计算机组成原理课程设计答辩记录表

|  |  |  |  |
| --- | --- | --- | --- |
| 学 院 | 专业 / 学号 | 学 生 | 指导教师 |
| 计算机科学与技术学院（人工智能学院） | 22计算机科学与技术/  2022334323029 | 张雅瑞 | 许建龙 杨东鹤 |
| 答辩时间 (年月日/时刻) | 2024年6月22日  12：00-12：00 | 答 辩 地 点 | 10-413 |
| 答辩记录(可另附页)：  问题1：如何自行设计新指令的微程序  回答1：  新旧指令可以在同一程序中并存使用；然而，值得注意的是，每当教学机重启后，新指令对应的微码必须至少重新加载一次。只有成功加载了相应的微码，新指令才能在系统中得到执行，这标志着新指令已成功整合进教学机的指令系统中。此外，由于新指令在汇编程序实现之后才被设计，因此它们在程序中仅能通过机器码来调用，否则就需要对相关的汇编程序进行扩展。  幸运的是，加载新的微指令和使用新指令的过程变得相当便捷，这使得我们能够专注于微程序设计本身。不过，新旧指令的微程序之间的协调与配合不容忽视：新指令的具体执行功能虽然独立设计，但在执行前的取指阶段、执行后的中断判断以及与下一条指令的衔接等方面，仍然依赖于原微程序的相关部分。从学习微程序设计的角度看，这些公共的取指和后续处理过程对所有指令都是通用的，并且相对简单易懂。掌握其实现方法和细节，以及它们与指令执行过程的衔接方式，对于深入理解微程序设计至关重要。  如果希望在自己的微程序中实现特定的取指和后续处理过程，那么必须确保这些新过程在新旧指令切换时不产生冲突。一种简单的方法是，在自定义的多条新指令中，选择其中几条进行独特的取指和后续处理设计，而其余的新指令则利用原有的微程序段来完成这些通用过程。特别地，至少有一条新指令应使用原指令的取指过程，另一条新指令应使用原指令的后续处理过程，以确保与原有指令的流畅衔接。当这样的两条新指令被“夹”在其他新指令序列中时，整个序列将能够在任何程序中正确执行。  学习计算机微程序设计的核心在于掌握设计不同格式与功能的新指令的执行步骤，以及每一步中微指令字各字段的编码方法。这不仅是理解计算机指令执行过程的关键，也是学习微程序设计的重点。  问题2：解释 [ADDR1]=[ADDR1] -[ADDR2]  回答2：  该指令设计为三字长度，旨在从两个指定的地址中取出内容，执行相减操作，并将结果存回第一个地址。考虑到第一个地址在整个操作过程中都会被频繁使用，为了提高执行效率，我们可以采取一种优化策略：首先，将第二个操作数从对应地址取出并暂存于Q寄存器中。随后，再取出第一个操作数，与Q寄存器中的值进行相减运算。最后，将相减的结果写回第一个地址位置。通过这种方法，我们减少了重复访问内存的需要，从而提升了指令的执行效率。 | | | |