西安科技大学

计算机学院实验报告

班级：网络工程1801 姓名：吴斌 学号：18408020129

时间：二零二零年八月 评分： 教师签名：

实验名称：CPU与简单模型机设计实验

实验目的;

1. 掌握一个简单CPU的组成原理。
2. 在掌握部件单元电路的基础上，进一步构造一台基本模型计算机。
3. 定义五条机器指令，编写相应的微程序，并上机调试掌握整机概念。

实验原理：

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）、微程序控制器（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5-1所示。这个CPU在写入相应的微指令后，就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。

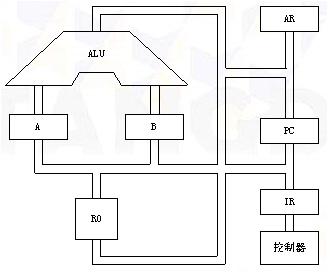


图5-1 基本CPU构成原理图

程序计数器（PC）原理图如图5-2，其余部件在前面的实验中都已用到，在此不再讨论。

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相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。

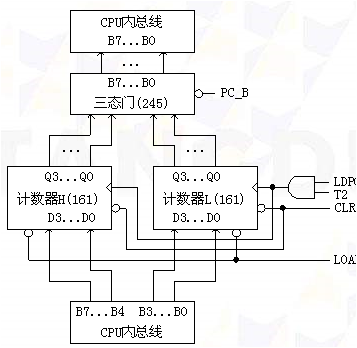
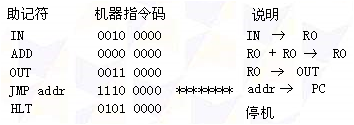


图5-2 程序计数器(PC)原理图

本模型机和前面微程序控制器实验相比，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）：



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5-3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5-1所示。

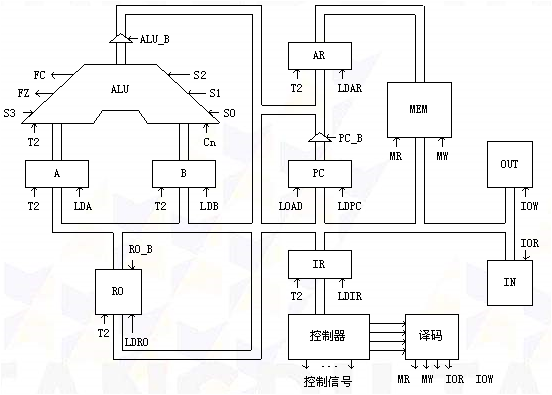
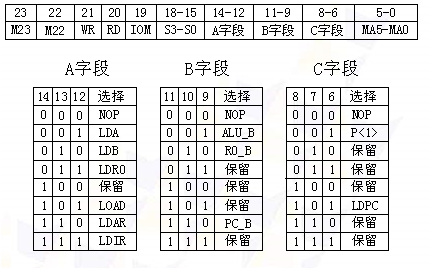


图5-3 数据通路图

表5-1 微指令格式



系统涉及到的微程序流程见图5-4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3-4所示，由于“取指”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。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，剩下的其它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。

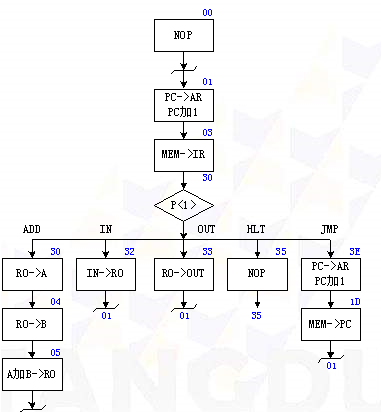


图5-4 简单模型机微程序流程图

当全部微程序设计完毕后，应将每条微指令代码化，表5-2即为将图5-4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5-2 二进制微代码表

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | 010 | 000 | 000101 |
| 05 | 04 B2 01 | 00000 | 1001 | 011 | 001 | 000 | 000001 |
| 1D | 10 51 41 | 00010 | 0000 | 101 | 000 | 101 | 000001 |
| 30 | 00 14 04 | 00000 | 0000 | 001 | 010 | 000 | 000100 |
| 32 | 18 30 01 | 00011 | 0000 | 011 | 000 | 000 | 000001 |
| 33 | 28 04 01 | 00101 | 0000 | 000 | 010 | 000 | 000001 |
| 35 | 00 00 35 | 00000 | 0000 | 000 | 000 | 000 | 110101 |
| 3C | 00 6D 5D | 00000 | 0000 | 110 | 110 | 101 | 011101 |

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。



实验内容与结果：

1. 按图5-5连接实验线路。
2. 写入实验程序，并进行校验，分两种方式，手动写入和联机写入。

1) 手动写入和校验

(1) 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 单元的数据写到该单元的低8 位。
  3. 将时序与操作台单元的开关KK5 置为‘加1’档。
  4. IN 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。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 单元的数据写到该单元的高8 位。
  5. 重复①、②、③、④四步，将表5-2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，连续两次按动时序与操作台的开关 ST，MC 单元的指数据指示灯 M7——M0 显示该单元的低8位。
   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 单元的指数据指示灯 M15——M8 显示该单元的中8 位，MC 单元的指数据指示灯 M23——M16 显示该单元的高8 位。
   5. 重复①、②、③、④四步，完成对微代码的校验。如果校验出微代码写入错误，重新写入、校验，直至确认微指令的输入无误为止。

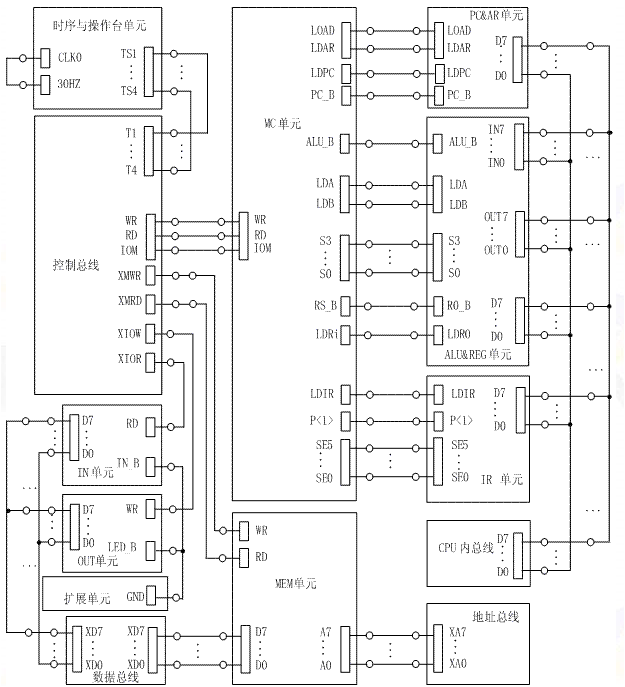


图5-5 实验接线图

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‘主存’档，KK5 置为‘置数’档。
   2. 使用CON 单元的SD07——SD00 给出地址，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 单元的数据写到该存储器单元。
   3. 将时序与操作台单元的开关KK5 置为‘加1’档。
   4. IN 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中。然后地址会又自加1，只需在IN 单元输入后续地址的数据，连续两次按动时序与操作台的开关ST，即可完成对该单元的写入。
   5. 亦可重复①、②两步，将所有机器指令写入主存芯片中。
2. 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 ‘主存’档，KK5 置为‘置数’档。

* 1. 使用CON 单元的SD07——SD00 给出地址，连续两次按动时序与操作台的开关ST， CPU内总线的指数据指示灯 D7——D0 显示该单元的数据。
  2. 将时序与操作台单元的开关KK5 置为‘加1’档。
  3. 连续两次按动时序与操作台的开关ST，地址自动加1，CPU 内总线的指数据指示灯 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 内总线的指数据指示灯 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，则返回写入，然后校验，直至确认输入的所有指令准确无误。
  4. 亦可重复①、②两步，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。

2) 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，以代替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：

CPU 与简单模型机实验指令文件

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

$P 00 20 ; START: IN R0 从IN 单元读入数据送R0

$P 01 00 ; ADD R0,R0 R0 和自身相加，结果送R0

$P 02 30 ; OUT R0 R0 的值送OUT 单元显示

$P 03 E0 ; JMP START 跳转至00H 地址

$P 04 00 ;

$P 05 50 ; HLT 停机

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

; //\*\*\*\* 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 ; R0->B

$M 05 04B201 ; A 加B->R0

$M 1D 105141 ; MEM->PC

$M 30 001404 ; R0->A

$M 32 183001 ; IN->R0

$M 33 280401 ; R0->OUT

$M 35 000035 ; NOP

$M 3C 006D5D ; PC->AR,PC 加1

; //\*\* End Of MicroController Data \*\*//

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的机器指令和微指令，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，以修改微指令为例，先用鼠标左键单击指令区的‘微存’TAB 按钮，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 位数据并回车，编辑框消失，并以红色显示写入的数据。

3. 运行程序 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和微程序地址清零为00H，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的开关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，观察微地址显示灯是否和流程一致。每运行完一条微指令，观测一次CPU 内总线和地址总线，对照数据通路图，分析总线上的数据是否正确。 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 单元显示的数判别程序执行是否正确。

方法二：联机运行 将时序与操作台单元的开关KK1 和KK3 置为‘运行’档，进入软件界面，选择菜单命令 “【实验】—【简单模型机】”，打开简单模型机数据通路图。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单元值的2倍。在数据通路图和微程序流中观测指令的执行过程，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。

总结

掌握了一个简单CPU的组成原理。在掌握部件单元电路的基础上，进一步构造了一台基本模型计算机。定义了五条机器指令，编写了相应的微程序，并上机调试掌握了整机概念。