目录

1 第四章 cpu 的组成

2

3 概述

3.1 cpu 的构造法

对于一个 cpu , 其中分为三个部分. 第一个部分是处理器, 第二部分是内存, 第三部分是总线.

处理器, 里面由 ALU, 寄存器, PC

内存, 就是内存相关的, Cache 缓存, memory 内存.

总线, 就是 IO 相关的

3.2 PC

PC 就是存储着指令, 其告诉 ALU 该如何处理寄存器.

3.3 ALU

ALU 就是运算单元, 其能够执行 add sub 等指令, 对寄存器进行操作.

3.4 CACHE

3.4.1 Cache 简介

Cache 是缓存, 是一种更为快速的内存

对于这个存储单元的层级, 我们可以回想一下, 寄存器是稀少而快速, 内存是数量多而没那么快速, 缓存应该是介于这两者之间. 价格也是介于两者之间.

Cache 的构想应该和 register 的构想很类似, 内存之中数据暂存在 cache 之中, 以此达成加速的目的.

3.4.2 SRAM

Cache memory operates between 10 to 100 times faster than RAM, requiring only a few nanoseconds to respond to a CPU request. The name of the actual hardware that is used for cache memory is high-speed static random access memory (SRAM).

实现的 Cache 的硬件为 SRAM, 全称为 high-speed static random access memory. 于 SRAM 对应的便是 DRAM 后者用于 memory 的构造之中. S for static , SRAM 之中的数据, 只要 power 没有断开, 不同于 DRAM 的数据那样, 就能够已知保存, 但是 DRAM 之中的数据, 几秒钟后就会消失, 需要周期性地 refresh 一下. 这就是为什么其称为 static 的. SRAM use latches and transistor to store data.

3.4.3 Cache 的多级建构

Cache 的构造实际上更为复杂, Cache 分为了三个 Level, L3 速度最慢, 是用来辅助 L1, L2 的.

Cache memory is a small amount of fast memory that is used to store frequently accessed data. It is located close to the CPU and is used to reduce the average time it takes to access data from the main memory.

There are three general levels of cache: L1 cache, L2 cache, and L3 cache. Each level is differentiated by its speed, size, and proximity to the CPU.

L1 cache, or primary cache, is extremely fast but relatively small. It is usually embedded in the processor chip as CPU cache[3].

L2 cache, or secondary cache, is often more capacious than L1 but slower. It is usually located on the processor chip or on a separate chip on the motherboard[3].

L3 cache, or tertiary cache, is larger than L2 but slower. It is usually located on a separate chip on the motherboard[3].

Is there anything else you would like to know about cache memory?

Computers, Explained -SearchStorage. https://www.techtarget.com/searchstorage/definition/cache-memory 访问时间 2023/4/1.

(2) Explainer: L1 vs. L2 vs. L3 Cache | TechSpot. https://www.techspot.com/article/2066-cpu-l1-l2-l3-cache/ 访问时间 2023/4/1.

(3) How Does CPU Cache Work and What Are L1, L2, and L3 Cache? - MUO. https://www.makeuseof.com/tag/what-is-cpu-cache/ 访问时间 2023/4/1.

3.5 Memory

3.5.1 TODO describe the memory

4 interlude

4.1 Havard architecture and Princeton architecture

程序有存储的地方, 数据也有存储的地方, 我们常将其抽象出来, 认为这两个部分是存在一个地方里的. 但实际上有区别.

数据的存储单元, 程序的存储单元, 在哈佛架构之中是分开的. 也就是说, 在哈佛架构中, 存在两个单元, 分别由两个总线进行单元和处理器之间的数据传输

5 组件的硬件构成

5.1 Processor 的构成

5.1.1 ALU Register PC Extender

1. ALU 是算术运算单元, 即, 进行 and or add sub 运算的单元.
2. Register 我们很熟了, 支持读写操作. PC 我们也很熟了, 支持 +4 , + imm 操作.
3. Extender 是 imm 生成单元也是扩展单元, 我们有的时候需要进行一些数据的符号扩展.

5.1.2 ALU

输入两个 32 位数据, 输出一个 32 位的数据. 进行位运算或者加减运算.

1. ALU 的接线 一个一位 ALU 应
   (a) 根据 ALUop 的值, 决定操作
   (b) 根据输入 A B 给出结果
   (c) 判断是否溢出, Overflow 为 1 如果其溢出
   (d) 判断结果是否为 0, Zero 为 1 如果其为 0
   (e) 有一个用于串联的进位 Carry Out.
2. ALUop 的功能

ALUop	操作	
0000	and	
0001	or	
0010	add	
0110	sub	
0111	slt	set on less than
1100	nor	nor

3. 构建简单的 ALU 以 Mutiplexer 为基础, 而后构建 and or add 操作 and 使用 and 门, or 使用 or 门, add 使用一个一位 Full adder. 构建是简单的.
4. one bit ALUs 的串联 [4/5] 在串联之中我们要实现
   ☒ sub 操作
   ☒ slt 操作
   ☐ nor 操作
   ☒ Overflow 判断
   ☒ Zero 判断

(a) sub 操作 引入 Binvert 后更名为 Bnegate, 其有两个操作

i. 在 one bit ALU 之中, 使 B 取反

ii. 接入末位 ALU 也就是 ALU0 的 CarryIn 也就是进位之中

因为 Binvert 是一个信号来的, 高电平使能, 那么其要触发的时候, 值为 1, 并且还是一个一位的信号.

这就有 R = A+B'+1. 也就有 R = A-B

(b) slt 操作 If A<B then result equates 1 else result equates 0.

我们用 Less 作为 one bit ALU 的输入信号.

我们只需要计算出 A - B 的值, 让后将这个值的符号传回末位 (Less 在其他位的值均为 0), 最后 result 直接等于 Less 就行了.

(c) nor 操作

(d) Overflow 判断 设输入的两个符号位为 s1 s2 , 结果的符号位为 s3, 那么有

Overflow = (s1 and s2) xor s3

就有, 当溢出发生的时候, Overflow 为 1.

(e) Zero 判断 每一位结果取 not-or 就行了.

5.1.3 PC

1.

5.1.4 Register

我们应该有这些功能:

1. 根据 Register 编号 Rw 将 busW 写入到寄存器之中
2. 根据 Register 编号 Ra Rb 将寄存器的值输出到 busA, busB 上

并且读操作不应收到时钟控制.