**计算机组成原理**

**实验教程**



西安唐都科教仪器公司

2017

**版权声明**

本书的版权归西安唐都科教仪器开发有限责任公司所有，保留一切权利。非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本书的部分或全部，并以任何形式传播。

西安唐都科教仪器开发有限责任公司，1999-2020(C)，All right reserved.

计算机组成原理与系统结构实验教程

©版权所有 非经许可 严禁复制

技术支持邮箱：tangdukejiao@126.com

唐都公司网址：<http://www.tangdu.com/>

**实验一 基本运算器实验**

计算机的一个最主要的功能就是处理各种算术和逻辑运算，这个功能要由CPU中的运算器来完成，运算器也称作算术逻辑部件ALU。通过基本的运算器实验，了解运算器的基本结构。

**一、 实验目的**

(1) 了解运算器的组成结构。

(2) 掌握运算器的工作原理。

**二、 实验设备**

PC机一台，TD-CMA实验系统一套。

**三、 实验原理**

本实验的原理如图1-1所示。

运算器内部含有三个独立运算**部件，分**别为算术、逻辑和移位运算部件，要处理的数据存于暂存器A和暂存器B，三个部件同时接受来自A和B的数据（有些处理器体系结构把移位运算器放于算术和逻辑运算部件之前，如ARM），各部件对操作数进行何种运算由控制信号S3…S0和CN来决定，任何时候，多路选择开关只选择三部件中一个部件的结果作为ALU的输出。如果是影响进位的运算，还将置进位标志FC，在运算结果输出前，置ALU零标志。ALU中所有模块集成在一片CPLD中。

逻辑运算部件由逻辑门构成，较为简单，而后面又有专门的算术运算部件设计实验，在此对这两个部件不再赘述。移位运算采用的是桶形移位器，一般采用交叉开关矩阵来实现，交叉开关的原理如图1-2所示。图中显示的是一个4×4的矩阵（系统中是一个8×8的矩阵）。每一个输入都通过开关与一个输出相连，把沿对角线的开关导通，就可实现移位功能，即：

(1) 对于逻辑左移或逻辑右移功能，将一条对角线的开关导通，这将所有的输入位与所使用的输出分别相连,而没有同任何输入相连的则输出连接0。

(2) 对于循环右移功能，右移对角线同互补的左移对角线一起激活。例如，在4位矩阵中使用‘右1’和‘左3’对角线来实现右循环1位。

(3) 对于未连接的输出位，移位时使用符号扩展或是0填充，具体由相应的指令控制。使用另外的逻辑进行移位总量译码和符号判别。



**图1-1 运算器原理图**

运算器部件由一片CPLD实现。ALU的输入和输出通过三态门74LS245连到CPU内总线上，另外还有指示灯标明进位标志FC和零标志FZ。请注意：实验箱上凡是印标注有马蹄形标记‘  ’，表示这两根排针之间是连通的。图中除T4和CLR，其余信号均来自于ALU单元的排线座，实验箱中所有单元的T1、T2、T3、T4都连接至控制总线单元的T1、T2、T3、T4，CLR都连接至CON单元的CLR按钮。T4由时序单元的TS4提供，其余控制信号均由CON单元的二进制数据开关模拟给出。控制信号中除T4为脉冲信号外，其余均为电平信号，其中ALU\_B为低有效，其余为高有效。



**图1-2 交叉开关桶形移位器原理图**

暂存器A和暂存器B的数据能在LED灯上实时显示，原理如图1-3所示（以A0为例，其它相同）。进位标志FC、零标志FZ和数据总线D7…D0的显示原理也是如此。



**图1-3 A0显示原理图**

ALU和外围电路的连接如图1-4所示，图中的小方框代表排针座。

运算器的逻辑功能表如表1-1所示，其中S3 S2 S1 S0 CN为控制信号，FC为进位标志，FZ为运算器零标志，表中功能栏内的FC、FZ表示当前运算会影响到该标志。



**图1-4 ALU和外围电路连接原理图**

**表1-1 运算器逻辑功能表**

|  |  |  |  |
| --- | --- | --- | --- |
| **运算类型** | **S3 S2 S1 S0** | **CN** | **功 能** |
| 逻辑运算 | 0000 | X | F=A（直通） |
| 0001 | X | F=B（直通） |
| 0010 | X | F=AB （FZ） |
| 0011 | X | F=A+B （FZ） |
| 0100 | X | F=/A （FZ） |
| 移位运算 | 0101 | X | F=A不带进位循环右移B（取低3位）位 （FZ） |
| 0110 | 0 | F=A逻辑右移一位 （FZ） |
| 1 | F=A带进位循环右移一位 （FC，FZ） |
| 0111 | 0 | F=A逻辑左移一位 （FZ） |
| 1 | F=A带进位循环左移一位 （FC，FZ） |
| 算术运算 | 1000 | X | 置FC=CN （FC） |
| 1001 | X | F=A加B （FC，FZ） |
| 1010 | X | F=A加B加FC （FC，FZ） |
| 1011 | X | F=A减B （FC，FZ） |
| 1100 | X | F=A减1 （FC，FZ） |
| 1101 | X | F=A加1 （FC，FZ） |
| 1110 | X | （保留） |
| 1111 | X | （保留） |

\*表中“X”为任意态，下同

**四、 实验步骤**

(1) 按图1-5连接实验电路，并检查无误。图中将用户需要连接的信号用圆圈标明（其它实验相同）。



**图1-5 实验接线图**

(2) 将时序与操作台单元的开关KK2置为‘单步’档,开关KK1、KK3置为‘运行’档。

(3) 打开电源开关，如果听到有‘嘀’报警声，说明有总线竞争现象，应立即关闭电源，重新检查接线，直到错误排除。然后按动CON单元的CLR按钮，将运算器的A、B和FC、FZ清零。

(4) 用输入开关向暂存器A置数。

① 拨动CON单元的SD27…SD20数据开关，形成二进制数01100101（或其它数值），数据显示亮为‘1’，灭为‘0’。

② 置LDA=1，LDB=0，ALU-B=1，按动时序单元的ST按钮，产生一个T4上沿，则将二进制数01100101置入暂存器A中，暂存器A的值通过ALU单元的A7…A0八位LED灯显示。

(5) 用输入开关向暂存器B置数。

① 拨动CON单元的SD27…SD20数据开关，形成二进制数10100111（或其它数值）。

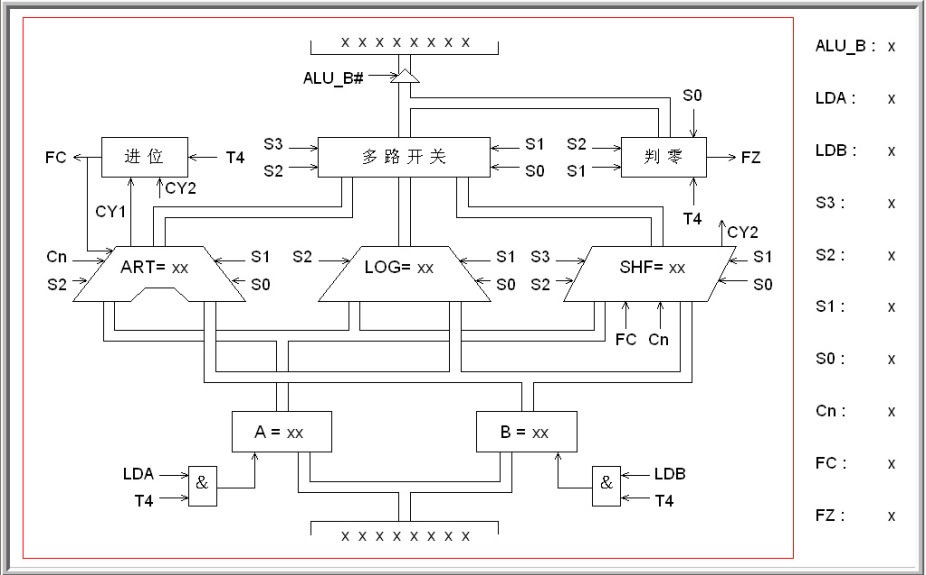
② 置LDA=0，LDB=1，按动时序单元的ST按钮，产生一个T4上沿，则将二进制数10100111

置入暂存器B中，暂存器B的值通过ALU单元的B7…B0八位LED灯显示。

(6) 改变运算器的功能设置，观察运算器的输出。置ALU\_B=0、LDA=0、LDB=0，然后按表1-1置S3、S2、S1、S0和Cn的数值，并观察数据总线LED显示灯显示的结果。如置S3、S2、S1、S0为0010，运算器作逻辑与运算，置S3、S2、S1、S0为1001，运算器作加法运算。

如果实验箱和PC联机操作，则可通过软件中的数据通路图来观测实验结果（软件使用说明请看附录一），方法是：打开软件，选择联机软件的“【实验】—【运算器实验】”，打开运算器实验的数据通路图，如图1-6所示。进行上面的手动操作，每按动一次ST按钮，数据通路图会有数据的流动，反映当前运算器所做的操作，或在软件中选择“【调试】—【单周期】”，其作用相当于将时序单元的状态开关KK2置为‘单步’档后按动了一次ST按钮，数据通路图也会反映当前运算器所做的操作。

重复上述操作，并完成表1-2。然后改变A、B的值，验证FC、FZ的锁存功能。



**图1-6 数据通路图**

**五、实验结果**

**表1-2 运算结果表**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **运算类型** | **A** | **B** | **S3 S2 S1 S0** | **CN** | **结果** |
| 逻辑运算 | 65 | A7 | 0 0 0 0 | X | F=( 65 ) FC=( ) FZ=( ) |
| 65 | A7 | 0 0 0 1 | X | F=( A7 ) FC=( ) FZ=( ) |
|  |  | 0 0 1 0 | X | F=( ) FC=( ) FZ=( ) |
|  |  | 0 0 1 1 | X | F=( ) FC=( ) FZ=( ) |
|  |  | 0 1 0 0 | X | F=( ) FC=( ) FZ=( ) |
| 移位运算 |  |  | 0 1 0 1 | X | F=( ) FC=( ) FZ=( ) |
|  |  | 0 1 1 0 | 0 | F=( ) FC=( ) FZ=( ) |
| 1 | F=( ) FC=( ) FZ=( ) |
|  |  | 0 1 1 1 | 0 | F=( ) FC=( ) FZ=( ) |
| 1 | F=( ) FC=( ) FZ=( ) |
| 算术运算 |  |  | 1 0 0 0 | X | F=( ) FC=( ) FZ=( ) |
|  |  | 1 0 0 1 | X | F=( ) FC=( ) FZ=( ) |
|  |  | 1 0 1 0（FC=0） | X | F=( ) FC=( ) FZ=( ) |
| 1 0 1 0（FC=1） | X | F=( ) FC=( ) FZ=( ) |
|  |  | 1 0 1 1 | X | F=( ) FC=( ) FZ=( ) |
|  |  | 1 1 0 0 | X | F=( ) FC=( ) FZ=( ) |
|  |  | 1 1 0 1 | X | F=( ) FC=( ) FZ=( ) |

## 实验二 静态随机存储器实验

存储器是计算机各种信息存储与交换的中心。在程序执行过程中，所要执行的指令是从存储器中获取，运算器所需要的操作数是通过程序中的访问存储器指令从存储器中得到，运算结果在程序执行完之前又必须全部写到存储器中，各种输入输出设备也直接与存储器交换数据。把程序和数据存储在存储器中，是冯·诺依曼型计算机的基本特征，也是计算机能够自动、连续快速工作的基础。

### 一、 实验目的

掌握静态随机存储器RAM工作特性及数据的读写方法。

### 二、 实验设备

PC机一台，TD-CMA实验系统一套。

### 三、 实验原理

实验所用的静态存储器由一片6116（2K×8bit）构成（位于MEM单元），如图2-1所示。6116有三个控制线：CS（片选线）、OE（读线）、WE（写线），其功能如表2-1所示，当片选有效（CS=0）时，OE=0时进行读操作，WE=0时进行写操作，本实验将CS常接地。



图2-1 SRAM 6116引脚图

由于存储器（MEM）最终是要挂接到CPU上，所以其还需要一个读写控制逻辑，使得CPU能控制MEM的读写，实验中的读写控制逻辑如图2-2所示，由于T3的参与，可以保证MEM的写脉宽与T3一致，T3由时序单元的TS3给出。IOM用来选择是对I/O还是对MEM进行读写操作，RD=1时为读，WR=1时为写。

表2-1 SRAM 6116功能表





图2-2 读写控制逻辑

实验原理图如图2-3所示，存储器数据线接至数据总线，数据总线上接有8个LED灯显示D7…D0的内容。地址线接至地址总线，地址总线上接有8个LED灯显示A7…A0的内容，地址由地址锁存器（74LS273，位于PC&AR单元）给出。数据开关（位于IN单元）经一个三态门（74LS245）连至数据总线，分时给出地址和数据。地址寄存器为8位，接入6116的地址A7…A0，6116的高三位地址A10…A8接地，所以其实际容量为256字节。



图2-3 存储器实验原理图

实验箱中所有单元的时序都连接至时序与操作台单元，CLR都连接至CON单元的CLR按钮。实验时T3由时序单元给出，其余信号由CON单元的二进制开关模拟给出，其中IOM应为低（即MEM操作），RD、WR高有效，MR和MW低有效，LDAR高有效。

### 四、实验步骤

(1) 关闭实验系统电源，按图2-4连接实验电路，并检查无误，图中将用户需要连接的信号用圆圈标明。

(2) 将时序与操作台单元的开关KK1、KK3置为运行档、开关KK2置为‘单步’档。

(3) 将CON单元的IOR开关置为1（使IN单元无输出），打开电源开关，如果听到有‘嘀’报警声，说明有总线竞争现象，应立即关闭电源，重新检查接线，直到错误排除。



图2-4 实验接线图

(4) 给存储器的00H、01H、02H、03H、04H地址单元中分别写入数据11H、12H、13H、14H、15H。由前面的存储器实验原理图（图2-3）可以看出，由于数据和地址由同一个数据开关给出，因此数据和地址要分时写入，先写地址，具体操作步骤为：先关掉存储器的读写（WR=0，RD=0），数据开关输出地址（IOR=0），然后打开地址寄存器门控信号（LDAR=1），按动ST产生T3脉冲，即将地址打入到AR中。再写数据，具体操作步骤为：先关掉存储器的读写（WR=0，RD=0）和地址寄存器门控信号（LDAR=0），数据开关输出要写入的数据，打开输入三态门（IOR=0），然后使存储器处于写状态（WR=1，RD=0，IOM=0），按动ST产生T3脉冲，即将数据打入到存储器中。写存储器的流程如图2-5所示（以向00地址单元写入11H为例）：



图2-5 写存储器流程图

(5) 依次读出第00、01、02、03、04号单元中的内容，观察上述各单元中的内容是否与前面写入的一致。同写操作类似，也要先给出地址，然后进行读，地址的给出和前面一样，而在进行读操作时，应先关闭IN单元的输出（IOR=1），然后使存储器处于读状态（WR=0，RD=1，IOM=0），此时数据总线上的数即为从存储器当前地址中读出的数据内容。读存储器的流程如图2-6所示（以从00地址单元读出11H为例）：



图2-6 读存储器流程图

如果实验箱和PC联机操作，则可通过软件中的数据通路图来观测实验结果（软件使用说明请看附录1），方法是：打开软件，选择联机软件的“【实验】—【存储器实验】”，打开存储器实验的数据通路图，如图2-7所示。

进行上面的手动操作，每按动一次ST按钮，数据通路图会有数据的流动，反映当前存储器所做的操作（即使是对存储器进行读，也应按动一次ST按钮，数据通路图才会有数据流动），或在软件中选择“【调试】—【单周期】”，其作用相当于将时序单元的状态开关置为‘单步’档后按动了一次ST按钮，数据通路图也会反映当前存储器所做的操作，借助于数据通路图，仔细分析SRAM的读写过程。

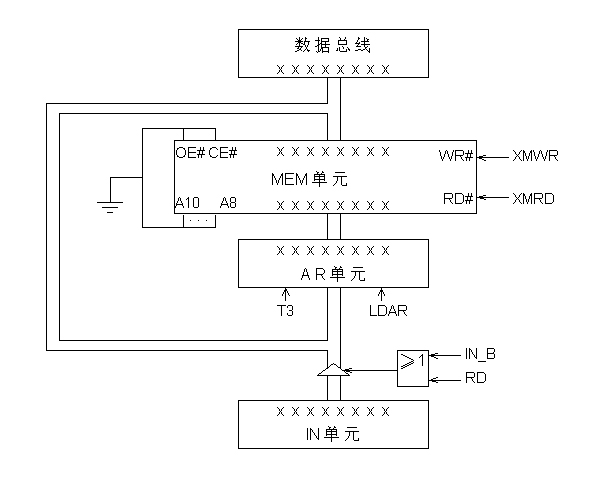


图2-7 数据通路图

五、实验结果

# 实验三 系统总线与总线接口

总线是计算机中连接各个功能部件的纽带，是计算机各部件之间进行信息传输的公共通路。总线不只是一组简单的信号传输线，它还是一组协议。分时与共享是总线的两大特征。所谓共享，在总线上可以挂接多个部件，它们都可以使用这一信息通路来和其他部件传送信息。所谓分时，同一总线在同一时刻，只能有一个部件占领总线发送信息，其他部件要发送信息得在该部件发送完释放总线后才能申请使用。总线结构是决定计算机性能、功能、可扩展性和标准化程度的重要因素。

### 一、实验目的

1．理解总线的概念及其特性。

2．掌握控制总线的功能和应用。

### 二、实验设备

PC机一台，TD-CMA实验系统一套。

### 三、实验原理

由于存储器和输入、输出设备最终是要挂接到外部总线上，所以需要外部总线提供数据信号、地址信号以及控制信号。在该实验平台中，外部总线分为数据总线、地址总线、和控制总线，分别为外设提供上述信号。外部总线和CPU内总线之间通过三态门连接，同时实现了内外总线的分离和对于数据流向的控制。地址总线可以为外部设备提供地址信号和片选信号。由地址总线的高位进行译码，系统的I/O地址译码原理见图3-1（在地址总线单元）。由于使用A6、A7进行译码， I/O地址空间被分为四个区，如表3-1所示：



图3-1 I/O地址译码原理图

表3-1 I/O地址空间分配

|  |  |  |
| --- | --- | --- |
| A7 A6 | 选定 | 地址空间 |
| 00 | IOY0 | 00-3F |
| 01 | IOY1 | 40-7F |
| 10 | IOY2 | 80-BF |
| 11 | IOY3 | C0-FF |

为了实现对于MEM和外设的读写操作，还需要一个读写控制逻辑，使得CPU能控制MEM和I/O设备的读写，实验中的读写控制逻辑如图3-2所示，由于T3的参与，可以保证写脉宽与T3一致，T3由时序单元的TS3给出。IOM用来选择是对I/O设备还是对MEM进行读写操作，IOM=1时对I/O设备进行读写操作，IOM=0时对MEM进行读写操作。RD=1时为读，WR=1时为写。



图3-2 读写控制逻辑

在理解读写控制逻辑的基础上我们设计一个总线传输的实验。实验所用总线传输实验框图如图3-3所示，它将几种不同的设备挂至总线上，有存储器、输入设备、输出设备、寄存器。这些设备都需要有三态输出控制，按照传输要求恰当有序的控制它们，就可实现总线信息传输。



图3-3 总线传输实验框图

### 四、实验步骤

基本输入输出功能的总线接口实验。

（1）根据挂在总线上的几个基本部件，设计一个简单的流程：

① 输入设备将一个数打入R0寄存器。

② 输入设备将另一个数打入地址寄存器。

③ 将R0寄存器中的数写入到当前地址的存储器中。

④ 将当前地址的存储器中的数用LED数码管显示。

（2）按照图3-4实验接线图进行连线。

（3）具体操作步骤图示如下：

进入软件界面，选择菜单命令“【实验】—【简单模型机】”，打开简单模型机实验数据通路图。

将时序与操作台单元的开关KK1、KK3置为‘运行’档，开关KK2置为‘单拍’档，CON单元所有开关置0（由于总线有总线竞争报警功能，在操作中应当先关闭应关闭的输出开关，再打开应打开的输出开关，否则可能由于总线竞争导致实验出错）， 按动CON单元的总清按钮CLR，然后通过运行程序，在数据通路图中观测程序的执行过程。

1. 输入设备将11H打入R0寄存器。

将IN单元置00010001，K7置为1，关闭R0寄存器的输出；K6置为1，打开R0寄存器的输入；WR、RD、IOM分别置为0、1、1，对IN单元进行读操作；LDAR置为0，不将数据总线的数打入地址寄存器。连续四次点击图形界面上的“单节拍运行”按扭（运行一个机器周期），观察图形界面，在T4时刻完成对寄存器R0的写入操作。

1. 将R0中的数据11H打入存储器01H单元。

将IN单元置00000001（或其他数值）。K7置为1，关闭R0寄存器的输出；K6置为0，关闭R0寄存器的输入；WR、RD、IOM分别置为0、1、1，对IN单元进行读操作；LDAR置为1，将数据总线的数打入地址寄存器。连续四次点击图形界面上的“单节拍运行”按扭，观察图形界面，在T3时刻完成对地址寄存器的写入操作。

先将WR、RD、IOM分别置为1、0、0，对存储器进行写操作；再把K7置为0，打开R0寄存器的输出；K6置为0，关闭R0寄存器的输入； LDAR置为0，不将数据总线的数打入地址寄存器。连续四次点击图形界面上的“单节拍运行”按扭，观察图形界面，在T3时刻完成对存储器的写入操作。

1. 将当前地址的存储器中的数写入到R0寄存器中。

将IN单元置00000001（或其他数值），K7置为1，关闭R0寄存器的输出；K6置为0，关闭R0寄存器的输入；WR、RD、IOM分别置为0、1、1，对IN单元进行读操作；LDAR置为1，不将数据总线的数打入地址寄存器。连续四次点击图形界面上的“单节拍运行”按扭，观察图形界面，在T3时刻完成对地址寄存器的写入操作。

将K7置为1，关闭R0寄存器的输出；K6置为1，打开R0寄存器的输入；WR、RD、IOM分别置为0、1、0，对存储器进行读操作；LDAR置为0，不将数据总线的数打入地址寄存器。连续四次点击图形界面上的“单节拍运行”按扭，观察图形界面，在T3时刻完成对寄存器R0的写入操作。



**图3-4 实验接线图**

**注：由于采用简单模型机的数据通路图，为了不让悬空的信号引脚影响通路图的显示结果，将这些引脚置为无效。在接线时为了方便，可将管脚接到CON单元闲置的开关上，若开关打到‘1’，等效于接到‘VCC’；若开关打到‘0’，等效于接到‘GND’。**

1. 将R0寄存器中的数用LED数码管显示。

先将WR、RD、IOM分别置为1、0、1，对OUT单元进行写操作；再将 K7置为0，打开R0寄存器的输出；K6置为0，关闭R0寄存器的输入； LDAR置为0，不将数据总线的数打入地址寄存器。连续四次点击图形界面上的“单节拍运行”按扭，观察图形界面，在T3时刻完成对OUT单元的写入操作。



五、实验结果

## 实验四 微程序控制器实验

### 一、 实验目的

(1) 掌握微程序控制器的组成原理。

(2) 掌握微程序的编制、写入，观察微程序的运行过程。

### 二、 实验设备

PC机一台，TD-CMA实验系统一套。

### 三、 实验原理

微程序控制器的基本任务是完成当前指令的翻译和执行，即将当前指令的功能转换成可以控制的硬件逻辑部件工作的微命令序列，完成数据传送和各种处理操作。它的执行方法就是将控制各部件动作的微命令的集合进行编码，即将微命令的集合仿照机器指令一样，用数字代码的形式表示，这种表示称为微指令。这样就可以用一个微指令序列表示一条机器指令，这种微指令序列称为微程序。微程序存储在一种专用的存储器中，称为控制存储器,微程序控制器原理框图如图4-1所示。



图4-1 微程序控制器组成原理框图

控制器是严格按照系统时序来工作的，因而时序控制对于控制器的设计是非常重要的，从前面的实验可以很清楚地了解时序电路的工作原理，本实验所用的时序由时序单元来提供，分为四拍TS1、TS2、TS3、TS4。

微程序控制器的组成见图4-2，其中控制存储器采用3片2816的E2PROM，具有掉电保护功能，微命令寄存器18位，用两片8D触发器（273）和一片4D（175）触发器组成。微地址寄存器6位，用三片正沿触发的双D触发器（74）组成，它们带有清“0”端和预置端。在不判别测试的情况下，T2时刻打入微地址寄存器的内容即为下一条微指令地址。当T4时刻进行测试判别时，转移逻辑满足条件后输出的负脉冲通过强置端将某一触发器置为“1”状态，完成地址修改。

**图4-2 微程序控制器原理图**

在实验平台中设有一组编程控制开关KK3、KK4、KK5（位于时序与操作台单元），可实现对存储器（包括存储器和控制存储器）的三种操作：编程、校验、运行。考虑到对于存储器（包括存储器和控制存储器）的操作大多集中在一个地址连续的存储空间中，实验平台提供了便利的手动操作方式。以向00H单元中写入332211为例，对于控制存储器进行编辑的具体操作步骤如下：首先将KK1拨至‘停止’档、KK3拨至‘编程’档、KK4拨至‘控存’档、KK5拨至‘置数’档，由CON单元的SD05——SD00开关给出需要编辑的控存单元首地址（000000），IN单元开关给出该控存单元数据的低8位（00010001），连续两次按动时序与操作台单元的开关ST（第一次按动后MC单元低8位显示该单元以前存储的数据，第二次按动后显示当前改动的数据），此时MC单元的指示灯MA5——MA0显示当前地址（000000），M7——M0显示当前数据（00010001）。然后将KK5拨至‘加1’档，IN单元开关给出该控存单元数据的中8位（00100010），连续两次按动开关ST，完成对该控存单元中8位数据的修改，此时MC单元的指示灯MA5——MA0显示当前地址（000000），M15——M8显示当前数据（00100010）；再由IN单元开关给出该控存单元数据的高8位（00110011），连续两次按动开关ST，完成对该控存单元高8位数据的修改此时MC单元的指示灯MA5——MA0显示当前地址（000000），M23——M16显示当前数据（00110011）。此时被编辑的控存单元地址会自动加1（01H），由IN单元开关依次给出该控存单元数据的低8位、中8位和高8位配合每次开关ST的两次按动，即可完成对后续单元的编辑。



编辑完成后需进行校验，以确保编辑的正确。以校验00H单元为例，对于控制存储器进行校验的具体操作步骤如下：首先将KK1拨至‘停止’档、KK3拨至‘校验’档、KK4拨至‘控存’档、KK5拨至‘置数’档。由CON单元的SD05——SD00开关给出需要校验的控存单元地址（000000），连续两次按动开关ST，MC单元指示灯M7——M0显示该单元低8位数据（00010001）；KK5拨至‘加1’档，再连续两次按动开关ST，MC单元指示灯M15——M8显示该单元中8位数据（00100010）；再连续两次按动开关ST，MC单元指示灯M23——M16显示该单元高8位数据（00110011）。再连续两次按动开关ST，地址加1，MC单元指示灯M7——M0显示01H单元低8位数据。如校验的微指令出错，则返回输入操作，修改该单元的数据后再进行校验，直至确认输入的微代码全部准确无误为止，完成对微指令的输入。



位于实验平台MC单元左上角一列三个指示灯MC2、MC1、MC0用来指示当前操作的微程序字段，分别对应M23——M16、M15——M8、M7——M0。实验平台提供了比较灵活的手动操作方式，比如在上述操作中在对地址置数后将开关KK4拨至‘减1’档，则每次随着开关ST的两次拨动操作，字节数依次从高8位到低8位递减，减至低8位后，再按动两次开关ST，微地址会自动减一，继续对下一个单元的操作。

微指令字长共24位，控制位顺序如表4-1：

表4-1 微指令格式



其中MA5…MA0为6位的后续微地址，A、B、C为三个译码字段，分别由三个控制位译码出多位。C字段中的P<1>为测试字位。其功能是根据机器指令及相应微代码进行译码，使微程序转入相应的微地址入口，从而实现完成对指令的识别，并实现微程序的分支，本系统上的指令译码原理如图4-3所示，图中I7…I2为指令寄存器的第7…2位输出，SE5…SE0为微控器单元微地址锁存器的强置端输出，指令译码逻辑在IR单元的INS\_DEC（GAL20V8）中实现。

从图4-2中也可以看出，微控器产生的控制信号比表3-1中的要多，这是因为实验的不同，所需的控制信号也不一样，本实验只用了部分的控制信号。

本实验除了用到指令寄存器（IR）和通用寄存器R0外，还要用到IN和OUT单元，从微控器出来的信号中只有IOM、WR和RD三个信号，所以对这两个单元的读写信号还应先经过译码，其译码原理如图4-4所示。IR单元的原理图如图4-5所示，R0单元原理如图4-7所示，IN单元的原理图见图4-3所示，OUT单元的原理图见图4-6所示。



图4-3 指令译码原理图



图4-4 读写控制逻辑 图4-5 IR单元原理图



图4-6 OUT单元原理图 图4-7 R0原理图

本实验安排了四条机器指令，分别为ADD（0000 0000）、IN（0010 0000）、OUT（0011 0000）和HLT（0101 0000），括号中为各指令的二进制代码，指令格式如下：



实验中机器指令由CON单元的二进制开关手动给出，其余单元的控制信号均由微程序控制器自动产生，为此可以设计出相应的数据通路图，见图4-8所示。

几条机器指令对应的参考微程序流程图如图4-9所示。图中一个矩形方框表示一条微指令，方框中的内容为该指令执行的微操作，右上角的数字是该条指令的微地址，右下角的数字是该条指令的后续微地址，所有微地址均用16进制表示。向下的箭头指出了下一条要执行的指令。P<1>为测试字，根据条件使微程序产生分支。

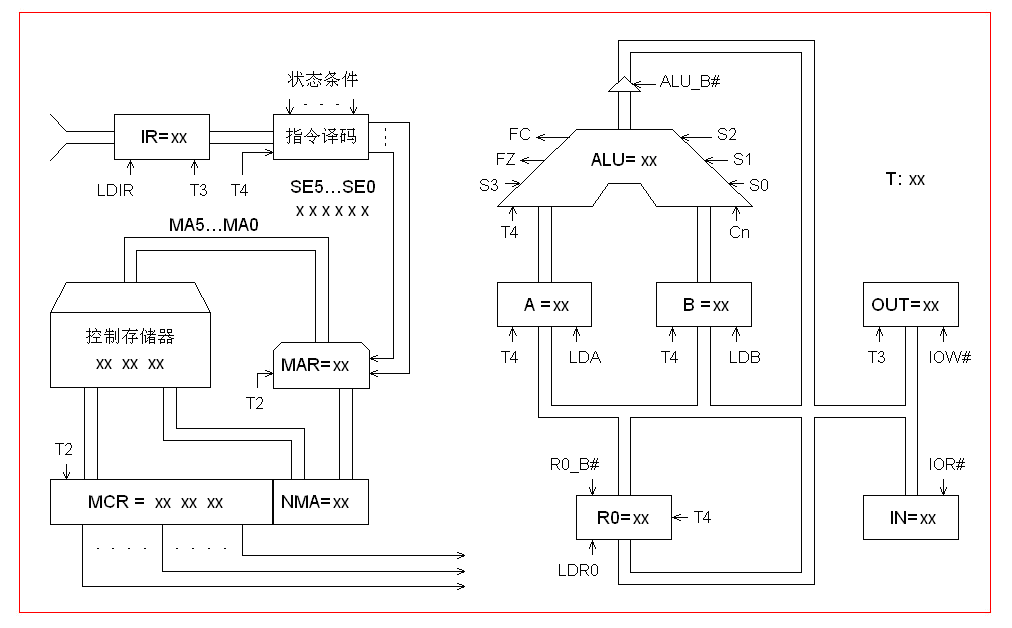


图4-8 数据通路图



图4-9 微程序流程图

将全部微程序按微指令格式变成二进制微代码，可得到表4-2的二进制代码表。

表4-2 二进制微代码表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **地址** | **十六进制** | **高五位** | S3-S0 | **A字段** | **B字段** | **C 字段** | **MA5-MA0** |
| 00 | 00 00 01 | 00000 | 0000 | 000 | 000 | 000 | 000001 |
| 01 | 00 70 70 | 00000 | 0000 | 111 | 000 | 001 | 110000 |
| 04 | 00 24 05 | 00000 | 0000 | 010 | 010 | 000 | 000101 |
| 05 | 04 B2 01 | 00000 | 1001 | 011 | 001 | 000 | 000001 |
| 30 | 00 14 04 | 00000 | 0000 | 001 | 010 | 000 | 000100 |
| 32 | 18 30 01 | 00011 | 0000 | 011 | 000 | 000 | 000001 |
| 33 | 28 04 01 | 00101 | 0000 | 000 | 010 | 000 | 000001 |
| 35 | 00 00 35 | 00000 | 0000 | 000 | 000 | 000 | 110101 |

### 四、 实验步骤

1. 按图4-10所示连接实验线路，仔细查线无误后接通电源。如果有‘滴’报警声，说明总线有竞争现象，应关闭电源，检查接线，直到错误排除。



图4-10 实验接线图

2. 对微控器进行读写操作，分两种情况：手动读写和联机读写。

1) 手动读写

(1) 手动对微控器进行编程（写）

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘编程’档，KK4置为‘控存’档，KK5置为‘置数’档。

② 使用CON单元的SD05——SD00给出微地址，IN单元给出低8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的低8位。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ IN单元给出中8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的中8位。IN单元给出高8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的高8位。

⑤ 重复①、②、③、④四步，将表4-2的微代码写入2816芯片中。

(2) 手动对微控器进行校验（读）

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘校验’档，KK4置为‘控存’档，KK5置为‘置数’档。

② 使用CON单元的SD05——SD00给出微地址，连续两次按动时序与操作台的开关ST，MC单元的指数据指示灯 M7——M0显示该单元的低8位。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ 连续两次按动时序与操作台的开关ST，MC单元的指数据指示灯 M15——M8显示该单元的中8位，MC单元的指数据指示灯 M23——M16显示该单元的高8位。

⑤ 重复①、②、③、④四步，完成对微代码的校验。如果校验出微代码写入错误，重新写入、校验，直至确认微指令的输入无误为止。

2) 联机读写

(1) 将微程序写入文件

联机软件提供了微程序下载功能，以代替手动读写微控器，但微程序得以指定的格式写入到以TXT为后缀的文件中，微程序的格式如下：



如$M 1F 112233，表示微指令的地址为1FH，微指令值为11H（高）、22H（中）、33H（低），本次实验的微程序如下，其中分号‘；’为注释符，分号后面的内容在下载时将被忽略掉。

(2) 写入微程序

用联机软件的“【转储】—【装载】”功能将该格式（\*.TXT）文件装载入实验系统。装入过程中，在软件的输出区的‘结果’栏会显示装载信息，如当前正在装载的是机器指令还是微指令，还剩多少条指令等。

(3) 校验微程序

选择联机软件的“【转储】—【刷新指令区】”可以读出下位机所有的机器指令和微指令，并在指令区显示。检查微控器相应地址单元的数据是否和表3-2中的十六进制数据相同，如果不同，则说明写入操作失败，应重新写入，可以通过联机软件单独修改某个单元的微指令，先用鼠标左键单击指令区的‘微存’TAB按钮，然后再单击需修改单元的数据，此时该单元变为编辑框，输入6位数据并回车，编辑框消失，并以红色显示写入的数据。

; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* //

; // //

; // 微控器实验指令文件 //

; // //

; // By TangDu CO.,LTD //

; // //

; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* //

; //\*\*\*\* Start Of MicroController Data \*\*\*\* //

$M 00 000001 ; NOP

$M 01 007070 ; CON(INS)->IR, P<1>

$M 04 002405 ; R0->B

$M 05 04B201 ; A加B->R0

$M 30 001404 ; R0->A

$M 32 183001 ; IN->R0

$M 33 280401 ; R0->OUT

$M 35 000035 ; NOP

; //\*\*\*\*\* End Of MicroController Data \*\*\*\*\* //

3. 运行微程序

运行时也分两种情况：本机运行和联机运行。

1) 本机运行

① 将时序与操作台单元的开关KK1、KK3置为‘运行’档，按动CON单元的CLR按钮，将微地址寄存器（MAR）清零，同时也将指令寄存器（IR）、ALU单元的暂存器A和暂存器B清零。

② 将时序与操作台单元的开关KK2置为‘单拍’档，然后按动ST按钮，体会系统在T1、 T2、T3、T4节拍中各做的工作。T2节拍微控器将后续微地址（下条执行的微指令的地址）打入微地址寄存器，当前微指令打入微指令寄存器，并产生执行部件相应的控制信号；T3、T4节拍根据T2节拍产生的控制信号做出相应的执行动作，如果测试位有效，还要根据机器指令及当前微地址寄存器中的内容进行译码，使微程序转入相应的微地址入口，实现微程序的分支。

③ 按动CON单元的CLR按钮，清微地址寄存器（MAR）等，并将时序与单元的开关KK2置为‘单步’档。

④ 置IN单元数据为00100011，按动ST按钮，当MC单元后续微地址显示为000001时，在CON单元的SD27…SD20模拟给出IN指令00100000并继续单步执行，当MC单元后续微地址显示为000001时，说明当前指令已执行完；在CON单元的SD27…SD20给出ADD指令00000000，该指令将会在下个T3被打入指令寄存器（IR），它将R0中的数据和其自身相加后送R0；接下来在CON单元的SD27…SD20给出OUT指令00110000并继续单步执行，在MC单元后续微地址显示为000001时，观查OUT单元的显示值是否为01000110。

2) 联机运行

联机运行时，进入软件界面，在菜单上选择【实验】－【微控器实验】，打开本实验的数据通路图，也可以通过工具栏上的下拉框打开数据通路图，数据通路图如图4-8所示。

将时序与操作台单元的开关KK1、KK3置为‘运行’档，按动CON单元的总清开关后，按动软件中单节拍按钮，当后续微地址（通路图中的MAR）为000001时，置CON单元SD27…SD20，产生相应的机器指令，该指令将会在下个T3被打入指令寄存器（IR），在后面的节拍中将执行这条机器指令。仔细观察每条机器指令的执行过程，体会后续微地址被强置转换的过程，这是计算机识别和执行指令的根基。也可以打开微程序流程图，跟踪显示每条机器指令的执行过程。

按本机运行的顺序给出数据和指令，观查最后的运算结果是否正确。

## 实验五 简单模型机设计实验

在前面的章节中，我们重点讨论计算机中每个部件的组成及特性，本实验中，我们将重点讨论如何完整设计一台模型计算机，进一步建立整机的概念。

### 一、实验目的

(1) 掌握一个简单CPU的组成原理。

(2) 在掌握部件单元电路的基础上，进一步将其构造一台基本模型计算机。

(3) 为其定义五条机器指令，编写相应的微程序，并上机调试掌握整机概念。

### 二、实验设备

PC机一台，TD-CMA实验系统一套。

### 三、实验原理

本实验要实现一个简单的CPU，并且在此CPU的基础上，继续构建一个简单的模型计算机。CPU由运算器（ALU）、微程序控制器（MC）、通用寄存器（R0），指令寄存器（IR）、程序计数器（PC）和地址寄存器（AR）组成,如图5-1所示。这个CPU在写入相应的微指令后，就具备了执行机器指令的功能，但是机器指令一般存放在主存当中，CPU必须和主存挂接后，才有实际的意义，所以还需要在该CPU的基础上增加一个主存和基本的输入输出部件，以构成一个简单的模型计算机。



图5-1 基本CPU构成原理图

除了程序计数器（PC），其余部件在前面的实验中都已用到，在此不再讨论。系统的程序计数器（PC）和地址寄存器（AR）集成在一片CPLD芯片中 。CLR连接至CON单元的总清端CLR，按下CLR按钮，将使PC清零，LDPC和T3相与后作为计数器的计数时钟，当LOAD为低时，计数时钟到来后将CPU内总线上的数据打入PC。



图5-2 程序计数器(PC)原理图

本模型机和前面微程序控制器实验相比，新增加一条跳转指令JMP，共有五条指令：IN（输入）、ADD（二进制加法）、OUT（输出）、JMP（无条件转移），HLT（停机），其指令格式如下（高４位为操作码）：



其中JMP为双字节指令，其余均为单字节指令，\*\*\*\*\*\*\*\*为addr对应的二进制地址码。微程序控制器实验的指令是通过手动给出的，现在要求CPU自动从存储器读取指令并执行。根据以上要求，设计数据通路图，如图5-3所示。

本实验在前一个实验的基础上增加了三个部件，一是PC（程序计数器），另一个是AR（地址寄存器），还有就是MEM（主存）。因而在微指令中应增加相应的控制位，其微指令格式如表5-1所示。

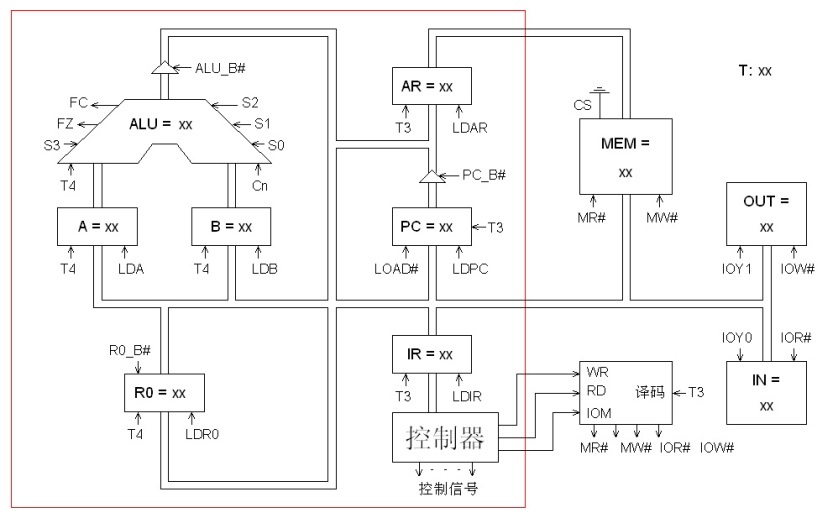


图5-3 数据通路图

表5-1 微指令格式



系统涉及到的微程序流程见图5-4所示，当拟定“取指”微指令时，该微指令的判别测试字段为P<1>测试。指令译码原理见图4-3所示，由于“取指”微指令是所有微程序都使用的公用微指令，因此P<1> 的测试结果出现多路分支。本机用指令寄存器的高6位（IR7—IR2）作为测试条件，出现５路分支，占用５个固定微地址单元，剩下的其它地方就可以一条微指令占用控存一个微地址单元随意填写，微程序流程图上的单元地址为16进制。



图5-4 简单模型机微程序流程图

当全部微程序设计完毕后，应将每条微指令代码化，表5-2即为将图5-4的微程序流程图按微指令格式转化而成的“二进制微代码表”。

表5-2 二进制微代码表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 十六进制 | 高五位 | S3-S0 | A字段 | B字段 | C 字段 | MA5-MA0 |
| 00 | 00 00 01 | 00000 | 0000 | 000 | 000 | 000 | 000001 |
| 01 | 00 6D 43 | 00000 | 0000 | 110 | 110 | 101 | 000011 |
| 03 | 10 70 70 | 00010 | 0000 | 111 | 000 | 001 | 110000 |
| 04 | 00 24 05 | 00000 | 0000 | 010 | 010 | 000 | 000101 |
| 05 | 04 B2 01 | 00000 | 1001 | 011 | 001 | 000 | 000001 |
| 1D | 10 51 41 | 00010 | 0000 | 101 | 000 | 101 | 000001 |
| 30 | 00 14 04 | 00000 | 0000 | 001 | 010 | 000 | 000100 |
| 32 | 18 30 01 | 00011 | 0000 | 011 | 000 | 000 | 000001 |
| 33 | 28 04 01 | 00101 | 0000 | 000 | 010 | 000 | 000001 |
| 35 | 00 00 35 | 00000 | 0000 | 000 | 000 | 000 | 110101 |
| 3C | 00 6D 5D | 00000 | 0000 | 110 | 110 | 101 | 011101 |

设计一段机器程序，要求从IN单元读入一个数据，存于R0，将R0和自身相加，结果存于R0，再将R0的值送OUT单元显示。

根据要求可以得到如下程序，地址和内容均为二进制数。

**地 址 内 容 助记符 说 明**

00000000 00100000 ; START: IN R0 从IN单元读入数据送R0

00000001 00000000 ; ADD R0,R0 R0和自身相加，结果送R0

00000010 00110000 ; OUT R0 R0的值送OUT单元显示

00000011 11100000 ; JMP START 跳转至00H地址

00000100 00000000 ;

00000101 01010000 ; HLT 停机

### 四、实验步骤

1. 按图5-5连接实验线路。

2. 写入实验程序，并进行校验，分两种方式，手动写入和联机写入。

1) 手动写入和校验

(1) 手动写入微程序

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘编程’档，KK4置为‘控存’档，KK5置为‘置数’档。

② 使用CON单元的SD05——SD00给出微地址，IN单元给出低8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的低8位。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ IN单元给出中8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的中8位。IN单元给出高8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的高8位。

⑤ 重复①、②、③、④四步，将表5-2的微代码写入2816芯片中。



图5-5 实验接线图

(2) 手动校验微程序

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘校验’档，KK4置为‘控存’档，KK5置为‘置数’档。

② 使用CON单元的SD05——SD00给出微地址，连续两次按动时序与操作台的开关ST，MC单元的指数据指示灯 M7——M0显示该单元的低8位。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ 连续两次按动时序与操作台的开关ST，MC单元的指数据指示灯 M15——M8显示该单元的中8位，MC单元的指数据指示灯 M23——M16显示该单元的高8位。

⑤ 重复①、②、③、④四步，完成对微代码的校验。如果校验出微代码写入错误，重新写入、校验，直至确认微指令的输入无误为止。

1. 手动写入机器程序

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘编程’档，KK4置为‘主存’档，KK5置为‘置数’档。

② 使用CON单元的SD07——SD00给出地址，IN单元给出该单元应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该存储器单元。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ IN单元给出下一地址（地址自动加1）应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元中。然后地址会又自加1，只需在IN单元输入后续地址的数据，连续两次按动时序与操作台的开关ST，即可完成对该单元的写入。

⑤ 亦可重复①、②两步，将所有机器指令写入主存芯片中。

1. 手动校验机器程序

①将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘校验’档，KK4置为‘主存’档，KK5置为‘置数’档。

② 使用CON单元的SD07——SD00给出地址，连续两次按动时序与操作台的开关ST，CPU内总线的指数据指示灯 D7——D0显示该单元的数据。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ 连续两次按动时序与操作台的开关ST，地址自动加1，CPU内总线的指数据指示灯 D7——D0显示该单元的数据。此后每两次按动时序与操作台的开关ST，地址自动加1，CPU内总线的指数据指示灯 D7——D0显示该单元的数据，继续进行该操作，直至完成校验，如发现错误，则返回写入，然后校验，直至确认输入的所有指令准确无误。

⑤ 亦可重复①、②两步，完成对指令码的校验。如果校验出指令码写入错误，重新写入、校验，直至确认指令码的输入无误为止。

2) 联机写入和校验

联机软件提供了微程序和机器程序下载功能，以代替手动读写微程序和机器程序，但是微程序和机器程序得以指定的格式写入到以TXT为后缀的文件中，微程序和机器程序的格式如下：

本次实验程序如下，程序中分号‘；’为注释符，分号后面的内容在下载时将被忽略掉：

; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* //

; // //

; // CPU与简单模型机实验指令文件 //

; // //

; // By TangDu CO.,LTD //

; // //

; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* //

; //\*\*\*\*\*\* Start Of Main Memory Data \*\*\*\*\*\* //

$P 00 20 ; START: IN R0 从IN单元读入数据送R0

$P 01 00 ; ADD R0,R0 R0和自身相加，结果送R0

$P 02 30 ; OUT R0 R0的值送OUT单元显示

$P 03 E0 ; JMP START 跳转至00H地址

$P 04 00 ;

$P 05 50 ; HLT 停机

; //\*\*\*\*\*\*\* End Of Main Memory Data \*\*\*\*\*\*\* //

; //\*\*\*\* Start Of MicroController Data \*\*\*\* //

$M 00 000001 ; NOP

$M 01 006D43 ; PC->AR,PC加1

$M 03 107070 ; MEM->IR, P<1>

$M 04 002405 ; R0->B

$M 05 04B201 ; A加B->R0

$M 1D 105141 ; MEM->PC

$M 30 001404 ; R0->A

$M 32 183001 ; IN->R0

$M 33 280401 ; R0->OUT

$M 35 000035 ; NOP

$M 3C 006D5D ; PC->AR,PC加1

; //\*\* End Of MicroController Data \*\*//

选择联机软件的“【转储】—【装载】”功能，在打开文件对话框中选择上面所保存的文件，软件自动将机器程序和微程序写入指定单元。

选择联机软件的“【转储】—【刷新指令区】”可以读出下位机所有的机器指令和微指令，并在指令区显示，对照文件检查微程序和机器程序是否正确，如果不正确，则说明写入操作失败，应重新写入，可以通过联机软件单独修改某个单元的指令，以修改微指令为例，先用鼠标左键单击指令区的‘微存’TAB按钮，然后再单击需修改单元的数据，此时该单元变为编辑框，输入6位数据并回车，编辑框消失，并以红色显示写入的数据。

3. 运行程序

方法一：本机运行

将时序与操作台单元的开关KK1、KK3置为‘运行’档，按动CON单元的总清按钮CLR，将使程序计数器PC、地址寄存器AR和微程序地址为00H，程序可以从头开始运行，暂存器A、B，指令寄存器IR和OUT单元也会被清零。

将时序与操作台单元的开关KK2置为‘单步’档，每按动一次ST按钮，即可单步运行一条微指令，对照微程序流程图，观察微地址显示灯是否和流程一致。每运行完一条微指令，观测一次CPU内总线和地址总线，对照数据通路图，分析总线上的数据是否正确。

当模型机执行完JMP指令后，检查OUT单元显示的数是否为IN单元值的2倍，按下CON单元的总清按钮CLR，改变IN单元的值，再次执行机器程序，从OUT单元显示的数判别程序执行是否正确。

方法二：联机运行

将时序与操作台单元的开关KK1和KK3置为‘运行’档，进入软件界面，选择菜单命令“【实验】—【简单模型机】”，打开简单模型机数据通路图。

按动CON单元的总清按钮CLR，然后通过软件运行程序，选择相应的功能命令，即可联机运行、监控、调试程序，当模型机执行完JMP指令后，检查OUT单元显示的数是否为IN单元值的2倍。在数据通路图和微程序流中观测指令的执行过程，并观测软件中地址总线、数据总线以及微指令显示和下位机是否一致。

## 实验六 复杂模型机设计实验

### 一、实验目的

综合运用所学计算机组成原理知识，设计并实现较为完整的计算机。

### 二、实验设备

PC机一台，TD-CMA实验系统一套。

### 三、实验原理

下面讲述一下模型计算机的数据格式及指令系统。

**1．数据格式**

模型机规定采用定点补码表示法表示数据，字长为８位，8位全用来表示数据（最高位不表示符号），数值表示范围是： 0≤X≤28－1。

**2．指令设计**

模型机设计三大类指令共十五条，其中包括运算类指令、控制转移类指令，数据传送类指令。运算类指令包含三种运算，算术运算、逻辑运算和移位运算，设计有6条运算类指令，分别为：ADD、AND、INC、SUB、OR、RR，所有运算类指令都为单字节，寻址方式采用寄存器直接寻址。控制转移类指令有三条HLT、JMP、BZC，用以控制程序的分支和转移，其中HLT为单字节指令，JMP和BZC为双字节指令。数据传送类指令有IN、OUT、MOV、LDI、LAD、STA共6条，用以完成寄存器和寄存器、寄存器和I/O、寄存器和存储器之间的数据交换，除MOV指令为单字节指令外，其余均为双字节指令。

**3．指令格式**

所有单字节指令（ADD、AND、INC、SUB、OR、RR、HLT和MOV）格式如下：

|  |  |  |
| --- | --- | --- |
| ７６５４ | ３２ | １０ |
| OP-CODE | RS | RD |

其中，OP-CODE为操作码，RS为源寄存器，RD为目的寄存器，并规定：

|  |  |
| --- | --- |
| RS或RD | 选定的寄存器 |
| 00  01  10  11 | R0  R1  R2  R3 |

IN和OUT的指令格式为：

|  |  |  |  |
| --- | --- | --- | --- |
| ７６５４（1） | ３２（1） | １０（1） | 7—0（2） |
| OP-CODE | RS | RD | P |

其中括号中的1表示指令的第一字节，2表示指令的第二字节，OP-CODE为操作码，RS为源寄存器，RD为目的寄存器，P为I/O端口号，占用一个字节，系统的I/O地址译码原理见图6-1（在地址总线单元）。



图6-1 I/O地址译码原理图

由于用的是地址总线的高两位进行译码，I/O地址空间被分为四个区，如表6-1所示：

表6-1 I/O地址空间分配

|  |  |  |
| --- | --- | --- |
| A7 A6 | 选定 | 地址空间 |
| 00 | IOY0 | 00-3F |
| 01 | IOY1 | 40-7F |
| 10 | IOY2 | 80-BF |
| 11 | IOY3 | C0-FF |

系统设计五种数据寻址方式，即立即、直接、间接、变址和相对寻址，LDI指令为立即寻址，LAD、STA、JMP和BZC指令均具备直接、间接、变址和相对寻址能力。

LDI的指令格式如下，第一字节同前一样，第二字节为立即数。

|  |  |  |  |
| --- | --- | --- | --- |
| ７６５４（1） | ３２（1） | １０（1） | 7—0（2） |
| OP-CODE | RS | RD | data |

LAD、STA、JMP和BZC指令格式如下。

|  |  |  |  |
| --- | --- | --- | --- |
| ７６５４（1） | ３２（1） | １０（1） | 7—0（2） |
| OP-CODE | M | RD | D |

其中M为寻址模式，具体见表6-2，以R2做为变址寄存器RI。

表6-2 寻址方式

|  |  |  |
| --- | --- | --- |
| 寻址模式M | 有效地址E | 说　明 |
| 00  01  10  11 | E = D  E =（D）  E =（RI）+ D  E =（PC）+ D | 直接寻址  间接寻址  RI变址寻址  相对寻址 |

**4．指令系统**

本模型机共有15条基本指令，表6-3列出了各条指令的格式、汇编符号、指令功能。

表6-3 指令描述



### 四、总体设计

本模型机的数据通路框图如图6-2所示。

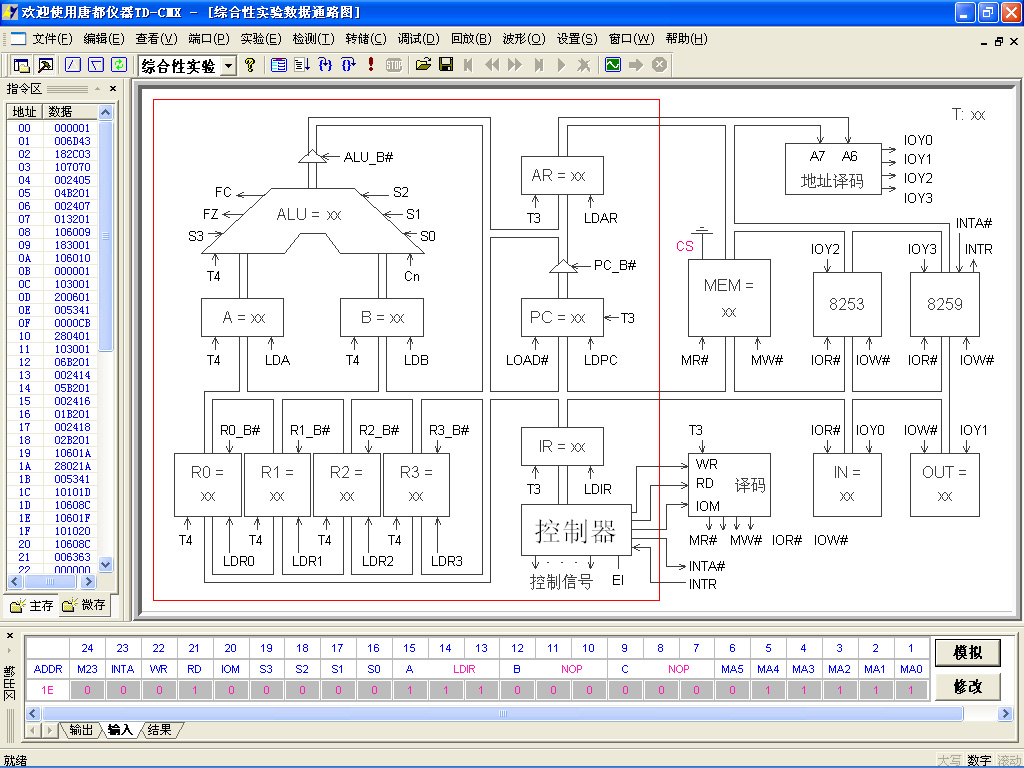


图6-2 数据通路框图

和前面的实验相比，复杂模型机实验指令多，寻址方式多，只用一种测试已不能满足设计要求，为此指令译码电路需要重新设计。如图6-3所示在IR单元的INS\_DEC中实现。





图6-3 指令译码原理图

本实验中要用到四个通用寄存器R3…R0，而对寄存器的选择是通过指令的低四位，为此还得设计一个寄存器译码电路，在IR单元的REG\_DEC（GAL16V8）中实现，如图6-4所示。



图6-4 寄存器译码原理图

根据机器指令系统要求，设计微程序流程图及确定微地址，如图6-5所示。按照系统建议的微指令格式，见表6-4，参照微指令流程图，将每条微指令代码化，译成二进制代码表，见表6-5，并将二进制代码表转换为联机操作时的十六进制格式文件。

表6-4 微指令格式



**图6-5 微程序流程图**

表6-5 二进制代码表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 地址 | 十六进制表示 | 高五位 | S3-S0 | A字段 | B字段 | C 字段 | UA5-UA0 |
| 00 | 00 00 01 | 00000 | 0000 | 000 | 000 | 000 | 000001 |
| 01 | 00 6D 43 | 00000 | 0000 | 110 | 110 | 101 | 000011 |
| 03 | 10 70 70 | 00010 | 0000 | 111 | 000 | 001 | 110000 |
| 04 | 00 24 05 | 00000 | 0000 | 010 | 010 | 000 | 000101 |
| 05 | 04 B2 01 | 00000 | 1001 | 011 | 001 | 000 | 000001 |
| 06 | 00 24 07 | 00000 | 0000 | 010 | 010 | 000 | 000111 |
| 07 | 01 32 01 | 00000 | 0010 | 011 | 001 | 000 | 000001 |
| 08 | 10 60 09 | 00010 | 0000 | 110 | 000 | 000 | 001001 |
| 09 | 18 30 01 | 00011 | 0000 | 011 | 000 | 000 | 000001 |
| 0A | 10 60 10 | 00010 | 0000 | 110 | 000 | 000 | 010000 |
| 0B | 00 00 01 | 00000 | 0000 | 000 | 000 | 000 | 000001 |
| 0C | 10 30 01 | 00010 | 0000 | 011 | 000 | 000 | 000001 |
| 0D | 20 06 01 | 00100 | 0000 | 000 | 011 | 000 | 000001 |
| 0E | 00 53 41 | 00000 | 0000 | 101 | 001 | 101 | 000001 |
| 0F | 00 00 CB | 00000 | 0000 | 000 | 000 | 011 | 001011 |
| 10 | 28 04 01 | 00101 | 0000 | 000 | 010 | 000 | 000001 |
| 11 | 10 30 01 | 00010 | 0000 | 011 | 000 | 000 | 000001 |
| 12 | 06 B2 01 | 00000 | 1101 | 011 | 001 | 000 | 000001 |
| 13 | 00 24 14 | 00000 | 0000 | 010 | 010 | 000 | 010100 |
| 14 | 05 B2 01 | 00000 | 1011 | 011 | 001 | 000 | 000001 |
| 15 | 00 24 16 | 00000 | 0000 | 010 | 010 | 000 | 010110 |
| 16 | 01 B2 01 | 00000 | 0011 | 011 | 001 | 000 | 000001 |
| 17 | 00 24 18 | 00000 | 0000 | 010 | 010 | 000 | 011000 |
| 18 | 02 B2 01 | 00000 | 0101 | 011 | 001 | 000 | 000001 |
| 1B | 00 53 41 | 00000 | 0000 | 101 | 001 | 101 | 000001 |
| 1C | 10 10 1D | 00010 | 0000 | 001 | 000 | 000 | 011101 |
| 1D | 10 60 8C | 00010 | 0000 | 110 | 000 | 010 | 001100 |
| 1E | 10 60 1F | 00010 | 0000 | 110 | 000 | 000 | 011111 |
| 1F | 10 10 20 | 00010 | 0000 | 001 | 000 | 000 | 100000 |
| 20 | 10 60 8C | 00010 | 0000 | 110 | 000 | 010 | 001100 |
| 28 | 10 10 29 | 00010 | 0000 | 001 | 000 | 000 | 101001 |
| 29 | 00 28 2A | 00000 | 0000 | 010 | 100 | 000 | 101010 |
| 2A | 04 E2 2B | 00000 | 1001 | 110 | 001 | 000 | 101011 |
| 2B | 04 92 8C | 00000 | 1001 | 001 | 001 | 010 | 001100 |
| 2C | 10 10 2D | 00010 | 0000 | 001 | 000 | 000 | 101101 |
| 2D | 00 2C 2E | 00000 | 0000 | 010 | 110 | 000 | 101110 |
| 2E | 04 E2 2F | 00000 | 1001 | 110 | 001 | 000 | 101111 |
| 2F | 04 92 8C | 00000 | 1001 | 001 | 001 | 010 | 001100 |
| 30 | 00 16 04 | 00000 | 0000 | 001 | 011 | 000 | 000100 |
| 31 | 00 16 06 | 00000 | 0000 | 001 | 011 | 000 | 000110 |
| 32 | 00 6D 48 | 00000 | 0000 | 110 | 110 | 101 | 001000 |
| 33 | 00 6D 4A | 00000 | 0000 | 110 | 110 | 101 | 001010 |
| 34 | 00 34 01 | 00000 | 0000 | 011 | 010 | 000 | 000001 |
| 35 | 00 00 35 | 00000 | 0000 | 000 | 000 | 000 | 110101 |
| 36 | 00 6D 51 | 00000 | 0000 | 110 | 110 | 101 | 010001 |
| 37 | 00 16 12 | 00000 | 0000 | 001 | 011 | 000 | 010010 |
| 38 | 00 16 13 | 00000 | 0000 | 001 | 011 | 000 | 010011 |
| 39 | 00 16 15 | 00000 | 0000 | 001 | 011 | 000 | 010101 |
| 3A | 00 16 17 | 00000 | 0000 | 001 | 011 | 000 | 010111 |
| 3B | 00 00 01 | 00000 | 0000 | 000 | 000 | 000 | 000001 |
| 3C | 00 6D 5C | 00000 | 0000 | 110 | 110 | 101 | 011100 |
| 3D | 00 6D 5E | 00000 | 0000 | 110 | 110 | 101 | 011110 |
| 3E | 00 6D 68 | 00000 | 0000 | 110 | 110 | 101 | 101000 |
| 3F | 00 6D 6C | 00000 | 0000 | 110 | 110 | 101 | 101100 |

根据现有指令，在模型机上实现以下运算：从IN单元读入一个数据，根据读入数据的低4位值X，求1+2+…+X的累加和，01H到0FH共15个数据存于60H到6EH单元。

根据要求可以得到如下程序，地址和内容均为二进制数。

地 址 内 容 助记符 说 明

00000000 00100000 ; START: IN R0,00H 从IN单元读入计数初值

00000001 00000000

00000010 01100001 ; LDI R1,0FH 立即数0FH送R1

00000011 00001111

00000100 00010100 ; AND R0,R1 得到R0低四位

00000101 01100001 ; LDI R1,00H 装入和初值00H

00000110 00000000

00000111 11110000 ; BZC RESULT 计数值为0则跳转

00001000 00010110

00001001 01100010 ; LDI R2,60H 读入数据始地址

00001010 01100000

00001011 11001011 ; LOOP: LAD R3,[RI],00H 从MEM读入数据送R3，

变址寻址，偏移量为00H

00001100 00000000

00001101 00001101 ; ADD R1,R3 累加求和

00001110 01110010 ; INC RI 变址寄存加1，指向下一数据

00001111 01100011 ; LDI R3,01H 装入比较值

00010000 00000001

00010001 10001100 ; SUB R0,R3

00010010 11110000 ; BZC RESULT 相减为0，表示求和完毕

00010011 00010110

00010100 11100000 ; JMP LOOP 未完则继续

00010101 00001011

00010110 11010001 ; RESULT: STA 70H,R1 和存于MEM的70H单元

00010111 01110000

00011000 00110100 ; OUT 40H,R1 和在OUT单元显示

00011001 01000000

00011010 11100000 ; JMP START 跳转至START

00011011 00000000

00011100 01010000 ; HLT 停机

01100000 00000001 ; 数据

01100001 00000010

01100010 00000011

01100011 00000100

01100100 00000101

01100101 00000110

01100110 00000111

01100111 00001000

01101000 00001001

01101001 00001010

01101010 00001011

01101011 00001100

01101100 00001101

01101101 00001110

01101110 00001111

### 五、实验步骤

1. 按图6-6连接实验线路，仔细检查接线后打开实验箱电源。

2. 写入实验程序，并进行校验，分两种方式，手动写入和联机写入。

1) 手动写入和校验

(1) 手动写入微程序

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘编程’档，KK4置为‘控存’档，KK5置为‘置数’档。

② 使用CON单元的SD05——SD00给出微地址，IN单元给出低8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的低8位。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ IN单元给出中8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的中8位。IN单元给出高8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的高8位。

⑤ 重复①、②、③、④四步，将表6-5的微代码写入2816芯片中。

(2) 手动校验微程序

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘校验’档，KK4置为‘控存’档，KK5置为‘置数’档。

② 使用CON单元的SD05——SD00给出微地址，连续两次按动时序与操作台的开关ST，MC单元的指数据指示灯 M7——M0显示该单元的低8位。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ 连续两次按动时序与操作台的开关ST，MC单元的指数据指示灯 M15——M8显示该单元的中8位，MC单元的指数据指示灯 M23——M16显示该单元的高8位。

⑤ 重复①、②、③、④四步，完成对微代码的校验。如果校验出微代码写入错误，重新写入、校验，直至确认微指令的输入无误为止。

1. 手动写入机器程序

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘编程’档，KK4置为‘主存’档，KK5置为‘置数’档。

② 使用CON单元的SD7——SD0给出地址，IN单元给出该单元应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该存储器单元。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ IN单元给出下一地址（地址自动加1）应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元中。然后地址会又自加1，只需在IN单元输入后续地址的数据，连续两次按动时序与操作台的开关ST，即可完成对该单元的写入。

⑤ 亦可重复①、②两步，将所有机器指令写入主存芯片中。

1. 手动校验机器程序

① 将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘校验’档，KK4置为‘主存’档，KK5置为‘置数’档。

② 使用CON单元的SD7——SD0给出地址，连续两次按动时序与操作台的开关ST，CPU内总线的指数据指示灯 D7——D0显示该单元的数据。

③ 将时序与操作台单元的开关KK5置为‘加1’档。

④ 连续两次按动时序与操作台的开关ST，地址自动加1，CPU内总线的指数据指示灯 D7——D0显示该单元的数据。此后每两次按动时序与操作台的开关ST，地址自动加1，CPU内总线的指数据指示灯 D7——D0显示该单元的数据，继续进行该操作，直至完成校验，如发现错误，则返回写入，然后校验，直至确认输入的所有指令准确无误。

⑤ 亦可重复①、②两步，完成对指令码的校验。如果校验出指令码写入错误，重新写入、校验，直至确认指令的输入无误为止。

2) 联机写入和校验

联机软件提供了微程序和机器程序下载功能，以代替手动读写微程序和机器程序，但是微程序和机器程序得以指定的格式写入到以TXT为后缀的文件中，本次实验程序如下，程序中分号‘；’为注释符，分号后面的内容在下载时将被忽略掉。



图5-6 实验接线图

; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* //

; // //

; // 复杂模型机实验指令文件 //

; // //

; // By TangDu CO.,LTD //

; // //

; //\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* //

; //\*\*\*\*\*\* Start Of Main Memory Data \*\*\*\*\*\* //

$P 00 20 ; START: IN R0,00H 从IN单元读入计数初值

$P 01 00

$P 02 61 ; LDI R1,0FH 立即数0FH送R1

$P 03 0F

$P 04 14 ; AND R0,R1 得到R0低四位

$P 05 61 ; LDI R1,00H 装入和初值00H

$P 06 00

$P 07 F0 ; BZC RESULT 计数值为0则跳转

$P 08 16

$P 09 62 ; LDI R2,60H 读入数据始地址

$P 0A 60

$P 0B CB ; LOOP: LAD R3,[RI],00H 从MEM读入数据送R3，

变址寻址，偏移量为00H

$P 0C 00

$P 0D 0D ; ADD R1,R3 累加求和

$P 0E 72 ; INC RI 变址寄存加1，指向下一数据

$P 0F 63 ; LDI R3,01H 装入比较值

$P 10 01

$P 11 8C ; SUB R0,R3

$P 12 F0 ; BZC RESULT 相减为0，表示求和完毕

$P 13 16

$P 14 E0 ; JMP LOOP 未完则继续

$P 15 0B

$P 16 D1 ; RESULT: STA 70H,R1 和存于MEM的70H单元

$P 17 70

$P 18 34 ; OUT 40H,R1 和在OUT单元显示

$P 19 40

$P 1A E0 ; JMP START 跳转至START

$P 1B 00

$P 1C 50 ; HLT 停机

$P 60 01 ; 数据

$P 61 02

$P 62 03

$P 63 04

$P 64 05

$P 65 06

$P 66 07

$P 67 08

$P 68 09

$P 69 0A

$P 6A 0B

$P 6B 0C

$P 6C 0D

$P 6D 0E

$P 6E 0F

; //\*\*\*\*\* End Of Main Memory Data \*\*\*\*\*//

; //\*\* Start Of MicroController Data \*\*//

$M 00 000001 ; NOP

$M 01 006D43 ; PC->AR, PC加1

$M 03 107070 ; MEM->IR, P<1>

$M 04 002405 ; RS->B

$M 05 04B201 ; A加B->RD

$M 06 002407 ; RS->B

$M 07 013201 ; A与B->RD

$M 08 106009 ; MEM->AR

$M 09 183001 ; IO->RD

$M 0A 106010 ; MEM->AR

$M 0B 000001 ; NOP

$M 0C 103001 ; MEM->RD

$M 0D 200601 ; RD->MEM

$M 0E 005341 ; A->PC

$M 0F 0000CB ; NOP, P<3>

$M 10 280401 ; RS->IO

$M 11 103001 ; MEM->RD

$M 12 06B201 ; A加1->RD

$M 13 002414 ; RS->B

$M 14 05B201 ; A减B->RD

$M 15 002416 ; RS->B

$M 16 01B201 ; A或B->RD

$M 17 002418 ; RS->B

$M 18 02B201 ; A右环移->RD

$M 1B 005341 ; A->PC

$M 1C 10101D ; MEM->A

$M 1D 10608C ; MEM->AR, P<2>

$M 1E 10601F ; MEM->AR

$M 1F 101020 ; MEM->A

$M 20 10608C ; MEM->AR, P<2>

$M 28 101029 ; MEM->A

$M 29 00282A ; RI->B

$M 2A 04E22B ; A加B->AR

$M 2B 04928C ; A加B->A, P<2>

$M 2C 10102D ; MEM->A

$M 2D 002C2E ; PC->B

$M 2E 04E22F ; A加B->AR

$M 2F 04928C ; A加B->A, P<2>

$M 30 001604 ; RD->A

$M 31 001606 ; RD->A

$M 32 006D48 ; PC->AR, PC加1

$M 33 006D4A ; PC->AR, PC加1

$M 34 003401 ; RS->RD

$M 35 000035 ; NOP

$M 36 006D51 ; PC->AR, PC加1

$M 37 001612 ; RD->A

$M 38 001613 ; RD->A

$M 39 001615 ; RD->A

$M 3A 001617 ; RD->A

$M 3B 000001 ; NOP

$M 3C 006D5C ; PC->AR, PC加1

$M 3D 006D5E ; PC->AR, PC加1

$M 3E 006D68 ; PC->AR, PC加1

$M 3F 006D6C ; PC->AR, PC加1

; //\*\* End Of MicroController Data \*\*//

选择联机软件的“【转储】—【装载】”功能，在打开文件对话框中选择上面所保存的文件，软件自动将机器程序和微程序写入指定单元。

选择联机软件的“【转储】—【刷新指令区】”可以读出下位机所有的机器指令和微指令，并在指令区显示，对照文件检查微程序和机器程序是否正确，如果不正确，则说明写入操作失败，应重新写入，可以通过联机软件单独修改某个单元的指令，以修改微指令为例，先用鼠标左键单击指令区的‘微存’TAB按钮，然后再单击需修改单元的数据，此时该单元变为编辑框，输入6位数据并回车，编辑框消失，并以红色显示写入的数据。

3. 运行程序

方法一：本机运行

将时序与操作台单元的开关KK1、KK3置为‘运行’档，按动CON单元的总清按钮CLR，将使程序计数器PC、地址寄存器AR和微程序地址为00H，程序可以从头开始运行，暂存器A、B，指令寄存器IR和OUT单元也会被清零。

将时序与操作台单元的开关KK2置为‘单步’档，每按动一次ST按钮，即可单步运行一条微指令，对照微程序流程图，观察微地址显示灯是否和流程一致。每运行完一条微指令，观测一次数据总线和地址总线，对照数据通路图，分析总线上的数据是否正确。

当模型机执行完OUT指令后，检查OUT单元显示的数是否正确，按下CON单元的总清按钮CLR，改变IN单元的值，再次执行机器程序，从OUT单元显示的数判别程序执行是否正确。

方法二：联机运行（软件使用说明请看附录1）

进入软件界面，选择菜单命令“【实验】—【复杂模型机】”，打复杂模型机实验数据通路图，选择相应的功能命令，即可联机运行、监控、调试程序。

按动CON单元的总清按钮CLR，然后通过软件运行程序，当模型机执行完OUT指令后，检查OUT单元显示的数是否正确。在数据通路图和微程序流中观测指令的执行过程，并观测软件中地址总线、数据总线以及微指令显示和下位机是否一致。

# 附录1 软件使用说明

**软件运行环境**

操作系统：Windows 98/NT/2000/XP

最低配置：

CPU：奔腾300MHz

内存：64MB

显示卡：标准VGA，256色显示模式以上

硬盘：20MB以上

光驱：标准CD-ROM

**安装软件**

安装操作如下：

可以通过“资源管理器”，找到光盘驱动器本软件安装目录下的‘安装CMA.EXE’，双击执行它，按屏幕提示进行安装操作。

“TD-CMA”软件安装成功后，在“开始”的“程序”里将出现“CMA”程序组，点击“CMA”即可执行程序。

**卸载软件**

联机软件提供了自卸载功能，使您可以方便地删除“TD-CMA”的所有文件、程序组或快捷方式。单击【开始】/【程序】打开“CMA”的程序组，然后运行“卸载”项，就可执行卸载功能，按照屏幕提示操作即可以安全、快速地删除“TD-CMA”。

**（一） 界面窗口介绍**

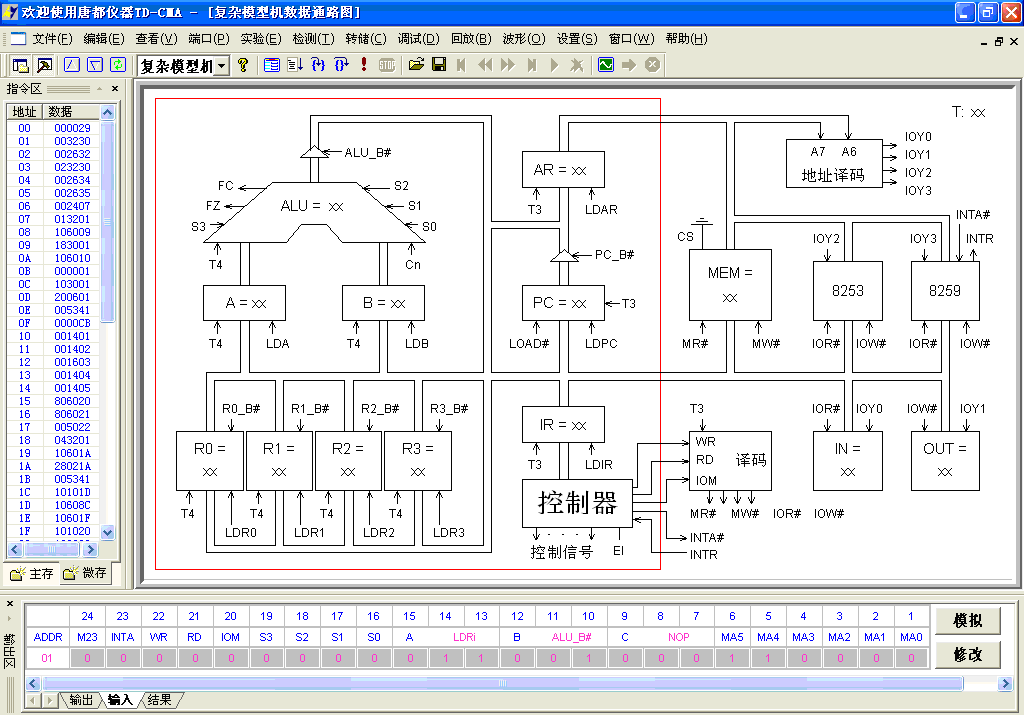
主界面如附图1-1所示，由指令区、输出区和图形区三部分组成。

**指令区：**

分为机器指令区和微指令区，指令区下方有两个Tab按钮，可通过按钮在两者之间切换。

机器指令区：分为两列，第一列为主存地址（00—FF，共256个单元），第二列为每个地址所对应的数值。串口通讯正常且串口无其它操作，可以直接修改指定单元的内容，用鼠标单击要修改单元的数据，此时单元格会变成一个编辑框，即可输入数据，编辑框只接收两位合法的16进制数，按回车键确认，或用鼠标点击别的区域，即可完成修改工作。按下ESC键可取消修改，编辑框会自动消失,恢复显示原来的值，也可以通过上下方向键移动编辑框。

微指令区：分为两列，第一列为微控器地址（00—3F，共64个单元），第二列为每个地址所对应的微指令，共6字节。修改微指令操作和修改机器指令一样，只不过微指令是6位，而机器指令是2位。



附图1-1 软件主界面

**输出区：**

输出区由输出页、输入页和结果页组成。

输出页：在数据通路图打开，且该通路中用到微程序控制器，运行程序时，输出区用来实时显示当前正在执行的微指令和下条将要执行的微指令的24位微码及其微地址。当前正在执行微指令的显示可通过菜单命令“【设置】—【当前微指令】”进行开关。

输入页：可以对微指令进行按位输入及模拟，鼠标左键单击ADDR值，此时单元格会变成一个编辑框，即可输入微地址，输入完毕后回车，编辑框消失，后面的24位代表当前地址的24位微码，微码值用红色显示，鼠标左键单击微码值可使该值在0和1之间切换。在数据通路图打开时，按动‘模拟’按钮，可以在数据通路中模拟该微指令的功能，按动‘修改’按钮则可以将当前显示的微码值下载到下位机。

结果页：用来显示一些提示信息或错误信息，保存和装载程序时会在这一区域显示一些提示信息。在系统检测时，也会在这一区域显示检测状态和检测结果。

**图形区：**

可以在此区域编辑指令，显示各个实验的数据通路图、示波器界面等。

**（二）菜单功能介绍**

1. **文件菜单项：**

文件菜单提供了以下命令：

①．新建（N）：

在CMA中建立一个新文档。在文件新建对话框中选择您所要建立的新文件的类型。

②．打开（O）

在一个新的窗口中打开一个现存的文档。您可同时打开多个文档。您可用窗口菜单在多个打开的文档中切换。

③．关闭（C）

关闭包含活动文档的所有窗口。CMA会建议您在关闭文档之前保存对您的文档所做的改动。如果您没有保存而关闭了一个文档，您将会失去自从您最后一次保存以来所做的所有改动。在关闭一无标题的文档之前，CMA会显示另存为对话框，建议您命名和保存文档。

④．保存（S）

将活动文档保存到它的当前的文件名和目录下。当您第一次保存文档时，CMA显示另存为对话框以便您命名您的文档。如果在保存之前，您想改变当前文档的文件名和目录，您可选用另存为命令。

⑤．另存为（A）…

保存并命名活动文档。CMA会显示另存为对话框以便您命名您的文档。

⑥．打印（P）…

打印一个文档。在此命令提供的打印对话框中，您可以指明要打印的页数范围、副本数、目标打印机，以及其它打印机设置选项。

⑦．打印预览（V）

按要打印的格式显示活动文档。当您选择此命令时，主窗口就会被一个打印预览窗口所取代。这个窗口可以按它们被打印时的格式显示一页或两页。打印预览工具栏提供选项使您可选择一次查看一页或两页，在文档中前后移动，放大和缩小页面，以及开始一个打印作业。

⑧．打印设置（R）…

选择一台打印机和一个打印机连接。在此命令提供的打印设置对话框中，您可以指定打印机及其连接。

⑨．最近使用文件

您可以通过此列表，直接打开最近打开过的文件，共四个。

⑩．退出（X）

结束CMA的运行阶段。您也可使用在应用

程序控制菜单上的关闭命令。

**2．编辑菜单项：**

①．撤消（U）

撤消上一步编辑操作。

②．剪切（T）

将当前被选取的数据从文档中删除并放置于剪贴板上。如当前没有数据被选取时，此命令则不可用。

③．复制（C）

将被选取的数据复制到剪切板上。如当前无数据被选取时，此命令则不可用。

④．粘贴（P）

将剪贴板上内容的一个副本插入到插入点处。如剪贴板是空的，此命令则不可用。



**3．查看菜单项：**

查看菜单提供了以下命令：

①．工具栏（T）

显示和隐藏工具栏，工具栏包括了CMA中一些最普通命令的按钮。当工具栏被显示时，在菜单项目的旁边会出现一个打勾记号。

②．指令区（W）

显示和隐藏指令区，当指令区被显示时，在菜单项目的旁边会出现一个打勾记号。

③．输出区（O）

显示和隐藏输出区，当输出区被显示时，在菜单项目的旁边会出现一个打勾记号。

④．状态栏（S）

显示和隐藏状态栏。状态栏描述了被选取的菜单项目或被按下的工具栏按钮，以及键盘的锁定状态将要执行的操作。当状态栏被显示时，在菜单项目的旁边会出现一个打勾记号。

端口菜单项

**4．端口菜单项：**

端口菜单提供了以下命令：

①．端口选择…

选择通讯端口，选择该命令时会弹出附图1-2所示对话框。该命令会自动检测当前系统可用的串口号，并列于组合框中，选择某一串口后，按确定键，对选定串口进行初始化操作，并进行联机测试，报告测试结果，如果联机成功，则会将指令区初始化。



附图1-2 串口选择对话框

②．端口测试（T）

对当前选择的串口进行联机通讯测试，并报告测试结果，只测一次，如果联机成功，则会将指令区初始化。如串口不能正常初始化，此命令则不可用。

**5．实验菜单项：**



实验菜单提供了以下命令：

①．运算器实验

打开运算器实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

②．存储器实验

打开存储器实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

③．微控器实验

打开微控器实验数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

④．简单模型机

打开简单模型机数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑤ 复杂模型机

打开复杂模型机数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑥ RISC模型机

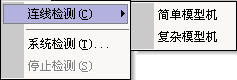
打开RISC模型机数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑦ 重叠模型机

打开重叠模型机数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

⑧ 流水复杂模型机

打开流水模型机数据通路图，如果该通路图已经打开，则把通路激活并置于最前面显示。

**6．检测菜单项：**

检测菜单提供了以下命令：

①．连线检测（C）

1. 简单模型机

对简单模型机的连线进行检测，并在‘输出区’的‘结果页’显示相关信息。

1. 复杂模型机

对复杂模型机的连线进行检测，并在‘输出区’的‘结果页’显示相关信息。

②．系统检测（T）…

启动系统检测，可以进行系统或是整机检测。

③．停止检测（S）

停止系统检测。

**7．转储菜单项：**

转储菜单提供了以下命令：

①．装载数据…

将上位机指定文件中的数据装载到下位机中，您选择该命令会弹出打开文件对话框。

可以打开任意路径下的\*.TXT文件，如果指令文件合法，系统将把这些指令装载到下位机中，装载指令时，系统提供了一定的检错功能，如果指令文件中有错误的指令，将会导致系统退出装载，并提示错误的指令行。

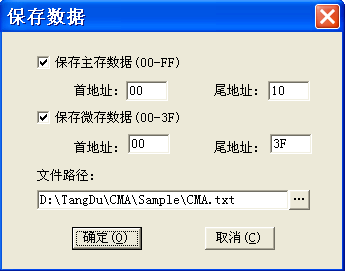
指令文件中指令书写格式如下：



例如机器指令$P00FF,“$”为标记号，“P”代表机器指令，“00”为机器指令的地址，“FF”为该地址中的数据。微指令$M00AA77FF,“$”为标记号，“M”代表微指令，“00”为机器指令的地址，“AA77FF”为该地址中的数据。

②．保存数据…

将下位机中（主存，微控器）的数据保存到上位机中，选择该命令会弹出一个保存对话框，如附图1-3所示。



附图1-3 保存数据对话框

可以选择保存机器指令，此时首尾地址输入框将会变亮，否则首尾地址输入框将会变灰，在允许输入的情况下您可以指定需要保存的首尾地址，微指令也是如此，数据到保存指定路径的\*.TXT格式文件中。

③．刷新指令区

从下位读取所有机器指令和微指令，并在指令区显示。

**8．调试菜单项：**

调试菜单提供了以下命令：

①．微程序流程图…

当微控器实验、简单模型机和综合性实验中任一数据通路图打开时，可用此命令来打开指定的微程序流程图，选择该命令会弹出打开文件对话框。

②．单节拍

向下位机发送单节拍命令，下位机完成一个节拍的工作。

③．单周期

向下位机发送单周期命令，下位机完成一个机器周期的工作。

④．单步机器指令

向下位机发送单步机器指令命令，下位机运行一条机器指令。

⑤．连续运行

向下位机发送连续运行命令，下位机将会进入连续运行状态。

⑥．停止运行

如果下位机处于连续运行状态，此命令可以使得下位机停止运行。

**9．回放菜单项：**

回放菜单提供了以下命令。

①．打开…

打开现存的数据文件。

②．保存…

保存当前的数据到数据文件。

③．首端

跳转到首页。

④．向前

向前翻一页。

⑤．向后

向后翻一页。

⑥．末端

跳转到末页。

⑦．播放

连续向后翻页。

⑧．停止播放

停止连续向后翻页。

**10．波形菜单项：**

波形菜单提供了以下命令：

①．打开（O）

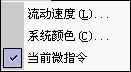
打开示波器窗口。

②．运行（R）

启动示波器，如果下位机正运行程序则不启动。

③．停止（S）

停止处于启动状态的示波器。

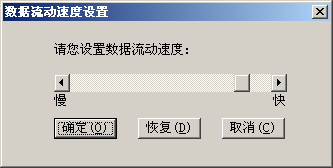


**11．设置菜单项：**

设置菜单提供了以下命令：

①．流动速度（L）…

设置数据通路图中数据的流动速度，选择该命令会弹出一个流动速度设置对话框，如附图1-4所示。拖动滑动块至适当位置，点击‘确定’按钮即完成设置。



附图1-4 流动速度设置对话框

②．系统颜色（C）…

设置数据通路图、微程序流程图和示波器的显示颜色，选择该命令会弹出一个设置对话框，如附图1-5所示。

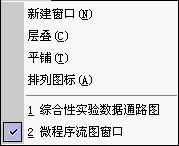


附图1-5 系统颜色设置对话框

分为三页，分别为通路图、微流图和示波器，按动每页的TAB按钮，可在三页之间切换。选择某项要设置的对象，然后按下‘更改’按钮，或直接用鼠标左键点击要设置对象的颜色框，可弹出颜色选择对话框，选定好颜色后，点击‘应用’按钮相应对象的颜色就会被修改掉。

③．当前微指令

设置‘输出区’的‘输出页’是否显示当前微指令，当前微指令用灰色显示，并在地址栏标记为‘C’，下条将要执行的微指令标记为‘N’。

**12．窗口菜单项：**

窗口菜单提供了以下命令。这些命令使您能在应用程序窗口中安排多个文档的多个视图：

①．新建窗口（N）

打开一个具有与活动的窗口相同内容的新窗口。您可同时打开数个文档窗口以显示文档的不同部分或视图。如果您对一个窗口的内容做了改动，所有其它包含同一文档的窗口也会反映出这些改动。当您打开一个新的窗口，这个新窗口就成了活动的窗口并显示于所有其它打开窗口之上。

②．层叠（C）

按相互重叠形式来安排多个打开的窗口。

③．平铺（T）

按互不重叠形式来安排多个打开的窗口。

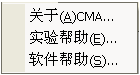
④．排列图标（A）

在主窗口的底部安排被最小化的窗口的图标。如果在主窗口的底部有一个打开的窗口，则有可能会看不见某些或全部图标，因为它们在这个文档窗口的下面。

⑤．窗口选择

CMA在窗口菜单的底部显示出当前打开的文档窗口的清单。有一个打勾记号出现在活动的窗口的文档名前。从该清单中挑选一个文档可使其窗口成为活动窗口。

**13．帮助菜单项：**

帮助菜单提供以下的命令，为您提供使用这个应用程序的帮助：****

①．关于（A）CMA…

显示您的CMA版本的版权通告和版本号码。

②．实验帮助（E）…

显示实验帮助的开场屏幕。从此开场屏幕，您可跳到关于CMA所提供实验的参考资料。

③．软件帮助（S）…

显示软件帮助的开场屏幕。从此开场屏幕，您可跳到关于使用CMA设备的参考资料。

**工具栏命令按钮介绍：**

 显示或隐藏指令区。

 显示或隐藏输出区。

 保存下位机数据。

 向下位机装载数据。

 刷新指令区数据。

 打开实验帮助。

 打开微程序流程图。

 单节拍运行。

 单周期运行。

 单机器指令运行。

 连续运行。

 停止运行。

 打开实验数据文件。

 保存实验数据。

 跳转到首页。

 向前翻页。

 向后翻页。

 跳转到末页。

 连续向后翻页。

 停止向后翻页。

 打开示波器窗口。

 启动示波器。

 停止示波器。