# 计算机组成原理课程设计报告

**班级：计算机 班 姓名： 学号：**

**完成时间： 2019.1.3**

一、**课程设计目的**

**1．在实验机上设计实现机器指令及对应的微指令（微程序）并验证，从而进一步掌握微程序设计控制器的基本方法并了解指令系统与硬件结构的对应关系；**

**2．通过控制器的微程序设计，综合理解计算机组成原理课程的核心知识并进一步建立整机系统的概念；**

**3．培养综合实践及独立分析、解决问题的能力。**

**二、课程设计的任务**

**针对COP2000实验仪，从详细了解该模型机的指令/微指令系统入手，以实现乘法和除法运算功能为应用目标，在COP2000的集成开发环境下，设计全新的指令系统并编写对应的微程序；之后编写实现乘法和除法的程序进行设计的验证。**

**三、 课程设计使用的设备（环境）**

**1．硬件**

* **COP2000实验仪**
* **PC机**

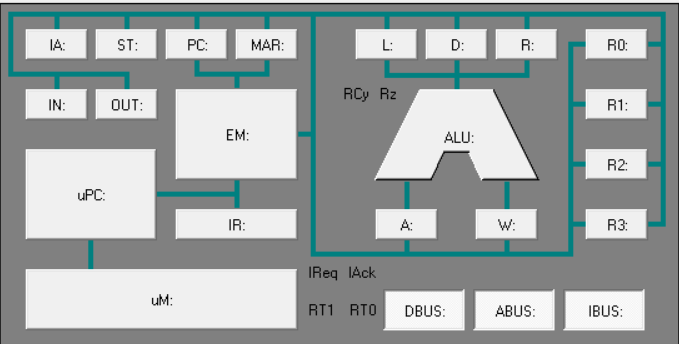
**2．软件**

* **COP2000仿真软件**

**四、课程设计的具体内容（步骤）**

**1．详细了解并掌握COP 2000模型机的微程序控制器原理，通过综合实验来实现**

**该模型机指令系统的特点：**



如图所示COP2000模型机包括中断向量寄存器IA，堆栈寄存器ST，程序计数器PC，地址寄存器MAR，输入输出端口IN、OUT，程序存储器EM，微程序计数器uPC，指令寄存器IR，微程序存储器uM，左移门L，直通门D，右移门R，运算器ALU，累加器A，工作寄存器W，寄存器组R0-R3，数据总线DBUS，地址总线ABUS，指令总线IBUS以及中断控制电路，跳转控制电路。其中运算器和中断控制电路以及跳转控制电路用CPLD实现，其他电路都是由离散的数字电路组成。

模型机为8位机，数据总线、地址总线都为8位，模型机的指令码为8位，根据指令类型的不同，可以有0-2个操作数。指令码的最低两位用来选择R0-R3寄存器，在微程序控制方式中，用指令码作为微地址来寻址微程序存储器，找到执行该指令的微程序。而在组合逻辑控制方式中，按时序用指令码产生相应的控制位。在本模型机中，一条指令最多分四个状态周期，一个状态周期为时钟脉冲，每个状态周期产生不同的控制逻辑，实现模型机的各种功能。模型机有24位控制位以控制寄存器的输入、输出，选择运算器的运算功能，存储器的读写。

该模型机的寻址方式共分为五种：

|  |  |  |  |
| --- | --- | --- | --- |
| 模型机的寻址方式 | 寻址方式说明 | 指令例子 | 对应的指令说明 |
| 累加器寻址 | 操作数为累加器A | CPL A | 将累加器A值取反 |
| 隐含寻址累加器A | OUT | 将累加器A的值输出到输出端口寄存器OUT |
| 寄存器寻址 | 参与运算的数据在R0-R3的寄存器 | ADD A，R0 | 将寄存器R0的值加上累加器A的值，再存入累加器A中 |
| 寄存器间接寻址 | 参与运算的数据在存储器EM中，数据的地址在寄存器R0-R3中 | MOV A，@R1 | 将寄存器R1的值做为地址，把存储器EM中该地址的内容送入累加器A中 |
| 存储器直接寻址 | 参与运算的数据在存储器EM中，数据的地址为指令的操作数 | AND A，40H | 将存储器EM中40H单元的数据与累加器A的值做逻辑与运算，结果存入累加器A |
| 立即数寻址 | 参与运算的数据为指令的操作数 | SUB A，#10H | 从累加器A中减去立即数10H，结果存入累加器A |

模型机的缺省的指令分为几大类：算术运算指令、逻辑运算指令、移位指令、数据传输指令、跳转指令、中断返回指令、输入/输出指令。

简单地罗列举例一些指令集：

|  |  |  |  |
| --- | --- | --- | --- |
| MOV A, R? | 011100xx |  | 将寄存器R?的值送到累加器A中 |
| MOV A, @R? | 011101xx |  | 将间址存储器的值送到累加器A中 |
| MOV A, MM | 011110xx | MM | 将存储器MM地址的值送到累加器A中 |
| MOV A, #II | 011111xx | II | 将立即数II送到累加器A中 |
| MOV R?, A | 100000xx |  | 将累加器A的值送到寄存器R?中 |
| MOV @R?, A | 100001xx |  | 将累加器A的值送到间址存储器中 |
| MOV MM, A | 100010xx | MM | 将累加器A的值送到存储器MM地址中 |
| MOV R?, #II | 100011xx | II | 将立即数II送到寄存器R?中 |

**该模型机微指令系统的特点（包括其微指令格式的说明等）：**

在微程序控制方式中，用指令码作为微地址来寻址微程序存储器，找到执行该指令的微程序。而在组合逻辑控制方式中，按时序用指令码产生相应的控制位。在本模型机中，一条指令最多分四个状态周期，一个状态周期为一个时钟脉冲，每个状态周期产生不同的控制逻辑，实现模型机的各种功能。模型机有24位控制位以控制寄存器的输入输出，选择运算器的运算功能，存储器的读写。即微指令是以直接表示法进行编码的，其特点是操作控制字段中的每一位代表一个微命令，0表示该位有效，选中了该操作，这种方法输出直接用于控制，但是由于微指令字较长，因而造成控制存储器容量较大。

24位控制位的详细介绍如下：

|  |  |
| --- | --- |
| 操作控制信号 | 控制信号的说明 |
| XRD | 外部设备读信号，当给出了外设的地址后，输出此信号，从指定外设读数据。 |
| EMWR | 程序存储器EM写信号 |
| EMRD | 程序存储器EM读信号 |
| PCOE | 将程序计数器PC的值送到地址总线ABUS上。 |
| EMEN | 将程序存储器EM与数据总线DBUS接通，由EMWR和EMRD决定是将DBUS数据写到EM中，还是从EM读出数据送到DBUS |
| IREN | 将程序存储器EM读出的数据打入指令寄存器IR和微指令计数器uPC |
| EINT | 中断返回时清除中断响应和中断请求标志，便于下次中断 |
| ELP | PC打入允许，与指令寄存器的IR3、IR2位结合，控制程序跳转 |
| MAREN | 将数据总线DBUS上数据打入地址寄存器MAR |
| MAROE | 将地址寄存器MAR的值送到地址总线ABUS上 |
| OUTEN | 将数据总线DBUS上数据送到输出端口寄存器OUT里 |
| STEN | 将数据总线DBUS上数据存入堆栈寄存器ST中 |
| RRD | 读寄存器组R0-R3，寄存器R?的选择由指令的最低两位决定 |
| RWR | 写寄存器组R0-R3，寄存器R?的选择由指令的最低两位决定 |
| CN | 决定运算器是否带进位移位，CN=1带进位，CN=0不带进位 |
| FEN | 将标志位存入ALU内部的标志寄存器 |
| X2、X1、X0 | X2、X1、X0三位组合来译码选择将数据送到DBUS上的寄存器 |
| WEN | 将数据总线DBUS的值打入工作寄存器W中 |
| AEN | 将数据总线DBUS的值打入累加器A中 |
| S2、S1、S0 | S2、S1、S0三位组合决定ALU做何种运算 |

COP2000模型机指令的最低两位（IR0和IR1）用来寻址R0-R3四个寄存器；IR2和IR3与ELP微控制信号，Cy和Z两个程序状态信号配合，控制PC的置数即程序的转移。各种转移的条件判断逻辑如下所示：

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ELP | IR3 | IR2 | Cy | Z | LDPC |
| 1 | X | X | X | X | 1 |
| 0 | 0 | 0 | 1 | X | 0 |
| 0 | 0 | 0 | 0 | X | 1 |
| 0 | 0 | 1 | X | 1 | 0 |
| 0 | 0 | 1 | X | 0 | 1 |
| 0 | 1 | X | X | X | 0 |

当ELP=1时，LDPC=1，不允许PC被预置

当ELP=0时，LDPC由IR3，IR2，Cy，Z确定

当IR3 IR2 = 1 X时，LDPC=0，PC被预置

当IR3 IR2 = 0 0时，LDPC=非Cy，当Cy=1时，PC被预置

当IR3 IR2 = 0 1时，LDPC=非Z，当Z=1时，PC被预置

COP2000中有7个寄存器可以向数据总线输出数据, 但在某一特定时刻只能有一个寄存器输出数据，由X2,X1,X0决定那一个寄存器输出数据。

|  |  |
| --- | --- |
| X2 X1 X0 | 输出寄存器 |
| 0 0 0 | IN\_OE 外部输入门 |
| 0 0 1 | IA\_OE 中断向量 |
| 0 1 0 | ST\_OE 堆栈寄存器 |
| 0 1 1 | PC\_OE PC寄存器 |
| 1 0 0 | D\_OE 直通门 |
| 1 0 1 | R\_OE 右移门 |
| 1 1 0 | L\_OE 左移门 |
| 1 1 1 | 没有输出 |

COP2000中的运算器由一片EPLD实现. 有8种运算, 通过S2,S1,S0来选择.

数据由寄存器A及寄存器W给出, 运算结果输出到直通门D。

|  |  |
| --- | --- |
| S2 S1 S0 | 功能 |
| 0 0 0 | A+W 加 |
| 0 0 1 | A-W 减 |
| 0 1 0 | A|W 或 |
| 0 1 1 | A&W 与 |
| 1 0 0 | A+W+C 带进位加 |
| 1 0 1 | A-W-C 带进位减 |
| 1 1 0 | ~A A取反 |
| 1 1 1 | A 输出A |

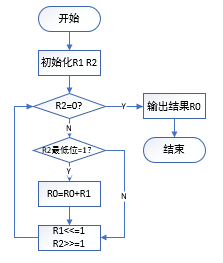
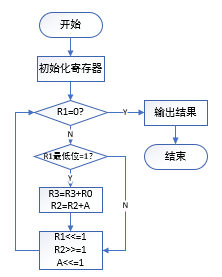
理解以上24个开关的作用是设计微指令的最重要的地方，输出只可以是一个设备，而输入可以是多个设备。

模型机的微指令有许多是可以并行的：比如下面两个模块，取指令和算术逻辑单元进行计算可以并行，这样一来，两个机器周期就可以压缩为一个机器周期。这么做的缺点是微程序可读性差。用好并行性在真正的实践中非常有用。

**2. 计算机中实现乘法和除法的原理**

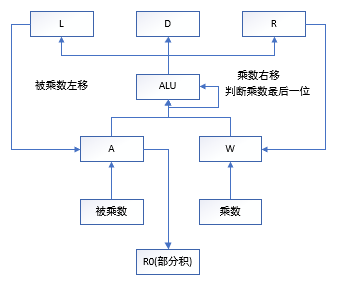
**（1）无符号乘法(已扩展为8位乘8位)**

**①算法流程图：**

4位乘4位无符号乘法流程图：

8位乘8位无符号数乘法流程图：

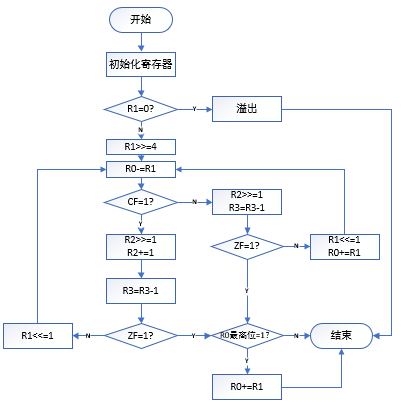
**②硬件原理框图：**

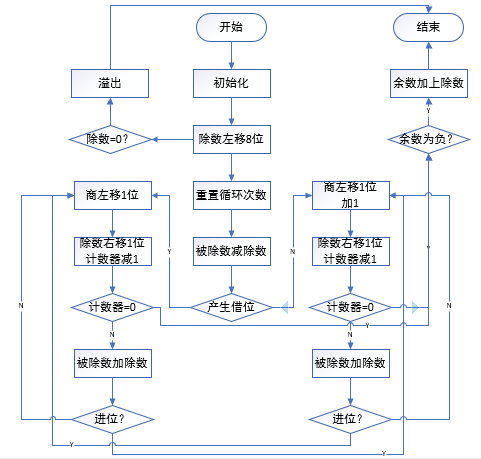
4位乘4位无符号乘法硬件原理框图：

**（2）无符号除法(已扩展为16位除8位)**

**①算法流程图：**

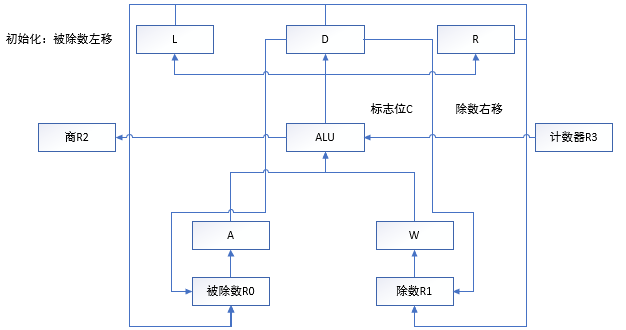
8位除4位无符号除法算法流程图：



16位除8位无符号除法算法流程图：

**②硬件原理框图：**

8位除4位无符号除法硬件原理框图：



**3．对应于以上算法如何分配使用COP2000实验仪中的硬件**

**（初步分配，设计完成后再将准确的使用情况填写在此处）**

1. 无符号乘法：

无符号乘法对应于COP2000实验仪的硬件具体分配使用情况如下表所示：

|  |  |
| --- | --- |
| 硬件名称 | 功能描述 |
| 寄存器R0 | 计算时用来存放部分积和最后的积 |
| 寄存器R1 | ① 初始化时，用来存放被乘数；  ② 在程序执行的过程中，用来存放向左移位后的被乘数。 |
| 寄存器R2 | ① 初始化时，用来存放乘数；  ② 在程序执行的过程中，用来存放向右移位后的乘数。 |
| 累加器A | 执行ADD A,R?（加法）、SHL R?（左移一位）、SHR R?（右移一位）等命令时所必须使用的寄存器。 |
| 寄存器W | 执行ADD A,R?（加法）、TEST R?,#II（测试R2的末位）等双操作数命令时所必须使用的寄存器。 |
| 左移门L | 用来实现相应数据左移一位的运算，并能够控制该运算后的结果是否输出到数据总线。 |
| 直通门D | 用来控制ALU的执行结果是否输出到数据总线。 |
| 右移门R | 用来实现相应数据右移一位的运算，并能够控制该运算后的结果是否输出到数据总线。 |
| 程序计数器PC | ① 控制程序按顺序正常执行；  ② 当执行转移指令时，从数据线接收要跳转的地址，使程序能够按需要自动执行。  ③ 当要从EM中读取数据时，由PC提供地址。 |
| 存储器EM | 存储指令和数据。 |
| 微程序计数器μPC | 向微程序存储器μM提供相应微指令的地址。 |
| 微程序存储器μM | 存储相应指令的微指令。 |
| 输出寄存器OUT | 可以将运算结果输出到输出寄存器OUT（本实验未用）。 |
| 堆栈ST | 当存储于累加器A的值将要受到破坏时，将其数据保存在堆栈ST中，使程序能够正常地执行。 |

1. 无符号除法：

无符号除法对应于COP2000实验仪的硬件具体分配使用情况如下表所示：

|  |  |
| --- | --- |
| 硬件名称 | 实现算法功能描述 |
| 寄存器R0 | 初始化时，用来存放被除数和计算后的余数。 |
| 寄存器R1 | ① 初始化时，用来存放除数；  ② 在程序执行的过程中，用来存放向右移位后的除数。 |
| 寄存器R2 | 在程序执行过程中，用来保存当前算得的商。 |
| 寄存器R3 | 当作计数器使用，用来控制程序是否结束（初始值5）。 |
| 累加器A | ① 计算时用来存放中间结果；  ② 执行ADD A,R?（加法）、SUB A,R?（减法）等命令时所必须使用的寄存器。 |
| 寄存器W | 执行SUB A,R?（减法）等双操作数命令时所必须使用的寄存器。 |
| 左移门L | 用来实现相应数据左移一位的运算，并能够控制该运算后的结果是否输出到数据总线。 |
| 直通门D | 用来控制ALU的执行结果是否输出到数据总线。 |
| 右移门R | 用来实现相应数据右移一位的运算，并能够控制该运算后的结果是否输出到数据总线。 |
| 程序计数器PC | ① 控制程序按顺序正常执行；  ② 当执行转移指令时，从数据线接收要跳转的地址，使程序能够按需要自动执行。  ③ 当要从EM中读取数据时，由PC提供地址。 |
| 存储器EM | 存储指令和数据。 |
| 微程序计数器μPC | 向微程序存储器μM提供相应微指令的地址。 |
| 微程序存储器μM | 存储相应指令的微指令。 |
| 输出寄存器OUT | 可以将运算结果输出到输出寄存器OUT（本实验未用）。 |
| 堆栈ST | 当存储于累加器A的值将要受到破坏时，将其数据保存在堆栈ST中，使程序能够正常地执行。 |

**4．在COP2000集成开发环境下设计全新的指令/微指令系统**

**设计结果如表所示（可按需要增删表项）**

1. **新的指令集**

**（如果针对乘除法设计了两个不同指令集要分别列表）**

**4位乘4位无符号乘法的指令：**

|  |  |  |  |
| --- | --- | --- | --- |
| **助记符** | **机器码1** | **机器码2** | **指令说明** |
| **\_FATCH\_** | **000000xx**  **00-03** |  | **实验机占用，不可修改。复位后，所有寄存器清0，首先执行 \_FATCH\_ 指令取指。** |
| **ADD R?,A** | **000001xx 04-07** |  | **将R寄存器中的值与A寄存器的值相加之后，将结果返回到R寄存器中** |
| **TEST R?,#II** | **000010xx 08-0B** | **II** | **将R寄存器中的结果与立即数II进行与运算，只更改CF,ZF,不更改R的值** |
| **MOV R?,#II** | **000011xx 0C-0F** | **II** | **将立即数送到寄存器R中** |
| **MOV A,R?** | **000100xx 10-13** |  | **将R寄存器的内容送到A寄存器中** |
| **SHL R?** | **000101xx 14-17** |  | **将R寄存器中的内容左移一位，之后送到R寄存器中** |
| **SHR R?** | **000110xx 18-1B** |  | **将R寄存器中的内容右移一位，之后送到R寄存器中** |
| **GOTO ,MM** | **000111xx 1C-1F** | **MM** | **直接跳转到地址为MM的指令** |
| **JC MM** | **001000xx 20-23** | **MM** | **如果CF=1，就跳转到地址为MM的指令** |
| **JZ MM** | **001001xx 24-27** | **MM** | **如果ZF=1，就跳转到地址为MM的指令** |
| **FUNEND** | **001010xx 28-2B** |  | **程序结束的标志** |

**新的微指令集：**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **助记符** | **状态** | **微地址** | **微程序** | **数据输出** | **数据打入** | **地址输出** | **运算器** | **移位控制** | **μPC** | **PC** |
| **\_FATCH** | **T0** | **00** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADD R? A** | **T2** | **04** | **FFF7EF** | **寄存器值R?** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T1** | **05** | **FFFA98** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **加运算** |  | **+1** |  |
|  | **T0** | **06** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **TEST R?,#II** | **T3** | **08** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  | **T2** | **09** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **0A** | **FFFE9B** | **ALU直通** | **标志位C,Z** |  | **与运算** |  | **+1** |  |
|  | **T0** | **0B** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV R?,#II** | **T1** | **0C** | **C7FBFF** | **存储器值EM** | **寄存器R?** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T0** | **0D** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV A,R?** | **T1** | **10** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T0** | **11** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SHL R?** | **T2** | **14** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **15** | **FFF9DF** | **ALU左移** | **寄存器R?** |  | **A输出** | **左移** | **+1** |  |
|  | **T0** | **16** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SHR R?** | **T2** | **18** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **19** | **FFF9BF** | **ALU右移** | **寄存器R?** |  | **A输出** | **右移** | **+1** |  |
|  | **T0** | **1A** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **GOTO MM** | **T1** | **1C** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **1D** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JC MM** | **T1** | **20** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **21** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JZ MM** | **T1** | **24** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **25** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **FUNEND** | **T0** | **28** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  |  |  |  |  |  |  |  |  |  |  |

**8位乘8位的无符号乘法的指令：**

|  |  |  |  |
| --- | --- | --- | --- |
| **助记符** | **机器码1** | **机器码2** | **指令说明** |
| **\_FATCH\_** | **000000xx**  **00-03** |  | **实验机占用，不可修改。复位后，所有寄存器清0，首先执行 \_FATCH\_ 指令取指。** |
| **ADD R?,A** | **000001xx 04-07** |  | **将R寄存器中的值与A寄存器的值相加之后，将结果返回到R寄存器中** |
| **ADD R?,#II** | **000010xx 08-0B** | **II** | **将R寄存器中的结果与立即数II进行加运算，将结果返回到R寄存器中** |
| **SUB R?,A** | **000011xx 0C-0F** |  | **将R寄存器中的值与A寄存器的值相减之后，将结果返回到R寄存器中** |
| **SUB R?,#II** | **000100xx 10-13** | **II** | **将R寄存器中的结果与立即数II进行减运算，将结果返回到R寄存器中** |
| **TEST R?,#II** | **000101xx 14-17** | **II** | **将R寄存器中的结果与立即数II进行与运算，只更改CF,ZF,不更改R的值** |
| **PUSH A** | **000110xx 18-1B** |  | **将A寄存器中的值放在堆栈寄存器ST中** |
| **POP A** | **000111xx 1C-1F** |  | **将堆栈寄存器ST中的值放在A寄存器中** |
| **MOV A,R?** | **001000xx 20-23** |  | **将R寄存器的内容送到A寄存器中** |
| **MOV R?,#II** | **001001xx 24-27** | **II** | **将立即数送到寄存器R中** |
| **SHL R?** | **001010xx 28-2B** |  | **将R寄存器中的内容左移一位，之后送到R寄存器中** |
| **SHR R?** | **001011xx 2C-2F** |  | **将R寄存器中的内容右移一位，之后送到R寄存器中** |
| **JC MM** | **001100xx 30-33** | **MM** | **如果CF=1，就跳转到地址为MM的指令** |
| **JZ MM** | **001101xx 34-37** | **MM** | **如果ZF=1，就跳转到地址为MM的指令** |
| **GOTO MM** | **001110xx 38-3B** | **MM** | **直接跳转到地址为MM的指令** |
| **FUNEND** | **001111xx 3C-3F** |  | **程序结束的标志** |
| **SAL A** | **010000xx 40-43** |  | **将A寄存器中的内容带进位左移一位，之后送到A寄存器中** |
| **ADC R?,A** | **010001xx 44-47** |  | **将R寄存器中的值与A寄存器的值带进位相加之后，将结果返回到R寄存器中** |
| **SAL R?** | **010010xx 48-4B** |  | **将R寄存器中的内容带进位左移一位，之后送到R寄存器中** |
|  |  |  |  |

**新的微指令集：**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **助记符** | **状态** | **微地址** | **微程序** | **数据输出** | **数据打入** | **地址输出** | **运算器** | **移位控制** | **μPC** | **PC** |
| **\_FATCH** | **T0** | **00** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADD R? A** | **T2** | **04** | **FFF7EF** | **寄存器值R?** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T1** | **05** | **FFFA98** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **加运算** |  | **+1** |  |
|  | **T0** | **06** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADD R? ,#II** | **T3** | **08** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T2** | **09** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  | **T1** | **0A** | **FFFA98** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **加运算** |  | **+1** |  |
|  | **T0** | **0B** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUB R?,A** | **T3** | **0C** | **FFFF8F** | **ALU直通** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T2** | **0D** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **0E** | **FFFA99** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **减运算** |  | **+1** |  |
|  | **T0** | **0F** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUB R? ,#II** | **T3** | **10** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T2** | **11** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  | **T1** | **12** | **FFFA99** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **减运算** |  | **+1** |  |
|  | **T0** | **0F** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **TEST R?,#II** | **T3** | **14** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  | **T2** | **15** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **16** | **FFFE9B** | **ALU直通** | **标志位C,Z** |  | **与运算** |  | **+1** |  |
|  | **T0** | **17** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **PUSH A** | **T1** | **18** | **FFEF9F** | **ALU直通** | **堆栈寄存器ST** |  | **A输出** |  | **+1** |  |
|  | **T0** | **19** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **POP A** | **T1** | **1C** | **FFFF57** | **堆栈寄存器ST** | **ALU直通** |  | **A输出** |  | **+1** |  |
|  | **T0** | **1D** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV A,R?** | **T1** | **20** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T0** | **21** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV R?,#II** | **T1** | **24** | **C7FBFF** | **存储器值EM** | **寄存器R?** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T0** | **25** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SHL R?** | **T2** | **28** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **29** | **FFF9DF** | **ALU左移** | **寄存器R?** |  | **A输出** | **左移** | **+1** |  |
|  | **T0** | **2A** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SHR R?** | **T2** | **2C** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **2D** | **FFF9BF** | **ALU右移** | **寄存器R?** |  | **A输出** | **右移** | **+1** |  |
|  | **T0** | **2E** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JC MM** | **T1** | **30** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **31** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JZ MM** | **T1** | **34** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **35** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **GOTO MM** | **T1** | **38** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **39** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **FUNEND** | **T0** | **3C** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SAL A** | **T1** | **40** | **FFFED7** | **ALU左移** | **寄存器A 标志位C,Z** |  | **A输出** | **带进位左移** | **+1** |  |
|  | **T0** | **41** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADC R? A** | **T3** | **44** | **FFFF8F** | **ALU直通** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T2** | **45** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **46** | **FFFB9C** | **ALU直通** | **寄存器R?** |  | **带进位加运算** |  | **+1** |  |
|  | **T0** | **47** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SAL R?** | **T2** | **48** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **40** | **FFFADF** | **ALU左移** | **寄存器R? 标志位C,Z** |  | **A输出** | **带进位左移** | **+1** |  |
|  | **T0** | **41** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |

**8位除4位无符号除法的指令：**

|  |  |  |  |
| --- | --- | --- | --- |
| **助记符** | **机器码1** | **机器码2** | **指令说明** |
| **\_FATCH\_** | **000000xx**  **00-03** |  | **实验机占用，不可修改。复位后，所有寄存器清0，首先执行 \_FATCH\_ 指令取指。** |
| **ADD R?,A** | **000001xx 04-07** |  | **将R寄存器中的值与A寄存器的值相加之后，将结果返回到R寄存器中** |
| **ADD R?,#II** | **000010xx 08-0B** | **II** | **将R寄存器中的结果与立即数II进行加运算，将结果返回到R寄存器中** |
| **SUB R?,A** | **000011xx 0C-0F** |  | **将R寄存器中的值与A寄存器的值相减之后，将结果返回到R寄存器中** |
| **SUB R?,#II** | **000100xx 10-13** | **II** | **将R寄存器中的结果与立即数II进行减运算，将结果返回到R寄存器中** |
| **TEST R?,#II** | **000101xx 14-17** | **II** | **将R寄存器中的结果与立即数II进行与运算，只更改CF,ZF,不更改R的值** |
| **PUSH A** | **000110xx 18-1B** |  | **将A寄存器中的值放在堆栈寄存器ST中** |
| **POP A** | **000111xx 1C-1F** |  | **将堆栈寄存器ST中的值放在A寄存器中** |
| **MOV A,R?** | **001000xx 20-23** |  | **将R寄存器的内容送到A寄存器中** |
| **MOV R?,#II** | **001001xx 24-27** | **II** | **将立即数送到寄存器R中** |
| **SHL R?** | **001010xx 28-2B** |  | **将R寄存器中的内容左移一位，之后送到R寄存器中** |
| **SHR R?** | **001011xx 2C-2F** |  | **将R寄存器中的内容右移一位，之后送到R寄存器中** |
| **JC MM** | **001100xx 30-33** | **MM** | **如果CF=1，就跳转到地址为MM的指令** |
| **JZ MM** | **001101xx 34-37** | **MM** | **如果ZF=1，就跳转到地址为MM的指令** |
| **GOTO MM** | **001110xx 38-3B** | **MM** | **直接跳转到地址为MM的指令** |
| **FUNEND** | **001111xx 3C-3F** |  | **程序结束的标志** |

**新的微指令集：**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **助记符** | **状态** | **微地址** | **微程序** | **数据输出** | **数据打入** | **地址输出** | **运算器** | **移位控制** | **μPC** | **PC** |
| **\_FATCH** | **T0** | **00** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADD R? A** | **T2** | **04** | **FFF7EF** | **寄存器值R?** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T1** | **05** | **FFFA98** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **加运算** |  | **+1** |  |
|  | **T0** | **06** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADD R? ,#II** | **T3** | **08** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T2** | **09** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  | **T1** | **0A** | **FFFA98** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **加运算** |  | **+1** |  |
|  | **T0** | **0B** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUB R?,A** | **T3** | **0C** | **FFFF8F** | **ALU直通** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T2** | **0D** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **0E** | **FFFA99** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **减运算** |  | **+1** |  |
|  | **T0** | **0F** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUB R? ,#II** | **T3** | **10** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T2** | **11** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  | **T1** | **12** | **FFFA99** | **ALU直通** | **寄存器R? 标志位C,Z** |  | **减运算** |  | **+1** |  |
|  | **T0** | **0F** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **TEST R?,#II** | **T3** | **14** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  | **T2** | **15** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **16** | **FFFE9B** | **ALU直通** | **标志位C,Z** |  | **与运算** |  | **+1** |  |
|  | **T0** | **17** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **PUSH A** | **T1** | **18** | **FFEF9F** | **ALU直通** | **堆栈寄存器ST** |  | **A输出** |  | **+1** |  |
|  | **T0** | **19** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **POP A** | **T1** | **1C** | **FFFF57** | **堆栈寄存器ST** | **ALU直通** |  | **A输出** |  | **+1** |  |
|  | **T0** | **1D** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV A,R?** | **T1** | **20** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T0** | **21** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV R?,#II** | **T1** | **24** | **C7FBFF** | **存储器值EM** | **寄存器R?** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T0** | **25** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SHL R?** | **T2** | **28** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **29** | **FFF9DF** | **ALU左移** | **寄存器R?** |  | **A输出** | **左移** | **+1** |  |
|  | **T0** | **2A** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SHR R?** | **T2** | **2C** | **FFF7F7** | **寄存器值R?** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **2D** | **FFF9BF** | **ALU右移** | **寄存器R?** |  | **A输出** | **右移** | **+1** |  |
|  | **T0** | **2E** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JC MM** | **T1** | **30** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **31** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JZ MM** | **T1** | **34** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **35** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **GOTO MM** | **T1** | **38** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **39** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **FUNEND** | **T0** | **3C** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
|  |  |  |  |  |  |  |  |  |  |  |

**16位除8位的无符号除法的指令：**

|  |  |  |  |
| --- | --- | --- | --- |
| **助记符** | **机器码1** | **机器码2** | **指令说明** |
| **\_FATCH\_** | **000000xx**  **00-03** |  | **实验机占用，不可修改。复位后，所有寄存器清0，首先执行 \_FATCH\_ 指令取指。** |
| **MOV**  **R?,#II** | **000001xx 04-07** | **II** | **将立即数II送入寄存器R中** |
| **TEST R?,#II** | **000010xx 08-0B** | **II** | **将R寄存器中的结果与立即数II进行与运算，只更改CF,ZF,不更改R的值** |
| **MOV A,R?** | **000011xx 0C-0F** |  | **将R寄存器的内容送入A寄存器中** |
| **JC MM** | **000100xx 10-13** | **MM** | **如果CF=1，就跳转到地址为MM的指令** |
| **JZ MM** | **000101xx 14-17** | **MM** | **如果ZF=1，就跳转到地址为MM的指令** |
| **JMP MM** | **000110xx 18-1B** | **MM** | **无条件跳转到地址为MM的指令** |
| **ADD**  **R?,A** | **000111xx 1C-1F** |  | **将R寄存器中的内容与A寄存器内容进行加运算，将结果放入R寄存器中** |
| **SHL R?** | **001000xx 20-23** |  | **将R寄存器中的内容左移一位，之后送到R寄存器中** |
| **FEND** | **001001xx 24-27** |  | **程序结束标志** |
| **SHR R?** | **001010xx 28-2B** |  | **将R寄存器中的内容右移一位，之后送到R寄存器中** |
| **PUSH A** | **001011xx 2C-2F** |  | **将A寄存器的内容送入堆栈寄存器ST中** |
| **POP A** | **001100xx 30-33** |  | **将堆栈寄存器ST中的内容送入A寄存器中** |
| **CMP**  **R?,A** | **001101xx 34-37** |  | **将R寄存器的内容与A寄存器的内容做减运算** |
| **ADD**  **R?,#II** | **001110xx 38-3B** | **II** | **将R寄存器的内容与立即数II做加运算，结果放入R寄存器中** |
| **SUB**  **R?,A** | **001111xx 3C-3F** |  | **将R寄存器的内容与寄存器A的内容做减运算，结果放入R寄存器中** |
| **MOV**  **R?,A** | **010000xx**  **40-43** |  | **将A寄存器的内容送入R寄存器中** |
| **SUB**  **R?,#II** | **010000xx**  **44-47** |  | **将R寄存器的内容与立即数II做减运算，结果放入R寄存器中** |
| **AND**  **R?,A** | **010010xx**  **48-4B** |  | **将R寄存器的内容与A寄存器的内容做与运算** |
| **RLC A** | **010011xx**  **4C-4F** |  | **将寄存器A的内容带进位循环左移一位，之后送入A寄存器中** |
| **ADC**  **R?,A** | **010100xx**  **50-53** |  | **将R寄存器的内容与寄存器A的内容做带进位加运算，结果放入R寄存器中** |
| **PUSH R?** | **010101xx**  **54-57** |  | **将R寄存器的内容送入堆栈寄存器ST中** |
| **RLC R?** | **010110xx**  **58-5B** |  | **将寄存器R的内容带进位循环左移一位，之后送入R寄存器中** |
| **MOV**  **A,#II** | **010111xx**  **5C-5F** | **II** | **将立即数II送入A寄存器中** |
| **MOV**  **MM,A** | **011000xx**  **60-63** |  | **将A寄存器的内容送入地址为MM的内存单元中** |
| **MOV**  **A,MM** | **011001xx**  **64-67** |  | **将MM的内存单元中内容送入A寄存器中** |
| **SUB**  **A,MM** | **011010xx**  **68-6B** |  | **将A寄存器中的内容与MM地址单元中的内容做减运算，结果放入A寄存器中** |
| **SUBC**  **A,MM** | **011011xx**  **6C-6F** |  | **将A寄存器中的内容与MM地址单元中的内容做带进位减运算，结果放入A寄存器中** |
| **RRC A** | **011100xx**  **70-73** |  | **将寄存器A的内容带进位循环右移一位，之后送入A寄存器中** |
| **SUBC**  **R?,A** | **011101xx**  **74-77** |  | **将R寄存器中的内容与A寄存器的内容做带进位减运算，结果放入R寄存器中** |

**新的微指令集：**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **助记符** | **状态** | **微地址** | **微程序** | **数据输出** | **数据打入** | **地址输出** | **运算器** | **移位控制** | **μPC** | **PC** |
| **\_FATCH** | **T0** | **00** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV**  **R?,#II** | **T1** | **04** | **C7FBFF** | **存储器值EM** | **寄存器R?** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T0** | **05** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **TEST**  **R?,#II** | **T3** | **08** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T2** | **09** | **FFF7F7** | **寄存器值R？** | **寄存器A** | **PC输出** | **A输出** |  | **+1** |  |
|  | **T1** | **0A** | **FFFE9B** | **ALU直通** | **标志位C,Z** | **PC输出** | **A输出** |  | **+1** |  |
|  | **T0** | **0B** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV**  **A,R?** | **T1** | **0C** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T0** | **0D** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JC MM** | **T1** | **10** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **11** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JZ MM** | **T1** | **14** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **15** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **JMP MM** | **T1** | **18** | **C6FFFF** | **存储器值EM** | **寄存器PC** | **PC输出** | **A输出** |  | **+1** | **写入** |
|  | **T0** | **19** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADD**  **R?,A** | **T2** | **1C** | **FFF7EF** | **寄存器值R？** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T1** | **1D** | **FFFA98** | **ALU直通** | **寄存器R?，标志位C,Z** |  | **加运算** |  | **+1** |  |
|  | **T0** | **1E** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SHL R?** | **T2** | **20** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **21** | **FFF9DF** | **ALU左移** | **寄存器R？** |  | **A输出** |  | **+1** |  |
|  | **T0** | **22** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** | **左移** | **写入** | **+1** |
| **FEND** | **T0** | **24** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SHR R?** | **T2** | **28** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **29** | **FFF9BF** | **ALU右移** | **寄存器R？** |  | **A输出** | **右移** | **+1** |  |
| **PUSH A** | **T1** | **2C** | **FFEF9F** | **ALU直通** | **堆栈寄存器ST** |  | **A输出** |  | **+1** |  |
|  | **T0** | **2D** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **POP A** | **T1** | **30** | **FFFF57** | **堆栈寄存器ST** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T0** | **31** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **CMP**  **R?,A** | **T3** | **34** | **FFFF8F** | **ALU直通** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T2** | **35** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **36** | **FFFE99** | **ALU直通** | **标志位C,Z** |  | **减运算** |  | **+1** |  |
|  | **T0** | **37** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADD**  **R?,#II** | **T3** | **38** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T2** | **39** | **C7FFEF** | **存储器值EM** | **寄存器W** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T1** | **3A** | **FFFA98** | **ALU直通** | **寄存器R？，标志位C,Z** |  | **加运算** |  | **+1** |  |
|  | **T0** | **3B** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUB**  **R?,A** | **T3** | **3C** | **FFFF8F** | **ALU直通** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T2** | **3D** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **3E** | **FFFA99** | **ALU直通** | **寄存器R？，标志位C,Z** |  | **减运算** |  | **+1** |  |
|  | **T0** | **3F** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV**  **R?,A** | **T1** | **40** | **FFFB9F** | **ALU直通** | **寄存器R？** |  | **A输出** |  | **+1** |  |
|  | **T0** | **41** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUB**  **R?,#II** | **T3** | **44** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T2** | **45** | **C7FFEF** | **存储器值EM** | **寄存器A** |  | **A输出** |  | **+1** | **+1** |
|  | **T1** | **46** | **FFFA99** | **ALU直通** | **寄存器R？，标志位C,Z** |  | **减运算** |  | **+1** |  |
|  | **T0** | **47** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **AND**  **R?,A** | **T2** | **48** | **FFF7EF** | **寄存器值R？** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T1** | **49** | **FFFE93** | **ALU直通** | **寄存器A，标志位C,Z** |  | **与运算** |  | **+1** |  |
|  | **T0** | **4A** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **RLC A** | **T1** | **4C** | **FFFED7** | **ALU左移** | **寄存器A，标志位C,Z** |  | **A输出** | **带进位左移** | **+1** |  |
|  | **T0** | **4D** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **ADC**  **R?,A** | **T2** | **50** | **FFF7EF** | **寄存器值R？** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T1** | **51** | **FFFA9C** | **ALU直通** | **寄存器R？，标志位C,Z** |  | **带进位加运算** |  | **+1** |  |
|  | **T0** | **52** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **PUSH R?** | **T1** | **54** | **FFE7FF** | **寄存器值R？** | **堆栈寄存器ST** |  | **A输出** |  | **+1** |  |
|  | **T0** | **55** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **RLC R?** | **T2** | **58** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **59** | **FFFADF** | **ALU左移** | **寄存器R？，标志位C,Z** |  | **A输出** | **带进位左移** | **+1** |  |
|  | **T0** | **5A** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV**  **A,#II** | **T1** | **5C** | **C7FFF7** | **存储器值EM** | **寄存器A** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T0** | **5D** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV**  **MM,A** | **T2** | **60** | **C77EFF** | **存储器值EM** | **地址寄存器MAR** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T1** | **61** | **B7BF9F** | **ALU直通** | **存储器EM** | **MAR输出** | **A输出** |  | **+1** |  |
|  | **T0** | **62** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **MOV**  **A,MM** | **T2** | **64** | **C77FFF** | **存储器值EM** | **地址寄存器MAR** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T1** | **65** | **D7BFF7** | **存储器值EM** | **寄存器A** | **MAR输出** | **A输出** |  | **+1** |  |
|  | **T0** | **66** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUB**  **A,MM** | **T3** | **68** | **C77FFF** | **存储器值EM** | **地址寄存器MAR** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T2** | **69** | **D7BFEF** | **存储器值EM** | **寄存器W** | **MAR输出** | **A输出** |  | **+1** |  |
|  | **T1** | **6A** | **FFFE91** | **ALU直通** | **寄存器A，标志位C,Z** |  | **减运算** |  | **+1** |  |
|  | **T0** | **6B** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUBC**  **A,MM** | **T3** | **6C** | **C77FFF** | **存储器值EM** | **地址寄存器MAR** | **PC输出** | **A输出** |  | **+1** | **+1** |
|  | **T2** | **6D** | **D7BFEF** | **存储器值EM** | **寄存器W** | **MAR输出** | **A输出** |  | **+1** |  |
|  | **T1** | **6E** | **FFFE95** | **ALU直通** | **寄存器A，标志位C,Z** |  | **带进位减运算** |  | **+1** |  |
|  | **T0** | **6F** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **RRC A** | **T1** | **70** | **FFFEB7** | **ALU右移** | **寄存器A，标志位C,Z** |  | **A输出** | **带进位右移** | **+1** |  |
|  | **T0** | **71** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |
| **SUBC**  **R?,A** | **T3** | **74** | **FFFF8F** | **ALU直通** | **寄存器W** |  | **A输出** |  | **+1** |  |
|  | **T2** | **75** | **FFF7F7** | **寄存器值R？** | **寄存器A** |  | **A输出** |  | **+1** |  |
|  | **T1** | **76** | **FFFA9D** | **ALU直通** | **寄存器R？，标志位C,Z** |  | **带进位减运算** |  | **+1** |  |
|  | **T0** | **77** | **CBFFFF** |  | **指令寄存器IR** | **PC输出** | **A输出** |  | **写入** | **+1** |

**5．用设计完成的新指令集编写实现无符号二进制乘法、除法功能的汇编语言程序**

**（1）乘法**

**4位乘法的算法流程图与汇编语言程序清单：**

**4位乘4位无符号乘法程序：**

MOV R0, #00H ;结果

MOV R1, #09H ;被乘数

MOV R2, #06H ;乘数

L1: TEST R2,#0FFH ;R2=0时结束

JZ LAST

TEST R2,#01H ;判断乘数最低位是否为1

JZ L2

MOV A,R1 ;相加

ADD R0,A

L2: SHL R1 ;被乘数左移

SHR R2 ;乘数右移

GOTO L1

LAST: FEND

GOTO LAST

**8位乘8位无符号乘法程序：**

MOV R0,#08AH ;R0被乘数

MOV R1,#16H ;R1乘数

MOV R2,#00H ;R2结果高8位

MOV R3,#00H ;R3结果低8位

MOV A,R3

PUSH A

MULT: TEST R1,#0FFH ;R1=0时结束

JZ LAST

TEST R1,#01H ;判断乘数最低位是否为1

JZ MULT2

MOV A,R0

ADD R3,A ;相加

POP A

ADC R2,A

MULT2:SHR R1

SAL R0

POP A ;A暂存被乘数高位

RCL A ;A带进位左移

PUSH A

GOTO MULT

LAST: FEND

GOTO LAST

**（2）除法**

**4位除法的算法流程图与汇编语言程序清单：**

**8位除4位无符号除法程序：**

MOV R0,#52H ;被除数

MOV R1,#07H ;除数

MOV R2,#00H ;商

MOV R3,#05H ;计数寄存器

TEST R1,#0FFH ;判断除数是否为0

JZ OVERFLOW ;除数为0时溢出

MOV A,R1

PUSH A ;保存除数

SHL R1

SHL R1

SHL R1

SHL R1 ;除数左移四位

MOV A,R1

SUB R0,A ;两数相减

JC ZERO ;CF=1表明商0

GOTO ONE ;商1

ZERO:SHL R2 ;商左移1位

SHR R1 ;除数右移1位

SUB R3,#01H;计数器减1

JZ REMAIN ;计数器为0

MOV A,R1

ADD R0,A ;相加

JC ONE ;CF=1商1

GOTO ZERO ;商0

ONE:SHL R2

ADD R2,#01H ;商左移1位，加1

SHR R1 ;除数右移1位

SUB R3,#01H ;计数器减1

JZ REMAIN ;计数器为0

MOV A,R1

SUB R0,A ;相减

JC ZERO ;CF=1商0

GOTO ONE ;商1

REMAIN:

TEST R0,#80H;检查R0最高位是否为1

JZ LAST

POP A

ADD R0,A ;加上原除数得到余数

GOTO LAST

OVERFLOW:

MOV R2,#0FFH

GOTO LAST

LAST:

FEND

GOTO LAST

**16位除8位无符号除法程序：**

MOV R0,#00H ;被除数高8位

MOV R1,#52H ;被除数低8位

MOV R2,#07H ;除数

MOV R3,#08H ;除数左移8位

MOV A,#00H ;立即数00H

MOV 91H,A

MOV 93H,A

MOV 95H,A ;使用内存空间91H,93H,95H

START: TEST R3,#0FH ;除数左移8位

JZ DIVI

ADD R2,#0

RLC R2

MOV A,91H ;91H除数高8位

RLC A

MOV 91H,A

SUB R3,#1

JMP START

DIVI: MOV A,R2 ;93H除数低8位

MOV 93H,A

MOV R2,#9 ;循环次数送入R2

MOV R3,#0

MOV A,#0

MOV 95H,A

DIVII: TEST R2,#0FH ;循环次数为0时结束

JZ STOP

MOV A,93H

SUB R1,A ;16位减法

MOV A,91H

SUBC R0,A

JC ZERO ;商0

ONE: ADD R3,#1 ;商1

SUB R2,#1 ;循环次数减1

JZ STOP

RLC R3 ;商左移

MOV A,95H ;R3商的低8位

RLC A

MOV 95H,A ;95H商的高8位

MOV A,91H ;除数右移

RRC A

MOV 91H,A

MOV A,93H

RRC A

MOV 93H,A

JMP DIVII

ZERO: SUB R2,#1 ;商0

JZ STOP

RLC R3

MOV A,95H ;商左移

RLC A

MOV 95H,A

MOV A,91H ;除数右移

RRC A

MOV 91H,A

MOV A,93H

RRC A

MOV 93H,A

MOV A,93H

ADD R1,A ;执行加法

MOV A,91H

ADC R0,A

JC ONE ;如果C=1，说明余数变正，商1，下次执行减法

JMP ZERO

STOP: TEST R0,#80H ;余数为负，进行负数处理，加上除数

JZ STOP1

MOV A,93H

ADD R1,A

MOV A,#0

ADC R0,A

STOP1: MOV A,95H

MOV R2,A

LAST: FEND

JMP LAST

**6．上述程序的运行情况（跟踪结果）**

**按下表填写描述以上各程序运行情况的内容。按每个程序一张表进行。**

**4位乘4位无符号乘法程序运行的过程：**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **汇 编 指 令** | **程序地址** | **机器码** | **指令说明** | **微程序** | **μPC** | **PC** | **运行时寄存器或存储器的值** |
| **\_FATCH\_** | **00** | **00** | **实验机占用，不可修改。复位后，所有寄存器清0，首先执行 \_FATCH\_ 指令取指。** | **CBFFFF** | **+1** | **写入** | **EM:0C** |
| **MOV R0,#00** | **00** | **0C00** | **将立即数00H送到寄存器R0中** | **C7FBFF**  **CBFFFF** | **+1**  **写入** | **+1**  **+1** | **EM:00 R0:00** |
| **MOV R1,#09** | **02** | **0C09** | **将立即数09H送到寄存器R1中** | **C7FBFF**  **CBFFFF** | **+1**  **写入** | **+1**  **+1** | **EM:0D**  **EM:09 R1:09** |
| **MOV R2,#06** | **04** | **0C06** | **将立即数06H送到寄存器R2中** | **C7FBFF**  **CBFFFF** | **+1**  **写入** | **+1**  **+1** | **EM:0E**  **EM:06 R2:06** |
| **TEST R2,#FF** | **06** | **08FF** | **将R2寄存器中的内容与0FFH进行与运算，更改C,Z标志位** | **C7FFEF**  **FFF7E7**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **EM:FF W:FF**  **A:06** |
| **JZ 14** | **08** | **2414** | **如果ZF=1，就跳转到地址码为14的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:24**  **EM:14** |
| **TEST R2,#01** | **0A** | **0801** | **将R2寄存器中的内容与01H进行与运算，更改C,Z标志位** | **C7FFEF**  **FFF7E7**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **W:01** |
| **JZ 10** | **0C** | **2410** | **如果ZF=1，就跳转到地址码为10的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:24**  **EM:10** |
| **SHL R1** | **10** | **14** | **将R1寄存器中的内容左移一位** | **FFF7E7**  **FFF9DF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:15**  **R1:12** |
| **SHR R2** | **11** | **18** | **将R2寄存器中的内容右移一位** | **FFF7E7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:1A**  **R2:03** |
| **GOTO 06** | **12** | **1C06** | **跳转到地址码为06的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM;1C**  **EM:06** |
| **TEST R2,#FF** | **06** | **08FF** | **将R2寄存器中的内容与0FFH进行与运算，更改C,Z标志位** | **C7FFEF**  **FFF7E7**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **EM:FF W:FF**  **A:03** |
| **JZ 14** | **08** | **2414** | **如果ZF=1，就跳转到地址码为14的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:24**  **EM:14** |
| **TEST R2,#01** | **0A** | **0801** | **将R2寄存器中的内容与01H进行与运算，更改C,Z标志位** | **C7FFEF**  **FFF7E7**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **W:01** |
| **JZ 10** | **0C** | **2410** | **如果ZF=1，就跳转到地址码为10的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:24**  **EM:10** |
| **MOV A,R1** | **0E** | **10** | **将R1寄存器中的内容送到A寄存器中** | **FFF7E7**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:11**  **A:12** |
| **ADD R0,A** | **0F** | **04** | **将R0寄存器的内容与A寄存器中的内容相加，之后将结果返回到R0寄存器中** | **FFF7EF**  **FFFA98**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:04**  **W;00**  **R0:12** |
| **SHL R1** | **10** | **14** | **将R1寄存器中的内容左移一位** | **FFF7E7**  **FFF9DF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:15**  **R1:24** |
| **SHR R2** | **11** | **18** | **将R2寄存器中的内容右移一位** | **FFF7E7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:1A**  **R2:01** |
| **GOTO 06** | **12** | **1C06** | **跳转到地址码为06的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM;1C**  **EM:06** |
| **TEST R2,#FF** | **06** | **08FF** | **将R2寄存器中的内容与0FFH进行与运算，更改C,Z标志位** | **C7FFEF**  **FFF7E7**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **EM:FF W:FF**  **A:01** |
| **JZ 14** | **08** | **2414** | **如果ZF=1，就跳转到地址码为14的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:24**  **EM:14** |
| **TEST R2,#01** | **0A** | **0801** | **将R2寄存器中的内容与01H进行与运算，更改C,Z标志位** | **C7FFEF**  **FFF7E7**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **W:01** |
| **JZ 10** | **0C** | **2410** | **如果ZF=1，就跳转到地址码为10的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:24**  **EM:10** |
| **MOV A,R1** | **0E** | **10** | **将R1寄存器中的内容送到A寄存器中** | **FFF7E7**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:11**  **A:24** |
| **ADD R0,A** | **0F** | **04** | **将R0寄存器的内容与A寄存器中的内容相加，之后将结果返回到R0寄存器中** | **FFF7EF**  **FFFA98**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:04**  **W;00**  **R0:36** |
| **SHL R1** | **10** | **14** | **将R1寄存器中的内容左移一位** | **FFF7E7**  **FFF9DF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:15**  **R1:48** |
| **SHR R2** | **11** | **18** | **将R2寄存器中的内容右移一位** | **FFF7E7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:1A**  **R2:00** |
| **GOTO 06** | **12** | **1C06** | **跳转到地址码为06的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM;1C**  **EM:06** |
| **TEST R2,#FF** | **06** | **08FF** | **将R2寄存器中的内容与0FFH进行与运算，更改C,Z标志位** | **C7FFEF**  **FFF7E7**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **EM:FF W:FF**  **A:00** |
| **JZ 14** | **08** | **2414** | **如果ZF=1，就跳转到地址码为14的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:24**  **EM:14** |
| **FEND** | **14** | **28** | **代表程序结束** | **CBFFFF** | **写入** | **+1** | **EM:28** |
|  |  |  |  |  |  |  |  |

**8位除4位无符号除法程序运行的过程：**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **汇 编 指 令** | **程序地址** | **机器码** | **指令说明** | **微程序** | **μPC** | **PC** | **运行时寄存器或存储器的值** |
| **\_FATCH\_** | **00** | **00** | **实验机占用，不可修改。复位后，所有寄存器清0，首先执行 \_FATCH\_ 指令取指。** | **CBFFFF** | **写入** | **+1** | **EM:24** |
| **MOV R0,#64** | **00** | **24** | **将立即数64H送到寄存器R0中** | **C7FBFF**  **CBFFFF** | **+1**  **写入** | **+1**  **+1** | **EM:64 R0:64** |
| **MOV R1,#09** | **02** | **24** | **将立即数09H送到寄存器R1中** | **C7FBFF**  **CBFFFF** | **+1**  **写入** | **+1**  **+1** | **EM:25**  **EM:09 R1:09** |
| **MOV R2,#00** | **04** | **24** | **将立即数00H送到寄存器R2中** | **C7FBFF**  **CBFFFF** | **+1**  **写入** | **+1**  **+1** | **EM:26**  **EM:00 R2:00** |
| **MOV R3,#05** | **06** | **24** | **将立即数05H送到寄存器R3中** | **C7FBFF**  **CBFFFF** | **+1**  **写入** | **+1**  **+1** | **EM:27**  **EM:05 R3:05** |
| **TEST R1,#FF** | **08** | **14FF** | **将R1寄存器中的内容与0FFH进行与运算，更改C,Z标志位** | **FFF7F7**  **C7FFEF**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:15**  **A:09**  **W:FF** |
| **JZ 3A** | **0A** | **343A** | **如果ZF=1，就跳转到地址码为3A的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:34**  **EM:3A** |
| **MOV A,R1** | **0C** | **20** | **将R1寄存器中的内容送到A寄存器中** | **FFF7F7**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:21**  **A:24** |
| **PUSH A** | **0D** | **18** | **将A寄存器中的内容送到堆栈寄存器ST中** | **FFEF9F**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:18**  **ST:09** |
| **SHL R1** | **0E** | **28** | **将R1寄存器中的内容左移一位** | **FFF7F7**  **FFF9DF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:29**  **R1:12** |
| **SHL R1** | **0F** | **28** | **将R1寄存器中的内容左移一位** | **FFF7F7**  **FFF9DF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:29**  **R1:24** |
| **SHL R1** | **10** | **28** | **将R1寄存器中的内容左移一位** | **FFF7F7**  **FFF9DF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:29**  **R1:48** |
| **SHL R1** | **11** | **28** | **将R1寄存器中的内容左移一位** | **FFF7F7**  **FFF9DF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:29**  **R1:90** |
| **MOV A,R1** | **12** | **20** | **将R1寄存器中的内容送到A寄存器中** | **FFF7F7**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:21**  **A:90** |
| **SUB R0,A** | **13** | **0C** | **将R0寄存器中的值减去A寄存器中的值后，返回到R0寄存器中** | **FFFF8F**  **FFF7F7**  **FFFA99**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1** | **EM:0C**  **W:90**  **A:64**  **R0:D4** |
| **JC 18** | **14** | **30 18** | **如果CF=1，就跳转到地址码为18的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:30**  **EM:18** |
| **SHL R2** | **18** | **28** | **将R2寄存器中的内容右移一位** | **FFF7E7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2A**  **A:00**  **R2:00** |
| **SHR R1** | **19** | **2C** | **将R1寄存器中的内容右移一位** | **FFF7F7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2D**  **A:90**  **R1:48** |
| **SUB R3,#01** | **1A** | **10 01** | **将R3寄存器的值减一，之后将结果返回到R3寄存器** | **FFF7F7**  **C7FFEF**  **FFFA99**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:13**  **A:05**  **EM:01 W:01**  **R3:04** |
| **JZ 32** | **1C** | **34 32** | **如果ZF=1，就跳转到地址码为32的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:34**  **EM:32** |
| **MOV A,R1** | **1E** | **20** | **将R1寄存器中的内容送到A寄存器中** | **FFF7F7**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:21**  **A:48** |
| **ADD R0,A** | **1F** | **04** | **将寄存器R0的值跟A寄存器的值相加之后返回到R0中** | **FFF7EF**  **FFFA98**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:04**  **W:D4**  **R0:1C** |
| **JC 24** | **20** | **30 24** | **如果CF=1，就跳转到地址码为24的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:30**  **EM:24** |
| **SHL R2** | **24** | **28** | **将R2寄存器中的内容右移一位** | **FFF7F7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2A**  **A:00**  **R2:00** |
| **ADD R2,#01** | **25** | **08 01** | **将R2寄存器的值加一之后返回到R2寄存器中** | **FFF7F7**  **C7FFEF**  **FFFA98**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **W:01**  **R2:01** |
| **SHR R1** | **27** | **2C** | **将R1寄存器中的内容右移一位** | **FFF7F7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2D**  **A:48**  **R1:24** |
| **SUB R3,#01** | **28** | **10 01** | **将R3寄存器的值减一，之后将结果返回到R3寄存器** | **FFF7F7**  **C7FFEF**  **FFFA99**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:13**  **A:04**  **EM:01 W:01**  **R3:03** |
| **JZ 32** | **2A** | **34 32** | **如果ZF=1，就跳转到地址码为32的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:34**  **EM:32** |
| **MOV A,R1** | **2C** | **20** | **将R1寄存器中的内容送到A寄存器中** | **FFF7F7**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:21**  **A:24** |
| **SUB R0,A** | **2D** | **0C** | **将R0寄存器中的值减去A寄存器中的值后，返回到R0寄存器中** | **FFFF8F**  **FFF7F7**  **FFFA99**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1** | **EM:0C**  **W:24**  **A:1C**  **R0:F8** |
| **JC 18** | **2E** | **30 18** | **如果CF=1，就跳转到地址码为18的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:30**  **EM:18** |
| **SHL R2** | **18** | **28** | **将R2寄存器中的内容右移一位** | **FFF7E7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2A**  **A:01**  **R2:02** |
| **SHR R1** | **19** | **2C** | **将R1寄存器中的内容右移一位** | **FFF7F7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2D**  **A:24**  **R1:12** |
| **SUB R3,#01** | **1A** | **10 01** | **将R3寄存器的值减一，之后将结果返回到R3寄存器** | **FFF7F7**  **C7FFEF**  **FFFA99**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:13**  **A:03**  **EM:01 W:01**  **R3:02** |
| **JZ 32** | **1C** | **34 32** | **如果ZF=1，就跳转到地址码为32的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:34**  **EM:32** |
| **MOV A,R1** | **1E** | **20** | **将R1寄存器中的内容送到A寄存器中** | **FFF7F7**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:21**  **A:12** |
| **ADD R0,A** | **1F** | **04** | **将寄存器R0的值跟A寄存器的值相加之后返回到R0中** | **FFF7EF**  **FFFA98**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:04**  **W:F8**  **R0:0A** |
| **JC 24** | **20** | **30 24** | **如果CF=1，就跳转到地址码为24的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:30**  **EM:24** |
| **SHL R2** | **24** | **28** | **将R2寄存器中的内容右移一位** | **FFF7F7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2A**  **A:02**  **R2:04** |
| **ADD R2,#01** | **25** | **08 01** | **将R2寄存器的值加一之后返回到R2寄存器中** | **FFF7F7**  **C7FFEF**  **FFFA98**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **W:01**  **R2:05** |
| **SHR R1** | **27** | **2C** | **将R1寄存器中的内容右移一位** | **FFF7F7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2D**  **A:12**  **R1:09** |
| **SUB R3,#01** | **28** | **10 01** | **将R3寄存器的值减一，之后将结果返回到R3寄存器** | **FFF7F7**  **C7FFEF**  **FFFA99**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:13**  **A:02**  **EM:01 W:01**  **R3:01** |
| **JZ 32** | **2A** | **34 32** | **如果ZF=1，就跳转到地址码为32的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:34**  **EM:32** |
| **MOV A,R1** | **2C** | **20** | **将R1寄存器中的内容送到A寄存器中** | **FFF7F7**  **CBFFFF** | **+1**  **写入** | **+1** | **EM:21**  **A:09** |
| **SUB R0,A** | **2D** | **0C** | **将R0寄存器中的值减去A寄存器中的值后，返回到R0寄存器中** | **FFFF8F**  **FFF7F7**  **FFFA99**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1** | **EM:0C**  **W:09**  **A:0A**  **R0:01** |
| **JC 18** | **2E** | **30 18** | **如果CF=1，就跳转到地址码为18的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:30**  **EM:18** |
| **GOTO 24** | **30** | **38 24** | **直接跳转到地址码为24的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:38**  **EM:24** |
| **SHL R2** | **24** | **28** | **将R2寄存器中的内容右移一位** | **FFF7F7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2A**  **A:05**  **R2:0A** |
| **ADD R2,#01** | **25** | **08 01** | **将R2寄存器的值加一之后返回到R2寄存器中** | **FFF7F7**  **C7FFEF**  **FFFA98**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:0A**  **W:01**  **R2:0B** |
| **SHR R1** | **27** | **2C** | **将R1寄存器中的内容右移一位** | **FFF7F7**  **FFF9BF**  **CBFFFF** | **+1**  **+1**  **写入** | **+1** | **EM:2D**  **A:09**  **R1:04** |
| **SUB R3,#01** | **28** | **10 01** | **将R3寄存器的值减一，之后将结果返回到R3寄存器** | **FFF7F7**  **C7FFEF**  **FFFA99**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:13**  **A:01**  **EM:01 W:01**  **R3:00** |
| **JZ 32** | **2A** | **34 32** | **如果ZF=1，就跳转到地址码为32的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:34**  **EM:32** |
| **TEST R0,#80** | **32** | **14** | **将R0寄存器中的内容与80H进行与运算，更改C,Z标志位** | **FFF7F7**  **C7FFEF**  **FFFE9B**  **CBFFFF** | **+1**  **+1**  **+1**  **写入** | **+1**  **+1** | **EM:14**  **A:01**  **W:FF** |
| **JZ 32** | **34** | **34 3E** | **如果ZF=1，就跳转到地址码为32的指令** | **C6FFFF**  **CBFFFF** | **+1**  **写入** | **写入**  **+1** | **EM:34**  **EM:3E** |
| **FEND** | **3E** | **3C** | **代表程序结束** | **CBFFFF** | **写入** | **+1** |  |

**7．设计结果说明**

**调试运行程序时是否出现问题，是否有重新调整指令/微指令系统设计的情况出现？请在此做具体说明。**

在考虑设计指令ADD的时候想到为什么不能是直接ADD R?,R?呢，后来才发现这个指令是不可能实现的，因为这个地方的机器码1的最后两位就是控制着R？的，因此一条指令里出现两个R？寄存器的话是不可能的，最后选择将指令拆开。

在设计跳转指令的时候，主要用到了JC，JZ，JMP，GOTO，在这里遇到了两次问题，第一次是不知道为什么没有在程序运行的时候成功跳转了，而后来才发现原来在设计微指令的时候有一个FEN的有效位需要去改变为有效才行。但是这么修改后还是不能正常跳转，查阅ELP、LDPC的资料才知道这个这个三条跳转指令看起来完全一模一样啊，但是我们需要去设计出不一样的IR3,IR2对应要求的才能正常使用跳转指令，JC第3、4位是00；JZ是01；JMP是10。通过简单调整设置指令的位置，实现了正常的跳转功能。

**五、本次课程设计的总结体会（不少于200字）**

**主要总结学到的具体知识、方法及设计中的切身体会；包括列出在设计的各个阶段出现的问题及解决方法。**

这次的计算机组成原理课程设计我收获颇丰，在掌握知识的同事，还增强了我克服困难的信心。总体来说，本次的课程设计难度适中，如果上课认真学习，完成本次课程设计是完全没有问题的，在实验的过程中，我复习了很多课上还有实验中的知识，同时也查阅了很多资料，问了老师一些问题，弄懂了很多课内外的知识点。

在设计算法的过程中，感触最深的就是逻辑一定要严密以及优化问题，无论在执行时间上还是硬件资源的使用问题上，优化都是困扰我一段时间的难题，加之还要考虑到COOP2000实验仪的硬件限制，花时间去了解COOP2000也是一件不可缺少的任务，不过效果还不错，我发现带进位移位时移出去的那一位被放进了C标志位中，这就启发我通过对部分积和乘数的带进位移位实现积的存储问题，从而使算法很容易拓展到8位乘法上。

在设计指令集时，基本上汇编程序已经写好，根据我要用到的指令来填指令集，并设计相应的微操作，基本完成后再对汇编程序进行优化，对指令集和微指令集进行修改，直到运行结果达到预期，填写跟踪过程，这一过程中花费时间较多，不过对各微指令控制位的功能确实越来越熟悉。

衷心希望学院能安排更多的此类实践项目，增强我的能力，促进课堂理论知识的消化，最后感谢计算机组成原理的几位老师的悉心指导和多位同学的帮助！