3-作业-SoC芯片架构设计

呼子博-2024111302

一、以五级流水的RISC-V微处理器为例，简述微处理器工作原理，应覆盖指令集、微架构、流水线等基本概念。

RISC-V 是一种精简指令集架构（RISC）

指令集：是微处理器与软件进行交互的接口，定义了处理器能够执行的基本操作。RISC-V 指令集以其简单性和模块化设计为特色，核心包括以下几类指令：指令长度固定为32 bit

- 算术和逻辑指令：如加法（ADD）、减法（SUB）、与（AND）、或（OR）。
- 数据传输指令：如加载（LW）和存储（SW）指令，用于将数据在寄存器和内存之间移动。
- 控制流指令：如跳转（JAL）、条件分支（BEQ、BNE），用于改变程序执行的顺序。
- 立即数指令：如 ADDI，直接使用常数作为操作数。
- 扩展指令（可选）：如浮点运算、向量运算等，可根据需求进行扩展。
  指令格式

1. **R 型（寄存器）**：
   - 用于算术和逻辑运算。
   - 格式：`opcode, rd, funct3, rs1, rs2, funct7`
   - 示例：`add rd, rs1, rs2`
2. **I 型（立即数）**：
   - 用于立即数操作、加载指令。
   - 格式：`opcode, rd, funct3, rs1, imm`
   - 示例：`addi rd, rs1, imm`
3. **S 型（存储）**：
   - 用于存储指令。
   - 格式：`opcode, imm[11:5], funct3, rs1, rs2, imm[4:0]`
   - 示例：`sw rs2, imm(rs1)`
4. **B 型（分支）**：
   - 用于条件分支指令。
   - 格式：`opcode, imm[12], imm[10:5], funct3, rs1, rs2, imm[4:1], imm[11]`
   - 示例：`beq rs1, rs2, offset`
5. **U 型（上位立即数）**：
   - 用于加载上位立即数。
   - 格式：`opcode, rd, imm[31:12]`
   - 示例：`lui rd, imm`

6. **J 型（跳转）**：
   - 用于无条件跳转。
   - 格式：`opcode, rd, imm[20], imm[10:1], imm[11], imm[19:12]`
   - 示例：`jal rd, offset`

微架构：是指实现指令集架构的具体硬件设计，如：

- 寄存器：存储处理器的临时数据和指令操作数。
- 算术逻辑单元（ALU）：执行算术和逻辑运算。
- 程序计数器（PC）：跟踪当前正在执行的指令地址。
- 指令存储器和数据存储器：分别存储程序指令和运行时数据。
- 控制单元：生成控制信号，协调各模块的工作。
- 流水线寄存器：用于在流水线各阶段之间传递数据。

流水线是一种提高微处理器执行效率的技术，将指令执行分解为多个阶段，每个阶段在不同的硬件模块上并行执行。五级流水线的 RISC-V 微处理器分为以下五个阶段：

1. 取指（IF, Instruction Fetch）
2. 指令解码（ID, Instruction Decode）
3. 执行（EX, Execute）
4. 访存（MEM, Memory Access）
5. 写回（WB, Write Back）
   在五级流水线中，多个指令可以同时处于不同的执行阶段，从而实现指令的并行处理。
   流水线的方式提高了指令吞吐量，但也引入了下面问题：
   (1) 数据冒险（Data Hazard）
- 当一条指令需要使用上一条指令的结果时，可能会因为数据未准备好而导致延迟。
- 解决方法：通过数据转发（Forwarding）或插入气泡（Stall）来处理。
  (2) 控制冒险（Control Hazard）
- 当遇到分支指令时，下一条指令的地址可能不确定。
- 解决方法：分支预测（Branch Prediction）或延迟槽（Delay Slot）技术。
  (3) 结构冒险（Structural Hazard）
- 当多个指令同时访问同一个硬件资源（如存储器）时，可能发生冲突。
- 解决方法：通过资源复用或增加硬件资源来避免。

二、简述微处理器和专用集成电路ASIC各自特点。

微处理器（Microprocessor）通用处理器，能够执行多种任务，广泛用于计算设备中。

- 功能：通用，可编程
- 灵活性：高
- 性能：较低，受限于指令集架构
- 开发周期：短（只需要开发软件）
- 开发成本：低

- 单位功耗：较高
- 应用场景：多任务通用应用
    - 如：PC、移动设备

专用集成电路（ASIC, Application-Specific Integrated Circuit）为特定应用设计的定制芯片，专注于执行特定任务。

- 功能：专用，功能固定
- 灵活性：低
- 性能：高，针对特定任务优化
- 开发周期：长（要从头开始进行硬件设计、验证和制造）
- 开发成本：高（非重复性工程成本 NRE 成本高）
- 单位功耗：较低
- 应用场景：高性能、固定功能应用
    - 如：通信、数据中心、视频解码器、工业控制和汽车电子

三、列举你知道的存储器类型，简述其工作原理，并解释“存储层次”这一概念。

> 计算机组成原理中的概念

存储器类型	工作原理	特点
寄存器（Registers）	- 寄存器是由触发器（Flip-Flops）组成的小型存储单元，用于存储处理器执行指令时的临时数据和操作数。 - 数据可以在一个时钟周期内被读取或写入。	位于处理器内部，速度最快，容量最小
高速缓存（Cache）	- 高速缓存使用 SRAM（静态随机存储器）技术实现。 - 它通过存储最近访问的数据或预测即将访问的数据，减少处理器对主存的访问频率。 - 使用**局部性原理**（时间局部性和空间局部性）优化性能： 　- **时间局部性**：最近访问的数据很可能会再次访问。 　- **空间局部性**：与当前访问数据地址相邻的数据很可能会被访问。	位于处理器和主存之间。缓解处理器与主存之间的速度差异，提高系统性能
主存（Main Memory）	- 通常由 DRAM（动态随机存储器）实现 DRAM 由电容和晶体管组成，每个存储单元存储一位数据。 - 数据通过对电容的充电和放电表示“1”或“0”，但由于电容会漏电，需定期刷新数据（称为“刷新操作”）。	直接与处理器和缓存相连，存储正在运行的程序和数据，作为处理器的主要工作存储器

存储器类型	工作原理	特点
固态存储（SSD, Solid-State Drive）	- 数据存储在 NAND 闪存单元中，通过电子隧穿效应改变浮动栅极的电荷状态来表示“1”或“0”。 - 由于是非易失性存储器，断电后数据仍然保留。	快于传统硬盘（HDD），但慢于主存
硬盘（HDD, Hard Disk Drive）	- 硬盘由旋转的磁性盘片和磁头组成。 - 磁头通过改变磁性盘片某一区域的磁化方向来存储数据（磁化方向表示二进制的“1”或“0”）。	大容量、长期存储
静态随机存取存储器SRAM	- 每个 SRAM 存储单元由多个晶体管组成，通常是六个（6T SRAM） - 通过改变反相器的状态来存储二进制数据（0 或 1）。 - 由于是静态的，数据在电源不断电的情况下会一直保持。	主要用于需要快速访问的存储器，如 CPU 缓存
动态随机存取存储器	- 每个 DRAM 存储单元由一个电容和一个晶体管组成。电容用于存储电荷，表示二进制数据（1 或 0）。晶体管用于控制对电容的访问。 - 刷新操作：由于电容会漏电，数据会逐渐丢失 - 为了保持数据，需要定期刷新电容中的电荷。 - 刷新频率通常在数毫秒级别。	主要用于计算机的主存（RAM），以其高存储密度和低成本著称。
只读存储器ROM	- 数据在制造过程中写入，通常不能被修改。 - 通过一系列固定的电路实现数据的存储 - 类型 - 掩模 ROM（Mask ROM）：数据在制造时写入，无法更改 - 可编程 ROM（PROM）：制造后可编程一次 - 可擦除可编程 ROM（EPROM）：可以通过紫外线擦除数据，再次编程 - 电可擦除可编程 ROM（EEPROM） - 闪存（Flash Memory）:一种特殊的 EEPROM，支持快速擦除和写入。 - 广泛用于 USB 驱动器、固态硬盘等。	主要用于存储固化的数据，这些数据在正常操作中不会被修改

其他的

- 按存储戒指分类：半导体、光、磁
- 按存储方式分类：随机存储器、顺序存储器
- 按读写功能分类：ROM、RAM
- 按用途分类：主存、辅存、高速缓存

存储层次，是计算机系统中不同类型存储器之间的分级结构，旨在平衡 **速度**、**容量** 和 **成本**，以优化性能和成本效益。越靠近顶层的，速度越快，容量越小，成本越高；靠近底层的，速度越慢，容量越大，成本越低。
存储器的层次结构主要体现在缓存-主存 以及 主存-辅存两个存储层次

- 缓存-主存，解决CPU和主存速度不匹配的问题
- 主存-辅存，解决存储系统容量问题

截止时间 2024年11月30日 23:59