**具有多个RISC-V和CGRA内核的低能耗异构计算**

**摘要**-组合多个CPU和CGRA内核的想法本身并不是最初的想法，但是很难在文献中找到此类架构的详细特性以及在引人注目的应用中进行的测量。尽管商用CPU，GPU和FPGA广泛可用，但没有商用CGRA，这可能归因于缺乏性能，能耗和成本指标。在本文中，我们介绍了一个由多个RISC-V CPU和Versat CGRA内核组成的异构计算平台。给出了该架构的多个实例的实现结果。选择的CPU是有前途的开源RISC-V架构，它从未在CPU / CGRA架构中出现过。本文介绍了两个RISC-V内核的独立实现：一个最小的RISC-V内核（可用作简单控制器）和一个性能更高的5级管道实现。 RISC-V内核是使用最新的Chisel HDL设计的，可用于自动执行与RTL编写有关的任务。所选的CGRA是已发布的Versat体系结构，已为其创建了4个不同的实例。给出了2个FPGA系列和ASIC技术节点的实现结果：面积，频率和功率。应用程序涵盖了数字音频和机器学习，证明了该平台在竞争性区域，频率和能源足迹方面的多功能性。

**介绍**

随着基于神经网络的物联网（IoT）和人工智能（AI）算法的出现，降低硅器件的成本和能耗变得越来越重要。使用高性能CPU，GPU，FPGA或这些[1]，[2]的组合的系统很有趣，但很难满足那些需要无处不在的低成本和低功耗设备的应用的需求。

高性能CPU可以并行执行一些操作，但是其大多数硬件专用于有效地处理指令，这对于预想的系统来说是很大的开销。 FPGA设计为构建任意数字电路，并且需要强大的配置基础架构，这对于低能耗IoT系统而言是过大的杀伤力。 GPU包含大量简化的von Neumann处理器，出于相同的目的，与专用硬件数据路径相比，它们需要更多的硬件。最后，完全用硬件实现的系统可能会具有最佳性能，但会以较大的硅片面积（较高的设备成本）为代价，并且缺乏可编程性。

鉴于上述方法的缺点，一个有趣的选择似乎是一个由几个简单的CPU内核与可编程硬件内核组合而成的系统， 内核与可编程硬件内核相结合，比FPGA更简单。 对于需要大量控制的算法，CPU的使用实际上是一种资源的节约，因为单个CPU可以调度和执行许多控制结构。 这相当于在一小块硬件中对控制任务进行时间复用。 对于并行任务，可以添加更多CPU。 以类似的方式进行推理，可以使用可编程硬件对数据处理任务进行时分复用：大型硬件电路可以分解为较小的部分，并在可编程硬件上依次运行。

由于FPGA对于大多数应用来说过于繁琐，因此适用于嵌入式设备的一种更合适的可重构硬件类型是粗粒度可重构阵列（CGRA）[3]。 CGRA是通过可编程开关互连的可编程功能单元（FU）的集合。 FU对常用宽度的数据字（例如8位，16位，32位或64位字）进行操作。 对CGRA进行编程后，它将实现可加速计算的硬件数据路径。

在这项工作中，提出了一个由多个RISC-V CPU [4]和多个Versat CGRA [5]组成的异构计算平台。 RISC-V内核是根据名为Adept的项目代码开发的。 在[6]中提出了一种类似的方法，但是这里我们以使用RISC-V CPU和Versat CGRA为例。

为了维持IoT革命并释放数百万工程师的创造力，拥有一套不含知识产权（IP）权利或许可费用的指令集架构是必不可少的[7]。 在少数可用的免费指令集架构（ISA）中，RISC-V似乎是最有前途的一种，并得到了学术界和工业界的广泛支持。 其他可用的开源解决方案要么依赖于特定技术，没有参数化选项，要么具有非鲁棒的开发环境[8]。

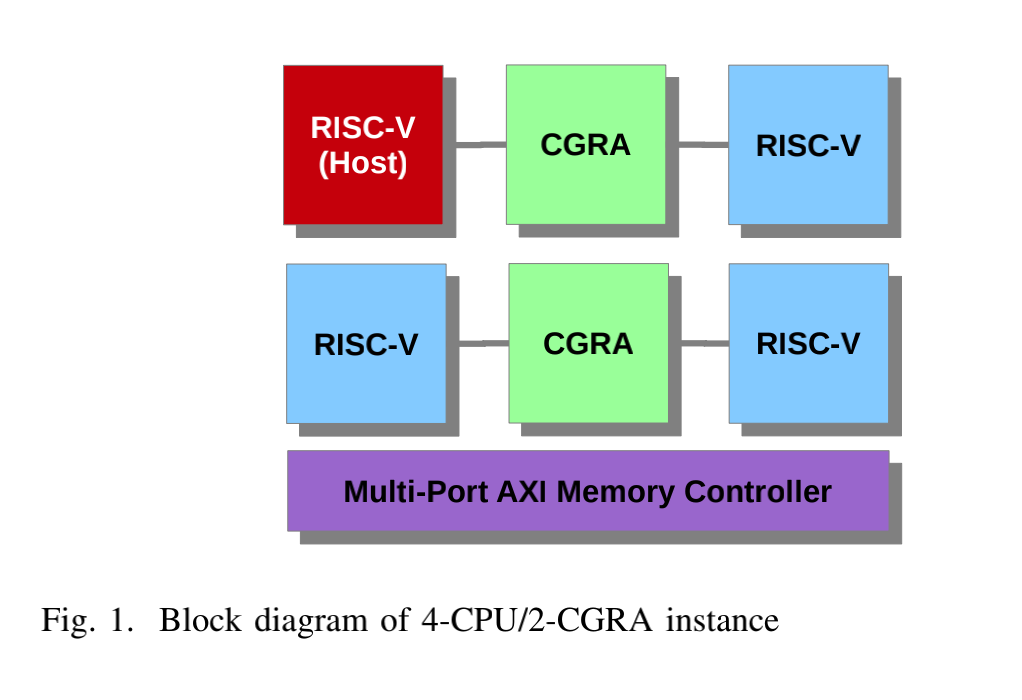
随着RISC-V [4]及其开源工具链的出现，特别强调Chisel硬件描述语言（HDL）[9]和Rocket Chip SoC Generator [10]，开源很可能成功 操作系统已扩展为可打开CPU硬件描述。

先前曾尝试创建一个称为Blackbird的片上系统（SoC），并尝试过使用OpenRISC开源处理器来构建可重配置系统的编程环境[8]。 Blackbird打算成为ORPSoC [11]（相当于Rocket Chip SoC Generator [10]的OpenRISC）的独立SoC。不幸的是，由于预算用完，这项工作失败了，主要是因为总是依赖于特定的FPGA板或专有的EDA工具，而Blackbird项目却无法支持这些依赖。而且，OpenRISC计划未能创建自己的足够成熟的生态系统，而且努力也显示出其自身过于复杂，因为该系统的许多基本部分要么缺失，要么仍由小型现有社区开发。实际上，尽管最初有热情，但RISC-V计划也可能发生同样的情况。但是，其当前的推动力远比OpenRISC所享有的强大，如今，诸如Github之类的Web平台促进了开源开发。

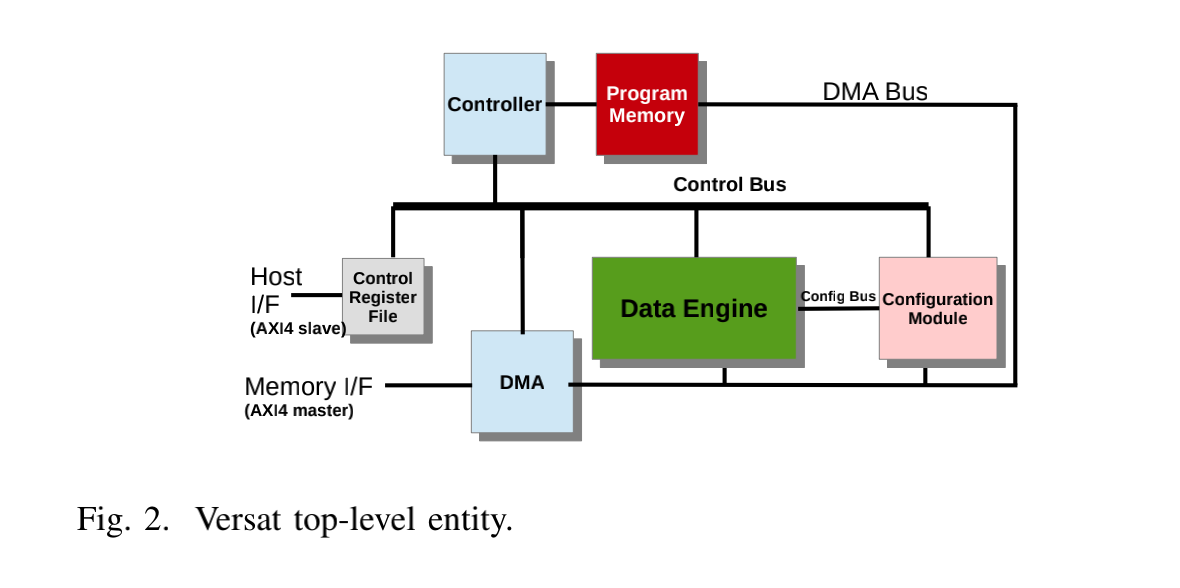
[8]中报道的工作最近一直在使用RISC-V处理器继续进行，并尝试建立一个名为Warpbird [12]的新SoC。 不幸的是，与使用OpenRISC内核时一样，该团队遇到了类似的困难，这导致了一个新的研究问题的发现：构建和不加考虑的SoC [12]。 现有的RISC-V开源项目证明不够模块化或没有足够的文档记录，因此其各个部分都可以轻松地重用。 实际上，事实证明，从头开始编写RISC-V内核要容易得多，并且开始使用Chisel语言进行Adept项目。 当Chisel生成可综合的Verilog代码时，本文提出的系统可移植到任何FPGA或ASIC技术。 GNU工具链支持RISC-V CPU，而Versat CGRA具有自己的编译器和汇编器[13]。

**结构**

所建议的体系结构可用于开发包含多个Adept RISC-V内核和Versat CGRA的SoC。 图1描绘了一个由4个RISC-V内核和2个Versat CGRA组成的示例系统。该系统使用ARM的高级可扩展接口（AXI）互连内核，这是总线的事实上标准。



在图中，两个CPU内核共享同一CGRA，该CGRA的2端口AXI从接口连接到其主机的AXI主接口。 CGRA使用循环机制在两个主机之间进行仲裁。 所有CPU和CGRA内核都通过连接到AXI从属多端口存储器控制器内核（其是第三方内核）的AXI主接口访问外部存储器。 必须为所有内核的程序和数据加载外部存储器。 这是由主机CPU完成的，该主机运行引导加载程序，并能够通过AXI或SPI外设（图中未显示）从外部设备获取这些数据。 接下来的小节将提供有关RISC-V CPU内核和Versat CGRA内核的详细信息。



1. Adept

为了构架这项工作，已经研究了四种RISC-V体系结构：Taiga [14]，一种高性能RISC-V软核； Rocket Chip [10]，这是加州大学伯克利分校和各自的SiFive初创公司使用的事实上的标准； PicoRV32 [15]，一种尺寸优化的架构； PULPino [16]，一种单核微控制器系统。这项工作中拟议的SoC有一组非常具体的要求：体积小，功耗低，目标FPGA和ASIC，并且具有较高的抽象级别描述。因此，未选择Rocket Chip，因为它具有太多功能且难以操作。未选择Taiga，因为它仅针对FPGA。未选择PULPino和PicoRV32，因为它们是用常规HDL编写的。在该项目中，现代编程语言中存在的抽象是必需的，即有助于RTL的参数化。因此，在考虑了所有4种架构之后，决定使用RV32IM ISA和新的Chisel3 HDL从头开始实现新的RISC-V内核[9]。新的架构称为Adept。

有两种Adept配置：Adept-3，这是一个三级管道低性能配置，它实现了基本整数（I）指令集； 和Adept-5，这是一个5级管道中等性能配置，它实现了整数乘法和除法（IM）指令集。 两种配置均针对硅面积进行了优化，仅支持裸金属应用。 两者均使用大小分别为8和16kB的指令和数据高速缓存。

1. Versat

Versat体系结构[5]如图2所示。Versat使用Controller运行程序，使用Data Engine（DE）进行数据密集型计算。 Controller程序存储在程序存储器（PM）中，该存储器具有1kB的引导ROM和8kB的用户RAM。 控制器使用控制总线访问系统中的各个模块。 引导ROM程序可以使用DMA或特殊外围设备（例如SPI内核）将用户程序从外部存储器或其他设备加载到PM中。 用户程序可以为各种加速度数据路径生成DE配置，并将其存储在配置模块（CM）中。 它还可以使用DMA引擎在CM和外部存储器之间移动配置。

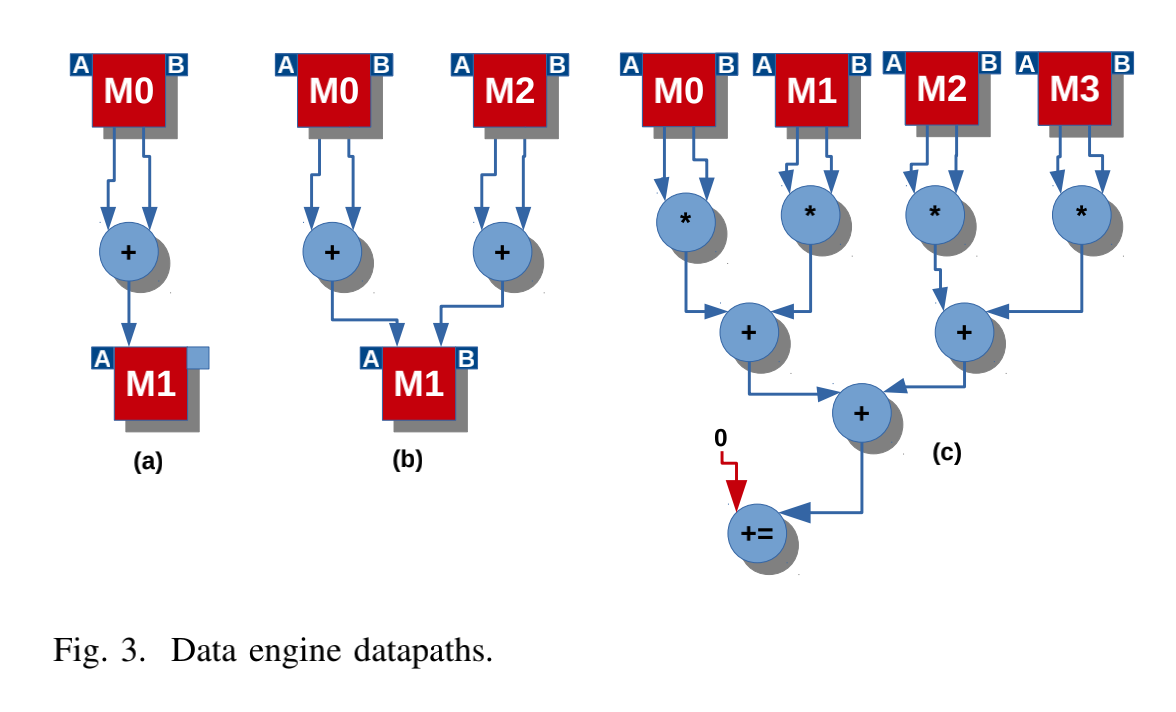
Versat内核具有主机接口和内存接口。 主机系统使用主机接口（AXI从机）来指示Versat加载和执行程序。 主机和Versat使用共享的控制寄存器文件（CRF）进行通信。 Versat程序还将CRF用作通用（1周期访问）寄存器文件。 DMA使用存储器接口（AXI主站）访问外部存储器。

可以生成包含不同DE模块的不同Versat实例。 DE由几个以全网格互连的功能单元（FU）组成。 当前的FU类型是：双端口嵌入式存储器，ALU（带有和不带有内部反馈以累加结果），乘法器和桶形移位器。

互连网格具有1个时钟周期的延迟，并且FU本身通过流水线传输以提高性能。 FU用于构建在数据流上工作的硬件数据路径。 示例数据路径如图3所示。图中的数据路径（a）是通过并行执行加载，添加和存储指令来利用指令级并行（ILP）的简单管道； 数据路径（b）展示了ILP加上数据级并行性（DLP）或线程级并行性（TLP）（如果两个添加项作为独立线程运行）； 数据路径（c）包含更多的FU，从而实现更多的并行性。

完整的网格结构看似过大，但它极大地方便了Versat的编程，省去了复杂的布局和路线算法。 这样，甚至可以使用其汇编语言对Versat进行编程，这是CGRA中罕见的功能。 由于使用的FU数量相对较少，全网状互连变得负担得起，仅占芯片面积的5％[5]。

当在FPGA中实现时，Versat是一种覆盖体系结构，也就是说，可重新配置的FPGA结构用于支持不同的Versat实例，但是也可以使用它们自己的更快的重新配置基础结构来重新配置实例本身。 当以ASIC实现时，Versat是一种可编程逻辑，可减少芯片面积，降低设计错误的风险并节省能源，同时提供类似于硬件的性能。



**工具链**

1. Adept

Adept RISC-V系统具有用于编译，调试和性能分析的功能全面的GNU工具链。 这些是成熟的工具，大多数已在上游合并，从7.1版开始，它们的状态可以在[17]中进行检查。 有一些选项可以使编译知道浮点单元，特定板的时钟频率等。

RISC-V已提供使用GDB的调试支持，但Adept尚无硬件支持。 调试模块和相应的JTAG接口尚未合并。

与Rocket Chip SoC Generator一起使用时，调试模块及其通过OpenOCD提供的软件支持[18]在Warpbird设置[12]中无法互操作，因此无法集成到Adept中。 Rocket Chip调试模块和JTAG接口需要在不同的环境中使用，或者需要从头开发新的环境。

新型硬件描述语言Chisel3中描述了Adept系统[9]。 此外，用户可以使用其首选的HDL向系统添加组件。 但是，作者建议使用Chisel，因为它可以自动执行并简化诸如调试，连接和接口生成之类的任务。

1. Versat

作为CGRA，Versat开发工具是非标准的。 Versat可以用汇编语言[5]和C ++方言[13]进行编程。

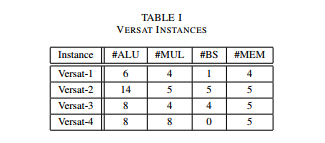
由于Versat控制器通过控制总线管理数据引擎（DE）和DMA引擎，因此可以间接对这些模块进行编程。为了对DE进行编程，将数据路径写入配置模块（CM）。为了操作其他模块，Versat控制器会像使用外设时一样访问其内存映射寄存器。

例如，要配置ALU0以将乘法器MUL1和MUL2的输出相加，程序需要将常量ALU ADD存储到ALU0 FNS功能选择配置寄存器，并将常量sMUL1和sMUL2存储到配置寄存器ALU0 selA和ALU0 selB。 Versat的全网格结构使任何FU的输入都可以连接到任何FU的输出。如果使用局部网格，则程序员必须牢记什么可以连接到什么，从而使汇编编程不可行。

尚未开发用于Versat的调试工具。调试失败的数据路径可能是一项艰巨的任务，必须采用多种技术。 Versat控制器在此过程中起着重要作用，因为它可以启动DE并在所需的瞬间停止它，并且可以读取和写入DE存储器的任何位置。在极端情况下，必须使用RTL仿真或集成逻辑分析仪对Versat进行调试。

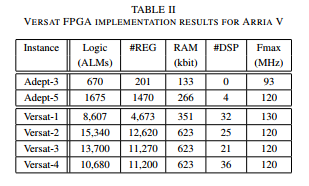
**结果**

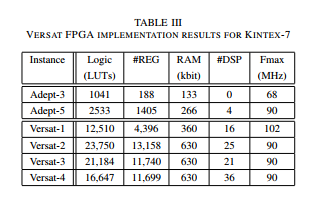
对于表1中描述的4个不同的Versat实例，我们得到了ALU数量（#ALU）、32x32=64位乘法器的数量（#MUL）、桶形移位器的数量（#BS）和嵌入式存储器（#MEM）的数量。



*A. FPGA Implementation Results*

Intel ARRIA V和Xilinx KINTEX-7器件的FPGA实现结果分别如表II和表III所示。



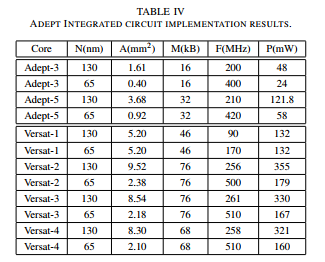


从这些结果可以看出，资源的使用是非常合理的。Adept处理器可与标准FPGA处理器相媲美，如Microblaze（Xilinx）或NIOS（Intel），而Versat内核则允许紧凑的实现，而定制的硬件则不能满足多种功能。Adept-3没有针对频率进行优化；它的关键路径是从回写阶段的多路复用器到指令获取阶段的分支执行块，并遍历ALU。Adept-5中的关键路径是从内存级到寄存器获取阶段的转发路径。

结果表明，Versat核心可以在对操作频率影响最小的情况下以不同的尺寸实现。然而，由于全网格结构，频率被限制在90-130mhz范围内，这取决于实例的复杂性。

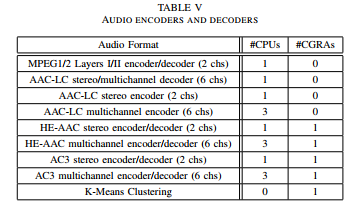
*B. ASIC Implementation Results*

Versat和Adept已在UMC 130 nm和TSMC 65 nm工艺中实现。表4显示了技术节点（N）、硅面积（A）、嵌入式存储器（M）、操作频率（F）和功耗（P）的结果。利用Cadence集成电路设计和仿真工具，得到了频率和功率的结果。利用相同的工具得到功率图，并从仿真中提取节点活动率。

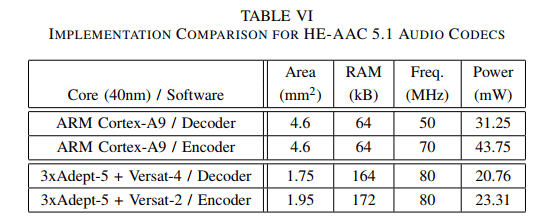


结果表明，所提出的异构计算平台在硅面积和功耗方面具有竞争力。一些老练的和Versat的核心在65nm的过程中消耗几百毫瓦，而单臂Cortex A9核心可以消耗大约相同的功率，在40nm的过程中占据更大的硅面积[5]。

1. 应用：目前的异构计算平台面向低成本电池供电设备的应用。这些应用程序可以受益于cgra的低能耗特性。表V概述了在当前体系结构中测试的应用程序。



将当前平台与运行相同应用程序的ARM Cortex A9处理器进行了比较。 不幸的是，无法与其他CGRA进行比较[3]，因为它们无法使用相同的应用程序进行编程。 在[5]中使用已发表的结果进行了比较，得出的结论是，它们取决于所比较的CGRA的资源。 在[19]中，与ARM Cortex-A9系统相比[20]，[21]，多维K-Means算法的运行速度提高了3.8倍，而在Versat上的能耗却减少了46.3倍。 此处，表VI中显示了针对HE-AAC 5.1音频编码器和解码器系统获得的结果。 Adept + Versat系统的结果已标准化为40nm，以便于直接比较。



对于HE-AAC 5.1音频解码器，ARM系统比拟议的实现大2.6倍，功耗高1.5倍。 对于HE-AAC 5.1音频编码器，ARM系统比建议的实现大2.4倍，并且功耗高1.9倍。 ARM Cortex A9系统使用在组装级别进行了优化的NEON SIMD单元。 所提出的系统需要使用比ARM系统稍高的工作频率和更多的内存。 但是，这可以通过使用简单的CPU和以数据为中心的CGRA来弥补，而这些CGRA消耗的逻辑资源要少得多。

**结论**

在这项工作中，提出了一种针对任何设计流程（FPGA或ASIC）的低功耗异构系统，该系统由RISC-V处理器和Versat CGRA内核组成。 设计好的RISC-V内核Adept具有两种配置：小型3级管道配置和性能更高的5级管道配置。 在这两种情况下，Adept都具有极富竞争力的能源足迹和资源使用率。 Versat CGRA允许以最小的能源需求和高性能执行大量应用程序。

RISC-V ISA可以开发使用一个或多个处理器实例的SoC，而无需支付许可费或特许权使用费。此外，RISC-V ISA得到了丰富的开源工具链和蓬勃发展的开发人员社区的支持。在这项工作中，得出的结论是，内部开发RTL仍然更好，因为现有的开源项目难以操纵和扩展。

当在FPGA中实现时，本平台构成了覆盖架构。与直接映射到FPGA架构相比，应用程序在其上的运行速度较慢。但是，使用快速的实时重新配置，Versat能够将大型虚拟硬件电路多路复用到其小型CGRA中，并支持原本不适合FPGA的应用。实施结果表明，可以在中档FPGA中实例化几个RISC-V和Versat内核，并运行实际应用，例如此处介绍的音频编码器和解码器。与配备NEON SIMD单元的ARM Cortex-A9系统相比，当以ASIC实施时，拟议的异构计算平台消耗的硅面积减少近3倍，能耗减少2倍。