苏州大学实验报告

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 院、系 | 计算机 | | 年级专业 | | 计算机科学与技术 | | 姓名 | 张延磊 | 学号 | 59 |
| 课程名称 | | CPU与简单模型机设计实验 | | | | | | | 成绩 |  |
| 指导教师 | |  | | 同组实验者 | | 无 | | 实验日期 |  | |

|  |  |
| --- | --- |
| 实 验 名 称 | 实验三 CPU与简单模型机设计实验 |

1. 实验目的

掌握一个简单CPU的组成原理，进一步将其构成一台基本模型计算机。

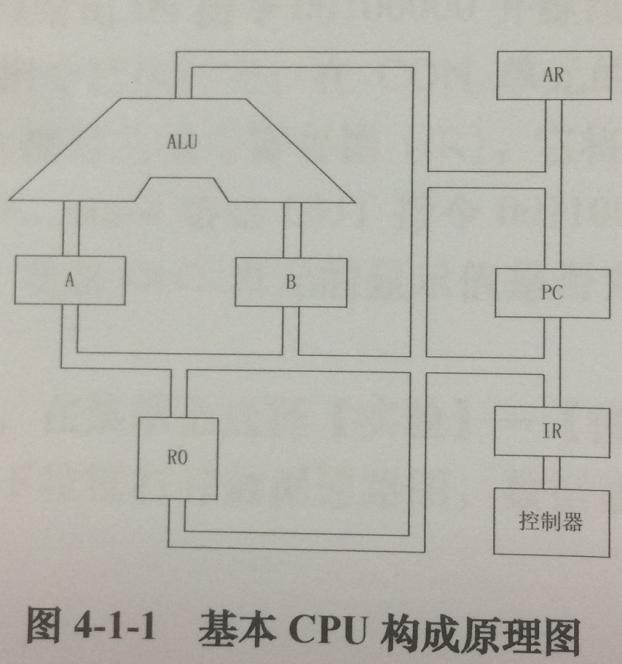
1. 实验内容

定义五条机器指令，编写相应的微程序，实现基本模型计算机功能。

设计一段机器程序，读入一个数据，存于寄存器中，再将该数据自加，结果输出单元显示。

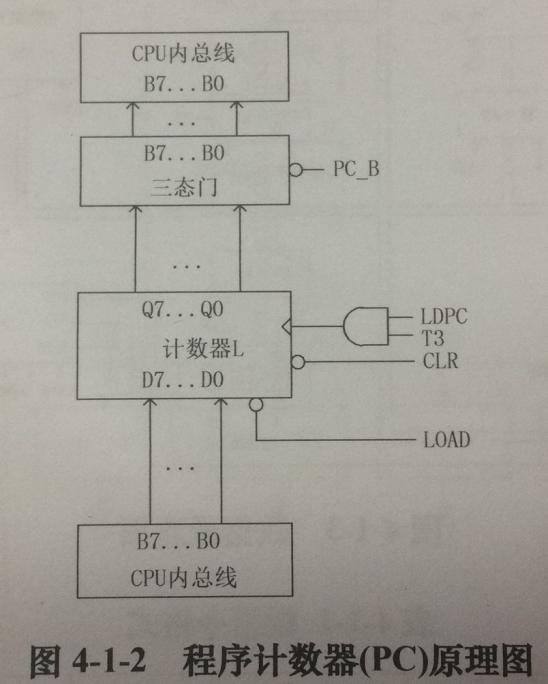
1. 实验原理(画出原理图)

本实验要实现一个简单的CPU，并且在此CPU的基础上，继续构建一个简单的模型计算机CPU由运算器（ALU）、位程序处理器（MC）、通用寄存器（R0），指令寄存器（IR）、程序计数器（PC）和地址寄存器（AR）组成，如图4-1-1所示。这个CPU在写入相应的微指令后，具备了执行机器指令的功能，但是机器指令一般存放在主存当中，CPU必须和主存挂接后，才有实际的意义，所以还需要在该CPU的基础上增加一个主存和基本的输入输出部件，以构成一个简单的模型计算机。



系统的程序计数器（PC）和地址寄存器（AR）集成在一片CPLD芯片中。CLR连接至CON单元的总清端CLR，按下CLR按钮，将使PC清零，LDPC和T3想与后作为计数器的计数时钟，当LOAD为低时，计数时钟到来后将CPU内总线上的数据打入PC。

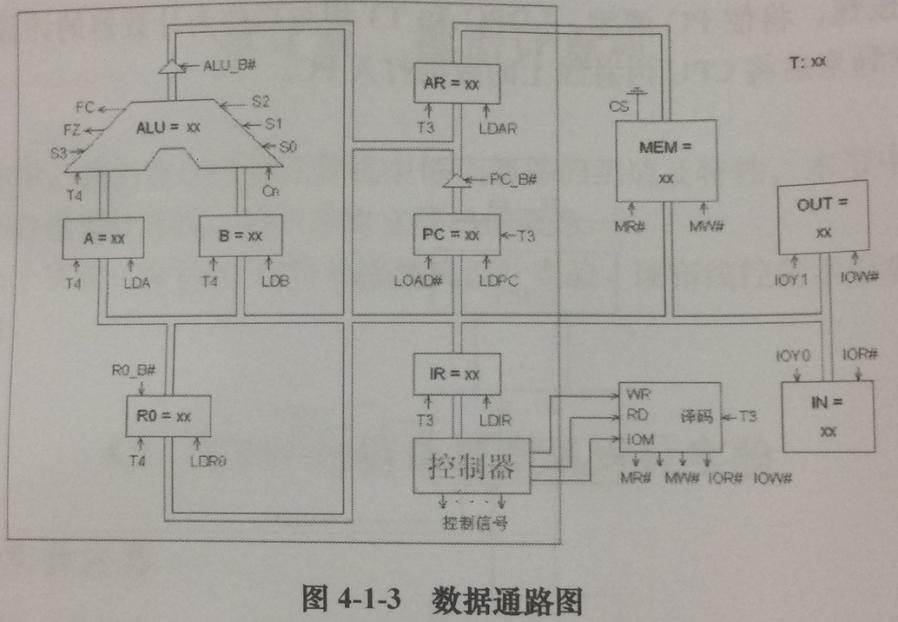
本模型机和前面微程序控制器试验相比，新增加一条跳转指令JMP，共有五条指令：



IN（输入）、ADD（二进制加法）、OUT（输出）、JMP（无条件转移），HLT（停机），其指令格式如下（高4位为操作码）：

|  |  |  |
| --- | --- | --- |
| 助记符 | 机器指令码 | 说明 |
| IN | 0010 0000 | IN → R0 |
| ADD | 0000 0000 | R0 + R0 → R0 |
| OUT | 0011 0000 | R0 → OUT |
| JMP addr | 1110 0000 \*\*\*\*\*\*\*\* | addr → PC |
| HLT | 0101 0000 | 停机 |

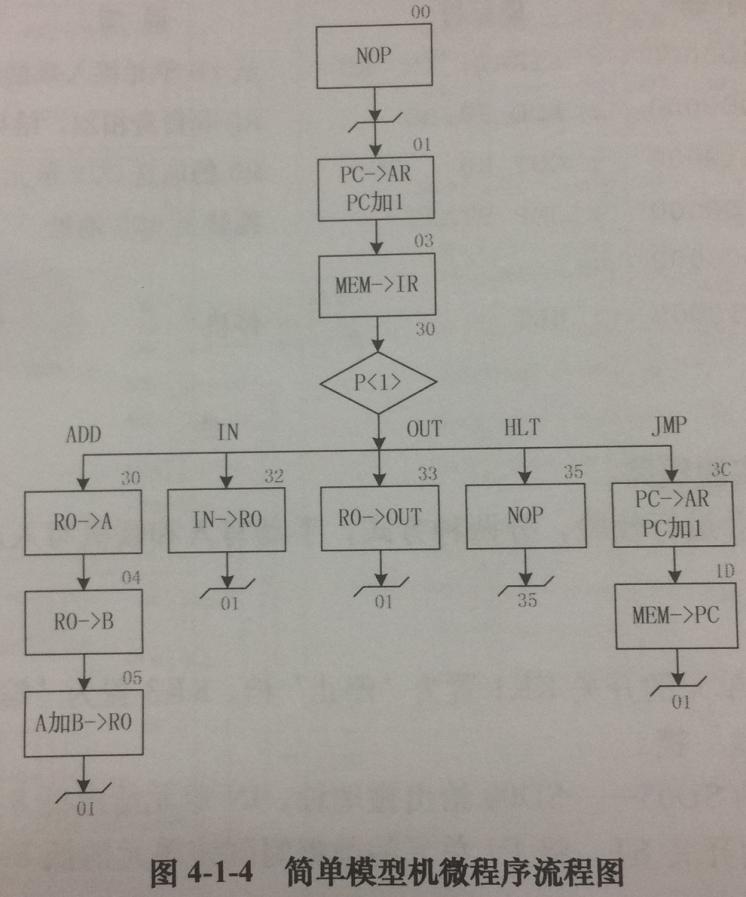
其中JMP为双字节指令，其余均为单字节指令，\*\*\*\*\*\*\*\*为addr 对应的二进制地址码。微程序控制器实验的指令是通过手动给出的，现在要求CPU自动从存储器读取指令并执行。根据以上要求，设计数据通路图，如图4-1-3所示。



本实验在前一个实验的基础上增加了三个部件，一是PC（程序计数器），另一个是AR（地址寄存器），还有就是MEM（主存）。因而在微指令中应增加相应的控制位，其微指令格式如表4-1-1所示。



系统涉及到的微程序流程见图4-1-4所示，当拟定“取指”微指令时，该微指令的判别测试字段为P<1>测试。指令译码原理见图3-2-3所示，由于“取指”微指令时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。本机用指令寄存器的高6位（IR7-IR2）作为测试条件，出现5路分支，占用5个固定微地址单元，剩下的其他地方就可以一条微指令占用控存一个微地址单元随意填写，微程序流程图上的单元地址为16进制。



当全部微程序设计完毕后，应将每条微指令代码化，表4-1-2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”。

设计一段代码程序，要求从IN单元读入一个数据，存于R0，将R0和自身相加，结果存于R0，再将R0的值送OUT单元显示。

设计一段机器程序，要求从IN单元读入一个数据，存于R0，将R0和自身相加，结果存于R0，再将R0的值送OUT单元显示。

根据要求可以得到如下程序，地址和内容均为二进制数。

|  |  |  |  |
| --- | --- | --- | --- |
| 地址 | 内容 | 助记符 | 说明 |
| 00000000 | 01000000 | ;START : IN R0 | 从IN单元读入数据送R0 |
| 00000001 | 00000000 | ;ADD R0,R0 | R0和自身相加，结果送R0 |
| 00000010 | 00110000 | ;OUT R0 | R0的值送OUT单元显示 |
| 00000011 | 11100000 | ;JMP START | 跳转至00H地址 |
| 00000100 | 00000000 | ; |  |
| 00000101 | 01010000 | ;HLT | 停机 |

1. 实验步骤
2. 按图4-1-5-A连接实验线路。
3. 写入实验程序，并进行校验，份两种方式，手动写入和联机写入。
4. 手动写入和校验
5. 手动写入微程序

①将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘编程’档，KK4置为‘控 存’档，KK5置为‘置数’档。

②使用CON单元的SD05----SD00给出微地址，IN单元给出低8位应写入的数据，连续两 次按动时序与操作台的开关ST，将IN单元的数据写入到该单元的低8位（观察MC单元

MC0与M7~M0）。

③将时序与操作台单元的开关KK5置为‘加1’档。

④IN单元给出中8位应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的 数据写到该单元的中8位（观察MC单元MC1与M15~M7）。IN单元给出高8位应写入

的 数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写入到该单元的高8位（观察MC单元MC2与M23~M16）。

⑤重复①、②、③、④四步，将表4-1-2的微代码写入2816芯片中。

1. 手动校验微程序

①将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘校验’档，KK4置为‘控 存’档，KK5置为‘置数’档。

②使用CON单元的SD05----SD00给出微地址，IN单元给出低8位应写入的数据，连续两 次按动时序与操作台的开关ST，将IN单元的数据指示灯M7~M0显示该单元的低8位

MC0=1。

③将时序与操作台单元的开关KK5置为‘加1’档。

④连续两次按动时序与操作台的开关ST，MC单元的数据指示灯M15----M8显示该单元的中8位MC1=1，MC单元的数据指示灯M23----M16显示该单元的高8位MC2=1。

⑤重复①、②、③、④四步，完成对微代码的校验。如果校验出微代码的写入错误，重新写入、校验，直至确认微代码的输入无误为止。

1. 手动写入机器程序

①将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘编程’档，KK4置为‘控 存’档，KK5置为‘置数’档。

②使用CON单元的SD07----SD00给出微地址，IN单元给出该单元应写入的数据，连续两 次按动时序与操作台的开关ST，将IN单元的数据写入到该存储器单元（观察CPU内总线

D7~D0与IN单元的数据一致，地址总线A7~A0与SD07~SD00一致）。

③将时序与操作台单元的开关KK5置为‘加1’档。

④IN单元给出下一地址（地址自动加1）应写入的数据，连续两次按动时序与操作台的开关ST，将IN单元的数据写到该单元的中。然后地址又自动加1，只需在IN单元输入后续地址数据，连续两次按动时序与操作台的开关ST，即完成对该单元的写入。

⑤重复①、②、③、④四步，将所有机器指令写入主存芯片中。

1. 手动校验机器程序

①将时序与操作台单元的开关KK1置为‘停止’档，KK3置为‘校验’档，KK4置为‘控 存’档，KK5置为‘置数’档。

②使用CON单元的SD07----SD00给出微地址，连续两次按动时序与操作台的开关ST，

CPU内总线的数据指示灯D7~D0显示该单元的数据。

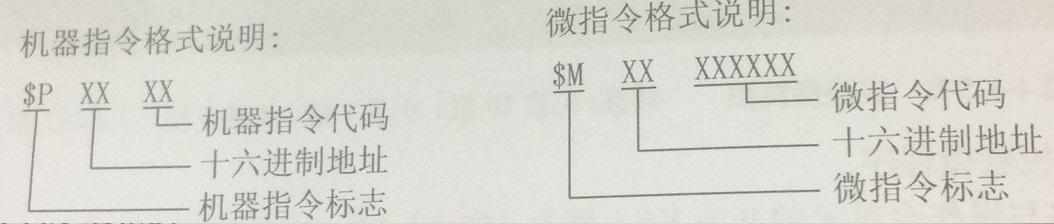
③将时序与操作台单元的开关KK5置为‘加1’档。

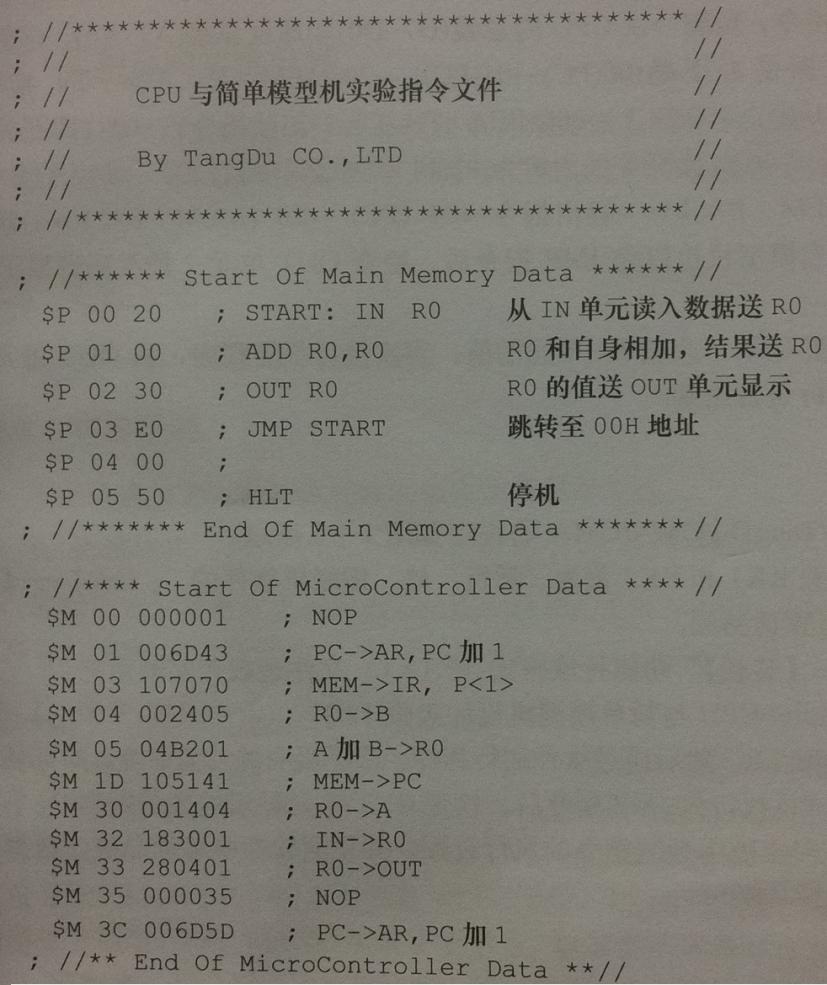
④连续两次按动时序与操作台的开关ST，地址自动加1，CPU内总线的数据指示灯D7~D0显示该单元的数据。此后每两次按动时序与操作台的开关ST，地址自动加1，CPU内总线的数据指示灯D7~D0显示该单元的数据，继续进行该操作，直至完成校验，如发现错误，返回写入，然后校验，直至确认输入的所有指令准确无误。

⑤重复①、②、③、④四步，完成对指令码的校验。如果校验出指令码写入错误，重新写入、校验，直至确认指令码的输入无误为止。

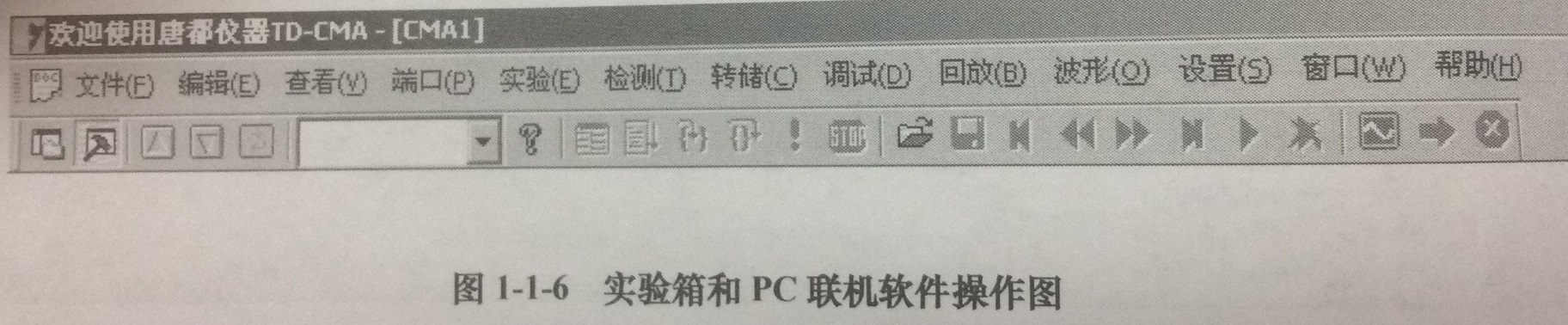
1. 联机写入和校验

联机软件提供了微程序和机器程序下载功能，以替代手动读写微程序和机器程序，但是微程序和机器程序得以指定的格式写入到以TXT为后缀的文件中，微程序和机器程序的格式如下：

 本次实验程序如下，程序中分号‘;’为注释符，分号后面的内容在下载时将被忽略掉：



单击【开始】/【程序】/TangDu/CMA/CMA的程序如图1-1-6所示。



选择联机软件的“【转储】-----【装载】”功能，在打开文件对话框中选择上面所保存的文件，软件自动将机器程序和微程序写入指定单元。

选择联机软件的“【转储】-----【刷新指令区】”可以读出下位机所有机器指令和微指令，并在指令区显示，对照文件检查微程序和机器程序是否正确，如果不正确，则说明写入操作失败，应重新写入，可以通过联机软件单独修改某个单元的指令，以修改指令为例，先用鼠标左键单击指令区的‘微存’，然后再单击需修改单元的数据，此时该单元变为编辑框，输入6位修改数据并回车，编辑框小时，并以红色显示写入的数据。

1. 运行本程序

方法一：本机运行

将时序与操作台单元的开关KK1、KK3置为‘运行’档，按动CON单元的总清按钮CLR，将使程序计数器PC、地址寄存器AR和微程序地址为0011，程序可以从头开始运行，暂存器A、B，指令寄存器IR和OUT单元也会被清零。

将时序与操作台单元开元KK2置为‘单步’档，每按动一次ST按钮，即可单步运行一条微指令，

对照微程序流程图（图4-1-4），观察微地址显示灯（MC单元MA5~MA0，与M23~M0)是否和流程一致。每运行完一条微指令，观测一次CPU内总线（D7~D0）和地址总线（A7~A0），对照数据通路图，分析总线上的数据是否正确（00（PC）→01（PC））→20（MEM→P<1>－IN指令）→02（读开关的内容为02到R0中）→02（PC）→00（MEM－P<1>－加法指令）→02（读开关的内容为02到R0中送A寄存器）→02（读开关的内容为02到R0中送B寄存器）→04（A+B=04到R0中）→03（PC）→30（MEM－P<1>－输出指令）→04（PC）→E0（MEM－P<1>－JMP指令）→05（PC）→00（PC）。当模型机执行完JMP指令后，检查OUT单元7段数码管显示的数是否为IN单元值的2倍。

按下CON单元的总清按钮CLR，改变IN单元的值，再次执行机器程序，从OUT单元7段数码管显示的数判别程序执行是否正确。

方法二： 联机运行

单击【开始】/【程序】/TangDu/CMA/CMA的程序如图1-1-6所示。

将时序与操作台单元的开关KK1和KK3置为‘运行’档，用联机软件的“【实验】-----【简单模型机】”打开简单模型数据通路图。

用联机软件的“【转储】-----【装载】”功能将该格式（\*.TXT）文件装载入实验系统（TangDu/CMA/CMA/Sample/CPU与简单模型机设计实验.Txt）。

按动CON单元的总清按钮CLR，然后通过软件运行程序，选择相应的功能命令，即可联机运行、监控、调试程序，当模型机执行完JMP指令后，检查OUT单元显示的数是否为IN单元值得2倍。在数据通路图和微程序流中观察指令的执行过程，并观测软件中地址总线、数据总线以及微指令显示和下位机是否一致。

1. 实验结果

输出的数值是输入的数值的两倍

1. 实验体会

熟悉了一个简单CPU的组成原理，初步体验了机器指令和微指令在指令文件中的书写方法。