

复杂指令集流水线系统设计

靳文兵^{1,2}, 左 琦¹

(1 北京理工大学 计算机学院, 北京 100081; 2 北方自动控制技术研究所, 山西 太原 030006)

摘 要: 摒弃传统流水线设计必须先将复杂指令集指令转化为精简指令集指令, 然后再按照精简指令集实现流水线的方法. 采用拓展的哈佛结构, 设计新型指令流水线前端多指令缓冲和双指令指针, 以及流水线中、后端双总线寄存器组和多端口数据存储器, 优化指令流水线结构, 实现高效率的复杂指令集指令流水线系统. 设计从理论上解决了复杂指令集流水线实现的两个难点: 寄存器和存储器读写冲突问题, 以及流水线各阶段功能和任务划分. VHDL 语言建模, 用 ModelSim 和 Xilinx 仿真、测试, 证明复杂指令集流水线系统设计可行.

关键词: 微处理器; 复杂指令集; 流水线; 多端口存储器; 读写冲突

中图分类号: TP32

文献标识码: A

文章编号: 1000-7180(2012)04-0019-04

Design of CISC Pipeline System

JIN Wen-bing^{1,2}, ZUO Qi¹

(1 School of Computer Science and Technology, Beijing Institute of Technology, Beijing 100081, China;

2 North Automatic Control Technology Institute, Taiyuan 030006, China)

Abstract: Design of pipeline without translation from CISC instructions to RISC instructions. Instruction buffers and caches are implemented in the front stage of CISC pipeline with double Program Counters pointing to opcode and operand separately. Multi-port registers and memories are designed to support simultaneously read and write by different stages. Optimization for pipeline functions is adopted in order to realize an effective CISC pipeline system. Final evaluation and test by ModelSim and Xilinx tool sets prove the design workable and preferable.

Key words: microprocessor; CISC; Pipeline; multi-port memory; read-write conflict

1 引言

指令流水线是计算机系统降低 CPI 指标的主要方法. 流水线设计的根本是处理器中关键部件的时空复用. 复杂指令集计算机(CISC)面向存储器或围绕个别寄存器设计, 存储器操作指令较多, 指令长度可变. 因此, CISC 系统实现指令流水线非常困难.

当前设计的 CISC 流水线, 均是先把 CISC 指令逐条动态翻译为 RISC 指令或类 RISC 指令, 然后再按照 RISC 指令方式实现指令流水线^[1-2]. 这种方法仅适合于 32 位及以上 CISC 指令系统. 但对较低位系统, 这种指令集动态转化的方法根本无法实现.

如何直接实现 CISC 指令流水线, 特别是在难度较大、应用面最广的 8 位 CISC 系统中, 一直是处

理器体系结构设计面临的难题. 实现 8 位 CISC 指令流水线难点主要如下

(1) 指令多字节、不定长, 取一条指令经常需要连续多次的存储器操作, 而指令流水线要求每个时钟周期取出一条有效指令.

(2) 指令集中没有专用存储器操作指令, 许多指令均可操作存储器, 使存储器存取操作频繁而没有规律可循.

(3) 数据通路中的寄存器和存储器不具有同时读、写性.

综上所述, CISC 指令流水线设计的根本问题是解决寄存器和存储器同时读、写冲突, 并合理安排流水线各阶段功能和任务, 优化流水线结构, 提高处理器整体运行效率.

当前对 8 位 CISC 系统指令流水线实现提出了一些设想,文献[3]中采用了放空流水线单元的做法,以保证对程序存储器的多次读取,这样造成处理器中关键部件经常处于空闲状态,不符合流水线设计的根本原则.文献[4-6]试图通过重点构建异步流水线来掩盖无法解决的流水线多次连续或不连续读取程序存储器的问题,构建的异步流水线取指阶段(IF)可能会因为多字节指令的多次读取存储器而占用多个时钟周期,使流水线中取指阶段与其他阶段的时间相差巨大,失去流水线建立的根本意义.另外,以上论文也没有涉及流水线中、后端同时读写寄存器和存储器冲突问题.

2 CISC 流水线设计

首先分析 RISC 指令集实现流水线设计的条件. RISC 限制对存储器的读写,只有 LOAD 和 STORE 指令可以读写存储器,其他指令均为寄存器类操作.因此,每一条流水线执行过程中,处理器对存储器的操作最多只有两次,第一次是在流水线前端的取指阶段,另一次是在流水线后端的读、写数据阶段.为此引入的哈佛结构,解决了两条流水线同时取指令和读写数据的冲突问题.围绕大量寄存器设计的 RISC 指令流水线,仍然存在流水线之间寄存器同时读写冲突的问题. RISC 设计了多总线结构,使寄存器组(Register File)具有多个读、写端口,解决了同时读、写寄存器冲突的问题. RISC 流水线设计成功的关键是解决了存储器和寄存器读写冲突问题.围绕通用寄存器组设计的运行方式,也极大地提高了处理器的运行速度.

2.1 流水线前端设计

CISC 指令流水线设计的首要问题是解决流水线前端取指操作和上一条指令取数据操作之间的冲突问题.分析 8 位 CISC 处理器的取指和指令执行过程如图 1 所示.受数据总线宽度影响,一条指令的取指和执行可能需要多次存储器操作.

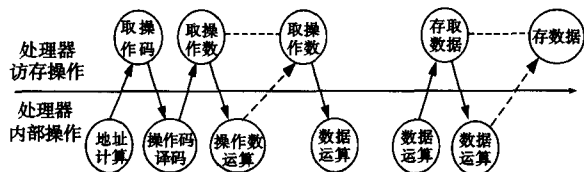
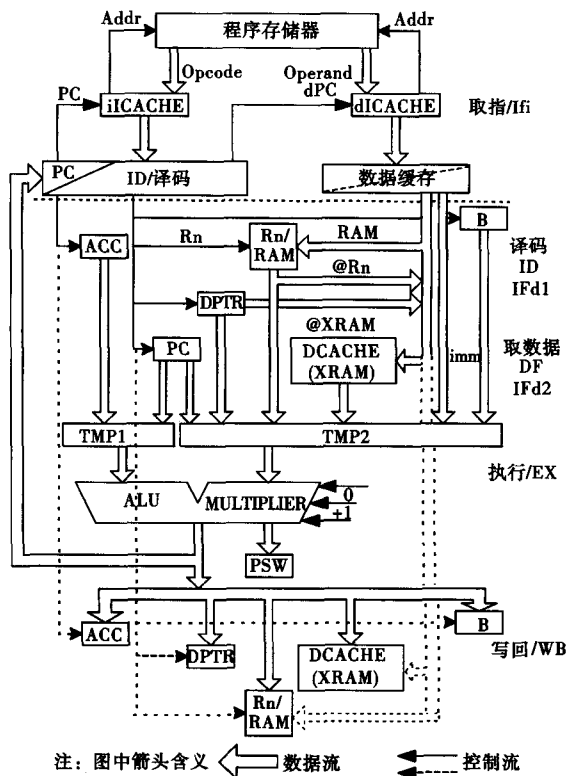


图 1 CISC 指令执行时处理器工作过程

设计提出适合于 CISC 指令集取指的新型结构:拓展的哈佛结构,即为程序存储器增加备份空间,同时通过拆分 PC 指针为操作码指针 PC 和操作

数指针 dPC 的方法,解决 CISC 指令频繁读取单一程序存储器造成的冲突问题.设计 CISC 典型系统 8 位 MCS-51 微控制器的五级流水线组织结构如图 2 所示.图中 iICACHE 和 dICACHE 分别代表指令操作码缓冲器和指令操作数缓冲器.指令流水线利用 PC 指针,在每一个时钟周期实现一次有效的指令操作码读取;在本条指令的译码(ID)阶段和取操作数(DF)阶段,利用 dPC 指针获取本条指令的操作数.在取指阶段的后半部,或译码阶段的前半部,通过映射而非解码,实现指令操作码指针 PC 的快速后移,并指向下一条指令的操作码. dPC 原则上指向 PC 的下一个字节,但是否紧跟 PC 或指向上一条指令的第二操作数则由上一条指令译码结果决定.对连续出现的 3 字节指令,采用编译调度技术^[7]解决指令操作数指针使用冲突问题,在调度无法解决指令冲突时,采用推迟后续流水线指令加载的延迟策略^[3].



注:图中箭头含义 ← 数据流 → 控制流

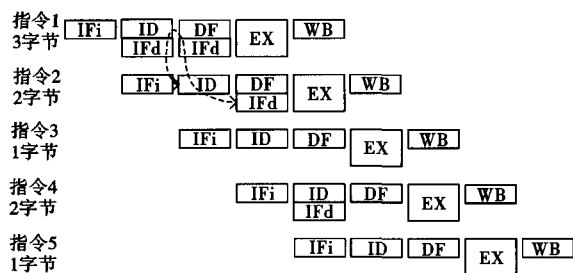
图 2 CISC 五级流水线组织结构

2.2 流水线中、后端设计

流水线中、后端设计主要解决寄存器和数据存储器读写冲突问题.由于 CISC 系统多是围绕存储器和 ACC 等特殊功能寄存器设计,一条指令的执行一般都存在对寄存器或存储器的多次读写问题,流水线设计将引发数据存取冲突.为了解决这种冲突,设计寄存器组为双总线结构,即具有一个读总线

端口和一个写总线端口,这种结构不仅能有效解决寄存器读写冲突,同时有利于硬件版图规划^[8]. 8 位 MCS-51 系统寄存器组包括累加器 ACC、程序计数器 PC 和 B 寄存器等,以及全部 RAM 区域. 设计使各种特殊功能寄存器与其功能部件分离,实现寄存器组统一规划和布局. 由于 XRAM 部分较大,设计采用多端口存储器,解决读写冲突问题^[9]. 图 2 中以分离的、具有同时存取数据能力的同名寄存器组和 DCACHE 来表示双总线结构的寄存器组和多端口 XRAM 结构. 设计寄存器组和存储器的同时并行读、写能力,解决了流水线中、后端的读写冲突问题.

根据五级流水线组织结构图,设计单周期 CISC 指令流水线操作时序图如图 3 所示. 对 MCS-51 全部 111 条指令执行过程分析可见,有 53 条指令全部占用流水线操作的 5 个阶段;50 条指令占用流水线操作的 4 个阶段(IF、ID、DF 或 WB 和 EX 阶段);仅有 8 条指令占用流水线的 3 个阶段(IF、ID 和 EX 阶段). 由此推算其理想静态 CPI 为 $(5+110)/111$, 流水线部件运行效率为 $(53 \times 5 + 50 \times 4 + 8 \times 3) / 111 \times 5 = 88.1\%$. 按照指令动态执行的 20%~80% 原则分析,流水线部件运行效率会更高. 图中虚线箭头代表指令操作数指针的移动.



IFi:取指令操作码; IFd:取指令操作码; ID:指令译码;

DF:取数据; EX:执行; WB:写回

图 3 单周期 CISC 流水线操作时序图

研究以难度较大的 8 位 CISC 系统 MCS-51 及其兼容系统为主,对于高位 CISC 系统指令流水线

的建立,也需要增加程序存储器备份和采用多端口数据存储器,以及扩展寄存器组总线等方法,解决读、写指令和数据冲突问题. CISC 指令流水线建立以后的问题,如流水线冒险、转移和数据冲突等,与 RISC 指令流水线并无实质性差别^[9].

3 仿真测试

按照 CISC 五级指令流水线组织结构和流水线操作时序图,设计用 VHDL 语言建模.模型借鉴了 Oregano Systems 提供的 MC8051 IP CORE,并在以下方面做了重要改动:

(1) 在实体 mc8051_top 的结构体中增加 ROM 备份元件 i_mc8051_drom, 增加 RAM 备份元件 i_mc8051_dram 和 XRAM 备份元件 i_mc8051_dramx, 以及相应的数据和地址变量, 以支持对存储器和寄存器组的同时读写操作。

(2) 改变状态机 FSM 译码模型为同步时钟上升沿驱动译码,并在取指部分映射下一指令操作码指针 PC(pc_o),根据指令译码结果推进指令操作数指针 dPC(dpc_o).

(3) 在实体 control_mem 结构体中增加对双存储器和寄存器组的读写控制指针和数据变量.

逻辑功能仿真选用 Mentor Graphics Corp. 提供的 ModelSim SE PLUS 6.0 软件, 图 4 所示为 CISC 五级指令流水线仿真波形图。由图可见, 在每一个时钟周期, 译码单元都从程序存储器中读取一条新的指令, 指令操作数指针 `dpc_o` 跟随指令操作码指针 `pc_o` 移动。另外, 仿真时在二进制程序代码中直接采用空指令 (NOP) 插在 3 字节指令后, 以模仿编译调度后的程序^[10-11]。

根据仿真测试结果,调整部分流水线阶段的任务,如对下一指令地址的确定,理论上它属于译码(ID)阶段,设计中将它调整到取指(IF)阶段,以均衡两段的任务量.优化流水线任务及结构后,采用

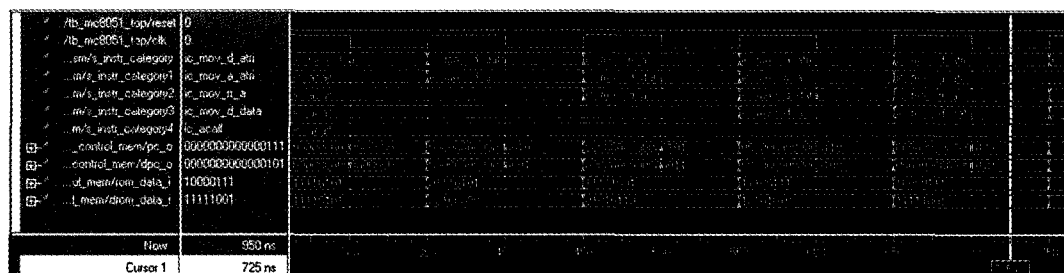


图 4 CISC 五级指令流水线仿真波形

XILINX 实现后仿测试^[9], 当主频在 120 MHz 时, 指令流水线仍能稳定运行.

4 结束语

CISC 系统具有指令密度高、存储空间少的特点. 当 Cache 大小固定时, CISC 系统中的 Cache 含更多的有效指令, Cache 命中率也相对较高, 所以存储器存取操作少, 处理器运行效率较高. CISC 系统广泛应用于当今台式机和微控制器系统中, 在计算机领域具有无法替代的优势地位. 在当前处理器速度的物理极限难以突破的情况下, 设计研究 CISC 指令流水线实现, 对于提高 CISC 系统指令执行速度, 降低指令 CPI, 具有重要的现实意义.

参考文献:

- [1] Hu Shiliang, Kim I, Mikko H, et al. An approach for implementing efficient superscalar CISC processors [C]//Proceedings of the Twelfth International Symposium on High-Performance Computer Architecture. Austin, Texas, USA; IEEE, 2006: 41-52.
- [2] 曹学飞, 张盛兵, 张骏. 32 位 CISC 微处理器流水线的设计[J]. 微电子学与计算机, 2007, 24(4): 186-189.
- [3] Intel Corporation. Intel architecture software developer's manual [D]. USA: Intel Corporation, 1999.
- [4] 牟刚, 贺前华, 江瑾. 基于流水线结构的高速嵌入式 MCU 设计[J]. 微电子学, 2004, 34(4): 425-431.
- [5] 潘琢金. C8051F020/1/2/3 混合信号 ISP FLASH 微控制器数据手册[EB/OL]. (2005-02-15). [2011-06-27]. <http://www.doc88.com/p-548082784.html>.
- [6] Lee J, Lee W, Cho K. A novel asynchronous pipeline

architecture for CISC type embedded controller [C]//Proceedings of the 45th Midwest Symposium on Circuits and Systems. Tulsa, Oklahoma, USA; IEEE, 2002: 675-678.

- [7] 王兵. 基于异步映射的流水线替换策略研究[D]. 上海: 上海交通大学, 2008.
- [8] Chen Changjiu, Cheng Weimin, Tsai Rueifu, et al. A pipelined asynchronous 8051 soft-core implemented with balsa [C]//Asia Pacific Conference on Circuits and Systems. Macao; IEEE, 2008: 976-979.
- [9] David A P, John L H. Computer architecture: a quantitative approach [M]. 4 th ed. San Francisco, USA: Morgan Kaufmann Publisher Inc, 2006.
- [10] Steve F 著, ARM SoC 体系结构[M]. 田泽, 于敦山, 盛世敏, 译. 北京: 北京航空航天大学出版社, 2002.
- [11] Wang Zuo, Shi Feng. N-Port memory mapping for LUT-based FPGAs [C]//Proceedings of the 17th ACM/SIGDA International Symposium on Field Programmable Gate Arrays. Monterey, California; ACM Press, 2009: 279-279.

作者简介:



靳文兵 男, (1968—), 博士研究生, 高级工程师. 研究方向为计算机体系结构.