به نام خدا

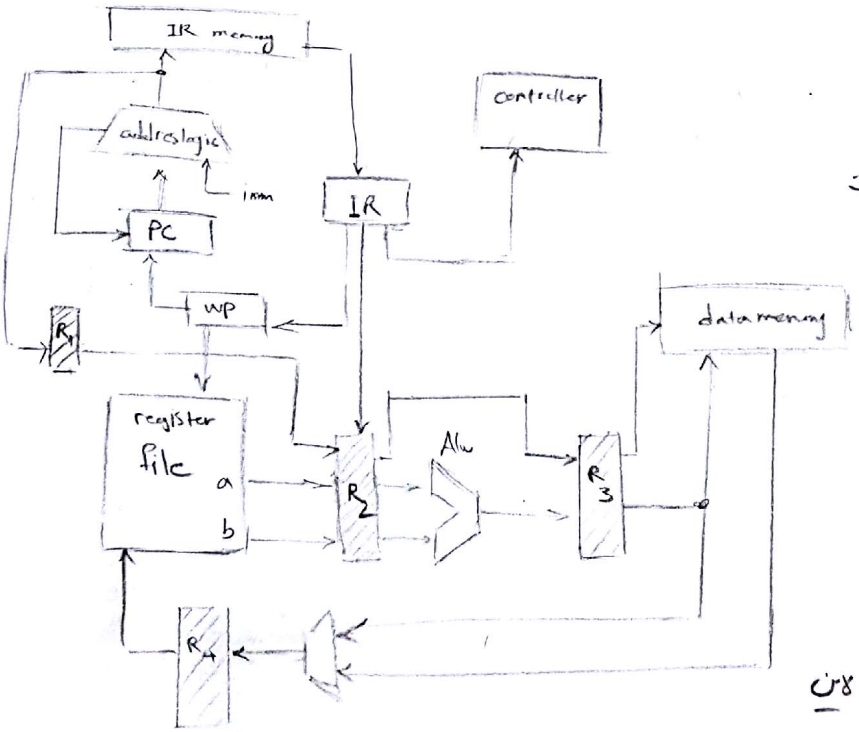
زهرا دهقانیان

9431039

طراحی پروژه کامپایلر:

* بخش اول (مشخص کردن State ها) استیت های این پایپ لاین همانند mips شامل 5 مرحله، fetch، decode، execute، memory، writeback است. در مرحله fetch، دستورات از memory به IR می‌رود. در decode رجیستر فایل، سیگنال های کنترلی Alu مشخص می‌شود. در مرحله execute سیگنال کنترلی وارد رجیسترهای مختلف شده و عملیات Alu انجام می‌شود. در مرحله memory، در صورت نیاز مقادیر مختلف از حافظه خوانده و یا در حافظه نوشته می‌شود. در مرحله آخر یعنی write back، مقادیر محاسبه شده و نیازمند نوشتن شدن در رجیسترها نوشته می‌شود.

* بخش دوم (طراحی datapath) برای طراحی باید بین تمام استیت ها، رجیستری برای ذخیره کردن داده و دستورات قرار داده شود. هم‌چنین برای این که بتوان مرحله fetch و memory را به طور همزمان انجام داد، حافظه دستورات را از حافظه داده ها جدا کرد.



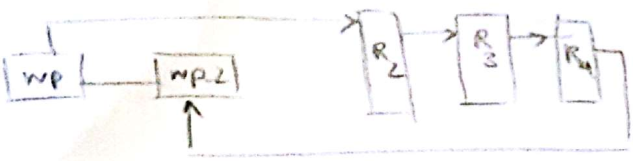
شکل 1-2 : شمای کلی پایپ لاین

به دلیل شمای کلی از کامپیوتر ساده هرکدام از رجیسترهای R_1 تا R_4 شامل چند رجیستر برای ذخیره بیت‌های مختلف لازم دستورات است.

* بخش سوم: انواع هازارد:

1.3: Structural Hazard: مشکلات مربوط به یکی بودن حافظه ها در این جام مانند Mips وجود داشت پس در ابتدای طرح اولیه با بهره گیری از معماری هاروارد برطرف شد

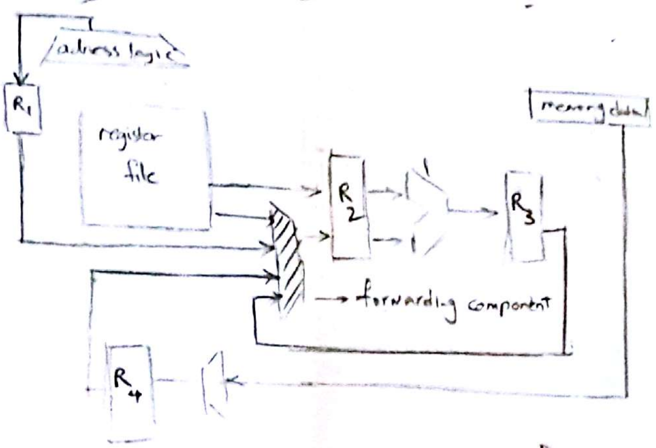
هازارد بعدی مربوط به زمانی است که بخواهیم به طور همزمان در writeback در رجیستری بنویسیم و در decode از رجیستری بخوانیم. برای حل این هازارد یک رجیستر wp برای نوشتن اضافه می کنیم. به این صورت که wp و مقادیر مشخص کننده R_s, R_D به شکل کد هستند L1 در صورت دریافت دستور awp ، wp-2 به همراه دستور به رجیسترهای میانی فرستاده می شود تا اگر زمانی به دستور writback رسید، مشکلی بوجود نیاید.



* wp هازارد داده ای نخواهد داشت. در واقع چون سیگنال های کنترلی تماما در استیت decode ، set می شود پس مقدار wp+I در پایان decode حاضر و در wp نوشته می شود و با هازاردی با مفهوم ست نشدن wp (در نتیجه اشتباه خواندن از رجیستر فایل) برخورد نخواهیم شد)

ـــ ادامه بخش: انواع هازارد:

2.3. Data Hazard: در این بخش تنها یک هازارد داریم هازارد read after write است. این هازارد زمانی رخ می‌دهد که یک رجیستر هنوز مقداردهی نشده ولی در دستور بعدی بلافاصله می‌خواهیم بخوانیم. برای حل این مشکل از روش formading استفاده می‌کنیم که یعنی از Alu یک خروجی برای رجیستر مایل نیز در نظر می‌گیریم



پس این بخش به datapath اضافه می‌شود:

این هازارد در هنگام برخورد با دستوراتی رخ می‌دهد که باید خروجی اش در R_D ذخیره شود (یعنی دستوراتی که مربوط به Alu هستند (ورودی های mux برابر است با خروجی های Alu (R_3)، خروجی حافظه (R_4)، خروجی از رجیستر مایل و خروجی addres logic برای دستور (PC+I)

```
f   d   e   m   w
    f   d   e   m   w
        f   d   e   m   w
```

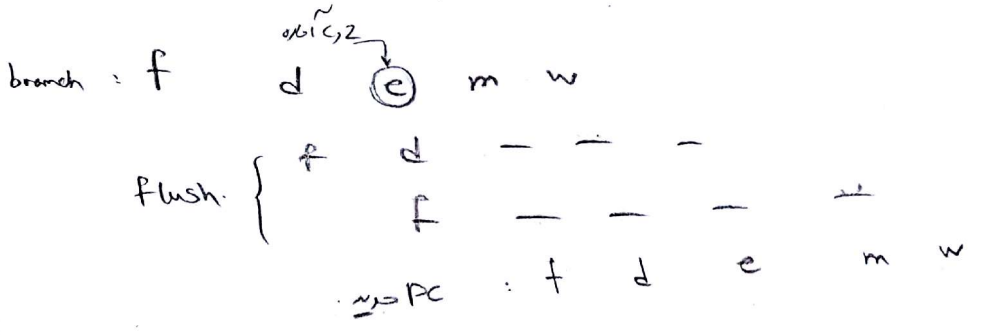
3.3. Control Hazard: این هازارد زمانی اتفاق می‌افتد که با دستورات jump و branch روبرو می‌شویم

در دستورات jump چون مقدار PC بعد از exe مشخص می‌شود، تا به این استیت برسیم تعدادی دستور اشتباه fetch کرده‌ایم (در دستور Jump Address در صورت نیاز با forwarding مانند بالا مقدار R_D را می‌توان زودتر دریافت کرد) به همین منظور خط پایپ لاین را flush می‌کنیم و با مقدار PC جدید که در مرحله exe مشخص می‌شود، ادامه می‌دهیم.

```
jump :     f   d   e   m   w
flushing { f   d   _   _   _
           {   f   _   _   _   _
PC جدید →          f   d   e   m   w
```

← ادامه بخش 3.3 .

برای بررسی branch ، باید توجه داشت که مقدار Z و C چون در مرحله exe قبلی محاسبه شده است و ما در مرحله exe به آن احتیاج داریم پس منطقی با محاسبه شرط branch نداریم . حال دو حالت داریم ، حالت اول این که در مرحله exe مشخص شود که نیاز به پرش نیست . که در این صورت پایپ لاین بدون مشکل پیش می‌رود . حالت دوم : نیاز به پرش باشد که همانند Jump باید با flushing پایپ لاین را تخلیه کنیم . که به این منظور باید reset رجیسترهای مختلف توسط controller فعال شود .



بخش 4 : کنترلر پایپ لاین :

در کنترلی که طراحی کردیم در مدار کنترلی فقط درنظر گرفتیم. مدار flushing ، مدار forwarding . برای کنترل مدار flushing (درصورت تشخیص نیاز به branch / Jump در مرحله exe) باید در دستور بعدی که بعد از این دستور وارد شده باید reset رجیترهای IR, R_2, R_1 را فعال کنیم که دستوراتی که در مرحله fetch و decode هستند را از پایپ لاین حذف کنیم و دستورات جدید با آدرس صحیح را fetch کنیم.

برای کنترل مدار forwarding نیز باید درصورتی که دستوری که در مرحله exe در حال خواندن از روی رجیتری باشد، که در مرحله mem در حال نوشتن روی همان رجیتر باشیم. و یا اگر دستور در مرحله mem با دستور موجود در مرحله wb یکسان باشد باید مدار forwarding را فعال کنیم. هنگام فعال شدن select مالکس مشخص شده در شکل را با توجه به عامل فعال کننده مالکس انتخاب می کنیم.

جزئی 5: datapath کلی