第一节 概述

一、输入输出系统的发展概况

1.早期是**分散连接**：每一个设备都有专门的控制电路，CPU 和 I/O 设备**串行**工作，采用程序查询方式。

2.接口模块和 DMA 阶段

总线连接，CPU 和 I/O 设备**并行**工作，信息交换方式为中断方式、DMA 方式。

3.具有通道结构的阶段

通道是一种简单处理器，能够执行通道程序，可以控制连接在通道上的 I/O 设备和主机直接进行信息传输。

4.具有 I/O 处理机的阶段

I/O 处理机可以是专用处理机，甚至可以采用和主机完全相同的处理器作为 I/O 处理器，在没有 I/O 设备的时候，该处理器可以作为主机处理器来进行处理数据。

随着 I/O 发展，数据的输入输出操作逐渐从 CPU 中分离出来，I/O 设备的独立性越来越强。

二、输入输出系统的组成

1.I/O 软件

分为 I/O 指令和通道指令两大类。

（1）I/O 指令：CPU 指令的一部分

指令格式：

操作码	命令码	设备码

操作码相当于一个标志，告诉 CPU 这是操作 I/O 的指令。命令码相当于操作码，指出了对 I/O 设备做什么操作，设备码给出的是 I/O 设备的地址或者其某个寄存器地址。

（2）通道指令：通道自身的指令

通常情况下，编程人员在编程中为了调用外部设备，应用程序需要添加广义 I/O 指令，指出数据传输的 I/O 设备，主存的首地址、传送字数、操作命令，操作系统根据广义 I/O 指令会编写一个通道指令，启动通道进行操作。

通道指令指出数据的首地址、传送字数和操作命令。一般来说这种指令都比较长，例如 IBM/370 的通道指令为 64 位。

2.I/O 硬件

设备　　I/O 接口

设备　　设备控制器　通道

三、I/O 设备与主机的联系方式

1.I/O 设备编址方式

（1）统一编址：就是把 I/O 设备的地址看作内存地址的一部分。如果所给出来的地址落入 I/O 设备的地址，那么这个操作就是对 I/O 设备进行操作。在这种情况下，CPU 可以直接利用取数、存数指令对 I/O 设备进行访问和控制，不需要其它单独指令。如果内存空间比较大，就可以采用这种方式。

（2）不统一编址：在内存地址空间之外专门设置一个地址空间。为了区分一条指令是对内存还是 I/O 操作，在单独编址的计算机中，有专门的 I/O 指令对 I/O 设备进行操作。

2.设备选址

用**设备选择电路**识别是否被选中。

3.传送方式

（1）串行

（2）并行

4.联络方式

（1）立即响应

一些结构简单，状态少的可以直接进行显示。

（2）异步工作采用应答信号

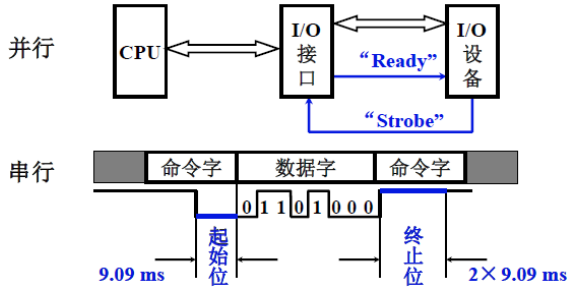

图 1.1 异步工作采用应答信号

先看并行传输。CPU 和接口之间一般是并行传输，异步工作主要是 I/O 接口和 I/O 设备进行传输的。为了进行传输，接口和设备之间有多条传输线组成的类似于总线的结构，数据并行的输入和输出，双方之间还要采用应答信号。比如说设备需要一些数据，接口准备好数据后，会发送一个“Ready”给设备，设备在接收到这个信号后，给出应答信号。

串行的方式格式如图 1.1 下所示，再下面给出了一个例子。起始位是一个 9.09ms 的低电平，然后给出数据字，最后给出终止位（2×9.09ms）。

（3）同步工作采用同步时标

必须有一个定宽定距的时标，通过时标来控制某一个时间点必须开始某个工作，在某个时间点必须结束某个工作。

5.I/O 设备与主机的连接方式

（1）辐射式连接（分散连接）

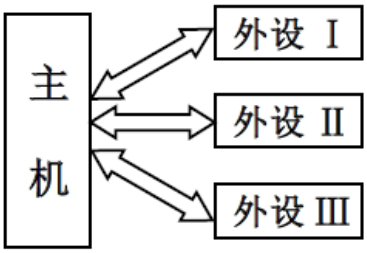

图 1.2 辐射式连接

每台设备都配有一套控制线路和一组信号线，增加一个设备就需要增加一套控制线路和

一组信号线。这种方式**不便于增删设备**。

（2）总线连接：**便于增删设备**。外部设备通过接口与主机连接，接口可以向外部设备传送主机命令，接口可以向主机传送外部设备的状态，接口还可以作为数据的缓存中转站，经过处理后传输给主机或者外部设备。

四、I/O 设备与主机信息传送的控制方式

1.程序查询方式

CPU 和 I/O 串行工作。踏步等待

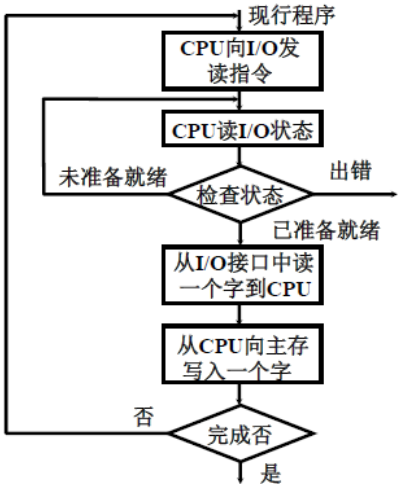

图 1.3 程序查询方式

如果要从外部设备读数据到内存，首先 CPU 向 I/O 发读指令，然后 CPU 读 I/O 接口中状态触发器的值，如果没有准备好，CPU 就一直去读，如果出错就出错处理，如果准备好了，那么从 I/O 接口中读一个字或字节到 CPU 的寄存器中，从 CPU 的寄存器向主存写入。最后判断这个任务是否完成，如果没有，则返回开头，如果完成了则结束。

图 1.4 里面是 CPU 检查 I/O 状态的局部放大图。一般来说，外部设备工作速度慢，CPU 速度快。外部设备接收到 CPU 信号后开始准备数据，准备数据需要较长时间，准备完成后才会把数据送到缓冲区，然后把状态置为 Ready，这个时候只有 CPU 检查到准备了好了才开始数据传输。CPU 先把数据读入寄存器中，再读入存储单元中。在外部存储器没有准备好的时候，CPU 一直原地等待，效率非常低。

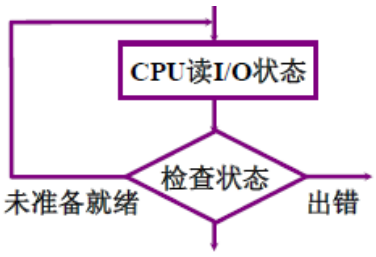

图 1.4 踏步等待

2.程序中断方式（部分并行）

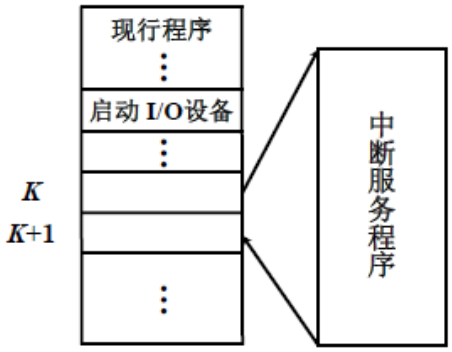

图 1.5 程序中断方式

CPU 在一定程度上被从 I/O 的输入输出中解放出来了。

I/O 工作第一步是自身准备，**CPU 不查询**，第二步是与主机交换信息，CPU 暂停现行程序。

如图 1.5 所示，CPU 执行主程序，在执行过程中遇到了一个 I/O 指令，CPU 启动 I/O 设备，I/O 设备开始准备，可能是输入也可能是输出，发出启动指令后，CPU 继续执行原来的程序。外部设备开始准备，当外部设备准备好后，向 CPU 提出一个中断请求，CPU 停止当前程序执行，转而和外部设备进行数据交换。具体过程如图 1.6 所示。

所谓部分并行是指在 I/O 准备数据的时候，CPU 还可以执行自己的程序。但是实际上 I/O 的数据交换还是 CPU 在进行，独立性仍然不强。

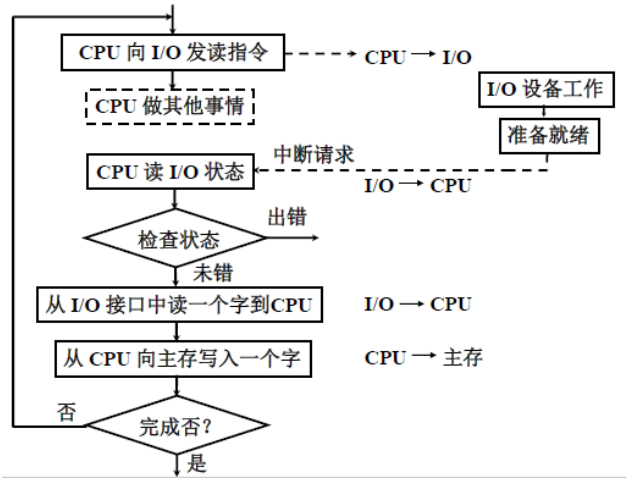

图 1.6 程序中断方式流程

3.DMA 方式

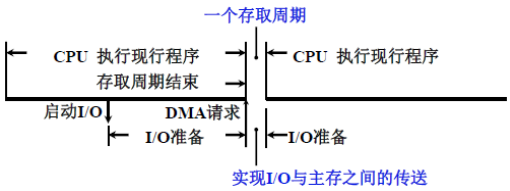

图 1.7 DMA 方式

主存和 I/O 之间有一条直接数据通道，不中断现行程序，采用方式是**周期挪用**（周期窃取），**CPU 和 I/O 并行工作**。如图 1.7，说明了 CPU 和 I/O 的并行性。在 CPU 正在执行现行程序的时候，碰到 I/O 的指令，CPU 发出指令启动 I/O，CPU 继续原来操作，I/O 在 DMA 控制器控制之下完成数据准备，数据准备好后，DMA 发出 DMA 请求，占用总线使用权，占用一个存取周期，开始外部设备和主存的交换，在这个存取周期，CPU 无法占用总线。这个

周期结束后，总线的使用权归还给 CPU。虽然在这个周期里面，CPU 不能使用总线，但是 CPU 还是可以执行其它操作。

4.三种方式的 CPU 工作效率比较

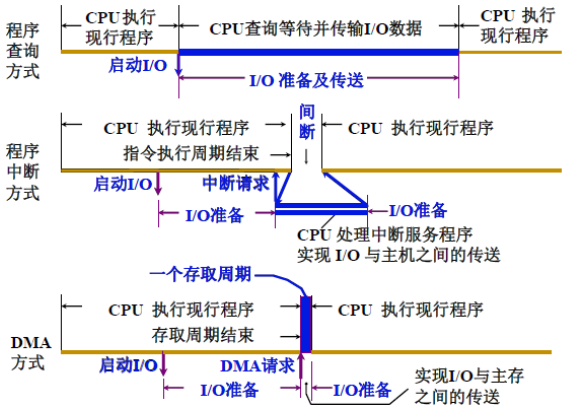

图 1.8 三种方式的效率比较

三种方式，CPU 的工作效率越来越高。第一种方式，CPU 大部分时间都在用来查询中；第二种方式，CPU 不需要查询，所以在 I/O 数据交换过程中只参与数据交换的过程；第三种方式，CPU 不参与 I/O 数据交换过程。

从程序查询方式到程序中断方式到 DMA 方式，I/O 系统的自治能力越来越强。

5.2 外部设备（非重点）

一、概述

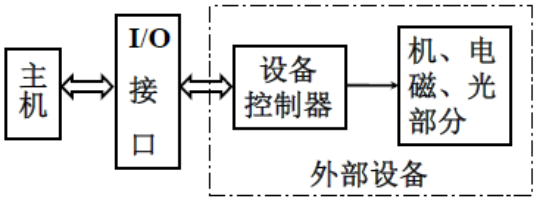

图 2.1 I/O 设备图

外部设备大致分为三类：

（1）人机交互设备：键盘、鼠标、打印机、显示器；

（2）计算机信息存储设备：磁盘、光盘、磁带；

（3）机-机通信设备：调制解调器等。

二、输入设备

1.键盘

2.鼠标：机械式；光电式。

3.触摸屏。

三、输出设备

1.显示器
（1）字符显示：字符发生器；
（2）图形显示：主观图像；
（3）图像显示：客观图像。
2.打印机
（1）击打式：点阵式（逐字、逐行）；
（2）非 ：激光（逐页）、喷墨（逐字）。

四、其它

1.A/D 模数、D/A 数模转换器；
2.终端：由键盘和显示器组成，完成显示控制与存储、键盘管理与通信控制；
3.汉字处理：汉字输入、汉字存储、汉字输出等。

五、多媒体设备

多种媒体技术和手段相结合进行综合应用，给人以更多视听上的享受。
关键技术包括数据压缩&解压缩、编码技术、专用芯片、语音图像识别、图片识别等。

5.3 I/O 接口

一、概述

我们这里的接口，既包括接口电路，也包括接口的控制器。

1.为什么要设置接口？
（1）实现设备的选择；
（2）实现数据缓冲达到速度匹配；
（3）实现数据串——并格式转换；
（4）实现电平转换；
（5）传送控制命令；
（6）反映设备的状态（“忙”、“就绪”、 “中断请求”）。

二、接口的功能和组成

1.总线连接方式的 I/O 接口电路
如图 3.1 所示，包括：
（1）设备选择线（单向线）；

（2）数据线（双向线）；
（3）命令线（单向线）；
（4）状态线（单向线）。

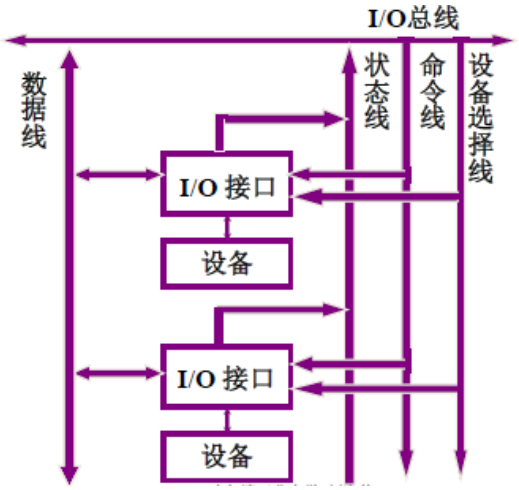

图 3.1 总线连接方式图

2.接口的功能和组成

功能	组成
选址功能	设备选择电路
传送命令的功能	命令寄存器、命令译码器
传送数据的功能	数据缓冲寄存器
反映设备状态的功能	设备状态标记

利用一系列标记来反映设备状态：

（1）完成触发器 D：用来标识设备是否准备好、数据是否准备好；

（2）工作触发器 B：用来标识外部设备工作状态，是否忙，若为 1，则表示忙，若为 0 则不忙；

（3）中断请求触发器 INTR：标识中断方式，设备准备好要向主机请求中断请求；

（4）屏蔽触发器：MASK：屏蔽触发器=1，表示尽管设备准备好了，仍然不能向主机申请中断。

3.I/O 接口的基本组成

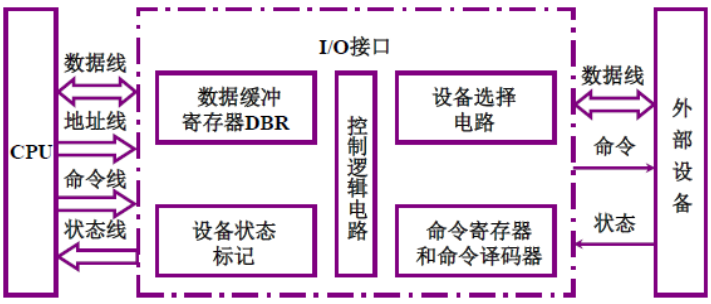

图 3.2 I/O 接口的基本组成

三、接口类型

1.按数据**传送方式**分类：

（1）并行接口：Intel 8255；
（2）串行接口：Intel 9251。
2.按**功能选择的灵活性**分类：
（1）可编程接口：Intel8255、Intel8251；
（2）不可编程接口：Intel8212。
3.按**通用性**分类：
（1）通用接口：Intel8255、Interl8251；
（2）专用接口：Interl9271、Intel8275。
4.按数据传送的控制方式分类：
（1）中断接口：Intel8259；
（2）DMA 接口：Intel8257。

5.4 程序查询方式

一、程序查询方式的流程

1.查询流程
（1）单个设备

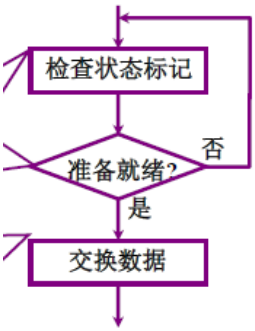

图 4.1 单个设备查询流程

具体过程在 5.1 节中已经讲过，在这里不再累述。这里使用三条指令：**测试指令**（检查状态标记）、**转移指令**（检查设备是否准备就绪）和**传送指令**（交换数据）。
（2）多个设备

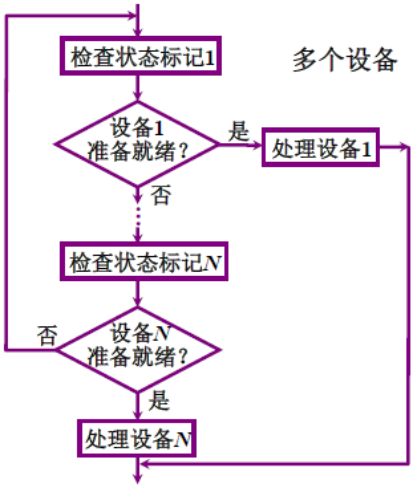

图 4.2 多个设备查询流程

按照优先级进行排序。优先级的最高最先接受检查，如果它已经准备好了，则处理设备，否则就继续往下检查。

2.程序流程

保存寄存器内容流程图如图 4.3 所示。要完成内存和外部设备数据输入输出，要借助 CPU 中某个寄存器来对数据进行暂存。如果这个数据是有用的，我们需要对这个寄存器的数据进行保存。我们可以把它写入某个内存单元、压入堆栈或者 CPU 放到其他闲置寄存器保存。

我们需要设置计数值，是为了限制 CPU 传输的数据量，内存和 I/O 到底传输多大数据，设置有两种方式：要传输 N 个字，则计数器存为 N，每次传输完一次减 1，另一种是设置为 -N，用补码表示，每次传输完加 1，直到计数器溢出（或 0），数据传输才会结束。

这之后，要设置主存缓冲区的首地址，启动外设，让外设准备，CPU 开始不断查询状态，若准备好了，开始进行数据传输，传送一个字，然后修改主存地址，修改计数值，若没有传送完成，则再启动外设，CPU 开始查询状态，然后继续往下走，若传送完了，就结束 I/O 传送。

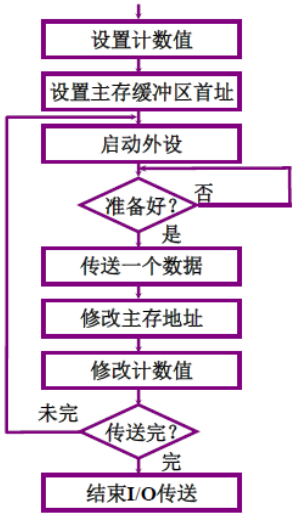

图 4.3 保存寄存器内容流程图

二、程序查询方式的接口电路

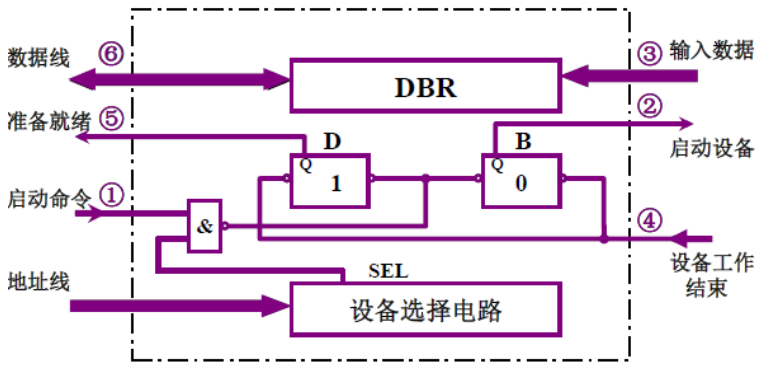

图 4.4 程序查询方式接口电路

DBR 是数据缓冲寄存器。

如图 4.4 所示，可以和图 3.2 来对比看看对应关系。图 4.4 所示最下面是设备选择电路，

它来确认这个设备是否参加传输的设备，这个信号是整个 I/O 接口电路的选择信号。如果 SEL 有效，启动命令有效，则这个电路则开始工作。

我们以数据输入（到内存中）来讲这个电路。CPU 通过地址线给出外部设备的地址，设备选择电路进行比较，如果相同则是启动被电路，SEL 和启动命令都有效，对两个状态标记进行复位或者置位，此时标记 D 为 0 表示尚未准备好，标记 B 置为 1 表示忙，开始启动设备。接收到 B=1 和启动命令后，数据传送到 DBR，这个设备工作结束，设备会通过设备状态线向接口电路送入设备工作结束的信号，然后修改 D=1，B=0，表示数据准备好了、设备工作完成。在这个时候及之前，CPU 一直在查询 D 的状态。当 D=1 了，CPU 通过数据线把数据进行读入，工作结束。

5.5 程序中断方式

一、中断的概念

学过，不再累述。

二、I/O 中断的产生

不再累述。

三、程序中断方式的接口电路

1.配置中断请求触发器和中断屏蔽触发器

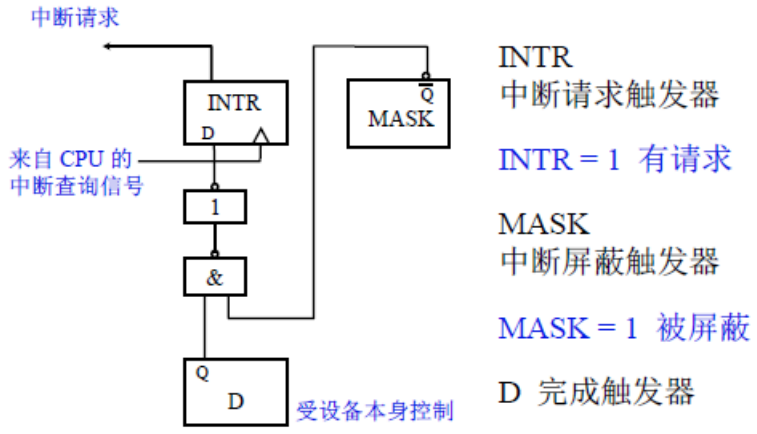

图 5.1 配置中断请求和中断屏蔽触发器

要进行数据传输的时候，设备准备好了，也就是 D=1 的时候，并且 MASK=0，也就是 Q’=1 的时候，INTR=1，在来自 CPU 的中断查询信号的作用下，INTR 会向 CPU 发出中断请求。若 MASK=1，那么这个请求就会被屏蔽。

2.排队器

若有多个中断同时向 CPU 请求，那么需要排队器对其按照优先级的大小进行排队。排队的方式有两种，一种是硬件，即在 CPU 内或在接口电路中（链式排队器）；一种是软件方式，采用查询的方法。

如图 5.2，每个接口的排队线路包含两个门：一个非，一个与非。这些电路相互连接，

构成链式排队器。在这个图里面，设备 1/2/3/4 优先级按降序排列。如果所有中断源都没有中断请求，INTR=1.如果有一个设备有请求，例如 INTR1=1，那么 INTR1’=0，那么经过一个与门之后输出 1，经过后面非门后 INTR2’=0，也就是说如果某一个优先级提出请求，它后面的都是 0，后面的都是 1。

如果 1 没有中断请求，2 有中断请求，那么 INTP2=1，INTR2’=0，那么与非门是高电平，INTR3’=0，同样后面也是 0。这里面可以看出，所有为 1 的最后一个 1 是优先级最高的。

硬件排队器对排队中，只有一个值为 1，这个表示了有中断请求的信号中优先级最高的，如图 5.3 所示。

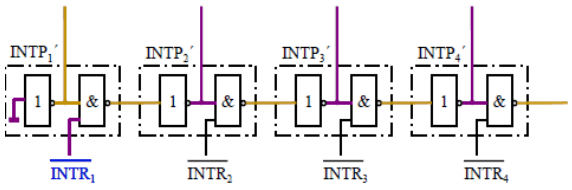

图 5.2 链式排队器

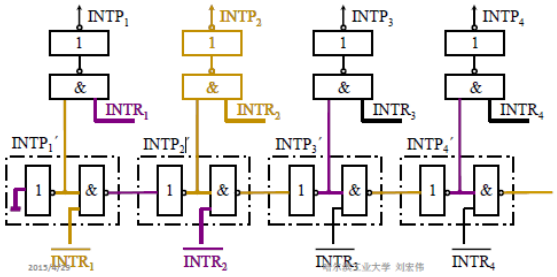

图 5.3 排队器 2

3.中断向量地址形成部件

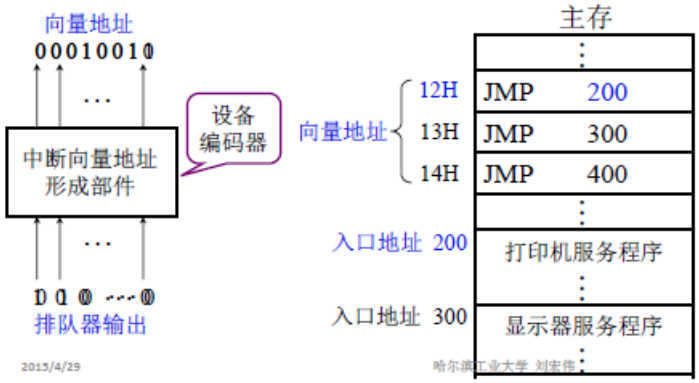

图 5.4 中断向量地址形成部件

如何找到中断服务的入口地址呢？有两种方法，首先是软件查表方式，还有一种是硬件向量法，即由**硬件**产生**向量地址**，再由**向量地址**找到**入口地址**。

要区分以下几个概念：

（1）中断号：假设支持 256 个中断，这 256 个中断被编号为 0 到 255，这就是中断号。

（2）中断向量：可以理解成中断服务程序的入口地址。比如说 X86 可以理解为中断服务程序的段地址和偏移量组成的向量，有时候也指程序状态字。比如说 CPU 中断，非体系寄存器或表示程序状态的寄存器指令无法读取，在计算机内部把它集成为一个字，称为程序状态字，这个时候中断向量就是和中断服务程序入口地址相关的段地址和偏移量，也包括执行中断服务程序的时候需要的一些状态信息。

（3）中断服务程序的入口地址：由中断向量地址来生成。向量地址指的是中断向量保存的内存单元的地址。

如图 5.4 左所示。中断向量地址形成部件的输入是来自排队器的的输出 INTP1/INTP2/…/INTPn，它的输出是中断向量（二进制表示），其位数与计算机可以处理中断源的个数有关，即一个中断源对应一个向量地址。本质上是一个**设备编码器**。

这里必须分清向量地址和纵横端服务程序的入口地址是两个不同的概念。图 5.4 右是通过向量地址寻找入口地址的一种方案，其中 12H、13H、14H 是向量地址，200、300 分别是打印机服务程序和显示器服务程序的入口地址。

向量地址是通过 CPU 中的系统总线的数据线送入 CPU 中的。

4.程序中断方式接口电路的基本组成

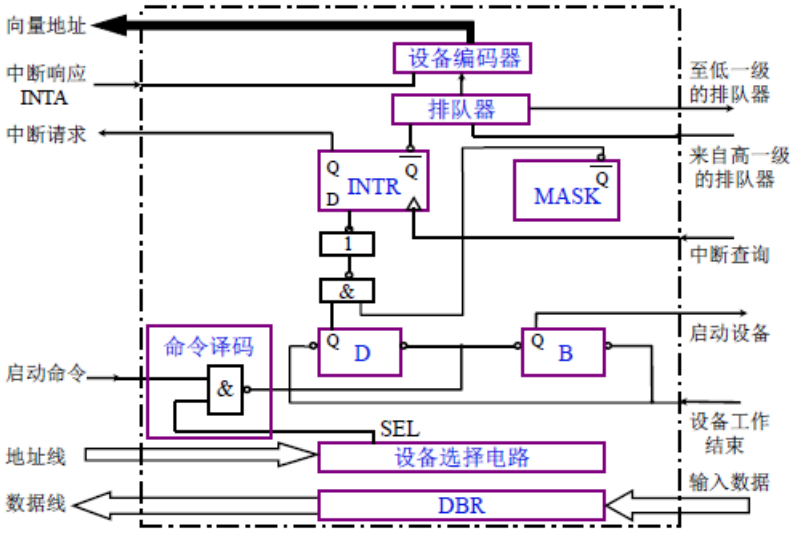

图 5.5 程序中断方式接口电路的基本组成

这是整个程序中断方式接口电路的基本组成。首先是地址线提供设备选择信号给设备选择电路，然后收到启动命令之后开始启动，对两个状态标记进行复位或置位，此时标记 D 为 0 表示尚未准备好，标记 B 置为 1 表示忙，开始启动设备。接收到 B=1 和启动命令后，向 CPU 发送中断请求，若此时 D=1 的时候，并且 MASK=0，也就是 Q'=1 的时候，INTR=1，在来自 CPU 的中断查询信号的作用下，INTR 会向 CPU 发出中断请求。若 MASK=1，那么这个请求就会被屏蔽。若已经排队完成，则会通过设备编码器寻找到向量地址，根据向量地址跳转到中断程序的入口地址。整个过程结束。

四、I/O 中断处理过程

1.CPU 响应中断的条件和时间

（1）条件

①允许中断触发器 **EINT=1**

②用**开中断**指令将 EINT 置“**1**”

③用**关中断**指令将 **EINT** 置“**0**”或**硬件自动复位**。

（2）时间

当 D=**1**（随机）且 MASK=**0** 时。并且在**每条指令执行阶段的结束前，CPU** 发**终端查询信号**（将 INTR 置“**1**”）。

2.I/O 中断处理过程——以输入为例

如图 5.5 所示，下面以输入设备为例，说明 I/O 中断处理的全过程。当 CPU 通过 I/O 指

令的地址码选中某设备后，则

①由 CPU 发启动 I/O 设备命令，将接口中的 B 置“1”，D 置“0”。

②接口启动输入设备开始工作。

③输入设备将数据送入数据缓冲寄存器。

④输入设备向接口发出“设备工作结束”信号，将 D 置“1”，B 置“0”，标志设备准备就绪。

⑤当设备准备就绪（D=1），且本设备未被屏蔽（MASK=0）时，在指令执行阶段的结束时刻，由 CPU 发出中断查询信号。

⑥设备中断请求触发器 INTR 置“1”，标志设备向 CPU 请求中断，与此同时 INTR 送至排队器，进行中断判优。

⑦若 CPU 允许中断（EINT=1），设备又被排队选中，即进入中断响应阶段，由中断响应信号 INTA 将排队器输出送至编码器形成向量地址。

⑧向量地址送至 PC，作为下一条指令的地址。

⑨由于向量地址中存放的是一条无条件转移指令，故这条指令结束后，无条件转至该设备的服务程序入口地址，开始执行中断服务程序，进入中断服务阶段，通过输入指令将数据缓冲寄存器的输入数据送至 CPU 的通用寄存器，再存入主存相关单元。

⑩中断服务程序的最后一条指令是中断返回指令，当其执行结束时，中断返回至原程序的断点处。整个过程结束。

五、中断服务程序流程

1.中断服务程序的流程

（1）保护现场

第一是保护程序的断点，断点包括中断返回后执行哪一条指令，即保存该指令的地址，还包括程序执行状态，由硬件完成，称为中断隐指令；第二是通用寄存器内容的保护，在中断服务程序中完成，利用进栈指令保存。

（2）中断服务：这是主体部分。对不同的 I/O 设备具有不同内容的设备服务。

（3）恢复现场：要求在推出服务程序前，将原程序中断时的“现场”恢复到原来寄存器中。通常可用取数指令（保存在存储器中）或出栈指令（保存在堆栈中）。

（4）中断返回：中断返回指令。

2.单重中断和多重中断

计算机在处理中断的过程中，有可能出现新的**更高级别**的中断请求，此时如果 CPU 暂停现行的中断服务程序去处理新的中断请求，这种现象称为**中断嵌套**，也称为**多重中断**。

如果 CPU 在执行中断服务程序时，对新的中断请求不予理睬，这种中断请求称为**单重中断**。

单重中断**不允许**中断**现行**的中断服务程序；

多重中断**允许**级别**更高**的中断源中断**现行**的中断服务程序。

3.单重中断和多重中断的服务程序流程

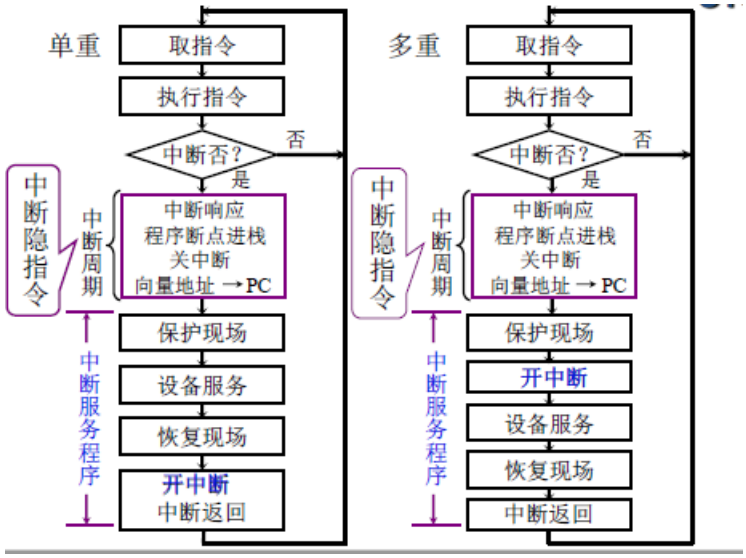

图 5.6 单重中断（左）和多重中断（右）的服务程序流程

比较图 5.6 的左和右，可以发现其区别在于“开中断”设置的时间不同。CPU 一旦响应了某中断源的中断请求后，便由迎检线路自动关中断，即中断允许触发器 EINT=0，以确保该中断服务程序的顺序执行。因此如果不用“开中断”指令将 EINT=0,1，则意味着 CPU 不能再响应其它任何一个中断源的中断请求。对于单重中断，**开中断指令设置在最后“中断返回”之前**，意味着在整个中断服务处理过程中，不能再响应其它中断源的请求。对于多重中断，**开中断指令提前至“保护现场”之后**，意味着在保护现场后，如有级别更高的中断源提出请求（**这是实现多重中断的必要条件**），CPU 也可以响应，即在此中断现行服务程序，转至新的中断服务程序。

4.主程序和服务程序抢占 CPU 示意图

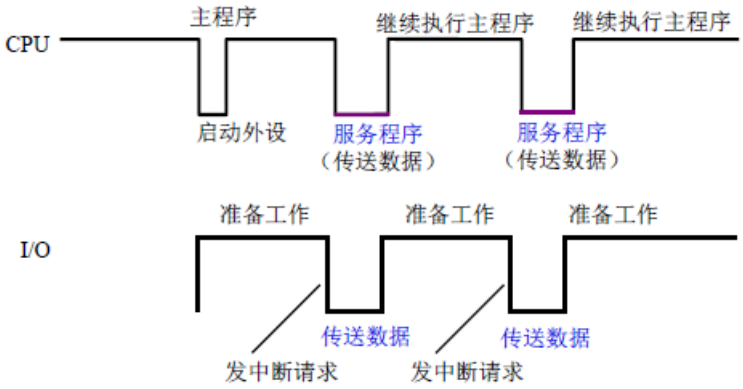

图 5.7 示意图

宏观上，CPU 和 I/O **并行**工作；**微观**上，CPU **中断现行程序**为 I/O 服务。

第六节 DMA 方式

一、DMA 方式的特点

DMA 是直接存储器访问的意思。

1.DMA 和程序中断两种方式的数据通路

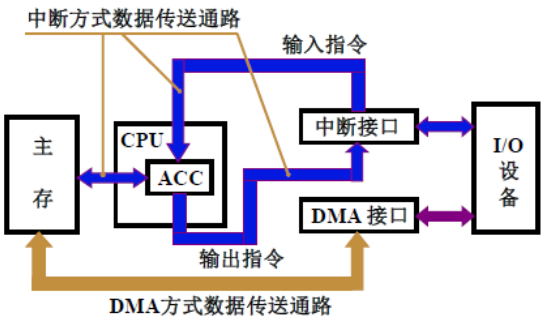

图 6.1 DMA 和程序中断两种方式的数据通路

由图 6.1 所示，由于主存和 DMA 接口之间有一条数据通路，因此主存和设备交换信息时，不通过 CPU，也不需要 CPU 暂停现行程序为设备服务，省去了保护现场和恢复现场，因此工作速度比程序中断方式的工作速度高。这一特点**特别适合于高速 I/O 或辅存与主存之间的信息交换**。因为高速 I/O 设备每次申请与主机交换信息时，都需要等待 CPU 做出中断响应后在进行，很可能因此丢失数据。

2.DMA 与主存交换数据的三种方式

在 DMA 方式中，由于 DMA 接口与 CPU 共享主存，这就有可能出现两者争用主存的冲突，为了有效分时使用主存，通常 DMA 与主存交换数据时采用如下三种方法：

（1）停止 CPU 访问主存。这种方式中，实际上只要外部设备要和内存数据交换，那么一块数据交换过程中，从第一个数据开始 CPU 就放弃了总线控制权，放弃了对内存的访问权，把这些权都交给了 DMA。**适合大量数据传输**。这种方式控制简单，但是 CPU 出于不工作状态或保持状态，未充分发挥 CPU 对主存的利用率。

（2）周期挪用（周期是指**访存周期**）。

DMA 访问主存有三种可能：

①CPU 此时不访问。那么就直接分配给 DMA 就可以了。

②CPU 正在访存。DMA 只能**等待**。

③CPU 与 DMA **同时请求访存**。此时 **CPU 将总线控制权让给 DMA**。

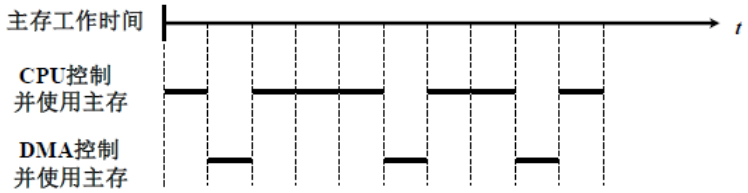

图 6.2 周期挪用示意图

如图 6.2 所示。数据准备好了，DMA 就会窃取一个或多个主存周期完成数据传输。在数据准备时候，CPU 可以控制和使用主存。提高了利用率。

（3）DMA 和 CPU 交替访问。实用性不强。

CPU 工作周期分为 C1 和 C2 两段时间，C1 专供 DMA 访存，C2 专供 CPU 访存。这种方法**不需要申请建立和归还**总线的使用权。如图 6.3 所示。

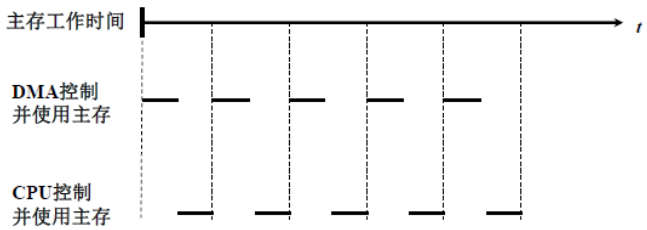

图 6.3 DMA 与 CPU 交替访问示意图

二、DMA 接口的功能和组成

1.DMA 接口功能

（1）向 CPU **申请** DMA 传送

（2）处理总线**控制权的转交**。

（3）**管理**系统总线、**控制**数据传送

（4）**确定**数据传送的**首地址和长度**，**修正**传送过程中的**数据地址和长度**。

（5）DMA 传送结束时，给出**操作完成信号**。

2.DMA 接口的组成

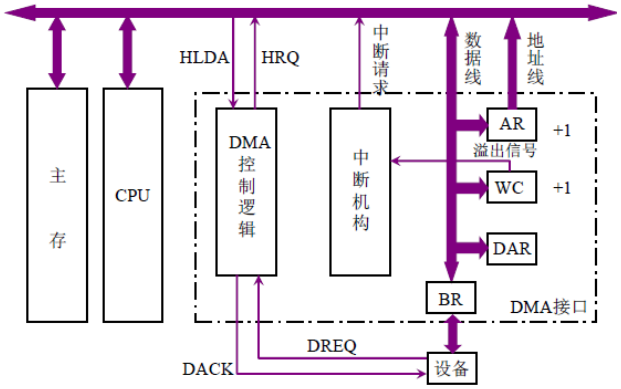

图 6.4 DMA 接口的组成

DMA 接口是一个黑盒子，根据功能可以分析其结构。要进行数据传输，CPU 要告诉 DMA 接口传输的内存地址是什么，所以需要一个地址寄存器（AR），需要知道数据量，所以需要一个计数器（WC），假设传入内存或者从内存输出，假设是从低地址开始，每次传输的数据单位和编址单位一致，每完成一个数据输入或输出，就要对地址进行修改，进行+1。WC 保存的是一个补码，每传输一次要+1，我们需要一个数据缓存器，即 BR。另外我们还需要一个设备地址寄存器（DAR），用来设备选择电路使用，还用来对硬盘进行访问时，保存硬盘的磁道号扇区号等。AR 把要访问的地址送给主存，通过数据线要给 AR 置值，设备地址也通过数据线。外部设备直接用 BR 相连。

输入输出需要 DMA 控制逻辑，用来内部协调，控制在给定时序发出给定信号。外部设备要向 DMA 控制逻辑 DREQ，DMA 控制逻辑给出应答 DACK 信号，DMA 还得通过总线向 CPU 发出总线请求信号 HRQ，CPU 回复 HLDA。DMA 接口电路还有一个中断机构，用于数据传输完成后对后续工作进行处理的。

具体各部分功能如下：

（1）主存地址寄存器（AR）。AR 用于存放主存中需要交换数据的地址。在 DMA 传送数据前，必须通过程序将数据在主存中的首地址送到主存地址寄存器。在 DMA 传送过程中，每交换一次数据，AR+1，直到一批数据传输完成。

（2）字计数器（WC）。WC 用于记录传送数据的总字数，通常以交换字数的补码值预置。在 DMA 传送过程中，每传送一个字，WC+1，直到计数器为 0 时，表示该批数据传送结束，于是 DMA 接口向 CPU 发送中断请求信号。

（3）数据缓冲寄存器（BR）。BR 用于暂存每次传送的数据。

（4）DMA 控制逻辑。DMA 控制逻辑负责管理 DMA 的传送过程，由控制电路、时序电路和命令状态控制寄存器组成。

（5）中断机构。当一批数据交换完毕，由“溢出信号”通过终端机构向 CPU 提出中断请

求，请求 CPU 作 DMA 操作的后续处理。

（6）设备地址寄存器（DAR）。DAR 存放 I/O 设备的设备码或表示设备信息存储区的寻址信息，如对硬盘进行访问时，保存硬盘的磁道号扇区号等。具体内容取决于设备的数据格式和地址的编址方式。

三、DMA 的工作过程

1.DMA 传送过程：预处理、数据传送、后处理

（1）预处理

通过几条输入输出指令预置如下信息：

①通知 DMA 控制逻辑传送方向（入/出）；

②设备地址→DMA 的 DAR；

③主存地址→DMA 的 AR；

④传送字数→DMA 的 WC。

（2）DMA 传送过程示意

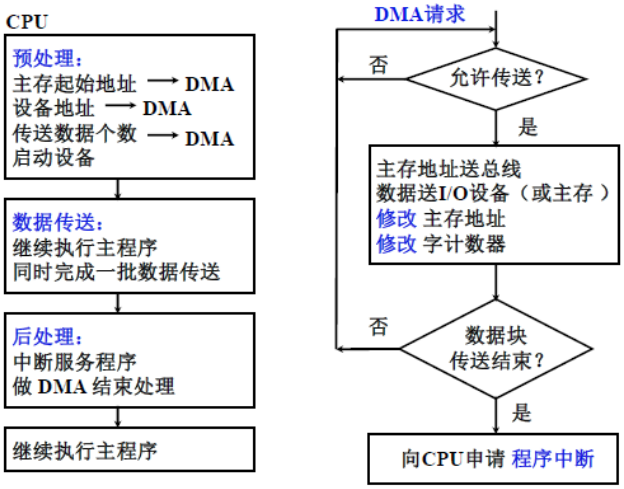

图 6.5 数据传送流程

图 6.5 左侧是 DMA 传送过程，右侧是数据传送阶段的细化。结合图 6.4 可以描述一下数据输入传送过程：

①当设备准备好一个字时，发出选通信号，将该字读到 DMA 的 BR 中，表示 BR“满”。

②与此同时，设备向 DMA 接口发请求（DREQ）。

③DMA 接口向 CPU 申请总线控制权（HRQ）。

④CPU 发回 HLDA 信号，表示允许将总线控制权交给 DMA 接口。

⑤将 DMA 主存地址寄存器中的主存地址送地址总线，并命令存储器写。

⑥通知设备已被授予一个 DMA 周期（DACK），并为交换下一个字做准备。

⑦将 DMA 数据缓冲寄存器的内容送数据总线。

⑧主存将数据总线上的信息写至地址总线指定的存储单元中。

⑨修改主存地址和字计数值。

⑩判断数据块是否传送结束，若未结束，继续传送；若已结束（WC 溢出），则向 CPU 申请程序中断，标志数据块传送结束。

结合图 6.4 可以描述一下数据输出传送过程，如图 6.6 所示：

①当 DMA 数据缓冲寄存器已将输出数据送至 I/O 设备后，表示数据缓冲寄存器已“空”。

②设备向 DMA 接口发请求（DREQ）。

③DMA 接口向 CPU 申请总线控制权（HRQ）。

④CPU 发回 HLDA 信号，表示允许将总线控制权交给 DMA 接口。

⑤将 DMA 主存地址寄存器中的主存地址送地址总线，并命令存储器**读**。

⑥通知设备已被授予一个 DMA 周期（DACK），并为交换下一个字做准备。

⑦主存将相应地址单元的内容通过数据总线读入到 DMA 的数据缓冲寄存器中。

⑧将 DMA 数据缓冲寄存器的内容送到输出设备，若为字符设备，则需将其拆成字符输出。

⑨修改主存地址和字计数值。

⑩判断数据块是否传送结束，若未结束，继续传送；若已结束，则向 CPU 申请程序中断，标志数据块传送结束。

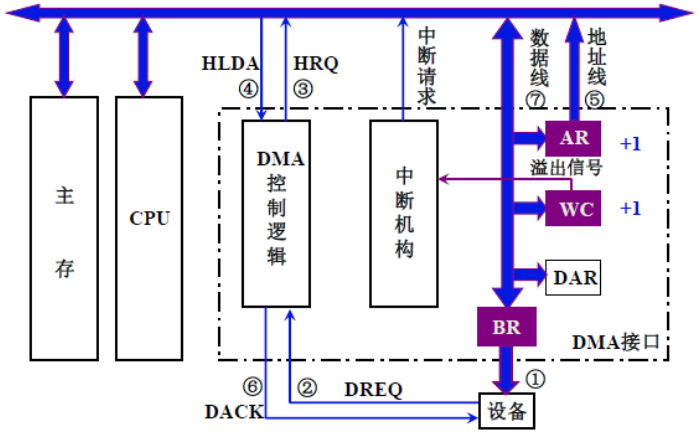

图 6.6 输出过程演示图

（3）后处理——**中断服务程序完**成

当 DMA 的中断请求得到响应，后，CPU 停止源程序的执行，转去执行**中断服务程序**，做一些 DMA 的结束工作，包括：

①校验送入主存的数据是否正确；

②决定是否继续使用 DMA 传送其它数据块，若继续传送，则又要对 DMA 进行初始化，若不需要传送，则停止外设；

③测试在传送过程中是否发生错误，若出错则转错误诊断及处理错误程序。

2.DMA 接口与系统的连接方式

（1）具有公共请求线的 DMA 请求

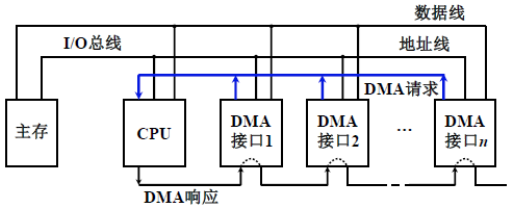

图 6.7 公共请求线的 DMA 请求

（2）独立的 DMA 请求

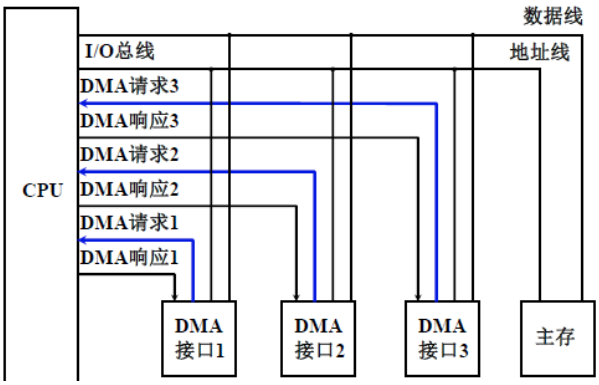

图 6.8 独立的 DMA 请求

排队工作是在 CPU 内部完成。

3.DMA 方式与程序中断方式的比较

	中断方式	DMA 方式
数据传送	程序	硬件
响应时间	指令执行结束	存取周期结束
处理异常情况	能	不能
中断请求	传送数据	后处理
优先级	低	高

四、DMA 接口类型

1.选择型：在**物理**上连接**多个**设备，在**逻辑**上**只允许连接一个**设备。如图 6.9 所示。

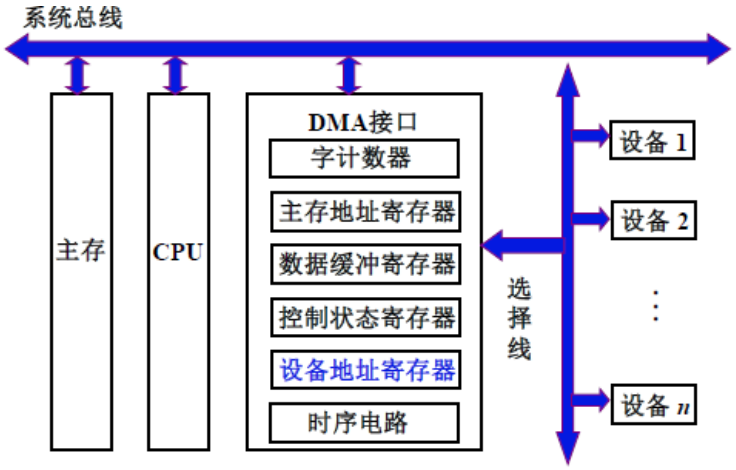

图 6.9 选择型接口

2.多路型：在**物理上**连接**多个**设备，在**逻辑上**允许连接**多个**设备同时工作（数据准备的时候），在**真正数据传输**的时候，只能**一个**设备工作。

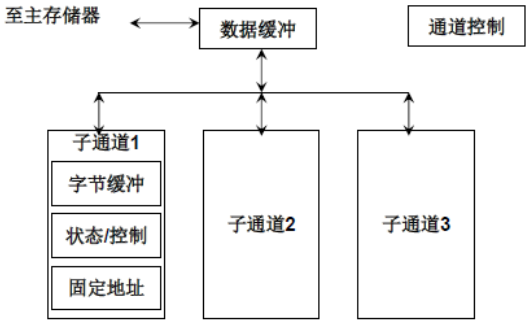

图 6.10 多路型接口

假设磁盘每隔 30 微秒提出数据请求，这 30 微秒包括数据传送和数据准备时间，磁带每隔 45 微秒提出一次 DMA 请求，打印机隔 150 微秒提出 DMA 请求，并假设真正数据传输是 5 微秒。一个多路型接口连接了这 3 个设备。

由于打印机是最早提出请求的，那么首先响应打印机请求，把 5 微秒时间给打印机数据传输，这个 5 秒结束后，磁盘和磁带同时发出请求，速度越高的设备优先级越高，所以优先响应磁盘的操作，用 5 微秒进行数据传输，响应完磁盘后响应磁带的请求。以此类推下去，这就是整个过程。

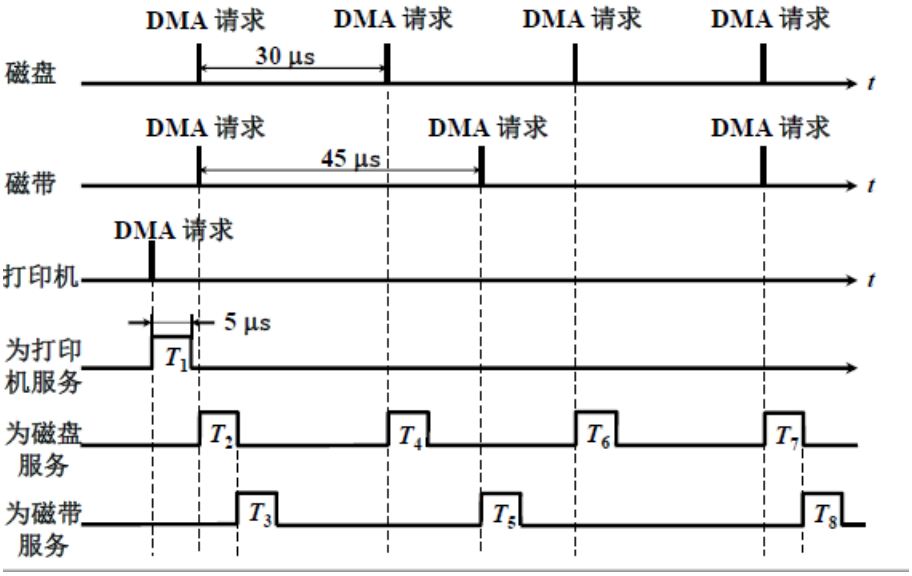

图 6.11 多路型 DMA 接口的工作原理