

嵌入式处理器的性能分析方法^{*}

盛永杰¹ 刘博勤² 陈章龙¹

(复旦大学计算机科学与工程系 上海200433)¹ (西南师范大学计算机与信息科学学院 重庆400715)²

Performance Analysis of Embedded Processor

SHENG Yong-Jie¹ LIU Bo-Qing² CHEN Zhang-Long¹

(Department of Computer Science & Engineering, Fudan University, Shanghai 200433, China)¹

(College of Computer and Information Science, Southwest Normal University, Chongqing 400715)²

Abstract This paper introduces the architecture of current embedded processors firstly. It focuses on some important factors that affect performance of embedded processor, and proposes some performance analyzing methods. At the last part, a generic performance analysis system is illustrated to clarify the idea mentioned before with a short example.

Keywords Embedded, CPU, Performance, Analysis

1 引言

嵌入式处理器的应用主要包括两个部分,一部分是工业应用,比如:自动化流水线、核电站的自动报警系统及应急处理等,这部分对处理器的实时响应、稳定性的要求一般来说比较高。另一部分是生活应用,比如:智能手机、PDA、信息家电和个人汽车上的自动控制等,无处不用到嵌入式处理器,它们对处理器的性能要求也越来越高。因此,如何评估嵌入式处理器的性能是一个重要的课题。

要提高嵌入式处理器的性能,有两个办法,一是提高处理器的主频,二是提高处理器运行的效率。提高主频首先是受到制造工艺的限制,主频不能无限制地提高,另外提高主频也会使处理器价格大幅度上升。提高处理器运行效率是在基本不增加开销的情况下,尽量减少处理器的空闲时间。所以不管处理器的主频有多高,提高处理器的运行效率总是受欢迎的。本文的重点在于如何评估一个处理器的运行效率。

目前嵌入式处理器以精简指令集计算机(RISC)为主,并且普遍采用流水线技术来加快执行部件的处理速度。

由于处理器和主存之间速度的不均衡,处理器的运行速度大大高于主存的存取速度,缓存技术被引入来提高处理器的性能。缓存是利用程序的局部性原理工作的,由于程序运行时,在一个短的时间内所访问的指令和数据都是相对簇聚的,如果我们能够把要访问的这一部分主存数据取出放在一个与处理器速度接近的临时小块存储器当中,就可以极大地改善处理器的性能,而这个临时小块存储器就是缓存。

2 性能分析

处理器的性能主要与流水线、缓存、快表的性能相关。当流水线由于各种相关而暂停时,处理器的性能下降;缓存访问失效时,处理器要等待一个主存访问时间;快表访问失效时,先要访问页表,再访问相应的地址,访问次数增加了,并且两次访问同样有缓存失效的可能。所有这些因素都极大地影响着处理器的性能。

对处理器性能的评价一般通过测试来进行,比如运行各种标准测试程序,来获得如CPI(Circle per Instruction)等总体性能参数;不能对具体的细节尤其是体系结构相关的细节进行测试,而这对于性能分析来说是不够的,要执行与体系结构相关的详细测试,需要在处理器内部增加相关测试逻辑电路。

2.1 流水线

当引入流水线以后,产生了一系列的新问题,在理想状态下,流水线满负荷运行,一个时钟周期执行一条指令,其效率为100%;但实际上,有许多因素影响流水线的效率,导致流水线暂停,这些因素可以分为三大类:数据相关、结构相关、控制相关。处理器的设计者的工作是消除或尽可能减少这些暂停,但每一种实现方法都是有代价的,其可行性往往取决于该方法的实现代价和它对流水线效率的影响。所以,统计各种因素对性能的影响的具体参数是十分必要的,它是处理器设计者的重要参考,也是编译器优化的重要依据。

流水线效率的测试,统计在一个测试过程中,特定类型的暂停事件发生的次数,每次暂停的时钟周期数,总的运行指令数。事件影响率:

事件影响率 = 事件次数 × 每次周期数 / 总的周期数

2.2 缓存

缓存通常是存储器层次结构中离CPU最近的存储层次,当CPU要访问的指令或数据不在缓存中时,产生缓存失效,缓存会通过一定的映射方法把主存中的相应块放置到缓存中来,由于程序指令和数据访问的局部性,配合缓存块的预取,缓存可以达到比较高的命中率。当缓存发生不命中时,由于受主存速度较慢以及主存和缓存之间的带宽有限的影响,一次失效的代价往往是几十个时钟周期,这一点与流水线暂停的几个周期有明显区别。

缓存失效率可以通过统计一个测试过程中缓存的特定类型失效次数,每一次失效的开销时钟周期数,总的运行时钟周期数来计算。

缓存失效率 = 失效次数 × 单次开销 / 总的周期数

^{*} 国家863项目“32位高速嵌入式CPU平台技术研究”(课题编号:2002AA1Z1009)。盛永杰 硕士生,研究方向为单片微型机应用系统设计。刘博勤 副教授,陈章龙 教授,主要从事嵌入式系统结构开发和应用。

2.3 快表

采用分页机制的存储器的页表一般很大,快表是页表的一个专用的高速缓冲区,它与缓存的原理类似,也利用了存储器访问的局部性原理。在采用虚存机制时,每一次访问存储器都要通过快表进行虚实地址转换,故快表的性能非常重要。

快表失效率的计算相对简单,它由快表访问不命中的次数及快表访问的总次数确定,计算公式如下:

快表失效率 = 不命中次数 / 总访问次数

综上所述,我们可以总结出如下一些处理器性能参数:1) CPI, 总的运行周期数/总的被执行的指令数。2) 指令 Cache 失效率, 指令 Cache 失效次数/总的被执行的指令数。3) 数据 Cache 失效率, 数据 Cache 失效次数/数据 Cache 访问次数。4) 等待取指延迟, 指令 Cache 失效次数/指令 Cache 无法提供指令。5) 数据写回比率, 数据 Cache 写回次数/数据 Cache 访问次数。6) 指令 TLB 失效率, 指令 TLB 失效/总的被执行的指令数。7) 数据 TLB 失效率, 数据 TLB 失效/数据 Cache 访问次数。

3 一个通用性能分析系统的构造

性能分析系统设计的基本原则:通用原则,应该提供一些基本的分析参数,而不是仅仅针对某一、二项性能指标,这样可以更好地适应不断发展和变化的分析需求,以及来自不同的应用领域的分析要求;精简原则,要求附加的硬件开销要尽可能的小,对处理器正常事务的处理的影响要尽可能小。

在处理器全速运行的情况下,采集处理器内部运行数据,可以有两种基本方法:一是在处理器内部增加新的计数寄存器,用来统计各种运行数据;二是增加协处理器专门用来统计运行数据。第一种方法会增加处理器内部的复杂度,不便于以后系统的更新、升级;第二种方法则可以尽可能地增加处理器内部的处理逻辑,并且可以增加一些附加处理,便于系统的维护升级。也可以将二种方法结合起来使用。

性能监测寄存器的设置,第一种方法为每一个基本参数设置一个寄存器,这样寄存器名空间会比较大,但可以同时测量多个参数;第二种方法设置几个通用参数寄存器,通过模式位的设置来控制所获取的参数种类,这种方法的缺点是一次只能测量部分运行参数,如果测试程序的运行流程是可重复的,可以通过重复执行来依次获得其它运行参数。

我们倾向于采用第二种方法,因为一般测试程序是可重复的,而且有些时候监测用户并不对所有参数感兴趣,图1是性能监测寄存器的结构。性能监测事件如表1所示。

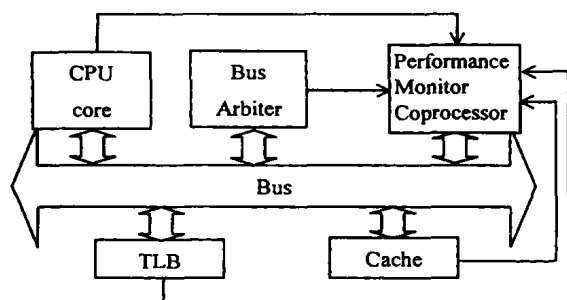


图1 性能监测寄存器的结构

表1 性能监测事件

事件号	事件
1	测试期间运行的总指令数
2	指令 Cache 失效次数
3	访问数据 Cache 的次数
4	数据 Cache 的不命中次数
5	不可缓存的指令 Cache 的访问次数
6	数据 TLB 的总访问次数
7	数据 TLB 的不命中次数
8	指令 TLB 的总访问次数
9	指令 TLB 的不命中次数
10	流水线停顿周期数
11	数据 Cache 写回次数
12	总线上各功能单元的阻塞周期数

4 应用及举例

一个合适的性能分析系统,提供了对处理器运行状况的真实记录,将这些记录与性能预测模型作对比,可以优化硬件的设计,并且可以优化对处理器的使用,比如指导编译器的优化设计。

Intel 公司的 XScale 芯片,片上协处理器 CP14 提供四个专用寄存器(R0~R3),来采集处理器运行期间的统计数据。其中 R0 为控制寄存器,控制 R1~R3 的工作模式;R1 为时钟计数器,用来统计处理器的总运行时钟周期数;R2 和 R3 为性能计数器,可以用来同时统计两种不同的性能参数。

计算 Xscale 指令 Cache 效率的例子如下:

```

; 设置性能监测模块
; 配置控制寄存器的模式值
MOV R0, #0x7777
MCR P14, 0, R0, C0, c0, 0 ; 写控制寄存器 CP-R0
; 以下开始计数
...
; IRQ 处理部分
MRC P14, 0, R1, C0, c0, 0 ; 读运行时时钟周期数
BIC R2, R1, #1
MCR P14, 0, R2, C0, c0, 0 ; 清中断并且暂停计数
MRC P14, 0, R3, C1, c0, 0 ; 读 CP-R1
MRC P14, 0, R4, C2, c0, 0 ; 读 CP-R2
MRC P14, 0, R5, C3, c0, 0 ; 读 CP-R3
; 结果处理部分
SUBS PC, R14, #4 ; 从中断返回
; 计算结果示例
; 我们不妨假定一下统计的结果
CP-R1 = 0x0000, 0020 ; 假定发生过一次溢出中断
执行的指令数 = CP-R2 = 0x6AAA, AAAA
指令 Cache 失效次数 = CP-R3 = 0x0555, 5555
指令 Cache 失效率 = 100 * CP-R3/CP-R2 = 5%
CPI = (CP-R1 + 2^32) / 执行的指令数 = 2.4 cycles/instruction
计算得到指令 Cache 失效率为 5%, CPI 为 2.4。
  
```

结论 通过这些具体的处理器性能分析量化参数,我们可以确定一个嵌入式处理器在运行一般程序时,尤其是特定的应用时,各种影响处理器性能的因素所占的比例,从而可以有针对性地采取优化措施,比如可以定制编译器的优化特性,也可以反过来指导嵌入式处理器本身的设计,特别是某种专用领域的处理器的设计。

参考文献

- 1 Patterson D A, Hennessy J L. Computer Architecture. Morgan Kaufmann Press. 机械工业出版社,影印(英文版,第二版),1999. 9
- 2 陈章龙,涂时亮. 嵌入式系统——Intel StrongARM 结构与开发. 北京航空航天大学出版社,2002. 10
- 3 Intel. Intel XScale Microarchitecture for the PXA250 and PXA210 Application Processors. 2002. 2