

引用格式:李岑,贺光辉. 用于实时目标检测的 FPGA 神经网络加速器设计[J]. 微电子学与计算机,2020,37(7):6-11. [LI C, HE G H. Design of FPGA-based neural network accelerator for real-time objective detection [J]. Microelectronics & Computer, 2020,37(7):6-11.]

用于实时目标检测的 FPGA 神经网络加速器设计

李岑,贺光辉

(上海交通大学 电子信息与电气工程学院,上海 200240)

摘 要: 在 FPGA 上实现 YOLO 等目标检测算法,需要从模型量化到硬件优化等多种优化方法. 为了缩短硬件延时,使用了三种技术:(1)利用层融合和位宽量化策略来降低计算复杂度;(2)利用具有 padding 跳过技术的基于列的流水线架构来减少启动时间;(3)利用设计空间探索算法来平衡流水线时间,提高 DSP 使用效率. 为了验证提出的神经网络加速器架构,在 ZC706 FPGA 上实现了具有 $1\,280\times 384$ 输入的 YOLO 网络. 与传统加速器相比,取得了 1.97 倍的延迟缩减或者 1.54 倍的 DSP 效率提升.

关键词: YOLO;FPGA 加速器;基于列的流水线架构;低延时;高 DSP 效率

中图分类号: TP183

文献标识码: A

文章编号: 1000—7180(2020)07—0006—06

Design of FPGA-based neural network accelerator for real-time objective detection

LI Cen, HE Guang-hui

(School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University,
Shanghai 200240, China)

Abstract: Implementing object detection algorithms, such as YOLO, in FPGA requires multi-level optimization, starting from model quantization to hardware optimization. To optimize hardware latency, three techniques are used: (1) bit-width quantization and layer fusion strategies are used to minimize the computation complexity, (2) a column-based pipeline architecture with padding skip technique is introduced to reduce the start-up time of pipeline and (3) a design space exploration algorithm is proposed to balance the pipeline and improve the DSP efficiency. To demonstrate the proposed neural network accelerator architecture, YOLO with $1\,280\times 384$ input is implemented on ZC706 FPGA and achieves a $1.97\times$ latency reduction or a $1.54\times$ DSP efficiency improvement over traditional accelerators.

Key words: YOLO;FPGA accelerator; column-based pipeline architecture; low latency; high DSP efficiency

1 引言

近年来,卷积神经网络(Convolutional Neural Network,CNN)领域取得了重大突破,极大地提高了基于 CNN 的目标检测算法的性能. 来自 PASCAL VOC 数据集^[1]的分类挑战结果证明,自 2007

年以来,目标检测算法的平均精度(Mean Average Precision,mAP)从 20%提高到了 85%. 目标检测算法的优异性能使它们被广泛应用于智能系统中,如机器人、自动驾驶和无人机等.

然而,目标检测算法的高准确度伴随着高计算复杂度的问题. 如果单纯使用 CPU 来计算,需要很

长时间才能完成,无法满足实时性高的应用场景.因此,需要使用FPGA等硬件进行加速计算,从而实现目标检测网络的实时推理.为了在FPGA上部署CNN,近年来已经有许多相关方法被提出,如模型量化^[2-3]和硬件优化^[4]等.同时,为了提高硬件资源的利用率,文献^[5]建立了分析模型,对硬件资源的分配进行设计空间探索,实现负载平衡.但是,其按照时分复用的方式逐个处理中间的隐藏层,使得从输入图片到最终输出的延迟增长.尽管现有技术通过降低算法的复杂度或提高硬件利用率成功地提高了吞吐量和能效,但是他们并没有太多关注整体延迟的优化.然而,对于实时性要求较高的应用,整个目标检测系统不仅需要实现高吞吐量和高能效,低延迟也是需要考虑的重要指标.

为了降低系统延时,本文利用层融合和位宽量化技术来降低计算复杂度;利用基于列的流水线架构缩短启动延时;利用设计空间探索算法提高DSP计算效率.相比传统设计,本文提出的加速器架构实现了1.97倍的延时缩减.

2 目标检测网络及优化策略

2.1 目标检测网络

与传统的图像分类网络不同,目标检测网络既可以识别图片中物体的类别,还可以推测物体的位置.因此,目标检测网络以其目标识别和定位的能力,被广泛应用于行人检测、车辆识别跟踪和智能安防监控等各个领域.

目前,常见的目标检测网络有Region with Convolutional Neural Network(R-CNN)、Faster R-CNN和YOLO等.其中,R-CNN是第一个将CNN成功用于目标检测的方法,相比传统方法,其基于区域建议的理念,取得了较高的识别精度^[6].然而,R-CNN将检测过程分为目标识别和定位两个步骤,使得算法执行速率较慢,即使是精简之后Faster R-CNN,也达不到实时检测的要求^[7].而YOLO则将目标检测问题转化为回归问题,同时进行目标识别和定位两个任务,极大提高了检测速度.然而,由于YOLO候选框尺寸的选取比较固定,其检测精度有所下降.为了解决这一问题,YOLO2被提出,可以很好地平衡目标检测精度和速度的需求.在本设计中,我们对YOLO2的精简版YOLO2-tiny进行硬件设计.相比其他YOLO版本,在保证检测精度的同时,YOLO2-tiny拥有更小的网络规模和计算复杂度.

2.2 层融合及位宽量化

本设计所选取的YOLO2-tiny网络结构如图1所示,主要包括卷积层、批归一化层(Batch Normalization, BN)和池化层.其中BN层可以解决隐藏层协变量偏移问题,使得非线性变换函数的输入值落入对输入比较敏感的区域,以避免梯度消失问题,加速训练过程.

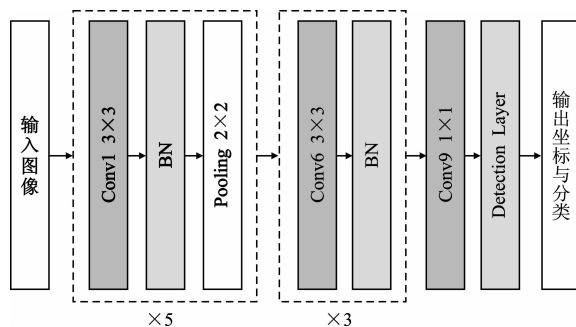


图1 YOLO2 tiny 网络结构

在推理阶段,BN层的操作如公式(1)所示:

$$y = \frac{\gamma}{\sqrt{\text{Var}[x] + \epsilon}} \cdot x + \left(\beta - \frac{\gamma \cdot E(x)}{\sqrt{\text{Var}[x] + \epsilon}} \right) \quad (1)$$

式中, x 是卷积计算结果, $\text{Var}[x]$ 是 x 方差的数学期望, $E(x)$ 是 x 均值的数学期望, γ 为缩放因子, ϵ 为较小常数防止分母为0, β 为偏置.由于 $\text{Var}[x]$ 、 $E(x)$ 、 γ 和 β 等参数可以在训练过程中得到,在推理过程中它们可以被认为是常数.因此,系数 $\frac{\gamma}{\sqrt{\text{Var}[x] + \epsilon}}$

和 $\frac{\gamma \cdot E(x)}{\sqrt{\text{Var}[x] + \epsilon}}$ 可以分别被融合到权重和偏置中.

通过这种方式,BN层在推理阶段可以完全被融合到卷积层中,而没有任何精度损失.融合之后可以减少硬件计算量,加快推理过程.

在BN层融合之后,由于CNN的冗余性和鲁棒性,可以对网络的激活值和权重进行量化.本文将YOLO2-tiny的激活值和权重在6~32比特之间进行量化,得到精度和量化位宽的对应关系如图2所示.从图中可以看出,在位宽不低于8比特时,精度损失不超过1%.因此,为了在保证检测精度的同时,最大限度地减轻硬件计算压力,在本设计中,网络激活值和权重位宽被定为8比特.

2.3 基于列的流水线设计

如图3(a)所示,在传统流水线中,只有等到前一层计算完全完成之后,才能开始计算下一层.这种流水线设计浪费大量的执行时间.此外,由于片上存储器的限制,计算的中间结果必须存储在片外

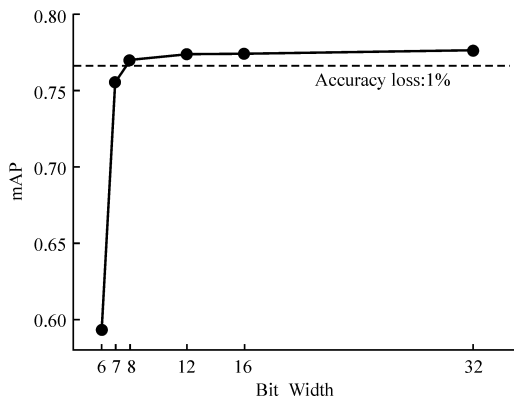


图2 YOLO2-tiny 检测精度与量化位宽的关系

DRAM 中,而且必须在下一层计算之前再次将它们取到片上,这会消耗许多额外的功耗.实际上,两个相邻层之间的数据并不是紧密耦合的.如图 3(b)所示,在一个 3×3 的卷积中,要输出下一层一列的结果,只需要前一层前 3 列的数据而不是整张特征图.因此,如图 3(c)所示,只要上一层准备好所需数据后,下一层就可以开始计算,而不需要等待整张图计算完成.这种基于列的流水线架构,大大缩短了流水线的启动延时,提高了系统的实时性.

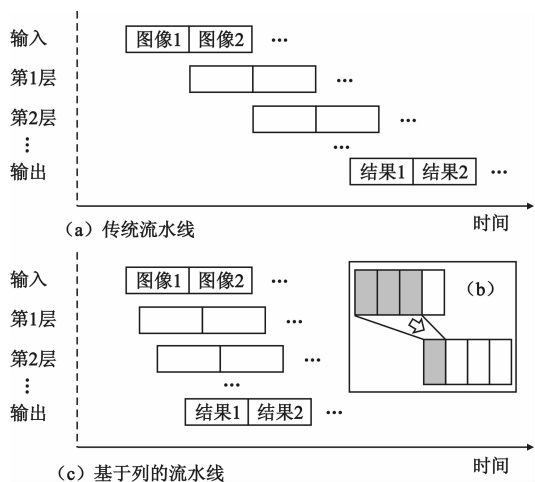


图3 基于列的流水线与传统流水线的区别

3 基于 FPGA 的硬件设计

3.1 系统架构设计

本设计的系统架构如图 4 所示.在初始化阶段,经过训练的权重通过 PCIe 总线,从主机端预载到 DRAM 上.在推断阶段,输入图像首先通过 PCIe 逐列送入输入 FIFO.然后,经由流水线中的卷积处理单元(convolutionprocess unit, CONV PE)逐级计算.最后,最终的计算结果通过 PCIe 被送到主机端进行后处理.各层 CONV PE 被组织成如 2.3 节所

述的基于列的流水线.特别地,为了进一步缩短计算时间,可以跳过第一列和最后一列计算中的 padding.

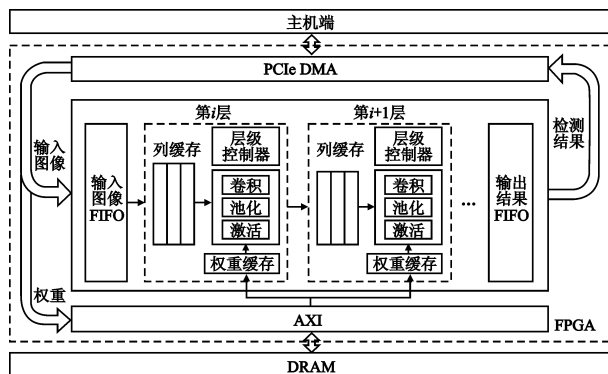


图4 系统架构图

与文献[8]相比,本设计的输入图像和输出结果直接在 FPGA 和 PC 之间传输,而无需 DRAM 的参与,可以缩短总体延迟.同时,基于列的流水线架构也可以大大减少延迟.此外,输入图像和中间结果不需要访问 DRAM,还可以提高系统能效.

3.2 卷积计算单元设计

图 5 展示了本设计的 CONV PE 的工作流程.由于片上存储资源受限,对于每一层,只能在片上存储 K_i 卷积核(对于 Ping-PongBuffer,实际存储器大小将设置为 $2K_i$).在每一个周期中,卷积处理单元从列缓存中取出 C_i 个输入,从权重缓存中取出 $K_i \times C_i$ 个权重,计算得到 K_i 个中间结果.

为了直观地阐述卷积计算的顺序,图 5 给出了一个简单的示例.在此示例中,4 个 $2 \times 2 \times 4$ 的卷积核与一个 $3 \times 3 \times 4$ 的输入特征图作卷积,生成 $2 \times 2 \times 4$ 的输出.其中 $C_i = 2, K_i = 2$ (参与计算的 DSP 数量为 $C_i \times K_i \div 2 = 2$).具体计算顺序如下所述:

步骤 1:每个周期,从输入缓存中取出 2 个输入,从权重缓存中取出 2×2 个权重.然后,将来自不同卷积核中相同位置的 2 个权重拼接在一起作为一个输入传送给封装好的乘法器,同时与 1 个输入相乘,得到 2 个中间结果.最后,来自不同 DSP 的相同位置的结果将被分别累加,生成 2 个累加的中间结果,CONV PE 的整个执行周期结束.

步骤 2:将步骤 1 循环八次,得到输出特征图上的 2 个最终结果.循环执行的顺序,先是通道方向,再是高度方向,最后是宽度方向.

步骤 3:卷积窗口向下滑动,重复步骤 2 的过程.8 个周期之后,得到输出特征图中的另外 2 个输出.此时,卷积窗口已经滑到第一列的最底端,表示

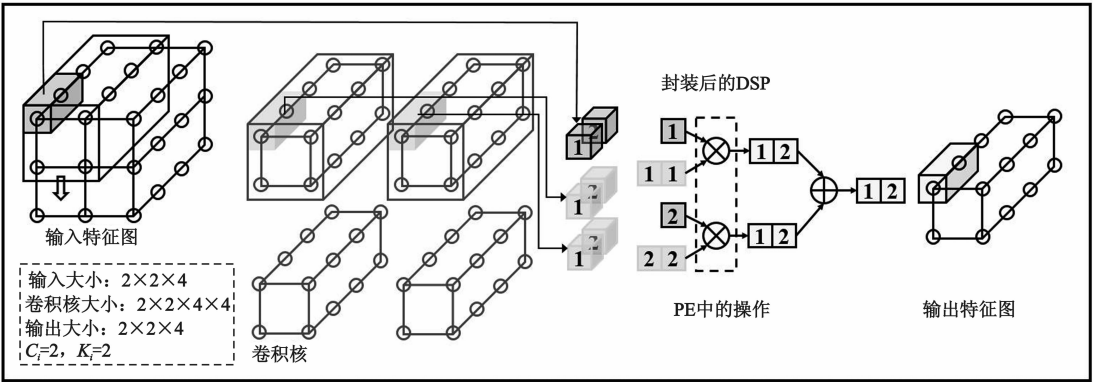


图 5 卷积计算顺序图

这两组卷积核对于第一列输出的计算已经完成,需要更新另外两组卷积核到权重缓存中,重复上述过程。

步骤 4:当所有的卷积核都计算完一次后,第一列输出计算完成,在输入特征图上沿宽度方向向右滑动卷积窗口,重复步骤 1~3,进行下一列输出的计算.总共需要 $8 \times 8 = 64$ 个周期,可完成整张输出特征图的计算。

不管输入特征图和内核的大小如何,都可以使用上述方法计算卷积.因此,可以将 CONV PE 封装成 IP,支持各种尺寸的卷积计算.此外,通过参数 C_i 和 K_i 的设置,CONV PE 可以被配置成不同的并行度(DSP 数量),以平衡每一层的计算时间。

3.3 设计空间探索算法设计

由于每层网络的参数(如输入特征图和卷积的尺寸)各不相同,输出每层卷积所需的计算量是不同的.因此,本文提出一种设计空间探索的算法,根据每层卷积的计算量来分配硬件资源,平衡每一层的流水线时间,提高硬件资源利用效率。

如图 6 所示,算法的输入是网络的参数和硬件资源约束,输出是硬件资源分配的策略.首先,根据网络各层的计算量来等比例分配 DSP 数量,并根据 DSP 数量来确定并行度参数 C_i 和 K_i .然后,根据并行度参数确定 BRAM 和带宽分配.如果 BRAM 和带宽满足硬件资源的约束,则直接输出资源配置结果.否则,按一定比例减少整体 DSP 数量,重复上述过程,直到满足所有硬件资源约束。

4 实验结果与分析

4.1 实验环境

为了测试所提出的低延迟硬件架构的性能,本文在 Xilinx ZC706 FPGA 上实现了 YOLO2-tiny 网

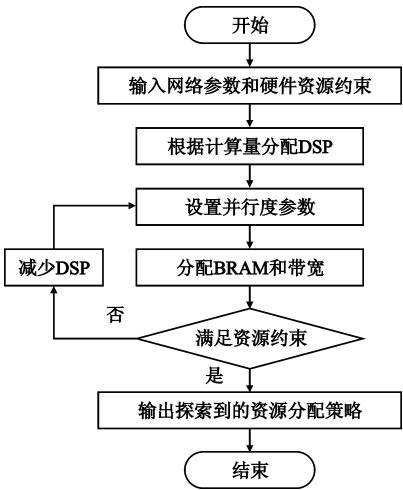


图 6 设计空间探索算法流程图

络,并在 KITTI 数据集上进行了测试.为了公平地与文献[9]中的参照标准进行比较,将 CONV7 和 CONV8 的卷积核数量更改为 512。

4.2 设计空间探索结果

3.3 节中给出的设计空间探索算法给出了分配硬件资源和平衡流水线的方法.根据算法,确定了 YOLO2-tiny 各层的并行度如表 1 所示。

表 1 各层硬件配置参数

名称	输入	卷积核	DSP	$C_i \times K_i$	周期数
Conv1	1 280×384×3	3×3×16	32	4×16	3 317 760
Conv2	640×192×16	3×3×32	64	4×32	4 423 680
Conv3	320×96×32	3×3×64	64	2×64	4 423 680
Conv4	160×48×64	3×3×128	64	4×32	4 423 680
Conv5	80×24×128	3×3×256	64	8×16	4 423 680
Conv6	40×12×256	3×3×512	64	16×8	4 423 680
Conv7	40×12×512	3×3×512	128	32×8	4 423 680
Conv8	40×12×512	3×3×512	128	32×8	4 423 680
Conv9	40×12×512	1×1×40	2	2×2	3 010 560
总计	—	—	610	—	—

从表中可以看出,除了第 1 层和第 9 层以外,其他卷积层的计算时间都完美地平衡了,这意味着设计空间探索算法取得了很好的效果. 第 1 层和第 9 层平衡不了的原因是:输入通道和输出通道的数量不是 2 的幂次,不能被并行度因子整除. 但是,第 1 层和第 9 层仅占所消耗 DSP 的 $34/610 = 5.6\%$. 因此,由于流水线时间不平衡而造成的 DSP 效率损失,可以忽略不计.

4.3 硬件实现结果

根据表 1 给出的硬件配置参数,将 YOLO2-tiny 映射到 Xilinx ZC706 FPGA 上. 如图 7 所示,输入图像通过 PCIe 从主机端发送到 FPGA. 在 FPGA 上进行推断之后,将检测结果发送回主机端进行后处理和显示.



图 7 基于 ZC-706 的原型验证系统

表 2 将本设计的性能指标与当前有关 YOLO 的硬件实现进行了比较. 基于文献[9],引入 $DSP_{efficiency}$ 来评估 DSP 的利用率,其定义为:

$$DSP_{efficiency} = \frac{Throughput}{\beta \times DSP_{used} \times f} \tag{2}$$

式中,分子表示实际达到的算力(GOPS),分母是理论上 DSP 的算力. β 表示一个 DSP 可以在一个周期内处理的操作(对于 Fix16 $\beta=2$,对于 Fix8 $\beta=4$).

表 2 性能比较

	文献[10]	文献[11]	文献[9]	本文
硬件平台	Cyclone V PCIe	VC-707	ZC-706	ZC-706
频率/Hz	117M	200M	200M	200M
网络类型	YOLO2-tiny	YOLO2-tiny	YOLO2-tiny	YOLO2-tiny
位宽	16 bits	8 bits	16 bits	8 bits
准确率	—	62%	75.7%	77%
DSP	122/300	1 024/3 600	680/900	610/900
BRAM	—	—	333/545	245/545
延时/ms	—	65	59	30
算力 GOPs	21.6	461.5	234	424
能效/(GOPs/W)	—	35.5	—	47
DSP效率	75.7%	56.3%	86%	86.9%

如表 2 所示,在网络模型和硬件平台相同的情况下,本设计的性能在系统延迟($1.97\times$)和 DSP 效率($1.01\times$)方面都优于文献[9]中的 DNN-Builder. 这意味着本设计中所采用的层融合和位宽量化策略以及具有 padding 跳过技术的基于列的流水线架构极大地改善了硬件的延迟. 此外,与文献[10]和文献[11]相比,本设计的 DSP 效率分别提高了 1.15 倍和 1.54 倍,这意味着本设计所提出的设计空间探索方法可以更好地平衡流水线性能,降低 DSP 空闲时间.

5 结束语

本文提出了一种低延迟硬件架构来优化实时目标检测系统的延迟. 通过层融合和位宽量化策略以及具有 padding 跳过技术的基于列的流水线技术,极大降低系统延迟. 同时,本设计中提出的设计空间探索算法可以有效平衡流水线并提高 DSP 效率. 硬件实现的结果显示,与目前常见的硬件设计相比,本设计的性能取得了 1.97 倍的延迟优化或者 1.54 倍的 DSP 效率提升.

参考文献:

[1] EVERINGHAM M, ESLAMI S M A, VAN GOOL L, et al. The Pascal Visual Object Classes Challenge: A Retrospective[J]. International Journal of Computer Vision, 2015, 111(1):98-136.

[2] NGUYEN D T, KIM H, LEE H, et al. An Approximate Memory Architecture for a Reduction of Refresh Power Consumption in Deep Learning Applications [C]// 2018 IEEE International Symposium on Circuits and Systems (ISCAS). Florence, Italy, 2018.

[3] NGUYEN D T, NGUYEN T N, KIM H, et al. A High-Throughput and Power-Efficient FPGA Implementation of YOLO CNN for Object Detection[C]// IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019. Sydney, Australia, 2019: 1-13.

[4] PREUßER T B, GAMBARDELLA G, FRASER N, et al. Inference of quantized neural networks on heterogeneous all-programmable devices[C]// 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). Dresden, Germany, 2018: 833-838.

[5] MOTAMEDI M, GYSEL P, AKELLA V, et al. Design space exploration of FPGA-based Deep Convolutional Neural Networks [C]// 2016 21st Asia and South Pacific Design Automation Conference (ASP-

- DAC). Macao, China, IEEE, 2016.
- [6] 吴佳祥, 刘辉, 贺光辉. 一种应用于高分辨率遥感图像目标检测的尺度自适应卷积神经网络[J]. 微电子学与计算机, 2018, 35 (8): 84-87.
- [7] 林健巍. YOLO 图像检测技术综述[J]. 福建电脑, 2019, 35(9): 80-83.
- [8] NGUYEN D T, NGUYEN T N, KIM H, et al. A High-Throughput and Power-Efficient FPGA Implementation of YOLO CNN for Object Detection[C]// IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019. Sydney, Australia, 2019: 1-13.
- [9] ZHANG X, WANG J, ZHU C, et al. DNNBuilder: an Automated Tool for Building High-Performance DNN Hardware Accelerators for FPGAs[C]// 2018 IEEE/ACM International Conference on Computer-Aided Design (ICCAD), 2018. Marrakech, Morocco, 2018.
- [10] WAI Y J, YUSSOF Z B M, SALIM S I B, et al. Fixed Point Implementation of Tiny-Yolo-v2 using OpenCL on FPGA[J]. International Journal of Advanced Computer Science and Applications, 2018, 9 (10): DOI: 10.14569/IJACSA. 2018. 091062.
- [11] YU J, GUO K, HU Y, et al. Real-time object detection towards high power efficiency[C]// 2018 Design, Automation & Test in Europe Conference & Exhibition (DATE). Dresden, Germany, 2018.

作者简介:

李 岑 男, (1995-), 硕士研究生. 研究方向为神经网络加速器设计. E-mail: lgc2013@sjtu. edu. cn.

贺光辉 男, (1980-), 博士, 副研究员. 研究方向为通信系统 VLSI 电路设计和随机计算设计等.

(上接第 5 页)

- [4] HUANG X, LIANG C, MA J. System architecture and implementation of MIMO sphere decoders on FPGA[J]. IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2008, 16(2): 188-197.
- [5] CHIU P L, HUANG L Z, CHAI L W, et al. A 684Mbps 57mW joint QR decomposition and MIMO processor for 4×4 MIMO-OFDM systems[C]// IEEE Asian Solid-State Circuits Conference 2011. Jeju, Korea, 2011: 309-312.
- [6] COLIERI S, ERGEN M, PURI A, et al. A study of channel estimation in OFDM systems[C]// Proceedings IEEE 56th vehicular technology conference. Vancouver, BC, Canada, 2002(2): 894-898.
- [7] EDFORS O, SANDELL M, VAN DE BEEK J J, et al. OFDM channel estimation by singular value decomposition[J]. IEEE Transactions on communications, 1998, 46(7): 931-939.

作者简介:

赵灿坤 男, (1997-), 硕士研究生. 研究方向为数字集成电路设计.

王自强 男, (1975-), 博士, 副研究员. 研究方向为数模混合集成电路设计. E-mail: wangzqi@tsinghua. edu. cn.