|  |
| --- |
| **北 京 邮 电 大 学**  **实 验 报 告**  **课程名称： 计算机组成原理实验课**  **实验名称: 微程序控制器及CPU取指综合测试报告**  **计算机学院 2022211304班 姓名：赵宇鹏**  **教师：杨秦 成绩\_\_\_\_\_\_**  **2024年5月16日** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **实验4：微程序控制器部分测试实验**  芯片及硬件电路分析：    微程序控制器的硬件原理框图如上图所示，在数据通路实验的基础上增加了控制信号转换模块，通过选中微程序控制模块切换至微程序模式，在微程序模式下，系统直接把指令集合在一起成为微程序，系统通过调用微程序来执行用户的命令，微程序模式采用SEL3-SEL0通过2选1选择器来对4个寄存器进行选择。微程序控制器主要有：产生数据通路操作所需控制信号。存储逻辑型的微程序；时序逻辑型的硬布线。控制部件→执行部件的控制命令：微命令；执行部件进行的操作：微操作；在一个CPU周期中，一组实现一定操作功能的微命令的组合：微指令。微指令序列构成微程序。执行当前的微指令时，必须指出后继微地址，以便当前微指令执行完毕后，取出下一条微指令。  控制存储器由5片58C65组成，58C65是一种8K×８位的E2PROM器件，TEC-8使用其中64个字节作为控制存储器，微地址µ A5-µ A0。微地址寄存器µ AR由D触发器74LS174LS组成。当按 下复位按钮CLR时，产生的信号CLR#(负脉冲)使微地址寄存器复位，µ A5-µ A0为00H，在一条微指令结束时，用T3的下降沿将微地址转移逻辑产生的下条微指令地址Nµ A5、Nµ A4-T~Nµ A0-T写入微地址寄存器。微地址转移逻辑由若干与门和或门组成，实现“与 -或”逻辑。  **微指令格式分析：**  微指令字长40 位，顺序字段11 位（判别字段P4～P0，后继微地址NμA5～NμA0），控制字段29 位，微命令直接控制。    Nµ A5-Nµ A0为下址，在微指令顺序执行的情况下为下一条微指令的地址，高三位自动被忽略为0，STOP控制在T3结束后时序发生器停止输出节拍脉冲T1、T2、T3，LIAR控制PC7-PC0在T3上升沿时写入中断地址寄存器IAR，INTDI为1时置允许中断标志为0，禁止计算机响应中断请求，INTEN为1时则置允许中断标志为1，允许计算机响应中断请求，IABUS为1时将中断寄存器IAR中的地址送往数据总线DBUS，PCADD为1时将当前PA值加上相对转移量生成新的PC。下图便是微程序的执行流程图。    P0,P1,P2,P3,P4分别为判断状态的条件，当P4等于1时说明系统发出了中断信号，要执行中断操作。以上信号都由系统执行微指令直接发出。  **流程图测试详解：**  要对微程序控制器进行测试，首先按照所给实验接线图示接好线，接好线后，将编程开关拨到“正常”位置，控制转换开关拨到“微程序”位置，合上电源，按CLR#按钮，使TEC-8实验系统处于初始状态，再将DP置1，用单微指令方式跟踪控制台操作，通过观看µ A5-µ A0的值来查看系统进入了哪一条微指令。  1、测试开始，首先需要先进行写存储器操作，将SWC置0，SWB置0，SWA置1，按下QD，此时对照流程图我们可以看到此时系统已经进入微地址为03所对应的微指令上，SBUS和LAR打开，说明此时要求我们给AR输入地址，在SW7-SW0中设置存储器地址，按QD按钮将此地址打入AR    我们可参照以上程序来输入指令，则此时我们初始需打入的地址为00H，按下QD后系统进入02，SBUS，MEMW打开，给AR对应地址赋值，在存储器中存储我们输入的指令，第一条指令对应16进制机器码为53H，在SW7-SW0中设置相应值按下QD打入存储器，ARINC值为1，AR地址自加一，因此无需我们对AR地址进行改变，每次输入指令后AR都会自加一，之后重复之前操作依此打入4CH，57H，21H，测试所需指令存储完毕，按下CLR#返回00状态。  2、我们可以看到指令里面对寄存器R3所存指针进行操作，因此我们需要给R3赋一个合理的初值，对照流程图，将SWC置1，SWB置0，SWA置0，按下QD，系统进入09，即开始执行写入寄存器操作，写入按顺序为R0，R1，R2，R3，由于R0，R1的值都会被覆盖，因此无需赋值，按累计3次QD后选中R3进行赋值，由于本次测试存储指令较少，我们就给R3赋值02H，则[R3]值为57H，按下CLR#返回00状态。  3、现在我们开始执行我们之前存入的指令，将SWC置0，SWB置0，SWA置0，按下QD，系统进入01，此时系统会从存储器中获取指令等待执行，P1灯亮，按下QD后判别P1通过，系统再通过机器指令的前四位来判断进入哪一个微指令，由于我们第一条指令前四位为0101，系统便会进入25，从相应寄存器中读取所需要的值，再按下QD进入0E，将读出的值57H存进R0寄存器中，观察D7-D0，我们也可以看到相应的数值，此外我们也可以通过PC7-PC0以及INS7-INS0观察当前执行指令的地址以及下一条指令的机器码，再按下QD，此时P4为0，没有中断信号，则返回01指令，重复以上操作。  4、在执行完以上四条指令后按下CLR#，返回初始状态，再将SWC置0，SWB置1，SWA置1，按下QD，系统进入07读寄存器状态，A7-A0显示R0的值为36H，B7-B0显示R1的值21H，再按下QD进入06，A7-A0显示R2，B7-B0显示R3值为03H，程序测试完毕，实验4完成。  **实验5：模型机CPU取指测试**  整体框图分析：    由于在实验4时已经详细分析了所有流程，在此处便简略介绍一下，在初始时系统处于00状态，通过控制开关SWC，SWB，SWA的值选择进入的操作，由µ A5~µ A0来显示下一条微指令的地址，若我们选择执行000操作，即取指执行操作，按下QD后则会进入01微指令，对存储器进行取指操作，再按下QD后系统通过取得指令的前四位来判断执行哪一条微指令，执行完微指令后返回到P4判断，若无中断信号，则P4判别字段为0，返回01操作，取出并执行下一条指令，若按下CLR#，则返回00初始微指令。  测试程序分析：  测试程序如下图所示    ，00H指令为赋值指令，将R3所存指针的值赋值给R0，此后01H指令，R3自加一为，02H再将R3所存指针的值赋值给R1，03H对应指令为R0 = R0 – R1，04H为判断跳转指令，若R0 = R0 – R1 后R0的值为0则跳转到0BH所对应指令处继续执行，否则继续执行05H指令，将R0的值存到R2所存地址所指向的空间，06H，R3自加1，07H再将R3所存指针的值赋值给R0，08H指令R0 = R0 + R1，09H又为一个跳转指令，若08H R0 = R0 + R1发生进位则跳转至0CH处执行操作，0AH，R2所存值自加一，0BH，将R2所存值存入R2所存指针中，0CH，将R0和R1的值进行一个与操作，0DH，输出R2，0EH，暂停操作，之后则为我们存入的值而非机器代码，因此当程序运行到STP时就应该执行CLR#操作，以免系统将后续地址所存数值当作机器指令来执行。  测试过程及结果详解：  将单拍开关DP设置为1，使程序在单微指令下运行。  按参考接线接好所有线，将编程开关拨到“正常”位置,控制转换开关拨到“微程序”位置，合上电源,按CLR#按钮,使TEC-8实验系统处于初始状态。  写存储器(WRM)：按下复位按钮CRL#，置SWC=0，SWB=0，SWA=1。①在SW7~SW0中设置存储器地址00H，采用单拍或连续方式，按QD按钮将此地址打入A②在SW7~SW0置好数据，按QD按钮，将数据写入AR指定的存储器单元，这时AR加1。③返回②。依次进行下去，直到按复位按钮CRL#为止。实现对RAM的手动写入，将所有的程序和需要的数值写入RAM。每一控制台指令操作完毕应使系统复位，然后转换控制台指令。  写寄存器：按下复位按钮CRL#，置SWC=1，SWB=0，SWA=0。按下QD选中R2，在SW7~SW0置好数据12H，按QD按钮，将数据写入R2，再按下QD，选中R3，在SW7~SW0置好数据0FH，按QD按钮，将数据写入R3，按下复位按钮CRL#。  执行指令：按复位按钮CLR，复位程序计数器PC为00H。将模式开关设置为 SWC=0、SWB=0、SWA=0，准备进入程序运行模式。按一次QD按钮，进入程序运行。每按一次QD按钮，执行一条微指令，直到程序执行到STP指令，按下CLR#复位。  读寄存器：按下复位按钮CRL#，置SWC=0，SWB=1，SWA=1按下QD进入读寄存器操作，得R0值为02H，R1值为23H，R2值为12H，R3值为11H。  读存储器：按下复位按钮CRL#，置SWC=0，SWB=1，SWA=0，按下QD，进入读取存储器模式，在SW7~SW0置好地址12H，按下QD将地址打入，此时D7-D0显示值为62H。  结果分析：R2值初始为12H，R3初始值为0FH，第一条指令R0 = 85H，第二条指令R3 = 10H，第三条指令，R1 = 23H，第四条指令，R0 = 62H不等于0，则继续执行第五条指令，不执行跳转操作，第五条指令，12H地址所存数值变为62H ,第六条指令R3 = 11H，第七条指令，R0 = 0EFH，第八条指令，加操作后R0 = 12H,发生进位，跳转至0CH处，执行与操作，则R0 = 02H，之后输出R2，暂停，按下CLR#复位。  因此，通过计算R0，R1，R2，R3预期值为02H，23H，12H，11H，且地址12H所存数值为62H，最终输出结果符合预期计算。  实验结论：  在本次实验中，我们进行了模型机CPU取指测试实验，探究了CPU组成与机器指令执行流程。  CPU基本功能：  1、程序的顺序执行：CPU保证指令按预定顺序执行，从而实现程序的逻辑。  2、管理控制操作：通过控制信号协调CPU内部及其与其他部件间的工作。  3、操作和执行定时：时序发生器产生的时序信号，确保各操作在正确时间点执行。  4、数据加工：运算器进行算术和逻辑运算，以完成数据处理任务。  控制器组件的作用：  1、程序计数器（PC）：跟踪将要执行的指令地址，保证指令按顺序执行。  2、指令寄存器（IR）：保存当前执行的指令，为译码提供依据。  3、指令译码器：将指令转换为具体的控制信号，指导接下来的操作。  4、时序发生器：生成时序信号，协调CPU内部操作的时序。  5、操作控制器：根据译码结果生成具体控制信号，管理各组件的操作和数据传输。  运算器组件的作用：  1、算术逻辑单元（ALU）：执行具体的算术和逻辑运算，是数据加工的核心。  2、累加器（AC）：存储运算的中间结果，便于连续运算操作。  3、数据缓冲寄存器（DR）：用于存储从内存或输入/输出设备接收的数据。  4、状态条件寄存器（PSW）：保存运算结果的状态，如运算结果是否为零或是否产生  位。  指令执行过程：  1、取指：从内存中获取指令，并将其存入指令寄存器。  2、译码：指令译码器将指令译码为微操作。  3、生成控制信号：控制器根据译码结果生成控制信号，指导下一步操作。  4、执行指令：运算器根据控制信号执行相应运算，结果存储在指定寄存器中。  微程序与机器指令周期：  每条机器指令对应一个微程序，通过多个微指令周期完成其执行。机器指令周期是由若干微指令周期组成的执行过程。  控制部件与执行部件的协同工作：  1、控制部件（控制器）生成控制信号，协调CPU及其他部件的操作。  2、执行部件（运算器）根据控制信号执行具体运算和数据处理任务。  通过本次实验，我理解了控制器根据不同的指令类型，从微程序存储器中读取相应的微指令，生成控制信号，驱动数据通路工作的原理，对计算机取指，执行操作的流程有了较深的了解。  **实验6：中断测试**  硬件中断原理分析：  TEC-8的中断系统只支持单级中断、单个中断请求，有中断屏蔽功能。系统有2条指令用于允许和屏蔽中断（DI关中断指令、EI开中断指令）。在时序发生器中，设置了一个允许中断触发器EN\_INT（DI or !EI），当它为1时，允许中断，当它为0时，禁止中断发生。复位脉冲CLR#使EN\_INT复位为0。      中断地址寄存器IAR是74LS374。当信号LIAR为1时，在T3的上升沿，将PC保存在IAR中。当信号IABUS为1时，IABUS中保存的PC送数据总线DBUS。由于本实验系统只有一个断点寄存器而无堆栈，因此仅支持一级中断而不支持多级中断。中断向量即中断服务程序的入口地址，本实验系统中由数据开关SD7-SD0提供。除指令EI、DI外，每条指令执行过程的最后一条微指令都包含判断位P4，用于判断有无中断发生，根据中断信号INT是否为1决定微程序分支。如果信号INT为1，则转微  地址11H，进入中断处理；如果信号INT为0，则转微地址01H，继续取下一条指令然后执行。NT为1转到微地址11H，该微指令产生INTDI信号，禁止新的中断发生，产生LIAR信号保存当前地址 (断点寄存器），产生STOP信号，等待手动设置中断向量（数据开关SD7~SD0设置中断地址），机器将中断向量读到PC后，转到中服务程序继续执行。执行指令IRET，从中断地址返回，该指令产生IABUS信号，恢复断点地址，产生信号LPC，将断点从数据总线装入PC，恢复被中断的程序。发生中断时，关中断由硬件负责。而中断现场(包括4个寄存器、进位标志C和结果为0标志Z)的保存和恢复由中断服务程序完成。中断服务程序的最后两条指令一般是开中断指  令EI和中断返回指令IRET。为了保证从中断服务程序能够返回到主程序，EI指令执行后，不允许立即被中断。因此，EI指令执行过程中的最后一条微指令中不包含P4判别位。     |  | | --- | | 1101 0000 | | 0100 0000 | | 0100 0000 | | 0100 0000 | | 0100 0000 | | 0100 0000 | | 0100 0000 | | 0100 0000 | | 0100 0000 | | 1001 0100 | | 0011 0000 | | 1101 0000 | | 1011 0000 |     测试程序分析：  00H为打开中断操作，允许中断发生，之后01H到08H执行R0自加一操作，09H执行跳转操作，跳转至R1所存地址，只要R1内存地址为01H-08H即可，这样便可形成循环，只有中断和CLR#复位才可打断，中断发生后，输入地址45H，则执行R0 = R0+R0操作，再打开中断，返回断点处，继续循环。  执行及结果分析：  接好线后,将编程开关拨到“正常”位置,控制转换开关拨到“微程序”位置，合上电源,按CLR#按钮,使TEC-8实验系统处于初始状态。  为了保证此程序能够循坏执行，应当将R1预先设置为01H。R0的初值设置为0。  将TEC-8连接成一个完整的模型计算机。将主程序和中断服务程序装入存储器，开关DP设置为连续运行方式(DP=0)，复位系统后按QD按钮，启动程序从00H开始执行。 PULSE按钮，产生一个中断请求信号PULSE ，中断主程序的运行。此时A7-A0灯会显示R0的值，PC7-PC0显示断点位置03H，将单拍开关DP设置为单拍方式(DP=1)，在数据开关上设置中断服务程序的入口地址45H。按QD按钮进入，一步步执行中断服务程序，直到返回到断点为止，返回后程序会继续执行循环程序，结果符合预期。  实验结论：  实验成功验证了TEC-8中断系统的功能，证明了单级中断请求和中断屏蔽功能的有效性，中断处理过程中的地址保存与恢复机制确保了程序在中断发生前后的连续性。实验结果与预期完全一致，说明TEC-8中断系统能够稳定地处理中断请求，并在中断服务程序执行完毕后正确返回主程序。在实验中，由于在中断地址的输入时产生了错误，同时忘记把单拍DP置1，导致程序在中断后运行出错，执行了系统随机分配的指令，导致寄存器和存储器内数据丢失，因此，我学习到了在实验中要注重细节，不能急躁，拥有充足的耐心才能成功完成实验。通过本次实验，我深入理解了中断系统的工作机制及其在计算机系统中的重要作用，了解了微程序控制器与中断控制器协调的基本原理，掌握了中断子程序和一般子程序的本质区别，掌握了中断的突发性和随机性。为进一步学习和应用中断技术打下了坚实的基础。  **实验心得**  在实验6中，由于在中断地址的输入时产生了错误，同时忘记把单拍DP置1，导致程序在中断后运行出错，执行了系统随机分配的指令，导致寄存器和存储器内数据丢失，因此，我学习到了在实验中要注重细节，不能急躁，拥有充足的耐心才能成功完成实验。通过本次实验，我系统地了解了微程序控制器的工作机制及其在CPU取指和执行指令过程中的关键作用。实验结果验证了理论知识的正确性，增强了我对计算机内部工作原理的理解和掌握。同时，实验过程中遇到的问题和解决方法进一步提升了我的实际操作能力和分析问题的能力。 |