TASKING®

揭开
TASKING RISC-V 编译器的神秘面纱
白皮书

揭开 TASKING RISC-V 编译器的神秘面纱

关于FuSa和符合信息安全标准的软件开发的突破性成果

作者: Gerard Vink

介绍

TASKING是领先的汽车功能安全行业创新编译器解决方案提供商，现已推出了适用于RISC-V架构的编译器工具集。在编译器这个领域中，一个问题浮现出来：这一产品是否能够极大的促进基于RISC-V的系统级芯片（SoCs）在汽车行业的普及？本白皮书深入探讨了这个问题。

本文的第一部分强调了RISC-V编译器的特点及其在FuSa相关环境中的使用。第二部分给出了TASKING专有编译器技术框架Viper的全面概述。该框架是所有TASKING编译器的基础，对于TASKING公司在汽车行业的成功起着关键作用。

RISC-V编译器

RISC-V指令集的设计旨在适应各种应用。RISC-V有着四个基本指令集架构（ISA）的正式版本，每个基本ISA可以与许多ISA扩展相结合。虽然某些ISA扩展明显影响着编译器的实现，如“整数乘法和除法的标准扩展”，但其他一些，特别是与内存保护有关的扩展，可能最初看起来对编译器没有直接影响，但却可能引起细微的副作用，导致某些编译器优化失效。

这种RISC-V SoCs的多样性和可配置性为编译器开发人员带来了重大挑战。支撑RISC-V编译器实现的编译器技术框架必须具备与所支持的ISA和扩展ISA相匹配的多样性和可配置性水平。基于各种框架应用经验得到的结论，TASKING的工程师普遍认为Viper是最适应性最强的编译器技术框架。

在重新定位Viper编译器的过程中，绕过了传统的C/C++代码编写方法。相反，大多数编译器的组件是使用特定领域的语言精心制作的。这种战略方法简化了重新定位过程并减少了错误。负责将特定领域输入转换为C/C++的翻译器，会对输入规范进行仔细检查，以确保编译器的早期版本在代码正确性和适用于功能安全相关开发方面有出色的表现。最终的编译器可执行文件是通过C/C++到机器代码的编译生成的，可以应用各种优化技术来提高编译器的执行速度。

在功能安全相关环境中使用编译器工具集，对工具集本身、其文档和支持组织提出了特定要求。凭借在汽车行业超过30年的经验，TASKING善于满足这些严格的要求。下图展示了一系列与功能安全相关的产品，这些产品支持您的组织，在产品的整个生命周期内能够满足功能安全和信息安全相关法规和标准。

图1. TASKING安全生命周期支持

TASKING对其功能安全和信息安全文档的成熟度感到非常自豪。值得注意的是，TASKING编译器是首批获得ISO/SAE 21434标准认证的编译器。这一投入不仅仅局限于编译器工具本身；相关函数库也正在进行ISO 26262和IEC 61508的认证工作。功能安全和信息安全的成熟度和结构化格式，使我们的客户能够快速进行他们的工具资格认证，以最小的工作量和成本实现功能安全和信息安全标准的合规性。

VIPER编译器技术框架

Viper编译器技术框架开发出来，是为了超长指令字（VLIW）体系结构（可以提供指令级和数据级并行性）能够以经济高效的方式构建高度优化的编译器。现代RISC和CISC ISA的特性与可选的DSP扩展是VLIW领域的一个子集，因此由Viper框架支持。请注意，以下文字描述的是编译器技术框架，因此该文本并非特定于RISC-V。

只有精确针对目标处理器进行配置（ISA、ISA扩展和指令管道行为）的编译器才能生成高效的机器代码。在可重新配置IP的背景下，只有可重定目标的编译器，即在编译时（相对于编译器构建时）可针对不同目标系统进行配置的编译器，才能利用对处理器核心所做的修改。除了由硬件可配置性引入的复杂性外，编译器开发还受到诸如指令级并行性（ILP）、单指令多数据（SIMD）和非正交指令集等设施的影响，这进一步增加了复杂性。

Viper框架提供了一个多用途环境，可快速高效地开发编译器，这些编译器可以轻松适应特定的核心架构变体。如图2所示，TASKING编译器分为两个主要组件：前端和后端（也称为代码生成器）。前端执行词法和语法分析、语义分析、优化和中间代码生成。前端的特点是其行为主要由源语言确定，并且在很大程度上独立于目标系统。由Viper前端生成的中间代码称为Medium level Intermediate Language（MIL）。 MIL是源代码的规范等效，包含了生成代码所需的所有信息（例如符号表和调试信息）。

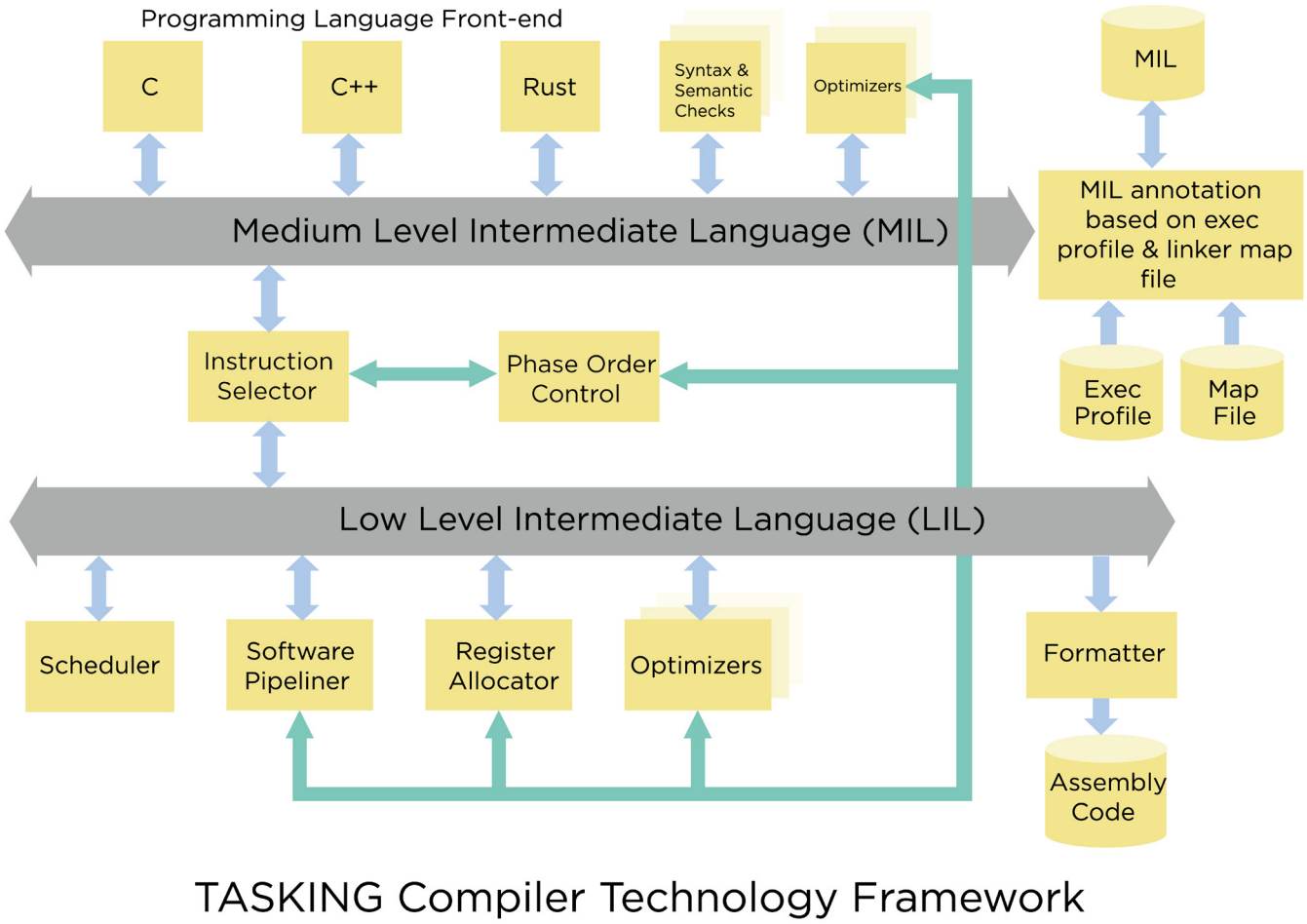

图2. Viper编译器技术框架

后端包含必须与目标系统对齐的编译器的那些字段。第一个后端阶段是指令选择器，它将前端的MIL代码转换为低级中间语言（Low-level Intermediate Language，LIL）。使用一种称为目标描述语言（Target Description Language，TDL）的特定领域语言定义了LIL对象（Lilops）。 Lilops是指目标处理器的指令，具有操作码、操作数和编译器使用的信息，例如指令吞吐量、指令延迟和流水线依赖关系等。

后端的下一步是在低级中间语言（Low Intermediate Language）级别进行指令调度、寄存器分配和优化。最终的后端阶段是生成汇编代码的格式化程序。中间语言MIL和LIL充当了在优化和代码生成阶段之间传输信息的总线。

Viper编译器构建框架提供了大量在MIL或LIL级别操作的本地和全局优化器。优化器的行为和调用顺序由阶段顺序控制组件管理，这取决于目标系统和应用软件。编译过程可以通过向编译器提供来自链接器映射文件和执行配置文件的数据来进一步引导。

VIPER前端

C和C++前端以及原型Rust前端都是有效的。前端提供了以下功能:

- DSP C/C++语言扩展,
- 用户可指定的调用约定,
- 内联汇编代码,
- 内置函数,
- 常规优化,
- 应用程序范围的优化,
- 循环变换,
- 用于控制寄存器压力的反馈循环
- 处理器和应用程序特定的优化。

后端（代码生成）

即使对于正交的、同质的RISC体系结构，最佳利用指令级别和数据级别的并行性也绝非易事。然而，对于DSP来说，代码生成显然更加复杂，而且需要其他技术。寄存器分配通常会受到寄存器集异构性的复杂性的影响。指令调度和软件流水线不再是独立的阶段，而必须与指令选择和寄存器分配进行交互。Viper后端被设计用于处理上述与DSP相关的复杂性。

后端的主要组件和部分（图3）包括:

- 预分配器为本地变量分配存储空间，读取和修改符号，并设置虚拟寄存器。
- 指令选择器读取前端生成的MIL和符号信息，并生成相应的LIL。
- 窥视孔（Peepholer ）使用LIL编辑器遍历LIL流并根据LIL编辑器模式文件进行改进。
- 通用LIL优化器执行许多不同的优化，由于TDL中Lilops的通用规范，优化算法的实现无需为特定目标架构定制。
- 指令调度器是一组代码生成器阶段，将Lilops组织成最适合目标处理器的顺序。指令调度器通常在寄存器分配之前和之后执行。
- 寄存器分配器为每个虚拟寄存器分配一个物理寄存器。后端和前端之间的反馈循环防止了前端优化导致寄存器压力过大，这对于优化的整体结果是有害的。
- 代码压缩器通过搜索相同的指令序列来减小代码大小。如果序列出现足够频繁，那么该序列的所有实例都将被替换为对单个现有副本的调用。
- 格式化程序读取LIL操作以生成汇编代码。它是代码生成器的最后阶段。

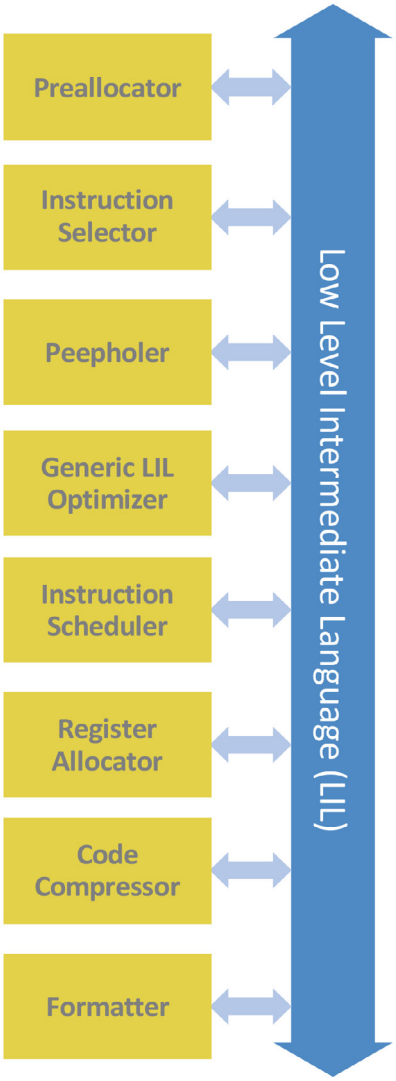

图3. Viper后端的主要组件和部分

指令调度器是一个复杂的、多阶段的组件，它与后端的其他部分（例如寄存器分配器）交互，为特定的目标系统生成最优的指令序列。调度器可以包括以下组件:

- IF转换器：将If-Then-Else结构替换为谓词指令，可更高效地处理。
- 谓词指令推广器：消除LIL操作之间的依赖关系。
- 分区器：适用于具有两个或更多寄存器集的目标系统，每个集分配给一个或多个功能单元。
- 软件流水线：优化内部循环的调度，跟踪每个寄存器类的寄存器使用情况，以避免寄存器溢出。这是促进自动向量化的重要组件。
- 列表调度器：与目标系统无关的实现，从目标系统的TDL文件中获取所有目标系统特定信息。列表调度器在软件流水线和寄存器分配之后运行，以便调度任何溢出代码。
- 延迟槽填充器：在寄存器分配之后执行，支持适用于那些受益于此的处理器的各种延迟槽概念。

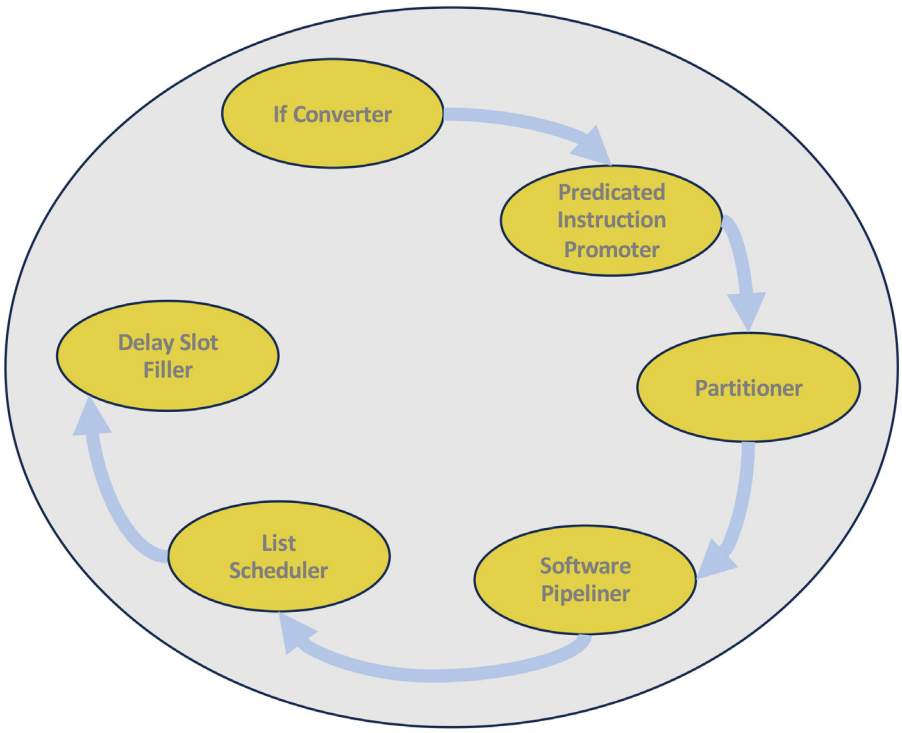

图4指令调度器的主要组件如

语言扩展和优化:

尽管ISO-C和C++编程语言旨在开发与底层硬件交互的应用程序，但对于典型的DSP架构特性缺乏支持。DSP-C是为了满足C级数字信号处理的特殊要求而开发的。它是ISO/IEC规范9899:2011(E)的扩展。

为了满足DSP的代码密度和性能要求，Viper支持DSP-C语言扩展、经典优化、应用程序范围的优化、目标系统依赖的优化和循环转换。通过利用DSP支持的并行数据和指令处理，可以实现执行速度的巨大提升--这是Viper技术的关键特性之一。

可配置的IP核和SOCS:

Viper框架允许构建编译器，在编译时可以指定其行为以便对硬件进行更改和调整。主要的架构修改，如引入新的指令和寻址类型或引入新的流水线架构，需要创建一个新的编译器。以下更改可以直接进行，无需创建新的编译器:

- 删除操作码、寄存器和寻址类型，。
- 改变功能单元的数量和组成。

流程成熟度、功能安全和信息安全:

产品开发流程符合ASPICE二级，确保产品质量可预测和可重复。编译器及其相关函数库的功能安全和信息安全认证所需证据的创建是无缝集成到开发过程中的。因此，第一个版本的编译器就是一个高质量的产品，适用于功能安全和信息安全相关的软件开发。