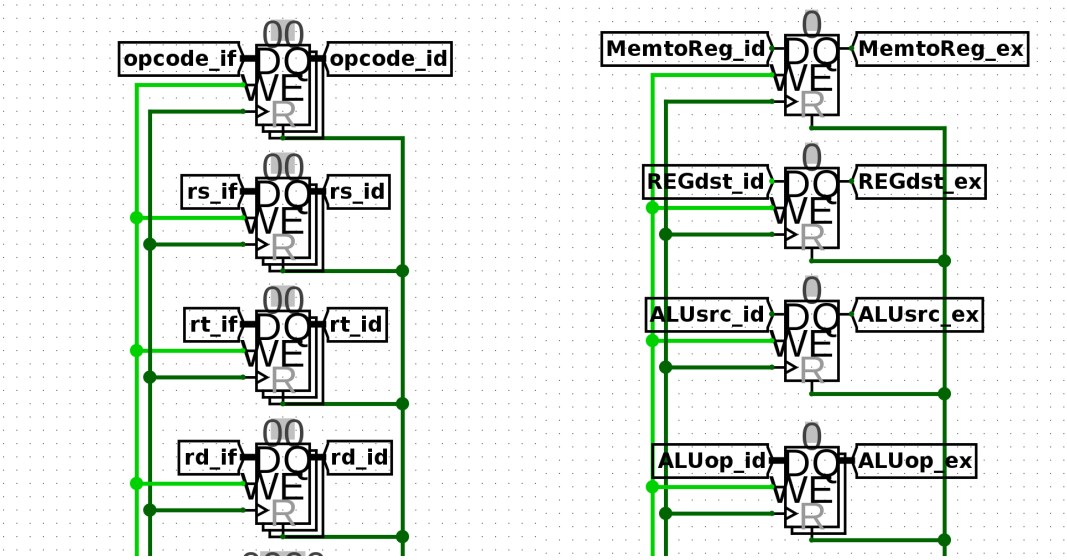
۱-۲ تغییرات لازم برای مدار

روشی که ما برای پیاده سازی این معماری استفاده کردیم به این صورت است که ابتدا دیتاپث را درست مانند منطق معماری pipline تقسیم بندی کردیم:

تقسیم بندی به این صورت است که رجیسترهای زیر هم قرار داده شده درواقع مرزبین ناحیه‌هارا مشخص میکنند و ناحیه‌ها به ترتیب مانند بخش‌های مورد نیاز این معماری هستند یعنی به ترتیب if بعد id سپس ex و mem و درنهایت wb که چون از یک رجیستر فایل استفاده میکنیم مرحله wb



شکل ۱: تقسیم بندی



شکل ۲: محتوای رجیسترها

مجددا وارد ناحیه دوم میشود.
همانند شکل ۱ که در بالا میبینید.

حالا محتوایی که این رجیسترها نگه میدارند به چه صورت است؟ درواقع رجیستر ها تمام سیگنال ها و داده های مورد نیاز را درخود ذخیره میکنند و با اینکار گویی اطلاعات را به قسمت بعدی میبرند. با اینکار عملا دستور و تمام متعلاقتش وارد استیج بعدی میشود بدون اینکه به دستورات دیگر که همزمان درحال اجرا در استیت های دیگر هستند آسیبی وارد کند و یا تداخلی داشته باشد.
محتوای رجیستر ها و نحوه انتقال در شکل ۲ قابل مشاهده است

۱-۳ نحوه استفاده از سیگنال‌ها

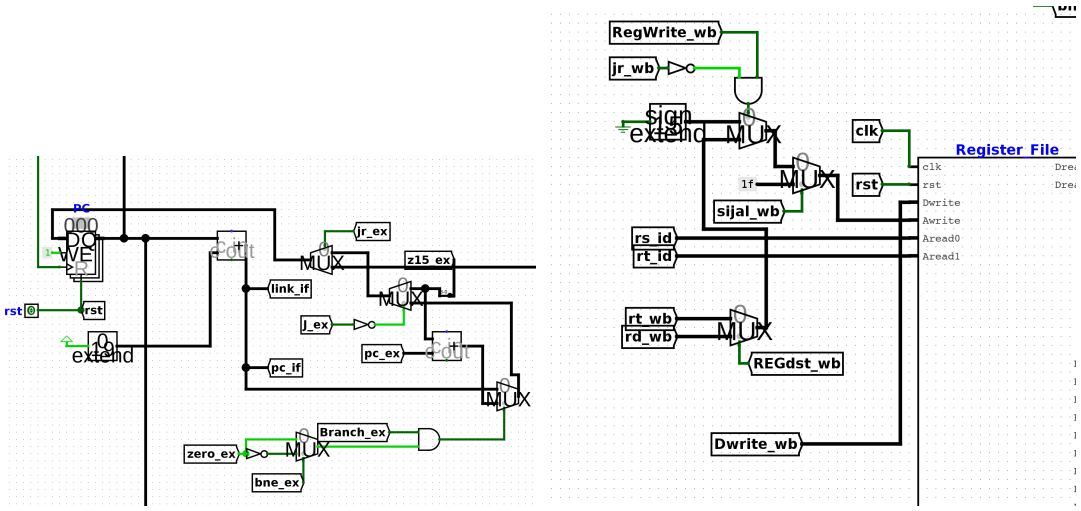
همانطور که پیش تر گفتیم،سیگنال ها همراه با دستورات حرکت میکنند یعنی برای مثال اگر در مرحله wb بخواهیم اطلاعات را داخل رجیستر فایل بنویسیم، اطلاعات را باید از dwrite_wb بگیریم و داخل رجیستر rd_wb بریزیم و نباید به طور مثال از rd_if استفاده کنیم چرا که اطلاعات داخل آن مربوط به دستور دیگری است که هنگامی که این دستور داخل استیج wb قرار دارد آن دستور در استیج if درحال اجراست

در شکل ۳ نحوه استفاده از سیگنال درست برای دستورات wb و حالات مختلف pc را مشاهده میکنید

در تصویر سمت چپ میبینیم که سیگنال +pc۴ همراه با دستور حرکت میکند یعنی به استیج if میرود و از آنجا ادامه پیدا میکند و از طرفی برای تضخیص دستورات جامپ و برنچ در استیج execute متوجه آن خواهیم شد در نتیجه از سیگنال های تولیدی در آن استیج استفاده میکنیم.
در شکل سمت چپ تفاوت wb و id در برخورد با رجیستر فایل را مشاهده میکنیم.

۱-۴ تغییر جزئی جاج

درنهایت برای هماهنگی pc مدار با ipc جاج تغییر جزئی ای در بخش while جاج انجام دادیم که در شکل ۴ قابل مشاهده است.



شکل ۳: استفاده از سیگنال‌های استیج مناسب

```
273         if (!fail_flag) begin
274             $display("ipc : ", ipc);
275             //exec_internal();
276             //#2;
277             while (InstDone !== 1 || PC !== ipc)begin
278                     #2;
279             end
280             exec_internal();
281             #2;  // waiting until your circuit is ready and in sync
282
283             for (j = 1; j < 32; j++) if (R[j] !== ireg[j]) fail_flag = 1;
```

شکل ۴: تغییر جزئی جاج

```
ipc :  22
ipc :  23
ipc :  24
ipc :  39
ipc :  40
ipc :  41
ipc :  42
load 000001fe
ipc :  43
mem : [1f9] :          0[1fa] :          0[1fb] :          0[1fc] :          0[1fd] :
ACCEPTED
       159 /       159
parsaivi@parsaivi-zenbook:~/Documents/arch/SUT_CA_4032_ProfAsadi_Judgement_System$
```

شکل ۵: نمره نهایی

۵-۱ داوری نهایی

درنهایت با اجرای جاج بر روی کد نمره کامل جاج توسط کد دریافت میشود: