**网络空间安全学院**



**《计算机组成与系统结构》实验报告**

**姓 名 王鸣一**

**学 号 2023211475**

**班 级 2023211804**

**邮 箱 2023211475@bupt.cn**

**任课教师 苑 洁**

**2024年 11月**

**任课教师打分表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **项目** | | **满分** | **分数** | **备注** |
| 平时成绩 | 考勤  （四次签到） | 20分 |  |  |
| 演示验收 | 20分 |  |  |
| 报告成绩 | 实验1 | 15分 |  |  |
| 实验2 | 15分 |  |  |
| 实验3 | 15分 |  |  |
| 实验4 | 15分 |  |  |
| **合计** | | **100分** |  |  |

实验1.运算器组成实验

分别完成微程序模式和独立模式实验，报告记录独立模式实验，报告内容包括：实验目的、实验内容、实验过程以及可探索和研究的问题（P21第八题）。

1. **实验目的**

1、熟悉逻辑测试笔的使用方法。  
2、熟悉TEC-8模型计算机的节拍脉冲T1、T2、T3；  
3、熟悉双端口通用寄存器组的读写操作；  
4、熟悉运算器的数据传送通路；  
5、验证74LS181的加、减、与、或功能；  
6、按表中提供的功能自行验证其中几种即可。

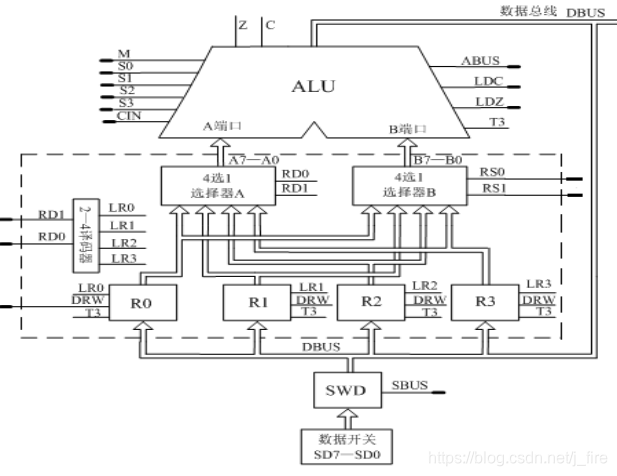
1. **实验内容**

1.按图所示，将运算器模块与实验台操作板上的线路进行连接。由于运算器模块内部的连线已经由印制电路板连接好，故接线任务仅仅是完成数据开关、控制信号模拟开关、与运算模块的外部连线。特别注意:为了建立清楚的整机概念，培养严谨的科研能力，手工连线时绝对有必要的。

2.用开关K15-KO向通用寄存器堆RF内的R3-RO寄存器置数据，然后读出R3-RO的数据，在数据总线DBUS上显示出来。

3.验证 ALU的正逻辑算术、逻辑运算功能。

注意:进位信号C是运算器ALU最高进Cn+4的反，既有进位为1，无进位为 0。



（实验一电路图）

1. **实验过程**
2. **数据存入**



首先连接好线路



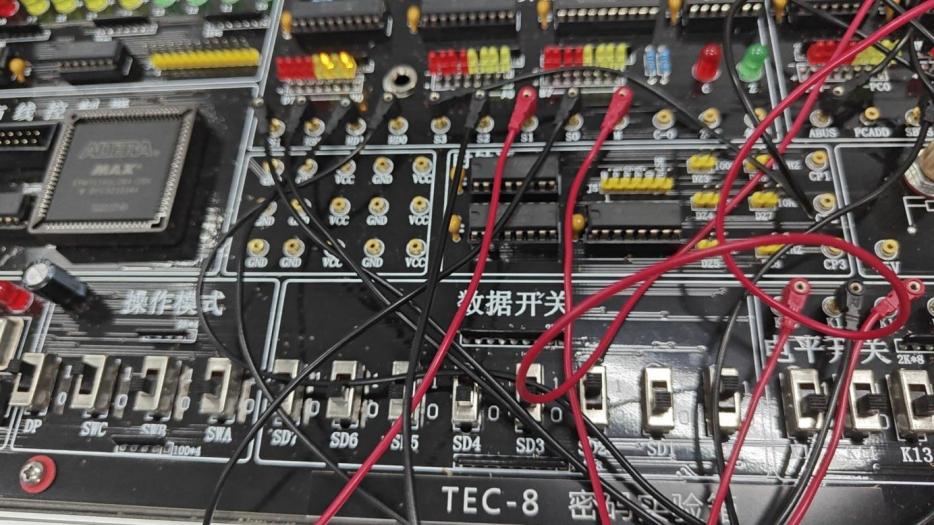
打开开关，黄色的灯亮，表示是独立模式

第一步，按一下CLR，清除计算机原来保存的数据，

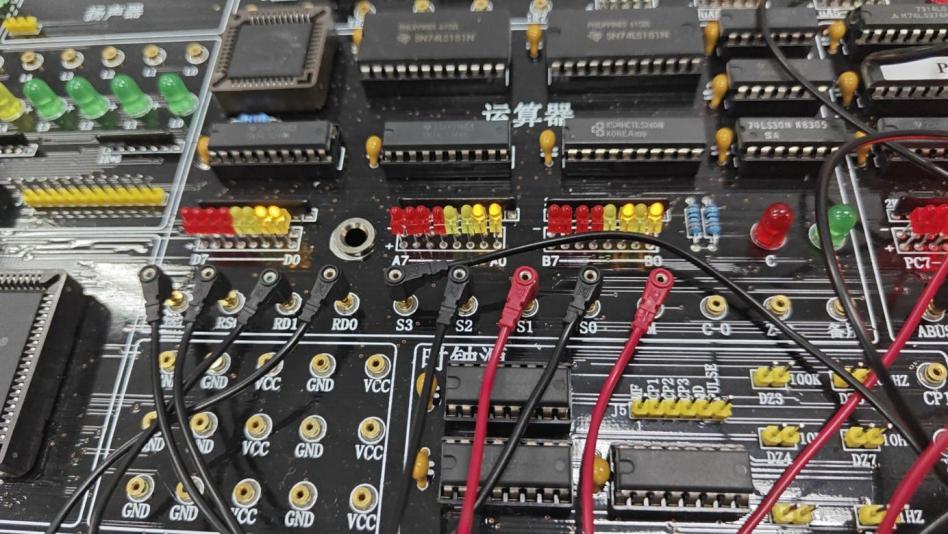
调节DP,SWC,SWR,SWA使操作模式为1100，表示可以存入数据

调节k3，也就是sbus，调为1，

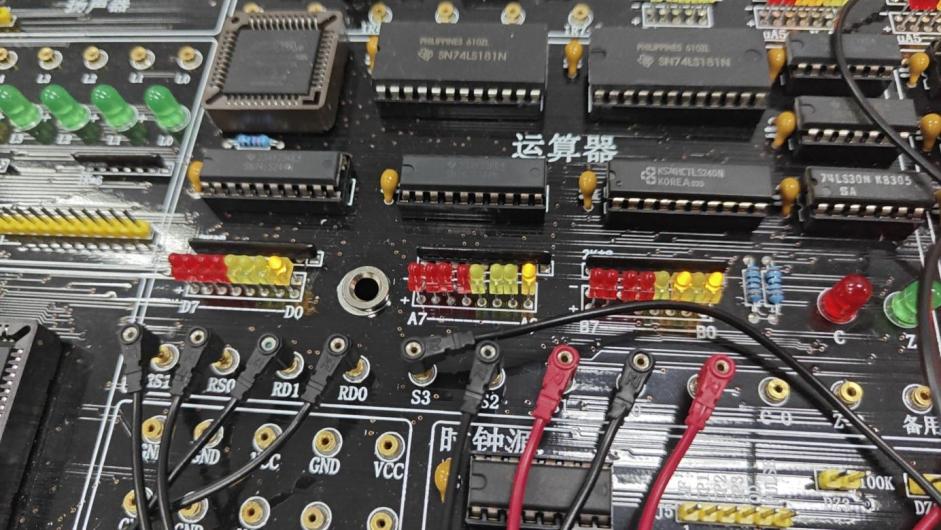
调节k4，也就是DRW，调为1，



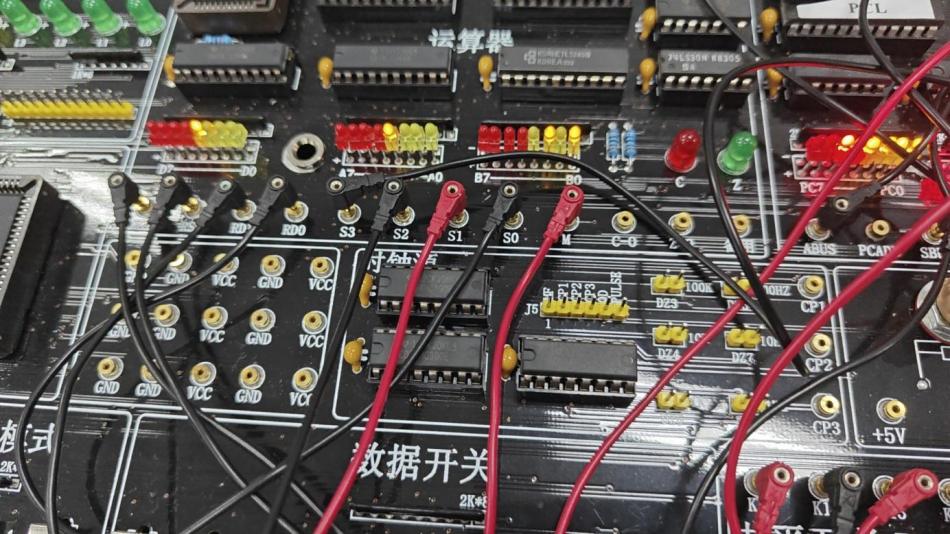
调节SD7~SD0将数据101，存入R0（k6，k5，为00）存储器



调节SD7~SD0将数据011，存入R1（k6，k5，为01）存储器



调节SD7~SD0将数据001，存入R2（k6，k5，为10）存储器

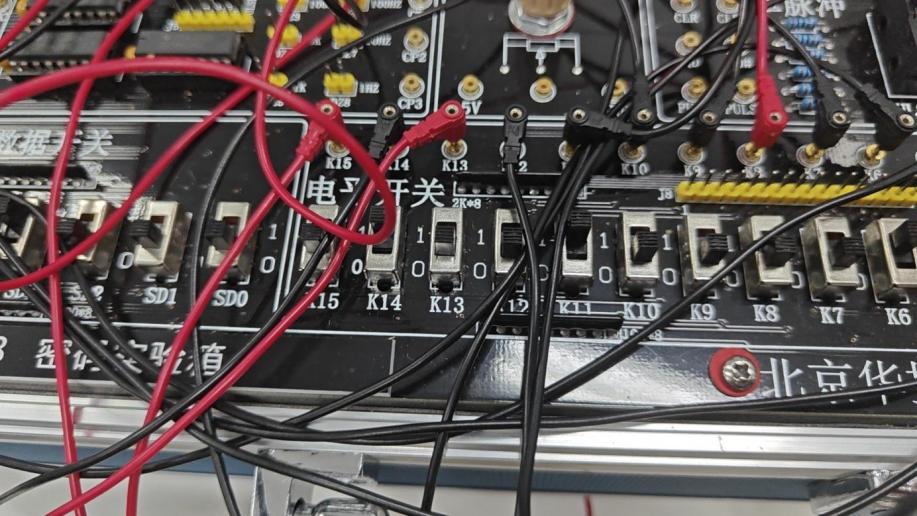


调节SD7~SD0将数据1000，存入R3（k6，k5，为11）存储器

完成存储。

1. **算术运算-加法**

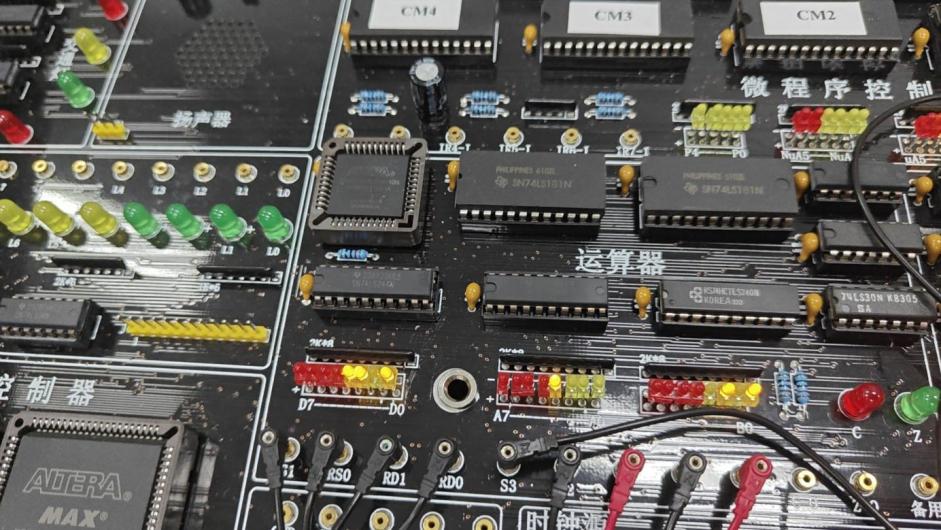
首先调节k15,使m为0，表示进行算术运算。



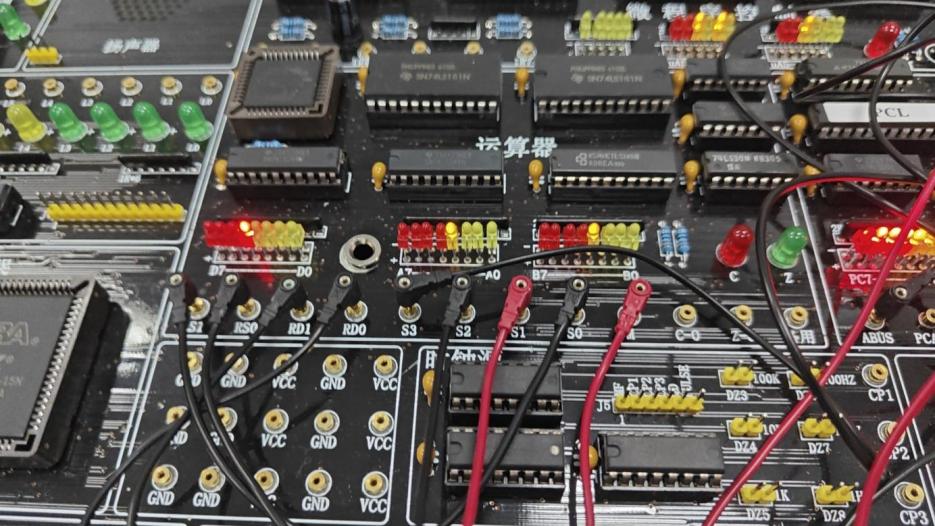
将k13到k11（s3到s0）调节成1001，表示进行加法，

调节k9(abus)为1，可以显示结果，

再将k10（cin）调为1，则可以进行加法了。



加数1000，被加数0101，结果为1101，结果正确。



加数1000，被加数1000，结果为10000，结果正确。

1. **算术运算-减法**

首先调节k10(cin)为0，可以显示进位，

将k13到k11（s3到s0）调节成0110，表示进行减法，

再将k9（abus）调节为1，使结果显示。

再将k8,k7,(LDC和LDZ)调为1，即可进行减法了。

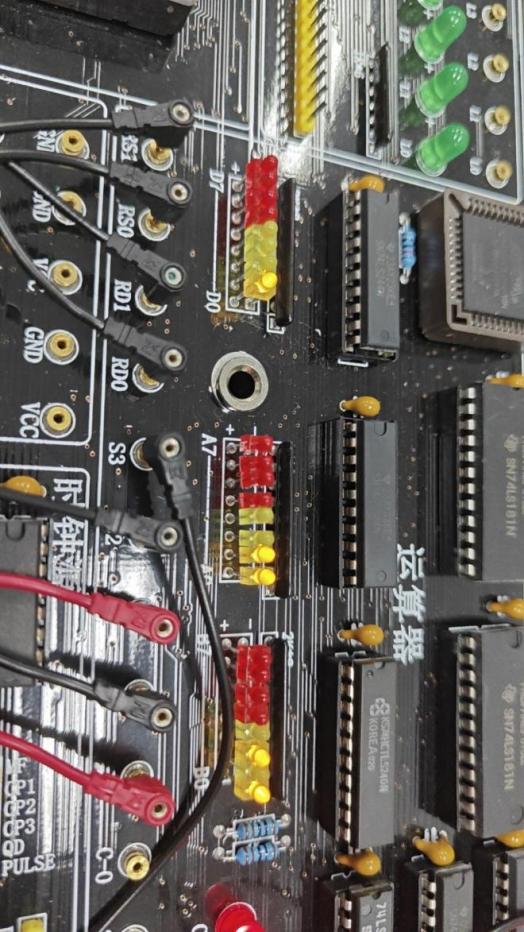


如图k6，k5，调为11，表示被减数为8，k2，k1，调为00，表示减数为5，8-5=3，即减法成立。

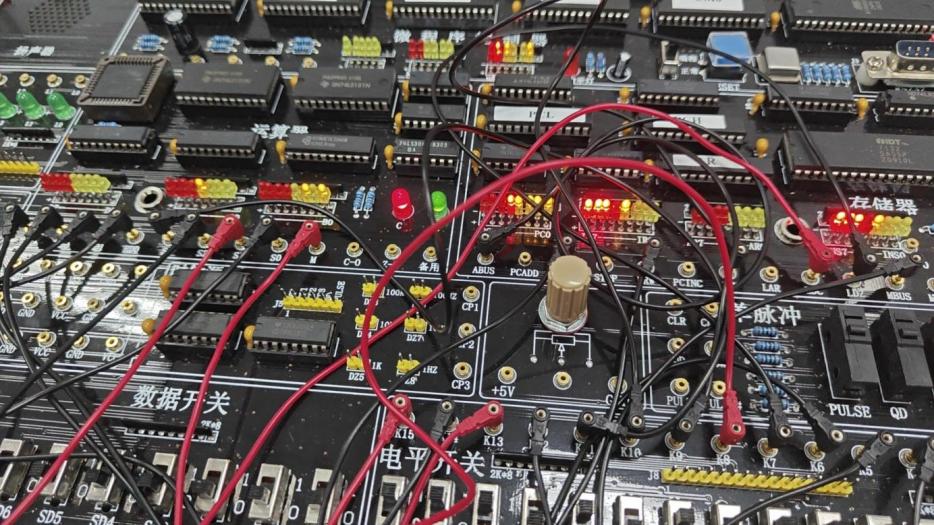
1. **逻辑运算-与**

首先，将k13到k11（s3到s0）调节成1011，k15(m)调节为1，表示进行逻辑与运算，

再将k9（abus）调节为1，表示显示运算结果。即可进行逻辑与运算。



011与101做逻辑与运算，结果为001，运算正确。



1000与0101做逻辑与运算，结果为0000，Z灯也亮了，表示运算正确。

1. **逻辑运算-或**

首先，将k13到k11（s3到s0）调节成1110，k15(m)调节为1，表示进行逻辑或运算，

再将k9（abus）调节为1，表示显示运算结果。即可进行逻辑或运算。



进行或运算，1000，或0101，结果为1101，结果正确。

1. **可探索和研究的问题**

1. ALU具有记忆功能吗？如果有，如何设计？

ALU本身不具有记忆功能。它的主要任务是执行算术和逻辑运算，如加法、减法、与、或、非等。ALU的设计通常包括一系列的逻辑门和电路，用于执行这些基本操作。如果需要记忆功能，通常需要额外的存储器组件，如寄存器或RAM。

设计一个具有记忆功能的ALU，可以通过以下几种方法实现：

使用寄存器存储结果\*\*：ALU本身不具有记忆功能，但可以通过与寄存器的结合来实现。寄存器可以存储ALU的计算结果，这样ALU就可以在需要时从寄存器中读取数据进行下一次计算。这种方法简单且有效，但需要额外的硬件资源。

状态机控制：通过设计一个状态机来控制ALU的操作，状态机可以根据ALU的当前状态和输入信号来决定ALU的下一步操作。状态机可以与寄存器配合使用，以存储中间结果和控制信号，从而实现更复杂的记忆功能。

多路选择器：使用多路选择器来选择ALU的不同输入和输出，这样可以在不同的操作之间进行切换，并且可以存储和重用之前的操作结果。这种方法可以提高ALU的灵活性，但可能会增加设计的复杂度。

流水线技术：在ALU设计中实现流水线技术，可以使得ALU在执行当前操作的同时，准备下一个操作的数据。这样，即使ALU本身不具有记忆功能，也可以通过流水线中的寄存器来存储中间结果，实现高效的数据处理。

可编程逻辑：使用可编程逻辑器件（如FPGA或CPLD）来设计ALU，可以灵活地实现各种复杂的逻辑和记忆功能。这种方法可以根据不同的需求来定制ALU的功能，但可能需要更多的设计工作和专业知识。

综上所述，设计具有记忆功能的ALU通常需要结合使用寄存器、状态机、多路选择器、流水线技术或可编程逻辑等方法。这些方法可以单独使用，也可以组合使用，以满足不同的设计需求和性能目标。

2. 为什么在ALU的A端口和B端口的数据确定后，在数据总线DBUS上能够直接观测运算的数据结果，而标志结果却在下一步才能观测到？

当ALU的A端口和B端口的数据确定后，ALU可以立即开始运算，并在数据总线DBUS上输出运算结果。这是因为数据运算是直接的，不需要额外的处理时间。

标志结果（如溢出、进位、零标志等）通常在运算完成后才能确定，因为它们依赖于运算结果的特定条件。例如，溢出标志需要比较结果是否超出了ALU能够表示的范围。这些标志的计算可能需要额外的逻辑电路来检测和设置，因此它们的结果会在运算完成后的下一步才能被观测到。

实验2.双端口存储器实验

分别完成微程序模式和独立模式实验，报告记录独立模式实验，报告内容包括：实验目的、实验内容、实验过程以及可探索和研究的问题（P29第八题）。

**一、实验目的**

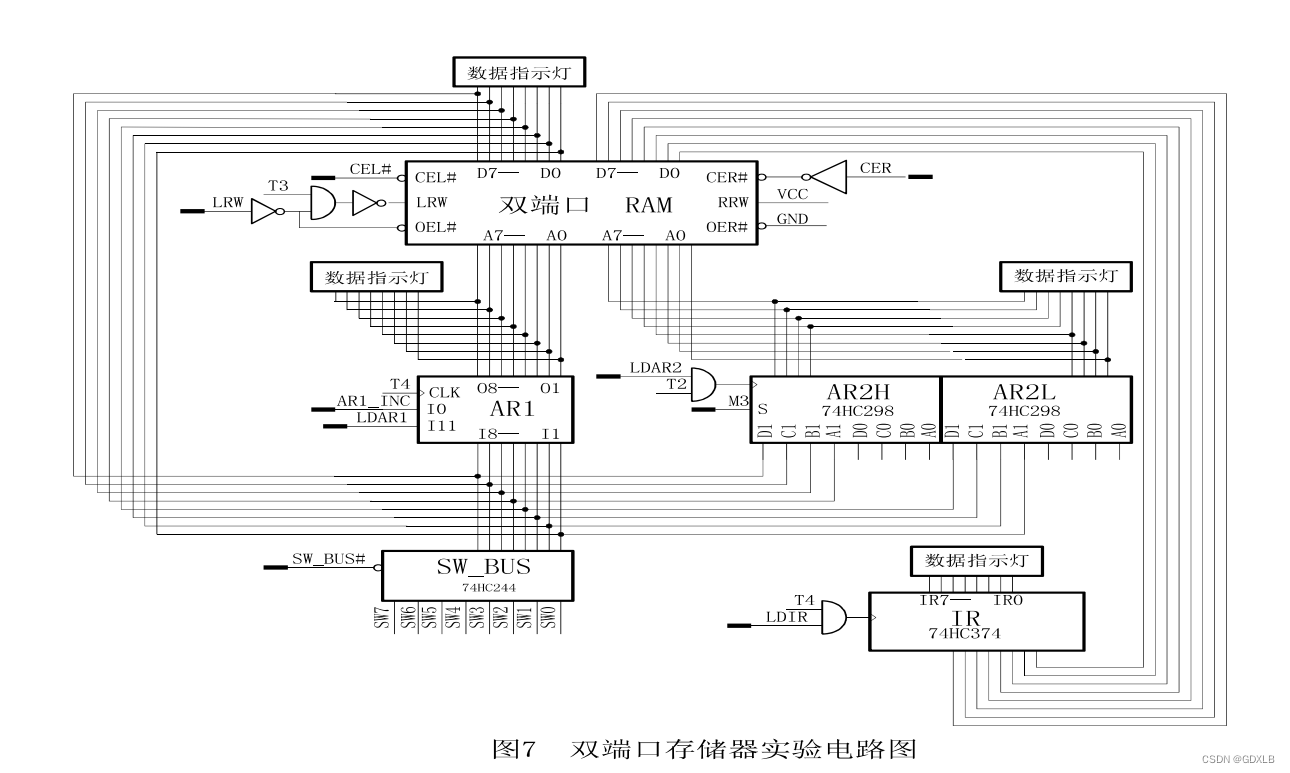
1、了解**双端口**静态随机**存储器**IDT7132的工作特性及使用方法；

2、了解半导体存储器怎样存储和读出数据；

3、了解双端口存储器怎样并行读写；

4、熟悉TEC-8模型计算机中存储器部分的数据通路。

**二、实验内容**



（实验二电路图）

1. 从存储器10H开始，通过左端口连续向双端口RAM中写入3个数：85H，60H，38H。在写的过程中，在右端口检测写的数据是否正确。
2. 从存储器地址10H开始，连续从双端口RAM的左端口和右端口同时读出存储器的内容。
3. **实验过程**

首先完成连线，然后设置操作模式1110；

第一步，SBUS（K15）播1，LAR（K13）播1，LPC（K9）播1，设地址为10H；



观察得到实现现象：AR，PC显示10。

第二步，SBUS（K15），ARINC（K14），MEMW（K12）播1，SD播85，点击QD，成功将85H存入R0；



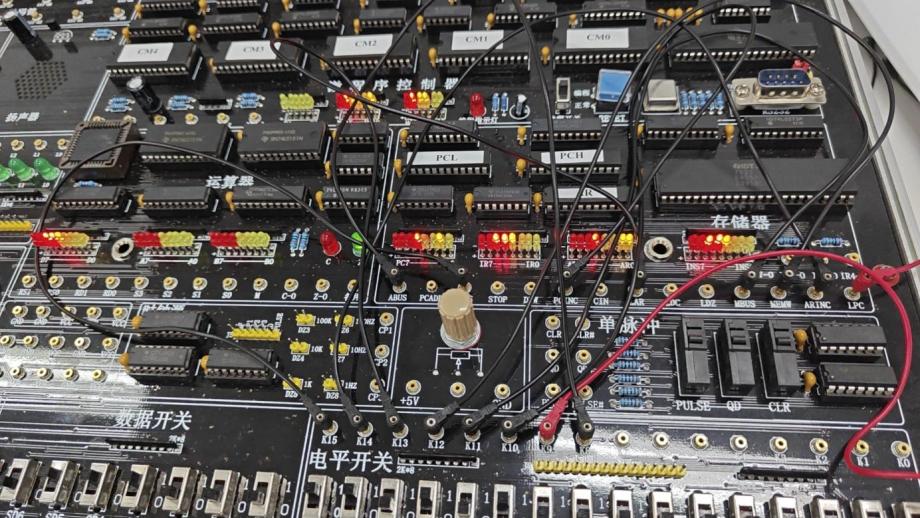
观察得到实现现象：AR显示11，PC显示10，INS与IR显示85.

第三步，SBUS（K15），ARINC（K14），MEMW（K12），PCINC（K10）播1；点击QD



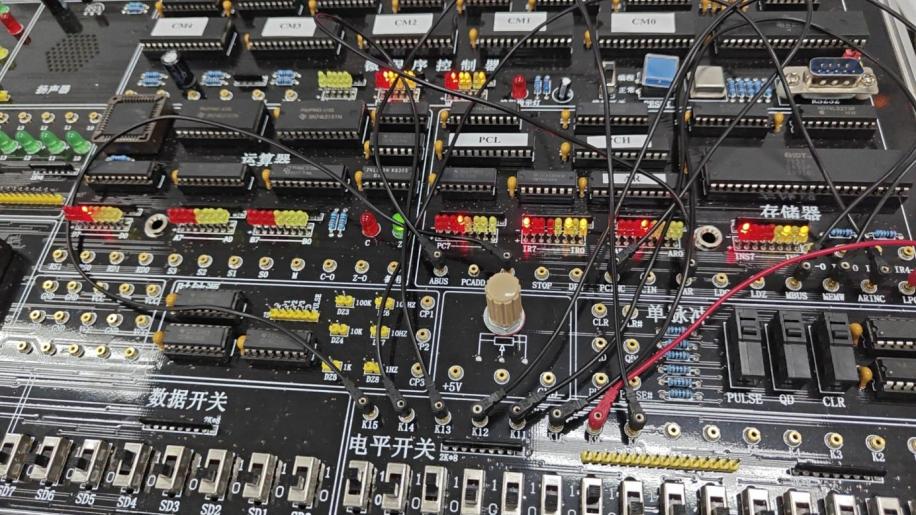
观察得到实验现象AR为12，PC为11，INS＝IR＝60。

第四步，SBUS（K15），ARINC（K14），MEMW（K12），PCINC（K10）播1，SD播38，点击QD；



观察得到实验现象AR为13，PC为12，INS＝IR＝38。

第五步：SBUS（K15），LAR（K13），LPC（K9）播1，SD播10，点击QD；



显示序号6的实验现象，AR＝PC＝10，INS＝85。

第七步：ARINC（K14），MEMW（K12），PCINC（K10）播1，按一下QD



显示序号7的实验现象，AR＝PC＝11，INS＝60。

第八步：ARINC（K14），MEMW（K12），PCINC（K10）播1，按一下QD



显示序号8的实验现象，AR＝PC＝12，INS＝38

**四、可探索和研究的问题**

1、在通过左端口向双端口RAM写数时，在右端口可以同时观测到左端口写入的数吗？为什么？

在通过左端口向双端口RAM写入数据时，在右端口是无法同时观测到左端口写入的数据的。这是因为在双端口RAM中，左端口和右端口是相互独立的，其内部电路也是单独的，因此左端口写入的数据不会直接传递到右端口。

左端口和右端口在处理数据时采用的是异步方式，因此它们之间的数据交换需要通过地址线和控制线进行同步。在写入数据到左端口时需要先发出写入地址和控制信号，告知RAM将数据写入指定地址中。而在读取数据时，需要发出读取地址和控制信号，RAM才会将数据从指定地址中读出并输出到右端口。因此，左端口写入的数据并不会直接传递到右端口，而是在RAM内部先进行存储和处理，然后再通过地址线和控制线向右端口输出数据。

此外，双端口RAM还存在写冲突的问题。当左端口和右端口同时写入不同数据到同一个地址时，由于只有一个数据总线，就会产生写冲突。为了避免写冲突，双端口RAM采用的是分时复用技术，即左端口和右端口轮流访问数据总线，每个时钟周期只有一个端口能够访问数据总线进行读写操作，这样就能够保证每个地址只会被一个端口写入相应的数据，从而避免了写冲突的问题。

综上所述，通过左端口向双端口RAM写入数据时，在右端口不会同时观测到左端口写入的数据。左端口和右端口是相互独立的。

实验3. 数据通路实验

分别完成微程序模式和独立模式实验，报告记录独立模式实验，报告内容包括：实验目的、实验内容、实验过程以及可探索和研究的问题（P36第八题）。

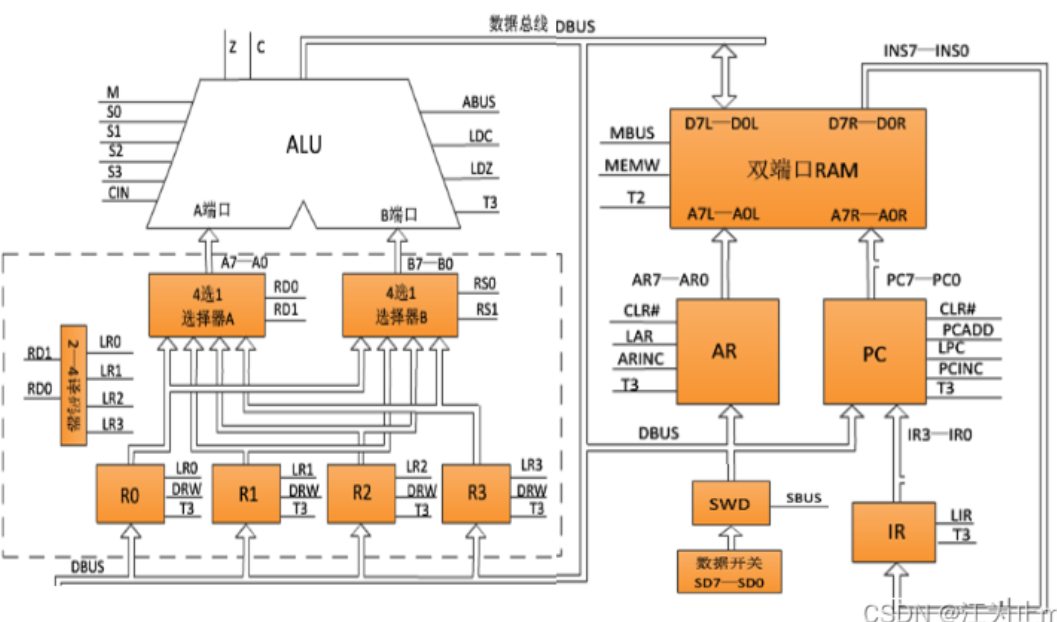
1. **实验目的**

1）进一步熟悉TEC-8模型[计算机](https://so.csdn.net/so/search?q=%E8%AE%A1%E7%AE%97%E6%9C%BA&spm=1001.2101.3001.7020" \t "https://blog.csdn.net/m0_63614804/article/details/_blank)的数据通路的结构

2）进一步掌握数据通路中各个控制信号的作用和用法

3）掌握数据通路中数据流动的路径

1. **实验内容**



（数据通路实验电路图）

将数75H写到寄存器R0，数28H写到寄存器R1，数89H写到寄存器R2，数32H写到寄存器R3；

将寄存器R0中的数写入存储器20H单元，将寄存器R1中的数写入存储器21H单元，将寄存器R2中的数写入存储器22H单元，将寄存器R3中的数写入存储器23H；

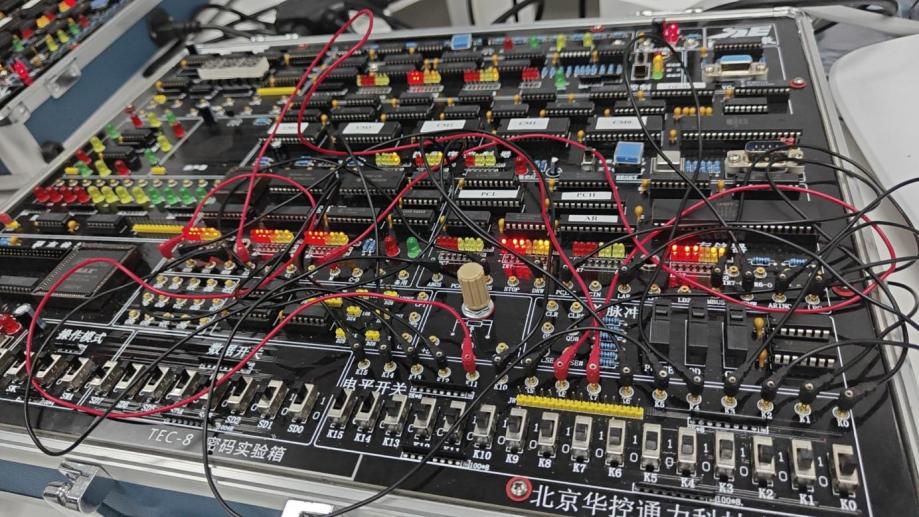
从存储器20H单元读出数到寄存器R3，从存储器21H单元读出数到寄存器R2，从存储器22H单元读出数到寄存器 R1，从存储器23H单元读出数到寄存器R0；

显示4个寄存器R0、R1、R2、R3的值，检查数据传送是否正确。

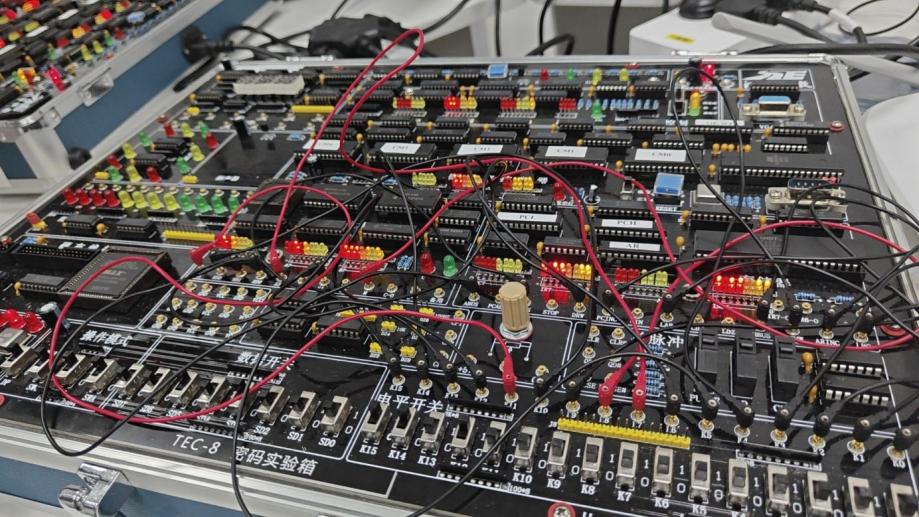
1. **实验过程**

打开开关，将操作模式设为1111，进行存数据

先将75H存入R0，



将28H存入R1，



将89H存入R2，



将32H存入R3。



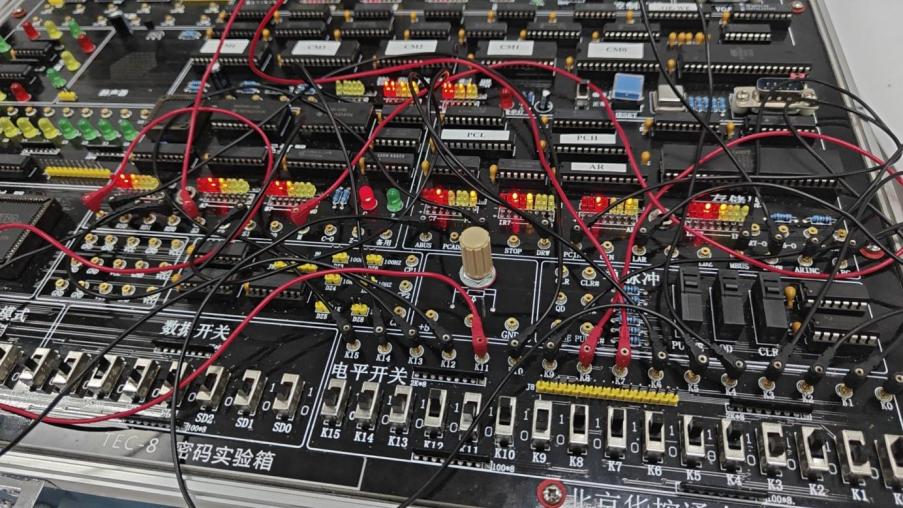
送存储地址20H



将R0的数据写入20H，75存入20H；



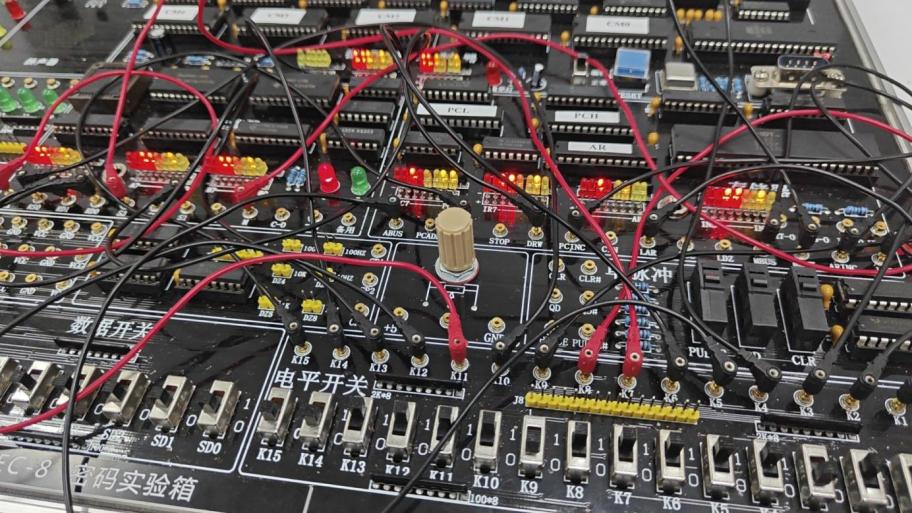
将R1的数据写入21H，28存入21H；



将R2的数据写入22H，89存入22H；



将R3的数据写入23H，32存入23H；



再次送存储地址20H；

点击QD，把20H的数据写入R3，R3变为75；



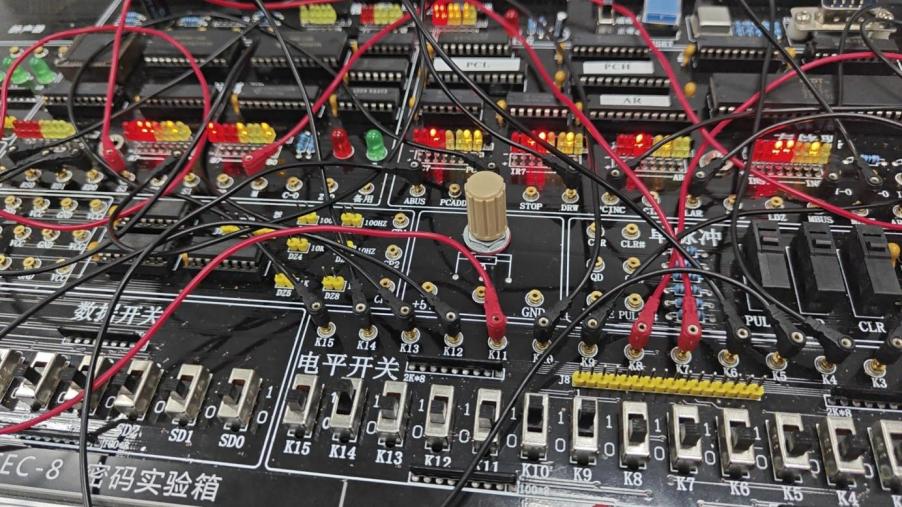
点击QD，把21H的数据写入R2，R2变为28；



点击QD，把22H的数据写入R1，R1变为89；



点击QD，把23H的数据写入R0，R0变为32。



最后再次观察R0，R1，R2，R3的值，R0为32，R1为89，R2为28，R3为65，实验完成。

1. **可探索和研究的问题**
2. 如果用I-cache 和 D-cache来代替双端口存储器,请提出一种数据通路方案

**数据通路设计**

****1）I-cache和D-cache的分离****

I-cache用于存储CPU即将执行的指令；

D-cache用于存储CPU即将访问的数据；

两者通过不同的数据通路与CPU相连，实现指令和数据的并行访问。

****2）指令访问通路****

CPU通过程序计数器（PC）提供指令地址；

该地址被发送到I-cache，I-cache尝试命中该指令；

如果I-cache命中，指令被直接发送到CPU的指令寄存器（IR）；

如果I-cache未命中，则需要从主存中读取指令，并更新I-cache。

****3）数据访问通路****

CPU通过地址生成单元（AGU）提供数据地址；

该地址被发送到D-cache，D-cache尝试命中该数据；

如果D-cache命中，数据被直接发送到CPU的数据寄存器（DR）或用于ALU的运算；

如果D-cache未命中，则需要从主存中读取数据，并更新D-cache；

****4）访问冲突处理****

由于I-cache和D-cache是分离的，它们可以同时访问不同的地址而不会发生冲突；

如果I-cache和D-cache需要同时访问相同的地址（例如，在自修改代码的情况下），则需要通过特定的机制来处理这种冲突。例如，可以设置一个仲裁器来决定哪个缓存优先访问该地址，或者通过软件来维护缓存之间的一致性。

****5）一致性维护****

在自修改代码的情况下，需要确保I-cache和D-cache之间的一致性。这可以通过硬件或软件来实现。硬件方案可能包括在D-cache写回时检查I-cache是否命中相同地址，并相应地更新I-cache。软件方案则需要在修改指令时显式地清除I-cache中相关的条目。

实验4. 微程序控制器实验

完成实验并报告记录，报告内容包括：实验目的、实验内容、实验过程以及可探索和研究的问题（P46第八题）。

1. **实验目的**
2. 掌握微程序控制器的原理；
3. 掌握TEC-8模型计算机中微程序控制器的实现方法，尤其是微地址转移逻辑的实现方法；
4. 理解条件转移对计算机的重要性。
5. **实验内容**

1、正确设置模式开关SWC、SWB、SWA，用单微指令方式跟踪除停机指令STP之外的所有指令的执行过程，记录下每一步的微地址μA5~µA0、判别位P4~P0和有关控制信号的值。对于JZ指令，跟踪Z=1、Z=0两种情况；对于JZ指令，跟踪C=1、C=0两种情况。

1. **实验过程**

**1、写存储器模式**

首先调节DP，SWC～SWA为1001，按QD表示进入写存储器模式，再设置地址为20H，按一下QD；设置数据为75H，按一下QD，将75H存入20H，按一下CLR结束写存储器模式。此时，微地址u\_A5-μ\_A0为00H，测得的微码分别是00H、05H、COH、02H和07H。

1. **读存储器模式**

将SWC～SWA设为010，按一下QD，再设置地址20H，按一下QD，读出20H中数据75H。



按一下CLR，结束读存储器模式。

微地址μ\_A5-μ\_A0为17H,测得的微码00H、05H、44H、00H、3FH。

1. **写寄存器模式**

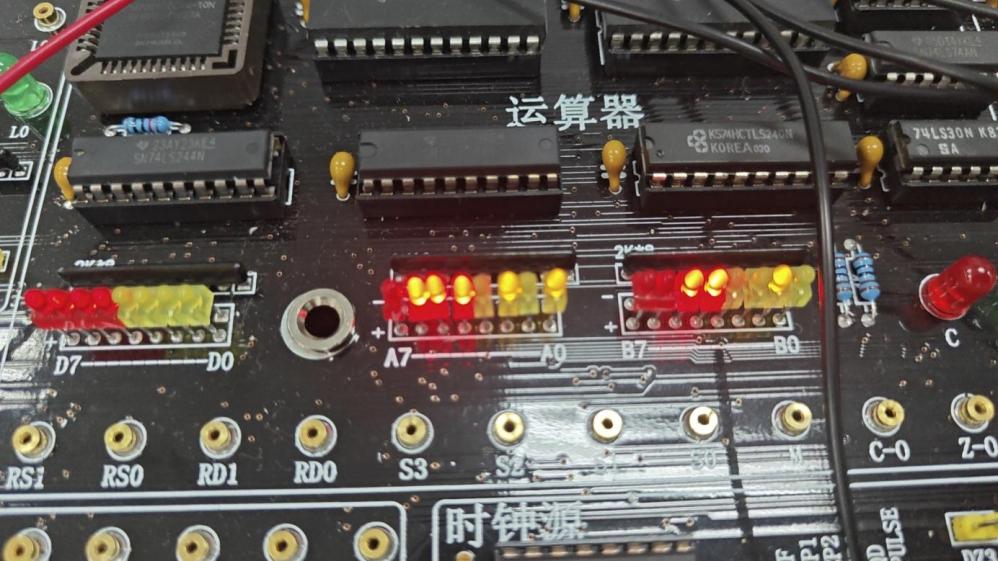
把SWC～SWA设为100，按一下QD，进入写寄存器模式。

依次将75H写入R0，将32H写入R1，将28H写入R2，将ABH写入R3。按一下CLR结束写寄存器模式。

微地址μ\_A5-μ\_A0为3FH,测得的微码是04H 09H C0H 00H 3EH。

1. **读寄存器模式**

按一下QD，读出R0的75H，与R1的32H。



再按一下QD，读出R2的28H，与R3的ABH



微地址μ\_A5-μ\_A0为3EH，测得的微码是00H 05H C8H 00H 3FH。

**5、启动程序运行模式**

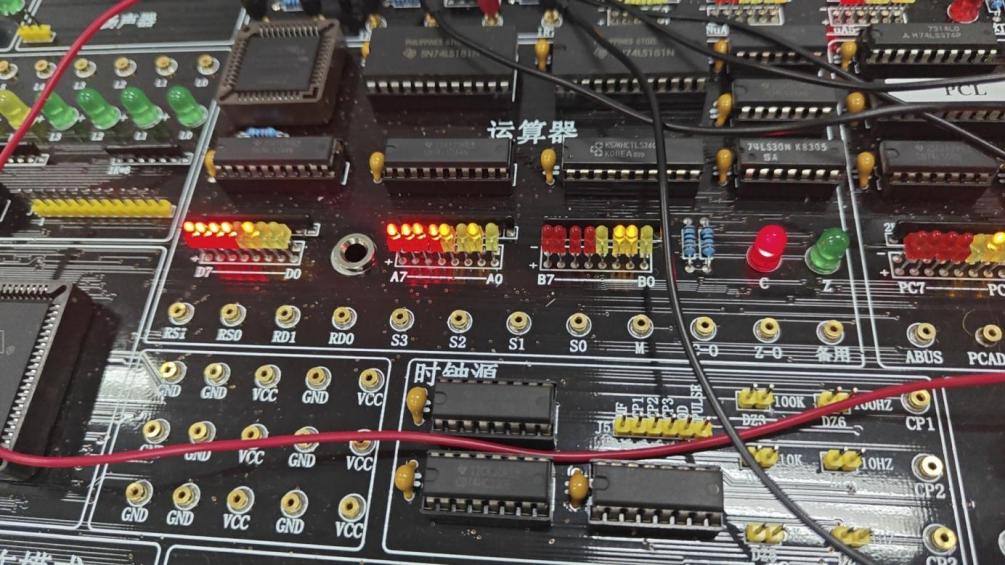
设置SWC～SWA为000，按QD，启动程序运行，

设置IR（7~4）为0001，验证加法功能；



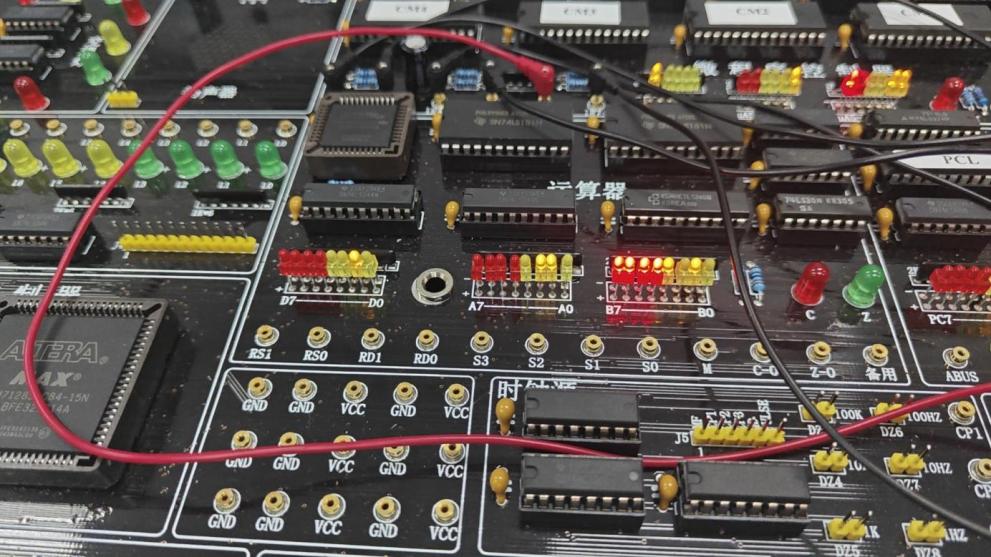
如图，D7~D0表示的数为A7~A0表示的数与B7~B0表示的数相加.

设置IR（7~4）为0010，验证减法功能；



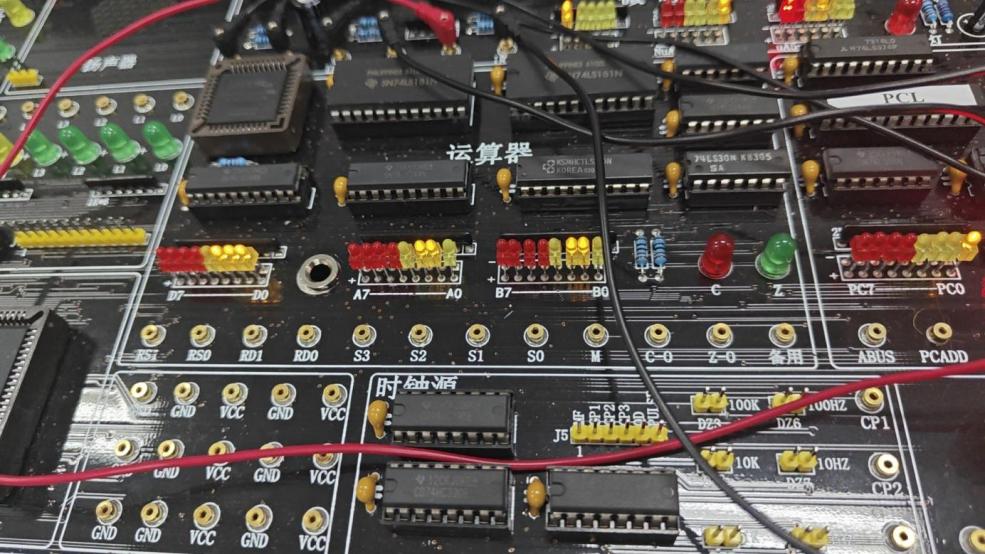
如图，D7~D0表示的数为A7~A0表示的数与B7~B0表示的数相减.

设置IR（7~4）为0011，验证逻辑与功能；



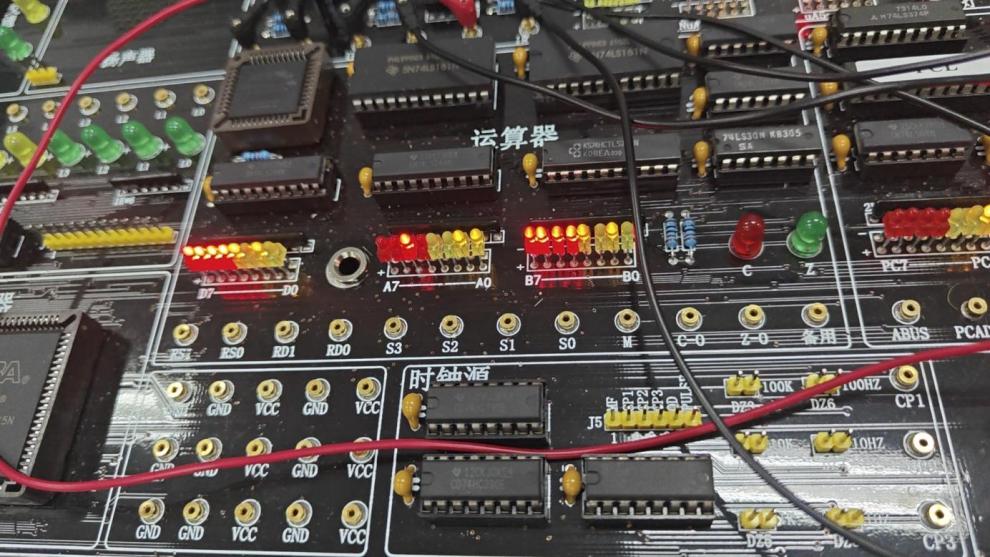
如图，D7~D0表示的数为A7~A0表示的数与B7~B0表示的数的各位逻辑与结果.

设置IR（7~4）为0100，验证加1功能；



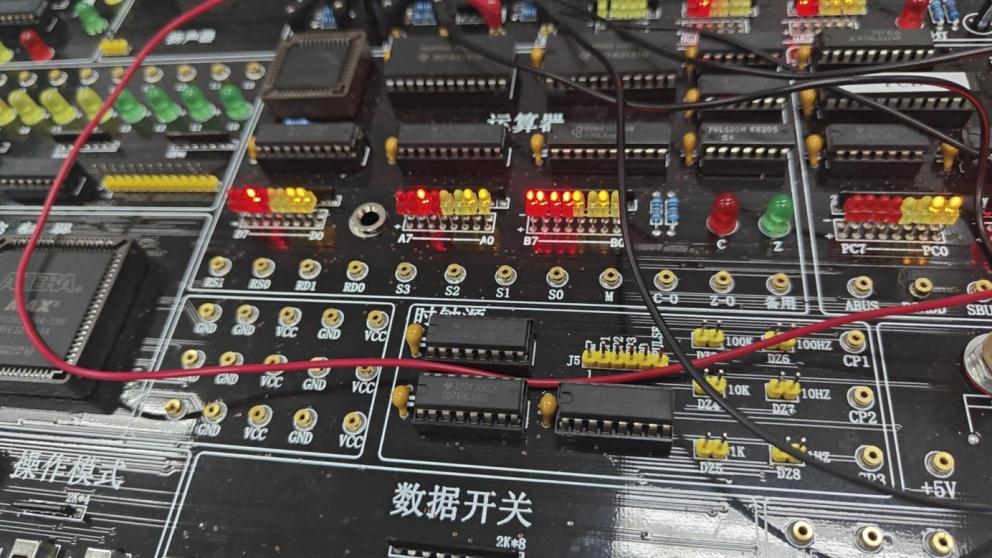
如图，D7~D0表示的数为A7~A0表示的数+1.

设置IR（7~4）为0101，验证取数功能；



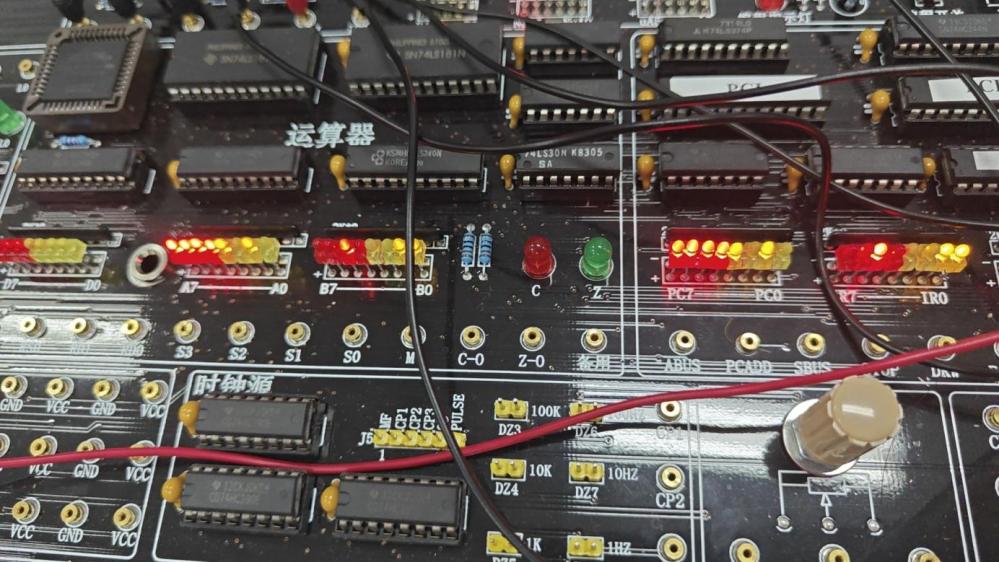
如图，D7~D0表示的数为B7~B0表示的数，即完成取数。

设置IR（7~4）为0110，验证存数功能；



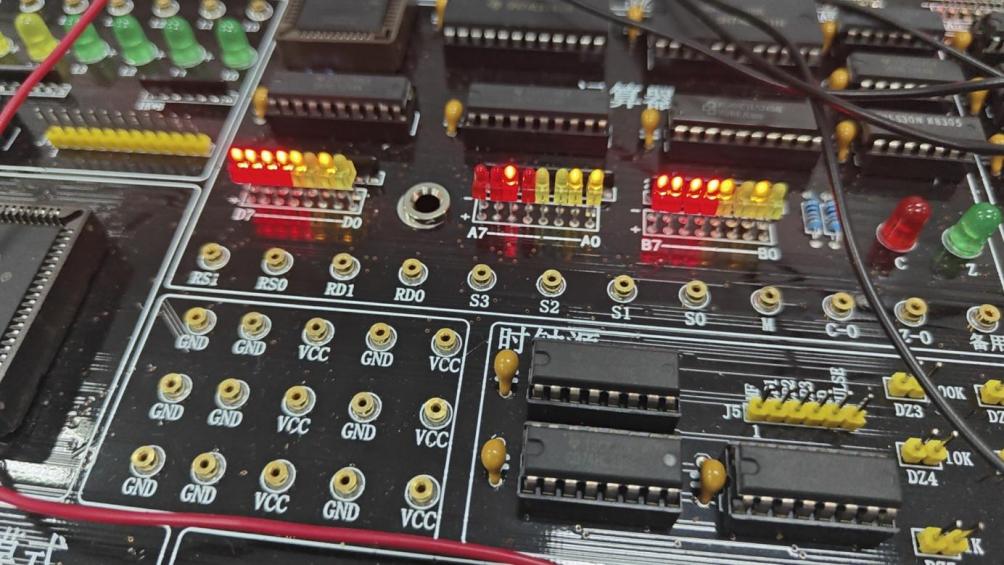
如图，D7~D0表示的数为A7~A0表示的数，即完成存数。

设置IR（7~4）为1001，验证无条件转移功能；



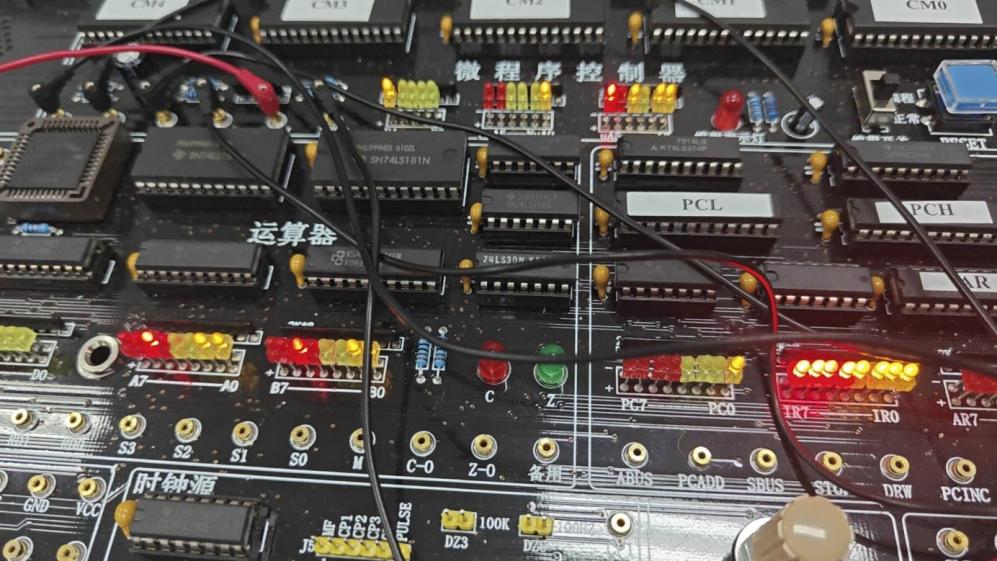
如图，D7~D0表示的数与PC7~PC0表示的数相等，B7~B0表示的数与IR7~IR0表示的数相等，即完成无条件转移。

设置IR（7~4）为1010，验证输出功能；



如图，D7~D0表示的数与B7~B0表示的数相等，即完成输出。

设置IR（7~4）为1011，观测中断返回功能的实验结果；



如图，中断返回的实验结果为当PC7~PC0为01H，且µA5~µA0信号显示2BH

设置IR（7~4）为1100，观测关中断功能的实验结果；



如图，关中断的实验结果为当PC7~PC0为01H，且µA5~µA0信号显示01H

设置IR（7~4）为1101，观测开中断功能的实验结果；



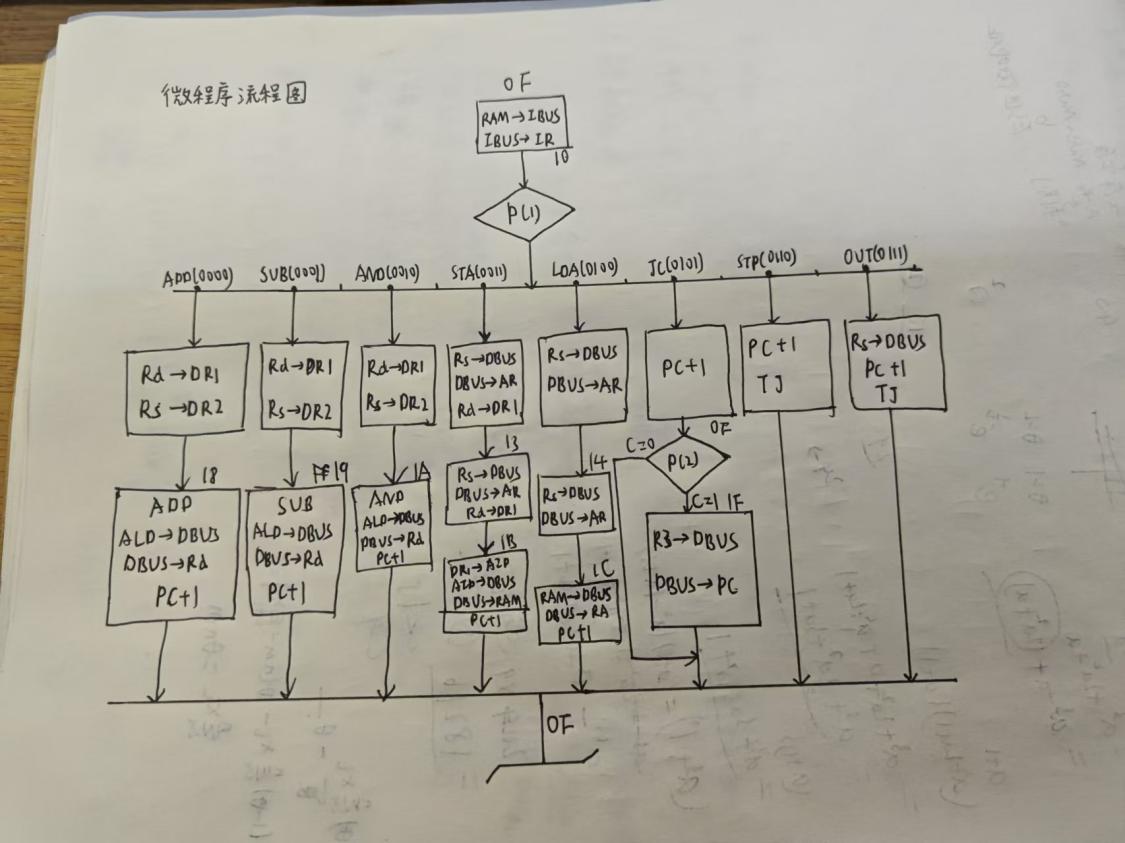
如图，开中断的实验结果为当PC7~PC0为01H，且µA5~µA0信号显示2DH

设置IR（7~4）为1110，观测停机功能的实验结果；

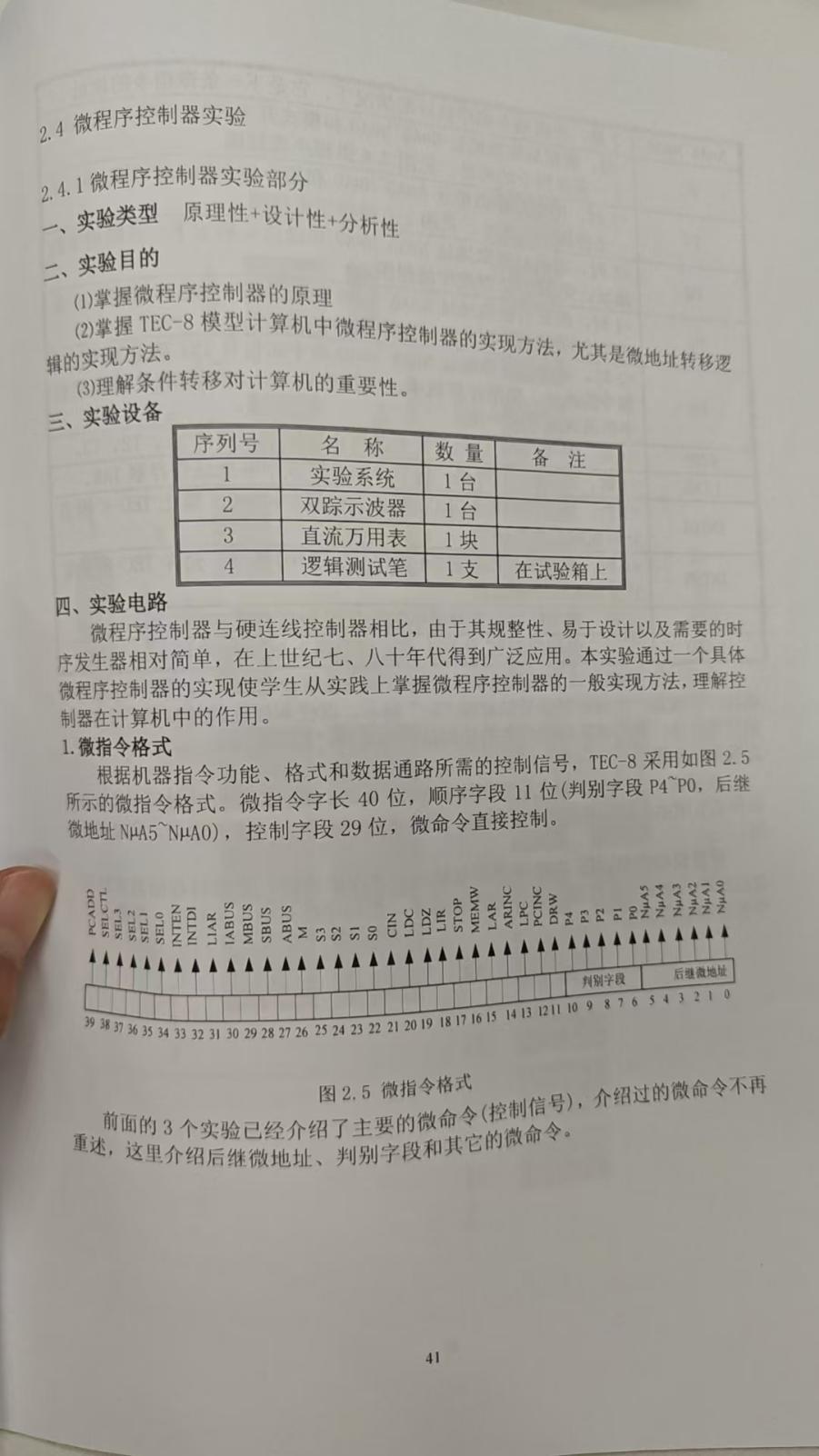


如图，停机的实验结果为当PC7~PC0为01H，且µA5~µA0信号显示2EH

1. **可探索和研究的问题**
2. 根据运算器组成实验、双端口存储器实验和数据通路实验的实验过程，画出这部分的微程序流程图。



1. 你能将图2.5中的微指令格式重新设计压缩长度吗？



如图2.5，原微指令格式共40位微指令字长，包括11位顺序字段（其中包括5位判别字段和6位后继微地址NµA5~NµA0）和29位控制字段。

可以从29为控制字段中拿出24位变为3\*8位，8就是2的三次方，3可以用两位表示，由3位地址和2位控制组合成这24位，就是前5位每位对应一个微指令，再有两位可以是01，10，11分别对应那三组微指令，也就是变成了5+2+3+11=21位。

住：这24位对应24条微指令，将这24条微指令分为3组，每组8条互相不冲突的微指令，以防止该指令系统的每一条机器指令不会同时要用哪一组中的多条指令。

即原来的微指令格式为：

[控制信号（29位）][判别（5位）][后继地址（6位）]

若经过分析后决定：控制信号可以通过编码减少到10位地址

新格式设计为：[控制信号（5位）（2位）（3位）][判别(5位)][后继地址(6位)]

这样总共从40位压缩到21位，达到了压缩目的。