第47卷 第4期 计 算 机 工 程 2021年4月
Vol.47 No.4 Computer Engineering April 2021

·体系结构与软件技术· 文章编号：1000-3428(2021)04-0180-07 文献标志码：A 中图分类号：TP338

超标量处理器乱序提交机制的研究与设计

李 昭[1]，刘有耀[1]，焦继业[2]，潘树朋[1]

（1.西安邮电大学 电子工程学院，西安 710100；2.西安邮电大学 计算机学院，西安 710100）

摘 要：针对超标量处理器中长周期执行指令延迟退休及持续译码导致的重排序缓存(ROB)阻塞问题，提出一种指令乱序提交机制。通过设计容量可配置的多缓存指令提交结构，实现存储器操作指令和ALU类型指令的分类退休，根据超标量处理器架构及性能需求对目标缓存和存储缓存容量进行参数化配置降低流水线阻塞风险，同时利用指令目的寄存器编码提交模式加快指令提交速率。实验结果表明，该机制提高了单次指令提交数量，基于该机制的超标量处理器相比传统基于ROB顺序提交机制的超标量处理器在减少硬件开销的情况下平均IPC指数提升46%，相比基于值预测、乱序退休和组提交的超标量处理器平均IPC指数增益为19%，综合性能更优。

关键词：超标量处理器；重排序缓存；指令分类退休；乱序提交；目的寄存器编码

开放科学（资源服务）标志码（OSID）：

中文引用格式：李昭，刘有耀，焦继业，等.超标量处理器乱序提交机制的研究与设计[J].计算机工程，2021，47(4)：180-186.

英文引用格式：LI Zhao，LIU Youyao，JIAO Jiye，et al.Research and design of out-of-order submission mechanism for superscalar processor[J].Computer Engineering，2021，47(4)：180-186.

Research and Design of Out-of-Order Submission Mechanism for Superscalar Processor

LI Zhao[1]，LIU Youyao[1]，JIAO Jiye[2]，PAN Shupeng[1]

（1.School of Electronic Engineering，Xi'an University of Posts & Telecommunications，Xi'an 710100，China；
2.School of Computer Science & Technology，Xi'an University of Posts & Telecommunications，Xi'an 710100，China）

【Abstract】To address blocking of Reorder Buffer（ROB）caused by delayed retirement of long-term execution instructions and continuous decoding in the superscalar processors，this paper proposes a mechanism for out-of-order submission of instructions.This mechanism designs a multi-buffer instruction submission structure with configurable capacity to implement classified retirement of memory operation instructions and ALU instructions.Based on the structure and performance requirements of superscalar processors，parameterized configuration is performed on the Target Buffer（TB）capacity and Memory Buffer（MB）capacity to reduce the risk of streamline blocking.In addition，the encoding submission mode of instruction destination register is used to accelerate instruction submission.Experimental results show that the proposed mechanism increases the number of single instruction submissions.The superscalar processor based on the proposed mechanism improves the average IPC index by 46% compared with the traditional ROB-based superscalar processor while the hardware overhead is reduced，and by 19% compared with the superscalar processors based on ratio prediction，out-of-order retirement，and group submission schemes.It has better comprehensive performance.

【Key words】superscalar processor；Reorder Buffer（ROB）；instruction classification retirement；out-of-order submission；destination register encoding

DOI：10.19678/j.issn.1000-3428.0057410

0 概述

传统超标量处理器通过指令级并行（Instruction Level Parallelism，ILP）提升处理器执行性能，为实现推测性执行、精确异常和寄存器回收等功能，引入重排序缓存（Reorder Buffer，ROB）机制[1]。将解码之后的指令按照程序顺序写入ROB中，每条指令占用ROB的一个表项，在指令执行阶段一直保存在ROB

基金项目：国家自然科学基金（61874087，61834005，61634004）。

作者简介：李 昭（1993—），男，硕士研究生，主研方向为专用集成电路设计；刘有耀，教授、博士；焦继业，高级工程师、博士；潘树朋，硕士研究生。

收稿日期：2020-02-17 **修回日期：**2020-04-02 **E-mail：**15667235439@163.com

中，直到提交阶段从ROB中逐条退休，使程序按顺序更新处理器的状态，保证了在分支预测失败和指令异常时处理器状态的正确。随着集成电路技术的发展，众多领域对处理器性能的要求越来越高，ROB机制的不足也逐渐显现。例如，当ROB head被长周期执行指令占用时，后续指令就不能从ROB中退出，即使这些指令独立于长周期执行指令，并且它们已经计算完成，也不允许提交，使得ROB利用率较低。同时，由于ROB容量有限，指令持续解码最终会导致ROB完全填满，无论是否有其他独立的指令，都不能进入发射队列，并且指令执行停滞[2]，从而使处理器陷入相当多的周期停顿[3]，对处理器性能造成影响。ROB阻塞问题的产生是由于ROB遵循指令顺序退出策略影响处理器性能，这些长周期执行指令一方面在多个周期内不可用，可能会长时间阻塞从属指令的执行，另一方面退出缓慢，已执行完的指令会在多个周期内占用关键资源，一旦资源用尽，处理器就会停止获取新指令并最终停顿，导致ROB等其他硬件资源利用率降低。针对以上问题，需要从提高电路硬件资源利用率以及指令提交控制逻辑等方面进行优化，简单的解决方案是扩大ROB容量以容纳更多的并行指令。随着基于ROB的微体系结构在指令提交阶段对某些关键资源的分布进行例化，会增加处理器面积和功耗方面的成本，还可能会影响处理器周期[4]。

目前，研究人员针对ROB阻塞问题提出了一些解决方案。文献[5]提出退休长周期指令的值预测机制，该机制在长周期执行指令阻塞ROB时，通过赋假值使其提前退出，其他指令正常执行，但不允许覆盖内存中的数据。在等待长周期执行指令操作完成后，处理器系统返回到检查点并恢复常规执行。该机制加快了整体执行速度，但是没有真正解决ROB的阻塞问题。文献[6]提出指令乱序退休机制来解决ROB阻塞问题，该机制将程序中所有可能产生异常和预测错误的指令设置为节点指令，当验证缓存（Valid Buffer，VB）检验某节点指令不会发生异常和预测错误时，其后面到下一个节点之间的指令会乱序退休。通过指令乱序退休缩小了长周期执行指令占用ROB的时间，但同时为保证乱序退休时ROB的正确维护，需要复杂的控制逻辑和大量的硬件资源进行支持。文献[7]提出组提交机制，该机制将程序按顺序分组，每组内包含若干寄存器相关的指令时，仅将使用同一寄存器的最后更新寄存器文件的指令输入ROB条目中。此外，在提交该组指令之前，可以更早地释放该组中最后更新寄存器文件的指令对应的寄存器。该机制增加了ROB的有效容量和物理寄存器的有效数量，但是有限的ROB资源不能满足管理组信息的高存储需求，并且由于指令组提交数量大，当预测错误发生时会冲刷更多的指令，造成严重的周期停滞。

为满足超标量处理器的高性能和小面积设计需求，本文在研究传统ROB机制以及多种主流ROB优化机制的基础上，提出一种指令快速提交机制。该机制通过容量可配置的多缓存指令提交结构，实现存储器操作指令和ALU类型指令的分类退休，并利用指令目的寄存器编码提交模式完成指令结果的乱序提交。

1　传统ROB机制性能研究

传统超标量处理器ROB的每个表项中都包含判断指令执行完毕的状态位Complete、指令异常类型判断标志位Exception、判断指令类型并决定其提交位置的状态位Type、寄存器重命名的映射关系等资源。这些资源导致了重排序缓存内部结构复杂，并且当指令产生异常时，必须在ROB内部等待异常处理完毕后才能继续后续指令的退出，这样会给处理器带来相当多的周期停滞，严重影响处理器性能。ROB容量越大，容纳的并行指令越多，造成阻塞的风险越小，对处理器性能提升有一定帮助。目前，提出的一些解决方案通过扩大主要的微处理器结构或对其进行高效管理来减轻ROB阻塞引起的性能下降问题[8]。由文献[2]可知此类解决方案的效果并不理想，因为当ROB尺寸持续增加到某一定值时，首先会对处理器性能提升产生免疫，其次增大ROB尺寸就必须增加其内部例化的各种资源，会增加处理器面积和功耗方面的成本，也可能影响处理器周期，造成处理器性能和面积的失衡。

通过对项目组在研的一款基于ROB机制的双发射超标量处理器的性能和面积进行评估，在功能仿真的硬件设计下载入现场可编程门阵列（Field Programmable Gate Array，FPGA）进行验证，运行Dhrystone和CoreMark基准测试程序对处理器性能进行评估。通过记录程序中指令执行总时长和执行总数得到平均IPC指数，以此作为处理器性能评判的指标。基于ROB机制的超标量处理器内核在0.11 μm CMOS工艺下使用Design Compiler完成逻辑综合，具体参数设置如表1所示。

表1　基于ROB机制的超标量处理器参数设置

Table 1　Parameters setting of superscalar processor based on ROB mechanism

参数名	参数值
Dhrystone平均IPC指数	0.73
CoreMark平均IPC指数	0.72
CMOS工艺/μm	0.110
ROB尺寸/10^4	3.39
内核尺寸/10^4	7.65

实验中所研究的超标量处理器架构最多可同时乱序发射并执行2条指令、顺序提交1条指令，其平均IPC

指数为0.73，与理论值相差较大[9]。由文献[10]结论可知，平均IPC指数达不到理想值是由于流水线停顿，其中ROB堵塞是造成流水线停顿的主要原因。通过实验对不同类型指令造成ROB阻塞的比例进行对比分析，得出存储器操作指令的比例最高。影响ROB阻塞的主要因素为：1）ROB尺寸难以匹配指令持续解码的速度，ROB变满而造成阻塞；2）由于指令顺序提交，因此当程序中长周期执行指令延迟退休时会阻塞ROB，导致ROB利用率较低。

为提高处理器的平均IPC指数，本文从上述两个影响因素入手，对ROB机制进行优化。首先对基于该处理器的RISC-V指令集中各指令功用及其指令编码格式进行分类研究，提出将存储器操作指令和其他指令分类退休、结果乱序提交的设计思路。针对影响因素1，该机制可以根据应用程序的需求动态调整资源分配[11]，将指令提交缓存模块容量根据处理器性能要求进行参数化配置。针对影响因素2，该机制中的指令目的寄存器编码提交模式可以增加单次指令提交数量，实现指令结果乱序提交。该机制一方面通过调节指令提交缓存模块容量来容纳更多的并行指令，另一方面通过指令乱序提交来加快指令提交速率，为后续等待提交的指令预留出足够的空间，从而有效降低ROB阻塞的风险，充分利用指令提交缓存模块的资源，提升超标量处理器的执行性能。

2 指令乱序提交机制

基于指令类型缓存（Type Buffer，TB）、目标缓存（Object Buffer，OB）和存储缓存（Memory Buffer，MB）的超标量流水线结构可实现指令乱序提交。指令目的寄存器编码提交模式支持指令结果乱序提交，相比传统超标量处理器中ROB顺序提交模式，将所有指令运行过程中的信息存储在ROB中进行退休管理，并考虑了性能和成本之间的折衷[12]。本文在此基础上，通过将ROB中的数据或地址字段保留在单独的结构中来减小ROB条目的宽度，而不会影响ROB接口[13]，一定程度上降低了长周期执行指令阻塞后续指令提交的风险，提高了单次指令提交数量。乱序提交的流水线结构如图1所示。

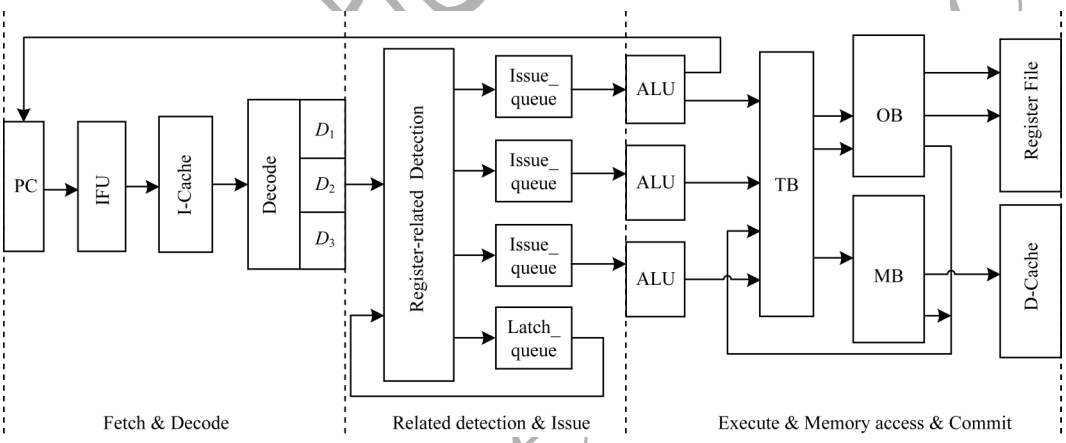

图1 乱序提交的流水线结构

Fig.1 Pipeline structure of out-of-order submission

乱序提交的流水线结构满足指令分类退休和结果乱序提交，流水线分为取指令（Fetch）和解码（Decode）、相关检测（Related detection）和发射（Issue）以及执行（Execute）、访存（Memory access）和提交（Commit）3级。该流水线结构的基本工作流程为：首先取指单元（Instruction Fetch Unit，IFU）在一个周期内从指令缓存（I-Cache）中取出3条指令，经过解码之后对这3条指令的目的寄存器依次编码为D_1、D_2、D_3，然后送入寄存器相关检测（Register-related Detection）模块中进行乱序发射评估。将满足发射条件的指令送入发射队列（Issue_queue）中，等待发射队列中指令的操作数准备就绪并且功能单元空闲时将指令发送给相应的ALU进行计算，而将与之前送入发列队列的指令存在写后读（Read After Write，RAW）相关的后续指令暂存在锁存队列（Latch_queue）中，等待与下一轮循环进入流水线的3条新指令共同进行寄存器相关检测。以此类推，可以为发射队列提供持续的并行指令源，实现指令的动态调度。ALU计算完毕后经过指令类型缓存TB的判断，将Load/Store指令的目的寄存器编码号（D_i，i=1，2，…，n）和读写地址、写数据送入存储缓存中，存储缓存再通过地址数据总线访问数据缓存（D-Cache），完成数据存取操作。其余指令的计算结果和目的寄存器编码号送入目标缓存中，等待指令目的寄存器编码号等于0时将结果提交至寄存器文件（Register File，RF）中完成一次提交动作。本文提出的新型超标量处理器流水线结构可实现同时3条指令乱序发射，乱序执行，2条指令乱序提交，极大地提升了指令提交速率，同时使用容量可配置的指令提交缓存结构TB、OB和MB代替复杂的ROB机制，可有效提高电路硬件资源的利用率。

2.1　指令分类退休机制

与开放式指令集体系结构(Instruction Set Architecture,ISA)不同,RISC-V设计简单、灵活和可扩展,这有助于将其移植到不同的技术和应用领域[14]。本文基于RISC-V指令集规整的指令编码格式,借助指令类型缓存、目标缓存和存储缓存的多缓存结构实现Load/Store指令和其余指令分类退休。指令类型缓存主要负责对ALU计算完成的指令进行类型判断后发送给相应的提交缓存模块,并且当指令提交缓存模块OB和MB没有剩余空间时负责暂存待提交指令的信息,等待目标缓存和存储缓存腾出足够的空间时再将指令计算结果和目的寄存编码送入其内部等待提交,防止因指令提交缓存变满而造成的流水线停顿,提供缓冲作用,使得流水线各级实现速度匹配。指令类型缓存还可以为关联寻址的指令提供源操作数,加快了指令执行速度,但是操作数读取端口的增加会导致处理器面积的增加。MB负责缓存Load/Store指令的信息,OB负责缓存其余指令的信息。本文根据微处理器性能及架构要求对目标缓存和存储缓存容量进行调整,保证流水线各级的速度匹配。基于TB+OB+MB的指令退休结构如图2所示。

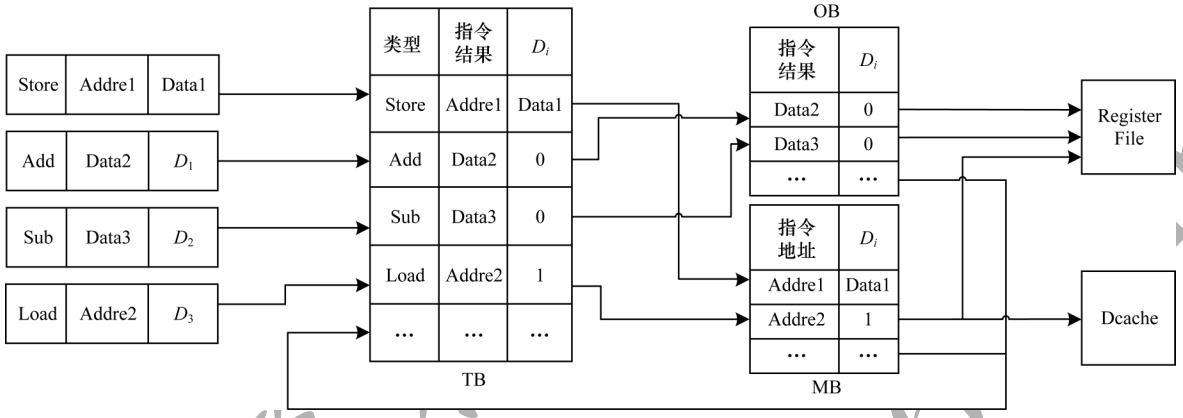

图2　基于TB+OB+MB的指令退休结构

Fig.2　Instruction retirement structure based on TB+OB+MB

基于TB+OB+MB的指令退休结构可有效解决当Store指令占用ROB head时而导致的流水线停顿问题,采用TB+OB+MB结构对不同功用指令进行分类退休,并与流水线其他各级模块相互协作。首先由指令类型缓存模块接收来自ALU模块输出的指令计算结果(Data &Addre)和目的寄存器编码号,然后将指令分类后的指令信息分别送入目标缓存和存储缓存中进行暂存,等待第1条指令提交后,目标缓存将该轮循环中所有指令的目的寄存器编码号减1,依此类推,直到指令目的寄存器编码号等于0的指令将其结果提交至寄存器文件中。如果检测到Load指令,则由TB负责将指令的读地址和目的寄存器编码号送入存储缓存中进行暂存,等待其目的寄存器编码减为0且完成数据读取操作后,再通过存储缓存模块将读取结果提交至寄存器文件中,完成Load指令提交动作。该结构可以实现单周期执行指令优先于Store长周期执行指令提交,即使Store指令发生了异常,也无需冲刷已经提交的单周期执行指令,因为在指令分类退休机制的控制下,与Store指令存在读后写(Write After Read,WAR)相关的指令在本轮循环中不满足提交的条件,所以该机制满足乱序提交下的精确异常,降低指令异常时冲刷流水线的风险。同时,随着循环轮数的增加,指令译码仍在继续,此时将导致目标缓存和存储缓存剩余空间不足,从而阻塞流水线,对处理器性能造成影响。

本文通过修改目标缓存和存储缓存容量来解决此问题,但是增大提交缓存模块的同时会影响处理器的面积和功耗,因此本文在每个程序中合理利用OB和MB,并且根据程序中不同类型的指令数量来确定OB和MB尺寸[15]。结合实际性能测试结果来合理分配OB+MB尺寸,目的是为了通过调节模块容量来更好地反映应用程序的需求,从而获得更合理的资源配置方案以及更好的能耗比,使处理器达到性能和面积的平衡。通过实验分析可知,针对不同的应用程序调整指令退休结构的硬件参数可获取最佳处理器性能,使得基于该结构的超标量处理器更适用于高性能和低功耗场景。

2.2　结果乱序提交机制

目前,多数超标量处理器主要通过指令乱序发射和并行执行、结果顺序提交来保证处理器性能,但当长周期执行指令占用ROB head时,顺序提交的缺陷就异常凸显。针对以上问题,本文提出的结果乱序提交机制利用目的寄存器编码提交模式在保证指令正确提交的前提下,最多可同时完成2条指令的乱序提交,相比传统ROB机制,该机制从单次指令提交数量和指令乱序提交两方面加快了指令提交速率,为指令提交缓存模块预留出足够的空间来容纳更多的并行指令,一定程度上提升了指令提交模块的利用率。指令目的寄存器编码提交结构如图3所示。

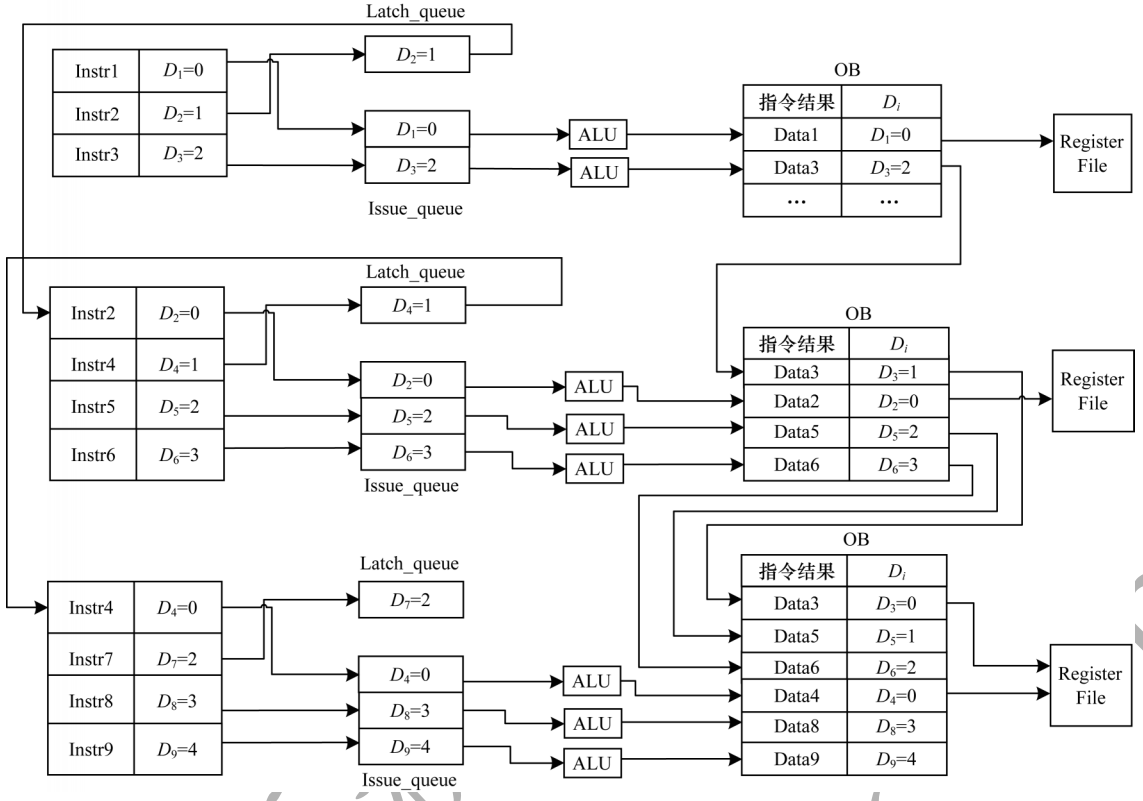

图3 指令目的寄存器编码提交结构

Fig.3 Coding submission structure of instruction destination register

指令目的寄存器编码提交模式的控制流程为首先在指令译码阶段对输出的3条指令中存在目的寄存器的指令编码为D_i=0,1,2,然后送入寄存器相关检测模块中进行指令相关性判断,将这3条指令中与之前指令存在RAW相关的后续指令送入锁存队列中进行暂存,其余指令由发射队列送入ALU中进行计算,等待指令计算完成,将计算结果和目的寄存器编码号送入目标缓存中等待提交。允许指令乱序提交的必要条件为指令计算完成和指令D_i=0[16]。满足这两个条件的指令可以将其计算结果提交至寄存器文件中,并且由目标缓存控制该轮循环中所有指令的目的寄存器编码号减1。该目的寄存器编码提交模式为不断循环迭代,在第2轮循环中,将锁存队列中指令的目的寄存器编码号减1后作为首条参与寄存器相关检测的指令,此时指令译码模块输出的3条新指令的目的寄存器编码规则必须按照程序顺序进行编码,并且第1条指令目的寄存器编码号必须等于上一轮循环中所有指令目的寄存器编码号的最大值减1。依此类推,这样就可以保证指令目的寄存器编码提交模式的循环迭代,从而通过控制指令提交顺序和数量来提升处理器性能。该指令目的寄存器编码提交模式可满足当长周期执行指令Store等待提交时,最多同时乱序提交2条与Store不相关的指令,因满足提交的2条指令不在同一轮循环中,它们之间可能存在RAW、WAR和写后写(Write After Write,WAW)相关问题[17]。为避免该问题带来的错误提交,本文通过指令目的寄存器编码提交模式中的循环轮数优先级可解决读后写相关问题,并且可在控制指令发射阶段避免写后读相关问题。当一轮循环中已发射的指令与其编号之后的锁存队列中的指令存在写后读相关时,在下一轮循环锁存队列中的指令可以从指令类型缓存中快速读取源操作数,TB的一个输出端口与读取端口复用,可在加快指令执行速度的同时减少读取端口的数量[18]。

本文中的结果乱序提交机制满足同时乱序提交2条指令,加快了指令提交速率,同时其控制逻辑相对简单,相比传统ROB顺序提交机制,极大地提升了程序运行速度。但当锁存队列中的指令与Store指令存在RAW相关时,由于Store指令执行时间不确定,可能先于锁存队列中的指令完成提交,这样会造成错误提交,需要冲刷流水线使处理器状态恢复正常。本文提出一种指令发射控制策略,当指令锁存队列变满时,先不允许发射检测出的Store指令,等待锁存队列中的指令执行完毕后再发射Store指令。这样虽然避免了提交错误而导致的流水线冲刷问题,但同时又带来了新的问题,使得在该发射控制策略下遇到上述情况时最差只能发射1条指令,此时不仅不能充分利用电路的硬件资源,影

响超标量处理器性能，而且还会产生不必要的功耗。可见，结果乱序提交机制还需综合考虑微处理器架构以及性能需求做进一步优化。

3　实验与结果分析

本节主要对基于指令乱序提交机制（简称本文机制）的超标量处理器性能和面积进行评估，并与基于ROB顺序提交机制、值预测机制[5]、乱序退休机制[6]和组提交机制[7]的超标量处理器进行性能对比。首先对整体设计进行功能仿真和逻辑综合后将生成的sof文件载入DE2_70 FPGA开发板上，通过运行Dhrystone、CoreMark和SPEC 2006基准测试程序获取改进后处理器的平均IPC指数。为估计芯片面积开销[19]，在0.11 μm CMOS工艺下对优化后的整体电路进行逻辑综合，得到处理器内核尺寸以及指令提交缓存模块尺寸。为进一步探究指令提交缓存模块尺寸与处理器性能的关系，通过修改OB和MB尺寸并对修改后的处理器按上述过程进行性能和面积评估。不同OB+MB尺寸下Dhrystone和CoreMark的平均IPC指数，如图4所示。

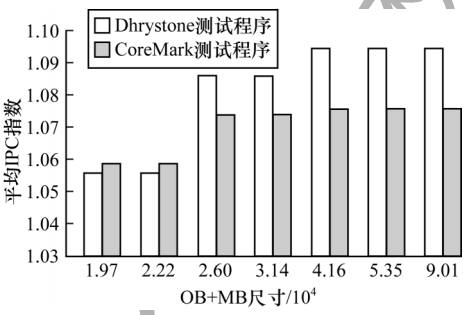

图4　不同OB+MB尺寸下Dhrystone和CoreMark的平均IPC指数

Fig.4　Average IPC index of Dhrystone and CoreMark at different OB+MB size

实验结果表明，当指令提交缓存模块OB+MB尺寸从1.97×10^4增长到2.60×10^4时，处理器的平均IPC指数逐渐增长并趋于稳定，当OB+MB尺寸从2.60×10^4增长到4.16×10^4时，平均IPC指数增幅并不明显，并且随着OB+MB的不断增大，平均IPC指数保持稳定不变，证明此时单纯增加指令提交缓存模块不能有效提升处理器性能，需要结合流水线其他各级模块进行综合考虑。为实现处理器性能和面积的平衡，根据上述结果可知，当OB+MB尺寸为2.60×10^4时的处理器内核尺寸为4.90×10^4，对应的平均IPC指标为1.078是最佳配置。本文通过运行SPEC CPU2006基准测试套件中的12个SPECint应用程序[20]来评估处理器性能，当OB+MB尺寸为2.60×10^4时，基于本文机制的超标量处理器平均IPC指数如图5所示。

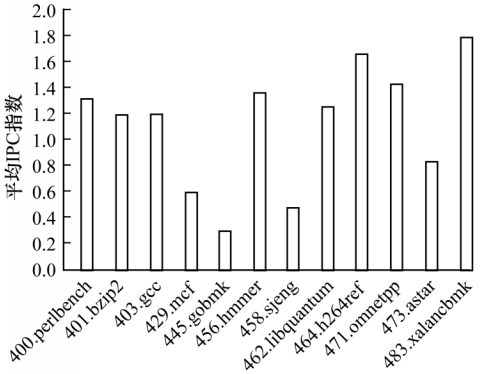

图5　当OB+MB尺寸为2.60×10^4时基于本文机制的超标量处理器平均IPC指数

Fig.5　Average IPC index of superscalar processor based on the proposed mechanism when the OB + MB size is 2.60×10^4

实验结果表明，当本文机制中的指令提交缓存模块OB+MB尺寸为2.60×10^4时，通过每个周期提交的IPC表示处理器在一个周期内可以执行的指令个数。在各个应用程序中差异较大[21]，基于本文机制的超标量处理器的平均IPC指数为1.11，相比基于ROB顺序提交机制的超标量处理器在使用较少的硬件资源情况下处理器性能提升了46%。将基于本文机制、值预测机制[5]、乱序退休机制[6]和组提交机制[7]的超标量处理器进行性能对比，如表2所示。可以看出，基于本文机制的超标量处理器相比基于值预测、乱序退休和组提交机制的超标量处理器平均IPC指数增益为19%，实现了更高的性能提升。

表2　基于4种机制的超标量处理器性能对比

Table 2　Performance comparison of superscalar processors based on four mechanisms

性能指标	值预测机制	乱序退休机制	组提交机制	本文机制
提交策略	顺序	乱序	顺序	乱序
发射数量	4	4	4	3
平均IPC指数	1.09	0.77	0.92	1.11

由实验分析可知，本文机制可通过调整OB和MB尺寸来提升处理器性能，但是单纯增加OB+MB尺寸并不能明显提升处理器性能，甚至随着OB+MB尺寸的不断增大，处理器性能将不再变化。出现该情况的主要原因为本文机制未充分考虑指令提交缓存模块与流水线其他各级模块的关系，只有实现各级模块速度的匹配，才能最大化处理器性能。

4　结束语

传统基于ROB顺序提交机制的超标量处理器由于长周期执行指令占用ROB head时间过长导致流水线阻塞，以及指令持续解码使得ROB无法容纳更多并

行指令，从而造成ROB资源利用率降低。为解决上述问题，本文提出一种指令乱序提交机制，利用指令目的寄存器编码提交模式实现在Store指令阻塞提交时，后续执行完毕的单周期执行指令优先于Store指令完成提交，保证单次最多同时提交2条指令，提高指令提交速率。实验结果表明，基于指令乱序提交机制的超标量处理器相比传统基于ROB顺序提交机制的超标量处理器在减少硬件开销的情况下性能提升46%，相比基于值预测、乱序退休和组提交机制的超标量处理器性能平均提升19%。后续可将该指令乱序提交机制应用于嵌入式处理器中，实现高性能和低功耗的处理器设计。

参考文献

[1] SMITH J E, PLESZKUN A R. Implementing precise interrupts in pipelined processors[J]. IEEE Transactions on Computers, 1988, 37(5): 562-573.

[2] CRISTAL A. Kilo-instruction processors: overcoming the memory wall[J]. IEEE Micro, 2005, 25(3): 48-57.

[3] KARKHANIS T, SMITH J E, BOSE P. Saving energy with just in time instruction delivery [C]//Proceedings of International Symposium on Low Power Electronics and Design. Washington D. C., USA: IEEE Press, 2002: 178-183.

[4] PALACHARLA S, JOUPPI N P, SMITH J E. Complexity-effective superscalar processors[C]//Proceedings of the 24th Annual International Symposium on Computer Architecture. Washington D. C., USA: IEEE Press, 1997: 206-218.

[5] MUTLU O, STARK J, WILKERSON C, et al. Runahead execution: an alternative to very large instruction windows for out-of-order processors[C]//Proceedings of the 9th International Symposium on High-Performance Computer Architecture. Washington D. C., USA: IEEE Press, 2003: 129-140.

[6] PETIT S, UBAL R, SAHUQUILLO J, et al. An efficient low-complexity alternative to the ROB for out-of-order retirement of instructions[C]//Proceedings of the 12th Euromicro Conference on Digital System Design, Architectures, Methods and Tools. Washington D. C., USA: IEEE Press, 2009: 635-642.

[7] AFRAM F, ZENG H, GHOSE K. A group-commit mechanism for ROB based processors implementing the X86 ISA [C]//Proceedings of the 19th International Symposium on High Performance Computer Architecture. Washington D. C., USA: IEEE Press, 2013: 47-58.

[8] BALASUBRAMONIAN R, DWARKADAS S. Reducing the complexity of the register file in dynamic superscalar processors[C]//Proceedings of the 34th ACM/IEEE International Symposium on Microarchitecture. Washington D. C., USA: IEEE Press, 2001: 237-248.

[9] SUN Caixia, SUI Bingcai, WANG Lei, et al. Counters based performance analysis and optimization of an out-of-order superscalar processor core[J]. Journal of National University of Defense Technology, 2016, 38(5): 14-19. (in Chinese)
孙彩霞，隋兵才，王蕾，等. 乱序超标量处理器核的性能分析与优化[J]. 国防科技大学学报，2016，38(5)：14-19.

[10] MARTI S P, BORRAS J S, RODRIGUZE P L, et al. A complexity-effective out-of-order retirement microarchitecture[J]. IEEE Transactions on Computers, 2009, 58(12): 1626-1639.

[11] PONOMAREV D, KUCUK G, GHOSE K. Dynamic resizing of superscalar datapath components for energy efficiency[J]. IEEE Transactions on Computers, 2006, 55(2): 199-213.

[12] LI Cunlu, DONG Dezun, LU Zhonghai, et al. ROB-Router: a reorder buffer enabled low latency network-on-chip router [J]. IEEE Transactions on Parallel and Distributed Systems, 2018, 29(9): 2090-2104.

[13] ZHANG S Z, WRIGHT A, BOURGEAT T, et al. Composable building blocks to open up processor design[C]//Proceedings of the 51st Annual IEEE/ACM International Symposium on Microarchitecture. Washington D. C., USA: IEEE Press, 2018: 68-81.

[14] SARTORI M L L, CALAZANS N L V. Go functional model for a RISC-V asynchronous organization—ARV [C]//Proceedings of the 24th IEEE International Conference on Electronics, Circuits and Systems. Washington D. C., USA: IEEE Press, 2017: 381-348.

[15] LI Cunlu, DONG Dezun, LIAO Xingke, et al. ROB-Router: low latency network-on-chip router microarchitecture using reorder buffer[C]//Proceedings of the 24th Annual Symposium on High-Performance Interconnects. Washington D. C., USA: IEEE Press, 2016: 68-75.

[16] BELL G B, LIPASTI M H. Deconstructing commit[C]//Proceedings of IEEE International Symposium on Performance Analysis of Systems and Software. Washington D. C., USA: IEEE Press, 2004: 68-77.

[17] LEE K, JEONG I, RO W W. Parallel in-order execution architecture for low-power processor[C]//Proceedings of International SoC Design Conference. Washington D. C., USA: IEEE Press, 2017: 65-66.

[18] KUCUK G, PONOMAREV D V, ERGIN O, et al. Complexity-effective reorder buffer designs for superscalar processors[J]. IEEE Transactions on Computers, 2004, 53(6): 653-665.

[19] XI S L, JACOBSON H, BOSE P, et al. Quantifying sources of error in McPAT and potential impacts on architectural studies[C]//Proceedings of the 21st International Symposium on High Performance Computer Architecture. Washington D. C., USA: IEEE Press, 2015: 577-589.

[20] JEONG I, LEE C, KIM K, et al. OverCome: coarse-grained instruction commit with handover register renaming[J]. IEEE Transactions on Computers, 2019, 68(12): 1802-1816.

[21] DAVID W W. Limits of instruction-level parallelism[C]//Proceedings of the 4th International Conference on Architectural Support for Programming Languages and Operating Systems. New York, USA: ACM Press, 1991: 176-188.

编辑 陆燕菲