

高等院校
计算机教材系列

张弥左 王兆月 邢立军 编著

微型计算机接口技术



机械工业出版社
China Machine Press

高等院校
计算机教材系列

张弥左 王兆月 邢立军 编著

微型计算机接口技术



机械工业出版社
China Machine Press

本书系统地介绍微型计算机接口技术及应用。涵盖了输入/输出、DMA技术、系统中断技术及I/O接口技术,还介绍了当前PC机普遍使用的USB接口技术和PCI总线等最新知识。本书内容丰富,面向系统,面向应用,除配有大量的例题和习题外,还列举了微机接口技术在实际应用中的例子。

本书既可以作为非计算机专业的相关教材,也可作为从事计算机应用的工作人员和大专类相关专业学生的参考书。

版权所有,侵权必究。

图书在版编目(CIP)数据

微型计算机接口技术 / 张弥左等编著. — 北京: 机械工业出版社, 2004.8

(高等院校计算机教材系列)

ISBN 7-111-14401-5

I. 微… II. 张… III. 微型计算机-接口-教材 IV. TP364.7

中国版本图书馆CIP数据核字(2004)第041904号

机械工业出版社(北京市西城区百万庄大街22号 邮政编码 100037)

策划编辑: 温莉芳

责任编辑: 华 章

北京中兴印刷有限公司印刷·新华书店北京发行所发行

2004年8月第1版第1次印刷

787mm×1092mm 1/16·22.25印张

印数: 0 001-4 000册

定价: 30.00元

凡购本书,如有倒页、脱页、缺页,由本社发行部调换
本社购书热线:(010) 68326294

前 言

在计算机科学与技术飞速发展的今天,计算机的硬件和软件系统,从类型、数量、功能到完备性等方面,已发展到了前所未有的阶段。计算机接口技术的学习和应用具有特别重要的意义。

计算机软硬件系统的构成,无论哪一类,基本是通过将各种高度专业化、集成化的硬件软件功能模块通过接口技术搭建起来,构成所需目标系统的方法来实现。所以,计算机接口技术已经以单元化、集成化、智能化为特征,发展成为一门体系独立的学科技术。

本书集系统性、知识性、应用性于一体,在内容选材上,力求全面反映微型计算机接口技术的基本内容,并适当增加了接口技术的较新内容,如USB总线及接口、PCI-X总线、现场总线、高级中断控制(APIC)等内容。本书在介绍单元接口功能的基础上,配有相应的例题及大量的思考与练习。

本书在编写方法上,力求由浅入深、通俗易懂。使在学习完本课程后,不但掌握各单元接口的结构和工作原理,而且学会通过硬件接口将处理器与外设系统连接起来,并编写软件驱动程序。

全书共分10章并加上有参考价值的附录。

第1章介绍微型计算机接口技术的概念,包括端口、输入输出方法及相应的基本接口电路。第2章介绍DMA技术原理,并以8237A为例,说明DMA技术的应用。第3章介绍中断系统,并以8259A为例,说明中断的实现方法。第4章全面、系统地介绍了总线的基本概念,着重介绍了目前广泛应用的PCI总线和PCI-X总线。第5、6、8章重点介绍输入输出系统的接口,包括并行通信及接口、串行通信及接口、A/D转换及接口、D/A转换及接口。通过8255A、8253、8250、8251A及各种A/D、D/A等芯片,说明了上述单元接口的实现方法及应用。第7章较系统地描述了通用串行总线(USB)接口,介绍了USB的体系结构、组成、数据流及USB主机、USB设备等。第9章介绍微机系统的常用外围设备,通过键盘、打印机、LED显示及鼠标等,讲述了人-机接口的实现。第10章是微机接口技术课程设计举例,是对前面各章的总结和综合应用。通过6个接口技术课程设计的例子,说明了从任务、要求、方案到接口电路具体实现的设计方法,旨在加深学生对接口技术基本原理的理解,提高学生应用接口技术的能力和培养学生的综合设计能力。本书在单元接口电路后配有12个相应的实验,每个单元实验和6个课程设计均用C语言给出参考程序。

为了提高教学质量,激发学生的学习兴趣,在教学中实现互动式教学,本书配有教师用CAI课件,实验与课程设计的汇编语言和C语言参考程序。请访问网站www.hzbook.com。

在学习本教材前,学生需有计算机组成原理、8086汇编语言及C语言程序设计、基本的模拟电子技术和数字电子技术等基础知识。

本书的第2、3、6章由王兆月编写。第8章中的传感器部分、第9章及书中的程序由邢立军编写、调试。其余各章由张弥左编写。本书在编写过程中得到了张凤祥教授、田心教授及李迎新

教授的热情支持和指导，在此一并表示感谢。由于编者水平所限，书中的错误和不妥之处难以避免，敬请读者指正。

编 者
2004年1月

目 录

前言

第1章 微型计算机接口技术概述	1
1.1 概述	1
1.1.1 接口电路	1
1.1.2 接口的功能	2
1.1.3 接口分类	3
1.2 CPU和输入/输出设备之间的信号	3
1.2.1 数据信息	4
1.2.2 状态信息	4
1.2.3 控制信息	4
1.3 输入/输出端口	5
1.3.1 输入/输出端口的分类	5
1.3.2 输入/输出端口的编址方法	6
1.3.3 I/O端口地址译码	7
1.3.4 CPU的输入/输出指令	8
1.4 CPU和输入输出设备信息交换	9
1.4.1 无条件传送方式	9
1.4.2 查询方式	9
1.4.3 中断控制方式	11
1.4.4 直接存储器存取(DMA)方式	12
1.4.5 输入/输出过程中的几个问题	12
1.4.6 接口与系统的连接	13
1.5 微机接口技术的发展	14
单元实验 基本的输入/输出接口实验	15
思考与练习	16
第2章 DMA控制器	19
2.1 DMA方式	19
2.2 DMA控制器8237A	20
2.2.1 8237A的基本功能和内部结构	20
2.2.2 8237A的工作过程及工作方式	23
2.2.3 8237A的内部寄存器	25
2.2.4 8237A的初始化	30
2.3 DMA控制器8237A应用举例	31
单元实验 DMA实验	36
思考与练习	38

第3章 中断技术	39
3.1 中断及中断处理过程	39
3.1.1 中断源及分类	39
3.1.2 中断处理过程	40
3.2 中断优先权	43
3.2.1 软件排优	43
3.2.2 硬件排优	44
3.3 80X86/Pentium中断系统	45
3.3.1 中断结构	45
3.3.2 中断向量表	47
3.3.3 中断向量表的建立	48
3.3.4 中断响应过程	50
3.3.5 高档微处理器的中断	52
3.4 8259A中断控制器	53
3.4.1 中断控制器8259A	53
3.4.2 8259A的工作方式	56
3.4.3 8259A的初始化编程	60
3.4.4 8259A应用举例	66
3.4.5 高级中断控制器APIC简介	71
3.5 中断程序设计	72
3.5.1 中断程序设计方法	72
3.5.2 硬中断程序设计举例	74
单元实验 8259A中断控制实验	76
思考与练习	77
第4章 总线技术	79
4.1 概述	79
4.1.1 总线标准的基本内容	79
4.1.2 总线分类	80
4.1.3 采用标准总线的优点	81
4.1.4 总线数据传输	81
4.1.5 总线通信方式	82
4.1.6 总线仲裁	84
4.2 局部总线	86
4.2.1 PC总线	86
4.2.2 ISA工业标准总线	86

4.2.3 MCA微通道结构总线	88	6.1.2 串行通信中的数据传送方向	161
4.2.4 EISA扩展的工业标准结构总线	88	6.1.3 串行输入/输出的实现	161
4.2.5 PCI局部总线	89	6.1.4 信号的调制和解调	163
4.2.6 PCI-X总线简介	92	6.1.5 串行通信协议	164
4.3 系统总线	94	6.2 可编程异步通信接口芯片8250	164
4.3.1 MULTI BUS总线	94	6.2.1 8250的结构和功能	164
4.3.2 STD总线	96	6.2.2 8250的引脚功能	170
4.4 通信总线	99	6.2.3 8250的初始化编程	172
4.4.1 IEEE 488并行通信总线	99	6.3 可编程串行通信接口芯片8251A	173
4.4.2 VXI总线	101	6.3.1 8251A的编程结构	173
4.4.3 Centronics总线	102	6.3.2 8251A的工作原理	175
4.4.4 RS-232C串行通信总线	102	6.3.3 8251A的引脚功能	177
4.4.5 RS-499/RS-423/RS-422/RS-485 总线	107	6.3.4 8251A的控制字	181
4.4.6 通用串行总线USB	109	6.3.5 8251A的初始化	183
4.5 现场总线	109	6.4 串行接口应用举例	184
4.5.1 概述	109	单元实验 8251A自发自收实验	186
4.5.2 CAN总线	110	思考与练习	189
思考与练习	116	第7章 USB接口技术	191
第5章 并行接口技术	119	7.1 概述	191
5.1 概述	119	7.2 USB的体系结构	193
5.2 可编程并行接口芯片8255A	120	7.2.1 USB的拓扑结构	193
5.2.1 8255A的内部结构	120	7.2.2 USB的物理接口	193
5.2.2 8255A的引脚及功能	121	7.2.3 USB电源	195
5.2.3 8255A的控制字	122	7.2.4 USB协议	196
5.2.4 8255A的工作方式	124	7.2.5 USB主机和USB设备	196
5.3 8255A应用举例	134	7.3 USB的数据流	198
5.4 可编程定时/计数器8253	139	7.3.1 USB构成	198
5.4.1 8253的内部结构	139	7.3.2 数据传输类型	201
5.4.2 8253的引脚及功能	140	7.4 USB主机(硬件、软件)	203
5.4.3 8253的控制字和初始化编程	141	7.4.1 主控制器功能	203
5.4.4 8253的工作方式	142	7.4.2 软件功能	205
5.5 8253应用举例	147	7.4.3 主控制器驱动程序(HCD)	207
5.5.1 应用举例	147	7.4.4 USB驱动程序(USB D)	208
5.5.2 8254-PIT	149	7.5 USB设备	212
单元实验	150	7.5.1 USB设备状态	213
思考与练习	156	7.5.2 通用USB设备操作	215
第6章 串行接口技术	159	7.5.3 USB设备请求	217
6.1 概述	159	7.5.4 标准设备请求	218
6.1.1 串行通信方式	159	7.5.5 USB设备描述符	223
		7.5.6 USB 2.0简介	228

7.6 IEEE 1394总线	229	9.2.1 键盘	276
7.6.1 IEEE 1394总线的特点	229	9.2.2 键的识别	277
7.6.2 IEEE 1394总线的工作模式	229	9.2.3 微机键盘及接口	283
7.6.3 IEEE 1394总线与USB的性能比较	233	9.2.4 BIOS键盘缓冲区与键盘中断	287
思考与练习	233	9.2.5 小键盘接口设计	288
第8章 数/模、模/数转换及接口	235	9.3 LED显示及其接口	290
8.1 概述	235	9.3.1 LED的工作原理	290
8.1.1 电信号的转换	235	9.3.2 LED显示器显示接口	292
8.1.2 常用传感器	236	9.4 鼠标器及其接口	295
8.2 数/模(D/A)转换及其接口	238	9.4.1 鼠标器的工作原理	295
8.2.1 D/A转换的工作原理	238	9.4.2 鼠标器接口	295
8.2.2 D/A器件的性能参数和术语	239	9.5 打印机及其接口	296
8.2.3 常用D/A转换器及接口	241	9.5.1 打印机工作原理	297
8.3 模/数(A/D)转换及其接口	252	9.5.2 主机与打印机的接口	299
8.3.1 采样和量化	252	单元实验	301
8.3.2 A/D转换的工作原理	253	思考与练习	305
8.3.3 A/D器件的性能参数和术语	256	第10章 接口技术课程设计举例	307
8.3.4 常用A/D转换器及接口	256	10.1 双机间并行通信设计	307
8.4 采样保持器和多路模拟开关	265	10.2 芯片间串行通信设计	309
8.4.1 采样保持器	265	10.3 PC间串行通信设计	312
8.4.2 多路模拟开关	266	10.4 PC与单片机间的通信设计	314
8.5 面对课题选择数/模、模/数芯片的 要点	268	10.5 数据采集板设计	316
单元实验	270	10.6 内窥镜清洗槽温度控制设计	319
思考与练习	273	附录A ASCII码表	323
第9章 常用外围设备及接口	275	附录B 系统中断	325
9.1 常用外围设备概述	275	附录C 总线信号定义	339
9.2 键盘及其接口	276	参考文献	348

第1章 微型计算机接口技术概述

1.1 概述

1.1.1 接口电路

一个实际的微型计算机系统，除了微处理器以外，还必须有各种接口电路。通过接口电路，微处理器可以接收外部设备送来的信息或将信息发送给外部设备。外部设备为什么一定要通过接口电路和主机总线相连呢？能不能将外部设备和CPU的数据总线、地址总线及控制总线直接连接起来呢？

从时序上看，CPU对外部设备的I/O操作和对存储器的读/写操作很类似，那么，是什么原因决定了存储器不需要接口电路，可以直接连在总线上，而输入/输出设备却一定要通过接口电路与总线相连呢？

为了回答上面两个问题，需要分析外部设备的I/O操作和存储器读/写操作之间的不同之处。

所有存储器都是用来保存信息的，功能单一；传送方式也单一，一次必定是传送一个字节或者一个字；品种很有限，只有只读类型和可读/可写类型。此外，存储器的存取速度基本上可以和CPU的工作速度匹配。这些决定了存储器可以通过总线和CPU相连，即通常说的直接将存储器挂在系统总线上。

但是，外部设备的功能却是各种各样的。有些外设作为输入设备，有些外设作为输出设备，也有些外设既作为输入设备又作为输出设备，还有一些外设作为检测设备或控制设备，而每一类设备本身可能又包括了各种工作原理不同的具体设备。对于一个具体设备来说，它所使用的信息可能是数字式的，也可能是模拟式的，而非数字式信号必须经过转换，使其成为对应的数字信号才能送到计算机总线。这种将模拟信号变为数字信号、或者反过来将数字信号变为模拟信号的功能是A/D、D/A接口来完成的。

大多数外部设备所用的信息是数字式的，不过，有些外设的信号是并行的，有些外设的信号是串行的。串行设备必须通过接口电路将串行信号变为并行信号，才能送入CPU；反过来，要将CPU送出的并行信号变为串行信号，才能送给串行设备。可见，接口电路还起到并行数据和串行数据的变换作用。如此说来，如果一个微型机系统中连接的是并行设备，是否可不用接口了呢？也不是的。因为CPU通过总线要和多个外设打交道，而在同一时刻CPU只和一个外设交换信息，就是说一个外设不能长期和CPU相连，只有被CPU选中的外设，才能接收数据总线上的数据或者将外部信息送到数据总线上。所以，即使是并行设备，也同样需要通过接口与总线相连。这就是并行接口要完成的工作。

除了上面这些原因外，外设的工作速度通常比CPU的速度低得多，而且不同的外设工作速度也不相同，这就要求接口电路对输入、输出过程能起一个缓冲和联络的作用。

对于输入设备来说,接口通常起变换和缓冲的作用。变换的含义包括模拟量到数字量的变换、串行数据到并行数据的变换以及电平的变换等。总之,目的是将输入设备送来的信息变换成CPU能接受的格式,并将其放在缓冲器中让CPU来接收。对于输出设备来说,接口要将CPU送来的输出并行数据放到缓冲器中,并将它变成外部设备所需要的形式,这种形式可以是串行的、并行的或模拟量等。

综上所述,I/O接口电路是为了解决计算机与外部设备之间的信息变换问题而提出来的,I/O接口是计算机和外设之间传送信息的桥梁,每个外设都必须通过接口和主机系统相连。接口技术就是专门研究CPU和外设之间的数据传送方式。

1.1.2 接口的功能

简单地讲,一个接口的基本功能是在系统总线和I/O设备之间传输信号,提供缓冲作用以满足接口两边的时序要求。

下面是从广义的角度概括出来的接口功能。对于一个具体的接口来说,未必全部具备这些功能,但必定具备其中的几个。

1. 寻址功能

首先,接口要对选择存储器和输入、输出的信号能够作出解释;此外,要对送来的片选信号进行识别,以便判断当前本接口是否被访问,如果受到访问,还要决定是接口中哪个寄存器或端口受到访问。

2. 输入/输出功能

接口要根据送来的读/写信号决定当前进行的是输入操作还是输出操作,并且随之能从总线上接收来自CPU的数据和控制信息,或者将数据或状态信息送到总线上。

3. 数据转换功能

接口不但要从外设输入数据或者将数据送往外部设备,并且要把CPU输出的并行数据转换成所连的外设可接收的格式(比如串行格式);或者反过来,把从外设输入的信息转换成并行数据送往CPU。

4. 联络功能

当从总线上接收一个数据,或者在把一个数据送到总线上后,接口往往发一个就绪信号通知CPU,数据传输已经完成,从而可以准备进行下一次传输。

5. 中断管理功能

作为中断控制器的接口应该具有发送中断请求信号和接收中断响应信号的功能,而且还有发送中断类型号的功能,此外,如果总线控制逻辑中没有中断优先级管理电路,那么,接口还应该具有优先级管理功能。

6. 复位功能

接口应该能接收复位信号,从而能使接口本身以及所连的外设重新启动。

7. 可编程功能

为了使一个接口可以工作于不同的方式,而且可以用软件来决定到底工作于哪一种方式,并能够用软件来设置控制信号,一个接口应该有可编程功能。当前,几乎所有大规模集成电路

接口芯片都具有这个功能。

8. 错误检测功能

在接口设计中，常常要考虑对错误的检测问题。当前多数可编程接口芯片一般能检测到以下两类错误：

1) 传输错误。由于接口和设备之间的连线常常受噪声干扰，从而引起传输错误，因此一般传输信息时，接口采用奇/偶校验位对传输错误进行检测。传输时，如果用奇校验，那么使信息中1的数目（包括校验位）为奇数。也就是说，所传输的数据中如果1的个数为奇数，则校验位为0，所传输的数据中如果1的个数为偶数，则校验位为1，这样，在传输一个数据时，1的总数目总是为奇数。同样的道理，如果用偶校验，那么，信息中1的数目（包括校验位）为偶数。接口在传输过程中，对信息校验之后，如果发现错误，则对状态寄存器中的相应位进行设置。而状态寄存器的内容可以通过程序进行读取并检测。

2) 覆盖错误。我们知道，当计算机输入数据时，实际上是从接口的输入缓冲寄存器中取数。如果计算机还没有取走数据，输入缓冲寄存器由于某种原因又被装入了新的数据，那么，就会产生一个覆盖错误。在输出时，也会有类似的情况，即输出缓冲寄存器中的数据在被外设取走以前，如果计算机又往接口输出一个新的数据，那么，原来的数据就被覆盖了。在产生覆盖错误时，接口也会在状态寄存器中设置相应的状态位。

1.1.3 接口分类

接口种类的划分方式很多，从其在整个系统中的工作性质和作用，可分为如下几类：

1) 接口按通用性可分为两类：专用接口和通用接口。专用接口即为某种用途或为某类外设而专门设计的接口电路，例如，CRT显示控制器、软磁盘控制器、键盘控制器、DMA控制器。通用接口是可供多种外部设备使用的标准接口，它可以连接各种不同的外设而不必增加附加电路。

2) 接口按可编程性可分为两类：可编程接口和不可编程接口。可编程性是指，在不改动硬件的情况下，用户只要修改初始化程序就可以改变接口的工作方式，大大增加了接口的灵活性和可扩充性。

3) 接口按与外设数据的传送方式可分为并行接口和串行接口两种。并行接口与外设间的数据传送按字长传送（即8位或16位二进制数同时传送），串行接口即接口与外设之间的数据传送是按位（一个二进制位）传送的。

4) 接口按工作对象可分为面向CPU的外围接口和面向外设的I/O接口两类：面向CPU的外围接口只能和CPU系统配套使用，以增强CPU的性能。例如，中断控制器可以提高CPU的中断控制能力，总线仲裁控制器可以提高CPU的总线控制能力。面向外设的输入/输出接口的工作对象是针对不同的外设，不同的外设可以通过此类接口连接到当前的微机系统中。例如，开关量可通过简单的锁存器连接到微机系统中，而可编程的并行接口可将一个微型打印机接到微机系统中。

1.2 CPU和输入/输出设备之间的信号

为了说明CPU和外设之间的数据传送方式，需先了解CPU和I/O设备之间的信号分类，通常，

CPU和I/O设备之间有几类信号。

1.2.1 数据信息

CPU和外部设备交换的基本信息就是数据。数据通常为8位、16位或32位。数据信息大致分为如下三种类型。

1. 数字量

这类是指由键盘、磁盘机、卡片机等读入的信息，或者主机送给打印机、磁盘机、显示器及绘图仪的信息，它们是二进制形式的数据或是以ASCII码表示的数据及字符，通常是8位的。

2. 模拟量

如果一个微型机系统是用于控制的，那么，多数情况下的输入信息就是现场的连续变化的物理量，如温度、湿度、位移、压力、流量等，这些物理量一般通过传感器先变成电压或电流，再经过放大。这样的电压或电流仍然是连续变化的模拟量，而计算机无法直接接收和处理模拟量，要经过模拟量往数字量（A/D）的转换，变成数字量，才能送入计算机。反过来，计算机输出的数字量要经过数字量往模拟量（D/A）的转换，变成模拟量，才能控制现场。

3. 开关量

开关量可表示两个状态，如开关的闭合和断开、电机的运转和停工、阀门的打开和关闭等，这样的量只要用一位二进制数（0、1）表示就可以了。

上面这些数据信息，一般是由外设通过接口传递给系统的。在输入过程中，数据信息由外设经过外设和接口之间的数据线进入接口，再到达系统的数据总线，从而送给CPU。在输出过程中，数据信息从CPU经过数据总线进入接口，再通过接口和外设之间的数据线送到外设。

1.2.2 状态信息

状态信息反映了当前外设所处的工作状态，是外设通过接口往CPU传送的。对于输入设备来说，通常用准备好（READY）信号来表明输入的数据是否准备就绪；对于输出设备来说，通常用忙（BUSY）信号表示输出设备是否处于空闲状态，如为空闲状态，则可接收CPU送来的信息，否则CPU要等待。

1.2.3 控制信息

控制信息是CPU通过接口传送给外设的，CPU通过发送控制信息控制外设的工作。如外设的启动信号和停止信号就是常见的控制信息。实际上，控制信息往往随着外设的具体工作原理不同而含义不同。

从信息含义上说，数据信息、状态信息和控制信息各不相同，应该分别传送。但在微型计算机系统中，CPU通过接口和外设交换信息时，只有输入指令（IN）和输出指令（OUT），所以，状态信息、控制信息也被广义地看成是一种数据信息。即状态信息作为一种输入数据，而控制信息作为一种输出数据。这样，状态信息和控制信息也通过数据总线来传送。但在接口中这三种信息进入不同的寄存器。具体地说，CPU送往外设的数据或者外设送往CPU的数据放在接口的数据缓冲器中，从外设送往CPU的状态信息放在接口的状态寄存器中，而CPU送往外设的控

制信息要送到接口的控制寄存器中。

1.3 输入/输出端口

由图1-1可见,一个接口电路中包含有一组寄存器。数据传送时,不同的信息进入不同的寄存器。通常把输入/输出(I/O)接口电路中能被CPU直接访问的寄存器或某些特定部件称为I/O端口。CPU通过这些端口发出命令,读取状态或传送数据。

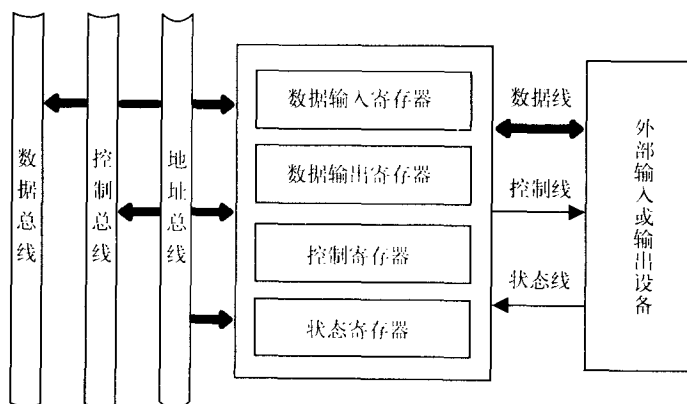


图1-1 接口电路的典型结构

1.3.1 输入/输出端口的分类

在接口电路中,按端口寄存器存放信息的物理意义来分,端口可分为3类:数据端口、状态端口和控制端口。

1. 数据端口

数据端口存放数据信息,通常为8位或16位。在输入过程中,数据信息由外设经过接口电路中的数据端口、到达系统的数据总线,送给CPU。在输出过程中,数据信息从CPU经过数据总线进入接口电路中的数据端口,再通过接口和外设间的数据线送到外设。

2. 状态端口

状态端口存放状态信息,即反映外设当前工作状态的信息,CPU可读取这些信息,查询外设当前的工作情况。对于输入接口电路,状态信息应能反映输入数据是否准备好;对于输出接口电路,状态信息应能反映输出设备的忙、闲状态。

3. 控制端口

控制端口存放控制信息,控制信息是CPU通过接口传送给外设的,以控制外设工作,如控制输入输出装置启动或停止等。对于可编程接口电路,控制信息还负责选择可编程接口芯片的工作方式等。

状态信息、控制信息与数据信息是不同性质的信息,必须要分别传送。但在大部分微型计算机中,只有输入指令(IN)和输出指令(OUT)。因此,状态信息和控制信息也被广义地看成是一种数据信息,即状态信息作为一种输入数据,而控制信息作为一种输出数据,这样,状态信息和控制信息也通过数据总线来传送。为了区别输入数据和状态信息,数据端口和状态端口

必须有不同的端口地址,为了区别输出数据和控制信息,数据端口和控制端口也必须有不同的端口地址。所以一个接口电路往往若干个端口地址,CPU寻址的是端口寄存器,而不是笼统地寻址外设接口电路。

1.3.2 输入/输出端口的编址方法

由I/O端口的含义可知,CPU对外设的输入、输出操作。实际上归结为对接口电路中I/O端口的读/写操作。对端口的操作也是有所不同的,有的端口只能读,有的端口只能写,有的端口既可读又可写。一般来说,状态端口只能读出,不能写入,是只读端口;控制端口只能写入不能读出,是只写端口;数据端口通常包含有输入端口和输出端口,数据输入端口是只读端口,数据输出端口是只写端口,所以,数据端口是一个既可读又可写的端口。

为了CPU对每个端口的访问,系统给每个I/O端口均赋予一个地址,称为端口地址。CPU要访问接口中的某一端口,需先将端口地址放入地址总线,用高位地址经地址译码选中该接口芯片,用低位地址选择具体要访问的端口,在读写命令的配合下(实际是执行输入输出指令)实现数据的传送。

为便于CPU的访问,每个端口均有一个端口地址,那么,系统如何来为每个端口安排端口地址呢?这就是I/O端口的寻址方式问题。通常有两种编址方式,一种是把I/O端口地址与存储器地址统一编址,即采用存储器映像方式;另一种是将I/O端口地址和存储器地址分开独立编址,即采用I/O映像方式。

1. 统一编址

所谓统一编址就是从存储器空间划出一部分地址空间给I/O设备。把I/O接口中的每一个端口当作存储单元一样进行访问,每一个端口占用一个存储单元地址,该编址方式有三个特点:

1) 由于I/O端口和存储器共用同一地址空间,可不设专门的I/O指令,凡对存储器可使用的指令(即所有访内指令)均可用于对端口的访问。

2) 由于访内指令类型多,功能齐全,用这些指令访问I/O设备,不仅使输入输出操作灵活、方便,而且还可直接对端口内容进行算术逻辑运算、移位等,还可给I/O端口有较大的编址空间,这对大型控制系统和数据通信系统是很有实际意义的。

3) CPU如何确定是访问存储器还是访外设端口,是通过用地址总线最高位为0或1来作为寻址标志。例如,设存储空间为64K,当 A_{15} 为1时,32K地址空间为I/O端口地址空间;而 A_{15} 为0时,32K地址空间为存储器地址空间,选用地址最高位作为I/O寻址标志目的是便于软件控制,当然也可以指定整个空间中某一部分用于I/O。

统一编址的缺点:端口占用了一定的存储空间,使存储器可寻址空间减小,此外由于用访内指令访问I/O端口,而访内指令功能强,执行时间较长,降低了数据交换速度。

2. 独立编址

独立编址,就是I/O端口地址空间不占用存储器地址空间,二者的地址空间相互独立完全分开,采用独立编址的特点是:

1) 需要用专门的I/O指令(IN、OUT)访问对端口进行操作,而I/O指令短,执行速度快,而且I/O指令和访内指令有明显的区别,从而使程序中I/O操作部分和其他操作部分层次清楚,便

于阅读和理解。

2) 要使用专门的信号来区分CPU是访问存储器还是访问I/O端口,在8086中用 $M/\overline{IO}$ 区分、8088中用 $\overline{M}/IO$ 区分;而且读写存储器与读写I/O端口使用分开的两组读/写信号,如在8086/8088中的最小模式下,用 $M/\overline{IO}$ 和 $\overline{RD}$ 、 $\overline{WR}$ 信号构成两组读写控制信号;在最大模式下用完全独立的两组读/写信号: I/O端口读/写用 $\overline{IOR}$ 和 $\overline{IOW}$,存储器读/写用 $\overline{MEMR}$ 和 $\overline{MEMW}$ 。

独立编址方式的缺点: 由于I/O指令功能简单、类型少,使I/O操作受到限制,另外由于要用专门的信号区分是访问存储器还是I/O端口,要求CPU设置专门的引脚信号,即要增加CPU的引脚信号。

1.3.3 I/O端口地址译码

在进行I/O操作时,CPU必须首先确定与自己交换信息的I/O端口(寄存器),那么通过什么方法把系统地址总线上的端口地址变为CPU所需要的I/O端口(寄存器)呢?这就是I/O端口地址译码问题。

I/O端口地址译码与存储器空间译码相似,也包括片选和字选(端口选择)两部分。片选有片外译码电路实现,用来确定接口芯片内部端口的地址范围。字选由芯片内部的地址译码电路实现,用来确定接口芯片内部端口的具体地址。一般字选问题已经由芯片内部解决,用户在设计I/O地址译码电路时只需要考虑片选译码电路。实现片选的方法有三种:全译码法、部分译码法和线选法。

- 全译码是指地址总线的所有地址线均参与地址译码,其中一部分参与片选,另一部分参与字选。全译码的优点是每一个端口都有惟一的地址,但结构较复杂。这种方法一般用在I/O空间较大的场合。
- 线选法是指系统的地址总线中只有少数几个地址线经过简单的逻辑或将其中的某个地址线直接作为接口芯片的片选信号。没有参加译码的地址信号不论为0或1都不会影响片选结果,因此一个端口可能有多个地址(即地址重叠)。线选法的优点是结构简单,但造成严重的地址空间浪费和地址重叠。这种方法一般用在I/O空间较小的场合。
- 部分译码法是最常用的片选方法,是前两种方法的结合,具有结构简单的优点,但它也存在地址重叠,适用于I/O空间较大的场合。

全译码、部分译码的一般做法是用地址的高位与控制信号组合经译码电路产生I/O接口芯片的片选信号($\overline{CS}$),实现接口芯片间的选择;用地址的低位部分直接连到I/O接口芯片的端口选择端实现具体端口的选择。

1. I/O端口地址译码电路

译码电路的输入信号由于CPU对I/O端口进行操作时,除了用地址信息来选中端口外,还需要相应的控制信号决定操作方式、数据流向。因此,I/O地址译码电路不仅仅与地址信号有关,还与相应的控制信号有关,它对地址和控制信号进行组合,产生对接口芯片的选择信号。所以,I/O地址译码器的输入,除地址信号外。还有控制信号,常用的控制信号有 $\overline{IOW}$ 、 $\overline{IOR}$ 、AEN(控制非DMA传送)等,这些信号也一起参加译码。

此外,地址译码电路的输出信号通常是低电平有效,如3—8译码器(74LS138)的8个输出

均为低电平有效，即低电平为选中，高电平为未选中。

2. 译码电路形式

常见的I/O端口地址译码电路有固定式端口译码电路（包括门电路译码、译码器译码）；开关式可选端口译码及比较器译码法等。下面仅对固定式端口译码作简单说明。

固定式端口地址译码是指接口中所用的端口地址是事先确定好的，不能改变，目前大部分接口卡中采用固定式译码，在固定式译码中，当只需用单个口地址时，一般用逻辑门电路译码较方便，如图1-2所示。电路可译出2E7H写操作端口地址。

当接口电路中需要使用多个端口地址时，采用译码器译码较方便，译码器的型号很多，常用是3—8译码器74LS138，可从其三个代码输入（A、B、C）端输入地址，从输出端输出8个端口地址，如图1-3所示。

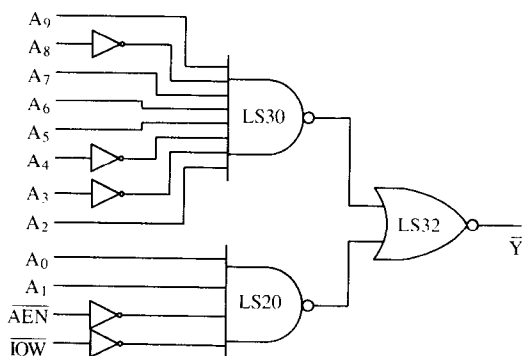


图1-2 固定译码电路

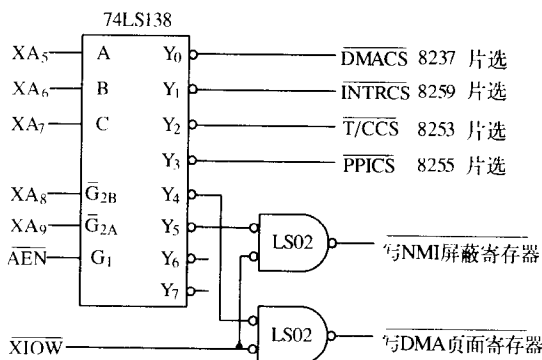


图1-3 译码器组成的译码电路

1.3.4 CPU的输入/输出指令

CPU的输入/输出指令是指独立编制的I/O指令。8086CPU在AL或AX与接口之间进行数据传送，I/O端口的寻址包括直接寻址和DX间接寻址两种。直接寻址是指仅用低8位地址线A₇~A₀译码产生I/O端口地址（8位），而A₁₅~A₈的输出为0；DX寄存器间接寻址，用A₁₅~A₀地址线译码产生I/O端口地址（16位）。CPU访问I/O端口时，地址线A₁₉~A₁₆都输出低电平。

1. 直接寻址输入/输出指令（可寻址范围是0~255，即256个端口地址）

IN	AL, n	; 8位端口地址，字节输入
IN	AX, n	; 16位端口地址，字输入
OUT	n, AL	; 8位端口地址，字节输出
OUT	n, AX	; 16位端口地址，字输出

2. DX寄存器间接寻址输入/输出指令（可寻址范围是0~65535，即64K个端口地址）

IN	AL, DX	; 8位端口地址，字节输入
IN	AX, DX	; 16位端口地址，字输入
OUT	DX, AL	; 8位端口地址，字节输出
OUT	DX, AX	; 16位端口地址，字输出

3. 80286和80386、80486的输入/输出指令

```
MOV  DX, PORT
LES  DI, BUFFER IN
```

```
INSB                                ; 8位传送
或 INSW                             ; 16位传送
MOV  PORT, DX
LDS  SI, BUFFER OUT
OUTSB                                ; 8位传送
或 OUTSW                             ; 16位传送
```

这里的输入/输出是直接对内存的RAM而言。输入时，用ES: DI指向RAM中目标缓冲区BUFFER IN；输出时，用DS: SI指向RAM中的源缓冲区BUFFER OUT。

1.4 CPU和输入输出设备信息交换

CPU与输入/输出设备的信息交换有无条件传送方式、查询方式、中断控制方式及存储器直接存取（DMA）方式。

1.4.1 无条件传送方式

无条件传送方式的特点是：假设输入接口数据已经准备好，或者假设输出设备是空闲的，此时，CPU无需查询状态，直接用IN或OUT指令完成与接口之间的数据传送。采用无条件方式的接口电路如图1-4和图1-5所示。

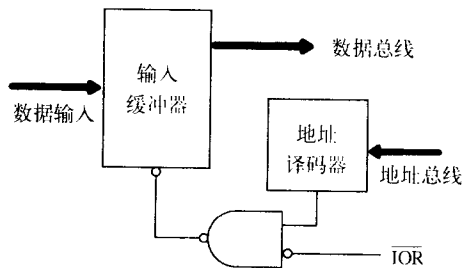


图1-4 无条件传送输入电路

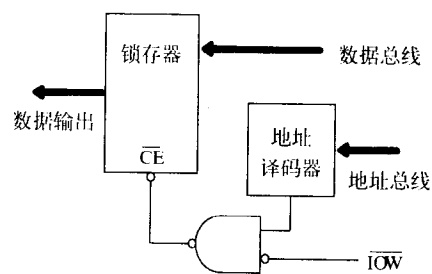


图1-5 无条件传送输出电路

输入时，由于数据保持时间相对于CPU的处理时间要长，输入端可直接用输入缓冲器与CPU的数据总线相连。当CPU执行输入指令时，I/O读信号 $\overline{IOR}$ 有效，来自输入设备的数据到达数据总线，传送给CPU。显然，CPU在执行输入指令时，要求外设的数据已经准备好，否则就会出错。

输出时，由于外设速度较慢，要求接口具有锁存能力，即要求CPU送给外设的数据应该在接口中保持一段时间。CPU执行输出指令时，I/O写信号 $\overline{IOW}$ 有效，CPU输出的信息经过数据总线进入输出锁存器，输出锁存器保持这个数据，直到外设取走。显然，CPU在执行输出指令时，必须保证锁存器是空闲的。

1.4.2 查询方式

在用查询方式接收数据之前，CPU要查询输入数据是否准备好；在用查询方式输出数据之前，CPU要查询输出设备是否空闲。只有确认外设已具备了传送条件后，才能用IN和OUT指令完成数据传送。与无条件传送方式相比，采用查询方式的接口电路要复杂一些，因为它要设置供CPU查询的部分电路。