**计算机组织与体系结构课程设计**

**㈠ 基本**[**模型机设计与实现**](#_top)

**一**．**实验目的**

1．深入理解基本模型计算机的功能、组成知识；

2．深入学习计算机各类典型指令的执行流程；

3．学习微程序控制器的设计过程和相关技术，掌握LPM\_ROM的配置方法。

4．在掌握部件单元电路实验的基础上，进一步将单元电路组成系统，构造一台基本模型计算机。

5．定义五条机器指令，并编写相应的微程序，上机调试，掌握计算机整机概念。掌握微程序的设计方法，学会编写二进制微指令代码表。

6．通过熟悉较完整的计算机的设计，全面了解并掌握微程序控制方式计算机的设计方法。

**二．实验原理**

1．在部件实验过程中，各部件单元的控制信号是人为模拟产生的，而本实验将能在微过程控制下自动产生各部件单元控制信号，实现特定的功能。实验中，计算机数据通路的控制将由微过程控制器来完成，CPU从内存中取出一条机器指令到指令执行结束的一个指令周期，全部由微指令组成的序列来完成，即一条机器指令对应一个微程序。

本实验采用五条机器指令：IN（输入）、ADD（二进制加法）、STA（存数）、OUT（输出）、JMP（无条件转移），其指令格式如下（最高4位二进制数为操作码）：

2．指令格式

（1）指令格式

采用寄存器直接寻址方式，其格式如下：

|  |  |  |  |
| --- | --- | --- | --- |
| 位 | 7654 | 32 | 10 |
| 功能 | 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  01  10 | | R0  R1  R2 | |
| 助记符 | 机器指令码 | | Addr地址码 | | 功能说明 | |
| IN  ADD addr  STA addr  OUT addr  JMP addr | 0 0H  1 0H XX H  2 0H XX H  3 0H XX H  4 0H XX H | | | | “INPUT”中的数据→R0  R0+[addr] ->R0  R0 -> [addr]  [addr] -> BUS  addr →PC | |

注：基本模型机无R1,R2寄存器，但实际电路中有R1,R2寄存器。

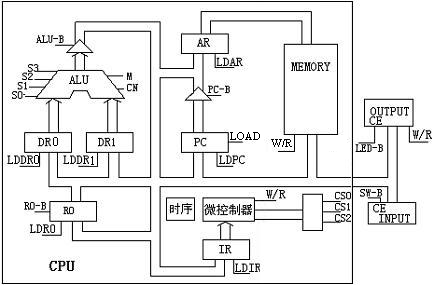


图1-1 数据通路框图

其中IN为单字长（8位二进制），其余为双字长指令，XX H 为addr对应的十六进制地址码。为了向RAM中装入程序和数据，检查写入是否正确，并能启动程序执行，还必须设计三个控制台操作微程序。

1、存储器读操作（KRD）：下载实验程序后按总清除按键（CLR）后，控制台SWA、

SWB为“0 0”时，可对RAM连续手动读出操作。

2、存储器写操作（KWE）：下载实验程序后按总清除按键（CLR）后，控制台

SWA、SWB为“0 1”时，可对RAM连续手动写操作。

1. 启动程序（RP）：下载实验程序后按总清除按键（CLR）后，控制台SWA、

SWB为“1 1”时，即可转入到微地址“01”号“取指令”微指令，启动程序运行。

根据以上要求设计数据通路框图，如图1-1所示。

|  |  |  |
| --- | --- | --- |
| SWB | SWA | 控制台指令 |
| 0  0  1 | 0  1  1 | 读内存（KRD）  写内存（KWE）  启动程序（RP） |

表1-1 24位微代码定义：

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 15 14 13 | 12 11 10 | 987 | 6 | 5 | 4 | 3 | 2 | 1 |
| S3 | S2 | S1 | S0 | M | Cn | WE | A9 A8 | | A | B | C | uA5 | uA4 | uA3 | uA2 | uA1 | uA0 |

**表1-2 A、B、C各字段功能说明：**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A字段** | | | | **B字段** | | | | **C字段** | | | |
| 15 | 14 | 13 | 选择 | 12 | 11 | 10 | 选择 | 9 | 8 | 7 | 选择 |
| 0 | 0 | 0 |  | 0 | 0 | 0 |  | 0 | 0 | 0 |  |
| 0 | 0 | 1 | LDRi | 0 | 0 | 1 | RS-B | 0 | 0 | 1 | P（1） |
| 0 | 1 | 0 | LDDR1 | 0 | 1 | 0 |  | 0 | 1 | 0 |  |
| 0 | 1 | 1 | LDDR2 | 0 | 1 | 1 |  | 0 | 1 | 1 |  |
| 1 | 0 | 0 | LDIR | 1 | 0 | 0 |  | 1 | 0 | 0 | P（4） |
| 1 | 0 | 1 | LOAD | 1 | 0 | 1 | ALU-B | 1 | 0 | 1 | LDAR |
| 1 | 1 | 0 | LDAR | 1 | 1 | 0 | PC-B | 1 | 1 | 0 | LDPC |

**24位微代码中各信号的功能**

(1) uA5—uA0：微程序控制器的微地址输出信号，是下一条要执行的微指令的微地址。

(2) S3、S2、Sl、S0：由微程序控制器输出的ALU操作选择信号，以控制执行16种算术操作或16种逻辑操作中的某一种操作。

(3) M：微程序控制输出的ALU操作方式选择信号端。M＝0执行算术操作；M＝l执行逻辑操作。

(4) /Cn：微程序控制器输出的进位信号，/Cn＝0表示ALU运算时最低位有进位，/Cn＝1则表示无进位。

(5) WE：微程序控制器输出的RAM控制信号。当/CE＝0时，如WE＝0为存储器读；如WE＝1为存储器写。

(6) A9、A8——译码后产生CS0、CS1、CS2信号，分别作为SW\_B、RAM、LED的选通控制信号。

(7) A字段（15、14、13）——译码后产生与总线相连接的各单元的输入选通信号（见表6-1）。

(8) B字段（12、11、10）——译码后产生与总线相连接的各单元的输出选通信号。

(9) C字段（9、8、7） ——译码后产生分支判断测试信号P(1)~P(4)和LDPC信号。

系统涉及到的微程序流程见图1-2。当执行“取指令”微指令时，该微指令的判断测试字段为P(1)测试。由于“取指令”微指令是所有微程序都使用的公用微指令，因此P(1)的测试结果出现多路分支（见图1-2左图）。用指令寄存器的高4位（IR7-IR4）作为测试条件，出现5路分支，占用5个固定地址单元。

控制台操作为P(4)测试（见图1-2右图），它以控制台信号SWB、SWA作为测试条件，出现了3路分支，占用3个固定微地址单元。当分支微地址单元固定后，剩下的其它地方就可以一条微指令占用控制存储器的一个微地址单元，随意填写。注意：微程序流程图上的微地址为8进制！

当全部微程序设计完毕后，应将每条微指令代码化，表1-3即为图1-2的微程序流程图按微指令格式转化而成的“二进制微代码表”。

**表1-3 二进制微代码表**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 微地址 | 微指令 | S3 S2 S1 S0 M CN WE A9 A8 | A | B | C | UA5—UA0 |
| 00 | 018110 | 000000011 | 000 | 000 | 100 | 010000 |
| 01 | 01ED82 | 000000011 | 110 | 110 | 110 | 000010 |
| 02 | 00C048 | 000000001 | 100 | 000 | 001 | 001000 |
| 03 | 00E004 | 000000001 | 110 | 000 | 000 | 000100 |
| 04 | 00B005 | 000000001 | 011 | 000 | 000 | 000101 |
| 05 | 01A206 | 000000011 | 010 | 001 | 000 | 000110 |
| 06 | 919A01 | 100100011 | 001 | 101 | 000 | 000001 |
| 07 | 00E00D | 000000001 | 110 | 000 | 000 | 001101 |
| 10 | 001001 | 000000000 | 001 | 000 | 000 | 000001 |
| 11 | 01ED83 | 000000011 | 110 | 110 | 110 | 000011 |
| 12 | 01ED87 | 000000011 | 110 | 110 | 110 | 000111 |
| 13 | 01ED8E | 000000011 | 110 | 110 | 110 | 001110 |
| 14 | 01ED96 | 000000011 | 110 | 110 | 110 | 010110 |
| 15 | 038201 | 000000011 | 000 | 001 | 000 | 000001 |
| 16 | 00E00F | 000000001 | 110 | 000 | 000 | 001111 |
| 17 | 00A015 | 000000001 | 010 | 000 | 000 | 010101 |
| 20 | 01ED92 | 000000011 | 110 | 110 | 110 | 010010 |
| 21 | 01ED94 | 000000011 | 110 | 110 | 110 | 010100 |
| 22 | 01A010 | 000000011 | 010 | 000 | 000 | 010000 |
| 23 | 018001 | 000000011 | 000 | 000 | 000 | 000001 |
| 24 | 062011 | 000001100 | 010 | 000 | 000 | 010001 |
| 25 | 010A01 | 000000010 | 000 | 101 | 000 | 000001 |
| 26 | 00D181 | 000000001 | 101 | 000 | 110 | 000001 |

指令寄存器（IR）：指令寄存器用来保存当前正在执行的一条指令。当执行一条指令时，先把它从内存取到缓冲寄存器中，然后再传送至指令寄存器。指令划分为操作码和地址码段，由二进制数构成，为了执行任何给定的指令，必须对操作码进行测试“P(1)”，通过节拍脉冲T4的控制，以便识别所要求的操作。

指令译码器: 根据指令中的操作码强置微控制器单元的微地址，使下一条微指令指向相应的微程序首地址。

实验中LCD液晶显示屏可以用来显示模型机CPU中各组成单元的内容。将B100\_C.sof文件下载到实验台后，按系统复位键，LCD液晶显示屏即显示CPU中各组成单元的内容。其功能说明如下：

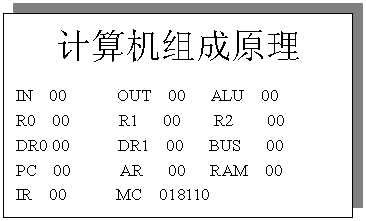


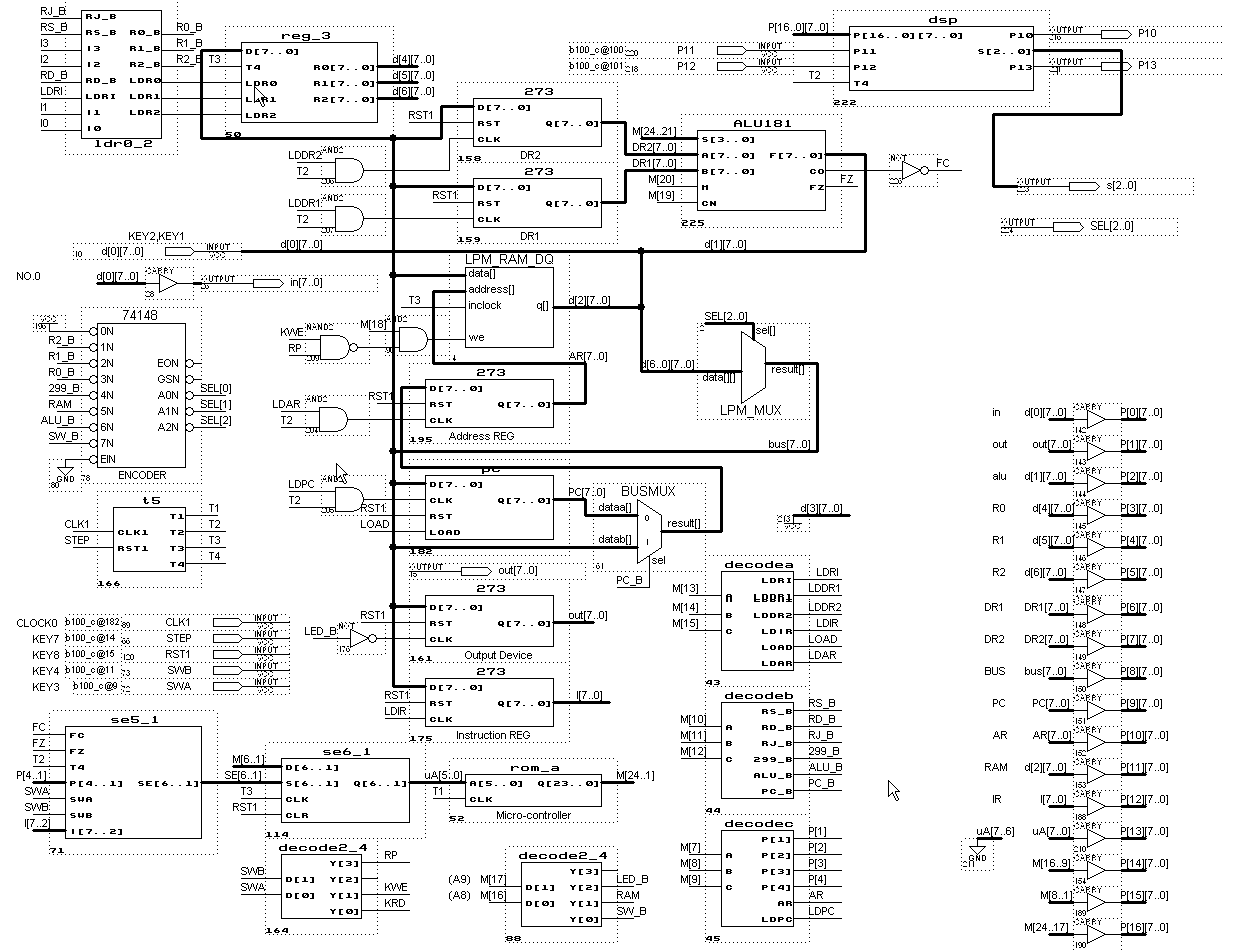
图1-2 LCD液晶显示屏

LCD液晶显示屏功能说明

|  |  |  |  |
| --- | --- | --- | --- |
| 名称 | 作 用 | 名称 | 作用 |
| IN | 输入单元INPUT | DR1 | 暂存器DR1 |
| OUT | 输出单元OUTPUT | DR2 | 暂存器DR2 |
| ALU | 算术逻辑单元 | PC | 程序计数器 |
| BUS | 内部数据总线 | AR | 地址寄存器 |
| R0 | 寄存器R0 | RAM | 程序/数据存储器 |
| R1 | 寄存器R1 | IR | 指令寄存器 |
| R2 | 寄存器R2 | MC | 微程序控制器 |

下面是一段模型机的测试程序：

|  |  |  |  |
| --- | --- | --- | --- |
| 地址（二进制） | 内容（二进制） | 助记符 | 说明 |
| 0000 0000 | 0000 0000 | IN | “INPUT DEVICE—〉R0 |
| 0000 0001 | 0001 0000 | ADD [0AH] | R0+[0AH] —〉R0 |
| 0000 0010 | 0000 1010 |  |  |
| 0000 0011 | 0010 0000 | STA [0BH] | R0—〉[0BH] |
| 0000 0100 | 0000 1011 |  |  |
| 0000 0101 | 0011 0000 | OUT [0BH] | [0BH] —〉LED |
| 0000 0110 | 0000 1011 |  |  |
| 0000 0111 | 0100 0000 | JMP [08H] | [08]—〉PC |
| 0000 1000 | 0000 0111 |  |  |
| 0000 1001 |  |  |  |
| 0000 1010 | 0000 1011 |  | 自定 |
| 0000 1011 |  |  | 求和结果 |

 图1-3 模型计算机电路原理图

实验程序1：

说明： 1、指令IN为单字节指令，指令码：00，其功能是将输入口IN的数据送到寄存 R0；

2、指令ADD [0AH]为双字节指令，指令码：100A，其功能是将R0中的数据加上RAM地址0AH中的数据（34H）；

3、指令STA [0BH]为双字节指令，指令码：200B，其功能是将R0中的数据送到RAM的0BH地址单元中；

4、指令OUT[0BH]为双字节指令，指令码：300B，其功能是将RAM的0BH地址单元中的数据送到OUT输出口上；

5、指令JMP [12H] 为双字节指令，指令码：4012，其功能是将其操作码下一地址单元（08H）的数据作为转跳地址。

三．实验步骤

1．微程序的输入：根据表1-3所对应的二进制微代码，编辑LPM\_ROM配置文件ROM\_11.mif(参考demoD\_cpu5文件夹中的同名文件)，并将其保存在与实验电路b100\_c.bdf工程所在的文件夹中，与实验电路b100\_c.bdf一同编译后，得到下载文件b100\_c.sof。下载配置文件bus\_c.sof下载到实验系统。实验板上的时钟clock0选择输入频率为1.5MHz。图1-3是示例原理图，详见b100\_c.bdf。

2．输入模型机的程序(示例工程文件是b100\_c.bdf )

（一）手动输入程序

（1）使用控制台KWE和KRD微程序将机器指令程序（“实验程序1”：按地址输入指令代码，如地址00、01、02、03、04…分别对应指令码00、10、0A、20、0B…）装入模型机CPU的程序RAM（LPM\_RAM\_DQ）中，并进行检查。根据图1-4控制台微程序流程图，在微指令的控制下，依次输入机器指令代码：

1. 以下将数据35、C4依次装入00、01地址为例：选择实验模式NO.0，输入数据显示于数码2、1上；
2. 将控制开关SWB、SWA（键4、键3）设置为：0、1；模型机的复位控制信号RST（键8）=1；

③ 机器指令代码的数据输入由键2、键1输入，先键入35，再按两次键7，即0->1->0，产生一个写入正脉冲，这时观察右上液晶屏上的输入端口IN=35；PC=00（当前将要输入的地址）；MC=018110微指令。

再按两次键7（地址寄存器加1），根据图1-4控制台微程序流程图，进入到KWE（01）分支，进入并执行了微地址“21”中的操作，这时控制此操作的微指令码 MC=01ED94，PC自动加1，PC=1。

④ 按键7，再产生一个脉冲，进入并执行了微地址“24”中的操作；观察液晶，数据35进入总线BUS=35，35进入RAM=35，此时微指令码MC=062011，此时将机器指令代码数据写入了LPM\_RAM中；

⑤ 此后每当出现MC=062011时，即可利用键2，键1输入待写入RAM的数据，此时如C4，连续按键7，再产生2个脉冲，即将C4写入RAM，PC加1，微指令码变成MC=062011；

⑥ 重复③—⑤的步骤，将“实验程序1”的全部机器指令代码输入RAM。

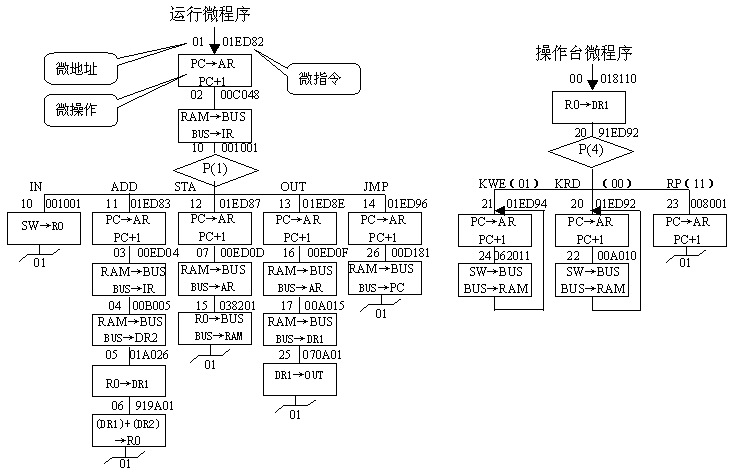


图1-4 微程序流程图(注,图中的DR1应改为DR0, DR2应改为DR1)

（2）以下是检查RAM中的内容。当全部机器指令代码输入模型机后，在微指令的控制下，依次检查LPM\_RAM中已输入的机器指令代码。步骤如下：

1. 按复位键8=0，使模型机中的PC复位；
2. 将控制开关SWB、SWA（键4、键3）设置为：0、0；
3. 复位信号RST（键8）=1；
4. 按键7，每两个2次单步运行（产生2个正脉冲），可读出LPM\_RAM中以写入的数据；根据图6-1的CPU部件和信息流程，对于读出的每一数据，仔细观察液晶上显示的MIC、PC、AR、IN、BUS、RAM、DR0、DR1的数据变化。重复以上步骤，依次检查LPM\_RAM中已输入的机器指令代码。

（二）自动配置LPM\_RAM

如果程序量大，手动输入效率太低，可以在计算机上编译好代码文件，并随同模型CPU设计文件一同编译进SOF下载文件中，直接下载进入FPGA。

（1）在QuartusII环境下，打开工程文件b100\_c.bdf，修改b100\_C.**b**df中LPM\_RAM\_DQ的参数，将初始化文件LPM\_FILE设置为：“./5\_ram.mif”；打开“5\_ram.mif”（在示例中已有此文件），根据“实验程序1”，在5\_ram.mif中输入全部机器指令代码（示例中已经输入）。

（2）将工程文件重新编译后，下载到实验台中，即完成LPM\_RAM的配置。

（3）根据以上的方法，复位信号RST（键8）=1；将控制开关SWB、SWA（键4、键3）设置为：0、0，按键7，每两个2次单步运行（产生2个正脉冲），检验配置进入FPGA中的程序代码。

（三）执行程序

（1）按1次系统复位键8，并置键8为高电平，使CPU允许正常工作；

（2）控制开关（键4、键3）设置为SWB、SWA=1,1，处于程序执行方式，观察图1-4控制台：RP（11）

（3）通过键2、键1输入运算数据，如56H，按4次单步键7，产生2个脉冲，执行2条微指令，进入到图1-4控制台的RP（11），此时的微指令地址是“23”，微指令码MC=008001；IN=56H

（4）再用键7产生1个脉冲，执行1条微指令，微程序流程进入图1-4左的“运行微程序”的最上块：此时PC=00送地址寄存器AR=00，PC自动加1，PC=01，MC=00ED82，IN=56；

注意。1、实验箱上数码管7、6显示的是下一节拍将要执行的微指令的微地址码；

2、数码管3显示的是进位情况，有进位，LED3显示1，无进位LED3显示0。

（5）键7产生1个脉冲，执行微指令MC=00C048（图1-4），RAM中的第一条指令码00进入BUS，再由BUS进入指令寄存器IR=00。键7再进1个脉冲，进入MC=001001，执行指令IN，送数IN🡪R0=56;

（6）键7产生1个脉冲，执行完IN指令后，返回到初始端，执行微指令MC=00ED82；

（7）键7产生2个脉冲后，执行微指令MC=00ED83，即执行ADD指令的第1步：PC送AR=02，PC+1=03，这时RAM中在02地址的0AH进入BUS=0A；

（8）键7产生1个脉冲，执行微指令MC=00E004，即执行ADD指令的第2步：BUS数据送AR=0A，由实验程序1可知，0A是ADD指令的加数的间接地址，而在地址0A中放有34H， 最后完成的加法是34+56；

（9）键7产生1个脉冲， MC=00B005，执行ADD的第3步：将RAM中0A地址的数据34送BUS，再送DR1=34H。注意，此时指令寄存器IR中仍放有ADD的操作码10H，PC=03指向下一条指令STA；

（10）键7产生1个脉冲，MC=01A206，执行ADD的第4步（微地址是05），R0送DR0=56；

（11）键7产生1个脉冲，MC=919A01，执行ADD的最后一步，DR0+DR1=8AH，通过总线BUS（=8A），送R0=8AH。键7再产生1个脉冲，返回到初始端，执行微指令MC=00ED82；

（12）键7产生2个脉冲，MC=00ED87，执行STA指令；键7再两2个脉冲后，和8A被写入RAM，地址单元是0B。键7一个脉冲后，返回到初始端，执行微指令MC=00ED82；

（13）键7产生2个脉冲，MC=00ED8E，执行OUT指令；此时OUT的功能是将RAM中指定地址的数据送到输出口“OUT”处（见图1-2液晶屏）。此时PC=7，AR=6；

（14）键7产生1个脉冲，MC=00E00F，首先BUS中地址0BH被锁进AR，于是RAM中对应地址0BH的数据8AH被送入BUS。键7再产生1个脉冲，MC=00A015，BUS中的数据8AH被锁进DR0；

（15）键7产生1个脉冲，MC=070A01，进入OUT指令的最后一步；8AH从DR0送入OUT口；

（16）键7产生1个脉冲，返回到初始端，执行微指令MC=00ED82；

（17）键7产生2个脉冲，MC=00ED96，指令寄存器IR=40操作码，开始执行JMP指令，此时PC=9，AR=8，故RAM中此地址的12H进入BUS=12H。注意PC！BUS的数据12H将进入PC成为下一条指令的执行地址；

（18）键7产生1个脉冲，MC=00D181，RAM=12，BUS=12，AR=08H，PC=12，IR=40开始执行JMP指令。

（19）键7产生1个脉冲，返回到初始端，执行微指令MC=00ED82；RAM=00（下一条指令码）。结束！

（四）EAB在系统读写

使用在系统EAB读写工具对模型CPU中的存放微程序的ROM和存放程序与数据的RAM进行观察和改写（图1-5）。

（五）用嵌入式逻辑分析仪了解CPU运行情况

可以利用实验系统上的液晶屏上的数据显示和嵌入式逻辑分析仪同时了解CPU的每一单步运行情况（图1-8）。注意，图1-7的嵌入式逻辑分析仪设置情况：采用时钟使用CPU的工作时钟（CLK1=1.5MHz），采样深度64位，触发位置：Pre..，触发信号用单步控制信号：STEP；触发方式：上升沿。

左侧的观察信号，data[1]是ALU；data[2]是RAM；P[10]是AR；P[12]是IR；

CPU运行的逻辑分析仪波形数据如图1-7所示。

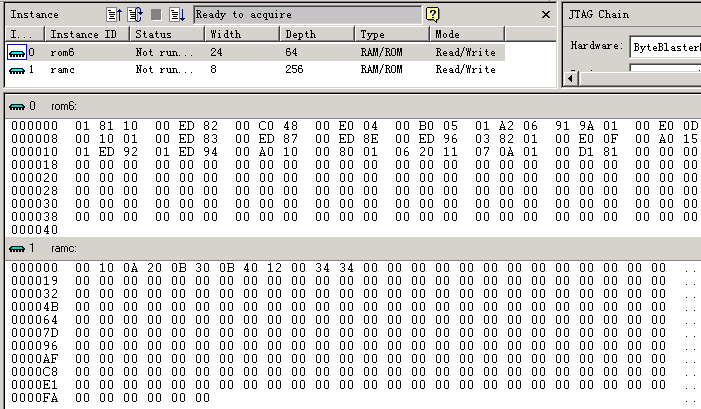


图1－5用在系统EAB读写工具对FPGA中的ROM和RAM进行观察和改写

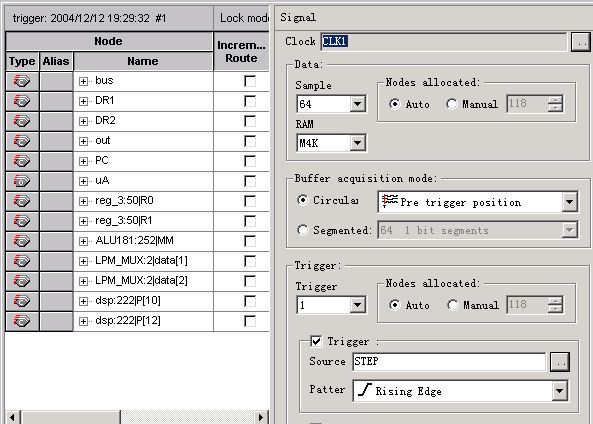


图1-6 嵌入式逻辑分析仪设置情况

四、实验要求

1．实验之前应认真准备，写出实验步骤和具体设计内容；

2．实验前应掌握所有控制信号的作用；

3．掌握在QuartusII环境下，采用图形编辑方法的设计技术；

4．掌握在微程序控制下机器指令的写入、读出、和程序执行方法；

5．掌握LPM\_RAM的配置方法，实现对机器指令输入；

6．掌握微程序的设计方法，学会编写二进制微指令代码表。

7．掌握对LPM\_ROM的配置方法，实现微指令代码表的输入。

8．通过液晶屏，观察各相关寄存器、ALU、DR1、PC、IR、AR、BUS等内容的变化情况，根据表6-2微程序控制流程，单步跟踪微程序的执行情况。通过INPUT（键2、键1）输入运算数据，跟踪程序的执行情况，并详细记录每条微指令执行后，相关单元输出数据的变化情况，依次执行机器指令，从而验证所设计的正确性。在完成基本验证实验后，根据这5条指令，自行设计程序、输入和调试，记录实验数据。

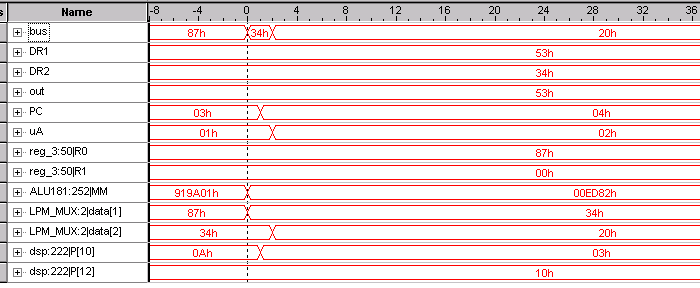


图1-7 嵌入式逻辑分析仪采样波形数据

五．实验报告

1．实验的原理，实验步骤和具体实验结果；

2．实验中遇到的主要问题和分析解决问题的思路；

3．通过实验，自己的学习经验和切身体会，以及对教学实验的意见和建议。

**六．思考题实验题**

1．若要修改算术逻辑单元ALU的功能，应如何用VHDL语言来实现？如何编程、如何用计算机进行功能仿真、时序仿真？如何通过GW48实验系统进行硬件功能验证？通过具体实验来回答。

2．用VHDL语言设计程序计数器单元PC。

3．用VHDL语言设计地址寄存器单元AR。

4．用VHDL语言设计指令寄存器单元IR。

5．将整个模型CPU都用用VHDL进行表达。

6．除了已有的IN、ADD、STA、OUT、JMP指令外，再设计减法指令SUB、带进位加ADDC、逻辑与AND、逻辑或OR和异或XOR共10条指令，编写相应微程序流程图，写出微程序代码表和相应的mif文件，并配置进LPM\_ROM中；编写由宏程序代码组成的3个程序，并用此CPU完成相应的程序功能。

7．适当修改ALU181.VHD，使之对不同操作产生的进位都能保留（锁存）。

**实验操作说明**： 下载B100\_C.SOF；选模式0，按一次复位键；8位数据in[7..0]由键2、键1输入（此值显示于键对应的数码管上）；SWB，SWA由键4，键3控制工作模式：模式00—>KRD（读出）；模式01—>KWR（写入）；模式11—>RP（程序执行）。RST1 —>模型CPU复位,键8控制，低电平有效。工作时应置高电平。STEP —>键7控制，单步执行键；Clock0 —选择1.5MHz，是CPU工作时钟。通过LCD液晶显示屏观察“基本模型机CPU”中各基本工作单元的内容。

8．将图1-8结合硬件实验详细说明此CPU对“实验程序1”的执行过程和图中各信号的含义及不同节拍下变化的原因和结果，如d0\_[7..0]、PC、AR、DR1、DR2、R0、CO、T4..T1、uA、bus[7..0]等等。

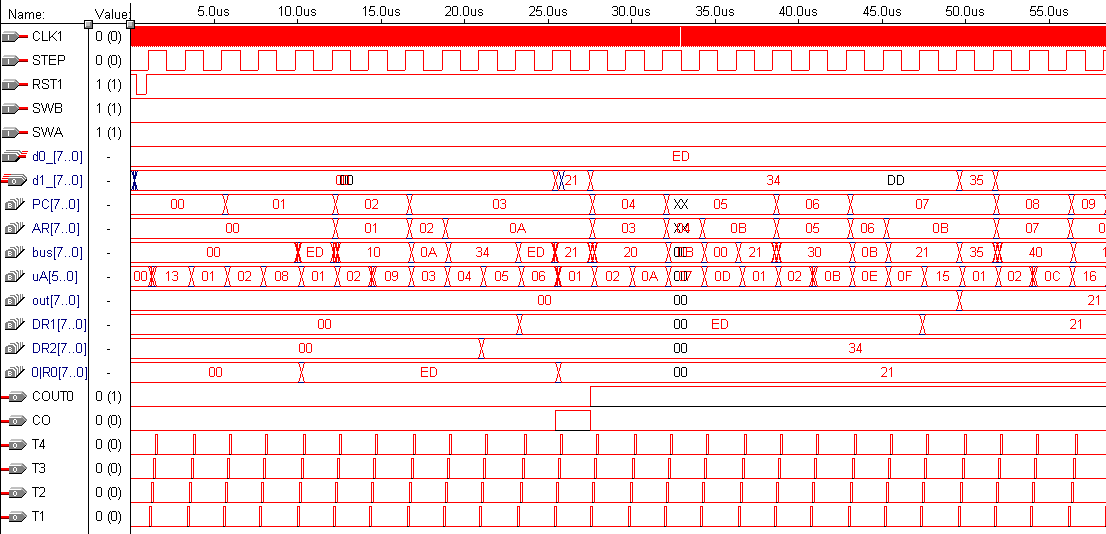
图1－8 模型CPU执行“实验程序1”的仿真波形图（注。其中的DR1，DR2分别表示液晶屏上的DR0/DR1）

图1-3中各信号的功能说明

(1) uA5—uA0 微程序控制器的微地址输出信号。

(2) IR7一IR5 指令寄存器IR7、IR6、IR5输出信号，输入至微程序控制器作为修改微地址的控制信号。

(3) CLK1——时钟信号源，输入频率为6~12MHz。

(4) T1~T4——时序信号发生器提供的四个标准输出信号，可以采用单拍或连续两种方式输出。

(5) s3、s2、sl、s0、——由微程序控制器输出的ALU操作选择信号，以控制执行16种算术操作或16种逻辑操作中的某一种操作。

(6) M—微程序控制输出的ALU操作方式选择信号端。M＝0执行算术操作；M＝l执行逻辑操作。

(7) —微程序控制器的输出的进位标志信号，＝0表示ALU运算时最低位有进位＝1，则表示无进位。

(8)SWE—器的微地址修改信号。

(9)SRD—微程序控制器的微地址修改信号。

(10) RST1—清“0”信号输入端。

(11)LDAR——微程序控制器的输入信号，将程序计数器的内容打入到存储器地址寄存器AR中，产生访问RAM的地址。

(12) /CE——微程序控制器输出的RAM选片信号，/CE＝0时，LPM\_RAM单元被选中。

(13) WE——微程序控制器输出的RAM控制信号。当/CE＝0时，如WE＝0为存储器读；如WE＝1为存储器写。

(14)BUS(7..0)——微程序控制器的内部数据总线。

(15) LDPC——微程序控制器输出的PC打入信号。

(16) LOAD——微程序控制器的输出信号。LOAD＝0时，PC(程序计数器)处于并行置数状态；LOAD＝1时，PC处于计数状态。

(17) ALU\_B—微程序控制器输出信号，控制运算器的运算结果是否送到总线BUS．低电平有效。

(18) PC\_B—微程序控制器输出信号，控制程序计数器的内容是否送到总线BUS，低电平有效。

(19) R0\_B—微程序控制器输出信号，控制寄存器R4的内容是否达到总线BUS，低电平有效。

(20) SW\_B—微程序控制器输出信号，控制（键2、键1）的八位数据是否送到总线，低电平有效。

(21) LDR0—微程序控制器的输出信号。控制把总线上的数据打入寄存器DR0。

(22) LDR1—微程序控制器输出信号，控制把总线上的数据打入寄存器DR1

(23) LDIR—微程序控制器输出信号，控制把总线上的数据(指令)输入到指令寄存器IR中。

(24)P(1)—微程序控制器输出的修改微地址P(1)，标志信号。用于机器指令的微程序分支测试。

(25)uA—微程序控制器的微地址寄存器输出控制信号，uA＝0，微地址信号输出。

(26)STEP—时序发生器启动控制信号。按2次STEP键，时序发生器可输出一组(单步)或连续的时序信号Tl、T2、T3、T4。

**㈡ 含多运算器的模型机设计与实现**

**一．实验目的**

1. 在基本模型CPU基础上，增加移位运算单元SHEFT，构建一台具有移位运算功能的模型CPU。

2．在5条基本机器指令基础上，增加4条移位运算指令，并编写相应

的微程序，上机调试，掌握CPU整机概念。3．进一步熟悉较完整的CPU设计，全面了解掌握微程序控制方式CPU的设计方法。

**二．实验原理**

在基本模型计算机实验的基础上，增加移位运算单元SHEFT。

1．数据格式： 模型机采用定点补码表示法表示数据，字长为8位，其格式如下：

|  |  |
| --- | --- |
| 7 | 6543210 |
| 符号 | 尾数 |

其中第7位为符号位，数值表示范围是：-1≤1。

2．指令格式：

（1）算术逻辑指令： 设计9条算术逻辑指令并用单字节表示，采用寄存器直接寻址方式，格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| 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  01  10 | R0  R1  R2 |

（2）移位发生器的功能 （实验中所用到的带移位运算功能的模型计算机的数据通路框图如图7-1所示。）

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| G | S1 | S0 | M | 功 能 |
| 0 | 0 | 0 | 任意 | 保持 |
| 0 | 1 | 0 | 0 | 循环右移 |
| 0 | 1 | 0 | 1 | 带进位循环右移 |
| 0 | 0 | 1 | 0 | 循环左移 |
| 0 | 0 | 1 | 1 | 带进位循环左移 |
| 任意 | 1 | 1 | 任意 | 装数 |

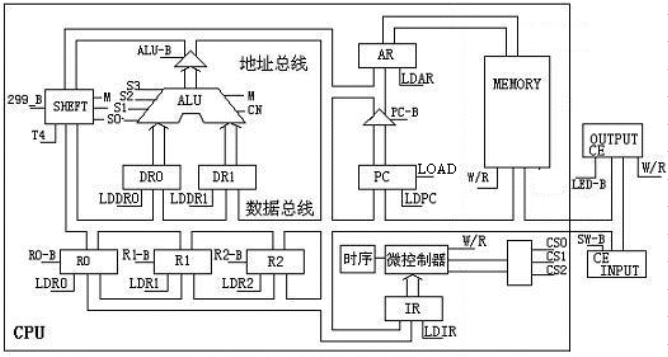


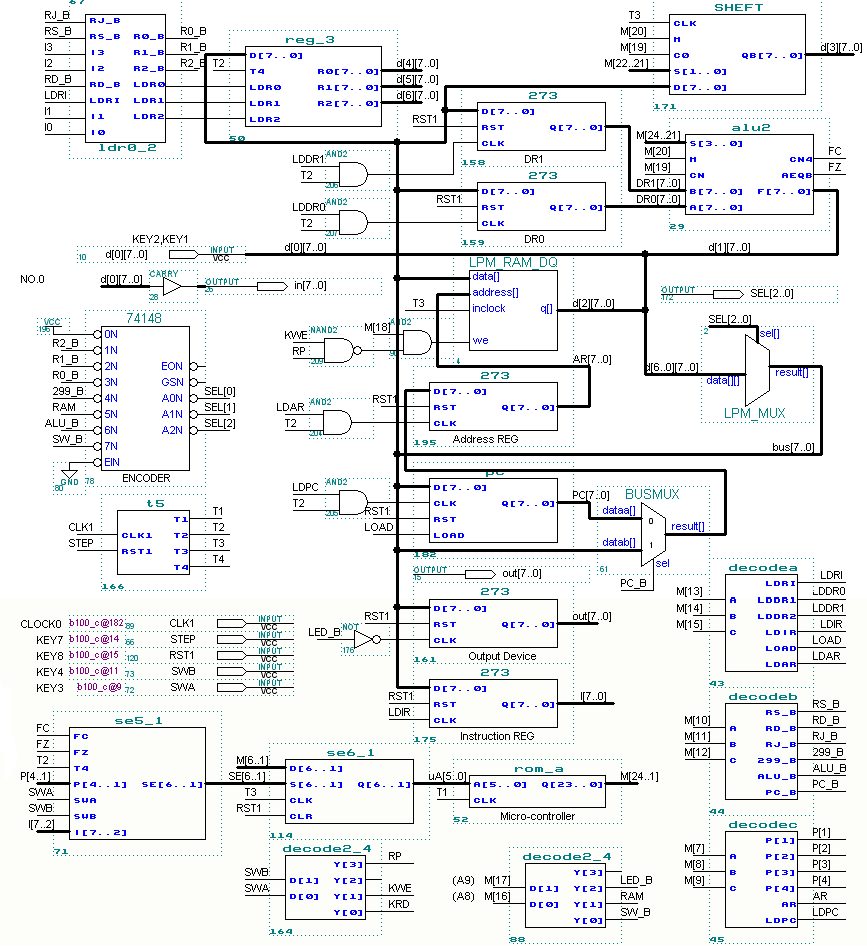
图7-1 带移位功能的模型机数据通路框图

微代码定义如表7-1所示。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 151413 | 121110 | 987 | 6 | 5 | 4 | 3 | 2 | 1 |
| S3 | S2 | S1 | S0 | M | Cn | WE | A9 | A8 | A | B | C | uA5 | uA4 | uA3 | uA2 | uA1 | uA0 |

表7-2 带移位运算模型机微程序

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 微地址 | 微指令 | S3 S2 S1 S0 M CN WE A9 A8 | A | B | C | uA5…uA0 |
| 00 | 018108 | 000000011 | 000 | 000 | 100 | 001000 |
| 01 | 01ED82 | 000000011 | 110 | 110 | 110 | 000010 |
| 02 | 00C050 | 000000001 | 100 | 000 | 001 | 010000 |
| …略，详见本实验工程文件 .\demoe\_cpu6\b100\_c.gdf中元件rom\_a中的文件rom\_11.mif | | | | | | |
| 22 | 019801 | 000000011 | 001 | 100 | 000 | 000001 |
| 23 | 198824 | 000110011 | 000 | 100 | 000 | 100100 |
| 24 | 019801 | 000000011 | 001 | 100 | 000 | 000001 |

图7-2 模型CPU内部结构图（详见./demoE\_cpu6/b100\_c.gdf ）

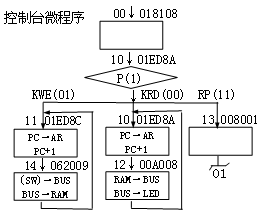


图7-3 控制台操作流程图

**三．实验步骤**

1．微程序的输入：表7-2所对应的二进制微代码可通过对FPGA中存放微代码的LPM-ROM配置文件rom\_11.mif的编辑、输入，并与实验电路一同编译后，得到下载文件b100\_c.sof。

若采用FPGA外部EEPROM存放微程序，将专用的EEPROM配置文件下载到实验系统中。

2．输入、编译实验电路文件：通过微机将编译通过的配置文件b100\_c.sof下载到实验系统。

3．输入的机器指令程序(按照上一章实验六介绍的方法)

（1）使用，在微程序控制下，通过键盘将机器指令程序写入LPM\_RAM程序存储器中，并进行检查。

（2） 对照执行程序的机器指令，将实验程序2中的代码正确地写入到LPM\_RAM的中初始化配置文件RAM\_6.mif中（在b100\_c.bdf中元件LPM\_RAM\_DQ中）。通过对LPM-RAM配置文件的编辑、输入，并与实验电路一同编译后，得到新的下载文件b100\_c.sof。

图7-4 微程序流程图

**实验程序2：**

|  |  |  |  |
| --- | --- | --- | --- |
| RAM地址（16进制）  00  01  02  03  04  05  06  07  08  09  0A  0B  0C  0D  0E | 内容（16进制）  00  10  0D  80  00  60  70  20  0E  30  0E  40  00  45 | 助记符  IN  ADD [0DH]  RLC  IN  RRC  RL  STA [0EH]  OUT [0EH]  JMP [addr] | 说 明  “INPUT DEVICE”→R0  R0+[0DH] →R0    “INPUT DEVICE”→R0    R0→[0EH]  [0EH] → OUTPUT  00→ PC  自定  存数单元 |

**四．实验要求**

1．实验之前应认真准备，写出实验步骤和具体设计内容；

2．实验前应了解所有控制信号的作用；

3．在Quartus II环境下，用图形编辑方法设计带进位模型机的数据通路；

4．在微程序控制下将机器指令写入程序存储器中，并读出检查；和程序单步调试执行方法；

5．LPM\_RAM的配置方法，实现对机器指令输入；

6．微程序的设计方法，学会编写二进制微指令代码表。

7．掌握对LPM\_ROM的配置方法，实现微指令代码表的输入。

8. 首先对已给的实例进行验证性实验，本实验工程文件在b100\_c.bdf中，是原理图文件，参考实验6，下载SOF文件b100\_c.sof。按照实验6中“（三）执行程序”给出的描述方法，依据图7-1、图7-3、图7-4和实验程序2详细描述实验程序2的微程序执行过程，并写入实验报告中。

9. 按照实验5的要求和方法，首先利用在系统EAB读写工具实时了解此CPU内的RAM和ROM中的数据，然后用嵌入式逻辑分析仪和实验箱上的液晶同时观察CPU的运行过程，给出报告。

10．自行编制一个新的的程序，根据此程序写出MIF微指令的ROM配置文件并更新rom\_11.mif文件，画出微程序流程图，更新LPM\_RAM\_DQ中的程序代码文件RAM\_6.mif，最后详细给出执行过程。

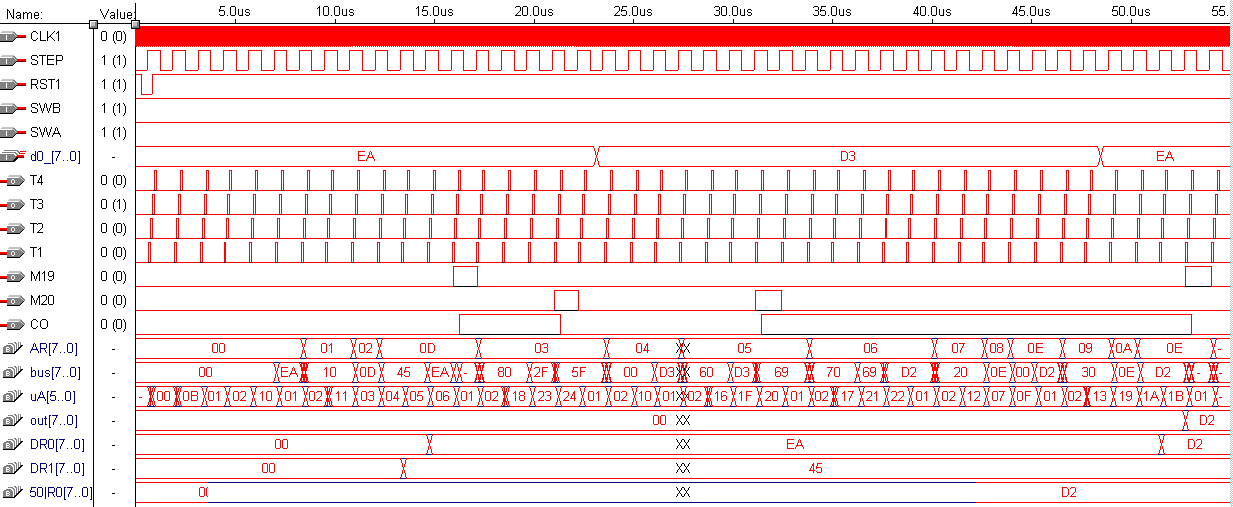


图7-5模型CPU执行“实验程序2”的仿真波形图

**五．实验报告**

1．实验的原理，实验步骤和具体实验结果；

2．实验中遇到的主要问题和分析解决问题的思路；

3．通过实验，自己的学习经验和切身体会，以及对教学实验的意见和建议。

**六．思考题和实验题**

1．参考直接转移指令“JMP [直接地址]”，编写出实现其他寻址方式的控制转移指令。

（1）立即寻址：转移地址为用立即数表示的绝对地址。“JMP 绝对地址”

（2）相对寻址：转移地址为在当前PC基础上加上相对转移偏移量“JMP 偏移地址”

（3）条件转移：根据运算标志FC、FZ的状态进行转移：

JC 相对地址 JZ相对地址

请设计相应微程序流程图，确定微程序代码，在实验台上验证所设计的功能。

2．设计一条比较指令“CMP rs，rd”，比较源操作数rs和目的操作数rd的大小，运算结果会影响标志位FC和FZ，将标志位保存在标志寄存器中。请编写相应微程序流程图，确定相应的微程序代码。

3．如果执行程序的机器指令存放在外部存储器EEPROM中，模型机的相应电路应如何修改？如何通过控制台SWB和SWA键设定为输入方式，输入执行程序的机器指令？

4．加法指令采用双地址格式，一个操作数采用隐含寻址方式，存放在寄存器R0中，另一个操作数在存储器中：

（1）若另一个操作数采用寄存器间接寻址，设计出指令的格式，并画出其微程序流程图。

（2）若另一个操作数采用存储器间接寻址，设计出指令的格式，并画出其微程序流程图。

（3）若另一个操作数采用变址寻址，设计出指令的格式，并画出其微程序流程图。

（4）若另一个操作数采用基址加变址寻址，设计出指令的格式，并画出其微程序流程图。

5．根据图7-5，在实验箱上完成相应实验，将实验数据与波形数据进行比较，并根据图7-5和电路图b100\_c.bdf,详细说明CPU对实验程序2的执行过程和图中所有信号在不同节拍下变化的原因和结果。

**㈢ 特定应用程序模型机设计与实现**

**一．实验目的**

1．综合运用所学计算机原理知识，设计并实现较为复杂的计算机。

2．设计指令系统。

3．编写简单程序，在所设计的复杂模型计算机上调试运行。

**二．实验原理**

在之前模型计算机实验的基础上，增强模型计算机功能。

1．数据格式：模型机采用定点补码表示法表示数据，字长为8位，其格式如下：

|  |  |
| --- | --- |
| 7 | 6543210 |
| 符号 | 尾数 |

其中第7位为符号位，数值表示范围是：-1≤1。

2．指令格式：

（1）算术逻辑指令：设计8条算术逻辑指令并用单字节表示，采用寄存器直接寻址方式，格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| 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  01  10 | R0  R1  R2 |

（2）移位发生器的功能 （实验中所用到的带移位运算功能的模型计算机的数据通路框图如图7-1所示。）

以实验㈣设计题目的第2题为例说明设计过程。

求1到任意一个整数之间的所有奇数之和并输出显示。

1．编写满足题目要求的汇编语言源程序

|  |  |  |
| --- | --- | --- |
|  | 汇编语言源程序： | 功 能 |
| LP0： | IN R0 | 从开关输入任意一个整数n—>R0 |
|  | MOV R1，1 | 将立即数1—>R1（R1存放参与运算的奇数） |
|  | MOV R2，0 | 将立即数0—>R2（R2存放累加和） |
| LP1： | CMP R0，R1 | 将R0中的整数n与R1中的奇数进行比较 |
|  | JB LP2 | 若R1<R0，则转到LP2处执行 |
|  | ADD R1，R2 | 否则，累加求和 |
|  | INC R1 | R1的内容加2，形成下一个奇数 |
|  | INC R1 |  |
|  | JMP LP1 | 跳转到LP1继续执行 |
| LP2： | OUT R2 | 输出累加和 |
|  | JMP LP0 | 重新开始 |

2．确定指令格式。 为了完成求和功能，需要使用8条指令，其中包括算术指令、I/O指令、访问、转移指令和加1指令。

（1）I/O指令。 输入（IN）和输出（OUT）指令采用单字节指令，其格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| 操作码 | addr | 目的寄存器 |

其中，addr=01时选中“INPUT DEVICE”中的键盘输入设备，addr=10时，选中“OUTPUT DEVICE”中的LCD点阵液晶屏作为输出设备。

（2）比较和相加指令。 比较指令（CMP）和相加指令（ADD）用单字节表示，采用寄存器直接寻址方式，其格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| 操作码 | 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  01  10 | R0  R1  R2 |

（3）转移指令。无条件转移（JMP）和结果为零或有进位转移指令（JB），指令格式为：

|  |  |
| --- | --- |
| 7654 | 3210 |
| 操作码 | XXXX |
| 地 址 | |

（4）MOV指令。 指令格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| 操作码 | XX | Rd |
| 立 即 数 | | |

（5）加1指令INC。 指令格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| 操作码 | XX | Rd |

（5）数据格式：

|  |  |
| --- | --- |
| 7 | 6543210 |
| 符号位 | 尾 数 |

3．指令系统。本模型机有8条基本指令。每条指令格式、助记符、功能如表9-1所示。

表9-1指令系统

|  |  |  |
| --- | --- | --- |
| 助记符号 | 指令格式 | 功能 |
| IN rd | |  |  |  | | --- | --- | --- | | 1000 | XX | rd | | input → rd寄存器 |
| OUT rd | |  |  |  | | --- | --- | --- | | 1111 | XX | rd | | rd → output |
| ADD rs，rd | |  |  |  | | --- | --- | --- | | 1100 | rs | rd | | rs + rd → rd |
| CMP rs，rd | |  |  |  | | --- | --- | --- | | 1010 | rs | rd | | rs –rd → rd |
| INC rd | |  |  |  | | --- | --- | --- | | 1101 | XX | rd | | Rd +1→ rd |
| MOV data，rd | |  |  |  | | --- | --- | --- | | 1001 | XX | rd | | data | | | | data rd |
| JMP addr | |  |  | | --- | --- | | 1110 | XXXX | | addr | | | Addr →PC |
| JB addr | |  |  | | --- | --- | | 1011 | XXXX | | addr | | | 若小于，则 addr → PC |

4.按照指令格式将汇编语言源程序汇编成机器代码。 与汇编语言源程序对应的机器语言源程序：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 助记符 | | 地址（十六进制） | 机器代码（十六进制） | 功能 |
| LP0： | IN R0 | 00 | 80 | Input → R0 |
|  | MOV R1，1 | 01 | 91 | 1→R1 |
|  |  | 02 | 01 |  |
|  | MOV R2，0 | 03 | 92 | 0→R2 |
|  |  | 04 | 00 |  |
| LP1： | CMP R0，R1 | 05 | A1 | R0-R1→R1 |
|  | JB L2 | 06 | B0 | (LP2) →PC |
|  |  | 07 | 0D |  |
|  | ADD R1，R2 | 08 | C6 | R1+R2→R2 |
|  | INC R1 | 09 | D1 | R1+1→R1 |
|  | INC R1 | 0A | D1 | R1+1→R1 |
|  | JMP L1 | 0B | E0 | (LP1)→PC |
|  |  | 0C | 05 |  |
| LP2： | OUT R2 | 0D | F2 | R2→output |
|  | JMP LP0 | 0E | E0 | (LP0)→PC |
|  |  | 0F | 00 |  |

5．确定微地址和微指令

表9-2 微地址和微指令表

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 微地址 | 微指令 | S3 S2 S1 S0 M CN WE A9 A8 | A | B | C | UA5—UA0 |
| 00 | 018110 | 000000011 | 000 | 000 | 100 | 010000 |
| 01 | 00ED82 | 000000001 | 110 | 110 | 110 | 000010 |
| 02 | 00C048 | 000000001 | 100 | 000 | 001 | 001000 |
| …略，详见本实验工程文件 .\demoe\_cpu78\b100\_8.gdf中元件rom\_a中的文件rom\_ex8.mif | | | | | | |
| 27 | 00D181 | 000000001 | 101 | 000 | 110 | 000001 |
| 30 | 00D181 | 000000001 | 101 | 000 | 110 | 000001 |
| 31 | 919A01 | 000000011 | 000 | 000 | 000 | 000001 |
| 32 | 919B41 | 100100011 | 001 | 101 | 101 | 000001 |

6．根据微指令表编写微指令LPM\_rom的初始化文件(示例文件在b100\_8.bdf中元件rom\_a中的文件rom\_ex8.mif)

7. 设计微程序流程图

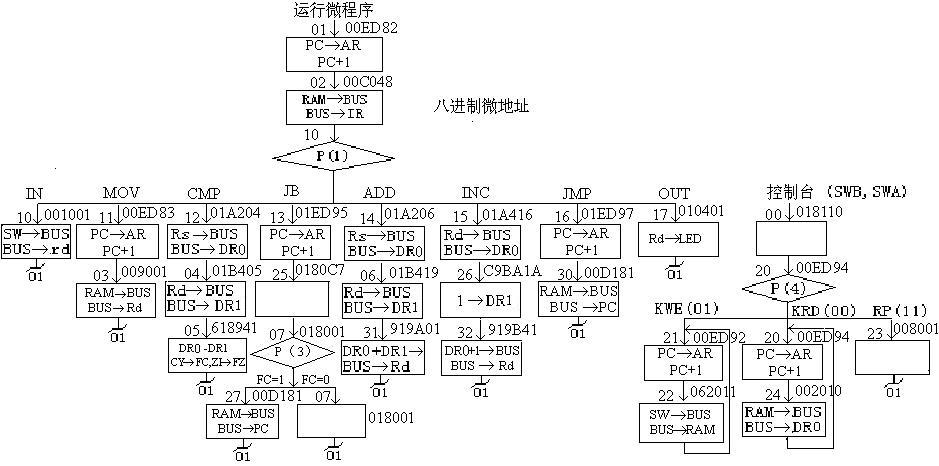
8．在MUX-PLUS II环境下设计模型机电路的工程文件(示例文件是.\demoe\_cpu78\b100\_8.gdf)。将模型机电路图文件与微指令LPM\_rom的初始化文件一起重新编译，并下载到实验台目标系统中(程序示例文件在.\demoe\_cpu78\b100\_8.gdf中元件LPM\_RAM\_DQ中的ram\_8.mif)。

9．在微程序的控制下通过实验台键盘输入程序的机器指令代码。

10．单步运行调试程序。 记录程序执行过程中的实验数据、通过LCD显示屏跟踪程序的执行情况，观察、分析所设计的微指令是否正确，发现问题及时调整、修改，重新进行编译、下载和单步调试。

11．连续运行。若单步调试正确，可在键盘（input）输入程序所需数据后将实验台从单步切换到连续运行方式。

12．实验报告：（1）设计过程。 指令系统、微程序流程图、汇编语言源程序和对应的机器语言源程序、模型机原理图，工作原理。（2）实验数据。 实验数据须经过实验教师签字验收。调试过程、问题排查、数据处理、结论。（3）对设计过程的分析和总结

图9-1 微程序流程图

**㈣** [**复杂模型机的设计与实现**](#_top)

**一．实验目的**

1．综合运用所学计算机原理知识，设计并实现较为完整的计算机。

2．设计指令系统。

3．编写简单程序，在所设计的复杂模型计算机上调试运行。

**二．实验原理**

1．数据格式: 模型机采用定点补码表示法表示数据，字长为8位，其格式如下：

|  |  |
| --- | --- |
| 7 | 6543210 |
| 符号 | 尾数 |

其中第7位为符号位，数值表示范围是：-1≤1。

2．指令格式: 所设计的指令分为四大类共十六条，其中包括算术逻辑指令、I/O指令、访问、转移指令和停机指令。

（1）算术逻辑指令。

设计9条算术逻辑指令并用单字节表示，采用寄存器直接寻址方式，其格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| 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  01  10 | R0  R1  R2 |

（2）访问指令及转移指令: 访问指令有2条，即存数（STA）、取数（LDA）；2条转移指令，即无条件转移（JMP）、结果为零或有进位转移指令（BZC），指令格式为：

|  |  |  |  |
| --- | --- | --- | --- |
| 76 | 54 | 32 | 10 |
| 00 | M | OP-CODE | Rd |
| D | | | |

其中，OP-CODE为操作码，rd为目的寄存器地址（用于LDA、STA指令）。D为位移量（正负均可），M为寻址模式，其定义如下：

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

在本模型机中规定变址寄存器RI为寄存器R2。

（3）I/O指令: 输入（IN）和输出（OUT）指令采用单字节指令，其格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| OP-CODE | addr | Rd |

其中，addr=01时选中“INPUT DEVICE”中的键盘输入设备，addr=10时，选中“OUTPUT DEVICE”中的LCD点阵液晶屏作为输出设备。

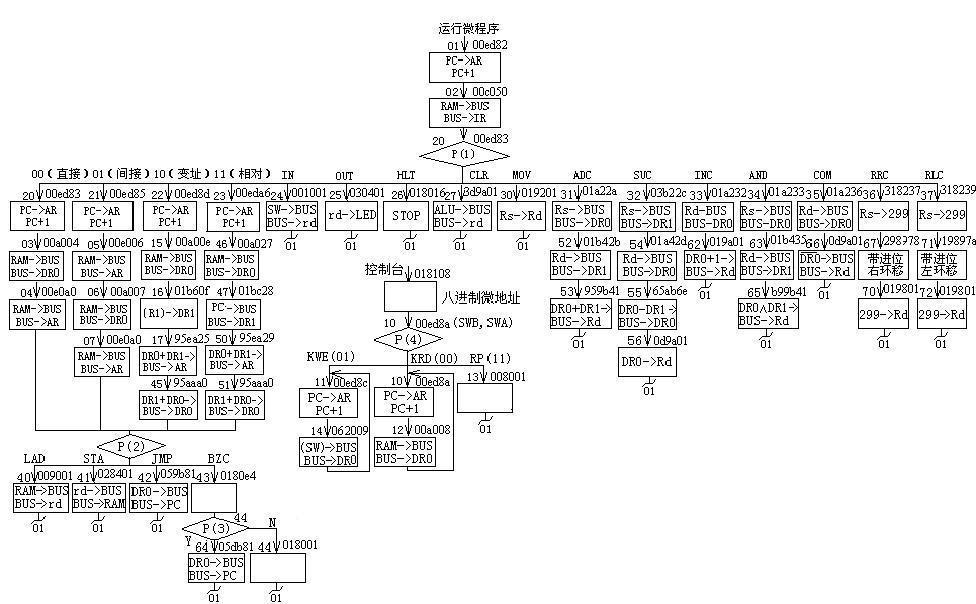
（4）停机指令: 指令格式如下：

|  |  |  |
| --- | --- | --- |
| 7654 | 32 | 10 |
| OP-CODE | 00 | 00 |

3．指令系统: 共有16条基本指令，其中算术逻辑指令7条，访问内存指令和程序控制指令4条，输入/输出指令2条，其他指令1条。各条指令的格式、汇编符号、功能如表8-1所示。

|  |  |  |
| --- | --- | --- |
| 助记符号 | 指令格式 | 功能 |
| CLR rd | |  |  |  | | --- | --- | --- | | 0111 | 00 | rd | | 0 → rd |
| MOV rs，rd | |  |  |  | | --- | --- | --- | | 1000 | rs | rd | | rs → rd |
| ADC rs，rd | |  |  |  | | --- | --- | --- | | 1001 | rs | rd | | rs + rd + cy → rd |
| SBC rs，rd | |  |  |  | | --- | --- | --- | | 1010 | rs | rd | | rs –rd – cy → rd |
| INC rd | |  |  |  | | --- | --- | --- | | 1011 |  | rd | | rd + 1 → rd |
| AND rs，rd | |  |  |  | | --- | --- | --- | | 1100 |  | rd | | rs ∧ rd → rd |
| COM rd | |  |  |  | | --- | --- | --- | | 1101 |  | rd | | → rd |
| RRC rs，rd | |  |  |  | | --- | --- | --- | | 1110 |  | rd | | 10 |
| RLC rs，rd | |  |  |  | | --- | --- | --- | | 1111 |  | rd | | 10 |
| LDA M，D，rd | |  |  |  |  | | --- | --- | --- | --- | | 00 | M | 00 | rd | | D | | | | | E → rs |
| STA M，D，rd | |  |  |  |  | | --- | --- | --- | --- | | 00 | M | 01 | rd | | D | | | | | rd → E |
| JMP M，D | |  |  |  |  | | --- | --- | --- | --- | | 00 | M | 10 | rd | | D | | | | | E → PC |
| BZC M，D | |  |  |  |  | | --- | --- | --- | --- | | 00 | M | 11 | rd | | D | | | | | 当CY=1或Z=1时，  E → PC |
| IN addr，rd | |  |  |  | | --- | --- | --- | | 0100 | 01 | rd | | addr → rd |
| OUT addr，rd | |  |  |  | | --- | --- | --- | | 0101 | 10 | rd | | rd → addr |
| HALT | |  |  |  | | --- | --- | --- | | 0110 | 00 | 00 | | 停机 |

本模型机的数据通路框图如图7-1。根据机器指令系统要求，设计微程序流程图及确定微地址，如图8-3。

图8-3 微程序流程图

**三．实验内容**

按照系统建议的微指令格式，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，并将二进制代码表转换为联机操作时的十六进制格式文件。

微代码定义如表7-1所示。

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 24 | 23 | 22 | 21 | 20 | 19 | 18 | 17 | 16 | 151413 | 121110 | 9 8 7 | 6 | 5 | 4 | 3 | 2 | 1 |
| S3 | S2 | S1 | S0 | M | Cn | WE | A9 | A8 | A | B | C | uA5 | uA4 | uA3 | uA2 | uA1 | uA0 |

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **A字段** | | | | **B字段** | | | | **C字段** | | | |
| 15 | 14 | 13 | 选择 | 12 | 11 | 10 | 选择 | 9 | 8 | 7 | 选择 |
| 0 | 0 | 0 |  | 0 | 0 | 0 |  | 0 | 0 | 0 |  |
| 0 | 0 | 1 | LDRi | 0 | 0 | 1 | RS-B | 0 | 0 | 1 | P（1） |
| 0 | 1 | 0 | LDDR0 | 0 | 1 | 0 | RD-B | 0 | 1 | 0 | P（2） |
| 0 | 1 | 1 | LDDR1 | 0 | 1 | 1 | RI-B | 0 | 1 | 1 | P（3） |
| 1 | 0 | 0 | LDIR | 1 | 0 | 0 | 299-B | 1 | 0 | 0 | P（4） |
| 1 | 0 | 1 | LOAD | 1 | 0 | 1 | ALU-B | 1 | 0 | 1 | AR |
| 1 | 1 | 0 | LDAR | 1 | 1 | 0 | PC-B | 1 | 1 | 0 | LDPC |

**实验程序3：**

|  |  |  |
| --- | --- | --- |
| 程序 | 助记符 | 功能 |
| $P00 44 | IN 01，R0 | Input =>R0 |
| $P01 46 | IN 01，R2 | Input =>R2 |
| $P02 98 | ADC R2，R0 | (R0)+(R2) =>R0 |
| $P03 81 | MOV R0，R1 | (R0) =>R1 |
| $P04 F5 | RLC R1，R1 | R1带进位左移 =>R1 |
| $P05 0C | BZC 00，00 | Cy、Z为1时，循环 |
| $P06 00 |  |  |

表7-2A微程序代码

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| 微地址 | 微指令 | S3 S2 S1 S0 M CN WE A9 A8 | A | B | C | UA5—UA0 |
| 00 | 018108 | 000000011 | 000 | 000 | 100 | 001000 |
| 01 | 00ED82 | 000000001 | 110 | 110 | 110 | 000010 |
| 02 | 00C050 | 000000001 | 100 | 000 | 001 | 010000 |
| …略，详见本实验工程文件 .\demoe\_cpu78\b100\_7.gdf中元件rom\_7中的文件rom\_ex7.mif | | | | | | |
| 71 | 19897a | 000110011 | 000 | 100 | 101 | 111010 |
| 73 | 019801 | 000000011 | 001 | 100 | 000 | 000001 |
| 74 | 070a08 | 000001110 | 000 | 101 | 000 | 001000 |
| 75 | 062009 | 000001100 | 010 | 000 | 000 | 001001 |

表7-2B微程序代码（详见本实验工程文件 .\demoe\_cpu78\b100\_7.gdf中元件rom\_7中的文件rom\_ex7.mif）

|  |  |  |  |
| --- | --- | --- | --- |
| 微地址 代码 | 微地址 代码 | 微地址 代码 | 微地址 代码 |
| $M00 018108  $M01 00ed82  $M02 00c050  …略  $M0E 01b60f  $M0F 95ea25 | $M10 00ed83  $M11 00ed85  …略  $M1D 01a236  $M1E 318237  $M1F 318239 | $M20 009001  $M21 028401  $M22 01db81  …略  $M2E 0d9a01  $M2F 01aa30 | …略  $M3A 019801  $M3B 070a08  $M3C 062009  $M3D 000000  $M3E 000000 |

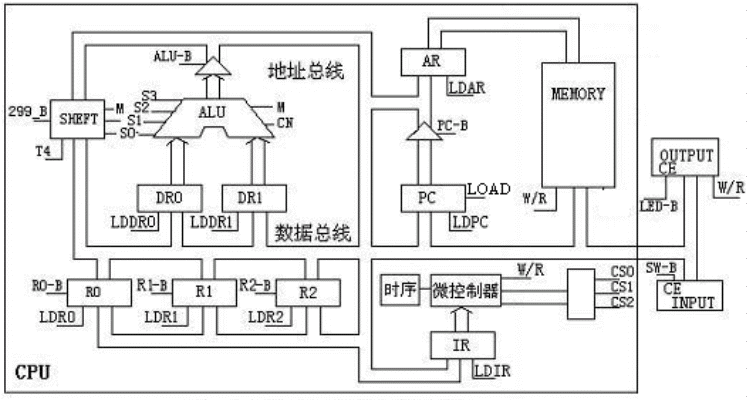


图8-1 复杂模型机数据通路框图

**四．实验步骤**

1、打开GW48系统的电源；编辑微指令的配置文件ROM\_EX7.mif（内容参见表8-2所示），并与模型机的工程文件B100\_C.gdf一起重新编译，生成下载文件B100\_C.SOF。

2、对照执行程序的机器指令，将实验程序3中的代码正确地写入到LPM\_RAM的中初始化配置文件RAM\_7.mif中（在b100\_7.bdf中元件LPM\_RAM\_DQ中）。通过对LPM-RAM配置文件的编辑、输入，并与实验电路一同编译后，得到新的下载文件b100\_7.sof，下载此文件到FPGA中；

３、用实验台的模式选择键选择模式“0”，按一次右侧的复位键；

４、in[7..0]——键2和键1,数据输入8位（此值显示于键对应的数码管上）；SWB、SWA — 键4、键3进行工作模式选择：00—KRD（读出）模式；01—KWR（写入）模式；11—RP（程序执行）模式。

RST1— 键8；CPU复位键，低电平复位；

STEP— 键7；单步执行键；Clock0——选择6~12MHz。

５、选择SWB、SWA=01（写入）模式。在微程序的控制下，通过实验台（键2

和键1）输入程序的机器指令代码，按单步执行键（键7），将机器指令代码写入程序存储器，直到输入全部指令代码。

６、选择SWB、SWA=00（读出）模式。在微程序的控制下，按单步执行键（键7）读出程序存储器机器指令代码，与写入指令代码进行比较。

７、单步执行程序，通过LCD液晶显示屏观察“复杂模型机CPU”中各基本工作单元内容。根据自己所设计的实验程序，按照微程序流程图，跟踪微指令和指令的执行情况。

**五．实验方法：**

1、手动输入程序代码。将SWB0、SWA=1和RST=1（1🡪0🡪1复位一次，并进入正常工作状态‘1’），系统处于输入RAM程序方式，即键4=0、键3=1、键8=1; 通过键1=键2输入RAM数据，通过键7产生输入脉冲，在微程序控制下将执行程序的机器指令代码输入到程序RAM中。

2、手动读出程序代码。将SWA=0、SWB=0和RST=1，系统处于读出RAM数据方式，即键3=0、键4=0,键8=1; 通过键7读出脉冲，在微程序控制下，根据地址寄存器AR的变化，通过LCD观察读出RAM中程序的机器指令代码数据，进行比较。

3、执行程序（由于有预先加载于RAM中的程序代码文件RAM\_7.mif，可直接运行程序）

首先对已给的实例进行验证性实验，本实验工程文件是b100\_c.bdf中

（1）按“系统复位”键使实验台复位，程序计数器清零PC=00，微地址寄存器清0。

（2）将SWB、SWA=11和RST置高电平，系统处于执行程序方式，即键3=1、键4=1,键8=1;

（3）键7产生脉冲，单步执行微指令。若要从“INPUT DEVICE”输入数据，可通过实验台键2、键1输入相应数据。根据微程序流程图，跟踪每一条微指令的执行情况，观察LCD显示的模型机中各单元的内容、分析程序执行情况、验证程序的功能。

（4）参考实验五，下载SOF文件b100\_7.sof。按照实验五中“（三）执行程序”给出的描述方法，依据图8-1、图8-3和实验程序3，详细描述实验程序3的微程序执行过程，并写入实验报告中。

分别输入加操作后无进位和有进位两对数分别进行实验，

设1） IN=82，IN=A9；

2）　N=45，IN=3C。通过数码3观察进位情况（注意，此程序要通过IN分别键入不同的数据！）

4．利用本章的16条指令，自行编制一个新的的程序，根据此程序写出MIF微指令的ROM配置文件并更新ROM\_EX7.mif文件，画出微程序流程图，更新LPM\_RAM\_DQ中的程序代码文件RAM\_7.mif，最后详细给出执行过程。

**六****．实验报告**

1. 实验目的、实验原理；
2. 实验步骤，实验记录数据和实验结果分析；
3. 对遇到的主要问题进行分析、提出解决问题的方法；

**七．设计实验题目**

1．输入任意几个整数，求其和并存储、输出显示。

2．求1到任意一个整数之间的所有奇数之和并输出显示。求1到任意一个整数之间的所有偶数之和并输出显示。

3．求1到任意一个整数之间的所有能被3整除的数之和并输出显示。

4． A、让实验台上的8个发光二极管D1~D8从左向右依次轮流循环显示。

B、让实验台上的8个发光二极管中的两个从右向左依次轮流循环显示。

5． A、让输出设备OUTPUT显示数据加1计数。 B、让输出设备OUTPUT显示数据减1计数。

6．对存储器RAM中40H~4FH单元的数据求和，结果存放到50H~51H中，计算其平均值存放到52H单元并输出显示。

7．存储器中存放在从40H和50H开始的两个多字节数相加，将结果存放在60H开始的存储单元中。

8．若要增加微程序可用的存储空间，对现有的模型机结构需作哪些改动？哪些控制部件需要改、怎样改？

9．对照图7-5，给出本实验中的CPU执行“实验程序3”的仿真波形图。