南京理工大学

NANJING UNIVERSITY OF SCIENCE & TECHNOLOGY

# 第 6 章 总线与I/O系统组织

主讲：张功萱

# 直接存储器访问方式

## 第6.8节

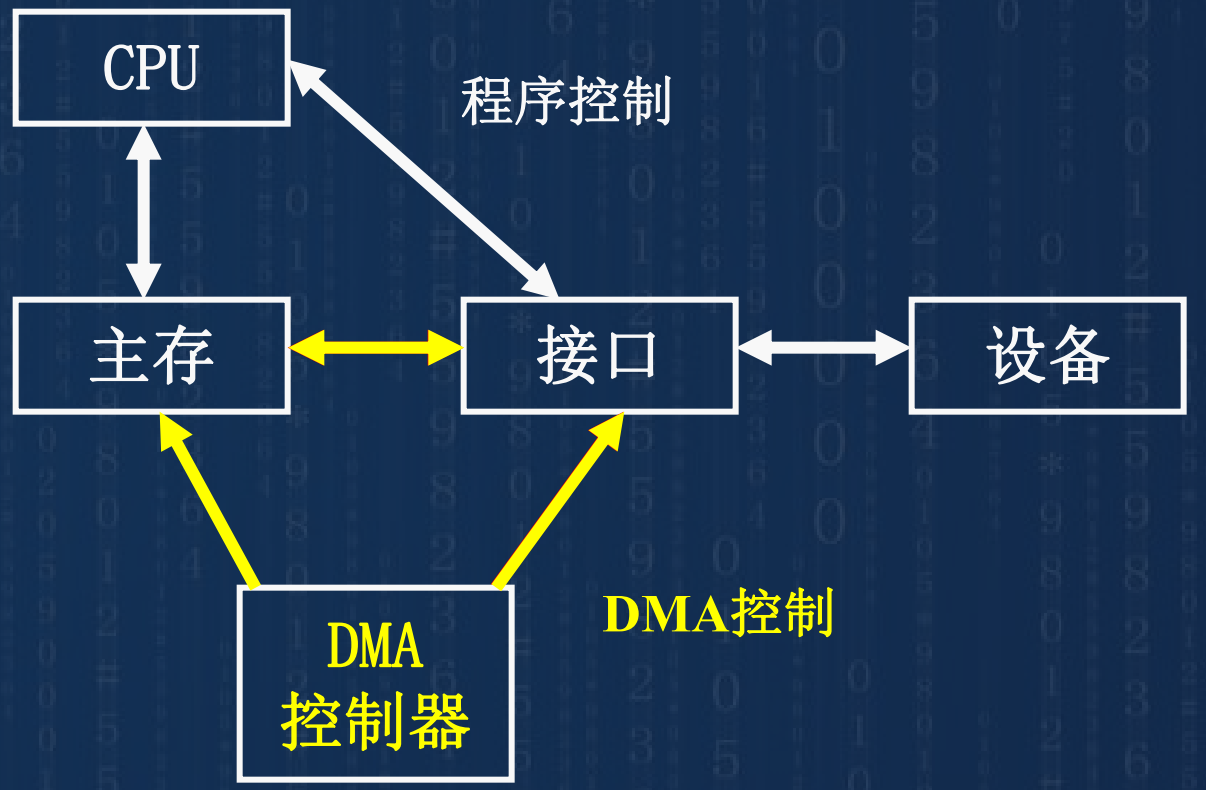
1. 为什么要引入DMA?
2. DMA的传送方式有哪些?
3. DMA有什么部件组成?
4. DMA有哪几个工作过程?
5. DMA与INT的区别是什么?

# 为什么要引入DMA方式？

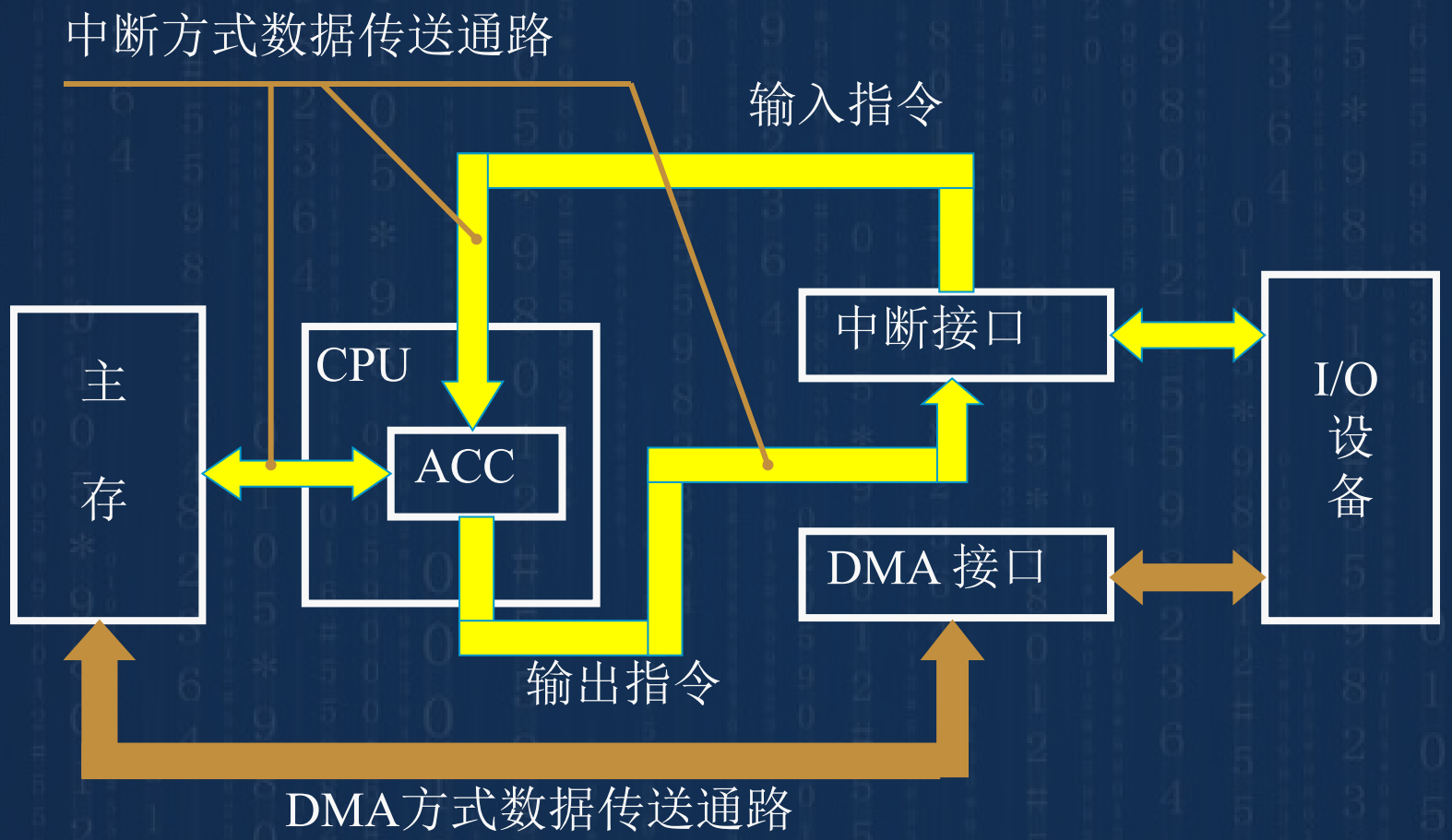
- 程序直接控制方式受“踏步”现象的限制，效率低下，不适合高速设备和主机间的数据传送。
- 中断控制方式虽比程序直接控制方式有效，CPU和外设有一定的并行度，但由于下列原因也不适合高速设备和主机间的数据传送。
  - 对I/O请求响应慢。每传送一个数据都要等待外设的中断请求，并增加许多中断响应和中断处理前、后的附加开销（保护断点、现场等），不能及时响应I/O请求。
  - 数据传送速度慢。数据传送由软件完成（由CPU执行相应的中断服务程序来完成），速度慢 。

- **DMA方式：**
- 以主存为中心，采用硬件手段在主存与I/O设备之间建立直接的数据传送通路，由DMA控制器（DMAC）取得总线控制权，控制主存与I/O设备之间的数据传送，在传送过程中不需要CPU的程序干预的数据传送控制方式。

- DMA方式主要用于高速外设按照连续地址直接访问存储器。



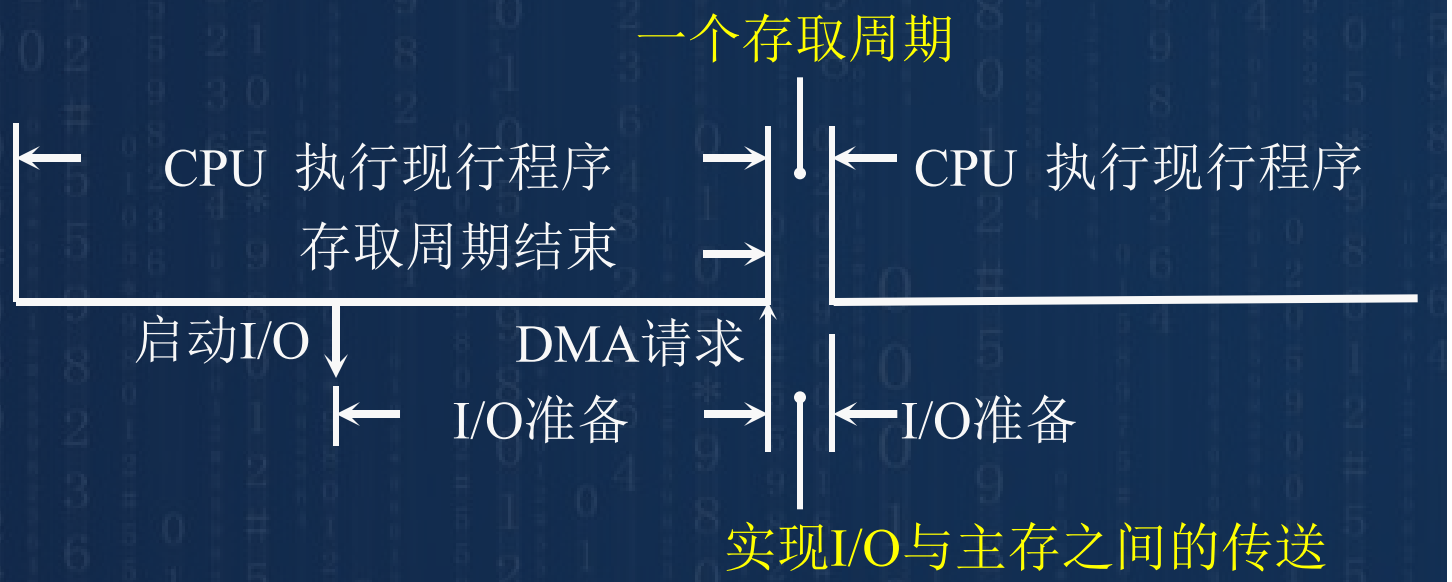
# DMA 和程序中断两种方式的数据通路



不中断现行程序

通过**周期挪用**等不同方式传送数据

**CPU 和 I/O 并行工作**



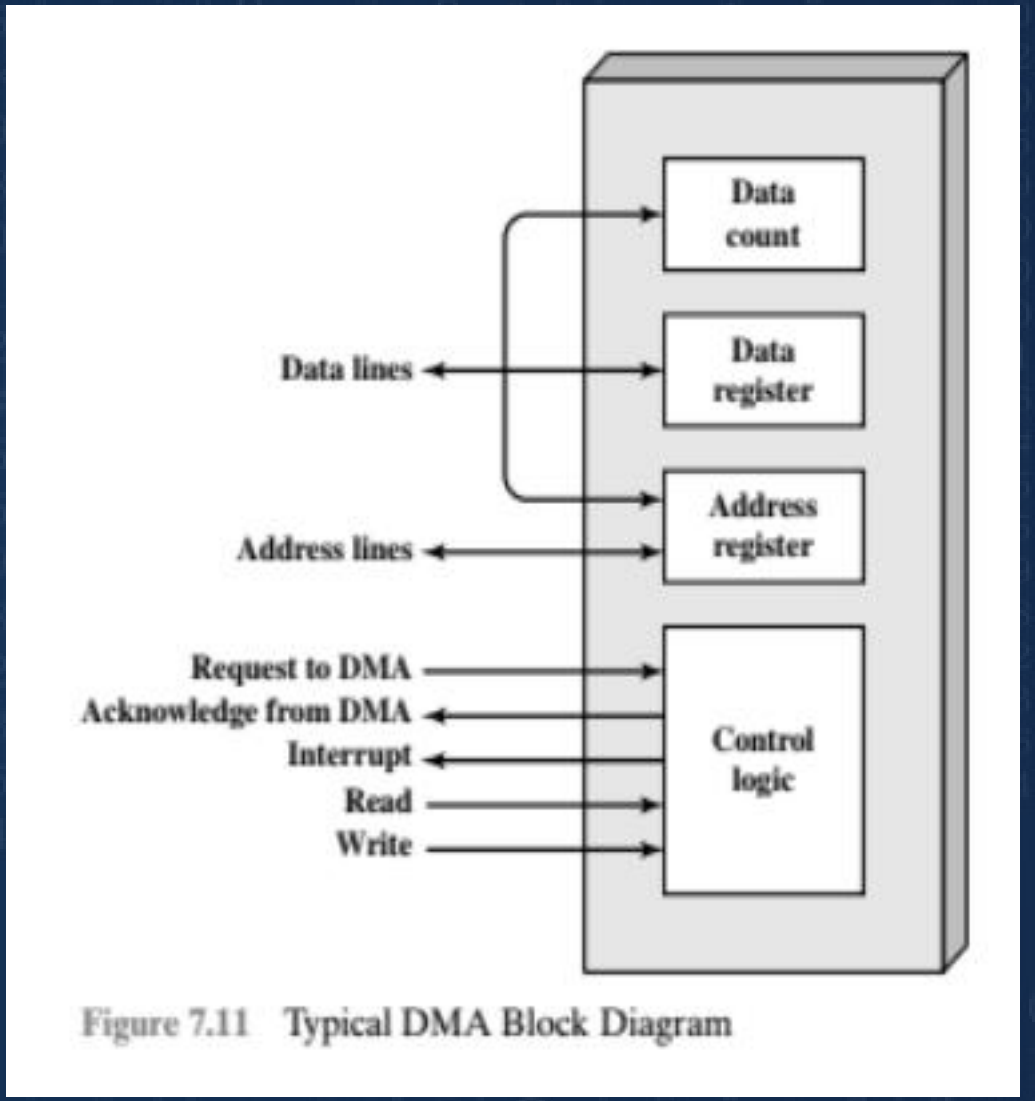




Figure 7.11 Typical DMA Block Diagram

# 6.8.1 DMA方式的特点与应用场合

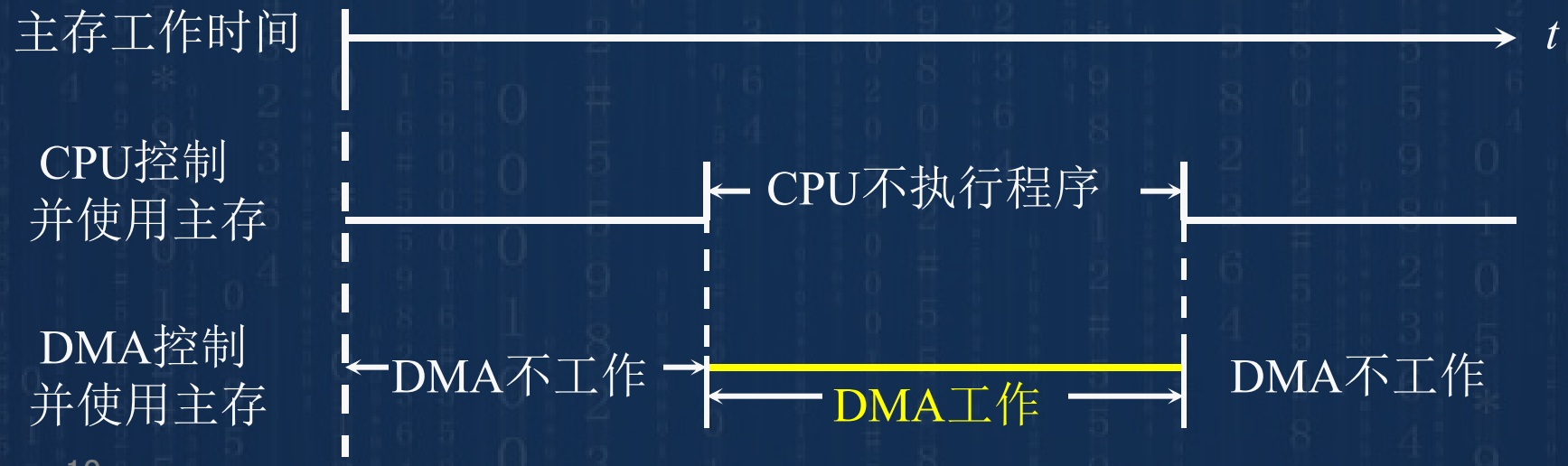
- **1. DMA方式的特点**
- (1) 以响应随机请求的方式，实现主存与**I/O**设备间的快速数据传送。
- (2) 采用**DMA**方式控制数据传送时，仅需占用系统总线，不切换程序，不存在保存断点、保护现场、恢复现场、恢复断点等操作。因此**DMA**传送的插入不影响**CPU**的程序执行状态，除了访问主存的冲突外，**CPU**可以继续执行自己的程序，提高了**CPU**的利用率。
- (3) **DMA**方式只能处理简单的数据传送，难以识别与处理复杂的情况。

- **2. DMA方式的应用**
- **DMA方式一般应用于主存与高速I/O设备间的简单数据传送(高速I/O设备如磁盘、磁带、光盘等外存储器)，以及其它带有局部存储器的外围设备、通信设备等。如：**
- (1) 磁盘与主存的成块数据传送
- (2) 通信设备的批量数据传送
- (3) 动态存储器的刷新
- (4) 大批量数据采集系统

- **DMA**传送是直接依靠硬件实现的，可用于快速的数据直传。但**DMA**方式本身不能处理复杂事态。因此，在某些场合常综合应用**DMA**方式与程序中断方式，二者互为补充。
- 典型的例子是磁盘调用，磁盘读写采用**DMA**方式进行数据传送，而对寻道是否正确的判别处理、批量传送结束后的善后处理，则采用程序中断方式。

# 6.8.2 DMA的传送方式

- 1. CPU停机方式
- 用CPU停机方式实现DMA传送时，CPU停止工作，让出对总线的控制权，而由DMAC接管总线，进行数据传送。数据传送结束后，再将总线交还给CPU。



# CPU停机方式的特点

- 优点：
- 控制简单，比较容易实现，是最常用、最简单的一种DMA实现方式，大部分DMAC都采用这种方式。
- 缺点：
- 由于在采用这种方式进行的DMA传送期间，使CPU处于空闲等待状态，降低了CPU的利用率，并且可能会影响到某些实时性很强的操作，如中断响应和对动态RAM的刷新等。

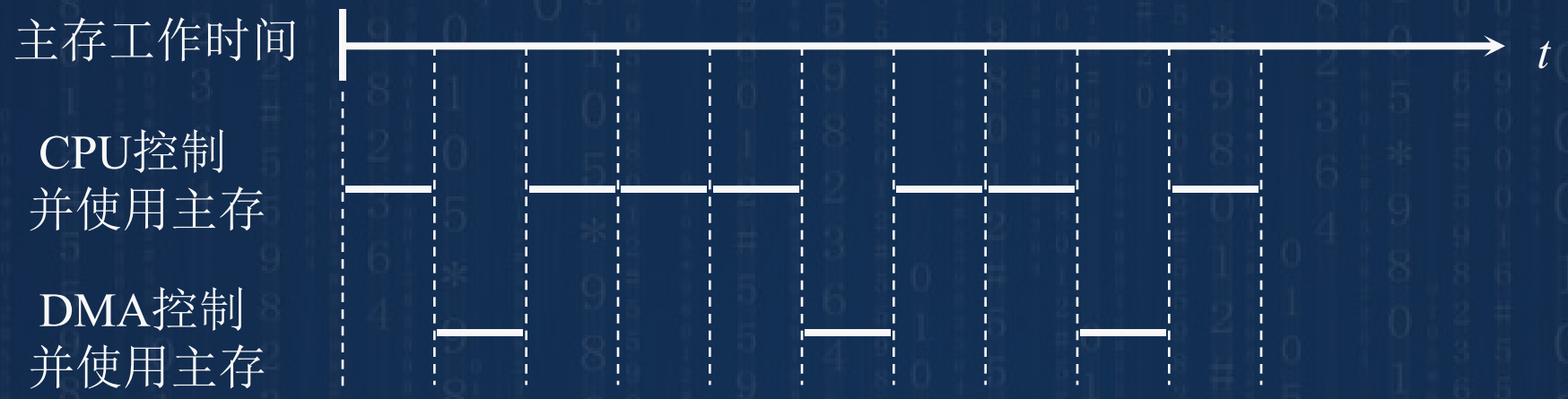
# CPU停机方式的应用

- 采用这种工作方式的I/O设备，在其接口中一般设置有存取速度较快的小容量存储器。I/O设备与小容量存储器先交换数据，然后小容量存储器再与主机交换数据，这样可减少DMA传送占用存储总线的时间，也即减少CPU暂停工作时间。这种方式也称为成组连续传送方式，它可减少系统总线控制权的交换次数，有利于提高输入输出速度。（适用于高速外设或单用户状态下的个人计算机。）

# 2. 周期挪用(周期窃取)方式

- 当I/O设备无DMA传送请求时，CPU正常访问主存。当I/O设备需要使用总线传送数据时，产生DMA请求，DMAC把总线请求发给CPU。
- ① 若CPU本身无使用总线的要求，CPU就把总线交给DMAC，由DMAC控制I/O设备使用总线
- ② 如果此时CPU也要使用总线，则CPU自身进入一个或几个“空闲总线周期”状态，即CPU让出一个或几个总线周期给DMAC（也称DMAC“挪用”一个总线周期），DMAC利用此总线周期控制传送一个数据字后，再把总线交还给CPU，以便CPU可以继续执行总线操作。

# 周期挪用




- 采用周期挪用方式时，外设要求DMA传送的三种情况：
- (1) 外设要求DMA传送时，CPU不需访问主存(如CPU正在执行乘法指令，由于乘法指令执行时间较长，此时CPU不需访问主存)，故外设访存与CPU不发生冲突。

- ⑵ 外设要求DMA传送时，CPU正在访存，此时必须等CPU存取周期结束后，CPU才能让出总线控制权。
- ⑶ 外设要求访存时，CPU也要求访存，这就出现了访存冲突。此时要求外设访存优先于CPU访存。因为外设不立即访存就可能丢失数据，这时DMAC要窃取一、二个存取周期，使CPU延缓一、二个存取周期再访存。
- 周期挪用方式的优点：
- 与CPU暂停访存的方式相比，周期挪用方式既实现了I/O传送，又较好地发挥了主存与CPU的效率，是一种广泛采用的方法。

# 周期挪用方式的缺点

- 每传送一个数据，DMA都要产生访问请求，待到CPU响应后才能传送，因此判优操作及总线切换操作非常频繁，其花费的时间开销较大。往往在传输一个数据块时，需要DMA控制器多次申请使用总线，这影响了DMA的数据传输速度。

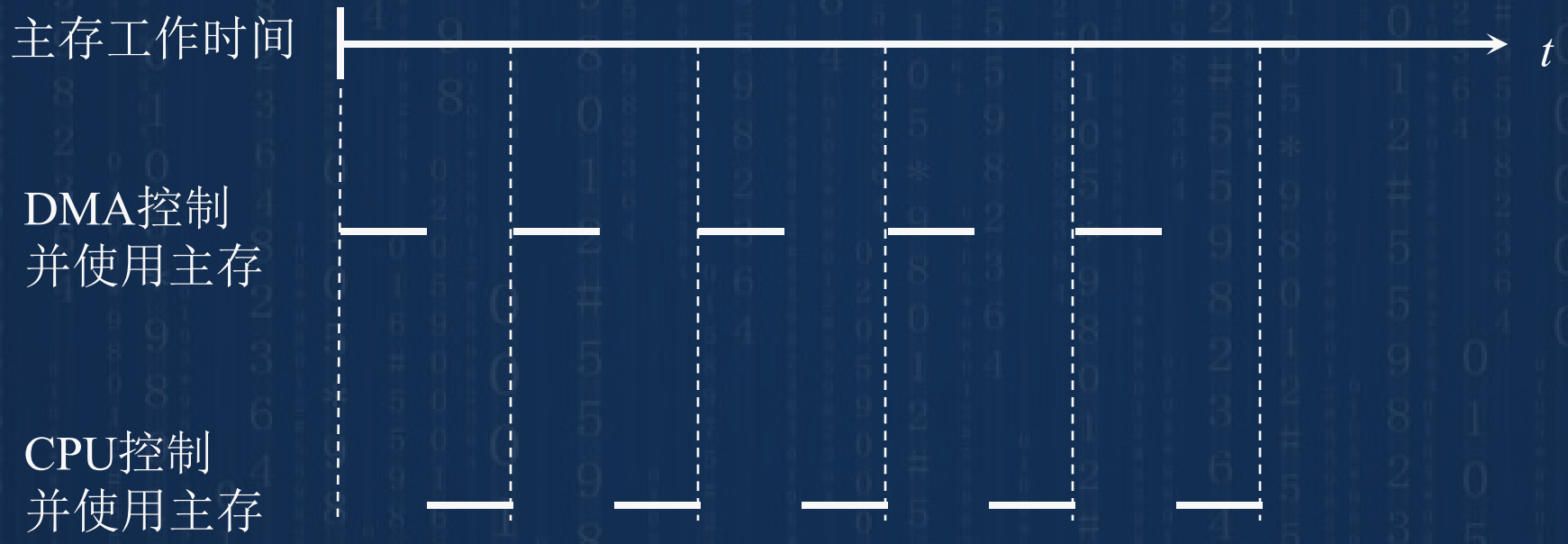
- 周期挪用适用于I/O设备接口控制器中数据缓冲器容量不大的场合，例如在接口控制器中仅设置一个数据寄存器的情形，这种方式也称为单字传送方式。
- 对具有较大容量数据缓冲存储器的高速外设来说是不合适的。

# 3. 交替访问内存方式

- 将一个CPU周期分为两个分周期，与DMA分别使用。其中一个专供DMA访存，另一个专供CPU访存。
- 这种方式不需要总线使用权的申请建立和归还过程，总线使用权是通过不同的周期分别控制的。
- 在这种工作方式下，CPU既不停止主程序的运行也不进入等待状态，在CPU不知不觉中完成了DMA的数据传送，故又有“透明的DMA”方式之称，当然周期扩展方式会使CPU的处理速度减慢，其相应的硬件逻辑也变得更为复杂。

CPU 工作周期 { $C_1$ 专供 DMA 访存
$C_2$ 专供 CPU 访存

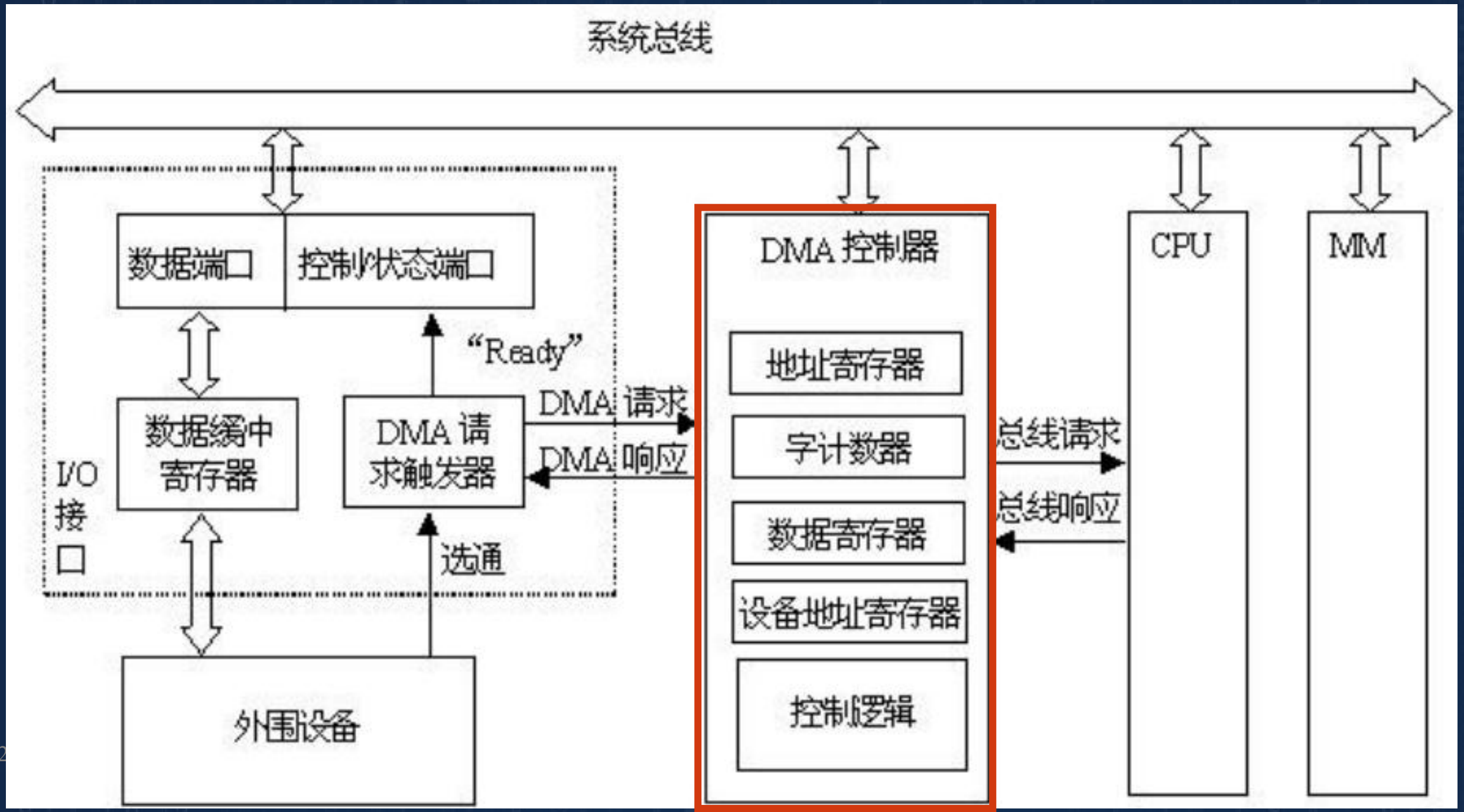
所有指令执行过程中的一个基准时间




不需要 申请建立和归还 总线的使用权

# 6.8.3 DMA的硬件组织

- 在目前的计算机系统中，通常专门设置了DMA控制器，并且较多采取DMA控制器与DMA接口相分离的方式。
- 1. DMA控制器（DMAC）
- DMAC负责申请、接管总线的控制权、发送地址和操作命令以及控制DMA传送过程的起始与终止。
- DMA控制器独立于具体I/O设备，可以为各个设备通用。


系统总线
数据端口
控制/状态端口
“Ready”
数据缓冲寄存器
DMA 请求触发器
DMA 请求
DMA 响应
I/O 接口
选通
外围设备
DMA 控制器
地址寄存器
字计数器
数据寄存器
设备地址寄存器
控制逻辑
总线请求
总线响应
CPU
MM

# DMAC的功能

- ① 接收外设的DMA请求，向CPU发出总线请求信号。请求CPU让出总线。
- ② 当CPU发出DMA响应信号之后，接管对总线的控制，进入DMA方式。
- ③ 对存储器寻址，输出和修改地址信息。
- ④ 向存储器和外设发出相应的读/写控制信号。
- ⑤ 控制传送的字节数，判断DMA传送是否结束。
- ⑥ 在DMA传送结束以后，向CPU发出结束DMA请求信号，释放总线，使CPU恢复对总线的控制，继续正常工作。

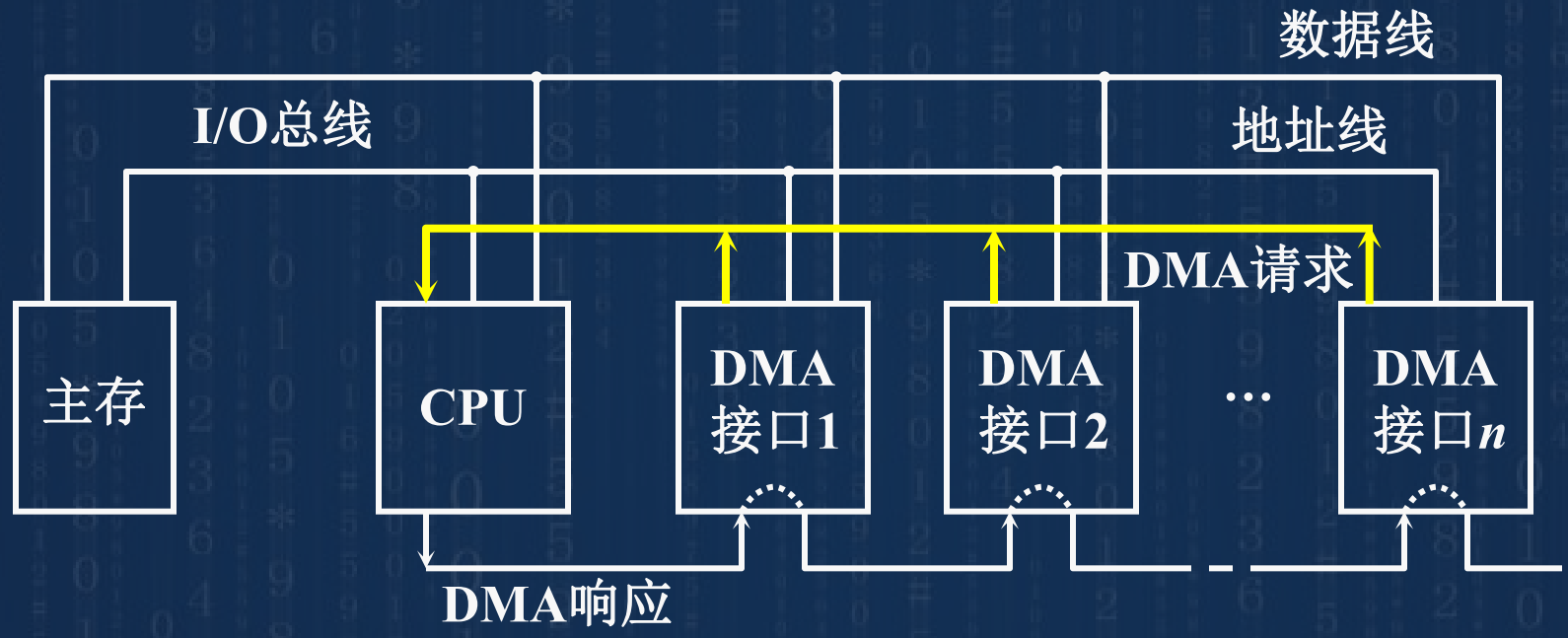
- **2. DMA接口**
- 用于实现与设备的连接和数据缓冲，反映设备的特定要求。

# DMA接口的功能

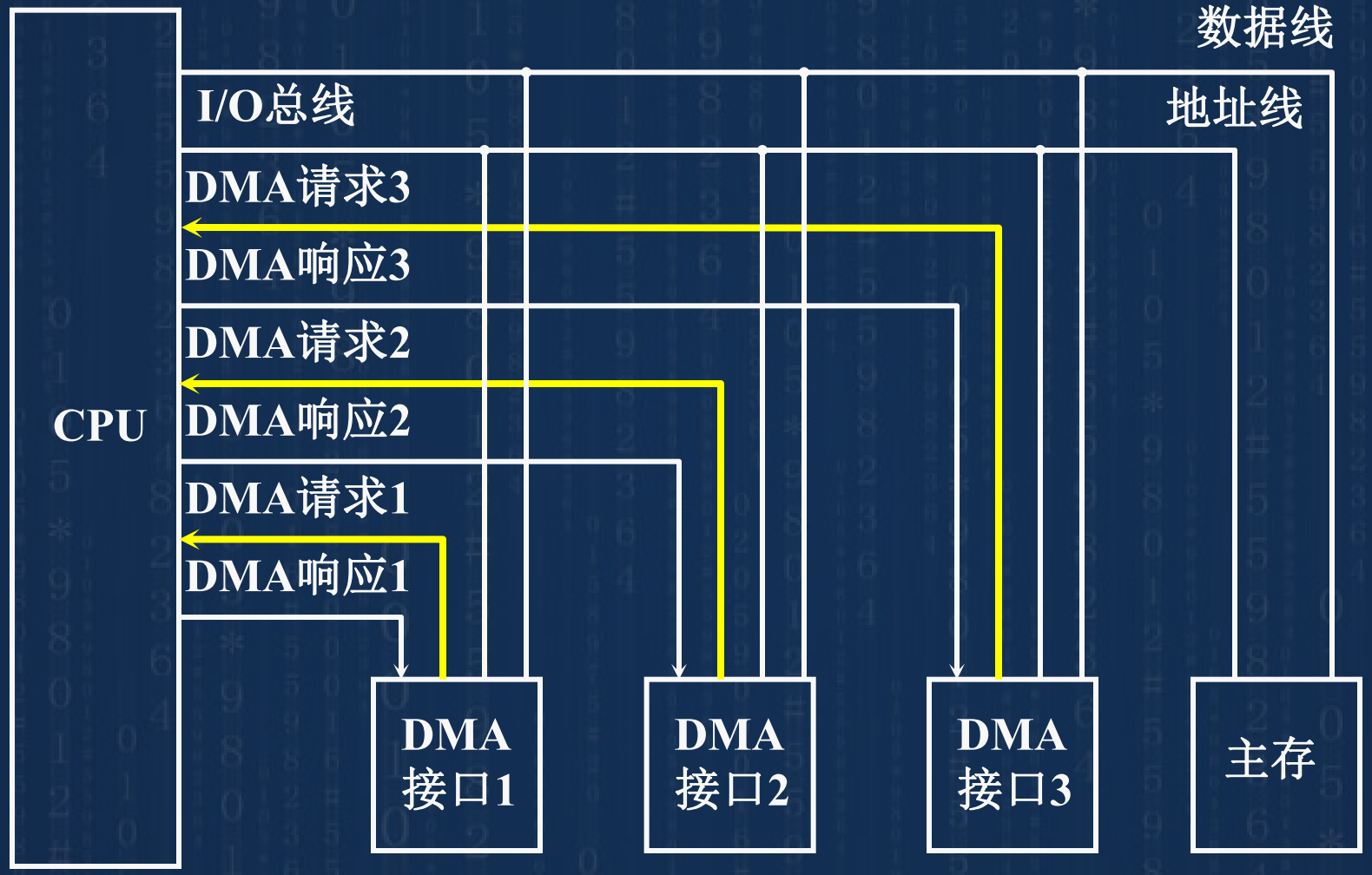
- ① 根据寻址信息访问I/O设备
- ② 将数据从设备读入数据缓冲寄存器，或将数据缓冲寄存器中的数据写入设备。
- ③ 在需要进行DMA传送时，向DMAC提出DMA请求。获得批准后，接口将数据缓冲寄存器中存放的数据经数据总线写入主存单元或将主存单元存放的内容写入接口中的数据缓冲寄存器。
- DMA接口中一般包含数据缓冲寄存器、I/O设备寻址部件、DMA请求逻辑等。

# 3. DMA接口与系统的连接方式

- (1) 具有公共请求线的 DMA 请求



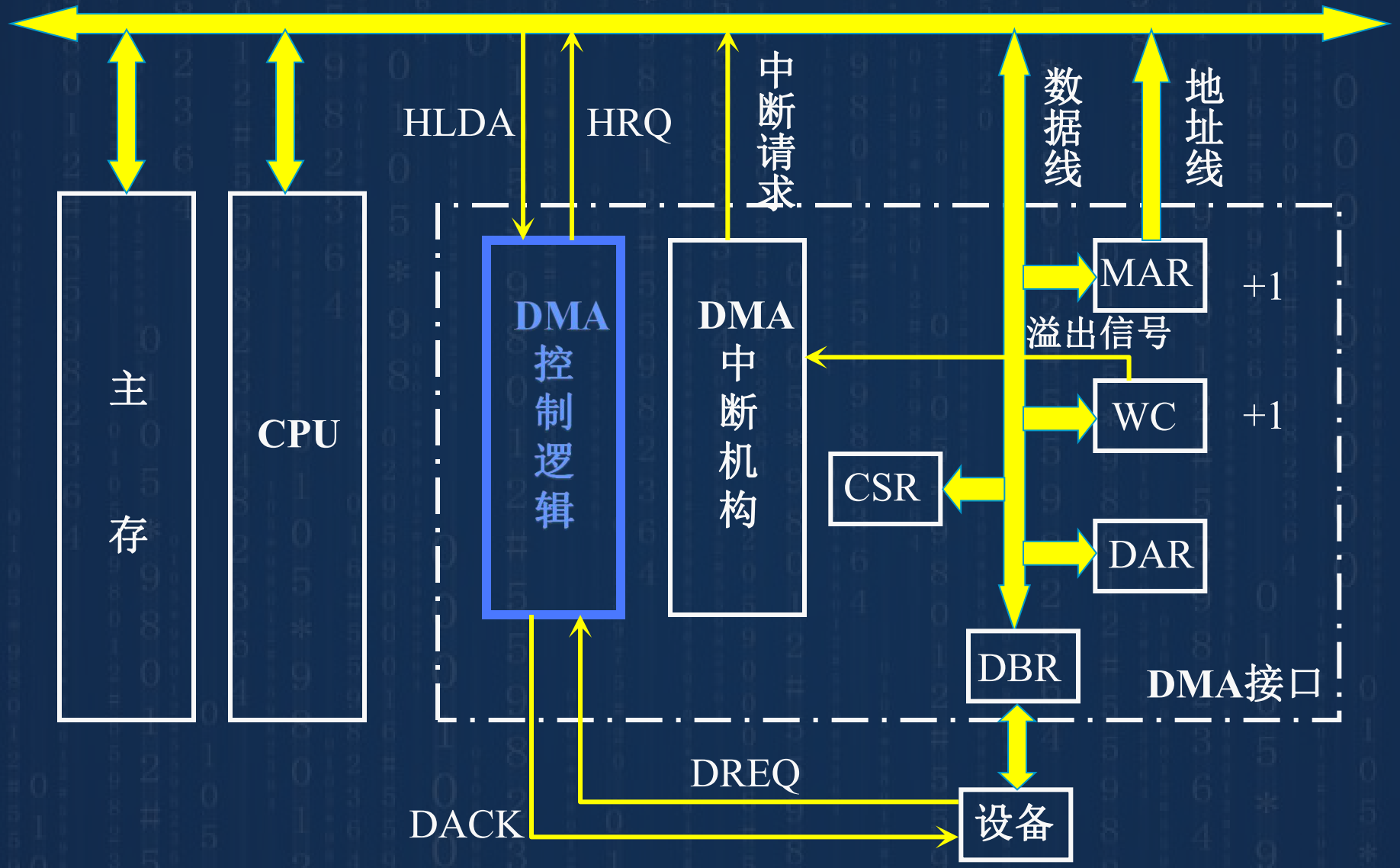
- **(2) 独立的 DMA 请求**


数据线
I/O总线
地址线
DMA请求3
DMA响应3
DMA请求2
DMA响应2
DMA请求1
DMA响应1
CPU
DMA
接口1
DMA
接口2
DMA
接口3
主存

# 6.8.4 DMA控制器的组成

- 为了实现DMAC的功能，DMAC内部除需要有接受和发送DMA请求和响应信号的能力外，还应具有地址寄存和计数功能，以便控制对存储器的寻址；具有传输量计数器，能够对传送的数据个数进行计数。

# DMAC的基本组成


HLDA
HRQ
中断请求
数据线
地址线
主存
CPU
DMA控制逻辑
DMA中断机构
MAR
+1
溢出信号
WC
+1
CSR
DAR
DBR
DMA接口
DREQ
DACK
设备

# 1.寄存器组

- (1) 主存地址寄存器MAR
- 用于存放主存中需要交换数据的地址。
- 在DMA传送前，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，将地址寄存器内容加/减1，指向下一单元，直到一批数据传送完毕为止。
- (2) 传输量计数器 WC
- 用于记录传送数据的总字数。
- 传输量计数器一般采用补码表示要传送的数据量。在DMA传送过程中，每传送一个字（或字节）,计数器自动加1，当WC内容溢出时，表示数据已全部传送完毕，DMAC发出DMA传送结束信号。



- (3) 数据缓冲寄存器DBR
- 用于暂存每次传送的数据。
- 通常DMA接口与主存之间采用字传送，而DMA与设备之间可能是字节或位传送。因此DMA接口中还可能包括有装配或拆卸字信息的硬件逻辑，如数据移位缓冲寄存器、字节计数器等。有的系统采用外设控制器上的数据缓冲器与内存单元之间通过数据总线直传的方法，这样就可以不用数据缓冲寄存器。
- (4) 设备地址寄存器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。如磁盘数据所在的区号、盘面号和柱面号。具体内容取决于设备的数据格式和地址的编址方式。

- **(5) 控制/状态寄存器 CSR**
- 存放有关控制和状态信息，如传送方式、读/写状态、传送完毕与否等。也可使用多个寄存器，分别存放控制字和状态字。
- **2. DMA控制逻辑**
- DMA控制逻辑负责完成DMA的预处理（初始化各类寄存器）、接收设备控制器送来的DMA请求信号、向设备控制器回答DMA允许（应答）信号、向系统申请总线以及控制总线实现DMA传输控制等工作。

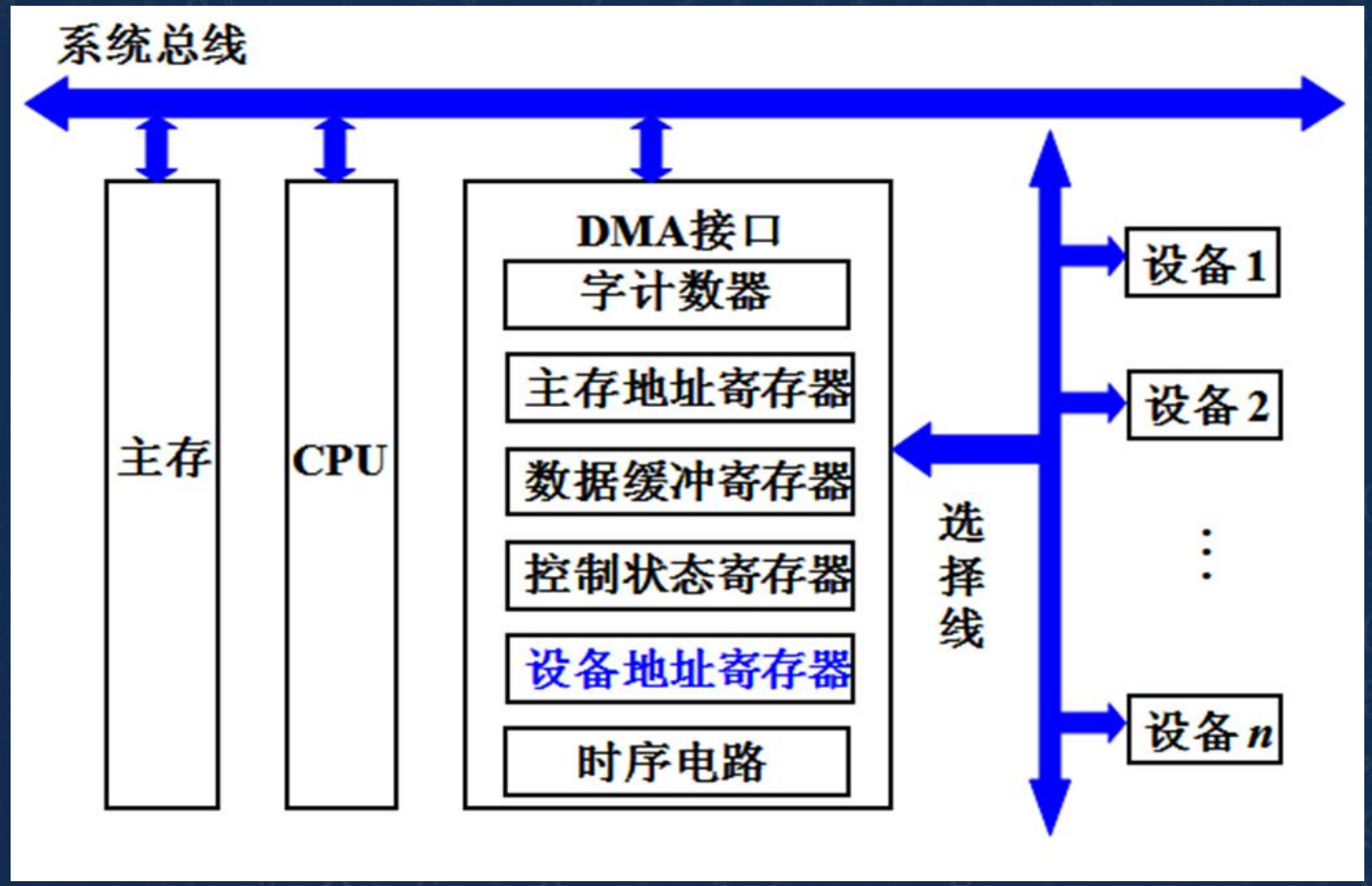


- **3. DMA中断控制逻辑**
- **DMA中断控制逻辑负责在DMA操作完成后向CPU发出中断请求，申请CPU对DMA操作进行后处理或进行下一次DMA传送的预处理。**
- **注意: DMA传送过程中的中断与I/O中断的技术相同，但中断的目的不同。I/O中断是为了数据的输入或输出，DMA传送过程中的中断是为了报告一批数据传送结束。它们是I/O系统中不同的中断事件。**
- **4.数据线、地址线和控制信号线**
- DMA控制器设置了与主机和I/O设备两个方向的数据线、地址线和控制信号线以及有关收发与驱动电路。



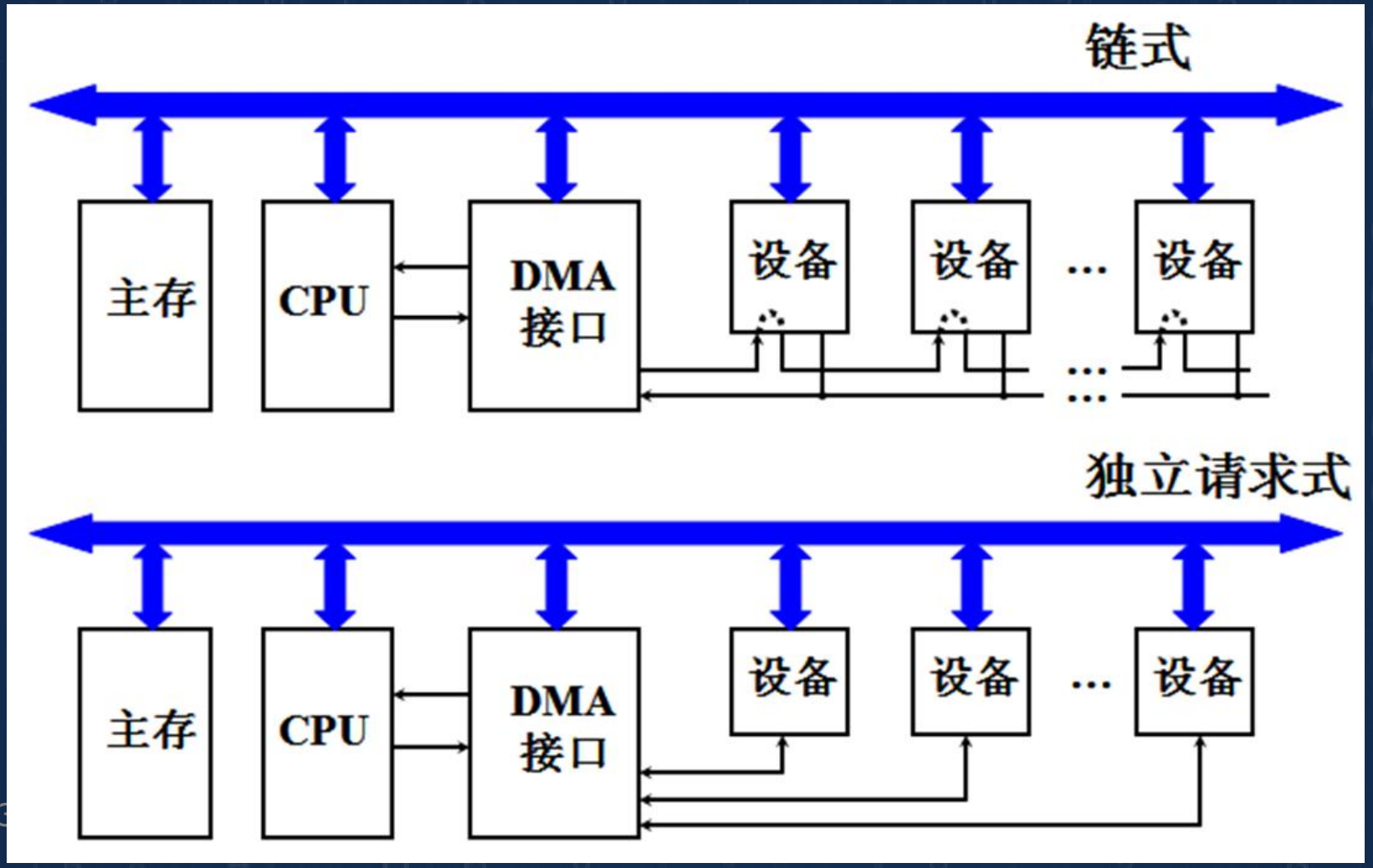
# *DMA接口的类型

- （1）选择型
- 物理上连接多个设备，逻辑上只许接一个设备。



# （2）多路型

- 物理上连接多个设备，逻辑上允许多个设备同时工作。



# 6.8.5 DMA控制的数据传送过程

- **1. DMA传送前预处理阶段**
- 在DMAC开始工作之前，CPU必须给它预置的信息：
- ① 控制寄存器写入DMA操作命令。给DMA控制逻辑指明数据传送方向是输入(主存写)还是输出(主存读)。
- ② 向DMA设备地址寄存器送入设备号，并启动设备。
- ③ 向DMA主存地址寄存器送入交换数据的主存起始地址。
- ④ 向传输量计数器送入交换数据的个数。



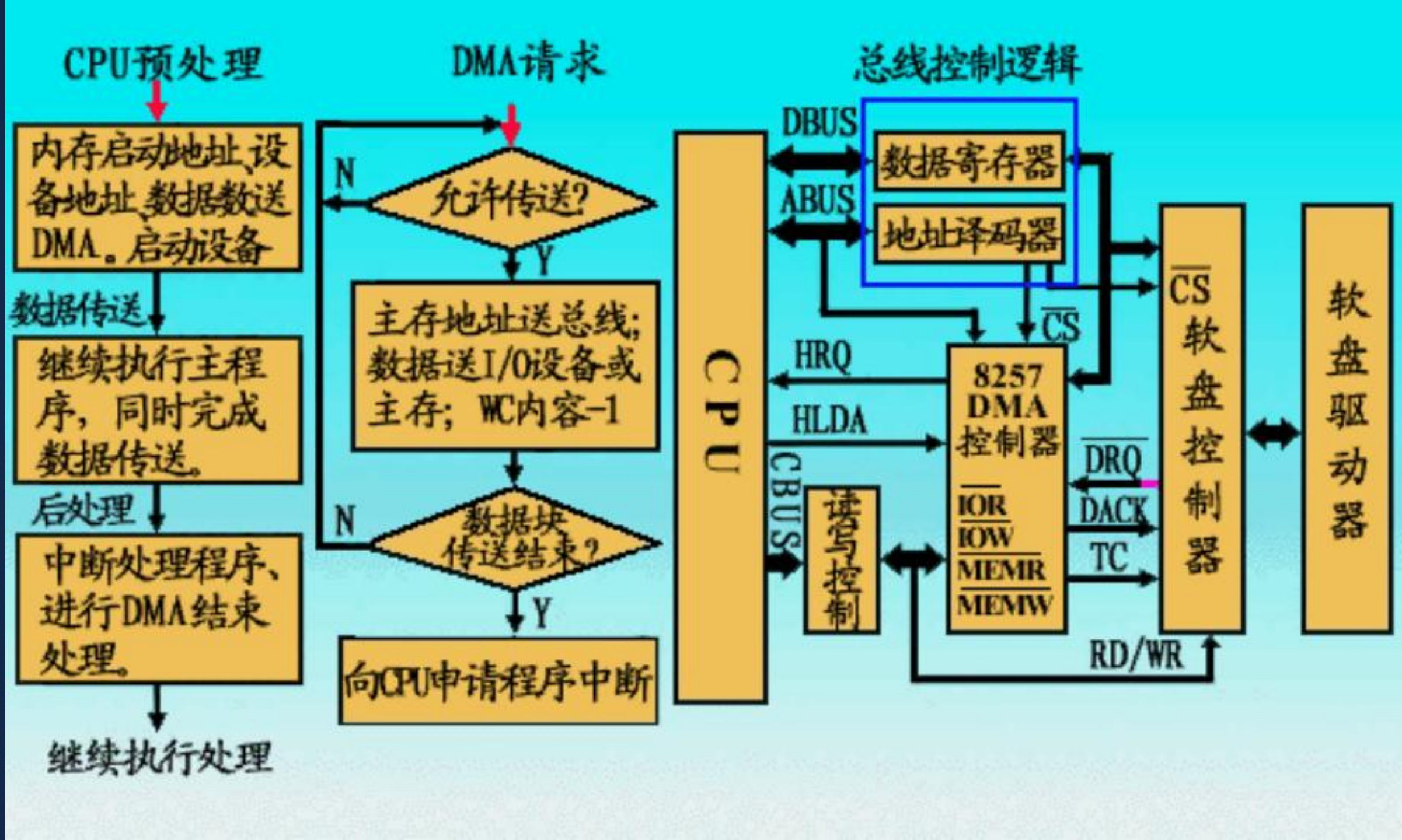
- 预处理工作由CPU执行几条输入输出指令完成，称为初始化工作。初始化工作完成后，CPU继续执行原来的程序。
- 当外部设备准备好发送的数据(输入)或上次接受的数据已经处理完毕(输出)时，它便通过DMA接口向CPU提出占用总线的申请，若有多个DMA同时申请，则按轻重缓急由硬件排队判优逻辑决定优先等级。待设备得到主存总线的控制权后，数据的传送便由该DMAC进行管理。

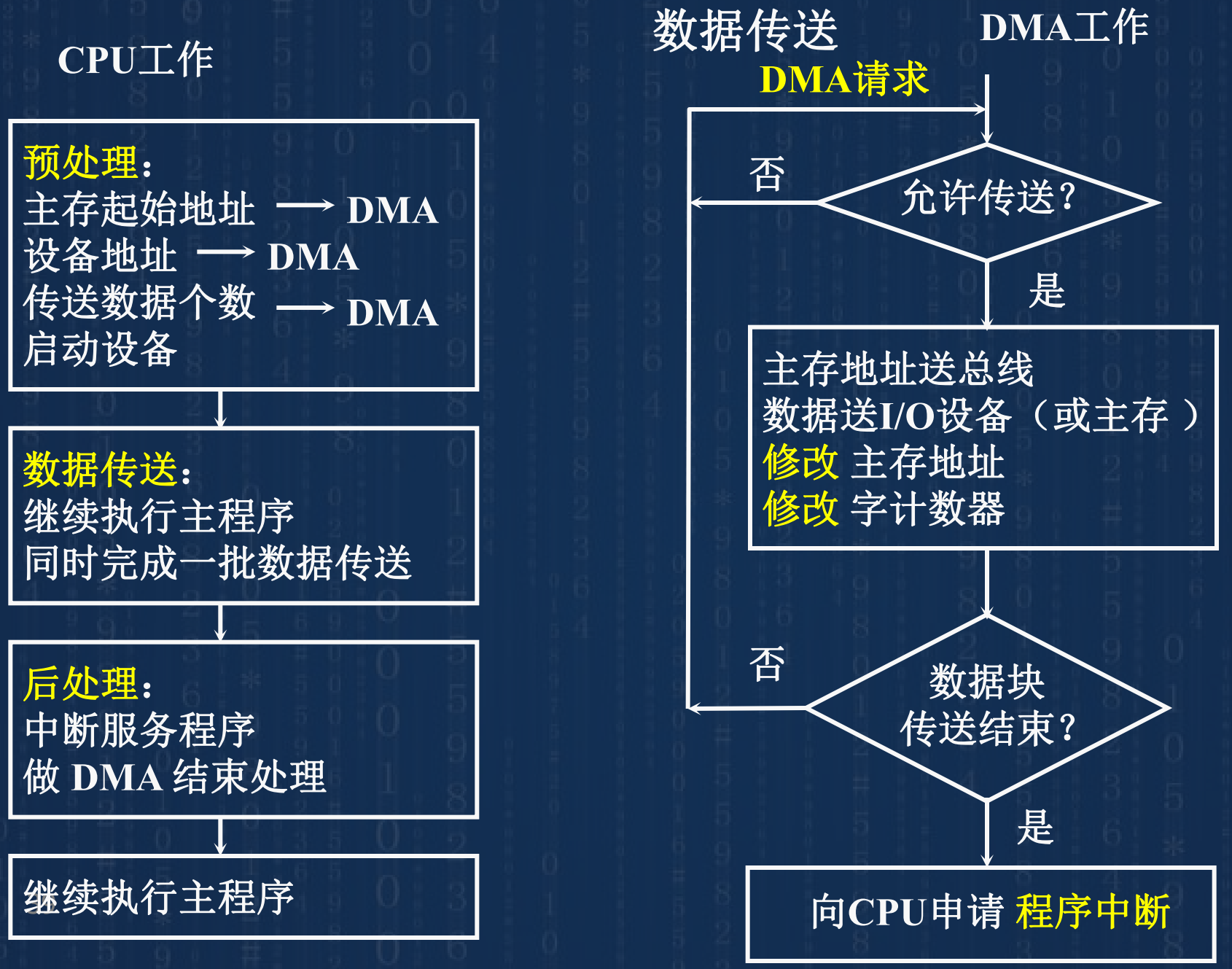
# 2. 数据传送操作

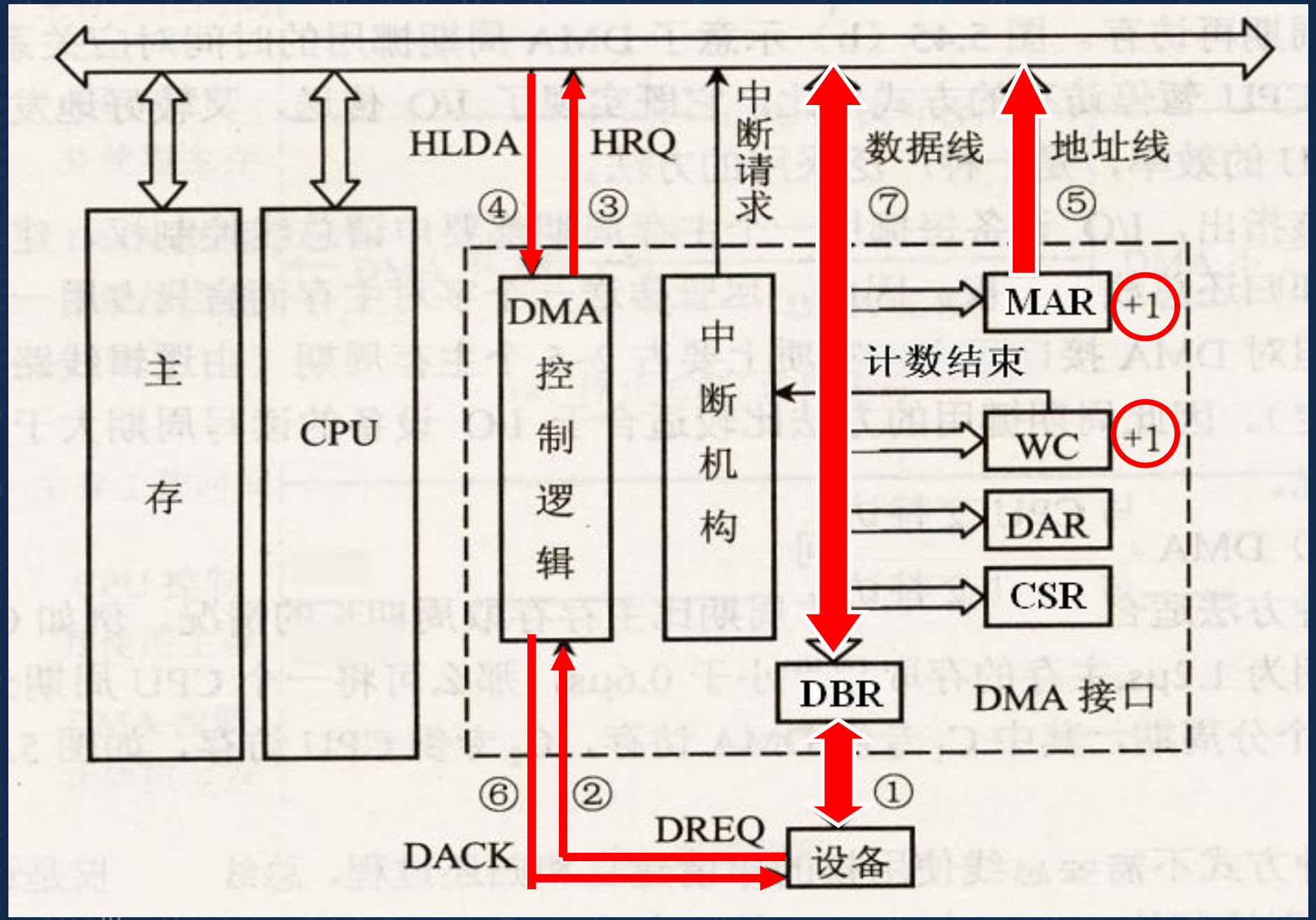
- DMAC获得总线后，即可按规定的传送方式，进行数据的输入或输出操作，直到将所有数据传输完毕，DMAC将总线交还给CPU。需要时还向CPU发出中断请求。

# 3. DMA传送后处理阶段

- CPU响应中断后，为DMA传送作结束处理工作。
- ① 校验送入主存的数据是否正确
- ② 决定是否继续用DMA方式传送，还是结束传送
- ③ 测试在传送过程中是否发生了错误
- ④ 判断传送工作是否正常结束


CPU预处理
内存启动地址、设备地址、数据数送DMA。启动设备
数据传送
继续执行主程序，同时完成数据传送。
后处理
中断处理程序、进行DMA结束处理。
继续执行处理
DMA请求
N
允许传送?
Y
主存地址送总线；数据送I/O设备或主存；WC内容-1
N
数据块传送结束?
Y
向CPU申请程序中断
CPU
总线控制逻辑
DBUS
ABUS
数据寄存器
地址译码器
CS
HRQ
HLDA
CBUS
读写控制
8257 DMA控制器
IOR
IOW
MEMR
MEMW
DRQ
DACK
TC
RD/WR
CS软盘控制器
软盘驱动器

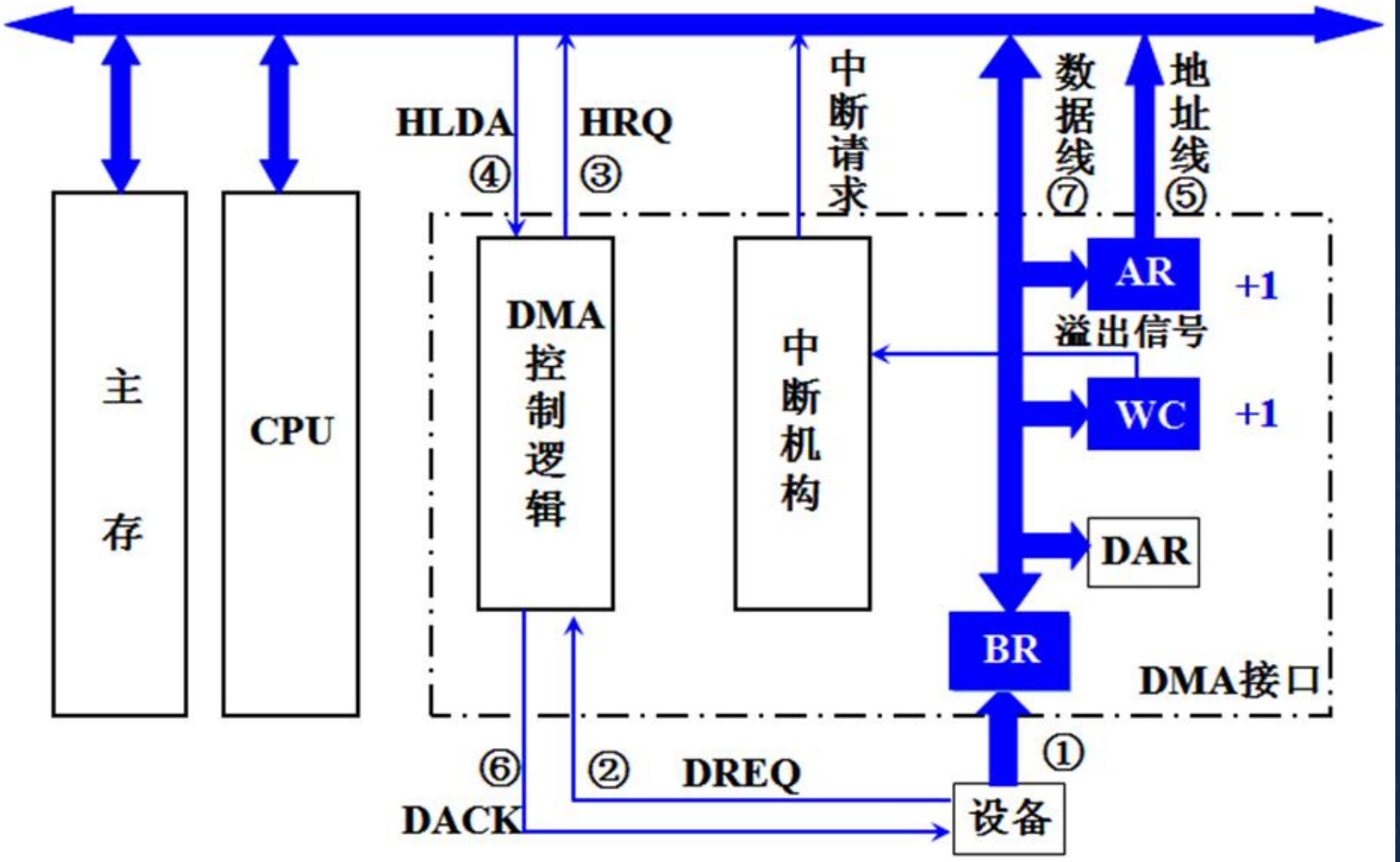

CPU工作
预处理：
主存起始地址 ⟶ DMA
设备地址 ⟶ DMA
传送数据个数 ⟶ DMA
启动设备
数据传送：
继续执行主程序
同时完成一批数据传送
后处理：
中断服务程序
做 DMA 结束处理
继续执行主程序
数据传送
DMA工作
DMA请求
否
允许传送？
是
主存地址送总线
数据送I/O设备（或主存）
修改 主存地址
修改 字计数器
否
数据块
传送结束？
是
向CPU申请 程序中断


主存
CPU
HLDA
④
HRQ
③
中断请求
数据线
⑦
地址线
⑤
DMA控制逻辑
中断机构
MAR
+1
计数结束
WC
+1
DAR
CSR
DBR
DMA 接口
⑥
DACK
②
DREQ
①
设备

- 例1：DMA数据输入过程（DMA写）
- 解：① 从设备读入一个字到DMA的数据缓冲寄存器BR中，表示数据缓冲寄存器“满”；
- ② 外设向DMAC发请求(DREQ)；
- ③ DMAC向CPU申请总线控制权(HRQ)；
- ④ CPU发回HLDA信号，表示允许将总线控制权交给DMAC；
- ⑤ 将DMA主存地址寄存器中的主存地址送地址总线；
- 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 向存储器发写命令；
- 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，则继续传送；若已结束，(字计数器计数为0)，则向CPU申请程序中断，标志数据块传送结束。

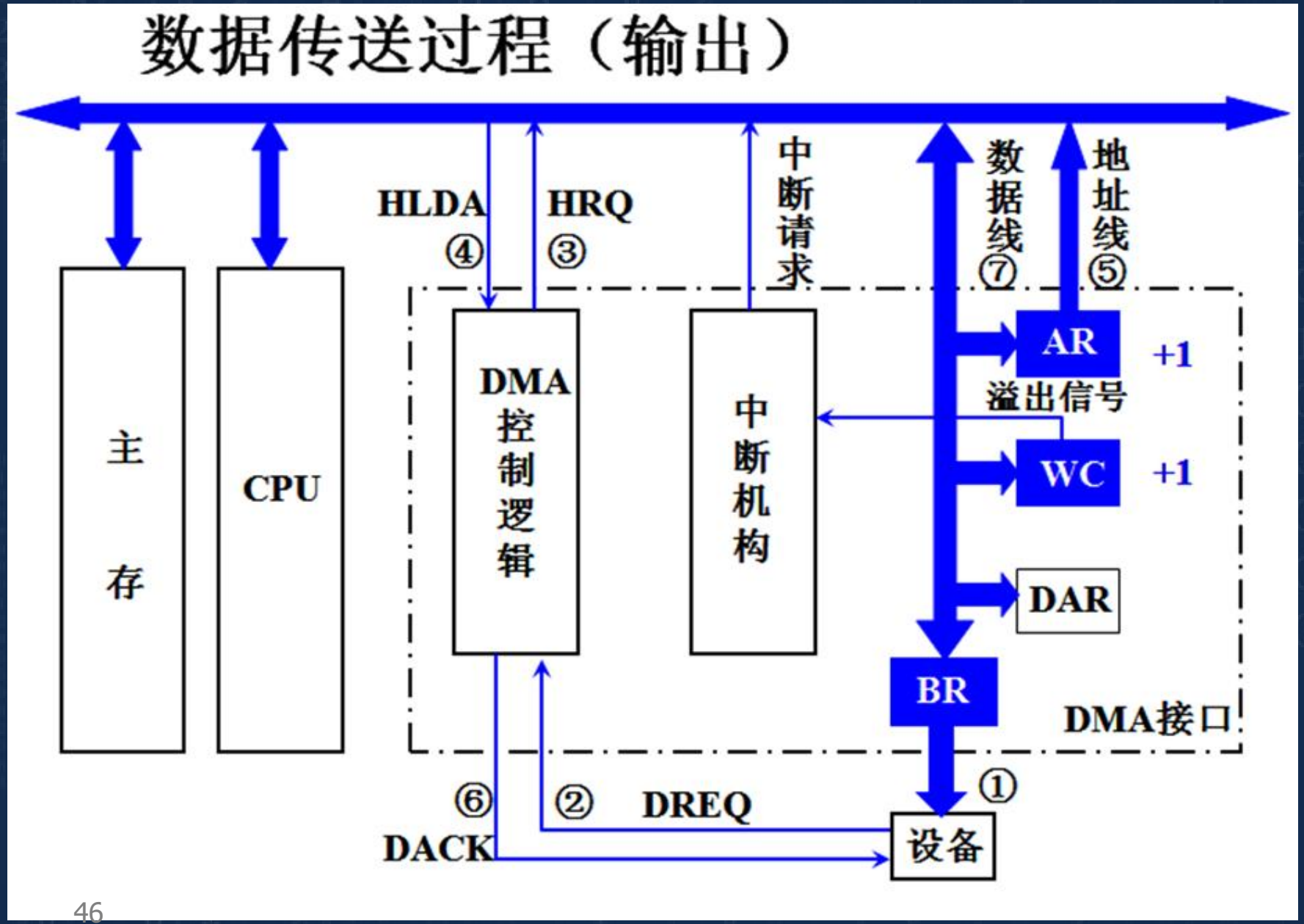
# 数据传送过程（输入）


主存
CPU
HLDA
④
HRQ
③
中断请求
数据线⑦
地址线⑤
DMA控制逻辑
中断机构
AR
+1
溢出信号
WC
+1
DAR
BR
DMA接口
⑥
DACK
②
DREQ
①
设备

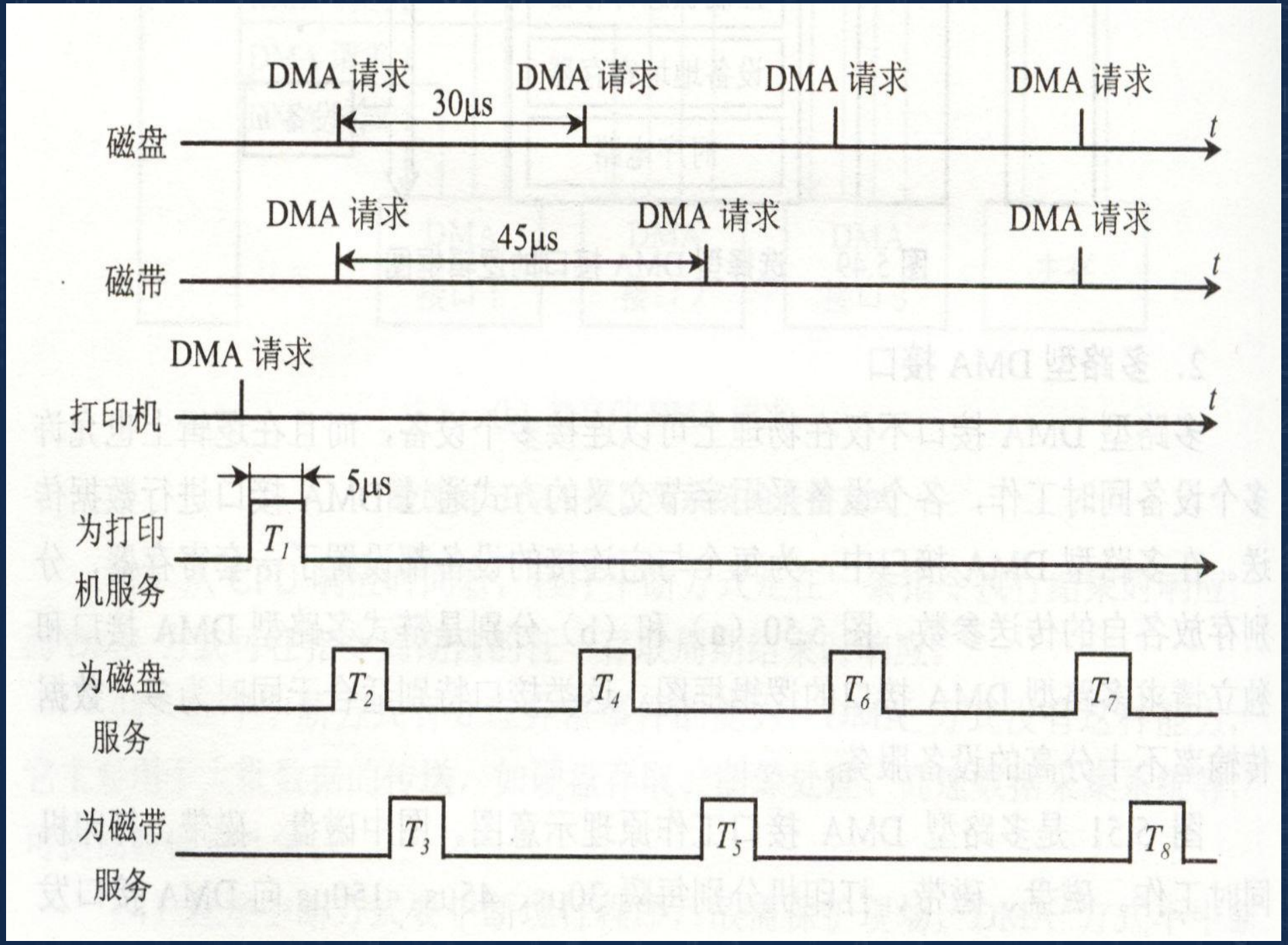
- 例2：DMA数据输出过程（DMA读）
- 解：① 当DMA数据缓冲寄存器已将输出数据送至I/O设备后，表示数据缓冲寄存器已“空”；
- ② 外设向DMAC发请求(DREQ)；
- ③ DMAC向CPU申请总线控制权(HRQ)；
- ④ CPU发回HLDA信号，表示允许将总线控制权交给DMAC使用；
- ⑤ 将DMA主存地址寄存器中的主存地址送地址总线，并发存储器读命令；
- ⑥ 通知设备已被授予一个DMA周期(DACK)，并为交换下一个字做准备；

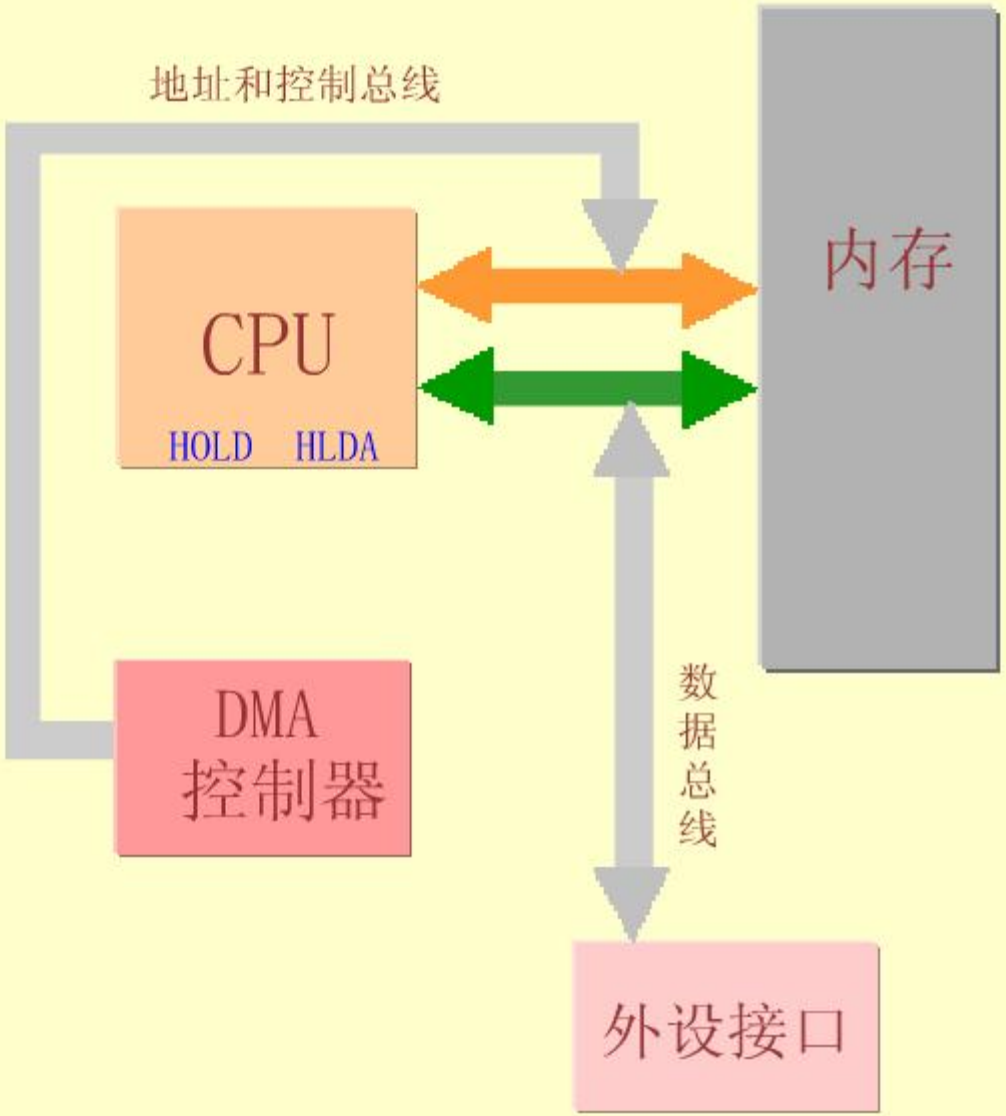
- ⑦ 主存将相应地址单元的内容通过数据总线读入到DMA的数据缓冲寄存器中；
- ⑧ 将DMA数据缓冲寄存器的内容送到输出设备；
- ⑨ 修改主存地址和字计数值；
- ⑩ 判断数据块是否已传送完毕，若未完，继续传送；若已送完，则向CPU申请程序中断。

# 数据传送过程（输出）


HLDA
④
HRQ
③
中断请求
数据线⑦
地址线⑤
主存
CPU
DMA控制逻辑
中断机构
AR
+1
溢出信号
WC
+1
DAR
BR
DMA接口
⑥
DACK
②
DREQ
①
设备

- 例 3：设磁盘、磁带、打印机同时工作。磁盘、磁带、打印机分别每隔30μs、45μs、150μs向DMA接口发DMA请求，磁盘的优先级高于磁带，磁带的优先级高于打印机。假设DMA接口完成一次DMA数据传送需5μs,打印机先请求DMA服务,稍后磁盘和磁带同时提出请求。请描绘DMA处理过程。
- 解：打印机首先发请求，故DMA接口首先为打印机服务(T1)，接着磁盘、磁带同时又有DMA请求，DMA接口按优先级别，先响应磁盘请求(T2)，再响应磁带请求(T3)，每次DMA传送都是一个字节。这样，在90多μs的时间里，DMA接口为打印机服务一次(T1)，为磁盘服务四次(T2、T4、T6、T7)为磁带服务三次(T3、T5、T8)。可见DMA接口还有很多空闲时间，可再容纳更多的外部设备。


DMA 请求
DMA 请求
DMA 请求
DMA 请求
30μs
磁盘
t
DMA 请求
DMA 请求
DMA 请求
45μs
磁带
t
DMA 请求
打印机
t
5μs
为打印机服务
$T_1$
为磁盘服务
$T_2$
$T_4$
$T_6$
$T_7$
为磁带服务
$T_3$
$T_5$
$T_8$



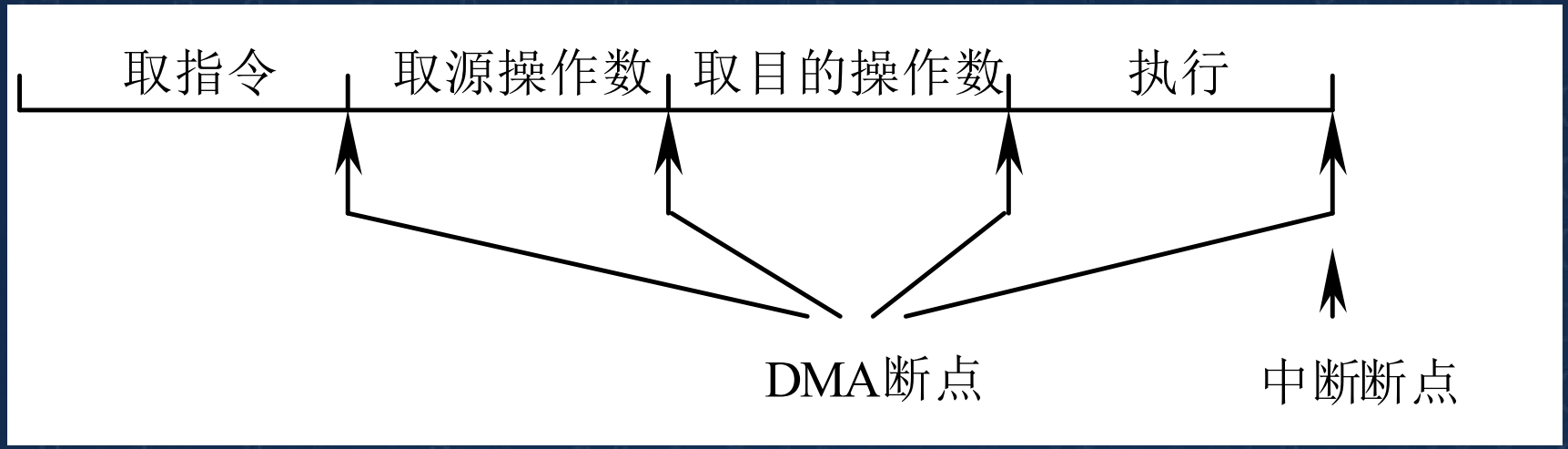

说明：

播放 停止

# DMA方式与程序中断的比较

| 程序中断 | DMA方式 |
| --- | --- |
| 以CPU为中心，采用软硬结合，以软件为主的方式，控制设备与主机之间的数据传送。 | 以主存为中心，采用硬件手段，控制设备与主存间直接进行数据传送。 |
| 因为需要程序切换，所以需要保护与恢复现场。 | 由DMA控制器直接控制数据传送。在数据传送期间，不需要CPU干预，不需保护与恢复现场。 |
| 适合于慢速外设。 | 适合于快速外设。 |
| 必须在一条指令执行结束后才能响应。 | 在一个访存周期结束后即可响应。 |
| 可实现多种处理功能 | 仅用于数据传送 |

# DMA方式与程序中断的比较（续）



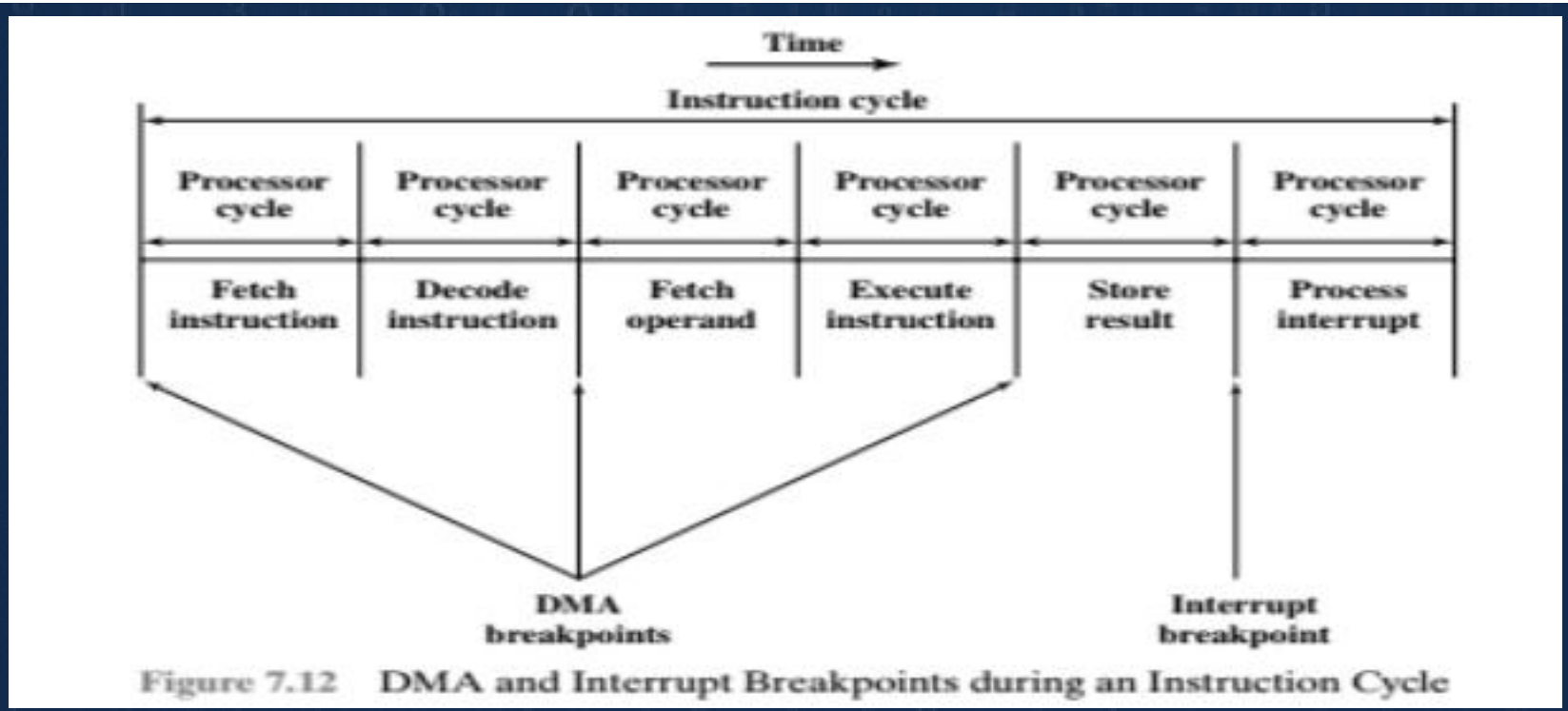




Figure 7.12 DMA and Interrupt Breakpoints during an Instruction Cycle

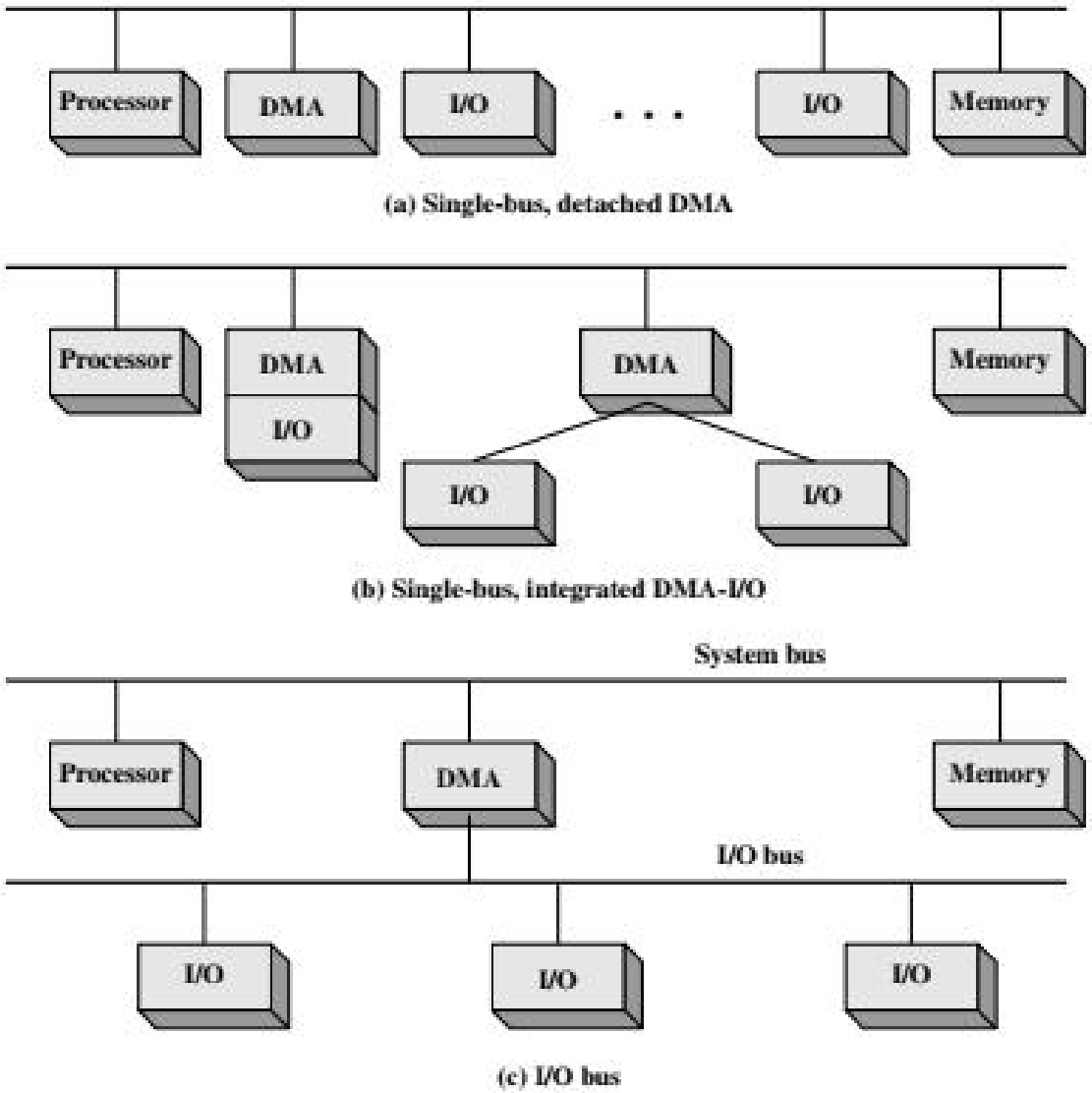



Figure 7.13 Alternative DMA Configurations

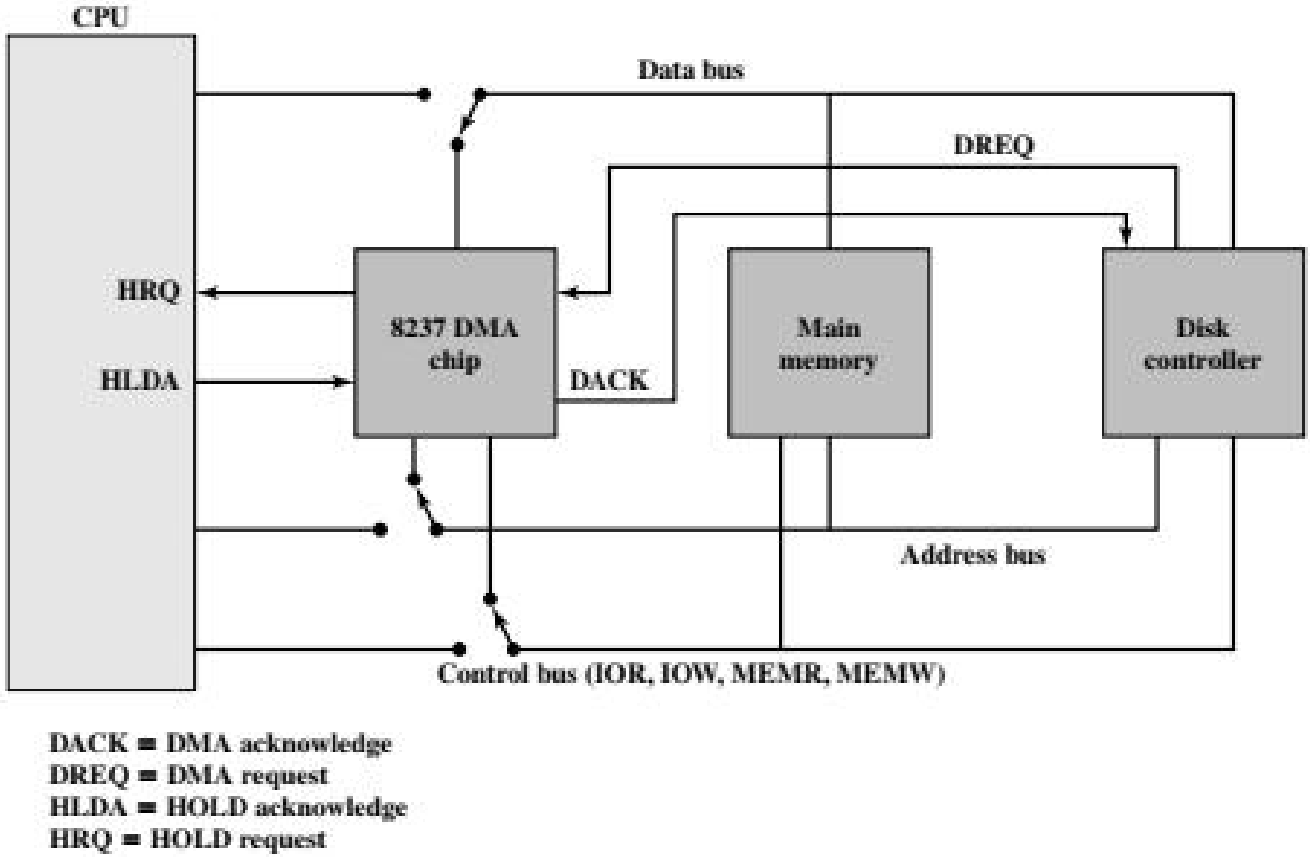


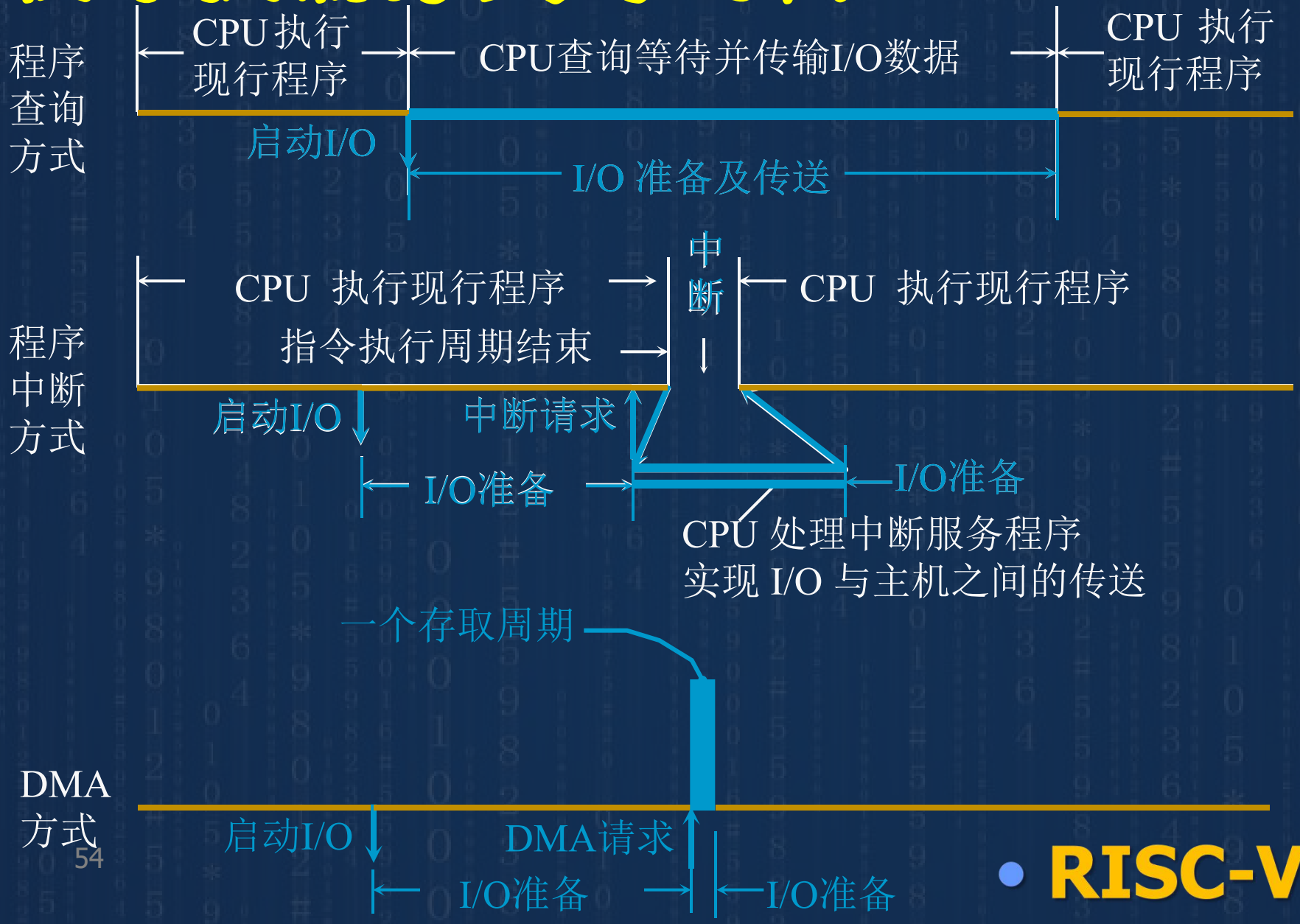

Figure 7.14 8237 DMA Usage of System Bus

Table 7.2 Intel 8237A Registers

| Bit | Command | Status | Mode | Single Mask | All Mask |
|---|---|---|---|---|---|
| D0 | Memory-to-memory E/D | Channel 0 has reached TC | Channel select | Select channel mask bit | Clear/set channel 0 mask bit |
| D1 | Channel 0 address hold E/D | Channel 1 has reached TC | | | Clear/set channel 1 mask bit |
| D2 | Controller E/D | Channel 2 has reached TC | Verify/write/ read transfer | Clear/set mask bit | Clear/set channel 2 mask bit |
| D3 | Normal/compressed timing | Channel 3 has reached TC | | Not used | Clear/set channel 3 mask bit |
| D4 | Fixed/rotating priority | Channel 0 request | Auto-initialization E/D | | Not used |
| D5 | Late/extended write selection | Channel 0 request | Address increment/ decrement select | | |
| D6 | DREQ sense active high/low | Channel 0 request | | | |
| D7 | DACK sense active high/low | Channel 0 request | Demand/single/block/ cascade mode select | | |

E/D = enable/disable
TC = terminal count

# 三种程序控制方式的比较






• RISC-V板卡演示

# 三种程序控制方式的例子

- 1. 程序直接控制方式的例子
- 设某程序查询方式的I/O系统挂有鼠标和硬盘两个外设，每个查询操作需要100个时钟周期。CPU的时钟频率为50MHz，且每秒对鼠标操作30次。若硬盘以32位字长为单位传输数据，即每32位被CPU查询一次，CPU访问硬盘的速率为2MB/s。求CPU对着两个设备查询所花费的时间比率，由此可得到什么结论？（不考虑预处理时间）

- 解： CPU的工作频率
- 50MHz=50 *$10^6$ (周期/s)
- （1） CPU每秒访问鼠标的总量： 30*100=3000 (周期/s)
- 查询鼠标的时间比为 3000/(50 *$10^6$ )=0.006%
- 不影响CPU的正常工作
- （2）CPU对硬盘的访问次数： 2MB/4B=500000 次/s
- 需要：100*500000=50 * $10^6$（周期/s）
- 故查询硬盘的时间比：50/50=100%
- 硬盘不适用程序查询方式。

- **2.** 程序中断控制方式的例子
- 设某外设传送信息的最高频率为每秒**40K**次,且相应的中断处理程序的执行时间为**40us**，问该外设是否可以采用中断方式工作？为什么？
- 解：外设传送一个数据的时间：
- **t=1/f=1/(40000/s)=25us**
- 而**I/O**中断一次的处理时间是**40us**
- 因此，中断有可能造成部分数据丢失，故不能采用中断方式。

- **3. DMA控制方式的例子**
- **（1）某磁盘采用DMA方式与主机交换信息，其传输速率为2MB/s。若DMA的预处理需要1000时钟周期，完成传输后的中断处理需要500个时钟周期。如果DMA平均传输的数据块长度为4KB，问磁盘工作时，50MHz的处理器需要多大的时间比率进行DMA操作？（忽略DMA与CPU争用主存情况）**

- 解：DMA的3个阶段中，预处理和后续中断需要CPU参与工作。
- 一次DMA传送需要一次预处理和一次后处理。
- DMA每秒传送磁盘数据执行次数：2MB/4KB =500 （次/s)
- 每次DMA操作需要CPU时间为：1000+500=1500 时钟周期
- CPU工作频率： 50 *$10^6$ （时钟周期/s）
- 所以，DMA操作占CPU时间的比率为：
- 500*1500/(50 *$10^6$ )=0.75/50=1.5%

- （2）设磁盘存储器转速为3000转/分，分8个扇区，每扇区存储1K字节，主存与磁盘存储器数据传送的宽度为16位（即每次传送16位）。假设一条指令最长执行时间是25μs，是否可采用一条指令执行结束时响应DMA请求的方案，为什么？若不行，应采取什么方案？

- 解：先算出磁盘传送速度，然后和指令执行速度进行比较得出结论。

  道容量=1KB×8÷16 =1K ×8 ×8 ÷16

  =1K ×4=4K字

  数传率=4K字×3000转/分

  =4K字×50转/秒 =200K字/秒

  一个字的传送时间=1/200K字/秒=5μs

  5 μs<<25 μs，所以不能采用一条指令执行结束响应DMA请求的方案，应采取每个CPU机器周期末查询及响应DMA请求方案（通常安排CPU机器周期=MM存取周期）。