文章编号：2096-1472(2018)-02-14-03
DOI:10.19644/j.cnki.issn2096-1472.2018.02.005

基于Open64上的特殊指令合成策略研究与实现

陈金娥[1]，黄胜兵[2]
(1.安徽医学高等专科学校，安徽 合肥 230601；
2.中国科学技术大学计算机科学与技术学院，安徽 合肥 230026)

摘　要：Open64是一个拥有GNU通用公共许可证的开源高性能编译器，设计结构好，分析优化全面，是编译器高级研究的理想平台。本文针对BWDSP处理器所提供的高效特殊运算指令，在Open64基础上研究并实现了面向BWDSP中的特殊指令合成策略。该策略通过扩展并重定向编译器后端模块，能够充分地利用BWDSP中的复数指令、累加指令、乘累加指令和平方和指令等特殊指令。实验结果表明，本文提出的特殊指令合成策略能够很大程度上提高程序的性能。

关键词：Open64编译器；特殊指令合成；超长指令字
中图分类号：TP311.54　　**文献标识码**：A

Research and Implementation of Special Instruction Synthesis Strategy Based on Open64

CHEN Jine[1],HUANG Shengbing[2]
(1.*Basic department of Anhui Medical College,Hefei* 230601,*China*;
2.*School of Computer Science and Technology,University of Science and Technology of China,Hefei* 230026,*China*)

Abstract: As an open-source high-performance compiler with GNU General Public License (GPL),Open64 is a good compiler for advanced research.Aiming at the special instructions provided by BWDSP,the paper researches and implements the special instruction synthesis strategy based on Open64.Extending and redirecting the compiler back-end,the strategy fully utilizes the special instructions of BWDSP,including complex,accumulation,multiply accumulation and sum-of-squares operations.The experimental results show that the program performance can be greatly improved with this special instruction synthesis strategy proposed in this paper.

Keywords: Open64 compiler;special instruction synthesis;Very Long Instruction Word (VLIW)

1 引言(Introduction)

BWDSP是一款采用分簇架构，支持SIMD的16发射的VLIW数字信号处理器。根据其面向的计算领域提供了大量的特殊指令[1,2]，包括复数指令、累加指令、乘累加指令和平方和指令等。

本文以Open64作为面向BWDSP体系结构的编译器研究框架。Open64是一个运行在Linux下的C、C++、Fortran编译器基础设施[3]，其前端将源程序转化为中间表示WHIRL，后端读入中间语言WHIRL，经过翻译生成CG阶段(Code Generation)的中间表示(CGIR)，再经过一系列优化，最终CGIR经过代码输出生成汇编程序[4]。Open64编译器的架构如图1所示。

本文主要解决如何将高级程序语言代码通过编译器直接生成BWDSP指令系统中的特殊指令，并在Open64编译基础设施上提出并实现了面向BWDSP体系结构的特殊指令合成策略，能够较好地利用BWDSP的特殊指令来提高应用程序的性能。

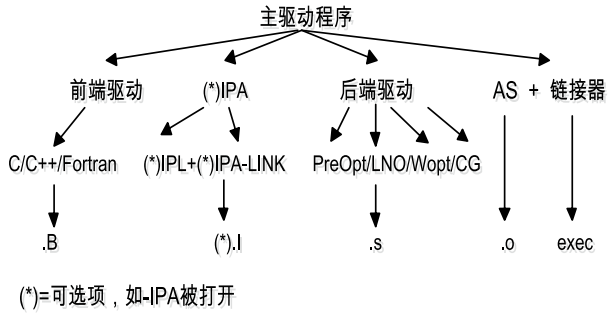

图1 Open64体系架构

Fig.1 The structure of Open64

2 特殊指令合成策略(Special instruction synthesis strategy)

2.1 复数指令合成

原Open64编译基础设施内部提供了浮点复数类型，因

此编译器前端能够直接处理浮点复数类型源代码。然而编译器后端为了能够生成处理复数形式的汇编代码，在中间语言WHIRL的Middle WHIRL上又将复数类型C4下降为Float类型来进行处理。为了充分利用BWDSP处理器的特点，必须将复数类型到浮点类型的转换过程进行屏蔽，使其直接处理复数类型的中间语言WHIRL，指令如下所示(以复数加法为例)：

```
C4C4LDID 0 <2,1,a> T<17,.predef_C4,4>
  C4C4LDID 0 <2,2,b> T<17,.predef_C4,4>
 C4ADD
 C4STID 0 <2,3,c> T<17,.predef_C4,4> {line：1/6}
```

这种带复数类型C4的中间语言WHIRL在后端的代码生成阶段(CG_Generate_Code)直接将其注释为BWDSP体系结构中的复数指令。

2.2 累加和乘累加类指令合成

原Open64编译器基础设施中累加和乘累加类指令通常是由数条汇编指令组成，如下所示(以c+=a*b为例)：

```
I4I4LDID 0 <2,1,a> T<4,.predef_I4,1>
  I4I4LDID 0 <2,2,b> T<4,.predef_I4,1>
 I4MPY
 I4I4LDID 0 <2,3,c> T<4,.predef_I4,1>
I4ADD
I4STID 0 <2,3,c> T<4,.predef_I4,1> {line：1/6}
```

面向BWDSP的Open64编译器在中间语言WHIRL上生成的累加操作直接进行指令注释，并不需要在其中插入相应的特殊规约处理指令。如下所示：

```
I4I4LDID 0 <2,1,a> T<4,.predef_I4,1>
  I4I4LDID 0 <2,2,b> T<4,.predef_I4,1>
 I4REDUCE_ADD
I4STID 0 <2,3,c> T<4,.predef_I4,1>
```

该类指令与复数指令的处理类似，将生成带累加操作的中间语言WHIRL，在后端代码生成阶段直接将其注释为BWDSP体系结构中的累加和乘累加类指令。

2.3 平方和类指令合成

平方和类指令合成与上述的两类特殊指令合成策略有所不同。由于Open64编译器基础设施中并没有提供相应的平方和类指令表示，因此需要在中间语言WHIRL中增加相应的WHIRL操作。面向BWDSP的Open64编译器中的平方和类特殊指令的合成策略框架如图2所示。

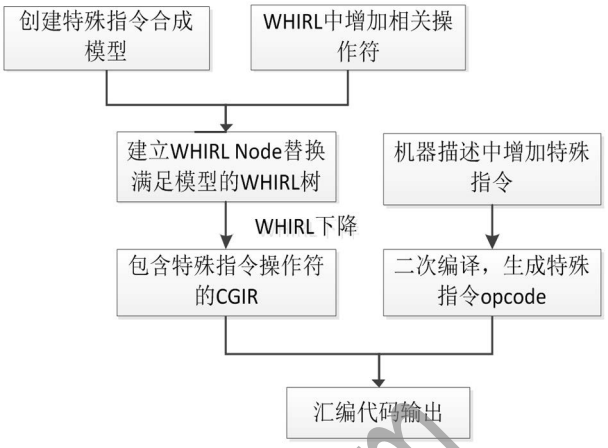

图2 特殊指令合成框架

Fig.2 The framework of special instruction synthesis

例如，平方和操作(例如c=a*a+b*b)在中间语言WHIRL上合成的中间表示如下所示：

```
I4I4LDID 0 <2,1,a> T<4,.predef_I4,1>
  I4I4LDID 0 <2,2,b> T<4,.predef_I4,1>
 I4SQURA
 I4STID 0 <2,3,c> T<4,.predef_I4,1> {line：1/6}
```

3 代码生成内部模块改进(Improvement of the internal modules in code generation)

面向BWDSP的Open64编译器后端的代码生成模块部分是与BWDSP体系结构最为密切相关的阶段，因此需要对代码生成内部的各个阶段进行相应的扩展和重定向，具体包括四个阶段。

3.1 机器描述

Open64中的机器描述文件称为Knobsfile[5]，可将其按照硬件资源信息分为指令集描述、运算资源描述、寄存器描述和汇编输出描述等四类信息。Open64的机器描述采用的是二次编译的方式设计的，编译器通过采用Intel提供的可解析特点格式的信息文件工具KAPI来生成BWDSP使用的数据和文件，即只需在v11-itanium-extra.knb和v12-itanium-extra.knb文件中添加相应特殊指令的机器描述，就可以生成相应动态链接库供编译器在使用过程中调用。

3.2 指令注释

Open64中的指令注释主要是将中间语言WHIRL转换为代码生成内部的中间表示CGIR，该过程位于CG_Expand中，特殊指令的注释主要分为如下几类。

(1)复数指令注释

原Open64框架中并没有提供复数寄存器对的概念，因此需要扩展虚拟寄存器的数据结构TN，在其中增加数据域TN_Pair用来表示复数寄存器对。指令注释过程中直接将带复数类

型C4的WHIRL节点注释成为复数指令。此外，对于复数乘法运算(如c=a*b)，BWDSP指令系统并没有提供完整的复数指令，因此需要在其中插入两条运算指令，如下所示：

```
Rm+1:m=[Um+=2,0] //复数a
Rn+1:n=[Un+=2,0] //复数b
qFRm+1:m_n+1:n=CFRm+1:m*CFRn+1:n
//插入运算指令
FRs+1=FRn+FRm
FRs=FRn+1-FRm+1
[Us+=2,0]=Rs+1:s //存储复数c
```

(2)累加和乘累加类指令注释

对于该类指令的注释，需要在虚拟寄存器数据结构TN中增加表示累加信息的数据域TN_Acc和乘累加信息的数据域TN_Macc,以便后面的寄存器分配处理。另外还需要在该类指令前面插入一条初始化指令。例如c+=a*b运算的指令注释如下所示：

```
Clr Macc //初始化清零
Rm=[Um+=1,0] //源操作数a
Rn=[Un+=1,0] //源操作数b
Macc0+=Rm*Rn //乘累加操作
Rs=Macc0
[Us+=1,0]=Rs //存储目的操作数c
```

(3)平方和类指令注释

平方和指令的注释与普通指令的注释完全一致，只需按照普通指令的注释方式进行指令注释即可。

3.3 寄存器分配

寄存器分配阶段是根据虚拟寄存器TN中的相关寄存器信息为每个虚拟寄存器TN分配相应的物理寄存器。Open64中的寄存器分配包括全局寄存器分配和局部寄存器分配[5]。传统的通用处理器中并没有提供A/B面寄存器，即复数寄存器对，故该处需要考虑寄存器对信息并进行相应的特殊处理。

面向BWDSP的Open64编译器沿用并扩展了原Open64中的寄存器分配算法，这里采用的策略是优先考虑A/B面寄存器等特殊情况。通过TN中的TN_Pair数据域，将TN寄存器对作为一个整体来处理，在寄存器分配时给其分配连续并且低位寄存器编号为偶数的两个物理寄存器，即复数寄存器对。

对于累加指令和乘累加指令的寄存器分配，则只需要根据虚拟寄存器TN中的数据域TN_Pair和TN_Macc分别分配相应的累加寄存器ACC和乘累加寄存器MACC即可。

3.4 汇编代码输出

面向BWDSP的Open64编译器中的汇编代码输出模块主要是根据BWDSP中的指令集特点，按照其相应的汇编格式输出相应的汇编代码程序。该过程位于cgemit.cxx文件中。对于复数指令的汇编代码，其中的两个寄存器对之间必须用”:”来表示，并且低位寄存器编号必须有偶数；对于累加指令则需要在累加寄存器的编号前面加上标识Acc；对于乘累加指令，则加上标识Macc等。

4 结论(Conclusion)

本文在Open64编译基础设施上设计并实现了上述的特殊指令合成策略，为了验证该方案的效果，我们选取用于测试DSP编译器性能的、在DSP数字信号处理领域具有典型应用的部分运算作为测试集，详见表1。

表1 基准测试集

Tab.1 Benchmark test set

测试基准	主要运算
累加	int sum+=a;
累乘	int c*=a;
卷积	int sum+=a*b;
平方和	int sum=a*a+b*b;
复数加法	float_Complex c=a+b;
复数减法	float_Complex c=a-b;
复数乘法	float_Complex c=a*b;

本文采用加速比来表示程序优化前后性能的好坏，加速比越大，表示程序经过特殊指令合成前后周期之间的差距越大，程序的优化性能越显著；相反则表示程序的优化性能不够明显。实验结果详见表2。

表2 特殊指令合成前后的时钟周期数(cycles)

Tab.2 The number of clock cycles before and after the special instruction is synthesized

运算	指令合成前	指令合成后	加速比
累加	356	345	1.03
累乘	356	345	1.03
卷积	362	339	1.07
平方和	347	333	1.04
复数加法	397	366	1.08
复数减法	397	366	1.08
复数乘法	407	382	1.06
平均			1.05

从表2中可以看出，针对表1中的7个基准测试用例，采用本文提出的特殊指令合成策略，其平均加速比大约为1.05。因此，通过本文提出的特殊指令合成策略，能够使得DSP中一

些常用的特殊操作运算程序的性能得到很大的提升。

参考文献(References)

[1] Jordans R,Jóźwiak L,Corporaal H,et al.Automatic instruction-set architecture synthesis for VLIW processor cores in the ASAM project[J].Microprocessors and Microsystems,2017:114-133.

[2] Choi H,Kim J S,Yoon C W,et al.Synthesis of application specific instructions for embedded DSP software[J]. IEEE Transactions on Computers,1999,48(6):603-614.

[3] Gautam Chakrabarti,Fred Chow G.Chakrabarti,F. Chow.Structure Layout Optimizations in the Open64 Compiler:Design,Implementation and Measurements.Gautam Chakrabarti,Open64 Wor kshop at CGO,2008.

[4] 王昊,黄光红,王向前.基于BWDSP100的传播分簇算法研究与实现[J].中国集成电路,2014,23(8):24-28.

[5] 蒋奕.龙芯1编译器中的指令调度相关优化[D].北京:中国科学院研究生,2004:8-11.

作者简介：

陈金娥(1979-)，女，硕士，助教.研究领域：软件工程.

黄胜兵(1990-)，男，硕士，工程师.研究领域：系统软件开发.

(上接第23页)

[3] Emina Aličković, Abdulhamit Subasi.Breast Cancer Diagnosis using GA Feature Selection and Rotation Forest[J].Neural Computing & Applications,2017,28(4):753-763.

[4] 毛林,陆全华,程涛.基于高维数据的集成逻辑回归分类算法的研究与应用[J].科技通报,2013(12):64-66.

[5] 谢忠红,张颖,张琳.基于逻辑回归算法的微博水军识别[J].微型机与应用,2017(16):67-69.

作者简介：

刘　蕾(1978-)，女，硕士，副教授.研究领域：数据挖掘,大数据.

(上接第26页)

目前Web Portal已经在材料科学、生命科学和工程力学等领域获得了广泛的应用，已经封装和定制的应用包括Amber、Gaussian、NAMD、Fluent和MATLAB等多个学科领域的应用，也可以很方便地增加更多的应用。

4 结论(Conclusion)

中山大学重视学科建设，整合校内各方面的科研资源构建国内高校先进的高性能计算服务平台。院系各科研团队利用该平台在各学科领域开展深入研究，在物理化学、环境大气科学、生命科学、光学工程、计算科学、海洋科学、材料科学、流体力学等多个领域发表高水平科研成果。基于平台开展的科学研究项目包括国家自然科学基金项目、国家科技重大专项、国家重点基础研究发展规划(973)项目、省自然科学基金项目和广东省科技攻关项目等，其中有两个项目获得2011年度国家科技进步二等奖，在此基础之上培养了一批具备交叉学科的科研人员，从整体上提升了中山大学科研学术水平，于2013年成功孵化并建成了中山大学国家超级计算广州中心。

参考文献(References)

[1] 郑宁,王冰,党岗.广州超级计算中心应用发展与研究[J].计算机工程与科学,2013,35(11):187-190.

[2] 迟学斌,胡永宏.我国超级计算发展状况研究[J].调研世界,2013(8):56-60.

[3] 张云泉,袁国兴.中国高性能计算及TOP100排行榜[EB/OL].http://www.samass.org.cn,2013-10-21.

[4] Meuer H，Simon H，Strohmaier E,et al.TOP500 super-computer sites [EB/OL].http://www.top500.org,2013-10-21.

[5] 林皎,张武生,徐伟平,等.百万亿次集群机的建设和部署[J].实验室研究与探索,2013,32(6):188-190.

[6] Monitoring with Ganglia,Massie,Matt;Li,Bernard;Nicholes,Brad 2012-11.

[7] 中山大学高兴能计算网格监控平台[EB/OL].http://hpccmonitor.sysu.edu.cn/ganglia,2013-10-21.

[8] CLUSTER RESOURCES,TORQUE Administrator's Guide version 2.4[EB/OL].http://www.clusterresources.com,2013-10-21.

[9] 牛铁,朱鹏,赵毅,等.超级计算环境配额系统设计与实现[J].计算机应用,2010,30(12):8-9;39.

[10] 李惠欢,杨敏,吴汝明.基于 TORQUE 的高性能计算平台记账系统[J].计算机应用与软件,2016(8):126-130.

[11] 广东省教育科研网格门户系统[EB/OL].http://hpcc.sysu.edu.cn,2013-10-21.

[12] 杨敏.广东省教育科研网格门户系统构建[J].武汉大学学报(理学版)2012,58(10):371-375.

[13] 杨敏,关伟豪,朱敏.面向超级计算中心的运营管理支撑平台的设计与实现[J].实验技术与管理,2015,32(6):243-246.

作者简介：

杨　敏(1979-)，女，硕士，工程师.研究领域：高性能计算系统架构，大数据系统架构.

李惠欢(1975-)，女，硕士，工程师.研究领域：软件开发.