دانشکده‌ی مهندسی کامپیوتر

طراحی سیستم‌های دیجیتال

استاد: دکتر امین فصحتی

طراحان: امیرحسین محمدپور و فرزام کوهی

بهار ۱۴۰۴

تمرین سوم

فرمت فایل ارسالی برای تمرین باید به صورت زیر باشد، در غیراینصورت تصحیح **نخواهد شد.**

```
HW_DSD_CPU_studentnumber/
|--- doc.pdf            # Documentation file
|--- src/               # codes and modules (in verilog)
|--- syn/               # output of synthese
```

مقدمه

در این تمرین، هدف طراحی و تحلیل یک پردازنده ساده ۱۶ بیتی چند چرخه‌ای[1] است که مجموعه‌ای محدود اما کاربردی از دستورات محاسباتی و حافظه‌ای را پشتیبانی می‌کند. این پردازنده با هدف درک بهتر مفاهیم پایه‌ی معماری مجموعه دستورات (ISA[2])، ساختار واحد محاسباتی، رجیستر فایل و نحوه آدرس‌دهی حافظه طراحی شده است. فرمت دستورات در دو قالب R-Type[3] و M-Type[4] تعریف شده و برای پیاده‌سازی عملیات‌های پایه، از الگوریتم‌های استاندارد و قابل پیاده‌سازی در سطح سخت‌افزار استفاده شده است که به هر کدام به طور مجزا اشاره خواهد شد.

شما باید بر اساس مشخصات داده‌شده برای اجزای مختلف پردازنده، واحد کنترل را نیز طراحی کنید، به‌گونه‌ای که سیگنال‌های کنترلی مناسب برای اجرای ترتیبی هر دستور را تولید کند. این طراحی باید هماهنگ با فرمت دستورات، ساختار رجیستر فایل، واحد محاسباتی و حافظه انجام گیرد.

لطفا توجه شود که کد شما باید بتواند سنتز شود.

---

[1] multi-cycle

[2] Instruction set architecture

[3] Register-type

[4] Memory-Type

ساختار دستورات و آپکدها

دستورات این پردازنده در دو فرمت اصلی تعریف می‌شوند: R-Type برای عملیات محاسباتی و M-Type برای عملیات حافظه.

فرمت R-Type

Opcode	rd	rs1	rs2	Unused
3bit	2bit	2bit	2bit	7bit

فرمت M-Type

Opcode	rd/rs	base	Address
3 bit	2bit	2bit	9bit

جدول آپکدها

دستور	نوع	Opcode	توضیح
ADD	R-Type	000	جمع ۶۴ بیتی
SUB	R-Type	001	تفریق ۶۴ بیتی
MUL	R-Type	010	ضرب با الگوریتم Karatsuba
DIV	R-Type	011	تقسیم با الگوریتم Restoring
LOAD	M-Type	100	بارگذاری ۲ بایت داده از آدرس reg[base]] + sign_ext(Address) به reg[rd]
STORE	M-Type	101	ذخیره ۲ بایت داده در آدرس reg[base]] + sign_ext(Address) از reg[rd]

توضیحات واحد محاسباتی (ALU)

واحد محاسباتی این پردازنده برای داده‌های ۶۴ بیتی علامتدار مکمل ۲[۵] طراحی شده و از عملیات‌های جمع، تفریق، ضرب و تقسیم پشتیبانی می‌کند. هر یک از این عملیات با الگوریتم خاصی پیاده‌سازی شده‌اند که در ادامه شرح داده می‌شود.

[۵] 2's Complement

جمع و تفریق: الگوریتم Carry Select Adder

برای پیاده‌سازی عملیات‌های جمع و تفریق، از الگوریتم Carry Select Adder استفاده شده است. در این روش:

- داده ۱۶ بیتی به ۴ بلوک ۴ بیتی تقسیم می‌شود.
- برای هر بلوک (به جز اولین بلوک)، دو جمع موازی انجام می‌شود: یکی با فرض بیت حمل ورودی صفر و دیگری با فرض بیت حمل ورودی یک.
- پس از تعیین بیت حمل از بلوک قبلی، نتیجه مناسب از بین دو جمع انتخاب می‌شود.
- این روش سرعت بیشتری نسبت به Ripple Carry Adder دارد و پیچیدگی منطقی قابل قبولی دارد.

برای تفریق نیز از همین ساختار استفاده می‌شود، با این تفاوت که عملوند دوم به‌صورت مکمل دو در آمده و سپس جمع انجام می‌شود.
توجه: این ساختار باید در سطح گیت پیاده‌سازی شود.

ضرب: الگوریتم Karatsuba با ضرب‌کننده ۸ بیتی Shift and Add

عملیات ضرب برای ورودی‌های ۱۶ بیتی با الگوریتم Karatsuba پیاده‌سازی شده است. این الگوریتم یک ضرب بازگشتی است که پیچیدگی زمانی آن کمتر از ضرب کلاسیک است.

- ابتدا دو عدد ۱۶ بیتی به دو نیمه ۸ بیتی تقسیم می‌شوند:

$$X = X_H \times ۲^۸ + X_L, \quad Y = Y_H \times ۲^۸ + Y_L$$

- سپس سه ضرب ۸ بیتی انجام می‌شود:

$$Z_۰ = X_L \times Y_L, \quad Z_۲ = X_H \times Y_H, \quad Z_۱ = (X_H + X_L) \times (Y_H + Y_L) - Z_۰ - Z_۲$$

- حاصل ضرب نهایی به‌صورت ترکیبی از این سه مقدار محاسبه می‌شود:

$$P = Z_۲ \times ۲^{۱۶} + Z_۱ \times ۲^۸ + Z_۰$$

- هر ضرب ۸ بیتی با استفاده از الگوریتم Shift and Add انجام می‌شود. در این الگوریتم:
  - بیت‌های عملوند دوم یکی یکی بررسی می‌شوند.
  - در هر مرحله، اگر بیت برابر یک باشد، عملوند اول به حاصل جمع افزوده می‌شود.
  - سپس شیفت انجام می‌شود و مرحله بعد تکرار می‌شود.

تقسیم: الگوریتم Restoring Division

برای پیاده‌سازی تقسیم از الگوریتم Restoring Division استفاده شده است که یک الگوریتم کلاسیک و ساده برای پیاده‌سازی سخت‌افزاری تقسیم است. در این الگوریتم:

- مقسوم و مقسوم‌علیه ۱۶ بیتی هستند.
- در هر مرحله، باقی‌مانده به چپ شیفت داده می‌شود و یک بیت از مقسوم وارد باقی‌مانده می‌شود.
- سپس مقسوم‌علیه از باقی‌مانده کم می‌شود.
- اگر نتیجه منفی شود، باقی‌مانده به حالت قبل بازمی‌گردد و بیت خارج‌قسمت مربوطه صفر در نظر گرفته می‌شود.
- اگر نتیجه مثبت باشد، بیت خارج‌قسمت برابر یک خواهد بود.
- این روند برای ۱۶ مرحله تکرار می‌شود و در نهایت خارج‌قسمت و باقی‌مانده نهایی به دست می‌آید.

رجیستر فایل

رجیستر فایل این پردازنده شامل ۴ رجیستر ۱۶ بیتی است که برای نگهداری مقادیر موقت و عملوندهای محاسباتی مورد استفاده قرار می‌گیرند. مشخصات این رجیستر فایل به صورت زیر است:

- تعداد رجیسترها: ۴ عدد، با نام‌گذاری x0 تا x3
- عرض هر رجیستر: ۱۶ بیت
- دو پورت خواندن: برای استخراج دو عملوند به‌صورت هم‌زمان
- یک پورت نوشتن: برای ذخیره نتیجه عملیات در رجیستر مقصد
- عملیات خواندن در لبه پایین رونده و نوشتن در لبه‌ی بالارونده کلاک انجام می‌شود، مشروط به فعال بودن سیگنال write_enable

ساختار حافظه اصلی

حافظه اصلی پردازنده یک حافظه کلمه‌ـآدرس پذیر است که کلمات آن ۲ بایتی هستند. این حافظه دارای مشخصات زیر است:

- اندازه: دارای 2^{16} آدرس که هر کدام ۲ بایت داده دارند.
- عرض دسترسی: ۱۶ بیت (۲ بایت)
- استفاده از این حافظه تنها از طریق دستورات LOAD و STORE و به عنوان حافظه دستورات (برای واکشی دستور) مجاز است.

دقت شود از آنجا که یک حافظه اصلی داریم، از آن هم برای نگهداری دستورات و هم داده استفاده می‌کنیم.

ساختار واحد کنترلی

واحد کنترلی شامل یک رجیستر PC است که یک نشانگر به آدرس آغاز دستورات در حافظه اصلی است. پس از دیکود کردن دستور، برای هر کدام از ALU و حافظه اصلی و رجیستر فایل، سینگال‌های کنترلی مربوطه مانند سیگنال خواندن، نوشتن و عملیات انجام شده را تعیین می‌کند. در انتها نیز پس از پایان اجرای دستور، سیگنال ready برابر ۱ شده و با افزوده شدن یک واحد به PC، دستور بعدی واکشی می‌شود.

تست و ارزیابی

شما باید یک (یا چند) تست بنچ شامل تمامی دستورات ذکر شده بنویسید که نتیجه آنها با استفاده از waveform قابل درک باشد. همچنین گزارشی از نحوه عملکرد و پیاده‌سازی ارائه دهید. همچنین کد خود را سنتز کرده و خروجی سنتز شده را نیز قرار دهید.