

**学生实验报告**

实验课名称：计算机组成原理

实验项目名称：微程序控制器实验

专业名称：人工智能

班级：2022240401

学号：2022905226

学生姓名：

同组人员姓名：

教师姓名：

2024 **年** 11 **月** 25 **日**

**目录**

[**一、实验名称** 4](#_Toc183513856)

[**二、实验目的** 4](#_Toc183513857)

[**三、实验内容及要求** 4](#_Toc183513858)

[**四、实验原理简述** 5](#_Toc183513859)

[**4.1** **微程序控制器基本原理** 5](#_Toc183513860)

[**4.2** **时序控制** 5](#_Toc183513861)

[**4.3** **硬件组成** 5](#_Toc183513862)

[**4.4** **存储器操作控制** 6](#_Toc183513863)

[**4.4.1**  **控制存储器编辑操作示例** 7](#_Toc183513864)

[**4.4.2 控制存储器校验操作** 8](#_Toc183513865)

[**五、实验设备及工具** 9](#_Toc183513866)

[**六、实验过程详述** 9](#_Toc183513867)

[**6.1** **实验准备** 9](#_Toc183513868)

[**6.2** **微控器读写操作** 11](#_Toc183513869)

[**6.2.1**  **手动读写操作** 12](#_Toc183513870)

[**6.2.1.1** **手动编程（写）操作** 12](#_Toc183513871)

[**6.2.1.2** **手动校验（读）操作** 12](#_Toc183513872)

[**6.2.2**  **联机读写操作** 13](#_Toc183513873)

[**6.2.2.1** **微程序文件编写** 13](#_Toc183513874)

[**6.2.2.2** **写入与校验操作** 14](#_Toc183513875)

[**6.3** **微程序运行** 14](#_Toc183513876)

[**6.3.1**  **本机运行** 14](#_Toc183513877)

[**6.3.2** **联机运行** 15](#_Toc183513878)

[**6.3.2.1** **数据通路观测** 15](#_Toc183513879)

[**6.3.2.2** **微程序流图观测** 15](#_Toc183513880)

[**6.3.2.3** **信号时序观测** 15](#_Toc183513881)

[**七、实验结果与分析** 17](#_Toc183513882)

[**7.1** **微程序执行结果分析** 17](#_Toc183513883)

[**7.1.1**  **指令执行过程记录** 17](#_Toc183513884)

[**7.1.1.1** **指令执行流程图** 17](#_Toc183513885)

[**7.1.1.2** **指令执行详细记录** 18](#_Toc183513886)

[**7.1.2**  **关键时序点分析** 18](#_Toc183513887)

[**7.1.2.1** **时序周期分析** 18](#_Toc183513888)

[**7.1.2.2** **关键信号分析** 19](#_Toc183513889)

[**7.1.3 执行结果验证** 20](#_Toc183513890)

[**7.2** **微程序控制器功能验证** 20](#_Toc183513891)

[**7.2.1 基本功能验证结果** 20](#_Toc183513892)

[**7.2.1.1** **控制存储器功能验证** 20](#_Toc183513893)

[**7.2.1.2** **微指令寄存器功能验证** 21](#_Toc183513894)

[**7.2.1.3** **微地址寄存器功能验证** 21](#_Toc183513895)

[**7.2.2**  **指令执行功能验证** 22](#_Toc183513896)

[**7.2.2.1** **IN指令验证** 22](#_Toc183513897)

[**7.2.2.2** **ADD指令验证** 22](#_Toc183513898)

[**7.2.2.3** **OUT指令验证** 23](#_Toc183513899)

[**八、思考题回答及实验心得** 24](#_Toc183513900)

[**8.1** **思考题解答** 24](#_Toc183513901)

[**8.1.1**  **时序分析** 24](#_Toc183513902)

[**8.1.2**  **详细解释** 26](#_Toc183513903)

[**8.2** **实验心得体会** 26](#_Toc183513904)

[**8.2.1**  **理论知识深入** 26](#_Toc183513905)

[**8.2.2 对实验的改进思考** 27](#_Toc183513906)

[**8.2.3**  **总结与展望** 27](#_Toc183513907)

# **一、实验名称**

微程序控制器实验

# **二、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。
3. 基于数据通路图，掌握微程序控制器的工作原理。
4. 基于微程序流程图，掌握微程序控制器的工作原理。
5. 基于信号时序图，掌握微程序控制器的工作原理。

# **三、实验内容及要求**

1. 实验内容
   * 熟悉微程序控制器的基本组成和工作原理
   * 掌握控制存储器的编程、校验操作方法
   * 编写并运行微程序，观察微程序的执行过程
   * 观察和分析微程序控制器在各个时序下的工作状态
2. 实验要求
   * 理解微程序控制器的数据通路图，掌握各个功能部件的作用
   * 正确操作实验平台，完成控制存储器的编程和校验
   * 观察并记录微程序执行过程中各个时序（T1-T4）的具体工作内容
   * 完成指定指令（IN、ADD、OUT等）的微程序执行，并观察结果
   * 绘制相关的微程序流程图和信号时序图
   * 详细记录实验过程，认真撰写实验报告

# **四、实验原理简述**

## **4.1 微程序控制器基本原理**

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。

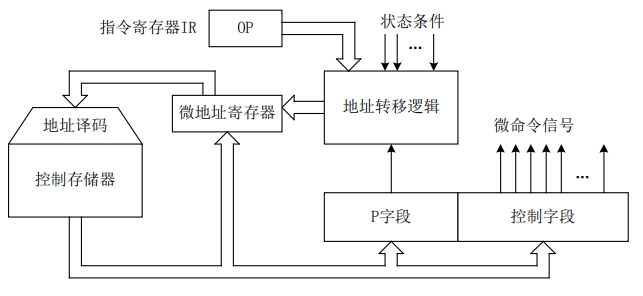
其执行方法是：

1. 将控制各部件动作的微命令的集合进行编码

2. 将微命令的集合仿照机器指令，用数字代码的形式表示（这种表示称为微指令）

3. 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1所示。



**图1：微程序控制器原理框图**

## **4.2 时序控制**

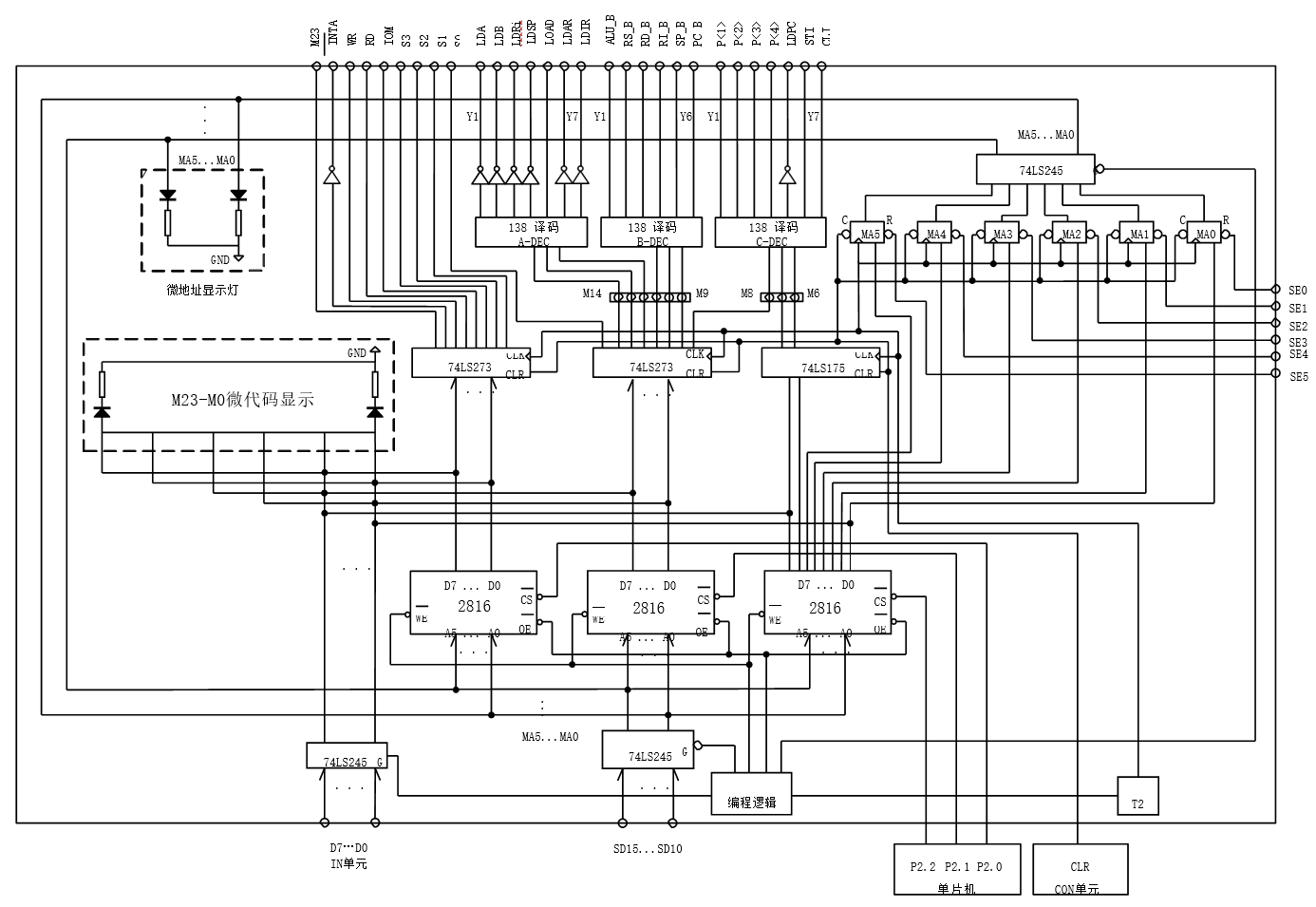
控制器是严格按照系统**时序**来工作的，因而时序控制对于控制器的设计是非常重要的。本实验所用的时序由时序单元来提供，分为四拍：

- TS1 TS2 TS3 TS4

## **4.3 硬件组成**

微程序控制器的组成见图2，其主要组成部分包括：

1. **控制存储器**
   * 采用3片E2PROM
   * 具有掉电保护功能
2. **微命令寄存器**
   * 18位宽度
   * 由两片8D触发器（273）和一片4D（175）触发器组成
3. **微地址寄存器**
   * 6位宽度
   * 由三片正沿触发的双D触发器（74）组成
   * 具有清“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2：微程序控制器组成原理图**

## **4.4 存储器操作控制**

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

**表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 B字段 C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 | 保留 |
| 1 | 1 | 0 | 保留 |
| 1 | 1 | 1 | LDIR |

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | 保留 |
| 1 | 1 | 1 | 保留 |

|  |  |  |  |
| --- | --- | --- | --- |
| 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 | 保留 |
| 1 | 1 | 0 | 保留 |
| 1 | 1 | 1 | 保留 |

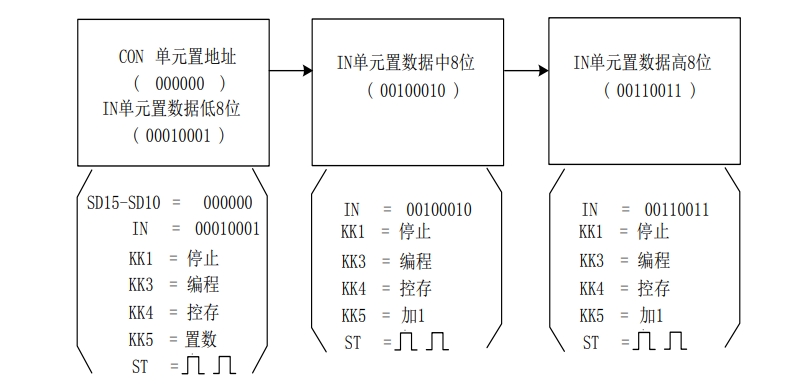
考虑到对于存储器（包括存储器和控制存储器）的操作大多集中在一个地址连续的存储空间中，实验平台提供了便利的手动操作方式。

### **4.4.1 控制存储器编辑操作示例**

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：

1. 初始开关设置：
   * KK1拨至’停止’档
   * KK3拨至’编程’档
   * KK4拨至’控存’档
   * KK5拨至’置数’档
2. 写入低8位数据：
   * CON单元的SD15——SD10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
   * 控制器单元显示：
     + MA5——MA0显示当前地址（000000）
     + M7——M0显示当前数据（00010001）
3. 写入中8位数据：
   * 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
   * 控制器单元显示：
     + MA5——MA0显示当前地址（000000）
     + M15——M8显示当前数据（00100010）
4. 写入高8位数据：
   * 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
   * 控制器单元显示：
     + 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），可继续编辑后续单元。



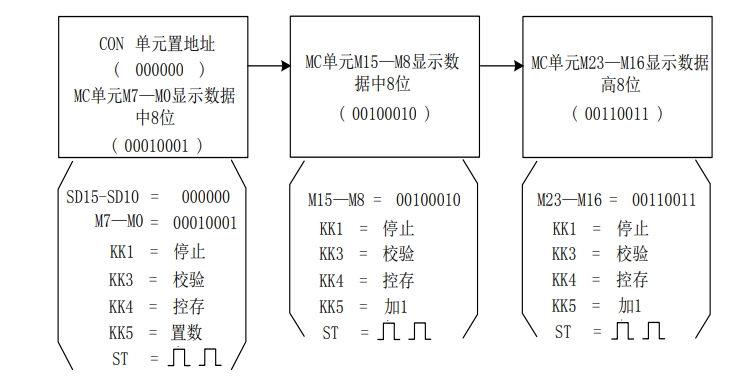
**图3：写入操作**

### **4.4.2 控制存储器校验操作**

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，具体操作步骤如下：

1. 初始开关设置：
   * KK1拨至’停止’档
   * KK3拨至’校验’档
   * KK4拨至’控存’档
   * KK5拨至’置数’档
2. 校验过程：
   * CON单元的SD15——SD10开关给出需要校验的控存单元地址（000000）
   * 连续两次按动ST，观察M7——M0显示（低8位）
   * KK5拨至’加1’档，再连续两次按动ST，观察M15——M8显示（中8位）
   * 再连续两次按动ST，观察M23——M16显示（高8位）
   * 再连续两次按动ST，地址加1，控制器单元指示灯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。



**图4：校验操作**

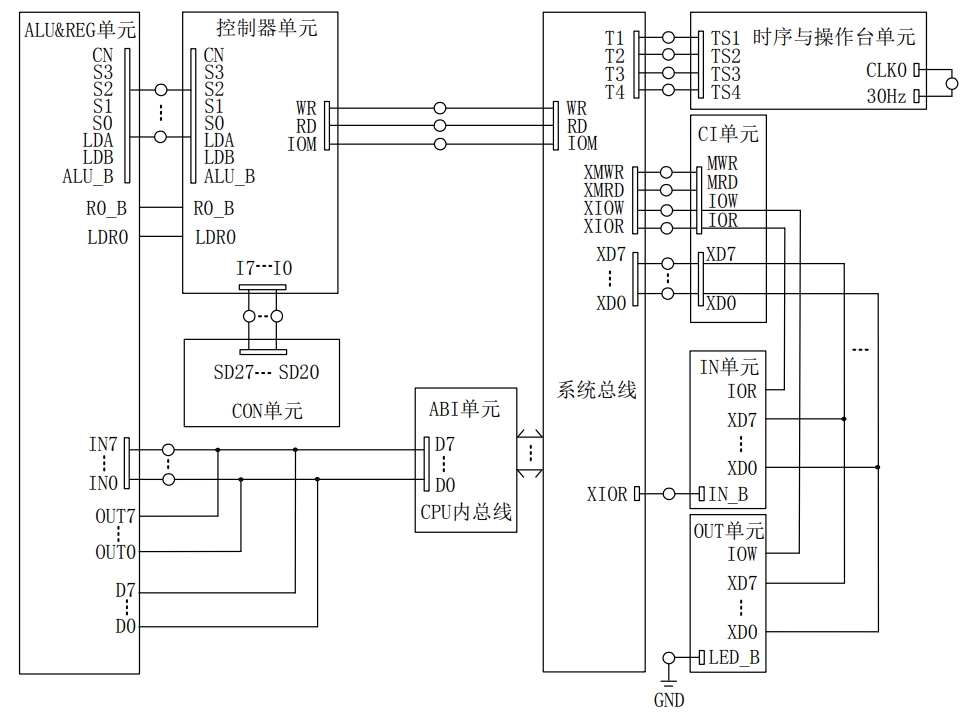
# **五、实验设备及工具**

PC 机一台，TDX-CMX 实验系统一套。

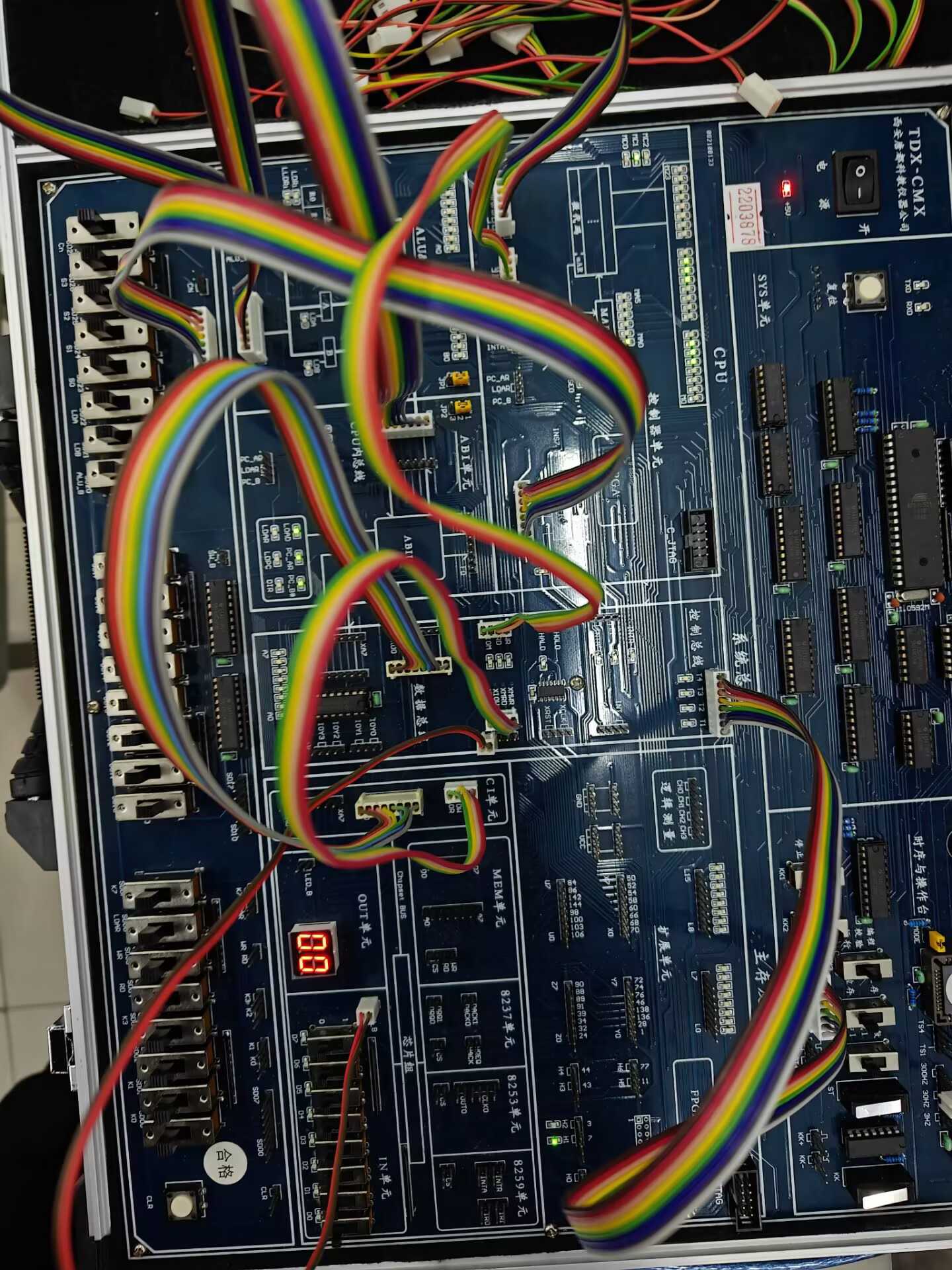
# **六、实验过程详述**

## **6.1 实验准备**

1. 系统设置
   * 将时序与操作台单元的”MODE”用短路块短接，使系统工作在四节拍模式
   * JP1用短路块将1、2短接
   * 按图3-2-10所示连接实验线路
   * 仔细查线无误后接通电源
   * 注意：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5：接线图**



**图6：本机操作和接线**

## **6.2 微控器读写操作**

微控器的读写操作分为两种情况：手动读写和联机读写。

**表 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 |

### **6.2.1 手动读写操作**

#### **6.2.1.1 手动编程（写）操作**

1. 初始开关设置
   * 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. 写入数据步骤
   * 使用CON单元的SD15——SD10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，写入低8位数据
   * 将KK5置为’加1’档
   * IN单元给出中8位数据，连续两次按ST写入
   * IN单元给出高8位数据，连续两次按ST写入
3. 重复以上步骤，将**表2**的微代码写入E2ROM芯片中

#### **6.2.1.2 手动校验（读）操作**

1. 初始开关设置
   * 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. 校验步骤
   * 使用CON单元的SD15——SD10给出微地址
   * 连续两次按动ST，观察M7——M0显示（低8位）
   * 将KK5置为‘加1’档
   * 连续两次按ST，观察M15——M8显示（中8位）
   * 再连续两次按ST，观察M23——M16显示（高8位）
3. 重复以上步骤完成校验，如有错误需重新写入直至正确

**表格

中度可信度描述已自动生成表格

描述已自动生成图7：联机数据写入**

### **6.2.2 联机读写操作**

#### **6.2.2.1 微程序文件编写**

1. 文件格式要求
   * 使用TXT后缀文件
   * 按指定格式编写，如：$M 1F 112233
     + 1F表示微指令地址
     + 11H（高）、22H（中）、33H（低）为微指令值
   * 使用分号(;)添加注释
2. 微程序代码

图片包含 文本

描述已自动生成

**图8：联机指令文件编写**

#### **6.2.2.2 写入与校验操作**

1. 写入微程序
   * 使用联机软件的“【转储】—【装载】”功能
   * 装载过程中可在‘结果’栏查看进度信息
2. 校验微程序
   * 选择“【转储】—【刷新指令区】”读取指令
   * 检查数据是否与表3-2-2相符
   * 如需修改：
     + 点击‘微存’TAB按钮
     + 点击需修改单元
     + 输入6位数据并回车
     + 修改后的数据将以红色显示

## **6.3 微程序运行**

本实验支持两种运行方式：本机运行和联机运行。

### **6.3.1 本机运行**

1. 初始设置
   * KK1、KK3置为’运行’档
   * 按动CON单元的CLR按钮清零（MAR、IR、ALU的A和B寄存器）
   * KK2置为’单拍’档
2. 运行过程观察
   * 按动ST按钮观察T1-T4各节拍工作情况
   * T2节拍：微控器更新微地址和微指令，产生控制信号
   * T3、T4节拍：执行控制信号对应的操作，必要时进行微程序分支
3. 指令执行测试
   * 清零后设置KK2为’单拍’档
   * IN单元置数据00100011
   * 执行IN指令（00100000）
   * 执行ADD指令（00000000）
   * 执行OUT指令（00110000）
   * 观察OUT单元显示值是否为01000110

### **6.3.2 联机运行**

#### **6.3.2.1 数据通路观测**

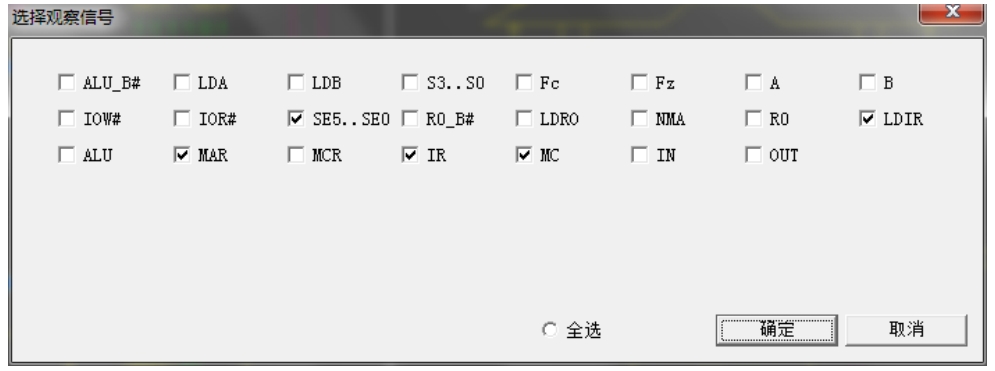
1. 打开方式
   * 选择【实验】－【微控器实验】
   * 或通过工具栏下拉框打开
2. 操作步骤
   * 按本机运行顺序给出数据和指令
   * 观察指令执行过程
   * 验证运算结果

#### **6.3.2.2 微程序流图观测**

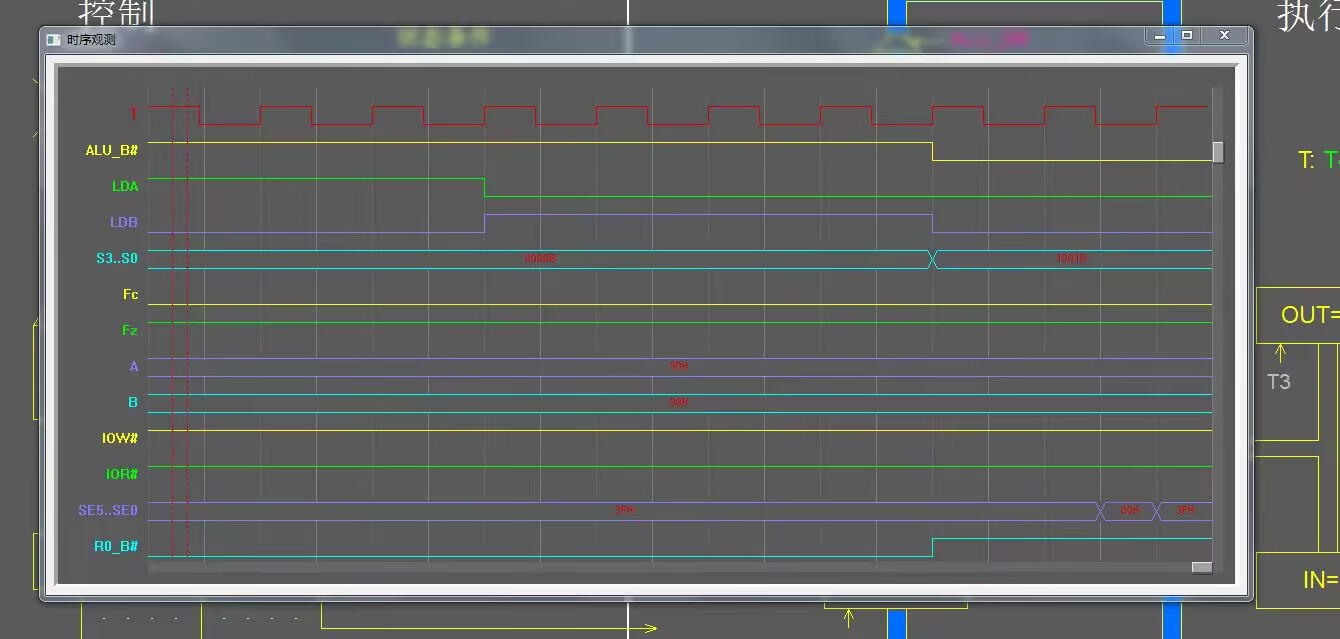
1. 打开方式：点击”【调试】—【微程序流图】”
2. 操作：跟踪每条机器指令的执行过程

#### **6.3.2.3 信号时序观测**

1. 打开方式
   * 点击信号观测窗口图标
   * 或选择”【调试】—【时序观测窗】”
   * 选择需要观察的信号
2. 观测步骤
   * 按动CLR按钮
   * KK2置为’单拍’档
   * 按动4次ST按钮观察第一个周期
   * 继续按动4次ST按钮观察第二个周期
   * 观察各个时序点的数据变化
3. 关键观测点
   * T2时刻：NMA地址传输到MAR
   * T3时刻：机器指令装载
   * T4时刻：指令译码和地址更新
   * 数据传输过程



**图9：选择窗口**



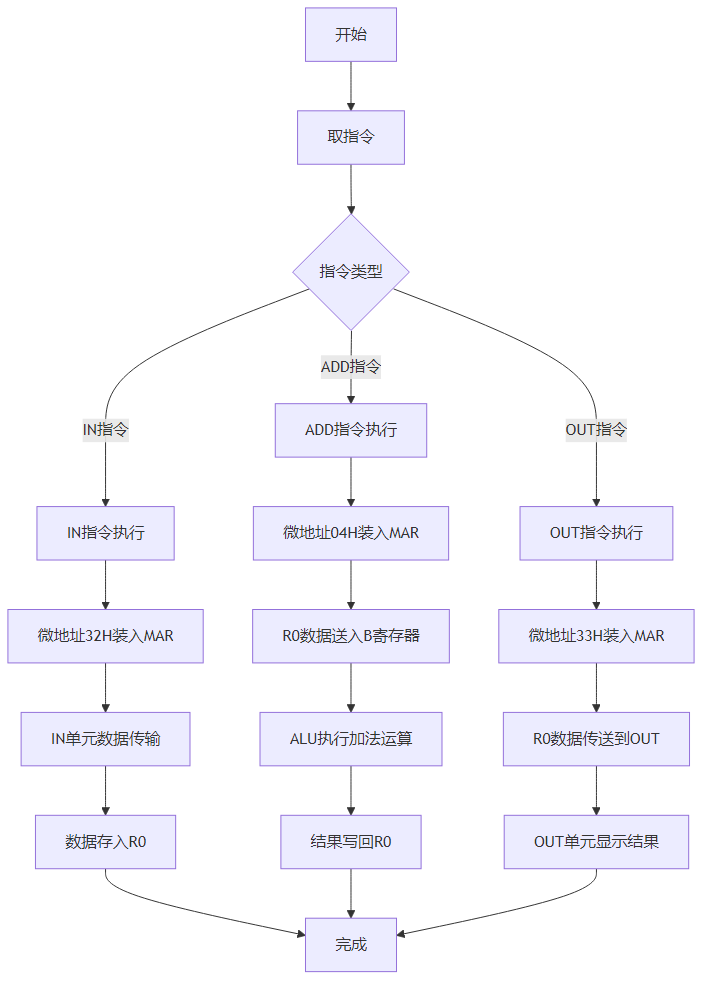
**图10：时序分析**

# **七、实验结果与分析**

## **7.1 微程序执行结果分析**

### **7.1.1 指令执行过程记录**

#### **7.1.1.1 指令执行流程图**



**图11：指令执行流程**

#### **7.1.1.2 指令执行详细记录**

下表记录了三条主要指令（IN、ADD、OUT）的执行过程及结果：

| 指令 | 机器码 | 微地址 | 执行过程 | 观察结果 | 微指令解析 |
| --- | --- | --- | --- | --- | --- |
| IN | 00100000 | 32H | 1. T2时刻：微地址32H装入MAR2. T3时刻：IN单元数据(00100011)传输3. T4时刻：数据存入R0 | R0寄存器值变为00100011 | 183001H：- IOM=1：外设操作- A字段=011：LDR0- 后续地址=01H |
| ADD | 00000000 | 04H | 1. T2时刻：微地址04H装入MAR2. T3时刻：R0数据送入B寄存器3. T4时刻：完成R0+R0运算 | R0寄存器值变为01000110 | 002405H：- S字段=0000：加法- B字段=010：R0\_B- 后续地址=05H |
| OUT | 00110000 | 33H | 1. T2时刻：微地址33H装入MAR2. T3时刻：R0数据传送到OUT3. T4时刻：显示结果 | OUT单元显示01000110 | 280401H：- IOM=1：外设操作- B字段=010：R0\_B- 后续地址=01H |

### **7.1.2 关键时序点分析**

#### **7.1.2.1 时序周期分析**

每个机器周期包含四个时序点（T1-T4），各时序点的具体工作如下：

1. **T1时刻（准备阶段）**
   * 开始新的机器周期
   * 清除上一周期的控制信号
   * 准备接收新的微指令
2. **T2时刻（控制信号生成）**
   * NMA → MAR的地址传送

* IN指令时，32H → MAR
  + 微指令装入微指令寄存器MCR
* 183001H → MCR（IN指令）
  + 控制信号解码和生成
* 微指令字段解析：  
  - 高5位：控制方式  
  - A、B、C字段：功能选择  
  - 低6位：后续地址

1. **T3时刻（数据传输）**
   * 数据总线活动

* 数据流向：  
  IN指令：IN单元 → R0  
  ADD指令：R0 → ALU.B  
  OUT指令：R0 → OUT单元
  + 寄存器操作执行
  + ALU运算（如ADD指令）

1. **T4时刻（完成和转换）**
   * 指令译码完成
   * 微程序分支判断
   * 为下一条指令准备

#### **7.1.2.2 关键信号分析**

1. **控制信号特征**
   * 微指令控制信号

* IOM：输入输出控制  
  WR/RD：读写控制  
  S3-S0：ALU操作选择
  + 时序控制信号
* CLK：基本时钟  
  T1-T4：四相非重叠时钟

1. **数据传输特征**
   * 数据总线状态

* 空闲：高阻态  
  传输：有效数据
  + 寄存器工作状态
* R0：数据暂存  
  MAR：地址保持  
  MCR：微指令保持

1. **地址控制特征**
   * 微地址更新

* 正常：后续地址字段指定  
  分支：由测试条件决定
  + 指令地址变化
* 取指：PC → MAR  
  执行：微地址 → MAR

### **7.1.3 执行结果验证**

1. **数据正确性验证**
   * IN指令：00100011正确写入R0
   * ADD指令：01000110（00100011 + 00100011）
   * OUT指令：01000110正确显示
2. **时序正确性验证**
   * 各时序点工作正常
   * 无信号竞争现象
   * 数据传输稳定可靠
3. **控制正确性验证**
   * 微程序分支正确
   * 指令译码准确
   * 操作序列符合设计

## **7.2 微程序控制器功能验证**

### **7.2.1 基本功能验证结果**

#### **7.2.1.1 控制存储器功能验证**

1. **写入功能验证**

* 测试过程：  
  - 设置地址：SD15-SD10 = 000000  
  - 写入数据：SD7-SD0 = 10110101  
  - 执行写入：按动ST开关  
    
  验证结果：  
  - M7-M0显示 = 10110101  
  - 数据成功写入

1. **读取功能验证**

* 测试过程：  
  - 设置读取地址：000000  
  - 执行读取：ST开关  
    
  验证结果：  
  - 显示与写入数据一致  
  - 读取操作稳定可靠

1. **地址递增功能**

* 测试步骤：  
  1. KK5拨至'加1'档  
  2. 按动ST开关  
  3. 观察地址变化  
    
  验证结果：  
  - 地址正确递增  
  - 顺序访问正常

#### **7.2.1.2 微指令寄存器功能验证**

1. **数据锁存功能**
   * 微指令正确加载到MCR
   * 保持时间满足要求
   * 数据稳定性良好
2. **控制信号输出**

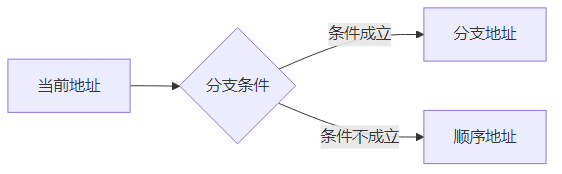
* 验证项目：  
  - 时序控制信号  
  - 数据选择信号  
  - ALU控制信号  
  - 寄存器控制信号  
    
  测试结果：  
  √ 所有控制信号正确生成  
  √ 信号时序符合设计要求  
  √ 无信号干扰和竞争现象

#### **7.2.1.3 微地址寄存器功能验证**

1. **地址保持能力**

* 测试方法：  
  - 加载测试地址  
  - 观察多个时钟周期  
  - 检查地址稳定性  
    
  结果：  
  - 地址保持稳定  
  - 无地址跳变

1. **地址更新机制**



**图12：微地址更新机制**

### **7.2.2 指令执行功能验证**

#### **7.2.2.1 IN指令验证**

1. **输入过程验证**

* 操作步骤：  
  1. 设置IN单元数据  
  2. 执行IN指令  
  3. 检查R0内容  
    
  验证结果：  
  ✓ 数据正确传输  
  ✓ 时序控制准确  
  ✓ R0更新正确

1. **控制信号观察**

| **时序** | **IOM** | **RD** | **WR** | **数据通路** |
| --- | --- | --- | --- | --- |
| T2 | 1 | 1 | 0 | IN→总线 |
| T3 | 1 | 1 | 0 | 总线→R0 |
| T4 | 0 | 0 | 0 | 完成 |

#### **7.2.2.2 ADD指令验证**

1. **运算过程验证**

* 测试数据：  
  - R0初值：00100011  
  - 执行：R0 + R0  
  - 期望结果：01000110  
    
  观察结果：  
  ✓ ALU运算正确  
  ✓ 结果写回正确  
  ✓ 零标志位正确

1. **ALU功能验证**

| **控制信号** | **功能选择** | **运算结果** | **标志位** |
| --- | --- | --- | --- |
| S3-S0=0000 | 加法 | 正确 | 正常 |

#### **7.2.2.3 OUT指令验证**

1. **输出过程验证**

* 测试步骤：  
  1. R0装入测试数据  
  2. 执行OUT指令  
  3. 观察输出单元  
    
  结果确认：  
  ✓ 数据显示正确  
  ✓ 输出稳定可靠

1. **时序特征分析**

* 关键时序点：  
  T2：地址准备  
  T3：数据传输  
  T4：输出锁存

**电脑屏幕的照片上有文字

描述已自动生成图片包含 图示

描述已自动生成图13：微程序运行结果**

# **八、思考题回答及实验心得**

## **8.1 思考题解答**

**问题**：机器指令30H（即OUT指令）执行时，观察寄存器R0中的数据何时送入OUT单元，是组合逻辑还是时序逻辑？受哪些信号影响？

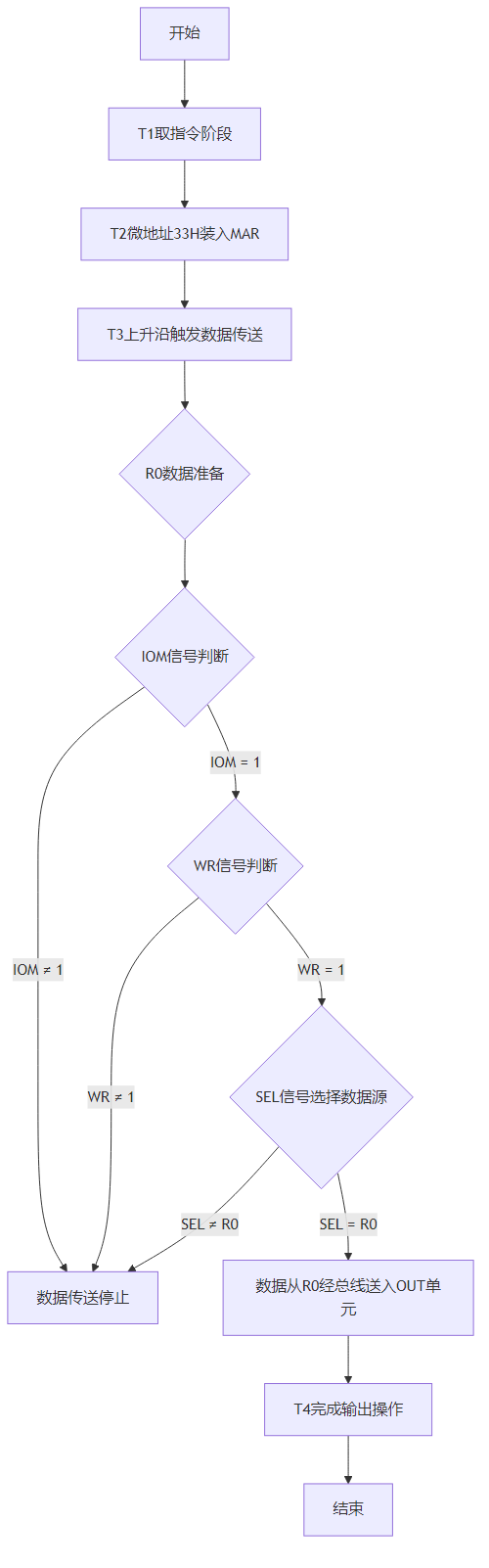
### **8.1.1 时序分析**

1. **数据传送时刻**

* OUT指令（30H）执行过程：  
  T1：取指令阶段  
  T2：微地址33H装入MAR  
  T3：R0数据送入OUT单元 ← 关键时刻  
  T4：完成输出操作

1. **逻辑类型判断**
   * 属于时序逻辑
   * 原因：
     + 数据传送受时钟信号控制
     + 需要在特定时序（T3）才能完成
     + 输出结果保持需要触发器支持
2. **控制信号分析**

* 影响数据传送的关键信号：  
  - T3：时序控制信号  
  - IOM：输入输出控制信号（=1）  
  - WR：写控制信号（=1）  
  - SEL：数据选择信号



**图15：30H指令送入OUT单元**

在流程图中：

* 从开始到T1取指令阶段、T2微地址装入MAR，逐步推进到T3时刻进行关键的数据传送判断。
* 在T3阶段，依次判断IOM、WR、SEL关键信号，只有当这些信号满足特定条件（IOM = 1开启I/O操作、WR = 1使能写操作、SEL选择R0作为数据源）时，数据才能从R0经总线成功送入OUT单元。
* 最后在T4完成输出操作并结束流程。

### **8.1.2 详细解释**

1. **时序逻辑的体现**
   * 数据传送必须在T3上升沿触发
   * OUT单元需要锁存机制保持数据
   * 整个过程同步于系统时钟
2. **关键信号的作用**

| **信号** | **状态** | **作用** |
| --- | --- | --- |
| T3 | 1 | 时序控制 |
| IOM | 1 | 开启I/O操作 |
| WR | 1 | 使能写操作 |
| SEL | R0 | 选择数据源 |

## **8.2 实验心得体会**

通过这次微程序控制器实验，我收获颇丰，不仅在知识层面有了更深入的理解，在技能和对实验过程的感悟上也取得了显著的进步。

### **8.2.1 理论知识深入**

在知识方面，此次实验让我真正将微程序控制器的理论知识与实际操作紧密结合起来。曾经在课本上学习到的微程序控制器组成原理、工作机制等知识，在亲手进行实验的过程中变得鲜活而具体。我深刻理解了每一个部件是如何协同工作来完成指令的翻译和执行的。在技能提升上，我掌握了**微程序编写**与调试这一关键技能。从最初小心翼翼地按照格式编写微程序，到通过不断调试使其正确运行，这个过程感觉很不容易。同时，数字系统时序分析能力也得到了锻炼，我学会了通过观察**各个时序点**的工作状态，来理解系统的运行逻辑，这对于今后深入研究计算机硬件系统至关重要。

实验过程中，我深切体会到了微程序控制的**灵活性**。通过编写不同的微程序，可以实现各种不同的指令功能，这种灵活性让我看到了计算机硬件设计的魅力所在。同时，我也深刻认识到硬件设计的严谨性。每一个开关的设置、每一条微指令的编写，都必须准确无误，否则就可能导致整个实验无法正常进行。所以在实验中应当养成严谨细致的工作习惯，每一个操作都反复核对，确保准确无误。

而调试过程则是整个实验的重中之重。在遇到微程序无法正常运行或者结果与预期不符的情况时，我需要仔细检查每一个步骤，从微指令的编写到时序的控制，不放过任何一个可能的问题点。这个过程虽然充满了挫折，但当最终成功解决问题时，那种成就感是无法言喻的。它让我明白，在面对复杂的硬件系统时，耐心和细心是解决问题的关键。

### **8.2.2 对实验的改进思考**

对于实验内容，我认为可以增加更多指令类型的实验，这样可以进一步拓展我们对微程序控制器的应用能力。同时，添加中断处理的实验内容也将使实验更加贴近实际的计算机系统运行情况，让我们更好地理解计算机在复杂环境下的工作机制。在实验方法上，引入自动化测试手段将大大提高实验效率，减少人为错误。增加波形显示功能可以更直观地观察时序信号的变化，有助于我们更好地理解系统的时序关系。提供更多调试工具则可以在遇到问题时更加方便快捷地进行排查和解决。

### **8.2.3 总结与展望**

通过这次实验，我对计算机组成有了全新的认识。我深刻理解了控制器在计算机系统中的核心地位，它就像计算机的大脑，指挥着各个部件有条不紊地工作。微程序设计的重要性也不言而喻，它为计算机指令系统的实现提供了一种高效而灵活的方式。同时，我也更加体会到硬件实现的复杂性，每一个小小的功能背后都需要众多部件的协同工作和精确控制。

在个人能力方面，这次实验全方位地提升了我的能力。硬件调试能力的提升让我在面对硬件问题时不再束手无策，而是能够通过系统的方法进行排查和解决。系统分析能力的增强使我能够从整体的角度去理解和把握计算机系统的运行逻辑。文档编写能力在撰写实验报告的过程中得到了锻炼，我学会了如何清晰准确地记录实验过程和结果。

展望未来，我将以此次实验为契机，继续深入学习计算机组成原理，进一步挖掘计算机硬件系统的奥秘。我希望能够提高自己的数字系统设计能力，在未来有机会参与到更复杂的计算机硬件项目中。同时，我也会不断加强硬件调试技能的训练，使自己成为一名在计算机硬件领域具备扎实专业能力的人才。这次实验不仅是一次学习的经历，更是我在计算机硬件领域探索道路上的一个重要里程碑，我将带着从这次实验中汲取的经验和教训，继续前行。