



机电一体化丛书

微计算机 控制机电设备集锦

李 华 编

36

1.1/1

電子工業出版社

微计算机控制机电设备集锦

(原理及实例)

李 华 编

电子工业出版社

0021360

内 容 提 要

本书是应用微计算机控制机电设备的原理和实例的总结。全书共四章。第一、二章系统地介绍了一位机原理和应用实例；第三、四章概要地介绍了八位机原理和实例分析。

本书从控制机电设备的需要出发，总结性地介绍了微计算机的内部结构、外部特性、接口器件在实例分析中，既有顺控又有数控，既有单机控制又有群控，其控制对象既有各类机床又有象电梯、机械手等。

本书可供从事机电一体化产品的设计、制造和旧设备改造的工程技术人员和工人阅读，也可供大专院校有关专业作教学用书。

JS352/12

微计算机控制机电设备集锦

(原理及实例)

李 华 编

责任编辑：邓又强

电子工业出版社出版(北京海淀区万寿路)

电子工业出版社发行 各地新华书店经销

山东电子工业印刷厂印刷(淄博市周村)

开本：787×1092毫米 1/16 印张：19.25，字数：456千字

1988年10月第一版 1988年10月第一次印刷

印数：1—7000册 定价：6.00元

ISBN 7-5053-0477-1/TP·64

编 者 的 话

《微计算机控制机电设备集锦》包括两部分。前两章介绍用于开关量控制的一位机；后两章介绍6800和Z80等常用的八位机。每一部分既讲基本原理，又有应用实例。

编者从机电设备控制等方面的实用出发，对一位机与八位机，对三种八位微处理器的外部特性及内部结构，对6800与Z80的指令系统和中断功能，对PIA和PIO两种并行输入/输出接口器件等均做了比较，归纳出它们的共性和特点，便于读者掌握和选用。

编者力求做到由浅入深、由简到繁、由易到难，并体现微机软件与硬件相结合等特点。在应用实例中既有顺控又有数控，既有单机控制又有群控，其控制对象既有机床又有其它机械设备。这些典型实例的设计思路、编程方法和系统构成不仅适于机电设备控制，对其它方面的应用也有参考价值。

书中部分内容是编者学习和应用微型计算机的总结。为满足教学需要，还参考、整理或引用了一些专家的论文著作，从中得到不少启示，特此致谢！

本书可供厂矿企业有一定基础并从事机电一体化产品的设计、制造或机电设备技术改造的工程技术人员和工人阅读，也可供大专院校有关专业作教学用书。为便于自学，本书起点较低，重点和难点叙述详细，并附有大量图表。

因编者水平有限、时间仓促，书中一定存在不少缺点甚至错误，敬请读者批评指正。

李 华

于河北工学院机械一系

目 录

第一章 一位机原理	1
1-1 一位机基本结构及工作概况	1
一、开关量控制对电脑的基本要求	1
二、一位机基本结构	2
三、一位机的工作概况	3
1-2 一位微处理器	4
一、一位微处理器的外部引线	4
二、一位微处理器的内部结构	4
1-3 一位机基本指令	7
一、一位机指令概述	7
二、输入指令	8
三、输出指令	10
四、运算类指令	13
五、输入/输出控制指令	29
六、标志指令	36
七、跳步指令	37
1-4 一位单板机	38
一、复位/启动部分	39
二、程序计数与存储部分	43
三、输入/输出部分	48
1-5 一位机编程举例	56
1-6 一位机系统设计	73
一、一位机基本系统	73
二、系统设计的一般考虑	74
三、输入/输出扩展	81
四、无条件转移	88
五、子程序调用	91
六、立即数传送	94
七、输入/输出接口	95
1-7 怎样提高系统可靠性	98
一、硬件措施	98
二、软件措施	102
第二章 一位机应用实例	107
2-1 水泵控制	107
2-2 动力头控制	111
一、动作要求	111

二、软件封锁方式	112
三、硬件转移方式	114
四、软硬混合方式	117
2-3 车床控制	118
一、控制对象	118
二、I型刀架控制	119
三、II型刀架控制	122
四、按循环方式划分模块的控制方案(方案一)	126
五、按控制对象划分模块的控制方案(方案二)	132
六、四线输入、动作判断的控制方案(方案三)	134
七、系统设计的几个问题	136
2-4 铣床控制	137
一、控制对象的动作要求	137
二、控制算法及程序设计	138
三、控制系统结构	153
2-5 钻床数控	158
一、加工实例	158
二、设计思想	159
三、定位与钻孔	160
四、主流程	163
五、程序编制	165
六、系统构成及特点	168
2-6 洗衣机控制	169
一、控制要求及硬件特点	169
二、软件编制	170
2-7 机械手控制	172
一、控制要求	173
二、硬件特点	173
三、程序设计	175
第三章 八位机概要	181
3-1 三种常用的八位微处理器	181
一、外部引线	181
二、内部结构	184
三、技术指标	190
四、指令系统	191
3-2 中断系统	203
一、6800中断系统	203
二、Z80中断系统	204
3-3 输入/输出接口	207
一、6820 PIA(外围接口适配器)	207
二、Z80-PIO(并行输入/输出)	210
三、计数/定时器(CTC)	216

第四章 八位机应用实例	224
4-1 微电脑顺控器	224
一、半自动专用铣床的顺序控制	224
二、有延时功能的顺控方案	227
4-2 微电脑控制电梯	228
一、电梯的控制要求	228
二、硬件系统	229
三、软件设计	232
4-3 微电脑控制步进电机	234
一、概述	234
二、硬件系统	235
三、软件设计	238
四、Z80 控制步进电机的通用程序	247
4-4 数控插补软件	253
一、逐点比较法的一种方案	253
二、直接函数运算法	256
三、任意二次曲线插补法	260
四、三次曲线的直接插补	265
五、四次曲线插补	269
六、高次曲线的插补速度	274
4-5 微电脑数控车床	275
一、控制对象与微型电脑	275
二、大功率步进电机的升降速控制软件化	277
三、主程序流程	278
4-6 微电脑控制曲轴磨床	279
一、系统结构及特点	280
二、程序模块	282
4-7 微电脑数控线切割机床	284
一、6800 数控线切割机	284
二、Z80 数控线切割机	289
三、8080 群控线切割机	293
参考资料	300

第一章 一位机原理

1-1 一位机基本结构及工作概况

目前微电脑一方面向集成度高、位数多、功能强，适于多种用途的方向发展；另一方面则是以 MC 14500 系列为代表的一位机。MC14500 系列产品是美国 Motorola 公司继八位机 6800 之后于 1977 年底首先推出的。它的结构简单、容易制作、价格便宜、使用方便，特别适于作逻辑判断、运算或处理，即开关量的控制。

一、开关量控制对电脑的基本要求

开关量指的是可以用一位二进制数“0”或“1”表示的两种状态或动作。例如：电平的高与低，按钮或行程开关的接通与断开，电磁铁或接触器线圈的得电与失电，灯的亮与灭，刀架或工作台的进与退，电动机的启动与停止，等等。对输入的一位数据进行逻辑判断、运算或处理，然后输出一位运算结果以控制机械中执行件的动作，称为开关量控制。电脑控制机械的示意如图 1-1 所示。

在控制系统中，电脑是机械的控制器，机械为电脑的控制对象。机械中的发令件（如按钮或行程开关等）状态以电平高或低的开关量形式输入电脑，经电脑处理后，又以开关量的形式输出给机械，控制机械中的执行件（如电磁铁或接触器等）动作。在控制过程中，有时要将电脑处理的中间结果作为暂存信号送回输入端；也有的输出信号既要控制机械的动作，又要作为自身的反馈信号。电脑对机械的控制是按照预先编制的程序进行的。电脑在整个系统中是一个可编程序控制器。

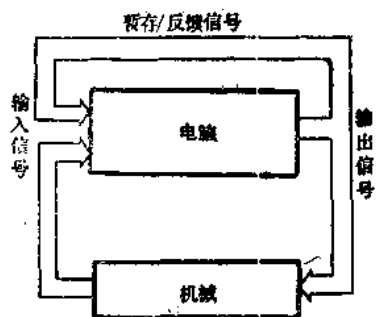


图 1-1 电脑控制机械示意

由上所述，不难看出开关量控制对电脑的基本要求为：

(1) 电脑应该具有程序计数与存储功能，以存储预先编制的控制程序，并由程序计数器 (PC) 按照规定的顺序给出指令地址。这样才能逐条地取出指令送给有关部件。

(2) 电脑应该具有多个输入端和多个输出端。输入端的数目主要取决于机械中发令件的个数，输出端的数目主要取决于机械中执行件的个数。此外，有一些输入端或输出端将被暂存或反馈信号占用。为了从众多的输入/输出端中选取某一个开关量作为当前的输入来源或输出目的，各输入/输出端有相应的地址（编号）。各输出端还应有锁存作用，以便记住当前的运算结果。

(3) 电脑应该具有对一位二进制数（开关量）进行逻辑判断、运算或处理的功能，以及控制输入/输出的作用。

(4) 电脑应该具有可扩展性,即可以根据实际需要增加存储容量和输入/输出端数。

这四个基本要求就决定了主要用于开关量控制的一位机的基本结构。

二、一位机基本结构

一位微电脑主要由核心部分(微处理器及复位/启动线路)、程序计数与存储部分(程序计数器与存储器)和输入/输出部分(输入选择器与输出锁存器),共三大部分组成,如图 1-2 所示。

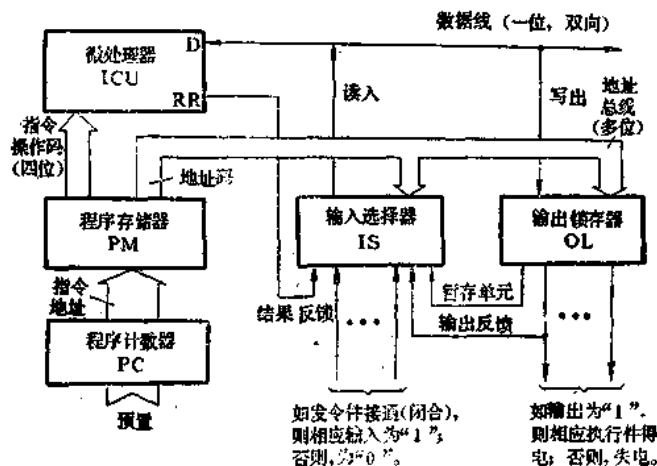


图 1-2 一位机基本结构

图 1-2 中未画复位/启动线路,控制信号、接口线路及电源等。

一位机的微处理器包括运算器、控制器和寄存器等,这与多位的微处理器类似;还包括产生时钟脉冲信号的振荡器,但不包括程序计数器,这与许多微处理器不同。它是在工业上进行开关量控制的核心器件,也叫工业控制单元 (Industrial Control Unit, 缩写作 ICU)。

该系统中的存储器仅用作存放控制程序(指令序列),因此称为程序存储器(Program Memory, 缩写作 PM)。一位机的程序存储器可以是“可擦洗、可编程的只读存储器”(EPROM),也可以是“随机(读/写)存储器”(RAM)。程序计数器(Program Counter, 缩写作 PC)在 ICU 发出的时钟脉冲信号控制下作加 1 计数,不断地向程序存储器提供当前所要执行的指令地址。为了实现程序转移,PC 的内容是可以预置的。

一位机 ICU 的输入量来自输入选择器 (Input Selector, 缩写作 IS)。它有多个数据输入端,可分别接各输入信号;但只有一个数据输出端,经数据线接 ICU 的数据端 D。由地址总线给定不同的输入地址,就可以把选中的输入信号“读”进微处理器。可见,输入选择器实际上是一个受地址控制的多路开关。一位机 ICU 的输出量送至输出锁存器 (Output Latch, 缩写作 OL)。它有一个数据输入端,也经数据线接 ICU 的数据端 D;但有多可存放数据的输出端,分别接各受控元件或暂存开关量。由地址总线给定不同的输出地址,微处理器就可以把输出信号“写”入相应的输出端,以便暂存信息或对机械设备进行有效的控制。

一般指令均由操作码和地址码两部分组成。程序存储器应将其操作码送入微处理器，以便产生相应的控制信号；而将地址码送至有关器件，以便完成规定的操作。显然，操作码线和地址码线都是由程序存储器发出的多位、单向线。微处理器的数据端 D 既可接收来自输入选择器的一位数据，又可向输出锁存器发送一位数据。因此，数据线是一位、双向的，微处理器的处理结果还可反馈回输入选择器的某一输入端，以便作为后续指令的操作数。

此外，为了使后续指令可以用以前的暂存量或输出量作为操作数，应该将这些量接到输入选择器的某些输入端，构成暂存单元或输出反馈。

在上述五种主要器件中，程序存储器是一般微电脑通用的。微处理器 ICU、程序计数器 PC、输入选择器 IS 和输出锁存器 OL 是一位机专用的，均为 14500 系列的配套器件。为了进一步理解一位机这五种基本器件在系统中的地位、作用及其相互关系，下面阐明一位机的工作概况。

三、一位机的工作概况

1. 开机前的准备工作

以上述五种器件为主，构成一位机的基本系统，实现对机械设备的控制，至少需要进行两项准备工作。一项是根据控制要求，用机器语言编写控制程序，并注入存储器。一般从第 0 单元起始存放程序，每个单元存放一条指令。另一项则是输入/输出接口。

2. 复位状态

一位机启动前处于复位状态。此时，微处理器 (ICU) 内部的振荡器停振、寄存器清零；ICU 外部的程序计数器 (PC) 及输出锁存器 (OL) 的各输出端均为“0”态。

3. 启动-运行状态

一位机从复位状态启动。初始时，程序计数器 PC 为全“0”，即指向程序存储器 PM 的第 0 单元，使微处理器 (ICU) 从第一条指令开始取指令并执行程序。接着，时钟脉冲使程序计数器 PC 的内容加 1，即指向程序存储器 PM 的第 1 单元，微处理器就执行第二条指令……依此类推，直到执行“转移”指令才改变这种顺序，若到“停机”指令才停止执行程序。

4. 取数、运算和存数指令的功能

在微处理器 ICU 内部除有运算器外，还有一个存放指令执行结果的寄存器，叫结果寄存器 (Result Register, 缩写作 RR)。一位机正常执行“取数”指令时，就将该指令地址码所选中的输入端信息取到 ICU 的 RR 中。一位机正常执行“运算”指令时，就将所选中的输入端信息与 RR 的原有内容作逻辑运算，然后把运算结果存入 RR。一位机正常执行“存数”指令时，就将 RR 的信息存到该指令地址码所选中的输出端。

5. 输入/输出控制

正常执行取数、运算等与输入信号有关的输入型指令，要求微处理器呈“开输入”状态。否则，当“关输入”时，将封锁“1”信号的输入。

正常执行存数等与输出信号有关的输出型指令，要求微处理器呈“开输出”状态。否则，当“关输出”时，将封锁任何信号的输出。

借助于关输入或关输出，可实现等效的条件转移，这是一位机的重要功能，将在

2-2 和 2-3 节作详细介绍。

当一位机执行“转移指令”时，就将转移地址置入 PC，打破按 PC+1 的执行顺序，实现无条件转移。当执行“停机指令”时，就使一位机复位，为再启动作好准备。

总之，一位机的基本结构及工作概况完全符合开关量控制对电脑的基本要求。但是，要想得心应手地使用它，发挥它的功能和长处，首先要全面掌握一位机的微处理器 (ICU) 及其指令系统。

1-2 一位微处理器

一、一位微处理器的外部引线

一位微处理器 (ICU) 的型号是 14500，其外形和引线如图 1-3 所示。

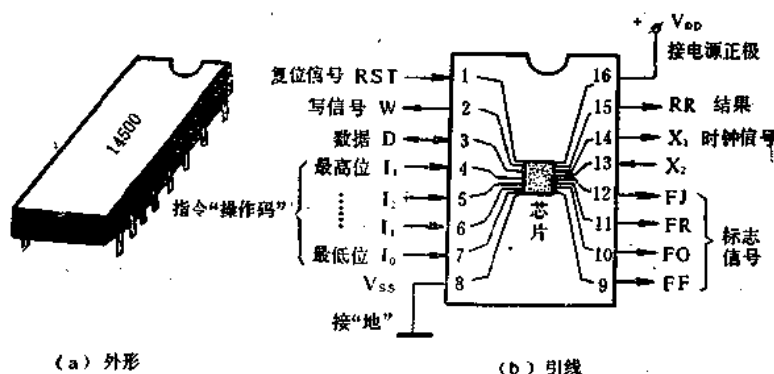


图 1-3 一位微处理器的外形和引线

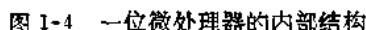
该器件共有十六条引线，分别与芯片相接。其编号由 1 到 16 是按照外壳缺口朝上，从左上角 1 起反时针排列至右上角 16 止。其中第 16 和第 8 引线分别接电源的正、负极。第 4、5、6、7 引线分别接程序存储器的“操作码”输出线 I_3 、 I_2 、 I_1 、 I_0 ，每一条指令的四位操作码都是按照这样的顺序输入微处理器的，因此引线的箭头向里。第 8 引线是数据线 D，微处理器可由此端输入或输出数据，输入选择器 IS 和输出锁存器 OL 均挂在这条数据线上，因此引线的箭头是双向的。第 15 引线是微处理内部“结果寄存器”RR 内容的输出端，箭头向外。此外，第 14 和 13 引线接微处理器内部的振荡器， X_1 为内部时钟信号输出端（箭头向外）， X_2 为外部时钟信号输入端（箭头向里）。第 1 引线为复位信号 RST 输入端。第 2 引线为写信号 W 输出端。第 9~12 为标志信号输出端，这几条引线信号的作用将在后面详述。

二、一位微处理器的内部结构

一位微处理器主要由运算部分、控制部分及寄存器等组成，其内部结构示意图如图 1-4。

一位微处理器的运算部分由逻辑单元 LU、结果寄存器 RR、输入允许寄存器 IENR、输出允许寄存器 OENR、多路器 MUX 及门电路组成。

结果寄存器 RR 是一个一位累加器，用来存放逻辑单元 LU 的运算结果，它的数据



逻辑单元 LU 是一位微处理器运算部分的核心。仅能进行逻辑运算，因此叫逻辑单元。它有两个输入端 (a, b) 和一个输出端 (c)。ICU 的外部数据可经第 3 引线 (D 端) 和与门 1 从 a 端输入 LU。LU 在指令的控制下，对由 a、b 端输入的两个一位开关量进行相应的逻辑运算，然后由 c 端输出一位结果，有关逻辑单元 LU 的详细工作情况，将结合指令系统介绍。

多路器 (Multiplex, 缩写作 MUX) 的输出端 f 有三种状态, 即“1”、“0”和浮置状态。浮置状态即高阻抗状态。当微处理器执行输出型指令时, 其操作命令可分别控制 MUX 将它 d 端输入的量 (即结果寄存器内容的原码) 或 e 端输入的量 (即结果寄存器内容的反码) 由 f 端输出至内部数据线, 以便经 ICU 的第 3 引线 (D 端) 往外传送。在执行其它指令时, MUX 的输出端 f 呈高阻抗状态, 不影响 ICU 外部的数据经数据线 D 的输入。

在一位微处理器的内部还有两组十分重要的“使能”(Enable)或允许电路。一组叫输入允许电路,由输入允许寄存器(Input Enable Register, 缩写作 IENR)和与门 1 组成;另一组叫输出允许电路,由输出允许寄存器(Output Enable Register, 缩写作 OENR)和与门 2 组成。这两个寄存器均是 D 型触发器,可被相应的指令置为“1”或“0”态。当 IENR=1 (即 IENR 的 Q 端为高电平)时,打开与门 1, ICU 呈开输入状态,允许 ICU 外部的数据经数据线 D 从 a 端输入逻辑单元 LU。“输入允许”就是使 ICU 的外部数据“1”能输入逻辑单元 LU 的意思。如 IENR=0 时,则关闭与门 1, ICU 呈关输入状态,故封锁(或禁止)ICU 外部的数据“1”输入逻辑单元 LU。请注意:上述“开输入”或“关输入”状态,只是对 ICU 外部数据“1”的输入而言, IENR 的状态对 ICU 外部数据“0”的输入无影响。换句话说, ICU “关输入”

并非逻辑单元 LU 的 a 端无输入, 而是不管外部的数据为“0”或“1”, 该端输入一律为“0”。

当 $OENR = 1$ (即输出允许寄存器的 Q 端为高电平) 时, 打开与门 2。在输出型指令的控制下有写脉冲从 ICU 的第 2 引线 (W 端) 输出, 故允许 ICU 内部的数据 (即结果寄存器内容的原码 RR 或反码 $\overline{RR}$) 经数据线 D 写入输出锁存器 OL 的某一输出端。结果寄存器的内容写入输出锁存器叫“写出”操作, 此时 ICU 呈开输出状态。“输出允许”就是使 ICU 的内部数据 (“1”或“0”) 能由输出锁存器 OL 输出的意思。如 $OENR = 0$ (即输出允许寄存器的 Q 端为低电平) 时, 则关闭与门 2, 即使执行输出型指令也无写脉冲从 ICU 的第 2 引线 (W 端) 输出, 故封锁 (或禁止) ICU 内部的数据 (RR 或 $\overline{RR}$) 经数据线 D 写入输出锁存器 OL 的某一输出端。此时, 结果寄存器的内容虽然也可以经 MUX 传送到数据线上, 但不能由输出锁存器 OL 输出。ICU 呈关输出状态, 仅有由结果寄存器至数据线的的数据传送, 没有由数据线至输出锁存器的“写出”操作, 在逻辑上等效于未执行输出型指令。一位机正是利用这个特性来实现程序中有条件的等效转移。

一位微处理器的控制部分主要由指令“操作码”的寄存、译码及控制逻辑等部件组成。程序存储器 PM 将指令的四位操作码从 ICU 的第 4、5、6、7 引线, 按照由最高位 I_3 到最低位 I_0 的顺序输入其控制部分, 经译码后产生相应的控制信号 (也叫操作命令)。这些操作命令分别加到有关部位, 实现该指令所规定的操作。

此外, 在 ICU 内部还设有环形自激振荡器 (OSC) 也叫时钟信号发生器, 其原理如图 1-5 所示。在 ICU 的第 13、14 引线外接一个电阻 R_X , 该振荡器即可产生时钟信号 X_1 , 其时钟频率 f_{CP} 由外接电阻调定。外接电阻 R_X 增大, 时钟频率 f_{CP} 降低。 R_X 可在十千欧到数百千欧之间选择。 f_{CP} 的常用范围是 $1\text{MHz} \sim 100\text{kHz}$ 左右。

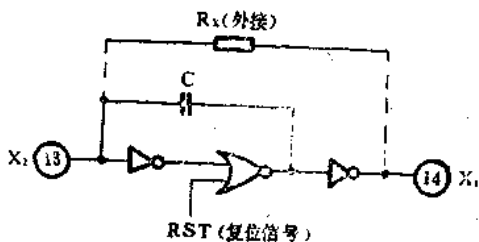


图 1-5 振荡器原理

时钟信号 X_1 除供 ICU 内部定时使用外, 还可由 ICU 的第 14 引线输出, 用作系统同步 (如程序计数器 PC 加 1 和输入/输出控制等)。如 ICU 使用从第 13 引线输入

的外部时钟信号 X_2 工作, 则 X_1 和 X_2 两端的外接电阻 R_X 不接。

当由 ICU 第 1 引线输入的复位信号 RST 为高电平 (“1”态) 时, 经整形后可使 ICU 内部的所有寄存器清零, 结果寄存器 RR 的输出端 (第 15 引线) 和各标志信号 FJ 、 FR 、 FO 、 FF 的输出端 (第 12、11、10、9 引线) 均为低电平 (“0”态)。此时, 内部时钟停振, X_1 停在高电平 (“1”态)。这就是 ICU 的复位状态。当 RST 回到低电平 (“0”态), 稍待延迟后振荡器 (OSC) 重新起振, ICU 开始工作。这就是 ICU 的启动状态。在 ICU 内部, RST 的输入端处有一只数百千欧的接地电阻 (图 1-4 未画), 以保证该输入端平时为低电平 (“0”态)。其后的整形电路, 对输入干扰信号有较好的抑制作用。

图 1-4 中的标志寄存器, 由四只 D 型触发器组成。当 ICU 执行标志指令时, 在控制信号的作用下, 标志寄存器可分别从第 12、11、10、9 引线输出相应的标志脉冲信

号, 以实现指令和功能的扩展, 或作外部控制信号用。

在图 1-4 左下角方框中的跳步触发器, 其作用是当 ICU 执行跳步指令时, 控制程序实现跳步, 即跳过下一条指令。

1-3 一位机基本指令

一、一位机指令概述

一位微处理器有四个操作码输入端 ($I_3 \sim I_0$), 即一位机指令的操作码由四位二进制数组成。如用一个字节(存储单元)来存放一条指令, 则其高四位为操作码, 低四位为操作数的地址码 ($A_3 \sim A_0$, 简称地址码)。四位二进制的地址码可以区分 0000~1111 共十六个输入/输出。这就是一位机指令的基本格式, 用横式表示为:

	高四位				低四位			
位号:	7	6	5	4	3	2	1	0
内容:	操作码				地址码			

因此, 我们称输入/输出各有十六个的一位机为基本系统。

一位机指令大致分为三类, 即输入/输出类、逻辑运算类和控制类。

输入/输出类指令是微处理器 ICU 内部的结果寄存器 (RR) 与输入选择器 IS 的某输入口 (I_i), 或与输出锁存器 OL 的某输出口 (O_i) 之间进行信息传输的指令。

逻辑运算类简称运算类指令。二元布尔运算共 16 种, 但一位机只能对一位的操作数进行运算。所以适合一位机的只有其中的九种运算, 即: 置“0”、置“1”、“与”、“或”、“非”、“与非”、“或非”、“同或”及“异或”。这九种运算可以分别制定为一位机指令。但为了简化集成电路, 也可以有选择地设计其中几种, 其余的由这几种运算组合实现。例如: 有了“或反”指令, 将结果寄存器 RR 的内容“或”其“反”码 $\overline{RR}$ 再送 RR ($RR + \overline{RR} \rightarrow RR$), 可实现 RR 置“1”。如有“与反”指令, 将结果寄存器 RR 的内容“与”其“反”码 $\overline{RR}$ 再送 RR ($RR \cdot \overline{RR} \rightarrow RR$), 可实现 RR 置“0”。

“与非”运算可由先“与”后“非”运算实现, 即 $\overline{A \cdot B} \rightarrow RR$ 等效于顺序执行 $A \rightarrow RR$, $RR \cdot B \rightarrow RR$ 和 $\overline{RR} \rightarrow RR$ 三条指令。“或非”运算可由先“或”后“非”运算实现, 即 $\overline{A + B} \rightarrow RR$ 等效于顺序执行 $A \rightarrow RR$, $RR + B \rightarrow RR$ 和 $\overline{RR} \rightarrow RR$ 三条指令。“异或”运算可由 $A \cdot B + A \cdot \overline{B}$ 实现; 如设有“同或”指令可由先“同或”后“取反”实现“异或”, 即 $A \oplus B \rightarrow RR$ 等效于顺序执行 $A \rightarrow RR$, $RR \odot B \rightarrow RR$ 和 $\overline{RR} \rightarrow RR$ 三条指令。

控制类指令包括输入/输出控制指令、跳步指令和标志指令。输入控制指令可控制输入型指令(输入指令及运算类指令)的执行。输出控制指令可控制输出型指令的执行。跳步指令又分无条件跳步和有条件(如结果寄存器的内容为“0”)跳步两种。标志指令的执行结果是由 ICU 的相应端输出标志脉冲。

以下我们将分类评述指令的功能、名称、符号、代码、工作时序及简单应用。

二、输入指令

一位机的输入指令有两条，即某输入口信息的原码输入指令和反码输入指令。

1. “取”指令

ICU “取”（或“读”）输入选择器 IS 某一输入口的信息 I_i ，并将它原封不动地传送到内部的结果寄存器中，叫原码输入指令，简称“取”指令。“取”指令的功能可表示为：

$$I_i \rightarrow RR$$

式中的 I_i 表示输入选择器某一输入口的信息（大写的 I 表示输入，小写的 i 表示该输入口的编号或地址） RR 代表 ICU 中的结果寄存器，即数据传送的目的地。箭头（ $\rightarrow$ ）指示传送方向。 $I_i \rightarrow RR$ 简读作“ I_i 送 RR ”。“取”指令的操作码为 0001，编程时可写作十六

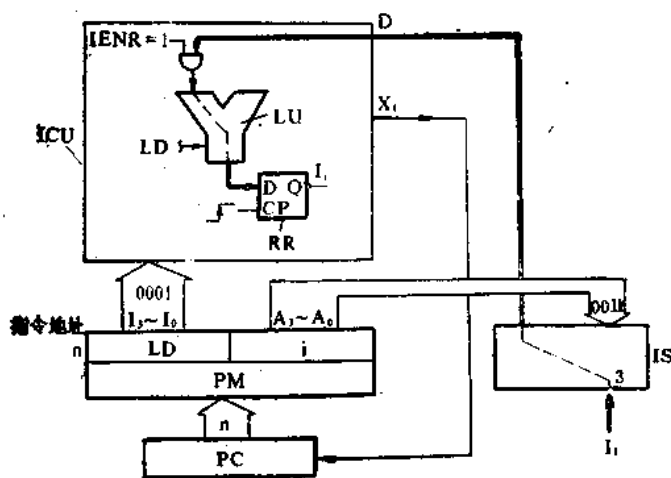


图 1-6 LD 指令的执行过程

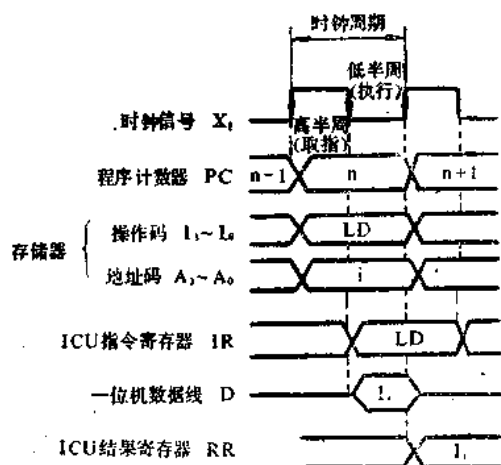


图 1-7 LD 指令的操作时序

进制“1”或助记符 LD（原文 Load）。

“取”指令的地址码为该输入口的二进制地址，编程时亦可写作相应的十六进制数或用符号表示的符号地址（如输入地址 i 或输入是 I_i ）。“取 LD”指令的执行过程如图 1-6 所示，其操作时序如图 1-7 所示。

图 1-6 中程序存储器 PM 的第 n 单元存放着一条“取”指令其高四位为操作码 0001（助记符 LD），低四位为某一输入口的地址码 i （例如 0011，即输入选择器 IS 的第 3 号输入口）。在 X_1 的上升边， $PC+1=n$ 存储器送出第 n 单元中的 LD i 指令，操作码 0001 送入微处理器

ICU, 地址码 0011 送到地址总线 $A_3 \sim A_0$ 。 $X_1 = 1$ 即时钟信号为高电平的期间 (简称 X_1 高半周) 是 ICU 从存储器取出指令, 因此叫取指周期。在 X_1 的下降边, 操作码被打入 ICU 内部的指令寄存器 (IR), 同时输入选择器 (IS) 根据地址总线 $A_3 \sim A_0$ 上的地址码把第 3 号输入口的信息 I_3 送到数据线 D。在开输入状态下, $X_1 = 0$, 即时钟信号为低电平的期间 (简称 X_1 低半周) 是 ICU 按 LD 的操作命令将输入信息 I_3 直接由逻辑单元 LU 的输入端 a 传送到输出端 c。在下一个 X_1 的上升边, I_3 被打入 RR, 取代以前的内容。要“取”其它输入口的信息并送到 ICU 内部的结果寄存器, 只需改变地址码。例如要“取” IS 第 0、1、2、4、5、6、7 的信息至 RR, 则指令可分别写作:

输入口地址	“取” 指 令		功 能
	二 进 制	十 六 进 制	
0	0001 0000	1 0	$I_0 \rightarrow RR$
1	0001 0001	1 1	$I_1 \rightarrow RR$
2	0001 0010	1 2	$I_2 \rightarrow RR$
4	0001 0100	1 4	$I_4 \rightarrow RR$
5	0001 0101	1 5	$I_5 \rightarrow RR$
6	0001 0110	1 6	$I_6 \rightarrow RR$
7	0001 0111	1 7	$I_7 \rightarrow RR$

2. “取反” 指令

ICU “取” (或“读”) 输入选择器 IS 某一输入口的信息 I_i , 并将其“反”码传送到内部的结果寄存器中, 叫反码输入指令, 简称“取反”指令。“取反”指令的功能可表示为:

$$\overline{I_i} \rightarrow RR$$

式中的 $\overline{I_i}$ 表示输入选择器某一输入口信息的反码。如 $I_i = 0$, 则 $\overline{I_i} = 1$ 。 $\overline{I_i} \rightarrow RR$ 简读作“ I_i 的非送 RR”。“取反”指令的操作码为 0010, 编程时可写作十六进制“2”或

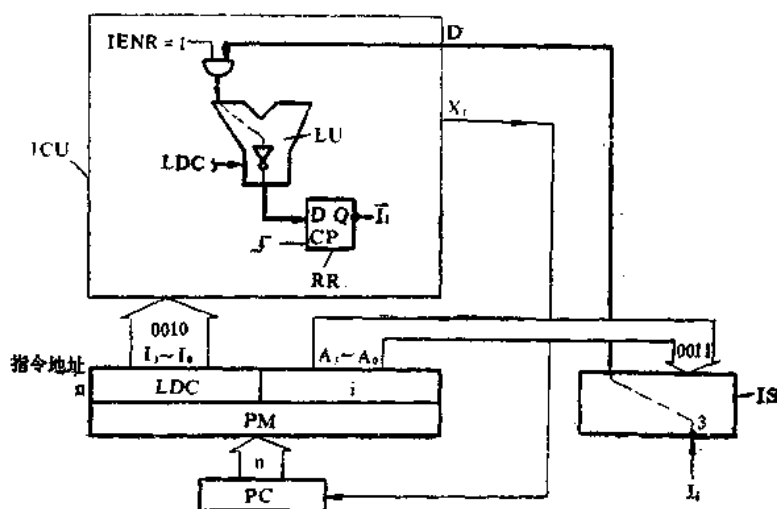


图 1-8 LDC 指令的执行过程

助记符 LDC (原文 Load Complement)。“取反”指令的地址码为该输入口的二进制地址,编程时亦可写作相应的十六进制数或符号地址(如 i 或 I_i)。“取反”(LDC)指令的执行过程如图 1-8 所示,其操作时序同 LD 指令。

设程序存储器 PM 的第 n 单元存放着一条“取反”指令(见图 1-8),其高四位为操作码 0010 (助记符 LDC),低四位为某一输入口的地址码 i (例如 0011,即输入选择器 IS 的第 3 号输入口)。在 X_1 的上升边, $PC+1=n$,存储器送出该单元中的 LDC i 指令,即 X_1 的高半周为取指周期。在开输入状态下, X_1 的低半周是 ICU 按 LDC 的操作命令将逻辑单元 LU 的输入端信息 I_i 取反后传送到输出端。在下一个 X_1 的上升边, I_i 被打入 RR。这是 LDC 指令与 LD 指令执行过程的不同点。执行本条 0010 0011 (十六进制 23) 指令,信息传送的途径是从 IS 的第 3 号输入口起,经 D 线和与门,又经 LU 取反后至 RR。如 IS 的第 3 号输入信息为 I_3 ,则执行结果是 RR 的 Q 端为 I_3 ,要“取”其它输入口信息的反码送 RR,也需改变地址码。例如

输入口地址	“取反”指令		功 能
	二 进 制	十 六 进 制	
0	0010 0000	2 0	$\bar{I}_0 \rightarrow RR$
1	0010 0001	2 1	$\bar{I}_1 \rightarrow RR$
2	0010 0010	2 2	$\bar{I}_2 \rightarrow RR$
4	0010 0100	2 4	$\bar{I}_4 \rightarrow RR$
5	0010 0101	2 5	$\bar{I}_5 \rightarrow RR$
6	0010 0110	2 6	$\bar{I}_6 \rightarrow RR$
7	0010 0111	2 7	$\bar{I}_7 \rightarrow RR$

三、输出指令

一位机的输出指令也有两条,即结果寄存器内容的原码输出指令和反码输出指令。

1. “存”指令

ICU 将结果寄存器内容的原码 RR “存”(或“写”)入外部输出锁存器 OL 的某一输出口 O_i ,叫原码输出指令,简称“存”指令。“存”指令的功能可表示为:

$$RR \rightarrow O_i$$

式中的 O_i 表示输出锁存器具有锁存作用的某一输出口(大写的 O 表示输出,小写的 i 表示该输出口的编号或地址,即传送的目的地)。RR 代表数据的来源。 $RR \rightarrow O_i$ 简读作“RR 送 O_i ”。“存”指令的操作码为 1000,编程时可写作十六进制“8”或助记符 STO (Store)。“存”指令的地址码为该输出口的二进制地址,编程时亦可写作相应的十六进制数或符号地址(如输出地址 j 或输出量 O_j)。“存”(STO)指令的执行过程如图 1-9 所示,其操作时序如图 1-10 所示。

该程序存储器的第 n 单元存放着一条“存”指令(见图 1-9),其高四位为操作码 1000 (助记符 STO),低四位为某一输出口的地址码 j (例如 0011,即输出锁存器 OL 的第 3 号输出口)。在 X_1 的上升边, $PC+1=n$,存储器送出该单元中的 STO j 指令,即 X_1 的高半周为取指周期。在 X_1 的下降边,操作码 1000 被打入指令寄存器 IR。与