赛灵思 中国通讯
Xilinx News

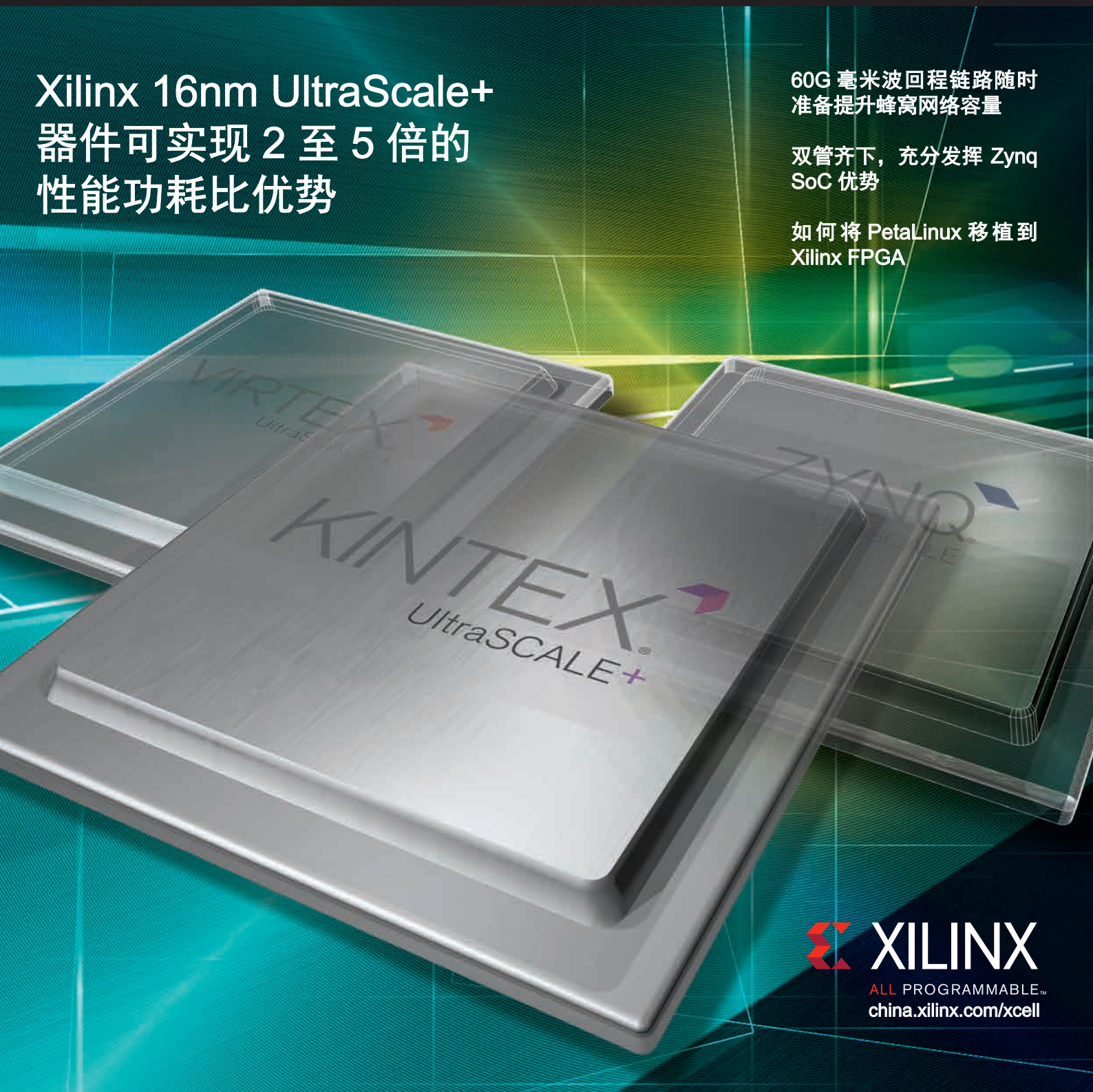
第五十五期 2015年 春季刊 Issue 55 Spring 2015

Xilinx 16nm UltraScale+ 器件可实现 2 至 5 倍的性能功耗比优势

60G 毫米波回程链路随时准备提升蜂窝网络容量

双管齐下，充分发挥 Zynq SoC 优势

如何将 PetaLinux 移植到 Xilinx FPGA



XILINX
ALL PROGRAMMABLE™
china.xilinx.com/xcell

MicroZed
FLEXIBILITY REDEFINED
MICROZED.ORG

AVNET
安富利电子元件

新品速递！

欢迎选用本系列产品！

- √ 低成本开发工具
- √ 现成的SOM解决方案
- √ 支持Linux的开箱即用
- √ 从原型机转向规模化生产非常容易

ZYNQ

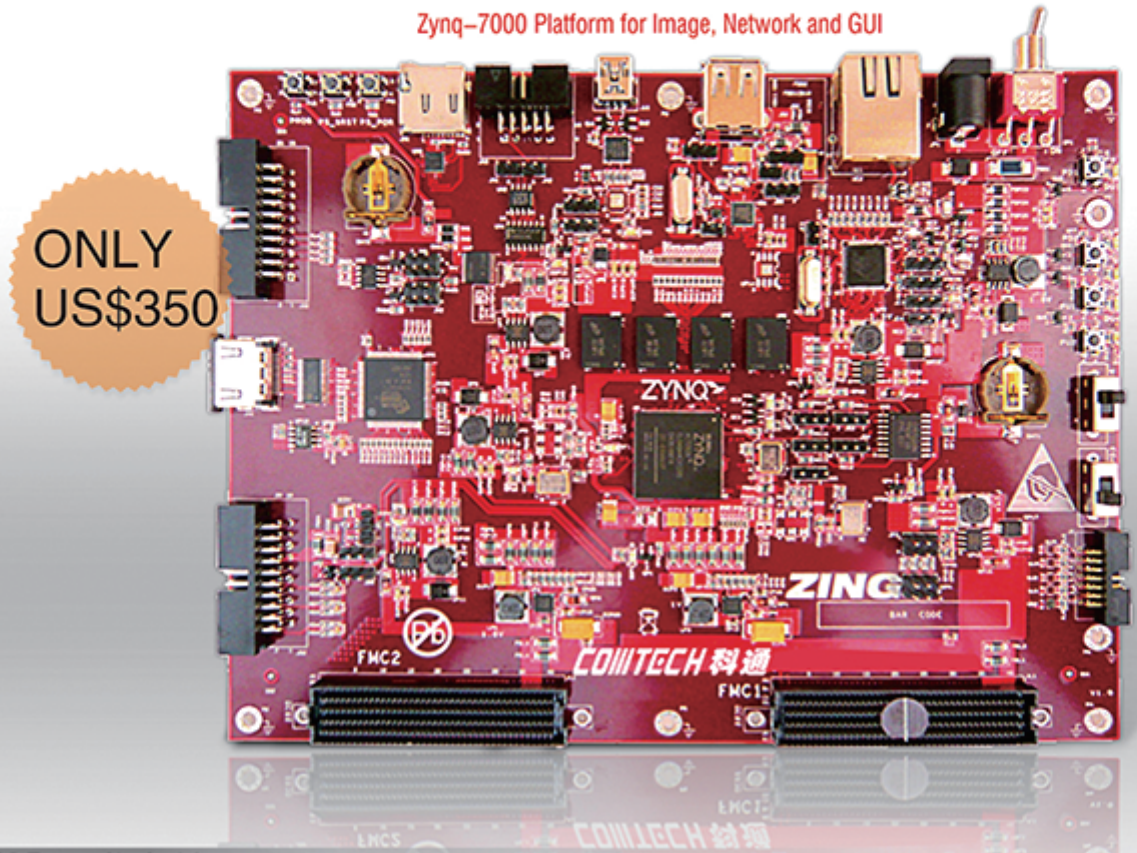
www.microzed.org

MicroZed™ 是一种基于Xilinx Zynq®-7000 全可编程系统级芯片(SoC) 的低成本开发电路板。它的独特设计让其既可以被用作单独的基本SoC实验用的评估板，也可以与基板组合，用作嵌入式系统化模块(SOM)。这种独立工作/SOM相结合的方法可以很快地将设计构思从概念转到投产，使MicroZed 成为基于SoC应用的理想平台。 MicroZed以一个网上社区平台为依托，用户可以在那里下载套件的相关文档和参考设计，也可以与其他进行Zynq设计的工程师合作。

DESIGNED BY AVNET

ZING Board 是基于 Xilinx Zynq™-7000 SoC 的开发套件，给高性能系统设计带来帮助，加速设计者创新产品的诞生。

Zynq–7000 Platform for Image, Network and GUI



ZING 开发套件

- Zing 评估开发板
  包含 XC7Z020CLG484-1C
- 参考设计、设计范例以及演示文件
- 开发板设计文档
- 包含所有软件和参考设计、演示以及文档
  帮助您快速入门
- 技术文档
- 电缆和电源
- TF卡

融合软/硬件协同编程，开启定制化SoC设计时代 SNOWLeo

主要特性

- Xilinx ZYNQ 7010/7020 AP SoC
- 双核ARM Cortex–A9 主频高达800MHz
- 28nm FPGA百万门可编程逻辑
- 512MB DDR3 SDRAM
- 千兆以太网/USB 2.0 OTG/USB UART
- HDMI 高清输出，支持3D 1080p显示
- TF卡和Nand Flash 双重启动模式
- 高速扩展IO连接器

操作系统支持 Linux POWERED · ANDROID · ubuntu

ZingSoM：体积最小的ZYNQ核心模块

专为OEM和小批量高附加值产品客户量身定制的ZYNQ最小系统模块，集成了ZYNQ系统所需的全部组件，包括DDR3，FLASH，GigaE和USB PHY芯片。预装Linux/Android 操作系统和常用IP，开箱即用。

▶ZingSoM 已经开始现货供应

丰富的FMC子卡支持行业应用
提供IP定制设计服务

更多详细信息请联系：xilinx_enquiry@comtech.com.cn

COMTECH 科通

www.comtech.com.cn

Xcell journal

出版商 Mike Santarini
mike.santarini@xilinx.com
408-626-5981

编辑 Jacqueline Damian

艺术总监 Scott Blair

设计/制作 Teie, Gelwicks & Associates
1-800-493-5551

广告销售 Dan Teie
1-800-493-5551
xcelladsales@aol.com

国际 Melissa Zhang, Asia Pacific
melissa.zhang@xilinx.com

Christelle Moraga, Europe/
Middle East/Africa
christelle.moraga@xilinx.com

Tomoko Suto, Japan
tomoko@xilinx.com

订购往期期刊 1-800-493-5551

Xcell PUBLICATIONS
www.xilinx.com/xcell/

Xilinx, Inc.
2100 Logic Drive
San Jose, CA 95124-3400
Phone: 408-559-7778
FAX: 408-879-4780
www.xilinx.com/xcell/

准备用Xilinx器件实现更多创新吧

自赛灵思于2011年开始发货业界首款全可编程 SoC以来，用户就一直在越来越多的终端市场上打造多种多样的创新产品。汽车、工业、科学、有线和无线通信、测量测试、广播以及消费电子等所有这些市场都已推出或即将推出采用Zynq® SoC的创新产品。如果过去几年您一直在阅读《赛灵思中国通讯》或者访问过新的“Xilinx午后加油站”博客，您或许已经注意这种器件的推广在不断深入。

当然，所有Zynq SoC相关文章的一个共同主题就是该器件通过在单一器件中集成ARM® 双核Cortex™-A9 MPCore处理器与7系列FPGA，并将它们互联在一起，可以实现令人惊叹的系统性能。通过3,000多个互联将处理系统和可编程逻辑连接在一起，Zynq SoC实现了双芯片ASSP/ASIC + FPGA无法企及的高性能。任何分立FPGA和ASSP外都不会有足够的I/O来完成所需的工作。高集成度还带来了另一大优势，就是降低功耗要求（也减少了材料清单（BOM）成本），因为双芯片换成单芯片后，系统所需的电源电路更少。实践证明，过去4年来，Zynq SoC荣膺的一系列由全球一级行业期刊颁发的创新奖项，确属实至名归。

《赛灵思中国通讯》怀着激动地心情终于能向您透露赛灵思下一代全可编程SoC的片上细节了！Zynq UltraScale+™ MPSoC预计将于明年早些时候发货。我建议您阅读本期封面专题，进一步了解有关该器件以及赛灵思新发布的其它16nm UltraScale+产品组合的详细信息。通过吸收最初Zynq SoC的经验与教训，充分利用用户反馈意见并将有关观点整合到产品发展规划中，赛灵思打造出了全可编程MPSoC，其系统集成度和系统性能功耗比相较已经相当惊艳的第一代Zynq SoC而言，实现了几何级的提升。

事实上，本期封面专题将告诉您，全新UltraScale+产品组合中的FPGA、3D IC和MPSoC相对于前代系统而言性能功耗比至少提升一倍，这要归功于其采用了台积公司的16nm FFT+工艺。更多性能功耗比优势得益于UltraRAM这种更大容量的新型存储器（赛灵思在大多数产品中采用这种存储器），以及全新的系统级互联技术SmartConnect。

迄今为止，利用Zynq UltraScale+ MPSoC，可实现最大的性能功耗比优势，这款大型全可编程SoC在单个芯片上集成了64位四核APU、双核RPU、图形处理器和一系列外设、安全特性和电源管理功能。Zynq MPSoC系统的性能功耗比相对28nm Zynq SoC系统提升了5倍。

您已经用ZynqSoC打造了一些令人称奇的系统。我期待着看到您用ZynqUltraScale+MPSoC带来更精彩的设计。随着赛灵思开始推出上述这些卓越产品，我希望您将继续在《赛灵思中国通讯》上发表文章，与同事分享您的设计经历。

Mike Santarini
发行人

谨以本期献给《赛灵思中国通讯》的长期广告创意总监Dan Teie，他不幸于1月逝世。Dan是一位运动健将、冒险家、战士、绅士，一个伟大的人，他深爱着他的家庭和朋友，对生活怀抱着无限的热情。我们怀念您，Dan。

领 先 一 代

业界首款
ASIC级架构FPGA

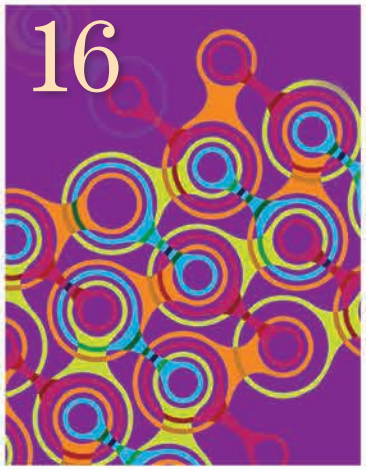
现已提供详细器件选型表、文档、设计工具和方法

- 消除DSP和包处理的瓶颈
- 显著提升定点及浮点运算性能与效率
- 集成二代3D IC系统及全新的3D IC宽存储器优化接口
- 海量I/O与存储器带宽，大幅降低时延
- 大幅降低功耗

XILINX
ALL PROGRAMMABLE™

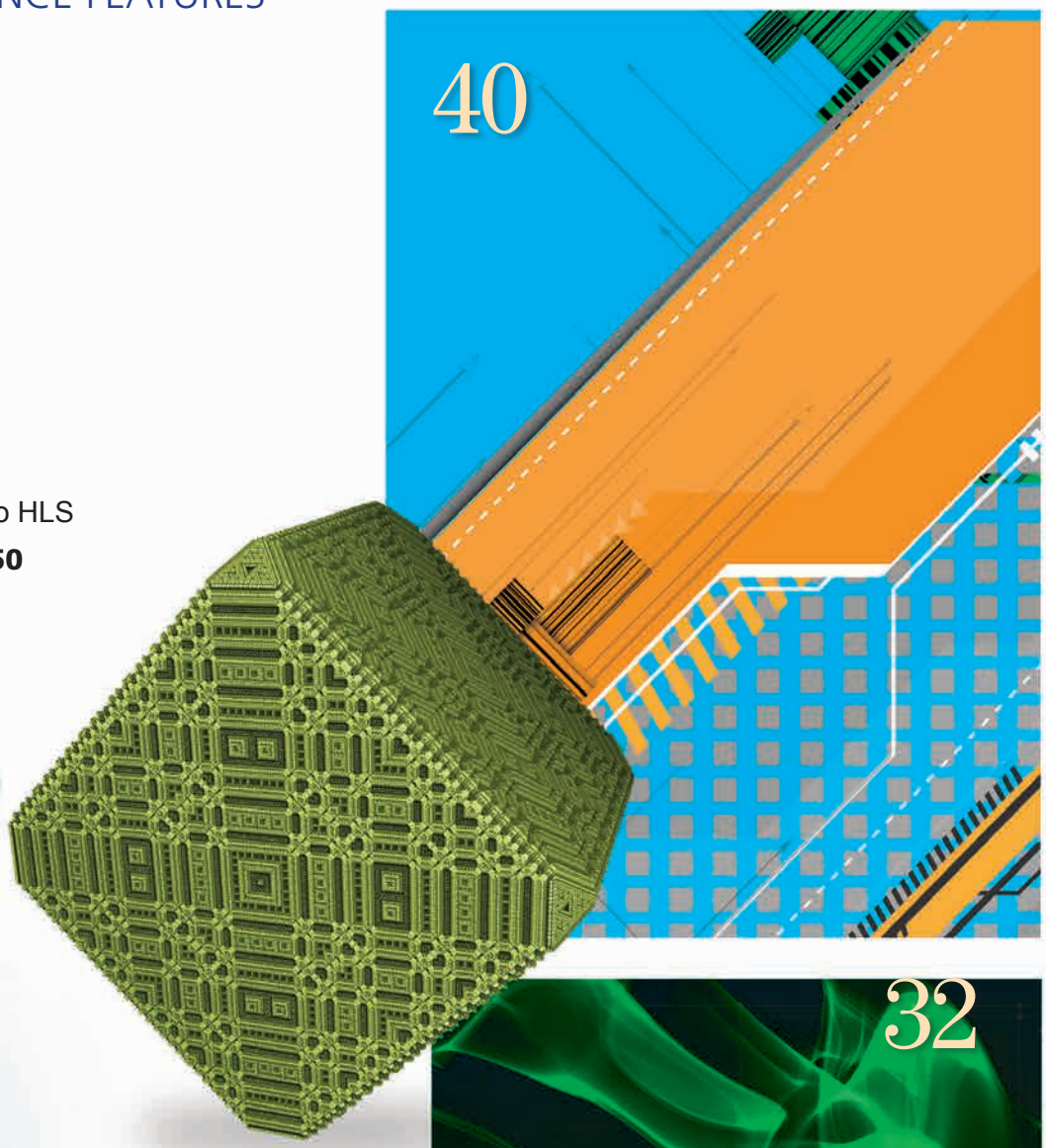
视点

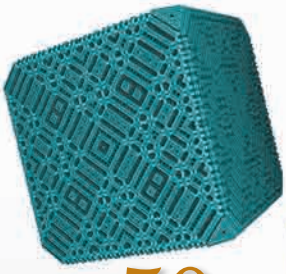
XCELLENCE BY DESIGN APPLICATION FEATURES



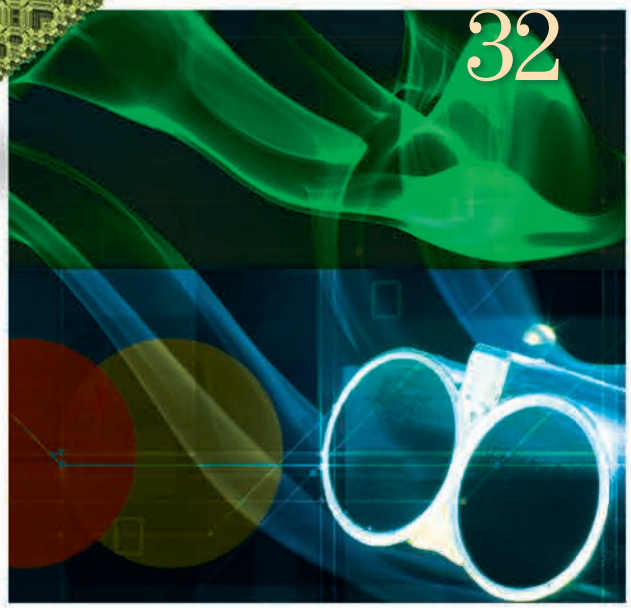
封面专题

THE XILINX XPERIENCE FEATURES





50



XTRA READING

APEX® AWARDS FOR PUBLICATION EXCELLENCE
Excellence in Magazine & Journal Writing
2010, 2011

APEX® AWARDS FOR PUBLICATION EXCELLENCE
Excellence in Magazine & Journal Design
2010, 2011, 2012

Xilinx 16nm UltraScale+器件实现2至5倍的性能功耗比优势

作者：Mike Santarini
赛灵思公司赛灵思杂志发行人
mike.santarini@xilinx.com

台积公司的 16nm FinFET 工艺与赛灵思最新 UltraRAM 和 SmartConnect 技术相结合，使赛灵思能够继续为市场提供超越摩尔定律的价值优势。

B

赛灵思凭借其 28nm 7 系列全可编程系列以及率先上市的 20nm UltraScale ™系列，获得了领先竞争对手整整一代优势，在此基础上，赛灵思刚刚又推出了其 16nm UltraScale+ ™系列器件。客户采用该器件系列构建的系统相比采用赛灵思 28nm 器件所设计的类似系统的性能功耗比可提升 2 至 5 倍。这些性能功耗比优势主要取决于三大方面：采用台积电公司 16FF+（即 16nm FinFET Plus）工艺的器件实现方案、赛灵思的片上 UltraRAM 存储器以及 SmartConnect 创新型系统级互联优化技术。

此外，赛灵思还推出了其第二代 Zynq® 全可编程 SoC。Zynq UltraScale 多处理 SoC (MPSoC) 在单个器件中完美集成了四核 64 位 ARM® Cortex ™ -A53 应用处理器、32 位 ARM Cortex-R5 实时处理器、ARM Mali-400MP 图形处理器、16nm FPGA 逻辑（带 UltraRAM）、众多外设、安全性与可靠性特性、以及创新型电源控制技术。该新型 Zynq UltraScale+ MPSoC 为用户提供了系统创建所需的一切，而且利用其打造出来的系统相比采用 28nm Zynq SoC 所设计的系统的性能功耗比提升 5 倍。

FINFET进一步扩展 ULTRASCALE系列，使其具有额外的节点价值优势

赛灵思公司芯片产品管理与营销高级总监 Dave Myron 指出：“采用 16nm UltraScale+ 系列，我们能够创建出比摩尔定律通常提供给用户的更高的

额外节点价值优势。我们能满足 LTE Advanced 与早期 5G 无线、Tb 级有线通信、汽车高级驾驶员辅助系统以及工业物联网应用等各种下一代应用需求。UltraScale+ 系列使用户能够实现更大的创新，同时在各自的市场中保持领先竞争对手。”

凭借其 UltraScale 系列产品，赛灵思能够同时通过两个工艺节点提供器件，即台积公司的 20nm 平面工艺（已经发货）和现在台积公司的 16FF+ 工艺（赛灵思预计将于 2015 年第四季度开始发货）。赛灵思将推出 16nm UltraScale+ 系列的 Virtex® FPGA 与 3D IC、Kintex® FPGA 以及新型 Zynq UltraScale+ MPSoC。

赛灵思公司新产品推出与解决方案市场营销总监 Mark Moran 表示，赛灵思决定于 2013 年开始推出其 20nm UltraScale 系列，而不是等台积公司的 16FF+ 工艺问世后才发布。这是因为在一些应用领域，早在一年半就迫切需要 20nm 器件 —— 其比 28nm 具有更高的性能和容量。

Moran 表示：“我们的整个产品系列在设计时充分考虑到市场需求。采用 20nm UltraScale 架构的器件的功能更适用于那些无需 UltraScale+ 提供的额外性能功耗比优势的市场和最终应用中的新一代产品。既然知道 16nm 紧跟其后，所以我先构建了 20nm FinFET。同时我们在 20nm 上进了大量的架构修改（我们知道这是 16nm 的基础），可以根据市场需要提高性能和价值水平。我们有客户已经着手在我们目前提供的 20nm 器件上进行开发，这样只要 16nm Ultra-Scale+ 器件一问世，他们就可以快速进行设计移植，进而加速设计上市进程。”

Myron 补充说，众多 Virtex UltraScale+ 器件会与 20nm Virtex Ultra-Scale 器件实现引脚兼容，这样，对需要额外性能功耗比优势的设计来说易于升级。

Myron 说：“从工具角度来说，20nm UltraScale 和 16nm UltraScale+ 器件看起来几乎一样。因此使用 16nm UltraScale+ 器件还有一大优势，那就是提升性能功耗比使其很容易达到性能和功耗目标要求。”

Myron 说 UltraScale+ FPGA 以及 3D IC 相比 28nm 7 系列 FPGA，性能功耗比提升 2 倍。同时，Zynq UltraScale+ MPSoC 凭借其额外的集成异构处理功能，相比采用 28nm Zynq SoC 构建的类似系统，性能功耗比提升 5 倍（如图 1 所示）。

源于台积公司16FF+工艺的性能功耗比优势

仅通过向 16nm FinFET 的工艺移植，赛灵思已推出了比 28nm 7 系列器件的性能功耗比高出 2 倍的器件。

图1 – 赛灵思16nm UltraScale+ FPGA和Zynq UltraScale+ MPSoC可为设计团队提供额外的节点价值优势。

Myron 指出：“台积公司的 16FF+ 是一种极其高效的工艺技术，这是因为其基本消除了此前采用平面晶体管实现的芯片工艺相关的晶体管电源泄漏情况。此外，我们还与台积公司通力合作，共同优化 UltraScale+ 器件，以充分利用该新工艺技术。至少（仅从该新工艺技术的创新角度来说），UltraScale+ 设计相比采用 28nm 7 系列器件实现的设计，性能功耗比提升两倍以上。

如需了解有关赛灵思 20nm UltraScale 架构，以及 FinFET 相比平面晶体管工艺的优势的详细说明，敬请访问：《赛灵思中国通讯第 49 期》。

在 UltraScale+ 系列中，赛灵思还将提供业界首款 3D-on-3D 器件——其采用台积公司 16FF+ 3D 晶体管技术实现的第三代堆叠硅片互联 3D IC。

Myron 指出，屡获殊荣的 7 系列 3D IC 通过在单个集成芯片上提供多个芯片，突破了摩尔定律的性能和容量极限。

Myron 指出：“凭借我们的同质 3D IC，我们能够突破摩尔定律的容量极限，从而可提供容量是 28nm 最大型单芯片 FPGA 容量 2 倍的器件。然后利用我们的首款异构器件，我们能够将 FPGA 芯片与高速收发器芯片组合在一起，提供 28nm 单芯片器件无法实现的高系统性能与带宽。利用 UltraScale+ 3D IC，我们将继续提供超越摩尔定律极限的高容量与性能。”

源于ULTRARAM的性能功耗比优势

Myron 说通过采用最新大型片上存储器 UltraRAM，众多 UltraScale+ 设计相对 28nm 将获得更多的性能功耗比提升。赛灵思将在大部分 UltraScale+ 器件中新增 UltraRAM。

Myron 指出：“从根本上来说，片上存储器（如 LUT RAM 或分布式 RAM 和 Block RAM）和片外存储器（DDR 或片外 SRAM 等）之间的差距越来越大。有很多处理器密集型应用需要不同类型存储器。尤其是当您设计更大型更复杂的设计时，就更需要较快速的片上存储器。Block RAM 太细太少。而如果您将存储器放在片外，不仅会增加功耗，让 I/O 变得复杂，而且还会增加材料清单（BOM）成本。

这就是赛灵思开发 UltraRAM 的原因。Myron 指出：“我们所做的就是增加片上存储器分层结构的层数，以及能够在设计中轻松实现大型存储器模块。我们不仅帮助设计人员轻松放置恰当尺寸的片上存储器，而且时序也有保障。”

通过 LUT 或分布式 RAM，设计人员可以添加 1b 和 kb 级大小的 RAM，而 BRAM 可让他们添加 10 Mb 大小的存储器模块。UltraRAM 允许采用 UltraScale+ 器件的设计人员用 100Mb 级的存储器块实现片上 SRAM（如图 2 所示）。这样做，设计人员只需较少量的片外 RAM (SRAM、RLDRAM 和 TCAM) 就能够打造出性能 / 能效更高的系统。同时还会降低材料清单（BOM）成本。最大型的 UltraScale+ 器件 VU13P 具有 432 Mb 的 UltraRAM。

源于SmartConnect的性能功耗比优势

另一项新技术 SmartConnect，可进一

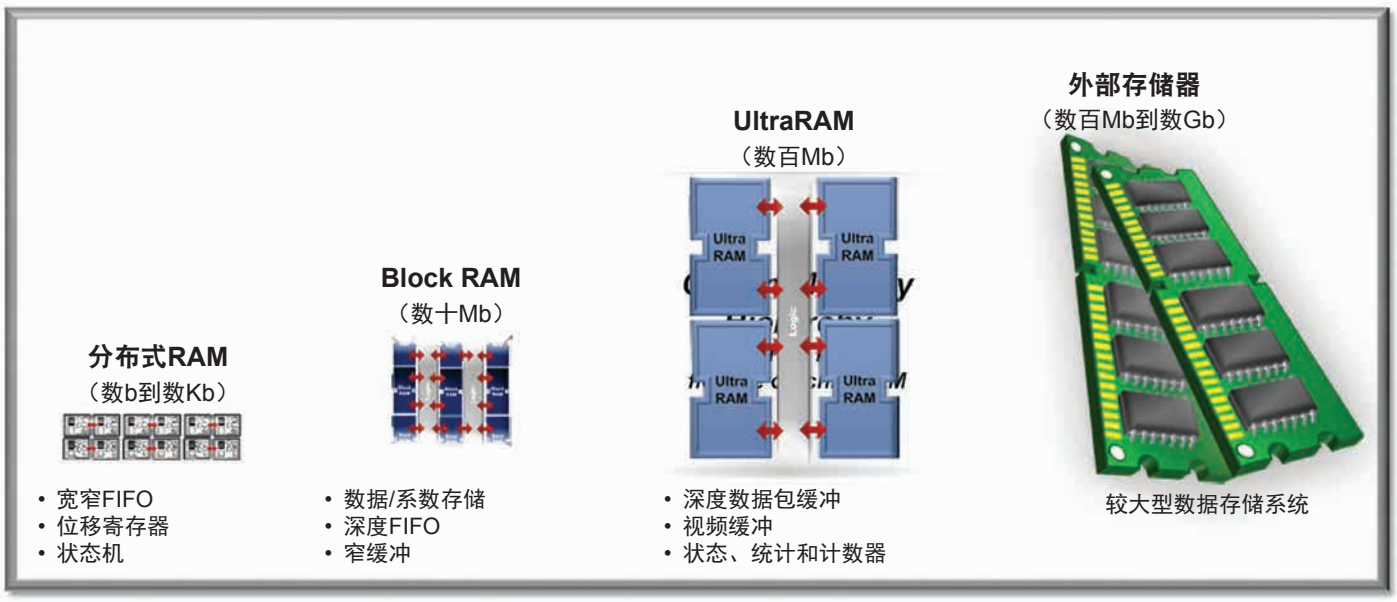


图2 – UltraRAM可填补片上存储器和片外存储器之间的存储器空白，从而使设计人员能够利用较大型的本地存储器模块创建性能更高、功耗更低的系统。

步提升 UltraScale+ 设计的性能功耗比优势。

Myron 说：“SmartConnect 是工具和硬件协同优化的结晶，也是一种智能方法，即便设计越来越复杂，也可轻松实现。”

传统上，当工程师在设计中填充的 IP 模块越多，开销（功耗和占用面积需求）就越大。Myron 说借助 SmartConnect，赛灵思已向 Vivado® 设计套件增加了一些优化功能，从而可以从系统级层面考虑整个设计。SmartConnect 具有最有效的互联拓扑结构，可实现最小的占位面积和最高的性能，从而充分发挥 AXI 互联的一些最新增强功能以及 16nm UltraScale+ 芯片的优势。

Myron 指出：“16nm UltraScale+ 器件在这个更高的协议层而不仅仅是在路由层上具有更高的效率。这意味着在 16nm FinFET 优势的基础上进一步提高性能功耗比优势。”

图 3 展示了一个真实的设计，其含有 8 个视频处理引擎，所有这些引擎均与处理器和储存器相连。Myron 说：也许奇怪，像这样的一个真实世界的设计，互连逻辑竟然差不多占用了设计总面积的一半。这不仅影响功耗，而且还会限制频率。而 SmartConnect 可以自动重组互连模块并在不影响性能的情况下将功耗降低 20%。

16nm ULTRASCALE FPGA标准测试

举例说明 FPGA 设计方案的性能功耗比优势，在 28nm Virtex-7 FPGA 中实现的 48 端口无线 CPRI 压缩与基带硬件加速器的功耗为 56W（如图 4）。在同一性能水平下运行的同一设计实现在 16nm Virtex UltraScale+ FPGA 中，功耗仅为 27 W，相比 28nm 设计功耗降低了 55%，性能功耗比提升了 2.1 倍。加上 UltraRAM 和 SmartConnect 提供的额外性能功耗比优势，实现在 Virtex UltraScale+ 中的设计相比 28 nm Virtex-7 FPGA 实现方案，性能功耗比提升了 2.7 倍，功耗降低了 63%。

同样，在 FPGA 功耗预算为 15W 的图像处理 PCI 模块中，28 nm Virtex-7 可实现每秒 525 次操作的性能。相比之下，实现在 16 nm UltraScale 中的同一设计则可实现每秒 1255 次操作的高性能，性能功耗比提升了 2.4 倍。加上 UltraRAM 和 SmartConnect 提供的额外性能功耗比优势，Virtex UltraScale + 实现方案相

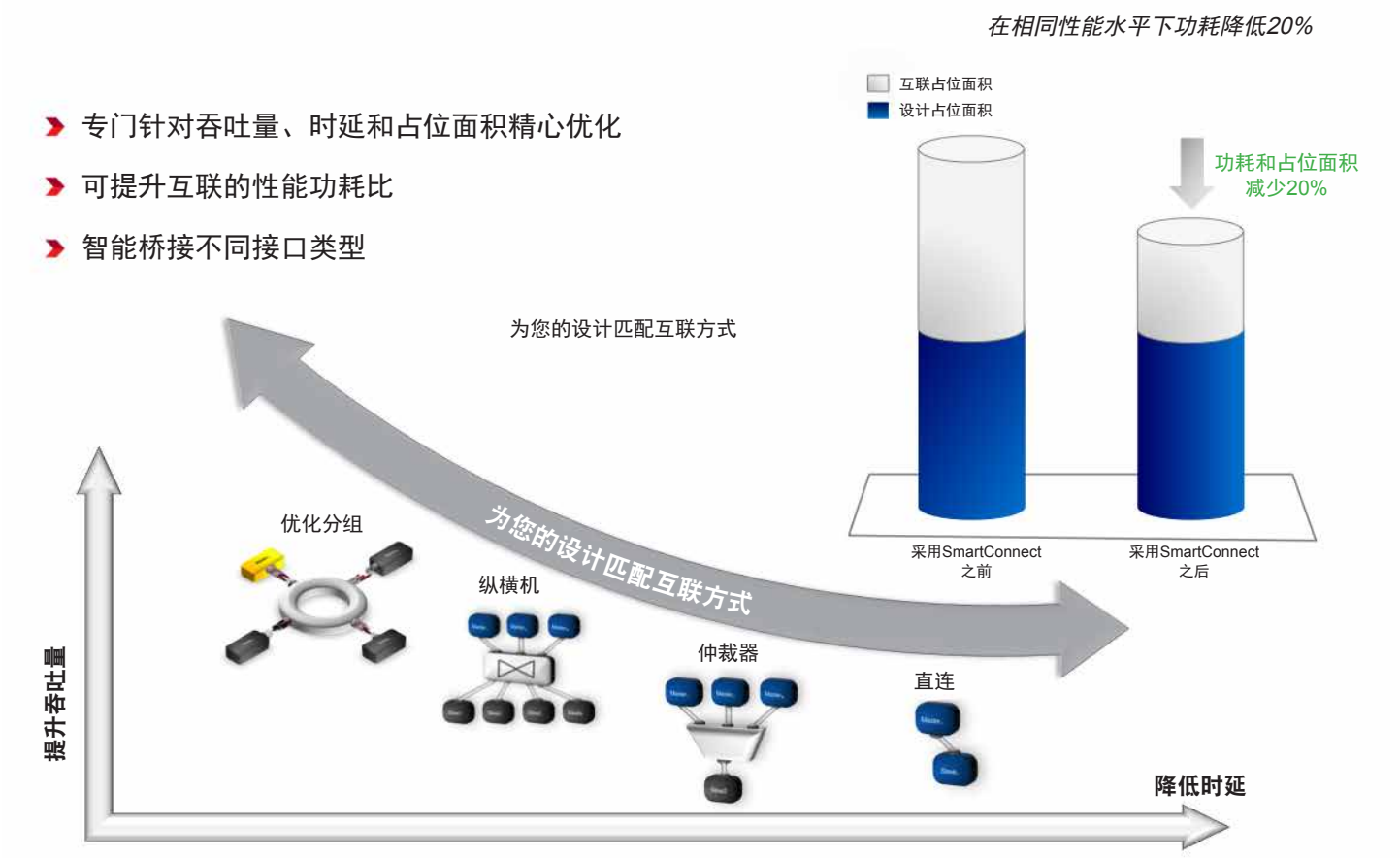


图3 – SmartConnect技术将互联所占用的面积削减达20%，这样在相同性能水平下，功耗可降低20%。

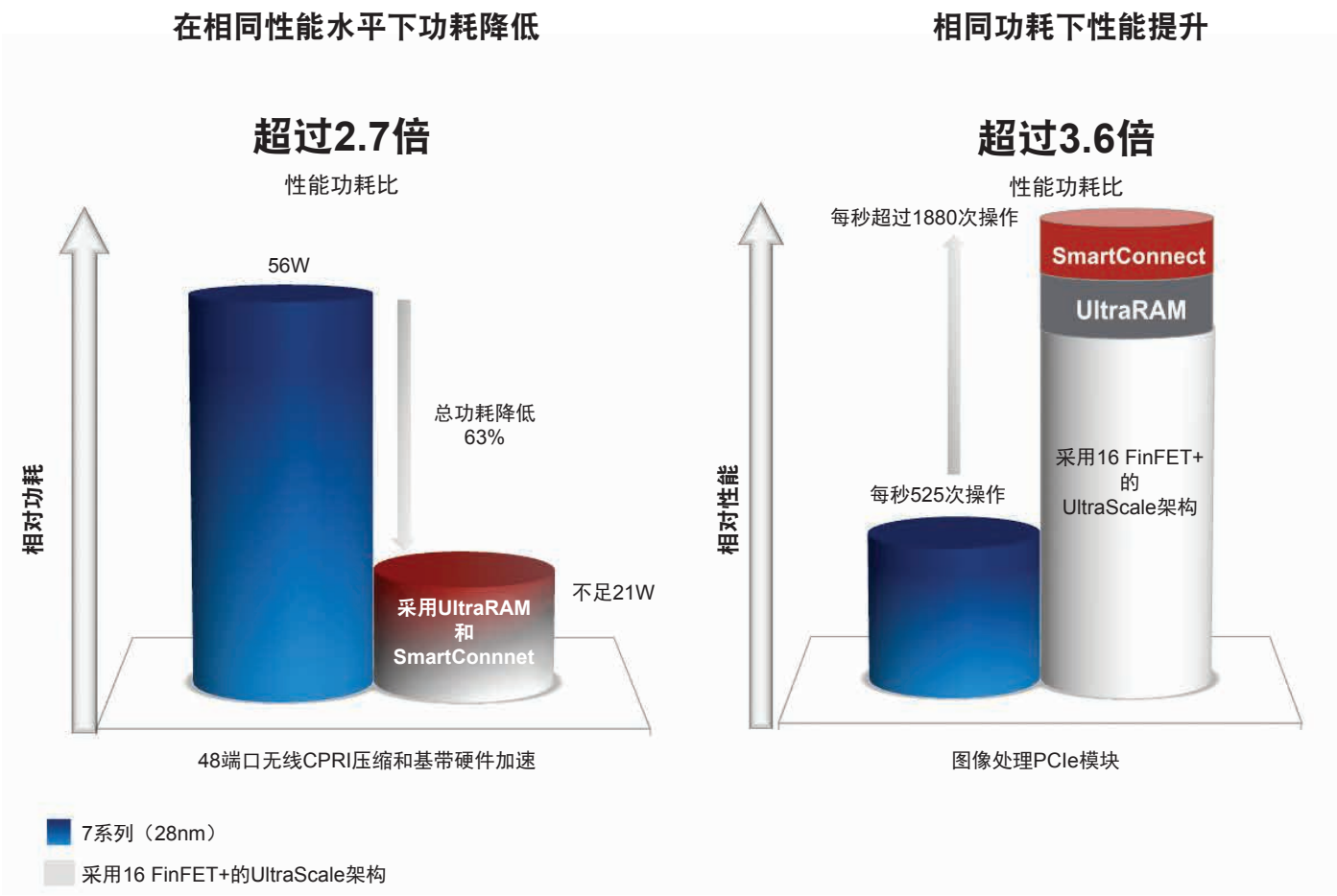


图4 – 16nm UltraScale+可为那些设法在相同功耗预算范围内更快速实现设计以及试图在相同性能水平下大幅降低功耗的设计人员保持其显著的性能功耗比优势

比 28 nm Virtex-7 FPGA 实现方案，性能功耗飙升 3.6 倍。

ZYNQ ULTRASCALE MPSOC可提供超过5倍的性能功耗比优势

尽管赛灵思原本可以采用台积公司 20 nm 工艺实现其第二代全可编程 SoC，但公司仍会选择等待采用台积公司的 16 nm FinFET 工艺来实现该器件。该器件的异构多处理特性集结合 16nm UltraScale 架构的性能功耗比优势，可以将 16nm Zynq UltraScale+ MPSoC 打造成更高效的中央处理系统控制器。该器件可提供超过 28 nm Zynq SoC 5 倍的性能。

去年，赛灵思针对 UltraScale MPSoC 架构推出了其“为合适任务提供合适引擎”的使用模型，但保留了有关 Zynq UltraScale+ MPSoC 器件应有的特定内核的细节。目前公司正发布全特性集 Zynq UltraScale+ MPSoC（如图 5 所示）。

当然，初始 28nm Zynq SoC 的最大增值是在单个器件中完美集成了 ARM 处理系统和可编程逻辑。Zynq SoC 的处理系统（PS）和可编程逻辑（PL）模块通过超过 3000 多个互联（峰值带宽运行速率约为 84 Gbps）连接在一起。PS 和 PL 之间的紧密相连所提供的吞吐量和性能不是一个包含 FPGA 和独立 ASSP 的双芯片系统架构能简简单单实现的。

目前借助 16nm UltraScale+ MPSoC，赛灵思显著提高了处理系统和可编程逻辑之间的性能，为器件提供了超过 6,000 次互联（峰值带宽运行速率为 500Gbps）。赛灵思公司全可编程 SoC 产品市场营销与管理总监 Barrie Mullins 指出：“这使得 Zynq UltraScale+ MPSoC 处理系统与逻辑系统之间的连接速率比采用 28nm Zynq SoC 可能实现的连接速率快 6 倍。而且双芯片（ASSP +FPGA）架构的系统性能远远落后于此。”

Mullins 说 Zynq UltraScale+ MPSoC 的核心是 64 位四核 ARM Cortex-A53 处理器，其可提供 2 倍于 28nm Zynq SoC 的双核 Cortex-A9 处理系统的性能。应用处理系统具有硬件虚拟化和非对称处理功能，可全面支持 ARM 的 TrustZone® 套件的安全

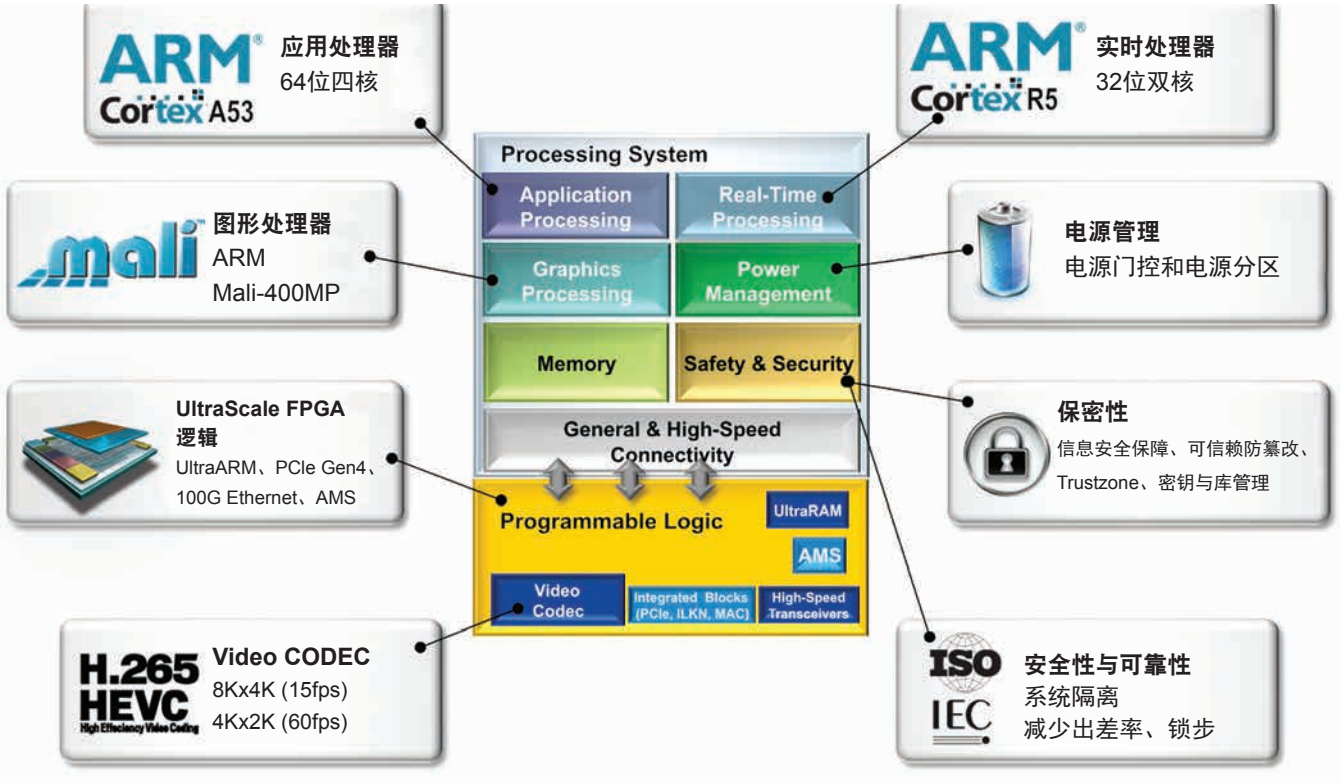


图5 – 16nm Zynq UltraScale+ MPSoC采用了一组丰富的处理引擎，设计团队能够为各项任务量身定制处理引擎，从而实现无与伦比的系统性能，进而显著提升其系统价值。

特性。

赛灵思还为 Zynq UltraScale+ MPSoC 提供了一个双核 ARM Cortex-R5 实时处理子系统，可帮助用户向其系统添加确定性操作。实时处理器可确保为需要最高级别吞吐量、安全性和可靠性的应用提供即时系统响应。

为进一步提升处理性能，Zynq UltraScale+ MPSoC 还内置了一系列的专用图形引擎。ARM Mali ™ -400MP 专用图形加速内核可帮助主 CPU 分担图形密集型任务。为协助 GPU，赛灵思向用于视频压缩 / 解压缩（支持 8Kx4K (15fps) 和 4Kx2K (60fps) 的 H.265 视频标准）的可编程逻辑块添加了一个硬化的视频编解码器内核。DisplayPort 源内核可帮助用户加速视频数据分组，同时还避免其系统使用外部 DisplayPort TX 发送器芯片。

Zynq UltraScale+ MPSoC 还具有一系列片上存储器增强功能。该产品系列中的最大型器件，其可编程逻辑中除 Block RAM 外，还包含 UltraRAM。同时 Zynq UltraScale+ MPSoC 的处理内核共享 L1 和 L2 高速缓存。

Zynq UltraScale+ MPSoC 还采用具备 ECC 功能的位数更宽的 72 位 DDR 接口内核（64 位 +ECC 的 8 位）。该接口能提供用于 DDR4 的 2,400Mbps 速率，可支持 32GB 容量的更大内存深度 DRAM。

Zynq UltraScale+ MPSoC 上的专用安全单元可提供军事级安全性，诸如安全启动、密钥与库管理，以及防篡改功能等 —— 这些都是设备间通信以及互联控制应用的标准需求。此外，Zynq UltraScale+ MPSoC 的可编程逻辑系统还采用了针对 150G Interlaken、100G Ethernet MAC 和 PCIe® Gen4 的集成连接功能块。板载模拟混合信号 (AMS) 内核有助于设计团队利用系统监控器（System Monitor）测试其系统。

借助所有这些功能，不是任何应用都会用到 MPSoC 中的每个引擎。因此，赛灵思为 Zynq UltraScale+ MPSoC 提供了一个极其灵活的专用电源管理单元 (PMU)。该内核使用户能够控制电源域和分区（粗 / 细精度），仅为系统正使用的处理单元供电。而且，设计团队能够对该内核进行编程，以实现动态操作，从而确保系统仅运行执行给定任务所需的功能，进而降低功耗。PMU 还可实现众多安全性和可靠性，比如信号和误差的检测与缓解、安全状态模式，以及系统隔离与保护。

Myron 表示，归功于上述探讨的

16nm 新增的所有这些处理功能，采用 Zynq Ultra-Scale+ MPSoC 构建的设计相比采用 28nm Zynq SoC 实现的设计，性能功耗比优势平均提升 5 倍。

16nm ZYNQ ULTRASCALE MPSOC 测试标准

为了说明 Zynq UltraScale+ MPSoC 的性能功耗比优势，让我们来看一下该器件服务的众多应用中的 3 个应用的标准测试结果，不同颜色用于演示处理引擎的多样性（如图 6 所示）。

为创建一个运行全 1080p 视频的视频会议系统，设计人员采用一个带有独立 H.264 ASSP 的 Zynq SoC。利用 Zynq UltraScale+ MPSoC 的优势，设计人员现在能够在单个 Zynq UltraScale+ MPSoC 中实现 4Kx2K UHD 系统，而且在相同功耗预算条件下，该系统相比双芯片系统而言，性能功耗比提高了 5 倍。

赛灵思公司高级 SoC 产品线经理 Sumit Shah 表示：“在需求使用 Zynq SoC 和两个 ASSP 的公共安全无线电应用中，现在您只需使用一个 Zynq UltraScale+ MPSoC 就可实现整个设计，而且相对此前的配置，系统功耗降低了 47%，性能提升了 2.5 倍，从而实现了 4.8 倍的性能功耗比优势。”

Shah 说，同样的，此前实现在两个 28nm Zynq SoC 上的汽车多摄像头驾驶员辅助系统，现在可以缩小到一个 Zynq UltraScale+ MPSoC 上。单芯片系统比双芯片设计的性能提升 2.5 倍，功耗降低 50%。相对此前实现方案而言，这可将性能功耗比净提升 5 倍。

针对所有 UltraScale+ 产品系列的早期客户参与计划正在如火如荼进行。首个流片和设计工具的早期试用版本预计将于 2015 年第二季度推出。公司有望在 2015 年第四季度开始向客户出货 UltraScale+ 器件。

如需了解有关 16nm UltraScale 系列性能功耗比优势的更多信息，敬请访：*china.xilinx.com/ultrascale*。如需进一步了解有关 Zynq UltraScale+ MPSoC 的信息，敬请访问：*china.xilinx.com/products/technology/ultrascale-mpsoc.html*。

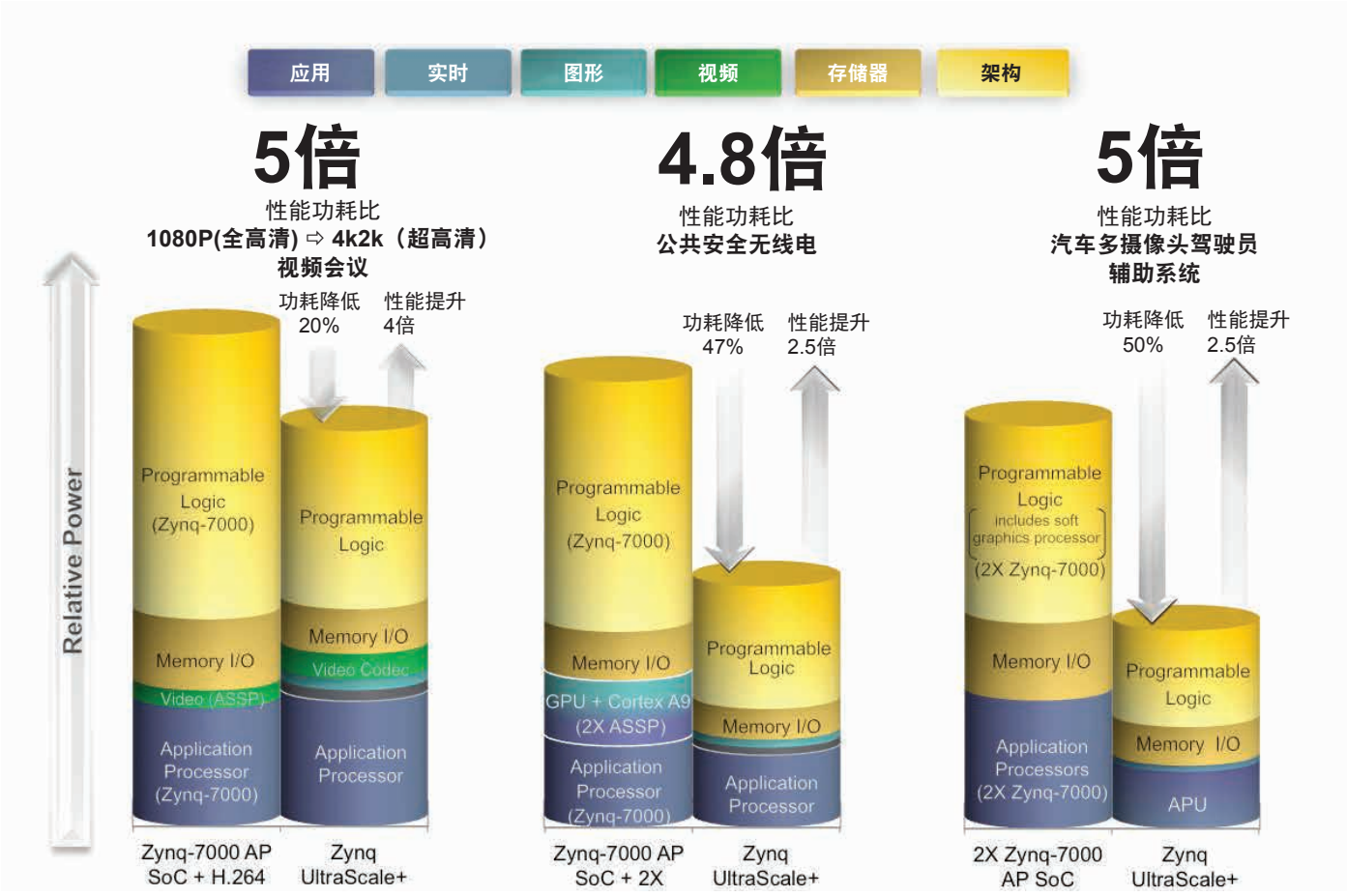


图6 – Zynq UltraScale+ MPSoC拥有丰富的处理模块、外设集和16nm逻辑块，可帮助设计团队创建出比采用28nm Zynq SoC实现的设计高出5倍性能功耗比优势的创新型系统。

60G毫米波回程链路随时准备提升蜂窝网络容量

作者：John Kilpatrick
咨询工程师
美国模拟器件公司（ADI）
John.Kilpatrick@analog.com

Robbie Shergill
战略应用经理
美国模拟器件公司（ADI）
Robbie.Shergill@analog.com

Manish Sinha
产品市场营销经理
赛灵思公司
manish.sinha@xilinx.com

基于赛灵思 Zynq SoC 的完整 60GHz 双向数据通信方案可提供小蜂窝回程市场所需的性能和灵活性。

全球蜂窝网络上对数据不断增长的需求迫使运营商想方设法在 2030 年前将容量提升 5,000 倍 [1]。要实现这一目标，需要将信道性能提升 5 倍，频谱分配提高 20 倍，蜂窝基站数量增加 50 倍。

许多此类新型蜂窝网络都将布置在室内，因为这里是流量的主要来源，而光纤则是将流量回传到网络的优先选择。但还有许多户外场合无法连接光纤或光纤连接成本过高，对于这种情况而言，无线回程是最可行的替代方案。

现可使用 5GHz 的免费频段，而且无需提供视距路径。但是，该带宽有限且由于流量和天线方向图大，无疑会受到该带宽其他用户的干扰。

对准备用于满足容量需求的数以千计的户外蜂窝而言，60GHz 的通信链路正在稳步兴起，将成为提供此类回程链路的有力竞争者。该频段也属于免费频段，但与 6GHz 以下的频段不同，它包含高达 9GHz 的可用带宽。此外，高频支持使用很窄的天线方向图，这样可在一定程度上提高抗干扰性。

由赛灵思和讯泰微波（Hittite Microwave，现属美国模拟器件公司（ADI）的子公司）共同开发的完整 60GHz 双向数据通信链路具有出色的性能和灵活性，能够满足小蜂窝回程市场的要求（图 1）。赛灵思负责开发该平台的数字调制解调器部分，而 AD 公司则负责开发毫米波射频部分。

如图 1 所示，创建该链路需要两个节点。每个节点包含一个发送器（配备一个调制器）及其相关的模拟发射链和一个接收器（配备一个解调器）及其相关的模拟接收链。

调制解调器卡与模拟和分立器件相集成。其包含振荡器（DPLL 模块），可确保频率综合的精度，并且所有的数字功能均在 FPGA 或 SoC 中执行。这种单载波调制解调器内核可支持从 QPSK 到 256QAM 的调制，信道带宽高达 500MHz，能够实现高达 3.5Gbps 的数据率。该调制解调器还可同时支持频分双工（FDD）和时分双工（TDD）传

稳健可靠的调制解调器设计方法能降低本地振荡器的相位噪声对本地晶振的影响。采用功能强大的LDPC编码技术可改善性能和链路预算。

输方式。

稳健可靠的调制解调器设计方法能降低本地振荡器的相位噪声影响，而采用功能强大的 LDPC 编码技术可改善性能和链路预算。

毫米波调制解调器

赛灵思毫米波调制解调器解决方案可帮助基础架构厂商为其无线回程网络开发成本优化的高度灵活的可定制链路。该解决方案主要面向赛灵思 Zynq®7000 全可编程 SoC 或 Kintex®-7 FPGA 器件，两者均属于赛灵思“领先一代”的 28nm 产品系列。

赛灵思解决方案具有完全的自适应性，其功耗低，尺寸小，可用于部署室内和全户外点对点链路以及点到多点微波链路。与其芯片产品一样，赛灵思的毫米波调制解调器解决方案发展路线图也极具前瞻性，使运营商能够独特地部署可扩展的现场可升级的系统。

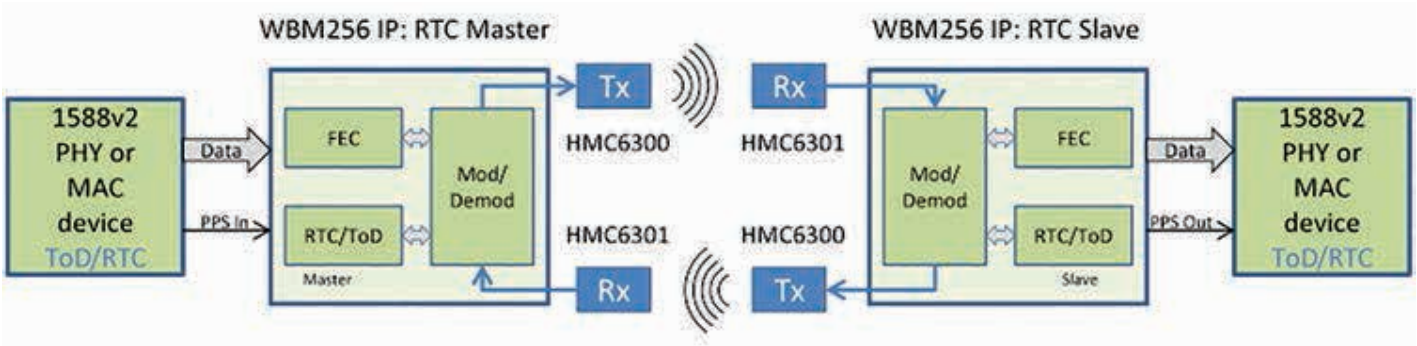
图 2 进一步显示了实现在 Zynq SoC 平台上的数字调制解调器的细节。平台的可扩展处理系统（PS）位于可编程逻辑（PL）旁边，其内置带有集成式存储器控制器和供外设使用的多标准 I/O 的双 ARM® Cortex ™ -A9 内核。

该片上系统（SoC）平台高度灵活。在本案例中，其用来执行各项数据和控制功能并实现硬件加速。图 2 所示的是集成式毫米波调制解调器解决方案以及配套的 PHY、控制器、系统接口和包处理器。

但是，用户可以根据所需的架构插入、更新或移除不同的模块。例如，用户可以选择实现 XPIC 组合器，这样可以将该调制解调器与另一个调制解调器以交叉极化模式加以使用。该解决方案实现在 PL 中，使用串行解串器和 I/O 作为各个数据路径的接口，比如调制解调器与包处理器之间的接口、包处理器和存储器之间的接口、调制解调器彼此之间的接口或 DAC/ADC 的接口。

该赛灵思调制解调器 IP 核的一些其它重要特性还包括：通过自适应编码和调制（ACM）功能实现的能够保持链路连续工作的自动无损和无误状态切换、

可改善 RF 功率放大器效率和线性的自适应数字闭环预校正（DPD）、能够保持时钟同步的同步以



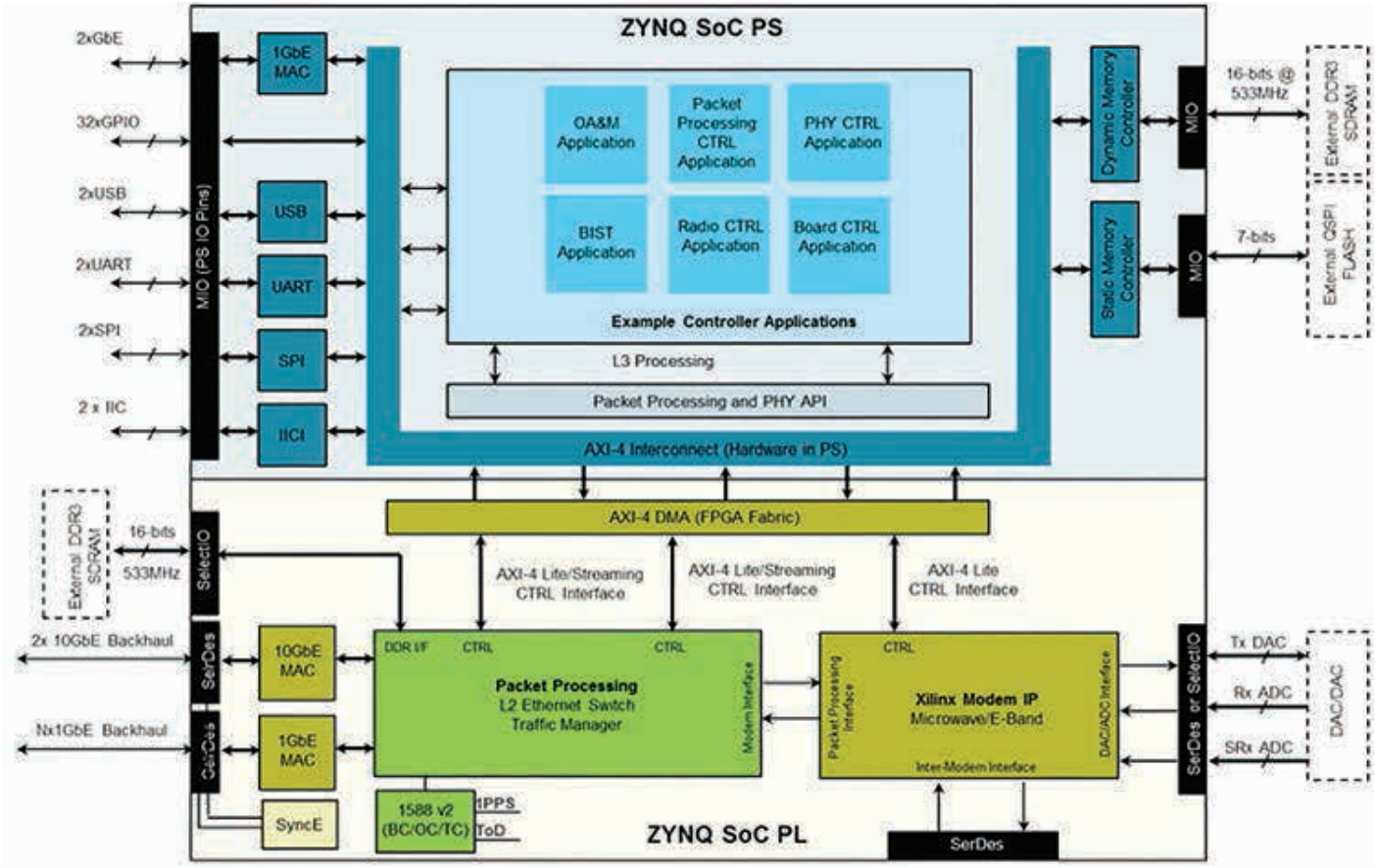


图2：用于无线调制解调器应用的All Programmable SoC

太网（SyncE）以及 Reed-Solomon 或低密度奇偶校验（LDPC）前向纠错（FEC）。可根据设计要求选择 FEC 功能。LPDC FEC 是无线回程应用的默认选择，而对于去程等低时延应用而言，Reed-Solomon FEC 则更加适合。

LDPC 实现经高度优化，并利用 FPGA 的并行性可完成编码器和解码器的计算工作。结果可使 SNR 实现显著改善。您可通过改变 LDPC 内核的迭代数量来应用不同级别的并行性，进而优化解码器的尺寸和功耗。此外，您还可根据信道带宽和吞吐量约束条件为解决方案建模。

该赛灵思调制解调器解决方案还配套提供强大的图形用户界面（GUI），用于实现显示和调试，并可提供信道带宽选择、调制方式选择等高层功能和硬件寄存器设置等底层功能。为让图 1 所示的解决方案实现 3.5Gbps 的吞吐量，该调制解调器 IP 核需要以 440MHz 的时钟速率运行。它将 5 个千兆位收发器（GT）用于连接接口，以支持 ADC 和 DAC，并把另外一些 GT 用于 10GbE 有效载荷或 CPRI 接口。

毫米波收发器芯片组

2014 年末，ADI 推出了自己的第二代硅锗（SiGe）60GHz 芯片组，其针对小蜂窝回程应用进行了大幅改进和优化。

HMC6300 发送器芯片是一款完整的模拟基带转毫米波上变频器。其采用以 250MHz 步进覆盖 57 到 66GHz 的改进型低相位噪声频率综合器，可支持至少 64QAM 的调制。输出功率已经提升到大约 16dBm 线性功率，同时集成式功率检测器可监测输出功率，使其不超出法规限制水平。

该发送器芯片可提供对 IF 和 RF 增益的模拟或数字控制。在使用更高阶调制的情况下，有时需要模拟增益控制，因为对幅度调制而言，离散增益改变可能会出错，导致出现误码。通过使用内置 SPI 接口可为数字增益控制提供支持。

对于需要在窄带信道中进行甚至更高阶调制的应用而言，可以向发送器中加入拥有更低相位噪声的外部 PLL/VCO，同时为内部合成器加设旁路。图 3 显示的是 HMC6300 的方框图。

该发送器能支持高达 1.8GHz 的

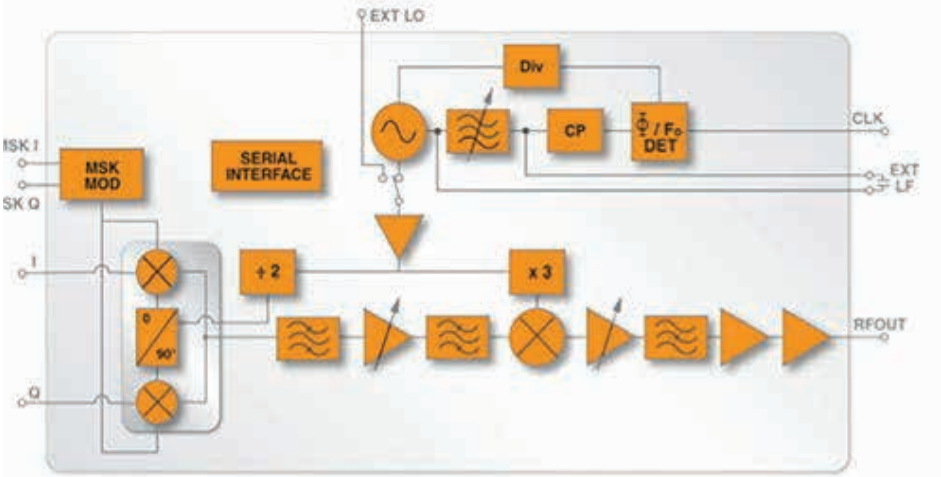


图3：HMC 6300 60GHz发送器IC方框图

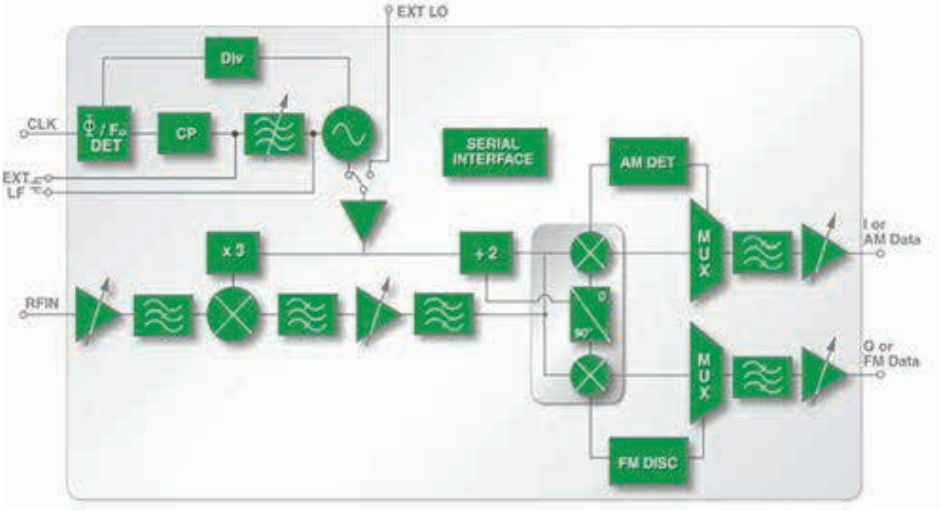


图4：HMC6301 60GHz接收器IC方框图

带宽。选配的 MSK 调制器可实现速率高达 1.8Gbps 的低成本数据传送，而无需使用高成本、高功耗的 DAC。

HMC6301 接收器芯片作为该器件的辅助器件，经过类似的优化能够满足小蜂窝回程的严苛要求。该接收器可将输入 P1dB 大幅提升到 -20dBm，并将 IIP3 显著提升到 -9dBm，从而处理短距链路，因为此时碟形天线的高增益会在接收器输入端产生高信号电平。

其它特性包括：最大增益设置下低至 6dB 的噪声因数；可调低通和高通基带滤波器；与发送器芯片中的新型综合器相同，且能在 57GHz 到 66GHz 频段支持 64QAM 调制的综合器；对 IF 和 RF 增益的模拟控制或数字控制。

图 4 显示了 HMC6301 接收器芯片的方框图。请注意，该接收器还包含一个 AM 检测器，用以解调开关键控（OOK）等幅度调制。此外，其还可提供 FM 鉴频器，用以解调简单的 FM 或 MSK 调制。

这就是用于为 QPSK 恢复正交基带输出和解调更复杂的 QAM 调制的 IQ 解调器之外的附加功能。

HMC6300 发送器和 HMC6301 接收器两者均将采用 4x6mm BGA 型晶圆级封装。它们将分别命名为 HMC6300BG46 和 HMC6301BG46，并定于 2015 年初提供样片。这些表面安装的器件可实现射频板的低成本制造。

图 5 所示的是实例毫米波调制解调器和射频系统的方框图。除 FPGA、调制解调器软件和毫米波芯片组外，该设计还包含一些其它组件。这其中包括 AD9234 双信道 12 位 1Gsps ADC；AD1944 四信道 16 位最高 2.8Gsps 发送器 DAC；以及 HMC7044 超低抖动时钟合成器（可支持 ADC 和 DAC IC 上使用的 JESD204B 串行数据接口）。

演示平台

赛灵思和 ADI 共同创建了一款演示平台实现方案，其采用位于赛灵思 KC705 开发板上的基于 FPGA 的调制解调器，配备包含 ADC、DAC 和时钟芯片的业界标准 FMC 电路板，以及两个射频模块评估板（图 6）。该演示平台包括用于调制解调器控制和视觉显示功能的笔记本电脑和用于复制典型毫米波链路路径损耗的可变 RF 衰减器。

该赛灵思 KC705 开发板采用可运行 WBM256 调制解调器固件 IP 核的 Kintex-7 XC7K325T-2FFG900C FPGA。开发板上的业界标准 FMC 夹层接插件可用于连接基带板和毫米波射频板。

毫米波模块可迅速插入到基带板上。模块具备用于 60GHz 接口的 MMPX 接插件以及用于可选配外部本地振荡器的 SMA 接插件。

该平台包含在频分双工连接的每个方向对应的 250MHz 信道中演示高达 1.1Gbps 点对点回程连接所需的全

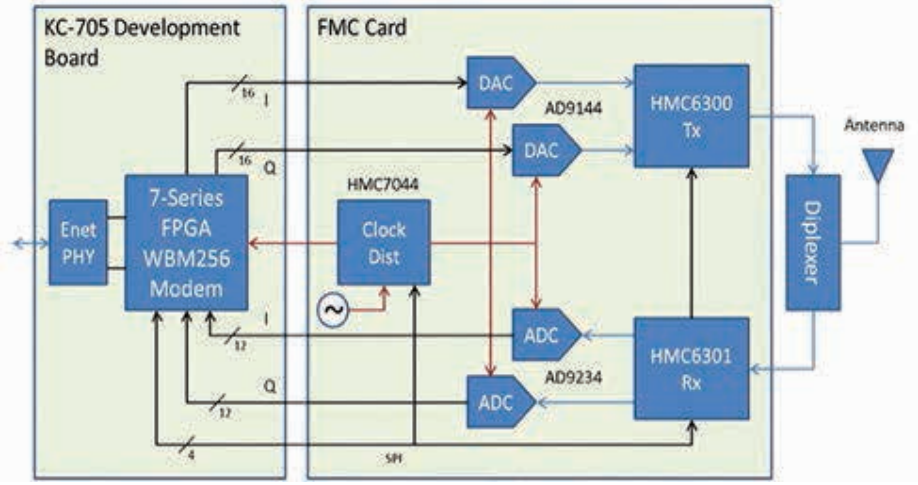


图5：使用赛灵思和ADI IC实现的实例参考设计

部硬件和软件。

模块化和可定制化

由于基于 FPGA 的平台能够实现高度模块化和可定制化，可为 OEM 厂商降低总拥有成本，因此 FPGA 越来越广泛地应用于各种无线回程解决方案中。此外，由于赛灵思 7 系列 FPGA/SoC 产品系列和高性能宽带 IP 核功耗明显下降，预计赛灵思的毫米波调制解调器解决方案将成为小蜂窝回程应用领域的领跑者。赛灵思 FPGA 和 SoC 非常适用于高速节能设计，并且其高速 GT 则可高效实现宽带通信和切换功能。赛灵思解决方案扩展能力出色，能够支持从运行在数百兆位速率的低端小蜂窝回程产品到同一硬件平台上速率为 3.5Gpbs 的回程产品的多种产品变化。

对于射频部分而言，收发器现已集成到基于芯片的 IC 中，并封装成表面安装的器件，便于实现低成本制造。ADI 的毫米波芯片组可满足小蜂窝部署的无线回程需要，并在功耗、尺寸、灵活性和功能方面稳占市场领先地位。此外，ADI 还可提供行业最佳数据转换器和时钟管理 IC，这都是该完整解决方案的关键组成部分。两家公司通力合作，旨在推动这一先进技术在整个行业中的广泛应用。

参考资料

1. 《关于超高容量网络的演进及颠覆性愿景》，国际无线工业联盟（IWPC），2014 年 4 月

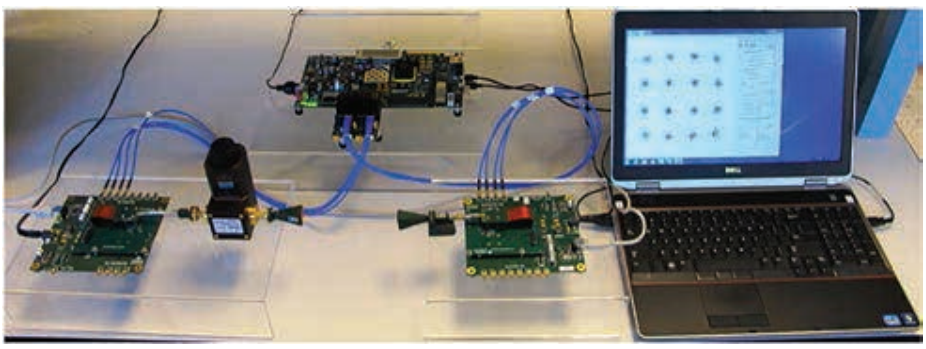


图6：工作中的演示平台

Xilinx SDAccel 和SDNet双双荣膺 Lightwave创新奖

光网络行业高度肯定赛灵思为系统工程师和软件工程师所提供的出色的技术支持，帮助他们充分利用全可编程逻辑和 SoC 器件的优势

2015 年 3 月 31 日，中国北京 - All Programmable 技术和器件的全球领先企业赛灵思公司（Xilinx, Inc.(NASDAQ:XLNX)）今天宣布其 SDAccel ™和 SDNet ™软件定义开发环境凭借其出色的技术成就双双荣膺 2015 年 Lightwave 创新奖。Lightwave 创新奖旨在表彰光网络行业中的顶尖产品和解决方案，所有奖项均由第三方光纤网络专家组成的独立评审团所选出。

凭借卓越的技术成就，赛灵思 SDAccel 和 SDNet 环境在设计工具类奖项中脱颖而出。所有产品都进行五分制打分，得分 4 分或以上表明评委认为该产品提供的技术特性和性能具有明显本质上的优势。赛灵思每个产品的得分均达到了 4.5 分。

赛灵思于 2014 年 3 月推出面向网络的软件定义规范环境（SDNet），也是赛灵思 SDx 系列的首个成员。借助该环境，系统设计师和软件架构师能够通过高级用户自定义规范和高度优化的 FPGA 技术轻松地创建高性能包处理系统。

赛灵思于 2014 年 11 月再为 SDx ™系列推出第二款产品，那就是用于加速的 SDAccel 软件定义开发环境的。采用该环境，系统工程师和软件工程师能通过 FPGA 为数据中心创建出功耗优化的、可重配置的加速计算环境。SDAccel 可在通用开发环境中将针对 OpenCL ™、C 和 C++ 的编程环境、优化编译技术，以及动态可重配置的加速器完美结合在一起。而赛灵思 SDx 系列的第三大成员 SDSoC 于 2015 年 3 月初宣布推出，可支持更广泛的嵌入式软件及系统设计人员充分利用 Zynq 全可编程 SoC 和 MPSoC 的优势。

MACsec IP核大幅提升数据中心安全性

作者：Paul Dillien
High Tech Marketing公司顾问
paul@high-tech-marketing.co.uk

Tom Kean博士
Algotronix公司总经理
tom@algotronix.com

数据中心设备设计人员将结合采用基于 FPGA 的内核来提供安全的高性能以太网链路。

云存储和 IT 服务外包对 IT 经理而言极富吸引力，因为这不仅能降低成本，而且还可减轻支持工作。然而有一个大的顾虑就是，这样做会使敏感数据流出公司防火墙外，造成安全隐患。这种顾虑是完全可以理解的，因为信息对于许多公司而言是最宝贵的资产，无论是会计、客户还是制造相关的数据。

而现在，设备制造商能够通过使用赛灵思基于 FPGA 的解决方案来提高性能和安全水平。满足以太网新标准 MACsec 要求的 Algotronix 综合安全子系统采用基于赛灵思 FPGA 的高性能、低时延、高能效 IP 核。

基于 FPGA 的解决方案比基于软件的解决方案速度要快得多。此外，专用硬件可接管系统处理器，使其处理其它任务，如深度数据包检查等。或者，设计人员也可采用成本更低的处理器。

加密和认证

保护信息的一个显著策略就是当数据在网络中传输和在数据中心周围移动时对其进行加密。一旦数据被非授权方渗透网络链路而拦截，数据加密能够确保其无法被读取。原则上，数据还应经过认证，从而确保其完整性。消息认证旨在检测原始加密数据是否已被篡改，包括因传输错误而造成变更，抑或是被攻击者为从中牟利而恶意破坏。

目前以太网传输已成为主流通信方式，这是一种既高效又具有可扩展性的高速传输方法。随着以太网标准的普及，以太网传输成本不断降低，这一优势使其更加引人注目，进而确保以太网继续成为首选的 L2 技术。不过，就在几年以前，以太网标准还没有任何加密规范要求，只能采用运行在通信协议栈上层的 IPsec 等技术来完成加密工作。

现在，根据 IEEE 802.1AE 标准，最新以太网标准扩展版本新增了大量安全措施。该技术在几年前正式确定，其采用集成式安全系统来加密并认证消息，同时检测并应对一系列网络攻击。该标准被称为“媒体接入控制安全（Media Access Control Security）”标准，常常简称为“MACsec”。Algotronix 从几年前就开始努力推出能够根据多种不同数据速率要求提供硬件加速加密功能的 IP 核。

（Algotronix 还可提供面向 IPsec 的 IP 核，该产品与 MACsec 产品的接口类似，对需要支持双重标准的系统而言是不错的选择。）

简要介绍 MACsec 系统，帮助了解规范的全面性，同时深入说明实现该规范的复杂程度。

信任实体

MACsec 指的是由网络上的节点组成的一系列信任实体。每个节点都能接收加密消息和明文消息，而系统策略则用于明确如何处理每条消息。内核包括明文消息的旁通选项，无需认证或验证。与 IPsec 等作为端到端技术运行在 L3/L4 的协议不同，只要数据包进入或离开以太网 LAN，MACsec 就能对每个数据包进行解密和验证。

MACsec 适用于星型或总线型 LAN 等以太网拓扑结构，也可支持点对点系统。

MACsec 标准采用安全实体（SecY）方法，也就是每个节点或实体都具备与其以太网源地址相链接的唯一密钥。为支持多个虚拟 SecY，我们设计出了该 IP 核的 1G 版本。因此，单个以太网 MAC 能针对多用户 LAN 等应用配备多个与之关联的 MACsec SecY。MACsec 通常与 IEEE 801.1X-2010 或互联网密钥交换（IKE）配合使用，可实现网络周围的安全密钥分配。

数据中心之所以会选择 L2 连接功能在数据中心内移动数据包，是为了提高速度，并最大程度地降低时延和减少数据包中的开销数据。相比之下，如果用诸如 IPsec 等安全的 L3 技术进行通信，消息必须传到协议上层进行处理，而这会增加时延。

此外，L2 解决方案也能避免创建 L3 安全策略这一复杂工作。

数据中心能够采用 MACsec 提供防火墙后台的保护，或将其用在数据中心之间的直接链路上。系统管理员可授权设备以安全方式进行通信。设备能够检测错误或误用情况，如拒绝服务攻击（DOS）。

符合可编程要求

市场因需求不同，日趋细分化。可定制 FPGA 解决方案理想适合于 MACsec。起初，MACsec 的设计是作为一项技术应用于城域网，而现在在数据中心中也找到了其用武之地，这就提高了对基于 FPGA 的解决方案的整体需求。

Algotronix 开发 MACsec 内核是一个自然演进，因为我们已经打造了一系列称为“AES-GCM”的加密引擎。这些内核的运行速率分别为 1G、10G 和 40G。我们通过流水线、提高时钟速率并从赛灵思 Artix® 器件逐步发展到 Kintex® 器件乃至 Virtex® FPGA，来实现上述速率的。我们将利用这些技术来推动 Virtex UltraScale ™器件上的吞吐量，使其达到 100G。

我们使用 FPGA 中的 IP 核能够实现多种不同性能，可支持从 1GbE 到 10 GbE 的不同速率（即，内核在最坏情况下的实际吞吐量）。此外我们还计划推出 40G 和 100G 的版本。这比基于软件的系统要快得多。内核通常直连接到硬件 MAC（如图 1 所示），因为 FPGA 芯片上的嵌入式存储器的软件会尽可能足够快地传输数据，以满足其吞吐量要求。如果在硬件上实现安全功能，同时从未向软件提供未加密密钥，那么系统就不那么

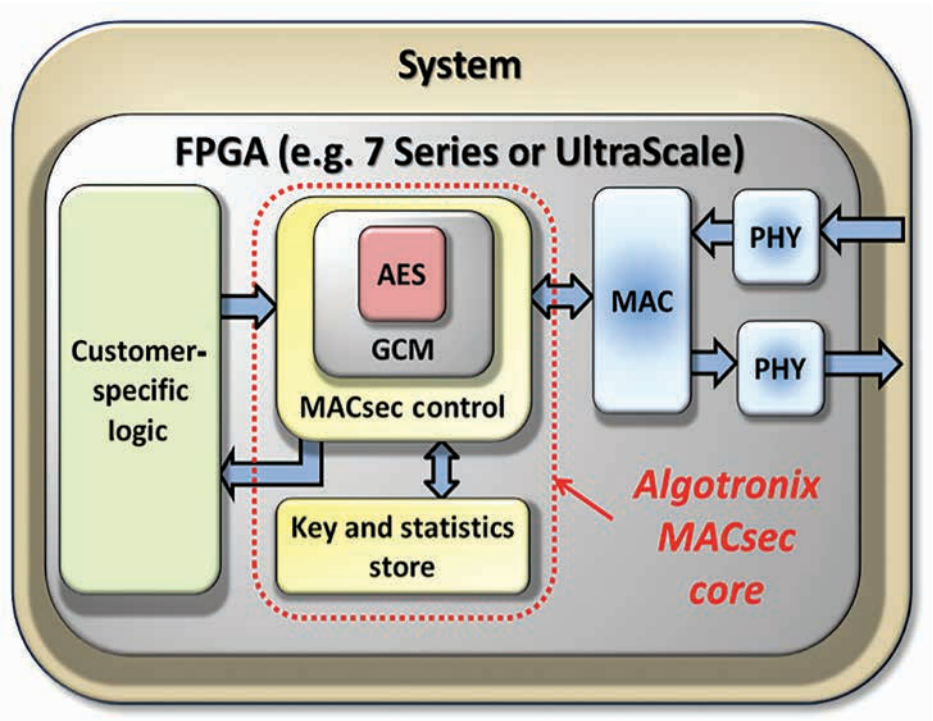


图1 – 整个MACsec IP核位于FPGA内，可实现最大安全性。

这样就算IT专业人士必须考虑系统的整个软件层面的情况时，也能更方便地分析系统漏洞。

容易受到特洛伊木马（Trojan horse）和病毒等常见软件攻击。

另一个重要考虑事项就是 FPGA 进行算法加速的系统要大幅降低功耗。加速的算法包括加密函数等，免得再用软件去实现加速。FPGA 比软件解决方案的能效明显要高得多。

所有 Algotronix 加密内核都内置了一项重要属性，那就是能够在 Block RAM 或 FPGA 架构的查找表（LUT）中实现称为“S-Boxes”的关键模块。有了该属性，客户可通过综合平衡两种资源类型便能利用现有资源实现设计。比方说，如果 MACsec 内核外的设计未占用大量的 BRAM，那么就可用 Block RAM 来实现 S-Boxes，否则就用 LUT 来实现。

MACSEC细节

MACsec 系统的设计理念是：每个数据源使用不同的加密密钥。接收到消息后，接收器会在片上 CAM 的列表中进行查找，明确用以解密数据包的正确密钥。每个数据包都有编号，确保能检测并拒绝接收重复或重新发送的数据包，这种方法可防范“中间人”攻击。

MACsec 还会收集有关被拒收的数据包数量的统计数据以及拒绝的原因。提供统计数据以支持攻击检测是超出基本加密隐私、认证和防止重发功能之外的更高一层的安全性，能让系统管理器主动应对正在进行的攻击。

我们采取的方法是对业经验证的 AES-GCM 内核周围的 MACsec 逻辑进行“打包”。就此而言，设计高效快速的加密内核只是设计挑战的一部分。MACsec 标准涉及面广，包括许多变量。

举例来说，该标准最初只指定 128 位的加密密钥。采用 128 位密钥，数据进行 10 次转换（被称为“轮”）后在内核中完成加密过程。该标准经修订后可提供 256 位加密密钥，整个数字加密过程历经 14 轮。这是通过添加流水线级数并提高密钥存储所需的内存带宽才实现的。

MACsec 与以太网流量类型无关，也对更高层协议透明。推出这些内核后，就能方便地将 MACsec 添加到系统中，从而进一步提高网络防护。配备 MACsec 的站点仍能与未采用 MACsec 额外安全保障机制的其它站点进行通信。

从媒体接入控制器（MAC）将以太网数据包提供给 MACsec 内核。您可结合使用 1G MACsec 内核、片上收发器和三模以太网 MAC（TEMAC）构建高效的小型解决方案。每个数据包都包含发起传输的源码的目的地和地址。该标准保存在 MACsec 系统中，但一个重要的因素是，在多次反射传输中，“源码”将是传递数据包的最终设备的地址。因此，与可被视为端到端方案的 IPsec 不同，MACsec 是以逐跳方式工作的。对于每次跳跃，MACsec 都要求输入端的所有加密数据进行解密，然后使用分配给传输设备的唯一密钥再重新加密。解密的明文可在每一级提供数据包检查功能，如图 2 所示，也能供流量管理器用以管理数据流。

在 MACsec 标准中，图 3 给出的报头包含附加字段“MAC 安全标签（SecTAG）”，其可定义 EtherType，并标明数据包是否加密。数据附加在 ICV 字段的消息末尾，则表示已经认证。

ICV 协同加密密钥，可认证包括报头和 MACsec 标签的帧，进而确保帧的源地址和目的地地址都不会被篡改。我们在 FPGA 架构中实现该逻辑，确保其能够具备快速的可预测的时序，从而最大程度地降低时延。

MACsec 内核包括连接到每个源地址的查找表。该表包含的密钥必须能够用来成功解密消息，我们精心设计该功能，使其能够高效实现在 LUT 和器件的 Block RAM 中。我们充分利用 FPGA 解决方案的灵活性，采用实现方案选项（如可采用 128 位或 256 位密钥，也可修改内核支持的虚拟 SecY 数量）来设计内核。

新标准的另一个重要特性就是，MACsec 可收集数据包级的统计数

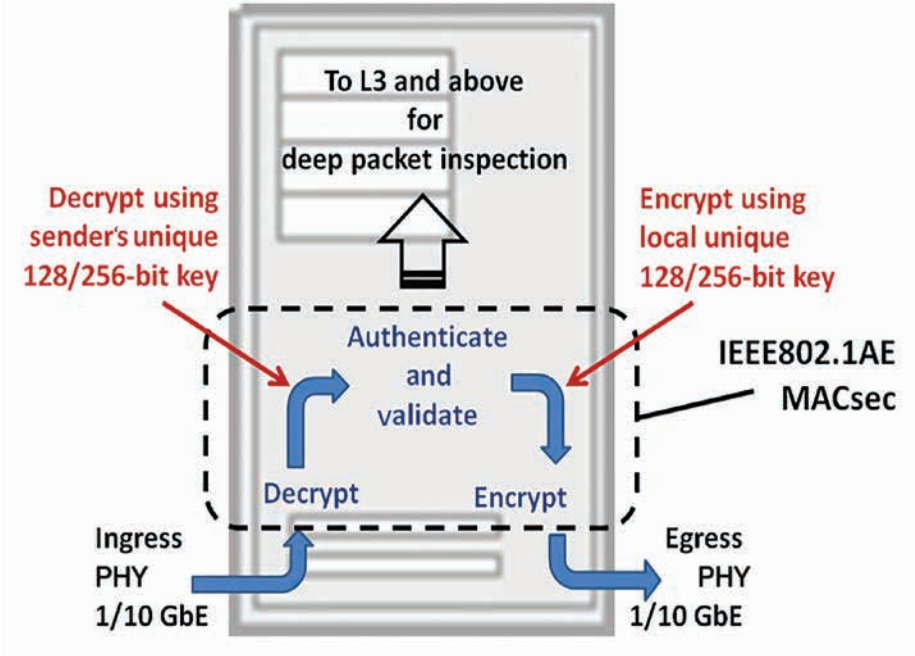


图2 – 消息在入端口被解密，并在出端口被加密。

据。系统管理员能够了解有关信息（如多少数据包因为延迟而被拒收，或者因为无效解密密钥或使用错误密钥而未通过完整性检查），并将这些统计数据与正确传输的数据包数量进行比较。

MACsec 标准可面向点对点应用提供精简选项。这样就无需采用 CAM 从数据包中的显式安全通道标识符和单点到多点操作的选择方案中确定密钥。我们的内核还可支持关联于单个以太网的多个虚拟 SecY，这样，不同的密钥就能用来加密从 MAC 传输到不同目的地的数据。MACsec 标准将这种配置定义为多用户局域网，因为这就像这些目的地位于不同以太网 LAN 上一样。该特性使得系统能够通过使用不同密钥加密输出来对接收设备进行分区。

数据中心可能会采用多个 SecY 来创建虚拟分区，这样客户 A 的数据就可通过唯一的加密密钥与客户 B 的数据划分开。

数据中心内部通信可根据需要进行组织来分隔选定的机架，进而提供虚拟隔离区。这种功能可保护数据完整性，并应对数据中心和云应用中的隔离问题。无论是意外错误连接还是恶意行为（见图 4），MACsec 系统都能检测到未经认证的数据包，系统管理员可通过设置策略将其隔离或删除。

所有数据加密和解密都在端口级进行。除了附加的 MACsec 报头和较少的额外时延，打开端口级加密不会增加开支，也不会对性能造成其它影响。

通过采用符合 IEEE 802.1AE 要求的加密 Ethernet Lecel 2 方案，设备厂商现在能用这些内核推动其系统特色化。基于云的用户可能与其他用户相互之间不信任，但他们现在能够从 MACsec 提供的数据机密大获裨益，并且数据源认证功能可进一步保护他们的数据。

设备制造商则能选择可用的 IP 核来满足 1Gb 和 10 Gb 以太网吞吐量的需求。

这种架构设计能通过 Kintex 或 Virtex FPGA 器件轻松实现 10Gbps 的速度。在最坏情况下，该设计只需更改每个数据包的密钥便可支持巨型帧和最小型数据包。内核符合全面规范要求，每个 MACsec 内核都能支持各种常用的 FPGA 产品系列。

配套提供源码

Algotronix 采取了不同寻常的措施，即为所有许可的内核提供 HDL 源码。这样做的主要动机是支持客户检查，以便确保代码不含病毒或特洛伊木马代码，而且不会强制进入非授权状态或操作。有了源码，就能降低客户安全审核的成本和复杂性。此外，源码可加速设计进程，因为工程师能够方便地尝试使用诸如加密、解密或加密 / 解密等不同配置参数和密钥长度，并了解其各自仿真内核中的信号状态。

您可对内核进行配置，通过实现较宽的数据路径来提高吞吐量，或通过选择较窄的数据宽度来最大程度地减小 FPGA 封装尺寸。拥有源码还有其它更多优势，包括更便于了解内核工作情况；也让文档记录和归档变得更快捷方便。

此外，还配套提供了广泛的验证测试平台，可帮助客户在 ModelSim 等工具中确认操作是否正确。测试平台包括 MACsec 的行为模型和 MACsec IP 核的自检版本，能针对行为模型检查可综合硬件的输出。这种自检设计可在用户仿真中实现实例

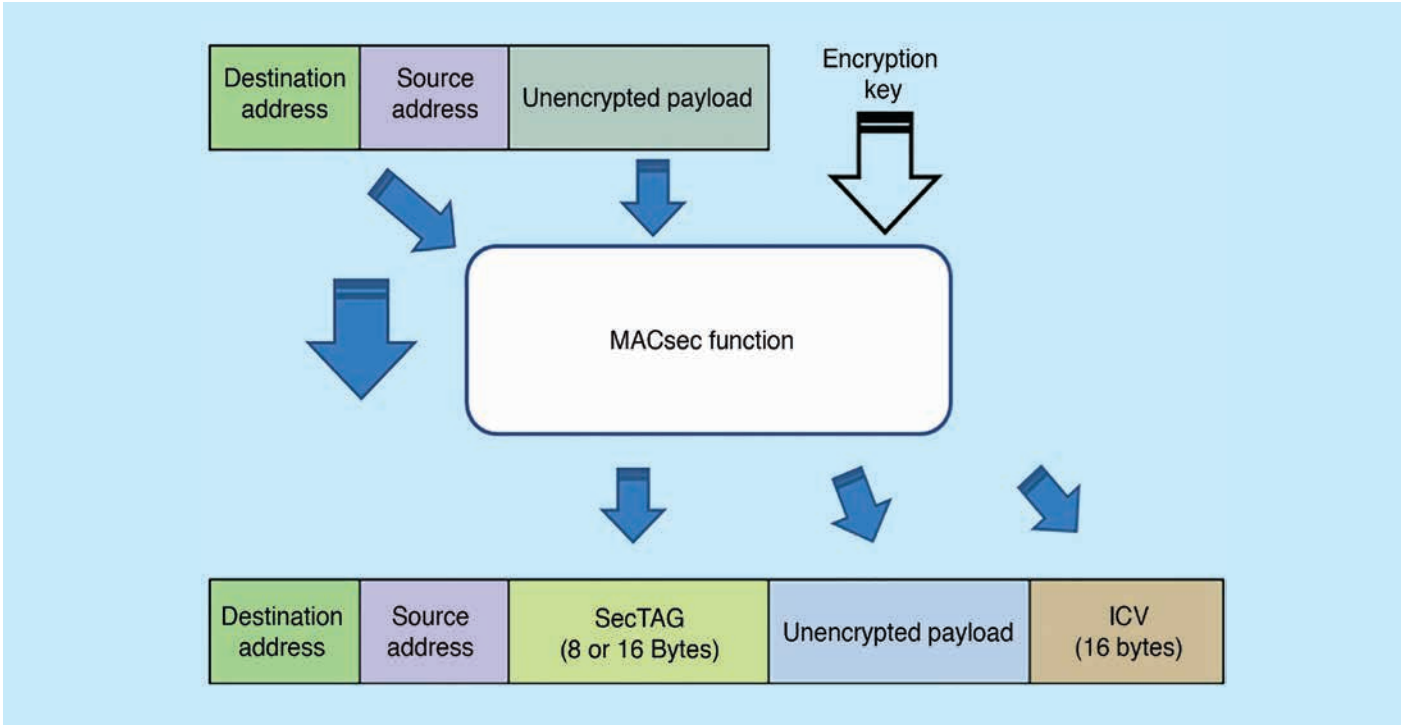


图3 – MACsec帧结构包括MAC安全标签（SecTAG）字段，其可定义EtherType，并标明数据包是否加密。

化，便于测试实际用户设计环境下的内核表现，并在错误驱动的情况下提供有用的诊断信息。

内核可提供许多选项，因此精确的资源数量将取决于您如何选择参数，如数据速率、密钥长度和所选SecY数量以及其它。然而，赛灵思网站IP部分列出的10G MACsec内核采用6,638个slice、20,916个LUT和53个BRAM块。如需获取许可证选项，敬请联系Algotronix。

赛灵思低功耗FPGA与Algotronix MACsec内核的完美结合为设备制造商实现产品差异化提供了高性能、低时延的解决方案。安全特性使得数据中心能够确保其客户机密，同时还可帮助安全管理员检测并打击恶意行为。

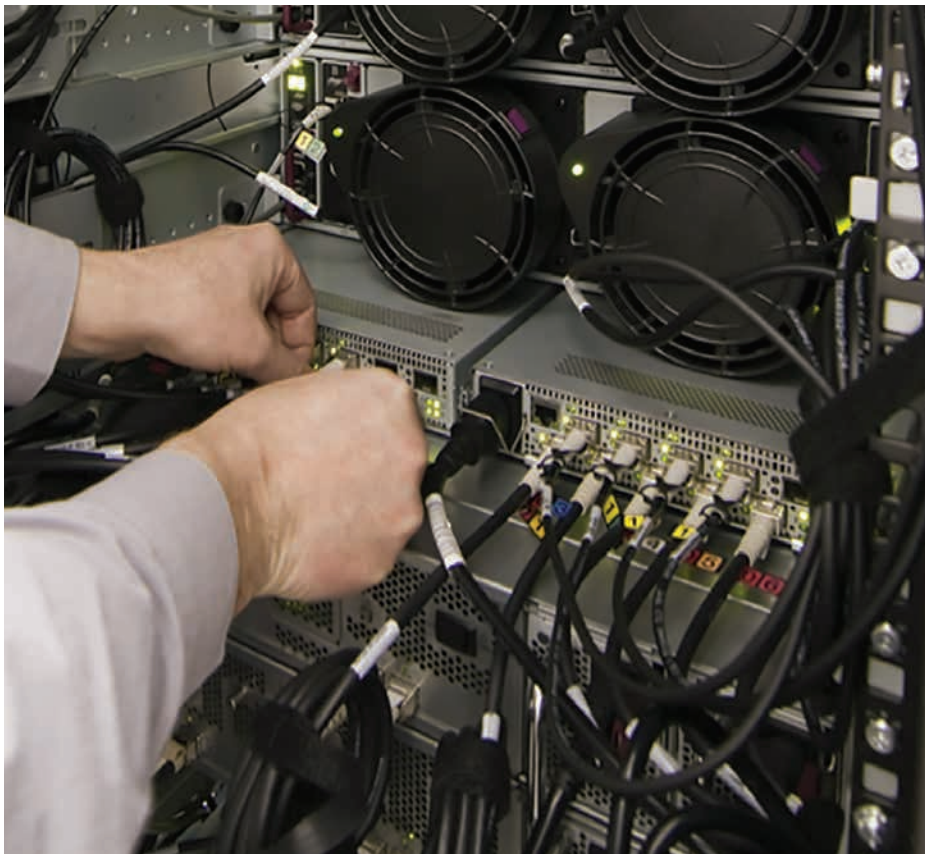


图4 – MACsec将拒绝通过错误连接抵达的数据包，无论是因为意外情况造成还是恶意行为导致。

利用Xilinx Zynq SoC简化您的“热”测试

作者：Lei Guan
研究员
爱尔兰阿尔卡特朗讯公司贝尔实验室
lei.guan@alcatel-lucent.com

本文介绍一种使用 Zynq SoC 和赛灵思 IP 核简化高速光学收发器模块热测试的方法。

随着数据中心内部光学收发器模块的传输速度提高到前所未有的高度，数据中心内每个机架的温度也在不断大幅上升。机架中有多个这种发热的高速模块堆叠在一起，加之有多个机架并排摆放，这样，温度倍增。温度的急剧上升可能会导致超过芯片的热限制，从而造成灾难性的芯片故障，继而对整个数据中心系统产生不利影响。因此，工程师在设计光学收发器模块时必须考虑到热属性。设计人员必须要将注意力集中在热源上，并尝试用模块级甚至机架级的高效冷却方法对热源加以控制。

工程师在测试光学模块的热属性时通常有两种选择。他们可以使用复杂的网络数据生成器来创建高速（10-Gbps）链路，然后对光学模块的热属性进行测试；或者充分利用具有可调预设电压和电流的“热等效”模块，这样无需使用真正的高速数据即可仿真模拟热学条件并评估热属性。

这两种方案都不够理想。第一种方案需要专业的高速网络数据生成器，因此操作起来成本很高；而第二种方法又太抽象。热等效模块无法完全反映物理交换行为所引起的温度变化。

不过，最近我的团队在爱尔兰阿尔卡特朗讯贝尔实验室通过使用赛灵思 Zynq®-7000 全可编程 SoC 平台和赛灵思 IP 核完成光学模块的热属性测试工作，

我选择赛灵思ZC706评估板的原因在于主器件上的GTX收发器可以轻松达到10Gbps的单线数据传输速度。

从根本上简化了这一过程。我们来仔细了解一下如何成功简化测试。

预设计分析

这种热测试的基本要求是不断用10Gbps数据激发XFP光收发器，同时使用IR摄相头跟踪和描述温度变化特性。

我选择赛灵思ZC706评估板作为开发主机，因为主器件——即Zynq-7000 SoC XC7Z045（速度等级-2）上的GTX收发器可以轻松达到10Gbps的单线数据传输速率。Zynq SoC器件包含一个采用ARM®内核的处理系统（PS）和一个Kintex®-7

FPGA可编程逻辑（PL）架构。首先，PL晶片上的资源足以处理10Gbps双工数据传输。然后，我们可在日后需要的时候使用PS生成特定用户数据模式。

我们的热学团队将一块Finisar XFP评估板用作光学收发器的外壳。该FDB-1022评估板可作为功能强大的评估主板，能够很好地评估最先进的10Gbps XFP光学收发器。SMA连接器可用于差分数据输入和输出。该评估板经配置后可直接通过SMA连接器连接1/64时钟（即，156.25 MHz = 10 GHz/64），进而为模块提供时钟。

系统设计

在进行FPGA开发工作的七年时间里，我发现尽可能多地使用赛灵思内核可以显著缩短设计周期。在本设计中，我采取了相同的策略，并从集成式误码率测试器（IBERT）内核开始着手。您可利用该内核进行数据模式的生成和验证，从而评估Zynq SoC上的GTX收发器。然后，为了对设计正确布线，我创建了一个基于混合模式时钟管理器（MMCM）内核的相位对齐时钟分布单元，可同时对FPGA架构上的GTX收发器和XFP评估板上的光学收发器提供时钟。图1为系统方框图。

针对该设计项目，我使用了赛灵思的老式工具ISE®设计套件，并分三步完成这项工作。

第一步，使用CORE Generator™工具创建IBERT内核。

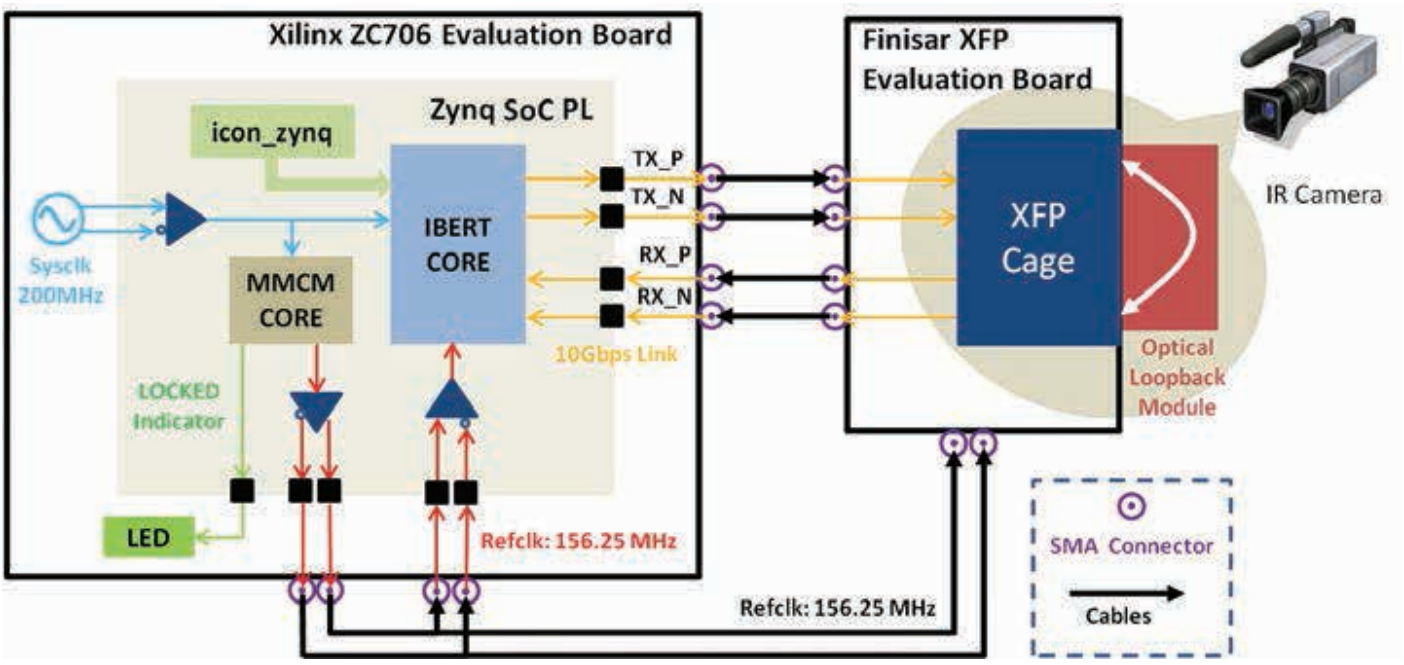


图1 – 所建议的系统的方框图，包含连接实例。

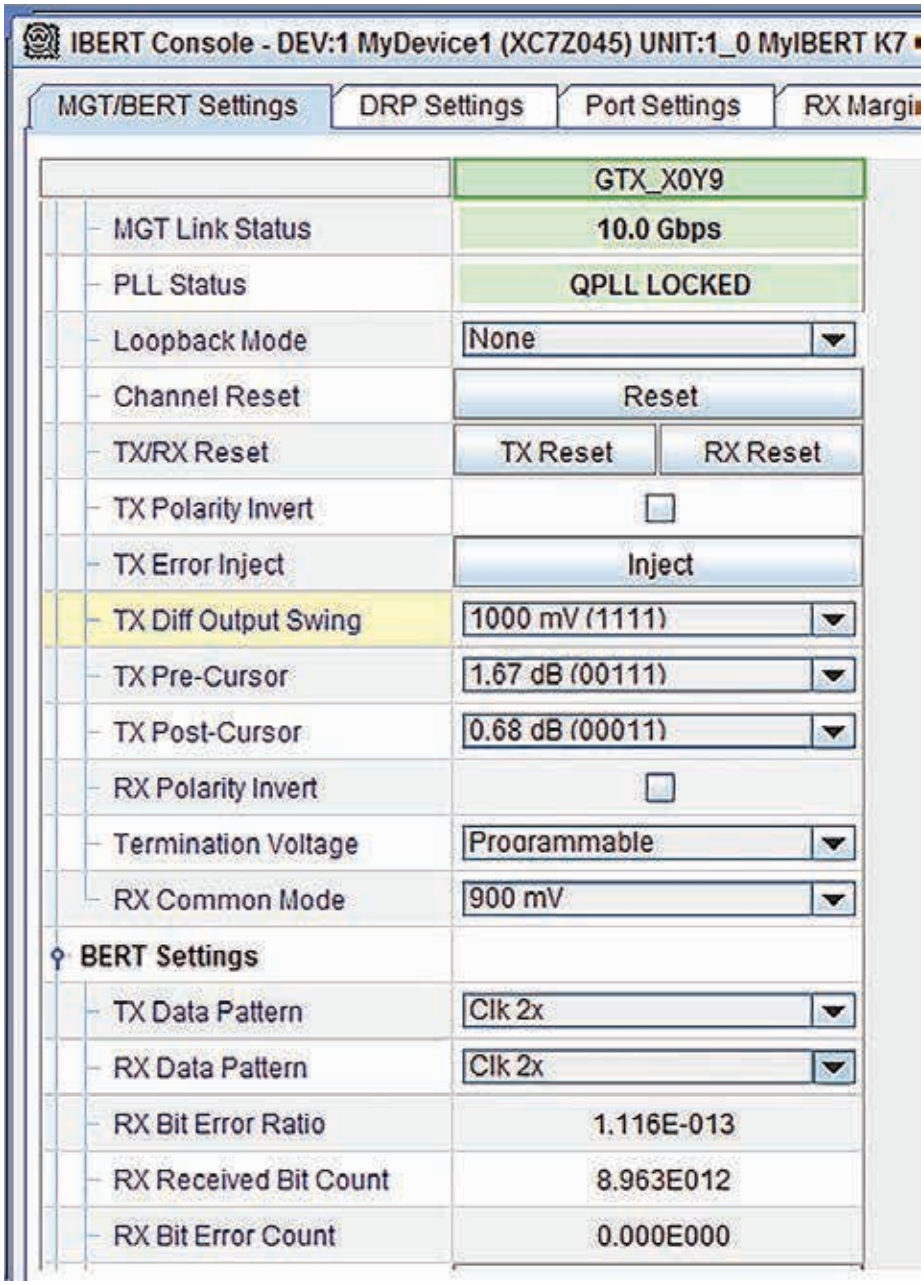


图2 – ChipScope Pro屏幕截图

这里提供了一些针对该 IBERT 7 系列 GTX（ChipScope ™ Pro）IBERT 内核的关键设置。在我的设计中，IBERT 系统时钟来自开发板上的外部时钟源，即 200MHz 差分时钟，P 引脚位置 = H9，N 引脚位置 = G9。GTX 时钟模式独立于 QUAD 111；并且我将线路速率设置为最大速率 = 10Gbps。我把 GTX 的参考时钟设置为 Refclk = 156.25 MHz，且 Refclk 时钟源 = MGTREFCLK1 111。

第二步，我使用 CORE Generator 创建了一个 MMCM 内核。首先必须正确设置该工具的时钟向导。为此，我将时钟特性设置为频率综合和相位对齐。输入时钟必须与开发板上的系统时钟相同（即 200MHz）。我还将目标派生时钟设置为 156.25MHz，占空比设置为 50%。我使用两个额外信号（RESET 和 LOCKED）来控制和指明 MMCM 内核。

第三步，用赛灵思工具对所有元素进行集成。在本项目中，我使用的是 ISE 设计套件 14.4。以后我打算改用 Vivado® 设计套件，以便最大程度地提高芯片性能。

我首先在 ISE 中创建一个新的项目，然后将 IBERT 内核文件夹（example_ibert_gtx.vhd、ib- ert_gtx _ top.ucf、ibert_core. ngc 和 icon_zynq. ngc）移动到 ISE 项目中。然后，从 MMCM 内核文件夹（步骤 2）将 mmcm_core. vhd 添加到 ISE 项目。再然后，将 example_ibert_gtx.vhd 用作顶层模块，对 mmcm_core 进行实例化，并将三个新信号（CLK_ OUTPUT_P、CLK_OUTPUT_N 和 LED_REFCLK）添加到设计中，随后在 ibert_gtx_top. ucf 中进行相应的引脚分配。

系统测试

在生成 .bit 文件后，FPGA 设计就可随时用于仿真具有 10Gbps 链路的 XFP 光学收发器。我把两块开发板连接起来（如图 1 所示），然后打开 ChipScope Pro 分析器，用新建的 .bit 文件配置器件。接下来，双击 IBERT 控制板，会弹出一个新的图形用户界面（如图 2 所示）。我们可以使用该界面对预定义的数据模式进行优化，例如 Clk 2x (1010 … .)，以及伪随机二进制序列（PRBS），进而彻底评估光学收发器的热性能。

通过将赛灵思内核与 ZC706 评估板结合起来使用，即可轻松构建用以评估高速光学收发器的测试平台。在本设计中，我们展示了对单个 XFP 模块的评估。不过，您可以直接应用这种设计方法来快速构建一个用来测试多个光学收发器模块的逻辑内核。

如需了解更多信息，敬请通过邮箱 *lei.guan@al- catel-lucent.com* 联系作者。

双管齐下，充分发挥Zynq SoC优势

利用赛灵思 Zynq SoC 上的两个 ARM A9 内核可以显著提高您的系统性能。

作者：Adam P. Taylor
e2v公司电气系统部总工程师
aptaylor@theiet.org

赛灵思 Zynq®-7000 全可编程 SoC 的众多优势之一就是拥有两个 ARM® Cortex ™ -A9 板载处理器。不过，很多裸机应用和更为简单的操作系统只使用 Zynq SoC 处理系统（PS）中两个 ARM 内核中的一个，这种设计方案可能会限制系统性能。

根据所开发的应用类型不同，可能需要这两个处理器都运行裸机应用，或者需要在每个处理器上运行不同的操作系统。例如，其中一个处理器执行关键计算任务，从而运行裸机 /RTOS 应用，同时第二个处理器通过 Linux 提供 HMI 和通信功能。

什么是多处理?

这两种方案都属于多处理。简单定义：多处理就是在一个系统中使用一个以上的处理器。多处理架构可允许一次执行多个指令，但并非必须如此。

多核处理包括两种类型：对称和非对称。

对称多处理是通过将负载分配给多个内核，从而能够同时运行多个软件任务。而非对称多处理（AMP）则是使用专用处理器，或者针对特定应用或任务在相同处理器上执行应用。

根据定义，使用 Zynq SoC 上的两个内核执行裸机应用或不同操作系统都属于非对称多处理。Zynq SoC 上的 AMP 可能涉及如下几种组合：

- 在内核 0 和内核 1 上运行不同操作系统；
- 在内核 0 上运行操作系统，在内核 1 上运行裸机应用（反之亦然）；
- 在两个内核上均运行裸机应用，执行不同程序。

当您决定在 Zynq SoC 上创建 AMP 系统时必须考虑一个实际问题，即 ARM 处理器内核同时包含必须进行正确寻址的私有资源和共享资源。这两个处理器都有私有的 L1 指令和数据高速缓存、定时器、监视时钟以及中断控制器（针对共享和私有中断）。另外还存在一些共享资源，常见的有 I/O 外设、片上存储器、中断控制器分配器、L2 高速缓存和位于 DDR 存储器中的系统内存（见图 1）。这些私有和共享资源均需要精心管理。

每个 PS 核都有自己的中断控制器，能够利用软件中断实现自身与一个或两个内核的中断。这些中断通过 ARM 的分布式中断控制器技术完成分配。

由于针对每个内核执行的程序都位于 DDR 存储器内，因此您必须特别注意以确保对这些应用进行正确分割。

建立AMP

建立 AMP 并使其运行在 Zynq SoC 上所需的关键因素是引导载入程序，该程序会在第一个应用载入到存储器后寻找第二个可执行文件。赛灵思在 XAPP1079 中提供了有用的应用指南和源代码。该文档包含修改后的第一阶段引导载入程序（FSBL）和独立 OS，可用来创建 AMP 系统。

首先要做的是下载与应用说明配套提供的 ZIP 文件，再将 FSBL 和 OS 这两个要素解压到期望的工作目录。然后，必须给名为 SRC “design” 的文件夹重新命名。现在，非常重要的一点是一定要确保软件开发套件（SDK）知道

使用软件中断与硬件中断基本相似，区别只在于您如何触发它们。

这些新文件（修改后的 FSBL 和独立 OS，两者兼备）的存在。因此，下一步需要更新您的 SDK 库，以便使其知道这些新文件的存在。

这很容易实现。在 SDK 中赛灵思工具菜单下选择“库”，然后选择“新建”，随之导航到目录位置 < 您的工作目录 >\ app1079\design\work\sdk_repo，如图 2 所示。

处理器间的通信

为 AMP 设计创建应用之前，您需要考虑应用如何进行通信（如有需要）。最简单的方法是使用片上存储器。Zynq SoC 配备 256KB 的片上 SRAM，可从以下四个源地址进行访问：

- 利用侦测控制单元（SCU）从任意内核进行访问；
- 利用 SCU 通过 AXI 加速器一致性端口（ACP）从可编程逻辑进行访问；
- 利用片上存储器（OCM）互联通过高性能 AXI 端口从可编程逻辑进行访问；
- 也是利用 OCM 从中央互联进行访问。

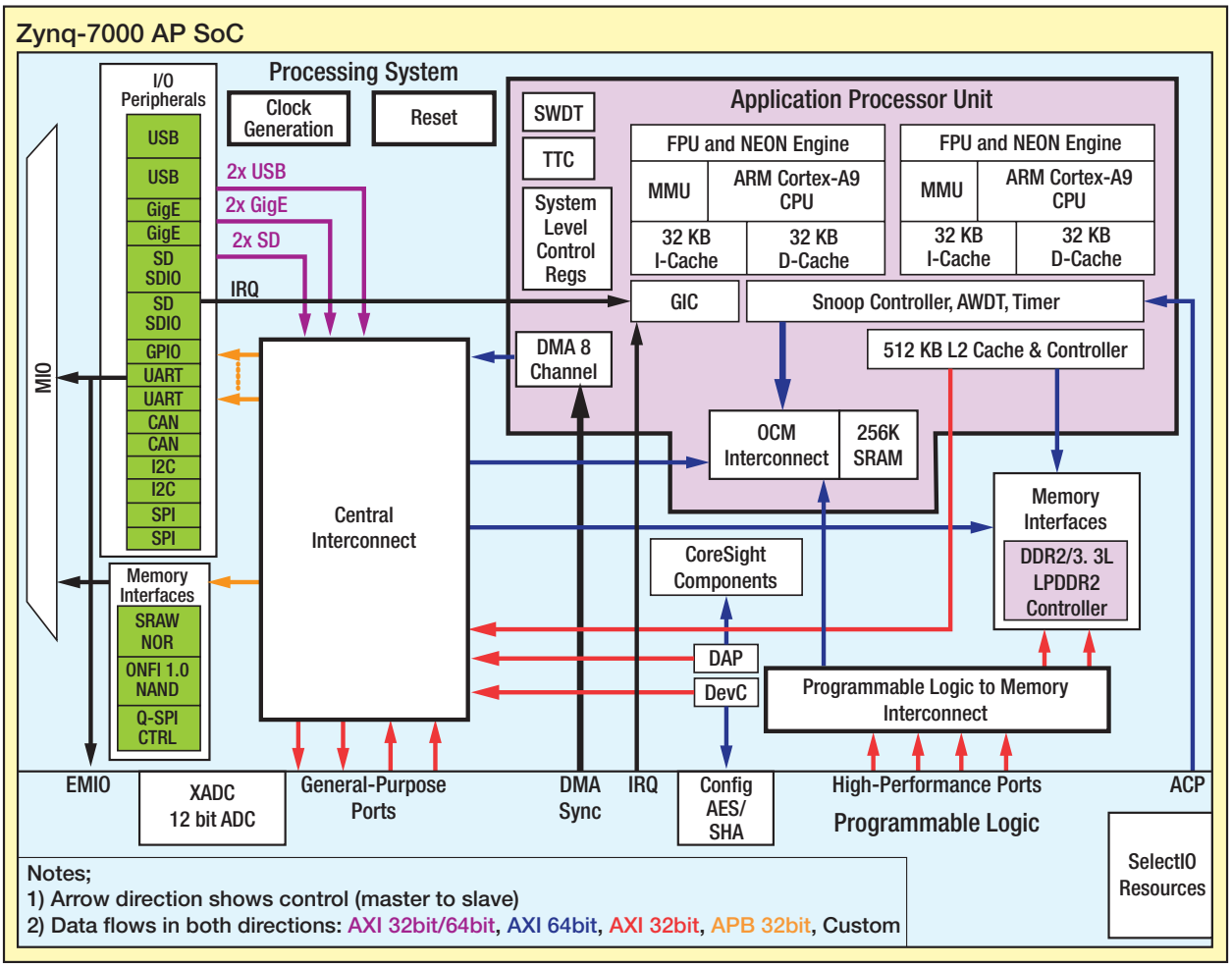


图1 – Zynq SoC处理系统，显示私有和共享资源

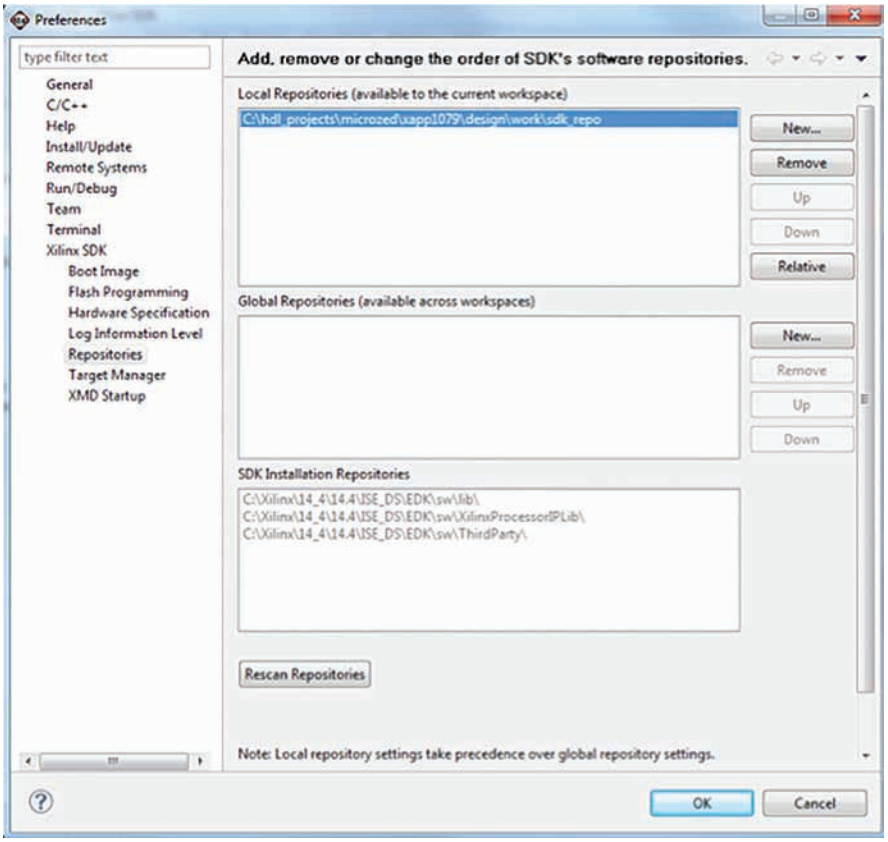


图2 — 将您的新文件添加到库

由于这些不同的访问源都能对片上存储器进行读写，因此尤为重要的一点是，在使用 OCM 之前一定要首先详细了解其的运行方式。

既然 OCM 有多个访问源，那么显然应该定义一个仲裁和优先级形式。由于侦测控制单元需要最低时延（SCU 既可以是处理器内核也可以是 AXI ACP 接口），因此 SCU 从这些访问源的读操作就具有最高优先级，紧接着是 SCU 写操作，然后是 OCM 互联读 / 写操作。用户可通过将片上存储器控制寄存器中的 SCU 写操作的优先级设置为低来颠倒 SCU 写操作和 OCM 互联访问的优先级。

OCM 本身结构为 128 位字，分成四个 64KB 分区，并位于 PS 地址空间的不同位置。初始配置下，前三个 64KB 区块布置在地址空间的起始位置，最后一个 64KB 区块置于地址空间的末尾（见图 5）。

简单的片上存储器实例

您可使用赛灵思 I/O 函数访问 OCM，以便从所选的存储器地址读取和写入数据。这些函数包含在 Xil_IO.h 中，可支持在 CPU 地址空间内存储和访问 8 位、16 位或 32 位字符型、短整型或整型数据。使用这些函数时，只需知道您希望访问的地址以及想要在此存储的值即可。如果是写操作，方法如下，

```
Xil_Out8(0xFFFF0000,0x55);
read_char = Xil_In8(0xFFFF0000);
```

使用该技术时要确保两个地址指向片上存储器中的相同位置，尤其是当不同人编写不同内核程序时更应如此，为此更好的方法是使用共同的头文件。该文件将包含针对特定传输的相关操作地址的宏定义，例如：

```
#define LED_PAT 0xFFFF0000
```

另一种备选方法是让两个程序都使用指示器来访问存储单元。您可以通过使用宏命令定义指向恒定地址的指示器（一般用 C 语言）来实现这一点：

```
#define LED_OP (*(volatile
  unsigned int *)(0xFFFF0000))
```

此外，您还可以对地址再次进行宏定义，以确保该地址为两个应用程序的共用地址。这种方法无需使用赛灵思 I/O 库，而是通过指示器实现简单访问。

处理器间的中断

Zynq SoC 中的每个内核都有 16 个软件生成的中断。如上文所提到的，每个内核都能实现自身与另一个内核或两个内核的中断。使用软件中断与使用硬件中断基本相似，区别只在于您如何触发它们。若使用软件中断，正在接收的应用就无需针对更新数据而对目标存储单元进行轮询。

就像使用任何硬件中断时一样，您需要对两个内核中的通用中断控制器进行配置。敬请参阅《赛灵思中国通讯》第 52 期的“如何在 Zynq SoC 上使用中断”以了解更多相关信息。

然后，您可以使用 xscugic.h 中提供的 XScuGic_SoftwareIntr 函数在正在更新的内核中触发软件中断。该命令将向该指定内核发出一个软件中断，再由该内核进行适当操作：

```
XScuGic_SoftwareIntr(<GIC In-
stance Ptr>, <SW Interrupt ID>,
<CPU Mask>)
```

您必须为内核0和内核1应用对DDR存储器进行正确分段，否则会存在其中一个应用破坏另一个应用的风险。

创建应用

将文件添加到库之后，下个阶段就是生成 AMP 解决方案的三个重要部分：AMP 第一阶段引导载入程序、内核 0 应用和内核 1 应用。您必须为每个部分生成一个不同的板支持包（BSP）。

您需要做的第一件事是用 SDK 创建一个新的 FSBL。选择“新建应用项目”，创建一个支持 AMP 的 FSBL 项目。这与创建一般 FSBL 的过程没有什么不同。不过，这次您需要选择“Zynq FSBL for AMP”模板，如图 3 所示。

完成 AMP FSBL 创建之后，接下来需要为第一个内核创建应用。一定要选择内核 0 和您的首选操作系统，并允许其创建自己的 BSP，如图 4 所示。

创建应用之后，您需要正确定义应用在 DDR 存储器中的位置（应用将从该位置执行）。为此，您需要编辑图 5 中的链接器脚本，以显示 DDR 的基地址和大小。这一点很重要，因为如果没有为内核 0 和内核 1 应用对 DDR 存储器进行正确分段，就会存在其中一个应用破坏另一个应用的风险。

完成分段之后，您现在可以编写希望在内核 0 上执行的应用，因为该内核是 AMP 系统中的主管。内核 0 必须启动内核 1 应用的执行。您需要将图 6 中的代码段包含在应用中。这段代码禁用片上存储器上的高速缓

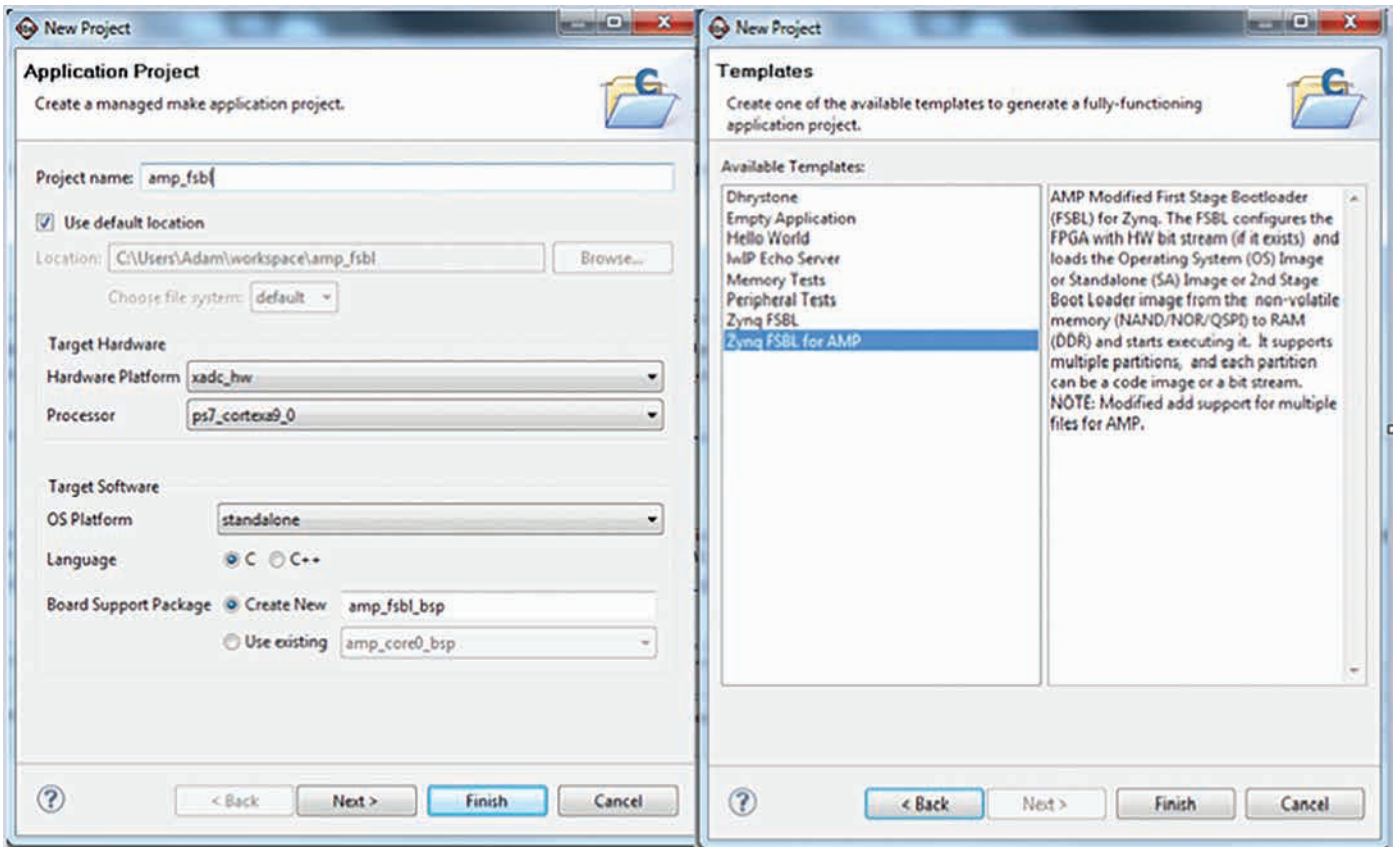


图3 – 为AMP设计选择第一阶段引导载入程序

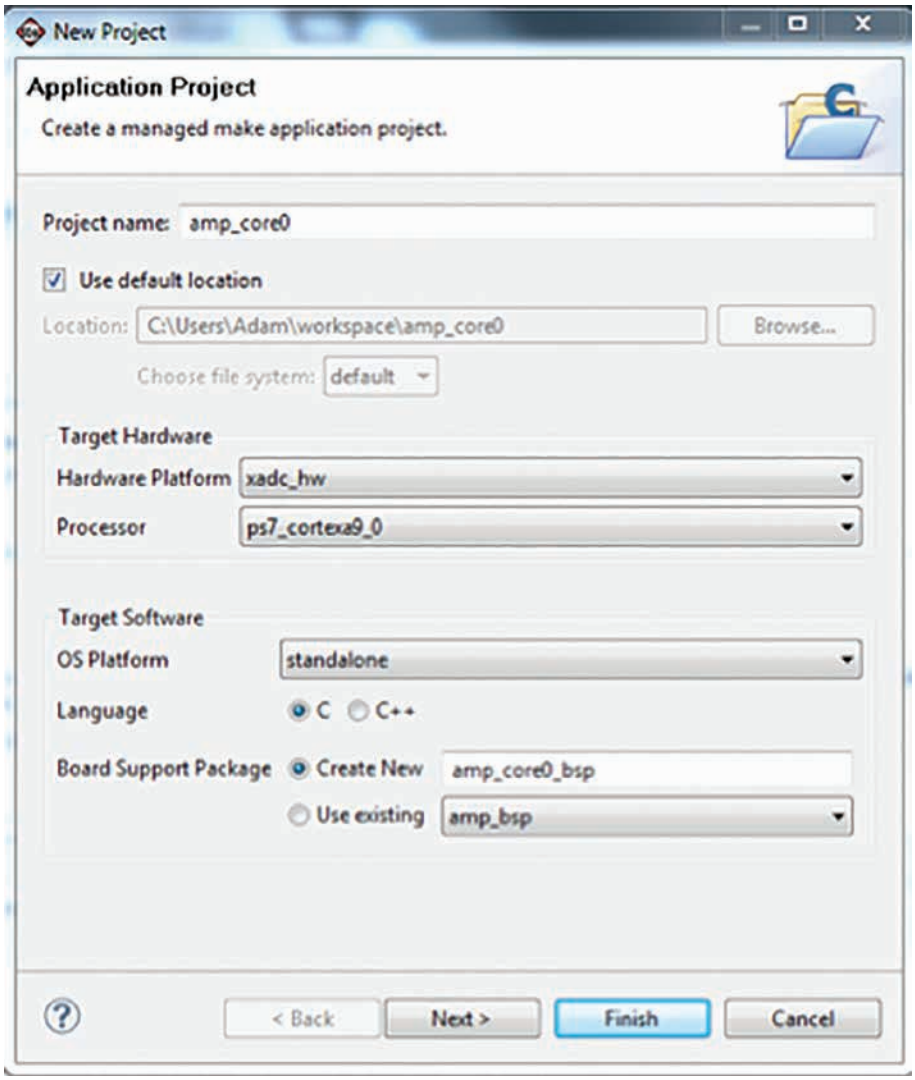


图4 – 为内核0创建应用和BSP

存，并将内核 1 程序的起始地址写到一个内核 1 将会访问的地址。一旦内核 0 执行 Set Event（SEV）命令，内核 1 便开始执行其程序。

下一步是为内核 1 创建 BSP。一定要使用修改后的独立 OS（standalone_amp，如图 7 所示），这一点很重要，因为它能防止 PS 侦测控制单元的重新初始化。就这一点而言，在创建项目时不要像对待内核 0 那样允许其自动生成 BSP。必须确保在 CPU 选项中选择内核 1。

既然您已经为内核 1 创建了 BSP，那么接下来首先需要修改 BSP 的设置，才能继续创建您想要在内核 1 上运行的应用。这非常简单，只需要向 BSP 驱动器部分的配置中添加一个额外的编译器标志：–DUSE_AMP=1。

这一步完成后，您就可以任意为内核 1 创建应用了。务必选择内核 1 作为处理器，并使用您刚刚创建的 BSP。创建新应用之后，您需要再次在 DDR 存储器中定义正确的存储单元，而内核 1 程序将从此处执行。您可按照之前的方法通过编辑内核 1 应用的链接器脚本来完成设定。与第一个内核一样，在该应用中同样要禁用片上存储器上的高速缓存 —— 该高速缓存可用来在这两个处理器之间进行通信。

将所有组件完美整合

在创建应用和构建项目之后，您现在应已拥有以下组件：

- AMP FSBL ELF；
- 内核 0 ELF；
- 内核 1 ELF；
- BIT 文件，用来为预期能够实现 AMP 的 Zynq 器件定义配置。

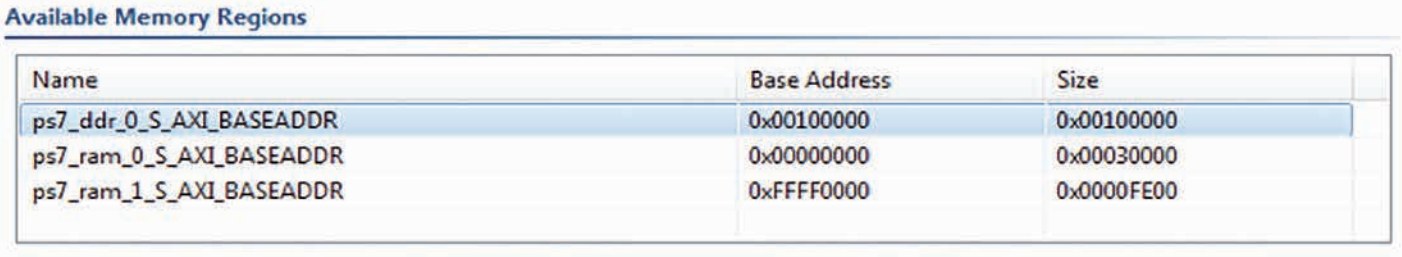


图5 – 内核0的DDR位置和大小

使用所提供的工具在Zynq SoC上创建非对称多处理应用可以变得非常简单。

```
#include <stdio.h>
#include "xil_io.h"
#include "xil_mmu.h"
#include "xil_exception.h"
#include "xpseudo_asm.h"
#include "xscugic.h"

#define sev() __asm__("sev")
#define CPU1STARTADR 0xfffffff0
#define COMM_VAL  (*(volatile unsigned long *)(0xFFFF0000))


int main()
{

    //Disable cache on OCM
    Xil_SetTlbAttributes(0FFFF0000,0x14de2);      // s=b1 TEX=b100 AP=bll, Domain=bllll, C=b0, B=b0
    Xil_Out32(CPU1STARTADR, 0x00200000);
    dmb(); //waits until write has finished
    sev();
```

图6 – 通过编码禁用片上存储器上的高速缓存

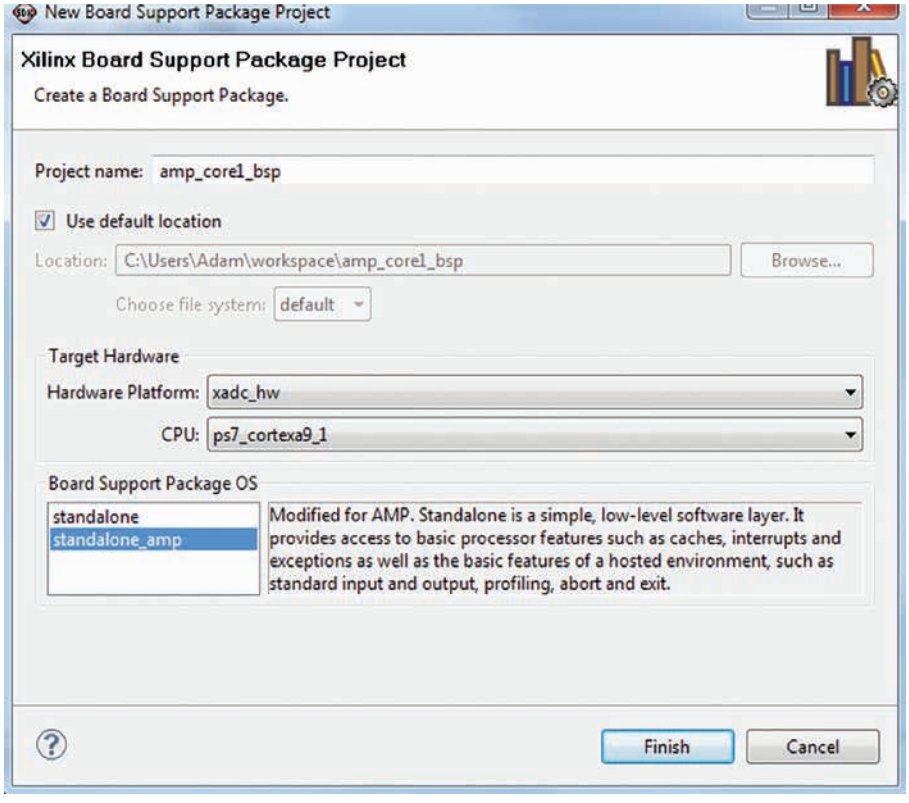


图7 – 为内核1创建BSP

为了使 Zynq SoC 从所选的配置存储器中引导，您需要一个 .bin 文件。要创建该文件，您还需要一个 BIF 文件。BIF 文件规定了应使用哪些文件创建 BIN 文件以及它们的顺序。不要使用 SDK 中的“创建 Zynq”引导映像，而应使用 ISE® 设计套件命令提示符和 BAT 文件（BAT 文件是 XAPP1079 的一部分，位于下载目录 \design\work\bootgen）。该目录包含一个 BIF 文件和一个 cpu1_bootvec.bin，后者作为修改后的 FSBL 的一部分，用于阻止其查找和加载更多应用。

要生成 BIN 文件，您需要将生成的三个 ELF 文件复制到 bootgen 目录，并对 BIF 文件进行编辑以确保其中的 ELF 名称正确无误（如图 8 所示）。

现在您可打开一个 ISE 命令提示符，并导航至 bootgen 目录。在这里运行 createboot.bat。该步骤将创建

boot.bin 文件（如图 9 所示）。

然后，您可将该文件下载到 Zynq SoC 上的非易失性存储器中。该器件的引导将使两个内核启动并执行其各自的程序。

使用所提供的工具在 Zynq SoC 上创建非对称多处理应用可以变得非常简单。使用片上存储器或 DDR 分区可以很容易地实现两个内核之间的通信。

```
the_ROM_image
{

     [bootloader] amp_fsbl.elf
                  download.bit
                    amp_cpu0.elf
                    app_cpu1.elf

  //write start vector address 0xFFFFFFF0 with 0xFFFFFFF
  //This load address triggers fsbl to continue
  [load = 0xFFFFFFF0] cpu1_bootvec.bin
}
```

图8 – 修改BIF文件

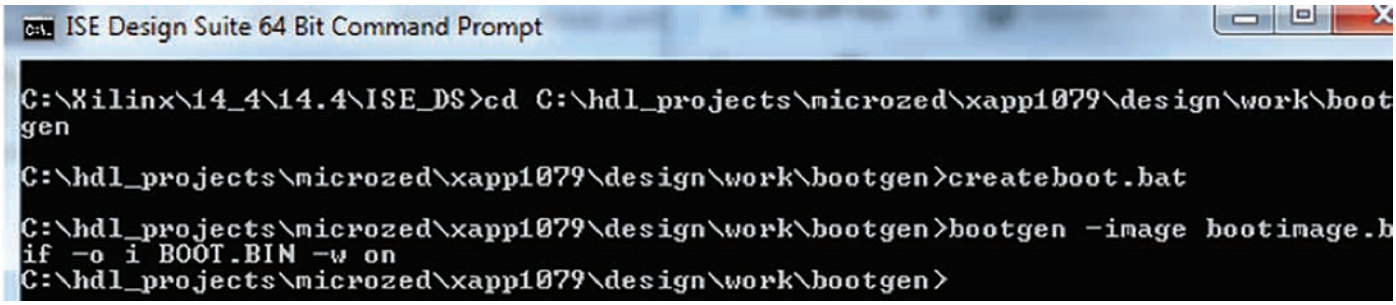


图9 – 创建将在Zynq SoC上运行的boot.bin文件

Xilinx 发布面向全可编程 SoC 和 MPSoC 的 SDSoC™ 开发环境

公司进一步丰富其 SDx 产品系列，并持续将用户群拓展至更广阔的系统及软件工程师社区

2015 年 3 月 9 日，中国北京 - All Programmable 技术和器件的全球领先企业赛灵思公司（Xilinx, Inc. (NASDAQ:XLNX)）今天宣布推出面向全可编程 SoC 和 MPSoC 的 SDSoC ™ 开发环境。作为赛灵思 SDx ™ 系列开发环境的第三大成员，SDSoC 开发环境让更广阔的系统和软件开发者群体也能获益于“全可编程”SoC 和 MPSoC 器件的强大优势。SDSoC 环境可提供大大简化的类似 ASSP 的编程体验，其中包括简便易用的 Eclipse 集成设计环境（IDE）以及用于异构 Zynq® 全可编程 SoC 和 MPSoC 部署的综合开发平台。SDSoC 结合使用业界首款 C/C++ 全系统优化编译器，可提供系统级特性描述、利用可编程逻辑实现软件自动加速、自动系统连接生成，以及各种库以加速编程工作。此外，它还能帮助最终用户和第三方平台开发人员快速定义、集成和验证系统级解决方案，并可通过定制编程环境为最终客户提供支持。

类似 ASSP 的编程体验：系统和嵌入式软件工程师采用 SDSoC，可以将运行在裸机或 Linux 和 FreeRTOS 等操作系统上的 C/C++ 作为 Eclipse IDE 的输入。

全系统优化的编译器：SDSoC 可针对 ARM 处理器和可编程逻辑提供全系统优化的编译器。此外，SDSoC 可提供赛灵思库以及赛灵思联盟成员 Auviz Systems 公司可选硬件优化库。

系统级的特性描述：SDSoC 可以快速估算系统性能，能较早地快速生成并探索最佳整体系统性能和功耗。

面向平台开发人员的专家级使用模型：SDSoC 可为 Zynq 全可编程 SoC 开发板（如 ZC702、ZC706 等）以及第三方及市场特定平台（如 Zedboard、MicroZed、ZYBO 和视频图像开发套件）提供板支持包（BSP）。帮助软件开发人员和系统架构师抽象平台细节，从而简化了异构化更智能系统的创建、集成与验证工作。利用赛灵思提供的或客户创建的平台，SDSoC 都能实现真正的软件可配置更智能系统。

如何将PetaLinux移植到Xilinx FPGA上

作者：
Sweta
通信工程系研究生
印度班加罗尔PES理工学院（PES Institute of Technology）
sweta.v.walikar@gmail.com

Srikanth Chintala
卫星和无线部研发工程师
印度班加罗尔远程信息处理开发中心（C-DOT）
chintala@cdot.in

Manikandan J
电子通信工程系（EC）信号处理领域教授和领域带头人
研究创新孵化学（CORI）教授
印度班加罗尔PES大学（PES University）
manikandanj@pes.edu

用户可轻松将这款高稳健操作系统安装到目标 FPGA 平台上，以供嵌入式设计项目使用。

从最初不起眼的胶合逻辑开始，FPGA 已经历了漫长的发展道路。当前 FPGA 的逻辑容量和灵活性已将其带入了嵌入式设计的中心位置。目前，在单个可编程芯片上可实现一个完整系统，这种架构有助于软硬件的协同设计，并能将软硬件应用进行集成。

这些基于 FPGA 的嵌入式设计种类需要稳健的操作系统。PetaLinux 应运而生，已成为众多嵌入式设计人员青睐的对象。它以开源免费的方式提供，支持包括赛灵思 MicroBlaze® CPU 和 ARM® 处理器在内的多种处理器架构。要将 PetaLinux 移植到特定的 FPGA 上，必须针对目标平台定制、配置和构建内核源代码、引导载入程序、器件树和根文件系统。

对于 PES 大学和 C-DOT 的一个设计项目而言，我们的研发团队准备移植 PetaLinux 并在采用 Kintex®-7 XC7K325T FPGA 的赛灵思 KC705 评估板上运行多个 PetaLinux 用户应用。结果证明整个过程相当便捷。

选择PetaLinux的原因

在详细介绍具体做法之前，有必要花点时间来探讨针对基于 FPGA 的嵌入式系统提供的操作系统选项。PetaLinux 是 FPGA 上最常用的操作系统，另外还有 µClinux 和 Xilkernel。µClinux 为 Linux 发行版，是一款包含小型 Linux 内核的移植型 Linux 操作系统，适用于无存储器管理单元（MMU）的处理器 [1]。µClinux 配备有各种库、应用和工具链。Xilkernel 就其本身而言，是一款小型、高稳健性、模块化内核，能够提供高于 µClinux 的定制性能，有助于用户通过定制内核来优化其设计尺寸与功能 [2]。

同时，PetaLinux 也是一款完整的 Linux 发行版及开发环境，适用于基于 FPGA 的片上系统（SoC）设计。PetaLinux 包含预配置二进制可引导映像、面向赛灵思器件的完全可定制 Linux 以及配套提供的 PetaLinux 软件开发套件（SDK）[3]。其中 SDK 包括用于自动完成配置、构建和部署过程中各种复杂工作的工具和实用程序。赛灵思提供可免费下载的 PetaLinux 开发包，其中包括针对各种赛灵思 FGPA 开发套件而设计的硬件参考项目。同时包含在内的还有适用于赛灵思 FPGA 的内核配置实用程序、交叉编译器等软件工具、硬件设计创建工具以及大量其它设计辅助功能。

据报道，Xilkernel 的性能优于 µClinux[4]，而 PetaLinux 的性能又优于 Xilkernel [5]。由于这个原因，特别是由于已针对我们赛灵思目标板提供的软件包原因，我们为我们的项目选择了 PetaLinux。移植 PetaLinux 的另一大优势是用户可以轻松实现远程编程。这就意味着用户可使用远程接入方式，通过远程登录，采用新的配置文件（或比特流文件）加载 FPGA 目标板。

有两种方法可以创建用于构建PetaLinux系统的软件平台：在Linux终端上使用PetaLinux命令或通过下拉菜单使用GUI。

开始安装

下面详细介绍我们项目团队安装 PetaLinux 的方法。第一步，我们下载了 PetaLinux 软件包 12.12 版以及用于 Kintex-7 目标板的电路板支持包（BSP）。然后运行了 PetaLinux SDK 安装程序，并在控制台上使用下列命令把 SDK 安装到了 /opt/Petalinux-v12.12-final 目录下：

```
@ cd /opt
@ cd /opt/PetaLinux-v12.12-final-full.tar.gz
@ tar zxf  PetaLinux-v12.12-final-full.tar.gz
```

随后，我们把从赛灵思网站获得的 PetaLinux SDK 许可证复制并拷贝到 .xilinx 和 .Petalogix 文件夹中。接下来，我们使用下列命令获取适当设置，设置了 SDK 的工作环境：

```
@ cd /opt/PetaLinux-v12.12-final
@ source settings.sh
```

为验证工作环境是否设置正确，我们使用了以下命令：

```
@ echo $PETALINUX
```

如果环境设置正确，将显示 PetaLinux 的安装路径。

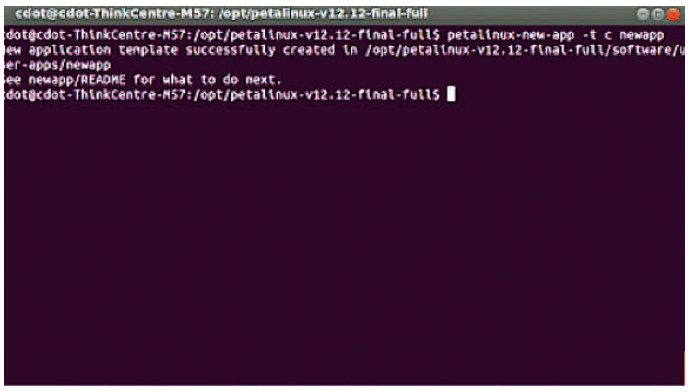


图1：用于用户设置的Linux终端窗口截屏

在本案例中，PetaLinux 的安装路径是 /opt/PetaLinux-v12.12-final。

接下来的工作是安装 BSP，其中包含必要的设计文件、配置文件和预构建软硬件包。这些软硬件包已经通过测试，可随时下载到目标板上。另外软件包还可用于在快速仿真器（QEMU）系统仿真环境下的引导。为了安装 BSP，我们在 path /opt 中创建了一个名为“bsp”的文件夹，并使用下列命令复制了 KC705 BSP 的 ZIP 文件：

```
@ cd  /opt/PetaLinux-v12.12-final-full
@ source settings.sh
@ source /opt/Xilinx/14.4/ISE_DS/settings32.sh
@ PetaLinux-install-bsp /bsp/Xilinx-KC705
    -v12.12-final.bsp
```

构建为新平台定制的 PetaLinux 系统，有两种创建和配置软件平台的方法。一种方法是使用 Linux 终端，在 PetaLinux 命令对应的路径位置使用 PetaLinux 命令，如图 1 所示。第二种方法是通过下拉菜单使用 GUI，如图 2 所示。您可使用其中任何一种方法来选择平台，配置 Linux 内核，配置用户应用和构建镜像。在操作系统安装完成后，就可使用 PetaLinux 控制台。而使用 GUI 则需要完成 PetaLinux SDK 插件的安装。完成该插件的安装后，就可使用 PetaLinux Eclipse SDK 中提供的 PetaLinux GUI 设置各种配置（图 2）。该 GUI 具有各种特性，如用户应用和库开发，以及 PetaLinux 及硬件平台的调试、构建和配置等。

硬件构建

我们为项目使用了基于 Kintex-7 FPGA 的 KC705 评估板。设计需要的硬件接口有用于监控输出的 RS232 接口、用于编程 FPGA 的 JTAG 接口以及用于远程编程的以太网接口。除了 PetaLinux SDK，所推荐设计需要的其它软件还

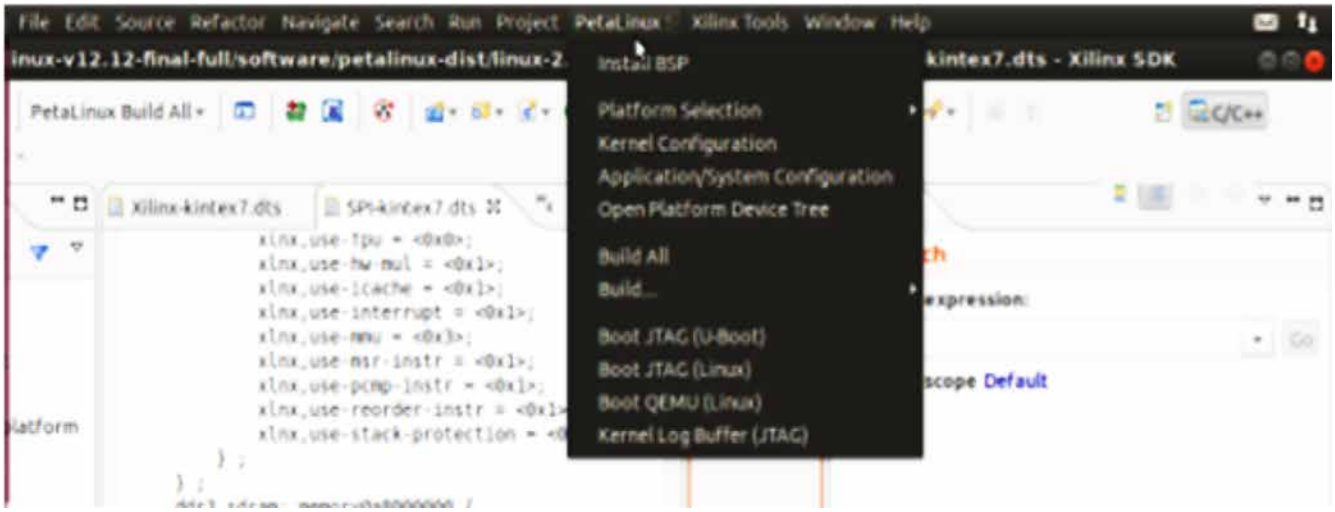


图2：用于用户设置的PetaLinux SDK菜单截屏

包括 Xil-inx Platform Studio (XPS) [6,7] 和赛灵思软件开发套件（SDK）[7]。

在该嵌入式设计的硬件部分，我们的第一项任务就是使用 XPS 中的基本系统构建器（BSB）设计基于 MicroBlaze 处理器的硬件平台。BSB 允许选择目标板上提供的一系列外设。您还可根据应用需求添加或删除外设。我们所推荐应用采用的内核或外设集包括带 8Mb 存储器的外部存储器控制器、在中断情况下启用的定时器、波特率为 115,200Bps 的 RS232 UART、以太网、非易失性存储器以及 LED。完成选择后，我们就获得了硬件外设及其总线接口（图 3）。对于基于 MicroBlaze 处理器的设计，PetaLinux 需要支持 MMU 的 CPU。因此我们在 XPS 窗口中双击 MicroBlaze_0 实例，选择了带 MMU 的低端 Linux。

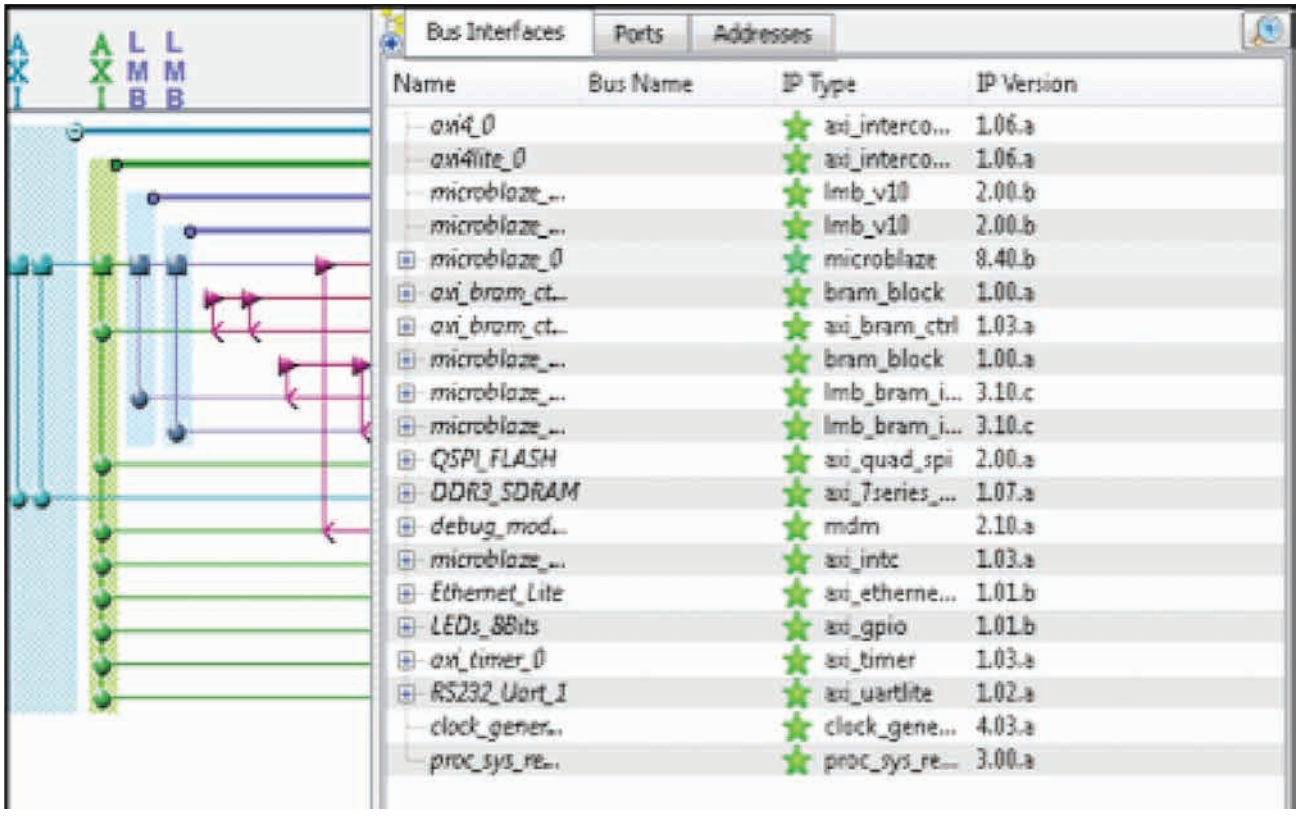


图3：FPGA的硬件配置

此时硬件设计已完成。现在可以使用第一阶段引导载入程序引导该内核。

接下来，我们使用三步转换流程将硬件配置转换为比特流。首先，我们使用 XPS 生成了代表嵌入式硬件平台的网表。随后，我们将设计映射到 FPGA 逻辑中。最后我们将实现的设计转换为能够下载到 FPGA 上的比特流。XPS 的最终输出是 system.bit 和 sys-tem_bd.bmm 文件。

生成比特流后，我们将硬件平台描述导出到 SDK，以便在 SDK 中观察目标硬件平台。导出的系统 xml 文件包含 SDK 编写应用软件并在目标硬件平台上对其进行调试所需的信息。我们的下一项任务是使用 Xilinx Tools → Repository → New 在 SDK 中添加一个 PetaLinux 库，然后选择 PetaLinux 的安装路径。在本实例中，该路径为 $PetaLinux/Hardware/edk_user_repository。

接下来，我们使用 File → Board support package → PetaLinux 创建了 PetaLinux BSP。我们根据所需的应用选择必要的驱动程序，配置了 PetaLinux BSP。随后我们通过构建 BSP 并创建和配置第一阶段的引导载入程序应用（fs-boot），引导了内核。该 BSP 可建立硬件和引导应用之间的交互。SDK 的输出为 fs-boot.elf。可使用数据到存储器转换器命令 data2mem 将 system.bit、system_bd.bmm 和 fs-boot.elf 合并为一个名为 download.bit 的统一比特流文件，用作最终的 FPGA 比特流。

此时硬件设计已完成，其它方面还包括一个 MicroBlaze 内核和运行其上的 PetaLinux 操作系统。现在我们可以使用第一阶段的引导载入程序引导内核。

构建软件

完成硬件平台的构建后，我们使用下列命令创建了针对硬件的定制 PetaLinux 软件平台：

```
$ cd/opt/PetaLinuxv12.12
$  PetaLinux-new-platform -c <CPU-ARCH> -v
   <VENDOR> -p <PLATFORM>
```

其中 –c <cpu-arch> 为支持的 CPU 类型（这里是 MicroBlaze 处理器）、–v <vendor> 为厂商名称（这里是赛灵思），而 –p <platform> 则为产品名称（这里是 KC705）。软件平台的配置文件在安装 PetaLinux 的目录下生成，即 */opt/PetaLinuxv12.12/software/ PetaLinux-dist/vendors/Xilinx/ KC705*。

为定制与硬件匹配的软件平台模板，我们使用 *Peta-Linux-copy-autoconfig* 命令将现有平台配置与内核配置进行了合并。该命令可生成硬件配置文件 *Xilinx-KC705.dts*, *xparameters.h* 和 *config.mk*。

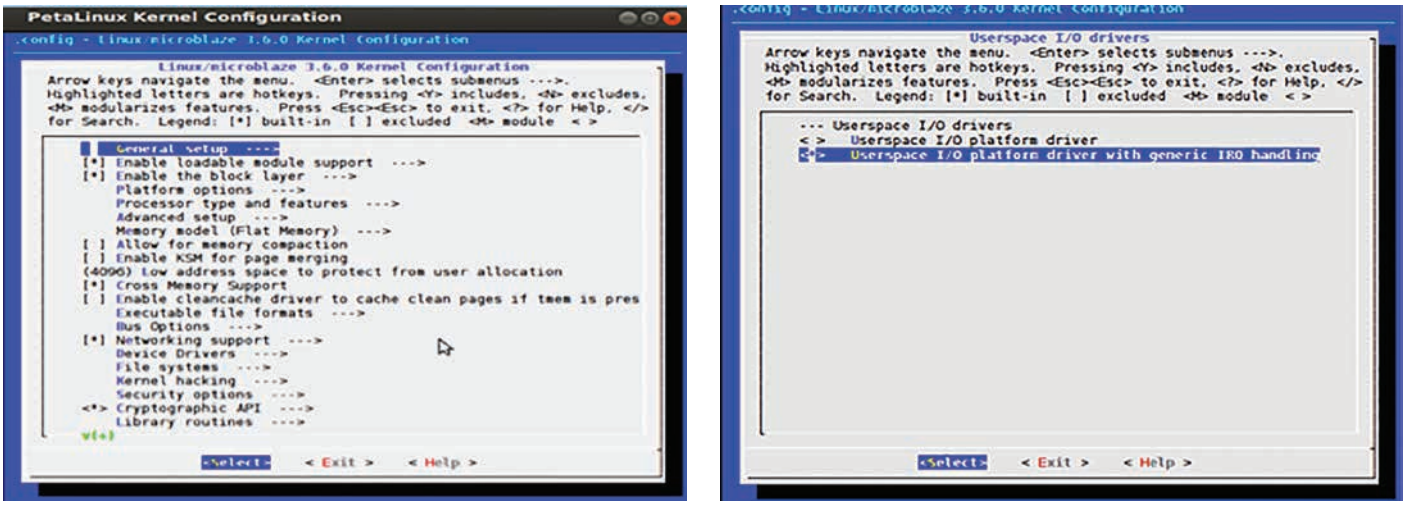


图4：内核配置菜单

我们使用 GUI（PetaLinux SDK → Kernel Configuration）打开内核配置菜单，配置了 Linux 内核。此外，您也可以在 Linux 终端上使用下列命令完成该工作：

```
$ cd /opt/PetaLinux_v12.12
$  PetaLinux-config-kernel
```

我们在内核配置弹出窗口中启用该应用的驱动程序（如图 4 所示）。为通过用户空间输入 / 输出（UIO）接口访问设备，完成所提出的工作，我们在内核配置菜单中启用了 UIO 驱动程序。

内核配置完成后，我们设计了一些应用。PetaLinux 可提供用于 C 语言和 C++ 编程的用户应用模板 [8]。这些模板包括应用源代码和 Makefile 文件，方便为目标芯片配置和编译应用并将其安装在根文件系统中。创建新的 PetaLinux 用户应用，既可使用 GUI（File → PetaLinux New Application），也可在 Linux 终端上输入下列命令：

```
$ cd /opt/PetaLinux_v12.12
$  PetaLinux-config-apps
```

随后我们为该用户应用起了个文件名。在本实例中，我们创建了 gpio-dev-mem-test 和 gpio-uio-test 用户应用，并根据应用要求修改了模板源代码。

接下来我们使用 GUI 构建了 PetaLinux 系统映像（如图 2 所示）。此外，您还可以在 Linux 终端上使用 make 命令完成该任务，如下图所示：

```
$ cd $PETALINUX/software/ PetaLinux-dist
$ make
```

支持操作系统（OS）和定制用户应用的软件平台以及我们前文讨论过的硬件设计现已可供使用。

测试运行在设备上的PetaLinux

下面介绍 PetaLinux 的引导方式。MicroBlaze 处理器可处理驻留在 Block RAM 中的代码。第一阶段的引导载入程序（fs-boot）将初始化基本硬件、执行 fs-boot.elf、搜索通用引导载入程序或 U-Boot、在闪存分区中进行寻址（因为 U-Boot 的地址已在配置 fs-boot 时设定）。随后，fs-boot 将从闪存中的 U-Boot 分区中获取 U-Boot 映像，将其发送到设备的 DDR3 存储器并运行内核。一旦构建好所有引导所需的映像后，您就可以通过 JTAG、以太网或快速仿真器在硬件上测试这些映像了。QEMU 是一种仿真器和虚拟机，允许您运行 PetaLinux 操作系统 [9]。下面讨论所有这三种解决方案的引导方法。

JTAG 是编程和测试 FPGA 设计的传统方法。为使用 JTAG 对 FPGA 进行编程，我们使用了下拉菜单“Xilinx Tool → Program the FPGA”并下载了之前生成的 download.bit 文件。随后我们使用 GUI（PetaLinux SDK → BOOT JTAG [Li-nux]）将映像下载到了电路板上，如图 2 所示。您也可以在 Linux 终端上使用下列命令：

```
$ cd/opt/PetaLinux_v12.12/software/
  PetaLinux-dist
$  PetaLinux-jtag-boot -i images/image.elf
```

此外，您还可使用 U-Boot 执行间接内核引导，从而引导 PetaLinux。系统首先使用 GUI（PetaLinux SDK → BOOT JTAG [U-Boot]）或以下命令通过 JTAG 接口下载 U-Boot 来进行引导。

```
$ cd $PETALINUX/software/ PetaLinux-dist
$  PetaLinux-jtag-boot -i images/u-boot.elf
```

图 6 是 U-Boot 控制台的快照。

值得注意的是，FPGA 电路板连接的是以太网接口。您必须在 XPS 的硬件资源部分选择以太网接口。一旦 U-Boot 引导成功，就要检查服务器和主机的 IP 地址是否相同。如果 IP 地址不同，请在 U-Boot 终端上使用下列命令设置主机 IP。

```
u-boot>print serverip   // prints
192.168.25.45(server ip)
u-boot>print ipaddr     // prints IP address of
the board as
 // 192.168.25.68
u-boot>set serverip <HOST IP> // Host IP
192.168.25.68
u-boot>set serverip 192.168.25.68
```

现在服务器（PC）和主机（KC705 电路板）具有相同的 IP 地址。请通过服务器运行网络引导命令，下载 PetaLinux 映像和引导程序：

```
u-boot> run netboot
```

运行网络引导命令后，您应该能够看到 PetaLinux 控制台，如图 5 所示。

最后您可使用 GUI（PetaLinux SDK → BOOT QEMU [Linux]）或以下命令执行内核引导，这也很重要。

```
$ cd $ PETALINUX/software/ PetaLinux-dist
$ PetaLinux-qemu-boot -i images/image.elf
```

使用这种快速方法，我们将看到图 7 所示信息。

测试运行在设计上的应用

完成 PetaLinux 引导的测试后，接下来就是测试专为 PetaLinux 设计的用户应用。MicroBlaze 处理器将 Kintex-7 FPGA 电路板上的硬件外设视为一组存储寄存器。每个寄存器都有自己的基址和结束地址。要访问一个外设，用户必须知道它的基址和结束地址。您可以在设备树源（*.dts）文件中找到有关地址的详细信息。就本设计而言，我们开发并测试了四款应用，分别是访问 DDR3、使用 /dev/mem 访问 GPIO、使用 UIO 访问 GPIO 和文件传输。

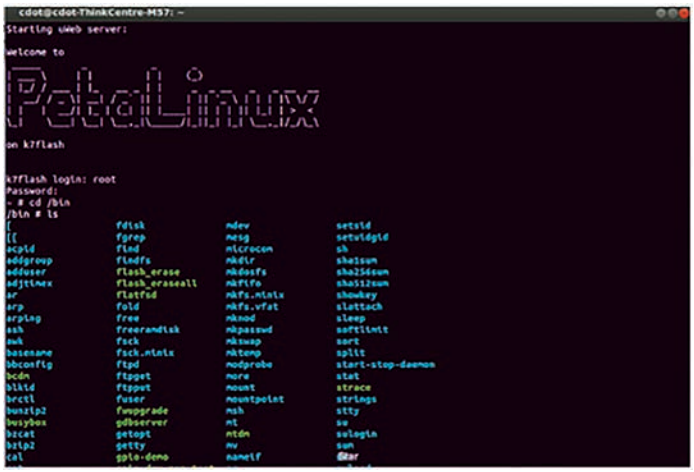


图5：确认操作系统引导成功的PetaLinux控制台快照

1. 访问 DDR3

我们使用名为 DDR3-test.c 的 PetaLinux 应用访问 DDR3 存储器。该应用经过精心设计，可向 DDR 存储器位置写入数据并从这里读取数据。DDR3 是双列直插式存储器模块，可提供用于存储用户代码和数据的 SDRAM。如上文所述，用户需要知道 DDR 存储器的开始地址和结束地址，分别是 0xC0000000 和 0xC7FFFFFF。存储器的容量为 512 兆字节。Linux 内核驻留在 DDR 存储器的初始存储器位置。因此需要选择 DDR3 存储器的写入位置，以避免破坏 Linux 内核。我们使用以下命令向 DDR3 存储器写入数据：

```
#DDR3-test -g 0xc7000000 -o 15
```

其中 DDR3-test 是应用名称、-g 是 DDR3 存储器的物理地址、-o 是输出、15 是准备在 0xc7000000 位置写入 DDR3 存储器的值。为测试该值是否能写入预计的位置，我们使用以下命令从 DDR3 存储器读取数据：

```
#DDR3-test -g 0xc7000000 -i
```

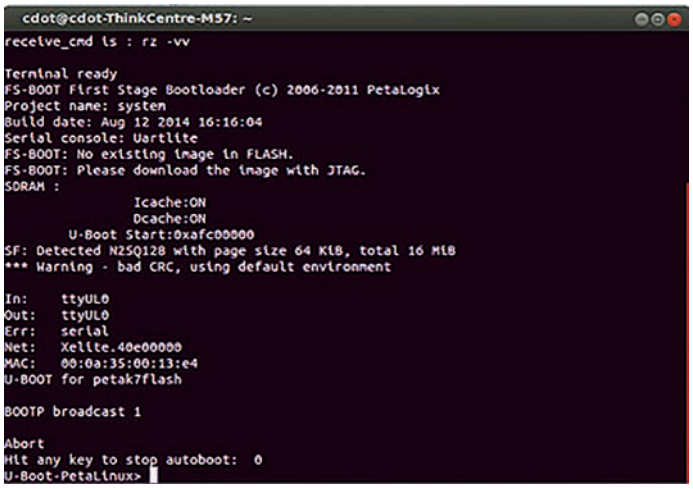


图6：通过通用引导载入程序（U-Boot）进行的间接内核引导

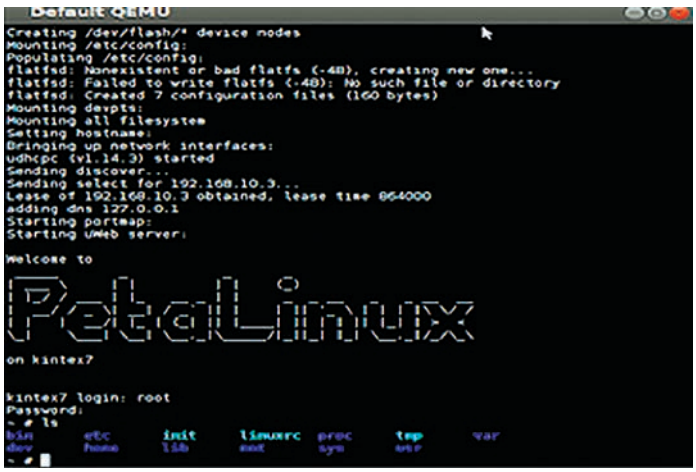


图7：通过QEMU运行PetaLinux

该应用旨在控制8位离散输出，可通过将板载LED连接至GPIO进行测试。

值 15 显示在终端上，这说明 DDR3 存储器读写操作正在成功进行。

2. 使用 /dev/mem 访问 GPIO

对于接下来的应用测试，我们使用名为 gpio-dev-mem-test.c 的 PetaLinux 应用访问了通用 I/O（GPIO）。该应用的设计目的是控制 8 位离散输出并通过将板载 LED 连接至 GPIO 来测试该输出。要从用户空间访问任何设备，就要先打开 /dev/mem，然后使用 mmap() 将设备映射至存储器。我们所使用 LED GPIO 的开始地址和结束地址分别是 0x40000000 和 0x4fffffff。

GPIO 外设具有两个寄存器：数据寄存器（GPIO_DATA）和方向寄存器（GPIO_TRI_OFFSET）。为了读取 GPIO 的状态，我们将方向位设置为 1（即 GPIO_TRI_OFFSET=1）并且从数据寄存器读取数据。为了将数据写入到 GPIO，我们设置方向位为 0 并写入值到数据寄存器。在 PetaLinux 终端上使用下列命令将数据写入到 GPIO：

```
#gpio-dev-mem-test -g 0x40000000 -o 255
```

其中 gpio-dev-mem-test 为应用名称，-g 为 GPIO 物理地址，-o 为输出，255 为从 GPIO（连接到 LED）发送的值。LED 按编写的程序点亮时，测试的结果就得到了验证。

3. 使用 UIO 访问 GPIO

访问 GPIO 的另一个途径是通过用户空间输入 / 输出。我们通过 UIO，使用名为 gpio-uio-test.c 的 PetaLinux 应用访问了 GPIO。该应用旨在控制 8 位离散输出，可通过将板载 LED 连接至 GPIO 进行测试。UIO 设备在文件系统中表现为 /dev/uioX。为通过 UIO 访问 GPIO，我们打开了 /dev/uioX 或 sys/class/ui0，然后使用了 mmap() 调用。我们配置了内核使之支持 UIO，并在内核中启用了 UIO 框架。随后我们使用名为“Compatibility”的参数，根据 UIO 设备（而非标准 GPIO 设备）对 LED 的 GPIO 控制方式进行了设置。此外，我们还将设备的标签从 gpio@40000000 修改成了 leds@40000000。

然后我们重新构建了 PetaLinux，并使用 UIO 测试了 GPIO 访问。我们使用以下命令，获得了所加载 UIO 模块的详细信息：

```
# ls /sys/class/uio/
 uio0 uio1 uio2
```

UIO 的名称和地址可在 /sys/class/uio/uioX 下找到。我们使用以下命令通过 UIO 驱动程序访问了 GPIO LED：

```
# cd "/sys/class/uio/uioX
# gpio-uio-test -d /dev/uio1 -o 255
```

其中 gpio-uio-test 为应用名称、-d 为设备路径、-o 为输出、255 为通过 UIO 传递给 GPIO 的值。使用以上命令，LED 按写入到 GPIO 线路上的数据点亮，验证了该结果。

4. 文件传输应用

最后一项测试，我们将文件从服务器传输到了客户端，这里的服务器是主机 PC，客户端是 KC705 电路板。在这项测试中，我们使用以太网线缆连接服务器和客户端，并使用了小型文件传输协议（TFTP）。这种协议因简单而出名，通常用于自动传输配置文件或引导文件。为测试使用 TFTP 从服务器向客户端传输文件的情况，我们在 /tftpboot 位置为服务器 PC 创建了一个名为 test 的文件。我们使用以下命令在文件中写入了“世界，你好”并查看了该文件中的内容（如图 8 所示）：

```
@ echo "Hello World" > /tftpboot/test
@ more /tftpboot/test
```

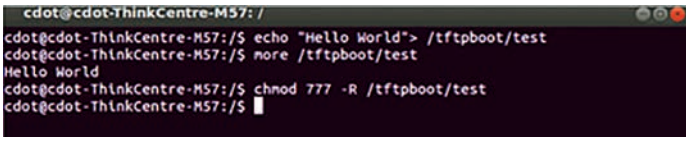


图8：在服务器中创建文件的快照

为从服务器接收该文件，我们在以客户端方式运行在 KC705 电路板上的 PetaLinux 终端窗口中输入以下获取命令（-g）：

```
# tftp -r test -g 192.168.25.68
# ls -a
```

在客户端中创建了一个名为“test”的新文件（如图 9 所示）。我们可以使用更多内容命令查看该文件的内容，如图 9 所示：

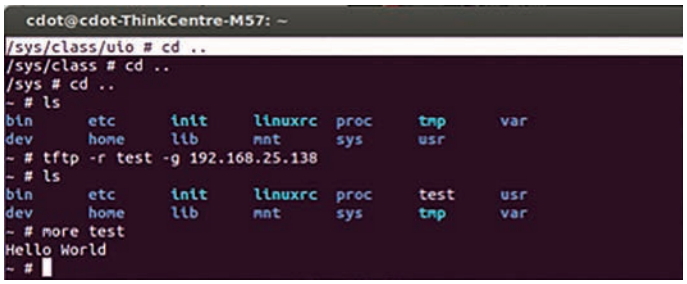


图9：在客户端接收文件的快照

同样，如果要从客户端向服务器传输文件，可先在客户端机器上创建一个名为 test1 的文件，其内容为“PetaLinux OS”。然后在运行在客户端上的 PetaLinux 终端中使用以下“放置”命令（-p），便可将该文件从客户端传输至服务器，如图 10 所示：

```
# tftp -r test1 -p 192.168.25.68
```

图10：从客户端到服务器传输文件的快照

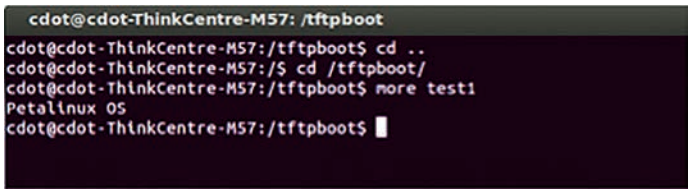


图11：在服务器中接收文件的快照

可在服务器中创建一个空白文件 test1，其内容可在文件传输工作完成后读取。该内容可图 11 所示方法进行验证。

在 FPGA 上实现嵌入式系统和运行 PetaLinux，操作起来非常简单直观。接下来，我们计划使用远程编程实现设计，即引导文件通过以太网传输，客户端能够运行新应用。

参考资料

1. Kynan Fraser，《运行 UClinux 的 MicroBlaze》，摘自《高级计算机体系架构》：*http://www.cse.unsw.edu.au/~cs4211*
2. 赛灵思公司 Xilkernel 3.0 版，2006 年 12 月
3. 赛灵思公司《PetaLinux SDK 用户指南》UG976，2013 年 4 月
4. Gokhan Ugurel 和 Cuneyt F. Bazlamacci，《在 MicroBlaze 上 Xilkernel 和 µ C/OS-II 上下文切换时间及内存占用比较》，摘自 2011 年 12 月土耳其布尔萨的《第 7 届电子电气工程国际学术会议》第 52 页至 55 页。
5. Chenxin Zhang、Kleves Lamaj、Monthadar Al Jaberi 和 Praveen Mayakar，《极小型网络附加存储（NAS）》”，摘自隆德大学机械学院（Lunds Tekniska Hogskola）《项目报告，高级嵌入式系统课程》2008 年 11 月
6. 赛灵思公司《Platform Studio 用户指南》UG113，1.0 版，2004 年 3 月
7. 赛灵思公司《EDK 概念、工具和技巧：高效嵌入式系统设计实际操作指南》UG683，14.1 版本，2002 年 4 月
8. 赛灵思公司《PetaLinux 应用开发指南》UG981，2013 年 4 月
9. 赛灵思公司《PetaLinux QEMU 仿真指南》UG982，2013 年 11 月

依元素科技培训课程时间表 2015/4 至 2015/6

培训课程	培训时间	4月	5月	6月
Vivado设计套件工具流程	1天	1日 上海	6日 北京	3日 深圳
		8日 成都	13日 深圳	10日 武汉
基于ISE软件工程导航器用户的Vivado设计套件	1天	8日 武汉	20日 北京	17日 武汉
基于Vivado的FPGA设计基础	2 天	13-14日 北京	18-19日 西安	11-12日 深圳
Vivado设计套件的高级工具和技术	2天	20-21日 成都	11-12日 北京	18-19日 上海
Vivado设计套件静态时序分析和Xilinx设计约束	2天	13-14 深圳	14-15日 北京	11-12日 上海
面向软件设计人员的基于 C 语言的 HLS 编码	1天	8日 西安	27日 北京	24日 西安
面向硬件设计人员的基于 C 语言的 HLS 编码	1天	1日 武汉	6日 上海	3日 武汉
基于C语言设计：Vivado HLS高层次综合	2 天	27-28日 西安	4-5日 上海	25-26日 深圳
		2-3日 北京	18-19日 南京	1-2日 北京
Zynq全面可编程SoC架构介绍	1天	22日 北京	13日 上海	10日 北京
Zynq 全面可编程SoC系统架构	2 天	16-17日 深圳	21-22日 北京	15-16日 深圳
		13-14日 上海	28-29日 成都	11-12日 西安
嵌入式系统开发	2 天	20-21日 成都	25-26日 北京	18-19日 深圳
嵌入式系统软件开发	2 天	27-28 深圳		29-30 深圳
嵌入式开放源码 Linux 开发	2天		18-19日 上海	
高级嵌入式系统软件设计	2 天	23-24日 上海	28-29日 深圳	22-23日 上海
基于ISE的FPGA 设计基础	1 天		27日 深圳	
利用 Vivado逻辑分析器的调试和验证	2天	27-28日 深圳	11-12日 上海	25-26日 杭州
面向性能的设计	2 天	9-10日 上海		8-9日 上海
高级 FPGA 设计	2 天	16-17日 武汉		15-16日 上海
使用 PlanAhead 分析与设计工具进行基本设计	1 天	22日 北京	13日 西安	17日 北京
使用 PlanAhead 分析与设计工具进行高级设计	2 天	27-28 日 武汉	21-22日 上海	15-16 日 西安
Xilinx 部分重配置工具和技术	2 天	23-24 北京	28-29日 上海	22-23 北京
利用 Spartan-6 和 Virtex-6 系列进行设计	2 天	9-10日 深圳		8-9日 深圳
使用 7 系列产品进行设计	2天	23-24日 深圳	4-5日 北京	22-23日 深圳
		27-28日 北京	11-12日 上海	29-30日 北京
Xilinx FPGA 的信号完整性和电路板设计	2 天		25-26日 上海	
设计LogiCORE PCI Express 系统	2 天	2-3日上海	14-15日 北京	1-2日上海
利用以太网 MAC 控制器进行设计	2天		25-26日 北京	
利用千兆位级串行 I/O 进行设计	2 天	9-10日 北京	21-22日 深圳	8-9日 北京
利用System Generator 进行DSP设计	2 天	29-30日 北京	18-19日 深圳	29-30日 武汉
Xilinx FPGA 的基本 DSP 实现技术	2 天		20日 南京	
利用 VHDL 进行设计	2天		27日 成都	
利用 Verilog 进行设计	2天	15日 深圳	25-26日 上海	8-9日 福州
Vivado设计套件的基本Tcl脚本	1天	22日 武汉	20日 深圳	3日 成都
UltraFast设计方法	1天	1日 北京	6日 西安	10日 深圳
UltraScale架构设计	2天	29-30日 上海	4-5日 成都	25-26日 上海

Xilinx在线培训课程系列（WebEx）	培训课程	4月	5月	6月
以在线培训方式实施现场课堂教学和实验，学员于线上学习。面向全国或是海外华人工程师参加的中文FPGA培训课程，适合交通不便或工作繁忙不便参加现场培训的工程师。课程安排Q&A时间，老师现场解答学员在学习和实验中遇到的问题，提供最新的实验项目现场操作并进行专业辅导，直接带给学员FPGA项目设计的亲身体验。授课老师都获Xilinx认证，并具有丰富的FPGA系统项目经验。	FPGA 设计基础（免费）	1日	6日	17日
	面向性能的设计	2-3日	4-5日	1-2日
	高级 FPGA 设计	9-10日	11-12日	8-9日
	高级PlanAhead 分析与设计	13-14日	14-15日	11-12日
	利用 7系列产品进行设计	16-17日	18-19日	15-16日
	Zynq 所有可编程SoC系统架构	20-21日	21-22日	18-19日
	Vivado设计套件工具流程（免费）	15日	27日	24日
依元素科技高级FPGA培训课程系列	**培训课程**	**4月**	**5月**	**6月**
以FPGA应用方向与案例式教学为主的FPGA实战课程，使用Xilinx最新的FPGA培训课程为基础，理论部份针对工程师在设计上最常见需求和问题来安排培训内容，实践部分结合实际项目案例培养动手能力和解决实际问题的调适能力，是有一定 FPGA设计基础工程师，想提升设计能力的最好选择课程，授课老师都获Xilinx认证，并具有丰富的FPGA系统项目经验。	基于FPGA的网络设备开发实战	2-3日 北京	4-5日 北京	1-2日 北京
	基于Xilinx FPGA的DSP系统设计		7-8日 北京	
	设计高速串行传输电路	23-24日 北京	11-12日 北京	25-26日 北京
	通用FPGA接口电路设计技术		14-15日 北京	
	基于Xilinx FPGA的高速存储接口设计	27-28日 北京	18-19日 北京	4-5日 北京
	Xilinx FPGA设计高级进修班	29-30日 北京		11-12日 北京
	深入浅出FPGA在高速光网络中的开发应用		28-29日 北京	

有关报名注意事项：

请联系：北京: 电话: 010-8275-7632, 传真: 010-62166151

深圳: 电话: 0755-86186715, 传真:0086-755-86186700,

电子邮件: training@e-elements.com

地址: 北京市海淀区北三环西路32号恒润国际大厦801

网址: www.e-elements.com

遥遥领先，Xilinx发布业界首款16nm UltraScale+产品系列

发布全新 UltraScale+ FPGA、SoC 和 3D IC 系列，应用涵盖 L TE Advanced、早期 5G 无线、Tb 级有线通信、汽车高级驾驶员辅助系统（ADAS），以及工业物联网（IoT）等。

2015 年 2 月 25 日，中国北京 - All Programmable 技术和器件的全球领先企业赛灵思公司 (NASDAQ: XLNX)今日宣布，其 16nm UltraScale+™ 系列FPGA、3D IC 和 MPSoC 凭借新型存储器、3D-on-3D 和多处理SoC（MPSoC）技术，再次实现了领先一代的价值优势。为实现更高的性能和集成度，UltraScale+ 系列还采用了全新的互联优化技术——SmartConnect。这些新的器件进一步扩展了赛灵思的 UltraScale 产品系列 （现从 20nm 跨越至 16nm FPGA、SoC 和 3D IC 器件），同时利用台积公司的 16FF+ FinFET 3D 晶体管技术大幅提升了性能功耗比。通过系统级的优化，UltraScale+ 提供的价值远远超过了传统工艺节点移植所带来的价值，系统级性能功耗比相比 28nm 器件提升了 2 至 5 倍，还实现了遥遥领先的系统集成度和智能化，以及最高级别的安全性。

新扩展的赛灵思 UltraScale+ FPGA 系列包括赛灵思市场领先的 Kintex® UltraScale+ FPGA 和 Virtex® UltraScale+ FPGA 以及 3D IC 系列，而Zynq® UltraScale+ 系列则包含业界首款全可编程 MPSoC。凭借这些新的产品组合，赛灵思能够满足各种下一代应用需求，包括 LTE Advanced、早期 5G 无线、Tb 级有线通信、汽车驾驶员辅助系统，以及工业物联网(IoT)应用等。

赛灵思可编程产品部执行副总裁兼总经理 Victor Peng 表示：“面向各种下一代应用，赛灵思的 16nm FinFET FPGA 和 MPSoC 可以提供领先一代的价值优势。我们全新的 UltraScale+ 16nm 产品组合提供了高出 2 至 5 倍的系统性能功耗比，实现了系统集成和智能化的巨大飞跃，以及客户所需要的最高级别的保密性和安全性。这些功能显著地扩大了赛灵思的现有市场。”

XILINX®
欢迎各位作出反馈讯息和建议
传真：(010) 5939 3005
电邮：webmarketing_apac@xilinx.com

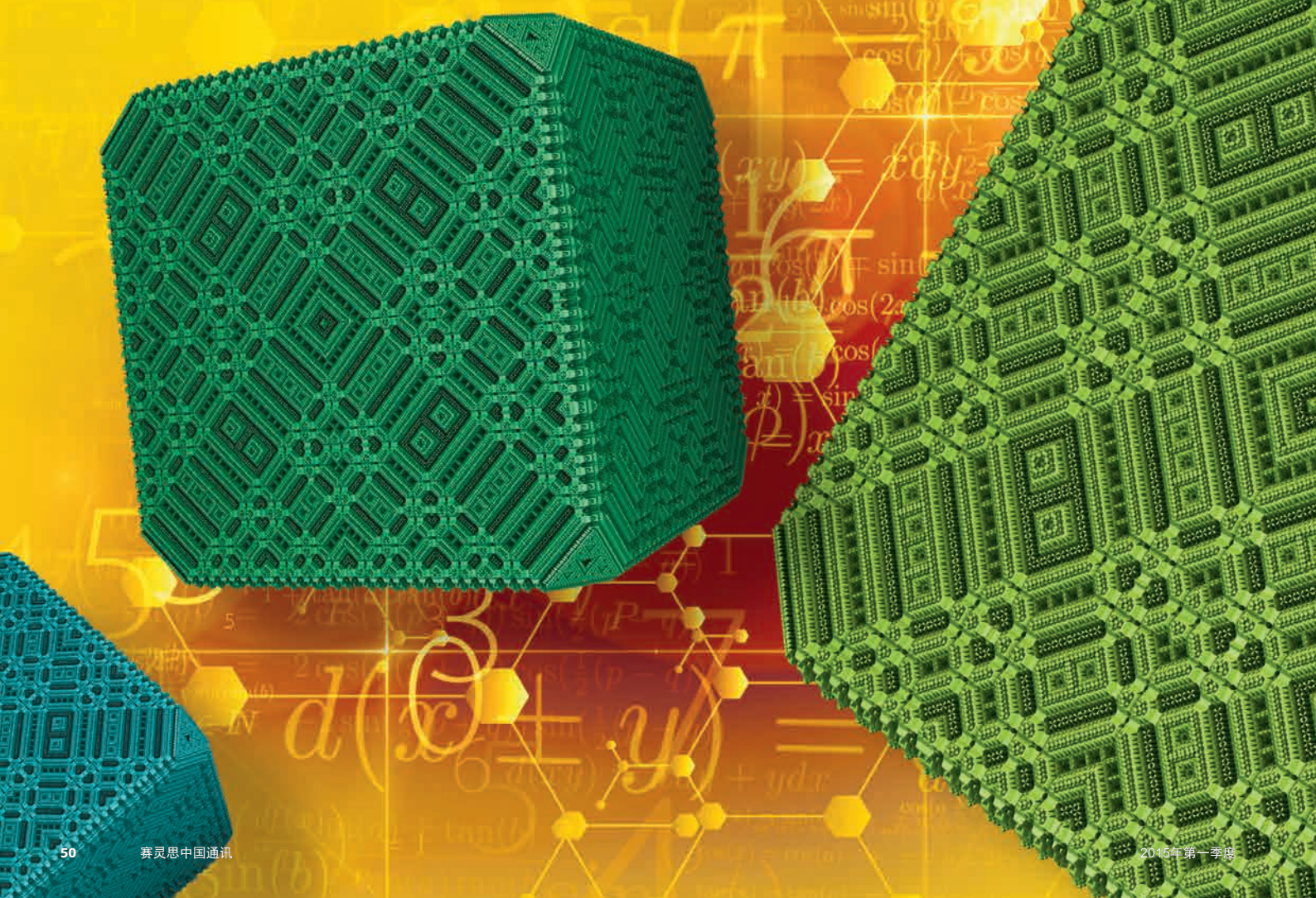
赛灵思 中国 /香港代表处

香港	电话：(852)2424 5200	传真：(852)2494 7159
上海	电话：(86)21-5131 6060	传真：(86)21-5198 1020
深圳	电话：(86)755-8660 6588	传真：(86)755-2583 0986
北京	电话：(86)010-5651 7300	传真：(86)10-5939 3005

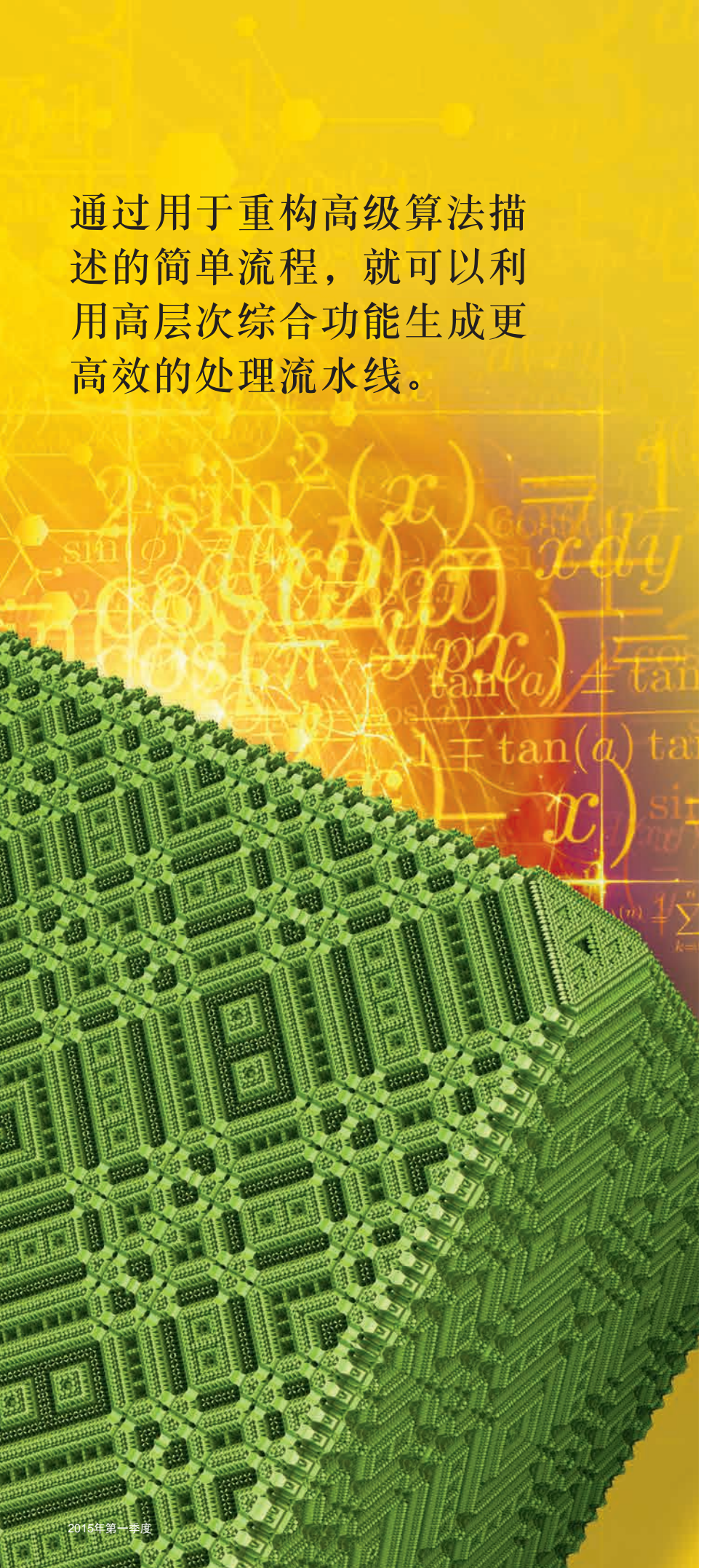
电邮：ask-china@xilinx.com

尝试通过算法重构和Vivado HLS生成高效的处理流水线

作者：Shaoyi Cheng
博士候选人
加州大学伯克利分校
sh_cheng@berkeley.edu



通过用于重构高级算法描述的简单流程，就可以利用高层次综合功能生成更高效的处理流水线。



W

如果您正在努力开发计算内核，而且采用常规内存访问模式，并且循环迭代间的并行性比较容易提取，这时，Vivado® 设计套件高层次综合 (HLS) 工具是创建高性能加速器的极好资源。通过向 C 语言高级算法描述中添加一些编译指示，就可以在赛灵思 FPGA 上快速实现高吞吐量的处理引擎。结合使用软件管理的 DMA 机制，就可以比通用处理器提速数十倍。

然而，实际应用中经常会遇到难以处理的复杂内存访问问题，尤其是当突破科学计算和信号处理算法领域时更是如此。我们设计出了一种简单方法，可供您在此类情况下生成高效的处理流水线。在详细介绍之前，我们首先了解一下 Vivado HLS 的工作原理，更重要的是了解它何时不起作用。

HLS工具如何起作用?

高层次综合功能试图获取由高级语言描述的控制数据流图 (CDFG) 中的并行性。对计算操作和内存访问进行分配和调度时，应根据它们之间的依赖约束和目标平台的资源约束来执行。电路中特定操作的激活与某个时钟周期相关，同时，沿数据路径综合的中央控制器协调整个 CDFG 的执行。

单纯在内核上应用HLS可以建立一条具有众多指令级并行性的数据路径。但是当它被激活时，就需要频繁停下来等待数据送入。

由于调度工作是在静态下完成的，因此加速器运行时间的行为相当简单。所生成电路的不同部分相互之间以相同步调运行；并不需要动态的相关性检查机制，例如高性能 CPU 上出现的那种。例如，在图 1(a) 所示的函数中，循环索引添加和 curInd 的加载可以并行处理。此外，下次迭代可以在当前迭代完成前开始。

同时，由于浮点乘法通常使用上次迭代的乘法结果，因此可以开始新迭代的最短间隔受到浮点乘法器时延的限制。该函数的执行调度如图 2(a) 所示。

该方案何时达不到理想效果？

这种方案的问题在于整个数据流图严格按调度运行。片外通信产生的拖延会传播到整个处理引擎，从而导致性能大幅下降。当内存访问模式已知，数据能在需要使用之前移动到芯片上，或者如果数据集足够小，则可完全高速缓存在 FPGA 上，这类情况下不会有问题。然而，就很多有趣的算法而言，数据访问取决于计算结果，而且内存占用决定了需要使用片外 RAM。现在，在内核上单纯应用 HLS

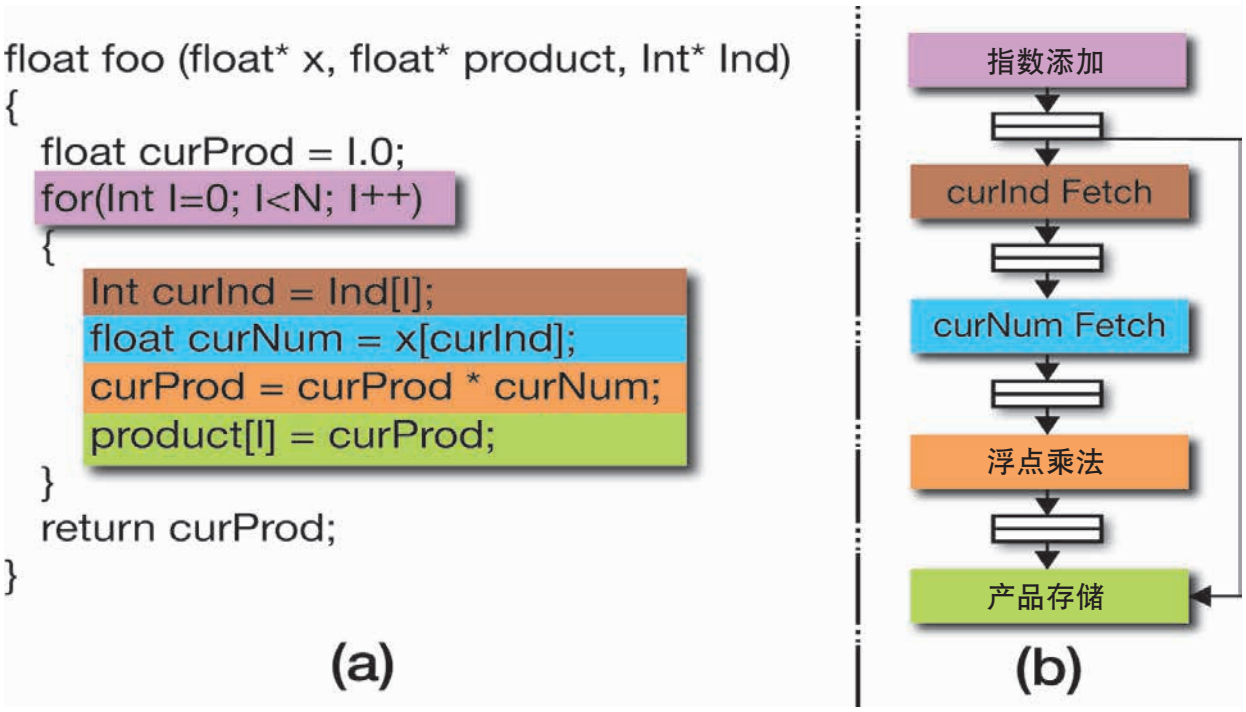


图1 – 设计实例：(a) 包含不规则内存访问模式的函数；(b) 重构得到的流水线结构

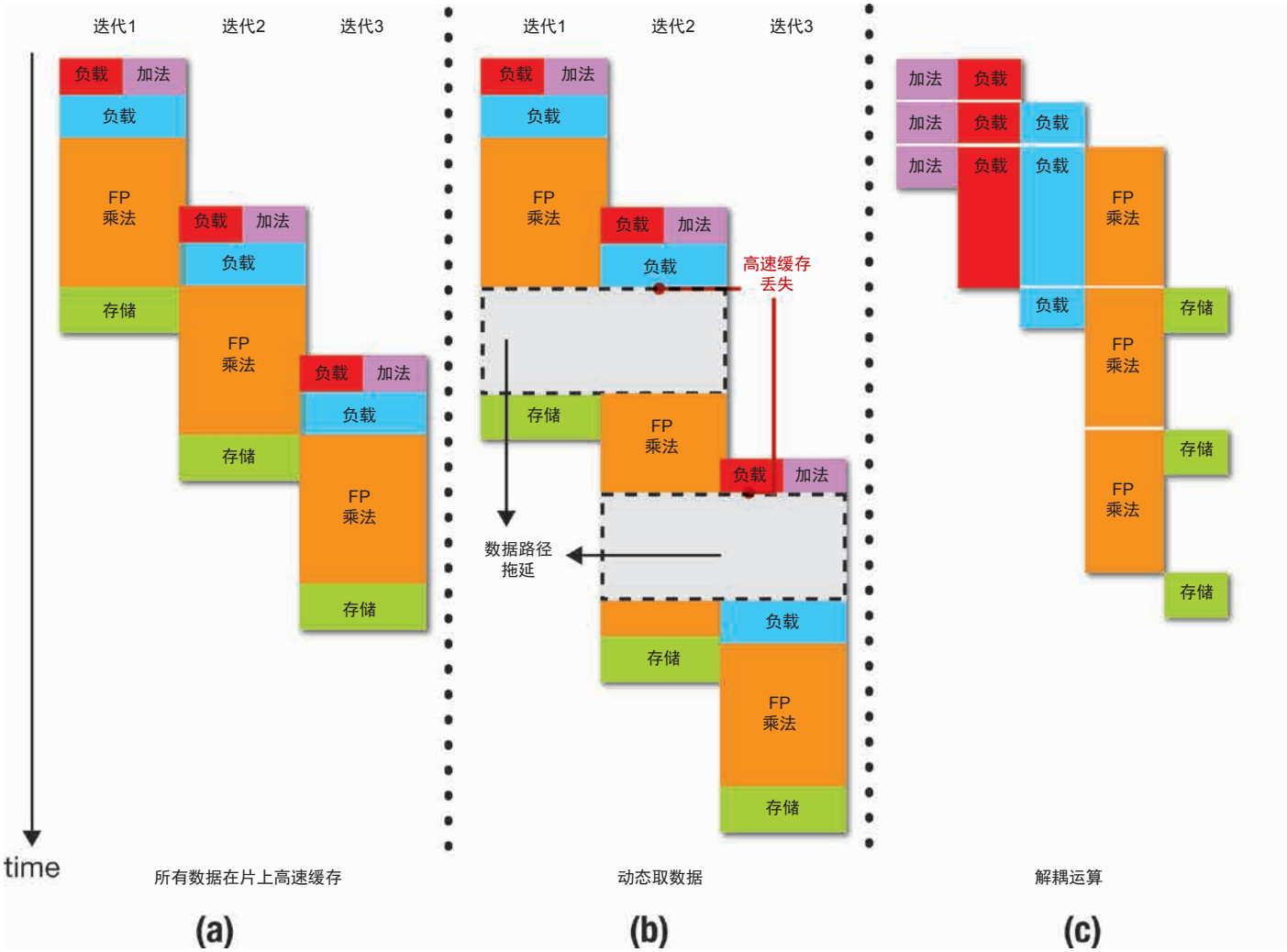


图2 – 不同情形下的执行调度：(a) 当所有数据都在片上高速缓存；(b) 动态取数据；(c) 解耦运算

可建立一条具有众多指令级并行性的数据路径。但是，当它被激活时，就需要频繁停下来等待数据送入。

图 2(b) 给出了针对实例函数生成的硬件模块的执行情况，此时数据集太大，需要动态送入片上高速缓存。注意减速程度如何反映所有高速缓存缺失时延的综合影响。不过，情况并非一定如此，因为计算图中有些部分的进展不需要立即提供内存数据。这些部分应该可以向前移动。执行调度中这点额外自由度有可能产生显著影响，就像我们看到的那样。

重构/解耦实例

我们看一下刚才的实例函数。假设浮点乘法的执行和数据访问没有全部由统一的安排联系在一起。当一个负载运算符等待数据返回时，另一个负载运算符可以开始新的内存请求，乘法器的执行也能向前移动。为达到此目的，每项内存访问都应该由一个模块来负责，并按各自的调度运行。此外，乘法器单元应该与所有内存操作异步执行。

不同模块间的数据相关性，通过硬件 FIFO 来通信。对于我们的实例而言，可能的重构形式如图 1(b) 所示。用于各阶段之间通信的硬件队列可以缓冲已经取回但尚未使用的数据。当内存访问部件因高速缓存缺失而出现拖延时，当前已产生的积压数据还可以继续供乘法器单元使用。在经历较长时间后，形成的拖延时间会被浮点乘法的长时延掩盖。

图 2(c) 给出了使用解耦处理流水

线时的执行调度。这里，通过 FIFO 的时延没有考虑在内，不过如果迭代量很大，该时延的影响会达到最小。

我们如何进行重构?

为了给解耦处理模块生成流水线，首先需要将初始 CDFG 中的指令进行组合以构成子图。为使所得的实现方案性能最大化，聚类方法必须满足几个要求。

首先，正如我们之前所见，Vivado HLS 工具在前面的迭代完成之前使用软件流水线发起新的迭代。CDFG 中最长循环依赖的时延决定可发起新迭代的最小间隔，最终会限制加速器所能实现的总吞吐量。因此，很重要的一点在于这些依赖循环不能遍历多个子图，例如用于模块间通信的 FIFO 总是会增加时延。

其次，应该将内存操作与涉及长时延计算的依赖循环分开，这样高速缓存缺失就会被慢速的数据处理所“掩盖”。在这里，“长时延”是指操作需要一个周期以上的时间才能完成；在这里，我们使用 Vivado HLS 调度来获取这一指标。例如，乘法是长时延操作，而整数加法不是。

最后，为了将高速缓存缺失引起的拖延影响限定在局部范围内，您需要将每个子图中的内存操作数量减至最少，尤其是在需要寻址存储空间中的不同部分时更是如此。

第一个要求 —— 防止依赖循环遍历多个子图 —— 很容易满足，只需要找到原始数据流图中的强连通分量 (SCC)，并在将它们分为不同集群之前将其打开变成节点。这样，我们就得到一个有向的非循环图，其中有些节点是简单指令，其它则为一组相关的操作。

要满足第二和第三个要求，即分离内存操作和局部化拖延的影响，我们可以对这些节点进行拓扑排序，然后将它们分区。最简单的分区方法是在每个内存操作或长时延 SCC 节点后画一条“边界”。图 3 展示了如何将此方案应用于我们的实例。集群与图 1 中流水线结构之间的对应关系应该做到显而易见。每个子图都是一个新的 C 函数，可独立通过 HLS 推送。这些子图在执行时相互间的步调并不一致。

我们构建了一个简单的源到源转换工具，用以执行重构。我们使用赛灵思 IP 核，支持 FIFO，以连接所生成的独立模块。当然，重构给定计算内核的方法不止一种，而且设计空间探索仍在进行中。

流水线化内存访问

有了解耦处理流水线的初步实施方案后，我们就可以对其执行几项优化，以提高其效率。正如我们所见，当使用 HLS 映射 C 函数时，内存读取出现阻塞。这个问题也出现在流水线中的个别阶段。例如，负责加载 x[curInd] 的模块在等待数据时可能会产生拖延，即使在下个 curInd 已经就绪而且 FIFO 下游有足够空间的情况下亦是如此。

为了解决这个问题，我们可以做一下转变以简化内存访问。对于某个特定阶段，我们不在 C 函数中执行简单的内存加载，而是将地址推送到新的 FIFO。然后，单独实例化一个新的硬件模块，以读取地址 FIFO 送出的地址，并将它们发送到内存子系统。返回的数据被直接推送到下游 FIFO。现在，内存访问得到了有效的流水线化。

地址的推送操作可在 Vivado HLS 中通过向 FIFO 接口的内存存储来代

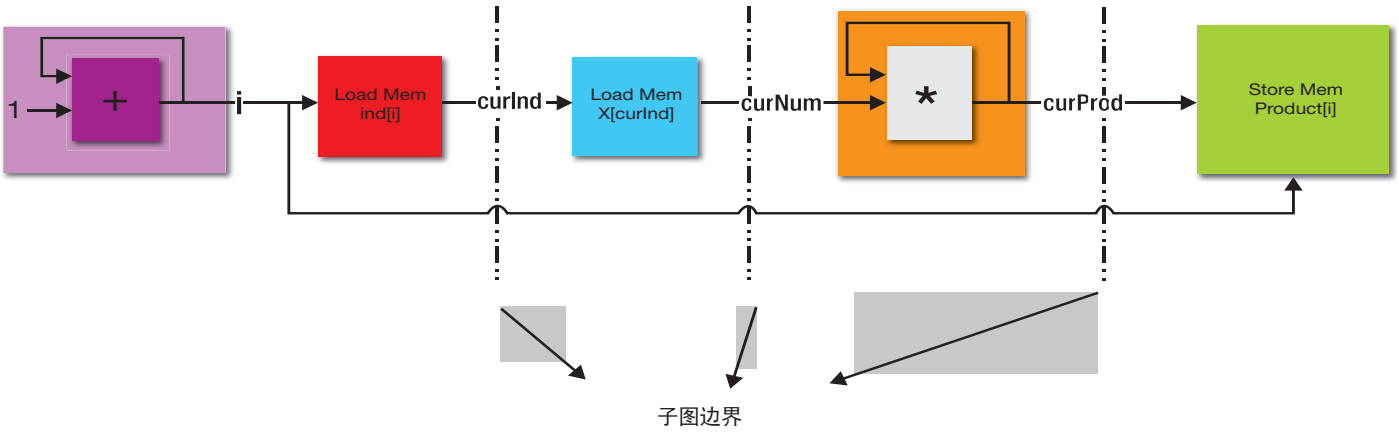


图3 – 对子图的重构

```
for (w = 1; w <= W; w++) {
  int option1 = opt[n-1][ w];
  int option2 = -999999;
  int opt_without = opt[n-1][ w-cur_weight];
  if (cur_weight <= w)
     option2 = cur_profit + opt_without;
     opt[n][w] = option1> option2? option1:option2;
     sol [n][ w] = option2> oprion1? 1:0;
}
```

图4 – 背包问题

表，不过，用以监测下游 FIFO 和发送内存请求的硬件模块则采用 Verilog 实现。这是因为在由 Vivado HLS 综合的内存接口中，外发地址和响应数据没有捆绑在一起。不过这是一个简单模块，能在不同基准测试中重用很多次，因此设计工作就被摊销了。

复制或通信?

在重构内核并生成解耦处理流水线的过程中，用来在不同阶段移动数据的 FIFO 会形成很大开销。通过复制少量计算指令可以去除一些 FIFO，这样通常很有好处，因为即使是最小深度的 FIFO 也会占用不少 FPGA 资源。

一般而言，在权衡利弊以探究最佳设计点的过程中，您可以使用成本模型和规范的优化技术。但在大多数基准测试中，仅仅为它的每个用户复制简单的循环计数器就可以节省很多面积，这也正是我们所做的。在这个引导性实例中，该优化是指复制 i 的整数加法器，因此存储结果 [i] 时不需要从其它模块获得索引。

内存的突发访问

第三项优化是内存的突发访问（burst-memory access）。为了更高效地利用内存带宽，我们希望通过一次内存事务处理携带多个数据字。AXI 总线协议允许您指定突发长度；而且，通过对解耦 C 函数进行一些小的修改，并利用流水线化的内存访问模块，我们就可利用该功能。

除了生成地址以外，解耦 C 函数中每个内存操作符还要在连续存储块被访问时计算突发长度。循环计数器的复制还有助于突发访问的生成，因为被访问的字数量可以在每个解耦函数中本地确定。

实验评估

我们应用上述方案做了几个案例研究。为评估这种方法的优势，我们将使用该方案生成的解耦处理流水线 (DPP) 与单纯使用 HLS 生成的加速器进行比较。当为单纯或 DPP 实现方案调用 Vivado HLS 时，我们将目标时钟频率设置到 150MHz，并在布局布线后使用所能达到的最高时钟速率。此外，我们针对加速器和内存子系统之间的交互尝试了不同的机制。所用的端口为 ACP 和 HP。我们为每个端口在可重配置阵列上实例化一个 64KB 高速缓存。

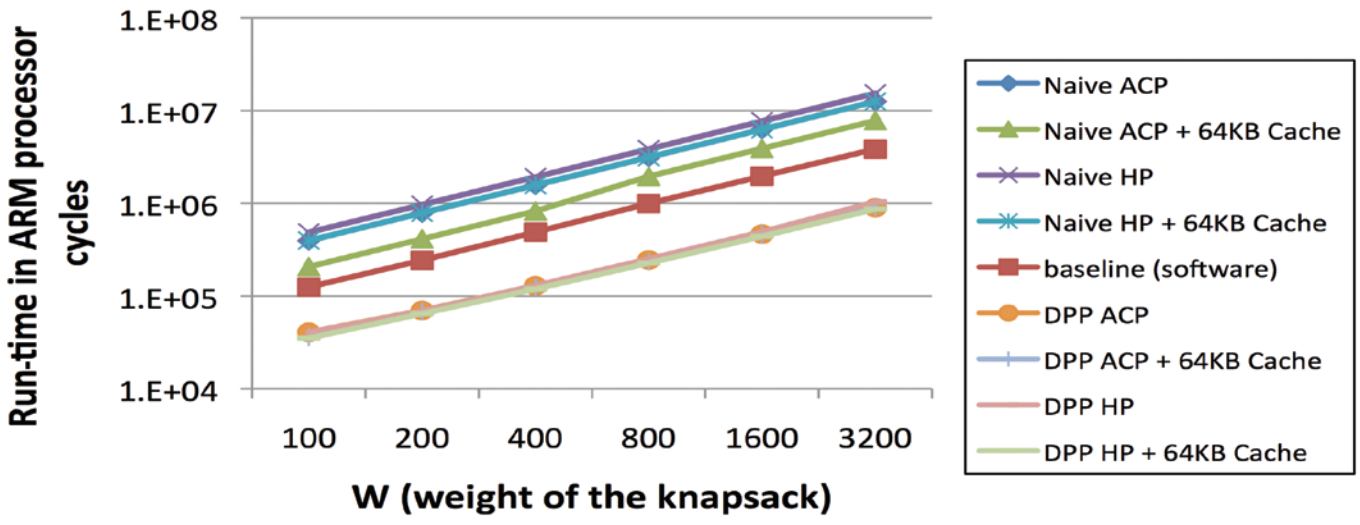


图5 – 针对背包问题的运行时间比较

本实验所用的物理器件是赛灵思的 Zynq®-7000 XC7Z020 全可编程 SoC，安装在 ZedBoard 评估平台上。我们还在 Zynq SoC 的 ARM® 处理器上运行应用的软件版本，并将其性能作为实验的基准。生成的所有加速器功能齐全，无需任何 DMA 机制将数据移入和移出可重配置架构。

```
for(s =0; s<dim; s++)
{
  int kend = ptr[s];
  int k;
  float curY = y[s];
  for(k = kbegin; k<kend; k++){
    int curInd = indArray[k];
    curY = curY +valArray[k] * xvec[curInd];
   }
   Y[s] = curY;
   kbegin = kend;
}
```

图6 – 稀疏矩阵向量乘法

案例研究1：背包问题

众所周知，背包问题是一个组合问题，可以通过动态编程来求解。内核的结构如图 4 所示。其中黑体字的变量都是在运行时间从内存读取。因此，无法确切知道从哪个位置加载的变量 opt_without。当 w 和 n 比较大时，我们无法在片上缓冲整个 opt 阵列。我们只能让计算引擎取回所需的部分。

图 5 给出了运行时间对比情况，将使用我们的方案 (DPP) 生成的加速器与单纯通过 HLS 推送函数而生成的加速器进行比较。图中还显示了在 ARM 处理器上运行函数时的性能。我们将 n（项数）固定为 40，使 w（背包的总重量）在 100 至 3,200 之间变化。

从对比中很容易看出，通过单纯使用 Vivado HLS 来映射软件内核这种方法得到的加速器性能比基准要求慢很多。Zynq SoC 上的超标量无序式 ARM 内核能很大程度开拓指令级并行性，而且具有一个高性能片上高速缓存。Vivado HLS 工具提取的附加并行性显然不足以补偿硬处理器内核对于可编程逻辑的时钟频率优势以及来自可重配置阵列的更长的数据访问时延。

不过，当内核被解耦，分成多个处理阶段时，性能就会明显比 ARM 处理器性能高出约 4.5 倍。另外，当使用 DPP 时，各种内存访问机制之间的差别相当小 —— 使用我们的方案时，受内存访问时延的影响要小很多。

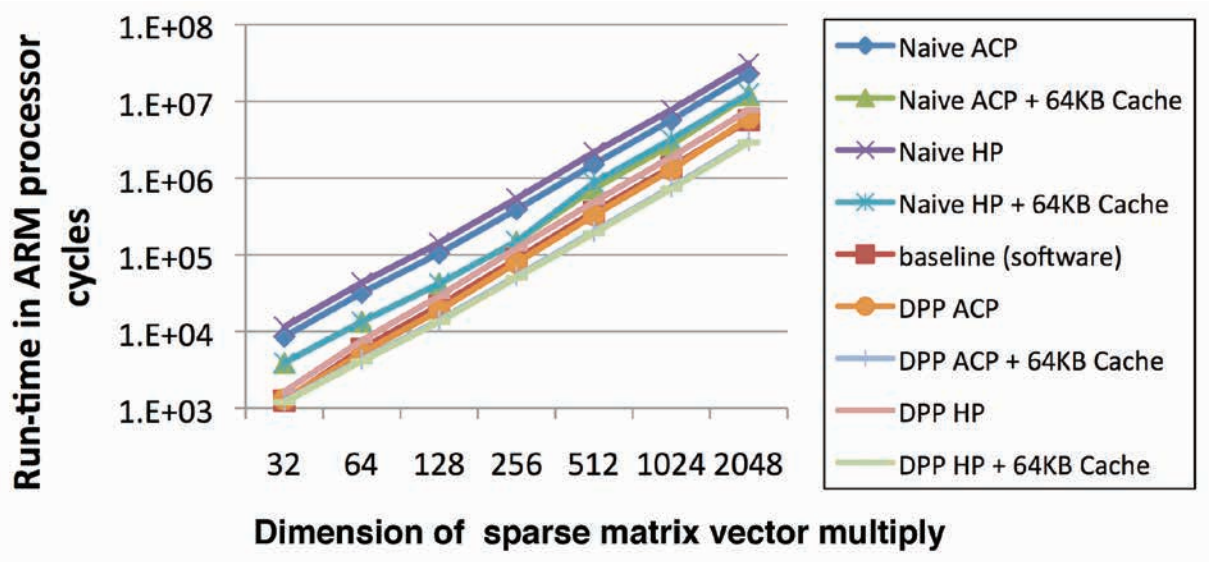


图7 – 针对稀疏矩阵向量乘法的运行时间对比

案例研究2：稀疏矩阵向量乘法

稀疏矩阵向量 (SpMV) 乘法是一个计算内核，已经在各种研究项目中以很多不同方法进行过研究、变换和基准确定。这里，我们的目的不是使用特殊数据结构和存储分配方式构建最佳性能的 SpMV 乘法，而是想根据最基本的算法描述看看在使用 Vivado HLS 时重构传递能提供多少优势。

如图 6 所示，在我们的实验中，稀疏矩阵以压缩稀疏行 (CSR) 格式存储。在取回数字以进行实际的浮点乘法之前，需要先执行来自索引数组的负载。用来决定访问哪个控制流程和内存位置的数值只有在运行时间才知道。

在图 7 所示的运行时间对比中，矩阵的平均密度为 1/16，尺寸在 32 和 2,048 之间变化。

此处，单纯的映射法在性能上再次落后于软件版。当不使用 FPGA 上的高速缓存时，用我们的方法生成的解耦处理流水线在性能上几乎与基准性能相同。

当在可重配置阵列上实例化一个 64KB 高速缓存时，DPP 的性能接近基准的两倍。与之前的基准相比，高速缓存的增加对 DPP 的性能具有更显著的影响。

```
for(k=0; k<V; k++)
   for(i=0; i<V; i++)
    if(i!=k) {
      int dik = dis[i][k];
      for(j=0; j<V; j++)
       if(j!=k) {
         int dkj =  dist[k][j];
         int dij =  dist[i][j];
         if(dik +  dkj < dij )
           dist[i][j] = dik + dkj;
       }
    }
```

图8 –Floyd-Warshall算法

案例研究3：FLOYD-WARSHALL 算法

Floyd-Warshall 是一种图形算法，用来找到任意一对顶点之间成对的最短路径。内存访问模式比之前的基准要简单。因此，有可能存在一种方法可以设计出 DMA+ 加速器结构，以获得很好的计算重叠和片外通信。我们的方案能试着自动实现这种重叠，但是我们尚未进行相关的研究，以表明绝对最佳与实际所得之间的差距。

不过，与之前的基准一样，我也进行了运行时间对比。这里，我们使图形的大小在 40 个节点至 160 个节点之间变化。每个节点平均有全部节点的 1/3 作为其邻点。

得到的结果与背包问题中的十分类似。解耦处理流水线所实现的性能约为软件基准的 3 倍，吞吐量达到任何单纯映射法的两倍多。当使用 DPP 时，对 FPGA 高速缓存的影响也很小，展示出了对于内存访问时延的容限。

我们这种简单的技术构建出的处理流水线可以更好地使用内存带宽，而且对内存时延有更好的容限，因此能够改善 Vivado HLS 的性能。所描述的方法可对控制数据流图中的内存访问和较长的依赖循环解耦，这样高速缓存缺失就不会拖延加速器的其它部分。

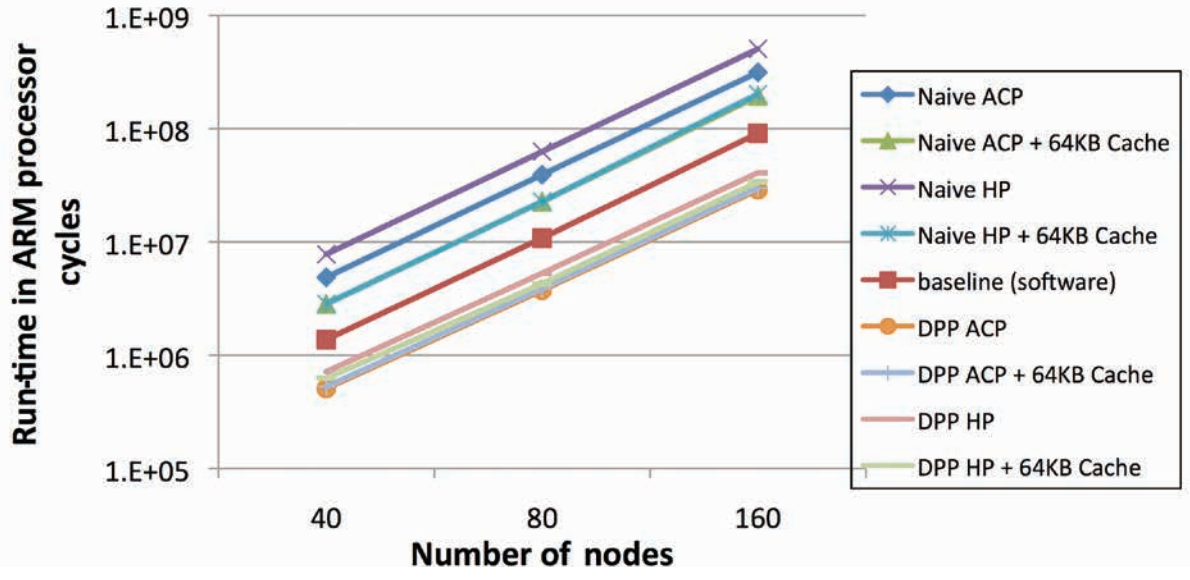


图9 – 针对Floyd-Warshall算法的运行时间比较

Xilinx联盟计划合作伙伴的最新技术

重点介绍了赛灵思联盟合作伙伴生态系统的最新技术更新

赛灵思联盟计划是指与赛灵思合作推动全可编程技术发展的认证公司组成的全球性生态系统。赛灵思创建了这个生态系统，旨在利用开放平台和标准以满足客户需求并致力于帮助它取得长期成功。包括IP提供商、EDA厂商、嵌入式软件提供商、系统集成商和硬件供应商等在内的赛灵思联盟成员助力提升您的设计生产力，同时最大限度地降低风险。下面为您分享一些精彩案例。

面向ZYNQ ULTRASCALE+ MPSOC的基于LINUX的多核框架

鉴于即将推出的赛灵思 Zynq® UltraScale+ ™ MPSoC 具备更高的容量、性能和复杂度，因此应用开发人员需要采用新的改进的软件开发范式来有效管理并充分发挥该器件提供的异构处理能力。

Mentor Graphics 的 Mentor 嵌入式多核框架作为一种支持基础结构可以管理计算资源的生命周期以及异构多处理环境中的处理器间通信。Mentor 产品组合的初始集成展示了在管理生命周期和通信的四核 ARM® Cortex ™ -A53 上运行的 SMP Linux。Nucleus RTOS 运行于采用 Mentor 嵌入式多核框架的 ARM Cortex-R5 内核上。

Mentor 的 Sourcery Codebench 工具可提供一种用来设计非对称多处理 (AMP) 系统的集成开发环境。开发人员在对异构内核上的异构软件环境进行调试和特性描述时需要面对特有的挑战。Mentor 的嵌入式开发工具为用户避免了这些复杂问题，并使用户深入了解系统运行时间。这些工具包括如下：

- 用于在 AMP 系统中进行资源分区的工具（今年晚些时候供货）
- 用于构建和封装远程固件 / 应用程序的工具
- 用于调试 AMP 系统中出现的每个软件环境的 IDE
- 能对每个 OS/ 应用环境进行特性描述并以统一的时间基准来分析数据的工具

如需了解更多信息，敬请访问网址 http://www.mentor.com/embedded-software/.

MATHWORKS扩大对 ZYNQ-7000全可编程SOC的支持

赛灵思联盟计划成员 MathWorks 在 2014b 版本中扩大了对赛灵思 Zynq-7000 全可编程 SoC 的支持。最新版 MATLAB® 和 Simulink® 使工程师和科研人员能够在统一工具环境中利用基于模型的设计更自信地加快生产进度。

MathWorks 与赛灵思一致保持密切合作，共同进行技术开发，成功向市场推出了这种创新型指导流程。该工作流程可借助自动生成的 C 语言代码和 HDL 代码，便于客户开发和仿真算法，迅速完成验证并部署到 Zynq 上。该方法充分利用 Zynq SoC 的双核 ARM Cortex-A9 处理器以及强大的可编程逻辑架构，使工程师和科研人员能够在单个芯片上设计并实现算法，从而减少最终系统的板级空间和功耗。

除了对于赛灵思 ISE® 设计套件和 Zynq 智能驱动套件的已有支持外，最新版本还能与赛灵思 Vivado® 设计套件和 Zynq SDR 开发平台进行集成。这样，工程师和科研人员就可在硬件开发平台上快速实现原型设计，然后用 Vivado 设计套件将生成的 C 和 HDL 代码整合到生产环境。

MATLAB 2014b 版本现可立即提供对赛灵思 SoC 的扩展支持。

此外，MathWorks 还提供为期两天的培训课程，以帮助工程师快速掌握和使用该技术。如需了解更多信息，敬请访问 2014b 版本亮点介绍页。

INTELLIPROP发布针对赛灵思7系列和ULTRASCALE FPGA的NVME IP核

赛灵思联盟成员 IntelliProp 与赛灵思协作推出行业标准 NVMe 主机接口和器件接口 IP 核。IntelliProp 的 NVMe Host (IPC-NV164-HI) 和 NVMe Device (IPC-NV163-DT) 内核使得与赛灵思 FPGA 上实现的 PCIe® 存储设计的通信成为可能。IntelliProp 的 NVMe IP 核完全符合非易失性存储器 Express 行业规范。它们提供具有一个处理器接口的应用层，以实现对寄存器和存储器的访问。这两款 IP 核支持到系统总线的连接，能实现无缝的 IP 访问并可方便地与任何系统设计集成。它们充分发挥赛灵思 7 系列和 UltraScale FPGA 中硬 PCIe 模块的功能，并支持 Verilog 和 VHDL 语言。

IntelliProp 的 NVMe 内核可集成到 7 系列和 UltraScale FPGA 中，以提供行业兼容型 PCIe Gen1、Gen2 或 Gen3 接口。NVMe Host IP 核通过与 NVMe 兼容型主机应用集成，实现与 NVMe 驱动器连接，进而可以向系统存储器队列发送命令并与端点设备的寄存器组进行交互。NVMe Device IP 核提供一个用以处理主机命令和执行 PCIe 数据管理任务的 NVMe 兼容型器件应用。两款 IP 核都用来实现主机系统存储器与 PCIe 连接器件间的高效数据移动。

用户应用程序可通过 Vivado 设计套件创建，并针对 IP Integrator 来封装，利用参考设计实现快速开发。IntelliProp 的 IP 核在定价上颇具竞争力，可立即订购。敬请访问 *http://www.intelliprop.com* 或 *info@intelliprop.com* 了解产品详情和订购选择。

TOPIC通过网上购物加速嵌入式开发

赛灵思高级合作伙伴 Topic Embedded Solutions 开了一个销售高质量嵌入式解决方案的网上商店。该公司提供集成解决方案的完整模块化产品组合，旨在显著缩短开发周期。

首先，基于 Zynq SoC 的 Miami SoM 是一款可直接编程和使用的工业级模块，标配有启动快速的完整主线 Linux 板支持包 (BSP)。板支持包不断更新，可在线提供，确保客户保持最新的 Linux 软件开发。

完整的 Florida 载板系列可与 Topic 的 SoM 实现全系统集成。针对医疗和常规应用的专用载板提供丰富的现成功能，可满足开发、原型设计甚至生产需求。载板的完整原理图和布局图在购买时配套提供，以确保快速和成功地实现集成。

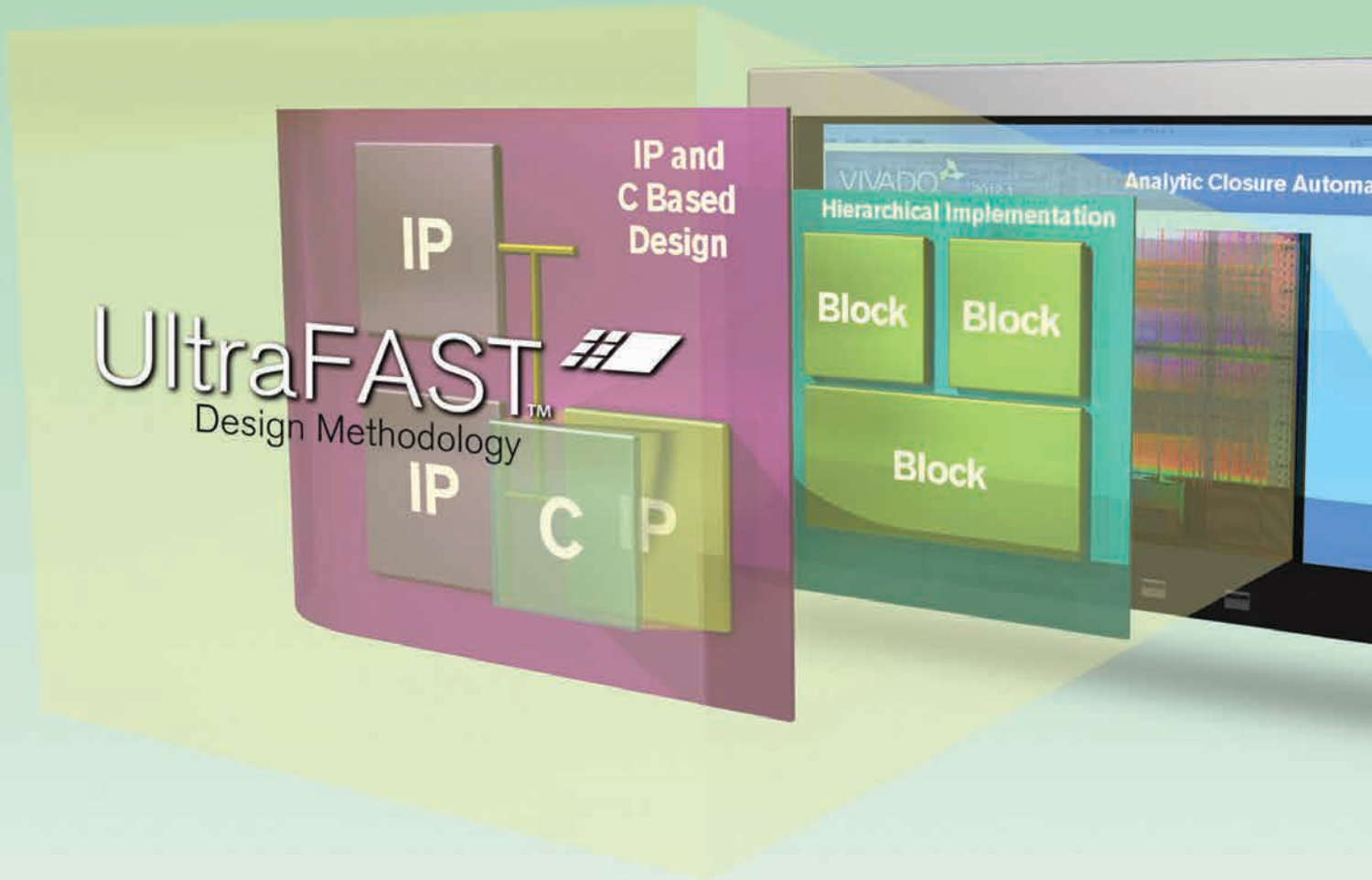
Topic 刚刚还发布了一款新的 PCIe 版本 Florida 载板，理想适用于视频、信号或高速数据处理等数据加速应用。

越来越多的全集成开发套件可简化研究、原型设计和快速工程设计。这些套件包含触摸屏、专用 I/O、所有相关线缆和参考设计。

您可通过 Topic 的全球分销合作伙伴，以及在 Topic 网上商店 *www.topicproducts.com* 上找到 Topic 嵌入式产品。

赛灵思推出

Vivado® 设计套件的 UltraFast™ 设计方法



赛灵思的 UltraFast™ 设计方法可加速设计进程并可预测设计周期。

VIVADO

XILINX