燕山大学

计算机组成原理实验报告

Principles of Computer Organization Experiment Report

学生所在学院：信息科学与工程学院

学生所在班级：计算机科学与技术 **21-3** 班

学生姓名：李梦浩

学生学号：**202111070121**

指导教师：任大伟、张秀杰、许莹

教　务　处

2023 年 **11** 月

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

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。

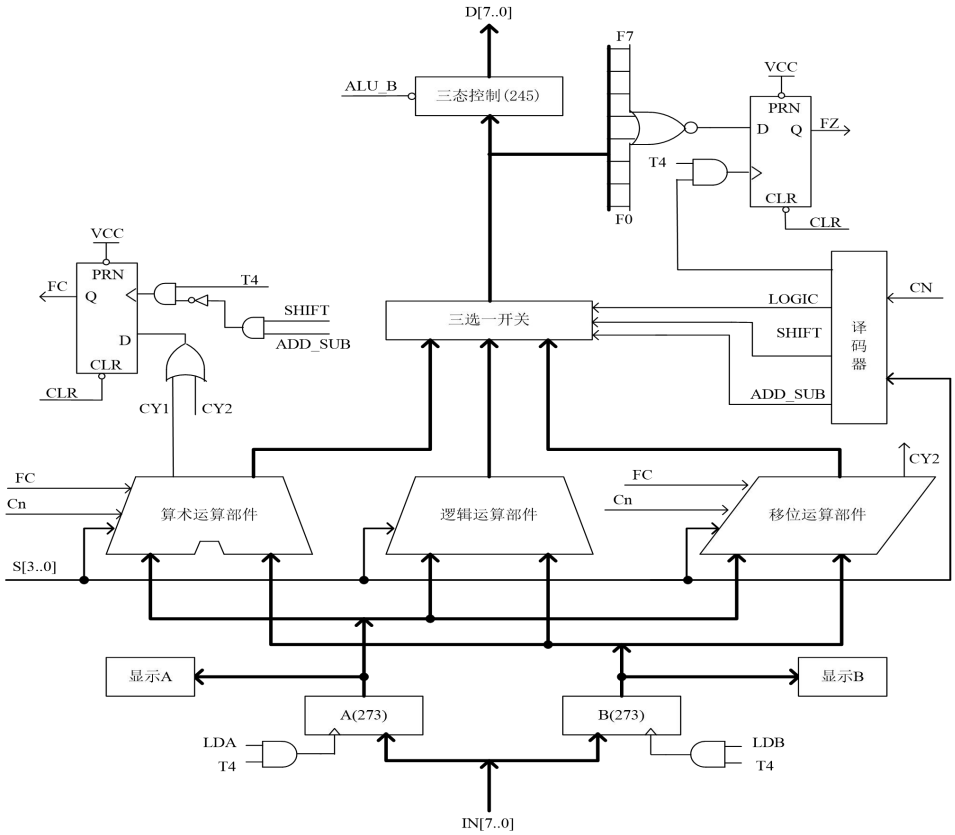


图 1-1　运算器原理图

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。

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）。

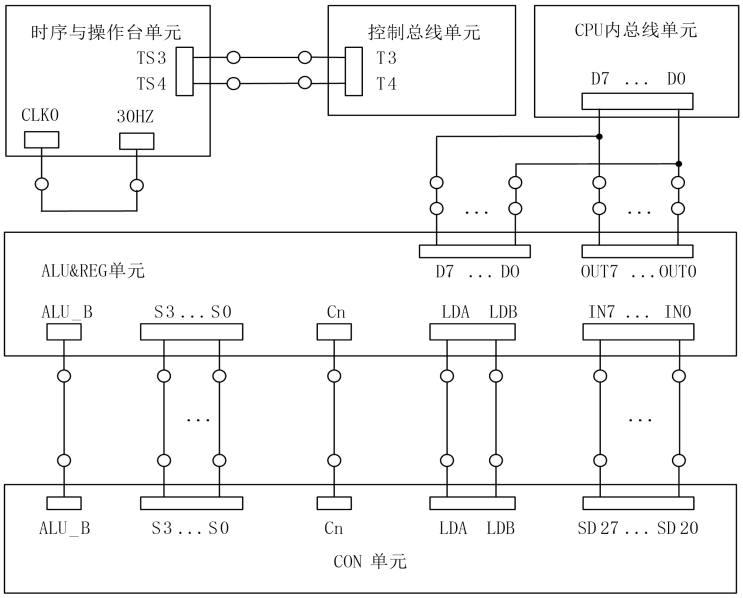


图 1-5 实验接线图

(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。

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。

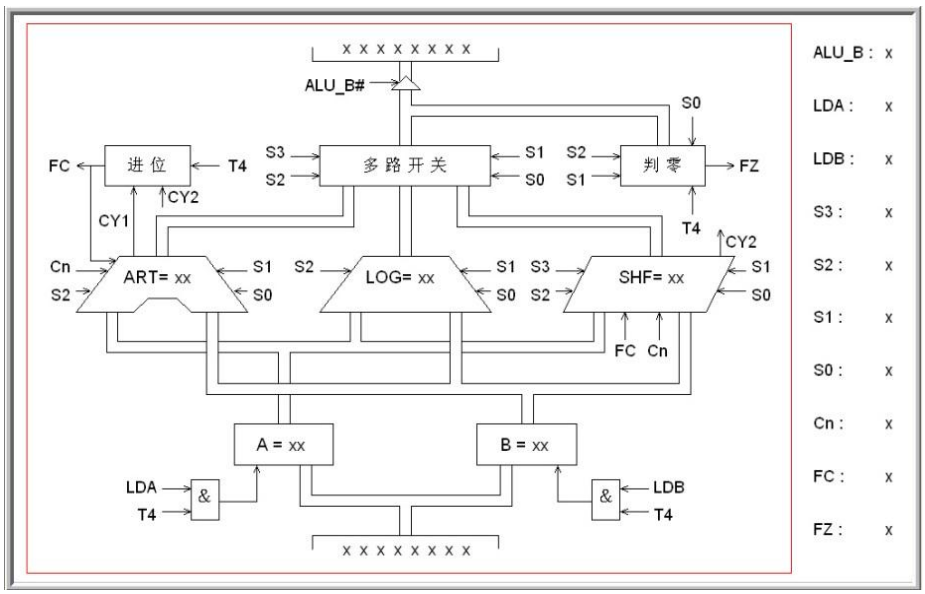


图 1-6　数据通路图

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=(0)　FZ=(0)
	65	A7	0 0 0 1	X	F=(A7)　FC=(0)　FZ=(0)
	65	A7	0 0 1 0	X	F=(25)　FC=(0)　FZ=(0)
	65	A7	0 0 1 1	X	F=(E7)　FC=(0)　FZ=(0)
	65	A7	0 1 0 0	X	F=(9A)　FC=(0)　FZ=(0)
移位运算	65	A7	0 1 0 1	X	F=(CA)　FC=(0)　FZ=(0)
	65	A7	0 1 1 0	0	F=(32)　FC=(0)　FZ=(0)
				1	F=(B2)　FC=(1)　FZ=(0)
	65	A7	0 1 1 1	0	F=(CA)　FC=(1)　FZ=(0)
				1	F=(CA)　FC=(0)　FZ=(0)
算术运算	65	A7	1 0 0 0	X	F=(65)　FC=(0)　FZ=(0)
	65	A7	1 0 0 1	X	F=(0C)　FC=(1)　FZ=(0)
	65	A7	1 0 1 0（FC=0）	X	F=(0D)　FC=(1)　FZ=(0)
			1 0 1 0（FC=1）	X	F=(0D)　FC=(1)　FZ=(0)
	65	A7	1 0 1 1	X	F=(BE)　FC=(1)　FZ=(0)
	65	A7	1 1 0 0	X	F=(64)　FC=(0)　FZ=(0)
	65	A7	1 1 0 1	X	F=(66)　FC=(0)　FZ=(0)

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。

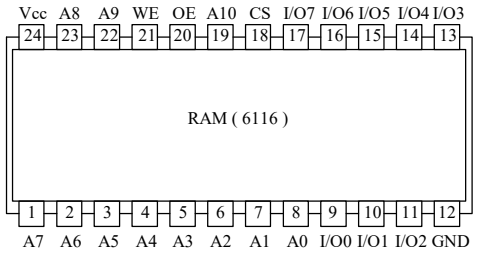


图 2-1　SRAM 6116 引脚图

表 2-1　SRAM 6116 功能表

$\overline{CS}$	$\overline{WE}$	$\overline{OE}$	功能
1	×	×	不选择
0	1	0	读
0	0	1	写
0	0	0	写

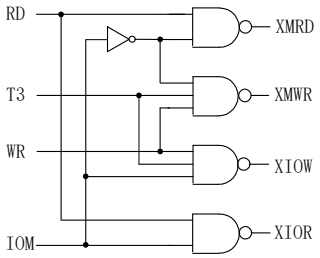


图 2-2　读写控制逻辑

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。

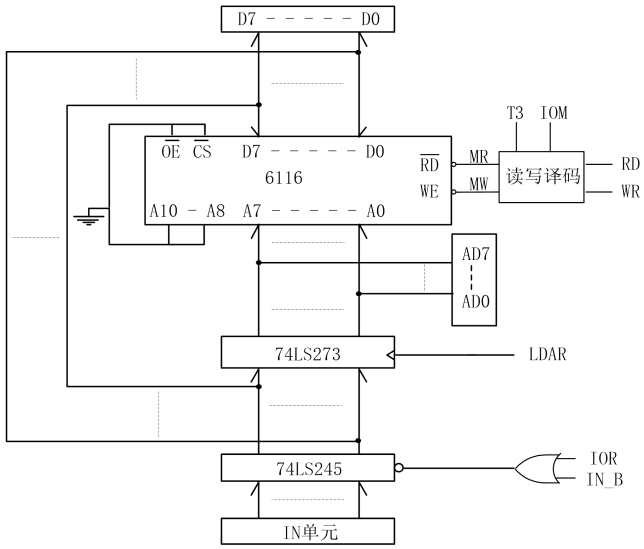


图 2-3　存储器实验原理图

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。

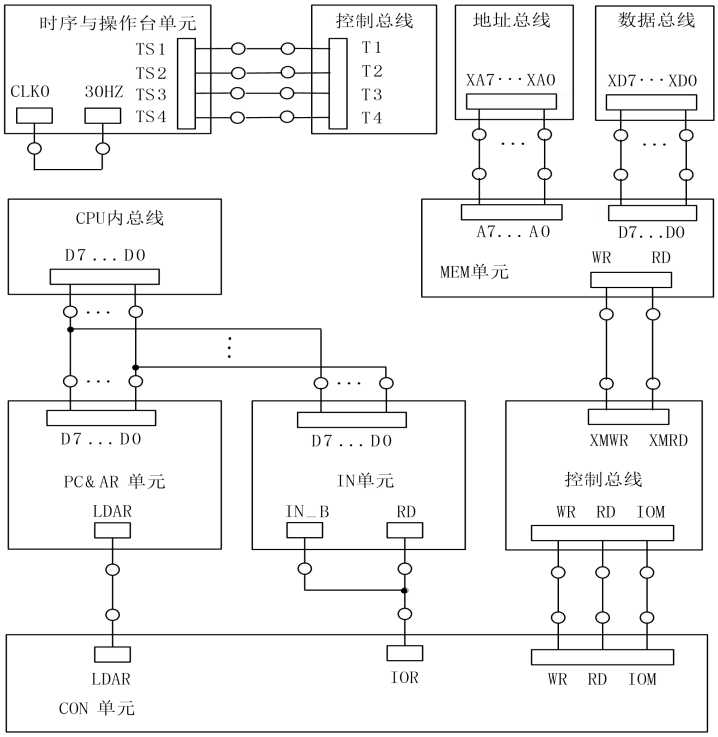


图 2-4　实验接线图

(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）：

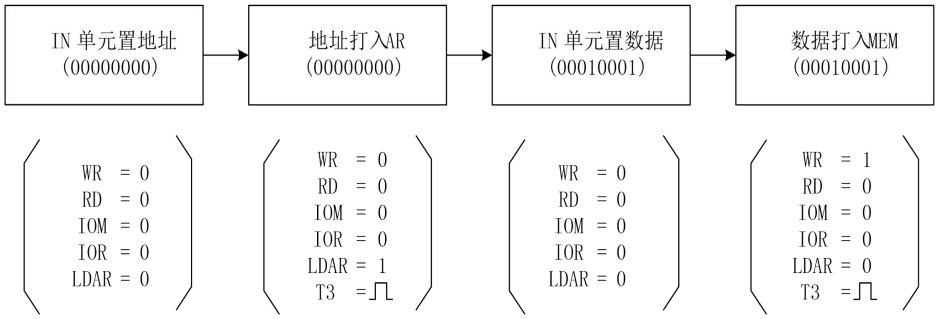


图 2-5　写存储器流程图

(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）：

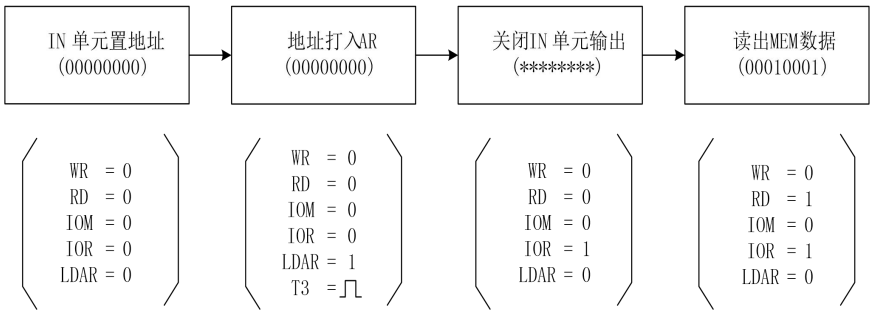


图 2-6　读存储器流程图

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。

五、 实验结果

表 2-2 实验结果

地址单元	写入数据	读出数据
00H	11H	11H
01H	12H	12H
02H	13H	13H
03H	14H	14H
04H	15H	15H

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：

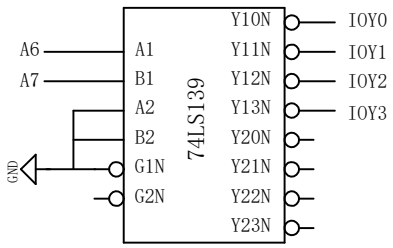


图 3-1　I/O 地址译码原理图

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。

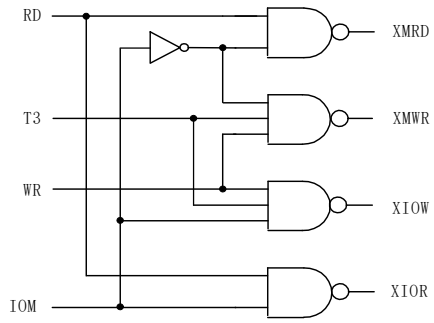


图 3-2　读写控制逻辑

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。

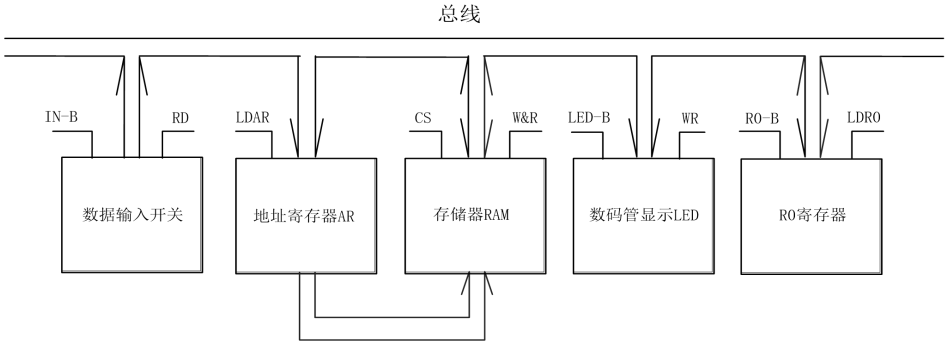


图 3-3　总线传输实验框图

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。

② 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。

③ 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。

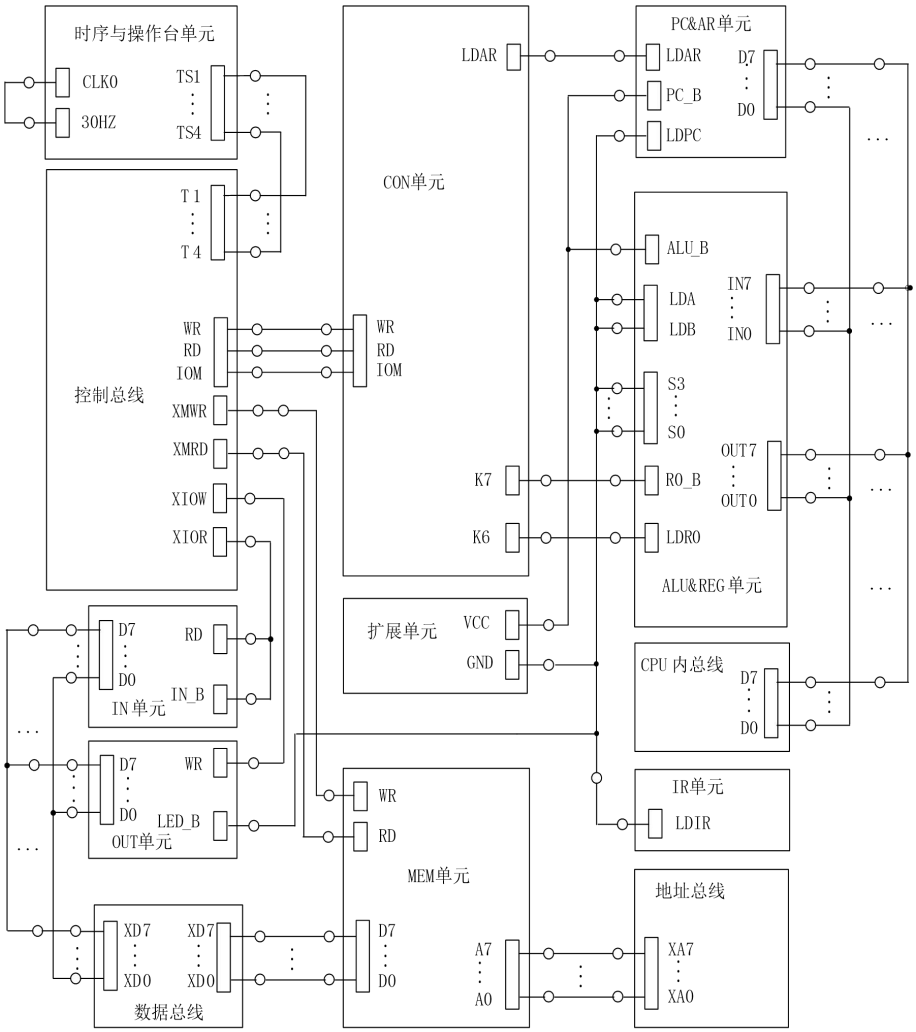


图 3-4　实验接线图

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’。

④ 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。

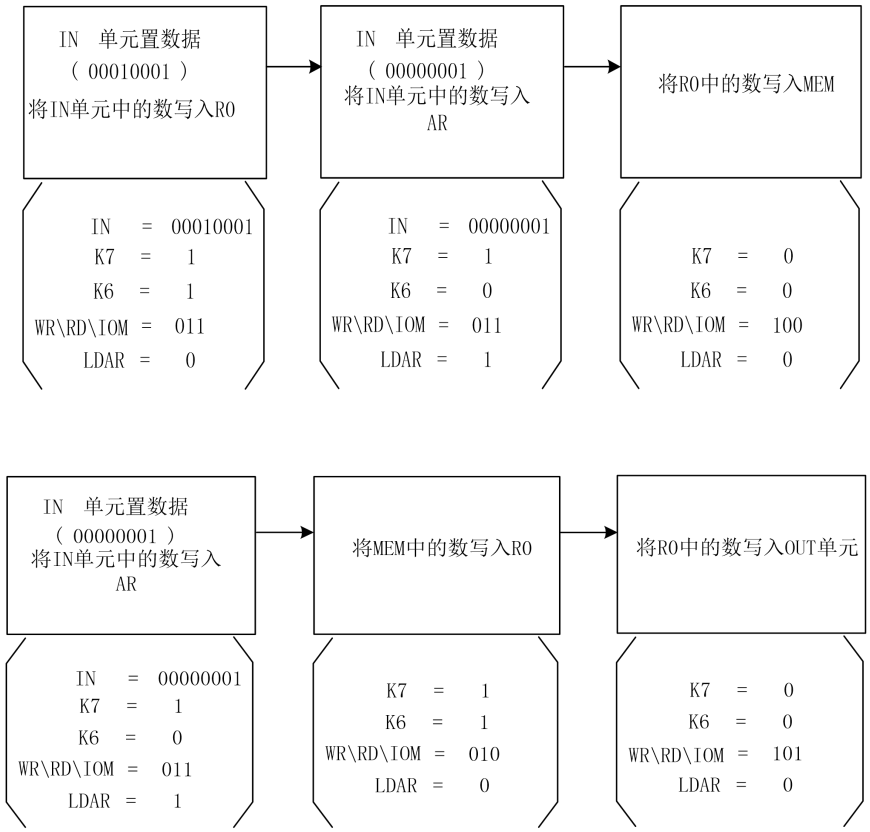


图 3-5 实验流程

六、 实验结果

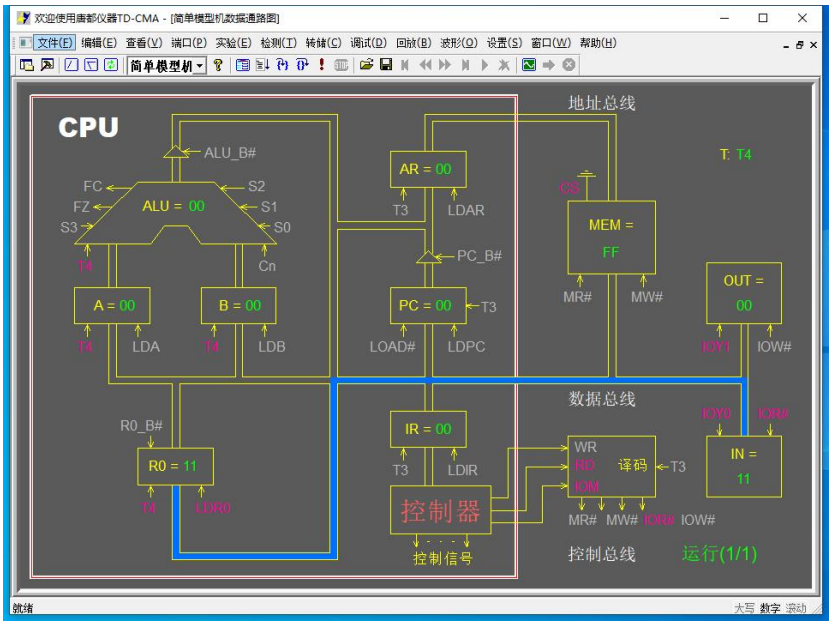


图 3-6 实验流程 3-1

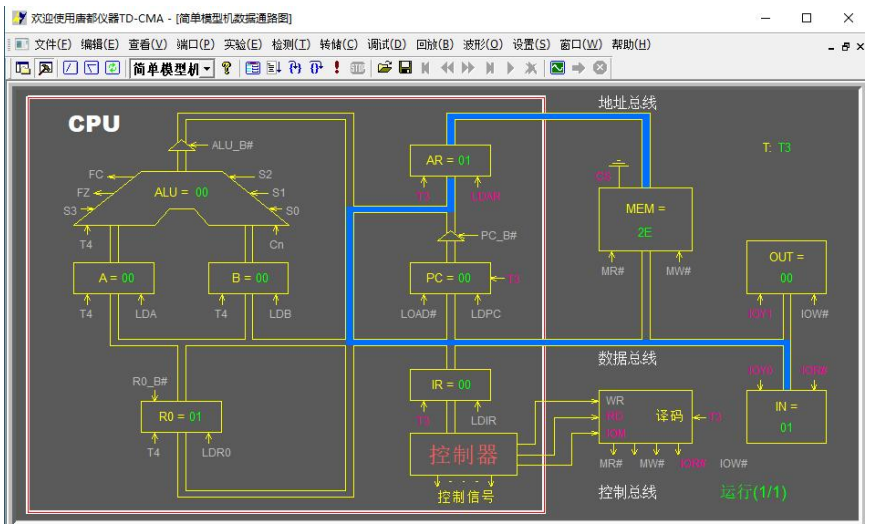


图 3-7 实验流程 3-2

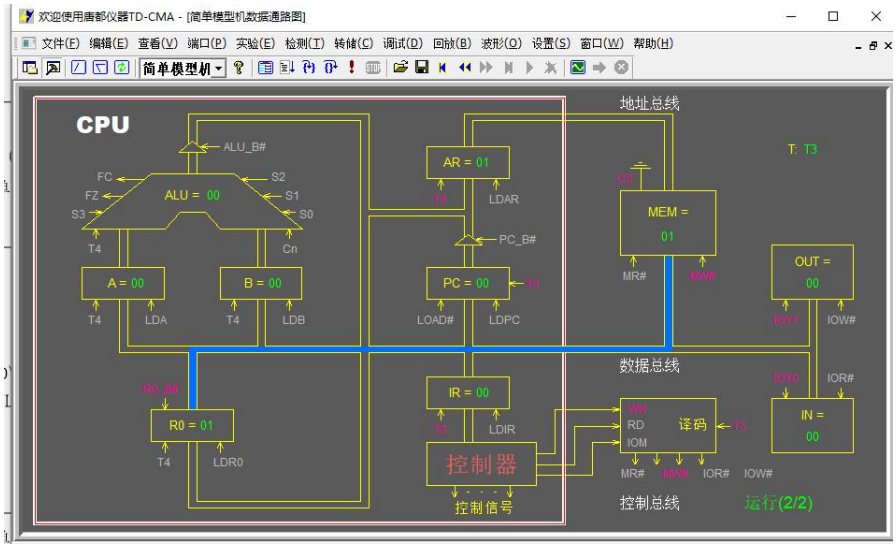


图 3-8 实验流程 3-3

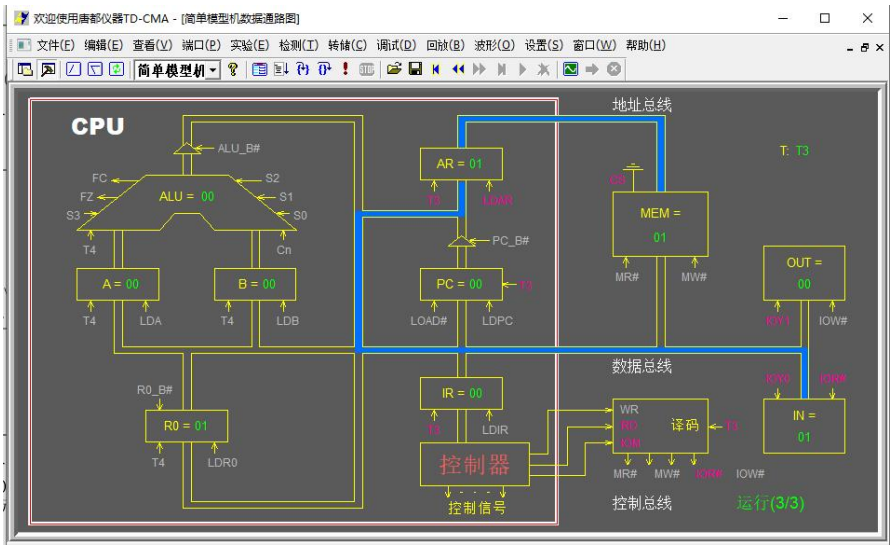


图 3-9 实验流程 3-4

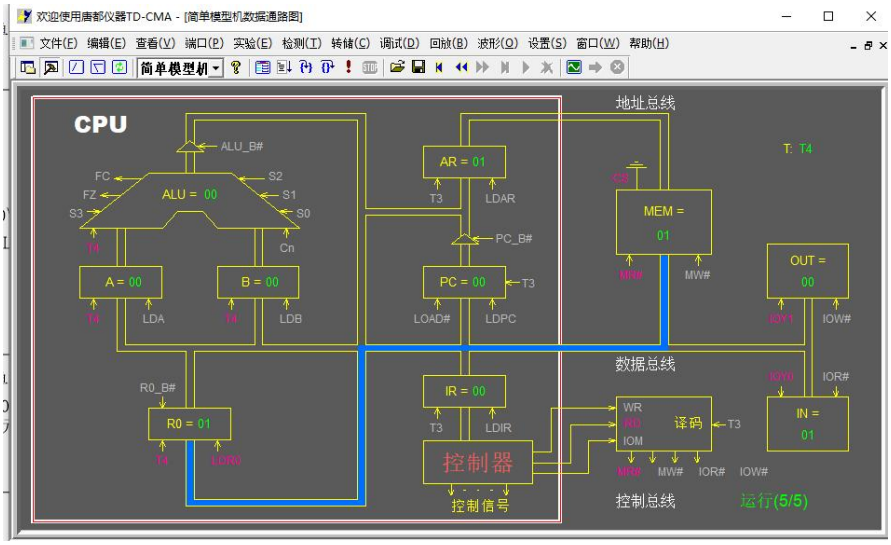


图 3-10 实验流程 3-5

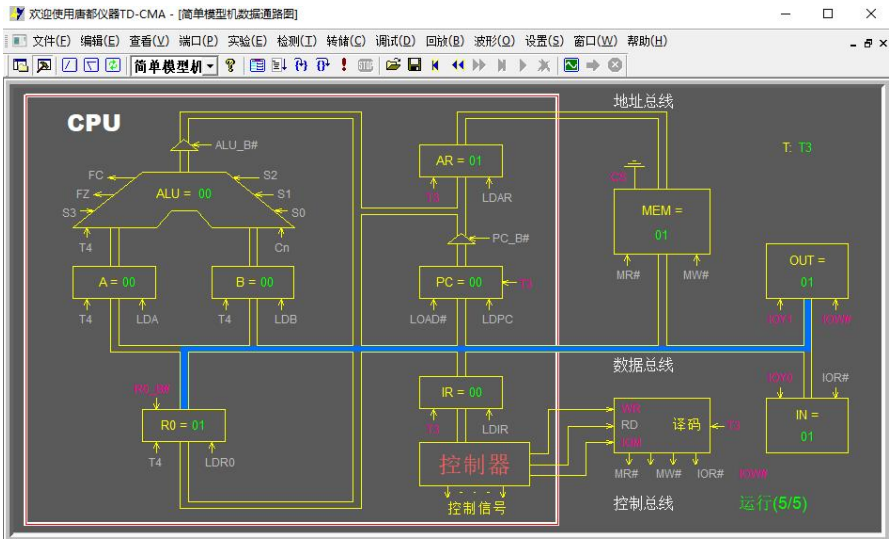


图 3-11 实验流程 3-6

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。

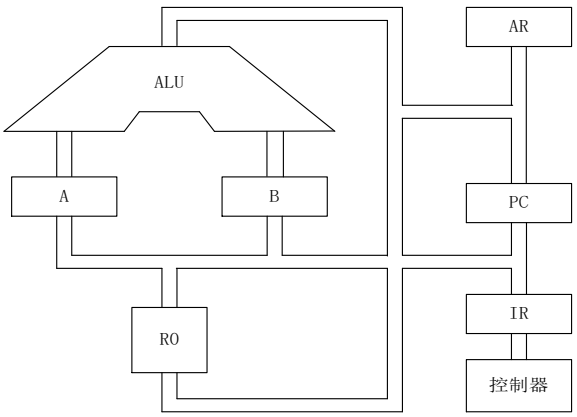


图 5-1　基本 CPU 构成原理图

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。

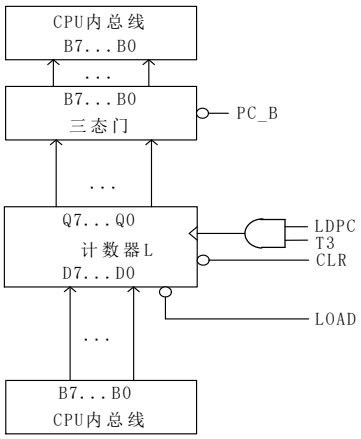


图 5-2　程序计数器(PC)原理图

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）：

助记符	机器指令码		说明
IN	0010 0000		IN → R0
ADD	0000 0000		R0 + R0 → R0
OUT	0011 0000		R0 → OUT
JMP addr	1110 0000	********	addr → PC
HLT	0101 0000		停机

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。

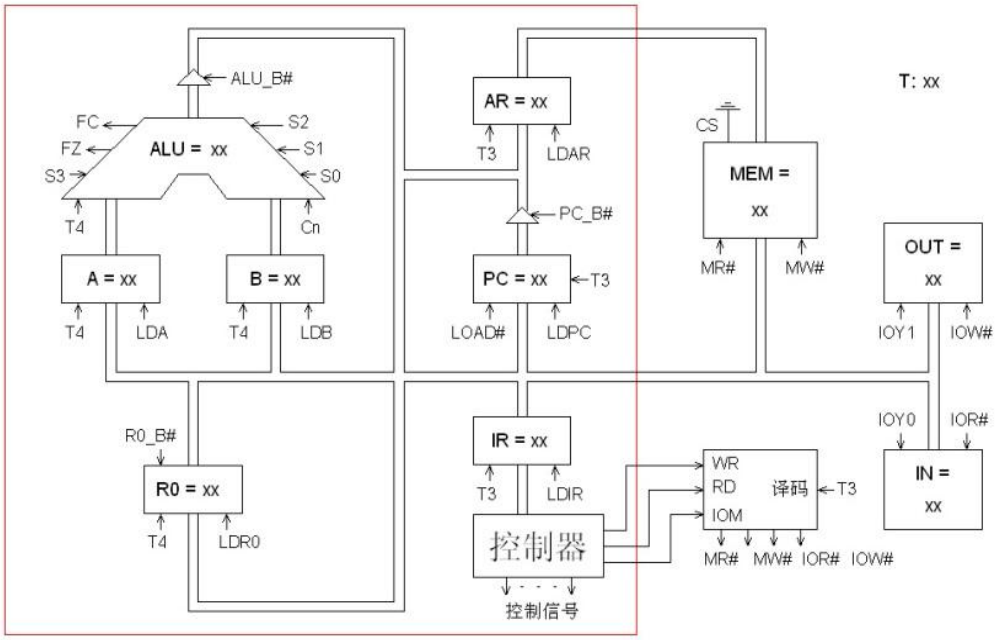


图 5-3　数据通路图

表 5-1 微指令格式

23	22	21	20	19	18-15	14-12	11-9	8-6	5-0
M23	M22	WR	RD	IOM	S3-S0	A字段	B字段	C字段	MA5-MA0

A字段

14	13	12	选择
0	0	0	NOP
0	0	1	LDA
0	1	0	LDB
0	1	1	LDR0
1	0	0	保留
1	0	1	LOAD
1	1	0	LDAR
1	1	1	LDIR

B字段

11	10	9	选择
0	0	0	NOP
0	0	1	ALU_B
0	1	0	R0_B
0	1	1	保留
1	0	0	保留
1	0	1	保留
1	1	0	PC_B
1	1	1	保留

C字段

8	7	6	选择
0	0	0	NOP
0	0	1	P<1>
0	1	0	保留
0	1	1	保留
1	0	0	保留
1	0	1	LDPC
1	1	0	保留
1	1	1	保留

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。

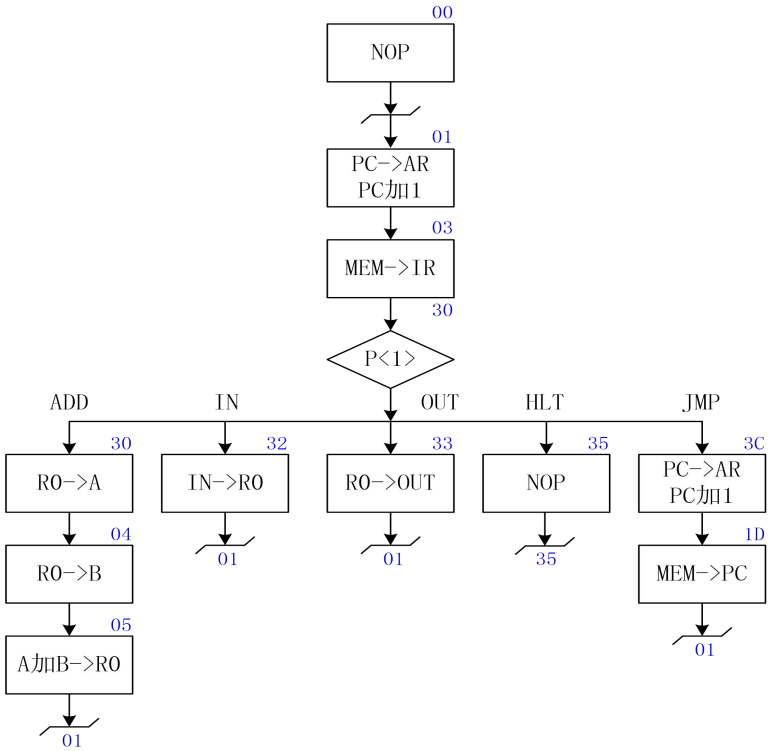


图 5-4　简单模型机微程序流程图

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。

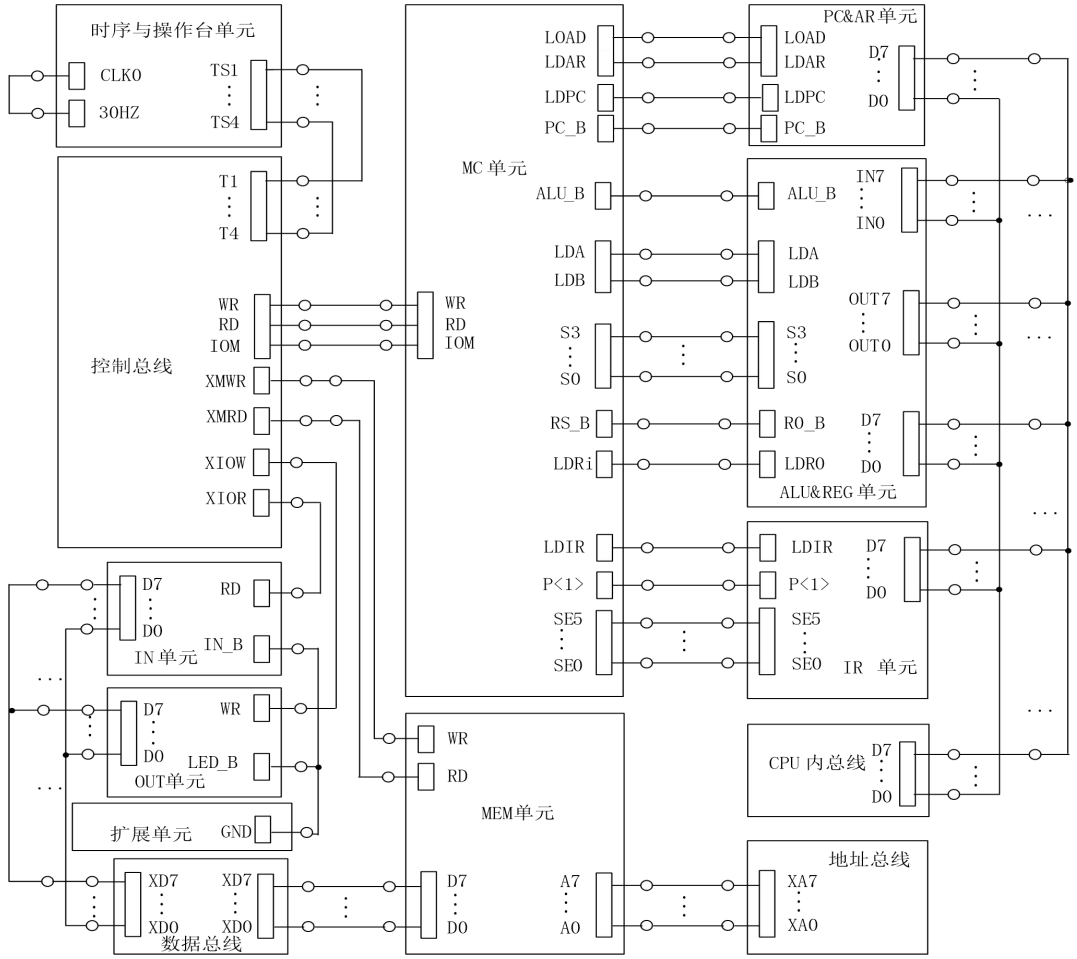


图 5-5　实验接线图

(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。

(2) 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。

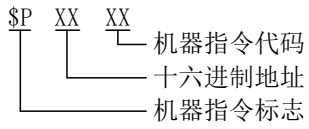
④ 连续两次按动时序与操作台的开关 ST，地址自动加 1，CPU 内总线的指数据指示灯 D7——D0 显示该单元的数据。此后每两次按动时序与操作台的开关 ST，地址自动加 1，CPU 内总线的指数据指示灯 D7——D0 显示该单元的数据，继续进行该操作，直至完成校验，如发现错误，则返回写入，然后校验，直至确认输入的所有指令准确无误。

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。

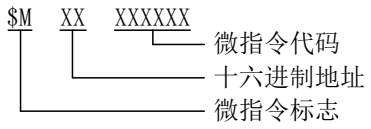
2) 联机写入和校验

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：

机器指令格式说明:



微指令格式说明:



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。

五、 实验结果

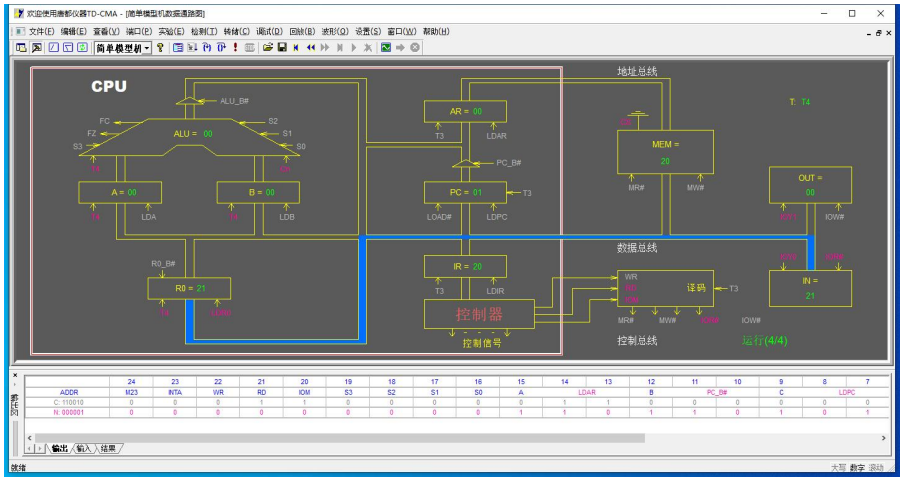


图 5-6 数据输入

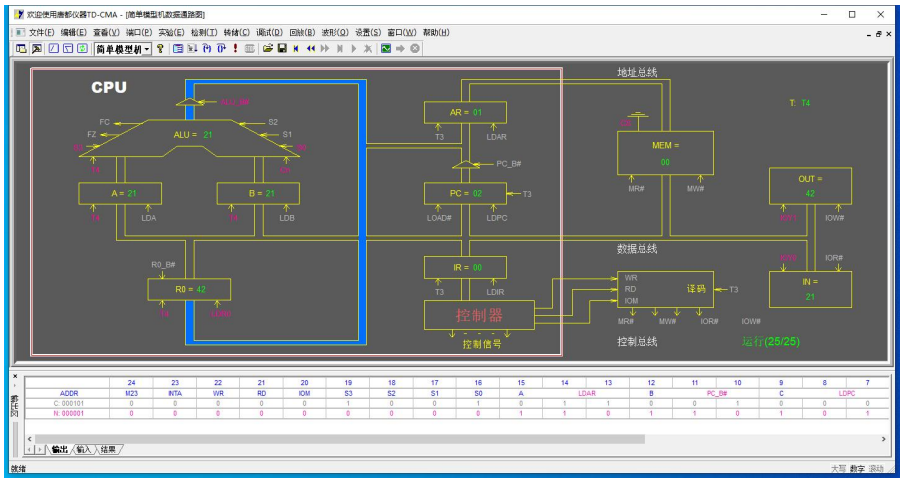


图 5-7 数据相加

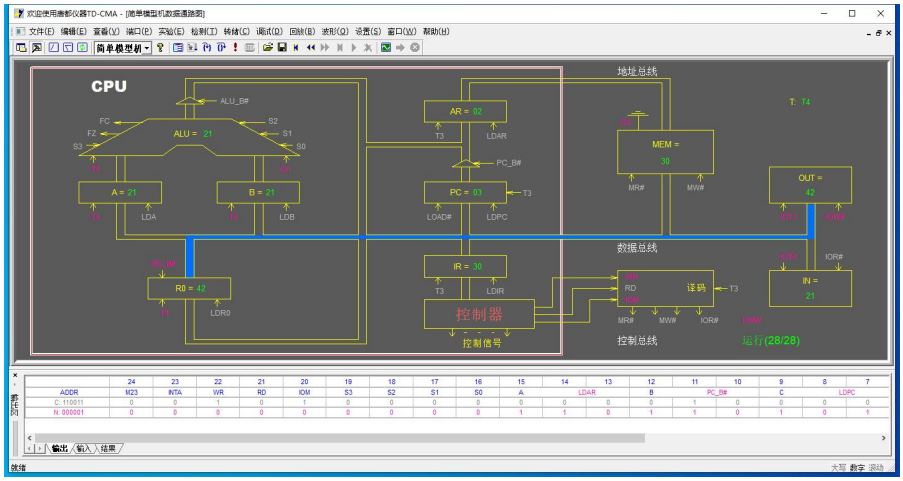


图 5-8 数据输出

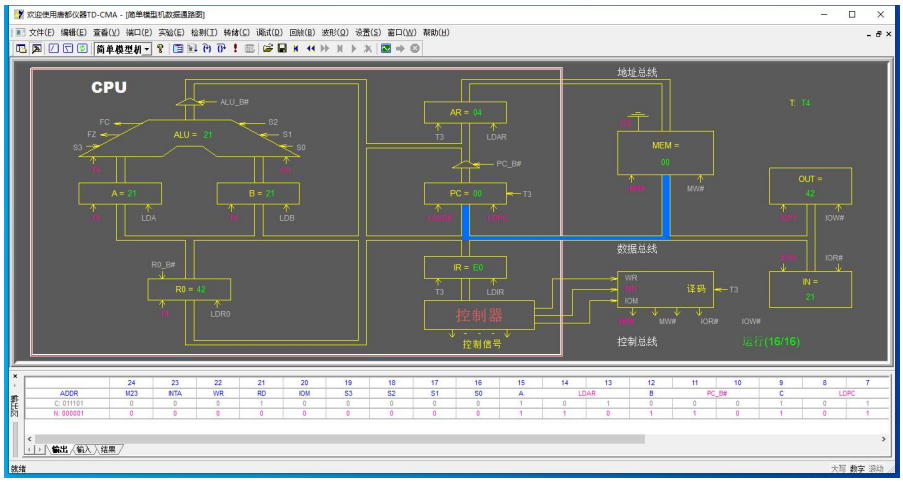


图 5-9 循环实验 5

封面设计： 贾丽

地 址：中国河北省秦皇岛市河北大街 438 号
邮 编：066004
电 话：0335-8057068
传 真：0335-8057068
网 址：http://jwc.ysu.edu.cn