

基于异构平台的三角矩阵回代加速求解研究

时睿,左芸帆,闫浩*

(东南大学 集成电路学院,南京 210096)

摘要: 瞬态电路仿真中常建立线性系统模型,而顺序求解多右端项的三角矩阵十分耗时。为了提高瞬态电路仿真中耗时的三角矩阵回代速度,提出了一种基于异构平台的并行计算方法快速求解三角矩阵。通过优先计算与解向量相关的乘法,挖掘了回代计算的并行性。设计了核心是多个浮点计算功能的运算阵列以及主从两层状态机的控制模块。相比于使用 MKL 求解库的 Intel 24 核 CPU 平台,本架构基于 XCZU15EG 的 Zynq UltraScale 系列 FPGA 进行了线性矩阵求解实验,实验所用矩阵均为对称正定、对角占优且稠密度均大于 50%。提出的加速架构求解的平均加速比达到 22 倍,求解误差在 $10^{-17} \sim 10^{-14}$ 内。实验结果表明,该架构一定程度上提高了矩阵求解速度,适合于较高维度线性矩阵的前后向回代求解。

关键词: 三角矩阵求解;硬件加速;现场可编程门阵列;瞬态仿真

中图分类号: TP332.1

文献标识码: A

Research on accelerating triangular matrix backpropagation based on heterogeneous platforms

SHI Rui, ZUO Yunfan, YAN Hao

(School of Integrated Circuits, Southeast University, Nanjing 210096, China)

Abstract: Transient circuit simulations often necessitate the construction of linear system models, where the sequential solution of triangular matrices with multiple right-hand terms becomes a time-intensive process. To expedite the computational efficiency of back-substitution for these matrices in transient circuit simulations, this paper proposes a parallel computing method based on a heterogeneous platform. The method prioritizes the computation of multiplications relevant to the solution vector, exploiting the inherent parallelism of back-substitution calculations. The architecture features a core operation array with multiple floating-point calculation units and a control module employing a two-tiered master-slave state machine. Using the Zynq UltraScale series FPGA, specifically the XCZU15EG model, our architecture is compared to an Intel 24-core CPU platform utilizing the MKL solving library in linear matrix resolution experiments. The matrices used exhibit characteristics of being symmetric positive definite, diagonally dominant, and dense with a sparsity exceeding 50%. The proposed acceleration architecture achieves an average speedup factor of 22, with solution errors falling within the range of 10^{-17} to 10^{-14} . The experiment results demonstrate the architecture's significant enhancement of matrix solution speed, especially suitable for forward and backward substitution resolution of high-dimensional linear matrices in transient circuit simulations.

Keywords: triangular matrix solution; hardware acceleration; FPGA; transient simulation

0 引言

在瞬态电路仿真中,常将电路问题等效成求解线性系统。在仿真过程中,顺序地求解三角矩阵十分耗时^[1],并且瞬态仿真通常要进行上千次的前后向回代计算,因此求解三角矩阵是优化瞬态电路仿真的重点。为了加速求解,

研究人员提出了各自的方法。

2012 年, Xiong 等人^[2]使用重排序来重构数据依赖,基于有向无环图预先得知数据依赖关系,其在 CPU 上实现了 LU 分解后的前后向回代的并行处理,在 NICS LU 的基础上,前后向回代过程实现的加速比超过两倍。2018 年, Wang X 等人^[3]提出了一种新的稀疏层次图块的数据布

基金项目:江苏省研究生实践创新计划“面向 CCS 时序模型的矩阵回代 FPGA 加速”(SJCX220052)。

时睿(硕士研究生),主要研究方向为硬件加速设计;左芸帆(博士研究生),主要研究方向为大规模互连线可靠性仿真及优化。

* 通信作者, E-mail: yanhao@seu.edu.cn。

局方式,在 GPU 上并行解稀疏三角矩阵,与在 CPU 上执行串行算法相比,速度平均提升了 6.9 倍。2019 年,参考文献[4]提出基于众核申威架构的方法,它的层间同步只发生在能够快速通信的寄存器中,减少了同步和数据重用的代价。2020 年,Rio 等人^[5]在 FPGA 上实现了 LDLT 回代求解矩阵,与 ARM-A9 CPU 的实现相比,求解效率提升近两倍。2021 年,参考文献[6]建立关于数据依赖的有向无环图,并通过静态调度 FPGA 实现稀疏三角矩阵的求解,其基于 FPGA 的硬件加速算法实测性能是传统 CPU/GPU 求解算法的 5.9 倍。

上述的 CPU、GPU、众核 GPU 等高性能处理器平台中,各类算法都在一定程度上加速求解线性矩阵。不过由于 FPGA 低功耗、高灵活性的特点,基于 FPGA 硬件加速的方法越来越多。本文设计了三角矩阵加速架构,主要贡献如下:

- ① 并行算法:采用了按列并行计算的方法,避免了顺序计算的数据依赖性。
- ② 带有专用存储的运算单元:为适应不同维度的矩阵,每个运算单元中增加了可以暂存中间数据的 FIFO,增加了运算单元的灵活性。
- ③ 显著的加速性能:与 24 核 CPU 相比,在 FPGA 上运算阵列在求解矩阵时实现平均 22 倍的加速比。

1 矩阵求解算法研究

1.1 矩阵求解基本流程

线性方程可以写为如下形式:

$$Ax = b \quad (1)$$

当使用直接方法求解线性系统时,一般先对矩阵 LDLT 分解,再通过顺序的前后向回代来有效地求解关联的三角矩阵。具体来说,前后向回代涉及 3 个关键步骤: $Ly=b$ 、 $Dr=y$ 和 $L^T x=r$ 。其中,D 为对角矩阵。前向回代过程计算解向量:

$$\begin{cases} y_1 = b_1 \\ y_2 = b_2 - l_{21}y_1 \\ y_3 = b_3 - l_{31}y_1 - l_{32}y_2 \\ \vdots \\ y_n = b_n - \sum_{i=1}^{n-1} l_{in}y_i \end{cases} \quad (2)$$

同理,解对角矩阵与后向回代过程使用如下公式:

$$Dr = y \quad r_k = y_k / d_{kk} \quad 1 \leq k \leq n \quad (3)$$

$$L^T x = r \quad x_k = r_k - \sum_{j=k+1}^n u_{kj}^T x_j \quad 1 \leq k \leq n \quad (4)$$

一般地,求解三角矩阵的前后向回代时间较长,其中耗时的原因有 2 个:一方面在顺序求解三角矩阵的过程

中,存在数据依赖性的求解阻碍了并行计算的可能;另一方面,LU 分解后的矩阵存在对角线元素,计算解向量需耗时较长的除法运算。为避免回代中的除法,本文采用 LDLT 分解矩阵。

1.2 求解三角矩阵的并行算法

为解决上述计算解向量时的数据依赖问题,本文采用了一种流水线并行计算方法。首先计算与已知解向量相关的所有乘法,即可在每个步骤中实现多个运算单元的并行计算,降低了数据依赖性。

如下伪代码描述了如何避免数据依赖从而并行多个计算单元进行求解:

```

1: j = 2; //当前矩阵更新的解向量元素批次
2: i = 1; //当前列内更新的元素行批次
3: y1 = b1; //y1 无需计算直接赋值
4: while (j ≤ n)
5:   do in parallel (i = j : p : n) //p 为运算单元的数量
6:     while (i ≤ n) //当前列内更新计算是否结束
7:       Load&Compute(yj-1 × li,j-1 + yi*);
8:       //运算单元阵列并行计算更新
9:     end while
10:    Load(bj)&Compute(yj);
11:    //读取向量 b 元素并计算解向量元素 y
12:  end while
13:  j = j + 1;
14: end while

```

其中,p 为运算单元的数量,n 为矩阵维度;外部循环遍历 j,内部循环遍历 i;y₁ 不需要计算。内循环允许并行检索元素 l_{i,j-1},并执行乘法累加运算。更新的计算结果 y_i^{*} 将被临时存储在运算单元的 FIFO 内,以便在后续的外循环中逐列更新中间结果。

如图 1 所示,在得到 y₁ 的结果后,可以同时求解 l₂₁ * y₁, l₃₁ * y₁, ..., l_{n1} * y₁,这样即可通过减法计算出 y₂。在下一个步骤将并行求解 l₃₂ * y₂, l₄₂ * y₂, ..., l_{n2} * y₂。后续的计算步骤以此类推。

y_j 的更新依赖于前面的解决方案元素 y₁ 到 y_{j-1}。根据列更新,将三角矩阵的计算分为 n-1 个步骤。在每个步骤中,计算任务被进一步根据运算单元的数量划分。

2 架构与实现

2.1 矩阵求解的数据流

本文提出的矩阵回代求解任务框架如图 2 所示。其中,核心部分是加速器中的运算单元阵列,重点关注加速器计算性能,验证过程在 FPGA 的 PL 端进行。

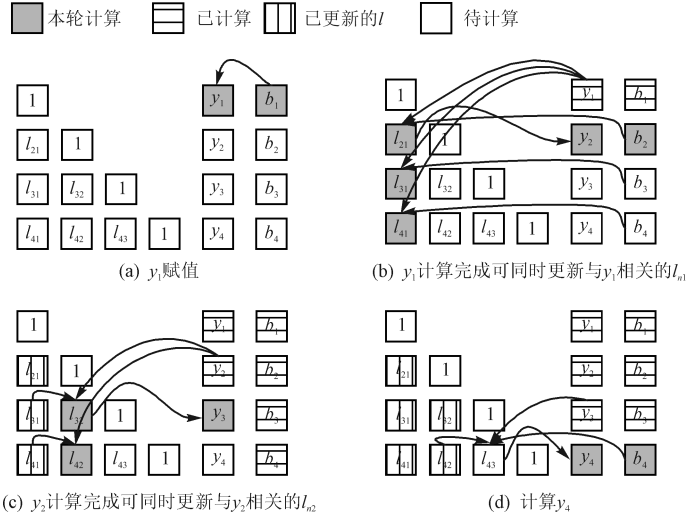


图 1 求解三角矩阵的步骤

Fig. 1 Steps for solving a triangular matrix

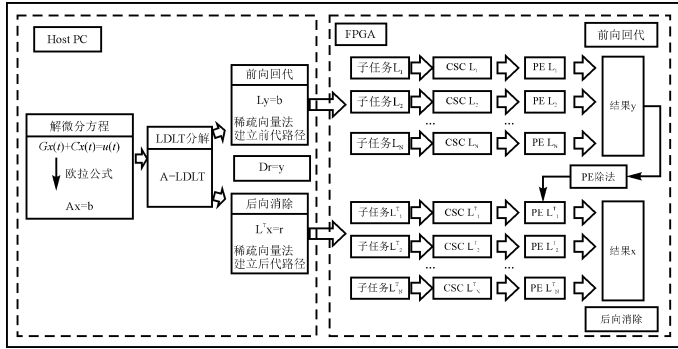


图 2 求解线性矩阵任务划分示意图

Fig. 2 Schematic diagram of task division for solving linear matrices

在 PC 侧求解线性矩阵时,首先通过 LDLT 分解得到单位下三角矩阵 L 和对角矩阵 D ,矩阵数据 L 按列存储在 DRAM 中。再按照 3 个步骤 $Ly=b$ 、 $Dr=y$ 和 $L^T x=r$ 顺序地计算。在 FPGA 部分,基于上述的并行算法,反向回代过程被划分为 p 个子任务,每个子任务都已分配到相应的运算单元。完成前向回代,得到解向量 y 。然后,每个 y 的元素除以相应的对角线元素在除法单元中使用除法运算,求得解向量 r 。最后反向回代,最终得到结果向量。前向回代与后向回代求解在不同阶段进行,因此可以对运算阵列进行分时复用,前后向回代求解过程均利用该运算阵列计算以节省资源。

2.2 硬件架构框架

本文主要关注加速器求解电路,如图 3 所示,在 FPGA 侧的三角矩阵求解加速器模块包括主控制器、存储器控制器、运算阵列,以及除法运算单元。

2.2.1 数据交互

利用 DMA 通过与外部的片外 DRAM 进行矩阵 L 、对

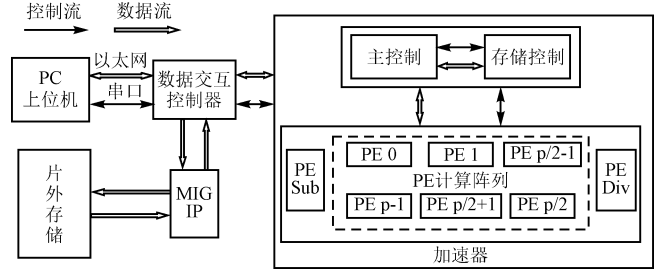


图 3 硬件加速器并行实现结构

Fig. 3 Hardware accelerator parallel implementation structure
角矩阵 D 、向量 b 以及写回解向量 x 数据进行交互;配置信号对加速器求解矩阵所需的参数进行配置。矩阵数据从 PC 上位机下载到片外 DRAM 并使用 DMA 搬运到片上 FIFO 中,而后控制单元读取片上 FIFO 中的数据并分配至各个运算单元的专用 RAM 中。然而,受 RAM 读取速率的限制,多个运算单元无法在单个计算周期内读取全部所需的计算元素,从而限制了加速效率。

2.2.2 运算阵列

在硬件加速器中,运算单元是整个体系结构的计算组件。阵列中所有运算单元的结构都相同,每个运算单元内设有乘法单元、加法与减法单元、存储单元。矩阵回代求解验证框架示意图如图 4 所示,待求解数据从片外传入并存储在片上 FIFO 中。控制单元的状态机根据运算单元数量和矩阵维度进行整体计算架构的数据控制。

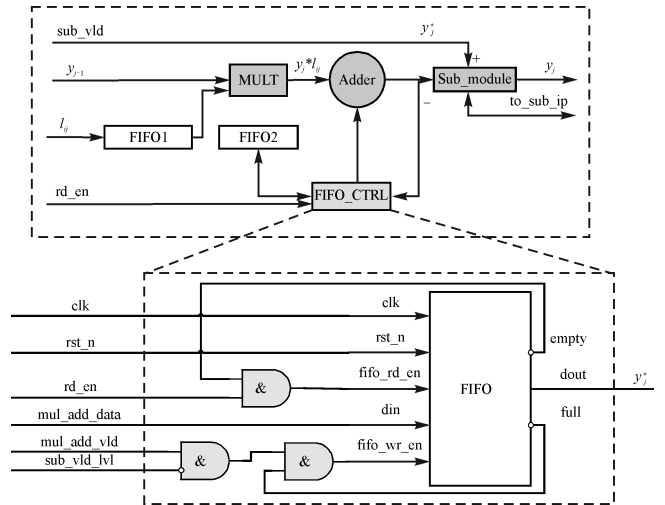


图 4 矩阵回代求解验证框架示意图

Fig. 4 Schematic diagram of matrix backpropagation solution validation framework

在每次计算的开始,上级控制单元读取 FIFO 数据以检索将要计算的元素。计算的中间结果 y_i^* 存储在运算单元的 FIFO 中,以备下一轮使用更新计算结果。一旦队列中的所有元素在当前迭代中全部更新,则开始下一次逐列并行计算的迭代,直到整个计算过程结束。

2.2.3 控制模块

主控制器模块生成控制信号以实现计算、数据检索、减法和除法操作。主控制器模块根据目前运算单元阵列数量 p 和矩阵维度 n 对运算单元阵列进行参数配置以及整个求解过程的计算控制,若一次完整的前向、单位化以及后向回代求解均完成,则接收计算结果解向量 x 并更新电路矩阵;对于单次求解过程中前向、单位化和后向回代过程的切换也由该模块来完成,该模块本质上是一个具有两个层级的状态机。状态机的跳转示意图如图 5 所示。

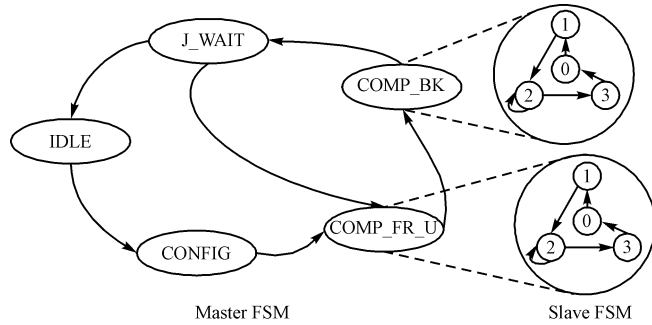


图 5 控制模块的主从两层状态机

Fig. 5 Master slave two-layer state machine of control module

在系统初始化完成后,主状态机会进入 IDLE 状态,等待数据输入至片上存储中。当数据准备完成后,若加速器收到开始配置信号,则进入 CONFIG 参数配置状态,再进行独立的前后向回代,J_WAIT 表示计算完成等待指令。前后向回代分别通过从状态机实现:“0”为空闲状态;“1”表示当前为第一个解向量无需计算直接赋值;“2”表示计算求解状态,在该状态中循环完成列间计算步骤;“3”表示求解完成。

3 实验结果

在本节中,XCZU15EG 的 Zynq UltraScale 系列 FPGA 为实验提供了丰富的硬件资源,它具有 3 528 个 DSP, 26.2 Mb 的块 RAM 和 6 GB 的外部 DRAM 存储,可以满足矩阵数据的存储需求。

本实验以 MATLAB 上求解的结果作为标准。实验所用矩阵均为对称正定、对角占优且稠密度均大于 50%,矩阵维度在 128~8 192 内。在 FPGA 平台上实现了本文提出的并行算法与硬件加速回代架构,并与在 24 核 Intel CPU 平台上使用 MKL 求解器的速度进行对比。

3.1 加速器性能比较

表 1 展示了本文所设计的架构在 200 MHz 频率下使用 8 个运算单元时与在 24 核 Intel CPU 平台上使用 MKL 算法求解不同维度矩阵时的时间与加速比。如表 1 所列,硬件计算架构相比 CPU 的求解加速比随着矩阵维数的增

加而增加。硬件加速平台整体速度远高于使用 CPU 并行计算的速度,算数平均加速比约为 22 倍,最大加速比达到 24 倍。

表 1 不同平台求解矩阵的性能对比

Table 1 Performance comparison of solving matrices on different platforms

矩阵	维度	非零元	时间/ms		加速比	误差
			CPU	FPGA*		
Case1	128	9.4×10^3	0.30	0.02	18.52	2.8×10^{-16}
Case2	512	1.6×10^5	5.62	0.26	21.83	4.6×10^{-16}
Case3	1024	6.2×10^5	23.16	1.03	22.48	9.0×10^{-16}
Case4	2048	2.5×10^6	94.83	4.12	23.00	2.3×10^{-15}
Case5	4096	9.9×10^6	391.32	16.49	23.73	7.7×10^{-15}
Case6	8192	4.0×10^7	1617.27	65.98	24.51	2.8×10^{-14}
算数平均加速比					22.345	

CPU:使用 MKL; FPGA*:使用本文提出的硬件架构

本文提出的计算架构在 FPGA 上求解的误差幅度在 $10^{-17} \sim 10^{-14}$ 。误差可能来源于两个方面:首先是乘累加的运算单元在计算期间的固有误差;其次在前后向的迭代中误差也会逐渐累积。本实验误差远高于 SPICE 电路仿真器的误差要求^[7],上述实验验证了加速器电路的功能正确性,符合标准电路仿真对求解精度的要求。

3.2 带宽限制

在保证结果误差可控的情况下,进行了实验。FPGA 的 DDR 工作频率为 1 200 MHz,因此 32 位宽度和理论最大带宽为 9.6 GB/s。表 2 说明了多个运算阵列求解不同维度矩阵所用的时间。在计算频率为 200 MHz 的条件下,随着运算单元的个数增加,求解速度也随之加快。然而,当运算单元超过 8 个,受数据传输带宽的限制,求解速度不会持续提升。

表 2 多运算阵列求解不同维度矩阵所用的时间

Table 2 Time taken for solving matrices with different dimensions using multiple operational arrays

矩阵维度	运算单元数		
	2	4	≥ 8
128	0.044	0.024	0.016
512	0.659	0.337	0.257
1024	2.611	1.321	1.030
2048	10.397	5.229	4.122
4096	41.493	20.807	16.494
8192	165.779	83.011	65.982

在硬件资源丰富的理想场景下,硬件加速平台的求解速度相较 24 核 Intel CPU 平台上 MKL 求解器的加速比将呈现持续上升的趋势。如图 6 所示,由于带宽因素,求

解加速比被限制在大约 20 倍。运算单元数量增加不会进一步提高求解加速比。

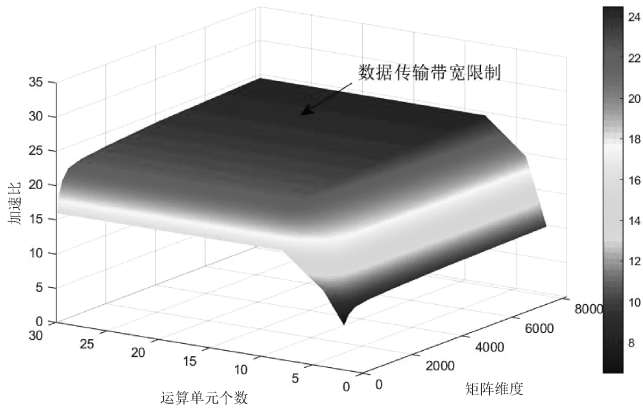


图 6 多运算单元在求解不同维度矩阵时的加速比

Fig. 6 Acceleration ratio of multiple operation units in solving matrices with different dimensions

3.3 相关指标对比

参考文献[8]中实现了基于 3 个运算单元阵列下 96 阶矩阵的 LDLT 分解和回代求解,共需 80 000 个时钟周期,根据该参考文献对两部分求解复杂度分析可相应推断其前后向回代求解时间,与本文提出的回代求解加速器对比如表 3 所列。

表 3 FPGA 平台不同架构相比 24 核 CPU 平台的表现

Table 4 Performance of 24-core CPU platform compared to different architectures on FPGA platform

架构	运算单元数量	最大频率/MHz	加法器数量	时间/ms	加速比
参考文献[8]	3	200.0	6	0.023 7	7.05
本文架构	3	203.6	4	0.017 2	9.44
	8	197.6	31	0.009 0	18.56

参考文献[8]采用了与本文相同的 LDLT 方法实现回代求解加速,在求解矩阵维度为 96、3 个运算单元实现时,本文所提加速器在同样为 3 个运算单元实现时对比该参考文献加速比为 1.38 倍;在部署 8 个运算单元时,加速比为 2.63 倍,在理想情况下存储带宽足够时,对比该参考文献可进一步提升至 4.83 倍,证明了本文所提加速器性能有所提升,以上的加速都是相较于 24 核 Intel CPU 平台的求解表现。本文将运算单元运算所用的减法器提取至运算单元阵列外部,将电路对加法器的数量需求从 $2p$ 个降低至 $p+1$ 个, p 为运算单元的个数,提高了电路资源利用率。

3.4 实验小结

本节针对前文所提出的三角矩阵回代求解加速电路

进行实验和性能分析,首先对前文设计的各个子模块进行功能验证,基于目标 FPGA 平台进行实际测试验证,加速器求解误差精度在 $10^{-17} \sim 10^{-14}$ 内。在目标平台集成多个运算单元对不同阶数矩阵进行加速器电路原型验证,实验结果表明,在存储带宽限制的情况下,与 CPU 平台 Intel 24 核 MKL 并行计算相比,最大加速比达到 24.51 倍。相比同类研究,相同运算单元数量下求解相同维度矩阵时加速比达到 1.38 倍,同时减少了电路资源占用。所提加速器电路能够满足 SPICE 电路仿真下不同维度的矩阵回代求解,为加速电路仿真计算提供了支持,且具有一定工程应用价值。

4 结 语

在本文研究中,各项实验基于 FPGA 的硬件电路。所提出的算法有效避免了求解时的数据依赖,实现了多个运算单元并行。将结果与 CPU 平台的并行算法对比,本文架构求解的计算速度更快。但是,由于硬件平台带宽的局限性,数据的计算速度高于传输速度和求解速度。因此,在主频率固定时,未来实验考虑增加带宽;增加存储器的端口数量;采用存储分体,避免访存冲突。未来将进一步优化硬件加速器的设计,并应用于更加复杂的实际应用场景中。

参考文献

- [1] ZHUOXUAN SHEN, TONG DUAN, VENKATA DINAVAH. Design and Implementation of Real-Time Mpsoc-Based Electromagnetic Transient Emulator of CIGR'E DC Grid for HIL Application[J]. IEEE Power and Energy Technology Systems Journal, 2018, 5(3).
- [2] XIONG X, WANG J. Parallel forward and back substitution for efficient power grid simulation[C]//Proceedings of the International Conference, 2013.
- [3] WANG X, LIU W, WEI X, et al. swSpTRSV: a fast sparse triangular solve with sparse level tile layout on sunway architectures [C]//the 23rd ACM SIGPLAN Symposium. ACM, 2018.
- [4] 刘侃,王欣亮,许平,等. 申威众核处理器上的三对角并行求解器[J]. 计算机科学与探索, 2019, 13(10): 10. DOI: 10.3778/j.issn.1673-9418.1811030.
LIU K, WANG X L, XU P, et al. Tridiagonal parallel solver on Shenwei multi-core processors[J]. Frontiers of Computer Science & Technology, 2019, 13(10): 10. DOI: 10.3778/j.issn.1673-9418.1811030 (in Chinese).
- [5] A DEL RIO RUIZ, K BASTERRETxea. Towards the Development of a CAD Tool for the Implementation of High-Speed Embedded MPCs on FPGAs[C]//2020 European Control Conference (ECC), St. Petersburg, Russia, 2020: 941-

94. doi:10.23919/ECC51009.2020.9143666.
- [6] 吴志勇,王晞阳,陈继林.一种基于FPGA并行加速的稀疏矩阵求解方法[J].电力系统保护与控制,2021,49(11):155-162. DOI:10.19783/j.cnki.pspc.200948.
- WU ZH Y, WANG X Y, CHEN J L. A Sparse Matrix Solution Method Based on FPGA Parallel Acceleration[J]. Power System Protection and Control, 2021, 49(11): 155 - 162. DOI:10.19783/j.cnki.pspc.200948 (in Chinese).
- [7] STANFORD UNIVERSITY. SPICE Reference[EB/OL]. [2023-11-15]. https://web.stanford.edu/class/ee133/handouts/general/spice_ref.pdf.
- [8] SUN R, LIU P, XUE J, et al. Bax: A bundle adjustment accelerator with decoupled access/Execute Architecture for Visual Odometry[J]. IEEE Access, 2020(8): 75530 - 75542.

(责任编辑:薛士然 收稿日期:2023-11-15)