一种基于 ASIC 实现的流水线架构 8051 内核设计

史大龙[1]，唐 建[1]，周远远[2]，黄 鲁[2]
(1. 中国科学技术大学 电子科学与技术系，安徽 合肥 230027；
2. 中国科学技术大学 信息科学实验中心，安徽 合肥 230027)

摘 要：以对传统 8051 微控制器的分析为基础，在保证指令集不变的情况下，给出了一种基于 ASIC 的微控制器设计。该设计采用三级流水线结构，提高了指令的执行效率。仿真和测试结果显示，所设计的 8051 内核可以正常工作。

关键词：微控制器；流水线；ASIC
中图分类号：TN47 **文献标识码**：A **文章编号**：1674-7720(2013)08-0018-03

Design of an 8051 core with pipeline structure based on AISC

Shi Dalong[1], Tang Jian[1], Zhou Yuanyuan[2], Huang Lu[2]
(1. Department of Electronic Science & Technology, University of Science & Technology of China, Hefei 230027, China;
2. Labrary Center of Science and Technology, University of Science & Technology of China, Hefei 230027, China)

Abstract: Based on the analysis of original 8051 and under the condition of keeping instruction set unchangeable, a MCU based on ASIC is designed. This design applies three stage pipeline to improve the effect of execution. This design of 8051 core can work well according to the simulation and test.

Key words: MCU; pipeline; ASIC

8051 微控制器至今具有非常广泛的应用。其指令集简单易懂，许多指令可直接访问 I/0 引脚，便于迅速操作外围设备。但在电路集成度和工作频率越来越高、SoC 结构越来越复杂的今天，高达 12 的 CPI (Cycle Per Instruction)使得指令的执行效率十分低下。因此，设计一个指令执行效率高且兼容全部 8051 指令集的硬件架构具有重要意义。

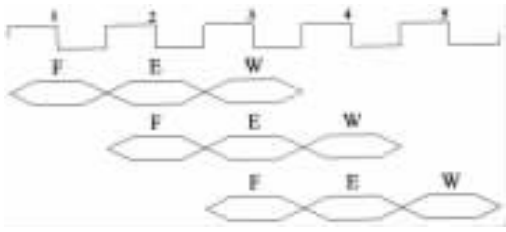
1 8051 的流水线设计

1.1 三级流水线的确定

按照计算机指令系统划分，8051 微控制器属于 CISC 架构。这种架构的特点是追求用最少的指令来完成所需的任务，因此它有数目庞大的指令集以提高编程效率；而由于不同功能的指令繁多，其代价是每条指令的长度和执行所占用的机器周期均不相同。与 CISC 架构对应的是 RISC 架构，虽然它只有少数指令，执行复杂运算时需要将多条指令组合操作，但这些指令绝大多数都可在一个周期内完成，有很高的指令执行效率[1]。因此，对 8051 微控制器进行某些类似 RISC 架构的改造，会使微控制器的执行效率得以提高。

比较常见的流水线结构有三级流水线结构和五级流水线结构。五级流水线结构一般包括取指令、译码、执行、访存和写回五级。其优点是把整个执行过程分散到 5 个时钟周期内，有助于提高硬件设计的主时钟频率；其缺点是由于前后指令之间的相关性，级数越大，每条指令的流水线之间的冲突就会越大，需要用一些“额外的”逻辑消除这些冲突或者在流水线队列中插入大量的“气泡”。三级流水线结构一般包括取指令、执行和写回三级。与五级流水线结构相比，其“执行”级包括了五极流水线中的“译码”、“执行”和“访存”三级，级数的减少有助于减少流水线中的冲突冒险，使得整个流水线结构相对简单，适用于比较简单、数据率较低的 SoC 系统中[2]。本设计采取三级流水线的结构，即在第一个时钟周期内指令 1 进行取指令；第二个时钟周期内指令 1 执行，指令 2 取指令；第三个时钟周期内指令 1 进行写回，指令 2 执行，指令 3 取指令。

8051 的指令集是 CISC 指令集，每条指令的执行(如图 1 中的 E 阶段)并不都是只需要一个时钟周期，为了方便问题的表述，在图 1 和后文中，将执行阶段描述为一个时钟周期。



F：取指令；E：执行；W：写回

图 1 三级流水线时序示意图

1.2 一种提高取指令效率的方法

如前文所述，8051 的指令集中每条指令的长度并不相同，如“CLR A”长度为1 B，“LJMP address”长度为 3 B。如果程序以字节为单位顺次在程序存储器中排列，那么在取指令阶段读取长字节指令时，就要花费多个时钟周期。因此，改变程序在程序存储器中的存储方式，使在取指令阶段能够在一个时钟周期内读出多长度的字节，对于提升微控制器的执行效率具有一定意义。

为此，采取将一个程序存储器分为 4 个子程序存储器的方式，子程序存储器 0 存储第 0、4、8、12…个字节，子程序存储器 1 存储第 1、5、9、13…个字节，子程序存储器 2 存储第 2、6、10、14…个字节，子程序存储器 3 存储第 3、7、11、15…个字节。这样可以将高 14 位作为各自子程序存储器的输入地址总线，完成一个周期内取出 4 B 的工作。这种方式与将整个存储器分为非 2 的整数次幂个子存储器的方式相比，其各个子存储器的地址产生逻辑简单容易得多。

当程序执行时，程序计数器向程序存储器发送某一个地址，通过下文给出的组合逻辑控制，一并读出以这个请求地址为首地址的连续 4 B 内容。由于 8051 的指令集中最长的指令是 3 B 指令，所以按照上述方法一定可以在一个周期之内将一条指令的全部字节取出。因为指令的第一个字节的内容可以唯一确定该指令的长度，所以，不仅可以判断出在取出的 4 B 中究竟有多少条属于此指令，而且可以预测出下一条指令的地址，为下一周期的取指令做好准备[3]。

例如程序存储器的输入地址 Addr=0x0021 时，预期读出第 0x0021、0x0022、0x0023 和 0x0024 共 4 B，这种情况下，0x0021、0x0022 和 0x0023 字节在各子程序存储器中的地址为 2’b00 0000 0000 1000(即 Addr 的高 14 位)，而 0x0024 在子程序存储器 0 中的地址为 2’b00 0000 0000 1001(即 Addr 的高 14 位加 1)。因此，将高 14 位地址 Addr[15:2]和 Addr[15:2]+1 作为二选一选择器的两个输入，Addr[1]和 Addr[0]作为选择器的控制端，即可实现上文提到的一个周期内依次取出 4 B 的操作。各子程序存储器的输入地址如表 1 所示，改进后的程序存储器如图 2 所示。

表 1 4 种情况下各子程序存储器的输入地址

ADDR[1:0]	2’b00	2’b01	2’b10	2’b11
ROM0 输入地址	Addr[15:2]	Addr[15:2]+1	Addr[15:2]+1	Addr[15:2]+1
ROM1 输入地址	Addr[15:2]	Addr[15:2]	Addr[15:2]+1	Addr[15:2]+1
ROM2 输入地址	Addr[15:2]	Addr[15:2]	Addr[15:2]	Addr[15:2]+1
ROM3 输入地址	Addr[15:2]	Addr[15:2]	Addr[15:2]	Addr[15:2]

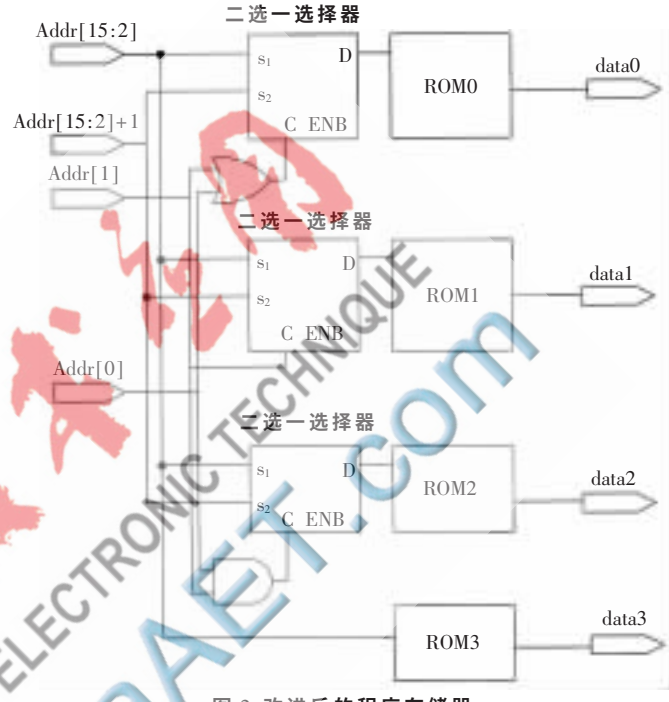


图 2 改进后的程序存储器

1.3 流水线冲突冒险的处理

在程序计数器预测下一指令的地址时，采取的方式是根据当前已经读取指令的地址和长度来计算下一指令的地址，当第 N 条指令执行完毕，程序需要发生跳转时(如图 3 时刻 1)，第 N+1 条指令已经完成取指令，而这条已经取过的指令是按照无跳转情况发生的情况下的“下一条指令”，即此时这种预测策略就会发生错误。但这条指令只完成了取指令，并没有执行，在此，设置一个周期的等待信号给执行模块，将执行模块暂停一个周期(时刻 1 和时刻 2 之间)，一个周期后，程序完成跳转，跳转后的指令进入流水线。

另外的一种流水线冲突发生在后一条指令要读取前一条指令运算结果的时刻，如图 4 所示。在这种情况

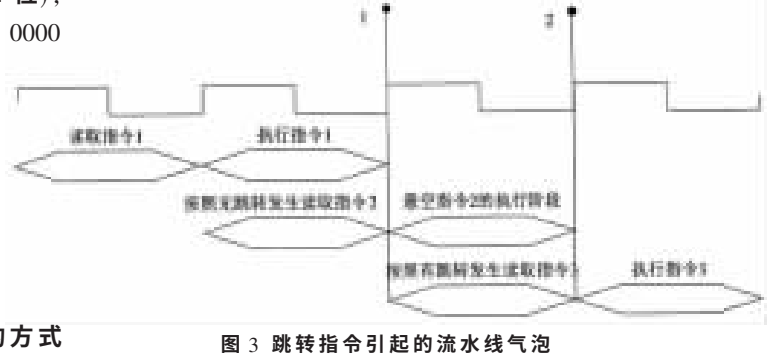


图 3 跳转指令引起的流水线气泡

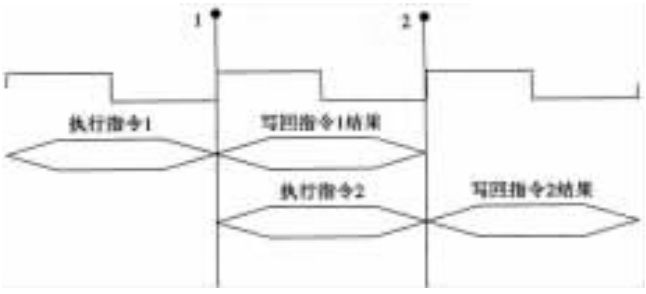


图 4 同一时刻对同一地址进行读取造成的流水线冲突

下,前一指令执行阶段完成,即将进入写回阶段;后一条指令等待执行,需要从合适的位置读取操作数。这时,前一条指令的结果尚未写入到数据存储器中,后一条指令读取的是更新前的操作数(时刻 1)。因此,如果在某一个时钟周期对同一个地址进行读写,那么写操作照常进行,而读操作则不经过存储器直接读取即将写入存储器的数据,这样就能避免读取尚未更新的数据的问题[4]。

2 仿真及流片测试

用 Keil4(单片机系列的编译软件)对 C 代码和汇编代码进行编译链接产生二进制的机器码,存入程序存储器作为被执行程序以供仿真,多个机器码覆盖 8051 指令集的全部指令。

图 5 所示为取指令阶段仿真波形图。pc 指示指令的物理地址,该物理地址为首地址的 4 B 内容被读出,并将前 3 字节(8051 最长指令)依次赋给 operator1、operator2 和 operator3,跟据 operator1 的内容,可以判断 op_len(即该指令的长度值)并计算下一指令的 pc 值。

图 5 关于取指令实现和解决流水线冲突的部分仿真结果

在 pc=0x73 周期,完成取指令"SJMP EE"(对应机器码为 0x80,0xEE);下一个时钟周期按照无跳转发生的情况完成下一条指令的取指,同时执行这个跳转指令;执行结束后,程序跳转到 0x63(0x73+指令长度+0xEE)这一正确位置继续执行。

经仿真验证分析,该设计可以解决程序跳转以及对同一地址进行读写等易造成流水线冲突的问题,并执行出正确的结果。

将此设计在 SMIC 180 nm CMOS 工艺下进行综合,可实现时钟频率为 250 MHz 的情况。进一步的性能提升(如提高频率、降低功耗)可通过改善集成电路工艺实现。

流片后测试应用 Altra 公司的 Stratix-Ⅱ FPGA 芯片和自制的开发板产生测试向量(即微控制器的输入信号),用逻辑分析仪捕捉芯片的输出信号。经测试,流片后的微控制器可以根据二进制机器码正确执行。

本文提出了一种流水线结构的 8051 设计,采用三级流水线结构,在指令集不变的情况下提高了指令执行效率,满足工程实际需要。该设计可以作为单独的控制器微芯片使用,也可和其他 IP 一起进行 SoC 设计。

参考文献

[1] GOLZE U. VLSI chip design with the hardware description language Verilog[M]. 北京:北京航空航天大学出版社,2005.

[2] RAMIREZ A, SANTANA O J, LARRIBA-PEY J L, et al. Fetching instruction streams[C]. Proceedings of the 35th Annual ACM/IEEE International Symposium on Microarchitecture, 2002:371-382.

[3] SIMSIC J, TERAN S. 8051 core specification[DB/OL]. (2001-09-25)[2013-01-21]. http://www.opencores.org.

[4] 倪继利,陈曦,李挥. CPU 源代码分析与芯片设计及 Linux 移植[M]. 北京:电子工业出版社,2007.

(收稿日期:2013-01-21)

作者简介:

史大龙,男,1988 年生,硕士研究生,主要研究方向:数字集成电路设计。

唐建,男,1972 年生,讲师,主要研究方向:智能信息处理。

周远远,男,1978 年生,实验师,主要研究方向:电子测试。