What did you learn?

> 硬件方面的介绍

Number 1 : Different Types of Processors

- 通用处理器（general-purpose processor）
- 具有指令集扩展的通用处理器（general-purpose processor with instruction set extensions）
- 专用处理器/协处理器（special-purpose processor/co-processor）
- FPGA（Field Programmable Gate Array）

图灵机：理解为**可计算**就行，是一个现实可以存在，可以计算出来结果的东西。

> 更深层次的理解：好多悖论其实是由于尴尬的**自指**：当一个系统强大到一定程度时，终究会遇到无法处理自己的窘境。因此，不存在一台图灵机，可以判定任意图灵机是否会停机，也不存在一台图灵机，可以判定任意图灵机是否会输出自己的描述。即，图灵机不是万能的，因此是现实存在的东西。

1. **通用处理器**：一般通用处理器没有严格定义，普遍上认为处理器是"图灵完备"（可以解决图灵机可以解决的任何问题）的就可以了，即可以计算任何实际可以计算的内容。一些研究人员已经证明只需要一条指令即可实现图灵完备。在现代处理器的背景下，大多数可编程 CPU 都被认为是通用的。例如，Intel 的 x86 处理器，ARM 处理器，MIPS 处理器，PowerPC 处理器，SPARC 处理器等等。

   > 通用化的代价通常是性能的损失，就像一个博士当然可以学会怎么打螺丝，怎么装系统，怎么写论文。但是让一个博士去一直打螺丝，那么他的效率就会大大降低。因此，通用处理器的性能通常不如专用处理器。

2. **具有指令集扩展的通用处理器**：处理器设计者可以将指令集扩展合并到基本微架构（micro-architecture）中以适应重复性任务。从功能上讲，微架构功能可能没有差异，但实际上，最终用户可能会获得巨大的性能提升。

   > 拿密码学中的 AES 举例，如果有一台带有 AES 加密磁盘的台式机，想读取数据的话就得进行一个 CPU 中断，然后进行解密再把数据读到缓存里。本来访问未命中的缓存就很慢了，再加上解密，速度就更慢了。这时候 AES 就是复杂的重复任务，并且对于通用 CPU，别无选择，只能将解密实现为线性流操作。 Intel 和 AMD 都认识到磁盘加密的需求以及 AES 对次级存储（secondary storage）访问带来的开销负担，并于 2010 年左右开发了 AES-NI x86 指令集扩展，以加速其桌面 CPU 系列上的磁盘加密。

3. **专用处理器/协处理器**：如果想完全加速某个运算，最好的办法就是设计一个专用处理器或者专用集成电路（Application-specific integrated circuit (**ASIC**)），以灵活性换速度。这些类型的处理器通常与通用处理器紧密耦合，因此称为协处理器。协处理器一般会被封装，但是不一定会进通用框架，例如CPU可以集成声卡、显卡、网卡等等。这种附加功能通过专用寄存器和协处理器公开，该协处理器被视为通用处理器必须管理的单独组件。

   > 就像死神里的灭火皇子汪怀达斯一样，就是专门用来封印流刃若火，才被制造出来的，也正因此其丧失了语言、意识、知识与理智，仅保留了灭火与战斗这两项能力

4. **FPGA**：现场可编程门阵列。介于通用和专用之间，适用于需要高性能的表现（*专用*），但是需要（不频繁）修改（*通用*）的场景。FPGA 提供了可重新编程的灵活性，同时生成专用逻辑来计算目标应用。其与

> 通用处理器的主要区别在于用户如何设计和构建应用程序。为了充分利用硬件，用户必须使用硬件描述语言（Verilog 或 VHDL）将应用程序描述为一组硬件组件和事件（这个过程就像在用 FPGA 制作通用/专用处理器的原型机）。但是拥有两者的优先就意味着两者的缺点都有，不仅设计麻烦，而且能耗和硬件也不小。目前，ARM 可以将 FPGA 作为一个灵活的协处理器使用。

总之，通用处理器能够计算任何可计算的东西。类似地，对于具有指令集扩展的通用处理器，它在特定应用中可能表现更好。专用处理器（或协处理器）在执行特定任务时速度非常快，但无法计算除该任务之外的任何内容。 FPGA 可用于构建上述所有硬件，但与 ASIC 解决方案相比，牺牲了速度以换取灵活性。

Number 2 : What is the difference between a multi-core processor and a vector processor?

研究这个问题前首先得搞清楚啥是并行计算，并行和多核的概念往往是相辅相成的。

1. 什么是并行计算（parallel computing）？

聊并行之前，肯定得先聊聊串行（serial）。串行就类似于所有的活儿都交给闪电侠做，做完一个再做一个，任务量少点还行，多了就麻了，这种再快也有瓶颈。有没有一种方法可以提高计算速度而不受处理器速度瓶颈的影响？答案就是并行计算（依靠人民群众）。

并行计算就是把问题拆分成数个小问题，这些小问题可以被同时单独计算（把大事件分成小事件，每个小事件交给不同的群众去做）。至于能提升多少取决于算法本身，可以用阿姆达尔定律(Amdahl's law)进行分析。至于实际怎么处理，就是多核处理器和矢量处理器的活儿了。

> 1. 阿姆达尔定律：在并行计算中，如果要提升整体速度，那么必须提升串行部分的速度，否则并行部分再快也没用。简单讲，如果一个程序有 10% 的串行部分，那么最多只能提升 10 倍的速度。
> 2. 人民英雄史观对个人英雄史观的绝对胜利

2. 什么是多核处理器（multi-core processor）？

多个人从事一个项目，每个人都被赋予不同的任务，但所有人都为同一个项目做出贡献。可能需要一些额外的组织工作，但完成项目的总体速度会更快。

3. 什么是矢量处理器（vector processor）？

矢量处理器是一种计算单个指令（就像串行处理器一样），但在一维数组中排列的**多个数据集**上执行这个指令的处理器（与处理单个数据集的标准串行处理器不同）。这里的想法是，如果对程序中的不同数据集多次执行相同的操作，为什么不针对所有数据集直接执行一次指令呢？缩写词 SIMD（Single Instruction Multiple Data）通常用于表示以这种方式工作的指令。

> 就像放电影一样。一个人一个厅就是串行；同时开好多个厅，每个厅一个人看就是多核；一个厅里放、好多个人同时看就是矢量。 食堂里一个师傅做一盘菜就是串行；好几个师傅同时做这盘菜就是多核；一个师傅做全部的辣椒炒肉、另一个师傅做全部的过油肉就是矢量。

总的来说，多核处理器有多个工作线程；矢量处理器有一种同时对多个任务执行相同操作的方法。

Number 3: Computational and storage power of different form factors

首先估计以下部件的相对计算和存储能力：

- 智能卡（smart card）

- 微控制器（micro-controller）/传感器节点（sensor node）
- 嵌入式或移动计算机（embedded or mobile computer）
- 笔记本电脑或台式电脑（laptop- or desktop-class computer.）

很显然我们可以通过比较处理器的时钟速度（clock speed）来评估处理器的速度，但是根据上一章的内容，我们会发现这样做会有很大的误导性：如果两个都是2GHz的处理器，但其中一个开启了并行计算，那么处理速度显然会加快。所以很难找到一个**直接的定量测量方法。**

对于某些特定设备（比如general purpose graphics cards，类似于GPU），经常用每秒浮点运算次数（FLOPS，floating point operations per second）来衡量其计算能力。但是这种方法也存在不合理性，所以现在的测量方法是比较解决某些特定问题的时间，设置多条基线（参考网站（CompuBench））， 而不是简单依靠某个定量指标。

> 就跟人一样，各有所长，得多方面比较，不能只用一个分数就把所有人分为三教九流

至于衡量设备的存储能力则相对简单很多了，只需要简单比较设备能够在永久存储中保存信息的大致字节数即可。

> 人话：直接比大小

智能卡（smart card）：

- 计算能力最弱，不同实现的时钟速度也不同，大致在20MHz左右
- 存储能力在2KiB左右

微控制器（micro-controller）：

- 单个集成电路上的小型计算机，包含一个处理器核心、存储器和可编程输入/输出外围设备
- 计算能力和存储能力随着定义变化很大
- 最典型传感器节点计算能力与智能卡相似，但是存储空间大点，a few KiB ~ a few MiB

嵌入式或移动计算机（embedded or mobile computer）：

- 计算能力和存储能力更大，且范围也更大，可以看看自己手机的配置
- 电量也随着时间的推移而增加

笔记本电脑或台式电脑（laptop- or desktop-class computer）：

- 计算能力和存储能力基本上比手机强（起码同时代上比）
- 具体比较则需要考虑更多的因素，比如如果某个CPU有图形处理单元（graphics processing unit，GPU，理解为集显），那它处理图像的能力就比没有的强