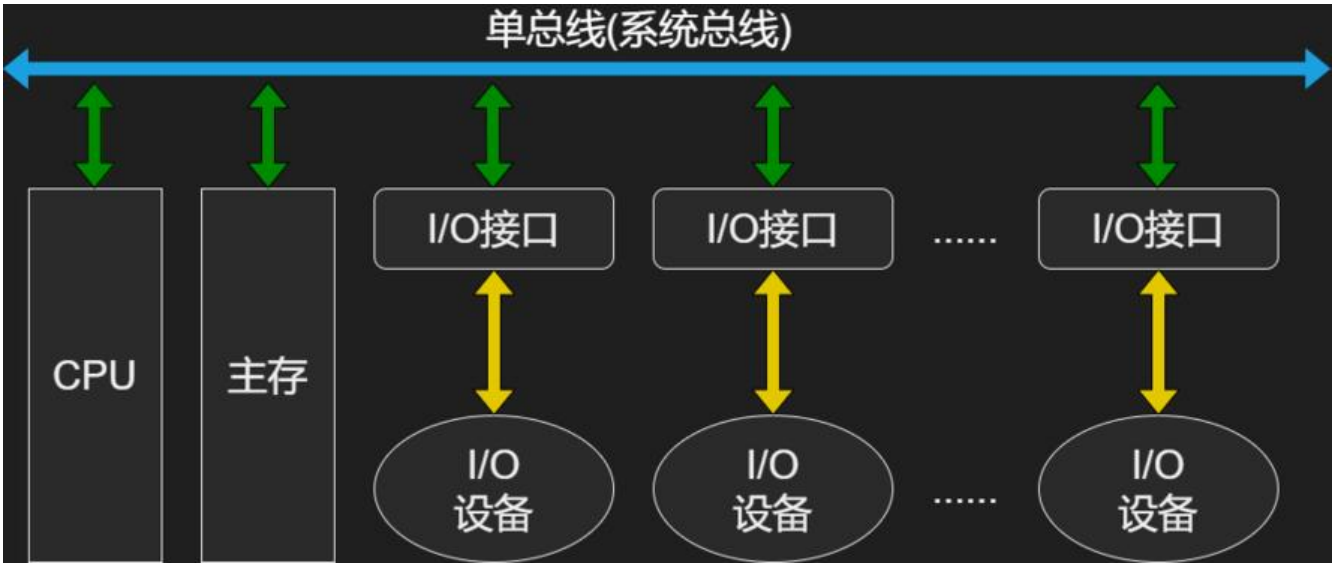
链滴

总线结构及其演进

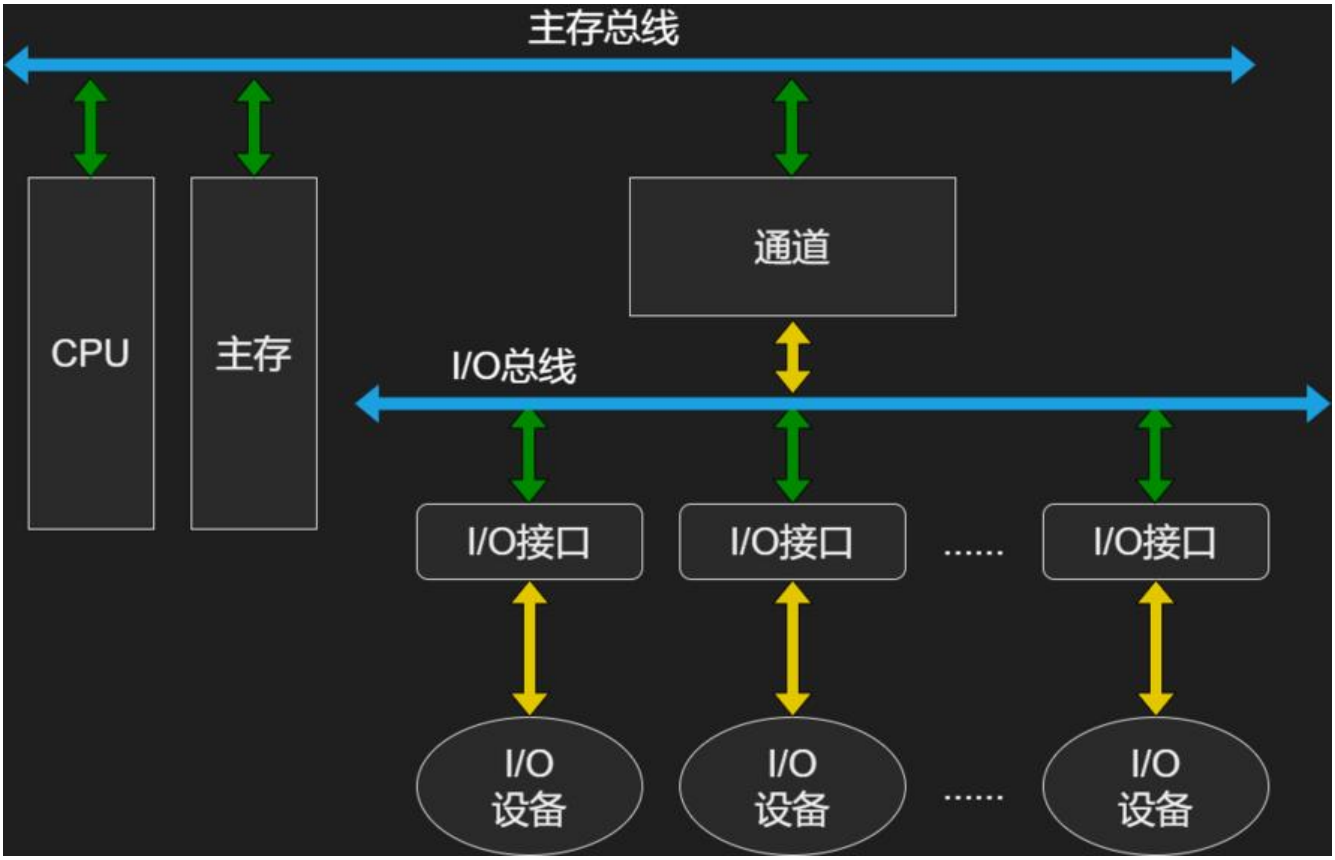
作者：zhangkeyang
原文链接：https://ld246.com/article/1613219734520
来源网站：链滴

1. 单总线结构



这种结构的缺点：CPU、主存以及所有的I/O设备都使用同一根总线进行数据传输，总线非常忙碌，性能成为了整个计算机系统的瓶颈。

2. 双总线结构

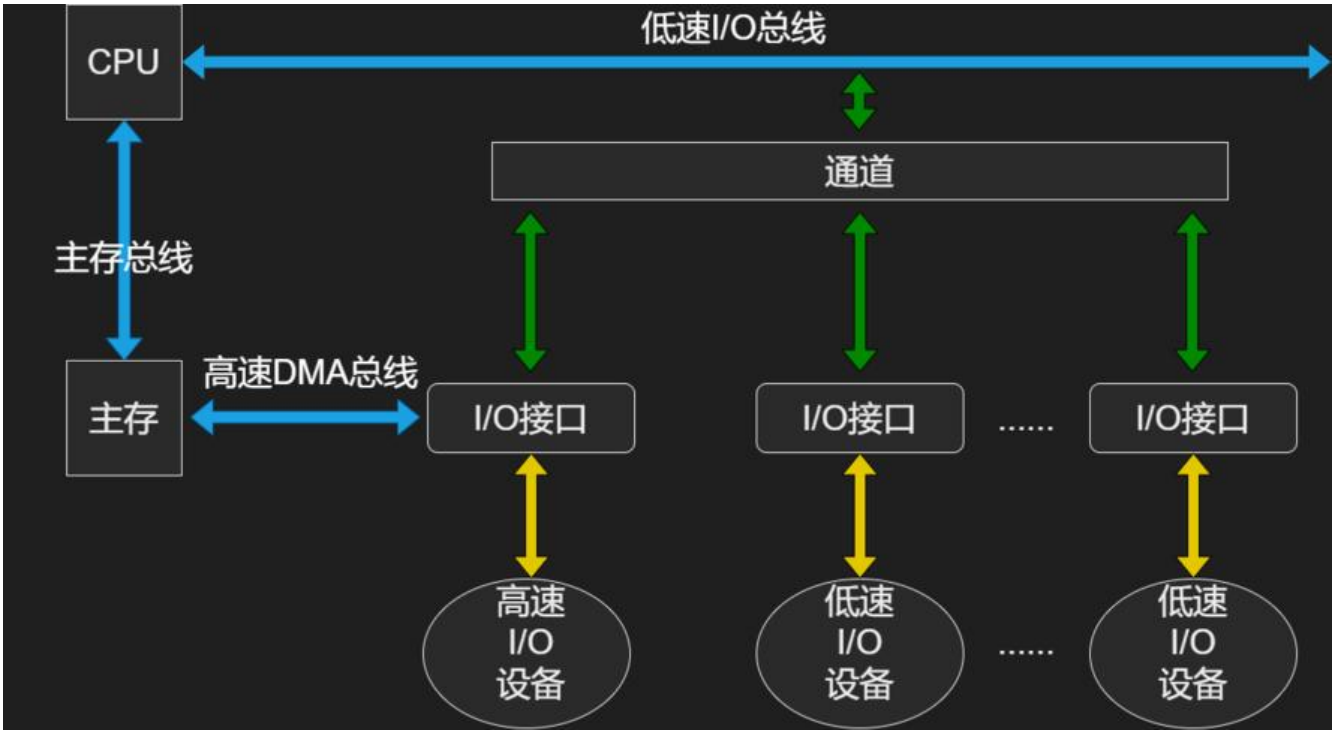


其中的**通道**是一种具有特殊功能的控制器，具有自己的指令系统，控制程序一般由操作系统来实现。算机系统使用这种控制器对所有的I/O设备进行统一管理。这种总线结构将CPU和主存之间的通讯与多I/O设备进行解耦，提高了CPU与主存的利用率，付出的代价是外设中的数据无法直接送入主存中也就是说，当需要对外设进行操作时，还是会存在阻塞现象。其实很多早期的计算机系统采用的都是

原文链接：总线结构及其演进

种设计结构。

3. 三总线结构

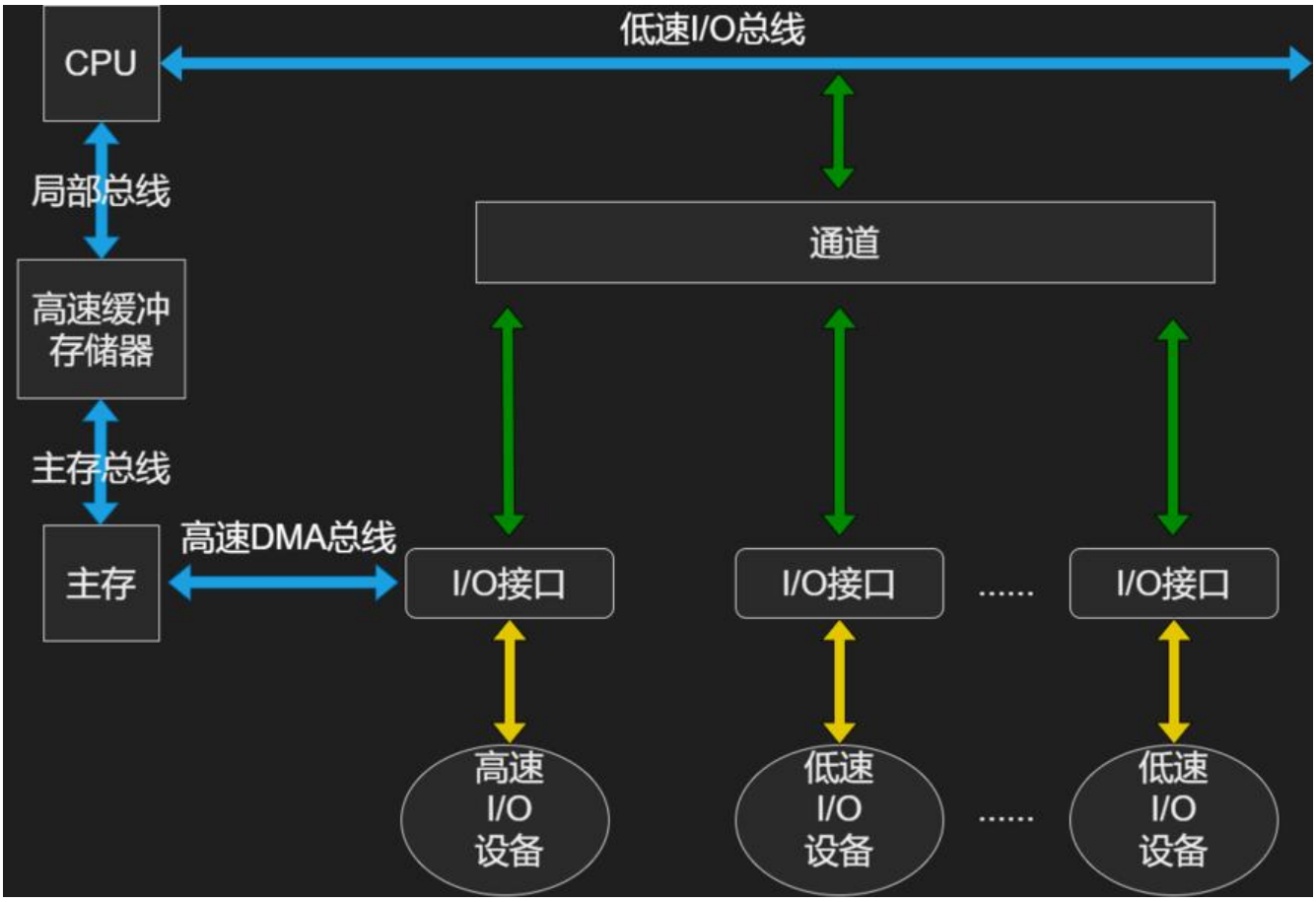


实际上双总线结构已经可以满足早期的计算需求了。但是现代计算机中，外设的种类和数量都大幅增，设计人员认为有必要进一步降低总线的压力，从而提高系统运行效率。经过调研和统计后人们发现有一部分外设的运行速度很快，并且需要频繁地与主存交换数据(例如硬盘、打印机等支持每秒钟传输千个字节至数十千个字节以上，并且每次传输都以固定长度的字节数为传输单元的设备)，于是将这些设称为**高速外设**；反之，其它大多数外设运行地都比较慢，并且与主存的交互不是很频繁(例如鼠标、盘、语音输入等支持每秒钟几个字节至几百个字节的设备)，将这些设备称为**低速外设。**

实际上，对于低速外设，优化的空间是很小的，而且即使是优化了设计，计算机系统整体的性能也不提高到哪里去，毕竟他们与系统中其它部件的交互并不频繁，投入-产出比很低。因此，没有必要让些低速外设接入主存总线，因为他们会打断CPU与主存的数据传输，从而形成系统瓶颈；只需要针对速外设进行硬件设计上的优化即可。

为了解决高速外设与主存之间的数据传输问题，设计人员引入第三根总线，专门用于在主存与高速外之间交换数据，这样就避免了高速外设对I/O总线的竞争，让高速设备可以直接与主存交换数据。这技术称为**直接主存访问，即DMA技术。**DMA总线的造价是昂贵的，并且硬件设计也很复杂，因此高设备仍然可以通过低速I/O总线来进行数据交互，但此时数据必须经过CPU。

5. 四总线结构



现代CPU的发展速度非常快(根据摩尔定律，CPU的性能每2年就可以翻倍)；但是，主存的发展却远远不上CPU的发展，这相当于原本齐头并进的两兄弟，其中一个突然拉胯，拖了兄弟的后腿。为了解决个问题，系统设计人员引入了**高速缓冲存储器(cache)**，它的读写速度非常接近于CPU，但造价很昂，大量使用缓存硬件将导致计算机系统的造价飙升。

好在人们发现，绝大多数计算机程序都遵循所谓的**局部性原理**，也就是说当CPU读取了主存中的某个据时，其相邻地址的数据往往也会被读取。所以，设计人员将容量很小的缓存硬件嵌入CPU当中；当CU读取主存中的数据时，连带将这个数据周围的数据也一起读取出来，放入缓存当中；下次要读取数时，先看看缓存中有没有数据，如果有数据就直接将其送入CPU；否则再访问主存，并再将其周围数放入缓存当中，以此类推。

这样，仅仅使用很少的缓存，就可以很大程度上解决CPU与主存的速度不均衡的问题。

5. 补充

上面的4.1——4.4节展示了计算机系统总线结构的发展历程。其实整个计算机系统是不断地向前发展，并且每个硬件的厂商都提出了各自不同的设计思路和方案，并且在此基础上演化出了各种经典的硬架构。但是，无论厂商如何设计与演化，最终的产品其实都遵循以上的设计思路；并且，他们各自的件出厂时会针对不同的操作系统提供驱动程序，屏蔽掉不同硬件架构之间的实现细节，但是以上特性归要设计专有的指令集。绝大多数情况下，我们没有必要对每种硬件的架构细节都了如指掌，只要了他们共同支持的特性即可，这些特性足够帮助我们写出性能卓越的程序了。