汇编语言与接口技术

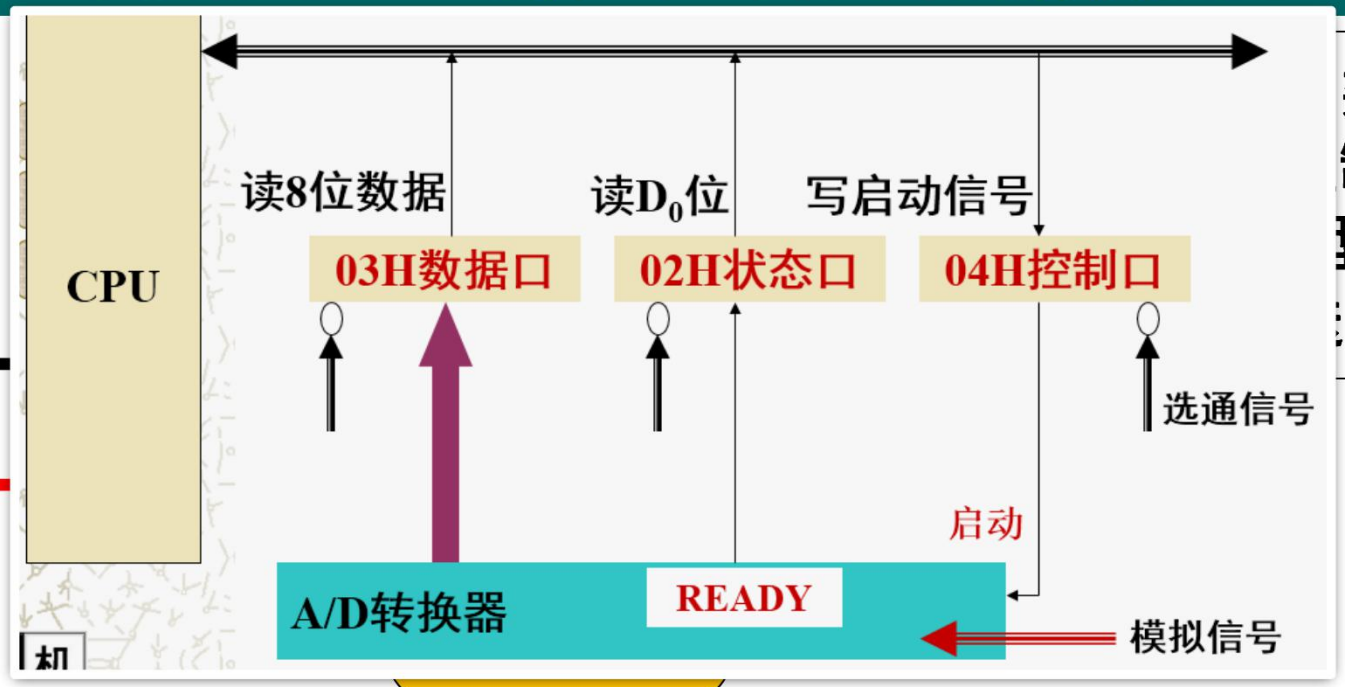
北京科技大学

第6章 输入输出接口及数据传输控制方式

- 本章介绍----微机与外设之间的数据传输问题
- 关于此问题，已经明确：
  - 微机与外设在物理上以接口(芯片)连接
  - 数据以端口传送
  - 8086/8088系统中，端口地址独立编制
  - 用16条地址线寻址端口，可寻64K个8位端口
  - 端口有直接寻址和寄存器间接寻址两种方式
  - 使用IN指令和OUT指令与累加器传送数据

本章主要介绍

与CPU和存储器之间的数据传送不同；外设工作速度慢，更多的时候不能直接传送数据，而应先行检测外设“准备就绪？”的状态，而后才启动数据传送



类型
制和
型信

AX

接口电路中的信号_数据信号

- **开关量**：以一根数据线与地线间**电压状态**表示的数据信息（高电平为1，低电平为0）；一个开关量表示一个二进制位
- **脉冲量**：以一根数据线与地线间**电压变化**表示数据，输出的**电平由低到高再由高到低**，如CPU的主频等
- **数字量**：多个开关量以二进制形式按一定规律组合起来的数据，一般为8位、16位、32位
- **模拟量**：以一根数据线与地线间电压或电流连续变化表示连续变化的物理量，如用0到5V的电压，表示0到100 ^{0}C的温度

接口电路中的信号_状态信号

也称为握手信息

状态信息是反映外设当前工作的状态信息

▲就绪否？对于输入设备，状态信息将表示是否准备好数据；只有数据准备好， CPU才能启动的一次读入操作

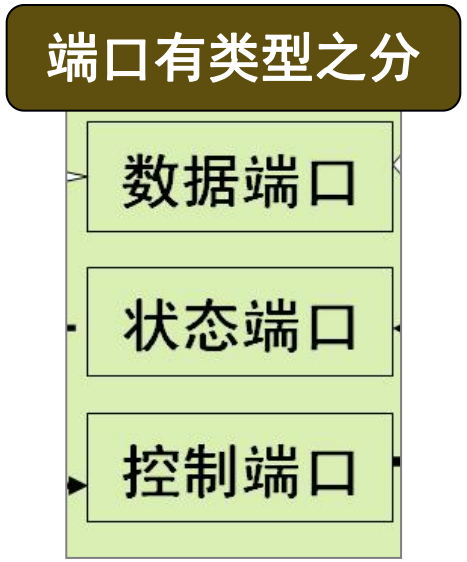
▲空闲吗？对于输出设备，状态信息将反映是否已经把上一个数据处理完，而处于“空闲”状态；只有上一个数据处理完， CPU才能启动的下一次的输出操作

接口电路中的信号_控制信号

控制信息是CPU通过端口传送给外设的控制信息，用来控制外设的工作状态(如：叫醒、关闭、设置或改变工作方式等信号)

不同信号的区分问题

- 接口中的三种信息，**都通过数据总线在CPU和接口(端口)之间传送**；从形式上看，三种信息并没有差别，都是二进制信息；**如10000010**：
  - 既可能是一个数据信息
  - 也可能是一个状态信息
  - 还可能是一个控制信息
  - **区分他们：根据保存和传送的端口类型**

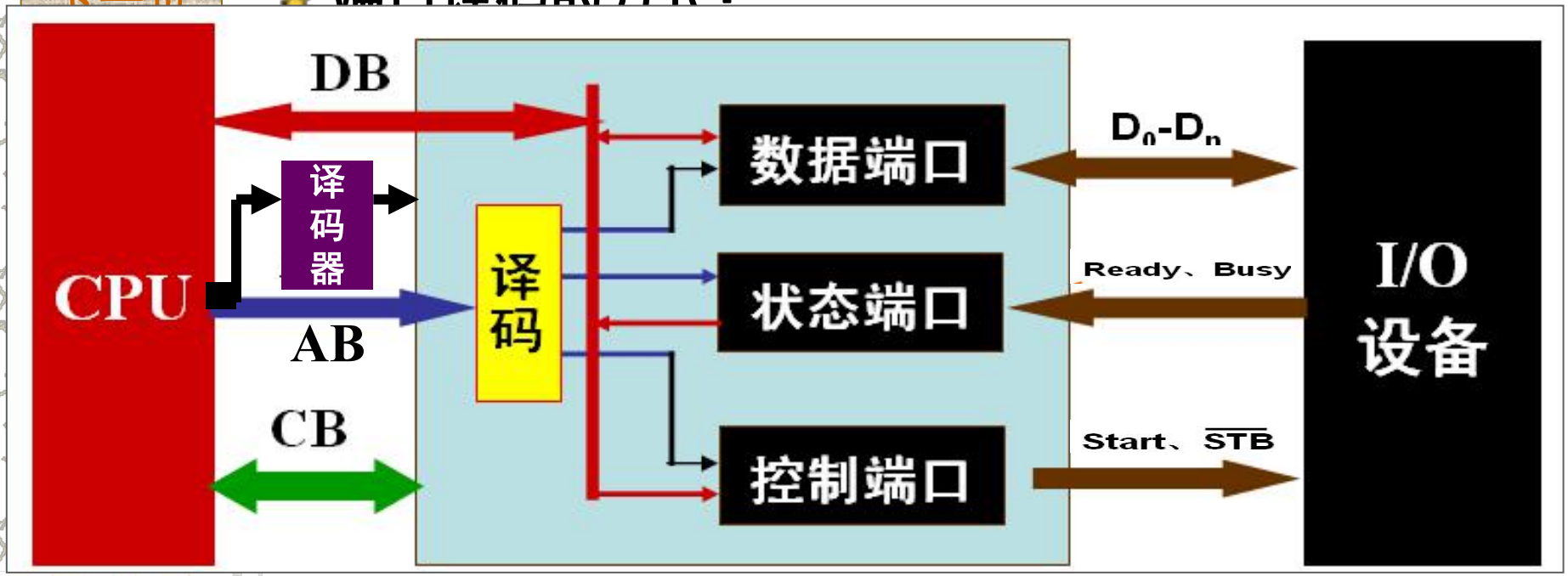


端口有类型之分

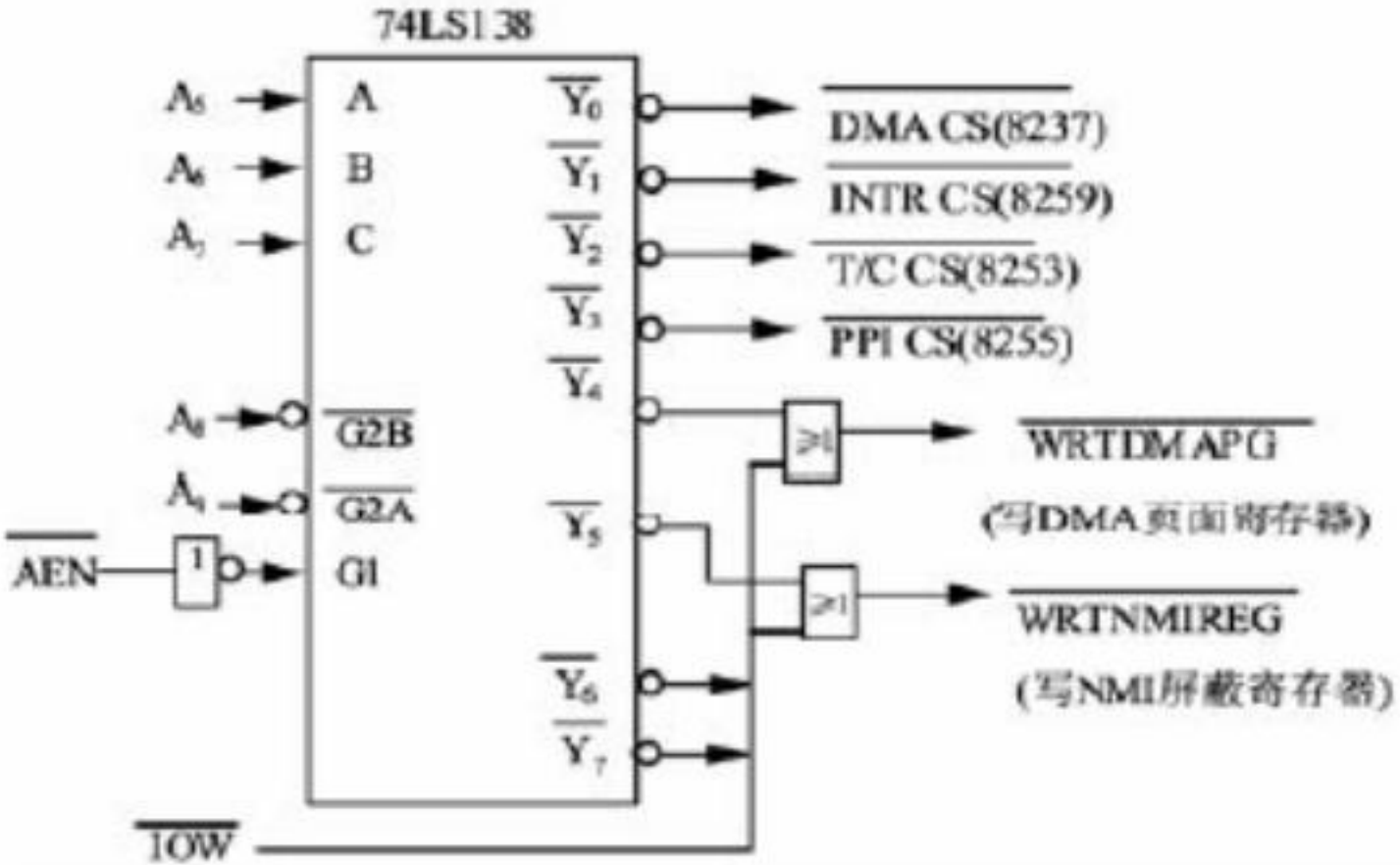
- 端口是接口电路中的寄存器
- 根据端口内暂存和将传送的信息类型的不同，可分为：
  - ◆数据端口、控制端口和状态端口；分别进行不同的操作：
  - ◆ 数据的输入/输出：CPU对数据端口进行一次读或写操作
  - ◆ 控制信息的输出：CPU把控制或操作信息写入控制端口
  - ◆ 状态信息的输入：CPU对状态端口进行一次读操作

端口译码问题

- 微机系统往往有多个不同类型的接口；在接口电路中，又各自包含数量不等、类型不同的端口
- 端口译码的方式：



接口地址译码图示例


74LS138
A
B
C
G2B
G2A
G1
AEN
1
IOW
Y0
Y1
Y2
Y3
Y4
Y5
Y6
Y7
DMA CS(8237)
INTR CS(8259)
T/C CS(8253)
PPI CS(8255)
≥1
≥1
WRTDMAPG
(写DMA页面寄存器)
WRTNMIREG
(写NMI屏蔽寄存器)

CPU与外设之间的数据传送方式

- CPU与外设之间的数据传送有两类不同的实现方式
  - **程序控制方式**--通过CPU执行程序中的**I/O指令**（IN和OUT）来实现和完成
    - 又分为：无条件传送方式和查询传送方式
  - **非程序控制方式**—依靠非I/O指令方式，通过**中断**或专门的**硬件**组件，实现数据传送

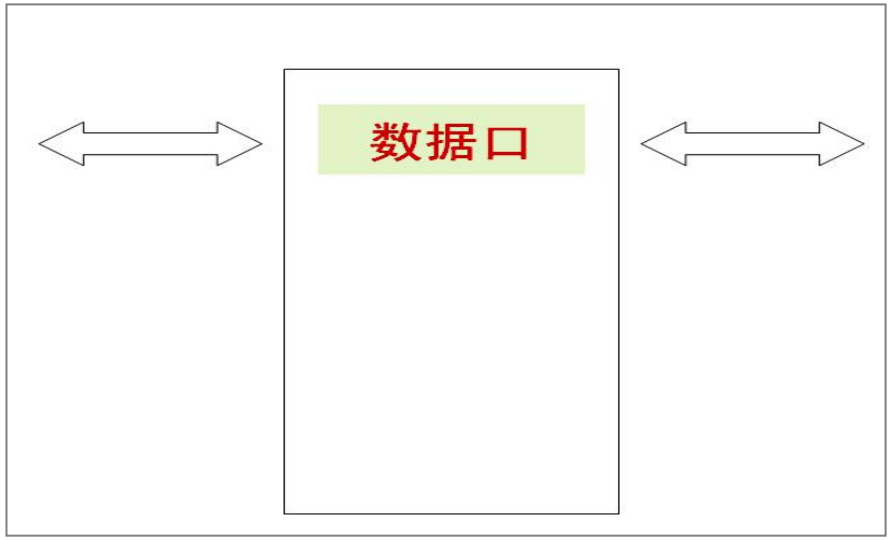
无条件传送方式

无条件是指CPU不首先检测外设的状态，在需要与外设交换信息的时候，无需询问，就可立即启用I/O指令

无需检测，是因为设备不需特别准备，一直处于“就绪”状态

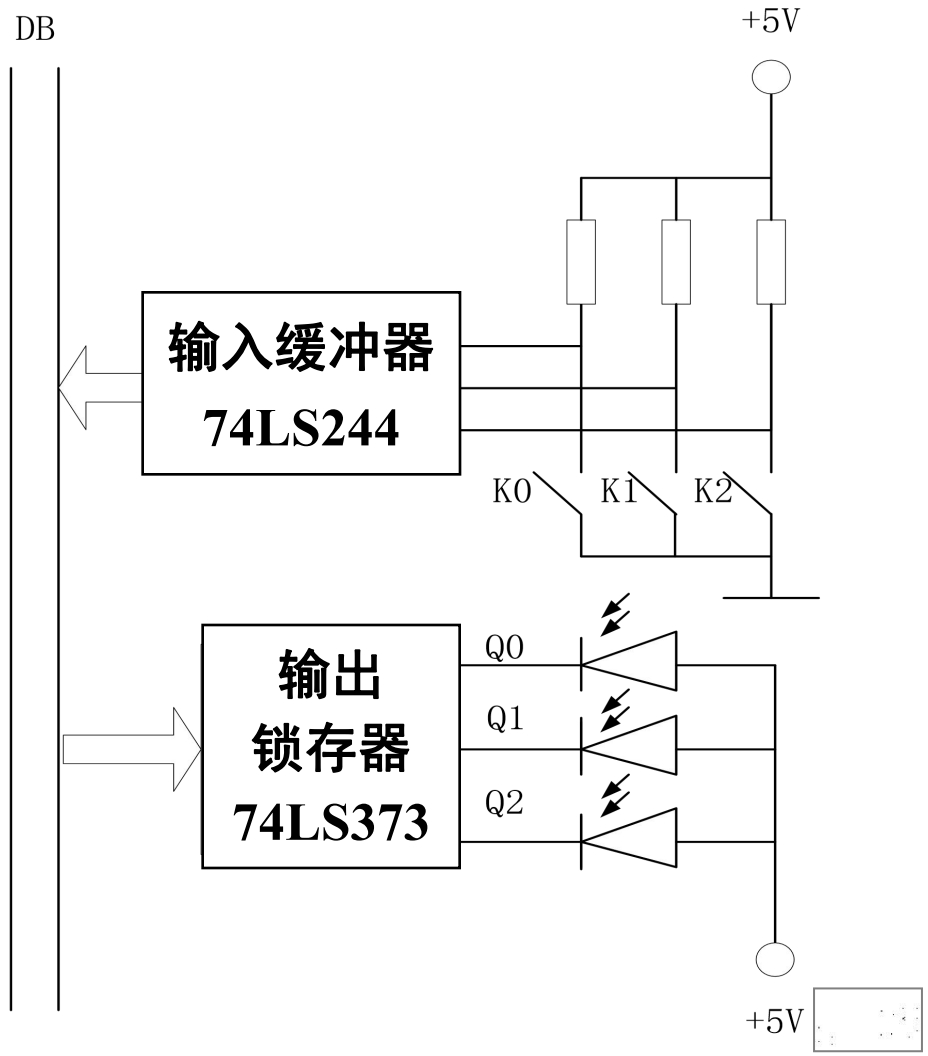
在这种方式下，CPU和外设之间只有数据信息的传送，不需要状态和控制信息的传送

接口电路比较简单



无条件传送方式例

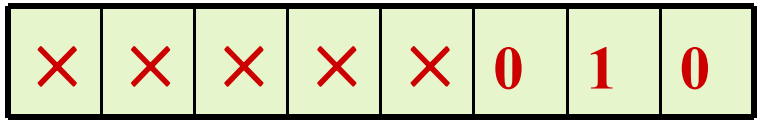
- 控制开关K0～K2打开(K0=1)则对应发光二极管Q0～Q2亮(Q0=0)
- 开关闭合(K0=0)则对应发光二极管不亮(Q0=1)
- 开关信号是输入信号；亮与不亮是输出信号



无条件传送方式例的典型代码

```
DON：IN  AL，IN_PORT      ；读入输入端口的开关状态信号
     XOR  AL，0FFH        ；各位求反
     OUT OUT_PORT，AL     ；通过输出端口输出点亮信号
     JMP  DON             ；准备读下一个输入开关状态
```

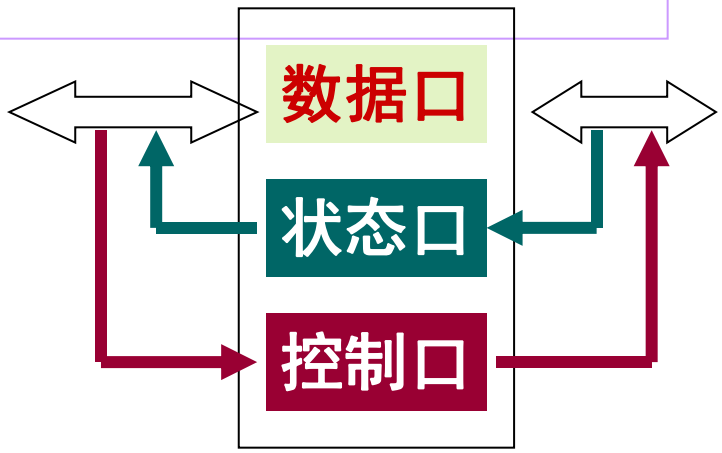
IN_PORT端口数据



OUT_PORT端口数据

数据的“可靠性”传送问题

- 对一般设备而言，因其工作特性和特点不同，不能保证总是处于“就绪”状态
- 实现“可靠性”传送，必须事先测试“就绪否”
- 实现“可靠性”传送，需要发送“叫醒”、“结束”，或其他的控制信号

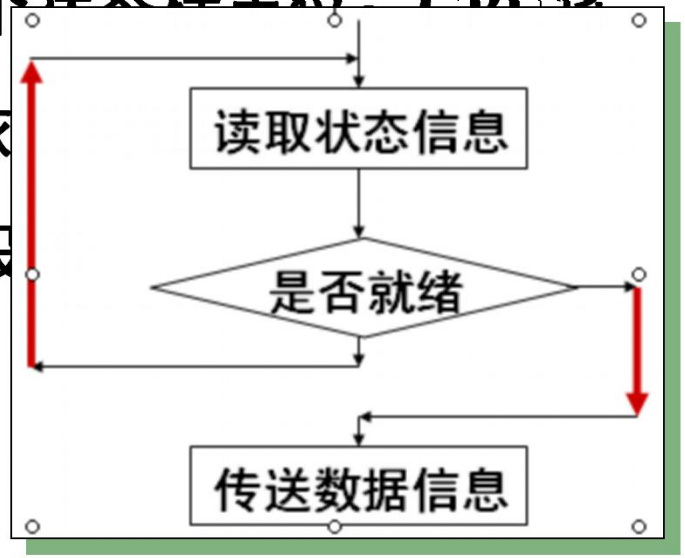


查询传送方式

- 在执行I/O指令之前时，CPU将需要首先主动地查询设备是否“就绪”；确认就绪后，才启动数据传送
- 查询式传送的一般流程：
  - 先从状态端口读入状态字，了解外设的工作状态
  - 如果状态信息是“准备好”，启动IN或OUT指令完成数据的传送
  - 如果状态信息是“没有准备好”，则继续查询，直到“准备好”

状态字的意义

- “状态字”是一组状态端口中的二进制数
- 其中的某一位(状态标志位) 的0/1别表示某个具体设备的就绪与否
- 如有多个设备，可定义多个状态标志位，CPU将采用轮询的方法，按顺序依
- 某位就绪，即服务对应的设
- 然后继续查询



查询传送的两个环节

- 查询环节
  - 寻址状态端口；
  - 读取接口中状态端口的标志信息
  - 若不
    续查
- 传送环节
  - 寻址接口芯片中的数据端口
  - 若为输入操作，通过输入IN指令从数据端口，读取数据
  - 若为输出操作，通过输出OUT指令向数据端口，输出数据

查询输入程序实现

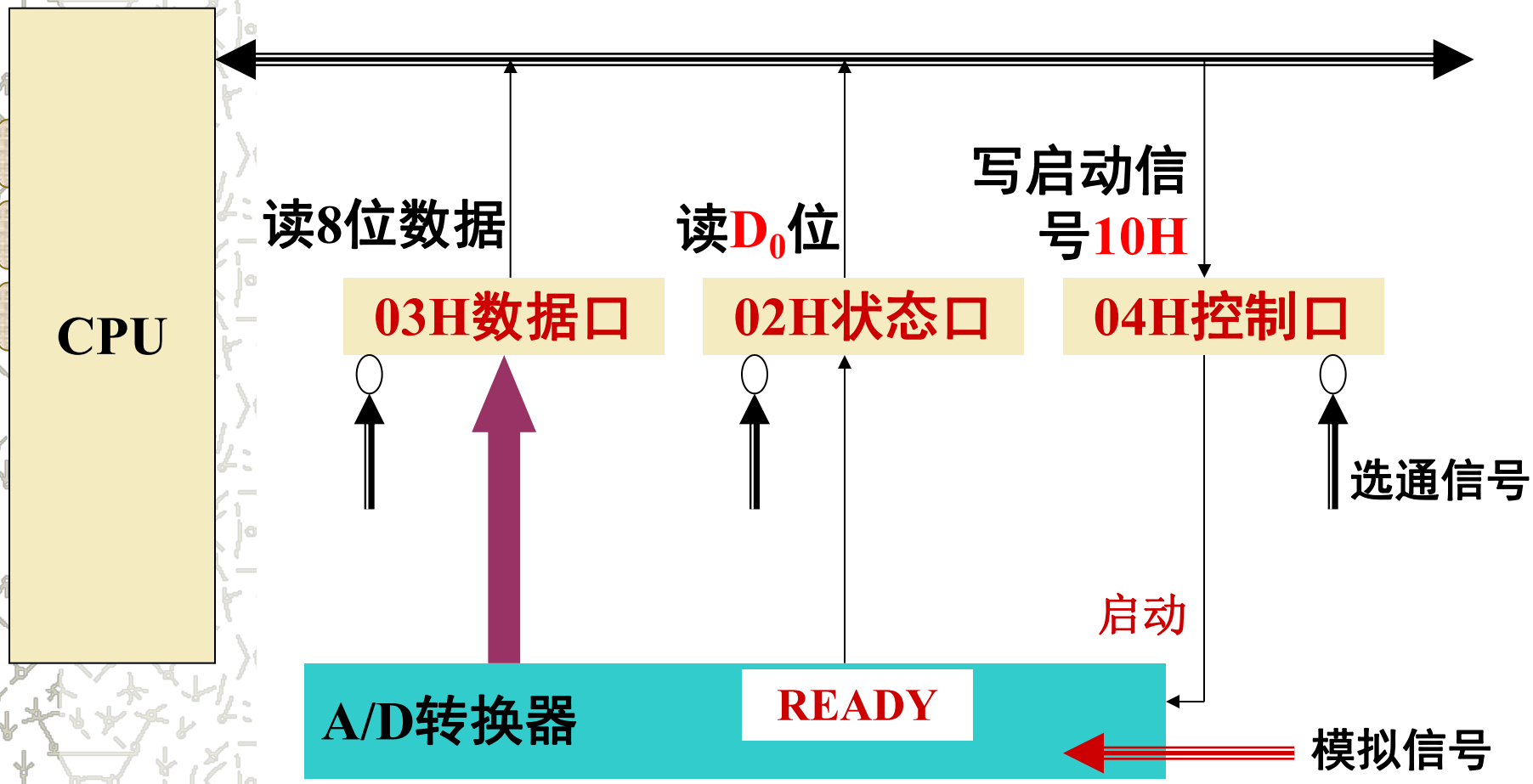
```
        MOV  DX，STATUS_PORT   ；DX指向状态端口
START：IN   AL，DX            ；读状态端口信息，间接寻址
        TEST  AL，XXH          ；测试状态标志位Dx
        Jxx   START            ；外设未就绪继续查
        MOV  DX，DATA_PORT     ；DX指向数据端口
        IN  AL, DX             ；从数据端口读取数据
```

上

下

退

从A/D转换器输入数据信息例



从A/D转换器输入数据信息例

```
START:   MOV  AL, 10H      ;10H为操作启动信号
         OUT  04H,AL       ;送控制口,启动A/D
LOOP1:   IN   AL,02H       ;读状态口
         AND  AL,01H       ;测试D0位
         JZ   LOOP1        ;继续吗?
         IN   AL,03H       ;读取数据
         MOV  STORE, AL    ;数据存入内存
         HLT               ;暂停
```

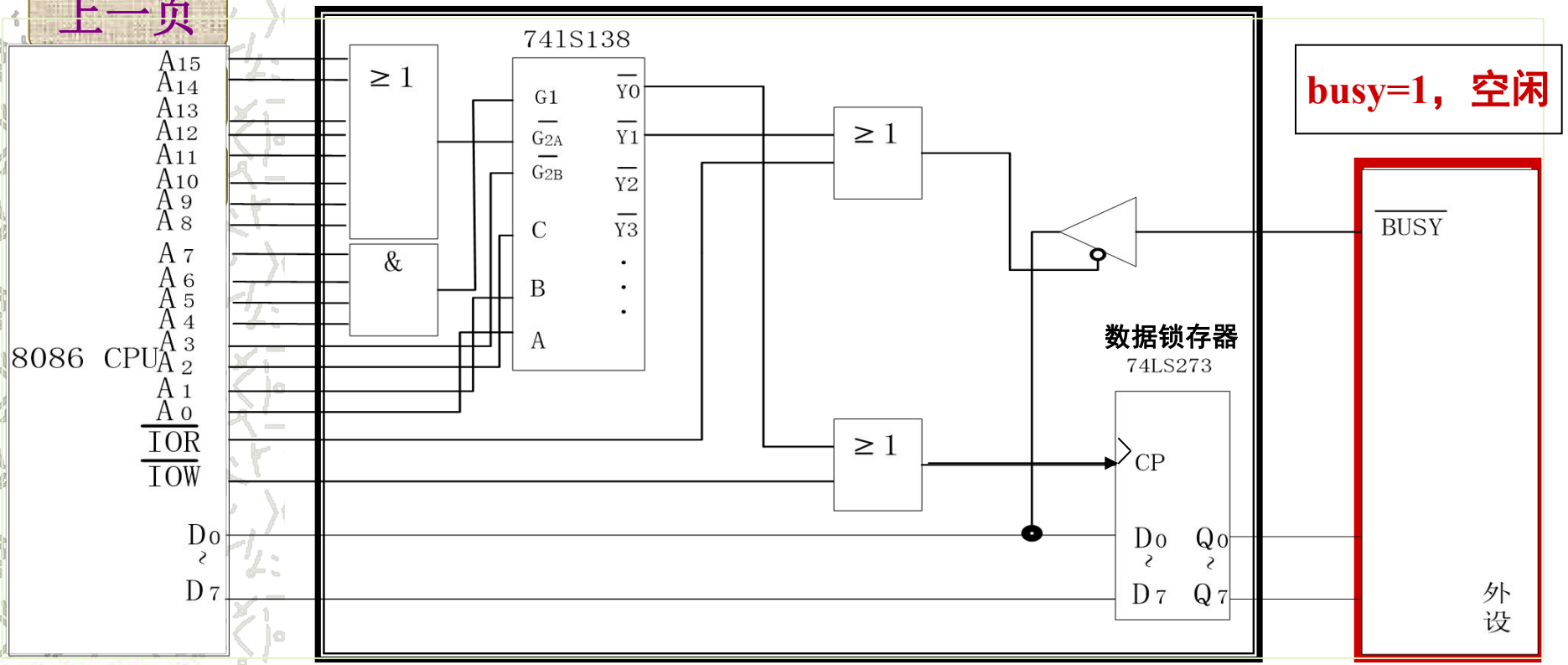
查询输出程序实现

```
        MOV  DX，STATUS_PORT     ；DX指向状态端口
START：  IN  AL，DX              ；读状态端口信息
         TEST  AL，XXH           ；测试状态标志位Dx
         Jxx   START             ；未就绪
                                 ；继续查询
        MOV  DX，DATA_PORT       ；DX指向数据端口
        MOV  AL，BUF             ；取数据
        OUT  DX，AL              ；向外设输出数据
```

查询传送的特别例

- 试编程实现将BUF为首地址的顺序100个存储单元中的数据，利用查询方式输出到外设

上一页



查询传送的例

首先$\overline{IOR}$读状态信号

待BUSY为高电平时，发$\overline{IOW}$，送出数据到外设

```
START:  MOV  SI,BUF
        MOV  CX,100
GOON:   MOV DX,_____
WAIT:   IN   AL,DX
        AND  AL,_____
        J___ WAIT
        MOV  AL,[SI]
        MOV  DX,_____
        OUT  DX,AL
        INC  SI
        DEC  CX
        J____GOON
        MOV  AH,4CH
        INT 21H
```

查询传送方式的问题

- 查询传送方式，CPU需要花大量的时间去读状态字，效率低；涉及多个外设操作时，需要采用轮询机制，因此实时性差
- CPU在每个T状态中都必须主动地检测是否“就绪”，会影响CPU的工作效率
- 适用于外设不多，实时性要求不高的系统

中断传送方式

在需要进行I/O操作时，CPU将启动(叫醒)外设；但不是立即进行数据传送，而是继续去作自己的工作；此时外设和CPU是并行工作的，即外设做数据传送的准备，CPU继续执行其他的程序

当外设准备就绪后，向CPU发出中断请求

一旦CPU响应该请求，就将暂停原程序的执行，而转去执行一段预先安排好的中断服务程序；该段服务程序将完成CPU与外设之间的数据传送

中断传送方式特点

- 由于实现了“并行工作”，中断传送是一种比查询传送效率更高的程序传送方式
- 数据传送的中断服务程序需预先设计好
- 中断请求由外设随机向CPU提出
- CPU对请求的检测是有规律的：一般是在每一个指令的最后一个时钟周期，采样中断请求输入引脚
- 但是
  - 每传送一次数据，就要中断一次
  - 不适用大数据量传送

直接存储器存取DMA方式 (Direct Memory Access)

- 希望克服程序控制传送的不足：

  外设 → CPU控制 → 存储器

  外设 ← CPU控制 ← 存储器

- 直接存储器存取 DMA：

  外设 → 存储器

  外设 ← 存储器

- CPU释放总线，I/O操作完全由DMA管理

DMA控制的工作原理

DMA控制的基本工作原理

数据传送发生在外设(接口)与存储器之间

配置专门芯片—DMA控制器

需要传送数据时，外设向DAM申请，DAM向CPU请求总线，CPU响应并让出总线

DAM管理数据的传送的具体事务

整个过程无程序参与,由DMA控制传送