



计算机组成原理课程设计

姓　　名　　 毛圣钧

班　　级　　 网络工程01

学　　号　　 202103151511

提交日期　　 2023.5.22

目录

[一、实验目的 1](#_Toc103858468)

[二、实验内容 1](#_Toc103858469)

[三、实验原理 1](#_Toc103858470)

[3.1指令系统及分析 1](#_Toc103858471)

[3.2指令框图及分析 1](#_Toc103858472)

[3.3指令系统对应微程序二进制代码实现及分析 1](#_Toc103858473)

[3.4机器程序及分析 1](#_Toc103858474)

[四、实验步骤 2](#_Toc103858475)

[4.1微程序写入及校验 2](#_Toc103858476)

[4.2机器程序写入及校验 2](#_Toc103858477)

[五、实验结果及分析 2](#_Toc103858478)

[5.1演示程序一 2](#_Toc103858479)

[5.2演示程序二 2](#_Toc103858480)

[六、实验问题及思考 2](#_Toc103858481)

[七、实验总结 3](#_Toc103858482)

[7.1验收答辩问题回答 3](#_Toc103858483)

[7.2实验心得体会 3](#_Toc103858484)

# 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复杂模型机。从而理解并掌握计算机的整机结构；掌握基本功能模块的工作原理和操作方法；掌握控制器的微程序实现；训练逻辑分析和故障排查能；理解并掌握机器指令、微指令各个字段的含义。

# 二、实验内容

1. 设计并实现一套完整的指令系统；
2. 利用上述指令系统，设计并实现完整的计算机；
3. 利用该计算机实现通过海伦公式计算三角形面积的平方的功能

# 三、实验原理

## 3.1指令系统及分析

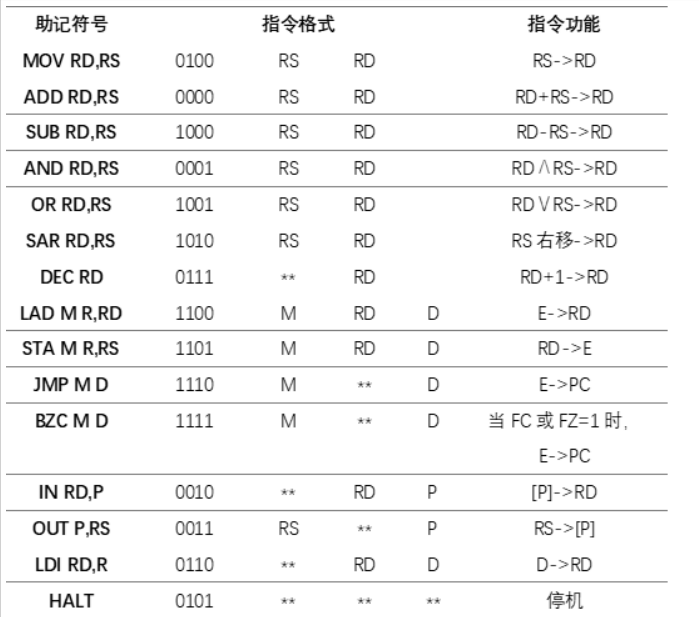
### 3.1.1指令设计

根据海伦公式计算中所需要的指令，设计三大类模型机指令共十五条，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、DEC、SUB、OR、SAR，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。15条指令如下



### 3.1.2指令格式

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

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 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 | R0 |
| 01 | R1 |
| 10 | R2 |
| 11 | R3 |

LDI 的指令格式如下，第一字节内容与上文同，第二字节为立即数

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7-0(2) |
| OP-CODE | | | | RS | | RD | | DATA |

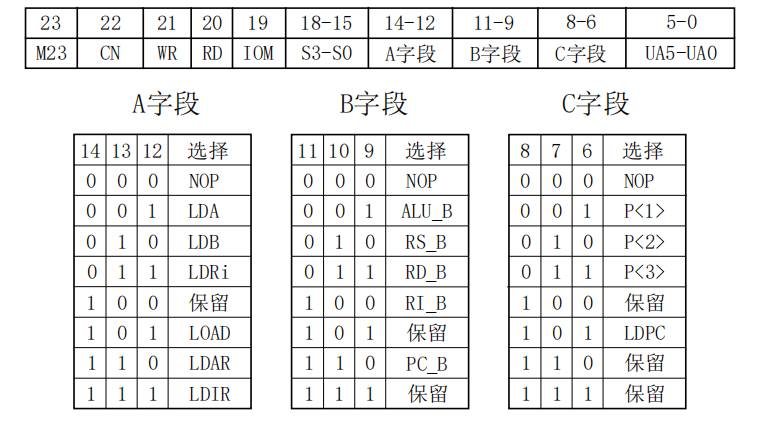
LAD、STA、JMP 和 BZC 指令格式如下

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 | 7-0(2) |
| OP-CODE | | | | M | | RD | | DATA |

其中 M 为寻址模式，具体见表 2，以 R2 作为变址寄存器 RI

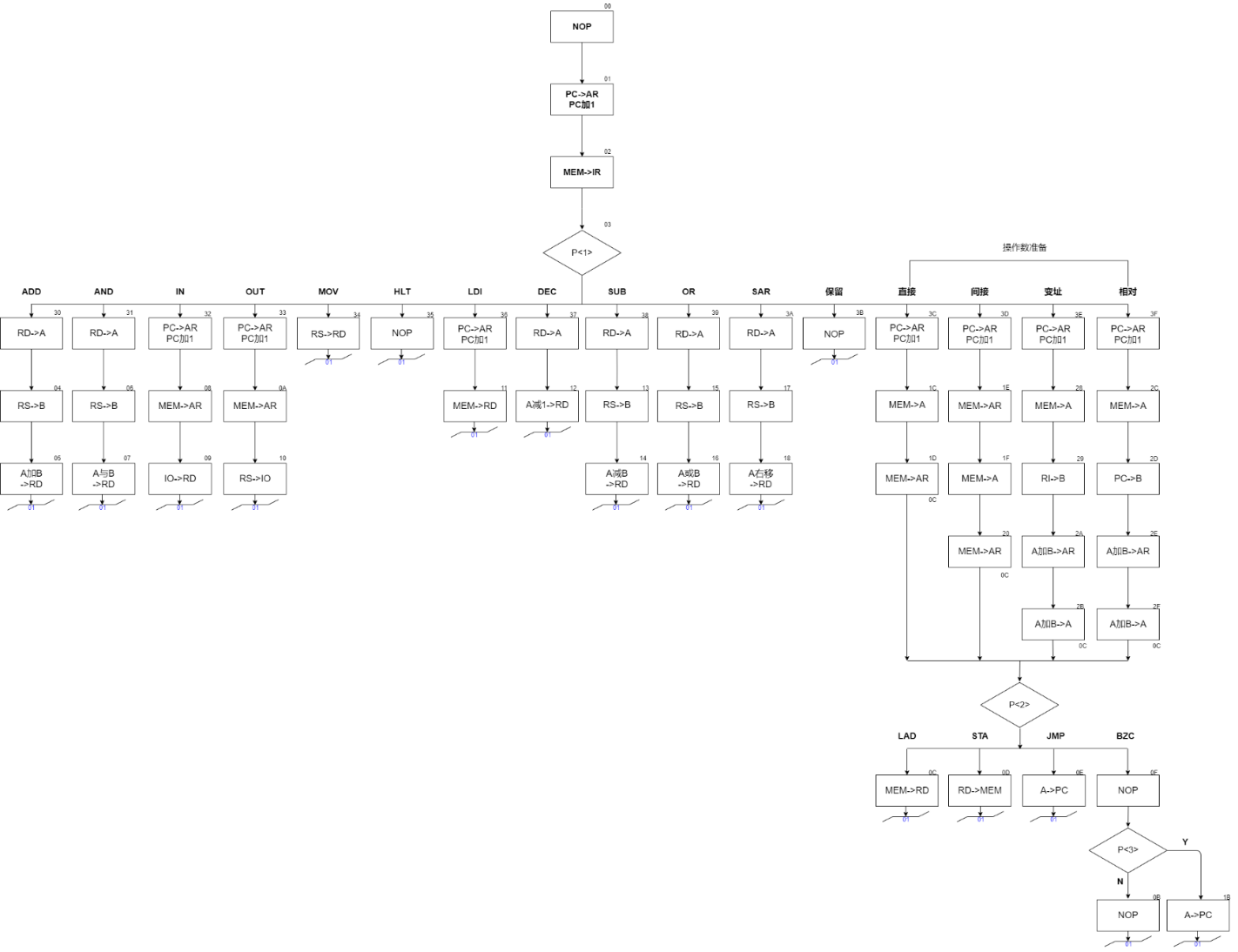
|  |  |  |
| --- | --- | --- |
| 寻址模式M | 有效地址E | 说明 |
| 00 | E=D | 直接寻址 |
| 01 | E=(D) | 间接寻址 |
| 10 | E=(RI)+D | RI变址寻址 |
| 11 | E=(PC)+D | 相对寻址 |

### 3.1.3微指令格式



## 3.2指令框图及分析

根据设计的机器指令要求，设计微程序流程图及确定微地址，如下图所示。



## 3.3指令系统对应微程序二进制代码实现及分析

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，如下表，并将将二进制代码表转化为联机操作时的十六进制格式文件。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | 011 | 000 | 000101 |
| 05 | 04 B2 01 | 00000 | 1001 | 011 | 001 | 000 | 000001 |
| 06 | 00 24 07 | 00000 | 0000 | 010 | 011 | 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 | 001 | 100 | 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 32 01** | **00000** | **1100** | **011** | **001** | **000** | **000001** |
| 13 | 00 24 14 | 00000 | 0000 | 010 | 011 | 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 | 011 | 000 | 011000 |
| **18** | **03 32 01** | **00000** | **0110** | **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 |

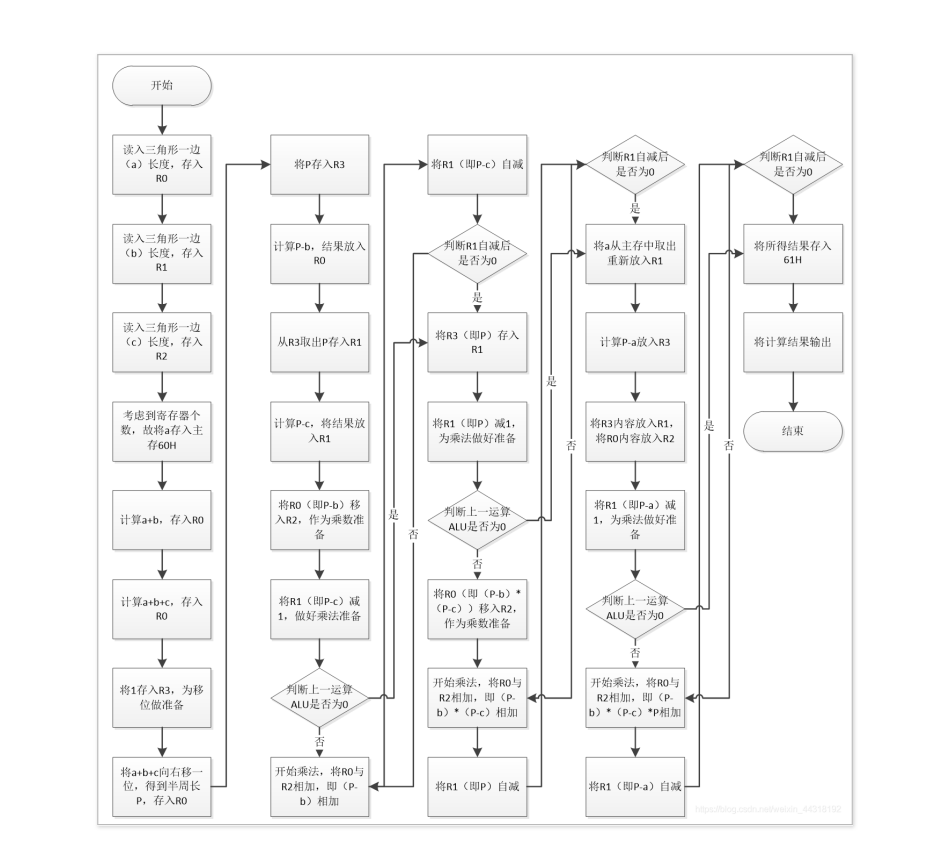
红色标注的为本系统中修改的微指令

地址12的微指令代表了减一运算A-1->RD，MA5~MA0为6位的后续微地址，表示下一条微指令的地位为01，A字段为011表示LDRi，代表第一个操作数来自寄存器，B字段为001表示ALU\_B，代表操作数B是ALU模块第二个操作数，这里ALU\_B为1，C字段为000表示NOP，代表不执行任何操作，S3~S0为1100，表示算术运算自减1，高五位全0，不进行任何操作。

地址为18的微指令代表了位运算A右移->RD，MA5~MA0为6位的后续微地址，表示下一条微指令的地位为01，A字段为011表示LDRi，代表第一个操作数来自寄存器，B字段为001表示ALU\_B，代表操作数B是ALU模块第二个操作数，这里ALU\_B为1，C字段为000表示NOP，代表不执行任何操作，S3~S0为0110，表示逻辑右移1位，高五位全0，不进行任何操作。

## 3.4机器程序及分析

根据实验计算海伦公式的目标，以及基于此设计的机器指令，在模型机实现以下运算：从 IN 单元读入三角形三边（要求输入三边一定能构成三角形，且半周长为偶数），求此三角形的面积平方，将所求结果存于 61H 单元，机器指令流程图如下图所示。



根据流程图设计程序，如下表所示，地址和内容均为二进制

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 地址 | | | 内容 | 助记符 | | 说明 |
| 00000000 | | | 00100000 | IN R0,00H | | 读入 a |
| 00000001 | | | 00000000 |  | |  |
| 00000010 | | | 00100001 | IN R1,00H | | 读入 b |
| 00000011 | | | 00000000 |  | |  |
| 00000100 | | | 00100010 | IN R2,00H | | 读入 c |
| 00000101 | | | 00000000 |  | |  |
| 00000110 | | | 11010000 | STA 60H,R0 | | 将 a 存入 60H |
| 00000111 | | | 01100000 |  | |  |
| 00001000 | | | 00000100 | ADD R0,R1 | | a+b 存入 R0 |
| 00001001 | | | 00001000 | ADD R0,R2 | | a+b+c 存入 R0 |
| 00001010 | | | 01100011 | LDI R3,01H | | 将 1 存入 R3 |
| 00001011 | | | 00000001 |  | |  |
| 00001100 | | | 10101100 | SAR R0,R3 | | a+b+c 向右移一位，R0=P |
| 00001101 | | | 01000011 | MOV R3,R0 | | 将 p 移入 R3 |
| 00001110 | | | 10000100 | SUB R0,R1 | | R0=P-B |
| 00001111 | | | 01001101 | MOV R1,R3 | 将 P 存入 R1 | |
| 00010000 | | | 10001001 | SUB R1,R2 | P-C 存入 R1 | |
| 00010001 | | | 01000010 | MOV R2,R0 | 将 R0 移入 R2，作为乘数准备 | |
| 00010010 | | | 01110001 | DEC R1 | 将另一个乘数减 1，做好准备 | |
| 00010011 | | | 11110000 | BZC RESULT | 判断 | |
| 00010100 | | | 00011011 |  |  | |
| 00010101 | | | 00001000 | LOOP:ADD R0,R2 | 开始乘法 | |
| 00010110 | | | 01110001 | DEC R1 |  | |
| 00010111 | | | 11110000 | BZC RESULT | 判断 | |
| 00011000 | | | 00011011 |  |  | |
| 00011001 | | | 11100000 | JMP LOOP | 回到 LOOP 开始 | |
| 00011010 | | | 00010101 |  |  | |
| 00011011 | | | 01001101 | MOV R1,R3 | 将 P 存回 R1 | |
| 00011100 | | | 01110001 | DEC R1 | 将另一个乘数减 1，做好准备 | |
| 00011101 | | | 11110000 | BZC RESULT | 判断 | |
| 00011110 | | | 00100110 |  |  | |
| 00011111 | | | 01000010 | MOV R2,R0 | 将 R0 移入 R2，作为乘数准备 | |
| 00100000 | | | 00001000 | LOOP:ADD R0,R2 | 开始乘法 | |
| 00100001 | | | 01110001 | DEC R1 |  | |
| 00100010 | | | 11110000 | BZC RESULT | 判断 | |
| 00100011 | | | 00100110 |  |  | |
| 00100100 | | | 11100000 | JMP LOOP | 回到 LOOP 开始 | |
| 00100101 | | | 00100000 |  |  | |
| 00100110 | | | 11000001 | LAD 00 R1,60H | 将 a 重新放入 R1 | |
| 00100111 | | | 01100000 |  |  | |
| 00101000 | | | 10000111 | SUB R3,R1 | R3=P-A | |
| 00101001 | | | 01001101 | MOV R1,R3 | R3->R1 | |
| 00101010 | | | 01000010 | MOV R2,R0 | R0->R2 | |
| 00101011 | | | 01110001 | DEC R1 |  | |
| 00101100 | | | 11110000 | BZC RESULT | 判断 | |
| 00101101 | | | 00110100 |  |  | |
| 00101110 | | 00001000 | | LOOP:ADD R0,R2 | 开始乘法 | |
| 00101111 | | 01110001 | | DEC R1 |  | |
| 00110000 | | 11110000 | | BZC RESULT | 判断 | |
| 00110001 | | 00110100 | |  |  | |
| 00110010 | | 11100000 | | JMP LOOP | 回到 LOOP 开始 | |
| 00110011 | 00101110 | | |  |  | |
| 00110100 | 11010000 | | | STA 61H,R0 | 将所得结果存入 61H | |
| 00110101 | 01100001 | | |  |  | |
| 00110110 | 00110000 | | | OUT 40H,R0 | 将所得结果输出 | |
| 00110111 | 01000000 | | |  |  | |

# 四、实验步骤

## 4.1微程序写入及校验

导入以下微程序

; //\*\* 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 063201 ; A减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 033201 ; A右移->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

经检验，程序写入成功。

## 4.2机器程序写入及校验

导入以下机器程序

$P 00 20 ;IN R0,00H

$P 01 00

$P 02 21 ;IN R1,00H

$P 03 00

$P 04 22 ;IN R2,00H

$P 05 00

$P 06 D0 ;STA 60H,R0

$P 07 60

$P 08 04 ;ADD R0,R1

$P 09 08 ;ADD R0,R2

$P 0A 63 ;LDI R3,01H

$P 0B 01

$P 0C AC ;SAR R0,R3

$P 0D 43 ;MOV R3,R0

$P 0E 84 ;SUB R0,R1

$P 0F 4D ;MOV R1,R3

$P 10 89 ;SUB R1,R2

$P 11 42 ;MOV R2,R0

$P 12 71 ;DEC R1

$P 13 F0 ;BZC RESULT

$P 14 1B

$P 15 08 ;LOOP:ADD R0,R2

$P 16 71 ;DEC R1

$P 17 F0 ;BZC RESULT

$P 18 1B

$P 19 E0 ;JMP LOOP

$P 1A 15

$P 1B 4D ;MOV R1,R3

$P 1C 71 ;DEC R1

$P 1D F0 ;BZC RESULT

$P 1E 26

$P 1F 42 ;MOV R2,R0

$P 20 08 ;LOOP:ADD R0,R2

$P 21 71 ;DEC R1

$P 22 F0 ;BZC RESULT

$P 23 26

$P 24 E0 ;JMP LOOP

$P 25 20

$P 26 C1 ;LAD 00 R1,60H

$P 27 60

$P 28 87 ;SUB R3,R1

$P 29 4D ;MOV R1,R3

$P 2A 42 ;MOV R2,R0

$P 2B 71 ;DEC R1

$P 2C F0 ;BZC RESULT

$P 2D 34

$P 2E 08 ;LOOP:ADD R0,R2

$P 2F 71 ;DEC R1

$P 30 F0 ;BZC RESULT

$P 31 34

$P 32 E0 ;JMP LOOP

$P 33 2E

$P 34 D0 ;STA 61H,R0

$P 35 61

$P 36 30 ;OUT 40H,R0

$P 37 40

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

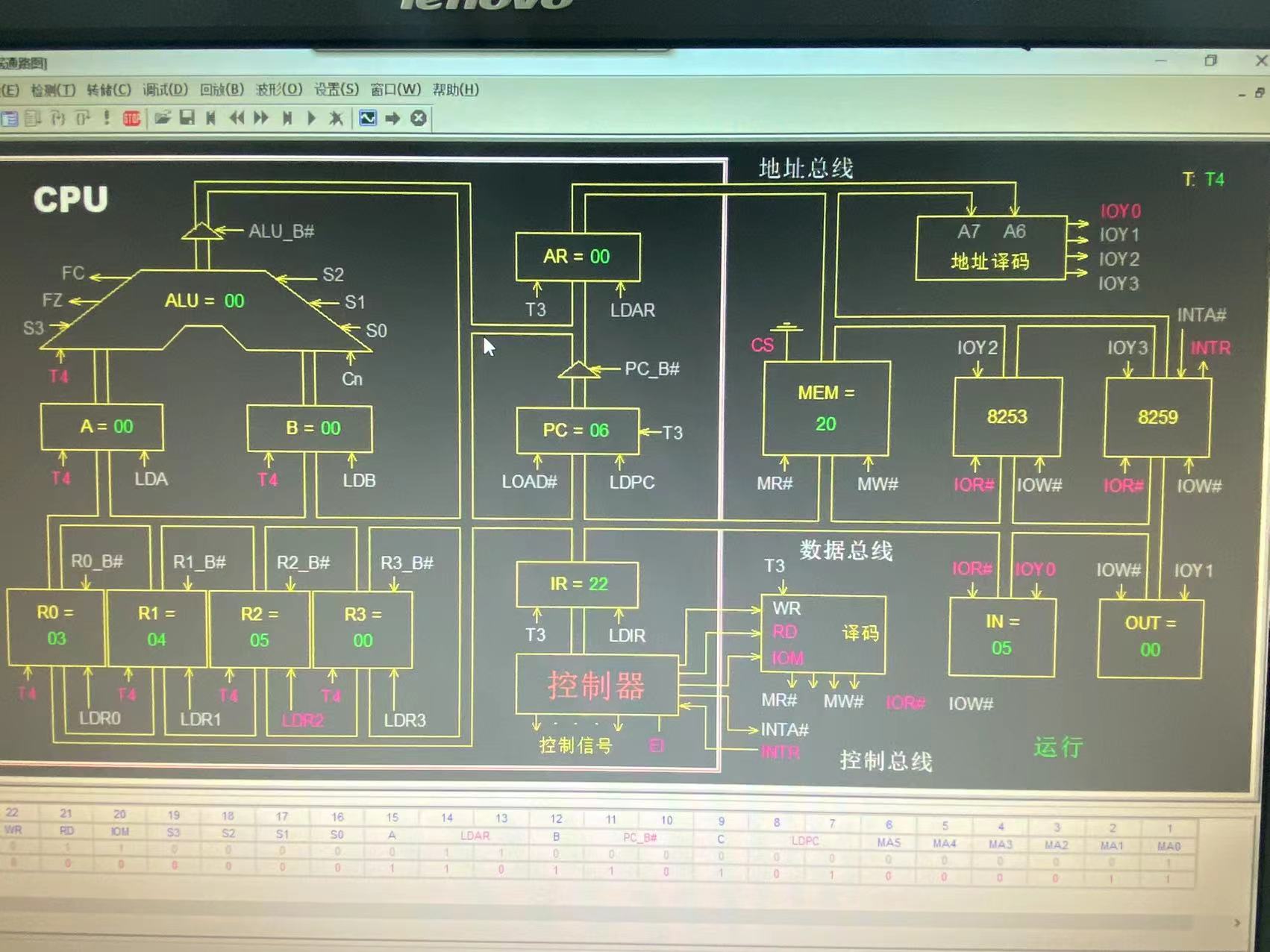
经检验，程序导入成功

# 五、实验结果及分析

## 5.1演示程序一

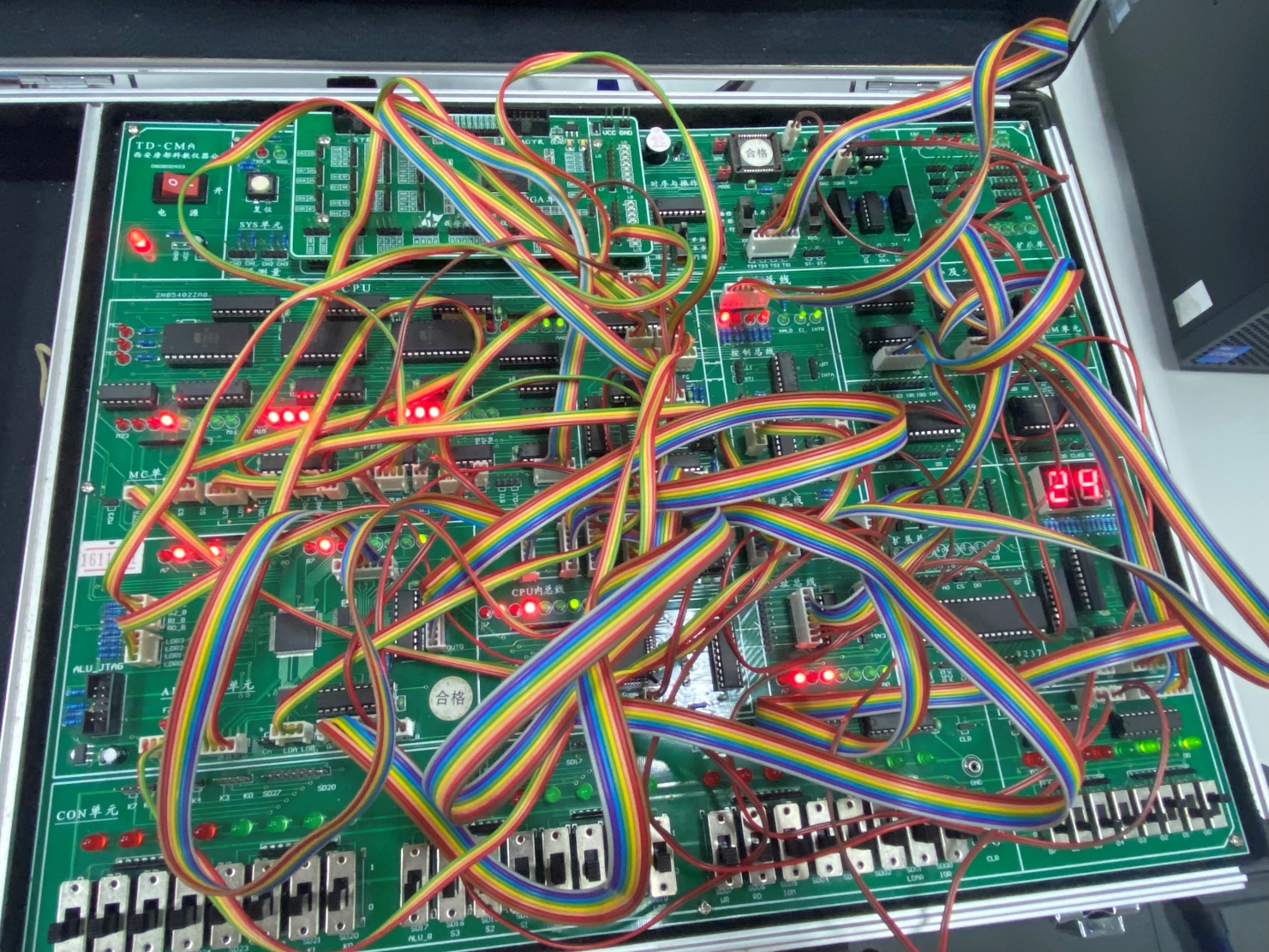
数据：

输入三角形三边3，4，5，公式计算得出三角形面积为6，预期输出结果为36，十六进制为24



结果：

最终输出结果36，即十六进制的24，结果正确



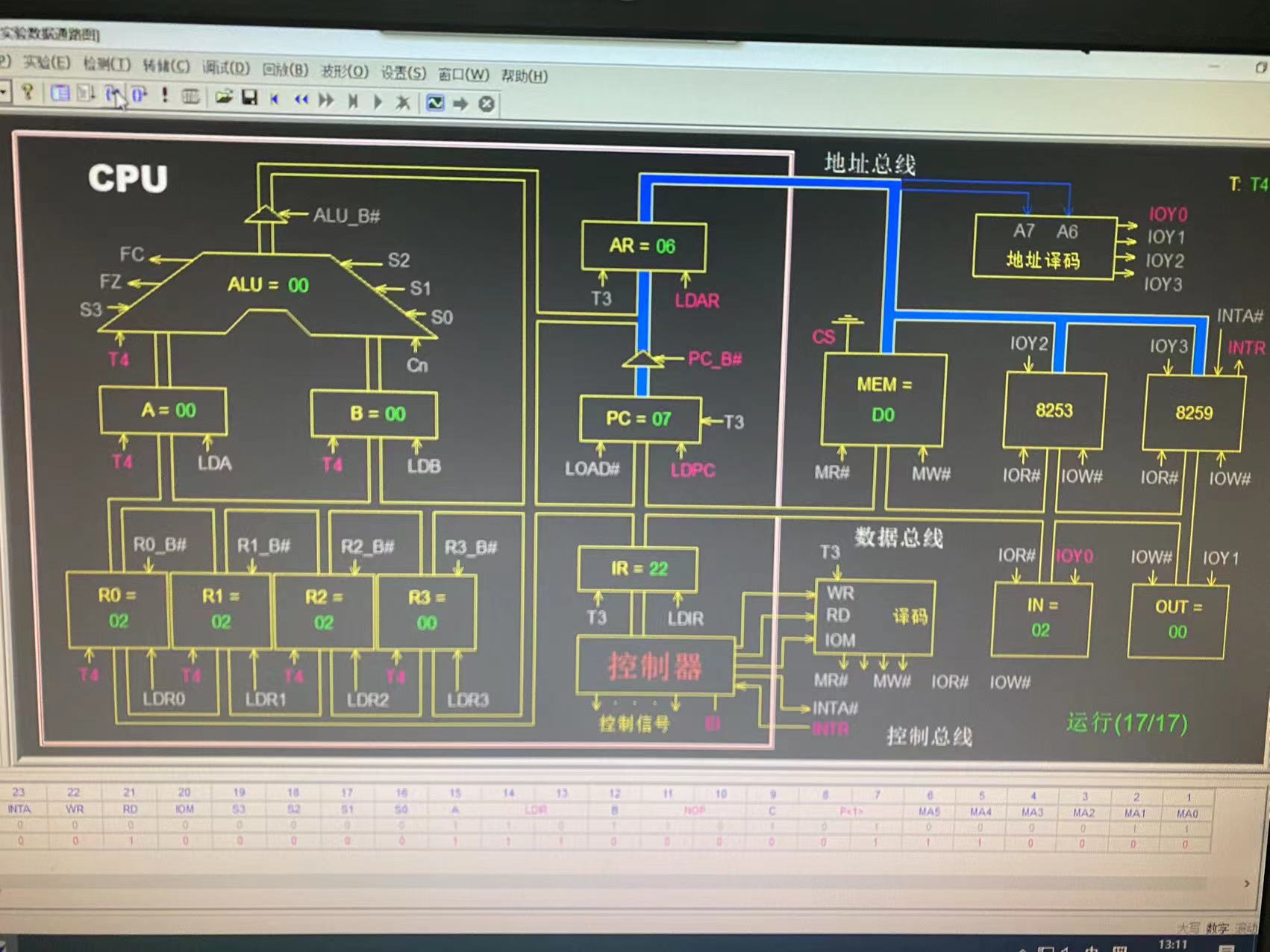
分析：

* + - 1. 逐个将3，4，5输入到寄存器R0，R1，R2中
      2. 计算p=(a+b+c)/2=6，放入R3
      3. 计算p-b=2，放入R0
      4. 计算p-c=1，放入R1
      5. 计算(p-b)\*(p-c)=2，放入R2
      6. 计算(p-b)\*(p-c)\*p=12，放入R0
      7. 计算p-a=3，放入R3
      8. 计算(p-b)\*(p-c)\*p\*(p-a)=36，存入61H中
      9. 输出

## 5.2演示程序二

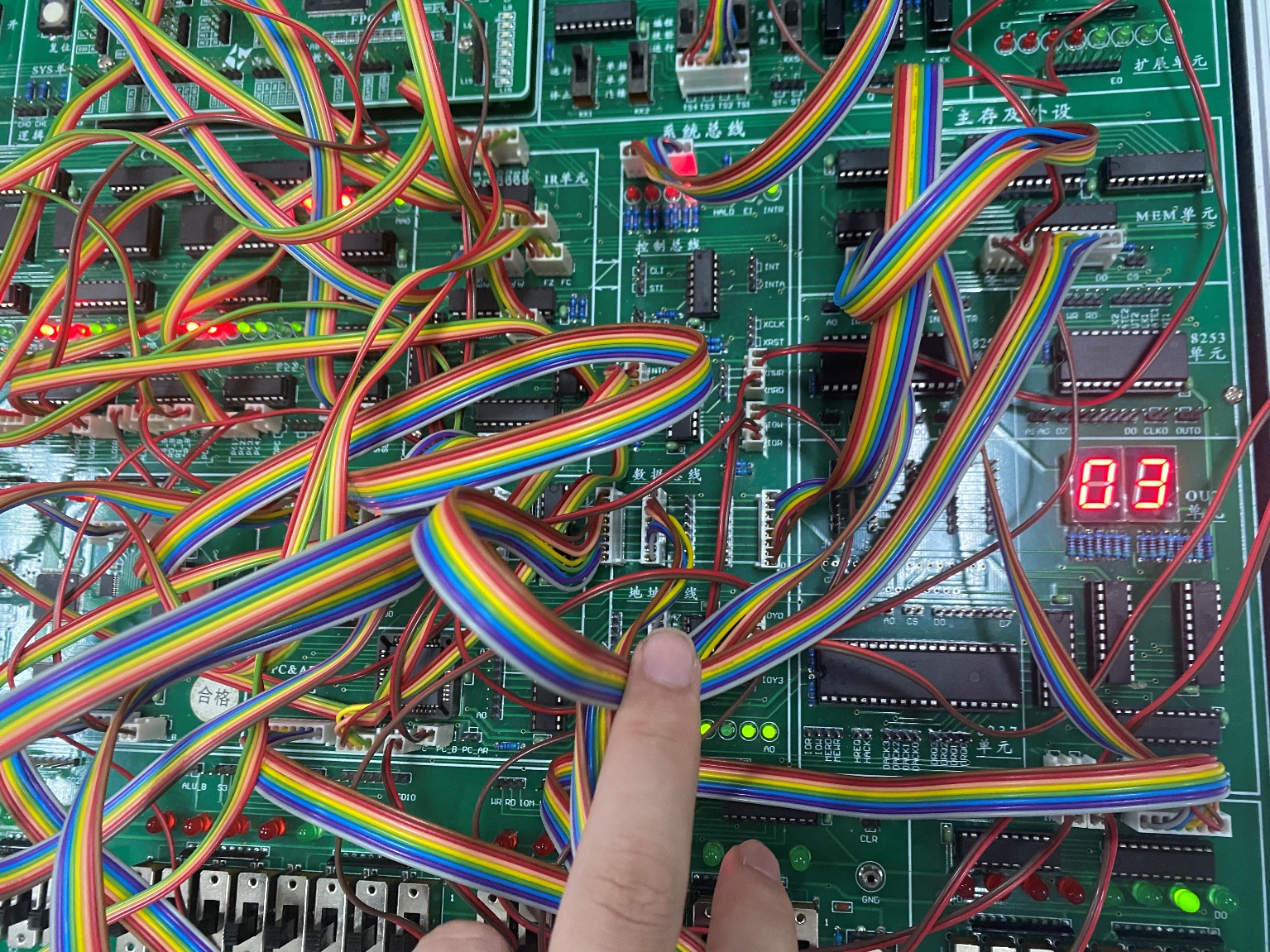
数据：

输出三角形三边2，2，2，公式计算得出三角形面积为，预期输出结果为3，十六进制为3



结果：

最终输出结果3，即十六进制的3，结果正确



分析：

* + - 1. 逐个将2，2，2输入到寄存器R0，R1，R2中
      2. 计算p=(a+b+c)/2=3，放入R3
      3. 计算p-b=1，放入R0
      4. 计算p-c=1，放入R1
      5. 计算(p-b)\*(p-c)=1，放入R2
      6. 计算(p-b)\*(p-c)\*p=3，放入R0
      7. 计算p-a=1，放入R3
      8. 计算(p-b)\*(p-c)\*p\*(p-a)=3，存入61H中
      9. 输出

# 六、实验问题及思考

1、当前所实现计算机，是否完整？如果不完整，还缺少哪些部件？

否，缺少开根号运算和对小数的运算

2、当前所实现计算机，是否能实现除法运算？如果能，可通过哪些指令实现除法运算？

能，通过ADD、SUB和逻辑左移的方法就可以实现除法运算

3、当前所实现计算机，还能实现哪些更复杂的计算？请举例说明。

实现计算等差数列、斐波那契数列、二分查找、冒泡排序等

4、当前所实现计算机，指令系统的双字长指令是如何实现的？

将指令存储在两个连续的单字长中

# 七、实验总结

## 7.1验收答辩问题回答

（1）当前并不能做到对奇数半周长的精确计算，在验收答辩之后，经过计算，通过令L=a+b+c,S=L\*(L-2\*a)\*(L-2\*b)\*(L-2\*c)/16，即S=(a+b+c)\*(a+b-c)\*(a+c-b)\*(b+c-a)/16的方法，的确可以一定程度上缩小计算奇数半周长的误差，并且可以避免奇数半周长除2以后可能与某条边相同的情况，导致无法计算出面积，如以2，2，3为边长的三角形。

（2）DEC机器指令跳转到RD->A的微指令，首先DEC机器指令的二进制编码为01110001，根据下表可知，I7I6I5I4=0111，MA5MA4=11，所以下一条微指令地址为110111，化为十六进制即37。

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 类型 | SE5 | SE4 | SE3 | SE2 | SE1 | SE0 |
| I7I6!=11 | MA5 | MA4 | I7 | I6 | I5 | I4 |
| I7I6==11 | MA5 | MA4 | 1 | 1 | I3 | I2 |

（3）答辩时未对24位微指令进行仔细学习和理解，只是粗浅的对其进行应用，在答辩后对其进行了仔细学习，现已可以将微指令具体翻译为二进制微代码，如RS->RD，将存放在RS寄存器中的数据放入RD中，首先高五位不需要用到，全部置零，S3~S0为0000表示运算结果为RS本身，A字段为011表示从寄存器中取出数值放入RD中，B字段为010表示从RS里取出数据，C字段为000表示不操作，即可实现RS->RD，最后六位则用于表示下一微指令地址，视具体情况而定。

（4）本程序的后续改进方向还可以通过中断功能来输入三角形的三边长

7.2实验心得体会

通过对本次课设的学习，让我对机器指令、微指令有了更深的理解，对他们的各个字段都仔细学习了一下，让我对汇编层的机器代码和运行都有了更深刻的了解，同时，还通过设计修改微指令，进一步提高了我的思考能力，实践能力。