

基于 ARM 指令集的通用 DSP 中指令相关处理方法

王 旭,付家为,何 虎

(清华大学 微电子学研究所,北京 100084)

摘 要: 针对现有处理器中指令相关检测模块占用面积大、延时长和功耗高等问题,提出了一套用于通用数字信号处理器中指令相关的检测方法,并且提出了指令相关的解决办法. 通过设定一组锁定标识,可以检测同一周期内多条指令能否同时发射,也可以检测不同流水级的各组指令之间的相关性,实现指令的动态调度. 利用超长指令字(VLIW)与超标量融合的方式,在不增加硬件开销的前提下,解决指令相关的问题. 经过基于 GEM5 的仿真平台进行验证,结合实例证明了该方法的可行性.

关键词: 指令相关; 锁定标识; 动态调度; 超长指令字

中图分类号: TP332

文献标识码: A

文章编号: 1000-7180(2016)09-0010-05

Instructions Dependence Dispose Method in General Purpose Digital Signal Processor Based on ARM Instruction Set Architecture

WANG Xu, FU Jia-wei, HE Hu

(Institute of Microelectronics, Tsinghua University, Beijing 100084, China)

Abstract: In view of instructions dependence detection module with large area, long time delay and high power consumption features in existing processor, a new instructions dependence detection and dispose method for Digital Signal Processor (DSP) is proposed to solve the problem. The method could detect the dependence between instructions in the same cycle, and also instructions dependence in different pipelines by setting a set of locking signal. Dynamic scheduling is implemented in this way. Using the fused microarchitecture integrating both Very Long Instruction Word (VLIW) and superscalar could dispose the dependence without extra hardware. The feasibility of this method is proved by examples based on the GEM5 simulator system.

Key words: instructions dependence; locking signal; dynamic scheeduling; VLI

1 引言

超标量处理器为了实现指令级并行,常使用动态调度的方法^[1]. 在指令的顺序执行过程中,除了发射槽的宽度之外,指令之间存在的^{数据相关}成为了制约最大限度的指令级并行的重要因素. 检测同一周期内的指令之间是否存在数据相关是实现指令动态调度的关键. 在超标量处理器的体系结构设计中多采用流水线技术,实现指令级的并行后,检测相邻流水级的各组指令之间是否存在数据相关也是确保程序能够正确执行,动态调度指令的关键.

传统的检测方法使用的是寄存器依赖表,该表存储了每一条正在执行的指令的目的寄存器信息. 在经过解码后,每条指令在进入下一个流水级的同时,其目的寄存器信息写入寄存器依赖表,以供判定后续指令发射的可行性^[2]. 这种检测方法的硬件开销成本比较大,处理器的发射数增多时,会很大程度增加模块的面积,带来功耗的问题,相关逻辑会占用很大的资源.

本文以基于 ARM 指令集的通用数字信号处理器(DSP)作为项目背景,提出了一种处理器中硬件开销小的指令相关性处理的方法,主要包括相关性

检测和相关性的解决两个方面. 相关性检测方法不仅适用于检测同一周期内多条指令能否同时发射,也适用于检测不同流水级的各组指令之间的相关性. 而且方法的实现并不会涉及很复杂的电路设计,能够很大程度上减小硬件开销. 同时,利用超长指令字与超标量融合的方式对程序的指令进行优化,在不增加硬件开销的前提下,解决程序中指令相关的问题,可以降低功耗,提升处理器的性能.

2 基于 ARM 指令集的通用 DSP 设计

近年来,许多片上系统(SOC)使用 ARM+DSP 这种体系结构^[3]. 例如,Integra 处理器使用 ARM Cortex-A8 处理器和高性能 DSP. ARM 核与 DSP 核支持不同的指令集,需要各自独立的开发环境,这样会增加软件开发的复杂度. 两个核之间的通信也将成为一个难点. 基于 ARM 指令集的通用 DSP 可以取代这种双核结构,使用单核结构既可实现处理器的通用性,又可以将处理器运用到数字信号处理领域. 同时为用户提供一个统一的开发环境,减少软件的开发时间和成本.

2.1 ARM 指令集简介

ARM 指令的一般编码格式如图 1 所示,每条 ARM 指令占有 4 个字节,指令长度为 32 位.

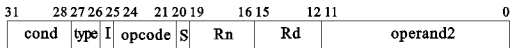


图 1 典型 ARM 指令编码格式

其中,cond:指令条件执行的编码;type:指令类型码;I:标记第二操作数是否为立即数类型;opcode:指令操作码;S:决定指令的操作是否影响状态寄存器 CPSR;Rn:指令第一个源操作数的寄存器编号;Rd:指令目的寄存器编号;operand2:指令的第二个操作数.

数据处理指令的 oprand2 有三种构成方式:立即数方式、寄存器方式和寄存器移位方式,分别对应不同的寻址方式^[4]. 指令的类型主要分为数据处理指令、分支指令和存储/加载指令等. 对内存进行存储/加载数据操作的指令 Load/Store 的地址计算方法主要分为寄存器间接寻址和基址加变址寻址.

2.2 通用 DSP 简介

为了兼容 ARM 指令集,在清华大学高性能处理器实验室自主研发的 Lily2 处理器的基础上开发出一款新型处理器,命名为 Lotus. 超标量处理器程序兼容性好,但硬件开销大,功耗高. 超长指令字处理器便于低功耗低面积实现指令的高并发,缺点是

需要软件编译,通用性差. Lotus 将两种处理器的优势结合,可在低功耗低面积的前提下实现最大限度的指令级并行. 该处理器具有如下特点:

(1)处理器包括六个独立的功能单元,A、M、D 单元各 2 个. 其中 A 单元为算术逻辑功能单元,执行加减法算数操作,移位操作和逻辑运算操作. M 单元为乘法功能单元,执行乘法,乘累加,点积等运算,也可以执行部分算术和逻辑等运算. D 单元为数据传输功能单元,执行存储器访问等指令.

(2)处理器支持 32 位的指令格式,并且支持条件执行的指令.

(3)10 级指令流水线,指令执行效率高.

(4)具有 15 个通用寄存器,一个状态寄存器 CPSR 和 PC.

该款高性能数字信号处理器采用混合架构的模式,能够实现顺序超标量模式的双发射和超长指令字模式下的六发射,在处理实时交互事务以及信号处理等密集运算事务方面均有优秀的性能表现.

3 指令相关检测方法

3.1 指令相关类型

现代处理器设计中常采用流水线技术,这就势必会带来一些指令相关性冲突的问题. 指令相关性主要分为控制相关、资源相关和数据相关^[5]. 其中控制相关可以通过延迟跳转技术和分支预测的方法得以解决^[6]. 而在 Lotus 处理器中并不存在资源相关的问题,所以数据相关的处理成为关键.

数据相关的产生是由于在多条指令同时执行的过程中后面的指令会用到前面指令的执行结果. 根据指令间对同一寄存器读和写操作的先后顺序,可将数据相关性的类型分为读后写相关、写后读相关和写后写相关三种. 为了确保程序的正确执行,必须有相应的方法将这些数据相关检测出来.

3.2 现有数据相关检测方法

现有典型数据相关的检测方法是使用寄存器依赖表,在 Lily2 处理器中正是使用了这种方法. 该表存储了每一条正在执行的指令的目的寄存器信息. 在经过预解码后,每条指令在进入下一个流水级的同时,其目的寄存器信息写入到寄存器依赖表,以供判定后续指令是否可以同时发射. 后续指令预解码后,其源寄存器、目的寄存器必须与寄存器依赖表中的表项进行冲突检测,得到的结果作为该指令是否可分发的条件. 这种检测方法确实可行,但是硬件开销的成本比较大,并且预解码模块和冲突检测模块

在同一流水级,会很大程度增加整体模块的面积,带来功耗的问题,相关逻辑会占用很大的资源.

3.3 设定锁定标识组的检测方法

针对现有典型数据相关检测方法的不足,本文提出一种新的数据相关检测方法.通过设定一组锁定标识,锁定标识与处理器内的通用寄存器组一一对应,每个通用寄存器都有一个对应的锁定标识.指令解码后,将其源操作数与目的操作数分别与对应的锁定标识相比较,来判断指令的相关性与并行性.锁定标识根据每个周期指令的解码信息进行更新,每个周期更新一次.这种方法不仅适用于检测同一周期内多条指令能否同时执行,也适用于检测不同流水级的各组指令之间的相关性,为实现指令的动态调度提供了基础的保证,而且该方法的实现并不会涉及很复杂的电路设计,能够很大程度上减小硬件开销.

在 Lotus 处理器中,指令数据相关性检测的流程如图 2 所示,具体包括以下几个步骤:

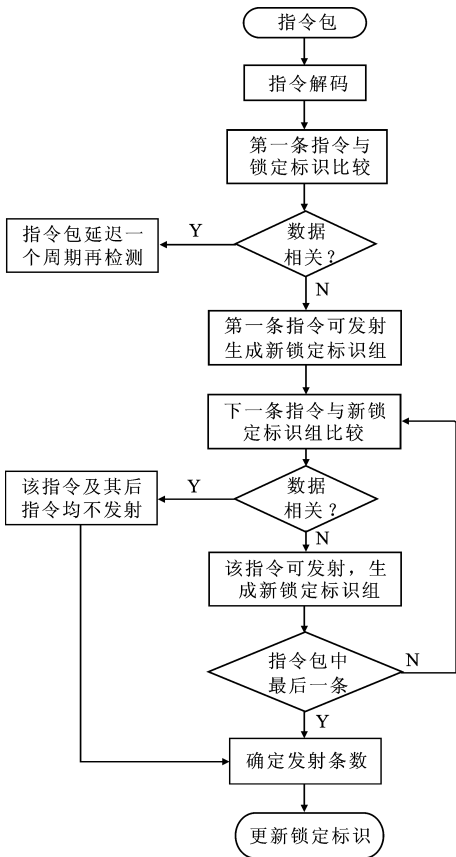


图 2 数据相关性检测流程图

(1)每个指令包中含有两条指令,将这两条指令分别进行解码,得到第一条指令与第二条指令的源操作数的寄存器编号,目的操作数的寄存器编号以

及指令执行所占用的周期数目.

(2)将指令包中的第一条指令的源操作数和目的操作数分别与对应的锁定标识进行匹配.如果该条指令用到的源操作数或者目的操作数被锁定,那么该指令包的两条指令需要在下一周期与更新后的锁定标识重新进行匹配,该周期可同时发射的指令条数为 0;否则,第一条指令可以在这个周期发射,同时将该指令的执行周期数传递给该指令目的寄存器对应的锁定标识,其余的锁定标识保持原来的值不变.这样第一条指令经过检测后,生成了一组新的锁定标识,记为锁定标识组 1.

(3)将指令包中的第二条指令的源操作数和目的操作数分别与锁定标识组 1 中的锁定标识进行匹配;如果该条指令用到的源操作数或者目的操作数被锁定,那么该指令不能和它前面的指令一起发射,需要延后一个周期再做判断,该指令将会成为下一周期的指令包中的第一条指令.这样确定了该周期可以发射的指令条数为 1;否则,指令包中的这两条指令与之间不存在数据相关,与上一周期发射的指令之间也不存在数据相关,可以同时发射.将该指令的执行周期数传递给锁定标识组 1 中目的寄存器对应的锁定标识,锁定标识组 1 中其余的锁定标识保持不变;这样得到了锁定标识组 2,确定了该周期可同时发射的指令条数为 2.

(4)根据每个周期可同时发射的指令条数,得到其对应的锁定标识组的编号.在下一个周期的开始,将该编号对应的锁定标识组的值各减去 1 赋给相应的锁定标识,实现锁定标识的更新.

结合实例介绍一下基于锁定标识组的检测方法在实际运用中的实现.表 1 是基准测试程序 dsp-stone 中的 matrix 算法的其中一段指令代码,指令所属的功能单元与执行时刻已经标出.通常来说,A 单元执行指令需要一个时钟周期,M 单元执行指令需要两个时钟周期,D 单元执行指令需要三个时钟周期.其中存在数据相关的部分已标出.

第一个指令包中的两条指令并不存在数据相关,可以并发,同时执行.但 ldr 指令需要 3 个时钟周期才能将结果写回到 r0,而下一个指令包中的两条指令均需用到 r0 的值,对于这种写后读相关,r0 对应的锁定标识需要 3 个周期才能解除锁定,两条加法指令在第 4 个时钟周期可以同时执行.同样的,对于第四个指令包中的 ldr 指令,r4 对应的锁定标识也需要 3 个周期才能解除锁定,所以在第 9 个时钟周期执行 mov 指令,而其间的 ldr 指令没有用到

被锁定的寄存器 r7,因而可以顺序执行.按照这种规则,指令的执行过程与顺序如表 1 所示.在第 16 个时钟周期执行 ldr 指令,在第 18 个时钟周期执行完毕.通过设定锁定标识组,检测指令的数据相关,实现指令的动态调度.

表 1 超标量模式下指令执行顺序与周期序列

周期序列	执行指令	所属功能单元
1	mov r9, #0	A
	ldr r0, [pc, #136]	D
4	add fp, r0, #800	A
	add sl, r0, #832	A
5	add r6, fp, r9, lsl, #2	A
	ldr r8, [fp, r9, lsl, #2]	D
6	ldr r4, [pc, #120]	D
	mov r5, r0	A
7	ldr r7, [r6, #36]	D
9	mov r3, r4	A
	mov r1, r6	A
10	ldrr2, [r3, #-32]!	D
13	mul r2, r2, r8	M
15	str r2, [r5], #4	D
	ldr ip, [r3, #4]!	D
16	ldr lr, [r1, #4]!	D
	cmp r3, r4	A

4 解决指令相关的方法

在 Lotus 处理器中使用分支预测的方法解决了指令的控制相关.单个周期内指令的执行条数不会超过功能单元个数,因而不会发生资源相关.详细介绍一下数据相关的解决办法.

解决数据相关冲突的常用办法有两种,分别是通过编译的手段和数据旁路技术.编译程序通过在两条相关指令之间插入无关指令或者空操作指令而推迟指令的执行,这种方法对编译器的要求较高,容易降低执行效率.数据旁路技术通过内部数据通路把下一条指令用到的结果在写入寄存器之前就直接传递过去.效率较高,但是控制复杂,并且增加了硬件的开销.这两种方法各有利弊,使用超长指令字和超标量融合的方式能将两者的优势结合起来,在不增加硬件复杂度的前提下,解决数据相关问题.

在实际指令的执行过程中,进行复杂运算的指令往往集中在程序的某一段,并且其功能经常是重复性的运算,其中不乏存在大量的数据相关.如果能将这一段代码进行超长指令字的优化,提高指令的并行性,并且通过静态调度的方式最大程度上消除数据相关,那么将极大提高指令的执行效率.

超长指令字和超标量两种模式的切换可以通过

MSR 和 MRS 这两条指令实现,通过修改控制寄存器 CPSR 的第 23 位实现模式切换的控制.超长指令字模式支持指令的六发射,优化后的指令如果能够满足一定的顺序,即可最大限度的发挥超长指令字模式的优势,缩短程序的执行时间.

结合优化实例详细介绍一下超长指令字和超标量融合的方法.如果表 1 的指令完全在超标量模式下运行,动态地检测指令相关性,执行完这段代码共需 18 个时钟周期.如果对此段代码进行超长指令字模式的优化,对指令执行顺序进行重新编排与打包,得到的结果如表 2 所示.超长指令字模式下在分发级会检测指令包中指令所属功能单元的排序,如果满足 A>M>D 这种排列顺序,则这些指令可以同时执行.str 指令虽然为 D 单元指令,但是 1 个时钟周期后即可解除对目的寄存器的占用.在插入 3 个 nop 指令的情况下,优化后的代码只需要 10 个时钟周期即可完成同样的操作.

表 2 优化后指令执行顺序与周期序列

周期序列	执行指令	所属功能单元
1	mov r9, #0	A
	ldr r0, [pc, #136]	D
	ldr r4, [pc, #120]	D
2	nop	NOP
3	nop	NOP
4	add fp, r0, #800	A
	add sl, r0, #832	A
	mov r5, r0	M
	mov r3, r4	M
5	add r6, fp, r9, lsl, #2	A
	ldr r8, [fp, r9, lsl, #2]	D
	ldr r2, [r3, #-32]!	D
6	mov r1, r6	A
	ldr r7, [r6, #36]	D
	ldr ip, [r3, #4]!	D
7	ldr lr, [r1, #4]!	D
8	cmp r3, r4	A
	mul r2, r2, r8	M
9	nop	NOP
10	str r2, [r5], #4	D

由此可见,优化后不仅消除了指令的相关性,也提高了指令的并行性,同时提高了执行效率.将这种方法扩展到一些运算密集型的程序序列中,对核心部分代码进行优化,将缩短整个程序的执行时间,提升处理器的性能.

5 实验结果

5.1 FPGA 综合结果

使用综合工具 Vivado,选取型号为

XC7K325T-2FFG900C 的 FPGA 对设计进行综合验证,设定锁定标识组的检测模块占用的 LUTs 的个数仅为 100,而且这一模块并不会影响关键路径.可见这种检测方法并不会占用很多的资源,模块占用的面积也很小,却可以高效地检测指令之间的相关性,实现指令的动态调度.

5.2 GEM5 仿真结果

基于超长指令字和超标量两种模式的融合,可以通过模拟器 GEM5 作为实验仿真平台来进行验证. GEM5 是一款集成了多个指令集系统和 CPU 模型的模块化离散事件驱动的模拟器^[7-8]. 在 GEM5 的基础上,加入超长指令字模式,实现两种模式融合的处理器的架构.

仿真实验采用 DSPStone^[9] 的部分基准测试程序作为处理器性能的衡量标准. 其中 DSPStone 的测试程序包括: fft, fir2dim, matrix, convolution, lms, n_real_updates 和 n_complex_updates. 实验所用的测试程序均由 GCC 编译器在 O2 选项下编译. 对这些测试程序的核心代码部分进行超长指令字方式的优化,得到的结果如表 3 所示.

表 3 DSPStone 测试结果(周期数)

测试程序	超标量	混合模式	性能提升/%
fft	1 993	1 698	14. 8
fir2dim	530	494	6. 8
matrix	2 178	2 030	6. 8
convolution	162	131	19. 1
lms	439	284	35. 3
n_real_updates	75	62	17. 3
n _ complex _ up- dates	658	450	31. 6

分析以上数据可知,利用超长指令字与超标量融合的这种方式可以有效地提高指令的并行性,缩短指令执行的周期数,提升处理器的性能. 这种方法结合了超标量和超长指令字的优点. 既能动态调度指令,检测指令的相关性. 又能在不增加硬件开销的前提下,利用软件层面的静态调度消除指令的相关性,实现最大限度的指令级并行.

6 结束语

本文介绍了一种基于 ARM 指令集的通用 DSP 中指令相关的处理方法. 针对传统检测方法的不足,提出了一种改进的相关性检测方法. 不仅能够检测同一个指令包里面的指令是否存在数据相关,也能检测不同流水级相邻指令包之间的指令是否存在数据相关. 利用超长指令字和超标量两种模式融合的

方法,能够消除核心代码部分的指令相关,提高指令的并行度,而且不会增加硬件的开销. 本文提出的指令相关处理方法能够提高处理器的性能.

参考文献:

[1] Steven G, Christianson B, Collins R, et al. A superscalar architecture to exploit instruction level parallelism [J]. Microprocessors & Microsystems, 1997, 20(7): 391-400.

[2] 陈峰扬, 杜勇, 郭德源, 等. 一种 VLIW-Superscalar 混合微处理器结构[J]. 微电子学与计算机, 2013, 30(11): 1-5.

[3] Bruckner H P, Wielage W, Blume H. Intuitive and interactive movement sonification on a heterogeneous RISC/DSP platform[C]// The 18th Annual International Conference on Auditory Display. Atlanta, 2012: 75-82.

[4] Kim D H. Addressing mode extension to the ARM/thumb architecture[J]. Advances in Electrical & Computer Engineering, 2014, 14(2): 85-88.

[5] Wolfe M, Banerjee U. Data dependence and its application to parallel processing[J]. International Journal of Parallel Programming, 1987, 16(2): 137-178.

[6] Palermo G, Sam M, Silvan C, et al. Branch prediction techniques for low-power VLIW processors [C] // ACM Great Lakes Symposium on Vlsi 2003. Washington, Dc, USA, 2003: 224-228.

[7] Butko A, Garibotti R, Ost L, et al. Accuracy evaluation of GEM5 simulator system [C] // Reconfigurable Communication-centric System-on-Chip (ReCoSoC), 2012 7th International Workshop on. UK: York, IEEE, 2012: 1-7.

[8] Binkert N, Beckmann B, Black G, et al. The gem5 simulator [J]. ACM Sigarch Computer Architecture News, 2011, 39(2): 1-7.

[9] Agarwal N, Dimopoulos N. DSPstone benchmark of CoDels automated clock gating platform[C] //IEEE Computer Society Symposium on VLSI. IEEE Computer Society. France, Montpellier, 2007: 508-509.

作者简介:

王 旭 男, (1990-), 硕士. 研究方向为通用数字信号处理器设计. E-mail: xu-wang13@mails. tsinghua. edu. cn.

付家为 男, (1989-), 硕士. 研究方向为通用数字信号处理器设计.

何 虎 男, (1974-), 博士, 副教授. 研究方向为高性能数字信号处理器系统设计.