RISC－V 架构的开源处理器及 SoC 研究综述

雷思磊

（酒泉卫星发射中心，酒泉 735000）

摘要：RISC－V 是一种新的指令集架构，发布以来得到了大量关注，在描述了 RISC－V 的产生背景、基本设计的基础上，简单比较了其与现有的开源指令集架构、商业指令集架构的优劣，然后详细介绍了现有的采用 RISC－V 架构的开源处理器、开源 SoC，并展望了 RISC－V 的未来发展。

关键词：RISC－V；Rocket；BOOM；SoC

中图分类号：TP368.1　　**文献标识码**：A

Research on Open Source Processor and SoC Based on RISC-V

Lei Silei

（Jiuquan Satellite Launch Center，Jiuquan 735000，China）

Abstract：RISC-V is a new instruction set architecture(ISA)，and it gets a lot of attention since the release. On the basis of describing the background and the basic design of RISC-V，the advantages and disadvantages with the existing open-source instruction set architecture and commercial instruction set architecture are compared. Then the existing open source processor and SoC using RISC-V architecture are described. Finally，the future development of RISC-V is prospected.

Key words：RISC-V；Rocket；BOOM；SoC

引　言

RISC－V 是加州大学伯克利分校（University of California at Berkeley，以下简称 UCB）设计并发布的一种开源指令集架构，其目标是成为指令集架构领域的 Linux，应用覆盖 IoT（Internet of Things）设备、桌面计算机、高性能计算机等众多领域。其产生是因为 UCB 的研究人员在研究指令集架构的过程中，发现当前指令集架构存在如下问题[1]：

① 绝大多数指令集架构都是受专利保护的，比如：x86、MIPS、Alpha，使用这些架构需要授权，限制了竞争的同时也扼制了创新。

② 当前的指令集架构都比较复杂，不适合学术研究，而且很多复杂性是因为一些糟糕的设计或者背负历史包袱所带来的。

③ 当前的指令集架构都是针对某一领域的，比如：x86 主要是面向服务器、ARM 主要是面向移动终端，为此对应的指令集架构针对该领域做了大量的领域特定优化，缺乏一个统一的架构可以适用多个领域。

④ 商业的指令集架构容易受企业发展状况的影响，比如：Alpha 架构就随着 DEC 公司的被收购而几近消失。

⑤ 当前已有的各种指令集架构不便于针对特定的应用进行自定义扩展。

为此，UCB 的研究人员 Krste Asanovic、Andrew Waterman、Yunsup Lee 决定设计一种新的指令级架构，并决定以 BSD 授权的方式开源，希望借此可以有更多创新的处理器产生，有更多的处理器开源，并以此降低电子产品成本[2]。RISC－V 自 2014 年正式发布以来，受到了包括谷歌、IBM、Oracle 等在内的众多企业，以及包括剑桥大学、苏黎世联邦理工大学、印度理工学院、中国科学院在内的众多知名学府与研究机构的关注和参与，围绕 RISC－V 的生态环境逐渐完善，并涌现了众多开源处理器及 SoC 采用 RISC－V 架构，这些处理器既有标量处理器，也有超标量处理器，既有单核处理器，也有多核处理器。本文简单介绍 RISC－V 架构的基本设计，并详细描述目前采用 RISC－V 架构的开源处理器与 SoC。

1　RISC－V 简介

1.1　RISC－V 的基本设计

RISC－V 是一个典型三操作数、加载-存储形式的 RISC 架构，包括 3 个基本指令集和 6 个扩展指令集，如表 1 所列，其中 RV32E 是 RV32I 的子集，不单独计算。

表 1 RISC－V 的指令集组成

指令集类型	名 称	指令数	说 明
基本指令集	RV32I	47	整数指令，包含：算术、分支、访存。32 位寻址空间，32 个 32 位寄存器
	RV32E	47	指令与 RV32I 一样，只是寄存器数量变为 16 个，用于嵌入式环境
	RV64I	59	整数指令，64 位寻址空间，32 个 64 位寄存器
	RV128I	71	整数指令，128 位寻址空间，32 个 128 位寄存器
扩展指令集	M	8	包含 4 条乘法、2 条除法、2 条余数操作指令
	A	11	包含原子操作指令，比如：读-修改-写，比较-交换等
	F	26	包含单精度浮点指令
	D	26	包含双精度浮点指令
	Q	26	包含四倍精度浮点指令
	C	46	压缩指令集，其中的指令长度是 16 位，主要目的是减少代码大小

基本指令集的名称后缀都是 I，表示 Integer，任何一款采用 RISC－V 架构的处理器都要实现一个基本指令集，根据需要，可以实现多种扩展指令集，例如：如果实现了 RV32IM，表示实现了 32 位基本指令集和乘法除法扩展指令集。如果实现了 RV32IMAFD，那么可以使用 RV32G 来表示，表示实现了通用标量处理器指令集。本文只介绍 RV32I 的基本情况。

RV32I 指令集有 47 条指令，能够满足现代操作系统运行的基本要求，47 条指令按照功能可以分为如下几类：

① 整数运算指令，实现算术、逻辑、比较等运算。

② 分支转移指令，实现条件转移、无条件转移等运算，并且没有延迟槽。

③ 加载存储指令，实现字节、半字、字的加载和存储操作，采用的都是寄存器相对寻址方式。

④ 控制与状态寄存器访问指令，实现对系统控制与状态寄存器的原子读-写、原子读-修改、原子读-清零等操作。

⑤ 系统调用指令，实现系统调用、调试等功能。

1.2 RISC－V 的优势

1.2.1 与开源指令集架构比较

在 RISC－V 发布之前，实际上已经有几种开源指令级架构，包括 SPARC V8、OpenRISC，其中 SUN 发布的开源多核多线程处理器 OpenSparcT1、OpenSparcT2，以及欧空局的 LEON3 采用的就是 SPARC V8，OpenRISC 也有同名的开源处理器，RISC－V 与前两者的比较如表 2 所列。此外，OpenRISC 的许可证为 GPL，这意味着所有的指令集改动都必须开源，而 RISC－V 的许可证是较为宽松的 BSD License 授权。

表 2 RISC－V 与 SPARC V8、OpenRISC 的比较[2]

项 目	SPARC V8	OpenRISC	RISC－V
分为基本指令集＋扩展指令集		是	是
支持压缩指令			是
支持四倍精度浮点运算	是		是
寻址空间			
32bit	是	是	是
64bit		是	是
128bit			是
软件支持			
GCC	是	是	是
LLVM	是	是	是
Linux	是	是	是
QEMU	是	是	是

1.2.2 与商业指令集架构比较

UCB 的研究人员设计了一款采用 RISC－V 指令集架构的开源处理器 Rocket，并且成功流片了 11 次，其中采用台积电 40 nm 工艺时的性能，与采用同样工艺的、都是标量处理器的 ARM Cortex－A5 的性能对比如表 3 所列，可见 Rocket 占用面积更小，且功耗更低，性能更优。

表 3 ARM Cortex－A5 与采用 RISC－V 指令集架构的 Rocket 比较[2]

项 目	ARM Cortex－A5	RISC－V Rocket	Ratio
寄存器宽度	32	64	2
主频	＞1GHz	＞1GHz	1
Dhrystone	1.57DMIPS/MHz	1.72DMIPS/Hz	1.1
面积(不包含 Cache)	0.27mm^2	0.14mm^2	0.5
面积(包含 16KB Cache)	0.53mm^2	0.39mm^2	0.7
动态功耗	＜0.08mW/MHz	0.034mW/MHz	＞0.4

2 基于 RISC－V 的开源处理器研究现状

2.1 标量处理器——Rocket

Rocket 是 UCB 设计的一款 64 位、5 级流水线、单发射顺序执行处理器，主要特点有：支持 MMU，支持分页虚拟内存，所以可以移植 Linux 操作系统；具有兼容 IEEE 754－2008 标准的 FPU；具有分支预测功能，具有 BTB(Branch Prediction Buff)、BHT(Branch History Table)、RAS(Return Address Stack)。

Rocket 是采用 Chisel(Constructing Hardware in an Scala Embedded Language)编写的，这也是 UCB 设计的一种开源的硬件编程语言，是 Scala 语言的领域特定应用，

可以充分利用 Scala 的优势，将面向对象(object orientation)、函数式编程(functional programming)、类型参数化(parameterized types)、类型推断(type inference)等概念引入硬件编程语言，从而提供更加强大的硬件开发能力。Chisel 除了开源之外，还有一个优势就是使用 Chisel 编写的硬件电路，可以通过编译得到对应的 Verilog 设计，还可以得到对应的 C++模拟器。Rocket 使用 Chisel 编写，就可以很容易得到对应的软件模拟器。同时，因为 Chisel 是面向对象的，所以 Rocket 的很多类可以被其他开源处理器、开源 SoC 直接使用。

2.2 超标量乱序执行处理器——BOOM

BOOM(Berkeley Out-of-Order Machine)是 UCB 设计的一款 64 位超标量、乱序执行处理器，支持 RV64G，也是采用 Chisel 编写，利用 Chisel 的优势，只使用了 9 000 行代码，流水线可以划分为 6 个阶段：取指、译码/重命名/指令分配、发射/读寄存器、执行、访存、回写。

借助于 Chisel，BOOM 是可参数化配置的超标量处理器，可配置的参数包括：

① 取指、译码、提交、指令发射的宽度。

② 重排序缓存 ROB(Re-Order Buffer)、物理寄存器的大小。

③ 取指令缓存、RAS、BTB、加载、存储队列的深度。

④ 有序发射还是无序发射。

⑤ L1 Cache 的路数。

⑥ MSHRs(Miss Status Handling Registers)的大小。

⑦ 是否使能 L2 Cache。

UCB 已经在 40 nm 工艺上对 BOOM 进行了流片，测试结果如表 4 所列。可见 BOOM 与商业产品 ARM Cortex-A9 的性能要略优，体现在面积小、功耗低。

表 4 BOOM 与 ARM Cortex-A9 的性能对比[3]

项 目	ARM Cortex-A9	RISC-V BOOM-2W
ISA	32bit ARM v7	64bit RISC-V(RV64G)
微架构	2wide，4 发射乱序处理器	2wide，3 发射乱序处理器
流水线级数	8	6
性能	3.59 CoreMarks/MHz	3.91 CoreMarks/MHz
工艺	台积电 40nm	台积电 40nm
面积(含 32K 缓存)	约 2.5mm^2	约 1.00mm^2
主频	1.4GHz	1.5GHz
功耗	0.5～1.9 W (2 cores + L2) @ TSMC 40nm，0.8～2.0GHz	0.25 W (1 core+L1) @ TSMC 45nm，1GHz

2.3 处理器家族——SHAKTI

SHAKTI[4]是印度理工学院的一个计划，目标是设计一系列适合不同应用环境的、基于 RISC-V 的开源处理器，以及一些 IP 核，以便搭建 SoC。这些处理器是 E-Class、C-Class、I-Class、M-Class、S-Class、H-Class、T-Class、N-Class，目前已经开源的是前三个，使用 Bluespec System Verilog 编写。

E-Class：32 位标量处理器，3 级流水线，支持 RISC-V 的 C(Compress)扩展，目标是超低功耗处理器。

C-Class：32 位或者 64 位标量处理器，3～8 级流水线，支持 MMU、具有容错功能、支持 RISC-V 的 C 扩展，目标也是超低功耗处理器。

I-Class：64 位、1～8 核，乱序执行处理器，共享 L2 Cache、支持双线程、SIMD/VPU，目标是通用处理器。

M-Class：I-Class 的增强版，增加了指令发射大小、支持四线程、最高支持 16 核，目标是通用处理器、低端服务器和移动应用。

S-Class：64 位、超标量多线程处理器，支持 L3 Cache、RapidIO、HMC(Hybrid Memory Cube)、向量处理单元，还有协处理器用于数据库访问、加密算法、机器学习，最高支持 64 核，目标是通用处理器、服务器。

H-Class：64 位、32～128 核、支持多线程、顺序或者乱序执行处理器，具有向量处理单元，目标是高性能计算。

T-Class：64 或者 128 位处理器，其中通过为存储器引入 Tag，从而增强其安全性。

N-Class：目标是通过自定义的扩展进行网络数据处理。

2.4 嵌入式应用处理器——ORCA

PicoRV32 是由 VectorBlox 公司设计的一款 32 位标量处理器，目标是应用于嵌入式领域，采用 VHDL 编写，实现了 RV32IM，也可以移除其中的 M 扩展，也就是移除乘法除法扩展，从而减少芯片占用资源，甚至可以移除与定时器有关的指令，从而仅仅实现 RV32E。当 ORCA 作为一个软核下载到 FPGA 上的时候，其资源占用与主频如表 5 所列。

表 5 ORCA 不同配置时的资源占用与主频[5]
(以 Altera's Cyclone IV 为目标 FPGA)

ORCA 配置	最高主频/MHz	LUT4 占用
RV32E	138	1700
RV32I	133.53	1900
RV32IM	97	2500
RV32IM(除法不使用硬件实现)	101	2100

2.5 其他开源处理器

(1) RI5CY

RI5CY 是由苏黎世联邦理工大学和波罗尼亚大学联

合设计的一款小巧的4级流水线开源处理器，实现了RV32IC以及RV32M中乘法指令mul，其目标是作为并行超低功耗处理器项目PULP(Parallel Ultra Low Power)的处理器核，所以RI5CY在RISC-V的基础上增加了许多扩展，包括硬件循环、乘累加、高级算术指令等。采用UMC的65 nm工艺进行流片，RI5CY主频达到654 MHz，动态功耗是17.5 μW/MHz[6]，采用System-Verilog编写。

(2) RIDECORE

RIDECORE (RIsc-v Dynamic Execution CORE)是由东京工业大学设计发布的一款超标量乱序执行处理器，实现了RV32IM，6级流水线分别是取指、译码、指令分配、发射、执行、提交，可以同时取两条指令、对两条指令译码、提交两条指令，采用的是Gshare分支预测机制。

(3) Hwacha

Hwacha是由UCB开发的一款向量处理器，UCB将Hwacha作为RISC-V的一个非标准扩展Xhwacha，已经以28 nm和45 nm的工艺流片多次，主频在1.5 GHz以上，目前还在研发中，正在修改OpenCL的编译器，以适合Hwacha，UCB计划以开源的形式发布其代码。

(4) f32c

f32c是由萨格勒布大学设计发布的32位、5级流水线、标量处理器，原本实现的是MIPS指令集，后来添加实现了RISC-V指令集，处理器包括分支预测、直接映射缓存，同时发布的还有SDRAM控制器、SRAM控制器、视频FrameBuffer、SPI控制器、UART、GPIO等IP，使用VHDL编写代码。使用f32c处理器核，萨格勒布大学发布了FPGArduino项目，该项目将一块FPGA开发板变为一个Arduino板，并且可以使用Arduino IDE进行程序编译下载。

(5) Z-scale/V-scale

Z-scale是UCB发布的针对嵌入式环境的32位、3级流水线、单发射标量处理器，实现了RV32IM，指令总线和数据总线都是AHB-Lite。Z-scale采用是Chisel编写代码，利用Rocket中的代码，仅增加了604行代码就实现了Z-scale。V-scale是Z-scale对应的Verilog版本。

(6) sodor

sodor是UCB发布的针对教学的32位开源处理器系列，采用Chisel编码实现，可以很容易的得到对应的C++模拟器。sodor系列有5种处理器，分别是单周期处理器、2级流水线处理器、3级流水线处理器、5级流水线处理器、可执行微码的处理器。

(7) PicoRV32

PicoRV32是由RISC-V开发者Clifford Wolf设计发布的一款大小经过优化的开源处理器，实现了RV32IMC，并且根据不同环境可配置为实现RV32E、RV32I、RV32IC、RV32IM、RV32IMC。内置一个可选择的中断控制器，其特点是小巧，在Xilinx7系列芯片上占用750～2 000个LUT，速度可以达到250～400 MHz。PicoRV32采用Verilog编写代码。

(8) Tom Thumb

Tom Thumb是由RISC-V开发者Maikmerten设计发布的一款32位、6级流水线开源处理器，实现了RV32I，目标是尽量减少FPGA的资源占用，在Cyclone IV系列FPGA上大约占用资源1200 LEs，采用VHDL编写代码。

(9) FlexPRET

FlexPRET[7]是由UCB设计发布的5级流水线、多线程处理器，目标是使用在实时嵌入式应用中，线程数量可配置为1～8。为了提高嵌入式处理器的资源利用率，每个硬件线程被标记为硬实时(hard real-time thread)或者软实时(soft real-time thread)，硬实时线程按照固定的频率被调度，如果当前没有硬实时线程可调度，再调度软实时线程。使用Chisel编写代码。

(10) YARVI

YARVI(Yet Another RISC-V Implementation)是由RISC-V开发者Tommy Thorn设计发布的一款简单的、32位开源处理器，实现了RV32I，使用Verilog作为开发语言。其出发点不在于性能，而是要能够清晰、准确地实现RV32I。

3 基于RISC-V的开源SoC研究现状

3.1 Rocket-Chip

UCB为了方便用户学习，同时也为了便于重复使用已设计好的硬件模块，在GitHub上建立了Rocket-Chip Generator的项目，其中包括了Chisel、GCC、Rocket处理器，以及围绕Rocket的一系列总线单元、外设、缓存等，并且采用了参数化的配置方法，从而可以方便地创建不同性能要求的基于Rocket处理器的SoC。采用Chisel编写，主要的子模块如下：

① Chisel：UCB设计的开源硬件编程语言。

② Hardfloat：参数可配置的、兼容IEEE 754-2008标准的浮点单元。

③ Riscv-tools：开发工具，包括GCC、Newlib，以及移植的Linux。

④ Rocket：Rocket处理器，包括L1 Cache。

⑤ Uncore：实现了需要与Rocket紧密连接的功能单元，比如L2 Cache、L1 Coherence Hub等。

⑥ Juntions：实现了不同协议的接口之间的转换。

⑦ Rocketchip：顶层模块，同时也实现了内部总线 TileLink 向外部总线 AXI 或者 AHB 的转换。

前文介绍的 BOOM、Z－scale 都可以通过配置 Rocket－Chip的不同参数得到。

3.2 LowRISC

LowRISC 是由剑桥大学为主的一些研发人员成立的非营利性组织，主要是设计发布基于 RISC－V 指令集的 64 位开源 SoC，其成员有树莓派的合作者，目标是希望将设计的 SoC 做成像树莓派那样价格便宜、功能丰富、拥有大量用户的开源硬件。LowRISC 发布的 SoC 的名称也是 LowRISC，是在 Rocket-Chip 的基础上改进开发的，采用 System Verilog 编写改进部分的代码。主要特点是：

① Tagged Memory。给每一个存储位置都增加了一个 Tag，目前是双字（64 位）对应一个 Tag（4 位），目的是防止控制流劫持攻击，同时也有其他的一些用处，比如：垃圾回收、设置 watchpoint 等。为了实现 Tagged Memory，LowRISC 为 RISC－V 增加了两条指令用来读写 Tag。2015 年 4 月发布的 0.1 版本中具有该功能。

② Untethered。早期的 Rocket-Chip 需要依赖于一个通用处理器的协助才能够启动，才能够访问串口、网口、SD 卡等外设，Untethered LowRISC 通过实现（Memory mapping I/O）、片上 NASTI interconnect 等功能，解决了上述问题。2015 年 11 月发布的 0.2 版本中具有该功能。

③ Trace Debugging。引入了 Open SoC Debug，支持 Trace Debugging，可以收集指令执行记录，便于离线或者在线分析。2016 年 7 月发布的 0.3 版本中具有该功能。

3.3 PULPino

PULPino 是苏黎世联邦理工大学和波罗尼亚大学联合发布的基于 RISC－V 的开源处理器，其处理器核 RI5CY 在前文已述，苏黎世联邦理工大学和波罗尼亚大学本来设计的项目是 PULP，这是一个多核 SoC 项目，考虑到这个项目太复杂，有许多 IP、自定义工具集，不方便开源，所以开发者决定先开源一个单核 SoC 项目，即 PULPino。PULPino 直接使用了 PULP 项目的许多 IP。

PULPino 具有一个 AXI 互连总线，另外还有一个 APB 总线，用来连接低速外设，比如：GPIO、UART、I^2C 控制器、SPI Master 控制器等。调试模块支持 Advanced Debug Unit。PULPino 包括一个 Boot ROM，其中可以写入 BootLoader，从而实现在启动的时候从外部 Flash 读入程序并执行。

3.4 RISC－V VHDL

RISC－V VHDL 是俄罗斯的 GNSS Sensor 公司发布的基于 Rocket 的开源 SoC，其前身是莫斯科物理技术学院的一个项目。该项目的处理器核直接用的是 Rocket，可以配置为只有 L1 Cache，也可以配置为包括 L2 Cache，在此基础上，提供了大量的 IP 核，采用类似 LEON3 的 GRLIB 库的方式，所有的 IP 核都是即插即用，RISC－V VHDL 提供了一个 AXI 总线，IP 核都挂载在该总线上。IP 核包括：UART、GPIO、中断控制器、以太网控制器，此外还支持 DSU（Debug Support Unit），均采用 VHDL 编写代码。RISC－V VHDL 中大多数 IP 核都是开源的，唯一商业的是 GNSSLIB，这是一个与定位导航有关的库，也是 RISC－V VHDL 的特色。

结 语

RISC－V 的发展十分迅速，除了前文已述的基于 RISC－V 的开源处理器、开源 SoC 大量涌现，还有很多的商用处理器也计划采用 RISC－V 指令集架构，比如 PULSAR[8]，采用的是一个异构多处理器结构，其中包括 2 个标量处理器 Rocket，还包括两个超标量乱序执行处理器 BOOM。RISC－V 的发展一方面得益于自身设计吸取了 RISC 接近 40 年的经验教训，使得架构设计更加合理；另一方面得益于日渐成熟的软件生态，UCB 提供了针对 RISC－V 的开源编译器 GCC、LLVM，还提供了开源仿真器 Spike、QEMU，社区还移植了 FreeBSD、Debian、Gentoo、Yocto、Genode 等操作系统。

在第 4 届 RISC－V 专题研讨会上宣布成立了 RISC－V 基金会，吸纳了众多实力雄厚的商业公司和知名研究机构，其中包括中国科学院。可以预见，RISC－V 即将进入一个快速发展的阶段，应该会在以下几个方面有突破进展：有若干成熟的、可商业化的、采用 RISC－V 架构的芯片问世，并得到大规模应用；性能逼近主流桌面处理器；主流处理器与采用 RISC－V 架构的开源处理器组成的异构系统；移植到 RISC－V 架构的操作系统更加稳定可靠；采用上百个简单 RISC－V 核的多核并行计算；计算机教学中采用 RISC－V 作为范例教学；调试功能得到进一步加强。对于国内而言，RISC－V 提供了一个很好的参考，可以用来实现自主可控的处理器。

参考文献

[1] A Waterman, Y Lee, DA Patterson, et al. The RISC－V Instruction Set Manual, Volume I: User-Level ISA[J]. Eecs Department, 2011, 7(9):475.

[2] Krste Asanovic, David Patterson. The Case for Open Instruction Sets[J]. MICROPROCESSOR report, 2014(8): 1－7.

[3] Celio, Christopher and Patterson, David A. and Asanovic, Krste, The Berkeley Out-of-Order Machine (BOOM): An Industry-Competitive, Synthesizable, Parameterized RISC-V Processor[R]. EECS Department, University of California, Berkeley, June 2015.

结　语

本文从多核并行、多核通信、多核同步、多核调试这几个方面介绍了如何搭建一款基于动芯基带芯片的多核仿真平台，并通过实际测试验证了多核仿真平台这几个功能的正确性。多核仿真平台的成功搭建将给动芯基带芯片的开发与研制带来极大的便利性。

参考文献

[1] 吴哲凯. 可配置多核仿真器的研究与实现[D]. 南京：东南大学，2012.

[2] 林明亮. 基于 SimpleScalar 的拥有存储与总线扩展能力的异构多核仿真器[D]. 上海：上海交通大学，2007.

[3] 罗汉青，梁利平，叶甜春. 一种多核指令集仿真器构建技术[J]. 计算机应用研究，2013，30(10)：3035－3037.

[4] 李德明，全盛程，叶进. 基于 SoC 的多核处理器并行仿真机制的研究[J]. 微电子学与计算机，2014，31(2)：23－26.

[5] 付琳，胡锦，梁利平. 指令集仿真器的关键技术[J]. 计算机应用，2015，35(5)：1421－1425.

[6] 苏雅丽. 一种多核处理器时钟精确并行仿真技术[J]. 赤峰学院学报：自然科学版，2015，31(3)：12－13.

[7] 喻之斌，金海，邹南海. 计算机体系结构软件模拟技术[J]. 软件学报，2008，19(4)：1051－1068.

[8] 高翔，张福新，等. 基于龙芯 CPU 的多核全系统仿真器 SimOS－Goodson[J]. 软件学报，2007，18(4)：1047－1055.

[9] 戴鹏，魏来，王明江，等. 一种多核处理系统通信机制的仿真模型[J]. 微电子学与计算机，2012，29(6)：1－6.

（责任编辑：薛士然　收稿日期：2016-11-18）

[4] N. Gala, A. Menon, R. Bodduna, G. S. Madhusudan, V. Kamakoti, SHAKTI Processors: An Open-Source Hardware Initiative[C]//29th International Conference on VLSI Design and 2016 15th International Conference on Embedded Systems (VLSID), Kolkata, India, 2016.

[5] PULPino User Manul[EB/OL]. [2016－09]. http://www.pulp-platform.org/documentation/.

[6] Michael Zimmer, David Broman, Chris Shaver, et al. FlexPRET: A Processor Platform for Mixed-Criticality Systems [C]//Proceedings of the 20th IEEE Real-Time and Embedded Technology and Application Symposium (RTAS), April, 2014.

[7] PEYRET Thomas, VENTROUX Nicolas, OLIVIER Thomas. HETEROGENEOUS MULTICORE BASED ON RISC-V PROCESSORS AND FD-SOI SILICON PLATFORM[C]//4th RISC-V Workshop Proceedings, 2016.

雷思磊（工程师），主要研究方向为处理器架构、嵌入式处理器应用等。

（责任编辑：杨迪娜　收稿日期：2016-09-12）

连科技工业级物联网构筑全球化技术与生态平台

自连电子科技（上海）有限公司日前在深圳召开主题为“物语·自连”的新产品发布会，并率先推出集无线控制器、物联网开发平台、物联网中间件、云平台接入技术和智能 APP 控制于一体的完整物联网解决方案。该解决方案包括了自连科技的全系列无线控制器 AiControllers 和其拥有自主知识产权的三大物联网技术平台 AiDK、AiSDK 和 AiDMS，具备快速开发、高可靠、低延时等特性，可应用于医疗电子、工业设备、精密仪器仪表、音乐器械和智能家居等多个领域。自连科技本次推出的完整物联网解决方案通过了多项国际认证，将助力物联网企业面向全球市场推出支持其相关行业加速实现智能化、信息化和数字化的定制物联网方案，共同实现“借.平台东风 现.端云速连”的美好愿景。

与全球领先伙伴合作支持中国物联网应用系统走向全球

自连科技的物联网解决方案得到了全球众多领先合作伙伴的支持，为中国的医疗设备和工业制造等行业走向全球提供了强有力的技术支撑。其中，赛普拉斯（Cypress）提供了一系列高品质的 Wi－Fi/BT/BLE 芯片和完整的 WICED 无线连接解决方案，支持多种无线协议标准和硬件产品组合，助力自连科技开发出了高性能，高可靠的无线连接产品；而微软（Microsoft）则更是在物联网的端云对接方面与自连科技展开了深入的合作。基于微软技术由世纪互联运营的 Microsoft Azure 云服务侧重于云端最先进的物联网 PAAS 服务和大数据分析，而自连科技的 AiDMS 侧重于物联网数据汇聚、消息封装、协议转换等。二者的配合相辅相成，客户可以轻松实现端云互通而将其重点放在业务系统的开发上。目前，微软和自连科技的合作案例已有 3 例，应用包括工业 UPS、智能制造和工业网关等。未来双方还会继续合作，扩展人工智能（AI）、商业智能分析（BI）等在物联网领域中的应用。

用技术创新协助客户实现其物联网生态的差异化

当前，各个产业都不再希望简单地加载物联网技术，而是需要实现符合自己产业特点的“＋物联网”，从而构建或优化自己的生态系统。自连科技的完整物联网解决方案是符合医疗、工业自动化领域需求的差异化、创新性解决方案，可以助力医疗设备企业实现产品的优化和差异化，帮助传统工业企业构建物联网系统来优化运营能力，并提升生产、制造能力。近两年，自连科技与国内外多家器械、运营、服务类企业建立了合作，其中用于医疗健康、远程工业的无线控制器销售量实现了大幅增长。