1位微处理器

MC14500B

第一机械工业部自动化研究所二室
一九八〇年一月译
一九八一年五月印

目　　录

序

在电子控制和机电控制设备中碰到的很多问题都包含有面向决定的任务。同时，这些决定常常导致诸如将某物开或关那样简单的命令。例如：限位开关是否闭合？定时间隔是否完结？当继电器A、B和C闭合时接通泵P17，给双向三端可控硅发送20个脉冲，接通红外光管，计数60个脉冲后启动马达M1以及很多类似的工作。

当然，解决这类问题的方法有很多种，最初，大量使用原理简单而又容易维护的继电器，但是，继电器体积大，价格贵，耗电量大，长期运行的可靠性差，同时由于不能方便地适应系统的变更，因而使用受到限制。

其后，发展了固态逻辑，这些器件体积小，价格很便宜，耗电量只是一个继电器耗电量的一部分，而且在保持原理简单和容易维护的同时具有很高的长期运行的可靠性，但是，一旦装入系统，它们不容易做到程序可编，而且系统的变更不能迅速而廉价地得以实现。因而使用仍然受到限制。

计算机和微型机是可以采用的，但是，它们趋向于使任务过于复杂化，同时常常需要经过高级训练的人员来发展和维护这个系统。

已经发展了一种较为简单的装置，它是设计用来一次对一个输入和输出进行运算的，而且结构内容与继电器系统相似，工业控制领域称这种装置为：可编程序逻辑控制器（PLC）。

MOTOROLA公司的工业控制单元（ICU）MC14500B是可编程序逻辑控制器的中央处理部分的单片结构型式。它的特点如下：

（1）只有16条指令；

（2）编程简单，容易理解。

（3）容易掌握，一般人员就能维护。

（4）采用外存储器以进行灵活的系统设计。

（5）能特殊设计以满足用户的特殊需要。

（6）容易扩展以适应各种规模和复杂程度的系统需要。

（7）具有程序可编的优点。

（8）符合B系列CMOS JEDEC规格。

（9）抗噪声容限大。

（10）静态电流小。

（11）工作电压3～18V。

（12）静态工作。

（13）时钟频率范围宽，典型的是1 MHZ，V_{DD} = 5 V，一个时钟周期执行一条指令。

（14）指令输入端与TTL兼容。

（15）对于面向决定的任务而言，比微处理器好。

（16）应用范围广：从继电器梯形图的逻辑处理到中速串行数据处理，还能取代微处理器的基本系统以减轻其过重负担。

这本手册可作为这一器件的设计和应用手册。

第一章 引 论

MOTOROLA公司的MC14500B是单片1位静态CMOS处理器，它最适用于完成面向决定的任务。这种处理器封装在一个16个插脚的管壳中，同时其特点是具有16条4位的指令，这些指令对1位双向数据总线上出现的数据和ICU内1位累加结果寄存器中的数据进行逻辑运算、所有的运算都是按位进行的。

ICU由使用外部电阻的内部振荡器产生的单相时钟信号来定时，另外，时钟信号也可由外部振荡器来控制。无论在哪种场合，时钟信号总是要用的，以便与其它系统同步。每一条ICU指令均在一个时钟周期内执行完毕。时钟频率可在很大范围内变化。当时钟频率为1MHZ时，在60HZ电源线的半个周波内可执行约8300个脉冲即8300条指令。

在一个系统中，ICU可和100多种标准B系列CMOS逻辑器件一起使用，这就可以针对应用来组成系统，同时也可以使用户需要的硬件和软件的合适搭配得以实现。

作为一个最初的例子，在图1.1中，示出了一个最小ICU系统的方框图，在图中除ICU外还有4个部件方框，这些方框是：

①ICU，或称：系统的中央控制器。

②存储器：可以是永久性的只读有储器(ROM)也可以是暂时性的随机存储器(RAM)，里面存入程序步，在其中既包含有各种指令，也包含有输入、输出地址。

③程序计数器：用来使机器按指令的顺序步进。

④输入和输出：根据存储器中包含的信息，分别由机器来加以选择。

请注意，只要存储器的位数足以满足I/O结构的寻址要求，这个系统的输入、输出点数就几乎可以无限制地扩展。

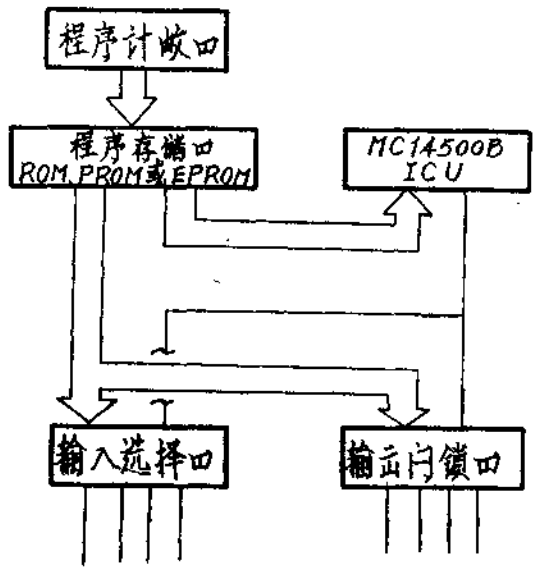


图1.1 基本的ICU系统

对于复杂计算或并行数据处理这样一些功能，1位机是难以适应的。所以，当有许多计算时，用1位机是不适合的，当一项任务主要是计算或数据纪录时，采用多位处理器是恰当的。如果是属于面向决定和面向命令的任务，采用1位机最合适。对于兼有决定和计算的任务，则要取决于经济性以及设计者对这两种器件的熟悉和喜爱程度。在某些情况下，组合应用MC6800MPU和MC14500BICU可能是最好的解决办法。

MC14500B的功能图示于图1.2中，ICU的中心是结果寄存器（RR），它是个1位累加器，用来存放逻辑运算的结果。这些结果由逻辑单元(LU)给出，它的输入是来自外部的数据和RR的信号。指令通过4条指令线（I_0,I_1,I_2,I_3）加到这个片子上，同时在X_1的下降沿

时存入指令寄存器（IR）中。

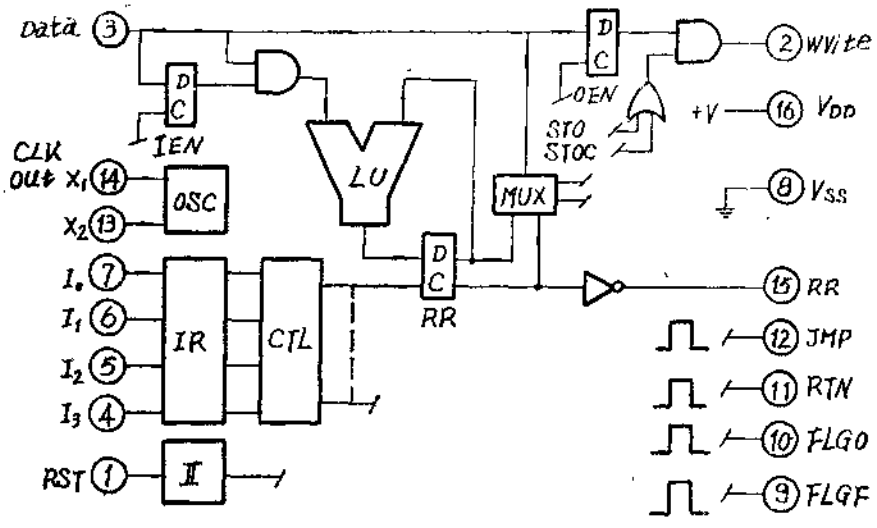


图1.2 MC14500B方框图

列在图（表）1.3中的指令经控制逻辑回路（CTL）译码后给LU传送相应的逻辑命令，有些指令经CTL译码后给插脚9～12传送输出标志（JMP，RTN，FLGO，FLGF），这些输出标志用作外部控制信号，它们在X_1下降沿以后的整个时钟周期内有效。

图（表）1.3 MC14500B的指令系统

指令码		助记符	动作
*0	0000	NOPO	各寄存器没有变化，RR→RR，FLGO←几
*1	0001	LD	取。取数据送入RR，Data→RR
*2	0010	LDC	取反。 $\overline{Data}$→RR
*3	0011	AND	逻辑与。 RR·D→RR
*4	0100	ANDC	逻辑与反。 RR·$\overline{D}$→RR
*5	0101	OR	逻辑或。 RR+D→RR
*6	0110	ORC	逻辑或反。 RR+$\overline{D}$→RR
*7	0111	XNOR	异或非。 如果RR=D，则RR←1
*8	1000	STO	存。 RR→Data，插脚Write←1
*9	1001	STOC	存反。 $\overline{RR}$←Data，插脚Write←1
*A	1010	IEN	允许输入。 D→IEN寄存器
*B	1011	OEN	允许输出。 D→OEN寄存器
*C	1100	JMP	转移。 JMP标志←几
*D	1101	RTN	返回。 RTN标志←几，并跳过下一条指令。
*E	1110	SKZ	跳越。 如果RR=0，则跳过下一条指令。
*F	1111	NOPF	各寄存器没有变化，RR→RR，FLGF←几

定时信号由芯片内的振荡器（OSC）产生，工作频率由联接在插脚13和14之间的外部电阻来决定。图1.4示出了频率和电阻值之间的关系。插脚14的方波输出除在ICU内部使用外，还作为总的系统时钟。此外，系统也可使用从插脚13引入的外部时钟。

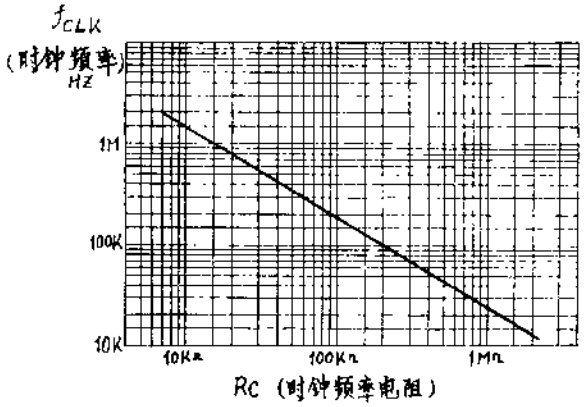


图1.4 典型的时钟频率与电阻（Rc）的关系

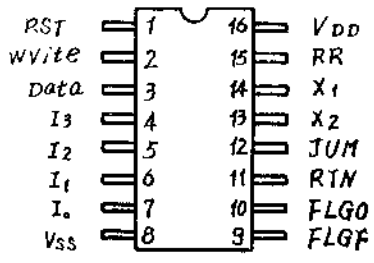


图1.5 插脚排列

（注：图中的JUM应改为JMP）

两个闩锁器，即：允许输入寄存器（IEN）和允许输出寄存器（OEN），分别控制ICU的数据输入和输出，IEN为高电平时，允许向LU输入数据。OEN为高电平时，允许送出Write信号。要注意的是，这两个寄存器都是经由Data插脚来置位的。

主复位插脚（RST）为高电平时将所有寄存器清零，同时使ICU内的标志信号变为零。当RST为高电平时，振荡器插脚（X_1）也为高电平。当RST变为低电平后，稍经延时振荡器起振。此外，RR的状态经缓冲后，由插脚15输出。

ICU芯片封装在一个双列直插式管壳中，有塑料的和陶瓷的两种，其温度范围和封装型式如下：

MC14500BAL：陶瓷封装，军用温度范围；

MC14500BCL：陶瓷封装，民用温度范围；

MC14500BCP：塑料封装，民用温度范围、

插脚排列如图1.5所示。

ICU的极限容量和电气特性示于图（表）1.6中。这些特性符合CMOS B系列器件的JEDEC B系列规格，其推荐用电源电压范围为5～15Vdc。在电噪声的工业环境中、推荐用15Vdc的电源电压，以便最佳地利用CMOS逻辑器件的极好的抗噪声特性。ICU除了能和100多种B系列器件一起工作外，还可和非B系列CMOS器件共同使用。关于与ICU相兼容的很多器件的详细资料，请参考：“MotoroLa Semiconductor Data Library，CMOS VoLume 5/SeriesB”（“MOTOROLA公司半导体数据图书，CMOS第5卷/B系列”）。

开关特性和解释波形分别示于图（表）1.7和图1.8中，所有的时间均以插脚14的时钟信号为基准。在本书中，仅给出了V_{DD}=10Vdc时的典型时间规范，最新规范请参考MC14500B的数据单。

图（表）1.6　　电 气 特 性

特　　性		符　号	V_{DD} (V_{dc})	25°C 典型值	单　位
输出电压 （$V_{in}=V_{DD}$或0） （$V_{in}=0$或V_{DD}）	"0"电平	V_{OL}	10	0	V_{dc}
	"1"电平	V_{OH}	10	10	V_{dc}
输入电压* RST, D, X_2 （$V_O=9.0$或$1.0V_{dc}$） （$V_O=1.0$或$9.0V_{dc}$）	"0"电平	V_{iL}	10	4.50	V_{dc}
	"1"电平	V_{iH}	10	5.50	V_{dc}
输入电压 I_0, I_1, I_2, I_3 （$V_O=9.0$或$1.0V_{dc}$） （$V_O=1.0$或$9.0V_{dc}$）	"0"电平	V_{iL}	10	2.2	V_{dc}
	"1"电平	V_{iH}	10	3.1	V_{dc}
输出驱动电流 D, Write （$V_{OH}=9.5V_{dc}$） （$V_{OL}=0.5V_{dc}$）	供给电流	I_{OH}	10	−6.0	mA_{dc}
	吸收电流	I_{OL}	10	6.0	mA_{dc}
输出驱动电流（AL器件） 其它输出端 （$V_{OH}=9.5V_{dc}$） （$V_{OL}=0.5V_{dc}$）	供给电流	I_{OH}	10	−2.25	mA_{dc}
	吸收电流	I_{OL}	10	2.25	mA_{dc}
输出驱动电流（CL/CP器件） 其它输出端 （$V_{OH}=9.5V_{dc}$） （$V_{OL}=0.5V_{dc}$）	供给电流	I_{OH}	10	−2.25	mA_{dc}
	吸收电流	I_{OL}	10	2.25	mA_{dc}
输入电流（RST）		I_{in}	15	150	μA_{dc}
输入电流（AL器件）		I_{in}	15	±0.00001	μA_{dc}
输入电流（CL/CP器件）		I_{in}	15	±0.00001	μA_{dc}
输入电容（DATA）		C_{in}	—	1.5	PF
输入电容（所有其它输入端） （$V_{in}=0$ V）		C_{in}	—	5.0	PF
静态电流（AL器件） （一片）		I_{DD}	10	0.010	μA_{dc}
静态电流（CL/CP器件） （一片）		I_{DD}	10	0.010	μA_{dc}

* 最低温度$T_{LOW}=-55°C$（AL器件），−40°C（CL/CP器件）
最高温度$T_{high}=+125°C$（AL器件），+85°C（CL/CP器件）
"0"电平和"1"电平的最小抗噪声容限为$2.0V_{dc}$（$V_{DD}=10V_{dc}$）

图（表）1.7　　开 关 特 性

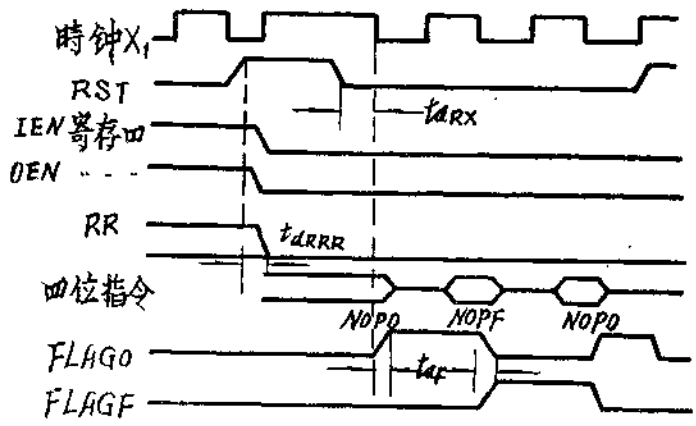
特　　性	符　号	V_{DD}	ALCLCP 典型值	单　位
传输延迟时间				
X_1到RR	t_{dR}	10	110	ns
从X_1到FLAGF,FLAGO,RTN,JMP	t_{dF}	10	100	ns
从X_1到WRITE	t_{dW}	10	125	ns
从X_1到DATA	t_{dD}	10	120	ns
从RST到RR	t_{dRRR}	10	110	ns
从RST到X_1	t_{dRX}	10	120	ns
从RST到FLAGF,FLAGO,RTN,JMP	t_{dRF}	10	90	ns
从RST到WRITE,DATA	t_{dRW}	10	110	ns
最小时钟脉冲宽度,X_1	PW_C	10	40	ns
最小复位脉冲宽度,RST	PW_R	10	50	ns
建立时间				
指令	t_{iS}	10	125	ns
数据	t_{DS}	10	50	ns
保持时间				
指令	t_{iH}	10	0	ns
数据	t_{DH}	10	30	ns

注：T_A（环境温度）=25°C

对于X（时钟）和I（指令）输入端，t_r（前沿）=t_f（后沿）=20ns；

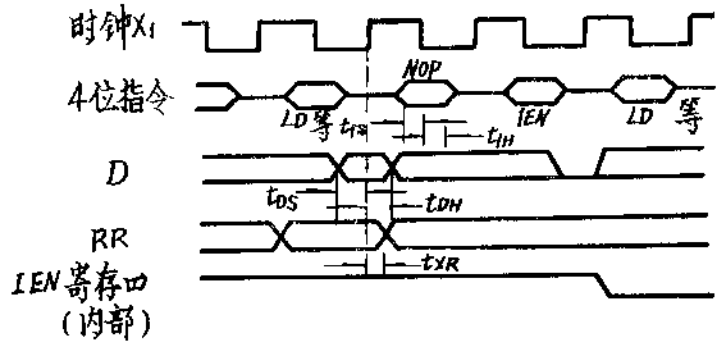
对于JMP,X,RR,FLAGO,FLAGF端，C_L（负载电容）=50PF；

对于DATA和Write端，C_L=130PF+1个TTL负载。

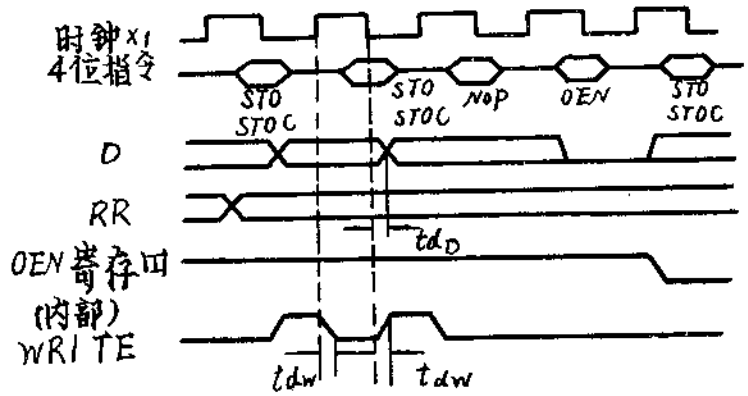


（a）NOPO,NOPF指令

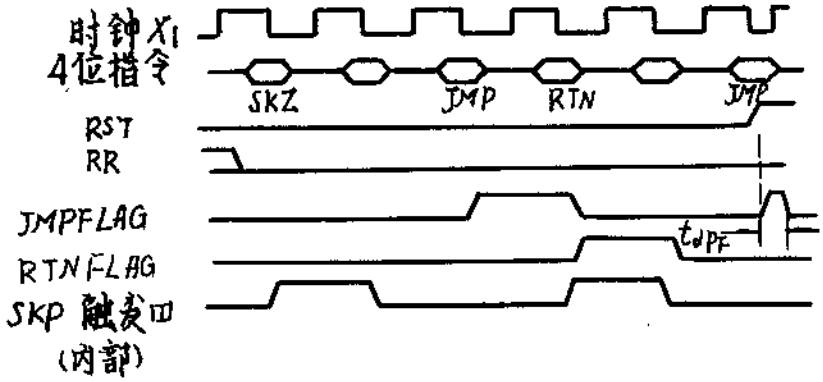
RR,IEN,OEN寄存器不受影响



(b) LD,LDC,AND,ANDC,OR,ORC,XNOR和IEN指令，
RST = 0 时有效



(c) STO.STOC,OEN指令
RST = 0 时有效



(d) SKZ,JMP,RTN指令
RR,IEN,REN寄存器不受影响

图1.8 定时波形

第二章　基本概念

图2.1所示方框图是以ICU为基础的小型PLC系统的一个例子。和ICU一起使用的部件除存储器外都是标准的CMOS部件。

ICU系统根据存储程序式处理器的原理进行工作，一组称为指令的命令，存放在ICU系统的存储器中，每一条命令指示ICU系统执行16种操作中的一种。

系统从存储器中，“取出”一条命令和执行该命令所需的信息，然后执行该命令。执行完毕，再从存储器中按顺序地“取出”下一条命令，这一过程反复进行，无限循环。

一、LD（取）和STO（存）指令

LOAD（简写为LD）是一条典型的指令，这条指令指示ICU系统取某一个输入点的逻辑电平（逻辑“1”或逻辑“0”）并将其存入ICU内部的结果寄存器中，使用LD指令时，用户需将LD指令和采样输入地址在存储器中编好程序。系统的动作如下：系统存储器把LD指令送给ICU（即取出该指令），同时把采样输入地址送给输入选择器。于是，被选中输入点的逻辑电平就经过ICU的1位数据总线传送给1位结果寄存器（参看图2.1）。

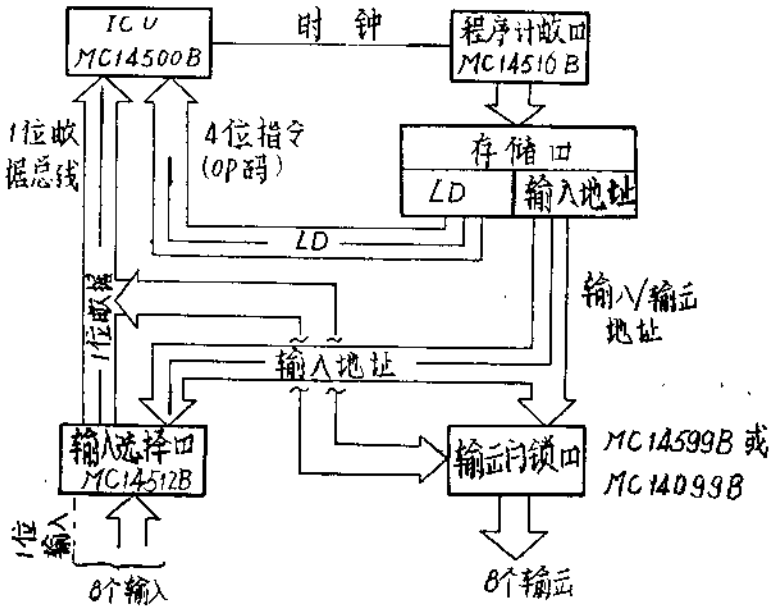


图2.1　典型小系统的构成和数据流向

（译者注：原文均把MC14512B误写为MC14512）

另一条典型的指令是STORE（简写为STO），这条指令指示ICU系统将其1位结果寄存器中的数据传送给某一个输出闩锁器。使用STO指令时，用户需将STO指令和接收数据

的输出闩锁器的地址在存储器中编好程序。系统的动作如下：系统存储器把 STO 指令送给 ICU，同时把要选择的输出闩锁器的地址送给输出器件。于是，ICU内 1 位结果寄存器中的数据就经过ICU的 1 位数据总线传送给被选中的闩锁器（参看图2.2）。

可见，数据既可送入系统，也可从系统送出。

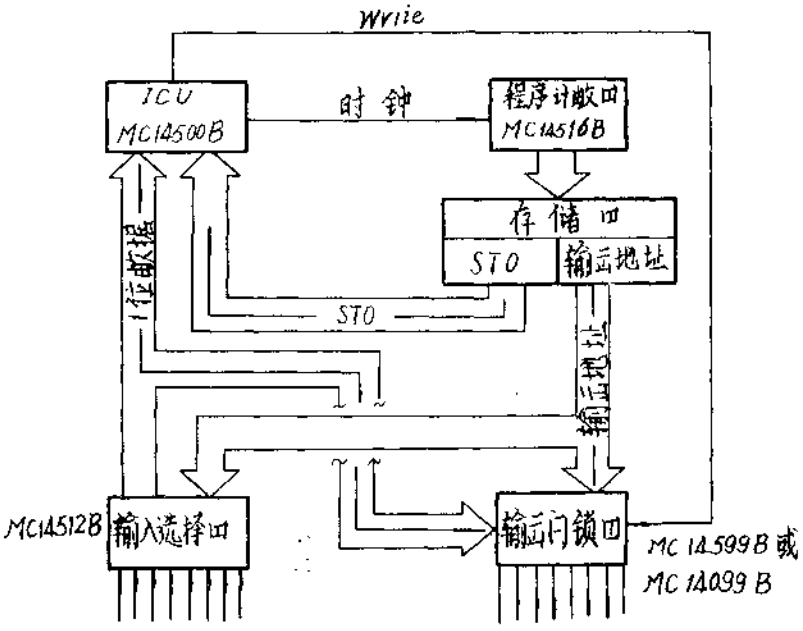


图2.2 STO指令的系统动作

二、系统部件

1. 存储器

存储器（参看图2.3）存放指挥整个系统执行规定任务的程序。程序由指令和地址组成。指令以 4 位操作码（OP 码）的形式送到ICU 去，而地址（用二进制数来表示）则规定了 ICU 1 位双向数据总线和输入、输出器件之间数据传送的路径。

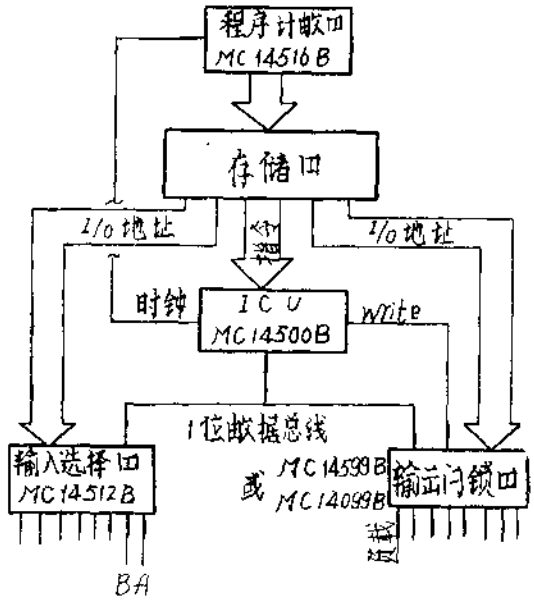


图2.3 基本系统

2. ICU

ICU是系统的中央控制单元。它控制其内部寄存器和其 1 位双向数据总线之间的数据流向，它对结果寄存器中的数据和 1 位双向数据总线上的数据进行逻辑运算、同时向其它系统部件发送控制信号以便协调系统的工作。

3. 程序计数器

程序计数器（PC）向ICU系统存储器指示应执行的指令的地址。PC顺序地进行二进制加法计数，达最大值后返回到零，然后再重新计数。这就使存储器中的指令序列能重复出现，形成所谓循环程序。

4. 输入选择器

输入选择器用来决定在某一特定的操作中使用哪一个输入点。ICU系统存储器把输入地址送给输入选择器，于是，选择器选出该点数据，送到ICU 1位双向数据总线上以供ICU使用。这样，就从很多输入点中选出一个点来，

5. 输出闩锁器

除数据流向相反外，输出闩锁器与输入选择器非常相似，当ICU收到一条将结果寄存器的数据送出的指令(STO)时，它就把该数据传送到1位双向数据总线上，同时通过WRITE控制线给输出闩锁器发一个信号。于是，输出器件就把该数据送到存储器给出的地址所指定的闩锁器中。

三、AND（与）指令

在举例往下讲之前，有必要先讲述另一条指令，即：AND指令。AND指令的动作如下：系统存储器把AND指令的操作码送给ICU，同时把某一个输入地址送给输入选择器、于是，指定地址的输入数据被选出并送到ICU的双向数据总线上，然后，数据总线上的信息和结果寄存器中的数据进行“与”运算，运算结果成为结果寄存器的新内容、请注意：只有当结果寄存器原先的内容和输入数据均为逻辑“1”时、结果寄存器的最终内容才为逻辑“1”、真值表如下：

输入	“与”	结果寄存器初始内容	=	结果寄存器新内容
0		0		0
0		1		0
1		0		0
1		1		1

例

图2.3所示的基本系统，很适合于解决用继电器梯形图或逻辑图形式表达的问题。图2.4示出了用这两种形式表达的问题：LOAD = A · B

（A）继电器梯形图　　（B）等效逻辑图

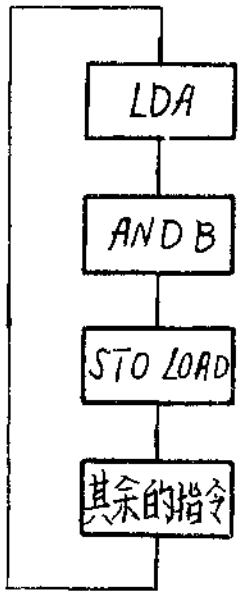
图2.4 LOAD = A·B

可见，当A和B均闭合时（即为逻辑“1”时），LOAD得电（即为逻辑“1”）。

ICU解这个问题不仅只进行一次，而是程序每循环一次进行一次。因此，如果程序中有1000条指令，同时，时钟频率为500KHZ，那末，每秒钟将对输入采样500次（即每2ms采样一次）而且在某一输入点变化后的2ms内，相应的输出点将得电或失电。这就称为循环控制结构。

图2.5示出了解这个问题所需的ICU程序。

当然，指令的顺序可很容易地改为：LDB；ANDA；STO LOAD。



将输入A的状态取入ICU的结果寄存器(RR)中。

将输入B的状态和ICU的结果寄存器中的数据(现在是A的状态)进行“与”运算，其结果变为结果寄存器的新内容。

将RR中的数据传送给称为LOAD的输出，从而使负载设备投入运行或停止工作。

执行程序的其余部分，然后循环返回。

图2.5 LOAD = A·B的程序举例

第三章 基础编程知识和指令系统

一、累加结果寄存器

读者一定注意到，AND指令引入了累加结果寄存器的概念。执行这条指令时。ICU对其1位双向数据总线上的数据和其结果寄存器中的数据进行“与”运算，运算结果变为结果寄存器的新内容。值得指出的是，结果寄存器总能接收任何一条ICU逻辑指令的运算结果。这样，结果寄存器就把每一条ICU逻辑指令的运算结果累加起来了。这一点与加法机相类似，这种机器在每一次运算之后，总把累计总数显示出来。

二、反码指令

有时要求，当一个输入为逻辑“0”而另一个输入为逻辑“1”时，使某一输出动作。这种情况出现在使用常闭触点的继电器控制系统中，也出现在有反相器的固态逻辑系统中。图3.1示出了这种情况的一个例子。

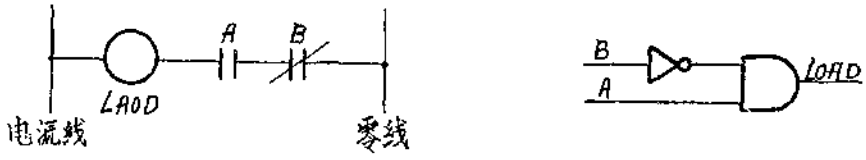
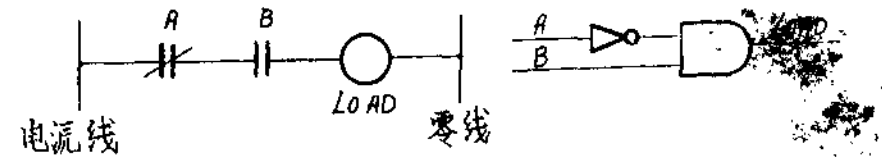


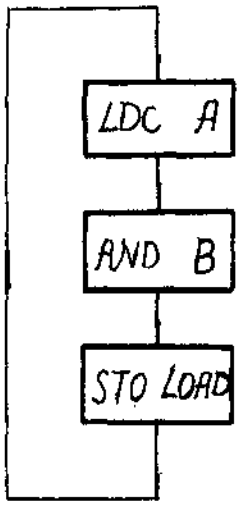
图3.1 反码信号的例子

ICU的指令系统已经为这种情况作了准备，有几条逻辑“反码”指令在进行运算之前，先将ICU双向数据总线上的数据的逻辑电平反相。

三、LDC（取反）指令

这些指令的一个例子是取反指令，简写为LDC。这条指令的动作如下：ICU系统存储器把LDC指令送给ICU，同时把运算所用输入地址送给输入选择器，于是，输入选择器把被选中输入点的数据选出并送到ICU双向数据总线上。ICU把该数据反相并将结果存入其1位结果寄存器中。如果，被选中输入点是逻辑“0”，那末，结果寄存器将接收到逻辑“1”。图3.2示出了使用LDC指令来解图3.1所示问题的ICU程序。读者在往下阅读之前，必须确认这一程序的动作。





把输入A的反码取入结果寄存器。

将输入B和结果寄存器的内容(输入A的反码)进行“与”运算。其结果变成结果寄存器的新内容。

将结果寄存器的数据传给称为LOAD的输出闩锁器。

请注意：只有A信号为逻辑“0”而B信号为逻辑“1”时，STO指令才传送逻辑“1”信号给输出闩锁器。

图3.2 LDC指令的使用

四、ANDC（与反）指令

逻辑反码指令的另一个例子是“与反”指令，简写为ANDC。ANDC指令的动作如下：ICU系统存储器把ANDC指令送给ICU，同时把被选输入点的地址送给输入选择器。于是，输入选择器把该数据选出并送到ICU的1位双向数据总线上。ICU把该数据反相，然后将此数据和结果寄存器中的数据进行“与”运算，运算结果变为结果寄存器的新内容。如果，被选中输入点是逻辑“0”而结果寄存器原先是逻辑“1”。那末，结果寄存器将接收到逻辑“1”。

利用这条指令，ICU就能进行一些更为复杂的“链式”计算。图3.3示出了一个这样的例子，图（表）3.4示出了解图3.3所示问题的ICU程序。

在回顾这些指令的动作之后，读者必能确认：只有A＝1，B＝0，C＝1和D＝0时，负载设备才接收到逻辑“1”信号。

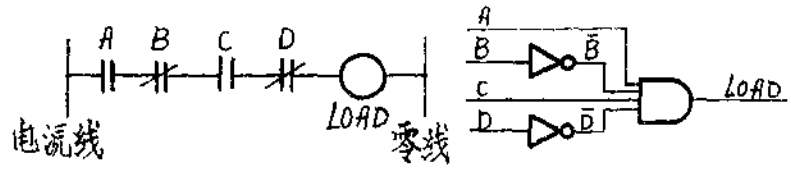


图3.3 链式计算的例子

语句	算符	操作数	说明
*1	LD	A	结果寄存器←A
*2	ANDC	B	结果寄存器←A·$\bar{B}$
*3	AND	C	结果寄存器←A·$\bar{B}$·C
*4	ANDC	D	结果寄存器←A·$\bar{B}$·C·$\bar{D}$
*5	STO	LOAD	结果寄存器=A·$\bar{B}$·C·$\bar{D}$→LOAD

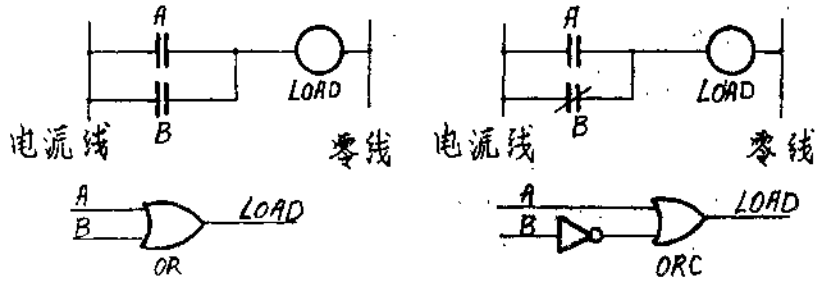
图(表)3.4 解图3.3所示链式计算的程序

五、OR(或)和ORC(或反)指令

在很多情况下要求，当两个输入中的任何一个是逻辑“1”时，使某一输出动作。这时需用“或”指令OR。OR指令的动作如下：ICU系统存储器把OR指令送给ICU，同时把运算所用输入地址送给输入选择器。于是，输入选择器选出指定地址的数据并送到ICU的双向数据总线上。然后，ICU将该数据和结果寄存器中的内容进行“或”运算。并把运算结果返回到结果寄器中去。

在需要逻辑反相的场合，ICU还有一条“或反”指令，简写为ORC。这条指令的动作，除在运算前将输入数据反相外，其余很类似于OR指令。图3.5示出了一些使用OR或ORC的例子。

在使用OR指令的例子中，如果输入A或输入B或两者均为逻辑“1”时，负载设备就会接收到逻辑“1”信号。在ORC的例子中，如果输入A为逻辑“1”或输入B为逻辑“0”时，负载信号就会接收到逻辑“1”信号。



*1	LD	A	RR←A
*2	OR	B	RR←A+B
*3	STO	LOAD	A+B→RR→LOAD

OR指令的使用

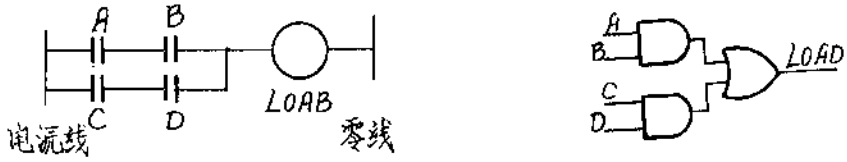
*1	LD	A	RR←A
*2	ORC	B	RR←A+$\bar{B}$
*3	STO	LOAD	A+$\bar{B}$→RR→LOAD

ORC指令的使用

图3.5 OR和ORC指令的使用

六、暂存单元的使用

在工业控制中碰到的很多逻辑结构是一些串联继电器触点支路与另外的串联继电器触点支路相并联。图3.6示出了这种结构的一个例子。

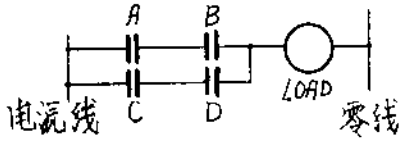


继电器梯形图　　等效逻辑图

图3.6　串联－并联组合

(注：左图中LOAB应改为LOAD)

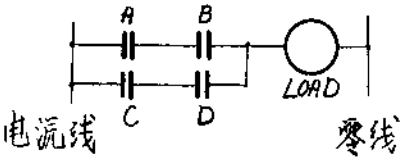
在处理这类问题时，为要正确实现所需逻辑功能，而把LD，LDC，AND，ANDC，OR和ORC指令直接联接未必总是可能的。在某些场合，在处理问题的剩余部份之前，有必要把中间结果暂存起来。在这些场合，编程人员必须依据求解表达式的需要，使用LD，AND和ANDC指令求解串联支路，然后把结果存入暂存单元中。然后必须对第二个串联支路求解并和存在暂存单元中的数据进行“或”运算。最后，运算结果用来使输出设备投入运行或停止工作。图3.7示出了这类问题的编程中的一个普遍的错误而图3.8说明了解这一问题的正确途径。图3.8示出了利用暂存单元解这一问题的正确方法。




```
*1  LD    A      RR←A
*2  AND   B      RR←A·B
*3  OR    C      RR←A·B+C  **错误
*4  AND   D      RR←(A·B+C)D
*5  STO   LOAD   RR→LOAD
```

**请注意：最终表达式错误地导致了D项可和所有其它项分配相“与”。例如：如果A，B和C是逻辑“1”而输入D是逻辑“0”那末，负载设备就会接收到逻辑“0”，这是错误的，因为当输入A和B是逻辑“1”时，负载设备就能动作。

图3.7　错误的编程例子




```
*1  LD    A      RR←A
*2  AND   B      RR←A·B
*3  STO   TEMP   RR=A·B→TEMP
*4  LD    C      RR←C
*5  AND   D      RR←C·D
*6  OR    TEMP   RR←C·D+TEMP=A·B+C·D
*7  STO   LOAD   RR=A·B+C·D→LOAD
```

在这一程序中，A“与”B的逻辑结果先暂存起来，然后C“与”D的结果和原先存在暂存单元中的数据相“或”。最后，把正确的逻辑信号传送给负载设备。此例说明了在进行处理之前，需要暂存单元。

图3.8　解题的正确方法