|  |
| --- |
| **北 京 邮 电 大 学**  **实 验 报 告**  **课程名称\_\_\_\_计算机组成原理实验\_\_**  **实验名称\_\_\_实验4--6 + 拓展实验\_\_\_\_**  **\_计算机\_学院\_\_314\_\_班 姓名\_\_魏生辉\_\_\_\_**  **教师\_\_\_刁婷老师\_\_\_\_\_ 成绩\_\_\_\_\_\_**  **\_\_2025\_\_年\_5\_月\_18\_日** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **实验四、微程序控制器实验**   1. **实验任务及目的**   **①实验任务:**  （1）正确设置操作模式开关SWC、SWB、SWA，用单微指令方式（DP=1）**跟**  **踪控制台操作读寄存器、写寄存器、读存储器、写存储器的执行过程**。    （2）记录每一步操作的微地址、判别位、下地址等信号的值。  正确设置指令操作码IR7-IR4，用单微指令方式（DP=1）**跟踪ADD、SUB、**  **AND、JC、LD指令的执行过程。**    （3）记录每一步操作的微地址、判别位、下地址等信号的值。  对于JC指令，跟踪C=1和C=0两种情况  **②实验目的:**  （1）掌握微程序控制器的原理；  （2）掌握TEC-Plus/TEC-8模型计算机中微程序控制器的实现方法，微地址转移逻辑的实现方法   1. **实验电路分析**     **图 1微程序控制电路**  **①微程序控制电路的理解与分析**  **（1）控制存储器**：  电路上半部分的CM4、CM3、CM2、CM1、CM0五个芯片共同构成了控制存储器。这些存储器用于存放微指令。  接下来我们分析一下微指令格式：  每条微指令为40位宽。这40位被划分为两个主要字段：  **控制字段** (Control Field)：占据29位，负责产生控制计算机执行单元进行各种操作的控制信号。图中CM4、CM3、CM2以及CM1的部分I/O口（如I/O7至I/O0）连接到这些控制信号输出，例如A-SELCTL、A-LDAR、A-MEMW等。  **顺序字段**：占据11位，用于决定下一条将要执行的微指令的地址。这11位中的一部分（如图中A5-A0连接到CM芯片的地址输入端）直接或间接参与下一微地址的形成。  **存储分配**：CM0存储微指令的最低8位，CM1存储接下来的8位，以此类推，CM4存储微指令的最高8位。这是一种分布式存储，共同构成一个完整的40位微指令字。  **（2）微地址寄存器**：  中间部分的REG6芯片构成了微地址寄存器:  该寄存器用于暂存当前正在执行或即将从控制存储器中读取的微指令的地址。控制存储器的地址输入端 (A0-A5) 与微地址寄存器的输出端 (µA0 - µA5) 相连接，表明微地址寄存器的内容直接决定了从控制存储器中选取的微指令。  下面我们来理解一下更新机制：在一条微指令执行完毕并准备取下一条微指令时，由时钟信号T3的下降沿触发，微地址转移逻辑产生的下一条微指令地址 (NµA5，NµA4-T ~ NµA0-T) 被装入微地址寄存器。CLR# 输入用于异步清零微地址寄存器，R输入用于同步复位或置位。  **（3）微地址转移逻辑**：  该逻辑单元由一系列与门（AND）和或门（OR）组合而成。其核心任务是根据当前微指令的顺序字段、外部状态/条件信号以及特定控制信号，计算并确定下一条微指令的地址。下面是具体的分析。  ②**微地址转移逻辑的理解和分析**  微地址转移逻辑中，最关键的输入信号包括来自当前微指令的后继微地址位 N**μ**A0～N**μ**A5、判别测试位 P0～P3，以及反映处理器状态和模式的信息源。这六个位宽字段为逻辑提供了基础地址值，而四个位的判别信号则决定了地址生成的方式：当 P0 为 1 时，模式开关 SWA、SWB、SWC 的组合会插入到低位，实现在不同运行模式下的微程序跳转；当 P1 为 1 时，指令寄存器 IR 的高四位（IR7～IR4）用作低位地址，实现基于机器指令操作码的自动分支；当 P2 或 P3 为 1 时，进位标志 C 或零标志 Z 会替换最低位，以支持条件为“有进位”或“结果为零”时的微程序跳转。除非有对应的判别位被置 1，否则所有低位均由原始的 N**μ**A 字段直接传递。  地址更新逻辑采用了典型的**与-或**组合电路结构，将上述各类信号有序汇聚到每一个地址位的输出端。具体而言，N**μ**A5 和 N**μ**A4 始终直接通过或门输出，不受任何判别信号的影响。而对中低位 **μ**A3～**μ**A1，则通过一组与门将 P0 与 SW 信号、P1 与 IR 位分别相与，再通过或门与 N**μ**A 原值合并，这样可在条件触发时用开关或指令码覆盖原地址，否则沿用默认字段。当涉及最低位 **μ**A0 时，电路进一步并入了 P2∧C 与 P3∧Z 两路控制，这就使得“进位”或“零”条件下的跳转也能被准确响应。  以 P1=1、其余 P=0 的典型情境为例：假设当前微指令的 N**μ**A 字段为 100000₂（即 0x20），且指令寄存器高四位 IR7～IR4 为 0011₂（即 0x3）。此时，μA5 和 μA4 仍为 N**μ**A5-4 的“10”；**μ**A3、**μ**A2、**μ**A1 和 **μ**A0 则分别由 IR7～IR4 的“0、0、1、1”填充。综合后得到新的微地址为 10 0011₂（0x23），于 T3 下降沿并行写入微地址寄存器，下一周期由地址 0x23 取出微指令，实现了“按指令操作码跳转”。  **为了清晰地展示每一位下一微地址的生成逻辑，我总结了不同 P1 和相关 IR 位条件下的输出：**   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 下一微地址位 | 当前相关位 | 条件 (P1) | 条件 (对应IR位) | 逻辑表达式 | NµA\_i-T 的值 | | NµA5-T | NµA5 | X (任意) | X (任意) | NµA5 | NµA5 | | NµA4-T | NµA4 | X (任意) | X (任意) | NµA4 | NµA4 | | NµA3-T | NµA3 | 0 | X (IR7) | NµA3 OR (0 AND IR7) = NµA3 | NµA3 | |  | NµA3 | 1 | IR7 = 0 | NµA3 OR (1 AND 0) = NµA3 | NµA3 | |  | NµA3 | 1 | IR7 = 1 | NµA3 OR (1 AND 1) = NµA3 OR 1 | 1 | | NµA2-T | NµA2 | 0 | X (IR6) | NµA2 OR (0 AND IR6) = NµA2 | NµA2 | |  | NµA2 | 1 | IR6 = 0 | NµA2 OR (1 AND 0) = NµA2 | NµA2 | |  | NµA2 | 1 | IR6 = 1 | NµA2 OR (1 AND 1) = NµA2 OR 1 | 1 | | NµA1-T | NµA1 | 0 | X (IR5) | NµA1 OR (0 AND IR5) = NµA1 | NµA1 | |  | NµA1 | 1 | IR5 = 0 | NµA1 OR (1 AND 0) = NµA1 | NµA1 | |  | NµA1 | 1 | IR5 = 1 | NµA1 OR (1 AND 1) = NµA1 OR 1 | 1 | | NµA0-T | NµA3 | 0 | X (IR4) | NµA3 OR (0 AND IR4) = NµA3 | NµA3 | |  | NµA3 | 1 | IR4 = 0 | NµA3 OR (1 AND 0) = NµA3 | NµA3 | |  | NµA3 | 1 | IR4 = 1 | NµA3 OR (1 AND 1) = NµA3 OR 1 | 1 |  1. **微程序流程图分析**     **图2 微程序流程图**  每次系统复位后，整个流程会从顶部的P0条件判断开始。如果P0的特定条件（如图中所示的 SEL3=0, SEL2=0, SEL1=1, SEL0=1 且 CLR=0）得到满足，系统便会依据外部开关SWC、SWB和SWA的组合状态，进入一个预设的操作模式。  这些开关的不同取值，例如000、001或010，实际上是为微控制器选择了不同的工作路径，比如“取指模式”、“寄存器存模式”或“寄存器读模式”等。一旦操作模式确定，微控制器便会严格按照流程图中箭头的指引，一步步地执行构成该模式的一系列微指令，这些微指令是控制CPU内部各种硬件单元（如总线、寄存器组、算术逻辑单元ALU等）执行基本操作的最小命令。  我们可以重点关注一下当SWC、SWB、SWA的组合为000时所启动的“取指模式”。这通常是CPU执行程序时最基本和最频繁的模式。进入此模式后，微程序首先会执行一系列初始化取指操作，如图中状态01所示，这包括LIR、PCINC以及P1。紧接着，系统会检查是否存在中断请求（INT标志）。若无中断（INT=0），微控制器便会根据当前指令寄存器IR中的高4位（IR7至IR4）进行指令译码。这高4位编码唯一地标识了当前需要执行的指令类型，例如0001代表ADD，0010代表SUB，0101代表LD等等。  我们以执行一条ADD指令为例来具体说明：假设系统处于取指模式且无中断，当前指令的IR7-IR4编码为0001。微程序会跳转到标记为21的状态序列。在此序列中，一系列针对加法操作的微指令会被依次执行，如S=0001用于配置ALU执行加法运算，CIN可能用于设置初始进位，ABUS用于选择并传送操作数到ALU，DRW用于写数据，以及LDZ和LDC用于根据运算结果更新零标志位和进位标志位。在这一系列微指令执行完毕后，系统会进行一个P4条件判断，它将决定微程序的下一步走向，可能是返回到取指周期的开端以获取并执行下一条指令，或者是根据P4的状态跳转到其他处理流程。如果之前检测到中断（INT=1），则会转入中断处理子程序，如状态11所示，执行INTDI、LIAR等操作。  其他由SWC、SWB、SWA决定的模式，如“寄存器存”（001）、“寄存器读”（010）、“直接存取”（011）等，它们各自对应着一套独特的微操作序列，用于完成特定的数据处理或控制任务，认真研究后发现真的很有意思。   1. **实验过程及结果**  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **①写存储器** | | | | | | | | | |  | | **序号** | | **操作** | **操作的数据** | **操作目的** | **操作结果** | | | | | | | | | **控制信号（有效的信号）** | **μA** | **NμA** | **D7-D0** | **A7-A0** | **B7-B0** | **AR7-AR0** | | | 1 | | CLR | 无 | 复位 | SEL3-SEL0=0011 P0=1 | 00H | 01H |  |  |  |  | | | 2 | | SW=001 QD | 无 | 操作模式设置为写存储器 | SBUS=1，LAR=1，SETCTL=1,STOP=1 | 03H | 02H |  |  |  |  | | | 3 | | QD | 00H | 选定初始地址 | SBUS=1，MEMW=1，ARINC=1，STOP=1，SELCTL=1 | 02H | 02H | 00H |  |  | 00H | | | 4 | | QD | 00H | 写入数据00H | SBUS=1，MEMW=1，ARINC=1，STOP=1，SELCTL=1 | 02H | 02H | 00H |  |  | 01H | | | 5 | | QD | 01H | 在01H中写入01H | SBUS=1，MEMW=1，ARINC=1，STOP=1，SELCTL=1 | 02H | 02H | 01H |  |  | 02H | | | 6 | | QD | 02H | 在02H中写入02H | SBUS=1，MEMW=1，ARINC=1，STOP=1，SELCTL=1 | 02H | 02H | 02H |  |  | 03H | | | 7 | | QD | 03H | 在03H中写入03H | SBUS=1，MEMW=1，ARINC=1，STOP=1，SELCTL=1 | 02H | 02H | 03H |  |  | 04H | |   **②读存储器**   |  | | --- | |  | | **操作** | | | **操作的数据** | **操作目的** | **操作结果** | | | | | | | | **控制信号（有效的信号）** | **μA** | **NμA** | **D7-D0** | **A7-A0** | **B7-B0** | **AR7-AR0** | | CLR | | | 无 | 复位 | SEL3-SEL0=0011 P0=1 | 00H | 01H |  |  |  |  | | SW=010 QD | | | 00H |  | SBUS=1，LAR=1，STOP=1 | 05H | 04H | 03H | 00H | 00H | 00H | | QD | | | 无 |  | MBUS=1，ARINC=1，STOP=1 | 04H | 04H | 00H | 00H | 00H | 00H | | QD | | | 无 |  | MBUS=1，ARINC=1，STOP=1 | 04H | 04H | 01H | 00H | 00H | 01H | | QD | | | 无 |  | MBUS=1，ARINC=1，STOP=1 | 04H | 04H | 02H | 00H | 00H | 02H | | QD | | | 无 |  | MBUS=1，ARINC=1，STOP=1 | 04H | 04H | 03H | 00H | 00H | 03H | |  | | | 22H |  |  |  |  |  |  |  |  | | QD | | | 无 |  | MBUS=1，ARINC=1，STOP=1 | 04H | 04H | 00H | 00H | 00H | 22H |   **③写寄存器**   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **序号** | **操作** | **操作的数据** | **操作目的** | **操作结果** | | | | | | | **控制信号（有效的信号）** | **μA** | **NμA** | **D7-D0** | **A7-A0** | **B7-B0** | | 1 | CLR | 无 | 复位 | SEL3-SEL0=0011 P0=1 | 00H | 01H |  |  |  | | 2 | SW=100 QD | 无 | 写寄存器模式 | SBUS=1 SEL3-SEL0=0001 SELCTL=1 DRW=1 STOP=1 | 09H | 08H |  |  |  | | 3 | QD | 11H | 置数进R0 | SEL3~SEL0=0100,DRW=1,SELCTL=1,SBUS=1,STOP=1 | 08H | 0AH | 11H |  | 11H | | 4 | QD | 03H | 置数进R1 | SEL3~SEL0=1001,DRW=1,SELCTL=1,SBUS=1,STOP=1 | 0AH | 0CH | 03H |  | 03H | | 5 | QD | 04H | 置数进R2 | SEL3~SEL0=1110,DRW=1,SELCTL=1,SBUS=1,STOP=1 | 0CH | 00H | 04H |  | 04H | | 6 | QD | 44H | 置数进R3 | SEL3~SEL0=0011,DRW=1,SELCTL=1,SBUS=1,STOP=1 | 00H | 20H | 44H |  | 44H |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **④读寄存器（从R0-R3中分别读出它们存储的数据）** | | | | | | | | | | | | **序号** | | **操作** | **操作的数据** | **操作目的** | **操作结果** | | | | | | | **控制信号（有效的信号）** | **μA** | **NμA** | **D7-D0** | **A7-A0** | **B7-B0** | | 1 | | CLR | 无 | 复位 | SEL3-SEL0=0011 P0=1 | 00H | 01H |  |  |  | | 2 | | SW=011 QD | 无 | 选择操作模式为读寄存器 | RS1～RS0=01，RD1~RD0=00，STOP=1 | 07H | 06H | 00H | 00H | 03H | | 3 | | QD | 无 |  | RS1～RS0=11，RD1~RD0=10，STOP=1 | 06H | 00H | 00H | 04H | 44H |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **⑤指令执行-LD** | | | | | | | | | | | | |  | |  | |  | | | **序号** | | **操作** | | **操作的数据** | **操作目的** | | **操作结果** | | | | | | | | | | | | | **控制信号（有效的信号）** | **μA** | **NμA** | **D7-D0** | **A7-A0** | **B7-B0** | **PC7-PC0** | | **INS7-INS0** | | **IR7-IR0** | | | 1 | | CLR | | 无 | 复位 | | SEL3-SEL0=0011 P0=1 | 00H | 01H |  |  |  |  | |  | |  | | | 2 | | SW=000 QD | | 无 | 设操作模式SW=000 | | LIR=1,PCINC=1,P1=1 | 01H | 20H |  |  |  |  | |  | |  | | | 3 | | IR7～IR4=0101，QD | | 无 | 选择微命令为LD | | ABUS=1，LAR=1，M=1，S3～S0=0010 | 25H | 0EH | 22H | 22H | 22H | 01H | | 01H | | 00H | | | 4 | | QD | | 无 |  | | P4=1，MBUS=1，DRW=1 | 0EH | 01H | 00H | 22H | 22H | 01H | | 01H | | 00H | | | 5 | | QD | | 无 |  | | LIR=1，PCINC=1，AR7～AR0=22H | 01H | 20H | 00H | 00H | 00H | 01H | | 01H | | 00H | | | **⑥指令执行-ADD** | | | | | | | | | | | | | | | | | | | | | | | | |  |  |  | | **序号** | | **操作** | | | **操作的数据** | **操作目的** | | **操作结果** | | | | | | | | | | | | | | | | | | | | | **控制信号** | | | | | | | | | | | | **μA** | **NμA** | **D7-D0** | **A7-A0** | **B7-B0** | **PC7-PC0** | **INS7-INS0** | **IR7-IR0** | | 1 | | CLR | | | 无 | 复位 | | SEL3-SEL0=0011 P0=1 | | | | | | | | | | | | 00H | 01H |  |  |  |  |  |  | | 2 | | SW=000 QD | | | 无 | 设置操作模式SW=000 | | LIR=1,PCINC=1,P1=1 | | | | | | | | | | | | 01H | 20H |  |  |  | 00H | 00H |  | | 3 | | IR7～IR4=0001，QD | | | 无 | 选择执行微命令为ADD | | S=1001,CIN=1,ABUS=1,DRW=1,LDZ=1,  LDC=1,P4=1 | | | | | | | | | | | | 21H | 01H | 22H | 11H | 11H | 01H | 01H | 00H | | 4 | | QD | | | 无 | 将运算结果存储到寄存器R0 | | LIR=1,PCINC=1,P1=1 | | | | | | | | | | | | 0H | 21H |  |  |  | 01H | 01H | 01H |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **⑦指令执行-ST** | | | | | | | | | | |  | |  | |  | | | **序号** | | **操作** | **操作的数据** | **操作目的** | **操作结果** | | | | | | | | | | | | | **控制信号（有效的信号）** | **μA** | **NμA** | **D7-D0** | **A7-A0** | **B7-B0** | **PC7-PC0** | | **INS7-INS0** | | **IR7-IR0** | | | 1 | | CLR | 无 | 复位 | SEL3-SEL0=0011 P0=1 | 00H | 01H |  |  |  |  | |  | |  | | | 2 | | SW=000 QD | 无 | 设操作模式 | LIR=1,PCINC=1,P1=1 | 01H | 20H |  |  |  |  | |  | |  | | | 3 | | IR7～IR4=0110，QD | 无 | 选择微命令为STO | M=1，S3～S0=1111，ABUS=1，LAR=1 | 26H | 10H | 00H | 00H | 00H | 01H | | 01H | | 00H | | | 4 | | QD | 无 |  | M=1，S3～S0=1010，ABUS=1，MEMW=1，P4=1 | 10H | 01H | 00H | 00H | 00H | 01H | | 01H | | 00H | | | 4 | | QD | 无 | 将00H存到（00H）中 | LIR=1，PCINC=1 | 01H | 20H | 00H | 00H | 00H | 01H | | 01H | | 00H | |  1. **实验收获及体会**   通过本次实验，我收获颇丰，感触良多。  在实验初期，我对微程序的概念和计算机内部指令的执行流程的理解尚停留在理论层面，对实验箱的微程序相关操作也感到相当生疏。然而，随着实验的逐步深入，这种状况得到了显著的改善。  在执行具体的实验操作，如将程序写入存储器、给寄存器赋初值、跟踪执行LD、ADD、ST等指令的过程中，我切身体会到了机器指令与微指令之间的紧密联系。  总的来说，本次实验极大地巩固和深化了我对计算机组成原理的理解。我不再仅仅是知道计算机能执行指令，而是深入了解了它是如何取出指令、如何分析指令、如何一步步执行指令，以及如何在一条指令结束后自动转到下一条指令的。  **实验五、CPU组成与机器指令的执行**   1. **实验任务及目的**   **①实验任务:**   1. 预习任务：完成对给定程序的手工汇编。 2. 通过简单的连线构成能够运行程序的TEC-8/TEC-Plus模型计算机。 3. 将程序写入存储器，给寄存器R2、R3赋初值。 4. 跟踪执行程序，用单拍方式运行一遍，用连续方式运行一遍，详细记录实验过程及结果。 5. 用实验台操作检查程序运行结果。   **②实验目的:**  （1）用微程序控制器控制数据通路，将相应的信号线连接，构成一台能够运行测试程序的CPU。  （2）执行一个简单的程序，掌握机器指令与微指令的关系。  （3）理解计算机如何取出指令、如何执行指令、如何在一条指令执行结束之后自动取出下一条指令并执行，从而牢固建立计算机整机概念。   1. **程序的手工汇编结果**  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **地址** | **指令** | **二进制机器代码** | **地址** | **指令** | **二进制机器代码** | | 00H | LD R0,[R3] | 01010011 | 0AH | INC R2 | 01001000 | | 01H | INC R3 | 01001100 | 0BH | ST R2,[R2] | 01101010 | | 02H | LD R1,[R3] | 01010111 | 0CH | AND R0,R1 | 00110001 | | 03H | SUB R0,R1 | 00100001 | 0DH | OUT R2 | 10100010 | | 04H | JZ 0BH | 10000110 | 0EH | STP | 11100000 | | 05H | ST R0,[R2] | 01101000 | 0FH | 85H | 10000101 | | 06H | INC R3 | 01001100 | 10H | 23H | 00100011 | | 07H | LD R0,[R3] | 01010011 | 11H | EFH | 11101111 | | 08H | ADD R0,R1 | 00010001 | 12H | 00H | 00000000 | | 09H | JC 0CH | 01110010 | 13H | 00H |  |  1. **实验过程及结果**   **实验过程记录表（单拍）**   |  |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | 指令 | μA | NμA | P | INS | PC | AR | IR | A | B | D | |  | 000001 | 100000 | 00010 | 01010011 | 00H | 00H | 00000000 |  |  |  | | LD R0,[R3] | 100101 | 001110 | 00000 | 01010011 | 01H | 00H | 01010011 |  |  |  | |  | 001110 | 000001 | 10000 | 01001100 | 01H | 0fH | 01010011 |  | 0FH | 85H | |  | 000001 | 100000 | 00010 | 01001100 | 01H | 0fH | 01010011 | 85H | 0FH |  | | INC R3 | 100100 | 000001 | 10000 | 01010111 | 02H | 0FH | 01001100 | 0FH | 85H | 10H | |  | 000001 | 100000 | 00000 | 01010111 | 02H | 0FH | 01001100 | 10H | 85H |  | | LD R1,[R3] | 100101 | 001110 | 00000 | 00100001 | 03H | 0FH | 01010111 |  | 10H | 10H | |  | 001110 | 000001 | 10000 | 00100001 | 03H | 10H | 01010111 |  | 10H | 23H | |  | 000001 | 100000 | 00000 | 00100001 | 03H | 10H | 01010111 | 23H | 10H |  | | SUB R0,R1 | 100010 | 000001 | 10000 | 10000110 | 04H | 10H | 00100001 | 85H | 23H | 62H | |  | 000001 | 100000 | 00000 | 10000110 | 04H | 10H | 00100001 | 62H | 23H |  | | JZ 0BH | 101000 | 010010 | 00000 | 01101000 | 05H | 10H | 10000110 | 23H | 12H |  | |  | 010010 | 000001 | 10000 | 01101000 | 05H | 10H | 10000110 | 23H | 12H |  | |  | 000001 | 100000 | 00000 | 01101000 | 05H | 10H | 10000110 | 23H | 12H |  | | ST R0,[R2] | 100110 | 010000 | 00000 | 01001100 | 06H | 10H | 01101000 | 12H | 62H | 12H | |  | 010000 | 000001 | 10000 | 01001100 | 06H | 12H | 01101000 | 12H | 62H | 62H | |  | 000001 | 100000 | 00000 | 01001100 | 06H | 12H | 01101000 | 12H | 62H |  | | INC R3 | 100100 | 000001 | 10000 | 01010011 | 07H | 12H | 01001100 | 10H | 62H | 11H | |  | 000001 | 100000 | 00000 | 01010011 | 07H | 12H | 01001100 | 11H | 62H |  | | LD R0,[R3] | 100101 | 001110 | 00000 | 00010001 | 08H | 12H | 01010011 | 62H | 11H | 11H | |  | 001110 | 000001 | 10000 | 00010001 | 08H | 11H | 01010011 | 62H | 11H | EFH | |  | 000001 | 100000 | 00000 | 00010001 | 08H | 11H | 01010011 | EFH | 11H |  | | ADD R0,R1 | 100001 | 000001 | 10000 | 01110010 | 09H | 11H | 00010001 | EFH | 23H | 12H | |  | 000001 | 100000 | 00000 | 01110010 | 09H | 11H | 00010001 | 12H | 23H |  | | JC 0CH | 100111 | 010010 | 00000 | 01001000 | 0AH | 11H | 01110010 | 12H | 12H |  | |  | 010011 | 000001 | 10000 | 01001000 | 0AH | 11H | 01110010 | 12H | 12H |  | |  | 000001 | 100000 | 00000 | 00110001 | 0CH | 11H | 01110010 | 12H | 12H |  | | AND R0,R1 | 100001 | 000001 | 10000 | 10100010 | 0DH | 11H | 00110001 | 12H | 23H | 02H | |  | 000001 | 100000 | 00000 | 10100010 | 0DH | 11H | 00110001 | 02H | 23H |  | | OUT R2 | 101010 | 000001 | 10000 | 11100000 | 0EH | 11H | 10100010 | 02H | 12H | 12H | |  | 000001 | 100000 | 00000 | 11100000 | 0EH | 11H | 10100010 | 02H | 12H |  | | STP | 101110 | 000001 | 10000 | 10000101 | 0FH | 11H | 11100000 | 02H | 02H |  | |  | 000001 | 100000 | 00000 | 10000101 | 0FH | 11H | 11100000 | 02H | 02H |  |   **连续方式**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | 执行操作前 | R0 | R1 | R2 | R3 | |  | 00H | 00H | 12H | 0FH | | 执行操作后 | R0 | R1 | R2 | R3 | |  | 02H | 23H | 12H | 11H |  1. **实验收获及体会**   本次实验单拍执行模式让我印象深刻，它清晰地揭示了每一条机器指令是如何被分解为一系列微操作，这让我对课本上“取指周期”、“执行周期”等抽象概念有了具象化的理解。  然而，实验过程并非一帆风顺。最让我记忆犹新，也是投入最多精力去排查的，是在程序执行到最后的STP指令时，计算机并没有如预期般停止运行，而是仿佛陷入了某种循环或者继续执行了未知的操作。起初，我反复检查实验台的连线，担心是某个关键的控制信号线没有接好，一遍又一遍地确认，但问题依旧。随后，我又怀疑是不是寄存器的初值设置有误，导致程序流程在STP前就发生了意外跳转，但仔细核对R2、R3的初值后，也排除了这个可能。  那段时间，我确实有些沮丧和困惑。明明前面的指令都能正确执行，为何偏偏STP指令失效？在连续运行模式下，更是无法观察到它具体在哪里出了问题。无奈之下，我只能重新回到最基础的部分——我手工汇编的二进制机器代码。我拿出当时的汇编结果，与实验PPT上的标准指令编码逐条、逐位地进行比对。就在我几乎要放弃，准备向老师求助的时候，终于在STP指令的二进制编码上发现了一个微小的差异！原来，在手工转换为二进制机器码的过程中，我不慎将STP指令（11100000）的一个比特位写错了  这个小小的STP指令编码错误，却耗费了我大量的时间和精力，也让我深刻体会到了计算机科学的严谨性。  **实验六、中断原理实验**   1. **实验任务及目的**   **①实验任务:**   1. 理解中断相关指令，以及每个信号的意义和变化条件 2. 将主程序和中断服务程序手工汇编成二进制机器代码 3. 通过简单的连线构成能够运行程序的TEC-8模型计算机。 4. 将主程序和中断服务程序装入存储器，给寄存器R1赋初值01H，R0赋初值0。 5. 执行三遍主程序和中断服务程序，详细记录中断有关信号变化情况，特别记录好断点和R0 6. 将主程序种地址为00H的EI指令改为DI，重新运行程序，记录现象。   **②实验目的（源自其他班ppt）:**  （1）从硬件、软件结合的角度，模拟中断的过程；  （2）通过简单的中断系统掌握中断的相关概念；  （3）了解微程序控制器与中断控制器协调的基本原理；  （4）掌握中断子程序和一般子程序的本质区别,掌握中断的突发性和随机性。   1. **程序的手工汇编结果（包括主程序和中断服务程序）**  |  |  |  | | --- | --- | --- | | **主程序机器代码** | | | | **地址** | **指令** | **二进制机器代码** | | 00H | EI\DI | 11010000 | | 01H | INC R0 | 1000000 | | 02H | INC R0 | 1000000 | | 03H | INC R0 | 1000000 | | 04H | INC R0 | 1000000 | | 05H | INC R0 | 1000000 | | 06H | INC R0 | 1000000 | | 07H | INC R0 | 1000000 | | 08H | INC R0 | 1000000 |      |  |  |  | | --- | --- | --- | | **中断服务程序机器代码** | | | | **地址** | **指令** | **二进制机器代码** | | 45H | ADD R0,R0 | 00100000 | | 46H | EI | 11010000 | | 47H | IRET | 10110000 |  1. **实验过程及结果**  |  |  |  | | --- | --- | --- | | 执行程序顺序 | PC断点值 | 中断时的R0 | | 第1遍 | 05H | 15H | | 第2遍 | 06H | 84H | | 第3遍 | 09H | C0H |  1. **实验收获及体会**   本次实验，练就了我逐条对照、严谨校验的良好习惯，我不仅掌握了中断的基本概念、相关指令的用法，更对微处理器如何响应外部事件、如何在软硬件层面协同工作有了更具体的认识。 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **拓展实验**  **一、新设计的主程序的指令序列及手工汇编结果：**  为清晰起见，下表列出了修改后**主程序**/**中断服务程序**的指令和新地址（其对应的原实验五指令）、二进制机器码以及必要的注释。   |  |  |  |  | | --- | --- | --- | --- | | **新地址** | **指令** | **二进制机器代码** | **注释** | | **00H** | **EI** | **11010000** | **使能中断** | | **01H** | **LD R0,[R3]** | **01010011** |  | | **02H** | **INC R3** | **01001100** |  | | **03H** | **LD R1,[R3]** | **01010111** |  | | **04H** | **SUB R0,R1** | **00100001** |  | | **05H** | **JZ 0BH** | **10000110** | **若Z=1, PC -> 0CH (原目标0BH, 新目标0BH+1)** | | **06H** | **ST R0,[R2]** | **01101000** | **若Z=0则执行** | | **07H** | **INC R3** | **01001100** |  | | **08H** | **LD R0,[R3]** | **01010011** |  | | **09H** | **ADD R0,R1** | **00010001** |  | | **0AH** | **JC 0CH** | **01110010** | **若C=1, PC -> 0DH (原目标0CH, 新目标0CH+1)** | | **0BH** | **INC R2** | **01001000** | **若C=0则执行** | | **0CH** | **ST R2,[R2]** | **01101010** | **JZ的目标指令。执行后顺序到0DH** | | **0DH** | **AND R0,R1** | **00110001** | **JC的目标指令。也可从0CH顺序到达** | | **0EH** | **OUT R2** | **10100010** |  | | **0FH** | **JMP 00H** | **10010000** | **跳转回00H，实现循环** |   **表1：主程序的指令序列及手工汇编**   |  |  |  |  | | --- | --- | --- | --- | | **地址** | **指令** | **二进制机器代码** | **注释** | | **45H** | **ADD R0,R0** | **00010000** | **将寄存器R0的内容加倍** | | **46H** | **EI** | **11010000** | **开中断** | | **47H** | **IRET** | **10110000** | **中断返回** |   **表2：中断服务程序汇编代码**   1. **设计思路**   本次综合实验的核心目标是将实验五的单次执行程序转变为一个能够连续运行并响应中断的主程序，该主程序在发生中断时，将调用实验六的ISR进行处理，处理完毕后返回主程序继续执行。为实现此目标，需对原实验五程序进行以下关键修改：  使能中断：必须在主程序中加入EI指令，以允许CPU识别和响应中断请求。此指令将置于修改后程序的最开始部分。  程序循环**：原实验五程序末尾的STP（Stop）指令需要被替换为一条无条件跳转指令JMP**，该指令将跳转回修改后主程序的起始地址，从而实现主程序的循环执行。这为中断的发生提供了持续的执行背景。  指令与地址调整：在原实验五程序代码的起始处插入EI指令，会导致其后所有原有指令的地址向后顺移一位。因此，原程序中所有绝对地址跳转指令（如JZ和JC的目标地址）也必须相应地增加1。  数据区重定位：实验五程序使用了存储在地址0FH、10H和11H的数据，R3寄存器初始指向0FH。由于EI指令的加入以及原有指令的顺移，原位于0EH的STP指令（现被JMP指令替代）的新地址将是0FH。这意味着0FH单元现在存放的是指令而非数据。因此，原始数据（85H, 23H, EFH）必须迁移到新的内存地址（例如，30H, 31H, 32H）。相应地，R3寄存器的初始值也需要更新，指向这个新数据块的起始地址。  ISR的放置：实验六的ISR（起始于45H）假定位于一个与主程序及其重定位数据区不冲突的内存区域。  在进行这些修改时，必须确保原实验五程序的核心计算逻辑和指令执行顺序在主程序的每一次循环迭代中得到保留。中断服务程序对R0寄存器的修改是一种预期的交互行为，这在允许改变寄存器操作数的约束下是可接受的。虽然如果R0在中断期间被修改，计算结果会发生变化，但原程序指令的执行序列保持不变。  一个重要的考量点是ISR与主程序状态之间的交互。ISR中的ADD R0,R0指令直接修改R0寄存器的内容。而主程序（原实验五的逻辑）在其运算过程中广泛使用R0作为操作数（例如，SUB R0,R1、ADD R0,R1、AND R0,R1）或作为数据源（例如，ST R0,）。如果一个中断发生在R0被主程序加载数据之后（如执行完LD R0,）、但在其被后续算术/逻辑指令使用之前，那么ISR将会执行，使得R0的值加倍。随后，主程序将恢复执行，并使用这个已被ISR修改过的R0值。   1. **运行结果**   为保证实验的可复现性和结果分析的准确性，实验开始前必须设定明确的初始状态   |  |  |  |  | | --- | --- | --- | --- | | **元件** | **地址** | **初始值** | **注释说明** | | PC | N/A | 00H | 指向修改后主程序的起始地址 | | R0 | N/A | 00H | (参照实验六初始值) | | R1 | N/A | 00H | (参照实验六初始值) | | R2 | N/A | 12H | 源自实验五，用作指针和数据存储 | | R3 | N/A | 30H | 指向新的数据区基地址 | | 中断标志IF | N/A | 禁止 (CPU复位状态)，后由EI开启 | 程序00H处的EI指令执行后将变为允许状态 | | 内存数据1 | 30H | 85H | 原M[0FH]的内容 | | 内存数据2 | 31H | 23H | 原M[10H]的内容 | | 内存数据3 | 32H | EFH | 原M[11H]的内容 | | 主程序代码 | 00H-0FH | (见表1) | 加载修改后的主程序机器码 | | 中断服务程序代码 | 45H-47H | (见表2) | 加载ISR机器码 | | 内存单元M[12H] | 12H | 00H | ST R0, 和 ST R2, 的目标存储单元 |   **表3：初始状态说明**  **单次循环中发生中断的追踪片段**   |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | | **步骤** | **PC** | **指令** | **R0** | **R1** | **R2** | **R3** | **M[12H]** | **标志 Z,C** | **注释** | | 1 | 00H | EI | 00H | 00H | 12H | 30H | 00H | – , – | 开中断，IF 变为 1 | | 2 | 01H | LD R0, | 00H | 00H | 12H | 30H | 00H | – , – | R3 (30H) → M[30H]  (85H) → R0 | | 3 | 02H | (执行后) | 85H | 00H | 12H | 30H | 00H | (N, N) | R0 被加载为 85H；  此时发生中断 | | 4 | 45H | ADD R0, R0 | 85H | 00H | 12H | 30H | 00H | (N, N) | 进入 ISR，  准备将 R0 (85H) 加倍 | | 5 | 46H | (执行后) | 0AH | 00H | 12H | 30H | 00H | (N, Y) | R0 = 85H + 85H = 10AH  →  0AH（溢出 C=1） | | 6 | 46H | EI | 0AH | 00H | 12H | 30H | 00H | (N, Y) | 重新开中断 | | 7 | 47H | IRET | 0AH | 00H | 12H | 30H | 00H | (N, Y) | 中断返回，  PC 恢复到 02H，  标志恢复 | | 8 | 02H | INC R3 | 0AH | 00H | 12H | 30H | 00H | (N, N) | 主程序从 02H 继续；  R0 值已被 ISR 修改为 0AH | | 9 | 03H | (执行后) | 0AH | 00H | 12H | 31H | 00H | (N, N) | R3 递增为 31H | | 10 | 03H | LD R1, | 0AH | 00H | 12H | 31H | 00H | (N, N) | R3 (31H) → M[31H]  (23H) → R1 | | 11 | 04H | (执行后) | 0AH | 23H | 12H | 31H | 00H | (N, N) | R1 被加载为 23H | | 12 | 04H | SUB R0, R1 | 0AH | 23H | 12H | 31H | 00H | (N, N) | R0 = R0 – R1 =  0AH – 23H | | 13 | 05H | (执行后) | E7H | 23H | 12H | 31H | 00H | (N, Y) | R0 = E7H（借位 C=1），  Z=0 | | 14 | 05H | JZ 0CH | E7H | 23H | 12H | 31H | 00H | (N, Y) | Z=0，不跳转；  PC → 06H | | 15 | 06H | ST R0, | E7H | 23H | 12H | 31H | 00H | (N, Y) | R0 (E7H) → M[12H] | | 16 | 07H | (执行后) | E7H | 23H | 12H | 31H | E7H | (N, Y) | M[12H] 被更新为 E7H | |  |  |  |  |  |  |  |  |  | **主程序继续运行** |   **表4：单次循环中发生中断的示意性执行追踪片段**  由于程序设计为无限循环且中断发生具有随机性，不存在一个固定的“最终状态”，我们下面进行**多次循环与中断发生后的**详细分析：  **多次循环与中断发生后对终态的分析**   |  |  |  | | --- | --- | --- | | **元件** | **最终值** | **注释** | | R0 | (不确定) | 取决于最后一次中断相对于LD R0或算术指令的发生时机 | | R1 | (不确定) | 取决于R3的值以及对应内存单元的内容 | | R2 | (不确定) | 因INC R2指令（若JC条件不满足且JZ条件不满足时，路径会经过0BH）的执行而改变 | | R3 | (不确定) | 会在数据区内循环递增，具体值取决于执行的循环次数 | | M[12H] | (不确定) | 反映最后一次通过ST R0,或ST R2,写入的值 | | OUT R2的输出 | (序列值) | 将会是一系列根据R2在执行OUT R2指令时的值而变化的输出 |   **表5：多次循环与中断发生后寄存器及关键内存的最终状态**   1. **总结**   通过细致的地址调整和指令替换，修改后的主程序在每次从00H到0FH的循环迭代中，完整地保留了实验五程序的核心指令序列（如LD, INC, SUB, JZ, ST, ADD, JC, AND, OUT）。对条件跳转指令JZ 0CH（原JZ 0BH）和JC 0DH（原JC 0CH）的目标地址进行的修正，确保了即使在代码整体位移一个字节后，条件分支依然能跳转到正确的对应指令，维持了原有的控制流程结构。  老师第一节课曾提到“做实验之前，应该先学习理论，掌握原理，实践是用来巩固学习的，而不是通过实验学习陌生的知识”。本次综合实验的设计与执行，正是对这一深刻体会的进一步印证和升华。  至此，本学期计组实验，完结撒花。 |

|  |
| --- |
|  |