第4章　输入/输出接口技术

学习要点

1. 了解接口的基本概念

2. 了解CPU与外设之间的数据传送方式

第4章 输入/输出接口技术

第1节 接口技术的概述

一、接口的概念和功能

二、I/O接口电路的典型结构

三、端口的编址和操作

第2节 CPU与外设间的数据传送方式

一、无条件传送方式

二、条件传送方式

三、中断传送方式

四、DMA传送方式

第1节　接口技术的概述

一、 接口的概念和功能

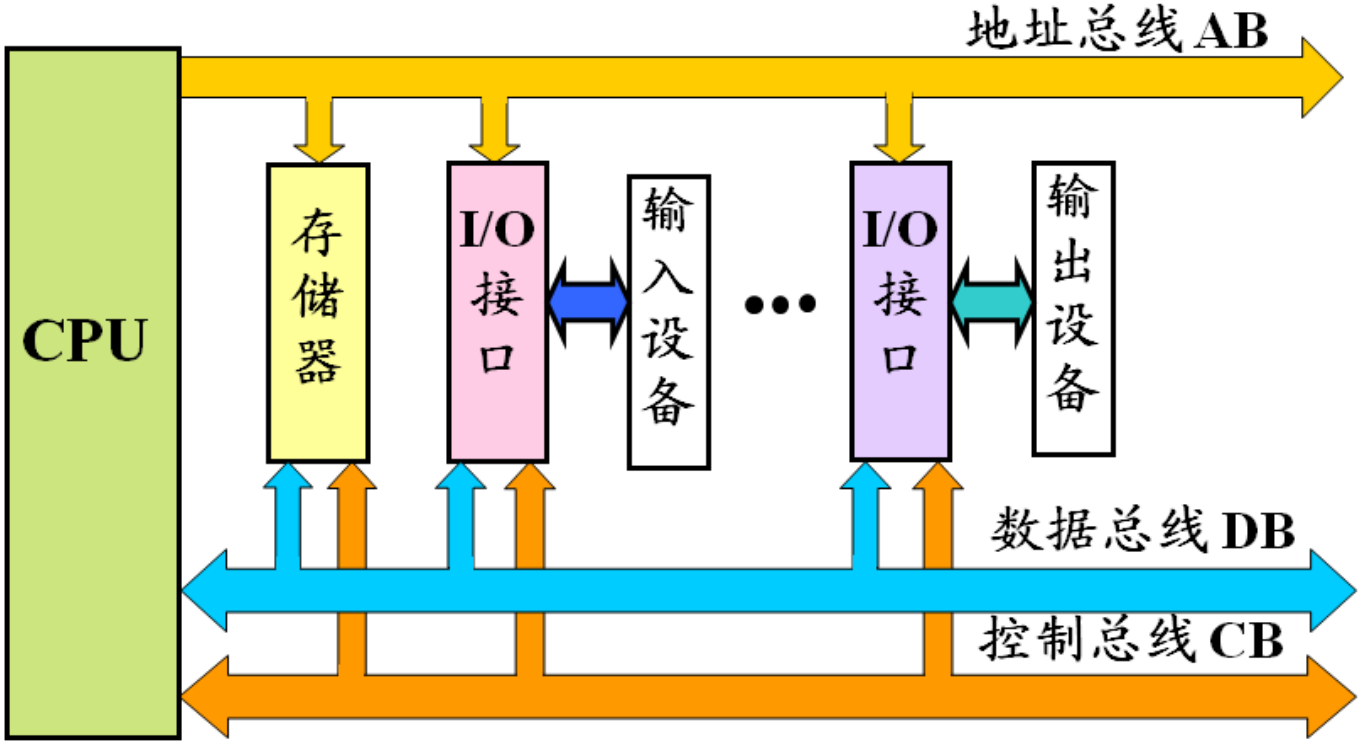
1. 接口和接口技术

2. 常见外设的信号类型、特点和接口电路的作用

1. 接口和接口技术

接口

指CPU、存储器、外设之间通过总线进行连接的电路，
是CPU与外界进行信息交换的桥梁，
用于实现CPU与外设间速度、电平、信息类型等的正确匹配。



接口技术

是研究CPU如何与外部世界在信息类型、通信方式、信号电平、操作时序上进行最佳匹配，实现双方高效、可靠地交换信息的一门技术，是软件、硬件结合的体现，是计算机应用的关键。

外部世界：除CPU以外的所有设备或电路，包括存储器、输入/输出设备等

2、常见外设的信号类型、特点和接口电路的作用

外设是用来实现人机交互的一些机电设备。

外设的信息类型、速度、通信方式与CPU不匹配，不能直接挂在总线上，必须通过接口和系统相连

	CPU	接口作用	外设
信息类型	数字量	模/数转换(A/D) ⇐ ⇒ 数/模转换(D/A)	模拟量
		三态缓冲、锁存	数字量
工作速度	快	解决传送方式	慢
通信方式	并行	串/并转换 ⇐ ⇒ 并/串转换	串行
		三态缓冲、锁存	并行

二、I/O接口电路的典型结构

- 各种接口电路

  并行接口

  定时器

  串行接口

  中断接口

  A/D转换接口

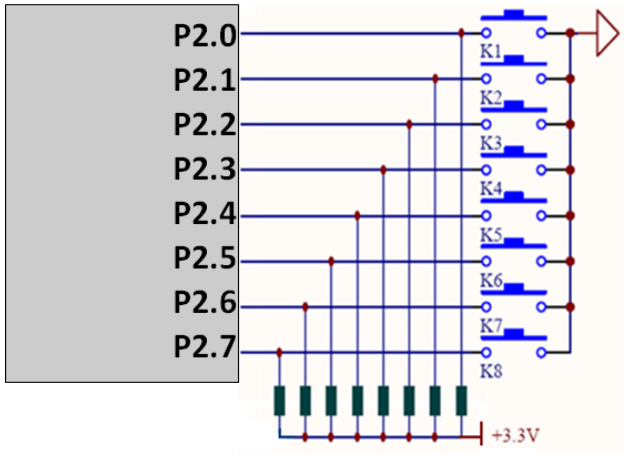
  D/A转换接口

  ……

可编程集成接口电路

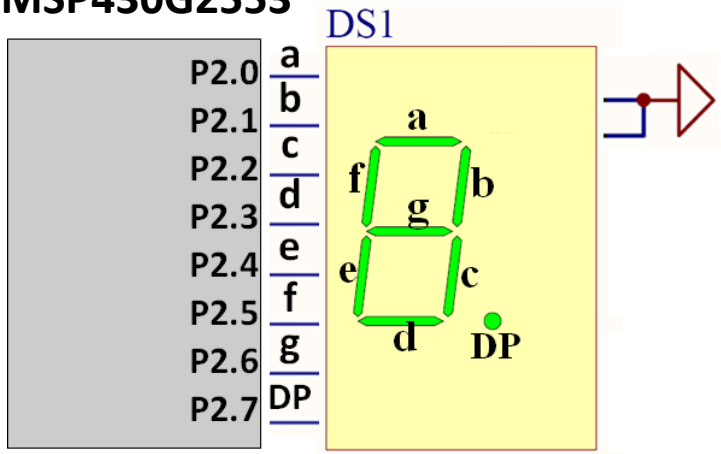
为增加接口电路的灵活性，采用可编程的方式设计接口电路，通过对接口芯片编程，设置接口芯片的工作状态。

MSP430G2553



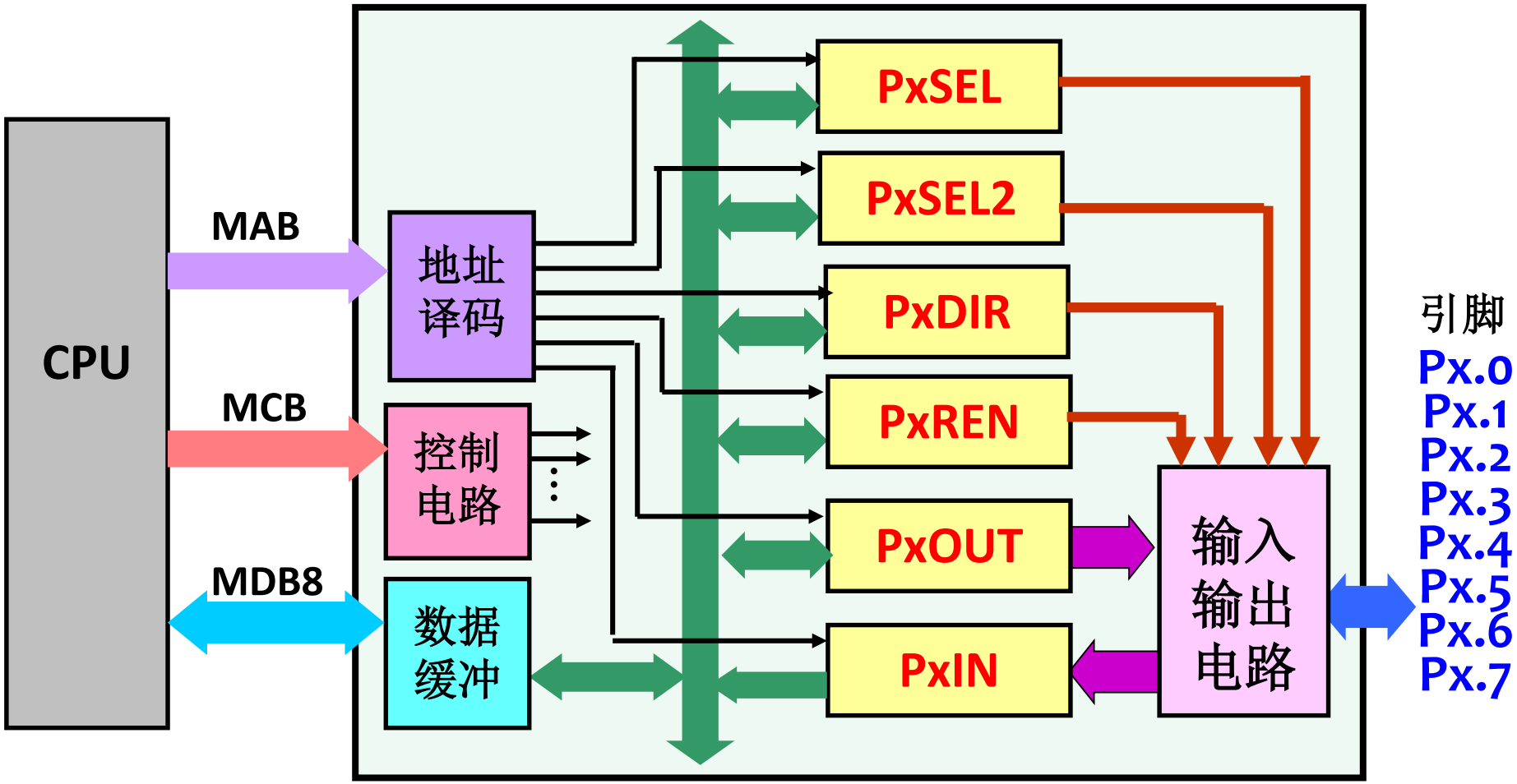

```
P2SEL  =0;    //置P2.0~2为基本I/O
P2SEL2  =0;
P2DIR  =0;    //置P2.0~2为输入
Key=P2IN;     //读入按键状态
```

MSP430G2553

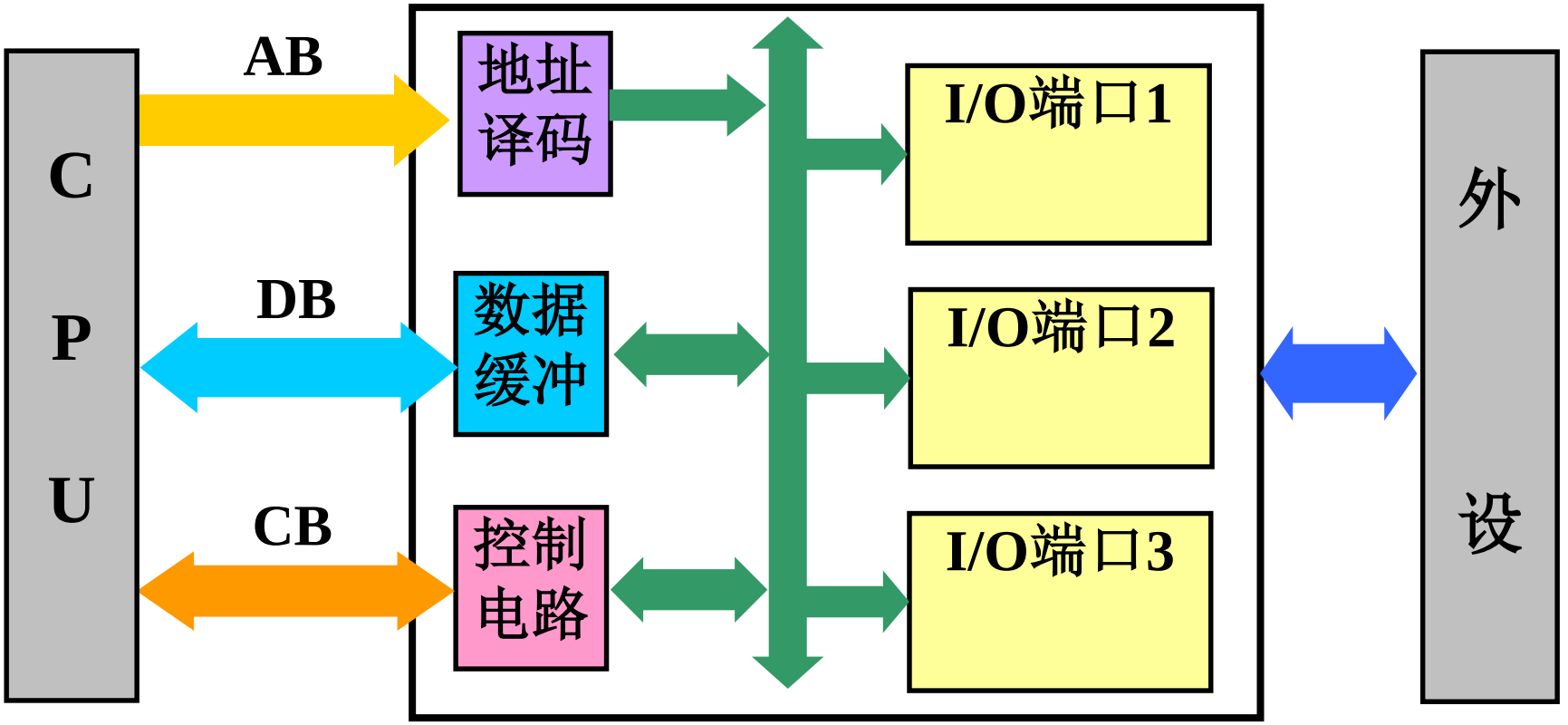



```
P2SEL  =0;       //置P2.0~2为基本I/O
P2SEL2  =0;
P2DIR  =0xFF;    //置P2.0~2为输出
P2OUT=0x3F;      //输出显示信息
```

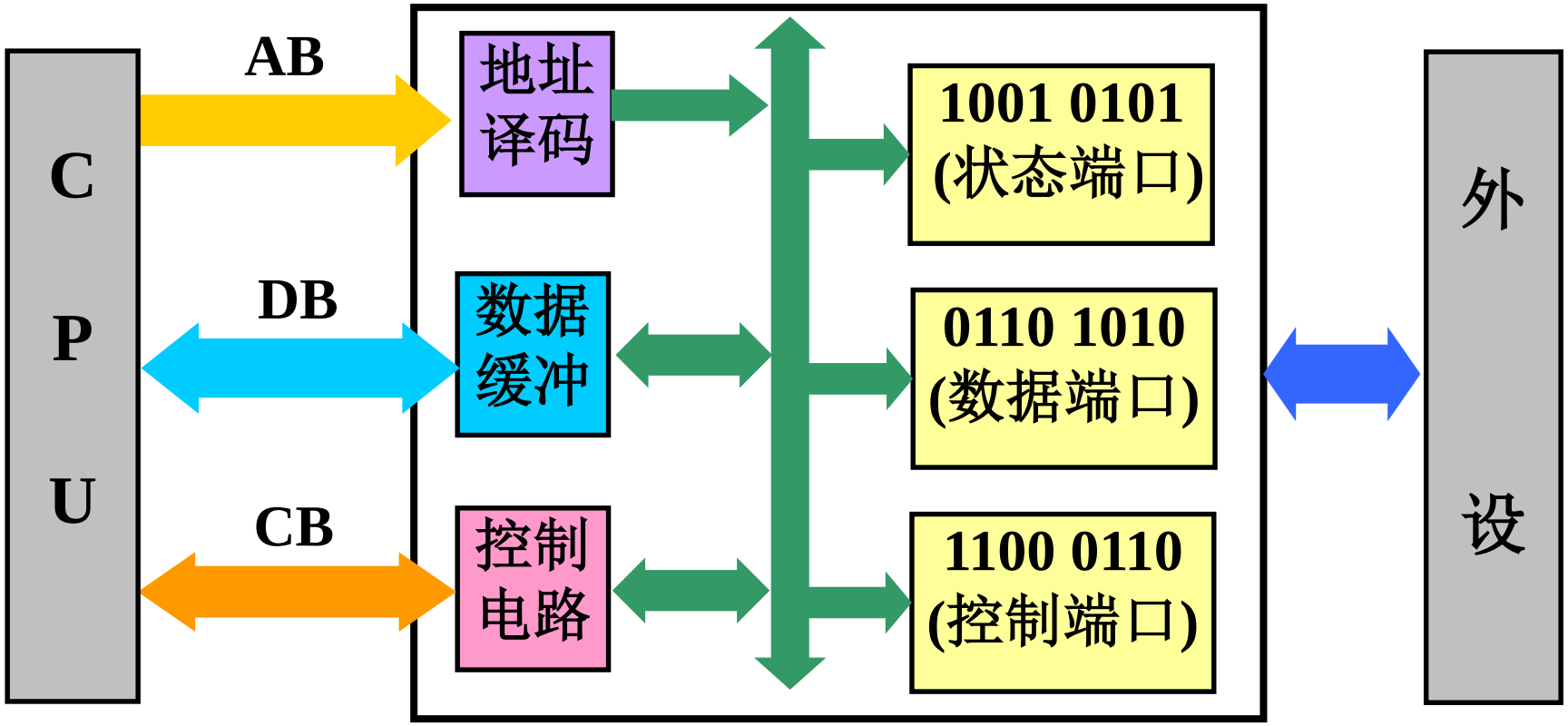
MSP430 I/O端口x 基本输入/输出示意图
(不考虑其他模块功能时，相关编程I/O寄存器)



I/O接口电路的典型结构



- 从编程角度看，接口内部主要包括一个或多个CPU可以进行读/写操作的寄存器，又称为I/O端口。
- 各I/O端口由端口地址区分。



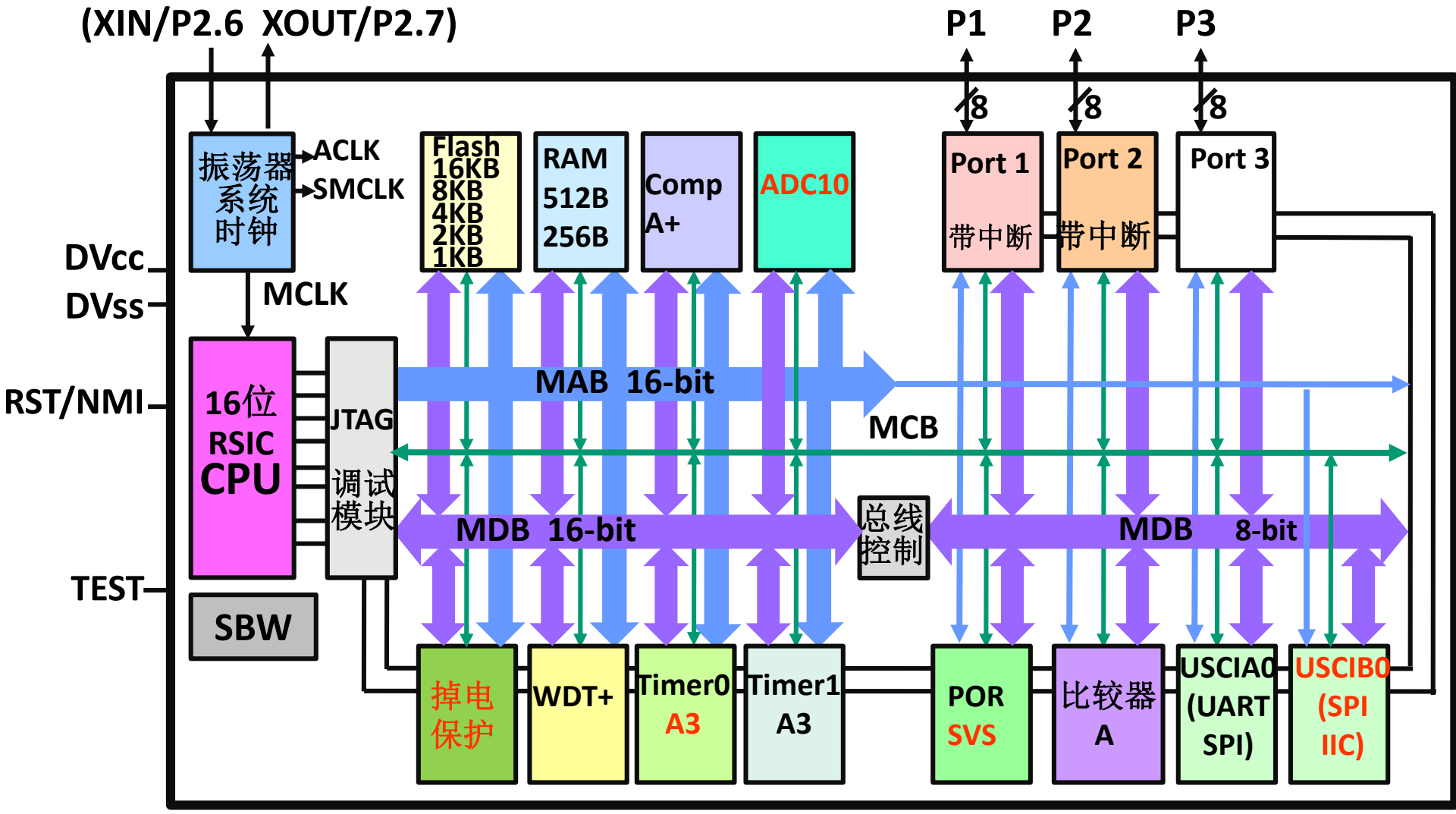
- 按存放信息的不同，I/O端口可分为三种类型

  数据端口：用于存放CPU与外设间传送的数据信息

  状态端口：用于暂存外设的状态信息

  控制端口：用于存放CPU对外设或接口的控制信息，控制外设或接口的工作方式。

MSP430G2553内部各模块含CPU可以操作的I/O寄存器

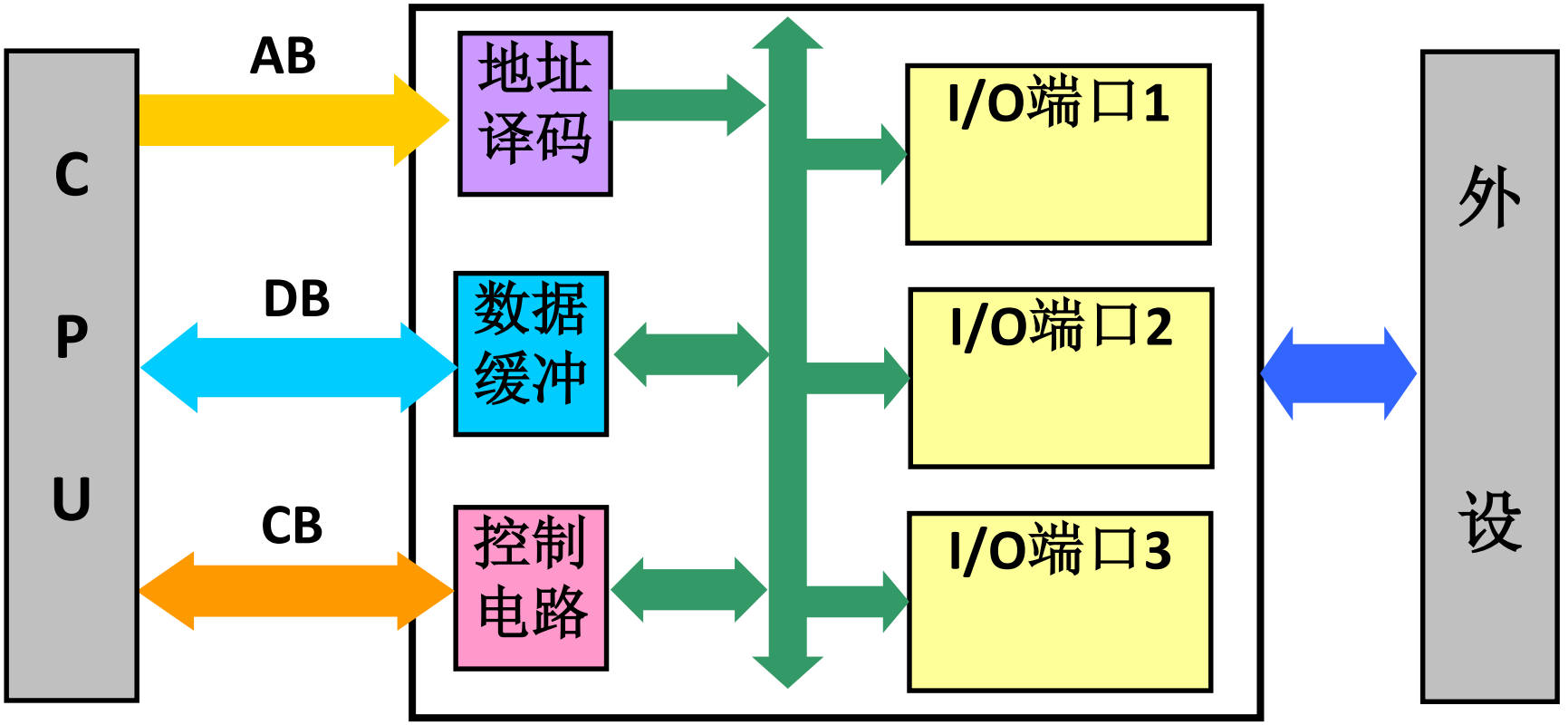


MSP430G2553内部各模块I/O寄存器例

模块	端口名称	端口地址
特殊功能	IE1	000h
	IE2	001h
	IFG1	002h
	IFG2	003h
P1	P1IN	020h
	P1OUT	021h
	P1DIR	022h
	P1IFG	023h
	P1IES	024h
	P1IE	025h
	P1SEL	026h
	P1REN	027h
	P1SEL2	041h

模块	端口名称	端口地址
P2	P2IN	028h
	P2OUT	029h
	P2DIR	02Ah
	P2IFG	02Bh
	P2IES	02Ch
	P2IE	02Dh
	P2SEL	02Eh
	P2REN	02Fh
	P2SEL2	042h
P3	P3IN	018h
	P3OUT	019h
	P3DIR	01Ah
	P3SEL	01Bh
	P3REN	010h
	P3SEL2	043h

模块	端口名称	端口地址
基本时钟	DCOCTL	056h
	BCSCTL1	057h
	BCSCTL2	058h
	BCSCTL3	053h
比较器A	CACTL1	059h
	CACTL2	05Ah
	CAPD	05Bh
ADC10	ADC10AE0	04Ah
	ADC10AE1	04Bh
	ADC10DT0	048h
	ADC10DT1	049h
	ADC10SA	1BCh
	ADC10MEM	1B4h
	ADC10CTL0	1B0h
	ADC10CTL1	1B2h



- CPU对外设输入/输出的控制，

是通过对接口电路中各I/O端口的读/写操作完成。

这些端口如何编址，如何操作?

三、I/O端口的编址

1. 端口与存储器统一编址

2. 端口与存储器分别独立编址

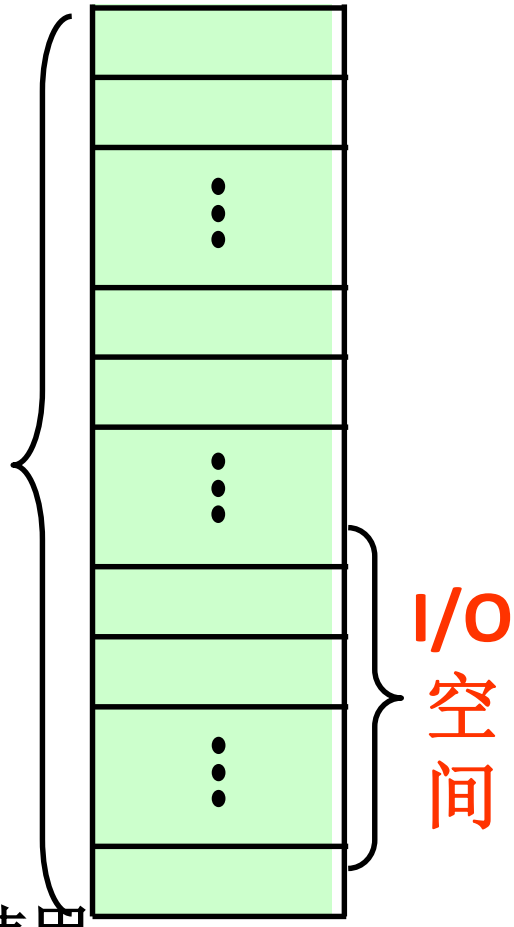
1. 端口与存储器统一编址 (存储器映射方式)

例 MSP430单片机

motorola的M6800系列

瑞萨单片机系列

8051单片机系列



特点:

- I/O端口相当于存储器的一部分，使存储器容量减小
- 对I/O端口的读/写与对存储器的读/写相同，所有可对存储器操作的指令对I/O端口均可使用
- 指令系统中不专设I/O指令，由地址区分存储器还是I/O 端口寄存器

例

```
                &P2IN
MOV.B    &0x028, R4    //读P2IN端口, 0x028是P2IN 的地址
MOV.B    &0x200, R5    //读存储器单元, 0x200是RAM单元的地址
```

2. 端口与存储器分别独立编址
（I/O映射方式）

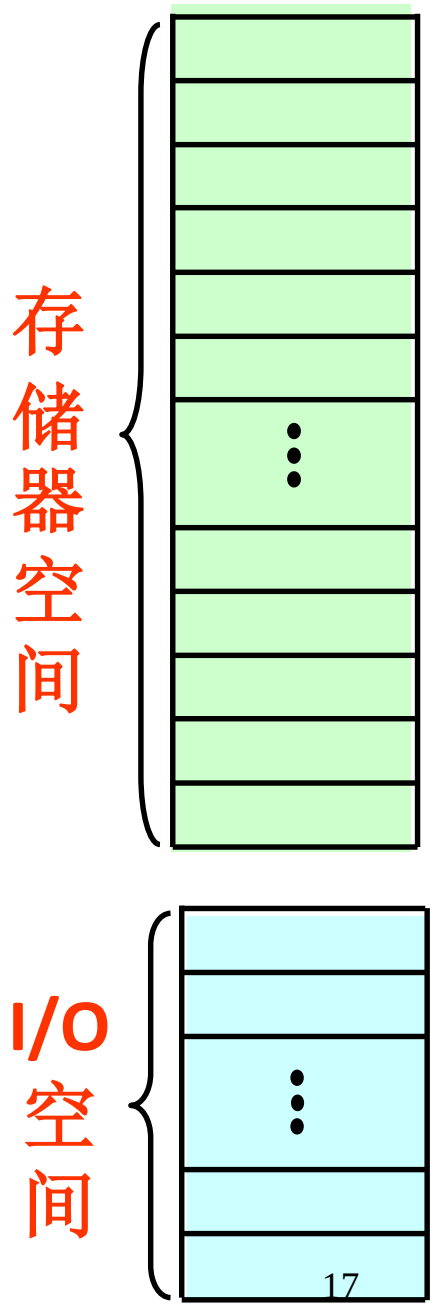
例 Intel的80x86系列、Z80系列

特点：

- 端口与存储器分别独立编址
  端口不占用存储器空间
- 设有专门的 I/O指令对端口进行读写，
  对存储器操作的指令不能用于I/O端口

例 MOV [10H], AL 对存储器操作

OUT 10H, AL 对端口操作



msp430单片机C语言控制端口的方法

(在第3章中已介绍)

- **端口寄存器的写操作**

```
P2DIR&=~ (BIT0+BIT5);          //置P2.0和P2.5为输出
P2OUT |=BIT0+BIT5;             //置P1.0、P1.5为1
```

- **端口寄存器的读操作**

```
unsigned char  data，key;                  //变量定义
data = P1IN;    //读取端口寄存器P1IN的内容赋值给变量
if ((P1IN&BIT2)==0) { ……};     //测试P1IN的P1.2
```

第2节　CPU与外设间的数据传送方式

CPU与外设的工作速度不一致，

如何使两者高效、可靠地进行数据传送，

是本节讨论的问题。

有以下几种传送方式：

一、 无条件传送方式

二、 条件传送方式 (查询方式)

三、 中断传送方式

四、 DMA传送方式

(Direct Memory Access)

一、无条件传送方式 (同步传送方式)

- **实现方法**

  CPU不查询外设工作状态，

  与外设速度的匹配通过在软件上延时完成，

  在程序中直接用I/O指令，完成与外设的数据传送

- **特点**

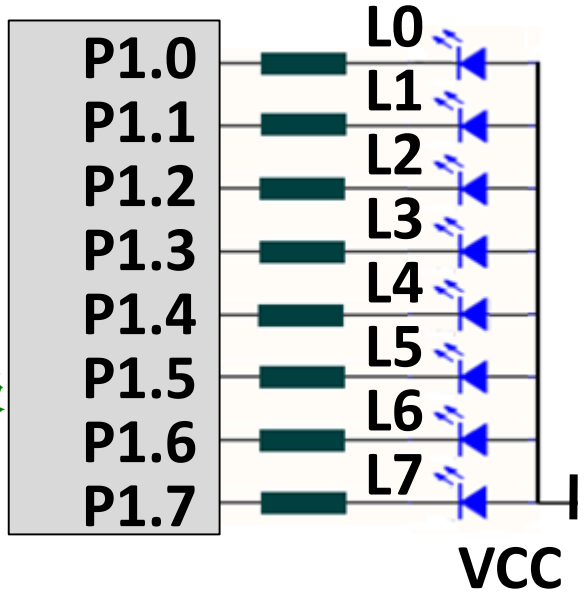
  1. 适用于外设动作时间已知，

  在CPU与外设进行数据传送时，外设保证已准备好的情况

  2. 软硬件十分简单。

第3章 例2中对LED灯的控制采用无条件传送方式

```
#include "msp430.h"
void   delay( );
int    main( void )
{ unsigned char LEDdata[ ]={1,2,4,8,0x10,0x20,0x40,0x80};
  unsigned  int  i;
  WDTCTL = WDTPW + WDTHOLD;  //关闭看门狗
  P1SEL=0;    //设置P2为基本I/O
  P1SEL2=0;
  P1OUT=0xFF;          //使8个LED全灭
  P1DIR=0xFF;          //设置P1为输出端口
  while(1)                    //无限循环
  { for (  i=0; i<8; i++ ) //8个LED依次点亮
    {  P1OUT=~LEDdata[i];  //注意LED是共阳极
       delay();              //调用延时子程
    }
  };
}
void delay( )
{ unsigned  int  i;                  //定义函数变量
  for ( i=0; i<0xffff;  i++ );              //延时
}
```



二、条件传送方式(查询传送方式)

- **实现方法：**

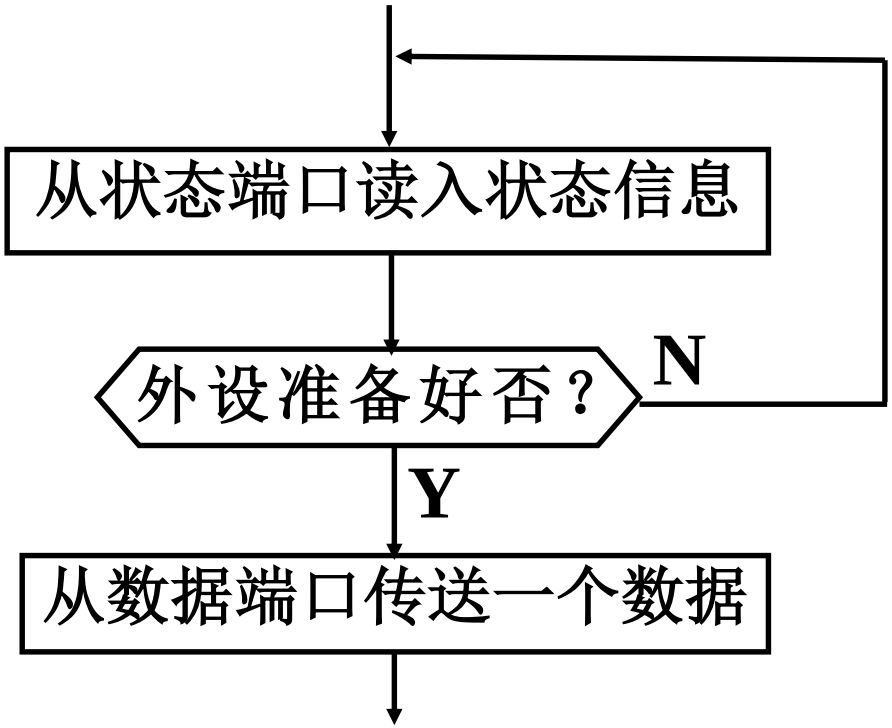
  在与外设进行数据传送前，CPU先查询外设状态，

  当外设准备好后，才执行I/O指令，实现数据传送

- **特点：**

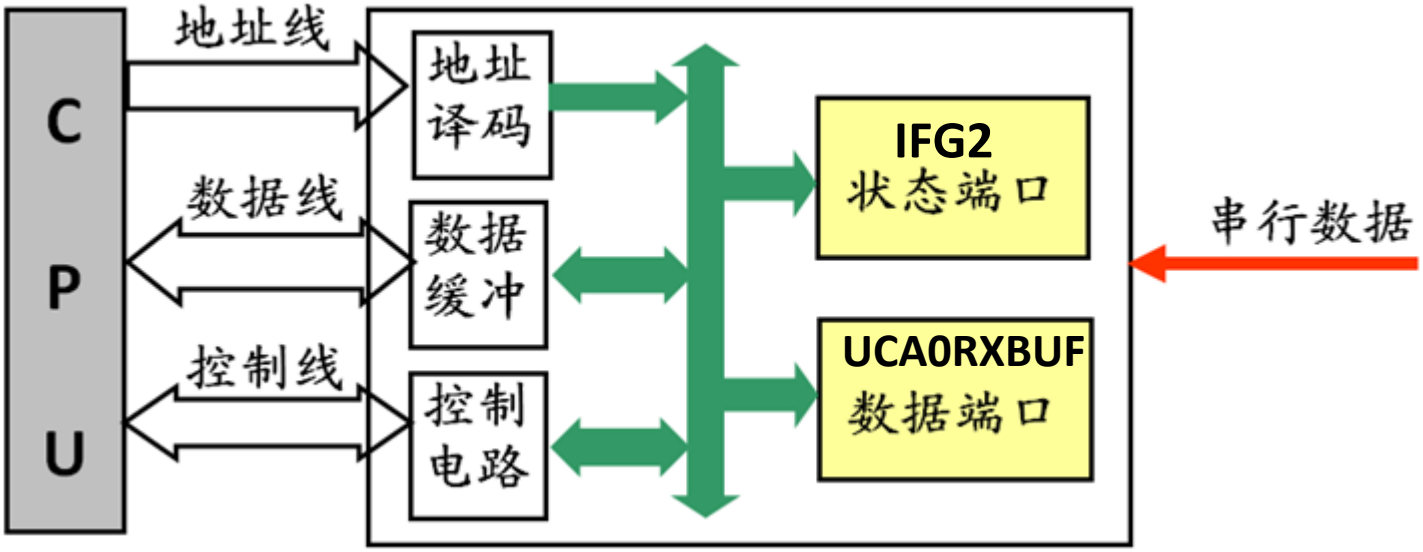
  1. CPU通过不断查询外设状态，实现与外设的速度匹配
  2. CPU的工作效率低

- 查询传送方式，编程流程：



在MSP430的串行接口中，
寄存器IFG2的D0位为1,表示接收到新的数据，
CPU可读取寄存器UCA0RXBUF中的数据，
CPU读取新数据后，将自动清除置寄存器IFG2的D0位为0

	7	6	5	4	3	2	1	0
IFG2					UCB0TXIFG	UCB0RXIFG	UCA0TXIFG	UCA0RXIFG
					rw-1	rw-0	rw-1	rw-0



MSP430的串行接收示意图

编程实现采用查询方式从串行接口接收20个字节数据，
存放到到首地址为buffer的RAM中

```
  ....
  char    buffer[20];
  unsigned  int  j;
  ...
  for (j=0; j<20; j++)
  {while( (IFG2&UCA0RXIFG) == 0 );   //检测接收缓冲器是否满
            buffer[j]= UCA0RXBUF;     //接收一个字符并保存
};
   ...
```

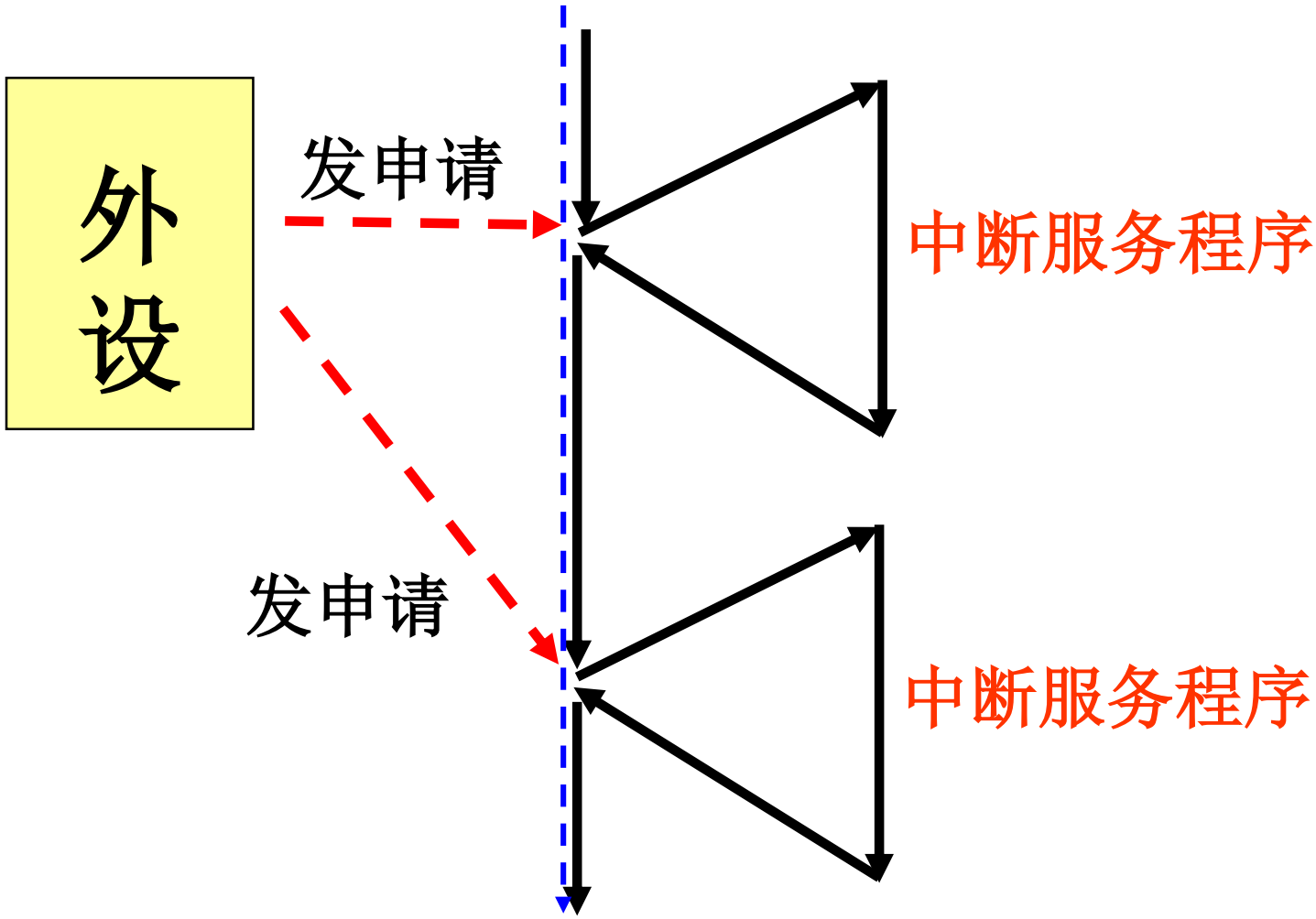
三、中断传送方式

- 实现方法：

1. 当外设准备好，向CPU发出中断请求
2. CPU在满足响应中断的条件下，发出中断响应信号；
3. CPU暂停当前的程序，转 去执行中断服务程序，

   完成与外设的数据传送；
4. CPU从中断服务程序返回，继续执行被中断的程序

中断方式下
CPU执行程序流程



■ 中断传送方式的特点：

1. CPU和外设大部分时间处在并行工作状态，

只在CPU响应外设的中断申请后，

进入数据传送的过程

2. 中断传送方式提高了CPU的效率，

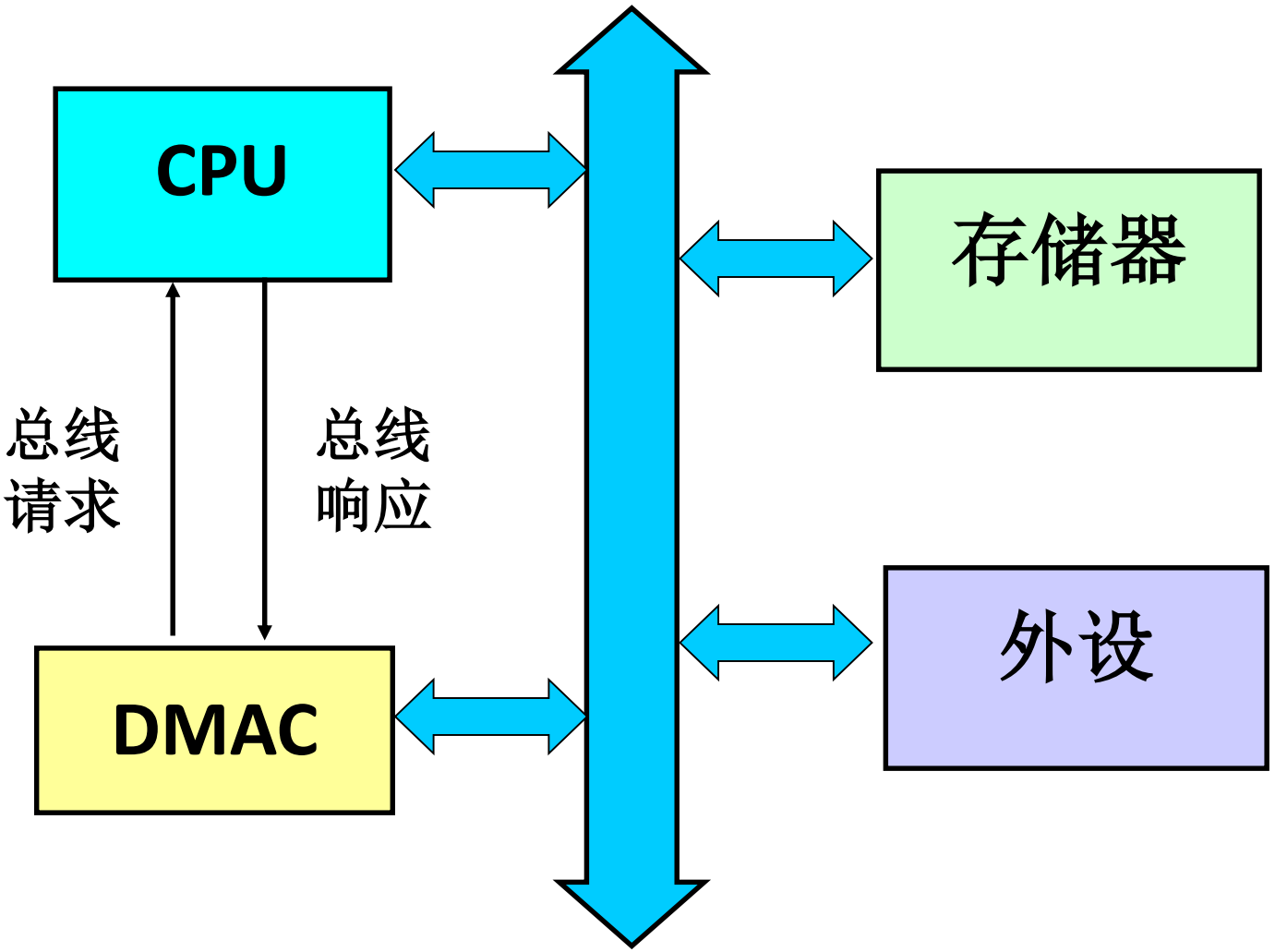
也提高CPU响应外设请求的实时性

四、DMA 传送方式(直接存储器存取方式)

- 实现方法：

1. 由专用接口芯片DMA控制器 (称DMAC) 控制传送过程，
2. 当外设需传送数据时, 通过 DMAC向CPU发出总线请求；
3. CPU发出总线响应信号，释放总线；
4. DMAC接管总线，控制外设、存储器之间直接数据传送

DMA 传送方式过程



■ DMA传送方式的特点

1. 外设和存储器之间，直接进行数据传送，

不通过CPU, 传送效率高。

适用于在存储器与高速外设、

或两个高速外设之间进行大批量数据传送。

2. 电路结构复杂，硬件开销较大。