مستندات طراحی معماری پردازنده RISC-V

۱. مقدمه و فلسفه طراحی

این مستند، جزئیات کامل معماری پردازنده ۳۲ بیتی طراحی شده بر اساس مجموعه دستورالعمل **RISC-V (نسخه RV32I به همراه افزونه M)** را تشریح می‌کند. هدف اصلی این طراحی، ساخت یک پردازنده کارا، قابل فهم و ماژولار بوده است که بتواند مجموعه دستورالعمل‌های الزامی پروژه را به درستی اجرا کند.

معماری ما بر اساس اصول زیر بنا شده است:

- **معماری تک‌چرخه‌ای (Single-Cycle):** برای سادگی در طراحی و دیباگ کردن، یک معماری تک‌چرخه‌ای انتخاب شد. در این مدل، هر دستورالعمل در یک پالس ساعت کامل اجرا می‌شود. این رویکرد، درک جریان داده و منطق کنترل را بسیار ساده‌تر می‌کند.
- **معماری Von Neumann:** طبق الزامات پروژه، از یک حافظه واحد برای نگهداری همزمان دستورالعمل‌ها (Code) و داده‌ها (Data) استفاده شده است. این کار، طراحی حافظه را ساده کرده و انعطاف‌پذیری بیشتری را فراهم می‌کند.

۲. اجزای اصلی مسیر داده (Datapath Components)

مسیر داده، اسکلت اصلی پردازنده است که از مجموعه‌ای از قطعات سخت‌افزاری و اتصالات بین آن‌ها تشکیل شده است. در ادامه، هر یک از این قطعات تشریح می‌شوند.

۲.۱. شمارنده برنامه (Program Counter - PC)

یک رجیستر ۳۲ بیتی است که آدرس دستورالعمل بعدی که باید از حافظه واکشی شود را در خود نگه می‌دارد. منطق به‌روزرسانی PC به شرح زیر است:

- **حالت عادی:** در هر پالس ساعت، مقدار PC با ۴ جمع شده و آدرس دستور بعدی (PC + 4) محاسبه می‌شود.
- **حالت پرش (Branch):** در صورت برقراری شرط یک دستور انشعاب (مانند beq)، مقدار PC با یک آفست محاسبه شده جمع شده و به آدرس مقصد پرش می‌کند.

۲.۲. حافظه دستور و داده (Instruction & Data Memory)

با توجه به معماری Von Neumann، از دو قطعه حافظه مجزا در Logisim برای شبیه‌سازی یک حافظه واحد استفاده شده است:

- **Instruction Memory (ROM):** یک حافظه فقط-خواندنی (ROM) که برنامه باینری (.bin) در آن بارگذاری می‌شود. این حافظه دستورالعمل‌ها را بر اساس آدرس دریافتی از PC واکشی می‌کند.
- **Data Memory (RAM):** یک حافظه خواندنی-نوشتنی (RAM) که برای اجرای دستورات lw و sw استفاده می‌شود. آدرس مورد نظر از خروجی ALU و داده برای نوشتن از فایل ثبات‌ها

تامین می‌شود.

۲.۳. فایل ثبات‌ها (Register File)

این قطعه به صورت یک مدار سفارشی (Subcircuit) ساخته شده است و قلب پردازنده محسوب می‌شود.

- **ساختار:** شامل ۳۲ رجیستر ۳۲ بیتی است.
- **قابلیت‌ها:**
  - **دو پورت خواندن همزمان:** با گرفتن آدرس‌های ۵ بیتی `rs1` و `rs2`، می‌تواند محتوای دو رجیستر را به صورت همزمان روی خروجی‌های `ReadData1` و `ReadData2` قرار دهد.
  - **یک پورت نوشتن:** با گرفتن آدرس ۵ بیتی `rd`، داده ۳۲ بیتی `WriteData` و فعال شدن سیگنال کنترلی `RegWrite`، مقدار جدید را در رجیستر مقصد می‌نویسد.

۲.۴. واحد محاسبه و منطق (ALU)

این واحد نیز به صورت یک مدار سفارشی ساخته شده و مسئول انجام تمام عملیات‌های حسابی و منطقی است.

- **ورودی‌ها:** دو ورودی داده ۳۲ بیتی (`A` و `B`) و یک ورودی کنترل ۴ بیتی (`ALUControl`).
- **خروجی‌ها:** یک خروجی ۳۲ بیتی برای نتیجه (`Result`) و یک خروجی ۱ بیتی (`Zero`) که در صورت صفر بودن نتیجه، فعال می‌شود.
- **عملیات پشتیبانی شده:** این ALU قادر به انجام عملیات‌های `ADD`, `SUB`, `AND`, `OR`, `XOR` و `SLT` بر اساس کد دریافتی از واحد کنترل ALU است.

۲.۵. واحد گسترش علامت (Sign Extend)

این واحد مسئول تبدیل مقادیر ثابت (۱۲ (immediate بیتی که از دستورالعمل استخراج می‌شوند، به یک عدد ۳۲ بیتی با حفظ علامت است. این کار برای انجام صحیح محاسبات در ALU ضروری است.

۲.۶. مالتی‌پلکسرها (Multiplexers)

از مالتی‌پلکسرها در نقاط کلیدی مسیر داده برای انتخاب بین منابع مختلف داده استفاده شده است:

- **`ALUSrc` Mux:** تصمیم می‌گیرد که ورودی دوم ALU از فایل ثبات‌ها (`rs2`) بیاید یا از واحد گسترش علامت (`immediate`).
- **`MemToReg` Mux:** تصمیم می‌گیرد که داده‌ای که باید در فایل ثبات‌ها نوشته شود، از خروجی ALU می‌آید یا از خروجی حافظه داده.
- **Mux انتخاب PC:** تصمیم می‌گیرد که مقدار بعدی PC، حالت عادی (`PC+4`) باشد یا آدرس مقصد یک دستور پرش.

۳. واحد کنترل (Control Unit)

واحد کنترل، مغز متفکر پردازنده است که با رمزگشایی دستورالعمل، سیگنال‌های لازم برای مدیریت مسیر داده را تولید می‌کند. این واحد از دو بخش اصلی تشکیل شده است:

۳.۱. واحد کنترل اصلی

این واحد به صورت یک مدار سفارشی مبتنی بر **ROM** طراحی شده است.

- **ورودی:** opcode هفت بیتی دستورالعمل.
- **خروجی:** تمام سیگنال‌های کنترلی اصلی مانند RegWrite, ALUSrc, MemToReg, MemWrite, Branch, Jump و ALUOp.
- **منطق کار:** opcode به عنوان آدرس به ROM داده می‌شود و ROM کلمه کنترلی از پیش برنامه‌ریزی شده برای آن دستور را روی خروجی قرار می‌دهد.

۳.۲. واحد کنترل ALU

این واحد یک مدار ترکیبی کوچک است که تصمیم می‌گیرد ALU دقیقاً چه عملیاتی را باید انجام دهد.

- **ورودی‌ها:** سیگنال ۲ بیتی ALUOp (از واحد کنترل اصلی) و فیلدهای funct3 و funct7 دستورالعمل.
- **خروجی:** یک کد ۴ بیتی که مستقیماً به ورودی کنترل ALU اصلی متصل می‌شود.
- **منطق کار:** بر اساس مقدار ALUOp، تصمیم می‌گیرد که آیا عملیات باید یک جمع ساده باشد (برای lw/sw)، یک تفریق باشد (برای beq)، یا بر اساس فیلدهای funct یک دستور R-type تعیین شود.

۴. جریان داده و ریزعملیات‌ها (RTL)

که در یک فایل جدا به طور مفصل بررسی کردیم(داخل پوشه duc)